COLING

International Conference on Computational Linguistics

Proceedings of the Conference and Workshops

COLING

Volume 29 (2022), No. 1

Proceedings of the Main Conference

The 29th International Conference on Computational Linguistics

October 12 - 17, 2022
Gyeongju, Republic of Korea

GRAND CHALLENGE

NAVER | LINE KiSTi www.kisti.re.kr 한국과학기술정보연구원 Korea Institute of Science and Technology Information

PLATINUM

Saltlux Baidu 百度 42dot

GOLD

NC LG Electronics AIRC

42MARU Google Samsung Research

kt amazon | science DATAFORCE BY TRANSPERFECT Flitto

SILVER

kakaobrain synapsoft

kakaoenterprise Ministry of Culture, Sports and Tourism National Institute of Korean Language

BRONZE

BeringLab 언어정보연구원 Institute of Language and Information Studies Imagine Is Real IIR TECH (주)이르테크

ETRI 한국전자통신연구원 Electronics and Telecommunications Research Institute 小牛翻译 NiuTrans.com VAIV teddysum

SUPPORTERS

lexicala BY K DICTIONARIES diqode

Copyright of each paper stays with the respective authors (or their employers).

ISSN 2951-2093

Preface: General Chairs

Welcome to COLING 2022 – the 29th International Conference on Computational Linguistics. Held in Gyeongju, this is the first COLING in the Republic of Korea!

We visited Gyeongju in 2016, as a site visit together with the local chair Key-Sun Choi, and were amazed by its beauty and its great historical significance. Our report to the ICCL was extremely positive. So here we are, delighted to be General Chairs of COLING in the beautiful capital of the old Silla Kingdom.

COLING is organized under the auspices of the International Committee on Computational Linguistics (ICCL, https://ufal.mff.cuni.cz/iccl). The ICCL is a very special committee, with neither bylaws nor bank accounts. The sole function of the ICCL is to ensure that a COLING is held every two years and that the conference is not only scientifically robust but also conducive to the sharing of ideas and cultural experiences in a congenial and inclusive environment.

COLING has evolved over the years, together with the changes in our field. But the mission of the ICCL to maintain the COLING "spirit" has never changed: we want COLING to be an inclusive conference that welcomes diversified participants and ideas. We also want to underline the fact that Language is what defines our field and the subject of our scientific inquiries. Thus, we pay special attention to works that help us understand Language, including its complexities, diversity, and robust reflection and facilitation of individual and collective human behaviors and actions. This is why the theme of COLING 2022 is ***NLP for the Grand Challenges of Our Time***. We would like to highlight that, through effective processing of language big data, computational linguistics will play a crucial role in understanding the nature of the grand challenges, how people react to these challenges, and how to manage effective collective behaviors to tackle these challenges.

Recall that for COLING the congenial and inclusive environment for exchanges of ideas is part of the gene of the conference that is as important as its scientific excellence. That's why COLING has kept the tradition of the "excursion" that typically allows participants to be immersed in a new cultural or ecological environment.

A highly efficient committed team has worked for the organization of COLING 2022, having also to cope with the unpredictable consequences of Covid-19 and related policies around the world. We are having a hybrid COLING because of the still present difficulties to travel from or back to some parts of the world. This decision had a strong impact on the organization of the conference program, both for the main conference and workshops and tutorials.

It is our pleasure to warmly thank, also on behalf of the ICCL, all the Chairs, too many to mention them all here, for the wonderful work they have done. It has been a pleasure and a privilege for us to work together with such a competent and talented team: they made our work as General chairs very easy.

We do owe a special thanks to Hansaem Kim, James Pustejovsky and Leo Wanner, the Program chairs, for their hard work in managing so smoothly an impressive number of submissions, many more than we expected. And we wish to express our deepest gratitude to the Local chairs – Key-Sun Choi and Pum-Mo Ryu – who have worked with such great dedication in all the various phases of the conference organization, always keeping everything under control. Not an easy task in particular during the pandemic time. We enjoyed the weekly calls with the Program chairs and the Local chair: they have been very useful for an efficient interaction among all of us.

We also want to thank the generosity of all the sponsors for their great support of COLING 2022, especially our grand challenge sponsors.

Last but not least, our heartfelt thanks go to all the colleagues (so many) who submitted their work to COLING 2022, to the organizers of Workshops and Tutorials, to the participants (both in-person and

remote), and to the many students among them. The participation of young researchers keeps alive the continuous growth of our community in COLING, and our joint efforts to tackle the grand challenges.

We hope that you benefit not only from the scientific program but also, for both the in-person and virtual participants, from the social parts of COLING. We hope you get from this COLING both new exciting ideas and also new friends.

Enjoy COLING 2022 in Gyeongju!

Nicoletta Calzolari and Chu-Ren Huang
General Chairs of COLING 2022 and ICCL Members

Preface: Program Chairs

Welcome to the 29th International Conference on Computational Linguistics (COLING'2022)!

The past two years have been a very difficult time for conference travel, due to the continued effect of global travel restrictions caused by the COVID-19 pandemic. And while some restrictions are still being enforced, we are nevertheless excited to be able to offer COLING 2022 as a combination in-person and hybrid event in the beautiful historical city of Gyeongju in the Republic of Korea.

COLING 2022 received 2253 submissions, 2026 of which were reviewed. From these, 632 regular papers have been accepted, 83% for long papers, and 17% for short papers. We are gratified to note the increasing interest and enthusiasm that researchers in our community have shown for COLING. We would like to thank all authors who submitted to the conference, and congratulate those whose papers were accepted for conference presentation. We also hope that those authors whose papers were not accepted this year can profit from reviews and feedback, and we hope to see you in 2024!

We would particularly like to thank the amazing pool of 1935 reviewers, who made the conference possible, with their time devoted to such careful assessment of the papers assigned to them. Much of the heavy lifting in the review process fell on the 44 Senior Area Chairs, and we are extremely grateful to them for their efforts and time in assigning and negotiating the reviews. We would also like to express our thanks to various members of the organizing committee, who have helped out in many different ways, at all the various stages in planning, organization, and implementation of the conference: the General Co-Chairs, Nicoletta Calzolari (The National Research Council, Italy), Chu-Ren Huang (Hong Kong Polytechnic Univ.,HongKong);

Local Organization Co-Chairs: Key-Sun Choi (KAIST, Korea), Pum-Mo Ryu (Busan Univ. of Foreign Studies, Korea). Workshop Chairs: Sadao Kurohashi (Kyoto Univ., Japan), Patrizia Paggio (University of Copenhagen, Denmark, L'Universitata Malta, Malta), Nianwen Xue (Brandeis University, USA); Tutorial Chairs: Hsin-Hsi Chen (National Taiwan Univ., Taiwan), Lucia Donatelli (Saarland Univ., Germany), Heng Ji (Univ. of Illinois at Urbana-Champaign, USA); Sponsorship

Co-Chairs: Zhong He (Baidu, China), Tony Kyungil Lee (Saltlux, Korea), Enrico Santus (Bayer-Global, Germany); Web Chair: Sung-Pil Choi (Kyonggi Univ., Korea); Virtual Infrastructure Co-Chairs: Seokhwan Kim (Amazon Alexa AI, USA), Younggyun Hahm (teddysum, Korea); Virtual Social Chair: Chanjun Park (Upstage, Korea Univ., Korea); Ethics Board: Nikhil Krishnaswamy (Colorado State Univ., USA), Constantine Lignos (Brandeis Univ., USA), Donia Scott (Univ. of Sussex Falmer, UK); Publicity Chair: Huyen Nguyen Thi Minh (Vietnam National Univ., Vietnam); Publication Co-Chairs: Francis Bond (Palacký Univ. Olomouc, Czechia), Seung-Hoon Na (Jeonbuk National Univ., Korea); Diversity Co-Chairs: Lori Levin (Carnegie Mellon Univ., USA), Rachel Roxas (National Univ., Philippines); Handbook Chair: Eun-kyung Kim (Daejeon Univ., Korea); Industry Session Chair: Yuseop Kim (Hallym Univ., Korea); Student Volunteer Chair: Jin Young Yeo (Yonsei Univ., Korea); Poster Session Chair: Hyun-Je Song (Jeonbuk Nat'l Univ., Korea).

We are particularly excited and grateful to the three keynote speakers for this year's conference: Martha Palmer from University of Colorado Boulder, USA; Marco Baroni from ICREA and Pompeu Fabra University, Spain; and Kentaro Inui from Tohoku University and RIKEN, Japan. This year, COLING, as part of its theme, is addressing the seven societal Grand challenges: Health, demographic change and wellbeing; Food security, sustainable agriculture and forestry, marine and maritime and inland water research, and the bioeconomy; Secure, clean and efficient energy; Smart, green and integrated transport; Climate action, environment, resource efficiency and raw materials; Changing world – inclusive, innovative and reflective societies; Secure societies – protecting freedom and security of the world. To this end we have two Grand Challenge invited speakers: Ping Li from The Hong Kong Polytechnic University, Hong Kong, and Helen Meng from The Chinese University of Hong Kong. In addition, we

have organized five paper sessions focusing on one of these grand challenge themes.

To all of those mentioned above, we again express our deepest thanks for helping in so many different ways, to make COLING 2022 a memorable event, and a return to an in-person conferencing experience. We hope you enjoy the conference.

Hansaem Kim
James Pustejovsky
Leo Wanner
COLING 2022 Program Committee Co-Chairs

Preface: Chair of the ICCL

This conference is the 29th in the COLING series. The first COLING took place in New York in 1965, and since the 6th COLING in Ottawa (1976), the conference has been held regularly once every two years. The initial phase of COLING established computational linguistics as an academic discipline and subsequently, during its long history, the conference series has broadened its scope to become truly interdisciplinary, allowing researchers with interests in diverse aspects of language and its processing to get together and exchange their insights and ideas.

COLING provides participants with a venue in which to learn about the most recent and prominent research achievements in broad areas of computational linguistics and natural language processing. More importantly, through interaction with researchers from different disciplines and with different perspectives, participants form new insights/ideas and gain long-term research perspectives. While a divergence of science and engineering in our field is becoming increasingly prominent(1), the ICCL (International Committee of Computational Linguistics), under the auspices of which the COLING series is organized, believes that interaction among researchers of different disciplines is more important than ever.

As an informal, voluntary-based committee, the ICCL relies heavily on the ingenuities of local hosts, who have organized their COLING conferences with unique attributes. The format and characteristics of COLING have evolved thanks to the bottom-up initiatives of local hosts and program chairs, and through their interaction with the ICCL. This is particularly true for this year's COLING: due to the ever-changing situation around the pandemic and the rapid expansion of our field, we have had to make difficult decisions and to improvise solutions to allow for unpredictable events. We owe a great deal to the ingenuity and flexibility of our colleagues in Korea. On behalf of the ICCL, I would like to express our sincere gratitude to the local organizing committee and to the chair, Professor Key-Sun Choi.

I would also like to extend my gratitude to the two members of ICCL who take on the role of General Co-Chairs: Professors Nicoletta Calzolari and Chu-Ren Huang. Without them, it would be impossible to coordinate the many committees and people involved in this conference.

Thanks to the tremendous efforts of all those involved, I am convinced that we will experience a fantastic conference, which will be both scientifically exciting and full of fond memories.

I extend a warm welcome to all of you, and hope that you enjoy the conference!

Junichi Tsujii, Chair of the ICCL

(1) Tsujii, Junichi: Natural language Processing and Computational Linguistics, Computational Linguistics (2021) 47 (4): 707–727. https://direct.mit.edu/coli/article/47/4/707/107177/Natural-Language-Processing-and-Computational

---

ICCL : Aravind Joshi, Petr Sgall, Makoto Nagao and Martin Kay

The ICCL (International Committee of Computational Linguistics) was formed with the aims of running the COLING conference series and of maintaining the continuity of its spirits, e.g., pioneering new research fields, facilitating interaction among participants with diverse interests, respecting the equality of all languages, recognizing the potential value of non-mainstream research, etc.

To reflect the interdisciplinarity which we believe is crucial for COLING, the ICCL is comprised of members from diverse fields, e.g., theoretical linguistics, corpus linguistics, artificial intelligence, cognitive science, machine translation, language engineering, etc. (1);

Although ICCL membership is for life, the long history of the committee inevitably means that we have lost some of its members over time. In particular, recent years have witnessed the loss of four highly respected members: Aravind Joshi (2017), Petr Sgall (2019), Makoto Nagao (2021) and finally Martin Kay (2021), who had been the longest serving Chair of the ICCL when he stepped down in 2014.

All four of these venerable individuals showed strong intellectual leadership and inspired many researchers to follow them in forming their own schools of thoughts. They enriched the fields of computational linguistics and natural language processing by exploring new ideas and establishing new research areas: tree adjoining grammar, cognitive models, discourse annotation (Aravind Joshi[(2)]); functional generative description, functional view of Prague school of linguistics (Petr Sgall[(3)]); example-based machine translation, language engineering, digital libraries (Makoto Nagao[(4)]); parsing, feature-based grammar formalisms, man-machine cooperation in translation (Martin Kay[(5)]). Current members of the ICCL inherit not only their research achievements, but also their valuable insights into language and their collective conceptions about the directions that our research fields should take. We, the members of the ICCL, are ready to carry the torch that they have handed to us.

Junichi Tsujii, Chair of the ICCL

(1) Members of the ICCL: `https://ufal.mff.cuni.cz/iccl/committee-members`

(2) Aravind Joshi: `https://almanac.upenn.edu/articles/aravind-joshi-engineering`

(3) Petr Sgall: `https://ufal.mff.cuni.cz/petr-sgall`

(4) Makoto Nagao: `https://nagao-memorial.kuee.kyoto-u.ac.jp/en/`

(5) Martin Kay: `https://direct.mit.edu/coli/article/48/1/1/108842/Obituary-Martin-Kay`

Preface: Local Chairs

We are happy to be able to hold the conference at a time when the Covid-19 situation has subsided and the freedom to enter and leave Korea is increasing without PCR test. Since the organizing committee has held weekly meetings since last year's August, the leaves of trees change from summer to autumn, winter, spring, summer, and autumn again, and now they are turning into beautiful autumn leaves.

You are in Gyeongju, the capital of the 1000-year-old Kingdom of Shilla[1] (BC57-AD935). In Silla, the Idu script[2] was used, and it is said to be a phonetic symbol derived from Chinese characters. The current Korean alphabet Hangul is a character set invented based on linguistics. For example, the letter corresponding to 'g' is 'ㄱ', which describes the shape of the tongue in the mouth when the sound 'g' is made. Each letter was made according to the principle and shape of the vocal organs, so that Hangul was created to be able to use all human pronunciations as a phonetic symbol.

This COLING 2022 begins in the week of Hangul Day. The 9th of October is a national holiday to commemorate the creation and proclamation of Hangul[3] (AD 1446) in the Basic Law of the Korean Language.

The number of Korean speakers is about 77.3 million, which ranks 14th among all languages in the world, but the proportion is only 1.004%. Although the amount of Korean corpora is overwhelmingly smaller in quantity and diversity compared to English data, the data for natural language processing in Korean language is increasing, centering on the National Institute of the Korean Language[4]. It is open to researchers.

Although there were many difficulties in organizing the conference due to the Covid-19 Pandemic, we are very happy that many papers have been submitted and that we have been able to support the presentation. We would like to thank those who come in person even from the Covid-19 situation, and we are trying to do our best for those who participate online.

In particular, we would like to thank Nicoletta Calzorari and Chu-Ren Huang for their active support, who conducted on-site due diligence for this COLING to be held in the year 2018 and also served as the general chairs. We would like to thank Hansaem Kim, Leo Wanner and James Pustejovsky, program chairs. Since the beginning of September last year, the General co-chairs and the Program co-chairs have held weekly meetings every Thursday, thanking you for designing and implementing every step of the way, including home page, sponsoring, CFP, grand-challenge theme setting, reviewing and all of the matters. Hansaem Kim, as a program co-chair and as a key executive of the local organizer, put all her heart into every detail.

The local organizing committee has held weekly meetings every Tuesday since last August. Attended weekly meetings of this local organization we would like to express deep thanks to Hansaem Kim, who was in charge of both the Program chair and local organizing, Industry session chair Yuseop Kim, Virtual social chair Chanjun Park, Sponsoring chair Tony Kyungil Lee, Virtual Platform chairs Seokhwan Kim and Young-gyun Hahm, Web chair Sung-pil Choi, Publication chairs Francis Bond and Seung-Hoon Na, Handbook chair Eun-Kyung Kim, Poster session chair Hyun-Je Song and all the members.

First of all, I would like to thank Dr. Jae-Gil Choi of HICO for encouraging and supporting the Korean hosting of COLING 2022. Also, I would like to thank Genicom for preparing the conference and Underline for the online conference.

This conference would not have been possible without the support of sponsors: Naver, Line, KISTI,

[1] https://en.wikipedia.org/wiki/Silla
[2] https://en.wikipedia.org/wiki/Idu_script
[3] https://en.wikipedia.org/wiki/Hangul
[4] https://corpus.korean.go.kr/

and all of the industries, institutes, universities, and government. We would like to express our deepest gratitude to the Korean Institute of Information Scientists and Engineers[5] for their support as the conference host, and to Gyeongju City for their full and attractive support.

Key-Sun Choi and Pum-Mo Ryu
Local Organization Chairs of COLING 2022

[5] https://www.kiise.or.kr/

Organizers

General Co–Chairs

Nicoletta Calzolari, The National Research Council, Italy
Chu-Ren Huang, Hong Kong Polytechnic University, Hong Kong

Program Co–Chairs

Hansaem Kim, Yonsei University, Korea
James Pustejovsky, Brandeis University, USA
Leo Wanner, ICREA and Pompeu Fabra University, Spain

Local Organization Co–Chairs

Key–Sun Choi, KAIST, Korea
Pum–Mo Ryu, Busan University of Foreign Studies, Korea

Tutorials Co–Chairs

Hsin-Hsi Chen, National Taiwan University, Taiwan
Lucia Donatelli, Saarland University, Germany
Heng Ji, University of Illinois at Urbana–Champaign, USA

Workshops Co–Chairs

Sadao Kurohashi, Kyoto University, Japan
Patrizia Paggio, University of Copenhagen, Denmark; L-Università ta' Malta, Malta
Nianwen Xue, Brandeis University, USA

Virtual Infrastructure Co–Chairs

Seokhwan Kim, Amazon Alexa AI, USA
Younggyun Hahm, teddysum, Korea

Sponsorship Co–Chairs

Zhong He, Baidu, China
Tony Kyungil Lee, Saltlux, Korea
Enrico Santus, Bayer-Global, Germany

Web Chair

Sung-Pil Choi, Kyonggi University, Korea

Virtual Social Chair

Chanjun Park, Upstage, Korea University, Korea

Ethics Board

Nikhil Krishnaswamy, Colorado State University, USA
Constantine Lignos, Brandeis University, USA
Donia Scott, University of Sussex Falmer, UK

Publicity Chair

Huyen Nguyen Thi Minh, Vietnam National University, Vietnam

Publication Co–Chairs

Francis Bond, Palacký Univ. Olomouc, Czechia
Seung-Hoon Na, Jeonbuk National University, Korea

Diversity Co–Chairs

Lori Levin, Carnegie Mellon University, USA
Rachel Roxas, National University, Philippines

Handbook Chair

Eun-kyung Kim , Daejeon University, Korea

Industry Session Chair

Yuseop Kim, Hallym University, Korea

Student Volunteer Chair

Jin Young Yeo, Yonsei University, Korea

Poster Session Chair

Hyun-Je Song, Jeonbuk National University, Korea

Senior Area Chair

Dirk Heylen, University of Twente
Maria Wolters, University of Edinbur
Thierry Declerck, DFKI GmbH
Björn Schuller, University of Augsburg / Imperial College London
Kemal Oflazer, Carnegie Mellon University in Qatar
Yue Zhang, Westlake University
Sabine Schulte im Walde, University of Stuttgart
Constantine Lignos, Brandeis University
Owen Rambow, Stony Brook University
Wolfgang Minker, Ulm University
Eva Hajicova, Charles University
Vincent Ng, University of Texas at Dallas
Nancy Ide, Vassar College / Brandeis University
Jimmy Lin, University of Waterloo
Donia Scott, University of Sussex
Sanda Harabagiu, University of Texas at Dallas
Mohit Iyyer, University of Massachusetts Amherst
David McDonald, Smart Information Flow Technologies (dba SIFT, LLC)
Sujian Li, Peking University
Yiqun Liu, Tsinghua University
Kang Liu, Institute of Automation, Chinese Academy of Sciences
Christian Boitet, Université Grenoble-Alpes (UGA, LIG-GETALP)
Rodrigo Agerri, HiTZ Center - Ixa, University of the Basque Country UPV/EHU
Satoshi Sekine, Riken, AIP
Jung-Woo Ha, NAVER CLOVA AI Lab

Xiaodong Liu, Microsoft Research
Serge Sharoff, University of Leeds
Alessandro Lenci, University of Pisa
Miriam Butt, University of Konstanz
Xavier Carreras, dMetrics
Jun Suzuki, Tohoku University / RIKEN Center for AIP
Philipp Koehn, Johns Hopkins University
Josef van Genabith, DFKI
Wanxiang Che, Harbin Institute of Technology
Ryan Cotterell, ETH Zürich
Jeffrey Flanigan, UC Santa Cruz
Simon Dobnik, University of Gothenburg
Matthew Stone, Rutgers University
Claire Gardent, CNRS/LORIA
Preslav Nakov, Mohamed bin Zayed University of Artificial Intelligence
Paolo Rosso, Universitat Politècnica de València
Erik Cambria, Nanyang Technological University
Tommaso Caselli, Rijksuniversiteit Groningen
Shuai Wang, Amazon AI
Helen Meng, The Chinese University of Hong Kong

Area Chairs

Björn Schuller, University of Augsburg / Imperial College London
Thierry Declerck, DFKI GmbH
Dirk Heylen, University of Twente
Maria Wolters, University of Edinburgh
Gina-Anne Levow, University of Washington
Florian Metze, Carnegie Mellon University
Xixin Wu, The Chinese University of Hong Kong
Pengfei Liu, Centre for Perceptual and Interactive Intelligence
Pierpaolo Basile, Department of Computer Science, University of Bari Aldo Moro
Valerio Basile, University of Turin
Davide Buscaldi, LIPN, Université Paris 13
Elena Cabrio, Université Côte d'Azur, Inria, CNRS, I3S
Roman Klinger, University of Stuttgart
Els Lefever, LT3, Ghent University
Qian Liu, Nanyang Technological University
Nikola Ljubešić, Jožef Stefan Institute
Rui Mao, Nanyang Technological University
Constantin Orăsan, University of Surrey
Viviana Patti, University of Turin, Dipartimento di Informatica
Ruifeng Xu, Harbin Institute of Technology, Shenzhen
Tanmoy Chakraborty, Indraprastha Institute of Information Technology Delhi (IIIT-D), India
Elisabetta Fersini, University of Milano-Bicocca
Zeerak Talat, Simon Fraser University
Sara Tonelli, FBK
Vasudeva Varma, IIIT Hyderabad
Bertie Vidgen, Alan Turing Institute
Arkaitz Zubiaga, Queen Mary University of London
Guangsheng Bao, Westlake University

Deng Cai, The Chinese University of Hong Kong
Thiago Castro Ferreira, Federal University of Minas Gerais
Vera Demberg, Saarland University
Ondřej Dušek, Charles University
Angela Fan, Facebook AI Research
Albert Gatt, Utrecht University
Dimitra Gkatzia, Edinburgh Napier University
Fei Liu, University of Central Florida
Yang Liu, Microsoft
Qi Liu, The University of Hong Kong
Jiangming Liu, Tencent
Shashi Narayan, Google
Leonardo F. R. Ribeiro, TU Darmstadt
Anastasia Shimorina, Orange
Jinsong Su, Xiamen University
Jiajun Zhang, Institute of Automation Chinese Academy of Sciences
Stella Frank, Unversity of Copenhagen
Casey Kennington, Boise State University
Parisa Kordjamshidi, Michigan State University
Nikhil Krishnaswamy, Colorado State University
Jie Lei, Meta Platforms, Inc
Carina Silberer, University of Stuttgart
Jesse Thomason, University of Southern California
Diedrich Wolter, University of Bamberg
Jiang Guo, Massachusetts Institute of Technology
Yijia Liu, Bytedance Inc.
Yuxuan Wang, Zhejiang Lab
Meishan Zhang, Harbin Institute of Technology (Shenzhen), China
Cristina España-Bonet, DFKI GmbH
Yvette Graham, ADAPT, Trinity College Dublin
Julia Kreutzer, Google
Qun Liu, Huawei Noah's Ark Lab
Chi-kiu Lo, National Research Council of Canada
Kenton Murray, Johns Hopkins University
Juan Pino, Facebook
Dipti Sharma, IIIT, Hyderabad
Taro Watanabe, Nara Institute of Science and Technology
Wilker Aziz, University of Amsterdam
Dan Goldwasser, Purdue University
Gholamreza Haffari, Monash University
Katsuhiko Hayashi, Hokkaido University
Sosuke Kobayashi, Preferred Networks / Tohoku University
Lingpeng Kong, The University of Hong Kong
Joseph Le Roux, Université Sorbonne Paris Nord
Pranava Madhyastha, City, University of London
Tetsuji Nakagawa, Google Japan G.K.
Stefan Riezler, Heidelberg University
Sho Takase, Tokyo Institute of Technology
Duygu Ataman, New York University
Kalika Bali, Microsoft Research Labs
Harald Hammarström, Uppsala University

Kengatharaiyer Sarveswaran, University of Jaffna
Alexandros Tantos, Aristotle University of Thessaloniki
Ashwini Vaidya, Indian Institute of Technology Delhi
Menno van Zaanen, South African Centre for Digital Language Resources
Gözde Şahin, UKP Lab, Technische Universität Darmstadt
Chris Biemann, Universität Hamburg
Gemma Boleda, Universitat Pompeu Fabra / ICREA
Emmanuele Chersoni, Hong Kong Polytechnic University
Paul Cook, University of New Brunswick
Diego Frassinelli, University of Konstanz
Simonetta Montemagni, Istituto di Linguistica Computazionale "Antonio Zampolli"
Thierry Poibeau, LATTICE (CNRS & ENS/PSL)
Agata Savary, Paris-Saclay University
Tim Van de Cruys, University of Leuven
Ivan Vulić, University of Cambridge
Sina Ahmadi, National University of Ireland Galway
Sara Carvalho, NOVA CLUNL – Faculty of Social Sciences and Humanities – NOVA University Lisbon | CLLC - University of Aveiro | School of Technology and Management – University of Aveiro
Anca Dinu, University of Bucharest
Miloš Jakubíček, Lexical Computing
Anisia Katinskaia, University of Helsinki
Penny Labropoulou, ILSP / Athena RC
Ekaterina Lapshinova-Koltunski, Universität des Saarlandes
Paul Rayson, Lancaster University
Claudia Soria, Consiglio Nazionale delle Ricerche - Istituto di Linguistica Computazionale “A. Zampolli”
Marko Tadić, University of Zagreb, Faculty of Humanities and Social Sciences
George Tambouratzis, ILSP/Athena R.C.
Seid Muhie Yimam, Universität Hamburg
Marcos Zampieri, Rochester Institute of Technology
Hao Cheng, Microsoft Research
Manuel Ciosici, University of Southern California – Information Sciences Institute
Matthias Gallé, Naver Labs E
Junjie Hu, University of Wisconsin-Madison
Yoon Kim, MIT
Liyuan Liu, Microsoft Research
Colin Raffel, University of North Carolina/Hugging Face
Minjoon Seo, KAIST
Alessandro Sordoni, Microsoft Research
Kang Min Yoo, NAVER AI Lab
Gorka Azkune, University of Basque Country
Jeremy Barnes, University of the Basque Country EHU/UPV
Eduardo Blanco, The University of Arizona
Chenhui Chu, Kyoto University
Takashi Inui, University of Tsukuba
Egoitz Laparra, University of Arizona
Roser Morante, VU Amsterdam
Yugo Murawaki, Kyoto University
Alessio Palmero Aprosio, Fondazione Bruno Kessler
Paolo Torroni, Alma Mater - Università di Bologna
Yongfeng Zhang, Rutgers University
Khalid Al Khatib, Groningen University

Yanan Cao, Institute of Information Engineering, Chinese Academy of Science
Yubo Chen, Institute of Automation, Chinese Academy of Sciences
Wenliang Chen, Soochow University
Nicolas Dugué, LIUM
Tao Gui, Fudan University
Xianpei Han, Institute of Software, Chinese Academy of Sciences
Jialong Han, Amazon
Lifu Huang, Virginia Tech
Evangelos Kanoulas, University of Amsterdam
Chenliang Li, Wuhan University
Peng Li, Institute for AI Industry Research (AIR), Tsinghua University, China
Fei Li, Wuhan University
Tingwen Liu, Institute of Information Engineering, Chinese Academy of Sciences
Ming Liu, HIT
Thien Nguyen, University of Oregon
Dheeraj Rajagopal, Carnegie Mellon University
David Reymond, Université de Toulon
Alan Ritter, Georgia Institute of Technology
Mathieu Roche, CIRAD, TETIS
Chengjie Sun, Harbin Institute of Technology
Yannick Toussaint, LORIA, Université de Lorraine
Tong Xu, University of Science and Technology of China
Rui Yan, Renmin University of China
Jie Yang, Zhejiang University
Min Yang, Chinese Academy of Sciences
Muyun Yang, Harbin Institute of Technology
Daojian Zeng, Hunan Normal University
Qi Zhang, Fudan University
Richong Zhang, Beihang University
Peng Zhang, Tianjin University
Wayne Xin Zhao, RUC
Sewon Min, University of Washington
Guenter Neumann, DFKI & Saarland University
Axel-Cyrille Ngonga Ngomo, Paderborn University
Jianmo Ni, Google
Panupong Pasupat, Google
Christian Retoré, University of Montpellier
Pedro Rodriguez, Meta FAIR
Mrinmaya Sachan, ETH Zurich
Keh-Yih Su, Institute of Information Science, Academia Sinica
Ping Wang, Stevens Institute of Technology
Gijs Wijnholds, Utrecht University
Caiming Xiong, Salesforce
Steven Bird, Charles Darwin University
Walter Daelemans, University of Antwerp, CLiPS
Karën Fort, Sorbonne Université and LORIA
Jin-Dong Kim, Database Center for Life Science
Kevin Duh, Johns Hopkins University
Claudia Hauff, Delft University of Technology
Robert Krovetz, Lexical Research
Joel Mackenzie, The University of Queensland

Alexander Mehler, Goethe-University Frankfurt am Main
Antonio Rinaldi, University of Napoli Federico II
Lucy Lu Wang, Allen Institute for AI
Shafiq Joty, Nanyang Technological University; Salesforce AI Research
Massimo Poesio, Queen Mary University of London
Hannah Rashkin, Google Research
Manfred Stede, University of Potsdam
Bonnie Webber, University of Edinburgh
Ingo Siegert, Otto von Guericke University Magdeburg
Zoraida Callejas, University of Granada
Patrick Gebhard, DFKI
Stefan Hillmann, Quality and Usability Lab, Technische Universität Berlin
Kristiina Jokinen, AIRC, AIST
Michael McTear, Ulster University
Catherine Pelachaud, CNRS-ISIR, Sorbonne Université
Stefan Schaffer, DFKI GmbH
Stefan Ultes, Mercedes-Benz AG
Fei Xia, University of Washington
Lisa Beinborn, Vrije Universiteit Amsterdam
Stefan L. Frank, Radboud University
Kyle Gorman, The Graduate Center, City University of New York
Jordan Kodner, Stony Brook University
Jackson Lee, Independent Researcher
Stephanie M. Lukin, U.S. Army Research Laboratory
Mark-Jan Nederhof, University of St Andrews
Yohei Oseki, University of Tokyo
Jonathan Rawski, San Jose State University
H. Andrew Schwartz, Stony Brook University
Edward Stabler, UCLA
Michael White, The Ohio State University
Zhongqing Wang, Soochow University
Deyi Xiong, Tianjin University
Xiang Lorraine Li, University of Massachusetts Amherst
Torsten Zesch, Computational Linguistics, FernUniversität in Hagen
Jeff Heflin, Lehigh University

Reviewers

Jun Suzuki, Christian Boitet, Frederic Bechet, Qian Chen, Hsin-Min Wang, Lucien Carroll, Guillermo Cámbara, Juan-María Garrido, Shujie Liu, Steven Moran, Trung Hieu Nguyen, Mohamed Lichouri, Bin Liang, Dong Zhang, Zhiyang Teng, Xin Zhang, Lucia Passaro, Bowen Zhang, Alexandra Balahur, Momchil Hardalov, Xiabing Zhou, Gavin Abercrombie, Abhishek Agarwal, Oscar Araque, Steve Azzolin, Jorge Baptista, Francesco Barbieri, Somnath Basu Roy Chowdhury, Fernando Batista, Adrian M.P. Brasoveanu, Henrico Brum, Jiahua Chen, Xinhong Chen, Zifeng Cheng, Alessandra Teresa Cignarella, Alexandra Ciobotaru, Davide Colla, Géraldine Damnati, Orphee De Clercq, Loic De Langhe, Lingjia Deng, Asif Ekbal, Mai ElSherief, Aline Etienne, Chuang Fan, Liri Fang, Javier Fernandez-Cruz, Flavius Frasincar, Dan Friedman, Aitor García Pablos, Mengshi Ge, Voula Giouli, Nathan Green, Anne Göhring, Thomas Haider, Kishaloy Halder, Kai He, Mohammed Moshiul Hoque, Wenjun Hou, Qinmin Vivian Hu, Yongfeng Huang, Ming Huang, Tin Huynh, Donghong Ji, Tianyu Jiang, Omkar Joshi, Hiroshi Kanayama, Diptesh Kanojia, Zixuan Ke, Roman Kern, Gyuhak Kim, Anna Koufakou, Taja Kuzman, Sofie Labat, Huiyuan Lai, Mirko Lai, Mateusz Lango, Ji-Ung Lee, Ruifan Li, Wei Li, Dongyang Li, Zaijing Li, Tao Li,

Yongqi Li, Yunlong Liang, Qika Lin, Bing Liu, Chenyang Lyu, Yukun Ma, Nianzu Ma, Mateus Machado, Isa Maks, Aaron Maladry, Ilia Markov, Santiago Marro, Sahisnu Mazumder, Gosse Minnema, Karo Moilanen, Negar Mokhberian, Antonio Moreno-Ortiz, Rajdeep Mukherjee, Usman Naseem, Kiet Nguyen, Duc-Vu Nguyen, Pin Ni, Malvina Nissim, Shramay Palta, Akshata Phadte, Flor Miriam Plaza-del-Arco, Daniel Preotiuc-Pietro, Pavel Přibáň, Jiexing Qi, Shenbin Qian, Daniele Radicioni, Krzysztof Rajda, Markus Reiter-Haas, Alexey Romanov, Ramon Ruiz-Dolz, Hadeel Saadany, Annika Marie Schoene, Samira Shaikh, Ming Shen, Lei Shu, Nádia Silva, Kanishka Silva, Pranaydeep Singh, Rachele Sprugnoli, Shikha Srivastava, Niklas Stoehr, Marco Antonio Stranisci, Tomek Strzalkowski, Yoshi Suhara, Chi Sun, Sandesh Swamy, Jidong Tian, Khanh Tran, Dietrich Trautmann, Enrica Troiano, Maria Mihaela Trușcă, Oto Vale, Andreas van Cranenburgh, Rik van Noord, Aswathy Velutharambath, Jingjing Wang, Rui Wang, Jiaan Wang, Chunliu Wang, Zihan Wang, Hao Wang, Jie Wang, Yang Wu, Lianwei Wu, Jingbo Xia, Yueshen Xu, Chao Yang, Linyi Yang, Zonglin Yang, Shuo Yang, Ziyu Yang, Xiaocui Yang, Zbynek Zajic, Jiandian Zeng, Qingkai Zeng, Zepeng Zhai, Kun Zhang, Zhe Zhang, Wenxuan Zhang, Xulang Zhang, Mao Zhang, Xinliang Frederick Zhang, Yanyan Zhao, Qinghua Zhao, Qiji Zhou, Yuxiang Zhou, Xiaofei Zhu, Luyao Zhu, Sabine Bergler, Paula Carvalho, Simona Frenda, Ramiro H. Gálvez, Tunazzina Islam, Jong Park, Michael Wiegand, Liang-Chih Yu, Rafael Anchiêta, Harika Abburi, Ayme Arango Monnar, Pepa Atanasova, Imene Bensalem, John Cardiff, Yi-Ling Chung, Mithun Das, Dimitar Dimitrov, Goran Glavaš, Jose Manuel Gomez-Perez, Aiqi Jiang, Pica Johansson, Ashiqur KhudaBukhsh, Svetlana Kiritchenko, Katerina Korre, Thomas Mandl, Sarah Masud, John P. McCrae, Sandip Modha, Animesh Mukherjee, Tharindu Ranasinghe, Josef Ruppenhofer, Sheikh Sarwar, Manish Shrivastava, Serena Villata, Zhen Xu, Wenjie Yin, Berlin Chen, Varun Kumar, Xintong Li, Yu Zhang, Lorenzo Gatti, Ran Levy, Wei Lin, Maofu Liu, Manabu Okumura, Chuan Qin, Hiroya Takamura, Yan Zhang, Suyang Zhu, Omid Kashefi, Khyathi Raghavi Chandu, Luyang Huang, Wei Li, Bernd Bohnet, Kim Gerdes, Seong-Bae Park, Mihir Kale, Yun Chen, Steffen Eger, Atsushi Fujita, Yinpeng Guo, Yevgeniy Puzikov, Akihiro Tamura, Xun Wang, Junhui Li, Hou Pong Chan, Yuntian Deng, Sadid A. Hasan, Mayank Singh, Hwaran Lee, Kushal Arora, Bin Bi, Belinda Z. Li, Haoran Li, Jingjing Li, Vivek Kumar Rangarajan Sridhar, Wei Wang, Chen Wu, Yicheng Zou, Pavel Braslavski, Ziqiang Cao, Shubham Chandel, Shen Gao, Liang Pang, Han Wang, Reinald Kim Amplayo, Xuefeng Bai, Daniel Braun, Michele Cafagna, João Campos, Christophe Cerisara, Tuhin Chakrabarty, Guanyi Chen, Yulong Chen, Sangwoo Cho, Liam Cripwell, Yong Dai, Yue Dong, Jiaorong Fan, Tao Ge, Yanjie Gou, Kelvin Han, Gu Hao, Jonathan Herzig, Yi Huang, Xiaotong Jiang, Hui Jiang, David Kauchak, Shaopeng Lai, Wuwei Lan, David Langlois, Wenqiang Lei, Piji Li, Huayang Li, Yafu Li, Anna Liednikova, Changqing Liu, Yahui Liu, Xinglin Lyu, Chang Ma, Yannick Parmentier, Lucas Pavanelli, Dongqi Pu, Haritz Puerto, Clement Rebuffel, Andre Ribeiro, Rudolf Rosa, Tian Shi, William Soto Martinez, Somayajulu Sripada, Qingfeng Sun, Yan Sym, Marc Tanti, Tatiana Vodolazova, Stephen Wan, Yile Wang, Cunxiang Wang, Longbiao Wang, Di Wang, Ante Wang, Saizhuo Wang, Lei Wang, Yumo Xu, Xinnuo Xu, Haoran Yang, Yongjing Yin, Da Yin, Ningyu Zhang, Xiaodong Zhang, Yao Zhao, Haiteng Zhao, Zhixuan Zhou, Shilong Bao, Sabine Schulte im Walde, Wei-Ning Hsu, Afsaneh Fazly, Rahul Aralikatte, Yupeng Hu, Jiaming Xu, Elham Alighardash, Malihe Alikhani, Mattias Appelgren, Chongyang Bai, Digbalay Bose, Gerlof Bouma, Elia Bruni, Andreas Bulling, Joyce Chai, Jaemin Cho, Robin Cooper, Nigel Crook, Abhishek Das, Arash Eshghi, Mehdi Ghanimifard, Thomas Groechel, Ting Han, Nikolai Ilinykh, Waylon Jepsen, Dimosthenis Kontogiorgos, Stefan Kopp, Jae Hee Lee, Annika Lindh, Weiyu Liu, Andy Luecking, Leon Martin, Cynthia Matuszek, Dhruva Patil, Sandro Pezzelle, Hareesh Ravi, Arijit Ray, Robert Ross, Ramon Sanabria, Piyush Sharma, Vésteinn Snæbjarnarson, Ece Takmaz, Alberto Testoni, Jörg Tiedemann, Jialu Wang, Furu Wei, Rui Xia, Guangyue Xu, Sina Zarrieß, Shiyue Zhang, Junwei Zhou, Mathieu Constant, Kilian Evang, Daniel Hershcovich, Takuya Matsuzaki, Alexis Nasr, Shiva Taslimipoor, Veronika Vincze, Canasai Kruengkrai, Shuhei Kurita, Yutai Hou, Shaohui Kuang, Xiuming Qiao, Darsh Shah, Dechuan Teng, David Vilares, Wenduan Xu, Qi Zhang, Laurent Besacier, Francisco Casacuberta, Yong Cheng, Colin Cherry, Eunah Cho, Mattia A. Di Gangi, Roland Kuhn, Yoo Rhee Oh, Katsuhito Sudoh, Min Zhang, Yang Zhao, Chrysoula Zerva, Marcel Bollmann, Masaaki Nagata, Marion Weller-Di Marco, Zhisong Zhang, Hai Zhao, Martin Benjamin, Federico Gaspari, Fatiha Sadat, Nasredine Semmar, Dimitar Shterionov, Lifeng Han, Alessandro Raganato, Qianlong Du, Rongxiang

Weng, Zhen Yang, Rachel Bawden, Josep Crego, Xiangyu Duan, Bryan Eikema, Christian Federmann, Jindřich Libovický, Lemao Liu, Xuebo Liu, Irina Temnikova, Zhaopeng Tu, Yong Wang, Xiangpeng Wei, Hongfei Xu, He Yanqing, Wen Zhang, Jingyi Zhang, Hong Kook Kim, Damien Lolive, Felix Burkhardt, Pu-Jen Cheng, Danilo Croce, Ruihai Dong, Lin Gui, Marco Polignano, Jerome Bellegarda, Mladen Karan, Devamanyu Hazarika, Ashutosh Modi, Shuhuai Ren, Longxu Dou, Phong Le, Zeqi Lin, Antoine Rozenknop, Jordi Atserias Batalla, Michael Bloodgood, Boxing Chen, Leshem Choshen, Cuong Hoang, Arun kumar Jayapal, Jie Jiang, Shun Kiyono, Maoxi Li, Xutan Peng, Jannis Vamvas, David Vilar, Biao Zhang, Mozhi Zhang, Roberto Basili, Gábor Berend, Damir Cavar, Steven Schockaert, Chenghao Yang, Mohamed Amine Belabbes, Shi Feng, Ye Liu, Kosuke Nishida, Asad Abdi, Željko Agić, Shimaa Baraka, Nathaniel Berger, Ed Cannon, Yuan Cao, Rumen Dangovski, Jinhua Du, Ambedkar Dukkipati, Shansan Gong, Michael Hagmann, Taku Hasegawa, Benjamin Heinzerling, Tsutomu Hirao, Tatsuya Hiraoka, Estevam Hruschka, Guangneng Hu, Takahiro Ishihara, Ran Iwamoto, Abhyuday Jagannatha, Shoaib Jameel, Sneha Jha, Tianwen Jiang, Zhuolin Jiang, Md. Yasin Kabir, Hidetaka Kamigaito, Masahiro Kaneko, Atsushi Keyaki, Sammy Khalife, Daiki Kimura, Praveen Kolli, Anna Kruspe, Matthieu Labeau, Tsz Kin Lam, Zhouhan Lin, Hongfu Liu, Suchismit Mahapatra, Masato Mita, Daichi Mochihashi, Marie-Francine Moens, Makoto Morishita, Deepak Muralidharan, Vlad Niculae, Masaaki Nishino, Mayumi Ohta, Yui Oka, Oleg Okun, John Pavlopoulos, Leticia Pinto-Alva, Ariadna Quattoni, Benjamin Ruppik, Monjoy Saha, Motoki Sato, Shigehiko Schamoni, Chao Shang, Edwin Simpson, Ryota Tanaka, Nadi Tomeh, Yuta Tsuboi, Kalyan Veeramachaneni, Marta Villegas, Zhiguang Wang, Koki Washio, Chunyang Xiao, Lei Xu, Naoki Yoshinaga, Eric Zelikman, Chuxu Zhang, Jun Zhang, Wei Zhu, Tanel Alumäe, Minghui Dong, Jaime Lorenzo-Trueba, Sakriani Sakti, Sebastian Stüker, Genta Winata, Noëmi Aepli, Chenchen Ding, Peter Dirix, Ramy Eskander, Laurette Pretorius, Tanja Samardzic, Kevin Scannell, Thuy Vu, Ife Adebara, David Adelani, Tafseer Ahmed, Stergios Chatzikyriakidis, Roald Eiselen, Tanja Gaustad, Iris Hendrickx, Athanasios Karasimos, Parameswari Krishnamurthy, Ritesh Kumar, Bornini Lahiri, Stella Markantonatou, Chenggang Mi, Emmanuel NGUE UM, Atul Kr. Ojha, Hemant Patil, Surangika Ranathunga, Caitlin Richter, Prof. Shikhar Kumar Sarma, Thoudam Doren Singh, Sunayana Sitaram, Brij Mohan Lal Srivastava, Medari Tham, Sree Ganesh Thotempudi, Daan van Esch, Rania Voskaki, Friedel Wolff, Marie Candito, Bo Chen, Gene Kim, Debayan Banerjee, Alessandro Bondielli, Francesca Carota, Anne Cocos, Simone Conia, Gaël Dias, Andrew Gargett, Jonas Golde, Amulya Gupta, Hans Ole Hatzel, Elisabetta Jezek, Chen Jia, Thomas Kober, Jean-Pierre Koenig, Syrielle Montariol, Alexander Panchenko, Paolo Pedinotti, Michael Regan, Asad Sayeed, Arulmozi Selvaraj, Pia Sommerauer, Simon Tannert, Rong Xiang, Judit Ács, Yue Zhang, Kemal Oflazer, Mireia Farrús, Hassan Alhuzali, Dang Tuan Nguyen, Ivandré Paraboni, Jakub Piskorski, Wonsuk Yang, Cristina Bosco, Koji Murakami, Thomas François, Chao Jiang, Tomoyuki Kajiwara, Keshav Singh, Dana Dannélls, Kyosuke Nishida, Guillaume Wisniewski, Bharat Ram Ambati, Ann Bies, Jayeol Chun, Dominique Estival, Carlos Gómez-Rodríguez, Richard Johansson, Arne Köhn, Zhenghua Li, Joakim Nivre, Petya Osenova, Sara Stymne, Alham Fikri Aji, George Foster, Zhengxian Gong, Amir Hazem, Sho Hoshino, Maja Popović, Prokopis Prokopidis, Yves Scherrer, Víctor M. Sánchez-Cartagena, Vincent Vandeghinste, Leonardo Zilio, Raquel Justo, Shay B. Cohen, Rodrigo Wilkens, Djamé Seddah, Mourad Abbas, Stefanie Dipper, Katerina Frantzi, Fabiola Henri, Muhammad Humayoun, Girish Jha, Tracy Holloway King, Ivan Vladimir Meza Ruiz, Emmanuel Morin, Leonardo Campillos-Llanos, Maria Pia di Buono, Francesca Frontini, Jorge Gracia, Grzegorz Kondrak, François Lareau, Gianluca Lebani, Daniel Loureiro, Yuichiroh Matsubayashi, Alexandre Rademaker, Livy Real, Stan Szpakowicz, Hongzhi Xu, Basilio Calderone, Felice Dell'Orletta, Alessio Miaschi, Dong Nguyen, Ana Brassard, Daisuke Kawahara, Setareh Nasihati Gilani, Saku Sugawara, Dian Yu, Abdulaziz Alhamadani, Fernando Alva-Manchego, Maaz Amjad, Dimosthenis Antypas, João António Rodrigues, Ekaterina Artemova, Ai Ti Aw, Petra Bago, Daniel Berleant, Alexander Bondarenko, Emanuela Boros, Donald Brown, Elena Bruches, Caroline Brun, Bert-Jan Butijn, Chenxuan Cui, Aijun DAI, Debajyoti Datta, Edmund Dervakos, Bill Dolan, Chris Chinenye Emezue, Maria Eskevich, David Eyers, Amany Fashwan, Andrea Favalli, Besnik Fetahu, André Freitas, Sebastian Gehrmann, Anatole Gershman, Cristina Giannone, Luís Gomes, Nizar Habash, Stefan Heindorf, Eduard Hovy, Canming Huang, Daniil Huryn, Hitoshi Isahara, Alexandra Ivoylova, Myeongjun Jang, Adam Jatowt, Bogeun Jo, Salam Khalifa, Vivek

Khetan, Denis Kleyko, Bruce Kogut, Daan Kolkman, Olga Kovaleva, Maria Kunilovskaya, Donghun Lee, Colin Leong, Adam Lion-Bouton, Diane Litman, Wenge Liu, Chuang Liu, Duanchen Liu, Valeriy Lobov, Ehsan Lotfi, Pilar López Úbeda, Debanjan Mahata, Valentin Malykh, Hamdy Mubarak, Garrett Nicolai, Pedro Ortiz Suarez, Ana Belén Parras Portillo, Agnieszka Patejuk, Pavel Pecina, Deenuka Perera, Dominic Petrak, Dmitri Roussinov, Alla Rozovskaya, Anna Rumshisky, Ananya Sai B, Rodrigo Santos, Aditya Shah, Anastassia Shaitarova, Tatiana Shavrina, João Ricardo Silva, Dawn Song, Aitor Soroa, Julius Steen, Benno Stein, Kevin Stowe, Fiona Anting Tan, Alexey Tikhonov, Doan Nam Long Vu, Houfeng Wang, Dingzirui Wang, Chenguang Wang, Baoxin Wang, Bingdong Wang, Chengkun Wei, Shira Wein, Daniel Whitenack, Magdalena Wolska, Dayong Wu, Bo Xu, Seunghyun Yoon, Dechen Zhan, Bin Zhou, Dawei Zhu, Pierre Zweigenbaum, Zdeněk Žabokrtský, Li Dong, Bang Liu, Maximin Coavoux, Hailong Cao, Chi Hu, Jan Niehues, Tsuyoshi Okita, Lijun Wu, Heng Yu, François Yvon, Jiali Zeng, Zhenisbek Assylbekov, Vinay Praneeth Boda, Nadir Durrani, Ankit Gupta, Shizhu He, Eugene Jang, Gyuwan Kim, Chen Li, Yuanxing Liu, Nelson F. Liu, Salvatore Romeo, Tianxiang Sun, Yinfei Yang, Octavia Şulea, Terra Blevins, Katharina Kann, Tomáš Brychcín, Allyson Ettinger, Jiaqi Guo, Julie Weeds, Yiming Cui, Ning Ding, Bill Yuchen Lin, Arpit Sharma, Sanghwan Bae, Hyunsoo Cho, Won Ik Cho, Won Ik Cho, Hojun Cho, Kenneth Church, Chengyu Dong, Linyuan Gong, Xiaodong Gu, Han Guo, Donghoon Ham, Yu-Jung Heo, Joel Jang, Hwiyeol Jo, Jaewook Kang, Geonmin Kim, Sungdong Kim, Jin-Hwa Kim, Sungju Kim, Miyoung Ko, Donghyun Kwak, Sang-Woo Lee, Hwanhee Lee, Dong-Hyun Lee, Chia-Hsuan Lee, Koanho Lee, Chen Liang, Hongyin Luo, Kaixin Ma, Arya D. McCarthy, Hongyuan Mei, Kyoung-Woon On, ChaeHun Park, Jihyeon Roh, Cory Shain, Lucas Torroba Hennigen, Zirui Wang, Chengyu Wang, Yaqing Wang, Zilong Wang, Shijie Wu, Zeqiu Wu, Yiqing Xie, Dongkuan Xu, Ruohong Zhang, Yu Zhang, Simiao Zuo, Dimitrios Kokkinakis, Oana Cocarascu, Aditya Joshi, Henning Wachsmuth, Fabio Massimo Zanzotto, Yuki Arase, Forrest Sheng Bao, Nadjet Bouayad-Agha, Pablo Duboue, Helena Caseli, Hiroki Ouchi, Kenji Sagae, Rachel Roxas, Ankit Srivastava, Dhanasekar Sundararaman, Xiaofeng Wu, Zheng Yuan, Kourosh Meshgi, Andrey Kutuzov, Nemanja Djuric, Pengfei Duan, Haihong E, Isao Goto, Cyril Labbe, Anastassia Loukina, Michael Flor, Sendong Zhao, Alvaro Rodrigo, Eiji Aramaki, Christian Chiarcos, Nils Reiter, Nikolaos Aletras, Timothy Baldwin, Monojit Choudhury, Hsin-Hsi Chen, Qin Dai, Francesco Corcoglioniti, Yutaka Sasaki, Tyler Baldwin, Eric Brown, Montse Cuadros, Ernesto William De Luca, Jana Diesner, Zuohui Fu, Adam Funk, Koldo Gojenola, Natalia Grabar, Anupam Guha, Dongfang Li, Huy Nguyen, Haris Papageorgiou, Francesco Ronzano, Junsheng Zhou, Pooja Aggarwal, Preethi Raghavan, Jifan Yu, Claudia Borg, Lena Jäger, Felix Sasaki, Muthu Kumar Chandrasekaran, Gareth Jones, Piotr Przybyła, Stergos Afantenos, Hen-Hsen Huang, Fan Yang, Nate Blaylock, Sudeep Gandhe, Nikhil Mehta, Jiahuan Pei, Eran Raveh, Tatsuki Kuribayashi, Svetla Koeva, Yuanhe Tian, Bayan AbuShawar, Reina Akama, Lourdes Araujo, Aitziber Atutxa Salazar, Ander Barrena, Guntis Barzdins, Steven Bethard, Arantza Casillas, Roberto Centeno, Lei Chen, Jonathan Chevelu, Dhivya Chinnappa, Agustín Delgado, Alberto Diaz, Luis Espinosa Anke, Agata Filipowska, Fumiyo Fukumoto, Dimitris Galanis, Andrea Galassi, Iker García-Ferrero, Luis Gasco Sánchez, Iakes Goenaga, Asier Gutiérrez-Fandiño, Gahgene Gweon, Siegfried Handschuh, Oliver Hellwig, Andrea Horbach, Md Mosharaf Hossain, Shaohan Huang, Yoshinobu Kano, Yasutomo Kimura, Mamoru Komachi, Kanako Komiya, Mikel Larrañaga, Alberto Lavelli, John Lee, Oier Lopez de Lacalle, Inigo Lopez-Gazpio, Wolfgang Maier, Rathnakara Malatesha, Juan Martinez-Romo, Raquel Martínez, Ditty Mathew, Shigeki Matsubara, Alessandro Mazzei, Kathleen McKeown, Alice Millour, Maryam Sadat Mirzaei, Abhijit Mishra, Junta Mizuno, Benjamin Molinet, Manuel Montes, Antonio Moreno-Sandoval, Hideki Nakayama, Maite Oronoz, Alicia Pérez, Traian Rebedea, Georg Rehm, Federico Ruggeri, Oscar Sainz, Hiroki Sakaji, Ander Salaberria, Christian Sallaberry, Iñaki San Vicente, Minoru Sasaki, Yan Song, Ander Soraluze, Gorka Urbizu, Masao Utiyama, Manuel Vilares Ferro, Elena Volodina, Boyi Xie, Masaharu Yoshioka, Philip Yu, Xinchen Yu, Sahar Ghannay, Bo An, Lidong Bing, Hongjie Cai, Cuiyun Gao, Zhijiang Guo, Yaojie Lu, Eugenio Martínez-Cámara, Marina Santini, Zihan Wang, Zhen Wu, Kang Xu, Yllias Chali, Junfan Chen, Brian Davis, Heba Elfardy, Po Hu, Zhiyuan Hu, Zhuoxuan Jiang, Muyang Ma, Martina Miliani, Sudip Naskar, Naoaki Okazaki, Patrick Saint-Dizier, Frank Schilder, Juan-Manuel Torres-Moreno, Zhaowei Wang, Jie Yang, Haitao Zheng, Peng Zhou, Adam Grycner, Manling Li, Di Lu,

Yuan Yao, Salah Aït-Mokhtar, Éric de la Clergerie, Jae Sung Lee, Yuji Matsumoto, Kehai Chen, Yuhang Guo, Amy Siu, Xing Wang, Boliang Zhang, Zhu Junguo, Hsun-Hsien Shane Chang, Anna Corazza, Cyril Goutte, Vasileios Lampos, Yong Liu, Wei Lu, Yujia Qin, Yatian Shen, Xiaozhi Wang, Jiawei Yong, Rute Costa, Yongbin Liu, Antonio Pareja-Lora, Antske Fokkens, Hugo Gonçalo Oliveira, Gabriella Lapesa, Diana Santos, Camilo Thorne, Clare Voss, Dina Wonsever, Horacio Saggion, Cornelia Caragea, Phillippe Langlais, Chiara Alzetta, Jinho D. Choi, Arantza Diaz de Ilarraza, Maud Ehrmann, Thierry Hamon, Leonhard Hennig, Pedram Hosseini, Ming Jiang, John Lawrence, Amir Pouran Ben Veyseh, Roland Roller, Hai-Long Trieu, Zuchao Li, Daniel Campos, Yixin Cao, Fenia Christopoulou, Xu Han, Haoyan Liu, Zihan Wang, Jun Araki, Marco Basaldella, Christos Christodoulopoulos, Bonaventura Coppola, Xiao Ding, Haibo Ding, Diana Galvan-Sosa, Shu Guo, Linmei Hu, Filip Ilievski, Harksoo Kim, Man Lan, Changliang Li, Aakanksha Naik, Qiang Ning, Yadollah Yaghoobzadeh, Weixin Zeng, Li Zhang, Haoyu Zhang, Bowei Zou, Annalina Caputo, Jena D. Hwang, Gustavo Aguilar, Akiko Aizawa, Adrien Bardet, Loïc Barrault, Guanqun Bi, Tianyi Bi, Shuqing Bian, Pengfei Cao, Mengyun Cao, Boxi Cao, Zheng Chaoqun, Chong Chen, Jia Chen, Jingqiang Chen, Jiawei Chen, Si Chen, Chao Chen, Wei Chen, Gong Cheng, Xin Cong, Shiyao Cui, Hai Cui, Joe Davison, Wenmin Deng, Qian Dong, Shihan Dou, Junwen Duan, Fang Fang, Zheng Fang, Tianqing Fang, Hao Fei, Changkai Feng, Olivier Ferret, Pengzhi Gao, Yi Guan, Bin Guo, Zhongyuan Han, Ridong Han, Xiaohui Han, Chuzhan Hao, Wanwei He, Ruifang He, Tao He, Guoxiu He, Yining Hua, Quzhe Huang, Changzhen Ji, Wenbin Jiang, Zhengbao Jiang, Junyi Li, Guanlin Li, Jingye Li, Bobo Li, Jing Li, Fenghuan Li, Chengming Li, Xiangsheng Li, Mingda Li, Shaobo Li, Jinpeng Li, Dan Li, Zhucong Li, Miya Liang, Hongyu Lin, Yijiang Liu, MeiLing Liu, Jian Liu, Fangchao Liu, Yu Liu, Wei Liu, Qianying Liu, Luyang Liu, Qingbin Liu, Yinxia Lou, Wenpeng Lu, Pan Lu, Xin Lv, Wei-Yun Ma, Ruotian Ma, Xing Ma, Clint Solomon Mathialagan, Zhijie Nie, Youcheng Pan, Tao Peng, Furong Peng, Maxime Peyrard, Pragaash Ponnusamy, Yuqiu Qian, Yaqiong Qiao, Yanxia Qin, Zhaopeng Qiu, Yunqi Qiu, Feiliang Ren, Yubing Ren, Ruiyang Ren, Julien Rossi, Mingyue Shang, Yunqiu Shao, Ying Shen, Jiawei Sheng, Shaoyun Shi, Yifan Song, Chuancheng Song, Dianbo Sui, Kai Sun, Zhongkai Sun, Yang Sun, YiHeng Sun, Hongye Tan, Jialong Tang, Qingyue Wang, Shi Wang, Bo Wang, Chao Wang, Yanlin Wang, Pengfei Wang, Peifeng Wang, Shoujin Wang, Xiao Wang, Wenhao Wu, Ting Wu, Shengqiong Wu, Zhijing Wu, Junshuang Wu, Jiageng Wu, Yuting Wu, Yu Xia, Yang Xiao, Xiaohui Xie, Yuqing Xie, Minghui Xie, Yunhe Xie, Yongxiu Xu, Zhiyang Xu, Chunpu Xu, Wang Xu, Chenwei Yan, Hang Yang, Wenting Ye, Changlong Yu, Xiangrong Zeng, WenYu Zhan, Jingtao Zhan, Zhengyan Zhang, Hao Zhang, Fan Zhang, Junqi Zhang, Wei Emma Zhang, Junchi Zhang, Hu Zhang, Chen Zhang, Haoran Zhang, Liwen Zhang, ZheYu Zhang, Xiaokun Zhang, Jinming Zhao, Jun Zhao, Xin Zhao, Xiaoyan Zhao, Xinyan Zhao, Zihao Zheng, Dequan Zheng, Yaowei Zheng, Long Zhou, Bo Zhou, Yang Zhou, Xin Zhou, Kun Zhou, Peilin Zhou, Ying Zhou, Yuhao Zhou, Xinyu Zou, Xinyu Zuo, Saadullah Amin, Akari Asai, Qingqing Cao, Philipp Cimiano, Yifan Gao, Pengyu Hong, Ziming Huang, Peter Jansen, Rik Koncel-Kedziorski, Kalpesh Krishna, Linqing Liu, Alex Lưu, Mohammad Hadi Mohammadi, Nicholas Monath, Saeed Najafi, Massimo Nicosia, Naoki Otani, Siddharth Patwardhan, Naveen Raman, Candace Ross, Chenglei Si, Stalin Varanasi, Marc Verhagen, Shufan Wang, Yizhong Wang, Dan Wang, Yunfang Wu, Fangyuan Xu, Dongxu Zhang, Chen Zhao, Wenlong Zhao, Jennifer Williams, Alexandra I. Cristea, Lun-Wei Ku, Debora Nozza, Simone Paolo Ponzetto, Richard Evans, Brigitte Krenn, Annemarie Friedrich, Eleftherios Avramidis, Jeff Good, Kellie Webster, Helen Aristar-Dry, Senthil Kumar B, Bharathi Raja Chakravarthi, Jens Lemmens, Krister Lindén, Nitin Madnani, Kathy McCoy, Margot Mieskes, Barbara Plank, Tyler Schnoebelen, Elke Teich, Esau Villatoro-Tello, Wang Gao, Baoli Li, Efstathios Stamatatos, Alberto Bugarín-Diz, Suchet Chachra, John Conroy, Xinyu Dai, Ioannis Konstas, Yanyan Lan, Marina Litvak, Vibhu Mittal, Jian-Yun Nie, Xiaolei Huang, Braja Gopal Patra, Xian Zhong, Santiago Egea Gómez, Feng Ji, Benjamin Roth, Saeideh Shahrokh Esfahani, Ferhan Ture, Dong Zhou, Nina Zhou, Christoph Tillmann, Beatrice Daille, Yoshihiko Hayashi, Aminul Islam, Huyen Nguyen, Xunjie Zhu, Alberto Barrón-Cedeño, Vivek Gupta, Malik Altakrori, Daniel Cer, Hye-Jin Min, Taeuk Kim, Lei Chen, Junhao Liu, Sun Kim, He Bai, Florian Boudin, Jon Chamberlain, Jin-Woo Chung, Emrah Inan, Anastasia Krithara, Luis Gerardo Mojica de la Vega, Sheshera Mysore, Zhong Qian, Andon Tchechmedjiev, Christian Wartena, yanmin chen, Pan Du, Djoerd Hiemstra, Nikhil Londhe,

Zhuoer Wang, Ashutosh Adhikari, Senthil Chandramohan, Zhiyu Chen, Gerard de Melo, Kalpit Dixit, Zhicheng Dou, Hui Fang, Luke Gallagher, Nazli Goharian, Alexander Henlein, Lin Li, Sheng-Chieh Lin, Zhenghao Liu, Weizhi Ma, Cristiano Russo, Peng Shi, Cristian Tommasino, Mohamed Trabelsi, Ji Xin, Jiawei Zhang, Wei Zhong, Chloé Braud, Ivan Habernal, Yufang Hou, I-Ta Lee, Yu-Yun Chang, Yang Gao, Helmut Horacek, Leila Kosseim, Junyi Jessy Li, Sharid Loáiciga, Costanza Navarretta, Ielka van der Sluis, Ping Jian, Andrei Popescu-Belis, Haixia Chai, Johan Bos, Debopam Das, Pascal Denis, Kurt Eberle, Michael Roth, Amalia Todirascu, Peifeng Li, Katja Markert, Barbara Di Eugenio, Ines Rehbein, Takenobu Tokunaga, Amir Zeldes, Deniz Zeyrek, Heike Zinsmeister, Yu Hong, Naoya Inoue, Rajkumar Pujari, Grégoire Winterstein, Maxime Amblard, Arie Cattan, Prafulla Kumar Choubey, Markus Egg, Patrick Ehlen, Giacomo Ferrari, Sucheta Ghosh, Akhilesh Deepak Gotmare, Fengyu Guo, Patrick Huber, Mikel Iruskieta, Sungho Jeon, Jad Kabbara, Helena Moniz, Mathieu Morey, Noriki Nishida, Maciej Ogrodniczuk, Haoruo Peng, Martin Pereira-Fariña, Dan Simonson, Wei Song, Changxing Wu, Wenlin Yao, Qingyu Yin, Hikaru Yokono, Juntao Yu, Frances Yung, Guan-Lin Chao, Shiyin Kang, Kazunori Komatani, Xianchao Wu, Koichiro Yoshino, Zhengyu Zhou, Annalena Aicher, Dipankar Das, Wei Gao, Patrick Paroubek, Rui Wang, Simon Wells, Andrea Kahn, Anusha Balakrishnan, Xiuying Chen, Shaobo Cui, Leyang Cui, Nina Dethlefs, Yimai Fang, Karthik Gopalakrishnan, Jie Hao, Bin He, Lisa Jin, Gwénolé Lecorvé, Wenjie Li, Yanran Li, Ramesh Manuvinakurike, Alex Marin, Zhaochun Ren, Kristina Striegnitz, María Inés Torres, Kartikeya Upasani, Jian Wang, Wei Wei, Chien-Sheng Wu, Yinhe Zheng, Qingfu Zhu, Timo Baumann, Zhiqi Huang, Michael Johnston, Igor Shalyminov, Lucia Donatelli, Marco Del Tredici, Trung Bui, Angel Chang, Kotaro Funakoshi, Xiujun Li, Haoyu Song, Chongyang Tao, Kun Xu, Jie Cao, Yang Feng, Yang Li, Nikita Moghe, Derek F. Wong, Giuseppe Castellucci, Lin Chen, Jen-Tzung Chien, Milica Gasic, Christian Geishauser, Raghav Gupta, Kevin Small, Qing Zhang, Bing Zhao, Chinnadhurai Sankar, Lifeng Jin, Carl Vogel, Jana Götze, Alessandra Zarcone, Ron Artstein, Claire Bonial, Susanne Burger, Luisa Coheur, Shih-Hung Wu, Qiang Zhou, JinYeong Bak, David Griol, Zeyang Lei, Ruhi Sarikaya, Zhuosheng Zhang, Pushpak Bhattacharyya, Eugénio Ribeiro, Haoyang Wen, Congying Xia, Yujiu Yang, Danish Contractor, Julian Eisenschlos, Ridong Jiang, Jung-Jae Kim, Jiangtong Li, Junyu Lin, Kai Sun, Haoyu Wang, Baoxun Wang, Bowen Wu, Rishita Anubhai, Shichao Dong, Chuan Meng, Lasguido Nio, Kiyonori Ohtake, Eriks Sneiders, Jian Sun, Lei Yu, Paul Piwek, Jan Alexandersson, Donna Byron, Elisa Ferracane, Rahul Goel, Yulia Grishina, Ryuichiro Higashinaka, Takeshi Homma, Stephen Pulman, Matthew Purver, Sylvie Saget, Nancy Chen, Deborah Dahl, Aswarth Abhilash Dara, Shutong Feng, Jun Gao, Behnam Hedayatnia, Matthias Kraus, Ye Liu, Gaurav Pandey, Sapan Patel, Lina M. Rojas Barahona, Ethan Selfridge, Hiroaki Sugiyama, Jianqiao Zhao, Ari Rappoport, Aixiu An, Gosse Bouma, John Hale, Pavel Logacev, Claudia Marzi, William Schuler, Mark Steedman, Benoit Crabbé, Krista Lagus, Shaorong Yan, Jelke Bloem, Berthold Crysmann, Mika Hämäläinen, Chara Tsoukala, Géraldine Walther, Valentina Benedettini, Hiram Calvo, Rui Chaves, Marcello Ferro, Nabil Hathout, Maria Liakata, Giorgio Magri, Ludovica Pannitto, Alessandro Panunzi, Yuval Pinter, Ingo Plag, Giulia Rambelli, Michael Zock, Sabine Bartsch, Dag Haug, Omri Abend, Maria Barrett, Dorothee Beermann, António Branco, Dominique Brunato, Daniel Hardt, Dandan Huang, Olga Seminck, Giulia Venturi, Yao Yao, Jonathan Brennan, Brian Dillon, Kumiko Tanaka-Ishii, Denis Kiselev, Maria Becker, Sanghoun Song, Rachel Ryskin, Ethan Wilcox, Hendrik Buschmeier, Gregory Mills, Anne Abeille, Matthieu Bach, Beata Beigman Klebanov, Marc Brysbaert, Spencer Caplan, Tejaswini Deoskar, Pablo Gamallo, Muskan Garg, Keith Harrigian, Joshua K. Hartshorne, Tim Hunter, Shaoxiong Ji, Sylvain Kahane, Najoung Kim, Ningyun Li, Li Lucy, Jessica Ouyang, Iris Oved, Chandni Saxena, Shane Steinert-Threlkeld, Prayag Tiwari, Adithya V Ganesan, Xianzhi Wang, Andrew Yates, Seren Yenikent, Tao Yu, Tianlin Zhang, Huijun Zhang, Jenny Zhang, Ayah Zirikly, Björn Schuller, Thierry Declerck, Dirk Heylen, Maria Wolters, Gina-Anne Levow, Florian Metze, Xixin Wu, Pengfei Liu, Pierpaolo Basile, Valerio Basile, Davide Buscaldi, Elena Cabrio, Roman Klinger, Els Lefever, Qian Liu, Nikola Ljubešić, Rui Mao, Constantin Orăsan, Viviana Patti, Ruifeng Xu, Tanmoy Chakraborty, Elisabetta Fersini, Zeerak Talat, Sara Tonelli, Vasudeva Varma, Bertie Vidgen, Arkaitz Zubiaga, Guangsheng Bao, Deng Cai, Thiago Castro Ferreira, Vera Demberg, Ondřej Dušek, Angela Fan, Albert Gatt, Dimitra Gkatzia, Fei Liu, Yang Liu, Qi Liu, Jiangming Liu, Shashi Narayan, Leonardo F. R. Ribeiro, Anastasia Shimorina, Jinsong

Su, Jiajun Zhang, Stella Frank, Casey Kennington, Parisa Kordjamshidi, Nikhil Krishnaswamy, Jie Lei, Carina Silberer, Jesse Thomason, Diedrich Wolter, Jiang Guo, Yijia Liu, Yuxuan Wang, Meishan Zhang, Cristina España-Bonet, Yvette Graham, Julia Kreutzer, Qun Liu, Chi-kiu Lo, Kenton Murray, Juan Pino, Dipti Sharma, Taro Watanabe, Wilker Aziz, Dan Goldwasser, Gholamreza Haffari, Katsuhiko Hayashi, Sosuke Kobayashi, Lingpeng Kong, Joseph Le Roux, Pranava Madhyastha, Tetsuji Nakagawa, Stefan Riezler, Sho Takase, Duygu Ataman, Kalika Bali, Harald Hammarström, Kengatharaiyer Sarveswaran, Alexandros Tantos, Ashwini Vaidya, Menno van Zaanen, Gözde Şahin, Chris Biemann, Gemma Boleda, Emmanuele Chersoni, Paul Cook, Diego Frassinelli, Simonetta Montemagni, Thierry Poibeau, Agata Savary, Tim van de Cruys, Ivan Vulić, Sina Ahmadi, Sara Carvalho, Anca Dinu, Miloš Jakubíček, Anisia Katinskaia, Penny Labropoulou, Ekaterina Lapshinova-Koltunski, Paul Rayson, Claudia Soria, Marko Tadić, George Tambouratzis, Seid Muhie Yimam, Marcos Zampieri, Hao Cheng, Manuel Ciosici, Matthias Gallé, Junjie Hu, Yoon Kim, Liyuan Liu, Colin Raffel, Minjoon Seo, Alessandro Sordoni, Kang Min Yoo, Gorka Azkune, Jeremy Barnes, Eduardo Blanco, Chenhui Chu, Takashi Inui, Egoitz Laparra, Roser Morante, Yugo Murawaki, Alessio Palmero Aprosio, Paolo Torroni, Yongfeng Zhang, Khalid Al Khatib, Yanan Cao, Yubo Chen, Wenliang Chen, Nicolas Dugué, Tao Gui, Xianpei Han, Jialong Han, Lifu Huang, Evangelos Kanoulas, Chenliang Li, Peng Li, Fei Li, Tingwen Liu, Ming Liu, Thien Nguyen, Dheeraj Rajagopal, David Reymond, Alan Ritter, Mathieu Roche, Chengjie Sun, Yannick Toussaint, Tong Xu, Rui Yan, Jie Yang, Min Yang, Muyun Yang, Daojian Zeng, Qi Zhang, Richong Zhang, Peng Zhang, Wayne Xin Zhao, Sewon Min, Guenter Neumann, Axel-Cyrille Ngonga Ngomo, Jianmo Ni, Panupong Pasupat, Christian Retoré, Pedro Rodriguez, Mrinmaya Sachan, Keh-Yih Su, Ping Wang, Gijs Wijnholds, Caiming Xiong, Steven Bird, Walter Daelemans, Karën Fort, Jin-Dong Kim, Kevin Duh, Claudia Hauff, Robert Krovetz, Joel Mackenzie, Alexander Mehler, Antonio Rinaldi, Lucy Lu Wang, Shafiq Joty, Massimo Poesio, Hannah Rashkin, Manfred Stede, Bonnie Webber, Ingo Siegert, Zoraida Callejas, Patrick Gebhard, Stefan Hillmann, Kristiina Jokinen, Michael McTear, Catherine Pelachaud, Stefan Schaffer, Stefan Ultes, Fei Xia, Lisa Beinborn, Stefan L. Frank, Kyle Gorman, Jordan Kodner, Jackson Lee, Stephanie M. Lukin, Mark-Jan Nederhof, Yohei Oseki, Jonathan Rawski, H. Andrew Schwartz, Edward Stabler, Michael White, Kelley Lynch, Gitit Kehat, Kenneth Lai, Eva Maria Vecchi, David R. Mortensen, Maciej Piasecki, Carlos Ramisch, Dian Yu, Ryohei Sasano, Jingxuan Tu, Zekun Wang, Ruobing Xie, Cheng Yang, Jie Zou, Zi-Yi Dou, Eben Holderness, Kyeongmin Rim, Bingyang Ye, Zhongqing Wang, Deyi Xiong, Xiang Lorraine Li, Xiang Lorraine nan, Torsten Zesch, Jeff Heflin

Table of Contents

Do Language Models Make Human-like Predictions about the Coreferents of Italian Anaphoric Zero Pronouns?
James A. Michaelov and Benjamin K. Bergen 1

Language Acquisition through Intention Reading and Pattern Finding
Jens Nevens, Jonas Doumen, Paul Van Eecke and Katrien Beuls 15

Stability of Syntactic Dialect Classification over Space and Time
Jonathan Dunn and Sidney Wong 26

Subject Verb Agreement Error Patterns in Meaningless Sentences: Humans vs. BERT
Karim Lasri, Olga Seminck, Alessandro Lenci and Thierry Poibeau 37

Measuring Morphological Fusion Using Partial Information Decomposition
Michaela Socolof, Jacob Louis Hoover, Richard Futrell, Alessandro Sordoni and Timothy J. O'Donnell 44

Smells like Teen Spirit: An Exploration of Sensorial Style in Literary Genres
Osama Khalid and Padmini Srinivasan 55

Metaphorical Polysemy Detection: Conventional Metaphor Meets Word Sense Disambiguation
Rowan Hall Maudslay and Simone Teufel 65

Machine Reading, Fast and Slow: When Do Models "Understand" Language?
Sagnik Ray Choudhury, Anna Rogers and Isabelle Augenstein 78

Hierarchical Attention Network for Explainable Depression Detection on Twitter Aided by Metaphor Concept Mappings
Sooji Han, Rui Mao and Erik Cambria 94

Multi-view and Cross-view Brain Decoding
Subba Reddy Oota, Jashn Arora, Manish Gupta and Raju S. Bapi 105

Visio-Linguistic Brain Encoding
Subba Reddy Oota, Jashn Arora, Vijay Rowtula, Manish Gupta and Raju S. Bapi 116

Gestures Are Used Rationally: Information Theoretic Evidence from Neural Sequential Models
Yang Xu, Yang Cheng and Riya Bhatia 134

Revisiting Statistical Laws of Semantic Shift in Romance Cognates
Yoshifumi Kawasaki, Maëlys Salingre, Marzena Karpinska, Hiroya Takamura and Ryo Nagata 141

Character Jacobian: Modeling Chinese Character Meanings with Deep Learning Model
Yu-Hsiang Tseng and Shu-Kai Hsieh 152

COMMA: Modeling Relationship among Motivations, Emotions and Actions in Language-based Human Activities
Yuqiang Xie, Yue Hu, Wei Peng, Guanqun Bi and Luxi Xing 163

Exploring Semantic Spaces for Detecting Clustering and Switching in Verbal Fluency
Özge Alacam, Simeon Schüz, Martin Wegrzyn, Johanna Kißler and Sina Zarrieß 178

Neuro-Symbolic Visual Dialog
Adnen Abdessaied, Mihai Bâce and Andreas Bulling 192

LINGUIST: Language Model Instruction Tuning to Generate Annotated Utterances for Intent Classification and Slot Tagging
Andy Rosenbaum, Saleh Soltan, Wael Hamza, Yannick Versley and Markus Boese 218

Adaptive Natural Language Generation for Task-oriented Dialogue via Reinforcement Learning
Atsumoto Ohashi and Ryuichiro Higashinaka 242

TAKE: Topic-shift Aware Knowledge sElection for Dialogue Generation
Chenxu Yang, Zheng Lin, Jiangnan Li, Fandong Meng, Weiping Wang, Lanrui Wang and Jie Zhou 253

Dynamic Dialogue Policy for Continual Reinforcement Learning
Christian Geishauser, Carel van Niekerk, Hsien-chin Lin, Nurul Lubis, Michael Heck, Shutong Feng and Milica Gašić 266

GRAVL-BERT: Graphical Visual-Linguistic Representations for Multimodal Coreference Resolution
Danfeng Guo, Arpit Gupta, Sanchit Agarwal, Jiun-Yu Kao, Shuyang Gao, Arijit Biswas, Chien-Wei Lin, Tagyoung Chung and Mohit Bansal 285

Learning to Improve Persona Consistency in Multi-party Dialogue Generation via Text Knowledge Enhancement
Dongshi Ju, Shi Feng, Pengcheng Lv, Daling Wang and Yifei Zhang 298

Improving Top-K Decoding for Non-Autoregressive Semantic Parsing via Intent Conditioning
Geunseob Oh, Rahul Goel, Chris Hidey, Shachi Paul, Aditya Gupta, Pararth Shah and Rushin Shah 310

Autoregressive Entity Generation for End-to-End Task-Oriented Dialog
Guanhuan Huang, Xiaojun Quan and Qifan Wang 323

Continual Few-shot Intent Detection
Guodun Li, Yuchen Zhai, Qianglong Chen, Xing Gao, Ji Zhang and Yin Zhang 333

"Mama Always Had a Way of Explaining Things So I Could Understand": A Dialogue Corpus for Learning to Construct Explanations
Henning Wachsmuth and Milad Alshomary 344

Schema Encoding for Transferable Dialogue State Tracking
Hyunmin Jeon and Gary Geunbae Lee 355

A Personalized Dialogue Generator with Implicit User Persona Detection
Itsugun Cho, Dongyang Wang, Ryota Takahashi and Hiroaki Saito 367

Incorporating Casual Analysis into Diversified and Logical Response Generation
Jiayi Liu, Wei Wei, Zhixuan Chu, Xing Gao, Ji Zhang, Tan Yan and Yulin Kang 378

Reciprocal Learning of Knowledge Retriever and Response Ranker for Knowledge-Grounded Conversations
Jiazhan Feng, Chongyang Tao, Zhen Li, Chang Liu, Tao Shen and Dongyan Zhao 389

CR-GIS: Improving Conversational Recommendation via Goal-aware Interest Sequence Modeling
Jinfeng Zhou, Bo Wang, Zhitong Yang, Dongming Zhao, Kun Huang, Ruifang He and Yuexian Hou 400

GRASP: Guiding Model with RelAtional Semantics Using Prompt for Dialogue Relation Extraction
Junyoung Son, Jinsung Kim, Jungwoo Lim and Heuiseok Lim 412

PEPDS: A Polite and Empathetic Persuasive Dialogue System for Charity Donation
Kshitij Mishra, Azlaan Mustafa Samad, Palak Totala and Asif Ekbal 424

DialAug: Mixing up Dialogue Contexts in Contrastive Learning for Robust Conversational Modeling
Lahari Poddar, Peiyao Wang and Julia Reinspach 441

A Closer Look at Few-Shot Out-of-Distribution Intent Detection
Li-Ming Zhan, Haowen Liang, Lu Fan, Xiao-Ming Wu and Albert Y.S. Lam 451

CGIM: A Cycle Guided Interactive Learning Model for Consistency Identification in Task-oriented Dialogue
Libo Qin, Qiguang Chen, Tianbao Xie, Qian Liu, Shijue Huang, Wanxiang Che and Zhou Yu . 461

CorefDiffs: Co-referential and Differential Knowledge Flow in Document Grounded Conversations
Lin Xu, Qixian Zhou, Jinlan Fu, Min-Yen Kan and See-Kiong Ng 471

SelF-Eval: Self-supervised Fine-grained Dialogue Evaluation
Longxuan Ma, Ziyu Zhuang, Weinan Zhang, Mingda Li and Ting Liu 485

Open-Domain Dialog Evaluation Using Follow-Ups Likelihood
Maxime De Bruyn, Ehsan Lotfi, Jeska Buhmann and Walter Daelemans 496

Joint Goal Segmentation and Goal Success Prediction on Multi-Domain Conversations
Meiguo Wang, Benjamin Yao, Bin Guo, Xiaohu Liu, Yu Zhang, Tuan-Hung Pham and Chenlei Guo 505

Slot Dependency Modeling for Zero-Shot Cross-Domain Dialogue State Tracking
Qingyue Wang, Yanan Cao, Piji Li, Yanhe Fu, Zheng Lin and Li Guo 510

Section-Aware Commonsense Knowledge-Grounded Dialogue Generation with Pre-trained Language Model
Sixing Wu, Ying Li, Ping Xue, Dawei Zhang and Zhonghai Wu 521

Using Multi-Encoder Fusion Strategies to Improve Personalized Response Selection
Souvik Das, Sougata Saha and Rohini K. Srihari 532

A Multi-Dimensional, Cross-Domain and Hierarchy-Aware Neural Architecture for ISO-Standard Dialogue Act Tagging
Stefano Mezza, Wayne Wobcke and Alan Blair 542

SPACE-2: Tree-Structured Semi-Supervised Contrastive Pre-training for Task-Oriented Dialog Understanding
Wanwei He, Yinpei Dai, Binyuan Hui, Min Yang, Zheng Cao, Jianbo Dong, Fei Huang, Luo Si and Yongbin Li 553

ET5: A Novel End-to-end Framework for Conversational Machine Reading Comprehension
Xiao Zhang, Heyan Huang, Zewen Chi and Xian-Ling Mao 570

CoHS-CQG: Context and History Selection for Conversational Question Generation
Xuan Long Do, Bowei Zou, Liangming Pan, Nancy F. Chen, Shafiq Joty and Ai Ti Aw 580

Semantic-based Pre-training for Dialogue Understanding
Xuefeng Bai, Linfeng Song and Yue Zhang 592

Distribution Calibration for Out-of-Domain Detection with Bayesian Approximation
Yanan Wu, Zhiyuan Zeng, Keqing He, Yutao Mou, Pei Wang and Weiran Xu 608

Tracking Satisfaction States for Customer Satisfaction Prediction in E-commerce Service Chatbots
Yang Sun, Liangqing Wu, Shuangyong Song, Xiaoguang Yu, Xiaodong He and Guohong Fu .. 616

Towards Multi-label Unknown Intent Detection
Yawen Ouyang, Zhen Wu, Xinyu Dai, Shujian Huang and Jiajun Chen 626

Pan More Gold from the Sand: Refining Open-domain Dialogue Training with Noisy Self-Retrieval Generation
Yihe Wang, Yitong Li, Yasheng Wang, Fei Mi, Pingyi Zhou, Xin Wang, Jin Liu, Xin Jiang and Qun Liu 636

MulZDG: Multilingual Code-Switching Framework for Zero-shot Dialogue Generation
Yongkang Liu, Shi Feng, Daling Wang and Yifei Zhang 648

Target-Guided Open-Domain Conversation Planning
Yosuke Kishinami, Reina Akama, Shiki Sato, Ryoko Tokuhisa, Jun Suzuki and Kentaro Inui .. 660

Does GPT-3 Generate Empathetic Dialogues? A Novel In-Context Example Selection Method and Automatic Evaluation Metric for Empathetic Dialogue Generation
Young-Jun Lee, Chae-Gyun Lim and Ho-Jin Choi 669

DialogueEIN: Emotion Interaction Network for Dialogue Affective Analysis
Yuchen Liu, Jinming Zhao, Jingwen Hu, Ruichen Li and Qin Jin 684

Towards Enhancing Health Coaching Dialogue in Low-Resource Settings
Yue Zhou, Barbara Di Eugenio, Brian Ziebart, Lisa Sharp, Bing Liu, Ben Gerber, Nikolaos Agadakos and Shweta Yadav 694

Generalized Intent Discovery: Learning from Open World Dialogue System
Yutao Mou, Keqing He, Yanan Wu, Pei Wang, Jingang Wang, Wei Wu, Yi Huang, Junlan Feng and Weiran Xu 707

DialMed: A Dataset for Dialogue-based Medication Recommendation
Zhenfeng He, Yuqiang Han, Zhenqiu Ouyang, Wei Gao, Hongxu Chen, Guandong Xu and Jian Wu 721

Speaker Clustering in Textual Dialogue with Pairwise Utterance Relation and Cross-corpus Dialogue Act Supervision
Zhihua Su and Qiang Zhou 734

TopKG: Target-oriented Dialog via Global Planning on Knowledge Graph
Zhitong Yang, Bo Wang, Jinfeng Zhou, Yue Tan, Dongming Zhao, Kun Huang, Ruifang He and Yuexian Hou 745

Extractive Summarisation for German-language Data: A Text-level Approach with Discourse Features
Freya Hewett and Manfred Stede 756

End-to-End Neural Bridging Resolution
Hideo Kobayashi, Yufang Hou and Vincent Ng 766

Investigating the Performance of Transformer-Based NLI Models on Presuppositional Inferences
Jad Kabbara and Jackie Chi Kit Cheung 779

Re-Examining FactBank: Predicting the Author's Presentation of Factuality
John Murzaku, Peter Zeng, Magdalena Markowska and Owen Rambow 786

The Role of Context and Uncertainty in Shallow Discourse Parsing
Katherine Atwell, Remi Choi, Junyi Jessy Li and Malihe Alikhani 797

Improving Commonsense Contingent Reasoning by Pseudo-data and Its Application to the Related Tasks
Kazumasa Omura and Sadao Kurohashi 812

A Survey in Automatic Irony Processing: Linguistic, Cognitive, and Multi-X Perspectives
Qingcheng Zeng and An-Ran Li 824

Towards Identifying Alternative-Lexicalization Signals of Discourse Relations
René Knaebel and Manfred Stede 837

Topicalization in Language Models: A Case Study on Japanese
Riki Fujihara, Tatsuki Kuribayashi, Kaori Abe, Ryoko Tokuhisa and Kentaro Inui 851

"No, They Did Not": Dialogue Response Dynamics in Pre-trained Language Models
Sanghee J. Kim, Lang Yu and Allyson Ettinger 863

New or Old? Exploring How Pre-Trained Language Models Represent Discourse Entities
Sharid Loáiciga, Anne Beyer and David Schlangen 875

Dialo-AP: A Dependency Parsing Based Argument Parser for Dialogues
Sougata Saha, Souvik Das and Rohini K. Srihari 887

ConnPrompt: Connective-cloze Prompt Learning for Implicit Discourse Relation Recognition
Wei Xiang, Zhenglin Wang, Lu Dai and Bang Wang 902

A Distance-Aware Multi-Task Framework for Conversational Discourse Parsing
Yaxin Fan, Peifeng Li, Fang Kong and Qiaoming Zhu 912

Linguistically Motivated Features for Classifying Shorter Text into Fiction and Non-Fiction Genre
Arman Kazmi, Sidharth Ranjan, Arpit Sharma and Rajakrishnan Rajkumar 922

Semantic Sentence Matching via Interacting Syntax Graphs
Chen Xu, Jun Xu, Zhenhua Dong and Ji-Rong Wen 938

Hierarchical Information Matters: Text Classification via Tree Based Graph Neural Network
Chong Zhang, He Zhu, Xingyu Peng, Junran Wu and Ke Xu 950

SelfMix: Robust Learning against Textual Label Noise with Self-Mixup Training
Dan Qiao, Chenchen Dai, Yuyang Ding, Juntao Li, Qiang Chen, Wenliang Chen and Min Zhang 960

Community Topic: Topic Model Inference by Consecutive Word Community Discovery
Eric Austin, Osmar R. Zaïane and Christine Largeron 971

Where to Attack: A Dynamic Locator Model for Backdoor Attack in Text Classifications
Heng-yang Lu, Chenyou Fan, Jun Yang, Cong Hu, Wei Fang and Xiao-jun Wu 984

Locally Distributed Activation Vectors for Guided Feature Attribution
Housam K. B. Bashier, Mi-Young Kim and Randy Goebel 994

Addressing Leakage in Self-Supervised Contextualized Code Retrieval
Johannes Villmow, Viola Campos, Adrian Ulges and Ulrich Schwanecke 1006

A Domain Knowledge Enhanced Pre-Trained Language Model for Vertical Search: Case Study on Medicinal Products
Kesong Liu, Jianhui Jiang and Feifei Lyu 1014

CONCRETE: Improving Cross-lingual Fact-checking with Cross-lingual Retrieval
Kung-Hsiang Huang, ChengXiang Zhai and Heng Ji 1024

E-VarM: Enhanced Variational Word Masks to Improve the Interpretability of Text Classification Models
Ling Ge, ChunMing Hu, Guanghui Ma, Junshuang Wu, Junfan Chen, JiHong Liu, Hong Zhang, Wenyi Qin and Richong Zhang 1036

Attribute Injection for Pretrained Language Models: A New Benchmark and an Efficient Method
Reinald Kim Amplayo, Kang Min Yoo and Sang-Woo Lee 1051

Towards Robust Neural Retrieval with Source Domain Synthetic Pre-Finetuning
Revanth Gangi Reddy, Vikas Yadav, Md Arafat Sultan, Martin Franz, Vittorio Castelli, Heng Ji and Avirup Sil 1065

Parameter-Efficient Neural Reranking for Cross-Lingual and Multilingual Retrieval
Robert Litschko, Ivan Vulić and Goran Glavaš 1071

LIME: Weakly-Supervised Text Classification without Seeds
Seongmin Park and Jihwa Lee 1083

Multi-Stage Framework with Refinement Based Point Set Registration for Unsupervised Bi-Lingual Word Alignment
Silviu Vlad Oprea, Sourav Dutta and Haytham Assem 1089

EM-PERSONA: EMotion-assisted Deep Neural Framework for PERSONAlity Subtyping from Suicide Notes
Soumitra Ghosh, Dhirendra Kumar Maurya, Asif Ekbal and Pushpak Bhattacharyya 1098

Dense Template Retrieval for Customer Support
Tiago Mesquita, Bruno Martins and Mariana Almeida 1106

Exploring Label Hierarchy in a Generative Way for Hierarchical Text Classification
Wei Huang, Chen Liu, Bo Xiao, Yihua Zhao, Zhaoming Pan, Zhimin Zhang, Xinyun Yang and Guiquan Liu 1116

MuSeCLIR: A Multiple Senses and Cross-lingual Information Retrieval Dataset
Wing Yan Li, Julie Weeds and David Weir 1128

Complicate Then Simplify: A Novel Way to Explore Pre-trained Models for Text Classification
Xu Zhang, Zejie Liu, Yanzheng Xiang and Deyu Zhou 1136

Adaptive Feature Discrimination and Denoising for Asymmetric Text Matching
Yan Li, Chenliang Li and Junjun Guo 1146

Rethinking Data Augmentation in Text-to-text Paradigm
Yanan Chen and Yang Liu 1157

ConTextING: Granting Document-Wise Contextual Embeddings to Graph Neural Networks for Inductive Text Classification
Yen-Hao Huang, Yi-Hsin Chen and Yi-Shin Chen 1163

Virtual Knowledge Graph Construction for Zero-Shot Domain-Specific Document Retrieval
Yeon Seonwoo, Seunghyun Yoon, Franck Dernoncourt, Trung Bui and Alice Oh 1169

MICO: Selective Search with Mutual Information Co-training
Zhanyu Wang, Xiao Zhang, Hyokun Yun, Choon Hui Teo and Trishul Chilimbi 1179

DPTDR: Deep Prompt Tuning for Dense Passage Retrieval
Zhengyang Tang, Benyou Wang and Ting Yao 1193

BERT-Flow-VAE: A Weakly-supervised Model for Multi-Label Text Classification
Ziwen Liu, Josep Grau-Bove and Scott Allan Allan Orr 1203

Welcome to the Modern World of Pronouns: Identity-Inclusive Natural Language Processing beyond Gender
Anne Lauscher, Archie Crowley and Dirk Hovy 1221

Threat Scenarios and Best Practices to Detect Neural Fake News
Artidoro Pagnoni, Martin Graciarena and Yulia Tsvetkov 1233

From Polarity to Intensity: Mining Morality from Semantic Space
Chunxu Zhao, Pengyuan Liu and Dong Yu 1250

SOS: Systematic Offensive Stereotyping Bias in Word Embeddings
Fatma Elsafoury, Steve R. Wilson, Stamos Katsigiannis and Naeem Ramzan 1263

Bigger Data or Fairer Data? Augmenting BERT via Active Sampling for Educational Text Classification
Lele Sha, Yuheng Li, Dragan Gasevic and Guanliang Chen 1275

Debiasing Word Embeddings with Nonlinear Geometry
Lu Cheng, Nayoung Kim and Huan Liu 1286

Debiasing Isn't Enough! – on the Effectiveness of Debiasing MLMs and Their Social Biases in Downstream Tasks
Masahiro Kaneko, Danushka Bollegala and Naoaki Okazaki 1299

Quantifying Bias from Decoding Techniques in Natural Language Generation
Mayukh Das and Wolf Tilo Balke 1311

A Study of Implicit Bias in Pretrained Language Models against People with Disabilities
Pranav Narayanan Venkit, Mukund Srinath and Shomir Wilson 1324

Social Norms-Grounded Machine Ethics in Complex Narrative Situation
Tao Shen, Xiubo Geng and Daxin Jiang 1333

Bias at a Second Glance: A Deep Dive into Bias for German Educational Peer-Review Data Modeling
Thiemo Wambsganss, Vinitra Swamy, Roman Rietsche and Tanja Käser 1344

Dynamic Relevance Graph Network for Knowledge-Aware Question Answering
Chen Zheng and Parisa Kordjamshidi 1357

SISER: Semantic-Infused Selective Graph Reasoning for Fact Verification
Eunhwan Park, Jong-Hyeon Lee, DongHyeon Jeon, Seonhoon Kim, Inho Kang and Seung-Hoon Na 1367

Answering Numerical Reasoning Questions in Table-Text Hybrid Contents with Graph-based Encoder and Tree-based Decoder
Fangyu Lei, Shizhu He, Xiang Li, Jun Zhao and Kang Liu 1379

Perform like an Engine: A Closed-Loop Neural-Symbolic Learning Framework for Knowledge Graph Inference
Guanglin Niu, Bo Li, Yongfei Zhang and Shiliang Pu 1391

Table-based Fact Verification with Self-labeled Keypoint Alignment
Guangzhen Zhao and Peng Yang 1401

IMCI: Integrate Multi-view Contextual Information for Fact Extraction and Verification
Hao Wang, Yangguang Li, Zhen Huang and Yong Dou 1412

Prompt Combines Paraphrase: Teaching Pre-trained Models to Understand Rare Biomedical Words
Haochun Wang, Chi Liu, Nuwa Xi, Sendong Zhao, Meizhi Ju, Shiwei Zhang, Ziheng Zhang, Yefeng Zheng, Bing Qin and Ting Liu 1422

Self-Supervised Intermediate Fine-Tuning of Biomedical Language Models for Interpreting Patient Case Descriptions
Israa Alghanmi, Luis Espinosa-Anke and Steven Schockaert 1432

Evaluating and Mitigating Inherent Linguistic Bias of African American English through Inference
Jamell Dacon, Haochen Liu and Jiliang Tang 1442

Can We Guide a Multi-Hop Reasoning Language Model to Incrementally Learn at Each Single-Hop?
Jesus Lovon-Melgarejo, Jose G. Moreno, Romaric Besançon, Olivier Ferret and Lynda Tamine 1455

Modeling Hierarchical Reasoning Chains by Linking Discourse Units and Key Phrases for Reading Comprehension
Jialin Chen, Zhuosheng Zhang and Hai Zhao 1467

Hierarchical Representation-based Dynamic Reasoning Network for Biomedical Question Answering
Jianguo Mao, Jiyuan Zhang, Zengfeng Zeng, Weihua Peng, Wenbin Jiang, Xiangdong Wang, Hong Liu and Yajuan Lyu 1480

ArT: All-round Thinker for Unsupervised Commonsense Question Answering
Jiawei Wang and Hai Zhao 1490

Teaching Neural Module Networks to Do Arithmetic
Jiayi Chen, Xiao-Yu Guo, Yuan-Fang Li and Gholamreza Haffari 1502

An Augmented Benchmark Dataset for Geometric Question Answering through Dual Parallel Text Encoding
Jie Cao and Jing Xiao 1511

Competence-based Question Generation
Jingxuan Tu, Kyeongmin Rim and James Pustejovsky 1521

Coalescing Global and Local Information for Procedural Text Understanding
Kaixin Ma, Filip Ilievski, Jonathan Francis, Eric Nyberg and Alessandro Oltramari 1534

Original Content Is All You Need! an Empirical Study on Leveraging Answer Summary for WikiHowQA Answer Selection Task
Liang Wen, Juan Li, Houfeng Wang, Yingwei Luo, Xiaolin Wang, Xiaodong Zhang, Zhicong Cheng and Dawei Yin 1546

Case-Based Abductive Natural Language Inference
Marco Valentino, Mokanarangan Thayaparan and André Freitas 1556

Semantic Structure Based Query Graph Prediction for Question Answering over Knowledge Graph
Mingchen Li and Shihao Ji 1569

Repo4QA: Answering Coding Questions via Dense Retrieval on GitHub Repositories
Minyu Chen, Guoqiang Li, Chen Ma, Jingyang LI and Hongfei Fu 1580

Addressing Limitations of Encoder-Decoder Based Approach to Text-to-SQL
Octavian Popescu, Irene Manotas, Ngoc Phuoc An Vo, Hangu Yeo, Elahe Khorashani and Vadim Sheinin 1593

Mintaka: A Complex, Natural, and Multilingual Dataset for End-to-End Question Answering
Priyanka Sen, Alham Fikri Aji and Amir Saffari 1604

Can Edge Probing Tests Reveal Linguistic Knowledge in QA Models?
Sagnik Ray Choudhury, Nikita Bhutani and Isabelle Augenstein 1620

Conversational QA Dataset Generation with Answer Revision
Seonjeong Hwang and Gary Geunbae Lee 1636

DABERT: Dual Attention Enhanced BERT for Semantic Matching
Sirui Wang, Di Liang, Jian Song, Yuntao Li and Wei Wu 1645

Locate Then Ask: Interpretable Stepwise Reasoning for Multi-hop Question Answering
Siyuan Wang, Zhongyu Wei, Zhihao Fan, Qi Zhang and Xuanjing Huang 1655

Less Is Better: Recovering Intended-Feature Subspace to Robustify NLU Models
Ting Wu and Tao Gui 1666

CORN: Co-Reasoning Network for Commonsense Question Answering
Xin Guan, Biwei Cao, Qingqing Gao, Zheng Yin, Bo Liu and Jiuxin Cao 1677

Logical Form Generation via Multi-task Learning for Complex Question Answering over Knowledge Bases
Xixin Hu, Xuan Wu, Yiheng Shu and Yuzhong Qu 1687

CMQA: A Dataset of Conditional Question Answering with Multiple-Span Answers
Yiming Ju, Weikang Wang, Yuanzhe Zhang, Suncong Zheng, Kang Liu and Jun Zhao 1697

To What Extent Do Natural Language Understanding Datasets Correlate to Logical Reasoning? A Method for Diagnosing Logical Reasoning.
Yitian Li, Jidong Tian, Wenqing Chen, Caoyun Fan, Hao He and Yaohui Jin 1708

ArcaneQA: Dynamic Program Induction and Contextualized Encoding for Knowledge Base Question Answering
Yu Gu and Yu Su 1718

Unsupervised Question Answering via Answer Diversifying
Yuxiang Nie, Heyan Huang, Zewen Chi and Xian-Ling Mao 1732

Weakly Supervised Formula Learner for Solving Mathematical Problems
Yuxuan Wu and Hideki Nakayama 1743

Reducing Spurious Correlations for Answer Selection by Feature Decorrelation and Language Debiasing
Zeyi Zhong, Min Yang and Ruifeng Xu 1753

Understanding and Improving Zero-shot Multi-hop Reasoning in Generative Question Answering
Zhengbao Jiang, Jun Araki, Haibo Ding and Graham Neubig 1765

Domain Adaptation for Question Answering via Question Classification
Zhenrui Yue, Huimin Zeng, Ziyi Kou, Lanyu Shang and Dong Wang 1776

Prompt-based Conservation Learning for Multi-hop Question Answering
Zhenyun Deng, Yonghua Zhu, Yang Chen, Qianqian Qi, Michael Witbrock and Patricia Riddle 1791

GLAF: Global-to-Local Aggregation and Fission Network for Semantic Level Fact Verification
Zhiyuan Ma, Jianjun Li, Guohui Li and Yongjing Cheng 1801

Exploiting Hybrid Semantics of Relation Paths for Multi-hop Question Answering over Knowledge Graphs
Zile Qiao, Wei Ye, Tong Zhang, Tong Mo, Weiping Li and Shikun Zhang 1813

Adaptive Threshold Selective Self-Attention for Chinese NER
Biao Hu, Zhen Huang, Minghao Hu, Ziwen Zhang and Yong Dou 1823

Cluster-aware Pseudo-Labeling for Supervised Open Relation Extraction
Bin Duan, Shusen Wang, Xingxian Liu and Yajing Xu 1834

Few-shot Named Entity Recognition with Entity-level Prototypical Network Enhanced by Dispersedly Distributed Prototypes
Bin Ji, Shasha Li, Shaoduo Gan, Jie Yu, Jun Ma, Huijun Liu and Jing Yang 1842

Different Data, Different Modalities! Reinforced Data Splitting for Effective Multimodal Information Extraction from Social Media Posts
Bo Xu, Shizhou Huang, Ming Du, Hongya Wang, Hui Song, Chaofeng Sha and Yanghua Xiao 1855

Augmentation, Retrieval, Generation: Event Sequence Prediction with a Three-Stage Sequence-to-Sequence Approach
Bo Zhou, Chenhao Wang, Yubo Chen, Kang Liu, Jun Zhao, Jiexin Xu, Xiaojian Jiang and Qiuxia Li 1865

Generating Temporally-ordered Event Sequences via Event Optimal Transport
Bo Zhou, Yubo Chen, Kang Liu, Jun Zhao, Jiexin Xu, Xiaojian Jiang and Qiuxia Li 1875

Improving Continual Relation Extraction through Prototypical Contrastive Learning
Chengwei Hu, Deqing Yang, Haoliang Jin, Zhen Chen and Yanghua Xiao 1885

Prompt-based Text Entailment for Low-Resource Named Entity Recognition
Dongfang Li, Baotian Hu and Qingcai Chen 1896

Key Mention Pairs Guided Document-Level Relation Extraction
Feng Jiang, Jianwei Niu, Shasha Mo and Shengda Fan 1904

A Hybrid Model of Classification and Generation for Spatial Relation Extraction
Feng Wang, Peifeng Li and Qiaoming Zhu 1915

Mining Health-related Cause-Effect Statements with High Precision at Large Scale
Ferdinand Schlatt, Dieter Bettin, Matthias Hagen, Benno Stein and Martin Potthast 1925

Find the Funding: Entity Linking with Incomplete Funding Knowledge Bases
Gizem Aydin, Seyed Amin Tabatabaei, George Tsatsaronis and Faegheh Hasibi 1937

KiPT: Knowledge-injected Prompt Tuning for Event Detection
Haochen Li, Tong Mo, Hongcheng Fan, Jingkun Wang, Jiaxi Wang, Fuhao Zhang and Weiping Li 1943

OneEE: A One-Stage Framework for Fast Overlapping and Nested Event Extraction
Hu Cao, Jingye Li, Fangfang Su, Fei Li, Hao Fei, Shengqiong Wu, Bobo Li, Liang Zhao and Donghong Ji 1953

Joint Language Semantic and Structure Embedding for Knowledge Graph Completion
Jianhao Shen, Chenguang Wang, Linyuan Gong and Dawn Song 1965

Event Detection with Dual Relational Graph Attention Networks
Jiaxin Mi, Po Hu and Peng Li 1979

A Multi-Format Transfer Learning Model for Event Argument Extraction via Variational Information Bottleneck
Jie Zhou, Qi Zhang, Qin Chen, Qi Zhang, Liang He and Xuanjing Huang 1990

RSGT: Relational Structure Guided Temporal Relation Extraction
Jie Zhou, Shenpo Dong, Hongkui Tu, Xiaodong Wang and Yong Dou 2001

Learning Hierarchy-Aware Quaternion Knowledge Graph Embeddings with Representing Relations as 3D Rotations
Jinfa Yang, Xianghua Ying, Yongjie Shi, Xin Tong, Ruibin Wang, Taiyan Chen and Bowei Xing 2011

Two Languages Are Better than One: Bilingual Enhancement for Chinese Named Entity Recognition
Jinzhong Ning, Zhihao Yang, Zhizheng Wang, Yuanyuan Sun, Hongfei Lin and Jian Wang . . . 2024

Read Extensively, Focus Smartly: A Cross-document Semantic Enhancement Method for Visual Documents NER
Jun Zhao, Xin Zhao, WenYu Zhan, Tao Gui, Qi Zhang, Liang Qiao, Zhanzhan Cheng and Shiliang Pu 2034

STAD: Self-Training with Ambiguous Data for Low-Resource Relation Extraction
Junjie Yu, Xing Wang, Jiangjiang Zhao, Chunjie Yang and Wenliang Chen 2044

Flat Multi-modal Interaction Transformer for Named Entity Recognition
Junyu Lu, Dixiang Zhang, Jiaxing Zhang and Pingjian Zhang 2055

MetaSLRCL: A Self-Adaptive Learning Rate and Curriculum Learning Based Framework for Few-Shot Text Classification
Kailin Zhao, Xiaolong Jin, Saiping Guan, Jiafeng Guo and Xueqi Cheng 2065

A Simple Temporal Information Matching Mechanism for Entity Alignment between Temporal Knowledge Graphs
Li Cai, Xin Mao, Meirong Ma, Hao Yuan, Jianchao Zhu and Man Lan 2075

DCT-Centered Temporal Relation Extraction
Liang Wang, Peifeng Li and Sheng Xu 2087

Document-level Biomedical Relation Extraction Based on Multi-Dimensional Fusion Information and Multi-Granularity Logical Reasoning
Lishuang Li, Ruiyuan Lian, Hongbin Lu and Jingyao Tang 2098

Simple Yet Powerful: An Overlooked Architecture for Nested Named Entity Recognition
Matias Rojas, Felipe Bravo-Marquez and Jocelyn Dunstan 2108

ERGO: Event Relational Graph Transformer for Document-level Event Causality Identification
Meiqi Chen, Yixin Cao, Kunquan Deng, Mukai Li, Kun Wang, Jing Shao and Yan Zhang 2118

DRK: Discriminative Rule-based Knowledge for Relieving Prediction Confusions in Few-shot Relation Extraction
Mengru Wang, Jianming Zheng, Fei Cai, Taihua Shao and Honghui Chen 2129

DocQueryNet: Value Retrieval with Arbitrary Queries for Form-like Documents
Mingfei Gao, Le Xue, Chetan Ramaiah, Chen Xing, Ran Xu and Caiming Xiong 2141

DoSEA: A Domain-specific Entity-aware Framework for Cross-Domain Named Entity Recogition
Minghao Tang, Peng Zhang, Yongquan He, Yongxiu Xu, Chengpeng Chao and Hongbo Xu . . 2147

Incremental Prompting: Episodic Memory Prompt for Lifelong Event Detection
Minqian Liu, Shiyu Chang and Lifu Huang 2157

Recent Advances in Text-to-SQL: A Survey of What We Have and What We Expect
Naihao Deng, Yulong Chen and Yue Zhang 2166

An MRC Framework for Semantic Role Labeling
Nan Wang, Jiwei Li, Yuxian Meng, Xiaofei Sun, Han Qiu, Ziyao Wang, Guoyin Wang and Jun He 2188

PCBERT: Parent and Child BERT for Chinese Few-shot NER
Peichao Lai, Feiyang Ye, Lin Zhang, Zhiwei Chen, Yanggeng Fu, Yingjie Wu and Yilei Wang2199

Label Smoothing for Text Mining
Peiyang Liu, Xiangyu Xi, Wei Ye and Shikun Zhang 2210

Diverse Multi-Answer Retrieval with Determinantal Point Processes
Poojitha Nandigam, Nikhil Rayaprolu and Manish Shrivastava 2220

Improving Deep Embedded Clustering via Learning Cluster-level Representations
Qing Yin, Zhihua Wang, Yunya Song, Yida Xu, Shuai Niu, Liang Bai, Yike Guo and Xian Yang 2226

Decoupling Mixture-of-Graphs: Unseen Relational Learning for Knowledge Graph Completion by Fusing Ontology and Textual Experts
Ran Song, Shizhu He, Suncong Zheng, Shengxiang Gao, Kang Liu, Zhengtao YU and Jun Zhao 2237

CETA: A Consensus Enhanced Training Approach for Denoising in Distantly Supervised Relation Extraction
Ruri Liu, Shasha Mo, Jianwei Niu and Shengda Fan 2247

MedDistant19: Towards an Accurate Benchmark for Broad-Coverage Biomedical Relation Extraction
Saadullah Amin, Pasquale Minervini, David Chang, Pontus Stenetorp and Guenter Neumann 2259

Decorrelate Irrelevant, Purify Relevant: Overcome Textual Spurious Correlations from a Feature Perspective
Shihan Dou, Rui Zheng, Ting Wu, SongYang Gao, Junjie Shan, Qi Zhang, Yueming Wu and Xuanjing Huang 2278

Event Causality Identification via Derivative Prompt Joint Learning
Shirong Shen, Heng Zhou, Tongtong Wu and Guilin Qi 2288

Event Causality Extraction with Event Argument Correlations
Shiyao Cui, Jiawei Sheng, Xin Cong, Quangang Li, Tingwen Liu and Jinqiao Shi 2300

SCL-RAI: Span-based Contrastive Learning with Retrieval Augmented Inference for Unlabeled Entity Problem in NER
Shuzheng Si, Shuang Zeng, Jiaxing Lin and Baobao Chang 2313

A Relation Extraction Dataset for Knowledge Extraction from Web Tables
Siffi Singh, Alham Fikri Aji, Gaurav Singh and Christos Christodoulopoulos 2319

Automatic Keyphrase Generation by Incorporating Dual Copy Mechanisms in Sequence-to-Sequence Learning
Siyu Wang, Jianhui Jiang, Yao Huang and Yin Wang 2328

Dependency-aware Prototype Learning for Few-shot Relation Classification
Tianshu Yu, Min Yang and Xiaoyan Zhao 2339

MECI: A Multilingual Dataset for Event Causality Identification
Viet Dac Lai, Amir Pouran Ben Veyseh, Minh Van Nguyen, Franck Dernoncourt and Thien Huu Nguyen 2346

Method Entity Extraction from Biomedical Texts
Waqar Bin Kalim and Robert E. Mercer 2357

Optimal Partial Transport Based Sentence Selection for Long-form Document Matching
Weijie Yu, Liang Pang, Jun Xu, Bing Su, Zhenhua Dong and Ji-Rong Wen 2363

LightNER: A Lightweight Tuning Paradigm for Low-resource NER via Pluggable Prompting
Xiang Chen, Lei Li, Shumin Deng, Chuanqi Tan, Changliang Xu, Fei Huang, Luo Si, Huajun Chen and Ningyu Zhang 2374

Cross-modal Contrastive Attention Model for Medical Report Generation
Xiao Song, Xiaodan Zhang, Junzhong Ji, Ying Liu and Pengxu Wei 2388

Domain-Specific NER via Retrieving Correlated Samples
Xin Zhang, Yong Jiang, Xiaobin Wang, Xuming Hu, Yueheng Sun, Pengjun Xie and Meishan Zhang 2398

Type-enriched Hierarchical Contrastive Strategy for Fine-Grained Entity Typing
Xinyu Zuo, Haijin Liang, Ning Jing, Shuang Zeng, Zhou Fang and Yu Luo 2405

Document-Level Relation Extraction via Pair-Aware and Entity-Enhanced Representation Learning
Xiusheng Huang, Hang Yang, Yubo Chen, Jun Zhao, Kang Liu, Weijian Sun and Zuyu Zhao . 2418

Improving Zero-Shot Entity Linking Candidate Generation with Ultra-Fine Entity Type Information
Xuhui Sui, Ying Zhang, Kehui Song, Baohang Zhou, Guoqing Zhao, Xin Wei and Xiaojie Yuan 2429

CofeNet: Context and Former-Label Enhanced Net for Complicated Quotation Extraction
Yequan Wang, Xiang Li, Aixin Sun, Xuying Meng, Huaming Liao and Jiafeng Guo 2438

Supporting Medical Relation Extraction via Causality-Pruned Semantic Dependency Forest
Yifan Jin, Jiangmeng Li, Zheng Lian, Chengbo Jiao and Xiaohui Hu 2450

Aspect-based Sentiment Analysis as Machine Reading Comprehension
Yifei Yang and Hai Zhao 2461

Nested Named Entity Recognition as Corpus Aware Holistic Structure Parsing
Yifei Yang, Zuchao Li and Hai Zhao 2472

DESED: Dialogue-based Explanation for Sentence-level Event Detection
Yinyi Wei, Shuaipeng Liu, Jianwei Lv, Xiangyu Xi, Hailei Yan, Wei Ye, Tong Mo, Fan Yang and Guanglu Wan 2483

Data Augmentation for Few-Shot Knowledge Graph Completion from Hierarchical Perspective
Yuanzhou Yao, Zhao Zhang, Yongjun Xu and Chao Li 2494

CLIO: Role-interactive Multi-event Head Attention Network for Document-level Event Extraction
Yubing Ren, Yanan Cao, Fang Fang, Ping Guo, Zheng Lin, Wei Ma and Yi Liu 2504

COPNER: Contrastive Learning with Prompt Guiding for Few-shot Named Entity Recognition
Yucheng Huang, Kai He, Yige Wang, Xianli Zhang, Tieliang Gong, Rui Mao and Chen Li . . . 2515

Few Clean Instances Help Denoising Distant Supervision
Yufang Liu, Ziyin Huang, Yijun Wang, Changzhi Sun, Man Lan, Yuanbin Wu, Xiaofeng Mou and Ding Wang 2528

SEE-Few: Seed, Expand and Entail for Few-shot Named Entity Recognition
Zeng Yang, Linhai Zhang and Deyu Zhou 2540

Ruleformer: Context-aware Rule Mining over Knowledge Graph
Zezhong Xu, Peng Ye, Hui Chen, Meng Zhao, Huajun Chen and Wen Zhang 2551

Are People Located in the Places They Mention in Their Tweets? A Multimodal Approach
Zhaomin Xiao and Eduardo Blanco 2561

Multi-modal Contrastive Representation Learning for Entity Alignment
Zhenxi Lin, Ziheng Zhang, Meng Wang, Yinghui Shi, Xian Wu and Yefeng Zheng 2572

Nonparametric Forest-Structured Neural Topic Modeling
Zhihong Zhang, Xuewen Zhang and Yanghui Rao 2585

KGE-CL: Contrastive Learning of Tensor Decomposition Based Knowledge Graph Embeddings
Zhiping Luo, Wentao Xu, Weiqing Liu, Jiang Bian, Jian Yin and Tie-Yan Liu 2598

A Coarse-to-fine Cascaded Evidence-Distillation Neural Network for Explainable Fake News Detection
Zhiwei Yang, Jing Ma, Hechang Chen, Hongzhan Lin, Ziyang Luo and Yi Chang 2608

Document-level Event Factuality Identification via Machine Reading Comprehension Frameworks with Transfer Learning
Zhong Qian, Heng Zhang, Peifeng Li, Qiaoming Zhu and Guodong Zhou 2622

Unregulated Chinese-to-English Data Expansion Does NOT Work for Neural Event Detection
Zhongqiu Li, Yu Hong, Jie Wang, Shiming He, Jianmin Yao and Guodong Zhou 2633

Finding Influential Instances for Distantly Supervised Relation Extraction
Zifeng Wang, Rui Wen, Xi Chen, Shao-Lun Huang, Ningyu Zhang and Yefeng Zheng 2639

A Simple Model for Distantly Supervised Relation Extraction
Ziqin Rao, Fangxiang Feng, Ruifan Li and Xiaojie Wang 2651

Augmenting Legal Judgment Prediction with Contrastive Case Relations
Dugang Liu, Weihao Du, Lei Li, Weike Pan and Zhong Ming 2658

Constrained Regeneration for Cross-Lingual Query-Focused Extractive Summarization
Elsbeth Turcan, David Wan, Faisal Ladhak, Petra Galuscakova, Sukanta Sen, Svetlana Tchistiakova, Weijia Xu, Marine Carpuat, Kenneth Heafield, Douglas Oard and Kathleen McKeown 2668

Programmable Annotation with Diversed Heuristics and Data Denoising
Ernie Chang, Alex Marin and Vera Demberg 2681

Text-to-Text Extraction and Verbalization of Biomedical Event Graphs
Giacomo Frisoni, Gianluca Moro and Lorenzo Balzani 2692

Multimodal Semi-supervised Learning for Disaster Tweet Classification
Iustin Sirbu, Tiberiu Sosea, Cornelia Caragea, Doina Caragea and Traian Rebedea 2711

Automated Essay Scoring via Pairwise Contrastive Regression
Jiayi Xie, Kaiwei Cai, Li Kong, Junsheng Zhou and Weiguang Qu 2724

Medical Question Understanding and Answering with Knowledge Grounding and Semantic Self-Supervision
Khalil Mrini, Harpreet Singh, Franck Dernoncourt, Seunghyun Yoon, Trung Bui, Walter W. Chang, Emilia Farcas and Ndapa Nakashole 2734

A Progressive Framework for Role-Aware Rumor Resolution
Lei Chen, Guanying Li, Zhongyu Wei, Yang Yang, Baohua Zhou, Qi Zhang and Xuanjing Huang 2748

Uncertainty-aware Propagation Structure Reconstruction for Fake News Detection
Lingwei Wei, Dou Hu, Wei Zhou and Songlin Hu 2759

A Unified Propagation Forest-based Framework for Fake News Detection
Lingwei Wei, Dou Hu, Yantong Lai, Wei Zhou and Songlin Hu 2769

CLoSE: Contrastive Learning of Subframe Embeddings for Political Bias Classification of News Media
Michelle YoungJin Kim and Kristen Marie Johnson 2780

Grammatical Error Correction: Are We There Yet?
Muhammad Reza Qorib and Hwee Tou Ng 2794

CXR Data Annotation and Classification with Pre-trained Language Models
Nina Zhou, Ai Ti Aw, zhuo han Liu, Cher heng Tan, Yonghan Ting, wen xiang chen and Jordan sim zheng Ting 2801

uChecker: Masked Pretrained Language Models as Unsupervised Chinese Spelling Checkers
Piji Li 2812

Boosting Deep CTR Prediction with a Plug-and-Play Pre-trainer for News Recommendation
Qijiong Liu, Jieming Zhu, Quanyu Dai and Xiaoming Wu 2823

Improving Fake News Detection of Influential Domain via Domain- and Instance-Level Transfer
Qiong Nan, Danding Wang, Yongchun Zhu, Qiang Sheng, Yuhui Shi, Juan Cao and Jintao Li 2834

Student Surpasses Teacher: Imitation Attack for Black-Box NLP APIs
Qiongkai Xu, Xuanli He, Lingjuan Lyu, Lizhen Qu and Gholamreza Haffari 2849

Combining Compressions for Multiplicative Size Scaling on Natural Language Tasks
Rajiv Movva, Jinhao Lei, Shayne Longpre, Ajay Gupta and Chris DuBois 2861

PlugAT: A Plug and Play Module to Defend against Textual Adversarial Attack
Rui Zheng, Rong Bao, Qin Liu, Tao Gui, Qi Zhang, Xuanjing Huang, Rui Xie and Wei Wu . . 2873

Automatic ICD Coding Exploiting Discourse Structure and Reconciled Code Embeddings
Shurui Zhang, Bozheng Zhang, Fuxin Zhang, Bo Sang and Wanchun Yang 2883

Towards Summarizing Healthcare Questions in Low-Resource Setting
Shweta Yadav and Cornelia Caragea 2892

Doc-GCN: Heterogeneous Graph Convolutional Networks for Document Layout Analysis
Siwen Luo, Yihao Ding, Siqu Long, Josiah Poon and Soyeon Caren Han 2906

Analytic Automated Essay Scoring Based on Deep Neural Networks Integrating Multidimensional Item Response Theory
Takumi Shibata and Masaki Uto 2917

DP-Rewrite: Towards Reproducibility and Transparency in Differentially Private Text Rewriting
Timour Igamberdiev, Thomas Arnold and Ivan Habernal 2927

Harnessing Abstractive Summarization for Fact-Checked Claim Detection
Varad Bhatnagar, Diptesh Kanojia and Kameswari Chebrolu 2934

Learning to Generate Explanation from e-Hospital Services for Medical Suggestion
Wei-Lin Chen, An-Zi Yen, Hen-Hsen Huang and Hsin-Hsi Chen 2946

DeltaNet: Conditional Medical Report Generation for COVID-19 Diagnosis
Xian Wu, Shuxin Yang, Zhaopeng Qiu, Shen Ge, Yangtian Yan, Xingwang Wu, Yefeng Zheng, S. Kevin Zhou and Li Xiao 2952

MCS: An In-battle Commentary System for MOBA Games
Xiaofeng Qi, Chao Li, Zhongping Liang, Jigang Liu, Cheng Zhang, Yuanxin Wei, Lin Yuan, Guang Yang, Lanxiao Huang and Min Li 2962

A Two Stage Adaptation Framework for Frame Detection via Prompt Learning
Xinyi Mou, Zhongyu Wei, Changjian Jiang and Jiajie Peng 2968

Summarizing Patients' Problems from Hospital Progress Notes Using Pre-trained Sequence-to-Sequence Models
Yanjun Gao, Dmitriy Dligach, Timothy Miller, Dongfang Xu, Matthew M. M. Churpek and Majid Afshar 2979

Human-in-the-loop Robotic Grasping Using BERT Scene Representation
Yaoxian Song, Penglei Sun, Pengfei Fang, Linyi Yang, Yanghua Xiao and Yue Zhang 2992

Automated Chinese Essay Scoring from Multiple Traits
Yaqiong He, Feng Jiang, Xiaomin Chu and Peifeng Li 3007

Semantic-Preserving Adversarial Code Comprehension
Yiyang Li, Hongqiu Wu and Hai Zhao 3017

Continually Detection, Rapidly React: Unseen Rumors Detection Based on Continual Prompt-Tuning
Yuhui Zuo, Wei Zhu and Guoyong GUET Cai 3029

AiM: Taking Answers in Mind to Correct Chinese Cloze Tests in Educational Applications
Yusen Zhang, Zhongli Li, Qingyu Zhou, Ziyi Liu, Chao Li, Mina Ma, Yunbo Cao and Hongzhi Liu 3042

TreeMAN: Tree-enhanced Multimodal Attention Network for ICD Coding
Zichen Liu, Xuyuan Liu, Yanlong Wen, Guoqing Zhao, fen xia and Xiaojie Yuan 3054

Gated Mechanism Enhanced Multi-Task Learning for Dialog Routing
Ziming Huang, Zhuoxuan Jiang, Ke Wang, Juntao Li, Shanshan Feng and Xian-Ling Mao . . . 3064

Negation, Coordination, and Quantifiers in Contextualized Language Models
Aikaterini-Lida Kalouli, Rita Sevastjanova, Christin Beck and Maribel Romero 3074

Tales and Tropes: Gender Roles from Word Embeddings in a Century of Children's Books
Anjali Adukia, Patricia Chiril, Callista Christ, Anjali Das, Alex Eble, Emileigh Harrison and Hakizumwami Birali Runesha 3086

CLOWER: A Pre-trained Language Model with Contrastive Learning over Word and Character Representations
Borun Chen, Hongyin Tang, Jiahao Bu, Kai Zhang, Jingang Wang, Qifan Wang, Hai-Tao Zheng, Wei Wu and Liqian Yu 3098

On the Nature of BERT: Correlating Fine-Tuning and Linguistic Competence
Federica Merendi, Felice Dell'Orletta and Giulia Venturi 3109

LayerConnect: Hypernetwork-Assisted Inter-Layer Connector to Enhance Parameter Efficiency
Haoxiang Shi, Rongsheng Zhang, Jiaan Wang, Cen Wang, Yinhe Zheng and Tetsuya Sakai . . 3120

Effect of Post-processing on Contextualized Word Representations
Hassan Sajjad, Firoj Alam, Fahim Dalvi and Nadir Durrani 3127

Does BERT Rediscover a Classical NLP Pipeline?
Jingcheng Niu, Wenjie Lu and Gerald Penn 3143

HG2Vec: Improved Word Embeddings from Dictionary and Thesaurus Based Heterogeneous Graph
Qitong Wang and Mohammed J Zaki 3154

Transferring Knowledge from Structure-aware Self-attention Language Model to Sequence-to-Sequence Semantic Parsing
Ran Ji and Jianmin Ji 3164

Enhancing Contextual Word Representations Using Embedding of Neighboring Entities in Knowledge Graphs
Ryoko Tokuhisa, Keisuke Kawano, Akihiro Nakamura and Satoshi Koide 3175

Generic Overgeneralization in Pre-trained Language Models
Sello Ralethe and Jan Buys 3187

How about Time? Probing a Multilingual Language Model for Temporal Relations
Tommaso Caselli, Irene Dini and Felice Dell'Orletta 3197

CogBERT: Cognition-Guided Pre-trained Language Models
Xiao Ding, Bowen Chen, Li Du, Bing Qin and Ting Liu 3210

Can Transformers Process Recursive Nested Constructions, Like Humans?
Yair Lakretz, Théo Desbordes, Dieuwke Hupkes and Stanislas Dehaene 3226

NSP-BERT: A Prompt-based Few-Shot Learner through an Original Pre-training Task —— Next Sentence Prediction
Yi Sun, Yu Zheng, Chao Hao and Hangping Qiu 3233

MetaPrompting: Learning to Learn Better Prompts
Yutai Hou, Hongyuan Dong, Xinghao Wang, Bohan Li and Wanxiang Che 3251

Parameter-Efficient Mixture-of-Experts Architecture for Pre-trained Language Models
Ze-Feng Gao, Peiyu Liu, Wayne Xin Zhao, Zhong-Yi Lu and Ji-Rong Wen 3263

Pre-trained Token-replaced Detection Model as Few-shot Learner
Zicheng Li, Shoushan Li and Guodong Zhou 3274

Evaluating Diversity of Multiword Expressions in Annotated Text
Adam Lion-Bouton, Yagmur Ozturk, Agata Savary and Jean-Yves Antoine 3285

CausalQA: A Benchmark for Causal Question Answering
Alexander Bondarenko, Magdalena Wolska, Stefan Heindorf, Lukas Blübaum, Axel-Cyrille Ngonga Ngomo, Benno Stein, Pavel Braslavski, Matthias Hagen and Martin Potthast 3296

MACRONYM: A Large-Scale Dataset for Multilingual and Multi-Domain Acronym Extraction
Amir Pouran Ben Veyseh, Nicole Meister, Seunghyun Yoon, Rajiv Jain, Franck Dernoncourt and Thien Huu Nguyen 3309

Curating a Large-Scale Motivational Interviewing Dataset Using Peer Support Forums
Anuradha Welivita and Pearl Pu 3315

CCTC: A Cross-Sentence Chinese Text Correction Dataset for Native Speakers
Baoxin Wang, Xingyi Duan, Dayong Wu, Wanxiang Che, Zhigang Chen and Guoping Hu . . . 3331

RealMedDial: A Real Telemedical Dialogue Dataset Collected from Online Chinese Short-Video Clips
Bo Xu, Hongtong Zhang, Jian Wang, Xiaokun Zhang, Dezhi Hao, Linlin Zong, Hongfei Lin and Fenglong Ma 3342

TempoWiC: An Evaluation Benchmark for Detecting Meaning Shift in Social Media
Daniel Loureiro, Aminette D'Souza, Areej Nasser Muhajab, Isabella A. White, Gabriel Wong, Luis Espinosa-Anke, Leonardo Neves, Francesco Barbieri and Jose Camacho-Collados 3353

Automatic Generation of Large-scale Multi-turn Dialogues from Reddit
Daniil Huryn, William M. Hutsell and Jinho D. Choi 3360

ConFiguRe: Exploring Discourse-level Chinese Figures of Speech
Dawei Zhu, Qiusi Zhan, Zhejian Zhou, Yifan Song, Jiebin Zhang and Sujian Li 3374

Twitter Topic Classification
Dimosthenis Antypas, Asahi Ushio, Jose Camacho-Collados, Vitor Silva, Leonardo Neves and Francesco Barbieri 3386

Layer or Representation Space: What Makes BERT-based Evaluation Metrics Robust?
Doan Nam Long Vu, Nafise Sadat Moosavi and Steffen Eger 3401

Evaluating the Performance of Transformer-based Language Models for Neuroatypical Language
Duanchen Liu, Zoey Liu, Qingyun Yang, Yujing Huang and Emily Prud'hommeaux 3412

TERMinator: A System for Scientific Texts Processing
Elena Bruches, Olga Tikhobaeva, Yana Dementyeva and Tatiana Batura 3420

LipKey: A Large-Scale News Dataset for Absent Keyphrases Generation and Abstractive Summarization
Fajri Koto, Timothy Baldwin and Jey Han Lau 3427

Understanding Attention for Vision-and-Language Tasks
Feiqi Cao, Soyeon Caren Han, Siqu Long, Changwei Xu and Josiah Poon 3438

Effective Data Augmentation for Sentence Classification Using One VAE per Class
Frédéric Piedboeuf and Philippe Langlais 3454

NLG-Metricverse: An End-to-End Library for Evaluating Natural Language Generation
Giacomo Frisoni, Antonella Carbonaro, Gianluca Moro, Andrea Zammarchi and Marco Avagnano 3465

TestAug: A Framework for Augmenting Capability-based NLP Tests
Guanqun Yang, Mirazul Haque, Qiaochu Song, Wei Yang and Xueqing Liu 3480

KoCHET: A Korean Cultural Heritage Corpus for Entity-related Tasks
Gyeongmin Kim, Jinsung Kim, Junyoung Son and Heuiseok Lim 3496

MonoByte: A Pool of Monolingual Byte-level Language Models
Hugo Abonizio, Leandro Rodrigues de Souza, Roberto Lotufo and Rodrigo Nogueira 3506

Wizard of Tasks: A Novel Conversational Dataset for Solving Real-World Tasks in Conversational Settings
Jason Ingyu Choi, Saar Kuzi, Nikhita Vedula, Jie Zhao, Giuseppe Castellucci, Marcus Collins, Shervin Malmasi, Oleg Rokhlenko and Eugene Agichtein 3514

K-MHaS: A Multi-label Hate Speech Detection Dataset in Korean Online News Comment
Jean Lee, Taejun Lim, Heejun Lee, Bogeun Jo, Yangsok Kim, Heegeun Yoon and Soyeon Caren Han 3530

Domain- and Task-Adaptation for VaccinChatNL, a Dutch COVID-19 FAQ Answering Corpus and Classification Model
Jeska Buhmann, Maxime De Bruyn, Ehsan Lotfi and Walter Daelemans 3539

Benchmarking Automated Clinical Language Simplification: Dataset, Algorithm, and Evaluation
Junyu Luo, Junxian Lin, Chi Lin, Cao Xiao, Xinning Gui and Fenglong Ma 3550

WikiHan: A New Comparative Dataset for Chinese Languages
Kalvin Chang, Chenxuan Cui, Youngmin Kim and David R. Mortensen 3563

Visual Recipe Flow: A Dataset for Learning Visual State Changes of Objects with Recipe Flows
Keisuke Shirai, Atsushi Hashimoto, Taichi Nishimura, Hirotaka Kameko, Shuhei Kurita, Yoshitaka Ushiku and Shinsuke Mori 3570

IMPARA: Impact-Based Metric for GEC Using Parallel Data
Koki Maeda, Masahiro Kaneko and Naoaki Okazaki 3578

Evons: A Dataset for Fake and Real News Virality Analysis and Prediction
Kriste Krstovski, Angela Soomin Ryu and Bruce Kogut 3589

Are Pretrained Multilingual Models Equally Fair across Languages?
Laura Cabello Piqueras and Anders Søgaard 3597

Possible Stories: Evaluating Situated Commonsense Reasoning under Multiple Possible Scenarios
Mana Ashida and Saku Sugawara 3606

DiaBiz.Kom - towards a Polish Dialogue Act Corpus Based on ISO 24617-2 Standard
Marcin Oleksy, Jan Wieczorek, Dorota Drużyłowska, Julia Klyus, Aleksandra Domogała, Krzysztof Hwaszcz, Hanna Kędzierska, Daria Mikoś and Anita Wróż 3631

Towards Explainable Evaluation of Language Models on the Semantic Similarity of Visual Concepts
Maria Lymperaiou, George Manoliadis, Orfeas Menis Mastromichalakis, Edmund G. Dervakos and Giorgos Stamou 3639

Establishing Annotation Quality in Multi-label Annotations
Marian Marchal, Merel Scholman, Frances Yung and Vera Demberg 3659

Biographically Relevant Tweets – a New Dataset, Linguistic Analysis and Classification Experiments
Michael Wiegand, Rebecca Wilm and Katja Markert 3669

BECEL: Benchmark for Consistency Evaluation of Language Models
Myeongjun Jang, Deuk Sin Kwon and Thomas Lukasiewicz 3680

KoBEST: Korean Balanced Evaluation of Significant Tasks
Myeongjun Jang, Dohyung Kim, Deuk Sin Kwon and Eric Davis 3697

A New Public Corpus for Clinical Section Identification: MedSecId
Paul Landes, Kunal Patel, Sean S. Huang, Adam Webb, Barbara Di Eugenio and Cornelia Caragea 3709

A Data-driven Approach to Named Entity Recognition for Early Modern French
Pedro Ortiz Suarez and Simon Gabay 3722

Reproducibility and Automation of the Appraisal Taxonomy
Pradeesh Parameswaran, Andrew Trotman, Veronica Liesaputra and David Eyers 3731

Few-Shot Table Understanding: A Benchmark Dataset and Pre-Training Baseline
Ruixue Liu, Shaozu Yuan, Aijun Dai, Lei Shen, Tiangang Zhu, Meng Chen and Xiaodong He 3741

Tafsir Dataset: A Novel Multi-Task Benchmark for Named Entity Recognition and Topic Modeling in Classical Arabic Literature
Sajawel Ahmed, Rob van der Goot, Misbahur Rehman, Carl Kruse, Ömer Özsoy, Alexander Mehler and Gemma Roig 3753

Resource of Wikipedias in 31 Languages Categorized into Fine-Grained Named Entities
Satoshi Sekine, Kouta Nakayama, Masako Nomoto, Maya Ando, Asuka Sumida and Koji Matsuda 3769

Accuracy meets Diversity in a News Recommender System
Shaina Raza, Syed Raza Bashir and Usman Naseem 3778

Dynamic Nonlinear Mixup with Distance-based Sample Selection
Shaokang Zhang, Lei Jiang and Jianlong Tan 3788

MultiCoNER: A Large-scale Multilingual Dataset for Complex Named Entity Recognition
Shervin Malmasi, Anjie Fang, Besnik Fetahu, Sudipta Kar and Oleg Rokhlenko 3798

Extracting a Knowledge Base of COVID-19 Events from Social Media
Shi Zong, Ashutosh Baheti, Wei Xu and Alan Ritter 3810

Accounting for Language Effect in the Evaluation of Cross-lingual AMR Parsers
Shira Wein and Nathan Schneider 3824

QSTS: A Question-Sensitive Text Similarity Measure for Question Generation
Sujatha Das Gollapalli and See-Kiong Ng 3835

Noun-MWP: Math Word Problems Meet Noun Answers
Taehun Cha, Jaeheun Jung and Donghun Lee 3847

ViNLI: A Vietnamese Corpus for Studies on Open-Domain Natural Language Inference
Tin Van Huynh, Kiet Van Nguyen and Ngan Luu-Thuy Nguyen 3858

InferES : A Natural Language Inference Corpus for Spanish Featuring Negation-Based Contrastive and Adversarial Examples
Venelin Kovatchev and Mariona Taulé 3873

ParaZh-22M: A Large-Scale Chinese Parabank via Machine Translation
Wenjie Hao, Hongfei Xu, Deyi Xiong, Hongying Zan and Lingling Mu 3885

ESimCSE: Enhanced Sample Building Method for Contrastive Learning of Unsupervised Sentence Embedding
Xing Wu, Chaochen Gao, Liangjun Zang, Jizhong Han, Zhongyuan Wang and Songlin Hu . . . 3898

Measuring Robustness for NLP
Yu Yu, Abdul Rafae Khan and Jia Xu 3908

CSL: A Large-scale Chinese Scientific Literature Dataset
Yudong Li, Yuqing Zhang, Zhe Zhao, Linlin Shen, Weijie Liu, Weiquan Mao and Hui Zhang 3917

Singlish Message Paraphrasing: A Joint Task of Creole Translation and Text Normalization
Zhengyuan Liu, Shikang Ni, Ai Ti Aw and Nancy F. Chen 3924

CINO: A Chinese Minority Pre-trained Language Model
Ziqing Yang, Zihang Xu, Yiming Cui, Baoxin Wang, Min Lin, Dayong Wu and Zhigang Chen 3937

One Word, Two Sides: Traces of Stance in Contextualized Word Representations
Aina Garí Soler, Matthieu Labeau and Chloé Clavel 3950

Prepositions Matter in Quantifier Scope Disambiguation
Aleksander Leczkowski, Justyna Grudzińska, Manuel Vargas Guzmán, Aleksander Wawer and Aleksandra Siemieniuk 3960

Modelling Commonsense Properties Using Pre-Trained Bi-Encoders
Amit Gajbhiye, Luis Espinosa-Anke and Steven Schockaert 3971

COIN – an Inexpensive and Strong Baseline for Predicting Out of Vocabulary Word Embeddings
Andrew Schneider, Lihong He, Zhijia Chen, Arjun Mukherjee and Eduard Dragut 3984

DynGL-SDP: Dynamic Graph Learning for Semantic Dependency Parsing
Bin Li, Miao Gao, Yunlong Fan, Yikemaiti Sataer, Zhiqiang Gao and Yaocheng Gui 3994

Knowledge Is Flat: A Seq2Seq Generative Framework for Various Knowledge Graph Completion
Chen Chen, Yufei Wang, Bing Li and Kwok-Yan Lam 4005

Modelling Frequency, Attestation, and Corpus-Based Information with OntoLex-FrAC
Christian Chiarcos, Elena-Simona Apostol, Besim Kabashi and Ciprian-Octavian Truică 4018

Contrast Sets for Stativity of English Verbs in Context
Daniel Chen and Alexis Palmer 4028

Multilingual and Multimodal Topic Modelling with Pretrained Embeddings
Elaine Zosa and Lidia Pivovarova 4037

Zero-shot Script Parsing
Fangzhou Zhai, Vera Demberg and Alexander Koller 4049

Word Sense Disambiguation with Knowledge-Enhanced and Local Self-Attention-based Extractive Sense Comprehension
Guobiao Zhang, Wenpeng Lu, Xueping Peng, Shoujin Wang, Baoshuo Kan and Rui Yu 4061

A Novel Multi-Task Learning Approach for Context-Sensitive Compound Type Identification in Sanskrit
Jivnesh Sandhan, Ashish Gupta, Hrishikesh Terdalkar, Tushar Sandhan, Suvendu Samanta, Laxmidhar Behera and Pawan Goyal 4071

Testing Large Language Models on Compositionality and Inference with Phrase-Level Adjective-Noun Entailment
Lorenzo Bertolini, Julie Weeds and David Weir 4084

Does BERT Recognize an Agent? Modeling Dowty's Proto-Roles with Contextual Embeddings
Mattia Proietti, Gianluca Lebani and Alessandro Lenci 4101

Towards Structure-aware Paraphrase Identification with Phrase Alignment Using Sentence Encoders
Qiwei Peng, David Weir and Julie Weeds 4113

CILex: An Investigation of Context Information for Lexical Substitution Methods
Sandaru Seneviratne, Elena Daskalaki, Artem Lenskiy and Hanna Suominen 4124

Emotion Enriched Retrofitted Word Embeddings
Sapan Shah, Sreedhar Reddy and Pushpak Bhattacharyya 4136

Metaphor Detection via Linguistics Enhanced Siamese Network
Shenglong Zhang and Ying Liu 4149

Fast and Accurate End-to-End Span-based Semantic Role Labeling as Word-based Graph Parsing
Shilin Zhou, Qingrong Xia, Zhenghua Li, Yu Zhang, Yu Hong and Min Zhang 4160

Unsupervised Lexical Substitution with Decontextualised Embeddings
Takashi Wada, Timothy Baldwin, Yuji Matsumoto and Jey Han Lau 4172

Transparent Semantic Parsing with Universal Dependencies Using Graph Transformations
Wessel Poelman, Rik van Noord and Johan Bos 4186

Multilingual Word Sense Disambiguation with Unified Sense Representation
Ying Su, Hongming Zhang, Yangqiu Song and Tong Zhang 4193

A Transition-based Method for Complex Question Understanding
Yu Xia, Wenbin Jiang, Yajuan Lyu and Sujian Li 4203

Semantic Role Labeling as Dependency Parsing: Exploring Latent Tree Structures inside Arguments
Yu Zhang, Qingrong Xia, Shilin Zhou, Yong Jiang, Guohong Fu and Min Zhang 4212

Noisy Label Regularisation for Textual Regression
Yuxia Wang, Timothy Baldwin and Karin Verspoor 4228

Detecting Suicide Risk in Online Counseling Services: A Study in a Low-Resource Language
Amir Bialer, Daniel Izmaylov, Avi Segal, Oren Tsur, Yossi Levi-Belz and Kobi Gal 4241

Does Meta-learning Help mBERT for Few-shot Question Generation in a Cross-lingual Transfer Setting for Indic Languages?
Aniruddha Roy, Rupak Kumar Thakur, Isha Sharma, Ashim Gupta, Amrith Krishna, Sudeshna Sarkar and Pawan Goyal 4251

Revisiting Syllables in Language Modelling and Their Application on Low-Resource Machine Translation
Arturo Oncevay, Kervy Dante Rivas Rojas, Liz Karen Chavez Sanchez and Roberto Zariquiey 4258

Aligning Multilingual Embeddings for Improved Code-switched Natural Language Understanding
Barah Fazili and Preethi Jyothi 4268

Fashioning Local Designs from Generic Speech Technologies in an Australian Aboriginal Community
Éric Le Ferrand, Steven Bird and Laurent Besacier 4274

Few-Shot Pidgin Text Adaptation via Contrastive Fine-Tuning
Ernie Chang, Jesujoba O. Alabi, David Ifeoluwa Adelani and Vera Demberg 4286

Penalizing Divergence: Multi-Parallel Translation for Low-Resource Languages of North America
Garrett Nicolai, Changbing Yang and Miikka Silfverberg 4292

Assessing Digital Language Support on a Global Scale
Gary F. Simons, Abbey L. L. Thomas and Chad K. K. White 4299

Persian Natural Language Inference: A Meta-learning Approach
Heydar Soudani, Mohammad Hassan Mojab and Hamid Beigy 4306

Global Readiness of Language Technology for Healthcare: What Would It Take to Combat the Next Pandemic?
Ishani Mondal, Kabir Ahuja, Mohit Jain, Jacki O'Neill, Kalika Bali and Monojit Choudhury . 4320

Adapting Pre-trained Language Models to African Languages via Multilingual Adaptive Fine-Tuning
Jesujoba O. Alabi, David Ifeoluwa Adelani, Marius Mosbach and Dietrich Klakow 4336

Noun Class Disambiguation in Runyankore and Related Languages
Joan Byamugisha 4350

Improving Low-resource RRG Parsing with Cross-lingual Self-training
Kilian Evang, Laura Kallmeyer, Jakub Waszczuk, Kilu von Prince, Tatiana Bladier and Simon Petitjean 4360

A Simple and Effective Method to Improve Zero-Shot Cross-Lingual Transfer Learning
Kunbo Ding, Weijie Liu, Yuejian Fang, Weiquan Mao, Zhe Zhao, Tao Zhu, Haoyan Liu, Rong Tian and Yiren Chen 4372

Towards Multi-Sense Cross-Lingual Alignment of Contextual Embeddings
Linlin Liu, Thien Hai Nguyen, Shafiq Joty, Lidong Bing and Luo Si 4381

How to Parse a Creole: When Martinican Creole Meets French
Ludovic Mompelat, Daniel Dakota and Sandra Kübler 4397

Byte-based Multilingual NMT for Endangered Languages
Mengjiao Zhang and Jia Xu 4407

BRCC and SentiBahasaRojak: The First Bahasa Rojak Corpus for Pretraining and Sentiment Analysis Dataset
Nanda Putri Romadhona, Sin-En Lu, Bo-Han Lu and Richard Tzong-Han Tsai 4418

WordNet-QU: Development of a Lexical Database for Quechua Varieties
Nelsi Melgarejo, Rodolfo Zevallos, Hector Gomez and John E. Ortega 4429

When the Student Becomes the Master: Learning Better and Smaller Monolingual Models from mBERT
Pranaydeep Singh and Els Lefever 4434

Zero-shot Disfluency Detection for Indian Languages
Rohit Kundu, Preethi Jyothi and Pushpak Bhattacharyya 4442

Evaluating Word Embeddings in Extremely Under-Resourced Languages: A Case Study in Bribri
Rolando Coto-Solano 4455

Applying Natural Annotation and Curriculum Learning to Named Entity Recognition for Under-Resourced Languages
Valeriy Lobov, Alexandra Ivoylova and Serge Sharoff 4468

Taking Actions Separately: A Bidirectionally-Adaptive Transfer Learning Method for Low-Resource Neural Machine Translation
Xiaolin Xing, Yu Hong, Minhan Xu, Jianmin Yao and Guodong Zhou 4481

HCLD: A Hierarchical Framework for Zero-shot Cross-lingual Dialogue System
Zhanyu Ma, Jian Ye, Xurui Yang and Jianfeng Liu 4492

GraDA: Graph Generative Data Augmentation for Commonsense Reasoning
Adyasha Maharana and Mohit Bansal 4499

Eureka: Neural Insight Learning for Knowledge Graph Reasoning
Alex X. Zhang, Xun Liang, Bo Wu, Xiangping Zheng, Sensen Zhang, Yuhui Guo, Jun Wang and Xinyao Liu 4517

CitRet: A Hybrid Model for Cited Text Span Retrieval
Amit Pandey, Avani Gupta and Vikram Pudi 4528

A Weak Supervision Approach for Predicting Difficulty of Technical Interview Questions
Arpita Kundu, Subhasish Ghosh, Pratik Saini, Tapas Nayak and Indrajit Bhattacharya 4537

Reinforcement Learning with Large Action Spaces for Neural Machine Translation
Asaf Yehudai, Leshem Choshen, Lior Fox and Omri Abend 4544

Noise Learning for Text Classification: A Benchmark
Bo Liu, Wandi Xu, Yuejia Xiang, Xiaojun Wu, Lejian He, Bowen Zhang and Li Zhu 4557

Mitigating the Diminishing Effect of Elastic Weight Consolidation
Canasai Kruengkrai and Junichi Yamagishi 4568

Token and Head Adaptive Transformers for Efficient Natural Language Processing
Chonghan Lee, Md Fahim Faysal Khan, Rita Brugarolas Brufau, Ke Ding and Vijaykrishnan Narayanan 4575

Don't Judge a Language Model by Its Last Layer: Contrastive Learning with Layer-Wise Attention Pooling
Dongsuk Oh, Yejin Kim, Hodong Lee, H. Howie Huang and Heuiseok Lim 4585

SHAP-Based Explanation Methods: A Review for NLP Interpretability
Edoardo Mosca, Ferenc Szigeti, Stella Tragianni, Daniel Gallagher and Georg Groh 4593

A Simple Log-based Loss Function for Ordinal Text Classification
François Castagnos, Martin Mihelich and Charles Dognin 4604

Ask Question First for Enhancing Lifelong Language Learning
Han Wang, Ruiliu Fu, Xuejun Zhang, Jun Zhou and Qingwei Zhao 4610

DoubleMix: Simple Interpolation-Based Data Augmentation for Text Classification
Hui Chen, Wei Han, Diyi Yang and Soujanya Poria 4622

Large Sequence Representation Learning via Multi-Stage Latent Transformers
Ionut-Catalin Sandu, Daniel Voinea and Alin-Ionut Popa 4633

MockingBERT: A Method for Retroactively Adding Resilience to NLP Models
Jan Jezabek and Akash Singh 4640

Equivariant Transduction through Invariant Alignment
Jennifer C. White and Ryan Cotterell 4651

Where Does Linguistic Information Emerge in Neural Language Models? Measuring Gains and Contributions across Layers
Jenny Kunz and Marco Kuhlmann 4664

Accelerating Inference for Pretrained Language Models by Unified Multi-Perspective Early Exiting
Jun Kong, Jin Wang, Liang-Chih Yu and Xuejie Zhang 4677

Topology Imbalance and Relation Inauthenticity Aware Hierarchical Graph Attention Networks for Fake News Detection
Li Gao, Lingyun Song, Jie Liu, Bolin Chen and Xuequn Shang 4687

Temporal Knowledge Graph Completion with Approximated Gaussian Process Embedding
Linhai Zhang and Deyu Zhou 4697

CILDA: Contrastive Data Augmentation Using Intermediate Layer Knowledge Distillation
Md Akmal Haidar, Mehdi Rezagholizadeh, Abbas Ghaddar, Khalil Bibi, Phillippe Langlais and Pascal Poupart 4707

Pro-KD: Progressive Distillation by Following the Footsteps of the Teacher
Mehdi Rezagholizadeh, Aref Jafari, Puneeth S.M. Saladi, Pranav Sharma, Ali Saheb Pasand and Ali Ghodsi 4714

Classical Sequence Match Is a Competitive Few-Shot One-Class Learner
Mengting Hu, Hang Gao, Yinhao Bai and Mingming Liu 4728

Unsupervised Domain Adaptation for Text Classification via Meta Self-Paced Learning
Nghia Ngo Trung, Linh Ngo Van and Thien Huu Nguyen 4741

WARM: A Weakly (+Semi) Supervised Math Word Problem Solver
Oishik Chatterjee, Isha Pandey, Aashish Waikar, Vishwajeet Kumar and Ganesh Ramakrishnan 4753

Attention Networks for Augmenting Clinical Text with Support Sets for Diagnosis Prediction
Paul Grundmann, Tom Oberhauser, Felix Gers and Alexander Löser 4765

PARSE: An Efficient Search Method for Black-box Adversarial Text Attacks
Pengwei Zhan, Chao Zheng, Jing Yang, Yuxiang Wang, Liming Wang, Yang Wu and Yunjian Zhang 4776

A Closer Look at Parameter Contributions When Training Neural Language and Translation Models
Raúl Vázquez, Hande Celikkanat, Vinit Ravishankar, Mathias Creutz and Jörg Tiedemann . . . 4788

KNOT: Knowledge Distillation Using Optimal Transport for Solving NLP Tasks
Rishabh Bhardwaj, Tushar Vaidya and Soujanya Poria 4801

An Information Minimization Based Contrastive Learning Model for Unsupervised Sentence Embeddings Learning
Shaobin Chen, Jie Zhou, Yuling Sun and Liang He 4821

Learn2Weight: Parameter Adaptation against Similar-domain Adversarial Attacks
Siddhartha Datta 4832

Sentence-aware Adversarial Meta-Learning for Few-Shot Text Classification
Suhe Wang, Xiaoyuan Liu, Bo Liu and Diwen Dong 4844

Reweighting Strategy Based on Synthetic Data Identification for Sentence Similarity
TaeHee Kim, ChaeHun Park, Jimin Hong, Radhika Dua, Edward Choi and Jaegul Choo 4853

MaxMatch-Dropout: Subword Regularization for WordPiece
Tatsuya Hiraoka 4864

Adaptive Meta-learner via Gradient Similarity for Few-shot Text Classification
Tianyi Lei, Honghui Hu, Qiaoyang Luo, Dezhong Peng and Xu Wang 4873

Vocabulary-informed Language Encoding
Xi Ai and Bin Fang 4883

OpticE: A Coherence Theory-Based Model for Link Prediction
Xiangyu Gui, Feng Zhao, Langjunqing Jin and Hai Jin 4892

Smoothed Contrastive Learning for Unsupervised Sentence Embedding
Xing Wu, Chaochen Gao, Yipeng Su, Jizhong Han, Zhongyuan Wang and Songlin Hu 4902

Knowledge Distillation with Reptile Meta-Learning for Pretrained Language Model Compression
Xinge Ma, Jin Wang, Liang-Chih Yu and Xuejie Zhang 4907

RotateCT: Knowledge Graph Embedding by Rotation and Coordinate Transformation in Complex Space
Yao Dong, Lei Wang, Ji Xiang, Xiaobo Guo and Yuqiang Xie 4918

Can Data Diversity Enhance Learning Generalization?
Yu Yu, Shahram Khadivi and Jia Xu 4933

Generate-and-Retrieve: Use Your Predictions to Improve Retrieval for Semantic Parsing
Yury Zemlyanskiy, Michiel de Jong, Joshua Ainslie, Panupong Pasupat, Peter Shaw, Linlu Qiu, Sumit Sanghai and Fei Sha 4946

Coarse-to-Fine: Hierarchical Multi-task Learning for Natural Language Understanding
Zhaoye Fei, Yu Tian, Yongkang Wu, Xinyu Zhang, Yutao Zhu, Zheng Liu, Jiawen Wu, Dejiang Kong, Ruofei Lai, Zhao Cao, Zhicheng Dou and Xipeng Qiu 4952

Automatic Label Sequence Generation for Prompting Sequence-to-sequence Models
Zichun Yu, Tianyu Gao, Zhengyan Zhang, Yankai Lin, Zhiyuan Liu, Maosong Sun and Jie Zhou 4965

Unsupervised Sentence Textual Similarity with Compositional Phrase Semantics
Zihao Wang, Jiaheng Dou and Yong Zhang 4976

A Generalized Method for Automated Multilingual Loanword Detection
Abhijnan Nath, Sina Mahdipour Saravani, Ibrahim Khebour, Sheikh Mannan, Zihui Li and Nikhil Krishnaswamy 4996

FeatureBART: Feature Based Sequence-to-Sequence Pre-Training for Low-Resource NMT
Abhisek Chakrabarty, Raj Dabre, Chenchen Ding, Hideki Tanaka, Masao Utiyama and Eiichiro Sumita 5014

Multi-level Community-awareness Graph Neural Networks for Neural Machine Translation
Binh Nguyen, Long Nguyen and Dien Dinh 5021

On the Complementarity between Pre-Training and Random-Initialization for Resource-Rich Machine Translation
Changtong Zan, Liang Ding, Li Shen, Yu Cao, Weifeng Liu and Dacheng Tao 5029

ngram-OAXE: Phrase-Based Order-Agnostic Cross Entropy for Non-Autoregressive Machine Translation
Cunxiao Du, Zhaopeng Tu, Longyue Wang and Jing Jiang 5035

Language Branch Gated Multilingual Neural Machine Translation
Haoran Sun and Deyi Xiong 5046

Iterative Constrained Back-Translation for Unsupervised Domain Adaptation of Machine Translation
Hongxiao Zhang, Hui Huang, Jiale Gao, Yufeng Chen, Jinan Xu and Jian Liu 5054

Linguistically-Motivated Yorùbá-English Machine Translation
Ife Adebara, Muhammad Abdul-Mageed and Miikka Silfverberg 5066

Dynamic Position Encoding for Transformers
Joyce Zheng, Mehdi Rezagholizadeh and Peyman Passban 5076

PAEG: Phrase-level Adversarial Example Generation for Neural Machine Translation
Juncheng Wan, Jian Yang, Shuming Ma, Dongdong Zhang, Weinan Zhang, Yong Yu and Zhoujun Li 5085

Noise-robust Cross-modal Interactive Learning with Text2Image Mask for Multi-modal Neural Machine Translation
Junjie Ye, Junjun Guo, Yan Xiang, Kaiwen Tan and Zhengtao Yu 5098

Speeding up Transformer Decoding via an Attention Refinement Network
Kaixin WU, Yue Zhang, Bojie Hu and Tong Zhang 5109

Interactive Post-Editing for Verbosity Controlled Translation
Prabhakar Gupta, Anil Nelakanti, Grant M. Berry and Abhishek Sharma 5119

Addressing Asymmetry in Multilingual Neural Machine Translation with Fuzzy Task Clustering
Qian Wang and Jiajun Zhang 5129

Learning Decoupled Retrieval Representation for Nearest Neighbour Neural Machine Translation
Qiang Wang, Rongxiang Weng and Ming Chen 5142

Semantically Consistent Data Augmentation for Neural Machine Translation via Conditional Masked Language Model
Qiao Cheng, Jin Huang and Yitao Duan 5148

Informative Language Representation Learning for Massively Multilingual Neural Machine Translation
Renren Jin and Deyi Xiong 5158

Rare but Severe Neural Machine Translation Errors Induced by Minimal Deletion: An Empirical Study on Chinese and English
Ruikang Shi, Alvin Grissom II and Duc Minh Trinh 5175

QUAK: A Synthetic Quality Estimation Dataset for Korean-English Neural Machine Translation
Sugyeong Eo, Chanjun Park, Hyeonseok Moon, Jaehyung Seo, Gyeongmin Kim, Jungseob Lee and Heuiseok Lim 5181

Improving Both Domain Robustness and Domain Adaptability in Machine Translation
Wen Lai, Jindřich Libovický and Alexander Fraser 5191

CoDoNMT: Modeling Cohesion Devices for Document-Level Neural Machine Translation
Yikun Lei, Yuqi Ren and Deyi Xiong 5205

Improving Non-Autoregressive Neural Machine Translation via Modeling Localness
Yong Wang and Xinwei Geng 5217

Categorizing Semantic Representations for Neural Machine Translation
Yongjing Yin, Yafu Li, Fandong Meng, Jie Zhou and Yue Zhang 5227

Adversarial Training on Disentangling Meaning and Language Representations for Unsupervised Quality Estimation
Yuto Kuroda, Tomoyuki Kajiwara, Yuki Arase and Takashi Ninomiya 5240

Alleviating the Inequality of Attention Heads for Neural Machine Translation
Zewei Sun, Shujian Huang, Xinyu Dai and Jiajun Chen 5246

Adapting to Non-Centered Languages for Zero-shot Multilingual Translation
Zhi Qu and Taro Watanabe 5251

Towards Robust Neural Machine Translation with Iterative Scheduled Data-Switch Training
Zhongjian Miao, Xiang Li, Liyan Kang, Wen Zhang, Chulun Zhou, Yidong Chen, Bin Wang, Min Zhang and Jinsong Su 5266

Cross-lingual Feature Extraction from Monolingual Corpora for Low-resource Unsupervised Bilingual Lexicon Induction
Zihao Feng, Hailong Cao, Tiejun Zhao, Weixuan Wang and Wei Peng 5278

Language-Independent Approach for Morphological Disambiguation
Alymzhan Toleu, Gulmira Tolegen and Rustam Mussabayev 5288

SUN: Exploring Intrinsic Uncertainties in Text-to-SQL Parsers
Bowen Qin, Lihan Wang, Binyuan Hui, Bowen Li, Xiangpeng Wei, Binhua Li, Fei Huang, Luo Si, Min Yang and Yongbin Li 5298

Deciphering and Characterizing Out-of-Vocabulary Words for Morphologically Rich Languages
Georgie Botev, Arya D. McCarthy, Winston Wu and David Yarowsky 5309

PSSAT: A Perturbed Semantic Structure Awareness Transferring Method for Perturbation-Robust Slot Filling
Guanting Dong, Daichi Guo, Liwen Wang, Xuefeng Li, Zechen Wang, Chen Zeng, Keqing He, Jinzheng Zhao, Hao Lei, Xinyue Cui, Yi Huang, Junlan Feng and Weiran Xu 5327

String Editing Based Chinese Grammatical Error Diagnosis
Haihua Xie, Xiaoqing Lyu and Xuefei Chen 5335

The Fragility of Multi-Treebank Parsing Evaluation
Iago Alonso-Alonso, David Vilares and Carlos Gómez-Rodríguez 5345

FactMix: Using a Few Labeled In-domain Examples to Generalize to Cross-domain Named Entity Recognition
Linyi Yang, Lifan Yuan, Leyang Cui, Wenyang Gao and Yue Zhang 5360

Speaker-Aware Discourse Parsing on Multi-Party Dialogues
Nan Yu, Guohong Fu and Min Zhang 5372

Iterative Span Selection: Self-Emergence of Resolving Orders in Semantic Role Labeling
Shuhei Kurita, Hiroki Ouchi, Kentaro Inui and Satoshi Sekine 5383

Revisiting the Practical Effectiveness of Constituency Parse Extraction from Pre-trained Language Models
Taeuk Kim 5398

Position Offset Label Prediction for Grammatical Error Correction
Xiuyu Wu, Jingsong Yu, Xu Sun and Yunfang Wu 5409

Parsing Natural Language into Propositional and First-Order Logic with Dual Reinforcement Learning
Xuantao Lu, Jingping Liu, Zhouhong Gu, Hanwen Tong, Chenhao Xie, Junyang Huang, Yanghua Xiao and Wenguang Wang 5419

Yet Another Format of Universal Dependencies for Korean
Yige Chen, Eunkyul Leah Jo, Yundong Yao, KyungTae Lim, Miikka Silfverberg, Francis M. Tyers and Jungyeul Park 5432

Enhancing Structure-aware Encoder with Extremely Limited Data for Graph-based Dependency Parsing
Yuanhe Tian, Yan Song and Fei Xia 5438

Simple and Effective Graph-to-Graph Annotation Conversion
Yuxuan Wang, Zhilin Lei, Yuqiu Ji and Wanxiang Che 5450

BiBL: AMR Parsing and Generation with Bidirectional Bayesian Learning
Ziming Cheng, Zuchao Li and Hai Zhao 5461

Multi-Layer Pseudo-Siamese Biaffine Model for Dependency Parsing
Ziyao Xu, Houfeng Wang and Bingdong Wang 5476

Belief Revision Based Caption Re-ranker with Visual Semantic Information
Ahmed Sabir, Francesc Moreno-Noguer, Pranava Madhyastha and Lluís Padró 5488

Towards Understanding the Relation between Gestures and Language
Artem Abzaliev, Andrew Owens and Rada Mihalcea 5507

Building Joint Relationship Attention Network for Image-Text Generation
Changzhi Wang and Xiaodong Gu 5521

Learning to Focus on the Foreground for Temporal Sentence Grounding
Daizong Liu and Wei Hu 5532

Are Visual-Linguistic Models Commonsense Knowledge Bases?
Hsiu-Yu Yang and Carina Silberer 5542

Visual Prompt Tuning for Few-Shot Text Classification
Jingyuan Wen, Yutian Luo, Nanyi Fei, Guoxing Yang, Zhiwu Lu, Hao Jiang, Jie Jiang and Zhao Cao 5560

Systematic Analysis of Image Schemas in Natural Language through Explainable Multilingual Neural Language Processing
Lennart Wachowiak and Dagmar Gromann 5571

How to Adapt Pre-trained Vision-and-Language Models to a Text-only Input?
Lovisa Hagström and Richard Johansson 5582

ACT-Thor: A Controlled Benchmark for Embodied Action Understanding in Simulated Environments
Michael Hanna, Federico Pedeni, Alessandro Suglia, Alberto Testoni and Raffaella Bernardi . 5597

In-the-Wild Video Question Answering
Santiago Castro, Naihao Deng, Pingxuan Huang, Mihai Burzo and Rada Mihalcea 5613

Towards Better Semantic Understanding of Mobile Interfaces
Srinivas Sunkara, Maria Wang, Lijuan Liu, Gilles Baechler, Yu-Chung Hsiao, Jindong Chen, Abhanshu Sharma and James W. W. Stout 5636

End-to-end Dense Video Captioning as Sequence Generation
Wanrong Zhu, Bo Pang, Ashish V. Thapliyal, William Yang Wang and Radu Soricut 5651

SANCL: Multimodal Review Helpfulness Prediction with Selective Attention and Natural Contrastive Learning
Wei Han, Hui Chen, Zhen Hai, Soujanya Poria and Lidong Bing 5666

Dual Capsule Attention Mask Network with Mutual Learning for Visual Question Answering
Weidong Tian, Haodong Li and Zhong-Qiu Zhao 5678

Emergence of Hierarchical Reference Systems in Multi-agent Communication
Xenia Ohmer, Marko Duda and Elia Bruni 5689

Scene Graph Modification as Incremental Structure Expanding
Xuming Hu, Zhijiang Guo, Yu Fu, Lijie Wen and Philip S. Yu 5707

Overcoming Language Priors in Visual Question Answering via Distinguishing Superficially Similar Instances
Yike Wu, Yu Zhao, Shiwan Zhao, Ying Zhang, Xiaojie Yuan, Guoqing Zhao and Ning Jiang . 5721

Efficient Multilingual Multi-modal Pre-training through Triple Contrastive Loss
Youhan Lee, KyungTae Lim, Woonhyuk Baek, Byungseok Roh and Saehoon Kim 5730

LOViS: Learning Orientation and Visual Signals for Vision and Language Navigation
Yue Zhang and Parisa Kordjamshidi 5745

GAP: A Graph-aware Language Model Framework for Knowledge Graph-to-Text Generation
Anthony Colas, Mehrdad Alvandipour and Daisy Zhe Wang 5755

Content Type Profiling of Data-to-Text Generation Datasets
Ashish Upadhyay and Stewart Massie 5770

CoLo: A Contrastive Learning Based Re-ranking Framework for One-Stage Summarization
Chenxin An, Ming Zhong, Zhiyong Wu, Qin Zhu, Xuanjing Huang and Xipeng Qiu 5783

Of Human Criteria and Automatic Metrics: A Benchmark of the Evaluation of Story Generation
Cyril Chhun, Pierre Colombo, Fabian M. Suchanek and Chloé Clavel 5794

Selective Token Generation for Few-shot Natural Language Generation
Daejin Jo, Taehwan Kwon, Eun-Sol Kim and Sungwoong Kim 5837

A-TIP: Attribute-aware Text Infilling via Pre-trained Language Model
Dongyuan Li, Jingyi You, Kotaro Funakoshi and Manabu Okumura 5857

Multi Graph Neural Network for Extractive Long Document Summarization
Xuan-Dung Doan, Le-Minh Nguyen and Khac-Hoai Nam Bui 5870

Improving Zero-Shot Multilingual Text Generation via Iterative Distillation
Ernie Chang, Alex Marin and Vera Demberg 5876

Using Structured Content Plans for Fine-grained Syntactic Control in Pretrained Language Model Generation
Fei-Tzin Lee, Miguel Ballesteros, Feng Nan and Kathleen McKeown 5882

PrefScore: Pairwise Preference Learning for Reference-free Summarization Quality Assessment
Ge Luo, Hebi Li, Youbiao He and Forrest Sheng Bao 5896

Multi-Attribute Controlled Text Generation with Contrastive-Generator and External-Discriminator
Guisheng Liu, Yi Li, Yanqing Guo, Xiangyang Luo and Bo Wang 5904

Coordination Generation via Synchronized Text-Infilling
Hiroki Teranishi and Yuji Matsumoto 5914

KHANQ: A Dataset for Generating Deep Questions in Education
Huanli Gong, Liangming Pan and Hengchang Hu 5925

Multi-Figurative Language Generation
Huiyuan Lai and Malvina Nissim 5939

Enhancing Task-Specific Distillation in Small Data Regimes through Language Generation
Husam Quteineh, Spyridon Samothrakis and Richard Sutcliffe 5955

Boosting Code Summarization by Embedding Code Structures
Jikyoeng Son, Joonghyuk Hahn, HyeonTae Seo and Yo-Sub Han 5966

Comparative Graph-based Summarization of Scientific Papers Guided by Comparative Citations
Jingqiang Chen, Chaoxiang Cai, Xiaorui Jiang and Kejia Chen 5978

JPG - Jointly Learn to Align: Automated Disease Prediction and Radiology Report Generation
Jingyi You, Dongyuan Li, Manabu Okumura and Kenji Suzuki 5989

Automatic Nominalization of Clauses
John S. Y. Lee, Ho Hung Lim, Carol Webster and Anton Melser 6002

Benchmarking Compositionality with Formal Languages
Josef Valvoda, Naomi Saphra, Jonathan Rawski, Adina Williams and Ryan Cotterell 6007

Source-summary Entity Aggregation in Abstractive Summarization
José Ángel González, Annie Louis and Jackie Chi Kit Cheung 6019

How to Find Strong Summary Coherence Measures? A Toolbox and a Comparative Study for Summary Coherence Measure Evaluation
Julius Steen and Katja Markert 6035

Summarizing Dialogues with Negative Cues
Junpeng Liu, Yanyan Zou, Yuxuan Xi, Shengjie Li, Mian Ma and Zhuoye Ding 6050

ALEXSIS-PT: A New Resource for Portuguese Lexical Simplification
Kai North, Marcos Zampieri and Tharindu Ranasinghe 6057

APPDIA: A Discourse-aware Transformer-based Style Transfer Model for Offensive Social Media Conversations
Katherine Atwell, Sabit Hassan and Malihe Alikhani 6063

View Dialogue in 2D: A Two-stream Model in Time-speaker Perspective for Dialogue Summarization and beyond
Keli Xie, Dongchen He, Jiaxin Zhuang, Siyuan Lu and Zhongfeng Wang 6075

Denoising Large-Scale Image Captioning from Alt-text Data Using Content Selection Models
Khyathi Raghavi Chandu, Piyush Sharma, Soravit Changpinyo, Ashish V. Thapliyal and Radu Soricut 6089

Meta-CQG: A Meta-Learning Framework for Complex Question Generation over Knowledge Bases
Kun Zhang, Yunqi Qiu, Yuanzhuo Wang, Long Bai, Wei Li, Xuhui Jiang, Huawei Shen and Xueqi Cheng 6105

Graph-to-Text Generation with Dynamic Structure Pruning
Liang Li, Ruiying Geng, Bowen Li, Can Ma, Yinliang Yue, Binhua Li and Yongbin Li 6115

Multi-Perspective Document Revision
Mana Ihori, Hiroshi Sato, Tomohiro Tanaka and Ryo Masumura 6128

A Survey of Automatic Text Summarization Using Graph Neural Networks
Marco Ferdinand Salchner and Adam Jatowt 6139

Phrase-Level Localization of Inconsistency Errors in Summarization by Weak Supervision
Masato Takatsuka, Tetsunori Kobayashi and Yoshihiko Hayashi 6151

PoliSe: Reinforcing Politeness Using User Sentiment for Customer Care Response Generation
Mauajama Firdaus, Asif Ekbal and Pushpak Bhattacharyya 6165

Focus-Driven Contrastive Learning for Medical Question Summarization
Ming Zhang, Shuai Dou, Ziyang Wang and Yunfang Wu 6176

ArgLegalSumm: Improving Abstractive Summarization of Legal Documents with Argument Mining
Mohamed Elaraby and Diane Litman 6187

Semantic Overlap Summarization among Multiple Alternative Narratives: An Exploratory Study
Naman Bansal, Mousumi Akter and Shubhra Kanti Karmaker Santu 6195

Analyzing the Dialect Diversity in Multi-document Summaries
Olubusayo Olabisi, Aaron Hudson, Antonie Jetter and Ameeta Agrawal 6208

Multi-Document Scientific Summarization from a Knowledge Graph-Centric View
Pancheng Wang, Shasha Li, Kunyuan Pang, Liangliang He, Dong Li, Jintao Tang and Ting Wang 6222

Generation of Patient After-Visit Summaries to Support Physicians
Pengshan Cai, Fei Liu, Adarsha Bajracharya, Joe Sills, Alok Kapoor, Weisong Liu, Dan Berlowitz, David Levy, Richeek Pradhan and Hong Yu 6234

HeterGraphLongSum: Heterogeneous Graph Neural Network with Passage Aggregation for Extractive Long Document Summarization
Tuan-Anh Phan, Ngoc-Dung Ngoc Nguyen and Khac-Hoai Nam Bui 6248

GRETEL: Graph Contrastive Topic Enhanced Language Model for Long Document Extractive Summarization
Qianqian Xie, Jimin Huang, Tulika Saha and Sophia Ananiadou 6259

PINEAPPLE: Personifying INanimate Entities by Acquiring Parallel Personification Data for Learning Enhanced Generation
Sedrick Scott Keh, Kevin Lu, Varun Gangal, Steven Y. Feng, Harsh Jhamtani, Malihe Alikhani and Eduard Hovy 6270

Mind the Gap! Injecting Commonsense Knowledge for Abstractive Dialogue Summarization
Seungone Kim, Se June Joo, Hyungjoo Chae, Chaehyeong Kim, Seung-won Hwang and Jinyoung Yeo 6285

Type-dependent Prompt CycleQAG : Cycle Consistency for Multi-hop Question Generation
Seungyeon Lee and Minho Lee 6301

UPER: Boosting Multi-Document Summarization with an Unsupervised Prompt-based Extractor
Shangqing Tu, Jifan Yu, Fangwei Zhu, Juanzi Li, Lei Hou and Jian-Yun Nie 6315

DISK: Domain-constrained Instance Sketch for Math Word Problem Generation
Tianyang Cao, Shuang Zeng, Xiaodan Xu, Mairgup Mansur and Baobao Chang 6327

Context-Tuning: Learning Contextualized Prompts for Natural Language Generation
Tianyi Tang, Junyi Li, Wayne Xin Zhao and Ji-Rong Wen 6340

PSP: Pre-trained Soft Prompts for Few-Shot Abstractive Summarization
Xiaochen Liu, Yang Gao, Yu Bai, Jiawei Li, Yinan Hu, Heyan Huang and Boxing Chen 6355

Continuous Decomposition of Granularity for Neural Paraphrase Generation
Xiaodong Gu, Zhaowei Zhang, Sang-Woo Lee, Kang Min Yoo and Jung-Woo Ha 6369

Paraphrase Generation as Unsupervised Machine Translation
Xiaofei Sun, Yufei Tian, Yuxian Meng, Nanyun Peng, Fei Wu, Jiwei Li and Chun Fan 6379

Summarize, Outline, and Elaborate: Long-Text Generation via Hierarchical Supervision from Extractive Summaries
Xiaofei Sun, Zijun Sun, Yuxian Meng, Jiwei Li and Chun Fan 6392

CoCGAN: Contrastive Learning for Adversarial Category Text Generation
Xin Sheng, Linli Xu, Yinlong Xu, Changcun Bao, Huang Chen and Bo Ren 6403

An Efficient Coarse-to-Fine Facet-Aware Unsupervised Summarization Framework Based on Semantic Blocks
Xinnian Liang, Jing Li, Shuangzhi Wu, Jiali Zeng, Yufan Jiang, Mu Li and Zhoujun Li 6415

CHAE: Fine-Grained Controllable Story Generation with Characters, Actions and Emotions
Xinpeng Wang, Han Jiang, Zhihua Wei and Shanlin Zhou 6426

Chinese Couplet Generation with Syntactic Information
Yan Song 6436

Noise-injected Consistency Training and Entropy-constrained Pseudo Labeling for Semi-supervised Extractive Summarization
Yiming Wang, Qianren Mao, Junnan Liu, Weifeng Jiang, Hongdong Zhu and Jianxin Li 6447

Question Generation Based on Grammar Knowledge and Fine-grained Classification
Yuan Sun, Sisi Liu, Zhengcuo Dan and Xiaobing Zhao 6457

CM-Gen: A Neural Framework for Chinese Metaphor Generation with Explicit Context Modelling
Yucheng Li, Chenghua Lin and Frank Guerin 6468

Psychology-guided Controllable Story Generation
Yuqiang Xie, Yue Hu, Yunpeng Li, Guanqun Bi, Luxi Xing and Wei Peng 6480

Few-shot Table-to-text Generation with Prefix-Controlled Generator
Yutao Luo, Menghua Lu, Gongshen Liu and Shilin Wang 6493

Text Simplification of College Admissions Instructions: A Professionally Simplified and Verified Corpus
Zachary W. Taylor, Maximus H. Chu and Junyi Jessy Li 6505

On the Role of Pre-trained Language Models in Word Ordering: A Case Study with BART
Zebin Ou, Meishan Zhang and Yue Zhang 6516

Visual Information Guided Zero-Shot Paraphrase Generation
Zhe Lin and Xiaojun Wan 6530

Improving Abstractive Dialogue Summarization with Speaker-Aware Supervised Contrastive Learning
Zhichao Geng, Ming Zhong, Zhangyue Yin, Xipeng Qiu and Xuanjing Huang 6540

Diversifying Neural Text Generation with Part-of-Speech Guided Softmax and Sampling
Zhixian Yang, Pengxuan Xu and Xiaojun Wan 6547

Enhancing Pre-trained Models with Text Structure Knowledge for Question Generation
Zichen Wu, Xin Jia, Fanyi Qu and Yunfang Wu 6564

LFKQG: A Controlled Generation Framework with Local Fine-tuning for Question Generation over Knowledge Bases
Zichu Fei, Xin Zhou, Tao Gui, Qi Zhang and Xuanjing Huang 6575

Demystifying Neural Fake News via Linguistic Feature-Based Interpretation
Ankit Aich, Souvik Bhattacharya and Natalie Parde 6586

Measuring Geographic Performance Disparities of Offensive Language Classifiers
Brandon Lwowski, Paul Rad and Anthony Rios 6600

Offensive Content Detection via Synthetic Code-Switched Text
Cesa Salaam, Franck Dernoncourt, Trung Bui, Danda Rawat and Seunghyun Yoon 6617

A Survey on Multimodal Disinformation Detection
Firoj Alam, Stefano Cresci, Tanmoy Chakraborty, Fabrizio Silvestri, Dimiter Dimitrov, Giovanni Da San Martino, Shaden Shaar, Hamed Firooz and Preslav Nakov 6625

Why Is It Hate Speech? Masked Rationale Prediction for Explainable Hate Speech Detection
Jiyun Kim, Byounghan Lee and Kyung-Ah Sohn 6644

Domain Classification-based Source-specific Term Penalization for Domain Adaptation in Hate-speech Detection
Tulika Bose, Nikolaos Aletras, Irina Illina and Dominique Fohr 6656

Generalizable Implicit Hate Speech Detection Using Contrastive Learning
Youngwook Kim, Shinwoo Park and Yo-Sub Han 6667

Social Bot-Aware Graph Neural Network for Early Rumor Detection
Zhen Huang, Zhilong Lv, Xiaoyun Han, Binyang Li, Menglong Lu and Dongsheng Li 6680

A Contrastive Cross-Channel Data Augmentation Framework for Aspect-Based Sentiment Analysis
Bing Wang, Liang Ding, Qihuang Zhong, Ximing Li and Dacheng Tao 6691

Sentiment Interpretable Logic Tensor Network for Aspect-Term Sentiment Analysis
Bowen Zhang, Xu Huang, Zhichao Huang, Hu Huang, Baoquan Zhang, Xianghua Fu and Liwen Jing 6705

Detecting Minority Arguments for Mutual Understanding: A Moderation Tool for the Online Climate Change Debate
Cedric Waterschoot, Ernst van den Hemel and Antal van den Bosch 6715

A Multi-turn Machine Reading Comprehension Framework with Rethink Mechanism for Emotion-Cause Pair Extraction
Changzhi Zhou, Dandan Song, Jing Xu and Zhijing Wu 6726

Structural Bias for Aspect Sentiment Triplet Extraction
Chen Zhang, Lei Ren, Fang Ma, Jingang Wang, Wei Wu and Dawei Song 6736

Unsupervised Data Augmentation for Aspect Based Sentiment Analysis
David Z. Chen, Adam Faulkner and Sahil Badyal 6746

A Sentiment and Emotion Aware Multimodal Multiparty Humor Recognition in Multilingual Conversational Setting
Dushyant Singh Chauhan, Gopendra Vikram Singh, Aseem Arora, Asif Ekbal and Pushpak Bhattacharyya 6752

TSAM: A Two-Stream Attention Model for Causal Emotion Entailment
Duzhen Zhang, Zhen Yang, Fandong Meng, Xiuyi Chen and Jie Zhou 6762

Entity-Level Sentiment Analysis (ELSA): An Exploratory Task Survey
Egil Rønningstad, Erik Velldal and Lilja Øvrelid 6773

Learning from Adjective-Noun Pairs: A Knowledge-enhanced Framework for Target-Oriented Multimodal Sentiment Classification
Fei Zhao, Zhen Wu, Siyu Long, Xinyu Dai, Shujian Huang and Jiajun Chen 6784

Towards Exploiting Sticker for Multimodal Sentiment Analysis in Social Media: A New Dataset and Baseline
Feng Ge, Weizhao Li, Haopeng Ren and Yi Cai 6795

Natural Language Inference Prompts for Zero-shot Emotion Classification in Text across Corpora
Flor Miriam Plaza-del-Arco, María-Teresa Martín-Valdivia and Roman Klinger 6805

CommunityLM: Probing Partisan Worldviews from Language Models
Hang Jiang, Doug Beeferman, Brandon Roy and Deb Roy 6818

Composition-based Heterogeneous Graph Multi-channel Attention Network for Multi-aspect Multi-sentiment Classification
Hao Niu, Yun Xiong, Jian Gao, Zhongchen Miao, Xiaosu Wang, Hongrun Ren, Yao Zhang and Yangyong Zhu 6827

CoNTACT: A Dutch COVID-19 Adapted BERT for Vaccine Hesitancy and Argumentation Detection
Jens Lemmens, Jens Van Nooten, Tim Kreutz and Walter Daelemans 6837

SSR: Utilizing Simplified Stance Reasoning Process for Robust Stance Detection
Jianhua Yuan, Yanyan Zhao, Yanyue Lu and Bing Qin 6846

Transferring Confluent Knowledge to Argument Mining
João António Rodrigues and António Branco 6859

When to Laugh and How Hard? A Multimodal Approach to Detecting Humor and Its Intensity
Khalid Alnajjar, Mika Hämäläinen, Jörg Tiedemann, Jorma Laaksonen and Mikko Kurimo . . 6875

Modeling Aspect Correlation for Aspect-based Sentiment Analysis via Recurrent Inverse Learning Guidance
Longfeng Li, Haifeng Sun, Qi Qi, Jingyu Wang, Jing Wang and Jianxin Liao 6887

Analyzing Persuasion Strategies of Debaters on Social Media
Matti Wiegmann, Khalid Al Khatib, Vishal Khanna and Benno Stein 6897

KC-ISA: An Implicit Sentiment Analysis Model Combining Knowledge Enhancement and Context Features
Minghao Xu, Daling Wang, Shi Feng, Zhenfei Yang and Yifei Zhang 6906

Domain Generalization for Text Classification with Memory-Based Supervised Contrastive Learning
Qingyu Tan, Ruidan He, Lidong Bing and Hwee Tou Ng 6916

A Zero-Shot Claim Detection Framework Using Question Answering
Revanth Gangi Reddy, Sai Chetan Chinthakindi, Yi R. Fung, Kevin Small and Heng Ji 6927

Asymmetric Mutual Learning for Multi-source Unsupervised Sentiment Adaptation with Dynamic Feature Network
Rui Li, Cheng Liu and Dazhi Jiang 6934

Target Really Matters: Target-aware Contrastive Learning and Consistency Regularization for Few-shot Stance Detection
Rui Liu, Zheng Lin, Huishan Ji, Jiangnan Li, Peng Fu and Weiping Wang 6944

Joint Alignment of Multi-Task Feature and Label Spaces for Emotion Cause Pair Extraction
Shunjie Chen, Xiaochuan Shi, Jingye Li, Shengqiong Wu, Hao Fei, Fei Li and Donghong Ji . 6955

Causal Intervention Improves Implicit Sentiment Analysis
Siyin Wang, Jie Zhou, Changzhi Sun, Junjie Ye, Tao Gui, Qi Zhang and Xuanjing Huang 6966

COMMA-DEER: COmmon-sense Aware Multimodal Multitask Approach for Detection of Emotion and Emotional Reasoning in Conversations
Soumitra Ghosh, Gopendra Vikram Singh, Asif Ekbal and Pushpak Bhattacharyya 6978

EmoMent: An Emotion Annotated Mental Health Corpus from Two South Asian Countries
Thushari Atapattu, Mahen Herath, Charitha Elvitigala, Piyanjali de Zoysa, Kasun Gunawardana, Menasha Thilakaratne, Kasun de Zoysa and Katrina Falkner 6991

LEGO-ABSA: A Prompt-based Task Assemblable Unified Generative Framework for Multi-task Aspect-based Sentiment Analysis
Tianhao Gao, Jun Fang, Hanyu Liu, Zhiyuan Liu, Chao Liu, Pengzhang Liu, Yongjun Bao and Weipeng Yan 7002

A Hierarchical Interactive Network for Joint Span-based Aspect-Sentiment Analysis
Wei Chen, Jinglong Du, Zhao Zhang, Fuzhen Zhuang and Zhongshi He 7013

MuCDN: Mutual Conversational Detachment Network for Emotion Recognition in Multi-Party Conversations
Weixiang Zhao, Yanyan Zhao and Bing Qin 7020

UECA-Prompt: Universal Prompt for Emotion Cause Analysis
Xiaopeng Zheng, Zhiyue Liu, Zizhen Zhang, Zhaoyang Wang and Jiahai Wang 7031

One-Teacher and Multiple-Student Knowledge Distillation on Sentiment Classification
Xiaoqin Chang, Sophia Yat Mei Lee, Suyang Zhu, Shoushan Li and Guodong Zhou 7042

Making Parameter-efficient Tuning More Efficient: A Unified Framework for Classification Tasks
Xin Zhou, Ruotian Ma, Yicheng Zou, Xuanting Chen, Tao Gui, Qi Zhang, Xuanjing Huang, Rui Xie and Wei Wu 7053

A Multi-Task Dual-Tree Network for Aspect Sentiment Triplet Extraction
Yichun Zhao, Kui Meng, Gongshen Liu, Jintao Du and Huijia Zhu 7065

Exploiting Unlabeled Data for Target-Oriented Opinion Words Extraction
Yidong Wang, Hao Wu, Ao Liu, Wenxin Hou, Zhen Wu, Jindong Wang, Takahiro Shinozaki, Manabu Okumura and Yue Zhang 7075

Learnable Dependency-based Double Graph Structure for Aspect-based Sentiment Analysis
Yinglong Ma and Yunhe Pang 7086

A Structure-Aware Argument Encoder for Literature Discourse Analysis
Yinzi Li, Wei Chen, Zhongyu Wei, Yujun Huang, Chujun Wang, Siyuan Wang, Qi Zhang, Xuanjing Huang and Libo Wu 7093

Mere Contrastive Learning for Cross-Domain Sentiment Analysis
Yun Luo, Fang Guo, Zihan Liu and Yue Zhang 7099

Exploiting Sentiment and Common Sense for Zero-shot Stance Detection
Yun Luo, Zihan Liu, Yuefeng Shi, Stan Z. Li and Yue Zhang 7112

Modeling Intra- and Inter-Modal Relations: Hierarchical Graph Contrastive Learning for Multimodal Sentiment Analysis
Zijie Lin, Bin Liang, Yunfei Long, Yixue Dang, Min Yang, Min Zhang and Ruifeng Xu 7124

AMOA: Global Acoustic Feature Enhanced Modal-Order-Aware Network for Multimodal Sentiment Analysis
Ziming Li, Yan Zhou, Weibo Zhang, Yaxin Liu, Chuanpeng Yang, Zheng Lian and Songlin Hu7136

Keyphrase Prediction from Video Transcripts: New Dataset and Directions
Amir Pouran Ben Veyseh, Quan Hung Tran, Seunghyun Yoon, Varun Manjunatha, Hanieh Deilamsalehy, Rajiv Jain, Trung Bui, Walter W. Chang, Franck Dernoncourt and Thien Huu Nguyen 7147

Event Extraction in Video Transcripts
Amir Pouran Ben Veyseh, Viet Dac Lai, Franck Dernoncourt and Thien Huu Nguyen 7156

Recycle Your Wav2Vec2 Codebook: A Speech Perceiver for Keyword Spotting
Guillermo Cámbara, Jordi Luque and Mireia Farrús 7166

Improving Code-switched ASR with Linguistic Information
Jie Chi and Peter Bell 7171

Language-specific Effects on Automatic Speech Recognition Errors for World Englishes
June Choe, Yiran Chen, May Pik Yu Chan, Aini Li, Xin Gao and Nicole Holliday 7177

A Transformer-based Threshold-Free Framework for Multi-Intent NLU
Lisung Chen, Nuo Chen, Yuexian Zou, Yong Wang and Xinzhong Sun 7187

Unsupervised Multi-scale Expressive Speaking Style Modeling with Hierarchical Context Information for Audiobook Speech Synthesis
Xueyuan Chen, Shun Lei, Zhiyong Wu, Dong Xu, Weifeng Zhao and Helen Meng 7193

Incorporating Instructional Prompts into a Unified Generative Framework for Joint Multiple Intent Detection and Slot Filling
Yangjun Wu, Han Wang, Dongxiang Zhang, Gang Chen and Hao Zhang 7203

Adaptive Unsupervised Self-training for Disfluency Detection
Zhongyuan Wang, Yixuan Wang, Shaolei Wang and Wanxiang Che 7209

Do language models make human-like predictions about the coreferents of Italian anaphoric zero pronouns?

James A. Michaelov
Department of Cognitive Science
University of California, San Diego
j1michae@ucsd.edu

Benjamin K. Bergen
Department of Cognitive Science
University of California, San Diego
bkbergen@ucsd.edu

Abstract

Some languages allow arguments to be omitted in certain contexts. Yet human language comprehenders reliably infer the intended referents of these *zero* pronouns, in part because they construct expectations about which referents are more likely. We ask whether Neural Language Models also extract the same expectations. We test whether 12 contemporary language models display expectations that reflect human behavior when exposed to sentences with zero pronouns from five behavioral experiments conducted in Italian by Carminati (2005). We find that three models—XGLM 2.9B, 4.5B, and 7.5B—capture the human behavior from all the experiments, with others successfully modeling some of the results. This result suggests that human expectations about coreference can be derived from exposure to language, and also indicates features of language models that allow them to better reflect human behavior.

1 Introduction

In Italian, like other *pro-drop* ('pronoun-dropping') languages, verbal arguments that would usually be expressed by pronouns in languages such as English can be omitted under certain circumstances. For example, consider the sentence in (1) from Carminati (2005).

(1) *Quando Maria ha chiamato Mario, era contenta*.
'When Maria called Mario, [she] was happy.'

In this sentence, the referent of the 'dropped' pronoun—generally referred to as a *zero* or *null* pronoun—can be inferred from the fact that the adjective *contenta* is feminine; thus, Maria is the most likely subject of the second clause. Resolving the referents of anaphoric zero pronouns like this is a long-standing, important, and active area of research in natural language understanding (Jurafsky and Martin, 2021; for examples, see Zhao and Ng, 2007; Taira et al., 2008; Imamura et al., 2009; Watanabe et al., 2010; Kong and Zhou, 2010; Poesio et al., 2010; Chen and Ng, 2013; Yoshino et al., 2013; Iida et al., 2016; Aloraini and Poesio, 2020; Song et al., 2020a; Ueda et al., 2020; Konno et al., 2020, 2021; Yang et al., 2020; Kim et al., 2021; Umakoshi et al., 2021; Chen et al., 2021; Yang et al., 2022).

It has been argued that aiming for human-likeness in natural language processing systems is vital if we want our natural language understanding systems to behave not only as humans do, but also as human users expect them to (see, e.g., Keller, 2010; Ettinger, 2020; Eisape et al., 2020). This is particularly true for zero anaphora resolution, and pronoun resolution more generally. As an illustration of the prominence of reference resolution, one pronoun resolution task, the Winograd Schema Challenge (Levesque et al., 2012; based on work by Winograd, 1972), has been referred to as 'an alternative to the Turing Test' (Levesque et al., 2012, p. 552).

So, how do humans resolve coreference? The evidence suggests that we use a range of cues—for example, agreement information as in (1), but also factors such as world knowledge and common sense (Winograd, 1972; Hobbs, 1979; Kehler et al., 2007; Kehler and Rohde, 2013; Sakaguchi et al., 2019). In addition, pronoun resolution is shaped by our expectations about the next entity that is likely to be mentioned and what argument it should take (Kehler et al., 2007; Kehler and Rohde, 2013; Nieuwland and Van Berkum, 2006). For example, crosslinguistic work has found a bias towards expecting a subject pronoun to refer to an antecedent subject (for discussion, see Carminati, 2005). This has been demonstrated experimentally with sentences such as those in (2).

(2) *John seized the comic from Bill. He____*

When presenting experimental participants with

Proceedings of the 29th International Conference on Computational Linguistics, pages 1–14
October 12–17, 2022.

sentences such as (2) where the content following the pronoun has been removed, Stevenson et al. (1994) found that the vast majority of people expect *he* to refer to *John* rather than *Bill*. The effect of expectations such as these are so powerful that we may often not even realize that a sentence is grammatically ambiguous at all in most situations (Nieuwland and Van Berkum, 2006).

The same principles apply in zero anaphora resolution. Carminati (2005), for example, tests human expectations by investigating how long it takes for experimental participants to read stimuli with certain linguistic features, based on the well-established finding that contextually expected words are read faster than unexpected words, demonstrating an increased processing difficulty when these expectations are violated (see, e.g. Forster, 1981; Levy, 2008; Luke and Christianson, 2016; Brothers and Kuperberg, 2021). Carminati (2005) finds that the main clauses of sentences such as (1)—that is, the part of the sentence containing the zero subject pronoun, i.e., *era contenta* ('[she] was happy')—are read faster when the zero pronoun co-refers with a subject antecedent, as in (1), than when it co-refers with the antecedent object, as in (3).

(3) *Quando Maria ha chiamato Mario, era contento*.
'When Maria called Mario, [he] was happy.'

The question, then, if we want human-like zero anaphora resolution, is how to test whether a given zero anaphora resolution system is able to reflect these human expectations. In the present study, we propose a method to do just that.

The vast majority of recent pronoun resolution systems base their approach around using the representations learned by contemporary transformer language models—for example, in the zero pronoun anaphora resolution literature alone, researchers have used pretrained transformers such as monolingual (Song et al., 2020b; Ueda et al., 2020; Konno et al., 2020, 2021; Kim et al., 2021; Chen et al., 2021; Umakoshi et al., 2021) and multilingual (Aloraini and Poesio, 2020; Kim et al., 2021) BERT models (Devlin et al., 2019), as well as XLM-R (Conneau et al., 2020; for an example see Yang et al., 2022).

For these systems, there is a clear way to test for human-like-ness. We can directly probe the extent to which the representations learned by the language models take into account the factors that lead to coreference expectations in humans by testing how similar the predictions of language models are to those of human comprehenders—if they exhibit the same pattern of predictive behavior as humans in a given context, this demonstrates that they are sensitive to the same factors as humans in this context. We do this by comparing the reading times reported by Carminati (2005) to the surprisals of 12 contemporary transformer language models (Devlin et al., 2019; Conneau and Lample, 2019; De Mattei et al., 2020; Schweter, 2020; Conneau et al., 2020; de Vries and Nissim, 2021; Lin et al., 2021) for the same stimuli.

A key question is whether it is possible for language models to learn any of these human-like expectations at all, given that they can only rely on the statistics of language. For this reason, the results of the present study should be of interest both from a natural language understanding perspective, as discussed above, and also from a psycholinguistics perspective.

From the natural language understanding perspective, the present study presents an approach for 'pre-evaluating' a language model's suitability as a basis for a zero anaphora resolution system. Specifically, if a language model can model a specific effect in human language processing—that is, if an experimental manipulation that elicits a significant difference in reading time also results in a significant difference in that language model's surprisal in the same direction—this demonstrates that it is able to take into account the relevant factors that underlie human comprehender expectation. For example, if a language model can successfully model the subject antecedent preference, this suggests that it has learned that all else being equal, subject antecedents are more likely to be the coreferents of zero subject pronouns, and thus, crucially, that this pattern is in some way represented in the contextual embeddings that can be used as the representations underlying a zero anaphora resolution system.

From the psycholinguistics perspective, this study explores the extent to which it is possible that specific patterns in zero anaphora coreference expectations can be learned on the basis of the statistics of language alone. There is substantial work demonstrating that some expectations are highly correlated with language statistics, and thus may be at least partly derived from them (Levy, 2008; Monsalve et al., 2012; Smith and Levy, 2013;

Frank et al., 2015; Michaelov and Bergen, 2020; Szewczyk and Federmeier, 2022). However, other work has suggested that coreference expectations are instead (or in addition) at least partly based on semantic knowledge, world experience, and conceptual salience (Hobbs, 1979; Harley and Ritter, 2002; Carminati, 2005; Kehler et al., 2007; Kehler and Rohde, 2013). Nonetheless, since the predictions of language models are derived from language statistics alone, if even one language model can successfully model a given effect (after adjusting for multiple comparisons), this provides in-principle evidence that the effect can be successfully learned using distributional information alone.

2 General Method

The experiments reported by Carminati (2005) were self-paced reading experiments. Participants were native speakers of Italian asked to read Italian sentences on a computer. Stimuli were similar to those discussed in the previous section, with a subordinate clause (e.g. *Quando Maria ha chiamato Mario*; 'When Maria called Mario') first presented, followed by the main clause (e.g. either *era contenta* '[she] was happy' or *era contento* '[he] was happy'). The time taken by participants to read the main clause—which includes the word that disambiguates the null subject pronoun—was recorded.

To measure the language model's expectations, we used surprisal (negative log-probability) based on a large body of evidence that language model surprisal generally correlates well with reading time (see, e.g. Levy, 2008; Monsalve et al., 2012; Smith and Levy, 2013; Goodkind and Bicknell, 2018) and other metrics of processing difficulty that are thought to correlate with human expectations such as the neural N400 response (Frank et al., 2015; Aurnhammer and Frank, 2019; Michaelov and Bergen, 2020; Merkx and Frank, 2021).

To model each effect, we compared whether specific linguistic features of the stimuli that elicited a significant difference in human reading times also led to a significant difference in language model surprisal. For example, we investigate whether, like reading time, surprisal is significantly lower when the referent of a zero subject pronoun is a subject antecedent compared to an object antecedent, among other patterns in reading time reported by Carminati (2005). The language models were all presented with the same stimuli as the human participants, which are provided by Carminati (2005) in an appendix to the original paper.

To match reading time, surprisal was calculated over the whole of the main clause in each stimulus item. This was done by calculating the sum of the surprisals of the main clauses' constituent words, which is equivalent to taking the negative logarithm of the product of their probabilities.

We ran the stimuli through 12 transformer language models—5 monolingual and 7 multilingual. Two of the monolingual models were autoregressive transformer networks: GePpeTto (De Mattei et al., 2020) and the small English GPT-2 retrained on Italian (de Vries and Nissim, 2021). The three remaining monolingual models were masked language models: UmBERTo (Parisi et al., 2021) trained on the Italian subcorpus of OSCAR, and the Base and XXL versions of the Italian BERT models (Schweter, 2020). The multilingual models also included autoregressive and masked language models. The autoregressive models were three different sizes of XGLM (Lin et al., 2021): the 2.9B, 4.5B, and 7.5B parameter models. The masked language models were XLM-100 (Conneau and Lample, 2019), and the Base and Large versions of XLM-R (Conneau et al., 2020).

The aim in using this range of models was to test whether there are any model types or characteristics made them better suited to capturing human behavior—for instance, whether the models were autoregressive or masked, or monolingual or multilingual. Previous systems designed to resolve zero pronoun anaphora of the kind described here appear to be predominantly based on masked language models; however, autoregressive models such as GPT-2 have been successfully used in similar systems (Maqbool et al., 2022).

We are also interested in whether monolingual or multilingual models are better suited to the task of zero pronoun anaphora resolution—while cross-lingual transfer may help with some phenomena (Guarasci et al., 2022), there is also evidence that it can cause harm to model performance in others (Wang et al., 2020). There is currently mixed evidence with respect to zero pronoun anaphora resolution—Kim et al. (2021), for example, find that a monolingual Korean BERT-based model performs better than the standard multilingual BERT model; while Yang et al. (2022) finds that their model, based on XLM-R, is better than a model based on a Chinese-only BERT (Song et al., 2020b). We include both multilingual BERT and XLM-R

in our analyses, in addition to the Base Italian BERT model, which has previously been evaluated in terms of its capacity to learn non-anaphoric null subject and agreement phenomena in Italian (Guarasci et al., 2021).

To test whether each model successfully modeled each effect, we constructed linear mixed-effects models predicting model surprisal with experimental manipulation as a main effect and a random intercept of sentence frame, where sentence frame refers to a set of stimuli that differ only by experimental condition (e.g., the previously discussed *Quando Maria ha chiamato Mario, era contenta* and *Quando Maria ha chiamato Mario, era contento* are two stimuli with the same sentence frame).

For three of the five analyses—the two analyses in Section 3.1 where the coreferent is distinguished by gender, and the analysis in Section 3.2—we tested whether the relevant experimental manipulation was a significant predictor of language model surprisal by constructing a null regression with only the random intercept of sentence frame and running a likelihood ratio test investigating whether adding the experimental manipulation improved model fit.

The remaining two analyses correspond to two different tests utilized by Carminati (2005) to analyze the results of a single experiment (Experiment 4 of the original paper). Crucially, Carminati (2005) tests whether there is an interaction between coreferent argument (whether it is the antecedent subject or object) and coreferent person (whether the coreferent is in the first or second person or in the third person), but also whether there is a main effect of each of these. To test whether there is an interaction (in Section 3.3), we construct a linear mixed-effects model with and without the interaction, and run a likelihood ratio test comparing the two. In addition to the interaction, Carminati (2005) finds a main effect of coreferent argument but not of person. Thus, we also test for the main effect of coreferent argument, which we report in Section 3.1. Because we want to investigate whether the main effect of coreferent argument explains a significant amount of the variance in surprisal while also accounting for the effect of a possible interaction, instead of using a likelihood ratio test, we opt for a Type III ANOVA with Satterthwaite's method for estimating degrees of freedom (Kuznetsova et al., 2017).

The details of the results of the statistical analyses that were run by Carminati (2005) are provided in the original paper. The full results of the statistical analyses that we ran are provided in Appendix A. The results of both sets of statistical analyses are summarized in Figure 1.

All language models were run in *Python* (Van Rossum and Drake, 2009), using the *PyTorch* (Paszke et al., 2019) implementation of each model, as provided by the *transformers* package (Wolf et al., 2020). Statistical analysis and data manipulation were carried out in *R* (R Core Team, 2020) using *Rstudio* (RStudio Team, 2020) and the *tidyverse* (Wickham et al., 2019), *lme4* (Bates et al., 2015), *lmerTest* (Kuznetsova et al., 2017), *ggsignif* (Ahlmann-Eltze and Patil, 2021), and *cowplot* (Wilke, 2020) packages. The stimuli, code used to run the models, and code used to run the statistical analyses are provided on Github[1]. Note that all p-values reported in this analysis have been corrected for multiple comparisons (Benjamini and Hochberg, 1995; R Core Team, 2020).

3 Manipulation-level results and discussion

In this section, we compare the performance of the 12 language models tested with human behavior on five of the experimental manipulations carried out by Carminati (2005). Note that two additional studies from that paper focus on a different question—the effects of distractor referents on processing time. Although at least one model was able to capture each of these human results, they are not included here because they address a different set of phenomena.

3.1 Subject vs. object antecedent referent

Carminati (2005) investigates the subject antecedent preference discussed in Section 1 in three experiments. In Experiments 1 and 2 of the original paper, both antecedents are names associated with different genders, as illustrated by the example from Experiment 1 shown in (4).

(4) (a) *Quando Lucia ha telefonato a Marco, era appena tornata da Londra.*
'When Lucia has telephoned Marco, [she] had just come back from London.'

(b) *Quando Lucia ha telefonato a Marco, era appena tornato da Londra.*
'When Lucia has telephoned Marco, [he] had just come back from London.'

[1] https://github.com/jmichaelov/italian-zero-anaphora-prediction

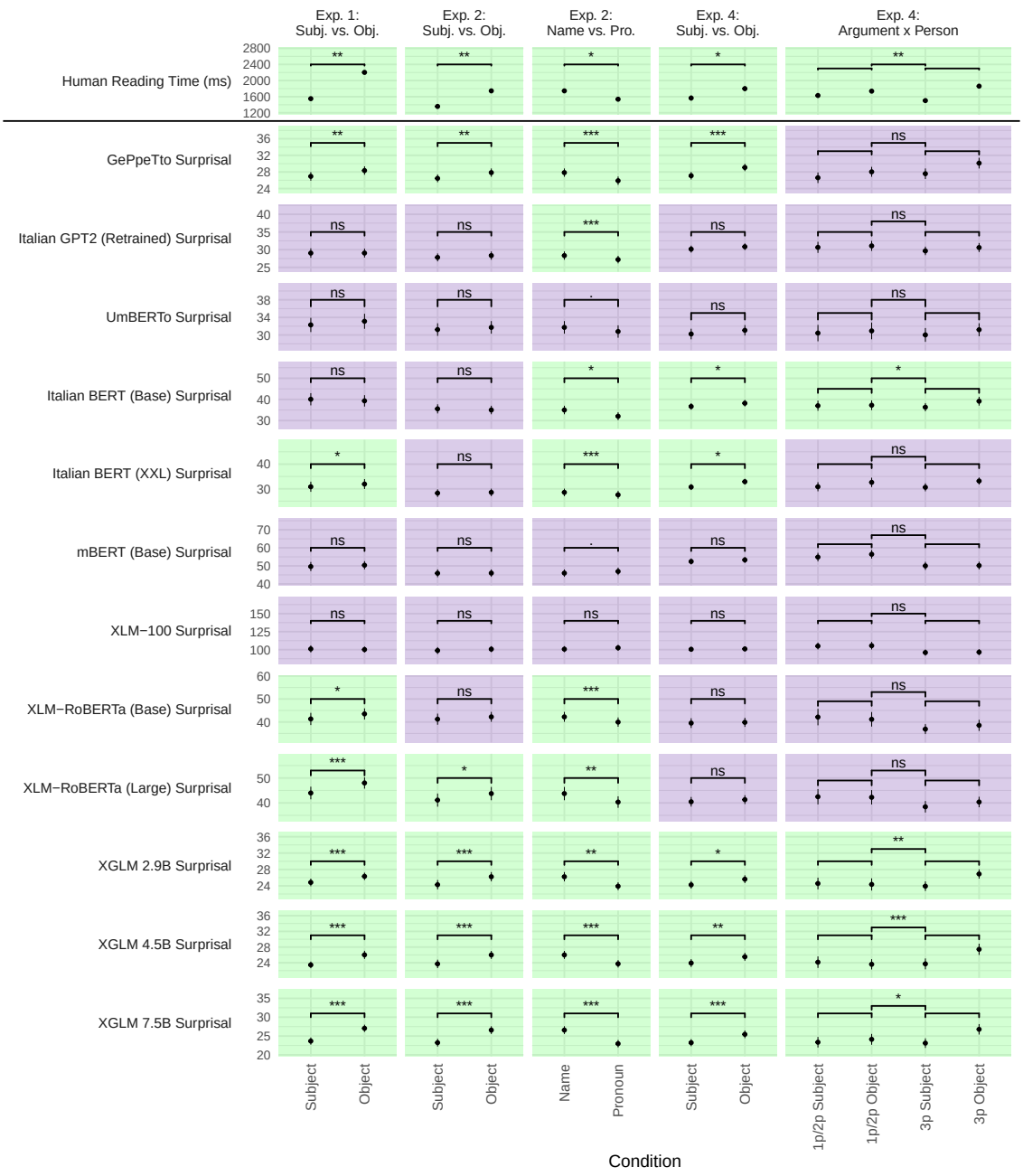

Figure 1: Mean reading time and surprisal of each model elicited by main clauses for each experimental condition in each experiment. All significant differences are shown: following convention, '***' indicates $p < 0.001$, '**' indicates $p < 0.01$, '*' indicates $p < 0.5$, '.' indicates marginal significance where $p < 0.1$, and 'ns' indicates $p \geq 0.1$. For easier comparison across models and experiments, comparisons with statistically significant results are colored green; non-significant results are colored purple. Note that the relevant p-values have been corrected for multiple comparisons using the method of Benjamini and Hochberg (1995); for test statistics and degrees of freedom, see Appendix A. Details of the statistical tests for reading time are provided by Carminati (2005). For language model surprisal, error bars indicate standard error; no metric of error is provided by Carminati (2005).

Because *tornato/tornata* ('come back') agrees with the gender of the zero subject pronoun, its referent can be resolved to be the subject antecedent (*Lucia*) in (4a) and the object antecedent (*Marco*) in (4b). Carminati (2005) found, as expected, that main clauses where the zero subject pronoun coreferred with the subject antecedent (like (4a)) were read faster than those where they had an object antecedent coreferent (like (4b)), suggesting an expectation for a subject antecedent coreferent.

In Experiment 4 of the original paper, grammatical person was manipulated rather than grammati-

cal gender, as illustrated by the example in (5).

(5) (a) *Quando ho litigato con Maria, ero molto prepotente.*
'When [I] quarrelled with Maria, [I] was very pushy.'

(b) *Quando ho litigato con Maria, era molto prepotente.*
'When [I] quarrelled with Maria, [she] was very pushy.'

Similarly, because *ero/era* ('was') either agrees with the first person or third person, the zero subject pronoun can be resolved as co-referring with the speaker (in (5a)) or with Maria (in (5b)). As in the aforementioned other experiments, Carminati (2005) finds that speakers read sentences like (5a) faster than sentences like (5b), again demonstrating a preference for subject antecedent coreferents over object antecedent coreferents.

Looking at the results of the models, we can see that only GePpeTto and the XGLMs successfully model this effect in all three experiments. This appears to suggest that autoregressive models may be better at learning that the subject antecedent is the more likely referent; however, it should be noted that in each of the individual studies, at least one masked language model also successfully modeled the effect. Nonetheless, the robustness of similarity between these autoregressive models' predictions and human expectations may be partly explained by the evidence suggesting that autoregressive models are more sensitive to word order than masked language models, to the extent that they are able to encode positional information even without explicit positional encodings (Haviv et al., 2022); conversely, masked language models appear to be relatively insensitive to word order (Sinha et al., 2021; Gupta et al., 2021). Given that the dominant pattern in Italian is Subject-Verb-Object (see Guarasci et al., 2022) and the subject was always first in the subordinate clause, it is therefore unsurprising that autoregressive models would be better able to predict that the first entity mentioned (the subject) is more likely as the subject of the zero pronoun than the second entity mentioned (the object).

3.2 Name vs. pronoun antecedent referent

In addition to investigating the differences in how humans process zero anaphora in sentences with subject and object antecendent coreferents, Carminati (2005) also investigated how the form in which antecedents are presented impacts processing. As a further part of Experiment 2 of the original paper, Carminati (2005) investigates how processing is impacted when the object coreferent is presented as a name or a pronoun, an example of which is provided in (6).

(6) (a) *Quando Maria cerca Roberto, diventa ansioso.*
'When Maria looks for Roberto, [he] becomes anxious.'

(b) *Quando Maria lo cerca, diventa ansioso.*
'When Maria looks for him, [he] becomes anxious.'

In both sentences, it is the object antecedent that is the referent of the zero pronoun in the main clause, violating the subject antecedent preference. Carminati (2005) finds that main clauses with zero pronouns referring to antecedent objects are easier to process (read faster) when this antecedent object is a pronoun.

The results for the language models, shown in Figure 1, suggest that this is a relatively easy pattern for language models to learn—9 of the 12 models show a significant effect and the remaining 3 show a marginal effect in the correct direction. Thus it is clear that this general rule—that an entity referred to by an antecedent pronoun is more likely to be the referent of a zero pronoun—is possible to learn based on the statistics of language. The fact that this effect relies on the form of the antecedents rather than word order could explain why there is no difference between autoregressive and masked language models in this case.

3.3 Antecedent argument by grammatical person interaction

In addition to investigating the subject antecedent effect, in Experiment 4 of the original paper, Carminati (2005) investigates how this effect interacts with the the grammatical person of antecedents (i.e., first, second, or third-person). In general, previous work suggests that first and second-person antecedents are more likely to be referents of reduced or zero pronouns (Ariel, 1991; Siewierska, 1999, 2003; Carminati, 2005), but as has been discussed, subject antecedents are also more likely to be their referents. Thus, Carminati (2005) compares the effect of the person of the coreferent antecedent when it is in both subject and object position, as exemplified in (7).

(7) (a) *Quando ho/hai litigato con Maria, ero/eri molto prepotente.*
'When [I/you] quarrelled with Maria, [I/you] was/were very pushy.'

(b) *Quando Maria ha litigato con me/te, ero/eri molto prepotente.*
'When Maria quarrelled with me/you, [I/you] was/were very pushy.'

(c) *Quando Maria ha litigato con me/te, era molto prepotente.*
'When Maria quarrelled with me/you, [she] was very pushy.'

(d) *Quando ho/hai litigato con Maria, era molto prepotente.*
'When [I/you] quarrelled with Maria, [she] was very pushy.'

While Carminati (2005) does not find a main effect of grammatical person, the results show an interaction between person and antecedent referent argument status (i.e. whether it is a subject or object). Specifically, the difference in reading time between subject and object antecedent referents is reduced when the antecedent coreferent is in the first or second person. In other words, the subject antecedent effect is weaker with first and second person coreferents. This, Carminati (2005) argues, shows that the bias towards a first or second-person coreferent modulates the bias against an object coreferent—in other words, humans still expect a first or second-person coreferent even if it is an object antecedent.

Four of the models—Italian BERT Base and the XGLMs—manage to model this interaction. While this suggests the the effect—which is complicated as it relies on correctly weighting the effects of argument status and person—is difficult to learn based on the statistics of language, it nevertheless demonstrates that it is indeed possible.

4 General Discussion

4.1 Implications for human language processing

We can now return to the two questions that motivated this work. First, we look at whether the reading time effects in humans can be explained on the basis of the statistics of language.

As seen in Figure 1, each experimental result was successfully modeled by at least four language models, after correcting for multiple comparisons. This shows that it is possible to learn cues based on the statistics of language that result in human-like expectations about the referents of zero subject pronouns in Italian. The fact that the XGLM transformers were consistently able to model all the effects demonstrates that the patterns underlying the results of the experiments can all be learned by the same system—and therefore, in principle, it should also be possible for a neurocognitive system implementing lexical prediction in humans (for accounts of such a system and what it might learn, see, e.g., Kutas et al., 2011; Lewis and Bastiaansen, 2015; Lupyan and Clark, 2015; Frank et al., 2015; Bornkessel-Schlesewsky and Schlesewsky, 2019; Aurnhammer and Frank, 2019; Michaelov and Bergen, 2020; Kuperberg et al., 2020; Merkx and Frank, 2021; Brothers and Kuperberg, 2021). Thus, the present study provides evidence that the expectations that humans form about possible referents in anaphora may be derived from language statistics, at least in part.

4.2 Implications for work on language models

Model	Experiments modeled
GePpeTto	4/5
It. GPT2 (Retrained)	1/5
UmBERTo	0/5
It. BERT (Base)	3/5
It. BERT (XXL)	3/5
mBERT	0/5
XLM-100	0/5
XLM-R (Base)	2/5
XLM-R (Large)	3/5
XGLM 2.9B	5/5
XGLM 4.5B	5/5
XGLM 7.5B	5/5

Table 1: Number of experiments successfully modeled by each language model.

The number of experiments successfully modeled by each language model is shown in Table 1, revealing that the XGLM models performs best overall, successfully modeling the results of all 5 experiments investigated. After the XGLMs, GePpeTto models the most experiments (4/5), followed by XLM-R Large and the Italian BERTs (3/5). The remaining transformers only successfully model 2 or fewer of the experiments.

At this level of analysis, some patterns begin to emerge. First, the best models are the XGLM transformers and GePpeTto. This suggests that autore-

gressive models may in fact be best able to model the effects. As discussed in Section 3.1, this may be due to their comparatively high sensitivity to word order. One issue that confounds this interpretation is that the XGLM models are also larger and trained more data on than the other models. However, the fact that GePpeTto was trained on 13GB of text, while the other monolingual models (which were all masked language models) were trained on the same amount or more data and performed worse, suggests that, at the very least, monolingual autoregressive models may more efficiently learn biases in zero anaphora processing than monolingual masked language models. Whether or not autoregressive models continue to out-perform masked language models as the training set increases in size is a question for further research. Overall, then, we see that in our sample of models, autoregressive monolingual and multilingual models are more human-like in their expectations of zero subject pronoun referents than their masked language model counterparts.

Another question that we can address with the present results is that of the effect of multilinguality on the human-likeness of the models' expectations. First, while GePpeTto and Retrained Italian GPT-2 are trained on the same Italian corpus, the former greatly out-performs the latter. This suggests that training a model on one language and then re-training it on another does not necessarily improve the representations that a model learns—in fact, in this case, it interferes with the model's ability to make predictions in a human-like fashion. On the other hand, XLM-R Large is trained on data from 100 languages successfully models human processing at least as well as any monolingual model but GePpeTto—including Italian BERT XXL, which is trained on 80GB of Italian text compared to XLM-R's 30GB. Thus, it may be the case that with more training data, and with a larger number of languages (including more closely-related languages—XLM-R is also trained on other Romance languages), there is some cross-linguistic transfer that can aid in predicting the referent of a null subject pronoun in a human-like manner (see Guarasci et al., 2022, for a recent similar finding). Finally, the XGLMs—autoregressive multilingual models—are the best performing models overall. Thus, the results of this study seem to suggest that with enough overall data, and when multilingual language models are trained on more languages, cross-linguistic transfer can improve their human-likeness in terms of their predictions. A question for future work is to investigate under what circumstances multilinguality hurts or harms the human-likeness of language model predictions—for example, based on how related the languages the model is trained on are to each other, or how widespread the phenomenon under investigation is. For example, the subject antecedent preference is also present in English with overt pronoun anaphora (Smyth, 1994; Chambers and Smyth, 1998; Kehler et al., 2007; Kehler and Rohde, 2013).

Finally, as discussed in Section 2, Yang et al. (2022) show that a zero pronoun anaphora resolution system based on XLM-R performs better than one based on multilingual BERT (Song et al., 2020b). Concurrently, in the present study, we see that either XLM-R model is better able to model zero anaphora processing effects than multilingual BERT. While there are other factors at play, this result is consistent with our prediction that better modeling of human expectations may lead to better performance when using the models' representations for zero pronoun anaphora resolution, based on the idea that the representations learned by the model better allow it to make human-like predictions, and thus are more useful for systems aiming to resolve zero anaphora in a human-like way. In the present study, XGLM models perform better than the other models, and thus, based on this, we suggest that XGLM transformers may be better models upon which to base future zero pronoun anaphora resolution system than other current publicly available pretrained models.

5 Conclusion

We present the first study investigating whether language models make the same predictions as humans when processing zero pronoun anaphora. For each the 5 effects we investigate, we find that there are at least four models that successfully do so; and three models, XGLM 2.9B, 4.5B, and 7.5B, successfully do so in all 5. This suggests that human processing of zero pronoun anaphora may at least partly rely on our statistical knowledge of language. Furthermore, this approach provides a useful way to investigate how human-like the referent predictions of language models are, which is vital if we are to use their representations for zero anaphora resolution systems.

Acknowledgements

We would like to thank Maria Nella Carminati for making her stimuli available, and the anonymous reviewers for their helpful comments. We would also like to thank the other members of the Language and Cognition Lab at UCSD for their valuable discussion and the University of California, and the San Diego Social Sciences Computing Facility Team for technical assistance. This work was partially supported by a 2021-2022 Center for Academic Research and Training in Anthropogeny Annette Merle-Smith Fellowship awarded to James A. Michaelov, and the RTX A5000 used for this research was donated by the NVIDIA Corporation.

References

Constantin Ahlmann-Eltze and Indrajeet Patil. 2021. Ggsignif: R Package for Displaying Significance Brackets for 'ggplot2'.

Abdulrahman Aloraini and Massimo Poesio. 2020. Cross-lingual Zero Pronoun Resolution. In *Proceedings of the 12th Language Resources and Evaluation Conference*, pages 90–98, Marseille, France. European Language Resources Association.

Mira Ariel. 1991. The function of accessibility in a theory of grammar. *Journal of Pragmatics*, 16(5):443–463.

Christoph Aurnhammer and Stefan L. Frank. 2019. Evaluating information-theoretic measures of word prediction in naturalistic sentence reading. *Neuropsychologia*, 134:107198.

Douglas Bates, Martin Mächler, Ben Bolker, and Steve Walker. 2015. Fitting linear mixed-effects models using lme4. *Journal of Statistical Software*, 67(1):1–48.

Yoav Benjamini and Yosef Hochberg. 1995. Controlling the False Discovery Rate: A Practical and Powerful Approach to Multiple Testing. *Journal of the Royal Statistical Society. Series B (Methodological)*, 57(1):289–300.

Ina Bornkessel-Schlesewsky and Matthias Schlesewsky. 2019. Toward a Neurobiologically Plausible Model of Language-Related, Negative Event-Related Potentials. *Frontiers in Psychology*, 10.

Trevor Brothers and Gina R. Kuperberg. 2021. Word predictability effects are linear, not logarithmic: Implications for probabilistic models of sentence comprehension. *Journal of Memory and Language*, 116:104174.

Maria Nella Carminati. 2005. Processing reflexes of the Feature Hierarchy (Person > Number > Gender) and implications for linguistic theory. *Lingua*, 115(3):259–285.

Craig G. Chambers and Ron Smyth. 1998. Structural Parallelism and Discourse Coherence: A Test of Centering Theory. *Journal of Memory and Language*, 39(4):593–608.

Chen Chen and Vincent Ng. 2013. Chinese Zero Pronoun Resolution: Some Recent Advances. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*, pages 1360–1365, Seattle, Washington, USA. Association for Computational Linguistics.

Shisong Chen, Binbin Gu, Jianfeng Qu, Zhixu Li, An Liu, Lei Zhao, and Zhigang Chen. 2021. Tackling Zero Pronoun Resolution and Non-Zero Coreference Resolution Jointly. In *Proceedings of the 25th Conference on Computational Natural Language Learning*, pages 518–527, Online. Association for Computational Linguistics.

Alexis Conneau, Kartikay Khandelwal, Naman Goyal, Vishrav Chaudhary, Guillaume Wenzek, Francisco Guzmán, Edouard Grave, Myle Ott, Luke Zettlemoyer, and Veselin Stoyanov. 2020. Unsupervised Cross-lingual Representation Learning at Scale. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 8440–8451, Online. Association for Computational Linguistics.

Alexis Conneau and Guillaume Lample. 2019. Cross-lingual language model pretraining. In *Advances in Neural Information Processing Systems*, volume 32. Curran Associates, Inc.

Lorenzo De Mattei, Michele Cafagna, Felice Dell'Orletta, Malvina Nissim, and Marco Guerini. 2020. GePpeTto Carves Italian into a Language Model: Italian Conference on Computational Linguistics 2020. In *Proceedings of the Seventh Italian Conference on Computational Linguistics (CLiC-it 2020)*, Bologna, Italy. CEUR-WS.org.

Wietse de Vries and Malvina Nissim. 2021. As Good as New. How to Successfully Recycle English GPT-2 to Make Models for Other Languages. In *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*, pages 836–846, Online. Association for Computational Linguistics.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, Minneapolis, Minnesota. Association for Computational Linguistics.

Tiwalayo Eisape, Noga Zaslavsky, and Roger Levy. 2020. Cloze Distillation: Improving Neural Language Models with Human Next-Word Prediction. In *Proceedings of the 24th Conference on Computational Natural Language Learning*, pages 609–619, Online. Association for Computational Linguistics.

Allyson Ettinger. 2020. What BERT Is Not: Lessons from a New Suite of Psycholinguistic Diagnostics for Language Models. *Transactions of the Association for Computational Linguistics*, 8:34–48.

K. I. Forster. 1981. Priming and the effects of sentence and lexical contexts on naming time: Evidence for autonomous lexical processing. *The Quarterly Journal of Experimental Psychology Section A*, 33(4):465–495.

Stefan L. Frank, Leun J. Otten, Giulia Galli, and Gabriella Vigliocco. 2015. The ERP response to the amount of information conveyed by words in sentences. *Brain and Language*, 140:1–11.

Adam Goodkind and Klinton Bicknell. 2018. Predictive power of word surprisal for reading times is a linear function of language model quality. In *Proceedings of the 8th Workshop on Cognitive Modeling and Computational Linguistics (CMCL 2018)*, pages 10–18, Salt Lake City, Utah. Association for Computational Linguistics.

Raffaele Guarasci, Stefano Silvestri, Giuseppe De Pietro, Hamido Fujita, and Massimo Esposito. 2021. Assessing BERT's ability to learn Italian syntax: A study on null-subject and agreement phenomena. *Journal of Ambient Intelligence and Humanized Computing*.

Raffaele Guarasci, Stefano Silvestri, Giuseppe De Pietro, Hamido Fujita, and Massimo Esposito. 2022. BERT syntactic transfer: A computational experiment on Italian, French and English languages. *Computer Speech & Language*, 71:101261.

Ashim Gupta, Giorgi Kvernadze, and Vivek Srikumar. 2021. BERT & Family Eat Word Salad: Experiments with Text Understanding. *Proceedings of the AAAI Conference on Artificial Intelligence*, 35(14):12946–12954.

Heidi Harley and Elizabeth Ritter. 2002. Person and Number in Pronouns: A Feature-Geometric Analysis. *Language*, 78(3):482–526.

Adi Haviv, Ori Ram, Ofir Press, Peter Izsak, and Omer Levy. 2022. Transformer Language Models without Positional Encodings Still Learn Positional Information.

Jerry R. Hobbs. 1979. Coherence and Coreference*. *Cognitive Science*, 3(1):67–90.

Ryu Iida, Kentaro Torisawa, Jong-Hoon Oh, Canasai Kruengkrai, and Julien Kloetzer. 2016. Intra-Sentential Subject Zero Anaphora Resolution using Multi-Column Convolutional Neural Network. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 1244–1254, Austin, Texas. Association for Computational Linguistics.

Kenji Imamura, Kuniko Saito, and Tomoko Izumi. 2009. Discriminative Approach to Predicate-Argument Structure Analysis with Zero-Anaphora Resolution. In *Proceedings of the ACL-IJCNLP 2009 Conference Short Papers*, pages 85–88, Suntec, Singapore. Association for Computational Linguistics.

Daniel Jurafsky and James H. Martin. 2021. *Speech and Language Processing*, third edition. [Online Draft].

A. Kehler, L. Kertz, H. Rohde, and J. L. Elman. 2007. Coherence and Coreference Revisited. *Journal of Semantics*, 25(1):1–44.

Andrew Kehler and Hannah Rohde. 2013. A probabilistic reconciliation of coherence-driven and centering-driven theories of pronoun interpretation. *Theoretical Linguistics*, 39(1-2):1–37.

Frank Keller. 2010. Cognitively Plausible Models of Human Language Processing. In *Proceedings of the ACL 2010 Conference Short Papers*, pages 60–67, Uppsala, Sweden. Association for Computational Linguistics.

Youngtae Kim, Dongyul Ra, and Soojong Lim. 2021. Zero-anaphora resolution in Korean based on deep language representation model: BERT. *ETRI Journal*, 43(2):299–312.

Fang Kong and Guodong Zhou. 2010. A Tree Kernel-Based Unified Framework for Chinese Zero Anaphora Resolution. In *Proceedings of the 2010 Conference on Empirical Methods in Natural Language Processing*, pages 882–891, Cambridge, MA. Association for Computational Linguistics.

Ryuto Konno, Shun Kiyono, Yuichiroh Matsubayashi, Hiroki Ouchi, and Kentaro Inui. 2021. Pseudo Zero Pronoun Resolution Improves Zero Anaphora Resolution. *arXiv:2104.07425 [cs]*.

Ryuto Konno, Yuichiroh Matsubayashi, Shun Kiyono, Hiroki Ouchi, Ryo Takahashi, and Kentaro Inui. 2020. An Empirical Study of Contextual Data Augmentation for Japanese Zero Anaphora Resolution. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 4956–4968, Barcelona, Spain (Online). International Committee on Computational Linguistics.

Gina R. Kuperberg, Trevor Brothers, and Edward W. Wlotko. 2020. A Tale of Two Positivities and the N400: Distinct Neural Signatures Are Evoked by Confirmed and Violated Predictions at Different Levels of Representation. *Journal of Cognitive Neuroscience*, 32(1):12–35.

Marta Kutas, Katherine A. DeLong, and Nathaniel J. Smith. 2011. A look around at what lies ahead: Prediction and predictability in language processing. In Moshe Bar, editor, *Predictions in the Brain: Using Our Past to Generate a Future*, pages 190–207. Oxford University Press, New York, NY, US.

Alexandra Kuznetsova, Per B. Brockhoff, and Rune H. B. Christensen. 2017. lmerTest Package: Tests in Linear Mixed Effects Models. *Journal of Statistical Software*, 82:1–26.

Hector Levesque, Ernest Davis, and Leora Morgenstern. 2012. The Winograd Schema Challenge. In *Thirteenth International Conference on the Principles of Knowledge Representation and Reasoning*.

Roger Levy. 2008. Expectation-based syntactic comprehension. *Cognition*, 106(3):1126–1177.

Ashley G. Lewis and Marcel Bastiaansen. 2015. A predictive coding framework for rapid neural dynamics during sentence-level language comprehension. *Cortex*, 68:155–168.

Xi Victoria Lin, Todor Mihaylov, Mikel Artetxe, Tianlu Wang, Shuohui Chen, Daniel Simig, Myle Ott, Naman Goyal, Shruti Bhosale, Jingfei Du, Ramakanth Pasunuru, Sam Shleifer, Punit Singh Koura, Vishrav Chaudhary, Brian O'Horo, Jeff Wang, Luke Zettlemoyer, Zornitsa Kozareva, Mona Diab, Veselin Stoyanov, and Xian Li. 2021. Few-shot Learning with Multilingual Language Models. *arXiv:2112.10668 [cs]*.

Steven G. Luke and Kiel Christianson. 2016. Limits on lexical prediction during reading. *Cognitive Psychology*, 88:22–60.

Gary Lupyan and Andy Clark. 2015. Words and the World: Predictive Coding and the Language-Perception-Cognition Interface. *Current Directions in Psychological Science*, 24(4):279–284.

M.H. Maqbool, Luxun Xu, A.B. Siddique, Niloofar Montazeri, Vagelis Hristidis, and Hassan Foroosh. 2022. Zero-label Anaphora Resolution for Off-Script User Queries in Goal-Oriented Dialog Systems. In *2022 IEEE 16th International Conference on Semantic Computing (ICSC)*, pages 217–224.

Danny Merkx and Stefan L. Frank. 2021. Human Sentence Processing: Recurrence or Attention? In *Proceedings of the Workshop on Cognitive Modeling and Computational Linguistics*, pages 12–22, Online. Association for Computational Linguistics.

James A. Michaelov and Benjamin K. Bergen. 2020. How well does surprisal explain N400 amplitude under different experimental conditions? In *Proceedings of the 24th Conference on Computational Natural Language Learning*, pages 652–663, Online. Association for Computational Linguistics.

Irene Fernandez Monsalve, Stefan L. Frank, and Gabriella Vigliocco. 2012. Lexical surprisal as a general predictor of reading time. In *Proceedings of the 13th Conference of the European Chapter of the Association for Computational Linguistics*, pages 398–408. Association for Computational Linguistics.

Mante S. Nieuwland and Jos J. A. Van Berkum. 2006. Individual differences and contextual bias in pronoun resolution: Evidence from ERPs. *Brain Research*, 1118(1):155–167.

Loreto Parisi, Simone Francia, and Paolo Magnani:. 2021. UmBERTo Commoncrawl Cased.

Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Kopf, Edward Yang, Zachary DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. 2019. PyTorch: An Imperative Style, High-Performance Deep Learning Library. In *Advances in Neural Information Processing Systems*, volume 32. Curran Associates, Inc.

Massimo Poesio, Olga Uryupina, and Yannick Versley. 2010. Creating a Coreference Resolution System for Italian. In *Proceedings of the Seventh International Conference on Language Resources and Evaluation (LREC'10)*, Valletta, Malta. European Language Resources Association (ELRA).

R Core Team. 2020. *R: A Language and Environment for Statistical Computing*. R Foundation for Statistical Computing, Vienna, Austria.

RStudio Team. 2020. *RStudio: Integrated Development Environment for r*. RStudio, PBC., Boston, MA.

Keisuke Sakaguchi, Ronan Le Bras, Chandra Bhagavatula, and Yejin Choi. 2019. WinoGrande: An Adversarial Winograd Schema Challenge at Scale. *arXiv:1907.10641 [cs]*.

Stefan Schweter. 2020. Italian BERT and ELECTRA models. Zenodo.

Anna Siewierska. 1999. Reduced pronominals and argument prominence. In *Proceedings of the LFG99 Conference*, page 14, University of Manchester. CSLI Publications.

Anna Siewierska. 2003. Reduced pronominals and argument prominence. In Miriam Butt and Tracy Holloway King, editors, *Nominals: Inside and Out*, Studies in Constraint-Based Lexicalism. CSLI Publications, Stanford, Calif.

Koustuv Sinha, Robin Jia, Dieuwke Hupkes, Joelle Pineau, Adina Williams, and Douwe Kiela. 2021. Masked Language Modeling and the Distributional Hypothesis: Order Word Matters Pre-training for Little. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 2888–2913, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Nathaniel J. Smith and Roger Levy. 2013. The effect of word predictability on reading time is logarithmic. *Cognition*, 128(3):302–319.

Ron Smyth. 1994. Grammatical determinants of ambiguous pronoun resolution. *Journal of Psycholinguistic Research*, 23(3):197–229.

Linfeng Song, Kun Xu, Yue Zhang, Jianshu Chen, and Dong Yu. 2020a. ZPR2: Joint Zero Pronoun Recovery and Resolution using Multi-Task Learning and BERT. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 5429–5434, Online. Association for Computational Linguistics.

Linfeng Song, Kun Xu, Yue Zhang, Jianshu Chen, and Dong Yu. 2020b. ZPR2: Joint Zero Pronoun Recovery and Resolution using Multi-Task Learning and BERT. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 5429–5434, Online. Association for Computational Linguistics.

Rosemary J. Stevenson, Rosalind A. Crawley, and David Kleinman. 1994. Thematic roles, focus and the representation of events. *Language and Cognitive Processes*, 9(4):519–548.

Jakub M. Szewczyk and Kara D. Federmeier. 2022. Context-based facilitation of semantic access follows both logarithmic and linear functions of stimulus probability. *Journal of Memory and Language*, 123:104311.

Hirotoshi Taira, Sanae Fujita, and Masaaki Nagata. 2008. A Japanese Predicate Argument Structure Analysis using Decision Lists. In *Proceedings of the 2008 Conference on Empirical Methods in Natural Language Processing*, pages 523–532, Honolulu, Hawaii. Association for Computational Linguistics.

Nobuhiro Ueda, Daisuke Kawahara, and Sadao Kurohashi. 2020. BERT-based Cohesion Analysis of Japanese Texts. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 1323–1333, Barcelona, Spain (Online). International Committee on Computational Linguistics.

Masato Umakoshi, Yugo Murawaki, and Sadao Kurohashi. 2021. Japanese Zero Anaphora Resolution Can Benefit from Parallel Texts Through Neural Transfer Learning. In *Findings of the Association for Computational Linguistics: EMNLP 2021*, pages 1920–1934, Punta Cana, Dominican Republic. Association for Computational Linguistics.

Guido Van Rossum and Fred L. Drake. 2009. *Python 3 Reference Manual*. CreateSpace, Scotts Valley, CA.

Zirui Wang, Zachary C. Lipton, and Yulia Tsvetkov. 2020. On Negative Interference in Multilingual Models: Findings and A Meta-Learning Treatment. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 4438–4450, Online. Association for Computational Linguistics.

Yotaro Watanabe, Masayuki Asahara, and Yuji Matsumoto. 2010. A Structured Model for Joint Learning of Argument Roles and Predicate Senses. In *Proceedings of the ACL 2010 Conference Short Papers*, pages 98–102, Uppsala, Sweden. Association for Computational Linguistics.

Hadley Wickham, Mara Averick, Jennifer Bryan, Winston Chang, Lucy D'Agostino McGowan, Romain François, Garrett Grolemund, Alex Hayes, Lionel Henry, Jim Hester, Max Kuhn, Thomas Lin Pedersen, Evan Miller, Stephan Milton Bache, Kirill Müller, Jeroen Ooms, David Robinson, Dana Paige Seidel, Vitalie Spinu, Kohske Takahashi, Davis Vaughan, Claus Wilke, Kara Woo, and Hiroaki Yutani. 2019. Welcome to the tidyverse. *Journal of Open Source Software*, 4(43):1686.

Claus O. Wilke. 2020. *Cowplot: Streamlined Plot Theme and Plot Annotations for 'Ggplot2'*.

Terry Winograd. 1972. Understanding natural language. *Cognitive Psychology*, 3(1):1–191.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Remi Louf, Morgan Funtowicz, Joe Davison, Sam Shleifer, Patrick von Platen, Clara Ma, Yacine Jernite, Julien Plu, Canwen Xu, Teven Le Scao, Sylvain Gugger, Mariama Drame, Quentin Lhoest, and Alexander Rush. 2020. Transformers: State-of-the-Art Natural Language Processing. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 38–45, Online. Association for Computational Linguistics.

Jingxuan Yang, Si Li, Sheng Gao, and Jun Guo. 2022. CorefDPR: A Joint Model for Coreference Resolution and Dropped Pronoun Recovery in Chinese Conversations. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 30:571–581.

Jingxuan Yang, Kerui Xu, Jun Xu, Si Li, Sheng Gao, Jun Guo, Ji-Rong Wen, and Nianwen Xue. 2020. Transformer-GCRF: Recovering Chinese Dropped Pronouns with General Conditional Random Fields. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 137–147, Online. Association for Computational Linguistics.

Koichiro Yoshino, Shinsuke Mori, and Tatsuya Kawahara. 2013. Predicate Argument Structure Analysis using Partially Annotated Corpora. In *Proceedings of the Sixth International Joint Conference on Natural Language Processing*, pages 957–961, Nagoya, Japan. Asian Federation of Natural Language Processing.

Shanheng Zhao and Hwee Tou Ng. 2007. Identification and Resolution of Chinese Zero Pronouns: A Machine Learning Approach. In *Proceedings of the 2007 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning (EMNLP-CoNLL)*, pages

541–550, Prague, Czech Republic. Association for Computational Linguistics.

A Full results of statistical analyses

Experiment 1: Subject vs. object antecedent referent

Model	Chisq(df=1)	Corrected *p*
GePeTto	**12.3**	**0.002**
GPT-2 Italian	<0.1	0.993
UmBERTo	3.5	0.109
It BERT Base	0.4	0.614
It BERT XXL	**6.3**	**0.025**
mBERT	1.4	0.376
XLM-100	0.2	0.758
XLM-R Base	**5.3**	**0.041**
XLM-R Large	**19.2**	**<0.001**
XGLM 2.9B	**13.9**	**<0.001**
XGLM 4.5B	**19.1**	**<0.001**
XGLM 7.5B	**32.9**	**<0.001**

Table 2: Results of the likelihood ratio tests in Experiment 1. Models for which there is a significant effect of the manipulation tested are shown in bold.

Experiment 2: Subject vs. object antecedent referent

Model	Chisq(df=1)	Corrected *p*
GePeTto	**8.8**	**0.008**
GPT-2 Italian	0.9	0.482
UmBERTo	0.7	0.514
It BERT Base	0.9	0.482
It BERT XXL	0.6	0.570
mBERT	<0.1	0.956
XLM-100	0.9	0.482
XLM-R Base	1	0.468
XLM-R Large	**7.5**	**0.015**
XGLM 2.9B	**17.5**	**<0.001**
XGLM 4.5B	**23.6**	**<0.001**
XGLM 7.5B	**26.5**	**<0.001**

Table 3: Results of the likelihood ratio tests for all models in Experiment 2.1. Models for which there is a significant effect of the manipulation tested are shown in bold.

Experiment 2: Name vs. pronoun object antecedent referent

Model	Chisq(df=1)	Corrected *p*
GePeTto	**29.8**	**<0.001**
GPT-2 Italian	**17.3**	**<0.001**
UmBERTo	4.9	0.050
It BERT Base	**7.6**	**0.015**
It BERT XXL	**17.5**	**<0.001**
mBERT	4.3	0.068
XLM-100	2.7	0.168
XLM-R Base	**14.7**	**<0.001**
XLM-R Large	**11.7**	**0.002**
XGLM 2.9B	**12.4**	**0.002**
XGLM 4.5B	**16.4**	**<0.001**
XGLM 7.5B	**26.6**	**<0.001**

Table 4: Results of the likelihood ratio tests for all models. Models for which there is a significant effect of the manipulation tested are shown in bold.

Experiment 4: Subject vs. object antecedent referent

Model	F(1,60)	Corrected *p*
GePeTto	**34.6**	**<0.001**
GPT-2 Italian	1.5	0.359
UmBERTo	1.1	0.434
It BERT Base	**7.4**	**0.019**
It BERT XXL	**9.1**	**0.010**
mBERT	0.7	0.529
XLM-100	<0.1	0.815
XLM-R Base	<0.1	0.830
XLM-R Large	0.5	0.575
XGLM 2.9B	**7.1**	**0.022**
XGLM 4.5B	**12**	**0.003**
XGLM 7.5B	**18.4**	**<0.001**

Table 5: Results of the ANOVAs for all models. Models for which there is a significant effect of the manipulation tested are shown in bold.

Experiment 4: Argument x Person Interaction

Model	Chisq(df=1)	Corrected *p*
GePeTto	2.7	0.168
GPT-2 Italian	0.2	0.709
UmBERTo	0.2	0.739
It BERT Base	**5**	**0.048**
It BERT XXL	0.3	0.681
mBERT	0.4	0.607
XLM-100	<0.1	0.993
XLM-R Base	1.1	0.434
XLM-R Large	0.8	0.482
XGLM 2.9B	**8.9**	**0.008**
XGLM 4.5B	**18.5**	**<0.001**
XGLM 7.5B	**7.4**	**0.015**

Table 6: Results of the likelihood ratio tests for all models. Models for which there is a significant effect of the manipulation tested are shown in bold.

Language Acquisition through Intention Reading and Pattern Finding

Jens Nevens[1] and **Jonas Doumen**[2,3] and **Paul Van Eecke**[1,2,3*] and **Katrien Beuls**[4*]
[1]Artificial Intelligence Laboratory, Vrije Universiteit Brussel, Pleinlaan 2, B-1050 Brussels
[2]Itec, imec research group at KU Leuven, E. Sabbelaan 51, B-8500 Kortrijk
[3]KU Leuven, Faculty of Arts, Blijde Inkomststraat 21, B-3000 Leuven
[4]Faculté d'informatique, Université de Namur, rue Grandgagnage 21, B-5000 Namur
{jens,paul}@ai.vub.ac.be
jonas.doumen@kuleuven.be
katrien.beuls@unamur.be

Abstract

One of AI's grand challenges consists in the development of autonomous agents with communication systems offering the robustness, flexibility and adaptivity found in human languages. While the processes through which children acquire language are by now relatively well understood, a faithful computational operationalisation of the underlying mechanisms is still lacking. Two main cognitive processes are involved in child language acquisition. First, children need to reconstruct the intended meaning of observed utterances, a process called intention reading. Then, they can gradually abstract away from concrete utterances in a process called pattern finding and acquire productive schemata that generalise over form and meaning. In this paper, we introduce a mechanistic model of the intention reading process and its integration with pattern finding capacities. Concretely, we present an agent-based simulation in which an agent learns a grammar that enables them to ask and answer questions about a scene. This involves the reconstruction of queries that correspond to observed questions based on the answer and scene alone, and the generalization of linguistic schemata based on these reconstructed question-query pairs. The result is a productive grammar which can be used to map between natural language questions and queries without ever having observed the queries.

1 Introduction

Language is a unique hallmark of human intelligence. Our linguistic systems do not only meticulously serve our communicative needs, they are also incredibly robust to noise, adaptive to change and they can be learnt efficiently. While the processes that drive language acquisition in children are by now relatively well understood, a faithful computational operationalisation of the underlying mechanisms is still lacking. Having such a mechanistic model would, however, constitute a crucial step towards the development of truly intelligent agents in the field of artificial intelligence (Mikolov et al., 2016; Lake et al., 2019).

The idea that children acquire language by actively participating in communicative interactions and making use of general cognitive capacities has been elaborately documented in studies on usage-based language acquisition (Bybee, 2013; Ellis et al., 2016; Ellis and Ogden, 2017). In particular, two highly complementary cognitive processes have been identified to play a key role: *intention reading* and *pattern finding* (Tomasello, 2003, 2009). First, children need to understand the communicative intentions of their interlocutors. In a process called intention reading, they reconstruct the intended meaning of the utterances they observe. Then, they can gradually abstract away from concrete utterances and meaning representations in a process called pattern finding, and acquire productive schemata that generalise over form and meaning. Theoretical as well as empirical evidence has been abundantly provided for both intention reading (Bruner, 1983; Sperber and Wilson, 1986; Meltzoff, 1995; Nelson, 1998) and pattern finding (Goldberg, 1995; Croft, 2000; Diessel, 2004; Goldberg, 2006)

In this paper, we introduce a mechanistic model of the intention reading process and its integration with pattern finding capacities. Concretely, we present an agent-based simulation in which an artificial agent learns a construction grammar that enables it to ask and answer questions about scenes it observes. The learning task thus involves the reconstruction of queries that correspond to observed questions based on the answer and scene alone, as well as the generalization of linguistic schemata based on these reconstructed question-query pairs. The learner gradually acquires a fully productive grammar, consisting of form-meaning mappings, that can be used for both language comprehension, i.e. observing a question and mapping it onto a

*Shared last authors.

Proceedings of the 29th International Conference on Computational Linguistics, pages 15–25
October 12–17, 2022.

query, and language production, i.e. expressing a query in the form of an interrogative linguistic expression.

When it comes to intention reading, the learning challenge amounts to the reconstruction of a query based on a question-answer pair and a scene, without ever observing the query itself. The learner agent is endowed with an inventory of primitive operations, which it can combine to compose new queries. The query composition process allows the agent to hypothesize about the meaning of a question given the scene and the answer to the question. The space of all possible queries that lead to the observed answer in the given scene is typically very large. At the same time, most of these queries are not adequate representations of the meaning of the question and only lead to the correct answer in this specific scene.

The second challenge is to learn abstract schemata. Pairing an observed utterance with its reconstructed meaning yields a form-meaning mapping, called a construction (Fillmore, 1988). Initially, the learner has no way of knowing which parts of the form correspond to which parts of the meaning. Hence, it stores this mapping holistically. Through the observation of different form-meaning mappings, pattern finding mechanisms allow the agent to generalise over reoccurring form-meaning patterns, thereby capturing the compositional structure of the language.

Intention reading and pattern finding are highly complementary. Specifically, intention reading facilitates pattern finding by providing meaning hypotheses. In turn, pattern finding constrains the search process involved in intention reading by providing partial analyses. This interplay between intention reading and pattern finding is key in successfully tackling the learning challenge, and constitutes the main contribution of this paper.

We validate our methodology using the CLEVR benchmark dataset (Johnson et al., 2017a), in which the communicative task of the agents consists in asking and answering questions about scenes of geometric figures. Over many interactions, the learner incrementally acquires a fully operational grammar that can be used for both language comprehension and production. We show that the acquired grammar effectively solves the visual question answering task.

The contributions of this work are both theoretical and practical. On the theoretical side, the presented work provides computational evidence for the cognitive plausibility of usage-based theories of language acquisition, in particular concerning intention reading and pattern finding. On the practical side, this paper introduces a powerful new methodology that allows autonomous agents to acquire an effective communication system with human-like properties through task-oriented interactions in their native environment.

The remainder of this paper is structured as follows. Section 2 introduces the dataset. In Section 3, we introduce the technical foundations of our methodology. Section 4 describes the experimental setup for learning construction grammars through communicative interactions. Section 5 presents the experimental results. Related work is discussed in Section 6. Finally, Section 7 reflects on the results and contributions of our work. The code of this experiment is made available through the open-source Babel toolkit[1] (Steels and Loetzsch, 2010; Nevens et al., 2019b).

2 Data

The CLEVR dataset (Johnson et al., 2017a) consists of (i) rendered scenes with geometric figures of various shapes, sizes, colours and materials, (ii) English questions about these scenes, and (iii) answers to these questions. The questions test a variety of reasoning skills, including attribute identification (*"What size is the yellow cube?"*), counting (*"How many large cylinders are there?"*), existence (*"Is there a red ball?"*), comparison (*"Are there more spheres than cylinders?"*), spatial relationships (*"What shape is the thing right of the purple cube?"*) and logical operations (*"How many things are either spheres or cylinders?"*).

This dataset was chosen because it satisfies two criteria. First, it offers visually grounded linguistic expressions. The objects about which the agents communicate are actual referents in the agents' environment. Second, it contains a large number of scenes and plenty of similar, yet non-identical questions. Such examples are necessary for any kind of generalisation process and are consistent with theoretical and empirical evidence of how children learn language (Tomasello, 2003; Tamminen et al., 2015). Other datasets that fit these two criteria could also be used.

For the experiment in this paper, we have selected a subset of the CLEVR questions. Specif-

[1] https://emergent-languages.org

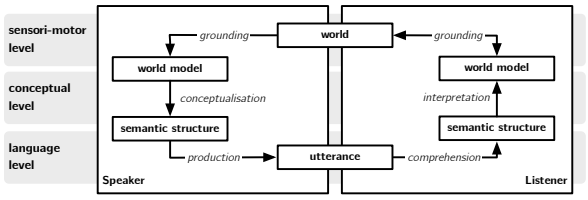

Figure 1: Processes involved in a language game.

ically, questions concerning comparison, spatial relations and logical operations have been left out. The main reason for this is that these are more complex cognitive operations that correspond to longer and more complex questions. Such questions are far removed from the type that children are faced with. Starting from CLEVR's validation split, our final dataset consists of 10,044 unique questions. Each question can be used in any of the 15,000 scenes of the validation split.

3 Technical Foundations

Our methodology builds on three main technical foundations: (i) the language game paradigm (Section 3.1), (ii) procedural cognitive semantics (Section 3.2) and (iii) computational construction grammar (Section 3.3).

3.1 The Language Game Paradigm

The *language game* paradigm (Steels, 1995, 2001) studies how linguistic conventions arise in a population of agents through local interactions and coordination. Every interaction, or language game, takes place between two agents, called the *speaker* and the *listener*, and models a particular communicative task, e.g. drawing the attention to an object in the environment. The semiotic cycle (Steels, 2012) in Figure 1 provides a schematic overview of the processes involved for the speaker and the listener. These processes take place across three different levels: the sensorimotor level, the conceptual level and the language level. In the following sections, we elaborate on the technical foundations of the processes taking place on the conceptual level (Section 3.2) and on the language level (Section 3.3). In Section 4, we concretely describe how the various processes in the semiotic cycle have been implemented in terms of these technical foundations in order to operationalise the mechanistic model of intention reading and its integration with pattern finding.

3.2 Procedural Cognitive Semantics

Incremental Recruitment Language (IRL) (Van den Broeck, 2008; Spranger et al., 2012) operationalises key insights from procedural cognitive semantics (Woods, 1968; Winograd, 1972; Johnson-Laird, 1977). Specifically, it treats the meaning of natural language utterances as programs that can be executed algorithmically in terms of the agents' representation of the environment, i.e. its world model. Such programs capture the logical structure underlying utterances in the form of *semantic networks*. An example semantic network is shown in Figure 2. The symbols preceded by question marks, as in ?OBJECT-1, are logic variables. Semantic networks are made up of predicates that are declaratively combined by sharing variables. The predicates in semantic networks represent either *semantic entities* or *primitive operations*.

Semantic entities are concepts known by the agent. They are introduced in semantic networks through BIND statements, as in (BIND SHAPE ?SHAPE-1 CUBE), binding the concept CUBE of type SHAPE to the variable ?SHAPE-1. In this experiment, a repertoire of semantic entities is given a priori to the agents. This repertoire includes the various colours, shapes, sizes and materials present in the CLEVR dataset. However, these concepts can also be learned through communicative interactions, e.g. as in Nevens et al. (2020).

Primitive operations represent the basic cognitive capabilities of the agent. In this experiment, six operations are made available. These are based on annotations provided with the CLEVR dataset. Primitive operations are implemented as multi-directional predicates with typed arguments that operate over the agents' world model and semantic repertoire. From the argument(s) that are bound, e.g. via a BIND statement or via the output of other predicates, a predicate can compute new bindings

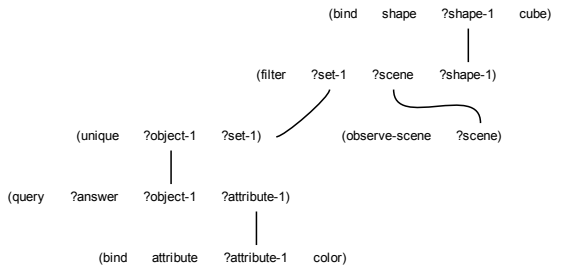

Figure 2: Semantic network for the question *"What color is the cube?"*.

for the unbound argument(s).

IRL provides the computational architecture for (i) automatically composing predicates into programs and (ii) evaluating programs in terms of data structures that represent the agents' environment. The composition of predicate networks is a combinatorial search process where predicates are added incrementally and linked together by unifying their variable arguments until a communicative goal is reached. Type information of the arguments is used to determine which arguments of predicates can be linked. The evaluation of semantic networks consists in finding values, i.e. are concrete referents in the environment or the agent's semantic repertoire, for all variables in the network. A variable-value pairing is called a binding. Every semantic network has exactly one *target variable* of which the binding holds the communicative goal or intention of the corresponding utterance. To illustrate, the evaluation of the semantic network in Figure 2 goes as follows. The predicate OBSERVE-SCENE retrieves the set of all objects in the scene and binds this to the variable ?SCENE. The FILTER predicate accesses this set via the shared variable ?SCENE, together with the shape CUBE via ?SHAPE-1. The predicate then filters this set such that only cubes remain and the result is bound to ?SET-1. Next, the UNIQUE predicate checks if ?SET-1 contains a single object. If so, that object is bound to ?OBJECT-1. Finally, the QUERY predicate retrieves the color of ?OBJECT-1 and binds the result to ?ANSWER. The binding of this variable is indeed the answer to the question. Other primitive operations that are available are EXIST, which checks whether the cardinality of a set is greater than zero, and COUNT, which computes the cardinality of a set.

3.3 Computational Construction Grammar

The agents' language comprehension and production capabilities are operationalised using Fluid Construction Grammar[2] (FCG – Steels, 2017; van Trijp et al., 2022). FCG is a computational operationalisation of the basic tenets of construction grammar (Fillmore, 1988; Goldberg, 1995; Kay and Fillmore, 1999; Croft, 2001) and supports bi-directional construction-based language processing.

Corresponding to different stages of child language acquisition (Tomasello, 2003), we consider three types of constructions in this experiment:

Holophrase constructions constitute a holistic mapping between an entire form and its entire meaning representation. For example, a mapping between the question *"What color is the cube?"* and the semantic network shown in Figure 2 would be a holophrase construction.

Item-based constructions are generalisations over holophrase constructions that capture their similarities and differences, both with respect to form and meaning. For example, a construction associating the form '*What is the color of the ?X?*' with its meaning of querying the color of the referent of ?X would be an item-based generalisation over '*What is the color of the cube?*' with its meaning of querying the color of the cube and '*What is the color of the sphere?*' with its meaning of querying the color of the sphere.

Lexical constructions provide arguments that can fill slots in item-based constructions. For example, the form *"cube"* associated with its meaning of filtering for the prototype of the concept CUBE can fill the ?X slot in the item-based construction covering '*What is the color of the ?X?*'.

Holophrase constructions only allow to comprehend the exact same utterance or produce the exact same meaning as the observation it was learnt from. Item-based constructions, on the other hand, cover a wider range of utterances and meanings through their slots, but require lexical constructions for filling those slots. When item-based and lexical constructions combine, the lexical arguments are inserted into the item-based slots, resulting in a complete utterance and a complete semantic network. The possible combinations of slots and arguments emerge through language use (Pine and Lieven, 1997; Croft, 2001). In FCG, these combinations are captured as links in a dynamically updated network, called the categorial network (Van Eecke, 2018; Steels et al., 2022). This network is consulted during constructional language processing. Hence, the links in this network determine which item-based and lexical constructions can combine.

4 The Elicitation Game

We set up a language game in a tutor-learner scenario, which we call the *elicitation game*. The agents are situated in scenes from the CLEVR dataset. The tutor is an agent that has an established linguistic inventory which allows to comprehend and produce all questions from the CLEVR

[2] https://www.fcg-net.org

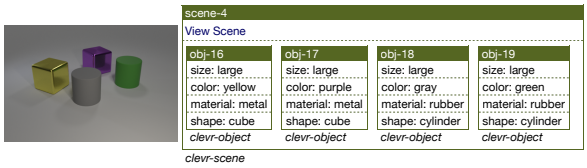

Figure 3: The agents are situated in scenes from the CLEVR dataset (left). These are represented symbolically (right).

dataset (cf. the grammar presented in Nevens et al. (2019a)). The learner starts with an empty construction inventory, but is endowed with the six primitive operations described above, a repertoire of semantic entities, and a number of learning mechanisms. Both the tutor and the learner can take on the discourse roles of speaker and listener. The communicative task of the elicitation game is the following. The speaker has a concept in mind and wants to elicit that concept from the listener. Therefore, it has to come up with a question about the objects in the scene to which the listener should provide the answer. The game succeeds if the listener's answer refers to the concept the speaker had in mind. Following the semiotic cycle (Figure 1), the interactions consist of the following steps:

1. Both agents perceive a randomly selected scene. Scenes are represented symbolically, as shown in Figure 3.

2. Each agent is randomly assigned a discourse role: speaker or listener.

3. The speaker selects a concept from the CLEVR dataset. This can be an object attribute (colour, size, material or shape), a number between 0 and 10, 'yes' or 'no'.

4. The speaker tries to come up with a question that has the chosen concept as its answer. This process involves two steps, namely *conceptualising* a semantic network and *producing* a question that expresses this meaning representation.

5. The listener observes the question produced by the speaker.

6. The listener tries to answer the question. This process also involves two steps, namely *comprehending* the question, i.e. mapping it onto a meaning representation, and *interpreting* this meaning representation in the current scene to come up with the answer.

7. The speaker checks whether the listener replies with the answer it had in mind. This determines the outcome of the game: success or failure.

8. If the game fails, the tutor provides feedback to the learner. Specifically, it reveals the correct answer to the question that was asked. This is a learning opportunity for the learner agent.

In the following sections, the processes of conceptualisation, production, comprehension and interpretation are discussed in detail. Afterwards, the learning mechanisms operationalising intention reading and pattern finding are introduced.

4.1 Conceptualisation

Conceptualisation is performed by the speaker to come up with the query it wants to ask. The speaker uses its own inventory of primitive operations to compose a semantic network such that the evaluation of that network, i.e. the answer to the constructed query, results in the concept the speaker wants to elicit.

4.2 Production

In production, the speaker uses its own inventory of form-meaning mappings, or constructions, to map the semantic network composed in the previous step to a natural language utterance, in this case a question. The tutor can express all valid semantic networks. When acting as the speaker, the learner will try to use its acquired holophrase, item-based and lexical constructions to express the semantic network. However, the learner's construction inventory may be inadequate for performing this mapping, causing the interaction to fail.

4.3 Comprehension

Comprehension is the inverse process of production. The listener uses its own construction inventory to try and map the observed utterance, in this case a question, to its underlying meaning representation. When acting as the listener, the learner's construction inventory may be inadequate for performing this mapping, causing the interaction to proceed with a blank answer.

4.4 Interpretation

Interpretation is performed by the listener to compute its hypothesis about the answer to the question.

This is done by evaluating the semantic network that results from comprehension. The listener's hypothesis is the value of the target variable of that semantic network.

4.5 Learning Mechanisms

Learning takes place when the interaction has failed, i.e. when the learner acting as the listener cannot retrieve the meaning underlying an observed question or the applied form-meaning mappings result in an incorrect hypothesis for the answer. The outcome of the learning mechanisms is to make new form-meaning mapping(s), through intention reading and pattern finding, in order to be more successful in future interactions.

Intention reading is performed by the learner to reconstruct a hypothesis of the meaning underlying the observed question. Similar to conceptualisation, this is done by composing a semantic network. The goal of the composition process is to construct a semantic network leading to the tutor's intention, i.e. the answer that was revealed at the end of the interaction. Crucially, the number of possible semantic networks that lead to the provided answer in the current scene is typically very large, and most of those networks will not be adequate representations of the meaning of the question. The problem faced by intention reading is thus twofold. First, the agent needs to overcome the enormous search space of possible semantic networks. Second, the agent needs to overcome incorrect meaning hypotheses.

Pattern finding allows the learner to generalise over reoccurring patterns on both the form side, which can be observed, and the meaning side, which is reconstructed through intention reading. The goal is not to learn holophrase constructions for every observation, but to learn more general item-based and lexical constructions that cover multiple observations, including novel ones. Given that both the form side and the meaning side of constructions are represented as sets of predicates, set difference operations that use unification to compare the elements are used to find the overlapping and non-overlapping parts.

The learner is endowed with five learning mechanisms that operationalise intention reading and pattern finding. These mechanisms are active in the inverse order of their presentation below.

Learning holophrases At the start of the experiment, the learner's construction inventory is empty. When it observes novel utterances, the only thing it can do is to create holophrase constructions. Specifically, the meaning is hypothesised through intention reading and paired with the observed utterance. Holophrase constructions form the basis of the learning process. Other learning operators will generalise over them.

Generalising over holophrases Whenever possible, pattern finding will compare newly created mappings between observed utterances and their reconstructed meanings against previously acquired holophrase constructions. When a minimal difference is found on both the form side and the meaning side, a generalisation can be learned. On the form side, a minimal difference refers to a single token, while on the meaning side, this is a single predicate. An item-based construction will capture the overlapping parts of the form and meaning, while a lexical construction captures the non-overlapping parts. A link is added to the categorial network indicating that the arguments of the lexical construction are suitable for filling the item-based slots on the form side and the meaning side. Three cases of this learning mechanism can be identified: (i) the new form-meaning pairing differs from the holophrase construction by substituting a minimal difference, (ii) the new form-meaning pairing extends the holophrase construction by a minimal difference, and (iii) the new form-meaning pairing reduces the holophrase construction by a minimal difference.

Learning from partial meanings This learning mechanism creates new constructions that can combine with existing constructions to analyse the observed utterance. Concretely, the acquisition of item-based and lexical constructions can lead to the partial comprehension of novel utterances. The resulting partial meaning is used by intention reading to hypothesise about the meaning underlying the observed question. Crucially, the partial meaning drastically reduces the search space faced by intention reading, as large parts of the search space that do not contain this partial meaning can be pruned. This is how the interplay between intention reading and pattern finding allows to overcome the intractability of the intention reading process. Three cases of this learning mechanism exist. First, partial meaning provided by one or more lexical constructions results in an item-based construction with an equal number of slots. This case is illus-

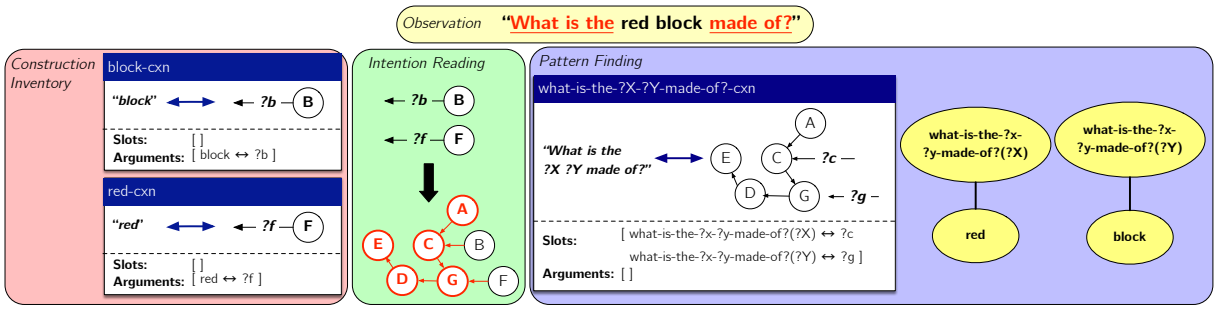

Figure 4: Schematic representation of the learning mechanism that completes a partial meaning provided by lexical constructions. The RED-CXN and BLOCK-CXN provide a partial meaning (red box) for the observed utterance (yellow box). Intention reading creates a meaning hypothesis, reusing the partial meaning (green box). Pattern finding creates an item-based construction with two slots and corresponding links in the categorial network (blue box).

trated in Figure 4. Second, partial meaning provided by an item-based construction leads to a single lexical construction. If multiple lexical items are missing, there would be referential uncertainty, which is not explored here. Third, partial meaning provided by a combination of one or more lexical constructions and an item-based construction also leads to maximally one lexical construction. In all three cases, slot-and-argument links are also added to the categorial network.

Learning slot-argument links This learning mechanism handles cases where previously acquired item-based and lexical constructions cover the observed utterance, but where the slot-argument combination has not been observed before. Due to the absence of that link in the categorial network, the corresponding constructions cannot combine, causing comprehension or production to fail. In comprehension, the learner simply adds the slot-argument combination it observed. In production, the learner creatively tries out slot-argument combinations. However, these links are only consolidated when the interaction turns out to be successful. Note that intention reading is not required for this learning mechanism.

Lateral inhibition Lateral inhibition facilitates the self-organisation of the learner's construction inventory (Steels, 1995). It is used at the end of every interaction, including successful ones. Concretely, it models the entrenchment of constructions (Schmid, 2007) by updating their scores. New constructions obtain a default score of 0.5. Scores are bound between 0 and 1.

The scores of constructions are updated based on the outcome of the game. If the game fails, the scores of the constructions used during the game are decreased by 0.4. These constructions were inadequate for the communicative task and should therefore be used less often in the future. If the game succeeds, the scores are increased by 0.1, while scores of competing constructions are decreased by 0.1. Competing constructions are constructions that also could have contributed to the comprehension or production process. When reaching a score of 0, constructions are removed from the construction inventory. The exact values used to alter construction scores do not influence the global dynamics of the learning process, as long as these values are positive and negative respectively.

The presented learning mechanisms do not posit a built-in bias towards more abstract constructions. However, given that more abstract constructions are inherently applicable in a wider range of situations, they will therefore be used more frequently. As a result of lateral inhibition, this will result in higher entrenchment scores for more abstract constructions and in lower scores for less abstract constructions. By updating scores of constructions through lateral inhibition and by preferring constructions with a higher score during comprehension and production, a positive feedback loop is created between the success and use of constructions. This feedback loop ensures that only constructions that can be used successfully in the communicative task remain, while unsuccessful constructions gradually disappear. This way, incorrect meaning hypotheses generated by intention reading can be overcome.

5 Experimental Results

This section presents the validation of our methodology on the CLEVR data. The presented results

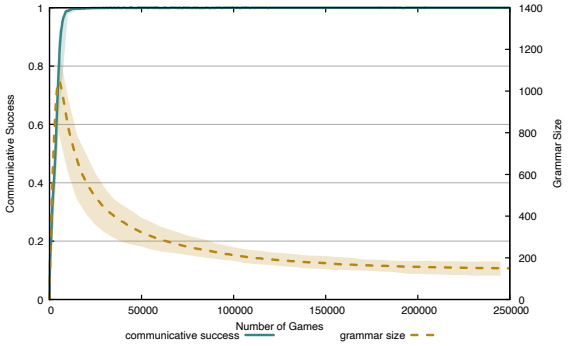

Figure 5: Evolution of communicative success (left axis) and grammar size (right axis) over time.

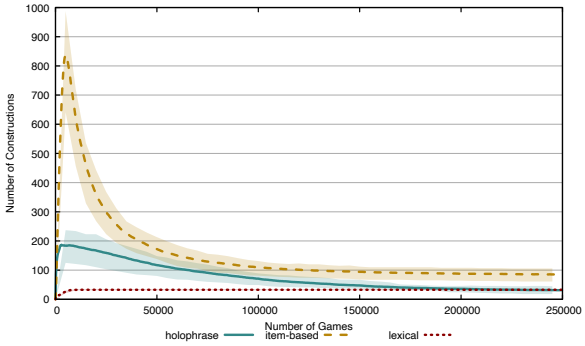

Figure 6: Evolution of the number of constructions over time, split per type.

are based on ten independent runs of 250,000 interactions each. The filled areas around the lines on the plots represent the 5th and 95th percentile.

Figure 5 presents the main results of the experiment. It shows the *communicative success* and the *grammar size* over time. Both metrics start at 0. The communicative success rises quickly, with more than 78% of the interactions being successful after 5,000 games. This is half the number of possible utterances from our subset of data. The grammar size also reaches its peak at this point, with on average 1,048 constructions being learned. Success reaches over 98% after 10,000 interactions and does not go below 99.9% from interaction 25,000 onwards. It is only after 200,000 interactions that the success reaches a stable 100%. This is because it takes a long time for all incorrect constructions to be cleared from the construction inventory. Specifically, the learner needs to observe just the right question in just the right scene to find out that a previously acquired form-meaning mapping is incorrect. The grammar size gradually decreases, reaching 492 constructions after 25,000 interactions and 149 constructions by the end of the experiment.

Figure 6 breaks down the grammar size per construction type. At the start of the experiment, only holophrase constructions are learned. Soon after, the learner can start to generalise over them. The construction inventory peaks after 5,000 interactions, reaching 184 holophrase constructions and 838 item-based constructions. Afterwards, more abstract item-based constructions gradually become dominant and overtake their less abstract competitors, including the holophrase constructions. By the end of the experiment, 31 holophrase constructions and 85 item-based constructions remain. There is less competition among the lexical constructions. After 10,000 interactions, 33 lexical constructions are learned and this remains stable until the end. The theoretical maximum of 35 lexical constructions was reached in four out of ten experimental runs. After 10,000 interactions, 33 lexical constructions are learned and this remains stable until the end. We note that it is *not* the goal of the experiment to reach one particular set of constructions. Hence, the absolute number of constructions at a given time is not important. Instead, the goal is to become successful at the communicative task and learn an efficient construction inventory for doing so.

6 Related Work

Prior agent-based models have also studied the constructivist co-acquisition of syntax and semantics through task-oriented interactions (Gerasymova and Spranger, 2010; Beuls et al., 2010; Spranger and Steels, 2015). While these models have provided important insights, this paper advances the state of the art in two ways. First, the presented experiment operates on a much larger scale. In contrast to prior models, this work does not focus on a specific linguistic phenomenon, such as the Russian aspectual system (Gerasymova and Spranger, 2010), the Hungarian agreement system (Beuls et al., 2010) or English spatial language (Spranger and Steels, 2015). The utterances being considered in this work are far more complex, both in terms of morpho-syntax and semantics. Second, the presented experiment provides fewer scaffolds. The agent does not receive a segmentation of input utterances, a predefined lexicon as in Beuls et al. (2010) or a taxonomy guiding the generalisation process of constructions as in Spranger and Steels (2015). The agent only relies on a number of basic cognitive operations and a collection of concepts. The latter can also be learned from sub-symbolic

observations through the language game methodology, as shown by Nevens et al. (2020).

When it comes to the field of visual question answering, existing approaches typically tackle the task in one of two ways. Either, a large end-to-end neural network maps an image and a question to the answer, e.g. as in Malinowski et al. (2015). Alternatively, RNNs are used to map the question onto a query which is then executed on the image, e.g. as in Johnson et al. (2017b). Both of these approaches rely on huge amounts of training data. The second approach additionally requires questions annotated with queries to train the RNN. Further, both approaches rely on black-box architectures, making it unclear how and why an answer was generated. Our methodology overcomes these shortcomings. Similar to the first approach, the agent is only presented with images, questions and their answers and autonomously reconstructs the underlying queries. Our methodology is more data-efficient and the agents' representations and reasoning processes are fully transparent. Finally, the agents' communication system is open-ended and completely bidirectional using the same representations and processing mechanisms, which is not possible using current neural network architectures.

7 Concluding Discussion

The contributions of this paper span two areas.

Usage-based Language Acquisition The experiment presented in this paper provides computational evidence for the cognitive plausibility of theories from usage-based language acquisition, in particular intention reading and pattern finding (Tomasello, 2003). We have operationalised these capacities and their interplay in an agent-based simulation, which has indeed revealed learning dynamics that are similar to those observed in the literature. Starting from holophrases, the agent learns to generalise over them, gradually leading to more and more abstract schemata.

Autonomous Agents Most importantly, this paper pushes forward the state of the art in the development of autonomous agents with communication systems offering human-like properties. In particular, we have introduced a novel methodology that allows an agent to acquire an inventory of constructions that facilitates bi-directional language processing and is suitable for solving a communicative task. This grammar is acquired through situated, task-oriented interactions with indirect supervision only. Given only utterances, feedback on their underlying intentions and a collection of primitive cognitive operations, the agent engages in a highly non-trivial process of meaning creation, operationalised through intention reading, and combines this with a process of schema abstraction, operationalised through pattern finding. We show that the search space involved in intention reading, i.e. the composition of semantic programs, can effectively be constrained through its integration with pattern finding and that together, these processes allow the agent to bootstrap a successful communication system. The presented methodology is completely transparent, both in terms of the applied learning operators and the resulting inventory of constructions. The agent learns incrementally, acquiring productive linguistic structures even after a single interaction. The agent's grammar is open-ended and the lateral inhibition dynamics enable the agent to remain ever-adaptive, e.g. when the environment or the communicative task changes.

Acknowledgements

The research reported in this paper was funded by the Research Foundation Flanders (FWO) through grants 1SB6219N (JN) and 75929 (PVE), imec's Smart Education research programme, with support from the Flemish government (JD), and the European Union's Horizon 2020 research and innovation programme under grant agreement no. 951846 (KB).

References

Katrien Beuls, Kateryna Gerasymova, and Remi van Trijp. 2010. Situated learning through the use of language games. In *Proceedings of the 19th Annual Machine Learning Conference of Belgium and The Netherlands (BeNeLearn)*, pages 1–6.

Jerome Bruner. 1983. *Learning to use language*. Oxford University Press, Oxford.

Joan L. Bybee. 2013. Usage-based theory and exemplar representations of constructions. In Thomas Hoffmann and Graeme Trousdale, editors, *The Oxford Handbook of Construction Grammar*, pages 49—-69. Oxford University Press.

William Croft. 2000. *Explaining language change: An evolutionary approach*. Pearson Education, Harlow.

William Croft. 2001. *Radical construction grammar: Syntactic theory in typological perspective*. Oxford University Press, Oxford.

Holger Diessel. 2004. *The acquisition of complex sentences*. Cambridge University Press, Cambridge.

Nick C. Ellis and Dave C. Ogden. 2017. Thinking about multiword constructions: Usage-based approaches to acquisition and processing. *Topics in Cognitive Science*, 9(3):604–620.

Nick C. Ellis, Ute Römer, and Matthew Brook O'Donnell. 2016. *Usage-Based Approaches to Language Acquisition and Processing: Cognitive and Corpus Investigations of Construction Grammar*. Wiley-Blackwell, Malden.

Charles J Fillmore. 1988. The mechanisms of "construction grammar". In *Annual Meeting of the Berkeley Linguistics Society*, volume 14, pages 35–55.

Kateryna Gerasymova and Michael Spranger. 2010. Acquisition of grammar in autonomous artificial systems. In *Proceedings of the 19th European Conference on Artificial Intelligence (ECAI-2010)*, pages 923–928.

Adele Goldberg. 1995. *Constructions: A construction grammar approach to argument structure*. University of Chicago Press, Chicago.

Adele Goldberg. 2006. *Constructions at work: The nature of generalization in language*. Oxford University Press, Oxford.

Justin Johnson, Bharath Hariharan, Laurens van der Maaten, Li Fei-Fei, C. Lawrence Zitnick, and Ross Girshick. 2017a. CLEVR: A diagnostic dataset for compositional language and elementary visual reasoning. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 2901–2910.

Justin Johnson, Bharath Hariharan, Laurens van der Maaten, Judy Hoffman, Li Fei-Fei, C Lawrence Zitnick, and Ross Girshick. 2017b. Inferring and executing programs for visual reasoning. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 2989–2998.

Philip N Johnson-Laird. 1977. Procedural semantics. *Cognition*, 5(3):189–214.

Paul Kay and Charles Fillmore. 1999. Grammatical constructions and linguistic generalizations: The what's x doing y? construction. *Language*, 75(1):1–33.

Brenden M Lake, Tal Linzen, and Marco Baroni. 2019. Human few-shot learning of compositional instructions. In *Proceedings of the 41st Annual Conference of the Cognitive Science Society*, pages 611–617.

Mateusz Malinowski, Marcus Rohrbach, and Mario Fritz. 2015. Ask your neurons: A neural-based approach to answering questions about images. In *Proceedings of the 2015 IEEE International Conference on Computer Vision*, pages 1–9. IEEE.

Andrew Meltzoff. 1995. Understanding the intentions of others: re-enactment of intended acts by 18-month-old children. *Developmental Psychology*, 31(5):838–850.

Tomas Mikolov, Armand Joulin, and Marco Baroni. 2016. A roadmap towards machine intelligence. In *Proceedings of the International Conference on Intelligent Text Processing and Computational Linguistics*, pages 29–61.

Katherine Nelson. 1998. *Language in cognitive development: The emergence of the mediated mind*. Cambridge University Press, Cambridge.

Jens Nevens, Paul Van Eecke, and Katrien Beuls. 2019a. Computational construction grammar for visual question answering. *Linguistics Vanguard*, 5(1):20180070.

Jens Nevens, Paul Van Eecke, and Katrien Beuls. 2019b. A practical guide to studying emergent communication through grounded language games. In *AISB Language Learning for Artificial Agents Symposium*, pages 1–8.

Jens Nevens, Paul Van Eecke, and Katrien Beuls. 2020. From continuous observations to symbolic concepts: A discrimination-based strategy for grounded concept learning. *Frontiers in Robotics and AI*, 7:84.

Julian M Pine and Elena VM Lieven. 1997. Slot and frame patterns and the development of the determiner category. *Applied Psycholinguistics*, 18(2):123–138.

Hans-Jörg Schmid. 2007. Entrenchment, salience, and basic levels. In Dirk Geeraerts and Hubert Cuyckens, editors, *The Oxford handbook of cognitive linguistics*, pages 117–138. Oxford University Press, New York.

Dan Sperber and Deirdre Wilson. 1986. *Relevance: Communication and cognition*. Harvard University Press, Cambridge, MA.

Michael Spranger, Simon Pauw, Martin Loetzsch, and Luc Steels. 2012. Open-ended procedural semantics. In Luc Steels and Manfred Hild, editors, *Language Grounding in Robots*, pages 153–172. Springer, New York.

Michael Spranger and Luc Steels. 2015. Co-acquisition of syntax and semantics: an investigation in spatial language. In *Proceedings of the Twenty-Fourth International Joint Conference on Artificial Intelligence*, pages 1909–1915, Palo Alto, Ca. AAAI Press.

Luc Steels. 1995. A self-organizing spatial vocabulary. *Artificial Life*, 2(3):319–332.

Luc Steels. 2001. Language games for autonomous robots. *IEEE intelligent systems*, 16:16–22.

Luc Steels. 2012. Grounding language through evolutionary language games. In *Language Grounding in Robots*, pages 1–22. Springer, New York.

Luc Steels. 2017. Basics of Fluid Construction Grammar. *Constructions and Frames*, 9(2):178–225.

Luc Steels and Martin Loetzsch. 2010. Babel: A tool for running experiments on the evolution of language. In *Evolution of Communication and Language in Embodied Agents*, pages 307–313. Springer, Berlin.

Luc Steels, Paul Van Eecke, and Katrien Beuls. 2022. Usage-based learning of grammatical categories. *arXiv preprint arXiv:2204.10201*.

Jakke Tamminen, Matthew H. Davis, and Kathleen Rastle. 2015. From specific examples to general knowledge in language learning. *Cognitive Psychology*, 79:1–39.

Michael Tomasello. 2003. *Constructing a Language: A Usage-Based Theory of Language Acquisition*. Harvard University Press, Harvard.

Michael Tomasello. 2009. The usage-based theory of language acquisition. In *The Cambridge handbook of child language*, pages 69–87. Cambridge University Press, Cambridge.

Wouter Van den Broeck. 2008. Constraint based compositional semantics. In *The Evolution of Language: Proceedings of the 7th International Conference (EVOLANG7)*, pages 338–345. World Scientific.

Paul Van Eecke. 2018. *Generalisation and specialisation operators for computational construction grammar and their application in evolutionary linguistics Research*. Ph.D. thesis, Vrije Universiteit Brussel, Brussels: VUB Press.

Remi van Trijp, Katrien Beuls, and Paul Van Eecke. 2022. The FCG editor: An innovative environment for engineering computational construction grammars. *PLOS ONE*, 17(6):1–27.

Terry Winograd. 1972. Understanding natural language. *Cognitive Psychology*, 3(1):1–191.

W. A. Woods. 1968. Procedural semantics for a question-answering machine. In *Proceedings of the December 9-11, 1968, Fall Joint Computer Conference, Part I*, pages 457–471, New York, NY, USA.

Stability of Syntactic Dialect Classification Over Space and Time

Jonathan Dunn[1] and **Sidney Wong**[2]
Department of Linguistics
University of Canterbury
[1]New Zealand Institute for Language, Brain and Behaviour
[2]Geospatial Research Institute Toi Hangarau
jonathan.dunn@canterbury.ac.nz
sidney.wong@pg.canterbury.ac.nz

Abstract

This paper analyses the degree to which dialect classifiers based on syntactic representations remain stable over space and time. While previous work has shown that the combination of grammar induction and geospatial text classification produces robust dialect models, we do not know what influence both changing grammars and changing populations have on dialect models. This paper constructs a test set for 12 dialects of English that spans three years at monthly intervals with a fixed spatial distribution across 1,120 cities. Syntactic representations are formulated within the usage-based Construction Grammar paradigm (CxG). The decay rate of classification performance for each dialect over time allows us to identify regions undergoing syntactic change. And the distribution of classification accuracy within dialect regions allows us to identify the degree to which the grammar of a dialect is internally heterogeneous. The main contribution of this paper is to show that a rigorous evaluation of dialect classification models can be used to find both variation over space and change over time.

1 Geographic Variation Over Time

This paper experiments with the stability of dialect classification models over space and time in order to determine the degree to which they capture language variation and change. The assumption in previous work has been that a geo-referenced corpus (Davies and Fuchs, 2015; Cook and Brinton, 2017; Dunn, 2020) captures the linguistic behaviour of specific populations. This paper experiments with the spatial and temporal stability of dialect models by systematically constructing monthly test sets spanning a three-year period. This allows us to evaluate the continuing effectiveness of dialect models over time, an important criteria for determining their validity. Because different locations represent different populations, we use spatial sampling to construct test sets which represent different local populations within each country. This allows us to determine the degree to which a dialect like New Zealand English adequately represents the varied populations within New Zealand.

Dialect classification is the task of predicting the location of origin for the individual who produced a given sample (Dunn, 2019b; Chakravarthi et al., 2021; Gaman et al., 2020). Thus, dialect classification, by focusing on the latent properties of geo-referenced samples, differs from geolocation (Rahimi et al., 2017) which focuses on predicting the location of the sample itself and from geo-characterization (Adams and McKenzie, 2018) which focuses on predicting attributes of the location. While all three tasks rely on geographic information, dialect classification is unique in modelling variations in the linguistic system. Beyond this, dialect classification is part of ensuring that NLP represents the world's population, including non-standard and non-western populations.

The temporal evaluation (Section 6) shows that most dialects share the same performance decay rate. This indicates a general effect of model decay rather than cases of change over time within individual dialects. The spatial evaluation, however, shows that prediction accuracy for all dialects is spatially-conditioned within countries (Section 7). This indicates that, while dialect models capture proto-typical populations within each country, they do not equally describe all local populations.

The experiments in this paper use construction grammar (CxG: Goldberg 2006; Langacker 2008; Croft 2013) to represent syntactic structure for the purpose of observing dialectal variation. CxG is a *usage-based* approach to syntax, a bottom-up theory of language in which frequent exposure is hypothesized to lead to the emergence of grammatical units (Hopper, 1987; Bybee, 2006). The use of syntactic representations for dialect classification ensures that the model does not rely on extraneous information like place names or local topics of in-

Proceedings of the 29th International Conference on Computational Linguistics, pages 26–36
October 12–17, 2022.

terest. From this perspective, a GRAMMAR is a set of constructions that together represent the structure of a language. A DIALECT MODEL is a matrix of spatial weights in which the number of rows corresponds to the number of constructions in the grammar and the number of columns corresponds to the number of dialects. These weights, learned using a Linear SVM, support dialect classification and also represent spatial variation in the grammar.

In order to undertake a spatio-temporal evaluation, we collect a balanced corpus of tweets to represent 12 varieties of English around the world. The basic experimental paradigm is to train models on a fixed period (July through December 2018) and then test those models at monthly intervals from 2019 to 2021. Each monthly test set maintains the same geographic distribution as the training data, so that fluctuations in performance are not caused by changes in the locations represented.

After considering related work on dialect classification and other geographic models (Section 2), we consider the corpora used in these experiments (Section 3). We then present the syntactic representations used (Section 4) and the basic experimental methods (Section 5). The performance of dialect models over time is presented in Section 6 and the performance over space in Section 7. The main contribution of this paper is to show that the performance of dialect classification models remains stable over time but that there is significant spatial variation in performance within dialect areas.

2 Related Work

Early work showed that part-of-speech trigrams are able to distinguish between some regional dialects (Sanders, 2007), a method that continues to appear in recent work (Kreutz and Daelemans, 2018). Similar methods have been used for authorship analysis (Hirst and Feiguina, 2007) and for characterizing immigrant populations (Nerbonne and Wiersma, 2006). In other contexts, non-syntactic features can out-perform syntactic features for modelling dialects (Kroon et al., 2018), so that many approaches to distinguishing between dialects are similar to language identification models (Ali, 2018).

More recent work has modelled geographic syntactic variation by combining grammar induction with geospatial text classification (Dunn, 2018a, 2019b,c). The use of grammar induction to learn a syntactic feature space mitigates the fact that most grammars represent standard varieties (Jørgensen et al., 2015), thus poorly representing many dialects around the world. In this paradigm, the learned grammar provides a feature space (c.f., Section 4) and the frequency of grammatical constructions in each sample is used to model dialects: a bag-of-constructions approach to text classification.

Most work on geographic variation is focused on lexical variation (Eisenstein et al., 2010) and change (Eisenstein et al., 2014). Recent work has shown a close correspondence between lexical variation in tweets and lexical variation in a dialect survey (Grieve et al., 2019). This work is important for showing that digital usage mirrors face-to-face usage. Other work has shown that geographic variation can be taken into account during language identification to ensure the inclusion of non-standard varieties (Jurgens et al., 2017). Models of lexical variation have generally failed to account for polysemy, so that competition between senses is not captured (Zenner et al., 2012), but more recent work has been able to account for polysemy in this context (Lucy and Bamman, 2021).

A related line of work uses language data to model non-linguistic properties of populations and places. For example, the problem of geo-location is to predict the location of a user given properties of a document (Wing and Baldridge, 2014; Alex et al., 2016; Rahimi et al., 2017). This task differs from dialect classification in that named entities and topic features can provide significant information. A related task is to model the characteristics of a particular place rather than the population of that place (Adams, 2015; Adams and McKenzie, 2018; Hovy and Purschke, 2018; Villegas et al., 2020). While there is a close connection between a place and its population, this line of work remains focused on characterizing non-linguistic attributes.

This paper makes two main contributions: *First*, it experiments with geographic syntactic variation over time and within dialect regions, significantly expanding our understanding of geographic variation in syntax. *Second*, from a more practical perspective, this paper evaluates the degree to which geographic models remain robust over space and time, an evaluation not previously available.

3 Geographic Language Data

This paper draws on social media data from the *Corpus of Global Language Use* (CGLU), using geo-referenced tweets that are identified for language using the idNet package (Dunn, 2020). The

Circle	Region	Country	N. Cities	N. Words
Inner-Circle	Oceania	Australia	98	3.9 mil
Inner-Circle	Oceania	New Zealand	99	2.0 mil
Inner-Circle	North American	Canada	95	4.9 mil
Inner-Circle	North American	United States	86	4.5 mil
Inner-Circle	European	Ireland	100	3.6 mil
Inner-Circle	European	United Kingdom	89	5.5 mil
Total Inner-Circle	**3**	**6**	**567**	**24.4 mil**
Outer-Circle	African	Ghana	69	1.1 mil
Outer-Circle	African	Kenya	98	1.8 mil
Outer-Circle	South Asian	India	96	2.5 mil
Outer-Circle	South Asian	Pakistan	100	1.0 mil
Outer-Circle	Southeast Asian	Malaysia	99	0.8 mil
Outer-Circle	Southeast Asian	Philippines	91	1.1 mil
Total Outer-Circle	**3**	**6**	**553**	**8.37 mil**

Table 1: Inventory of Regions, Countries, and Cities for Data Collection (One Month)

collection method for social media in the CGLU involves geographic searches from co-ordinates of individual cities. Here we sample from 1,120 cities representing 12 countries and six regions, as shown in Table 1. This table shows the amount of data by place by month. The total data set contains six months for training and 36 months for testing. The data set as a whole is visualized at earthLings.io.

This corpus is designed to provide a balanced representation of different varieties of English over time. The colonial history of English has led to a distinction within the World Englishes paradigm (Kachru, 1990) between *inner-circle* varieties that represent the first diaspora (e.g., Canada) and *outer-circle* varieties that represent the second diaspora (e.g., India). We include six dialects/varieties each from the inner-circle and outer-circle groups.

Within each group we include three regions, each with two country-level varieties. As shown in Table 1, the inner-circle group contains three regions: Oceania (Australia and New Zealand), North America (Canada and the US), and Europe (the UK and Ireland). The collection of data from these countries is distributed across 567 cities, where each city represents a 50km radius from the city center. For each month, we sample 24.4 million words representing these inner-circle varieties.

The outer-circle group also contains three regions: Africa (Ghana and Kenya), South Asia (India and Pakistan), and Southeast Asia (Malaysia and the Philippines). The collection of data from these countries is distributed across 553 cities, with a comparable sample of 8.37 million words for each month across the training and testing periods.

To maintain a comparable geographic distribution over time, we maintain the same number of samples from each city. This means, for example, that the relative influence of Brisbane and Perth in Australia remain constant over time. A *sample* for the purposes of this paper is an aggregation of individual tweets from the same place and time until the sample reaches 500 words. These larger samples provide more syntactic information for each dialect than do individual tweets. While previous work has used samples of 1,000 words (Dunn, 2019b), here we use smaller samples in order to increase the capacity for error analysis. As with many tasks, there is a trade-off between the higher accuracy provided by larger sample sizes and the flexibility provided by smaller sample sizes.

The distribution of samples across cities is taken from the training period (2018). Thus, the density of data by location across time is fixed to represent the density during the training period. This allows us to control for changes in the collection: for example, if Wellington began to produce more data in 2021, this change in distribution within New Zealand would appear to be syntactic variation while actually reflecting a change in the means of observation. Data collection spans from 07-2018 until 12-2021, a period of 42 months. The training period is 2018 and the testing period is 2019 through 2021. The geographic distribution across countries, as shown in Table 1, is held constant across this period, controlling for other sources of variation that might impact dialect models.

4 Syntactic Representations

This section details the main ideas of construction grammar (CxG), including both (i) the grammar induction algorithm used to learn syntactic representations here and (ii) examples of constructions used in the dialect models. The basic approach here is, first, to use grammar induction to learn a grammar and, second, to use the frequency of the constructions in that grammar to undertake geospatial text classification (Dunn, 2019b,c).

CxG can be distinguished from other approaches to syntax given its three core ideas: First, CxG posits a continuum between the lexicon and the grammar rather than a strict separation (for example, into a vocabulary and a set of phrase structure rules). This CONSTRUCTICON contains both lexical items and traditional syntactic structures. For example, a grammar-and-lexicon approach would analyze (a) below as an intransitive sentence by labelling the verb *laugh* as intransitive. The problem is that verb valency is quite fluid, as shown in (b) and (c). The CxG analysis of this fluidity is that (a) represents an INTRANSITIVE construction into which *laugh* is merged and (b)/(c) represent a CAUSED-MOTION construction into which *laugh* is merged. Thus, the fluidity of the argument structure here is explained by an underlying construction, itself meaningful, which interacts with specific lexical items. (Note that the grammar used for modelling dialects does not contain any individual lexical items as constructions).

(a) Peter *laughed*.
(b) The audience *laughed* Peter off the stage.
(c) His marriage *laughed* Peter into rehab.
(d) Peter *laughed* all the way to the bank.

A second main idea in CxG is that syntactic structure varies in its level of abstractness, with some representations being quite item-specific. The constructicon is an inheritance hierarchy in which fully-productive constructions like the CAUSED-MOTION construction in (b)/(c) have item-specific children like the idiom in (d). Essentially, (d) is a non-compositional and idiomatic version of the construction in (b)/(c) with some of the slots constrained to require a fixed phrase.

(e) [SYN:NP – SYN:VP]
(f) [SYN:NP – SYN:VP – SEM:*object* – SEM:*loc*]
(g) [SYN:NP – SYN:VP – LEX:*all the way to the bank*]

A third main idea in CxG is that constructions are constraint-based representations in which slot-fillers are drawn from lexical, syntactic, and semantic categories. Each unit in a construction is a *slot*, separated by dashes in (e)/(f)/(g) above. Each slot is defined using a *slot-constraint*. For example, the INTRANSITIVE construction in (e) can be represented using only syntactic constraints. In contrast, the CAUSED-MOTION construction in (f) has two semantic constraints; these are labelled for purposes of exposition as *object* and *location*. The construction in (g) is item-specific and idiomatic, so that it can only be described using lexical constraints. The point, then, is that different levels of abstraction are captured in CxG using different types of slot-constraints.

This paper draws on previous approaches to the unsupervised learning of constructions (Dunn, 2017, 2018b). The first challenge is to build the inventory of lexical, syntactic, and semantic constraints that constructions are built on. Here we use the most frequent 100k words across the entire corpus of tweets as the lexicon. The syntactic constraints are drawn from the Universal Part-of-Speech tagset (Petrov et al., 2012) as implemented by the Ripple-Down-Rules tagger (Nguyen et al., 2016). The semantic constraints are drawn from fastText embeddings (Grave et al., 2019) clustered into discrete semantic domains using k-means. A complete inventory of these semantic domains is provided in the supplementary material; this approach ignores polysemy in lexical items when defining semantic constraints, using a single representation for each word-form.

From the perspective of varying levels of abstractness, syntactic constraints are the most general because they are divided into the smallest inventory of labels (only 14). Lexical constraints are the least general, with a lexicon of 100k words. And semantic constraints are in the middle, with an inventory of 1,000 domains. This parameter choice (i.e., using 1,000 semantic domains) results from the desired granularity in domains, falling between the very general syntactic constraints and the very specific lexical constraints. Thus, constructions are a sequence of slots, each of which is defined by a slot-constraint. Each type of slot-constraint (lexical, semantic, and syntactic) differ in their level of abstractness. For instance, lexically-defined constructions are more idiomatic and item-specific than syntactically-defined constructions.

This work relies on a loss function based on Minimum Description Length (Goldsmith, 2006; Grünwald and Rissanen, 2007) and a construction parser with a beam-search strategy (Dunn, 2019a) that operates on top of a psychologically-plausible association measure, the ΔP (Ellis, 2007). The contribution of this paper is to analyze syntactic variation across space over time using previous work on computational CxG; thus, we do not provide a fuller description of the framework here. Previous work has shown that these grammars converge onto stable representations as the amount of training data is increased (Dunn and Tayyar Madabushi, 2021), that grammars of individuals are significantly different than grammars of groups of individuals (Dunn and Nini, 2021), and that transformer-based language models can be fine-tuned using constructional information (Tayyar Madabushi et al., 2020).

The grammar used in these experiments is learned from the training period (2018) but includes a wider pool of 18 English-speaking countries in order to provide a global grammar of English. This larger training corpus for grammar induction contains 478 million words. The fastText embeddings are trained on this same extended corpus, but covering the entire period in order to increase the amount of data available for training; this larger corpus contains 4.2 billion words. This results in a single grammar that contains 6,119 individual constructions, some of which are shown in (h) through (n) below. Dialect models are learned by parsing each sample using this grammar, counting the frequency of each construction in each sample, and using the resulting feature space for dialect classification. The complete grammar, along with examples from the training data for each construction, is available in the supplementary material.

The following examples illustrate the nature of constructions; both constructions like (h) and examples like (h1) are drawn from the grammar used in the experiments. Each slot in (h) is separated by dashes and each slot-constraint is defined using lexical (LEX), syntactic (SYN), or semantic (SEM) categories. Lexical constraints are words given in italics; syntactic constraints are drawn from part-of-speech tags; and semantic constraints are formulated using numbers that refer to clustered embeddings, such as <443> in (k). For dialect classification, each construction (h) provides a feature and the frequency of that construction (h1 through h3) provides a sample-specific quantification.

(h) [LEX:*it* – SYN:AUX – SYN:V]
(h1) ‘it is set’
(h2) ‘it was shut’
(h3) ‘it can go’

The first example, in (h), shows a simple clause with an expletive *it* as subject and a variable auxiliary verb. The example in (i) is a lexically-constrained noun phrase with *ability* as the head of an infinitival verb. A further lexically-constrained noun phrase in (j) shows the importance of a tweet-specific grammar: *ur* replaces the more traditional *your* as the pronoun.

(i) [LEX:*ability* – LEX:*to* – SYN:V]
(i1) ‘ability to focus’
(i2) ‘ability to live’
(i3) ‘ability to wait’

(j) [LEX:*ur* – SYN:ADJ – SYN:N]
(j1) ‘ur new journey’
(j2) ‘ur own money’
(j3) ‘ur mad tunes’

The adposition phrase in (k) contains a semantic constraint on the complement noun, in this case a type of location. As an example of how constructions themselves can be meaningful, (l) shows a copula construction with an ending conjunction. But the construction as a whole marks a caveat on the evaluation that is expressed by the copula.

(k) [SYN:ADP – SYN:N – <443>]
(k1) ‘along airport road’
(k2) ‘in union station’
(k3) ‘into police station’

(l) [SYN:N – LEX:*was* – SYN:ADJ – SYN:CC]
(l1) ‘bike was awesome but’
(l2) ‘birthday was great and’
(l3) ‘movie was better but’

The more complicated verb phrase in (m) contains a main verb, *myself* as a direct object, and an infinitival verb. This implicitly constrains the main verb to verbs of thinking like *compare* and *tell*, showing that implicit semantic constraints arise from interactions between slots. Finally, the complex noun phrase in (n) reflects a specific template of NP + ADP. In this way, constructions capture grammatical units of varying size and abstractness.

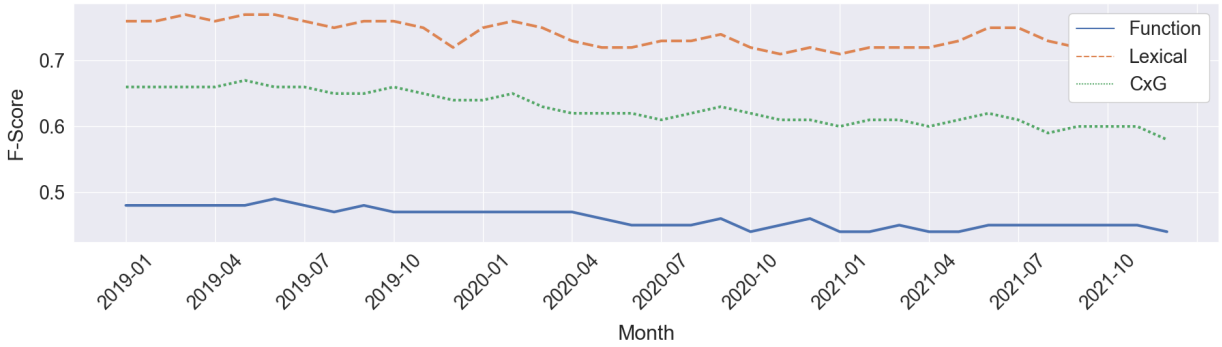

Figure 1: F-Score Against Baselines Over Time, All Varieties

(m) [SYN:V – LEX:*myself* – LEX:*to* – SYN:V]
(m1) 'allowing myself to hope'
(m2) 'forcing myself to sleep'
(m3) 'tell myself to stop'

(n) [LEX:*the*–SYN:N – LEX:*of* – SYN:DET – SYN:N]
(n1) 'the happiness of another person'
(n2) 'the owner of the station'
(n3) 'the masters of the game'

This section has presented CxG as a paradigm for usage-based syntax and reviewed previous work on computational CxG. An unsupervised construction grammar is learned from the training period, providing an adaptable feature space that contains structures from many different dialects. As the discussed examples show, these learned constructions provide a rich syntactic feature space for modelling geographic variation in production over time.

5 Dialect Models

The task of dialect classification or identification is to predict the location of origin for the author of a sample given some set of linguistic features. The classification here predicts country-level dialects like New Zealand English or Australian English. From the perspective of linguistics, dialect classification allows us to study variation in a high-dimensional space: variation across an entire grammar (Dunn, 2019b) rather than variation in individual and independent features (Grieve et al., 2019). From the perspective of NLP, dialect classification is part of the general problem of ensuring that language technology represents the world's population rather than privileged sub-sets of the world's population (Dunn and Adams, 2020).

Because part of the goal is to model spatio-temporal variation in the grammar, a dialect model takes the form of a matrix in which each feature (a construction in the grammar) is a row and each dialect (a country-level label) is a column. This matrix represents the degree to which a given part of the grammar is subject to geographic variation. Taken row-wise, this matrix provides a measure of whether a particular construction varies across space. And, taken column-wise, this matrix provides a description of each dialect that, for example, can be compared with every other dialect to determine which are the most similar. As discussed below, dialect models are implemented as Linear SVMs that are trained using a bag-of-constructions approach in which the parser counts how many times each construction occurs in each sample.

Using the data from 2018 for training, we compare three models: *First*, a syntactic model based on the frequencies of the constructional features described above. *Second*, a baseline model that uses the frequency of function words like *of* or *was*, a common baseline for problems in authorship analysis (Grieve, 2007; Stamatatos, 2009; Argamon, 2018) when content words need to be avoided. *Third*, for the purpose of comparison, we include a unigram lexical model with TF-IDF weighting and function words removed so that it contains no syntactic information. Each of these models are implemented as a Linear SVM. Within this task, SVMs remain competitive, as shown by recent shared tasks on Romanian dialect identification (Gaman et al., 2020) and on identifying similar Uralic languages (Chakravarthi et al., 2021). In each case, we use a development set to determine parameters.

In each case, we train three models: INNER contains only inner-circle varieties like American English; OUTER contains only outer-circle varieties like Indian English; and ALL contains all 12 varieties. These are trained on the data from 2018 and tested on data from 2019, 2020, and 2021. The reason for maintaining separate models in some

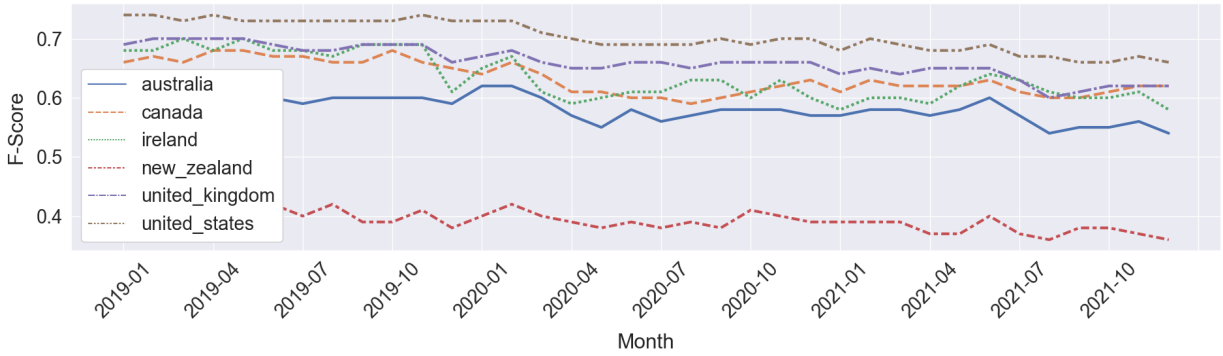

Figure 2: F-Score by Country Over Time for CxG Model, Inner-Circle Varieties

conditions is that inner-circle varieties have significantly more training and testing data available, which could lead to higher performance as an artifact. Thus, for example, the inner-circle condition contains only training and testing data from the six countries listed as *inner-circle* in Table 1.

As an initial analysis, the f-scores of each of these three models over time is shown in Figure 1, with the y-axis indicating the weighted f-score and the x-axis indicating time. All three classifiers are well above the majority baseline. The lowest performing is the function word model, a weak approximation for syntactic variation. The highest performing is the lexical model. This hierarchy remains stable across the three year testing period.

AU	CA	IE	NZ
australia	canada	ireland	nz
australian	canadian	irish	zealand
mate	ontario	dublin	auckland
melbourne	trump	cork	jacinda
sydney	toronto	limerick	te
abc	vancouver	galway	kiwi
brisbane	trudeau	lads	liked
labor	km	hurling	lincoln
nsw	kpa	county	hamilton
turnbull	alberta	final	kph

Table 2: Top Lexical Features By Country

Given the results in Figure 1, could we use the lexical model to examine dialects? The issue, as in previous work, is that the information contained in this model does not represent linguistic variation. Table 2 shows the top lexical items for four inner-circle countries: Australia, Canada, Ireland, and New Zealand. Most of these terms are place-names (like *australia*), place-specific named-entities (like *abc*), or people associated with these countries (like *jacinda*). Only a few terms would qualify as dialectal variants, for example *mates* vs *lads*. As a representation of latent linguistic variation, the lexical model is not relevant; we thus focus on the syntactic models in the remaining analysis.

6 Syntactic Variation Over Time

We begin the analysis by looking at the weighted average f-score by model for the beginning of the test period (2019-01) and the end (2021-12), as shown in Table 3. This represents the impact of time on the overall accuracy. First, we see that outer-circle models have better performance. The most likely reason for this is that outer-circle varieties are more distinct from one another, in part because these varieties exist in more linguistically-diverse settings. For example, the US is less linguistically diverse than India in digital settings (Dunn et al., 2020). Although outer-circle varieties have a higher average f-score, they also have a greater change in f-score. This indicates more variability over time.

	Function	Grammar
Inner-Only, 2019-01	0.44	0.66
Inner-Only, 2021-12	0.40	0.59
Inner-Only Decline	*0.04*	*0.07*
Outer-Only, 2019-01	0.75	0.83
Outer-Only, 2021-12	0.66	0.75
Outer-Only Decline	*0.09*	*0.08*
All Dialects, 2019-01	0.48	0.66
All Dialects, 2021-12	0.44	0.58
All Dialects Decline	*0.04*	*0.08*

Table 3: Change in Performance Over Time by Model

Second, we notice in Table 3 that the relative performance of function words and the CxG model remain similar across the testing period. The full grammar model always out-performs the function

word baseline. We use a regression analysis to model the decay rate for each dialect in the CxG models, examining the amount of change in precision and recall over time (c.f., Figure 2). The basic idea here is that a consistent decay rate indicates model error while a faster rate of decay for individual dialects indicates change in those dialects themselves. Among inner-circle varieties, only NZ has a significant difference from the others, for recall but not for precision. A decline in precision would mean that samples from other dialects have become more similar to NZ; this does not happen. The observed decline in recall means that samples from NZ have become more similar to other dialects. This indicates that there has been a significant change in NZ but not in other dialects. No outer-circle varieties have a different decay rate, so that only NZ shows this type of change.

	AU	CA	IE	NZ	UK	US
AU	0	0	0	0	0	0
CA	0	0	0	0	0	0
IE	0	0	0	0	0	-.04
NZ	.16	0	0	0	.28	0
UK	0	0	0	0	0	0
US	0	0	0	0	0	0
	GH	**IN**	**KE**	**MY**	**PK**	**PH**
GH	0	-1.01	0	0	-.28	-.40
IN	0	0	0	-.91	0	0
KE	0	0	0	0	0	0
MY	0	0	0	0	0	0
PK	-.20	0	0	0	0	.13
PH	0	0	0	0	0	0

Table 4: Changing Relationships Between Dialects Using a VECM Analysis of False Positive Errors

The decay rate represents the overall trend for a given dialect but it does not take into account the specific errors made. The confusion matrix for each dialect provides a monthly representation of the distribution of false positive errors. For example, in the CxG model that includes all dialects, Canadian English has 1,488 false positives as American English in the first test period, but only 48 with India and 7 with Pakistan. This distribution of false positive errors over time provides a more detailed view of the classifier's performance. Because the classification model itself does not change after training, changes in the distribution of errors reflect changes that have arisen in a given dialect after training.

The question here is whether the relationship between dialects (geographic variation) changes over time. We model this using a Vector Error Correction Model (VECM: Lütkepohl and Krätzig 2004). This model checks for relationships between multiple time series, which in this case reflect changing error patterns between dialects. The data represents a non-stationary time series because the number of errors in all dialects increases over time (i.e., there is a decline in performance as shown in Table 3). To partially control for the increase in errors over time, we examine the relative frequency of false positives by country by month. The VECM model allows us to determine if there is a significant long-term trend in the distribution of errors from a given dialect, robust to short-term variations.

We examine the significant changes by country for the inner-circle and outer-circle models with CxG features in Table 4. Only significant changes are shown; negative values indicate that samples for the row have become more frequently mistaken for the column. Thus, for the inner-circle varieties, NZ becomes more similar over time to Australia and the UK. This means that, in addition to lower classification performance, NZ is also subject to the most change in the way it is situated among other dialects. Outer-circle varieties on the whole are subject to more change in error distribution over time than inner-circle varieties. The analysis of decay rates also shows that NZ was subject to change over time; the difference is that this analysis takes into account the distribution of errors rather than viewing the error rate as a black box. The outer-circle varieties have a changing error distribution, but not a changing error rate.

7 Syntactic Variation Within Countries

While previous work has viewed a dialect area as a homogeneous entity, here we have sampled from approximately 100 points for each country and maintained a consistent sample over time. To what degree is the performance of dialect classifiers driven by geographic trends within a country? If a country like Australia has a single dominant grammar, then the performance of the syntax-based classifier should be relatively consistent within that country. To test this hypothesis, we look at the average accuracy over time for samples collected from each point within a country.

This is shown in Table 5 with a global Moran's *I* used as a measure of spatial autocorrelation within a country (Anselin, 1988). A common method in

	Moran's *I*	Mean Acc.	Min	Max
AU	0.30	61%	18%	83%
CA	0.54	65%	07%	100%
IE	0.17	58%	35%	89%
NZ	0.20	36%	08%	62%
UK	0.22	73%	41%	82%
US	0.18	79%	53%	97%
	Moran's *I*	**Mean Acc.**	**Min**	**Max**
GH	0.30	86%	42%	94%
IN	0.38	84%	27%	95%
KE	0.24	89%	62%	97%
MY	0.70	79%	50%	95%
PK	0.42	70%	15%	87%
PH	0.20	77%	37%	88%

Table 5: Geographic Variation in Performance

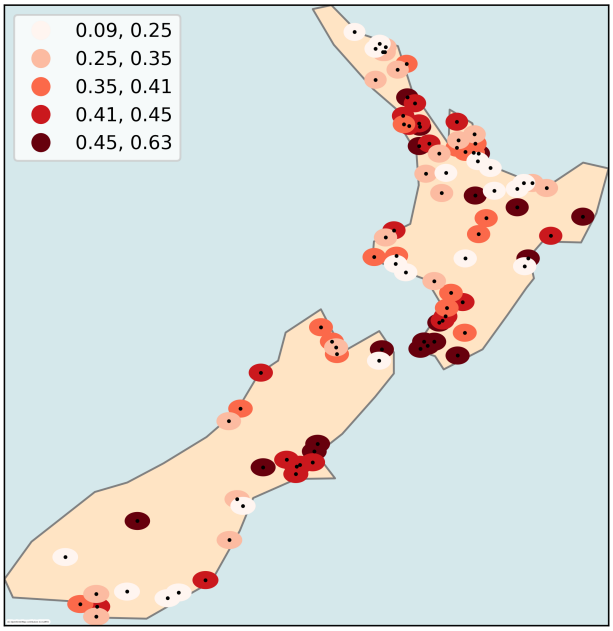

Figure 3: Map of Average City-Level Accuracy, NZ

geospatial statistics, Moran's *I* measures the correlation in a single variable (here, prediction accuracy for dialect classification) across different locations. This measure has values closer to 1 when the variable is highly spatially organized and closer to 0 when there is no spatial organization. Given that there are different numbers of samples from each location, it is possible that a generic Moran's *I* would view sparse locations as outliers; thus, we use the Empirical Bayes rate adjustment to control for the level of precision in each location as well (Xia and Carlin, 1998; Anselin et al., 2006).

The table also shows the mean accuracy across cities and the min and max accuracy. These results show that there is an effect for location: the dialect models work well in some places and not so well in others. The Moran's *I* determines whether this variation in performance is spatially structured. Because different locations represent different populations, these are measures of how well the dialect models work for the entire population of a country. Full maps and spatial results are available in the supplementary material.

All countries have a significant spatial pattern to their accuracy distribution. Within inner-circle countries, Canada has the highest deviation, with a wide range in accuracy and a significant spatial structure to that variation. The US and UK have the highest accuracy, while NZ performs much worse than other dialects, perhaps because of the change over time discussed above. To explore this further, we visualize the internal variation for NZ, the inner-circle dialect with the lowest performance and the most change over time, in Figure 3. Each collection point is a dot and the shading in the surrounding radius represents the accuracy for that collection area. Darker colors represent higher accuracy. The main cities (Auckland, Wellington, Christchurch) have the most consistent performance. But areas with known distinct linguistic landscapes like Northland (far north) and Southland (far south) have much lower accuracy. More rural areas around the country have consistently lower accuracy as well. The main point in this spatial error analysis is that, because different locations represent different populations, the observed variations in accuracy show that these dialect models do not equally represent all populations within the country.

8 Conclusions

This paper has shown that syntax-based dialect classifiers can reveal both spatial and temporal patterns in linguistic variation. We find that the models remain robust over time, with a fixed decay rate, with the exception of change observed in NZ. This means that, while classification performance does decline, the rate of decline is predictable and evenly distributed. Within dialect regions, however, there is a significant spatial effect on performance. This evaluation is important for establishing an understanding of how dialect models and other geographic models function in the face of on-going linguistic change and population change over space and time. Here, even the best dialect models do not equally represent all speakers of a dialect.

References

B. Adams. 2015. Finding Similar Places using the Observation-to-Generalization Place Model. *Journal of Geographical Systems*, 17(2):137–156.

B. Adams and G. McKenzie. 2018. Crowdsourcing the character of a place: Character-level convolutional networks for multilingual geographic text classification. *Transactions in GIS*, 22(2):394–408.

B. Alex, C. Llewellyn, C. Grover, J. Oberlander, and R. Tobin. 2016. Homing in on Twitter users: Evaluating an enhanced geoparser for user profile locations. *Proceedings of the International Conference on Language Resources and Evaluation*, pages 3936–3944.

M. Ali. 2018. Character Level Convolutional Neural Network for Arabic Dialect Identification. In *Proceedings of the Workshop on NLP for Similar Languages, Varieties and Dialects*, pages 122–127.

L. Anselin. 1988. *Spatial Econometrics: Methods and Models*. Kluwer Academic Publishers, Dordrecht.

L. Anselin, N. Lozano-Gracia, and J. Koschinky. 2006. Rate transformations and smoothing. Technical report, Spatial Analysis Laboratory, Department of Geography, University of Illinois, Urbana, IL.

S. Argamon. 2018. Computational Forensic Authorship Analysis: Promises and Pitfalls. *Language and Law*, 5(2):7–37.

J. Bybee. 2006. From Usage to Grammar: The mind's response to repetition. *Language*, 82(4):711–733.

B. Chakravarthi, G. Mihaela, R. Ionescu, H. Jauhiainen, T. Jauhiainen, K. Lindén, N. Ljubešić, N. Partanen, R. Priyadharshini, C. Purschke, E. Rajagopal, Y. Scherrer, and M. Zampieri. 2021. Findings of the VarDial evaluation campaign 2021. In *Proceedings of the Workshop on NLP for Similar Languages, Varieties and Dialects*, pages 1–11.

P. Cook and J. Brinton. 2017. Building and Evaluating Web Corpora Representing National Varieties of English. *Language Resources and Evaluation*, 51(3):643–662.

W. Croft. 2013. Radical Construction Grammar. In *The Oxford Handbook of Construction Grammar*, pages 211–232. Oxford University Press.

M. Davies and R. Fuchs. 2015. Expanding horizons in the study of World Englishes with the 1.9 billion word Global Web-based English Corpus (GloWbE). *English World-Wide*, 36(1):1–28.

J. Dunn. 2017. Computational Learning of Construction Grammars. *Language & Cognition*, 9(2):254–292.

J. Dunn. 2018a. Finding Variants for Construction-Based Dialectometry: A Corpus-Based Approach to Regional CxGs. *Cognitive Linguistics*, 29(2):275–311.

J. Dunn. 2018b. Modeling the Complexity and Descriptive Adequacy of Construction Grammars. *In Proceedings of the Society for Computation in Linguistics*, pages 81–90.

J. Dunn. 2019a. Frequency vs. Association for Constraint Selection in Usage-Based Construction Grammar. In *Proceedings of the Workshop on Cognitive Modeling and Computational Linguistics*, pages 117–128.

J. Dunn. 2019b. Global Syntactic Variation in Seven Languages: Toward a Computational Dialectology. *Frontiers in Artificial Intelligence*, 2:15.

J. Dunn. 2019c. Modeling Global Syntactic Variation in English Using Dialect Classification. In *Proceedings of the Workshop on NLP for Similar Languages, Varieties and Dialects*, pages 42–53.

J. Dunn. 2020. Mapping languages: the Corpus of Global Language Use. *Language Resources and Evaluation*, 54:999–1018.

J. Dunn and B. Adams. 2020. Geographically-Balanced Gigaword Corpora for 50 Language Varieties. In *Proceedings of the International Language Resources and Evaluation Conference*, pages 2528–2536.

J. Dunn, T. Coupe, and B. Adams. 2020. Measuring Linguistic Diversity During COVID-19. In *Proceedings of the Workshop on Natural Language Processing and Computational Social Science*, pages 1–10.

J. Dunn and A. Nini. 2021. Production vs Perception: The Role of Individuality in Usage-Based Grammar Induction. In *Proceedings of the Workshop on Cognitive Modeling and Computational Linguistics*, pages 149–159.

J. Dunn and H Tayyar Madabushi. 2021. Learned Construction Grammars Converge Across Registers Given Increased Exposure. In *Proceedings of the Conference on Computational Natural Language Learning*, pages 268–278.

J. Eisenstein, B. O'Connor, N. Smith, and E. Xing. 2010. A latent variable model for geographic lexical variation. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, pages 221–227, 287.

J. Eisenstein, B. O'Connor, N. Smith, and E. Xing. 2014. Diffusion of lexical change in social media. *PloSOne*, 10:1371.

N. Ellis. 2007. Language Acquisition as Rational Contingency Learning. *Applied Linguistics*, 27(1):1–24.

M. Gaman, D. Hovy, R. Ionescu, H. Jauhiainen, T. Jauhiainen, K. Lindén, N. Ljubešić, N. Partanen, C. Purschke, Y. Scherrer, and M. Zampieri. 2020. A Report on the VarDial Evaluation Campaign 2020. In *Proceedings of the Workshop on NLP for Similar Languages, Varieties and Dialects*, pages 1–14.

A. Goldberg. 2006. *Constructions at work: The nature of generalization in language*. Oxford University Press, Oxford.

J. Goldsmith. 2006. An Algorithm for the Unsupervised Learning of Morphology. *Natural Language Engineering*, 12(4):353–371.

E. Grave, P. Bojanowski, P. Gupta, A. Joulin, and T. Mikolov. 2019. Learning word vectors for 157 languages. In *Proceedings of the International Conference on Language Resources and Evaluation*, pages 3483–3487.

J. Grieve. 2007. Quantitative authorship attribution: An evaluation of techniques. *Literary and Linguistic Computing*, 22(3):251–270.

J. Grieve, C. Montgomery, A. Nini, A. Murakami, and D. Guo. 2019. Mapping Lexical Dialect Variation in British English Using Twitter. *Frontiers in Artificial Intelligence*, 2:11.

P. Grünwald and J. Rissanen. 2007. *The Minimum Description Length Principle*. MIT Press.

G. Hirst and O. Feiguina. 2007. Bigrams of syntactic labels for authorship discrimination of short texts. *Literary and Linguistic Computing*, 22(4):405–417.

P. Hopper. 1987. Emergent Grammar. In *Proceedings of the Berkeley Linguistics Society*, pages 139–157.

D. Hovy and C. Purschke. 2018. Capturing regional variation with distributed place representations and geographic retrofitting. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, pages 4383–4394.

A. Jørgensen, D. Hovy, and A. Søgaard. 2015. Challenges of studying and processing dialects in social media. *Proceedings of the Workshop on Noisy User-Generated Text*, pages 9–18.

D. Jurgens, Y. Tsvetkov, and D. Jurafsky. 2017. Incorporating Dialectal Variability for Socially Equitable Language Identification. In *Proceedings of the Annual Meeting of the Association for Computational Linguistics*, pages 51–57.

B. Kachru. 1990. *The Alchemy of English The spread, functions, and models of non-native Englishes*. University of Illinois Press, Urbana-Champaign.

T. Kreutz and W. Daelemans. 2018. Exploring Classifier Combinations for Language Variety Identification. *Proceedings of the Workshop on NLP for Similar Languages, Varieties and Dialects*, pages 191–198.

M. Kroon, M. Medvedeva, and B. Plank. 2018. When Simple n-gram Models Outperform Syntactic Approaches Discriminating between Dutch and Flemish. *Proceedings of the Workshop on NLP for Similar Languages, Varieties and Dialects*, pages 225–244.

R. Langacker. 2008. *Cognitive Grammar A basic introduction*. Oxford University Press.

L. Lucy and D. Bamman. 2021. Characterizing English variation across social media communities with BERT. *Transactions of the Association for Computational Linguistics*, 9:538–556.

H. Lütkepohl and M. Krätzig. 2004. *Applied Time Series Econometrics*. Cambridge University Press.

J. Nerbonne and W. Wiersma. 2006. A measure of aggregate syntactic distance. In *Proceedings of the Workshop on Linguistic Distances*, pages 82–90.

Dat Quoca Nguyen, Dai Quocb Nguyen, Dang Ducc Pham, and Son Baod Pham. 2016. A robust transformation-based learning approach using ripple down rules for part-of-speech tagging. *AI Communications*, 29(3):409–422.

S. Petrov, D. Das, and R. McDonald. 2012. A universal part-of-speech tagset. In *Proceedings of the International Conference on Language Resources and Evaluation*, pages 2089–2096.

A. Rahimi, T. Baldwin, and T. Cohn. 2017. Continuous representation of location for geolocation and lexical dialectology using mixture density networks. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, pages 167–176.

N. Sanders. 2007. Measuring syntactic difference in British English. In *Proceedings of the ACL Student Research Workshop*, pages 1–6.

E. Stamatatos. 2009. A survey of modern authorship attribution methods. *Journal of the American Society for Information Science and Technology*, 60(3):538–556.

H. Tayyar Madabushi, L. Romain, D. Divjak, and P. Milin. 2020. CxGBERT: BERT meets Construction Grammar. In *Proceedings of the International Conference on Computational Linguistics*, pages 4020–4032.

D. Villegas, D. Preoţiuc-Pietro, and N. Aletras. 2020. Point-of-Interest Type Inference from Social Media Text. In *arXiv*, 2009.14734.

B. Wing and J. Baldridge. 2014. Hierarchical Discriminative Classification for Text-Based Geolocation. In *Proceedings of the Conference on Empirical Methods in NLP*, pages 336–348.

H. Xia and B. Carlin. 1998. Spatio-temporal models with errors in covariates: Mapping Ohio lung cancer mortality. *Statistics in Medicine*, 17:2025–2043.

E. Zenner, D. Speelman, and D. Geeraerts. 2012. Cognitive Sociolinguistics meets loanword research: Measuring variation in the success of anglicisms in Dutch. *Cognitive Linguistics*, 23(4):749–792.

Subject Verb Agreement Error Patterns in Meaningless Sentences: Humans vs. BERT

Karim Lasri[ə,β] **Olga Seminck**[ə] **Alessandro Lenci**[β] **Thierry Poibeau**[ə]
[ə]Lattice (École Normale Supérieure-PSL, CNRS, U. Sorbonne Nouvelle)
[β]University of Pisa
karim.lasri@ens.psl.eu olga.seminck@cnrs.fr
alessandro.lenci@unipi.it thierry.poibeau@ens.psl.eu

Abstract

Both humans and neural language models are able to perform subject-verb number agreement (SVA). In principle, semantics shouldn't interfere with this task, which only requires syntactic knowledge. In this work we test whether meaning interferes with this type of agreement in English in syntactic structures of various complexities. To do so, we generate both semantically well-formed and nonsensical items. We compare the performance of BERT-base to that of humans, obtained with a psycholinguistic online crowdsourcing experiment. We find that BERT and humans are both sensitive to our semantic manipulation: They fail more often when presented with nonsensical items, especially when their syntactic structure features an attractor (a noun phrase between the subject and the verb that has not the same number as the subject). We also find that the effect of meaningfulness on SVA errors is stronger for BERT than for humans, showing higher lexical sensitivity of the former on this task.

1 Introduction

Subject Verb Agreement (SVA) is a grammatical constraint in English and several other natural languages, such that verbs must agree in number with their subject. Linguistic theories generally assume that SVA obeys two main principles: i.) **structure dependence** (SD) - SVA is governed by phrasal structure, rather than surface linear order (i.e., the verb agrees with the syntactic subject); ii.) **meaning independence** (MI) - SVA is a morphosyntactic constraint that holds for meaningless sentences too (e.g., *Colorless green ideas sleep furiously*) (Chomsky, 1956, 1971; Chomsky et al., 1976). However, previous research has shown that humans are prone to making agreement errors with specific constructions (Bock and Miller, 1991; Hartsuiker et al., 2001), for example when a noun with a different number (also called an **attractor**) occurs between the subject (the **cue**) and the verb (the **target**). See (1) from Bock and Miller (1991) for an example where agreement can be disturbed by an attractor, as human subjects often show preference for a syntactically ill-formed sentence:

(1) [The **readiness**]$_{\text{subject}}$ [of our conventional forces$_{\text{attractor}}$]$_{\text{PP}}$ [*are*]$_{\text{verb}}$ at an all-time low.

This evidence suggests that the SD principle of SVA might be weaker than it is typically assumed and can be disrupted or disturbed under specific conditions. At the same time, such violation prompts the need to carefully test whether the MI principle of SVA is also compromised in subjects' grammaticality judgments.

SVA has become a widespread testbed to investigate the syntactic knowledge that **neural language models** (NLMs) are able to acquire. A key question is to explore to what extent their competence of syntax obeys the same constraints as those of humans, by comparing the behavior of NLMs with subjects' judgments. In this paper, we pursue this goal by focusing on the SD and MI properties of SVA in humans and NLMs. Its contribution is twofold: i.) we collect original human data on meaningful and meaningless stimuli featuring syntactic structures of varying complexities; ii.) we analyze and compare the error patterns in humans and in NLMs. This allows us to address the following questions: i.) are the SD and MI assumptions about SVA truly supported by human judgments? ii.) do humans and NLMs make similar SVA errors in structures with attractors and/or in meaningless sentences? This may help understand to what extent NLMs rely on syntactic knowledge abstracted from training examples. By comparing the error patterns of humans and NLMs on SVA, we can derive important information about the the nature of their linguistic competence: Is the ability of NLMs on SVA completely meaning independent? Is it influenced by the complexity of the syntactic structure?

Proceedings of the 29th International Conference on Computational Linguistics, pages 37–43
October 12–17, 2022.

Structure	Structure description	Example
A	Simple agreement	[The **author**]$_{\text{subject}}$ [*laughs/*laugh*]$_{\text{verb}}$
B	In a sentential complement	[The mechanics]$_{\text{subject}}$ [said]$_{\text{VComp}}$ [[the **author**]$_{\text{SComp}}$ [*laughs/*laugh*]$_{\text{verb}}$]$_{\text{Comp}}$
C	Across a prepositional phrase	[The **mechanic**]$_{\text{subject}}$ [near the author]$_{\text{PP}}$ [*smiles/*smile*]$_{\text{verb}}$
D	Across a subject relative clause	[The **author**]$_{\text{subject}}$ [that likes the movie]$_{\text{subj. RC}}$ [*laughs/*laugh*]$_{\text{verb}}$
E	In a short verb phrase coordination	[The **author**]$_{\text{subject}}$ [laughs and *swims/*swim*]$_{\text{VP Coord}}$
F	Across an object relative clause	[The **author**]$_{\text{subject}}$ [that the mechanics like]$_{\text{obj. RC}}$ [*smiles/*smile*]$_{\text{verb}}$
G	Within an object relative clause	[The author]$_{\text{subject}}$ [that the **mechanics** *like/*likes*]$_{\text{obj. RC}}$ [smiles]$_{\text{verb}}$
H	Across an object relative clause (*no that*)	[The **author**]$_{\text{subject}}$ [the mechanics like]$_{\text{obj. RC}}$ [*smiles/*smile*]$_{\text{verb}}$
I	Within an object relative clause (*no that*)	[The author]$_{\text{subject}}$ [the **mechanics** *like/*likes*]$_{\text{obj. RC}}$ [smiles]$_{\text{verb}}$

Table 1: Agreement structures used in this study, from Marvin and Linzen (2018). The cue is bolded, the target is in italic, and the attractor is underlined. For each target, we display both the correct and incorrect inflections.

2 Related Work

Linzen et al. (2016) tested Long Short Term Memory (LSTM) language models and their performance on the SVA task. They found that these models can capture a non-trivial amount of grammatical structure but that they are insufficient for capturing complex syntax-sensitive dependencies. In Marvin and Linzen (2018) the capacity of LSTM language models to perform the SVA task is compared with human data. In particular for sentences including attractors, the models perform worse than humans. Linzen and Leonard (2018) tested recurrent neural networks (RNNs) on the task to see whether the error pattern was similar to humans. They concluded that despite important similarities, there was a different behavior within relative clauses. In particular, RNNs are sensitive to the number of attractors, whereas humans are not.

Gulordava et al. (2018) investigated the role of semantics on the performance of RNNs on SVA, testing such neurals models against meaningless, or "nonce", sentences built from various syntactic constructions. They used meaningless sentences where RNNs could not rely on semantic or lexical cues. For Italian, they found that the performance of an LSTM model and the performance of humans were comparable for meaningless items and semantically congruent ones. Lasri et al. (2022) tested a transformer model, BERT-base (Devlin et al., 2019), on its capacity of subject-verb agreement on the items from Marvin and Linzen (2018) and on semantically disrupted sentences featuring the same syntactic constructions. They found that although the model generalized well for simple templates, it failed on meaningless items with attractors.

3 Experimental Setup

In this section, we describe the procedure to construct the experimental items used to collect human judgments with crowdsourcing and to test NLM behavior on SVA.

Items We test the syntactic structures in Table 1, four of which include an attractor. For each syntactic template, we generate 30 meaningless sentences by sampling random words with the correct part-of-speech at each position of the template, using an extensive vocabulary (see Appendix A for more details). The generated sentences are meaningless in the sense that our generation procedure does not include any semantic constraint designed to grant meaning to the sentence. The resulting word sequence should therefore be semantically ill-formed, similar to the classic example *Colorless green ideas sleep furiously*. As some generated sentences can accidentally happen to be interpretable, we also tried to manually remove such sentences when crafting our stimuli. However, as meaning is a subtle notion, some sentences might be interpretable in very specific contexts after our manual filtering. Still, we expect such cases to be very rare in our dataset. Every minimal pair consists of sentences similar to (2).

(2) a. *The admissions sings.
b. The admissions sing.

We also sample 30 meaningful sentences for each structure from Marvin and Linzen (2018) to collect human data.[1] As the filtered M&L data used to collect human performances resulted in a very limited number of sentences built on a limited vocabulary, we extract BERT's performance on the whole dataset for M&L, following the procedure in Goldberg (2019).

We thus collect human performance on 30 items for each of our 2 conditions (Nonce and M&L), and each of our 9 syntactic structures, for a total of

[1] We will call this data **M&L**. We filter out sentences where the verb is 'be', as this verb is very frequent in English.

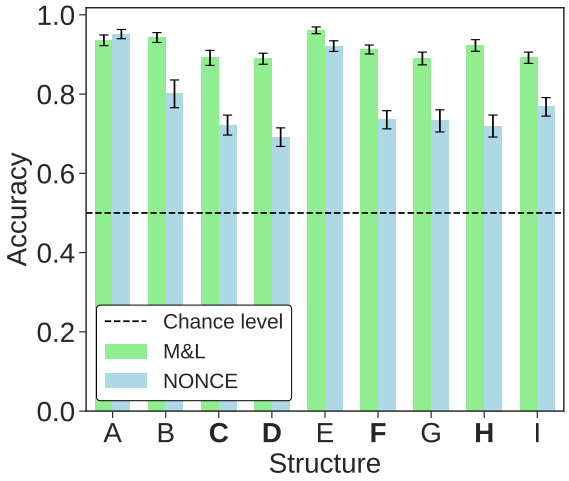

Figure 1: Human accuracies on the SVA task. Structures where an attractor is present are displayed in bold. The error bars displayed represent the 95% confidence interval.

540 items. More details are given in Appendix B.[2]

Neural Language Model Our tested NLM is BERT (Devlin et al., 2019), a bi-directional pre-trained transformer model. BERT has been shown to possess a number of syntactic abilities (Jawahar et al., 2019; Goldberg, 2019), the nature of which remain scarcely understood (Baroni, 2019). For each item, we present BERT with sentences masked at the target position and compare the probability that BERT assigns to each verb inflection.

Collection of Human Judgements We collect our human data using the online click working platform Prolific.[3] We implemented a binary choice experiment in Psychopy (Peirce, 2007) hosted on Pavlovia[4] where participants were presented with a minimal pair, such as in (2), and asked which sentence was the most correct. In order to prevent habituation to our stimuli and task, we used 64% of filler items. We recruited 300 participants to obtain 20 responses per item and kept the responses of 270 participants in our final data set. Their mean speed to judge one item was 6.9 seconds. See Appendix B for more details about human data collection.

[2]Our data is available at https://github.com/karimlasri/agreement-humans-bert

[3]http://prolific.co

[4]https://pavlovia.org

4 Results

We first discuss the human data, which we then compare with BERT's performance.

4.1 Error Patterns for Humans

In this analysis, we compare human accuracy on the nonce stimuli and on Marvin and Linzen's (2018) sentences. In Figure 1, we break down the results by syntactic structure to observe whether the construction type affects the human judgments. We notice a performance drop in all structures with nonce sentences, except for A where the apparent increase is not significant, as shown by the error bars. Interestingly, the structures with an attractor (bolded in the x-axis) are those for which the performance drop seems to be the highest. We also observe high performance drops in sentences where there is no attractor (B, G and I).

Looking at Table 1, we can see that these structures are more complex than the structures where the effect of meaningfulness is low (A and E). Indeed, they contain either a complement (B), or a relative clause (G, I). Surprisingly, we observe a similar pattern on meaningless sentences in (F) and (G): humans seem to be perturbed as much by the attractor within the object relative clause (F), as they are by the material in the main clause (G), if sentences are meaningless. This evidence in comprehension seems opposite to Bock and Cutting's (1992) claim that agreement production is only sensitive to information within the clause of the target. This evidence hints at the possibility of a difference in the mechanisms that support SVA in production and comprehension.

Metric	Correlation
M&L Accuracy	0.61
Nonce Accuracy	0.65
Accuracy Drop	0.52

Table 2: Coefficient of determination between BERT's and human performance on SVA, averaged across syntactic structures. The accuracy drop condition represents the difference between average performance on M&L's stimuli and our nonce stimuli, as seen in Figure 3.

4.2 How Similar are the Human and BERT's Error Patterns?

In this analysis, we compare the performance achieved by BERT with human performance, on each of our stimuli types. Figure 2 displays the

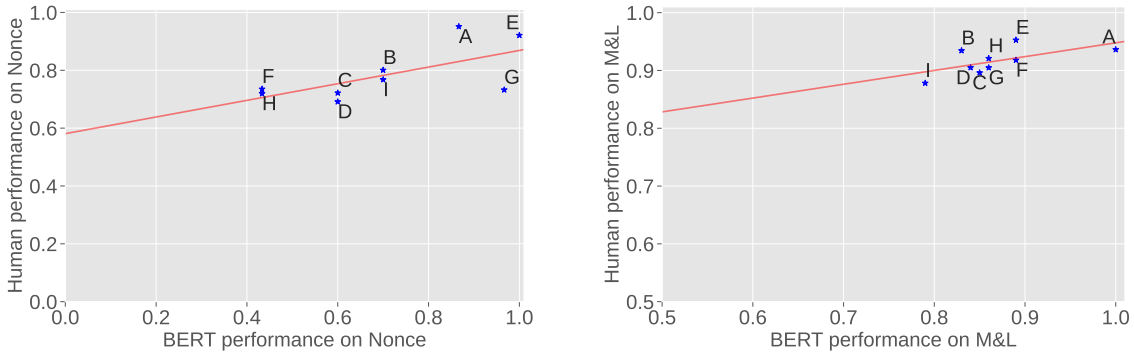

(a) Human nonce performance vs. BERT

(b) Human M&L performance vs. BERT

Figure 2: A comparison of human performance against BERT's performance on each of our structures.

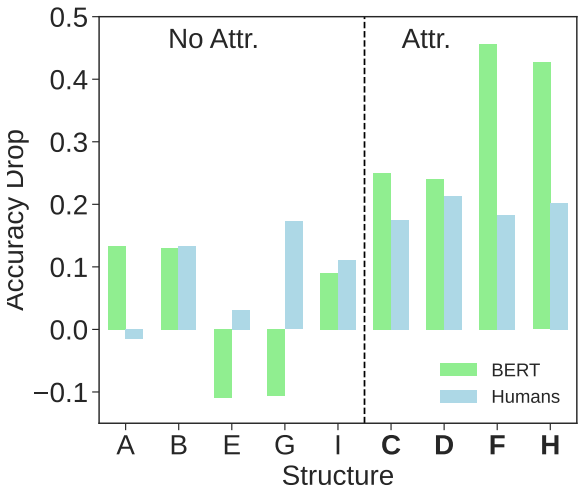

Figure 3: Performance drops between M&L and nonce stimuli.

result obtained by humans against BERT's performance for each syntactic template, for both meaningful and meaningless sentences. Interestingly, there seems to be a fairly high alignment between the results for each syntactic construction, and for each source of stimuli. We display the R^2 correlation measurement of our fit in Table 2. The latter confirms the observed alignment, as we obtain quite high correlations (0.61 for meaningful sentences and 0.65 for nonce sentences).

However, we observe that while the variation in performance obtained by humans across templates seems quite low, BERT's performance does seem to be more affected by the different structures. This is especially true in the case of nonce sentences, as seen in Figure 2a. We also observe a difference in performance decrease in Figure 3, as BERT's performance drops are overall higher in presence of an attractor compared to those of humans. On the other hand, BERT has a higher performance drop on (A) and humans on (G). This in turn could be explained by the fact that (G) is a hard sentence to process for humans, the target of the agreement being within an embedded relative clause, while BERT could rely on local context in this case as observed by Lasri et al. (2022).

5 Discussion

5.1 Lexicalization and Syntactic Generalization

While subject-verb agreement is sometimes considered as a purely syntactic phenomenon, our results show that actually humans also rely on semantics, which goes against the meaning independence hypothesis. Our results also show that BERT is also highly dependent on semantics, a finding in line with Bernardy and Lappin (2017), who mention that "*deep neural networks require large vocabularies to form substantive lexical embeddings in order to learn structural patterns*". This highlights the strong connection between the ability to process linguistic structure and the semantic content of sentences.

5.2 Structure Dependence

Throughout this study, we observed that the performance of both humans and BERT were sensitive to the syntactic structure used in our items. Humans clearly obtain lower performance on sentences that are more complex to process when they are meaningless, including but not limited to sentences presenting an attractor. This variation in performance seems to reflect variation in structure complexity, which upholds SD. On the other hand, BERT seems to be mostly sensitive to sentences with attractors. This evidence rather shows a violation of SD, as at-

tractors are only related to the target by linear order, in line with evidence found by Lasri et al. (2022). While human and BERT's results seem to correlate to a large extent, these divergences could reflect a difference in processing. For instance, SVA in sentence comprehension for humans could depend on having read the whole sentence, while BERT could rely more on local context for this task. Indeed, a fine-grained analysis performed in previous work showed BERT to be mostly sensitive to the replacement of linearly close tokens (Lasri et al., 2022).

6 Conclusion

Throughout this work, we have shown that the ability to perform SVA is highly dependent on the syntactic construction when presented with meaningless sentences. The failures of humans seem to align well with those of BERT overall, as sentences with attractors tend to compromise meaning independence when processing the agreement relation. Despite these similarities, we further show that the performance drop is generally higher in BERT on meaningless sentences, and that humans are more perturbed by complex constructions without an attractor. This finding can in turn reflect differences in processing syntactic structure, either reflecting more reliance on local context for BERT, or a difference between agreement processing in production and comprehension, which could be the source of the partial mismatch in the observed error patterns.

Ethics Statement

The authors foresee no ethical concerns with the work presented in this paper.

Acknowledgements

This work was funded in part by the French government under management of Agence Nationale de la Recherche as part of the "Investissements d'avenir" program, reference ANR-19-P3IA-0001 (PRAIRIE 3IA Institute).

References

Marco Baroni. 2019. Linguistic generalization and compositionality in modern artificial neural networks. *Philosophical Transactions of the Royal Society B*, 375(1791).

Jean-Phillipe Bernardy and Shalom Lappin. 2017. Using deep neural networks to learn syntactic agreement. In *Linguistic Issues in Language Technology, Volume 15, 2017*. CSLI Publications.

Steven Bird, Ewan Klein, and Edward Loper. 2009. *Natural language processing with Python: analyzing text with the natural language toolkit*. " O'Reilly Media, Inc.".

Kathryn Bock and J.Cooper Cutting. 1992. Regulating mental energy: Performance units in language production. *Journal of Memory and Language*, 31(1):99–127.

Kathryn Bock and Carol A Miller. 1991. Broken agreement. *Cognitive psychology*, 23(1):45–93.

Noam Chomsky. 1956. Three models for the description of language. *IRE Transactions on information theory*, 2(3):113–124.

Noam Chomsky. 1971. *Problems of knowledge and freedom: The Russell lectures*. Pantheon Books.

Noam Chomsky et al. 1976. *Reflections on language*. Temple Smith London.

European Commission. 2018. 2018 reform of eu data protection rules.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, Minneapolis, Minnesota. Association for Computational Linguistics.

Allyson Ettinger. 2020. What BERT is not: Lessons from a new suite of psycholinguistic diagnostics for language models. *Transactions of the Association for Computational Linguistics*, 8:34–48.

Yoav Goldberg. 2019. Assessing bert's syntactic abilities. *CoRR*, abs/1901.05287.

Kristina Gulordava, Piotr Bojanowski, Edouard Grave, Tal Linzen, and Marco Baroni. 2018. Colorless green recurrent networks dream hierarchically. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 1195–1205, New Orleans, Louisiana. Association for Computational Linguistics.

Robert J Hartsuiker, Inés Antón-Méndez, and Marije Van Zee. 2001. Object attraction in subject-verb agreement construction. *Journal of Memory and Language*, 45(4):546–572.

Ganesh Jawahar, Benoît Sagot, and Djamé Seddah. 2019. What does BERT learn about the structure of language? In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 3651–3657, Florence, Italy. Association for Computational Linguistics.

Karim Lasri, Alessandro Lenci, and Thierry Poibeau. 2022. Does BERT really agree ? fine-grained analysis of lexical dependence on a syntactic task. In *Findings of the Association for Computational Linguistics: ACL 2022*, pages 2309–2315, Dublin, Ireland. Association for Computational Linguistics.

Tal Linzen, Emmanuel Dupoux, and Yoav Goldberg. 2016. Assessing the ability of LSTMs to learn syntax-sensitive dependencies. *Transactions of the Association for Computational Linguistics*, 4:521–535.

Tal Linzen and Brian Leonard. 2018. Distinct patterns of syntactic agreement errors in recurrent networks and humans. *CoRR*, abs/1807.06882.

Rebecca Marvin and Tal Linzen. 2018. Targeted syntactic evaluation of language models. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 1192–1202, Brussels, Belgium. Association for Computational Linguistics.

George A Miller. 1995. Wordnet: a lexical database for english. *Communications of the ACM*, 38(11):39–41.

Jonathan W Peirce. 2007. Psychopy—psychophysics software in python. *Journal of neuroscience methods*, 162(1-2):8–13.

Robyn Speer, Joshua Chin, Andrew Lin, Sara Jewett, and Lance Nathan. 2018. Luminosoinsight/wordfreq: v2.2.

A Vocabulary Selection

We used the vocabulary of Lasri et al. (2022) for the generation of our nonce items and filtered it. We selected a vocabulary of nouns and verbs that checked the following criteria:

1. We filter out tokens that are ambiguous, i.e. tokens which can either be a noun or a verb. We used Wordnet (Miller, 1995) implemented in the NLTK library (Bird et al., 2009) in python 3 to check whether a word was not classified as a noun and a verb by checking whether there was no synset in the other category.

2. We filter out words using their relative frequency measured by using the python library wordfreq (Speer et al., 2018).[5] We choose to filter too frequent words because some were ambiguous with another category (*e.g.* in the noun vocabulary we can find *good, well, one*). We decided to remove infrequent words to prevent that participants would not know their meaning. For example, this filtered out *polynomial*, and *consonant* from the noun vocabulary.

3. We only keep words with a length ranging from 3 and 8 characters, in order to prevent big differences in size between the items produced by one template.

4. We make a subdivision between transitive and intransitive verbs in order to correctly fill the templates.

[5] https://pypi.org/project/wordfreq/

B Collection of Human Judgments

Setup To collect our human judgements, we recruited participants on the working platform Prolific. Participants were redirected to the Pavlovia page hosting our experiment. First, they had to give their informed consent. Their data was processed in accordance to the European General Data Protection Regulation (Commission, 2018), and no sensitive data has been collected. After being informed, participants were shown brief instructions about the forced choice task. For each item, they were presented with two sentences, and asked to select the one that seemed more acceptable using the keyboard arrows. Each session started with three training items followed by feedback. When the training was finished, they were notified that the experiment started and that they would not receive feedback anymore. Each participant was presented with 100 items and thereafter received a message that the experiment was over.

Number of Items and Participants In total, the participants replied to 100 items: 64 fillers, and 36 experimental item. 18 were from the nonsense condition and 18 from the M&L data set. As every condition features 9 different structures, 2 structures of each category where shown per participant. In order to collect 20 responses per item, we recruited 300 participants with 15 different versions of the online experiment, which can be found in the supplement material of this article.

Fillers Our filler items where from Ettinger (2020). They feature semantically appropriate and inappropriate completions. We also used filler items with correct and incorrect determiners among 'a/an' depending on the following noun to feature syntax-oriented fillers as well.

Selection of Participants We only accepted participants with the United States nationality with English as a first language between the age 18 and 60 years old. We excluded participants that already contributed to another version of the experiment.

Reward Participants got rewarded 2.25£ for a participation that was estimated to take 15 minutes. This was estimated to be a 'good' hourly pay by the Prolific platform. We rejected participants that performed the experiment in less than 5 minutes.

Filtering and Loss of Participants In our final data set we have the contributions of 270 participants. Participants that performed at chance level (50 % accuracy) were filtered out. Furthermore, we lost some data of participants that did not close the experiment correctly in Pavlovia.

Measuring Morphological Fusion Using Partial Information Decomposition

Michaela Socolof
McGill University/Mila

Jacob Louis Hoover
McGill University/Mila

Richard Futrell
University of California, Irvine

Alessandro Sordoni
Microsoft Research

Timothy J. O'Donnell
McGill University/Mila
Canada CIFAR AI Chair

Abstract

Morphological systems across languages vary when it comes to the relation between form and meaning. In some languages, a single meaning feature corresponds to a single morpheme, whereas in other languages, multiple meaning features are bundled together into one morpheme. The two types of languages have been called agglutinative and fusional, respectively, but this distinction does not capture the graded nature of the phenomenon. We develop a mathematically precise way of characterizing morphological systems using partial information decomposition, a framework for decomposing mutual information into three components: unique, redundant, and synergistic information. We show that highly fusional languages are characterized by high levels of synergy.

1 Introduction

Languages are, to a large extent, systematic; there are predictable patterns in the way that meanings are mapped to forms. However, languages differ when it comes to the nature of the relation between meaning and form. This variability is particularly apparent in the domain of morphology, and underlies the distinction between so-called **agglutinative** and **fusional** languages (von Humboldt, 1825; Greenberg, 1960). The two types of languages differ in the extent to which multiple **units of meaning** are expressed by a single morpheme. In this paper, a unit of meaning simply refers to a semantic (or grammatical) feature such as *plural* or *accusative*. Highly agglutinative languages have words that are built up of clearly separable morphemes, each of which corresponds to an individual unit of meaning. The relationship between meaning and form in these languages is thus highly systematic. On the other hand, highly fusional languages fuse together multiple units of meaning into a single affix that cannot be decomposed in any obvious way, and so are less systematic.

Hungarian		Russian	
Meaning	*Form*	*Meaning*	*Form*
cat-SG-DAT	macská-∅-nak	cat-SG-DAT	кот-у
cat-PL-DAT	macská-k-nak	cat-PL-DAT	кот-ам
cat-SG-TERM	macská-∅-ig	cat-SG-GEN	кот-а
cat-PL-TERM	macská-k-ig	cat-PL-GEN	кот-ов

Table 1: In Hungarian (left), every unit of meaning tends to correspond to a morpheme hence the meaning-form relationship is systematic. On the contrary, in Russian (right) such correspondence cannot be found. We aim to quantify the degree of systematicity in meaning-form relations across morphological systems.

In Table 1, we illustrate the meaning-form mapping for words in Hungarian (an agglutinative language) and Russian (a fusional language). In the Hungarian paradigm, *singular*, *plural*, *dative*, and *terminative* each always correspond to a single morpheme, which is the same across contexts. In Russian, the affixes package together multiple units of meaning and cannot be decomposed: there are no morphemes that individually correspond to *singular*, *plural*, *dative*, or *genitive*—rather, the form of the suffix depends on multiple meaning units.

The agglutinative versus fusional distinction captures a core intuition about the different ways meaning can correspond to morphological form, but the distinction is binary and therefore does not characterize the graded nature of the phenomenon (Greenberg, 1960)—that is, the fact that different languages (and, indeed, specific domains within a language) show varying degrees of fusion. In this paper, we take an information-theoretic approach to quantifying systematicity in meaning-form relations across morphological systems.

The core insight we draw upon is that meanings can contribute information about a linguistic form in three different ways. First, a unit of meaning can provide information about the form that no other

Proceedings of the 29th International Conference on Computational Linguistics, pages 44–54
October 12–17, 2022.

unit of meaning provides. This is called **unique** information. Second, a unit of meaning can provide the exact same information about the form that another unit of meaning provides. This is called **redundant** information. Third, a unit of meaning can, in combination with some other unit of meaning, jointly provide information that is not provided by either on its own. This is called **synergistic** information. Going on these definitions, we expect fusional languages to have a higher relative amount of synergy than agglutinative languages.

We argue that these three kinds of information in morphological systems correspond precisely to existing notions of unique, redundant, and synergistic information in the information theory literature. In particular, the **Partial Information Decomposition** (PID) framework, introduced by Williams and Beer (2010), decomposes the mutual information between a target variable and two (or more) source variables into unique, redundant, and synergistic information. This decomposition of mutual information into three components makes up the **information profile** of a system. When we take form to be the target variable and the individual meaning features to be the source variables, the information profile gives the amount of information conveyed individually, concurrently, or jointly, by units of the meaning about form. Crucially, two systems can have equal mutual information between meaning and form, but different information profiles, corresponding to different degrees of morphological fusion. Therefore, PID offers a mathematically precise way of placing morphological systems along an agglutinative-to-fusional spectrum.

In summary, our contributions are as follows. We use the PID framework to develop a novel measure of the systematicity of meaning-form mappings in morphological systems. To validate our method, we first carry out two simulations using artificial languages for which we can control the degree of morphological fusion. We show that languages possessing a low relative amount of synergistic information are the most systematic. Finally, we apply the decomposition to morphological systems in 22 real languages, successfully recapitulating existing linguistic categorizations in a graded way.

2 Partial information decomposition

2.1 The problem

A fundamental property of language is that linguistic forms depend on the meaning being communicated. Information theory gives us a way of quantifying this dependence, with **mutual information**, a measure of how much one random variable informs us about another random variable (Shannon, 1948; Fano, 1961). Let M and F be discrete random variables representing meaning and form, respectively. The mutual information $I(M;F)$ between M and F can be expressed as:

$$I(M;F) = \sum_{m \in M} \sum_{f \in F} P(m,f) \log \frac{P(m,f)}{P(m)P(f)}. \quad (1)$$

In a linguistic system, both the meaning and the form have internal structure, and it is the relationship between subparts of these structures that we are interested in. We therefore define both M and F as ensemble random variables, made up of sets of random variables corresponding to the individual units of meaning and form. As an example, consider the two toy languages in Table 2. In both languages, M is an ensemble random variable made up of two binary random variables (one for each column). Similarly, F is composed of two binary random variables. Assuming a uniform distribution on the inputs, the mutual information between M and F in both languages is 2 bits, since it takes 2 bits of information on average to communicate about the meaning. However, the mutual information on its own does not tell us whether the relation between meaning and form variables is one-to-one, many-to-one, etc. In the language on the left, one variable (i.e., column) on the meaning side fully determines each variable on the form side. In the second language, both meaning variables are needed to correctly predict each form variable.

Since mutual information does not tell us how the information is distributed among the pieces of meaning and form, we want to decompose mutual

M	$\rightarrow$	F
aa		00
ab		01
ba		10
bb		11

M	$\rightarrow$	F
aa		00
ab		01
ba		11
bb		10

Table 2: (left) An example of a fully *systematic*, or one-to-one, code, in which each variable in F is informative about a variable of M. (right) This code is *less systematic* because the value of each F variable depends on more than one M variable. Here $F = \mathrm{CNOT}(M)$. Both codes have $I(M;F) = 2$ bits.

Collection	Associated information about T
$\{S_1\}$	Unique (U_1) of S_1
$\{S_2\}$	Unique (U_2) of S_2
$\{S_1\}\{S_2\}$	Redundancy ($R_{1,2}$) of S_1 and S_2
$\{S_1, S_2\}$	Synergy ($S_{1,2}$) of S_1 and S_2

Table 3: Collections and associated information quantities for the case of two source variables about a target variable T.

information based on how each meaning variable contributes information—on its own, redundantly with other variables, or jointly with other variables.

2.2 Partial Information Decomposition

Decomposing mutual information requires extending traditional information theory to handle multivariate interactions, such as that between two or more meaning variables that jointly provide information about a form variable. Williams and Beer (2010)'s PID framework provides an influential solution to the decomposition problem; we briefly summarize the framework here.

Williams and Beer (2010) set up the problem as a decomposition of the ways that *source* variables provide information about a *target* variable. Consider the simple case of two source variables S_1 and S_2 and a target variable T. Let a *collection* be a grouping of one or more nonzero subsets of source variables such that none of the subsets is a superset of any other. There are four such collections: $\{S_1\}$, $\{S_2\}$, $\{S_1\}\{S_2\}$, and $\{S_1, S_2\}$. Each collection is then associated with a particular quantity of information, summarized in Table 3.[1] The sum of these quantities is the mutual information $I(S_1, S_2; T)$. For the sake of brevity, we will use U, R, and S as shorthand for unique, redundant, and synergistic information, and subscripts to indicate information about T from source variables S_1 and/or S_2.

Williams and Beer show that the collections can be naturally structured into a partially-ordered lattice, shown for the case of two source variables in Figure 1. At the bottom is the information provided redundantly by S_1 and S_2. The next level up is the information provided uniquely by S_1 and the information provided uniquely by S_2. At the top is the information jointly contributed by S_1 and S_2, i.e., the synergy. An important feature of the lattice is that the mutual information between a set of sources and the target is the sum of all nodes

[1]This can be generalized to an arbitrary number of source variables. See Williams and Beer (2010) for details.

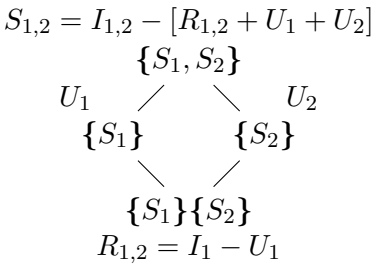

Figure 1: Partial information lattice for the case of two source variables. The equations at each node are abbreviated versions of equations (2)–(4), showing how to solve for redundant, unique, and synergistic information, starting at the bottom of the tree.

below and including the collection consisting of that particular set of sources. This means that the values at all nodes in the entire lattice add up to the mutual information provided by the two sources S_1 and S_2 about the target, as expressed in Equation 2. It also means that the mutual information between a single source S_1 and the target is made up of the unique information in S_1 plus whatever information is redundant between S_1 and S_2, expressed in Equation 3 (and the same for S_2, in Equation 4).

$$I(S_1, S_2; T) = R_{1,2} + U_1 + U_2 + S_{1,2} \quad (2)$$
$$I(S_1; T) = R_{1,2} + U_1 \quad (3)$$
$$I(S_2; T) = R_{1,2} + U_2 \quad (4)$$

These equations all have a mutual information term on the left, which we have a definition for and can therefore compute. However, we do not at this point know how to compute any of the terms on the right, so we have a system of three equations with four unknowns, which we cannot solve.

Gutknecht et al. (2020), building on Williams and Beer (2010), show that with a definition of either redundant information or unique information, it is possible to solve the system of equations 2–4 for the remaining variables using a Möbius inversion function to move recursively up the lattice. Much work in the PID literature has focused on formulating an independent definition for redundant or unique information (e.g., Williams and Beer, 2010; Bertschinger et al., 2014; Finn and Lizier, 2018; Makkeh et al., 2021). A number of solutions have been proposed, and there is as yet not total consensus on the "best" measure. Below, we will adopt one such measure, which is both common in the literature and intuitive for our application—that of Bertschinger et al. (2014).

Bertschinger et al. give an independent definition for unique information. Their measure is based on the intuition that the unique information of S_1 should reflect the information about T which is *only* available from S_1, regardless of the choice of S_2. This is operationalized by adversarially computing the minimum possible conditional mutual information $I_Q(S_1 : T \mid S_2)$, minimizing over all possible joint distributions $Q(S_1, S_2, T)$ that have the same marginals as the true distribution P:

$$U_1 = \min_{Q \in \Delta_P} I_Q(S_1; T \mid S_2) \tag{5}$$

where

$$\begin{aligned}\Delta_P = \{Q \in \mathfrak{P}(S_1, S_2, T) \mid \\ \textstyle\sum_{s_2' \in \mathcal{S}_2} Q(s_1, s_2', t) = P(s_1, t) \wedge \\ \textstyle\sum_{s_1' \in \mathcal{S}_1} Q(s_1', s_2, t) = P(s_2, t) \\ \forall t \in \mathcal{T}, s_1 \in \mathcal{S}_1, s_2 \in \mathcal{S}_2\}\end{aligned}$$

where $\mathfrak{P}$ is the set of all joint distributions.

The Bertschinger et al. (2014) formulation of PID is known to give intuitive results on a number of canonical example distributions; for example in the mapping from the second variable of meaning M to the second variable of form F in the codes of Table 2, we get a unique information of 1 for the fully systematic example and 0 for the less systematic example. In Section 3.2 we define a measure of morphological fusion based on the Bertschinger et al. (2014) formulation.

3 Methods

We compute PID between meaning and form of noun paradigms in suffixing languages from UniMorph (Sylak-Glassman, 2016), which contains annotated morphological data for 167 languages using a universal schema. An example paradigm is in Table 4. All of the languages in our experiment have noun paradigms with exactly two non-stem meaning feature categories: CASE and NUMBER.

3.1 Defining meaning and form variables

In order to compute the partial information decomposition, we first need to define our source and target random variables. Since we are interested in how each component of meaning contributes individually or jointly to determining linguistic forms, we treat meaning variables as our sources and form variables as our targets.

Meaning	*Form*
cat-NOM-SG	кот
cat-NOM-PL	коты
cat-GEN-SG	кота
cat-GEN-PL	котов
cat-DAT-SG	коту
cat-DAT-PL	котам
cat-INS-SG	котом
cat-INS-PL	котам
cat-ESS-SG	коте
cat-ESS-PL	котах

Table 4: Subset of paradigm for the Russian noun кот.

Source Meaning Variables Consider the morphological paradigm for the Russian noun кот in Table 4, which consists of inflected forms of the word paired with their grammatical information. For our source variables, we treat each meaning feature category (CASE, NUMBER, and the stem) as a random variable with values that range over the possible feature values (e.g., *nominative* or *singular* for CASE and NUMBER, respectively).

Target Form Variables In order to define our target random variables, it is necessary to decompose the suffixes in some way, since treating the entire suffix as a target would not allow us to investigate its degree of internal agglutination or fusion. To define random variables over forms, we adopt an alignment-based approach, breaking up the suffixes into morphological slots and treating each slot as a random variable whose values range over the different aligned sequences that appear in the slot. We perform the alignment using LingPy's morphological aligner (List and Forkel, 2021). In order to compute PID, it is necessary for the number of random variables to be consistent across all words in the paradigm, so we pad empty slots with a dummy character. The number of random variables, then, is determined by the word with the longest suffix in the paradigm. In the majority of alignments, each slot ends up containing a one- or two-character sequence. An example alignment of several Russian words and the resulting form slots is shown in Table 5.

Our application differs from the original PID formulation in that we are dealing with multiple target variables. In Section 3.2 we propose an expectation-based approximation of PID for the joint distribution over multiple targets. In what follows, meaning random variables are denoted by $\mathcal{M} = \{M_1, \ldots, M_n\}$, while the form random variables are denoted by $\mathcal{F} = \{F_1, \ldots, F_m\}$. M_1 and

M_1	M_2	M_3	F_1	F_2	F_3	F_4
cat	GEN	SG	кот	а	-	-
cat	DAT	PL	кот	а	м	-
cat	INS	PL	кот	а	м	и

Table 5: Random variable structure for three word forms in Russian.

F_1 represent the stem's meaning (e.g., *cat*) and the stem's form (e.g., кот), respectively (Table 5).

3.2 Computing PID

Within each language, we compute PID on each noun's paradigm individually. Our motivation for treating each noun separately is that in many languages, morphological paradigms vary based on features of particular stems. For example, in a language with a gender distinction, the combination of meaning features *accusative+plural* might be expressed differently on masculine versus feminine nouns. We argue that this is not relevant to the notion of agglutinative versus fusional that we are interested in. If *accusative* and *plural* are expressed by separate morphemes in masculine as well as feminine nouns, then the fact that their specific forms vary with gender does not make the language any less agglutinative. It would be possible to extend our approach to handle stem-specific features in a dataset that made this information available, but since UniMorph does not annotate these features, we proceed with computing PID on each noun separately. With this approach, we are essentially treating the stem as a proxy for any stem-specific information, and conditioning all of our probability distributions, and thus our PID quantities, on the stem. In Appendix A, we give an example of how aligning multiple UniMorph-style paradigms without accounting for stem-specific features can obscure systematic regularities.

Since we treat each paradigm separately, the form and meaning variables M_1 and F_1 corresponding to the stem are generally uninteresting to us, as they remain constant throughout each paradigm. This approach is equivalent to computing the information-theoretic quantities in PID conditioned on the stem variables M_1, F_1. We are left with exactly two source variables, which correspond to CASE and NUMBER, and $m-1$ target variables. In what follows, for simplicity, we relabel the meaning variable corresponding to CASE as M_1 and the one corresponding to NUMBER as M_2.

We are now challenged with computing the PID between M_1, M_2 and the set of form variables $F \in \mathcal{F} - \{F_1\}$. We do this by taking the expectation of the PID quantities of these two variables with each form variable as target, separately. We first compute the PID between the two meaning variables M_1, M_2 and one of the form variables $F \in \mathcal{F} - \{F_1\}$ and obtain the values of unique $U_{1,2\to F}$, redundant $R_{1,2\to F}$, and synergistic $S_{1,2\to F}$ information, where we have made the dependence on the particular form variable F explicit in the subscript for clarity. We then normalize these quantities by the mutual information $I(M_1, M_2; F)$ to obtain the relative amount of each type of information. For each combination of meaning variables M_1, M_2 and one form variable F, the proportion of unique, redundant, and synergistic information that M_1 and M_2 give about F is:

$$\bar{U}_{1,2\to F} \coloneqq \frac{U_{1\to F} + U_{2\to F}}{I(M_1, M_2; F)} \tag{6}$$

$$\bar{R}_{1,2\to F} \coloneqq \frac{R_{1,2\to F}}{I(M_1, M_2; F)} \tag{7}$$

$$\bar{S}_{1,2\to F} \coloneqq \frac{S_{1,2\to F}}{I(M_1, M_2; F)} \tag{8}$$

To compute the total average amount of each type information for a given language, we average these quantities across the full set of form variables of the paradigm and across paradigms. Let $h \in \{\bar{U}, \bar{R}, \bar{S}\}$. The average amount of information of type h in the language is:

$$\bar{h} = \frac{1}{|\mathcal{N}|} \sum_{n\in\mathcal{N}} \frac{1}{|\mathcal{F}|} \sum_{F\in\mathcal{F}-\{F_1\}} h_{1,2\to F} \tag{9}$$

where $\mathcal{N}$ is the set of paradigms in our dataset. We give pseudo-code for this process in Appendix B.

We use an implementation of Bertschinger et al. (2014)'s measure given in Wollstadt et al. (2018). Computing PID for the full set of nouns in every language is computationally intensive, so instead we repeatedly subsample the paradigms for $|\mathcal{N}| = 10$ different nouns, randomly selected, from each language. We do this 100 times per language.

4 Experiments

We first validate that Bertschinger et al. (2014)'s PID measure captures the phenomenon we are interested in by running the measure on noun paradigms in two sets of artificial languages. After validating the measure, we then apply the PID framework to noun paradigms in 22 real languages, showing that the proportion of synergy characterizes the degree of fusion in a linguistic system.

4.1 Artificial languages — intuition

Our artificial languages are generated based on the intuition that in a highly agglutinative language, each inflection corresponds to a single unit of meaning, whereas in a highly fusional language, each inflection corresponds to a combination of meanings. We operationalize these intuitions by generating random languages where inflections are sampled either conditioned on single meaning features (agglutinative) or sampled conditioned on pairs of meaning features (fusional). We test on a set of very simple artificial languages as well as a set of artificial languages that were generated to match a number of statistical properties of real languages in our dataset, and thus control for a variety of linguistic phenomena.

4.2 Artificial languages — simple

We generated fifteen very simple artificial languages. In each language, the noun paradigms had six cases and three numbers. The first five languages were "agglutinative," where the suffixes were two-character strings, with one character independently generated conditionally on one meaning variable. A second set of five "fusional" languages were generated such that each suffix was a random two-character string sampled conditionally on both meaning features. Finally, as a sanity check we generated a set of five baseline languages that were intended to be as synergistic as possible. Under the Bertschinger et al. (2014) measure, XOR is a maximally synergistic boolean function. Therefore, the control languages were generated using XOR. Each suffix was a single character long with two possible realizations corresponding to the boolean values output by the XOR function and given by $F(\text{case}, \text{number}) = (\text{case} \in C)$ XOR $(\text{number} \in N)$, where C and N are random nonempty proper subsets of the possible case and number values, respectively. The PID results for these artificial languages in Figure 2 confirm that the measure captures the differences between these artificial languages as expected. All five agglutinative languages have 100% unique information, while the XOR languages have majority synergistic information. The fusional languages fall in the middle, with a proportion of synergy between 20% and 40%.

4.3 Artificial languages — linguistic controls

In our second experiment, we validate our measures using more linguistically-realistic artificial languages that are matched to real languages for specific properties, such as the size of the character vocabulary, phonotactic restrictions, and average suffix length, as well as other properties that may correlate with the agglutinative/fusional distinction. We do this by generating agglutinative and fusional versions of existing languages.

We began by selecting six languages whose noun paradigms are given in UniMorph. Each of the languages in UniMorph is labelled as either agglutinative or fusional, based on information from linguistic analysis; two of our chosen languages (Hungarian and Turkmen) are labeled as agglutinative, and the remaining four (Ukrainian, German, Latin, and Northern Sami) are labeled as fusional. For each language, we trained a 3-gram model on all the language's inflected nouns, to approximate the language's phonotactics, and used this model to generate artificial paradigms for that language. For each language we sampled fifty artificial agglutinative paradigms and fifty artificial fusional paradigms following the sampling scheme outlined above. To sample an artificial fusional paradigm, we used the 3-gram model to generate random suffixes for the stem, jointly conditioned on case and number. To generate an artificial agglutinative paradigm, we generated independent strings for each value of case and number, and concatenated them (in either order, but consistent within a paradigm). For both types, we sampled suffixes with a range of lengths to roughly match that of the suffixes in the real language. The PID results are shown in Figure 3. These results confirm that our PID measure captures the difference between agglutinative and fusional paradigms in the expected way: The agglutinative versions of the languages had proportionally less synergistic and more unique information than the fusional versions, regardless of which type of language they were generated from.

4.4 Real languages

We investigate whether PID provides a way of measuring morphological fusion by computing PID on noun paradigms from 22 languages in UniMorph. Seven of our languages are labeled as agglutinative, and the remaining ones as fusional. Our results are given in Figure 4, which shows the relative amount of unique, redundant, and synergistic information for each language. Languages with an asterisk and solid black outline are those that were labelled as

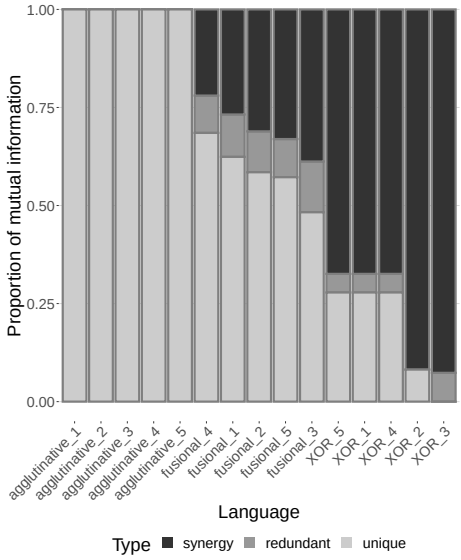

Figure 2: Results of partial information decomposition on noun paradigms in baseline artificial languages. The languages are sorted by relative amount of synergy.

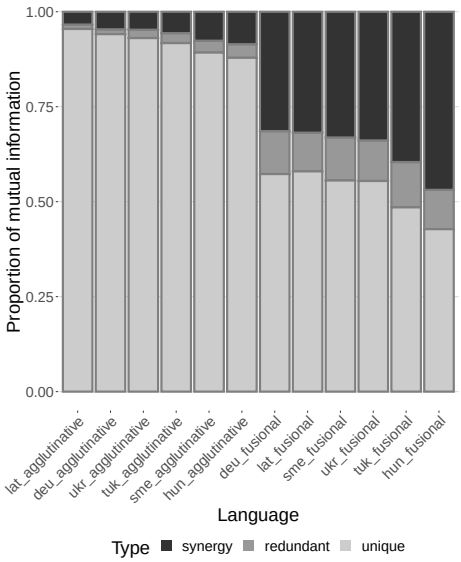

Figure 3: Results of partial information decomposition on noun paradigms in linguistically-controlled artificial languages. The languages are sorted by relative amount of synergy.

agglutinative in UniMorph. We find that the seven agglutinative languages fall on the side of lowest synergy, though there were also a few fusional languages that had low synergy.

As baselines for the PID measure, we present (1) a plot of the average amount of mutual information between meaning and form in the individual nominal paradigms across the 22 languages in our experiment (Figure 5 in Appendix C), and (2) a plot of the average number of suffix slots in each language (Figure 6 in Appendix C). The baselines suggest that high mutual information and high suffix length are often present in agglutinative languages, but our artificial experiments reveal that when we control for these factors, PID successfully captures the amount of fusion present in a system.

5 Discussion

Our results suggest that PID does indeed capture the spectrum between agglutinative and fusional. We also find that there is more unique information overall than redundant or synergistic information, which points to an overall high level of systematicity in morphology. Redundant information makes up the smallest proportion of information overall, suggesting that morphological systems are not particularly redundant. This raises the question of whether other domains in language show similar levels of redundancy, and how the low amount of redundancy should be accounted for.

While PID seems to be able to capture morphological fusion, it is important to note that the agglutinative/fusional classification system is very coarse—when we apply a single label to each language, we miss fine-grained distinctions such as the fact that different domains within a language can have different degrees of fusion. For this reason, we believe that when evaluating any measure of fusion, it is best to examine the actual paradigms. Let us consider a paradigm from Latin, shown in Table 6. Latin falls far to the right side of the spectrum, and we can see in the paradigm that there is a lack of systematicity among the suffixes. For example, the *-s* in column F_5 appears with both singular and plural, and across four different cases. The PID values for each combination of variables are given in Table 7. We can see that for every combination of two meaning variables and one form variable, there is more synergy than any other type of information.

6 Related work

There is a growing literature on information-theoretic approaches to problems in morphology and syntax. One line of work looks at the trade-off between the surprisal of a linguistic form and the

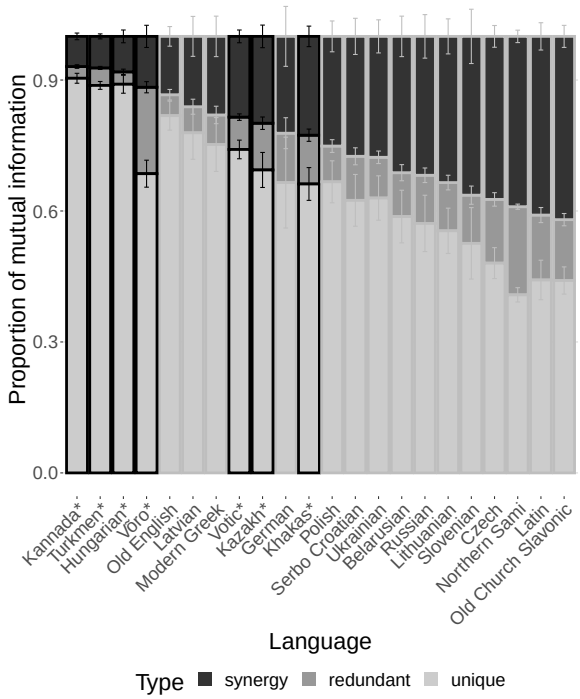

Figure 4: Results of partial information decomposition on noun paradigms in 24 languages. The languages are sorted by relative amount of synergy. Asterisks and dark borders represent languages labeled as agglutinative in UniMorph.

M_1	M_2	M_3	F_1	F_2	F_3	F_4	F_5
nur	NOM	SG	nur	-	-	u	s
nur	NOM	PL	nur	ū	s	-	-
nur	GEN	SG	nur	ū	s	-	-
nur	GEN	PL	nur	-	-	uu	m̄
nur	DAT	SG	nur	-	-	u	ī
nur	DAT	PL	nur	i	b	u	s
nur	ACC	SG	nur	-	-	u	m
nur	ACC	PL	nur	ū	s	-	-
nur	ABL	SG	nur	ū	-	-	-
nur	ABL	PL	nur	i	b	u	s
nur	VOC	SG	nur	-	-	u	s
nur	VOC	PL	nur	ū	s	-	-

Table 6: Random variable structure for a Latin noun.

s_1	s_2	t	U_1	U_2	R	S
M_2	M_3	F_2	0.323	0.135	0.16	0.865
M_2	M_3	F_3	0.445	0.39	0.014	0.61
M_2	M_3	F_4	0.355	0	0.136	0.833
M_2	M_3	F_5	0.689	0	0.095	1

Table 7: PID values (unique, redundant, synergistic) for the Latin paradigm in Table 6, unnormalized.

time it takes to produce (Pimentel et al., 2021); the trade-off between surprisal and memory in accounting for word and morpheme order cross-linguistically (Hahn et al., 2021); and mutual information as a measure of the relationship between grammatical gender and co-occurring words (Williams et al., 2021). Accounting for patterns of word and morpheme order across languages using information theory has yielded a variety of proposed measures (Hahn et al., 2020; Dyer et al., 2021).

Closely related to our work is Rathi et al. (2021), which proposes a measure of *informational fusion* in morphology, based on Wu et al. (2019)'s definition of morphological irregularity. Let ℓ be a lexeme, σ a semantic feature combination, and w a surface form. *Informational fusion* is defined as:

$$\phi(w) = -\log p(w \mid \mathcal{L}_{-\sigma}, \sigma, \ell) \qquad (10)$$

Informational fusion is a measure of the surprisal of a surface form given the rest of the paradigm. Unlike the PID approach, which involves segmenting the suffix and finding the information profile of each subpart, informational fusion is computed with respect to un-segmented forms, and does not make reference to individual morphemes. PID gives us a way of investigating the exact question we are interested in—to what extent do units of the meaning individually or jointly contribute information about individual units of the form? We use PID to get at the fine-grained distinctions between information profiles, an approach that we believe can be extended to study compositionality more generally.

7 Conclusion

We have proposed a novel way of characterizing morphological systems cross-linguistically, using partial information decomposition. PID allows us to decompose the mutual information between meaning and form into three distinct components: unique, redundant, and synergistic information. We argued that the relative amount of synergistic information provides a mathematically precise and intuitive measure of the degree of fusion in a morphological system. We carried out a study on noun paradigms, demonstrating the promise of this approach in this specific domain. Our study applies PID at the level of morphemes, and suggests extensions to word- and sentence-level domains, potentially leading to a more general theory of compositionality. We see PID as an exciting tool for investigating the information profile of any system in which meaning features are expressed by linguistic forms.

Acknowledgments

We thank the Montreal Computational & Quantitative Linguistics Lab and the Language Processing Group at the University of California, Irvine for helpful feedback. We also gratefully acknowledge the support of the Natural Sciences and Engineering Research Council of Canada, the Fonds de Recherche du Québec, and the Canada CIFAR AI Chairs Program.

References

Nils Bertschinger, Johannes Rauh, Eckehard Olbrich, Jürgen Jost, and Nihat Ay. 2014. Quantifying unique information. *Entropy*, 16:2161–2183.

William Dyer, Richard Futrell, Zoey Liu, and Greg Scontras. 2021. Predicting cross-linguistic adjective order with information gain. In *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*, pages 957–967, Online. Association for Computational Linguistics.

Robert M. Fano. 1961. *Transmission of information: a statistical theory of communications*. MIT Press.

Conor Finn and Joseph T. Lizier. 2018. Pointwise partial information decomposition using the specificity and ambiguity lattices.

Joseph H. Greenberg. 1960. A quantitative approach to the morphological typology of language. *International Journal of American Linguistics*, 26(3):178–194.

Aaron J. Gutknecht, Michael Wibral, and Abdullah Makkeh. 2020. Bits and pieces: Understanding information decomposition from part-whole relationships and formal logic. *CoRR*, abs/2008.09535.

Michael Hahn, Judith Degen, and Richard Futrell. 2021. Modeling word and morpheme order in natural language as an efficient tradeoff of memory and surprisal. *Psychological Review*.

Michael Hahn, Dan Jurafsky, and Richard Futrell. 2020. Universals of word order reflect optimization of grammars for efficient communication. *Proceedings of the National Academy of Sciences*, 9117(5):2347–2353.

Wilhelm von Humboldt. 1825. Über das Entstehen der grammatischen Formen und ihren Einfluss auf die Ideenentwicklung. In *Abhandlungen der Königlichen Akademie der Wissenschaften zu Berlin: Aus den Jaren 1822 und 1823*, pages 401–430.

Johann-Mattis List and Robert Forkel. 2021. A Python library for historical linguistics. version 2.6.9.

Abdullah Makkeh, Aaron J. Gutknecht, and Michael Wibral. 2021. Introducing a differentiable measure of pointwise shared information.

Tiago Pimentel, Clara Meister, Elizabeth Salesky, Simone Teufel, Damián Blasi, and Ryan Cotterell. 2021. A surprisal–duration trade-off across and within the world's languages. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics.

Neil Rathi, Michael Hahn, and Richard Futrell. 2021. An information-theoretic characterization of morphological fusion. In *EMNLP*.

C. E. Shannon. 1948. A mathematical theory of communication. *Bell System Technical Journal*, 27(3):379–423.

John Sylak-Glassman. 2016. The composition and use of the Universal Morphological Feature Schema (UniMorph Schema).

Adina Williams, Ryan Cotterell, Lawrence Wolf-Sonkin, Damian Blasi, and Hanna Wallach. 2021. On the relationships between the grammatical genders of inanimate nouns and their co-occurring adjectives and verbs. *Transactions of the Association for Computational Linguistics*, 9:139–159.

Paul L. Williams and Randall D. Beer. 2010. Nonnegative decomposition of multivariate information.

Patricia Wollstadt, Joseph T. Lizier, Raul Vicente, Conor Finn, Mario Martinez-Zarzuela, Pedro Mediano, Leonardo Novelli, and Michael Wibral. 2018. Idtxl: The Information Dynamics Toolkit xl: a Python package for the efficient analysis of multivariate information dynamics in networks. *Journal of Open Source Software*, 4(34):1081.

Shijie Wu, Ryan Cotterell, and Timothy J. O'Donnell. 2019. Morphological irregularity correlates with frequency. In *Association for Computational Linguistics*.

A

As an illustration of how computing PID on the full set of noun paradigms without accounting for stem-conditioned features can obscure the patterns, consider the following toy paradigms:

M_1	M_2	F_1
NOM	SG	a
NOM	PL	b
ACC	SG	a
ACC	PL	b

Table 8: Toy language, noun 1.

M_1	M_2	F_1
NOM	SG	a
NOM	PL	a
ACC	SG	c
ACC	PL	c

Table 9: Toy language, noun 2.

In the first paradigm, M_2 uniquely determines F_1. In the second paradigm, M_1 uniquely determines F_2. For both nouns, there is 1 bit of unique information, and no redundant or synergistic information. Thus all of the mutual information between meaning and form in this language is unique. However, if we compute PID on the full set of forms without conditioning on noun 1 and noun 2, we get 0.66 bits of unique information, 0.016 bits of redundant information, and 0.077 bits of synergistic information. This irregularity comes from the fact that the suffix *-a* serves different functions for the different nouns, but the PID measure considers both types of *-a* to be the same realization of F_1. Crucially, this means we can get synergy in a language whose individual paradigms do not actually have any synergy.

B

```python
def compute_pid(paradigm):
  N = paradigm.num_nouns
  F = paradigm.num_F # num form variables
  M = 2              # num meaning variables (2)
  V = numpy.zeros((N, M + F))
  # fill the matrix of values
  vtoi = dict()
  for n in range(N):
    for m in range(M):
      # convert string value of s to int
      value = paradigm[n].meaning[m]
      if value not in vtoi:
        vtoi[value] = len(vtoi)
      V[n, m] = vtoi[value]
    for f in range(F):
      # convert string value of f to int
      value = paradigm[n].form[f]
      if value not in vtoi:
        vtoi[value] = len(vtoi)
      V[n, f] = vtoi[value]
  # compute PID for each target var
  bar_u, bar_r, bar_s = 0, 0, 0
  for f in range(F):
    u, r, s, mi = pid(
        V, sources=[0, 1], target=2 + f
    )  # Bertschinger's PID using IDTXL
    bar_u += u / mi # avg. unique
    bar_r += r / mi # avg. redundant
    bar_s += s / mi # avg. synergy
  return bar_s/F, bar_u/F, bar_r/F
```

Listing 1: Python-style pseudo-code for computing relative PID quantities for a given paradigm.

C

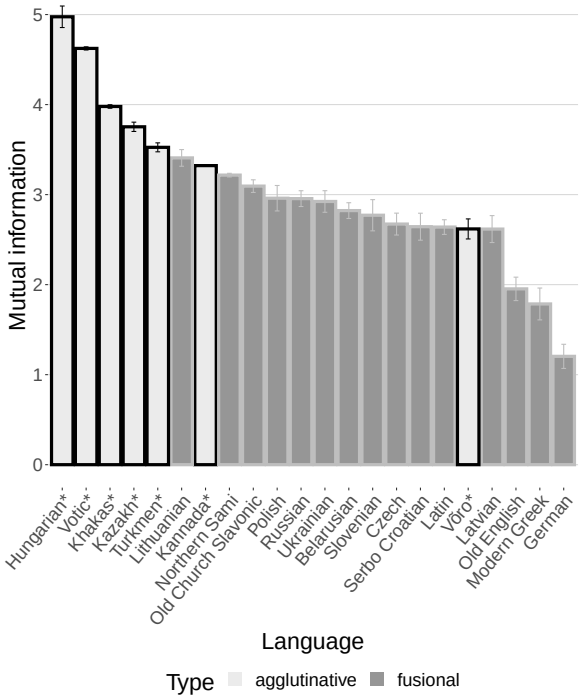

Figure 5: Average amount of mutual information between meaning and form in the nominal paradigms of 22 languages. Asterisks and dark borders represent languages labeled as agglutinative in UniMorph.

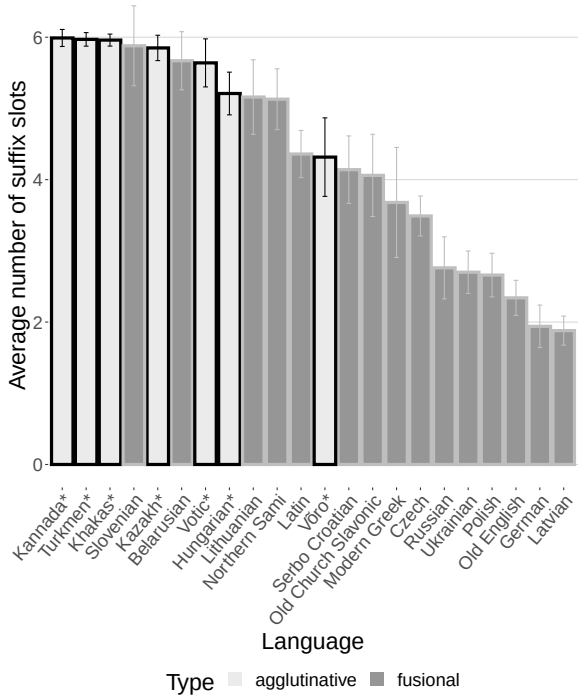

Figure 6: Average suffix length in the nominal paradigms of 22 languages. Asterisks and dark borders represent languages labeled as agglutinative in UniMorph.

Smells like Teen Spirit: An Exploration of Sensorial Style in Literary Genres

Osama Khalid, Padmini Srinivasan
University of Iowa
{osama-khalid, padmini-srinivasan}@uiowa.edu

Abstract

It is well recognized that sensory perceptions and language have interconnections through numerous studies in psychology, neuroscience, and sensorial linguistics. Set in this rich context we ask whether the use of sensorial language in writings is part of linguistic style? This question is important from the view of stylometrics research where a rich set of language features have been explored, but with insufficient attention given to features related to sensorial language. Taking this as the goal we explore several angles about sensorial language and style in collections of lyrics, novels, and poetry. We find, for example, that individual use of sensorial language is not a random phenomenon; choice is likely involved. Also, sensorial style is generally stable over time - the shifts are extremely small. Moreover, style can be extracted from just a few hundred sentences that have sensorial terms. We also identify representative and distinctive features within each genre. For example, we observe that 4 of the top 6 representative features in novels collection involved individuals using olfactory language where we expected them to use non-olfactory language.

1 Introduction

Sensory perceptions shape how we use language and communicate (Paradis, 2003). When we use sensorial words (i.e. words with meanings connected to our senses) like *fuzzy* or *stinky*, besides communicating sensorial experiences these also stimulate perceptual systems in the recipient's mind (Speed and Majid, 2020).

The space of senses – sometimes called the "Aristotelian" senses (Sorabji, 1971), include the five modalities: visual, auditory, haptic, gustatory, and olfactory. Relatively recently, linguistics and psychologists have added a sixth sense — interoception (Craig, 2002). This refers to the perception of sensations from inside the body, both physical such as hunger and pain, and emotional, such as joy. This sensory space has been the basis of much prior research.

Sensorial Linguistics is about studying how language relates to the senses. A key focus has been to study how different sensorial experiences and perceptions are packaged into linguistic units (Winter, 2019). Researchers have looked at how some senses dominate in language (Winter et al.), how sensorial language varies across lexical categories (Lievers and Winter, 2018) and how sensory experiences influence sensorial language (Croijmans et al., 2019; Murphy, 2019). However, the domain of sensorial linguistics is still nascent with many unexplored questions.

Stylometrics: As individuals grow, besides consciously learning a vocabulary they also develop a linguistic style. Some stylistic elements may be acquired subconsciously and others by choice. Stylistic choices can reflect the individual's social reality or affective state (Savoy, 2020). Several categories of stylistic features have been identified, such as the use of function words and language complexity (Holmes, 1998). Stylometrics is important for goals such as author attribution and affect classification.

Style in sensorial language use: A key limitation in stylometrics is that linguistic style around sensorial language has not been studied systematically. A person who wants to express being depressed has several word choices. She could use "sad" or the less frequent "downcast". Her propensity to choose one or the other may be considered as part of her linguistic style. Consider a cloudy scene. A person may use visual language focusing on color, and say "*the clouds are white*". Another may use haptic language focusing on texture, "*the clouds are fluffy*". While sensorial language is clearly important for communication, we do not yet know if there are distinguishable patterns in sensory language use at the level of individuals, texts, etc. This gap in stylometrics motivates us to ask the following about sensorial language style:

Proceedings of the 29th International Conference on Computational Linguistics, pages 55–64
October 12–17, 2022.

- **RQ 1:** Is the notion of sensorial style meaningful or is it a product of random chance?
- **RQ 2:** How much data do we need to get a stable representation of sensorial style?
- **RQ 3:** Does sensorial style vary with time?
- **RQ 4:** Which features are representative and distinctive of the individuals within each genre?

2 Representing Sensorial Language Style

Sensorial style may be represented at different levels of abstractions. At the lowest level, we can represent the proportion of an individual's language that is sensorial and also examine the frequencies of different sensorial words. At a higher level of abstraction, we can ask how frequent are different sense modalities (visual, auditory, etc.) in an individual's language. Alternatively, we can represent style by the extent to which a person's use of sensory modalities aligns with general expectations. This is related to synaesthesia, where one sense modality is used when another is expected - a well studied phenomenon in sensorial linguistics (Lievers, 2015; De Ullmann, 1945). As an example in their work, Lievers and Huang (2016a) developed a lexicon of perception that they used to automatically identify perception related synaesthetic metaphors. Similarly we also approach the problem of sensorial style through the lens of their synaesthetic usage.

In the 2010 animated film 'Despicable Me', the character of Agnes hugs a unicorn and says "It is so *fluffy*". One reason why this quote acquired somewhat of a meme status is because it subverted audience expectations of a more visual word like "*pretty*" or "*white*" to describe the unicorn. Instead she opts for the more unexpected haptic word "*fluffy*". This substitution of visual language for haptic, a synaesthesia, might indicate that Agnes' perceives the world in a more tactile manner rather than in a visual way. In order to assess if this is a stylistic tendency, we can examine all of Agnes' language use and ask the general question: to what extent does she use haptic language in contexts where we generally expect visual language? We can ask similar questions related to each combination of expected versus observed sensory modalities. Observations for all combinations, including the homogeneous non-synaesthesia ones, are accumulated to form Agnes' (or any other individual's, group's or genre's) sensory style representation.

Terminology and notation: More formally, we consider a sentence to be a "sensorial sentence" if it has at least one word or phrase that appears in a sensorial lexicon. Further, we define a "sense-focused sentence" to be a sensorial sentence with a single sensorial term selected as focus term. Thus, if a sensorial sentence has n sensorial terms then we derive from it n sense-focused sentences.

Assume $S_i{=}\{S_{i1}, S_{i2}, ..., S_{in}\}$ is the set of n sense-focused sentences identified from the writings of individual $i \in I$. Let $C = \{H, V, I, O, G, A\}$ represent the modalities: Haptic, Visual, Interoceptive, Olfactory, Gustatory, Auditory respectively. Using $\bar{N}$ to represent concepts that are "not-sensorial" we define $\widetilde{C}$ as: $\widetilde{C} = C \cup \{\bar{N}\}$.

We also define two functions. $F(\hat{s}_{ij})$ is a sensory lexicon lookup function that returns the sensorial category $c \in C$ for the focused sensorial term $\hat{s}_{ij}$ in the sense focused sentence S_{ij}. E.g., given the sensorial sentence "*The unicorn is **white** and **fluffy***", we have two sense-focused sentences S_{ij} and S_{ik} corresponding to the two sensorial terms, $\hat{s}_{ij}$ and $\hat{s}_{ik}$. For S_{ij} with the focus term "***white***" $F(\hat{s}_{ij})$ will return V. For S_{ik} with focus term "***fluffy***" it will return H.

The second function we define is $M(S_{ij})$ which returns the "expected" modality $c \in \widetilde{C}$ for the same focus term in S_{ij} We describe this function in Section 2.1.

Calculating observed to expected ratios: We represent an individual's sense-focused sentences as a list of length $|S_i|$. Each entry is a pair of expected and observed modalities of the form $[(M(S_{ij}), (F(\hat{s}_{ij})]$. For observed modality $y \in C$ and expected modality $x \in \widetilde{C}$, the observed to expected ratio α_i^{xy} for individual i is:

$$\alpha_i^{xy} = \frac{|\{S_{it} : F(\hat{s}_{it}) = y \text{ and } M(S_{it}) = x\}|}{|\{S_{it} : M(S_{it}) = x\}|} \quad (1)$$

Note that these ratios are the informative units of sensorial style. For example, if the ratio is 1 when x and y are the same modality, a homogeneous combination, then the individual's use of that modality is highly aligned with general expectation. On the other hand, if it is close to 0 then she deviates considerably from the expected use of modality x, i.e. there is greater synaesthesia.

Style vectors: For each $x \in \widetilde{C}$, we then concatenate its 6 ratios into a vector of the form:

$$s_i^x = \bigsqcup_{y \in C} \alpha_i^{xy} \quad (2)$$

It follows that

$$\sum_{y \in C} \alpha_i^{xy} = \begin{cases} 1, & \text{if } |\{S_{it} : M(S_{it}) = x\}| \geq 1 \\ 0, & \text{otherwise} \end{cases}$$

We can now define the sensorial style vector u_i of $i \in I$ as a concatenation of the seven vectors, one for each expected modality. The size of u_i is $x \times y = 42$.

$$u_i = \bigsqcup_{x \in \widetilde{C}} s_i^x \tag{3}$$

2.1 Implementing function $M(S_{ij})$

Given a sense-focused sentence, function M returns the expected modality $c \in \widetilde{C}$ of the sentence's focus sensorial term as per general expectation in English. We leverage RoBERTa-MLM[1] as a stand-in for general English language usage. RoBERTa is a transformer based language model pre-trained on 160 GB of data (Liu et al., 2019). Prior works like (Mosbach et al., 2020) and (Sinha et al., 2020) have shown that language models learn the norms of the language on which they are trained. This makes them ideal for our task. We mask the focus sensorial words in our sentences and input them to the model. RoBERTa returns the probabilities for all the words in its vocabulary at each masked location. Probabilities represent likelihood of appearance of the words at that location. We use these probabilities to identify the expected sense modality at each masked location as follows.

Let $W = \{(w_1, p_1), (w_2, p_2) \ldots (w_N, p_N)\}$ be the ranked set of words returned by RoBERTa for a masked location (location of focus sensorial term) in S_{ij}; top ranked has highest probability. Using $F(w_k)$, we lookup the sense for each word in the top 100. We combine this information to get an aggregate probability score $\Pi(c, S_{ij})$ for each modality c as follows:

$$\Pi(c, S_{ij}) = \sum_{\substack{k \leq 100 \\ F(w_k) = c}} p_k \tag{4}$$

For greater confidence we only include S_{ij} in our analysis if its majority modality has $\Pi(c, S_{ij}) > 0.5$. We then define

$$M(S_{ij}) = \underset{c \in \widetilde{C}}{\operatorname{argmax}}\, \Pi(c, S_{ij}) \tag{5}$$

In essence, $M(S_{ij})$ returns the expected modality with the highest aggregate probability for the focused sense word in S_{ij} as determined using RoBERTa.

[1]We experimented with BERT as well, however, RoBERTa gave us more accurate results.

2.2 Sensorial Lexicon

	Lexicon			
	Original		Modified	
Modality	N	%	N	%
Visual	29552	75.0	9419	50.2
Interoceptive	3546	9.0	3449	18.4
Auditory	4528	11.5	3803	20.3
Haptic	675	1.7	972	5.2
Gustatory	890	2.3	890	4.7
Olfactory	216	0.5	216	1.2
Total	39407	100	18749	100

Table 1: Distribution of modalities in original Lynott et al. (2020) lexicon and our modified subset lexicon.

We use the sensorimotor norms lexicon published recently (Lynott et al., 2020) which has 39,954 concepts from the English Language. Brysbaert et al. estimates that the average adult lexicon is composed of approximately $42,000$ words. Therefore this lexicon approximates a significant majority of everyday English.

Each concept was rated by annotators along a 0-5 scale for the six modalities (Auditory, Gustatory, Haptic, Olfactory, Visual, Interoceptive) For example "*fluffy*" is rated 4.41 for Haptic, 0.29 for Gustatory, 3.77 for Visual, 0.35 for Interoceptive and 0 for Auditory and Olfactory. The dimension with the highest rating is the dominant modality.

Dominance alone is not enough to ensure that a concept belongs to a particular sensorial modality since almost half of the concepts score less than 2.55 on any sense modality. Therefore, we filter the lexicon by ranking all concepts in a given modality by their rating and selecting only those in the top quartile. This ensures strong alignment to dominant modalities. Table 1 describes the lexicons[2].

3 Methods

3.1 RQ 1: Is Sensorial Style a Product of Random Chance?

In order to be meaningful our representation of sensorial style should not be a product of random chance. If an individual chooses sensorial modalities randomly and not deliberately then we expect her observed and expected modality distributions to be independent of each other. For example in "*the clouds are **white***" the expected modality may be visual but the individual randomly chooses from one of the six senses. We use this random model in our analysis.

[2]There are other sensory lexicons like (Lievers and Huang, 2016b) and (Lynott and Connell, 2013). Besides being the largest and most recent the Lynott et al. (2020) sensorimotor lexicon is the only one to include ratings for interoception.

As a first step, for any $i \in I$ with a set of n sense-focused sentences S_i, we define, $\Gamma(i)$, the distribution of the sense modalities in C observed in S_i. For each sense-focused term $\hat{s_{ij}}$ in S_i, we use a function $\bar{F}(\hat{s_{ij}}, \Gamma(i))$ that returns a random modality $c \in C$ with distribution $\Gamma(i)$.

For each $i \in I$, we create m random pseudo-documents $\mathcal{R}_i = \{\mathcal{R}_i^1, \mathcal{R}_i^2 \ldots \mathcal{R}_i^m\}$. Each random pseudo-document has the same set of sense-focused sentences S_i. However, instead of using $F(\hat{s}_{ij})$ to look up the modality, we use $\bar{F}(\hat{s}_{ij}, \Gamma(i))$ to get a random modality. Equation 6, a modification of equation 1, gives us $\bar{\alpha}_{ik}^{xy}$ which is used to calculate the style vector u_i^k for random pseudo-document $\mathcal{R}_i^k$.

$$\bar{\alpha}_{ik}^{xy} = \frac{\left|\{S_{ij} : \bar{F}(\hat{s}_{ij}, \Gamma(i)) = y \text{ and } M(S_{ij}) = x\}\right|}{|\{S_{ij} : M(S_{ij}) = x\}|} \quad (6)$$

Thus, for each $i \in I$ with sensorial style vector u_i, we have m random style vectors $\{u_i^1, u_i^2 \ldots u_i^m\}$ generated from the random pseudo-documents in $\mathcal{R}_i$.

Let $U_i = \{u_i\} \cup \{u_i^1, u_i^2 \ldots u_i^m\}$. For each vector $v \in U_i$, we calculate its average cosine similarity with all other elements in U_i. Ranking the elements of U_i by decreasing order of average similarity we check whether the style vector $u_i \in U_i$ has lower average similarity than at least 95% of the vectors in U_i (i.e. p-value < 0.05). If so, we infer with 95% confidence that i's style vector, u_i, is not random and therefore likely a product of an individual stylistic choice.

3.2 RQ 2: How much data is needed to describe sensorial style?

Given the set of sense focused sentences S_i, where $|S_i|=n$, we randomly sample subsets from S_i of size k and compute the style vector from each sample. We explore how increasing the values for k affect style convergence.

For a given sentence set size k, we identify m random samples (with replacement) of S_i, each of size k. Thus, we create a set of sentence sets, $T_i^k = \{\hat{T}_i^{k1}, \hat{T}_i^{k2}, \ldots, \hat{T}_{ki}^m\}$. For each sentence set $\hat{T}_i^{kj} \in T_i^k$, we use the method discussed in Section 2 to generate the corresponding style vector $\hat{u}_i^{kj}$. This gives us a set of m sensorial style vectors, $\hat{\boldsymbol{u}}_{\boldsymbol{i}}^{\boldsymbol{k}} = \{\hat{u}_i^{k1}, \hat{u}_i^{k2}, \ldots, \hat{u}_i^{km}\}$. We then use cosine similarity to calculate the average pairwise similarities between all elements in $\hat{\boldsymbol{u}}_{\boldsymbol{i}}^{\boldsymbol{k}}$. We recompute this average self-similarity $\overline{\text{sim}(\hat{\boldsymbol{u}}_{\boldsymbol{i}}^{\boldsymbol{k}})}$ for different values of k in increasing steps of r. We say that the style of the individual has converged for a minimum of k sensorial sentences if $\overline{\text{sim}(\hat{\boldsymbol{u}}_{\boldsymbol{i}}^{\boldsymbol{k}})} \approx \overline{\text{sim}(\hat{\boldsymbol{u}}_{\boldsymbol{i}}^{\boldsymbol{k+r}})}$, where $k + r$ is the next sentence set size tested.

Genre	# Authors	# Works	#Sentences	# Sensorial Sentences	# Sensorial Expressions
Novels	130	317	1,525,894	156,570 (10%)	474,299
Lyrics	5,321	20,785	1,007,090	754,572 (75%)	1,501,501
Poetry	1,246	3,315	85,236	4,979(6%)	8,209

Table 2: Dataset details for each genre. The percentage of total sentences that are sensorial is in parentheses.

3.3 RQ 3: Does sensorial style vary over time?

Here we investigate whether style vectors evolve over time spans. We segment the writings by time and consider how the average similarity in style varies with temporal distance. We first identify all pairs of time points t_a and t_b that are γ duration apart. We then build style vectors for each author with text anchored at t_a and for each author with text anchored at t_b. We then compute the average pairwise similarity between t_a and t_b style vectors. We repeat this for all values of γ that are of interest.

We use a notion of windowing around the time points (t_a and t_b) to reduce noise. Each window is of size δ and distributed equally around each time point. For example, for t_a we create an individual's style vector from all the sense-focused sentences that were published in the range $t_a - \frac{\delta}{2} < \tau_a < t_a + \frac{\delta}{2}$.

3.4 RQ 4: Which features are representative and distinctive of the individuals within each genre?

A genre can be represented by the set of sensorial style vectors of its members. Each style vector is composed of 42 features that explore synaesthesia. We are interested in exploring which features are representative of the members of a genre, and also features that make the members distinct.

We consider a sensorial style feature to be representative if the variation in its usage is low. This would indicate that the members use the feature in a consistent manner. Formally, a stylistic feature α^p is more representative for the members of a genre than another feature α^q if its standard deviation $\sigma(\alpha^p)$, across all the members is lower than the standard deviation $\sigma(\alpha^q)$. At the other end, a high variation would indicate that the feature is distinctive amongst the members.

4 Datasets

We analyze 3 literary genres — novels, poems, music lyrics. Compared to poems and lyrics, novels

can span tens of thousands of sentences. Additionally, novels and poetry are generally associated with a single author. Lyrics are sometimes collaborations, however, we assume an artist would not perform a song that is in a style they do not like. Thus, we assume music lyrics to be a reflection of the artist's style.

Novels: We collected English language novels from the Domestic fiction genre of Project Gutenberg[3]. There were 317 works written by 130 authors, with the earliest by Henry Fielding from the early 18th century and the latest by Rebecca West from the mid-20th century.

Lyrics: We collected songs that were listed on the Billboard Hot 100 charts, 1963 to 2021 (inclusive). This weekly chart ranking of song popularity is considered the industry standard (Whitburn, 2010). We assume the first time a song is listed to be its year of production. We obtained song lyrics using the Genius API[4]. There are $20,785$ song lyrics.

Poetry: Following works like (Lou et al., 2015), we used the corpus of poems available on the Poetry Foundation's[5] website. To make this dataset more comparable to the lyrics dataset, we only included works published after 1963.

5 Results

Genre	$> 95^{th}$	N
Novels	112	123
Lyrics	701	735
Poetry	20	85

Table 3: Number of individuals with lower average similarity than 95% of random vectors.

We present our results in two parts. First, we make our general observations. Second, we present results related to our specific research questions.

5.1 General Observations

Domination over lower senses: The five Aristotelian sensorial modalities have classically been thought of as part of a hierarchy with vision and audition dominating over the three so-called "lower senses"—Touch, Taste, Smell (Howes, 2010)[6]. This hierarchy manifests in the frequency of language use with the visual and auditory modalities being used more often than the lower senses (Majid et al., 2018). We have consistent results. Figure 2 shows that for all three genres visual and auditory dominate over the "lower senses". Concepts associated with haptic, gustatory and olfactory modalities — combined, form less than 10% of the total sensorial language.

While auditory dominates the "lower senses" in all three genres, it occurs less than half as often as visual. Going beyond the classical five senses, in all cases interoception dominates the three "lower senses" surpassing audition in this regard. Additionally, in lyrics, interoception is as common as visual. Clearly interoception with its emphasis on sensations within the body, both physical and emotional, is important in language.

Sensorial style across genres: We investigate how sensorial style varies across genres. Using the method described in Section 2 we calculate genre-level sensorial style vectors by combining sentence sets at the genre level. We show the distribution over 42 sensorial combinations in Figure 1 for just Lyrics. Each cell represents an expected-observed modality combination.

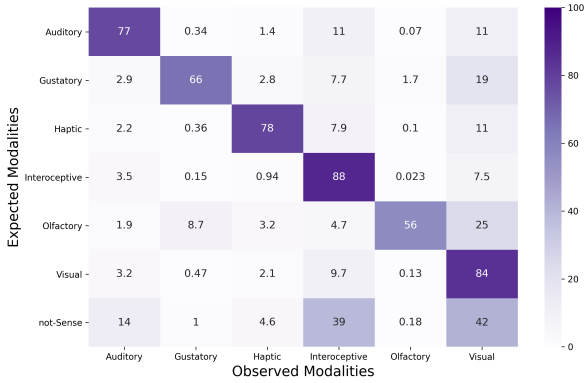

Figure 1: Distribution of expected-observed modalities in Lyrics. Note that we calculate proportions using equation 1, however, for illustrative purposes we show sensorial distribution as percentages.

Observed modalities are largely as expected with some exceptions: The diagonal values which are in the range 56 to 88% indicate that the observed modalities are generally consistent with expected modalities. That is, the individuals in our datasets select from the 6 sensorial modalities in a manner that is consistent with the general norms of language use. The highest consistency is for interoceptive and the lowest is for olfactory. We observe this trend across all genres (see appendix).

Looking at off diagonal values, we observe visual language used in 25% of the cases where we

[3] https://www.gutenberg.org/

[4] Some lyrics were not available on http://www.genius.com, because they were instrumentals like the "*Star Wars Theme*" which hit No. 1 in 1977, or were not in the Genius database.

[5] The Poetry Foundation was established in 2003, and one of its goals is to make "the best poetry" accessible (https://www.poetryfoundation.org/foundation/about).

[6] Note interoception is generally not considered in discussions of this hierarchy, possibly because of its relatively recent inclusion (Connell et al., 2018).

	Novels	Lyrics	Poetry
	$I-O$	$I-O$	$A-G$
	$\bar{N}-O$	$\bar{N}-O$	$A-O$
	$A-O$	$V-O$	$G-I$
	$I-G$	$A-O$	$H-O$
	$V-O$	$I-G$	$O-A$
	$A-G$	$V-G$	$O-H$
Range	(0.00,0.01)	(0.00,0.02)	(0.00,0.00)

Table 4: The top representative features for each genre.

	Novels	Lyrics	Poetry
	$G-G$	$H-H$	$A-A$
	$O-O$	$G-G$	$I-I$
	$H-H$	$A-A$	$H-H$
	$G-V$	$H-V$	$\bar{N}-V$
	$H-V$	$G-V$	$G-G$
	$A-A$	$A-I$	$\bar{N}-I$
Range	(0.43,0.15)	(0.44,0.19)	(0.47,0.27)

Table 5: The top distinctive features for each genre.

Each feature is an (expected,observed) pair e.g., $I-O$ in means that we observe Olfactory language when we expected Interoceptive language. This is the most representative feature for Novels and Lyrics as it has the lowest standard deviation. We include the range of variances of the top most distinctive and representative features.

expected olfactory language, 19% and 11% of the cases where we expected gustatory and haptic language, respectively. This usage of visual language as a replacement for lower senses was observed across all 3 genres. This replacement or *cross-modal compensation* might be because the lower senses do not have a strong relation with the perceptual system and consequently individuals might be relying on visual language as a semantic scaffold to compensate for the weaker perceptual system of the lower senses (Speed and Majid, 2020).

We also observe (in all three genres) that in more than 90% of instances where we expected to see non-sensorial language we instead observed interoception. This might also be because interoception dominates in our data and is consistent with observations about higher senses in the literature (Majid et al., 2018).

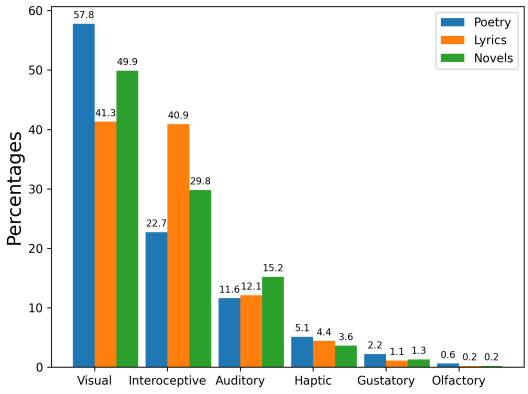

Figure 2: Distribution of observed modalities.

5.2 Results for research questions

RQ1: Is sensorial style a product of random chance? We investigate whether sensorial style is motivated by the individual choices or whether it is a product of randomness. Table 3 provides the results. If sensorial style is a non random phenomenon and a product of individual choice and intent, we expect the sensorial style vectors to be distinct from random vectors. That is, we expect them to have a lower average similarity as compared to random vectors (generated by our random model). Methodological details are in Section 3.1).

Considering all individuals with more than 10 sense-focused sentences[7], more than 90% of the individual sensorial style vectors in the novels and lyrics datasets are non-random. However, in the poetry dataset only around 23% of the individual vectors in the poetry dataset were non-random. A possible reason for the difference in poetry is likely data sparsity — fewer sensorial expressions/ author (see Table 2). Exploring this intuition further we find that the non-random vectors in poetry have on average 159 sense-focused sentences while the remaining vectors that looked random had on average 24 sense-focused sentences. Similarly, 8 out of the 10 most prolific individuals had non-random vectors. However, none of the 10 least prolific individuals had non-random vectors. These support our intuition regarding data sparsity being the cause of the difference in poetry.

RQ2: How much data do we need to get a stable representation of sensorial style? We evaluate the average similarity for each individual with progressively larger samples sizes, k, of their sense-focused sentences. We chose a range of values of k from $k=1$ to $k=10$ with a granularity of 1, from $k=10$ to $k=100$, and $k=100$ to $k=1,000$ with granularities of 10 and 100. In Figure 3 we summarize these results with the median of average similarity across all individuals in each genre. We say the sensorial style vector has converged at a k value when the graph becomes more or less horizontal from that point onward. From the figure we see that as k increases similarity increases (within the m samples of size k). We note that lyrics reaches a median average similarity of 0.90

[7]Because of the volume of lyrics, we limit the analysis to individuals with >500 sense-focused sentences.

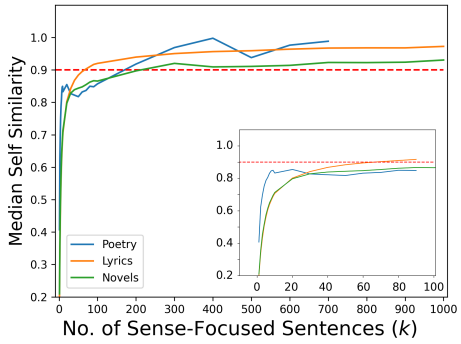

Figure 3: Convergence of style vectors as a function of k, the number of sense-focused sentences sampled.

with a sample of less than 100 sense focused sentences. In contrast, we need between 200 and 300 sentences to get the same 0.9 median average similarity for novels. Compared to novels and lyrics the plot for poetry has some fluctuations at $k \geq 400$, perhaps because there are only 7 poets with more than 100 sentences.

RQ 3: Does sensorial style vary over time?

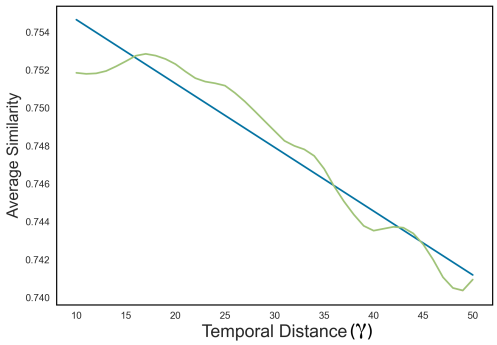

Figure 4: Average similarity between music lyrics as a function of their temporal distance. The blue line represents the linear approximation of the relationship. As temporal distance increases similarity decreases slightly.

We now explore whether sensorial style has changed over time in lyrics. As observed earlier, the poetry dataset is sparse so we do not include it. And with novels, it is sometimes challenging to pinpoint a single publication year, since some texts are written over multiple years[8]. Using the method described in Section 3.3, we measure average similarity in style of lyrics between pairs of individual sensorial style vectors that are a temporal distance γ away from each other. We use window length $\delta = 1.5$ years. Figure 4 illustrates our findings.

We observe that as temporal distance between works increases the similarity of sensorial style decreases. The augmented Dickey-Fuller test had a $p-$value $= 0.96$ meaning we cannot reject H_0, that the distribution is not-stationary (Cheung and Lai, 1995). However, the decrease is very slight; the 50 year drop in similarity is only 0.01. Approximating the relationship between the average similarity and temporal distance as a linear process, we note that the average similarity decreases very slowly at a rate of 3.37×10^{-4} per year.

RQ 4: Which features are representative and distinctive of the individuals within each genre? Table 4, shows the top-6 features that are representative of the members of each genre. We observe that these are all synaesthetic. Additionally the use of olfactory language in non-olfactory contexts is a representative feature in a majority of table cells (10/18). The standard deviations of top representative features is relatively low (between 0.00 to 0.02). This would indicate that the level of consistency for these top features is generally consistent for the three genres. One takeaway from these observations can be that in synaesthetic contexts, individuals are more prone to using lower senses (like olfaction) in a more consistent manner.

Table 5, shows the top-6 features that had the highest standard deviation between the members of each genre and were the most distinctive. The distinctive features were predominantly non-syneasthetic. In the cases where the distinctive feature was synaesthetic, the observed modality was either interoceptive or visual. This would indicate that there is greater diversity in expressions that rely on higher senses as a semantic scaffold.

For each genre, we rank all the sensorial style features by the standard deviation and compare them using Pearson's correlation. We observe that, the features are highly correlated. Lyrics had a high correlation with both novels (0.75) and poetry (0.81), while poetry and novels had a slightly lower correlation of 0.48.

Can sensorial style be used for prediction tasks? We investigate whether sensorial style features can be used to identify genre. We compare against other standard style representations: LIWC (Tausczik and Pennebaker, 2010) and content-free words (CFW) vectors (Hughes et al., 2012). We use standard 5 fold cross validation for each experiment to train and test a random forest classifier. We consider the most prolific 50 authors/genre.

Table 6 shows the results. We observe that sensorial style predicts genre with a high level of accuracy ($> 90\%$). While the other representations achieve close to perfect accuracies, key to note is

[8] As an example, Louisa May Alcott's *Little Women* was published in two volumes spanning two years, 1868 and 1869.

Method	Baseline	Sensorial Style	LIWC	CFW
Features	—	42	73	307
Accuracy	0.33	0.91	0.99	0.99

Table 6: Prediction accuracy of the different features.

that our goal is less about beating baselines and more about understanding the kinds of signals conveyed by sensorial style.

6 Case Study: Sensorial style in Lyrics

As a small illustration, we explore how sensorial style varies across different songs composed by the same artist. We consider all 962 artists who had at least 5 songs in the Hot 100 and extract a sensorial vector for each song. We then measure the average pair-wise cosine similarity (self similarity) amongst the songs of each artist. Almost 80% of the artists had an average self similarity ≥ 0.70. Only two artists had a self similarity < 0.50.

The rapper *NF* was the most consistent artist with an average self similarity of 0.93. Conversely, the least consistent artist was the rock musician *Tommy James* with an average similarity of 0.42. For example, in the song "When I Grow Up", *NF* used auditory language non-synaesthetically in 85.7% of the cases and for the song "NO NAME" this happened in 76.9% of the cases. The similarity between these two songs was over 95%.

In contrast, "Nothing to Hide" and "Ball and Chain" by *Tommy James'* had a similarity of 0.32. In "Ball and Chain", the artist uses visual language in all the instances where it was expected. However in "Nothing to Hide", he uses interoceptive language synaesthetically instead of visual in about 57% of the cases. This case study demonstrates a method for exploring sensorial style and their variations across writings at the individual level.

7 Related Work

There are no directly comparable studies examining sensorial style for large numbers of individuals that consider interoception. Instead we briefly review closely allied topics.

Sensorial language: Sensorial language is not uniformly distributed across the six sensory modalities as reflected in sensorial lexicons, such as the one we use (Lynott et al., 2020). This is also observed in large text collections. In their analysis of 8 million words from around 7,000 English texts (Koblet and Purves, 2020) found over 28,000 visual descriptions and only 78 referring to the olfactory modality. Similar findings for multiple corpora are observed in (Winter et al., 2018). Our results are consistent with these prior works.

Sensorial language, the brain & emotion: The salience of sensorial words is known to be highly correlated with the volumes of cortical activation in the brain (Reilly et al., 2020). Lievers (2015) show that there is directionality to how senses are substituted for each other. Winter (2016) found that gustatory and olfactory words (e.g., 'stinky', 'delicious') are on average more emotionally valenced than visual and auditory words and these also appear in more emotionally valenced sentences.

Bubl et al. (2010), show that alterations in mental states have a direct effect on perception; specifically, that depression directly impacts how the color blue was perceived. Kernot et al. (2016), found a decrease in the novelist Iris Murdoch's use of olfactory language following her diagnosis of depression and Alzheimer's. We credit this study for providing us with the hint that sensorial language may lead to a sensorial style. We take their sensorial style analysis forward with larger collections of authors, several genres and a more informative representation of sensorial style.

8 Limitations and Conclusions

We have shown that individuals have sensorial language style and that this sensorial style is a non random phenomenon for novelists and musicians and therefore is likely developed intentionally. Interestingly, we also found that it takes just a few hundred sentences to extract stable sensorial style representations. We also show that sensorial style in lyrics largely stable over time; the average similarity decreases at a rate of 3.37×10^{-4} per year.

Additionally, we show that sensorial style vectors seem to perform well at genre identification. The performance was high (> 0.90), however, it was not close to perfect as with other style representations. The question about how sensorial style representations can be improved to increase performance requires further investigation.

Our study is limited in that our method relies heavily on the underlying Lynott et al. (2020) lexicon, and as with similar studies, is only as good as the lexicon. Additionally, we assume that each term is associated with a single sensorial modality. However, as research in psychology and neurology has shown, sensorial language is cross-modal. We leave this analysis to future work. In summary, we take a first step towards showing that sensorial style has a legitimate role in stylometrics research.

References

Marc Brysbaert, Michaël Stevens, Paweł Mandera, and Emmanuel Keuleers. 2016. How many words do we know? practical estimates of vocabulary size dependent on word definition, the degree of language input and the participant's age. *Frontiers in psychology*, 7:1116.

Emanuel Bubl, Elena Kern, Dieter Ebert, Michael Bach, and Ludger Tebartz Van Elst. 2010. Seeing gray when feeling blue? depression can be measured in the eye of the diseased. *Biological psychiatry*, 68(2):205–208.

Yin-Wong Cheung and Kon S Lai. 1995. Lag order and critical values of the augmented dickey–fuller test. *Journal of Business & Economic Statistics*, 13(3):277–280.

Louise Connell, Dermot Lynott, and Briony Banks. 2018. Interoception: the forgotten modality in perceptual grounding of abstract and concrete concepts. *Philosophical Transactions of the Royal Society B: Biological Sciences*, 373(1752):20170143.

Arthur D Craig. 2002. How do you feel? interoception: the sense of the physiological condition of the body. *Nature reviews neuroscience*, 3(8):655–666.

Ilja Croijmans, Laura J Speed, Artin Arshamian, and Asifa Majid. 2019. Measuring multisensory imagery of wine: The vividness of wine imagery questionnaire. *Multisensory research*, 32(3):179–195.

Stephen De Ullmann. 1945. Romanticism and synaesthesia: A comparative study of sense transfer in keats and byron. *PMLA*, 60(3):811–827.

David I Holmes. 1998. The evolution of stylometry in humanities scholarship. *Literary and linguistic computing*, 13(3):111–117.

David Howes. 2010. *Sensual relations: Engaging the senses in culture and social theory*. University of Michigan Press.

James M Hughes, Nicholas J Foti, David C Krakauer, and Daniel N Rockmore. 2012. Quantitative patterns of stylistic influence in the evolution of literature. *Proceedings of the National Academy of Sciences*, 109(20):7682–7686.

David Kernot, Terry Bossomaier, and Roger Bradbury. 2016. The impact of depression and apathy on sensory language. *Open Journal of Modern Linguistics*, 7(1):8–32.

Olga Koblet and Ross S Purves. 2020. From online texts to landscape character assessment: Collecting and analysing first-person landscape perception computationally. *Landscape and Urban Planning*, 197:103757.

Francesca Strik Lievers. 2015. Synaesthesia: A corpus-based study of cross-modal directionality. *Functions of language*, 22(1):69–95.

Francesca Strik Lievers and Chu-Ren Huang. 2016a. A lexicon of perception for the identification of synaesthetic metaphors in corpora. In *Proceedings of the Tenth International Conference on Language Resources and Evaluation (LREC'16)*, pages 2270–2277, Portorož, Slovenia. European Language Resources Association (ELRA).

Francesca Strik Lievers and Chu-Ren Huang. 2016b. A lexicon of perception for the identification of synaesthetic metaphors in corpora. In *Proceedings of the Tenth International Conference on Language Resources and Evaluation (LREC'16)*, pages 2270–2277.

Francesca Strik Lievers and Bodo Winter. 2018. Sensory language across lexical categories. *Lingua*, 204:45–61.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Roberta: A robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692*.

Andres Lou, Diana Inkpen, and Chris Tanasescu. 2015. Multilabel subject-based classification of poetry. In *The Twenty-Eighth International Flairs Conference*.

Dermot Lynott and Louise Connell. 2013. Modality exclusivity norms for 400 nouns: The relationship between perceptual experience and surface word form. *Behavior research methods*, 45(2):516–526.

Dermot Lynott, Louise Connell, Marc Brysbaert, James Brand, and James Carney. 2020. The lancaster sensorimotor norms: multidimensional measures of perceptual and action strength for 40,000 english words. *Behavior Research Methods*, 52(3):1271–1291.

Asifa Majid, Seán G Roberts, Ludy Cilissen, Karen Emmorey, Brenda Nicodemus, Lucinda O'grady, Bencie Woll, Barbara LeLan, Hilário De Sousa, Brian L Cansler, et al. 2018. Differential coding of perception in the world's languages. *Proceedings of the National Academy of Sciences*, 115(45):11369–11376.

Marius Mosbach, Stefania Degaetano-Ortlieb, Marie-Pauline Krielke, Badr M Abdullah, and Dietrich Klakow. 2020. A closer look at linguistic knowledge in masked language models: The case of relative clauses in american english. *arXiv preprint arXiv:2011.00960*.

Claire Murphy. 2019. Olfactory and other sensory impairments in alzheimer disease. *Nature Reviews Neurology*, 15(1):11–24.

Carita Paradis. 2003. Is the notion of linguistic competence relevant in cognitive linguistics. *Annual Review of Cognitive Linguistics*, pages 247–271.

Jamie Reilly, Maurice Flurie, and Jonathan E Peelle. 2020. The english lexicon mirrors functional brain activation for a sensory hierarchy dominated by vision and audition: Point-counterpoint. *Journal of neurolinguistics*, 55:100895.

Jacques Savoy. 2020. *Machine Learning Methods for Stylometry*. Springer.

Koustuv Sinha, Prasanna Parthasarathi, Joelle Pineau, and Adina Williams. 2020. Unnatural language inference. *arXiv preprint arXiv:2101.00010*.

Richard Sorabji. 1971. Aristotle on demarcating the five senses. *The Philosophical Review*, 80(1):55–79.

Laura J Speed and Asifa Majid. 2020. Grounding language in the neglected senses of touch, taste, and smell. *Cognitive neuropsychology*, 37(5-6):363–392.

Yla R Tausczik and James W Pennebaker. 2010. The psychological meaning of words: Liwc and computerized text analysis methods. *Journal of language and social psychology*, 29(1):24–54.

Joel Whitburn. 2010. *The Billboard Book of Top 40 Hits*. Billboard Book of Top 40 Hits.

Bodo Winter. 2016. Taste and smell words form an affectively loaded and emotionally flexible part of the english lexicon. *Language, Cognition and Neuroscience*, 31(8):975–988.

Bodo Winter. 2019. *Sensory linguistics: Language, perception and metaphor*, volume 20. John Benjamins Publishing Company.

Bodo Winter, Marcus Perlman, and Asifa Majid. Vision dominates in perceptual language: English sensory vocabulary is geared towards usage.

Bodo Winter, Marcus Perlman, and Asifa Majid. 2018. Vision dominates in perceptual language: English sensory vocabulary is optimized for usage. *Cognition*, 179:213–220.

A Distribution of Modalities

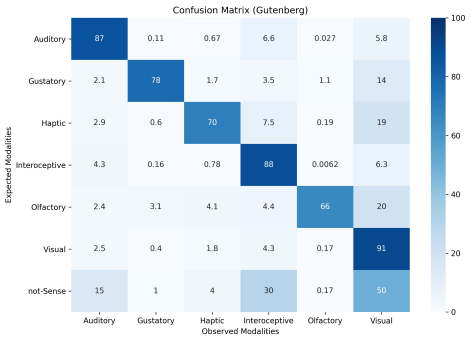

Figure 5: Distribution of expected-observed modalities in the Novels Dataset.

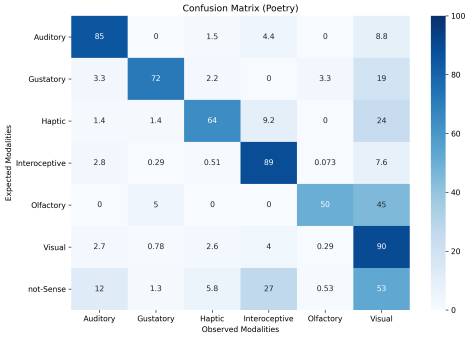

Figure 6: Distribution of expected-observed modalities in the Poetry Dataset.

Metaphorical Polysemy Detection: Conventional Metaphor meets Word Sense Disambiguation

Rowan Hall Maudslay[1,2] **Simone Teufel**[1]
[1]Dept. of Computer Science & Technology [2]Magdalene College
University of Cambridge
{rh635,sht25}@cam.ac.uk

Abstract

Linguists distinguish between novel and conventional metaphor, a distinction which the metaphor detection task in NLP does not take into account. Instead, metaphoricity is formulated as a property of a token in a sentence, regardless of metaphor type. In this paper, we investigate the limitations of treating conventional metaphors in this way, and advocate for an alternative which we name *metaphorical polysemy detection* (MPD). In MPD, only conventional metaphoricity is treated, and it is formulated as a property of word senses in a lexicon. We develop the first MPD model, which learns to identify conventional metaphors in the English WordNet. To train it, we present a novel training procedure that combines metaphor detection with *word sense disambiguation* (WSD). For evaluation, we manually annotate metaphor in two subsets of WordNet. Our model significantly outperforms a strong baseline based on a state-of-the-art metaphor detection model, attaining an ROC-AUC score of .78 (compared to .65) on one of the sets. Additionally, when paired with a WSD model, our approach outperforms a state-of-the-art metaphor detection model at identifying conventional metaphors in text (.659 F1 compared to .626).

1 Introduction

Linguists differentiate between two types of metaphor: novel and conventional. While novel metaphors are creative expressions made in a particular situation by one particular individual, conventional metaphors are those which have been widely adopted by a language community. Consider:

(1) The *attack* began at dawn.

(2) The government has come under *attack*.

(3) The government *torpedoed* the housing bill.

Example (1) is a literal usage of *attack* which refers to a military offensive. Example (2) is a conventional metaphor referring to intense verbal criticism.

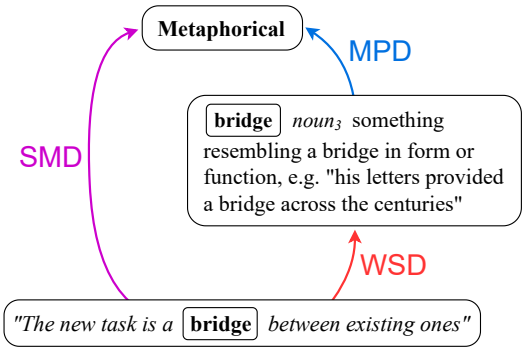

Figure 1: Metaphorical polysemy detection is the missing link between standard metaphor detection and word sense disambiguation

Example (3) is a novel metaphor, which reuses the well-established imagery from (2), but extends it with the word *torpedoed*.

Detecting novel metaphors and conventional metaphors are fundamentally different tasks. Novel metaphors are creative usages of words, which violate statistical patterns of language (e.g. Wilks, 1975, 1978), and which extend the meaning of a word into unexpected areas of semantic space. Conventional metaphors, on the other hand, can be considered lexicalised: examples (1) and (2) come from the gloss of two senses of *attack* in WordNet (Miller, 1995), where the novel sense of *torpedoed* in example (3) is not yet captured. Despite these inherent differences, in NLP the standard metaphor detection task, abbreviated as SMD here, makes no distinction between novel and conventional metaphor.

Although recent works have produced resources which distinguish between novel and conventional metaphors (Parde and Nielsen, 2018; Do Dinh et al., 2018), both metaphor types are annotated as properties of tokens, suitable for SMD. In this work, we demonstrate the shortcomings of this formulation, and argue that conventional metaphoricity is best treated not as a property of word occurrences in a sentence, but of word senses in a lexicon. With this

Proceedings of the 29th International Conference on Computational Linguistics, pages 65–77
October 12–17, 2022.

Sense	Definition	Metaphor?
$adopt_1$	take into one's family (e.g. *They adopted two children from Nicaragua*)	✗
$adopt_2$	choose and follow; as of theories, ideas, policies, strategies or plans	✓
$adopt_3$	take on a certain form, attribute, or aspect (e.g. *he adopted an air of superiority*)	✓
$adopt_4$	take up the cause, ideology, practice, method, of someone and use it as one's own (e.g. *They adopted the Jewish faith*)	✓

Table 1: Verbal senses of *adopted* in WordNet

in mind, we investigate the problem of assigning metaphoricity ratings to word senses, a problem we name metaphorical polysemy detection (MPD). We build the first model of MPD, which identifies metaphorical senses in WordNet (Miller, 1995).

No training data is available for this task. We design a novel training regime which utilises existing resources, which works by decomposing SMD into two steps: word sense disambiguation (WSD) and MPD (see Figure 1). More specifically, we pair our MPD model with a state-of-the-art WSD model, and train them in conjunction on SMD data, treating word sense as a latent variable.

To investigate the performance of our model, we establish an evaluation framework for MPD. To collect test data, we perform an annotation study, and label two subsets of WordNet for metaphoricity ($\kappa = 0.78$). Metaphor detection is typically evaluated using F1-score, which measures how well a model can judge metaphoricity in absolute terms. For MPD, we additionally introduce a new quantity for evaluation, which we call *relative metaphoricity*. It measures whether a model is able to correctly identify whether one sense is more metaphorical than another, even if it is unable to correctly determine where they sit around a threshold. It is calculated using ROC-AUC.

On one of our test sets, consisting of words from a large resource of conventional metaphors (the Master Metaphor List; Lakoff et al., 1991), our MPD model attains .78 ROC-AUC and .60 F1, significantly higher than a strong baseline which uses a state-of-the-art SMD model (MelBERT; Choi et al., 2021), which scores .65 and .54 respectively. Additionally, for SMD on conventional metaphors, when our model is paired with a WSD model it attains .659 F1, significantly better than the state-of-the-art SMD model (.626).

2 Metaphor in the Lexicon

A fundamental feature of language is *polysemy*, the phenomenon of a wordform exhibiting multiple meanings which are systematically related. When meanings are related by metaphorical similarity, the phenomenon is called *metaphorical polysemy*. An example is the word *attack*, which describes either a physical confrontation, or—metaphorically—heated criticism (e.g. "she *attacked* my arguments"). Instances of metaphorical polysemy are called *conventional metaphors*.

In this section, we describe how conventional metaphors materialise in lexica (§2.1). With this in mind, we illustrate the shortcomings of the existing metaphor detection task (§2.2), and propose an alternative formulation which is more suitable for conventional metaphors (§2.3).

2.1 Metaphoricity of Word Senses

In a lexicon, metaphoricity can be formulated as a property of word senses. Consider the example definitions of the word *adopted* in WordNet (Miller, 1995), which are shown in Table 1 (some senses are excluded for brevity). Sense $adopt_1$ is literal, but the other senses are conventional metaphors.

The conventionality of a metaphor lends itself to treatment as a continuous property. For instance, a metaphor used by an entire language community would be highly conventionalised, while a metaphor used by speakers in a particular geographic region would be moderately conventionalised. *Novel metaphors* sit at the extreme other end of the spectrum: they are creative expressions with ad-hoc meanings.[1] By definition, lexica de-

[1]Novel metaphors can be further subdivided into metaphors which are novel lexicalisations of conventional imagery, as with example (3) above, and metaphors which create new imagery.

signed to encode widespread language use will only encode the most conventional of metaphors as word senses.

In order to formulate metaphoricity as a property of word senses, a lexicon needs to encode the literal and metaphorical meanings of a wordform as separate senses. The criteria that need to be met for two meanings to be considered distinct is highly contentious, with some lexicographers arguing that senses should be represented as spectra rather than discrete units (see Cruse, 1986, Ch. 3). In practice, however, lexica describe senses as discrete objects. Moreover, in most lexica the granularity of these senses is high enough to separate metaphorical senses from literal ones, which thus matches our criteria. That being said, no existing lexicon systematically labels senses for metaphoricity (the decision of whether to label metaphorical senses as such is left to the lexicographer's discretion).

If word senses in a lexicon like WordNet were labeled for metaphoricity, this would be useful in a number of areas of study. When a language community need to reference a newly-arisen meaning, instead of creating a new wordform, often an existing word is extended metaphorically (e.g. a computer *mouse*). This process of metaphorical extension is a cornerstone of lexical-semantic language change (Koch, 2016), and knowing which word senses were metaphorical would open up new possibilities to study this phenomenon. The metaphors which do enter the lexicon participate in sophisticated patterns, preserving the structure of their literal domains: just as one can *attack* an opponent in a debate, so too can claims be *defended*, different rhetorical *strategies* adopted, and so on. These patterns are known as *conceptual metaphors* (Lakoff and Johnson, 1980), and knowing which word senses in a lexicon were metaphorical would create new opportunities to analyse their substructures (this has been previously recognised by Lönneker, 2003).

2.2 Issues with Standard Metaphor Detection

In the *standard metaphor detection* (SMD) task, metaphoricity is formulated as a property of a token in a sentence. The dataset which is almost universally used for this is the VUAMC (Steen et al., 2010). It contains tokenised English sentences, in which words are annotated with binary metaphoricity labels. It was the subject of two recent metaphor detection shared tasks (Leong et al., 2018, 2020).

The VUAMC was annotated following an adapted version of the Metaphor Annotation Procedure (MIP; Crisp et al., 2007). To annotate each token, MIP involves a three-stage process:

1. Establish the contextual meaning of the word, based on the other words in the sentence.

2. Determine whether the word has a more basic meaning which occurs in different contexts (where "basic" meanings are defined as those more concrete, related to bodily action, less vague, and/or historically older).

3. If the word does have a more basic meaning in different contexts, decide whether the meaning in this context can be understood in comparison with the more basic meaning.

If the wordform does have a more basic meaning which the contextual meaning relates to, the token is labeled as a metaphor.

To perform well in SMD, a model must emulate this procedure. There are two drawbacks to this task formulation:

Learning WSD Implicitly For conventional metaphors, stage 1 of MIP is very similar to *word sense disambiguation* (WSD). In WSD, the goal is to identify the sense evoked by a wordform in a sentence. This is a hard task in its own right, and, in order to perform well, models of SMD are expected to learn to do this implicitly.

We can check whether an SMD model has learnt to do this by analysing its predictions for wordforms which appear in multiple distinct contexts which all evoke the same sense. Tokens which evoke the same sense should always receive the same metaphoricity prediction (either they should all be metaphorical, or all literal). This is because, for conventional metaphors, metaphoricity is a property of a word sense (§2.1). After the sense of a token is established, stages 2 and 3 of MIP should always yield the same prediction; if a model makes inconsistent predictions for different invocations of the same sense, that suggests that it is incorrectly establishing the contextual meaning of the token.

To investigate whether a state-of-the-art model is able to adequately perform stage 1 of MIP, we perform an error analysis of MelBERT (Choi et al., 2021). More specifically, we compute MelBERT's metaphoricity predictions for every token in an English sense-tagged corpus (SemCor; Miller et al.,

Sentence Excerpt	Prediction
*...the South Carolina nullifiers **adopted** the principle of state interposition...*	Metaphorical
*...the methods he **adopted** to accomplish...*	Metaphorical
*...the denomination's 16 basic beliefs **adopted** in 1966...*	Metaphorical
*...the Government's new feed grain program was **adopted**; the program...*	Literal
*...the Albany Plan of Union, which, had it been **adopted**, might...*	Literal
*...the use of target-hunting noses on the projectiles has been **adopted**, and...*	Literal

Table 2: MelBERT makes inconsistent predictions for occurrences of *adopted* with sense $adopt_2$ in SemCor

1994), then extract all the predictions of the same word sense, which should always elicit a consistent prediction. Table 2 shows MelBERT's predictions for different instances of the word *adopted* which all evoke the same metaphorical sense, $adopt_2$.[2] MelBERT misclassifies the bottom three examples as literal. In general, for word senses which occur multiple times in the data, MelBERT gave contradictory classifications 31.8% of the time. It becomes more likely that a system will make an inconsistent classification when there are more occurrences of a word sense in the data: for word senses which occurred 15 or more times, the misclassification rate was 49.3%.

This error analysis suggests that MelBERT is not performing stage 1 of MIP adequately. The errors in Table 2 result not from a failure to reason about metaphoricity, but from MelBERT incorrectly establishing the contextual meaning of *adopted* in the first place. We argue that this is beyond the remit of metaphor detection, since it is a hard challenge in its own right, covered by other NLP tasks.

Word's Senses Undefined After stage 1 of MIP has been performed, stages 2 and 3 require information about how the contextual meaning of a wordform relates to the other senses it may take. More specifically, models need to determine the most basic sense of the wordform, and relate that sense to the contextual meaning. However, the VUAMC does not provide information about other meanings a wordform could assume. Because of this, model architects typically assume that static word embeddings will represent the basic sense of a wordform (e.g. Choi et al., 2021; Mao et al., 2018; Zayed et al., 2018; Wu et al., 2018). We argue that this assumption is problematic, because static word embeddings do not distinguish between word senses, and are sensitive to occurrence frequency. If one sense is much more frequent than others, static word embeddings will primarily encode that sense—but the basic sense of a wordform is often not the most frequent.

Fortunately, we know that the most basic sense of a wordform will be one of its senses in a lexicon. If we model metaphoricity as a property of word senses, we create scope for researchers to explicitly investigate the relations between different senses, which we argue is the core challenge posed by metaphor.

2.3 A New Task Formulation for Conventional Metaphor

Let $w \in \mathcal{W}$ be all the wordforms in a language. To represent metaphoricity, let $\mathcal{M}$ be the set $\{0, 1\}$ with members m, such that 1 represents metaphorical and 0 represents literal.

A token $t \in \mathcal{T}$ is an occurrence of a wordform in a sentence. Using $\circ$ to denote concatenation, we define a token as a wordform w surrounded by a prefix and suffix string ($\mathbf{p}$ and $\mathbf{s}$) in a sentence:

$$\mathcal{T} = \{\langle \mathbf{p}, w, \mathbf{s} \rangle \mid \mathbf{p} \circ w \circ \mathbf{s} \in \mathcal{W}^*\} \quad (1)$$

In SMD, the goal is to determine whether a token is metaphorical, i.e. to construct a model of $p(m \mid t)$.

A word sense is a particular meaning of a word, which are typically represented in lexica as definitions. Let $\mathcal{D}$ be the complete set of definitions in a lexicon (in WordNet these correspond to synsets), with elements d. A lexicon maps a wordform w to a subset of the definitions $\mathcal{D}_w \subseteq \mathcal{D}$.[3] Some lexica (WordNet included) associate multiple synonymous wordforms with the same definition; because of this, we define a word sense $s \in \mathcal{S}$ as a tuple consisting of a wordform and a definition:

$$\mathcal{S} = \{\langle w, d \rangle \mid w \in \mathcal{W}, d \in \mathcal{D}_w\} \quad (2)$$

[2] Data here comes from our re-implementation of MelBERT trained on the 'all' subset, described in §4.3.

[3] Formally, if $\mathbb{P}(x)$ denotes the powerset of x, then a lexicon is a map $\mathcal{W} \to \mathbb{P}(\mathcal{D})$.

Because conventional metaphors are lexicalised, we propose that the task of detecting conventional metaphors should be formulated as modelling the metaphoricity of word senses, i.e. constructing models of $p(m \mid s)$. We call this *metaphorical polysemy detection* (MPD). This formulation of conventional metaphoricity alleviates the problems with SMD outlined in §2.2.

3 Learning Conventional Metaphor

In this section, we describe the first MPD model (§3.1). To train it, it is embedded into an SMD pipeline (§3.2).

3.1 An Architecture for MPD

To model MPD (see §2.3), we implement $p(m \mid s)$ as a multi-layer perceptron (MLP). In eq. (2), we defined a word sense s as a tuple consisting of a wordform w and a definition d. Our MPD model's MLP is input with a pair of embeddings, for w and d. Suppose we have a k-dimensional embeddings space for words $w \in \mathcal{W}$, and a k-dimensional embedding space for definitions $d \in \mathcal{D}$. Let us define two functions to map inputs into these spaces, $\texttt{TypeEmb} : \mathcal{W} \to \mathbb{R}^k$ and $\texttt{SynsetEmb} : \mathcal{D} \to \mathbb{R}^k$.[4] We concatenate these embeddings, then pass them through an MLP and a sigmoid to get a probability distribution. Letting $\texttt{MLP}_{\boldsymbol{\psi}}^{\langle u,v \rangle} : \mathbb{R}^u \to \mathbb{R}^v$ be an MLP parameterised by $\boldsymbol{\psi}$ with an input size u and output size v, we have

$$\begin{aligned} \mathbf{h} &= \texttt{TypeEmb}(w) \circ \texttt{SynsetEmb}(d) \\ p_{\boldsymbol{\theta}}(m \mid s) &= \sigma\left(\texttt{MLP}_{\boldsymbol{\theta}}^{\langle 2k,1 \rangle}(\mathbf{h})\right) \end{aligned} \tag{3}$$

3.2 Teasing MPD out of SMD using WSD

Training this model would be trivial if we had $\langle m, s \rangle$ tuples, but no training data of this form currently exists. In WSD, the task is to produce models of $p(s \mid t)$ for a corpus $\langle s, t \rangle \in \mathcal{C}_{\text{WSD}}$. In SMD, on the other hand, the task is to produce models of $p(m \mid t)$, using a corpus $\langle m, t \rangle \in \mathcal{C}_{\text{SMD}}$. No corpus contains both m and s annotations in parallel. In this section, we describe a training procedure that works by pairing an MPD model with a WSD model, and training them in conjunction on SMD, thus making it possible to use existing resources of $\mathcal{C}_{\text{WSD}}$ and $\mathcal{C}_{\text{SMD}}$ to train MPD. We do not believe that this procedure is an optimal way to learn MPD, as errors in the WSD component will degrade MPD performance. However, given that these are the only resources currently available, we design a method to use them for MPD, and in doing so empirically test whether they can result in a sufficiently good MPD model.

By marginalising out word sense, we can decompose SMD into two parts, introducing sense s as a latent variable. The first part disambiguates senses (WSD), and the second predicts metaphoricity based on the sense (MPD):

$$\underbrace{p_{\langle \boldsymbol{\theta}, \boldsymbol{\phi} \rangle}(m \mid t)}_{\text{SMD}} = \sum_{s \in \mathcal{S}} \underbrace{p_{\boldsymbol{\theta}}(m \mid s)}_{\text{MPD}} \underbrace{p_{\boldsymbol{\phi}}(s \mid t)}_{\text{WSD}} \tag{4}$$

where $\boldsymbol{\theta}$ and $\boldsymbol{\phi}$ are disjoint sets of parameters (colours correspond with Figure 1). Introducing WSD into SMD like this makes explicit what SMD models typically attempt to learn implicitly (see §2.2). The WSD component outputs a probability distribution over the senses $s \in \mathcal{S}$, and the MPD component outputs the probability that each s is metaphorical.[5] This formulation assumes that metaphoricity is conditionally independent of token given sense; we assume that the context of a word's usage tells us its sense, and that that alone is enough to predict metaphoricity.

Taken as a whole, the combined model in eq. (4) can be trained end-to-end by minimising its cross-entropy on an SMD dataset,

$$\mathcal{L}_{\text{SMD}}(\boldsymbol{\theta}, \boldsymbol{\phi}) = -\sum_{\substack{\langle m,t \rangle \\ \in \mathcal{C}_{\text{SMD}}}} \frac{\log p_{\langle \boldsymbol{\theta}, \boldsymbol{\phi} \rangle}(m \mid t)}{|\mathcal{C}_{\text{SMD}}|} \tag{5}$$

However, this approach to training would leave the model to infer s implicitly. Instead, we can complement this objective with another, which trains the WSD model in isolation on another dataset ($\mathcal{C}_{\text{WSD}}$), also using cross-entropy,

$$\mathcal{L}_{\text{WSD}}(\boldsymbol{\phi}) = -\sum_{\substack{\langle s,t \rangle \\ \in \mathcal{C}_{\text{WSD}}}} \frac{\log p_{\boldsymbol{\phi}}(s \mid t)}{|\mathcal{C}_{\text{WSD}}|} \tag{6}$$

These two objectives can be combined into a multi-task objective, using α as a hyperparameter which regulates the ratio between them, yielding the final global loss function

$$\begin{aligned} \mathcal{L}(\boldsymbol{\theta}, \boldsymbol{\phi}) = \quad & \alpha \quad \cdot \mathcal{L}_{\text{SMD}}(\boldsymbol{\theta}, \boldsymbol{\phi}) + \\ & (1 - \alpha) \cdot \mathcal{L}_{\text{WSD}}(\boldsymbol{\phi}) \end{aligned} \tag{7}$$

[4] For $\texttt{SynsetEmb}$ we use ARES embeddings (Scarlini et al., 2020), and for $\texttt{TypeEmb}$ we use BERT embeddings of wordforms (following Choi et al., 2021); see App. D.

[5] For a particular token $t = \langle \mathbf{p}, w, \mathbf{s} \rangle$ and sense $s = \langle w', d \rangle$, the probability $p(s \mid t)$ will only be non-zero if $w = w'$; in practice, this can be used to reduce the computation.

Any supervised WSD architecture can be used in conjunction with our MPD model.

4 Evaluation

In this section, we describe the evaluation data we collected for MPD (§4.1), and introduce relative metaphoricity, an aspect of metaphor which it is only possible to evaluate with MPD (§4.2). We then describe our experimental setup (§4.3) and results (§4.4).

4.1 Evaluation Data for MPD

Our approach to training (§3.2) alleviates the need for labeled MPD training data. However, we still need evaluation data. For this we collect annotation. In this section, we describe the main evaluation set we collected. This evaluation set consists of data from the Master Metaphor List, which is the largest list of conventional metaphors (collated by Lakoff et al., 1991).

Data for Annotation We sample 250 commonly cited metaphor examples from the MML, filtering out wordforms which have more than 10 or fewer than 3 senses. We annotate the remaining 98 wordforms for metaphoricity (a total of 554 word senses).

Annotation Procedure We adapt MIP stages 2 and 3 for the annotation task (our guidelines are presented in App. A). For MPD, we require a lexicon which consistently separates literal and metaphorical senses (see §2.1). We use the annotation phase to also check whether WordNet complies with this. Along with options for "metaphorical" and "literal", annotators also have the option to label senses as "mixed", meaning that the sense combined literal and metaphorical definitions. Our data is annotated by two judges, who are native speakers of English.

Analysis At least one of the annotators labeled a sense as "mixed" 5.4% of the time. We conclude from this that although WordNet mixes some literal and metaphorical senses, for the most part the granularity of the sense inventory was high enough to separate them. Of those which neither thought was mixed, inter-annotator agreement was measured at $\kappa = 0.78$ (N=539, n=2, k=2): where the senses were clearly distinct, annotation was consistent. Of the senses that the judges agreed upon, 52% were metaphorical and 48% were literal.

4.2 Relative Metaphoricity

Metaphor detection performance is typically reported using F1-score, which factors in the precision and recall of a classifier. This evaluates a model's ability to judge whether the metaphoricity level of an input passes a threshold to be considered metaphorical, in an absolute sense.

With absolute metaphoricity judgments, there are lots of edge cases, which do not cleanly fit into a metaphorical–literal binary. For example, while a physical *attack* (e.g. an outlaw with a knife) is widely accepted as being literal, and an *attack* with words (e.g. in the perpetrator's court hearing) is widely accepted as being metaphorical, the case is less clear-cut for a sporting *attack* (e.g. when they are playing football with other inmates): here we have a physical act which can result in injury, but that takes place within the confines of a game, and certainly does not involve weapons.[6] It is difficult to decide whether an edge case like this is metaphorical or not, if those are the only options. Most people would likely agree, however, that a sporting attack is *more* metaphorical than an attack with a knife, and *less* metaphorical than an attack within a debate.

We are not the first to note these flaws in absolute metaphoricity judgements; they have previously been used to motivate proposals for the treatment of metaphor on graded or continuous scales (e.g. Dunn, 2014; Mohler et al., 2016). Even though MPD is a binary classification task, it additionally gives us the opportunity to evaluate whether a model can judge which of a word's senses are more metaphorical than others, even if it gets the exact threshold wrong. We call this a measure of *relative metaphoricity*.

Beneath the surface, models assign a probability of metaphoricity, rather than an absolute judgement. With MPD, we can compute a model's predictions for all the senses of a word, and investigate whether it puts these probabilities in the right order, even if it gets absolute predictions "wrong" in an absolute sense. A metric for this is described in §4.3.

4.3 Experimental Setup

In this section, we describe the models and data we experimented with. Additional details can be found in the appendices.

[6] *Ad extremum*, this line of reasoning might lead us to ask whether a sporting *attack* in rugby (or even boxing) would be more literal than it would be in non-contact sports.

MPD Models We compare our MPD model (**Ours**) to several baselines. To establish a lower-bound of performance for MPD, we compute two baselines which make predictions randomly or by choosing the most common class (**Random** and **Majority**). In addition to these, we compute a strong baseline: using a state-of-the-art SMD model (MelBERT; Choi et al., 2021), we compute the metaphoricity probability of all occurrences of s in WSD data, and take the average (**MelBERT Average**). More formally, let $t \in \mathcal{C}^s_{\text{WSD}}$ be every token in the WSD data which elicits a specific word sense s. We compute the average metaphoricity probability of that word sense by taking the mean of MelBERT's metaphoricity prediction for all tokens that evoke that sense:

$$p(m \mid s) = \underset{t \in \mathcal{C}^s_{\text{WSD}}}{\text{mean}} \underbrace{p(m \mid t)}_{\text{MelBERT}} \tag{8}$$

If $|\mathcal{C}^s_{\text{WSD}}| = 0$ (i.e. there are no occurrences of s in the WSD data) then we default to the random baseline (and take the probability as random).

WSD Models We experiment with two WSD models: a baseline (**BERT WSD Baseline**) and a state-of-the-art approach (**EWISER**; Bevilacqua and Navigli, 2020). Additional description is left to App. B.

SMD Models We train the combined model (**WSD+MPD**) from eq. (4), using our MPD model and both WSD models (we select the highest performing variant). We compare the overall SMD performance of this combined model to two other approaches, a baseline (**BERT SMD Baseline**) and a high-performing model (**MelBERT**). Additional description is left to App. C.

MPD Data To evaluate MPD performance, we use the evaluation set described in §4.1 (**MML**). We also collected a second set using the same procedure, designed to evaluate model generalisation ability to words which are out-of-vocabulary of the training data (**OOV**). We randomly sampled 100 wordform from the WordNet vocabulary which are not included in the vocabularies of $\mathcal{C}_{\text{SMD}}$ or $\mathcal{C}_{\text{WSD}}$, and asked one of the annotators from §4.1 to follow the same procedure as before. The final evaluation sets for MML and OOV consist of 535 and 492 word senses respectively, after senses which the annotator labeled as "mixed" were removed. As an additional third set, we use the data of Mohammad et al. (2016), who annotated verbs in WordNet for metaphoricity (**Verbs**). Their data consists of 1,679 word sense annotations, covering 440 verbal wordforms.

WSD Data For $\mathcal{C}_{\text{WSD}}$, we use SemCor (Miller et al., 1994). In SemCor each token is annotated with the WordNet sense it evokes. We remove all datapoints with trivial solutions (for which there is only a single sense to choose from, i.e. $|\mathcal{D}_w| = 1$).

SMD Data For $\mathcal{C}_{\text{SMD}}$, we use the VUAMC (see §2.2). We remove all part-of-speech types and words not in WordNet (meaning we remove all prepositions, which are almost always labeled as metaphorical in VUAMC, and so are easy to predict). Additionally, because our focus is on conventional metaphors, when investigating SMD performance, we also experiment using a subset of this data in which novel metaphors are filtered out (we refer to this subset as **Conventional**, as opposed to **All**). We achieve this using the annotation layer of Do Dinh et al. (2018).[7]

Metrics To evaluate MPD for absolute metaphoricity, we compute the F1-score. For relative metaphoricity, we compute (for each wordform individually) the area under a receiver operating characteristic curve (ROC-AUC, see Fawcett, 2004 for discussion). An ROC curve is a plot of the true positive rate against the false positive rate, as a threshold shifts from 0 to 1. This tells us whether the metaphoricity probabilities assigned to the senses of a wordform are in correct high–low ordering, even if they are not properly calibrated around the .5 threshold. ROC-AUC values range from 1 (perfectly correct) to 0 (perfectly incorrect), where .5 indicates no correlation. To get a value for a whole evaluation set, we take the mean of the ROC-AUC scores of all wordforms in the set. To evaluate SMD, we compute the F1-score, and for WSD, where the goal is multi-class classification, we compute the micro-averaged F1-score.

Significance Testing We use a two-tailed Monte Carlo permutation test with $r = 1{,}000$ permutations. We experiment with two significance levels, $\alpha = 0.05$ and 0.01, and differentiate between these by indicating the significance level using * and ** respectively.

[7] In this data, metaphor novelty is scored with continuous values in the $[-1, 1]$ interval; as a threshold, we use 0.2.

MPD Model		Relative (ROC-AUC)			Absolute (F1)		
		MML	*OOV*	*Verbs*	*MML*	*OOV*	*Verbs*
Ours	(with EWISER WSD)	**.78****	**.64***	.70	.54	.37	.43
	(with BERT WSD Baseline)	.72	.60	**.71****	**.60***	**.41***	**.47****
MelBERT Average		.65	.50	.54	.54	.33	.37
Random		.54	.51	.51	.49	.33	.35
Majority		.50	.50	.50	.00	.00	.00

Table 3: Metaphorical polysemy detection results

SMD Model	F1	
	Conventional	*All*
Ours (WSD+MPD)	**.659****	.631
MelBERT	.626	.638
BERT SMD Baseline	.619	.625

Table 4: Standard metaphor detection results

WSD Model	Objective	F1
EWISER	WSD	.768
	WSD & SMD	.766
BERT WSD Baseline	WSD	.740
	WSD & SMD	.741

Table 5: Word sense disambiguation results

4.4 Results

Metaphorical Polysemy Detection MPD results are presented in Table 3. We compare two versions of our MPD model, trained in combination with different WSD implementations. In all experimental settings (each column), a variant of our model performs significantly better than the highest-performing baseline. Our model's results are highest for the MML set (.78 ROC-AUC and .60 F1), perhaps because the examples here exhibited the clearest cases of metaphoricity. In general, the EWISER WSD variant's results are higher for relative metaphoricity, whilst the baseline WSD variant's results are higher for absolute metaphoricity. The reason for this is unclear. Results are lowest for the OOV set, which shows that the model has difficulties generalising beyond training data. Generalising to never-before-seen senses is a well-known issue in WSD: Bevilacqua and Navigli themselves note that their model, EWISER, relies too much on corpus supervision. This problem will affect MPD models if they are trained with our methodology (§3.2); to improve on MPD, it will be necessary to collect dedicated training data. Nevertheless, our results show that it is possible to learn MPD using existing resources.

Standard Metaphor Detection SMD results are presented in Table 4, for both variants of $\mathcal{C}_{\text{SMD}}$ (one with only conventional metaphors, one with any). When trained and evaluated on only conventional metaphors, our combined MPD+WSD model significantly outperforms the state-of-the-art, scoring .659 F1 (compared to MelBERT's .626). When all metaphors are included, MelBERT's results are higher that our model (.638 compared to .631), but in this case the difference is insignificant (at both significance levels). That our model's performance is diminished in this setting is unsurprising, since this data will contain novel metaphors which are not encoded as senses in WordNet. We expect that with improvements to WSD generalisation, it should be possible to improve the SMD performance of a combined MPD+WSD model, for both variants. As mentioned above, generalising to unseen senses is a common issue in WSD. Only 9.4% of words in the SMD (All) test set have all their senses covered in the WSD training data, which is likely to have a substantial effect on the combined model's performance.

Word Sense Disambiguation Table 5 compares the performance of WSD models which are paired with MPD models and trained jointly on SMD, as opposed to trained on WSD alone. Although the auxiliary SMD objective leads marginal numerical differences (-0.02 and $+0.01$ for EWISER and the baseline WSD model respectively), in both cases these differences are not significantly distinguishable (at either significance level), suggesting that utilising SMD data to train WSD models is not useful. This is likely because of the relatively small overlap between the vocabularies of the datasets.

5 Conclusion and Future Work

In this paper, we argued that the standard metaphor detection (SMD) task in NLP is ill-suited to conventional metaphor, because it conflates metaphor detection with word sense disambiguation (WSD). As an alternative, we proposed metaphorical polysemy detection (MPD). We constructed the first MPD model, which identifies conventional metaphors in WordNet. To train it, we employed a novel training technique which exploits resources designed for SMD and WSD. To evaluate MPD, we collected two sets of evaluation data, and proposed a new performance measure based on relative judgements of metaphoricity. Our model performed significantly better than a state-of-the-art SMD model in all MPD evaluation settings (e.g. attaining .78 ROC-AUC and .60 F1 on a subset based on the master metaphor list). Additionally, we found that pairing our model with a WSD model led to state-of-the-art results for token-based conventional metaphor detection (.659 compared to .626). We make our code and MPD evaluation data publicly available.[8]

Making serious improvements in MPD will require the collection of dedicated training data. Training MPD using SMD data necessitated the inclusion of a WSD model, which will negatively affect MPD performance when it makes mistakes. Additionally, our model did not utilise information about the set of definitions a wordform is associated with ($\mathcal{D}_w$), and instead made the naïve assumption that this information will be encoded in a static embedding of w. Set- and graph-based architectures which exploited this information would be a natural thing to explore. With more data and better models, it may be possible to synthesise a full set of WordNet metaphoricity annotation, and even to extend this synthetic annotation to multilingual WordNet versions. This would be a valuable resource, which would open the door to study other questions surrounding metaphorical polysemy.

Acknowledgements

The first author would like to acknowledge the help of Laurence Midgley and Ryan Singh, for conversations about conditionality, and would also like to gratefully acknowledge Magdalene College and the Cambridge Trust, who funded him for the duration of this work.

[8] https://github.com/rowanhm/metaphorical-polysemy-detection

References

Michele Bevilacqua and Roberto Navigli. 2020. Breaking through the 80% glass ceiling: Raising the state of the art in word sense disambiguation by incorporating knowledge graph information. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 2854–2864, Online. Association for Computational Linguistics.

Minjin Choi, Sunkyung Lee, Eunseong Choi, Heesoo Park, Junhyuk Lee, Dongwon Lee, and Jongwuk Lee. 2021. MelBERT: Metaphor detection via contextualized late interaction using metaphorical identification theories. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 1763–1773, Online. Association for Computational Linguistics.

Peter Crisp, Raymond Gibbs, Alice Deignan, Low, Graham, Gerard Steen, Cameron, Lynne, Elena Semino, Joe Grady, Alan Cienki, and Zoltan Kövecses. 2007. MIP: A method for identifying metaphorically used words in discourse. *Metaphor and Symbol*, 22(1):1–39.

David A. Cruse. 1986. *Lexical Semantics*. Cambridge University Press.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, Minneapolis, Minnesota. Association for Computational Linguistics.

Erik-Lân Do Dinh, Hannah Wieland, and Iryna Gurevych. 2018. Weeding out conventionalized metaphors: A corpus of novel metaphor annotations. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 1412–1424, Brussels, Belgium. Association for Computational Linguistics.

Jonathan Dunn. 2014. Measuring metaphoricity. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 745–751, Baltimore, Maryland. Association for Computational Linguistics.

Tom Fawcett. 2004. ROC graphs: Notes and practical considerations for researchers. *Machine learning*, 31(1):1–38.

Peter Koch. 2016. Meaning change and semantic shifts. In Päivi Juvonen and Maria Koptjevskaja-Tamm, editors, *The Lexical Typology of Semantic Shifts*, chapter 2, pages 21–66. De Gruyter Mouton.

George Lakoff, Jane Espenson, and Alan Schwartz. 1991. *Master Metaphor List (Second Draft)*. University of California.

George Lakoff and Mark Johnson. 1980. *Metaphors We Live By*. University of Chicago Press.

Chee Wee (Ben) Leong, Beata Beigman Klebanov, Chris Hamill, Egon Stemle, Rutuja Ubale, and Xianyang Chen. 2020. A report on the 2020 VUA and TOEFL metaphor detection shared task. In *Proceedings of the Second Workshop on Figurative Language Processing*, pages 18–29, Online. Association for Computational Linguistics.

Chee Wee (Ben) Leong, Beata Beigman Klebanov, and Ekaterina Shutova. 2018. A report on the 2018 VUA metaphor detection shared task. In *Proceedings of the Workshop on Figurative Language Processing*, pages 56–66, New Orleans, Louisiana. Association for Computational Linguistics.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Roberta: A robustly optimized BERT pretraining approach. *CoRR*, abs/1907.11692.

Birte Lönneker. 2003. Is there a way to represent metaphors in WordNets? Insights from the Hamburg Metaphor Database. In *Proceedings of the ACL 2003 Workshop on the Lexicon and Figurative Language*, pages 18–27, Sapporo, Japan. Association for Computational Linguistics.

Ilya Loshchilov and Frank Hutter. 2017. Fixing weight decay regularization in Adam. *CoRR*, abs/1711.05101.

Rui Mao, Chenghua Lin, and Frank Guerin. 2018. Word embedding and WordNet based metaphor identification and interpretation. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1222–1231, Melbourne, Australia. Association for Computational Linguistics.

George A. Miller. 1995. WordNet: A lexical database for English. *Commun. ACM*, 38(11):39–41.

George A. Miller, Martin Chodorow, Shari Landes, Claudia Leacock, and Robert G. Thomas. 1994. Using a semantic concordance for sense identification. In *Proceedings of the Workshop on Human Language Technology*, HLT '94, page 240–243, USA. Association for Computational Linguistics.

Saif Mohammad, Ekaterina Shutova, and Peter Turney. 2016. Metaphor as a medium for emotion: An empirical study. In *Proceedings of the Fifth Joint Conference on Lexical and Computational Semantics*, pages 23–33, Berlin, Germany. Association for Computational Linguistics.

Michael Mohler, Mary Brunson, Bryan Rink, and Marc Tomlinson. 2016. Introducing the LCC metaphor datasets. In *Proceedings of the Tenth International Conference on Language Resources and Evaluation (LREC'16)*, pages 4221–4227, Portorož, Slovenia. European Language Resources Association (ELRA).

Natalie Parde and Rodney Nielsen. 2018. A corpus of metaphor novelty scores for syntactically-related word pairs. In *Proceedings of the Eleventh International Conference on Language Resources and Evaluation (LREC 2018)*, Miyazaki, Japan. European Language Resources Association (ELRA).

Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Kopf, Edward Yang, Zachary DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. 2019. PyTorch: An imperative style, high-performance deep learning library. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett, editors, *Advances in Neural Information Processing Systems 32*, pages 8024–8035. Curran Associates, Inc.

Fabian Pedregosa, Gaël Varoquaux, Alexandre Gramfort, Vincent Michel, Bertrand Thirion, Olivier Grisel, Mathieu Blondel, Peter Prettenhofer, Ron Weiss, Vincent Dubourg, Jake Vanderplas, Alexandre Passos, David Cournapeau, Matthieu Brucher, Matthieu Perrot, and Édouard Duchesnay. 2011. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12(85):2825–2830.

Prajit Ramachandran, Barret Zoph, and Quoc V. Le. 2017. Searching for activation functions. *CoRR*, abs/1710.05941.

Bianca Scarlini, Tommaso Pasini, and Roberto Navigli. 2020. With more contexts comes better performance: Contextualized sense embeddings for all-round word sense disambiguation. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 3528–3539, Online. Association for Computational Linguistics.

Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. 2014. Dropout: A simple way to prevent neural networks from overfitting. *Journal of Machine Learning Research*, 15(56):1929–1958.

Gerard J. Steen, Aletta G. Dorst, J. Berenike Herrmann, Anna Kaal, Tina Krennmayr, and Trijntje Pasma. 2010. *A Method for Linguistic Metaphor Identification: From MIP to MIPVU*. John Benjamins.

Yorick Wilks. 1975. A preferential, pattern-seeking, semantics for natural language inference. *Artificial Intelligence*, 6(1):53–74.

Yorick Wilks. 1978. Making preferences more active. *Artificial Intelligence*, 11(3):197–223.

Chuhan Wu, Fangzhao Wu, Yubo Chen, Sixing Wu, Zhigang Yuan, and Yongfeng Huang. 2018. Neural metaphor detecting with CNN-LSTM model. In *Proceedings of the Workshop on Figurative Language Processing*, pages 110–114, New Orleans, Louisiana. Association for Computational Linguistics.

Omnia Zayed, John Philip McCrae, and Paul Buitelaar. 2018. Phrase-level metaphor identification using distributed representations of word meaning. In *Proceedings of the Workshop on Figurative Language Processing*, pages 81–90, New Orleans, Louisiana. Association for Computational Linguistics.

A Annotation Guidelines

*We are studying **conventional metaphors**. A conventional metaphor is a metaphorical usage of a word that is common enough to appear in dictionaries. Consider the two definitions below for the English verb "flood".*

1. *cover with liquid, usually water (e.g. "the swollen river flooded the village")*
2. *supply with an excess of (e.g. "flood the market with tennis shoes")*

The first is a literal meaning, while the second is a conventional metaphor (in the example sentence, the market is not literally being flooded). Words often have more than two meanings, and can have multiple literal meanings and/or multiple metaphorical meanings.

Your task is to identify which definitions are conventional metaphors. To annotate a word:

1. *Read all of the definitions the word is associated with. Using the definitions (and the synonyms and example sentences), try and get a feel for the meaning of each definition and how they are different.*
2. *Then, for each of the word's definitions:*
   (a) *Determine if another of the word's definitions is more "basic" than the current one. Basic definitions tend to be*
      - *more concrete (they describe things which are easier to see, hear, feel, smell, and/or taste);*
      - *related to the body or the physical world;*
      - *more precise (less vague);*
      - *historically older*

      Basic meanings are not necessarily the first definitions of a word in the list, and are not necessarily the most frequent meanings of a word.
   (b) *If the word has a more basic definition than the current one, decide whether the current definition can be understood as an extension of a more basic one.*

If a definition can be understood as an extension of a more basic one belonging to the word, label it as "metaphorical". Otherwise, label it as "literal". There is one exception:

Sometimes, the definitions can be ambiguous, and combine metaphorical and literal meanings. Sometimes the definition is explicitly ambiguous (it might say that it is meant "metaphorically or literally", or that it can apply to something "concrete or abstract"), and other times it is just vague. If the definition is vague, then you should attempt to resolve the ambiguity by looking at the example sentences. For example, if all the examples are clearly metaphorical, say that this definition is metaphorical. If it cannot be disambiguated, you should select "mixed".

B Additional WSD Models

WSD Baseline Suppose we have another k-dimensional embedding space for tokens, retrieved by $\texttt{TokenEmb} : \mathcal{T} \mapsto \mathbb{R}^k$.[9] As a baseline WSD model, we predict a distribution over $\mathcal{S}$ from $\texttt{TokenEmb}(t)$:

$$\mathbf{h} = \texttt{TokenEmb}(t) \tag{9}$$
$$p_{\boldsymbol{\phi}}(s \mid t) = \texttt{softmax}\big(\texttt{MLP}_{\boldsymbol{\phi}}^{\langle k, |\mathcal{S}| \rangle}(\mathbf{h})\big)$$

In practice, we renormalise the output distribution so only senses with $d \in \mathcal{D}_w$ have nonzero probabilities.

EWISER For a high-performing WSD model, we experiment with a reimplementation of EWISER (Bevilacqua and Navigli, 2020). Where O is a matrix where the ith column corresponds to a b-dimensional embedding of $d_i \in \mathcal{D}$, $\texttt{SynsetEmb}(d_i)$, and A is an adjacency matrix of size $|\mathcal{D}| \times |\mathcal{D}|$, where 1 indicates that two synsets are connected, and 0 indicates they are disconnected, EWISER is defined as

$$\mathbf{h} = \texttt{MLP}_{\boldsymbol{\phi}}^{\langle k, k \rangle}\big(\texttt{TokenEmb}(t)\big) \tag{10}$$
$$\mathbf{z} = \mathbf{h}^T O$$
$$p_{\boldsymbol{\phi}}(m \mid t) = \sigma\big(\mathbf{z} A^T + \mathbf{z}\big)$$

In the original paper, a linear layer rather than an MLP is used, and many different experimental settings are explored; we only experiment with the setting in which S is initialised with only hypernyms set to 1, and O and S are kept frozen. For

[9] For $\texttt{TokenEmb}$ we use BERT embeddings; see App. D.

the definition representations and token representations, we use the same ones as the other models (see App. D), and use the standard activation functions we use throughout, rather than the Swish activation function (Ramachandran et al., 2017) which Bevilacqua and Navigli use. In practice, we renormalise the output distribution so only senses with $d \in \mathcal{D}_w$ have nonzero probabilities.

C Additional SMD Models

SMD Baseline We compute a simple SMD baseline, which takes a contextualised embedding and passes it through an MLP to make predictions

$$\mathbf{h} = \texttt{TokenEmb}(t) \tag{11}$$
$$p_{\boldsymbol{\psi}}(m \mid t) = \sigma\big(\texttt{MLP}_{\boldsymbol{\psi}}^{\langle k,1\rangle}(\mathbf{h})\big)$$

where $\boldsymbol{\psi}$ are sets of parameters.

MelBERT For a high-performing SMD model, we experiment with a reimplementation of MelBERT (Choi et al., 2021). Let $\texttt{SentEmb} : \mathcal{T} \mapsto \mathbb{R}^k$ return a k-dimensional sentence embedding (for BERT, this can correspond to the BOS token). We define MelBERT as

$$\mathbf{h}_{\text{tok}} = \texttt{TokenEmb}(t) \tag{12}$$
$$\mathbf{h}_{\text{SPV}} = \texttt{MLP}_{\boldsymbol{\psi}_1}^{\langle 2k,k\rangle}\big(\mathbf{h}_{\text{tok}} \circ \texttt{SentEmb}(t)\big)$$
$$\mathbf{h}_{\text{MIP}} = \texttt{MLP}_{\boldsymbol{\psi}_2}^{\langle 2k,k\rangle}\big(\mathbf{h}_{\text{tok}} \circ \texttt{TypeEmb}(w)\big)$$
$$\mathbf{h}_{\text{both}} = \mathbf{h}_{\text{MIP}} \circ \mathbf{h}_{\text{SPV}}$$
$$p_{\langle \boldsymbol{\psi}_1,\boldsymbol{\psi}_2,\boldsymbol{\psi}_3\rangle}(m \mid t) = \sigma\big(\texttt{linear}_{\boldsymbol{\psi}_3}^{\langle 2k,1\rangle}(\mathbf{h}_{\text{both}})\big)$$

where $\boldsymbol{\psi}_1$, $\boldsymbol{\psi}_1$, $\boldsymbol{\psi}_3$ are sets of parameters. We also use BERT rather than RoBERTa (Liu et al., 2019), for parity with other models.

D Implementation and Training Details

Embedding Spaces For $\texttt{TokenEmb}(t)$, we use the output of BERT base (Devlin et al., 2019). Following Bevilacqua and Navigli, we average the last four layers, and for wordforms which correspond to multiple BERT tokens, we use the first. For $\texttt{SynsetEmb}(d)$, we average all of the word sense ARES embeddings (Scarlini et al., 2020) associated with d (this approach is also following Bevilacqua and Navigli), and pass them through SVD (default parameters from `scikit-learn`, Pedregosa et al., 2011) to make them the same dimentionality as BERT. For $\texttt{TypeEmb}(w)$, we follow Choi et al. (2021) and compute the BERT embedding where the input is w on its own. In practice, then, the dimentionality of all our embedding spaces are $k = 768$.

Data Splits Having performed the filtration described in §4.3, and additionally removing any datapoint which does not align with the BERT tokenisation scheme, we finally compute our own datasplits, shown in Table 6.

Dataset	# Train	# Dev	# Test
VUAMC (All)	75,395	8,818	9,594
VUAMC (Conv.)	71,920	8,539	9,169
SemCor	141,025	17,701	17,481

Table 6: SMD and WSD datasplits

Because of our data filtering process, numbers shown for EWISER and MelBERT in our paper cannot be compared with the originals.

Implementation We implement our models in `PyTorch` (Paszke et al., 2019). Our MLP is implemented so each middle layer is the same size, which is controlled by a hyperparameter. Each layer consists of Dropout (Srivastava et al., 2014) then a linear layer, then a ReLU activation function (ReLU is excluded from the output of the final linear layer). As an optimiser, we use AdamW (Loshchilov and Hutter, 2017). We train in batches of 128 datapoints at a time (from both SMD and WSD datasets simultaneously).

Training and Loss In practise, after training the objective in eq. (7), we freeze the WSD model, set α to 1, then continue training to finetune the MPD component. This means that if initially $\alpha = 0$, the WSD and MPD subcomponents are trained sequentially. If initially $\alpha = 1$, meanwhile, the distribution over s is inferred implicitly, without learning WSD. More specifically, every 50 iterations we compute the loss on the development set, and if this loss does not decrease for 5 consecutive checks then our early stopping criteria is met. The first time this criteria is met we freeze the WSD component, divide the learning rate by a divisor (a hyperparameter), set $\alpha = 1$, recover the best model on the development set so far, and resume training (in effect fine-tuning the MPD component); the second time it is met we recover the best model and stop training completely.

#	Experiment	Model	α	x	n_ϕ	n_θ	h_ϕ	h_θ	γ	γ_d
1	SMD (All)	Ours (w/ WSD Baseline)	0.8	0.1	1	3	500	300	5×10^{-4}	1
2		MelBERT		0.2	$3^\dagger$	$1^\ddagger$	$500^\dagger$	$300^\ddagger$	1×10^{-4}	10
3		BERT SMD Baseline		0.2	2^*		300^*		5×10^{-4}	1
4	SMD (Conv.)	Ours (w/ EWISER)	0.8	0.2	3	4	500	300	1×10^{-4}	1
5		MelBERT		0.1	$3^\dagger$	$3^\ddagger$	$300^\dagger$	$500^\ddagger$	1×10^{-4}	10
6		BERT SMD Baseline		0.2	3^*		500^*		1×10^{-4}	1
7	WSD (WSD & SMD)	EWISER	0.2	0.1	3	4	300	500	1×10^{-4}	10
8		Baseline	0.2	0.1	3	3	300	500	1×10^{-4}	10
9	WSD (WSD only)	EWISER		0.1	4		500		1×10^{-4}	10
10		Baseline		0.1	1		300		5×10^{-4}	1
11	MPD	Ours (w/ EWISER)	0.4	0.2	3	4	500	300	1×10^{-4}	1
12		Ours (w/ WSD Baseline)	0.2	0.1	1	3	500	300	5×10^{-4}	1
13		MelBERT Average	Uses model 2 (above)							

Table 7: Final hyperparameters of the models presented in §4

E Hyperparameter Tuning

Hyperparameter Search For each value of $\alpha \in \{0.0, 0.2, 0.4, 0.6, 0.8, 1.0\}$, we perform a random search over the following hyperparameters:

- The number of layers of $\text{MLP}_{\boldsymbol{\theta}}$ and $\text{MLP}_{\boldsymbol{\phi}}$, denoted n_θ and n_ϕ, sampled from $\{1, 2, 3, 4\}$.
- The dimensionality of the hidden state of $\text{MLP}_{\boldsymbol{\theta}}$ and $\text{MLP}_{\boldsymbol{\phi}}$, denoted h_θ and h_ϕ, sampled from $\{100, 300, 500\}$.
- The Dropout (Srivastava et al., 2014), denoted x, sampled from $\{0.1, 0.2, 0.3, 0.4\}$.
- The learning rate, denoted γ, sampled from $\{0.005, 0.001, 0.0005, 0.0001\}$.
- The learning rate divisor, denoted γ_d, sampled from $\{1, 10\}$.

For each model, and each value of α, we train 20 samples from this hyperparameter space.

Model Selection For SMD, we choose models with the best F1 on the VUAMC development set. For WSD, we do the same but on the SemCor set. For MPD, we choose the models with the best mean SMD and WSD performance (again on the development sets).

Final Hyperparameters The final hyperparameters are shown in Table 7 (* are hyperparameters of ψ not ϕ; † are hyperparameters of ψ_1 not ϕ; ‡ are hyperparameters of ψ_2 not θ).

Machine Reading, Fast and Slow: When Do Models "Understand" Language?

Sagnik Ray Choudhury[1*, 2], Anna Rogers[2], and Isabelle Augenstein[2]

[1]University of Michigan
[2]University of Copenhagen
sagnikrayc@gmail.com, arogers@sodas.ku.dk, augenstein@di.ku.dk

Abstract

Two of the most fundamental challenges in Natural Language Understanding (NLU) at present are: (a) how to establish whether deep learning-based models score highly on NLU benchmarks for the 'right' reasons; and (b) to understand what those reasons would even be. We investigate the behavior of reading comprehension models with respect to two linguistic 'skills': coreference resolution and comparison. We propose a definition for the reasoning steps expected from a system that would be 'reading slowly', and compare that with the behavior of five models of the BERT family of various sizes, observed through saliency scores and counterfactual explanations. We find that for comparison (but not coreference) the systems based on larger encoders are more likely to rely on the 'right' information, but even they struggle with generalization, suggesting that they still learn specific lexical patterns rather than the general principles of comparison.

1 Introduction

Generally, human decisions may be based on deliberate, careful reasoning ('slow thinking') or quick heuristics ('fast thinking') (Kahneman, 2011). These two processes have parallels in the realm of reading comprehension (RC): a human reader would ideally fully process the text to answer questions, but in practice, we may deliberately skim rather than read to save effort. Even capable students may be misled by superficial cues (Ackerman et al., 2013).

The previous generations of NLP models have already achieved high performance on many RC benchmarks, but they were found to often 'read fast', i.e. rely on shallow patterns (Chen et al., 2016; Jia and Liang, 2017; Rychalska et al., 2018). Fine-tuned Transformer-based models (Devlin et al., 2019) still have similar shortcomings

Context: Leo Strauss was a political philosopher and classicist. He was born in Germany ... Thoughts on Machiavelli is a book by Leo Strauss ...
Question: Where was the author of Thoughts of Machiavelli born ?
Answer: Germany

Figure 1: A sample question from the SQuAD (Rajpurkar et al., 2016) dataset. green tokens are the words that a reader relying on coreference resolution would take into account, and red tokens are the words that could be used to answer the question with entity type matching.

(Sugawara et al., 2020; Rogers et al., 2020; Sen and Saffari, 2020; Kassner and Schütze, 2020, inter alia) in RC, as well as other tasks (McCoy et al., 2019; Jin et al., 2020).

Consider the example in Figure 1. A human reader would ideally construct the coreference chain resolving the pronoun 'he' to 'Leo Strauss'. A possible heuristic-based solution is entity type matching (Jia and Liang, 2017): a model could observe that a 'where' question can only be answered by a 'location' and among two such entities ('Germany' and 'United States') the correct answer ('Germany') is closer to 'born'. Such heuristic reasoners will not generalize to unseen examples. Thus a key challenge in building trustworthy and explainable RC systems is to make sure their decisions are based on valid reasoning steps. However, it is difficult to establish: (a) what that reasoning should be; and (b) whether a blackbox system adheres to it.

The present study proposes a framework for the analysis of RC models that includes: (a) defining the expected reasoning; (b) analysing model performance using explainability techniques. In particular, we contribute a case study for RC questions involving coreference resolution and comparison: we define the expected 'reasoning' for them (§2) and use a combination of saliency-based and coun-

* Work done while employed at the University of Copenhagen

Proceedings of the 29th International Conference on Computational Linguistics, pages 78–93
October 12–17, 2022.

terfactual explanations (§3) to analyze RC systems based on BERT and RoBERTa encoders of various sizes (§4). Overall, we find that the larger models are more likely to rely on the 'right' information, but even they seem to learn specific lexical patterns rather than underlying linguistic phenomena.

2 When do RC Model 'Understand' A Text?

2.1 Understanding in Humans

The phenomenon of 'natural language understanding' is not yet sufficiently well defined even for human speakers, although it is pursued by at least three different fields: philosophy of mind (e.g. Grimm, 2021; Dellsén, 2020), psychology (e.g. Christianson, 2016; Zwaan, 2016), and pedagogy (e.g. Lander, 2010; Duffin and Simpson, 2000). We cannot do this topic justice within the scope of this paper, but let us briefly outline the key premises about human understanding that we rely on in our work:

- Understanding is not truth-connected: it is "a merely psychological state" (Grimm, 2012);
- Its objects are something like 'connections' or 'relations' of the phenomenon X to other phenomena (Grimm, 2021);
- It is not binary: teachers routinely talk of 'levels of understanding', 'continuum of understanding' or 'partial understanding' (Nurhuda et al., 2017);
- It is different from 'knowledge', i.a. since it is "not transmissible[1] in the same sense as knowledge is" (Burnyeat and Barnes, 1980).

If human understanding is about establishing connections between new and existing conceptualizations, its success depends on the pre-existence of a suitable set of conceptualizations, to which the connections can be established (this is why e.g. algebra is taught in schools before differential calculus). The set of conceptualizations that each of us possesses is unique, since it depends on our experience of the world (cf. Fillmore's 'semantics of understanding' (Fillmore, 1985)). This, together with other factors like level of motivation, attention etc., explains the variation in human understanding: we may grasp different sets of possible connections between different aspects of the new phenomenon and our pre-existing worldview.

2.2 'Understanding' in Machines

Much research on human understanding focuses on mechanisms that fundamentally do not apply to current NLP systems, such as the distinction between 'knowledge' and 'understanding' or the fact that humans will fail to understand if they don't have suitable pre-existing conceptualizations (while an encoder will encode text even if its weights are random). Since the mechanism (and its results) is so fundamentally different, terms like 'natural language understanding' or 'reading comprehension'[2] for the current NLP systems are arguably misleading. It would be more accurate to talk instead of 'natural language processing' and 'information retrieval'.

While terms like 'understanding' are widely (mis)applied to models in AI research (Mitchell, 2021), their definitions are scarce. Turing famously posited that the question "can machines think?" is too ill-defined to deserve serious consideration, and replaced it with a behavioral test (conversation with a human judge) for when we would say that thinking occurs (Turing, 1950). Conceptually, this is still the idea underlying the 'NLU' benchmarks used today: we assume that for models to perform well on collections of tests such as GLUE (Wang et al., 2018, 2019), some capacity for language understanding is required, and hence if our systems get increasingly higher scores on such behavioral tests, this would mean progress on 'NLU'. However, just like the Turing test itself turned out to be "highly gameable" (Marcus et al., 2016), so are our tests[3] (Sugawara et al., 2020; Rogers et al., 2020; Sen and Saffari, 2020; Kassner and Schütze, 2020; McCoy et al., 2019; Jin et al., 2020, inter alia).

All this suggests that, at the very least, we need a better specification for the success criteria for such behavioral tests. Instead of asking *"Does my RC model "understand" language?"* we could ask: *"Does my RC model produce its output based*

[1] This is why, as any teacher knows from practice, simply presenting the students with definitions or principles does not necessarily result in understanding of those principles or definitions.

[2] Marcus and Davis (2019) dispute even the applicability of the term "reading", declaring the current QA/RC systems "functionally illiterate" since they cannot draw the implicit inferences crucial for human reading.

[3] In fact, the larger the dataset, the more of likely spurious patterns are to occur (Gardner et al., 2021). This presents a fundamental problem for data-hungry deep learning systems: "the models, unable to discern the intentions of the data set's designers, happily recapitulate any statistical patterns they find in the training data" (Linzen, 2020).

on valid information retrieval and inference strategies?" Then the next question is to specify what strategies would be valid and acceptable, which is possible to do on case-by-case basis.

With respect to 'machine reading comprehension', a recent proposal by Dunietz et al. (2020) is based on whether a model can extract certain information that should be salient for a human reader (e.g. spatial, temporal, causal relations in a story). However, a model can extract such 'right' information through a 'wrong' process, e.g. some shallow heuristic. Hence the definition of 'NLU' needs at least two components: (a) the specific information that the model is expected to be 'extract' from the text; and (b) a valid process with which such 'extraction' is performed. And this would still not be enough: the model could have simply memorized *both* the right answer and the strategy to find it for some limited set of examples. We argue that the third key prerequisite is the ability to generalize: to *consistently* use the 'right' information-seeking strategy in novel contexts.[4]

Thus we propose the following general success criteria for NLP systems:

Definition 1 *A NLP system has human-level competence with respect to its task X iff:*

(a) it is able to correctly perform the task X (identify the target information in QA, correctly classify texts, generate an appropriate translation etc.);
(b) it does so by relying predominantly on information that a competent human speaker would also find relevant[5];
(c) it does so consistently under distribution shifts that do not pose challenges to competent human speakers.

2.3 Reasoning an RC Model *should* Perform

The second principle in our Def. 1 is that the model should rely on the 'right' information. While models can discover patterns unknown to humans, a competent human reader should at least find such patterns relevant post-factum.

What information-seeking strategy is needed depends on the type of question and the context. Rogers et al. (2022) propose a classification of RC 'skills' into five main groups: situation/world modeling, different types of inference/logical reasoning, the ability to combine information in multi-step reasoning, knowing what kind of information is needed and where to find it, and interpreting/manipulating linguistic input. A single question may require the competency of several types of 'skills'.

This study contributes an empirical investigation on two RC 'skills' in the broad category of 'interpreting/manipulating linguistic input': coreference resolution and comparison. Both of them rely on the contextual information and linguistic competence. Assuming that a human reader would first read the question and then read the context in order to find the answer, they would need to perform roughly three steps: (a) to interpret the 'question' (akin to its transformation to a formal semantic representation or a query); (b) to identify the relevant information in the context through establishing the referential equality between expressions in the question and in the context; (c) to use that information to perform the operation of comparison or coreference resolution (see Table 1).[6]

2.4 Reasoning an RC Model *does* Perform

Having established what reasoning steps an RC model *should* perform, the next step would be to ascertain whether that is the case for specific models. But generally, the interpretability of DL models is an actively developed research area (Belinkov and Glass, 2019; Molnar, 2022). In this study, we rely on a combination of two popular post-hoc explanation techniques, but we also discuss their limitations, and expect that new methods could soon be developed and used in the overall paradigm for the analysis of RC models that we propose.

Attribution/saliency-based methods Li et al. (2016); Sundararajan et al. (2017) provide a saliency score for each token in the input, which shows how 'important' a given token is for the model decision in this instance. Figure 2 illustrates that such scores may not necessarily map onto human rationales.

To establish whether a model performs a given reasoning step (see Table 1), we define the following partition of the token space: the tokens the model *should* find important (positive) vs the ones it *should not* (negative). For example, to know if the model 'attends' to the entities being compared, we can define the positive partition as `{blind,`

[4]This does not preclude errors (humans make them too).

[5]Note that this leaves room for NLP systems to rely on patterns humans may not be even aware of, as long as such patterns are valid. E.g. if a system learned to make health outcome predictions based on latent information about unknown drug interactions, that would be the discovery of new knowledge that the experts would then accept – but not if its predictions were based on a spurious correlation with Marvel movie release dates.

[6]This definition could be developed further for more complex cases of coreference and comparison, or to model other variations of the human reading process, but this approximation suffices for our purposes and our RC data (see §3.1).

	Example	Step	Relevant Spans
Comparison	**Context**: Blind Shaft is a 2003 film about a pair of brutal con artists operating in the illegal coal mines of present day northern China. The Mask Of Fu Manchu is a 1932 pre-Code adventure film directed by Charles Brabin. **Question**: Which film came out earlier, Blind Shaft or The Mask Of Fu Manchu? **Answer**: The Mask Of Fu Manchu	Interpreting the question	*came out* relation: <film, release date> film entities: *Blind Shaft*, *The Mask Of Fu Manchu* *earlier:* date comparison **target:** min(release date$_{\textit{Blind Shaft}}$, release date$_{\textit{The Mask of Fu Manchu}}$)
		Identifying relevant information through referential equality	*Blind Shaft*$_{\mathbf{q}}$:= *Blind Shaft*$_{\mathbf{c}}$ *The Mask Of Fu Manchu*$_{\mathbf{q}}$:= *The Mask Of Fu Manchu*$_{\mathbf{c}}$. *came out*$_{\mathbf{q}}$:= <date, film> construction$_{\mathbf{c}}$ release dates: <*Blind Shaft*, *2003*>, <*The Mask Of Fu Manchu*, *1932*>
		Value comparison	**solution:** *earlier*$_{\mathbf{q}}$:= *min*$_{\mathbf{c}}$ $min(1932, 2003) = 1932$
Coreference	**Context:** Barack Obama was the 44th president of the US. He was born in Hawaii. **Question:** Who was born in Hawaii? **Answer:** Barack Obama.	Interpreting the question	*born* relation: <person, location> *Hawaii*: location **target:** *born*: <*Hawaii*, UNK>
		Identifying relevant information through referential equality	*Hawaii*$_{\mathbf{q}}$:= *Hawaii*$_{\mathbf{c}}$ *born* relation: <*he, Hawaii*>
		Coref. resolution	<*Barack Obama, he*> **solution:** *born* <*Barack Obama, Hawaii*>

Table 1: The basic reasoning steps for answering comparison and coreference questions.

[CLS] which film came out earlier , blind shaft or the mask of fu manchu ? [SEP] blind shaft is a 2003 film about a pair of brutal con artists operating in the illegal coal mines of present day northern china . the mask of fu manchu is a 1932 pre - code adventure film directed by charles bra ##bin . [SEP]

Figure 2: IG saliency scores example. Green/red denotes positive/negative scores.

`shaft, mask, of, fu, munchu}` and the negative partition as `{northern, china}`. If the model consistently follows this strategy, the average score should be higher for the positive rather than negative partition.

A limitation of saliency explanations is that they are not always faithful, i.e., do not reflect a model's true decision process (Atanasova et al., 2020, 2022a; Ye et al., 2021). Also, even when they are faithful, i.e., when we can reliably say that a model places more 'importance' on token i than token j in an instance, this does not imply that a *set* of tokens I is more salient than another set J.

Counterfactual explanations have the form: "had X not occurred, Y would not have occurred" (Molnar, 2022). In NLP, they are based on input perturbations (Kaushik et al., 2020; Gardner et al., 2020; Sen et al., 2021; Atanasova et al., 2022b). In our case, it translates to "had the model not relied on information X, it could not have answered both the original and the perturbed instance correctly". Thus the perturbation has to change the correct label, unlike for contrast sets (Gardner et al., 2020).

Counterfactual (CF) explanations are considered to be more faithful, since they identify input features that impact predictions. However, they typically have to be manually generated (Kaushik et al., 2020), which makes large-scale CF generation prohibitively expensive (Khashabi et al., 2020).

We rely on both types of explanations as parallel sources of evidence about RC model reasoning, and define their alignment as follows:

Definition 2 ***Explanation Alignment.*** *A CF and saliency-based explanation align when: (a) both the original and the counterfactually modified instance are answered correctly,*[7]*; and (b) the positive partition has a statistical significantly higher average saliency score than the negative partition.*

We define the alignment score as follows:

Definition 3 ***Alignment Score****: The Alignment Score for a <dataset, model, reasoning step> triple is the proportion of instances in that dataset for which different kinds of explanations align (according to our Def. 2).*

We interpret a high alignment score as evidence that both kinds of explanations are faithful, and the model indeed performs the expected reasoning steps.

[7] i.e. there is an exact match between the predicted and the correct answer.

3 Methodology

3.1 Datasets and Models

For **Coreference**, we use the Quoref (Dasigi et al., 2019) dataset (20K training and 2.4K validation instances) where the annotators were asked to *design* questions for a given text so that answering those would require resolving anaphora. For **Comparison**, we sample questions from HotpotQA (Yang et al., 2018) and 2WikiMultiHopQA (Ho et al., 2020): two datasets with questions manually annotated with their reasoning type (bridge or comparison). We select the 'comparison' questions containing comparative adjectives or adverbs in them (23K training, 3K validation instances). These resources are based on Wikipedia and have multiple passages as contexts, but the sentences (typically 2-3) necessary to answer a question are marked as 'supporting facts'. Since we are not focusing on the multi-hop information retrieval skill, we limit the contexts to these sentences.

We experiment with five pre-trained Transformer-based encoders of the BERT family: $\text{RoBERTa}_{\text{large}}$ (Liu et al., 2019), $\text{BERT}_{\text{large-cased}}$, $\text{BERT}_{\text{base-cased}}$ (Devlin et al., 2019), $\text{BERT}_{\text{medium}}$, and $\text{BERT}_{\text{small}}$ (Turc et al., 2019; Bhargava et al., 2021). These BERT models differ mainly in the structure of architecture blocks and the number of parameters, while RoBERTa also has a different training corpus and optimization. Since larger models were shown to generalize better for some use cases (Hendrycks et al., 2020; Bhargava et al., 2021), we investigate whether they also are more likely to be right for the right reasons.

We fine-tune each encoder using the architecture in Devlin et al. (2019) (see the appendix for details) and evaluate them on the validation set (as the test sets are not public). We use the standard evaluation metrics in extractive QA: **F1-Score** (the percentage of token overlap between predicted and 'gold' answers, averaged over all data points), and **Exact-match** (the number of data points where the predicted answer matches the 'gold' answer).

3.2 Counterfactual Explanations

Our formulation of reasoning (Table 1) consists of three basic steps for both coreference and comparison: interpreting the question, identifying the relevant information through referential equality, and the target operation on the identified information (coreference resolution or value comparison). We focus on the final step, since: (a) it implicitly relies on correct semantic parsing of the question and the context; (b) referential equality in our data is in large part trivial: most entities have the same surface form in the question and the text.

An obvious semantically valid perturbation that should change the prediction (and thus test for the model's understanding of the comparison operation) is to replace the comparative adjectives with their antonyms (Figure 3d). Since our sample only contains 6 tokens used as comparison operators, we define appropriate replacements manually.[8]

For coreference questions, a competent RC model would at least resolve the coreference chain for the target entity. A context can have many coreference clusters, so we need to identify the relevant one. In the Quoref dataset, we use the instances where the relevant cluster itself contains the answer entity[9] (see Figure 3a), and therefore, can be extracted automatically. This leaves us with $55\%(1329/2418)$ of the validation instances. These are further subsampled to manually create 100 CF instances by inserting a new sentence, which includes the new and excludes the old answer (see Figure 3b). Similarly to the comparison questions, the original answer entity remains in the context. If the model uses the 'shortcut' of choosing the most frequent entity in the context (Wu et al., 2021), it should not be able to answer both the original and the perturbed instance correctly.

3.3 Saliency-based Explanations

We obtain token saliency scores from two families of attribution/saliency methods: Occlusion (DeYoung et al., 2020), a method based on perturbations, and Integrated Gradients (IG, Sundararajan et al. (2017)), a method based on gradients.[10]

Design decisions: RC models typically predict two scores (t_s, t_e) for each token t: the probability of t being the start and the end of the answer span. Any attribution method produces two scores (A^t_{start}, A^t_{end}) for each token t, indicating how 'im-

[8] earlier↔later, first→later, more recently→earlier, older↔younger.

[9] We extract the clusters using an off-the-shelf coreference resolver (Clark and Manning, 2016) implemented in Spacy.

[10] Atanasova et al. (2020) shows that for Transformer based architectures, Occlusion is the best perturbation method by two evaluation criteria: agreement with human rationale and faithfulness. A recent paper by Ye et al. (2021) finds IG to be one of the most faithful gradient-based methods for extractive QA, only outperformed by Layerwise Attention Attribution (LAA), a method proposed in the paper itself. We leave LAA and other popular explainability methods such as LIME (Ribeiro et al., 2016) for future work.

Context: Górecki said of the work, ... I had a grandfather who was in Dachau, an aunt in Auschwitz. ...
Question: What is the last name of the person who had an aunt at Auschwitz?
Answer: Górecki

(a) A coreference question from Quoref (the relevant coreference cluster tokens in green)

Context: Górecki said of the work... I had a grandfather who was in Dachau. **I had a nephew named Mike Wazowski** . He had an aunt in Auschwith ...
Question: What is the last name of the person who had an aunt at Auschwitz?
Answer: Wazowski

(b) CF perturbation for 3a (added tokens are **bold-faced**)

Context: Blind Shaft is a 2003 film... The Mask Of Fu Manchu is a 1932 pre-Code adventure film...
Question: Which film came out earlier, Blind Shaft or The Mask Of Fu Manchu?
Answer: The Mask Of Fu Manchu

(c) A comparison question from 2WikiMultiHopQA

Context: Blind Shaft is a 2003 film... The Mask Of Fu Manchu is a 1932 pre-Code adventure film...
Question: Which film came out later , Blind Shaft or The Mask Of Fu Manchu?
Answer: Blind Shaft

(d) CF perturbation for 3c

Figure 3: Examples of CF perturbations used in this study.

Comparison: **Question**: Which film came out more recently , Blind Shaft or The Mask Of Fu Manchu ?
Coreference: **Context**: Barack Obama was the 44th president of the US . He was born in Hawaii. **Question:** Who was born in Hawaii?

Figure 4: Positive and negative partitions for saliency explanations.

portant' t is for predicting the start/end of the answer span. Following Kokhlikyan et al. (2020), we use A_{start} in all our saliency experiments.[11]

For Occlusion, we calculate A^t_{start} by replacing t in the input with a baseline token (`MASK`) and measuring the change in t_s. DNNs map an input vector to a scalar value (loss/ class probability). Gradient-based methods measure A^t_{start} using the gradient of the token t w.r.t. this scalar function (we use $argmax(t_s)$). IG sums these gradient values along a linear path from a baseline to the current instance. Both Occlusion and IG need a baseline token, which for us is the `MASK` token.

Gradient-based methods in NLP do not produce a scalar saliency score, i.e., A^t_{start} is a vector because the input is an embedding *matrix* and not a vector. Two common ways to summarize this vector to a scalar are: (a) scalar product between the input and the gradient vector (Han et al., 2020); or (b) l_p norm, where $p \in 1, 2$ (Atanasova et al., 2020). We use l_2 norm (see the discussion in §4.3).

Token partitions: Figure 4 shows the token partitions used for the same reasoning steps (comparison and coreference resolution) that we also target with the CF perturbations. For comparison the positive partition consists of the **question** token(s) expressing the comparison operation (e.g. 'more recently'). The negative partition consists of question tokens that are not in the set of entities or values that need to be compared, or in the set of verbs (which could capture the relation between the entities and their values). For coreference resolution, the positive partition is the **context** tokens in the relevant coreference cluster (§3.2). The negative partition is the set of context tokens that are not in: (a) the positive partition; and (b) match the question tokens.

4 Results & Analysis

4.1 Base Model Performance

As a sanity check, we fine-tune all models on the data described in §3.1 (Table 2). For coreference, the F1-Score of our best model (RoBERTa$_{\text{large}}$) is slightly better (82.10) than the previously reported score (79.64, Wu et al. (2021)). The comparison instances are sampled from *parts* of two datasets, and so a direct comparison is not possible.[12]

The size and the model family matter: RoBERTa performs better than BERT for two models of the same size, and the larger models do better. Interestingly, the difference is more pronounced for the Quoref dataset, where the instances have longer contexts and the questions are more complex.

[11] We also briefly experimented with $(A_{start} + A_{end})/2$ for Occlusion but it yielded very similar saliency ranking of the tokens on a 100 sample subset of the Comparison dataset.

[12] The best model (RoBERTa$_{\text{large}}$) has an F1-Score of 92%, slightly better than the highest score reported on the HotpotQA leaderboard (89.14%) and much better than the baseline model for the 2WikiMultiHopQA dataset (65.02, (Ho et al., 2020)).

	Comparison		Coreference	
	F1	EM	F1	EM
RoBERTa$_{\text{large}}$	92.08	91.07	82.10	79.39
BERT$_{\text{large-cased}}$	89.23	88.57	71.91	68.47
BERT$_{\text{base-cased}}$	89.38	88.37	64.62	59.38
BERT$_{\text{medium}}$	86.45	85.96	60.16	54.82
BERT$_{\text{small}}$	71.44	69.87	50.94	43.39

Table 2: Average (3 runs) results of different models on Comparison and Coreference datasets. The STD varies between $0.01 - 0.72\%$. Green indicates the best scores.

	Coreference		Comparison	
	og	cf	og	cf
RoBERTa$_{\text{large}}$	92.0	70.7	99.4	98.9
BERT$_{\text{large-cased}}$	86.2	50.8	98.9	93.1
BERT$_{\text{base-cased}}$	82.5	39.2	98.4	91.8
BERT$_{\text{medium}}$	74.0	35.8	97.4	96.5
BERT$_{\text{small}}$	67.2	29.4	68.2	45.3

Table 3: F1-Score for the original and the CF perturbations. Red denotes significant drop.

4.2 Counterfactual Explanations

Table 3 compares the F1-Score of the original ('og') vs counterfactual ('cf') instances. For the comparison questions, the performance on the original and CF instances are very close for all models except BERT$_{\text{small}}$. Bigger models consistently perform better, but in most cases the difference with the next larger model is relatively small.

For coreference questions, CF instances are much more difficult for all models. Even the best model RoBERTa$_{\text{large}}$ experiences a 24% drop. All BERT models perform poorly: even the larger ones have a 40% performance drop (BERT$_{\text{large-cased}}$). Thus, the CF tests show that the models are more likely to follow the expected reasoning strategy for comparison, but not for the coreference questions.

4.3 Alignment Score

For statistical significance testing in 'Expectation Alignment' (Def. 2), we use a one-tailed independent t_test ($p = 0.05$) with the null hypothesis that *the positive partition does not have a higher average saliency score*. Table 4 shows the 'Alignment Score' (Def. 3) results for comparison and coreference resolution (§3.3), using saliency scores from IG and Occlusion.

Ideally, for a random partition of tokens in any instance, the positive and the negative partitions should have similar saliency scores. For a dataset, they should be *significantly* different in $\approx 0\%$ cases.[13] For Occlusion, the saliency scores are significantly different in only $5.6 - 8.2\%$ instances for a random partition. Recall that in §3.3 we discussed 3 summarizers for IG. Among all of them, the l_2 norm is the only one where this happens in $5.2 - 7.3\%$ cases, for the other two the numbers are between $11.3 - 28.9\%$.

	Coreference		Comparison	
	IG	Occ	IG	Occ
RoBERTa$_{\text{large}}$	33.3	69.7	33.8	67.0
BERT$_{\text{large-cased}}$	12.5	58.3	34.1	65.9
BERT$_{\text{base-cased}}$	21.4	42.9	83.8	69.0
BERT$_{\text{medium}}$	81.8	36.4	82.2	42.0
BERT$_{\text{small}}$	83.3	33.3	86.3	16.3

Table 4: Alignment score between counterfactual explanations vs IG (Integrated Gradients) or Occ (Occlusion). Green indicates methods with $> 80\%$ alignment.

Table 4 shows that, counter-intuitively, for both comparison and coreference questions the larger models overall have *lower* IG alignment scores, meaning that they do not pay as much 'attention' to the tokens we defined as important. This is despite the fact that for comparison the above CF experiment suggests that the models do perform the expected reasoning operations. One possible explanation is that IG simply does not reliably capture the model's reasoning process, and Occlusion does better at that because its trend in alignment is the opposite of IG: bigger models tend to have significantly higher alignment scores.[14]

Another possible explanation is that IG explanations are in fact faithful, but, having more 'attention' to the tokens we defined as important is counter-productive. Consider that the BERT$_{\text{small}}$ model achieves an Exact-match of 87% on the original questions containing the comparative tokens 'earlier', 'first' and 'older' (which are 2.1 times more frequent in the training data than all others), and an Exact-match of 28% on the other original questions. Yet overall the model performs poorly, and thus the reliance on these highly frequent comparative adjectives could be a bug rather than a feature. As this hypothesis brings into question the overall utility of saliency-based explanations for testing for the 'correct' reasoning steps, we hope it will be investigated in more depth in future work.

[13] Aggregation of local explanations such as saliency scores are not *guaranteed* to produce faithful global explanations (Setzu et al., 2021), but this is a convincing evidence.

[14] The lack of alignment between the two techniques is consistent with the findings of Atanasova et al. (2020).

	Supporting Facts			Paragraphs		
	OG	CF	CF-ood	OG	CF	CF-ood
$\text{RoBERTa}_{\text{large}}$	99.4	98.9	77.2	98.7	96.4	74.8
$\text{BERT}_{\text{large-cased}}$	98.9	93.1	68.7	98.0	90.8	67.5
$\text{BERT}_{\text{base-cased}}$	98.4	91.8	58.1	97.0	86.8	59.9
$\text{BERT}_{\text{medium}}$	97.4	96.5	64.4	96.2	86.3	66.3
$\text{BERT}_{\text{small}}$	68.2	45.3	57.1	68.3	47.6	58.8

Table 5: F1-Score for the original (OG) comparison questions and their counterfactual perturbations in (CF) and out (CF-ood) of the training distribution. The models are provided either a smaller context of supporting facts or full paragraphs. Red indicates a significant drop in performance.

	Coreference		SQuAD	
	F1-Score	EM	F1-Score	EM
Token overlap	21.5	12.9	26.68	21.64
LCS	17.2	12.9	19.59	15.97
Position	12.3	7.9	21.62	16.32
Sentence encoder	20.43	9.67	25.91	20.61

Table 6: Results for different heuristic methods on the coreference and SQuAD datasets. Green indicates the best score.

4.4 Generalization Tests

Table 3 shows that when measured with CF tests, most models do not follow the expected coreference resolution strategy, but they do so for comparison. Still, based on our success criteria (Def. 1), we cannot yet conclude that they 'understand' comparison. A human would be able to disassociate the logical operation of comparison from the surface realizations, i.e., they would be able to answer a question correctly with either of the surface forms 'younger' and 'more junior'.

For the CF experiments reported up until this point the perturbations were in-distribution, i.e., the training data had both the original question "who is younger" and the CF "who is older". Now we replace the comparative adjectives with antonyms that are not in the training data (see the appendix for details). We also increase the context size by using full paragraphs instead of just the sentences marked as 'supporting facts', to see if the models would be 'distracted' by more information.

Table 5 shows a considerable drop in performance for CF-ood condition for all models. The larger models generalize better: $\text{RoBERTa}_{\text{large}}$ and $\text{BERT}_{\text{large-cased}}$ perform 2% and 8% worse for CF questions, whereas $\text{BERT}_{\text{small}}$ exhibits a 29% reduction. The 'supporting facts only' condition is overall easier than the 'paragraphs' condition.

4.5 Heuristics for Coreference Questions

Since the CF tests (§4.2) do not show that BERT models can cope with the altered coreference chains, we have to conclude that they do not follow the expected reasoning steps. Though given the above-chance performance they must follow some other strategy. We test the hypothesis that many of the coreference questions can be answered by simple heuristics and that the models resort to those. Specifically, we define an unsupervised dataset-independent heuristic method consisting of two steps: sentence selection and phrase extraction.

Sentence Selection: Among all the context sentences $\{c_i\}$, select the one that is the 'closest' to the question q. We experiment with 4 options for similarity: **token-overlap** (number of common tokens in q and c_i), **sentence encoder** (cosine similarity between the sentence embeddings of q and c_i created by a sentence encoder (Reimers and Gurevych, 2019)), **LCS** (number of tokens in the Longest Common Subsequence between q and c_i), and **position** (simply taking the first sentence in the context following Ko et al. (2020)).

Phrase Extraction: We assume that the model would also learn to look for a named entity in the selected sentence. The question dictates the *type* of this entity (e.g. 'where' $\rightarrow$ location, 'who' $\rightarrow$ person name). The type could be determined by a simple mapping between 'wh' question words and entity types, but this can fail (e.g. for the question "Who won the World Cup in 2002?" the expected answer is a location, not a person). Therefore, we fine-tune a Transformer model to predict the answer type from the question.[15]

Table 6 shows the best heuristic has an F1-Score of 21.5% on the coreference dataset, and 26.68% on SQuAD (Rajpurkar et al., 2016), which we use for validation. The SQuAD score is comparable to the previously reported result of 26.7% in Sen and Saffari (2020) for an algorithm predicting entity types heuristically, and choosing the entity from the whole context instead of the best possible sentence. Ray Choudhury et al. (2022) uses a similar approach to find Quoref questions that can be answered heuristically, but our algorithm has

[15]The accuracy for this model is 85.7%. See the appendix for results from multiple models and loss functions.

more sentence selection strategies, and unlike ours, Ray Choudhury et al. (2022) only uses one loss function in the phrase extraction model.

Nevertheless, the best heuristic algorithm performs considerably worse than the smallest $\text{BERT}_{\text{small}}$ model (51%, Table 2). Performance alone cannot reveal whether this strategy is used in the instances where it *would* be sufficient, but this result shows that even the smaller models must either rely on a more successful (but still imperfect) strategy, or at least rely on more than one heuristic. The problem with discovering potential 'shortcuts' in low-performing models is complicated as these strategies are not necessarily human-interpretable: González et al. (2021) show that humans struggle to predict the answer chosen by poorly performing RC models, even when the saliency explanations for that answer is shown, because these answers simply do not align with human RC strategies.

5 Discussion and Related Work

Our work continues the emerging trend of research on being 'right for the right reasons' (McCoy et al., 2019; Chen and Durrett, 2019; Min et al., 2019; Atanasova et al., 2022b, inter alia). We contribute stricter success criteria for behavioral tests of NLP models (Def. 1), and, for the RC task, develop the methodology of: (a) defining what information the model should rely on for a given linguistic, logical, or world knowledge 'skill'; (b) systematically testing the behavior of RC models with interpretability techniques for whether they rely on that information. This is most closely related to the work on 'defining comprehension' by Dunietz et al. (2020), though their testing is limited to probing the models with RC questions. Another related study is the QED framework (Lamm et al., 2021), annotating Natural Questions (Kwiatkowski et al., 2019) with 'explanations' of the expected reasoning process. Their expected reasoning process also contains 3 steps, partly similar to ours: selecting a relevant sentence, referential equality, and deciding on whether this sentence entails the predicate in the question. However, the goals of QED are to: (a) predict both the answer and the explanation for a question; and (b) understand if explanations help QA models. Such explanation annotations are unavailable for most datasets, and few QA models produce explanations. Therefore, our approach of: (a) defining expected reasoning steps; and (b) using model interpretations to validate such steps applies to a broader class of models.

This study is also related to the overall efforts to define what kinds of 'skills' RC models can be expected to exhibit (Sugawara et al., 2018; Schlegel et al., 2020; Rogers et al., 2022). While these works focus on the high-level taxonomies of 'skills', we contribute practical definitions for two linguistic 'skills' (comparison and coreference resolution) which could be used for analyzing model performance. Implicitly, research proposing RC resources that target various specific 'skills' (e.g. TempQuestions (Jia et al., 2018) for temporal order, MathQA (Amini et al., 2019) for numerical reasoning, etc.) also contributes to this area, but they typically rely on broad linguistic definitions rather than on steps for machine reasoning.

The saliency techniques we rely on have previously been used for extractive QA (Madsen et al., 2021), but we are among the first (Ye et al., 2021) to investigate their correlation with counterfactual explanations. For counterfactual perturbations, we also ensure that the perturbations are human-interpretable and change the prediction, which is not the case for adding incomprehensible text (Kaushik and Lipton, 2018), removing words from questions, shuffling the context (Sen and Saffari, 2020), or replacing context tokens with random tokens (Sugawara et al., 2020).

6 Conclusion

Making progress towards trustworthy NLP models requires specific definitions for the behavior expected of these models in different situations. We propose a framework for RC model analysis that involves: (a) the definition of the expected 'reasoning' steps; (b) analysis of model behavior. We contribute such definitions for two linguistic 'skills' (comparison and coreference resolution), and use parallel explainability techniques to investigate whether RC models based on BERT family encoders answer such questions correctly for the right reasons. We find that to be the case for comparison, but not for coreference. Moreover, we find that, even for comparison, the models 'break' when encountering out-of-distribution counterfactual perturbations, suggesting that they memorize specific lexical patterns rather than learn more general reasoning 'skills'. As such, more research is needed on developing definitions and tests for specific 'skills' expected of NLU models, as well as on more faithful interpretability techniques.

7 Acknowledgements

We would like to thank Pepa Atanasova, Gary Marcus, Mark Steedman, and Bonnie Webber for the discussion of various aspects of this work. We also thank the anonymous reviewers for their time and insightful comments.

References

Rakefet Ackerman, David Leiser, and Maya Shpigelman. 2013. Is Comprehension of Problem Solutions Resistant to Misleading Heuristic Cues? *Acta Psychologica*, 143(1):105–112.

Aida Amini, Saadia Gabriel, Shanchuan Lin, Rik Koncel-Kedziorski, Yejin Choi, and Hannaneh Hajishirzi. 2019. MathQA: Towards Interpretable Math Word Problem Solving with Operation-Based Formalisms. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 2357–2367, Minneapolis, Minnesota. Association for Computational Linguistics.

Pepa Atanasova, Jakob Grue Simonsen, Christina Lioma, and Isabelle Augenstein. 2020. A Diagnostic Study of Explainability Techniques for Text Classification. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing, EMNLP 2020, Online, November 16-20, 2020*, pages 3256–3274. Association for Computational Linguistics.

Pepa Atanasova, Jakob Grue Simonsen, Christina Lioma, and Isabelle Augenstein. 2022a. Diagnostics-Guided Explanation Generation. *Proceedings of the AAAI Conference on Artificial Intelligence*, 36(10):10445–10453.

Pepa Atanasova, Jakob Grue Simonsen, Christina Lioma, and Isabelle Augenstein. 2022b. Fact Checking with Insufficient Evidence. *Transactions of the Association for Computational Linguistics*, 10:746–763.

Yonatan Belinkov and James Glass. 2019. Analysis Methods in Neural Language Processing: A Survey. *Transactions of the Association for Computational Linguistics*, 7:49–72.

Prajjwal Bhargava, Aleksandr Drozd, and Anna Rogers. 2021. Generalization in NLI: Ways (Not) To Go Beyond Simple Heuristics. In *Proceedings of the Second Workshop on Insights from Negative Results in NLP*, pages 125–135, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

M. F. Burnyeat and Jonathan Barnes. 1980. Socrates and the Jury: Paradoxes in Plato's Distinction between Knowledge and True Belief. *Proceedings of the Aristotelian Society, Supplementary Volumes*, 54:173–206.

Danqi Chen, Jason Bolton, and Christopher D. Manning. 2016. A Thorough Examination of the CNN/Daily Mail Reading Comprehension Task. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 2358–2367. Association for Computational Linguistics.

Jifan Chen and Greg Durrett. 2019. Understanding Dataset Design Choices for Multi-hop Reasoning. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT 2019, Minneapolis, MN, USA, June 2-7, 2019, Volume 1 (Long and Short Papers)*, pages 4026–4032. Association for Computational Linguistics.

Kiel Christianson. 2016. When Language Comprehension Goes Wrong for the Right Reasons: Good-enough, Underspecified, or Shallow Language Processing. *Quarterly Journal of Experimental Psychology*, 69(5):817–828.

Kevin Clark and Christopher D. Manning. 2016. Deep Reinforcement Learning for Mention-Ranking Coreference Models. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing, EMNLP 2016, Austin, Texas, USA, November 1-4, 2016*, pages 2256–2262. The Association for Computational Linguistics.

Pradeep Dasigi, Nelson F. Liu, Ana Marasovic, Noah A. Smith, and Matt Gardner. 2019. Quoref: A Reading Comprehension Dataset with Questions Requiring Coreferential Reasoning. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing, EMNLP-IJCNLP 2019, Hong Kong, China, November 3-7, 2019*, pages 5924–5931. Association for Computational Linguistics.

Finnur Dellsén. 2020. Beyond Explanation: Understanding as Dependency Modelling. *The British Journal for the Philosophy of Science*, 71(4):1261–1286.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT 2019, Minneapolis, MN, USA, June 2-7, 2019, Volume 1 (Long and Short Papers)*, pages 4171–4186. Association for Computational Linguistics.

Jay DeYoung, Sarthak Jain, Nazneen Fatema Rajani, Eric Lehman, Caiming Xiong, Richard Socher, and Byron C. Wallace. 2020. ERASER: A Benchmark to Evaluate Rationalized NLP Models. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, ACL 2020, Online, July 5-10, 2020*, pages 4443–4458. Association for Computational Linguistics.

Janet M. Duffin and Adrian P. Simpson. 2000. A Search for Understanding. *The Journal of Mathematical Behavior*, 18(4):415–427.

Jesse Dunietz, Gregory Burnham, Akash Bharadwaj, Owen Rambow, Jennifer Chu-Carroll, and David A. Ferrucci. 2020. To Test Machine Comprehension, Start by Defining Comprehension. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, ACL 2020, Online, July 5-10, 2020*, pages 7839–7859. Association for Computational Linguistics.

Charles J Fillmore. 1985. Frames and the Semantics of Understanding. *Quaderni di Semantica*, 6(2):222–254.

Matt Gardner, Yoav Artzi, Victoria Basmov, Jonathan Berant, Ben Bogin, Sihao Chen, Pradeep Dasigi, Dheeru Dua, Yanai Elazar, Ananth Gottumukkala, Nitish Gupta, Hannaneh Hajishirzi, Gabriel Ilharco, Daniel Khashabi, Kevin Lin, Jiangming Liu, Nelson F. Liu, Phoebe Mulcaire, Qiang Ning, Sameer Singh, Noah A. Smith, Sanjay Subramanian, Reut Tsarfaty, Eric Wallace, Ally Zhang, and Ben Zhou. 2020. Evaluating Models' Local Decision Boundaries via Contrast Sets. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 1307–1323, Online. Association for Computational Linguistics.

Matt Gardner, William Merrill, Jesse Dodge, Matthew E. Peters, Alexis Ross, Sameer Singh, and Noah A. Smith. 2021. Competency Problems: On Finding and Removing Artifacts in Language Data. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing, EMNLP 2021, Virtual Event / Punta Cana, Dominican Republic, 7-11 November, 2021*, pages 1801–1813. Association for Computational Linguistics.

Ana Valeria González, Anna Rogers, and Anders Søgaard. 2021. On the Interaction of Belief Bias and Explanations. In *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*, pages 2930–2942, Online. Association for Computational Linguistics.

Stephen Grimm. 2012. The Value of Understanding. *Philosophy Compass*, 7(2):103–117.

Stephen Grimm. 2021. Understanding. In Edward N. Zalta, editor, *The Stanford Encyclopedia of Philosophy*, summer 2021 edition. Metaphysics Research Lab, Stanford University.

Xiaochuang Han, Byron C. Wallace, and Yulia Tsvetkov. 2020. Explaining Black Box Predictions and Unveiling Data Artifacts through Influence Functions. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, ACL 2020, Online, July 5-10, 2020*, pages 5553–5563. Association for Computational Linguistics.

Dan Hendrycks, Xiaoyuan Liu, Eric Wallace, Adam Dziedzic, Rishabh Krishnan, and Dawn Song. 2020. Pretrained Transformers Improve Out-of-Distribution Robustness. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 2744–2751, Online. Association for Computational Linguistics.

Xanh Ho, Anh-Khoa Duong Nguyen, Saku Sugawara, and Akiko Aizawa. 2020. Constructing A Multi-hop QA Dataset for Comprehensive Evaluation of Reasoning Steps. In *Proceedings of the 28th International Conference on Computational Linguistics, COLING 2020, Barcelona, Spain (Online), December 8-13, 2020*, pages 6609–6625. International Committee on Computational Linguistics.

Robin Jia and Percy Liang. 2017. Adversarial Examples for Evaluating Reading Comprehension Systems. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing, EMNLP 2017, Copenhagen, Denmark, September 9-11, 2017*, pages 2021–2031. Association for Computational Linguistics.

Zhen Jia, Abdalghani Abujabal, Rishiraj Saha Roy, Jannik Strötgen, and Gerhard Weikum. 2018. TempQuestions: A Benchmark for Temporal Question Answering. In *Companion of the The Web Conference 2018 on The Web Conference 2018 - WWW '18*, pages 1057–1062, Lyon, France. ACM Press.

Di Jin, Zhijing Jin, Joey Tianyi Zhou, and Peter Szolovits. 2020. Is BERT Really Robust? A Strong Baseline for Natural Language Attack on Text Classification and Entailment. In *The Thirty-Fourth AAAI Conference on Artificial Intelligence, AAAI 2020, The Thirty-Second Innovative Applications of Artificial Intelligence Conference, IAAI 2020, The Tenth AAAI Symposium on Educational Advances in Artificial Intelligence, EAAI 2020, New York, NY, USA, February 7-12, 2020*, pages 8018–8025. AAAI Press.

Daniel Kahneman. 2011. *Thinking, Fast and Slow*. Farrar, Straus and Giroux, New York.

Nora Kassner and Hinrich Schütze. 2020. Negated and Misprimed Probes for Pretrained Language Models: Birds Can Talk, But Cannot Fly. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, ACL 2020, Online, July 5-10, 2020*, pages 7811–7818. Association for Computational Linguistics.

Divyansh Kaushik, Eduard H. Hovy, and Zachary Chase Lipton. 2020. Learning The Difference That Makes A Difference With Counterfactually-Augmented Data. In *8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, April 26-30, 2020*. OpenReview.net.

Divyansh Kaushik and Zachary C. Lipton. 2018. How Much Reading Does Reading Comprehension Require? A Critical Investigation of Popular Benchmarks. In *Proceedings of the 2018 Conference on*

Empirical Methods in Natural Language Processing, Brussels, Belgium, October 31 - November 4, 2018, pages 5010–5015. Association for Computational Linguistics.

Daniel Khashabi, Tushar Khot, and Ashish Sabharwal. 2020. More Bang for Your Buck: Natural Perturbation for Robust Question Answering. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing, EMNLP 2020, Online, November 16-20, 2020*, pages 163–170. Association for Computational Linguistics.

Yoon Kim. 2014. Convolutional Neural Networks for Sentence Classification. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1746–1751, Doha, Qatar. Association for Computational Linguistics.

Diederik P. Kingma and Jimmy Ba. 2015. Adam: A Method for Stochastic Optimization. In *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*.

Miyoung Ko, Jinhyuk Lee, Hyunjae Kim, Gangwoo Kim, and Jaewoo Kang. 2020. Look at the First Sentence: Position Bias in Question Answering. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing, EMNLP 2020, Online, November 16-20, 2020*, pages 1109–1121. Association for Computational Linguistics.

Narine Kokhlikyan, Vivek Miglani, Miguel Martin, Edward Wang, Bilal Alsallakh, Jonathan Reynolds, Alexander Melnikov, Natalia Kliushkina, Carlos Araya, Siqi Yan, and Orion Reblitz-Richardson. 2020. Captum: A unified and generic model interpretability library for PyTorch. *CoRR*, abs/2009.07896.

Tom Kwiatkowski, Jennimaria Palomaki, Olivia Redfield, Michael Collins, Ankur P. Parikh, Chris Alberti, Danielle Epstein, Illia Polosukhin, Jacob Devlin, Kenton Lee, Kristina Toutanova, Llion Jones, Matthew Kelcey, Ming-Wei Chang, Andrew M. Dai, Jakob Uszkoreit, Quoc Le, and Slav Petrov. 2019. Natural questions: a benchmark for question answering research. *Trans. Assoc. Comput. Linguistics*, 7:452–466.

Matthew Lamm, Jennimaria Palomaki, Chris Alberti, Daniel Andor, Eunsol Choi, Livio Baldini Soares, and Michael Collins. 2021. QED: A Framework and Dataset for Explanations in Question Answering. *Trans. Assoc. Comput. Linguistics*, 9:790–806.

Arthur D. Lander. 2010. The Edges of Understanding. *BMC Biology*, 8(1):40.

Jiwei Li, Will Monroe, and Dan Jurafsky. 2016. Understanding Neural Networks through Representation Erasure. *CoRR*, abs/1612.08220.

Tal Linzen. 2020. How Can We Accelerate Progress Towards Human-like Linguistic Generalization? In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, ACL 2020, Online, July 5-10, 2020*, pages 5210–5217. Association for Computational Linguistics.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. RoBERTa: A Robustly Optimized BERT Pretraining Approach. *CoRR*, abs/1907.11692.

Andreas Madsen, Nicholas Meade, Vaibhav Adlakha, and Siva Reddy. 2021. Evaluating the Faithfulness of Importance Measures in NLP by Recursively Masking Allegedly Important Tokens and Retraining. *CoRR*, abs/2110.08412.

Gary Marcus and Ernest Davis. 2019. *Rebooting AI: Building Artificial Intelligence We Can Trust*. Knopf Doubleday Publishing Group.

Gary Marcus, Francesca Rossi, and Manuela Veloso. 2016. Beyond the Turing Test. *AI Magazine*, 37(1):3–4.

Tom McCoy, Ellie Pavlick, and Tal Linzen. 2019. Right for the Wrong Reasons: Diagnosing Syntactic Heuristics in Natural Language Inference. In *Proceedings of the 57th Conference of the Association for Computational Linguistics, ACL 2019, Florence, Italy, July 28- August 2, 2019, Volume 1: Long Papers*, pages 3428–3448. Association for Computational Linguistics.

Tomás Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013. Efficient Estimation of Word Representations in Vector Space. In *1st International Conference on Learning Representations, ICLR 2013, Scottsdale, Arizona, USA, May 2-4, 2013, Workshop Track Proceedings*.

Sewon Min, Eric Wallace, Sameer Singh, Matt Gardner, Hannaneh Hajishirzi, and Luke Zettlemoyer. 2019. Compositional Questions Do Not Necessitate Multi-hop Reasoning. In *Proceedings of the 57th Conference of the Association for Computational Linguistics, ACL 2019, Florence, Italy, July 28- August 2, 2019, Volume 1: Long Papers*, pages 4249–4257. Association for Computational Linguistics.

Melanie Mitchell. 2021. Why AI is Harder than We Think. In *Proceedings of the Genetic and Evolutionary Computation Conference*, GECCO '21, page 3, New York, NY, USA. Association for Computing Machinery.

Christoph Molnar. 2022. *Interpretable Machine Learning*, 2 edition. LeanPub.

T Nurhuda, D Rusdiana, and W Setiawan. 2017. Analyzing Students' Level of Understanding on Kinetic Theory of Gases. *Journal of Physics: Conference Series*, 812:012105.

Sameer Pradhan, Alessandro Moschitti, Nianwen Xue, Hwee Tou Ng, Anders Björkelund, Olga Uryupina, Yuchen Zhang, and Zhi Zhong. 2013. Towards Robust Linguistic Analysis using OntoNotes. In *Proceedings of the Seventeenth Conference on Computational Natural Language Learning, CoNLL 2013, Sofia, Bulgaria, August 8-9, 2013*, pages 143–152. ACL.

Yada Pruksachatkun, Phil Yeres, Haokun Liu, Jason Phang, Phu Mon Htut, Alex Wang, Ian Tenney, and Samuel R. Bowman. 2020. jiant: A Software Toolkit for Research on General-Purpose Text Understanding Models. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics: System Demonstrations*, pages 109–117, Online. Association for Computational Linguistics.

Pranav Rajpurkar, Jian Zhang, Konstantin Lopyrev, and Percy Liang. 2016. SQuAD: 100, 000+ Questions for Machine Comprehension of Text. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing, EMNLP 2016, Austin, Texas, USA, November 1-4, 2016*, pages 2383–2392. The Association for Computational Linguistics.

Sagnik Ray Choudhury, Nikita Bhutani, and Isabelle Augenstein. 2022. Can Edge Probing Tests Reveal Linguistic Knowledge in QA Models? In *Proceedings of the 29th International Conference on Computational Linguistics*. International Committee on Computational Linguistics.

Nils Reimers and Iryna Gurevych. 2019. Sentence-BERT: Sentence Embeddings using Siamese BERT-Networks. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 3982–3992, Hong Kong, China. Association for Computational Linguistics.

Marco Túlio Ribeiro, Sameer Singh, and Carlos Guestrin. 2016. "Why Should I Trust You?": Explaining the Predictions of Any Classifier. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, San Francisco, CA, USA, August 13-17, 2016*, pages 1135–1144. ACM.

Anna Rogers, Matt Gardner, and Isabelle Augenstein. 2022. QA Dataset Explosion: A Taxonomy of NLP Resources for Question Answering and Reading Comprehension. *Computing Surveys (CSUR)*, to appear.

Anna Rogers, Olga Kovaleva, Matthew Downey, and Anna Rumshisky. 2020. Getting Closer to AI Complete Question Answering: A Set of Prerequisite Real Tasks. In *The Thirty-Fourth AAAI Conference on Artificial Intelligence, AAAI 2020, The Thirty-Second Innovative Applications of Artificial Intelligence Conference, IAAI 2020, The Tenth AAAI Symposium on Educational Advances in Artificial Intelligence, EAAI 2020, New York, NY, USA, February 7-12, 2020*, pages 8722–8731. AAAI Press.

Barbara Rychalska, Dominika Basaj, Anna Wróblewska, and Przemyslaw Biecek. 2018. Does It Care What You Asked? Understanding Importance of Verbs in Deep Learning QA System. In *Proceedings of the 2018 EMNLP Workshop BlackboxNLP: Analyzing and Interpreting Neural Networks for NLP*, pages 322–324. Association for Computational Linguistics.

Viktor Schlegel, Marco Valentino, André Freitas, Goran Nenadic, and Riza Batista-Navarro. 2020. A Framework for Evaluation of Machine Reading Comprehension Gold Standards. In *Proceedings of The 12th Language Resources and Evaluation Conference, LREC 2020, Marseille, France, May 11-16, 2020*, pages 5359–5369. European Language Resources Association.

Indira Sen, Mattia Samory, Fabian Flöck, Claudia Wagner, and Isabelle Augenstein. 2021. How Does Counterfactually Augmented Data Impact Models for Social Computing Constructs? In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 325–344, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Priyanka Sen and Amir Saffari. 2020. What do Models Learn from Question Answering Datasets? In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing, EMNLP 2020, Online, November 16-20, 2020*, pages 2429–2438. Association for Computational Linguistics.

Mattia Setzu, Riccardo Guidotti, Anna Monreale, Franco Turini, Dino Pedreschi, and Fosca Giannotti. 2021. GLocalX - From Local to Global Explanations of Black Box AI Models. *Artif. Intell.*, 294:103457.

Saku Sugawara, Kentaro Inui, Satoshi Sekine, and Akiko Aizawa. 2018. What Makes Reading Comprehension Questions Easier? In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing, Brussels, Belgium, October 31 - November 4, 2018*, pages 4208–4219. Association for Computational Linguistics.

Saku Sugawara, Pontus Stenetorp, Kentaro Inui, and Akiko Aizawa. 2020. Assessing the Benchmarking Capacity of Machine Reading Comprehension Datasets. In *The Thirty-Fourth AAAI Conference on Artificial Intelligence, AAAI 2020, The Thirty-Second Innovative Applications of Artificial Intelligence Conference, IAAI 2020, The Tenth AAAI Symposium on Educational Advances in Artificial Intelligence, EAAI 2020, New York, NY, USA, February 7-12, 2020*, pages 8918–8927. AAAI Press.

Mukund Sundararajan, Ankur Taly, and Qiqi Yan. 2017. Axiomatic Attribution for Deep Networks. In *Proceedings of the 34th International Conference on Machine Learning, ICML 2017, Sydney, NSW, Australia, 6-11 August 2017*, volume 70 of *Proceedings of Machine Learning Research*, pages 3319–3328. PMLR.

Iulia Turc, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. Well-Read Students Learn Better: The Impact of Student Initialization on Knowledge Distillation. *CoRR*, abs/1908.08962.

A. M. Turing. 1950. Computing Machinery and Intelligence. *Mind*, 59(236):433–460.

Alex Wang, Yada Pruksachatkun, Nikita Nangia, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel R. Bowman. 2019. SuperGLUE: A Stickier Benchmark for General-Purpose Language Understanding Systems. In *Advances in Neural Information Processing Systems 32: Annual Conference on Neural Information Processing Systems 2019, NeurIPS 2019, December 8-14, 2019, Vancouver, BC, Canada*, pages 3261–3275.

Alex Wang, Amapreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel R. Bowman. 2018. GLUE: A Multi-Task Benchmark and Analysis Platform for Natural Language Understanding. In *Proceedings of the 2018 EMNLP Workshop BlackboxNLP: Analyzing and Interpreting Neural Networks for NLP*, pages 353–355, Brussels, Belgium. Association for Computational Linguistics.

Mingzhu Wu, Nafise Sadat Moosavi, Dan Roth, and Iryna Gurevych. 2021. Coreference Reasoning in Machine Reading Comprehension. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing, ACL/IJCNLP 2021, (Volume 1: Long Papers), Virtual Event, August 1-6, 2021*, pages 5768–5781. Association for Computational Linguistics.

Zhilin Yang, Peng Qi, Saizheng Zhang, Yoshua Bengio, William W. Cohen, Ruslan Salakhutdinov, and Christopher D. Manning. 2018. HotpotQA: A Dataset for Diverse, Explainable Multi-hop Question Answering. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing, Brussels, Belgium, October 31 - November 4, 2018*, pages 2369–2380. Association for Computational Linguistics.

Xi Ye, Rohan Nair, and Greg Durrett. 2021. Connecting Attributions and QA Model Behavior on Realistic Counterfactuals. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing, EMNLP 2021, Virtual Event / Punta Cana, Dominican Republic, 7-11 November, 2021*, pages 5496–5512. Association for Computational Linguistics.

Matthew D. Zeiler. 2012. ADADELTA: An Adaptive Learning Rate Method. *CoRR*, abs/1212.5701.

Rolf A. Zwaan. 2016. Situation Models, Mental Simulations, and Abstract Concepts in Discourse Comprehension. *Psychonomic Bulletin & Review*, 23(4):1028–1034.

A Appendix

A.1 QA Model Training

For training the QA models in §3.1 The questions and contexts are concatenated, and a linear layer on top of the encoder is used to predict the probability of a context token i being the start ($P_{i,s}$) or end ($P_{i,e}$) of an answer. The score ($S_{i,j}$) for a span with start token i and end token j is computed as $P_{i,s} + P_{j,e}$. For all valid combination of i and j, the span with the highest score is chosen as the answer. A cross entropy loss between the actual and predicted start/end positions is minimized.

The models were trained for 10 epochs with a batch size of 16 using the Adam optimizer (Kingma and Ba, 2015) ($\beta_1 = 0.9, \beta_2 = 0.99, \epsilon =$ `1e-8`, `weight_decay` $= 0.01$) and gradient clipping. The learning rate (LR) was kept at `1e-05` with a linear warm-up schedule (staring LR=0). The models were evaluated on a subset of the validation data every 500 mini-batches with early stopping on 100 evaluations (Pruksachatkun et al., 2020). The LR and batch size was determined by a small grid search on the coreference dataset: LR=`{1e-05, 1e-04, 1e-03}`, batch size = $\{8, 16, 32\}$.

A.2 Antonym Replacements for CF Generation

The antonym replacements for the generalization test (§4.4) are described below:

- first → less recently
- older → less old, more junior, less mature, less grown-up
- earlier → subsequently, thereafter, less recently
- later → less recently
- younger → more old, less junior, more mature, more grown-up
- more recently → less recently, longer ago

A.3 Supervised Entity Type Predictor

Our goal is to build a classifier to predict the answer entity type from the question (§4.5). A sample data point is shown in Figure 5. The entity types are defined in the Ontonotes-5 dataset (Pradhan et al., 2013). The answer entity type is detected from the context using an off-the-shelf entity detector implemented in Spacy.[16] When the answer is not a named entity, or the entity detector fails to determine its type, that question is discarded.

Text: What is the full name of the person who is the television reporter that brings in a priest versed in Catholic exorcism rites?
Label: PER

Figure 5: A sample instance for answer entity type classifier.

The classification models are trained on the **training** portion of Quoref and SQuAD which is further divided into train/dev/test (70/20/10) split for training and evaluation. The distribution of the class labels is very skewed.

Models: We use two types of models: 1) a fine-tuned 12 layer 768 dimensional $\text{BERT}_{\text{base-cased}}$ model; and 2) a popular word convolutional model for sentence classification (Kim, 2014) using three parallel filters (size 3, 4, and 5) and 300 dimensional Google News Word2Vec representations (Mikolov et al., 2013).

BERT model: This model is trained for 5 epochs, with Adam optimizer (Kingma and Ba, 2015) with a weight decay of `1.0e-08` and a learning rate of `1.0e-05`. The sequence max length is kept at 128. We search for two hyperparameters: 1) number of epochs: 3-7, increasing by 1; and 2) learning rate: `1.0e-05`, `5.0e-05`, `1.0e-04`.

WordConv model: This model is trained for 40 epochs, with Adadelta optimizer (Zeiler, 2012) with a learning rate of `1.0e-05`. The sequence max length is again kept at 128.

For both models, accuracy was used as the early stopping metric. We minimized the cross entropy (CE) loss in general, but for the WordConv model, a weighted CE loss was also implemented to account for the training data class-imbalance in Quoref. That did not improve the results significantly and was not used in the $\text{BERT}_{\text{base-cased}}$ model. Table 7 shows the detailed results. Finally, we choose the fine-tuned $\text{BERT}_{\text{base-cased}}$ model as the entity detector as it performs the best. Ray Choudhury et al. (2022) also proposes a model to determine the answer entity type from a question, but the major difference is the label space. The model in Ray Choudhury et al. (2022) is trained to predict a label of "UNKNOWN_ENTITY" when the an-

[16] `https://spacy.io`

Dataset	Model	Accuracy	Macro F1
SQuAD	$\text{BERT}_{\text{base-cased}}$	76.4	56.2
	WordConv	72.4	44.9
Coref	$\text{BERT}_{\text{base-cased}}$	85.7	73.9
	WordConv	85.0	67.6
	WordConv Weighted BCE	85.3	69.7

Table 7: Models for supervised entity type selection. Green indicates the best results.

swer span is a) not a named entity or b) the entity detector can not find its type. However, an "UNKNOWN_ENTITY" label does not help the final algorithm (heuristic answer selection) to find the correct answer span. Therefore, our model never predicts this label, and consequently, has a better accuracy than Ray Choudhury et al. (2022). It potentially makes a mistake on the test data points that fall in the previous two categories, but the final algorithm is no worse than Ray Choudhury et al. (2022).

Hierarchical Attention Network for Explainable Depression Detection on Twitter Aided by Metaphor Concept Mappings

Sooji Han*, **Rui Mao***, **and Erik Cambria**
School of Computer Science and Engineering
Nanyang Technological University
{sooji.han,rui.mao,cambria}@ntu.edu.sg

Abstract

Automatic depression detection on Twitter can help individuals privately and conveniently understand their mental health status in the early stages before seeing mental health professionals. Existing black-box methods for depression detection largely focus on improving classification performance. However, explaining model decisions is imperative in health research because decision-making can often be high-stakes and life-and-death. In this work, we propose a novel explainable model for depression detection on Twitter. It comprises a novel encoder combining hierarchical attention mechanisms and feed-forward neural networks. To support psycholinguistic studies, our model leverages metaphorical concept mappings as input in order to also detect implicit manifestations of depression.

1 Introduction

Depression is a serious health and social issue that afflicts many individuals in modern society and its prevalence is predicted to increase globally. People with depression are likely to express their feelings and mental states over their social media before seeing health professionals (Guntuku et al., 2017; Ansari et al., 2022). An automatic, efficient approach for depression identification is imperative to recommend adequate treatment, achieving remission and preventing relapse. Recent studies on automatic depression detection on social media (Gui et al., 2019; Lin et al., 2020; Ji et al., 2022; Zogan et al., 2021) have largely focused on achieving higher detection accuracy. However, it is impossible to explain and interpret those black-box models that rely on state-of-the-art (SOTA) deep learning techniques. The recent development of explainable AI emphasizes that it is crucial for health professionals to fully comprehend, monitor and trust the AI decision-making mechanisms.

People suffering from depression often use metaphors to describe their emotions and the experience of living with mental illness (Coll-Florit et al., 2021; Roystonn et al., 2021). In psychotherapy, metaphors are a pivotal tool for helping people with depression better understand themselves and their problems and facilitating effective communication between therapists and patients (Kopp, 2013; Siegelman, 1993). This is because metaphorical expressions implicitly reflect people's different ways of understanding the same target. Metaphor is not only a linguistic phenomenon, but also a reflection of cognitive mappings of source and target concepts (Lakoff and Johnson, 1980). Analyzing metaphor concept mappings (MCMs) helps us understand the inner world of people with depression. Metaphoric expressions associated with depression have been widely studied in psychology, particularly as a form of case studies (Roystonn et al., 2021; Coll-Florit et al., 2021). To the best of our knowledge, however, there has not been an automatic method that leverages MCM features extracted from a large corpus for depression detection. We are motivated to bridge the gap and offer better insights into automatic depression detection on social media and conceptual metaphor understanding. Furthermore, we argue that psychological and psycholinguistic research communities can benefit from automated, explainable tools for studying the relationship between depression and metaphors.

In this work, we propose an explainable framework[1] for depression detection on Twitter, called *Hierarchical Attention Network (HAN)*. We propose a novel attention-based encoder which allows HAN to learn important inputs for user-level binary classification (i.e., depressed and non-depressed users). To further improve the interpretability of depression detection, we introduce MCMs as an additional feature into the model.

∗ These authors contributed equally.

[1]The source code is available at https://github.com/senticnet/depression-detection

Proceedings of the 29th International Conference on Computational Linguistics, pages 94–104
October 12–17, 2022.

Health professionals and potential patients can use learned features (i.e., characteristics of depressive tweets and MCMs) as justification. We evaluate our model on a publicly available Twitter depression detection dataset (Shen et al., 2017) and show that HAN achieves the SOTA performance. It outperforms the strongest baseline (Zogan et al., 2021) by increasing an F1 score by 6.0% on average. Additionally, our newly proposed encoder outperforms several classical encoders. In particular, HAN improves LSTM (Hochreiter and Schmidhuber, 1997) (the most competitive benchmark encoder for our task) by 1.9% on a validation set with a quarter of the number of parameters of LSTM. Finally, we visualize and analyze examples of attention weights learned by HAN to demonstrate its explainability.

The main contributions of this work can be summarized as follows: (1) We propose an explainable model for depression detection on Twitter. Unlike most SOTA methods employing attention mechanisms at word level (Vaswani et al., 2017; Liu et al., 2021), our model employs context-level attention mechanisms to identify the relative importance of certain tweets and metaphors, which is crucial for filtering out less significant information in the final representation of contexts and justifying the outputs of the model. (2) We introduce MCMs as a feature to improve explainability and performance. This also helps a better understanding of the cognition of depressive individuals. (3) We demonstrate that HAN achieves outstanding performance and produces accurate and explainable results with a smaller number of training parameters than classical encoders via extensive experiments.

2 Related Work

Traditional studies on depression focus on social, psychological and biological factors, which are not often readily available. This paper mainly focuses on social media texts and machine-learning-based depression detection. Several studies in psychology have reported that conceptual metaphors are used to express and understand the experience of depression, but they are often used unconsciously and pass unnoticed. Research into metaphors can help better understand individuals with depression.

Depression Detection on Social Media. Zogan et al. (2021) proposed a model combining CNNs (LeCun et al., 1989) and BiGRUs (Deng et al., 2019) for learning users' behavior and textual contents. For user behavior modeling, manually curated features, which are associated with emotions, domains, topics and social media metadata, were employed. Some research exploited sentiment analysis techniques for depression detection. Rao et al. (2020) proposed a hierarchical architecture leveraging gated units and CNNs to learn textual contents of social media posts and users' emotional states expressed in posts. Aragon et al. (2021) proposed an emotion-aware SVM-based model which learns emotional dynamics expressed in social media posts. Chiong et al. (2021) proposed 90 features, based on sentiment lexicons and textual contents and used them as input to depression detection classifiers. A recent trend is to exploit multimodal learning frameworks for depression detection (Gui et al., 2019; Chiu et al., 2021; Lin et al., 2020; Yang et al., 2018). Gui et al. (2019) proposed a multimodal multi-agent reinforcement learning model incorporating BiGRU and VGGNet (Simonyan and Zisserman, 2014) to learn texts and images posted by users on Twitter, respectively. Chiu et al. (2021) proposed a multimodal BiLSTM-based (Schuster and Paliwal, 1997) architecture jointly learning texts, images and temporal behaviors (i.e., time intervals between posts) on Instagram. Lin et al. (2020) proposed a multimodal model comprising a CNN and a BERT. It jointly learns representations of images and texts and fused them using a low-rank multimodal fusion method. Zhang et al. (2021) proposed a model combining BiLSTM and CNN based on metaphor features and text. However, their metaphor features are shallow, e.g., Part-of-Speech (PoS) tags and the number of metaphors.

Metaphor Understanding. Traditional metaphor studies on depression were mainly based on qualitative analysis and case studies (Roystonn et al., 2021; Coll-Florit et al., 2021). This is because of a lack of automatic tools that help psycholinguistic researchers parse and analyze metaphorical expressions from large corpora. Recently, automatic metaphor processing has achieved significant developments. Metaphors can be identified with sequence-tagging-based models (Mao et al., 2019; Choi et al., 2021; Chen et al., 2021; Mao and Li, 2021). Then, the identified metaphors can be interpreted by linguistic meanings (Bollegala and Shutova, 2013; Mao et al., 2018, 2021) or concept mappings (Mason, 2004; Shutova et al., 2017; Ge et al., 2022). Linguistic metaphor interpretation focuses on paraphrasing metaphors into their lit-

eral counterparts. For example, Mao et al. (2022a) proposed a system for metaphor identification and interpretation, called MetaPro. It can be used as a text pre-processing technique to achieve metaphor paraphrases from end to end. Thus, NLP techniques for downstream tasks, such as sentiment analysis (Mao et al., 2022a) or machine translation (Mao et al., 2018), can achieve better performance on the effectiveness of metaphor paraphrasing. However, Lakoff and Johnson (1980) argued that metaphor is not only a linguistic phenomenon, but also a reflection of humans' cognition. Given "this is the *core*[2] of the matter", *core* implies "importance" (target) is "interiority" (source) (Lakoff et al., 1991). Thus, one can analyze the inner world of depressed people from their metaphoric expressions and the associated concept mappings, e.g., IMPORTANCE IS INTERIORITY.

In this paper, we identify several limitations of existing works on depression detection. Existing methods have largely focused on improving classification performance by using advanced encoders, features and deep architectures, while leaving model outputs inexplicable. The majority of SOTA methods are limited to textual contents of posts or rely on shallow features based on social media metadata. To our best knowledge, it is the first work incorporating MCMs into machine-learning-based depression detection on social media. Additionally, our model comprises context-level explainable encoders while word-level attention mechanisms have been widely employed in SOTA methods. This helps better understand how certain tweets and MCMs are used by depressed individuals, thereby justifying model predictions.

3 Methodology

3.1 Problem statement

In our task, a user (u_k) is represented as a set of tweets ($\mathbb{X}_k$) and a set of the associated MCMs ($\mathbb{M}_k$) in the tweets. Therefore, a set of users is denoted by $\mathbb{U} = \{u_1, \cdots, u_i\}$, where each user $u_k = [\mathbb{X}_k, \mathbb{M}_k]$. A set of a user's **tweet contents** is denoted by $\mathbb{X}_k = \{x_{k,1}, \cdots, x_{k,n}\}$ which contains n tweets. x is a tweet represented as a sequence of words. A set of a user's **MCMs** is denoted by $\mathbb{M}_k = \{m_{k,1}, \cdots, m_{k,s}\}$ which contains s mappings. m is an MCM that is represented as a sequence of "A IS B", where A is a target concept, B is a source concept, and IS a relation mapping A to B. An example is "IMPORTANCE IS INTERIORITY". The task is to predict the most probable label ($\hat{y}_k$) for a user u_k, where $\hat{y}_k \in \{0, 1\}$. $\hat{y}_k = 1$ if u_k is a depressed user, $\hat{y}_k = 0$ otherwise. y_k denotes a ground truth label.

[2]A metaphor is in italics.

3.2 Model Architecture

The overall architecture of HAN is shown in Figure 1. Given a set of tweets ($\mathbb{X}$) of a user, we first obtain embeddings of all tweets in $\mathbb{X}$ using a pre-trained language model. BERT-base-uncased (Devlin et al., 2019) is used to be in line with our strongest baseline (Zogan et al., 2021). Special tokens [CLS] and [SEP] are added at the beginning and at the end of each tweet x_ϵ, respectively. The padded sequence "[CLS] x_ϵ [SEP]" is fed into BERT. The vector representation at the [CLS] position of BERT output is used as the embedding of x_ϵ. We obtain an embedding matrix of all tweets in $\mathbb{X}$, denoted by T_0. Formally,

$$T_0 = BERT(\mathbb{X}). \tag{1}$$

Similarly, we obtain an embedding matrix of all MCMs in $\mathbb{M}$, denoted by C_0. Formally,

$$C_0 = BERT(\mathbb{M}). \tag{2}$$

Details about the acquisition of $\mathbb{M}$ are described in Section 3.3.

HAN consists of l attention-based encoder layers. The ith encoder layer is defined as $HAN_i(\cdot)$ (see Section 3.4 for details), where $i \in \{1, 2, \cdots l\}$. Given a query vector and a key matrix, $HAN_i(\cdot)$ yields an updated query vector and key matrix. Thus, given the query vector (v^t_{i-1}) of tweets (t) and the tweet embedding (key) matrix T_{i-1} in the $(i-1)^{th}$ layer, the updated v^t_i and T_i are given by

$$v^t_i, T_i = HAN^t_i(v^t_{i-1}, T_{i-1}). \tag{3}$$

Similarly, given the query vector (v^c_{i-1}) of MCM (c) and the MCM embedding (key) matrix C_{i-1} in the $(i-1)^{th}$ layer, the updated v^c_i and C_i are given by

$$v^c_i, C_i = HAN^c_i(v^c_{i-1}, C_{i-1}). \tag{4}$$

For the first layers of the first training step, the inputs v^t_0 and v^c_0 (trainable parameters) are randomly initialized. For the other training steps, v^t_0 and v^c_0 are values learned in the previous step.

Next, the output of the last layer of the tweet encoder (v^t_l) and that of the MCM encoder (v^c_l) are concatenated ($\oplus$). The concatenated representation is fed to three feed-forward

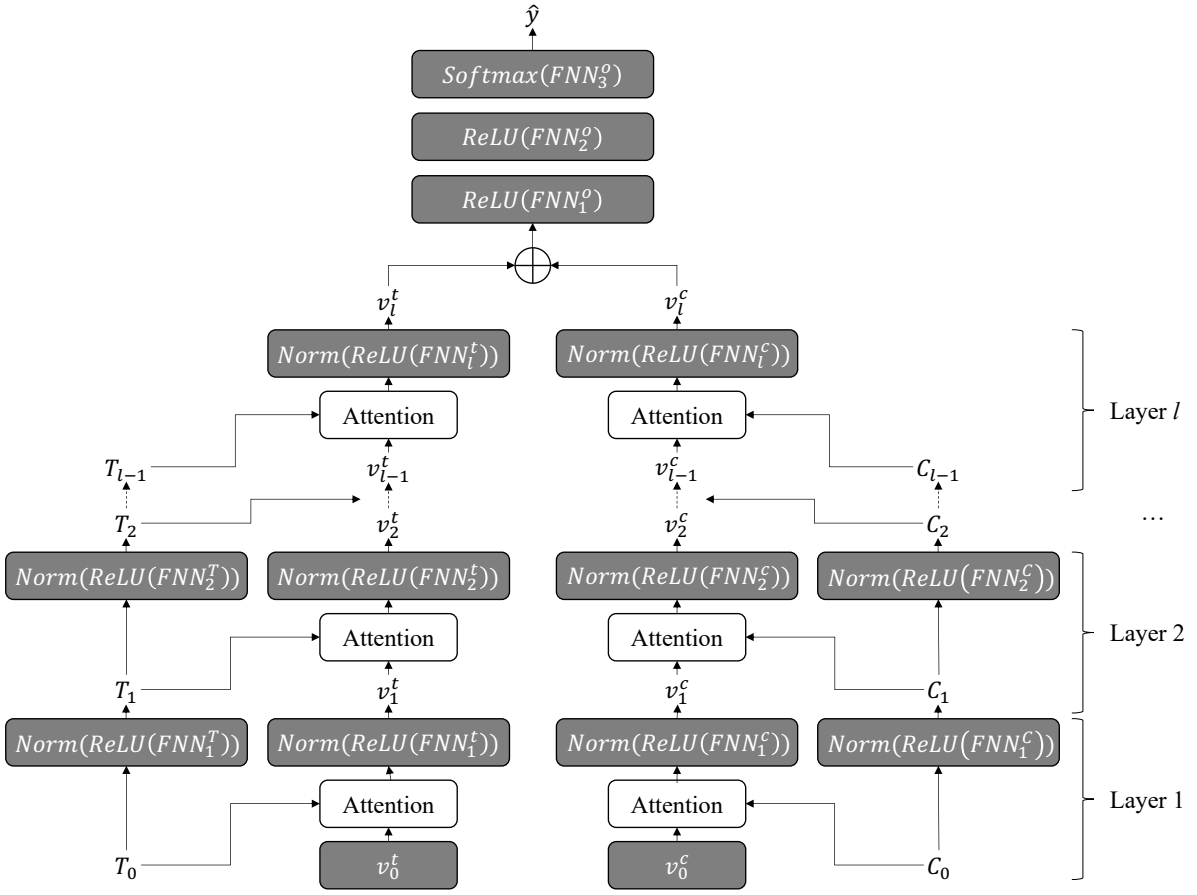

Figure 1: Hierarchical Attention Network. Grey boxes denote computational layers with trainable parameters. Plain text denotes input and output. T and C denote tweet and MCM embeddings, respectively.

neural networks (FNNs), denoted by $FNN^o(\cdot)$. The first two $FNN^o(\cdot)$ are activated by ReLU ($ReLU(\cdot)$) (Agarap, 2018). The last FNN, $FNN_3^o(\cdot)$, is activated by the Softmax. We do not change the size of the hidden state given by the outputs of the first two FNN layers. The last FNN layer projects the hidden state into a vector of the label size. Then, the probability of a predicted label ($\hat{y}$) is given by

$$h = ReLU(FNN^o(v_l^t \oplus v_l^c))_{\times 2} \quad (5)$$

$$\hat{y} = Softmax(FNN_3^o(h))), \quad (6)$$

where h is the hidden state after the first two ($\times 2$) FNNs. Cross-entropy loss is used to optimize the parameters in the model and is given by

$$\mathcal{L} = CrossEntropy(\hat{y}, y). \quad (7)$$

3.3 Concept mapping acquisition

Concept mapping acquisition process consists of three components: **a) metaphor identification** ($MI(\cdot)$) (Mao and Li, 2021), **b) metaphor paraphrasing** ($MP(\cdot)$) (Mao et al., 2021) and **c) concept mapping generation** ($CG(\cdot)$) (Ge et al., 2022). These algorithms are used because they enable the end-to-end acquisition of MCM features without pre-processing and domain-specific knowledge. Here, we briefly introduce their algorithms, inputs and outputs. For the details, please refer to the original papers. Given a tweet ($\mathbb{x}_\epsilon$) comprising g tokens τ, i.e., $\mathbb{x}_\epsilon = \{\tau_{\epsilon,1}, \tau_{\epsilon,2}, \cdots, \tau_{\epsilon,g}\}$. The **metaphor identification** module (MI) is a multi-task-learning-based sequence tagging model, yielding a metaphor label sequence (r_ϵ) and a PoS label sequence (ρ_ϵ) defined by

$$r_\epsilon, \rho_\epsilon = MI(\mathbb{x}_\epsilon), \quad (8)$$

where $r_\epsilon = \{r_{\epsilon,1}, r_{\epsilon,2}, \cdots, r_{\epsilon,g}\}$ and $\rho_\epsilon = \{\rho_{\epsilon,1}, \rho_{\epsilon,2}, \cdots, \rho_{\epsilon,g}\}$. $r_{\epsilon,j} \in \{\text{metaphor}, \text{literal}\}$ and $\rho_{\epsilon,j}$ is a Universal-Dependency-scheme-based PoS label, where $j \in \{1, 2, \cdots, g\}$ denotes the index of a token in $\mathbb{x}_\epsilon$. To boost model performance, Mao and Li (2021) proposed a Gated Bridging Mechanism for soft-parameter sharing between the metaphor identification and PoS tagging tasks.

Next, given an identified metaphoric open-class word[3] $\tau_{\epsilon,j}$ (i.e., one of verbs, nouns, adjectives and adverbs), the **metaphor paraphrasing** module first lemmatizes $\tau_{\epsilon,j}$ as $\tau^{\iota}_{\epsilon,j}$. Then, a pre-trained language model is used to select the best fit word ($\omega_{\epsilon,j}$) from a candidate set that consists of hypernyms and synonyms of $\tau^{\iota}_{\epsilon,j}$ in WordNet (Fellbaum, 1998) and their inflections with the same PoS. The best fit word denotes a candidate word that appears in the context and has the highest probability.

A probability is given by a masked word prediction of the pre-trained language model, which has been widely used in prompt-based zero-short learning tasks (Mao et al., 2022b). The best fit word $\omega_{\epsilon,j}$ is lemmatized as $\omega^{\iota}_{\epsilon,j}$, which is considered the lemma of the paraphrased metaphor $\tau_{\epsilon,j}$ in the context of x_{ϵ}. The above process is defined by

$$\omega^{\iota}_{\epsilon,j} = MP(\tau_{\epsilon,j}, \rho_{\epsilon,j}). \tag{9}$$

Finally, the **concept mapping generation** module abstracts the source concept ($\mathrm{A}_{\epsilon,j}$) from $\tau^{\iota}_{\epsilon,j}$ and the target concept ($\mathrm{B}_{\epsilon,j}$) from $\omega^{\iota}_{\epsilon,j}$. Formally,

$$\mathrm{A}_{\epsilon,j} = CG(\tau^{\iota}_{\epsilon,j}), \tag{10}$$

$$\mathrm{B}_{\epsilon,j} = CG(\omega^{\iota}_{\epsilon,j}). \tag{11}$$

$CG(\cdot)$ is a knee algorithm (Satopaa et al., 2011) and a WordNet-based conceptualization method, proposed by Ge et al. (2022). It abstracts a word into a concept by looking up a hypernym that can cover the major senses of a word. After obtaining $\mathrm{A}_{\epsilon,j}$ and $\mathrm{B}_{\epsilon,j}$, the concept mapping is defined as

$$\mathrm{MCM}_{\epsilon,j} = \mathrm{B}_{\epsilon,j} \text{ IS } \mathrm{A}_{\epsilon,j}. \tag{12}$$

Ge et al. (2022) argued that Lakoff et al. (1991) summarized concept mappings with different patterns due to the subjectivity of annotators. We take the concept mappings given by Eq. 12, which follows one of the concept mapping principles of Lakoff et al. (1991) (see Section 2). We obtain all concept mappings in $\mathbb{x}_{\epsilon}$. If no metaphor is detected in $\mathbb{x}_{\epsilon}$, concept mapping is none for such a tweet. All concept mappings from all tweets of each user are collected, forming an MCM feature set ($\mathbb{M}$) for depression detection.

3.4 Hierarchical attention network encoder

The HAN encoder ($HAN(\cdot)$) is based on scaled dot-product attention and FNNs. Attention mechanisms enable the model to identify input features (i.e., tweets and MCMs) that are highly significant and useful for depression detection, thereby enhancing model explainability. FNNs allow feature embeddings to better fit the task via multiple non-linear projections. Unlike self-attention (Vaswani et al., 2017), the feature information of our encoder is not shared with each other within each feature set, i.e., T_i and C_i in Eqs. 3 and 4. Thus, features fed to the last encoder layer (i.e., T_{l-1} and C_{l-1}) represent the features of individual inputs even after several non-linear projections. Important features are learned by query vectors. These are the main differences between our explainable encoder and other classical black-box-like encoders, e.g., LSTM, BiLSTM, GRU (Cho et al., 2014), BiGRU and Transformer (Vaswani et al., 2017), hidden states of which cannot be easily disentangled after encoding.

Given a query vector ($q_{i-1} \in \mathbb{R}^{1\times d}$) and a key matrix ($K_{i-1} \in \mathbb{R}^{o\times d}$) in the $(i-1)^{th}$ layer, where d denotes an embedding size and o denotes the number of input features, attention weights ($w_i \in \mathbb{R}^{1\times o}$) in the ith layer are given by

$$w_i = Softmax\left(\frac{q_{i-1} \otimes K^{\intercal}_{i-1}}{\sqrt{d}}\right), \tag{13}$$

where $\otimes$ denotes matrix product. The query vector ($q_i \in \mathbb{R}^{1\times d}$) in the ith layer is given by the weighted sum of the vectors in K_{i-1} and a non-linear projection. Formally,

$$q_i = LN(ReLU(FNN^{query}_i(w_i \otimes K_{i-1}))), \tag{14}$$

where $LN(\cdot)$ denotes layer normalization (Ba et al., 2016). The key matrix ($K_i \in \mathbb{R}^{o\times d}$) in the ith layer is defined by

$$K_i = LN(ReLU(FNN^{key}_i(K_{i-1}))). \tag{15}$$

The input and output of the HAN encoder have the same size. For the tweet content encoder, q, K and o denote a tweet query vector (v^t), a tweet embedding matrix (T) and the number of tweets (n), respectively. For the MCM encoder, q, K, and o denote an MCM query vector (v^c), an MCM embedding matrix (C) and the number of MCMs (s), respectively.

We use the attention weights from the last (the lth) encoder layer as the final representation of tweet contents and MCMs. w_l shows important inputs that have higher attention weights for the

[3]Closed-class words are not paraphrased because they do not convey much semantic information in their context.

Dataset	Total # of tweets		Mean # of tweets per user	
	Positive	Negative	Positive	Negative
D1	156,013	153,328	72	75
D2	151,538	119,188	71	58
D3	142,057	118,611	66	58
D4	143,725	124,925	66	61
D5	148,039	134,700	69	66

Table 1: Statistics of the five randomly sampled datasets. The number of positive users and that of negative users are 2,159 and 2,049 for all the datasets.

Model	P	R	F1	Acc.
Gui et al. (2019)	0.900	0.901	0.900	0.900
Lin et al. (2020)	0.903	0.870	0.886	0.884
Zogan et al. (2021)	0.909	0.904	0.912	0.901
$\text{HAN}_{\text{ours}}\text{-Avg}_{\text{D1-D5}}$	0.975	0.969	**0.972**	**0.971**
D1	0.981	0.965	0.973	0.973
D2	0.988	0.956	0.972	0.971
D3	0.972	0.972	0.972	0.971
D4	0.968	0.970	0.969	0.968
D5	0.964	0.981	0.972	0.971

Table 2: Depression detection results. Our model result is averaged over the five testing sets (D1-D5).

depression status prediction of a user. Analysis results of attention weights to demonstrate model explainability are described in Section 5.5.

4 Experiments

4.1 Datasets and pre-processing

Table 1 presents the statistics of the datasets used in our experiments. We use a publicly available Twitter dataset, called MDL (Shen et al., 2017), which was designed for depression detection. In this dataset, Twitter users, who have posted tweets containing pre-defined patterns (i.e., I'm/I was/I am/I've been diagnosed depression), were labeled as depressive (i.e., positive). Those who never posted any tweet containing the term "depress" were labeled as users not suffering from depression (i.e., negative). Due to the updates of MDL over time, its statistics varies across existing works which used it for their experiments (Gui et al., 2019; Lin et al., 2020; Zogan et al., 2021). We argue that the model of Zogan et al. (2021) is the most comparable to our model as regards architecture and features, e.g., employing two independent encoders to encode textual features from multiple sources. To make our results comparable with this work, 2,159 positive and 2,049 negative users are randomly sampled from the latest version of MDL. For a fair comparison, we generate five datasets with randomly selected users. 60%, 20% and 20% of the full dataset are used for train, validation and test sets, respectively, which results in 2,524 users in a train set and 842 users in each of validation and test sets. We exclude tweets with less than 4 tokens because they are less informative. URLs and mentions are removed because they are likely to introduce noise (Gao et al., 2020).

4.2 Baselines

We compare our model with three depression detection baselines.

• Gui et al. (2019): A reinforcement-learning-based model based on cooperative multi-agent policy gradients. Tweet texts and images are encoded using GRUs and VGGNets, respectively.

• Lin et al. (2020): A model comprising a CNN and a BERT for learning images and texts, respectively. The final representation of inputs is obtained via low-rank multimodal fusion.

• Zogan et al. (2021): A model jointly learning tweet texts and user behavior using CNNs and Bi-GRUs. BERT-base and BART-large are used for tweet text modeling.

We do not benchmark the work by Zhang et al. (2021) because their model was designed for classifying different types of mental disorders. Besides, their model (i.e., CNNs+BiLSTMs) is similar to the architecture proposed by Zogan et al. (2021).

4.3 Setups

We employ two HAN encoder layers (i.e., $l = 2$). The maximum input length (i.e., the maximum numbers of tweets and MCMs per user) is set to 200. The batch size (i.e., the number of users per batch) is 64. Dropout rates for query vectors and key matrices are set to 0.2. The learning rate and weight decay of the Adam optimizer (Kingma and Ba, 2015) are set to 1e-4 and 1e-5, respectively. BERT-base-uncased is used to obtain tweet and MCM embeddings. The model is trained with a GeForce GTX 1080 Ti GPU with CUDA 9.2 (NVIDIA et al., 2020) and PyTorch 1.7.1 (Paszke et al., 2019). Following our baselines, four performance metrics are adopted in our experiments: accuracy (Acc.), precision (P), recall (R) and F1 score (F1). P, R and F1 are computed with respect to the positive class, i.e., depressive.

Model	F1 on MDL-validation						F1 on IMDL-validation					
	D1	D2	D3	D4	D5	Avg	D1	D2	D3	D4	D5	Avg
HAN	0.985	0.960	0.971	0.976	0.975	**0.973**	0.939	0.911	0.933	0.927	0.931	**0.928**
HAN-MCM	0.972	0.947	0.963	0.967	0.962	**0.962**	0.914	0.897	0.914	0.905	0.918	**0.909**
Δ	0.013	0.013	0.008	0.009	0.013	**0.011**	0.025	0.014	0.019	0.022	0.014	**0.019**

Table 3: Ablation study results on validation sets, measured by F1 score. Δ is defined by $F1_{\text{HAN}} - F1_{\text{HAN-MCM}}$.

5 Results

5.1 Classification performance

As shown in Table 2, our proposed model advances SOTA performance for all of the five datasets. HAN achieves an average F1 score of 97.2% and an accuracy of 97.1%. The comparison results show that HAN yields the increases of 6.0% and 7.0% in F1 score and accuracy over the strongest baseline model (Zogan et al., 2021), respectively. We observe that performance is almost identical for different randomly sampled datasets (i.e., D1-5), which shows that HAN is robust to different characteristics of users on Twitter.

5.2 Ablation study

A set of exploratory experiments is conducted to study the relative contribution of MCMs in our model. To this end, we generate a variation of MDL, called Implicit Twitter Depression Data (IMDL), by removing explicit linguistic cues for depression (i.e., "I'm/I was/I am/I've been diagnosed depression" and words containing "depress", "anxiety", "bipolar" and "disorder") from all tweets. We evaluate our full model (HAN) and HAN without the MCM encoder, called HAN-MCM, using both MDL and IMDL. Table 3 shows average F1 scores achieved with the validation sets. Overall, HAN and HAN-MCM achieve higher performance on MDL than on IMDL in terms of F1 score. This verifies our hypothesis that the removal of explicit linguistic cues for depression from tweets makes the task more difficult. The ablation study of the internal baseline model (i.e., HAN-MCM) proves that MCMs can provide additional information effective in identifying depressive users.

For MDL, HAN outperforms HAN-MCM by 1.1%. It is worth noting that the performance difference is slightly larger for IMDL (1.9%), which indicates that MCMs can provide effective complementary evidence when no explicit cues for depression exist in tweets. The above experiments on both datasets demonstrate the usefulness of understanding MCMs in identifying depressive users.

# of HAN layers	1	2	4	8
MDL D1 validation	0.542	**0.985**	0.983	0.979

Table 4: F1 scores for different numbers of encoder layers.

5.3 Hyperparameter analysis

The major hyperparameter of HAN is the number of encoder layers (i.e., l). We experiment with different values (i.e., 1, 2, 4 and 8) for the MDL D1 validation set. Table 4 shows that the best F1 score is achieved when l is set to 2. Using more than two layers does not reap benefits of an increase in model performance.

5.4 Encoder benchmarking

To prove the effectiveness of our HAN encoder, we compare it with six widely used encoders: LSTM, BiLSTM, GRU, BiGRU, TF-first and HAN-TF. HAN encoders are replaced with each of them in our framework. For LSTM and GRU, the final representation of input features is the hidden state of the last token. For BiLSTM and BiGRU, the concatenation of the forward and backward hidden states of the last and first tokens is used. TF-first and HAN-TF are encoders based on Transformer. In TF-first, Transformer is used as an encoder.

The hidden state of the first input instance[4] (a tweet or an MCM) of each user is used by Eq. 5. In HAN-TF, we use Transformer to replace the input matrix projection layer (i.e., FNN in Eq. 15) of HAN encoders. The size of the input and output hidden states of each encoder is set to 768, which is in line with that of BERT-base-uncased. The size of FNNs and the number of heads in Transformer are 768 and 8, respectively. The other hyperparameters remain the same. CNNs are not used as a baseline because they need to be used with other encoders to learn context dependencies (Wang et al., 2016; Rhanoui et al., 2019).

[4]We experimented with different fusion methods (e.g., summation, average and linear transformation of concatenated representations) and found that using the hidden state of the first input instance works best.

	LSTM	BiLSTM	GRU	BiGRU	TF-first	HAN-TF	HAN
F1 on MDL D1 val. ↑	0.966	0.965	0.961	0.959	0.898	0.976	**0.985**
# of param. per layer ↓	4.72M	9.45M	3.54M	7.09M	3.55M	4.14M	**1.18M**

Table 5: Comparison results of different encoder layers. ↑ denotes that the higher the value is, the better the model is. ↓ denotes that the lower the value is, the better the model is.

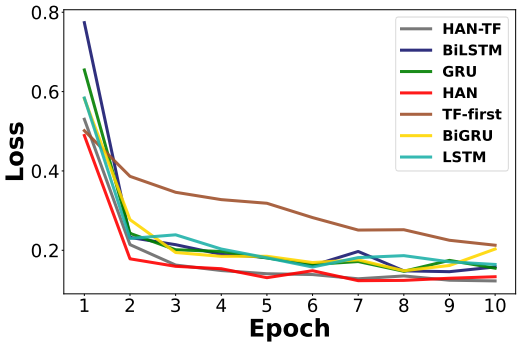

Figure 2: Training loss curves for different encoders obtained using the MDL D1.

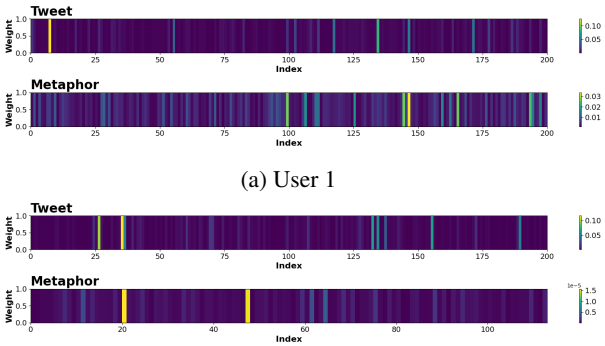

(a) User 1

(b) User 2

Figure 3: Visualization of attention weights for two depressed users. The lighter the color bar of an instance (tweet or MCM) is, the higher its attention weight is.

Table 5 shows F1 scores achieved with the MDL D1 validation set and the number of parameters per encoder layer. HAN encoder achieves better results than all the baseline encoders on our task in terms of F1 score. HAN and HAN-TF outperform LSTM by 1.9% and 1.0%, respectively. Although HAN and HAN-TF achieve comparable performance, it is worth noting that using Transformer instead of FNNs in the HAN encoder significantly increases the number of parameters (+2.96M). The number of parameters of the HAN encoder is the smallest among all the encoders. The parameter size of HAN is just a third of that of the second smallest encoder (GRU).

Figure 2 shows training loss curves for different encoders plotted using the MDL D1. HAN (the red line) converges faster than the other encoders. Overall, the experiments on different encoders prove that HAN has advantages over the others in terms of effectiveness and efficiency.

5.5 Explainability demonstration

Figure 3 visualizes the attention weights (w_l given by Eq. 13) for tweets posted by two users with depression and MCMs in their tweets. As shown in the figure, HAN can selectively focus on the most important and useful tweets and metaphors by progressively refining feature maps. Attention weights are useful for justifying the decision-making mechanism of HAN because they quantitatively describe how much each tweet and MCM contributes to a predicted label (see Eq. 14). Higher attention weights denote greater utility of tweets and MCMs in detecting depression. Table 6 shows the top 5 tweets and MCMs, ranked according to the attention weights learned during training, for two sample users. User 1 tends to use negative expressions to describe personal feelings, state and emotions such as "bad", "sick", "hate" and "nervous".

The user also uses positive expressions, such as "I'll do great" and "everything will be ok". Such

User	Tweet	Metaphor
1	1. I hate how I can't tell if I have allergies or I'm getting sick.	1. LEVEL IS IMPORTANCE
	2. get better, I love you	2. PERSON IS EXTREMITY
	3. I'm slightly allergic to cats but I still have them and I don't CARE IF I SNEEZE	3. SITUATION IS HAPPENING
	4. I'm having a bad night	4. ATHLETE IS AREA
	5. So I'm so nervous for my MAC interview tomorrow but I know I'll do great. Everything will be okay	5. MORPHEME IS EXTREMITY
2	1. Today is not a good day: Driver, teen shot to death after vehicle hits and kills -year-old	1. CONCERN IS STATE
	2. Autistic th Grader Assaulted by School Cop, Now He is a Convicted Felon	2. POSITION IS DISAPPEARANCE
	3. Thank you Father, GM FB! I gotta start taking My butt to bed at night, woke late again	3. LEVEL IS IMPORTANCE
	4. Cellphone Video Surfaces Showing Moments After Police Shot -Year-Old Boy in the Back	4. FEELING IS ILL_HEALTH
	5. Freddie Gray dies one week after Baltimore arrest	5. ARTIFACT IS SUPPORT

Table 6: The top 5 tweets and metaphors, selected based on attention weights, for two sample users.

tweets, however, tend to express self-soothing for negative events. User 2 tends to repost tragic news and add personal comments. The two depressed users show different behaviors on social media, e.g., self-soothing and quoting tragic news. The listed MCMs in Table 6 show that both sample users have the same MCM in their tweets, i.e., LEVEL IS IMPORTANCE. The conceptual projection from LEVEL to IMPORTANCE may exacerbate depression because LEVEL simply refers to "a position on a scale of intensity or amount or quality", whereas IMPORTANCE normally refers to a subjective feeling about "the worthy of note" (Fellbaum, 1998).

The imageability of IMPORTANCE may increase stress and anxiety, and thus arouse more depressive feelings (Vedhara et al., 2003). For example, there is a tweet saying that "If a transgender student is bullied, they are put at a *greater* risk of suicide" posted by a depressed user in the dataset (Shen et al., 2017). In this tweet, "*greater*" is metaphorical. Its contextual meaning refers to a higher risk. "High" is one of the manifestations of the target concept LEVEL. However, the literal imageability of "great" likely refers to the source concept IMPORTANCE, e.g., "a great work of art" (Fellbaum, 1998). Thus, the metaphorical expression in the sample tweet also implies that the "risk of suicide" is high and important, which probably increases the subject's nervousness because of their perception about the importance of the risk. We also find that a metaphoric term "*great*" is common in the MCM LEVEL IS IMPORTANCE and its associated tweets posted by depressed users.

This case study demonstrates that we can further discover common MCMs and metaphorical language patterns among depressed individuals using our proposed model. In general, we argue that HAN is potentially useful for identifying depressed individuals and analyzing different types of such individuals, their cognition and risk factors.

6 Ethical Considerations

This research work was conducted based on a public dataset published by Shen et al. (2017). We solely used textual content for concept mapping acquisition, training, and evaluating the model. We did not leverage any information related to user profiles. We oppose the use of our model in any breach of data security, privacy protection, and ethics.

7 Conclusion

While most deep learning architectures for depression detection left the impact of different input features on model performance inexplicable, our work attempted to interpret what was going on in the model and justify model predictions. We proposed an attention-based encoder to better understand decision-making process for depression detection. We introduced novel metaphor concept mapping features into our model to investigate how depressed people describe their emotions and experiences. Our extensive experiments and comparative evaluations showed that our model could achieve SOTA performance. An ablation study proved the advantage of utilizing metaphors in depression detection. We argue that a better understanding of metaphors associated with depression can enhance interpretability and help health professionals provide tailored, timely therapy to patients. In future research, we plan to conduct a large-scale study to categorize different characteristics of depression using users' metaphorical and cognitive expressions.

Acknowledgments

We appreciate Rekha Samrajyutha Sajja and Shreyas Nagesh for evaluating metaphor concept mappings. We also appreciate reviewers and Frank Guerin for the constructive comments.

References

Abien Fred Agarap. 2018. Deep learning using rectified linear units (ReLU). *arXiv preprint arXiv:1803.08375*.

Luna Ansari, Shaoxiong Ji, Qian Chen, and Erik Cambria. 2022. Ensemble hybrid learning methods for automated depression detection. *IEEE Transactions on Computational Social Systems*, pages 1–9.

Mario Ezra Aragon, Adrian Pastor Lopez-Monroy, Luis-Carlos Gonzalez Gonzalez-Gurrola, and Manuel Montes. 2021. Detecting mental disorders in social media through emotional patterns-the case of anorexia and depression. *IEEE Transactions on Affective Computing*.

Jimmy Lei Ba, Jamie Ryan Kiros, and Geoffrey E Hinton. 2016. Layer normalization. *arXiv preprint arXiv:1607.06450*.

D Bollegala and E Shutova. 2013. Metaphor interpretation using paraphrases extracted from the web. *PLoS ONE*, 8(9).

Xin Chen, Zhen Hai, Suge Wang, Deyu Li, Chao Wang, and Huanbo Luan. 2021. Metaphor identification: A contextual inconsistency based neural sequence labeling approach. *Neurocomputing*, 428:268–279.

Raymond Chiong, Gregorious Satia Budhi, and Sandeep Dhakal. 2021. Combining sentiment lexicons and content-based features for depression detection. *IEEE Intelligent Systems*, 36(6):99–105.

Chun Yueh Chiu, Hsien Yuan Lane, Jia Ling Koh, and Arbee LP Chen. 2021. Multimodal depression detection on instagram considering time interval of posts. *Journal of Intelligent Information Systems*, 56(1):25–47.

Kyunghyun Cho, Bart van Merriënboer, Caglar Gulcehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. 2014. Learning phrase representations using RNN encoder–decoder for statistical machine translation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1724–1734.

Minjin Choi, Sunkyung Lee, Eunseong Choi, Heesoo Park, Junhyuk Lee, Dongwon Lee, and Jongwuk Lee. 2021. MelBERT: Metaphor detection via contextualized late interaction using metaphorical identification theories. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 1763–1773.

Marta Coll-Florit, Salvador Climent, Marco Sanfilippo, and Eulàlia Hernández-Encuentra. 2021. Metaphors of depression. Studying first person accounts of life with depression published in blogs. *Metaphor and Symbol*, 36(1):1–19.

Yaping Deng, Lu Wang, Hao Jia, Xiangqian Tong, and Feng Li. 2019. A sequence-to-sequence deep learning architecture based on bidirectional GRU for type recognition and time location of combined power quality disturbance. *IEEE Transactions on Industrial Informatics*, 15(8):4481–4493.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of NAACL-HLT*, pages 4171–4186, Minneapolis, Minnesota. Association for Computational Linguistics.

Christiane Fellbaum. 1998. *WordNet: An Electronic Lexical Database*. Bradford Books.

Jie Gao, Sooji Han, Xingyi Song, and Fabio Ciravegna. 2020. RP-DNN: A tweet level propagation context based deep neural networks for early rumor detection in social media. In *Proceedings of The 12th Language Resources and Evaluation Conference*, pages 6094–6105, Marseille, France. European Language Resources Association.

Mengshi Ge, Rui Mao, and Erik Cambria. 2022. Explainable metaphor identification inspired by conceptual metaphor theory. In *Proceedings of AAAI*, pages 10681–10689.

Tao Gui, Liang Zhu, Qi Zhang, Minlong Peng, Xu Zhou, Keyu Ding, and Zhigang Chen. 2019. Cooperative multimodal approach to depression detection in twitter. In *Proceedings of the AAAI conference on Artificial Intelligence*, volume 33, pages 110–117.

Sharath Chandra Guntuku, David B Yaden, Margaret L Kern, Lyle H Ungar, and Johannes C Eichstaedt. 2017. Detecting depression and mental illness on social media: An integrative review. *Current Opinion in Behavioral Sciences*, 18:43–49.

Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long short-term memory. *Neural computation*, 9(8):1735–1780.

Shaoxiong Ji, Tianlin Zhang, Luna Ansari, Jie Fu, Prayag Tiwari, and Erik Cambria. 2022. MentalBERT: Publicly available pretrained language models for mental healthcare. In *Proceedings of The 13th Language Resources and Evaluation Conference*, pages 7184–7190.

Diederik P Kingma and Jimmy Ba. 2015. Adam: A method for stochastic optimization. In *Proceedings of the International Conference on Learning Representations*.

Richard R Kopp. 2013. *Metaphor therapy: Using client generated metaphors in psychotherapy*. Routledge.

George Lakoff, Jane Espenson, and Alan Schwartz. 1991. *Master Metaphor List*, 2nd edition. University of California at Berkeley.

George Lakoff and Mark Johnson. 1980. *Metaphors We Live by*. University of Chicago press.

Yann LeCun, Bernhard Boser, John S Denker, Donnie Henderson, Richard E Howard, Wayne Hubbard, and Lawrence D Jackel. 1989. Backpropagation applied to handwritten zip code recognition. *Neural computation*, 1(4):541–551.

Chenhao Lin, Pengwei Hu, Hui Su, Shaochun Li, Jing Mei, Jie Zhou, and Henry Leung. 2020. Sensemood: Depression detection on social media. In *Proceedings of the 2020 International Conference on Multimedia Retrieval*, pages 407–411.

Qian Liu, Jie Lu, Guangquan Zhang, Tao Shen, Zhihan Zhang, and Heyan Huang. 2021. Domain-specific meta-embedding with latent semantic structures. *Information Sciences*, 555:410–423.

Rui Mao and Xiao Li. 2021. Bridging towers of multitask learning with a gating mechanism for aspect-based sentiment analysis and sequential metaphor identification. In *Proceedings of the 35th AAAI Conference on Artificial Intelligence*, pages 13534–13542.

Rui Mao, Xiao Li, Mengshi Ge, and Erik Cambria. 2022a. MetaPro: A computational metaphor processing model for text pre-processing. *Information Fusion*, 86-87:30–43.

Rui Mao, Chenghua Lin, and Frank Guerin. 2018. Word embedding and WordNet based metaphor identification and interpretation. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics*, volume 1, pages 1222–1231.

Rui Mao, Chenghua Lin, and Frank Guerin. 2019. End-to-end sequential metaphor identification inspired by linguistic theories. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 3888–3898.

Rui Mao, Chenghua Lin, and Frank Guerin. 2021. Interpreting verbal metaphors by paraphrasing. *arXiv preprint arXiv:2104.03391*.

Rui Mao, Qian Liu, Kai He, Wei Li, and Erik Cambria. 2022b. The biases of pre-trained language models: An empirical study on prompt-based sentiment analysis and emotion detection. *IEEE Transactions on Affective Computing*, pages 1–11.

Zachary J Mason. 2004. CorMet: A computational, corpus-based conventional metaphor extraction system. *Computational linguistics*, 30(1):23–44.

NVIDIA, Péter Vingelmann, and Frank H.P. Fitzek. 2020. CUDA, release: 9.2.148.

Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Kopf, Edward Yang, Zachary DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. 2019. PyTorch: An imperative style, high-performance deep learning library. In *Advances in Neural Information Processing Systems 32*, pages 8024–8035. Curran Associates, Inc.

Guozheng Rao, Yue Zhang, Li Zhang, Qing Cong, and Zhiyong Feng. 2020. MGL-CNN: A hierarchical posts representations model for identifying depressed individuals in online forums. *IEEE Access*, 8:32395–32403.

Maryem Rhanoui, Mounia Mikram, Siham Yousfi, and Soukaina Barzali. 2019. A CNN-BiLSTM model for document-level sentiment analysis. *Machine Learning and Knowledge Extraction*, 1(3):832–847.

Kumarasan Roystonn, Wen Lin Teh, Ellaisha Samari, Laxman Cetty, Fiona Devi, Shazana Shahwan, Nisha Chandwani, and Mythily Subramaniam. 2021. Analysis and interpretation of metaphors: Exploring young adults' subjective experiences with depression. *Qualitative Health Research*, 31(8):1437–1447.

Ville Satopaa, Jeannie Albrecht, David Irwin, and Barath Raghavan. 2011. Finding a "kneedle" in a haystack: Detecting knee points in system behavior. In *International conference on distributed computing systems workshops*, pages 166–171. IEEE.

Mike Schuster and Kuldip K Paliwal. 1997. Bidirectional recurrent neural networks. *IEEE Transactions on Signal Processing*, 45(11):2673–2681.

Guangyao Shen, Jia Jia, Liqiang Nie, Fuli Feng, Cunjun Zhang, Tianrui Hu, Tat-Seng Chua, and Wenwu Zhu. 2017. Depression detection via harvesting social media: A multimodal dictionary learning solution. In *IJCAI*, pages 3838–3844.

Ekaterina Shutova, Lin Sun, Elkin Darío Gutiérrez, Patricia Lichtenstein, and Srini Narayanan. 2017. Multilingual metaphor processing: Experiments with semi-supervised and unsupervised learning. *Computational Linguistics*, 43(1):71–123.

Ellen Y Siegelman. 1993. *Metaphor and Meaning in Psychotherapy*. Guilford Press.

Karen Simonyan and Andrew Zisserman. 2014. Very deep convolutional networks for large-scale image recognition. In *Proceedings of the International Conference on Learning Representations*.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. *Advances in neural information processing systems*, 30.

Kav Vedhara, Jeremy Miles, Paul Bennett, Sue Plummer, Deborah Tallon, Emily Brooks, Lone Gale, Katherine Munnoch, Christa Schreiber-Kounine, Clare Fowler, et al. 2003. An investigation into the relationship between salivary cortisol, stress, anxiety and depression. *Biological psychology*, 62(2):89–96.

Jin Wang, Liang-Chih Yu, K Robert Lai, and Xuejie Zhang. 2016. Dimensional sentiment analysis using a regional CNN-LSTM model. In *Proceedings of the 54th annual meeting of the association for computational linguistics*, pages 225–230.

Le Yang, Dongmei Jiang, and Hichem Sahli. 2018. Integrating deep and shallow models for multi-modal depression analysis—hybrid architectures. *IEEE Transactions on Affective Computing*, 12(1):239–253.

Dongyu Zhang, Nan Shi, Ciyuan Peng, Abdul Aziz, Wenhong Zhao, and Feng Xia. 2021. MAM: A metaphor-based approach for mental illness detection. In *International Conference on Computational Science*, pages 570–583. Springer.

Hamad Zogan, Imran Razzak, Shoaib Jameel, and Guandong Xu. 2021. DepressionNet: A novel summarization boosted deep framework for depression detection on social media. *arXiv preprint arXiv:2105.10878*.

Multi-view and Cross-view Brain Decoding

Subba Reddy Oota[1,3]*, **Jashn Arora**[1*] **Manish Gupta**[1,2] **and Bapi Raju Surampudi**[1]
[1]IIIT Hyderabad, India; [2]Microsoft, India; [3]INRIA, Bordeaux, France
subba-reddy.oota@inria.fr, jashn.arora@research.iiit.ac.in
gmanish@microsoft.com, raju.bapi@iiit.ac.in

Abstract

Can we build multi-view decoders that can decode concepts from brain recordings corresponding to any view (picture, sentence, word cloud) of stimuli? Can we build a system that can use brain recordings to automatically describe what a subject is watching using keywords or sentences? How about a system that can automatically extract important keywords from sentences that a subject is reading?

Previous brain decoding efforts have focused only on single view analysis and hence cannot help us build such systems. As a first step toward building such systems, inspired by Natural Language Processing literature on multilingual and cross-lingual modeling, we propose two novel brain decoding setups: (1) multi-view decoding (MVD) and (2) cross-view decoding (CVD). In MVD, the goal is to build an MV decoder that can take brain recordings for any view as input and predict the concept. In CVD, the goal is to train a model which takes brain recordings for one view as input and decodes a semantic vector representation of another view. Specifically, we study practically useful CVD tasks like image captioning, image tagging, keyword extraction, and sentence formation.

Our extensive experiments lead to MVD models with $\sim$0.68 average pairwise accuracy across view pairs and CVD models with $\sim$0.8 average pairwise accuracy across tasks. Analysis of the contribution of different brain networks reveals exciting cognitive insights: (1) Models trained on picture or sentence view of stimuli are better MV decoders than a model trained on word cloud view. (2) Our extensive analysis across 9 broad brain regions, 11 language sub-regions, and 16 visual sub-regions of the brain help us localize, for the first time, the parts of the brain involved in cross-view tasks like image captioning, image tagging, sentence formation, and keyword extraction. We make the code publicly available[1].

1 Introduction

Brain decoding models aim to understand what a subject is thinking, seeing, and perceiving by analyzing neural recordings. Thus, in the context of language, it may be beneficial to learn mappings between linguistic representation and the associated brain activation, and how we compose the linguistic meaning from different stimuli such as text (Pereira et al., 2018; Wehbe et al., 2014a), images (Eickenberg et al., 2017; Beliy et al., 2019), videos (Huth et al., 2016; Nishimoto et al., 2011), or speech (Zhao et al., 2014) by analyzing the evoked brain activity. Also, decoding the functional activity of the brain has numerous applications in education and healthcare.

Brain recordings can be obtained by providing stimuli to a subject in various forms. For example, a concept (like *apartment*) can be presented using: (1) Word Picture (WP) view: picture along with the concept word, (2) Sentence (S) view: sentence containing the word, or (3) Word cloud (WC) view: word cloud containing the word along with other semantically related words. Recent studies have made much progress using functional magnetic resonance imaging (fMRI) brain activity to reconstruct semantic vectors corresponding to linguistic items, including words (Mitchell et al., 2008; Pereira et al., 2018), phrases, sentences, and paragraphs (Wehbe et al., 2014a). However, all such studies have been limited to single-view analysis. Separate models are trained to process different views. Also, the decoding target is typically a semantic vector of the concept word.

In the Natural Language Processing (NLP) community, researchers have recently started focusing on building multi-lingual and cross-lingual systems (Conneau et al., 2018; Conneau and Lample, 2019; Xue et al., 2021). Multi-lingual systems

*The first two authors made equal contribution.

[1]https://tinyurl.com/MVCVBD

Proceedings of the 29th International Conference on Computational Linguistics, pages 105–115
October 12–17, 2022.

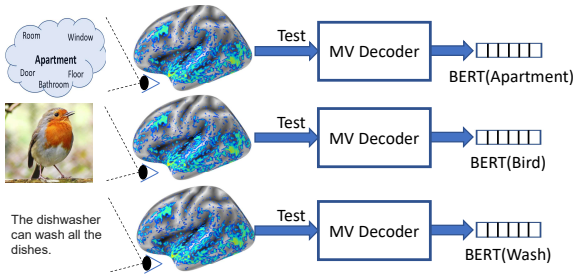

Figure 1: A multi-view decoder can be used to decode concepts using brain recordings for any view. Target is BERT representation of the concept word.

improve accuracy for low-resource languages and enable applications even in the absence of training data for low-resource languages. Cross-lingual systems take input in one language and produce output (e.g., summary) in another language. Inspired by this multi-lingual/cross-lingual shift in NLP, we propose two novel brain decoding setups: multi-view decoding (MVD) and cross-view decoding (CVD). Such setups are critical to build MV decoders which can decode concepts from brain recordings corresponding to any view (picture, sentence, word cloud) of stimuli or systems that can automatically describe using sentences or keywords what a subject is watching or automatically extract important keywords from sentences that a subject is reading.

In MVD, the goal is to build an MV decoder that can take brain recordings for any view as input and predict the concept. Fig. 1 shows examples of using an MV decoder. Such an MV decoder can be trained on data for any specific view. Multi-lingual models have shown huge zero-shot accuracy improvements for inference on low-resource language inputs across many NLP tasks (Conneau and Lample, 2019). Similarly, can we improve decoding accuracy using an MV decoder model for some views?

In CVD, the goal is to train a model which takes brain recordings for one view as input and decodes a semantic vector representation of another view. Fig. 2 shows examples of four such CVD tasks. Given an fMRI activation corresponding to a picture view of the stimuli, how accurately can we decode a sentence representing the picture? Which parts of the brain are involved in CVD tasks like image captioning, image tagging, keyword extraction, and sentence formation?

Historically, the fMRI brain activity has been decoded to a semantic vector representation of a view (word picture, sentence, word cloud) using a ridge-regression decoder (Pereira et al., 2018; Sun et al., 2019). In particular, earlier brain decoding works focused on hand-crafted features to train such decoder models (Mitchell et al., 2008; Wehbe et al., 2014a). Recently, many studies have shown accurate results in mapping the brain activity using neural distributed word embeddings for linguistic stimuli (Anderson et al., 2017; Pereira et al., 2018; Oota et al., 2018; Nishida and Nishimoto, 2018; Sun et al., 2019). To represent meaning, these studies use either word or sentence level embeddings extracted from the models trained on large corpora. Unfortunately, none of these addresses the open questions around multi-view decoding and cross-view decoding. Recently, Transformer-based models have been explored for brain encoding (Hollenstein et al., 2019), which inspires us to harness Transformer-based models like BERT (Devlin et al., 2019) for our brain decoding tasks.

Our main contributions are as follows. (1) We propose two novel brain decoding settings: multi-view decoding and cross-view decoding. (2) We build decoder models using Transformer-based methods and analyze brain network contributions across multi-view and cross-view tasks. (3) We augment the popular Pereira et al. (2018)'s dataset with pairwise-view relationships and use it to demonstrate the efficacy of our proposed methods. We make the code publicly available[2].

Our experiments lead us to the following insights: (1) Models trained on picture or sentence view are better MV decoders than models trained on word cloud view. Surprisingly, the MV decoder trained on sentence view leads to a zero-shot accuracy for word cloud stimuli, which is better than that obtained using the same-view word cloud model. (2) For the first time, we show language and visual sub-regions involved in four cross-view tasks. (3) High pairwise accuracies of 0.78, 0.83, 0.84, and 0.75 for image captioning, image tagging, keyword extraction, and sentence formation resp., help us conclude that cross-view decoding tasks using fMRI data are practically feasible.

2 Related Work

Advances in functional neuroimaging tools such as fMRI have made it easier to study the relationship between language/visual stimuli and functions of brain networks (Constable et al., 2004; Thirion et al., 2006; Fedorenko et al., 2010).

[2] https://tinyurl.com/MVCVBD

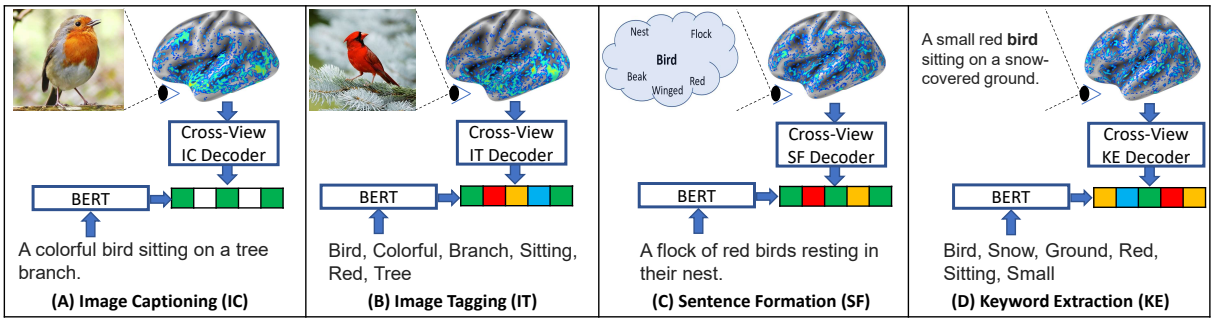

Figure 2: Cross-View Decoding Task (Input, output) Examples.

Initial brain decoding experiments studied the recovery of simple concrete nouns and verbs from fMRI brain activity (Mitchell et al., 2008; Palatucci et al., 2009; Nishimoto et al., 2011; Pereira et al., 2011) where the subject watches either a picture or a word. Unlike the earlier work, Wehbe et al. (2014a); Huth et al. (2016) built a model to decode the text passages instead of individual words. However, these studies used either simple or constrained sets of stimuli, which poses a question of generalization of these models. Recently, Pereira et al. (2018) explicitly decoded both words and sentences when subjects were shown both concrete and abstract stimuli. Affolter et al. (2020) reconstructed the sentences along with categorizing words or predicting the semantic vector representation from fMRI brain activity. Schwartz et al. (2019); Wang et al. (2020a) focused on understanding how multiple tasks activate associated regions in the brain.

To train ridge regression decoder models, earlier works focused on hand-crafted features (Mitchell et al., 2008; Wehbe et al., 2014a), which suffer from various drawbacks like inability to capture the context and sequential aspects of a sentence, inability to extract signals for abstract stimuli, etc. With the success of deep learning based word representations, multiple researchers have used distributed word embeddings for brain decoding models in place of carefully hand-crafted feature vectors (Huth et al., 2016; Anderson et al., 2017; Pereira et al., 2018; Oota et al., 2018; Nishida and Nishimoto, 2018; Sun et al., 2019; Wang et al., 2020b). Using the distributed sentence representations, Wehbe et al. (2014b); Jain and Huth (2018); Abnar et al. (2019); Sun et al. (2019) demonstrated that neural sentence representations are better for decoding whole sentences from brain activity patterns. Recently, Transformer models like BERT (Devlin et al., 2019) and GPT2 (Radford et al., 2019) have been found to be very effective for decoding (Gauthier and Levy, 2019; Toneva and Wehbe, 2019; Affolter et al., 2020). Inspired by such studies, we leverage BERT representations. Inspired by such studies, we leverage BERT representations. Unlike single-view analysis done in previous studies, multi-view and cross-view setups are the main focus of our work.

3 Methodology

3.1 Brain Imaging Dataset

We experiment with the popular dataset from (Pereira et al., 2018). It is obtained from 11 subjects (P01, M01, M02, M04, M07, M09, M10, M13, M15, M16, M17) where each subject read 180 concept words (abstract + concrete) in three different paradigms or views while functional magnetic resonance images (fMRI) were acquired. These contain 128 nouns, 22 verbs, 29 adjectives and adverbs, and 1 function word. In paradigm-1 (WP), participants were shown concept word along with picture with an aim to observing brain activation when participants retrieved relevant meaning using visual information. In paradigm-2 (S), the concept word presented in a sentence allows us to probe activity in the language areas associated with contextual information and meaning of a sentence. In paradigm-3 (WC), the concept word was presented in a word cloud format, surrounded by five semantically similar words. These paradigms provide brain representation of 180 concepts in three different views.

For each of the 180 concepts, the dataset contains five pictures, six sentences each containing the concept word, and a word cloud. For example, for a concept 'bird', dataset has (1) a picture p showing a red bird sitting on a tree branch, (2) sentence s like "A green bird flying in the sky", and (3) word cloud c with words "bird, purple, flock, winged, nest, beak". The dataset also has fMRIs for each of these three views. This dataset was

Task	Input	Output (View type)
Image captioning	Word+Picture fMRI	Caption (Sentence)
Image tagging	Word+Picture fMRI	Image tags (Word Cloud)
Keyword extraction	Sentence fMRI	Keywords (Word Cloud)
Sentence formation	Word-cloud fMRI	Sentence

Table 1: Cross-View Decoding Task Definitions

meant for single-view decoding and hence follows a star schema (concept at the center and specific views like word+picture, sentence, and word cloud around it). Clearly, we cannot use this dataset as is for cross-view decoding (CVD). For example, for the image captioning CVD task, it is wrong to take an fMRI with the stimuli being a picture showing a red bird sitting on a tree branch, and use it to decode a sentence "A green bird flying in the sky".

To enable cross-view decoding tasks, it was critical to build direct pairwise-view relationships (picture-sentence, picture-word cloud, sentence-word cloud, and word cloud-sentence). In other words, it was necessary to have captions and tags for image-view, keywords for sentence-view, and 3-4 sentences corresponding to wordcloud-view. Hence, we augment the dataset in Pereira et al. (2018) by obtaining target annotations manually. For example, for the fMRI associated with picture p, we manually annotated it with target sentence s'="A red bird sitting on a tree branch". Pairs like (p, s') are then used to train model for image captioning. Note that these manual annotations do not involve obtaining more fMRIs.

Fig. 2 shows the input and output examples for the four cross-view decoding tasks. We make the augmented dataset publicly available[2]. Note that we do not experiment with CVD tasks like image generation from sentences or word clouds since obtaining target annotations would mean that we need to draw images to augment the dataset. We leave it as part of future work.

3.2 Task Descriptions

We train the decoder regression models on 5000 informative voxels selected from fMRI brain activations and evaluate all the models using pairwise accuracy and rank-based decoding. Details of the informative voxel selection, the regression model, and metrics are discussed in the subsequent sections. The main goal of each decoder model is to predict a semantic vector representation of the stimuli in each experiment. The input view (word+picture, sentence, or word-cloud) and output representation (word, sentence, or word-cloud) differ across experiments. We follow K-fold cross-validation, in which all the data samples from K-1 folds were used for training, and the model was tested on samples of the left-out fold. We use the BERT-pooled output for obtaining output semantic representations. We also experimented with RoBERTa, but the results were very similar to BERT, and hence we omit them for lack of space.

Multi-View Concept Decoding For each subject in the dataset, for each of the three input views, we trained K=18 models (one for each fold) where each model is trained on the brain activity of 170 concepts and tested on left-out 10 concepts to predict vector representation of the concept word. The 5000 informative voxels were selected for 170 concepts in each fold, and the same voxel locations were chosen for test datasets. At test time, the input to each model can belong to any of the three views. Thus, for each subject, for each fold, we perform (1) three same-view train-test experiments and (2) six multi-view zero-shot train-test experiments with different input views at train and test time. Target is always fixed as a vector representation of the concept word. We use pairwise accuracy to report results.

Cross-view Decoding Tasks For each subject in the dataset, we learn models for the four cross-view decoding tasks (IC, IT, KE, SF) using 18 fold cross-validation. The input and output for each of these tasks is shown in Table 1. Fig. 2 shows an example for each task. As before, we use 5000 informative voxels, computed separately for each of the 11 subjects and each of the four tasks. The regression target is semantic vector representation.

3.3 Informative Voxel Selection

Inspired by the voxel selection method in (Pereira et al., 2018), we chose the informative voxels for our linear regression models as follows. The regression models are trained on each voxel and its 26 neighboring voxels to predict the semantic vector representation. For each voxel in the training part, the mean correlation was calculated between "true" (text-derived) and predicted representations, and the voxels corresponding to the top 5000 mean cor-

relation values were selected as informative voxels. Target semantic representations are word embeddings for multi-view zero-shot concept decoding and 'word or sentence or word-cloud' embedding for cross-view decoding experiments. Voxel selection provides meaningful cognitive insights across brain networks.

3.4 Model Architecture

We trained a ridge regression based decoding model to predict the semantic vector representation associated with the fMRI informative voxels for a type (view) of each language stimulus. Each dimension is predicted using a separate ridge regression model. Formally, we are given the informative voxel matrix $X \in \mathbb{R}^{N \times V}$ and stimuli vector representation $Y \in \mathbb{R}^{N \times D}$, where N denotes the number of training examples, V denotes the number of informative voxels (we fix it to 5000), and D denotes the embedding dimension of language stimuli. For BERT, D=768. The ridge regression objective function is $f(X_i) = \min_{W_{io}} \|Y_o - X_i W_{io}\|_F^2 + \lambda \|W_{io}\|_F^2$ where, X_i denotes the input voxels for view i (out of {word+picture, sentence, wordcloud}), Y_o denotes the matrix with embeddings o (out of {word, sentence, word cloud}), W_{io} denotes the learned weight coefficients for each input view i and output embedding o, $\|.\|_F$ denotes the Frobenius norm, and $\lambda > 0$ is a tunable hyper-parameter representing the regularization weight. Besides ridge regression, of course, various other models could be used. However, the goal of this paper is to analyze novel decoding setups using the most popular decoding model in neuro-science literature, namely, ridge regression. We leave exploration of complex models as part of future work. **Hyper-parameter Settings**: We used sklearn's ridge regression with default parameters, 18-fold cross-validation, Stochastic-Average-Gradient Descent Optimizer, Huggingface for BERT, MSE loss function and L2-decay (λ):1.0.

3.5 Brain Networks Selection

Inspired by Pereira et al. (2018) and based on the resting-state functional networks, we focused on four brain networks: Default Mode Network (DMN) (linked to the functionality of semantic processing) (Buckner et al., 2008; Binder et al., 2009), Language Network (related to language processing, understanding, word meaning, and sentence comprehension) (Fedorenko et al., 2011), Task Positive Network (related to attention, salience information) (Binder et al., 2009; Duncan, 2010; Power et al., 2011), and Visual Network (related to the processing of visual objects, object recognition) (Buckner et al., 2008; Power et al., 2011). We report the distribution of 5000 informative voxels across the four brain networks across various experiments in Section 4. Across all participants, voxel distribution across networks is as follows: 4670 (Language), 6490 (DMN), 11630 (TP), and 8170 (Visual). Note that the reported distributions in Section 4 do not add up to 1 because the contribution of the remaining brain networks is not considered.

4 Results and Cognitive Insights

Since we are the first to propose multi-view and cross-view tasks, unfortunately, there are no baselines to compare with. For the sake of comparison, we design a "chance-level" BERT (Random) baseline where models are trained using BERT embeddings of randomly chosen words as a target rather than BERT embeddings of the actual target word. For same-view experiments, our results are in line with that reported in (Pereira et al., 2018).

5 Evaluation Metrics

We use the popular pairwise and rank accuracy metrics for evaluation. **Pairwise Accuracy** To measure the pairwise accuracy, the first step is to predict all the test stimulus vector representations using a trained decoder model. Let S = [S_0, S_1,$\cdots$,S_n], $\hat{S}$ = [$\hat{S}_0$, $\hat{S}_1$,$\cdots$,$\hat{S}_n$] denote the "true" (text-derived) and predicted stimulus representations for n test instances resp. Given a pair (i, j) such that $0 \leq i, j \leq n$, score is 1 if *corr*(S_i,$\hat{S}_i$) + *corr*(S_j,$\hat{S}_j$) > *corr*(S_i,$\hat{S}_j$) + *corr*(S_j,$\hat{S}_i$), else 0. Here, *corr* denotes the Pearson correlation. Final pairwise matching accuracy per participant is the average of scores across all pairs of test instances.

Rank Accuracy We compared each decoded vector to all the "true" text-derived semantic vectors and ranked them by their correlation. The classification performance reflects the rank r of the text-derived vector for the correct word: $1 - \frac{r-1}{\#instances-1}$. The final accuracy value for each participant is the average rank accuracy across all instances.

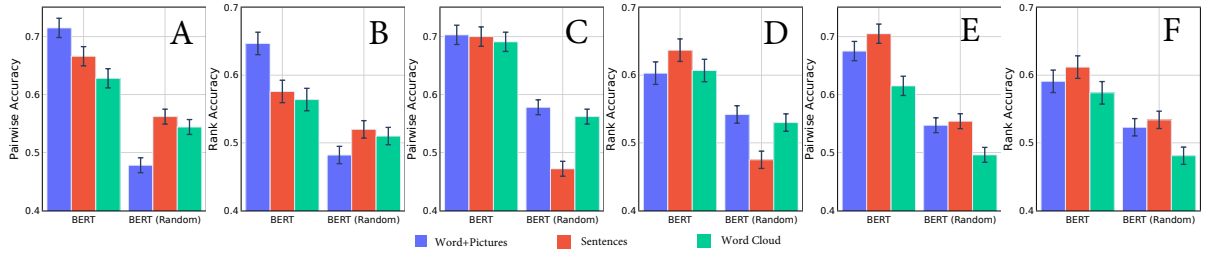

Figure 3: Model trained on *Word+Pictures* (A and B), *Sentences* (C and D), and *Word-Cloud* (E and F) view. MVD Pairwise and Rank accuracy when tested on Word+Picture/Sentence/Word-cloud views, averaged across all subjects.

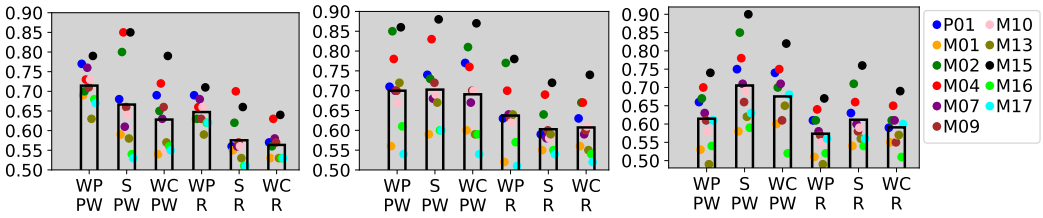

Figure 4: Model trained on *Word+Pictures* view (left), *Sentences* view (middle), *Word-Cloud* view (right). MVD Pairwise (PW) and Rank (R) accuracy when tested on Word+Picture (WP)/Sentence (S)/Word-cloud (WC) views. Each colored dot represents a subject. The bar plot shows averages.

5.1 Multi-View Concept Decoding

5.1.1 Pairwise and Rank Accuracy Results

Fig. 3 and Table 2 show detailed results for models trained on word+picture (WP), sentence (S), and word-cloud (WC) views and tested on each of the three views. Specifically, Fig. 3(A) shows pairwise accuracy results when we train using the WP view but infer using voxels corresponding to any of the three views. Ground-truth is the BERT embedding vector. In comparison to the "chance-level" BERT (Random) baseline with random target vectors, our proposed BERT embedding-based method is much better. Fig. 4 shows subject wise results.

Test↓/Train→	WP	S	WC
WP	0.72/0.65	0.70/0.60	0.68/0.59
S	0.67/0.58	0.70/0.64	0.71/0.61
WC	0.63/0.56	0.69/0.61	0.62/0.57

Table 2: Multi-View Zero-shot Concept Decoder Results (Pairwise/Rank Accuracy)

Same view versus MV zero-shot: In most cases, same-view results are better than multi-view zero-shot results. However, this does not hold for the WC view, where a model trained on sentence view performs better (Left green bars in Fig. 3 (C and D) vs. Fig. 3 (E and F)).

Can we train MV decoders that can decode concepts from brain recordings for any view? We experimented with three different MV decoders, each trained on one of the three views. Fig. 3 and the statistical significance test results in Table 3 show that either of the WP and sentence (S) views can be used to train MV decoders. This means that if we train a model with WP or S view fMRIs, and test it using any of the three views, the results are better or equivalent to any other model. This does not hold for the WC view. Thus, an MV decoder trained with a WC view is not very effective.

Setting 1	Setting 2	p-value
Train(WP)-Test(WP)	Train(S)-Test(WP)	0.098
Train(WP)-Test(WP)	Train(WC)-Test(WP)	0.026*
Train(S)-Test(WP)	Train(WC)-Test(WP)	0.474
Train(WP)-Test(S)	Train(S)-Test(S)	0.485
Train(WC)-Test(S)	Train(S)-Test(S)	0.469
Train(WP)-Test(S)	Train(WC)-Test(S)	0.420
Train(WP)-Test(WC)	Train(WC)-Test(WC)	0.691
Train(WP)-Test(WC)	Train(S)-Test(WC)	0.134
Train(S)-Test(WC)	Train(WC)-Test(WC)	0.045*

Table 3: p-values for measuring if setting 1 is stat significantly better than setting 2. Only rows with * mark denote statistically significant improvements.

5.1.2 Cognitive Insights based on Distribution of Informative Voxels

Table 4 and Fig. 5 show the distribution of informative voxels among four brain networks for various MV models. In this figure, (WP, D) means input view=WP (Word+picture), brain network=DMN (D). The figure clearly shows that a lot of informative voxels belong to the visual brain region for the WP view. Also, for sentence view, a large percentage of informative voxels are from the language region.

Figs. 6 to 8 show more distribution details by zooming further into language and visual regions.

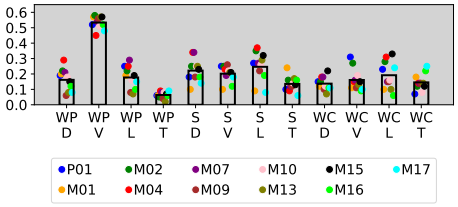

Figure 5: Distribution of informative voxels among four brain networks: DMN (D), Visual (V), Language (L), Task Positive (T). Models trained on Word+Picture (WP), Sentence (S) or Word-Cloud (WC) views.

When the model is trained on the WP view (unlike other views), Table 4 and Fig. 6 show that most informative voxels (about 53%) lie in the visual brain network, which is expected for the predominantly visual information-driven task.

	Word+Picture	Sentence	Word-Cloud
DMN	0.162	0.222	0.137
Visual	0.534	0.202	0.161
Language	0.177	0.246	0.192
Task-Positive	0.064	0.135	0.145

Table 4: Distribution of informative voxels among four brain networks for various Multi-View models

We also observe that DMN and Language network voxels are higher in the sentence view than in the word cloud view. Compared to the model trained on WP view, the distribution of voxels among the four brain networks shows that the model trained on sentence view has a higher percentage of voxels among the Language, DMN, and Task-positive networks and lower in the visual network. This is in line with our understanding that linguistic and attention skills are essential for understanding sentence stimuli. As for the model trained on the WC view compared to other views, we see that the informative voxels are spread equally among all the networks. From Fig. 6, we observe that in all the views, the region corresponding to language processing in the left hemisphere (Language_LH) has higher informative voxels than that of the right hemisphere (Language_RH). This is in line with the left hemisphere dominance for language processing (Binder et al., 2009). When the visual network dominates as in the case of WP view, the majority of these are located in the object processing area, followed by face and body processing areas. In the following, we investigate these two regions in detail.

In the language network, the distribution of informative voxels in the sub regions (LPTG, LMTG, LATG, LFus, LPar, LAngG, LIFGorb,

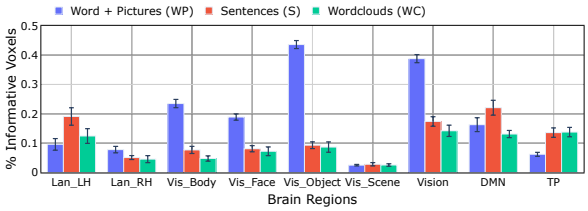

Figure 6: Distribution of informative voxels among nine brain regions for Multi-view Decoding

LIFG, LaMFG, LpMFG, and LmMFG) are shown in Fig. 7. We find that regions in the posterior (LPTG), middle (LMTG), and anterior (LATG) temporal gyrus share a higher percentage of informative voxels than other regions in the language network, such as those in the middle and inferior frontal areas. This indicates that the language functions sub-served by the temporal cortex, such as comprehension and semantic processing, are critical for processing sentences as well as multi-modal integration and thus are important for decoding across multiple views. Further, brain regions in the angular gyrus (LAngG) and parietal (LPar) each have >5% of informative voxels. These areas may be involved in attention, self-processing, and visio-linguistic integration.

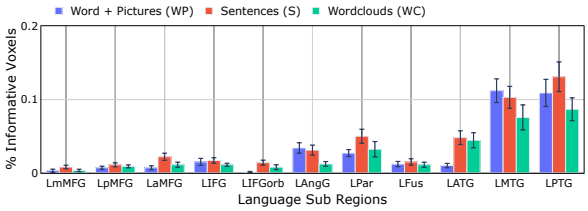

Figure 7: Distribution of informative voxels among eleven sub regions of Language network for MVD

Similarly, we explored the distribution of informative voxels across sub regions of the visual network, as shown in Fig. 8. In the visual sub regions, voxels in the bilateral occipital cortex (LLOC and RLOC) have more informative voxels than in other sub regions. In particular, the scene regions in the parahippocampal place area (such as RSC and PPA) display very few informative voxels, while the bilateral body area (REBA and LEBA) captures more

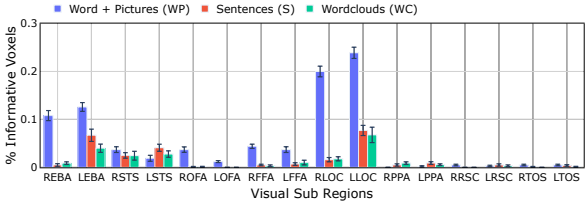

Figure 8: Distribution of informative voxels among sixteen sub regions of Visual network for MVD

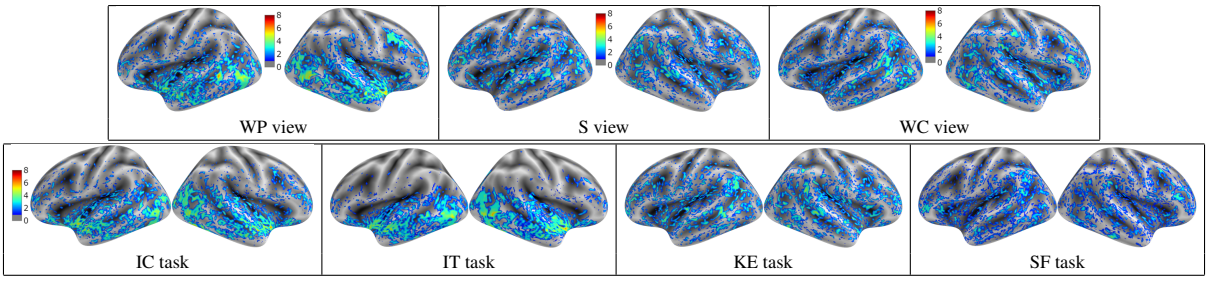

Figure 9: Brain Maps for Multi-View and Cross-View Decoding Tasks (plotted using nilearn Python library).

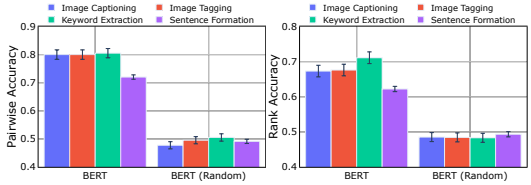

Figure 10: Cross-View Decoding Pairwise and Rank accuracy for Image Captioning (IC), Image Tagging (IT), Keyword Extraction (KE), and Sentence Formation (SF) averaged across all the subjects.

voxels in the WP view. Interestingly, activation in the superior temporal sulcus (RSTS and LSTS) in all views point out its role in visio-linguistic integration. Lastly, Fig. 9 shows the spatial distribution of informative voxels (plotted using nilearn Python library) across models trained on different forms of stimuli (WP, S, and WC). The value of each voxel is the fraction of 11 participants for whom that voxel was among the 5000 most informative.

5.1.3 Informative Voxel Overlap across Views

Given the distribution of informative voxels across four brain networks, we further examine how these voxels from one view overlap with those from another view. Table 5 shows that (1) the language network has a very high overlap compared to other brain networks in the WC-S pair. (2) 29% (and 25%) of visual voxels for the S (and WC) view are shared with visual voxels of the WP view. This makes sense since a large percentage of informative voxels for WP view are from the visual network.

	DMN	Visual	Language	Task Positive
WP-S	0.24/0.17	0.11/0.29	0.25/0.17	0.09/0.05
WC-S	0.25/0.16	0.25/0.20	0.30/0.22	0.07/0.07
WP-WC	0.14/0.16	0.08/0.25	0.15/0.15	0.06/0.03

Table 5: For each pair of views and each brain network, we show coverage ratios (second task on first/first task on second) of the voxels.

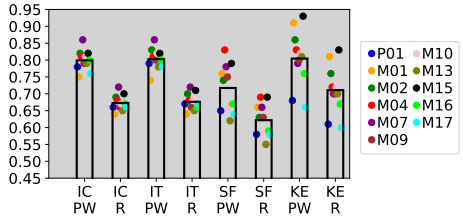

Figure 11: CVD Pairwise (PW) and Rank (R) accuracy for IC, IT, KE and SF tasks. Each colored dot represents a subject. The bar plot shows averages.

	IC	IT	SF	KE
DMN	0.114	0.067	0.152	0.214
Visual	0.572	0.736	0.154	0.236
Language	0.116	0.081	0.182	0.275
Task Positive	0.045	0.007	0.141	0.118

Table 6: Distribution of informative voxels among four brain networks for all 4 CVD Tasks.

5.2 Cross-View Decoding

5.2.1 Pairwise and Rank Accuracy Results

Fig. 10 illustrates pairwise and rank accuracy for Image Captioning (IC), Image Tagging (IT), Sentence Formation (SF), and Keyword Extraction (KE). Subject wise results are reported in Fig. 11. We observe that (1) our proposed BERT embedding-based method is much better compared to the "chance-level" baseline with random target vectors. (2) For all the four tasks, pairwise accuracy is ∼80%, and rank-based accuracy is ∼70% (except for SF), which shows that CVD is possible with good accuracy.

5.2.2 Cognitive Insights based on Distribution of Informative Voxels

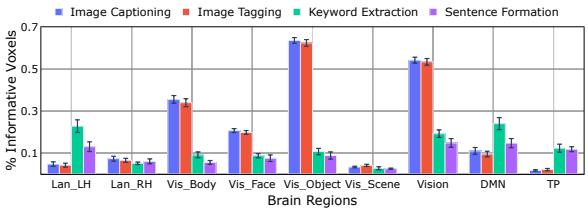

Figure 12: Distribution of informative voxels among nine brain regions for CVD tasks.

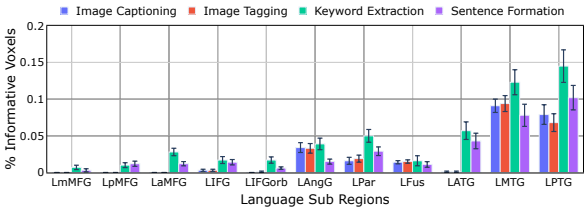

Figure 13: Distribution of informative voxels among 11 sub regions of Language network for CVD tasks.

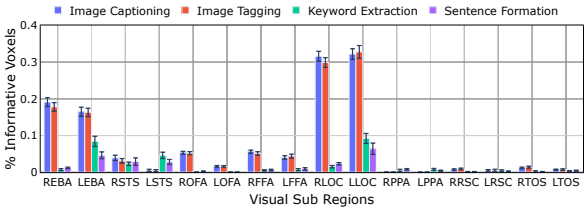

Figure 14: Distribution of informative voxels among 16 sub regions of Visual network for CVD tasks.

Fig. 12 shows the distribution of informative voxels among nine brain regions across all four tasks. As expected, a high percentage of visual voxels are involved in IC and IT tasks, and a high percentage of language voxels are involved in the SF and KE tasks, especially in the left hemisphere. Further, from Table 6, we observe that IC involves relatively higher language voxels compared to IT. This could be because generating a caption involves a higher level of language (sequence) skills than generating a set of keywords.

To further investigate the informative voxel distribution across Language and Visual networks, we display the sub region voxels distribution for the Language network in Fig. 13, and for the Visual network in Fig. 14. In all the tasks, the left hemisphere language network activation is dominated by activity in the temporal gyrus (middle: LMTG and posterior: LPTG) but more in the KE task. This clearly demonstrates the importance of language comprehension and semantic process common across the cross-view tasks. Further, the common activation in the angular gyrus (LAG) in all tasks points out the role of visio-linguistic integration critical for all the tasks. The activation profile of the vision network, in contrast, shows distinct activation differences across the tasks (IC & IT vs. KE & SF). IC and IT tasks are related to a higher proportion of informative voxels in the primary visual regions in the lateral occipital areas (LLOC, RLOC) and bilateral extrastriate body-related areas (REBA and LEBA). Domination of activation in the vision network in captioning and tagging tasks (IC and IT) as compared to predominantly sentence processing based tasks (KE and SF) is along expected lines.

	DMN	Visual	Language	Task Positive
IC-IT	0.27/0.44	0.70/0.54	0.32/0.45	0.07/0.32
IC-KE	0.31/0.17	0.11/0.27	0.28/0.12	0.12/0.05
IC-SF	0.16/0.12	0.07/0.25	0.14/0.09	0.08/0.03
IT-KE	0.27/0.08	0.08/0.25	0.22/0.07	0.05/0.01
IT-SF	0.13/0.05	0.06/0.27	0.10/0.05	0.04/0.00
KE-SF	0.19/0.26	0.20/0.29	0.22/0.32	0.09/0.08

Table 7: For each pair of CVD tasks and each brain network, we show coverage ratios (second task on first/first task on second) of the voxels.

The brain maps (see Fig. 9) corresponding to the IC and IT tasks clearly activate the visual cortex and the temporal cortex, the areas known for visual processing and object identification. On the other hand, the brain maps of KE and SF exhibit diffuse activation that includes the temporal and frontal regions known to be related to the sentence semantics. None of the maps show a left-hemisphere bias, which is often found in such semantic-related maps. Lack of frontal-lobe activation and the concentration of informative voxels in the sensory cortex suggest that the cross-view embedding may rely on some non-abstract domain-specific encoding rather than higher-level semantic concept encoding.

5.2.3 Informative Voxel Overlap across Tasks

Given the distribution of informative voxels across four brain networks, we further examine how these voxels from one task overlap with those from another task. Table 7 shows that (1) Many voxels overlap across different brain networks for IC and IT tasks. This is expected since the two tasks are very related. Interestingly, 44% of DMN voxels needed for IT are shared with IC. Similarly, as high as 70% of visual voxels needed for IC are shared with IT. (2) Similarly, KE and SF share a very good overlap across different brain networks, which is expected given the textual nature of the two tasks.

6 Conclusion

We studied brain decoding in the context of zero-shot multi-view concept decoding and cross-view decoding tasks. We studied four cross-view decoding tasks: image captioning, image tagging, sentence formation, and keyword extraction. We show that cross-view decoding is feasible with good accuracy. Brain network distribution analysis reveals insights about the importance of various parts of the brain for each of these tasks.

7 Ethical Statement

We reused publicly available Pereira dataset, downloadable from https://osf.io/crwz7/, for this work. Please read their terms of use[3] for more details. We did not collect any new dataset. We do not foresee any harmful uses of this technology.

References

Samira Abnar, Lisa Beinborn, Rochelle Choenni, and Jelle Zuidema. 2019. Blackbox meets blackbox: Representational similarity and stability analysis of neural language models and brains. In *Proceedings of the ACL-Workshop on Analyzing and Interpreting Neural Networks for NLP*, pages 191–203.

Nicolas Affolter, Beni Egressy, Damian Pascual, and Roger Wattenhofer. 2020. Brain2word: Decoding brain activity for language generation. *arXiv preprint arXiv:2009.04765*.

Andrew J Anderson, Douwe Kiela, Stephen Clark, and Massimo Poesio. 2017. Visually grounded and textual semantic models differentially decode brain activity associated with concrete and abstract nouns. *Transactions of the Association for Computational Linguistics*, 5:17–30.

Roman Beliy, Guy Gaziv, Assaf Hoogi, Francesca Strappini, Tal Golan, and Michal Irani. 2019. From voxels to pixels and back: Self-supervision in natural-image reconstruction from fmri. *arXiv preprint arXiv:1907.02431*.

Jeffrey R Binder, Rutvik H Desai, William W Graves, and Lisa L Conant. 2009. Where is the semantic system? a critical review and meta-analysis of 120 functional neuroimaging studies. *Cerebral cortex*, 19(12):2767–2796.

RL Buckner, JR Andrews-Hanna, and DL Schacter. 2008. The brain's default network: anatomy, function, and relevance to disease. *Annals of the New York Academy of Sciences*, 1124:1–38.

Alexis Conneau and Guillaume Lample. 2019. Cross-lingual language model pretraining. *Advances in neural information processing systems*, 32.

Alexis Conneau, Ruty Rinott, Guillaume Lample, Adina Williams, Samuel Bowman, Holger Schwenk, and Veselin Stoyanov. 2018. Xnli: Evaluating cross-lingual sentence representations. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 2475–2485.

R Todd Constable, Kenneth R Pugh, Ella Berroya, W Einar Mencl, Michael Westerveld, Weijia Ni, and Donald Shankweiler. 2004. Sentence complexity and input modality effects in sentence comprehension: an fmri study. *NeuroImage*, 22(1):11–21.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. Bert: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186.

John Duncan. 2010. The multiple-demand (md) system of the primate brain: mental programs for intelligent behaviour. *Trends in cognitive sciences*, 14(4):172–179.

Michael Eickenberg, Alexandre Gramfort, Gaël Varoquaux, and Bertrand Thirion. 2017. Seeing it all: Convolutional network layers map the function of the human visual system. *NeuroImage*, 152:184–194.

Evelina Fedorenko, Michael K Behr, and Nancy Kanwisher. 2011. Functional specificity for high-level linguistic processing in the human brain. *Proceedings of the National Academy of Sciences*, 108(39):16428–16433.

Evelina Fedorenko, Po-Jang Hsieh, Alfonso Nieto-Castañón, Susan Whitfield-Gabrieli, and Nancy Kanwisher. 2010. New method for fmri investigations of language: defining rois functionally in individual subjects. *Journal of neurophysiology*, 104(2):1177–1194.

Jon Gauthier and Roger Levy. 2019. Linking artificial and human neural representations of language. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 529–539.

Nora Hollenstein, A de la Torre, Nicolas Langer, and Ce Zhang. 2019. Cognival: A framework for cognitive word embedding evaluation. In *Proceedings of The SIGNLL Conference on Computational Natural Language Learning 2019*.

Alexander G Huth, Tyler Lee, Shinji Nishimoto, Natalia Y Bilenko, An T Vu, and Jack L Gallant. 2016. Decoding the semantic content of natural movies from human brain activity. *Frontiers in systems neuroscience*, 10:81.

Shailee Jain and Alexander G Huth. 2018. Incorporating context into language encoding models for fmri. In *Proceedings of the 32nd International Conference on Neural Information Processing Systems*, pages 6629–6638.

Tom M Mitchell, Svetlana V Shinkareva, Andrew Carlson, Kai-Min Chang, Vicente L Malave, Robert A Mason, and Marcel Adam Just. 2008. Predicting human brain activity associated with the meanings of nouns. *science*, 320(5880):1191–1195.

[3] https://github.com/CenterForOpenScience/cos.io/blob/master/TERMS_OF_USE.md

Satoshi Nishida and Shinji Nishimoto. 2018. Decoding naturalistic experiences from human brain activity via distributed representations of words. *Neuroimage*, 180:232–242.

Shinji Nishimoto, An T Vu, Thomas Naselaris, Yuval Benjamini, Bin Yu, and Jack L Gallant. 2011. Reconstructing visual experiences from brain activity evoked by natural movies. *Current biology*, 21(19):1641–1646.

Subba Reddy Oota, Naresh Manwani, and Raju S Bapi. 2018. fMRI Semantic Category Decoding Using Linguistic Encoding of Word Embeddings. In *International Conference on Neural Information Processing*, pages 3–15. Springer.

Mark Palatucci, Dean Pomerleau, Geoffrey E Hinton, and Tom M Mitchell. 2009. Zero-shot learning with semantic output codes. In *NIPS*.

Francisco Pereira, Greg Detre, and Matthew Botvinick. 2011. Generating text from functional brain images. *Frontiers in human neuroscience*, 5:72.

Francisco Pereira, Bin Lou, Brianna Pritchett, Samuel Ritter, Samuel J Gershman, Nancy Kanwisher, Matthew Botvinick, and Evelina Fedorenko. 2018. Toward a universal decoder of linguistic meaning from brain activation. *Nature communications*, 9(1):1–13.

Jonathan D Power, Alexander L Cohen, Steven M Nelson, Gagan S Wig, Kelly Anne Barnes, Jessica A Church, Alecia C Vogel, Timothy O Laumann, Fran M Miezin, Bradley L Schlaggar, et al. 2011. Functional network organization of the human brain. *Neuron*, 72(4):665–678.

Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, Ilya Sutskever, et al. 2019. Language models are unsupervised multitask learners. *OpenAI blog*, 1(8):9.

Dan Schwartz, Mariya Toneva, and Leila Wehbe. 2019. Inducing brain-relevant bias in natural language processing models. *arXiv preprint arXiv:1911.03268*.

Jingyuan Sun, Shaonan Wang, Jiajun Zhang, and Chengqing Zong. 2019. Towards sentence-level brain decoding with distributed representations. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, pages 7047–7054.

Bertrand Thirion, Edouard Duchesnay, Edward Hubbard, Jessica Dubois, Jean-Baptiste Poline, Denis Lebihan, and Stanislas Dehaene. 2006. Inverse retinotopy: inferring the visual content of images from brain activation patterns. *Neuroimage*, 33(4):1104–1116.

Mariya Toneva and Leila Wehbe. 2019. Interpreting and improving natural-language processing (in machines) with natural language-processing (in the brain). *arXiv preprint arXiv:1905.11833*.

Shaonan Wang, Jiajun Zhang, Nan Lin, and Chengqing Zong. 2020a. Probing brain activation patterns by dissociating semantics and syntax in sentences. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pages 9201–9208.

Shaonan Wang, Jiajun Zhang, Haiyan Wang, Nan Lin, and Chengqing Zong. 2020b. Fine-grained neural decoding with distributed word representations. *Information Sciences*, 507:256–272.

Leila Wehbe, Brian Murphy, Partha Talukdar, Alona Fyshe, Aaditya Ramdas, and Tom Mitchell. 2014a. Simultaneously uncovering the patterns of brain regions involved in different story reading subprocesses. *PloS one*, 9(11):e112575.

Leila Wehbe, Ashish Vaswani, Kevin Knight, and Tom Mitchell. 2014b. Aligning context-based statistical models of language with brain activity during reading. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 233–243.

Linting Xue, Noah Constant, Adam Roberts, Mihir Kale, Rami Al-Rfou, Aditya Siddhant, Aditya Barua, and Colin Raffel. 2021. mt5: A massively multilingual pre-trained text-to-text transformer. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 483–498.

Shijie Zhao, Xi Jiang, Junwei Han, Xintao Hu, Dajiang Zhu, Jinglei Lv, Tuo Zhang, Lei Guo, and Tianming Liu. 2014. Decoding auditory saliency from fmri brain imaging. In *Proceedings of the 22nd ACM international conference on Multimedia*, pages 873–876.

Visio-Linguistic Brain Encoding

Subba Reddy Oota[1,2], **Jashn Arora**[2], **Vijay Rowtula**[2], **Manish Gupta**[2,3], **Raju S. Bapi**[2]
[1]Inria Bordeaux, France, [2]IIIT Hyderabad, India, [3]Microsoft, Hyderabad, India
subba-reddy.oota@inria.fr, {jashn.arora, vijay.rowtula}@research.iiit.ac.in,
gmanish@microsoft.com, raju.bapi@iiit.ac.in

Abstract

Brain encoding aims at reconstructing fMRI brain activity given a stimulus. Earlier neural encoding models focused on brain encoding for single-mode stimuli: visual (pretrained CNNs) or text (pretrained language models). Few recent papers have also obtained separate visual and text representation models and performed late-fusion using simple heuristics. However, the human brain perceives the environment using information from multiple modalities, and previous works have not explored the co-attentive multi-modal encoding for visual and text reasoning. This paper systematically explores image and multi-modal Transformers' efficacy for brain encoding. Extensive experiments on two popular datasets, BOLD5000 and Pereira, provide the following insights. (1) We find that VisualBERT, a multi-modal Transformer, significantly outperforms previously proposed single-mode CNNs, image Transformers, and other previously proposed multi-modal models, thereby establishing new state-of-the-art. (2) The regions such as LPTG, LMTG, LIFG, and STS, which have dual functionalities for language and vision, have a higher correlation with multi-modal models, which reinforces the fact that these models are good at mimicking the human brain behavior. (3) The supremacy of visio-linguistic models raises the question of whether the responses elicited in the visual regions are affected implicitly by linguistic processing even when passively viewing images. Future fMRI tasks can verify this computational insight in an appropriate experimental setting. We make our code publicly available[1].

1 Introduction

Brain encoding aims at constructing neural brain activity recordings given an input stimulus. The two most studied forms of stimuli include vision and language. Since discovering of the relationship between language/visual stimuli and functions of brain networks (Constable et al., 2004; Thirion et al., 2006), researchers have been interested in understanding how the neural encoding models predict the fMRI (functional magnetic resonance imaging) brain activity. Recently, several brain encoding models were developed to (i) understand the ventral stream in biological vision (Yamins et al., 2014; Kietzmann et al., 2019; Bao et al., 2020) and (ii) study higher-level cognition like language processing (Gauthier and Levy, 2019; Schrimpf et al., 2020a; Schwartz et al., 2019). Previous work has mainly focused on independently understanding vision and text stimuli. However, the biological systems perceive the world by simultaneously processing high-dimensional inputs from diverse modalities such as vision, auditory, touch, and proprioception (Jaegle et al., 2021). In particular, how the brain effectively processes and provides its visual understanding through natural language and vice versa is still an open question in neuroscience.

Earlier studies mainly were related to neural encoding models that predict brain activity using representations of single-mode stimuli: visual or text. Convolutional neural networks (CNNs) were known to encode semantics from visual stimuli effectively. Interestingly, intermediate layers in deep CNNs trained on the ImageNet (Deng et al., 2009) categorization task can partially account for how neurons in intermediate layers of the visual system respond to any given image (Yamins et al., 2013, 2014; Güçlü and van Gerven, 2015; Yamins and DiCarlo, 2016; Wang et al., 2019). However, more recent and deeper CNNs did not further improve on measures of brain-likeness, even though their ImageNet performance has vastly increased (Russakovsky et al., 2015). Recently, Kubilius et al. (2019) proposed a shallow recurrent anatomical network, CORnet, which provided state-of-the-art results on the Brain-score (Schrimpf et al., 2020b) benchmark. Similar to CNN based visual encoding models, various studies leveraged

[1]https://tinyurl.com/VLBEncoding

Proceedings of the 29th International Conference on Computational Linguistics, pages 116–133
October 12–17, 2022.

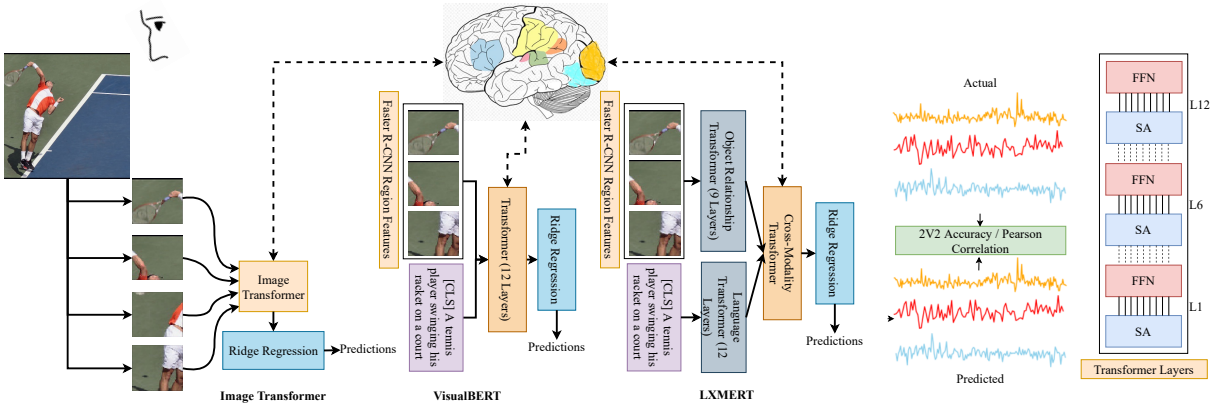

Fig. 1: Logical architecture of the proposed approach: We use features from image/multi-modal Transformers (like ViT, VisualBERT, and LXMERT) as input to the regression model to predict the fMRI activations for different brain regions. We evaluate the brain encoding results by computing 2V2 accuracy and Pearson correlation between actual and predicted activations. We also perform layer-wise correlation analysis between transformer layers and brain regions.

neural models like deep recurrent neural networks (RNNs), Transformer (Vaswani et al., 2017) based language models such as BERT (Devlin et al., 2019), RoBERTa (Liu et al., 2019), and GPT-2 (Radford et al., 2019) to predict the brain activity corresponding to semantic vectors of linguistic items, including words, phrases, sentences, and paragraphs (Gauthier and Levy, 2019; Schrimpf et al., 2020a).

Brain encoding for more brain regions is vital since input stimuli elicit diverse and distributed representations in the brain. These activation responses could be internally repurposed for several other brain tasks. Although previous neural encoding models have demonstrated promising results for processing one of the two brain regions (visual cortex V4 and prefrontal cortex IT), more efforts are needed to improve brain encoding for other parts of the brain. Further, previous works manually choose[2] particular CNN layers, whose activations were used for predicting brain activity specific to the datasets they work with (Kubilius et al., 2019). Applying such methods to other datasets will need dataset-specific, time-consuming manual identification of the best layer. We observe in our experiments that using *last layer* activations from VisualBERT leads to the best accuracy.

Unlike previous studies, which focus on single-modality (visual or language stimuli), some authors demonstrated that multi-modal models formed by combining text-based distributional information with visual representations provide a better proxy for human-like intelligence (Anderson et al., 2015; Oota et al., 2019). However, these methods extract representations from each mode separately (image features from CNNs and text features from pretrained embeddings) and then perform a simple late-fusion. Thus, they cannot effectively exploit semantic correspondence across the two modes at different levels. Such late-fusion-based multi-modal models are the closest to our work, and our experiments show that our models outperform them.

Recently, Transformer-based models were found to be very effective than CNNs, in all language and image-related tasks (Devlin et al., 2019). Image-based transformer models like ViT (Dosovitskiy et al., 2020), DEiT (Touvron et al., 2021), and BEiT (Bao et al., 2021) have been shown to provide excellent results compared to traditional CNNs on image classification tasks. Also, multi-modal Transformers like VisualBERT (Li et al., 2019), LXMERT (Tan and Bansal, 2019), and CLIP (Radford et al., 2021) have shown excellent results on visio-linguistic tasks like visual question answering, visual common-sense reasoning. Inspired by the success of language, image, and multi-modal Transformers, we build multi-modal transformer models to learn the joint representations of image content and natural language and use them for brain encoding. Overall, in this work, we investigate whether *image-based and multi-modal Transformers* can accurately perform fMRI encoding on the *whole brain*. Fig. 1 illustrates our method for brain encoding.

Specifically, we make the following contributions to this paper. (1) We present state-of-the-art

[2]Quoting from (Kubilius et al., 2019): "After testing every layer on both V4 and IT, we report the model's score as the score of the best layer per region."

brain encoding results using multi-modal Transformers. We also study the effectiveness of our models in a cross-data setting. (2) Our approach generalizes the use of Transformer-based architectures, removing the need to manually select specific layers as in existing CNN-based fMRI encoding architectures. (3) We uncover several cognitive insights about the association between fMRI voxels and representations of multi-modal/image Transformers and CNNs.

2 Brain Imaging Datasets

The following datasets are popularly used in the literature for studying brain encoding: Vim-1 (Kay et al., 2008), Harry Potter (Wehbe et al., 2014; Pereira et al., 2018), BOLD5000 (Chang et al., 2019), Algonauts (Cichy et al., 2019), and SS-fMRI (Beliy et al., 2019). Vim-1 has only black and white images, is only related to object recognition, and is subsumed by BOLD5000. SS-fMRI is smaller and very similar to BOLD5000. The Harry Potter dataset does not have images. Lastly, fMRIs have not been made publicly available for the Algonauts dataset. Hence, we experiment with BOLD5000 and Pereira Pereira et al. (2018) datasets in this work.

BOLD5000: BOLD5000 dataset was collected from four subjects where three subjects viewed 5254 natural images (ImageNet: 2051, COCO: 2135, Scenes: 1068) while fMRIs were acquired. The fourth subject was shown 3108 images only. Details of the visual stimuli and fMRI protocols of the dataset have been discussed in (Chang et al., 2019). We briefly summarize the details of the dataset in Table 1. The data covers five visual areas in the human visual cortex, i.e., the early visual area (EarlyVis); object-related areas such as the lateral occipital complex (LOC); and scene related areas such as the occipital place area (OPA), the parahippocampal place area (PPA), and the retrosplenial complex (RSC). Each image also has corresponding text labels: ImageNet has a few out of 1000 possible tags per image, COCO has five captions per image, and Scenes have one out of 250 possible categories per image.

Pereira: For the Pereira dataset Pereira et al. (2018), participants were shown a concept word and a picture to observe brain activation when participants retrieved relevant meaning using visual information. Sixteen subjects were presented images (six per concept) corresponding to 180 concepts (abstract + concrete), while fMRIs were acquired. Out of 180 concepts, 116 are concrete, and others are abstract. Here, we augmented the image captions using the concept word associated with each image in the picture view. As in (Pereira et al., 2018), we focused on nine brain regions corresponding to four brain networks: Default Mode Network (DMN) (linked to the functionality of semantic processing), Language Network (related to language processing, understanding, word meaning, and sentence comprehension, Task Positive Network (related to attention, salience information), and Visual Network (related to the processing of visual objects, object recognition).

We show number of instances and voxel distribution across various brain regions for the BOLD5000 and Pereira datasets in Tables 1 and 2 respectively.

		Number of Voxels in Each ROI									
ROIs→		PPA		LOC		EarlyVis		OPA		RSC	
↓Subjects	#Instances	LH	RH	LH	RH	LH	RH	LH	RH	LH	RH
Subject-1	5254	131	200	152	190	210	285	101	187	86	143
Subject-2	5254	172	198	327	561	254	241	85	95	59	278
Subject-3	5254	112	161	430	597	522	696	187	205	78	116
Subject-4	3108	157	187	455	417	408	356	279	335	51	142

Table 1: BOLD5000 Dataset Statistics. LH=Left Hemisphere. RH=Right Hemisphere.

	Number of Voxels in Each ROI								
ROIs→	Language		Vision					DMN	Task Positive
↓Subj	LH	RH	Body	Face	Object	Scene	Vision	RH	LH
P01	5265	6172	3774	4963	8085	4141	12829	17190	35120
M01	5716	5561	3934	4246	7357	3606	12075	17000	34582
M02	4930	5861	3873	4782	7552	3173	11729	15070	30594
M03	3616	4247	2838	3459	5956	2822	9074	12555	24486
M04	5906	5401	3867	4803	7812	3602	12278	18011	34024
M05	4607	4837	2961	4023	6609	3135	10417	14096	28642
M06	4993	5099	3424	4374	7300	4058	11986	16289	30109
M07	5629	5001	4190	4993	8617	3721	12454	17020	30408
M08	5083	5062	2624	4082	6463	3503	10439	14950	29972
M09	3513	3650	2876	3343	5992	2815	9003	12469	25167
M10	5458	5581	3232	4844	7445	3474	11530	16424	29400
M13	4963	4811	2675	4008	5809	3323	9848	14489	30608
M15	5315	6141	4112	4941	8323	3496	12383	15995	31610
M16	4726	5534	4141	4669	8060	4142	12503	15104	31758
M17	5854	5698	4416	4801	8831	4521	13829	16764	37463

Table 2: Pereira Dataset Statistics. LH=Left Hemisphere. RH=Right Hemisphere.

3 Task Descriptions

We train fMRI encoding models using Ridge regression on stimuli representations obtained using various models for both datasets, as shown in Fig. 1. The main goal of each fMRI encoder model is to predict fMRI voxel values for each brain region given stimuli. In all cases, we train a model per subject separately. Different brain regions are involved in processing stimuli involving objects and scenes. Similarly, some regions specialize in understand-

ing vision inputs while others interpret linguistic stimuli better.

To evaluate the generalizability of our models across objects vs. scenes understanding, we also perform cross-data experiments where the train images belong to one sub-dataset, and the test images belong to the other sub-dataset. Thus, for each subject, we perform (1) three same-sub-dataset train-test experiments and (2) six cross-sub-dataset train-test experiments.

Full dataset fMRI Encoding: Whenever we train and test on the same dataset, we follow K-fold (K=10) cross-validation. All the data samples from K-1 folds were used for training, and the model was tested on samples of the left-out fold.

Cross-data fMRI Encoding: In the BOLD5000 dataset, we have three sub-datasets: COCO, ImageNet, and Scenes. ImageNet images mainly contain objects. Scenes images are about natural scenes, while COCO images relate to both objects and scenes. For each of the three sub-datasets, we perform K-fold (K=10) cross-validation within the sub-dataset.

4 Methodology

We trained a ridge regression-based encoding model to predict the fMRI brain activity associated with the stimuli representation for each brain region. Each voxel value is predicted using a separate ridge regression model. Formally, we encode the stimuli as $X \in \mathbb{R}^{N \times D}$ and brain region voxels $Y \in \mathbb{R}^{N \times V}$, where N denotes the number of training examples, D denotes the dimension of input stimuli representation, and V denotes the number of voxels in a particular region. Although ridge regression is a very naïve way of modeling, it has been the most popular brain encoding technique in this line of work. We plan to experiment with other forms of regression methods in the future.

The input stimuli representation can be obtained using any of the following models: (i) pretrained CNNs, (ii) pretrained text Transformers (iii) image Transformers, (iv) late-fusion models, or (v) multi-modal Transformers. The ridge regression objective function for the ith example is given as follows.

$$f(X_i) = \min_W \|Y_i - X_i W\|_F^2 + \lambda \|W\|_F^2$$

Here, W are the learnable weight parameters, $\|.\|_F$ denotes the Frobenius norm, and $\lambda > 0$ is a tunable hyper-parameter representing the regularization weight. λ was tuned on a small disjoint validation set obtained from the training.

Next, we discuss different input stimuli representation methods. Pretrained CNNs and Image Transformers encode image stimuli only, while Pretrained text Transformers encode text stimuli only. Late fusion models and Multi-modal Transformers encode both text and image stimuli.

Pretrained CNNs: Inspired by the Algonauts challenge (Cichy et al., 2019), we extract the layer-wise features from different pretrained CNN models such as VGGNet19 (Simonyan and Zisserman, 2014) (MaxPool1, MaxPool2, MaxPool3, MaxPool4, MaxPool5, FC6, FC7, FC8), ResNet50 (He et al., 2016) (Block1, Block2, Block3, Block4, FC), InceptionV2ResNet (Szegedy et al., 2017) (Conv2D5, Conv2D50, Conv2D100, Conv2D150, Conv2D200, Conv2D_7b), and EfficientNetB5 (Tan and Le, 2019) (Conv2D2, Conv2D8, Conv2D16, Conv2D24, FC), and use them for predicting fMRI brain activity. We use adaptive average pooling on each layer to get features for each image.

Pretrained text Transformers: RoBERTa (Liu et al., 2019) builds on BERT's language masking strategy and has been shown to outperform several other text models on the popular GLUE NLP benchmark. We use the average-pooled representation[3] from RoBERTa to encode text stimuli.

Image Transformers: We used three image Transformers: Vision Transformer (ViT), Data Efficient Image Transformer (DEiT), and Bidirectional Encoder representation from Image Transformer (BEiT). Given an image, image Transformers output two representations: pooled and patches. We experiment with both representations.

Late-fusion models: In these models, the stimuli representation is obtained as a concatenation of image stimuli encoding obtained from pretrained CNNs and text stimuli encoding obtained from pretrained text Transformers. Thus, we experiment with these late-fusion models: VGGNet19+RoBERTa, ResNet50+RoBERTa, InceptionV2ResNet+RoBERTa and EfficientNetB5+RoBERTa. These models do not incorporate real information fusion but do concatenation across modalities.

Multi-modal Transformers: We experiment with these multi-modal Transformer models: Contrastive Language-Image Pre-training (CLIP),

[3] Average-pooled representation gave us better results compared to using the CLS representation.

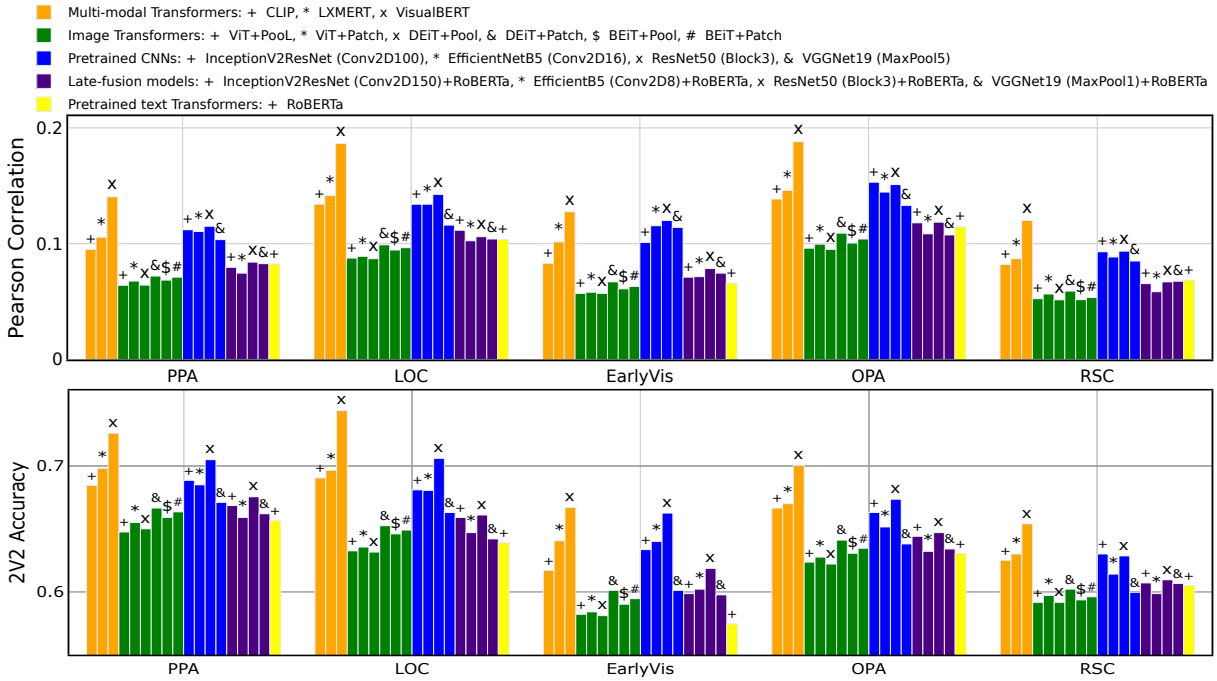

Fig. 2: BOLD5000 Results: Pearson correlation coefficient (top figure) and 2V2 (bottom figure) between predicted and actual responses across different brain regions using various models. Results are averaged across all participants. VisualBERT performs the best.

Learning Cross-Modality Encoder Representations from Transformers (LXMERT), and VisualBERT. These Transformers take both image and text stimuli as input and output a joint visio-linguistic representation. Specifically, the image input for these models comprises region proposals as well as bounding box regression features extracted from Faster R-CNN (Ren et al., 2015) as input features. These models incorporate information fusion across modalities at different levels of processing using co-attention and hence are expected to result in high-quality visio-linguistic representations.

Hyper-parameter Settings: We used sklearn's ridge-regression with default parameters, 10-fold cross-validation, Stochastic-Average-Gradient Descent Optimizer, Huggingface for Transformer models, MSE loss function, and L2-decay (λ) as 1.0. We used Word-Piece tokenizer for the linguistic Transformer input and Faster-RCNN (Ren et al., 2015) for extracting region proposals. All experiments were conducted on a machine with 1 NVIDIA GEFORCE-GTX GPU with 16GB GPU RAM. We make our code publicly available[1].

5 Experiments

5.1 Evaluation Metrics

We evaluate our models using popular brain encoding evaluation metrics described in the following. Given a subject and a brain region, let N be the number of samples. Let $\{Y_i\}_{i=1}^{N}$ and $\{\hat{Y}_i\}_{i=1}^{N}$ denote the actual and predicted voxel value vectors for the ith sample. Thus, $Y \in R^{N \times V}$ and $\hat{Y} \in R^{N \times V}$ where V is the number of voxels in that region.

$$\text{2V2 Accuracy} = \frac{1}{N_{C_2}} \sum_{i=1}^{N-1} \sum_{j=i+1}^{N} I[\{cosD(Y_i, \hat{Y}_i) + cosD(Y_j, \hat{Y}_j)\} < \{cosD(Y_i, \hat{Y}_j) + cosD(Y_j, \hat{Y}_i)\}]$$

where $cosD$ is the cosine distance function. $I[c]$ is an indicator function such that $I[c] = 1$ if c is true, else it is 0. The higher the 2V2 accuracy, the better. Pearson Correlation (PC) is computed as PC=$\frac{1}{N}\sum_{i=1}^{n} corr[Y_i, \hat{Y}_i]$ where corr is the correlation function.

5.2 Do multi-modal Transformers outperform other models?

Unfortunately, no previous work uses image Transformers or multi-modal Transformers for brain encoding. StepEnCog (Oota et al., 2019) is a late-fusion method, but it has a different setting where the model expects voxel values per brain slice rather than per brain region. Besides performing extensive evaluation using a large variety of models, we also compare our results with those obtained by two previously proposed baselines that lever-

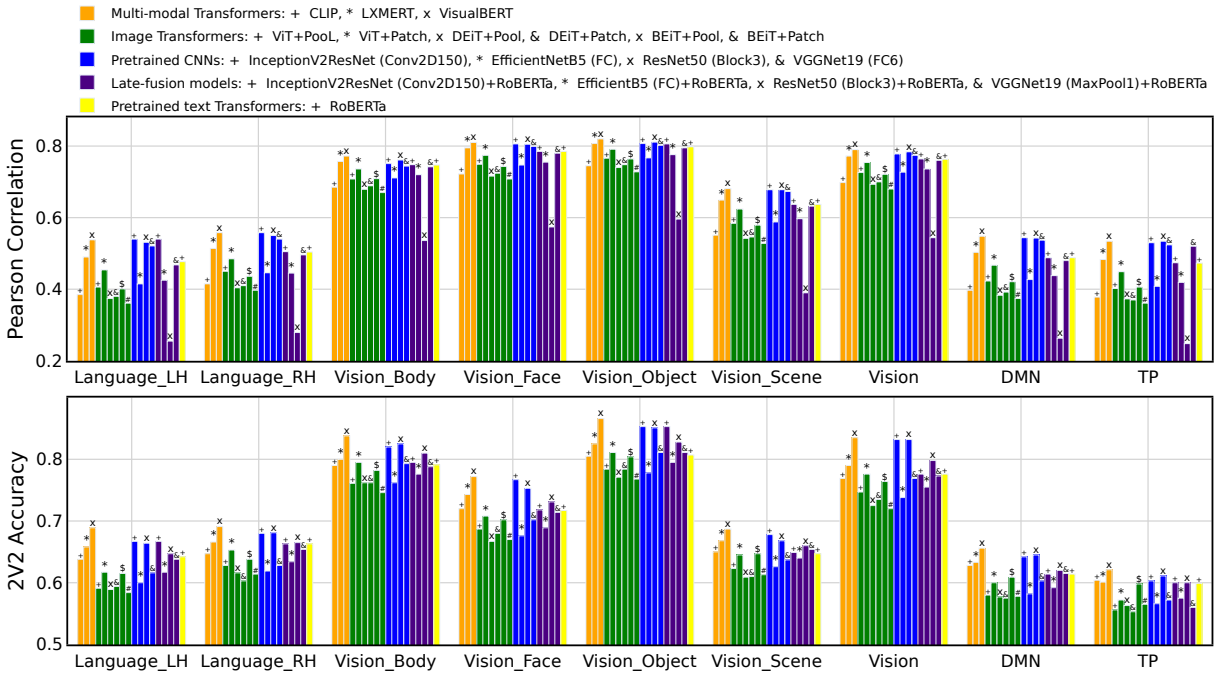

Fig. 3: Pereira Results: Pearson correlation coefficient (top figure) and 2V2 (bottom figure) between predicted and actual responses across different brain regions using a variety of models. Results are averaged across all participants. VisualBERT performs the best.

age pretrained CNN models: (Blauch et al., 2019) and (Wang et al., 2019) which use VGGNet.

We present the 2V2 accuracy and Pearson correlation results for models trained with different input representations (extracted from the best-performing layer of every pretrained CNN model and the last output layer of the Transformer model) on the two datasets: BOLD5000 and Pereira in Figs. 2 and 3, respectively. We also compare the results using many intermediate layer activations (not just the best) for CNN models and the last layer of Transformer models in Figs. 9 and 10 in the Appendix. Further, we also compare the results using all intermediate layer activations for Transformer models in Figs. 11 and 12 in the Appendix.

BOLD5000: We make the following observations from Fig. 2: (1) On both 2V2 accuracy and Pearson correlation, VisualBERT is better across all the models. (2) Other multi-modal Transformers such as LXMERT and CLIP perform as good as pretrained CNNs. We observed that image Transformers perform worse than pretrained CNNs. Late fusion models and RoBERTa has the least performance. (3) Late visual areas such as OPA (scene-related) and LOC (object-related) display a higher Pearson correlation with multi-modal Transformers, which is in line with the visual processing hierarchy. A higher correlation with all the visual brain ROIs with multi-modal Transformers demonstrates the power of jointly encoding visual and language information. (4) The patch representation of image Transformers shows an improved 2V2 accuracy and Pearson correlation compared to the Pooled representation. (5) Both InceptionV2ResNet and ResNet-50 have better performance among uni-modality models.

In order to estimate the statistical significance of the performance differences, we performed *post hoc* pairwise comparisons for all the subjects across the five brain ROIs. We found that VisualBERT is significantly better than LXMERT (second-best multi-modal Transformer) and InceptionV2ResNet (best pretrained CNN) for all ROIs except EarlyVis. Lastly, InceptionV2ResNet is significantly better than BEiT (best image Transformer) for all ROIs. Detailed p-values are mentioned in Table 3.

Pereira: We make the following observations from Fig. 3: (1) Similar to BOLD5000, multi-modal Transformers such as VisualBERT and LXMERT perform better. (2) Lateral visual areas such as

Models compared	PPA	LOC	EarlyVis	OPA	RSC
VisualBERT vs. LXMERT	0.044*	0.004*	0.076	0.049*	0.029*
VisualBERT vs. InceptionV2ResNet	0.049*	0.032*	0.521	0.041*	0.0354*
InceptionV2ResNet vs. BEiT	0.041*	0.003*	0.014*	0.188	0.203

Table 3: p-values for *post hoc* pairwise comparisons for BOLD5000 dataset

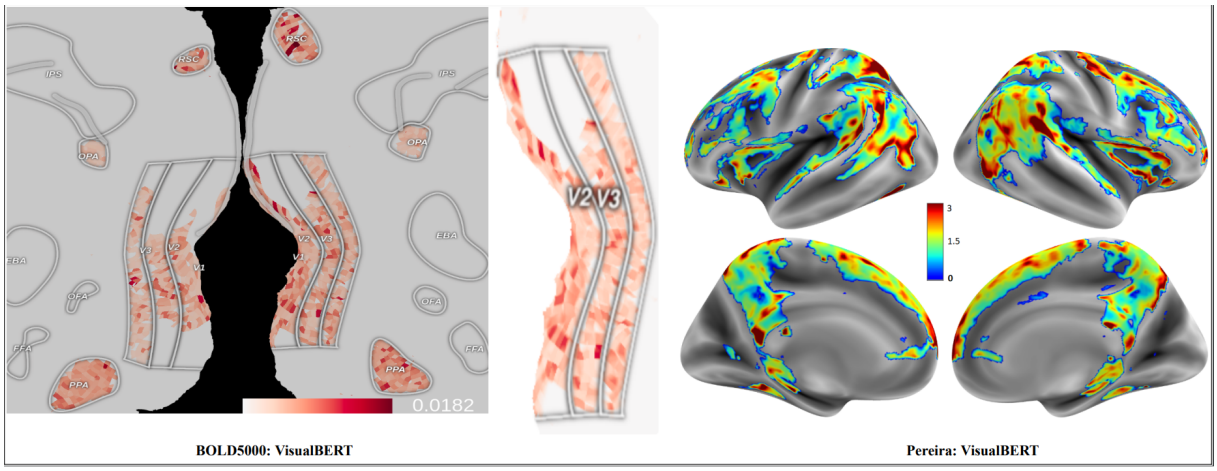

Fig. 4: MAE between actual and predicted voxels: (a) left figure is zoomed on V2 and V3 brain areas for VisualBERT on BOLD5000 subject 1. Note that V1 and V2 are also called EarlyVis areas, while V3 is also called LOC area. (b) the right figure is for VisualBERT on the Pereira dataset subject 2.

Models compared	Language_LH	Language_RH	Vision_Body	Vision_Face	Vision_Object	Vision_Scene	Vision	DMN	TP
VisualBERT vs. LXMERT	0.046*	0.039*	0.052	0.048*	0.046*	0.045*	0.047*	0.040*	0.035*
VisualBERT vs. ResNet	0.049*	0.038*	0.048*	0.048*	0.078	0.217	0.048*	0.046*	0.049*
ResNet vs. ViT	0.009*	0.043*	0.041*	0.047*	0.046*	0.042*	0.038*	0.022*	0.023*

Table 4: p-values for *post hoc* pairwise comparisons for Pereira dataset

Vision_Object, Vision_Body, Vision_Face, and Vision areas display higher correlation with multi-modal Transformers. A higher correlation with all the visual brain regions, language regions, DMN, and TP with multi-modal Transformers, demonstrates that the alignment of visual-language understanding helps.

In order to estimate the statistical significance of the performance differences, we performed *post hoc* pairwise comparisons for all the subjects across the nine brain ROIs. We found that VisualBERT is statistically significantly better than LXMERT (second-best multi-modal Transformer) for all ROIs except Vision_Body. Further, VisualBERT is statistically significantly better than ResNet (best pretrained CNN) for all ROIs except Vision_Object and Vision_Scene. Lastly, ResNet is statistically significantly better than ViT (best image Transformer) for all ROIs. Detailed p-values are mentioned in Table 4.

As further analysis, in Fig. 4, we show the mean absolute error (MAE) between the actual and predicted voxels across brain regions using VisualBERT. Comparing with similar brain charts for other models (shown in Figs. 13 and 14 in the Appendix), we notice that the error magnitudes are very small for the majority of the voxels. We observe that MAE values are relatively higher for EarlyVis areas and lowest for OPA for BOLD5000.

5.3 Model size vs. Efficacy Comparison

We plot a comparison of model size with Pearson Correlation averaged across all subjects for BOLD5000 in Fig. 5. Compared to LXMERT, VisualBERT is not just more accurate but also much smaller. VisualBERT is much more accurate than image Transformers while being almost the same size. Lastly, pretrained CNNs are smaller than VisualBERT but are less accurate even when the particular layer activations are cherry-picked. We observe similar trends for the Pereira dataset, as shown in Fig. 6. We hope that smaller models can be helpful for faster fine-tuning of new datasets.

5.4 Single Stream vs. Dual Stream Models

Since single stream (VisualBERT) and dual-stream (CLIP, LXMERT, and ViLBERT) models fuse language and images at different times. We report

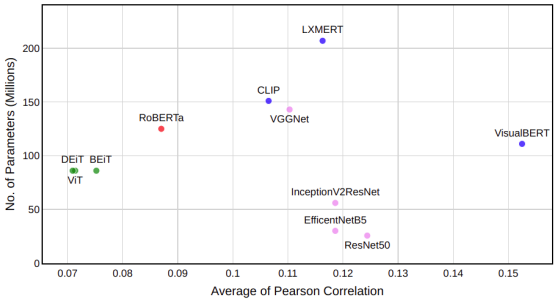

Fig. 5: BOLD5000: #Parameters vs. Avg Pearson Corr.

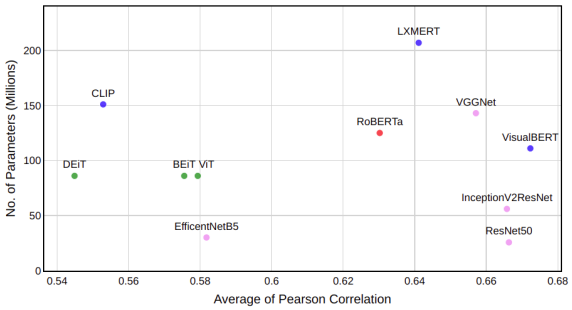

Fig. 6: Pereira: #Parameters vs. Avg Pearson Corr.

the comparison of single-stream vs. dual-stream with Pearson Correlation (PC) averaged across all subjects for BOLD5000 in Table 5 (top). Compared to dual-stream models (CLIP, LXMERT, and ViLBERT), VisualBERT showcases much better performance.

5.5 Is Linguistic Information Important in Multi-Modal Transformers?

Is the improvement in prediction performance of vision+language models over vision-only models due to the added linguistic information? For example, what happens if we randomize the language captions in BOLD5000, feed the model the correct image with a wrong caption, and train the encoding model to predict the correct image-elicited brain recording? We report the comparison of multi-modal transformers with correct caption vs. random caption using Pearson Correlation averaged across all subjects for BOLD5000 in Table 5. We observe that linguistic information is crucial to better performance with multi-modal Transformers.

5.6 Cross-Data fMRI Encoding

Fig. 7 illustrates PC for cross-data encoding on BOLD5000 using three multi-modal Transformers (VisualBERT, LXMERT, and CLIP). We also show results for a baseline method (Blauch et al., 2019). We observe that (1) multi-modal Transformers outperform the baseline results across all the five brain regions for cross-data tasks. (2) PC score is higher for the model trained on COCO and tested on ImageNet in the object-selective visual area LOC (lateral occipital cortex), which makes sense since COCO has many objects. (3) Similarly, the scene-selective brain areas such as RSC and OPA have a higher correlation for the COCO-Scenes, ImageNet-Scenes, and Scenes-Scenes tasks. (4) EarlyVisual areas have a lower correlation than other brain regions across the three tasks. (5) Overall, the models trained on COCO or ImageNet report a higher correlation than those trained on Scenes.

Models compared	PPA	LOC	EarlyVis	OPA	RSC
CLIP	0.095	0.134	0.083	0.139	0.082
LXMERT	0.106	0.142	0.102	0.146	0.087
VisualBERT	**0.141**	**0.187**	**0.128**	**0.188**	**0.12**
ViLBERT	0.057	0.078	0.052	0.087	0.045
CLIP-Random	0.020	0.024	0.033	0.031	0.002
LXMERT-Random	0.035	0.041	0.035	0.049	0.029
VisualBERT-Random	0.072	0.102	0.062	0.109	0.060
ViLBERT-Random	0.018	0.011	0.013	0.017	0.017

Table 5: Single stream (VisualBERT) vs. Dual stream (CLIP, LXMERT, and ViLBERT) models with BOLD5000: Pearson correlation computed between predicted and actual responses across different brain regions. Results are averaged across all participants. VisualBERT performs the best. The bottom four rows display the model performance when a random-caption is provided with the correct image as input.

6 Cognitive Insights: Does Language Influence Vision?

BOLD5000 dataset comprises brain responses from visual areas (early visual, scene-related, and object-related) when visual stimuli are presented to the subjects. Although only visual information is present in the stimuli, it is conceivable that participants implicitly invoke appropriate linguistic representations that, in turn, influence visual processing (Lupyan et al., 2020). Thus, it is not surprising that computational models such as multi-modal Transformers (VisualBERT, and LXMERT) that learn a joint representation of language and vision show superior performance on the 'purely' visual response data in BOLD5000 (see Figs. 2 and 4(a)).

Further, the performance of these models is naturally good in the case when text and image are shown to the participants, and whole-brain responses are captured as in the case of the Pereira dataset (see Figs. 3 and 4(b)). We further investigate the role of different sub-ROIs of the language and visual networks. For this, we compare the predicted responses of the best encoding model, i.e., VisualBERT, with the ground truth (observed) responses of various language and visual sub-regions (see Fig 8). We notice that the classical language areas in the temporal gyrus (LMTG and LPTG) and the inferior frontal gyrus (LIFG) are more accurately predicted than the other sub ROIs of the language network. These sub-ROIs (LMTG, LPTG, and LIFG) are highly involved in language comprehension and semantic processing. Interestingly, the second-best correlations are seen for multi-modal

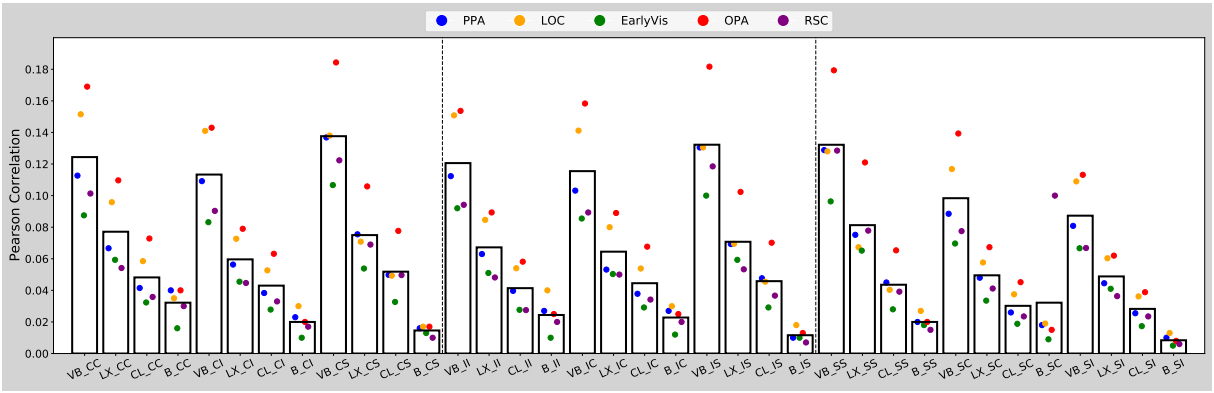

Fig. 7: Cross-Data Results for BOLD5000 dataset. VB=VisualBERT, LX=LXMERT, CL=CLIP, B=Baseline (Blauch et al., 2019), INC=InceptionV2ResNet. CC=Train and test on COCO, CI=Train on COCO and test on ImageNet, CS=Train on COCO and test on Scenes.)

integration areas in the temporo-parietal regions (LAngG, LFus, LPar) and higher-order processing and attention-related areas in the middle frontal region (MFG).

In the visual sub-ROIs (Fig. 8), we observe that the superior temporal sulcus (bilaterally but more in the left: LSTS) is more accurately predicted than other sub-ROIs. Surprisingly, LSTS is implicated in various social processes, ranging from language perception to simulating the mental processes of others. Also, the sub-ROIs such as LLOC, LFFA, LOFA, and LEBA have a higher correlation. These areas are involved in more visual-related functions such as object recognition, face perception, face recognition, and body recognition.

Based on the intuition from the computational experiments, we make the following testable prediction for future fMRI experiments. Instead of a passive viewing task, if participants were to perform a naming task/decision-making task on the objects/scenes, we expect to see more pronounced and focused activation in the visual areas during the language-based task compared to passive viewing.

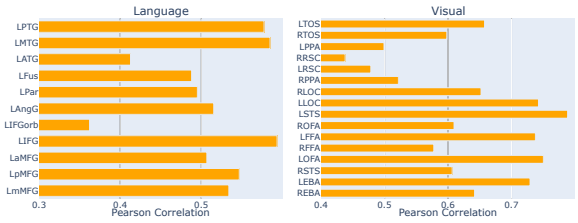

Fig. 8: Pearson correlation results across the Language Sub ROIs and Visual Sub ROIs for Pereira dataset, between predicted and true responses across different brain regions using a variety of models. Results are averaged across all participants and are obtained using VisualBERT.

7 Conclusion

We studied the effectiveness of multi-modal modeling for brain encoding. We found that VisualBERT, which jointly encodes text and visual input using cross-modal attention at multiple levels, performs the best. Our experiments on BOLD5000 and Pereira datasets lead to interesting cognitive insights. These insights indicate that fMRIs reveal reliable responses in scenes and object selection visual brain areas, which shows that cross-view decoding tasks like image captioning or image tagging are practically possible with reasonable accuracy. We plan to explore this as part of future work. We also plan to explore correlations between brain voxel space and representational feature space in the future. Finally, the combined strength of joint (audio, vision, and text) modalities remains to be investigated.

8 Ethical Statement

We reused publicly available datasets for this work: BOLD5000 and Pereira. We did not collect any new dataset. BOLD5000 dataset, except the stimulus images and their original annotations, is licensed under a Creative Commons 0 License. Please read their terms of use[4] for more details. Pereira dataset can be downloaded from https://osf.io/crwz7/. Please read their terms of use[5] for more details. We do not foresee any harmful uses of this technology.

[4] https://bold5000-dataset.github.io/website/terms.html

[5] https://github.com/CenterForOpenScience/cos.io/blob/master/TERMS_OF_USE.md

References

Andrew James Anderson, Elia Bruni, Alessandro Lopopolo, Massimo Poesio, and Marco Baroni. 2015. Reading visually embodied meaning from the brain: Visually grounded computational models decode visual-object mental imagery induced by written text. *NeuroImage*, 120:309–322.

Hangbo Bao, Li Dong, and Furu Wei. 2021. Beit: Bert pre-training of image transformers. *arXiv preprint arXiv:2106.08254*.

Pinglei Bao, Liang She, Mason McGill, and Doris Y Tsao. 2020. A map of object space in primate inferotemporal cortex. *Nature*, 583(7814):103–108.

Roman Beliy, Guy Gaziv, Assaf Hoogi, Francesca Strappini, Tal Golan, and Michal Irani. 2019. From voxels to pixels and back: self-supervision in natural-image reconstruction from fmri. In *Proceedings of the 33rd International Conference on Neural Information Processing Systems*, pages 6517–6527.

Nicholas M Blauch, Filipe De Avila Belbute Peres, Juhi Farooqui, Alireza Chaman Zar, David Plaut, and Marlene Behrmann. 2019. Assessing the similarity of cortical object and scene representations through cross-validated voxel encoding models. *Journal of Vision*, 19(10):188d–188d.

Nadine Chang, John A Pyles, Austin Marcus, Abhinav Gupta, Michael J Tarr, and Elissa M Aminoff. 2019. Bold5000, a public fmri dataset while viewing 5000 visual images. *Scientific data*, 6(1):1–18.

Radoslaw Martin Cichy, Gemma Roig, Alex Andonian, Kshitij Dwivedi, Benjamin Lahner, Alex Lascelles, Yalda Mohsenzadeh, Kandan Ramakrishnan, and Aude Oliva. 2019. The algonauts project: A platform for communication between the sciences of biological and artificial intelligence. *arXiv e-prints*, pages arXiv–1905.

R Todd Constable, Kenneth R Pugh, Ella Berroya, W Einar Mencl, Michael Westerveld, Weijia Ni, and Donald Shankweiler. 2004. Sentence complexity and input modality effects in sentence comprehension: an fmri study. *NeuroImage*, 22(1):11–21.

Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. 2009. Imagenet: A large-scale hierarchical image database. In *2009 IEEE conference on computer vision and pattern recognition*, pages 248–255. Ieee.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. Bert: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186.

Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, et al. 2020. An image is worth 16x16 words: Transformers for image recognition at scale. In *International Conference on Learning Representations*.

Jon Gauthier and Roger Levy. 2019. Linking artificial and human neural representations of language. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 529–539.

Umut Güçlü and Marcel AJ van Gerven. 2015. Deep neural networks reveal a gradient in the complexity of neural representations across the ventral stream. *Journal of Neuroscience*, 35(27):10005–10014.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. 2016. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778.

Andrew Jaegle, Felix Gimeno, Andrew Brock, Andrew Zisserman, Oriol Vinyals, and Joao Carreira. 2021. Perceiver: General perception with iterative attention. *arXiv preprint arXiv:2103.03206*.

Kendrick N Kay, Thomas Naselaris, Ryan J Prenger, and Jack L Gallant. 2008. Identifying natural images from human brain activity. *Nature*, 452(7185):352–355.

Tim C Kietzmann, Courtney J Spoerer, Lynn KA Sörensen, Radoslaw M Cichy, Olaf Hauk, and Nikolaus Kriegeskorte. 2019. Recurrence is required to capture the representational dynamics of the human visual system. *Proceedings of the National Academy of Sciences*, 116(43):21854–21863.

Jonas Kubilius, Martin Schrimpf, Kohitij Kar, Rishi Rajalingham, Ha Hong, Najib Majaj, Elias Issa, Pouya Bashivan, Jonathan Prescott-Roy, Kailyn Schmidt, et al. 2019. Brain-like object recognition with high-performing shallow recurrent anns. *Advances in Neural Information Processing Systems*, 32:12805–12816.

Liunian Harold Li, Mark Yatskar, Da Yin, Cho-Jui Hsieh, and Kai-Wei Chang. 2019. Visualbert: A simple and performant baseline for vision and language. *arXiv preprint arXiv:1908.03557*.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Roberta: A robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692*.

Gary Lupyan, Rasha Abdel Rahman, Lera Boroditsky, and Andy Clark. 2020. Effects of language on visual perception. *Trends in cognitive sciences*.

Subba Reddy Oota, Vijay Rowtula, Manish Gupta, and Raju S Bapi. 2019. Stepencog: A convolutional lstm autoencoder for near-perfect fmri encoding. In *2019 International Joint Conference on Neural Networks (IJCNN)*, pages 1–8. IEEE.

Francisco Pereira, Bin Lou, Brianna Pritchett, Samuel Ritter, Samuel J Gershman, Nancy Kanwisher, Matthew Botvinick, and Evelina Fedorenko. 2018. Toward a universal decoder of linguistic meaning from brain activation. *Nature communications*, 9(1):1–13.

Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, et al. 2021. Learning transferable visual models from natural language supervision. *Image*, 2:T2.

Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, Ilya Sutskever, et al. 2019. Language models are unsupervised multitask learners.

Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun. 2015. Faster r-cnn: Towards real-time object detection with region proposal networks. *Advances in neural information processing systems*, 28:91–99.

Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael Bernstein, et al. 2015. Imagenet large scale visual recognition challenge. *International journal of computer vision*, 115(3):211–252.

Martin Schrimpf, Idan Blank, Greta Tuckute, Carina Kauf, Eghbal A Hosseini, Nancy Kanwisher, Joshua Tenenbaum, and Evelina Fedorenko. 2020a. The neural architecture of language: Integrative reverse-engineering converges on a model for predictive processing. *BioRxiv*.

Martin Schrimpf, Jonas Kubilius, Ha Hong, Najib J Majaj, Rishi Rajalingham, Elias B Issa, Kohitij Kar, Pouya Bashivan, Jonathan Prescott-Roy, Franziska Geiger, et al. 2020b. Brain-score: Which artificial neural network for object recognition is most brain-like? *BioRxiv*, page 407007.

Dan Schwartz, Mariya Toneva, and Leila Wehbe. 2019. Inducing brain-relevant bias in natural language processing models. *Advances in Neural Information Processing Systems*, 32:14123–14133.

Karen Simonyan and Andrew Zisserman. 2014. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*.

Christian Szegedy, Sergey Ioffe, Vincent Vanhoucke, and Alexander A Alemi. 2017. Inception-v4, inception-resnet and the impact of residual connections on learning. In *Thirty-first AAAI conference on artificial intelligence*.

Hao Tan and Mohit Bansal. 2019. Lxmert: Learning cross-modality encoder representations from transformers. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 5100–5111.

Mingxing Tan and Quoc Le. 2019. Efficientnet: Rethinking model scaling for convolutional neural networks. In *International Conference on Machine Learning*, pages 6105–6114. PMLR.

Bertrand Thirion, Edouard Duchesnay, Edward Hubbard, Jessica Dubois, Jean-Baptiste Poline, Denis Lebihan, and Stanislas Dehaene. 2006. Inverse retinotopy: inferring the visual content of images from brain activation patterns. *Neuroimage*, 33(4):1104–1116.

Hugo Touvron, Matthieu Cord, Matthijs Douze, Francisco Massa, Alexandre Sablayrolles, and Hervé Jégou. 2021. Training data-efficient image transformers & distillation through attention. In *International Conference on Machine Learning*, pages 10347–10357. PMLR.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in neural information processing systems*, pages 5998–6008.

Aria Wang, Michael Tarr, and Leila Wehbe. 2019. Neural taskonomy: Inferring the similarity of task-derived representations from brain activity. *Advances in Neural Information Processing Systems*, 32:15501–15511.

Leila Wehbe, Brian Murphy, Partha Talukdar, Alona Fyshe, Aaditya Ramdas, and Tom Mitchell. 2014. Simultaneously uncovering the patterns of brain regions involved in different story reading subprocesses. *in press*.

Daniel Yamins, Ha Hong, Charles Cadieu, and James J DiCarlo. 2013. Hierarchical modular optimization of convolutional networks achieves representations similar to macaque it and human ventral stream.

Daniel LK Yamins and James J DiCarlo. 2016. Using goal-driven deep learning models to understand sensory cortex. *Nature neuroscience*, 19(3):356–365.

Daniel LK Yamins, Ha Hong, Charles F Cadieu, Ethan A Solomon, Darren Seibert, and James J DiCarlo. 2014. Performance-optimized hierarchical models predict neural responses in higher visual cortex. *Proceedings of the national academy of sciences*, 111(23):8619–8624.

A Do multi-modal Transformers perform better encoding compared to intermediate layer representations from pretrained CNNs?

We present the 2V2 accuracy and Pearson correlation for models trained with representations extracted from the last layer of multi-modal Transformers and all the lower to higher-level representations from pretrained CNNs on the two datasets: BOLD5000 and Pereira in Figs. 9 and 10, respectively.

We make the following observations from Fig. 9: (1) With respect to 2V2 and Pearson correlation, the multi-modal Transformer, VisualBERT, performs better than all the internal representations of pretrained CNNs. (2) In the pretrained CNNs, intermediate blocks have better correlation scores as compared to lower or higher level layer representations. (3) Other multi-modal Transformers, CLIP, and LXMERT, have marginal improvements over all the models except intermediate blocks such as Conv2D150 in InceptionV2ResNet.

We make the following observations from Fig. 10: (1) With respect to 2V2 and Pearson correlation, the multi-modal Transformer, VisualBERT, performs better than all the internal representations of pretrained CNNs. (2) Similar to BOLD5000, the intermediate blocks have better correlation scores as compared to lower or higher level layer representations in the pretrained CNNs on Pereira Dataset. (3) Other multi-modal Transformer, LXMERT, have equal performance with intermediate blocks of each pretrained CNN model.

B Do multi-modal Transformers perform better encoding in their layers?

Given the hierarchical processing of visual or visual-language information across the Transformer layers, we further examine how these Transformer layers encode fMRI brain activity using image and mulit-modal Transformers. We present the layer-wise encoding performance results on two datasets: BOLD5000 and Pereira in Figs. 11 and 12, respectively.

We make the following observations from Fig. 11: (i) The multi-modal Transformer, VisualBERT, have consistent performance across the layers from 1 to 12. (ii) The LXMERT model have marginal decreasing performance from intermediate layer (L7) to higher layers. (iii) The image Transformers have higher Pearson correlation for early visual areas in the lower layers whereas higher visual areas such as LOC, OPA, and PPA have an increasing correlation in higher layers. (iv) This clearly indicates that the hierarchy of processing of visual stimulus in the human brain is similar to image Transformer layers.

We make the following observations from Fig. 12: (i) The multi-modal Transformers, VisualBERT, have consistent performance across the layers from 1 to 12. (ii) The LXMERT model have marginal decreasing performance from lower to higher layers. (iii) The image Transformer, ViT, has higher Pearson correlation for early visual areas in the lower layers whereas higher visual areas such as Vision_Body, Vision_Face, and Vision_Obj have an increasing correlation in higher layers.

C Brain Maps for various models for BOLD5000 Dataset

Fig. 13 shows mean absolute errors (MAE) between actual and predicted voxels for various models on the BOLD5000 dataset. Notice that the magnitude of errors is much higher for a majority of voxels, compared to that with the VisualBERT model as shown in Fig. 4(a). Also, the multi-modal Transformers, VisualBERT (MAE range: 0 to 0.0181) and LXMERT (MAE range: 0 to 0.0188), have lower MAE compared to both image Transformers (MAE range: 0 to 0.02) and pretrained CNNs (MAE range: 0 to 0.0236).

D Brain Maps for various models for Pereira Dataset

Fig. 14 shows mean absolute errors (MAE) between actual and predicted voxels for various models on the Pereira dataset. Notice that the magnitude of errors is much higher for a majority of voxels, compared to that with the VisualBERT model as shown in Fig. 14(a). Also, the multi-modal Transformers, VisualBERT and LXMERT, and InceptionV2ResNet+Conv2D150 have lower MAE compared to both image Transformers and other pretrained CNNs.

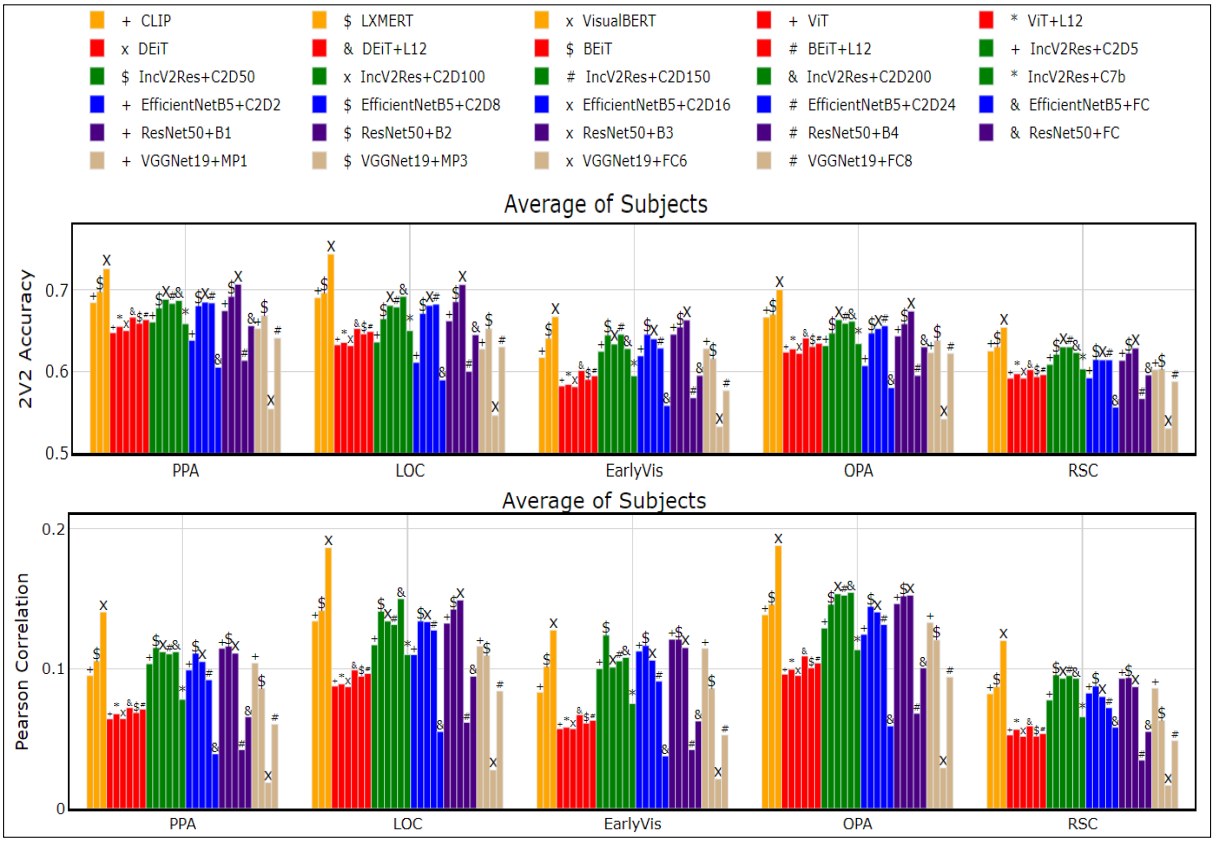

Fig. 9: BOLD5000: 2V2 (top Fig.) and Pearson correlation coefficient (bottom Fig.) between predicted and true responses across different brain regions using variety of models. Results are averaged across all participants. Pretrained CNN results are shown for all layers while multi-modal Transformer results are shown for last layers only.

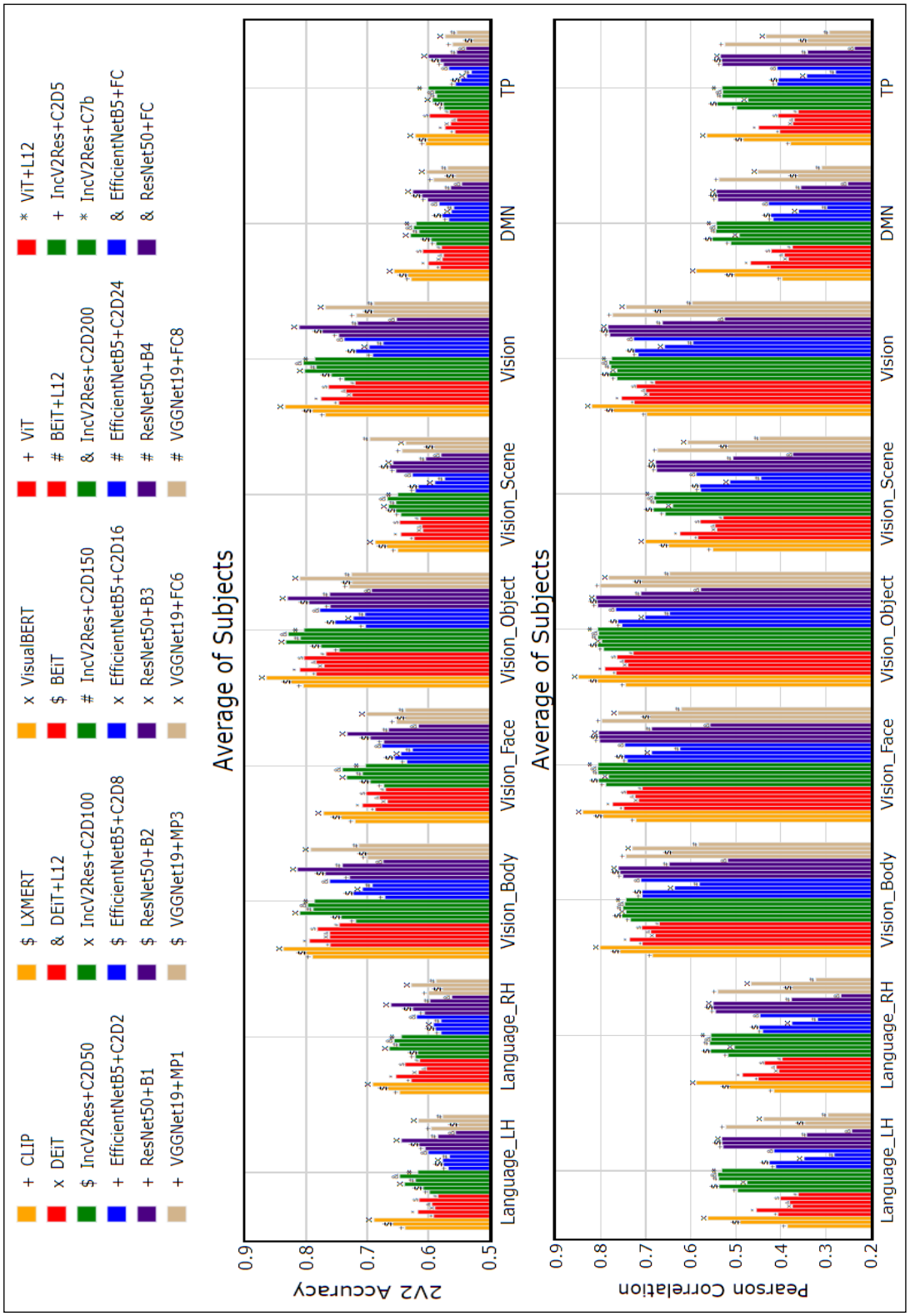

Fig. 10: Pereira dataset: 2V2 (top Fig.) and Pearson correlation coefficient (bottom Fig.) between predicted and true responses across different brain regions using variety of models. Results are averaged across all participants. Pretrained CNN results are shown for all layers while multi-modal Transformer results are shown for last layers only.

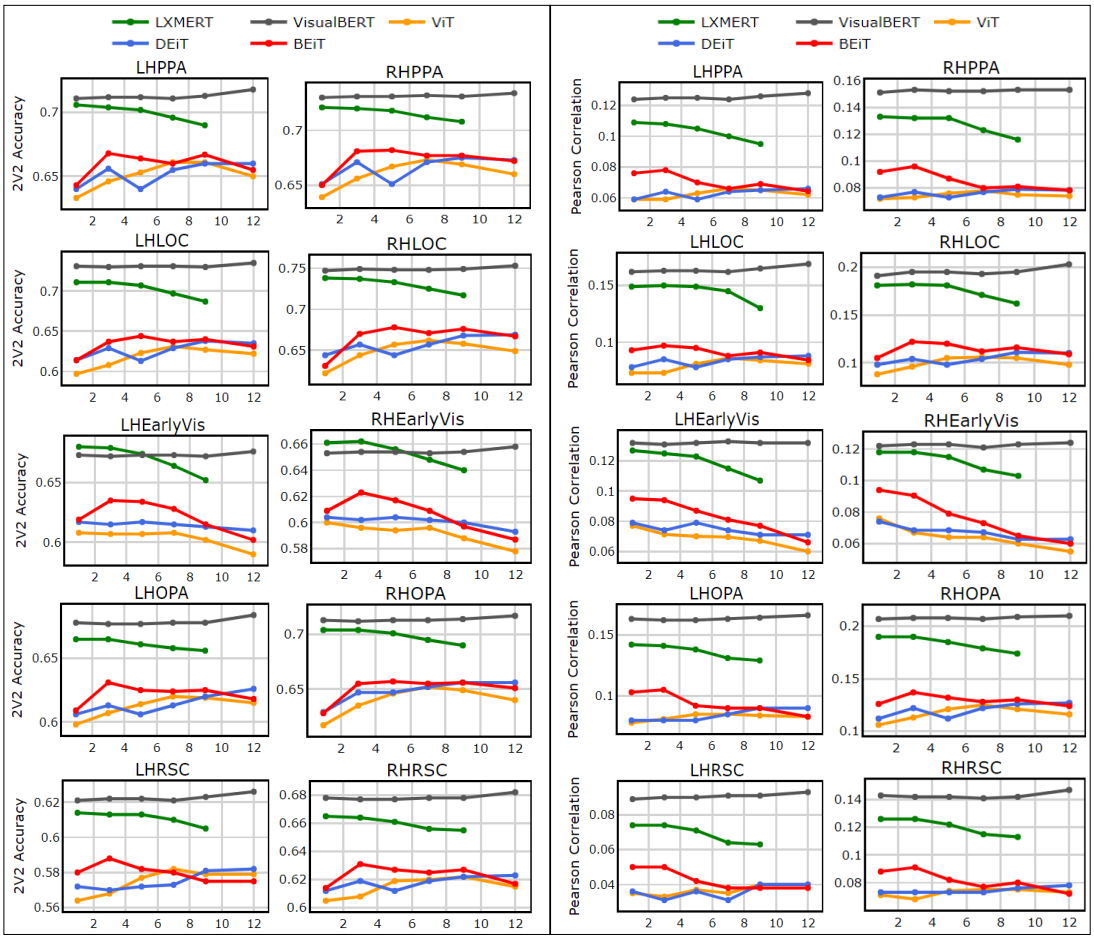

Fig. 11: BOLD5000: 2V2 (left) and Pearson correlation coefficient (right) between predicted and true responses across different brain regions using Transformer models. Results are averaged across all participants. The results are shown for all layers of image and multi-modal Transformers. Note that LXMERT has only 9 layers.

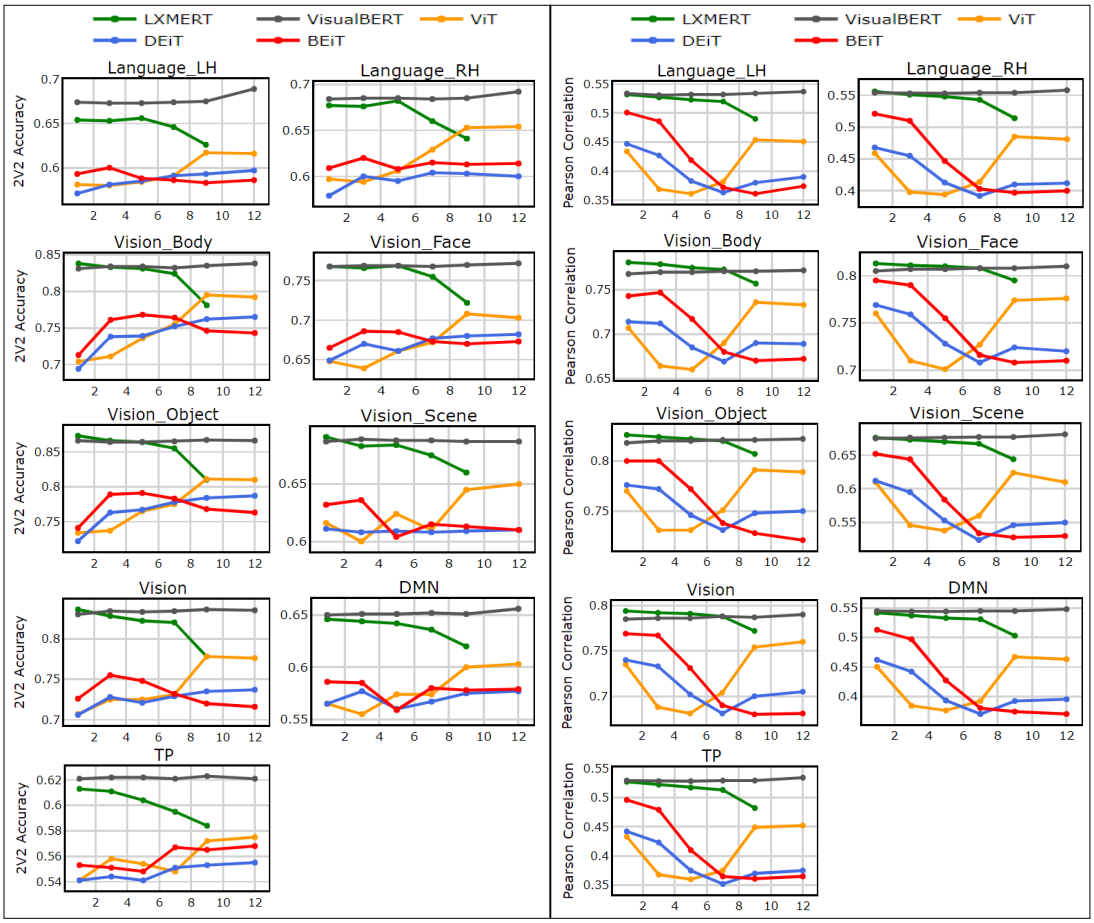

Fig. 12: Pereira: 2V2 (left) and Pearson correlation coefficient (right) between predicted and true responses across different brain regions using Transformer models. Results are averaged across all participants. The results are shown for all layers of image and multi-modal Transformers. Note that LXMERT has only 9 layers.

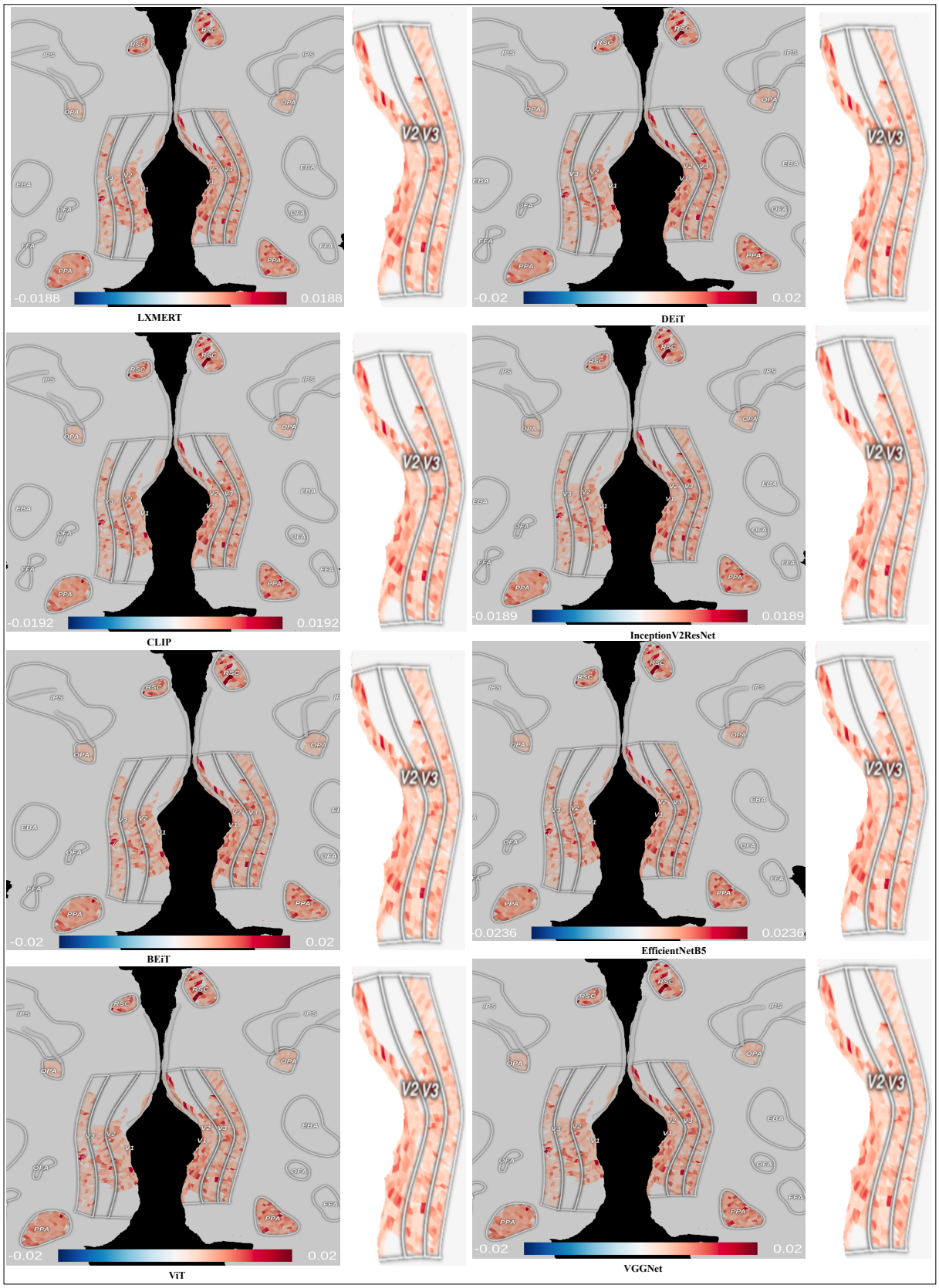

Fig. 13: MAE between actual and predicted voxels zoomed on V2 and V3 brain areas for various models. Note that V1 and V2 are also called EarlyVis area, while V3 is also called LOC area.

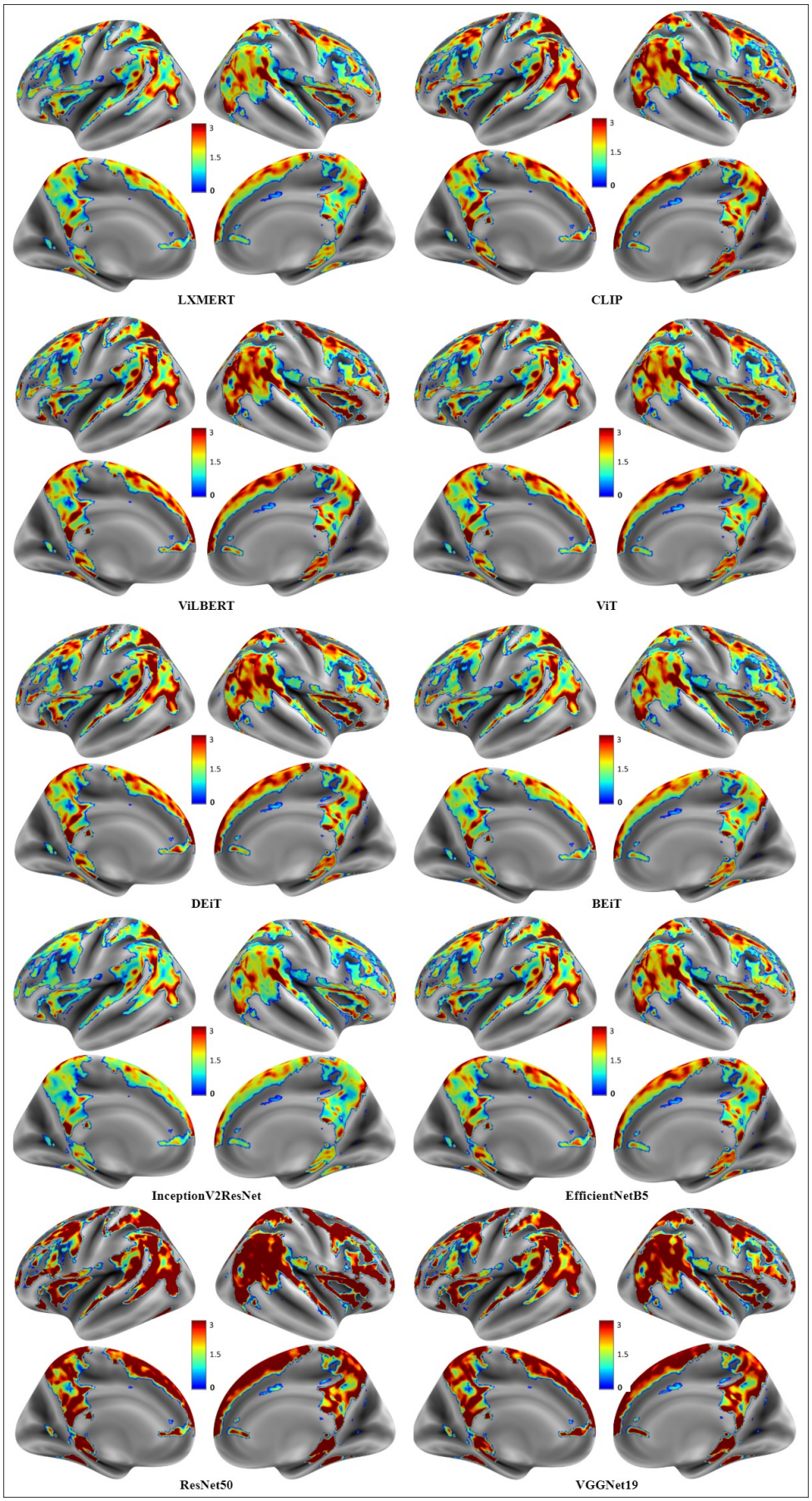

Fig. 14: MAE between actual and predicted voxels zoomed on V2 and V3 brain areas for various models. Note that V1 and V2 are also called EarlyVis area, while V3 is also called LOC area.

Gestures Are Used Rationally: Information Theoretic Evidence from Neural Sequential Models

Yang Xu
Department of Computer Science
San Diego State University
yang.xu@sdsu.edu

Yang Cheng
University of Southern California
ycheng04@usc.edu

Riya Bhatia
Carnegie Mellon University
riyabhat@andrew.cmu.edu

Abstract

Verbal communication is companied by rich non-verbal signals. The usage of gestures, poses, and facial expressions facilitates the information transmission in verbal channel. However, few computational studies have explored the non-verbal channels with finer theoretical lens. We extract gesture representations from monologue video data and train neural sequential models, in order to study the degree to which non-verbal signals can effectively transmit information. We focus on examining whether the gestures demonstrate the similar pattern of entropy rate constancy (ERC) found in words, as predicted by Information Theory. Positive results are shown to support the assumption, which leads to the conclusion that speakers indeed use simple gestures to convey information that enhances verbal communication, and the production of non-verbal information is rationally organized.

1 Introduction

Communication is a multi-modal process, in which information from verbal and non-verbal modalities are mixed into one channel. It has been revealed from a long history of empirical studies that speakers' expression in visual modality, including gestures, body poses, eye contacts and other types of non-verbal behaviors, play critical roles in face-to-face communication, as they add subtle information that is hard to convey in verbal language (Pease and Pease, 2008; Krauss et al., 1996). However, it remains an untested idea to view these sparse and random non-verbal signals as a formal communication channel that transmits "serious" information, which has seldom been validated by computational studies. A key missing step is to explore whether the non-verbal information can be quantified.

The questions that are worth further investigation include: How rich is the information contained in these non-verbal channels? What are their relationships to verbal information? Can we understand the meanings of different gestures, poses, and motions embedded in spontaneous language in a similar way to understanding word meanings? The goal of this study is to take a simple yet necessary first step approaching the above questions, by examining a basic Information Theoretic property of gestures that comes along with verbal language. Some preliminary but prospective results are presented.

2 Related Work

2.1 Gestures as non-verbal communication

There is vast literature on the connection between gesture and language in human communication. Gestures, defined as "the spontaneous hand movements produced in rhythm with speech" (Clough and Duff, 2020) naturally co-occur with spoken language. According to the thorough survey from (Clough and Duff, 2020), the *communication* function of gestures is one of the main focus of early studies. McNeill (1992) has classified gestures into two categories, representative and non-representative, in which the former has clearer semantic meanings (e.g., depicting objects and describing locations), while the latter refers to the brief, repetitive movements that has little substantive meanings.

2.2 Non-verbal communication in natural language processing

The recent advances of deep neural network-based machine learning techniques provide new methods to understand the non-verbal components of human communication. Many existing works primarily focus on using multi-modal features as clues for a variety of inference tasks, including video content understanding and summarization (Li et al., 2020; Bertasius et al., 2021), as well as more specific ones such as predicting the shared attention among speakers (Fan et al., 2018) and semantic-aware action segmentation (Gavrilyuk et al., 2018; Xu et al.,

Proceedings of the 29th International Conference on Computational Linguistics, pages 134–140
October 12–17, 2022.

2019). More recently, models that include multiple channels have been developed to characterize context-situated human interactions (Fan et al., 2021). Advances in representation learning have enabled researchers to study theoretical questions with the tools of multi-modal language models.

2.3 Information theories

Information theory (Shannon, 1948) has been broadly applied in computational linguistics as the theoretic background for the probabilistic models of language. This also provides philosophical explanations to a broad spectrum of linguistic phenomena. One example that interests researchers the most is the assumption/principle of *entropy rate constancy* (ERC). Under this assumption, communication in any form (written or spoken) should optimize the rate of information transmission rate by keeping the overall entropy rate constant.

In natural language, *entropy* refers to the predictability of words (tokens, syllables) estimated with probabilistic language models. Genzel and Charniak (2002, 2003) first formulated a method to examine ERC for written language, by decomposing the entropy term into *local* and *global* entropy:

$$H(s|context) = H(s|L) - I(s, C|L) \quad (1)$$

in which s can be any symbol whose probability can be estimated, such as a word, punctuation, or sentence. C and L refer to the global and local contexts for s, among which C is purely conceptual and only L can be operationally defined. By ERC, the left term in eq. (1) should remain an invariant against the position of s. It results in an expectation that the first term on the right $H(s|L)$ should *increase* with the position of s, because the second term $I(s, C|L)$, i.e., the mutual information between s and itself global context should always decrease (see Genzel and Charniak (2003)'s paper). Xu and Reitter (2016, 2017, 2018) has confirmed the pattern in spoken language.

Now, the goal of this study is to extend the application scope of ERC to the non-verbal realm. If the s in eq. (1) represents any symbol that carries information, for example, a gesture, then the same *increase* pattern should be observed within a sequence of gestures. ERC can be interpreted as a "rational" strategy for the information sender (speaker) because it requires less predictable content (higher local entropy) to occur at a later position within the message, which maximizes the likelihood for the receiver (listener) to successfully decode information with the least effort. The question here is to examine whether we "speak" rationally by gestures.

3 Question and Hypothesis

Our hypothesis is: non-verbal communication also conforms to the principle of ERC. To test it, we approximate the local entropy ($H(s|L)$) of non-verbal "tokens" using the perplexity scores obtained from neural sequential models, and correlate it with the utterances' relative positions within the monologue data. If we can find that $H(s|L)$ increases with utterance position, is similar to verbal language, then it supports the hypothesis.

4 Methods

4.1 Data collection and pre-processing

The video data that we use is collected from several YouTube channels. All the videos are carefully selected based on the standards that each video must contain only one speaker who faces in front of the camera, and whose hands must be visible. 12 videos from 5 hosts are collected, and the mean duration is 15.0 minutes ($SD = 7.0$).

The pre-processing step is to extract the full-body landmark points of the speaker, in preparation for the next gesture representation step. For this task, we use BlazePose (Bazarevsky et al., 2020), which is a lightweight convolutional neural network-based pose estimation model provided in `MediaPipe`[1]. It outputs 33 pose landmarks of the human body detected in each frame.

4.2 Extract gesture labels

The next step is to represent gestures so that they can be encoded by the neural sequential model. There are various ways of creating *continuous* representations for gestures/poses, such as the pose embedding technique (Mori et al., 2015). However, it is difficult to obtain a set of gestures that are *universal* across speakers using such continuous representations. Thus, for the purpose of this study, we extract *discrete* gesture labels, by categorizing the hands positions into grids. We divide the front space of speaker into 3×3 regions, i.e., indicated by integer numbers from 1 to 9. Each *hand* is assigned a number based on which region it falls into. Next, we use the combination of both hands

[1] https://google.github.io/mediapipe/

to create a unique gesture label for that frame. For example, as shown in fig. 1b, the speaker's left and right hands fall into region 9 and 8, which determines its gesture label as <72>. For convenience, we use one integer ID (instead of the merged ID connected by a hyphen) to denote each of these 81 gestures: <1>, <2>, ..., <81>. The total number of gesture labels is $9 \times 9 = 81$. Note that 81 is the theoretical maximum number, and the actual count depends on the size of data.

4.3 Prepare gesture sequences

After obtaining the discrete gesture labels for all video frames, we prepare the gesture sequences based on the time stamped text transcript for each video. We use the automatically generated text transcript in `.vtt` format, which contains the *start* and *end* time stamps for each word token in the subtitle. See the following example:

```
<00:00:00.510><c> let's</c>
<00:00:00.780><c> talk</c>
<00:00:01.020><c> about</c>
```

in which each the start time stamp is appended to the head, and the start time for a token is the end time for the previous token. In this example, the token *talk* elapses from 0.780 to 1.020 in seconds. Multiplying the time stamps with frame rate (24 FPS) returns that the word elapses from the 19th frame to the 24th. Then, for each frame within the range of $[19, 24]$, we extract a gesture label using the method described in Section 4.2, resulting in a sequence of gesture labels, $[g_{19}, g_{20}, \ldots, g_{24}]$. This sequence represents a continuous change of gestures during the articulation of the corresponding word, which in most cases, consists of identical gesture labels. Therefore, we select the median label g among $[g_{19}, \ldots, g_{24}]$ as a compact representation.

For a sentence of N wordss, we obtain the median gesture label for each token, $\{g_1, g_2, \ldots, g_N\}$. Despite the down sampling effect of using the median label, there is still large amount of repetition in the resulted sequence. For example, in the first row of table 1, the median gesture label is the same <24> for the first 6 tokens, which means that the speaker did not move his/her hands during that period of time. It makes sense that we treat these repeated gesture labels just as one label. By merging the 6 repeats of <24> and 2 repeats of <36>, we get a compressed gesture sequence, {<36>, <24>}, which means the speaker has made two distinct gestures during the utterance. For each median gesture sequence of length N, we obtain its compressed version $\{\hat{g}_1, \hat{g}_2, \ldots, \hat{g}_{N'}\}$, where $N' \leq N$. See table 1 for examples.

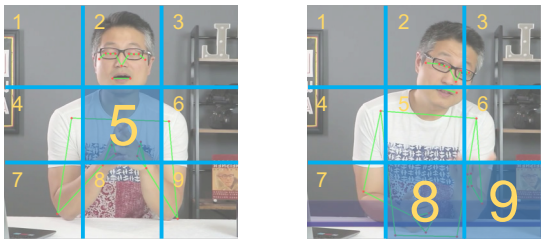

(a) Both hands in region 5 → label <25>.

(b) Right hand in region 9, left in 8 → label <72>.

Figure 1: Create discrete gesture labels based on landmark positions of both hands.

4.4 Sequential models for gesture input

We implement two neural network-based models for the sequential modeling tasks, using LSTM (Hochreiter and Schmidhuber, 1997) and Transformer (Vaswani et al., 2017) encoders. The model takes as input a sequence of gesture labels (median g or compressed $\hat{g}$) and convert them to the embedding space. Then the gesture embeddings are fed to the LSTM/Transformer encoders to capture the temporal dependency between gestures, which compute a dense representation for gestures at each time step. Lastly, the dense representation at the previous time step is used to predict the gesture label at the next time step using a softmax output. The model architecture is shown in fig. 2.

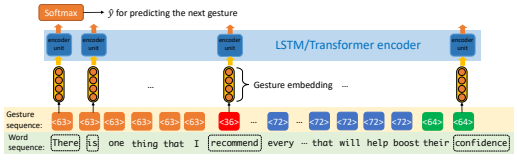

Figure 2: Architecture of the sequential model for encoding gesture input and next time-step prediction.

The learning task is to predict the next gesture label, i.e., minimizing the negative log probability:

$$NLL = -\sum_{t=1}^{T} \log P(g_t | g_1, g_2, \ldots, g_{t-1}) \quad (2)$$

in which $g_1, \ldots, g_{t-1}$ is all the gesture tokens before g_t within the same utterance. An exponential conversion of eq. (2) leads to the local entropy term,

Word tokens in utterance	Median gesture g of each token	Compressed gesture sequence $\hat{g}$
going to give you a flatter look glossy	<24> <24> <24> <24> <24> <24> <36> <36> ($N = 8$)	<24> <36> ($N' = 2$)
now this is really your preference	<40> <72> <64> <64> <40> <40> ($N = 6$)	<40> <72> <64> <40> ($N' = 4$)
I think most of us can get on board	<63> <63> <63> <63> <63> <63> <63> <63> <63> ($N = 9$)	<63> ($N' = 1$)

Table 1: Examples of gesture sequences. Integers wrapped by "<>" are gesture labels. For each sequence, its compressed version is shorter in length: $N' < N$

.

$H(g|L) = \exp(NLL)$, which is the target variable of our interest. This learning task is no different from conventional language modeling tasks, except that the input here is non-verbal tokens. Detailed model hyper-parameters and training procedures are included in appendix A.1.

5 Results

5.1 Summary of data

53 videos are collected from 4 YouTube channels (i.e., 4 distinct speakers). The average length of videos is 723.7 seconds (*SD* = 438.1). There are 17.9K lines of automatically generated subtitles consisting of 121.5K word tokens in total. 81 distinct gesture labels are extracted. The total count of the median gesture label is the same as that of the word tokens (121.5K). The compressed gesture labels has a much smaller total count 26120.

The top 5 most frequent gesture labels are <63>, <56>, <64>, <72> and <36>. The frequency distribution of gesture labels roughly follows the Zipf's law, which is a common distribution pattern in natural language data (Zipf, 2013; Piantadosi, 2014) (See fig. 3). Gesture label <63> is the dominant gesture throughout the data. It is gestural position where the speaker's right hand (from his/her perspective) is in region 7, and left hand region 9.

5.2 Examine hypothesis: local entropy increases with utterance position

The local entropy of each gesture sequence (median and compressed, respectively) is plotted against the corresponding utterance's position in fig. 4, which shows a visible increasing trend.

We use linear models to verify the correlations between local entropy and utterance position. It is confirmed that utterance position is a significant predictor of local entropy with positive β coefficients. For raw gestures, the *betas* are smaller: $\beta_{\text{LSTM}} = 1.6 \times 10^{-3}$ ($p < .05$), $\beta_{\text{Trm}} = 2.3 \times 10^{-3}$ ($p < .01$); for compressed gestures: $\beta_{\text{LSTM}} = 0.097$, $\beta_{\text{Trm}} = 0.093$ ($p < .001$). Therefore, the increase of local entropy is statistically significant. It supports our hypothesis.

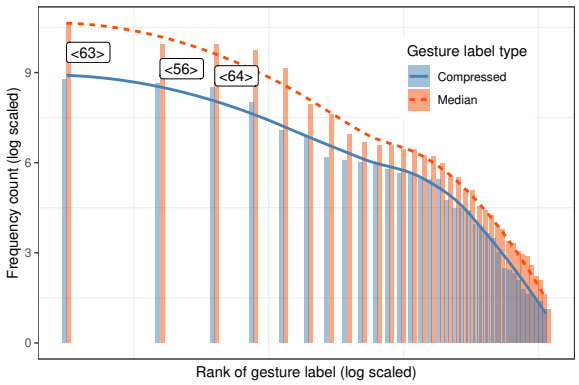

Figure 3: Frequency count against the rank gesture labels in logarithm transformed scales. Top three most frequent gesture labels annotated.

5.3 Analysis of typical gesture

We select three highly frequent gesture labels <63>, <56> and <72>, and show some typical screenshots in fig. 5. In these gestures, the positions of both hands are at the mid-lower position in front of the body. Gesture <63> has two hands evenly distant from the center, while gesture <56> captures a movement to the right and gesture <72> to the left. In general, these are very commonly seen patterns in daily communication.

6 Discussion and Conclusions

Our results confirms that the way gestures are used as a complementary non-verbal communication side-channel follows the principle of entropy rate constancy (ERC) in Information Theory. It means that the information encoded in hand gestures, albeit subtle, is actually organized in a *rational* way that enhances the decoding/understanding of information from a receiver's perspective. The main contribution is that we extend the scope of ERC to realm of non-verbal communication.

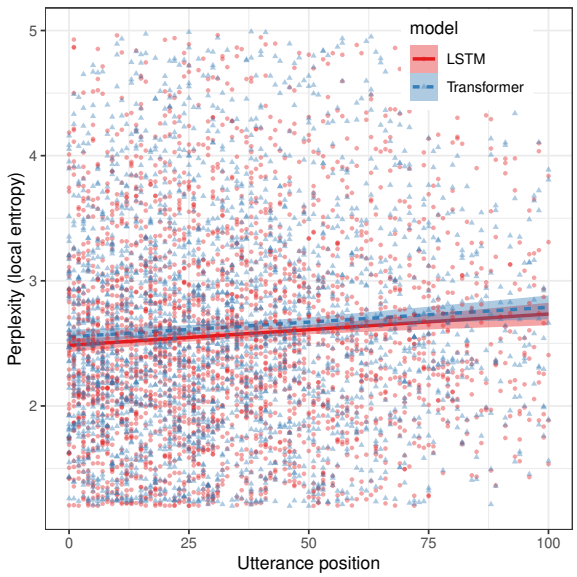

(a) Raw gesture

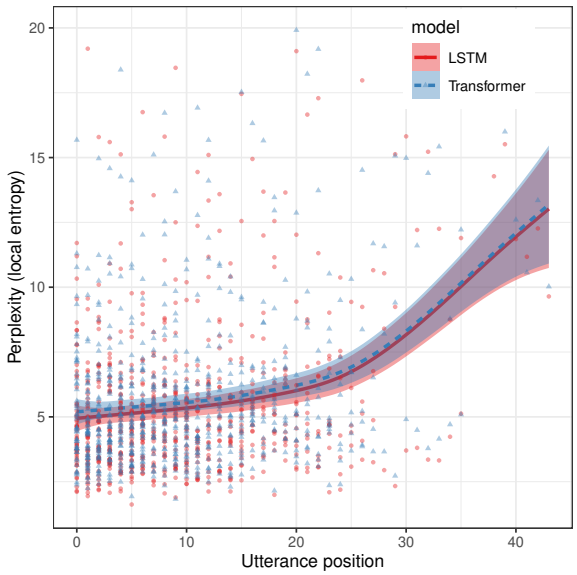

(b) Compressed gesture

Figure 4: Local entropy of gesture sequences increases with utterance position. 95% CIs are shown.

There are two explanations for what causes the observed entropy increasing pattern: First, more rare gestures (higher entropy) near the later stage of communication; Second, the entropy for the same gesture also increases during the communication. While the latter indicates a more sophisticated and interesting theory about gesture usage, both explanations requires further investigation.

While the motivation of this study is theoretical, but we believe the idea of extracting discrete gesture labels from spontaneous monologue/dialogue also has potentials in application. For instance, into better analysis of speaker intensions, sentiments,

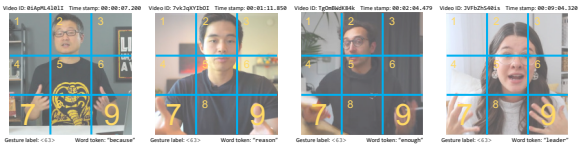

(a) Gesture label <63>

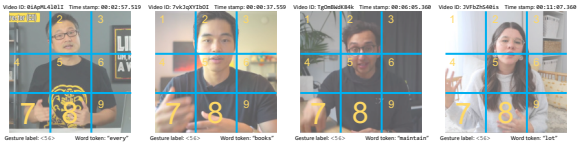

(b) Gesture label <56>

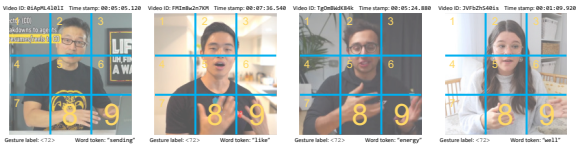

(c) Gesture label <72>

Figure 5: Typical screenshots for the top frequent gesture labels <63>, <56> and <72>.

and other implicit messages. For future work, we plan to use a larger dataset with a higher variety in genres (public speech, etc.) and examine more advanced representation method. such as continuous embedding and clustering. It is also interesting to interpret the semantic meanings of gestures and other non-verbal features by examining their semantic distance from words/utterances in vector space. More specifically, non-parametric clustering algorithms can be used to identify distinct the actions or poses of a person, which provides a way to extract more general gesture/action/pose labels for training.

Acknowledgements

This work is supported by National Science Foundation (CRII-HCC: 2105192). We thank Chen Song for providing valuable ideas on modeling.

References

Valentin Bazarevsky, Ivan Grishchenko, Karthik Raveendran, Tyler Zhu, Fan Zhang, and Matthias Grundmann. 2020. Blazepose: On-device real-time body pose tracking. *arXiv preprint arXiv:2006.10204*.

Gedas Bertasius, Heng Wang, and Lorenzo Torresani. 2021. Is space-time attention all you need for video understanding? *arXiv preprint arXiv:2102.05095*.

Sharice Clough and Melissa C Duff. 2020. The role of gesture in communication and cognition: Implications for understanding and treating neurogenic communication disorders. *Frontiers in Human Neuroscience*, page 323.

Lifeng Fan, Yixin Chen, Ping Wei, Wenguan Wang, and Song-Chun Zhu. 2018. Inferring shared attention in social scene videos. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 6460–6468.

Lifeng Fan, Shuwen Qiu, Zilong Zheng, Tao Gao, Song-Chun Zhu, and Yixin Zhu. 2021. Learning triadic belief dynamics in nonverbal communication from videos. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 7312–7321.

Kirill Gavrilyuk, Amir Ghodrati, Zhenyang Li, and Cees GM Snoek. 2018. Actor and action video segmentation from a sentence. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 5958–5966.

Dmitriy Genzel and Eugene Charniak. 2002. Entropy rate constancy in text. In *Proceedings of the 40th Annual Meeting on Association for Computational Linguistics*, pages 199–206, Philadelphia, PA.

Dmitriy Genzel and Eugene Charniak. 2003. Variation of entropy and parse trees of sentences as a function of the sentence number. In *Proceedings of the 2003 Conference on Empirical Methods in Natural Language Processing*, pages 65–72, Sapporo, Japan.

Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long short-term memory. *Neural computation*, 9(8):1735–1780.

Robert M Krauss, Yihsiu Chen, and Purnima Chawla. 1996. Nonverbal behavior and nonverbal communication: What do conversational hand gestures tell us? In *Advances in experimental social psychology*, volume 28, pages 389–450. Elsevier.

Linjie Li, Yen-Chun Chen, Yu Cheng, Zhe Gan, Licheng Yu, and Jingjing Liu. 2020. Hero: Hierarchical encoder for video+ language omni-representation pre-training. *arXiv preprint arXiv:2005.00200*.

David McNeill. 1992. Hand and mind1. *Advances in Visual Semiotics*, page 351.

Greg Mori, Caroline Pantofaru, Nisarg Kothari, Thomas Leung, George Toderici, Alexander Toshev, and Weilong Yang. 2015. Pose embeddings: A deep architecture for learning to match human poses. *arXiv preprint arXiv:1507.00302*.

Barbara Pease and Allan Pease. 2008. *The definitive book of body language: The hidden meaning behind people's gestures and expressions*. Bantam.

Steven T Piantadosi. 2014. Zipf's word frequency law in natural language: A critical review and future directions. *Psychonomic Bulletin and Review*, 21(5):1112–1130.

Claude Elwood Shannon. 1948. A mathematical theory of communication. *The Bell System Technical Journal*, 27:379–423.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. *Advances in neural information processing systems*, 30.

Fei Xu, Kenny Davila, Srirangaraj Setlur, and Venu Govindaraju. 2019. Content extraction from lecture video via speaker action classification based on pose information. In *2019 International Conference on Document Analysis and Recognition (ICDAR)*, pages 1047–1054. IEEE.

Yang Xu and David Reitter. 2016. Entropy converges between dialogue participants: explanations from an information-theoretic perspective. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics*, pages 537–546, Berlin, Germany.

Yang Xu and David Reitter. 2017. Spectral analysis of information density in dialogue predicts collaborative task performance. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 623–633, Vancouver, Canada. Association for Computational Linguistics.

Yang Xu and David Reitter. 2018. Information density converges in dialogue: Towards an information-theoretic model. *Cognition*, 170:147–163.

George Kingsley Zipf. 2013. *The Psycho-Biology of Language: An Introduction to Dynamic Philology*. Routledge.

A Appendix

A.1 Hyper-parameters and training procedures

For the LSTM-based encoder, embedding size is 300, hidden size is 200, number of layers is 2; a fully connected layer is used as the decoder connecting the encoder output and the softmax; dropout layers of probability 0.2 are applied to the outputs of both the encoder and decoder. For the Transformer-based encoder, model size is 20, hidden size is 100, number of layers is 2; same fully connected linear decoder is used; dropout layers of probability 0.5 are used at the position encoding, and each transformer encoder layer. To enable the one-direction (left to right) modeling effect, a mask matrix (of 0 and 1s) in an upper-triangular shape is used together with each input sequence.

Model parameters are randomly initialized. Training is done within 40 epochs, with batch size of 20, at and initial learning rate $lr = 0.05$. SGD optimizer with default momentum is used for training the LSTM model; Adam optimizer is used for training the Transformer model. Data are split to 80% for training and 20% for testing. After each training epoch, evaluation is done over the test set, and the model with lowest perplexity scores is saved as the best one.

Models are implemented with PyTorch. `torch.nn.CrossEntropyLoss` module is used as the loss function. The mathematical meaning of the output from this function is the negative logarithm likelihood (*NLL* in eq. (2)), and thus we compute the exponential values of the output to get the local entropy scores. The entropy scores used in the plot and statistical analysis are obtained from both train and test sets.

Revisiting Statistical Laws of Semantic Shift in Romance Cognates

Yoshifumi Kawasaki
The University of Tokyo
ykawasaki@g.ecc.u-tokyo.ac.jp

Maëlys Salingre
Mejiro University
m.salingre@mejiro.ac.jp

Marzena Karpinska
University of Massachusetts Amherst
mkarpinska@cs.umass.edu

Hiroya Takamura
AIST
takamura.hiroya@aist.go.jp

Ryo Nagata
Konan University
nagata-coling2022@ml.hyogo-u.ac.jp

Abstract

This article revisits statistical relationships across Romance cognates between lexical semantic shift and six intra-linguistic variables, such as frequency and polysemy. Cognates are words that are derived from a common etymon, in this case, a Latin ancestor. Despite their shared etymology, some cognate pairs have experienced semantic shift. The degree of semantic shift is quantified using cosine distance between the cognates' corresponding word embeddings. In the previous literature, frequency and polysemy have been reported to be correlated with semantic shift; however, the understanding of their effects needs revision because of various methodological defects. In the present study, we perform regression analysis under improved experimental conditions, and demonstrate a genuine negative effect of frequency and positive effect of polysemy on semantic shift. Furthermore, we reveal that morphologically complex etyma are more resistant to semantic shift and that the cognates that have been in use over a longer timespan are prone to greater shift in meaning. These findings add to our understanding of the historical process of semantic change.

1 Introduction

The Romance languages, such as present-day French, Italian, and Spanish, are sister languages which evolved from Vulgar Latin (Alkire and Rosen, 2010). They share numerous cognates, that is, words derived from a common etymon: for instance, the Latin verb HABERE "to have" developed into *avoir* (fr), *avere* (it), and *haber* (es).[1] Whereas French and Italian still maintain the original meaning "to have", it is no longer used as such in Spanish, where it is primarily used as an auxiliary to form perfect tense, while the notion of possession is generally expressed with *tener*. Likewise, the meanings and functions of some cognate pairs have diverged despite their common etymology. In the current paper, we investigate factors at play in semantic shift[2] using a computational approach.

In recent years, the analysis of diachronic semantic change using computational methods, inter alia, distributed representation of words, has gained increasing research interest (Dubossarsky et al., 2016; Hamilton et al., 2016; Takamura et al., 2017; Kutuzov et al., 2018; Uban et al., 2019; Hengchen et al., 2021; Kutuzov et al., 2021; Montariol and Allauzen, 2021; Schlechtweg et al., 2021; Tahmasebi et al., 2021; Uban et al., 2021a). The degree of lexical semantic shift has been conventionally quantified using cosine distance between relevant words, represented in the form of embeddings. A number of researchers have investigated the relationships between semantic change and frequency, polysemy, or prototypicality (Dubossarsky et al., 2015; Hamilton et al., 2016; Uban et al., 2019, 2021a). However, these effects have since been called into question (Dubossarsky et al., 2017).

Following this line of research, the present study revisits statistical relationships across Romance cognates between semantic shift and six intra-linguistic variables, including frequency and polysemy. For this purpose, we apply regression analysis and compare partial regression coefficients of individual variables while controlling for the others. By employing cognates, we are able to investigate whether particular trends hold across all explored languages. For the sake of simplicity, we assume

[1] Hereinafter, "fr" stands for French, "it" for Italian, and "es" for Spanish. Latin words are put in small capitals.

[2] We refer to the resultant difference in meaning between cognates as *shift*, while the term *change* is reserved for transition of a word's sense to another as well as for more general usage.

Proceedings of the 29th International Conference on Computational Linguistics, pages 141–151
October 12–17, 2022.

that their common ancestor, Latin, was uniform at the primitive stage before it developed into the Romance languages, although in reality Latin was not immune to variation across time, geography, or social stratification (Adams, 2007, 2013).

Our principal contributions are three: (i) amending flaws in the past research, we demonstrate that the law of conformity and the law of innovation (Hamilton et al., 2016) both hold for Romance cognates; (ii) exploring hitherto unexplored attributes of Latin etyma, we reveal that cognates whose etyma are morphologically more complex are more resistant to semantic shift; and (iii) considering temporal gap between words in lexical incorporation to a language, we show that words that have been in use over a longer timespan are prone to diverge more in meaning.

The rest of the paper is organized as follows. In Section 2, we review related research and point out its shortcomings. Section 3 describes our improved methodology. In Section 4 we present the experimental setup and results, followed by discussion in Section 5. Section 6 concludes the paper, pointing to future research directions.

2 Related Work

Two studies deserve special mention; both of them tackled statistical analysis of diachronic semantic change using word embeddings. A milestone in the field was Hamilton et al. (2016), who analyzed historical corpora in several languages and proposed two laws concerning semantic change: one is the *law of conformity*, according to which frequency is negatively correlated with semantic change; the other is the *law of innovation*, stating that polysemy is positively correlated with semantic change. However, these laws, as well as that of prototypicality (Dubossarsky et al., 2015), were revised (Dubossarsky et al., 2017) on the grounds that (i) the effect of frequency turned up even under control conditions, where no correlation was to be expected; and (ii) polysemy and prototypicality as they were defined by that research were highly collinear with frequency.

Inquiry into Romance cognates was first pursued by Uban et al. (2019) and later extended in Uban et al. (2021a). In contrast to Hamilton et al. (2016), Uban et al. (2019, 2021a) witnessed a positive correlation between frequency and degree of semantic shift; words that underwent more semantic shift tend to be more frequent. However, the experiments were conducted under non-comparable setups. Unlike Hamilton et al. (2016), who used cosine distance, Uban et al. (2019, 2021a) quantified the magnitude of semantic shift with what they denominated "falseness score." This value was calculated by subtracting the similarity score for a non-cognate translation pair (e.g., *long* "long" (fr) – *largo* "id." (es)) from the one for a cognate pair judged as false friends (e.g., *long* (fr) – *luengo* "id. (erudite wording)" (es), both tracing back to LONGU)[3]. *False friend* refers to either of two cognates in different languages that have diverged semantically despite their common etymology (Penny, 2002). Focusing on false friend pairs inevitably translates into exclusion of the cognate pairs that preserve commonality in meaning; this would prevent us from gaining insights on general tendencies across overall cognate pairs. In addition, the use of falseness scores was hardly justified. Accordingly, it is imperative to settle the dispute over the diametric effect of frequency by implementing experiments under identical conditions, such that genuine cognate pairs as well as false friend pairs are covered.

Furthermore, there is still room for improvement. First, Uban et al. (2021a) defined polysemy of a Romance word as the number of synsets in WordNet that the word is part of. Thus, the polysemy score of a cognate pair was computed as the average of number of synsets for the two words in question. However, if the ultimate goal is to determine factors that could induce semantic shift, it is more appropriate to consider polysemy prior to eventual semantic shift. Second, Uban et al. (2021a) leveraged publicly available multilingual word embeddings (Conneau et al., 2018). Notwithstanding their usefulness, however, there is a risk that the embeddings are undesirably affected by the alignment algorithm employed; if a cognate pair is aligned in advance, then it naturally becomes more similar than it should be.

3 Methods

3.1 Overview

To circumvent the series of methodological defects adduced above, we took the following measures. (i) In the same spirit as Hamilton et al. (2016), we examined all the cognate pairs and measured degree of semantic shift with cosine distance score.

[3] *Id.* (idem) represents having parallel sense to its counterpart.

Then we applied linear regression analysis to explore statistical relationships between the degree of semantic shift and six intra-linguistic variables specified in Section 4.2. (ii) Independently from frequency counts, we defined polysemy as the number of word sense entries of Latin etyma in a dictionary. The use of Latin polysemy has the added advantage of yielding a polysemy score for the earliest stage of evolution, prior to semantic shift. (iii) Before undertaking our main analysis, we ensured the absence of a priori correlation between frequency and semantic affinity for random word pairs, which are presumed to present low similarity. (iv) We created our own cognates list from scratch to procure fit-for-purpose vector representation of words, as described in the next section[4].

3.2 Construction of multilingual embeddings

We limited the scope of study to three Romance languages: French, Italian, and Spanish. We used a Wikipedia dump as of December 2018 to acquire static word embeddings separately in each language. The relative uniformity of Wikipedia in style and topic is a suitable property to ensure, as much as possible, that any difference stemming from comparison of cognate embeddings is imputable to the nature thereof instead of the corpus on which training was performed. Data cleaning was carried out using `Wiki-cirrus`[5] for French and `Wikiextractor`[6] for Italian and Spanish. Then we performed lemmatization with `TreeTagger`[7]. No distinction was made between homonyms, due to their paucity.

Target words for analysis were restricted to nouns, adjectives, and verbs. To prevent low frequency from disturbing the reliability of the obtained embeddings, we opted to focus on highly frequent cognates. Specifically, a separate cognates list was created for each word class, such that at least one member of a cognate set was found among the most frequent 300 lemmas in any of the three languages; it was possible for a cognate to rank among the top 300 in one language while its counterpart did not in another. The number of cognate pairs added up to 487 for nouns, 477 for adjectives, and 493 for verbs, although the cognates were not always found in all three languages. Cognate identification was implemented by looking into etymological description in dictionaries[8].

Unilingual embeddings were learned by `word2vec` in the `gensim` library (Řehůřek and Sojka, 2010)[9]. The default hyperparameters were adopted, with the exception of `vector_size=600` and `min_count=50`. To compare word embeddings in various languages, we need to align the separately obtained embeddings in a common cross-lingual vector space. The alignment was executed using the supervised method of MUSE (Conneau et al., 2018)[10], that is, the linear mapping proposed by Mikolov et al. (2013a) with the orthogonality constraint, such that the sum of the squared errors across the inter-lingual seed pairs was as low as possible. The seed pairs with an acknowledged parallel meaning between the two languages were retrieved from MUSE bilingual dictionaries (Conneau et al., 2018)[11]; the cognate pairs that are targets of our analysis were eliminated therefrom. In any combination of languages, seed pairs amounted to about ten thousand. The choice of the language onto which the embeddings of other languages were mapped had marginal effect on the subsequent analysis. In the following, we report results obtained from French–Spanish cognate pairs aligned in French vector space for illustrative purposes. See Section 5.3 for discussion regarding different embedding spaces and language combinations.

3.3 Validation of embeddings

Following the commonplace procedure, the pairwise similarity between cognates was defined as the cosine similarity score between the corresponding embeddings. Based on the distributional hypothesis (Harris, 1954; Firth, 1957), word embeddings are claimed to capture lexical meaning to a certain degree. Before going any further, it is pivotal to assess the quality of the acquired embeddings to ensure the soundness of our principal analysis. To this end, we conducted the following two experiments.

[4] The data are available upon request.

[5] https://www.mediawiki.org/wiki/Help:CirrusSearch

[6] https://github.com/attardi/wikiextractor

[7] https://www.cis.lmu.de/~schmid/tools/TreeTagger/

[8] French: https://www.cnrtl.fr
Italian: https://www.etimo.it
Spanish: https://dle.rae.es

[9] https://radimrehurek.com/gensim/models/word2vec.html

[10] https://github.com/facebookresearch/MUSE

[11] https://github.com/facebookresearch/MUSE

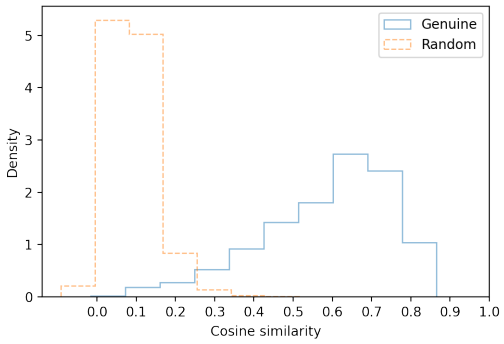

Figure 1: Density distribution of pair-wise similarity scores for ca. 4K genuine seed pairs and 1000K random word pairs.

First, we analyzed the pair-wise similarity distribution of the seed pairs that were employed when aligning word embeddings; it is essential that they present high similarity scores. For this, we chose seed pairs each word of which appears more than ten thousand times in the corpus of the relevant language. This threshold was set for fear that the embeddings of infrequent words might be under-learned. The number of seed pairs selected amounted to nearly four thousand for any language pair. The histogram with solid contour line in Figure 1 illustrates the density distribution of similarity scores for the frequent seed pairs. As expected, they show relatively high similarity, with mean of 0.66 and standard deviation of 0.15.

Next, we repeated the operation with 100,000 randomly generated pairs from among the seed words. These random pairs ought to show minimal similarity scores. The histogram with dotted contour line in Figure 1 illustrates the density distribution. Unsurprisingly, the mean similarity score that resulted was low at 0.10, with standard deviation of 0.06. It is worth noting that a similarity score of 0.1 can arise in our case, even when given two unrelated words.

Inasmuch as the validity of the obtained embeddings is confirmed, we can safely make use of them on the premise that cosine similarity reflects semantic affinities reasonably, if not completely. Table 1 provides examples of the five most and five least similar French–Spanish cognate pairs. A smaller similarity score means a greater degree of semantic shift, and vice versa.

To make an overall evaluation of the computed affinity scores, we inspected the proportion of the cognate pairs that are enumerated in MUSE as

French	Spanish	Sim.
construire “to construct”	constuir “id.”	0.87
provoquer “to provoke”	provocar “id.”	0.87
évêque “bishop”	obispo “id.”	0.86
détruire “to destroy”	destruir “id.”	0.86
féminin “feminine”	femenino “id.”	0.86
avoyer “to set saw”	aviar “to prepare”	0.05
atteindre “to reach”	atañer “to pertain”	0.05
mener “to take”	menar “to turn”	0.04
saison “season”	sazón “seasoning”	0.02
maire “mayor”	mayor “bigger, older”	0.00

Table 1: Most (upper half) and least (lower half) similar five French–Spanish cognate pairs.

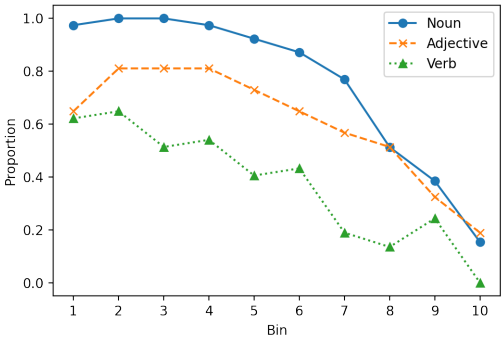

Figure 2: Proportion of cognate pairs enumerated in MUSE as translation pairs in ten equally divided bins for respective word classes. The leftmost bin corresponds to the top 10% translation pairs that are most similar, the second bin to the following 10%, and so on in the same way.

translation pairs across ten equally divided bins for respective parts-of-speech. The result is shown in Figure 2. The leftmost bin corresponds to the top 10% translation pairs that are most similar, and the second bin to the following 10%, continuing in the same way. We can observe that, moving rightwards, the drop in proportion goes roughly in line with decreasing similarity across the word classes; the most similar cognate pairs are mostly among the inter-lingual translation pairs, whereas the dissimilar pairs, for the most part, are outside of them and thus regarded as false friend pairs.

Moreover, we notice that the proportion varies across the parts-of-speech. Specifically, the verbs consistently present the lowest proportion, which is indicative of this word class being more inclined to shift, followed by adjectives and nouns. This tendency partially conforms to the conclusion arrived at by Dubossarsky et al. (2016), except that they discovered an inverse order between nouns and adjectives.

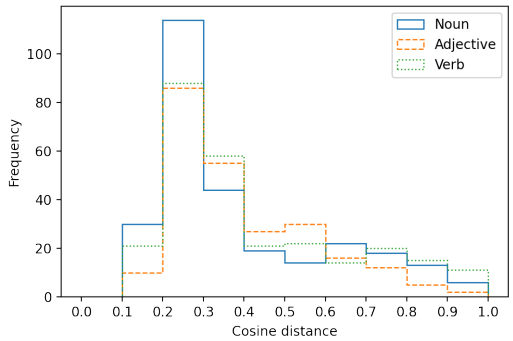

Figure 3: Distribution of cosine distance scores for French–Spanish cognate pairs in different word classes.

4 Regression Analysis

We applied linear regression analysis to explore statistical relationships between the degree of semantic shift and six intra-linguistic factors, detailed below. For the pair-wise scatter plots between the standardized variables, see Appendix A.

4.1 Dependent variable

The dependent variable DIST_{rom} is the log-transformed pair-wise cosine distance score between cognates, defined as $\log(dist_{cos}(cog1, cog2)) = \log(1 - \cos(cog1, cog2))$, where $\cos(cog1, cog2)$ stands for cosine similarity of a cognate pair $(cog1, cog2)$. This variable represents the magnitude of semantic shift that the cognate pair in question underwent. Figure 3 illustrates the distribution of cosine distance scores for French–Spanish cognate pairs in different word classes. The distribution is apparently skewed to the right in all classes, with mean of 0.39 and median of 0.29 for nouns, 0.40 and 0.34 for adjectives, and 0.42 and 0.33 for verbs. A rather long right tail implies a large number of dissimilar pairs.

4.2 Independent variables

We established six intra-linguistic factors as independent variables. The first three relate to Latin etyma and the remaining three to Romance descendants. Despite their potential contribution to semantic change, extra-linguistic variables such as socio-cultural, historical, political, technological factors as well as language contact were outside our scope (Penny, 2002; Newman, 2015; Hamilton et al., 2016; Dubossarsky et al., 2017).

4.2.1 Latin features

We leveraged three features concerning Latin etyma: frequency, polysemy, and word length. These are variables that relate to the phase prior to semantic shift and have been ignored in past studies (Uban et al., 2019, 2021a). For nouns and adjectives, we deemed the Latin accusative to be the etymological form, for it is the case from which Romance descendants are typically derived (Alkire and Rosen, 2010). As is conventionally done, word-final -M was omitted if applicable (e.g., we adopted ANNU for ANNUM "year"). For verbs, the infinitive was used for convenience, as the representative form for Latin as well as for the Romance languages. With respect to the infinitival form of Latin deponent verbs (Oniga, 2014), we employed reconstructed active forms of the corresponding conjugation class (e.g., JOCARI "to joke" was transformed into *JOCARE). As for irregular verbs including ESSE "to be", FERRE "to carry", and VELLE "to want", we used original infinitival forms.

Frequency of Latin etymon (FREQ_{lat}) This is defined as the log-transformed relative frequency of the Classical Latin etymon per 10,000 words retrieved from the online database on classical languages PhiloLogic4[12]. Even though the Romance languages did not evolve from Classical Latin (written language), but Vulgar Latin (spoken language) (Alkire and Rosen, 2010), since there was no resource available for the latter, we utilized the materials at hand on the literary language. The same applies to other Latin features below.

Latin polysemy (POLY_{lat}) This is defined as the log-transformed number of word sense entries for the Latin etymon in the *Oxford Latin Dictionary* (Glare, 2012). Note, however, that taking logarithm did not remedy the right-skewed distribution.

Word length of Latin etymon (LEN_{lat}) This is defined as the number of characters in the Latin etymon. Longer words tend to be morphologically more complex due to affixation or derivation, which in turn helps restrict the semantic range of the base form. For example, PRAEVIDERE "to foresee", which is constructed from a base VIDERE "to see"

[12] http://artflsrv02.uchicago.edu/cgi-bin/perseus/LatinFrequency.pl?author=&title=&genre=&editor=&language=NOT+English&displaymorethan=10&displaylessthan=10000000&sortby=decreasingFreq&searchby=searchbylemma (as of January 28th, 2021).

and a prefix PRAE- "ahead", exhibits a more restricted sense than its base. In fact, the similarity score between *prévoir* (fr) and *prever* (es), derived from PRAEVIDERE, is as high as 0.72, while it is 0.35 between *voir* (fr) and *ver* (es), derived from VIDERE.

4.2.2 Romance features

The following four variables relate to the phase by which semantic shift occurred in relevant cases.

Mean frequency of Romance cognate pair (FREQ$_{rom}$) Defined as the log-transformed harmonic mean $\log \mathrm{Harmonic}(freq_{cog1}, freq_{cog2})$, where $freq_w$ is the relative frequency of a word w in the corresponding corpus, and $cog1$ and $cog2$ are cognate words. We found it more opportune to use the harmonic mean than the arithmetic mean in handling average ratio, thanks to its property of being biased toward the smaller value. The merit of using the harmonic mean was empirically verified in the following regression analysis.

To ensure that there is no a priori correlation between semantic shift and pair-wise mean frequency, we examined 100,000 randomly generated word pairs from among the seed words used in aligning embeddings. As expected, only a negligible correlation was seen, of -0.06, which signifies that our analysis is practically free from the spurious effect of frequency (Dubossarsky et al., 2017).

Mean embeddings' norm in Romance cognate pairs (NORM$_{rom}$) This is defined as the arithmetic mean over the embeddings' norm of the words that compose a cognate pair. We took into account the norm, which is a hitherto ignored attribute of embeddings, with the aim of assessing if its effect is detected independently from frequency and polysemy; the algorithm underlying the skip-gram model with negative sampling (Mikolov et al., 2013b) entails that the norm of a word embedding grows large when its frequency is high and is oligosemous, being used consistently in analogous contexts.

Mean edit distance between Latin etymon and Romance cognate pair (EDIT) This is defined as the arithmetic mean over the normalized edit distances between Latin etymon l and each word forming a cognate pair (r, r'), as follows:

$$\frac{1}{2}\left(\frac{edit(r,l)}{\frac{|r|+|l|}{2}} + \frac{edit(r',l)}{\frac{|r'|+|l|}{2}}\right), \qquad (1)$$

where $edit(\cdot)$ is the unnormalized edit distance, which is normalized with division by mean word length between l and r (Levenshtein, 1966), and $|\cdot|$ represents word length. Although it is desirable to make phonetic comparisons, we instead quantified graphical displacement for the sake of simplicity.

We argue that edit distance can be viewed as a proxy for how long the words have been used in a language or when they came to form the language's lexica; words with large edit distances with respect to Latin etymon are regarded as *inherited words*, which underwent typical phonetic alterations through oral transmission in the respective descendant languages and thus ended up having a distinct appearance from their ancestor (Penny, 2002). Conversely, the words with small edit distances correspond to *learned words*, which were relatively recently borrowed into the Romance languages with minimal phonetic alterations from the Medieval Latin (Penny, 2002). A dichotomous distinction between inheritance and recent borrowing was introduced by Uban et al. (2021b), who made the distinction by consulting dictionaries. Our novelty consists in quantifying the difference in an automatic manner using edit distance.

Prior to computing edit distance, all the diacritics were removed from the vowels. For example, *á* and *à* were both transformed into *a*. For sake of simplicity, the consonants were left intact despite the fact that some characters, such as Spanish *ñ* and French *ç*, did not exist in Latin, and that phonetic values of some letters are not identical between Latin and the Romance languages: for instance, CITARE [k] "to set in motion" in contrast to *citar* (es) [θ] "to cite".

Table 2 presents five French–Spanish cognate pairs with the largest (upper half) and the smallest (lower half) edit distance scores. We readily notice the aforesaid tendency; the cognate pairs with the largest edit distances correspond to inherited words with various modifications, while those with the smallest edit distances belong to learned vocabulary that virtually maintains its etymological form.

4.3 Setup

The dataset comprised those cognate pairs that bear Romance forms in the two languages of interest and for which information on frequency and polysemy of the Latin etymon was available; we ignored cognates whose etyma do not trace back to

Latin	French	Spanish	Dist.
CAPUT "head"	chef	jefe	1.00
VICE "time"	fois	vez	0.93
AURU "gold"	or	oro	0.93
RUSSU "red"	roux	rojo	0.89
EPISCOPU "bishop"	évêque	obispo	0.86
BASE "base"	base	base	0.00
ENORME "enormous"	énorme	enorme	0.00
SERIE "series"	série	serie	0.00
CELEBRE "busy"	célèbre	célebre	0.00
ANIMAL "animal"	animal	animal	0.00

Table 2: Five French–Spanish cognate pairs with largest (upper half) and smallest (lower half) edit distances.

Latin. This constraint almost halved the number of effective cognate pairs, down to 281 for nouns, 243 for adjectives, and 270 for verbs, hence, 794 in total.

Prior to performing regression analysis, all the variables were standardized to have zero mean and unit variance, allowing for directly comparing the scale of regression coefficients. Note that the log-transformed variables scale according to the power laws with the degree of semantic shift. We did not log-transform LEN_{lat}, NORM_{rom}, or EDIT, since these did not exhibit a right-skewed distribution; nor, at this point, did we distinguish among different word classes. See Section 5.2 for a survey discriminating them. In spite of a large correlation of 0.73 between FREQ_{lat} and POLY_{lat}, we did not exclude either one, for their variance inflation factors were less than three. We implemented model selection methods in terms of AIC values (Akaike, 1974).

4.4 Results

The best model chosen, which only dropped NORM_{rom}, is summarized in Table 3. All the retained covariates were statistically significant at 0.05 level except FREQ_{lat}[13]. Standardized partial regression coefficients turned negative for FREQ_{rom} (-0.54) and LEN_{lat} (-0.21), and positive for POLY_{lat} (0.10) and EDIT (0.13). The adjusted R-squared of 0.35 means that 35% of the total variance was accounted for by the model[14].

5 Discussion

5.1 Independent variables

Based on the outcome of the regression analysis, we discuss how individual explanatory variables correlate with the degree of semantic shift.

[13] All subsequent significance tests are at $p < 0.05$.

[14] Henceforth, adjusted R-squared is abbreviated $Adj.R^2$.

	Coef.	SE	t	$p > \|t\|$
Intercept	0.00	0.03	0.00	1.00
FREQ_{lat}	-0.08	0.04	-1.82	0.07
POLY_{lat}	**0.10**	**0.04**	**2.28**	**0.02**
LEN_{lat}	**-0.21**	**0.03**	**-6.29**	**0.00**
FREQ_{rom}	**-0.54**	**0.03**	**-18.40**	**0.00**
NORM_{rom}	–	–	–	–
EDIT	**0.13**	**0.03**	**4.07**	**0.00**

Table 3: Results of regression analysis on distance scores of French–Spanish cognate pairs ($N = 794$, $Adj.R^2 = 0.35$). NORM_{rom} was kept out by model selection methods.

FREQ_{lat} This predictor did not result statistically significant; its effect might have been absorbed by POLY_{lat} and LEN_{lat}, which it is collinear with.

POLY_{lat} A positive coefficient (0.10) denotes that the cognates with a more polysemous etymon tend to undergo a larger semantic shift. This finding agrees with the insights of Hamilton et al. (2016) and Uban et al. (2019, 2021a) that polysemy is positively correlated with semantic shift. Nonetheless, it should be highlighted that this finding hints at a causal effect, because we exploited the polysemy at the initial stage of linguistic development. That said, why polysemy leads to larger semantic shift requires explanation. We argue that, even when the rate of semantic change is constant per given time unit, polysemous words are more likely to digress into diverse directions in each language: for example, TRAHERE "to drag", which possessed 22 word sense entries, developed different senses in the daughter languages: *trarre* "to draw" (it), *traire* (fr) "to milk", and *traer* (es) "to bring". Thus, these cognates exhibit large distance scores, around 0.85.

LEN_{lat} A negative coefficient (-0.21) backs up our hypothesis that the cognates with a longer Latin etymon tend to undergo smaller semantic shift. We conjecture that longer words are less susceptible to meaning shift, because they tend to have restricted senses, which indirectly supports the law of innovation (Hamilton et al., 2016). Actually, there is a negative correlation of -0.34 between word length and polysemy in Latin (Appendix A).

FREQ_{rom} A negative coefficient (-0.54) sustains the alleged law of conformity (Hamilton et al., 2016) and, in turn, dismisses the opposite conclusion reached by Uban et al. (2019, 2021a). The largest absolute coefficient indicates its significant

contribution to the model's predictive power.

As to the negative correlation between frequency and semantic shift, we suppose that frequently used words are entrenched enough to resist semantic shift (Bybee, 2015). However, we cannot assert this with full confidence, because FREQ$_{rom}$ relates to a time point posterior to semantic shift, and so the observed frequency might well be the fruit of semantic shift.

It is also noteworthy that $Adj.R^2$ diminished considerably from 0.35 to 0.11 when using an arithmetic mean instead of a harmonic mean. The property of the harmonic mean of being biased toward the smaller value could be beneficial in cases where only one of the cognates has suffered semantic change, accompanied by considerable variation in frequency, and where consequently they became dissimilar.

NORM$_{rom}$ That the model kept out this predictor implies that its effect is incapable of being detected independently from frequency and polysemy.

EDIT A positive coefficient (0.13) confirms our hypothesis that inherited words tend to go through larger semantic shift.

5.2 Effect of word class

When fitting the model separately for separate word classes, we found a slight variation in $Adj.R^2$: 0.38 for nouns, 0.31 for adjectives, and 0.49 for verbs. This unequal behavior might imply that nouns are more susceptible than verbs, and adjectives more than nouns, to extra-linguistic factors disregarded in this study: socio-cultural circumstances, technological advances, language contact, metaphorical extensions, to name a few. As to the best models selected, LEN$_{lat}$ and FREQ$_{rom}$ were retained and were statistically significant across the parts-of-speech, exhibiting comparable coefficients; therefore, the effect of these variables appear to be extensive. In contrast, the remaining variables displayed differing behaviors from one class to another.

5.3 Consequences of aligned embedding spaces and language combinations

Thus far, for illustrative purposes, we have solely focused on French–Spanish cognate pairs aligned in French embedding space. It is indispensable also to assess the potential consequences that different embedding spaces and language combinations could bring about. There are nine settings in total; three language pairs for each of three embedding spaces. A close examination reveals that (i) different embedding spaces have practically null effect on the outcome of regression analysis; and (ii) different language pairings, in turn, slightly affect the composition of the best model, the scale of regression coefficients, and accordingly $Adj.R^2$.

Embedding space	Language pair	$Adj.R^2$	N
French	French–Italian	0.29	812
	French–Spanish	0.35	794
	Italian–Spanish	0.35	842
Italian	French–Italian	0.29	812
	French–Spanish	0.33	794
	Italian–Spanish	0.38	842
Spanish	French–Italian	0.27	812
	French–Spanish	0.35	794
	Italian–Spanish	0.39	842

Table 4: Adjusted R-squared for respective language pairs in different embedding spaces.

Table 4 is the summary of $Adj.R^2$ for respective language pairs in different embedding spaces in which the alignment was done. It is noteworthy that, in every embedding space, the score is consistently largest for the Italian–Spanish pair, followed by French–Spanish and French–Italian pairs. This suggests that some unconsidered variables are more at play for the French–Italian pair than for the French–Spanish pair, and more for French–Spanish pair than for Italian–Spanish pair.

Figure 4 depicts radar charts presenting regression coefficients of the best-fitted model for respective language pairs. The embeddings were aligned in the vector space that the legend indicates. A marker at the origin represents rejection of the variable in question. The dashed circle represents coefficients being equal to zero. The intercept was omitted for having a value almost equal to zero in every case. Regression analysis was run without parts-of-speech distinction. The almost overlapping lines demonstrate that the general trends commented above hold across the different settings, although not without exceptions: (i) NORM$_{rom}$ was retained for Italian–Spanish pairs in any embedding space, while FREQ$_{lat}$ was dropped. Considered in conjunction, the effects of FREQ$_{lat}$ might have been offset by that of NORM$_{rom}$, thereby exhibiting a negative correlation with semantic shift ; and (ii) EDIT was accidentally dropped in French–Spanish combinations in Italian embedding space for unknown reasons.

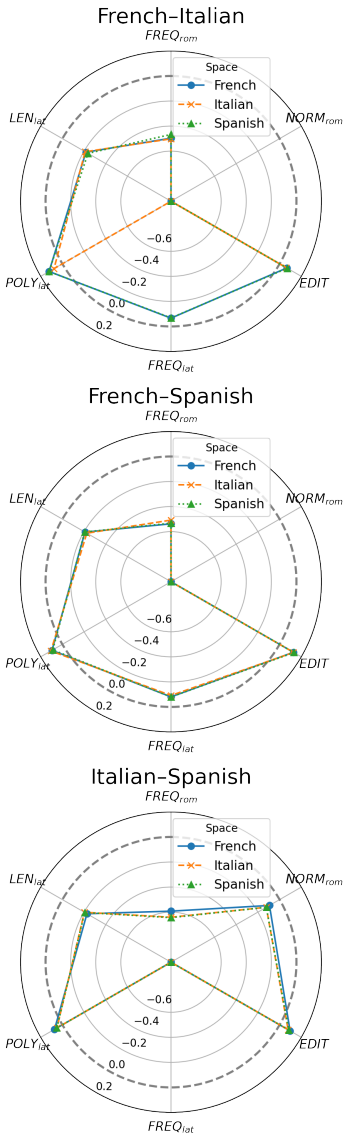

Figure 4: Regression coefficients of the best-fitted model for respective language pairs aligned in different embedding spaces.

5.4 A posteriori prediction at Latin era

It is an intriguing question how well one can make prediction on semantic shift. With this objective, we performed an additional regression analysis exploiting only Latin features. Model selection was performed with AIC. As Table 5 shows, FREQ_{lat} and LEN_{lat} were retained and became statistically significant. The fitted coefficients imply that frequent and long Latin etyma tend to undergo less semantic shift, which directly underpins the law of conformity and indirectly the law of innovation (Hamilton et al., 2016), if it is appropriate to associate word length with polysemy. Although barely 6% of the variance was explained, we found this phenomenon in accordance with the reflection that "small effects may be a priori more credible than large ones" (Dubossarsky et al., 2017).

	Coef.	SE	t	$p > \|t\|$
Intercept	0.00	0.03	0.00	1.00
FREQ_{lat}	**-0.10**	**0.04**	**-2.74**	**0.01**
POLY_{lat}	–	–	–	–
LEN_{lat}	**-0.27**	**0.04**	**-7.10**	**0.00**

Table 5: Results of regression analysis on distance scores of French–Spanish cognate pairs ($N = 794$, $Adj.R^2 = 0.06$). Only Latin features were employed. POLY_{lat} was kept out by model selection methods.

6 Conclusions

In this study, we revisited statistical relationships between semantic shift and intra-linguistic variables across the Romance languages. Our principal contributions are three: (i) we demonstrated that the law of conformity and the law of innovation (Hamilton et al., 2016) both hold for Romance cognates by amending flaws in the past research; (ii) we revealed that cognates whose etyma are morphologically more complex are more resistant to semantic shift by exploring hitherto unexplored attributes of Latin etyma; and (iii) we showed that words that have been in use over a longer timespan are prone to diverge more in meaning by considering temporal gap in lexical incorporation to a language.

One limitation of our study is a lack of comparison with Latin. Without it, we would not have a complete picture of historical semantic change; we need to understand how much and in what direction it has evolved. To meet this aim, it is a sine qua non to obtain Latin embeddings trained upon Classical Latin sources, thereby allowing for direct comparisons with the Romance ones. Also, it will be an interesting extension to elucidate types of semantic change that have occurred in relevant cases (Kutuzov et al., 2018), such as specialization, generalization, melioration, and pejoration (Traugott and Dasher, 2005). In addition, we need to address qualitative difference in meaning, including typicality, hypernymy, and hyponymy, since we did not go further in this paper than to define polysemy as the number of word sense entries in the dictionary. Such an improvement could be achieved by taking advantage of information in WordNet.

Acknowledgements

This work was supported by JSPS KAKENHI Grant Number JP18K11456.

References

James Noel Adams. 2007. *The Regional Diversification of Latin 200 BC-AD 600*. Cambridge University Press, New York.

James Noel Adams. 2013. *Social Variation and the Latin Language*. Cambridge University Press, New York.

Hirotugu Akaike. 1974. A New Look at the Statistical Model Identification. *IEEE Transactions on Automatic Control*, 19(6):716–723.

Ti Alkire and Carol Rosen. 2010. *Romance Languages: A Historical Introduction*. Cambridge University Press, New York.

Joan Bybee. 2015. *Language Change*. Cambridge University Press, Cambridge.

Alexis Conneau, Guillaume Lample, Marc'Aurelio Ranzato, Ludovic Denoyer, and Hervé Jégou. 2018. Word Translation Without Parallel Data.

Haim Dubossarsky, Yulia Tsvetkov, Chris Dyer, and Eitan Grossman. 2015. A Bottom Up Approach to Category Mapping and Meaning Change. In *Word Structure and Word Usage*, pages 66–70, Pisa.

Haim Dubossarsky, Daphna Weinshall, and Eitan Grossman. 2016. Verbs Change More Than Nouns: A Bottom-up Computational Approach to Semantic Change. *Lingue e Linguaggio*, 15(1):5–25.

Haim Dubossarsky, Daphna Weinshall, and Eitan Grossman. 2017. Outta control: Laws of semantic change and inherent biases in word representation models. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 1136–1145, Copenhagen, Denmark. Association for Computational Linguistics.

J.R. Firth. 1957. A Synopsis of Linguistic Theory 1930-55. *Studies in Linguistic Analysis: Special Volume of the Philological Society*, pages 1–32.

P.G.W. Glare, editor. 2012. *Oxford Latin Dictionary*, 2nd edition. Oxford University Press, Oxford.

William L. Hamilton, Jure Leskovec, and Dan Jurafsky. 2016. Diachronic word embeddings reveal statistical laws of semantic change. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1489–1501, Berlin, Germany. Association for Computational Linguistics.

Zellig S. Harris. 1954. Distributional Structure. *WORD*, 10(2-3):146–162.

Simon Hengchen, Nina Tahmasebi, Dominik Schlechtweg, and Haim Dubossarsky. 2021. Challenges for Computational Lexical Semantic Change. In Nina Tahmasebi, Lars Borin, Adam Jatowot, Yang Xu, and Simon Hengchen, editors, *Computational Approaches to Semantic Change*, chapter 11, pages 341–372. Language Science Press, Berlin.

Andrey Kutuzov, Lilja Øvrelid, Terrence Szymanski, and Erik Velldal. 2018. Diachronic word embeddings and semantic shifts: a survey. In *Proceedings of the 27th International Conference on Computational Linguistics*, pages 1384–1397, Santa Fe, New Mexico, USA. Association for Computational Linguistics.

Andrey Kutuzov, Lidia Pivovarova, and Mario Giulianelli. 2021. Grammatical profiling for semantic change detection. In *Proceedings of the 25th Conference on Computational Natural Language Learning*, pages 423–434, Online. Association for Computational Linguistics.

Vladimir Iosifovich Levenshtein. 1966. Binary Codes Capable of Correcting Deletions, Insertions and Reversals. *Soviet Physics Doklady*, 10(8):707–710.

Tomás Mikolov, Quoc V. Le, and Ilya Sutskever. 2013a. Exploiting Similarities among Languages for Machine Translation. *CoRR abs/1309.4*.

Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013b. Distributed Representations of Words and Phrases and Their Compositionality. In *Proceedings of the 26th International Conference on Neural Information Processing Systems*, volume 2 of *NIPS'13*, pages 3111–3119, Red Hook, NY, USA. Curran Associates Inc.

Syrielle Montariol and Alexandre Allauzen. 2021. Measure and evaluation of semantic divergence across two languages. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 1247–1258, Online. Association for Computational Linguistics.

John Newman. 2015. *Semantic Shift*, 1st edition, pages 266–280. Routledge, New York.

Renato Oniga. 2014. *Latin: A Linguistic Introduction*. Oxford University Press, New York.

Ralph Penny. 2002. *A History of the Spanish Language*. Cambridge University Press, Cambridge.

Dominik Schlechtweg, Barbara McGillivray, Simon Hengchen, Haim Dubossarsky, and Nina Tahmasebi. 2021. SemEval-2020 Task 1: Unsupervised Lexical Semantic Change Detection. In *Proceedings of the 14th International Workshop on Semantic Evaluation*, pages 1–23, Barcelona (Online).

Nina Tahmasebi, Lars Borin, and Adam Jatowt. 2021. Survey of Computational Approaches to Lexical Semantic Change Detection. In Nina Tahmasebi,

Lars Borin, Adam Jatowot, Yang Xu, and Simon Hengchen, editors, *Computational Approaches to Semantic Change*, chapter 1, pages 1–91. Language Science Press, Berlin.

Hiroya Takamura, Ryo Nagata, and Yoshifumi Kawasaki. 2017. Analyzing semantic change in Japanese loanwords. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 1, Long Papers*, pages 1195–1204, Valencia, Spain. Association for Computational Linguistics.

Elizabeth Closs. Traugott and Richard B. Dasher. 2005. *Regularity in Semantic Change*. Cambridge University Press, Cambridge.

Ana Uban, Alina Maria Ciobanu, and Liviu P. Dinu. 2019. Studying laws of semantic divergence across languages using cognate sets. In *Proceedings of the 1st International Workshop on Computational Approaches to Historical Language Change*, pages 161–166, Florence, Italy. Association for Computational Linguistics.

Ana-Sabina Uban, Alina Maria Ciobanu, and Liviu P Dinu. 2021a. Cross-lingual Laws of Semantic Change. In Nina Tahmasebi, Lars Borin, Adam Jatowot, Yang Xu, and Simon Hengchen, editors, *Computational Approaches to Semantic Change*, chapter 7, pages 219–260. Language Science Press, Berlin.

Ana Sabina Uban, Alina Maria Cristea, Anca Dinu, Liviu P. Dinu, Simona Georgescu, and Laurentiu Zoicas. 2021b. Tracking semantic change in cognate sets for English and Romance languages. In *Proceedings of the 2nd International Workshop on Computational Approaches to Historical Language Change 2021*, pages 64–74, Online. Association for Computational Linguistics.

Radim Řehůřek and Petr Sojka. 2010. Software Framework for Topic Modelling with Large Corpora. In *Proceedings of the LREC 2010 Workshop on New Challenges for NLP Frameworks*, pages 45–50, Valletta. ELRA.

A Scatter plot and correlation of the variables

Figure 5 displays pair-wise scatter plots between the standardized variables for French–Spanish cognate pairs. On the top right of each panel is shown Pearson's correlation coefficient.

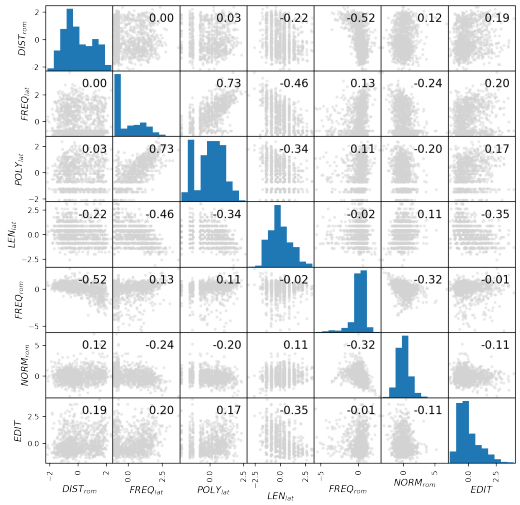

Figure 5: Scatter plot and correlation of the standardized variables for French–Spanish cognate pairs.

Character Jacobian: modeling Chinese character meanings with deep learning model

Yu-Hsiang Tseng
Graduate Institute of Linguistics
National Taiwan University
seantyh@gmail.com

Shu-Kai Hsieh
Graduate Institute of Linguistics
National Taiwan University
shukaihsieh@ntu.edu.tw

Abstract

Compounding, a prevalent word-formation process, presents an interesting challenge for computational models. Indeed, the relations between compounds and their constituents are often complicated. It is particularly so in Chinese morphology, where each character is almost simultaneously bound and free when treated as a morpheme. To model such word-formation process, we propose the Notch (Nonlinear Transformation of Character embeddings) model and the character Jacobians. The Notch model first learns the non-linear relations between the constituents and words, and the character Jacobians further describe the character's role in each word. In a series of experiments, we show that the Notch model predicts the embeddings of the real words from their constituents and helps account for the behavioral data of the pseudowords. Moreover, we also demonstrated that character Jacobians reflect the characters' meanings. Taken together, the Notch model and character Jacobians may provide a new perspective on studying the word-formation process and morphology with modern deep learning.

1 Introduction

Recent years have witnessed a growing interest in modeling the internal semantic dynamics of compounds. Indeed, compounding is sometimes argued as a language universal, and it is claimed to be *protolinguistic fossils* which modern languages frequently elaborate. Compounds are usually loosely defined as forming words with two independent words, such as blackboard and pineapple (Libben, 2014; Bauer, 2009; Jackendoff, 2002). It is apparent that the number of potential combinations is already enormous, even in this simplest form of the two-constituent compound.

The productivity of compounding is particularly evident in Chinese word-formation. Due to the debating nature of Chinese wordhood, there is a vague boundary between word-forming affix and bound root when treating a Chinese character (字 zì) as a morpheme (Huang et al., 2017; Hsieh et al., 2018; Tseng et al., 2020). There is considerable flexibility in Chinese characters, where morphemes can be joined with one another either preceding or following them. For example, the following four words all start with the characters which are the ends of the previous one: 長老 zhǎng lǎo "elder", 老師 lǎo shī "teacher", 師範 shī fàn "teacher-training", 範圍 fàn wéi "area". The versatility of characters leads Hoosain (1992) to describe Chinese text as "a continuous parade of meaningful individual characters (morpheme)" (see Packard (2000) for a complete introduction in Chinese morphology).

However, the productivity of Chinese characters is hard to capture by a computational model. A traditional natural language pipeline starts with word segmentation, which removes the sublexical cues (characters) in the first step. Even for the later distributional semantic model with sub-word information, such as FastText (Bojanowski et al., 2016), it cannot accommodate the different meanings carried by individual Chinese characters. Recent deep learning models, e.g., BERT (Devlin et al., 2018), provide contextualized embeddings of each token. But, as the final representations are mixed (hence contextualized), it is unclear how the embeddings could relate to the original character tokens.

Therefore, we propose the Notch model (Nonlinear Transformation of Character embeddings), and with which we derive character Jacobians. We first train the Notch model to learn the relationships between the constituent

Proceedings of the 29th International Conference on Computational Linguistics, pages 152–162
October 12–17, 2022.

characters' and the whole word's meaning. Next, we show that character Jacobians derived from the model capture the different roles of characters in the whole word. The rest of this paper is organized as follows. First, in section 2, we briefly review the relationship between Chinese characters and words and their complicated behaviors in Chinese morphology. Section 3 describes the Notch model and shows that it not only predicts the real words but helps account for the behavioral data of pseudoword. Finally, in 4, we explore the character Jacobians and evaluate them on a Chinese affixoid dataset.

2 Related works

The vague boundary between affixation and compounding does not only occur in Chinese word-formation. For example, Plag (2003) points out the problem of *neoclassical elements*, such as the word *bio-logy*. While one would tend to treat *bio-*, and *-logy* as prefixes and suffixes, which would violate the basic assumption about the word structure; that is, there will be no *root* in this word. Plag argues that these words, often called combining forms, should be best treated as compounds.

Studying compounds and their constituents poses interesting questions in computational modeling. As the meaning of the compounds may be free from its composing elements, determining the meaning of a newly encountered compound is thus difficult (Jackendoff, 2002). One interesting attempt is to model the meanings of the constituents separated from their free-word counterparts (Günther and Marelli, 2021; Libben, 2014). The difference between the *as-constituent* and free-word representations is called *semantic shift*. A linear model is then built to simultaneously estimate the semantic shifts of the constituents and their linear relations with the compounds by linear algebra.

In Chinese word formation, a character may play different morpho-semantic roles in different words, even if they are in the same position. Therefore, it may not be straightforward to accommodate such versatility into a single linear transformation. Yet, we could consider the relations between the constituents and compounds as a complex non-linear function; the linear transformation is then a local approximation at that specific local neighborhood. Here, we use the Jacobian matrix to obtain the local linear approximation, which is previously used to construct a saliency map and understand network properties (Papernot et al., 2016; Wang et al., 2016). However, before we compute and evaluate the Jacobian, we should first build our non-linear function between constituent and word, that is, the Notch model.

3 The Notch model

The purpose of the Notch model is two-fold. First, it should learn the semantic relationships between the variable-length character sequences (i.e., the constituents) and the whole words. As the semantics of constituents and words are both described by a semantic vector space, the model-learned relationship is essentially a function of $R^{kn} \rightarrow R^{n}$, with n being the semantic space's dimension and k being the word length. Second, the model also provides the Jacobians with which we characterize the character's role in a given word. Therefore, we first train the model to predict the whole word's embedding from the embeddings of its constituents. [1]

3.1 Model training

We trained the Notch model to learn the relations between whole words and constituents in a given embedding space. The embedding space we used was the Chinese word embeddings from Tencent AI lab (Song et al., 2018). The embedding dataset consisted of two million words with 100 dimensions. As the Tencent embedding is a more task-oriented NLP resource, entries (both simplified and traditional Chinese) included may contain both fine-grained words (words from conventional word segmentation) and coarse-grained words (short phrases or compounds in linguistic senses) [2]. Therefore, we chose the first 500 thousand of them for the more commonly used words. For comparison, in a manually-segmented Taiwan Mandarin corpus

[1] The code is available at https://github.com/lopentu/character-jacobian

[2] The steps of word segmentation of Tencent embeddings are described on https://ai.tencent.com/ailab/nlp/en/embedding.html

of 15M characters (CKIP, 1998), the number of unique words (word types) is 217K, and 122K of them occurred more than once in the corpus (i.e., words that are not hapax legomena).

These 500 thousand words are further randomly split into 490K training words and 10K testing words. Each word corresponds to a 100-dimension word vector. The word vectors were all first normalized into unit length. The training words comprise 1,809,674 character tokens, which are 8,749 unique characters (character types). There are 7,792 single-character words, 119,062 two-character words, 103,099 three-character words, 166,330 four-character words, and 93,717 words having five or more characters.

The Notch model's architecture was based on a pre-trained BERT (Devlin et al., 2018) (based on `bert-base-chinese` model) and followed by a task-specific fully-connected layer. At first, the model took a sequence of variable-length characters as input, which were first tokenized by character with the pretrained `bert-base-chinese` tokenizer. Next, the BERT's encoded representation of the first `[CLS]` token was further transformed with a fully-connected layer, which is responsible for projecting the embeddings from the BERT model space of 768-dimension into Tencent's embedding space of 100-dimension. The model was trained with a mean-squared-error objective, where the model tried to minimize the error between the predicted embeddings and the actual embeddings. AdamW (Loshchilov and Hutter, 2017) was used for optimization. The learning rate was 1e-4, β_1 was 0.9, $beta_2$ was 0.999 and L2 weight decay was 0.01. The learning rate was first warmed up for 100 steps and linearly decayed for the rest of the training. The model was trained for one epoch with a batch size of 32. The training took 25 minutes on a P100 GPU.

3.2 Evaluation on real words

We evaluate the model with the top-k accuracies of its predicted embeddings. Specifically, if the model's predicted vectors have the true embeddings as their closest k neighbors, the model's predictions are counted as correct.

The evaluation results are shown in Table 1. The overall top 1 accuracy of the Notch model's predicted embeddings is .67, the top 10 accuracy is .84. As the model's predictions are compared among the 10,000 candidates' word embeddings, the chance level of the top 1 and top 10 accuracy would be 1e-4 and 1e-3. The results indicate the model captures the relationships between the constituent characters and the corresponding word embeddings. The accuracies vary among different word lengths. The two-character word's accuracy is lower than the four-character one's (.63 vs. .75, respectively). As the Chinese four-character words are mostly idioms known for their semantic opaqueness, the pattern might initially seem counter-intuitive. However, a closer look into the word embeddings reveals that these four-character words are mostly coarse-grained words, which act like short-phrases. For example, 乘坐高鐵, which could be considered a two-word short phrase, 乘坐 chéng zuò "riding" and 高鐵 gāo tiě "high-speed rail".

Len.	N	Top 1	Top 5	Top 10
1	162	.73	.85	.86
2	2,522	.63	.78	.81
3	2,123	.66	.79	.84
4	3,375	.75	.87	.90
≥5	1,818	.57	.72	.77
All	10,000	.67	.80	.84

Table 1: Top-k accuracies when predicting word embeddings from the constituents. All top-k accuracies are calculated based on the whole testing set. That is, the chance levels are randomly guessing a word among the 10,000 words regardless of their word length.

To further explore the model predictions, Table 2 shows samples of the prediction errors. The error patterns indicate that the model predicts the *meanings* not only from the character semantics but the general word-level information. For example, the second one is a transparent two-character word, 脫除 tuō chú "get rid of"; the composing characters of which both have meanings related to removal. The model thus consistently predicts the words with related meanings. Moreover, the predictions may also be related to word-level properties. For instance, the third example is the name of green tea, the meaning of

	Targets	Predictions	Translations
1	司仪 (16)	钦差; 刘大人; 侍立	*emcee: imperial commisioner; Lord Liu; wait upon*
2	脱除 (19)	去除, 消退, 卸去	*get rid of: discard; fade away; remove*
3	碧螺春 (17)	关汉卿, 庐山恋, 孝庄秘史	*bi-luo-chun (a green tea): Guan Hanqing, a Chinese playwright; Romance on Lushan Mountain, a Chinese movie; Xiaozhuang Epic, a TV drama*
4	观看中 (15)	正片中, 表演中, 故事当中	*watch-ing: in the film; during the performance; in the story*
5	社交距离 (16)	彼此了解, 交流能力, 像朋友一样	*social distancing: mutual understanding; communication skills; (be) like friends*
6	釜底抽薪 (14)	抢功, 破罐子破摔, 千夫所指	*solve the root of the problem (an idiom): take credit; (totally) giving up; blamed by everyone*

Table 2: The prediction errors of the Notch model. The numbers in the brackets are the location of the targets in the predictions; that is, the number 1 indicates the prediction is correct as the Top-1 prediction. The loose translations are provided in the last columns, where the words before the colons are target words, followed by three predictions separated by semicolons.

which is not decomposable to its constituents. However, while not semantically correct, the model's predictions are also other proper playwright or movie names. Similarly, the model does not capture the word meaning for a fully opaque term (e.g., the last one), but the predictions are related to other opaque idioms.

While the model's accuracies might suggest the words' semantics (word embeddings) could be almost determined from their constituents, there are caveats in this interpretation. On the one hand, the model inputs are character tokens, and the model has an embedding layer in its first layer. Therefore the model could learn or tune character embeddings to predict the final word embeddings. In this sense, the model does compute the final word embeddings based on the constituent embeddings. On the other hand, the BERT's transformer architecture keeps mixing and warping the constituent embeddings in its 12 encoding layers (Lee-Thorp et al., 2021; Tolstikhin et al., 2021; Mai et al., 2022); therefore, the final output of the `[CLS]` is no longer a simple linear combination of their constituents. That is, the constituent embeddings are themselves changing based on extra-constituent information; thus, it is context-dependent (Baggio et al., 2012). Finally, we are using a distributed semantic model to operationalize word semantics, the predictability may be the characteristic of the vector semantic model rather than word semantics itself.

Nonetheless, it is still interesting to ask what the Notch model captures among these constituents. Having demonstrated the Notch model could predict the word embeddings in real words, we next examine whether the model could predict the embeddings of novel constituent combinations, i.e., pseudowords. As pseudowords, by definition, do not occur in our linguistic uses, they cannot readily be given word embeddings to be compared to model predictions. Therefore, we turn to behavioral data in psycholinguistics to evaluate the model's prediction of pseudowords.

3.3 Evaluation on Pseudowords

In this experiment, we try to evaluate the Notch model by how the predicted embeddings shed light on the behavioral data of pseudowords. Although word recognition has been well studied, pseudowords in psycholinguistics, especially in lexical decision tasks (LDT), are experimental stimuli which experimenters have little interest in. However, regardless of real words or pseudowords, the information is accumulated over time for both words and pseudowords in an LDT task (Yap et al., 2015; Ratcliff et al., 2004). That is, pseudowords are not *devoid of meaning* (Hendrix and Sun, 2021; Chuang et al., 2021). Moreover, the behavioral data of the pseudowords provide unique insights into lexical processes when the word-level information does not yet influence them.

Therefore, we derive two semantic indices from the Notch model and use statistical mod-

els to examine their effects on pseudoword processing. First, we identify the 50 nearest semantic neighbors of a pseudoword based on the model predicted vectors. The neighbors are selected among the whole 500K words in the dataset. Next, we create a vector whose elements are the 50 neighbor distances in ascending order, π_w, where w is the given pseudoword. Basing on the vector π_w, we calculate the average distance of the closest 5 neighbors, `smSimTop5`, and the difference of distances between the .90 and .10 quantiles, `smSimRange`. Specifically, the `smSimTop5` indicates how close the pseudoword is to a real word, and `smSimRange` captures how densely populated the pseudoword located in the word embedding space. The smaller the `smSimRange`, the more densely-packed the neighbors are in a given (hyper-)sphere; the larger the value, the more sparsely-populated the pseudoword is located.

We use the pseudoword data from the MELD-SCH dataset (Tsang et al., 2018). The dataset contains lexical decision data on 25,156 words from 504 native Chinese speakers. There are equal numbers of real words and pseudowords in the dataset, i.e., 12,578 words for each word type. Each of these word types has 1,020 one-character, 10,022 two-character, 949 three-character, and 587 four-character words. The dataset also includes additional character-level information for each word, e.g., number of strokes, character frequency, and number of meanings. This additional information serves as the statistical model's controlled variables, or the semantic indices' effects may be proxies or surrogates for other character-level effects. The inclusion of character-level variables also implies a single statistical model could not accommodate words of different lengths as they require a different number of variables. Therefore, we select two-character words as they are the most commonly occurred words in the MELD-SCH and the Tencent word embedding dataset.

We include 12 explanatory variables on this analysis, which are two semantic indices derived from the Notch model, namely `smSimTop5` and `smSimRange`; 10 character level indices, that is, character frequency for first and second constituents (`C1logcf`, `C2logcf`), number of strokes (`C1stroke`, `C2stroke`), number of words formed(`C1lognwf`, `C2lognwf`), number of meanings(`C1nom`, `C2nom`), number of pronunciations (`C1nop`, `C2nop`). These variables are used to predict error rates and response time in their respective models [3].

The statistical results are shown in Figure 1. The left panel shows the variable importance of 12 variables with respect to error rates and response times. The importance scores are estimated by the "mean decrease in accuracy" following the permutation principle in a 100-tree random forest (Hothorn and Zeileis, 2015). The figure shows the Notch-derived semantic index, `smSimRange`, is among the most important features in both error rate and response time models, along with the number of the word formed in second constituents `C2lognwf`. The other semantic index, `smSimTop5`, however, is the fourth most important feature in error rates but is the seventh one in response time.

A closer look at the variable effects with the generalized additive model (GAM) (Wood, 2011) also shows consistent patterns. [4] Here, we include the semantic indices and other important variables, namely the number of words formed and character frequency in the GAM models. All included variables are highly significant in the models. In particular, the partial effects shown in the center and right panels of Figure 1 indicate a nearly positive linear effect of both `smSimRange` and `smSimTop5`. The patterns suggest that the Notch-derived indices, especially `smSimRange`, help explain the behavioral error rates and response times. Specifically, when the pseudowords are in a less populated area (large `smSimRange` values), the participants tend to respond slower

[3] Due to their distribution characteristics, the character frequency, and error rates, number of meanings, and number of pronunciations are log-transformed. Response time is reciprocally transformed and multiplied by -1000 to keep the same sign and direction.

[4] Two GAM models have nearly the same explanatory variables, only RT model has an additional `C1stroke` due to its importance in the random forest analysis. `smSimRange` and `smSimTop5` are included as thin plate regression spline smoothing terms. Other pairwise character-level information is included as tensor-product smoothing terms. The worst concuvrities of the semantic indices are .13 and .09. The Pearson correlation between `smSimRange` and `smSimTop5` is .20

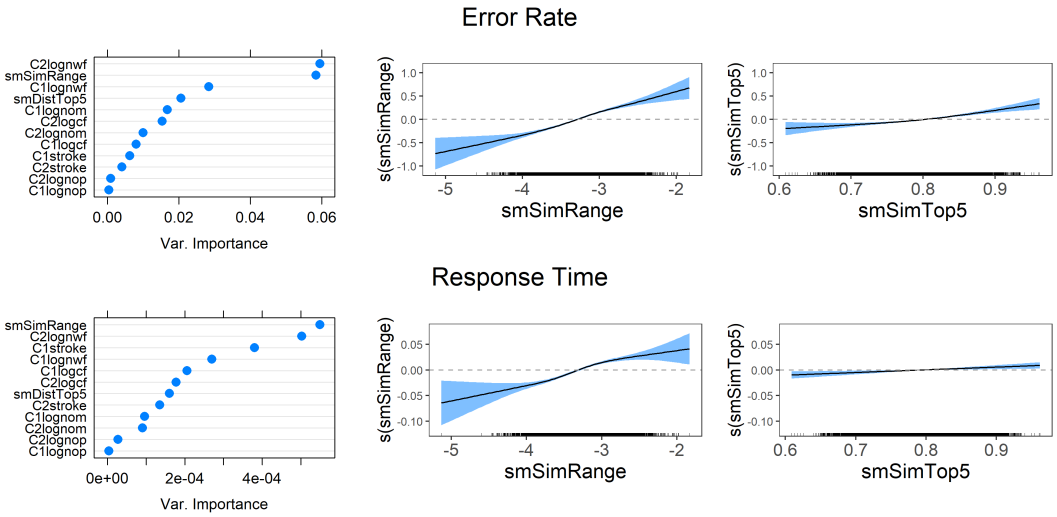

Figure 1: The statistical results of the Notch-model derived semantic indices: `smSimDist` and `SimRange`. The left panel shows the variable importance. The center and right panels show the partial effects estimated by GAM.

and make more errors. Similarly, when the pseudowords have similar real word neighbors (larger `smSimTop5`), the responses tend to be slow and more error-prone. Although the underlying lexical process is inevitably more complicated, the results already demonstrate the relevance of the Notch-model derived semantic indices.

After real words and pseudowords evaluations, we established the Notch model can capture relationships between the constituent and word embeddings. Such capacity would suggest the model may learn the representations of how a character would function as a constituent in a word. Therefore, we further probe into the model to examine its character-level representations.

4 Character Jacobian

In this section, we extract and evaluate the character-level information in the Notch model. Being a model based on BERT, the Notch model learns the contextualized embeddings of each token in its input sequence. The characteristic has been applied to the word sense disambiguation task, where the model is successfully employed to create sense embeddings (Loureiro and Jorge, 2019; Scarlini et al., 2020). Therefore, an interesting question to ask is, how could we extract the character-level context-dependent information from the Notch model?

Specifically, this character-level information should ideally differentiate the word contexts the character occurs. For example, the character 手 shǒu means "hand" as an independent word. However, the same character could occur at the start of the word and carries different meanings, such as 手氣 shǒu qì "luck" and 手臂 shǒu bì "arm". In addition, the meanings are also different when occurring at the end of the word: 歌手 gē shǒu "singer", or 分手 fēn shǒu "break up". That is, the character-level information should have different representations for the same character when it is used differently.

There are at least three different approaches to extract character-level information from the Notch model. The first one is to use the BERT input embeddings directly. Since the `bert-base-chinese` is a by-character tokenizer, the input embeddings act like character embeddings. However, this approach does not have access to other characters in the same word; therefore, it is not context-sensitive as we require. The second approach uses the token embeddings in the later layers of the BERT model, with the advantage that the token embeddings would be context-sensitive after layers of transformation. Yet, it also implies that the token embeddings have already incorporated or mixed the represen-

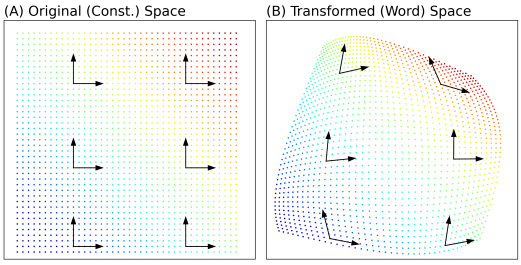

Figure 2: A visualization of the Jacobian matrix. The rectangular grid (left) is transformed into a warped mesh (right) by a non-linear function. The Jacobian matrix describes how the grid is transformed locally, as shown by the black arrows.

tations from other tokens. Hence, it is not straightforward to attribute the token to the input character anymore. The third approach, which is simultaneously context-sensitive and character-specific, is through the Jacobian matrix.

4.1 Jacobian matrix

The Jacobian matrix is a matrix whose elements are the first-order partial derivatives of a vector-valued function:

$$\nabla F(X) = \begin{bmatrix} \frac{\partial F_1(X)}{\partial x_1} & \cdots & \frac{\partial F_1(X)}{\partial x_n} \\ \vdots & \ddots & \vdots \\ \frac{\partial F_m(X)}{\partial x_1} & \cdots & \frac{\partial F_m(X)}{\partial x_n} \end{bmatrix}$$

As the Notch model's inputs are discrete characters, we need first to convert the input characters to vectors to compute the Jacobian. Therefore, we leverage the input embeddings in the BERT embedding layer and convert the characters to vectors without introducing additional parameters. The converted embeddings are from the raw embeddings that are not yet involved with positional and sequence type encodings. The converted vectors are then used as the model input. Therefore, the Notch model could be considered as a function that brings the input vectors from the space of $R^{768 \cdot k}$, where k is the word length, to the word embedding space of R^{768}.

The Jacobian of the input characters could be considered as the linear projection matrix that best approximates the non-linear transformation at a specific location in space. Informally, it describes how a slight perturbation of the character's input vector *nudges* the word vectors in the embedding space. Figure 2 shows a simple illustration of a non-linear function transforming a rectangular grid into a warped mesh. Note that in the figure, although the transformation is non-linear, each transformed arrows are still tangent to its transformed grid. The Jacobian provides a way to describe how each black arrow are warped in its specific location.

Moreover, as the function (i.e., the Notch model) is context-sensitive to its input characters, the Jacobian automatically encodes the context in which the character occurs. In addition, the Jacobian is the first derivative of the character embeddings. We can directly attribute the matrix to that character. Thus, the Jacobian matrix satisfies the requirements of character-level information.

The downside of this approach would be that obtaining Jacobian matrices is relatively computationally expensive. As opposed to the input embedding and token embedding approach, which only requires a vector of 768 dimensions (i.e., the model dimension) for each character in each context, the Jacobian approach will require a 100×768 matrix and additional steps to compute. However, the issue should be alleviated with the advancement of algorithms (Baydin et al., 2022) and the hardware.

In the following evaluation, we only focus on the two-character word hence the character Jacobian, $J^{(c_i)}$, is defined by the Jacobian, $\nabla F(c_i)$:

$$J^{(c_i)} \triangleq \nabla F(c_i) = \frac{\partial F(\nu_1, \nu_2)}{\partial \nu_i}$$

where ν_1 and ν_2 are the corresponding input vectors.

4.2 Evaluation on affixoids clustering

In this evaluation, we aim to examine to what extent the character Jacobians captures the characters' role in the words. The dataset we used is the "Common Affixoids Database" compiled by the CKIP group at Academia Sinica, Taiwan. The dataset includes different roles of a character, consisting of prefixes,

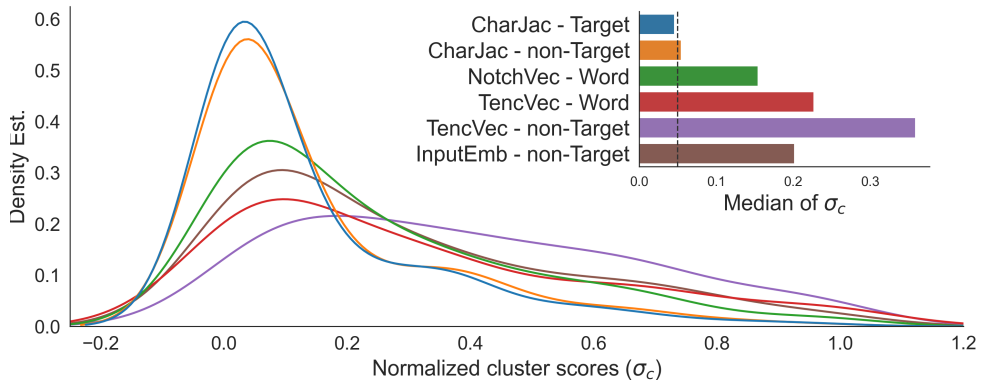

Figure 3: The distribution of normalized cluster scores, σ_c under different conditions. The inset figure shows the median of σ_c. The dashed line marks the position of .05 for the visual reference.

suffixes, or morphological roots. [5]. Characters with different roles or different meanings will have separate entries. There are 2,471 unique characters and 4,893 entries in the dataset.

Among these entries, we first identified a set of characters that have more than one entry in the dataset. For example, 土 had one entry indicating land or clay, such as 土石 tǔ shí "earth and stones" or 土堤 tǔ tí "embankment"; also, it had another entry indicating "native or local", such as 土狗 tǔ gǒu "native dog" or 土著 tǔ zhù "indigenous people". We extracted at least two instances (i.e., example words) for each entry and at most five instances whose word frequencies were larger than one. As a result, we selected 796 unique characters and 1,765 entries. Among these entries, there are 7,072 instances.

For each character, we compute the character Jacobians of each instance in different entries. For example, 土 has two entries, and each entry has two instances, then we compute four character Jacobians for each instance. These Jacobians are then compared to each other to obtain a distance measure. Here we use the L1-norm. If the character Jacobians indeed capture the characters' roles in different instances, the distances between character Jacobians of the same entry should be closer to those from other entries. That is, we could evaluate the character Jacobians by measuring the clustering performance implied by their pairwise distances. Here, we assess the clustering with the averaged silhouette score (Rousseeuw, 1987).

Furthermore, to establish a reference, we build a null distribution for each averaged silhouette score with random permutations. We randomly permute the instance labels 1,000 times for each character and compute one silhouette score. That is, we calculate the same averaged silhouette score as if the meanings of the characters no longer group the example words. These permuted scores will form a null distribution to which the silhouette score will be compared. Finally, we compute normalized cluster scores σ_c for each character to indicate the clustering performance. The normalized cluster scores are defined by $1 - P_{\text{null}}(X)$, that is, the probability of obtaining the values higher than the observed silhouette scores assuming the null hypothesis is true. A lower σ_c would indicate the corresponding silhouette score is less likely to result from the random chance, hence, the better clustering.

In addition to the character Jacobian of the target character (`CharJac-Target`, e.g. 土 in 土石), we also include five conditions for comparison. The `CharJac-nonTarget` refers to the character Jacobian of the character in non-target position (e.g. 石 in 土石). This condition show the roles of the other character in the same word. Next, the `NotchVec-Word` and `TencVec-Word` are computing the same silhouette scores but using the word vectors from the Notch predictions and Tencent embeddings respectively. These conditions provide the baseline for word-level semantics. Finally, the `TencVec-nonTarget` and `InputEmb-nonTarget` both compute the

[5] The dataset is publicly available at http://turing.iis.sinica.edu.tw/affix

scores based on the embeddings of the character at non-target position. The former one uses the single-character word embeddings from Tencent dataset, while the later one uses the input embeddings from the first layer of Notch model. These two conditions serve as the baselines for the information the non-target character could provide in clustering.

The results are shown in Figure 3. It can be seen the two conditions of character Jacobian have distinct distributions compared to other conditions. The inset plot in Figure 3 further shows the median of σ_c for each condition. It is apparent that the character Jacobians, regardless of the target or non-target position, could form the clusters better, as indicated by the lower values compared to other conditions. The comparison across conditions additionally reveal that the clustering results cannot be achieved by the word-level semantic (the `NotchVec-Word` and `TencVec-Word` condition), or by considering the character as single-character alone (the `TencVec-nonTarget` and `InputEmb-nonTarget`). The results demonstrate that character Jacobian captures an important aspect of the character's role in the word.

5 Conclusion

In this paper, we present the Notch model, from which we derive the character Jacobians. In a series of experiments, we show that the model predicts the embeddings of the real words from their constituents and helps account for the behavioral data of the pseudowords. In addition, we also show that character Jacobians capture characters' roles in the words, which reflect the meanings of the characters.

The approach to study compounding by modeling the word embeddings and exploring their Jacobians could also be applied to other languages. Multilingual language models and word embeddings are readily available in the community. However, the most interesting question is how to study compounding if the language's writing system may introduce spaces in the compounds, such as the case in English. The commonly-used word embeddings only include entries with no interword spaces. That is, there will be entries for *earthquake*, *airport*, but no entries for *rush hour*, *coffee mug*. Moreover, it could be argued that the interword spaces bear significance in cognitive processing (Juhasz et al., 2005). It will be thus an interesting future work to systematically study compounding in this case.

Character Jacobians open up new possibilities to study Chinese characters or morphology with deep learning models. However, as the Jacobian is an abstract mathematical object, other future works include further investigating its relation with morphological rules and lexical categories and how it connects to the distributional semantics.

Acknowledgements

This study was supported by Ministry of Education, Taiwan, Grant Number 110L9A001; and Ministry of Science and Technology (MOST), Taiwan, Grant Number MOST 110-2634-F-001-011.

References

Giosuè Baggio, Michiel Van Lambalgen, and Peter Hagoort. 2012. The processing consequences of compositionality. In *The Oxford handbook of compositionality*, pages 655–672. Oxford University Press.

Laurie Bauer. 2009. Typology of compounds. In *The Oxford handbook of compounding*.

Atılım Güneş Baydin, Barak A. Pearlmutter, Don Syme, Frank Wood, and Philip Torr. 2022. Gradients without backpropagation.

Piotr Bojanowski, Edouard Grave, Armand Joulin, and Tomas Mikolov. 2016. Enriching word vectors with subword information. *arXiv preprint arXiv:1607.04606*.

Yu-Ying Chuang, Marie Lenka Vollmer, Elnaz Shafaei-Bajestan, Susanne Gahl, Peter Hendrix, and R Harald Baayen. 2021. The processing of pseudoword form and meaning in production and comprehension: A computational modeling approach using linear discriminative learning. *Behavior research methods*, 53(3):945–976.

CKIP. 1998. *Academia Sinica Balanced Corpus: Content and description (Technical Report No.95-02/98-04)*.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*.

Fritz Günther and Marco Marelli. 2021. Caoss and transcendence: Modeling role-dependent constituent meanings in compounds. *Morphology*, pages 1–24.

Peter Hendrix and Ching Chu Sun. 2021. A word or two about nonwords: Frequency, semantic neighborhood density, and orthography-to-semantics consistency effects for nonwords in the lexical decision task. *Journal of Experimental Psychology: Learning, Memory, and Cognition*, 47(1):157.

Rumjahn Hoosain. 1992. Psychological reality of the word in chinese. In *Advances in psychology*, volume 90, pages 111–130. Elsevier.

Torsten Hothorn and Achim Zeileis. 2015. partykit: A modular toolkit for recursive partytioning in R. *Journal of Machine Learning Research*, 16:3905–3909.

Shu-Kai Hsieh, Yu-Hsiang Tseng, Chih-Yao Lee, and Chiung-Yu Chiang. 2018. Fluid annotation: A granularity-aware annotation tool for Chinese word fluidity. In *Proceedings of the Eleventh International Conference on Language Resources and Evaluation (LREC 2018)*, Miyazaki, Japan. European Language Resources Association (ELRA).

Chu-Ren Huang, Hsieh Shu-Kai, and Chen Keh-Jiann. 2017. *Mandarin Chinese words and parts of speech: A corpus-based study*. Routledge.

Ray Jackendoff. 2002. *Foundations of language: Brain, meaning, grammar, evolution*. Oxford University Press, USA.

Barbara J Juhasz, Albrecht W Inhoff, and Keith Rayner. 2005. The role of interword spaces in the processing of english compound words. *Language and cognitive processes*, 20(1-2):291–316.

James Lee-Thorp, Joshua Ainslie, Ilya Eckstein, and Santiago Ontanon. 2021. Fnet: Mixing tokens with fourier transforms.

Gary Libben. 2014. The nature of compounds: A psychocentric perspective. *Cognitive neuropsychology*, 31(1-2):8–25.

Ilya Loshchilov and Frank Hutter. 2017. Decoupled weight decay regularization.

Daniel Loureiro and Alípio Jorge. 2019. Language modelling makes sense: Propagating representations through WordNet for full-coverage word sense disambiguation. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 5682–5691, Florence, Italy. Association for Computational Linguistics.

Florian Mai, Arnaud Pannatier, Fabio Fehr, Haolin Chen, Francois Marelli, Francois Fleuret, and James Henderson. 2022. Hypermixer: An mlp-based green ai alternative to transformers.

Jerome L. Packard. 2000. *The Morphology of Chinese: A Linguistic and Cognitive Approach*. Cambridge University Press.

Nicolas Papernot, Patrick McDaniel, Somesh Jha, Matt Fredrikson, Z Berkay Celik, and Ananthram Swami. 2016. The limitations of deep learning in adversarial settings. In *2016 IEEE European symposium on security and privacy (EuroS&P)*, pages 372–387. IEEE.

Ingo Plag. 2003. *Word-formation in English*. Cambridge University Press.

Roger Ratcliff, Pablo Gomez, and Gail McKoon. 2004. A diffusion model account of the lexical decision task. *Psychological review*, 111(1):159.

Peter J Rousseeuw. 1987. Silhouettes: a graphical aid to the interpretation and validation of cluster analysis. *Journal of computational and applied mathematics*, 20:53–65.

Bianca Scarlini, Tommaso Pasini, and Roberto Navigli. 2020. SensEmBERT: Context-Enhanced Sense Embeddings for Multilingual Word Sense Disambiguation. In *Proceedings of the Thirty-Fourth Conference on Artificial Intelligence*, pages 8758–8765. Association for the Advancement of Artificial Intelligence.

Yan Song, Shuming Shi, Jing Li, and Haisong Zhang. 2018. Directional skip-gram: Explicitly distinguishing left and right context for word embeddings. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 2 (Short Papers)*, pages 175–180, New Orleans, Louisiana. Association for Computational Linguistics.

Ilya Tolstikhin, Neil Houlsby, Alexander Kolesnikov, Lucas Beyer, Xiaohua Zhai, Thomas Unterthiner, Jessica Yung, Andreas Steiner, Daniel Keysers, Jakob Uszkoreit, Mario Lucic, and Alexey Dosovitskiy. 2021. Mlp-mixer: An all-mlp architecture for vision.

Yiu-Kei Tsang, Jian Huang, Ming Lui, Mingfeng Xue, Yin-Wah Fiona Chan, Suiping Wang, and Hsuan-Chih Chen. 2018. Meld-sch: A megastudy of lexical decision in simplified chinese. *Behavior research methods*, 50(5):1763–1777.

Yu-Hsiang Tseng, Shu-Kai Hsieh, Pei-Yi Chen, et al. 2020. Computational modeling of affixoid behavior in chinese morphology. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 2879–2888.

Shengjie Wang, Abdel-rahman Mohamed, Rich Caruana, Jeff Bilmes, Matthai Plilipose, Matthew Richardson, Krzysztof Geras, Gregor Urban, and Ozlem Aslan. 2016. Analysis of deep neural networks with extended data jacobian matrix. In *Proceedings of The 33rd International Conference on Machine Learning*, volume 48 of *Proceedings of Machine Learning Research*, pages 718–726, New York, New York, USA. PMLR.

S. N. Wood. 2011. Fast stable restricted maximum likelihood and marginal likelihood estimation of semiparametric generalized linear models. *Journal of the Royal Statistical Society (B)*, 73(1):3–36.

Melvin J Yap, Daragh E Sibley, David A Balota, Roger Ratcliff, and Jay Rueckl. 2015. Responding to nonwords in the lexical decision task: Insights from the english lexicon project. *Journal of Experimental Psychology: Learning, Memory, and Cognition*, 41(3):597.

COMMA: Modeling Relationship among Motivations, Emotions and Actions in Language-based Human Activities

Yuqiang Xie Yue Hu[†] Wei Peng Guanqun Bi Luxi Xing
Institute of Information Engineering, Chinese Academy of Sciences, Beijing, China
School of Cyber Security, University of Chinese Academy of Sciences, Beijing, China
{xieyuqiang,huyue,pengwei,biguanqun,xingluxi}@iie.ac.cn

Abstract

Motivations, emotions, and actions are interrelated essential factors in human activities. While motivations and emotions have long been considered at the core of exploring how people take actions in human activities, there has been relatively little research supporting analyzing the relationship between human mental states and actions. We present the first study that investigates the viability of modeling motivations, emotions, and actions in language-based human activities, named **COMMA** (**Co**gnitive Fra**m**ework of Hu**m**an **A**ctivities). Guided by **COMMA**, we define three natural language processing tasks (emotion understanding, motivation understanding and conditioned action generation), and build a challenging dataset **HAIL**[‡] through automatically extracting samples from Story Commonsense. Experimental results on NLP applications prove the effectiveness of modeling the relationship. Furthermore, our models inspired by **COMMA** can better reveal the essential relationship among motivations, emotions and actions than existing methods.

1 Introduction

Human activities are continuous interactions between external environment (physical world and social events, etc.) and internal mind (motivations, emotions, etc.). For example, Figure 1 demonstrates human activities of character 'I' about 'eating bread' in external environment, as well as the mental states of 'I'. The motivation of character 'I' is a **physiological** need. Conditioned on this motivation and history actions, 'I' **enjoyed the part of bread** on the direction of **joy** emotion. While human mental states have long been considered at the core of exploring how people take actions between the lines in language-based human activities, there has been relatively little research supporting analyzing the relationship between human mental states and actions. It is challenging to comprehensively model the relationship of motivations, emotions and actions in language-based human activities, which can allow researchers to reason the essential causes of human activities from the cognitive perspective and supply reasonable explanations. This technology will have a profound impact on various natural language processing (NLP) downstream applications, such as intelligent dialogue, controllable text generation, recommendation systems, and public opinion analysis.

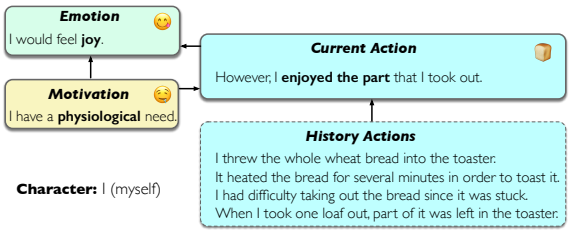

Figure 1: An example of human activity. History actions and motivation cause current action. And emotion is the effect of motivation and actions. Here, motivation/action/emotion is colored by yellow/cyan/green.

In recent years, traditional sentiment analysis technology has been widely used (Socher et al., 2013; Hamilton et al., 2016), which mainly focuses on sentiment detection. Although the current state-of-the-art sentiment analysis system can detect the polarity of text (Zhang et al., 2018) or consider fine-grained categories (a.k.a. aspects) to make predictions (Pontiki et al., 2016), the analysis of predictions and interpretations of its causes are still limited. Lately, a large amount of work introduces human motivations into sentiment analysis and action analysis (Rashkin et al., 2018a,b; Sap et al., 2019a,b; Peng et al., 2022a). However, the aforementioned works focus on the analysis of the relationship between "motivations and actions" or "emotions and actions", without modeling a unified consideration of the relationship among motivations, emotions and actions.

[†]Corresponding author.

[‡]We will make our dataset and code publicly available at https://github.com/IndexFziQ/COMMA.

Proceedings of the 29th International Conference on Computational Linguistics, pages 163–177
October 12–17, 2022.

Researches on human activities have increased over the past two decades with many fields contributing including psychology, computer science and so on. In this paper, we focus on works in two human mental states, motivations and emotions, that drive human activities. As for motivation, psychologist Hull (Hull, 1974) believes that motivation is the drive for human actions and explains why people initiate, continue or terminate a certain action at a particular time. From area of emotion, numerous theories (Cacioppo and Gardner, 1999; Kagan, 2007; Smith, 2016) that attempt to explain the origin, function, and other aspects of emotions have fostered more intense research on emotion topic. Psychologist Plutchik (Plutchik, 1980) establishes a general psycho-evolutionary theory of emotion, which introduces eight specific distinct basic emotions. Each of basic emotions represents adaptation to a prototypical task in human activity.

Aiming at modeling the relationship among human motivations, emotions and actions in language-based human individual activities, we propose a general **Co**gnitive Fra**m**ework of Hu**m**an **A**ctivities (**COMMA**). These relationships will help the researchers of NLP areas track the cause of people's emotions and actions, and give a more reasonable explanation and analysis for results. To verify the effectiveness of our framework, we propose three NLP understanding/generation tasks, including emotion understanding, motivation understanding, and conditioned action generation. More concretely, we construct a new dataset **HAIL** (**H**uman **A**ctivities **I**n **L**ife) by automatically extracting samples with complete mental state annotation from Story Commonsense (Rashkin et al., 2018a). Experimental results on NLP applications prove the effectiveness of modeling the relationship. Furthermore, our models inspired by **COMMA** can better reveal the essential relationship among motivations, emotions and actions than existing methods.

2 COMMA

Human activities are interactions between internal mind of people and the external environment. In this part, we will describe the basic elements of **COMMA** and modeling the relationship among elements in details.

2.1 Basic Elements in COMMA

Motivations are the innate physical or psychological drives of human beings, and are the origin of human activities. Different psychological theories have different classification rules for human motivations. We utilize *hierarchy of needs* of Maslow (1943) (***physiological needs, stability, love and belonging, esteem, self-actualization***).

Emotions are the psychological responses of motivations to the degree of satisfaction with the external environment. We employ the *wheel of emotions* of (Plutchik, 1980) and use eight basic emotional dimensions (***joy, trust, sadness, surprise, fear, disgust, anger, and anticipation***). It has become a common choice in the existing emotion categorization literature (Mohammad and Turney, 2013; Zhou et al., 2016; Rashkin et al., 2018a).

Actions are people's behaviors that interact with the external environment. Actions are under the psychological condition of "one has a certain need and develops on the direction of future emotion" in this paper. Limited to the annotations of our based data Story Commonsense (Rashkin et al., 2018a), we treat story events as actions, which is a sentence in language-based form. That is, ***a story event equals to an action*** in this paper.

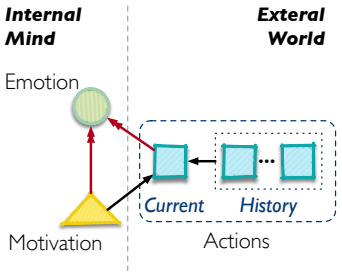

Figure 2: Relationship Modeling in **COMMA**.

2.2 Relationship Modeling

As demonstrate in Fig. 2, **COMMA** is composed of internal mind and external environment, where people own internal mind (motivations, emotions) and take actions in external environment. The solid line and the dotted line represent forward and reverse reasoning respectively. We will attempt to model relationships among the basic elements by answering the next two questions.

Q1: Where are actions from? Following the view of Hull (Hull, 1974), motivation is the drive for human actions. Intuitively, current action also caused by history actions. As shown in Fig. 2, motivation and history actions leads to the development of current action together. Meanwhile, actions develop in the direction of the future emotion. For instance, one wanted to eat, and one could eat some

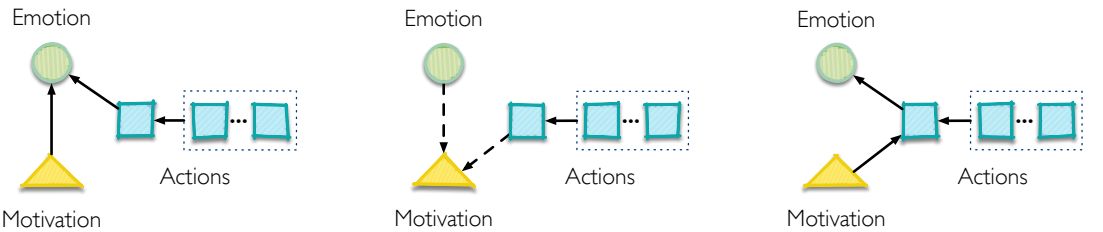

(a) *Emotion Understanding* (b) *Motivation Understanding* (c) *Conditioned Action Generation*

Figure 3: Three tasks for modeling motivations, emotions, and actions in language-based human activities. The solid line and the dotted line represent forward and reverse reasoning respectively.

food and felt happy then. Conversely, one could felt sad because one had nothing to eat.

Q2: Where are emotions from? Emotions are mental states brought on by neurophysiological changes, variously associated with thoughts, feelings, behavioral responses (Panksepp, 1998; Cabanac, 2002). In simplicity, emotions come from the interaction between actions in the external world and complex mental states. In this work, we predigest this complicated process. As demonstrated in Fig. 2, emotion is conditioned by whether the action satisfy the primary motivation, similar to the statement in Li and Hovy (2017). For example, one wanted to eat, and one would happy if he ate some food, either sad if no restaurant opened.

All in all, relationships of motivations, emotions and actions are demonstrated in Fig. 2:

(1) Motivation and history actions cause action;

(2) Emotion is effect of motivation and action.

These relationships will help the researchers track human's motivations, emotions and actions.

3 Tasks and Data

To verify the effectiveness of **COMMA**, we propose emotion understanding, motivation understanding, and conditioned action generation tasks. Correspondingly, we build a **HAIL** dataset by automatically extracting samples with complete mental state annotation from Story Commonsense (Rashkin et al., 2018a).

3.1 Task Definition

The annotations of all elements in **COMMA** is defined as follows:

$\mathcal{A}$: The current action.

$\mathcal{H}$: The history actions.

$\mathcal{C}$: The character of current action.

$\mathcal{M}$: The motivation to drive the current action.

$\mathcal{E}$: The resulted emotion of the current action.

Emotion Understanding (EU) We formulate emotion understanding as sequence classification problems consisting of history actions, current action and people's motivations as context and a objective resulted emotion. As shown in Fig. 3(a), given the motivation $\mathcal{M}$, character $\mathcal{C}$ and all actions ($\mathcal{H}$ and $\mathcal{A}$), the EU task is to select the most plausible emotion $\mathcal{E}$.

Motivation Understanding (MU) As demonstrated in Fig. 3(b), compared with EU, motivation understanding is a reversed reasoning process. Given the emotion $\mathcal{E}$, character $\mathcal{C}$ and all actions ($\mathcal{H}$ and $\mathcal{A}$), the MU task is to reversely reason about the most plausible motivation $\mathcal{M}$.

Conditioned Action Generation (CAG) From Fig. 3(c)), CAG is the task of generating a valid action $\mathcal{A}$ and predicting the desired emotion $\mathcal{E}$ conditioned on the history actions $\mathcal{H}$, character $\mathcal{C}$ and motivation $\mathcal{M}$. Formally, the task requires to maximize $P(\mathcal{A}, \mathcal{E}|\mathcal{H}, \mathcal{C}, \mathcal{M})$.

3.2 Data Collection

To verify the effectiveness of **COMMA**, we construct a new dataset **HAIL** (**H**uman **A**ctivities **I**n **L**ife) for the above three tasks by automatically extracting from the existing resource, Story Commonsense(Rashkin et al., 2018a). Story Commonsense dataset manually annotates human motivations and emotions of the event in daily commonsense stories. It is an important resource for studying the causality of motivations, actions, and emotions in language-based individual activities. Note that, the characters of the actions (story events in this paper) are required to have both motivation and emotion labels in our collected data **HAIL**. In order to obtain such *(motivation, action, emotion)* samples, we align the motivation prediction and emotion prediction data sets of Story Commonsense guided by the story id and the character of the current story

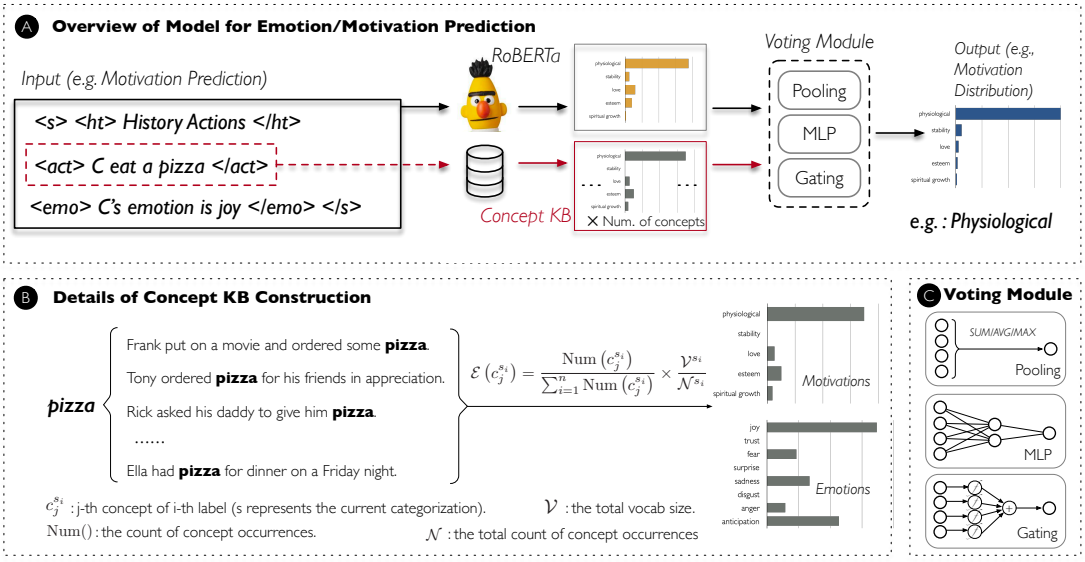

Figure 4: Model overview (A) for emotion and motivation understanding tasks (emotion understanding as an example). B is the detail of concept knowledge base construction. C shows three options of voting module in A.

event. In all, we extract 13,568 examples from Story Commonsense that meet our requirements. The I/O of three tasks are summarized in Table 1. More details, please refer to Appendix A.2.

EU Task		MU Task		CAG Task	
Input	output	Input	output	Input	output
$\mathcal{H}$,$\mathcal{A}$,$\mathcal{C}$,$\mathcal{M}$	$\mathcal{E}$	$\mathcal{H}$,$\mathcal{A}$,$\mathcal{C}$,$\mathcal{E}$	$\mathcal{M}$	$\mathcal{H}$,$\mathcal{C}$,$\mathcal{M}$	$\mathcal{A}$,$\mathcal{E}$

Table 1: Input and Output of tasks in HAIL.

4 Methodology

4.1 Model for Emotion Understanding

Our method combines Human Activity Encoder, Concept Knowledge Base with a voting module component, which is shown in Fig. 4. All inputs are refactored by prompt templates and special tokens (Appendix A.1) to improve understanding.

4.1.1 Human Activity Encoder

Here, we use ROBERTA (Liu et al., 2019) as our language-based human activity encoder. It is a improved robust BERT (Devlin et al., 2019) which shows state-of-the-art results in many NLP tasks. We use the hidden state representation of <s> as the sentence representation h_s.

4.1.2 Concept Knowledge Base

For emotion understanding and motivation understanding tasks, we introduce knowledge bases to calculate the distribution of commonsense knowledge in language-based actions of all motivation/emotion categories. In this paper, commonsense knowledge means commonsense concepts (i.e., words) with significant meanings that appear in language-based actions.

We build the Motivation Concept Knowledge Base (**MCKB**) in three steps. Firstly, we extract representative commonsense concepts (details in Appendix A.3). Then, we count the number of occurrences of each commonsense concepts in the category of motivations. The last step is to calculate word frequency. The knowledge distribution of each concept is computed as below:

$$\mathcal{E}\left(c_j^{s_i}\right) = \frac{\text{Num}\left(c_j^{s_i}\right)}{\sum_{i=1}^{n}\text{Num}\left(c_j^{s_i}\right)} \times \frac{\mathcal{V}^{s_i}}{\mathcal{N}^{s_i}} \quad (1)$$

where $c_j^{s_i}$ is j-th concept of i-th label (s represents the current categorization), Num is the number of concept $c_j^{s_i}$ occurrences. $\mathcal{V}$ is the size of the concepts vocabulary. $\mathcal{N}$ is the total number of all concept occurrences.

4.1.3 Classifier with Voting Gate

For emotion understanding task, we respectively calculate neural distribution of ROBERTA and knowledge distribution of MCKB. Lastly, we utilize a voting gate module to vote and integrate these two distributions.

Neural Distribution of Encoder Once the sentence encoding h_s is extracted, we then compute a probability distribution over labels, P_z, by the hidden representation from the classifier token

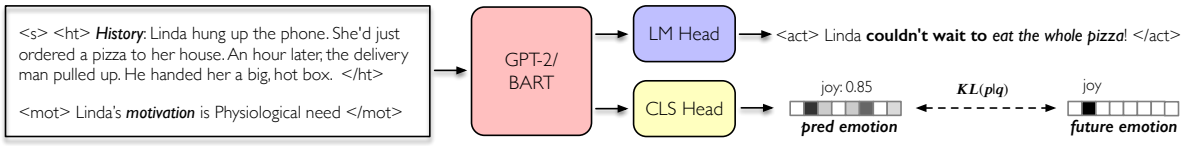

Figure 5: An example and the model architecture (GPT-2 or BART with language model head and emotion prediction head) for conditioned action generation task. Human mental states and key words in action are *italic*. The words expressing future emotion in action are **bolden**.

$h_s \in \mathbb{R}^H$ through an MLP:

$$P_z = W_2 \tanh\left(W_1 h_s + b_1\right) \tag{2}$$

where $W_1 \in \mathbb{R}^{H\times H}$, $b_1 \in \mathbb{R}^H$ and $W_2 \in \mathbb{R}^{N\times H}$, N is the number of labels. The model's predicted answer corresponds to the label of motivations with the highest probability.

Knowledge Distribution of KBs First, we use NLP parsing methods to extract representative commonsense concepts corresponding to the current action. Second, we use each commonsense concept to retrieve the corresponding distribution in concept KBs. In this way, the distribution of all commonsense knowledge $\{P_{c_1}, P_{c_2}, \ldots, P_{c_n}\}$ in the current motivation category is obtained. More details, please refer to Appendix A.3.

Voting Gate As shown in Fig. 4, the specific voting method is as follows:

$$P_f = \mathcal{F}_v(P_z, [P_{c_1}, P_{c_2}, ..., P_{c_n}]) \tag{3}$$

where n is the number of related concepts to action. Among them, $\mathcal{F}_v$ denotes voting ensemble by pooling (such as AVER, MAX, and SUM pooling), multi-layer perceptron (MLP) or gating mechanisms. Finally, the selected label of the largest probability is used as the final prediction result.

4.2 Model for Motivation Understanding

Similar to the model for emotion understanding, we build emotion concept knowledge base (**ECKB**). The difference is the categorization and the dimension of **ECKB**. The remaining modules are the same as Emotion Understanding.

4.3 Model for Conditioned Action Generation

As shown in Fig. 5, for conditioned action generation task, we employ pre-trained transformer (Vaswani et al., 2017) based language models (LM) because of their exceptional performance across related NLG tasks (Forbes et al., 2020; Rudinger et al., 2020; Sakaguchi et al., 2020; Peng et al., 2022b). Specifically, we select two standard text generation models for conditioned action generation task, BART and GPT2. **1. BART** (Lewis et al., 2020) is an encoder-decoder architecture; **2. GPT2** (Radford et al., 2019) is a single "standard" LM. We call the models trained in our settings as **COG-BART** and **COG-GPT2** for performing experiments. Besides, in order to control the direction of action generation is oriented to the future emotion, we adopt a emotion predictor to minimum the distance between given emotion of action and the given emotion label. To teach the model semantic information of the input text, which are motivations and all actions, we design prompt template for action generator (Table 9 in Appendix A.1).

4.4 Training

Emotion Understanding and Motivation Understanding For encoders of these two tasks, we adopt the general MLP classification head and obtain the distribution on each label after fine-tuning with cross-entropy loss:

$$\mathcal{L}_{\text{CLS}} = -\sum_{i=1}^{n} p\left(x_i\right) \log\left(q\left(x_i\right)\right) \tag{4}$$

where x_i represent one sample in EU and MU tasks.

Conditioned Action Generation GPT-2 and BART is trained to learn to produce the action $\mathcal{A}$ of the given history actions $\mathcal{H}$, motivation $\mathcal{M}$ and corresponding character C. To achieve this goal, our approach is trained to maximize the conditional log-likelihood of predicting the object tokens of $\mathcal{A}$:

$$\mathcal{L}_{\text{LM}} = -\sum_{t=|\mathcal{H}|+|\mathcal{M}|+|\mathcal{C}|}^{|\mathcal{H}|+|\mathcal{M}|+|\mathcal{C}|+|\mathcal{A}|} \log P\left(x_t \mid x_{<t}\right) \tag{5}$$

What's more, the predicted emotion distribution of emotion predictor is supervised by the given emotion label distribution with KL-divergence.

$$\mathcal{L}_{\text{KL}} = \mathbf{KL}(p(e_i)|q(e_i)) \tag{6}$$

where $p(e_i)$ is the predicted emotion distribution, and $q(e_i)$ is the given emotion label distribution.

To summarize, the total loss is:

$$\mathcal{L} = \lambda_1 \mathcal{L}_{\text{LM}} + \lambda_2 \mathcal{L}_{\text{KL}} \quad (7)$$

where λ_* is the hyper-parameter controlling the proportion of each part.

5 Experimental Setup

5.1 Baselines

We select the following three baselines for emotion understanding and motivation understanding tasks: **1. GRU** (Chung et al., 2014) is a one-layer bi-GRU encodes the input text and concatenates the final time step hidden states from both directions to yield the sentence representation h_s. **2. BERT** (Devlin et al., 2019) is a standard pre-trained language model. We concatenate sentences using specific separator tokens (`[CLS]` and `[SEP]`). Finally, we take the hidden state representation of `[CLS]` in the last layer of BERT as the overall representation h_s of sentence pairs. **3. RoBERTa** (Liu et al., 2019) is a improved robust BERT which shows state-of-the-art results in many NLP tasks. We use the hidden state representation of <s> as the sentence representation h_s.

For conditioned action generation task, we choose GPT2 (Radford et al., 2019) fine-tuned on ROCStories (Mostafazadeh et al., 2016) and **HAIL** as baselines.

5.2 Implement Details

We train baselines and our models on 9k **HAIL** training examples, then select hyper-parameters based on the best performing model on the dev set (2k), and then report results on the test set (2k). We employ GPT2 large (1.5B), BART large (680M) and ROBERTA large (340M) for our model. We implement our methods with HuggingFace[§] (Wolf et al., 2020) PyTorch (Paszke et al., 2019). We use V-100 GPU to run the experiments. More details, refer to Appendix A.4.

5.3 Metrics

Automatic Metrics. We report the micro-averaged precision (**P**), recall (**R**), and **F1** score[¶] for emotion and motivation understanding tasks.

For conditioned action generation task, we adopt three automatic measures to evaluate the generated textual action distribution both on content quality and rationality. We use the following measures: **(1)** Perplexity (**PPL**) as an indicator of fluency. A smaller value is better. **(2) BLEU** (Papineni et al., 2002) score with n is 1, 2, 4. **(3) Rouge** (Li et al., 2016) score with n is 1, 2 or L.

Human Evaluation Metrics. We also conduct a human evaluation of generated action. Crowdworkers are required to evaluate actions on a 0-3 scale (3 being very good) from two different perspectives: (1) content quality to indicate whether the generated action is **fluent and coherent**, and (2) content rationality to assess whether it **follows the given motivations and emotions**.

5.4 Results of Automatic Evaluation

Emotion Understanding. We show results on the test set in Table 2. Our approach, which using prompt template, constructed KBs, ROBERTA and voting module, achieves the highest score of all models. It is interesting that the emotion understanding task is hard for pre-trained language models with only 59.12 F1 score. This task would need more knowledge and reasoning abilities.

Motivation Understanding. As shown in Table 3, we can conclude that our method outperform other models. However, the improvement of our baseline in this task is small. It is possible that motivation understanding task needs the ability of reasoning. All in all, motivation understanding task is challengeable for the state-of-the-art models in natural language understanding tasks.

Conditioned Action Generation. Table 4 shows that our COG-GPT2 and COG-BART outperforms all baselines, indicating that it can serve as good base action generation model. We can conclude that the BLEU-1 score of COG-GPT2 models is the best. For Rouge score, COG-BART model shows best performance. One reason could be that the summarization task is helpful for generating text with larger recall.

Summary. In conclusion, our approach shows better performance than other implemented state-of-the-art models with the relationship of motivations, emotions and actions. The results of all tasks verify the feasibility of **COMMA**.

5.5 Results of Human Evaluation

We also performed manual evaluation for conditioned action prediction. We randomly selected 100 instances from the test set and used the evaluated model to generate actions. In our work, we compare the generated stories in pairs, and each

[§]https://github.com/huggingface/transformers
[¶]https://github.com/scikit-learn/scikit-learn

Models	P	R	F1
GRU (Chung et al., 2014)	36.23	36.76	36.51
$\text{BERT}_{\text{BASE}}^{\dagger}$ (Devlin et al., 2019)	47.63	54.34	49.77
$\text{BERT}_{\text{LARGE}}^{\dagger}$ (Devlin et al., 2019)	53.95	55.23	53.23
$\text{ROBERTA}_{\text{BASE}}^{\dagger}$ (Liu et al., 2019)	51.47	55.64	53.09
$\text{ROBERTA}_{\text{LARGE}}^{\dagger}$ (Liu et al., 2019)	54.36	58.27	55.93
OURS	**56.75**	**60.39**	**59.12**
(1) w/o ROBERTA	54.04	58.77	58.77
(2) w/o Knowledge Base	54.73	58.90	57.62
(3) w/o prompt template	54.95	58.23	56.23
(4) w/o voting module	55.81	59.01	57.54
(5) w/o motivation	38.38	52.88	45.74

Table 2: Results of Emotion Understanding. Our Approach is our proposed model. † means following the experimental settings in papers.

Models	P	R	F1
GRU (Chung et al., 2014)	40.53	40.27	40.89
$\text{BERT}_{\text{BASE}}^{\dagger}$ (Devlin et al., 2019)	60.57	60.80	60.28
$\text{BERT}_{\text{LARGE}}^{\dagger}$ (Devlin et al., 2019)	61.17	61.45	60.96
$\text{ROBERTA}_{\text{BASE}}^{\dagger}$ (Liu et al., 2019)	61.53	61.62	61.06
$\text{ROBERTA}_{\text{LARGE}}^{\dagger}$ (Liu et al., 2019)	63.50	63.97	63.57
OURS	**64.57**	**64.56**	**63.96**
(1) w/o ROBERTA	59.68	59.24	58.77
(2) w/o Knowledge Base	62.34	62.35	62.57
(3) w/o prompt template	63.22	63.76	62.19
(4) w/o voting module	63.65	63.93	62.44
(4) w/o emotion	64.02	63.95	63.42

Table 3: Results of Motivation Understanding. Our Approach is our proposed model. † means following the experimental settings in papers.

pair is evaluated by 3 judges. The last two columns of Table 4 report the average improvements as well as absolute scores for content quality and rationality. We can conclude that models with our designed prompt template and training loss outperform the models pre-trained on story corpus with the language model objective. It is interesting that the content of generated actions is fluent and grammatical, which indicates that GPT2 and BART is good at organize natural language.

6 Analysis and Discussion

6.1 Ablation Study

To analyze the importance of different modules in our baseline models, we perform ablation study on our approach in emotion and motivation understanding. As shown in Table 2 and Table 3, (1) denotes that the semantic representation of ROBERTA is crucial for understanding tasks. Compared (1) and (2), we find that ROBERTA and KBs have the

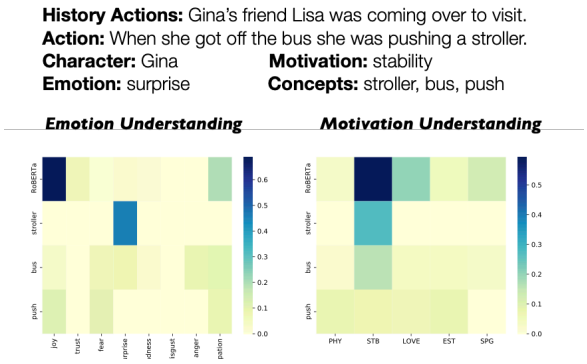

Figure 6: Case study for emotion and motivation understanding. Our framework can give better interpretability.

similar scores in motivation understanding. (3) and (4) indicate the importance of prompt template and voting modules designed for ROBERTA. (5) strength that joint modeling motivation, emotion and action is helpful for emotion and motivation understanding.

6.2 Human A/B Test

Human A/B test is also conducted. We try to directly compare our model with other baselines. We randomly sample 100 examples each for our model and baseline models. Three annotators are given generated responses from either our model or baselines in random order and are asked to choose a better one. They can either choose one of the responses or select "Tie" when the quality of provided options are hard to access. Results in Table 5 confirm that the responses from **COMMA** are more preferred by human judges.

6.3 Case Study

Emotion and Motivation Understanding. Fig. 6 illustrates the distribution of ROBERTA and our concept knowledge base. Our method can bring with better interpretability with the knowledge of key words in human activities. Fig. 6 shows that knowledge base predicts correctly in emotion understanding and help motivation understanding.

Conditioned Action Generation. Since the proposed models can generate actions conditioned on one's motivation, they can be used to unfold action in diverse situations for a combination of history actions, character, motivation, and emotion. We demonstrate this capability in Table 6. $\surd$ means reasonable. 🤦 means that the generated action can not express the corresponding aspect. 🤔 represents that the consistency is debatable. It can be concluded that **motivation and emotion are all**

Models	Automatic Eval							Human Eval	
	PPL	BLEU-1	BLEU-2	BLEU-4	Rouge-1	Rouge-2	Rouge-L	Content	Plausible
GPT2+ROC‡	12.47	17.24	6.26	2.16	6.86	0.23	6.44	2.36	0.79
GPT2+HAIL‡	11.83	16.46	5.81	1.92	7.32	0.38	6.71	2.24	0.56
COG-GPT2	6.85	22.48	**7.54**	**2.85**	10.86	1.05	10.25	2.79	**2.12**
w/o $\mathcal{E}$	7.99	22.29	7.58	2.81	10.22	0.94	9.66	2.85	1.79
w/o $\mathcal{M}$	8.56	21.71	7.12	2.48	10.57	0.96	10.06	2.72	1.63
COG-BART	**6.58**	**24.51**	2.26	0.31	**18.71**	**3.11**	**17.24**	**2.87**	1.98
w/o $\mathcal{E}$	7.65	23.62	2.01	0.22	17.68	2.74	16.25	2.86	1.58
w/o $\mathcal{M}$	8.86	23.98	1.94	0.16	18.53	2.72	16.99	2.79	1.85

Table 4: Automatic and human evaluation results of our COG-GPT2 and COG-BART models on conditioned action generation. COG-GPT2 and COG-BART are trained with the combination of character $\mathcal{C}$, motivation $\mathcal{M}$ and are required to predict emotion $\mathcal{E}$. ‡ represents pre-training GPT2 with language model objective on the corresponding corpus. ROC is ROCStories (Mostafazadeh et al., 2016) and **HAIL** is the train set of our proposed dataset.

Methods	Win	Loss	Tie	κ
COG-GPT2 v.s. GPT2+ROC	**54.2%**	19.5%	26.3%	30.8
COG-GPT2 v.s. GPT2+HAIL	**49.4%**	18.7%	31.9%	29.6
COG-BART v.s. GPT2+ROC	**53.3%**	18.4%	28.3%	28.9
COG-BART v.s. GPT2+HAIL	**54.3%**	14.6%	31.1%	31.3

Table 5: Human A/B Test of **COMMA**. Results show that **COMMA** performs baseline models sufficiently. κ denotes Fleiss' kappa (all are fair agreement or moderate agreement). The p-value of scores < 0.05 in sign test.

Models	Given $\mathcal{H}$ = *Kim and her glass went on a field trip to an aquarium.* $\mathcal{C}$=Kim, $\mathcal{M}$=spirit growth, $\mathcal{E}$=joy	R
GPT2+ROC	*Kim has a very good sense of humor .*	🤦
GPT2+HAIL	*The personality of Kim could be humorous.*	🤦
COG-GPT2	*She was able to get a job at a local restaurant.*	√
w/o $\mathcal{E}$	*She was so excited to go.*	🤔
w/o $\mathcal{M}$	*Kim was a very hard working woman.*	🤦
COG-BART	*Kim had always wanted to go to the beach.*	√
w/o $\mathcal{E}$	*Kim had always wanted to be a pilot.*	√
w/o $\mathcal{M}$	*Kim and her friends decided to go on a date.*	🤔
Human	*Kim enjoyed looking at the sea creatures.*	√

Table 6: Case study of conditioned action generation task for all models, which are tested with history actions, character Kim, motivation spirit growth, emotion joy. Rationality is abbreviated as **R**. The colored text means generated action satisfy the aspects of mental states.

important for action generation. COG-GPT-2 tends to generate short but reasonable actions. But actions generated by COG-BART usually are long but repetitive. From the samples tagged by thinking face, we can see that only motivations or emotions are hard to make action prediction.

Furthermore, we perform some case studies with different inputs. The actions, with the inputs of specific history actions $\mathcal{H}$, character $\mathcal{C}$, motivation $\mathcal{M}$ or emotion $\mathcal{E}$, are generated by COG-GPT-2, COG-BART or human writing. From the first line of Fig. 7, *Jose has a spirit growth need* and *Jose's emotional expectation is joy*. As demonstrated by the three actions, we can conclude that the COG-GPT-2, COG-BART based models can generate reasonable actions. Interestingly, COG-BART can guess that the destination of the trip is Las Vegas, which is competitive to human writing. In the second example, all actions can not clearly express that the emotional expectation is *joy*, where *a big bowl* generated by COG-GPT-2 could somewhat show the happiness of Tom. Last but not least, *Tim was afraid to go outside* and *Tim went to the store to buy a new pair of shoes* are plausible corresponding to the stability need. It is possible that stability need is more abstract for pre-trained language model (PLM) to understand. Besides, we can find that all actions lack the expression of *trust* emotion. One reason is that the PLM based models are insensitive to emotional inputs, which is challengeable in future work.

6.4 Visualization Analysis

To verify the claim **COMMA** *can reveal the essential relationship among motivations, emotions, and actions*, we conduct a visualization analysis of relationships among motivations, actions, and emotions with our approach. Fig. 8 demonstrates the matrix of final prediction probability of motivations and emotions in emotion understanding tasks. The matrix makes motivations, actions and emotions close together and shows that motivations (spiritual growth) have the future emotion (i.e. anticipation($195.8E^{-3}$)). With this matrix, we can better reveal the essential relationship among motivations, emotions, and actions. Therefore, we can

Input	Actions	*C*	*M*	*E*
Character: Jose	*Cog-GPT-2: Jose was very excited to go on a trip.*	✓	✓	✓
Motivation: spirit growth	*Cog-BART: Jose was excited to go on vacation to Las Vegas.*	✓	✓	✓
Emotion: joy	*Human: He book a trip to the Yukon to fulfill his dream.*	✓	✓	✓
Character: Tom	*Cog-GPT-2: Tom was eating a big bowl of cereal.*	✓	✓	🤔
Motivation: physiology	*Cog-BART: Tom went to the grocery store to buy some food.*	✓	✓	🤦
Emotion: joy	*Human: Marc offers to buy Tom skyline chilli if he pays him back.*	✓	✓	🤦
Character: Tim	*Cog-GPT-2: Tim was afraid to go outside.*	✓	🤔	🤦
Motivation: stability	*Cog-BART: Tim went to the store to buy a new pair of shoes.*	✓	🤦	🤦
Emotion: trust	*Human: He freaked out and cried uncontrollably.*	✓	✓	🤔

Figure 7: Case study with different inputs on conditioned action generation task. $\surd$ means reasonable. 🤦 means that the generated action can not express the corresponding aspect. 🤔 represents that the consistency is debatable. The colored text means generated action satisfy the aspects of mental states.

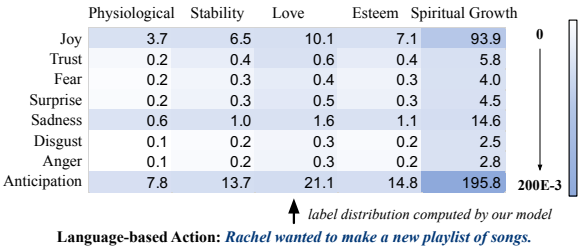

Figure 8: Visualization of final distributions of model. Each element of the above matrix uses scientific notation, and the exponent is -3.

supply more deep explanations about the relationship of motivations, actions, and emotions based on visualization analysis.

7 Related Work

There have been many large-scale language-based resources to explore human mental state, such as motivations and emotions. (Rashkin et al., 2018b; Sap et al., 2019a; Hwang et al., 2020) explore the human mental states in narrative text with series of "if-then" relationships. SOCIAL IQA was introduced by (Sap et al., 2019b) for probing emotional and social intelligence in a variety of everyday situations. Most similar to our work, Rashkin et al. (2018a) put forward Story Commonsense, which is the causal reason for the changes in the psychological state of the characters in the story.

Recently, a lot of work has begun to consider introducing various mental state of human beings into sentiment analysis and other NLP downstream tasks. Li and Hovy (2017) explore the importance of human motivations for sentiment analysis and consider emotion as a specific event or entity that realizes the mental state of human satisfaction with oneself. Otani and Hovy (2019) regard human motivation as the driving force of human emotions, and take motivation detection as the first step of emotion detection, which improves the sentiment analysis of evaluation. (Du et al., 2019; Ammanabrolu et al., 2021; Xu et al., 2020; Brahman and Chaturvedi, 2020) use the knowledge generated by COMET regarded the psychological state of COMET as a condition for story generation.

All in all, the existing language-based resources and works focus on the binary relationship between action and each mental state. Diversely, we first propose a cognitive framework that aims to analyze comprehensive relationships among motivations, emotions and actions in language-based human individual activities.

8 Conclusion and Future work

In this paper, we propose a **Co**gnitive Fra**m**ework of Hu**m**an **A**ctivities (**COMMA**). To verify the effectiveness of our cognitive framework, we introduce three challenging NLP tasks, automatically construct a dataset **HAIL**, and propose the corresponding methods. Experimental results show a better understanding of the relationship among motivations, emotions and actions under our **COMMA** than existing methods. Modeling the relationship among motivations, emotions and actions in human activities can allow researchers to reason the essential causes of human activities from the cognitive perspective and supply reasonable explanations.

In future work, we will explore **COMMA** on various NLP downstream applications, such as intelligent dialogue, controllable text generation and public opinion analysis. Besides, it is interesting to further analyse more complex human activities.

Acknowledgement

We thank all anonymous reviewers for their constructive comments and useful advice. Also thanks for the discussion with Yunpeng Li, Yajing Sun, Yongxiu Xu, Ping Guo, Xinyu Zhang, Yao Dong and Yige Chen. This work is supported by the National Natural Science Foundation of China (No.62006222 and No.U21B2009). Thanks for COLING organizers and the proposed pre-trained language models, data, codes.

Contribution List Yuqiang Xie: Idea, Paper Writing, Coding; Yue Hu: Guiding, Discussion; Wei Peng: Discussion, Coding; Guanqun Bi: Discussion, Paper Polish; Luxi Xing: Review.

Thanks for the hard work and dedication of all team members.

References

Prithviraj Ammanabrolu, W. Cheung, William Broniec, and Mark O. Riedl. 2021. Automated storytelling via causal, commonsense plot ordering. In *AAAI*.

Faeze Brahman and Snigdha Chaturvedi. 2020. Modeling protagonist emotions for emotion-aware storytelling. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing, EMNLP 2020, Online, November 16-20, 2020*, pages 5277–5294. Association for Computational Linguistics.

Michel Cabanac. 2002. What is emotion? *Behavioural Processes*, 60:69–83.

John T. Cacioppo and Wendi L. Gardner. 1999. Emotion. *Annual Review of Psychology*, 50:191–214. PMID: 10074678.

Junyoung Chung, Çaglar Gülçehre, KyungHyun Cho, and Yoshua Bengio. 2014. Empirical evaluation of gated recurrent neural networks on sequence modeling. *CoRR*, abs/1412.3555.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT 2019, Minneapolis, MN, USA, June 2-7, 2019, Volume 1 (Long and Short Papers)*, pages 4171–4186. Association for Computational Linguistics.

Li Du, Xiao Ding, Ting Liu, and Zhongyang Li. 2019. Modeling event background for if-then commonsense reasoning using context-aware variational autoencoder. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing, EMNLP-IJCNLP 2019, Hong Kong, China, November 3-7, 2019*, pages 2682–2691. Association for Computational Linguistics.

Maxwell Forbes, Jena D. Hwang, Vered Shwartz, Maarten Sap, and Yejin Choi. 2020. Social chemistry 101: Learning to reason about social and moral norms. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing, EMNLP 2020, Online, November 16-20, 2020*, pages 653–670. Association for Computational Linguistics.

William L. Hamilton, Kevin Clark, Jure Leskovec, and Dan Jurafsky. 2016. Inducing domain-specific sentiment lexicons from unlabeled corpora. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing, EMNLP 2016, Austin, Texas, USA, November 1-4, 2016*, pages 595–605. The Association for Computational Linguistics.

C. L. Hull. 1974. Essentials of behavior. In *New Haven: Published for the Institute of Human Relations by Yale University Press*.

Jena D. Hwang, Chandra Bhagavatula, Ronan Le Bras, Jeff Da, Keisuke Sakaguchi, Antoine Bosselut, and Yejin Choi. 2020. COMET-ATOMIC 2020: On symbolic and neural commonsense knowledge graphs. *CoRR*, abs/2010.05953.

Jerome Kagan. 2007. What is emotion?: History, measures, and meanings. In *Yale University Press*.

Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Veselin Stoyanov, and Luke Zettlemoyer. 2020. BART: denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, ACL 2020, Online, July 5-10, 2020*, pages 7871–7880. Association for Computational Linguistics.

Jiwei Li, Michel Galley, Chris Brockett, Jianfeng Gao, and Bill Dolan. 2016. A diversity-promoting objective function for neural conversation models. In *NAACL HLT 2016, The 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, San Diego California, USA, June 12-17, 2016*, pages 110–119. The Association for Computational Linguistics.

Jiwei Li and Eduard Hovy. 2017. *Reflections on Sentiment/Opinion Analysis*, pages 41–59. Springer International Publishing, Cham.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Roberta: A robustly optimized BERT pretraining approach. *CoRR*, abs/1907.11692.

Abraham Harold Maslow. 1943. A theory of human motivation. In *Psychological review*.

Saif Mohammad and Peter D. Turney. 2013. Crowdsourcing a word-emotion association lexicon. *Comput. Intell.*, 29(3):436–465.

Nasrin Mostafazadeh, Nathanael Chambers, Xiaodong He, Devi Parikh, Dhruv Batra, Lucy Vanderwende, Pushmeet Kohli, and James F. Allen. 2016. A corpus and cloze evaluation for deeper understanding of commonsense stories. In *NAACL HLT 2016, The 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, San Diego California, USA, June 12-17, 2016*, pages 839–849. The Association for Computational Linguistics.

Yixin Nie, Adina Williams, Emily Dinan, Mohit Bansal, Jason Weston, and Douwe Kiela. 2020. Adversarial NLI: A new benchmark for natural language understanding. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, ACL 2020, Online, July 5-10, 2020*, pages 4885–4901. Association for Computational Linguistics.

Naoki Otani and Eduard H. Hovy. 2019. Toward comprehensive understanding of a sentiment based on human motives. In *Proceedings of the 57th Conference of the Association for Computational Linguistics, ACL 2019, Florence, Italy, July 28- August 2, 2019, Volume 1: Long Papers*, pages 4672–4677. Association for Computational Linguistics.

Jaak Panksepp. 1998. Affective neuroscience: The foundations of human and animal emotions. *Psychology*.

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. Bleu: a method for automatic evaluation of machine translation. In *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics, July 6-12, 2002, Philadelphia, PA, USA*, pages 311–318. ACL.

Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Köpf, Edward Yang, Zachary DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. 2019. Pytorch: An imperative style, high-performance deep learning library. In *Advances in Neural Information Processing Systems 32: Annual Conference on Neural Information Processing Systems 2019, NeurIPS 2019, December 8-14, 2019, Vancouver, BC, Canada*, pages 8024–8035.

Wei Peng, Yue Hu, Yuqiang Xie, Luxi Xing, and Yajing Sun. 2022a. Cogintac: Modeling the relationships between intention, emotion and action in interactive process from cognitive perspective. *CoRR*, abs/2205.03540.

Wei Peng, Yue Hu, Luxi Xing, Yuqiang Xie, Yajing Sun, and Yunpeng Li. 2022b. Control globally, understand locally: A global-to-local hierarchical graph network for emotional support conversation. In *Proceedings of the Thirty-First International Joint Conference on Artificial Intelligence, IJCAI 2022, Vienna, Austria, 23-29 July 2022*, pages 4324–4330. ijcai.org.

Robert Plutchik. 1980. A general psychoevolutionary theory of emotion. In *Theories of emotion*.

Maria Pontiki, Dimitris Galanis, Haris Papageorgiou, Ion Androutsopoulos, Suresh Manandhar, Mohammad Al-Smadi, Mahmoud Al-Ayyoub, Yanyan Zhao, Bing Qin, Orphée De Clercq, Véronique Hoste, Marianna Apidianaki, Xavier Tannier, Natalia V. Loukachevitch, Evgeniy V. Kotelnikov, Núria Bel, Salud María Jiménez Zafra, and Gülsen Eryigit. 2016. Semeval-2016 task 5: Aspect based sentiment analysis. In *Proceedings of the 10th International Workshop on Semantic Evaluation, SemEval@NAACL-HLT 2016, San Diego, CA, USA, June 16-17, 2016*, pages 19–30. The Association for Computer Linguistics.

Alec Radford, Jeff Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. 2019. Language models are unsupervised multitask learners. In *OpenAI Blog*.

Hannah Rashkin, Antoine Bosselut, Maarten Sap, Kevin Knight, and Yejin Choi. 2018a. Modeling naive psychology of characters in simple commonsense stories. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics, ACL 2018, Melbourne, Australia, July 15-20, 2018, Volume 1: Long Papers*, pages 2289–2299. Association for Computational Linguistics.

Hannah Rashkin, Maarten Sap, Emily Allaway, Noah A. Smith, and Yejin Choi. 2018b. Event2mind: Commonsense inference on events, intents, and reactions. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics, ACL 2018, Melbourne, Australia, July 15-20, 2018, Volume 1: Long Papers*, pages 463–473. Association for Computational Linguistics.

Rachel Rudinger, Vered Shwartz, Jena D. Hwang, Chandra Bhagavatula, Maxwell Forbes, Ronan Le Bras, Noah A. Smith, and Yejin Choi. 2020. Thinking like a skeptic: Defeasible inference in natural language. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: Findings, EMNLP 2020, Online Event, 16-20 November 2020*, pages 4661–4675. Association for Computational Linguistics.

Keisuke Sakaguchi, Ronan Le Bras, Chandra Bhagavatula, and Yejin Choi. 2020. Winogrande: An adversarial winograd schema challenge at scale. In *The Thirty-Fourth AAAI Conference on Artificial Intelligence, AAAI 2020, The Thirty-Second Innovative Applications of Artificial Intelligence Conference, IAAI 2020, The Tenth AAAI Symposium on Educational Advances in Artificial Intelligence, EAAI 2020, New York, NY, USA, February 7-12, 2020*, pages 8732–8740. AAAI Press.

Maarten Sap, Ronan Le Bras, Emily Allaway, Chandra Bhagavatula, Nicholas Lourie, Hannah Rashkin, Brendan Roof, Noah A. Smith, and Yejin Choi. 2019a. ATOMIC: an atlas of machine commonsense for if-then reasoning. In *The Thirty-Third AAAI Conference on Artificial Intelligence, AAAI 2019, The Thirty-First Innovative Applications of Artificial Intelligence Conference, IAAI 2019, The Ninth AAAI Symposium on Educational Advances in Artificial Intelligence, EAAI 2019, Honolulu, Hawaii, USA, January 27 - February 1, 2019*, pages 3027–3035. AAAI Press.

Maarten Sap, Hannah Rashkin, Derek Chen, Ronan Le Bras, and Yejin Choi. 2019b. Social iqa: Commonsense reasoning about social interactions. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing, EMNLP-IJCNLP 2019, Hong Kong, China, November 3-7, 2019*, pages 4462–4472. Association for Computational Linguistics.

Rishi Sharma, James Allen, Omid Bakhshandeh, and Nasrin Mostafazadeh. 2018. Tackling the story ending biases in the story cloze test. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics, ACL 2018, Melbourne, Australia, July 15-20, 2018, Volume 2: Short Papers*, pages 752–757. Association for Computational Linguistics.

Tiffany Watt Smith. 2016. The book of human emotions. In *Little, Brown, and Company*.

Richard Socher, Alex Perelygin, Jean Wu, Jason Chuang, Christopher D. Manning, Andrew Y. Ng, and Christopher Potts. 2013. Recursive deep models for semantic compositionality over a sentiment treebank. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing, EMNLP 2013, 18-21 October 2013, Grand Hyatt Seattle, Seattle, Washington, USA, A meeting of SIGDAT, a Special Interest Group of the ACL*, pages 1631–1642. ACL.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, December 4-9, 2017, Long Beach, CA, USA*, pages 5998–6008.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Rémi Louf, Morgan Funtowicz, Joe Davison, Sam Shleifer, Patrick von Platen, Clara Ma, Yacine Jernite, Julien Plu, Canwen Xu, Teven Le Scao, Sylvain Gugger, Mariama Drame, Quentin Lhoest, and Alexander M. Rush. 2020. Transformers: State-of-the-art natural language processing. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations, EMNLP 2020 - Demos, Online, November 16-20, 2020*, pages 38–45. Association for Computational Linguistics.

Peng Xu, Mostofa Patwary, Mohammad Shoeybi, Raul Puri, Pascale Fung, Anima Anandkumar, and Bryan Catanzaro. 2020. MEGATRON-CNTRL: controllable story generation with external knowledge using large-scale language models. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing, EMNLP 2020, Online, November 16-20, 2020*, pages 2831–2845. Association for Computational Linguistics.

Lei Zhang, Shuai Wang, and Bing Liu. 2018. Deep learning for sentiment analysis: A survey. *Wiley Interdiscip. Rev. Data Min. Knowl. Discov.*, 8(4).

Deyu Zhou, Xuan Zhang, Yin Zhou, Quan Zhao, and Xin Geng. 2016. Emotion distribution learning from texts. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing, EMNLP 2016, Austin, Texas, USA, November 1-4, 2016*, pages 638–647. The Association for Computational Linguistics.

A Appendix

A.1 Prompt Template for Input

In emotion understanding task, a model is given history actions $\mathcal{H}$, character $\mathcal{C}$, a label of motivations $\mathcal{M}$ and a textual action A. In motivation understanding task, inputs of model are history actions $\mathcal{H}$, character $\mathcal{C}$, a emotional label $\mathcal{E}$ and a textual action A. In conditoned action generation task, history actions $\mathcal{H}$, character $\mathcal{C}$, a label of motivations $\mathcal{M}$ and a emotional label $\mathcal{E}$ are given to the generator. We design simple prompt templates to expand the semantic information of the motivation and emotion labels, and also indicate the character owning the motivation and emotion. All templates for EU and MU tasks are as the following:

```
C's history actions are __.
C's motivation is __ .
C's action is __.
C's emotion is __.
```

Table 9 shows the template for conditioned action generation. In summary, this technique can enrich the semantic information of the labels and bring the labels with the given character's information. Ablation studies show the effectiveness of prompt template.

A.2 Data Collection

To verify the effectiveness of our cognitive framework, we construct a new dataset **HAIL** (**H**uman **A**ctivities **I**n **L**ife) for the above four tasks by automatically extracting from the existing resource, Story Commonsense(Rashkin et al., 2018a). Story Commonsense dataset manually annotates human motivations and emotions of the event in daily commonsense stories. It is an important resource for studying the causality of motivations, actions, and emotions in language-based individual activities. Note that, the actors of the actions are required to have both motivations and emotion labeling in our collected data **HAIL**. In order to obtain such *(motivation, action, emotion)* samples, we utilize NLTK[*] (a natural language processing toolkit) and design some rules. In all, we extract 13,568 examples in Story Commonsense that meet our requirements. Fig. 11 denotes the data statistics of label distributions in **HAIL**, including motivations and emotions. The label distribution is relatively uniform, which is conducive to the learning of the model.

Data Analysis We perform analysis about the gender bias of open-text actions in HAIL. As is shown Fig. 10, our dataset have a good distribution considering the gender of individual in all actions.

This mechanism ensures that there is a clear and agreed-upon relationship between *needs-action-emotion* in the story, and avoids subjectivity and ambiguity in SCT (Sharma et al., 2018) and certain NLU tasks (Nie et al., 2020).

[*] http://www.nltk.org/

Hyper-parameter	Value
LR	{1e-5, 2e-5}
Batch size	{16, 32, 64}
Gradient norm	1.0
Warm-up	0.1
Max. input length (# subwords)	200
Epochs	{3, 5, 10}

Table 7: Hyper-parameters of models based on BERT and ROBERTA for emotion and motivation understanding tasks.

Hyper-parameter	Value
LR	1e-5
λ_1	1
λ_2	1.5
Batch size	32
Gradient norm	1.0
Warm-up	0.1
Max. input length (# subwords)	200
Max. output length (# subwords)	60
Max # Epochs	30

Table 8: Hyper-parameters of models based on BART and GPT-2 for action prediction task.

A.3 Knowledge Distribution of KBs

The knowledge bases can give the knowledge distribution of the motivation/emotion category according to the commonsense concepts appeared in the action, which corresponds to *Knowledge Distribution*. Specifically, we use tools such as NLTK[†] and Spacy[‡], and then remove stop words and high-frequency words to extract representative commonsense concepts corresponding to the current action. Finally, we use each commonsense concept to retrieve the corresponding distribution in MCKB or ECKB. In this way, the distribution of all commonsense knowledge $\{P_{c_1}, P_{c_2}, \ldots, P_{c_n}\}$ in the current motivation/emotion category is obtained.

Base on the training set of our proposed **HAIL**,

[†] http://www.nltk.org/

[‡] https://spacy.io/

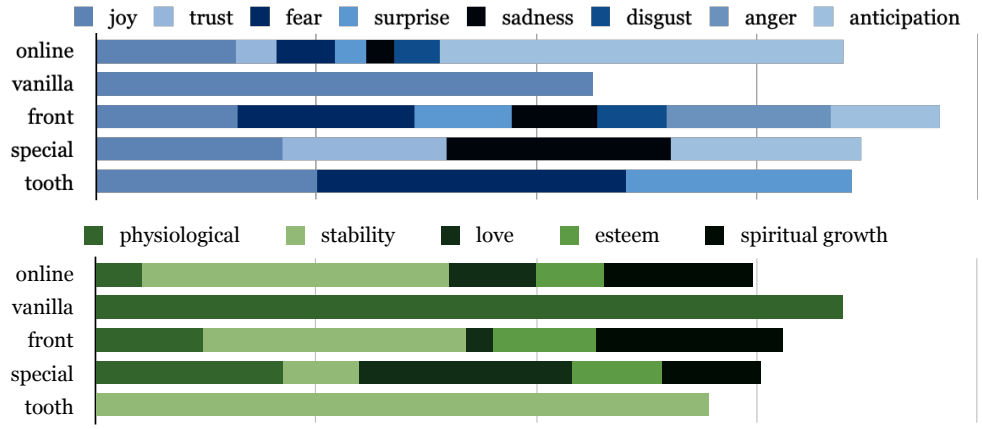

Figure 9: Examples of distribution of motivations Concept KB (below) or Emotion Concept KB (top).

Input Prompt Template	Output Prompt Template
[ht] C's history actions are __ [/ht] and [mot] C has __ motivation [/mot]	[act] __ [/act]

Table 9: In conditioned action generation task, input formats of COG-BART and COG-GPT2.

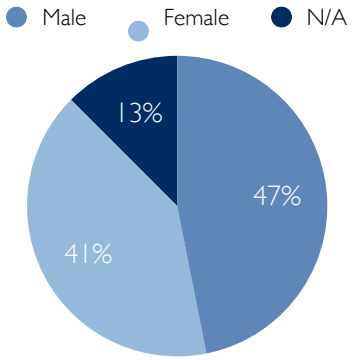

Figure 10: Analysis of gender bias of HAIL.

we automatically construct knowledge bases of motivations and emotions. Examples of them are shown individually in Fig. 9. These two KBs can be used to make prediction or assist the decision-making of the deep model, Moreover, it can also be used to evaluate or explain the forecast results.

A.4 Implement Details

We train baseline models on 9k **HAIL** training examples, then select hyper-parameters based on the best performing model on the dev set (2k), and then report results on the test set (2k). The hyper-parameters of BART and GPT-2 is shown in Table 7. The hyper-parameters of BERT and ROBERTA is shown in Table 8. We use V-100 GPU to run the experiments.

A.5 Comparison of Different Inputs

In order to analyze the effectiveness of our baseline models for action generation, we also perform some case studies with different inputs. The actions, with the inputs of specific history actions $\mathcal{H}$, character $\mathcal{C}$, motivation $\mathcal{M}$ or emotion $\mathcal{E}$, are generated by COG-GPT-2, COG-BART or human writing. From the first line of Fig. 12, *Jose has a spirit growth need* and *Jose's emotional expectation is joy*. As demonstrated by the three actions, we can conclude that the COG-GPT-2, COG-BART based models can generate reasonable actions. Interestingly, COG-BART can guess that the destination of the trip is Las Vegas, which is competitive to human writing. In the second example, all actions can not clearly express that the emotional expectation is *joy*, where *a big bowl* generated by COG-GPT-2 could somewhat show the happiness of Tom. Last but not least, *Tim was afraid to go outside* and *Tim went to the store to buy a new pair of shoes* are plausible corresponding to the stability need. It is possible that stability need is more abstract for pre-trained language model (PLM) to understand. Besides, we can find that all actions lack the expression of *trust* emotion. One reason is that the PLM based models are insensitive to emotional inputs, which is challengeable in future work.

A.6 Future work

Modeling the relationship among motivations, emotions and actions in human activities can allow re-

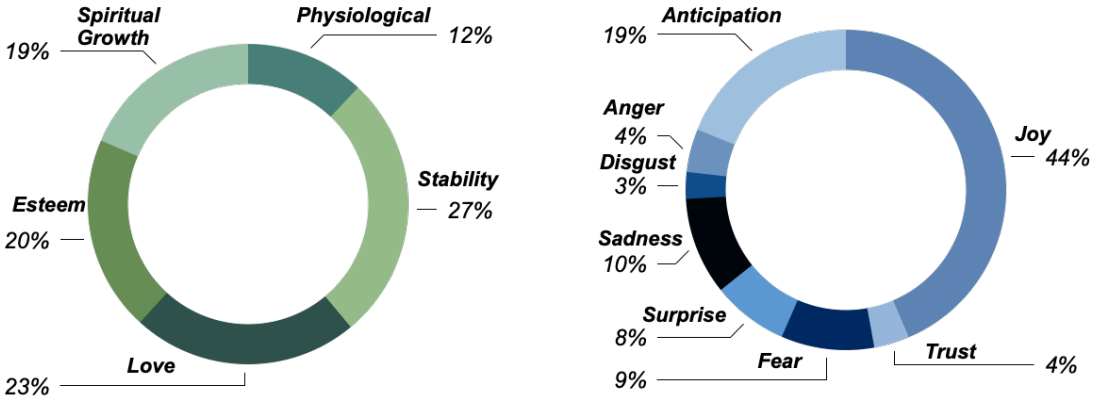

Figure 11: Data statistics of label distributions in **HAIL**, including Human motivations (Left Pic) and Emotion Reactions (Right Pic).

Input	Actions	*C*	*M*	*E*
Character: Jose	*Cog-GPT-2: Jose was very excited to go on a trip.*	√	√	√
Motivation: spirit growth	*Cog-BART: Jose was excited to go on vacation to Las Vegas.*	√	√	√
Emotion: joy	*Human: He book a trip to the Yukon to fulfill his dream.*	√	√	√
Character: Tom	*Cog-GPT-2: Tom was eating a big bowl of cereal.*	√	√	🤔
Motivation: physiology	*Cog-BART: Tom went to the grocery store to buy some food.*	√	√	🤦
Emotion: joy	*Human: Marc offers to buy Tom skyline chilli if he pays him back.*	√	√	🤦
Character: Tim	*Cog-GPT-2: Tim was afraid to go outside.*	√	🤔	🤦
Motivation: stability	*Cog-BART: Tim went to the store to buy a new pair of shoes.*	√	🤦	🤦
Emotion: trust	*Human: He freaked out and cried uncontrollably.*	√	√	🤔

Figure 12: Case study with different inputs on conditioned action generation task. The actions are generated by COG-GPT-2, COG-BART or human writing with the specific character, motivation and emotion. √ means reasonable. 🤦 means that the generated action can not express the corresponding aspect. 🤔 represents that the consistency is debatable. The colored text means generated action satisfy the aspects of mental states.

searchers to reason the essential causes of human activities from the cognitive perspective and supply reasonable explanations. In future work, we will explore **COMMA** on various NLP downstream applications, such as intelligent dialogue, controllable text generation and public opinion analysis. Besides, it is interesting to further analyse more complex human activities.

Exploring Semantic Spaces for Detecting Clustering and Switching in Verbal Fluency

Özge Alaçam[1], Simeon Schüz[1], Martin Wegrzyn[2], Johanna Kißler[2] and Sina Zarrieß[1]
[1]Computational Linguistics, [2]Department of Psychology,
University of Bielefeld, Bielefeld, Germany
{oezge.alacam, simeon.schuez, martin.wegrzyn,
johanna.kissler, sina.zarriess}@uni-bielefeld.de

Abstract

In this work, we explore the fitness of various word/concept representations in analyzing an experimental verbal fluency dataset providing human responses to 10 different category enumeration tasks. Based on human annotations of so-called clusters and switches between sub-categories in the verbal fluency sequences, we analyze whether lexical semantic knowledge represented in word embedding spaces (GloVe, fastText, ConceptNet, BERT) is suitable for detecting these conceptual clusters and switches within and across different categories. Our results indicate that ConceptNet embeddings, a distributional semantics method enriched with taxonomical relations, outperforms other semantic representations by a large margin. Moreover, category-specific analysis suggests that individual thresholds per category are more suited for the analysis of clustering and switching in particular embedding sub-space instead of a one-fits-all cross-category solution. The results point to interesting directions for future work on probing word embedding models on the verbal fluency task.

1 Introduction

The intrinsic evaluation of lexical knowledge represented in word embeddings has been of long-standing interest in distributional semantics (Levy et al., 2015; Hill et al., 2015), and remains an important topic in work on interpreting large-scale black-box language models (Pezzelle et al., 2021; Vulić et al., 2020; Bommasani et al., 2020). While pre-trained contextualized word representations have recently been evaluated in many novel, (psycho-) linguistically motivated probing tasks (Belinkov and Glass, 2019; Ettinger, 2020; Finlayson et al., 2021), the assessment of lexical semantics in word embeddings still commonly focuses on traditional benchmarks of human similarity annotations (Hill et al., 2015), datasets of analogies (Drozd et al., 2016) or taxonomic relations such as hypernymy (Baroni and Lenci, 2011; Glavaš and Vulić, 2018).

In this paper, we conduct an evaluation of word embeddings on the so-called verbal fluency task (Shao et al., 2014), where participants are asked to enumerate as many different words for a given category as possible within a given time interval (often 60 seconds), see Figure 1 for an example response to the category *hobby*. The resulting production data are a rich source of participants' lexical-conceptual knowledge and typically show an interesting clustering-switching pattern where consecutive words either relate to the same sub-category (*handball, football,horseback riding* as a cluster for "sports" in Figure 1) or switch between sub-categories (*playing the guitar, model railway* as a switch from "music" to "playing" in Figure 1). Verbal fluency is a very well-known and widely used cognitive performance test used in psychology, neuro- and psycholinguistics where robust and automatic methods for analyzing clustering-switching patterns would be highly welcome (Kim et al., 2019). Yet, to date, the analysis of verbal fluency data received little attention in research on computational semantics and word embeddings (Pauselli et al., 2018; Linz et al., 2017; Pakhomov and Hemmy, 2014). The study by Linz et al. (2017) constitutes a noticeable exception, but is restricted to verbal fluency responses to a single category ("animals") and does not rely on human annota-

1	Malen *[painting]*	Schaffend *[creating]*
2	Lesen *[reading]*	Lyrik *[lyric]*
3	Schreiben *[writing]*	Lyrik *[lyric]*
4	Handball *[handball]*	Sport *[sport]*
5	Fußball *[football]*	Sport *[sport]*
6	Reiten *[horseback riding]*	Sport *[sport]*
7	Musik *[music]*	Musik *[music]*
8	Gitarre spielen *[playing the guitar]*	Musik *[music]*
9	Modelleisenbahn *[model railway]*	Spielen *[playing]*
10	Modellflugzeug *[model airplane]*	Spielen *[playing]*
11	Sammeln *[collecting]*	Sammeln *[collecting]*
12	Stricken *[knitting]*	Schaffend *[creating]*
13	Sticken *[embroidery]*	Schaffend *[creating]*
14	Nähen *[sewing]*	Schaffend *[creating]*

Figure 1: Elicited word sequence and annotated sub-categories from the hobbies domain

Proceedings of the 29th International Conference on Computational Linguistics, pages 178–191
October 12–17, 2022.

tions of clusters and switches.

We base our study on a recently collected dataset of German semantic verbal fluency responses that provides a much wider range of categories than previous studies and, additionally, has been analyzed in terms of clustering-switching patterns by human judges.[1] Importantly, our verbal fluency-based evaluation of lexical knowledge in embedding spaces rests on human production and judgement data such that it may display different aspects of taxonomical-conceptual knowledge as compared to existing benchmarks for lexical relation prediction derived from standard lexical resources (Baroni and Lenci, 2011; Glavaš and Vulić, 2018). For instance, the sub-categories for the "hobby" category illustrated in Figure 1 could not be easily retrieved from WordNet (Fellbaum, 2010) which, curiously, lists "speleology" as the only direct hyponym for the most common synset of "hobby".

In the following, we compare different word embeddings for German (BERT-base, GloVe, Fasttext, ConceptNet), investigating to what extent distances in sub-spaces for categories like *animals, body parts, clothes* reflect conceptual switches found in verbal fluency data and to what extent it is possible to derive generic, cross-category distance thresholds for the accurate detection of clusters and switches. Our results indicate that ConceptNet embeddings (Speer et al., 2017), a distributional semantics method enriched with taxonomical relations, outperforms other semantic representations by a large margin. On the other hand, merely taxonomical relations (as represented in GermaNet Hamp and Feldweg (1997)) have significant but weak correlations indicating that they are useful as accompanying modalities to word embeddings. Moreover, category-specific analysis suggests that individual thresholds per category are more suited for the analysis of clustering and switching in particular embedding sub-spaces instead of a one-fits-all cross-category solution. A final experiment using simple clustering algorithms further corroborates the findings of the switching analyses. Overall, the results point to interesting directions for future work on probing word embedding models on the verbal fluency task.

[1]The dataset has not been released for ethical reasons, but can be obtained under restricted conditions, upon request.

2 Background

The verbal fluency task is a very well-known neuropsychological test that is used in clinical contexts for diagnosing, e.g., neurodegenerative diseases as well as in research on the cognitive processes underlying lexical knowledge, access, retrieval and executive control (Shao et al., 2014). Participants' verbal fluency responses are typically scored in terms of the number of correct words produced for the category, whereas more fine-grained analyses measure the amount of switching and clustering in the word sequence (Troyer et al., 1997). The semantic analysis of verbal fluency is commonly addressed by the manually defined subcategories for "animals" established in Troyer et al. (1997)'s study, and extending these to other categories and languages is a notorious challenge in psychology (Kim et al., 2019). While there has been a lot of interest in psychology in using word embeddings for scoring semantic fluency (Benigni et al., 2021; Qiu and Johns, 2021; Kim et al., 2019; Paula et al., 2018; Linz et al., 2017), the task entered the radars of the NLP community only recently. One potential reason is that switching and clustering literature are restricted to an extremely limited number of categories (such as *animals*, *groceries*), although there are standardized tools that slightly extend the list of the categories — e.g. the popular RWT (Regensburger Wortflüssigkeitstest) (Aschenbrenner et al., 2000) includes five categories (*animals*, *hobbies*, *occupations*, *groceries* and *first names*).

Quantitative analyses of verbal fluency data have shown that enumeration speed and diversity within a category are very category-dependent and that categories can be more or less easy to enumerate. For automatic methods that measure clustering and switching, a first key step is to define appropriate thresholds that pinpoint switch boundaries for subcategory changes. Kim et al. (2019) investigate different ways of automatically scoring semantic fluency in English and Korean. Using a traditional word2vec model (Mikolov et al., 2013), they predict categorical switches in collected sequences if the predicted similarity between adjacent items drops below a defined threshold, similar to the approach in (Linz et al., 2017). Complementary to this, the authors propose a model which aligns words from fluency sequences with Wikipedia articles and predicts categorical switches when the intersection of articles linked to adjacent words drops below a certain threshold. Despite well-known dif-

ferences between verbal fluency categories, the robustness and quality of these threshold-based methods across categories has, to the best of our knowledge, not yet been analyzed. Moreover, from an NLP perspective, traditional word2vec embeddings can be expected to achieve a lower performance in analyzing fine-grained conceptual relations as compared to various more recent embedding methods that capture global word distributions as in GloVe (Pennington et al., 2014), subword representations as in fastText (Bojanowski et al., 2017) or integrated taxonomical knowledge as ConceptNET Numberbatch (Egozi et al., 2011). Finally, contextualized embeddings from transformer language models such as BERT (Devlin et al., 2019) constitute to be an obvious method to explore. However, to be used in such enumeration task, this dynamic embedding method needs to be transformed into static embeddings as detailed in Section 3.3. In brief, the main contribution of this paper is to explore (i) state-of-the-art word embeddings for data collected in an ongoing verbal fluency study in wide categorical variety and (ii) the options for automatic scoring mechanisms for this broad range of categories. To our knowledge, this is the first (NLP-powered) study that systematically analyzes verbal fluency task across such categorical variety through various semantic representations and evaluation metrics together.

3 Experiments

3.1 Data

In this section, we describe the data collection, annotation and cleaning protocols. Detailed information is provided in Appendix 7.1

Participants. 125 participants originally attended the study, and 114 of them completed it. After cleaning, 100 participants are included in the following analysis (age: 18-63 ($mean$ = 26), $gender$: 87 *female*, 10 *male*, 3 *non-binary*).

Semantic Categories. The initial dataset contains 24 conceptual categories. However, not all of them resulted in sufficient data for statistical analysis. Second, some categories like *amphibians* and *precious stones* elicit a considerable amount of rare words which do not exist in the vocabulary of the methods used here. Furthermore, some categories are very subjective and less related to linguistic lexical knowledge as, e.g., *first names*). Therefore based on qualitative and descriptive analysis, we narrow the 24 categories down to those that have at least 75 words produced by the probands, which are available in all embeddings' vocabulary list, and with a minimum average of 5 words per annotated subcategory. This leaves us with the following 10 categories: *occupations, groceries, hobbies, animals, weapons, vessels, fabrics, countries, clothes, body parts* and *insects*. The entire list can be found in Appendix 7.2.

Subcategory Annotations The words in verbal fluency sequences have been manually annotated with their subcategories (e.g. *pets*, *birds*, *jungle* animals for the *animal* category) by five paid, trained annotators, each annotating 4-5 of the 24 categories. Based on this annotation, we are able to determine *switches* (positions where the left and right word have a different subcategory and *clusters* (sequences of words with the same category).

Data Cleaning. We remove sequences that contain less then 5 items, resulting in 960 sequences in total. The words in the sequences were processed using off-the-shelf NLP text processing tools like; SpaCy Lemmatizer[2], Compound Splitter[3] and Spell Checker[4] for German. Compound words are generally common in German and the vocabulary used by participants also frequently contains compound words such as "Klavierspielen" (piano playing), "Krankenpfleger" (health nurse), "Fahrradfahren" (bike riding). Unfortunately, many of the compounds do not exist in the vocabulary of GloVe, ConceptNet, or GermaNet whereas fastText and BERT embeddings can deal with out-of-vocabulary tokens due to their sub-word tokenization method. In order to address this discrepancy for the non-subword methods, the Python compound splitter package has been used for the words not found in the vocabulary following the lemmatization and spell-check. As a result, the compound vector would be the average of the part vectors.

Table 1 presents basic statistics for word counts and sub-category switch counts observed in the sequences within each category and across categories (as *global*) following the method used by Kim et al. (2019). This overview highlights the differences in the characteristics of the categories: participants enumerated almost 20 items on average for the *animals* and *countries*, and around or below 10 items for *fabrics*, *insects*, and *vessels*. Correspondingly,

[2] https://spacy.io/models/de
[3] https://github.com/dtuggener/CharSplit
[4] https://pypi.org/project/pyspellchecker/

switch counts for *animals* and *hobbies* are significantly higher as compared to categories which are less easy to enumerate.

3.2 Methods

We now introduce our automatic switch detection methods, that we will evaluate on the human subcategory annotations. The goal is to investigate whether it is possible to determine a "one-fits-all" metric that can generalize across various semantic categories and to further explore category-dependent characteristics that cause deviation from the overall pattern.

In addition to comparing human-annotations with the word/concept embedding methods (GloVe, fastText, ConceptNet and BERT-base), we further investigate how mere taxonomic relations (by employing GermaNet) perform on switch detection (i) as a standalone method (Section 4.3) and (ii) as a complementary source of information, combined with embedding-based decisions (Section 4.4).

We utilize several metrics in order to test the correlations between human annotations and embedding representations. First, we compare the the number of switches determined by the human annotators against the aggregated similarity score calculated for the pairs in the sequences (for each method, Section 4.1). We consider several parameters (*mean*, *maximum*, *minimum* and *standard deviation*) for our aggregation method. Since *mean* values exhibit the highest correlation scores, we select them as the suitable scoring metric for reporting.

Next, we try to detect switch boundaries (subcategory changes) in sequences, using the similarity scores between word pairs in the sequences. To decide whether a given word pair marks a subcategory switch, we apply the threshold cut-off methods described in Kim et al. (2019). We test two threshold variations: (i) Median threshold and (ii) 25-Percentile (25P) threshold. A switch boundary is marked where the cosine similarity between two adjacent words falls below the respective threshold.

For the individual embedding methods, the binary threshold-based decisions whether pairs of words mark switch boundaries or belong to the same subcategory are then compared against the human annotations using Cohen Kappas and Chi-square statistics (using the *scipy* package[5], Section 4.2). The median and 25P thresholds are calculated per category as well as globally, by taking all similarity values in the entire data into account. The entire list of calculated thresholds can be found in Appendix 7.3.

We complement our embedding-based analyses with GermaNet (Hamp and Feldweg, 1997), a lexical-semantic network for German that allows for a rule-based, explainable analysis of the switch boundaries. GermaNet groups nouns, verbs, and adjectives into synsets and links these synsets with lexical semantic relations (containing a total of 205K lexical units in 159K synsets). Using the Python API for GermaNet (*germanetpy*[6]), we extract the lexical units and synsets for the word pairs given their category.

We explore the following metrics for scoring similarity between word pairs based on synset relations: (i) shortest path distance, (ii) path-based (PB) similarity and (iii) information content (IC) based similarity (Resnik, 1999; Leacock and Chodorow, 1998). The details of these metrics can be found in the GermaNet website with a source code[7]. Path-based relatedness measures compute the semantic relatedness between two concepts based on the shortest path between two synsets in the hypernym relation. However, quantifying semantic distances merely based on length in the hypernym relation is intuitively not a flawless concept (Jiang and Conrath, 1997). The IC-based metric combines the structural information in the hypernym relation with the word frequencies (GermaNet raw frequency lists). The relatedness of two synsets is measured in terms of the information content of the least common synset that is a hypernym to both synsets. The word frequencies are used to compute the information content, which scores concepts from specific to general. If a very specific synset is compared to a very general one, the relatedness score will be low. This makes IC-based measures more suited for the similarity annotations. The formulas of these measures are available in Gurevych and Niederlich (2005). In the following, we focus on the IC-based metric due to its superior performance on our data. Detailed scores for all three metrics are provided in Appendix 7.5.

[5] https://docs.scipy.org/doc/scipy/reference/stats.html

[6] https://pypi.org/project/germanetpy/

[7] https://github.com/Germanet-sfs/germanetTutorials

Table 1: Basic statistics (Max, min, and average values of sequences and sub-category switches)

Categories	Word Count in a Sequence	Sub-category switch in a sequence	Total Word Count	Subcategory Count
animals	**Max: 30, Min: 5, Mean: 19.11**	Max: 14.0, Min: 1.0, Mean: 7.8	1659	22
body parts	**Max: 31 , Min: 5 , Mean: 18.2**	Max: 15.0, Min: 3.0 , Mean: 8.37	1527	8
clothes	Max: 24, Min: 7, Mean: 16.5	Max: 13.0, Min: 3.0, Mean: 8.14	1434	15
countries	**Max: 36, Min: 10, Mean: 18.5**	Max: 13.0, Min: .0, Mean: 4.6	1752	6
fabrics	Max: 17, Min: 5, Mean: 7.8	Max: 8.0, Min: .0, Mean: 3.1	537	15
groceries	Max: 25, Min: 6, Mean: 16.6	**Max: 16.0, Min: 3.0, Mean: 9.3**,	1520	14
hobbies	Max: 22, Min: 5, Mean: 14.4	**Max: 15.0, Min: .0, Mean: 7.7**	1158	31
insects	Max: 17, Min: 5, Mean: 9.8	Max: 11.0, Min: 2.0, Mean: 6.4	773	14
occupations	Max: 17, Min: 5, Mean: 12.5	Max: 13.0, Min: 3.0, Mean: 8.3	964	19
vessels	Max: 17, Min: 5, Mean: 10.1	Max: 12.0, Min: 1.0, Mean: 5.9	753	9
Global	Max: 36, Min: 5, Mean: 13.9	Max: 14, Min: 0, Mean: 6.8	12077	153

3.3 Semantic Space Representations

As introduced before, we investigate GloVe (1.31M vocab, 300 dimensional)[8] and fastText (65B tokens, 20M vocab, 300 dim.)[9] as general static word representations for German. As a third method, we test the ConceptNET Numberbatch word embeddings, which are enriched by ConceptNet taxonomic relations (594K vocab, 300 dim.) (Speer et al., 2017). Considering the task at hand, those relations might facilitate the enumeration.

Furthermore, we include BERT embeddings as one of the currently most popular models in NLP. Here, one potential challenge is that sequences of words in verbal fluency data differ substantially from the context that these models are trained for. Many layers of linguistic information like syntax or morphology that transformers learn to represent in their latent layers (Tenney et al., 2019) are not instrumental for this task. In this respect, verbal fluency data differs from most existing probing setups which prompt language models with "regular" linguistic inputs. Therefore, we convert contextualized BERT embeddings (2,350M tokens, 31K vocab, 512 dim.) to static word embeddings following the method explained in Bommasani et al. (2020). For this, we sample 20 sentences from the German Wikipedia 2 Corpus[10] for each item in our vocabulary, and compute their vectors using the *dbmdz/bert-base-german-cased* model[11]. After applying a pooling strategy, we end up with a static/single representations for each word.

[8] https://www.deepset.ai/german-word-embeddings

[9] https://fasttext.cc/docs/en/crawl-vectors.html

[10] https://github.com/GermanT5/wikipedia2corpus

[11] https://huggingface.co/dbmdz/bert-base-german-cased

4 Pairwise Switch Analysis

4.1 Switch Count - Similarity Score Correlations

The correlation between the switch counts per sequence (in total 960 sequences) and the mean cosine similarity score of the sequences is analyzed using the Pearson Correlation coefficient (scipy.stats.pearsonr[12]). Sequences with more sub-category switches are expected to have lower (mean) similarity scores. As illustrated in Table 2, the negative correlation is strong for only some of the categories such as *animals*, *countries*, *groceries*, and *insects*. The switch counts for GloVe and BERT embedding methods do not display convincing alignment with the human annotations, whereas fastText and ConceptNet embeddings are more in line with human decisions. This analysis shows that despite some significant and strong correlated categories, especially for the categories *clothes*, *fabrics*, *occupations* and *vessels*, no correlation has been observed indicating that the switch detection methods applied in the following section based on word embeddings might be challenging on these categories.

Table 2: Pearson Correlation Analysis Results on Total Switch Count and Mean Similarity Scores

Categories	GloVe	fastText	ConceptNet	BERT
animals	–.17, n.s.	**–.25, p.<.05**	**–.24, p.<.05**	–.07, n.s.
body parts	–.19, n.s.	.09, n.s.	**.23, p.<.05**	.30, p.<.01
clothes	.03, n.s.	–.147, n.s.	–.14, n.s.	.04, n.s.
countries	**–.43, p.<.01**	**–.39, p.<.01**	**–.40, p.<.01**	.02, n.s.
fabrics	–.11, n.s.	.01, n.s.	–.19, n.s.	.12, n.s.
groceries	**–.34, p.<.01**	**–.24, p. <.05**	**–.27, p.<.01**	–.08, n.s.
hobbies	–.144, n.s.	.03, n.s.	–.17, n.s.	.058, n.s.
insects	–.19, n.s.	**–.38, p.<.01**	**–.27, p.<.01**	–.13, n.s.
occupations	–.00, n.s.	–.16, n.s.	–.08, n.s.	.058, n.s.
vessels	.17, n.s.	–.01, n.s.	–.03, n.s.	.06, n.s.

[12] https://docs.scipy.org/doc/scipy/reference/stats.html

4.2 Embedding-based Switch Detection

As the previous metric returns mixed results, we continue our analysis by turning the continuous similarity scores into discrete switch boundaries.

Overall, switch detection based on 25-Percentile thresholds seems to achieve significant correlations with human annotations (Table 3). While the correlations are weak for GloVe and BERT embeddings, fastText and especially ConceptNet are correlated with the human annotation at various strengths. Categories like *animals*, *hobbies* and *countries* show stronger correlations, while less common categories like *vessels*, *fabrics*, *insects* achieve lower scores.

In addition to this, Table 4 presents the median thresholds, which are more conservative by design. The overall results confirm the category-dependent variations. ConceptNet scores are closer to human scores in 8 of 9 categories, showing strong-to-moderate correlations. Furthermore, the median thresholds seem to be aligned better with the human annotations than the 25-Percentile thresholds for this enumeration task. Similar to previous results, categories like *vessels*, and *fabrics* results in less alignment for this metric as well.

Global threshold vs. category threshold. In order to investigate how a global threshold compares to category-dependent ones, we test the alignment scores of both kinds of thresholds to human-annotations. On average, as illustrated in Figure 2, category-dependent threshold decisions demonstrates slightly better correlations with human annotations.The improvements are particularly evident with regard to some specific configurations: BERT ($avg = .02$, $max = .18$ in insects), ConceptNet ($avg = .01$, $max = .15$ in insects), fastText ($avg = .01$, $max = .11$ in body parts), GloVe ($avg = .02$, $max = .10$ in countries).

4.3 Taxonomy-based Switch Detection

Table 5 presents average shortest path and IC-based similarity scores per category. Based on these scores, median thresholds for each category are calculated following the threshold cut-off method explained in Section 3.2. In detecting sub-category switches, IC-based similarity shows alignment with the human annotations at various levels. The results indicate considerable differences between the categories. For example, whereas high correlations are observed for *hobbies*, *body parts*, *animals*, *occupations* and interestingly *insects*, the correlations are weaker for other categories like *clothes* or *fabrics*.

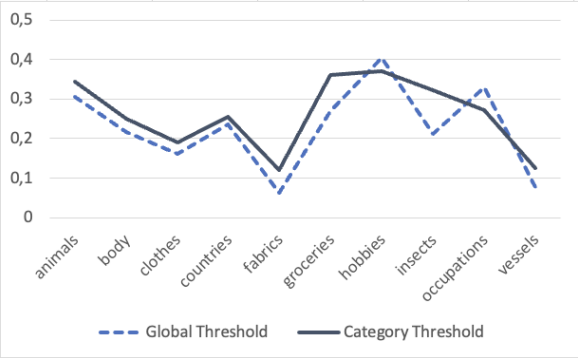

Figure 2: Correlations between the automated scores (Global- vs. Category-Thresholds) and human-annotations — averaged for all methods

4.4 Integrated Switch Detection

To investigate the influence of taxonomic relations in a more explicit way, we combined the decisions on subcategory switch from each embedding model with the decisions calculated using GermaNet IC-based scores (after converting to switch sequences using Median threshold). For each pair in a sequence, the combined metric predicts a switch if there is a switch detected in either of the sequences.

Figure 3 shows the correlation patterns (against human-annotated switch boundaries) across categories for each embedding method with (W+G) and without (W) GermaNet integration. Despite category-based significant differences between W and W+G conditions, it is difficult to conclude on an overall pattern that can explain the variations across categories. However, except ConceptNet ($avg = -.01$, $max = +.07$, higher improvement for *groceries*, *fabrics*), the other representation methods benefited from the inclusion of GermaNet relations especially for some categories; BERT ($avg = +.11$, $max = +.19$, high diff. in *body parts*, *vessels*, *fabrics*, *groceries*), fastText ($avg = +.08$, $max = +.12$, high diff. in fabrics), GloVe ($avg = +.014$, $max = +.23$, high diff. in *vessels*, *insects*, *fabrics*, *groceries*). Thus, the detection of switch boundaries for *vessels* and *fabrics* benefited most from the combined metric, which suggests that embedding models and lexical resources represent complementary aspects of lexical knowledge for these categories. Moreover, GloVe and BERT embeddings benefit more from GermaNet-informed scores (W+G) than fastText and ConceptNet, both for single categories and in the global evaluation (the last item in the graph).

Table 3: Interrater agreement between human annotations and 25-Percentile Thresholds for each embedding

Categories	Human-Annotated Data versus Embeddings' 25-Percentile Threshold			
	BERT	**GloVe**	**ConceptNet**	**fastText**
animals	κ: .07, Corr.: .08, p. <.01	κ: .17, Corr.: .18, p. <.01	**κ: .42, Corr.: .46, p. <.01**	κ: .28, Corr.: .31, p. <.01
body parts	κ: .10 , Corr.: .12 ,p. <.01	κ: .29, Corr.: .32 , p. <.01	**κ: .317, Corr.: .35, p. <.01**	κ: .25, Corr.: .28 , p. <.01
clothes	κ: .07, Corr.: .08, p. <.01	κ: .04, Corr.: .04, n.s.	**κ: .22, Corr.: .26, p. <.01**	κ: .12, Corr.: .14, p. <.01
countries	κ: .10, Corr.: .10, p. <.01	**κ: .43, Corr.: .43, p. <.01**	κ: .39, Corr.: .39, p. <.01	κ: .34, Corr.: .34, p. <.01
fabrics	κ: .11, Corr.: .12, p. <.01	κ: –.08, Corr.: –.09, p. <.05	**κ: .10, Corr.: .12, p. <.01**	κ: .02, Corr.: .02, n.s.
groceries	κ: .13, Corr.: .17, p. <.01	κ: .23, Corr.: .29, p. <.01	**κ: .26, Corr.: .33, p. <.01**	κ: .21, Corr.: .27, p. <.01
hobbies	κ: .15, Corr.: .19, p. <.01	κ: .24, Corr.: .29, p. <.01	**κ: .39, Corr.: .48, p. <.01**	κ: .22, Corr.: .27, p. <.01
insects	κ: .13, Corr.: .20, p. <.01	κ: .08, Corr.: .13, p. <.01	**κ: .15, Corr.: .25, p. <.01**	κ: .11, Corr.: .18, p. <.01
occupations	κ: –.01, Corr.: –.02, n.s.	κ: .09, Corr.: .13, p. <.01	**κ: .17, Corr.: .25, p. <.01**	κ: .15, Corr.: .22, p. <.01
vessels	κ: –.06, Corr.: –.09, p. <.05	κ: .02, Corr.: .02, n.s.	**κ: .12, Corr.: .16, p. <.01**	κ: .08, Corr.: .10, p. <.01

Table 4: Interrater agreement between human annotations and Median Thresholds for each embedding

Categories	Human-Annotated Data versus Embeddings' Median Threshold			
	BERT	**GloVe**	**ConceptNet**	**fastText**
animals	κ: .10, Corr.: .10, p. <.01	κ: .311, Corr.: .31, p. <.01	**κ: .53, Corr.: .54, p. <.01**	κ: .40, Corr.: .40, p. <.01
body parts	κ: –.04 , Corr.: –.037 , n.s.	**κ: .41, Corr.: .41, p. <.01**	κ: .24, Corr.: .24, p. <.01	κ: .38, Corr.: .38, p. <.01
clothes	κ: .07, Corr.: .07, p. <.01	κ: .17, Corr.: .17, p. <.01	**κ: .33, Corr.: .33, p. <.01**	κ: .23, Corr.: .23, p. <.01
countries	κ: .144, Corr.: .164, p. <.01	**κ: .35, Corr.: .39, p. <.01**	κ: .31, Corr.: .36, p. <.01	κ: .30, Corr.: .34, p. <.01
fabrics	**κ: .20, Corr.: .21, p. <.01**	κ: –.08, Corr.: –.08, n.s.	κ: .08, Corr.: .08, n.s.	κ: .10, Corr.: .10, p. <.05
groceries	κ: .17, Corr.: .18, p. <.01	κ: .34, Corr.: .34, p. <.01	κ: .33, Corr.: .34, p. <.01	**κ: .40, Corr.: .41, p. <.01**
hobbies	κ: .17, Corr.: .17, p. <.01	κ: .37, Corr.: .37, p. <.01	**κ: .61, Corr.: .61, p. <.01**	κ: .39, Corr.: .39, p. <.01
insects	κ: .24, Corr.: .27, p. <.01	κ: .23, Corr.: .26, p. <.01	**κ: .40, Corr.: .45, p. <.01**	κ: .29, Corr.: .33, p. <.01
occupations	κ: .03, Corr.: .04, n.s.	κ: .21, Corr.: .23, p. <.01	**κ: .38, Corr.: .40, p. <.01**	κ: .35, Corr.: .37, p. <.01
vessels	κ: –.03, Corr.: –.03, n.s.	κ: .03, Corr.: .03, n.s.	κ: .13, Corr.: .13, p. <.01	**κ: .15, Corr.: .15, p. <.01**

Table 5: (left) GermaNet PB scores, (middle) IC-based relatedness scores, (right) correlation scores between threshold-based sequences and human annotations

Categories	Shortest Path Distance (Mean)	IC-Relatedness Score (Mean)	IC-based Relatedness (with median threshold)
animals	7.11	.27	Corr: –.27, **p.<.01**
body parts	5.99	.21	Corr: –.23, **p.<.01**
clothes	4.08	.31	Corr: –.06, **p.<.01**
countries	2.48	.23	Corr: –.03, n.s.
fabrics	5.48	.30	Corr: –.12, **p.<.05**
groceries	5.38	.26	Corr: –.05, **p.<.01**
hobbies	7.07	.11	Corr: –.44, **p.<.01**
insects	6.37	.37	Corr: –.23, **p.<.01**
occupations	7.61	.20	Corr: –.24, **p.<.01**
vessels	3.46	.26	Corr: –.21, **p.<.01**

ConceptNet and fastText scores with or without GermaNet converge on similar values in the global category.

4.5 Discussion

The results show that automatic prediction of switches aligns best with human annotations when using (i) ConceptNet , (ii) a median-based threshold switch detection and (iii) category-specific thresholds. Strong correlations have been achieved for the *hobbies*, and *animals*, and moderate correlations for the *occupations*, *insects*, *groceries*, *clothes*, *countries* and *body parts* (revisiting Table 4). Performance of automatic switch prediction is worst on *vessels* and *fabrics*. This aligns with the fact that these are the categories resulted with the lowest word counts (see Table 1). The exception here is *insects* with few item produced for this category and subcategory boundaries well represented in the embedding methods. We speculate that this might be due to *insects* occurring in more defined and narrow contexts in the training data, whereas fabrics and vessels may occur in a wider range of contexts. Furthermore, we obtain stronger alignment between automatic prediction and human annotation when taxonomic relations are included via implicit co-learning (e.g. ConceptNet) or explicit integration (e.g. joint metric in Section 4.4). Thus, for this task, taxonomic relations are indispensable and should be part of the automatic scoring mechanisms for better alignment with the human annotations. The selection of threshold methods for defining switch boundaries also plays an important role for getting closer to human annotations. 25-percentile is statistically correlated for almost all categories and representation methods, but overall at weak levels. Decisions based on Median thresholds display stronger alignment, and increase the correlations of all representation methods at a

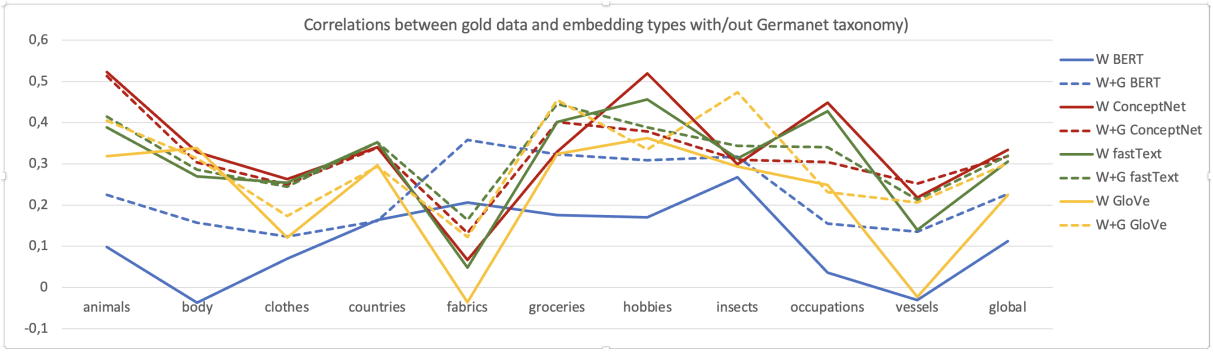

Figure 3: Correlations between the gold annotation and each embedding condition (with/out GermaNet taxonomy.

substantial degree.

The poor results using BERT embeddings could originate from the discrepancy of the task to language modeling, or the lack of training during the conversion from contextualized to static word embeddings. The performance might be improved by increasing the sample size. To train models this large for obtaining static word embeddings seems unreasonably expensive, both computationally and ecologically – especially considering the good performance of simpler approaches enriched with taxonomic relations.

5 Clustering Analysis

The pairwise switch analysis revealed considerable differences between categories as well as favourable results for ConceptNet as a semantic representation. In the following, we report results from an additional clustering-based analysis, in which we take a more global perspective: Instead of investigating adjacent items in fluency sequences, we look at the global semantic organization of all lexical items in the respective categories.

For each category c, we encode the assigned subcategories for individual items into sparse binary vectors. This transformation is necessary for K-Means clustering. This gives us a feature vector with dimensions $(n_{words_c}, n_{subcats_c})$ for human annotations, where n_{words_c} is the number of lexical items and $n_{subcats_c}$ the number of subcategories in the respective category. As simplified, lets assume that we have only three subcategories. A word is represented by the vector [1,0,0] if the word has been assigned to subcategory 1 but not 2 and 3.

We then retrieve the same lexical items from the semantic representations described in Section 3.3, and use each of the feature vectors to fit a *K-Means* clustering model. The k parameter is set depending on the number of annotated subcategories, determined as $k_c = \frac{n_{subcats_c}}{2}$. This parameter reflects the complexity of the domain, i.e., it should allow a large number of clusters for categories with many subcategories. However, there are a few highly fine-grained categories (e.g., fabrics with 28 subcategories for 148 unique words). Therefore, we scaled the value of k for all domains by a constant value.

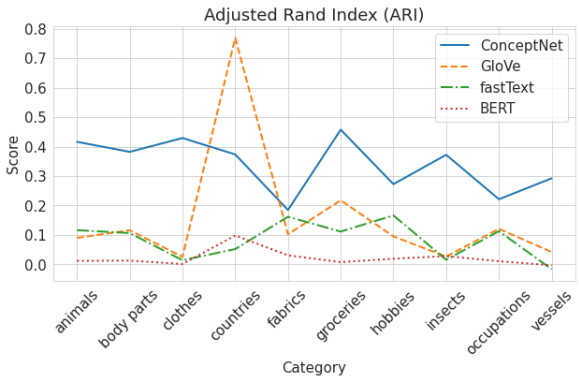

Figure 4: Adjusted Rand Index (ARI) scores for different word embeddings & categories in the fluency data

For evaluation, we rely on the *Adjusted Rand Index (ARI)* (Hubert and Arabie, 1985; Steinley, 2004) for comparing the clusterings based on ConceptNet, GloVe, fasttext and BERT with the results for annotated subcategories. We use the *scikit-learn*[13] library for both clustering and evaluation.

The scores reported in Table 6 and visualized in Figure 4 confirm our previous findings: We see large differences between the categories investigated, with ConceptNet outperforming other semantic representations. One noticeable exception from this is the category *countries*, where GloVe performs surprisingly well. As a possible explanation, we suggest that for different countries the

[13] https://scikit-learn.org/. For the detailed definition of ARI, please visit https://scikit-learn.org/stable/modules/generated/sklearn.metrics.adjusted_rand_score.html

	ConceptNet	GloVe	fastText	BERT
animals	0.42	0.09	0.12	0.01
body parts	0.38	0.12	0.11	0.01
clothes	0.43	0.03	0.01	0.00
countries	0.37	0.77	0.05	0.10
fabrics	0.18	0.10	0.16	0.03
groceries	0.46	0.22	0.11	0.01
hobbies	0.27	0.10	0.17	0.02
insects	0.37	0.03	0.02	0.03
occupations	0.22	0.12	0.11	0.01
vessels	0.29	0.04	-0.02	0.00

Table 6: Adjusted Rand Index (ARI) scores for different word embeddings & categories in the fluency data

textual context might be very informative. As it is based on lexical co-occurrence, this might result in GloVe representations fairly consistent with human categorization.

6 Conclusion

In this paper, we have explored a range of semantic spaces and switch detection methods for the analysis of the verbal fluency data. To the best of our knowledge, this is the first study that (i) incorporates the taxonomic relations using NLP techniques, (ii) explores a wide variety of semantic categories (10 categories), and (iii) explores the fitness of semantic representations in German for this task.

NLP solutions so far are limited to typical/frequent categories like fruits and animals, leaving the annotation of other categories to laborious manual methods. To develop an automatic scoring mechanism, in-depth analysis for less frequent categories is necessary. Our results revealed various category-specific characteristics.

We showed that choosing individual threshold strategies to detect switch boundaries is essential and a "one-fits-all" solution (using a global threshold) results in less aligned sequences. Still, it can be kept as an option since the degradation is not large for the easy to enumerate categories.

In addition to providing an another perspective to analyze the verbal fluency data for psycholinguistic research, this study also prepares the ground for investigating interesting NLP tasks, like subcategory prediction/generation of the upcoming items. From that perspective, Nighojkar et al. (2022) claims that transformer-based language models perform better on cognitive modeling (more specifically, on predicting the next items given a sequence) than the static approaches. However, their evaluation does not include a comparison to knowledge-enriched models. Although their task differs from exploring semantic spaces to detect category switches, category-specific variations are observed from their results regarding 5 categories (fruits, vegetables, animals, supermarket items, tools, and foods).

Furthermore, applying mere taxonomic relations using a synset taxonomy falls behind embedding methods but proved to be instrumental as an accompanying information source, especially for the hard to enumerate categories.

These results highlight that the task is more challenging than it seems, and we need to go beyond out-of-the-box NLP approaches by understanding the nature of these categories and the task. Future studies aim to improve the integrated switch detection method around the taxonomy-enriched representations using additional modalities and knowledge-graph enriched BERT models.

References

Steffen Aschenbrenner, Oliver Tucha, and Klaus W Lange. 2000. *Regensburger Wortflüssigkeits-Test: RWT*. Hogrefe, Verlag für Psychologie.

Marco Baroni and Alessandro Lenci. 2011. How we blessed distributional semantic evaluation. In *Proceedings of the GEMS 2011 Workshop on Geometrical Models of Natural Language Semantics*, pages 1–10.

Yonatan Belinkov and James Glass. 2019. Analysis methods in neural language processing: A survey. *Transactions of the Association for Computational Linguistics*, 7:49–72.

Barbara Benigni, Monica Dallabona, Elena Bravi, Stefano Merler, and Manlio De Domenico. 2021. Navigating concepts in the human mind unravels the latent geometry of its semantic space. *Complexity*, 2021.

Piotr Bojanowski, Edouard Grave, Armand Joulin, and Tomas Mikolov. 2017. Enriching word vectors with subword information. *Transactions of the Association for Computational Linguistics*, 5:135–146.

Rishi Bommasani, Kelly Davis, and Claire Cardie. 2020. Interpreting Pretrained Contextualized Representations via Reductions to Static Embeddings. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 4758–4781, Online. Association for Computational Linguistics.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for*

Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers), pages 4171–4186, Minneapolis, Minnesota. Association for Computational Linguistics.

Aleksandr Drozd, Anna Gladkova, and Satoshi Matsuoka. 2016. Word embeddings, analogies, and machine learning: Beyond king - man + woman = queen. In *Proceedings of COLING 2016, the 26th International Conference on Computational Linguistics: Technical Papers*, pages 3519–3530, Osaka, Japan. The COLING 2016 Organizing Committee.

Ofer Egozi, Shaul Markovitch, and Evgeniy Gabrilovich. 2011. Concept-based information retrieval using explicit semantic analysis. *ACM Transactions on Information Systems (TOIS)*, 29(2):1–34.

Allyson Ettinger. 2020. What bert is not: Lessons from a new suite of psycholinguistic diagnostics for language models. *Transactions of the Association for Computational Linguistics*, 8:34–48.

Christiane Fellbaum. 2010. WordNet. In *Theory and Applications of Ontology: Computer Applications*, pages 231–243. Springer Netherlands.

Matthew Finlayson, Aaron Mueller, Sebastian Gehrmann, Stuart Shieber, Tal Linzen, and Yonatan Belinkov. 2021. Causal analysis of syntactic agreement mechanisms in neural language models. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 1828–1843, Online. Association for Computational Linguistics.

Goran Glavaš and Ivan Vulić. 2018. Discriminating between lexico-semantic relations with the specialization tensor model. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 2 (Short Papers)*, pages 181–187, New Orleans, Louisiana. Association for Computational Linguistics.

Iryna Gurevych and Hendrik Niederlich. 2005. Computing semantic relatedness of germanet concepts. In *Sprachtechnologie, mobile Kommunikation und linguistische Ressourcen: Proceedings of Workshop" Applications of GermaNet II" at GLDV*, pages 462–474.

Birgit Hamp and Helmut Feldweg. 1997. GermaNet - a lexical-semantic net for German. In *Automatic Information Extraction and Building of Lexical Semantic Resources for NLP Applications*.

Felix Hill, Roi Reichart, and Anna Korhonen. 2015. SimLex-999: Evaluating semantic models with (genuine) similarity estimation. *Computational Linguistics*, 41(4):665–695.

Lawrence Hubert and Phipps Arabie. 1985. Comparing partitions. *Journal of classification*, 2(1):193–218.

Jay J. Jiang and David W. Conrath. 1997. Semantic similarity based on corpus statistics and lexical taxonomy. In *Proceedings of the 10th Research on Computational Linguistics International Conference*, pages 19–33, Taipei, Taiwan. The Association for Computational Linguistics and Chinese Language Processing (ACLCLP).

Najoung Kim, Jung-Ho Kim, Maria K. Wolters, Sarah E. MacPherson, and Jong C. Park. 2019. Automatic scoring of semantic fluency. *Frontiers in Psychology*, 10.

Claudia Leacock and Martin Chodorow. 1998. Combining local context and wordnet similarity for word sense identification. *WordNet: An electronic lexical database*, 49(2):265–283.

Omer Levy, Steffen Remus, Chris Biemann, and Ido Dagan. 2015. Do supervised distributional methods really learn lexical inference relations? In *Proceedings of the 2015 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 970–976.

Nicklas Linz, Johannes Tröger, Jan Alexandersson, and Alexandra Konig. 2017. Using Neural Word Embeddings in the Analysis of the Clinical Semantic Verbal Fluency Task. In *Proceedings of the 12th International Conference on Computational Semantics (IWCS 2017)*, pages 1–7, Montpellier, France).

Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. 2013. Distributed representations of words and phrases and their compositionality. In *Advances in neural information processing systems*, volume 26, pages 3111–3119.

Animesh Nighojkar, Anna Khlyzova, and John Licato. 2022. Cognitive modeling of semantic fluency using transformers. In *Proceedings of the 31th International Joint Conference on Artificial Intelligence: Cognitive Aspects of Knowledge Representation*, Vienna, Austria.

Serguei VS Pakhomov and Laura S Hemmy. 2014. A computational linguistic measure of clustering behavior on semantic verbal fluency task predicts risk of future dementia in the nun study. *Cortex*, 55:97–106.

Felipe Paula, Rodrigo Wilkens, Marco Idiart, and Aline Villavicencio. 2018. Similarity measures for the detection of clinical conditions with verbal fluency tasks. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 2 (Short Papers)*, pages 231–235, New Orleans, Louisiana. Association for Computational Linguistics.

Luca Pauselli, Brooke Halpern, Sean D Cleary, Benson S Ku, Michael A Covington, and Michael T Compton. 2018. Computational linguistic analysis

applied to a semantic fluency task to measure derailment and tangentiality in schizophrenia. *Psychiatry research*, 263:74–79.

Jeffrey Pennington, Richard Socher, and Christopher D Manning. 2014. GloVe: Global vectors for word representation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP*, pages 1532–1543, Doha, Qatar. Association for Computational Linguistics.

Sandro Pezzelle, Ece Takmaz, and Raquel Fernández. 2021. Word representation learning in multimodal pre-trained transformers: An intrinsic evaluation. *Transactions of the Association for Computational Linguistics*, 9:1563–1579.

Mengyang Qiu and Brendan Johns. 2021. A distributional and sensorimotor analysis of noun and verb fluency. *PsyArXiv*.

Philip Resnik. 1999. Semantic similarity in a taxonomy: An information-based measure and its application to problems of ambiguity in natural language. *Journal of artificial intelligence research*, 11:95–130.

Zeshu Shao, Esther Janse, Karina Visser, and Antje S. Meyer. 2014. What do verbal fluency tasks measure? predictors of verbal fluency performance in older adults. *Frontiers in Psychology*, 5:1–10.

Robyn Speer, Joshua Chin, and Catherine Havasi. 2017. Conceptnet 5.5: An open multilingual graph of general knowledge. *Proceedings of the AAAI Conference on Artificial Intelligence*, 31(1).

Douglas Steinley. 2004. Properties of the hubert-arable adjusted rand index. *Psychological methods*, 9(3):386.

Ian Tenney, Dipanjan Das, and Ellie Pavlick. 2019. BERT rediscovers the classical NLP pipeline. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 4593–4601, Florence, Italy. Association for Computational Linguistics.

Angela K Troyer, Morris Moscovitch, and Gordon Winocur. 1997. Clustering and switching as two components of verbal fluency: evidence from younger and older healthy adults. *neuropsychology*, 11(1):138.

Ivan Vulić, Edoardo Maria Ponti, Robert Litschko, Goran Glavaš, and Anna Korhonen. 2020. Probing pretrained language models for lexical semantics. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 7222–7240, Online. Association for Computational Linguistics.

7 Appendix

7.1 Data Collection Details

- The experiment was conducted online using Qualtrics.
- The task is explained in writing with an accompanying example during the instruction. During the test, they are asked to type words given the category. Typos were corrected where necessary.
- Each of the 24 categories was presented on a separate page. The timer (60sec) started immediately upon presentation. The order is randomized for each participant. Participants needed to click through all pages. Empty results for animals, hobbies, and groceries are not expected. As sanity check, no response for these categories is evaluated as failure (10 participants). Two participants were dropped based on the typing-speed-test. Two participants made free association (e.g., switching from mice to cheese) instead of enumerating within the class.
- Annotators are instructed to follow a commonsense approach: e.g., "cow" would get assigned to a subcategory like "farm animal" but not to its biological taxonomy.
- The annotation process was conducted in two levels. First, the entire world list, which was produced in the experiment, was checked. An annotator could assign a word to multiple categories, e.g., "lion" as "cat-like", "savanna/desert" and "zodiac sign"; these ratings were done without the annotators knowing the context in which the word was produced. In the second run, subcategories for words are checked in the context of a participant's concrete answers, e.g., "lion" in the context of "cat" and "panther" would be assigned to "cat-like" but "lion" in the context of "Capricorn" would be assigned to "zodiac sign."
- Participants were mainly students of psychology receiving credit points
- The study was approved by the ethics board of the Universität Bielefeld.

7.2 Categories

As mentioned in 3.1, based on qualitative and descriptive analysis, we narrow the 24 categories down to those (i) that have at least 75 words produced in total, (ii) which are available in all embeddings' vocabulary list, and (iii) with a minimum average of 5 words per annotated subcategory. It should be noted that categories like *first names* or *computer games* are problematic for distributional

semantic methods. This leaves us with 10 categories marked in bold below.

- "Amphibians": "Amphibian",
- **"Animals": "Tiere",**
- "Body parts": "Körperteile",
- **"Clothes": "Kleidungsstücke",**
- **"Countries": "Länder",**
- "Currencies": "Währungen",
- "Dances": "Tänze",
- **"Fabrics": "Stoffe",**
- "First names": "Vornamen',
- "Flowers": "Blumen',
- "Gods of antiquity": "Götter der Antike",
- **"Groceries": "Lebensmittel",**
- **"Hobbies": "Hobbies",**
- **"Insects": "Insekten",**
- "Metals": "Metalle",
- "Mountains": "Berge",
- **"Occupations": "Berufe",**
- "Precious stones": "Edelsteine",
- "Spices": "Gewürze",
- "Trees": "Baume",
- "Tropical fruits": "tropische Früchte",
- **"Vessels": "Behälter",**
- "Weapons": "Waffen",
- "Wines": "Weinsorten"

Table 7: Subcategory Switch Thresholds for GermaNet IC-Path Based Similarity Scores

	Median	25 Percentile
animals	.28	.25
body parts	.22	.05
clothes	.31	.30
countries	.23	.23
fabrics	.30	.17
groceries	.26	.05
hobbies		
insects	.37	.36
occupations	.20	.12
vessels	.26	.26
global	.25	.12

7.3 Subcategory Switch Thresholds

Table 9 presents the subcategory switch thresholds calculated with respect to median and 25-Percentile values for 4 different embedding spaces.

7.4 Mean Cosine Distance Scores

Average cosine distance scores between pairs across categories and approaches are presented in Table 8.

7.5 GermaNet Distance and Similarity Metrics

Shortest path distance (SD) given category: The shortest path calculation starts with finding the most similar synset for each word in the pair given the category. For example, for the pair <monkey, dog> in the *animal* category, first, the most relevant synsets for the word "monkey" and "dog" for the category *animal* are calculated separately. Later, the minimum path distance between these two synsets is measured.

Path-based (PB) relatedness: Unlike the previous metric that returns absolute path distance between synsets, path-based relatedness measures compute the semantic relatedness between two concepts based on the shortest path between two synsets in the hypernym relation. The shortest path length is the minimal number of nodes forming a path between the two synsets in the relation. It is also useful to disambiguate word senses (e.g. mouse as animal or electronic equipment")

IC-based relatedness. This measure is explained in the main paper (Section).

To illustrate, Figure 5 shows one example sequence produced in the *animal* category with

Table 8: Average cosine distance scores between pairs across categories and approaches. (The numbers in bold format indicates highest similarity within the category, while the underscore indicates second highest scores.

	GloVe	**fastText**	**ConceptNet**	**BERT**
animals	0,29	0,48	0,38	**0,72**
body parts	0,42	0,52	0,44	**0,68**
clothes	0,22	0,45	0,50	**0,72**
countries	0,54	0,56	0,41	**0,70**
fabrics	0,27	0,48	0,45	**0,73**
groceries	0,31	0,48	0,43	**0,70**
hobbies	0,32	0,39	0,26	**0,70**
insects	0,18	0,46	0,49	**0,80**
occupations	0,31	0,43	0,30	**0,68**
vessels	0,23	0,45	0,46	**0,70**

Table 9: Subcategory Switch Thresholds for Word Embeddings

	Median				25 Percentile			
	GloVe	**fastText**	**ConceptNet**	**BERT**	**GloVe**	**fastText**	**ConceptNet**	**BERT**
animals	.28	.48	.37	.71	.17	.38	.21	.64
body parts	.32	.54	.45	.67	.46	.44	.31	.62
clothes	.20	.47	.55	.73	.09	.33	.36	.62
countries	.55	.57	.37	.70	.46	.48	.26	.65
fabrics	.24	.46	.44	.74	.13	.37	.28	.66
groceries	.30	.49	.43	.70	.17	.37	.30	.61
hobbies	.34	.41	.22	.68	.20	.27	.08	.62
insects	.16	.47	.47	0.82	.08	.37	.32	.80
occupations	.35	.46	.29	.67	.19	.35	.16	.61
vessels	.24	.46	.45	.68	.14	.38	.32	.62
global	.31	.49	.41	.71	.17	.37	.24	.63

these above-mentioned GermaNet scores. The first method has no normalization, and although it does a reasonable job for the overall sequence, it returns the same value for <cat, dog> and <rat, mouse> pairs. PB metric addresses the normalization issue, still treats these pairs in a same way. On the other hand, with the inclusion of word frequency values obtained from a large corpus, it becomes more sensitive for these pairs while flattening the other differences in the less frequent items. Since word enumeration during a verbal fluency task results in rare and participant-dependent word pair formations as well as stereotypical pairs, exploring various metrics is instrumental for understanding the task dynamics and developing a technique for automatic scoring.

w1	**w2**	**w3**	**w4-w11**	**w11**	**w12**	**w13**	**w14**	**w15**
Affe (monkey)	*Katze (cat)*	*Hund (dog)*	…	*Pelikan (pelican)*	*Fisch (fisch)*	*Hai (shark)*	*Ratte (rat)*	*Maus (mouse)*

	w1-w2	***w2-w3***	***w3-w4***	***…***	***w11-w12***	***w12-w13***	***w13-w14***	***w14-w15***	…
SD:	8	**2**	11	…	*9*	*5*	*8*	***2***	…
PB:	0.77	**0.94**	0.68	…	*0.85*	*0.88*	*0.77*	***0.94***	…
IC:	0.11	**0.30**	0.04	…	*0.25*	*0.25*	*0.25*	***0.41***	…

Figure 5: Shortest Distance, Path-based and IC-based similarity scores using GermaNet.

Table 10: GermaNet shortest path distance and similarity scores between consecutive synsets

Categories	**Shortest Path Distance**	**PB-based Similarity**	**IC-based Similarity**
animals	**Corr: .21, p.<.01**	**Corr: -.30, p.<.01**	**Corr: –.27, p.<.01**
body parts	**Corr: .19, p.<.01**	**Corr: –.19, p.<.01**	**Corr: –.23 , p.<.01**
clothes	Corr: –.04, n.s.	Corr: .02, n.s.	**Corr: –.06, p.<.01**
countries	**Corr: .077, p.<.01**	**Corr: –.06, p.<.01**	Corr: –.025, n.s.
fabrics	Corr: –.04, n.s.	Corr: –.08, n.s.	**Corr: –.12, p.<.05**
groceries	Corr: .01, n.s.	Corr: –.05, n.s.	**Corr: –.05, p.<.01**
hobbies	**Corr: .282, p.<.01**	**Corr: –.296, p.<.01**	**Corr: –.442, p.<.01**
insects	Corr: .010, n.s.	**Corr: –.18, p.<.05**	**Corr: –.23, p.<.01**
occupations	Corr: .11, n.s.	**Corr: –.11, p.<.05**	**Corr: –.24, p.<.01**
vessels	Corr: .09, n.s.	Corr: –.11, n.s.	**Corr: –.21, p.<.01**

Neuro-Symbolic Visual Dialog

Adnen Abdessaied[*], **Mihai Bâce**, **Andreas Bulling**
Institute for Visualization and Interactive Systems (VIS)
University of Stuttgart, Germany
adnen.abdessaied@vis.uni-stuttgart.de
mihai.bace@vis.uni-stuttgart.de
andreas.bulling@vis.uni-stuttgart.de

Abstract

We propose Neuro-Symbolic Visual Dialog (NSVD)[1] —the first method to combine deep learning and symbolic program execution for multi-round visually-grounded reasoning. NSVD significantly outperforms existing purely-connectionist methods on two key challenges inherent to visual dialog: long-distance co-reference resolution as well as vanishing question-answering performance. We demonstrate the latter by proposing a more realistic and stricter evaluation scheme in which we use *predicted* answers for the full dialog history when calculating accuracy. We describe two variants of our model and show that using this new scheme, our best model achieves an accuracy of 99.72% on CLEVR-Dialog —a relative improvement of more than 10% over the state of the art —while only requiring a fraction of training data. Moreover, we demonstrate that our neuro-symbolic models have a higher mean first failure round, are more robust against incomplete dialog histories, and generalise better not only to dialogs that are up to three times longer than those seen during training but also to unseen question types and scenes.

1 Introduction

Modelled after human-human communication, visual dialog involves reasoning about a visual scene through multiple question-answering rounds in natural language (Das et al., 2019). Its multi-round nature gives rise to one of its unresolved key challenges: co-reference resolution (Kottur et al., 2018; Das et al., 2019). That is, as dialogs unfold over time, questions tend to include more and more pronouns, such as "it", "that", and "those" that have to be resolved to the appropriate previously-mentioned entities in the scene. Co-reference resolution is profoundly challenging (Das et al., 2019; Hu et al., 2017), even for models specifically designed for this task (Kottur et al., 2018). Existing models follow a purely connectionist approach and suffer from several limitations: first, they require large amounts of training data, which is prohibitive for most settings. Second, these models are not explainable, making it difficult to troubleshoot their logic when co-references are incorrectly resolved. Finally, current models lack generalisability, in particular for real-world dialogs that include incomplete or inaccurate dialog histories, longer dialogs than those seen during training, or unseen question types. While neuro-symbolic hybrid models have proven effective as a more robust, explainable, and data-efficient alternative, e.g. for VQA (Yi et al., 2018), video QA (Yi et al., 2020), or commonsense reasoning (Arabshahi et al., 2021), they have not yet been explored for visual dialog.

We fill this gap by proposing Neuro-Symbolic Visual Dialog (NSVD) —the first neuro-symbolic method geared towards visual dialog. Our method combines three novel contributions to disentangle vision and language understanding from reasoning: First, it introduces two different program generators: a caption and a question program generator, the former of which induces a program from the caption to initialise the knowledge base of the executor at the beginning of each dialog. Second, a question program generator that predicts a program in each round using not only the current question but also the dialog history. We describe two variants of this generator: one that uses a question encoder to concatenate the caption and question-answer pairs of previous rounds to encode the dialog history, as well as one that stacks them. Third, a symbolic executor with a dynamic knowledge base keeps track of all entities mentioned in the dialog.

NSVD also addresses another limitation of existing models that was "hidden" by the dominant evaluation scheme used so far: vanishing question-answering performance over the course of the dia-

*Corresponding Author.

[1]Project page: https://perceptualui.org/publications/abdessaied22_coling/

Proceedings of the 29th International Conference on Computational Linguistics, pages 192–217
October 12–17, 2022.

log. Adopted from VQA (Antol et al., 2015), the dominant scheme assumes that the model has full access to the dialog history, in particular all *ground truth* answers. We argue that this assumption is overly optimistic and overestimates real-world performance on the visual dialog task. We instead propose a more realistic and stricter evaluation scheme in which prediction in the current round is conditioned on previous *predicted* answers. This scheme better represents real-world dialogs in which communication partners rarely know whether their previous answers were correct or not.

Through extensive experiments on CLEVR-Dialog (Kottur et al., 2019), we show that our models are significantly better at resolving co-references and at maintaining performance over many rounds. Using our stricter evaluation scheme, we still achieve an accuracy of 99.72% while requiring only a fraction of the training data. Our results further suggest that NSVD has a higher mean First Failure Round, is more robust to incomplete dialog histories, and generalises better to dialogs that are up to three times longer than those seen during training as well as to unseen question types and scenes. The contributions of our work are threefold: (1) We introduce the first neuro-symbolic visual dialog model that is more robust again incomplete histories, is significantly better at resolving co-references and at maintaining performance over more rounds on CLEVR-Dialog. (2) We contribute a new Domain Specific Language (DSL) for CLEVR-Dialog that we augment with ground truth caption and question programs. (3) We unveil a fundamental limitation of the dominant evaluation scheme for visual dialog models and propose a more realistic and stricter alternative that better represents real-world dialogs.

2 Related Work

Neuro-Symbolic Models and Reasoning. Many works have used end-to-end connectionist models on the CLEVR dataset (Johnson et al., 2017a) with varying degrees of success (Johnson et al., 2017b; Hu et al., 2017; Perez et al., 2018; Hudson and Manning, 2018). NS-VQA (Yi et al., 2018) was one of the first neuro-symbolic models on CLEVR, achieving a near-perfect test accuracy. Mao et al. proposed NS-CL, a neuro-symbolic VQA network that, in contrast to NS-VQA, learned simply by looking at images and reading question-answer pairs. In parallel, other works explored neuro-symbolic models for mono-modal conversational settings (Williams et al., 2017; Suhr et al., 2018; Arabshahi et al., 2021). Recently, Andreas et al. introduced a method that represents dialog states as a dataflow graph to better deal with co-references. Although a number of works have demonstrated the significant potential of neuro-symbolic methods for mono-modal task-oriented dialog settings or tasks at the intersection of computer vision and natural language processing, we are the first to use them for the multi-modal visual dialog task.

Visual Dialog. Das et al. introduced early visual dialog models that used an encoder-decoder approach to rank a set of possible answers. Others explored explicit reasoning based on the dialog structure (Zheng et al., 2019; Niu et al., 2019; Gan et al., 2019). However, these models focused mainly on the real-world VisDial dataset (Das et al., 2019). Although popular, this dataset is not well-suited to study a key challenge of visual dialog, i.e. co-reference resolution, because it lacks complete annotation of all images and dialogs. Several works have focused on co-reference resolution in videos (Ramanathan et al., 2014; Rohrbach et al., 2017) and 3D data (Kong et al., 2014). Kottur et al. introduced CLEVR-Dialog – a fully-annotated diagnostic dataset for multi-round visual reasoning with a grammar grounded in the CLEVR scene graphs. More recently, Shah et al. introduced models that build on the Memory, Attention, and Composition (MAC) network (Hudson and Manning, 2018). Although their models achieved promising results on CLEVR-Dialog, they are computationally and memory inefficient (Shah et al., 2020). While the neuro-symbolic approach has significant potential to address these shortcomings, it has not yet been explored for visual dialog.

3 Method

Our method consists of four components (see Figure 1): a scene understanding method, a program generator with caption and question encoders and a decoder, and a symbolic program executor with a dynamic knowledge base.

Scene Understanding. We used a pre-trained Mask R-CNN (He et al., 2017) to predict segmentation masks and attributes (colour, shape, material, size) for each entity in the visual scene. We then learned the 3D coordinates of each segment paired with the original image using a ResNet-34 (He

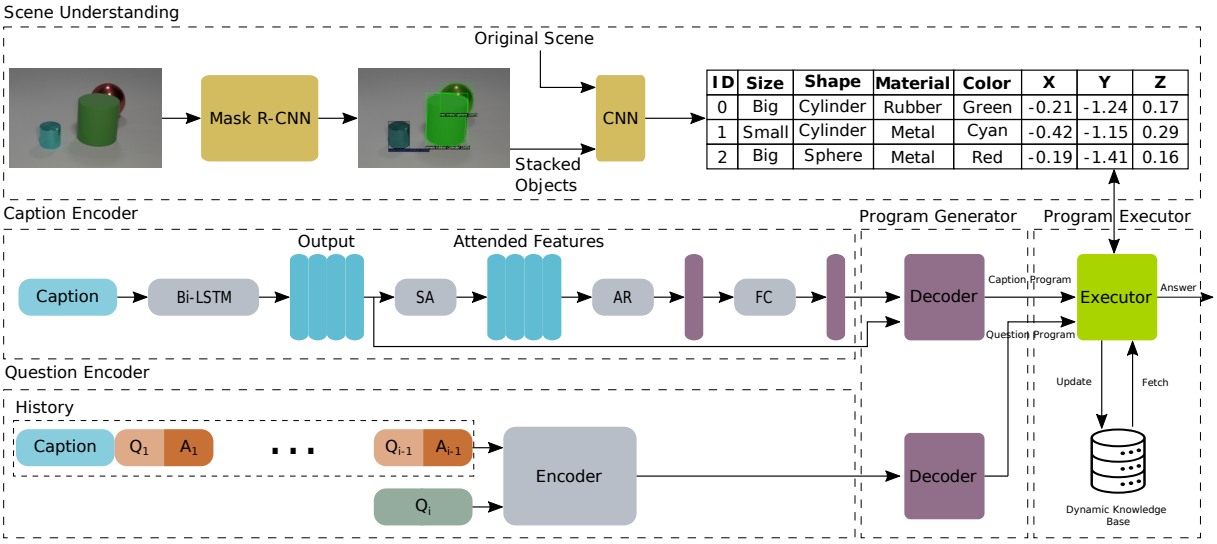

Figure 1: Overview of our method: first, a structured scene representation is created. Then, a caption program is induced and run by our executor to initialise its knowledge base. At each subsequent round, the question and the history are used to induce a program that answers the question and updates the dynamic knowledge base.

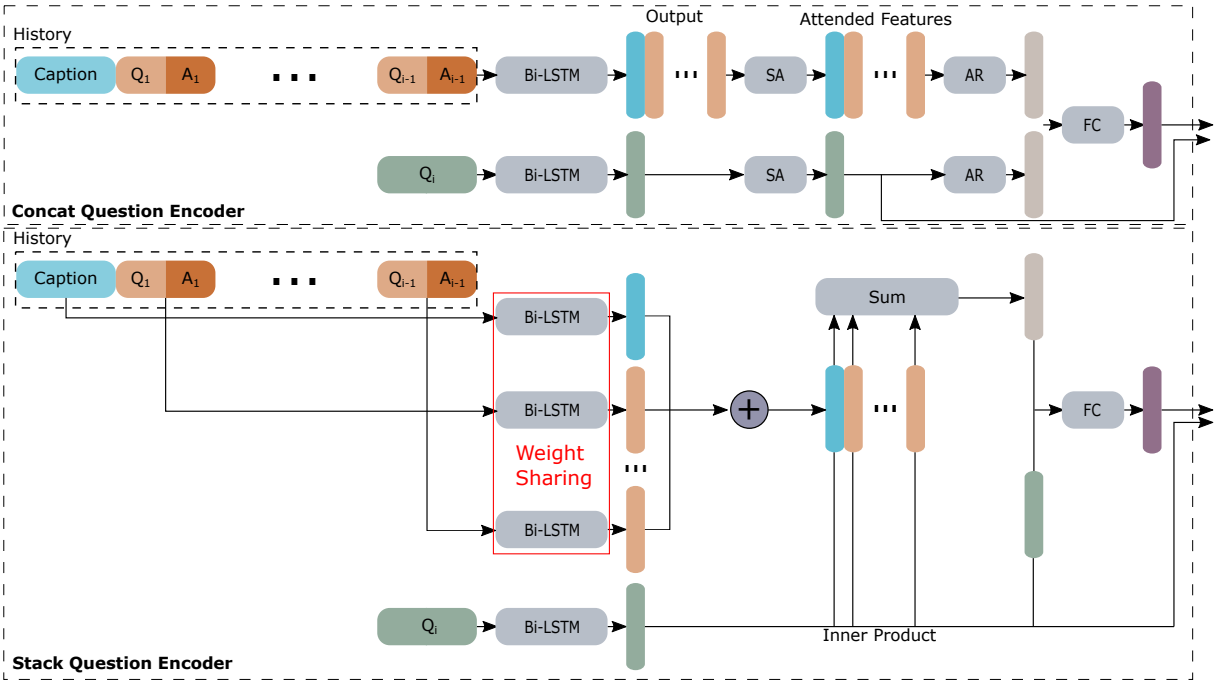

Figure 2: **Top:** The concatenative encoder ("concat") takes the question and the concatenated previous rounds as input and outputs a latent vector and a question representation to the decoder. **Bottom:** The stacking encoder ("stack") takes the question and the previous rounds as input and attends to each round separately. Then, it outputs a latent vector and a question representation to the decoder.

et al., 2016). The Mask R-CNN was pre-trained on the CLEVR-mini dataset (Yi et al., 2018).

DSL for CLEVR-Dialog. Because CLEVR-Dialog implements its own grammar and vocabulary, we designed a novel domain-specific language (DSL) for it by implementing a collection of deterministic functions in Python that our symbolic executor can run over a CLEVR scene. In previous works (Johnson et al., 2017b; Yi et al., 2018; Mao et al., 2019), these functional modules shared the same input/output interface and were arranged one after another to predict the answer. Instead, we followed a stricter approach by executing only one function that expects a *different* number of input arguments to answer a particular question. The full

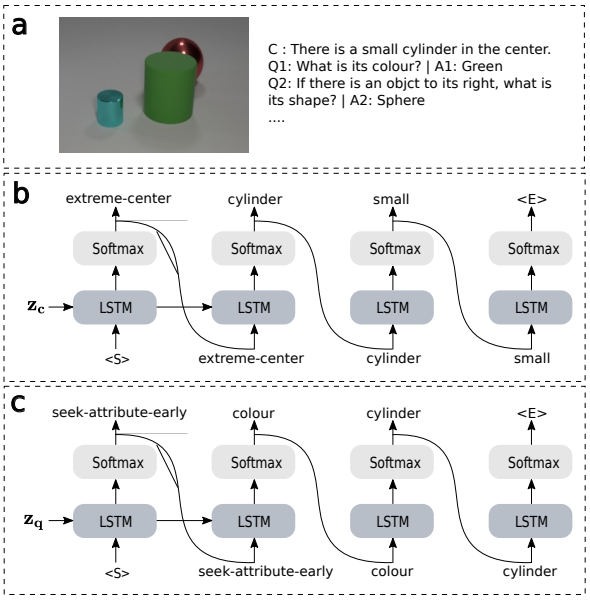

Figure 3: **a:** An example of a CLEVR-Dialog instance with a caption and two rounds. **b:** Inference sample of the caption program generation. **c:** Inference sample of the question program generation on the first round.

list of our functions, their arguments, and expected output can be found in Appendix A.1.

Program Generation. Semantic parsing methods were shown to be effective for mapping sentences to logical forms via a knowledge base or a program (Guu et al., 2017; Liang et al., 2013; Suhr et al., 2018). We adopted this approach and used a sequence-to-sequence model with an encoder-decoder structure to generate the programs. Although we used the same decoder, we propose two types of encoders that differ in the way they encode the dialog history (Figure 1).

Caption Encoder. The caption encoder first embeds the caption tokens into a 300-dim. space to give $\{\mathbf{w_{c_j}}\}_{j=1}^{n_c}$ that are then fed into a bi-directional LSTM (Hochreiter and Schmidhuber, 1997). The self-attended (Vaswani et al., 2017) LSTM outputs $\mathbf{C} = [\mathbf{c_1}, ..., \mathbf{c_{n_c}}]$ are reduced following:

$$\mathbf{a} = \text{softmax}(\text{MLP}(\mathbf{C}))$$
$$\bar{\mathbf{z}}_\mathbf{c} = \sum_{i=1}^{n_c} a_i \mathbf{c_i}.$$

Finally, the latent vector $\mathbf{z_c} = \text{Linear}(\bar{\mathbf{z}}_\mathbf{c})$ and the LSTM output $\mathbf{C}$ are passed to the decoder.

Question Encoders. To generate the question program at the current round i, we use not only the question Q_i but also the history $H_i = [C, Q_1, A_1, .., Q_{i-1}, A_{i-1}]$ of previous question-answer pairs including the caption. We propose two different encoders based on how the question interacts with the history.

– Concat Encoder: The concat encoder is similar in structure to the caption encoder. The caption and the question-answer pairs of previous rounds are concatenated to form the history H_i. Then, the tokens of the current question Q_i and history H_i are embedded into a 300-dim. space to give $\{\mathbf{w}_{\mathbf{q_j}}^{(\mathbf{i})}\}_{j=1}^{n_q}$ and $\{\mathbf{w}_{\mathbf{h_j}}^{(\mathbf{i})}\}_{j=1}^{n_h}$, respectively, which are then processed by two separate bi-directional LSTMs (Figure 2). The reduced attended question and history features $\bar{\mathbf{z}}_\mathbf{q}$ and $\bar{\mathbf{z}}_\mathbf{h}$ are obtained in a similar manner to $\bar{\mathbf{z}}_\mathbf{c}$. Finally, $\bar{\mathbf{z}}_\mathbf{q}$ and $\bar{\mathbf{z}}_\mathbf{h}$ are concatenated and linearly transformed to produce the question latent vector $\mathbf{z_q} = \text{Linear}([\bar{\mathbf{z}}_\mathbf{q}, \bar{\mathbf{z}}_\mathbf{h}])$. Similarly, $\mathbf{z_q}$ and the question LSTM output $\mathbf{Q_i} = [\mathbf{q_1}, ..., \mathbf{q_{n_q}}]$ are passed to the decoder.

– Stack Encoder: The approach of concatenating the question-answer pairs to form the history suffers from two main drawbacks. First, since the history is processed by an LSTM, its encoding becomes inefficient, in particular for later rounds as the LSTM tends to forget crucial information that was mentioned in the first rounds. Second, this approach does not scale well for longer dialogs as it becomes computationally and memory demanding especially because we use self-attention (Vaswani et al., 2017) at a later stage in the concat encoder. To overcome these limitations, we introduce the stack encoder that separately encodes each question-answer pair in order to equally preserve the information from all previous rounds. The question Q_i and each previous round $R_{j<i}$, including the caption, are embedded then processed by separate bidirectional LSTMs (Figure 2). The last hidden states are used as feature representations of the question and previous rounds, i.e.

$$\mathbf{q} = [\overrightarrow{\mathbf{h}_{Q_i}}, \overleftarrow{\mathbf{h}_{Q_i}}] \text{ and } \mathbf{r_{j<i}} = [\overrightarrow{\mathbf{h}_{R_{j<i}}}, \overleftarrow{\mathbf{h}_{R_{j<i}}}],$$

where $\overrightarrow{\mathbf{h}_{(.)}}$ and $\overleftarrow{\mathbf{h}_{(.)}}$ are the bi-directional LSTM's last forward and backward hidden states, respectively. $\bar{\mathbf{z}}_\mathbf{h}$ is obtained by applying an inner-product attention between the question and history features:

$$\mathbf{a} = \text{softmax}(\mathbf{q^T H}),$$
$$\mathbf{H} = [\mathbf{r_0}, ..., \mathbf{r_{i-1}}].$$

Finally, $\mathbf{q}$ and $\bar{\mathbf{z}}_\mathbf{h} = \sum_{j=1}^{i-1} a_j \mathbf{r_j}$ are concatenated and linearly transformed to produce the latent question vector $\mathbf{z_q} = \text{Linear}([\mathbf{q}, \bar{\mathbf{z}}_\mathbf{h}])$. Similarly, $\mathbf{z_q}$

and the question LSTM output $\mathbf{Q} = [\mathbf{q_1}, ..., \mathbf{q_{n_q}}]$ are passed to the decoder.

Decoder. We use the same decoder architecture to generate all the caption as well as the question programs. First, the ground truth program sequence Y_i of the i-th dialog round is embedded into a 300-dim. space to give $\{\mathbf{w}_{\mathbf{y_j}}^{(\mathbf{i})}\}_{j=1}^{n_y}$ which are then processed by a simple LSTM whose hidden states are initialised by the encoder latent vector, i.e. $\mathbf{z_c}$ or $\mathbf{z_q}$. The output $\mathbf{P}$ of the LSTM is used with the encoder output, i.e. $\mathbf{C}$ or $\mathbf{Q}$, to generate a context vector $\mathbf{\Delta}$ following:

$$\mathbf{A} = \text{softmax}(\mathbf{Q^T P}),$$
$$\mathbf{\Delta} = \mathbf{A^T Q}.$$

Finally, the context vector $\mathbf{\Delta}$ is concatenated with the program output and the result is mapped to the program vocabulary dimension followed by a softmax function to obtain a distribution for the current program token y_j, i.e.

$$p(y_j|Y_{[1:j-1]}; Q_i, H_i,) \sim \text{softmax}(\text{Linear}(\tanh([\mathbf{P}, \mathbf{\Delta}]))),$$

where $Y_{[1:j-1]}$ is the sequence of previous ground truth program tokens. For training, we follow the teacher forcing strategy by Williams and Zipser. For inference, we first start with the <S> and sequentially generate the next program token until we reach the end token <E>. Figure 3 illustrates an example of the CLEVR-Dialog dataset alongside the generated programs, i.e. the program `extreme-centre(cylinder, small)` was generated from the caption "*There is a small cylinder in the center*" to initialize our executor and its knowledge base and the program `seek-attribute-early(colour, cylinder)` was generated to answer the first question of the dialog "*What is its colour?*". Our full DSL grammar and further concrete examples can be found in the Appendices A.1 and A.8, respectively.

Executor. We add a dynamic knowledge base to the symbolic executor to keep track of the previously-mentioned entities in the dialog. It is initialised at the beginning of each dialog by executing the caption program. For instance, by executing the caption program `extreme-centre(cylinder, small)`, the executor searches for the centre entity satisfying the function's arguments and stores it in the knowledge base under the handle `small-cylinder`. The executor interacts with its knowledge base via two main operations:

`fetch`. The `fetch`-operation is performed when executing a function that requires co-reference resolution. Given a set of attributes, the executor fetches the appropriate entity in the knowledge base by searching the stored handles. For example `seek-attribute-early(colour, cylinder)` first searches the previously stored handles and fetches the corresponding entity (in our example that is the cylinder mentioned in the caption with the handle `small-cylinder`) and then queries its colour to answer the question.

`update`. The `update`-operation is performed after each question function. We differentiate between four update types:

1. *Handle update:* If a fetched entity is referenced by a new attribute, its handle in the knowledge base should be updated accordingly. If the colour of the previous cylinder is red, then its handle changes from `small-cylinder` to `small-cylinder-red`.

2. *Conversation subject update:* If the question program addresses a new entity, the latter becomes the new conversation subject. In our example, the conversation subject is still the small red cylinder. However, the question program `exist-obj-exclude-early(colour, small, cylinder)` searches for other potential entities that share the same colour as the previous small cylinder. If there is one, it becomes the new conversation subject.

3. *Seen entities update:* Each time a new entity is addressed, the executor saves it in its knowledge base together with the appropriate handle.

4. *Groups update:* Some questions refer to a group of entities, e.g. `count-attribute(red)` counts all red entities in the scene. These sets might be relevant for subsequent questions, e.g. `count-attribute-group(large)` counts how many of the previous red entities are large.

4 Experiments

We modified the publicly available code for CLEVR-Dialog (Kottur et al., 2019) to generate datasets with ground truth caption and question programs required to train our program generators. Similar to (Kottur et al., 2019), we used the $70,000$

training and $15,000$ validation CLEVR images as our visual groundings when generating the dialogs. We left out the CLEVR test images because they lack ground truth scene annotations. For each image, we generated five dialogs each consisting of $L = 10$ question-answer rounds as in (Kottur et al., 2019). We used $1,000$ training images and their corresponding dialogs to create a validation set and tested our models and the baselines on the dialogs generated using the CLEVR validation images.

Performance Evaluation. Alongside the answer accuracy, the First Failure Round (FFR), i.e. the number of dialog rounds necessary for a model to make its first mistake, is commonly used to evaluate visual dialog models. Although popular, this metric has one major limitation as it only allows us to compare the performance of models across datasets with the same dialog length but not across datasets with different ones. Thus, we propose the *Normalised First Failure Round* (NFFR) $\in [0, 1]$ as an improvement and use it alongside the answer accuracy to assess the performance of all models. See Appendix A.2 for more details.

History during Evaluation. One key limitation in the way that visual dialog models are currently evaluated is the use of *ground truth* answers when calculating the correctness of an answer in any given round (Kottur et al., 2018; Das et al., 2019; Shah et al., 2020). The problem of this approach is that it leads to overly-optimistic performance that do not reflect the true capabilities of the models in real-world scenarios: in real-world dialogs, full information on which previous answers were correct or not is typically not available. We instead propose to condition the generation of the current answer on all previous *predicted* answers. This new evaluation scheme is geared to the visual dialog task, better represents real-world use, and is stricter. We call these evaluation schemes "Hist. + GT" and "Hist. + Pred.", respectively.

5 Results

Visual Dialog Performance. After validating our implemented logic (see Appendix A.4), we compared the performance of our models with the visual dialog MAC networks introduced by Shah et al. (2020). We limited our comparison to their top three performing models given that these outperformed the previous state of the art (Kottur et al., 2018) by 30% in accuracy. Furthermore, we compared our models to the Hybrid Code Networks (HCN) (Williams et al., 2017) that also operate on symbolic dialog state representation but follow a different approach to parse programs than our generative one. They represent programs as templates in an action space and select the one with the highest probability during inference. This action space might become intractable if the DSL has many functions and arguments.

Model	Hist. + GT		Hist. + Pred.	
	Acc.	NFFR ↑	Acc.	NFFR ↑
MAC-CQ	$97.34^{\ddagger}$	0.92	41.10	0.15
+ CAA	$97.87^{\ddagger}$	0.94	$89.39^{\ddagger}$	0.75
+ MTM	$97.58^{\ddagger}$	0.92	$70.39^{\ddagger}$	0.46
HCN	75.88	0.34	$74.42^{\ddagger}$	0.32
NSVD-concat	$99.59^{\ddagger}$	0.98	$99.59^{\ddagger}$	0.98
NSVD-stack	$\mathbf{99.72}^{\ddagger}$	**0.99**	$\mathbf{99.72}^{\ddagger}$	**0.99**

Table 1: Performance comparison of our models with the state of the art on CLEVR-Dialog *test*. Results are shown for both "Hist. + GT" and "Hist. + Pred." Our proposed models are highlighted in grey; best performance is in bold. ‡ represents $p < 0.00001$ compared to the second best score in the respective column.

Table 1 shows the performance of our models and the baselines on the test split using both evaluation schemes ("Hist. + GT" and "Hist. + Pred."). As the table shows, our models achieve new state-of-the-art performance with *NSVD-stack* topping with an overall accuracy of 99.72% and a NFFR $= 0.99$. The high NFFR demonstrates our models' ability to answer correctly across all rounds of the dialogs with only few failures in between. More fine-grained evaluations (e.g. on individual rounds, question categories and types) are available in the Appendix A.5. While Table 1 shows results obtained when training on the entire dataset, our method achieves the same performance when trained on only 20% of the data, while the performance of other methods deteriorate significantly with less data as shown in Appendix A.6.

History Length vs Co-reference Distance. CLEVR-Dialog provides co-reference distances for each question, i.e. the number of rounds between the current and earlier mention of an entity in a question. A co-reference distance of 1 means that the co-referent was mentioned in the previous question while a co-reference distance of 10 means that the question at round 10 refers to an entity in the caption. "All" and "None" mean that the question either depends on all previous rounds or is

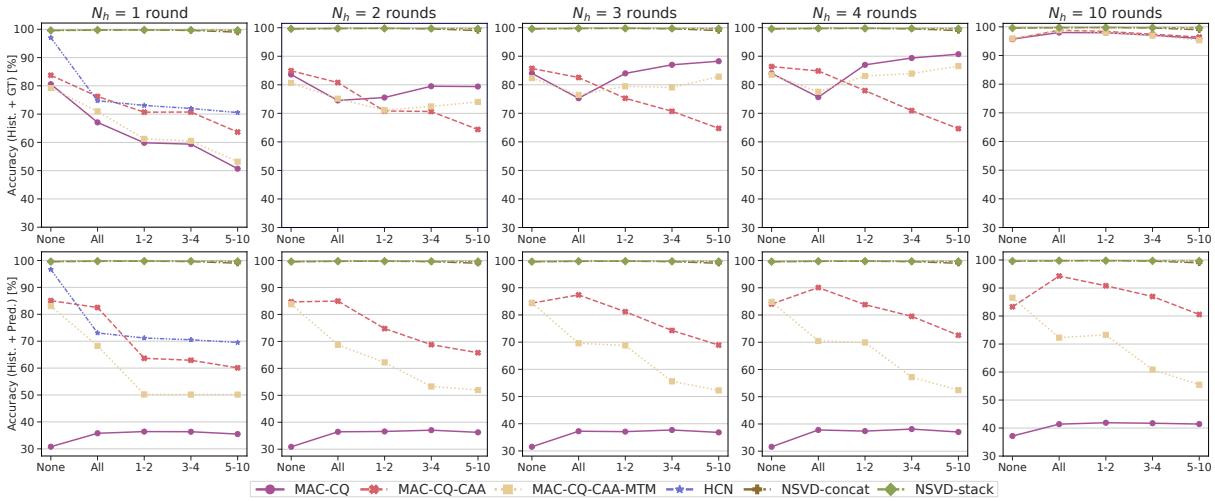

Figure 4: Robustness for different co-reference distance bins and varying number of rounds in the history. All models were trained with full histories. Independent of the evaluation scheme, the performance of our models is only slightly affected by incomplete histories across all bins. In contrast, performance of the baselines deteriorates when the full history is not available, especially for questions with large co-reference distances.

stand-alone, i.e. it does not depend on the history.

To assess performance with respect to the co-reference distance, we evaluated accuracy on different co-reference distance bins. In our evaluations, we further limited the histories to the last N_h question-answer rounds to assess the robustness of the models to incomplete dialog histories. All of the models were trained with histories containing all previous rounds except for HCN (Williams et al., 2017) that only uses the last round.

Our models consistently achieve a performance of over 99% across all co-reference distance bins, independent of the evaluation scheme (Figure 4). Furthermore, their performance is only slightly affected by incomplete dialog histories. Contrarily, performance of all connectionist baselines deteriorates quickly without access to the complete history. This deterioration is more conspicuous when the "Hist. + Pred." evaluation scheme is used (second row of Figure 4). However, their performance is consistent with the difficulty levels of the co-reference distance bins, i.e. the accuracy decreases with increasing co-reference distance. In contrast, this behaviour is not reflected by the popular evaluation approach "Hist. + GT" (first row, middle three plots of Figure 4). The most likely reason for this is that, as is currently common practice when evaluating visual dialog models, the ground truth answers of all previous rounds are used for prediction.

Generalisation to Unseen Scenes and Attributes. In previous experiments, our training and validation sets had similar distributions both in the number of objects (between three and 10) as well their sizes, shapes, colours, and materials. To further test generalisability, we created a new training set consisting of 1500 images in which we restricted the type of objects to small, rubber cubes and spheres with the colours grey, red, or blue. We kept the number of objects in this dataset between three and 10. For testing, we generated three datasets consisting of 1000 images each in which we allowed all CLEVR object classes (cubes, spheres and cylinders) and materials (rubber, metal) to appear. However, we excluded the training colours and increased the number of objects $N_{objects}$ in each one to 10, 15, and 20, respectively. Finally, we generated three fine-tuning datasets containing 1500 images each in a similar way to the testing ones. Figure 5 illustrates some examples of our new images. As in (Kottur et al., 2019), all dialogs had a length of 10 rounds.

We can see that the purely-connectionist models outperform the neuro-symbolic ones without fine-tuning in all scene complexities (Table 2). This outcome is expected since these models rely on a Mask-RCNN to understand the scenes. By increasing their complexities, i.e. more objects and attributes, the Mask-RCNN fails to accurately reconstruct these scenes which is reflected by the poor test accuracies. However, after fine-tuning, neuro-symbolic models are the best performing with our best model *NSVD-stack* scoring 99.33%, 70.62%, and 64.95% on the test datasets with 10,

	Model	$N_{objects} = 10$				$N_{objects} = 15$				$N_{objects} = 20$			
		Without FT		After FT		Without FT		After FT		Without FT		After FT	
		Acc.	NFFR ↑	Acc.	NFFR ↑	Acc.	NFFR ↑	Acc.	NFFR ↑	Acc.	NFFR ↑	Acc.	NFFR ↑
Hist. + GT	MAC-CQ	38.52†	0.14	53.49*	0.21	**36.87***	**0.13**	48.89*	0.19	**36.12***	**0.13**	47.44*	0.18
	+ CAA	37.72‡	0.14	53.35†	0.21	36.82†	0.13	48.67†	0.18	35.52‡	0.13	47.43†	0.18
	+ MTM	**38.59***	**0.14**	52.62	0.21	36.22‡	0.13	47.41	0.17	36.01*	0.13	46.54	0.17
	HCN	19.59	0.11	73.07‡	0.30	14.42	0.11	56.65‡	0.22	12.33	0.11	53.14‡	0.19
	NSVD-concat	25.05*	0.12	99.32‡	0.97	18.51*	0.11	70.59‡	0.44	15.67‡	0.11	64.82‡	0.38
	NSVD-stack	24.95‡	0.12	**99.33***	**0.98**	18.45‡	0.11	**70.62***	**0.44**	15.65*	0.11	**64.95***	**0.38**
Hist. + Pred.	MAC-CQ	**37.74***	**0.14**	52.26†	0.21	36.32*	0.12	47.58*	0.18	35.36‡	0.13	46.30†	0.18
	+ CAA	37.70‡	0.13	51.36†	0.21	**36.76***	0.13	47.08‡	0.18	**35.57***	**0.13**	45.56‡	0.17
	+ MTM	36.29‡	0.14	50.58	0.20	35.62‡	**0.14**	45.67	0.17	33.98‡	0.13	44.02	0.17
	HCN	19.50	0.11	71.55‡	0.29	14.40	0.11	55.55‡	0.21	12.26	0.11	51.95‡	0.19
	NSVD-concat	25.05*	0.12	99.32‡	0.97	18.51*	0.11	70.59‡	0.44	15.67‡	0.11	64.82‡	0.38
	NSVD-stack	24.95‡	0.12	**99.33***	**0.98**	18.45‡	0.11	**70.62***	**0.44**	15.65*	0.11	**64.95***	**0.38**

Table 2: Results when training on simple scenes and testing on more complex ones. Best results in bold. $\ddagger$, $\dagger$, and $*$ represent $p < 0.00001$, $p < 0.05$ and $p \geq 0.05$ compared to the second best score in each column, respectively.

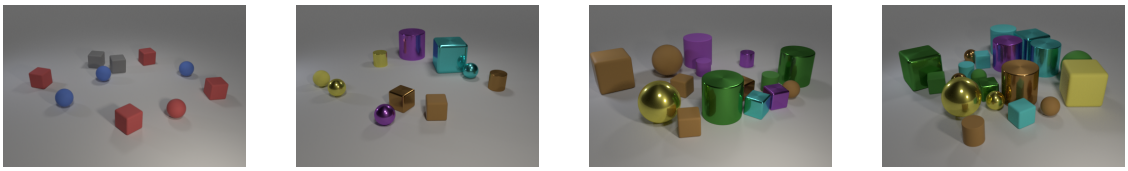

Figure 5: **From left to right:** Samples of a training image, fine-tuning image with 10 objects, fine-tuning image with 15 objects, and fine-tuning image with 20 objects.

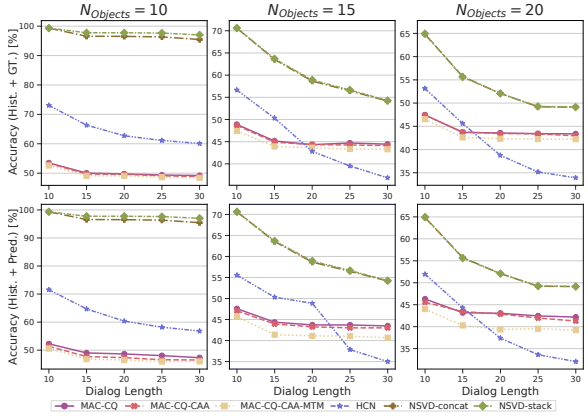

Figure 6: Answer accuracy for different dialog lengths and scene complexities. Our models generalise better to longer dialogs *without* the need for any fine-tuning.

15, and 20 objects, respectively.

Generalisation to Longer Dialogs. Typically, evaluation of visual dialog models is limited to dialogs having the same length as the training data, i.e. $L = 10$ (Kottur et al., 2018; Das et al., 2019; Shah et al., 2020). In order to assess the generalisation capabilities of our models to longer dialogs, we used the testing images of the previous experiment to generate three dialog datasets with increasing numbers of rounds, i.e. $L = 15$, 20, 25, and 30, respectively. That is, our testing datasets for this experiment not only contain dialogs that are up to three times the length of the training dialogs but also visual scenarios never seen during training. Finally, we evaluated the best fine-tuned models of the previous experiment on this data *without* fine-tuning them again on longer dialogs. Figure 6 shows that our models generalise better across all datasets for both evaluation schemes. As expected, the performance of all models decreases with longer dialogs and more complex scenes. However, our models suffer less and still significantly outperform all baselines in all test scenarios. Our best model *NSVD-stack* achieves an overall answer accuracy of 97.02%, 54.25%, and 49.14% on the longest dialogs ($L = 30$) that are created from scenes with 10, 15, and 20 objects, respectively.

Generalisation to Unseen Questions Types. Similar to prior works (Johnson et al., 2017a; Yi et al., 2018; Mao et al., 2019), we addressed the generalisability to new scenes and object combinations in our previous experiments. However, generalisability to unseen questions remains unexplored. To address this, we created two splits (AA and BB) on CLEVR-Dialog as follows: we first split the CLEVR validation images into two disjoint halves A and B. We then split the question types into split A and split B. We randomly assigned half of the question types in each category to split A and the other half to split B to prevent biasing either one to a particular question category. For each image

	Model	Without FT		FT on split BB	
		Acc.	NFFR ↑	Acc.	NFFR ↑
Hist. + GT	MAC-CQ	36.12‡	0.14	40.33†	0.16
	+ CAA	35.09†	0.14	40.36*	0.16
	+ MTM	34.73	0.15	35.09	0.13
	HCN	47.67‡	0.14	70.43‡	0.27
	NSVD-concat	64.07‡	0.24	99.44‡	0.96
	NSVD-stack	**71.55**‡	**0.28**	**99.51**†	**0.97**
Hist. + Pred.	MAC-CQ	35.09‡	0.13	39.53‡	0.15
	+ CAA	36.40‡	0.14	37.72‡	0.15
	+ MTM	6.19	0.09	7.03	0.09
	HCN	46.91‡	0.13	68.82‡	0.25
	NSVD-concat	64.07‡	0.24	99.44‡	0.96
	NSVD-stack	**71.55**‡	**0.28**	**99.51**†	**0.97**

Table 3: Results when training on split AA and testing on split BB. Best results in bold. ‡, †, and ∗ represent $p < 0.00001$, $p < 0.05$ and $p \geq 0.05$ compared to the second best score in each column, respectively.

in both splits, we generated five dialogs consisting of 10 rounds as in (Kottur et al., 2019). Split AA contains a training and a validation set based on $6,000$ and 1500 images, respectively. Split BB has a fine-tuning, a validation, and a test set generated on 2000, 500, and 5000 images, respectively. The desired behaviour for a model that generalises well is to perform well on split BB when only trained on split AA.

NSVD-concat and *NSVD-stack* achieve an accuracy of 64.07% and 71.55%, respectively, when tested on split BB without fine-tuning, thereby significantly outperforming all baselines (Table 3). However, low NFRR values indicate that first failures occur early on in the dialog. After fine-tuning all models on a small amount of data from split BB, our models achieve accuracies and NFRR values comparable to previous experiments. In stark contrast, purely-connectionist baselines' performance only improves by a small margin with the highest jump of 5.26% being achieved by *MAC-CQ-CAA*. This shows an impressive data efficiency of the neuro-symbolic models that, in contrast to the data-hungry purely-connectionist baselines, are able to learn and adapt efficiently from a very small amount of data. More details regarding the training data efficiency can be found in the Appendix A.6.

6 Conclusion and Future Work

We proposed Neuro-Symbolic Visual Dialog (NSVD) —the first hybrid method to combine deep learning and symbolic program execution for multi-round visual reasoning —and a new, stricter and more realistic evaluation scheme for visual dialog. Our method outperforms state-of-the-art purely-connectionist baselines on CLEVR-Dialog and sets a new near-perfect test accuracy of 99.72%. Furthermore, NSVD has a higher mean First Failure Round, is more robust to incomplete dialog histories, and generalises better to longer dialogs and to unseen question types and scenes. These performance improvements are not to be seen in isolation of the stricter supervision our models have as they require a supervised fine-tuning of a Mask-RCNN in addition to a supervised training of the program parsers. Finally, additional evaluations show that our models generalise to other scene domains (Appendix A.7) and we expect them to even generalise to naturalistic datasets as well if they provide the necessary supervision requirements for our models similar to the VQA scenario (Wang et al., 2017; Gan et al., 2017). To the best of our knowledge, such datasets for visual dialog do not yet exist.

Acknowledgments

We would like to thank the anonymous reviewers for their insightful comments, Jiayuan Mao for providing us with the Minecraft data, and Dominike Thomas for paper editing support. M. Bâce was funded by a Swiss National Science Foundation (SNSF) Early Postdoc Mobility Fellowship (No. 199991). A. Bulling was funded by the European Research Council (ERC; grant agreement 801708).

References

Jacob Andreas, John Bufe, David Burkett, Charles Chen, Josh Clausman, Jean Crawford, Kate Crim, Jordan DeLoach, Leah Dorner, Jason Eisner, Hao Fang, Alan Guo, David Hall, Kristin Hayes, Kellie Hill, Diana Ho, Wendy Iwaszuk, Smriti Jha, Dan Klein, Jayant Krishnamurthy, Theo Lanman, Percy Liang, Christopher H. Lin, Ilya Lintsbakh, Andy McGovern, Aleksandr Nisnevich, Adam Pauls, Dmitrij Petters, Brent Read, Dan Roth, Subhro Roy, Jesse Rusak, Beth Short, Div Slomin, Ben Snyder, Stephon Striplin, Yu Su, Zachary Tellman, Sam Thomson, Andrei Vorobev, Izabela Witoszko, Jason Wolfe, Abby Wray, Yuchen Zhang, and Alexander Zotov. 2020. Semantic machines et al. task-oriented dialogue as dataflow synthesis. *Transactions of the Association for Computational Linguistics*, 8:556–571.

Stanislaw Antol, Aishwarya Agrawal, Jiasen Lu, Margaret Mitchell, Dhruv Batra, C Lawrence Zitnick, and Devi Parikh. 2015. VQA: Visual question answering. In *Proceedings of the International Conference on Computer Vision*, pages 2425–2433.

Forough Arabshahi, Jennifer Lee, Mikayla Gawarecki, Kathryn Mazaitis, Amos Azaria, and Tom M.

Mitchell. 2021. Conversational neuro-symbolic commonsense reasoning. In *Proceedings of the Conference on Artificial Intelligence*.

Abhishek Das, Satwik Kottur, Khushi Gupta, Avi Singh, Deshraj Yadav, Stefan Lee, Jose M.F. Moura, Devi Parikh, and Dhruv Batra. 2019. Visual Dialog. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 41(5):1242–1256.

Chuang Gan, Yandong Li, Haoxiang Li, Chen Sun, and Boqing Gong. 2017. VQS: Linking Segmentations to Questions and Answers for Supervised Attention in VQA and Question-Focused Semantic Segmentation. In *Proceedings of the International Conference on Computer Vision*, pages 1829–1838.

Zhe Gan, Yu Cheng, Ahmed Kholy, Linjie Li, Jingjing Liu, and Jianfeng Gao. 2019. Multi-step reasoning via recurrent dual attention for visual dialog. In *Proceedings of the Annual Meeting of the Association for Computational Linguistics*.

Kelvin Guu, Panupong Pasupat, Evan Liu, and Percy Liang. 2017. From language to programs: Bridging reinforcement learning and maximum marginal likelihood. In *Proceedings of the Annual Meeting of the Association for Computational Linguistics*, pages 1051–1062.

Kaiming He, Georgia Gkioxari, Piotr Dollár, and Ross Girshick. 2017. Mask r-cnn. In *Proceedings of the International Conference on Computer Vision*, pages 2961–2969.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. 2016. Deep residual learning for image recognition. In *Proceedings of the Conference on Computer Vision and Pattern Recognition*, pages 770–778.

Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long short-term memory. *Neural Comput.*, 9(8):1735–1780.

Ronghang Hu, Jacob Andreas, Marcus Rohrbach, Trevor Darrell, and Kate Saenko. 2017. Learning to reason: End-to-end module networks for visual question answering. In *Proceedings of the International Conference on Computer Vision*, pages 804–813.

Drew A Hudson and Christopher D Manning. 2018. Compositional attention networks for machine reasoning. In *Proceedings of the International Conference on Learning Representations*.

Justin Johnson, Li Fei-Fei, Bharath Hariharan, C. Lawrence Zitnick, Laurens Van Der Maaten, and Ross Girshick. 2017a. CLEVR: A diagnostic dataset for compositional language and elementary visual reasoning. In *Proceedings of the Conference on Computer Vision and Pattern Recognition*, pages 1988–1997.

Justin Johnson, Bharath Hariharan, Laurens van der Maaten, Judy Hoffman, Li Fei-Fei, C Lawrence Zitnick, and Ross Girshick. 2017b. Inferring and executing programs for visual reasoning. In *Proceedings of the International Conference on Computer Vision*.

Diederik P Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*.

Chen Kong, Dahua Lin, Mohit Bansal, Raquel Urtasun, and Sanja Fidler. 2014. What are you talking about? text-to-image coreference. In *Proceedings of the Conference on Computer Vision and Pattern Recognition*, pages 3558–3565.

Satwik Kottur, José MF Moura, Devi Parikh, Dhruv Batra, and Marcus Rohrbach. 2018. Visual coreference resolution in visual dialog using neural module networks. In *Proceedings of the European Conference on Computer Vision*, pages 153–169.

Satwik Kottur, José M.F. Moura, Devi Parikh, Dhruv Batra, and Marcus Rohrbach. 2019. Clevr-dialog: A diagnostic dataset for multi-round reasoning in visual dialog. In *Proceeding of the Conference of the North American Chapter of the Association for Computational Linguistics*, volume 1, pages 582–595.

Percy Liang, Michael I. Jordan, and Dan Klein. 2013. Learning dependency-based compositional semantics. *Computational Linguistics*, 39(2).

Jiayuan Mao, Chuang Gan, Pushmeet Kohli, Joshua B. Tenenbaum, and Jiajun Wu. 2019. The neuro-symbolic concept learner: Interpreting scenes, words, and sentences from natural supervision. In *Proceedings of the International Conference on Learning Representations*, pages 1–28.

Yulei Niu, Hanwang Zhang, Manli Zhang, Jianhong Zhang, Zhiwu Lu, and Ji-Rong Wen. 2019. Recursive visual attention in visual dialog. In *Proceedings of the Conference on Computer Vision and Pattern Recognition*.

Ethan Perez, Florian Strub, Harm de Vries, Vincent Dumoulin, and Aaron C. Courville. 2018. Film: Visual reasoning with a general conditioning layer. In *Proceedings of the Conference on Artificial Intelligence*.

Vignesh Ramanathan, Armand Joulin, Percy Liang, and Li Fei-Fei. 2014. Linking people in videos with "their" names using coreference resolution. In *Proceedings of the European Conference on Computer Vision*, pages 95–110.

Anna Rohrbach, Marcus Rohrbach, Siyu Tang, Seong Joon Oh, and Bernt Schiele. 2017. Generating descriptions with grounded and co-referenced people. In *Proceedings of the Conference on Computer Vision and Pattern Recognition*, pages 4979–4989.

Muhammad Shah, Shikib Mehri, and Tejas Srinivasan. 2020. Reasoning Over History: Context Aware Visual Dialog. In *Proceedings of the International*

Workshop on Natural Language Processing Beyond Text, pages 75–83. Association for Computational Linguistics.

Alane Suhr, Srinivasan Iyer, and Yoav Artzi. 2018. Learning to map context-dependent sentences to executable formal queries. In *Proceedings of the Annual Conference of the North American Chapter of the Association for Computational Linguistic*, pages 2238–2249.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Ł ukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in Neural Information Processing Systems*.

Peng Wang, Qi Wu, Chunhua Shen, Anthony Dick, and Anton Van Den Hengel. 2017. Explicit knowledge-based reasoning for visual question answering. In *Proceedings of the International Joint Conference on Artificial Intelligence*, pages 1290–1296.

Jason D. Williams, Kavosh Asadi, and Geoffrey Zweig. 2017. Hybrid code networks: Practical and efficient end-to-end dialog control with supervised and reinforcement learning. In *Proceedings of the Annual Meeting of the Association for Computational Linguistics*, volume 1, pages 665–677.

Ronald J Williams and David Zipser. 1989. A learning algorithm for continually running fully recurrent neural networks. *Neural Comput.*, 1(2):270–280.

Jiajun Wu, Joshua B Tenenbaum, and Pushmeet Kohli. 2017. Neural Scene De-rendering. In *Proceedings of the Conference on Computer Vision and Pattern Recognition*.

Kexin Yi, Chuang Gan, Yunzhu Li, Pushmeet Kohli, Jiajun Wu, Antonio Torralba, and Joshua B. Tenenbaum. 2020. CLEVRER: collision events for video representation and reasoning. In *Proceedings of the International Conference on Learning Representations*.

Kexin Yi, Jiajun Wu, Chuang Gan, Antonio Torralba, Pushmeet Kohli, and Joshua B Tenenbaum. 2018. Neural-Symbolic VQA: Disentangling Reasoning from Vision and Language Understanding. In *Advances in Neural Information Processing Systems*.

Zilong Zheng, Wenguan Wang, Siyuan Qi, and Song-Chun Zhu. 2019. Reasoning visual dialogs with structural and partial observations. In *Proceedings of the Conference on Computer Vision and Pattern Recognition*.

A Appendix

A.1 The CLEVR-Dialog DSL

Our Domain Specific Language (DSL) for CLEVR-Dialog is depicted in Table 4. We present each function with its expected argument types, output, and knowledge base operations. The argument types are further defined in Table 5. We use the variables `attr`, `attr_obj_1`, `attr_obj_2` and `attr_i` for `i = 1,..,4` to denote the set of possible CLEVR attributes, i.e. colour, material, shape, and size. Furthermore, the variable `pos` denote one possible position, i.e. `right`, `left`, `front`, or `behind`. Finally, the variable `num` denotes the set of possible numerical values , i.e. between 0 and N, where N is the maximum number of objects in the scene. If not explicitly stated otherwise, we assume $N = 10$.

A.2 Normalised First Failure Round

The Normalised First Failure Round NFFR is calculated as

$$\text{NFFR} = \frac{1}{N}\sum_{i=1}^{N}\frac{1}{L+1}\sum_{j=1}^{L}\Delta_j^{(i)}\delta_{\text{pred}_j^{(i)},\text{gt}_j^{(i)}}\alpha_j^{(i)},$$

where N is the total number of dialogs, L is the length of each dialog, and $\text{pred}_j^{(i)}$ and $\text{gt}_j^{(i)}$ are the predicted and ground truth answers at round j of dialog i, respectively. Furthermore, for each round j of every dialog i, we define $\Delta_j^{(i)}$, $\delta_{\text{pred}_j^{(i)},\text{gt}_j^{(i)}}$, and $\alpha_j^{(i)}$ as:

$$\Delta_j^{(i)} = \begin{cases} j & \text{if} \quad j \leq L \\ L+1 & \text{if} \quad j = L \wedge \delta_{\text{pred}_j^{(i)},\text{gt}_j^{(i)}} = 1 \end{cases},$$

$$\delta_{\text{pred}_j^{(i)},\text{gt}_j^{(i)}} = \begin{cases} 1 & \text{if} \quad \text{pred}_j^{(i)} \neq \text{gt}_j^{(i)} \\ 0 & \text{otherwise} \end{cases},$$

$$\alpha_j^{(i)} = \begin{cases} 0 & \text{if } \exists k < j \text{ s.t. } \delta_{\text{pred}_k^{(i)},\text{gt}_k^{(i)}} = 1 \\ 1 & \text{otherwise} \end{cases}.$$

By definition, we set the NFFR of a model to be $L + 1$ if it correctly answers all L dialog rounds.

A.3 Training Details

Our Models. In order to encode our raw data, we generated two different vocabularies from the training data: one that handles the captions, questions, and answers in form of natural language and another that deals with ground truth caption and question programs. For encoding, we pad all captions, questions, and programs to a maximum length of $n_q = 21$, $n_c = 16$, and $n_y = 6$, respectively. Then, the corresponding tokens are transformed into a 300-dim. space using either the encoder text embedding or the decoder program embedding. We

	Func. Name	Func. Args.	Func. Out.	fetch	update			
					Handle	Conv. Subj.	Seen Objs.	Groups
Caption Programs	count-att	attr	none	✗	✓	✗	✓	✓
	extreme-right	⊎ [attr_1,..,attr_4]	none	✗	✓	✓	✓	✗
	extreme-left	⊎ [attr_1,..,attr_4]	none	✗	✓	✓	✓	✗
	extreme-behind	⊎ [attr_1,..,attr_4]	none	✗	✓	✓	✓	✗
	extreme-front	⊎ [attr_1,..,attr_4]	none	✗	✓	✓	✓	✗
	extreme-centre	⊎ [attr_1,..,attr_4]	none	✗	✓	✓	✓	✗
	unique-obj	⊎ [attr_1,..,attr_4]	none	✗	✓	✓	✓	✗
	obj-relation	attr_obj_1,pos,attr_obj_2	none	✗	✓	✓	✓	✗
Question Programs	count-all	-	num	✗	✗	✗	✗	✓
	count-other	-	num	✗	✗	✓	✓	✗
	count-all-group	-	num	✗	✗	✗	✗	✗
	count-attribute	attr	num	✗	✓	✓	✓	✓
	count-attribute-group	attr	num	✗	✓	✓	✓	✓
	count-obj-rel-imm	pos	num	✗	✗	✓	✓	✓
	count-obj-rel-imm-2	pos	num	✗	✗	✓	✓	✓
	count-obj-rel-early	pos,attr	num	✓	✓	✓	✓	✓
	count-obj-exclude-imm	attr_type	num	✗	✗	✓	✓	✓
	count-obj-exclude-early	attr_type,attr	num	✓	✗	✓	✓	✓
	exist-other	-	yes/no	✗	✗	✗	✓	✓
	exist-attribute	attr	yes/no	✗	✓	✗	✓	✓
	exist-attribute-group	attr	yes/no	✗	✓	✓	✓	✓
	exist-obj-rel-imm	pos	yes/no	✗	✗	✓	✓	✓
	exist-obj-rel-imm2	pos	yes/no	✗	✗	✓	✓	✓
	exist-obj-rel-early	pos,attr	yes/no	✓	✓	✓	✓	✓
	exist-obj-exclude-imm	attr_type	yes/no	✗	✗	✓	✓	✓
	exist-obj-exclude-early	attr_type,attr	yes/no	✓	✗	✗	✗	✓
	seek-attr-imm	attr_type	attr	✗	✓	✗	✗	✗
	seek-attr-imm2	attr_type	attr	✗	✓	✗	✗	✗
	seek-attr-early	attr_type,attr	attr	✓	✓	✓	✓	✗
	seek-attr-sim-early	attr_type,attr	attr	✓	✓	✓	✓	✗
	seek-attr-rel-imm	attr_type	attr	✗	✓	✓	✓	✗
	seek-attr-rel-early	attr_type,pos,attr	attr	✓	✓	✓	✓	✗

Table 4: Our Domain Specific Language (DSL) for CLEVR-Dialog. We present each function with its expected argument types, output, and knowledge base operations. Given a set of n arguments, the ⊎ operator selects a subset of $m \leq n$ possible ones, e.g. ⊎[attr_1,..,attr_4] = [attr_2,attr_3].

$\texttt{attr\_1} \in \texttt{COLOURS=[blue,brown,cyan,grey,green,purple,red,yellow]},$
$\texttt{attr\_2} \in \texttt{MATERIALS=[rubber,metal]},$
$\texttt{attr\_3} \in \texttt{SHAPES=[cube,cylinder,sphere]},$
$\texttt{attr\_4} \in \texttt{SIZES=[large,small]},$
$\texttt{attr,attr\_obj\_1,attr\_obj\_2} \in \bigcup \{\texttt{COLOURS,MATERIALS,SHAPES,SIZES}\},$
$\texttt{attr\_type} \in \texttt{[colour,material,shape,size]},$
$\texttt{pos} \in \texttt{[right,left,front,behind]},$
$\texttt{num} \in \texttt{[0,1,2,3,4,5,6,7,8,9,10]}.$

Table 5: Argument types of our DSL.

trained the caption and question program generators separately. We used a 2-layered bi-directional LSTM with a hidden size of 256 in both caption and question encoders. In the decoder, we used a 2-layered LSTM with a hidden size of 512. We fixed the batch size to 64 and used the Adam optimiser (Kingma and Ba, 2014) with a learning rate of 7×10^{-4} to train for *one* epoch. Every 2000 iteration, we validated the models based on the program accuracy in order not to select one that follows a flawed logic, i.e. a wrong program, to predict an answer.

Baselines. We trained the purely-connectionist baselines using the official code [2] we obtained from the authors of Shah et al. (2020). We used the same hyperparameters and training scheme, i.e. we trained the models for a maximum of 25 epochs and used early stopping when the validation accuracy did not improve for five consecutive epochs. For the HCN models, we used a publicly available codebase[3] for training with the same hyper-parameters as in (Williams et al., 2017).

Runtime Analysis. We conducted all of our experiments on a single NVIDIA Tesla V100 GPU. The training runtimes for one epoch when using a batch size of 64 are illustrated in Figure 7. As can be seen, all the purely-connectionist baselines need more than seven hours to complete one epoch. Although NSVD-concat concatenates the previous dialog rounds to form the history similarly to the baselines, it is more computationally efficient as it only needs circa 22 minutes to complete one epoch. Finally, HCN needs around 19 minutes to complete one epoch and NSVD-stack only around 14 which solidifies our aforementioned efficiency claims.

[2] https://github.com/ahmedshah1494/clevr-dialog-mac-net/tree/dialog-macnet

[3] https://github.com/jojonki/Hybrid-Code-Networks

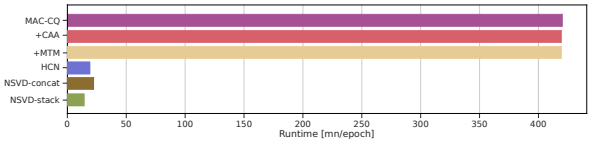

Figure 7: Training runtime comparison of the models. The training was conducted on a single NVIDIA Tesla V100 GPU with batch size 64.

Model	Prog. Acc.		Executor Acc.
	Caption	Question	
Caption-Net	99.79	-	
NSVD-concat	-	99.87	99.99
NSVD-stack	-	99.99	

Table 6: Quantitative analysis of our models' logic. The high program accuracies demonstrate that our models follow the implemented logic to predict the correct answer, i.e. they do not execute false programs that by chance might lead to a correct prediction. In addition, when tested with the ground truth scene annotations and programs, our executor reaches an answer-accuracy of 99.99% showcasing its flawless logic.

A.4 Quantitative Analysis of (our) Logic.

To quantify the logical capabilities of our models, we measured the caption and question program accuracies on the test split. As we can see from Table 6, our model achieves 99.79% caption program accuracy and our best question generator reaches the 99.99% accuracy mark. That is, they do not follow a flawed logic, i.e. a wrong program, to predict the correct answer. Furthermore, we evaluated the logic of our program executor by measuring its answer accuracy when provided with the ground truth programs and scene annotations of the test split. The last column of Table 6 shows that it reaches 99.99% answer accuracy indicating that its logic is close to flawless. The reason why it does not reach the 100% mark is that some scene captions cannot be uniquely interpreted leading to potential

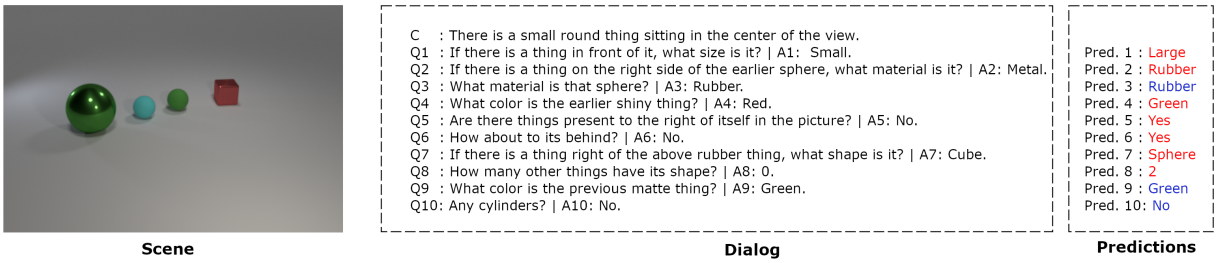

Figure 8: Example of a situation where a caption cannot be uniquely interpreted. The program `extreme-centre(cylinder, small)` induced from the caption "there is a small round thing sitting in the centre of the view" does not lead to a unique initialisation of our executor's knowledge base as there are *two* small spheres in the centre of the scene. By our logic, we consider it to be the cyan one. Incorrectly initialising the knowledge base leads to confusion when answering the subsequent questions. The blue and red colours indicate a match or a mismatch between the predicted answer and the ground truth, respectively.

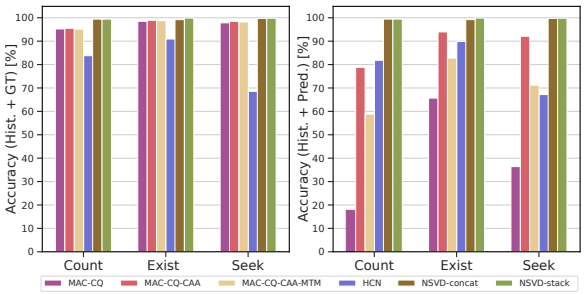

Figure 9: Detailed performance comparison for the different question categories (`Count`, `Exist`, `Seek`). Independent of the evaluation approach, our models achieve new state-of-art results on all question categories. These improvements are statistically significant with $p < 0.0001$ in all categories.

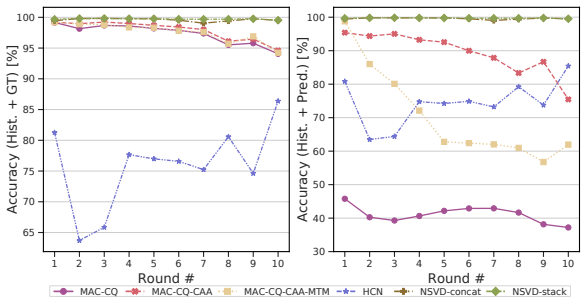

Figure 10: Accuracy for different question rounds. Our models significantly outperform the baselines with $p < 0.0001$ in all question rounds.

confusions when answering the subsequent questions. Figure 8 illustrates a concrete example of such a case. The caption "there is a small round thing sitting in the centre of the view" induces the program `extreme-centre(cylinder, small)`. However, this can be interpreted in two different ways since there are *two* small spheres in the centre of the scene, i.e. the cyan and the green ones. Therefore, the performance of our executor at answering the following dialog questions depends on which object is considered as the central one. By our logic, we consider it to be the cyan one. The subsequent questions, ground truth answers and predictions are also shown in Figure 8.

A.5 Fine-grained Evaluations

When looking at the performance for different question rounds (Figure 10), we can see that while the performance of the baselines starts to deteriorate quickly with longer dialogs, our models achieve a consistently-high performance across all rounds.

The inability to maintain performance becomes even more apparent (especially for the purely-connectionist baselines) when using the stricter "Hist. + Pred." evaluation scheme. The best baseline, *MAC-CQ-CAA*, suffers from a drop of 8.48% in accuracy and 0.19 in NFFR. The same observation can be made when analysing performance for the different CLEVR-Dialog question categories (`Count`, `Exist`, and `Seek`). Our models outperform all baselines for all categories and both evaluation schemes (Figure 9). These improvements are statistically significant with $p < 0.00001$.

Figure 11 illustrates the performance of our models on individual question types compared to the baselines. As can be seen, our models outperform the baselines on almost all question types with *NSVD-stack* topping them all with an accuracy of over 98% for all types. As seen from previous experiments, the gap between our models and the baseline becomes ever more conspicuous when the "Hist. + Pred." evaluation scheme is deployed (bottom table of Figure 11).

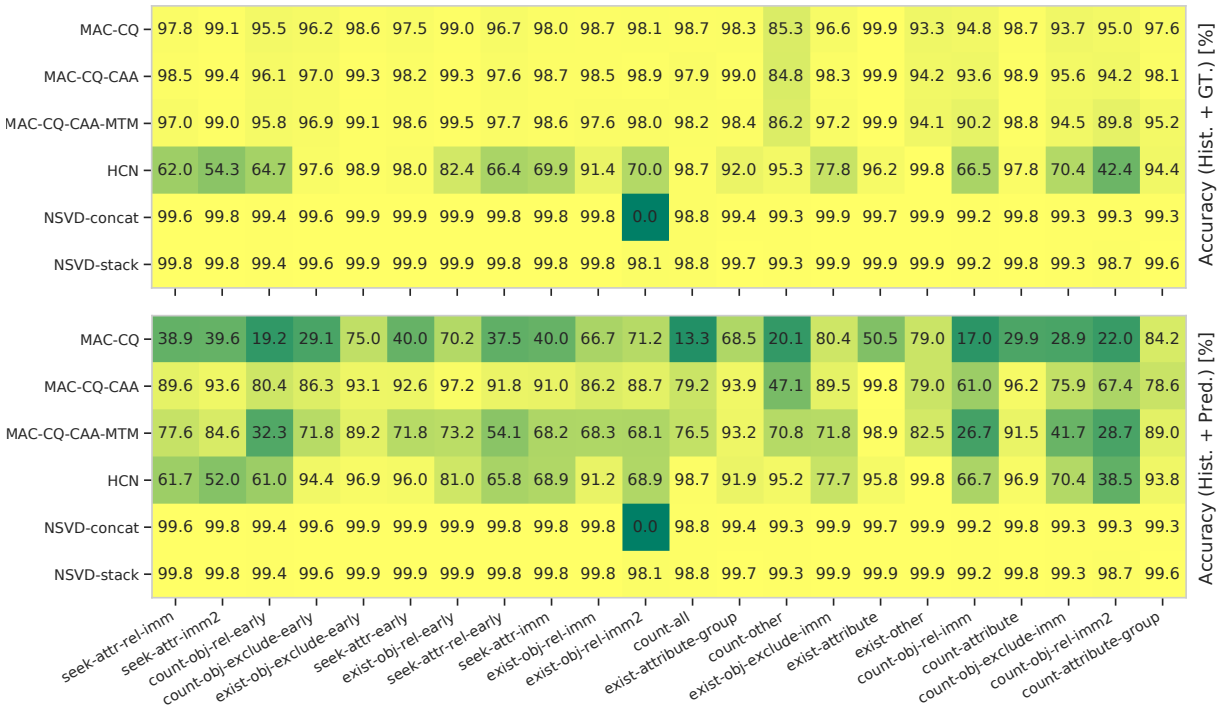

Figure 11: Performance comparison on the individual question types. **Top:** Models were evaluated following the "Hist. + GT" evaluation scheme. **Bottom:** Models were evaluated following the "Hist. + Pred." evaluation scheme

A.6 Data Efficiency

To study the data efficiency of our models further, we trained them and the baselines on $20\%, 40\%, 60\%, 80\%$, and 100% of the available training data. Not to bias the data towards any specific question type, we used the same distribution of question types to construct the reduced training sets. After training, we evaluated the models on the test split using both evaluation schemes. Our models not only outperform all baselines for the same reduced dataset but also in the extreme case of training the baselines with 100% of the data and our models with only 20% (Figure 13). While the performance of the neuro-symbolic models is only slightly affected by the size of the training data, the connectionist baselines' performance deteriorates with less data. This deterioration becomes even more significant when the stricter "Hist. + Pred." evaluation scheme is used.

Model	Hist. + GT		Hist. + Pred.	
	Acc.	NFFR ↑	Acc.	NFFR ↑
MAC-CQ	64.30*	0.27	59.96‡	0.24
+ CAA	64.28‡	0.27	57.69‡	0.23
+ MTM	61.55‡	0.25	52.04‡	0.20
HCN	47.31	0.14	46.50	0.14
NSVD-concat	91.57‡	0.76	91.57‡	0.76
NSVD-stack	**92.46**‡	**0.83**	**92.46**‡	**0.83**

Table 7: Performance comparison on Minecraft-Dialog *test*. Results are shown for both "Hist. + GT" and "Hist. + Pred." Our proposed models are highlighted in grey; best performance is in bold. ‡ and ∗ represents $p < 0.00001$ and $p \geq 0.05$ compared to the second best score in the respective column, respectively.

A.7 Generalisation to Other Scene Domains

In this experiment, we show that our method could be extended to a new reasoning testbed. Contrarily to CLEVR, the new scenes are grounded in Minecraft and are, as can be seen in Figure 12, drastically different in terms of context and scene constellations. Specifically, they comes with more entities (12 vs 3 in CLEVR) that have drastically different visual appearances. These entities can be grouped in a hierarchical manner (e.g. "a cow" and "a wolf" are both "animals" whereas "an animal" and "a human" are both "creatures").

Following (Yi et al., 2018), were rendered $10,000$ using the generation tool provided by Wu et al. (2017). The scenes consist of three to six objects in a 2D plane that are sampled from 12 entities with 4 different facing directions. Finally, we filtered out scenes that contain fully-occluded objects. We used $5,273$ images for training, $1,500$ for validation, and $1,000$ for testing. Furthermore, we adapted the dialog generation tool provided by Kottur et al. (2019) to be able to account for the

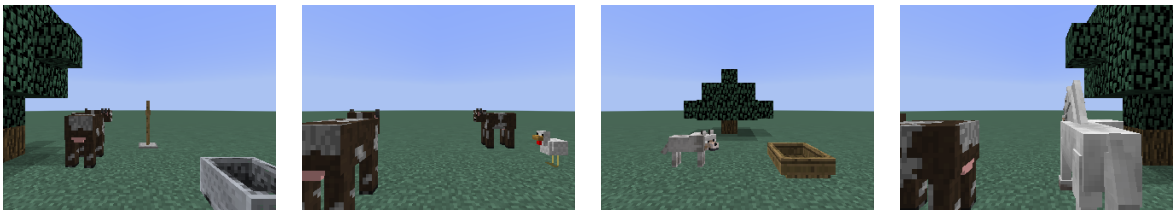

Figure 12: Sample images from the Minecraft dataset.

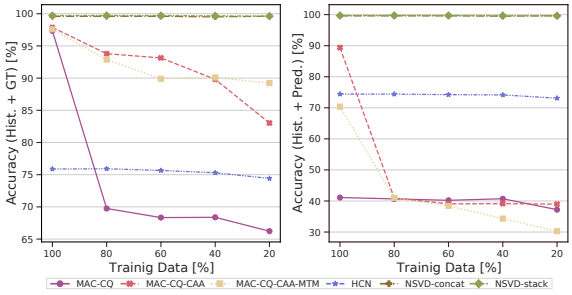

Figure 13: Accuracy when trained on limited amounts of data (20%, 40%, 60%, 80%, and 100% of the overall training data) and evaluated on the test split following the "Hist. + GT" and "Hist. + Pred." approaches.

different scene properties.

Similar to previous experiments, we generated five dialogs for every image consisting of 10 rounds each. We call this dataset Minecraft-Dialog. The results are summarised in Table 7. We compared our models to the same baselines as in previous experiments in terms of accuracy as well as NFRR. Once again, our neuro-symbolic models managed to significantly outperform all the baselines by achieving 92.64% and 91.57% accuracies for *NSVD-stack* and *NSVD-concat*, respectively, while maintaining high NFRR values.

Contrarily, the best connectionist model achieved test accuracies of 64.30% and 59.96% using the "Hist. + GT" and "Hist. + Pred." evaluation schemes, respectively. Finally, HCN achieved its best performance of 47.31 accuracy and 0.14 NFRR when the "Hist. + GT" evaluation scheme was used. Compared to CLEVR-Dialog (Table 1), the performance of our models witnessed a drop in this experiment which is attributed to the difficulty of the Minecraft scenes that come with heavy-occluded and diverse objects. However, the promising results of our models showcase that they indeed can generalise to new scene domains other then CLEVR.

A.8 Input/Output Samples

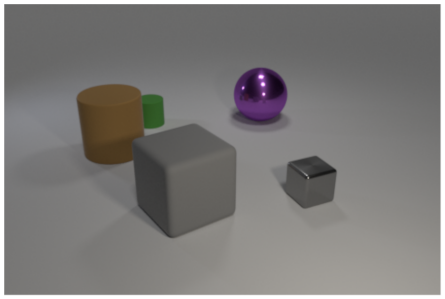

Scene

```
C   : There is a block to the right of all object.
Q1  : If there is an object behind it, what is its shape? | A1:  Cylinder.
Q2  : And material? | A2: Rubber.
Q3  : What about size? | A3: Large.
Q4  : And colour? | A4: Brown.
Q5  : What is the count of objects the previous block has to its front? | A5: 1.
Q6  : How many other objects have the same size as that large object? | A6: 2.
Q7  : Does the picture have other objects that share the same color with the aforementioned cylinder? | A7:No.
Q8  : What color is the aforementioned cube? | A8: Grey.
Q9  : If there is an object in front of it , what is its material? | A9: Rubber.
Q10: Does the previous brown object have objects to its left? | A10: No.
```

Dialog

GT programs

```
P0 :extreme-right(cube)
P1 :seek-attr-rel-imm(shape, behind)
P2 :seek-attr-imm2(material)
P3 :seek-attr-imm2(size)
P4 :seek-attr-imm2(colour)
P5 :count-obj-rel-early(front, cube)
P6 :count-obj-exclude-early(size, large)
P7 :exist-obj-exclude-early(color, cylinder)
P8 :seek-attr-early(color, cube)
P9 :seek-attr-rel-imm(material, front)
P10:exist-obj-rel-early(left, brown)
```

Pred. programs & answers (NSVD-Stack)

```
P0 :extreme-right(cube)                    Pred. 0 : Init.
P1 :seek-attr-rel-imm(shape, behind)       Pred. 1 : Cylinder
P2 :seek-attr-imm2(material)               Pred. 2 : Rubber
P3 :seek-attr-imm2(size)                   Pred. 3 : Large
P4 :seek-attr-imm2(colour)                 Pred. 4 : Brown
P5 :count-obj-rel-early(front, cube)       Pred. 5 : 1
P6 :count-obj-exclude-early(size, large)   Pred. 6 : 2
P7 :exist-obj-exclude-early(color, cylinder) Pred. 7 : No
P8 :seek-attr-early(color, cube)           Pred. 8 : Grey
P9 :seek-attr-rel-imm(material, front)     Pred. 9 : Rubber
P10:exist-obj-rel-early(left, brown)       Pred. 10: No
```

Pred. programs & answers (NSVD-Concat)

```
P0 :extreme-right(cube)                    Pred. 0 : Init.
P1 :seek-attr-rel-imm(shape, behind)       Pred. 1 : Cylinder
P2 :seek-attr-imm2(material)               Pred. 2 : Rubber
P3 :seek-attr-imm2(size)                   Pred. 3 : Large
P4 :seek-attr-imm2(colour)                 Pred. 4 : Brown
P5 :count-obj-rel-early(front, cube)       Pred. 5 : 1
P6 :count-obj-exclude-early(size, large)   Pred. 6 : 2
P7 :exist-obj-exclude-early(color, cylinder) Pred. 7 : No
P8 :seek-attr-early(color, cube)           Pred. 8 : Grey
P9 :seek-attr-rel-imm(material, front)     Pred. 9 : Rubber
P10:exist-obj-rel-early(left, brown)       Pred. 10: No
```

Figure 14: Inference example of our models on a test instance. Both of our models executed correct programs to predict the answers. The blue colour indicates a match between the predicted program/answer and the ground truth.

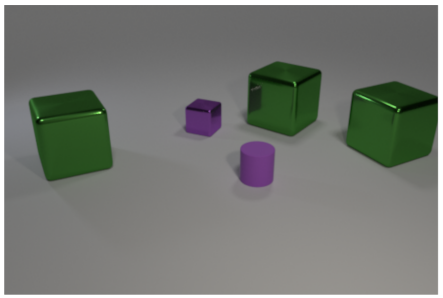

Scene

```
C   : There is a rubber thing in front of a large thing in the view.
Q1  : How many other things are in the image? | A1:  3.
Q2  : What color is the above large thing? | A2: Green.
Q3  : What is the shape of the previous matte thing? | A3: Cylinder.
Q4  : What material is the previous big thing? | A4: Metal.
Q5  : If there is a thing to the left of that matte thing , what is its color? | A5: Purple.
Q6  : Does it have things to to the right of itself? | A6: Yes.
Q7  : And to its left? | A7: Yes.
Q8  : If there is a thing behind the earlier purple thing , what shape is it? | A8: Cube.
Q9  : If there is a thing in front of the previous purple thing , what is its shape? | A9: Cube.
Q10: What number of red things are present? | A10: 0.
```

Dialog

GT programs

```
P0 :obj-relation(rubber, large, front)
P1 :count-other()
P2 :seek-attr-early(color, large)
P3 :seek-attr-early(shape, rubber)
P4 :seek-attr-early(material, large)
P5 :seek-attr-rel-early(color, left, rubber)
P6 :exist-obj-rel-imm(right)
P7 :exist-obj-rel-imm2(left)
P8 :seek-attr-rel-early(shape,behind,purple)
P9 :seek-attr-rel-early(shape,front,purple)
P10:count-attribute(red)
```

Pred. programs & answers (NSVD-Stack)

```
P0 :obj-relation(rubber, large, front)       Pred. 0 : Init.
P1 :count-other()                            Pred. 1 : 3
P2 :seek-attr-early(color, large)            Pred. 2 : Green
P3 :seek-attr-early(shape, rubber)           Pred. 3 : Cylinder
P4 :seek-attr-early(material, large)         Pred. 4 : Metal
P5 :seek-attr-rel-early(color, left, rubber) Pred. 5 : Purple
P6 :exist-obj-rel-imm(right)                 Pred. 6 : Yes
P7 :exist-obj-rel-imm2(left)                 Pred. 7 : Yes
P8 :seek-attr-rel-early(shape,behind,purple) Pred. 8 : Cube
P9 :seek-attr-rel-early(shape,front,purple)  Pred. 9 : Cube
P10:count-attribute(red)                     Pred. 10: 0
```

Pred. programs & answers (NSVD-Concat)

```
P0 :obj-relation(rubber, large, front)       Pred. 0 : Init.
P1 :count-other()                            Pred. 1 : 3
P2 :seek-attr-early(color, large)            Pred. 2 : Green
P3 :seek-attr-early(shape, rubber)           Pred. 3 : Cylinder
P4 :seek-attr-early(material, large)         Pred. 4 : Metal
P5 :seek-attr-rel-early(color, left, rubber) Pred. 5 : Purple
P6 :exist-obj-rel-imm(right)                 Pred. 6 : Yes
P7 :count-obj-rel-imm2(left)                 Pred. 7 : 1
P8 :seek-attr-rel-early(shape,behind,purple) Pred. 8 : Cube
P9 :seek-attr-rel-early(shape,front,purple)  Pred. 9 : Cube
P10:count-attribute(red)                     Pred. 10: 0
```

Figure 15: Inference example of our models on a test instance. While *NSVD-stack* answered all questions correctly, *NSVD-concat* failed at round 7. The blue and red colours indicate a match or a mismatch between the predicted program/answer and the ground truth, respectively.

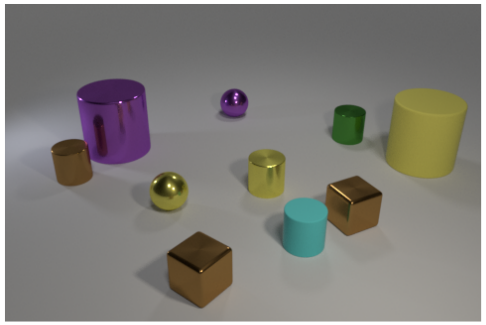

Scene

C : The view has a cyan thing on the left side of a block.
Q1 : What is the count of balls , if present? | A1: 2.
Q2 : What is the size of that block? | A2: Small.
Q3 : If there is a thing in front of the previous cyan thing, what is its material? | A3: Metal.
Q4 : What color is the above cube? | A4: Brown.
Q5 : What is the size of the above metallic thing? | A5: Small.
Q6 : Does it have things to in front of itself in the image? | A6: No.
Q7 : And to its left? | A7 : Yes.
Q8 : If there is a thing in front of the previous brown thing, what material is it? | A8: Rubber.
Q9 : What shape is the above shiny thing? | A9: Cube.
Q10: How many things are in the scene? | A10: 10.

Dialog

```
P0 :obj-relation(cyan, cube, left)
P1 :count-attribute(sphere)
P2 :seek-attr-early(size, cube)
P3 :seek-attr-rel-early(material,front,cyan)
P4 :seek-attr-early(color, cube)
P5 :seek-attr-early(size, metal)
P6 :exist-obj-rel-imm(front)
P7 :exist-obj-rel-imm2(left)
P8 :seek-attr-rel-early(material,front,brown)
P9 :seek-attr-early(shape, metal)
P10:count-all()
```

GT Programs

P0 :obj-relation(sphere, cube, left)	Pred. 0 : Init
P1 :count-attribute(sphere)	Pred. 1 : 0.
P2 :seek-attr-early(size, cube)	Pred. 2 : Small.
P3 :seek-attr-rel-early(material,front,small)	Pred. 3 : None.
P4 :seek-attr-early(color, cube)	Pred. 4 : Gray.
P5 :seek-attr-early(size, rubber)	Pred. 5 : Small.
P6 :exist-obj-rel-imm(front)	Pred. 6 : No.
P7 :count-obj-rel-imm2(left)	Pred. 7 : 0.
P8 :seek-attr-rel-early(material,front,rubber)	Pred. 8 : None.
P9 :seek-attr-early(shape, rubber)	Pred. 9 : Cube.
P10:count-all()	Pred. 10: 4.

Pred. programs & answers (NSVD-Stack)

P0 :obj-relation(sphere, cube, left)	Pred. 0 : Init
P1 :count-attribute(sphere)	Pred. 1 : 0.
P2 :seek-attr-early(size, cube)	Pred. 2 : Small.
P3 :seek-attr-rel-early(material,front,gray)	Pred. 3 : None.
P4 :seek-attr-early(color, cube)	Pred. 4 : Gray.
P5 :seek-attr-early(size, sphere)	Pred. 5 : Small.
P6 :exist-obj-rel-imm(right)	Pred. 6 : Yes.
P7 :exist-obj-rel-imm2(left)	Pred. 7 : No.
P8 :seek-attr-rel-early(material,front,gray)	Pred. 8 : None.
P9 :seek-attr-early(shape, rubber)	Pred. 9 : Cube.
P10:count-all()	Pred. 10: 4.

Pred. programs & answers (NSVD-Concat)

Figure 16: Inference example of our models on a test instance with 10 objects before fine-tuning. The blue and red colours indicate a match or a mismatch between the predicted program/answer and the ground truth, respectively.

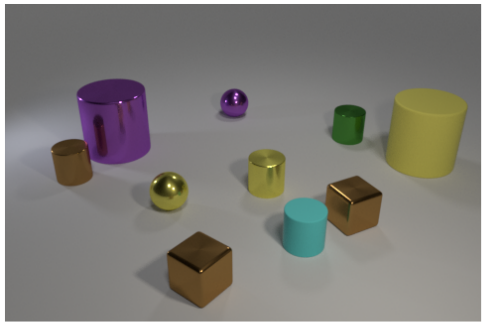

Scene

C : The view has a cyan thing on the left side of a block.
Q1 : What is the count of balls , if present? | A1: 2.
Q2 : What is the size of that block? | A2: Small.
Q3 : If there is a thing in front of the previous cyan thing, what is its material? | A3: Metal.
Q4 : What color is the above cube? | A4: Brown.
Q5 : What is the size of the above metallic thing? | A5: Small.
Q6 : Does it have things to in front of itself in the image? | A6: No.
Q7 : And to its left? | A7 : Yes.
Q8 : If there is a thing in front of the previous brown thing, what material is it? | A8: Rubber.
Q9 : What shape is the above shiny thing? | A9: Cube.
Q10: How many things are in the scene? | A10: 10.

Dialog

```
P0 :obj-relation(cyan, cube, left)
P1 :count-attribute(sphere)
P2 :seek-attr-early(size, cube)
P3 :seek-attr-rel-early(material,front,cyan)
P4 :seek-attr-early(color, cube)
P5 :seek-attr-early(size, metal)
P6 :exist-obj-rel-imm(front)
P7 :exist-obj-rel-imm2(left)
P8 :seek-attr-rel-early(material,front,brown)
P9 :seek-attr-early(shape, metal)
P10:count-all()
```

GT Programs

P0 :obj-relation(cyan, cube, left)	Pred. 0 : Init
P1 :count-attribute(sphere)	Pred. 1 : 2.
P2 :seek-attr-early(size, cube)	Pred. 2 : Small.
P3 :seek-attr-rel-early(material,front,cyan)	Pred. 3 : Metal.
P4 :seek-attr-early(color, cube)	Pred. 4 : Brown.
P5 :seek-attr-early(size, metal)	Pred. 5 : Small.
P6 :exist-obj-rel-imm(front)	Pred. 6 : No.
P7 :exist-obj-rel-imm2(left)	Pred. 7 : Yes.
P8 :seek-attr-rel-early(material,front,brown)	Pred. 8 : Rubber.
P9 :seek-attr-early(shape, metal)	Pred. 9 : Cube.
P10:count-all()	Pred. 10: 10.

Pred. programs & answers (NSVD-Stack)

P0 :obj-relation(cyan, cube, left)	Pred. 0 : Init
P1 :count-attribute(sphere)	Pred. 1 : 2.
P2 :seek-attr-early(size, cube)	Pred. 2 : Small.
P3 :seek-attr-rel-early(material,front,cyan)	Pred. 3 : Metal.
P4 :seek-attr-early(color, cube)	Pred. 4 : Brown.
P5 :seek-attr-early(size, metal)	Pred. 5 : Small.
P6 :exist-obj-rel-imm(front)	Pred. 6 : No.
P7 :exist-obj-rel-imm2(left)	Pred. 7 : Yes.
P8 :seek-attr-rel-early(material,front,brown)	Pred. 8 : Rubber.
P9 :seek-attr-early(shape, metal)	Pred. 9 : Cube.
P10:count-all()	Pred. 10: 10.

Pred. programs & answers (NSVD-Concat)

Figure 17: Inference example of our models on a test instance with 10 objects after fine-tuning. The blue and red colours indicate a match or a mismatch between the predicted program/answer and the ground truth, respectively.

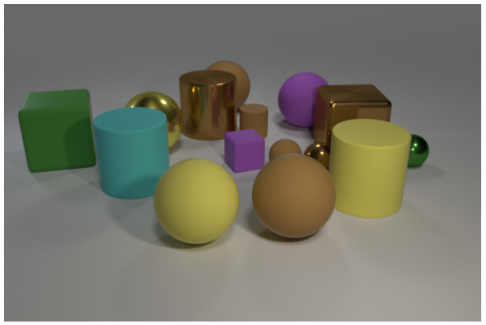

Scene

C : There are 2 green things in the scene.
Q1 : How many of them are cubes? | A1: 1.
Q2 : If there is a thing right of it, what is its color? | A2: Yellow.
Q3 : And material? | A3: Metal.
Q4 : And size? | A4: Large.
Q5 : What material is the aforementioned block? | A5: Rubber.
Q6 : Does the scene have other things that share the same material with the previous metal thing? | A6: Yes.
Q7 : Does the image have other things that share the same shape with the aforementioned rubber thing? | A7 : Yes.
Q8 : What is the shape of the previous yellow thing? | A8: Sphere.
Q9 : What size is the above rubber thing? | A9: Large.
Q10: What number of things are present to its right? | A10: 14.

Dialog

```
P0 :count-att(green)
P1 :count-attribute-group(cube)
P2 :seek-attr-rel-imm(color, right)
P3 :seek-attr-imm2(material)
P4 :seek-attr-imm2(size)
P5 :seek-attr-early(material, cube)
P6 :exist-obj-exclude-early(material, metal)
P7 :exist-obj-exclude-early(shape, rubber)
P8 :seek-attr-early(shape, yellow)
P9 :seek-attr-early(size, rubber)
P10:count-obj-rel-imm(right)
```

GT Programs

```
P0 :count-att(sphere)                          Pred. 0 : Init
P1 :count-attribute-group(cube)                Pred. 1 : 0.
P2 :seek-attr-rel-imm(color, right)            Pred. 2 : Error.
P3 :seek-attr-imm2(material)                   Pred. 3 : Error.
P4 :seek-attr-imm2(size)                       Pred. 4 : Error.
P5 :seek-attr-early(material, cube)            Pred. 5 : Error.
P6 :exist-obj-exclude-early(material,rubber)   Pred. 6 : Error.
P7 :exist-obj-exclude-early(shape, rubber)     Pred. 7 : Error.
P8 :seek-attr-early(shape, small)              Pred. 8 : Error.
P9 :seek-attr-early(size, rubber)              Pred. 9 : Error.
P10:count-obj-rel-imm(right)                   Pred. 10: Error.
```

Pred. programs & answers (NSVD-Stack)

```
P0 :count-att(cube)                            Pred. 0 : Init
P1 :count-attribute-group(cube)                Pred. 1 : 4.
P2 :seek-attr-rel-imm(color, right)            Pred. 2 : Error.
P3 :seek-attr-imm2(material)                   Pred. 3 : Error.
P4 :seek-attr-imm2(size)                       Pred. 4 : Error.
P5 :seek-attr-early(material, cube)            Pred. 5 : Error.
P6 :exist-obj-exclude-early(material,shpere)   Pred. 6 : Error.
P7 :exist-obj-exclude-early(shape, rubber)     Pred. 7 : Error.
P8 :seek-attr-early(shape, sphere)             Pred. 8 : Error.
P9 :seek-attr-early(size, rubber)              Pred. 9 : Error.
P10:count-obj-rel-imm(right)                   Pred. 10: Error.
```

Pred. programs & answers (NSVD-Concat)

Figure 18: Inference example of our models on a test instance with 15 objects before fine-tuning. The blue and red colours indicate a match or a mismatch between the predicted program/answer and the ground truth, respectively. `Error` means that the predicted program encountered an exception during execution.

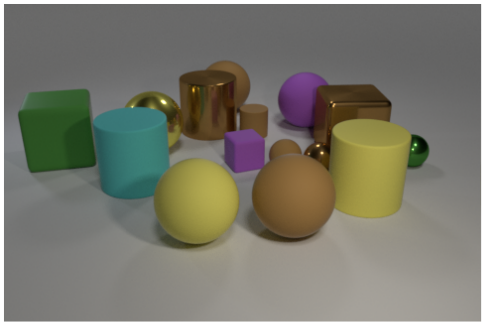

Scene

C : There are 2 green things in the scene.
Q1 : How many of them are cubes? | A1: 1.
Q2 : If there is a thing right of it, what is its color? | A2: Yellow.
Q3 : And material? | A3: Metal.
Q4 : And size? | A4: Large.
Q5 : What material is the aforementioned block? | A5: Rubber.
Q6 : Does the scene have other things that share the same material with the previous metal thing? | A6: Yes.
Q7 : Does the image have other things that share the same shape with the aforementioned rubber thing? | A7 : Yes.
Q8 : What is the shape of the previous yellow thing? | A8: Sphere.
Q9 : What size is the above rubber thing? | A9: Large.
Q10: What number of things are present to its right? | A10: 14.

Dialog

```
P0 :count-att(green)
P1 :count-attribute-group(cube)
P2 :seek-attr-rel-imm(color, right)
P3 :seek-attr-imm2(material)
P4 :seek-attr-imm2(size)
P5 :seek-attr-early(material, cube)
P6 :exist-obj-exclude-early(material, metal)
P7 :exist-obj-exclude-early(shape, rubber)
P8 :seek-attr-early(shape, yellow)
P9 :seek-attr-early(size, rubber)
P10:count-obj-rel-imm(right)
```

GT Programs

```
P0 :count-att(green)                          Pred. 0 : Init
P1 :count-attribute-group(cube)               Pred. 1 : 1.
P2 :seek-attr-rel-imm(color, right)           Pred. 2 : Cyan.
P3 :seek-attr-imm2(material)                  Pred. 3 : Rubber.
P4 :seek-attr-imm2(size)                      Pred. 4 : Large.
P5 :seek-attr-early(material, cube)           Pred. 5 : Rubber.
P6 :exist-obj-exclude-early(material, metal)  Pred. 6 : Yes.
P7 :exist-obj-exclude-early(shape, rubber)    Pred. 7 : Yes.
P8 :seek-attr-early(shape, yellow)            Pred. 8 : Cylinder.
P9 :seek-attr-early(size, rubber)             Pred. 9 : Large.
P10:count-obj-rel-imm(right)                  Pred. 10: 13.
```

Pred. programs & answers (NSVD-Stack)

```
P0 :count-att(green)                          Pred. 0 : Init
P1 :count-attribute-group(metal)              Pred. 1 : 1.
P2 :seek-attr-rel-imm(color, right)           Pred. 2 : Cyan.
P3 :seek-attr-imm2(material)                  Pred. 3 : Rubber.
P4 :seek-attr-imm2(size)                      Pred. 4 : Large.
P5 :seek-attr-early(material, cube)           Pred. 5 : Rubber.
P6 :exist-obj-exclude-early(material, metal)  Pred. 6 : Yes.
P7 :exist-obj-exclude-early(shape, rubber)    Pred. 7 : Yes.
P8 :seek-attr-early(shape, yellow)            Pred. 8 : Cylinder.
P9 :seek-attr-early(size, rubber)             Pred. 9 : Large.
P10:count-obj-rel-imm(right)                  Pred. 10: 13.
```

Pred. programs & answers (NSVD-Concat)

Figure 19: Inference example of our models on a test instance with 15 objects after fine-tuning. The blue and red colours indicate a match or a mismatch between the predicted program/answer and the ground truth, respectively.

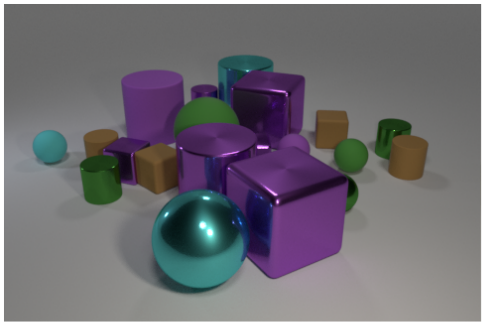

Scene

C : There is a matte object to the right of all objects in the view.
Q1 : If there is an object in front of it, what is its color? | A1: Brown.
Q2 : What about size? | A2: Small.
Q3 : What size is the earlier rubber object? | A3: Small.
Q4 : Does the previous brown object have objects to its behind? | A4: Yes.
Q5 : If there is an object to the left of the earlier brown object, what is its color? | A5: Purple.
Q6 : Are there objects present left of the above brown object? | A6: Yes.
Q7 : How many of them are blocks? | A7 : 1.
Q8 : What is its material? | A8: Metal.
Q9 : What size is it? | A9: Small.
Q10: How many other objects are in the view? | A10: 4.

Dialog

```
P0 :extreme-right(rubber)
P1 :seek-attr-rel-imm(color, front)
P2 :seek-attr-imm2(size)
P3 :seek-attr-early(size, rubber)
P4 :exist-obj-rel-early(behind, brown)
P5 :seek-attr-rel-early(color, left, brown)
P6 :exist-obj-rel-early(left, brown)
P7 :count-attribute-group(cube)
P8 :seek-attr-imm(material)
P9 :seek-attr-imm(size)
P10:count-other()
```

GT Programs

```
P0 :extreme-right(rubber)                    Pred. 0 : Init
P1 :seek-attr-rel-imm(color, front)          Pred. 1 : Gray.
P2 :seek-attr-imm2(size)                     Pred. 2 : Small.
P3 :seek-attr-early(size, rubber)            Pred. 3 : Small.
P4 :exist-obj-rel-early(behind, rubber)      Pred. 4 : Yes.
P5 :seek-attr-rel-early(color, left, rubber) Pred. 5 : Blue.
P6 :exist-obj-rel-early(left, rubber)        Pred. 6 : Yes.
P7 :count-attribute-group(cube)              Pred. 7 : 2.
P8 :seek-attr-imm(material)                  Pred. 8 : Rubber.
P9 :seek-attr-imm(size)                      Pred. 9 : Small.
P10:count-other()                            Pred. 10: 0.
```

Pred. programs & answers (NSVD-Stack)

```
P0 :extreme-right(rubber)                    Pred. 0 : Init
P1 :seek-attr-rel-imm(color, front)          Pred. 1 : Gray.
P2 :seek-attr-imm2(size)                     Pred. 2 : Small.
P3 :seek-attr-early(size, rubber)            Pred. 3 : Small.
P4 :exist-obj-rel-early(behind, cube)        Pred. 4 : Yes.
P5 :seek-attr-rel-early(color, left, red)    Pred. 5 : Blue.
P6 :exist-obj-rel-early(left, gray)          Pred. 6 : Yes.
P7 :count-attribute-group(cube)              Pred. 7 : 2.
P8 :seek-attr-imm(material)                  Pred. 8 : Rubber.
P9 :seek-attr-imm(size)                      Pred. 9 : Small.
P10:count-other()                            Pred. 10: 0.
```

Pred. programs & answers (NSVD-Concat)

Figure 20: Inference example of our models on a test instance with 20 objects before fine-tuning. The blue and red colours indicate a match or a mismatch between the predicted program/answer and the ground truth, respectively.

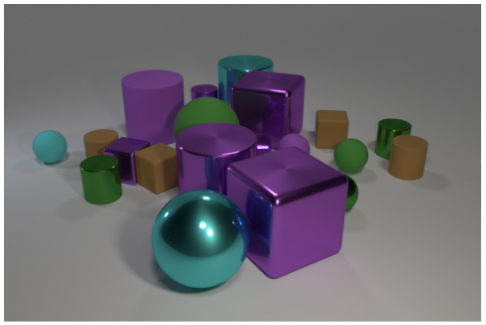

Scene

C : There is a matte object to the right of all objects in the view.
Q1 : If there is an object in front of it, what is its color? | A1: Brown.
Q2 : What about size? | A2: Small.
Q3 : What size is the earlier rubber object? | A3: Small.
Q4 : Does the previous brown object have objects to its behind? | A4: Yes.
Q5 : If there is an object to the left of the earlier brown object, what is its color? | A5: Purple.
Q6 : Are there objects present left of the above brown object? | A6: Yes.
Q7 : How many of them are blocks? | A7 : 1.
Q8 : What is its material? | A8: Metal.
Q9 : What size is it? | A9: Small.
Q10: How many other objects are in the view? | A10: 4.

Dialog

```
P0 :extreme-right(rubber)
P1 :seek-attr-rel-imm(color, front)
P2 :seek-attr-imm2(size)
P3 :seek-attr-early(size, rubber)
P4 :exist-obj-rel-early(behind, brown)
P5 :seek-attr-rel-early(color, left, brown)
P6 :exist-obj-rel-early(left, brown)
P7 :count-attribute-group(cube)
P8 :seek-attr-imm(material)
P9 :seek-attr-imm(size)
P10:count-other()
```

GT Programs

```
P0 :extreme-right(rubber)          Pred. 0 : Init
P1 :seek-attr-rel-imm(color, front)  Pred. 1 : Green.
P2 :seek-attr-imm2(size)           Pred. 2 : Small.
P3 :seek-attr-early(size, rubber)  Pred. 3 : Small.
P4 :exist-obj-rel-early(behind, brown)  Pred. 4 : Yes.
P5 :seek-attr-rel-early(color, left, brown)  Pred. 5 : Green.
P6 :exist-obj-rel-early(left, brown)  Pred. 6 : Yes.
P7 :count-attribute-group(cube)    Pred. 7 : 6.
P8 :seek-attr-imm(material)        Pred. 8 : Rubber.
P9 :seek-attr-imm(size)            Pred. 9 : Small.
P10:count-other()                  Pred. 10: 0.
```

Pred. programs & answers (NSVD-Stack)

```
P0 :extreme-right(rubber)          Pred. 0 : Init
P1 :seek-attr-rel-imm(color, front)  Pred. 1 : Green.
P2 :seek-attr-imm2(size)           Pred. 2 : Small.
P3 :seek-attr-early(size, rubber)  Pred. 3 : Small.
P4 :exist-obj-rel-early(behind, brown)  Pred. 4 : Yes.
P5 :seek-attr-rel-early(color, left, brown)  Pred. 5 : Green.
P6 :exist-obj-rel-early(left, brown)  Pred. 6 : Yes.
P7 :count-attribute-group(metal)   Pred. 7 : 11.
P8 :seek-attr-imm(material)        Pred. 8 : Rubber.
P9 :seek-attr-imm(size)            Pred. 9 : Small.
P10:count-other()                  Pred. 10: 0.
```

Pred. programs & answers (NSVD-Concat)

Figure 21: Inference example of our models on a test instance with 20 objects after fine-tuning. The blue and red colours indicate a match or a mismatch between the predicted program/answer and the ground truth, respectively.

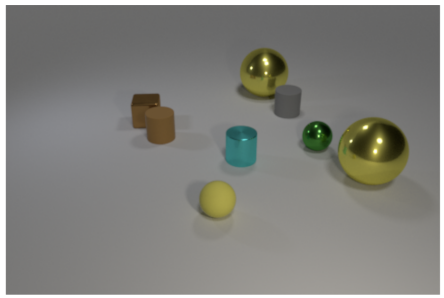

Scene

```
C   : No other green object except for exactly one.
Q1  : If there is an object to the left of it, what is its color? | A1:  Gray.
Q2  : What about material? | A2: Rubber.
Q3  : How about shape? | A3: Cylinder.
Q4  : What material is the aforementioned green object? | A4: Metal.
Q5  : Are there objects present behind the above gray object? | A5: Yes.
Q6  : What is the count of objects the earlier green object has to its behind? | A6: 4.
Q7  : Are there purple objects? | A7: No.
Q8  : If there is an object to the left of that cylinder, what is its material? | A8: Metal.
Q9  : If there is an object on the right side of it, what size is it? | A9: Small.
Q10: Are there objects present on the left side of the above green object? | A10: Yes.
```

Dialog

GT programs

```
P0 :unique-obj(green)
P1 :seek-attr-rel-imm(color, left)
P2 :seek-attr-imm2(material)
P3 :seek-attr-imm2(shape)
P4 :seek-attr-early(material, green)
P5 :exist-obj-rel-early(behind, gray)
P6 :count-obj-rel-early(behind, green)
P7 :exist-attribute(purple)
P8 :seek-attr-rel-early(material,left,cylinder)
P9 :seek-attr-rel-imm(size, right)
P10:exist-obj-rel-early(left, green)
```

Pred. programs & answers (NSVD-Stack)

```
P0 :unique-obj(green)                          Pred. 0 : Init.
P1 :seek-attr-rel-imm(color, left)             Pred. 1 : Gray
P2 :seek-attr-imm2(material)                   Pred. 2 : Rubber
P3 :seek-attr-imm2(shape)                      Pred. 3 : Cylinder
P4 :seek-attr-early(material, green)           Pred. 4 : Metal
P5 :exist-obj-rel-early(behind, gray)          Pred. 5 : Yes
P6 :count-obj-rel-early(behind, green)         Pred. 6 : 4
P7 :exist-other()                              Pred. 7 : Yes
P8 :seek-attr-rel-early(material,left,cylinder) Pred. 8 : Metal
P9 :seek-attr-rel-imm(size, right)             Pred. 9 : Small
P10:exist-obj-rel-early(left, green)           Pred. 10: Yes
```

Pred. programs & answers (NSVD-Concat)

```
P0 :unique-obj(green)                          Pred. 0 : Init.
P1 :seek-attr-rel-imm(color, left)             Pred. 1 : Gray
P2 :seek-attr-imm2(material)                   Pred. 2 : Rubber
P3 :seek-attr-imm2(shape)                      Pred. 3 : Cylinder
P4 :seek-attr-early(material, green)           Pred. 4 : Metal
P5 :seek-attr-rel-early(color, left, gray)     Pred. 5 : Yellow
P6 :seek-attr-early(size, green)               Pred. 6 : Small
P7 :exist-other()                              Pred. 7 : Yes
P8 :seek-attr-rel-early(material,left,cylinder) Pred. 8 : Metal
P9 :seek-attr-rel-imm(size, right)             Pred. 9 : Small
P10:seek-attr-early(size, green)               Pred. 10: Small
```

Figure 22: Inference example of our models on a test instance of split BB before fine-tuning. While *NSVD-stack* failed at answering only one question, *NSVD-concat* failed at 4 rounds. The blue and red colours indicate a match or a mismatch between the predicted program/answer and the ground truth, respectively.

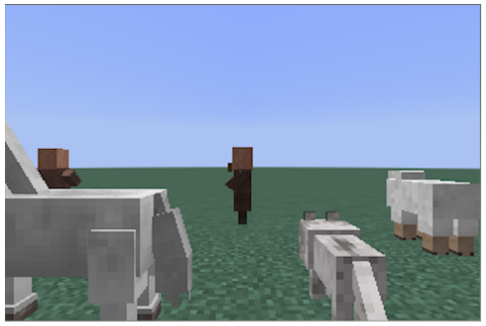

Scene

C : The view has a filly.
Q1 : If present, how many humans are in the view? | A1: 2.
Q2 : If there is a thing to the left of the aforementioned horse , what is its class? | A2: Villager.
Q3 : Does the aforementioned mare have things to its front? | A3: No.
Q4 : What direction is the above villager facing | A4: Backward.
Q5 : What is the direction of the previous mare? | A5: Left.
Q6 : If there is a thing on the right side of it, what is its direction? | A6 : Left.
Q7 : What about class? | A7 : Villager.
Q8 : Are there other things that are of same direction as that object facing backward? | A8: Yes.
Q9 : What is the nature of that filly? | A9: Animal.
Q10: What is the count of other things that share its direction? | A10: 1

Dialog

```
P0 :unique-obj(horse)
P1 :count-attribute(human)
P2 :seek-attr-rel-early(class, left, horse)
P3 :exist-obj-rel-early(front, horse)
P4 :seek-attr-early(direction, villager)
P5 :seek-attr-early(direction, horse)
P6 :seek-attr-rel-imm(direction, right)
P7 :seek-attr-imm2(class)
P8 :exist-obj-exclude-early(direction, backward)
P9 :seek-attr-early(nature, horse)
P10:count-obj-exclude-imm(direction)
```

GT Programs

```
P0 :unique-obj(horse)                          Pred. 0 : Init
P1 :count-attribute(human)                     Pred. 1 : 2.
P2 :seek-attr-rel-early(class, left, horse)    Pred. 2 : Villager.
P3 :exist-obj-rel-early(front, horse)          Pred. 3 : No.
P4 :seek-attr-early(direction, villager)       Pred. 4 : Backward.
P5 :seek-attr-early(direction, horse)          Pred. 5 : Left.
P6 :seek-attr-rel-imm(direction, right)        Pred. 6 : Left.
P7 :seek-attr-imm2(class)                      Pred. 7 : Villager.
P8 :exist-obj-exclude-early(direction, backward) Pred. 8 : Yes.
P9 :seek-attr-early(nature, horse)             Pred. 9 : Animal.
P10:count-obj-exclude-imm(direction)           Pred. 10: 1.
```

Pred. programs & answers (NSVD-Stack)

```
P0 :unique-obj(horse)                          Pred. 0 : Init
P1 :count-attribute(human)                     Pred. 1 : 2.
P2 :seek-attr-rel-early(class, left, horse)    Pred. 2 : Villager.
P3 :exist-obj-rel-early(front, horse)          Pred. 3 : No.
P4 :seek-attr-early(direction, villager)       Pred. 4 : Backward.
P5 :seek-attr-early(direction, horse)          Pred. 5 : Left.
P6 :seek-attr-rel-imm(direction, right)        Pred. 6 : Left.
P7 :seek-attr-imm2(class)                      Pred. 7 : Villager.
P8 :exist-obj-exclude-early(direction, backward) Pred. 8 : Yes.
P9 :seek-attr-early(nature, horse)             Pred. 9 : Animal.
P10:count-obj-exclude-imm(direction)           Pred. 10: 1.
```

Pred. programs & answers (NSVD-Concat)

Figure 23: Inference example of our models on a test instance of Minecraft-Dialog. The blue and red colours indicate a match or a mismatch between the predicted program/answer and the ground truth, respectively.

LINGUIST: Language Model Instruction Tuning to Generate Annotated Utterances for Intent Classification and Slot Tagging

Andy Rosenbaum*
Amazon, Cambridge, USA
andros@amazon.com

Saleh Soltan
Amazon, New York, USA
ssoltan@amazon.com

Wael Hamza
Amazon, Dallas, USA
waelhamz@amazon.com

Yannick Versley
Amazon, Aachen, Germany
yversley@amazon.de

Markus Boese
Amazon, Aachen, Germany
boesem@amazon.de

Abstract

We present LINGUIST, a method for generating annotated data for Intent Classification and Slot Tagging (IC+ST), via fine-tuning AlexaTM 5B, a 5-billion-parameter multilingual sequence-to-sequence (seq2seq) model, on a flexible instruction prompt. In a 10-shot novel intent setting for the SNIPS dataset, LINGUIST surpasses state-of-the-art approaches (Back-Translation and Example Extrapolation) by a wide margin, showing absolute improvement for the target intents of +1.9 points on IC Recall and +2.5 points on ST F1 Score. In the zero-shot cross-lingual setting of the mATIS++ dataset, LINGUIST outperforms a strong baseline of Machine Translation with Slot Alignment by +4.14 points absolute on ST F1 Score across 6 languages, while matching performance on IC. Finally, we verify our results on an internal large-scale multilingual dataset for conversational agent IC+ST and show significant improvements over a baseline which uses Back-Translation, Paraphrasing and Slot Catalog Resampling. To our knowledge, we are the first to demonstrate instruction fine-tuning of a large-scale seq2seq model to control the outputs of multilingual intent- and slot-labeled data generation.

1 Introduction

Conversational agents typically rely on large quantities of labeled training data to understand user requests through Intent Classification and Slot Tagging (IC+ST) (Tur and De Mori, 2011). Such data is plentiful for existing usage patterns (although costly to annotate), yet scarce for new intents/slots and new languages. A growing trend to address this problem is to generate synthetic training data, e.g. via Paraphrasing, Back-Translation (BT), slot replacement, and Example Extrapolation (Ex2). (Jolly et al., 2020; Xie et al., 2020; Zhang et al., 2020; Lee et al., 2021). In this work, we propose a novel data generation method called **L**anguage model **IN**struction tuning to **G**enerate annotated **U**tterances for **I**ntent classification and **S**lot **T**agging (**LINGUIST**).

```
INPUT:
<language> English </language>
<intent> GetWeather </intent>
<include>
 [1 * ] , [3 snow ] [5 tomorrow ]
</include>
<labels>
 [1=geographic_poi , [2=country ,
 [3=condition_description , [4=city ,
 [5=timeRange
</labels>
<examples>
 Will the weather be okay in
  [1 Yellowstone National Park ]
  [5 one week from now ] ? <br>
 will it [3 rain ] at the [1 Statue of Liberty ]
  at [5 noon ] <br>
 What's the weather like
  at [1 Disneyworld ] in [5 november ] <br>
 I need the weather info for
  the [1 Guggenheim Museum ] in [2 Spain ] <br>
  What is the weather forecast
  for [5 October 12, 2022 ] in [4 Gyeongju ]
</examples>
OUTPUTS:

1. Give the [1 National Wildlife Refuge ]
   forecast for [3 snow ] [5 tomorrow ]

2. Can I get the [3 snow ] forecast
   for [1 Lake Tahoe ] [5 tomorrow ]?

3. I want to know if it will [3 snow ]
   [5 tomorrow ] at
   [1 Mount Rainier State Park ].
```

Figure 1: LINGUIST uses an instruction prompt to generate data with both user-requested slot values ("snow") and model-generated values ("*"). This model **has not seen any training data for GetWeather intent, or for the slot tag geographic_poi**: it was fine-tuned only on the other 6 SNIPS intents.

Our LINGUIST method addresses several important gaps in the existing literature: (1) controlling the generated data to include specific slot types and values, (2) cross-lingual and multilingual data generation, and (3) ability to leverage the intent

*Correspondence Author: <andros@amazon.com>. Author contributions are listed in Appendix A.

Proceedings of the 29th International Conference on Computational Linguistics, pages 218–241
October 12–17, 2022.

and slot names to inform the generation. The key to these achievements is our design of a novel instruction prompt (Figure 1), consisting of natural language descriptions for the desired model outputs. We first fine-tune a large pre-trained seq2seq Transformer (Vaswani et al., 2017) model to learn how to generate annotated utterances following the prompt instructions. Then, for a novel intent or slot with only a few or even zero training examples, we apply the model to generate similar data, which we add to the training set for an IC+ST model.

We demonstrate the effectiveness of LINGUIST on three datasets by showing substantial improvements over strong baselines. (i) On a 10-shot novel intent setting with English SNIPS (Coucke et al., 2018), LINGUIST improves over Back-Translation and Ex2 by +1.9 points absolute on IC and +2.5 points absolute on ST. (ii) On cross-lingual mATIS++ (Xu et al., 2020), LINGUIST out-performs the best Machine Translation plus slot alignment reported by Xu et al., by +4.14 points in ST across 6 languages, while matching performance on IC. (iii) Finally, to demonstrate the success of our method on a real-world conversational agent system, we apply LINGUIST on an internal dataset containing hundreds of intents and slot types across 4 languages, and show large improvements over a baseline which uses Back-Translation and Paraphrasing.

We also show LINGUIST can generate IC+ST-annotated data from *zero examples*, using only the natural language intent and slot label names. LINGUIST achieves 80.0 IC Recall and 56.9 ST F1 Score on new SNIPS intents despite never seeing a single example for the new intents. To our knowledge, LINGUIST is the first system capable of generating IC+ST-annotated data in this setting.

2 Related Work

Large-scale Language Models (LLMs) such as GPT (Radford et al., 2019; Brown et al., 2020) and AlexaTM 20B (Soltan et al., 2022) excel at performing novel tasks with only a few examples via in-context learning, i.e. without requiring any model parameter updates. For example, Sahu et al. (2022) apply GPT-3 to generate variations of examples from a given class. Wang et al. (2021) generate both text and class label together via GPT-3, towards eliminating the need for human labeling.

Pre-training then fine-tuning of seq2seq models was introduced in English BART and multilingual mBART (Lewis et al., 2020a; Liu et al., 2020). T5 and mT5 (Raffel et al., 2020; Xue et al., 2021) extended the idea by framing more downstream tasks as text-to-text. FLAN (Wei et al., 2022) introduced **instruction tuning**, where a large-scale seq2seq model is fine-tuned on instruction prompts from a variety of tasks, in order to generalize to new tasks without any further parameter updates.

Prior work also explores conditioning generation on intent and slot labels. Ding et al. (2020) train a conditional language model on a mixture of annotated and unannotated text, allowing to sample novel annotated utterances. Malandrakis et al. (2019) train a seq2seq model from interpretation-text pairs, applying variational auto-encoders for more diversity. Jolly et al. (2020) expand on this, exploring different sampling strategies, adding more variety by shuffling slot names, and examining the behavior where a new intent is introduced with limited training data. Panda et al. (2021) extend this to the multilingual setting. Generative Insertion Transformers (Kumar et al., 2022) generate carrier phrases for a target intent and containing specific entities. A limitation of all these approaches is that the trained model cannot generalize to novel intents and slots at inference time.

Generative Conversational Networks (Papangelis et al., 2021) are trained via reinforcement learning to generate annotated data from seed examples, studying English IC+ST and other tasks.

The closest relative of LINGUIST is Example Extrapoloation (Ex2) (Lee et al., 2021), which generates annotated IC+ST data using a seq2seq model and provided seed examples. We compare LINGUIST and EX2 in more detail in section 3.3.

A widely used paraphrasing method is **Back-Translation** (BT), i.e. translating text from one language into another "pivot" language, then back again. Bannard and Callison-Burch (2005) extract paraphrases directly from parallel corpora. Sennrich et al. (2016) and Edunov et al. (2018) use BT for Machine Translation and Xie et al. (2020) for data augmentation on classification tasks.

Other approaches directly target the Paraphrasing task: Prakash et al. (2016) learns an LSTM model by supervised training on a paraphrase corpus, whereas Kumar et al. (2020) use an unsupervised denoising task, in both cases only using text and not covering slot labels. Cho et al. (2019) explore Paraphrasing via Semi-Supervised Learning.

A different approach to data augmentation is

token replacement: SeqMix (Zhang et al., 2020) replaces tokens with the nearest neighbor in the embedding space, Dai and Adel (2020) replaces slots with synonyms, or mentions from other instances of the same label. Easy Data Augmentation (Wei and Zou, 2019) includes synonym replacement, random insertion, random swap, and random deletion for classification task. Zheng et al. (2021) benchmark the success of LLMs on few-shot settings.

3 LINGUIST Data Generator Model

LINGUIST provides three key innovations compared to prior data generation work: (1) controls for slot types and values (either user-supplied or model-generated) to include in the outputs; (2) multilingual and cross-lingual generation; and (3) ability to leverage the natural language label names to inform generation, enabling a new "Label Names Only" setting (Section 4.2.2).

3.1 LINGUIST Prompt Design

We control the generation output via a novel prompting scheme, as shown in Figure 1. The prompt contains five blocks: (i) the output `<language>`, (ii) the `<intent>` name, (iii) which slot types and values to `<include>` in the output, (iv) a mapping from `<labels>` to numbers, and (v) up to 10 `<examples>`, each belonging to the same intent, and including zero or more of the available labels.

The `<include>` block instructs LINGUIST which slot types and values to generate, such as `[3 snow ]`, where the number corresponds to the **slot type** (in this case, `[3=condition_description`), and the content inside the brackets is the value to use for that slot (here "snow"). To increase the diversity of outputs, the user may also instruct the model to generate a value for a slot, using the **"wildcard" token**, e.g. `[1 * ]` indicates that the model should sample a value for slot number 1.

As shown in Figure 1, LINGUIST learns to produce a rich sample of values even for intents and slot types it never saw during fine-tuning. For example, in this case the wildcard is for `[1=geographic_poi` and the model outputs sensible values such as "Lake Tahoe", despite this phrase never appearing the fine-tuning data.

3.2 Training the LINGUIST model

We fine-tune a pre-trained seq2seq model on pairs of LINGUIST instruction prompts and corresponding annotated target utterances, derived from a deduplicated IC+ST task dataset R. Specifically, we format an instruction prompt p_i targeting each utterance $t_i \in R$, including in p_i up to 10 *other* example utterances $E = \{e_j\}_{j=1}^{10} \in R$ s.t. $\forall j,\ e_j \neq t_i$ and $\text{intent}(e_j) = \text{intent}(t_i)$. To make the generation robust to the number of provided examples, we do not always include all 10 in the prompt, but instead randomly select k examples from E, with k chosen randomly between 0 and 10, or the number of utterances available that share the same intent as t_i, whichever is smaller. We never duplicate utterances in the prompt. Finally, we produce a corpus of training prompts equal in size to the original IC+ST training set.

To reduce the tendency for the model to overfit on the intent and slot labels (as observed by Lee et al., 2021), we drop out the label names for both at a rate of 0.2, replacing the label name e.g. `GetWeather` with a random sequence of between 1 and 5 letters like `A_Q_Y`. (Ablation in Appendix E.) The intuition for masking the labels rather than skipping the `<intent>` and `<labels>` blocks is to provide the model a consistent signal for position embeddings, and always allowing it to attend to these tags if it wishes to.

To jointly teach the model both to copy user-supplied slot values like `[3 snow ]` and to produce appropriate values for the wildcard `[1 * ]`, we format the training prompts with examples of both. For the prompt p_i targeting utterance t_i, we randomly select from a Geometric distribution $d \sim \text{Geo}(0.5)$ $(0 \leq d \leq \#\text{ slots in } t_i)$ slots and replace their values with `"*"` in the `<include>` block of the prompt. The effect is that approximately 50% of utterances have all slot values replaced by the wildcard token, 25% of utterances keep one slot value, etc.

We do not add the tags `<intent>`, `[1`, etc. to the model's sentencepice (Kudo and Richardson, 2018) tokenizer vocabulary (Appendix C.2).

3.3 Comparing LINGUIST to Ex2

The closest relative to our approach is Example Extrapolation (Ex2, Lee et al., 2021), which also produces slot-labeled text from seed examples. The novelty of LINGUIST compared to Ex2 is threefold: (i) instructions to control the slot types and values

generated, (ii) multilingual and cross-lingual, and (iii) the ability to include label and slot names in the prompt. In particular, EX2 showed that *anonymizing* the labels improved on IC however hurt ST. Our LINGUIST model improves over our implementation of Ex2 on both IC and ST, and furthermore the labels enable LINGUIST to perform one-shot and zero-shot for new intents and slots, as we show in Section 5.1.3 and Appendix B.

4 Experimental Setup

This section describes the datasets, tasks, IC+ST model, baseline gata generation methods, and metrics that we use to evaluate LINGUIST.

4.1 Datasets

4.1.1 SNIPS Dataset

The SNIPS dataset (Coucke et al., 2018) is a public IC+ST benchmark consisting of 7 intents, each with between 2 and 14 slot types (39 unique slot types in total). It includes around 2k training utterances and 100 validation utterances per intent. In order to avoid overfitting our method on the small validation set, at the beginning of our experiments, we partition the training set into 97% Train and 3% Development. We use our Development set split for iterating on all modeling and data processing decisions, including the hyperparameters for LINGUIST and hyperparameters and selection of best checkpoint for the encoder fine-tuning on IC+ST. Only at the very end of our experiments, we evaluate and report on the Validation set. See Table 1 for counts of Train/Dev/Valid utterances.

Intent	Train	Dev.	Valid.
AddToPlaylist	1884	58	100
BookRestaurant	1914	59	100
GetWeather	1940	60	100
PlayMusic	1940	60	100
RateBook	1898	58	100
SearchCreativeWork	1896	58	100
SearchScreeningEvent	1901	58	100
Total	13373	411	700

Table 1: Data counts per intent for SNIPS.

4.1.2 Multilingual ATIS++

For cross-lingual experiments we evaluate on mATIS++ (Xu et al., 2020), which consists of human-translated text and annotations from the original English travel information requests ATIS dataset (Hemphill et al., 1990) plus Hindi and Turkish translations from mATIS (Upadhyay et al., 2018). Our experiments cover the 7 languages that mATIS++ shares with our pretrained model: English, Spanish, German, French, Portuguese, Japanese, and Hindi, with 4488 (HI: 1440) training utterances covering 18 (HI: 17) intents and 84 (HI: 75) slots.

To demonstrate the cross-domain adaptation of the LINGUIST method, we use the **MASSIVE** dataset (FitzGerald et al., 2022b) covering 51 languages with parallel versions of the (English-only) SLU Resource Package (Bastianelli et al., 2020) utterances, covering 20k utterances per language across 18 domains, 60 intents and 55 slots. We use MASSIVE only to train a LINGUIST model, then apply the model to mATIS++. In order to keep mATIS++ as a novel domain, we exclude the somewhat related `transport` domain from MASSIVE when we train the LINGUIST model.

4.1.3 Internal Dataset

To demonstrate the value of our method to a real-world setting, we benchmark on an internal large-scale multilingual dataset representative of requests to a conversational agent. We consider five portions of the dataset, known as *features*, namely: CameraControl, ClockSettings, HomeSecurity, Music, and Timers, each containing one or more intents, and one or more associated slots. For each feature, there is a "starter" training set comprised of a few dozens of annotated utterances which were curated for the new feature, and a test set containing hundreds of annotated utterances pertaining to that new feature. Additionally, there is a large training dataset E of annotated utterances from existing features. *The Existing Features training data E does not contain examples of any of the new features.*

4.2 Evaluation Tasks

4.2.1 New-Intent Few-Shot (NIFS)

As shown in Figure 2, we simulate the introduction of a new intent into an existing well-resourced dataset. Given a training dataset $R = \bigcup_{j=1}^{m} D_j$ containing data D_j for m intents $j = 1 \ldots m$, we select an intent $i \in \{1, \ldots, m\}$, and reduce its training data to only a small number K of "starter" utterances $S_i \subset D_i$. We apply various data augmentation techniques on S_i to create augmented data A_i. Finally, we train an IC+ST model using $R'_i = S_i \cup A_i \cup \{D_j\}_{j \neq i}$, i.e. the concatenation of starter and augmented data for intent i with the unmodified data for all other intents.

The internal dataset is already split in this way, however at the *feature* rather than intent level.

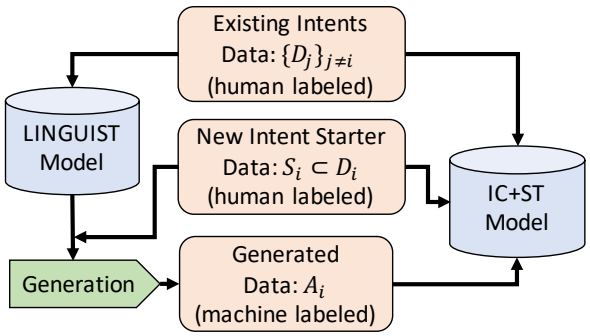

Figure 2: New-Intent Few-Shot (NIFS) Setup.

For SNIPS, we create 7 NIFS settings, one for each intent, reducing its training data down to only K=10 starter utterances S_i. We create 5 versions, each with a different random seed for selecting S_i, and always including at least one example for all slot types that occur for intent i.

To demonstrate the ability of LINGUIST to generalize to new intents and slots at inference time, **we exclude the new intent's starter utterances from fine-tuning**. For each intent i, we train a LINGUIST model on the other 6 intents $\{D_j\}_{j \neq i}$. Then, *during inference*, we forumlate prompts with the starter utterances between `<example>` and `</example>`, and generate more data. Note, this generation step **does not require any model parameter updates**.

4.2.2 NIFS Label Names Only (LNO)

In this more challenging variant of NIFS, *only the intent and slot label names* are available for the starter utterances, not their text or annotation. This is useful when developing new intents as we need only specify which slot types can go together, and need not curate or annotate any real examples. Notably, to the best of our knowledge, **LINGUIST is the first system capable of generating intent- and slot-annotated data in this setting** (as shown in Figure A4, Appendix B.1.4), by attending to the natural language label names in the prompt.

4.2.3 Zero-Shot Cross-Lingual

For mATIS++, we evaluate in the *zero-shot cross-lingual* setting, where real training data is available only for English. We fine-tune an IC+ST model on the English training data plus any augmented examples generated from this data, and evaluate the model on the test sets from all languages.

4.3 Models

Our experiments rely on two pre-trained models: (1) **AlexaTM 5B** (described next in Section 4.3.1) which we fine-tune on the LINGUIST prompts and use it to generate IC+ST training data, and (2) **xlm-roberta-base** which we fine-tune on the IC+ST task including the data generated from LINGUIST.

4.3.1 AlexaTM 5B

AlexaTM 5B is a multilingual seq2seq Transformer (Vaswani et al., 2017) model pre-trained similar to AlexaTM 20B (Soltan et al., 2022), however with denoising objective only (i.e. without Causal Language Modeling objective). Like AlexaTM 20B, the architecture is derived from the HuggingFace (Wolf et al., 2020) BART (Lewis et al., 2020b) class, and consists of 29 encoder and 24 decoder layers, hidden dimension 2560, and 32 attention heads. The model is trained on 900B tokens of Wikipedia and mC4 (Xue et al., 2021) of 12 languages as used in AlexaTM 20B. We used a maximum sequence length of 512 and a batch size of 1M tokens. The encoder weights of AlexaTM 5B are initialized with the 2.3B-parameter Alexa Teacher Model encoder (FitzGerald et al., 2022a) trained on MLM task on the same data, frozen during the first half of the AlexaTM 5B pre-training. Hyperparameters used for fine-tuning AlexaTM 5B on LINGUIST are described in Appendix C.

4.3.2 IC+ST Fine-tuning

For SNIPS and mATIS++, following Chen et al. (2019), we fine-tune a BERT-style model for joint IC+ST. On top of the encoder hidden states, we attach two separate classification heads, one for IC and another for ST. Each head consists of two layers of 256 hidden dimension, with gelu activation, dropout 0.2, and layer norm. The IC head utilizes representation from the first token of the sequence (`[CLS]`), while the ST head utilizes the first subword token of each word.

For our encoder, we use `xlm-roberta-base` (Conneau et al., 2020) (12 layers, 768 hidden dimension), from the HuggingFace (Wolf et al., 2020) implementation. We fine-tune with batch size 128 for 3k updates (i.e. 30 epochs for the full-size data). We freeze the embedding layer; all other parameters are free to update during training. We use Adam (Kingma and Ba, 2015) with peak learning rate 3e-5, increased linearly from 0 to 600 updates, then decayed linearly to 0 until the end of training.

To avoid over-fitting on the official SNIPS Validation dataset, we use our Development split (Section 4.1.1) for early stopping, selecting the checkpoint with best performance on ST. All of our IC+ST fine-tuning runs for SNIPS use identical hyper-parameters, regardless of the data generation method being explored. For each data generaiton method, we train and test 7 different Joint IC+ST models $\{M_i\}_{i=1}^{7}$ in NIFS setting: each using a combination of the modified data for intent i, and the unmodified data for all other intents.

For mATIS++, we follow the same model architecture settings and train for 2k updates (64 epochs for English only data, or 9 epochs when using data from all the 7 languages.) We select the checkpoint with best ST F1 Score on the English dev set only.

For our internal benchmark, we use similar settings, however with a smaller internal Transformer-based encoder for fine-tuning on the IC+ST task.

4.4 Baseline Data Generation Methods

The **Interpretation-Conditioned Language Model** (ICLM) Jolly et al. (2020) generates unlabeled text conditioned on intent and provided slot values, with a separate label projection step to recover the full slot annotation. ICLM does not generate novel slot values. Our implementation uses a small Transformer (Vaswani et al., 2017) architecture with $\sim$ 37M parameters, and a simple character-level Levenshtein distance measure to project the slot labels. We produce 50 outputs per input, then filter/de-duplicate (see Appendix J).

We apply **Back-Translation** (BT) using two separate MT systems to show the influence of the translation model. The first uses the open-source Sockeye toolkit (Hieber et al., 2018) and a small (91M parameters) Transformer which has been fine-tuned on around 10k utterances of annotated parallel data. We use fast_align (Dyer et al., 2013) to project the slot labels to the generated utterances. We call this system **"BT-Small"**. We use M=1 forward and N=10 backward translations to obtain 10 paraphrases, and then filter and deduplicate (see Appendix K). We use French (SNIPS) or English (Internal) respectively as pivot languages.

For a stronger BT baseline "**BT-5B**", we build an MT system by fine-tuning AlexaTM 5B on WMT14 (retrieved from HuggingFace datasets) jointly on en→fr and fr→en using an instruction prompt (prefixing the input text with `Translate to French:` or `Translate to English:`, respectively) to control the translation direction. We use SimAlign (Jalili Sabet et al., 2020) to project the slot labels to the paraphrased text. For SNIPS, we use French as the pivot language, with beam search 10 both forward and backward, producing 100 outputs per original sentence, then filter and de-duplicate the outputs (Appendix L). BT-5B was not available for the Internal Benchmark.

For SNIPS, we implement **Example Extrapolation** (Ex2, Lee et al. (2021)) with the default "fully anonymized labels" setting, again fine-tuning from AlexaTM 5B, training a separate version for each intent's experiment, as described in 4.2.1.

Slot Catalog Resampling is a simple approach to data augmentation which samples entities from a catalog for a particular label. For example, given an utterance like "play jason mraz" we might sample "weeezer" from a catalog of artist names, to get "play weezer". We only use Slot Catalog Resampling for the Internal Benchmark, as there are no slot catalogs available for SNIPS or mATIS++.

4.5 Metrics

4.5.1 Metrics for SNIPS and mATIS++

We use separate metrics to measure (1) support for the new intent, while (2) not harming the overall performance across all intents. For (1), we run the model on a test set containing *only* the new intent. We refer to this as the *Local* Intent Recall (IR), and *Local* ST F1 Score. To measure (2), we run the model on the combined test set of all intents together, and call this the *Global* Intent Accuracy (IA) and *Global* ST F1 Score. In both cases, for ST F1 Score, we ignore the "O" (non-entity) tag, using the seqeval (Nakayama, 2018) implementation.

When training data is modified for a particular intent i, the Local metrics for i change across methods as expected, whereas changes in Global metrics (see Appendix G) are very small for all methods.

For the cross-lingual **mATIS++** experiments, we report (Global) intent accuracy and Slot F1, since we are doing cross-lingual transfer for the whole dataset, and not targeting specific intents.

4.5.2 Metrics for Internal Benchmark

For the internal benchmark, we only evaluate in the *Local* setting. We measure Semantic Error Rate (SemER: Su et al., 2018 or Appendix O) which jointly evaluates the IC and ST performance. Lower SemER indicates improvement to the system. We report relative reduction in SemER, where a negative number indicates improvement.

Modified Intent / Data	Full	s10-NoUps	s10	s10 +ICLM	s10 +BT-Small	s10 +BT-5B	s10 +Ex2	s10 +LINGUIST
AddToPlaylist	100.0	95.6 ±6.8	98.3 ±2.1	97.5 ±2.2	98.3 ±2.4	**99.4** ±0.5	97.5 ±1.6	93.9 ±3.2
BookRestaurant	100.0	91.0 ±3.4	93.5 ±2.4	93.8 ±1.8	92.5 ±1.7	**94.6** ±1.1	91.3 ±5.4	**94.6** ±1.5
GetWeather	100.0	98.8 ±0.4	98.8 ±0.4	99.4 ±0.5	99.6 ±0.5	99.8 ±0.4	99.8 ±0.4	**100.0** ±0.0
PlayMusic	99.0	70.8 ±10.1	77.1 ±6.6	79.8 ±8.4	76.5 ±5.8	84.0 ±2.8	83.8 ±7.5	**90.4** ±4.7
RateBook	100.0	99.0 ±0.0	99.6 ±0.5	**100.0** ±0.0	99.8 ±0.4	99.6 ±0.5	99.8 ±0.4	**100.0** ±0.0
SearchCreativeWork	100.0	69.0 ±8.8	76.9 ±9.3	74.2 ±9.1	73.3 ±13.6	79.4 ±11.3	80.8 ±3.9	**83.3** ±6.9
SearchScreeningEvent	95.2	66.5 ±5.6	73.1 ±8.2	72.1 ±3.7	71.5 ±6.1	73.5 ±10.8	77.3 ±9.1	**81.9** ±3.9
Average	99.2	84.4 ±2.9	88.2 ±1.9	88.1 ±2.4	87.4 ±2.9	90.1 ±1.6	90.0 ±2.4	**92.0** ±0.8

(a) SNIPS New-Intent Few-Shot (NIFS) results on **Local Intent Recall**.

Modified Intent / Data	Full	s10-NoUps	s10	s10 +ICLM	s10 +BT-Small	s10 +BT-5B	s10 +Ex2	s10 +LINGUIST
AddToPlaylist	94.1	76.8 ±2.9	81.2 ±2.5	78.4 ±2.4	**82.0** ±1.8	81.3 ±1.7	80.6 ±3.1	80.9 ±3.4
BookRestaurant	96.4	71.9 ±2.3	81.3 ±2.1	80.4 ±1.4	81.2 ±1.2	83.3 ±2.5	78.6 ±4.5	**83.4** ±1.7
GetWeather	97.8	74.7 ±3.9	84.9 ±5.4	82.9 ±4.8	82.3 ±4.5	84.0 ±2.8	82.9 ±4.2	**85.4** ±2.8
PlayMusic	91.7	42.0 ±4.3	59.2 ±2.2	58.0 ±3.4	56.1 ±3.1	65.4 ±4.2	67.6 ±6.2	**70.1** ±1.8
RateBook	99.7	89.4 ±1.5	95.0 ±0.9	**95.4** ±0.8	93.5 ±3.3	93.6 ±1.5	94.7 ±0.6	94.8 ±1.7
SearchCreativeWork	100.0	56.2 ±10.5	70.9 ±11.3	67.6 ±9.8	68.9 ±11.1	72.9 ±12.1	75.2 ±5.2	**79.3** ±5.0
SearchScreeningEvent	96.6	56.4 ±7.4	71.6 ±3.6	72.8 ±4.6	69.8 ±4.6	74.2 ±3.6	79.0 ±4.3	**82.3** ±3.4
Average	96.6	66.8 ±2.2	77.7 ±2.0	76.5 ±2.1	76.3 ±2.5	79.2 ±2.8	79.8 ±2.1	**82.3** ±1.3

(b) SNIPS New-Intent Few-Shot (NIFS) results on **Local ST F1 Score**.

Table 2: Our main results on SNIPS Validation set (Section 4.1.1). For each cell (i, j), we train a joint IC+ST encoder on the combination of data from intent i modified according to strategy j, and all other intents' data unmodified. "Full" is trained on the full dataset without any modifications; for "s10-NoUps", the data for intent i is reduced to only 10 "starter" examples, and are *Not Up-sampled*; for "s10", the starter utterances are up-sampled to N_i, the original data size for intent i. For the remaining columns, the up-sampled starter utterances for intent i are mixed with augmented data derived from them using a particular method, which is re-sampled to N_i in size. "s10+X" uses ICLM, BT-Small, BT-5B, respectively. "s10+Ex2" uses our internal 5B seq2seq model with Ex2, "s10+LINGUIST" uses data generated by our LINGUIST method. We bold (underline) the mean for the method with best (second best) results. Experiments are run across five random seeds, as mean ± standard deviation.

5 Results

5.1 SNIPS Results

The main results are presented in Table 2a for Local Intent Recall and Table 2b for Local ST F1 Score.

5.1.1 Baseline Results on SNIPS

An upper bound for the New-Intent Few-Shot (NIFS) setting, is a model trained on the full dataset, which we train and report ("Full" in the tables) at 99.2 for Local Intent Recall and 96.6 for Local ST F1 Score. Reducing to 10 utterances ("s10-NoUps") harms both IC and ST, (although ST more substantially), however simply up-sampling (duplicating) the starter utterances ("s10") recovers a sizeable portion of the performance lost.

The rest of the columns use a mix (weighted 0.5/0.5) of the up-sampled 10 starter utterances, plus augmented data derived from them via the specified methods. In all cases, we re-sample the final amount of data for the target intent to match the count in the original unmodified dataset.

We find that ICLM and BT-Small do not improve on Local Intent Recall or Local ST F1 Score compared to "s10", whereas BT-5B is a strong baseline, achieving 90.1 vs 88.2 for IC and 79.2 vs 77.7 for ST. Compared to BT-5B, Ex2 matches for IC at 90.0 and only slightly improves for ST at 79.8.

5.1.2 LINGUIST Results on SNIPS

We train 7 versions of the LINGUIST model one for each heldout intent, as described in section 4.2.1.

Utilizing the ability of LINGUIST to both copy slot values and produce novel values, we format multiple prompt versions p_{ik} from each starter utterance s_i. The first, dubbed "copy-all" instructs LINGUIST to copy all the slot values, while producing new carrier phrases. Note that LINGUIST may also re-order the slots in the sentence.

Then, for each slot type k, we create a new version of the prompt replacing the value for k with the wildcard "$\star$", instructing LINGUIST to produce a new value for the slot, while copying the other slot values as they are, and generating a suitable carrier phrase. We refer to this strategy as "sample-each". We use *top_k* sampling with $k = 50$ and tempera-

ture 0.3 to generate 100 utterances per prompt.

After filtering the generated utterances (see Appendix M for details), we mix the up-sampled 10 starter utterances with the LINGUIST-generated data. We use identical settings for LINGUIST fine-tuning and generation across all 35 runs (7 intents times 5 random seeds) for the SNIPS-NIFS benchmark. Following the setting of the other baselines, we fine-tune the IC+ST model on the concatenation of the augmented and mixed data for intent i with the original data for all other intents.

Compared to Ex2 ("s10+Ex2"), LINGUIST improves by **+2.0 points absolute on Local Intent Recall** (from 90.0 to 92.0), and **+2.5 points absolute on Local ST F1 Score** (from 79.8 to 82.3).

Finally, we show that the improvements in Local metrics for the new intent do not cause harm to the overall system, and in fact provide a small improvement. As shown in Table 15a and Table 15b (Appendix G.1), "s10+LINGUIST" improves upon "s10+Ex2" by +0.3 points absolute on both Global Intent Accuracy and Global Slot F1 Score.

5.1.3 LINGUIST Results on SNIPS (LNO)

We report on the Label Names Only (LNO) setting described in Section 4.2.2. For these results, we used LINGUIST models trained without label name dropout, which we found to perform significantly better (ablation shown in Appendix E).

As show in Table 3 for IC and Table 4 for ST, despite having **zero real examples for the new intents, LINGUIST achieves 80.0 on Local Intent Recall and 56.9 on Local ST F1 Score**. (Global metrics are shown in Appendix G.2.) While this is still far behind using the real text and annotation from these 10 examples ("s10"), it represents significant progress towards true zero-shot development of new intents and slots in IC+ST systems.

Modified Intent / Data	s10	LINGUIST (via s10 LNO)
AddToPlaylist	98.3 ± 2.1	68.6 ± 18.7
BookRestaurant	93.5 ± 2.4	92.1 ± 5.3
GetWeather	98.8 ± 0.4	99.6 ± 0.5
PlayMusic	77.1 ± 6.6	85.4 ± 3.1
RateBook	99.6 ± 0.5	100.0 ± 0.0
SearchCreativeWork	76.9 ± 9.3	66.7 ± 5.8
SearchScreeningEvent	73.1 ± 8.2	47.7 ± 9.0
Average	88.2 ± 1.9	**80.0** ± 2.6

Table 3: Local Intent Recall results on SNIPS in the New Intent Few-Shot Label Names Only (NIFS-LNO) setting. "s10" results are copied from Table 2a.

Modified Intent / Data	s10	LINGUIST (via s10 LNO)
AddToPlaylist	81.2 ± 2.5	45.9 ± 9.1
BookRestaurant	81.3 ± 2.1	74.8 ± 3.5
GetWeather	84.9 ± 5.4	71.2 ± 3.8
PlayMusic	59.2 ± 2.2	55.6 ± 3.1
RateBook	95.0 ± 0.9	55.0 ± 6.3
SearchCreativeWork	70.9 ± 11.3	59.3 ± 5.7
SearchScreeningEvent	71.6 ± 3.6	36.6 ± 6.1
Average	77.7 ± 2.0	**56.9** ± 2.5

Table 4: Local ST F1 Score results on SNIPS in the New Intent Few-Shot Label Names Only (NIFS-LNO) setting. "s10" results are copied from Table 2b.

5.2 mATIS++ Results

Our mATIS++ results are shown in Tables 5 (Intent Accuracy), and 6 (Slot F1). The main focus is **"avg-0S", the average zero-shot performance across the 6 non-en languages** (de, es, fr, hi, ja, pt).

5.2.1 Baseline Results on mATIS++

An upper bound for zero-shot cross-lingual IC+ST is multilingual training, where a model is trained jointly on the real data for all languages, ("all") which achieves 97.17 for IC and 90.72 for ST. Reducing to English only data ("en") harms average zero-shot Intent by 5.0 points, and Slot F1 by 23.6 points. As our baseline, we report the numbers from the best cross-lingual system ("MT+soft-align") in (Xu et al., 2020), which uses a specialized transformer architecture for slot alignments, achieving 94.88 on IC, and 79.84 on ST.

5.2.2 Fine-tuning LINGUIST on MASSIVE

We first fine-tune a LINGUIST model on the MASSIVE dataset, following the process from Section 3.2. We formulate monolingual prompts for each of the 7 languages, and *cross-lingual* prompts from English to the other 6 languages, which is straightforward: for each training utterance in e.g. French, we select up to 10 English training examples that have the same intent, to include in the prompt with the French utterance as the target, setting "French" in the `<language>` block of the prompt.

To demonstrate not only cross-lingual and cross-schema (mATIS++ label names and annotations conventions are different from MASSIVE) but also *cross-domain* transfer of LINGUIST, we exclude the `transport` domain from MASSIVE, as it has some overlap with the travel information domain of mATIS++. We also exclude two other domains for validation early stopping (Appendix C).

5.2.3 LINGUIST Results on mATIS++

Then, for inference on mATIS++, we first create monolingual English prompts, then use a cloud-based MT system to translate the slot values into the target language, set the `<language>` tag in the prompt, and generate 10 annotated utterances. See Figure A5 (Appendix B.2) for an example.

We select the output with lowest perplexity, and use the English IC+ST model to verify its intent, discarding it if the prediction mismatches the intent from the prompt, in which case we simply copy over an English utterance from the same intent, to maintain the class distribution. (See Appendix F.) The final dataset contains N original English examples, and N examples for each other language.

Compared to the "MT+soft-align" method of Xu et al. (2020), LINGUIST is on-par for IC (from 94.88 to 95.06), and **improves ST F1 Score by 4.14 points absolute** (from 79.84 to 83.98). We note that ST, being a structured prediction task, is inherently more challenging than IC, so our ST results are of particular interest. Moreover, the improvement on both IC and ST is particularly large for Japanese, which tends to be challenging for alignment with English, since the two languages having very different linguistic characteristics.

Lang	all	en	en+MT soft-align	en+ LINGUIST
en	98.10	97.77	–	97.77
de	97.32	90.51	96.66	94.08
es	97.21	95.20	97.20	97.10
fr	98.10	93.64	97.49	96.88
hi	95.20	88.62	92.81	94.08
ja	97.86	90.99	88.33	95.38
pt	97.32	93.97	96.78	92.86
avg-0S	97.17	92.16	94.88	**95.06**

Table 5: Results on mATIS++ Intent Accuracy.

Lang	all	en	en+MT soft-align	en+ LINGUIST
en	95.26	95.96	–	95.07
de	94.54	80.15	89.00	84.61
es	88.27	81.24	76.42	86.89
fr	92.69	77.29	79.64	83.83
hi	85.58	62.61	78.56	76.61
ja	92.76	24.52	79.10	86.32
pt	90.49	76.64	76.30	85.63
avg-0S	90.72	67.08	79.84	**83.98**

Table 6: Results on mATIS++ Slot F1.

5.3 Internal Dataset Results

Table 7 shows SemER on our Internal Dataset.

5.3.1 Baseline Results on Internal Dataset

For each feature i we train an IC+ST model M_i combining the Existing Features data E, upsampled starter utterances S_i, augmented utterances A_i produced from S_i via Slot Catalog Resampling, ICLM, and BT-Small. We evaluate on the feature's test set T_i, reporting Local SemER.

5.3.2 LINGUIST Results on Internal Dataset

We fine-tune a single LINGUIST model on instruction prompts formatted from the Existing Features dataset E (Section 4.1.3.), which *does not contain any examples of the new features*. Then, for each feature, i, following a similar procedure described in Section 5.1.2, we format prompts from the starter utterances S_i and apply LINGUIST to generate more data G_i. Finally, we follow the same data mixing, training, and testing procedure for each feature i as in the baseline (Section 5.3.1). As shown in Table 7, LINGUIST results in 7.9% to 25.2% relative SemER reduction across four languages compared to the baseline of combined Catalog Resampling, ICLM, and BT-Small.

Feature/Lang	**de**	**es**	**fr**	**ja**
CameraControl	-	-33.3%	-	-
ClockSettings	-1.6%	-12.3%	+1.8%	-
HomeSecurity	-	-	-	-31.5%
Music	-36.8%	-	-30.6%	-12.3%
Timers	-27.5%	-15.4%	-20.0%	-
Average	**-11.8%**	**-20.8%**	**-7.9%**	**-25.2%**

Table 7: Internal Dataset Results. Each number is relative reduction in SemER, from baseline (combined Slot Catalog Resampling, BT-Small, and ICLM) to LINGUIST. A negative number indicates improvement.

6 Conclusion and Future Work

We introduced LINGUIST, a novel method for annotated data generation, via fine-tuning a large-scale pre-trained multilingual seq2seq model. Our method generalizes to new intents and slots in challenging few-shot, zero-shot, and cross-lingual settings, which we have shown on three datasets.

In future work, we wish to explore ways to improve the generation output, e.g. human-in-the-loop and reinforcement learning. We would also like to include more controls in the prompt such as text style, and explore generation for multi-turn dialogues, and more complex and nested semantics.

Acknowledgements

We thank Christophe Dupuy, Weitong Ruan, and the anonymous reviewers for feedback on our work.

References

Colin Bannard and Chris Callison-Burch. 2005. Paraphrasing with bilingual parallel corpora. In *Proceedings of the 43rd Annual Meeting of the Association for Computational Linguistics (ACL'05)*, pages 597–604, Ann Arbor, Michigan. Association for Computational Linguistics.

Emanuele Bastianelli, Andrea Vanzo, Pawel Swietojanski, and Verena Rieser. 2020. SLURP: A spoken language understanding resource package. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 7252–7262, Online. Association for Computational Linguistics.

Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel Ziegler, Jeffrey Wu, Clemens Winter, Chris Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. 2020. Language models are few-shot learners. In *Advances in Neural Information Processing Systems*, volume 33, pages 1877–1901. Curran Associates, Inc.

Qian Chen, Zhu Zhuo, and Wen Wang. 2019. Bert for joint intent classification and slot filling.

Eunah Cho, He Xie, and William M. Campbell. 2019. Paraphrase generation for semi-supervised learning in NLU. In *Proceedings of the Workshop on Methods for Optimizing and Evaluating Neural Language Generation*, pages 45–54, Minneapolis, Minnesota. Association for Computational Linguistics.

Alexis Conneau, Kartikay Khandelwal, Naman Goyal, Vishrav Chaudhary, Guillaume Wenzek, Francisco Guzmán, Edouard Grave, Myle Ott, Luke Zettlemoyer, and Veselin Stoyanov. 2020. Unsupervised cross-lingual representation learning at scale. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 8440–8451, Online. Association for Computational Linguistics.

Alice Coucke, Alaa Saade, Adrien Ball, Théodore Bluche, Alexandre Caulier, David Leroy, Clément Doumouro, Thibault Gisselbrecht, Francesco Caltagirone, Thibaut Lavril, Maël Primet, and Joseph Dureau. 2018. Snips voice platform: an embedded spoken language understanding system for private-by-design voice interfaces. *CoRR*, abs/1805.10190.

Xiang Dai and Heike Adel. 2020. An analysis of simple data augmentation for named entity recognition. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 3861–3867, Barcelona, Spain (Online). International Committee on Computational Linguistics.

Bosheng Ding, Linlin Liu, Lidong Bing, Canasai Kruengkrai, Thien Hai Nguyen, Shafiq Joty, Luo Si, and Chunyan Miao. 2020. DAGA: Data augmentation with a generation approach for low-resource tagging tasks. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 6045–6057, Online. Association for Computational Linguistics.

Chris Dyer, Victor Chahuneau, and Noah A. Smith. 2013. A simple, fast, and effective reparameterization of IBM model 2. In *Proceedings of the 2013 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 644–648, Atlanta, Georgia. Association for Computational Linguistics.

Sergey Edunov, Myle Ott, Michael Auli, and David Grangier. 2018. Understanding back-translation at scale. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 489–500, Brussels, Belgium. Association for Computational Linguistics.

Angela Fan, Mike Lewis, and Yann Dauphin. 2018. Hierarchical neural story generation.

Jack FitzGerald, Shankar Ananthakrishnan, Konstantine Arkoudas, Davide Bernardi, Abhishek Bhagia, Claudio Delli Bovi, Jin Cao, RAKESH CHADA, Amit Chauhan, Luoxin Chen, Anurag Dwarakanath, Satyam Dwivedi, Turan Gojayev, Karthik Gopalakrishnan, Thomas Gueudre, Dilek Hakkani-Tur, Wael Hamza, Jonathan Hueser, Kevin Martin Jose, Haidar Khan, Beiye Liu, Jianhua Lu, Alessandro Manzotti, Pradeep Natarajan, Karolina Owczarzak, Gokmen Oz, Enrico Palumbo, Charith Peris, Chandana Satya Prakash, Stephen Rawls, Andy Rosenbaum, Anjali Shenoy, Saleh Soltan, Mukund Harakere, Liz Tan, Fabian Triefenbach, Pan Wei, Haiyang Yu, Shuai Zheng, Gokhan Tur, and Prem Natarajan. 2022a. Alexa teacher model: Pretraining and distilling multi-billion-parameter encoders for natural language understanding systems. In *KDD 2022*.

Jack FitzGerald, Christopher Leo Hench, Charith S. Peris, Scott Mackie, Kay Rottmann, A. Patricia Domínguez Sánchez, Aaron Nash, Liam Urbach, Vishesh Kakarala, Richa Singh, Swetha Ranganath, L. Crist, Misha Britan, Wouter Leeuwis, Gokhan Tur, and P. Natarajan. 2022b. Massive: A 1m-example multilingual natural language understanding dataset with 51 typologically-diverse languages. *ArXiv*, abs/2204.08582.

Charles T. Hemphill, John J. Godfrey, and George R. Doddington. 1990. The atis spoken language systems pilot corpus. In *HLT*.

Felix Hieber, Tobias Domhan, Michael Denkowski, David Vilar, Artem Sokolov, Ann Clifton, and Matt Post. 2018. The sockeye neural machine translation toolkit at AMTA 2018. In *Proceedings of the 13th*

Conference of the Association for Machine Translation in the Americas (Volume 1: Research Track), pages 200–207, Boston, MA. Association for Machine Translation in the Americas.

Ari Holtzman, Jan Buys, Li Du, Maxwell Forbes, and Yejin Choi. 2019. The curious case of neural text degeneration.

Masoud Jalili Sabet, Philipp Dufter, François Yvon, and Hinrich Schütze. 2020. SimAlign: High quality word alignments without parallel training data using static and contextualized embeddings. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: Findings*, pages 1627–1643, Online. Association for Computational Linguistics.

Shailza Jolly, Tobias Falke, Caglar Tirkaz, and Daniil Sorokin. 2020. Data-efficient paraphrase generation to bootstrap intent classification and slot labeling for new features in task-oriented dialog systems. In *Proceedings of the 28th International Conference on Computational Linguistics: Industry Track*, pages 10–20, Online. International Committee on Computational Linguistics.

Diederik P. Kingma and Jimmy Ba. 2015. Adam: A method for stochastic optimization. *CoRR*, abs/1412.6980.

Taku Kudo and John Richardson. 2018. SentencePiece: A simple and language independent subword tokenizer and detokenizer for neural text processing. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 66–71, Brussels, Belgium. Association for Computational Linguistics.

Manoj Kumar, Yuval Merhav, Haidar Khan, Rahul Gupta, Anna Rumshisky, and Wael Hamza. 2022. Controlled data generation via insertion operations for NLU. In *Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies: Industry Track*, pages 54–61, Hybrid: Seattle, Washington + Online. Association for Computational Linguistics.

Varun Kumar, Ashutosh Choudhary, and Eunah Cho. 2020. Data augmentation using pre-trained transformer models. In *Proceedings of the 2nd Workshop on Life-long Learning for Spoken Language Systems*, pages 18–26, Suzhou, China. Association for Computational Linguistics.

Kenton Lee, Kelvin Guu, Luheng He, Timothy Dozat, and Hyung Won Chung. 2021. Neural data augmentation via example extrapolation. *ArXiv*, abs/2102.01335.

Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Veselin Stoyanov, and Luke Zettlemoyer. 2020a. Bart: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. In *ACL*.

Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Veselin Stoyanov, and Luke Zettlemoyer. 2020b. BART: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7871–7880, Online. Association for Computational Linguistics.

Yinhan Liu, Jiatao Gu, Naman Goyal, Xian Li, Sergey Edunov, Marjan Ghazvininejad, Mike Lewis, and Luke Zettlemoyer. 2020. Multilingual denoising pre-training for neural machine translation.

Nikolaos Malandrakis, Minmin Shen, Anuj Goyal, Shuyang Gao, Abhishek Sethi, and Angeliki Metallinou. 2019. Controlled text generation for data augmentation in intelligent artificial agents. In *Proceedings of the 3rd Workshop on Neural Generation and Translation*, pages 90–98, Hong Kong. Association for Computational Linguistics.

Hiroki Nakayama. 2018. seqeval: A python framework for sequence labeling evaluation. Software available from https://github.com/chakki-works/seqeval.

Subhadarshi Panda, Caglar Tirkaz, Tobias Falke, and Patrick Lehnen. 2021. Multilingual Paraphrase Generation For Bootstrapping New Features in Task-Oriented Dialog Systems. In *Proceedings of the 3rd Workshop on Natural Language Processing for Conversational AI*, pages 30–39, Online.

Alexandros Papangelis, Karthik Gopalakrishnan, Aishwarya Padmakumar, Seokhwan Kim, Gokhan Tur, and Dilek Hakkani-Tur. 2021. Generative conversational networks.

Aaditya Prakash, Sadid A. Hasan, Kathy Lee, Vivek Datla, Ashequl Qadir, Joey Liu, and Oladimeji Farri. 2016. Neural paraphrase generation with stacked residual LSTM networks. In *Proceedings of COLING 2016, the 26th International Conference on Computational Linguistics: Technical Paper s*, pages 2923–2934, Osaka, Japan. The COLING 2016 Organizing Committee.

Alec Radford, Jeff Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. 2019. Language models are unsupervised multitask learners.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J. Liu. 2020. Exploring the limits of transfer learning with a unified text-to-text transformer. *Journal of Machine Learning Research*, 21(140):1–67.

Jeff Rasley, Samyam Rajbhandari, Olatunji Ruwase, and Yuxiong He. 2020. Deepspeed: System optimizations enable training deep learning models with

over 100 billion parameters. In *Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery Data Mining*, KDD '20, page 3505–3506, New York, NY, USA. Association for Computing Machinery.

Gaurav Sahu, Pau Rodríguez López, Issam H. Laradji, Parmida Atighehchian, David Vázquez, and Dzmitry Bahdanau. 2022. Data augmentation for intent classification with off-the-shelf large language models. In *4th ACL Workshop on NLP for Conversational AI*, volume abs/2204.01959.

Rico Sennrich, Barry Haddow, and Alexandra Birch. 2016. Improving neural machine translation models with monolingual data. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 86–96, Berlin, Germany. Association for Computational Linguistics.

Saleh Soltan, Shankar Ananthakrishnan, Jack G. M. FitzGerald, Rahul Gupta, Wael Hamza, Haidar Khan, Charith S. Peris, Stephen Rawls, Andrew Rosenbaum, Anna Rumshisky, Chandan Prakash, Mukund Sridhar, Fabian Triefenbach, Apurv Verma, Gokhan Tur, and Premkumar Natarajan. 2022. Alexatm 20b: Few-shot learning using a large-scale multilingual seq2seq model. *ArXiv*, abs/2208.01448.

Chengwei Su, Rahul Gupta, Shankar Ananthakrishnan, and Spyridon Matsoukas. 2018. A re-ranker scheme for integrating large scale nlu models. *2018 IEEE Spoken Language Technology Workshop (SLT)*, pages 670–676.

Gokhan Tur and Renato De Mori. 2011. *Spoken Language Understanding: Systems for Extracting Semantic Information from Speech*. Wiley.

Shyam Upadhyay, Manaal Faruqui, Gokhan Tur, Dilek Z. Hakkani-Tur, and Larry Heck. 2018. (almost) zero-shot cross-lingual spoken language understanding. *2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 6034–6038.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Ł ukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc.

Zirui Wang, Adams Wei Yu, Orhan Firat, and Yuan Cao. 2021. Towards zero-label language learning. *CoRR*, abs/2109.09193.

Jason Wei, Maarten Bosma, Vincent Zhao, Kelvin Guu, Adams Wei Yu, Brian Lester, Nan Du, Andrew M. Dai, and Quoc V Le. 2022. Finetuned language models are zero-shot learners. In *International Conference on Learning Representations*.

Jason Wei and Kai Zou. 2019. EDA: Easy data augmentation techniques for boosting performance on text classification tasks. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 6382–6388, Hong Kong, China. Association for Computational Linguistics.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Remi Louf, Morgan Funtowicz, Joe Davison, Sam Shleifer, Patrick von Platen, Clara Ma, Yacine Jernite, Julien Plu, Canwen Xu, Teven Le Scao, Sylvain Gugger, Mariama Drame, Quentin Lhoest, and Alexander Rush. 2020. Transformers: State-of-the-art natural language processing. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 38–45, Online. Association for Computational Linguistics.

Qizhe Xie, Zihang Dai, Eduard Hovy, Thang Luong, and Quoc Le. 2020. Unsupervised data augmentation for consistency training. In *Advances in Neural Information Processing Systems*, volume 33, pages 6256–6268. Curran Associates, Inc.

Weijia Xu, Batool Haider, and Saab Mansour. 2020. End-to-end slot alignment and recognition for cross-lingual nlu.

Linting Xue, Noah Constant, Adam Roberts, Mihir Kale, Rami Al-Rfou, Aditya Siddhant, Aditya Barua, and Colin Raffel. 2021. mT5: A massively multilingual pre-trained text-to-text transformer. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*.

Rongzhi Zhang, Yue Yu, and Chao Zhang. 2020. SeqMix: Augmenting active sequence labeling via sequence mixup. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 8566–8579, Online. Association for Computational Linguistics.

Yanan Zheng, Jing Zhou, Yujie Qian, Ming Ding, Jian Li, Ruslan Salakhutdinov, Jie Tang, Sebastian Ruder, and Zhilin Yang. 2021. Fewnlu: Benchmarking state-of-the-art methods for few-shot natural language understanding.

A Author Contributions

Andy proposed and implemented LINGUIST; performed the experiments on SNIPS, MASSIVE, and mATIS++; fine-tuned LINGUIST for the internal dataset; and pre-trained the Alexa Teacher Model 2.3B encoder to warm-start AlexaTM 5B seq2seq.

Saleh implemented the seq2seq pre-training and fine-tuning code, pre-trained AlexaTM 5B, provided guidance on experiments and modeling choices for LINGUIST, and helped write the paper.

Wael drove the strategic vision for the project, provided deep technical feedback throughout, and implemented the backbone of our codebase.

Yannick and Markus assisted with the internal dataset preparation and evaluation, the baseline methods ICLM and BT-Small for SNIPS, and contributed to the paper write-up and technical review. Yannick also suggested MASSIVE as the dataset for transferring LINGUIST to mATIS++.

B Sample Model Outputs

B.1 English Outputs

This section uses a LINGUIST model from 6 SNIPS intents: `AddToPlaylist`, `BookRestaurant`, `PlayMusic`, `RateBook`, `SearchCreativeWork`, and `SearchScreeningEvent`.

B.1.1 One-shot Novel Ambiguous Intent

We show (Figure A1) the LINGUIST model generating annotated utterances for a new intent not seen in training data, given only a single example. Furthermore, when the example is ambiguous or under-specified, the model can leverage the intent label to generate more relevant outputs.

```
INPUT:
<language> English </language>
<intent> DownloadGame </intent>
<include> [1 zelda ] </include>
<labels> [1=game </labels>
<examples> get [1 zelda ] </examples>

OUTPUTS:

1. I'd like to play [1 zelda ]
2. I want to play [1 zelda ]
3. I want to download [1 zelda ]
4. I need to download [1 zelda ]
5. Where can I download [1 zelda ] ?
```

(a) The prompt with `DownloadGame` intent.

```
INPUT:
<language> English </language>
<intent> PurchaseGame </intent>
<include> [1 zelda ] </include>
<labels> [1=game </labels>
<examples> get [1 zelda ] </examples>

OUTPUTS:

1. I'd like to buy the game [1 zelda ]
2. I'd like to buy [1 zelda ]
3. I want to buy [1 zelda ]
4. Where can I buy [1 zelda ] ?
5. Please buy [1 zelda ]
```

(b) The prompt with `PurchaseGame` intent.

Figure A1: For the same example and slots, LINGUIST can leverage the intent name to generate different outputs.

B.1.2 One-shot Novel Ambiguous Slot

Similarly, (Figure A2) from a single example for a novel slot, LINGUIST can generate novel values, following the label name to disambiguate an ambiguous word such as "bat" (animal or sports equipment).

```
INPUT:
<language> English </language>
<intent> DisplayImage </intent>
<include> [1 * ] </include>
<labels> [1=animal </labels>
<examples>
 show me a picture of a [1 bat ] </examples>

OUTPUTS:

1. I'd like to see a picture of a [1 dog ]
2. Please show me a picture of a [1 deer ] .
3. Show me a photo of a [1 horse ] .
4. Show me a picture of a [1 zebra ]
5. Show me a [1 elephant ] picture .
```

(a) The prompt with `animal` slot.

```
INPUT:
<language> English </language>
<intent> DisplayImage </intent>
<include> [1 * ] </include>
<labels> [1=sports_equipment </labels>
<examples>
 show me a picture of a [1 bat ] </examples>

OUTPUTS:

1. I'd like to see a picture of a [1 volleyball ]
2. Show me a photo of a [1 soccer ball ]
3. Show me a [1 football net ]
4. show me a [1 puck ]
5. Show me a [1 golf club ]
```

(b) The prompt with `sports_equipment` slot.

Figure A2: For the same example and slots, LINGUIST can leverage the slot name to generate different outputs.

B.1.3 Zero-shot Novel Intent and Slot

We show (Figure A3) that LINGUIST can also generate in a **zero-shot** setting, prompted only on the intent and slot names, without any full text examples. By changing the slot type from `baseball_team_name` to `hockey_team_name`, the model can generate for the requested sports team. Remarkbly, although the model never saw utterances about sports during fine-tuning, it can accuratey sample correct team names, presumably relying on the world knowledge it acquired during pre-training. In the outputs, all baseball teams are actual MLB, and the hockey team names are NHL, with one exception of `the Wildcats` being basketball. The slot `year` appeared in the SNIPS fine-tuning data, so the model may be using values either from world knowledge or from other SNIPS intents.

```
INPUT:
<language> English </language>
<intent> DraftPlayer </intent>
<include>
 [1 * ] , [2 Patrice Bergeron ] , [3 * ]
</include>
<labels>
 [1=baseball_team_name , [2=player, [3=year
</labels>
<examples> </examples>

OUTPUTS:

1. I want to draft [2 Patrice Bergeron ] from
   [1 Houston Astros ] for the [3 20th season ].

2. I want to draft [2 Patrice Bergeron ] for the
   [1 Washington Nationals ] in [3 2008 ].

3. I'd like to draft [2 Patrice Bergeron ] from
   [1 the Boston Red Sox ] in
   [3 2019 ]

4. I want to draft [2 Patrice Bergeron ] for
   [1 San Francisco Giants ] in [3 2007 ]

5. I need to draft [2 Patrice Bergeron ] from
   [1 Toronto Blue Jays ] for [3 2016 ]
```

(a) The prompt with `baseball_team_name` intent.

```
INPUT:
<language> English </language>
<intent> DraftPlayer </intent>
<include>
 [1 * ] , [2 Patrice Bergeron ] , [3 * ]
</include>
<labels>
 [1=hockey_team_name , [2=player, [3=year
</labels>
<examples> </examples>

OUTPUTS:

1. I would like to draft [2 Patrice Bergeron ] from
   [1 Vancouver Canucks ] in [3 2015 ]

2. I want to draft [2 Patrice Bergeron ] from
   [1 New Jersey Devils ] in [3 2023 ]

3. Draft the [2 Patrice Bergeron ] from
   [1 the St. Louis Blues ] for the upcoming
   [3 2013 ] season.

4. I want to add [2 Patrice Bergeron ] to
   [1 the Wildcats ] roster for [3 2012 ].

5. Draft [2 Patrice Bergeron ] for
   [1 Boston Bruins ] in [3 2014 ]
```

(b) The prompt with `hockey_team_name` intent.

Figure A3: LINGUIST can perform zero-shot generation based only on the intent and slot names, with no full text example. It can also leverage world knowledge from pre-training to generate novel slot values that incorporate world knowledge such as valid team names for professional sports.

B.1.4 Label Names Only

Finally, as shown in Figure A4, the most ambiguous generation scenario we explore is "Label Names Only" (introduced in Section 4.2.2), where the model sees no examples, and all slot values are open-ended via the wildcard instruction `*`. Remarkably, LINGUIST can still generate useful outputs in this case, although with some more noise: for example, the text of Output 2 does not correspond to GetWeather intent, and the word `time` is mistakenly tagged as a slot value `timeRange`. Nevertheless, as shown in Section 5.1.3, training with LINGUIST-generated data in this setting can achieve reasonable accuracy, representing significant progress towards true zero-shot data generation for novel intents and slots in IC+ST systems.

B.2 Cross-Lingual Novel Intent and Slots

In Figure A5 we show an example of LINGUIST performing **few-shot cross-lingual annotated data generation on a novel domain**, where it sees only 7 examples in English, and can generate diverse, fluent, and correctly annotated outputs in French.

The model here was trained on MASSIVE both monolingual and cross-lingual prompts, as described in Section 5.2.2. We discarded three of the 18 MASSIVE domains during training, namely `audio` and `cooking` to use as validation sets, and `transport`, in order to keep mATIS++, which covers travel information, as a novel domain.

```
INPUT:
<language> English </language>
<intent> GetWeather </intent>
<include>
 [1 * ] , [2 * ] , [3 * ]
</include>
<labels>
 [1=city , [2=timeRange , [3=state
</labels>
<examples> </examples>

OUTPUTS:

1. What is the weather like for [1 Oyster Bay ] [3 AK ] at [2 midnight ]

2. What is the [2 time ] for [1 Red Bank ], [3 West Virginia ]

3. What is the weather like in [1 North Lake Tahoe ] [3 North Dakota ]
   at [2 1 minute from now ]
```

Figure A4: LINGUIST generation in Label Names Only (LNO) setting, where there are no examples or slot values, and the model must rely entirely on the label names to generate outputs.

```
INPUT:
<language> French </language>
<domain> travelinfo </domain>
<intent> flight </intent>
<include> [2 lundi ] , [3 baltimore ] , [4 dallas ] </include>
<labels>
 [1=depart_time.period_of_day , [2=depart_date.day_name ,
 [3=toloc.city_name , [4=fromloc.city_name
</labels>
<examples>
     show me the flights from [4 atlanta ] to [3 denver ] on [2 friday ]
<br> what flights go from [4 charlotte ] to [3 baltimore ] in the [1 morning ]
<br> show me all the flights from [4 philadelphia ] to [3 cincinnati ]
<br> flights on [2 friday ] from [4 newark ] to [3 tampa ]
<br> what flights leave from [4 chicago ] to [3 seattle ] on [2 saturday ]
<br> list flights from [4 detroit ] to [3 san diego ] on [2 thursday ]
<br> [4 dallas ] to [3 baltimore ] [2 monday ]
</examples>

OUTPUTS:

1. combien de vols y a-t-il entre [4 dallas ] et [3 baltimore ] le [2 lundi ]

2. quels vols y a-t-il [2 lundi ] entre [4 dallas ] et [3 baltimore ]

3. quel est le meilleur vol pour [2 lundi ] entre [4 dallas ] et [3 baltimore ]

4. s'il vous plait trouvez les vols partant de [4 dallas ] le [2 lundi ]
   à destination de [3 baltimore ]

5. quelles sont les vols qui partent le [2 lundi ] de [4 dallas ]
   à [3 baltimore ]
```

Figure A5: LINGUIST generating on a Cross-lingual English to French example from mATIS++.

C LINGUIST Training Details

C.1 Hyperparamters and Early Stopping

As shown in Figure A6, we find that the model converges after a very small number of updates. Specifically, we train with batch size 512 for 400 updates (i.e. around 18 epochs for SNIPS, around 2.5 epochs for MASSIVE), using a very small learning rate 5e-7 with Adam (Kingma and Ba, 2015), warmed up over the first 100 updates, then kept constant for the rest of training. (The internal dataset is much larger, so for that, we train for 4k updates instead, and use a larger learning rate of 1e-6.)

For MASSIVE, we removed two additional small domains `audio` and `cooking` from training and early stop once Token Accuracy plateaus on these domains, which occurs around 400 updates. The Token Accuracy is the percentage of subword tokens for which the model's top-1 hypothesis matches the ground truth (higher is better). It requires only a single forward pass to compute, without needing auto-regressive decoding. In early experiments, we found Token Accuracy to be more reliable than perplexity at predicting downstream performance.

As shown in Figure A6, the Token Accuracy continues to improve for the domains seen during training, however plateaus after 400 updates for the two novel domains, suggesting overfitting beyond 400 updates. The token accuracy is similar for same-language (left) and cross-lingual (right) prompts, suggesting that the model can jointly learn both tasks to a similar level of performance.

For the SNIPS runs, the data is so limited to only 6 intents per run, so removing another intent to check for early stopping could harm the model performance. Therefore, we simply use 400 updates again, as that worked for MASSIVE.

We use DeepSpeed (Rasley et al., 2020) ZeRO Stage 2 to accelerate training.

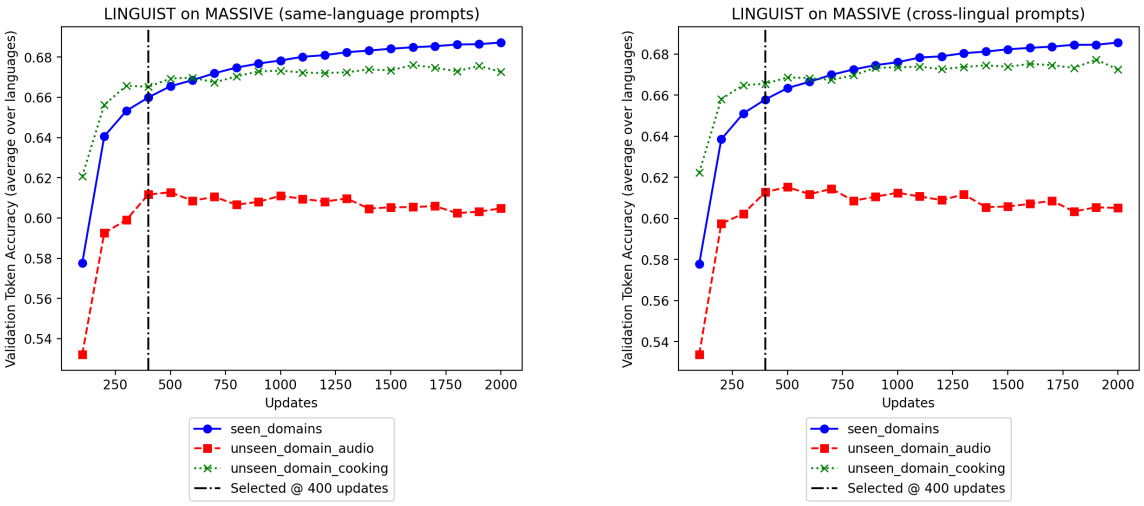

(a) Validation Token Accuracy on same-language prompts.

(b) Validation Token Accuracy on cross-lingual prompts.

Figure A6: Validation Accuracy across updates of fine-tuning LINGUIST on the MASSIVE dataset.

C.2 Tokenizer Choices

As mentioned in Section 3.2, we keep the original sentencepice (Kudo and Richardson, 2018) tokenizer of the model; we do not explicitly add vocabulary items for `<intent>`, `[1`, etc. This choice is motivated by two intuitions:

(1) We hypothesize that these tags' resemblance to markup languages like HTML/XML may help the model learn that they are instructions rather than content words, by relying on data seen during pre-training which was formatted similarly. An earlier version of the LINGUIST prompt had only the open tags such as `<intent PlayMusic>`, and we found that the pre-trained model produced matching closing tags such as `</intent>` before any fine-tuning, suggesting it had learned some knowledge of the tag structure from pre-training.

(2) We hypothesize that the model may be able to generalize at inference time to utterances with a larger number of slots than seen during fine-tuning, by tokenizing the numbered brackets. For example, the largest numbered bracket seen when fine-tuning on MASSIVE is `[10`, tokenized as `['_[', '10']`. For inference on mATIS++, 47% of the prompts contain numbered brackets between `[11` and `[24`. If we had added vocabulary items for these, they would be stuck as randomly initialized tokens at inference time, and therefore unlikely to produce high quality generation. Instead, we see in the generated outputs many examples where the model handles these larger numbered brackets without a problem.

D Impact of Model Size

To evaluate the impact of the model size used for LINGUIST data generation, we evaluate on mATIS++ with a 10x smaller model, following the same procedure of first fine-tuning on MASSIVE (Section 5.2.2), then running cross-lingual inference on mATIS++ (Section 5.2.3).

We use **AlexaTM-Large 500M**, which is trained using the same data as AlexaTM 20B (Soltan et al., 2022) and AlexaTM 5B (Section 4.3.1). Like AlexaTM 5B, AlexaTM-Large 500M uses only the denoising objective (no Causal Language Modeling). The architecture contains 12 encoder and 12 decoder layers, with hidden size 1024, the same as (m)BART (Lewis et al., 2020b; Liu et al., 2020).

As show in Tables 8 for IC and 9 for ST, switching to the smaller model loses 1.62 points on IC (from 95.06 to 93.44), and 1.74 points on ST (from 83.98 to 82.24). While LINGUIST with the smaller model under-performs the baseline "en+MT soft-align" on IC by 1.44 points (93.44 compared to 94.88), it still out-performs on ST by 2.40 points (82.24 compared to 79.84), showing the value of the LINGUIST on the more challenging ST task, even with a smaller model.

Lang	all	en	en+MT soft-align	en+ LINGUIST	en+ LINGUIST (AlexaTM-Large 500M)
en	98.10	97.77	–	97.77	97.88
de	97.32	90.51	96.66	94.08	95.65
es	97.21	95.20	97.2	97.10	95.87
fr	98.10	93.64	97.49	96.88	95.98
hi	95.20	88.62	92.81	94.08	86.94
ja	97.86	90.99	88.33	95.38	91.55
pt	97.32	93.97	96.78	92.86	94.64
avg-0S	97.17	92.16	94.88	**95.06**	93.44

Table 8: Results on mATIS++ Intent Accuracy, showing impact of model size. All except "en+LINGUIST (AlexaTM-Large 500M)" are copied from Table 5.

Lang	all	en	en+MT soft-align	en+ LINGUIST	en+ LINGUIST (AlexaTM-Large 500M)
en	95.26	95.96	–	95.07	95.51
de	94.54	80.15	89	84.61	83.02
es	88.27	81.24	76.42	86.89	85.45
fr	92.69	77.29	79.64	83.83	82.19
hi	85.58	62.61	78.56	76.61	75.36
ja	92.76	24.52	79.1	86.32	83.23
pt	90.49	76.64	76.3	85.63	84.18
avg-0S	90.72	67.08	79.84	**83.98**	82.24

Table 9: Results on mATIS++ Slot F1, showing impact of model size. All except "en+LINGUIST (AlexaTM-Large 500M)" are copied from Table 6.

E Ablation Study on Label Name Dropout

In early experiments, we found Label Name Dropout (LNDrop, described in Section 3.2) helped reduce the model's tendency to overfit on the label names seen during training. However, at the end of our

experiments, the ablation study in this section (Table 10 for Local IC Recall and Table 11 for Local ST F1 Score) shows an improvement from removing the label dropout. The impact is small on "s10" (where the model sees both the label names and the annotated text for the 10 starter examples), however is quite larger in "LINGUIST (via s10 LNO)" (i.e. Label Names Only, Section 4.2.2), where the model sees only the label names.

We hypothesize that other changes we made along the way such as reducing the number of model updates via early stopping (Section C.1) and reducing the learning rate may have introduced regularization which makes label name dropout less necessary, and may surface a side effect of adding undesirable noise. In future work, we would like to study this more thoroughly, and investigate whether label dropout helps in cases where at inference time, the label names are either not available, or are not descriptive of the utterance semantics.

Modified Intent / Data	s10 +LINGUIST LNDrop: yes	s10 +LINGUIST LNDrop: **no**	LINGUIST (via s10 LNO) LNDrop: yes	LINGUIST (via s10 LNO) LNDrop: **no**
AddToPlaylist	93.9 ± 3.2	92.3 ± 7.8	20.0 ± 11.6	68.6 ± 18.7
BookRestaurant	94.6 ± 1.5	93.8 ± 2.3	86.5 ± 4.9	92.1 ± 5.3
GetWeather	100.0 ± 0.0	99.6 ± 0.5	99.2 ± 0.8	99.6 ± 0.5
PlayMusic	90.4 ± 4.7	90.0 ± 3.1	76.2 ± 6.0	85.4 ± 3.1
RateBook	100.0 ± 0.0	99.8 ± 0.4	99.6 ± 0.5	100.0 ± 0.0
SearchCreativeWork	83.3 ± 6.9	92.5 ± 4.0	66.1 ± 13.4	66.7 ± 5.8
SearchScreeningEvent	81.9 ± 3.9	79.4 ± 2.8	44.2 ± 6.7	47.7 ± 9.0
Average	92.0 ± 0.8	**92.5** ± 1.5	70.3 ± 2.6	**80.0** ± 2.6

Table 10: Local Intent Recall results on SNIPS comparing with and without Label Name Dropout (LNDrop), in settings NIFS vanilla (left two columns) and NIFS-LNO (Label Names Only, Section 4.2.2) (right two columns). "s10+LINGUIST (LNDrop: yes)" results are copied from Table 2a and "LINGUIST (via s10 LNO) LNDrop: no" results are copied from Table 3.

Modified Intent / Data	s10 +LINGUIST LNDrop: yes	s10 +LINGUIST LNDrop: **no**	LINGUIST (via s10 LNO) LNDrop: yes	LINGUIST (via s10 LNO) LNDrop: **no**
AddToPlaylist	80.9 ± 3.4	80.4 ± 2.1	56.6 ± 8.0	45.9 ± 9.1
BookRestaurant	83.4 ± 1.7	82.8 ± 4.0	70.7 ± 3.5	74.8 ± 3.5
GetWeather	85.4 ± 2.8	83.1 ± 3.6	70.0 ± 3.4	71.2 ± 3.8
PlayMusic	70.1 ± 1.8	69.3 ± 2.5	54.2 ± 3.2	55.6 ± 3.1
RateBook	94.8 ± 1.7	96.1 ± 1.0	51.0 ± 11.1	55.0 ± 6.3
SearchCreativeWork	79.3 ± 5.0	85.3 ± 3.7	55.5 ± 11.2	59.3 ± 5.7
SearchScreeningEvent	82.3 ± 3.4	80.4 ± 1.4	29.7 ± 2.9	36.6 ± 6.1
Average	82.3 ± 1.3	**82.5** ± 1.7	55.4 ± 2.3	**56.9** ± 2.5

Table 11: Local ST F1 Score results on SNIPS comparing with and without Label Name Dropout (LNDrop), in settings NIFS vanilla (left two columns) and NIFS-LNO (Label Names Only, Section 4.2.2) (right two columns). "s10+LINGUIST (LNDrop: yes)" results are copied from Table 2b and "LINGUIST (via s10 LNO) LNDrop: no" results are copied from Table 4.

F Ablation Study on Filtering Generated Annotated Utterances for mATIS++

We present an ablation study on filtering the outputs of LINGUIST for mATIS++, with results presented in Table 13 for Intent Accuracy, and Table 14 for Slot F1.

F.1 Filtering Methods

As introduced in Section 5.2.3, for each annotated English utterance in mATIS++, we generate 10 annotated utterances in each of the zero-shot languages (German, Spanish, French, Hindi, Japanese, Portuguese). Then, for each language, we select the single[1] generated annotated utterance with lowest perplexity which also passes `Valid-Filter`, described next.

[1]In future work, we would like to evaluate the impact on mATIS++ of including more than one output per input prompt, as we have done for SNIPS and for the Internal Dataset.

We apply two filtering methods, (1) `Valid-Filter`, and (2) `English-IC-Filter`, and report the "Pass Rate" of each in Table 12, as the portion of utterances that pass the filter.

For `Valid-Filter` we discard utterances that have invalid brackets like `[2 [ ]`, or do not respect the prompt, by either generating too few or too many slots, or not copying the value when requested. The Pass Rate for `Valid-Filter` filter is 71.5%, averaged across the languages.

For `English-IC-Filter`, we classify the intent of the generated utterance's text, using the English-only IC+ST model, and discard the utterance if the predicted intent disagrees with the intent from the LINGUIST prompt. We note that the English-only IC+ST model ("en" in Table 13) already performs quite well on IC for other languages, achieving 92.16 Intent Accuracy, so we expect it to contain a strong signal to filter out noisy generated utterances. The Pass Rate for `English-IC-Filter` is 83.8%, suggesting that the remaining 16.2% of the utterances are likely to correspond to an intent other than what was requested in the prompt, which we discuss further below (Section F.3).

After cascading the two filters, the overall Pass Rate is 59.9%.

As a final step, observing that some intents may have lost more data than others, we apply a simple fix which we call `Balance-Classes` to recover the original per-intent class distribution: we simply copy over English utterances from the intents that lack enough data.

Lang	`Valid-Filter` Pass Rate	`English-IC-Filter` Pass Rate	Cascaded Pass Rate	Num Outputs from LINGUIST	Num Copied from English	Total
de	73.8	77.1	56.9	2393	1816	4209
es	77.2	88.2	68.1	2867	1342	4209
fr	77.3	80.2	62.0	2609	1600	4209
hi	75.4	85.6	64.6	2717	1492	4209
ja	50.5	84.8	42.8	1801	2408	4209
pt	74.7	87.2	65.2	2744	1465	4209
avg	71.5	83.8	59.9	2522	1687	4209

Table 12: Pass Rate of LINGUIST generated utterances for mATIS++

F.2 Impact of Filtering

The impact of filtering is presented in Table 13 for Intent Accuracy, and Table 14 for Slot F1, where our main result is "avg-0S", the average of the zero-shot languages (de, es, fr, hi, ja, pt). We observe that `English-IC-Filter` improves IC by 0.89 points absolute (from 93.09 to 93.98) and `Balance-Classes` improves IC by a further 1.08 points absolute (from 93.98 to 95.06), with both methods having minimal impact on Slot F1.

Lang	all	en	en+MT soft-align	en+ LINGUIST (NoFilter)	en+ LINGUIST (+English-IC-Filter)	en+ LINGUIST (+English-IC-Filter) (+Balance-Classes)
en	98.10	97.77	–	97.21	97.77	97.77
de	97.32	90.51	96.66	94.31	93.64	94.08
es	97.21	95.20	97.2	94.75	96.88	97.10
fr	98.10	93.64	97.49	93.97	96.43	96.88
hi	95.20	88.62	92.81	88.17	92.19	94.08
ja	97.86	90.99	88.33	91.78	91.44	95.38
pt	97.32	93.97	96.78	95.54	93.30	92.86
avg-0S	97.17	92.16	94.88	93.09	93.98	**95.06**

Table 13: Intent Accuracy results on ablation study of Filtering for mATIS++.

F.3 Intent Mismatch Discussion

We discuss an intuition about why LINGUIST in a cross-domain setting such as MASSIVE to mATIS++ might produce outputs that do not exactly match the prompted intent. Notice that the prompt contains only 10 examples of the *target* intent, and no examples of *other* intents from the new domain. For example,

Lang	all	en	en+MT soft-align	en+ LINGUIST (NoFilter)	en+ LINGUIST (+English-IC-Filter)	en+ LINGUIST (+English-IC-Filter) (+Balance-Classes)
en	95.26	95.96	–	94.16	95.01	95.07
de	94.54	80.15	89	83.40	85.26	84.61
es	88.27	81.24	76.42	85.92	85.47	86.89
fr	92.69	77.29	79.64	84.65	84.71	83.83
hi	85.58	62.61	78.56	78.35	75.89	76.61
ja	92.76	24.52	79.1	85.72	85.38	86.32
pt	90.49	76.64	76.3	84.45	85.06	85.63
avg-0S	90.72	67.08	79.84	83.75	83.63	**83.98**

Table 14: Slot F1 results on ablation study of Filtering for mATIS++.

mATIS++ contains several closely related intents, such as "flight" which asks to list and book flights, "flight_time" which asks about the *time* of a flight, and "airfare" which asks about the *price* of a flight. We find that when the prompts contain only "flight" examples, the model tends to over-generalize and produce some requests asking for time or price instead, which harms IC, necessitating a post-processing method such as `English-IC-Filter`. In future work, we plan to explore methods to incorporate few-shot data from *other* intents in the domain while generating for the target intent, to mitigate this problem.

G SNIPS Results on Global Metrics

We report the resutls on SNIPS Global Metrics as mentioned in Section 5.1.

G.1 SNIPS Results on Global Metrics: NIFS

Our main results are on the New-Intent Few-Shot setting (NIFS, described in Section 4.2.1). Table 15a shows Global Intent Accuracy, and Table 15b shows Global ST F1 Score. These correspond to the Local metrics shown in Tables 2a for Local IC and 2b for Local ST.

As all our methods target a new-intent setting, as expected, they do not substantially impact the Global metrics. Nonetheless, LINGUIST does provide a small improvement of +0.3 points absolute on both IC and ST compared to Ex2.

Modified Intent / Data	Full	s10-NoUps	s10	s10 +ICLM	s10 +BT-Small	s10 +BT-5B	s10 +Ex2	s10 +**LINGUIST**
AddToPlaylist	99.1	98.7 ±1.0	98.9 ±0.3	98.9 ±0.3	98.9 ±0.3	99.1 ±0.1	99.0 ±0.3	98.4 ±0.6
BookRestaurant	99.1	97.7 ±0.4	98.2 ±0.3	98.2 ±0.2	97.9 ±0.2	98.2 ±0.1	97.9 ±0.8	98.2 ±0.3
GetWeather	99.1	98.9 ±0.1	98.9 ±0.1	98.9 ±0.2	99.0 ±0.0	99.0 ±0.1	99.1 ±0.2	99.1 ±0.1
PlayMusic	99.1	95.3 ±1.5	96.1 ±1.0	96.4 ±1.3	96.0 ±0.8	97.0 ±0.4	97.1 ±1.1	97.9 ±0.6
RateBook	99.1	99.0 ±0.1	98.9 ±0.1	99.0 ±0.1	99.0 ±0.1	99.0 ±0.1	99.0 ±0.0	99.0 ±0.1
SearchCreativeWork	99.1	95.2 ±1.1	96.5 ±1.2	96.0 ±1.2	95.8 ±1.8	96.7 ±1.4	96.7 ±0.8	97.3 ±1.0
SearchScreeningEvent	99.1	95.3 ±0.8	96.3 ±1.1	95.9 ±0.5	95.8 ±0.8	96.2 ±1.5	96.8 ±1.2	97.4 ±0.5
Average	99.1	97.2 ±0.4	97.7 ±0.2	97.6 ±0.4	97.5 ±0.3	97.9 ±0.2	97.9 ±0.3	**98.2** ±0.1

(a) SNIPS New-Intent few-shot results on **Global Intent Accuracy**.

Modified Intent / Data	Full	s10-NoUps	s10	s10 +ICLM	s10 +BT-Small	s10 +BT-5B	s10 +Ex2	s10 +**LINGUIST**
AddToPlaylist	96.7	94.0 ±0.4	94.2 ±0.5	94.2 ±0.4	94.9 ±0.4	94.5 ±0.3	94.7 ±0.4	94.6 ±0.6
BookRestaurant	96.7	92.7 ±0.4	94.1 ±0.6	94.1 ±0.3	94.2 ±0.3	94.5 ±0.3	93.9 ±1.0	94.6 ±0.4
GetWeather	96.7	93.8 ±0.5	94.9 ±0.7	94.7 ±0.6	94.6 ±0.4	95.0 ±0.3	94.7 ±0.7	95.0 ±0.4
PlayMusic	96.7	91.3 ±0.4	92.9 ±0.3	93.1 ±0.6	92.8 ±0.2	93.8 ±0.4	93.9 ±0.6	94.2 ±0.2
RateBook	96.7	95.0 ±0.3	95.8 ±0.4	95.8 ±0.3	95.5 ±0.6	95.5 ±0.4	96.0 ±0.2	96.1 ±0.4
SearchCreativeWork	96.7	92.9 ±1.0	94.2 ±0.9	93.9 ±1.0	94.2 ±1.0	94.1 ±1.1	94.6 ±0.8	95.0 ±0.5
SearchScreeningEvent	96.7	92.3 ±0.8	94.0 ±0.6	94.0 ±0.6	93.5 ±0.6	94.2 ±0.4	94.8 ±0.7	95.3 ±0.3
Average	96.7	93.1 ±0.2	94.3 ±0.2	94.2 ±0.2	94.3 ±0.3	94.5 ±0.3	94.7 ±0.3	**95.0** ±0.2

(b) SNIPS New-Intent few-shot results on **Global ST F1 Score**.

Table 15: Our results on SNIPS for the Global metrics, showing that the gains for Local metrics shown in Tables 2a and 2b do not cause harm to the system overall. See Section 5.1 for details.

G.2 SNIPS Results on Global Metrics: NIFS Label Names Only (LNO)

We show Global metrics for SNIPS in the NIFS-LNO setting (New-Intent Few-Shot Label Names Only, Section 4.2.2) in Tables 16a and 16b for Intent Accuracy and Slot F1 Score, respectively. These correspond to the Local metrics show in Tables 3 for Local IC and 4 for Local ST. The numbers for "LINGUIST (via s10 LNO)" are only a few points behind "s10", indicating that even when no real data is available for the novel intent, LINGUIST generated data can provide some support for the new intent, bringing the overall system performance close to where it would be with 10 real examples for that new intent.

Modified Intent / Data	s10	LINGUIST (via s10 LNO)
AddToPlaylist	98.9 ± 0.3	94.5 ± 2.8
BookRestaurant	98.2 ± 0.3	98.0 ± 0.7
GetWeather	98.9 ± 0.1	99.1 ± 0.1
PlayMusic	96.1 ± 1.0	97.3 ± 0.5
RateBook	98.9 ± 0.1	98.9 ± 0.1
SearchCreativeWork	96.5 ± 1.2	94.8 ± 0.8
SearchScreeningEvent	96.3 ± 1.1	92.6 ± 1.3
Average	97.7 ± 0.2	96.5 ± 0.4

(a) SNIPS NIFS-LNO results on **Global Intent Accuracy**.

Modified Intent / Data	s10	LINGUIST (via s10 LNO)
AddToPlaylist	94.2 ± 0.5	88.8 ± 1.9
BookRestaurant	94.1 ± 0.6	93.1 ± 0.9
GetWeather	94.9 ± 0.7	93.1 ± 0.6
PlayMusic	92.9 ± 0.3	92.6 ± 0.5
RateBook	95.8 ± 0.4	88.1 ± 1.1
SearchCreativeWork	94.2 ± 0.9	93.1 ± 0.5
SearchScreeningEvent	94.0 ± 0.6	90.0 ± 0.8
Average	94.3 ± 0.2	91.2 ± 0.4

(b) SNIPS NIFS-LNO results on **Global Slot F1 Score**.

Table 16: Global metrics (Intent Accuracy, (a), left; Slot F1 Score, (b), right) for SNIPS in the NIFS-LNO setting (New-Intent Few-Shot Label Names Only, Section 4.2.2). The numbers for "s10" are copied from Tables 15a and 15b, for IC and ST, respecitvely.

H Intent Bleeding Case Study

As described in Ex2 (Lee et al., 2021), we also observed intent "bleeding", where the model would produce outputs like one of the fine-tuning intents, despite the prompt and examples being from a novel intent. We noticed this particularly strongly when generating for `AddToPlaylist` intent on SNIPS, where the model had a strong tendency to return utterances starting with "play", which overlaps with the closely related `PlayMusicIntent` from fine-tuning. Consequently, this harmed IC results (Table 2a) for this intent. A very simple fix is to use "n-gram blocking", where the model is prevented from generating phrases like "play", "I want to play", etc. during generation. We found that this mitigates the issue, and we can get near 100% on Intent Recall for AddToPlaylist. However custom designing which n-grams to block requires effort from human experts, and does not scale to a large number of new intents and languages, so in future work, we would like to explore more automated and scalable solutions.

I Generation Hyperparameters

For all three datasets, we use top-k sampling (Fan et al., 2018). For SNIPS, we use top_k=50, temperature=0.3, and produce 100 outputs per input. For mATIS++, we use top_k=50, temperature=0.3, and produce 10 outputs per input. For the internal dataset, we top_k=20, temperature=1.0, and produce 20 outputs per input. We observe that when LINGUIST is trained on the much larger internal dataset compared to SNIPS, it produces less noisy outputs. Thus, we allow a higher temperature of 1.0. We use the same settings for all intents.

We also benchmarked beam search and nucleus sampling (Holtzman et al., 2019) for generation, and found both to perform worse overall on the internal datasets and on SNIPS compared to top-k sampling.

J Filtering ICLM Outputs

We discard any outputs containing the `<unk>` token, which happens less than 1% of the time. The number of outputs (after de-duplication) are reported in Table 17.

K Filtering BT-Small Outputs

The small model has a fair amount of noise in its outputs, so we heuristically filter them, discarding any which contain repeated bigrams such as `play the song halo the song` and/or any trigram of

Modified Intent	Num outputs
AddToPlaylist	296
BookRestaurant	347
GetWeather	322
PlayMusic	255
RateBook	288
SearchCreativeWork	295
SearchScreeningEvent	273
Average	297

Table 17: The number of filtered and de-duplicated outputs from ICLM per intent. All numbers are averaged across the five random seeds.

the same word such as `of of of`. Success rate and number of outputs (after de-duplication) are reported in Table 18.

Modified Intent	SuccessRate	NumOutputs	AvgNumSlots
AddToPlaylist	70.2	64	2.7
BookRestaurant	72.8	73	3.2
GetWeather	60.4	60	2.3
PlayMusic	53.6	52	2.2
RateBook	70.8	71	3.8
SearchCreativeWork	41.6	42	1.8
SearchScreeningEvent	69.6	70	2.2
Average	62.7	62	2.6

Table 18: For each intent, the Success Rate of Back-Translation with the Small model, and Number of Generated Outputs, both averaged across the five random seeds. For reference, we also show the Average Number of Slots in the training data per intent.

L Filtering BT-5B outputs

The Back-Translated text with the 5B model is significantly cleaner than with the smaller model, so we do not apply any filtering on the output text itself. We do heuristically discard any outputs where we suspect the augmented utterance is missing a slot. Specifically, SimAlign in ArgMax mode only returns alignments across words that have mutual argmax between source and target. For any source word that is an entity tag (i.e., not "O"), if it is not aligned to an output word, then we consider the output invalid. For example, an input like `rate this book 5 out of 6` with a Back-Translated output `give this book a rating of 5` would typically have no output word aligned to the source word "6" (`best_rating` slot label), so the output would be discarded.

Success rate and number of outputs (after de-duplication) for BT-5B are reported in Table 19.

Modified Intent	SuccessRate	NumOutputs	AvgNumSlots
AddToPlaylist	66.2	411	2.7
BookRestaurant	82.8	423	3.2
GetWeather	72.0	311	2.3
PlayMusic	89.0	455	2.2
RateBook	79.2	478	3.8
SearchCreativeWork	85.5	451	1.8
SearchScreeningEvent	72.0	330	2.2
Average	78.1	408	2.6

Table 19: For each intent, the Success Rate of Back-Translation with the 5B model, and the number of outputs, both averaged across the five random seeds. For reference, we also show the Average Number of Slots in the training data per intent.

M Filtering LINGUIST Outputs

We apply heuristic filtering by discarding outputs which meet any of the following criteria: (1) copy one of the examples from the prompt verbatim; (2) fail to follow the prompt instructions, by not copying the

instructed slot value or by producing repeated, missing, extra, or malformed slot-tag numbers; (3) produce the literal wildcard instruction "*"; or (4) produce a content word containing a punctuation character in the set of {_<>[](){};}.[2]

In Table 20, we report the Success Rate as the portion of generated utterances which remain after filtering, and show the total number of generated utterances per intent. We observe a trend that success rate is generally lower when the prompt contains more slots, which is intuitive as the generation task is more challenging and has more chances to make a mistake. The success rates vary significantly by intent from 75.1 for BookRestaurant to 96.9 for GetWeather, with an average of 87.4 across the 7 intents.

Modified Intent	Success Rate	#Outputs	Average #Slots
AddToPlaylist	95.1	1230	2.7
BookRestaurant	75.1	2124	3.2
GetWeather	96.9	1197	2.3
PlayMusic	82.3	622	2.2
RateBook	78.0	1729	3.8
SearchCreativeWork	91.3	1154	1.8
SearchScreeningEvent	93.0	1370	2.2
Average	87.4	1346	2.6

Table 20: For each intent, the Success Rate of Generation, and Number of Generated Outputs, both averaged across the five random seeds. For reference, we also show the Average Number of Slots in the training data per intent.

N SNIPS Dataset Details

We retrieve the SNIPS dataset from https://github.com/sonos/nlu-benchmark/tree/master/2017-06-custom-intent-engines. For each intent, we use "full" training set file, e.g. `AddToPlaylist/train_AddToPlaylist_full.json` to split into Train and Development sets (described in Section 4.1.1). The validation data comes from the "validate" file for each intent, e.g. `AddToPlaylist/validate_AddToPlaylist.json`.

In `PlayMusic/train_PlayMusic_full.json`, as of the time of publishing, there is a bug in row 461 (0-based), where we replace `"Pop Punk Perfection <non_utf8_chars>"` with `"Pop Punk Perfection"` before processing the dataset.

We provide in Table 21 the row IDs (0-based) and md5sum of the training data subsets we use for the "s10" New-Intent Few-Shot (NIFS) setting, described in Section 4.2.1.

O SemER Metric

For the internal IC+ST benchmark (Sections 4.1.3, 4.5.2, and 5.3.2), we report on Semantic Error Rate (SemER) (Su et al., 2018) which jointly evaluates Intent Classification and Slot Filling. SemER is defined as follows: comparing a reference of tokens and their accompanying labels, count each of of these operations: (1) Correct slots, where the slot name and slot value is correctly identified, (2) Deletion errors, where the slot name is present in the reference but not in the hypothesis, (3) Insertion errors, where extraneous slot names are included in the hypothesis, (4) Substitution errors, where slot names from the hypothesis are included but with an incorrect slot value. Intent classification errors are substitution errors. Then, apply Equation 1 to compute the SemER.

$$\text{SemER} = \frac{\#\text{ Del} + \#\text{ Ins} + \#\text{ Sub}}{\#\text{ Cor} + \#\text{ Del} + \#\text{ Sub}} \quad (1)$$

[2]These characters do not appear in the text of any of the original training data, so are considered to be generation mistakes.

Seed	Intent	row IDs										md5sum
0	AddToPlaylist	81	271	314	495	561	636	856	1285	1615	1702	ade55e42e481f83c6617298d300758d8
	BookRestaurant	122	438	574	739	950	1252	1401	1420	1578	1728	60f6c7e4af848f3c7cfaedb02bb2f058
	GetWeather	163	348	454	529	870	932	966	1286	1368	1766	f37c6a4040e861d517e046719eb8da10
	PlayMusic	348	454	529	808	827	870	966	1286	1368	1766	a9899270bcfc3de3985d9b7149176916
	RateBook	125	129	181	690	739	1100	1243	1250	1600	1658	bcfa3063511c156738669da43f657284
	SearchCreativeWork	75	76	126	256	272	412	611	712	1216	1301	c8db82b42da972e2f3fbba4fb343ca97
	SearchScreeningEvent	76	117	236	261	411	785	919	1523	1856	1866	a32972f750a077709d22bd062b3a10f7
1	AddToPlaylist	15	26	80	91	459	637	723	735	844	1306	8463a2c95d6104cea85e096b1c9abb0f
	BookRestaurant	172	246	829	999	1061	1100	1203	1602	1717	1901	50cf89c558c1fd4387861968fbb8ea72
	GetWeather	155	466	857	957	1514	1673	1687	1748	1810	1930	4252f8c48f17062a003e73438254bdf8
	PlayMusic	155	466	857	957	1514	1683	1687	1748	1810	1930	94aa305f40be0d971b5fbf019fed0525
	RateBook	210	349	506	527	596	745	1174	1241	1295	1426	0b87d112f3073a31f6bb6b406e0d4f0c
	SearchCreativeWork	209	348	506	528	600	750	1175	1241	1299	1434	329937eef631008e1029f53ee7368513
	SearchScreeningEvent	210	522	660	862	880	951	983	1314	1766	1925	cdb4eb65ce6742a3106222d1e2b04add
2	AddToPlaylist	177	244	912	1044	1047	1218	1306	1374	1423	1541	f718676eafd6e2285ab11b2e35359aee
	BookRestaurant	252	469	555	849	969	1053	1113	1324	1570	1800	269ad74c558d8be458d1ee9dff612e21
	GetWeather	16	40	345	384	611	694	735	1071	1128	1587	8a4d66e79fc26c900c20baf99c7f429d
	PlayMusic	16	40	345	611	735	1071	1094	1128	1587	1840	45b22040849978b9590509ff53553661
	RateBook	561	840	937	1156	1234	1246	1314	1383	1401	1719	65b6a5bbc8ca80f4da4418362c16d299
	SearchCreativeWork	263	402	515	598	819	877	1116	1296	1657	1791	8d4f37ebd8f93e23a48f4ae60191303c
	SearchScreeningEvent	562	844	941	1160	1241	1253	1389	1393	1457	1635	4d4193a6a724b07a090bb483db8baded
3	AddToPlaylist	381	491	885	909	1341	1451	1459	1580	1778	1873	160fadf3bf9b76ce0af603992ae3a025
	BookRestaurant	42	245	570	702	1059	1227	1283	1330	1465	1887	319bd8b3fad9666ae1ebc4a3fa4bb9de
	GetWeather	90	127	502	522	759	910	957	1013	1242	1337	2188dd29d307e6200201eb0936b16a88
	PlayMusic	127	502	522	620	759	910	957	1093	1242	1840	3b1bb6d63a58ebe40b697aa1a7d9f1d7
	RateBook	42	70	372	447	768	1180	1594	1705	1838	1932	76fe7286f19552b820f0b9e4061bfe80
	SearchCreativeWork	42	70	372	447	772	1182	1207	1600	1711	1766	76de043a76b2c96d7e782ca890fb24d7
	SearchScreeningEvent	42	70	179	274	454	889	957	1058	1061	1256	6651011e32bbadaf55e169251d1f3a31
4	AddToPlaylist	26	58	276	328	403	574	834	1069	1644	1891	a2ce834b5ca753d67f88ff42530568f0
	BookRestaurant	228	270	519	946	1361	1482	1508	1832	1927	1936	d3d8c652313485e537b56e3279dddf80
	GetWeather	11	213	371	442	948	1040	1140	1280	1659	1835	b002ada961f3af1e95f34c46e2c3d047
	PlayMusic	11	213	277	371	442	948	1140	1280	1691	1835	81ed81653e9b7f8dbc0f05f38a97ecbb
	RateBook	58	128	208	815	876	891	941	1772	1784	1879	bce0c7d2d5b70a26378022ce3dba98fd
	SearchCreativeWork	58	127	207	817	894	943	1640	1699	1788	1878	54caeb85f764adac5856dd006d819418
	SearchScreeningEvent	58	128	596	894	952	962	1140	1365	1693	1928	767a9b5e0c83b4d043c74d57dcbba12e

Table 21: The Row IDs (0-based) used for the "s10" splits of the SNIPS dataset.

Adaptive Natural Language Generation for Task-oriented Dialogue via Reinforcement Learning

Atsumoto Ohashi **Ryuichiro Higashinaka**
Graduate School of Informatics, Nagoya University
ohashi.atsumoto.c0@s.mail.nagoya-u.ac.jp
higashinaka@i.nagoya-u.ac.jp

Abstract

When a natural language generation (NLG) component is implemented in a real-world task-oriented dialogue system, it is necessary to generate not only natural utterances as learned on training data but also utterances adapted to the dialogue environment (e.g., noise from environmental sounds) and the user (e.g., users with low levels of understanding ability). Inspired by recent advances in reinforcement learning (RL) for language generation tasks, we propose ANTOR, a method for **A**daptive **N**atural language generation for **T**ask-**O**riented dialogue via **R**einforcement learning. In ANTOR, a natural language understanding (NLU) module, which corresponds to the user's understanding of system utterances, is incorporated into the objective function of RL. If the NLG's intentions are correctly conveyed to the NLU, which understands a system's utterances, the NLG is given a positive reward. We conducted experiments on the MultiWOZ dataset, and we confirmed that ANTOR could generate adaptive utterances against speech recognition errors and the different vocabulary levels of users.

1 Introduction

In task-oriented dialogue systems, the role of the natural language generation (NLG) component is to convert a system's intentions, called dialogue acts (DAs), into natural language utterances and to convey DAs accurately to users (McTear, 2002; Gao et al., 2019). In recent years, data-driven language generation methods (Wen et al., 2015; Peng et al., 2020) using neural networks have been introduced to NLG for task-oriented dialogue systems, enabling natural utterance generation.

When such NLG is implemented in a realistic environment, however, it is essential to generate not only natural utterances as learned on training data but also utterances adapted to the dialogue environment and the user. For example, when interacting in a noisy environment, such as in a place with loud background noise or through a telephone, the system needs to use sentences and vocabulary that are less likely to be misrecognized. In addition, if the user is a child or a second language learner, it is necessary to generate utterances in plain terms that the user can easily understand. Therefore, it is essential for the NLG module to adaptively generate utterances for the dialogue environment and the user in real-world situations. However, it is challenging to implement optimal NLG using only supervised learning because it is not practical to create training data for every environment or user. Recently, for many generative tasks, such as machine translation, summary generation, and dialogue generation in open domains, many methods using reinforcement learning (RL) have been proposed. In these studies, non-differentiable objective functions, such as generated text quality and subjective user preferences, are used to optimize the language generation model and achieve high performance.

With this background in mind, this study proposes a method for **A**daptive **N**atural language generation for **T**ask-**O**riented dialogue via **R**einforcement learning (ANTOR)[1], which adapts to the dialogue environment and the user. In our method, a reward function using a natural language understanding (NLU) model is set up, and a pre-trained NLG model is fine-tuned by using RL. That is, the NLG generates a system utterance for a given DA, and the NLU provides a positive reward if it can successfully recognize the original DA from the utterance. Experiments using the MultiWOZ dataset (Budzianowski et al., 2018) are conducted with multiple environments and users simulating real-world conditions, such as speech recognition errors and the different vocabulary levels of users.

[1] In this paper, ANTOR refers to both the method of fine-tuning NLG and the fine-tuned NLG model. Our code and data are publicly available at https://github.com/nu-dialogue/antor

Proceedings of the 29th International Conference on Computational Linguistics, pages 242–252
October 12–17, 2022.

Our contribution is threefold:

- We propose ANTOR, a method for fine-tuning NLG for task-oriented dialogue via reinforcement learning. We conducted experiments using MultiWOZ to confirm that ANTOR can generate adaptive utterances for multiple NLUs with different model architectures.
- Experiments were conducted in a noisy environment where speech recognition errors caused by background noise were simulated. The results show that ANTOR could generate utterances with words less likely to cause speech recognition errors.
- Experiments were conducted using NLUs that simulated users with low vocabulary levels. The results confirmed that ANTOR was able to generate utterances using vocabulary appropriate for each vocabulary level.

2 Related Work

2.1 Natural Language Generation for Task-oriented Dialogue

Conventional NLG for task-oriented dialogues had used template-based and rule-based methods (Walker et al., 2002; Stent et al., 2004). There, templates and rules had to be carefully designed manually by experts in each domain. Later, a data-driven method using machine learning was proposed (Oh and Rudnicky, 2002; Angeli et al., 2010; Mairesse and Young, 2014). Kondadadi et al. (2013) proposed a method for statistically generating utterances using k-means clustering and support vector machines. Recently, many generation models based on end-to-end learning have been proposed by using deep learning (Wen et al., 2016; Tran and Nguyen, 2017; Su et al., 2018). Wen et al. (2015) proposed SC-LSTM, which controls utterance generation by using DA feature vectors and reading gates. SC-GPT (Peng et al., 2020) is a state-of-the-art model for MultiWOZ that achieves high performance by fine-tuning the language model GPT-2 (Radford et al., 2019) on a large number of task-oriented dialog datasets.

Some end-to-end models (Budzianowski et al., 2018; Chen et al., 2019) generate system utterances directly from a dialogue history instead of using NLG, which is known as a word-level policy. In particular, Zhao et al. (2019) and Mehri et al. (2019) optimize the word-level policy by RL to improve task completion. Although these methods use RL to generate system utterances, they do not deal with the NLG module itself and ways to make it adaptive to environments and users.

2.2 Adaptive Natural Language Generation for Task-oriented Dialogue

Methods have been proposed for generating utterances adapted to the user. Walker et al. (2004) used quantitative user modeling for multimodal dialogue to achieve speech production that takes user preferences into account. Janarthanam and Lemon (2010) used RL to create utterances that suit the user's domain knowledge. Dušek and Jurčíček (2016) proposed an NLG that can generate utterances exhibiting entrainment. Furthermore, Mairesse and Walker (2010) proposed PERSONAGE, an NLG that can generate utterances expressing Big Five personality traits. Our study differs from the above studies in that we optimize an existing NLG for the specific objective function of accurately conveying DAs for specific environments and users.

2.3 Natural Language Generation with Reinforcement Learning

In recent years, many methods have been proposed that use RL for language generation tasks (Luketina et al., 2019). There are machine translation methods (Wu et al., 2016; Bahdanau et al., 2016) using BLEU as the reward function, summary generation methods (Ranzato et al., 2015; Dong et al., 2018) using ROUGE, and story generation (Tambwekar et al., 2019). In addition, human feedback rather than automatic evaluation metrics is also used in many methods including machine translation (Kreutzer et al., 2018), summary generation (Ziegler et al., 2019; Stiennon et al., 2020), and open-domain dialogue (Hancock et al., 2019; Jaques et al., 2019). Our study examines the applicability of these recent advances to NLG in task-oriented dialogues.

3 Method

3.1 Task Overview

Figure 1 shows the overall task performed by NLG in this study. First, NLG takes a reference DA, representing system intentions, converts it into natural language, and outputs a system utterance. The user's NLU then predicts the system's DA (predicted DA) from the system utterance. The goal of ANTOR is to generate utterances such that the pre-

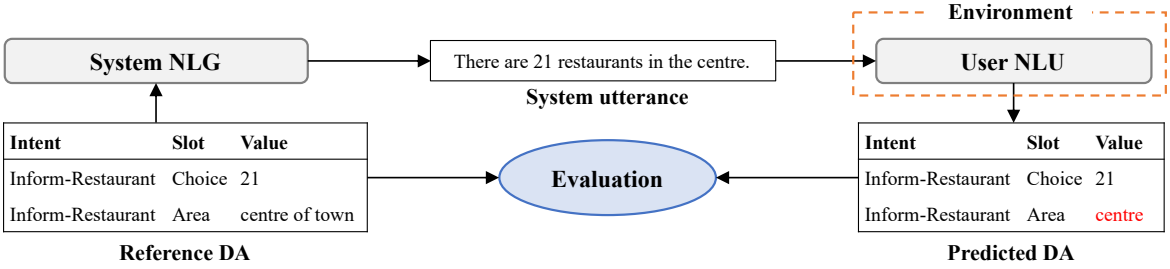

Figure 1: Overview of the task to be performed by NLG. NLG generates system utterance corresponding to reference DA. NLG is evaluated using reference DA and predicted DA, which NLU estimates from system utterance. The ability of User NLU to understand can vary and so can the environment.

dicted DA estimated by the NLU becomes equivalent to the reference DA. Note that this study uses automatic evaluation by comparing the reference DA and predicted DA; more down-to-earth evaluations using user subjective evaluations or user models are left for future work. In the following, the main concepts of the task, namely, DA, system NLG, user NLU, and evaluation, are described.

Dialogue Act The DA is a semantic representation of a system utterance. The reference DA A contains one or more triples consisting of intent I, slot s, and value v:

$$A = \{(I_1, s_1, v_1), ..., (I_{|A|}, s_{|A|}, v_{|A|})\}$$

I represents a system's intention in a domain. For example, in the restaurant domain, there are intentions such as "inform" and "request" (e.g., "Restaurant-Inform," "Restaurant-Request"). s and v indicate the category (e.g., "Choice" and "Area") and specific information belonging to s, respectively. The first line of the reference DA in Figure 1 indicates the semantics that there are 21 possible restaurants.

System NLG NLG generates a system utterance U on the basis of a given A. In this study, we assume a generative model with neural networks. Using the chain rule, a joint probability over $[A; U] = (x_1, ..., x_N)$ is modeled by a neural network ρ_θ with parameters θ:

$$\rho_\theta([A; U]) = \prod_{n=1}^{N} \rho_\theta(x_n | x_{<n}) \quad (1)$$

where N is the length of $[A; U]$. θ is trained by maximizing the log-likelihood (MLE) over dataset $D = \{[A_1; U_1], ..., [A_{|D|}; U_{|D|}]\}$:

$$\mathcal{L}(D) = \sum_{t=1}^{|D|} \sum_{n=1}^{N_t} \log \rho_\theta(x_n^t | x_{<n}^t) \quad (2)$$

where N_t is the length of $[A_t; U_t]$.

User NLU NLU predicts DA A' from U output by NLG. The structure of A' is the same as that of A. In this study, we assume a classification-based prediction model for intent detection and slot tagging. Intent detection performs multi-label classification of an utterance, and slot tagging categorizes each token in an utterance as to which slot it belongs. The training data D for NLG is the same as that for training NLU.

Evaluation The goal of NLG is to generate U such that $A = A'$. Therefore, the rate of concordance between A and A' is used to evaluate the NLG. Specifically, we use the F1 score calculated from true positive triples $A^{TP} = A \cap A'$, false negative triples $A^{FN} = A \cap \overline{A'}$, and false positive triples $A^{FP} = \overline{A} \cap A'$. In addition, following (Wang et al., 2020), Accuracy $= \frac{|A^{TP}|}{|A^{TP}|+|A^{FP}|+|A^{FN}|}$ is also used.

3.2 Fine-tuning via Reinforcement Learning

ANTOR is optimized by fine-tuning NLG pre-trained by MLE in Eq. (2) via RL. We use proximal policy optimization (PPO) (Schulman et al., 2017) for the RL algorithm. We initialize policy π_ϕ by using ρ_θ and add a randomly initialized linear layer that outputs a scalar value for a value network. Parameters ϕ are updated on the basis of the clipped surrogate objective $\mathcal{L}^{CLIP}(\phi)$. We incorporate an understanding of NLU into the reward for ANTOR. When computing the reward r, each utterance U generated by π_ϕ from $A \sim D$ is evaluated using A' predicted by NLU from U as follows:

$$r(A, A') = \mathrm{F1}(A, A') \frac{1}{|A^{TP}|} \sum_{(I,s,v) \in A^{TP}} \mathrm{idf}_D(I, s) \quad (3)$$

where $\mathrm{idf}_D(I, s)$ is the IDF value of (I, s) computed over all intent-slot pairs in D. This weighting

of F1 scores compensates for DAs that frequently occur in D (e.g., greetings) and DAs that occur infrequently.

Following Ziegler et al. (2019), to prevent π_ϕ from moving too far from ρ_θ, a penalty by Kullback–Leibler (KL) divergence is added to $r(A, A')$ as the final reward R:

$$R(A, A', U) = r(A, A') - \beta \log \frac{\pi_\phi(U|A)}{\rho_\theta(U|A)} \quad (4)$$

where β is the coefficient for the penalty. Algorithm 1 summarizes the fine-tuning process of ANTOR.

Algorithm 1 ANTOR with PPO

Require: Dataset D; NLU; Policy ρ_θ pre-trained via MLE by Eq. (2)

Initialize policy $\pi_{\phi_{old}} = \rho_\theta$
Randomly initialize value network in $\pi_{\phi_{old}}$
for i = 1, 2, ..., max iteration **do**
  for j = 1, 2, ..., batch size **do**
    Sample a reference DA A from D
    Sample an utterance U from A by $\pi_{\phi_{old}}$
    Get a predicted DA A' from U by NLU
    Compute reward $R(A, A', U)$ by Eq. (4)
    Compute advantage estimates
  end for
  Optimize $\mathcal{L}^{CLIP}(\phi)$, with pre-determined number of epochs and minibatch size
  $\phi_{old} \leftarrow \phi$
end for

4 Environment

We aim to confirm the feasibility of NLG that can robustly respond to the dialogue environment and the user, which does not exist in typical NLG training data. Therefore, we simulate two conditions in the following subsections. Note that D in this section is the same as the data for training NLG in Section 3.1; $D = \{[U_1; A_1], ..., [U_{|D|}; A_{|D|}]\}$.

4.1 Speech Recognition Error

When a dialogue system interacts with a user via voice, it is assumed that background noise makes it difficult for system utterances to be accurately conveyed to the user. Automatic speech recognition (ASR) error simulation is often used to construct a noisy channel between a user and system (Schatzmann et al., 2007; Fazel-Zarandi et al., 2019; Wang et al., 2020). Therefore, we apply perturbations that take the background noise into account to an utterance from NLG by using ASR error simulation. The noisy utterance is then used as input to NLU. Word substitution with a confusion matrix is used in the simulation (Figure 2). The TTS-ASR pipeline (Park et al., 2019) is used to construct the confusion matrix with the following procedure:

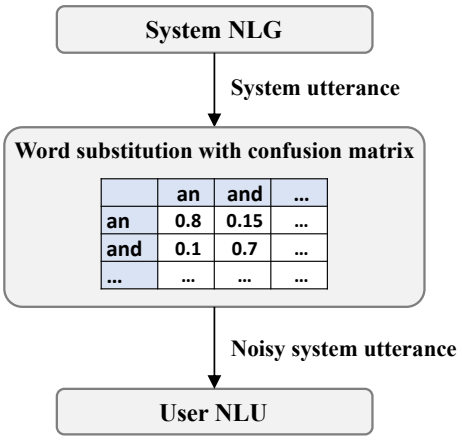

Figure 2: ASR error simulation to add noise to system utterance

1. Convert each U over D to audio data U^{Audio} using a text-to-speech (TTS) system.

2. Create $U^{NoisyAudio}$ by adding background noise to each U^{Audio}.

3. Recover each $U^{NoisyAudio}$ into text U^{Noisy} using an ASR system. This creates $D^{Noisy} = \{U_1^{Noisy}, ..., U_{|D|}^{Noisy}\}$.

4. Align words in U and U^{Noisy} with the Levenshtein distance and calculate the frequency of each word substituted and deleted from U to U^{Noisy}, resulting in an N-dimensional confusion matrix $M \in N^2$. N is the size of vocabulary $V = \{v_1, ..., v_N\}$ appearing in D or D^{Noisy}. Note that a special token denoting deletion is included in V.

Here, $M(i, j)$ indicates how often a word v_i is substituted into v_j. When simulating ASR errors, each word w in a system utterance is replaced by a word v_j according to the following probability:

$$p_w(v_j) = \begin{cases} \frac{M(i,j)}{\sum_{n=1}^{N} M(i,n)} & \text{if } \exists v_i \in V : w = v_i, \\ 0 & \text{otherwise} \end{cases}$$

4.2 Different Vocabulary Levels

In a real environment in which a dialogue system interacts, the users may not have a sufficient vocabulary, such as when they are children or second language learners. Therefore, NLG should use vocabulary and sentences appropriate to the user's vocabulary level. We can simulate the user's vocabulary level by adjusting the training data D for NLU as follows:

1. Prepare a word list $L = \{v_1, ..., v_{|L|}\}$ of the desired vocabulary level.
2. For each $[A; U] \in D$, if the lemma of a non-stop word[2] in U is not in L, the $[A; U]$ is excluded from D.

Using the adjusted training data, it is expected that the NLU can understand only the words in L.

5 Experiments

We wanted to confirm that ANTOR is capable of generating utterances adapted to the dialogue environment and the user. To verify the effectiveness of ANTOR, we conducted experiments using simulations.

5.1 Dataset

We used the MultiWOZ dataset (Budzianowski et al., 2018), which is a task-oriented dialogue dataset between a clerk and a tourist at a tourist information center. The dataset contains 10,438 dialogues in seven domains. We used only system utterances annotated with the clerk's DAs. We used a total of 56,750 utterances and DA pairs in the training data of MultiWOZ to train NLG and NLU. In addition, we also used the utterances to construct the confusion matrix used in the ASR error simulation.

5.2 Training Setup

ANTOR The 117M parameter version of the GPT-2 language model (Radford et al., 2019) was used as a base model. DAs were input to the model as a sequence of intent, slot, and value triples connected by the symbols "+" and "*"; if there were multiple triples, they were connected by commas ",". In addition, to control generation, special tokens "[ACT]" and "[RSP]" were added at the beginning of the DAs and the system utterance sequences, respectively, in order to indicate the start of each sequence. The following is an example input to the model:

```
[ACT] Inform-Restaurant + Choice
* 21, Inform-Restaurant + Area *
centre of town [RSP] There are
21 restaurants in the centre of
town.
```

In MLE, GPT-2 was trained on MultiWOZ for five epochs with a batch size of 8, following Peng et al. (2020). We used the Adam optimizer (Kingma and Ba, 2014) with a learning rate of 5e-5, and the learning rate decreased linearly with the number of steps.

For RL, 60 iterations were trained with a batch size of 1,024 (i.e., 1,024 utterances), and each batch was trained in 4 epochs with a minibatch size of 1. The coefficient β of the penalty for the KL divergence was set to 0.1. We use generalized advantage estimation (Schulman et al., 2015) (GAE) with a γ of 1.0 and λ of 0.95. The Adam optimizer was used with a learning rate of 5e-6, and the learning rate decreased linearly with the number of steps.

For fair evaluation, we trained ANTOR with five different random seeds. 7,372 pairs of DAs and system utterances from MultiWOZ test data were used for testing. The average of the five trials was used as the final score. A greedy search was used for utterance generation in a test.

User NLU As NLUs in our experiments, following the work of Liu et al. (2021), who evaluated NLUs with task-oriented dialogues, we used two models, MILU (Hakkani-Tür et al., 2016) and BERT (Devlin et al., 2019). Each model was trained by using pairs of DAs and system utterances from MultiWOZ training data. The learning rates were 1e-3 for MILU and 1e-4 for BERT as in (Liu et al., 2021).

5.3 Baselines

To evaluate the performance of ANTOR, we used three comparison models.

SC-LSTM (Wen et al., 2015) An LSTM-based method for controlling utterance generation with feature vectors related to DAs. We used a model pre-trained with MultiWOZ, which is available from ConvLab-2 (Zhu et al., 2020), a platform for task-oriented dialogue systems.

SC-GPT (Peng et al., 2020) A GPT-2 based model that has been trained on a large number

[2]We used the Python library Spacy for word tokenization, stop word determination, and word lemmatization.

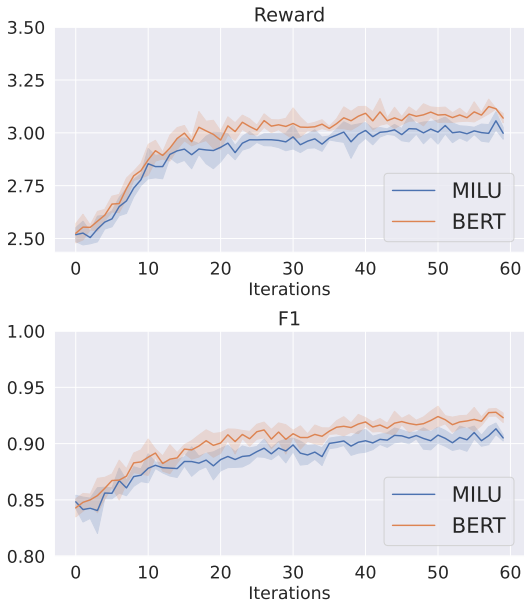

Figure 3: Increase in reward and F1 when ANTOR was trained in a clean environment using MILU and BERT, respectively.

of task-oriented dialogue datasets and further fine-tuned on MultiWOZ. In fine-tuning, training was done for five epochs with a batch size of 8, as reported in the official repository[3].

GPT-2 (Radford et al., 2019) A GPT-2 model fine-tuned on MultiWOZ using only MLE. The hyperparameters and input format were the same as those of ANTOR.

5.4 Experimental Procedure

The experiment was conducted in three stages using both MILU and BERT. First, we checked the effectiveness of ANTOR in clean environments with basic task-oriented dialogue. Next, we conducted two experiments: (1) in an ASR error simulation environment and (2) using NLUs trained only with low vocabulary levels.

5.5 Results in Clean Environment

Figure 3 shows the reward and F1 transition of ANTOR, indicating that the scores increased steadily. Table 1 shows the test scores for each model, indicating that ANTOR's accuracy and F1 outperformed the other models. These results show that ANTOR can learn utterance generation that fits both models of MILU and BERT. Note that the BLEU score of ANTOR was lower than those of SC-GPT and GPT-2. This BLEU score was calculated by comparing the utterances in MultiWOZ as references and the utterances generated by NLG as hypotheses. This means that ANTOR no longer generated utterances that appear in MultiWOZ in order to generate utterances tailored to NLU.

Model	MILU		BERT		
	Acc.	F1	Acc	F1	BLEU
SC-LSTM	74.0	78.6	73.6	77.7	25.3
SC-GPT	77.3	81.1	78.3	82.2	**29.9**
GPT-2	79.5	84.0	79.7	83.6	**29.9**
ANTOR (ours)	**86.7**	**89.8**	**87.8**	**90.7**	27.5

Table 1: Scores for each NLG model evaluated using MILU and BERT, respectively.

SNR	WER	Sub.	Ins.	Del.
0	30.4%	15.4%	8.5%	6.6%
5	23.9%	12.6%	9.1%	2.2%
10	21.5%	11.0%	9.2%	1.3%
20	19.9%	9.8%	9.1%	1.0%

Table 2: WER and percentage of error types for each SNR using TTS-ASR pipeline.

5.6 Conditions for Speech Recognition Error

We trained and evaluated ANTOR in an environment with ASR simulation. Google Cloud Text-to-Speech[4] and Speech-to-Text[5] were used for the TTS and ASR in the construction of the confusion matrix (see Section 4.1). The ESC-50 dataset (Piczak, 2015) was used as the background sound source, and it contains a total of 2,000 different sounds in five categories (e.g., natural soundscapes and urban noises). Randomly selected background noise was assigned to each utterance with a signal-to-noise ratio (SNR) of 0, 5, 10, and 20 dB. The range was selected so that the word error rate (WER) between original and noisy utterance text would be evenly distributed. Table 2 shows the WER of all of the data generated at each SNR and the percentages of substitution (Sub.), insertion (Ins.), and deletion (Del.) errors.

Table 3 shows the evaluation results for each model. Overall, ANTOR showed a higher accuracy and F1 than all three comparison models. These indicate that ANTOR can preferentially generate

[3] https://github.com/pengbaolin/SC-GPT

[4] https://cloud.google.com/text-to-speech

[5] https://cloud.google.com/speech-to-text

Model	SNR											
	0			5			10			20		
	Acc.	F1	WER	Acc.	F1	WER	Acc.	F1	WER	Acc.	F1	WER
SC-LSTM	46.6	53.8	26.9	50.6	57.9	17.8	52.2	59.6	14.8	53.5	60.8	13.1
SC-GPT	47.9	55.4	27.2	52.1	59.9	18.0	54.3	61.9	14.9	55.1	62.8	13.3
GPT-2	48.2	56.0	28.2	52.8	60.6	19.2	54.9	62.7	16.0	56.0	63.7	14.3
ANTOR (ours)	**51.9**	**59.4**	**26.8**	**56.9**	**64.2**	**18.5**	**59.9**	**66.9**	**14.8**	**60.9**	**67.9**	**13.7**

(a) MILU

Model	SNR											
	0			5			10			20		
	Acc.	F1	WER	Acc.	F1	WER	Acc.	F1	WER	Acc.	F1	WER
SC-LSTM	46.6	53.2	27.1	50.3	57.4	18.0	52.4	59.5	14.9	53.4	60.4	13.0
SC-GPT	48.9	56.4	27.2	53.5	60.8	18.2	55.1	62.4	15.0	56.2	63.4	13.2
GPT-2	48.7	56.2	28.3	53.5	61.0	19.3	55.6	62.9	16.1	56.5	63.9	14.1
ANTOR (ours)	**54.2**	**61.5**	**28.0**	**58.8**	**66.0**	**18.7**	**60.8**	**67.9**	**15.9**	**61.9**	**69.0**	**13.7**

(b) BERT

Table 3: Scores for methods evaluated in ASR error simulation environment with background noise at each SNR, using MILU and BERT, respectively. WER indicates how much error was imposed on NLG's output utterances.

Model		CEFR-J level							
		≤A1		≤A2		≤B1		≤B2	
NLG	NLU	Acc.	F1	Acc.	F1	Acc.	F1	Acc.	F1
SC-LSTM	MILU	47.4	53.9	55.7	62.5	62.4	68.8	63.6	69.8
SC-GPT	MILU	47.2	53.9	56.4	63.5	63.4	70.1	66.1	72.5
GPT-2	MILU	48.4	55.0	57.7	64.7	66.0	72.6	68.6	74.8
ANTOR (ours)	MILU	**54.7**	**61.1**	**63.5**	**69.8**	**72.5**	**78.0**	**75.0**	**80.1**
SC-LSTM	BERT	68.6	73.6	72.0	76.3	72.8	76.9	73.1	77.2
SC-GPT	BERT	70.3	75.9	73.3	77.9	77.3	81.4	77.6	81.9
GPT-2	BERT	65.3	70.8	74.9	79.4	79.1	83.5	78.2	82.5
ANTOR (ours)	BERT	**83.0**	**87.0**	**85.7**	**89.2**	**87.8**	**90.6**	**87.7**	**90.6**

Table 4: Scores for each NLG model when evaluated using MILU and BERT. Both MILU and BERT were trained using only vocabulary defined at each CEFR-J level.

Model	% vocab. level in generation			
	≤A1	≤A2	≤B1	≤B2
GPT-2	64.8	77.2	87.2	90.3
ANTOR w/ MILU	**68.1**	**79.8**	**88.0**	**91.0**
ANTOR w/ BERT	67.0	79.6	**88.0**	**91.0**

Table 5: Percentage of vocabulary levels to which words generated by GPT-2 and ANTOR belong. "w/ MILU" and "w/ BERT" are ANTOR models trained with MILU and BERT, respectively. Note that ANTOR is fine-tuned using NLU trained on data at the CEFR-J level indicated by each column.

words that are less likely to be confused. The above result shows that fine-tuning via RL enabled NLG to generate utterances adapted to the noisy environment, regardless of the noise intensity.

5.7 Conditions for Different Vocabulary Levels

We experimented with multiple NLUs trained on data that were gradually filtered along vocabulary levels. For word lists organized by vocabulary level, we used the Common European Framework of Reference (CEFR)'s English Vocabulary Profile[6]. The CEFR defines six levels of language acquisition, from A1 (beginner) to C2 (proficient, comparable to native speakers), with a word list for each level. In our experiment, we used the CEFR-J (Tono and Negishi, 2012) word list for Japanese-English learners[7]. We created four types of training data for NLU by filtering MultiWOZ data with a

[6] https://www.englishprofile.org/

[7] http://www.cefr-j.org/download_eng.html

Intent-slot pair	Num.	% TP change
(Request-Taxi, Depart)	142	14.5 → 90.4
(OfferBook-Train, People)	6	0.0 → 66.7
(Recommend-Hotel, Postcode)	12	14.3 → 80.0
(Recommend-Restaurant, Price)	49	20.8 → 84.0
(NoOffer-Hotel, none)	32	23.5 → 86.7

Table 6: Top five intent-slot pairs that were correctly recognized by BERT at a higher percentage (% TP) by ANTOR compared with GPT-2. "Num." indicates the number of times each intent-slot pair appeared during test.

focus on the four levels ≤A1, ≤A2, ≤B1, and ≤B2 (see Section 4.2). Note that C1 and C2 were not available in CEFR-J and were not used in our experiment. As a result of the filtering, the number of utterances in the datasets at the ≤A1, ≤A2, ≤B1, and ≤B2 levels was 11,190, 15,538, 24,311, and 28,999 utterances, respectively.

ANTOR was trained and evaluated using MILU and BERT trained on each of the four vocabulary levels. Table 4 shows the results. ANTOR outperformed all three comparison models at all vocabulary levels, both with MILU and with BERT. In particular, when using BERT at level ≤A1, the original GPT-2's accuracy and F1 were lower than SC-LSTM and SC-GPT. However, ANTOR had an accuracy and F1 that were significantly improved over the GPT-2 scores by 17.7% and 16.2%, respectively, and it outperformed the scores of SC-LSTM and SC-GPT.

We checked whether the vocabulary in the utterances generated by ANTOR actually changed due to RL. Table 5 shows the percentage of the vocabulary in the utterances that ANTOR had generated during the evaluation when trained with each NLU of each level. Note that stop words and proper nouns were excluded from the calculation. We see that both ANTOR w/ MILU and w/ BERT generated words at each level with a higher frequency than the original GPT-2. From these results, it is considered that the NLG was able to learn utterance generation tailored to the NLU's ability to understand.

6 Case Study

To see how ANTOR improved the performance of NLU, we analyzed the behavior of ANTOR. In this analysis, we used BERT because we thought that a difference from GPT-2 could be clearly seen since the F1 was improved more by using BERT than MILU as in Table 1. The case studies here are done

Ref. DA	[Request-Taxi-Depart-?, Request-Taxi-Leave-?]
GPT-2	I can help you with that. What time would you like to leave and where are you departing from?
BERT	[Request-Train-Depart-?, Request-Train-Leave-?]
ANTOR	Where would you like to **be picked up** and at what time?
BERT	[Request-Taxi-Depart-?, Request-Taxi-Leave-?]

(a) Case 1

Ref. DA	[Recommend-Restaurant-Name-charlie chan, Recommend-Restaurant-Price-cheap, Recommend-Restaurant-Postcode-cb21db]
GPT-2	charlie chan is a cheap restaurant and the postcode is cb21db.
BERT	[Inform-Restaurant-Name-charlie chan, Inform-Restaurant-Price-cheap, Inform-Restaurant-Postcode-cb21db]
ANTOR	I **recommend** charlie chan, it is in the cheap price range and the postcode is cb21db.
BERT	[Recommend-Restaurant-Name-Charlie chan, Recommend-Restaurant-Price-cheap, Recommend-Restaurant-Postcode-cb21db]

(b) Case 2

Table 7: Examples of utterances generated by GPT2 and ANTOR from DAs and DAs predicted by BERT for each utterance. Letters in red indicate DAs misrecognized by BERT. Letters in blue indicate words that may have influenced BERT's prediction.

in clean environments, but similar behaviors were observed for noisy environments and different user vocabulary levels.

First, we listed the intents and slots in the DAs for which the NLU prediction accuracy was considerably improved by the utterances of ANTOR compared with those of GPT-2 (Table 6). Next, we examined the utterances that each model generated from the listed DAs. Table 7 shows examples of utterances generated for (Request-Taxi, Depart) and (Recommend-Restaurant, Price), which have a particularly high occurrence as in Table 6. In case 1, BERT misidentified the train domain instead of the taxi domain from the GPT-2 utterances. In contrast, ANTOR correctly conveyed the DAs by explicitly using the phrase "be picked up." In case 2, the intention of "inform" was conveyed by GPT-2 instead of "recommend." However, ANTOR explicitly used the word "recommend" to correctly convey the DA.

These results suggest that fine-tuning NLG using RL enables NLG to generate utterances adapted to the NLU.

Note that since humans will not have the problems that the NLU had here because humans have a better understanding, we expect ANTOR to adapt differently when interacting with humans.

7 Summary and Future Work

This paper investigated whether NLG can generate utterances adapted to the dialogue environment and the user via RL. We proposed a method, ANTOR, and conducted experiments using MultiWOZ to confirm that ANTOR can generate such utterances for multiple NLUs with different model architectures. In addition, we also consistently confirmed the effectiveness of ANTOR for noisy environments and a user's vocabulary levels.

For future work, we plan to evaluate whether ANTOR optimized for NLU is also effective for humans. We are also interested in extending our method for practical use (e.g., real-time adaptation to users in an online dialogue environment). Furthermore, we would like to utilize methods to optimize an entire system with RL, such as (Mehri et al., 2019) and (Ohashi and Higashinaka, 2022), so that all modules of a system can be adapted to users.

Acknowledgments

This work was supported by JSPS KAKENHI Grant Number 19H05692. We used the computational resources of the supercomputer "Flow" at the Information Technology Center, Nagoya University.

References

Gabor Angeli, Percy Liang, and Dan Klein. 2010. A simple domain-independent probabilistic approach to generation. In *Proceedings of the 2010 Conference on Empirical Methods in Natural Language Processing*, pages 502–512.

Dzmitry Bahdanau, Philemon Brakel, Kelvin Xu, Anirudh Goyal, Ryan Lowe, Joelle Pineau, Aaron Courville, and Yoshua Bengio. 2016. An actor-critic algorithm for sequence prediction. *arXiv preprint arXiv:1607.07086*.

Paweł Budzianowski, Tsung-Hsien Wen, Bo-Hsiang Tseng, Iñigo Casanueva, Stefan Ultes, Osman Ramadan, and Milica Gašić. 2018. MultiWOZ - A Large-Scale Multi-Domain Wizard-of-Oz Dataset for Task-Oriented Dialogue Modelling. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 5016–5026.

Wenhu Chen, Jianshu Chen, Pengda Qin, Xifeng Yan, and William Yang Wang. 2019. Semantically Conditioned Dialog Response Generation via Hierarchical Disentangled Self-Attention. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 3696–3709.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 4171–4186.

Yue Dong, Yikang Shen, Eric Crawford, Herke van Hoof, and Jackie Chi Kit Cheung. 2018. BanditSum: Extractive Summarization as a Contextual Bandit. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 3739–3748.

Ondřej Dušek and Filip Jurčíček. 2016. A Context-aware Natural Language Generator for Dialogue Systems. In *Proceedings of the 17th Annual Meeting of the Special Interest Group on Discourse and Dialogue*, pages 185–190.

Maryam Fazel-Zarandi, Longshaokan Wang, Aditya Tiwari, and Spyros Matsoukas. 2019. Investigation of error simulation techniques for learning dialog policies for conversational error recovery. *arXiv preprint arXiv:1911.03378*.

Jianfeng Gao, Michel Galley, Lihong Li, et al. 2019. Neural approaches to conversational ai. *Foundations and trends® in information retrieval*, pages 127–298.

Dilek Hakkani-Tür, Gokhan Tur, Asli Celikyilmaz, Yun-Nung Vivian Chen, Jianfeng Gao, Li Deng, and Ye-Yi Wang. 2016. Multi-Domain Joint Semantic Frame Parsing using Bi-directional RNN-LSTM. In *Proceedings of The 17th Annual Meeting of the International Speech Communication Association*.

Braden Hancock, Antoine Bordes, Pierre-Emmanuel Mazare, and Jason Weston. 2019. Learning from Dialogue after Deployment: Feed Yourself, Chatbot! In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 3667–3684.

Srinivasan Janarthanam and Oliver Lemon. 2010. Learning to Adapt to Unknown Users: Referring Expression Generation in Spoken Dialogue Systems. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics*, pages 69–78.

Natasha Jaques, Asma Ghandeharioun, Judy Hanwen Shen, Craig Ferguson, Agata Lapedriza, Noah Jones, Shixiang Gu, and Rosalind Picard. 2019. Way off-policy batch deep reinforcement learning of implicit human preferences in dialog. *arXiv preprint arXiv:1907.00456*.

Diederik P Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*.

Ravikumar Kondadadi, Blake Howald, and Frank Schilder. 2013. A statistical nlg framework for aggregated planning and realization. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics*, pages 1406–1415.

Julia Kreutzer, Shahram Khadivi, Evgeny Matusov, and Stefan Riezler. 2018. Can Neural Machine Translation be Improved with User Feedback? In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 92–105.

Jiexi Liu, Ryuichi Takanobu, Jiaxin Wen, Dazhen Wan, Hongguang Li, Weiran Nie, Cheng Li, Wei Peng, and Minlie Huang. 2021. Robustness Testing of Language Understanding in Task-Oriented Dialog. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing*, pages 2467–2480.

Jelena Luketina, Nantas Nardelli, Gregory Farquhar, Jakob Foerster, Jacob Andreas, Edward Grefenstette, Shimon Whiteson, and Tim Rocktäschel. 2019. A Survey of Reinforcement Learning Informed by Natural Language. In *Proceedings of the Twenty-Eighth International Joint Conference on Artificial Intelligence*, pages 6309–6317.

François Mairesse and Marilyn A Walker. 2010. Towards personality-based user adaptation: psychologically informed stylistic language generation. *User Modeling and User-Adapted Interaction*, pages 227–278.

François Mairesse and Steve Young. 2014. Stochastic Language Generation in Dialogue using Factored Language Models. *Computational Linguistics*, pages 763–799.

Michael F. McTear. 2002. Spoken Dialogue Technology: Enabling the Conversational User Interface. *ACM Computing Surveys*, page 90–169.

Shikib Mehri, Tejas Srinivasan, and Maxine Eskenazi. 2019. Structured Fusion Networks for Dialog. In *Proceedings of the 20th Annual SIGdial Meeting on Discourse and Dialogue*, pages 165–177.

Alice H Oh and Alexander I Rudnicky. 2002. Stochastic natural language generation for spoken dialog systems. *Computer Speech & Language*, pages 387–407.

Atsumoto Ohashi and Ryuichiro Higashinaka. 2022. Post-processing Networks: Method for Optimizing Pipeline Task-oriented Dialogue Systems using Reinforcement Learning. In *Proceedings of the 23rd Annual Meeting of the Special Interest Group on Discourse and Dialogue*, pages 1–13.

Daniel S. Park, William Chan, Yu Zhang, Chung-Cheng Chiu, Barret Zoph, Ekin D. Cubuk, and Quoc V. Le. 2019. SpecAugment: A Simple Data Augmentation Method for Automatic Speech Recognition. In *Proc. Interspeech 2019*, pages 2613–2617.

Baolin Peng, Chenguang Zhu, Chunyuan Li, Xiujun Li, Jinchao Li, Michael Zeng, and Jianfeng Gao. 2020. Few-shot natural language generation for task-oriented dialog. *arXiv preprint arXiv:2002.12328*.

Karol J. Piczak. 2015. ESC: Dataset for Environmental Sound Classification. In *Proceedings of the 23rd Annual ACM Conference on Multimedia*, pages 1015–1018.

Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, Ilya Sutskever, et al. 2019. Language models are unsupervised multitask learners. *OpenAI blog*, page 9.

Marc'Aurelio Ranzato, Sumit Chopra, Michael Auli, and Wojciech Zaremba. 2015. Sequence level training with recurrent neural networks. *arXiv preprint arXiv:1511.06732*.

Jost Schatzmann, Blaise Thomson, and Steve Young. 2007. Error simulation for training statistical dialogue systems. In *Proceedings of 2007 IEEE Workshop on Automatic Speech Recognition & Understanding*, pages 526–531.

John Schulman, Philipp Moritz, Sergey Levine, Michael Jordan, and Pieter Abbeel. 2015. High-dimensional continuous control using generalized advantage estimation. *arXiv preprint arXiv:1506.02438*.

John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. 2017. Proximal policy optimization algorithms. *arXiv preprint arXiv:1707.06347*.

Amanda Stent, Rashmi Prasad, and Marilyn Walker. 2004. Trainable Sentence Planning for Complex Information Presentations in Spoken Dialog Systems. In *Proceedings of the 42nd Annual Meeting of the Association for Computational Linguistics*, pages 79–86.

Nisan Stiennon, Long Ouyang, Jeffrey Wu, Daniel Ziegler, Ryan Lowe, Chelsea Voss, Alec Radford, Dario Amodei, and Paul F Christiano. 2020. Learning to summarize with human feedback. In *Advances in Neural Information Processing Systems*, pages 3008–3021.

Shang-Yu Su, Kai-Ling Lo, Yi-Ting Yeh, and Yun-Nung Chen. 2018. Natural Language Generation by Hierarchical Decoding with Linguistic Patterns. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 61–66.

Pradyumna Tambwekar, Murtaza Dhuliawala, Lara J. Martin, Animesh Mehta, Brent Harrison, and Mark O. Riedl. 2019. Controllable Neural Story Plot Generation via Reward Shaping. In *Proceedings of the Twenty-Eighth International Joint Conference on Artificial Intelligence*, pages 5982–5988.

Yukio Tono and Masashi Negishi. 2012. The CEFR-J: Adapting the CEFR for English language teaching in Japan. *Framework & Language Portfolio SIG Newsletter*, pages 5–12.

Van-Khanh Tran and Le-Minh Nguyen. 2017. Natural Language Generation for Spoken Dialogue System using RNN Encoder-Decoder Networks. In *Proceedings of the 21st Conference on Computational Natural Language Learning*, pages 442–451.

M.A. Walker, S.J. Whittaker, A. Stent, P. Maloor, J. Moore, M. Johnston, and G. Vasireddy. 2004. Generation and evaluation of user tailored responses in multimodal dialogue. *Cognitive Science*, pages 811–840.

Marilyn A. Walker, Owen C. Rambow, and Monica Rogati. 2002. Training a sentence planner for spoken dialogue using boosting. *Computer Speech & Language*, pages 409–433.

Longshaokan Wang, Maryam Fazel-Zarandi, Aditya Tiwari, Spyros Matsoukas, and Lazaros Polymenakos. 2020. Data Augmentation for Training Dialog Models Robust to Speech Recognition Errors. In *Proceedings of the 2nd Workshop on Natural Language Processing for Conversational AI*, pages 63–70.

Tsung-Hsien Wen, Milica Gašić, Nikola Mrkšić, Lina M. Rojas-Barahona, Pei-Hao Su, David Vandyke, and Steve Young. 2016. Multi-domain Neural Network Language Generation for Spoken Dialogue Systems. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 120–129.

Tsung-Hsien Wen, Milica Gašić, Nikola Mrkšić, Pei-Hao Su, David Vandyke, and Steve Young. 2015. Semantically Conditioned LSTM-based Natural Language Generation for Spoken Dialogue Systems. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 1711–1721.

Yonghui Wu, Mike Schuster, Zhifeng Chen, Quoc V Le, Mohammad Norouzi, Wolfgang Macherey, Maxim Krikun, Yuan Cao, Qin Gao, Klaus Macherey, et al. 2016. Google's neural machine translation system: Bridging the gap between human and machine translation. *arXiv preprint arXiv:1609.08144*.

Tiancheng Zhao, Kaige Xie, and Maxine Eskenazi. 2019. Rethinking Action Spaces for Reinforcement Learning in End-to-end Dialog Agents with Latent Variable Models. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 1208–1218.

Qi Zhu, Zheng Zhang, Yan Fang, Xiang Li, Ryuichi Takanobu, Jinchao Li, Baolin Peng, Jianfeng Gao, Xiaoyan Zhu, and Minlie Huang. 2020. ConvLab-2: An Open-Source Toolkit for Building, Evaluating, and Diagnosing Dialogue Systems. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics: System Demonstrations*, pages 142–149.

Daniel M Ziegler, Nisan Stiennon, Jeffrey Wu, Tom B Brown, Alec Radford, Dario Amodei, Paul Christiano, and Geoffrey Irving. 2019. Fine-tuning language models from human preferences. *arXiv preprint arXiv:1909.08593*.

TAKE: Topic-shift Aware Knowledge sElection for Dialogue Generation

Chenxu Yang[1,2], **Zheng Lin**[1,2]*, **Jiangnan Li**[1,2], **Fandong Meng**[3],
Weiping Wang[1], **Lanrui Wang**[1,2], **Jie Zhou**[3]
[1]Institute of Information Engineering, Chinese Academy of Sciences, Beijing, China
[2]School of Cyber Security, University of Chinese Academy of Sciences, Beijing, China
[3]Pattern Recognition Center, WeChat AI, Tencent Inc, China
{lijiangnan,linzheng,wangweiping}@iie.ac.cn
{yangchenxu22,wanglanrui22}@mails.ucas.ac.cn
{fandongmeng,withtomzhou}@tencent.com

Abstract

Knowledge-grounded dialogue generation consists of two subtasks: knowledge selection and response generation. The knowledge selector generally constructs a query based on the dialogue context and selects the most appropriate knowledge to help response generation. Recent work finds that realizing who (the user or the agent) holds the initiative and utilizing the role-initiative information to instruct the query construction can help select knowledge. It depends on whether the knowledge connection between two adjacent rounds is smooth to assign the role. However, whereby the user takes the initiative only when there is a strong semantic transition between two rounds, probably leading to initiative misjudgment. Therefore, it is necessary to seek a more sensitive reason beyond the initiative role to help refine the history information used to construct the query. To address the above problem, we propose a Topic-shift Aware Knowledge sElector(TAKE). Specifically, we first annotate the topic shift and topic inheritance labels in multi-round dialogues via distant supervision. Then, we alleviate the noise problem in pseudo labels through curriculum learning and knowledge distillation. Extensive experiments on WoW show that TAKE performs better than strong baselines.[1]

1 Introduction

Due to the dull response generation problem in the general open-domain dialogue generation technology, an increasing number of researchers focus on knowledge-grounded dialogue generation (KGDG) (Ghazvininejad et al., 2017; Li et al., 2019; Chen et al., 2020a; Zhan et al., 2021a). By connecting the external knowledge base with the generation model as supplement information, the generated response becomes more engaging and informative.

Topic: French Bulldog	
User	(1) I love my French bulldog!
Agent	(2) Aww, I bet your dog is so cute. The French Bulldog is a small breed and also known as the Frenchie. **<French Bulldog's nickname>**
User	(3) Yes, it is. What **color** is a bulldog usually?
Agent	(4) They are a cross between bulldogs and ratters. Most are white or black. **<French Bulldog's color>**
User	(5) I see, do you own any pets?
Agent	(6) I have a pet snake. His name is Slinky. **<snake>**
User	(7) Cool! Is it safe to have snake pets?
Agent	(8) Yes, if you have the right enclosure. I have one that likes to eat prey much larger than his head. I feed him rats. **<snake>**

Table 1: An example of topic shift in WoW dataset. The utterances (2) to (4) exhibit a sub-topic shift: from dog's nickname to dog's color; the utterances (4) to (6) exhibit an obvious topic shift: from dog to snake; the utterances (6) to (8) exhibit a topic inheritance.

Knowledge selection plays a vital role in KGDG task (Meng et al., 2021). Since one can choose any reasonable knowledge to carry on the conversation, one-to-many relations exist between dialogue context and knowledge (Kim et al., 2020). Thus, selecting the most appropriate knowledge in the vast knowledge pool becomes a significant challenge. And most of the existing methods pay close attention to the design of the knowledge selector.

Some methods attempt to improve the accuracy of knowledge selection by discovering more features in the dialogue context (Meng et al., 2020; Zheng et al., 2020) or introducing extra posterior knowledge (Lian et al., 2019; Kim et al., 2020; Chen et al., 2021). These methods often directly

* Zheng Lin is the corresponding author.
[1]The code is available at https://github.com/iie-ycx/COLING2022-TAKE.

Proceedings of the 29th International Conference on Computational Linguistics, pages 253–265
October 12–17, 2022.

take the whole dialogue context as the input of the knowledge selector, ignoring that different parts of the context play different roles. Considering this, Meng et al. (2021) propose to decouple the knowledge selector according to the different part of the input and only keep part of the history information by introducing mixed-initiative (user-initiative and agent-initiative) characteristics. In their method, the user takes the initiative when the knowledge connection between two adjacent rounds is unsmooth. Such a judgement only works when there is a strong semantic transition between two rounds. However, the conversational direction can be changed by the user when the user shifts the topic from one to another relevant one. For example, the utterance flow (2) → (3) → (4) in Table 1 shows that the topic shifts from *dog's nickname* to *dog's color* by the user, which means that the user dominates the dialogue direction. Nevertheless, the knowledge connection here is smooth because the knowledge is still relevant to dogs, leading to an agent-initiative judgement. The misjudgement will make the model choose the improper part of (agent-related) history information to select knowledge.

To address the above problem, we bring the topic into multi-turn knowledge selection. Through our observation, we find that the topic shift and topic inheritance affect knowledge selection deeply. For the topic shift, it is generally caused by the active user's frequent questioning and the model should select knowledge according to the current user utterance. For the topic inheritance, it is mainly caused by relatively passive users who agree with what the agent says, and the model needs to find some relative topics to continue the conversation according to previously selected knowledge. We obtain the topic shift label via distant supervision (Mintz et al., 2009), where we regard the retrieving entity as the topic word. Considering that there may exist noises in the pseudo labels, we further alleviate their negative effects through curriculum learning and knowledge distillation (see section 2.6 for details).

Our contributions in this paper are as follows:

- For the KGDG task, we find that the topic shift triggers knowledge alteration, and propose a Topic-shift Aware Knowledge sElector (TAKE) to better locate the relevant parts from the dialogue history at an opportune moment.
- To overcome the noisy label problem introduced by distant supervision, we optimize the topic-shift aware knowledge selector through curriculum learning and knowledge distillation, which can effectively alleviate the negative influence of pseudo topic labels.
- Experimental results on WoW dataset show that compared with strong baselines, TAKE not only selects knowledge more accurately especially on the `unseen` test set, but also generates more informative responses on both automatic and human evaluation metrics.

2 Approach

2.1 Task Formulation

Suppose we have a t-rounds conversation $C = \{(X_t, Y_t)\}, t = 1, 2, \ldots, |C|$, where X_t and Y_t are the utterances of the user and the agent at turn t respectively. In each turn, before the dialogue agent generates responses, the model is externally connected with a knowledge pool $\mathcal{K}_t = \{K_1^t, K_2^t, \ldots, K_D^t\}$ which contains D pieces of knowledge. Given the current user utterance X_t, the dialogue history $\{X_i, Y_i\}_{i=1}^{t-1}$, the previous golden knowledge $\{K_i'\}_{i=1}^{t-1}$ and the current knowledge pool $\mathcal{K}_t$, our goal is to select the most appropriate knowledge K_t' from the pool and make use of the selected knowledge K_t^s to generate the response $Y_t = (y_1^t, y_2^t, \ldots, y_{|Y_t|}^t)$.

2.2 Overview of TAKE

As shown in Figure 1, our model TAKE contains three components: Mixed Encoder, Topic-aware Knowledge Selector and Decoder. In the following subsections, we first introduce the three components in section 2.3, 2.4 and 2.5. Next, we present how to utilize curriculum learning and knowledge distillation to alleviate the problem of noisy pseudo labels obtained by distant supervision in section 2.6. Finally, we detail our training strategy and loss functions.

2.3 Mixed Encoder

We take BERT as the backbone of Encoder. At turn t, given the user utterance X_t and the knowledge pool $\mathcal{K}_t = \{K_1^t, K_2^t, \ldots, K_D^t\}$, we mix the user utterance and the knowledge sentences following (Zhao et al., 2020b). Specifically, we concatenate X_t with [CLS] [SEP] token in BERT and the candidate knowledge K_i^t to acquire M_i^t. In this way, we can better use the multi-layer bidirectional attention mechanism in BERT to allow the dialogue context interact sufficiently with knowledge candidates.

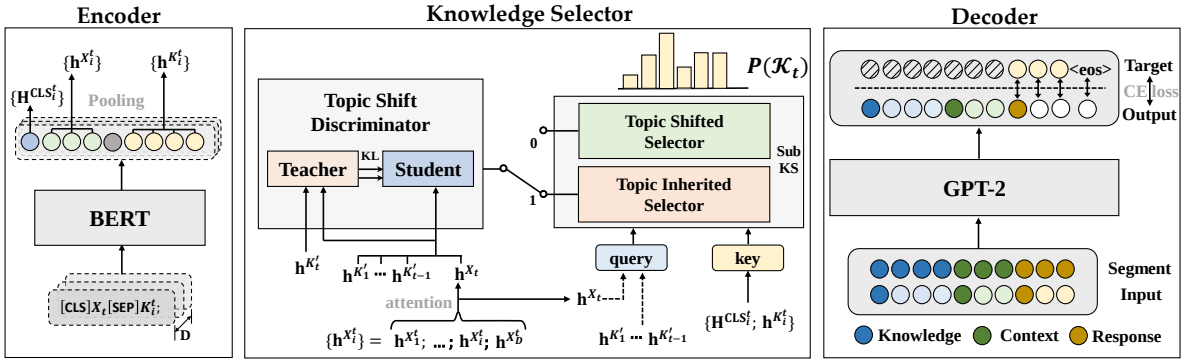

Figure 1: Architecture of the proposed model.

$$M_i^t = [\text{CLS}]X_t[\text{SEP}]K_i^t \tag{1}$$

We then take D concatenated context-knowledge pairs $M_t = \{M_1^t, M_2^t, \ldots, M_D^t\}$ as the input of the Encoder. Tokens in M_t will be encoded as word representations $[\mathbf{H}^{\text{CLS}_i^t}; \mathbf{H}^{X_i^t}; \mathbf{H}^{\text{SEP}_i^t}; \mathbf{H}^{K_i^t}]$. After that, we obtain sentence representations via mean pooling (Cer et al., 2018):

$$\begin{aligned} \mathbf{h}^{X_i^t} &= \text{pooling}(\mathbf{H}^{X_i^t}), \\ \mathbf{h}^{K_i^t} &= \text{pooling}(\mathbf{H}^{K_i^t}) \end{aligned} \tag{2}$$

In order to get a unified sentence representation of the user utterance, we integrate the D representations through additive attention mechanism and obtain $\mathbf{h}^{X_t}$:

$$\begin{aligned} s_i &= v_u^\top \tanh(W_u \mathbf{h}^{X_i^t}) \\ \alpha^t &= \text{SM}(\{s_i\}_{i=1}^D), \mathbf{h}^{X_t} = \sum_{i=1}^{D} \alpha_i^t \mathbf{h}^{X_i^t} \end{aligned} \tag{3}$$

where v_u and W_u are trainable weights, and SM means softmax function.

2.4 Topic-aware Knowledge Selector

The knowledge selector module we designed is composed of three parts: Topic Shift Discriminator, Topic-shifted Knowledge Selector and Topic-inherited Knowledge Selector. We name the latter two networks as Sub-KS for simplicity.

The topic shift discriminator can judge whether the topic shift or the topic inheritance will occur at the current turn and then choose one selector in Sub-KS. According to the previous analysis, in the case of topic shift, it is more likely that the user mentions a new topic. Hence the topic-shifted knowledge selector makes full use of the current user utterance to construct query vectors. Otherwise, the topic-shifted knowledge selector inherits a topic from the previous conversation and selects a knowledge under the topic.

Due to BERT's NSP pre-training scheme (Devlin et al., 2018), the [CLS] token is endowed with the ability to extract semantic information in sentences. We use a feedforward layer to extract the semantic associative information between the current user utterance and the knowledge candidates as:

$$\mathbf{u}_i^t = \text{Relu}(\text{FC}(\mathbf{H}^{\text{CLS}_i^t})) \tag{4}$$

where FC means fully connected layers.

Besides, we use the attention mechanism on knowledge candidates similar to function (3) and then extract the differential features between them through a multilayer perceptron(MLP).

$$\begin{aligned} \mathbf{h}^{K_t} &= \text{attention}(\{\mathbf{h}^{K_i^t}\}_{i=1}^D), \\ \mathbf{e}_t^k &= \text{Relu}(\text{FC}(\mathbf{h}^{K_t})) \end{aligned} \tag{5}$$

Topic-shifted Knowledge Selector: Given the current user utterance representation $\mathbf{h}^{X_t}$, the candidate knowledge representations $[\mathbf{h}^{K_1^t}, \ldots, \mathbf{h}^{K_D^t}]$, and the representations mentioned above $\mathbf{u}_i^t$, $\mathbf{e}_t^k$,we construct query vector and key vector as:

$$\begin{aligned} \mathbf{Q}_{sh} &= \text{MLP}([\mathbf{h}^{X_t}; \mathbf{e}_t^k]) \\ \mathbf{K}_i^{sh} &= \text{MLP}([\mathbf{u}_i^t; \mathbf{h}^{K_i^t}]) \\ \mathbf{K}_{sh} &= [\mathbf{K}_1^{sh}; \ldots; \mathbf{K}_D^{sh}] \end{aligned} \tag{6}$$

Given $\mathbf{Q}_{sh}$ and $\mathbf{K}_{sh}$, the topic-shifted knowledge selector predicts the distributions over the knowledge pool $\mathcal{K}_t$ by additive attention:

$$P(\mathcal{K}_t|S) = \text{SM}(v_0^\top \tanh(W_0 \mathbf{K}_{sh} + U_0 \mathbf{Q}_{sh})) \tag{7}$$

where v_0, W_0 and U_0 are trainable parameters.

Topic-inherited Knowledge Selector: Following (Meng et al., 2021), we apply a stack of transformer encoder blocks with positional embeddings

to integrate the previously selected knowledge and extract the inherited topic $\mathbf{h}_{trans}^{K'_{t-1}}$:

$$\begin{aligned}&[\mathbf{h}_{trans}^{K'_1};\ldots;\mathbf{h}_{trans}^{K'_{t-1}}] = \\ &\text{TransformerEncoder}([\mathbf{h}^{K'_1};\ldots;\mathbf{h}^{K'_{t-1}}])\end{aligned} \tag{8}$$

Then, given the representation of inherited topic $\mathbf{h}_{trans}^{K'_{t-1}}$, the candidate knowledge representations $[\mathbf{h}^{K_1^t},\ldots,\mathbf{h}^{K_D^t}]$ and the representation of knowledge difference $\mathbf{e}_t^k$, we construct query vector and key vector as:

$$\begin{aligned}\mathbf{Q}_{inh} &= \text{MLP}([\mathbf{h}_{trans}^{K_{t-1}};\mathbf{e}_t^k]) \\ \mathbf{K}_i^{inh} &= \text{MLP}(\mathbf{h}^{K_i^t}) \\ \mathbf{K}_{inh} &= [\mathbf{K}_1^{inh};\ldots;\mathbf{K}_D^{inh}]\end{aligned} \tag{9}$$

Similarly, the topic-inherited knowledge selector predicts the distributions $P(\mathcal{K}_t|inherit)$ by:

$$P(\mathcal{K}_t|I) = \text{SM}(v_1^\top \tanh(W_1\mathbf{K}_{inh} + U_1\mathbf{Q}_{inh})) \tag{10}$$

Topic Shift Discriminator: There are two topic shift discriminators at the training stage: teacher topic shift discriminator and student topic shift discriminator. The former is provided with the current golden knowledge as posterior information, and it generates soft labels of topic shift, which can guide the student model to distinguish noises. In section 3.3 we will explain how it works in detail. The latter generates hard labels indicating which Sub-KS works, and this corresponds to the 0/1 switch in Figure 1.

Given the integration of the previously selected knowledge $\mathbf{h}_{trans}^{K'_{t-1}}$and the current-turn user utterance representation $\mathbf{h}^{X_t}$, we first extract the topic information in the current user utterance $\mathbf{e}_t^x$ by a multilayer perceptron(MLP). Then we construct two discriminators based on whether they have access to posterior information as follows:

Teacher module:

$$\begin{aligned}\mathbf{v}_\mathcal{T} &= [\mathbf{h}^{K'_t};\mathbf{e}_t^x;\mathbf{h}_{trans}^{K'_{t-1}};\mathbf{e}_t^x - \mathbf{h}_{trans}^{K'_{t-1}};\mathbf{e}_t^x \odot \mathbf{h}_{trans}^{K'_{t-1}}] \\ P_\mathcal{T}(\mathcal{D}=S) &= \text{Sigmoid}(\text{FC}(\mathbf{v}_\mathcal{T}))\end{aligned} \tag{11}$$

Student module:

$$\begin{aligned}\mathbf{v}_\mathcal{T} &= [\mathbf{e}_t^x;\mathbf{h}_{trans}^{K'_{t-1}};\mathbf{e}_t^x - \mathbf{h}_{trans}^{K'_{t-1}};\mathbf{e}_t^x \odot \mathbf{h}_{trans}^{K'_{t-1}}] \\ P_\mathcal{S}(\mathcal{D}=S) &= \text{Sigmoid}(\text{FC}(\mathbf{v}_\mathcal{S}))\end{aligned} \tag{12}$$

where $\odot$ denotes element-wise product, and $\mathcal{D}=S$ means discrimination result is topic shift.

We train both of the two modules with Cross Entropy loss, $\hat{\mathbf{y}}_t$ represents the topic shift labels obtained by distant supervision:

$$L(\mathcal{D}) = -\frac{1}{|C|}\sum_{t=1}^{|C|}\hat{\mathbf{y}}_t \log(P(\mathcal{D})) \tag{13}$$

2.5 Decoder

We take GPT-2 as the backbone of Decoder. Similar to (Zhao et al., 2020b; Zheng and Huang, 2021), we define new tokens in the dictionary of GPT-2 Tokenizer: "<context>", "<response>" and "<knowledge>". These tokens are treated as segment embeddings to mark different information components.

At the training stage, the inputs of the decoder are the concatenation of the dialogue context, the golden knowledge and the responses. The training loss is the Cross Entropy on the responses. At the inference stage, given the selected knowledge K_{sel}^t and the dialogue context $\{(X_1,Y_1),\ldots,(X_{t-1},Y_{t-1}),X_t\}$, the decoder synthesizes the two parts to generate the current response until <eos>.

$$L_g = -\frac{1}{|Y_t|}\sum_{i=1}^{|Y_t|}\log(P(y_i^t|X_{\leq t},Y_{<t},K'_t,y_{<i}^t)) \tag{14}$$

2.6 Pseudo Label Learning

Our findings on topic shift in multi-round conversation suggest that topic shift helps knowledge selection. However, we still face a lack of labels. Inspired by distant supervision, we regard the entity words of retrieving knowledge candidates as topic words and obtain topic shift labels $\mathcal{D}'$. After that, we alleviate the noise problem in pseudo labels through curriculum learning and knowledge distillation. In this subsection, we describe our methods in detail.

Distant Supervision: During the construction of the KGDG dataset, the candidate knowledge is retrieved from Wikipedia by entity words through search engines. The entity word can retrieve the corresponding knowledge, which indicates that it highly summarizes the content of the knowledge sentence. However, the existing methods ignore this useful information. According to (Brown and Yule, 1983), the topic is the most frequently used term. We make a hypothesis according to the idea of distant supervision (Mintz et al., 2009): the entity used to search candidate is the topic word of

each knowledge sentence. Then, we mark topic shift labels according to the topic words. If the topic words of the current round appeared in the dialogue context, we mark the current round as topic inheritance with label 0; if the entity words of the current round have not appeared in the dialogue context, we mark the current round as topic shift with label 1. We will prove that the pseudo labels acquired in this way are instructing in section 3.6.

Noisy Label Learning: The hypothesis mentioned above is relatively strong. Therefore, we adopt curriculum learning (Bengio et al., 2009) and knowledge distillation (Hinton et al., 2015) methods to alleviate the noises and further optimize the model. Specifically, with the increase of training steps, our model gradually chooses the Sub-KS based on the output of the discriminator to reduce the dependence on the topic shift label $\mathcal{D}'$.

$$L_{CL} = -\frac{1}{|C|}\sum_{t=1}^{|C|}(p_i \log\left(P\left(\mathcal{K}_t\right)P\left(\mathcal{D}'\right)\right) + (1-p_i)\log\left(P\left(\mathcal{K}_t\right)P\left(\mathcal{D}_{\mathcal{S}}\right)\right)) \quad (15)$$

where p_i decreases with the training steps.

Besides, we distill knowledge between the teacher model and the student model through KL divergence loss:

$$L_{distill} = D_{KL}(P_{\mathcal{T}}||P_{\mathcal{S}}) \quad (16)$$

We train TAKE's knowledge selector in two stages:

$$\begin{aligned} &\text{Stage I: } L_{\mathcal{K}_1} = L_{ks} + \alpha(L_{\mathcal{T}}) \\ &\text{Stage II: } L_{\mathcal{K}_2} = L_{CL} + \alpha\left(L_{\text{distill}} + L_{\mathcal{S}}\right) \end{aligned} \quad (17)$$

and the original KS loss is defined as:

$$L_{ks} = -\frac{1}{|C|}\sum_{t=1}^{|C|}\hat{\mathbf{y}}_t \log\left(P\left(\mathcal{K}_t\right) \mid P\left(\mathcal{D}'\right)\right) \quad (18)$$

3 Experiments

3.1 Experimental Setup

Dataset. There are dozens of datasets chosen for evaluating the KGDG task before (Moghe et al., 2018; Zhou et al., 2018; Dinan et al., 2018). We take the most challenging one Wizard of Wikipedia(WoW) for experiments. The WoW data is obtained from a crowdsourcing data collection website. In data collection, the user side plays the role of the apprentice, and the agent side plays the role of the wizard. The wizard has access to the knowledge retrieved from Wikipedia as groundsource to generate informative responses, while the apprentice prefers speaking common utterances. In the WoW dataset, there are nearly 67 pieces of knowledge on average in a knowledge pool. The WoW dataset consists of 22,311 dialogues with 201,999 turns divided into training set/validation set/test set. The test set is further divided into test `seen` set and test `unseen` set. The conversation topics of the test `seen` set have appeared in the training set, while the topics of the test `unseen` set are brand-new. The latter contains out-of-domain data which is more challenging.

Baseline Models. We compare our TAKE model with several SOTA models, including:
MemNet: A model proposed by Dinan et al. (2018), which is regarded as the most basic baseline in the KGDG task.
SKLS: Kim et al. (2020) design the sequential latent knowledge selection model according to the idea of conditional variational auto-encoder.
DukeNet: Meng et al. (2020) design a knowledge selection network that takes knowledge tracking and knowledge transfer as a pair of dual tasks to provide feedback to each other.
DiffKS: A model considered the difference of knowledge between the two adjacent rounds of dialogue, proposed by Zheng et al. (2020). We take the decoupled version as one baseline.
KnowledGPT: Zhao et al. (2020b) design a joint training strategy, which uses the combination of RL and CL to improve the knowledge selection module and generation module. It takes GPT-2 as the decoder.
MIKe: Meng et al. (2021) introduce the concept of mixed-initiative into knowledge-grounded dialogue generation and train the initiative discriminator with a self-supervised learning strategy.
CoLV: A model proposed by Zhan et al. (2021a), the authors propose a collaborative latent variable model to integrate the diversity of KS and DG simultaneously in separate yet collaborative latent spaces.
MIKe + GPT2: We reinforce the MIKe model with GPT-2 as a strong baseline, and the KS module remains unchanged.

Evaluation Metrics. For automatic evaluation, we evaluate KS with accuracy and evaluate response generation quality with sentence-

Model	WoW Test Seen						WoW Test Unseen					
	BLEU-1	BLEU-4	RG-1	RG-L	METEOR	ACC	BLEU-1	BLEU-4	RG-1	RG-L	METEOR	ACC
MemNet	17.2	1.61	24.1	17.0	15.5	23.9	13.7	0.6	21.7	15.6	13.1	14.0
SKLS	18.9	1.8	24.5	17.6	16.0	26.8	17.3	1.1	21.0	16.1	13.7	18.3
DukeNet	18.6	2.6	25.4	18.8	17.3	26.2	16.3	1.8	23.2	16.9	15.4	20.1
DiffKS	18.8	2.2	24.8	17.9	16.8	25.6	17.4	1.7	23.6	16.8	14.7	19.8
MIKe	19.1	2.8	25.9	19.2	18.3	28.2	17.6	2.1	24.2	17.8	16.0	21.5
CoLV	-	2.9	20.6	-	-	**30.1**	-	2.1	19.7	-	-	18.9
KnowledGPT	19.5	-	24.7	-	-	28.0	17.7	-	22.3	-	-	25.4
MIKe + GPT2	20.4	3.3	26.7	20.2	19.4	28.2	18.8	2.5	25.1	18.6	17.4	21.5
TAKE	**20.8**	**3.6**	**27.1**	**20.5**	**19.9**	28.8	**20.1**	**3.3**	**26.2**	**19.7**	**18.9**	**25.8**

Table 2: Automatic Evaluation results on Wizard of Wikipedia.

Model	WoW Test Seen			WoW Test Unseen		
	informativeness	coherence	fluency	informativeness	coherence	fluency
Dukenet	1.63	1.96	1.69	1.57	1.89	1.61
MIKe	1.66	1.97	1.62	1.61	1.90	1.86
MIKe+GPT-2	1.84	2.09	**2.05**	1.70	2.15	2.27
TAKE	**1.88**	**2.14**	**2.05**	**1.90**	**2.21**	**2.29**

Table 3: Human Evaluation results on Wizard of Wikipedia. The improvement of TAKE to the best baseline (MIKe+GPT-2) is statistically significant (t-test with p-value < 0.05).

level BLEU-1 (Papineni et al., 2002), BLEU-4, ROUGE-1 (Lin, 2004), ROUGE-L and METEOR (Denkowski and Lavie, 2014). These metrics have been widely used in generation tasks before. For human evaluation, We randomly sample 50 responses in test `seen` set and 50 responses in test `unseen` set. By labeling manually, we find the proportion of the topic shift turn is close to the topic inheritance turn among these samples. Then we invite five knowledgeable annotators to score these samples in {0,1,2,3} considering the following three aspects: context coherence, fluency and informativeness(which response contains more knowledge and looks more informative). We compute Fleiss' kappa value (Fleiss, 1971) among different annotators to measure their agreement.

Implementation Details. We use PyTorch (Paszke et al., 2019) framework to implement our model. For the implementation of pre-training models BERT(110M) and GPT-2(117M), we utilize the open-source Hugging Face transformers (Wolf et al., 2020). The whole model is optimized with Adam (Kingma and Ba, 2014) algorithm and gradient clipping with a maximum gradient norm of 0.4. We use the gradient accumulation method (accumulation number is 16), and preprocess the knowledge pool by limiting the number of candidate knowledge to 32 and retaining the golden knowledge at the training stage to save GPU memory. The batch size is 2 for training KS and 4 for training DG. The learning rates are 1e-5 for BERT; 6e-5 for the topic-aware knowledge selector; 3e-5 for GPT-2. We adopt the linear scheduler with a warm-up strategy for the training of Bert and the knowledge selector. It takes total ten epochs for training stages I and II and five epochs for training the GPT-2 Decoder. The weight α in multi-task learning is set to 0.5. Our model is trained on one NVIDIA Geforce RTX 3090 GPU. For other settings, such as the hidden size, dropout rate, sentence length and so on, we keep consistent with MIKe.

3.2 Experimental Results

Table 2 demonstrates automatic evaluation results on WoW. Our model outperforms the typical baseline methods Memnet, SKLS, Dukenet and Diffks remarkably. These methods treat the dialogue context equally to construct query vectors, and the cursory construction hurts knowledge selection, thus leading to terrible generation results. In terms of KS performance, although TAKE has no obvious advantage in the test `seen` set compared with the strong baselines MIKe and Colv, it has a notable improvement in the test `unseen` set (+4.3% MIKe,+6.9% CoLV). We think that the promotion comes from a better location of history information of our method to conduct attention mechanism. Besides, TAKE has a significant improvement over all generation metrics, which indicates that TAKE can generate more informative and engaging responses. For fairness, we transplant the GPT-2 module in TAKE to MIKe to make a further com-

Model	WoW Test Seen						WoW Test Unseen					
	BLEU-1	BLEU-4	RG-1	RG-L	METEOR	ACC	BLEU-1	BLEU-4	RG-1	RG-L	METEOR	ACC
TAKE	**20.84**	**3.58**	**27.14**	**20.45**	**19.88**	**28.75**	**20.12**	**3.25**	**26.16**	**19.74**	**18.85**	**25.78**
-CL	20.65	3.41	27.08	20.24	19.69	28.22	19.95	3.17	26.05	19.62	18.76	25.11
-CL -distill	20.53	3.25	26.86	20.12	19.63	27.83	19.82	3.17	25.96	19.49	18.58	24.75
-label Soft	20.55	3.27	26.91	20.15	19.62	27.92	19.68	2.99	25.90	19.34	18.42	23.37
-label Hard	20.12	3.14	26.34	19.75	19.01	25.43	19.36	2.94	25.35	19.10	18.08	22.32

Table 4: Ablation study on the WoW dataset. -CL denotes removing the curriculum learning method. -distill denotes removing the teacher model and knowledge distillation method. -label Soft denotes removing the pseudo topic shift label for supervising and connecting the discriminator and Sub-KS with probability. -label hard denotes removing the pseudo topic shift label for supervising, and the discriminator still chooses Sub-KS through the gumbel-softmax trick.

parison. The results further emphasize that our topic-aware knowledge selector has stronger ability of KS than the mixed-initiative knowledge selector.

The human evaluation is shown in Table 3. The kappa values are between 0.65 and 0.75, denoting substantial agreement among the annotators .Thanks to GPT-2 model, TAKE is superior to the typical baselines on all metrics. As for the strong baseline MIKe+GPT2, although it is comparable with TAKE on fluency, our model exceeds it on informativeness and context coherence a lot. We think the reason is that TAKE selects knowledge which is more coherent with the dialogue context.

3.3 Ablation Study

In order to clarify the source of performance improvement in TAKE, we conduct ablations by removing particular modules from TAKE. The ablation results are shown in Table 4, which denotes all components are beneficial for TAKE. The two methods of noise alleviation in section 2.6 can improve the performance in the inference stage by making the model adapt to noises introduced by inaccurate discrimination in advance. Besides, because the topic transfer label is binary, the curriculum learning method can neutralize the noises in the pseudo labels. With the increase of the training rounds, TAKE has more possibilities on discriminating noisy samples which avoids the overfitting on wrong labels. The teacher model with posterior information learns part of the noises in the pseudo labels in advance and guides the student model to correct these samples in the form of soft labels. For the last two lines of the experiment, we find that the model tends to deteriorate without instructing labels. Only one of the two sub-KS is activated, and the other is idle. The reason is that the topic shift discriminator is not supervised by labels, hence only one of the two sub-KS has been fully trained. This set of experiments proves the importance of the pseudo labels.

3.4 Case Study

To better evaluate the performance of response generation, we randomly select some examples from the WoW dataset generated by DukeNet, MIKe, KnowledGPT and TAKE to make comparisons. In Table 5, the user asked when Instagram appeared at the current turn. Dukenet and the knowledGPT model do not capture this vital information to construct queries, so they select the wrong knowledge. MIKe mistakenly judges that the current round is an agent initiative dialogue because the connection between the history and current responses is smooth. Consequently, it selects knowledge about Instagram usage based on the previously selected knowledge. Only TAKE constructs the query vector based on the current user utterance and selects the most appropriate external knowledge. We post the remaining topic-inherited example in appendix A.2.

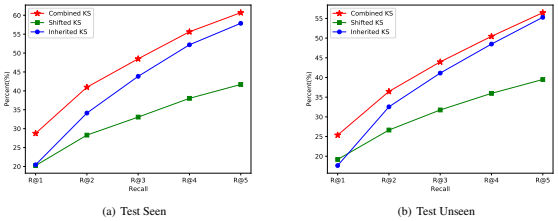

(a) Test Seen (b) Test Unseen

Figure 2: Recall experiment of TAKE on WoW dataset

3.5 Multi-Knowledge Integration Performance

By conducting experiments on Recall of KS and multi-sentence knowledge integration, we find that the one-to-many relations between the dialogue context and knowledge occur more frequently during topic inheritance. Under the framework we

Topic: Instagram	
K_0	no passages used.
K_1	Users can apply various digital filters to their images and add locations through geotags.
K_2	It was created by Kevin Systrom and Mike Krieger, and launched in October 2010 as a free mobile app exclusively for the iOS operating system.
K_3	They can add hashtags to their posts, linking the photos up to other content on Instagram featuring the same subject or overall topic.
	...
context	(1)Are you familiar with Instagram? (2) Oh yes, I love going on Instagram! (3) I think it makes the images look much more professional. (4) Yeah, they have many neat features. I wonder when it was started?
DukeNet	($K_0\times$) I'm not sure but I know that they are called " x ".
KnowledGPT	($K_0\times$) I'm not sure but I know that it was made in the world.
MIKe	($K_1\times$) I'm not sure, but I know that users can apply digital filters through geotags.
TAKE	($K_2\checkmark$) It was launched in October 2010 as a free mobile app exclusively for the iOS system.

Table 5: An example for case study. K_2 is the golden knowledge. Only TAKE focuses on the current user utterance to select the golden knowledge and integrates it in response.

Model	WoW Test Seen			WoW Test Unseen		
	BLEU-1	RG-1	METEOR	BLEU-1	RG-1	METEOR
MIKe+GPT2	20.44	26.74	19.42	18.77	25.06	17.40
MIKe+GPT2 R@2	20.28	26.65	19.27	18.76	24.92	17.47
TAKE	20.84	27.14	19.88	20.12	26.16	18.85
TAKE R@2	20.65	27.00	20.11	19.92	26.15	18.85
TAKE Inh R@2	20.89	27.24	20.14	20.21	26.26	18.94

Table 6: Top-2 knowledge integrated evaluation results on WoW. R@2 denotes integrating two knowledge; TAKE Inh R@2 denotes TAKE's topic-inherited KS integrates two knowledge.

proposed, the topic-inherited selector inherits topics in previous rounds. Even though TAKE does not select the golden knowledge, it chooses rational knowledge with closing topics. The golden knowledge probably ranks at the k-th positions (k>1). As shown in Figure 2, with the increase of Recall, the curve of the topic-inherited selector rises steeper, and the improvement of KS comes from topic-inherited rounds.

Apart from that, we find our topic-aware framework is very suitable for explicit multi-knowledge integration. Studies have shown that increasing the number of knowledge integrated into dialogue generation blindly hurts the generation performance (Bruyn et al., 2020), which can be confirmed by the first four lines in Table 6. This is because there is roughly no relationship between the rank-1 and rank-2 items of knowledge. However, the generation results improve if we recall top-2 knowledge only for topic-inherited rounds. The phenomenon explains that TAKE is robust, and it can utilize multi-knowledge effectively.

3.6 Analysis of Pseudo Label and Noisy Label Learning

To prove the effectiveness of our pseudo labels and noisy label learning methods, we further conduct experiments by replacing the output of the discriminator with random 0/1 labels or different proportion of pseudo labels at the inference stage. Table 7 and Figure 3 exhibit the results. If we use random topic-shift label, the Sub-KS performs terribly with the wrong part of history information. By comparison, the last line in Table 7 denotes that if TAKE can learn to discriminate topic shift the same as pseudo labels, the performance will improve much better. Apart from that, the rising trends in Figure 3 apparently shows that the more proportion of pseudo labels the model obtains, the better results it performs. In our experiments, TAKE's topic shift discriminator can predict about 78 percent pseudo labels after training. However, its knowledge selection accuracy is significantly higher than 78 percent pseudo-labels-given experiment during inference on both the `seen` and `unseen` test set. The phenomenon demonstrates that the curriculum learning and knowledge distillation methods alleviate the noisy pseudo label problem, and then get fine performance which can only be reached under a higher proportion of pseudo labels.

4 Related Work

Knowledge-grounded dialogue generation In recent years, the KGDG task has been a hot spot of research, and many new datasets have emerged (Zhou et al., 2018; Dinan et al., 2018; Eric et al., 2021; Komeili et al., 2022). The existing work of

Model	WoW Test Seen						WoW Test Unseen					
	BLEU-1	BLEU-4	RG-1	RG-L	METEOR	ACC	BLEU-1	BLEU-4	RG-1	RG-L	METEOR	ACC
TAKE(78%)	20.84	3.58	27.14	20.45	19.88	28.75	20.12	3.25	26.16	19.74	18.85	25.78
Random	19.58	2.95	25.84	19.40	18.57	23.60	18.59	2.64	24.70	18.67	17.35	18.04
80%	20.59	3.37	26.94	20.17	19.7	27.99	19.96	3.19	26.0	19.61	18.78	25.13
90%	20.98	3.65	27.37	20.61	20.03	29.97	20.3	3.32	26.37	19.93	19.04	27.34
Ideal(100%)	**21.22**	**3.79**	**27.70**	**20.86**	**20.40**	**32.63**	**20.44**	**3.45**	**26.66**	**20.13**	**19.31**	**28.63**

Table 7: More experiments on Wizard of Wikipedia. Random denotes that TAKE decides Sub-KS randomly. Ideal denotes that TAKE decides Sub-KS entirely depending on pseudo labels at the inference stage. 80% denotes that TAKE decides Sub-KS depending on 80% accurate pseudo labels.

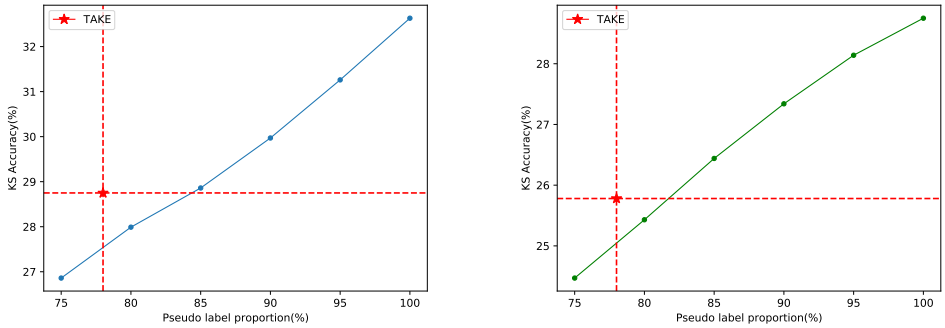

(a) Test Seen (b) Test Unseen

Figure 3: Analysis of model with different pseudo label proportions on WOW. The red dots indicate experiment results under normal settings.

KGDG has three improvement directions: improving the accuracy of knowledge selection; improving the integration of external knowledge in generation (Zheng et al., 2021; Cui et al., 2021; Zhao et al., 2020b); improving the low-resource scenarios performance (Zheng and Huang, 2021; Zhao et al., 2020a; Liu et al., 2021). We mainly focus on the first direction. Lian et al. (2019) first proposed to utilize posterior knowledge to improve KS. Following (Lian et al., 2019), Kim et al. (2020) propose a sequential latent knowledge selection model; Chen et al. (2020b) attempt to bridge the gap between prior and posterior knowledge selection; Zhan et al. (2021a) find that sampling latent variable also helps response generation and proposed a collaborative latent variable model. Other work discovers more features in dialogue context to model KS (Zheng et al., 2020; Meng et al., 2020).

Topic-shift related works There is no general definition for "topic" (Purver et al., 2011). However, the definition given by Owen (1985) inspires our work in this paper. Although there have been various types of research about topic (Glavas and Somasundaran, 2020; Si et al., 2021), there is little work combining topic shift with multi-round dialogue. Xie et al. (2021) introduce a new topic-shift aware dialog benchmark TIAGE and three tasks. Sevegnani et al. (2021) propose a new dialogue connection task when the topic shifts. Zhan et al. (2021b) utilizes a BiLSTM-CRF network to predict topic tags before knowledge selection.

5 Conclusion

In this paper, we propose a Topic-shift Aware Knowledge sElector(TAKE) model, which better locates the relevant parts from dialogue history to improve the performance of knowledge selection. Besides, we obtain topic shift labels inspired by the idea of distant supervision and adopt curriculum learning and knowledge distillation methods to alleviate the negative influence of noises. Experiments on the WoW show that our model outperforms the baselines, and the ablation study indicates that all components of our methods work. In the future, we will research on multi-sentence knowledge integration further and combine the labeling work with knowledge graph.

Acknowledgments

This work was supported by National Natural Science Foundation of China (No. 61976207, No. 61906187).

References

Yoshua Bengio, Jérôme Louradour, Ronan Collobert, and Jason Weston. 2009. Curriculum learning. In *Proceedings of the 26th Annual International Conference on Machine Learning*, ICML '09, page 41–48, New York, NY, USA. Association for Computing Machinery.

Gillian R Brown and George Yule. 1983. Brown, gillian, and george yule, discourse analysis . cambridge, ma: Cambridge university press, 1983.

Maxime De Bruyn, Ehsan Lotfi, Jeska Buhmann, and Walter Daelemans. 2020. Bart for knowledge grounded conversations. In *Converse@KDD*.

Daniel Cer, Yinfei Yang, Sheng-yi Kong, Nan Hua, Nicole Limtiaco, Rhomni St. John, Noah Constant, Mario Guajardo-Cespedes, Steve Yuan, Chris Tar, Yun-Hsuan Sung, Brian Strope, and Ray Kurzweil. 2018. Universal sentence encoder. *CoRR*, abs/1803.11175.

Xiuyi Chen, Feilong Chen, Fandong Meng, Peng Li, and Jie Zhou. 2021. Unsupervised knowledge selection for dialogue generation. In *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*, pages 1230–1244, Online. Association for Computational Linguistics.

Xiuyi Chen, Fandong Meng, Peng Li, Feilong Chen, Shuang Xu, Bo Xu, and Jie Zhou. 2020a. Bridging the gap between prior and posterior knowledge selection for knowledge-grounded dialogue generation. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 3426–3437, Online. Association for Computational Linguistics.

Xiuyi Chen, Fandong Meng, Peng Li, Feilong Chen, Shuang Xu, Bo Xu, and Jie Zhou. 2020b. Bridging the gap between prior and posterior knowledge selection for knowledge-grounded dialogue generation. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 3426–3437, Online. Association for Computational Linguistics.

Leyang Cui, Yu Wu, Shujie Liu, and Yue Zhang. 2021. Knowledge enhanced fine-tuning for better handling unseen entities in dialogue generation. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 2328–2337, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Michael Denkowski and Alon Lavie. 2014. Meteor universal: Language specific translation evaluation for any target language. In *Proceedings of the Ninth Workshop on Statistical Machine Translation*, pages 376–380, Baltimore, Maryland, USA. Association for Computational Linguistics.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. BERT: pre-training of deep bidirectional transformers for language understanding. *CoRR*, abs/1810.04805.

Emily Dinan, Stephen Roller, Kurt Shuster, Angela Fan, Michael Auli, and Jason Weston. 2018. Wizard of wikipedia: Knowledge-powered conversational agents. *CoRR*, abs/1811.01241.

Mihail Eric, Nicole Chartier, Behnam Hedayatnia, Karthik Gopalakrishnan, Pankaj Rajan, Yang Liu, and Dilek Hakkani-Tur. 2021. Multi-sentence knowledge selection in open-domain dialogue. In *Proceedings of the 14th International Conference on Natural Language Generation*, pages 76–86, Aberdeen, Scotland, UK. Association for Computational Linguistics.

Joseph L. Fleiss. 1971. Measuring nominal scale agreement among many raters. *Psychological Bulletin*, 76:378–382.

Marjan Ghazvininejad, Chris Brockett, Ming-Wei Chang, Bill Dolan, Jianfeng Gao, Wen-tau Yih, and Michel Galley. 2017. A knowledge-grounded neural conversation model.

Goran Glavas and Swapna Somasundaran. 2020. Two-level transformer and auxiliary coherence modeling for improved text segmentation. In *AAAI*.

Geoffrey E. Hinton, Oriol Vinyals, and Jeffrey Dean. 2015. Distilling the knowledge in a neural network. *ArXiv*, abs/1503.02531.

Byeongchang Kim, Jaewoo Ahn, and Gunhee Kim. 2020. Sequential latent knowledge selection for knowledge-grounded dialogue. *CoRR*, abs/2002.07510.

Diederik Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. *International Conference on Learning Representations*.

Mojtaba Komeili, Kurt Shuster, and Jason Weston. 2022. Internet-augmented dialogue generation. *ArXiv*, abs/2107.07566.

Zekang Li, Cheng Niu, Fandong Meng, Yang Feng, Qian Li, and Jie Zhou. 2019. Incremental transformer with deliberation decoder for document grounded conversations. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 12–21, Florence, Italy. Association for Computational Linguistics.

Rongzhong Lian, Min Xie, Fan Wang, Jinhua Peng, and Hua Wu. 2019. Learning to select knowledge for response generation in dialog systems. In *Proceedings of the Twenty-Eighth International Joint Conference on Artificial Intelligence, IJCAI-19*, pages 5081–5087. International Joint Conferences on Artificial Intelligence Organization.

Chin-Yew Lin. 2004. ROUGE: A package for automatic evaluation of summaries. In *Text Summarization Branches Out*, pages 74–81, Barcelona, Spain. Association for Computational Linguistics.

Shilei Liu, Xiaofeng Zhao, Bochao Li, Feiliang Ren, Longhui Zhang, and Shujuan Yin. 2021. A Three-Stage Learning Framework for Low-Resource Knowledge-Grounded Dialogue Generation. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 2262–2272, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Chuan Meng, Pengjie Ren, Zhumin Chen, Zhaochun Ren, Tengxiao Xi, and Maarten de Rijke. 2021. *Initiative-Aware Self-Supervised Learning for Knowledge-Grounded Conversations*, page 522–532. Association for Computing Machinery, New York, NY, USA.

Chuan Meng, Pengjie Ren, Zhumin Chen, Weiwei Sun, Zhaochun Ren, Zhaopeng Tu, and Maarten de Rijke. 2020. *DukeNet: A Dual Knowledge Interaction Network for Knowledge-Grounded Conversation*, page 1151–1160. Association for Computing Machinery, New York, NY, USA.

Mike Mintz, Steven Bills, Rion Snow, and Daniel Jurafsky. 2009. Distant supervision for relation extraction without labeled data. In *Proceedings of the Joint Conference of the 47th Annual Meeting of the ACL and the 4th International Joint Conference on Natural Language Processing of the AFNLP*, pages 1003–1011, Suntec, Singapore. Association for Computational Linguistics.

Nikita Moghe, Siddhartha Arora, Suman Banerjee, and Mitesh M. Khapra. 2018. Towards exploiting background knowledge for building conversation systems. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 2322–2332, Brussels, Belgium. Association for Computational Linguistics.

Marion Owen. 1985. G. brown and g. yule, discourse analysis. cambridge: Cambridge university press, 1983. pp. xii 288. - m. stubbs, discourse analysis. oxford: Basil blackwell, 1983. pp. xiv 272. *Journal of Linguistics*, 21(1):241–245.

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. Bleu: a method for automatic evaluation of machine translation. In *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics*, pages 311–318, Philadelphia, Pennsylvania, USA. Association for Computational Linguistics.

Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Köpf, Edward Z. Yang, Zach DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. 2019. Pytorch: An imperative style, high-performance deep learning library. *CoRR*, abs/1912.01703.

Matthew Purver, Arash Eshghi, and Julian Hough. 2011. Incremental semantic construction in a dialogue system. In *Proceedings of the Ninth International Conference on Computational Semantics (IWCS 2011)*.

Karin Sevegnani, David M. Howcroft, Ioannis Konstas, and Verena Rieser. 2021. OTTers: One-turn topic transitions for open-domain dialogue. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 2492–2504, Online. Association for Computational Linguistics.

Jiasheng Si, Deyu Zhou, Tongzhe Li, Xingyu Shi, and Yulan He. 2021. Topic-aware evidence reasoning and stance-aware aggregation for fact verification. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 1612–1622, Online. Association for Computational Linguistics.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Remi Louf, Morgan Funtowicz, Joe Davison, Sam Shleifer, Patrick von Platen, Clara Ma, Yacine Jernite, Julien Plu, Canwen Xu, Teven Le Scao, Sylvain Gugger, Mariama Drame, Quentin Lhoest, and Alexander Rush. 2020. Transformers: State-of-the-art natural language processing. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 38–45, Online. Association for Computational Linguistics.

Huiyuan Xie, Zhenghao Liu, Chenyan Xiong, Zhiyuan Liu, and Ann A. Copestake. 2021. Tiage: A benchmark for topic-shift aware dialog modeling. *ArXiv*, abs/2109.04562.

Haolan Zhan, Lei Shen, Hongshen Chen, and Hainan Zhang. 2021a. CoLV: A collaborative latent variable model for knowledge-grounded dialogue generation. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 2250–2261, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Haolan Zhan, Hainan Zhang, Hongshen Chen, Zhuoye Ding, Yongjun Bao, and Yanyan Lan. 2021b. Augmenting knowledge-grounded conversations with sequential knowledge transition. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 5621–5630, Online. Association for Computational Linguistics.

Xueliang Zhao, Wei Wu, Chongyang Tao, Can Xu, Dongyan Zhao, and Rui Yan. 2020a. Low-resource knowledge-grounded dialogue generation. *ArXiv*, abs/2002.10348.

Xueliang Zhao, Wei Wu, Can Xu, Chongyang Tao, Dongyan Zhao, and Rui Yan. 2020b. Knowledge-grounded dialogue generation with pre-trained language models. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 3377–3390, Online. Association for Computational Linguistics.

Chujie Zheng, Yunbo Cao, Daxin Jiang, and Minlie Huang. 2020. Difference-aware knowledge selection for knowledge-grounded conversation generation. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 115–125, Online. Association for Computational Linguistics.

Chujie Zheng and Minlie Huang. 2021. Exploring prompt-based few-shot learning for grounded dialog generation. *CoRR*, abs/2109.06513.

Wen Zheng, Natasa Milic-Frayling, and Ke Zhou. 2021. Knowledge-grounded dialogue generation with term-level de-noising. In *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*, pages 2972–2983, Online. Association for Computational Linguistics.

Kangyan Zhou, Shrimai Prabhumoye, and Alan W Black. 2018. A dataset for document grounded conversations. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 708–713, Brussels, Belgium. Association for Computational Linguistics.

A Example Appendix

A.1 Demonstrations of Pseudo Label

Table 8 shows a case on how we obtain pseudo labels.

A.2 More Cases

Example in Table 9 indicates that even though MIKe could select the golden knowledge like TAKE, TAKE has a better ability to integrate the ground source. MIKe tends to repeat the golden knowledge simply while TAKE generates more coherent and fluent responses.

Topic: Canada	
Agent	(1) I have many friends in Canada, located in the northern part of North America. Knowledge: <Canada> Canada is a country located in the northern part of North America. **Pseudo label: 0**
User	(2) I don't know anyone from Canada, what's it like?
Agent	(3) It has 10 provinces. Knowledge: <Canada> Its ten provinces and three territories extend from the Atlantic to the Pacific and northward into the Arctic Ocean, covering , making it the world's second-largest country by total area. **Pseudo label: 1**
User	(4) In which provinces are some of your friends?
Agent	(5) My friend lives in a place called Guelph. Canada is sparsely populated, most of its land is dominated by forest. Knowledge: <Canada> Canada is sparsely populated, the majority of its land territory being dominated by forest and tundra and the Rocky Mountains. **Pseudo label: 1**
User	(6) In Guelph, is it too cold?
Agent	(7)It can be very cold there, especially in Wellington County. Knowledge: <Guelph> The area – much of what became Wellington County – had been part of the Halton Block, a Crown Reserve for the Six Nations Iroquois. **Pseudo label: 0**
User	(8) What are some interesting places in Wellington County?
Agent	(9) Not many interesting places, the population there is only 90k. Knowledge: <Wellington County, Ontario> According to the 2016 Census, the population of the County was 90,932. **Pseudo label: 0**

Table 8: An example of labelling on the WoW dataset.

Topic: Horse training	
K_0	no passages used.
K_1	There is tremendous controversy over various methods of horse training and even some of the words used to describe these methods.
K_2	Historically, horses were trained for warfare, farm work, sport and transport.
K_3	Horses are trained to be manageable by humans for everyday care as well as for equestrian activities from horse racing to therapeutic horseback riding for people with disabilities.
K_4	Horse training refers to a variety of practices that teach horses to perform certain behaviors when asked to do so by humans.
	...
context	(1)It amazes me that some can take a wild horse and train it to be so obedient. Do you know anything about horse training? (2)Yes! I'm a horse trainer! I use a variety of practices that teach horses to perform behaviors when asked. I love my job. Horses are such smart animals! (3) Please tell me more about what you do. I love horses and would love to know more.
DukeNet	($K_2\times$) Well horses were trained for warfare, farm work, sport and transport.
KnowledGPT	($K_2\times$) Well, I love horses, they were trained for warfare, farm work and transport.
MIKe	($K_3\checkmark$) Well, horses are trained to be manageable by humans for everyday care as well as therapeutic horseback riding.
TAKE	($K_3\checkmark$) I love to train themto be manageable by humans for everyday care. I also love to ride them for therapeutic horseback riding.

Table 9: An example for case study. The utterances (2) to (4) exhibit a topic inheritance.

Dynamic Dialogue Policy for Continual Reinforcement Learning

Christian Geishauser, Carel van Niekerk, Hsien-Chin Lin,
Nurul Lubis, Michael Heck, Shutong Feng, Milica Gašić
Heinrich Heine University Düsseldorf, Germany
{geishaus, niekerk, linh, lubis, heckmi, fengs, gasic}@hhu.de

Abstract

Continual learning is one of the key components of human learning and a necessary requirement of artificial intelligence. As dialogue can potentially span infinitely many topics and tasks, a task-oriented dialogue system must have the capability to continually learn, dynamically adapting to new challenges while preserving the knowledge it already acquired. Despite the importance, continual reinforcement learning of the dialogue policy has remained largely unaddressed. The lack of a framework with training protocols, baseline models and suitable metrics, has so far hindered research in this direction. In this work we fill precisely this gap, enabling research in dialogue policy optimisation to go from static to dynamic learning. We provide a continual learning algorithm, baseline architectures and metrics for assessing continual learning models. Moreover, we propose the dynamic dialogue policy transformer (DDPT), a novel dynamic architecture that can integrate new knowledge seamlessly, is capable of handling large state spaces and obtains significant zero-shot performance when being exposed to unseen domains, without any growth in network parameter size. We validate the strengths of DDPT in simulation with two user simulators as well as with humans.

1 Introduction

Task-oriented dialogue systems are characterised by an underlying task or a goal that needs to be achieved during the conversation, such as managing a schedule or finding and booking a restaurant. Modular dialogue systems have a tracking component that maintains information about the dialogue in a belief state, and a planning component that models the underlying policy, i.e., the selection of actions (Levin and Pieraccini, 1997; Roy et al., 2000; Williams and Young, 2007; Zhang et al., 2020b). The spectrum of what a task-oriented dialogue system can understand and talk about is defined by an ontology. The ontology defines domains such as restaurants or hotels, slots within a domain such as the area or price, and values that a slot can take, such as the area being west and the price being expensive. As dialogue systems become more popular and powerful, they should not be restricted by a static ontology. Instead, they should be dynamic and grow as the ontology grows, allowing them to comprehend new information and talk about new topics – just like humans do.

In the literature, this is referred to as continual learning (Biesialska et al., 2020; Khetarpal et al., 2020a; Hadsell et al., 2020). A learner is typically exposed to a sequence of tasks that have to be learned in a sequential order. When faced with a new task, the learner should leverage its past knowledge (*forward transfer*) and be flexible enough to rapidly learn how to solve the new task (*maintain plasticity*). On the other hand, we must ensure that the learner does not forget how to solve previous tasks while learning the new one (prevent *catastrophic forgetting*). Rather, a learner should actually improve its behaviour on previous tasks after learning a new task, if possible (*backward transfer*).

Despite progress in continual learning (Lange et al., 2019; Parisi et al., 2019; Biesialska et al., 2020; Khetarpal et al., 2020a; Hadsell et al., 2020), there is – to the best of our knowledge – no work that addresses continual reinforcement learning (continual RL) of the dialogue policy, even though the policy constitutes a key component of dialogue systems. Research in this direction is hindered by the lack of a framework that provides suitable models, evaluation metrics and training protocols.

In modular task-oriented dialogue systems the input to the dialogue policy can be modelled in many different ways (Lipton et al., 2018; Weisz et al., 2018; Takanobu et al., 2019; Wang et al., 2015; Casanueva et al., 2018; Xu et al., 2020). An appropriate choice of state representation is key

Proceedings of the 29th International Conference on Computational Linguistics, pages 266–284
October 12–17, 2022.

to the success of any form of RL (Madureira and Schlangen, 2020). In continual RL for the dialogue policy, this choice is even more essential. Different dialogue domains typically share structure and behaviour that should be reflected in the state and action representations. The architecture needs to exploit such common structure, to the benefit of any algorithm applied to the model. In this work, we therefore centre our attention on this architecture. We contribute [1]

- the first framework for continual RL to optimise the dialogue policy of a task-oriented dialogue system, two baseline architectures, an implementation of the state-of-the-art continual RL algorithm (Rolnick et al., 2018) and continual learning metrics for evaluation based on Powers et al. (2021), and
- a further, more sophisticated, new continual learning architecture based on the transformer encoder-decoder (Vaswani et al., 2017) and description embeddings, which we call dynamic dialogue policy transformer (DDPT). Our architecture can seamlessly integrate new information, has significant zero-shot performance and can cope with large state spaces that naturally arise from a growing number of domains while maintaining a fixed number of network parameters.

2 Related Work

2.1 Continual Learning in Task-oriented Dialogue Systems

Despite progress in continual learning, task-oriented dialogue systems have been barely touched by the topic. Lee (2017) proposed a task-independent neural architecture with an action selector. The action selector is a ranking model that calculates similarity between state and candidate actions. Other works concentrated on dialogue state tracking (Wu et al., 2019) or natural language generation (Mi et al., 2020; Geng et al., 2021). Geng et al. (2021) proposed a network pruning and expanding strategy for natural language generation. Madotto et al. (2021) introduced an architecture called AdapterCL and trained it in a supervised fashion for intent prediction, state tracking, generation and end-to-end learning. However, that work focused on preventing catastrophic forgetting and did not address the dialogue policy. As opposed to the above-mentioned approaches, we consider continual RL to optimise a dialogue policy.

2.2 Dialogue Policy State Representation

In the absence of works that directly address continual learning for the dialogue policy, it is worth looking at approaches that allow dialogue policy adaptation to new domains and examining them in the context of continual learning requirements.

The first group among these methods introduces new parameters to the model when the domain of operation changes. The approaches directly vectorise the belief state, hence the size of the input vector depends on the domain (as different domains for instance have different numbers of slots) (Su et al., 2016; Lipton et al., 2018; Weisz et al., 2018; Takanobu et al., 2019; Zhu et al., 2020). In the context of continual learning such approaches would likely preserve the plasticity of the underlying RL algorithm but would score poorly on forward and backward transfer.

Another group of methods utilises a hand-coded domain-independent feature set that allows the policy to be transferred to different domains (Wang et al., 2015; Casanueva et al., 2018; Chen et al., 2018; Chen et al., 2020; Lin et al., 2021). This is certainly more promising for continual learning, especially if the requirement is to keep the number of parameters bounded. However, while such models might score well on forward and backward transfer, it is possible that the plasticity of the underlying RL algorithm is degraded. Moreover, developing such features requires manual work and it is unclear if they would be adequate for any domain.

Xu et al. (2020) go a step further in that direction. They propose the usage of embeddings for domains, intents, slots and values in order to allow cross-domain transfer. To deal with the problem of a growing state space with an increased number of domains, they propose a simple averaging mechanism. However, as the number of domains becomes larger, averaging will likely result in information loss. Moreover, their architecture still largely depends on predefined feature categories.

A third option is to exploit similarities between different domains while learning about a new domain. Gašić et al. (2015) use a committee of Gaussian processes together with designed kernel functions in order to define these similarities and therefore allow domain extension and training on new

[1] https://doi.org/10.5281/zenodo.7075192

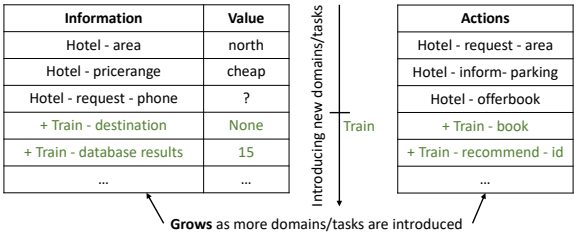

Figure 1: The amount of information that the dialogue agent must comprehend and the possible actions it can take increases as new domains/tasks are introduced.

domains. A similarity-based approach could in principle score well on all three continual learning measures. However, it is desirable to minimise the amount of manual work needed to facilitate continual learning.

2.3 Dialogue Policy Action Prediction

In the realm of domain adaptation, works assume a fixed number of actions that are slot-independent, and focus on the inclusion of slot-dependent actions when the domain changes (Wang et al., 2015; Casanueva et al., 2018; Chen et al., 2018; Chen et al., 2020; Lin et al., 2021). This allows seamless addition of new slots, but the integration of new intents or slot-independent actions requires an expansion of the model.

Works that allow new actions to be added to the action set compare the encoded state and action embeddings with each other (Lee, 2017; Xu et al., 2020; Vlasov et al., 2019), suggesting that exploiting similarities is key not only for state representations but also for action prediction.

With multi-domain dialogues it becomes necessary to be able to produce more than one action in a turn, which is why researchers started to use recurrent neural network (RNN) models to produce a sequence of actions in a single turn (Shu et al., 2019; Zhang et al., 2020a). RNNs are known however to only provide a limited context dependency.

3 Background

3.1 Continual Reinforcement Learning

In typical RL scenarios, an agent interacts with a stationary MDP $\mathcal{M} = \langle \mathcal{S}, \mathcal{A}, p, p_0, r \rangle$, where $\mathcal{S}$ and $\mathcal{A}$ constitute the state and action space of the agent, $p(s'|s,a)$ models the probability of transitioning to state s' after executing action a in state s, and $p_0(s)$ is the probability of starting in state s. The reward function r defines the observed reward in every time-step. The goal is to maximise the cumulative sum of rewards in that MDP.

In contrast, continual reinforcement learning focuses on non-stationary or changing environments (Hadsell et al., 2020). Generally speaking, the agent faces a sequence of Markov decision processes $\{\mathcal{M}_z\}_{z=1}^{\infty}$ (Lecarpentier and Rachelson, 2019; Chandak et al., 2020; Khetarpal et al., 2020b) with possibly different transition dynamics, reward functions or even state or action spaces. The variable z is often referred to as a task (or context) (Caccia et al., 2020; Normandin et al., 2021). While the MDP can change from episode to episode, it is often assumed that the agent is exposed to a fixed MDP for a number of episodes and then switches to the next MDP. Once a new task (or MDP) is observed, the old task is either never observed again or only periodically (Rolnick et al., 2018; Powers et al., 2021). The goal is to retain performance on all seen tasks. This requires the model to prevent catastrophic forgetting of old tasks while at the same time adapting to new tasks.

A state-of-the art method for continual RL that uses a replay memory is CLEAR (Rolnick et al., 2018). CLEAR manages the trade-off between preventing catastrophic forgetting and fast adaptation through an on-policy update step as well as an off-policy update step. The on-policy step is supposed to adapt the policy to the recent task by using the most recent dialogues while the off-policy step should lead to retaining performance on old tasks by updating on old experiences from the replay buffer. The off-policy update is further regularized such that policy and critic outputs are close to the historical prediction. More information on CLEAR is provided in the Appendix A.1.

In the context of dialogue, a task usually refers to a domain as defined in Madotto et al. (2021) and we will use these two terms interchangeably. As an example setting, a dialogue system is tasked with fulfilling user goals concerning hotel information and booking and after some amount of time with fulfilling goals related to train bookings. In terms of MDPs, the dialogue system first faces the MDP $\mathcal{M}_{z_1}, z_1 =$ hotel, for some amount of dialogues and afterwards $\mathcal{M}_{z_2}, z_2 =$ train. Once the train domain is introduced, the state and action space grows (as a result of the growing ontology) as depicted exemplarily in Figure 1. As a consequence, the model needs to understand new topics such as

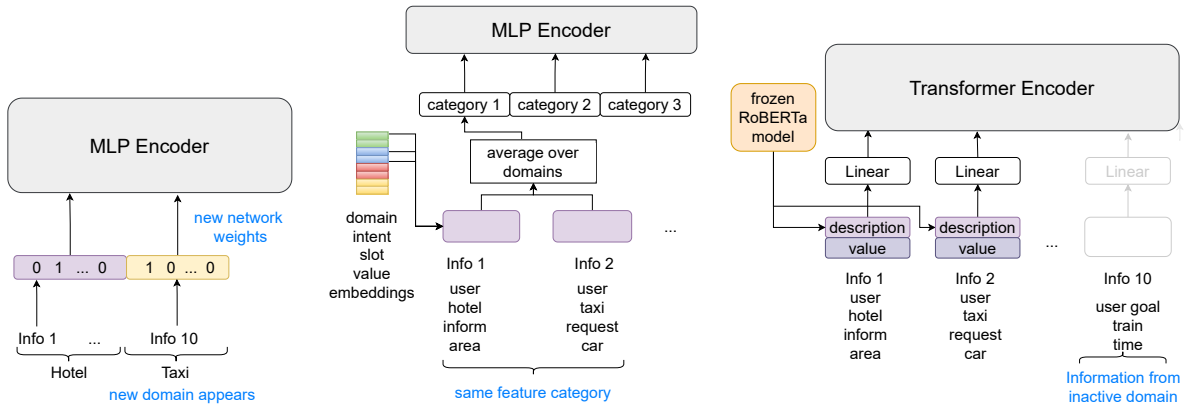

(a) Binary representation (Bin) (b) Semantic features (Sem) (c) Descriptions and values (DDPT)

Figure 2: State representation for different architectures. (a) *Bin* uses a flattened dialogue state with binary features, where the input size grows and new network weights need to be added when facing a new domain. (b) *Sem* uses the idea from Xu et al. (2020), using trainable embeddings for domain, intent, slot and value. The information corresponding to a specific feature category is then averaged over domains in order to be independent on the number of domains. (c) Our proposed DDPT model uses descriptions for every information which are embedded using a pretrained language model. The embedded description together with a value for the information is then fed into a linear layer and a transformer encoder.

the destination of the train and select new actions such as booking a train. In addition, the probability distributions p and p_0 of $\mathcal{M}_{z_2}$ are different compared to $\mathcal{M}_{z_1}$ since the probability that the user talks about hotels should be close to 0 while the probability that the agent's states contain information related to trains is close to 1.0.

3.2 Dialogue Policy in Modular Systems

In modular task-oriented dialogue systems, the decision of a dialogue policy is commonly based on the hidden information state of the dialogue system. This hidden information state, according to Young et al. (2007), should consist of the following information: the predicted user action, the predicted user goal and a representation of the dialogue history. For reactive behaviour by the policy, the user action is important as it includes information related to requests made by the user. The predicted user goal summarises the current goal of the user, including specified constraints. Lastly, the dialogue history representation captures the relevant information mentioned in the dialogue history, such as the latest system action. The state can also include the likelihood of the predicted acts, goal and dialogue history in the form of confidence scores. Moreover, the state often contains information about the database, for instance the number of entities that are available given the current predicted user goal.

Each domain that the system can talk about is either active, meaning that it has already been mentioned by the user, or inactive. The active domains can be derived from the user acts, from the user goal or tracked directly (van Niekerk et al., 2021).

Finally, the policy is supposed to take actions. As in (Shu et al., 2019; Zhang et al., 2020a), each action can be represented as a sequence of tuples $(domain, intent, slot)$. For instance, an action could be that the system requests the desired arrival time of a train or asks for executing a payment.

4 Dynamic Dialogue Policy Transformer

Our goal is to build a model that can talk about a potentially very large number of domains and is able to deal with new domains and domain extensions seamlessly without requiring any architectural changes. In particular, the number of model parameters should remain fixed. This is challenging since new domains require understanding of previously unseen information and the ability to talk about new topics.

Our approach is inspired by the way an employee would explain and act upon a novel task: 1) describe the information that can be used and the actions that can be taken in natural language, 2) restrict the focus to the information that is important for solving the task at hand, 3) when an action needs to be taken, this action is based on the information that was attended to (e.g. for the action to request the area, one would put attention on the information whether the area is already given). We propose an architecture that uses the transformer

encoder with information embeddings (Section 4.1 and Figure 2(c)) to fulfill 1) and 2) and the transformer decoder that leverages the domain gate (Section 4.2, 4.3 and Figure 3) to fulfill 3), which we call *dynamic dialogue policy transformer* (DDPT).

4.1 State Representation

Recall from Section 3.2 that the agent is provided with information on various concepts f for domain d_f: the user goal (domain-slot pairs), the user action (intents) and the dialogue history (system intents and database results). We assume that the agent has access to an external dictionary providing a natural language description $descr_f$ of each of these, e.g. "area of the hotel" or "number of hotel database results", which is common in dialogue state tracking (Rastogi et al., 2020; van Niekerk et al., 2021; Lee et al., 2021). See Appendix A.5 for the full list of descriptions. During a dialogue, the dialogue state or belief tracker assigns numerical values v_f, e.g. confidence scores for user goals or the number of data base results, etc. For every concept f we define the information embedding

$$\boldsymbol{e}_{\text{info}_f} = \text{Lin}\left(\left[\overline{\text{LM}}(descr_f), \text{Lin}(v_f)\right]\right) \in \mathbb{R}^h$$

where $\overline{\text{LM}}$ denotes applying a language model such as RoBERTa (Liu et al., 2019) and averaging of the token embeddings, and Lin denotes a linear layer. $\boldsymbol{e}_{\text{info}_f}$ represents information in a high-dimensional vector space. Intuitively, every information can be thought of as a node in a graph. The list of information embeddings are the input to a transformer encoder (Vaswani et al., 2017). The attention mechanism allows the agent to decide for every information embedding $\boldsymbol{e}_{\text{info}_f}$ on which other embeddings $\boldsymbol{e}_{\text{info}_g}$ it can put its attention. With a growing number of domains that the system can talk about, the number of information embeddings will increase, making it more difficult to handle the growing state space. However, we observe that only information that is related to active domains is important at the current point in time. Therefore, we prohibit the information embeddings from attending to information that is related to inactive domains in order to avoid the issue of growing state spaces. While the actual state space may be extremely large due to hundreds of domains, the effective state space remains small, making it possible to handle a very large number of domains. Our proposed state encoder is depicted in Figure 2(c).

In this way, the state representation meets the following demands: 1) new concepts can be understood and incorporated seamlessly into the state without a growth in network parameters, as long as they are descriptive; 2) the description embeddings from a language model allow forward transfer by exploiting similarities and common structure among tasks; 3) the value v_f allows numerical information such as confidence scores or other measures of model uncertainty to be included; 4) the state space will not be unreasonably large as information for inactive domains is masked.

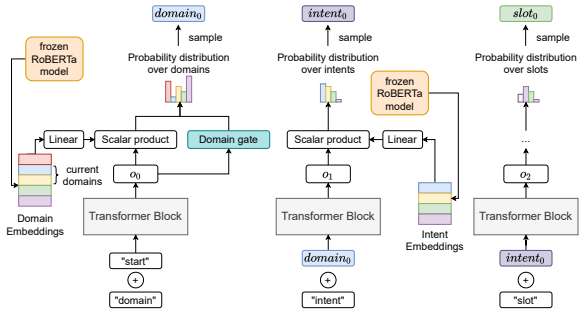

Figure 3: Proposed action prediction in DDPT using a transformer decoder. In every decoding step, a token embedding for domain, intent or slot informs the model what needs to be predicted and the previous output is fed into the decoder. In case of domain prediction, we propose a domain gate that decides whether to choose a domain that the user currently talks about.

4.2 Action Prediction

Similar to existing work (Shu et al., 2019; Zhang et al., 2020a) we separately predict domains, intents and slots for action prediction. We define a domain set $\mathcal{D}$, intent set $\mathcal{I}$ and slot set $\mathcal{S}$ as follows. The domain set $\mathcal{D}$ consists of all domains the model has seen so far plus an additional *stop* domain. The intent set $\mathcal{I}$ and slot set $\mathcal{S}$ consist of all intents and slots we can use for actions, respectively. Every domain, intent and slot has an embedding vector, which we obtain by feeding the token of the domain, intent or slot into our pretrained language model. The embedding vectors are then fed into a linear layer that produces vectors of size $\mathbb{R}^h$. We thus obtain domain, intent and slot embeddings $\boldsymbol{b}^d \; \forall d \in \mathcal{D}$, $\boldsymbol{b}^i \; \forall i \in \mathcal{I}$, and $\boldsymbol{b}^s \; \forall s \in \mathcal{S}$.

The policy first chooses a domain. Then, based on the domain, it picks an intent from the list of intents that are possible for that domain. Lastly, it picks an adequate slot from the set of possible slots for that domain and intent. This process repeats until the policy selects

the *stop* domain. This will lead to a sequence $(domain_m, intent_m, slot_m)_{m=0}^{n}$. We leverage a transformer decoder (Vaswani et al., 2017), the aforementioned embeddings for domains, intents and slots and similarity matching to produce the sequence. In every decoding step t the input to the transformer is $\boldsymbol{b}_{t-1} + \boldsymbol{l}_t$, where $\boldsymbol{b}_{t-1}$ is the embedding of the previous prediction and $\boldsymbol{l}_t$ is a token embedding for token *domain*, *intent* or *slot* that indicates what needs to be predicted in turn t. $\boldsymbol{b}_{-1}$ is an embedding of a *start* token.

If we need to predict a domain in step t, we calculate the scalar product between the decoder output vector $\boldsymbol{o}_t$ and the different domain embeddings $\boldsymbol{b}^d$ and apply the softmax function to obtain a probability distribution $\text{softmax}[\boldsymbol{o}_t \odot \boldsymbol{b}^d, d \in \mathcal{D}]$ over domains from which we can sample. Intent and slot prediction is analogous. In order to guarantee exploration during training and variability during evaluation, we sample from the distributions. While it is important to explore domains during training, during evaluation the domain to choose should be clear. We hence take the domain with the highest probability during evaluation.

As in the state representation, the embeddings using a pretrained language model allow understanding of new concepts (such as a new intent) immediately, which facilitates zero-shot performance. We do not fine-tune any embedding that is produced by the language model.

4.3 Domain Gate

If the policy is exposed to a new unseen domain, the most important point to obtain any zero-shot performance is that the policy predicts the correct domain to talk about. If we only use similarity matching of domain embeddings, the policy will likely predict domains it already knows. In dialogue state tracking we often observe that similarity matching approaches predict values they already know when faced with new unseen values, which leads to poor zero-shot generalisation (Rastogi et al., 2018). To circumvent that, we propose the domain gate. Let $\mathcal{D}_{\text{curr}}$ be the set of domains that the user talks about in the current turn. In every decoding step t where a domain needs to be predicted, the domain gate obtains $\boldsymbol{o}_t$ as input and predicts the probability p_{curr} of using a domain from $\mathcal{D}_{\text{curr}}$. When the policy needs to predict a domain in step t, it now uses the probability distribution given by $p_{\text{curr}} \cdot \text{softmax}[\boldsymbol{o}_t \odot \boldsymbol{b}_d, d \in \mathcal{D}_{\text{curr}}] + (1 - p_{\text{curr}}) \cdot \text{softmax}[\boldsymbol{o}_t \odot \boldsymbol{b}_d, d \notin \mathcal{D}_{\text{curr}}]$.

In this process, the policy does not have to predict the new domain immediately but can abstractly first decide whether it wants to use a domain that the user talks about at the moment. The decoding process is depicted in Figure 3.

5 Experimental Setup

5.1 Metrics

We follow the setup recently proposed by Powers et al. (2021), which assumes that our N tasks/domains $z_1, ..., z_N$ are represented sequentially and each task z_i is assigned a budget k_{z_i}. We can cycle through the tasks M times, leading to a sequence of tasks $x_1, ..., x_{N \cdot M}$. The cycling over tasks defines a more realistic setting than only seeing a task once in the agent's lifetime, in particular in dialogue systems where new domains are introduced but rarely removed.

Continual evaluation: We evaluate performance on all tasks periodically during training. We show the performance for every domain separately to have an in-depth evaluation and the average performance over domains for an overall trend whether the approaches continually improve.

Forgetting: We follow the definition proposed by Chaudhry et al. (2018) and Powers et al. (2021). Let $m_{i,k}$ be a metric achieved on task z_i after training on task x_k, such as the average return or the average dialogue success. For seeds s, tasks z_i and x_j, where $i < j$, we define

$$\mathcal{F}_{i,j} = \frac{1}{s} \sum_s \max_{k \in [0, j-1]} \{m_{i,k} - m_{i,j}\}. \quad (1)$$

$\mathcal{F}_{i,j}$ compares the maximum performance achieved on task z_i before training on task x_j to the performance for z_i after training on task x_j. If $\mathcal{F}_{i,j}$ is positive, the agent has become worse at past task z_i after training on task x_j, indicating forgetting. When $\mathcal{F}_{i,j}$ is negative, the agent has become better at task z_i, indicating backward transfer. We define $\mathcal{F}_i$ as the average over the $\mathcal{F}_{i,j}$ and $\mathcal{F}$ as the average over $\mathcal{F}_i$.

(Zero-Shot) Forward transfer: For seeds s, tasks z_i and z_j, where $j < i$, we define

$$\mathcal{Z}_{i,j} = \frac{1}{s} \sum_s m_{i,j}. \quad (2)$$

We do not substract initial performance as in Powers et al. (2021) as we are interested in the absolute

performance telling us how well we do on task z_i after training on a task z_j. We define $\mathcal{Z}_i$ as the average over the $\mathcal{Z}_{i,j}$ and $\mathcal{Z}$ as the average over $\mathcal{Z}_i$.

5.2 Baselines

We implemented two baselines in order to compare against our proposed DDPT architecture. We do not include a baseline based on expert-defined domain-independent features (Wang et al., 2015) as this requires a significant amount of hand-coding and suffers from scalabilility issues.

5.2.1 Baseline State Representations

We will abbreviate the following baselines with **Bin** and **Sem** that indicate their characteristic way of state representation.

Bin: The first baseline uses a flattened dialogue state for the state representation with *binary* values for every information which is the most common way (Takanobu et al., 2019; Zhu et al., 2020; Weisz et al., 2018). If a new domain d appears, the input vector must be enlarged in order to incorporate the information from d and new network parameters need to be initialised. The state encoding can be seen in Figure 2(a). This baseline serves as a representative of methods where new domains necessitate additional parameters.

Sem: The second baseline implements the idea from Xu et al. (2020), which uses trainable embeddings for domains, intents, slots and values that can capture *semantic* meaning and allow cross-domain transfer. Using trainable embeddings, one representation is calculated for every feature in every feature category (such as user-act, user goal, etc.) in every domain. The feature representations in a category are then averaged over domains to obtain a final representation. More information can be found in Appendix A.4. This baseline serves as a representative of methods where feature representations remain fixed.

5.2.2 Action Prediction for Baselines

Unlike DDPT, which uses a transformer for action prediction, the baselines Bin and Sem use an RNN model for action prediction (Shu et al., 2019; Zhang et al., 2020a). This model uses the decoding process explained in Section 4.2 with the exception that the baselines use trainable embeddings for domain, intent and slot (randomly initialised) instead of using embeddings from a pretrained language model as DDPT does. Moreover, they do not use the proposed domain gate.

5.3 Setup

We use ConvLab-2 (Zhu et al., 2020) as the backbone of our implementation. We take five different tasks from the MultiWOZ dataset (Budzianowski et al., 2018) which are hotel, restaurant, train, taxi and attraction. Hotel, restaurant and train are more difficult compared to attraction and taxi as they require the agent to do bookings in addition to providing information about requested slots. We exclude police and hospital from the task list as they are trivial. We use the rule-based dialogue state tracker and the rule-based user simulator provided in ConvLab-2 (Zhu et al., 2020) to conduct our experiments. Typically, the reward provided is -1 in every turn to encourage efficiency, and a reward of 80 or -40 for dialogue success or failure. A dialogue is successful if the system provided the requested information to the user and booked the correct entities (if possible). We stick to the above reward formulation with one exception: Instead of the turn level reward of -1, we propose to use information overload (Roetzel, 2019). The reason is that dialogue policies tend to over-generate actions, especially if they are trained from scratch. While the user simulator ignores the unnecessary actions, real humans do not. We define information overload for an action $(domain_m, intent_m, slot_m)_{m=1}^{n}$ as $r_{io} = -\rho \cdot n$, where $\rho \in \mathbb{N}$ defines the degree of the penalty. Information overload generalizes the reward of -1 in single action scenarios. We use $\rho = 3$ in the experiments.

We train each of the three architectures using CLEAR (Rolnick et al., 2018). We set the replay buffer capacity to 5000 dialogues and use reservoir sampling (Isele and Cosgun, 2018) when the buffer is full. We assign a budget of 2000 dialogues to restaurant, hotel and train and 1000 to attraction and taxi and cycle through these tasks two times, resulting in 16000 training dialogues in total. Since task ordering is still an open area of research (Jiang et al., 2020), we test three different permutations so that our results do not depend on a specific order. The domain orders we use are 1) *easy-to-hard*: attraction, taxi, train, restaurant, hotel 2) *hard-to-easy*: hotel, restaurant, train, taxi, attraction and 3) *mixed*: restaurant, attraction, hotel, taxi, train.

6 Results

6.1 Continual Evaluation

We show performance in terms of average return for all three task orders in Figure 4(a)-(c). The plots

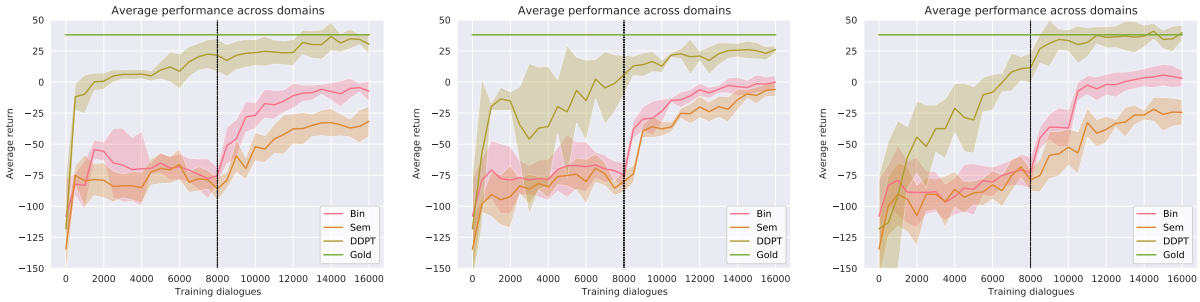

(a) easy-to-hard (b) hard-to-easy (c) mixed

Figure 4: Training Bin, Sem and DDPT (ours) using CLEAR on three different domain orders, each with 5 different seeds, by interacting with the rule-based user simulator. Each model is evaluated every 500 training dialogues on 100 dialogues per domain. The plots show the average return, where performance is averaged over domains. The vertical line at 8000 dialogues indicates the start of cycle 2. The shaded area represents standard deviation. Gold serves as an upper bound.

show the performance averaged over domains. We refer to Appendix A.8 for in-depth evaluations for each individual domain. The horizontal line **Gold** denotes an upper limit for the models that was obtained by training a Bin model separately on each domain until convergence. We can observe that DDPT outperforms the baselines regardless of task order, almost reaching the upper bound. We will see in Section 6.2 that the baselines suffer more from forgetting compared to DDPT, such that training on a new domain reduces performance on previous domains. We suspect that this contributes to the lower final performance of the baselines. Moreover, we can observe that the final performance of DDPT barely depends on a specific task order. Nevertheless, we can see that training starts off faster in easy-to-hard order, which shows that behaviour learned for attraction transfers well to other domains. Lastly, the second training cycle is necessary for increasing performance of the models. We note that even though it looks like the baselines don't learn at all in the first round, they do learn but tend to forget previous knowledge. This can be observed in detail in Appendix A.8.

6.2 Forward Transfer and Forgetting

We calculated forward and forgetting metrics as explained in Section 5.1. Table 1 shows success rates instead of average return because success is easier to interpret. We can see for every model the summary statistics $\mathcal{F}$ and $\mathcal{Z}$ measuring average forgetting and forward transfer, respectively. To obtain lower bounds we added forward and forgetting of a random model that is initialised randomly again every time it observes a domain.

Table 1 reveals that DDPT outperforms the baselines significantly in terms of absolute numbers and also relative numbers compared to the random performance. As expected, Bin shows almost no zero-shot performance improvement compared to the random model, whereas Sem obtains slight improvement. DDPT shows large forward transfer capabilities and strong robustness against forgetting. We attribute this to the frozen description and action embeddings stemming from the language model and the domain gate. The language model allows us to interpret new information and actions immediately, enabling the model to draw connections between learned tasks and new ones. At the same time, frozen embeddings are robust to forgetting. The domain gate allows the model to choose the domain more abstractly without initial exploration due to the decision between current or non-current domains, which facilitates zero-shot performance. Moreover, the baselines need to make a hard decision between domains (balancing between choosing a domain we learn about at the moment and old domains), whereas the domain decision for DDPT is abstracted through the domain gate, leading to robustness against forgetting. Both baselines perform substantially better than the lower bound, suggesting that these are non-trivial baselines.

6.3 Benefits of Domain Gate

In order to analyse the contribution of the domain gate to the forward capabilities of DDPT, we train a DDPT model without domain gate on the easy-to-hard order, where DDPT showed the highest forward transfer. From Table 2 we can observe that performance drops significantly for all domains if

	Easy-to-hard		Hard-to-easy		Mixed order		Random	
Model	$\mathcal{F}\downarrow$	$\mathcal{Z}\uparrow$	$\mathcal{F}\downarrow$	$\mathcal{Z}\uparrow$	$\mathcal{F}\downarrow$	$\mathcal{Z}\uparrow$	$\mathcal{F}\downarrow$	$\mathcal{Z}\uparrow$
Bin	0.14	0.39	0.14	0.45	0.14	0.38	0.43	0.39
Sem	0.20	0.39	0.17	0.37	0.18	0.29	0.43	0.26
DDPT	**0.01**	**0.73**	**0.02**	**0.68**	**0.03**	**0.57**	0.43	0.34

Table 1: Showing summary statistics in terms of success for forgetting $\mathcal{F}$ (ranging between -1 and 1, the lower the better) and forward transfer $\mathcal{Z}$ (ranging between 0 and 1, the higher the better).

	Taxi	Train	Restaurant	Hotel	$\mathcal{Z}\uparrow$
DDPT	0.90	0.76	0.73	0.53	0.73
DDPT w/o domain gate	0.68	0.19	0.57	0.28	0.43

Table 2: Forward transfer metrics $\mathcal{Z}_i$ in terms of success for different domains i trained on easy-to-hard order with and without domain gate.

the domain gate is not employed, which shows the importance of this mechanism.

6.4 Results on Transformer-based Simulator

In order to strengthen our results and show that they do not depend on the simulator used, we conducted an additional experiment using the transformer-based user simulator TUS (Lin et al., 2021). We only show results for the mixed order, having in mind that results have not been dependent on the domain order used. Figure 5 shows that DDPT again outperforms the baseline.

6.5 Results on Human Trial

We further validate the results by conducting a human trial. We compare Bin, Gold and DDPT, where Bin and DDPT were trained on the mixed domain order. We hire humans through Amazon Mechanical Turk and let them directly interact with our systems, thereby collecting 258, 278 and 296 dialogues for Bin, Gold and DDPT, respectively. After a user finished the dialogue we asked 1) whether the dialogue was successful (Success), 2) whether the system often mentioned something the user did not ask for such as a wrong domain (UnnecInfo) 3), whether the system gave too much information (TooMuchInfo) and 4) about the general performance (Performance). Table 3 shows that the upper bound Gold and DDPT perform equally well ($p > 0.05$) in every metric whereas Bin performs statistically significant worse. The low performance of Bin can be partially attributed to frequently choosing a wrong domain that humans are more sensitive to than a user simulator.

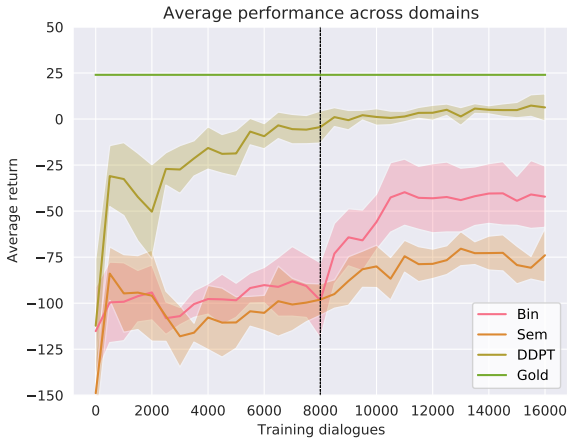

Figure 5: Training Bin, Sem and DDPT (ours) on the mixed domain order with the transformer based user simulator TUS.

	Success ↑	UnnecInfo ↓	TooMuchInfo ↓	Performance ↑
Bin	0.45	3.98	3.15	2.45
Gold	0.81	2.79	2.71	3.65
DDPT	0.77	2.75	2.56	3.67

Table 3: Human trial results where Bin, Gold and DDPT interacted with real users. There is no statistically significant difference ($p > 0.05$) between DDPT and Gold, while Bin is statistically significantly worse ($p < 0.05$) than Gold and DDPT.

Example dialogues are given in Appendix A.6.

7 Conclusion

In this work we provided an algorithm, baseline models and evaluation metrics to enable continual RL for dialogue policy optimisation. Moreover, we proposed a dynamic dialogue policy model called DDPT that builds on information descriptions, a pretrained language model and the transformer encoder-decoder architecture. It integrates new information seamlessly as long as it is descriptive, and obtains significant zero-shot performance on unseen domains while being robust to forgetting. The strengths of DDPT were validated in simulation with two simulators as well as humans. This opens the door for building evolving dialogue systems, that continually expand their knowledge and improve their behaviour throughout their lifetime.

Acknowledgements

This work is a part of DYMO project which has received funding from the European Research Council (ERC) provided under the Horizon 2020 research and innovation programme (Grant agree-

ment No. STG2018 804636). N. Lubis, C. van Niekerk, M. Heck and S. Feng are funded by an Alexander von Humboldt Sofja Kovalevskaja Award endowed by the German Federal Ministry of Education and Research. Computing resources were provided by Google Cloud and HHU ZIM.

References

Magdalena Biesialska, Katarzyna Biesialska, and Marta R. Costa-jussà. 2020. Continual lifelong learning in natural language processing: A survey. *CoRR*, abs/2012.09823.

Paweł Budzianowski, Tsung-Hsien Wen, Bo-Hsiang Tseng, Iñigo Casanueva, Stefan Ultes, Osman Ramadan, and Milica Gašić. 2018. MultiWOZ - a large-scale multi-domain Wizard-of-Oz dataset for task-oriented dialogue modelling. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 5016–5026, Brussels, Belgium. Association for Computational Linguistics.

Massimo Caccia, Pau Rodríguez, Oleksiy Ostapenko, Fabrice Normandin, Min Lin, Lucas Caccia, Issam H. Laradji, Irina Rish, Alexandre Lacoste, David Vázquez, and Laurent Charlin. 2020. Online fast adaptation and knowledge accumulation: a new approach to continual learning. *CoRR*, abs/2003.05856.

Iñigo Casanueva, Paweł Budzianowski, Pei-Hao Su, Stefan Ultes, Lina M. Rojas-Barahona, Bo-Hsiang Tseng, and Milica Gašić. 2018. Feudal reinforcement learning for dialogue management in large domains. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 2 (Short Papers)*, pages 714–719, New Orleans, Louisiana. Association for Computational Linguistics.

Yash Chandak, Georgios Theocharous, Shiv Shankar, Martha White, Sridhar Mahadevan, and Philip S. Thomas. 2020. Optimizing for the future in non-stationary mdps. *CoRR*, abs/2005.08158.

Arslan Chaudhry, Puneet Kumar Dokania, Thalaiyasingam Ajanthan, and Philip H. S. Torr. 2018. Riemannian walk for incremental learning: Understanding forgetting and intransigence. In *Computer Vision - ECCV 2018 - 15th European Conference, Munich, Germany, September 8-14, 2018, Proceedings, Part XI*, volume 11215 of *Lecture Notes in Computer Science*, pages 556–572. Springer.

Lu Chen, Bowen Tan, Sishan Long, and Kai Yu. 2018. Structured dialogue policy with graph neural networks. In *Proceedings of the 27th International Conference on Computational Linguistics*, pages 1257–1268, Santa Fe, New Mexico, USA. Association for Computational Linguistics.

Zhi Chen, Lu Chen, Xiaoyuan Liu, and Kai Yu. 2020. Distributed structured actor-critic reinforcement learning for universal dialogue management. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 28:2400–2411.

Lasse Espeholt, Hubert Soyer, Rémi Munos, Karen Simonyan, Volodymyr Mnih, Tom Ward, Yotam Doron, Vlad Firoiu, Tim Harley, Iain Dunning, Shane Legg, and Koray Kavukcuoglu. 2018. IMPALA: scalable distributed Deep-RL with importance weighted actor-learner architectures. In *Proceedings of the 35th International Conference on Machine Learning, ICML 2018, Stockholmsmässan, Stockholm, Sweden, July 10-15, 2018*, volume 80 of *Proceedings of Machine Learning Research*, pages 1406–1415. PMLR.

Milica Gašić, Nikola Mrkšić, Pei-Hao Su, David Vandyke, Tsung-Hsien Wen, and Steve Young. 2015. Policy committee for adaptation in multi-domain spoken dialogue systems. In *2015 IEEE Workshop on Automatic Speech Recognition and Understanding (ASRU)*, pages 806–812.

Binzong Geng, Fajie Yuan, Qiancheng Xu, Ying Shen, Ruifeng Xu, and Min Yang. 2021. Continual learning for task-oriented dialogue system with iterative network pruning, expanding and masking. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 2: Short Papers)*, pages 517–523, Online. Association for Computational Linguistics.

Raia Hadsell, Dushyant Rao, Andrei Rusu, and Razvan Pascanu. 2020. Embracing change: Continual learning in deep neural networks. *Trends in Cognitive Sciences*, 24:1028–1040.

David Isele and Akansel Cosgun. 2018. Selective experience replay for lifelong learning. In *Proceedings of the Thirty-Second AAAI Conference on Artificial Intelligence and Thirtieth Innovative Applications of Artificial Intelligence Conference and Eighth AAAI Symposium on Educational Advances in Artificial Intelligence*, AAAI'18/IAAI'18/EAAI'18. AAAI Press.

Minqi Jiang, Edward Grefenstette, and Tim Rocktäschel. 2020. Prioritized level replay. *CoRR*, abs/2010.03934.

Khimya Khetarpal, Matthew Riemer, Irina Rish, and Doina Precup. 2020a. Towards continual reinforcement learning: A review and perspectives. *CoRR*, abs/2012.13490.

Khimya Khetarpal, Matthew Riemer, Irina Rish, and Doina Precup. 2020b. Towards continual reinforcement learning: A review and perspectives. *CoRR*, abs/2012.13490.

Diederik P. Kingma and Jimmy Ba. 2015. Adam: A method for stochastic optimization. In *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*.

Matthias De Lange, Rahaf Aljundi, Marc Masana, Sarah Parisot, Xu Jia, Ales Leonardis, Gregory G. Slabaugh, and Tinne Tuytelaars. 2019. Continual learning: A comparative study on how to defy forgetting in classification tasks. *CoRR*, abs/1909.08383.

Erwan Lecarpentier and Emmanuel Rachelson. 2019. Non-stationary markov decision processes, a worst-case approach using model-based reinforcement learning. In *Advances in Neural Information Processing Systems*, volume 32. Curran Associates, Inc.

Chia-Hsuan Lee, Hao Cheng, and Mari Ostendorf. 2021. Dialogue state tracking with a language model using schema-driven prompting. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 4937–4949, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Kyusong Lee, Tiancheng Zhao, Alan W. Black, and Maxine Eskenazi. 2018. DialCrowd: A toolkit for easy dialog system assessment. In *Proceedings of the 19th Annual SIGdial Meeting on Discourse and Dialogue*, pages 245–248, Melbourne, Australia. Association for Computational Linguistics.

Sungjin Lee. 2017. Toward continual learning for conversational agents. *CoRR*, abs/1712.09943.

Esther Levin and Roberto Pieraccini. 1997. A stochastic model of computer-human interaction for learning dialogue strategies. In *EUROSPEECH 97*, pages 1883–1886.

Hsien-chin Lin, Nurul Lubis, Songbo Hu, Carel van Niekerk, Christian Geishauser, Michael Heck, Shutong Feng, and Milica Gašić. 2021. Domain-independent user simulation with transformers for task-oriented dialogue systems. In *Proceedings of the 22nd Annual Meeting of the Special Interest Group on Discourse and Dialogue*, pages 445–456, Singapore and Online. Association for Computational Linguistics.

Zachary Lipton, Xiujun Li, Jianfeng Gao, Lihong Li, Faisal Ahmed, and Li Deng. 2018. BBQ-networks: Efficient exploration in deep reinforcement learning for task-oriented dialogue systems. *Proceedings of the AAAI Conference on Artificial Intelligence*, 32(1).

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. RoBERTa: A robustly optimized BERT pretraining approach. *CoRR*, abs/1907.11692.

Andrea Madotto, Zhaojiang Lin, Zhenpeng Zhou, Seungwhan Moon, Paul Crook, Bing Liu, Zhou Yu, Eunjoon Cho, Pascale Fung, and Zhiguang Wang. 2021. Continual learning in task-oriented dialogue systems. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 7452–7467, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Brielen Madureira and David Schlangen. 2020. An overview of natural language state representation for reinforcement learning. *ArXiv*, abs/2007.09774.

Fei Mi, Liangwei Chen, Mengjie Zhao, Minlie Huang, and Boi Faltings. 2020. Continual learning for natural language generation in task-oriented dialog systems. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 3461–3474, Online. Association for Computational Linguistics.

Fabrice Normandin, Florian Golemo, Oleksiy Ostapenko, Pau Rodríguez, Matthew D. Riemer, Julio Hurtado, Khimya Khetarpal, Dominic Zhao, Ryan Lindeborg, Timothée Lesort, Laurent Charlin, Irina Rish, and Massimo Caccia. 2021. Sequoia: A software framework to unify continual learning research. *CoRR*, abs/2108.01005.

German Parisi, Ronald Kemker, Jose Part, Christopher Kanan, and Stefan Wermter. 2019. Continual lifelong learning with neural networks: A review. *Neural Networks*, 113:54–71.

Baolin Peng, Chenguang Zhu, Chunyuan Li, Xiujun Li, Jinchao Li, Michael Zeng, and Jianfeng Gao. 2020. Few-shot natural language generation for task-oriented dialog. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 172–182, Online. Association for Computational Linguistics.

Sam Powers, Eliot Xing, Eric Kolve, Roozbeh Mottaghi, and Abhinav Gupta. 2021. CORA: Benchmarks, baselines, and metrics as a platform for continual reinforcement learning agents.

Abhinav Rastogi, Raghav Gupta, and Dilek Hakkani-Tur. 2018. Multi-task learning for joint language understanding and dialogue state tracking. *arXiv preprint arXiv:1811.05408*.

Abhinav Rastogi, Xiaoxue Zang, Srinivas Sunkara, Raghav Gupta, and Pranav Khaitan. 2020. Towards scalable multi-domain conversational agents: The schema-guided dialogue dataset. *Proceedings of the AAAI Conference on Artificial Intelligence*, 34(05):8689–8696.

Peter Gordon Roetzel. 2019. Information overload in the information age: a review of the literature from business administration, business psychology, and related disciplines with a bibliometric approach and framework development. *Business Research*, 12(2):479–522.

David Rolnick, Arun Ahuja, Jonathan Schwarz, Timothy P. Lillicrap, and Greg Wayne. 2018. Experience replay for continual learning. *CoRR*, abs/1811.11682.

Nicholas Roy, Joelle Pineau, and Sebastian Thrun. 2000. Spoken dialogue management using probabilistic reasoning. In *Proceedings of the 38th Annual Meeting on Association for Computational Linguistics*, ACL '00, page 93–100, USA. Association for Computational Linguistics.

Lei Shu, Hu Xu, Bing Liu, and Piero Molino. 2019. Modeling multi-action policy for task-oriented dialogues. *CoRR*, abs/1908.11546.

Pei-Hao Su, Milica Gašić, Nikola Mrkšić, Lina Maria Rojas-Barahona, Stefan Ultes, David Vandyke, Tsung-Hsien Wen, and Steve J. Young. 2016. Continuously learning neural dialogue management. *CoRR*, abs/1606.02689.

Ryuichi Takanobu, Hanlin Zhu, and Minlie Huang. 2019. Guided dialog policy learning: Reward estimation for multi-domain task-oriented dialog. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 100–110, Hong Kong, China. Association for Computational Linguistics.

Carel van Niekerk, Andrey Malinin, Christian Geishauser, Michael Heck, Hsien-chin Lin, Nurul Lubis, Shutong Feng, and Milica Gašić. 2021. Uncertainty measures in neural belief tracking and the effects on dialogue policy performance. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 7901–7914, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc.

Vladimir Vlasov, Johannes E. M. Mosig, and Alan Nichol. 2019. Dialogue transformers. *CoRR*, abs/1910.00486.

Zhuoran Wang, Tsung-Hsien Wen, Pei-Hao Su, and Yannis Stylianou. 2015. Learning domain-independent dialogue policies via ontology parameterisation. In *Proceedings of the 16th Annual Meeting of the Special Interest Group on Discourse and Dialogue*, pages 412–416, Prague, Czech Republic. Association for Computational Linguistics.

Gellért Weisz, Paweł Budzianowski, Pei-Hao Su, and Milica Gašić. 2018. Sample efficient deep reinforcement learning for dialogue systems with large action spaces. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 26(11):2083–2097.

Jason D. Williams and Steve Young. 2007. Partially observable Markov decision processes for spoken dialog systems. *Comput. Speech Lang.*, 21(2):393–422.

Chien-Sheng Wu, Andrea Madotto, Ehsan Hosseini-Asl, Caiming Xiong, Richard Socher, and Pascale Fung. 2019. Transferable multi-domain state generator for task-oriented dialogue systems. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 808–819, Florence, Italy. Association for Computational Linguistics.

Yumo Xu, Chenguang Zhu, Baolin Peng, and Michael Zeng. 2020. Meta dialogue policy learning. *CoRR*, abs/2006.02588.

Steve Young, Jost Schatzmann, Karl Weilhammer, and Hui Ye. 2007. The hidden information state approach to dialog management. In *2007 IEEE International Conference on Acoustics, Speech and Signal Processing - ICASSP '07*, volume 4, pages IV–149–IV–152.

Yichi Zhang, Zhijian Ou, and Zhou Yu. 2020a. Task-oriented dialog systems that consider multiple appropriate responses under the same context. In *The Thirty-Fourth AAAI Conference on Artificial Intelligence, AAAI 2020, The Thirty-Second Innovative Applications of Artificial Intelligence Conference, IAAI 2020, The Tenth AAAI Symposium on Educational Advances in Artificial Intelligence, EAAI 2020, New York, NY, USA, February 7-12, 2020*, pages 9604–9611. AAAI Press.

Zheng Zhang, Ryuichi Takanobu, Minlie Huang, and Xiaoyan Zhu. 2020b. Recent advances and challenges in task-oriented dialog system. *CoRR*, abs/2003.07490.

Qi Zhu, Zheng Zhang, Yan Fang, Xiang Li, Ryuichi Takanobu, Jinchao Li, Baolin Peng, Jianfeng Gao, Xiaoyan Zhu, and Minlie Huang. 2020. ConvLab-2: An open-source toolkit for building, evaluating, and diagnosing dialogue systems. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics: System Demonstrations*, pages 142–149, Online. Association for Computational Linguistics.

A Appendix

A.1 Background on CLEAR

A.1.1 VTRACE Algorithm

VTRACE (Espeholt et al., 2018) is an off-policy actor critic algorithm. As such, it optimizes both a policy π_θ and a corresponding critic V_ψ that estimates the state-value function V of π_θ. Actor and critic are both updated using experience from a replay buffer $\mathcal{B}$.

Given a trajectory $\tau = (s_t, a_t, r_t)_{t=k}^{t=k+n}$ generated by a behaviour policy μ, the n-steps vtrace-target for $V(s_k)$ is defined as

$$v_k = V(s_k) + \sum_{t=k}^{k+n-1} \gamma^{t-k} (\prod_{i=k}^{t-1} c_i) \delta_t V,$$

where $\delta_t V = \rho_t(r_t + \gamma V(s_{t+1}) - V(s_t))$ is a temporal difference for V, and $\rho_t = \min(\overline{\rho}, \frac{\pi(a_t|s_t)}{\mu(a_t|s_t)})$ and $c_i = \min(\overline{c}, \frac{\pi(a_i|s_i)}{\mu(a_i|s_i)})$ are truncated importance sampling weights. The scalars $\overline{\rho}$ and $\overline{c}$ are hyperparameters where it is assumed that $\overline{\rho} \geq \overline{c}$.

The critic function is then optimized to minimize the gap between its prediction and the vtrace-target:

$$\mathcal{L}_{critic}(\psi) = \mathbb{E}_{\tau\sim\mathcal{B}}[(v_k - V_\psi(s_k))^2] \quad (3)$$

The actor is optimized using the following off-policy policy gradient:

$$\mathbb{E}_{\tau\sim\mathcal{B}}[\frac{\pi(a_k|s_k)}{\mu(a_k|s_k)} A_k \nabla_\theta \log \pi_\theta(a_k|s_k)] \quad (4)$$

where $A_k = (r_k + \gamma v_{k+1} - V_\psi(s_k))$ is an estimate of the advantage function. To prevent premature convergence, they add an entropy loss $\mathcal{L}_{entropy}(\theta)$ during optimization.

A.1.2 CLEAR

CLEAR is a continual learning algorithm that adapts VTRACE to fulfill the continual learning requirements. The goal is to obtain fast adaptation capabilities as well as preventing catastrophic forgetting. Fast adaptation is tackled by using the most recent trajectories instead of randomly sampling from the buffer $\mathcal{B}$ in Equations 3 and 4.

In order to prevent catastrophic forgetting, they sample non-recent experience from the replay buffer and update policy and critic using Equations 3 and 4. To further regularize these non-recent updates, they introduce regularization losses $\mathcal{L}_{\pi-reg}$ and $\mathcal{L}_{v-reg}$. $\mathcal{L}_{v-reg}$ forces the critic prediction to be close to the historic prediction through a mean-squared error loss. $\mathcal{L}_{\pi-reg}$ regularizes the actor to minimize the KL-divergence between the behaviour policy μ and current policy π_θ:

$$\mathcal{L}_{v-reg}(\psi) = \mathbb{E}_{\tau\sim\mathcal{B}}[(V_\psi(s_k) - V_{replay}(s_k))^2]$$
$$\mathcal{L}_{\pi-reg}(\theta) = \mathbb{E}_{\tau\sim\mathcal{B}}[\sum_a \mu(a_|s_k) \log \frac{\mu(a_|s_k)}{\pi_\theta(a_|s_k)}]$$

An online-offline ratio determines how many recent and non-recent experience is used in an update, thereby trading-off fast adaptation and catastrophic forgetting prevention.

A.2 Training details

For the baselines, the MLP encoder uses a 3-layer MLP with hidden dimension of 128 and RELU as activation function. We use a GRU with 2 layers and input size as well as hidden size of 128 for action decoding. The domain, intent and slot embeddings for action prediction have a size of 64. They are fed through a linear layer that projects it to a vector of size 128 (same size as GRU output) in order to allow computation of the scalar product with the GRU output. The semantic encoding in Sem uses an embedding size of 32 for domain, intent, slot and values. The critic for Bin and Sem has the same architecture as the MLP encoder, with an additional linear layer to project the output to a real valued number.

For the DDPT model, we use an input size and hidden size of 128 in both transformer encoder and decoder. We use two heads for the encoder and decoder, 4 transformer layers for the encoder and 2 for the decoder. The critic for DDPT has the same architecture as the transformer encoder, obtaining the same input as the policy module plus an additional CLS vector (as in RoBERTa). The output of the CLS vector is fed into a linear layer to obtain the critic prediction.

For every model, we use the same training configurations. We use the ADAM optimiser (Kingma and Ba, 2015) with a learning rate of 5e-5 and 1e-4 for policy and critic module, respectively. We sample a batch of 64 episodes for updating the model after every 2 new dialogues. The replay buffer size is set to 5000. For the VTRACE algorithm, the parameters $\bar{\rho}$ and $\bar{c}$ are set to 1.0. For CLEAR we use an online-offline ratio of 0.2, i.e. 20% of the dialogues in a batch are from the most recent dialogues and the remaining 80% from historical dialogues. The regularization losses are weighted by 0.1 and the entropy loss by 0.01.

We used a NVIDIA Tesla T4 provided by the Google Cloud Platform for training the models. The training of one model took 10 to 16 hours depending on the architecture used.

A.3 Masking of illegal actions

To aid the policy in the difficult RL environment, we add a simple masking mechanism that prohibits illegal actions. The action masking includes the following

- If the data base query tells us that entities for a domain are available, the policy is not allowed to say that there are no entities available.
- If there is no entity found with the current constraints, the policy is not allowed to inform on information about entities.
- The *Booking* domain is only usable for hotel and restaurant.

A.4 Baselines

As mentioned in Section 5.2, the second baseline incorporates the idea from Xu et al. (2020), which uses trainable embeddings for domains, intents and slots to allow cross-domain transfer. For every feature category (such as user-act, user goal, etc.) and every domain, it calculates for every feature in that category a representation using trainable domain, intent and slot embeddings. The features in a category are then averaged over domains to obtain a final representation.

For instance, considering the user-act category for a domain d, the user act $(d, i_k, s_k)_{k=0}^{n}$ is first embedded as $\hat{\boldsymbol{s}}_{\text{u-act},d} = \frac{1}{n}\sum_{k=0}^{n}[\boldsymbol{v}_d, \boldsymbol{v}_{i_k}, \boldsymbol{v}_{s_k}]$, where $\boldsymbol{v}_d, \boldsymbol{v}_{i_k}$ and $\boldsymbol{v}_{s_k}$ are trainable embeddings for domain d, intents i_k and slots s_k and afterwards fed through a residual block, leading to $\boldsymbol{s}_{\text{u-act},d} = \hat{\boldsymbol{s}}_{\text{u-act},d} + \text{ReLU}(\boldsymbol{W}_{\text{u-act}}\hat{\boldsymbol{s}}_{\text{u-act},d} + \boldsymbol{b}_{\text{u-act}})$. If there is no user-act for domain d, we use an embedding for *no-user-act* to indicate that. The overall feature representation for the user-act is then given by $\boldsymbol{s}_{\text{u-act}} = \frac{1}{|\mathcal{D}|}\sum_{d\in\mathcal{D}}\boldsymbol{s}_{\text{u-act},d}$.

The representations for different feature categories are then concatenated and fed into a multi-layer perceptron encoder. The state encoding can be seen in Figure 2(b). We abbreviate this baselines as *Sem* as it uses semantic features.

A.5 Descriptions

Our DDPT model uses descriptions for every possible information. This allows us to seamlessly deal with new information we have not seen before yet by leveraging a pretrained language model. The language model provides us token embeddings for the description, which are averaged in order to obtain the description embedding. The descriptions are built as follows.

- For every domain `d` and every slot `s` the user can inform on, the description is given by `user goal <d> <s>`. The corresponding value is 1, if that slot has been mentioned and 0 else.

- For every atomic user act `d i s` that was used in the current turn, the description is given by `user act <d> <i> <s>`. We consider each atomic user act as one information and only provide user acts that were used in the current turn to the model with a corresponding value of 1.

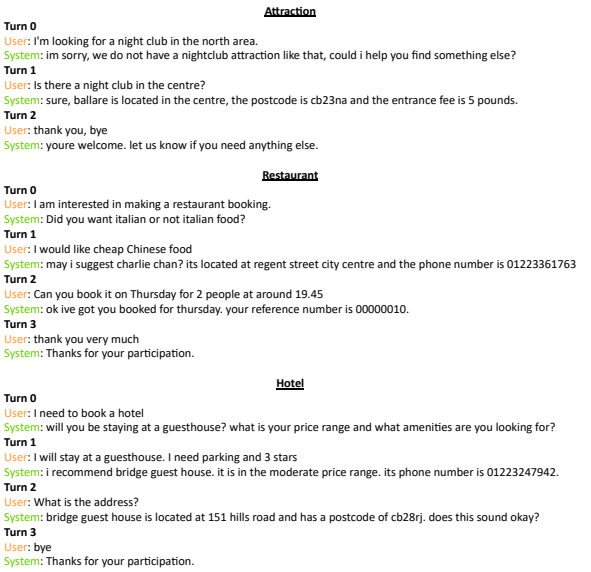

Figure 6: Example dialogues that were collected during the human trial. Users hired through Amazon Mechanical Turk interact with our DDPT model.

- For every atomic system act `d i s` that was used in the previous turn, the description is given by `last system act <d> <i> <s>` with a corresponding value of 1.

- For every domain `d` where a data base query is possible to obtain the number of entities that fulfill the user constraints, the description is given by `data base <d> <number of entities>` with a corresponding value indicating the number of search results.

- For every domain `d` where an entity can be booked, the description is given by `general <d> <booked>` with a binary indicating whether an entity has already been booked.

A.6 Human trial

We conducted a human trial to validate our results in simulation. The website was build using DialCrowd (Lee et al., 2018) and users were hired using Amazon Mechanical Turk. We used SetSUMBT (van Niekerk et al., 2021) as belief tracker and SC-GPT (Peng et al., 2020) as NLG module to accompany the dialogue policies Bin, Gold and DDPT in the dialogue system pipelines. Example dialogues, where DDPT interacted with users hired through Amazon Mechanical Turk, are depicted in Figure 6.

	Easy-to-hard			Hard-to-easy			Mixed order		
Task	Bin	Sem	DDPT	Bin	Sem	DDPT	Bin	Sem	DDPT
Attraction	/	/	/	0.43	0.35	0.83	0.60	0.33	0.79
Taxi	0.51	0.75	0.90	0.51	0.47	0.85	0.35	0.43	0.77
Train	0.21	0.18	0.76	0.23	0.15	0.28	0.17	0.09	0.34
Restaurant	0.47	0.36	0.73	0.62	0.52	0.74	/	/	/
Hotel	0.36	0.26	0.53	/	/	/	0.39	0.28	0.39
Average	0.39	0.39	0.73	0.45	0.37	0.68	0.38	0.29	0.57
Random	0.39	0.26	0.34	0.39	0.26	0.34	0.39	0.26	0.34

Table 4: Forward transfer table showing for every domain i the metric $\mathcal{Z}_i$ in terms of success rate, where numbers range between 0 and 1. The higher the number, the more forward transfer is achieved.

	Easy-to-hard			Hard-to-easy			Mixed order			Random
Task	Bin	Sem	DDPT	Bin	Sem	DDPT	Bin	Sem	DDPT	
Attraction	0.28	0.49	0.03	0.08	0.09	0.02	0.29	0.40	0.0	
Taxi	0.13	0.15	0.01	0.01	0.01	0.02	0.01	0.02	0.0	
Train	0.18	0.20	0.02	0.13	0.14	-0.01	0.03	0.03	0.0	
Restaurant	0.06	0.11	-0.01	0.16	0.19	0.0	0.22	0.26	0.09	
Hotel	0.04	0.07	0.0	0.32	0.41	0.07	0.14	0.19	0.03	
Average	0.14	0.20	0.01	0.14	0.17	0.02	0.14	0.18	0.03	0.43

Table 5: Forgetting table showing for every domain i the metric $\mathcal{F}_i$ in terms of success rate, where numbers range between -1 and 1. Negative numbers indicate backward transfer whereas positive numbers indicate forgetting.

	Easy-to-hard			Hard-to-easy			Mixed order		
Task	Bin	Sem	DDPT	Bin	Sem	DDPT	Bin	Sem	DDPT
Attraction	/	/	/	-88	-124	12	-16	-125	-3
Taxi	-91	-32	23	-65	-117	13	-85	-127	-12
Train	-149	-156	-17	-66	-180	-108	-140	-189	-112
Restaurant	-94	-119	-15	-15	-97	-19	/	/	/
Hotel	-121	-143	-81	/	/	/	-45	-139	-107
Average	-114	-113	-23	-58	-129	-25	-71	-145	-58

Table 6: Forward transfer table showing for every domain i the metric $\mathcal{Z}_i$ in terms of average return. The higher the number, the more forward transfer is achieved.

	Easy-to-hard			Hard-to-easy			Mixed order		
Task	Bin	Sem	DDPT	Bin	Sem	DDPT	Bin	Sem	DDPT
Attraction	99	151	6	34	36	2	93	126	1
Taxi	73	89	4	16	23	4	18	29	1
Train	68	68	1	43	49	-2	10	10	-1
Restaurant	35	38	-1	59	71	2	78	91	26
Hotel	12	21	-1	89	112	18	51	59	7
Average	58	73	2	48	58	5	50	63	7

Table 7: Forgetting table showing for every domain i the metric $\mathcal{F}_i$ in terms of average return. Negative numbers indicate backward transfer whereas positive numbers indicate forgetting.

A.7 Forward Transfer and Forgetting

We provide the forward and forgetting tables in terms of success rate and average return in Tables 4, 5, 6, 7.

A.8 Continual Evaluation

Here, we provide in-depth results for all experiments. Each graph shows the performance of a single domain during training. Moreover, we provide the average performance over domains in terms of success rate in Figure 7 to complement Figure 4.

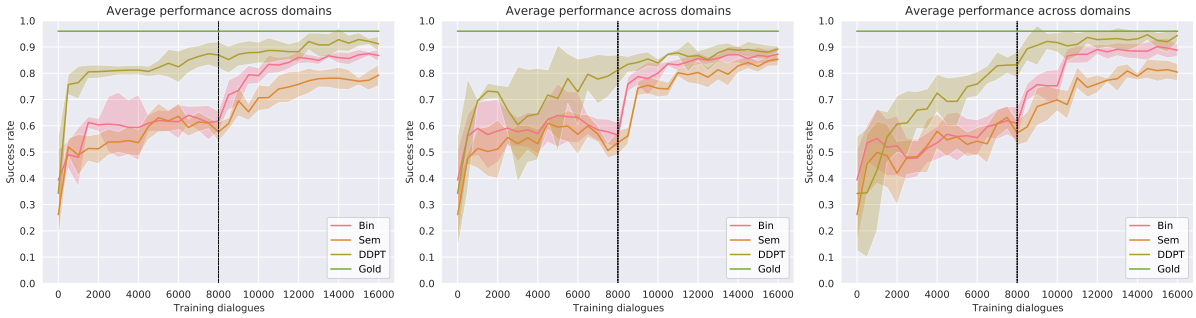

(a) easy-to-hard order (b) hard-to-easy order (c) mixed domain order

Figure 7: Training the three architectures Bin, Sem and DDPT using CLEAR on three different domain orders, each with 5 different seeds. Each model is evaluated every 500 training dialogues on 100 dialogues per domain. The plots show the success rate, where performance is averaged over domains. The vertical line at 8000 dialogues indicates the start of cycle 2.

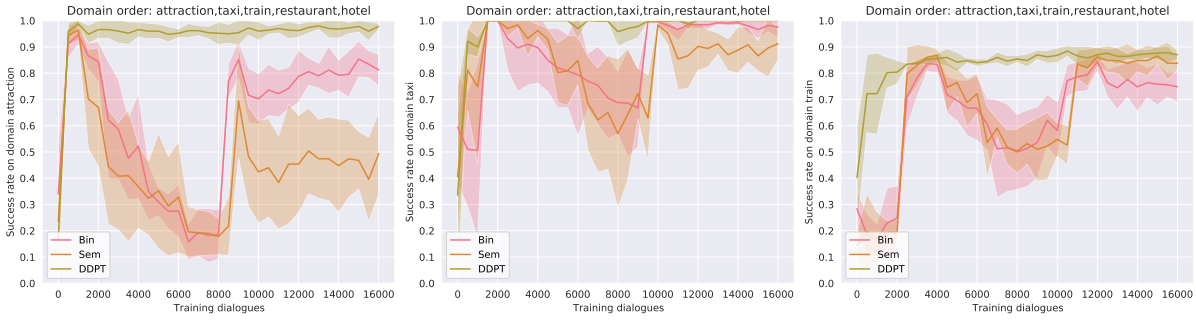

(a) Success rate on attraction domain (b) Success rate on taxi domain (c) Success rate on train domain

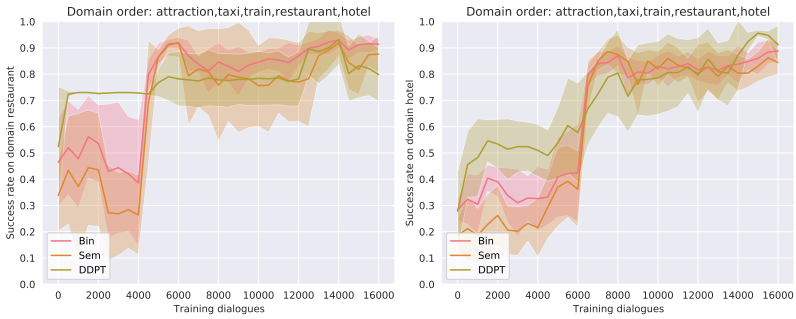

(d) Success rate on restaurant domain (e) Success rate on hotel domain

Figure 8: Success rate for each individual domain, where algorithms are trained in the order easy-to-hard.

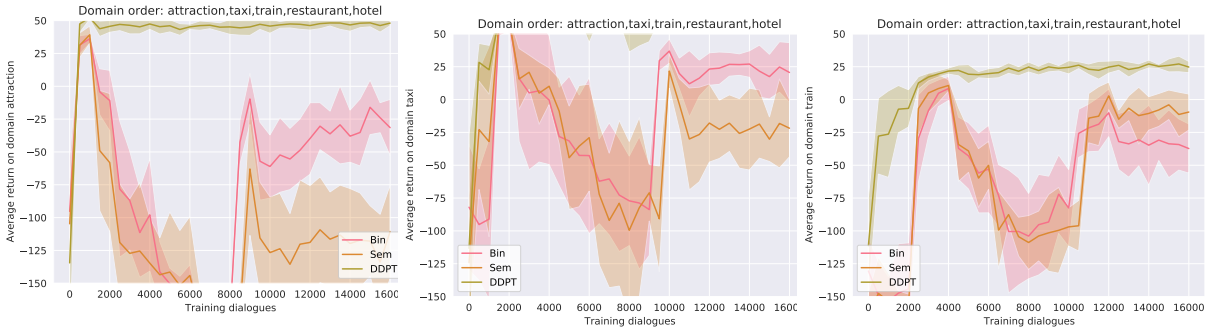

(a) Average return on attraction domain

(b) Average return on taxi domain

(c) Average return on train domain

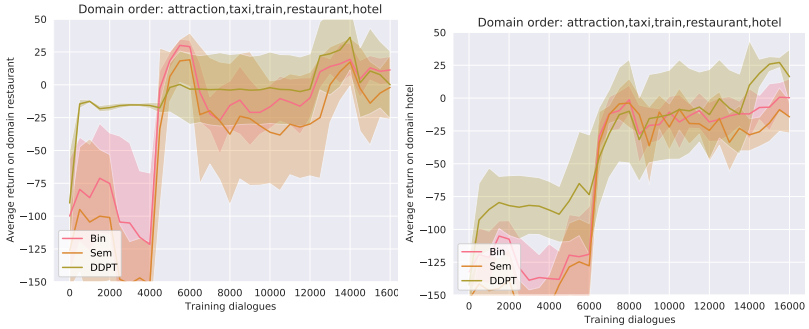

(d) Average return on restaurant domain

(e) Average return on hotel domain

Figure 9: Average return for each individual domain, where algorithms are trained in the order easy-to-hard.

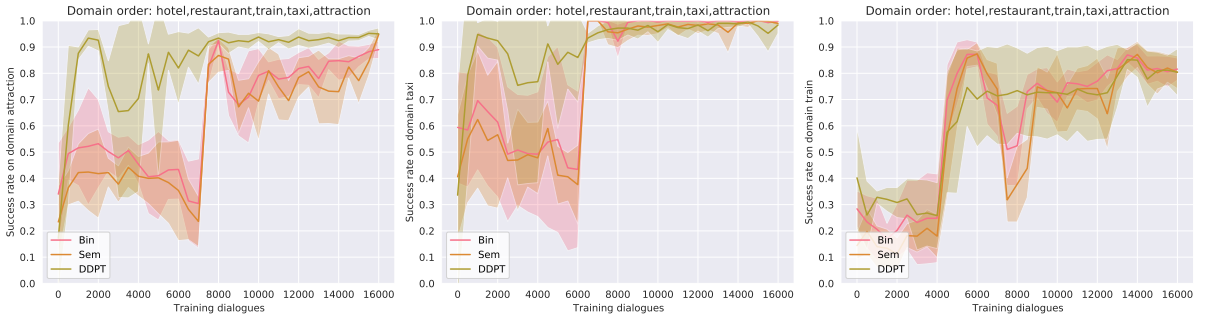

(a) Success rate on attraction domain

(b) Success rate on taxi domain

(c) Success rate on train domain

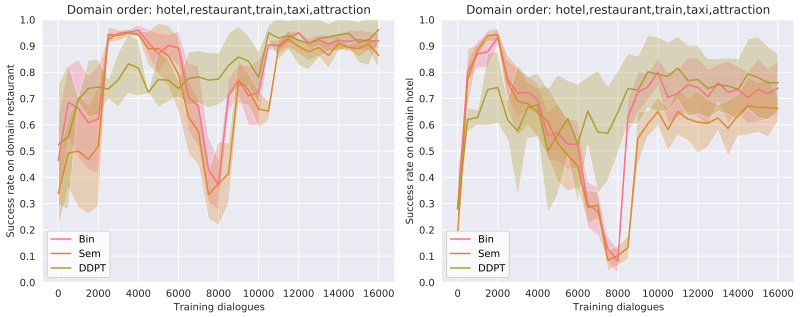

(d) Success rate on restaurant domain

(e) Success rate on hotel domain

Figure 10: Success rate for each individual domain, where algorithms are trained in the order hard-to-easy.

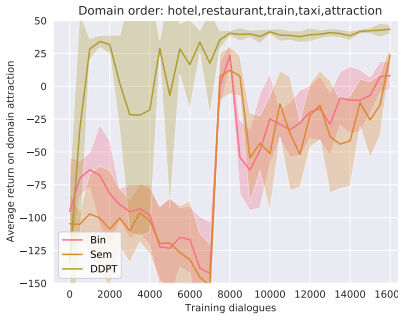

(a) Average return on attraction domain

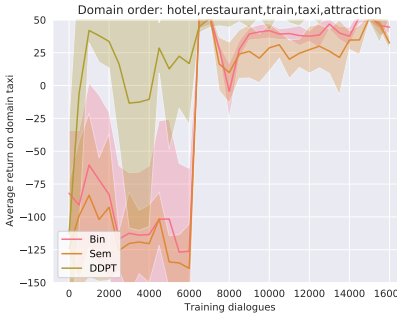

(b) Average return on taxi domain

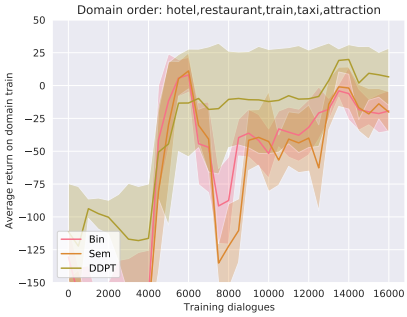

(c) Average return on train domain

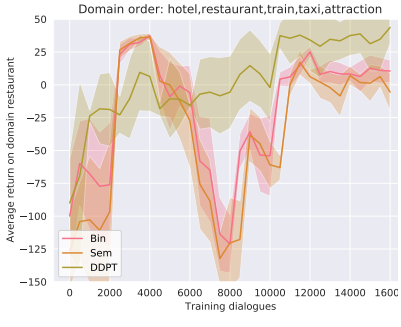

(d) Average return on restaurant domain

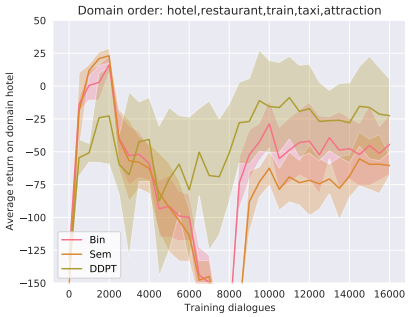

(e) Average return on hotel domain

Figure 11: Average return for each individual domain, where algorithms are trained in the order hard-to-easy.

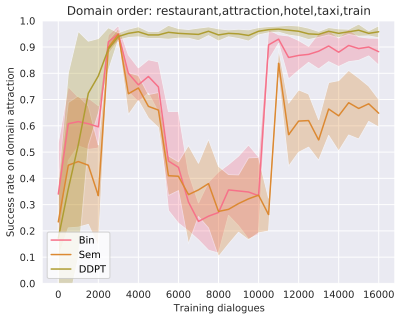

(a) Success rate on attraction domain

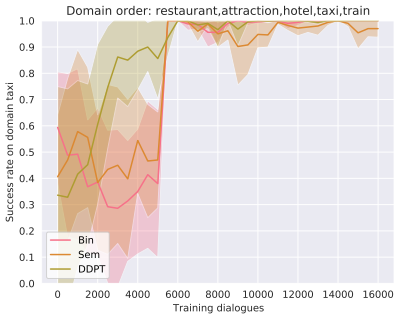

(b) Success rate on taxi domain

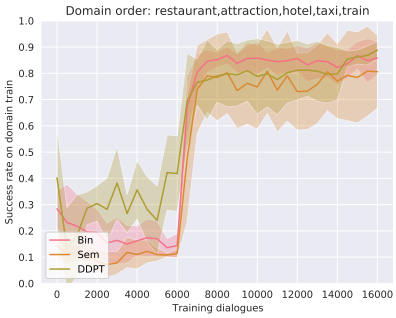

(c) Success rate on train domain

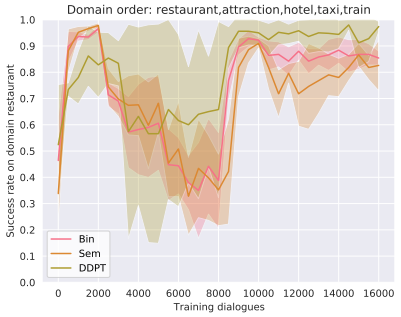

(d) Success rate on restaurant domain

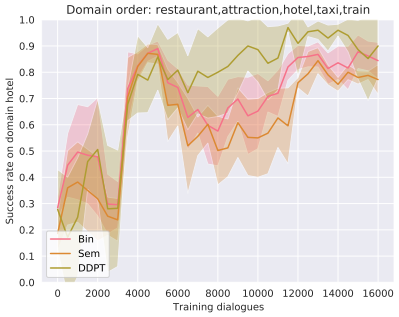

(e) Success rate on hotel domain

Figure 12: Success rate for each individual domain, where algorithms are trained in the order mixed.

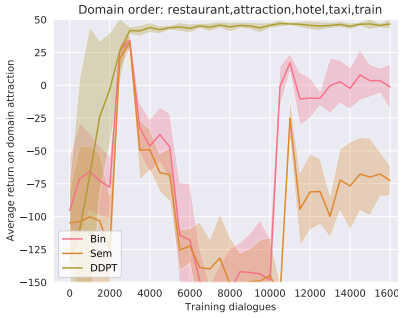

(a) Average return on attraction domain

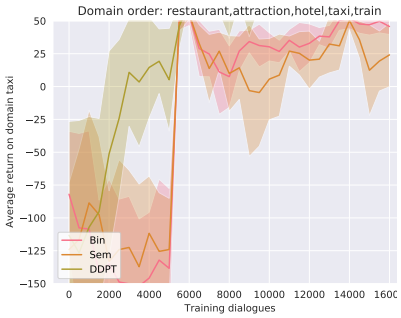

(b) Average return on taxi domain

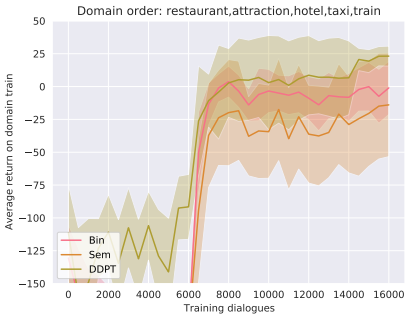

(c) Average return on train domain

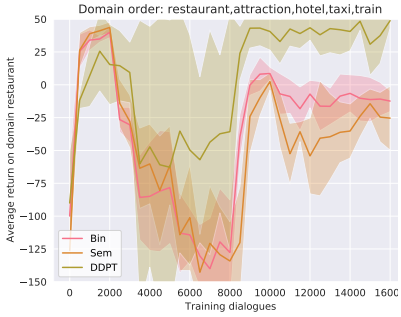

(d) Average return on restaurant domain

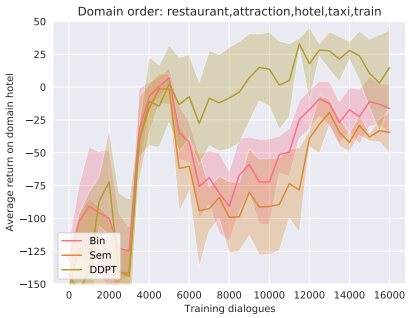

(e) Average return on hotel domain

Figure 13: Average return for each individual domain, where algorithms are trained in the order mixed.

GRAVL-BERT: Graphical Visual-Linguistic Representations for Multimodal Coreference Resolution

Danfeng Guo[1]*, **Arpit Gupta**[2], **Sanchit Agarwal**[2]†, **Jiun-Yu Kao**[2]
Shuyang Gao[2], **Arijit Biswas**[2], **Chien-Wei Lin**[2], **Tagyoung Chung**[2], **Mohit Bansal**[2,3]
[1]University of California, Los Angeles
[2]Amazon Alexa
[3]University of North Carolina, Chapel Hill

Abstract

Learning from multimodal data has become a popular research topic in recent years. Multimodal coreference resolution (MCR) is an important task in this area. MCR involves resolving the references across different modalities, e.g., text and images, which is a crucial capability for building next-generation conversational agents. MCR is challenging as it requires encoding information from different modalities and modeling associations between them. Although significant progress has been made for visual-linguistic tasks such as visual grounding, most of the current works involve single turn utterances and focus on simple coreference resolutions. In this work, we propose an MCR model that resolves coreferences made in multi-turn dialogues with scene images. We present GRAVL-BERT, a unified MCR framework which combines visual relationships between objects, background scenes, dialogue, and metadata by integrating Graph Neural Networks with VL-BERT. We present results on the SIMMC 2.0 multimodal conversational dataset, achieving the rank-1 on the DSTC-10 SIMMC 2.0 MCR challenge with F1 score 0.783. Our code is available at https://github.com/alexa/gravl-bert.

1 Introduction

Powered by advances in machine learning, intelligent agent systems have seen their capacity expand in recent years. Devices with intelligent assistants have become ubiquitous in everyday life. These systems can handle short task-oriented dialogues, but are limited to speech or text inputs and outputs. Motivated by the widespread adoption of such agents and multimodal devices with screens that house them, multimodal (visual-linguistic) understanding has become a promising discipline for researchers. The next-generation of intelligent assistants are expected to jointly understand multimodal data, i.e., text, image, video, and audio together and their associations.

Figure 1: An example of Multimodal coreference resolution (MCR). Based on the image and dialogue context, there can be multiple ways to refer to the highlighted red coat. Each of them require different information to locate the target.

Within multimodal understanding, an important area of research is Multimodal Coreference Resolution (MCR). It is a crucial capability as user references can span across modalities in a multimodal environment. Moreover, MCR is a challenging problem even when compared to text coreference resolution and visual question answering tasks because, in MCR, two participants can simultaneously refer to objects while looking at a scene from a shared perspective. In this dynamic frame of reference, the notion of left/right, first/second is constantly shifting and the model cannot rely on the fixed position of objects in the scene. Different from both VQA and textual anaphora resolution tasks, there are distinct ways to refer to an object in the MCR task. As the scene may contain a large number of similar objects, it is natural to refer to them by relative position with respect to other objects, front and back w.r.t the camera, and w.r.t. accessory objects like shelves and tables, so as to easily indicate the target object(s). Such references are unusual in other related tasks.

Furthermore, as our focus is on the MCR task within task-oriented dialogues, the objects being

*corresponding author: lyleguo@g.ucla.edu
†corresponding author: agsanchi@amazon.com

Proceedings of the 29th International Conference on Computational Linguistics, pages 285–297
October 12–17, 2022.

referenced are typically associated with a back-end database that also provides metadata information such as price, brand, and size for these objects. This adds an additional dimension to the coreferencing task as the users can also refer to objects based on these non-visual metadata attributes. Finally, as conversations involve multi-turn dialogue, MCR also requires reasoning over dialogue context to resolve references like "*show coat next to the shirt you suggested previously*". An example is shown in Figure 1. Resolving these coreference cases requires various multimodal information extraction and reasoning capabilities. For example, to identify "*the red coat next to the black one*", the model must infer the target's visual features (red), as well as its neighborhood information (a black coat). The user might also say "*I'll take the first one*", if they are choosing from one of the coats offered previously.

Notable progress has been made on multimodal frameworks. Recent models have shown excellent performance on various multimodal tasks including Visual Question Answering (VQA) (Huang et al., 2019b; Su et al., 2020), Visual Commonsense Reasoning (Zheng et al., 2020b; Su et al., 2020), Visual Grounding (Zheng et al., 2020a; Deng et al., 2018) and Image Captioning (Huang et al., 2019a). All these models take both images and text as inputs. Both visual and linguistic tokens are sent to an autoencoder to learn shared representation and perform downstream tasks. However, such methods are still not developed in a way to be able to solve complex MCR scenarios. For example, they may fail when a query has dialogue context instead of a single short sentence, or when there are anaphoric references in the query. Finally, most of these models are not designed to handle external knowledge sources or to scale to hundreds of objects in a single scene.

Motivated by these challenges, we propose a new framework, GRAVL-BERT(Graphical Visual-Linguistic BERT), that can simultaneously reason over dialogues, objects and their relationships, scene information, and object metadata. Our major contributions are as follows:

1. We present GRAVL-BERT, a unified BERT-based framework for encoding and reasoning over dialogues grounded in scenes.
2. GRAVL-BERT
   (a) Incorporates additional knowledge sources in the form of object metadata to also support coreferencing based on non-visual features like brand and price.
   (b) Represents scene objects as a graph and encodes them using Graphical Convolutional Network (GCN) to enable reasoning and coreferencing involving complex spatial relationships.
   (c) Adds information about object's surrounding by explicitly sampling from its neighborhood and generating captions describing the object. This enables coreferencing involving surrounding context (e.g., accessory objects like shelves and tables)
3. Finally, we show the importance of pre-training on dialogue dataset for the task of MCR.

We present results of GRAVL-BERT on the SIMMC 2.0 dataset (Kottur et al., 2021) which involves dialogues between a customer and an agent in the shopping domain. We participate in the SIMMC 2.0 challenge for the task of MCR, where the goal is to resolve the references and identify the target object(s) in the scenes. We achieve SOTA performance with 0.76 object-level F1 score on devtest set and 0.78 on test set. We note that this dataset is available for research and non-commercial use.

2 Related Works

Visual Grounding. Visual Grounding (VG) is an area very close to MCR. Given a query, it aims to find the most relevant target in an image. Some widely-used datasets for VG are RefCOCO, RefCOCO+, and RefCOCOg (Yu et al., 2016). Their queries are usually short and simple, e.g., the cat jumping over the fence. JR-Net (Jain and Gandhi, 2021) which achieves SOTA on this task encodes images and queries separately and then uses a joint-reasoning and a multi-level fusion module to merge the features and generate the results. VLT (Ding et al., 2021) converts image features into the same format as language token embeddings and uses BERT followed by a masked decoder to locate the target. A-ATT (Deng et al., 2018) concatenates visual and linguistic features together and uses accumulative attention layers to focus on the key targets.

Visual-Linguistic Frameworks. Multimodal frameworks that support visual (i.e., image, video) and linguistic inputs (i.e., caption, dialogue) can be fine-tuned for various tasks including MCR. In early works, most models encoded visual and

linguistic features separately and combined them only at a later stage. For example, both MTN (Le et al., 2019) and LXMERT (Tan and Bansal, 2019) have two separate encoders for visual and linguistic inputs and then use a query-aware encoder and cross-modality encoder respectively to extract visual features related to query text. In recent works, increasing number of high-performing models adopt an early-fusion strategy. They first extract the regions of interest (ROI) features from visual inputs, convert them into token embeddings, and concatenate with text embeddings. Then they employ BERT (Devlin et al., 2019) to learn the cross-modal associations and perform different tasks. For instance, ViL-BERT (Lu et al., 2019) concatenates text embeddings and image embeddings together and sends them to BERT. Oscar (Li et al., 2020b), VinVL (Zhang et al., 2021), 12-in-1 (Lu et al., 2020), Unicoder-VL (Li et al., 2020a), and Unified VLP (Zhou et al., 2020) all improve upon ViL-BERT by adding better image features, new pretraining strategies, or multiple datasets for pretraining. VL-BERT (Su et al., 2020) is currently one of the most popular benchmark frameworks for visual-linguistic tasks. It uses Fast R-CNN to extract visual features of objects and scene images, and concatenates them with linguistic token embeddings. The combined features are then fed into a BERT module, which is fine-tuned for various downstream tasks.

Graphical Models. Aside from the aforementioned works, another approach is to represent all ROIs in the scene as one graph. Graph R-CNN(Yang et al., 2018) and GCN-LSTM (Yao et al., 2018) encode images as graphs whose nodes represent objects and edges represent the relationships between objects. The generated graph representations can be used for downstream tasks like VQA and Image Captioning. (Damodaran et al., 2021; Yang et al., 2019) show that scene graphs improve model performance on these tasks.

3 Methodology

Problem Formulation. Given inputs (D, I, N, M, Q) where $D = \{D_1, D_2, ..., D_k\}$ is the dialogue text split into k turns, $I = \{I_1, I_2, ..., I_k\}$ are the scene images for the dialogue turns, $N_j = \{N_{j1}, N_{j2}, ...\}$ is the set of objects inside each scene $I_j \in I$, $M_{ij} = \{M_1, M_2, ...\}$ are the metadata attributes of each object N_{ij} and Q is the user query referring to one or more objects, our task is to predict a label $y_n \in \{0, 1\}$ for each object $n \in N$ that indicates whether the object n is being referenced by the query. The user query can involve spatial references, visual references, metadata based references, or any combination of these.

Model Details. Our model builds upon VL-BERT. We extend the framework from single utterance input to multi-turn dialogue input, with each dialogue turn associated with its respective scene image. In addition, the scene objects can have external knowledge base (e.g., metadata) associated with them. The architecture for our model is shown in Figure 2. The model takes 4 different streams of input: linguistic, visual, segment, and position. Input from different streams are combined via feature-wise addition.

The visual stream consists of the visual features from the whole scene, the candidate object and its surroundings. For each of the visual component, we use Fast-RCNN (Girshick, 2015) to extract the features and then augment it with bounding box location to add spatial information. We further add a GCN layer to explicitly capture the relative position of each object with respect to others.

The linguistic stream includes the dialogue context, user query and candidate object's metadata. We flatten the structure of metadata and convert it into a string of the form *"key1 value1 key2 value2 ..."*. We also add two special tokens, an integer feature T which denotes the distance from the turn when the object was last offered by agent, and a string S, which indicates that the corresponding visual features (in the visual stream) are from the object's surroundings and not the object itself.

The segment stream is used to distinguish the dialogue context, user query and object metadata inputs. It has three different values, corresponding to these input types. The position stream contains token positional embeddings, which is same as the one used in the original BERT model.

Note that, we feed only one object into the model at a time. This allows our system to scale well for scenes that may contain large number of objects. For instance, in the SIMMC 2.0 dataset, many scenes contain more than 100 objects with metadata sequence length larger than 50 for each object, making it impractical to feed all 5K object instances into the system. We supplement our model with GCN based structure to mitigate strong independence assumptions

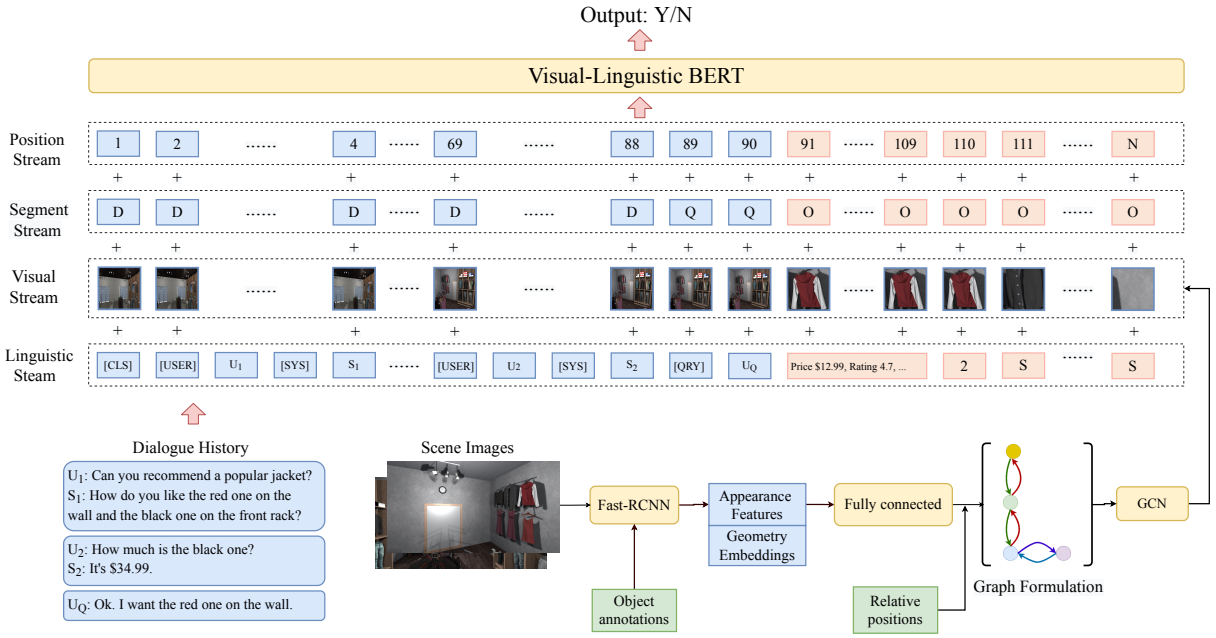

Figure 2: A visualization of our MCR model. It has four streams. The linguistic stream consists of dialogue, query, metadata text, coreference distance and an object surrounding indicator token S. In this example, the coreference distance is 2 because the candidate object is mentioned 2 turns earlier in the dialogue history. The visual stream consists of visual features of the whole scene, the candidate object and its surrounding area. The scene features and object features are repeated such that the visual stream has the same length as the linguistic stream. Image features are extracted using Fast-RCNN backbone and processed by a GCN module. The segment stream is to distinguish the dialogue, query and other tokens. The position stream indicates token positions.

implied by considering only one object at each inference step. Furthermore, feeding more objects to GCN is less memory-intensive than VL-BERT. This is because, for VL-BERT, the sequence length consists of $len(dialogue\_history) + len(metadata) + num(objects)$, while for GCN it is just $num(objects)$ and the dialogue history can be arbitrarily long (e.g., >200 tokens for SIMMC).

We next describe the three major contributions in our model: GCN Structure, Reference Distance, and Environmental Information Encoding.

3.1 GCN Structure

In a scene with multiple objects, it is natural to refer to an object using its attributes combined with spatial information relative to other objects in the scene. For instance, in the referring expression *"the black cat on the yellow sofa"*, object attributes are *"black cat"* and *"yellow sofa"* and the spatial relationship is *"on"*. Graphical approaches can effectively capture such spatial relations by creating edges between neighboring objects. We, therefore, introduce a GCN layer in our model to augment the raw visual features of an object (extracted from Fast-RCNN) with information of its neighbors by adding the spatial relationships between them.

Graph Formulation: We represent all the objects in a scene as one graph. The nodes represent the visual features of the objects and the edges represent the positional relationships between them. An example is shown in Figure 3. There are four basic edges: *top*, *bottom*, *left*, and *right*. We add an additional node that represents the features of the whole scene. It is connected to all the other nodes with a fifth edge type - *inside*, that indicates an object lies inside the scene. We expect this setup to capture the global information of the full scene into the object representations.

GCN Layer: For a node v with feature h_v^0, let its neighbors be $\mu \in \varepsilon$ whose features are h_μ^0. Each $\mu - v$ edge has a type $l \in L$. The purpose of GCN is to update h_v^0 using all h_μ^0.

We use the FiLM-GCN (Brockschmidt, 2020) model. Proposed in 2019, it is a GCN specially designed to support multiple edge types. Its equation is represented as

$$\beta_{l,v}^t, \gamma_{l,v}^t = g(h_v^t; \theta_{g,l})$$
$$h_v^{t+1} = l\Big(\sum_{u \xrightarrow{l} v \in \varepsilon} (\sigma(\gamma_{l,v}^t \odot W_l h_u^t + \beta_{l,v}^t); \theta_t)\Big)$$

For a node v, its current representation h_v^t is first

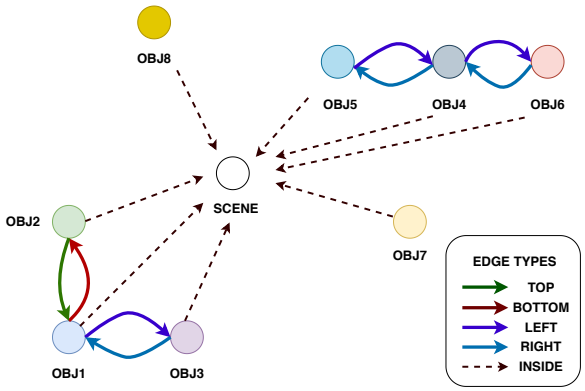

Figure 3: An example of graph formulation. It consists of five types of edges which indicate five positional relations between objects. The center *scene* node represents the features of the whole image.

Figure 4: A sampling of object's neighborhood. The center bounding box does not provide enough features for a model to recognize the ground cabinet. We supplement this input by adding features from its left, right, top and bottom directions.

passed to a function $g(\cdot)$ to compute two variables: the encoded representation $\beta^t_{l,v}$ and the element-wise weight factor $\gamma^t_{l,v}$ associated with each edge type l. The message passed from neighbors to v is represented as the element-wise product of $\gamma^t_{l,v}$ and $W_l h^t_\mu$. It is added to $\beta^t_{l,v}$ and passed to an activation function $\sigma(\cdot)$. The outputs are summed over all edges of v and finally sent to a linear function $l(\cdot)$ to become the new representation h^{t+1}_v.

3.2 Reference Distance

During conversation with a multimodal agent, users can refer to objects mentioned in an earlier turn of the dialogue. For instance, as shown in Figure 1, the user can refer to the red coat offered earlier by the system by saying *"the one you recommended before"*. To aid our model to *look back* in the dialogue history for resolving such references, we add a feature that indicates the distance from the query to the most recent system-mention of the candidate object in the dialogue history.

3.3 Environment Information Encoding

Scene images can contain visual entities that are not direct target objects, e.g., wall, table, shelves etc. As these entities are usually present in a small region of the image, their features may be attenuated during downsampling and not easily available to VL-BERT. To address this issue, we employ the following two approaches.

Object Surrounding: Sometimes the object region does not have all the information to allow for reference resolution, e.g., in Fig 4, given only central bounding box, it would be difficult to identify whether the jacket is on a table or cabinet. Therefore, we sample regions of fixed size around the object and feed them as supplementary information in the visual stream. Sampling only from the object's immediate neighbourhood is based on the intuition that people typically use items in the target object's vicinity to refer to it. The sampled region size is a tunable hyperparameter. Specifically, in this work, we consider eight surroundings regions from 8 directions as shown in Figure 4.

Image Captioning: Image captioning models generate descriptions that include the surrounding context in which an object is situated. For our task, these models generate captions that contain references to non-target objects, see Fig5. We use an off-the-shelf captioning model and generate captions for each object in the scene. We then augment our training dataset with captions as additional metadata. The captioning model that we use, is composed of an Alexnet (Krizhevsky et al., 2017) image feature encoder and a LSTM (Hochreiter and Schmidhuber, 1997) decoder. ROIs are extracted from scene images, resized and used as model inputs. We also perform a cleanup to remove redundancy from the generated captions before adding them to our training set.

4 Experiments and Results

4.1 Dataset

We evaluate our approach on Situated and Interactive Multimodal Conversations (SIMMC) 2.0 dataset(Kottur et al., 2021) released as part of DSTC10 Challenge 2021. It contains 11k task-oriented dialogues between a user and an agent, grounded in photo-realistic virtual reality (VR)

(a) (b)

Figure 5: Examples showing additional metadata extracted from captions generated for scene objects. We augment our training set with these generated attributes to help with resolving coreferences involving surrounding context.

		Precision	Recall	F1
Dev	Ours	0.74	0.83	0.78
Devtest	Ours	0.74	0.78	0.76
Test	Ours	N/A	N/A	**0.78**
	BART-based	N/A	N/A	0.76
	Huang et al. (2021)	N/A	N/A	0.73

Table 1: Our model GRAVL-BERT's performance w.r.t object-level precision, recall and F1 scores on SIMMC 2.0 evaluation sets. Our model is compared with other systems on test set. Only the F1 score is provided by the DSTC challenge officials.

scenes from fashion and furniture stores. Each example contains four elements: dialogue between user and the agent, scene images associated with each dialogue turn, object annotations and metadata for all objects within the scene, and the referring query. The dataset also provides spatial relationship between objects (left, right, top, bottom), which we use to construct graphs. The data is split into train (65%), dev (10%), dev-test (10%), and test-std (15%).

4.2 Experimental Setup

We continue to pre-train our model on in-domain dialogues from the SIMMC Dataset using masked language modeling (MLM) objective. We mask 30% dialogue tokens and train the model to predict these tokens. The pre-trained model was then fine-tuned for the MCR task by progressively building on techniques described in Section 3. We limit the dialogue context length to 3. This was done for two reasons. First, from a practical standpoint, 512 tokens is the maximum sequence length that the transformer module can consume. Secondly, it is reasonable to assume that most users will refer to objects seen recently in the context as opposed to far back in the conversation. Further, we downsample negative examples to maintain positive-negative ratio to $1:5$ for training. We use object-level binary cross entropy for loss. All models are trained on 4 Tesla V100-SXM2 GPUs.

As explained in Sec3.3, we use Alexnet-LSTM captioning model to capture each object's surrounding context by generating captions describing it. In order to fine-tune the captioning model, we mine queries from the SIMMC training set (using keyword-matching heuristics) that involve references based on surrounding objects like tables, racks, stands etc. The captioning model is then trained to generate these queries given enlarged bounding boxes enclosing the corresponding target objects as inputs. For this task, there are 1190 training examples. After training, we use the model to generate captions for all scene objects, and then perform a basic procedure to extract descriptions of surrounding context from the captions. Specifically, we create a pool of non-target objects, search the generated captions for these objects and construct phrases on them. Finally, these phrases such as "on the table" and "in the closet" are added to the dataset as additional metadata text. An example is presented in Figure 5.

For inference, we make predictions over all objects inside the scene. Objects with score ≥ 0.5 are marked as referred objects. We report the model performance with object-level precision, recall and F1 score. DSTC10 uses object-level F1 score as official metric. We report our results on devtest and test set.

4.3 Primary Results

The results are shown in Table 1. The test precision and recall are missing as the ground truth labels have not been released by the challenge officials and only F1 score is reported. Our model has the highest performance among 16 participating teams. It outperforms the second best system[1] (model based on BART (Lewis et al., 2020)) by $\sim 2.5\%$ and the third best system (model based on UNITER (Chen et al., 2020)) by $\sim 5\%$.

[1] https://github.com/KAIST-AILab/DSTC10-SIMMC

Experiments	Precision	Recall	F1
Vanilla VL-BERT	0.46	0.49	0.47
Mask-out Metadata	0.26	0.57	0.36
Mask-out Visual Feats	0.37	0.01	0.02

Table 2: Object-level precision, recall and F1 score of models trained using different kinds of inputs. Measured on validation set of SIMMC 2.0.

Experiments	Precision	Recall	F1
Vanilla VL-BERT	0.46	0.49	0.47
Dialogue Pre-trained VL-BERT	0.57	0.83	0.68
VD-BERT	0.53	0.89	0.66

Table 3: Comparison of VD-BERT and VL-BERT. Both pretrained on dialogue dataset using mask language modeling. Measured on validation set of SIMMC 2.0.

4.4 Ablation Study

Contribution of Metadata and Visual Features. To understand the impact of metadata information and visual features in the model's performance, we start with vanilla VL-BERT and mask-out i.e. zero-out either of the two features and then train and evaluate the system with all the other inputs. As seen in Table. 2, both metadata and visual features provide complementary information and contribute to the model's final performance. When trained without metadata features, the model cannot resolve coreferences based on attributes like brand and price. At the same time, visual features are essential for referencing based on visual characteristics like color and pattern. The F1 score drops close to zero when visual features are masked out because, in this dataset, most queries involving reference by metadata attributes also include visual characteristics, e.g., *the blue Nike one*.

Impact of Dialogue-Oriented Pre-training. To quantify the importance of pre-training on dialogue datasets, we train our system with original VL-BERT (pre-trained on Conceptual Captions (Sharma et al., 2018), Book Corpus (Zhu et al., 2015) and English Wikipedia datasets) and compare it with VL-BERT further pre-trained on SIMMC 2.0 dialogue dataset. As shown in Table 3, dialogue-specific pre-training provides significant performance gain on the multimodal coreference task. We also fine-tune a VD-BERT (Wang et al., 2020) model (pre-trained on VisDial (Das et al., 2019) visual dialogues dataset). VD-BERT has similar structure as VL-BERT but is trained on dialogues instead of captions. It provides similar gains as dialogue pre-trained VL-BERT reinforcing our hypothesis that dialogue pretraining is important for this task. We use VL-BERT as our base architecture because it has been pre-trained on much larger datasets compared to VD-BERT and thus can provide better generalization.

Experiments	Precision	Recall	F1
Pretrained VL-BERT	0.5734	0.8293	0.6780
+GCN	0.6432 (+0.0698)	0.8122 (−0.0175)	0.7179 (+0.0399)
+Reference Distance	0.7249 (+0.0822)	0.8238 (+0.0118)	0.7712 (+0.0538)
+Neighbour Features	0.7316 (+0.0067)	0.8248 (+0.0010)	0.7753 (+0.0041)
+Captions	0.7410 (+0.0093)	0.8306 (+0.0058)	0.7833 (+0.0080)

Table 4: Contribution of different modules. The experiments are cumulative. "+" means the current experiment is based on the above row with the indicated module added. The change of metrics corresponding to above row is shown in bracket. Performance is measured on validation set of SIMMC 2.0.

	Spatial	Non-tgt Objects	Dialogue History	Meta-data	Pure Visual
Pretrained VLBERT	5.0%	4.5%	3.7%	7.4%	4.0%
GRAVL-BERT	1.8% (−64%)	1.6% (−64%)	1.1% (−70%)	2.0% (−73%)	1.7% (−58%)

Table 5: Object-level error rate per coreference type.

Contribution of Various Model Components. To study the contribution of each component of our model architecture (GCN, reference distance, surrounding features, and captions), we perform ablation experiments. Starting from the dialogue pre-trained VL-BERT, we add our modifications incrementally and measure the gain in F1 score. The results are in Table 4. Adding GCN and reference distance provides significant gains. Adding surrounding features and captions further improves the performance by 0.41% and 0.80% respectively.

Note that, most queries refer to an object using multiple attributes. For example, *"the red Nike*

shirt to the left of the blue one in the cabinet.", which includes references by color (red), brand (Nike), relative position (left of the blue one) and absolute position (in the cabinet). Some of these information may be redundant and not jointly required to uniquely locate the target. This may explain only minor improvements in the last two rows in Table 4.

4.5 Qualitative Analysis

We examine our model's predictions to get a sense of its strengths and weaknesses. We provide a breakdown of the performance on different co-reference types in Table 5. For this analysis, we assigned examples to exactly one co-reference category based on heuristics. Compared with baseline, GRAVL-BERT provides significant improvement on all types of coreferences. We notice that the model is able to resolve complicated references such as in Figure 6a. This example involves pointing to multiple objects and requires visual understanding and spatial reasoning (absolute and relative). In Figure 6b, the model is able to successfully utilize long dialogue context to resolve coreferences to two objects. We also looked at several failure cases. To resolve the coreferences in these cases, a deeper scene understanding is required. For example, in Figure 6c, the model needs to infer the number of cabinets, and then locate the second one from the left. We show more examples in the Appendix.

5 Conclusion and Discussion

In this work, we proposed a multimodal framework GRAVL-BERT for MCR task. Our contribution lies in systematically combining relevant techniques such as utilizing external knowledge sources (metadata, generated captions), GCN, sampling object's neighborhood, and dialogue-oriented pre-training using a simple BERT-based architecture to perform MCR within dialogues grounded in scenes. We improved over the GPT-2 (Kottur et al., 2021) based baseline by 33.9% absolute and 2.5% over other concurrent work.

For future research in the topic we suggest several avenues. As mentioned in Section 4.5, to identify objects with complex references, global information is required e.g. number of cabinets in the scene, price ranking among objects. Currently, our model is unable to handle this complexity. We believe adding specific encoders to extract these global features will be helpful.

In the pre-training stage, we trained our model on dialogues using MLM. We expect that applying prompt in this stage may have a promising performance. Instead of training the model to learn to

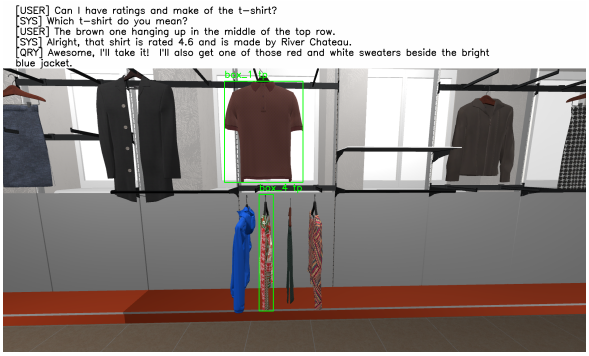

(a) An example with correct predictions. There are two target objects (highlighted with boxes), both of which our model gets right. Coreferencing requires both visual (*"brown one"*, *"red and white sweaters"*) and spatial understanding (*"middle of the top row"*, *"besides the bright blue jacket"*).

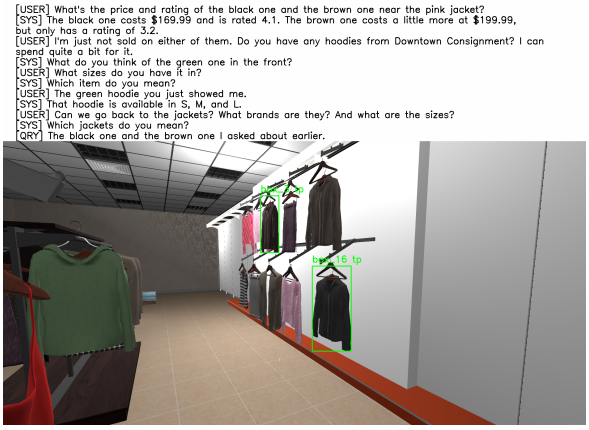

(b) A successful example where the model is able to utilize full dialogue context for coreferencing. In this case, the model needs to obtain context from the first turn to point to the two objects being referred in query.

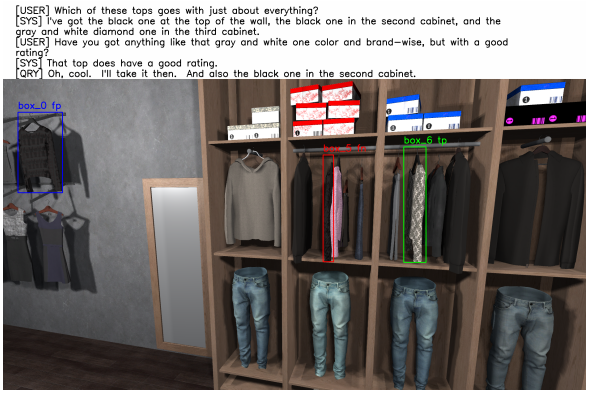

(c) An example of incorrect case. Our model is not able to locate "the second cabinet."

Figure 6: Model predictions on few examples from SIMMC 2.0 devtest set.

predict the randomly masked words, using a carefully designed prompt can teach model to focus on the information that is helpful to our main task.

Lastly, there are cases where the user provides very general descriptions and the information to resolve the coreference is insufficient. For example, the user refers to "the red sweater" while there are multiple red sweaters in the scene. In this situation, instead of trying to resolve the coreference, the system may attempt to disambiguate. (e.g., *"Which sweater do you mean?"*) We expect future work to distinguish these kinds of situations.

References

Marc Brockschmidt. 2020. GNN-FiLM: Graph neural networks with feature-wise linear modulation. In *ICML*, volume 119 of *PMLR*, pages 1144–1152. PMLR.

Yen-Chun Chen, Linjie Li, Licheng Yu, Ahmed El Kholy, Faisal Ahmed, Zhe Gan, Yu Cheng, and Jingjing Liu. 2020. Uniter: Universal image-text representation learning. In *ECCV*.

Vinay Damodaran, Sharanya Chakravarthy, Akshay Kumar, et al. 2021. Understanding the role of scene graphs in visual question answering. *CoRR*, abs/2101.05479.

Abhishek Das, Satwik Kottur, Khushi Gupta, Avi Singh, Deshraj Yadav, Stefan Lee, José M. F. Moura, Devi Parikh, and Dhruv Batra. 2019. Visual dialog. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 41:1242–1256.

Chaorui Deng, Qi Wu, Qingyao Wu, et al. 2018. Visual grounding via accumulated attention. In *CVPR*.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *ACL*, pages 4171–4186, Minneapolis, Minnesota. Association for Computational Linguistics.

Henghui Ding, Chang Liu, Suchen Wang, and Xudong Jiang. 2021. Vision-language transformer and query generation for referring segmentation. In *ICCV*.

Ross Girshick. 2015. Fast r-cnn. In *International Conference on Computer Vision (ICCV)*.

Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long Short-Term Memory. *Neural Computation*, 9(8):1735–1780.

L Huang, W Wang, Y Xia, and J Chen. 2019a. Adaptively aligned image captioning via adaptive attention time. In *NIPS*, pages 8942–8951.

Pingping Huang, Jianhui Huang, Yuqing Guo, Min Qiao, and Yong Zhu. 2019b. Multi-grained attention with object-level grounding for visual question answering. In *ACL*, pages 3595–3600, Florence, Italy. Association for Computational Linguistics.

Yichen Huang, Yuchen Wang, and Yik-Cheung Tam. 2021. Uniter-based situated coreference resolution with rich multimodal input. *arXiv preprint arXiv:2112.03521*.

Kanishk Jain and Vineet Gandhi. 2021. Comprehensive multi-modal interactions for referring image segmentation. *arXiv preprint arXiv:abs/2104.10412*.

Satwik Kottur, Seungwhan Moon, Alborz Geramifard, and Babak Damavandi. 2021. Simmc 2.0: A task-oriented dialog dataset for immersive multimodal conversations. *arXiv preprint arXiv:2104.08667*.

Alex Krizhevsky, Ilya Sutskever, and Geoffrey E. Hinton. 2017. Imagenet classification with deep convolutional neural networks. *ACM*, 60(6):84–90.

Hung Le, Doyen Sahoo, Nancy F. Chen, and S. Hoi. 2019. Multimodal transformer networks for end-to-end video-grounded dialogue systems. In *ACL*.

Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Veselin Stoyanov, and Luke Zettlemoyer. 2020. Bart: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. In *acl*, pages 7871–7880.

Gen Li, Nan Duan, Yuejian Fang, et al. 2020a. Unicoder-vl: A universal encoder for vision and language by cross-modal pre-training. *AAAI*, 34(07):11336–11344.

Xiujun Li, Xi Yin, Chunyuan Li, et al. 2020b. Oscar: Object-semantics aligned pre-training for vision-language tasks. In *ECCV*.

J. Lu, V. Goswami, M. Rohrbach, et al. 2020. 12-in-1: Multi-task vision and language representation learning. In *CVPR*, pages 10434–10443, Los Alamitos, CA, USA. IEEE Computer Society.

Jiasen Lu, Dhruv Batra, Devi Parikh, and Stefan Lee. 2019. Vilbert: Pretraining task-agnostic visiolinguistic representations for vision-and-language tasks. In *NIPS*.

Piyush Sharma, Nan Ding, Sebastian Goodman, and Radu Soricut. 2018. Conceptual captions: A cleaned, hypernymed, image alt-text dataset for automatic image captioning. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 2556–2565, Melbourne, Australia. Association for Computational Linguistics.

Weijie Su, Xizhou Zhu, Yue Cao, et al. 2020. Vl-bert: Pre-training of generic visual-linguistic representations. In *ICLR*.

Hao Tan and Mohit Bansal. 2019. LXMERT: learning cross-modality encoder representations from transformers. In *EMNLP*.

Yue Wang, Shafiq Joty, Michael Lyu, et al. 2020. VD-BERT: A Unified Vision and Dialog Transformer with BERT. In *EMNLP*, pages 3325–3338, Online. Association for Computational Linguistics.

Jianwei Yang, Jiasen Lu, Stefan Lee, et al. 2018. Graph r-cnn for scene graph generation. In *ECCV*, pages 690–706, Cham. Springer International Publishing.

Xu Yang, Kaihua Tang, Hanwang Zhang, and Jianfei Cai. 2019. Auto-encoding scene graphs for image captioning. In *CVPR*.

Ting Yao, Yingwei Pan, Yehao Li, and Tao Mei. 2018. Exploring visual relationship for image captioning. In *ECCV*.

Licheng Yu, Patrick Poirson, Shan Yang, et al. 2016. Modeling context in referring expressions. In *Computer Vision – ECCV 2016*, pages 69–85, Cham. Springer International Publishing.

Pengchuan Zhang, Xiujun Li, Xiaowei Hu, et al. 2021. Vinvl: Revisiting visual representations in vision-language models. In *CVPR*.

Wenbo Zheng, Lan Yan, Chao Gou, and Fei-Yue Wang. 2020a. Webly supervised knowledge embedding model for visual reasoning. In *CVPR*, pages 12442–12451.

Wenbo Zheng, Lan Yan, et al. 2020b. Webly supervised knowledge embedding model for visual reasoning. In *CVPR*.

Luowei Zhou, Hamid Palangi, et al. 2020. Unified vision-language pre-training for image captioning and vqa. *AAAI*, 34(07):13041–13049.

Yukun Zhu, Ryan Kiros, Richard S. Zemel, Ruslan Salakhutdinov, Raquel Urtasun, Antonio Torralba, and Sanja Fidler. 2015. Aligning books and movies: Towards story-like visual explanations by watching movies and reading books. *2015 IEEE International Conference on Computer Vision (ICCV)*, pages 19–27.

A Appendix

In this appendix, we showcase more examples of our model's predictions. The figures below show 5 examples, out of which the first 4 are correctly predicted by our model. In Figure 7a, the target is referred by its color (*green*) and location (*top right*). In Figure 7b and 8a, the target is referred by non-visible attribute, object brand (*Ocean Wears* and *Modern Arts*). In Figure 8b the target is referred using previous dialogue context (*..that grey sofa you pointed out*). The prediction in Figure 9a is wrong. In this case, the targets are referred as the ones *with good ratings*. Our model, currently, does not have a mechanism to compare the ratings of all objects and select the top-rated ones. This can be improved in the future.

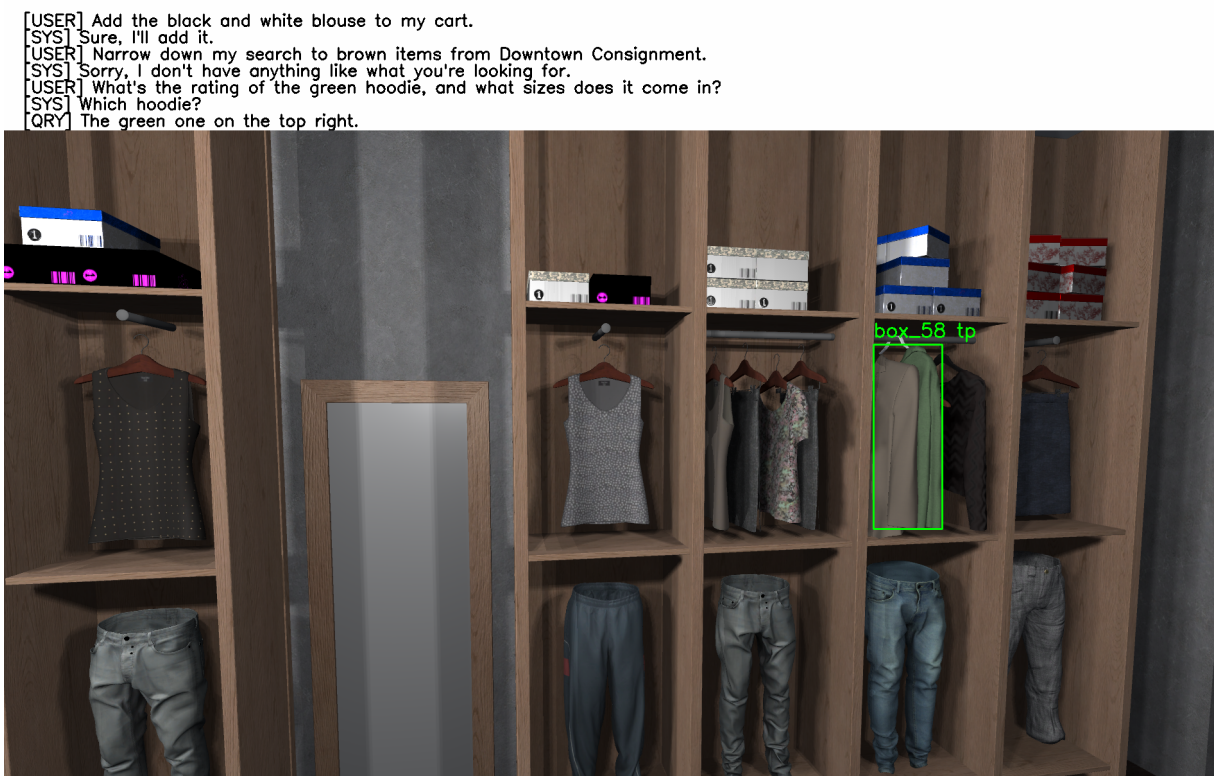

(a) An example of a correct case. The object in query is referred by its color and position.

(b) An example of a correct case. The object in query is referred by its brand.

Figure 7: More prediction results on devtest set.

[USER] Could I see a leather chair from Modern Arts?
[SYS] How about this chair up front? It's leather and comes from Modern Arts.
[QRY] Oh yeah, please add it to my cart.

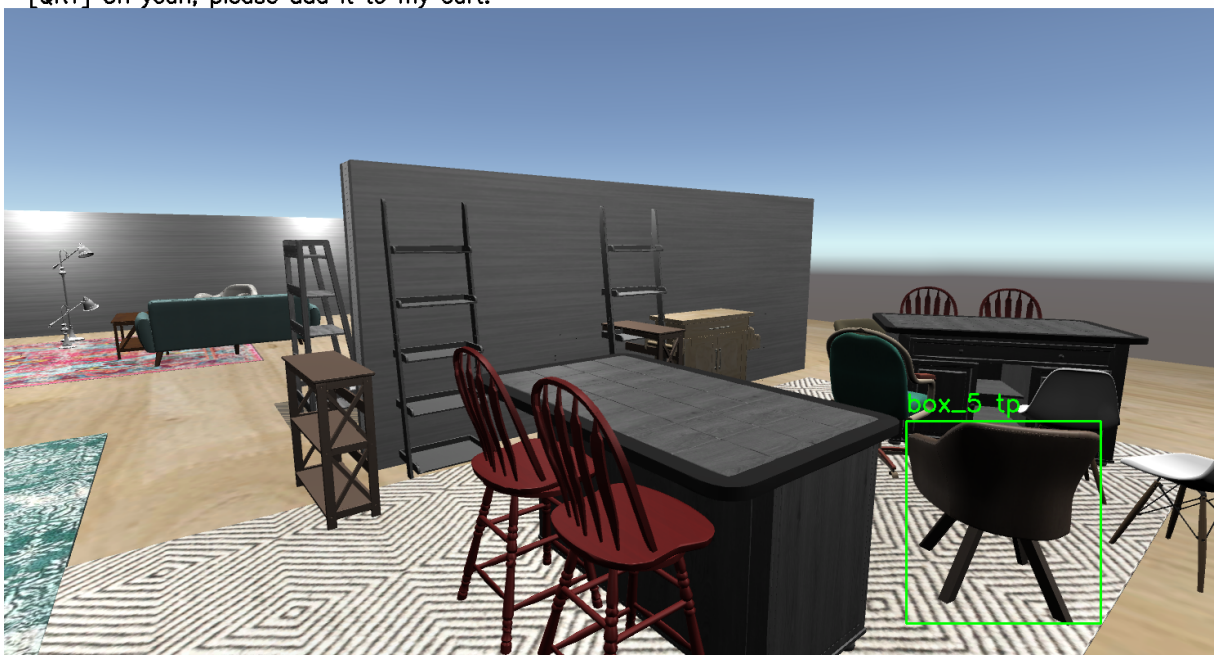

(a) An example of a correct case. The object in query is referred by its brand.

[USER] Could you spitball some suggestions for a Modern Arts sofa?
[SYS] Permit me to recommend the grey sofa in the very back.
[USER] Okay, now how about a leather armchair?
[SYS] Could you see yourself enjoying a movie in the beige chair straight ahead?
[USER] I want to see some more options.
[SYS] I'm sorry, but that's all I've got for you.
[QRY] Well, okay. I guess I'll go ahead and take one of those and that grey sofa you pointed out.

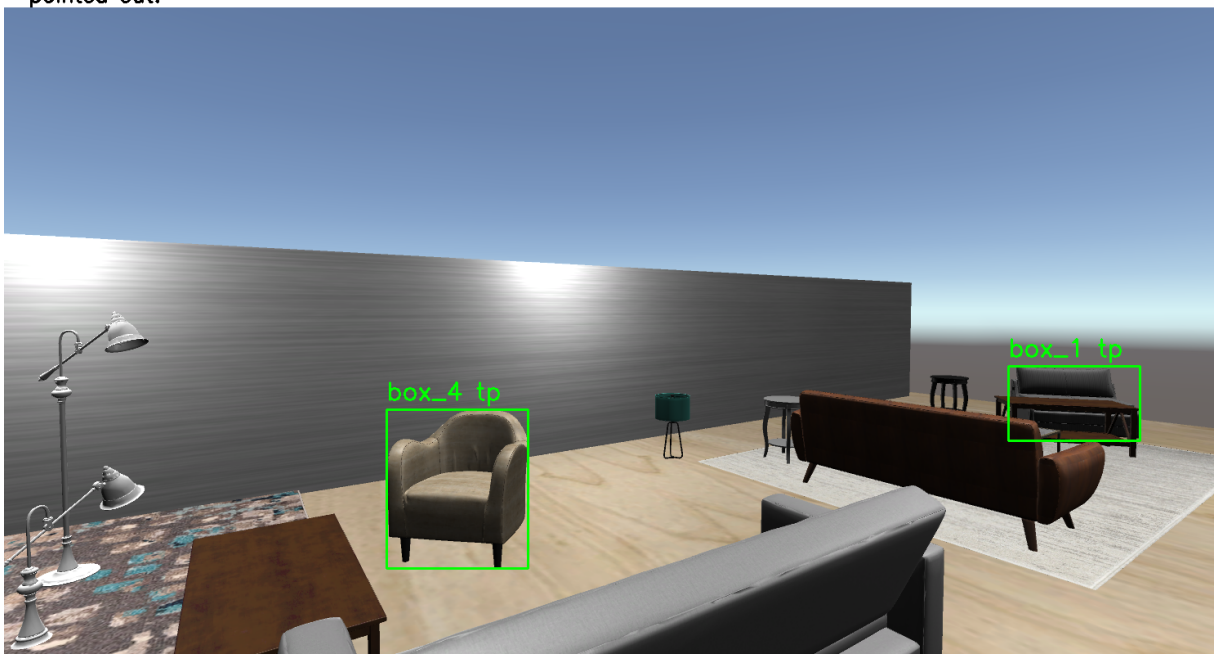

(b) An example of a correct case. The query refers to an object mentioned in early turns.

Figure 8: More prediction results on devtest set.

(a) An example of an incorrect case. Our model is unable to figure out which ones are "affordable" and "good" without comparing the price and rating of all objects.

Figure 9: More prediction results on devtest set.

Learning to Improve Persona Consistency in Multi-party Dialogue Generation via Text Knowledge Enhancement

Dongshi Ju, Shi Feng*, Pengcheng Lv, Daling Wang , Yifei Zhang
Northeastern University, Shenyang, China
judongshi@163.com, fengshi@cse.neu.edu.cn
ricardolvpc@foxmail.com
{wangdaling, zhangyifei}@cse.neu.edu.cn

Abstract

In an open-domain dialogue system, the consistent persona is a key factor to generate real and coherent dialogues. Existing methods suffer from the incomprehensive persona tags that have unique and obscure meanings to describe human's personality. Besides, the addressee information, which is closely related to express personality in multi-party dialogues, has been neglected. In this paper, we construct a multi-party personalized dialogue dataset and propose a graph convolution network model (PersonaTKG) with addressee selecting mechanism that integrates personas, dialogue utterances, and external text knowledge in a unified graph. Extensive experiments have shown that PersonaTKG outperforms the baselines by large margins and effectively improves persona consistency in the generated responses.

1 Introduction

Endowing a dialogue agent with a consistent persona has attracted an increasing amount of research interests, as it helps to deliver a more coherent and engaging conversation for users. Existing studies explore character personality through key-value persona pairs (Qian et al., 2018) or short descriptive sentences (Zhang et al., 2018b). The key-value pairs define a few of persona categories, such as name, gender, and age, which have clear semantic meanings. The descriptive text is declarative sentences with relatively fixed patterns that introduce one's persona information such as occupation and hobbies in the first person. Promising performance have been achieved on these 'well-defined' persona datasets (Mohapatra et al., 2021; Gu et al., 2021a; Song et al., 2020b).

Recently, Li et al. (2020) proposed a new dialogue dataset HLA-Chat using tags as persona information. HLA-Chat was collected from scripts of hit TV dramas, and the persona tags of the characters were tropes that are determined by audiences' impressions on TVTropes website[1]. For example, the famous character *Sheldon Cooper* from *The Big Bang Theory* has the persona tags *Book Dumb*, *Omnidisciplinary Scientist*, *Green Eyed Epiphany*, *Neat Freak*, *Token Minority*, etc. We can observe that these tags (i.e. tropes generated by TV audiences) are usually very distinctive and rare words, as they represent the unique persona of the character. Different from the general and comprehensible persona definition in Qian et al. (2018); Zhang et al. (2018b), the tags in HLA-Chat contain rich persona information but are difficult to understand, which set obstacles for the model to generate persona consistent responses. We also argue that this incomprehensible persona challenge is different from generating the response from sparse persona data (Zheng et al., 2020), where there are only limited personalized sentences in the dialogue context.

Intuitively, the persona consistency in the generated responses of HLA-Chat can be further improved by incorporating external knowledge. However, most of these rare persona words could not be found in the knowledge base or commonsense base such as ConceptNet (Speer et al., 2017) and ATOMIC (Sap et al., 2019). Thanks to TVTRopes, the audiences can contribute to the laconic and de-

A persona of Sheldon Cooper	
tag	Omnidisciplinary Scientist
sent	A scientist who knows everything about science.
doc	Related to the Nerd and the Mad Scientist, the Omnidisciplinary Scientist is a master of every branch of science, regardless of the branch in which they theoretically have a degree ...
Conversations	
Penny	I believe that when one door closes, another opens.
Sheldon Cooper	No,it doesn't. Not unless the two doors are conected by relays, or there are motion sensors involved.

Table 1: An example of persona tags with laconic and detailed interpretation and conversations involving the persona. The laconic interpretation (dubbed as sent) consists of one sentence, while the detailed interpretation is a long document that explicates the tag meaning.

*Corresponding author

[1] https://tvtropes.org/

Proceedings of the 29th International Conference on Computational Linguistics, pages 298–309
October 12–17, 2022.

tailed interpretation of these tropes on this wiki website, as shown in Table 1. In this paper, we collect the user generated interpretation on TVTRopes as external knowledge, and introduce text-based knowledge in the persona consistent response generation model with unique and incomprehensive persona tags.

On the other hand, most of the existing personalized response generation studies focus on bilateral dialogue (Wu et al., 2021; Majumder et al., 2021). In effect, the conversations in the real world often occurs between multiple speakers, which is called multi-party conversation (MPC). MPC is generally composed of speakers, utterances, and addressees (i.e. the recipient corresponding to an utterance). The expression of persona in the dialogue is often closely related to the addressee. For example, the addressee can be a friend, a lover, a stranger, etc. For these different relationships, every speaker may have a different way of expressing their personas, so predicting the addressee can assist the dialogue system to promote the persona consistency in the responses. Thus, we incorporate the addressee selecting mechanism into the response generation model and leverage a posterior selection module to improve the addressee prediction.

The complex context structure of MPC urges researchers to constantly seek novel and effective context modeling methods, where graph convolution networks have already achieved promising results (Liang et al., 2021; Hu et al., 2019). However, none of the existing hierarchy (Meng et al., 2018), role sensitive (Liu et al., 2019), or GCN-based modeling methods consider the different personas of the speakers as well as the correlation between personas and utterances. In this paper, we first utilize the hierarchical recurrent encoder-decoder structure (Serban et al., 2016; Xing et al., 2018) and the bidirectional GRU (Penghua and Dingyi, 2019) to model the multi-turn dialogue context and the personas of all the speakers. Then we build a unified graph with utterances and personas as nodes, and employ GCN to aggregate dialogue context information.

Our contributions are summarized as follows:

(i) We construct a new personalized dialogue dataset HLA-Chat++[2], where each incomprehensive persona tag has laconic and detailed text-based interpretations.

(ii) We propose a **Persona**-consistent response generation model based on **T**ext **K**nowledge enhanced **G**CN (PersonaTKG) with addressee selecting mechanism that integrates personas, dialogue utterances, and external text knowledge in a unified graph. To the best of our knowledge, this is the first study that explores the addressee selection in personalized dialogue generation tasks.

(iii) We conduct extensive experiments on HLA-Chat++ dataset, and the results have validated the effectiveness of incorporating text knowledge and addressee selection in improving persona consistency of generated responses.

2 Related Work

2.1 Persona Consistent Dialogue Generation

Maintaining persona consistency is essential to delivering more realistic and coherent conversations. To incorporate persona information into the dialogue system, Li et al. (2016) first used persona embedding to project each speaker into a dense vector. Kottur et al. (2017) proposed a neural dialogue model that simultaneously considers the contextual history of the speakers. However, these two models rely heavily on data with persona annotation, which are expensive and sparse. Qian et al. (2018) defined multiple key-value pairs to represent the personas of the speakers, including information such as name, gender, age, and residence, and explicitly displayed these values in response. Zhang et al. (2018b) constructed PERSONA-CHAT dataset and proposed to model persona information using memory networks. On PERSONA-CHAT dataset, Yavuz et al. (2019); Song et al. (2019) explored the effectiveness of copy mechanism and conditional variational encoder. Further researches are conducted to promote the consistency of personas, such as a generation network based on personas to guide knowledge selection (Lian et al., 2019), a transmitter-receiver framework to explicitly model the understanding between speakers (Liu et al., 2020), and a multi-stage dialogue response generation framework to delete the words that may lead to inconsistency in the response, then rewrite it (Song et al., 2020a) on this basis. Majumder et al. (2020) adopted common sense databases and interpretation resources to expand persona information. Although these models have achieved promising results, they all have limitations when the persona information is incomprehensible, and incapable to learn persona

[2]https://github.com/NEU-DataMining/HLA-ChatPlusPlus

consistent expression effectively.

2.2 Multi-party Dialogue Generation

Existing methods of building dialogue systems can be generally categorized into studying two-party conversations and multi-party conversations. However, the task scenario of the multi-party dialogue system is closer to that in real life. In addition to predicting response, selecting the addressee of an utterance is also an important task for MPC. Ouchi and Tsuboi (2016) first proposed the task of addressee and response selection, and Zhang et al. (2018a) have validated the effectiveness of jointly modeling addressee and response selection. Le et al. (2019) proposed the who to whom (W2W) model to solve the problem of missing and completing addressee in a dialogue history. Tan et al. (2019) proposed the Context-Aware Thread Detection (CATD) to address the consistency of context and input messages. For response generation, (Hu et al., 2019) firstly tried to use a graph to model multi-party dialogue history, and effectively used the dialogue structure information. Liu et al. (2019) proposed interlocutor aware contexts into recurrent encoder-decoder (ICRED) frameworks model, which used three role GRUs to update the speaker vector, and then used the speaker vector and addressee information to generate a response. Wang et al. (2020) proposed to select responses accurately based on tracking dynamic topic. Gururangan et al. (2020); Gu et al. (2021b) adopted a multi-task learning method in MPC, which proves the effectiveness of incorporating domain knowledge. However, previous MPC researches have not studied persona information of the speakers.

3 Dataset Construction

Film and television drama scripts are a common dataset source for dialogue system research, where there are high-quality personas, distinctive characters, and many rounds of dialogues. We collects 30 English scripts from the website[3] to construct a multi-party dialogue dataset. Inspired by Li et al. (2020), we employ the tropes on TVTRopes as the personas of the characters, which are annotated by the audiences with more representative and distinctive meanings.

For the crawled script web pages, the main preprocessing steps are as follows: (i) Use regular expressions to filter out HTML escape characters and non-dialogue contents such as scenes, narration, and background of the script, and then generate the original dialogue dataset according to each scene of the script; (ii) Split and screen out the speakers in the script to build a multi-party dialogue dataset; (iii) Associate the characters in the dataset with the tropes in TVTRopes as personas. The supporting characters with no persona information are deleted. We collect the *Laconic* and *Main* user generated interpretation of tropes on TVTRopes Namespce[4] as sentence-level and document-level text knowledge. Sentence-level (*Laconic*) knowledge briefly explains the persona tags in one sentence, and document-level (*Main*) knowledge further explains the persona tags in detail through examples, as shown in Appendix A.

Finally, we construct a new multi-party dialogue dataset, HLA-Chat++, which has 823,204 conversations with character persona annotations. According to the statistics, HLA-Chat++ has 239 characters in the dataset, with an average of 3,444 dialogues per character, and an average of 27,440 dialogues per TV dramas. Table 2 shows the example TV dramas with data size and the number of main characters. Only 8 TV dramas are selected due to the space limitation.

TV dramas	size	characters
Alias	19,312	9
Bones	42,952	8
Charmed	25,572	8
Friends	23,520	7
GilmoreGirls	105,303	19
Merlin	13,822	6
NCIS	33,900	9
QueerAsFolk	14,123	7

Table 2: The multi-party personalized dialogue dataset

	HLA-Chat++	HLA-Chat
number of dramas	30	38
data size	823,204	1,042,647
persona source	TVTropes	TVTropes
persona representation	persona tags	persona sentences
number of sentence-level knowledge	4,778	-
number of document-level knowledge	4,778	-
average length of sentence-level knowledge	7.5	-
average length of document-level knowledge	487.7	-

Table 3: The multi-party personalized dialogue dataset

[3] http://transcripts.foreverdreaming.org/

[4] https://tvtropes.org/pmwiki/index_report.php

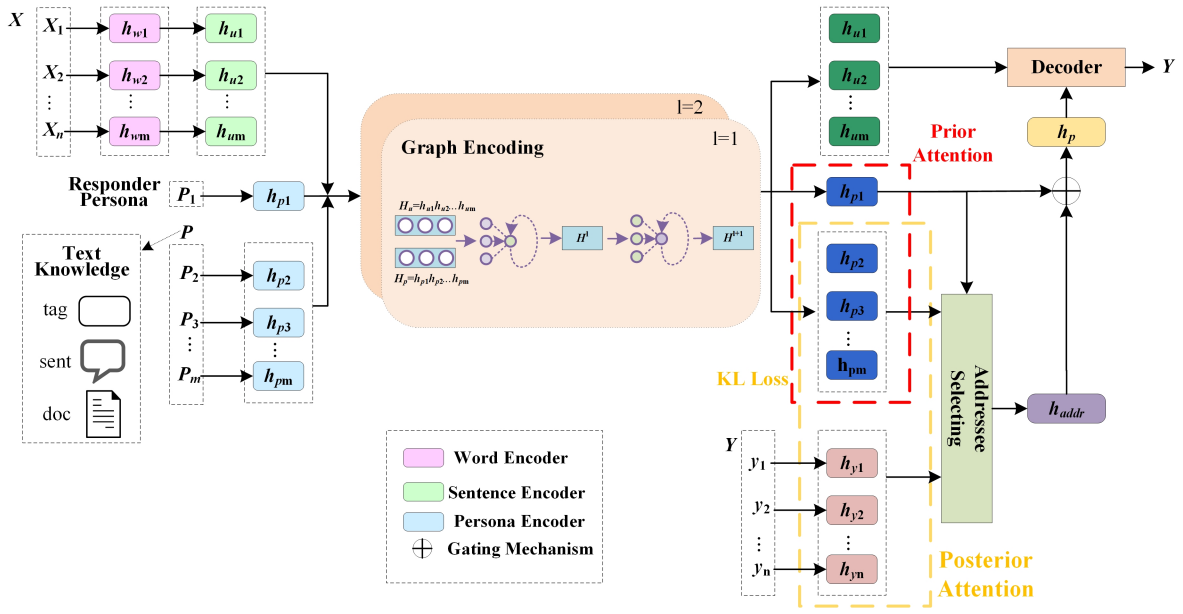

Figure 1: Framework of PersonaTKG

Compared with the original HLA-Chat, HLA-Chat++ has a smaller number of dialogues, as shown in Table 3. Li et al. (2020) processed the persona tags in HLA-Chat into descriptive sentences in the first person, just following the format in Zhang et al. (2018b). On the contrary, we retrain the persona tags, and collect the corresponding sentence-level and document-level interpretation of the tags, which can enrich the semantic meanings of these incomprehensible personas. Besides, HLA-Chat++ pays more emphasis on the multi-party dialogue structure, and preserves the relevant information in the dataset.

4 Model

PersonaTKG is shown in Figure 1, which is based on a Seq2Seq structure. The task can be formally defined as: given context $X = \{X_1, X_2, \ldots, X_m\}$, where X_i denotes the utterance of the speaker, and persona set $P = \{P_1, P_2, \ldots, P_k\}$, where k denotes k speakers, P_1 is the personas of the responder and $P_i (i > 1)$ is the personas of other speakers. The goal is to generate response $Y = y_1 y_2 \ldots y_n$ based on context and persona set, where y_i denotes the word generated in each step.

4.1 Context and Persona Encoder

In order to fully capture the information in multi-turn context, we adopt a hierarchical encoding strategy. The utterance encoder consists of word-level and sentence-level encoders, both of which are single-layer bidirectional GRU. The original data of the utterance encoder is X_i, the word-level encoder is responsible for encoding the utterances into vectorized representation $H_w = \{h_{w1}, h_{w2}, \ldots, h_{wm}\}$, which is calculated again by the sentence-level encoder, then the output of forward GRU and backward GRU are concatenated as the representation of X_i, namely $H_u = \{h_{u1}, h_{u2}, \ldots, h_{um}\}$, each h_{ui} contains the text information of X_i and the information immediately before and after X_i.

The original data of the persona encoder is the persona P_i of the speaker, which is encoded by single-layer bidirectional GRU to obtain the representation $H_p = \{h_{p1}, h_{p2}, \ldots, h_{pk}\}$. The representation of personas of the speakers is mainly used for subsequent addressee selection, considering that one can not accurately judge the addressee if it only contains persona information. If the personas of the speakers embed contextual information, it may be helpful for addressee prediction. Therefore, a graph convolution network is needed to further encode the representation of utterances and personas to aggregate information.

4.2 Graph Construction

The graph is represented as $\mathcal{G} = (\mathcal{V}, \mathcal{E})$, which is contracted from the utterances and persons in the following way.

Vertexes: including utterance nodes and persona nodes, and then utterance and persona in the dialogue are concatenated to represent a vertex $v_i \in \mathcal{V}$

in $\mathcal{G}$, and each vertex v_i is initialized with the cooresponding sequentially encoded feature vector $H = \{h_{u1}, h_{u2}, \ldots, h_{um}, h_{p1}, h_{p2}, \ldots, h_{pk}\}$. We denote this vector as the vertex feature, which will change according to the utterances and personas of different speakers.

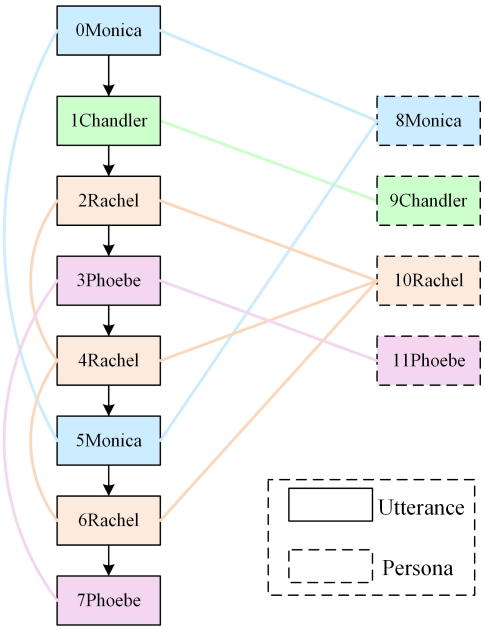

Figure 2: An example of edges construction. Edges of different speakers are marked with different colors. Edges with arrows are one-way sides, and those without arrows have sides in both directions.

Edges: To more fully model the context, we establish three kinds of relationships between the utterances and personas: (i) There is an edge between two adjacent utterances, which points from the one in front of time to the one in back, representing the relationship of time; (ii) There is an edge between the persona of the speaker and all the utterances that belong to the persona of the speaker; (iii) There is an edge between utterances that belong to the same speaker. The constructed edges are shown in Figure 2.

The specific way of building edges is to number all the utterance and persona nodes above starting from 0 and combine the number of the source and target nodes of an edge as the data form of an edge. For example, if utterance node 3 belongs to the persona of the speaker node 11, mark this edge as [3, 11]. In addition, each node is set to have an edge pointing to itself, which is to aggregate the information of neighbor nodes in the graph coding stage without losing the information of the node itself. According to our statistics, each dialogue graph has an average of 10.7 nodes and 24.6 edges.

The edge set $\mathcal{E}$ is initialed as an adjacency matrix and recorded as A, the GCN of each layer is calculated as follows:

$$H^{(l+1)} = \sigma\left(\tilde{D}^{-\frac{1}{2}}\tilde{A}\tilde{D}^{-\frac{1}{2}}H^{(l)}W^{(l)}\right) \quad (1)$$

where $\tilde{A} = A + I$ is the adjacency matrix with self connection, I is the unit matrix, $\tilde{D}_{ii} = \sum_j \tilde{A}_{ij}$, $W^{(l)}$ is the parameter of Layer l, $\sigma(\cdot)$ indicates the activation function, such as $\mathrm{ReLU}(\cdot) = \max(0, \cdot)$. Considering the small scale of the graph constructed by the dialogue, setting the number of layers of GCN to 2 can make each node effectively aggregate the information of adjacent nodes. Such sufficient information can predict the addressee more accurately in the subsequent process. Similarly, the utterance node also aggregates the persona information of the corresponding speaker, which will play an important role as a memory set in the decoding stage.

4.3 Addressee Selecting Mechanism

In multi-party dialogue, selecting the addressee is important to generate an appropriate response. Therefore, this paper incorporates the addressee selecting module. There is no addressee label in the dataset, and the existing methods usually can not supervise the learning of this module. Therefore, the end-to-end method is usually used to update the whole model with the final loss. It is noted that the addressee can be easily inferred from the ground truth response. Inspired by Lian et al. (2019), the ground truth can be regarded as a label to supervise the addressee selecting process in addition to calculating the loss as the standard answer and the generated response. Therefore, this paper adopts the method of a posterior selection and adds a KL divergence to the module as an additional loss during training.

First of all, both during the training stage and the testing stage, the standard process of calculating the addressee is the selection process based on the attention mechanism, which is called prior selection:

$$\begin{aligned} p_{\text{prior}} &= p\left(h_p = h_{pi} \mid h_{p1}\right) \\ &= \frac{\exp\left(h_{pi} \cdot h_{p1}\right)}{\sum_{j=2}^{k} \exp\left(h_{pj} \cdot h_{p1}\right)} \end{aligned} \quad (2)$$

where h_{p1} is the persona representation of the responder, as the query of prior selection attention,

and $h_{pi}(i > 1)$ is the representation of personas of other speakers in this set of conversations, here we adopt dot product attention and softmax for normalization. In the training stage, the ground truth can be used as a label to supervise the training addressee selecting module, that is, the representation of the response can be used as a query to calculate the attention weight with the personas representation of other speakers, which is called posterior selection:

$$\begin{aligned} p_{\text{posterior}} &= p\left(h_p = h_{pi} \mid h_y\right) \\ &= \frac{\exp\left(h_{pi} \cdot h_y\right)}{\sum_{j=2}^{k} \exp\left(h_{pj} \cdot h_y\right)} \end{aligned} \tag{3}$$

where h_y is the representation of the ground truth, as the query of posterior selection attention.

Obviously, the ideal situation is that even if there is no standard answer, the distribution of p_{prior} can be as close as possible to $p_{\text{posterior}}$. Therefore, in addition to negative log likelihood (NLL) loss, this paper also introduces KL divergence as an auxiliary loss other than NLL loss to measure the similarity between p_{prior} and $p_{\text{posterior}}$ .The formula of KL divergence is defined as follows:

$$\text{Loss}_{\text{KL}} = \sum_{i=2}^{k} p\left(h_p = h_{pi} \mid h_y\right) \log \text{K} \tag{4}$$

$$\text{K} = \frac{p\left(h_p = h_{pi} \mid h_y\right)}{p\left(h_p = h_{pi} \mid h_{p1}\right)} \tag{5}$$

In addition, the calculation formula of Loss_{NLL} is:

$$\text{Loss}_{\text{NLL}} = -\sum_{i=1}^{n} \log p\left(y_i \mid y < y_i; X; P\right) \tag{6}$$

where y_i is the word output in the current time step, n is the length of the response. The overall loss of PersonaTKG is:

$$\text{Loss} = \text{Loss}_{\text{NLL}} + \text{Loss}_{\text{KL}} \tag{7}$$

Ultimately, weighted sums of attention weight and personas representation of other speakers are used to obtain the predictive representation of the addressee. The formula is as follows:

$$h_{\text{addr}} = \sum_{i=2}^{k} \alpha_i h_{pi} \tag{8}$$

where i is accumulated from 2, α_i is the value of dimension i in $p_{\text{posterior}}$ (training stage) or p_{prior} (testing stage). Then the personas representation of the responder and the addressee are weighted and summed using the gating mechanism and sent to the decoder. The calculation process is performed as follows:

$$h_p = \alpha_p \cdot W_a h_{p1} + (1 - \alpha_p) \cdot W_b h_{\text{addr}} \tag{9}$$

$$\alpha_p = \sigma\left(V \cdot h_{p1}\right) \tag{10}$$

$$\sigma(x) = \text{Sigmoid}(x) = \frac{1}{1 + \exp(-x)} \tag{11}$$

where W_a, W_b and V are learnable parameters.

4.4 Response Decoder

The decoder is a language model that generates a response word by word based on the context and persona information. Let s_{t-1} be the hidden state of decoder and y_{t-1} be the embedding of word generated in the last time step, h_t is the semantic vector obtained by the weighted sum of attention calculation on the memory set $H = \{h_{u1}, h_{u2}, \ldots, h_{um}, h_{p1}, h_{p2}, \ldots, h_{pk}\}$ of the encoder. Then the output calculation process of GRU in the current time step is:

$$s_t = \text{GRU}\left(\left[y_{t-1}; h_t\right], s_{t-1}\right) \tag{12}$$

The word generated in current time step y_t is obtained after linear transformation and softmax normalization on hidden state s_t.

5 Experiments

5.1 Dataset

The experiments are conducted on our constructed dataset HLA-Chat++. The dataset is divided into train / valid / test set according to the proportion of 96%, 2% and 2%.

5.2 Baselines

We compared PersonaTKG with several strong baselines. To be fair, encoders of all models are implemented with HRED to handle multi-round contexts.

Seq2Seq: a Seq2Seq model with attention mechanism (Sutskever et al., 2014).

DialogueGCN: A dialogue emotion analysis algorithm with a graph neural network encoder (Ghosal et al., 2019). We implement the encoder and add a decoder for generation.

Model	PPL	BLEU-1/2%	Dist-1/2%	Emb E/A/G%	Per R/P/F1%	ACC
Seq2Seq	134.013	9.51/10.57	0.79/2.53	36.05/46.65/41.64	0.02/0.14/0.04	28.3
Per-Seq2Seq	125.889	9.74/10.61	0.97/3.07	36.35/46.74/42.59	0.02/0.12/0.03	28.9
DialogGCN	127.623	9.75/11.03	0.51/1.35	36.89/45.29/42.35	0.02/0.11/0.03	28.6
Per-DialogGCN	125.866	10.55/11.91	0.90/2.95	36.64/46.23/42.23	0.01/0.12/0.02	28.8
SIRNN	119.997	10.32/11.41	0.81/2.64	36.79/48.75/43.19	0.02/0.14/0.04	28.7
Per-SIRNN	120.565	10.75/11.54	0.78/2.43	36.26/48.70/42.67	0.02/0.15/0.05	28.8
PostKS	122.626	10.59/11.37	0.87/2.19	36.68/47.13/42.86	0.02/0.14/0.03	28.9
PersonaTKG+tag	117.063	11.74/12.70	0.85/2.79	36.98/50.09/43.61	0.03/0.18/0.04	29.4
PersonaTKG+doc	114.440	12.09/13.21	1.17/4.25	37.29/51.16/44.13	0.04/0.54/0.05	29.8
PersonaTKG+sent	**109.719**	**13.18/14.17**	**1.59/6.90**	**38.82/53.23/45.83**	**0.05/0.61/0.07**	**30.2**

Table 4: Automatic evaluation results

SIRNN: A multi-party dialogue model with addressee selecting mechanism (Zhang et al., 2018a). We implement the encoder and add a decoder for generation.

Per-: Persona encoder is added to the above three models for persona integration.

PostKS: A persona-based generative network with posterior selection mechanism to guide knowledge selection (Lian et al., 2019). It regards persona as knowledge.

Note that we do not adopt methods such as ALOHA (Li et al., 2020) as baselines, since our method is not built on a pre-trained language model. We leave this potential improvement to the future work.

5.3 Implementation Details

The dimension of word embedding is set to 300 initialized using GloVe (Pennington et al., 2014) pre-trained word vector and the vocabulary size is set to 50000. The word-level, sentence-level encoders in the hierarchical utterance encoder, and persona encoder are single-layer bidirectional GRUs with 800 hidden units and do not share parameters. The GCN is two-layered, and the number of input and output channels is set to 800. In order to facilitate calculation and avoid tedious dimension transformation, the nonlinear layer function after the convolution of the first layer graph is ReLU, and the Dropout Mechanism is used. The batch size is 80 and we use the Adam optimizer with an initial learning rate of 0.0005.

5.4 Automatic Evaluation

We use a variety of automatic evaluation metrics to comprehensively evaluate the performance of PersonaTKG from many aspects.

PPL: perplexity of the model, the smaller PPL score indicates a higher probability of the model producing a real response.

BLEU-1/2: the word-overlap scores of calculating unigrams and bigrams against the ground truth.

Dist-1/2: the proportions of distinct unigrams and bigrams in the generated responses.

Embedding-based metrics: Emb E calculates the semantic similarity between the generated response and the ground truth by averaging word embeddings. **Emb A** and **Emb G** calculate the semantic similarity between the generated response and the ground truth based on average and greedy matching, respectively.

ACC: the accuracy between the ground truth and generated response.

Per R/P/F1: the uni-gram Recall/Precision/F1 scores between the generated response and the persona set (Lian et al., 2019). Specifically, the set of non-stopwords in the generated response is represented by W_Y, and in predefined persona texts are represented by W_P, the calculation formulas of Per R and Per P are as follows:

$$\frac{|W_Y \cap W_P|}{|W_P|} \text{ and } \frac{|W_Y \cap W_P|}{|W_Y|} \tag{13}$$

and Per F1=2 (Per R $\cdot$ Per P) /(Per R + Per P).

Table 4 shows the results of the automatic evaluation, with the best results in bold. On various automatic evaluation metrics, PersonaTKG achieves the optimum compared to the baselines.

It was found that the SIRNN group increases the most, we conjecture that the addressee selecting mechanism of SIRNN plays an important role. By

	Fluency			Persona consistency			Semantic coherence		
	Win	Tie	Lose	Win	Tie	Lose	Win	Tie	Lose
PersonaTKG+sent vs PersonaTKG+tag	64	92	44	104	68	28	92	72	36
PersonaTKG+sent vs SIRNN	69	89	42	125	49	26	109	73	18

Table 5: Human evaluation results

Model	PPL	BLEU-1/2%	Dist-1/2%	Emb E/A/G%	Per R/P/F1%	ACC
PersonaTKG+sent	**109.719**	**13.18/14.17**	**1.59/6.90**	**38.82/53.23/45.83**	**0.05/0.61/0.07**	**30.2**
w/o GCN	118.502	10.70/11.86	1.22/3.35	36.75/49.13/43.47	0.02/0.40/0.04	28.9
w/o AS	117.977	11.15/12.53	1.06/2.07	37.15/50.27/43.48	0.02/0.34/0.04	29.1

Table 6: Ablation experiments

predicting the addressee in the encoding stage, a sensitive response to the addressee can be generated according to the given persona information. The improvement of PostKS relative to Seq2Seq proves the effectiveness of a posterior selection mechanism.

DialogGCN and PersonaTKG realize different graph encoding methods, which have a great improvement in BLEU compared with the Seq2Seq model and prove the effectiveness of graph structure encoding. Compared to DialogueGCN, PersonaTKG in the persona related metrics, word embedding metrics, accuracy, and other metrics get higher scores. It shows that considering the construction of various types of nodes in the context, we can extract more sufficient information than simply using the composition of sentence nodes and using the inherent connection between nodes as an edge is a more reasonable choice. The above analysis and experimental results show that PersonaTKG (PersonaTKG + tag) combined with addressee selecting mechanism and specific graph encoding achieves better results.

In addition, after adding two kinds of persona explanations, all metrics are improved compared with only the persona tag model, especially the Per P/R/F1, which show that the introduction of unstructured persona text knowledge can improve the persona consistency in response. However, the effect of adding document-level text knowledge is not good as adding sentence-level text knowledge. We conjecture that document-level text knowledge is too complex and noise is added. We also conducted the t-test on the models' performance. The results show that our PersonaTKG+sent model significantly outperforms the other baseline models with $p<0.01$.

5.5 Human Evaluation

To better evaluate the quality of the generated responses, we performed human evaluation. 200 generated responses were randomly sampled from the test set to 5 graduate students majoring in dialogue system, and compared with two relatively strong baselines in automatic evaluation from the following aspects:

Fluency: The generated response is smooth and free of syntax errors.

Persona consistency: The generated response is consistent with the persona of the speaker.

Semantic coherence: The generated response is semantically coherent to the context.

For each aspect, the evaluator can choose Win, Tie, and Lose. Win outperformed the other, Lose instead, and Tie represents both tied. Based on the results in Table 5, PersonaTKG significantly outperforms other baselines in all subjective evaluation metrics, indicating that PersonaTKG can extract useful information from the context with a wide range of information and complex structure.

5.6 Ablation Experiments

In order to investigate the effect of feature extraction of graph convolution network and the influence of addressee selecting mechanism on response generation, ablation experiments are carried out on these two modules.

w/o GCN: The GCN encoding process was removed.

w/o AS: The addressee selecting mechanism is removed and decoded directly with the persona of the responder.

The results are shown in Table 6. It can be observed that after removing the graph coding and ad-

context	(Doorbell rings. Sydney takes the pail of candy, and answers. Dixon, his wife Diane, and two kids in costumes stand there.) Kids: Trick or treat! Sydney: Hey, guys! There's more candy in there. Come on in! Diane: Good to see you. Sydney: Good to see you. Dixon: Sydney, can I talk to you for a second? Sydney: What's up? Dixon: Sydney, we've been working together a lot of years. I trust you.
Persona	Heroic BSOD They Do Too Dumb To Live Took ALevel In Badass Vitriolic Best Buds Ho Yay (Due to space constraints, only persona tags are given, sentence-level and document-level text knowledge are not given.)
Seq2Seq	i don't know.
Per-Seq2Seq	i don't know.
DialogGCN	i don't know.i don't know.
Per-DialogGCN	i don't know.
SIRNN	Sydney:what's meaning?
Per-SIRNN	Sydney:what?
PostKS	me too.
PersonaTKG+tag	Sydney: i know.
PersonaTKG+doc	Sydney: i know, i trust you, too.
PersonaTKG+sent	Sydney: i know, i trust you, too.

Table 7: Case study.

dressee selecting mechanism, all metrics decrease significantly. After removing the graph coding module, BLEU decreases significantly, indicating that graph coding is helpful for the fluency of generated responses. After removing the addressee selecting mechanism, Distinct decreases significantly, indicating that the addressee selecting mechanism has a great impact on the diversity of generated responses. In addition, we have conducted the t-test on the models' performance. The results show that our PersonaTKG+sent model significantly outperforms the ablated models with $p<0.01$.

5.7 Case Study

Table 7 shows an example of responses generated by different models along with the input message and persona set. It can be seen that simple models are difficult to model complex context, so Seq2Seq and DialogueGCN all generate a general reply *"i don't know."*. SIRNN with addressee selecting mechanism can predict the next speaker *Sydney*, Postks with a posterior mechanism can generate a contextual response *"me too."*.

PersonaTKG can not only predict the addressee but also generate a contextual response. After adding sentence-level and document-level text knowledge, the generated response *"i trust you, too."* is consistent with *Sydney's* persona *Vitriolic Best Buds*. *Sydeny* trust *Dixon* because she regards him as a deep down friend. It is proved that PersonaTKG improves the consistency between responses and personas of the speakers compared with other baselines.

6 Conclusion

In this paper, we first propose to use unstructured text knowledge in MPC to explain the incomprehensive persona tags. We construct a multi-party personalized dialogue dataset HLA-Chat++ based on English drama scripts and propose a model PersonaTKG with addressee selecting mechanism that integrates personas, dialogue utterances, and external text knowledge in a unified graph. The results show that the automatic and human evaluation are superior than other baselines, which demonstrate the effectiveness of our methods in improving persona consistency.

Acknowledgements

We would like to thank the reviewers for their constructive comments. This work was supported by the National Science Foundation of China (61872074, 62106039, 62272092).

References

Deepanway Ghosal, Navonil Majumder, Soujanya Poria, Niyati Chhaya, and Alexander Gelbukh. 2019. Dialoguegcn: A graph convolutional neural network for emotion recognition in conversation. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 154–164.

Jia-Chen Gu, Hui Liu, Zhen-Hua Ling, Quan Liu, Zhigang Chen, and Xiaodan Zhu. 2021a. Partner matters! an empirical study on fusing personas for personalized response selection in retrieval-based chatbots. In *Proceedings of the 44th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 565–574.

Jia-Chen Gu, Chongyang Tao, Zhenhua Ling, Can Xu, Xiubo Geng, and Daxin Jiang. 2021b. Mpc-bert: A pre-trained language model for multi-party conversation understanding. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (ACL-IJCNLP)*, pages 3682–3692.

Suchin Gururangan, Ana Marasović, Swabha Swayamdipta, Kyle Lo, Iz Beltagy, Doug Downey, and Noah A Smith. 2020. Don't stop pretraining: Adapt language models to domains and tasks. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics (ACL)*, pages 8342–8360.

Wenpeng Hu, Zhangming Chan, Bing Liu, Dongyan Zhao, Jinwen Ma, and Rui Yan. 2019. Gsn: A graph-structured network for multi-party dialogues. In *Proceedings of the 28th International Joint Conference on Artificial Intelligence (IJCAI)*, pages 5010–5016.

Satwik Kottur, Xiaoyu Wang, and Vitor R Carvalho. 2017. Exploring personalized neural conversational models. In *Proceedings of the 26th International Joint Conference on Artificial Intelligence (IJCAI)*, pages 3728–3734.

Ran Le, Wenpeng Hu, Mingyue Shang, Zhenjun You, Lidong Bing, Dongyan Zhao, and Rui Yan. 2019. Who is speaking to whom? learning to identify utterance addressee in multi-party conversations. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 1909–1919.

Aaron W Li, Veronica Jiang, Steven Y Feng, Julia Sprague, Wei Zhou, and Jesse Hoey. 2020. Aloha: Artificial learning of human attributes for dialogue agents. In *Proceedings of the AAAI Conference on Artificial Intelligence*, pages 8155–8163.

Jiwei Li, Michel Galley, Chris Brockett, Georgios Spithourakis, Jianfeng Gao, and William B Dolan. 2016. A persona-based neural conversation model. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (ACL)*, pages 994–1003.

Rongzhong Lian, Min Xie, Fan Wang, Jinhua Peng, and Hua Wu. 2019. Learning to select knowledge for response generation in dialog systems. In *Proceedings of the 28th International Joint Conference on Artificial Intelligence (IJCAI)*, pages 5081–5087.

Yunlong Liang, Fandong Meng, Ying Zhang, Yufeng Chen, Jinan Xu, and Jie Zhou. 2021. Infusing multi-source knowledge with heterogeneous graph neural network for emotional conversation generation. In *Proceedings of the AAAI Conference on Artificial Intelligence*, pages 13343–13352.

Cao Liu, Kang Liu, Shizhu He, Zaiqing Nie, and Jun Zhao. 2019. Incorporating interlocutor-aware context into response generation on multi-party chatbots. In *Proceedings of the 23rd Conference on Computational Natural Language Learning (CoNLL)*, pages 718–727.

Qian Liu, Yihong Chen, Bei Chen, Jian-Guang Lou, Zixuan Chen, Bin Zhou, and Dongmei Zhang. 2020. You impress me: Dialogue generation via mutual persona perception. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics (ACL)*, pages 1417–1427.

Bodhisattwa Prasad Majumder, Taylor Berg-Kirkpatrick, Julian J McAuley, and Harsh Jhamtani. 2021. Unsupervised enrichment of persona-grounded dialog with background stories. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (ACL-IJCNLP)*.

Bodhisattwa Prasad Majumder, Harsh Jhamtani, Taylor Berg-Kirkpatrick, and Julian McAuley. 2020. Like hiking? you probably enjoy nature: Persona-grounded dialog with commonsense expansions. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 9194–9206.

Zhao Meng, Lili Mou, and Zhi Jin. 2018. Towards neural speaker modeling in multi-party conversation: the task, dataset, and models. In *Proceedings of the AAAI Conference on Artificial Intelligence*, pages 8121–8122.

Biswesh Mohapatra, Gaurav Pandey, Danish Contractor, and Sachindra Joshi. 2021. Simulated chats for building dialog systems: Learning to generate conversations from instructions. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1190–1203.

Hiroki Ouchi and Yuta Tsuboi. 2016. Addressee and response selection for multi-party conversation. In *Proceedings of the 2016 Conference on Empirical*

Methods in Natural Language Processing (EMNLP), pages 2133–2143.

Zhai Penghua and Zhang Dingyi. 2019. Bidirectional-gru based on attention mechanism for aspect-level sentiment analysis. In *Proceedings of the 2019 11th International Conference on Machine Learning and Computing (ICMLC)*, pages 86–90.

Jeffrey Pennington, Richard Socher, and Christopher D Manning. 2014. Glove: Global vectors for word representation. In *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*, pages 1532–1543.

Qiao Qian, Minlie Huang, Haizhou Zhao, Jingfang Xu, and Xiaoyan Zhu. 2018. Assigning personality/profile to a chatting machine for coherent conversation generation. In *Proceedings of the 27th International Joint Conference on Artificial Intelligence (IJCAI)*, pages 4279–4285.

Maarten Sap, Ronan Le Bras, Emily Allaway, Chandra Bhagavatula, Nicholas Lourie, Hannah Rashkin, Brendan Roof, Noah A Smith, and Yejin Choi. 2019. Atomic: An atlas of machine commonsense for if-then reasoning. In *Proceedings of the AAAI Conference on Artificial Intelligence*, pages 3027–3035.

Iulian Serban, Alessandro Sordoni, Yoshua Bengio, Aaron Courville, and Joelle Pineau. 2016. Building end-to-end dialogue systems using generative hierarchical neural network models. In *Proceedings of the AAAI Conference on Artificial Intelligence*, pages 3776–3784.

Haoyu Song, Yan Wang, Weinan Zhang, Xiaojiang Liu, and Ting Liu. 2020a. Generate, delete and rewrite: A three-stage framework for improving persona consistency of dialogue generation. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics (ACL)*, pages 5821–5831.

Haoyu Song, Yan Wang, Weinan Zhang, Zhengyu Zhao, Ting Liu, and Xiaojiang Liu. 2020b. Profile consistency identification for open-domain dialogue agents. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 6651–6662.

Haoyu Song, Weinan Zhang, Yiming Cui, Dong Wang, and Ting Liu. 2019. Exploiting persona information for diverse generation of conversational responses. In *Proceedings of the 28th International Joint Conference on Artificial Intelligence (IJCAI)*, pages 5190–5196.

Robyn Speer, Joshua Chin, and Catherine Havasi. 2017. Conceptnet 5.5: an open multilingual graph of general knowledge. In *Proceedings of the AAAI Conference on Artificial Intelligence*, pages 4444–4451.

Ilya Sutskever, Oriol Vinyals, and Quoc V Le. 2014. Sequence to sequence learning with neural networks. In *Proceedings of the 27th International Conference on Neural Information Processing Systems (NIPS)*, pages 3104–3112.

Ming Tan, Dakuo Wang, Yupeng Gao, Haoyu Wang, Saloni Potdar, Xiaoxiao Guo, Shiyu Chang, and Mo Yu. 2019. Context-aware conversation thread detection in multi-party chat. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 6456–6461.

Weishi Wang, Steven CH Hoi, and Shafiq Joty. 2020. Response selection for multi-party conversations with dynamic topic tracking. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 6581–6591.

Chen Henry Wu, Yinhe Zheng, Xiaoxi Mao, and Minlie Huang. 2021. Transferable persona-grounded dialogues via grounded minimal edits. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 2368–2382.

Chen Xing, Yu Wu, Wei Wu, Yalou Huang, and Ming Zhou. 2018. Hierarchical recurrent attention network for response generation. In *Proceedings of the AAAI Conference on Artificial Intelligence*, pages 5610–5617.

Semih Yavuz, Abhinav Rastogi, Guan-Lin Chao, and Dilek Hakkani-Tur. 2019. Deepcopy: Grounded response generation with hierarchical pointer networks. In *Proceedings of the 20th Annual SIGdial Meeting on Discourse and Dialogue*, pages 122–132.

Rui Zhang, Honglak Lee, Lazaros Polymenakos, and Dragomir Radev. 2018a. Addressee and response selection in multi-party conversations with speaker interaction rnns. In *Proceedings of the AAAI Conference on Artificial Intelligence*, page 5690–5697.

Saizheng Zhang, Emily Dinan, Jack Urbanek, Arthur Szlam, Douwe Kiela, and Jason Weston. 2018b. Personalizing dialogue agents: I have a dog, do you have pets too? In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (ACL)*, pages 2204–2213.

Yinhe Zheng, Rongsheng Zhang, Minlie Huang, and Xiaoxi Mao. 2020. A pre-training based personalized dialogue generation model with persona-sparse data. In *Proceedings of the AAAI Conference on Artificial Intelligence*, pages 9693–9700.

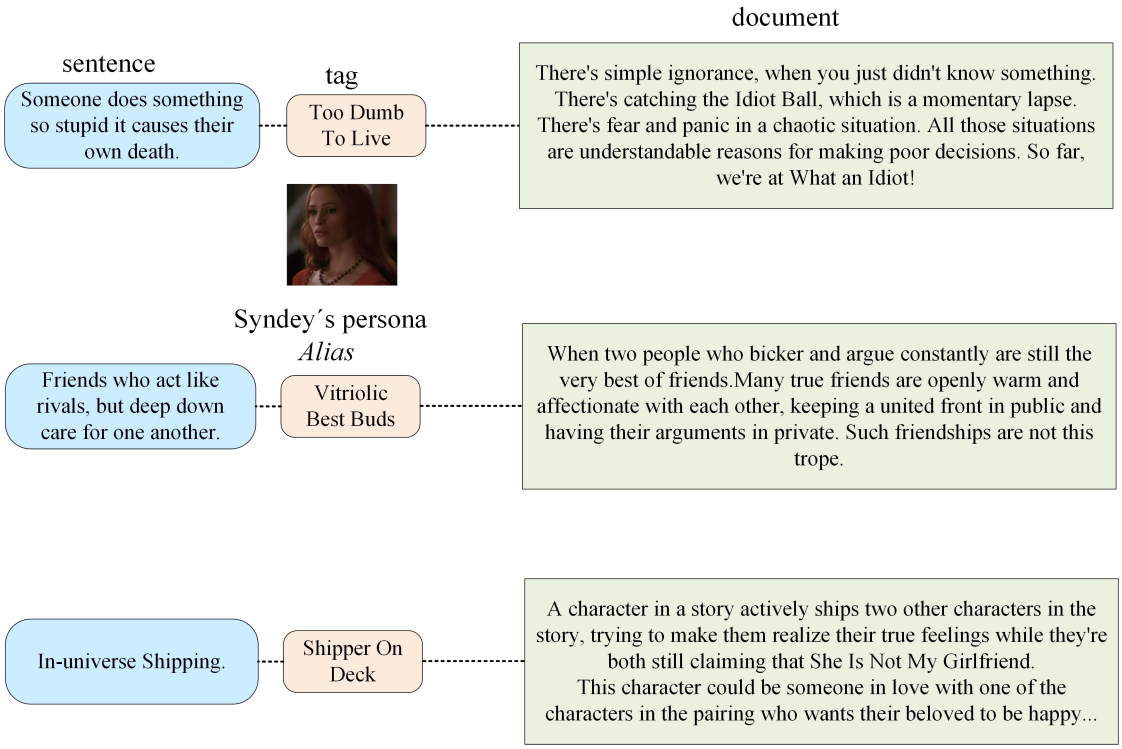

Figure 3: Three personas of Syndey in Alias

A Persona Explanation for Visualization

As shown in in Figure 3, three persona tags *Too Dumb To Live*, *Vitriolic Best Buds* and *Shipper On Deck* of Syndey in *Alias* are given, and explain them on sentence level and document level.

Improving Top-K Decoding for Non-Autoregressive Semantic Parsing via Intent Conditioning

Geunseob (GS) Oh, Rahul Goel, Chris Hidey,
Shachi Paul, Aditya Gupta, Pararth Shah, Rushin Shah
Google
{ohgs,goelrahul}@google.com

Abstract

Semantic parsing (SP) is a core component of modern virtual assistants like Google Assistant and Amazon Alexa. While sequence-to-sequence based auto-regressive (AR) approaches are common for conversational SP, recent studies (Shrivastava et al., 2021) employ non-autoregressive (NAR) decoders and reduce inference latency while maintaining competitive parsing quality. However, a major drawback of NAR decoders is the difficulty of generating top-*k* (i.e., *k*-best) outputs with approaches such as beam search. To address this challenge, we propose a novel NAR semantic parser that introduces intent conditioning on the decoder. Inspired by the traditional intent and slot tagging parsers, we decouple the top-level intent prediction from the rest of a parse. As the top-level intent largely governs the syntax and semantics of a parse, the intent conditioning allows the model to better control beam search and improves the quality and diversity of top-*k* outputs. We introduce a hybrid teacher-forcing approach to avoid training and inference mismatch. We evaluate the proposed NAR on conversational SP datasets, TOP & TOPv2. Like the existing NAR models, we maintain the $O(1)$ decoding time complexity while generating more diverse outputs and improving top-3 exact match (EM) by 2.4 points. In comparison with AR models, our model speeds up beam search inference by 6.7 times on CPU with competitive top-*k* EM.

1 Introduction

Neural sequence models are widely used for the task of conversational semantic parsing, which converts natural language utterances to machine-understandable meaning representations. Recent approaches (Chen et al., 2020; Rongali et al., 2020; Lialin et al., 2020; Aghajanyan et al., 2020a; Shrivastava et al., 2021) combine Transformer-based sequence models (Vaswani et al., 2017; Devlin et al., 2019) and Pointer Generator Networks (Vinyals et al., 2015; See et al., 2017). A vast majority of semantic parsing approaches (Gupta et al., 2018; Chen et al., 2020; Rongali et al., 2020; Lialin et al., 2020; Aghajanyan et al., 2020a; Yin et al., 2021) employ autoregressive (AR) decoders to generate structured output frames for the quality of output parses. During the AR decoding, output tokens are generated sequentially, conditioned on all previously generated tokens. As a result, the decoding time (i.e. inference latency) increases linearly with the decoding length. This not only limits the capacity to accommodate larger language models but may also degrade the user experience of intelligent conversational assistants (e.g. Google Assistant, Amazon Alexa) due to the high inference latency.

Query:	what is going on this weekend?
Label:	[in:get_event [sl:date_time this weekend]]
(Failure 1: repeated tokens)	[in:get_event [sl:date_time this weekend] [sl:date_time this weekend]] (Parse with the repeated slot *sl:data_time* and leaf nodes *this* and *weekend*)
(Failure 2: invalid syntax)	[in:get_event [sl:date_time (Incomplete parse; missing ']]')

Table 1: Examples of possible failures from the limited beam search of the existing NAR semantic parsers.

On the contrary, non-autoregressive (NAR) decoders are capable of parallel decoding and thus allow much faster inference. Compared to AR decoders, which perform O(n) decoding steps, the NAR decoding is typically achieved in either $O(1)$ (Babu et al., 2021; Shrivastava et al., 2021) or O($\log(n)$) (Zhu et al., 2020) steps. A recent study (Babu et al., 2021) built NAR parsers and reduced the latency up to 81% compared to AR parsers. Another study by Shrivastava et al. (2021) pro-

Proceedings of the 29th International Conference on Computational Linguistics, pages 310–322
October 12–17, 2022.

posed a NAR semantic parser called Span Pointer Network and cut the latency by 8.5-10x on CPU while achieving comparable performance to their AR benchmarks. All these studies suggest that the NAR decoders have significant latency benefits.

Although the recent NAR semantic parsers have decreased the performance gap from AR parsers on the exact match metric, one of the main limitations of NAR models still remain unsolved: beam search and top-k outputs. AR models leverage beam search algorithms (Wiseman and Rush, 2016), which are especially effective at producing top-k outputs (Wiseman and Rush, 2016; Gupta et al., 2018). This is possible since the beam search allows AR models to dynamically sort out less-probable candidate parses and adjust their decoding lengths. As a result, AR models are capable of generating diverse high-quality output parses.

On the other hand, the existing NAR parsers (Babu et al., 2021; Shrivastava et al., 2021) cannot employ the AR beam search as the output tokens of the NAR parsers are generated independently from each other. Instead, the existing NAR parsers perform a beam search by generating k candidate frame lengths. For each frame length, a single parse is produced, resulting in a total of k parses.

However, we find that the existing NAR beam search tends to produce duplicates of the most probable output parse. As exemplified in Table 1, top-k beam outputs often include outputs with repeated tokens and parses with invalid syntax (e.g., truncated or extended versions of the most-probable parse) rather than diverse output parses.

The ability to generate diverse top-k parses is important for modern conversational assistants. Table 2 presents an example that demonstrates how a query may correspond to different parses depending on the context. The diverse semantic parses can be leveraged in a downstream component that has more contextual information. An additional re-ranking module can also be employed to select the most relevant semantic parse.

This work focuses on improving the quality of top-k outputs of NAR conversational semantic parsing. Our idea is to leverage the fact that the top-level intent of a semantic parse mainly determines the syntax and semantics of the parse. This is the key driving point of the classic intent classification and slot tagging (Liu and Lane, 2016) based parsers, which were widely used traditional semantic parsers. We build our model upon the existing NAR (i.e., frame length conditioned NAR), but decouple the prediction of the top-level intent from the output. We leverage the conditional dependencies of the frame length and parse on the top-level intent for diverse top-k outputs.

Query: avoid bridges on my route
Parses from the Proposed NAR
(1a) [in:update_directions [sl:path_avoid [in:get _location [sl:category_location bridges]]]]
(2a) [in:get_directions [sl:path_avoid [in:get _location [sl:category_location bridges]]]]
Parses from the Baseline NAR
(1b) [in:update_directions [sl:path_avoid [in:get _location [sl:category_location bridges]]]]
(2b) [in:update_directions [sl:path_avoid [in:get _location [sl:category_location bridges

Table 2: Examples of beam outputs. The parses produced by the proposed NAR are both valid depending on the context. If the query was made after the user already had set a path, parse 1a is more relevant, else 2a. In contrast, the baseline NAR only produced one valid parse. Parse 2b is an invalid duplicate of parse 1b.

We evaluate the proposed NAR model on two datasets: TOP (Gupta et al., 2018) and TOPv2 (Chen et al., 2020) and demonstrate that we further close the gap between AR and NAR parsers. Most importantly, we show that the intent conditioning improves the quality and diversity of the top-k parses and the total inference time of our NAR beam search is 6.7x times shorter than the AR baseline on CPU.

2 Related Work

2.1 Non-autoregressive Semantic Parsing

Non-autoregressive sequence models have been an active research area across different fields of natural language processing for their fast inference speeds. While various designs of NAR decoders exist, recent works in machine translation utilize models with iterative refinements (Lee et al., 2018; Ghazvininejad et al., 2019, 2020), insertion-based models (Stern et al., 2019), and latent alignment methods (Libovický and Helcl, 2018; Chan et al., 2020; Saharia et al., 2020).

The task of semantic parsing (SP) is similar to the machine translation task as they both translate input sentences from one representation to another. For this reason, recent studies in SP have

adopted modeling techniques from machine translation. Zhu et al. (2020) leveraged the insertion-based seq2seq models for NAR semantic parsing and reduced the decoding time from $O(n)$ to $O(log(n))$ while matching the performance of AR models. Babu et al. (2021); Shrivastava et al. (2021) applied an iterative refinement method *Mask-Predict* to semantic parsing and brought the time complexity further down to $O(1)$ while performing comparably to the baseline AR models. As opposed to the original Mask-Predict model that performs multiple iterations of re-masking and prediction (Ghazvininejad et al., 2019), they only perform a single iteration of the masking and output prediction as they claim that task-oriented SP does not benefit much from iterative refinements (Babu et al., 2021; Shrivastava et al., 2021). In addition to the recent sequence-to-sequence NAR semantic parsers, traditional intent and slot filling model (Mesnil et al., 2014) is another approach with non-autoregressive decoding.

2.2 Baseline NAR

Span Pointer Network (Shrivastava et al., 2021), which is one of the two recent works that leveraged the mask-predict algorithm, showed the competitive performance to the AR parsers while significantly reducing their latency. For this reason, we utilize Span Pointer Network as a basis of our work and accordingly refer to it as *baseline NAR* to distinguish the NAR semantic parser we propose. It should be noted that we replicated Span Pointer Network as it is not publicly available.

The baseline NAR is built with a pre-trained encoder, a frame length module, and a Transformer decoder. The encoder is a language model such as BERT (Devlin et al., 2019) or RoBERTa (Liu et al., 2019) that takes input queries x and outputs their encoded representations e.

$$e_{1:l} = \text{Encoder}(x_{1:l}), \tag{1}$$

where l indicates the length of the source (i.e., input query). The frame length module takes the encoded representations and predicts the length of the frame (i.e., target parse) n followed by the generation of n mask tokens.

$$\begin{aligned} n &= \text{FrameLengthModule}(e_{1:l}). \\ [\text{MASK}]_{1:n} &= \text{MaskCreation}(n). \end{aligned} \tag{2}$$

The transformer decoder takes the N mask tokens as inputs and produce h for each token in the masked sequence.

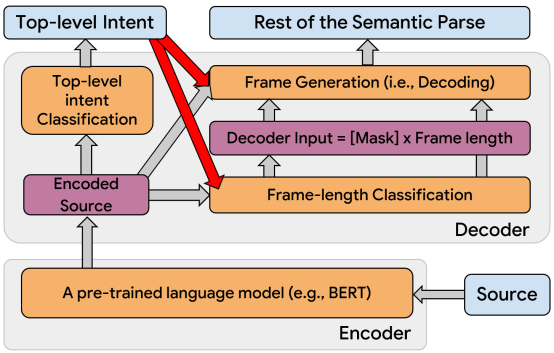

Figure 1: The proposed model architecture. By decoupling the top-level intent prediction from the output prediction, we perform conditional generation of the frame length and parse upon the top-level intent.

$$h_{1:n} = \text{Decoder}([\text{MASK}]_{1:n}; e_{1:l}). \tag{3}$$

Lastly, the pointer-generator mapping layer is used to convert h to the target frame y as follows.

$$y_{1:n} = \text{PTR}(h_{1:n}; e_{1:l}), \tag{4}$$

where y_k is either a token from the target vocab that consists of parse symbols (e.g., intents & slots) or copy of a source token. The output tokens are generated in parallel as they are conditionally independent given the source and frame length.

2.3 Beam search for AR vs baseline NAR

AR decoders produce multiple candidate output parses per source by employing the beam search algorithm (Wiseman and Rush, 2016). During the beam search with width k, the k most probable candidates are kept at each decoding step and less probable candidates are sorted out. The decoding process finishes when a special "[END]" token is encountered. For a decoding length n, the score of an AR parse is computed as follows.

$$S_{AR}(y) = p(y|x) = \prod_{j=1}^{n} p(y_j|y_{1:j-1}, x). \tag{5}$$

Since the AR decoders consider multiple candidate tokens y_j at each decoding step, they can generate a large number of unique parses.

In comparison, the baseline NAR only produces a single output parse per frame length. The baseline NAR mimics the beam search by generating top-k frame lengths, which results in k outputs. The output parse is scored using the joint probability of the frame length and output parse as follows.

$$S_{NAR,baseline}(y) = p(y, n|x) = p(y|n, x) \cdot p(n|x) = \prod_{j=1}^{n} p(y_j|n, x) \cdot p(n|x), \quad (6)$$

where $S_{NAR,baseline}(y)$ denotes the score of the baseline NAR output. $p(y|n, x)$ is obtained using the conditional independence of the output tokens given the frame length n and source x.

3 Proposed Approach

To mitigate the aforementioned limitation of the baseline NAR, we propose Intent-conditioned Non-autoregressive Neural Semantic Parser. Our approach is motivated by traditional semantic parsers such as Liu and Lane (2016), which performs an intent classification followed by slot tagging. Another motivation comes from an observation that the top-level intent of the parse largely governs the syntax and semantics of the rest of the parse.

Our idea is to decouple the top-level intent prediction from the output prediction as shown in Figure 1. This builds explicit dependencies of the frame length and the rest of the frame on the top-level intent. As a result, the joint probability of output and frame length is expressed as follows.

$$p(y, n|x) = p(y_{2:n}|n, y_1, x) \cdot p(n|y_1, x) \cdot p(y_1|x). \quad (7)$$

This facilitates effective conditional generation of the length and output, while keeping the decoding time complexity as $O(1)$; the number of decoding steps does not scale with the output length.

3.1 Intent Conditioning

The proposed NAR utilizes a pre-trained language model to obtain encoded source representation e. Then, it performs the top-level intent prediction, which is formulated as a multiclass classification of the size of the dimensionality of intent vocab.

$$\text{logits}(y_1) = \text{IntentModule}(e_{1:l}), \quad (8)$$

where y_1 refers to the top-level intent of the semantic parse or the first token of the parse. As the top-level intent prediction is decoupled, the intent module works with a smaller vocabulary size. Compared to the output vocabulary that combines a target and copy index vocabulary, the intent vocabulary is approximately 4 times smaller with a source length 32 on TOP dataset. We use a randomly initialized Transformer as the intent module.

We use the logits of the top-level intent, as opposed to the discrete intent class variable, as the inputs to all subsequent modules. This is because the dense logits convey information about the uncertainty of the intent prediction for each intent class and help the subsequent modules to condition on richer information. We found that this resulted in better performance than the model conditioned on the 1-dimensional discrete intent.

Subsequent to the top-level intent prediction, the frame length module takes the logits of the intent y_1 and produces the decoding length followed by the creation of mask tokens. After that, the initial mask tokens together with the frame length pass through a positional encoding layer to compute the encoded mask tokens, $[\text{MASK}]_{2:n}$.

$$n - 1 = \text{FrameLengthModule}(\text{logits}(y_1); e_{1:l}),$$
$$[\text{MASK}]_{2:n} = \text{PosEncoding}(\text{MaskCreation}, n). \quad (9)$$

The rest of the frame is generated using parallel decoding followed by the source token mapping via the pointer-generator mapping layer.

$$h_{2:n} = \text{Decoder}([\text{MASK}]_{2:n}; \text{logits}(y_1), e_{1:l}),$$
$$y_{2:n} = \text{PTR}(h_{2:n}; e_{1:l}), \;\; y_{1:n} = \text{cat}(y_1, y_{2:n}). \quad (10)$$

Finally, the output parse is obtained by concatenating the top-level intent and rest of the frame.

3.2 Training Objective of the Proposed NAR

The loss is a weighted sum of top-level intent classification loss $\mathcal{L}_{int}$, frame length classification loss $\mathcal{L}_{len}$, and output loss $\mathcal{L}_{out}$ as follows.

$$\mathcal{L} = \mathcal{L}_{out} + \lambda_{len} \cdot \mathcal{L}_{len} + \lambda_{int} \cdot \mathcal{L}_{int}. \quad (11)$$

We jointly optimize the three loss terms and employ label smoothing (LS) (Szegedy et al., 2016) as a regularizer to penalize overconfident predictions by computing negative log-likelihood (NLL) between the smoothed one-hot labels and predictions. That is, $\mathcal{L}_{int} = \text{NLL}(\text{LS}(y_{1,label}), y_{1,pred})$, $\mathcal{L}_{len} = \text{NLL}(\text{LS}(n_{label}), n_{pred})$, and $\mathcal{L}_{out} = \text{NLL}(\text{LS}(y_{2:n,label}), y_{2:n,pred})$.

3.3 Beam Search with the Proposed NAR

The proposed intent-conditioned NAR produces multiple output parses per source via the sequential conditioning on the top-level intent and length. Precisely, the model first computes top-$k1$ top-level intents. For each top-level intent, the length model

outputs top-$k2$ frame lengths. As a result, $k1 \cdot k2$ pairs of top-level intents and lengths are produced. Finally, the decoder generates a single parse per pair, yielding a total of $k1 \cdot k2$ output parses.

The proposed NAR performs a single decoder pass regardless of the beam size, $k1 \cdot k2$. This is possible as we cast the $k1 \cdot k2$ pairs as the batch dimension. As a result, the time complexity of the beam search remains constant, to the extent of the memory capacity. The memory increases linearly to the beam size.

The intent conditioning provides explicit control of the top-level intents via the selection of top-$k1$ unique intents. It also builds dependencies of the frame length and output on the top-level intents. We find that these help the model to improve diversity and quality of top-k output parses. To further validate the hypothesis, we devise two experiments to (1) identify the potential (i.e., upper limit of the accuracy) of the proposed NAR beam search and (2) examine the diversity and quality of the output parses compared to the baseline NAR beam search.

3.4 Hybrid Teacher Forcing

During our initial attempts to inspect the proposed model, we discovered that the inference performance is unstable. This was due to a discrepancy in the computation of the top-level intent during the training and inference. Recall that we designed the outputs of the top-level intent module to be logits (see Equation 8), as opposed to a one-dimensional or one-hot representation. The idea is to deliver richer information to downstream modules (e.g., length module and decoder). While we observed that this idea improves the exact match for the proposed NAR architecture, it makes the inference unstable. This was because we needed to sample the top-level intents, as (smoothed) one-hot vectors, and convert them back to the logits. Conversely, in the training, we do not sample top-level intents and instead use the direct outputs (dense logits) from the top-level intent module.

We first attempted to resolve the problem with teacher-forcing of the top-level intents. While this stabilized the inference, the training became unstable due to sparsity of the teacher logits and mismatch between the model and teacher intent logits.

We address this problem by leveraging a sampling strategy depicted in Figure 2. The idea is to uniformly sample logits from a pair of teacher and model logits as follows.

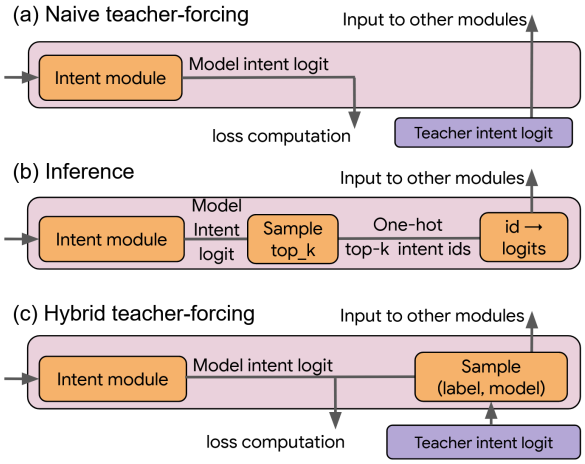

Figure 2: The computation of the top-level intent. (a) During training via naive teacher forcing, label intent logits are fed to the length module and decoder. (b) In the inference, top-k intents are sampled from the model logits and fed to the other modules. (c) Hybrid teacher-forcing uses both label and model logits for the training.

$$\text{logits}(y_1) \sim \mathcal{U}(\text{logits}(y_{1,\text{label}}), \text{logits}(y_{1,\text{model}})). \tag{12}$$

This essentially exposes the model, during the training, to the hidden space where the intents are sampled at the inference time. Without the hybrid teacher-forcing, the inference-time hidden space may not be well explored during the training, thus becomes source of instability.

We name this strategy as *hybrid teacher forcing* to reflect that the model leverages both teacher and model top-level intent logits. This sampling technique may be seen as a non-autoregressive variant of the scheduled sampling (Bengio et al., 2015). While the proposed model architecture is responsible for the performance improvement, the hybrid teacher-forcing technique helped our model to achieve the improvement by stabilizing the training and inference.

3.5 Scoring Beam Search Outputs

A scoring method is used to select the top-k parses from the $k1 \cdot k2$ pairs. We tested three methods. The first method S_1 uses the joint log probability of the top-level intent and frame length as the score.

$$S_1 = \log(p(n, y_1|x)) = \log(p(n|y_1, x) \cdot p(y_1|x)). \tag{13}$$

The second method S_2 uses the joint log probability of the output, length, and top-level intent.

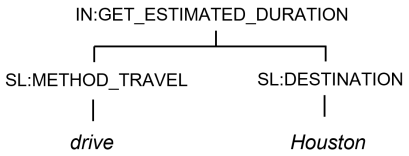

Figure 3: Decoupled representation of the semantic parse for a query "How long is my drive to Houston?".

$$S_2 = \log(p(y_{1:n}, n|x)) = \log\big(p(y_{2:n}|n, y_1, x) \cdot p(n|y_1, x) \cdot p(y_1|x)\big). \quad (14)$$

The last scoring method S_3 combines the joint probability $p(y_{1:n}, n|x)$ and a length penalty.

$$lp(y) = ((5 + l)/6)^{\alpha}, \quad S_3 = S_2(y)/lp(y), \quad (15)$$

where lp is the length penalty (Wu et al., 2016).

Depending on $k1$, $k2$, and scoring method, the top-k outputs may consist of parses with multiple distinct lengths and intents or parses with a single intent with multiple lengths.

3.6 Semantic Parse Representation

We represent semantic parses using the decoupled tree representation (Aghajanyan et al., 2020a) that is an extension of the compositional tree representation (Gupta et al., 2018). An example is depicted in Figure 3. Likewise, another query "What is happening in Boston on New Year's Eve" is semantically parsed into "*[in:get_event [sl:location Boston] [sl:date_time on New Year's Eve]]*".

An example of the decoupled tree representation is *Index form* that replaces the copy tokens with the indices to the corresponding source tokens. In this sense, the above semantic parse is represented as "*[in:get_event [sl:location 4] [sl:date_time 5 6 7 8]]*". Together with the pointer network, this significantly reduces the size of output vocabulary (Rongali et al., 2020). Another representation is *Span form* introduced in (Shrivastava et al., 2021). As opposed to specifying all indices of spans of copy tokens, the span form uses the start and end indices of the spans. That is, "*[in:get_event [sl:location 4 4] [sl:date_time 5 8]]*". The benefits of the span form include shorter target lengths and fixed number of leaf nodes that helps decoupling the syntax from the semantics. In addition, the number of valid frame length classes are halved as the frame lengths of parses in the span form are always even.

4 Experiments

4.1 Datasets

To quantify the benefits of our NAR parsers over the baseline NAR, we utilize two conversational semantic parsing datasets. The first dataset is TOP (Task Oriented Parsing) (Gupta et al., 2018), a collection of human-generated queries in English and the corresponding semantic parses that are represented as hierarchical trees. Another public dataset we explore is TOPv2 (Chen et al., 2020), which is an extension of TOP with six more domains.

4.2 Evaluation Metrics

We mainly utilize *exact match* (EM) to evaluate quality of both greedy decoding and beam decoding output. EM is defined as the percentage of queries whose label parses are correctly predicted.

Inference time is the metric we use to quantify the latency benefit of our model. We measure the model inference time on a TPU (Jouppi et al., 2017) from the Google Cloud TPUv2, a Nvidia Quadro P1000 GPU, and a CPU of the Intel Cascade Lake CPU platform with 8GB RAM. All measurements were obtained using the same machine on the same process. We used batch size 1 and computed latency on 1000 examples from the val set of the TOP dataset to reduce measurement noise and minimize the variance. We report per-example average latencies of these runs.

In addition, *diversity* of the output parses is computed to quantify the capability of the proposed NAR approach at producing distinct top-k outputs. We mainly compute two metrics: (1) number of unique top-level intents in top-k output parses and (2) the distinct n-grams (Li et al., 2016), which are commonly used diversity metrics in natural language processing (Vijayakumar et al., 2018; Xu et al., 2018). Specifically, we compute percentage of distinct n-grams by counting unique n-grams in top-k outputs, dividing the counts by the total number of output tokens, and scaling it by 100.

4.3 Implementation details

4.3.1 Encoder

We use pre-trained $\text{BERT}_{\text{BASE}}$ (L12/H768) as the encoders for all AR, baseline NAR, and proposed NAR for the experiments. We use uncased versions of them along with lowercased datasets as they generally performed better than the cased models with cased datasets. We leave it to future work to

quantify the performance with other pre-trained language models such as RoBERTa (Liu et al., 2019).

4.3.2 Decoder

We primarily compare the performances of two AR baseline decoders, the baseline NAR decoder, and the proposed NAR decoder. The two AR decoders are identical except that one is trained on the label parses represented in the index form and the other in the span form. All NAR decoders were trained on the label parses in the span form. We used the same decoder architecture for all models across all experiments. As discussed earlier, the baseline NAR is a replication of the NAR model proposed by Shrivastava et al. (2021). Specifically, we use the vanilla version that does not utilize the R3F loss (Aghajanyan et al., 2020b).

Unlike Shrivastava et al. (2021) and Babu et al. (2021) that use an MLP or CNN length module, our frame length module is a randomly initialized Transformer decoder; we found that it performs better as the attention layers perform more complex weighing on the encoder output. We used the same length module for both baseline and proposed NAR models. Similarly, we use a randomly initialized Transformer decoder for the top-level intent prediction. We point out that we use two separate Transformers, one for the top-level intent, the other for the frame length module, as opposed to a shared Transformer. Further details of the model parameters are described in the appendix.

4.3.3 Hyperparameters

We observed that the ratio of the loss terms is important. In general, higher top-level intent penalty and moderate frame length penalty resulted in better accuracy. We use the hybrid teacher forcing to train the proposed NAR for the experiments presented in this paper. We trained the models using Adam optimizer (Kingma and Ba, 2014) with exponential learning rate decay. Additionally, we use 1000 learning rate warmup steps. We also apply dropout to the source embeddings and the Transformer decoder for better generalization. The details of the parameters are described in the appendix.

5 Results

We show that the proposed intent-conditioned NAR further closes the gap between AR and NAR SP. Most importantly, our model greatly improves the quality of top-k output parses while maintaining the $O(1)$ decoding time of the baseline NAR.

Model	Latency			EM	
	TPU	GPU	CPU	TOPv1	TOPv2
Base.NAR	1.00x	1.00x	1.00x	82.56	84.86
Prop.NAR	1.01x	1.11x	1.09x	83.11	85.22
AR, index	2.23x	11.01x	4.75x	83.43	85.40
AR, span	2.26x	11.01x	4.58x	83.40	85.56

Table 3: Greedy decoding results with BERT encoder. Latency numbers are measured with TOPv1 dataset and relative to the baseline NAR latency on TPU.

We first present the greedy decoding results. We then investigate the upper-bound beam search performance of the baseline and proposed NAR. Next, we share the beam search results that quantify the quality and diversity of top-k output parses. Lastly, we share the results of the comparison of various beam scoring methods. We point out that we used the identical encoder & decoder architectures and model parameters for all AR & NAR models. In addition, the identical frame length module is used for both baseline and proposed NAR models.

5.1 Greedy Decoding

Table 3 presents the greedy decoding results. At each decoding step of AR models, the most probable token is selected. This repeats until an END token is produced. In NAR greedy decoding, beam width of 1 is used for both frame length and top-level intent. No scoring method is used.

The results indicate that our model improves the performance of baseline NAR by 0.4-0.5 EM while matching the latency of the prior NAR approach. Compared to AR baselines, our model cuts the greedy decoding inference time (i.e., latency of the semantic parser including both encoder and decoder) by 2.2x on TPU, 10x on GPU, and 4.2-4.4x on CPU.

5.2 Potential Impact of Intent Conditioning for Beam Search

We designed an experiment to investigate the potential benefits of the proposed beam search. We aimed to empirically quantify the upper bound of the beam search performance of the baseline NAR and our intent-conditioned NAR. In the experiment, we first trained the baseline and proposed NAR parsers, then fed the oracle (i.e., gold) frame length to the baseline NAR or the oracle top-level intent to the proposed NAR during the inference. To ensure

Models	Oracle EM
Variants of the baseline NAR	
Shallow decoder	83.34
Shallow decoder, Label smoothing	82.57
Deep decoder	83.73
Deep decoder, Label smoothing	83.03
Best of the baseline NAR	**83.73**
Variants of the proposed NAR	
Shallow decoder	86.22
Shallow decoder, Label smoothing	86.40
Deep decoder	86.89
Deep decoder, Label smoothing	85.66
Best of the proposed NAR	**86.89**

Table 4: Empirical upper-bound of beam search with NARs. To obtain the oracle EMs, we feed the gold frame length for the baseline NARs in the inference. Likewise, we use the gold top-level intent for the proposed NARs.

validity of the results, we ran the experiments with different model configurations.

As shown in Table 4, the proposed NAR consistently scored higher oracle EM across various model configurations. It should be noted that the proposed NARs only use the gold top-level intent (i.e., the gold length is not used for the inference with the proposed NARs). Table 5 confirms these results and shows that our NAR performs more effective beam-coding and achieves higher exact match for the top-*k* output.

5.3 Beam search

In table 5, we report the beam search results on TOP dataset as top-*k* EM for $k = 1, 2, 3$. If the label parse matches with any of the top-*k* output parses, it counts towards the top-*k* EM. We also report top-level intent match (IM) percents. The top-*k* parses are selected from the $k1 \cdot k2$ beam search outputs using the scoring method 3 (Equation 15). We used the same $k1 \cdot k2$ for the fair comparison. Specifically, we used $k2 = 25$ for the baseline NAR and $k1 = 25$, $k2 = 1$ for our NAR as there exist 25 distinct length classes and 25 distinct intent classes in TOP dataset. For the proposed NAR, we observed that a higher $k2$ has minor impact on top-*k* EMs, compared to a higher $k1$. We note that the hybrid teacher forcing (Equation 12) was used for the training and helped the models achieve consistent test set accuracy.

The results demonstrate a large gap between AR and the existing NAR models in top-*k* outputs. Notably, the top-3 EM from the existing NAR model is outperformed by the baseline AR by 3.5 points. Secondly, the proposed NAR outperforms baseline NAR in top-3 EM by 2.4 and top-3 IM by 2.8 points. Third, the EM and IM gaps between the proposed and baseline NAR widen as *k* increases. Lastly, we show that the proposed NAR reduces the performance gap from the autoregressive models. The inference time of our NAR beam search is 2.5x times shorter on TPU, 8.4x times shorter on GPU, and 6.7x times shorter on CPU. Our AR baselines manage parallel compute loads better than NAR models and thus result in smaller latency gaps on TPU.

Note that a parse can have more than 1 intent (e.g., the parses shown in Table 2). In this regard, we report the total intent prediction accuracy on top of the top-level intent match (IM) as follows. The intent prediction accuracy with top-1 prediction is 93.09 (baseline NAR), 93.25 (ours), 93.15 (AR, index form), 93.65 (AR, span form). The intent prediction accuracy with top-3 predictions are 94.09 (baseline NAR), 95.68 (ours), 95.17 (AR, index form), 95.33 (AR, span form). While baseline NAR only gained 1.00% intent accuracy improvement from top-1 to top-3, the proposed NAR gained 2.43% improvements. This agrees with the EM improvements that our model offers.

5.4 Diversity

Table 6 elaborates on the diversity of the output parses with the baseline and proposed NAR models. Specifically, we compute their average numbers of unique top-level intents and distinct n-gram (Li et al., 2016) percentages in top-3 output parses. We find that top-*k* parses of the proposed NAR outperforms the baseline NAR in all diversity metrics.

We observed that the baseline NAR beam search often duplicates the most probable parse with repeated tokens and/or invalid syntax. This suggests that the frame length conditioning alone is not effective at producing top-*k* outputs (see Table 1 and 2 for examples). Table 6 quantitatively confirms the observation with lower sentence-wise and corpus-wise distinct n-gram scores of the baseline NAR, compared to the proposed NAR.

Model	Latency			EM			IM		
	TPU	GPU	CPU	top-1	top-2	top-3	top-1	top-2	top-3
Baseline NAR	1x	1x	1x	82.61	83.44	83.62	94.59	95.14	95.22
Proposed NAR	1.12x	1.65x	1.25x	83.12	85.15	86.00	94.85	97.04	98.05
AR, index form	2.78x	13.92x	8.28x	83.44	86.46	87.34	94.61	95.95	96.44
AR, span form	2.84x	13.86x	8.33x	83.47	86.22	87.12	94.67	95.82	96.22

Table 5: Beam decoding results. The proposed NAR outperforms the baseline NAR in top-3 EM by 2.4 and top-3 IM by 2.8 points. Compared to AR models, our approach cuts a beam search inference latency by 6.7 times on CPU.

Metric (higher is more diverse)	Baseline NAR	Proposed NAR
top-3 Exact Match	83.62	86.00
# of unique 1st intents in top-3	1.12	3.00
Distinct 1-gram, sentence-wise	69.84	79.92
Distinct 2-gram, sentence-wise	85.93	94.12
Distinct 1-gram, corpus-wise	25.77	40.79
Distinct 2-gram, corpus-wise	40.36	53.77

Table 6: We measure the diversities of the top-3 parses of the baseline and proposed NAR models reported in Table 5. We report sentence-wise (i.e., distinct n-grams in each parse) and corpus-wise distinct n-grams (i.e., distinct n-grams in each collection of top-3 parses).

6 Conclusion

We present a novel non-autoregressive neural semantic parser: the intent-conditioned NAR for improved top-k decodings. The proposed model addresses the main limitation of the recent NAR models (i.e., the limited beam search capability) by decoupling the prediction of the top-level intent from output. This builds explicit dependencies of frame lengths and output parses on top-level intents and helps NAR semantic parsers to better control beam search. The proposed NAR model further closes the performance gap from AR models by improving the quality and diversity of top-k outputs. We highlight that the *decoding* time complexity is $O(1)$, regardless of output length or beam width.

References

Armen Aghajanyan, Jean Maillard, Akshat Shrivastava, Keith Diedrick, Michael Haeger, Haoran Li, Yashar Mehdad, Veselin Stoyanov, Anuj Kumar, Mike Lewis, and Sonal Gupta. 2020a. Conversational semantic parsing. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 5026–5035, Online. Association for Computational Linguistics.

Armen Aghajanyan, Akshat Shrivastava, Anchit Gupta, Naman Goyal, Luke Zettlemoyer, and Sonal Gupta. 2020b. Better fine-tuning by reducing representational collapse. In *International Conference on Learning Representations*.

Arun Babu, Akshat Shrivastava, Armen Aghajanyan, Ahmed Aly, Angela Fan, and Marjan Ghazvininejad. 2021. Non-autoregressive semantic parsing for compositional task-oriented dialog. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 2969–2978, Online. Association for Computational Linguistics.

Samy Bengio, Oriol Vinyals, Navdeep Jaitly, and Noam Shazeer. 2015. Scheduled sampling for sequence prediction with recurrent neural networks. *arXiv preprint arXiv:1506.03099*.

William Chan, Chitwan Saharia, Geoffrey Hinton, Mohammad Norouzi, and Navdeep Jaitly. 2020. Imputer: Sequence modelling via imputation and dynamic programming. In *International Conference on Machine Learning*, pages 1403–1413. PMLR.

Xilun Chen, Asish Ghoshal, Yashar Mehdad, Luke Zettlemoyer, and Sonal Gupta. 2020. Low-resource domain adaptation for compositional task-oriented semantic parsing. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 5090–5100, Online. Association for Computational Linguistics.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, Minneapolis, Minnesota. Association for Computational Linguistics.

Marjan Ghazvininejad, Omer Levy, Yinhan Liu, and Luke Zettlemoyer. 2019. Mask-predict: Parallel decoding of conditional masked language models. In

Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP), pages 6112–6121, Hong Kong, China. Association for Computational Linguistics.

Marjan Ghazvininejad, Omer Levy, and Luke Zettlemoyer. 2020. Semi-autoregressive training improves mask-predict decoding. *arXiv preprint arXiv:2001.08785*.

Sonal Gupta, Rushin Shah, Mrinal Mohit, Anuj Kumar, and Mike Lewis. 2018. Semantic parsing for task oriented dialog using hierarchical representations. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 2787–2792, Brussels, Belgium. Association for Computational Linguistics.

Norman P Jouppi, Cliff Young, Nishant Patil, David Patterson, Gaurav Agrawal, Raminder Bajwa, Sarah Bates, Suresh Bhatia, Nan Boden, Al Borchers, et al. 2017. In-datacenter performance analysis of a tensor processing unit. In *Proceedings of the 44th annual international symposium on computer architecture*, pages 1–12.

Diederik P Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*.

Jason Lee, Elman Mansimov, and Kyunghyun Cho. 2018. Deterministic non-autoregressive neural sequence modeling by iterative refinement. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 1173–1182, Brussels, Belgium. Association for Computational Linguistics.

Jiwei Li, Michel Galley, Chris Brockett, Jianfeng Gao, and Bill Dolan. 2016. A diversity-promoting objective function for neural conversation models. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 110–119, San Diego, California. Association for Computational Linguistics.

Vladislav Lialin, Rahul Goel, Andrey Simanovsky, Anna Rumshisky, and Rushin Shah. 2020. Update frequently, update fast: Retraining semantic parsing systems in a fraction of time. *arXiv preprint arXiv:2010.07865*.

Jindřich Libovický and Jindřich Helcl. 2018. End-to-end non-autoregressive neural machine translation with connectionist temporal classification. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 3016–3021, Brussels, Belgium. Association for Computational Linguistics.

Bing Liu and Ian Lane. 2016. Attention-based recurrent neural network models for joint intent detection and slot filling. *Interspeech 2016*, pages 685–689.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Roberta: A robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692*.

Grégoire Mesnil, Yann Dauphin, Kaisheng Yao, Yoshua Bengio, Li Deng, Dilek Hakkani-Tur, Xiaodong He, Larry Heck, Gokhan Tur, Dong Yu, et al. 2014. Using recurrent neural networks for slot filling in spoken language understanding. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 23(3):530–539.

Prafull Prakash, Saurabh Kumar Shashidhar, Wenlong Zhao, Subendhu Rongali, Haidar Khan, and Michael Kayser. 2020. Compressing transformer-based semantic parsing models using compositional code embeddings. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 4711–4717, Online. Association for Computational Linguistics.

Subendhu Rongali, Luca Soldaini, Emilio Monti, and Wael Hamza. 2020. Don't parse, generate! a sequence to sequence architecture for task-oriented semantic parsing. In *Proceedings of The Web Conference 2020*, pages 2962–2968.

Chitwan Saharia, William Chan, Saurabh Saxena, and Mohammad Norouzi. 2020. Non-autoregressive machine translation with latent alignments. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1098–1108, Online. Association for Computational Linguistics.

Abigail See, Peter J. Liu, and Christopher D. Manning. 2017. Get to the point: Summarization with pointer-generator networks. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1073–1083, Vancouver, Canada. Association for Computational Linguistics.

Bo Shao, Yeyun Gong, Weizhen Qi, Guihong Cao, Jianshu Ji, and Xiaola Lin. 2020. Graph-based transformer with cross-candidate verification for semantic parsing. In *Proceedings of the AAAI Conference on Artificial Intelligence*.

Akshat Shrivastava, Pierce Chuang, Arun Babu, Shrey Desai, Abhinav Arora, Alexander Zotov, and Ahmed Aly. 2021. Span pointer networks for non-autoregressive task-oriented semantic parsing. In *Findings of the Association for Computational Linguistics: EMNLP 2021*, pages 1873–1886, Punta Cana, Dominican Republic. Association for Computational Linguistics.

Mitchell Stern, William Chan, Jamie Kiros, and Jakob Uszkoreit. 2019. Insertion transformer: Flexible sequence generation via insertion operations. In *International Conference on Machine Learning*, pages 5976–5985. PMLR.

Christian Szegedy, Vincent Vanhoucke, Sergey Ioffe, Jon Shlens, and Zbigniew Wojna. 2016. Rethinking the inception architecture for computer vision. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2818–2826.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in neural information processing systems*, pages 5998–6008.

Ashwin K Vijayakumar, Michael Cogswell, Ramprasaath R Selvaraju, Qing Sun, Stefan Lee, David Crandall, and Dhruv Batra. 2018. Diverse beam search for improved description of complex scenes. In *Thirty-Second AAAI Conference on Artificial Intelligence*.

Oriol Vinyals, Meire Fortunato, and Navdeep Jaitly. 2015. Pointer networks. *Advances in Neural Information Processing Systems*, 28:2692–2700.

Sam Wiseman and Alexander M. Rush. 2016. Sequence-to-sequence learning as beam-search optimization. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 1296–1306, Austin, Texas. Association for Computational Linguistics.

Yonghui Wu, Mike Schuster, Zhifeng Chen, Quoc V Le, Mohammad Norouzi, Wolfgang Macherey, Maxim Krikun, Yuan Cao, Qin Gao, Klaus Macherey, et al. 2016. Google's neural machine translation system: Bridging the gap between human and machine translation. *arXiv preprint arXiv:1609.08144*.

Jingjing Xu, Xuancheng Ren, Junyang Lin, and Xu Sun. 2018. Diversity-promoting GAN: A cross-entropy based generative adversarial network for diversified text generation. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 3940–3949, Brussels, Belgium. Association for Computational Linguistics.

Pengcheng Yin, Hao Fang, Graham Neubig, Adam Pauls, Emmanouil Antonios Platanios, Yu Su, Sam Thomson, and Jacob Andreas. 2021. Compositional generalization for neural semantic parsing via span-level supervised attention. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 2810–2823, Online. Association for Computational Linguistics.

Qile Zhu, Haidar Khan, Saleh Soltan, Stephen Rawls, and Wael Hamza. 2020. Don't parse, insert: Multilingual semantic parsing with insertion based decoding. In *Proceedings of the 24th Conference on Computational Natural Language Learning*, pages 496–506, Online. Association for Computational Linguistics.

A Comparisons with Other Benchmarks

We compare the AR and NAR models we used with some of the benchmarks in the literature for the TOP & TOPv2 datasets. While there are benchmarks with other pre-trained encoders, we report the prior models with pre-trained BERT encoders to be consistent with the AR and NAR models used in this work (Table 7).

Encoders	TOP	TOPv2
Non-autoregressive Models		
$\text{BERT}_{\text{BASE}}$ baseline (greedy)	82.56	84.86
$\text{BERT}_{\text{BASE}}$ **proposed** (greedy)	83.11	85.22
Autoregressive Models		
$\text{BERT}_{\text{BASE}}$ (beam size = 4) (Rongali et al., 2020)	83.13	-
$\text{BERT}_{\text{BASE}}$ (beam size = 5) (Prakash et al., 2020)	85.01	-
Transformer + $\text{BERT}_{\text{BASE}}$ (Shao et al., 2020)	82.51	-
$\text{BERT}_{\text{BASE}}$ (greedy-ours, index)	83.43	85.40
$\text{BERT}_{\text{BASE}}$ (greedy-ours, span)	83.40	85.56

Table 7: Reported performance of semantic parsers with BERT encoders on TOP and TOPv2 datasets. Greedy refers to greedy decoding (i.e., beam size 1).

We note that the two baseline AR semantic parsers used in this work are based on the AR architecture presented in Rongali et al. (2020). Our AR baselines perform comparably against the reported numbers in the literature, as shown in Table 7.

B Scoring Beam Outputs

Table 8 depicts the performance of the three scoring methods on the TOP dataset. Each scoring method sorts the beam outputs of the proposed NAR differently (Equation 13-15).

Our study indicates that the scoring method 3 works the best for our NAR model, empirically demonstrating the effectiveness of the length penalty (Wu et al., 2016) for NAR models. We observed that high length penalties (e.g., $\alpha = 2.0 - 3.0$) often corresponded to the highest EM. For fair comparison, we also applied the length penalty with $\alpha = 1.0$ for the AR models for all experiments reported in this work.

Scoring Method	Exact Match		
	top-1	top-2	top-3
S_1 (Equation 13)	83.11	84.88	85.36
S_2 (Equation 14)	83.11	84.95	85.49
S_3 (Equation 15)			
$\alpha = 1.0$	83.13	85.21	85.81
$\alpha = 3.0$	83.12	85.15	**86.00**
$\alpha = 5.0$	83.15	85.08	85.93

Table 8: Performance of the three scoring methods.

C Model Architecture Details

Here, we elaborate on the details of the model architecture and parameters used across different experiments. Table 9 specifies hyper-parameters we used to build the AR and NAR models used across different experiments.

We note that we use two separate Transformers for the frame length and intent modules. Our hypothesis is that the Transformer output vector space is different for top-level intent and frame length. We intended the frame length module to learn its best output representation as independently from intent as possible. The frame length is still conditioned on the top-level intent as in Equation 9. This way, the backprop gradient to the length module Transformer (denoted as frame_len_h in Figure 4) is separated from the gradient to the intent module. Two separate Transformer may seem to be an overkill, however, consider the number of parameters of the model (Table 9). The 134.1M params of the proposed NAR is mostly from the encoder (BERT; L12/H768), which has 110M parameters. The frame length and intent module each has about 5.7M parameters, and the remainder is for the decoder. In terms of the network size, the separate Transformer does not make much difference as opposed to an alternative architecture that has a shared Transformer for the length and intent.

Compared to the AR models, the baseline NAR has roughly 12.5% more model parameters as it includes the frame length module in addition. The proposed NAR has approximately 4.4% more model parameters than the baseline NAR due to the addition of the intent module. Figure 4 depicts the detailed model architecture for the proposed NAR. The figure includes (sub)-modules, inputs, outputs, as well as the dimensionality of them.

	AR	baseline NAR	proposed NAR
Number of parameters	114.1M	128.4M	134.1M
Number of TPUs used for training		8 Google Cloud TPUv2	
decoder (layer/hidden/head)		L4/H256/HD2	
length module (layer/hidden/head)	N/A	L8/H256/HD4	
intent module (layer/hidden/head)	N/A	N/A	L8/H256/HD4
nonlinearity		Relu	
model dropout		0.0316	
source embeddings dropout		0.0022	
λ_{length}	N/A	10	
λ_{intent}	N/A	N/A	100
optimizer		Adam	
learning rate		0.00004	
learning rate warmup steps		1000	
learning rate scheduler		Exponential Decay	
batch size		256	

Table 9: Hyper-parameters related to the models used across different experiments presented in this work.

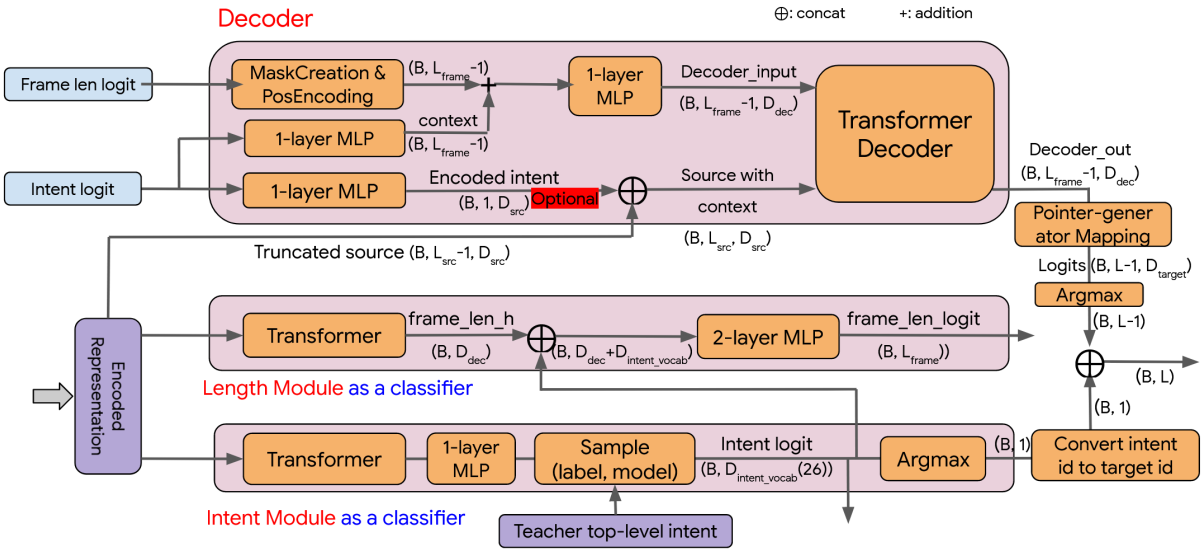

Figure 4: The detailed architecture of the proposed NAR.

Autoregressive Entity Generation for End-to-End Task-Oriented Dialog

Guanhuan Huang, Xiaojun Quan
School of Computer Science and Engineering
Sun Yat-sen University, Guangzhou, China
huanggh25@mail2.sysu.edu.cn
quanxj3@mail.sysu.edu.cn

Qifan Wang
Meta AI
Menlo Park, CA, USA
wqfcr@fb.com

Abstract

Task-oriented dialog (TOD) systems often require interaction with an external knowledge base to retrieve necessary entity (e.g., restaurant) information to support the response generation. Most current end-to-end TOD systems either retrieve the KB information explicitly or embed it into model parameters for implicit access. While the former approach demands scanning the KB at each turn of response generation, which is inefficient when the KB scales up, the latter approach shows higher flexibility and efficiency. In either approach, the systems may generate a response with conflicting entity information. To address this issue, we propose to generate the entity autoregressively first and leverage it to guide the response generation in an end-to-end system. To ensure entity consistency, we impose a trie constraint on entity generation. We also introduce a logit concatenation strategy to facilitate gradient backpropagation for end-to-end training. Experiments on MultiWOZ 2.1 single and CAMREST show that our system can generate more high-quality and entity-consistent responses.

1 Introduction

Task-oriented dialog (TOD) systems (Young et al., 2013; Budzianowski et al., 2018) have become prominent and drawn much attention from both academia and industries. Their mission is to help users accomplish specific tasks such as booking restaurants and reserving hotels through natural language conversations, where an external knowledge base (KB) is usually needed to support the generation of a system response. For example, when trying to recommend a restaurant, they will retrieve its address from the KB and generate a response.

Many recent state-of-the-art TOD systems (Mehri et al., 2019; Hosseini-Asl et al., 2020; Li et al., 2021) take a pipeline route that decomposes the task into modules that rely on intermediate annotations such as belief state and dialog act for supervision. These modules can be trained individually and assembled into a dialog system, mitigating the difficulty of generating a desired response directly from the dialog context and user utterance. Another motivation for the pipeline architecture is the necessity of querying KB with belief state, as shown in Figure 1, which would otherwise be non-trivial to realize in an end-to-end system. However, these annotations have to be crafted by human annotators, which is hardly realistic in practical scenes such as intelligent customer services where huge amounts of unannotated natural language conversations are accumulated. Besides, errors made in upstream modules may be propagated to downstream modules if they are not trained jointly.

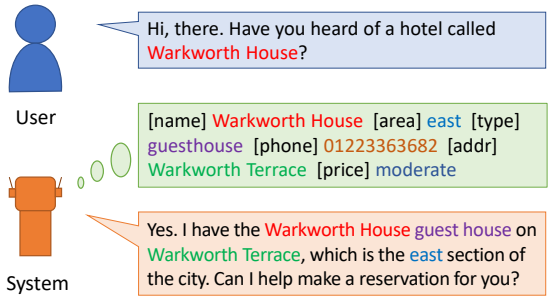

Figure 1: An example to show that task-oriented dialog systems need to retrieve information (middle) from a knowledge base (KB) to generate a qualified system response. Entity values in the KB are color-highlighted.

There are mainly two approaches to eliminating the reliance on intermediate annotations and generating system response in an end-to-end manner. First, entity information in the KB can be accessed by soft attention (Madotto et al., 2018; Reddy et al., 2019; Qin et al., 2020). To this end, a memory network is usually used to encode the KB, and attention and pointer are then utilized to retrieve entity information from the memory. These attention-based methods tend to become cumbersome when the KB scales up. Second, the KB information can be stored in model parameters to avoid direct in-

Proceedings of the 29th International Conference on Computational Linguistics, pages 323–332
October 12–17, 2022.

teraction with the KB during response generation (Madotto et al., 2020). This is partly motivated by the observation that pre-trained models such as BERT (Devlin et al., 2019) can carry certain relational and factual knowledge (Petroni et al., 2019). To embed the KB into model parameters, this approach first augments the original training set with KB entries and then encodes the training samples with a powerful encoder.

Despite the success in end-to-end TOD systems, one of the remaining problems is entity inconsistency during response generation (Qin et al., 2019), which means that the systems usually generate conflicting entity information in system responses. For example, they may generate a response "`Gourmet Kitchen is an Italian restaurant`" while `Gourmet Kitchen` is actually a `North American` restaurant. In this work, we aim to address this issue more scalably in our end-to-end TOD system. Following GPT-KE (Madotto et al., 2020), we first insert the KB into natural language dialogs by data augmentation, so that the KB can be embedded into model parameters whose size does not scale with the KB. Then, we predict the entity that will appear in the response autoregressively. To avoid generating an inconsistent entity, we impose a trie constraint on the decoding to ensure that the generated entity truly belongs to the KB. The generated entity is taken as an extra input to generate an entity-consistent system response. Besides, since tokens in the entity are integers, which hinders gradient backpropagation, we propose logit concatenation to allow for end-to-end training.

We evaluate our system on MultiWOZ 2.1 single (Budzianowski et al., 2018) and CAMREST (Wen et al., 2017), which are two task-oriented dialog benchmarks widely used in the literature. Experimental results show that it compares favorably with all the baselines. Particularly, it outperforms GPT-KE, a strong end-to-end TOD system that we follow, by a large margin. By ablation studies, we demonstrate that autoregressive entity generation assists in producing entity-consistent system responses in an end-to-end manner.

To our best knowledge, this is the first work that attempts to alleviate the entity inconsistency problem in TOD systems by generating the entity first and taking it as an input for response generation. The system can be trained end-to-end without accessing external KBs during response generation.

2 Related Work

End-to-end task-oriented dialog systems have drawn increasing attention in recent years. In one line of work, researchers propose to train the modules of a pipeline system jointly in an end-to-end framework, though they still require intermediate annotations for supervision. Among these works, SimpleTOD (Hosseini-Asl et al., 2020), SOLOIST (Peng et al., 2020), and UBAR (Yang et al., 2021) attempt to concatenate the dialog history, user utterance, belief state, dialog act, and system response into a long sequence, which is then modeled by a sequence-to-sequence generation model. HyKnow (Gao et al., 2021) extends the belief state to handle both structured and unstructured knowledge and trains the dialog state tracking and response generation modules jointly. Nevertheless, these systems are not the end-to-end solutions we pursue in this work since they still need intermediate annotations.

There are mainly two approaches to implementing intermediate annotations free end-to-end TOD systems. First, entity information in the KB can be accessed by soft attention. Mem2Seq (Madotto et al., 2018) combines the ideas of multi-hop attention over memory and a pointer network to incorporate KB information. Wen et al. (2018) proposed to compute a dialogue state representation from the dialog history and use it to interact with KB representations to retrieve entity information for response generation. GLMP (Wu et al., 2019) encodes the representations of dialog history and structural KB with a memory network and then passes the result to a decoder for response generation. DF-Net (Qin et al., 2020) includes a dynamic fusion module to generate a fused representation that explicitly captures the correlation between domains and uses it to query the KB. When the KB scales up, however, attention-based methods become less efficient.

Second, the KB information can be stored in model parameters to avoid further interaction with the KB during response generation. The motivation comes from the observation that pre-trained language models such as BERT (Devlin et al., 2019) and T5 (Raffel et al., 2020) can already carry certain relational and factual knowledge (Petroni et al., 2019). GPT-KE (Madotto et al., 2020) is the seminal dialog system towards this goal. It first augments the training set with KB entries and then learns a response generation model from the augmented set in an end-to-end fashion, thus abandoning the KB during response generation.

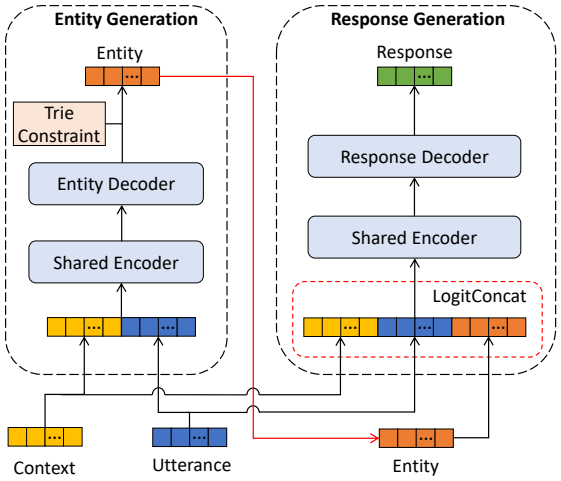

Figure 2: The architecture of our ECO system. The entity generation module takes the context and user utterance as input and generates a relevant entity. The response generation module takes the context, user utterance, and the generated entity as input to generate a system response. The two modules share the same encoder but have separate decoders. A trie constraint is imposed when generating the entity, and LogitConcat is proposed to facilitate end-to-end optimization.

3 Methodology

As shown in Figure 2, our Entity-COnsistent end-to-end (ECO) task-oriented dialog system begins by embedding the KB into training dialogs (§3.2). Following GPT-KE (Madotto et al., 2020), we augment the original training set with KB entries and abandon the KB afterward. Unlike GPT-KE, which conducts data augmentation in data pre-processing before training, ECO conducts augmentation for each batch of training samples, which reduces the size of augmented training samples while maintaining high coverage of the KB. We then predict the entity (§3.3) that will appear in the response and incorporate it into response generation (§3.4) to ensure entity consistency, where LogitConcat is proposed to facilitate end-to-end optimization.

3.1 Notations

Given a training set $\mathcal{D}_{tr} = \{D_1, D_2, \ldots, D_N\}$ of dialogs, where $D_i = \{U_{i,1}, R_{i,1}, \ldots, U_{i,T}, R_{i,T}\}$ contains T turns of user utterance and system response, we denote the conversational context of the t-th turn in dialog D_i as $C_{i,t} = \{U_{i,1}, R_{i,1}, \ldots, U_{i,t-1}, R_{i,t-1}\}$. A structured knowledge base is given in the form of a set of entities $KB = \{E_1, E_2, \ldots, E_M\}$, each of which is represented as a sequence $E_i = \{a_1, v_{i,1}, a_2, v_{i,2}, \ldots, a_K, v_{i,K}\}$ in which a_j and $v_{i,j}$ denote the jth attribute and its value for entity E_i, respectively. For simplicity, we assume each turn of dialog only relates to one entity and reformulate it as $\{U_{i,t}, R_{i,t}, E_{i,t}\}$. A user goal (Schatzmann et al., 2007) is defined for each dialog as $G_i = (G_{i,c}, G_{i,r})$, where $G_{i,c}$ specifies the constrained information (e.g., {location=`center`, price=`cheap`}) and $G_{i,r}$ denotes the required information (e.g., `address`, `name`).

Original dialog:	i am sorry but gourmet burger kitchen was the only north american restaurant in the centre area .
Template:	i am sorry but [name] was the only [food] restaurant in the [area] area .
New dialog:	i am sorry but da vinci pizzeria was the only italian restaurant in the north area .

Figure 3: An example to show how to construct a template from the training sample and generate a new sample from the template. Attributes are color-highlighted.

3.2 Knowledge Base Embedding

To embed the KB into the training set, we first extract all mentioned entity values in both user utterances and ground truth responses based on given span annotations in the original training set. Then, we match entity values with the KB to identify which entity is mentioned in the current turn of conversation. Templates are then constructed by replacing entity-related tokens in the conversation with special attribute placeholders. For example, `north american` in Figure 3 is replaced with the corresponding attribute placeholder `[food]`. This template generation function is denoted as $\text{DELEX}(\cdot)$, which is used to generate a set $\mathcal{D}_{tm}$ of templates from the original training set $\mathcal{D}_{tr}$:

$$\mathcal{D}_{tm} = \text{DELEX}(\mathcal{D}_{tr}). \quad (1)$$

Next, we generate new dialog samples with the templates in $\mathcal{D}_{tm}$. We refer to this process as data augmentation. To begin with, we obtain a set of KB entities, $\mathcal{G}_{mt} = \{E_1, E_2, \ldots, E_G\}$, that match the predefined user goals. Then, we randomly select an entity E_i from $\mathcal{G}_{mt}$ and replace the placeholders with the corresponding values of E_i. For instance, we replace `[food]` and `[area]` in Figure 3 with `italian` and `north`, respectively. The function of generating samples from templates is defined as $\text{RELEX}(\cdot)$, which is executed P times to insert P entities, producing a new set $\mathcal{D}_{au}$:

$$\mathcal{D}_{au} = \bigcup_{p=1}^{P} \text{RELEX}(\mathcal{D}_{tm}). \quad (2)$$

Note that usually only a subset of samples in $\mathcal{D}_{tr}$ can successfully match with entities in KB during data augmentation, making $\mathcal{D}_{au}$ not cover all the samples of $\mathcal{D}_{tr}$. For this reason, we join $\mathcal{D}_{tr}$ and $\mathcal{D}_{au}$ to get our final training set $\mathcal{D}_{fn}$.

$$\mathcal{D}_{fn} = \mathcal{D}_{tr} \bigcup \mathcal{D}_{au} \tag{3}$$

The selected entities during the above augmentation process are treated as ground truth entities for the corresponding dialog samples. This means that only the samples in $\mathcal{D}_{au}$ have entity labels while the samples in $\mathcal{D}_{tr}$ do not. Since all the placeholders in the templates are replaced with values from the same entity, this data augmentation process ensures that the augmented training samples contain consistent entity information.

3.3 Autoregressive Entity Generation

To predict the entity that will appear in the response, we propose to generate it autoregressively. For the sake of brevity, we use C_t and U_t to represent the current dialog context and user utterance, respectively. Then, we concatenate C_t and U_t, encode them into a vector representation, and take it as an input for entity generation:

$$\mathbf{g}_t = \mathrm{Enc}(\mathrm{Emb}([C_t; U_t])), \tag{4}$$

where $\mathrm{Emb}(\cdot)$ is the embedding function implemented by a global embedding matrix $\mathbf{W}_e$. $\mathrm{Enc}(\cdot)$ denotes the encoder which is shared with the response generation module (§3.4).

To generate an entity $\hat{E}_t$ autoregressively, the decoder iteratively predicts a token $\hat{e}_{t,k}$ based on the already generated sequence $\hat{E}_{t,<k}$ and vector representation $\mathbf{g}_t$:

$$\hat{P}(\hat{e}_{t,k}) = \mathrm{Dec}_e(\hat{e}_{t,k} | \hat{E}_{t,<k}, \mathbf{g}_t). \tag{5}$$

Since the gold entities of the samples in $\mathcal{D}_{au}$ are known, the cross-entropy loss of entity generation on $\mathcal{D}_{au}$ is defined as:

$$\mathcal{L}_{en} = \sum_{D \in \mathcal{D}_{au}} \sum_{E_t \in D} \mathrm{CELoss}(\hat{E}_t, E_t), \tag{6}$$

where E_t denotes the ground truth entity for the t-th turn of conversation.

For the samples in $\mathcal{D}_{tr}$, which have no entity labels, we do not calculate their loss during entity generation, but instead calculate their loss in response generation (§3.4) to realize end-to-end optimization like DualTKB (Dognin et al., 2020).

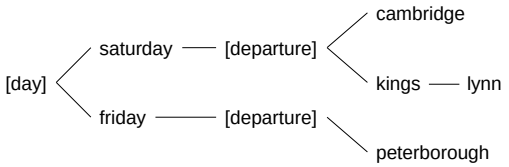

Figure 4: A trie with three entity sequences: `[day] saturday [departure] cambridge`, `[day] saturday [departure] kings lynn`, and `[day] friday [departure] peterborough`.

3.3.1 Trie Constraint

Inspired by GENRE (Cao et al., 2021), we construct a trie structure (a prefix tree) to ensure the generated entity truly belongs to the KB. For each entity in the KB, we construct a sequence as follows. For each value in an entity, we put its attribute placeholder to precede it and concatenate all pairs of attribute and value to form a sequence such as `[name] cityroomz [area] centre [type] hotel`. As depicted in Figure 4, a node in the trie denotes a token, and its child nodes denote all the succeeding tokens.

When decoding the k-th token $\hat{e}_{t,k}$ during the generation of entity $\hat{E}_t$, we have the decoded sequence $\hat{E}_{t,<k} = \{\hat{e}_{t,1}, \hat{e}_{t,2}, \ldots, \hat{e}_{t,k-1}\}$ in hand and walk through the trie along the path of $\hat{E}_{t,<k}$ to generate the next token. We use $\mathcal{E}_{t,k}$ to represent the set of possible tokens at this time step and re-compute $\hat{P}(\hat{e}_{t,k})$ as:

$$P(\hat{e}_{t,k}) = \begin{cases} \frac{\hat{P}(\hat{e}_{t,k})}{Z}, & \hat{e}_{t,k} \in \mathcal{E}_{t,k} \\ 0, & \text{else} \end{cases} \tag{7}$$

where

$$Z = \sum_{\hat{e}_{t,k} \in \mathcal{E}_{t,k}} \hat{P}(\hat{e}_{t,k}). \tag{8}$$

Since only tokens from $\mathcal{E}_{t,k}$ have non-zero probabilities in $P(\hat{e}_{t,k})$, the model always samples a token from $\mathcal{E}_{t,k}$. Therefore, the generated entity is guaranteed to be valid constantly.

3.4 Response Generation

During training, for each sample in $\mathcal{D}_{au}$, the model generates a response based on the context, user utterance, and the corresponding ground truth entity by concatenating and encoding them with the shared encoder defined in Eq. (4):

$$\mathbf{h}_t = \mathrm{Enc}(\mathrm{Emb}([C_t; U_t; E_t])). \tag{9}$$

For each sample in $\mathcal{D}_{tr}$ which has no ground truth entity label, the generated entity is used:

$$\mathbf{h}_t = \mathrm{Enc}(\mathrm{Emb}([C_t; U_t; \hat{E}_t])). \tag{10}$$

The response decoder then takes $\mathbf{h}_t$ as input and generates the response $\hat{R}_t$ token by token as:

$$P(\hat{r}_{t,k}) = \text{Dec}_r(\hat{r}_{t,k}|\hat{R}_{t,<k}, \mathbf{h}_t). \quad (11)$$

The cross-entropy loss is calculated between the generated response $\hat{R}_t$ and the ground truth response R_t:

$$\mathcal{L}_{re} = \sum_{D \in \mathcal{D}_{fn}} \sum_{R_t \in D} \text{CELoss}(\hat{R}_t, R_t). \quad (12)$$

3.4.1 Logit Concatenation

For those samples in $\mathcal{D}_{tr}$, since the tokens in each generated entity are integers, the gradients of response generation cannot be directly passed to the encoder during training. To address this, we modify Eq. (10) and input the distributions of generated entity tokens to the encoder. Specifically, for the k-th token $\hat{e}_{t,k}$ of a generated entity $\hat{E}_t$, its output distribution $P(\hat{e}_{t,k})$ over vocabulary from the entity decoder is first computed using Eq. (7) and then used to approximate $\hat{e}_{t,k}$ for gradient propagation. $P(\hat{e}_{t,k})$ can be encoded as:

$$\mathbf{h}_{t,k} = P(\hat{e}_{t,k})\mathbf{W}_e^T, \quad (13)$$

where $\mathbf{W}_e$ is the global embedding matrix introduced above.

If both $P(\hat{e}_{t,k})$ and $\mathbf{W}_e$ receive gradients during propagation, the training may collapse since it is much easier to update $\mathbf{W}_e$ than $P(\hat{e}_{t,k})$, which requires understanding the context and utterance to obtain relevant information. Therefore, we alter the equation by stopping gradients on $\mathbf{W}_e$:

$$\hat{\mathbf{h}}_{t,k} = P(\hat{e}_{t,k}) \cdot \text{StopGrad}(\mathbf{W}_e^T), \quad (14)$$

$$\hat{\mathbf{h}}_t = \{\hat{\mathbf{h}}_{t,1}, \ldots, \hat{\mathbf{h}}_{t,|\hat{E}_t|}\}. \quad (15)$$

We use $\hat{\mathbf{h}}_t$ as the representation of entity $\hat{E}_t$ and concatenate it with the embeded context C_t and user utterance U_t, which is then encoded to replace Eq. (10) during training:

$$\mathbf{h}_t = \text{Enc}(\text{Emb}([C_t; U_t]); \hat{\mathbf{h}}_t]). \quad (16)$$

Since $P(\hat{e}_{t,k})$ is a distribution vector rather than an integer, gradients can be backpropagated to the encoder during training. At inference time, we take the generated entity tokens rather than $P(\hat{e}_{t,k})$ as input for response generation, as described in Eq. (10). This brings a gap between training and inference, which will be studied in Section 4.5.

3.5 Joint Training

The final system is trained by minimizing the sum of entity loss $\mathcal{L}_{en}$ and response loss $\mathcal{L}_{re}$:

$$\mathcal{L} = \mathcal{L}_{en} + \mathcal{L}_{re}. \quad (17)$$

4 Experiments

4.1 Dataset

We conduct experiments on MultiWOZ 2.1 single (Budzianowski et al., 2018) and CAMREST (Wen et al., 2017). CAMREST consists of one domain of Cambridge restaurant booking while MultiWOZ 2.1 single consists of five domains: `Attraction`, `Hotel`, `Restaurant`, `Taxi`, and `Train`. Following previous work (Qin et al., 2020; Madotto et al., 2020), we select only the dialogues which involves a single domain from MultiWOZ 2.1 to form the MultiWOZ 2.1 single dataset. We follow the same pre-processing and augmentation procedures as GPT-KE (Madotto et al., 2020). Note that not all dialogs in the original training set can be successfully used to generate templates due to the diversity of entity values. On MultiWOZ 2.1 single, 63/116/289/59 templates are respectively generated for domains `Attraction`/`Hotel`/`Restaurant`/`Train`, and no template is generated for the `Taxi` domain since MultiWOZ 2.1 single does not provide KB for this domain. On CAMREST, 161 templates are constructed for data augmentation.

Following previous works (Qin et al., 2020; Madotto et al., 2020), we adopt BLEU, Inform, Success, and F1 as the metrics to evaluate model performance on MultiWOZ 2.1 single, and employ BLEU, F1, and Success on CAMREST. Inform and Success are calculated based on the given user goal of a dialog session, and inconsistent entity information will lower the two metrics. Meanwhile, an overall score is also calculated: Score $=$ BLEU $+$ (Inform $+$ Success)$/2$.

4.2 Measuring Entity Consistency

Measuring the entity consistency of a given system response remains a problem in task-oriented dialog systems. Qin et al. (2021) annotated three kinds of inconsistency by human experts on the KVRET (Eric et al., 2017) dataset, i.e., user query inconsistency, dialog history inconsistency, and knowledge base inconsistency. They then trained models as a form of automatic metrics to identify which kind of inconsistency appears in system responses.

	BLEU	Inform	Success	Score	F1	Consistency
Mem2Seq (Madotto et al., 2018)	6.60	-	-	-	21.62	-
DSR (Wen et al., 2018)	9.10	-	-	-	30.00	-
GLMP (Wu et al., 2019)	6.90	-	-	-	32.40	-
DF-Net (Qin et al., 2020)	9.40	-	-	-	35.10	-
GPT2 (Radford et al., 2019)	14.33	64.60	51.77	72.52	30.38	-
GPT-KE (Madotto et al., 2020)	**15.05**	72.57	64.16	83.42	39.58	54.46
BART-KE	12.80±0.22	70.94±2.05	61.36±2.12	78.95±2.05	39.31±0.22	52.96±0.48
ECO (ours)	12.61±0.20	**83.63±0.63**	**75.37±0.21**	**92.11±0.20**	**40.87±0.24**	**56.84±0.36**

Table 1: Main results on MultiWOZ. Scores of baselines except BART-KE are from original papers, and "-" denotes scores originally unavailable. BART-KE is the baseline implemented by replacing GPT-2 in GPT-KE with BART.

However, their method is trained on KVRET and may not be suitable to measure inconsistency on other datasets. Furthermore, the first few turns may include irrelevant entity information, such as providing several hotels for the user to choose, which makes it difficult to identify whether the generated response is dialog history consistent or not.

The above analysis motivates us to propose a new consistency metric that focuses on user query consistency and knowledge base consistency. It is a turn-level metric that requires all entity information in the user utterance and the system response to belong to the same entity in KB. To be specific, we first extract all entity information in the user utterance and the system response, and then search the knowledge base. If there is an entity that contains all the extracted information, this turn of conversation scores 1, and 0 otherwise. The final Consistency metric is calculated as the average score over all conversation turns. The method of extracting entity information from utterances and responses is the same as in calculating F1.

4.3 Experiment Settings

Different from GPT-KE, we use BART (Lewis et al., 2020) as our backbone model due to the limitation of computation power. We also replace GPT-2 (Radford et al., 2019) in GPT-KE with BART to form a new baseline, BART-KE. We set the max input sequence length to 256, the repeat times P in RELEX to 12, and the batch size to 12. Experiments are conducted on a single NVIDIA 2080ti and cost about 11G GPU RAM. We conduct ablation studies on MultiWOZ 2.1 single as it is a more challenging dataset with multiple domains of dialogs. For most variants of our method, we run 30 epochs and evaluate them per 5 epochs, saving a model checkpoint after each evaluation. We then select the best checkpoint based on model performance on the development set and finally report the test results. For the ablation setting of *w/ tr*, which lacks supervision from gold entity labels for training, we run 50 epochs to select the best.

	BLEU	F1	Success
KB-Trs	14.80	45.30	-
MLMN	13.61	54.85	-
BoSsNet	15.20	43.10	-
KBRet	**18.64**	55.76	62.03
GPT-KE	18.00	54.85	74.68
BART-KE	17.84±0.28	70.42±0.37	75.06±1.52
ECO (ours)	18.42±0.27	**71.56±0.39**	**78.77±1.85**

Table 2: Main results on CAMREST. KB-Trs (E. et al., 2019), MLMN (Reddy et al., 2019), BoSsNet (Raghu et al., 2019), KBRet (Qin et al., 2019), and GPT-KE (Madotto et al., 2020) are baselines for comparison.

4.4 Main Results

The overall results are shown in Table 1 and Table 2. We observe that ECO outperforms GPT-KE and other baselines by a large margin in all metrics except BLEU, showing that ECO can reach the user goals of this dialog dataset more effectively. The improvement of ECO over BART-KE suggests that ECO's success mainly comes from the model design rather than BART itself. Specifically, by generating an entity with trie constraint to help response generation, ECO obtains consistent entity information and improves entity consistency of generated response. On the other hand, we note that BART-based methods (BART-KE and ECO) achieve relatively lower BLEU scores than the GPT-2 family baselines (GPT-2 and GPT-KE) on MultiWOZ. The main reason should be that we do not post-train BART with language modeling objectives on the training set, which affects the fluency of generated responses, while responses in MultiWOZ are more diverse across domains.

	Percentage (%)	Inform	Success	F1
single inform	46.0	91.67±1.63	83.01±1.98	61.09±0.81
multi inform	54.0	76.78±1.55	68.85±1.77	31.03±0.73
single success	84.5	83.60±1.23	77.49±0.43	42.74±0.10
multi success	15.5	83.81±2.69	63.81±1.35	33.62±1.28
total	100.0	83.63±0.63	75.37±0.21	40.87±0.24

Table 3: Results of study on how multiple matched entities affect evaluation metrics, where single/multi inform/success refer to the situation with single/multiple matched entities when calculating Inform/Success, and Percentage (%) means the proportion of samples in the test set.

We also analyze why the improvement of F1 is much smaller than that of Inform and Success on MultiWOZ. Inform and Success are calculated based on user goals, and in some circumstances, there are multiple entities that match a user goal. However, only the one that is mentioned in the ground truth response is counted as correct in F1. Therefore, a large improvement on Inform and Success means ECO achieves user goals better, but the entity mentioned in the generated response may be different from the one in the ground truth. As shown in Table 3, over 50% of test samples have multiple matched entities when calculating Inform, and the percentage is 15.5% when calculating Success. Multiple matched entities reduce model performance on all metrics, especially on F1.

4.5 Ablation Studies

4.5.1 Training Datasets

Unlike samples in $\mathcal{D}_{tr}$, samples in $\mathcal{D}_{au}$ have ground truth entity labels, so their training objectives are different. We use ECO *w/ tr* to denote the training set that only contains samples from $\mathcal{D}_{tr}$ for end-to-end training, and use ECO *w/ au* to denote the training set that only contains samples from $\mathcal{D}_{au}$. As the results show in Table 4, ECO *w/ tr* has drops of 16.08 on Inform, 20.5 on Success, 4.42 on F1, and 4.41 on Consistency compared to ECO. Actually, without entity labels, the entity generation process is hard to converge, making response generation lack entity information as input and lowering model performance as a result.

On the other hand, ECO *w/ au* has less drops than ECO *w/ tr* compared to ECO on Inform, Success, and Consistency. However, the drops of ECO *w/ au* is more obvious on F1, which is caused by the fact that $\mathcal{D}_{au}$ does not includes the `Taxi` domain. These results demonstrate that the augmentation introduces more important KB information for response generation than the original training set.

4.5.2 Trie Constraint

In this work, the trie constraint is the key to guaranteeing entity consistency in a system response. Figure 5 presents an example of decoding an entity on the trie. Through filtering out non-kid nodes, the decoding path is restricted to a path on the trie. From Table 4, we note that ECO outperforms ECO *w/o trie* by 2.95 on Inform, 3.25 on Success, 1.06 on F1, and 0.53 on Consistency. Thus, we can conclude that the model generates more informative responses with the help of consistently generated entities, which accounts for the improvement.

4.5.3 Logit Concatenation

LogitConcat is proposed to enable backpropagation after concatenating the dialogue context, user utterance, and the generated entity when training on $\mathcal{D}_{tr}$. Without this component, the model parameters of the entity generator will not be updated. In Table 4, ECO shows a promising improvement of 5.31 on Inform, 4.87 on Success, 0.99 on F1, and 1.69 on Consistency over ECO *w/o LogitConcat*.

4.5.4 Gap between Training and Evaluation

During the evaluation, ECO uses the generated entity sequence as an input for response generation, which is different from the training phase that uses LogitConcat. To study the impact, we conduct an experiment that also applies LogitConcat during the evaluation. From the results (ECO *w/ LogitEval*) in Table 4, we observe obvious drops of ECO on Inform, Success, and Consistency. This phenomenon shows that applying LogitConcat during the evaluation weakens the consistency of the generated entities since the probability distribution in LogitConcat is not a valid entity from the KB.

4.5.5 The Number of Templates

To study how the number of templates affects model performance, we randomly select several subsets of templates from the whole template set.

	BLEU	Inform	Success	Score	F1	Consistency
GPT-KE	**15.05**	72.57	64.16	83.42	39.58	54.46
BART-KE	12.80±0.22	70.94±2.05	61.36±2.12	78.95±2.05	39.31±0.22	52.96±0.48
ECO	12.61±0.20	**83.63±0.63**	**75.37±0.21**	**92.11±0.20**	**40.87±0.24**	**56.84±0.36**
w/ au	8.94±0.06	79.20±2.26	56.34±0.55	76.71±0.94	30.38±1.67	55.49±0.33
w/ tr	11.21±0.37	67.55±4.41	54.87±4.38	72.42±4.07	36.45±1.17	52.43±1.89
w/o trie	12.40±0.36	80.68±0.91	72.12±1.25	88.80±0.81	39.81±0.25	56.31±0.62
w/o LogitConcat	12.52±0.11	78.32±0.96	70.50±2.46	86.93±1.64	39.88±0.40	55.15±0.84
w/ LogitEval	12.85±0.28	71.98±0.55	65.04±0.96	81.36±1.00	40.58±0.46	53.62±0.81

Table 4: Results of ablation studies. ECO *w/ au* represents the ECO variant trained on samples of $\mathcal{D}_{au}$, and ECO *w/ tr* represents the ECO variant trained on samples of $\mathcal{D}_{tr}$. ECO *w/o trie* means that ECO does not apply the trie constraint during entity generation, while ECO *w/o LogitConcat* means it does not apply LogitConcat during training. ECO *w/o StopGrad* represents the ECO variant that drops StopGrad in LogitConcat. ECO *w/ LogitEval* represents the ECO variant that applies LogitConcat during inference.

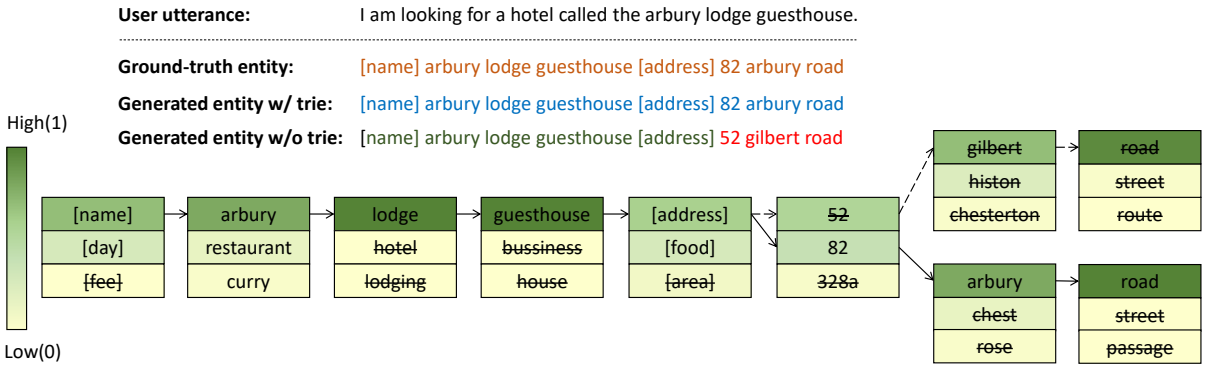

Figure 5: An example of decoding entity with the trie constraint. Darker backgrounds represent high decoding probabilities. The trie constraint filters out some tokens during decoding and results in a different decoding path, avoiding the red path which has a higher probability but generates inconsistent entity information.

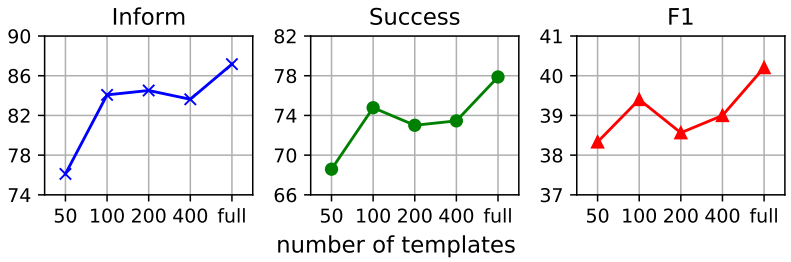

Figure 6: Ablation study of how the number of templates affects model performance on MuitiWOZ, where *full* means all the templates are used for knowledge base embedding.

As shown in Figure 6, the performance on all the metrics generally grows when the number of templates increases, but there are fluctuations when the number changes from 100 to 400.

5 Conclusion

We proposed an end-to-end task-oriented dialog system by encoding external knowledge into model parameters. To address entity inconsistency in system responses, we proposed to generate the entities first and took them as input to response generation. To ensure the generated entities are valid, a trie constraint was imposed on the generation, and a logit concatenation strategy was introduced to facilitate backpropagation for end-to-end training. Experiments demonstrate that this system can produce more high-quality and entity-consistent responses in an end-to-end manner. For future work, we will extend this system to handle multiple entities that are involved in each turn of conversation.

Acknowledgments

This work was supported by the National Natural Science Foundation of China (No. 62176270) and the Program for Guangdong Introducing Innovative and Entrepreneurial Teams (No. 2017ZT07X355).

References

Pawel Budzianowski, Tsung-Hsien Wen, Bo-Hsiang Tseng, Iñigo Casanueva, Stefan Ultes, Osman Ramadan, and Milica Gasic. 2018. Multiwoz - A large-scale multi-domain wizard-of-oz dataset for task-oriented dialogue modelling. In *Proceedings of EMNLP 2018*, pages 5016–5026. ACL.

Nicola De Cao, Gautier Izacard, Sebastian Riedel, and Fabio Petroni. 2021. Autoregressive entity retrieval. In *ICLR 2021*. OpenReview.net.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: pre-training of deep bidirectional transformers for language understanding. In *Proceedings of NAACL-HLT 2019*, pages 4171–4186. ACL.

Pierre L. Dognin, Igor Melnyk, Inkit Padhi, Cícero Nogueira dos Santos, and Payel Das. 2020. Dualtkb: A dual learning bridge between text and knowledge base. In *Proceedings of EMNLP 2020*, pages 8605–8616. ACL.

Haihong E., Wenjing Zhang, and Meina Song. 2019. Kb-transformer: Incorporating knowledge into end-to-end task-oriented dialog systems. In *2019 15th International Conference on Semantics, Knowledge and Grids (SKG)*, pages 44–48.

Mihail Eric, Lakshmi Krishnan, Francois Charette, and Christopher D. Manning. 2017. Key-value retrieval networks for task-oriented dialogue. In *Proceedings of the 18th Annual SIGdial Meeting on Discourse and Dialogue*, pages 37–49, Saarbrücken, Germany. Association for Computational Linguistics.

Silin Gao, Ryuichi Takanobu, Wei Peng, Qun Liu, and Minlie Huang. 2021. Hyknow: End-to-end task-oriented dialog modeling with hybrid knowledge management. In *Findings of ACL/IJCNLP 2021*, pages 1591–1602. ACL.

Ehsan Hosseini-Asl, Bryan McCann, Chien-Sheng Wu, Semih Yavuz, and Richard Socher. 2020. A simple language model for task-oriented dialogue. In *NeurIPS 2020*.

Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Veselin Stoyanov, and Luke Zettlemoyer. 2020. BART: denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. In *Proceedings of ACL 2020*, pages 7871–7880. ACL.

Yunhao Li, Yunyi Yang, Xiaojun Quan, and Jianxing Yu. 2021. Retrieve & memorize: Dialog policy learning with multi-action memory. In *Findings of ACL/IJCNLP 2021*, pages 447–459. ACL.

Andrea Madotto, Samuel Cahyawijaya, Genta Indra Winata, Yan Xu, Zihan Liu, Zhaojiang Lin, and Pascale Fung. 2020. Learning knowledge bases with parameters for task-oriented dialogue systems. In *Findings of EMNLP 2020*, pages 2372–2394. ACL.

Andrea Madotto, Chien-Sheng Wu, and Pascale Fung. 2018. Mem2seq: Effectively incorporating knowledge bases into end-to-end task-oriented dialog systems. In *Proceedings of ACL 2018*, pages 1468–1478. ACL.

Shikib Mehri, Tejas Srinivasan, and Maxine Eskénazi. 2019. Structured fusion networks for dialog. In *Proceedings of SIGdial 2019*, pages 165–177. ACL.

Baolin Peng, Chunyuan Li, Jinchao Li, Shahin Shayandeh, Lars Liden, and Jianfeng Gao. 2020. SOLOIST: few-shot task-oriented dialog with A single pre-trained auto-regressive model. *CoRR*, abs/2005.05298.

Fabio Petroni, Tim Rocktäschel, Sebastian Riedel, Patrick S. H. Lewis, Anton Bakhtin, Yuxiang Wu, and Alexander H. Miller. 2019. Language models as knowledge bases? In *Proceedings of EMNLP-IJCNLP 2019*, pages 2463–2473. ACL.

Libo Qin, Yijia Liu, Wanxiang Che, Haoyang Wen, Yangming Li, and Ting Liu. 2019. Entity-consistent end-to-end task-oriented dialogue system with KB retriever. In *Proceedings of EMNLP-IJCNLP 2019*, pages 133–142. ACL.

Libo Qin, Tianbao Xie, Shijue Huang, Qiguang Chen, Xiao Xu, and Wanxiang Che. 2021. Don't be contradicted with anything! ci-tod: Towards benchmarking consistency for task-oriented dialogue system. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing, EMNLP 2021, Virtual Event / Punta Cana, Dominican Republic, 7-11 November, 2021*, pages 2357–2367. Association for Computational Linguistics.

Libo Qin, Xiao Xu, Wanxiang Che, Yue Zhang, and Ting Liu. 2020. Dynamic fusion network for multi-domain end-to-end task-oriented dialog. In *Proceedings of ACL 2020*, pages 6344–6354. ACL.

Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, Ilya Sutskever, et al. 2019. Language models are unsupervised multitask learners. *OpenAI blog*, 1(8):9.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J. Liu. 2020. Exploring the limits of transfer learning with a unified text-to-text transformer. *J. Mach. Learn. Res.*, 21:140:1–140:67.

Dinesh Raghu, Nikhil Gupta, and Mausam. 2019. Disentangling Language and Knowledge in Task-Oriented Dialogs. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 1239–1255, Minneapolis, Minnesota. Association for Computational Linguistics.

Revanth Reddy, Danish Contractor, Dinesh Raghu, and Sachindra Joshi. 2019. Multi-level memory for task oriented dialogs. In *Proceedings of NAACL-HLT 2019*, pages 3744–3754. ACL.

Jost Schatzmann, Blaise Thomson, Karl Weilhammer, Hui Ye, and Steve J. Young. 2007. Agenda-based user simulation for bootstrapping a POMDP dialogue system. In *Proceedings of NAACL-HLT 2007*, pages 149–152. ACL.

Haoyang Wen, Yijia Liu, Wanxiang Che, Libo Qin, and Ting Liu. 2018. Sequence-to-sequence learning for task-oriented dialogue with dialogue state representation. In *Proceedings of COLING 2018*, pages 3781–3792. ACL.

Tsung-Hsien Wen, David Vandyke, Nikola Mrksic, Milica Gasic, Lina Maria Rojas-Barahona, Pei-Hao Su, Stefan Ultes, and Steve J. Young. 2017. A network-based end-to-end trainable task-oriented dialogue system. In *Proceedings of EACL 2017*, pages 438–449. ACL.

Chien-Sheng Wu, Richard Socher, and Caiming Xiong. 2019. Global-to-local memory pointer networks for task-oriented dialogue. In *Proceedings of ICLR 2019*. OpenReview.net.

Yunyi Yang, Yunhao Li, and Xiaojun Quan. 2021. Ubar: Towards fully end-to-end task-oriented dialog system with gpt-2. *Proceedings of AAAI 2021*, 35(16):14230–14238.

Steve J. Young, Milica Gasic, Blaise Thomson, and Jason D. Williams. 2013. Pomdp-based statistical spoken dialog systems: A review. *Proc. IEEE*, 101(5):1160–1179.

Continual Few-shot Intent Detection

Guodun Li[1*], **Yuchen Zhai**[1], **QiangLong Chen**[1],
Xing Gao[2], **Ji Zhang**[2], **Yin Zhang**[1†]
[1]College of Computer Science and Technology, Zhejiang University
[2]DAMO Academy, Alibaba Group
{guodun.li,zhaiyuchen,chenqianglong,zhangyin98}@zju.edu.cn
{gaoxing.gx,zj122146}@alibaba-inc.com

Abstract

Intent detection is at the core of task-oriented dialogue systems. Existing intent detection systems are typically trained with a large amount of data over a predefined set of intent classes. However, newly emerged intents in multiple domains are commonplace in the real world. And it is time-consuming and impractical for dialogue systems to re-collect enough annotated data and re-train the model. These limitations call for an intent detection system that could continually recognize new intents with very few labeled examples. In this work, we study the Continual Few-shot Intent Detection (CFID) problem and construct a benchmark consisting of nine tasks with multiple domains and imbalanced classes. To address the key challenges of (a) catastrophic forgetting during continuous learning and (b) negative knowledge transfer across tasks, we propose the Prefix-guided Lightweight Encoder (PLE) with three auxiliary strategies, namely Pseudo Samples Replay (PSR), Teacher Knowledge Transfer (TKT) and Dynamic Weighting Replay (DWR). Extensive experiments demonstrate the effectiveness and efficiency of our method in preventing catastrophic forgetting and encouraging positive knowledge transfer across tasks.

1 Introduction

Intent Detection (ID) is at the core of task-oriented dialogue systems. It aims at understanding the goals underlying user utterances and classifying them into different intents accurately (Zhang et al., 2020; Qin et al., 2021). Traditionally, the ID system is trained with plenty of labeled data to identify a predefined set of intent classes (Larson et al., 2019). However, newly emerged intents in multiple domains are commonplace in the real scenario. A naive approach to detecting new intents is to re-collect annotated data and re-train the model, which is time-consuming and impractical for dialogue systems in deployment. Thus, an intent detection system that could continually recognize new intents with very few labeled examples is called for.

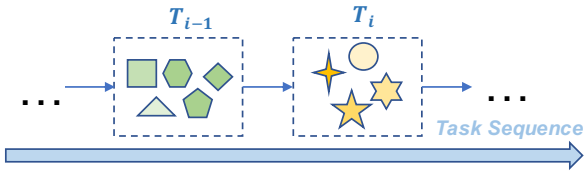

Setting	Few-shot	Continual	Multi-Domain	#Classes
FSID (Zhang et al., 2020)	✓	✗	✓	-
CID (Liu et al., 2021)	✗	✓	✓	-
FSCIL-ID (Xia et al., 2021)	✓	✓	✗	Balanced
CFID (Ours)	✓	✓	✓	Imblanced

Figure 1: Illustration of Continual Few-shot Intent Detection (CFID). Compared with existing works, CFID aims to recognize continually new intents from multiple domains with very few labeled examples and imbalanced number of classes across tasks.

Many efforts have been made in existing works to achieve this goal. Zhang et al. (2020) propose a discriminative nearest neighbor classification method to solve the Few-Shot Intent Detection (FSID) problem. They mainly focus on the data scarcity and ignore the ability to learn consecutive tasks. Liu et al. (2021) and Wang et al. (2021) propose promising solutions for the Continual Intent Detection (CID) problem. Nevertheless, they do not consider the few-shot setting, which is more realistic due to the scarcity of labeled data for new intents. The most recent work (Xia et al., 2021) provides the first study on few-shot class-incremental learning for intent detection (FSCIL-ID). However, it assumes (1) all tasks belong to the same domain and (2) the number of classes across few-shot tasks is balanced, which is not practical in the real world.

In this work, we define a more realistic problem as Continual Few-shot Intent Detection (CFID) and construct a benchmark consisting of nine tasks with multiple domains and imbalanced classes in 5-shot and 10-shot settings. As shown in Figure 1, the

[*] Work done during internship at Alibaba.
[†] Corresponding author: Yin Zhang.

Proceedings of the 29th International Conference on Computational Linguistics, pages 333–343
October 12–17, 2022.

system is provided with a sequence of tasks with limited labeled data and expected to continually learn on new intents while performing accurate classification on all previously seen tasks. Compared with existing works, CFID is more aligned with real scenario where the number of classes is highly imbalanced and task domains vary widely.

We consider addressing the problem from the intersection perspective of few-shot and lifelong learning. A strong baseline is to construct prototypical networks with a pre-trained language model (PrLM) and sequentially update all the weights on each task. However, there are two issues: (i) over-parameterization of PrLMs makes them prone to overfit the current task and cause *catastrophic forgetting* of previous knowledge (Ke et al., 2021a; Yuan et al., 2021). (ii) due to the domain gaps and imbalanced classes, the knowledge inherited from the past task may degrade the performance of the current, namely *negative knowledge transfer*.

To address the above issues, we propose a novel Prefix-guided Lightweight Encoder (PLE) with three auxiliary strategies. In detail, PLE adopts a parameter-efficient tuning paradigm to alleviate forgetting caused by over-parameterization, consisting of a lightweight Continual Adapter module to interact with a frozen PrLM, and a Prefix-guided Attention mechanism to guide the frozen PrLM. To further alleviate forgetting, we propose the Pseudo Samples Replay (PSR) strategy, which consolidates previous knowledge by replaying two essential samples that best approximate the previous tasks. To alleviate negative knowledge transfer, we propose the Teacher Knowledge Transfer (TKT) strategy, which transfers the task-specific knowledge into the current model via distillation to compensate for the performance drop of new tasks. Moreover, due to the variability of tasks, it is hard to identify whether a past task transfers positive or negative knowledge to the current. Thus, we propose the Dynamic Weighting Replay (DWR) strategy to balance learning new tasks and replaying old ones, which dynamically determines the learning weight of the old task in each iteration.

Our main contributions are as follows: 1) To the best of our knowledge, we are the first to formulate the Continual Few-shot Intent Detection (CFID) problem and construct a benchmark for it. 2) We propose a novel method PLE with three strategies for CFID to alleviate forgetting and negative transfer. 3) Extensive experiments show the effectiveness of our method in preventing forgetting and encouraging positive knowledge transfer across tasks.

2 Related Work

Traditional Intent Detection aims to classify intent in the utterance, which can be defined as a sentence classification task. Popular approaches such as Goo et al. (2018); Qin et al. (2019); Mehri et al. (2020) have achieved promising performance. However, such methods heavily rely on large amounts of labeled data.

Few-shot Intent Detection aims to classify accurately identify intents in few-shot settings. Zhang et al. (2020) solves it as a textual entailment problem and uses large-scale entailment datasets for pre-training. However, it is time-consuming and expensive to train with hundreds of intents. Mehri and Eric (2021) proposes an example-driven strategy to tackle this task, which learns to classify utterances by comparing them to examples. Luo et al. (2021) and Dopierre et al. (2021) solve the data scarcity by leveraging the label names or augmented samples. More recently, Zhang et al. (2021a,b) show the effectiveness of pre-training and contrastive fine-tuning on this task.

Continual Learning aims to learn a sequence of tasks incrementally. Most works in NLP domains focus on text classification tasks in continual settings (Ke et al., 2021c,a,b; Geng et al., 2021; Qin and Joty, 2022). The main problem for continual text classification is *catastrophic forgetting* and replay-based methods (Han et al., 2020; Cui et al., 2021) have been proven promising to alleviate the problem, which retain a few examples in previous tasks and continually replay them with new tasks.

Continual Intent Detection. Recently, Liu et al. (2021) and Wang et al. (2021) have made some efforts on the Continual Intent Detection task (CID). However, they did not further investigate with the few-shot setting, which is more challenging and crucial for the low-resource dialogue systems. The most similar to our work is (Xia et al., 2021), which firstly proposes the Few-shot Class-Incremental Learning for Intent Detection (FSCIL-ID). However, it is not aligned with the real scenario for the following reasons: (i) All tasks belong to the same domain without considering the domain gaps of different intents. (ii) The number of classes in emerging new tasks is fixed without considering the imbalance of classes across tasks in real systems.

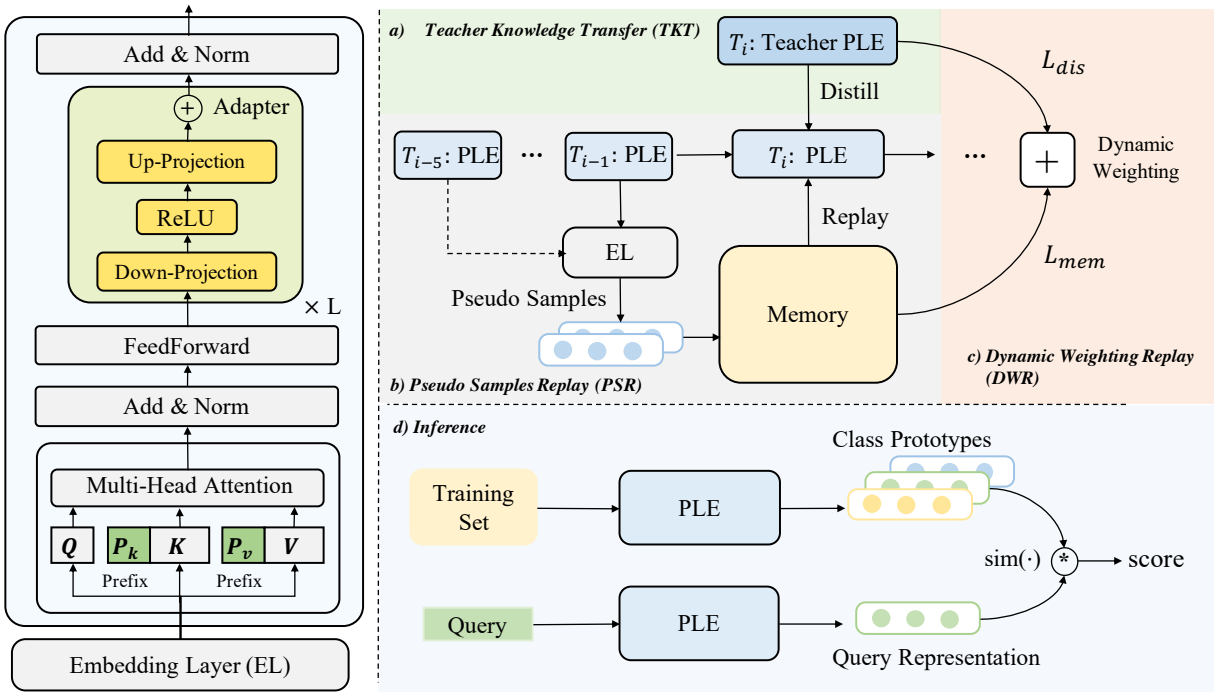

Figure 2: Overview of the proposed framework. The left side shows the structure of our PLE, which consists of a frozen PrLM and the Prefix (P_k, P_v) and Continual Adapter inserted into each layer. The trainable parameters are in green. The top right shows the process of learning for a new task $\mathcal{T}_i$ with three strategies: Teacher Knowledge Transfer (TKT), Pseudo Samples Replay (PSR), and Dynamic Weighting Replay (DWR). The lower right shows the distance-based classification pipeline at inference.

Summary. Existing works in few-shot intent detection mainly focus on the data scarcity and ignore the ability to learn consecutive tasks, which is essential for the online dialogue systems. The works in continual intent detection do not consider the data scarcity of emerging new intents. The newly proposed FSCIL-ID setting is also not aligned with the online dialogue systems. In contrast to those works, our work aims to recognize continually emerging new intents from multiple domains with very few labeled examples.

3 Methodology

3.1 Problem Formulation

In the CFID setting, given a sequence of n tasks $\{\mathcal{T}_1, \mathcal{T}_2, ..., \mathcal{T}_n\}$, each task $\mathcal{T}_i$ contains its own training set D^i_{train}, development set D^i_{dev}, and test set D^i_{test}. Each dataset D contains a series of samples $\{(x_i, y_i)\}|^{|D|}_{i=1}$, where y_i is the ground-truth intent class of the input utterance x_i. In particular, we describe its few-shot nature that there are only $K \in \{5, 10\}$ samples for each class in the training set. There are also a few (e.g., 10) samples for each class in the development set. This is because using a larger development set brings significant advantages and defeats the goal of few-shot learning (Gao et al., 2021). After learning $\mathcal{T}_i$, the model is evaluated separately on the test set of seen tasks. The setup is aligned with the real scenario, where the data privacy of different users is protected while the task information is available.

3.2 Overall Framework

As shown in Figure 2, the framework of the proposed method consists of one main module and three strategies of continual learning: 1) The lightweight PLE is responsible for extracting semantic features of the input utterances. 2) The TKT aims to transfer task-specific knowledge to the current model. 3) The PSR first selects two key samples per class and encodes them through the frozen embedding layer (EL) to generate pseudo samples and save them into the Memory. 4) The DWR is responsible for balancing the learning of new tasks and replaying past tasks.

3.3 Prefix-guided Lightweight Encoder (PLE)

PLE serves as the main module to alleviate catastrophic forgetting caused by over-parameterization. As a sub-module of PLE, the **Continual Adapter** is a full continual learning lightweight module

designed to capture knowledge across tasks and mitigate over-fitting by only tuning a small number of parameters, inspired by adapter-based tuning (Houlsby et al., 2019). More specifically, it consists of a down-projection with $W_{\text{down}} \in \mathbb{R}^{d\times r}$ to project the input hidden states $h \in \mathbb{R}^d$, followed by a nonlinear activation function $\text{ReLU}(\cdot)$, and an up-projection with $W_{\text{up}} \in \mathbb{R}^{r\times d}$, formally:

$$h \leftarrow h + \text{ReLU}(hW_{\text{down}})W_{\text{up}}. \tag{1}$$

Following (He et al., 2021), the adapter is inserted only after the feed forward layer of the transformer block. Note that the parameters of Continual Adapter are shared by each task and continually updated in the continual learning process while the PrLM is kept frozen.

To guide the frozen backbone in capturing task-specific knowledge dynamically, we further propose the **Prefix-guided Attention** mechanism inspired by prefix tuning (Li and Liang, 2021). It incorporates continuous prompts into the self-attention layer to guide the final self-attention flow. More specifically, we concatenate two sets of l tunable prefix vectors $P_k, P_v \in \mathbb{R}^{l\times d}$ to the keys and values of the multi-head attention at every layer. In this way, the computation of head_i is modified as:

$$\text{head}_i = \text{Attn}(XW_i^Q, [P_k^i, XW_i^K], [P_v^i, XW_i^V]), \tag{2}$$

where $X \in \mathbb{R}^{m\times d}$ is the input sequence representation and W_i^Q, W_i^K, and $W_i^V \in \mathbb{R}^{d\times d_h}$ are the parameter matrices. With the guidance of the prefix, the distribution of attention can be re-modulated dynamically in the continual learning process.

3.4 Teacher Knowledge Transfer (TKT)

TKT is to alleviate negative knowledge transfer across tasks, a phenomenon that impairs model performance on the current task. While most works (Ke et al., 2021a) design complicated dynamic architecture to encourage positive knowledge transfer, TKT can simply and explicitly distill task-specific knowledge into the model to compensate for the performance.

Concretely, we first train a teacher PLE individually. The parameters of prefix and adapters are randomly initialized to avoid transferring knowledge from past tasks and gain more task-specific knowledge from the current. Then, we transfer the task-specific knowledge into the continually learning model through knowledge distillation.

As for the teacher PLE f_θ^T, in each iteration, N classes are randomly selected from the label space, and then K samples are selected for the encoder to extract features. The obtained features are averaged for each class prototype: $\hat{y}_j = \frac{1}{K}\sum_{k=1}^{K} f_\theta^T(x_k)$. The teacher is optimized by minimizing the cross entropy loss L_{sim}, formally:

$$L_{\text{sim}} = -\sum_{i=1}^{N\times K}\sum_{j=1}^{N}\mathbb{I}(y_i = y_j)\times \log\frac{\exp(\text{sim}(f_\theta^T(x_i), \hat{y}_j)/\tau)}{\sum_{l=1}^{N}\exp(\text{sim}(f_\theta^T(x_i), \hat{y}_l)/\tau)}. \tag{3}$$

where $\text{sim}(\cdot)$ is the cosine similarity function and τ is a temperature hyper-parameter and $\mathbb{I}(\cdot)$ is the indicator function. To fully make use of $N \times K$ samples, we select one sample at a time from a class as a query and the rest of the samples as support samples to compute the prototype so that there are $N \times K$ times of nearest neighbor classification in parallel at each iteration.

As for the student PLE f_θ^S, it firstly inherits the previous knowledge by reusing the parameters of the last learned model. Then, it gains task-specific knowledge by training on the current task with knowledge distillation, formally:

$$L_{\text{dis}} = \sum_{i=1}^{N\times K} \|f_\theta^S(x_i) - f_\theta^T(x_i)\|. \tag{4}$$

3.5 Pseudo Samples Replay (PSR)

PSR is to consolidates previous knowledge in replaying-based ways. Concretely, after learning for new tasks, we first obtain the prototype feature of each class by averaging the features of all samples labeled as this class: $\hat{y}_j = \frac{1}{K}\sum_{k=1}^{K} f_\theta^S(x_k)$. Then we select the instance closest to the prototype of class as the most representative sample, and select the instance farthest to the prototype of class as the hardest sample. To avoid direct access to the raw texts for privacy, these two samples are encoded with the frozen PrLM to generate pseudo samples, whose embedding space is always not distorted during continual learning. Finally, we store the two samples in the memory for each class.

In this way, the goal of replaying can be achieved by storing a minimum number of samples (i.e, two samples per class). During replaying the pseudo samples, we randomly select N classes from the previous task and adopt the cross-entropy loss

L_{mem} to ensure intra-class compactness while increasing inter-class distances, formally:

$$L_{\text{mem}} = -\sum_{i=1}^{N\times 2}\sum_{j=1}^{N}\mathbb{I}(y_i = y_j)\times \log\frac{\exp(\text{sim}(f_\theta^S(x_i), \hat{y}_j)/\tau)}{\sum_{l=1}^{N}\exp(\text{sim}(f_\theta^S(x_i), \hat{y}_l)/\tau)}. \quad (5)$$

3.6 Dynamic Weighting Replay (DWR)

DWR is to find a good trade-off between learning new tasks and replaying. Due to the domain variety, it is hard to determine whether to replay more on old tasks (i.e., PSR) or distill more on new tasks (i.e., TKT). They can be regarded as two contradictory optimization objectives. It drives us to design DWR to dynamically decide the weights of the two objectives and get a Pareto optimal solution.

Concretely, we first randomly sample one previous task to replay at each iteration rather than all the previous tasks. Then, we adopt a Pareto-optimal weighting strategy (Sener and Koltun, 2018) inspired by multi-task learning. The learning weight of the sampled old task can be determined dynamically in each iteration. The total loss is defined as follows:

$$\begin{aligned} L &= \lambda_{\text{dis}} L_{\text{dis}} + \lambda_{\text{mem}} L_{\text{mem}}, \\ \lambda_{\text{dis}}, \lambda_{\text{mem}} &= \text{Pareto\_Solver}(L_{\text{dis}}, L_{\text{mem}}). \end{aligned} \quad (6)$$

The details of Pareto_Solver can be referred in Sener and Koltun (2018).

3.7 Inference

For a given utterance x in D_{test}^t, we calculate the similarity between the extracted feature of x and all class prototypes $\{\hat{y}_i\}$ in the t-th task and pick the one with the highest cosine similarity:

$$y^* = \arg\max_{\hat{y}_i \in \{\hat{y}_i\}} \text{Sim}(f_\theta^S(x), \hat{y}_i). \quad (7)$$

The prototype of class $\hat{y}_i$ can be obtained by averaging the features of training samples labeled as y_i through the current trained PLE.

4 Experiments

4.1 CFID Benchmark

As for the first work in CFID, we first collect nine popular intent detection datasets and arrange them in a fixed random order to construct the benchmark: CLINC150, ATIS, HWU64, BANKING77, MTOP, SNIPS, LEYZER, MSLU, and TOP. For each dataset, we randomly select $K = 5$ or 10 samples per class as a 5-shot or 10-shot training set and select 10 samples per class as a development set. Details of nine datasets are reported in Table 1.

Dataset	#Domain	#Class	#Train	#Dev	#Test
CLINC150	10	150	750/1500	1500	4500
ATIS	1	14	70/140	121	827
HWU64	18	64	320/640	640	1076
BANKING77	1	77	385/770	770	3080
MTOP	11	85	425/860	850	4354
SNIPS	7	7	35/70	70	1429
LEYZER	15	57	285/570	469	381
MSLU	3	12	60/120	120	7799
TOP	2	11	55/110	110	8196

Table 1: The statistics of datasets (5-shot/10-shot).

4.2 Evaluation Protocol

Following (Geng et al., 2021), we run all methods with the same task ordering during training. The test accuracy of each task is reported after all tasks are visited.

At time step t, following (Mehta et al., 2021), we employ the average accuracy A_t, average forgetting F_t and learning accuracy LA_t metrics after learning on the t-th task. Let $a_{t,i}$ denote the test accuracy on the task i after learning task t, those metrics are defined as follows:

$$\begin{aligned} A_t &= \frac{1}{t}\sum_{i=1}^{t} a_{t,i} \quad LA_t = \frac{1}{t}\sum_{i=1}^{t} a_{i,i} \\ F_t &= \frac{1}{t-1}\sum_{i=1}^{t-1}\max_{j\in\{1,\ldots,t-1\}}(a_{j,i} - a_{t,i}). \end{aligned} \quad (8)$$

A_t measures the average performance over all previously seen tasks. F_t measures how much the model has forgotten about all previously seen tasks after learning task t. LA_t measures the learning capability when the model sees the new task.

To measure the parameter efficiency, we also employ the following metrics: trainable parameters and storage parameters after learning n tasks.

4.3 Compared Methods

Since this is the first work in CFID, there is no prior method that solves exactly the same task. We extend the typical methods in the few-shot ID setting to the CFID setting to construct the following strong baselines.

- **Lifelong Classifier (LC)** consists of a pre-trained backbone and a task-specific classification layer. Each task shares a backbone and owns its specific layer.

Task ID	0	1	2	3	4	5	6	7	8	Avg.
Method	CLINC150	ATIS	HWU64	BANKING77	MTOP	SNIPS	LEYZER	MSLU	TOP	
LC	10.32/10.55	38.09/43.65	33.18/31.78	45.13/45.47	56.51/57.16	78.61/72.92	90.81/92.21	94.13/94.97	82.83/86.49	58.85/59.47
L-DNNC	83.85/85.82	71.46/81.74	74.38/78.13	63.91/71.40	**80.91/85.24**	**93.52**/93.52	92.65/95.28	95.17/**97.27**	88.14/**90.93**	82.67/86.59
L-PN	77.93/85.67	73.28/89.68	71.90/79.18	58.97/71.67	80.90/84.55	91.07/93.00	**93.00**/95.36	95.93/96.72	**88.98**/90.92	81.33/87.42
PN-AGEM	79.73/86.24	79.60/88.51	72.83/78.81	61.21/72.82	80.79/84.24	90.97/94.29	92.65/**95.63**	**96.15**/96.62	88.14/90.69	82.45/87.54
PLE (Ours)	**88.70***/**91.20***	**87.91***/**91.29***	**76.46***/**80.36**	**74.90***/**79.09***	76.14/80.64	93.40/**94.56**	89.68/91.16	95.06/96.18	88.27/88.91	**85.61/88.16**
PN-Joint	89.04/93.19	84.52/90.08	76.58/84.08	76.61/84.19	78.72/86.28	92.93/95.89	92.56/95.10	91.52/96.30	87.15/89.25	85.52/90.49

Table 2: Test accuracy (%) evaluated on the final model in 5-shot/10-shot regime after all 9 tasks are visited. We use Avg. to represent the average accuracy of all tasks for each method. * indicate statistically significant ($p < .05$) improvements over the best baseline.

Method	Avg. A_t	Avg. F_t	Avg. LA_t
LC	70.00/70.54	20.91/26.09	85.21/89.62
L-DNNC	84.22/87.32	4.91/4.54	**87.69**/90.54
L-PN	82.26/88.95	3.16/3.48	83.87/**91.44**
PN-AGEM	83.76/**89.03**	3.11/3.05	85.97/91.21
PLE (Ours)	**84.73**/88.10	**1.16/1.03**	85.49/88.79
SC	86.62/90.69	0.00/0.00	86.62/90.69
S-PN	86.90/90.76	0.00/0.00	86.90/90.76
S-PLE	86.93/90.48	0.00/0.00	86.93/90.48
PN-Joint	85.05/90.27	0.80/0.65	85.41/90.52

Table 3: Performance of different methods in 5-shot/10-shot regime. We use Avg. to All metrics are averaged over all time steps in three trials.

- **Lifelong DNNC (L-DNNC)**. DNNC (Zhang et al., 2020) is one of the state-of-the-art methods in the few-shot ID task, which solves it as a textual entailment problem and uses large-scale entailment datasets for pre-training. L-DNNC tunes the whole DNNC model in a sequential manner when a new task arrives.

- **Lifelong Prototypical Network (L-PN)**. Prototypical Network (PN) (Snell et al., 2017) is also a strong distance-based baseline for few-shot ID tasks. Lifelong PN (LPN) tunes the whole PN model during lifelong learning.

- **PN-AGEM**. We also compared with a strong replay-based lifelong learning method called AGEM (Chaudhry et al., 2019). It needs to maintain a memory for storing selected samples from previous tasks. We apply it to the prototypical network and get a variant referred to as PN-AGEM.

- **PN-Joint** stores all data from all seen previous tasks and trains the whole prototypical network with all data when learning the new task. It serves as an upper bound of the prototypical network.

We also test those baselines in a single-task setting to measure the knowledge transfer ability.

- **Single Classifier (SC)** trains one classifier for each task. Obviously, it suffer from serious parameter explosion problem when the number of tasks increasing.

- **Single Prototypical Network (S-PN)** trains one prototypical network for each task. It also suffers from the parameter explosion problem.

- **Single PLE (S-PLE)** is an extension of our PLE model, which trains one adapter with one prefix individually for each task.

4.4 Implementation Details.

We use a pre-trained model $\mathrm{SimCSE}_{\mathrm{base}}$ as the backbone for all experiments, because of its powerful text representation capabilities. For classifier-based experiments, the batch size is 4 and 8 in the 5/10-shot setting respectively. For all experiments except those using the PLE, the learning rate is 2e-5. For PLE, it is 1e-4. For the replay-based baseline, the memory size is the same as ours. For experiments with episode training, we chose N and K for each task based on the maximum memory capacity and ensured that the same values were used for each experiment. For DNNC, we follow the settings in Zhang et al. (2020).

4.5 Main Results

In this part, we report the test accuracy of each task, referred to as "Overall Performance" and provide more insights into the catastrophic forgetting and average performance at each time step, referred to as "Middle States Performance".

Overall Performance As shown in Table 2, we report the experimental results of our approach and baselines. From the results, we can observe that: **1)** Our proposed PLE outperforms previous baselines concerning the average accuracy of all tasks (85.61% and 88.16% for 5-shot and 10-shot settings), which demonstrates the effectiveness of our

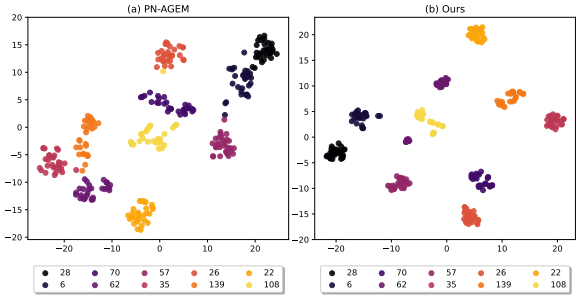

Figure 3: t-SNE visualization of PN-AGEM and Ours on the final model with test data of CLINC150. We randomly choose ten classes of the task to visualize.

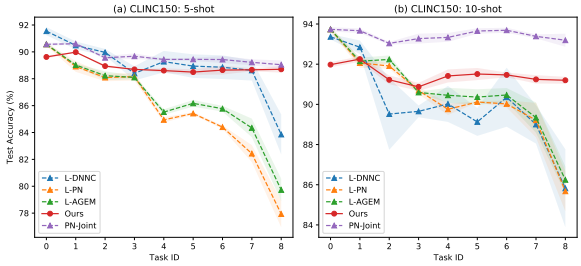

Figure 4: Test accuracy (%) of different methods on the CLINC150 dataset in 5-shot/10-shot CFID setting. Each curve denotes a kind of method. Shaded regions indicate standard deviation over three trials.

Method	#Trainable Params.	#Storage Params.
LC	125M+$\Delta\mathcal{T}_i$	125M+$\sum_{i=1}^{n}\Delta\mathcal{T}_i$
L-DNNC	125M	125M
L-PN	125M	125M
PN-AGEW	125M	125M
PLE (Ours)	**15M+15M=30M**	125M+15M=140M
SC	125M+$\Delta\mathcal{T}_i$	125M$\times n$ + $\sum_{i=1}^{n}\Delta\mathcal{T}_i$
S-PN	125M	125M$\times n$
S-PLE	15M	125M+15M$\times n$
PN-Joint	125M	125M

Table 4: Number of trainable and storage parameters in different methods. n denotes the number of tasks and $\Delta\mathcal{T}_i$ denotes the number of parameters of the task-specific layer. Here, the number of parameters of the PrLM and additional parameters of ours are 125M and 15M, respectively.

method. **2)** Our method still achieve better performance in the earlier accessed tasks, which demonstrates the superiority of our model in avoiding catastrophic forgetting.

In comparison, simply fine-tuning the backbone and the new involved classifier inevitably suffers from catastrophic forgetting. For example, the accuracy of LC on the first task is only 10.32%. We attribute it to the mismatch between the updated backbone and the classifier of the old task. For L-DNNC and L-PN, since they only have a shared encoder across tasks, catastrophic forgetting can be avoided. Thus they achieve 83.85% and 77.93% accuracy on the first task. Compared with L-PN, PN-AGEM is better on the early accessed tasks as it replays some samples of past tasks. As shown in Figure 3, compared with PN-AGEM, our PLE shows better intra-class compactness and larger inter-class distances.

For PN-Joint, it uses training data of all previously seen tasks at each step, which is more likely to be affected by negative knowledge transfer in a few cases. For tasks with very different domains, e.g., ATIS with flight domain, other tasks may transfer more negative knowledge to it. Thus, we can observe a worse performance than our method in this case (Accuracy of 90.08% vs. 91.29% on ATIS). It shows our effectiveness in alleviating this problem.

Middle States Performance As shown in Table 3, we report the average accuracy, forgetting, and learning accuracy of our method and baselines. All metrics are averaged over all time steps. From the results, we can observe that: Our proposed PLE outperforms previous baselines in the 5-shot CFID setting concerning Avg. A_t and Avg. F_t. It also has competitive performance in the 10-shot CFID setting and less forgetting than other baselines. For baselines in the single-task setting (i.e., SC, S-PN, and S-PLE), although they perform well, when the number of tasks is large, they inevitably suffer from parameter explosion.

However, there is a slight drop in Avg. LA_t in our method compared to others. A similar phenomenon can be observed in Table 2, i.e., for tasks newly visited, the test accuracy is not as good as other methods. We attribute it to a trade-off between learning about new tasks and preventing forgetting of past tasks. Freezing the backbone in our method damages the expressiveness but guarantees the overall performance of all tasks. Specifically, taking the earliest visited task CLINC150 as an example, Figure 4 shows the accuracy curves of the different methods throughout the continual learning. Compared with other methods, the performance of our method is relatively stable in the whole process, although the performance is not the best at the beginning.

Overall, our proposed PLE is a promising solution for the CFID problem with less forgetting and comparable performance.

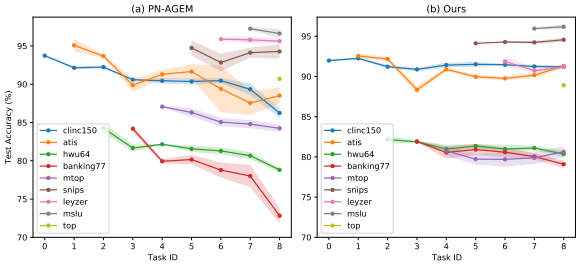

Figure 5: Test accuracy (%) of different tasks between PN-AGEM and Ours. Each curve in the sub-figure denotes a kind of task in the 10-shot setting. Shaded regions indicate standard deviation over three trials.

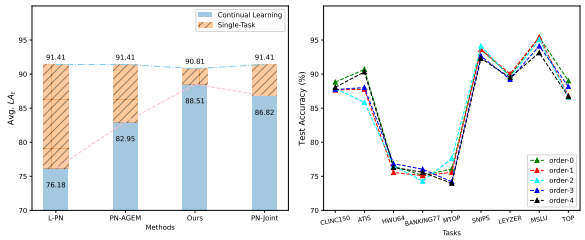

Figure 6: Left side: Avg. LA_t on ATIS in the 5-shot CFID and single-task settings, respectively. Right side: Test accuracy (%) of different tasks in the 5-shot setting with five different task orderings.

4.6 Efficiency & Robustness of PLE

Parameter Efficiency As shown in Table 4, we list the number of trainable and stored parameters in different methods. As the number of tasks increases, the stored parameters of the baselines in the single-task setting (i.e., SC, S-PN, and S-PLE) also increase, eventually leading to the explosion problem. Compared to them and other baselines, PLE achieves competitive performance and parameter efficiency with 76% less trainable parameters (from 125M to 30M) and only 15M additional parameter storage. As a result, it is possible to employ a larger pre-trained language model to achieve better performance.

Training and Inference Efficiency We observe that incorporating continuous prompts into PLE does not suffer from too slower training than other prototypical-based baselines (i.e., L-PN and PN-AGEM). However, for L-DNNC, despite its high performance in Avg. LA_t, it makes predictions by enumerating all the labels to decide whether a query and a label match or not, which is so time-consuming during training and inference.

Robustness for Task Ordering To analyze the effect of task ordering when PLE is learning different tasks, we randomly sample five different task orderings in the 5-shot setting. After all tasks are learned, we report the test accuracy over different orderings. As shown in the the right side of Figure 6, we can see our method is insensitive to different task orderings.

4.7 Knowledge Transfer Assessment

Assessing Backward Knowledge Transfer To assess the influence of learning new tasks on the performance of previous tasks (**backward transfer**), we visualize the curve of the test accuracy of different tasks between our method and the baseline with the lowest Avg. F_t of 3.05%, i.e., PN-AGEM in the 10-shot setting. As seen in the left side of Figure 5, the curve of PN-AGEM has a clear downward trend, while the curve of our method remains stable overall. In particular, we observe that on ATIS, the performance of PN-AGEM continues to decline, while our method goes through a phase of slight decline followed by an increase. It shows a promising ability to backward knowledge transfer.

Assessing Forward Knowledge Transfer To further assess the capability to learn new tasks with the help of knowledge from past tasks (**forward transfer**), we compare the results of Avg. LA_t in the single-task setting and continual learning setting. From Table 3, we observed that there is a significant drop in the extremely few-shot (i.e., 5-shot) regime. For example, there is a drop from 86.90 to 83.87 comparing S-PN and L-PN. In particular, there is still a drop of 1.49% compared to S-PN for the upper-bound baseline PN-Joint. It confirms the existence of negative knowledge transfer across tasks. We select the task ATIS to assess forward knowledge transfer, which is most affected by negative knowledge transfer (4% performance drop comparing PN-Joint with S-PN). From the left-top side of Figure 6, we can see our method has the highest Avg. LA_t and is closest to the performance in the single-task setting. It shows that our approach effectively reduces the effect of the negative knowledge transfer and enhances the effect of the forward knowledge transfer.

4.8 Analysis of Domain Variety

To simulate a realistic continual setting, we collect as many public datasets as possible, a few of which inevitably overlap in intent classes and domains, such as MSLU and MTOP. However, we count the number of similar domains in any two datasets and

Method	Avg. A_t	Avg. F_t	Avg. LA_t
Ours	**85.15/88.22**	0.99/0.83	85.79/88.75
w/o prefix	84.74/87.81	2.55/2.25	86.49/89.37
w/o memory	83.22/87.45	4.86/3.39	**86.55/89.83**
w/o TKT	82.35/87.25	**0.84/0.81**	82.84/87.82
w/o PSR	84.06/87.86	1.85/1.35	85.33/88.83
w/o DWR	84.63/87.88	1.11/1.43	85.63/88.59

Table 5: Ablation results in 5-shot/10-shot regime. All metrics are averaged over all time steps.

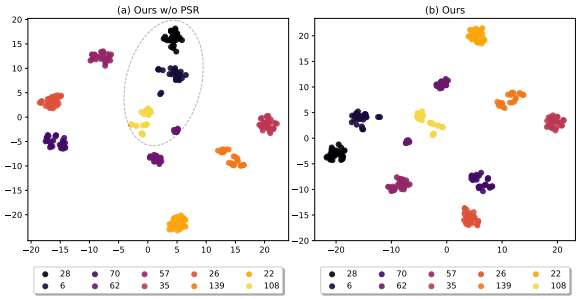

Figure 7: t-SNE visualization of ablation: w/o PSR and Ours on the final model with test data of CLINC150. The circle region shows our PLE has larger inter-class distances than the ablation variant.

find less than 1 domain overlap on average. There are also domain differences in some overlapping classes, such as the class "exchange_rate" in the "banking" and "travel" domain. It reflects the domain variety between tasks to some extent.

In general, our approach works well in very different domains. For example, the task ATIS belongs to the "flight" domain, which is different from all domains in another task, CLINC150. When continually learning ATIS after CLINC150, we observed there is a huge performance drop using L-PN baseline compared to S-PN in a single-task setting (accuracy of ATIS from 91% to 76%). This is mostly due to negative knowledge transfer caused by the domain gap between CLINC150 and ATIS. With the distillation strategy, our method alleviates this problem and achieves an accuracy of 89% when continually learning ATIS.

4.9 Ablation Study

As reported in Table 5, we conduct ablation studies to investigate the impact of different components of PLE in the 5-shot and 10-shot setting.

Specifically, we analyze the following variants: a) *w/o prefix* removes the prefix from the PLE. b) *w/o memory* removes the memory and merely adopting the TKT strategy. c) *w/o TKT* discards the TKT strategy and merely adopting the L_{sim} with the memory. d) *w/o PSR* randomly selects the same number of saved samples instead of using the PSR strategy. e) *w/o DWR* sets fixed weight hyper-parameters heuristically ($\lambda_{\text{dis}} = 0.9, \lambda_{\text{mem}} = 0.1$) instead of dynamically weighting.

From the results in Table 5, we can make the following observations. First, the introduction of prefix improves the performance of our method. Second, the variant without the TKT has a significant drop in performance on the Avg. A_t and Avg. LA_t. It confirms the existence of negative knowledge transfer across tasks. Using the TKT to gain more task-specific knowledge can effectively alleviate this problem. Also, the memory can effectively alleviate the catastrophic forgetting problem. Figure 7 shows that the PSR strategy is an efficient way to select saved samples.

Moreover, we observe that the Avg. LA_t and Avg. F_t are the highest in the "w/o memory" and "TKT" settings, respectively. It confirms a trade-off between learning new tasks and replaying past tasks. From the results in the "w/o DWR" setting, we can see the DWR strategy can effectively balance them and significantly improve the performance of our method.

5 Conclusion

In this paper, we define a more challenging yet practical problem as Continual Few-shot Intent Detection (CFID), where the system needs to handle continually emerging new intents with very few labeled data. To deal with the problem, we propose a novel prefix-guided lightweight encoder with three auxiliary strategies. Extensive experiments demonstrate the effectiveness and efficiency of our method in preventing catastrophic forgetting and encouraging positive knowledge transfer across tasks.

Acknowledgments

This work was supported by the NSFC projects (No. 62072399, No. U19B2042, No. 61402403), Chinese Knowledge Center for Engineering Sciences and Technology, MoE Engineering Research Center of Digital Library, Alibaba Group, Alibaba-Zhejiang University Joint Research Institute of Frontier Technologies, and the Fundamental Research Funds for the Central Universities (No. 226-2022-00070)

References

Arslan Chaudhry, Marc'Aurelio Ranzato, Marcus Rohrbach, and Mohamed Elhoseiny. 2019. Efficient lifelong learning with a-gem. In *ICLR*.

Li Cui, Deqing Yang, Jiaxin Yu, Chengwei Hu, Jiayang Cheng, Jingjie Yi, and Yanghua Xiao. 2021. Refining sample embeddings with relation prototypes to enhance continual relation extraction. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing, ACL/IJCNLP 2021, (Volume 1: Long Papers), Virtual Event, August 1-6, 2021*, pages 232–243. Association for Computational Linguistics.

Thomas Dopierre, Christophe Gravier, and Wilfried Logerais. 2021. Protaugment: Intent detection meta-learning through unsupervised diverse paraphrasing. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing, ACL/IJCNLP 2021, (Volume 1: Long Papers), Virtual Event, August 1-6, 2021*, pages 2454–2466. Association for Computational Linguistics.

Tianyu Gao, Adam Fisch, and Danqi Chen. 2021. Making pre-trained language models better few-shot learners. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 3816–3830, Online. Association for Computational Linguistics.

Binzong Geng, Min Yang, Fajie Yuan, Shupeng Wang, Xiang Ao, and Ruifeng Xu. 2021. Iterative network pruning with uncertainty regularization for lifelong sentiment classification. In *SIGIR '21: The 44th International ACM SIGIR Conference on Research and Development in Information Retrieval, Virtual Event, Canada, July 11-15, 2021*, pages 1229–1238. ACM.

Chih-Wen Goo, Guang Gao, Yun-Kai Hsu, Chih-Li Huo, Tsung-Chieh Chen, Keng-Wei Hsu, and Yun-Nung Chen. 2018. Slot-gated modeling for joint slot filling and intent prediction. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 2 (Short Papers)*, pages 753–757.

Xu Han, Yi Dai, Tianyu Gao, Yankai Lin, Zhiyuan Liu, Peng Li, Maosong Sun, and Jie Zhou. 2020. Continual relation learning via episodic memory activation and reconsolidation. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, ACL 2020, Online, July 5-10, 2020*, pages 6429–6440. Association for Computational Linguistics.

Junxian He, Chunting Zhou, Xuezhe Ma, Taylor Berg-Kirkpatrick, and Graham Neubig. 2021. Towards a unified view of parameter-efficient transfer learning. *CoRR*, abs/2110.04366.

Neil Houlsby, Andrei Giurgiu, Stanislaw Jastrzebski, Bruna Morrone, Quentin de Laroussilhe, Andrea Gesmundo, Mona Attariyan, and Sylvain Gelly. 2019. Parameter-efficient transfer learning for NLP. In *Proceedings of the 36th International Conference on Machine Learning, ICML 2019, 9-15 June 2019, Long Beach, California, USA*, volume 97 of *Proceedings of Machine Learning Research*, pages 2790–2799. PMLR.

Zixuan Ke, Bing Liu, Nianzu Ma, Hu Xu, and Lei Shu. 2021a. Achieving forgetting prevention and knowledge transfer in continual learning. pages 22443–22456.

Zixuan Ke, Bing Liu, Hu Xu, and Lei Shu. 2021b. CLASSIC: continual and contrastive learning of aspect sentiment classification tasks. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing, EMNLP 2021, Virtual Event / Punta Cana, Dominican Republic, 7-11 November, 2021*, pages 6871–6883. Association for Computational Linguistics.

Zixuan Ke, Hu Xu, and Bing Liu. 2021c. Adapting BERT for continual learning of a sequence of aspect sentiment classification tasks. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT 2021, Online, June 6-11, 2021*, pages 4746–4755. Association for Computational Linguistics.

Stefan Larson, Anish Mahendran, Joseph J. Peper, Christopher Clarke, Andrew Lee, Parker Hill, Jonathan K. Kummerfeld, Kevin Leach, Michael A. Laurenzano, Lingjia Tang, and Jason Mars. 2019. An evaluation dataset for intent classification and out-of-scope prediction. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*.

Xiang Lisa Li and Percy Liang. 2021. Prefix-tuning: Optimizing continuous prompts for generation. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing, ACL/IJCNLP 2021, (Volume 1: Long Papers), Virtual Event, August 1-6, 2021*, pages 4582–4597. Association for Computational Linguistics.

Qingbin Liu, Xiaoyan Yu, Shizhu He, Kang Liu, and Jun Zhao. 2021. Lifelong intent detection via multi-strategy rebalancing. *CoRR*, abs/2108.04445.

Qiaoyang Luo, Lingqiao Liu, Yuhao Lin, and Wei Zhang. 2021. Don't miss the labels: Label-semantic augmented meta-learner for few-shot text classification. In *Findings of the Association for Computational Linguistics: ACL/IJCNLP 2021, Online Event, August 1-6, 2021*, volume ACL/IJCNLP 2021 of

Findings of ACL, pages 2773–2782. Association for Computational Linguistics.

Shikib Mehri and Mihail Eric. 2021. Example-driven intent prediction with observers. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT 2021, Online, June 6-11, 2021*, pages 2979–2992. Association for Computational Linguistics.

Shikib Mehri, Mihail Eric, and Dilek Hakkani-Tür. 2020. Dialoglue: A natural language understanding benchmark for task-oriented dialogue. *CoRR*, abs/2009.13570.

Sanket Vaibhav Mehta, Darshan Patil, Sarath Chandar, and Emma Strubell. 2021. An empirical investigation of the role of pre-training in lifelong learning. *CoRR*, abs/2112.09153.

Chengwei Qin and Shafiq Joty. 2022. Continual few-shot relation learning via embedding space regularization and data augmentation. *CoRR*, abs/2203.02135.

Libo Qin, Wanxiang Che, Yangming Li, Haoyang Wen, and Ting Liu. 2019. A stack-propagation framework with token-level intent detection for spoken language understanding. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing, EMNLP-IJCNLP 2019, Hong Kong, China, November 3-7, 2019*, pages 2078–2087. Association for Computational Linguistics.

Libo Qin, Tianbao Xie, Wanxiang Che, and Ting Liu. 2021. A survey on spoken language understanding: Recent advances and new frontiers. In *IJCAI*.

Ozan Sener and Vladlen Koltun. 2018. Multi-task learning as multi-objective optimization. In S. Bengio, H. Wallach, H. Larochelle, K. Grauman, N. Cesa-Bianchi, and R. Garnett, editors, *Advances in Neural Information Processing Systems 31*, pages 525–536. Curran Associates, Inc.

Jake Snell, Kevin Swersky, and Richard Zemel. 2017. Prototypical networks for few-shot learning. In *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc.

Chengyu Wang, Haojie Pan, Yuan Liu, Kehan Chen, Minghui Qiu, Wei Zhou, Jun Huang, Haiqing Chen, Wei Lin, and Deng Cai. 2021. Mell: Large-scale extensible user intent classification for dialogue systems with meta lifelong learning. In *KDD '21: The 27th ACM SIGKDD Conference on Knowledge Discovery and Data Mining, Virtual Event, Singapore, August 14-18, 2021*, pages 3649–3659. ACM.

Congying Xia, Wenpeng Yin, Yihao Feng, and Philip S. Yu. 2021. Incremental few-shot text classification with multi-round new classes: Formulation, dataset and system. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT 2021, Online, June 6-11, 2021*, pages 1351–1360. Association for Computational Linguistics.

Fajie Yuan, Guoxiao Zhang, Alexandros Karatzoglou, Joemon M. Jose, Beibei Kong, and Yudong Li. 2021. One person, one model, one world: Learning continual user representation without forgetting. In *SIGIR '21: The 44th International ACM SIGIR Conference on Research and Development in Information Retrieval, Virtual Event, Canada, July 11-15, 2021*, pages 696–705. ACM.

Haode Zhang, Yuwei Zhang, Li-Ming Zhan, Jiaxin Chen, Guangyuan Shi, Xiao-Ming Wu, and Albert Y. S. Lam. 2021a. Effectiveness of pre-training for few-shot intent classification. In *Findings of the Association for Computational Linguistics: EMNLP 2021, Virtual Event / Punta Cana, Dominican Republic, 16-20 November, 2021*, pages 1114–1120. Association for Computational Linguistics.

Jianguo Zhang, Trung Bui, Seunghyun Yoon, Xiang Chen, Zhiwei Liu, Congying Xia, Quan Hung Tran, Walter Chang, and Philip S. Yu. 2021b. Few-shot intent detection via contrastive pre-training and fine-tuning. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing, EMNLP 2021, Virtual Event / Punta Cana, Dominican Republic, 7-11 November, 2021*, pages 1906–1912. Association for Computational Linguistics.

Jianguo Zhang, Kazuma Hashimoto, Wenhao Liu, Chien-Sheng Wu, Yao Wan, Philip S. Yu, Richard Socher, and Caiming Xiong. 2020. Discriminative nearest neighbor few-shot intent detection by transferring natural language inference. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing, EMNLP 2020, Online, November 16-20, 2020*, pages 5064–5082. Association for Computational Linguistics.

"Mama Always Had a Way of Explaining Things So I Could Understand": A Dialogue Corpus for Learning to Construct Explanations

Henning Wachsmuth *
Department of Computer Science
Paderborn University
henningw@upb.de

Milad Alshomary *
Department of Computer Science
Paderborn University
milad.alshomary@upb.de

Abstract

As AI is more and more pervasive in everyday life, humans have an increasing demand to understand its behavior and decisions. Most research on explainable AI builds on the premise that there is one ideal explanation to be found. In fact, however, everyday explanations are co-constructed in a dialogue between the person explaining (the explainer) and the specific person being explained to (the explainee). In this paper, we introduce a first corpus of dialogical explanations to enable NLP research on how humans explain as well as on how AI can learn to imitate this process. The corpus consists of 65 transcribed English dialogues from the Wired video series *5 Levels*, explaining 13 topics to five explainees of different proficiency. All 1550 dialogue turns have been manually labeled by five independent professionals for the topic discussed as well as for the dialogue act and the explanation move performed. We analyze linguistic patterns of explainers and explainees, and we explore differences across proficiency levels. BERT-based baseline results indicate that sequence information helps predicting topics, acts, and moves effectively.

1 Introduction

Explaining is one of the most pervasive communicative processes in everyday life, aiming for mutual understanding of the two sides involved. Parents explain to children, doctors to patients, teachers to students, seniors to juniors—or all the other way round. In explaining dialogues, one side takes the role of the *explainer*, the other the role of the *explainee*. Explainers seek to enable explainees to comprehend a given topic to a certain extent or to perform some action related to it (Rohlfing et al., 2021). This usually implies a series of dialogue turns where both sides request and provide different information about the topic. In line with the quote from the movie "Forrest Gump" in the title, how an explaining dialogue looks like is strongly affected by the specific explainer and explainee as well as by their interaction.

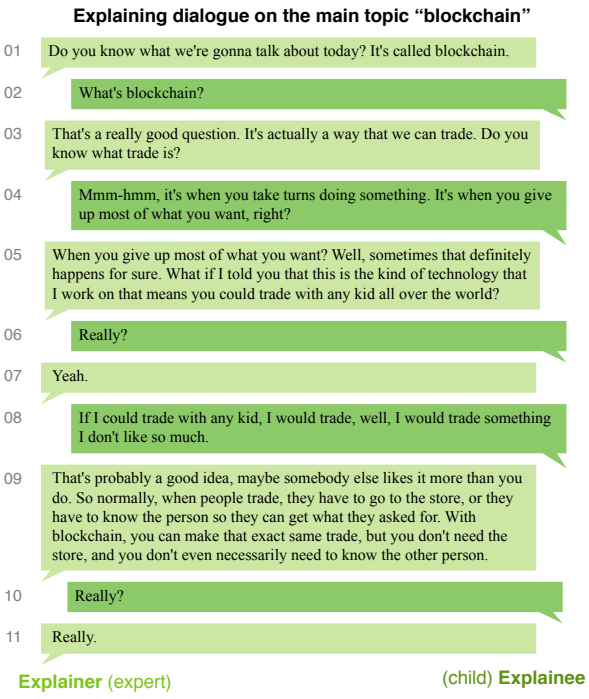

Figure 1: A short explaining dialogue from the video series *5 Levels*, included in the corpus presented in Section 3. Here, an expert explains blockchain to a child.

Consider the dialogue in Figure 1, where a technology expert explains the basic idea of blockchain to a 5-year old in a controlled setting. Beyond the explanations of the main topic (turns 05 and 09), the dialogue contains an explanation request (02), a test of prior knowledge (03), explanations from the explainee (04), and more. We observe that the explainer's explanations depend on the reaction of the explainee and that their level of depth is most likely adjusted to the explainee's proficiency.

The importance of studying how to explain has become apparent with the rise of research on explainable artificial intelligence, XAI (Barredo Arrieta et al., 2020). As AI finds its way into various

* Both authors contributed equally to this paper.

Proceedings of the 29th International Conference on Computational Linguistics, pages 344–354
October 12–17, 2022.

aspects of work and private life, humans interacting with respective systems, or being affected by them, have an increasing demand to understand their behavior and decisions. This demand has also been manifested in a *right to explanation* within the EU's General Data Protection Regulation (Goodman and Flaxman, 2017). Prior work on XAI largely starts from the premise that an ideal (monological) explanation exists for any behavior or decision, possibly dependent on the explainee at hand (Miller, 2019). According to Rohlfing et al. (2021), however, real explainability must account for the co-constructive nature of explaining emerging from interaction.

In natural language processing, early work modeled discourse structure of monological explanations (Bourse and Saint-Dizier, 2012), and a number of recent approaches generate respective explanations for XAI (Situ et al., 2021) and recommendation (Li et al., 2021). In contrast, the language of dialogical explanations is still understudied (details in Section 2). We argue that a better understanding of how humans explain in dialogues is needed, so that XAI can learn to interact with humans.

In this paper, we present a first corpus for computational research on how to explain in dialogues (Section 3). The corpus has been created as part of a big interdisplinary research project dealing with the construction of explainability.[1] It consists of 65 transcribed dialogical explanations from the American video series *5 Levels* freely published by the Wired magazine.[2] Five dialogues each refer to one of 13 science-related topics (e.g., "blockchain" or "machine learning"). They have the same explainer (an expert on the topic), but differ in the explainee's proficiency (from child to colleague).

To enable XAI to mimic human explainers, it has to learn what turn to make at any point in a dialogue. In discussion with humanities researchers, we model a turn for this purpose by the relation of its *topic* to the main topic (e.g., subtopic or related topic), its *dialogue act* (e.g., check question or informing statement), and its *explanation move* (e.g., testing prior knowledge or providing an explanation). We segmented the dialogues into a total of 1550 turns, and we let five independent professionals annotate each turn for these three dimensions.

In Section 4, we analyze linguistic patterns of explaining dialogues in the annotated corpus. We find clear signals for the explainer's alignment to the explainee's proficiency, such as the avoidance of deviating to related topics towards children. The roles of explainer and explainee are reflected in the varying use of dialogue acts and explanation moves, possibly stressed by the given setting.

To obtain baselines for the prediction of the three annotated dimensions, we evaluate three variants of BERT (Devlin et al., 2019) in 13-topic cross-validation on the corpus (Section 5). Our results reveal that modeling sequential dialogue interaction helps predicting a turn's topic, act, and move effectively. Improvements seem still possible, calling for more sophisticated approaches as well as for more explaining dialogue data in the future.[3]

In summary, the contributions of our paper are:

1. A manually annotated corpus for studying how humans explain in dialogical settings
2. Empirical insights into how experts explain to explainees of different proficiency levels
3. Baselines for predicting the topic, dialogue act, and explanation move of dialogue turns

2 Related Work

Explainable AI (XAI) largely focuses on the interpretability of learned models from the perspective of scientific completeness (Gilpin et al., 2018). Even though recent works tackle cognitive aspects, such as the trade-off between completeness and compactness (Confalonieri et al., 2019), Miller (2019) pointed out that this perspective is far away from the understanding of everyday explanations in the social sciences. Garfinkel (2009) argues that the key is to sort out what the explainer should actually explain, and Barredo Arrieta et al. (2020) stressed the importance of who is the explainee for XAI. Rohlfing et al. (2021) built on these works, but reasoned that explanations can only be successful in general, if they are co-constructed in interaction between explainer and explainee. The rationale is that explainees vary in their motives and needs, and they face different challenges (Finke et al., 2022). The corpus we present serves as a basis for studying the linguistic aspects of the explainer-explainee interaction computationally.

Natural language language processing (NLP) has notably dealt with the related genre of instructional texts, modeling their structure (Fontan and Saint-Dizier, 2008), extracting knowledge (Zhang et al.,

[1] Constructing Explainability, https://trr318.upb.de/en
[2] 5 Levels, https://www.wired.com/video/series/5-levels
[3] The corpus and the experiment code are freely available here: https://github.com/webis-de/COLING-22

2012), comprehending some meaning (Yagcioglu et al., 2018), or generating them (Fried et al., 2018). However, instructional text has a clear procedural style with distinctive surface features (Vander Linden, 1992), unlike explanations in general. For tutorial applications, Jordan et al. (2006) extracted concepts from explanation sentences, whereas Jansen et al. (2016) studied the knowledge needed for scientific explanations, and Son et al. (2018) identified causal explanations in social media. Towards a computational understanding of explaining, Bourse and Saint-Dizier (2012) modeled explanation structure with discourse relations (Mann and Thompson, 1988). In XAI and recommendation contexts, the generation of respective explanations is explored increasingly (Situ et al., 2021; Li et al., 2021).

However, our main goal is not to understand how to generate an explanation, but to model how people interact in an explanation process. For annotation, we thus rely on the widely accepted concept of dialogue acts (Stolcke et al., 2000; Bunt et al., 2010). Similar has been done for deliberative dialogues by Al Khatib et al. (2018). In addition, we model the *moves* that explainers and explainees make in their interaction, adapting the idea of rhetorical moves, in terms of communicative functions of text segments used to support the communicative objective of a full text (Swales, 1990). Wachsmuth and Stein (2017) proposed task-specific moves for monological arguments, but we are not aware of any work on moves for explanations, nor for dialogical settings.

Hence, we start by compiling data in this paper. Existing related corpora contain tutorial feedback for explanation questions (Dzikovska et al., 2012), answers to non-factoid questions (Dulceanu et al., 2018), and pairs of questions and responses from community question answering platforms (Nakov et al., 2017). Finally, the corpus of Fan et al. (2019) includes 270k threads from the Reddit forum *Explain like I'm Five* where participants explain a concept asked for in simple ways. While all these allow for in-depth analyses of linguistic aspects of explanations, none of them include explaining dialogues with multiple turns on each side. This is the gap we fill with the corpus that we introduce.

3 Data

This section introduces the corpus that we created to enable computational research on dialogical explanation processes of humans. We discuss our design choices with respect to the source and annotation, and we present detailed corpus statistics.

3.1 Explaining Dialogues on Five Levels

As source data, we decided to rely on explaining dialogues from a controlled setting in which two people explicitly meet to talk about a topic to be explained. While we thereby may miss some interaction behavior found in real-word explanation processes, we expect that such a setting best exhibits explaning dialogue features in their pure form.

In particular, we acquired the source dialogues in our corpus from *5 Levels*, an American online video series published by the Wired magazine. In each video of the series, one explainer explains a science-related or technology-related topic to five different explainees. The explainer is always an expert on the topic, whereas the explainees increase in terms of (assumed) proficiency on the topic:

1. a *child*,
2. a *teenager*,
3. an *undergrad* college sudent,
4. a *grad* student, and
5. a *colleague* in terms of another expert.

Every video starts with a few introductory words by the expert, before one dialogue follows the other.[4] Transcriptions are already provided in the videos' captions. So far, the first season of the series is available with a total of 17 videos. Table 1 lists all explained topics (*main topics* henceforth) in these videos, along with explainer information.

At the time of starting the annotation process discussed below, only 14 of the 17 videos had been accessible, and one of these had partly corrupted subtitles. We thus restricted the annotated corpus to the remaining 13 videos, summing up to 65 dialogues that correspond to a video length of 5.35 hours. Later, we added all dialogues from the other four videos in unannotated form to the corpus.

Before annotation, we manually segmented each dialogue into its single turns, such that consecutive turns in a dialogue alternate between explainer and explainee. Overall, the 65 dialogues consist of 1550 turns (23.8 turns per dialogue on average), 790 from explainers and 760 from explainees. The turns span 51,344 words (33.1 words per turn). On

[4] It is noteworthy that the videos seem to have been cut a little, likely for the sake of a concise presentation. We assume that this mainly removed breaks between dialogue turns only. While it limits studying non-verbal interaction in explaining, the effect for textual analyses of the dialogues should be low.

#	Topic	Explainer	Expertise
1	Harmony	Jacob Collier	Musician
2	Blockchain	Bettina Warburg	Political scientist
3	Virtual reality	John Carmack	Oculus CTO
4	Connectome	Bobby Kasthuri	Neuroscientist
5	Black holes	Varoujan Gorjian	NASA astronomer
6	Lasers	Donna Strickland	Professor
7	Sleep	Aric A. Prather	Sleep scientist
8	Dimensions	Sean Carroll	Theoret. physicist
9	Gravity	Janna Levin	Astrophysicist
10	Computer hacking	Samy Kamkar	Security researcher
11	Nanotechnology	George Tulevski	Nanotec. researcher
12	Origami	Robert J. Lang	Physicist
13	Machine learning	Hilary Mason	Hidden Door CEO
14	CRISPR	Neville Sanjana	Biologist
15	Memory	Daphna Shohamy	Neuroscientist
16	Zero-knowl. proof	Amit Sahai	Computer scientist
17	Black holes	Janna Levin	Astrophysicist

Table 1: All 17 main topics explained in the *5 Levels* dialogues, along with the explainers and their expertise. The 65 dialogues of the 13 topics listed in black are annotated in our corpus; the rest is provided unannotated.

average, an explainer's turn is double as long as an explainee's turn (43.7 vs. 22.1 words). While the general data size is not huge, we provide evidence in Sections 4 and 5 that it suffices to find common patterns of explanation processes. Limitations emerging from the size are discussed in Section 6.[5]

3.2 Annotations of Explanatory Interactions

The corpus is meant to provide a starting point for XAI systems that mimic the explainer's role within dialogical explanation processes. Our annotation scheme supports this purpose and is the result of extensive discussions in our interdisciplinary project with a big team of computer scientists, linguists, psychologists, and cognitive scientists. Where possible, we followed the literature, but the lack of research on human interaction in explaining (see Section 2) made us extend the state of the art in different respects.

In particular, we focus on turn-level category labels that capture the basic behavior of explainers and explainees in explaining dialogues. Our scheme models the three dimensions of dialogue turns that we agreed on to be needed for a computational understanding of the behavior:

- the relation of a turn's *topic* to the main topic,
- the *dialogue act* performed in the turn, and
- the *explanation move* made through the turn.

[5]We also extracted the time code (start and end milliseconds) of each segment from the videos, for which one caption is shown. This may serve multimodal studies in the future.

We discuss the labels considered for each of the three annotation dimensions in the following. Since all labels apply to both explainer and explainee in principle, we refer to a speaker and a listener below.

Topic Even though the dialogues we target have one defined main topic to be explained, what is explained in specific turns may vary due to the dynamics of explaining interaction (Garfinkel, 2009). Since we seek to learn how to explain in general rather than any specificities of the concrete 13 main topics in the corpus, we abstract from the latter, modeling only the relation of the topic discussed in a turn to the dialogue's main topic. In particular, a turn's topic may be annotated as follows:

- t_1 *Main topic.* The main topic to be explained;
- t_2 *Subtopic.* A specific aspect of the main topic;
- t_3 *Related topic.* Another topic that is related to the main topic;
- t_4 *No/Other topic.* No topic, or another topic that is unrelated to the main topic.

Dialogue Act To model the communicative functions of turns in dialogues, we follow the literature (Bunt et al., 2010), starting from the latest version of the ISO standard taxonomy of dialogue acts.[6] In explaining, specific dialogue acts are in the focus, though. In collaboration with the interdisciplinary team, we selected a subset of 10 acts that capture communication on a level of detail that is specific enough to distinguish key differences, but abstract enough to allow finding recurring patterns:

- d_1 *Check question.* Asking a check question;
- d_2 *What/How question.* Asking a what question or a how question of any kind;
- d_3 *Other question.* Asking any other question;
- d_4 *Confirming answer.* Answering a question with confirmation;
- d_5 *Disconfirming answer.* Answering a question with disconfirmation;
- d_6 *Other answer.* Giving any other answer;
- d_7 *Agreeing statement.* Conveying agreement on the last utterance of the listener;
- d_8 *Disagreeing statement.* Conveying disagreement accordingly;
- d_9 *Informing statement.* Providing information with respect to the topic stated in the turn;
- d_{10} *Other.* Performing any other dialogue act.

[6]DIT++ Taxonomy of Dialogue Acts, https://dit.uvt.nl

Explanation Move Finally, we aim to understand the explanation-specific moves that explainers and explainees make to work together towards a successful explanation process. Due to the lack of models of explaining dialogues (see Section 2, we started from recent theory of explaining (Rohlfing et al., 2021). Based on a first inspection of a corpus sample, we established a set of 10 explanation moves that a speaker may make in the process, at a granularity similar to the dialogue acts:[7]

- e_1 *Test understanding.* Checking whether the listener understood what was being explained;
- e_2 *Test prior knowledge.* Checking the listener's prior knowledge of the turn's topic;
- e_3 *Provide explanation.* Explaining any concept or a topic to the listener;
- e_4 *Request explanation.* Requesting any explanation from the listener;
- e_5 *Signal understanding.* Informing the listener that their last utterance was understood;
- e_6 *Signal non-understanding.* Informing the listener that the utterance was not understood;
- e_7 *Providing feedback.* Responding qualitatively to an utterance by correcting errors or similar;
- e_8 *Providing assessment.* Assessing the listener by rephrasing their utterance or giving a hint;
- e_9 *Providing extra info.* Giving additional information to foster a complete understanding;
- e_{10} *Other.* Making any other explanation move.

We note the hierarchical nature of the scheme with respect to dialogue acts and explanations; for example, d_1–d_3 could be merged as well as e_1–e_2. While some acts and moves are much more likely to be made by an explainer or an explainee, we did not restrict this to avoid biasing the annotators.[8]

3.3 Crowd-based Annotation Process

The restriction of the annotations to a manageable number of turn-level labels was also made to make the annotation process simple enough to carry it out with independent people. In particular, we hired five freelancers, working as content editors and annotators on the professional crowdworking platform *Upwork*. All were native speakers of English with a 90%+ job success rate on the platform. We clarified the task individually with each of them.

We provided guidelines based on the definitions above, along with general explanations and some examples. Using Label Studio,[9] we developed a task-specific user interface where each dialogue was shown as a sequence of turns and one label of each dimension could be assigned to a turn (if multiple labels seemed appropriate, the best fitting one). Each annotator labeled all 1550 turns. We paid $ 1115 for an overall load of 85 hours, that is, $ 13.12 per hour on average (with minor differences for annotators due to bonuses and varying durations).

Agreement In terms of the conservative measure Fleiss' κ, the inter-annotator agreement among all five was 0.35 for the topic, 0.49 for dialogue acts, and 0.43 for explanation moves. While these values indicate moderate agreement only, they are in line with related subjective labeling tasks of short texts such as news sentences (Al Khatib et al., 2016) and social media arguments (Habernal et al., 2018). Moreover, we exploited the multiple labels we have per turn to consolidate reliable annotations, as described in the following.

Output Annotations For consolidation, we rely on MACE (Hovy et al., 2013), a widely used technique for grading the reliability of crowdworkers based on their agreement with others. The MACE competence scores of the annotators suggest that all did a reasonable job in general, lying in the ranges 0.30–0.76 (topic), 0.58–0.82 (dialogue acts), and 0.45–0.85 (explanation moves) respectively. We applied MACE' functionality to derive one aggregate output label for each dimension from the five annotations weighted by competence scores.

3.4 The Wired Explaining Dialogue Corpus

Table 2 presents detailed general statistics of the three annotation dimensions. More insights into the distribution of annotations across proficiency levels follow in Section 4.

With respect to topic (t_1–t_4), about half of all turns explicitly discuss the *main topic* (27.7%), a *subtopic* (5.7%), or a *related topic* (16.8%). Explainees much more often mention none of these (62.8% vs. 37.3%), underlining the leading role of the explainer in dialogue setting.

[7] We decided to leave a distinction of different explaining types (such as causal or analogy-based explanations) to future work, as it does not match the level of detail in our scheme.

[8] For dialogue acts d_3, d_6, and d_{10} as well as explanation move e_{10}, the annotators had to name the label in free text. We provide these as part of the corpus, we give individual examples of other moves and acts in Section 4.

[9] Label Studio, https://labelstud.io

Label	Explainer #	Explainer %	Explainee #	Explainee %	Total #	Total %
t_1 Main topic	**301**	**38.1**	129	17.0	430	27.7
t_2 Subtopic	52	6.6	36	4.7	88	5.7
t_3 Related topic	142	18.0	118	15.5	260	16.8
t_4 Other/No topic	295	37.3	**477**	**62.8**	**772**	**49.8**
d_1 Check question	183	23.2	62	8.2	245	15.8
d_2 What/How question	77	9.7	38	5.0	115	7.4
d_3 Other question	3	0.4	10	1.3	13	0.8
d_4 Confirming answer	14	1.8	40	5.3	54	3.5
d_5 Disconfirm. answer	3	0.4	21	2.8	24	1.5
d_6 Other answer	2	0.3	23	3.0	25	1.6
d_7 Agreeing statement	75	9.5	190	25.0	265	17.1
d_8 Disagree. statement	2	0.3	10	1.3	12	0.8
d_9 Informing statement	**391**	**49.5**	**305**	**40.1**	**696**	**44.9**
d_{10} Other	40	5.1	61	8.0	101	6.5
e_1 Test understanding	56	7.1	0	0.0	56	3.6
e_2 Test prior knowledge	111	14.1	1	0.1	112	7.2
e_3 Provide explanation	**409**	**51.8**	**270**	**35.5**	**679**	**43.8**
e_4 Request explanation	47	5.9	95	12.5	142	9.2
e_5 Signal understanding	37	4.7	104	13.7	141	9.1
e_6 Signal non-underst.	1	0.1	16	2.1	17	1.1
e_7 Provide feedback	61	7.7	224	29.5	285	18.4
e_8 Provide assessment	10	1.3	1	0.1	11	0.7
e_9 Provide extra info	26	3.3	22	2.9	48	3.1
e_{10} Other	32	4.1	27	3.6	59	3.8
Σ	790	100.0	760	100.0	1550	100.0

Table 2: Corpus distribution of annotated topics (t_1–t_4), dialogue acts (d_1–d_{10}), and explanation moves (e_1–e_{10}) separately for explainer and explainee turns and in total. Per type, the highest value in a column is marked bold.

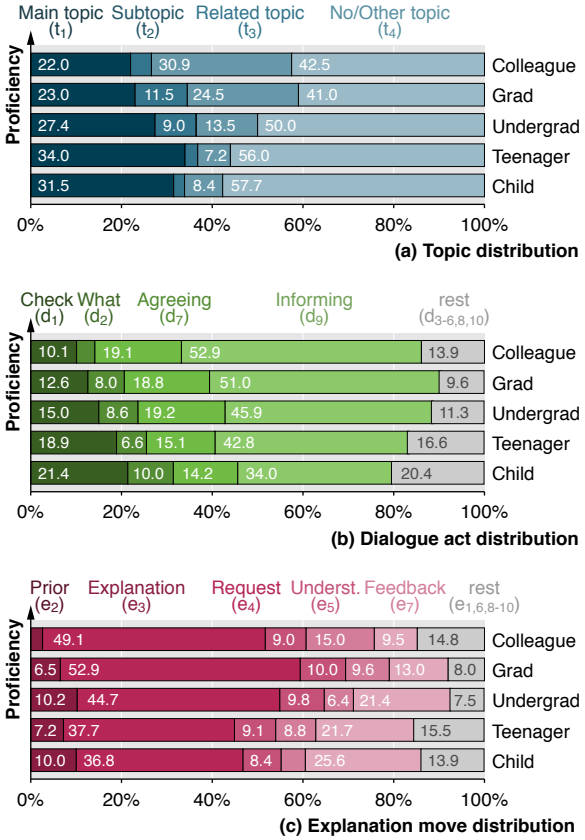

Figure 2: Distribution of topic, discourse act, and explanation act annotations in the corpus, depending on the proficiency of the explainee (from *Child* to *Colleague*).

For dialogue acts (d_1–d_{10}), we see that, quite intuitively, *informing statements* (44.9%) are dominant in explaining dialogues on both sides (explainer 49.5%, explainee 40.1%). However, also *agreeing statements* (17.1%) as well as *check questions* (15.8%) play an important role. The low frequency of *other questions* (0.8%) and *other* (6.5%) suggests that the selected set of dialogue acts cover well what happens in the given kind of dialogues, even though our annotators identifid sum acts, such as *disagreeing statements* (0.8%), rarely only.[10]

Similar holds for the explanation moves (e_1–e_{10}): only 3.8% of all 1550 turns belong to *other*.[11] As expected, the core of explaining is to *provide explanations* (43.8%), also explainees do so in 270 turns (35.5%). Besides, they often *provide feedback* (29.5%). Explainers rather *test prior knowledge* (14.1%) and *test understanding* often (7.1%), but also provide feedback sometimes (7.7%).

[10]Notable examples of other dialogue acts the annotators observed include *greetings* (e.g., "Hi, are you Bella?"), *casual chat* ("What do you do?"), and *gratitude* ("Thank you.").

[11]Here, other cases include *inquiry* ("Hi, are you Bella") and *introduction* ("Bella, I'm George, nice to meet you.").

4 Analysis

One main goal of the presented corpus is to learn how humans explain in dialogical settings. This section analyzes commonalities and differences regarding meta-information available in the corpus.

4.1 Explaining across Proficiency Levels

First, we explore to what extent explaining differs depending on the proficiency of the explainee. Figure 2 shows the distributions of the three annotated dimensions separately for the five given explainee levels. For dialogue acts and explanation moves, we distinguish only the most frequent labels and merge all others into a class *rest*.

With respect to topic, we see that particularly the discussion of *related topics* grows notably with the explainee's proficiency, from 8.4% of all annotations for children to 30.9% for colleagues. Conversely, the *main topic* is mentioned less in dialogues with more proficient explainees; the same holds for *no/other topic*. *Subtopics* are considered mainly with grads (11.5%) and undergrads (9.0%), possibly related to the way they learn.

Topic Sequences	Explainer	Explainee	Total
(Main, Rel, Main)	**24.6%**	7.7%	**15.4%**
(Main, Rel, Main, Rel, Main)	–	–	7.7%
(Main)	12.3%	**18.5%**	6.2%
(Rel, Main, Rel, Main, Rel, Main)	–	–	4.6%
(Main, Rel)	3.1%	10.8%	4.6%
(Rel, Main, Rel, Main)	3.1%	–	3.1%
(Main, Sub, Main)	–	–	3.1%
(Main, Sub, Main, Rel, Main)	4.6%	3.1%	3.1%

Table 3: Relative frequencies of all recurring sequences of *main*, *sub*, and *rel*ated topic in the corpus' dialogues and in the explainers and explainees' parts alone.

Labels	Act/Move Pair	Explainer	Explainee	Total
d_9/e_3	Informing/Explanation	**45.9%**	**31.3%**	**38.8%**
d_7/e_7	Agreeing/Feedback	3.9%	14.2%	9.0%
d_7/e_5	Agreeing/Understanding	3.5%	9.1%	6.3%
d_1/e_2	Check/Prior	10.5%	–	5.4%
d_1/e_4	Check/Request	2.7%	6.8%	4.7%
d_2/e_4	What/Request	3.0%	4.5%	3.7%
d_{10}/e_{10}	Other/Other	2.8%	2.6%	2.7%
d_1/e_1	Check/Understanding	5.1%	–	2.6%
d_4/e_7	Confirming/Feedback	1.4%	3.7%	2.5%
d_9/e_7	Informing/Feedback	0.5%	4.2%	2.3%

Table 4: Relative frequencies of the ten most frequent pairs of dialogue act and explanation move in the corpus and the differences for explainers and explainees.

Explainer			Explainee		
Word	**Frequency**	**Ratio**	**Word**	**Frequency**	**Ratio**
here	0.16%	4.20	yes	0.21%	5.12
around	0.12%	4.03	mean	0.14%	4.20
space	0.24%	3.32	stuff	0.11%	3.11
light	0.18%	2.96	oh	0.16%	2.75
earth	0.10%	2.65	yeah	0.65%	2.70
us	0.15%	2.39	many	0.12%	2.39
want	0.14%	2.28	interesting	0.12%	2.11
going	0.22%	2.19	well	0.21%	1.94
point	0.11%	2.11	like	1.10%	1.85
thing	0.18%	1.93	no	0.18%	1.83

Table 5: The top-10 words used specifically by explainers and explainees, respectively, along with the relative frequency (minimum 0.1%) and specificity ratio (e.g., explainees say "yes" 5.12 times as often as explainers).

For dialogue acts, the key difference lies between the proportion of *informing statements* and the number of questions asked (d_1 and d_2). Whereas the former monotonously goes up from 34.0% (child) to 52.9% (colleague), particularly the use of *check questions* is correlated inversely with proficiency, used mainly to test prior knowledge and to check understanding. A similar behavior can be observed for explanation moves. There, *providing feedback* shrinks from 25.6% to 9.5%, while *providing explanations* mostly grows, with peak at grads (52.9%). In contrast, how often people *request explanations* remains stable across proficiency levels.

4.2 Interactions of Topics, Moves, and Acts

Interactions of the annotated dimensions happen between the turns and within a turn. We analyze one example of each here, and, due the limited data size, we look at topics separately from dialogue act and explanation moves.

Inspired by the flow model of Wachsmuth and Stein (2017), Table 3 shows all eight sequences of topics that occur more than once among the 65 dialogues. Each sequence shows the ordering of topics being discussed, irrespective of how often each topic is mentioned in a row. Most dialogues start and end with the main topic, often in alternation with related topics, such as *(Main, Rel, Main)* in 15.4% of all cases (sometimes also with subtopics). The ordering of what *explainers* talk about is similar, whereas *explainees* often focus on the main topic only (18.5%).

Table 4 lists the top-10 pairs of acts and moves. *Informing statements* that *provide explanations* are most common across both explainers (45.9%) and explainees (31.3%). *Agreeing statements* (d_7) and *check questions* (d_1) cooccur with multiple moves, and especially *providing feedback* happens via different dialogue acts. As expected in the given setting, explainees never check for prior knowledge or understanding (d_1/e_2, d_1/e_1). Instead, they agree by providing feedback or signaling understanding (d_7/e_7, d_7/e_5) much more often than explainers.

4.3 Language of Explainers and Explainees

Finally, we investigate basic differences in the language of the two sides: We determine the words that are often used by explainers (at least 0.1% of all words) and rarely by explainees, or vice versa.

Table 5 presents the 10 most specific words on each side. Aside from some topic-specific words (e.g., "light"), the explainer's list includes typical words used in meta-language, as in this explanation to a teenager: "I *want* to know if you agree, sleep is the coolest *thing* you've ever heard of." On the explainee's side, we find multiple reactive words, such as "oh" and "interesting", but also indicators of vagueness, as in this colleague's response to an explanation of hacking: "So all kind of older logic and *stuff like* that. So, I *mean*, it's sort of based on, *like*, you're presented the little MUX chip."

5 Experiments

The second goal of the corpus is to serve the creation of XAI systems that mimic human explainers. As an initial endeavor, this section reports on baseline experiments on the computational prediction of topics, dialogue acts, and explanation moves.

5.1 Experimental Setup

We evaluate three models based on BERT (Devlin et al., 2019), along with a simple majority baseline, for predicting each dialogue turn dimension in 13-fold cross-topic validation: For each main topic, we trained one model on the other 12 topics and tested it against the labels of the respective dimension. We average the resulting F_1-scores over all 13 folds.[12] Figure 3 illustrates the three BERT variants.

BERT-basic The first model simply adds a classification head to BERT. It takes as input the dialogue's main topic and the turn's text, x_i (separated by `[SEP]`), as well as the label y_i to predict (topic t_i, dialogue act d_i, or explanation move e_i). We trained the model for five epochs, optimizing its F_1-score on the turns of two main topics. We balanced the training set using oversampling to prevent the model from only predicting the majority label.

BERT-sequence Turns made in explaining dialogues depend on previous turns, for example, a conclusion on the *main topic* may be preceded by a *related topic* (see Table 3). In the second model, we exploit such dependencies with turn-level sequence labeling: Given the sequence $(x_1, \ldots, x_n)$ of all turns in a dialogue, the input to predicting a label y_i of x_i is the turn's history $(x_1, \ldots, x_{i-1})$ along with all previously predicted labels $(y_1, \ldots, y_{i-1})$ of the same dimension. For each turn, we encode the history in a `CLS` embedding with BERT. Then, we pass all labels and `CLS` embeddings through a CRF layer to model the label's dependencies.

BERT-multitask Finally, the interaction of topic t_i, act d_i, and move e_i in a turn may be relevant. For example, an *informing statement* likely *provides an explanation* (see Table 4). Our third model thus learns to classify all three dimensions jointly in a multitask fashion, based on multitask-NLP.[13] We trained one multitask model each with one of the three dimensions as main task and the others as auxiliary tasks, oversampling with respect to the main task. To this end, we employ a shared BERT encoder and three classification heads, one for each task. The final loss is the weighted average of the three classification losses, with weight 0.5 for the main task and 0.25 for both others. We trained the models for 10 epochs allowing them to converge.

[12] All models start from the `bert-based-uncased`, and are trained with a learning rate of $2e^{-5}$ and a batch size of 4.

[13] Multitask NLP, https://multi-task-nlp.readthedocs.io

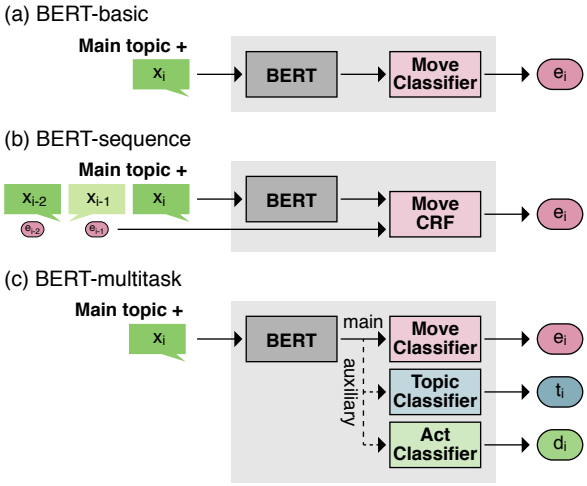

Figure 3: Sketch of the three evaluated models, here for predicting a turn's explanation move, e_i: (a) *BERT-basic* labels a turn in isolation. (b) *BERT-sequence* takes the labels of previous turns into account. (c) *BERT-multitask* classifies all three turn dimensions simultaneously.

Approach	Main T. (t_1)	Sub-T. (t_2)	Related T. (t_3)	No/Oth. T. (t_4)	Macro F_1-Score
BERT-basic	0.58	0.11	**0.44**	**0.89**	0.51
BERT-sequence	**0.61**	**0.13**	**0.44**	**0.89**	**0.52**
BERT-multitask	0.43	0.04	0.36	0.81	0.41
Majority baseline	0.00	0.00	0.00	0.66	0.17

Table 6: Topic prediction results: The F_1-scores of the evaluated BERT models for each considered relation to the main topic, t_1–t_4, as well as the macro-averaged F_1-score. The best value in each column is marked bold.

5.2 Results

Tables 6–8 show the individual and the macro F_1-scores for all three dimensions.

BERT-sequence performs best across all three labeling tasks, highlighting the impact of modeling the sequential interaction in dialogues. It achieves a macro F_1-score of 0.52 for topics, 0.47 for dialogue acts, and 0.43 for explanation moves. However, likely due to data sparsity, some labels remain hard to predict, such as *Subtopic* (t_2), *disagreement statements* (d_8), and *provide assessment* (e_8).

BERT-basic beats BERT-sequence on a few labels, such as *signal non-understanding* (e_8), but

Approach	Check Q. (d_1)	What/H. Q. (d_2)	Other Q. (d_3)	Confirm. A. (d_4)	Disconf. A. (d_5)	Other A. (d_6)	Agree. St. (d_7)	Disagr. St. (d_8)	Inform. St. (d_9)	Other (d_{10})	Macro F_1-Score
BERT-basic	**0.76**	**0.73**	0.00	0.33	**0.67**	0.00	0.51	0.00	**0.87**	0.57	0.44
BERT-sequence	**0.76**	0.72	0.00	**0.35**	**0.67**	0.00	****0.69**	0.00	**0.87**	**0.61**	**0.47**
BERT-multitask	0.54	0.49	0.00	0.29	0.59	0.00	0.53	**0.09**	0.84	0.44	0.38
Majority baseline	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.62	0.00	0.06

Table 7: Dialogue act prediction results: The F_1-scores of the evaluated BERT models for each considered dialogue act, d_1–d_{10}, as well as the macro-averaged F_1-score. The best value in each column is marked bold.

Approach	Test U. (e_1)	Test P.K. (e_2)	Provide Ex. (e_3)	Request Ex. (e_4)	Signal U. (e_5)	Signal N.U. (e_6)	Provide Fe. (e_7)	Provide As. (e_8)	Provide E.I. (e_9)	Other (e_{10})	Macro F_1-Score
BERT-basic	**0.27**	**0.64**	**0.84**	0.60	0.29	**0.34**	0.51	0.00	**0.11**	0.50	0.41
BERT-sequence	**0.27**	**0.64**	**0.84**	**0.64**	**0.33**	0.21	****0.60**	**0.15**	0.08	**0.56**	**0.43**
BERT-multitask	0.21	0.54	0.80	0.40	0.16	0.32	0.53	0.00	0.08	0.35	0.34
Majority baseline	0.00	0.00	0.61	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.06

Table 8: Explanation move prediction results: The F_1-scores of the evaluated BERT models for each considered explanation move, e_1–e_{10}, as well as the macro-averaged F_1-score. The best value in each column is marked bold.

cannot compete overall. *BERT-multitask* performs worst among the three models. We attribute this to the data imbalance: While oversampling helps with respect to the main task, it does not benefit the label distribution of the auxiliary tasks. Also, optimizing the loss weights of the three tasks may further aid multitask learning, but such an engineering of prediction models is not the focus of this work.

6 Conclusion

How humans explain in dialogical settings is still understudied. This paper has presented a first corpus for computational research on controlled explaining dialogues, manually annotated for topics, dialogue acts, and explanation moves. Our analysis has revealed intuitive differences in the language of explainers and explainees and their dependence on the explainee's proficiency. Moreover, baseline experiments suggest that a prediction of the annotated dimensions is feasible and benefits from modeling interactions. With these results, we lay the ground towards more human-centered XAI. We expect that respective systems need to learn to how to explain depending on the explainee's reactions, and how to proactively lead an explaining dialogue to achieve understanding on the explainee's side.

A limitation of the corpus lies in the restricted corpus size caused by the availability of source data, preventing deeper statistical analyses and likely rendering a direct training of dialogue systems on the corpus hard. Also, it remains to be explored what findings generalize beyond the controlled setting of the given dialogues. Future work should thus target both the scale and the heterogeneity of explaining data, in order to provide the pervasive communicative process of explaining the attention it deserves.

Acknowledgments

This work has been supported by the Deutsche Forschungsgemeinschaft (DFG, German Research Foundation), partially under project number TRR 318/1 2021 – 438445824 and partially under SFB 901/3 – 160364472. We thank Meisam Booshehri, Hendrik Buschmeier, Philipp Cimiano, Josephine Fisher, Angela Grimminger, and Erick Ronoh for their input and feedback to the annotation scheme. We also thank Akshit Bhatia for his help with the corpus preparation as well as the anonymous freelancers on Upwork for their annotations.

7 Ethical Statement

We do not see any immediate ethical concerns with respect to the research in this paper. The data included in the corpus is freely available. All participants involved in the given dialogues gave their consent to be recorded and received expense allowances, as far as perceivable from the Wired web resources. As discussed in the paper, the three freelancers in our annotation study were paid about $13 per hour, which exceeds the minimum wage in most US states and is also conform to the standards in the regions of our host institution. In our view, the provided prediction models target dimensions of dialogue turns that are not prone to be misused for ethically doubtful applications.

References

Khalid Al Khatib, Henning Wachsmuth, Johannes Kiesel, Matthias Hagen, and Benno Stein. 2016. A news editorial corpus for mining argumentation strategies. In *Proceedings of COLING 2016, the 26th International Conference on Computational Linguistics: Technical Papers*, pages 3433–3443. The COLING 2016 Organizing Committee.

Khalid Al Khatib, Henning Wachsmuth, Kevin Lang, Jakob Herpel, Matthias Hagen, and Benno Stein. 2018. Modeling deliberative argumentation strategies on wikipedia. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 2545–2555. Association for Computational Linguistics.

Alejandro Barredo Arrieta, Natalia Díaz-Rodríguez, Javier Del Ser, Adrien Bennetot, Siham Tabik, Alberto Barbado, Salvador Garcia, Sergio Gil-Lopez, Daniel Molina, Richard Benjamins, Raja Chatila, and Francisco Herrera. 2020. Explainable artificial intelligence (XAI): Concepts, taxonomies, opportunities and challenges toward responsible AI. *Information Fusion*, 58:82–115.

Sarah Bourse and Patrick Saint-Dizier. 2012. A repository of rules and lexical resources for discourse structure analysis: the case of explanation structures. In *Proceedings of the Eighth International Conference on Language Resources and Evaluation (LREC-2012)*, pages 2778–2785, Istanbul, Turkey. European Languages Resources Association (ELRA).

Harry Bunt, Jan Alexandersson, Jean Carletta, Jae-Woong Choe, Alex Chengyu Fang, Koiti Hasida, Kiyong Lee, Volha Petukhova, Andrei Popescu-Belis, Laurent Romary, Claudia Soria, and David Traum. 2010. Towards an ISO standard for dialogue act annotation. In *Proceedings of the Seventh International Conference on Language Resources and Evaluation (LREC'10)*, Valletta, Malta. European Language Resources Association (ELRA).

Roberto Confalonieri, Tarek R. Besold, Tillman Weyde, Kathleen Creel, Tania Lombrozo, Shane T. Mueller, and Patrick Shafto. 2019. What makes a good explanation? Cognitive dimensions of explaining intelligent machines. In *Proceedings of the 41th Annual Meeting of the Cognitive Science Society, CogSci 2019: Creativity + Cognition + Computation, Montreal, Canada, July 24-27, 2019*, pages 25–26.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, Minneapolis, Minnesota. Association for Computational Linguistics.

Andrei Dulceanu, Thang Le Dinh, Walter Chang, Trung Bui, Doo Soon Kim, Manh Chien Vu, and Seokhwan Kim. 2018. PhotoshopQuiA: A corpus of non-factoid questions and answers for why-question answering. In *Proceedings of the Eleventh International Conference on Language Resources and Evaluation (LREC 2018)*, Miyazaki, Japan. European Language Resources Association (ELRA).

Myroslava O. Dzikovska, Rodney D. Nielsen, and Chris Brew. 2012. Towards effective tutorial feedback for explanation questions: A dataset and baselines. In *Proceedings of the 2012 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 200–210, Montréal, Canada. Association for Computational Linguistics.

Angela Fan, Yacine Jernite, Ethan Perez, David Grangier, Jason Weston, and Michael Auli. 2019. ELI5: Long form question answering. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 3558–3567, Florence, Italy. Association for Computational Linguistics.

Josefine Finke, Ilona Horwath, Tobias Matzner, and Christian Schulz. 2022. (de)coding social practice in the field of xai: Towards a co-constructive framework of explanations and understanding between lay users and algorithmic systems. In *Artificial Intelligence in HCI*, pages 149–160, Cham. Springer International Publishing.

Lionel Fontan and Patrick Saint-Dizier. 2008. Analyzing the explanation structure of procedural texts: Dealing with advice and warnings. In *Semantics in Text Processing. STEP 2008 Conference Proceedings*, pages 115–127. College Publications.

Daniel Fried, Jacob Andreas, and Dan Klein. 2018. Unified pragmatic models for generating and following instructions. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 1951–1963, New Orleans, Louisiana. Association for Computational Linguistics.

Alan Garfinkel. 2009. *Forms of Explanation: Rethinking the Questions in Social Theory*, revised edition. Yale University Press, New Haven & London, New Haven; London.

Leilani H. Gilpin, David Bau, Ben Z. Yuan, Ayesha Bajwa, Michael Specter, and Lalana Kagal. 2018. Explaining explanations: An overview of interpretability of machine learning. ArXiv: 1806.00069.

Bryce Goodman and Seth Flaxman. 2017. European union regulations on algorithmic decision-making and a "right to explanation". *AI Magazine*, 38(3):50–57.

Ivan Habernal, Henning Wachsmuth, Iryna Gurevych, and Benno Stein. 2018. The argument reasoning comprehension task: Identification and reconstruction of implicit warrants. In *Proceedings of the 2018*

Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers), pages 1930–1940. Association for Computational Linguistics.

Dirk Hovy, Taylor Berg-Kirkpatrick, Ashish Vaswani, and Eduard Hovy. 2013. Learning whom to trust with MACE. In *Proceedings of the 2013 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, Atlanta, Georgia. Association for Computational Linguistics.

Peter Jansen, Niranjan Balasubramanian, Mihai Surdeanu, and Peter Clark. 2016. What's in an explanation? characterizing knowledge and inference requirements for elementary science exams. In *Proceedings of COLING 2016, the 26th International Conference on Computational Linguistics: Technical Papers*, pages 2956–2965, Osaka, Japan. The COLING 2016 Organizing Committee.

Pamela W. Jordan, Maxim Makatchev, and Umarani Pappuswamy. 2006. Understanding complex natural language explanations in tutorial applications. In *Proceedings of the Third Workshop on Scalable Natural Language Understanding*, pages 17–24, New York City, New York. Association for Computational Linguistics.

Lei Li, Yongfeng Zhang, and Li Chen. 2021. Personalized transformer for explainable recommendation. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 4947–4957, Online. Association for Computational Linguistics.

William C Mann and Sandra A Thompson. 1988. Rhetorical structure theory: Toward a functional theory of text organization. *Text-interdisciplinary Journal for the Study of Discourse*, 8(3):243–281.

Tim Miller. 2019. Explanation in artificial intelligence: Insights from the social sciences. *Artificial Intelligence*, 267:1–38.

Preslav Nakov, Doris Hoogeveen, Lluís Màrquez, Alessandro Moschitti, Hamdy Mubarak, Timothy Baldwin, and Karin Verspoor. 2017. SemEval-2017 task 3: Community question answering. In *Proceedings of the 11th International Workshop on Semantic Evaluation (SemEval-2017)*, pages 27–48, Vancouver, Canada. Association for Computational Linguistics.

Katharina J. Rohlfing, Philipp Cimiano, Ingrid Scharlau, Tobias Matzner, Heike M. Buhl, Hendrik Buschmeier, Elena Esposito, Angela Grimminger, Barbara Hammer, Reinhold Häb-Umbach, Ilona Horwath, Eyke Hüllermeier, Friederike Kern, Stefan Kopp, Kirsten Thommes, Axel-Cyrille Ngonga Ngomo, Carsten Schulte, Henning Wachsmuth, Petra Wagner, and Britta Wrede. 2021. Explanation as a social practice: Toward a conceptual framework for the social design of ai systems. *IEEE Transactions on Cognitive and Developmental Systems*, 13(3):717–728.

Xuelin Situ, Ingrid Zukerman, Cecile Paris, Sameen Maruf, and Gholamreza Haffari. 2021. Learning to explain: Generating stable explanations fast. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 5340–5355, Online. Association for Computational Linguistics.

Youngseo Son, Nipun Bayas, and H. Andrew Schwartz. 2018. Causal explanation analysis on social media. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 3350–3359, Brussels, Belgium. Association for Computational Linguistics.

Andreas Stolcke, Klaus Ries, Noah Coccaro, Elizabeth Shriberg, Rebecca Bates, Daniel Jurafsky, Paul Taylor, Rachel Martin, Carol Van Ess-Dykema, and Marie Meteer. 2000. Dialogue act modeling for automatic tagging and recognition of conversational speech. *Computational Linguistics*, 26(3):339–374.

John M. Swales. 1990. *Genre Analysis: English in Academic and Research Settings*. Cambridge University Press.

Keith Vander Linden. 1992. The expression of local rhetorical relations in instructional text. In *Proceedings of the 30th Annual Meeting of the Association for Computational Linguistics*, pages 318–320.

Henning Wachsmuth and Benno Stein. 2017. A universal model for discourse-level argumentation analysis. *Special Section of the ACM Transactions on Internet Technology: Argumentation in Social Media*, 17(3):28:1–28:24.

Semih Yagcioglu, Aykut Erdem, Erkut Erdem, and Nazli Ikizler-Cinbis. 2018. RecipeQA: A challenge dataset for multimodal comprehension of cooking recipes. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 1358–1368, Brussels, Belgium. Association for Computational Linguistics.

Ziqi Zhang, Philip Webster, Victoria Uren, Andrea Varga, and Fabio Ciravegna. 2012. Automatically extracting procedural knowledge from instructional texts using natural language processing. In *Proceedings of the Eighth International Conference on Language Resources and Evaluation (LREC'12)*, pages 520–527, Istanbul, Turkey. European Language Resources Association (ELRA).

Schema Encoding for Transferable Dialogue State Tracking

Hyunmin Jeon
Computer Science and Engineering
POSTECH, Pohang, South Korea
jhm9507@postech.ac.kr

Gary Geunbae Lee
Computer Science and Engineering
Graduate School of Artificial Intelligence
POSTECH, Pohang, South Korea
gblee@postech.ac.kr

Abstract

Dialogue state tracking (DST) is an essential sub-task for task-oriented dialogue systems. Recent work has focused on deep neural models for DST. However, the neural models require a large dataset for training. Furthermore, applying them to another domain needs a new dataset because the neural models are generally trained to imitate the given dataset. In this paper, we propose **S**chema **E**ncoding for **T**ransferable **D**ialogue **S**tate **T**racking (SET-DST), which is a neural DST method for effective transfer to new domains. Transferable DST could assist developments of dialogue systems even with few dataset on target domains. We use a schema encoder not just to imitate the dataset but to comprehend the schema of the dataset. We aim to transfer the model to new domains by encoding new schemas and using them for DST on multi-domain settings. As a result, SET-DST improved the joint accuracy by 1.46 points on MultiWOZ 2.1.

1 Introduction

The objective of task-oriented dialogue systems is to help users achieve their goals by conversations. Dialogue state tracking (DST) is the essential sub-task for the systems to perform the purpose. Users may deliver the details of their goals to the systems during the conversations, e.g., what kind of food they want the restaurant to serve and at what price level they want to book the hotel. Thus, the systems should exactly catch the details from utterances. They should also communicate with other systems by using APIs to achieve users' goals, e.g., to search restaurants and to reserve hotels. The goal of DST is not only to classify the users' intents but also to fill the details into predefined templates that are used to call APIs.

Recent work has used deep neural networks for DST with supervised learning. They have improved the accuracy of DST; however, they require a large dataset for training. Furthermore, they need a new dataset to be trained on another domain. Unfortunately, the large dataset for training a DST model is not easy to be developed in real world. The motivation of supervised learning is to make deep neural networks imitate humans. But, they actually imitate the given datasets rather than humans. Someones who have performed hotel reservation work could easily perform restaurant reservation work if some guidelines are provided, but neural models may have to be trained on a new dataset of the restaurant domain. The difference between humans and neural models is that humans can learn how to read guidelines and to apply the guidelines to their work. This is why transfer learning is important to train neural models on new domains.

In this paper, we propose **S**chema **E**ncoding for **T**ransferable **D**ialogue **S**tate **T**racking (SET-DST), which is a neural DST method with transfer learning by using dataset schemas as guidelines for DST. The motivation of this study is that humans can learn not only how to do their work, but also how to apply the guidelines to the work. We aim to make a neural model learn how to apply the schema guidelines to DST beyond how to fill predefined slots by simply imitating the dataset on multi-domain settings. The schema includes metadata of the dataset, e.g., which domains the dataset covers and which slots have to be filled to achieve goals. SET-DST has a schema encoder to represent the dataset schema, and it uses the schema representation to understand utterances and to fill slots. Recently, transfer learning has been becoming important because development of new datasets is costly. Transfer learning makes it possible to pre-train neural models on large-scale datasets to effectively fine-tune the models on small-scale downstream tasks.

We used SGD (Rastogi et al., 2020) as the large-scale dataset, and evaluated SET-DST on MultiWOZ 2.1 (Eric et al., 2020), which is a standard benchmark dataset for DST, as the downstream task. SET-DST achieved state-of-the-art accuracy

Proceedings of the 29th International Conference on Computational Linguistics, pages 355–366
October 12–17, 2022.

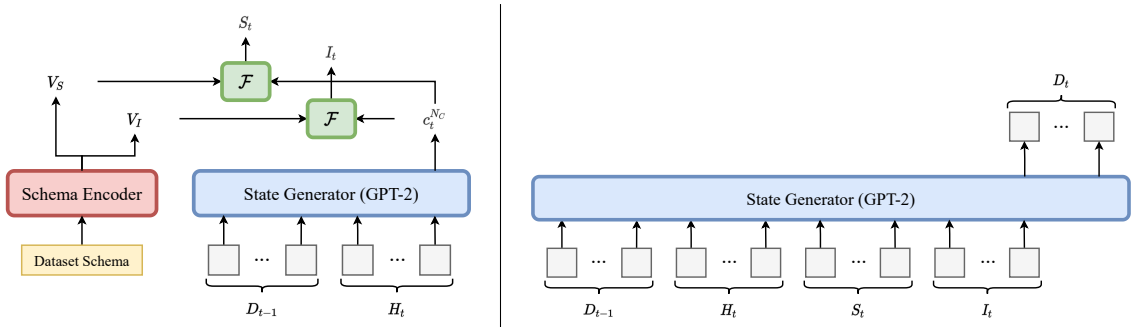

(a) Schema encoding for active slots and intents classification.

(b) Dialogue state generation.

Figure 1: Overview of SET-DST. The schema encoder takes the dataset schema and generates slot vectors and intent vectors. The state generator takes the previous dialogue state D_{t-1} and the dialogue history H_t to calculate active scores of slots and intents. $\mathcal{F}$ is an score function to calculate whether the slots or intents are activated on turn t. Then, the state generator additionally takes the activated slots and intents to generate the current dialogue state D_t. S_t indicates the activated slots and I_t indicates the activated intents.

on the downstream DST task. We further confirmed that SET-DST worked well on the small downstream dataset. This result demonstrates that transfer learning with schema encoding improves the performance of neural DST models and the efficiency of few-shot learning on DST.

2 Related Work

Traditional DST models extract semantics by using natural language understanding (NLU) modules to generate dialogue states (Williams, 2014; Wang and Lemon, 2013). The limitation of these models is that they rely on features extracted by humans.

Recent work has focused on building end-to-end DST models without hand-crafted features. Zhong et al. (2018) use global modules to share parameters between different slots. Nouri and Hosseini-Asl (2018) improve the latency by removing inefficient recurrent layers. Transferable DST models that can be adapted to new domains by removing the dependency on the domain ontology are proposed (Ren et al., 2018; Wu et al., 2019). Zhou and Small (2019) attempt to solve DST as a question answering task using knowledge graph.

More recently, large-scale pre-trained language models such as BERT (Devlin et al., 2019) and GPT-2 (Radford et al., 2019) are used for DST. The pre-trained BERT acts as an NLU module to understand utterances (Lee et al., 2019; Zhang et al., 2020a; Kim et al., 2020; Heck et al., 2020). GPT-2 makes it possible to solve DST as a conditional language modeling task (Hosseini-Asl et al., 2020; Peng et al., 2021).

Rastogi et al. (2020) propose the baseline method that defines the schema of dataset and uses it for training and inference. A drawback of them is that the calculation cost is high because they use the domain ontology and access all values to estimate the dialogue state. DST models that uses schema graphs to encode the relation between slots and values are proposed (Chen et al., 2020; Zhu et al., 2020). However, they focus on encoding the relation between slots and values of the given domains not on adaptation to new domains.

In this paper, we focus on making the model learn how to understand the schema and how to apply it to estimate the dialogue state, not just on encoding the in-domain relation.

3 Schema Encoding for Transferable Dialogue State Tracking

In this section, we describe the architecture of SET-DST and how to optimize it. Figure 1 shows the overview of our method. The model consists of the schema encoder and the state generator. SET-DST generates the dialogue state in two steps: (a) schema encoding and classification, and (b) dialogue state generation. In this paper, we define some terms as follows.

Schema Metadata of the dataset, e.g., what domains, services, slots, and intents the dataset covers. A dataset has a schema that describes the dataset.

Domain What domains the conversation goes on, e.g., restaurant, hotel, and attraction. A conversation can go on multiple domains.

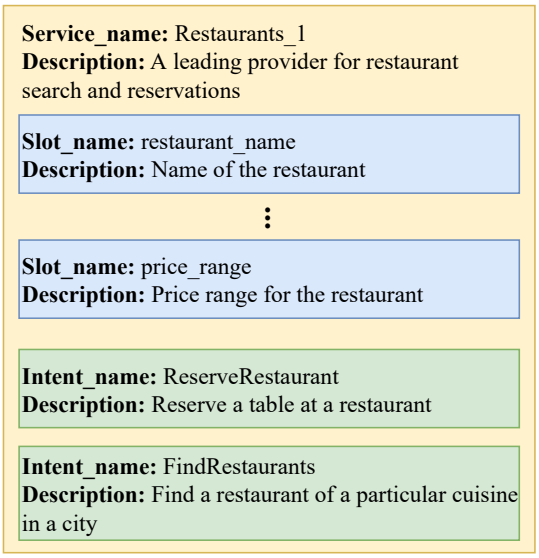

Figure 2: Example of schema for restaurant search and reservation service including slots and intents.

Service What services the system provides to users. It is similar to domain, but application-level. For example, restaurant domain can have two different services: (1) a service for searching and reserving restaurants and (2) a service focused on searching and comparing restaurants. In real world, a service corresponds to an application.

Action Abstract actions of users to achieve their goals during conversations, e.g., to inform the system their requirements or to request the system for some information. Appendix B demonstrates the details of the user actions covered in this paper.

Slot The details of the user goals, e.g., the type of food and the price range of hotel. Slots are predefined based on the domains or services that the system should cover, and the slots are filled by DST. The schema includes the information of slots.

Value The values that have actual meaning for the corresponding slots, e.g., cheap or expensive about the price range of hotel. The systems should match slot-value pairs from conversations.

Intent Sub-goals to achieve the final goals of users. A goal consists of one or more intents, and an intent is achieved over one or more conversation turns. In real world, an intent corresponds to an API. For example, to search restaurants or to book hotels should be performed by APIs of external systems. Furthermore, The dialogue system should predict the slot-value pairs which correspond to arguments to call APIs.

3.1 Schema Encoding

We use the pre-trained BERT[1] for the schema encoder. Figure 2 shows an example of the schema for *Restaurant_1* service that is a service to search and reserve restaurants. Services, slots, and intents consist of name and short description. The name and description of the service in the schema are fed into BERT to generate service vector v_R as

$$\begin{aligned} o_R &= \texttt{BERT}(\texttt{[CLS]}\, n_R : d_R\, \texttt{[SEP]}) \\ v_R &= W_R \cdot o_R^{\texttt{[CLS]}} \in \mathbb{R}^h \end{aligned}, \quad (1)$$

where n_R is the service name, d_R is the service description, and h is the hidden size. $o_R^{\texttt{[CLS]}}$ is the output of `[CLS]` token, and $W_R \in \mathbb{R}^{h \times h}$ is a fully connected (FC) layer. `[CLS]` and `[SEP]` are special tokens that mean the start and end of the sentence, respectively. The service in Figure 2 can be represented as `[CLS] Restaurants_1 : A leading provider for restaurant search and reservations [SEP]` to be fed into BERT. The slots and intents in the schema are also fed into BERT to generate slot vectors $V_S = \{v_S^1, \cdots v_S^{N_S}\} \in \mathbb{R}^{N_S \times h}$ and intent vectors $V_I = \{v_I^1, \cdots, v_I^{N_I}\} \in \mathbb{R}^{N_I \times h}$, respectively, as follows:

$$\begin{aligned} o_S^j &= \texttt{BERT}\left(\texttt{[CLS]}\, n_S^j : d_S^j\, \texttt{[SEP]}\right) \\ v_S^j &= W_S \cdot o_S^{j,\texttt{[CLS]}} \in \mathbb{R}^h, \quad j \in [1, N_S] \end{aligned}, \quad (2)$$

$$\begin{aligned} o_I^k &= \texttt{BERT}\left(\texttt{[CLS]}\, n_I^k : d_I^k\, \texttt{[SEP]}\right) \\ v_I^k &= W_I \cdot o_I^{k,\texttt{[CLS]}} \in \mathbb{R}^h, \quad k \in [1, N_I] \end{aligned}. \quad (3)$$

N_S and N_I mean the number of slots and intents for the service, respectively. n_S^j is the j-th slot name, and d_S^j is the j-th slot description. $o_S^{j,\texttt{[CLS]}}$ is the output of `[CLS]` token from the j-th slot, and $W_S \in \mathbb{R}^{h \times h}$ is an FC layer. Similarly, n_I^k is the k-th intent name, and d_I^k is the k-th intent description. $o_I^{k,\texttt{[CLS]}}$ is the output of `[CLS]` token from the k-th intent, and $W_I \in \mathbb{R}^{h \times h}$ is an FC layer. The schema encoder takes v_R, V_S, and V_I to update the slot vectors V_S and intent vectors V_I

with attention mechanism as follows:

$$\begin{aligned} a_S &= \texttt{softmax}\left(V_S \cdot v_R\right) \in \mathbb{R}^{N_S} \\ v_{R,S} &= \left(V_S\right)^T \cdot a_S \in \mathbb{R}^h \end{aligned}, \quad (4)$$

$$v_S^j = W_{RS} \cdot \left(v_{R,S} \oplus v_S^j\right) \in \mathbb{R}^h, \quad (5)$$

$$\begin{aligned} a_I &= \texttt{softmax}\left(V_I \cdot v_R\right) \in \mathbb{R}^{N_I} \\ v_{R,I} &= \left(V_I\right)^T \cdot a_I \in \mathbb{R}^h \end{aligned}, \quad (6)$$

$$v_I^k = W_{RI} \cdot \left(v_{R,I} \oplus v_I^k\right) \in \mathbb{R}^h. \quad (7)$$

$W_{RS} \in \mathbb{R}^{h \times 2h}$ and $W_{RI} \in \mathbb{R}^{h \times 2h}$ are FC layers, and $\oplus$ means the concatenation of two vectors. The slot vectors and intent vectors updated with reference to the service vector are used for next steps: classification and generation.

3.2 Slot and Intent Classification

SET-DST takes the slot vectors and intent vectors to classify what slots and intents are activated by users. We use the pre-trained GPT-2[1] for the state generator that encodes the dialogue history and generates the dialogue state as a sequence of words. The state generator encodes the dialogue history H_t that is accumulated during the conversation and the previous dialogue state D_{t-1} to calculate the context vector C_t as

$$\begin{aligned} C_t &= \left\{c_t^1, \cdots, c_t^{N_C}\right\} \in \mathbb{R}^{N_C \times h} \\ &= \texttt{GPT-2}\left(D_{t-1} \oplus H_t\right), \end{aligned} \quad (8)$$

where $N_C = |C_t|$, and c_t^i means the GPT-2 output of the i-th word. Then, the last output of C_t is used to classify which slots and intents are activated in the current conversation as follows:

$$P\left(s_t^j = \texttt{Active}\right) = \mathcal{F}\left(c_t^{N_C}, v_S^j\right), \quad (9)$$

$$P\left(i_t^k = \texttt{Active}\right) = \mathcal{F}\left(c_t^{N_C}, v_I^k\right), \quad (10)$$

where $P(s_t^j = \texttt{Active})$ means the probability that the j-th slot is activated on turn t, and $P(i_t^k = \texttt{Active})$ means the probability that the k-th intent is activated on turn t. v_S^j and v_I^k indicate the slot vector of the j-th slot and the intent vector of the k-th intent, respectively, calculated by the schema encoder. $\mathcal{F}$ is a projection layer to calculate the probabilities using the context vector, slot vector, and intent vector. We define $\mathcal{F}(x, y)$ as a function transforming vectors x and y into a probability scalar as

$$\begin{aligned} h_1 &= \texttt{tanh}\left(W_1 \cdot x\right) \\ h_2 &= \texttt{tanh}\left(W_2 \cdot \left(h_1 \oplus y\right)\right), \\ \mathcal{F}(x, y) &= \sigma\left(W_3 \cdot h_2\right) \end{aligned} \quad (11)$$

where $W_1 \in \mathbb{R}^{h \times h}$, $W_2 \in \mathbb{R}^{h \times 2h}$, and $W_3 \in \mathbb{R}^{1 \times h}$ are FC layers. Activate slots and intents are classified based on the probabilities $P(s_t^j = \texttt{Active})$ and $P(i_t^j = \texttt{Active})$. We define the slots activated on turn t as $S_t = \{s_t^j | P(s_t^j = \texttt{Active}) \geq \alpha,\ j \in [1, N_S]\}$ and the intents activated on turn t as $I_t = \{i_t^k | P(i_t^k = \texttt{Active}) \geq \alpha,\ k \in [1, N_I]\}$. s_t^j and i_t^k means the j-th slot name and the k-th intent name, respectively. α is a threshold to classify the slots and intents based on the probabilities. Activated slots S_t and intents I_t classified on this step are used to generate the dialogue state. On the generation step, S_t and I_t further contains slot vectors and intent vectors, which are calculated on the encoding step, in addition to the names in the text form. For example, S_t = {restaurant_name, price_range} can be represented as

$$\begin{aligned} &E(\text{“}\texttt{Slots:\{restaurant\_name:}\text{”}) \oplus \\ &v_S^1 \oplus E(\text{“}\texttt{;price\_range:}\text{”}) \oplus v_S^2 \oplus E(\text{“}\texttt{\}}\text{”}) \end{aligned} \quad (12)$$

before being fed into GPT-2, where E is the embedding layer to project the slot names into vector space of the same size as the slot vectors. To generate the dialogue state that consists of the slot-value pairs, the system should recognize not only the values but also the name of slots. The values can be extracted from user utterances, and which slots are activated can be predicted by the classification step; however, the exact slot names should be provided in the text form to construct slot-value structure matching given schema. By this process, the system can recognize the name of activated slots and intents before generating the dialogue state.

3.3 Dialogue State Generation

SET-DST has the state generator that generates dialogue state using the dialogue history, schema representation, and previous dialogue state accumulated during the conversation. In this paper, we define the dialogue state as a list of slot-value pairs that mean the details of an user goal. We also define the concept called user state that is a sequence

[1]The pre-trained models are available at https://github.com/huggingface/transformers.

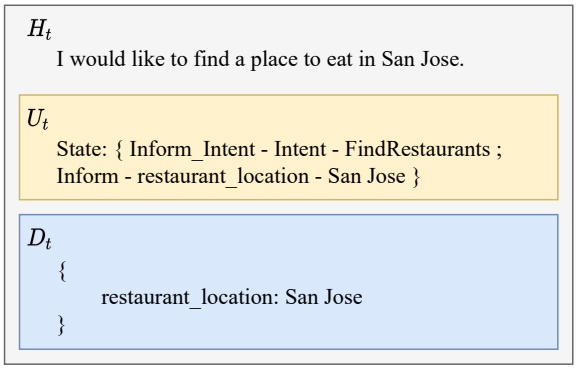

(a) Example in restaurants domain.

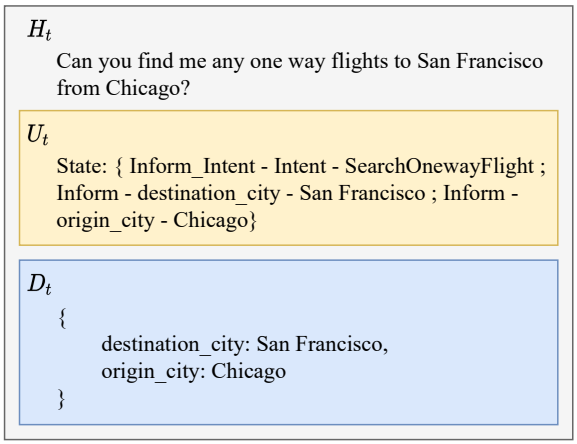

(b) Example in flights domain.

Figure 3: Examples of user state and dialogue state corresponding to user utterance. U_t is a sequence of words, and D_t is a list of slot-value pairs.

of action-slot-value triples to generalize semantics from various user utterances. The state generator recurrently generates the user state as a sequence of words, instead of generating the dialogue state in the structured form directly. Then, the dialogue state is updated by extracting the slot-value pairs from the user state. The user state U_t on turn t is generated based on the previous dialogue state D_{t-1}, dialogue history H_t, active slots S_t, and active intents I_t as follows:

$$\tilde{u}_t^l = \texttt{GPT-2}\left(D_{t-1} \oplus H_t \oplus S_t \oplus I_t \oplus U_t^{1:l-1}\right), \tag{13}$$

$$U_t = \left\{ u_t^l \middle| u_t^l = \texttt{argmax}\left(W_{vocab} \cdot \tilde{u}_t^l\right), \; l \in [1, N_U] \right\} \in \mathbb{R}^{N_U}, \tag{14}$$

where $U_t^{1:l-1} = \{u_t^1, \cdots, u_t^{l-1}\}$ and $N_U = |U_t|$. u_t^l means the l-th word of the user state, and $W_{vocab} \in \mathbb{R}^{N_{vocab} \times h}$ is an FC layer to project the hidden state to vocabulary space with size of N_{vocab}. Figure 3 shows how to generate D_t from U_t. U_t is generated word-by-word over time steps until `[EOS]`, a special word to terminate the generation, is detected. Then, D_t is updated by extracting the slot-value pairs from U_t. In task-oriented dialogue system, the dialogue state is used to call API. In Figure 3a, `San Jose` is passed as the value of an argument named `restaurant_location` to call the API named `FindRestaurants`. However, we aim not to build the full task-oriented dialogue system but to generate the dialogue state in this study.

3.4 Optimization

SET-DST is optimized over two steps: (1) slot and intent classification, and (2) state generation. We freeze the pre-trained BERT during training to preserve the broad and general knowledge that is learned from large corpus. In classification task, the system is trained by using binary cross-entropy. Equation 9 is used to calculate the slot loss $\mathcal{L}_t^S$ with slot labels $Y_t^S = \{y_{t,1}^S, \cdots, y_{t,N_S}^S\}$ as

$$\begin{aligned}\mathcal{L}_t^S = -\frac{1}{N_S}\sum_{j=1}^{N_S} \beta \cdot y_{t,j}^S \cdot \log P\left(s_t^j\right) \\ + \left(1 - y_{t,j}^S\right) \log\left(1 - P\left(s_t^j\right)\right),\end{aligned} \tag{15}$$

where $y_{t,j}^S \in \mathbb{R}^1$ is the binary value of j-th slot on turn t, and β is a hyperparameter to consider the ratio of active slots out of total slots. Based on Equation 10, the intent loss $\mathcal{L}_t^I$ is calculated with intent labels $Y_t^I = \{y_{t,1}^I, \cdots, y_{t,N_I}^I\}$ as

$$\begin{aligned}\mathcal{L}_t^I = -\frac{1}{N_I}\sum_{k=1}^{N_I} \beta \cdot y_{t,k}^I \cdot \log P\left(i_t^k\right) \\ + \left(1 - y_{t,k}^I\right) \log\left(1 - P\left(i_t^k\right)\right),\end{aligned} \tag{16}$$

where $y_{t,k}^I \in \mathbb{R}^1$ is the binary value of k-th intent on turn t. In state generation step, the system is trained as a conditional language model that recurrently generates words over time steps. The state loss $\mathcal{L}_t^U$ is calculated base on Equation 14 with the state label $Y_t^U = \{y_{t,l}^U, \cdots, y_{t,N_U}^U\}$ as

$$\mathcal{L}_t^U = -\frac{1}{N_U}\sum_{l=1}^{N_U} \left(y_{t,l}^U\right)^T \log P\left(u_t^l\right), \tag{17}$$

where $y_{t,l}^U \in \mathbb{R}^{N_{vocab}}$ is the one-hot vector that indicates the l-th word of the gold-standard user

state on turn t. The final joint loss is the sum of above losses:

$$\mathcal{L}_t = \mathcal{L}_t^S + \mathcal{L}_t^I + \mathcal{L}_t^U. \quad (18)$$

We use Adam optimizer (Kingma and Ba, 2014) to minimize $\mathcal{L}_t$.

4 Experiments

In this section, we describe our experiments including the datasets, evaluation metric, and results.

4.1 Experimental Setups

We used two datasets MultiWOZ 2.1[2] and Schema-Guided Dialogue (SGD)[3] to evaluate our system. MultiWOZ consists of conversations between a tourist and a guide, e.g., booking hotels and searching trains. SGD deals with conversations between a virtual assistant and an user ranging over various domains, e.g., events, restaurants, and media. The dataset also provides a schema that includes services, intents, and slots with short descriptions to help understanding the conversations. In this study, we followed the schema proposed in SGD. MultiWOZ has about 10,400 dialogues, and SGD has about 22,800 dialogues.

The datasets propose joint accuracy as the metric to evaluate DST systems. Joint accuracy measures whether a system successfully predicts all slot-value pairs mentioned on the conversations. In every turn, the system updates dialogue state, and the joint accuracy is calculated based on the accumulated dialogue state.

4.2 Experimental Details

The motivation of SET-DST is to make the system interpret the schema and refer it for efficiently tracking the dialogue state. In the experiments, our goal is to verify that SET-DST works well for our purpose by improving the performance of DST and the efficiency on few-shot settings with the schema encoding.

The experiments are divided into two steps: (1) pre-training on SGD and (2) fine-tuning on MultiWOZ. In the pre-training step, SET-DST is optimized to encode the schema for DST. In the fine-tuning step, the capability that encodes given schema is transferred to encode new schema for improvement of the performance and efficiency. We conducted the experiments by adjusting the rate of few-shot data during fine-tuning to focus on the fine-tuning step. The training data for few-shot settings was randomly sampled from the training set of MultiWOZ, and the random seed was fixed for consistency of sampling. We also conducted experiments to verify whether SET-DST successfully works on the pre-training step, although the major part in our experiments is the fine-tuning on MultiWOZ including few-shot settings.

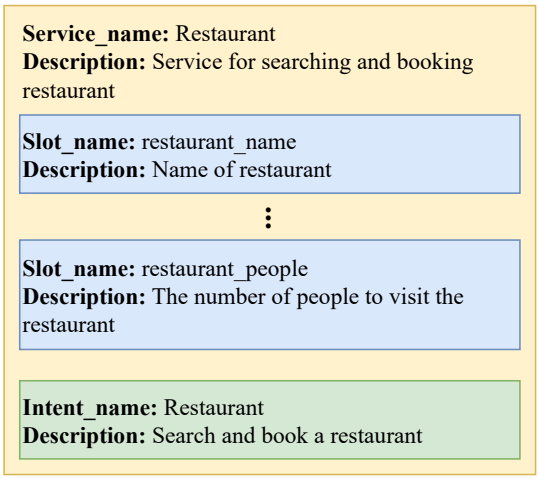

Figure 4: Example of schema that is temporarily created for MultiWOZ dataset.

SET-DST needs not only slot information but also a schema. However, MultiWOZ has no schema and no concepts of service and intent; thus, we created a schema for MultiWOZ including services, slots, intents, and corresponding descriptions. Figure 4 shows an example of the schema for MultiWOZ. In our experiments on MultiWOZ, an intent means activated domain. In other words, the system classifies an intent as active when the domain of conversation is changed or a conversation starts. MultiWOZ further has no labels for activated intents, thus we automatically added the labels by tracking active domains and detecting whether new domains are active.

We further tried to fine-tune the system without intents because it is possible that the concepts of intent are unnatural in MultiWOZ. In this setting, Equation 3, 6, 7, 10, 16 are ignored, I_t is removed from Equation 13, and $\mathcal{L}_t^I$ is removed from Equation 18. In Figure 3a, U_t is replaced with `State: { Inform - restaurant_location - San Jose },`

[2] https://github.com/budzianowski/multiwoz.

[3] https://github.com/google-research-datasets/dstc8-schema-guided-dialogue.

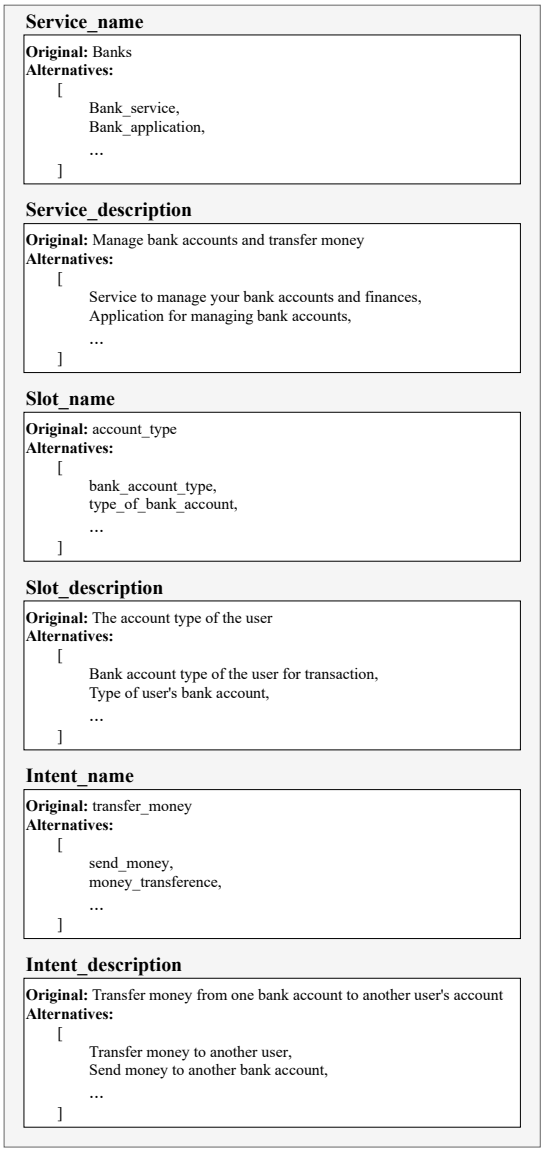

Figure 5: Example of alternatives for schema augmentation.

and in Figure 3b, U_t is replaced with `State: { Inform - destination_city - San Francisco ; Inform - origin_city - Chicago }`.

A dataset schema can be variously defined depending on the developer, and our goal is to make the system represent any schema for DST. In the pre-training step, we manually augmented the schema of SGD dataset to avoid overfitting to the given schema. The schema provides names of services, slots, and intents with short descriptions. We defined some alternatives of the names and descriptions, and sampled inputs for the schema encoder from the augmented schema. Figure 5 shows some examples of the alternatives for bank service.

In dialogue state, multiple slots and intents can be activated at once. However, the order has no meaning in dialogue state. The state generator is trained to generate the dialogue state based on textual label, so it is possible that the order causes wrong optimization and overfitting. Thus, we shuffled the order of slots and intents when making the labels.

	JA
TRADE (Wu et al., 2019)	45.60%*
DSTQA (Zhou and Small, 2019)	51.17%
LABES-S2S (Zhang et al., 2020b)	51.45%
DST-Picklist (Zhang et al., 2020a)	53.30%
MinTL-BART (Lin et al., 2020)	53.62%
TripPy (Heck et al., 2020)	55.29%
SimpleTOD (Hosseini-Asl et al., 2020)	55.76%
PPTOD (Su et al., 2021)	57.45%
ConvBERT-DG (Mehri et al., 2020)	58.70%
TripPy+SCoRe (Yu et al., 2020)	60.48%
TripPy+CoCoAug (Li et al., 2020)	60.53%
TripPy+SaCLog (Dai et al., 2021)	60.61%
SET-DST (Ours)	60.39%
SET-DST w/o intent	**62.07**%

Table 1: DST results on the test set of MultiWOZ in joint accuracy. *: the result is reported by Eric et al. (2020).

We used BERT-base-uncased[4] model for the schema encoder and GPT-2[5] model for the state generator. In our experiments, the pre-training step took about two days, and the fine-tuning step took about a day on a TitanRTX GPU. Table 3 lists hyperparameters that are used in our experiments.

4.3 Experimental Results

Table 1 compares the evaluation results of SET-DST to the previous methods on the test set of MultiWOZ. In our experiments, SET-DST achieved new state-of-the-art joint accuracy when fine-tuned without intent.

Table 2 shows the evaluation results on few-shot settings and the improvement by pre-training. When we used less training data, the pre-training with schema encoding was more effective for DST. SET-DST performed reasonably well with only about 20% of the training data. In most cases, the models fine-tuned without intents achieved higher joint accuracy on MultiWOZ.

[4] https://huggingface.co/bert-base-uncased.

[5] https://huggingface.co/gpt2.

Few-shot rate		JA	
		w/ intent	w/o intent
w/ pre-training	100%	60.39%	62.07%
	30%	53.43%	56.43%
	25%	53.07%	55.61%
	20%	52.73%	54.37%
	15%	40.41%	51.29%
	10%	31.20%	29.91%
w/o pre-training	100%	58.37%	59.10%
	30%	31.08%	48.96%
	25%	30.39%	30.28%
	20%	22.80%	22.35%
	15%	19.38%	17.09%
	10%	10.21%	15.10%

Table 2: Evaluation results on few-shot settings with considering pre-training.

5 Discussion

Schema Encoding In this study, our goal is to transfer a pre-trained DST model to a low-resource domain without limiting the transference as language model level by using schema encoding. We pre-trained SET-DST on SGD which is a relatively large dataset and fine-tuned it on MultiWOZ to transfer the schema encoding. As a result, the pre-training significantly improved the accuracy on DST. Table 2 shows that the pre-training was more effective when the target dataset was small. To satisfy the joint accuracy evaluation, the system should not only predict values, but also exactly match the names of slots defined on the dataset. We believe that the tuning process was not completed when we used just 10% of the target dataset; on the other side, the system could match the slot names defined on MultiWOZ not SGD with 20% of the dataset. Pre-trained language models have been already used in many fields. However, our method could tackle general DST beyond language modeling on various domains. We believe that SET-DST can assist the development of DST systems in real world without large dataset on the target domain.

Intent on Fine-tuning In this paper, we define the intents as sub-goals to be achieved through a service. SGD has a schema for dialogues between a virtual assistant and an user. Thus, it is assumed that a system achieves the user's sub-goals by using APIs, and an intent corresponds to an API. Virtual assistant should tackle various services that could consist of one more intents, e.g., to check account balance and to transfer money in bank service. Unlike that, MultiWOZ has no schema and considers no APIs as intents. Thus, the schema that we temporarily created for experiments in the same form as the schema of SGD could cause confusion in generation of dialogue state. We believe that this is why the results without intent were slightly higher in the experiments. Another reason would be the incorrect labels for intents that we automatically created for experiments on MultiWOZ.

The joint accuracy that has been proposed as an evaluation metric for DST considers only slot-value pairs. However, task-oriented dialogue systems should call APIs of external systems to achieve goals, e.g., to search restaurants and to reserve hotels. The systems that predict only slot-value pairs would be insufficient to replace rule-based traditional systems in real-world. Even though use of intents made no improvement in joint accuracy, we believe that encoding the schema including intents is meaningful in terms of approaching more realistic DST.

Backbone Model We used BERT for schema encoding and GPT-2 for classification and generation. Using only GPT-2 was an option in our study. However, we fixed the pre-trained BERT during optimization to preserve the broad and general knowledge learned on pre-training and to use the knowledge for encoding given schema. BERT is an excellent backbone model to encode text for downstream tasks; thus, we used BERT instead of GPT-2 for the schema encoding.

6 Conclusion

Transfer learning that makes it possible to apply a pre-trained model to new domains has been attempted a lot. However, the attempts for DST have been just to use large-scale pre-trained models as language models. In this paper, we have proposed SET-DST, which is an effective method for DST with transfer learning by using schema encoding. We have demonstrated how to encode the schema for transferable DST and how to use the schema representation for dialogue state generation. Our experiments show that the schema encoding improves joint accuracy even in few-shot settings.

Even though our approach could perform DST well on target domain with few-shot settings, it required some new data to be fine-tuned. As part of our future work, we plan to design a DST model for zero-shot settings.

Acknowledgements

This work was supported by Institute for Information &communications Technology Planning & Evaluation (IITP) grant funded by the Korea government (MSIT) (No. 2021-0-00354, Artificial intelligence technology inferring issues and logically supporting facts from raw text), and by the MSIT (Ministry of Science and ICT), Korea, under the ITRC (Information Technology Research Center) support program (IITP-2022-2020-0-01789) supervised by the IITP (Institute for Information & Communications Technology Planning & Evaluation).

References

Lu Chen, Boer Lv, Chi Wang, Su Zhu, Bowen Tan, and Kai Yu. 2020. Schema-guided multi-domain dialogue state tracking with graph attention neural networks. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pages 7521–7528.

Yinpei Dai, Hangyu Li, Yongbin Li, Jian Sun, Fei Huang, Luo Si, and Xiaodan Zhu. 2021. Preview, attend and review: Schema-aware curriculum learning for multi-domain dialog state tracking. *arXiv preprint arXiv:2106.00291*.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. Bert: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186.

Mihail Eric, Rahul Goel, Shachi Paul, Abhishek Sethi, Sanchit Agarwal, Shuyang Gao, Adarsh Kumar, Anuj Goyal, Peter Ku, and Dilek Hakkani-Tur. 2020. Multiwoz 2.1: A consolidated multi-domain dialogue dataset with state corrections and state tracking baselines. In *Proceedings of The 12th Language Resources and Evaluation Conference*, pages 422–428.

Michael Heck, Carel van Niekerk, Nurul Lubis, Christian Geishauser, Hsien-Chin Lin, Marco Moresi, and Milica Gasic. 2020. Trippy: A triple copy strategy for value independent neural dialog state tracking. In *Proceedings of the 21th Annual Meeting of the Special Interest Group on Discourse and Dialogue*, pages 35–44.

Ehsan Hosseini-Asl, Bryan McCann, Chien-Sheng Wu, Semih Yavuz, and Richard Socher. 2020. A simple language model for task-oriented dialogue. *arXiv preprint arXiv:2005.00796*.

Sungdong Kim, Sohee Yang, Gyuwan Kim, and Sang-Woo Lee. 2020. Efficient dialogue state tracking by selectively overwriting memory. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 567–582.

Diederik P Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*.

Hwaran Lee, Jinsik Lee, and Tae-Yoon Kim. 2019. Sumbt: Slot-utterance matching for universal and scalable belief tracking. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 5478–5483.

Shiyang Li, Semih Yavuz, Kazuma Hashimoto, Jia Li, Tong Niu, Nazneen Rajani, Xifeng Yan, Yingbo Zhou, and Caiming Xiong. 2020. Coco: Controllable counterfactuals for evaluating dialogue state trackers. *arXiv preprint arXiv:2010.12850*.

Zhaojiang Lin, Andrea Madotto, Genta Indra Winata, and Pascale Fung. 2020. Mintl: Minimalist transfer learning for task-oriented dialogue systems. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 3391–3405.

Shikib Mehri, Mihail Eric, and Dilek Hakkani-Tur. 2020. Dialoglue: A natural language understanding benchmark for task-oriented dialogue. *arXiv preprint arXiv:2009.13570*.

Elnaz Nouri and Ehsan Hosseini-Asl. 2018. Toward scalable neural dialogue state tracking model. *arXiv preprint arXiv:1812.00899*.

Sungjoon Park, Jihyung Moon, Sungdong Kim, Won Ik Cho, Jiyoon Han, Jangwon Park, Chisung Song, Junseong Kim, Yongsook Song, Taehwan Oh, et al. 2021. Klue: Korean language understanding evaluation. *arXiv preprint arXiv:2105.09680*.

Baolin Peng, Chunyuan Li, Jinchao Li, Shahin Shayandeh, Lars Liden, and Jianfeng Gao. 2021. Soloist: Building task bots at scale with transfer learning and machine teaching. *Transactions of the Association for Computational Linguistics*, 9:907–824.

Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. 2019. Language models are unsupervised multitask learners.

Abhinav Rastogi, Xiaoxue Zang, Srinivas Sunkara, Raghav Gupta, and Pranav Khaitan. 2020. Towards scalable multi-domain conversational agents: The schema-guided dialogue dataset. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pages 8689–8696.

Liliang Ren, Kaige Xie, Lu Chen, and Kai Yu. 2018. Towards universal dialogue state tracking. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 2780–2786.

Yixuan Su, Lei Shu, Elman Mansimov, Arshit Gupta, Deng Cai, Yi-An Lai, and Yi Zhang. 2021. Multi-task pre-training for plug-and-play task-oriented dialogue system. *arXiv preprint arXiv:2109.14739*.

Zhuoran Wang and Oliver Lemon. 2013. A simple and generic belief tracking mechanism for the dialog state tracking challenge: On the believability of observed information. In *Proceedings of the SIGDIAL 2013 Conference*, pages 423–432.

Jason D Williams. 2014. Web-style ranking and slu combination for dialog state tracking. In *Proceedings of the 15th Annual Meeting of the Special Interest Group on Discourse and Dialogue (SIGDIAL)*, pages 282–291.

Chien-Sheng Wu, Andrea Madotto, Ehsan Hosseini-Asl, Caiming Xiong, Richard Socher, and Pascale Fung. 2019. Transferable multi-domain state generator for task-oriented dialogue systems. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 808–819.

Tao Yu, Rui Zhang, Alex Polozov, Christopher Meek, and Ahmed Hassan Awadallah. 2020. Score: Pre-training for context representation in conversational semantic parsing. In *International Conference on Learning Representations*.

Jianguo Zhang, Kazuma Hashimoto, Chien-Sheng Wu, Yao Wang, S Yu Philip, Richard Socher, and Caiming Xiong. 2020a. Find or classify? dual strategy for slot-value predictions on multi-domain dialog state tracking. In *Proceedings of the Ninth Joint Conference on Lexical and Computational Semantics*, pages 154–167.

Yichi Zhang, Zhijian Ou, Min Hu, and Junlan Feng. 2020b. A probabilistic end-to-end task-oriented dialog model with latent belief states towards semi-supervised learning. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 9207–9219.

Victor Zhong, Caiming Xiong, and Richard Socher. 2018. Global-locally self-attentive encoder for dialogue state tracking. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1458–1467.

Li Zhou and Kevin Small. 2019. Multi-domain dialogue state tracking as dynamic knowledge graph enhanced question answering. *arXiv preprint arXiv:1911.06192*.

Su Zhu, Jieyu Li, Lu Chen, and Kai Yu. 2020. Efficient context and schema fusion networks for multi-domain dialogue state tracking. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: Findings*, pages 766–781.

A Hyperparameters

We list the hyperparameters used in our experiments for reproducibility.

Hidden size	768
Embedding size	768
Vocabulary size	30522
Dropout	0.3
Early stopping count	5
Max epochs	40
Min epochs	20
Batch size	8
Learning rate	3e-5
Gradient clipping	10
α	0.5
β (on SGD)	3
β (on MultiWOZ)	5

Table 3: Hyperparameters used for the experiments in this paper.

B User Actions Set

Table 4 lists the user actions covered in this paper. When the user state U_t is generated, each element has three types: (1) action-slot-value triple, (2) action-slot pair, and (3) only action; e.g., (1) `Inform-restaurant_location-SanJose`, (2) `Request-restaurant_address`, and (3) `Negate`.

Action Name	Need Slot	Need Value
Inform	✓	✓
Inform_Intent*	✓	✓
Request	✓	×
Request_Alts	×	×
Affirm	×	×
Affirm_Intent*	✓	✓
Select	✓	✓
Negate	×	×
Negate_Intent	×	×
Thank_You	×	×
Goodbye	×	×

Table 4: List of user actions covered in this paper. * just need an intent as the value, but we added a dummy slot, `Intent`, to keep the shape of action-slot-value triple; e.g., we used `Inform_Intent-Intent-FindRestaurants` instead of `Inform_Intent-FindRestaurants` when we make the user state.

C Pre-traing Results

We evaluated the pre-training performance on KLUE[6] dataset (Park et al., 2021), which is a Korean dataset for DST, in addition to SGD. Table 5 shows the pre-training results on SGD and KLUE. SET-DST outperformed the baselines. These results demonstrate that SET-DST successfully performs DST with just pre-training. In the experiment on KLUE, we used KLUE-BERT[7] and KoGPT-2[8] that are large-scale language models pre-trained on Korean corpus.

		JA
SGD	Baseline (Rastogi et al., 2020)	25.40%
	SET-DST (Ours)	55.56%
KLUE	Baseline (Park et al., 2021)	50.22%
	SET-DST (Ours)	57.61%

Table 5: Pre-training results on SGD and KLUE compared to their baselines.

D Example of dialogues

We add an example of dialogue with the corresponding user state U_t and dialogue state D_t. Table 6 shows an dialogue on three domains: hotels, flights, and attractions.

[6] https://github.com/KLUE-benchmark/KLUE.

[7] https://huggingface.co/klue/bert-base.

[8] https://huggingface.co/skt/kogpt2-base-v2.

User:	I want to find me a 4 star hotel in Sydney.
U_t:	Inform_Intent - Intent - SearchHotel ; Inform - hotel_location - Sydney ; Inform - star_rating - 4
D_t:	{ **hotel_location: Sydney**, **star_rating: 4** }
System:	I found 10 option that you may like. A 4 star hotel called 28 Hotel Sydney.
User:	Can you smoke here? Tell me the address.
U_t:	Request - smoking_allowed ; Request - street_address
D_t:	{ hotel_location: Sydney, star_rating: 4 }
System:	No, smoking is not allowed here. 28 Regent Street, Chippendale New South Wales 2008, Australia is their address.
User:	Sounds okay. Find me a round trip flight to get there. I will start my trave on day after tomorrow and I want Premium Economy class tickets.
U_t:	Select - place_name - 28 Hotel Sydney ; Inform_Intent - Intent - SearchRoundtripFlights ; Inform - flights_departure_date - day after tomorrow ; Inform - seating_class - Premium Economy
D_t:	{ hotel_location: Sydney, star_rating: 4, **place_name: 28 Hotel Sydney**, **flights_departure_date: day after tomorrow**, **seating_class: Premium Economy** }
System:	On which day you will return?
User:	12th of March is the date on which I will return.
U_t:	Inform - return_date - 12th of March
D_t:	{ hotel_location: Sydney, star_rating: 4, place_name: 28 Hotel Sydney, flights_departure_date: day after tomorrow, seating_class: Premium Economy **return_date: 12th of March** }
System:	From which city you want to depart?
User:	From Las Vegas.
U_t:	Inform - origin_airport - Lag Vegas
D_t:	{ hotel_location: Sydney, star_rating: 4, place_name: 28 Hotel Sydney, flights_departure_date: day after tomorrow, seating_class: Premium Economy return_date: 12th of March, **origin_airport: Las Vegas** }
System:	I have 1 flight for you. You like United Airlines? The onward flight takes off at 4:30 am and return is take off at 4:55 pm. It has a layover and ticket cost $697.
User:	Good. Find me some attractions there that have free entry and is child-friendly.
U_t:	Inform_Intent - Intent - FindAttractions ; Inform - free_entry - True ; Inform - good_for_kids - True ; Inform - category - dontcare
D_t:	{ hotel_location: Sydney, star_rating: 4, place_name: 28 Hotel Sydney, flights_departure_date: day after tomorrow, seating_class: Premium Economy return_date: 12th of March, origin_airport: Las Vegas, **free_entry: True** **good_for_kids: True**, **category: dontcare** }
System:	You can check out a Sports Venue called ANZ Stadium.
User:	Good. Tell me their phone number.
U_t:	Select - attraction_name - ANZ Stadium ; Request - phone_number
D_t:	{ hotel_location: Sydney, star_rating: 4, place_name: 28 Hotel Sydney, flights_departure_date: day after tomorrow, seating_class: Premium Economy return_date: 12th of March, origin_airport: Las Vegas, free_entry: True good_for_kids: True, category: dontcare, **attraction_name: ANZ Stadium** }
System:	2 9298 3777 is the phone number.
User:	Great. That's all that I wanted for now. Bye.
U_t:	GoodBye

Table 6: Example of dialogue including the user state and dialogue state that we defined in this paper.

A Personalized Dialogue Generator with Implicit User Persona Detection

Itsugun Cho[1] **Dongyang Wang**[1*] **Ryota Takahashi**[1*] **Hiroaki Saito**[1]
Keio University, Japan[1]
{choitsugun, wangdongyang, ryota.0226.tokky}@keio.jp
hxs@ics.keio.ac.jp

Abstract

Current works in the generation of personalized dialogue primarily contribute to the agent presenting a consistent personality and driving a more informative response. However, we found that the generated responses from most previous models tend to be self-centered, with little care for the user in the dialogue. Moreover, we consider that human-like conversation is essentially built based on inferring information about the persona of the other party. Motivated by this, we propose a novel personalized dialogue generator by detecting an implicit user persona. Because it is hard to collect a large number of detailed personas for each user, we attempted to model the user's potential persona and its representation from dialogue history, with no external knowledge. The perception and fader variables were conceived using conditional variational inference. The two latent variables simulate the process of people being aware of each other's persona and producing a corresponding expression in conversation. Finally, posterior-discriminated regularization was presented to enhance the training procedure. Empirical studies demonstrate that, compared to state-of-the-art methods, our approach is more concerned with the user's persona and achieves a considerable boost across both automatic metrics and human evaluations.

1 Introduction

Personalized dialogue modeling is an attractive research topic in deep learning, where studies have explored the possibility of incorporating personal facts into the end-to-end generative framework. The established practice of assigning agents a predefined character improves the engagingness and consistency of open-domain dialogue. However, such models cannot generate distinguishable responses while interacting with different users because they do not take into consideration who the other party is. As Shum et al. (2018) pointed out, a good chit-chat bot not only generates interesting responses but also resonates with interlocutors. However, there has been little research conducted into how to make the agent effectively mine a user's persona to generate customized responses.

Figure 1: An example of dialogue generation with the implicit persona detection. The incorporated persona and the corresponding user's real persona are in bold.

To this end, this research studied personalized dialogue generation in which we aimed to have the agent recognize the other party's potential persona by exploiting the dialogue itself and output personalized responses conditioned on the different target users. A simple illustration depicting this process is provided in Figure 1. Inspired by the impressive effectiveness of conditional variational autoencoders (CVAEs) (Sohn et al., 2015; Zhao et al., 2017) with diverse response modeling, we propose a personalized dialogue generator that detects an implicit user persona using conditional variational inference. Specifically, our model fits the profile descriptions of the other party to a multivariate isotropic Gaussian distribution using a latent variable (perception variable) during training. Because responses from the real-world dialogue are not always persona-related (i.e., persona-sparse issue; Zheng et al., 2020), we also introduce another

*Equal contribution.

Proceedings of the 29th International Conference on Computational Linguistics, pages 367–377
October 12–17, 2022.

latent variable (fader variable) to control the weight of persona-related aspects exhibited in the response. During inference, the decoder is designed to acquire the persona features from the perception and fader variables to produce a response that incorporates the user's various potential persona information inferred from the context. Note that the textual profiles are only leveraged during training that is tasked with learning the latent distribution over the user's persona. And during inference, the raw observed data that yields latent variables only includes the context without the explicit persona.

We argue that it is impractical to collect a large quantity of available profiles involved with specific users. Thus, our model has better universality than methods that require providing extra information as generation material. CVAEs have been proved to improve the response diversity at the discourse level (i.e., one-to-many nature; Zhao et al., 2017). Our model achieves "one context to many responses" by sampling and reconstructing with stochasticity for persona distribution and responses, just as we can initiate different chats with a user from aspects of the user's persona. Experimental results on the ConvAI2 dataset demonstrate the superiority of the proposed model over the baselines in both automatic metrics and human evaluations. The interpretability and effectiveness of our approach are clarified in the discussion. The main contributions of this paper can be summarized as:
(1) To the best of our knowledge, this is the first attempt to build a user-targeted personalized dialogue agent via conditional variational inference, which not only proposes a new model but also provides insight into manners of latent information mining and representation.
(2) A new training scheme is designed to mitigate the disastrous local optimum issue that often occurs in the Bayesian architecture on text generation tasks. Evaluation reveals our scheme yielded better performance than previous strategies.
(3) Empirical verification was carried out both quantitatively and qualitatively and confirmed the high levels of convincingness of our model.

2 Methodology

2.1 Problem Scenario

The task can be formally defined as a dialogue corpus $\mathscr{C} = (C_i, R_i, P_i)_{i=1}^{n}$, where C_i refers to a context that includes multiple utterances, with R_i a response and P_i a textual profile containing multiple descriptions of the other party (i.e., the target interlocutor of R_i). Our goal is that, by learning the potential dependencies among P, C, and R from $\mathscr{C}$, one can generate diverse responses $\bar{R} = (\bar{R}_1, \bar{R}_2, ..., \bar{R}_m)$ for a new context $\bar{C}$. $\bar{R}$ is expected to be relevant to the other party's real persona, which means the mutual information should be maximized as much as possible. Moreover, in cases where $\bar{C}$ is the persona-sparse context, $\bar{R}$ should mainly cohere with the context.

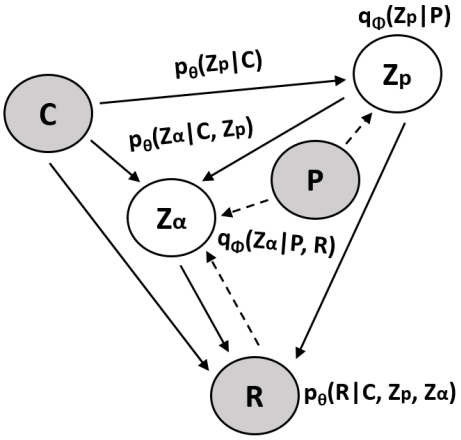

Figure 2: The solid lines are conditional dependencies and dashed lines denote variational approximation. The profile P, context C, and response R are observed data. The variational parameters ϕ are learned jointly with the conditional parameters θ.

2.2 Overview

As described in the introduction, our approach incorporates a pair of latent variables utilized for bridging the potential dependencies among P, C, and R. Perception variable Z_p is adopted to capture the latent distribution over P that constructs a connection between C and R by the user's implicit persona. Fader variable Z_α is adopted to indicate how much persona information in Z_p is carried by R under C. Figure 2 gives the directed graphical model of our approach. The conditional distribution over the above variables can be factorized as $p(R, Z_p, Z_\alpha|C) = p(R|C, Z_p, Z_\alpha)p(Z_\alpha|C, Z_p)p(Z_p|C)$. Our objective is to represent it with deep neural networks, where we denote $p_\theta(R|C, Z_p, Z_\alpha)$ as a response decoder and $p_\theta(Z_p|C)$ and $p_\theta(Z_\alpha|C, Z_p)$ as the prior networks. $p(R, Z_p, Z_\alpha|C)$ depicts a process that is from the prior networks to draw out implicit persona and its representation from C, prompting the response decoder to restore R under the information only sourced in C. Thereby, we would

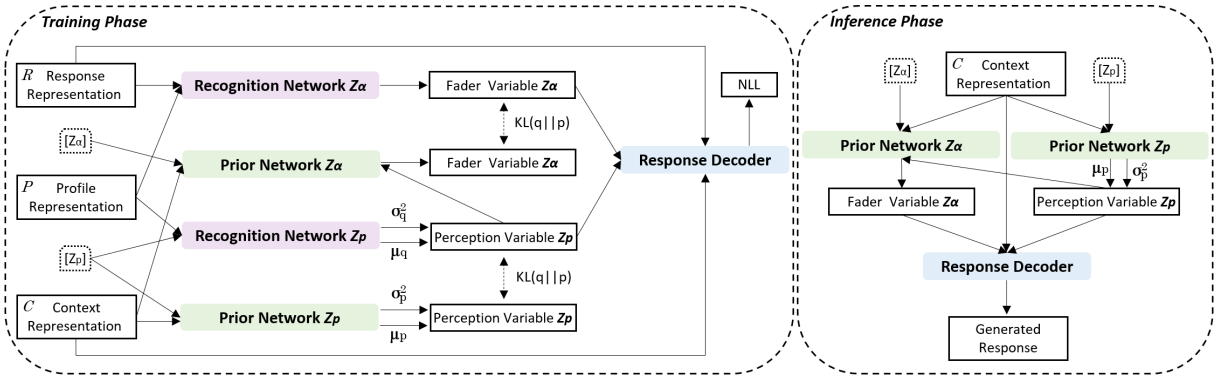

Figure 3: Illustration of the model architecture. The two prior networks share parameters.

maximize the conditional likelihood $p_\theta(R|C) = \iint p_\theta(R|C, Z_p, Z_\alpha)p_\theta(Z_\alpha|C, Z_p)p_\theta(Z_p|C)dZ_p dZ_\alpha$.

However the marginalization over Z_p and Z_α are intractable integrals (i.e., a context theoretically corresponds to a continuous user persona space). Hence, our model is trained with the stochastic gradient variational Bayes (SGVB) framework (Kingma and Welling, 2013) by maximizing the variational lower bound. According to the above definition of the perception and fader variables, we refer to variational distribution $q_\phi(Z_p|P)$ and $q_\phi(Z_\alpha|P, R)$ as the recognition networks to approximate the true posterior $p(Z_p|C, R) \propto p(R|C, Z_p)p(Z_p|C)$ and $p(Z_\alpha|C, R, Z_p) \propto p(R|C, Z_p, Z_\alpha)p(Z_\alpha|C, Z_p)p(Z_p|C)$, respectively. The evidence lower bound (ELBO) of our approach can be deduced as follows:

$$\begin{aligned}&\mathcal{L}(\theta, \phi; P, C, R) = \\ &- KL(q_\phi(Z_p|P)||p_\theta(Z_p|C)) \\ &- KL(q_\phi(Z_\alpha|P, R)||p_\theta(Z_\alpha|C, Z_p)) \\ &+ \mathbb{E}_{q_\phi(Z_p|P);q_\phi(Z_\alpha|P,R)}[\log p_\theta(R|C, Z_p, Z_\alpha)]\end{aligned} \tag{1}$$

where $KL(\cdot||\cdot)$ denotes the KL divergence. Details about the derivation are provided in Appendix A.

2.3 Model Details

Figure 3 shows the architecture of our model. We define the input representation as follows:
(1) The input embedding of each token is the sum of corresponding word embedding and position embedding. To differentiate the user character in dialogue history, we add role embedding into the utterances generated by the other party. With minor exploitation of notation, we also use P, C, and R to denote input representations in the following.
(2) The different utterances in context or different descriptions in the profile are separated by the special token [SEP]. The beginning and end of the context or profile are appended with the special tokens [BOS] and [EOS], respectively.
(3) The special token of the perception variable and the fader variable are denoted as $[Z_p]$ and $[Z_\alpha]$, respectively. For the special token of the latent variable, the position embedding is set to empty.

We hypothesize the perception variable follows multivariate Gaussian distribution with a diagonal covariance matrix. The input representations $concat([Z_p], C)$ and $concat([Z_p], P)$ are fed to the prior network $p_\theta(Z_p|C) \backsim \mathcal{N}(\boldsymbol{\mu}_p, \boldsymbol{\sigma}_p^2\mathbf{I})$ and the recognition network $q_\phi(Z_p|P) \backsim \mathcal{N}(\boldsymbol{\mu}_q, \boldsymbol{\sigma}_q^2\mathbf{I})$, respectively, where $concat(\cdot, \cdot)$ denotes concatenation. Both networks are three-layer transformer encoders (Vaswani et al., 2017) with a two-layer fully connected network. The means $\boldsymbol{\mu}_p$, $\boldsymbol{\mu}_q$ and variances $\boldsymbol{\sigma}_p^2$, $\boldsymbol{\sigma}_q^2$ are derived as follows:

$$\begin{bmatrix}\boldsymbol{\mu}_p \\ \log(\boldsymbol{\sigma}_p^2)\end{bmatrix} = \boldsymbol{W}_p\, \boldsymbol{h}_{[Z_p]} + \boldsymbol{b}_p \tag{2}$$

$$\begin{bmatrix}\boldsymbol{\mu}_q \\ \log(\boldsymbol{\sigma}_q^2)\end{bmatrix} = \boldsymbol{W}_q\, \boldsymbol{h}_{[Z_p]} + \boldsymbol{b}_q \tag{3}$$

where $\boldsymbol{h}_{[Z_p]} \in \mathbb{R}^D$ is the final hidden state of $[Z_p]$ from the transformer encoder, and $\boldsymbol{W}_p \in \mathbb{R}^{K\times D}$, $\boldsymbol{W}_q \in \mathbb{R}^{K\times D}$, and $\boldsymbol{b}_p \in \mathbb{R}^K$, $\boldsymbol{b}_q \in \mathbb{R}^K$ denote the weight matrices of the fully connected network. We obtain samples of the perception variable from $\mathcal{N}(\boldsymbol{\mu}_q, \boldsymbol{\sigma}_q^2\mathbf{I})$ during training or $\mathcal{N}(\boldsymbol{\mu}_p, \boldsymbol{\sigma}_p^2\mathbf{I})$ during inference. As sampling is not differentiable, the reparametrization trick (Kingma and Welling, 2013) is employed for effective training.

The input representation $concat(Z_p, [Z_\alpha], C)$ is fed to the prior network $p_\theta(Z_\alpha|C, Z_p)$, which is a three-layer transformer encoder, and the final hidden state of $[Z_\alpha]$ is specified as a fader variable. The recognition network $q_\phi(Z_\alpha|P, R)$

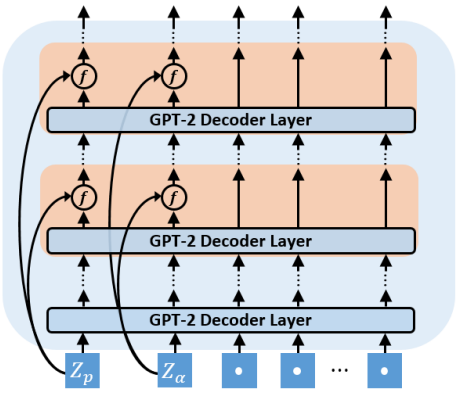

Figure 4: Over each multiple transformer layer (four layers in our experiments), a weighted sum $\boldsymbol{f}$ is operated between the hidden state of latent variables and the original latent variables. Both inputs of operation $\boldsymbol{f}$ are weighted 0.5 in our implementation.

without parameters concerns a similarity function of $(P_i, R_i)_{i=1}^n$ pairs. We obtain the fader variable from $q_\phi(Z_\alpha|P, R)$ during training or $p_\theta(Z_\alpha|C, Z_p)$ during inference. The response decoder $p_\theta(R|C, Z_p, Z_\alpha)$ is built by a GPT-2 pre-trained language model (Radford et al., 2019). The input representations $concat(Z_p, Z_\alpha, C, R)$ or $concat(Z_p, Z_\alpha, C)$ are fed to the response decoder during training or inference, respectively. Note that we put Z_p and Z_α before C, R, or C to form the input representations due to the autoregressive property of GPT-2. To facilitate the backpropagation of the perception and fader variables, and also to enhance the effect of these variational signals on generation in decoding, we considered an injection scheme that is illustrated in Figure 4.

2.4 Posterior-Discriminated Regularization

Training the text data with VAEs / CVAEs often falls into a trivial local optimum where the decoder learns to ignore the latent variable, causing the approximate posterior to mimic the prior. This phenomenon is referred to as "posterior collapse." The state-of-the-art solutions include re-weighting the KL term (KL annealing, cyclic annealing; Bowman et al., 2016; Fu et al., 2019), introducing a neural network to calculate bag-of-word (BOW) loss (Zhao et al., 2017), and modifying the training procedure (aggressive training; He et al., 2019).

Ideally, if the approximate posterior $q_\phi(Z|X) \backsim \mathcal{N}(\boldsymbol{\mu}_q, \boldsymbol{\sigma}_q^2\mathbf{I})$ (i.e., $q_\phi(Z_p|P)$ in our experiment) is perfect, Z is a non-trivial latent representation of input X, whereby we suppose that Z should be especially dissimilar for various posterior inputs. We designed a scheme augmenting the distinction of conditional posteriors that forces the decoder to reconstruct results from the latent variable whose features vary notably. We consider training an auxiliary cost with minimizing the following.

$$\mathcal{L}_{Po\text{-}di} = \sum_{i=1}^{n}(Min[KL(q_\phi(Z_i|X_i)||q_\phi(\bar{Z}_i|\bar{X}_i)) - \lambda, 0])^2 \quad (4)$$

where X_i denotes the i-th training data, and $\bar{X}_i$ refers to the input other than X_i. The distinction objective λ drives up KL divergence between the posteriors over different inputs. In our implementation, this computation is dealt with as mini-batch processing which the data is random sampling without replacement. The auxiliary cost can be added to ELBO to form the final loss function.

$$\mathcal{L}'(\theta, \phi; P, C, R) = \mathcal{L}(\theta, \phi; P, C, R) + \mathcal{L}_{Po\text{-}di} \quad (5)$$

Despite being conceptually simple, the benefit of this idea is that it is task-independent and easy to train without introducing new model components. $\mathcal{L}_{Po\text{-}di}$ achieves better performance, as we will detail in Section 4.3 by comparing the above methods.

3 Experiments

3.1 Corpus

We evaluated our approach on the ConvAI2 benchmark dataset, which is an extended version with a new hidden testing set of the PERSONA-CHAT dataset (Zhang et al., 2018). The dialogues were collected from crowd-workers who were asked to act as two interlocutors having a conversation to get to know each other. The persona of both interlocutors is explicitly described using several profile sentences. This dataset contains 17,878 / 1,000 multi-turn dialogues conditioned on 1,155 / 100 profiles for train / dev, each profile consisting of at least five descriptions. Because the testing set is hidden, we used the validation set as the testing set in our experiments and randomly sampled 500 dialogues from the training set for the validation. To suit our goals, we removed some self-centered utterances that only scratched the surface.

3.2 Baselines

The following five state-of-the-art generative baseline methods were considered in our experiments. **HRED** is a persona-free dialogue model built by

hierarchical RNN, proposed in Serban et al. (2016). This model is one of the traditional seq2seq architectures widely applied for comparison.

CVAE is a persona-free dialogue model utilizing a conditional variational autoencoder to learn a latent distribution over conversational factors. This model was proposed by Zhao et al. (2017).

TTransfo is a GPT-based personalized dialogue model with multi-task learning proposed by Wolf et al. (2019b). This model obtained state-of-the-art performance on automatic metrics in the Second Conversational Intelligence Challenge.

P^2 BOT is a GPT-based personalized dialogue model with the reinforce algorithm proposed by Liu et al. (2020). This is the latest state-of-the-art model for dialogue generation on persona-chat.

DialoGPT is a pre-trained dialogue model proposed by Zhang et al. (2020). This model is based on GPT-2 using the Reddit comments dataset. We compared ours to the version of model size 345M, which had the best result reported in the paper.

3.3 Implementation

Our implementation was based on the PyTorch (Paszke et al., 2019) and HuggingFace libraries (Wolf et al., 2019a). The response decoder GPT-2 was set to 16 heads, 24 layers, 1024 dimensional hidden state, and with 345M parameters. All input representation refers to the embedding tables of GPT-2, and the embedding size was the same setting as the size of latent variables, which was fixed at 1024. The distinction objective λ was set to 0.15. The Adam algorithm (Kingma and Ba, 2015) was utilized for optimization with a learning rate of 2.6e-5, and a warmup step of 3000. Responses were generated by nucleus filtering (Holtzman et al., 2019) where top-k and top-p were set to 4 and 0.8, respectively. BPE algorithm (Sennrich et al., 2016) was used for word tokenization, the token vocabularies of GPT2, with a size of 50,257, were shared by the prior and posterior networks.

3.4 Evaluation

3.4.1 Automatic Metrics

We followed previous work and employed **Perplexity (PPL)** (Sutskever et al., 2014) and **Distinct** (Li et al., 2015). The PPL measures the negative log-likelihood of the ground-truth sequence output by the model. A lower PPL generally indicates that the learned language model is more human-like. The Distinct is calculated as the number of distinct unigrams and bigrams divided by the total number of generated words. This metric assesses the degrees of word-level diversity for generated responses.

Furthermore, we propose a new metric to estimate the level of the correlation of generated response and the user's persona, which is named **P.Distance (Persona Distance)**. For word embedding trained under the language model, the distance between vectors in the respective space is proportional to the relative co-occurrence of words they represent. Therefore, we employed the pre-trained Google News (300D)[1] word2vec to measure the closeness between the response and corresponding profile in the vector space. We removed stop words for the profile and the generated response, then extracted the keywords of each response-profile pair by word frequency of the training set. For the i-th profile keyword embedding $\boldsymbol{p}_i$, we can make the similarity matrix as follows:

$$\boldsymbol{M}_i = [Sim(\boldsymbol{p}_i, \boldsymbol{r}_1), Sim(\boldsymbol{p}_i, \boldsymbol{r}_2), \ldots, Sim(\boldsymbol{p}_i, \boldsymbol{r}_n)] \quad (6)$$

where $Sim(\cdot, \cdot)$ is a cosine similarity function, and $\boldsymbol{r}_i$ is the embedding of the i-th response keyword. The P.Distance can be calculated as follows:

$$P.Distance = Ave(Max(\boldsymbol{M}_1), Max(\boldsymbol{M}_2), \ldots, Max(\boldsymbol{M}_n)) \quad (7)$$

3.4.2 Human Metrics

We engaged six native speakers[2] to annotate the quality of generated responses based on the following criteria. The scale of these metrics is [0, 1, 2], and for each dialogue, the generated responses by all models were order shuffled in the evaluation.

Coherence measures whether the response is consistent with the context. *Score 0*: The response is not related to the context. *Score 1*: The response mentions something related to the context but is not coherent. *Score 2*: The response is coherent with the context and not generic.

Engagingness assesses how well the response endeavors to continue the dialogue. *Score 0*: The response is generic or poor quality, which makes it difficult to continue the dialogue. *Score 1*: The response is boring, but it is still acceptable to continue the dialogue. *Score 2*: The response is interesting and the dialogue can be developed.

[1] https://code.google.com/archive/p/word2vec/

[2] All the annotators are graduate students recruited from the internet whose are not relevant to this study.

Model	PPL	Distinct-1 / 2	P.Distance	Coherence	Engagingness	P.Relevancy
HRED	21.095	0.078 / 0.225	0.246	0.657	0.703	0.670
CVAE	19.501	0.116 / 0.405	0.258	0.557	0.673	0.523
TTransfo	18.011	0.142 / 0.400	0.329	0.840	0.703	0.877
DialoGPT	**14.966**	0.139 / 0.417	0.359	1.037	0.883	0.900
P^2 BOT	16.620	0.083 / 0.268	0.370	0.953	0.920	0.873
Ours	15.671	**0.167 / 0.538**	**0.401**	**1.177**	**1.203**	**1.207**

Table 1: Evaluation results on the ConvAI2 dialogue corpus, the best score in each metric are in bold. For our model and CVAE, the latent variables were sampled N times to generate N responses, and the final evaluation scores were acquired by average (N = 3 in our experiments); 50 dialogues were randomly sampled from the testing set for human evaluation. The statistical test showed the differences are significant with p-value < 0.05.

P.Relevancy (Persona Relevancy) estimates the degree of a response being relevant to the other party's persona. And the persona of the other party is required to be inferred from the context. *Score 0*: What the response mentions is irrelevant to the other party's persona. *Score 1*: The response involves a question to the other party and is not generic. *Score 2*: What the response mentions is related to the persona of the other party.

3.4.3 Results

Table 1 shows the evaluation results. We can see that, compared to baseline, our approach was superior in all metrics except PPL. Nonetheless, this metric also achieves highly competitive performance. CVAE gains a higher Distinct score than HRED and P^2 BOT, which could be attributed to the variational autoencoder catching discourse-level diversity. Our model surpassed CVAE on Distinct and P.Distance, which suggests that the implicit persona modeling can better reflect the specific user's persona, creating more informative responses. The comparison with DialoGPT can be seen as an ablation study since our model would degenerate into a GPT-2 with removal of the latent variables. It can be observed that ours outperformed DialoGPT overall, which reveals that the proposed latent variables are beneficial for generating more user-related and diversified responses. Ours is slightly inferior on PPL, which is to be expected due to stochasticity for the language model brought by the latent variables.

On the other side, the personalized dialogue models TTransfo and P^2 BOT received an undesirable P.Relevancy. And P^2 BOT had only a slightly improved Engagingness compared to DialoGPT, which indicates that even with a specific personality, responses that lack consideration for the interlocutor may limit the attraction for people to continue the exchange. By contrast, ours obtained meaningful advances in Engagingness and P.Relevancy, which demonstrates that responses relevant to the other party's persona can motivate the user to participate actively in conversation. When it comes to Coherence, both HRED and CVAE attained lower scores compared to all other large-scale transformer-based models. This is not surprising because pre-trained language models have proved to have better language understanding capability than traditional RNNs. The Fleiss' kappa (Fleiss, 1971) score with human judges was around 0.33, which can be regarded as "fair agreement."

4 Discussion

4.1 Analyzing Latent Variables

One assumption is that individual persona features could be classified in latent space. Additionally, previous research (Zhao et al., 2017) has identified that the posterior network can grasp the clustering of high-dimensional discrete samples. Thus, we wanted to check if the perception variable can be learned in the explainable collections. All profiles in the training set were classified into six predefined categories by employing a pre-trained zero-shot classifier (Lewis et al., 2020). The classifier calculates the probabilities of category attribution in the manner of building profiles and categories into premise-hypothesis pairs (Yin et al., 2019). Figure 5 visualizes the posterior perception variables in 2D space using t-SNE (Maaten and Hinton, 2008). We discovered that the latent space learned by Z_p is correlated with the profile categories. Recall that perception variable is devised to refine the user's implicit persona, and this result is in line with our initial conception.

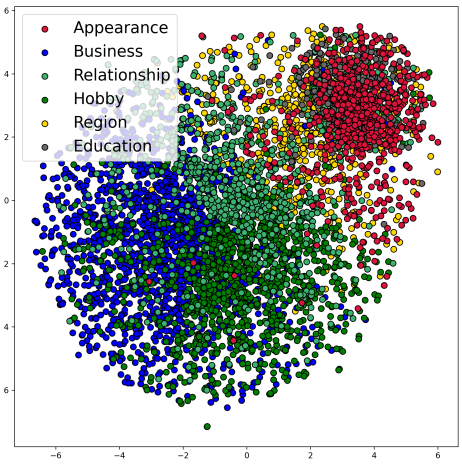

Figure 5: The visualization of perception variable.

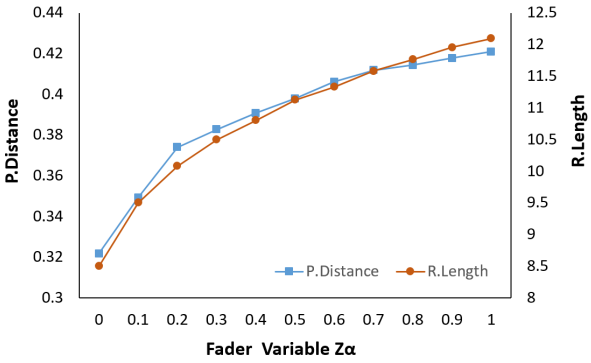

Figure 6: Controllability analysis for fader variable. The increment is set as 0.1.

User Persona:	
I like sports, especially basketball. I am tall.	
I like listening to music. I am a student.	
1st Turn	U: Hi, I like listening to music.
	r_1: Cool, my favorite **musician** is Todd Lee.
	r_2: Who is your favorite **singer**?
	r_3: That's cool do you have pets?
2nd Turn	U: I also enjoy playing basketball.
	r_1: That is fun. What **position** do you play?
	r_2: What **team** do you like the least?
	r_3: Oh, that's interesting. How long have you played basketball?
3rd Turn	U: I play center.
	r_1: Oh, that's awesome. How **tall** are you?
	r_2: Ah, nice. Do you play for a **team** or just for fun?
	r_3: Oh, that is cool. I am starting to get into the **NBA**.
4th Turn	U: I am 6 foot 3. I often play with my roommates.
	r_1: Wow, that's tall. Are you in **school**?
	r_2: I bet you can play **baseball** too.
	r_3: I am 6 feet 1 inches tall. Do you have siblings?

Table 2: The involved personas in responses are in bold.

Then we studied the impact of the fader variable in modeling response generation. Because the fader variable aims to control the representation of implicit persona, we verified its effect on the generation by sliding the value. Specifically, we gradually boosted Z_α from 0 to 1 instead of the prior network $p_\theta(Z_\alpha|C, Z_p)$ during inference. The proximity between the generated response and ground-truth profile was computed by P.Distance. Figure 6 reveals the test result that the proximity had an inverse correlation with increasing value of the fader variable. Meanwhile, the length of generated response (the number of generated tokens in a response) showed an increasing tendency due to the fader variable controlling the amount of persona information the model was attempting to represent.

4.2 Case Study

For a more comprehensive qualitative analysis, we set up a hypothetical persona to interact with our model. Table 2 presents the diverse responses generated from our model in four consecutive turns. The responses were sampled randomly three times per turn, and utterances from the user follow by r_1 of the previous turn. As observed, most of the responses not only covered the user's persona but were also consistent with the context. Responses in each turn are significantly dissimilar, which suggests our model can perform various expressions by the proposed latent variables. That is in line with the "one-to-many" nature mentioned in the introduction. Further, we detected that our model was more likely to raise questions about the other party and seldom generated generic responses. Naturally, putting forth a question could contribute to keeping the conversation going. And such responses also have a high probability of hitting an aspect of the user's persona. That explains why our approach obtained a remarkable score on Engagingness and P.Relevancy in the human evaluation.

4.3 Probing $\mathcal{L}_{Po\text{-}di}$

The efficacy of $\mathcal{L}_{Po\text{-}di}$ in helping to alleviate "posterior collapse" was assessed by a comparative trial. We carried out the language modeling task on Penn Treebank (Marcinkiewicz, 1994) utilizing the VAE constructed by seq2seq architecture based on GRU.

Method	PPL	AU	KL cost
Standard VAE	58.391	0	0.049
+ KLA	53.564	3	2.508
+ CA	51.547	2	3.767
+ AT	50.000	6	5.320
+ BOW	48.637	11	11.165
+ $\mathcal{L}_{Po\text{-}di}$ ($\lambda = 0.12$)	48.467	13	14.727
+ $\mathcal{L}_{Po\text{-}di}$ ($\lambda = 0.15$)	47.674	**14**	16.096
+ $\mathcal{L}_{Po\text{-}di}$ ($\lambda = 0.18$)	**45.925**	**14**	22.047
+ $\mathcal{L}_{Po\text{-}di}$ ($\lambda = 0.21$)	46.883	**14**	25.126

Table 3: Automatic results for different methods.

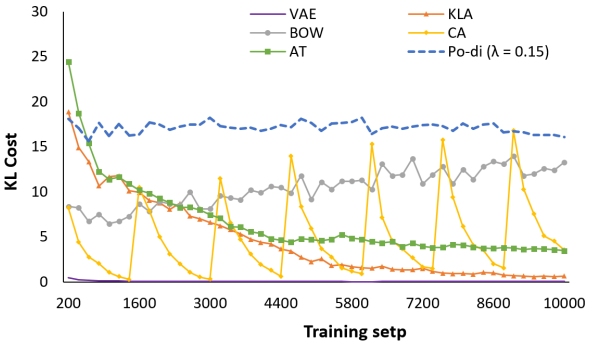

Figure 7: The convergence of KL costs during training.

We set the KL weight of KL annealing (KLA) to increase linearly from 0 to 1 in the first 5000 steps. Table 3 reports PPL, the number of active units (AU) (Burda et al., 2016), and KL cost for six kinds of training techniques on the testing set. We varied the distinction objective λ and report four settings between 0.12 and 0.21. In our experiments, the settings in this range obtained a sounder balance between PPL and KL cost. We can see that $\mathcal{L}_{Po\text{-}di}$ reconstructed the language model with lower perplexity while converging to a small but meaningful KL cost. $\mathcal{L}_{Po\text{-}di}$ retained more active units than others, which indicates a richer latent representation can be acquired by "pulling apart" the KL divergence between the different posteriors. Figure 7 presents the evolution of the KL cost during training. Compared to VAE without any strategies, the model with KLA can prevent the KL cost crashes at the beginning of training, but the effect is diminished by degrees after the KL weight climbs to 1. Although this problem is fixed by cyclic annealing (CA) and aggressive training (AT), they still have slightly poor performance on PPL. VAE with BOW gained comparable performance to ours, whereas $\mathcal{L}_{Po\text{-}di}$ without introducing any supplemental neural network still mitigated "posterior collapse."

5 Related Work

5.1 Variational Autoencoders (VAEs)

The VAEs (Kingma and Welling, 2013; Rezende et al., 2014) were proposed for image generation and applied by Bowman et al. (2016) for natural language generation. Then, the CVAEs (Yan et al., 2016; Sohn et al., 2015) were proposed to enable more controllable generation that conditioned certain attributes. Zhao et al. (2017) adopted the CVAE for the task of multi-turn dialogue modeling, which learns a distribution over dialogue acts to capture discourse-level variations. The above models achieve various generations by drawing latent variables from the learned distribution.

5.2 Personalized Dialogue Models

Recently, there has been much research exploring different approaches to the task of personalized dialogue generation (Yang et al., 2020; Song et al., 2020; Zheng et al., 2020; Wu et al., 2020; Xu et al., 2021). P^2 BOT (Liu et al., 2020) and TTransfo (Wolf et al., 2019b) are recognized state-of-the-art baselines on persona-chat. P^2 BOT proposes a transmitter-receiver and mutual persona perception framework that fuses supervised training and self-play fine-tuning for enhancing the quality of personalized dialogue generation. TTransfo combines transfer learning and the Transformer model, and fine-tuning is performed on the pre-trained model by optimizing the multi-task objective function to improve the fluency of personalized responses.

The aforementioned approaches involve conditioning responses on the additional agent's persona. Instead, the variational method allows us to be flexible in handling the effects of conditions (i.e., context) and is independent of external knowledge. Our method further integrates the details of the user into the inference process.

6 Conclusion and Future Work

This paper presented a new implicit persona detection generator to achieve a user-personalized response. We establish the persona exploration and dialogue generation in a unified framework that supplies a way of leveraging the potential facts in dialogue. Experiments on a large public dataset demonstrated that our approach had superior performance in producing user-specific responses. Humans typically continue exchanges by drilling into

the content of the conversation. From this perspective, the PersonalDialog dataset (Zheng et al., 2019) may be more appropriate for our approach. In the future, we plan to use this dataset to study if the inference of implicit personas can be strengthened. We would also conduct further experiments to examine whether there is an interpretable association between the prior network and the recognition network in terms of what they have learned. Eventually, we plan to perfect $\mathcal{L}_{Po\text{-}di}$ by having weights adaptively regulate the KL divergence.

References

Samuel Bowman, Luke Vilnis, Oriol Vinyals, Andrew Dai, Rafal Jozefowicz, and Samy Bengio. 2016. Generating sentences from a continuous space. In *Proceedings of The 20th SIGNLL Conference on Computational Natural Language Learning*, pages 10–21.

Yuri Burda, Roger B Grosse, and Ruslan Salakhutdinov. 2016. Importance weighted autoencoders. In *International Conference on Learning Representations*.

Joseph L Fleiss. 1971. Measuring nominal scale agreement among many raters. *Psychological bulletin*, 76(5):378.

Hao Fu, Chunyuan Li, Xiaodong Liu, Jianfeng Gao, Asli Celikyilmaz, and Lawrence Carin. 2019. Cyclical annealing schedule: A simple approach to mitigating kl vanishing. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 240–250.

Junxian He, Daniel Spokoyny, Graham Neubig, and Taylor Berg-Kirkpatrick. 2019. Lagging inference networks and posterior collapse in variational autoencoders. In *International Conference on Learning Representations*.

Ari Holtzman, Jan Buys, Li Du, Maxwell Forbes, and Yejin Choi. 2019. The curious case of neural text degeneration. In *International Conference on Learning Representations*.

Diederik P Kingma and Jimmy Ba. 2015. Adam: A method for stochastic optimization. In *Proceedings of the 3rd International Conference on Learning Representations, ICLR*.

Diederik P Kingma and Max Welling. 2013. Auto-encoding variational bayes. *arXiv preprint arXiv:1312.6114*.

Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Veselin Stoyanov, and Luke Zettlemoyer. 2020. Bart: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7871–7880.

Jiwei Li, Michel Galley, Chris Brockett, Jianfeng Gao, and Bill Dolan. 2015. A diversity-promoting objective function for neural conversation models. In *Proceedings of the 2015 Conference of the North American Chapter of the Association for Computational Linguistics*, pages 110–119.

Qian Liu, Yihong Chen, Bei Chen, Jian-Guang Lou, Zixuan Chen, Bin Zhou, and Dongmei Zhang. 2020. You impress me: Dialogue generation via mutual persona perception. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 1417–1427.

Laurens Maaten and Geoffrey Hinton. 2008. Visualizing data using t-sne. *Journal of machine learning research*, 9(11).

Mary Ann Marcinkiewicz. 1994. Building a large annotated corpus of english: The penn treebank. *Using Large Corpora*, page 273.

Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, et al. 2019. Pytorch: An imperative style, high-performance deep learning library. In *Advances in neural information processing systems*, volume 32, pages 8024–8035.

Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, Ilya Sutskever, et al. 2019. Language models are unsupervised multitask learners. *OpenAI blog*, 1(8):9.

Danilo Jimenez Rezende, Shakir Mohamed, and Daan Wierstra. 2014. Stochastic backpropagation and approximate inference in deep generative models. In *International conference on machine learning*, pages 1278–1286. PMLR.

Rico Sennrich, Barry Haddow, and Alexandra Birch. 2016. Neural machine translation of rare words with subword units. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1715–1725.

Iulian V Serban, Alessandro Sordoni, Yoshua Bengio, Aaron Courville, and Joelle Pineau. 2016. Building end-to-end dialogue systems using generative hierarchical neural network models. In *Proceedings of the 30th AAAI Conference on Artificial Intelligence*.

Heung-Yeung Shum, Xiao-dong He, and Di Li. 2018. From ELIZA to XiaoIce: challenges and opportunities with social chatbots. *Frontiers of Information Technology Electronic Engineering*, 19(1):10–26.

Kihyuk Sohn, Honglak Lee, and Xinchen Yan. 2015. Learning structured output representation using deep conditional generative models. In *Advances in neural information processing systems*, volume 28, pages 3483–3491.

Haoyu Song, Wei-Nan Zhang, Jingwen Hu, and Ting Liu. 2020. Generating persona consistent dialogues by exploiting natural language inference. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pages 8878–8885.

Ilya Sutskever, Oriol Vinyals, and Quoc V Le. 2014. Sequence to sequence learning with neural networks. In *Advances in neural information processing systems*, volume 27, pages 3104–3112.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in neural information processing systems*, volume 30, pages 5998–6008.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Rémi Louf, Morgan Funtowicz, et al. 2019a. Huggingface's transformers: State-of-the-art natural language processing. *arXiv e-prints*, pages arXiv–1910.

Thomas Wolf, Victor Sanh, Julien Chaumond, and Clement Delangue. 2019b. Transfertransfo: A transfer learning approach for neural network based conversational agents. *arXiv preprint arXiv:1901.08149*.

Bowen Wu, MengYuan Li, Zongsheng Wang, Yifu Chen, Derek F Wong, Qihang Feng, Junhong Huang, and Baoxun Wang. 2020. Guiding variational response generator to exploit persona. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 53–65.

Fuyong Xu, Guangtao Xu, Yuanying Wang, Ru Wang, Qi Ding, Peiyu Liu, and Zhenfang Zhu. 2021. Diverse dialogue generation by fusing mutual persona-aware and self-transferrer. *Applied Intelligence*, pages 1–14.

Xinchen Yan, Jimei Yang, Kihyuk Sohn, and Honglak Lee. 2016. Attribute2image: Conditional image generation from visual attributes. In *European conference on computer vision*, pages 776–791. Springer.

Min Yang, Weiyi Huang, Wenting Tu, Qiang Qu, Ying Shen, and Kai Lei. 2020. Multitask learning and reinforcement learning for personalized dialog generation: An empirical study. *IEEE Transactions on Neural Networks and Learning Systems*, 32(1):49–62.

Wenpeng Yin, Jamaal Hay, and Dan Roth. 2019. Benchmarking zero-shot text classification: Datasets, evaluation and entailment approach. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 3914–3923.

Saizheng Zhang, Emily Dinan, Jack Urbanek, Arthur Szlam, Douwe Kiela, and Jason Weston. 2018. Personalizing dialogue agents: I have a dog, do you have pets too? In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 2204–2213.

Yizhe Zhang, Siqi Sun, Michel Galley, Yen-Chun Chen, Chris Brockett, Xiang Gao, Jianfeng Gao, Jingjing Liu, and William B Dolan. 2020. Dialogpt: Large-scale generative pre-training for conversational response generation. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics: System Demonstrations*, pages 270–278.

Tiancheng Zhao, Ran Zhao, and Maxine Eskenazi. 2017. Learning discourse-level diversity for neural dialog models using conditional variational autoencoders. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 654–664.

Yinhe Zheng, Guanyi Chen, Minlie Huang, Song Liu, and Xuan Zhu. 2019. Personalized dialogue generation with diversified traits. *arXiv preprint arXiv:1901.09672*.

Yinhe Zheng, Rongsheng Zhang, Minlie Huang, and Xiaoxi Mao. 2020. A pre-training based personalized dialogue generation model with persona-sparse data. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pages 9693–9700.

A Appendix

Derivation of ELBO

The conditional likelihood can be written as follows by introducing the terms of variational distribution and true posterior. We omitted the parameter identification θ and ϕ to save space in formula writing.

$$\log p(R|C)$$

$$= \int_{Z_\alpha} \int_{Z_p} q(Z_\alpha|P,R) q(Z_p|P) \log p(R|C) dZ_p dZ_\alpha$$

$$= \int_{Z_\alpha} \int_{Z_p} q(Z_\alpha|P,R) q(Z_p|P)$$
$$\log \frac{p(R|C)p(Z_p|C,R)q(Z_p|P)p(Z_\alpha|C,R,Z_p)q(Z_\alpha|P,R)}{p(Z_p|C,R)q(Z_p|P)p(Z_\alpha|C,R,Z_p)q(Z_\alpha|P,R)} dZ_p dZ_\alpha$$

$$= \int_{Z_\alpha} \int_{Z_p} q(Z_\alpha|P,R) q(Z_p|P)$$
$$\log \frac{p(R|C)p(Z_p|C,R)p(Z_\alpha|C,R,Z_p)q(Z_\alpha|P,R)}{q(Z_p|P)p(Z_\alpha|C,R,Z_p)q(Z_\alpha|P,R)} dZ_p dZ_\alpha$$
$$+ \int_{Z_\alpha} \int_{Z_p} q(Z_\alpha|P,R) q(Z_p|P) \log \frac{q(Z_p|P)}{p(Z_p|C,R)} dZ_p dZ_\alpha$$

$$= \int_{Z_\alpha} \int_{Z_p} q(Z_\alpha|P,R) q(Z_p|P)$$
$$\log \frac{p(R,Z_p|C)p(Z_\alpha|C,R,Z_p)q(Z_\alpha|P,R)}{q(Z_p|P)p(Z_\alpha|C,R,Z_p)q(Z_\alpha|P,R)} dZ_p dZ_\alpha$$
$$+ \int_{Z_p} q(Z_p|P) \log \frac{q(Z_p|P)}{p(Z_p|C,R)} dZ_p$$

The first term can be factorized into two parts. We assume the true posterior $p(Z_\alpha|C, R, Z_p)$ is independent of the integrals over Z_p. Thus, the formula can be re-written as follows:

$$
\begin{aligned}
&\log p(R|C) \\
&= \int_{Z_\alpha}\int_{Z_p} q(Z_\alpha|P,R)q(Z_p|P)\log\frac{q(Z_\alpha|P,R)}{p(Z_\alpha|C,R,Z_p)}dZ_p dZ_\alpha \\
&+ \int_{Z_p} q(Z_p|P)\log\frac{q(Z_p|P)}{p(Z_p|C,R)}dZ_p \\
&+ \iint q(Z_\alpha|P,R)q(Z_p|P)\log\frac{p(R,Z_p|C)p(Z_\alpha|C,R,Z_p)}{q(Z_p|P)q(Z_\alpha|P,R)}dZ_p dZ_\alpha \\
&\approx \underbrace{\int_{Z_\alpha} q(Z_\alpha|P,R)\log\frac{q(Z_\alpha|P,R)}{p(Z_\alpha|C,R,Z_p)}dZ_\alpha}_{KL(q(Z_\alpha|P,R)||p(Z_\alpha|C,R,Z_p))} \\
&+ \underbrace{\int_{Z_p} q(Z_p|P)\log\frac{q(Z_p|P)}{p(Z_p|C,R)}dZ_p}_{KL(q(Z_p|P)||p(Z_p|C,R))} \\
&+ \underbrace{\iint q(Z_\alpha|P,R)q(Z_p|P)\log\frac{p(R,Z_p|C)p(Z_\alpha|C,R,Z_p)}{q(Z_p|P)q(Z_\alpha|P,R)}dZ_p dZ_\alpha}_{\text{ELBO}}
\end{aligned}
$$

Where the first two terms are KL divergence between the true posterior and variational distribution. Since KL divergence is always greater than or equal to 0, to maximize the likelihood $\log p(R|C)$ can be converted to maximize ELBO, which can be reformulated as follows:

$$
\begin{aligned}
&\log p(R|C) \geq \text{ELBO} = \\
&-\int_{Z_\alpha}\int_{Z_p} q(Z_\alpha|P,R)q(Z_p|P)\log\frac{q(Z_p|P)q(Z_\alpha|P,R)}{p(R,Z_p,Z_\alpha|C)}dZ_p dZ_\alpha \\
&= -\int_{Z_\alpha}\int_{Z_p} q(Z_\alpha|P,R)q(Z_p|P) \\
&\log\frac{q(Z_p|P)q(Z_\alpha|P,R)}{p(Z_p|C)p(Z_\alpha|C,Z_p)p(R|C,Z_p,Z_\alpha)}dZ_p dZ_\alpha \quad (\text{Bayes}'\text{theorem}) \\
&= \int_{Z_\alpha}\int_{Z_p} q(Z_\alpha|P,R)q(Z_p|P)p(R|C,Z_p,Z_\alpha)dZ_p dZ_\alpha \\
&- \int_{Z_\alpha}\int_{Z_p} q(Z_\alpha|P,R)q(Z_p|P)\log\frac{q(Z_p|P)}{p(Z_p|C)}dZ_p dZ_\alpha \\
&- \int_{Z_\alpha}\int_{Z_p} q(Z_\alpha|P,R)q(Z_p|P)\log\frac{q(Z_\alpha|P,R)}{p(Z_\alpha|C,Z_p)}dZ_p dZ_\alpha \\
&\approx \underbrace{\iint q(Z_\alpha|P,R)q(Z_p|P)p(R|C,Z_p,Z_\alpha)dZ_p dZ_\alpha}_{\mathbb{E}_{q(Z_p|P);q(Z_\alpha|P,R)}[\log p(R|C,Z_p,Z_\alpha)]} \\
&- \underbrace{\int q(Z_p|P)\log\frac{q(Z_p|P)}{p(Z_p|C)}dZ_p}_{KL(q(Z_p|P)||p(Z_p|C))} - \underbrace{\int q(Z_\alpha|P,R)\log\frac{q(Z_\alpha|P,R)}{p(Z_\alpha|C,Z_p)}dZ_\alpha}_{KL(q(Z_\alpha|P,R)||p(Z_\alpha|C,Z_p))}
\end{aligned}
$$

We assume that the prior distribution $p(Z_\alpha|C, Z_p)$ is independent of the integrals over Z_p.

Incorporating Casual Analysis into Diversified and Logical Response Generation

Jiayi Liu[1], Wei Wei[2,3]*, Zhixuan Chu[4], Xing Gao[1], Ji Zhang[1], Tan Yan[4], Yulin Kang[4]
[1]Alibaba Group, China
[2]CCIIP Laboratory, Huazhong University of Science and Technology, China
[3]Joint Laboratory of HUST and Pingan Property & Casualty Research (HPL), China
[4]Ant Group, China
{ljy269999,chuzhixuan.czx,gaoxing.gx,zj122146}@alibaba-inc.com
weiw@hust.edu.cn, tanyan.ty@gmail.com, yulin.kyl@antgroup.com

Abstract

Although the Conditional Variational Auto-Encoder (CVAE) model can generate more diversified responses than the traditional Seq2Seq model, the responses often have low relevance with the input words or are illogical with the question. A causal analysis is carried out to study the reasons behind, and a methodology of searching for the mediators and mitigating the confounding bias in dialogues is provided. Specifically, we propose to predict the mediators to preserve relevant information and auto-regressively incorporate the mediators into generating process. Besides, a dynamic topic graph guided conditional variational auto-encoder (TGG-CVAE) model is utilized to complement the semantic space and reduce the confounding bias in responses. Extensive experiments demonstrate that the proposed model is able to generate both relevant and informative responses, and outperforms the state-of-the-art in terms of automatic metrics and human evaluations.

1 Introduction

With recent advances in deep learning and readily available large-scale dialogue data, generation-based methods have become one of the most prevailing methods for building dialogue systems. Based on the Seq2seq framework (Sutskever et al., 2014; Cho et al., 2014), generation-based models learn to map the input post to its corresponding response through an encoding-decoding strategy and are trained in end-to-end manners (Shang et al., 2015; Sordoni et al., 2015; Vinyals and Le, 2015). However, Seq2seq model tends to produce generic and safe responses (Li et al., 2015) such as "So am I" or "I don't know". Researchers conjecture that the cause of this phenomenon is that one certain post can be replied by multiple responses (*i.e.,* one-to-many mapping), and the maximum likelihood estimation (MLE) training would average out these responses and produce a more bland and generic candidate.

Post:	Have you had dinner?
Response1:	Yeah, sure!
Response2:	Yes, I had it at McDonald's.
Response3:	Nope, I'm busy with my work.
Response4:	Yes, I've had it. I tried a nearby restaurant that features Thai food.

Table 1: An illustration of a general question and its multiple valid responses. The *direct responding semantics* (marked in red) are semantically homogeneous because they have to reply the issue directly. The *supplementary semantics* are more diversified because they add more information to explain or supplement the corresponding *direct responding semantics*.

To tackle this problem and model the one-to-many mapping relationships in dialogues, (Zhao et al., 2017) firstly leverages Conditional Variational Auto-Encoder (CVAE) model to map the input post into a semantic distribution, instead of a fixed vector as used in the vanilla Seq2seq model. The decoder then decodes the sampled points from the semantic distribution to generate corresponding responses. This model significantly increases the diversity of responses, but it is hard to train as the valid responses are too few to shape a clear semantic distribution for each post. As a result, the CVAE model is inclined to learn some spurious statistical cues for predicting diversified words, which may have very low relevance with the input post. Other studies focus on re-using the model's components to fit the multiplicity of dialogues, for instance, the multiple mechanisms used in (Zhou et al., 2017) and (Zhou et al., 2018), the multi-head attention used in (Tao et al., 2018; Liu et al., 2022), and reinforced methods(Qiu et al., 2021). The most-related work in this line is the Multi-Mapping and Posterior Mapping Selection (MMPMS) (Chen et al., 2019) model, which directly builds multiple mapping modules to learn diversified semantics and

*Corresponding author.

Proceedings of the 29th International Conference on Computational Linguistics, pages 378–388
October 12–17, 2022.

generate responses. However, these studies haven't considered the intrinsic nature of this one-to-many phenomenon in dialogues.

We always face the trade-off between the accuracy of response and diversity of semantics, and cannot directly generate relevant and diversified responses from the original input post. To solve this dilemma and examine the nature of dialogues, we introduce the causal inference analysis (Pearl, 1995, 2000) into the dialogue generation task. Here, we assume between the input post and outcome response, there exists one mediator. The mediator can easily capture the relevant but simple response from the input post (input post $\rightarrow$ mediator) and also can pass the learned information to the outcome so as to preserve the relevance. In addition, when generating the diversified responses, the sampling steps in prior and posterior distributions of CVAE will act as the confounders between the input and outcome response. Therefore, we establish one causal graph including the mediator, confounder, input post, and segmented responses, i.e, *direct responding semantics* and *supplementary semantics*, to facilitate the information transmission and enrichment, and preserve the relevance and logicality.

Based on the above causal analysis, this work presents a unified end-to-end sentence-level auto-regressive model (SLARM) to predict the mediator and mitigate the confounding bias in generating diverse responses. We concrete the mediator by predicting the direct responding semantics, and leverage this mediator in an auto-regressive manner for response generation. A dialogue topic graph enhanced CVAE model with a larger semantic space is proposed to reduce the confounding bias in CVAE model, and thus make sure the transition is smooth and natural. In conclusion, the contributions of this work are three-fold:

1. It provides an in-depth analysis of the underlying causality involved in the dialogue generation task, and proposed a methodology of searching for the mediators and mitigating the confounding bias in dialogues.

2. It proposes an innovative dialogue generation model based on the established causal graph with mediator and confounder. The model predicts the *direct responding semantics* as mediators and generate the *supplementary semantics* in a unified auto-regressive manner using the proposed TGG-CVAE part to mitigate the confounding bias.

3. It conducts broad experiments on a real-world dialogue dataset, which demonstrates that our proposed approach outperforms the state-of-the-art methods and has the capability of enhancing the diversity of responses without the sacrifice of relevance.

2 Related Works

Diversified Generation models. Some researchers suggest that the maximum-likelihood training objective used in the seq2seq model will average out the targets and result in safe and commonplace responses. Several attempts have been made to tackle this problem by proposing diversity-promoting objective functions, such as Maximum Mutual Information (MMI) (Li et al., 2015), Inverse Token Frequency Loss (ITF) (Nakamura et al., 2018). Although these studies help mitigate the safe response problem, their performance is far from satisfactory. Recently, researchers have discovered that incorporating additional information can lead to more diverse responses. Such methods include predicting keywords to guide the generation process (Mou et al., 2016; Yao et al., 2017), and using latent variables such as (Zhao et al., 2017; Gao et al., 2019a,b; Wei et al., 2019, 2021). Some recent studies focus on the one-to-many relationship between a certain post and its multiple valid responses, which is a common phenomenon in real dialogues. For instance, (Zhou et al., 2017) and (Zhou et al., 2018) model the one-to-many mapping relationships through multiple latent mechanisms and leverage diverse mechanisms to enhance the diversity of generated responses. (Tao et al., 2018) leverages the multi-head attention to focus on different parts of the input post and generate diverse responses. The state-of-the-art model in this line is the Multi-Mapping and Posterior Mapping Selection (MMPMS) (Chen et al., 2019) model, which directly builds multiple mapping modules to learn diversified semantics and generate responses.

Causal Inference. Causal inference (Pearl, 2000; Rubin, 2005) has been an attractive research topic for a long time since it provides an effective way to uncover causal relationships in real-world problems. Nowadays, the combination of the incisive ideas in the causal inference and various deep learning model can help improve existing methodologies in a wide range of fields, such as treatment effect es-

timation with observational data (Li and Fu, 2017; Chu et al., 2020b, 2022b), causality analysis of graph networked data (Chu et al., 2021), continual learning (Hu et al., 2021; Chu et al., 2020a), natural language processing task (Yang et al., 2021; Niu et al., 2021; Abbasnejad et al., 2020), few-shot learning (Yue et al., 2020, 2021), domain adaptation (Bengio et al., 2019), clinical trials (Chu et al., 2022c), finance (Atanasov and Black, 2016), accounting (Gow et al., 2016), marketing campaigns (Chu et al., 2022a) and so on. It is very challenging to choose or define proper confounders and mediators so as to construct one reasonable causal graph for different new tasks. A confounder is related to both cause and effect in a study, and a mediator explains the process by which cause and effect are related. In this work, we aim to incorporate causal inference into the dialogue generation task to help the model balance the relevance and diversity of response semantics.

3 Causal Analysis

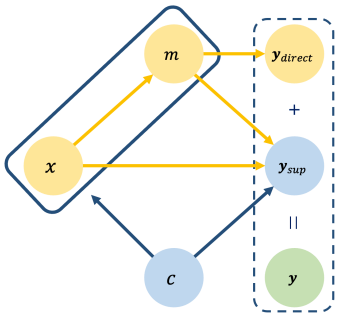

Figure 1: The causal graph of dialogue generation task. The dialogue generation task can be naturally abstracted to one causal graph involving input post x, confounder c, mediators m, direct responding semantics $\boldsymbol{y}_{direct}$, and supplementary semantics $\boldsymbol{y}_{sup}$. The direct responding semantics $\boldsymbol{y}_{direct}$ is the proxy variable of mediator m. The complete response y consists of direct responding semantics $\boldsymbol{y}_{direct}$, and supplementary semantics $\boldsymbol{y}_{sup}$.

In this section, we introduce the causal inference analysis (Pearl, 1995, 2000; Yao et al., 2021) into this task and define the mediator and confounder in the dialogue generation causal graph. A mediator is determined by input post and has causal effects on outcome response, and a confounder has causal effects on both input post and outcome response. Our objective is to leverage the causal relationship involved in the established causal graph to increase the diversity of response semantics, but at the same time, not to reduce the relevance of response to input post.

We assume there exists one mediator between the input post and outcome response. The mediator can easily capture the relevant but simple response from the input post (input post $\rightarrow$ mediator) and also can pass the learned information to outcome so as to preserve the relevance (mediator $\rightarrow$ outcome response). Except for the path via mediator, the input post is also directly predictive of the outcome response (input post $\rightarrow$ outcome response). In addition, we propose to use the CVAE to increase the diversity of responses. However, the sampling steps in prior and posterior distributions of CVAE will act as the confounder between the input combination (input post and mediator) and outcome response (input combination $\leftarrow$ confounder $\rightarrow$ outcome response). Because the input combination and response pairs maybe do not conform to the assumed prior or posterior distributions of CVAE, this confounding bias may make the model learn the spurious statistical cues for the prediction of diversified response, resulting in some linguistically similar but inconsistent or irrelevant expressions in the generated sentences. Therefore, reducing the confounding bias is essential for the dialogue generation task.

Corresponding to the above causal relationship, we split the complete response into two parts, i.e., *direct responding semantics* and *supplementary semantics*, as shown in Figure 1. The *direct responding semantics* represents the semantic part that can be directly leveraged to answer the input question. The direct responding semantics is the proxy variable of the mediator. The *Supplementary semantics* represents the peripheral semantic part that is either an explanation, a supplement, or an extension of the *direct responding semantics*. The *direct responding semantics* is semantically homogeneous because it has to solve the issue directly, and the *supplementary semantics* is more diversified because it adds more information to explain or supplement the *direct responding semantics*, or even change the topics to make conversation continue.

Although the high-quality observations of the mediators can reduce the confounding bias hidden in the causal structure by reducing the possibility of counting on the confounders, it is not enough to attain one relevant and diversified response in the complex dialogue generation task. In addition, unlike the mediator that can be represented by direct responding semantics, it is very challenging to define and construct the exact confounders clearly. Therefore, due to the complex

causal graph and hidden confounders, the front-door and back-door adjustments (Glymour et al., 2016; Pearl and Mackenzie, 2018) for reducing the confounding bias cannot be easily applied. Therefore, instead of the conventional causal intervention based on Pearl's do-calculus (Pearl and Mackenzie, 2018), we propose to exploit the dialogue topic graph to complement the semantic space and assign more relevant information into CVAE, which can enhance the diversity and keep the relevance of input post simultaneously.

4 Proposed Model

Our response generation task is defined as follows. Given an input post $\boldsymbol{x} = \{x_1, x_2, \cdots, x_T\}$, the problem is to generate the corresponding response sequence $\boldsymbol{y} = \{y_1, y_2, \cdots, y_{T''}\}$, where T is the length of the post and T'' is the length of response.

To address this problem, we propose to generate the response sequence in a sentence-level auto-regressive manner. Firstly, we predict the proxy variable of the mediator by maximizing the log-likelihood of the following formula:

$$\boldsymbol{y}^*_{direct} = \arg\max \Pr(\boldsymbol{y}|\boldsymbol{x}). \quad (1)$$

As mentioned above, this process produce general responses, but they are closely related to the input post and may help determine where the conversation should go. Hence, we preserve the causal path (input post $\rightarrow$ mediator) and then we can transmit the learned information to outcome so as to preserve the relevance (mediator $\rightarrow$ outcome response).

Then, an Sentence Level Auto-Regressive generating Model (SLARM) is proposed to produce diverse and informative responses based on the mediator and the dialogue topic graph. We first propose to utilize the predicted mediator in an auto-regressive manner:

$$\boldsymbol{y}^*_{sup} = \arg\max \Pr(\boldsymbol{y}|\boldsymbol{x}, \boldsymbol{y}^*_{direct}), \quad (2)$$

and then build a topic graph enhanced CVAE model to mitigate the confounding bias in traditional CVAE models. The auto-regressive training manner serves like a prompt to naturally inject the mediator into generation process, and the topic graph provides dynamic guidance to prevent the CVAE model from off-the-topic deviation and complement the semantic space.

4.1 Mediator Predictor

As aforementioned, we need to capture the relevant information with the input post and thus we need to predict the mediators in dialogue. Here, we propose to leverage Seq2seq-model with attention mechanism as the mediator predictor to generate *direct responding semantics*. This deterministic model can easily capture this simple semantic responding pattern and produce relevant response for our further processing.

4.2 Auto-Regressive Response Generator

So far, we have utilized the direct responding semantics generator to attain the mediator. Except for the path via mediator, the input post is also directly predictive of the diversified response. Now, based on the combination of input post and *direct responding semantics*, we aim to learn the *supplementary semantics*. The *supplementary semantics* is of great importance to provide useful information for interlocutors, and it can be rendered as an explanation, supplement, or extension of the previous *direct responding semantics*. This semantic part has great diversity and contains many relevant entities. Although the high-quality observations of the mediators can reduce the confounding bias hidden in CVAE, it is not enough to attain one relevant and diversified *supplementary semantics* in the complex dialogue generation task. Following the previous causal analysis, we propose to exploit the dialogue topic graph to complement the semantic space and assign more relevant information into CVAE. Therefore, we design a novel model, *i.e.,* topic graph guided CVAE model (TGG-CVAE), to extend the semantic space in the conversation and sample more diversified and relevant sentences, and leverage the dynamic guidance from the dialogue topic graph to provide smooth and natural transition from the *direct responding semantics* to the *supplementary semantics*. The model structure is depicted in Figure 2.

To generate the supplementary semantics, the proposed TGG-CVAE model takes in the input post and previously generated direct semantics response. We denote the input $\hat{\boldsymbol{x}}$ as:

$$\hat{\boldsymbol{x}} = \{x_1, x_2, ..., x_T, [SEP], y_1, y_2, ..., y_{T'}\}, \quad (3)$$

where the [SEP] token is a special token to separate the two sentences (Devlin et al., 2018). The goal of this model is to generate the *supplementary*

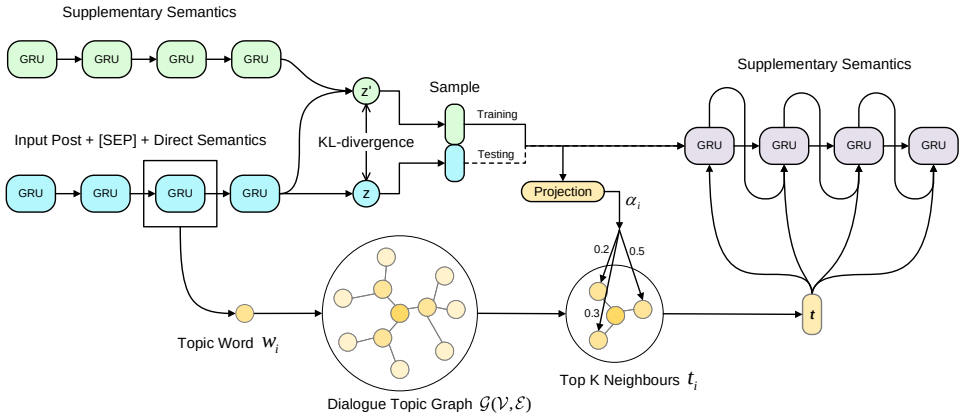

Figure 2: The architecture of our proposed TGG-CVAE model.

semantics:

$$\boldsymbol{y}_{sup} = \{y_{T'+1}, y_{T'+2}, \cdots, y_{T''}\}. \quad (4)$$

This model mainly consists of four components: a prior network, a posterior network, a topic guide network, and a decoding network. The prior network is trained to approximate $p_\theta(z|\hat{\boldsymbol{x}})$ while the posterior network is trained to approximate $q_\psi(z|\hat{\boldsymbol{x}}, \boldsymbol{y}_{sup})$, where θ and ψ are the network parameters and z is the latent variable. Here, z is assumed to follow multivariate Gaussian distribution (Zhao et al., 2017) and then we have:

$$p_\theta(z|\hat{\boldsymbol{x}}) \sim \mathcal{N}(\mu, \sigma^2\mathbf{I}) \quad (5)$$
$$q_\psi(z|\hat{\boldsymbol{x}}, \boldsymbol{y}_{sup}) \sim \mathcal{N}(\mu', \sigma'^2\mathbf{I}) \quad (6)$$

Typically, the prior network and the posterior network are RNN-based encoders that transform the input $\hat{\boldsymbol{x}}$ and $\boldsymbol{y}_{sup}$ into hidden states:

$$\boldsymbol{h}_x^i = f(\boldsymbol{h}_x^{i-1}, \hat{\boldsymbol{x}}_i) \quad (7)$$

$$\boldsymbol{h}_y^j = f(\boldsymbol{h}_y^{j-1}, \boldsymbol{y}_j), \quad (8)$$

where $i \in [1, T+T'+1]$ and $j \in [T'+1, T'']$. The last hidden states from the prior/posterior network are denoted as $\boldsymbol{h}_x$ and $\boldsymbol{h}_y$ respectively. The latent variable is estimated by parameterizing its mean and log variance:

$$\begin{bmatrix} \mu \\ \log(\sigma^2) \end{bmatrix} = \boldsymbol{W}_p(\boldsymbol{h}_x) + b_p \quad (9)$$

$$\begin{bmatrix} \mu' \\ \log(\sigma'^2) \end{bmatrix} = \boldsymbol{W}_r \begin{bmatrix} \boldsymbol{h}_x \\ \boldsymbol{h}_y \end{bmatrix} + b_r, \quad (10)$$

where the $\boldsymbol{W}_p$, $\boldsymbol{W}_r$ and b_p, b_r are the weights and biases for the prior network and posterior network respectively. Reparametrization trick (Kingma and Welling, 2014) is used to keep the gradient propagate successfully in networks via a differentiable transformation of an auxiliary noise variable ϵ:

$$z = \mu + \sigma\epsilon \quad (11)$$
$$z' = \mu' + \sigma'\epsilon \quad (12)$$

Then we can sample the latent variables z or z' from the prior network or the posterior network. However, in the testing stage, as the ground-truth response is not available, the posterior latent variable $\boldsymbol{z}'$ cannot be properly estimated. Therefore, we need to make sure that the prior network can fully acquire useful information from the posterior network by homogenizing $\boldsymbol{z}$ and $\boldsymbol{z}'$. Here, KL-divergence loss is leveraged in our model to minimize the discrepancy between the two latent distributions:

$$\begin{aligned} \mathcal{L}_{KL} &= KL(q_\psi(\boldsymbol{z}|\hat{\boldsymbol{x}}, \boldsymbol{y}_{sup})||p_\theta(\boldsymbol{z}|\hat{\boldsymbol{x}})) \\ &= \int q_\psi(\boldsymbol{z}|\hat{\boldsymbol{x}}, \boldsymbol{y}_{sup}) log \frac{q_\psi(\boldsymbol{z}|\hat{\boldsymbol{x}}, \boldsymbol{y}_{sup})}{p_\theta(\boldsymbol{z}|\hat{\boldsymbol{x}})} dz, \end{aligned} \quad (13)$$

from which we can derive the final formula for calculating KL-divergence loss:

$$\mathcal{L}_{KL} = log\frac{\sigma}{\sigma'} + \frac{\sigma'^2 + (\mu - \mu')^2}{2\sigma^2} - \frac{1}{2} \quad (14)$$

For the vanilla CVAE model, $\boldsymbol{z}$ or $\boldsymbol{z}'$ is directly fed as the input of the decoder for decoding from the semantic space. However, as aforementioned, the latent semantic space is too large to train well and the sampling steps in prior and posterior distributions of vanilla CVAE will act as the confounder

between the input combination (input post and *direct responding semantics*) and *supplementary semantics*. Because the input post and response pairs in the real data maybe do not conform to the assumed prior or posterior distributions of CVAE, this confounding bias may make the model learn the spurious statistical cues for prediction of diversified response, resulting in some linguistically similar but inconsistent or irrelevant expressions in the generated sentences. Therefore, reducing the confounding bias is essential for the *supplementary semantics* generation. We exploit the dialogue topic graph to complement the semantic space and assign more accurate and relevant relationship into CVAE so as to mitigate the confounding bias. Details of this strategy are as follows:

Firstly, the topic words $w_1, w_2, ..., w_m$ are extracted from $\hat{\boldsymbol{x}}$ using the TF-IDF method, and then they are placed into the dialogue topic graph $\mathcal{G}(\mathcal{V}, \mathcal{E})$ to find their nearest n neighbours $w_{11}, w_{12}, ..., w_{mn}$ according to the weight, where w_{ij} is the j-th neighbours of the topic word w_i. We then choose the neighbour with the highest probabilities:

$$t_1, ..., t_K = \underset{i\in[1,m], j\in[1,n]}{\text{argmax}_K} (\Pr(w_{ij}|w_i)), \quad (15)$$

to select top K topic words, namely, $t_1, ..., t_K$.

Secondly, since these topic words contribute differently to the generation of a response, we leverage the sampled latent variables to formulate a dynamic prior/posterior selection of the topic words. The sampled latent variables $\boldsymbol{z}$ (testing) or $\boldsymbol{z}'$ (training) are passed through a projection layer to produce a distribution over the K topic words, namely $\boldsymbol{\alpha} = \alpha_1, \alpha_2, ..., \alpha_K$. The final representation of the topic information is formulated as a weighted summation of the topic embeddings:

$$\boldsymbol{t} = \alpha_i \cdot \boldsymbol{t_i}, \quad i = 1, 2, 3, ..., k \quad (16)$$

where $\boldsymbol{t_i}$ is the embedding of the word t_i.

Thirdly, the topic information $\boldsymbol{t}$ and the sampled latent variable $\boldsymbol{z}$ or $\boldsymbol{z}'$ are fed into the decoder for generating the supplementary semantics:

$$\Pr(y_t|y_{1:t-1}, \boldsymbol{x}, \boldsymbol{y}_{direct}) = g(\boldsymbol{y}_{t-1}, \boldsymbol{s_t}, \boldsymbol{z}, \boldsymbol{t}), \quad (17)$$

from which we can find that each supplementary semantic word is conditioned on both the topic information and the sampled latent variable, and thus the sentences can be related to the previous words and have more diversity. Following (Sohn et al., 2015), we train the proposed model by maximizing the variational lower bound of the conditional log likelihood:

$$\begin{aligned}\mathcal{L}_{ELBO} = &- KL(q_\psi(\boldsymbol{z}|\hat{\boldsymbol{x}}, \boldsymbol{y}_{sup})||p_\theta(\boldsymbol{z}|\hat{\boldsymbol{x}})) \\ &+ \mathrm{E}_{q_\psi(\boldsymbol{z}|\hat{\boldsymbol{x}}, \boldsymbol{y}_{sup})}[\log p_\theta(\boldsymbol{y}_{sup}|\boldsymbol{z}, \hat{\boldsymbol{x}}, \boldsymbol{t})],\end{aligned} \quad (18)$$

where the $KL(.,.)$ denotes the KL-divergence of two distributions. Since the latent semantic distribution is easy to collapse (*a.k.a.,* KL-collapse problem), we add a bag-of-words loss $\mathcal{L}_{BOW}$ and use KL-annealing strategy to deal with this problem (Zhao et al., 2017). The final loss function of this proposed model is formulated as:

$$\mathcal{L} = \mathcal{L}_{ELBO} + \mathcal{L}_{BOW} + \mathcal{L}_{direct} \quad (19)$$

Note that, we also consider the possible circumstance where the responses do not contain any *supplementary semantics* by leveraging the [EOS] token as the placeholder. If the TGG-CVAE model predicts [EOS] token at the first step, this indicates that the *direct responding semantics* is already complete and it does not need any *supplementary semantics*.

5 Experimental Results

5.1 Dataset

We conduct experiments on a large-scale real-world dialogue dataset, *i.e.,* Short-Text Conversation (STC) dataset (Shang et al., 2015). This dataset is publicly available and is cleaned by the data publishers. It consists of 4,433,853 post-comment pairs collected from Chinese Weibo, a social media platform where people can chat online.

5.2 Evaluation Metrics

Automatic Evaluation. We adopted two widely-used metrics, *BLEU-n* (Papineni et al., 2002) and *Distinct-n* (Li et al., 2015), to automatically evaluate the dialogue generation models. *BLEU-n* score is a referenced evaluation metric to measure word overlap between the generated response and the reference. Note that in our experiment we apply smoothing function 7 (Chen and Cherry, 2014) to avoid the problem when no n-gram overlaps are found. *Distinct-n* score (Li et al., 2015) is used to determine word-level diversity of the generated response. It is measured by calculating the percentage of distinct n-grams in the generated responses.

Models	BLEU-1	BLEU-2	BLEU-3	BLEU-4	Distinct-1	Distinct-2
Seq2seq (Sutskever et al., 2014)	0.2392	0.1937	0.1646	0.1304	0.0549	0.1859
CVAE (Zhao et al., 2017)	0.2223	0.1808	0.1541	0.1222	0.0936	0.4208
MMPMS (Chen et al., 2019)	0.2246	0.1868	0.1612	0.1289	**0.0972**	**0.4214**
DCVAE (Gao et al., 2019b)	0.2124	0.1700	0.1436	0.1134	0.0405	0.1681
SLARM (ours)	**0.2657**	**0.2169**	**0.1850**	**0.1469**	0.0879	0.3685
SLARM w/o TGG (ours)	0.2569	0.2099	0.1792	0.1423	0.0967	0.4088
SLARM w/o CVAE (ours)	0.2544	0.2069	0.1763	0.1398	0.0881	0.2195

Table 2: Automatic evaluation results on STC dataset. The best results are in boldface and the second best results are underlined.

Models	Relevance	Informativeness	Fluency	Average
Seq2seq (Sutskever et al., 2014)	1.52	1.63	**2.68**	1.94
CVAE (Zhao et al., 2017)	1.45	1.73	2.49	1.89
MMPMS (Chen et al., 2019)	1.54	**2.02**	2.00	1.85
DCVAE (Gao et al., 2019b)	**1.96**	1.53	2.48	1.99
SLARM (ours)	1.57	1.82	2.67	**2.02**

Table 3: Human evaluation results on STC dataset. The best results are in boldface and the second best results are underlined.

Human Evaluation. We randomly sampled 100 posts from the test set and let the models generate corresponding responses. Three annotators were invited to rate the post-response pairs from three aspects: relevance (whether the response is relevant to the input post), informativeness (whether the response is informative) and fluency (whether the response has no grammar mistakes). A three-point scale (0,1,2) is used in the evaluation for the above aspects. When contradiction occurs between the first two annotators, the third annotator will resolve the disagreement. Fleiss' kappa (Fleiss and Cohen, 1973) is calculated to measure the inter-rater agreement between the first two annotators.

5.3 Baseline Models

Seq2seq (Bahdanau et al., 2014): it is a canonical seq2seq model with the attention mechanism. **CVAE** (Zhao et al., 2017): it is a conditional variational auto-encoder model. During testing, we randomly sample latent variables from the prior network and generate corresponding responses. **MMPMS** (Chen et al., 2019): it is a multi-mapping and posterior mapping selection model. We use their original implementation and hyper-parameter settings. **DCVAE** (Gao et al., 2019b): it is a discrete CVAE model. We use their original implementation and adopt the two-stage sampling strategy during testing.

5.4 Implementation Details

For our approach, we use 2-layers GRU units for encoders in the prior network/posterior network and the hidden size is set to 256. The embedding size and vocabulary size are set to 200 and 40,000 respectively. Word embeddings are randomly initialized and OOV (out-of-vocabulary) words are replaced with a special token UNK. Adam optimizer (Kingma and Ba, 2014) is used for optimization and the training batch size is 128. The initial learning rate is set to 0.5 and a learning rate decay operation is employed when the validation loss stops decreasing for three consecutive epochs. The decay rate is 0.99. The top 5 neighbors of the topic words in the dialogue graph are chosen and fed into the decoder.

5.5 Results

Automatic evaluation results are shown in Table 2. Notably, our SLARM model outperforms all of the baselines in terms of BLUE metric (with p-value < 0.05) and its performance is 11.2% ahead of the second best model. This verifies our assumption that splitting the to-be-generated responses into different semantic parts and separately generating them with suitable methods will enhance the overall performance. As for the Distinct metric, the performance of our model is moderate compared to the CVAE model and MMPMS model. This is because our main objective is not only boosting the diversity of responses but also promoting relevance

between posts and generated responses.

To further analyze the results, we conduct ablation studies by removing the Topic Graph Guided module (*i.e.,* SLARM w/o TGG) or replacing the CVAE module with traditional GRUs (*i.e.,* SLARM w/o CVAE). After removing TGG, the Distinct performance increases and the BLEU performance decreases. This indicates that our dynamic topic graph guiding strategy is effective in providing relevant information from posts and thus can increase BLEU scores. However, this strategy gets slightly lower Distinct scores because the restrained topics would reduce possibilities in choosing more diversified words. When the CVAE module is removed, the Distinct-2 score drops by a large margin, indicating the CVAE module is effective in extending the semantic space and sampling diversified phrases. The BLUE scores also decrease because the posterior network is essential in providing additional information to dynamically weigh the contribution of topic words. Hence, each component of our model complements each other, and thus the model has the self-adaptive capability to reach a balance between diversity and relevance.

Human evaluation results are shown in Table 3. The DCVAE model surpasses other models in relevance metric. This is owing to the fact that DCVAE model tends to re-use the words in the posts to generate a response, which makes the annotators give high relevance scores. The discrete latent variables from the prior and posterior network are pretrained to predict keywords in the post, and thus sampling from these variables tends to produce the same words in the post. However, the latent variables constrain the generation process, which leads to low informative scores. The MMPMS model performs the best from the informativeness aspect. This is because the auxiliary loss (*i.e.,* matching loss) is effective in encouraging the selection module to choose different and diverse mapping modules. However, some mapping modules are not well-trained and they generate ungrammatical sentences. Therefore their fluency score is rather low. The Seq2seq model gets the highest fluency score, as it often generates common and simple sentences.

Our proposed SLARM model outperforms all the baseline models in terms of the average score. For every single aspect, the SLARM model consistently obtains the second best scores. The second best relevance score indicates that first generating the *direct responding semantics* will assure the relevance with the post because it directly answers the question. The second best informative score shows that the proposed SLARM model can enhance the diversity and generate informative sentences. Our fluency score is also the second best and is close to the Seq2seq model's, which verifies that our methods can alleviate the grammatical problems when concatenating two semantic parts.

Note that the Fleiss' kappa for relevance, informativeness, and fluency are 0.4153, 0.4188, and 0.4378, respectively, indicating "moderate agreement" among the annotators.

5.6 Case Study

We present sampled 4 cased in our Appendix. As is shown in the figure, the Seq2seq model tends to generate safe and generic responses, such as case 1, 2, and 3. The response pattern generated by Seq2seq models often starts with "I also like..." or "Haha...", which makes the responses dull and boring. However, in cases 1 and 3, although these responses are generic, they are semantically appropriate and relevant according to the post. Therefore, this model is suitable for searching for the mediators in dialogue generation, which is actually observed in our model by *direct responding semantics*. The CVAE model is better at choosing diversified words such as "punchline" in case 3 and "autumn" in case 4. However, the confounding bias makes this model learn some spurious statistical cues for predicting diversified words, and thus these words are not logical with regard to the input question. The MMPMS model can produce informative sentences, such as in cases 1, 3, and 4. In case 3, the MMPMS model produces a response that is not only informative but also relevant to the input post, but responses in case 1 are irrelevant. Besides, another major problem is that some of the mapping modules are not well-trained and thus in case 2 we can see the generated sentence is ungrammatical and irrelevant. The DCVAE model tends to copy the input post, such as in case 4. This is the reason why the relevant score for DCVAE model is higher than other models. In some circumstances, DCVAE produces the same results as Seq2seq, such as in case 1 and 2.

We can conclude that the SLARM model performs the best and reaches a balance between relevance and diversity. The *direct responding semantics* (marked in red) in case 1, 2, 3, and 4 are very relevant to the input post, and the *supplementary se-*

Case	Posts	Seq2seq	CVAE	MMPMS	DCVAE	SLARM
1	我只喜欢雨天游泳。痛快。I only like to swim on rainy days. Nice!	我也喜欢这种感觉。I also like this feeling.	不错哦，可以注意哦。Very good, you should notice that.	看小图我以为我是沙发 I thought that was a couch according to the small picture.	我也喜欢这样的感觉 I also like this feeling.	游泳训练不是很好，注意安全 It is not very good to do such swimming exercise. Be careful.
2	很多人是复杂的动物，摸不透，搞不清 Many people are very complicated, and you can't figure them out.	哈哈，我也喜欢，我也喜欢 Haha, I like that too, I like that too.	这就是我们的生活了 This is our life.	他花生的花生是花生的花生和花生的花生 His peanuts peanuts peanuts peanuts.	我也是这么想的 I think so.	我承认，我是这样的。I should admit, that I'm of such kind.
3	《学习之神》,男主角不错，93 年的 The leading man in *The God of Learning* is very nice. He was born in 1993.	我也是这么认为的。I think so.	这是你的笑点吗? Is that your punchline?	哈哈，钟汉良的发色也很帅啊。Haha, Hanliang Zhong's hair style also looks handsome.	看一次笑一次…… Every time I see that I can't help laughing.	这个是我的最爱！我就被他打到了 He is my favorite! I'm only struck by him.
4	漓江的春天拍得真美！This photo of the Li River in spring is so beautiful!	春天在哪里？春天在哪里？ Where is spring? Where is spring?	我喜欢！我的秋天！I like that! This is my autumns!	美得让人不敢恭维！It is so beautiful that I dare not to compliment!	美！美！美！美！ Beautiful! Beautiful! Beautiful! Beautiful!	好美啊！求推荐！It is so beautiful! Please recommend me more!

Figure 3: Case study of the sampled 4 cases. For the SLARM model, words in red are generated by the *direct responding semantics* generator, while the rest are generated by *supplementary semantics* generator.

mantics provide more and diversified information to complete the response. In case 1, the *supplementary semantics* is generated to provide further instruction of being careful when swimming on rainy days. In case 3, the *supplementary semantics* re-emphasizes that the interlocutor is fond of the actor. Additionally, we can observe from the cases that with the dialogue topic graph guiding strategy, the transition from *direct responding semantics* to *supplementary semantics* is smooth and natural. Therefore, these cases fully demonstrate the model's capacity for generating the relevant and diversified responses via searching for the direct responding semantic parts as mediators in dialogues and then utilizing our proposed SLARM model to mitigate the confounding bias and thus enhance the diversity without the loss of relevance.

6 Conclusion

In this paper, we incorporate the causal analysis into the dialogue generation task by searching for the mediators and mitigating the confounding bias in dialogues. We thus propose a sentence level auto-regressive response generation model to first generate mediators to preserve relevance with the input post, and then generate the diversified semantics based on our proposed (SLARM) model. Extensive experimental results demonstrate the effectiveness of our approach. For future work, we are exploring more complicated and self-adaptive methods for locating mediators, and we are trying to leverage de-confounding methods to deal with the CVAE problem.

7 Acknowledgments

This work was supported in part by the National Natural Science Foundation of China under Grant No.61602197, Grant No.L1924068, Grant No.61772076, Grant No.62276110, in part by CCF-AFSG Research Fund under Grant No.RF20210005, and in part by the fund of Joint Laboratory of HUST and Pingan Property & Casualty Research (HPL).

References

Ehsan Abbasnejad, Damien Teney, Amin Parvaneh, Javen Shi, and Anton van den Hengel. 2020. Counterfactual vision and language learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 10044–10054.

Vladimir A Atanasov and Bernard S Black. 2016. Shock-based causal inference in corporate finance and accounting research. *Critical Finance Review*, 5:207–304.

Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. 2014. Neural machine translation by jointly learning to align and translate. *arXiv preprint arXiv:1409.0473*.

Yoshua Bengio, Tristan Deleu, Nasim Rahaman, Rosemary Ke, Sébastien Lachapelle, Olexa Bilaniuk, Anirudh Goyal, and Christopher Pal. 2019. A meta-transfer objective for learning to disentangle causal mechanisms. *arXiv preprint arXiv:1901.10912*.

Boxing Chen and Colin Cherry. 2014. A systematic comparison of smoothing techniques for sentence-level bleu. In *Proceedings of the Ninth Workshop on Statistical Machine Translation, WMT@ACL 2014*, pages 362–367.

Chaotao Chen, Jinhua Peng, Fan Wang, Jun Xu, and Hua Wu. 2019. Generating multiple diverse responses with multi-mapping and posterior mapping selection. In *Proceedings of the Twenty-Eighth International Joint Conference on Artificial Intelligence, IJCAI 2019*, pages 4918–4924.

Kyunghyun Cho, Bart Van Merriënboer, Caglar Gulcehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. 2014. Learning phrase representations using rnn encoder-decoder for statistical machine translation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing, EMNLP 2014*, pages 1724–1734.

Zhixuan Chu, Hui Ding, Guang Zeng, Yuchen Huang, Tan Yan, Yulin Kang, and Sheng Li. 2022a. Hierarchical capsule prediction network for marketing campaigns effect. *arXiv preprint arXiv:2208.10113*.

Zhixuan Chu, Stephen Rathbun, and Sheng Li. 2020a. Continual lifelong causal effect inference with real world evidence.

Zhixuan Chu, Stephen L Rathbun, and Sheng Li. 2020b. Matching in selective and balanced representation space for treatment effects estimation. In *Proceedings of the 29th ACM International Conference on Information & Knowledge Management*, pages 205–214.

Zhixuan Chu, Stephen L Rathbun, and Sheng Li. 2021. Graph infomax adversarial learning for treatment effect estimation with networked observational data. *arXiv preprint arXiv:2106.02881*.

Zhixuan Chu, Stephen L Rathbun, and Sheng Li. 2022b. Learning infomax and domain-independent representations for causal effect inference with real-world data. In *Proceedings of the 2022 SIAM International Conference on Data Mining (SDM)*, pages 433–441. SIAM.

Zhixuan Chu, Stephen L Rathbun, and Sheng Li. 2022c. Multi-task adversarial learning for treatment effect estimation in basket trials. In *Conference on Health, Inference, and Learning*, pages 79–91. PMLR.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*.

Joseph L Fleiss and Jacob Cohen. 1973. The equivalence of weighted kappa and the intraclass correlation coefficient as measures of reliability. *Educational and psychological measurement*.

Jun Gao, Wei Bi, Xiaojiang Liu, Junhui Li, and Shuming Shi. 2019a. Generating multiple diverse responses for short-text conversation. In *Proceedings of the Thirty-Third AAAI Conference on Artificial Intelligence, AAAI 2019*, pages 6383–6390.

Jun Gao, Wei Bi, Xiaojiang Liu, Junhui Li, Guodong Zhou, and Shuming Shi. 2019b. A discrete cvae for response generation on short-text conversation. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing, EMNLP 2019*, pages 1898–1908.

Madelyn Glymour, Judea Pearl, and Nicholas P Jewell. 2016. *Causal inference in statistics: A primer*. John Wiley & Sons.

Ian D Gow, David F Larcker, and Peter C Reiss. 2016. Causal inference in accounting research. *Journal of Accounting Research*, 54(2):477–523.

Xinting Hu, Kaihua Tang, Chunyan Miao, Xian-Sheng Hua, and Hanwang Zhang. 2021. Distilling causal effect of data in class-incremental learning.

Diederik P Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*.

Diederik P Kingma and Max Welling. 2014. Auto-encoding variational bayes. In *2nd International Conference on Learning Representations, ICLR 2014*.

Jiwei Li, Michel Galley, Chris Brockett, Jianfeng Gao, and Bill Dolan. 2015. A diversity-promoting objective function for neural conversation models. In *The 2016 Conference of the North American Chapter of the Association for Computational Linguistics, NAACL HLT 2016*, pages 110–119.

Sheng Li and Yun Fu. 2017. Matching on balanced nonlinear representations for treatment effects estimation. In *NIPS*.

Yuhang Liu, Wei Wei, Daowan Peng, Xianling Mao, Zhiyong He, and Pan Zhou. 2022. Depth-aware and semantic guided relational attention network for visual question answering. *IEEE Transactions on Multimedia*.

Lili Mou, Yiping Song, Rui Yan, Ge Li, Lu Zhang, and Zhi Jin. 2016. Sequence to backward and forward sequences: A content-introducing approach to generative short-text conversation. In *26th International Conference on Computational Linguistics, COLING 2016*, pages 3349–3358.

Ryo Nakamura, Katsuhito Sudoh, Koichiro Yoshino, and Satoshi Nakamura. 2018. Another diversity-promoting objective function for neural dialogue generation. *arXiv*.

Yulei Niu, Kaihua Tang, Hanwang Zhang, Zhiwu Lu, Xian-Sheng Hua, and Ji-Rong Wen. 2021. Counterfactual vqa: A cause-effect look at language bias. In

Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pages 12700–12710.

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. Bleu: a method for automatic evaluation of machine translation. In *Proceedings of the 40th annual meeting on association for computational linguistics*, pages 311–318. Association for Computational Linguistics.

Judea Pearl. 1995. Causal diagrams for empirical research. *Biometrika*, 82(4):669–688.

Judea Pearl. 2000. *Causality: models, reasoning and inference*, volume 29. Springer.

Judea Pearl and Dana Mackenzie. 2018. *The book of why: the new science of cause and effect*. Basic books.

Minghui Qiu, Xinjing Huang, Cen-Chieh Chen, Feng Ji, Chen Qu, Wei Wei, Jun Huang, and Yin Zhang. 2021. Reinforced history backtracking for conversational question answering. In *AAAI*.

Donald B Rubin. 2005. Causal inference using potential outcomes: Design, modeling, decisions. *Journal of the American Statistical Association*, 100(469):322–331.

Lifeng Shang, Zhengdong Lu, and Hang Li. 2015. Neural responding machine for short-text conversation. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics, ACL 2015*, pages 1577–1586.

Kihyuk Sohn, Honglak Lee, and Xinchen Yan. 2015. Learning structured output representation using deep conditional generative models. In *Advances in Neural Information Processing Systems 28, NIPS 2015*, pages 3483–3491.

Alessandro Sordoni, Michel Galley, Michael Auli, Chris Brockett, Yangfeng Ji, Margaret Mitchell, Jian-Yun Nie, Jianfeng Gao, and Bill Dolan. 2015. A neural network approach to context-sensitive generation of conversational responses. In *The 2015 Conference of the North American Chapter of the Association for Computational Linguistics, NAACL HLT 2015*, pages 196–205.

Ilya Sutskever, Oriol Vinyals, and Quoc V Le. 2014. Sequence to sequence learning with neural networks. In *Advances in Neural Information Processing Systems 27, NIPS 2014*, pages 3104–3112.

Chongyang Tao, Shen Gao, Mingyue Shang, Wei Wu, Dongyan Zhao, and Rui Yan. 2018. Get the point of my utterance! learning towards effective responses with multi-head attention mechanism. In *Proceedings of the Twenty-Seventh International Joint Conference on Artificial Intelligence, IJCAI 2018*, pages 4418–4424.

Oriol Vinyals and Quoc Le. 2015. A neural conversational model. In *ICML Workshop*.

Wei Wei, Jiayi Liu, Xian-Ling Mao, Guibing Guo, Feida Zhu, Pan Zhou, and Yuchong Hu. 2019. Emotion-aware chat machine: Automatic emotional response generation for human-like emotional interaction. *Proceedings of the 28th ACM International Conference on Information and Knowledge Management*.

Wei Wei, Jiayi Liu, Xian-Ling Mao, Guibing Guo, Feida Zhu, Pan Zhou, Yuchong Hu, and Shanshan Feng. 2021. Target-guided emotion-aware chat machine. *ACM Transactions on Information Systems (TOIS)*.

Xu Yang, Hanwang Zhang, Guojun Qi, and Jianfei Cai. 2021. Causal attention for vision-language tasks. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 9847–9857.

Lili Yao, Yaoyuan Zhang, Yansong Feng, Dongyan Zhao, and Rui Yan. 2017. Towards implicit content-introducing for generative short-text conversation systems. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing, EMNLP 2017*, pages 2190–2199.

Liuyi Yao, Zhixuan Chu, Sheng Li, Yaliang Li, Jing Gao, and Aidong Zhang. 2021. A survey on causal inference. *ACM Transactions on Knowledge Discovery from Data (TKDD)*, 15(5):1–46.

Zhongqi Yue, Tan Wang, Qianru Sun, Xian-Sheng Hua, and Hanwang Zhang. 2021. Counterfactual zero-shot and open-set visual recognition. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 15404–15414.

Zhongqi Yue, Hanwang Zhang, Qianru Sun, and Xian-Sheng Hua. 2020. Interventional few-shot learning. *arXiv preprint arXiv:2009.13000*.

Tiancheng Zhao, Ran Zhao, and Maxine Eskenazi. 2017. Learning discourse-level diversity for neural dialog models using conditional variational autoencoders. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics, ACL 2017*, pages 654–664.

Ganbin Zhou, Ping Luo, Rongyu Cao, Fen Lin, Bo Chen, and Qing He. 2017. Mechanism-aware neural machine for dialogue response generation. In *Proceedings of the Thirty-First AAAI Conference on Artificial Intelligence, AAAI 2017*, pages 3400–3407.

Ganbin Zhou, Ping Luo, Yijun Xiao, Fen Lin, Bo Chen, and Qing He. 2018. Elastic responding machine for dialog generation with dynamically mechanism selecting. In *Proceedings of the Thirty-Second AAAI Conference on Artificial Intelligence, AAAI 2018*, pages 5730–5737.

Reciprocal Learning of Knowledge Retriever and Response Ranker for Knowledge-Grounded Conversations

Jiazhan Feng[1,4], **Chongyang Tao**[1], **Zhen Li**[1,4], **Chang Liu**[1,2],
Tao Shen[3] and **Dongyan Zhao**[1,2,5*]
[1]Wangxuan Institute of Computer Technology, Peking University
[2]Center for Data Science, Peking University [3]University of Technology Sydney
[4]School of Artificial Intelligence, Peking University
[5]State Key Laboratory of Media Convergence Production Technology and Systems
{fengjiazhan,chongyangtao,lizhen63,liuchang97,zhaody}@pku.edu.cn
tao.shen@uts.edu.au

Abstract

Grounding dialogue agents with knowledge documents has sparked increased attention in both academia and industry. Recently, a growing body of work is trying to build retrieval-based knowledge-grounded dialogue systems. While promising, these approaches require collecting pairs of dialogue context and the corresponding ground-truth knowledge sentences that contain the information regarding the dialogue context. Unfortunately, hand-labeling data to that end is time-consuming, and many datasets and applications lack such knowledge annotations. In this paper, we propose a reciprocal learning approach to jointly optimize a knowledge retriever and a response ranker for knowledge-grounded response retrieval without ground-truth knowledge labels. Specifically, the knowledge retriever uses the feedback from the response ranker as pseudo supervised signals of knowledge retrieval for updating its parameters, while the response ranker also receives the top-ranked knowledge sentences from knowledge retriever for optimization. Evaluation results on two public benchmarks show that our model can significantly outperform previous state-of-the-art methods.

1 Introduction

Human-machine communication is one of the ultimate goals of artificial intelligence. Recently, building a dialogue system with intelligence has sparked increased attention in both academia and industry. Advanced work includes retrieval-based methods (Zhou et al., 2018b; Tao et al., 2019; Han et al., 2021) and generation-based methods (Li et al., 2016; Serban et al., 2016; Zhang et al., 2020). In this paper, we focus on the retrieval-based approaches since they are superior in providing informative and fluent responses to a human input by selecting a proper response from a pre-built index. However, such models are still limited in their ability to fully understand the human query and predict a more engaging response. To this end, some researchers have begun to ground dialogue agents with knowledge (Dinan et al., 2019; Gopalakrishnan et al., 2019; Gunasekara et al., 2019) since humans can naturally associate the content of the conversation with the background knowledge in his/her mind, which has led to improved performance.

Two prominent lines of research have evolved for this task. One is to build retrieval-based knowledge-grounded dialogue models by directly attending to all available knowledge entries (Gu et al., 2019; Zhao et al., 2019; Gu et al., 2020; Hua et al., 2020). The other is to separate the knowledge-grounded response retrieval process into two stages: knowledge retrieving and response ranking (Dinan et al., 2019; Gopalakrishnan et al., 2019; Tao et al., 2021), in which a knowledge retriever first selects relevant knowledge sentences from grounded documents, and then a response ranker incorporates the retrieved knowledge sentences from the knowledge retriever and ranks the candidate responses regarding the dialogue context. However, a long-standing issue on this task is that it is non-trivial to collect large-scale dialogues that are naturally grounded on a small set of knowledge sentences. To train such models, one should first collect pairs of dialogue context and the corresponding list of knowledge sentences that contains the information corresponding to the dialogue context. Unfortunately, hand-labeling data to that end is time-consuming, and many data sets and applications lack such knowledge annotations[1]. Therefore,

*Corresponding author: Dongyan Zhao.

[1]While some data sets, e.g., Wizard of Wikipedia (Dinan et al., 2019), have ground-truth knowledge labels, many other data sets do not, e.g., CMU_DoG (Zhou et al., 2018a).

Proceedings of the 29th International Conference on Computational Linguistics, pages 389–399
October 12–17, 2022.

the above two research lines both suffer from insufficient knowledge supervision. The former is prone to be affected by noise from irrelevant and redundant knowledge when conducting response retrieval, and the knowledge retrieving process of the latter suffers from the lack of labels indicating the ground-truth knowledge sentences. Hence, the challenge we consider is: *How to better optimize the knowledge retriever and response ranker jointly without ground-truth knowledge labels?*

To address the challenge, we follow the two-stage paradigm and propose a **Rec**iprocal learning approach to jointly optimize knowledge retriever and response ranker for response retrieval in **K**nowledge-**G**rounded **C**onversations. We name our model as RECKGC. In reciprocal learning, the knowledge retriever uses the feedback from the response ranker as pseudo supervised signals of knowledge retrieval for updating its parameters, while the response ranker also receives the top-ranked knowledge sentences from the knowledge retriever to optimize itself. We use the posterior estimate to train the knowledge retriever, and use the prior information to train the response ranker. By this means, the knowledge retriever and response ranker can be jointly optimized without ground-truth knowledge labels.

We conduct experiments on two public benchmarks including Wizard of Wikipedia (Dinan et al., 2019) and CMU_DoG (Zhou et al., 2018a). Evaluation results indicate that our model can significantly outperform the existing methods, and achieve new state-of-the-art performance on both data sets. Our contributions in this paper are two-fold: (1) proposal of a reciprocal learning of knowledge retriever and response ranker for knowledge-grounded response retrieval without ground-truth knowledge label; (2) Empirical verification of the effectiveness of the proposed learning approach on two public benchmarks.

2 Related Work

Early studies of retrieval-based dialogue systems focused on building single-turn context-response matching models that consider only a single utterance or several utterances in the context that are concatenated into a long sequence for response selection (Wang et al., 2013, 2015). Recently, more emphasis has been placed on response retrieval with multi-turn dialogue context and lots of impressive results have been obtained, including the dual LSTM model (Lowe et al., 2015), the sequential matching network (SMN) (Wu et al., 2017), the deep attention matching network (DAM) (Zhou et al., 2018b), the multi-hop selector network (MSN) (Yuan et al., 2019). With advances in pre-trained language models (Devlin et al., 2019a; Liu et al., 2019), some researchers also attempt to apply them on response selection: to represent each utterance-response pair with BERT and fuse these representations to compute the context-response matching score (Vig and Ramea, 2019); to directly treat the context as a long sequence and conduct context-response matching with BERT (Whang et al., 2020); to leverage fine-grained post-training for improving retrieval-based dialogue systems (Han et al., 2021).

Inspired by the ability of human beings to associate dialogue content with background knowledge in his/her mind, researchers have begun to ground dialogue agents with knowledge. Zhang et al. (2018) collect a persona-based dialogue corpus which utilizes the interlocutor's profile as background knowledge; Zhou et al. (2018a) publish a corpus which contains conversations grounded in articles about popular movies; Dinan et al. (2019) release another corpus with Wiki articles as grounded documents which cover a wide range of topics. At the same time, lots of representative models have been obtained. Zhao et al. (2019); Gu et al. (2019); Hua et al. (2020) successively put forward document-grounded matching network (DGMN), dually interactive matching network (DIM), and RSM-DCK which let the dialogue context and all knowledge sentences interact with candidate responses respectively with cross-attention mechanism. Gu et al. (2020) propose a document-grounded model named FIRE which first compute the importance score for each context turn and knowledge sentence, then further use them to weigh the corresponding representation. Dinan et al. (2019) also propose to joint learn the knowledge selection and response matching in a multi-task manner or a two-stage training procedure. This strategy, however, requires ground-truth knowledge labels annotated by human wizards, which is presumed absent in our paper. Recently, Tao et al. (2021) study response matching in knowledge-grounded conversations under a zero-resource setting. In particular, they propose decomposing the training of the knowledge-grounded response selection into three tasks and jointly training all tasks in a unified

pre-trained language model.

3 Methodology: RECKGC

In this section, we first formalize the task of knowledge-grounded response retrieval and then introduce our model from overview to several components to reciprocal learning of them.

3.1 Problem Formalization

Suppose that we have a knowledge-grounded dialogue data set $\mathcal{D} = \{C_i, K_i, r_i, y_i\}_{i=1}^{N}$, where C_i is a dialogue context that is the concatenated token sequence of multi-turn utterances, $K_i = \{k_1, k_2, \ldots, k_{n_k}\}$ is a collection of background knowledge for conversation with k_j the j-th knowledge sentence and n_k is the number of knowledge sentences; r_i is a candidate response; $y_i = 1$ indicates that r_i is a proper response for C_i and K_i, otherwise, $y_i = 0$. The goal is to learn a matching model $g(C, K, r)$ from $\mathcal{D}$, and thus for any new context-knowledge-response triple (C, K, r), $g(C, K, r)$ returns the matching degree between r and (C, K). Finally, given a series of candidate responses regarding the same (C, K), one can collect the matching scores and conduct response ranking.

3.2 Model Overview

Our model is composed of two modules, the knowledge retriever and the response ranker. Given an input dialogue context and a collection of background knowledge sentences, these modules are used in a two-step process to predict a response. First, the *knowledge retriever* selects a small subset of knowledge sentences from the knowledge collection where some of them contain relevant information regarding the dialogue context. Then these extracted knowledge sentences are processed by the *response ranker*, along with the dialogue context, to thoroughly examine the selected knowledge and contexts, and predict the matching degree of a candidate response. Figure 1 shows an illustration of our model and reciprocal learning procedure. For the knowledge retriever, we use a dual-encoder architecture (Bromley et al., 1993), which is efficient for processing potential massive of knowledge sentences. For the response ranker, we leverage the standard transformer architecture, which performs full attention over the inputs and gives considerable natural language understanding performance. Both of the modules can be initialized from pre-trained language models such as BERT (Devlin et al., 2019b) or RoBERTa (Liu et al., 2019).

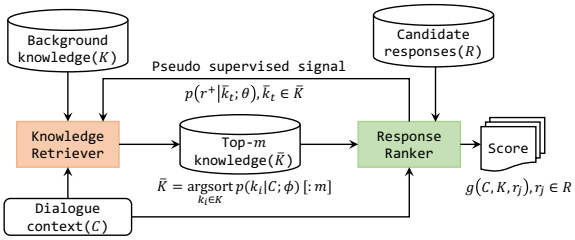

Figure 1: The illustration of our proposed model and reciprocal learning procedure.

The focus of this work is to train the knowledge retriever without ground-truth knowledge labels and conduct the reciprocal learning of knowledge retriever and response ranker in an end-to-end setting. We discuss each component and our training objective in detail below.

3.3 Knowledge Retriever

Given a dialogue context C and a collection of background knowledge $K = \{k_1, k_2, \ldots, k_{n_k}\}$, we propose a knowledge retriever is to select a relevant subset of knowledge sentences for the context. For this purpose, the retriever performs a ranking of the knowledge sentences conditioned on the dialogue context and outputs the top-ranked knowledge sentences.

Following Dinan et al. (2019), we leverage a knowledge retriever model composed of an embedder function $E_{retr}(\cdot)$ that maps any knowledge sentence $k_i \in K$ or dialogue context C to a d-dimensional vector, such that the similarity score between dialogue context C and a knowledge sentence k_i can be defined as a scaled dot product of their representation vectors:

$$s(C, k_i; \phi) = \frac{E_{retr}(C)^T E_{retr}(k_i)}{\sqrt{d}} \quad (1)$$

where $\sqrt{d}$ is a relevance score scaling referring to Sachan et al. (2021), and the retriever is parameterized by ϕ. Although in principle the embedder function $E_{retr}(\cdot)$ can be implemented by any neural networks, in this work we use BERT-Small (Turc et al., 2019), which is a smaller BERT (28M) compared to BERT-base (110M) to take advantage of pre-training while decreasing the number of network parameters. We take the representation at the `[CLS]` token as the output, thus $d = 512$. Differently from the traditional dual-encoder, we use the same encoding function $E_{retr}(\cdot)$ for the context and knowledge sentence by sharing parameters.

The probability of a knowledge sentence k_i being relevant to the context C is defined as:

$$p(k_i|C;\phi) = \frac{exp(s(C,k_i;\phi))}{\sum_{t=1}^{n_k} exp(s(C,k_t;\phi))} \quad (2)$$

where $k_i \in K$. Through Eq. 2, we can obtain the top-m knowledge sentences with the highest individual score as $\bar{K} = \{\bar{k}_1, \bar{k}_2, \ldots, \bar{k}_m\}$.

3.4 Response Ranker

Besides the knowledge retriever, our model consists of a response ranker that outputs the matching degree of a candidate response r_j based on retrieved knowledge sentences $\bar{K}$ and dialogue context C. We consider fine-tuning the existing PLMs, which is BERT-base (110M) in our paper, to obtain more competent dialogue modeling performance. Concisely, we first concatenate retrieved knowledge sentences $\bar{K}$, dialogue context C and candidate response r_j as a consecutive token sequence with special tokens separating them as,

$$\begin{aligned} x_j = \{&\texttt{[CLS]}, \bar{k}_1, \texttt{[SEP]}, \ldots, \bar{k}_m, \texttt{[SEP]}, \\ &C, \texttt{[SEP]}, r_j, \texttt{[SEP]}\} \end{aligned} \quad (3)$$

Then token, position and segment embeddings of each word of x_j are summated and fed into another embedder function $E_{rank}(\cdot)$ (i.e. BERT-base). Finally, we achieve the contextualized embedding $E_{rank}(x_j)$ and feed it into a multi-layer perceptron (MLP) to obtain the final matching degree of a candidate response r_j as:

$$h(C,\bar{K},r_j;\theta) = W_2 \cdot f(W_1 \cdot E_{rank}(x_j) + b_1) + b_2 \quad (4)$$

where W_1, W_2, b_1, b_2 are learnt parameters, $f(\cdot)$ is a `tanh` activation function, and the ranker is parameterized by θ. The probability of a candidate response r_j being proper to the context C and retrieved knowledge sentences $\bar{K}$ is calculated as

$$p(r_j|C,\bar{K};\theta) = \frac{exp(h(C,\bar{K},r_j;\theta))}{\sum_{t=1}^{n_r} exp(h(C,\bar{K},r_t;\theta))} \quad (5)$$

where n_r is the number of candidate responses regarding the C and $\bar{K}$. We denote the collection of candidate responses for the context C and background knowledge K as R which contains a ground-truth candidate response r^+, hence the size of R is n_r and $r_j \in R$. Now we can obtain the top candidate response with the highest probability as the output of knowledge-grounded dialogue system from Eq. 5.

3.5 Reciprocal Learning of Knowledge Retriever and Response Ranker

Contrary to previous work on knowledge-grounded response retrieval, we propose a reciprocal learning approach to jointly optimize the knowledge retriever and the response ranker in an end-to-end differentiable fashion. While in this paper, we assume that there are no ground-truth labels for extracting relevant knowledge, which is practical but makes the problem even more challenging.

In reciprocal learning, the trainable components consist of the knowledge retriever (ϕ) and response ranker (θ) parameters. For the training objective of the overall model, we propose to find ϕ and θ that would maximize the likelihood of a ground-truth response r^+ as:

$$p(r^+|C,K;\phi,\theta) = \sum_{\bar{K}\subset K} p(r^+|C,\bar{K};\theta)p(\bar{K}|C;\phi) \quad (6)$$

However, marginalizing over all possible values of $\bar{K}$, which is a subset of retrieved knowledge sentences, is intractable as it is essentially combinatorial. The log-likelihood of a particular value of $\bar{K}$ is:

$$\begin{aligned} \log p(r^+|C,\bar{K};\theta)p(\bar{K}|C;\phi) = \\ \underbrace{\log p(r^+|C,\bar{K};\theta)}_{\text{response ranker}} + \underbrace{\log p(\bar{K}|C;\phi)}_{\text{knowledge retriever}} \end{aligned} \quad (7)$$

where the first term is parameterized by response ranker θ and the second term is parameterized by knowledge retriever ϕ. We discuss how to optimize both components in detail below.

Optimization of the Response Ranker (θ) For updating θ, we maximize the first term of Eq. 7. Specifically, we first construct $\bar{K}$ by retrieving the top-m relevant knowledge sentences that have the highest similarity scores from retriever. The similarity score is computed by Eq. 1 based on the current value of knowledge retriever parameters ϕ. Since we already have the ground-truth response r^+, the training objective of the response ranker for each training sample can be defined as the negative log-likelihood loss:

$$\mathcal{L}_\theta = -\log p(r^+|C,\bar{K};\theta) \quad (8)$$

where the probability of the ground-truth response $p(r^+|C,\bar{K};\theta)$ can be computed by Eq. 5.

Optimization of the Knowledge Retriever (ϕ) For updating ϕ, we maximize the second term of Eq. 7. However, since there are no ground-truth labels for extracting relevant knowledge, we can not simply optimize the knowledge retriever (ϕ) by the negative log-likelihood loss similar to Eq. 8. To solve the problem, we consider incorporating the posterior information to provide additional guidance on obtaining appropriate knowledge during training, and the posterior estimate of the second term is formulated as $\log p(\bar{K}|C, r^+; \theta, \phi)$. Since it is non-trivial to maximize a probability of a set, we instead maximize the sum of the probability of each knowledge sentence $\bar{k}_t$ in the set $\bar{K}$, i.e. $\log \sum_{t=1}^{m} p(\bar{k}_t|C, r^+; \theta, \phi)$. The probability of each knowledge sentence $p(\bar{k}_t|C, r^+; \theta, \phi)$ can be further rewritten using the Bayes Rule:

$$p(\bar{k}_t|C, r^+; \theta, \phi) = \frac{p(r^+|C, \bar{k}_t; \theta)p(\bar{k}_t|C; \phi)}{p(r^+|C; \theta, \phi)} \propto p(r^+|C, \bar{k}_t; \theta)p(\bar{k}_t|C; \phi) \tag{9}$$

Here we choose not to normalize with denominator $p(r^+|C; \theta, \phi)$ because computing this quantity would necessitate summing over all n_k knowledge sentences[2]. The response ranker now computes the probability of ground-truth response $p(r^+|C, \bar{k}_t; \theta)$ conditioned on only one knowledge sentence $\bar{k}_t$ with a current value of θ.

In fact, in knowledge-based dialogues, lots of samples may be able to match ground-truth response only based on the dialogue context which contains enough retrieval clues (would be illustrated in the ablation study in Section 4.5). In this case, the contribution from knowledge would be very small and the training time may be increased. Meanwhile, we also consider introducing heuristic similarity unigram F1 (denoted as $\rho(\cdot, \cdot)$) between a retrieved knowledge $\bar{k}_t$ and ground-truth response r^+ as supplementary posterior information for supervising knowledge retriever, as we intuitively believe human responses have a strong correlation with the selected knowledge sentence. To better measure the contribution of $\bar{k}_t$, while reducing distractions of the dialogue context C, we estimate $p(r^+|C, \bar{k}_t; \theta)$ as:

$$p(r^+|\bar{k}_t; \theta) + \rho(\bar{k}_t, r^+) \tag{10}$$

where the probability $p(r^+|\bar{k}_t; \theta)$ can be obtained similar to Eq. 5 but the context C is removed from the input sequence of Eq. 3. In our preliminary experiments, we observe that the introduction of the dialogue context C makes the training of knowledge retriever unstable and degrades the performance.

Then, we compute $p(\bar{k}_t|C; \phi)$ by

$$p(\bar{k}_i|C; \phi) = \frac{exp(s(C, \bar{k}_i; \phi)/\tau)}{\sum_{t=1}^{m} exp(s(C, \bar{k}_t; \phi)/\tau)} \tag{11}$$

where $\bar{k}_t \in \bar{K}$. Note that there is a slight difference in form between Eq. 11 and Eq. 2 where we do not sum over all knowledge sentences K in the denominator which may be massive in practice. As an alternative, we introduce τ as a temperature hyperparameter assuming that knowledge sentences beyond the top-m contribute very small scores to the approximation. The training objective of knowl-

Algorithm 1: The proposed reciprocal learning approach

Input: Training set $\mathcal{D}$, knowledge retriever ϕ, response ranker θ, learning rate η_ϕ, η_θ, number of epochs N_{ep}, number of iterations N_{it};

Initialize knowledge retriever ϕ and response ranker θ with BERT-small and BERT-base respectively;

for $e = 1, 2, ..., N_{ep}$ **do**

  Shuffle training set $\mathcal{D}$;

  for $t = 1, 2, ..., N_{it}$ **do**

    Fetch a batch of training data $\mathcal{B}$;

    Obtain the top-m knowledge sentences $\bar{K}$ with current value of ϕ by Eq 2;

    Compute $\mathcal{L}_\theta$ with $\bar{K}$ by Eq. 3, 4, 5 and 8;

    Compute the gradients and **update** θ:

$$\theta \leftarrow \theta + \eta_\theta \frac{\partial \mathcal{L}_\theta(\mathcal{B})}{\partial \theta}$$

    Compute $\mathcal{L}_\phi$ with $\bar{K}$ and current value of θ by Eq. 9, 10, 11 and 12 ;

    Compute the gradients and **update** ϕ:

$$\phi \leftarrow \phi + \eta_\phi \frac{\partial \mathcal{L}_\phi(\mathcal{B})}{\partial \phi}$$

Output: ϕ, θ.

[2]Nevertheless, we observe that our training method still behaves well in practice.

Statistics	Wizard of Wikipedia				CMU_DoG		
	Train	Valid	Test Seen	Test Unseen	Train	Valid	Test
# Utterances	166,787	17,715	8,715	8,782	74,717	4,993	13,646
# Conversations	18,430	1,948	965	968	3,373	229	619
# Topics/Documents	1,247	599	533	58	30	30	30
Avg. # turns	9.0	9.1	9.0	9.1	22.2	21.8	22.0
Avg. # words per turn	16.4	16.4	16.4	16.1	18.6	20.1	18.1
Avg. # knowledge entries	61.2	61.5	60.8	61.0	31.3	30.4	31.8
Avg. # words per knowledge	37.2	37.6	36.9	37.0	27.2	28.2	27.0

Table 1: The statistics of two benchmarks.

edge retriever is to minimize the following loss:

$$\mathcal{L}_\phi = -\log \sum_{t=1}^{m} p(\bar{k}_t|C, r^+; \theta, \phi) \quad (12)$$

Joint Optimization of Overall Model In the overall model, the knowledge retriever and response ranker are jointly optimized in an end-to-end differentiable way. The training objective is to minimize:

$$\mathcal{L} = \mathcal{L}_\phi + \mathcal{L}_\theta \quad (13)$$

Intuitively, we train the response ranker using prior knowledge distribution in Eq. 2 since we do not introduce the information from ground-truth response r^+. Consequently, there is no mismatch between training and inference, which is useful when the ground-truth response is not known during inference. While for the knowledge retriever, we introduce additional information from r^+ for posterior estimate to learn from richer training signals rather than relying solely on the prior. Algorithm 1 demonstrates the pseudo code of our proposed reciprocal learning approach.

4 Experiments

To demonstrate the effects of the proposed models, we conduct experiments on two public data sets.

4.1 Benchmarks and Evaluation Metrics

We evaluate the proposed method on two public benchmarks including Wizard of Wikipedia (WoW) (Dinan et al., 2019) and CMU Document Grounded Conversations (CMU_DoG) (Zhou et al., 2018a). The statistics of the two benchmarks are shown in Table 1.

The first benchmark we employ is the Wizard of Wikipedia (WoW) (Dinan et al., 2019). During the conversation collection, one of the paired speakers is asked to play the role of a knowledgeable expert with access to the given knowledge collection, while the other one acts as a curious learner. The test set is divided into two subsets by Dinan et al. (2019): Test Seen and Test Unseen. The former shares 533 common topics with the training set, while the latter contains 58 new topics uncovered by the training or validation set. In the validation set or test set, the ratio between positive and negative responses is 1:99. Since the training data set do not contain negative responses, we adopt in-batch negatives consistent with Dinan et al. (2019), where the ground-truth responses of the other batch elements are treated as negative training responses.

The second benchmark we use is CMU_DoG data set published in Zhou et al. (2018a). Amazon Mechanical Turk is used to collect conversations based on certain knowledge documents in this data set. The knowledge topics are all about movies, which provide interlocutors with common topics to discuss in a natural way. Two situations are investigated to compel two paired workers to talk about the given documents. In the first one, only one interlocutor has access to the document, while the other does not. The interlocutor with access to the given knowledge document is instructed to introduce the movie to the other. In the second one, both interlocutors can see the given document and are required to talk about its content. Consistent with previous works, we follow Zhao et al. (2019) and merge data in the two scenarios to form a larger data set considering the small number of conversations in each scenario. The ratio between positive and negative responses is 1:19. For a fair comparison, we use the version of data released by DGMN (Zhao et al., 2019).

Consistent with the widely adopted settings on these two benchmarks, we employ recall n at k (i.e., R@k, where $n = 100$ for WoW and $n = 20$ for CMU_DoG and $k = \{1, 2, 5\}$) as the evaluation metrics of response ranking in Table 2 and Table 3, measuring if the ground-truth response

can be ranked in top k positions when there are n response candidates. For evaluating the performance of knowledge retrieving in Table 4, we also use recall n_k at k to measure if the ground-truth knowledge can be ranked in top k positions when there are n_k knowledge sentences on WoW data.

4.2 Baselines

As the characteristics of the two benchmarks are different (e.g. only WoW data provide the ground-truth knowledge labels), we compare the proposed model with the baselines on both data individually.

Baselines on WoW. 1) *IR Baseline* (Dinan et al., 2019) uses word overlap for response selection; 2) *BoW MemNet* (Dinan et al., 2019) is a memory network where knowledge sentences are embedded with bag-of-words representation, and the model jointly learns the knowledge selection and response matching; 3) *Two-stage Transformer* (Dinan et al., 2019) trains two individual Transformers for knowledge selection and response retrieval respectively. The best-performing model on knowledge selection is selected for dialogue retrieval; 4) *Transformer MemNet* (Dinan et al., 2019) is an extension of BoW MemNet, and the dialogue context, knowledge sentences and candidate responses are encoded with Transformer encoder that pre-trained on a large-scale corpus; 5) *PTKGC* (Tao et al., 2021) conduct knowledge-grounded response matching in a zero-resource setting, which decomposes the training of response selection into three tasks and jointly trains all tasks in a unified model. It should be noted that we do not compare with the MNDB model (Zhang et al., 2021), because the authors reconstruct the dataset and only retain 32 knowledge candidates for each dialogue, which make this task easier.

Baselines on CMU_DoG 1) *Starspace* (Wu et al., 2018) match the response using the cosine similarity between a concatenated sequence of dialogue context and knowledge, and the response candidate represented by StarSpace; 2) *BoW MemNet* (Zhang et al., 2018) is a memory network with the BOW representation of knowledge as memory entries; 3) *KV Profile Memory* (Zhang et al., 2018) is a key-value memory network grounded on knowledge profiles; 4) *Transformer* (Mazare et al., 2018) encode all utterances with a pre-trained Transformer similar to BoW MemNet; 5) *DGMN* (Zhao et al., 2019) lets the dialogue context and all knowledge sentences interact with the candidate response respectively through cross-attention; 6) *DIM* (Gu et al., 2019) is similar to DGMN and all utterance are encoded with BiLSTMs; 7) *RSM-DCK* (Hua et al., 2020) obtains query-aware knowledge representation and query-aware context representation for response matching; 8) *FIRE* (Gu et al., 2020) filters the context and knowledge first and then use the filtered context and knowledge to iteratively conduct response matching.

4.3 Technical Details

The response ranker and knowledge retriever are implemented with *transformers* library provided by huggingface[3]. Adam (Kingma and Ba, 2015) with $\beta_1 = 0.9$ and $\beta_2 = 0.999$ is the optimizer and the initial learning rate of knowledge retriever and response ranker are 1e-5 and 3e-5. We choose 32 as the size of mini-batches for training on WoW data and 8 on CMU_DoG. During the training, the maximum lengths of the knowledge sentence, dialogue context, and response candidate are set to 40, 80 and 60 on WoW data respectively, while on CMU_DoG we set the the maximum length of the dialogue context as 200. m is set as 5 on both data. τ is set to 0.2 on Eq. 11. We avoid computing the gradients of response ranker parameters θ during estimating $p(r^+|\bar{k}_t;\theta)$ on Eq. 10. Early stopping on the validation set is adopted as a regularization strategy. The best model is selected based on the validation performance.

4.4 Evaluation Results

Performance of Response Ranking. Table 2 and 3 provide the evaluation results of response selection on WoW and CMU_DoG respectively. Numbers in bold mean that improvement over the best baseline is statistically significant (t-test with $p < 0.05$). Our proposed RECKGC can significantly outperform state-of-the-art models across all metrics on both data. Besides, it is interesting to find that our model achieves more improvement gain on Test Unseen set than Test Seen compared with baselines. The results may be attributed to our model's superior generalization abilities on dialogues with new topics as compared to the previous work, demonstrating the advantages of our proposed reciprocal learning approach.

[3] https://github.com/huggingface/transformers

Models	Test Seen			Test Unseen		
	R@1	R@2	R@5	R@1	R@2	R@5
IR Baseline (Dinan et al., 2019)	17.8	-	-	14.2	-	-
BoW MemNet (Dinan et al., 2019)	71.3	-	-	33.1	-	-
Two-stage Transformer (Dinan et al., 2019)	84.2	-	-	63.1	-	-
Transformer MemNet (Dinan et al., 2019)	87.4	-	-	69.8	-	-
PTKGC (Tao et al., 2021)	89.5	96.7	98.9	69.6	85.8	96.3
DIM (Gu et al., 2019)	83.1	91.1	95.7	60.3	77.8	92.3
FIRE (Gu et al., 2020)	88.3	95.3	97.7	68.3	84.5	95.1
RECKGC	**92.6**	**97.2**	**99.2**	**76.7**	**88.7**	**96.6**

Table 2: Evaluation results of response selection on the test sets of the Wizard of Wikipedia data. Numbers in bold mean that improvement over the best baseline is statistically significant (t-test, p-value < 0.05).

Models	R@1	R@2	R@5
Starspace (Wu et al., 2018)	50.7	64.5	80.3
BoW MemNet (Zhang et al., 2018)	51.6	65.8	81.4
KV Profile Memory (Zhang et al., 2018)	56.1	69.9	82.4
Transformer (Mazare et al., 2018)	60.3	74.4	87.4
PTKGC (Tao et al., 2021)	66.1	77.8	88.7
DGMN (Zhao et al., 2019)	65.6	78.3	91.2
DIM (Gu et al., 2019)	78.7	89.0	97.1
RSM-DCK (Hua et al., 2020)	79.3	88.8	96.7
FIRE (Gu et al., 2020)	81.8	90.8	97.4
RECKGC	**84.0**	**92.9**	**98.2**

Table 3: Evaluation results of response selection on the test set of the CMU_DoG data. Numbers in bold mean that improvement over the best baseline is statistically significant (t-test, p-value < 0.05).

Performance of Knowledge Retrieving. Since the WoW data contain the ground-truth knowledge labels, we also assess the performance of knowledge retriever with Recall-based metrics in Table 4. Besides, we design two baselines where knowledge retriever (a dual-encoder) is merely trained with supervised or weakly supervised labels. First, we train it with ground-truth knowledge labels (denoted as *"Dual-Enc (supervised)"*). Then in the weakly supervised scenario, we consider $k_i \in K$ that has the highest $\rho(k_i, r^+)$ in each sample as pseudo ground-truth knowledge (denoted as *"Dual-Enc (weakly supervised)"*). We can find that training with ground-truth knowledge labels brings more improvement to the dual encoder than training with pseudo knowledge labels, indicating that pseudo labels are just a sub-optimal supervised learning signal. Notably, the knowledge retriever trained with our proposed reciprocal learning approach outperforms several supervised or weakly supervised baselines and obtains comparable results with *"Dual-Enc (supervised)"*, which proves the effectiveness of our approach.

4.5 Discussions

Ablation Study. We conduct a comprehensive ablation study to investigate the impact of different inputs, posterior information and learning strategies. Table 5 also provides the ablation results. Firstly, we remove the knowledge from the model, which is denoted as "RECKGC *(w/o. knowledge)*". This model is degraded into a traditional context-response matching paradigm. We can find that removing the knowledge will lead to a dramatic performance drop, which indicates that knowledge is important in response retrieval. However, this model can still outperform some baselines such as BoW MemNet, which proves that some ground-truth responses can be inferred only from the context. Then, we remove the $p(r^+|\bar{k}_t;\theta)$ and $\rho(\bar{k}_t, r^+)$ on Eq. 10 and denote them as "RECKGC *(w/o. kr)*" and "RECKGC *(w/o. f1)*" respectively. We can easily conclude that both posterior information is useful, as removing either information leads to a certain degree of performance degradation. Finally, to prove the advantages of joint learning, we also propose a two-stage training baseline (denoted as *"Two-stage training"*) where we first train the knowledge retriever with pseudo ground-truth knowledge, and then freeze the parameters of knowledge retriever and train the response ranker conditioned on top-m knowledge sentences provided by knowledge retriever. Our model can consistently outperform the model with two-stage training, which confirms the rationality of our reciprocal learning approach.

The Impact of the Number of Retrieved Knowledge Sentences. Furthermore, we investigate how the number of retrieved knowledge sentences

Models	Test Seen			Test Unseen		
	R@1	R@2	R@5	R@1	R@2	R@5
Random	2.7	-	-	2.3	-	-
IR Baseline	5.8	-	-	7.6	-	-
BoW MemNet	23.0	-	-	8.9	-	-
Transformer	22.5	-	-	12.2	-	-
PTKGC	22.0	31.2	48.8	23.1	32.1	50.7
Dual-Enc (weakly supervised)	22.3	33.1	54.3	21.5	31.6	53.1
Dual-Enc (supervised)	23.1	34.0	55.8	22.4	33.4	53.2
RECKGC	22.8	33.6	55.7	23.2	32.9	53.7

Table 4: The performance of knowledge retriever on the test sets of WoW data.

Models	Test Seen			Test Unseen		
	R@1	R@2	R@5	R@1	R@2	R@5
RECKGC	92.6	97.2	99.2	76.7	88.7	96.6
RECKGC (w/o. knowledge)	88.0	94.5	97.6	70.8	84.8	94.5
RECKGC (w/o. f1)	91.7	96.5	99.0	75.3	89.1	96.3
RECKGC (w/o. kr)	92.0	96.8	99.1	74.7	88.6	96.2
Two-stage training	89.2	95.5	98.5	71.6	86.9	95.9

Table 5: Ablation results on two test sets of WoW data.

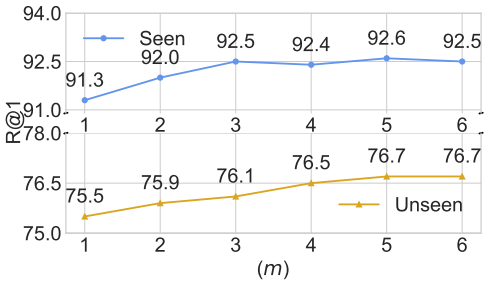

Figure 2: Performance of RECKGC across different number of retrieved knowledge sentences (i.e. m) on Test Seen set and Test Unseen set of WoW data.

(i.e. m) influences the model performance. Figure 2 shows the performance of response selection on test sets of WoW with respect to different m. The curves first monotonically increase until m reaches 5, and then stabilize when m keeps increasing. The reason could be that when only a few knowledge sentences are provided for dialogue, the model cannot capture enough information for response matching, but when the retrieved knowledge becomes sufficient, noise would be introduced into matching because redundant knowledge may be irrelevant to the current dialogue context.

5 Conclusion

In this paper, we study the retrieval-based knowledge-grounded dialogues. To effectively optimize the knowledge retriever and response ranker without ground-truth knowledge labels, we propose a reciprocal learning approach to jointly optimize the two components in an end-to-end way. Concretely, the knowledge retriever takes the feedback from the response ranker as pseudo supervised signals of knowledge retrieval, while the response ranker receives the top-ranked knowledge sentences from the knowledge retriever to optimize itself. By this means, our model can be trained without ground-truth knowledge labels. Evaluation results on two benchmarks indicate that our model can significantly outperform state-of-the-art methods.

Acknowledgement

We would like to thank the anonymous reviewers for their constructive comments and suggestions. This work was supported by National R&D project of China under contract No. 2021YFC3340303. Dongyan Zhao is the corresponding author.

References

Jane Bromley, James W Bentz, Léon Bottou, Isabelle Guyon, Yann LeCun, Cliff Moore, Eduard Säckinger, and Roopak Shah. 1993. Signature verification using a "siamese" time delay neural network. *International Journal of Pattern Recognition and Artificial Intelligence*, 7(04):669–688.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019a. Bert: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019b. BERT: Pre-training of

deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, Minneapolis, Minnesota. Association for Computational Linguistics.

Emily Dinan, Stephen Roller, Kurt Shuster, Angela Fan, Michael Auli, and Jason Weston. 2019. Wizard of wikipedia: Knowledge-powered conversational agents. In *International Conference on Learning Representations*.

Karthik Gopalakrishnan, Behnam Hedayatnia, Qinglang Chen, Anna Gottardi, Sanjeev Kwatra, Anu Venkatesh, Raefer Gabriel, Dilek Hakkani-Tür, and Amazon Alexa AI. 2019. Topical-chat: Towards knowledge-grounded open-domain conversations. In *INTERSPEECH*, pages 1891–1895.

Jia-Chen Gu, Zhen-Hua Ling, Xiaodan Zhu, and Quan Liu. 2019. Dually interactive matching network for personalized response selection in retrieval-based chatbots. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 1845–1854.

Jia-Chen Gu, Zhenhua Ling, Quan Liu, Zhigang Chen, and Xiaodan Zhu. 2020. Filtering before iteratively referring for knowledge-grounded response selection in retrieval-based chatbots. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: Findings*, pages 1412–1422.

Chulaka Gunasekara, Jonathan K. Kummerfeld, Lazaros Polymenakos, and Walter Lasecki. 2019. DSTC7 task 1: Noetic end-to-end response selection. In *Proceedings of the First Workshop on NLP for Conversational AI*, pages 60–67, Florence, Italy. Association for Computational Linguistics.

Janghoon Han, Taesuk Hong, Byoungjae Kim, Youngjoong Ko, and Jungyun Seo. 2021. Fine-grained post-training for improving retrieval-based dialogue systems. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 1549–1558, Online. Association for Computational Linguistics.

Kai Hua, Zhiyuan Feng, Chongyang Tao, Rui Yan, and Lu Zhang. 2020. Learning to detect relevant contexts and knowledge for response selection in retrieval-based dialogue systems. In *Proceedings of the 29th ACM International Conference on Information & Knowledge Management*, pages 525–534.

Diederik P Kingma and Jimmy Ba. 2015. Adam: A method for stochastic optimization. In *ICLR*.

Jiwei Li, Michel Galley, Chris Brockett, Jianfeng Gao, and William B Dolan. 2016. A diversity-promoting objective function for neural conversation models. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 110–119.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Roberta: A robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692*.

Ryan Lowe, Nissan Pow, Iulian Vlad Serban, and Joelle Pineau. 2015. The ubuntu dialogue corpus: A large dataset for research in unstructured multi-turn dialogue systems. In *Proceedings of the 16th Annual Meeting of the Special Interest Group on Discourse and Dialogue*, pages 285–294.

Pierre-Emmanuel Mazare, Samuel Humeau, Martin Raison, and Antoine Bordes. 2018. Training millions of personalized dialogue agents. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 2775–2779.

Devendra Sachan, Mostofa Patwary, Mohammad Shoeybi, Neel Kant, Wei Ping, William L. Hamilton, and Bryan Catanzaro. 2021. End-to-end training of neural retrievers for open-domain question answering. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 6648–6662, Online. Association for Computational Linguistics.

Iulian V Serban, Alessandro Sordoni, Yoshua Bengio, Aaron Courville, and Joelle Pineau. 2016. Building end-to-end dialogue systems using generative hierarchical neural network models. In *Proceedings of the Thirtieth AAAI Conference on Artificial Intelligence*, pages 3776–3783.

Chongyang Tao, Changyu Chen, Jiazhan Feng, Ji-Rong Wen, and Rui Yan. 2021. A pre-training strategy for zero-resource response selection in knowledge-grounded conversations. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 4446–4457, Online. Association for Computational Linguistics.

Chongyang Tao, Wei Wu, Can Xu, Wenpeng Hu, Dongyan Zhao, and Rui Yan. 2019. One time of interaction may not be enough: Go deep with an interaction-over-interaction network for response selection in dialogues. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 1–11, Florence, Italy. Association for Computational Linguistics.

Iulia Turc, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. Well-read students learn better: On the importance of pre-training compact models. *arXiv preprint arXiv:1908.08962v2*.

Jesse Vig and Kalai Ramea. 2019. Comparison of transfer-learning approaches for response selection in multi-turn conversations. In *Workshop on DSTC7*.

Hao Wang, Zhengdong Lu, Hang Li, and Enhong Chen. 2013. A dataset for research on short-text conversations. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*, pages 935–945.

Mingxuan Wang, Zhengdong Lu, Hang Li, and Qun Liu. 2015. Syntax-based deep matching of short texts. In *Proceedings of the 24th International Conference on Artificial Intelligence*, IJCAI'15, page 1354–1361. AAAI Press.

Taesun Whang, Dongyub Lee, Chanhee Lee, Kisu Yang, Dongsuk Oh, and HeuiSeok Lim. 2020. An effective domain adaptive post-training method for bert in response selection. In *Proc. Interspeech 2020*.

Ledell Yu Wu, Adam Fisch, Sumit Chopra, Keith Adams, Antoine Bordes, and Jason Weston. 2018. Starspace: Embed all the things! In *Thirty-Second AAAI Conference on Artificial Intelligence*.

Yu Wu, Wei Wu, Chen Xing, Ming Zhou, and Zhoujun Li. 2017. Sequential matching network: A new architecture for multi-turn response selection in retrieval-based chatbots. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 496–505.

Chunyuan Yuan, Wei Zhou, Mingming Li, Shangwen Lv, Fuqing Zhu, Jizhong Han, and Songlin Hu. 2019. Multi-hop selector network for multi-turn response selection in retrieval-based chatbots. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 111–120.

Chen Zhang, Hao Wang, Feijun Jiang, and Hongzhi Yin. 2021. Adapting to context-aware knowledge in natural conversation for multi-turn response selection. In *Proceedings of the Web Conference 2021*, pages 1990–2001.

Saizheng Zhang, Emily Dinan, Jack Urbanek, Arthur Szlam, Douwe Kiela, and Jason Weston. 2018. Personalizing dialogue agents: I have a dog, do you have pets too? In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 2204–2213.

Yizhe Zhang, Siqi Sun, Michel Galley, Yen-Chun Chen, Chris Brockett, Xiang Gao, Jianfeng Gao, Jingjing Liu, and William B Dolan. 2020. Dialogpt: Large-scale generative pre-training for conversational response generation. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics: System Demonstrations*, pages 270–278.

Xueliang Zhao, Chongyang Tao, Wei Wu, Can Xu, Dongyan Zhao, and Rui Yan. 2019. A document-grounded matching network for response selection in retrieval-based chatbots. In *Proceedings of the 28th International Joint Conference on Artificial Intelligence*, pages 5443–5449. AAAI Press.

Kangyan Zhou, Shrimai Prabhumoye, and Alan W Black. 2018a. A dataset for document grounded conversations. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 708–713.

Xiangyang Zhou, Lu Li, Daxiang Dong, Yi Liu, Ying Chen, Wayne Xin Zhao, Dianhai Yu, and Hua Wu. 2018b. Multi-turn response selection for chatbots with deep attention matching network. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1118–1127.

CR-GIS: Improving Conversational Recommendation via Goal-aware Interest Sequence Modeling

Jinfeng Zhou[1,2], **Bo Wang**[1,2]*, **Zhitong Yang**[1,2], **Dongming Zhao**[3], **Kun Huang**[3], **Ruifang He**[1,2], **Yuexian Hou**[1]

[1]College of Intelligence and Computing, Tianjin University, Tianjin, China
[2]State Key Laboratory of Communication Content Cognition,
[2]People's Daily Online, Beijing, China
[3]AI Lab, China Mobile Communication Group Tianjin Co., Ltd.
{jfzhou, bo_wang, yyyyyyzt}@tju.edu.cn

Abstract

Conversational recommendation systems (CRS) aim to determine a goal item by sequentially tracking users' interests through multi-turn conversation. In CRS, implicit patterns of user interest sequence guide the smooth transition of dialog utterances to the goal item. However, with the convenient explicit knowledge of knowledge graph (KG), existing KG-based CRS methods over-rely on the explicit separate KG links to model the user interests but ignore the rich goal-aware implicit interest sequence patterns in a dialog. In addition, interest sequence is also not fully used to generate smooth transited utterances. We propose CR-GIS with a parallel star framework. First, an interest-level star graph is designed to model the goal-aware implicit user interest sequence. Second, a hierarchical Star Transformer is designed to guide the multi-turn utterances generation with the interest-level star graph. Extensive experiments verify the effectiveness of CR-GIS in achieving more accurate recommended items with more fluent and coherent dialog utterances.

1 Introduction

Traditional recommendation systems often interact with users in a one-shot, one-directional manner (Jannach et al., 2021), that is, users passively receive the static recommendation list and the recommendation system lacks the ability to understand and proactively guide the dynamic shift of users' interests. Conversational Recommendation Systems (CRS) (Sun and Zhang, 2018; Li et al., 2018) solve these problems by supporting multi-turn goal-oriented (Kang et al., 2019; Zhou et al., 2020b) dialog to proactively track and guide real-time user interests shift (Gao et al., 2021).

In current studies, one most popular way of CRS is to determine an item meeting the user preference through multi-turn conversation. Regarding the item to be determined as the goal, we regard this kind of CRS as "goal-aware" CRS. In this paper, we suppose that the performance of goal-aware CRS is highly dependent on a goal-aware sequence of user interests which is expressed in the form of logically transited utterance sequence in dialog. From the view of recommendation, in a goal-aware CRS dialog, a proper dynamic user preference at any time should be coherent to the preorder interests and towards the final goal in the interest sequence. From the view of conversation, a goal-aware CRS dialog is a smoothly transited sequence of utterances guided by the sequence of user interest. Therefore, modeling the goal-aware sequence of user interests is essential for goal-aware CRS. For example, in the "Dialog" of Figure 1, there are similarities in storyline or other aspects between neighbor entities in the interest sequence, i.e., "*Shutter Island*" → "*Inception*" → "*Leonardo*" → "*Source Code*", and between each interest entity and the recommendation goal "*The Butterfly Effect*".

For modeling the goal-aware interest sequence, KG-based CRS methods are widely studied using knowledge graph (KG) (Zhou et al., 2020a; Lu et al., 2021; Zhou et al., 2022) to track the interest sequence in the dialog (Zhou et al., 2021; Ma et al., 2021). Although KG's explicit logical links between interest entities greatly facilitate the modeling of interest sequence, it also leads to over-rely on KG knowledge and weakens the key role of dialogue behavior in CRS. Specifically, we note two major consequent issues: (1) From the view of recommendation, most KG-based CRS model the user preference only on the explicit separate interest links in KG and ignore the rich implicit interests sequence in dialogue which is absent in KG knowledge. For example, in the "Dialog" of Figure 1, each interest entity may have implicit semantic relations with the goal "*The Butterfly Effect*" beyond KG links. Furthermore, the goal of the sequence is also absent in the user preference modeling. (2)

*Corresponding author.

Proceedings of the 29th International Conference on Computational Linguistics, pages 400–411
October 12–17, 2022.

From the view of conversation, current KG-based CRS mainly use entities in the interest sequence to enhance the semantic generation of every single response instead of using the entire sequence to enhance the smooth transition of multiple utterances towards the recommendation goal.

To address these two issues, we propose **CR-GIS**, which jointly improves the **C**onversation and **R**ecommendation by modeling **G**oal-aware **I**nterest **S**equence in CRS. To this end, we design a novel parallel star structure with two advantages: (1) For the first issue of implicit relations modeling in goal-aware interests sequence, on the base of pre-encoded explicit knowledge from KG, we propose an interest-level star graph to encode the implicit relations between interest entities in dialog. The satellite nodes of the star graph are interest entities in the current ongoing dialog, which are sequenced by their adjacency relationship in the dialog. The key advantage of interest-level star graph is that the central star node, which connects all satellite nodes, can be fused to the recommendation goal with the Mutual Information Maximization (MIM) method. This alignment makes the interest sequence modeling goal-oriented. (2) For the second issue of smooth transition in goal-aware conversation, to be paralleled to the interest-level star graph, we design a hierarchical Star Transformer encoder (Guo et al., 2019) for words and utterances. By aligning the utterance state representation of the "utterance-level star" with the "interest-level star", we enhance the CRS to generate smoothly transited utterances towards the recommendation goal.

Our contributions are summarized as follows:

(1) To sufficiently model the goal-aware user interest sequence in CRS, we propose an interest-level star graph to model the implicit sequence of interest entities in dialog and make the interest sequence modeling be aware of the recommendation goal with a goal-oriented fusion mechanism.

(2) To effectively generate the smoothly transited responses, we align an utterance-level star transformer to the interest-level star so as to make the responses generation follow the interest sequence and be aware of the recommendation goal.

(3) Extensive experiments conducted on OpenDialKG (Moon et al., 2019) and TG-ReDial (Zhou et al., 2020b) datasets demonstrate that our model outperforms the SOTA baseline models in successfully reaching recommendation goals through smooth transited utterances.

2 Related Work

Current CRS can be divided into two types. One is the recommendation-based CRS which aims to ask users questions about interests over pre-define slots (Sun and Zhang, 2018; Lei et al., 2020; Zou et al., 2020; Deng et al., 2021; Zhang et al., 2021; Kostric et al., 2021) and make responses considering users' feedback (Luo et al., 2020; Xu et al., 2021). Recommendation-based CRS mainly suffers from the inflexibility of pre-defined templates.

The other is the generation-based CRS (Li et al., 2018; Hayati et al., 2020) which understands user preferences (Moon et al., 2019; Zhou et al., 2020b; Lu et al., 2021; Zhou et al., 2022) and generates human-like responses (Liao et al., 2019; Liang et al., 2021) in line with user interests. Closely related to our work, Chen et al. (2019) and Zhou et al. (2020a) integrate KG knowledge to understand users' interest. However, they simply aggregate entities of KG in the utterance, instead of exploiting the implicit interest sequence in the conversation. Zhou et al. (2021) and Ma et al. (2021) adopt reasoning-based methods to predict the shift direction of user interest, but also limits to the explicit interest sequence in KG.

3 Problem Formalization

A KG $\mathcal{G}$ with entity set $\mathcal{E}$ and relation set $\mathcal{R}$ is $\mathcal{G} = \{(e, r, e') \mid e, e' \in \mathcal{E}, r \in \mathcal{R}\}$ where (e, r, e') is a relation r from the entity e to the entity e'. Suppose we have a CRS corpus D and a KG $\mathcal{G}$ parallel to D, in which the interest entities mentioned in D are linked to the entities in $\mathcal{G}$. $U = \{u_1, u_2, \ldots, u_n\}$ is the conversation history, where $u_i = \{w_{i,1}, w_{i,2}, \ldots, w_{i,m}\}$ is the word token sequence in the i-th utterance. $S = \{s_1, s_2, \ldots, s_k\}$ is the interest entity in each utterance of U, and $s_i \in S$ is linked to $\mathcal{G}$, i.e., $s_i \in \mathcal{E}$. In a response $Y = \{y_1, y_2, \ldots, y_m\}$, a recommendation goal entity set $G = \{g_1, g_2, \ldots, g_t\}$ is identified in advance, where $g_i \in G$ is an entity linking to $\mathcal{G}$, i.e., $g_i \in \mathcal{E}$. n,m,k,t represent the length of the historical utterance sequence, the length of the token sequence, the length of the interest entity sequence and the number of recommendation goals, respectively. Our task is to learn a recommendation model and a response generation model $P(Y|U, S, G)$ with the D and $\mathcal{G}$. The former model the implicit user interests sequence through S and G and help the latter to generate responses that smoothly progress to the recommendation goal G.

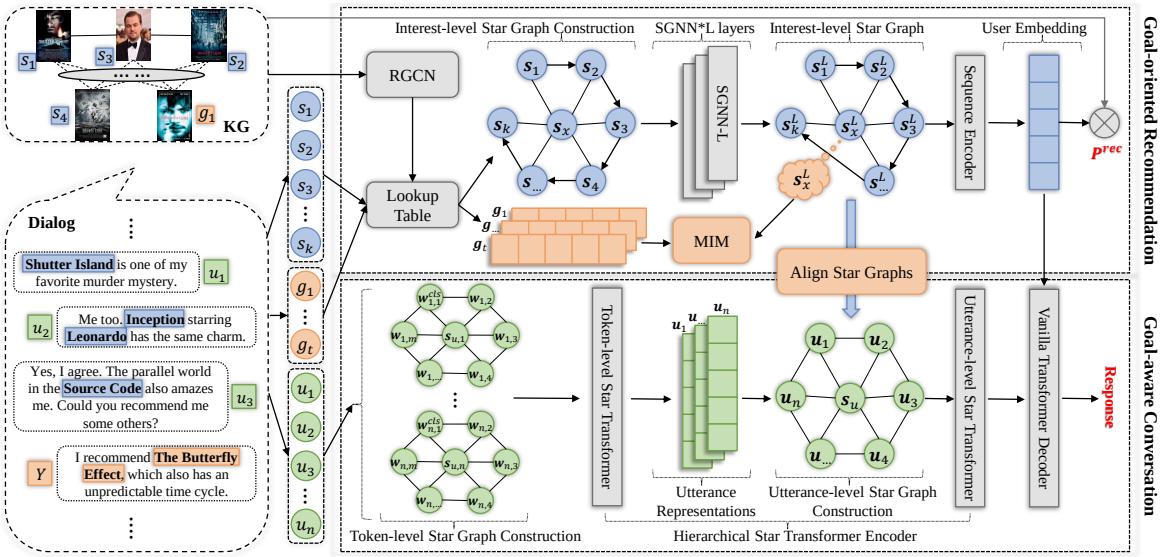

Figure 1: The architecture of the proposed CR-GIS model.

4 Approach

4.1 Architecture Overview

As shown in Figure 1, proposed CR-GIS contains five parts: (1) Explicit-Implicit relations encoder. The explicit KG relations encoder adopts R-GCN to learn the KG-based representation of entities. The interest entities sequence in the conversation is constructed into an interest-level star graph, in which the implicit user interest sequence is learned by the implicit dialog relations encoder employing a star graph neural network (SGNN). (2) The goal-oriented fusion module adopts mutual information maximization (MIM) to bridge the goal G and the star node of SGNN, which makes the user interest sequence modeling goal-aware. (3) The goal-aware recommendation module obtains the user's interests representation with a sequence encoder mining the interest sequence. (4) The goal-aware response generation module uses hierarchical Star Transformer to encode the token-level and utterance-level star graph from conversation history. (5) We align the interest-level star graph to the utterance-level star graph to improve the goal-aware transition of multi-turn response generation.

4.2 Explicit-Implicit Relations Encoder

Explicit KG Relations Encoder We use R-GCN (Schlichtkrull et al., 2018) to encode explicit KG relations and get the entity embedding matrix $\boldsymbol{E}$.

Implicit Dialog Relations Encoder With embedding matrix $\boldsymbol{E}$ of KG entities, we adopt star graph neural network (SGNN) (Pan et al., 2020) to encode the implicit semantic links between the entities in the interest sequence in dialog. Given the interest sequence $S = \{s_1, s_2, \ldots, s_k\}$ in dialog context, we construct an interest-level star graph with one star node s_x and each s_i is a satellite node. The adjacency relationship in the dialog between the satellite nodes $\{(s_i, s_{i+1}) \mid s_i, s_{i+1} \in S\}$ is maintained. Star node s_x is linked to each satellite s_i.

SGNN adopts a cyclic updating between s_x and each s_i. s_i is initialized by embedding matrix $\boldsymbol{E}$, and s_x is initialized as the average value of all s_i. The updated representation $\hat{\boldsymbol{s}}_i^{(l+1)}$ of s_i at the $(l+1)$-th layer of the SGNN is calculated as:

$$
\begin{aligned}
\boldsymbol{z}_i^{(l+1)} &= \sigma\left(\boldsymbol{W}_{z,1}\boldsymbol{a}_i^{(l+1)} + \boldsymbol{W}_{z,2}\boldsymbol{s}_i^{(l)}\right), \\
\boldsymbol{v}_i^{(l+1)} &= \sigma\left(\boldsymbol{W}_{v,1}\boldsymbol{a}_i^{(l+1)} + \boldsymbol{W}_{v,2}\boldsymbol{s}_i^{(l)}\right), \\
\tilde{\boldsymbol{s}}_i^{(l+1)} &= \rho\left(\boldsymbol{W}_{s,1}\boldsymbol{a}_i^{(l+1)} + \boldsymbol{W}_{s,2}\left(\boldsymbol{v}_i^{(l+1)} \odot \boldsymbol{s}_i^{(l)}\right)\right), \\
\hat{\boldsymbol{s}}_i^{(l+1)} &= \left(1 - \boldsymbol{z}_i^{(l+1)}\right) \odot \boldsymbol{s}_i^{(l)} + \boldsymbol{z}_i^{(l+1)} \odot \tilde{\boldsymbol{s}}_i^{(l+1)}, \\
\boldsymbol{a}_i^{(l+1)} &= \left[\boldsymbol{A}_i^I\left(\left(\left[\boldsymbol{s}_1^{(l)}; \ldots; \boldsymbol{s}_k^{(l)}\right]\right)^T \boldsymbol{W}_a^I + \boldsymbol{b}_a^I\right); \right. \\
&\quad \left. \boldsymbol{A}_i^O\left(\left(\left[\boldsymbol{s}_1^{(l)}; \ldots; \boldsymbol{s}_k^{(l)}\right]\right)^T \boldsymbol{W}_a^O + \boldsymbol{b}_a^O\right)\right], \qquad (1)
\end{aligned}
$$

where $\boldsymbol{W}_{z,1}, \boldsymbol{W}_{v,1}, \boldsymbol{W}_{s,1} \in \mathbb{R}^{d_e \times 2d_e}$ and $\boldsymbol{W}_{z,2}, \boldsymbol{W}_{v,2}, \boldsymbol{W}_{s,2} \in \mathbb{R}^{d_e \times d_e}$ are the learnable matrix. $\rho(\cdot)$ is the tanh function. $\sigma(\cdot)$ is the sigmoid function. $\odot$ is the Hadamard product. $\boldsymbol{s}_i^{(l)}$ is the representation of s_i at the l-th layer of the SGNN. $\boldsymbol{a}_i^{(l+1)}$ is the information propagated by the adjacent nodes of s_i on the interest-level star graph

through the incoming matrix $\boldsymbol{A}^I$ and the outgoing matrix $\boldsymbol{A}^O$ proposed by GGNN (Li et al., 2016). $[;]$ is the concatenation operation. $\boldsymbol{A}_i^I, \boldsymbol{A}_i^O \in \mathbb{R}^{1\times k}$ are the weights of the i-th row in the $\boldsymbol{A}^I$ and $\boldsymbol{A}^O$, respectively. $\boldsymbol{W}_a^I, \boldsymbol{W}_a^O \in \mathbb{R}^{d_e\times d_e}$ are the learnable matrix. $\boldsymbol{b}_a^I, \boldsymbol{b}_a^O \in \mathbb{R}^{d_e}$ are the bias vector. The information injected from the s_x is controlled by the self-attention mechanism to calculate the similarity $\gamma_i^{(l+1)}$ between each s_i and s_x. Furthermore, the representation of s_i at the $(l+1)$-th layer is $\boldsymbol{s}_i^{(l+1)} = \left(1-\gamma_i^{(l+1)}\right)\hat{\boldsymbol{s}}_i^{(l+1)} + \gamma_i^{(l+1)}\boldsymbol{s}_x^{(l)}$. The representation $\boldsymbol{s}_x^{(l+1)}$ of s_x at the $(l+1)$-th layer is obtained by aggregating all s_i at the $(l+1)$-th layer with attention mechanism. In this way, s_x represents the information of entire interest sequence.

4.3 Goal-oriented Fusion Mechanism

The goal-oriented fusion mechanism is motivated by the intuition that the user interest sequence is aware of the recommendation goals in the conversation. Therefore, to connect the interest-level star graph with the the goals, we bridge the star node of SGNN and the goals with the Mutual Information Maximization (Hjelm et al., 2019). Specifically, given the set of goal entities $G = \{g_1, g_2, \ldots, g_t\}$ and the representation $\boldsymbol{s}_x^{(l+1)}$ of the star node at the $(l+1)$-th layer, we design a loss function by the contrastive learning that maximizes the mutual information between the goal and the star node:

$$L_{MIM}(s_x, g_i) = \mathbb{E}_{g_i\in G}\big[f(s_x, g_i) - \log\sum_{\tilde{g}_j\in\tilde{G}}\exp\left(f(s_x, \tilde{g}_j)\right)\big], \quad (2)$$

where $\tilde{g}_j \in \tilde{G}$ is the negative node we randomly sampled from KG. $f(\cdot,\cdot)$ is a scoring function implemented with a bilinear mapping network:

$$f(s_x, g_i) = \sigma\left(\left(\boldsymbol{s}_x^{(l+1)}\right)^T \cdot \boldsymbol{W}_{MIM}\cdot \boldsymbol{g}_i\right), \quad (3)$$

where $\boldsymbol{W}_{MIM} \in \mathbb{R}^{d_e\times d_e}$ is the learnable matrix and $\sigma(\cdot)$ is the sigmoid function. $\boldsymbol{g}_i$ is the embedding of g_i by looking up from $\boldsymbol{E}$. Here, we adopt a pre-training way to enhance the goal aware ability of the sequence by minimizing the loss L_{MIM}.

4.4 Goal-oriented Recommendation

After the multi-dimensional information is fused into the entity representation, in the output of the L-layer SGNN, we obtain the star node embedding vector $\boldsymbol{s}_x^L$ and the embedding matrix $\boldsymbol{S}^L = \left[\boldsymbol{s}_1^L; \boldsymbol{s}_2^L; \ldots; \boldsymbol{s}_k^L\right]$ composed of entities in the interest sequence, i.e., satellite nodes. To model sequence information, we integrate a learnable position embedding matrix $\boldsymbol{P} \in \mathbb{R}^{k\times d_e}$ to enhance the sequence representation of interest entities. The sequence representation $\boldsymbol{S}_P^L$ is obtained by summing two embedding metrices: $\boldsymbol{S}_P^L = \boldsymbol{S}^L + \boldsymbol{P}$. In addition, the star node, as an intermediary between the sequence and the goal, contains personalized information of the goal. Therefore, we combine the position-enhanced interest sequence $\boldsymbol{S}_P^L$ and the star node $\boldsymbol{s}_x^L$ as the goal-enhanced sequence representation $\boldsymbol{S}_{P,x}^L = \left[\boldsymbol{S}_P^L; s_x^L\right]$.

To better capture the preference expressed in the interest sequence, we propose a sequence encoder, which is composed of a multi-head self-attention layer (*MHA*) and a point-wise feed-forward network. The *MHA* extracts the information of different representation subspaces, which is defined as:

$$\begin{aligned}\boldsymbol{S}_F &= \mathrm{MHA}(\boldsymbol{F}) = [\mathrm{head}_1; \ldots; \mathrm{head}_h]\,\boldsymbol{W}^O,\\ \mathrm{head}_i &= \mathrm{Attention}\left(\boldsymbol{F}\boldsymbol{W}_i^Q, \boldsymbol{F}\boldsymbol{W}_i^K, \boldsymbol{F}\boldsymbol{W}_i^V\right),\\ \mathrm{Attention}(\boldsymbol{Q},\boldsymbol{K},\boldsymbol{V}) &= \mathrm{softmax}\left(\frac{\boldsymbol{Q}\boldsymbol{K}^T}{\sqrt{d_e/h}}\right)\boldsymbol{V},\end{aligned} \quad (4)$$

where $\boldsymbol{F} = \boldsymbol{S}_{P,x}^L$ is the input of MHA. The projection matrix $\boldsymbol{W}_i^Q \in \mathbb{R}^{d_e\times d_h}, \boldsymbol{W}_i^K \in \mathbb{R}^{d_e\times d_h}, \boldsymbol{W}_i^V \in \mathbb{R}^{d_e\times d_h}$ and $\boldsymbol{W}^O \in \mathbb{R}^{d_e\times d_e}$ are the learnable parameters for each attention head. $d_h = d_e/h$ is the dimension of attention heads. $\boldsymbol{Q} = \boldsymbol{F}\boldsymbol{W}_i^Q, \boldsymbol{K} = \boldsymbol{F}\boldsymbol{W}_i^K$ and $\boldsymbol{V} = \boldsymbol{F}\boldsymbol{W}_i^V$ are query, key and value, respectively. $\sqrt{d_e/h}$ is the scale factor to avoid large inner product values.

Towards a nonlinear sequence encoder, we use a point-wise feed-forward network (FFN):

$$\begin{aligned}\boldsymbol{F} &= \left[\mathrm{FFN}\left(\boldsymbol{S}_{F,1}\right)^T; \ldots; \mathrm{FFN}\left(\boldsymbol{S}_{F,k}\right)^T\right],\\ \mathrm{FFN}(\boldsymbol{x}) &= \max\left(0, \boldsymbol{x}\boldsymbol{W}_{F,1} + \boldsymbol{b}_{F,1}\right)\boldsymbol{W}_{F,2} + \boldsymbol{b}_{F,2},\end{aligned} \quad (5)$$

where $\boldsymbol{W}_{F,1}, \boldsymbol{W}_{F,2}, \boldsymbol{b}_{F,1}, \boldsymbol{b}_{F,2}$ are trainable parameters. Note that the sequence encoder, i.e., MHA layer and FFN layer, can be multiply stacked. We take the first embedding vector of the matrix output by the sequence encoder as the user's preference representation $\boldsymbol{p}_u$ in current context, i.e., $\boldsymbol{p}_u = \boldsymbol{F}_1$.

Given the learned user preference, we calculate the probability of recommending an item: $P_i^{rec} = \mathrm{softmax}\left(\boldsymbol{p}_u^T \cdot e_i\right)$, where $\boldsymbol{e}_i$ is the learned item embedding looking up from $\boldsymbol{E}$. To train the recommendation module, we use cross-entropy as the optimization objective:

$$L_{REC} = -\sum_{i=1}^{M} \left[P_i \cdot \log\left(P_i^{rec}\right) - \right.$$
$$\left. (1 - P_i) \cdot \log\left(1 - P_i^{rec}\right) \right] + \alpha * L_{MIM}, \quad (6)$$

where i is the item index. α is a hyperparameter representing the weight of MIM loss.

4.5 Goal-aware Response Generation

Parallel to the interest-level star graph, we design a hierarchical Star Transformer to encode the dialog context by constructing token-level and utterance-level star graphs to capture the sequential semantic dependency between utterances. Injecting the interest entities in the goal-aware interest-level star graph into the utterance-level star graph, we promote the goal-aware ability of response generation.

Specifically, the topology of Star Transformer (Guo et al., 2019) is the same as SGNN, which is composed of a relay node (as the star node in SGNN) and n satellite nodes. Given a dialogue context $U = \{u_1, u_2, \ldots, u_n\}$ with n utterances, where $u_i = \{w_{i,1}, w_{i,2}, \ldots, w_{i,m}\}$ is the word token sequence of the i-th utterance. The token-level Star Transformer encoder is a token-level star graph which treats each word token as a satellite node, and the relay node acts as a virtual hub to gather and scatter information from and to the satellite nodes. It adopts a cyclic updating, in which the satellite node $h_{i,j}$ is initialized by word embedding $w_{i,j}$, i.e., $\boldsymbol{h}_{i,j}^{(0)} = \boldsymbol{w}_{i,j}$, and the relay node $s_{u,i}$ is initialized as the average of all tokens, i.e., $\boldsymbol{s}_{u,i}^{(0)} = \frac{1}{m}\sum_{j=1}^{m} \boldsymbol{h}_{i,j}^{(0)}$. Each token-level satellite node is updated at step t according to its adjacent nodes, including neighbor nodes $\boldsymbol{h}_{i,j-1}^{(t-1)}, \boldsymbol{h}_{i,j+1}^{(t-1)}$ in the text sequence, its previous state $\boldsymbol{h}_{i,j}^{(t-1)}$, the state of the relay node in the previous round $\boldsymbol{s}_{u,i}^{(t-1)}$, and its token embedding $\boldsymbol{w}_{i,j}$. Formally:

$$\boldsymbol{C}_{i,j}^{(t)} = \left[\boldsymbol{h}_{i,j-1}^{(t-1)}; \boldsymbol{h}_{i,j}^{(t-1)}; \boldsymbol{h}_{i,j+1}^{(t-1)}; \boldsymbol{s}_{u,i}^{(t-1)}; \boldsymbol{w}_{i,j}\right],$$
$$\boldsymbol{h}_{i,j}^{(t)} = \mathrm{MHA}\left(\boldsymbol{h}_{i,j}^{(t-1)}, \boldsymbol{C}_{i,j}^{(t)}, \boldsymbol{C}_{i,j}^{(t)}\right), \quad (7)$$

where $\boldsymbol{C}_{i,j}^{(t)}$ is the context of the j-th satellite node. For MHA, $\boldsymbol{h}_{i,j}^{(t-1)}$ is query, $\boldsymbol{C}_{i,j}^{(t)}$ is key and value.

The token-level relay node is updated by fusing all satellite nodes and its previous state $\boldsymbol{s}_{u,i}^{(t-1)}$.

$$\boldsymbol{s}_{u,i}^{(t)} = \mathrm{MHA}\left(\boldsymbol{s}_{u,i}^{(t-1)}, \left[\boldsymbol{s}_{u,i}^{(t-1)}; \boldsymbol{H}_i^{(t)}\right], \left[\boldsymbol{s}_{u,i}^{(t-1)}; \boldsymbol{H}_i^{(t)}\right]\right), \quad (8)$$

where $\boldsymbol{H}_i^{(t)} = \left[\boldsymbol{h}_{i,1}^{(t)}; \boldsymbol{h}_{i,2}^{(t)}; \ldots; \boldsymbol{h}_{i,m}^{(t)}\right]$. After T-step update, we merge the information of the relay node into the token-level satellite node, and obtain the hidden vector sequence of utterance u_i using $\psi(\cdot)$ which is a MHA layer with an FFN layer:

$$\left[\hat{\boldsymbol{h}}_{i,1}, \hat{\boldsymbol{h}}_{i,2}, \ldots, \hat{\boldsymbol{h}}_{i,m}\right] = \psi\left(\varphi\left(\left[\boldsymbol{h}_{i,1}^{(T)}; \boldsymbol{h}_{i,2}^{(T)}; \ldots; \boldsymbol{h}_{i,m}^{(T)}\right], \boldsymbol{s}_{u,i}^{(T)}\right)\right), \quad (9)$$

where $\varphi(\cdot)$ is an MLP layer. The token corresponding to the first hidden state $\hat{\boldsymbol{h}}_{i,1}$ of the hidden vector sequence is a special token $[CLS]$ used to aggregate the sequence representation and is inspired by Devlin et al. (2019). Therefore, we collect utterance representations derived from $[CLS]$, i.e., the representation of utterance $\boldsymbol{u}_i = \hat{\boldsymbol{h}}_{i,1}$, and input them into utterance-level Star Transformer encoder.

The utterance-level Star Transformer encoder constructs an utterance-level star graph using utterances as satellite nodes h_i, which is initialized by the representation of utterance, i.e., $\boldsymbol{h}_i^{(0)} = \boldsymbol{u}_i$. The relay node s_u is initialized as the average of satellite nodes, i.e., $\boldsymbol{s}_u^{(0)} = \frac{1}{n}\sum_{i=1}^{n} \boldsymbol{h}_i^{(0)}$. For the update of each utterance-level satellite node at step t, in addition to the information involved in the token-level node, we also inject the goal-aware interest entities information in the interest-level star graph into the updated representation of the corresponding utterance according to the "mentioned" relationship with the utterance:

$$\boldsymbol{C}_i^{(t)} = \left[\boldsymbol{h}_{i-1}^{(t-1)}; \boldsymbol{h}_i^{(t-1)}; \boldsymbol{h}_{i+1}^{(t-1)}; \boldsymbol{s}_u^{(t-1)}; \boldsymbol{u}_i; \boldsymbol{O}_i\right],$$
$$\boldsymbol{h}_i^{(t)} = \mathrm{MHA}\left(\boldsymbol{h}_i^{(t-1)}, \boldsymbol{C}_i^{(t)}, \boldsymbol{C}_i^{(t)}\right), \quad (10)$$

where $\boldsymbol{O}_i = \mathrm{MLP}\left([\boldsymbol{e}_{i,1}, \boldsymbol{e}_{i,2}, \ldots, \boldsymbol{e}_{i,j}]\right)$ is the embedding matrix composed of interest entities $e_{i,j}$ mentioned in the utterance u_i. Interest entity embedding $\boldsymbol{e}_{i,j}$ is obtained from the satellite node embedding matrix $\boldsymbol{S}^L$ output by the L-layer SGNN.

The goal-aware star node $\boldsymbol{s}_x^L$ from the interest-level star graph also affects the update of the utterance-level relay node:

$$\boldsymbol{s}_u^{(t)} = \mathrm{MHA}\left(\boldsymbol{s}_u^{(t-1)}, \left[\boldsymbol{s}_u^{(t-1)}; \boldsymbol{o}_x^L; \boldsymbol{H}^{(t)}\right], \left[\boldsymbol{s}_u^{(t-1)}; \boldsymbol{o}_x^L; \boldsymbol{H}^{(t)}\right]\right), \quad (11)$$

where $\boldsymbol{o}_x^L = \mathrm{MLP}\left(\boldsymbol{s}_x^L\right)$, and $\boldsymbol{H}^{(t)} = \left[\boldsymbol{h}_1^{(t)}; \boldsymbol{h}_2^{(t)}; \ldots; \boldsymbol{h}_n^{(t)}\right]$. After the T-step cyclic update, we further use an MLP layer with LayerNorm

Datasets		OpenDialKG	TG-ReDial
Corpus Info.	#Domains	Movie,Book	Movie
	#Dialogues	13,802	10,000
	#Utterances	126,104	129,392
	#Avg. Words	12.7	19.0
	#Split Ratio	7:1.5:1.5	8:1:1
KG Info.	#Entities	100,813	62,348
	#Relations	1,358	60
	#Triplets	1,190,658	802,578

Table 1: Statistics of datasets after preprocessing.

to fuse the relay node information into the representation of the utterance-level satellite nodes to obtain the enhanced goal-aware utterance representation $\boldsymbol{H}_u$, and take it as the initial decoding state: $\boldsymbol{H}_u = \varphi\left(\left[\boldsymbol{h}_1^{(T)}; \boldsymbol{h}_2^{(T)}; \ldots; \boldsymbol{h}_n^{(T)}\right], \boldsymbol{s}_u^{(T)}\right)$

In the decoding stage, we adopt the decoder framework of the vanilla Transformer. In order to further enhance the goal-aware ability of response generation and be in line with the user's current interest, inspired by Zhou et al. (2021), we incorporate user-preferred word bias in the output of the self-attention sub-layer of the decoder's i-th layer: $\boldsymbol{R}_{i-1} = \boldsymbol{R}_{i-1} + \eta\left(\boldsymbol{p}_u\right)$, where $\boldsymbol{R}_{i-1}$ is the input of the decoder at the i-th layer. $\eta(\cdot) : \mathbb{R}^{d_e} \rightarrow \mathbb{R}^{d_w}$, and d_w is the dimension of the hidden layer. To learn the response generation module, we set the negative log-likelihood loss as:

$$L_{GEN} = -\frac{1}{m}\sum_{t=1}^{m}\left[\log\left(P_{gen}\left(y_t \mid y_{1:t-1}, U, S, G\right)\right)\right] + \beta * L_{REC}, \quad (12)$$

where β is a hyperparameter that represents the weight of the recommendation loss.

5 Experiments

5.1 Experiment Setup

Datasets We conduct experiments on two CRS datasets. (1) OpenDialKG (Moon et al., 2019) is a parallel Dialog ↔ KG CRS corpus where the interest entities mentioned in the dialog are linked to KG. (2) TG-ReDial (Zhou et al., 2020b) is a topic-guided CRS corpus, in which each dialog is associated with a topic thread. The movies mentioned in the corpus are linked to a KG: CN-DBpedia (Xu et al., 2017). To make full use of the annotated topic, we add each topic as an entity to CN-DBpedia to obtain a topic-enhanced KG. We add new edges between the movie and the topic entities based on their real relationship in the Chinese movie review website Douban, i.e., whether a topic appears in the comments or tags of a movie. The statistics after preprocessing are in Table 1.

Models	OpenDialKG			TG-ReDial		
	R@1	R@10	R@25	R@1	R@10	R@25
TextCNN	0.058	0.176	0.236	0.178	1.034	1.872
KBRD	0.107	0.409	0.494	0.337	3.119	6.073
KGSF	0.120	0.433	0.522	0.419	3.761	6.925
RevCore	0.124	0.428	0.515	0.490	3.494	6.631
CRFR	0.130	0.453	0.538	0.646	4.189	7.509
C^2-CRS	0.112	0.458	0.535	0.935	4.401	7.991
CR-GIS	**0.182***	**0.473***	**0.547***	**1.211***	**6.202***	**10.469***
w/o GoInfo.	0.157	0.463	0.533	0.958	5.937	9.849
w/o ImpEnc.	0.129	0.401	0.487	0.709	4.185	7.476
w/o SeqEnc.	0.145	0.453	0.529	0.891	5.295	8.909

Table 2: Overall recommendation evaluation. w/o refers to removing CR-GIS components. "$*$" indicates the statistical significance for $p < 0.005$ compared with the best baseline. TG-ReDial results are reported in percentage.

Baselines We compare our CR-GIS with the following competitive models: (1) *TextCNN* (Kim, 2014) is a recommendation model extracting user preference from utterances with a CNN-based model. (2) *Transformer* (Vaswani et al., 2017) is a vanilla Transformer-based dialog generation model. (3) *KBRD* (Chen et al., 2019) is a Knowledge-Based CRS integrating item-oriented KG. (4) *KGSF* (Zhou et al., 2020a) is a KG-based Semantic Fusion CRS aligning the semantic space of two KGs. (5) *RevCore* (Lu et al., 2021) is a review-enhanced CRS. (6) *CRFR* (Zhou et al., 2021) is a Fragments Reasoning-based CRS focusing on multi-hop reasoning on KGs. (7) *C^2-CRS* (Zhou et al., 2022) is a CRS adopting coarse-to-fine contrastive learning. For a fair comparison, all KG-based CRS models share the same KG.

Implementation Details Our model is implemented with Pytorch. The dimensionality of KG embedding d_e and word embedding d_w are set to 128 and 300. The layers of R-GCN, SGNN and Star Transformer encoder are set to 1, 6 and 2. We use the Adam optimizer (Kingma and Ba, 2015), the batch size is 32, the learning rate is 0.001, and gradient clipping restricts the gradients within [0, 0.1]. The whole training process is split into three steps. First, we minimize the L_{MIM} loss for pretraining the goal-oriented information fusion module. After that, we minimize the L_{REC} loss with the weight α of 0.1. Finally, we minimize the L_{GEN} loss with the weight β of 0.5 on OpenDialKG dataset and 0 on TG-Redial dataset.

Models	OpenDialKG					TG-ReDial				
	Bleu-1	Bleu-2	Dist-1	Dist-2	HIT	Bleu-1	Bleu-2	Dist-1	Dist-2	HIT
Transformer	0.389	0.310	0.027	0.113	0.126	0.504	0.386	0.011	0.055	8.868
KBRD	0.404	0.318	0.055	0.164	0.251	0.506	0.395	0.025	0.082	9.105
KGSF	0.412	0.326	0.061	0.202	0.262	0.509	0.396	0.029	0.100	9.158
RevCore	0.406	0.321	0.054	0.193	0.240	0.511	0.403	0.020	0.079	9.002
CRFR	0.418	0.331	0.062	0.204	0.288	0.515	0.401	0.031	0.103	9.225
C^2-CRS	0.414	0.328	0.064	0.210	0.308	0.516	0.402	0.031	0.100	9.260
CR-GIS	**0.432***	**0.342***	**0.066**	0.207	**0.332***	**0.529***	**0.406***	**0.033***	0.107*	**9.537***
w/o GoInfo.	0.426	0.336	0.060	0.193	0.324	0.516	0.401	0.029	0.094	9.153
w/o Align.	0.428	0.336	0.059	0.191	0.314	0.518	0.402	0.030	0.095	9.447
w/o HiStar.	0.409	0.324	0.061	**0.220**	0.305	0.519	0.400	0.033	**0.111**	9.235

Table 3: Overall conversation evaluation. w/o refers to removing CR-GIS components. "$*$" indicates the statistical significance for $p < 0.05$ compared with the best baseline. The HIT of TG-ReDial are reported in percentage.

5.2 Evaluation on Recommendation

Overall Performance As shown in Table 2, we adopt the recognized Recall@K (K=1,10,25) metrics to evaluate the recommendation performance. CR-GIS significantly outperforms all baselines over all metrics. Specifically, compared with the best results of the baselines, CR-GIS improves the Recall@K (K=1,10,25) metrics by about 40%, 2.6%, 1.7%, and 29.5%, 40.9%, 31% on the OpenDialKG and TG-ReDial datasets, respectively. Although introducing more external knowledge (i.e., KBRD, KGSF, RevCore, C^2-CRS) and multi-hop KG reasoning (i.e., CRFR) has achieved staged success, the impressive performance of CR-GIS shows that it is necessary to model the goal-aware implicit user interest sequence in dialog.

Ablation Study In Table 2, we removed the key components in the recommendation module of CR-GIS for ablation study. **First**, we remove the goal-oriented information fusion mechanism, called "w/o GoInfo.". The Recall@K (K=1,10,25) results decrease slightly on OpenDialKG, but decrease significantly on TG-ReDial. It indicates the advantage of "goal-aware" recommendation, i.e., modeling the implicit relationship between the interest sequence and the goal, especially when there is a dialog goal guiding the conversation, e.g., TG-ReDial dataset. **Second**, we remove the implict dialog relations encoder, called "w/o ImpEnc.". Note that "GoInfo." is also removed due to there is no star node. We observe that the recommendation results are significantly decreased over all metrics. It confirms that to capture the association between interest entities in long-range conversation is a key factor in improving recommendation performance. Note that we didn't ablate the explicit KG relations encoder which is a shared KG encoder for all CRS baselines. **Third**, we remove the sequence encoder, called "w/o SeqEnc.", which is substituted by self-attention mechanism. The recommendation results distinctly decrease indicating that the modeling of user interests sequence is also crucial for obtaining the dynamic preferences in dialog.

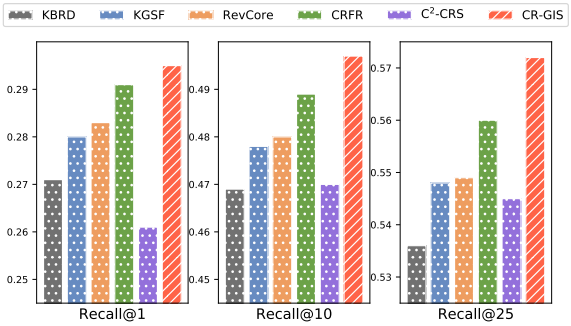

Figure 2: Analysis of CR-GIS's scalability on TG-ReDial, where topic is also treated as the recommendation goal.

The Scalability of CR-GIS To analyze the scalability of CR-GIS with constructed topic-enhanced KG, we consider the topic mentioned in the utterance as the recommendation goal to further examine the total recommendation performance on TG-ReDial. As shown in Figure 2, although the absolute value of the performance of all models has been significantly improved, CR-GIS still significantly outperforms all baselines. It further confirms that capturing the implicit association between entities in the interest flow pointing to the goal in the conversation can make up for the lack of information propagation on the KG.

5.3 Evaluation on Conversation

Overall Performance To automatically evaluate the conversation performance, we adopt Bleu-1/2,

Models	Flu.	Coher.	Info.	Proact.
Transformer	1.568	1.230	1.043	0.320
KBRD	1.611	1.248	1.192	0.347
KGSF	1.635	1.287	1.143	0.362
RevCore	1.654	1.252	1.215	0.445
CRFR	1.716	1.255	1.280	0.447
C^2-CRS	1.585	1.249	1.138	0.426
CR-GIS	**1.763***	**1.313***	**1.292***	**0.450***
kappa	0.534	0.509	0.505	0.516

Table 4: Human evaluation. "Flu.", "Coher.", "Info.", "Proact." respectively denote fluency, coherence, informativeness and proactivity. The agreement ratio $kappa \in [0.41, 0.6]$ denotes the moderate agreement. "$*$" indicates the statistical significance for $p < 0.05$ compared with the best baseline.

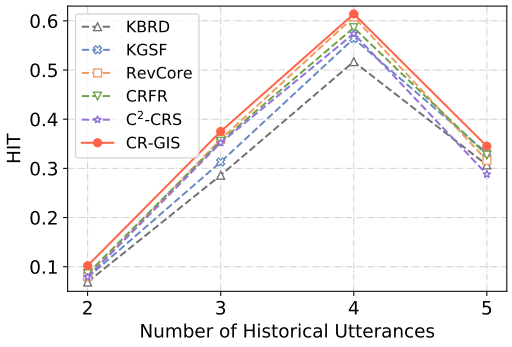

Figure 3: As the number of historical utterances increases, CR-GIS outperforms all baselines in terms of hitting entity and topic in generated response.

Distinct-1/2 and HIT. The Bleu and Distinct evaluate the fluency and diversity of responses, respectively. HIT evaluates the ratio of the goals in the golden response that is also hit by the generated response. In Table 3, CR-GIS exceeds all baselines. Specifically, compared with the best results of C^2-CRS, CR-GIS improves HIT by 7.8% and 3% on two datasets, respectively, which indicates that CR-GIS effectively perceives the goal, thus generating responses more in line with user interests. This is attributed to the goal-oriented fusion mechanism that enhances the goal expression in the interest-level star graph, and then the goal-aware interest-level star graph is aligned with the utterance-level star graph to finally endow CR-GIS's goal-aware ability of response generation. In addition, CR-GIS achieves 4.3%, 4.3% and 2.5%, 1% improvements on Bleu-1 and Bleu-2 compared with the best baselines on two datasets, respectively, which also demonstrates that CR-GIS prefers to capture more useful information in n-gram's level leading to more fluent responses.

Human Evaluation For human evaluation, we randomly sampled 200 context-response pairs per model from OpenDialKG. Five well-educated annotators evaluate each response with four indicators: fluency, coherence, informativeness and proactivity. Proactivity evaluates whether the dialog agent makes proactive actions such as recommending or asking user preference. The scores are settled from $\{0, 1, 2\}$ to estimate fluency, coherence, informativeness. The proactivity scores are assigned from $\{0, 1\}$. The agreement among the annotators is measured by the Fleiss' kappa (Fleiss, 1971). As the results in Table 4, the superior of CR-GIS on all indicators support the observations from automatic evaluations. **Case studies generated by different models are in the Appendix A.**

Ablation Study We also remove the key components of CR-GIS to discuss their contributions in Table 3. **First**, we remove the goal-oriented information fusion mechanism, called "w/o GoInfo.". The HIT has a slight drop on OpendialKG, but decreases significantly on TG-Redial. This is consistent with the observations on the Recall@K. **Second**, we remove the alignment between the interest-level and utterance-level star graphs, called "w/o Align.". In the results, Bleu and Dist decrease slightly, but HIT decreases distinctly on OpenDialKG than TG-Redial. This verifies that the goal-aware interests injected into the utterance representations has an impressive impact on the goal-aware generation. **Third**, we replace the hierarchical Star Transformer encoder with a hierarchical vanilla Transformer encoder, called "w/o HiStar.", and utilize MHA to maintain the alignment between the interest-level star graph and the utterance encoding. We find that although hierarchical Star Transformer would damage the diversity of generation to a certain extent, it has better advantages in terms of improving the quality of generation.

The Effectiveness of Goal-oriented Guidance To further explore the goal perception and generation abilities of CR-GIS, we examine the proportion of the generated response hitting the recommendation goals with the increase of the number of the utterances in the conversation history on the TG-Redial dataset. Besides movies, topics are also used as goals. As shown in Figure 3, as the number of historical utterances increases, the performance of all models increases first and then decreases. Simultaneously, CR-GIS always maintains the best

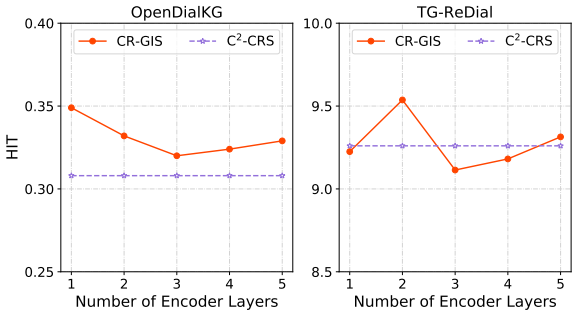

Figure 4: As the number of hierarchical Star Transformer encoder layers increases, CR-GIS consistently outperforms the best baseline C^2-CRS in most settings.

performance. This happens possibly because the topic often shift in the dialog and thus also change the direction of interest sequence.

Parameter Analysis We analyze the influence of the number of the hierarchical Star Transformer encoder layers on HIT. In Figure 4, when the number of layers is greater than 2, the HIT metric shows a downward trend. This happens possibly because the goal-aware interest-level star graph and the utterance-level star graph are overfitted during the alignment due to the number increasing of encoder layers. Furthermore, CR-GIS consistently exceeds the best baseline C^2-CRS in most cases, confirming the effectiveness and robustness of CR-GIS.

6 Conclusion

In this paper, we propose CR-GIS to model goal-aware interest sequence in CRS with an interest-level star graph whose star node is oriented to the recommendation goal. By using goal-oriented interest-level star graph to guide a utterance-level star graph, CR-GIS significantly improve the goal-aware ability in generating smoothly transited response. Extensive experiments verify the superiority of CR-GIS in terms of recommendation, conversation, and reaching recommendation goals.

Acknowledgements

This work was supported by National Natural Science Foundation of China (62272340, 61876128, 61876129, 61976154), State Key Laboratory of Communication Content Cognition (Grant No.A32003).

References

Qibin Chen, Junyang Lin, Yichang Zhang, Ming Ding, Yukuo Cen, Hongxia Yang, and Jie Tang. 2019. Towards knowledge-based recommender dialog system. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing, EMNLP-IJCNLP 2019, Hong Kong, China, November 3-7, 2019*, pages 1803–1813. Association for Computational Linguistics.

Yang Deng, Yaliang Li, Fei Sun, Bolin Ding, and Wai Lam. 2021. Unified conversational recommendation policy learning via graph-based reinforcement learning. In *SIGIR '21: The 44th International ACM SIGIR Conference on Research and Development in Information Retrieval, Virtual Event, Canada, July 11-15, 2021*, pages 1431–1441. ACM.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT 2019, Minneapolis, MN, USA, June 2-7, 2019, Volume 1 (Long and Short Papers)*, pages 4171–4186. Association for Computational Linguistics.

Joseph L Fleiss. 1971. Measuring nominal scale agreement among many raters. *Psychological bulletin*, 76(5):378.

Chongming Gao, Wenqiang Lei, Xiangnan He, Maarten de Rijke, and Tat-Seng Chua. 2021. Advances and challenges in conversational recommender systems: A survey. *AI Open*, 2:100–126.

Qipeng Guo, Xipeng Qiu, Pengfei Liu, Yunfan Shao, Xiangyang Xue, and Zheng Zhang. 2019. Star-transformer. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT 2019, Minneapolis, MN, USA, June 2-7, 2019, Volume 1 (Long and Short Papers)*, pages 1315–1325. Association for Computational Linguistics.

Shirley Anugrah Hayati, Dongyeop Kang, Qingxiaoyang Zhu, Weiyan Shi, and Zhou Yu. 2020. INSPIRED: toward sociable recommendation dialog systems. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing, EMNLP 2020, Online, November 16-20, 2020*, pages 8142–8152. Association for Computational Linguistics.

R. Devon Hjelm, Alex Fedorov, Samuel Lavoie-Marchildon, Karan Grewal, Philip Bachman, Adam Trischler, and Yoshua Bengio. 2019. Learning deep representations by mutual information estimation and maximization. In *7th International Conference on Learning Representations, ICLR 2019, New Orleans, LA, USA, May 6-9, 2019*. OpenReview.net.

Dietmar Jannach, Ahtsham Manzoor, Wanling Cai, and Li Chen. 2021. A survey on conversational recommender systems. *ACM Comput. Surv.*, 54(5):105:1–105:36.

Dongyeop Kang, Anusha Balakrishnan, Pararth Shah, Paul A. Crook, Y-Lan Boureau, and Jason Weston. 2019. Recommendation as a communication game: Self-supervised bot-play for goal-oriented dialogue. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing, EMNLP-IJCNLP 2019, Hong Kong, China, November 3-7, 2019*, pages 1951–1961. Association for Computational Linguistics.

Yoon Kim. 2014. Convolutional neural networks for sentence classification. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing, EMNLP 2014, October 25-29, 2014, Doha, Qatar, A meeting of SIGDAT, a Special Interest Group of the ACL*, pages 1746–1751. ACL.

Diederik P. Kingma and Jimmy Ba. 2015. Adam: A method for stochastic optimization. In *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*.

Ivica Kostric, Krisztian Balog, and Filip Radlinski. 2021. Soliciting user preferences in conversational recommender systems via usage-related questions. In *RecSys '21: Fifteenth ACM Conference on Recommender Systems, Amsterdam, The Netherlands, 27 September 2021 - 1 October 2021*, pages 724–729. ACM.

Wenqiang Lei, Xiangnan He, Yisong Miao, Qingyun Wu, Richang Hong, Min-Yen Kan, and Tat-Seng Chua. 2020. Estimation-action-reflection: Towards deep interaction between conversational and recommender systems. In *WSDM '20: The Thirteenth ACM International Conference on Web Search and Data Mining, Houston, TX, USA, February 3-7, 2020*, pages 304–312. ACM.

Raymond Li, Samira Ebrahimi Kahou, Hannes Schulz, Vincent Michalski, Laurent Charlin, and Chris Pal. 2018. Towards deep conversational recommendations. In *Advances in Neural Information Processing Systems 31: Annual Conference on Neural Information Processing Systems 2018, NeurIPS 2018, December 3-8, 2018, Montréal, Canada*, pages 9748–9758.

Yujia Li, Daniel Tarlow, Marc Brockschmidt, and Richard S. Zemel. 2016. Gated graph sequence neural networks. In *4th International Conference on Learning Representations, ICLR 2016, San Juan, Puerto Rico, May 2-4, 2016, Conference Track Proceedings*.

Zujie Liang, Huang Hu, Can Xu, Jian Miao, Yingying He, Yining Chen, Xiubo Geng, Fan Liang, and Daxin Jiang. 2021. Learning neural templates for recommender dialogue system. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing, EMNLP 2021, Virtual Event / Punta Cana, Dominican Republic, 7-11 November, 2021*, pages 7821–7833. Association for Computational Linguistics.

Lizi Liao, Ryuichi Takanobu, Yunshan Ma, Xun Yang, Minlie Huang, and Tat-Seng Chua. 2019. Deep conversational recommender in travel. *ArXiv*, abs/1907.00710.

Yu Lu, Junwei Bao, Yan Song, Zichen Ma, Shuguang Cui, Youzheng Wu, and Xiaodong He. 2021. Revcore: Review-augmented conversational recommendation. In *Findings of the Association for Computational Linguistics: ACL/IJCNLP 2021, Online Event, August 1-6, 2021*, volume ACL/IJCNLP 2021 of *Findings of ACL*, pages 1161–1173. Association for Computational Linguistics.

Kai Luo, Scott Sanner, Ga Wu, Hanze Li, and Hojin Yang. 2020. Latent linear critiquing for conversational recommender systems. In *WWW '20: The Web Conference 2020, Taipei, Taiwan, April 20-24, 2020*, pages 2535–2541. ACM / IW3C2.

Wenchang Ma, Ryuichi Takanobu, and Minlie Huang. 2021. Cr-walker: Tree-structured graph reasoning and dialog acts for conversational recommendation. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing, EMNLP 2021, Virtual Event / Punta Cana, Dominican Republic, 7-11 November, 2021*, pages 1839–1851. Association for Computational Linguistics.

Seungwhan Moon, Pararth Shah, Anuj Kumar, and Rajen Subba. 2019. Opendialkg: Explainable conversational reasoning with attention-based walks over knowledge graphs. In *Proceedings of the 57th Conference of the Association for Computational Linguistics, ACL 2019, Florence, Italy, July 28- August 2, 2019, Volume 1: Long Papers*, pages 845–854. Association for Computational Linguistics.

Zhiqiang Pan, Fei Cai, Wanyu Chen, Honghui Chen, and Maarten de Rijke. 2020. Star graph neural networks for session-based recommendation. In *CIKM '20: The 29th ACM International Conference on Information and Knowledge Management, Virtual Event, Ireland, October 19-23, 2020*, pages 1195–1204. ACM.

Michael Sejr Schlichtkrull, Thomas N. Kipf, Peter Bloem, Rianne van den Berg, Ivan Titov, and Max Welling. 2018. Modeling relational data with graph convolutional networks. In *The Semantic Web - 15th International Conference, ESWC 2018, Heraklion, Crete, Greece, June 3-7, 2018, Proceedings*, volume 10843 of *Lecture Notes in Computer Science*, pages 593–607. Springer.

Yueming Sun and Yi Zhang. 2018. Conversational recommender system. In *The 41st International ACM SIGIR Conference on Research & Development in Information Retrieval, SIGIR 2018, Ann Arbor, MI, USA, July 08-12, 2018*, pages 235–244. ACM.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, December 4-9, 2017, Long Beach, CA, USA*, pages 5998–6008.

Bo Xu, Yong Xu, Jiaqing Liang, Chenhao Xie, Bin Liang, Wanyun Cui, and Yanghua Xiao. 2017. Cndbpedia: A never-ending chinese knowledge extraction system. In *Advances in Artificial Intelligence: From Theory to Practice - 30th International Conference on Industrial Engineering and Other Applications of Applied Intelligent Systems, IEA/AIE 2017, Arras, France, June 27-30, 2017, Proceedings, Part II*, volume 10351 of *Lecture Notes in Computer Science*, pages 428–438. Springer.

Kerui Xu, Jingxuan Yang, Jun Xu, Sheng Gao, Jun Guo, and Ji-Rong Wen. 2021. Adapting user preference to online feedback in multi-round conversational recommendation. In *WSDM '21, The Fourteenth ACM International Conference on Web Search and Data Mining, Virtual Event, Israel, March 8-12, 2021*, pages 364–372. ACM.

Yiming Zhang, Lingfei Wu, Qi Shen, Yitong Pang, Zhihua Wei, Fangli Xu, Bo Long, and Jian Pei. 2021. Multi-choice questions based multi-interest policy learning for conversational recommendation. *arXiv preprint arXiv:2112.11775*.

Jinfeng Zhou, Bo Wang, Ruifang He, and Yuexian Hou. 2021. CRFR: improving conversational recommender systems via flexible fragments reasoning on knowledge graphs. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing, EMNLP 2021, Virtual Event / Punta Cana, Dominican Republic, 7-11 November, 2021*, pages 4324–4334. Association for Computational Linguistics.

Kun Zhou, Wayne Xin Zhao, Shuqing Bian, Yuanhang Zhou, Ji-Rong Wen, and Jingsong Yu. 2020a. Improving conversational recommender systems via knowledge graph based semantic fusion. In *KDD '20: The 26th ACM SIGKDD Conference on Knowledge Discovery and Data Mining, Virtual Event, CA, USA, August 23-27, 2020*, pages 1006–1014. ACM.

Kun Zhou, Yuanhang Zhou, Wayne Xin Zhao, Xiaoke Wang, and Ji-Rong Wen. 2020b. Towards topic-guided conversational recommender system. In *Proceedings of the 28th International Conference on Computational Linguistics, COLING 2020, Barcelona, Spain (Online), December 8-13, 2020*, pages 4128–4139. International Committee on Computational Linguistics.

Yuanhang Zhou, Kun Zhou, Wayne Xin Zhao, Cheng Wang, Peng Jiang, and He Hu. 2022. C2-CRS: coarse-to-fine contrastive learning for conversational recommender system. *CoRR*, abs/2201.02732.

Jie Zou, Yifan Chen, and Evangelos Kanoulas. 2020. Towards question-based recommender systems. In *Proceedings of the 43rd International ACM SIGIR conference on research and development in Information Retrieval, SIGIR 2020, Virtual Event, China, July 25-30, 2020*, pages 881–890. ACM.

A Case Study

Two cases generated on OpenDialKG and TG-ReDial datasets from seven models and ground truth are selected in Table 5. We use ***red text*** to annotate the defects in expressing the recommendation goals for each response generated by the baseline models. In comparison, the advantage of CR-GIS is that its responses are more coherent, fluent, and informative and achieve recommendation goals that are more in line with user interests. This observation further verifies the advantage of CR-GIS in modeling goal-aware implicit user interest sequence, which benefits the smooth transition of utterances towards recommendation goal.

	OpenDialKG
Context	I like J.K.Rowling. Me, too! She's great! Did you want to know some cool stuff about her? Yes, I want to know.
Transformer	Yes, I did. It was released in 2013. It was written by the film. (***incoherence***)
KBRD	JK Rowlings wrote this book (***less-informativeness***)
KGSF	Yes, I can recommend Harry Potter and the Sorcerer's Apprentice. (***less-coherence, less-informativeness***)
RevCore	Yes, I think she was in the Harry Potter and the Philosopher's Stone. Did you know that one also starred in the Philosopher's Stone? (***fact inconsistency, i.e., not "was in", fact is "wrote"***)
CRFR	Yes, he wrote The Tales of Beedle the Bard and The Cuckoo 's Calling. Have you seen any (***less-fluency, gender inconsistency***)
C^2-CRS	Yes, I didn't know that. Did he also write the Harry Potter series? (***less-informativeness, gender inconsistency***)
CR-GIS	**Sure, she wrote The Tales of Beedle the Bard, The Silkworm, which is a series. Have you read any of those?**
Ground-Truth	Did you know she wrote two books called The Silkworm as well as the book The Cuckoo's Calling?
	TG-ReDial
Context	好啊，我比较喜欢人性题材的，真实改编的电影。 Well, I prefer movies based on human nature and reality. 推荐一部《现代启示录》，真的展现出人性混沌无善恶，残酷壮丽一瞬间。 Recommend a Apocalypse Now, it really shows the chaos of human nature, no good and evil, cruel and magnificent moment. 你说这个影片能展现出人性混沌无善恶，残酷壮丽一瞬间，这个评价很高啊，一会看看。 You said that this movie can show the chaos of human nature without good and evil, cruel and magnificent for a moment, the evaluation is very high, I will see it later. 推荐你看《势不两立》，是关于人性的背叛，很现实的电影，两个主角都很饱满，对人性的塑造非常到位。 I recommend you to watch The Edge, which is a very realistic film about the betrayal of human nature. The two protagonists are full and the shaping of human nature is very in place. 感觉这样的电影展现的就是：这样的人越是苦难，越是闪耀人性光辉。能让我们感悟很多吧? I feel that what such a movie shows is: the more suffering such a person is, the more shining the brilliance of human nature. Can we learn a lot?
Transformer	还有一部关于人性的电影，叫《现代启示录》，这也是一部关于 There is also a movie about human nature called Apocalypse Now, which is also a movie about (***retell, nonfluency***)
KBRD	看过《UNK》吗，这是一部好电影，人性的原版看起来要看起来，但很爽。 Have you seen <UNK>, it's a good movie, the original version of human nature looks to look good, but it's cool. (***less-informativeness***)
KGSF	《双重人格》非常适合你，影片中的人性，人性，两个极端极端探讨。 The Double is very suitable for you, the movie's human nature, human nature, two extreme extreme exploration. (***repetition***)
RevCore	《人性污点到底是什么才是烂片之一。还有其他关于人性的电影吗? What the hell is The Human Stain is one of the bad movies. Are there any other movies about human nature? (***nonfluency, incoherence***)
CRFR	《越战创伤》非常好，老帕还挺对我胃口。 Casualties of War is very good, and Palma is kinda to my taste. (***less-coherence, none user-centric***)
C^2-CRS	行啊，最好是那种反映人性心理社会的讽刺题材电影。 Okay, it's better to be a satirical movie that reflects human psychology and society. (***less-informativeness***)
CR-GIS	《忏情记》满足你的要求，人性与法律，救赎与出卖。 **I Confess meets your requirements, human nature and law, redemption and betrayal.**
Ground-Truth	有一部电影叫《忏情记》，希区柯克演的人性题材的电影，结局是人性光辉的闪耀，也是神父之罪的阐述。 There is a movie called I Confess, a movie about human nature played by Hitchcock. The ending is the shining of human nature, and it is also an exposition of the sin of the priest.

Table 5: **Case Study**. Generated responses from baselines and CR-GIS. Context is the multi-turn conversation history in which the seeker and the recommender speak alternately.

GRASP: Guiding model with RelAtional Semantics using Prompt for Dialogue Relation Extraction

Junyoung Son[*], **Jinsung Kim**[*], **Jungwoo Lim**[*], **Heuiseok Lim**[†]
Computer Science and Engineering, Korea University
Republic of Korea
{s0ny,jin62304,wjddn803,limhseok}@korea.ac.kr

Abstract

The dialogue-based relation extraction (DialogRE) task aims to predict the relations between argument pairs that appear in dialogue. Most previous studies utilize fine-tuning pre-trained language models (PLMs) only with extensive features to supplement the low information density of the dialogue by multiple speakers. To effectively exploit inherent knowledge of PLMs without extra layers and consider scattered semantic cues on the relation between the arguments, we propose a **G**uiding model with **R**el**A**tional **S**emantics using **P**rompt (**GRASP**). We adopt a prompt-based fine-tuning approach and capture relational semantic clues of a given dialogue with 1) an argument-aware prompt marker strategy and 2) the relational clue detection task. In the experiments, **GRASP** achieves state-of-the-art performance in terms of both F1 and $F1_c$ scores on a DialogRE dataset even though our method only leverages PLMs without adding any extra layers.

Dialogue

1 S1: Hey guys! Hey!
2 S2: Hey **Pheebs**, guess who we saw today.
3 **S3**: Ooh, ooh, fun! Okay... um, Liam Neeson.
...
7 **S3**: The woman who cuts my hair!
8 S4: Okay, look, this could be a really long game.
9 S5: Your sister **Ursula**.
10 **S3**: Oh, really.
11 S5: Yeah, yeah, she works over at that place, uh...
12 **S3**: **Rift's**. Yeah, I know.
...
17 S6: Um, **Pheebs**, so, you guys just don't get along?
18 **S3**: It's mostly just dumb sister stuff, you know, I mean, like, everyone always thought of her as the pretty one, you know... ...

	Argument pair	Trigger	Relation Type
R1	(Pheebs, PER) (Ursula, PER)	sister	per:siblings
R2	(S3, PER) (Pheebs, PER)	none	per:alternate_names
R3	(Rift's, ORG) (Ursula, PER)	works over at	per:employees_or_members

Table 1: Example of DialogRE data. The arguments are bold, and the triggers are underlined. The arguments and triggers are scattered throughout the dialogue, which leads to low information density. The triggers determine the direction of the proper relation indirectly.

1 Introduction

The relation extraction (RE) task aims to extract semantic relations from unstructured text such as a sentence, a document, or even a dialogue. RE plays a critical role in information extraction and knowledge base construction as it can extract structured relational information (Ji et al., 2010; Swampillai and Stevenson, 2010). However, the utilization of sentence-level RE in a conversational setting is limited because numerous relational facts appear across multiple sentences with more than one speaker in a dialogue (Yao et al., 2021). Thus, the dialogue-based relation extraction (DialogRE) task, which includes argument pairs and their corresponding relations, has been proposed to encourage building a model that captures the underlying semantic spread in the dialogue, as presented in Table 1 (Yu et al., 2020).

In previous studies where state-of-the-art (SoTA) performance is achieved on DialogRE benchmarks, fine-tuning is employed on pre-trained language models (PLMs) (Lee and Choi, 2021; Long et al., 2021), such as BERT (Devlin et al., 2019) and RoBERTa (Liu et al., 2019). As fine-tuning requires the addition of extra layers on top of the PLMs and the training objectives are different from those used in the pre-training phase, PLMs cannot effectively exploit their learned knowledge in the downstream task, resulting in less generalized capability (Chen et al., 2021).

To effectively utilize knowledge from PLMs, several studies involving prompt-based fine-tuning have been conducted. They employ the PLM directly as a predictor and completing a cloze task

[*] These authors have equally contributed to this work
[†] Corresponding author

Proceedings of the 29th International Conference on Computational Linguistics, pages 412–423
October 12–17, 2022.

Figure 1: An illustration of (1) standard fine-tuning approach, and (2) prompt-based fine-tuning approach. In prompt-based fine-tuning approach, a set of *label words* are mapped into a *class set* by a certain mapping function.

to bridge the gap between pre-training and fine-tuning (Gao et al., 2020; Han et al., 2021). As presented in Figure 1, prompt-based fine-tuning treats the downstream task as a masked language modeling (MLM) problem by directly generating the textual response to a given template. In specific, prompt-based fine-tuning updates the original input on the basis of the template and predicts the *label words* with the `[MASK]` token. Afterwards, the model maps predicted *label words* to corresponding task-specific class sets.

However, the prompt-based fine-tuning approach is still not sufficient in terms of performance compared with the fine-tuning-based approach (Han et al., 2021; Chen et al., 2021). We attribute this phenomenon to the following properties of conversation: higher person-pronoun frequency (Wang and Liu, 2011) and lower information density (Biber, 1991) by multiple speakers[1]. Therefore, a prompt-based fine-tuning approach that collects the sparse semantics in the dialogue is required to understand relation between the arguments.

We propose a method named **G**uiding model with **R**el**A**tional **S**emantics using **P**rompt (**GRASP**) for DialogRE. To maximize the advantages of the prompt-based fine-tuning approach for the DialogRE task, we suggest an argument-aware prompt marking (APM) strategy and a relational clue detection (RCD) task. The APM strategy guides the model to the significant arguments scattered in the dialogue by carefully considering arguments. For our APM strategy, we conduct empirical study based on the diverse marker types to validate our APM strategy. Along with the APM strategy, the suggested RCD task with a training objective leads the model to pay attention to significant relational clues. Specifically, the model is trained to determine whether each token in a dialogue belongs to a subject, object, or trigger. As a result, the PLM is trained on RCD and MLM jointly. In the experiments, our method achieves SoTA performance at a significant level in the DialogRE task for both the full-shot and few-shot settings. Only PLMs are employed without the addition of an extra layer as a predictor, and **GRASP** exhibits a higher performance than other baselines. The significant performance improvement indicates that attending to significant semantic clues guides the PLMs to predict the correct relation with its inherent knowledge in both full-shot and few-shot settings. Moreover, we provide ablation studies and qualitative analysis on the robustness of **GRASP**.

Our contributions are as follows:

- We adopt a prompt-based fine-tuning approach to utilize a PLM's inherent knowledge directly for dialogues with relatively low information density.
- We introduce an APM strategy and a RCD task that guide PLMs on the significant relational clues, which are semantic information to predict relations.
- We demonstrate that our proposed method achieves SoTA performance on the DialogRE task in both full-shot and few-show settings.
- We conduct ablation studies and qualitative analysis to validate the robustness of **GRASP**.

The remainder of this paper is organized as follows. In Section 3, we present the entire process of our method in detail. The experimental setup and the results are explained in Section 4. The further analyses is provided in Section 5, and Section 6

[1]In the DialogRE dataset, 65.9% of relational triples involve arguments that never appear in the same utterance, demonstrating that multi-turn reasoning plays an important role.

presents the conclusions. Appendix 2 provides related works including the prompt-based learning, and the DialogRE.

2 Related Works

Prompt-based learning Prompt-based learning is a method of reducing the gap between the pre-training objective and that of fine-tuning. For example, language models such as BERT (Devlin et al., 2019) use masked language modeling (MLM) objective in pre-training phase where the model fills the `[MASK]` token whereas the model trains without `[MASK]` token in fine-tuning phase by adding an extra classifier layer. As a result, the discrepancy between training objectives prevents PLM from leveraging knowledge acquired from the pre-training enormous corpus (Chen et al., 2021). Also, prompt-based learning shows better performance than fine-tuning especially in the few-shot setting (Gao et al., 2020; Schick and Schütze, 2021; Li and Liang, 2021; Liu et al., 2021).

DialogRE Recent studies on the DialogRE dataset show a tendency to fine-tune the PLM with task-specific objectives and use the model with high-complexity (Xue et al., 2021; Long et al., 2021; Lee and Choi, 2021). In detail, Lee and Choi (2021) shows considerable performance with the contextualized turn representations from the diverse type of nodes and edges. Moreover, task-specific objectives of fine-tuning lead to a gap between pre-training and fine-tuning.

To overcome the limitation, Han et al. (2021) utilizes multiple `[MASK]` tokens for each argument and the relation with logical rules by concentrating subject and object in DialogRE. There also exists an approach that incorporates potential knowledge contained in relation labels into prompt construction with trainable virtual type words and answers words. This approach also carefully initializes the virtual tokens with implicit semantic words and employs prior distributions estimated from the data (Chen et al., 2021).

Despite the prompt-based approach's high potential, few prompting studies sufficiently consider low information density and difficulty of capturing intrinsic relational information of the data between the argument pair of dialogue relation extraction task. We focus on building a light model with prompt-based fine-tuning with implicit semantic information of the relation which alleviates the sparsity problem.

3 Methodology

An overview of **GRASP** is illustrated in Figure 2. First, an input with a prompt template is constructed by using the APM strategy. Then, the PLM receives the constructed input for prompt-based fine-tuning and estimates probability distributions of the model's vocabulary by using contextualized representations of the PLM. The RCD task lets the model predict the relational clue type of each token, and the model takes the `[MASK]` representation to predict a final relation in the MLM task. In other words, our model is trained through multi-task learning to encourage mutual communication between relational clues and the final relation for the argument pair.

3.1 Problem Formulation

Each example $\mathcal{X}$ includes dialogue D, subject a_1, and object a_2. Note that D denotes $\{s_1 : u_1, s_2 : u_2, \ldots, s_N : u_N\}$, where s_n is the n-th speaker and u_n is the corresponding utterance. Given $\mathcal{X} = \{D, a_1, a_2\}$, the goal of DialogRE is to predict relation y between arguments a_1 and a_2 by leveraging D. To describe the DialogRE task in terms of prompt-based fine-tuning, a template function, $\mathcal{T}(\cdot)$, is defined to map each example to $\mathcal{X}_{prompt} = \mathcal{T}(\mathcal{X})$. A `[MASK]` is inserted into $\mathcal{X}_{prompt}$, and used to predicting the label words of relation y. The formulation of $\mathcal{T}(\mathcal{X})$ is as follows:

$$\mathcal{T}(\mathcal{X}) = \text{“}\texttt{[CLS]}\, D\, \texttt{[SEP]}\, \texttt{[subj]}\, a_1\, \texttt{[subj]}\, \texttt{[MASK]}\, \texttt{[obj]}\, a_2\, \texttt{[obj]}\, \texttt{[SEP]}\text{”}. \quad (1)$$

Based on the structure of $\mathcal{T}(\cdot)$, we construct our template function, $\mathcal{T}'(\cdot)$, by applying two steps: transformation of D to D' with an argument-aware prompt marker, described in 3.2, and prompt initialization for `[subj]` and `[obj]`, explained in 3.3. Subsequently, we introduce the RCD task in 3.4 to train the model by utilizing $\mathcal{T}'(\cdot)$ with a multitask training strategy on MLM, as detailed in 3.5.

3.2 Argument-aware Prompt Marker

We propose an APM strategy that considers both speaker and non-speaker arguments. In previous studies, a dialogue is encoded by focusing on speaker information (Lee and Choi, 2021; Yu et al., 2020; Chen et al., 2021) without focusing non-speaker arguments. However, in the DialogRE dataset, approximately 77.4% of relation triples

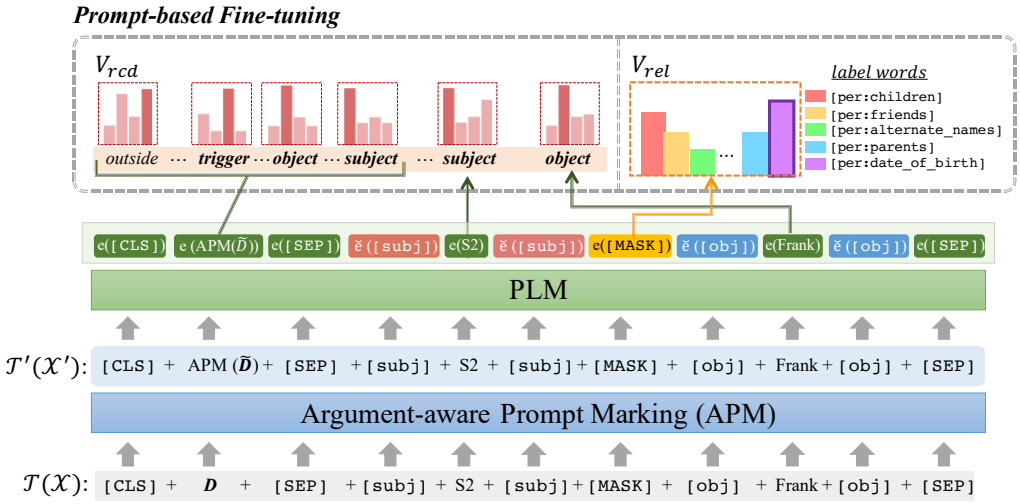

Figure 2: The overall model architecture of **GRASP**. By formalizing specific tasks as MLM tasks, the model predicts answers with the tokens from the model's vocabulary; for example, the *label words* of V_{rel} and V_{rcd} are from the model's vocabulary.

include at least one non-speaker argument, implying that consideration of non-speaker arguments is also inevitable to enhance the model's argument-awareness.

Our methods is inspired by the previous works of Soares et al. (2019) and Han et al. (2021); the former addresses the importance of entity markers regarding recognition of the entity position, and the latter improves the model performance by using specific additional punctuation in the model's original vocabulary as the entity marker. Accordingly, we insert the argument-aware prompt marker token, `[p]`, as an entity marker. The argument-aware prompt markers allow our model to obtain informative signs to determine which token is the indicative component for relation prediction. We initialize the feature of `[p]` as the embedding of the space token in the vocabulary of the model. Our empirical experiments reveal that the space token can perceive the start position of the arguments. Consequently, our prompt marker enhances the model to discriminate which part of the dialogue plays a critical role in predicting a relation.

Using the proposed argument-aware prompt marker, we strengthen the token replacement method of BERT$_s$ (Yu et al., 2020). Given example $\mathcal{X}$, BERT$_s$ constructs $\tilde{\mathcal{X}} = \{\tilde{D}, \tilde{a_1}, \tilde{a_2}\}$, where $\tilde{D} = \{\tilde{s}_1 : u_1, \tilde{s}_2 : u_2, \ldots, \tilde{s}_N : u_N\}$ and $\tilde{s_n}$ is

$$\tilde{s_n} = \begin{cases} \texttt{[S}_1\texttt{]} & \text{if } s_n = a_1 \\ \texttt{[S}_2\texttt{]} & \text{if } s_n = a_2 \\ s_n & \text{otherwise.} \end{cases} \quad (2)$$

`[S`$_1$`]` and `[S`$_2$`]` are special tokens for speakers. In addition, $\tilde{a}_m$ ($m \in \{1, 2\}$) is defined as `[S`$_m$`]` if $\exists\, n\ (s_n = a_m)$ and a_m otherwise.

Even though BERT$_s$ prevents the model from overfitting and demonstrates higher generalization capacity (Yu et al., 2020), BERT$_s$ does not consider non-speaker arguments. To explore the disregarded arguments, the APM strategy expands BERT$_s$ by considering both types of arguments in the DialogRE task. We define function `APM(`$\cdot$`)` that encodes an utterance by inserting `[p]` in front of each argument token. Given dialogue $\tilde{D}$ constructed by using BERT$_s$, we construct $D' = \{\tilde{s}_1 : u'_1, \tilde{s}_2 : u'_2, \ldots, \tilde{s}_N : u'_N\}$ by applying `APM`(u_n) for each utterance in $\tilde{D}$. Consequently, the APM strategy constructs $\mathcal{X}' = \{D', \tilde{a}_1, \tilde{a}_2\}$ based on $\tilde{\mathcal{X}}$.

$$u'_n = \begin{cases} \texttt{APM}(u_n) & \text{if } \exists\, m\ (\tilde{a}_m \in u_n)\ (m \in \{1, 2\}) \\ u_n & \text{otherwise} \end{cases} \quad (3)$$

For instance, `APM`$(\cdot)$ encodes the text "I am Tom Gordon" to `[I, am, [p], Tom, Gordon]`, inserting `[p]` in front of "Tom Gordon."

3.3 Prompt Construction

We update the constructed input $\mathcal{X}'$ with a template function and conduct a deliberate initialization. We utilize the prior distribution of argument types for initialization inspired by the study conducted by Chen et al. (2021). Prompt tokens `[subj]` and `[obj]` are used to inject argument-type information. We define the argument-type set, $\mathcal{AT}$ = {"PER," "ORG," "GPE," "VALUE,"

"STRING"}, using the types pre-defined in the dataset as depicted in Table 1. We calculate the distributions of argument types $\phi^{\texttt{[subj]}}$ and $\phi^{\texttt{[obj]}}$ over $\mathcal{AT}$ by using frequency statistics. We aggregate each argument type, $at \in \mathcal{AT}$, with the corresponding prior distribution to initialize prompt tokens `[subj]` and `[obj]`. The specific initialization equations are as follows:

$$\begin{aligned} \check{e}(\texttt{[subj]}) &= \sum_{at \in \mathcal{AT}} \phi_{at}^{\texttt{[subj]}} \cdot e(at) \\ \check{e}(\texttt{[obj]}) &= \sum_{at \in \mathcal{AT}} \phi_{at}^{\texttt{[obj]}} \cdot e(at), \end{aligned} \tag{4}$$

where $e(\cdot)$ is the embedding from the PLM of an input token and $\check{e}(\cdot)$ is the initialized embedding of the prompt token. Suppose the subject has prior distribution, $\phi^{\texttt{[subj]}} = \{$"PER": 0.5,'"ORG":0.5, "GPE":0.0,"VALUE":0.0,"STRING":0.0$\}$. The initial embedding of the `[subj]` token can be calculated as a weighted average, i.e., $\check{e}(\texttt{[subj]}) = 0.5 \cdot e(\text{"PER"}) + 0.5 \cdot e(\text{"ORG"})$.

Consequently, we can formalize $\mathcal{T}'(\cdot)$, which converts $\mathcal{X}'$ to $\mathcal{X}'_{prompt}$, where $\mathcal{X}'_{prompt}$ is an argument-enhanced input example, by using the APM strategy and prompt construction with deliberate initializations, i.e., $\mathcal{X}'_{prompt} = \mathcal{T}'(\mathcal{X}')$. Then, the final input structure for prompt-based fine-tuning is as follows:

$$\begin{aligned} \mathcal{T}'(\mathcal{X}') = \text{"}&\texttt{[CLS]}\, D' \,\texttt{[SEP]}\,\texttt{[subj]}\,\tilde{a}_1\,\texttt{[subj]} \\ &\texttt{[MASK]}\,\texttt{[obj]}\,\tilde{a}_2\,\texttt{[obj]}\,\texttt{[SEP]}\text{"}. \end{aligned} \tag{5}$$

In addition, for relation prediction by applying MLM, we also define $\mathcal{V}_{rel}$ as a set of *label words* in the model's vocabulary as illustrated in Figure 2. In detail, we utilize its metadata for the initialization of each relation representation to inject its semantics. For instance, we add a special token to the model's vocabulary, `[per:date_of_birth]` as a *label word*, and initialize this token by aggregating the embeddings of the words in the metadata, i.e., {"person," "date," "of," "birth"} for a class "per:date_of_birth."

3.4 Relational Clue Detection task

To improve the understanding capability on relational clues by employing a prompt-based fine-tuning approach, we introduce a RCD task. We define a set of *label words*, $\mathcal{V}_{rcd}$ ={`[subject]`, `[object]`, `[trigger]`, `[outside]`}, and add to the model's vocabulary. Then, we construct a sequence of *label words* for RCD, $\mathcal{C}_{rcd}$, by assigning each token in $\mathcal{X}'_{prompt}$ to the corresponding clue type word from $\mathcal{V}_{rcd}$. For instance, $\mathcal{C}_{rcd}$ is constructed as follows: {`[subject]`, `[trigger]`, `[object]`} when the token sequence is given by {*"Pheebs"*, *"lives in"*, *"LA"*}, where *"Pheebs"* is a subject and *"LA"* is an object argument, and *"lives in"* is a trigger.

The RCD task exploits the MLM head that is used to predict the `[MASK]` token. Except for the `[MASK]` token, each token is sequentially tagged with $\mathcal{V}_{rcd}$ using the meta-data provided in the dataset. In other words, the RCD task allows the model to identify which non-`[MASK]` tokens correspond to certain relational clue types. RCD supports the model in collecting scattered information from the entire dialogue by indicating where to focus in the dialogue to predict a relation. In this respect, the model pays considerably more attention to the semantic clues of the relation, such as triggers. Moreover, our model maintains a lightweight complexity by conducting the RCD task without an additional classifier. The loss for the RCD task over each token $x \in \mathcal{X}'_{prompt}$ is aggregated as follows:

$$\mathcal{L}_{RCD} = -\sum_{x \in \mathcal{X}'_{prompt}} \log P(x = \mathcal{C}_{rcd}(x) | \mathcal{X}'_{prompt}). \tag{6}$$

3.5 Model Training

Before the final relation-prediction, we mark the position of the trigger using the `[p]` token, i.e., the argument-aware prompt marker, based on the results of the RCD task. Given example $\mathcal{X}'_{prompt}$, the model predicts the *label words* of $\mathcal{V}_{rcd}$ for all non-`[MASK]` tokens. Subsequently, `[p]` is appended in front of the words predicted as a trigger to train the model to distinguish essential clues that encourage determining the relation.

Multitask Learning $\mathcal{M}_{rel} : \mathcal{Y} \rightarrow \mathcal{V}_{rel}$ is a mapping function that converts a class set, $\mathcal{Y}$, into a set of *label words*, $\mathcal{V}_{rel}$. For each input $\mathcal{X}'_{prompt}$, the purpose of MLM is to fill `[MASK]` with the relation *label words* in the model's vocabulary. As the model predicts the correct *label word* at the position of `[MASK]`, we can formulate $p(y|x) = P(\texttt{[MASK]} = \mathcal{M}_{rel}(y) | \mathcal{X}'_{prompt})$. The training objective of the relation prediction is to minimize

$$\mathcal{L}_{REL} = -\log P(\texttt{[MASK]} = \mathcal{M}_{rel}(y) | \mathcal{X}'_{prompt}). \tag{7}$$

To improve the model's ability to capture relational clues through the interaction between the MLM task for relation prediction and the RCD

Full-shot Setting				
Method	V1		V2	
	F1	$F1_c$	F1	$F1_c$
Fine-tuning based approach				
$BERT_s$ (Yu et al., 2020)	61.2	55.4	-	-
$RoBERTa_s$ (Lee and Choi, 2021)	-	-	71.3	63.7
Dual (Bai et al., 2021)	67.3	61.4	67.1	61.1
CoIn (Long et al., 2021)	72.3	-	-	-
TUCORE-GCN (Lee and Choi, 2021)	-	-	73.1	65.9
Prompt-based fine-tuning approach				
PTR (Han et al., 2021)	63.2	-	-	-
KnowPrompt (Chen et al., 2021)	68.6	-	-	-
$\mathbf{GRASP}_{base}$ (Our model)	69.2	62.4	69.0	61.7
$\mathbf{GRASP}_{large}$ (Our model)	**75.1** (+2.8)	**66.7** (+5.3)	**75.5** (+2.4)	**67.8** (+1.9)

Table 2: Performances of **GRASP** on test set of DialogRE. V1 and V2 represent the version of the dataset. The underlined scores are the previous SoTA performances. Subscript in parentheses represents advantages of **GRASP** over the best results of baselines (the underlined). Best results are bold.

task, we employ a multitask learning to train the **GRASP** model by using the joint loss expressed in Equations (6) and (7). Therefore, the final learning objective is to minimize the joint loss, where λ_1 and λ_2 are hyperparameters.

$$\mathcal{L}_{GRASP} = \lambda_1 \cdot \mathcal{L}_{RCD} + \lambda_2 \cdot \mathcal{L}_{REL} \quad (8)$$

4 Experiments

4.1 Experimental Setup

For the base PLMs, RoBERTa-base and RoBERTa-large are adopted, as denoted by $\mathbf{GRASP}_{base}$ and $\mathbf{GRASP}_{large}$, respectively. The evaluation metrics are the F1 and $F1_c$ (Yu et al., 2020) scores. $F1_c$ is an evaluation metric for supplementing the F1 score in a conversational setting and is computed by employing part of the dialogue necessary for predicting the relation between given arguments as input instead of the entire dialogue. The detailed settings for **GRASP** can be found in Appendix A.

In a full-shot setting, **GRASP** is compared with both fine-tuning-based and prompt-based fine-tuning approaches. TUCORE-GCN (Lee and Choi, 2021) is a typical fine-tuning-based model using turn-level features with a graph convolution network (Kipf and Welling, 2016). CoIn (Long et al., 2021) employs utterance-aware and speaker-aware representations, and Dual (Bai et al., 2021) models the relational semantics using abstract meaning representations (Banarescu et al., 2013). Moreover, PTR (Han et al., 2021) and KnowPrompt (Chen et al., 2021) are the prompt-based fine-tuning baseline models. In the few-shot setting, 8-, 16-, and 32-shot experiments were conducted based on LM-BFF (Gao et al., 2020) by using three different randomly sampled data.

4.2 Experimental Results

Full-shot setting As presented in Table 2, it is shown that $\mathbf{GRASP}_{large}$ surpasses all of the baseline models, including the current SoTA models, that is, CoIn and TUCORE-GCN. From the result in the full-shot setting, the baselines of the fine-tuning-based approach, such as TUCORE-GCN or CoIn, show better performance than those of the prompt-based fine-tuning approach. In particular, CoIn outperforms all of the other baselines, including PTR and KnowPrompt on V1, and TUCORE-GCN exhibits the best performance on V2. Interestingly, even though the performance of the prompt-based fine-tuning baselines is much lower than fine-tuning based models, our $\mathbf{GRASP}_{large}$ outperforms regardless of the way of training approach.

In addition, $\mathbf{GRASP}_{large}$ shows its efficiency in conversational settings by exceeding all the baselines in terms of $F1_c$, thereby indicating that our method effectively overcomes the low information density of dialogues. These results imply that guiding the model to pay attention to relational clues with a prompt-based fine-tuning approach can be more effective than adding additional features and layers. The slightly low performance of $\mathbf{GRASP}_{base}$ is attributed to the gap in the model size; for example, TUCORE-GCN has 401M

Few-shot Setting			
Method	Shot		
	K=8	*K=16*	*K=32*
Fine-tuning based approach			
RoBERTa (Chen et al., 2021)	29.8	40.8	49.7
TUCORE-GCN (Lee and Choi, 2021)	24.6	40.0	53.8
Prompt-based fine-tuning approach			
PTR (Han et al., 2021)	35.5	43.5	49.5
KnowPrompt (Chen et al., 2021)	<u>43.8</u>	50.8	55.3
$\mathbf{GRASP}_{base}$ (Our model)	**45.4**	<u>52.0</u>	<u>56.0</u>
$\mathbf{GRASP}_{large}$ (Our model)	36.0	**55.3**	**62.6**

Table 3: Low-resource RE performance of F1 scores (%) on different test sets. We use K = 8, 16, 32 (# of examples per class) for few-shot experiments. Best results are bold and the second place results are underlined.

weight parameters with RoBERTa-large model, and $\mathbf{GRASP}_{base}$ has 125M weight parameters with RoBERTa-base. Nevertheless, $\mathbf{GRASP}_{base}$ demonstrates 6.0%p and 0.6%p improvements compared with the other prompt-based fine-tuning baselines, i.e., PTR and KnowPrompt, respectively.

Few-shot setting As presented in Table 3, **GRASP** still exhibits robust performance in few-shot settings. $\mathbf{GRASP}_{base}$ outperforms the baselines of both the fine-tuning and prompt-based fine-tuning methods regardless of the number of shots, demonstrating 20%p or higher performance in the 8-shot setting compared with TUCORE-GCN and indicating that our method is more efficient than the fine-tuning method in a low-resource setting. $\mathbf{GRASP}_{base}$ also demonstrates improved performance compared with KnowPrompt in all-shot settings, indicating the effectiveness of the considerations on the properties of the dialogue with prompt-based fine-tuning. Except for the 8-shot setting, $\mathbf{GRASP}_{large}$ presents outstanding performance, achieving up to 15.3%p of absolute improvement in the 16-shot setting. Although $\mathbf{GRASP}_{large}$ outperforms the fine-tuning-based models and PTR, the limited performance of $\mathbf{GRASP}_{large}$ in the 8-shot setting can be attributed to an insufficient number of examples.

We also observe that the fine-tuning-based models, such as TUCORE-GCN, perform at least 5.7%p worse than the prompt-based fine-tuning models, such as PTR, in the 8-shot setting, indicating that the fine-tuning-based models may have difficulty in sufficiently capturing relational semantics when the data are extremely scarce. In particular, TUCORE-GCN indicates a 5.2%p lower performance than the fine-tuned RoBERTa, indicating that the high complexity requires a larger amount of data than the other models.

Method	Dev		Test	
	F1	$F1_c$	F1	$F1_c$
$\mathbf{GRASP}_{base}$	**70.3**	**63.3**	**69.0**	**61.7**
-APM	69.5	62.9	66.9	60.7
-RCD	69.2	62.8	67.7	61.0
-Prompt manual init.	68.1	61.9	65.9	59.8

Table 4: Ablation study on DialogRE dataset.

Ablation Study We conduct an ablation study to validate the effectiveness of the proposed modules. As shown in Table 4, each of the proposed modules improves the overall performance for both F1 and $F1_c$ settings. Without the APM strategy, the performance of $\mathbf{GRASP}_{base}$ decreases the F1 score by 0.8%p and the $F1_c$ score by 0.4%p on the development set, and the F1 score drops sharply by 2.1%p and $F1_c$ by 1.0%p on the test set. This result indicates that argument-awareness can be obtained through both speaker and non-speaker argument information. When the RCD task is excluded, the performance of $\mathbf{GRASP}_{base}$ decreases the F1 score by 1.1%p and $F1_c$ by 0.5%p on the development set and the F1 score by 1.3%p and $F1_c$ by 0.7%p for the test set. These results demonstrate that the RCD task alleviates the low information density of the dialogue by guiding the model to focus on relational clues.

In addition, the performance without the manual initialization of prompt construction of $\mathbf{GRASP}_{base}$ is reduced by 2.2%p for the F1 score and 1.4%p for the $F1_c$ score on the development set and by 3.1%p for the F1 score and 1.9%p for the $F1_c$ score on the test set. This result suggests that prompt construction is a basic step in training the model in the prompt-based fine-tuning case. The deliberate initialization of prompts is critical for modeling the tasks in an appropriate direction.

5 Analysis

5.1 Analysis on marker type for APM

To analyze argument-awareness regarding the types of markers, we conduct experiments on diverse prompt markers, as shown in Table 5. The result reveals that considering argument types leads to performance improvement in the model. Punctuation marker ";" shows comparable performance among other punctuation markers, and "!" and "@"

Marker Type	Input Example	F1
Entity marker	[CLS] [E1] **Frank** [/E1] lives in [E2] **Montauk** [/E2] . [SEP]	65.9
Type marker	[CLS] [E1:PER] **Frank** [/E1:PER] lives in [E2:GPE] **Montauk** [/E2:GPE] . [SEP]	66.7
Punctuation marker (!)	[CLS] ! **Frank** ! lives in ! **Montauk** ! . [SEP]	66.7
Punctuation marker (@)	[CLS] @ **Frank** @ lives in @ **Montauk** @ . [SEP]	65.3
Punctuation marker (;)	[CLS] ; **Frank** ; lives in ; **Montauk** ; . [SEP]	67.1
APM marker (front)	[CLS] [p] **Frank** lives in [p] **Montauk** . [SEP]	**69.0**
APM marker (surrounding)	[CLS] [p] **Frank** [p] lives in [p] **Montauk** [p] . [SEP]	65.9

Table 5: The performance based on the marker type. The arguments are bold. The embedding of [p] is initialized with the space token from the model's vocabulary. In type marker, [E1:PER] represents a start position of subject which has a person type and [/E1:PER] represents an end position of object that has the same type.

Dialogue
S1: Hey!!
S2: Hey!
S1: Guess what. **Frank Jr.**, and **Alice** got married!
S2: Oh my God!!
S1: And! And, they're gonna have a baby! And! And, they want me to grow it for them in my uterus.
S3: My God!
S4: Are you serious?
S1: Yeah
S5: You're really thinking about having sex with your brother?!
S1: Ewww! And "Oh no!" It's—they just want me to be the surrogate. It's her-it's her egg and her sperm, and I'm-I'm just the oven, it's totally their bun.
S5: Huh.

Argument pair	**Ground Truth**	**RoBERTa** Predicted Relation	**GRASP** Predicted (*subject, object, trigger*)	**GRASP** Predicted Relation
(Frank Jr., Alice)	*per:spouse*	unanswerable	(Frank Jr, Alice, got married)	per:spouse
(Alice, Frank Jr.)	*per:spouse*	per:siblings	(Alice, Frank Jr, got married)	per:spouse

Table 6: The qualitative analysis on the prediction of **GRASP** based on the comparison with the RoBERTa model. Predicted (*subject, object, trigger*) is that **GRASP** predicted on RCD task.

achieve a limited score. We presume that the higher frequency of ";" acts as a delimiter, which results in decent performance.

We also observe that the APM marker (front) performs the best, with a 69.0% F1 score among all other marker types. In addition, we conduct an experiment based on the position of the prompt marker, [p], by comparing two versions of the APM marker: APM marker (front) and APM marker (surrounding). The APM marker (surrounding) display 3.1%p lower performance than the APM marker (front). Based on these results, we empirically adopt the embedding initialization of our [p] prompt marker using the space token and located it in front of the arguments.

5.2 Qualitative Analysis on GRASP

Since we train **GRASP** attending on relational clues through the APM strategy and the RCD task, we further conduct analysis to investigate that the relational clues such as triggers contribute to predicting a relation between the arguments in a prompt-based manner, as shown in Table 6. Specifically, we compare $\mathbf{GRASP}_{large}$ with a fine-tuned RoBERTa-large model to validate our method.

We observe that the fine-tuned RoBERTa model struggles to capture the symmetrical relations including the trigger. The fine-tuned RoBERTa model fail to capture relational clues, such as "got married", misleading the model into predicting an inappropriate relations for both symmetrical relations between "Frank Jr." and "Alice." "got married" is a critical cue to distinguish the relations between "per:spouse" and "per:siblings" because this phrase implies a romantic relationship in a dictionary definition. In contrast, **GRASP**, which is trained using the RCD task and the APM strategy in a prompt-based manner, predicted the correct relations, capturing the correct relational clues including the trigger "got married" for both symmetrical argument pairs. This result presents the effectiveness of **GRASP** designed to guide the model on the relational clues, alleviating the difficulties of low information density in dialogues. Additional

Method	MELD	EmoryNLP
RoBERTa (Liu et al., 2019)	62.0	37.3
COSMIC (Ghosal et al., 2020)	65.2	38.1
TUCORE-GCN (Lee and Choi, 2021)	65.4	39.2
GRASP$_{large}$ (Ours)	**65.6**	**40.0**

Table 7: Experimental results of **GRASP**$_{base}$ on MELD and EmoryNLP tasks.

examples demonstrating similar phenomena for the symmetrical relations are provided in the Appendix B.

5.3 Analysis on the applicability of GRASP

To demonstrate the robustness of our APM strategy and RCD task, we evaluated **GRASP** on MELD (Poria et al., 2019) and EmoryNLP (Zahiri and Choi, 2018) datasets, which are designed for emotion recognition in conversations (ERC). MELD (Poria et al., 2019) is a multimodal dataset collected from a TV show named Friends and consists of seven emotion labels and 2,458 dialogues with only textual modality. EmoryNLP (Zahiri and Choi, 2018) is also collected from Friends and comprises seven emotion labels and 897 dialogues. Each utterance in these datasets is annotated with one of the seven emotion labels. The weighted-F1 is calculated to evaluate the MELD and EmoryNLP datasets.

For baselines, we employ the fine-tuned RoBERTa (Liu et al., 2019), COSMIC (Ghosal et al., 2020), and TUCORE-GCN models (Lee and Choi, 2021). COSMIC (Ghosal et al., 2020) uses RoBERTa-large as the encoder. It is a framework that models various aspects of commonsense knowledge by considering mental states, events, actions, and cause-effect relations for emotional recognition in conversations.

As presented in Table 7, **GRASP** is applied to other dialogue-based tasks by alleviating the low information density of the given dialogue. In particular, the performance of **GRASP** surpasses that of TUCORE-GCN, which is the current SoTA model in DialogRE, and those of the baseline specialized on ERC tasks, such as COSMIC in both MELD and EmoryNLP.

6 Conclusion

In this paper, we proposed **GRASP**, which is a method for guiding PLMs to relational semantics using prompt-based fine-tuning for the DialogRE task. We focused on alleviating the critical challenge in dialogues, that is, low information density, by effectively capturing relational clues. In **GRASP**, we constructed prompts with deliberate initialization and suggested 1) an APM strategy considering both speaker and non-speaker arguments and 2) the RCD task, which guides the model to determine which token belongs to the relational clues. Experimental results on the DialogRE dataset revealed that **GRASP** achieved SoTA performance in terms of the F1 and F1$_c$ scores, even though our method only leveraged a PLM without adding any extra layers.

Acknowledgements

This work was supported by Institute of Information & communications Technology Planning & Evaluation (IITP) grant funded by the Korea government (MSIT) (No. 2020-0-00368, A Neural-Symbolic Model for Knowledge Acquisition and Inference Techniques), the MSIT (Ministry of Science and ICT), Korea, under the ITRC (Information Technology Research Center) support program (IITP-2022-2018-0-01405) supervised by the IITP, and the MSIT, Korea, under the ICT Creative Consilience program (IITP-2022-2020-0-01819) supervised by the IITP.

References

Xuefeng Bai, Yulong Chen, Linfeng Song, and Yue Zhang. 2021. Semantic representation for dialogue modeling. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 4430–4445, Online. Association for Computational Linguistics.

Laura Banarescu, Claire Bonial, Shu Cai, Madalina Georgescu, Kira Griffitt, Ulf Hermjakob, Kevin Knight, Philipp Koehn, Martha Palmer, and Nathan Schneider. 2013. Abstract Meaning Representation for sembanking. In *Proceedings of the 7th Linguistic Annotation Workshop and Interoperability with Discourse*, pages 178–186, Sofia, Bulgaria. Association for Computational Linguistics.

Douglas Biber. 1991. *Variation across speech and writing*. Cambridge University Press.

Xiang Chen, Ningyu Zhang, Xin Xie, Shumin Deng, Yunzhi Yao, Chuanqi Tan, Fei Huang, Luo Si, and Huajun Chen. 2021. Knowprompt: Knowledge-aware prompt-tuning with synergistic optimization for relation extraction. *arXiv preprint arXiv:2104.07650*.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, Minneapolis, Minnesota. Association for Computational Linguistics.

Tianyu Gao, Adam Fisch, and Danqi Chen. 2020. Making pre-trained language models better few-shot learners. *arXiv preprint arXiv:2012.15723*.

Deepanway Ghosal, Navonil Majumder, Alexander Gelbukh, Rada Mihalcea, and Soujanya Poria. 2020. COSMIC: COmmonSense knowledge for eMotion identification in conversations. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 2470–2481, Online. Association for Computational Linguistics.

Xu Han, Weilin Zhao, Ning Ding, Zhiyuan Liu, and Maosong Sun. 2021. Ptr: Prompt tuning with rules for text classification. *arXiv preprint arXiv:2105.11259*.

Heng Ji, Ralph Grishman, Hoa Trang Dang, Kira Griffitt, and Joe Ellis. 2010. Overview of the tac 2010 knowledge base population track. In *Third text analysis conference (TAC 2010)*, volume 3, pages 3–3.

Thomas N Kipf and Max Welling. 2016. Semi-supervised classification with graph convolutional networks. *arXiv preprint arXiv:1609.02907*.

Bongseok Lee and Yong Suk Choi. 2021. Graph based network with contextualized representations of turns in dialogue. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 443–455, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Xiang Lisa Li and Percy Liang. 2021. Prefix-tuning: Optimizing continuous prompts for generation. *arXiv preprint arXiv:2101.00190*.

Xiao Liu, Yanan Zheng, Zhengxiao Du, Ming Ding, Yujie Qian, Zhilin Yang, and Jie Tang. 2021. Gpt understands, too. *arXiv preprint arXiv:2103.10385*.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Roberta: A robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692*.

Xinwei Long, Shuzi Niu, and Yucheng Li. 2021. Consistent inference for dialogue relation extraction. In *Proceedings of the Thirtieth International Joint Conference on Artificial Intelligence, IJCAI-21*, pages 3885–3891. International Joint Conferences on Artificial Intelligence Organization. Main Track.

Ilya Loshchilov and Frank Hutter. 2017. Decoupled weight decay regularization. *arXiv preprint arXiv:1711.05101*.

Soujanya Poria, Devamanyu Hazarika, Navonil Majumder, Gautam Naik, Erik Cambria, and Rada Mihalcea. 2019. MELD: A multimodal multi-party dataset for emotion recognition in conversations. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 527–536, Florence, Italy. Association for Computational Linguistics.

Timo Schick and Hinrich Schütze. 2021. Exploiting cloze-questions for few-shot text classification and natural language inference. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*, pages 255–269, Online. Association for Computational Linguistics.

Livio Baldini Soares, Nicholas Fitzgerald, Jeffrey Ling, and Tom Kwiatkowski. 2019. Matching the blanks: Distributional similarity for relation learning. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 2895–2905.

Kumutha Swampillai and Mark Stevenson. 2010. Intersentential relations in information extraction corpora. In *Proceedings of the Seventh International Conference on Language Resources and Evaluation (LREC'10)*.

Dong Wang and Yang Liu. 2011. A pilot study of opinion summarization in conversations. In *Proceedings of the 49th annual meeting of the Association for Computational Linguistics: Human language technologies*, pages 331–339.

Fuzhao Xue, Aixin Sun, Hao Zhang, and Eng Siong Chng. 2021. Gdpnet: Refining latent multi-view graph for relation extraction. In *Thirty-Fifth AAAI Conference on Artificial Intelligence, AAAI*, pages 2–9.

Yuan Yao, Jiaju Du, Yankai Lin, Peng Li, Zhiyuan Liu, Jie Zhou, and Maosong Sun. 2021. Codred: A cross-document relation extraction dataset for acquiring knowledge in the wild. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 4452–4472.

Dian Yu, Kai Sun, Claire Cardie, and Dong Yu. 2020. Dialogue-based relation extraction. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 4927–4940, Online. Association for Computational Linguistics.

Sayyed M Zahiri and Jinho D Choi. 2018. Emotion detection on tv show transcripts with sequence-based convolutional neural networks. In *Workshops at the thirty-second aaai conference on artificial intelligence*.

A Detailed experimental settings

Hyper-parameters	$\mathbf{GRASP}_{base}$	$\mathbf{GRASP}_{large}$
Learning rate	$5e-5$	$5e-6$
Max seq. len	512	
Batch size	8	
Num. epochs	30	
Joint ratio (λ_1 & λ_2)	0.7 / 0.3	

Table 8: Hyper-parameter values used in prompt-tuning process on test set.

GRASP is trained using AdamW (Loshchilov and Hutter, 2017) as an optimizer with no weight decay. The number of training epochs is set to 30 with early stopping, and the ratio of λ_1 and λ_2 for the joint loss is 0.7 to 0.3. A learning rate of $5e{-}5$, batch size of 8, and maximum sequence length of 512 are adopted for RoBERTa-base with identical parameters for RoBERTa-large, except for the learning rate of $5e{-}6$.

B Qualitative Analysis Examples

Table 9 shows additional examples to demonstrate the prediction tendency of **GRASP** and the fine-tuned RoBERTa models on the symmetrical relations described in Section 5.2.

Dialogue

S1: Hey! Hi!
S2: Hey!
S1: What's up?
S2: Well umm, **Chandler** and I are moving in together.
S1: Oh my God. Ohh, my little sister and my best friend. . . shaking up. Oh, that's great. That's great.
S3: Guys, I'm happy too.
S2: Okay, come here!
S3: Wow! Big day huh? People moving in, people getting annulled. . .
S2: Okay, I gotta go find Rachel but umm, if you guys see her could you please try to give her some really bad news so that mine doesn't seem so bad?
...

Argument pair	**Ground Truth**	**RoBERTa** Predicted Relation	**GRASP** Predicted (*subject, object, trigger*)	Predicted Relation
(**S2**, **Chandler**)	per:girl/boyfriend	per:roommate	(S2, Chandler, moving in together)	per:girl/boyfriend
(**Chandler**, **S2**)	per:girl/boyfriend	per:roommate	(Chandler, S2, moving in together)	per:girl/boyfriend

Dialogue

S1: Rach, Rach, I just remembered. I had a dream about **Mr. Geller** last night.
S2: Really?!
S1: Yeah, I dreamt that he saved me from a burning building and he was so brave and so strong! And it's making me look at him totally differently. Y'know, I mean he used to be just, y'know "Jack Geller **Monica** and **Ross**'s dad" and now he's he's "Jack Geller, dream hunk."
S2: I dunno. Y'know to me he'll always be "Jack Geller, walks in while you're changing."

Argument pair	**Ground Truth**	**RoBERTa** Predicted Relation	**GRASP** Predicted (*subject, object, trigger*)	Predicted Relation
(Mr. Geller, Monica)	*per:children*	unanswerable	(Mr Geller, Monica, dad)	per:children
(Mr. Geller, Ross)	*per:children*	unanswerable	(Mr Geller, Ross, dad)	per:children
(Ross, Mr. Geller)	*per:parents*	per:alternate_names	(Ross, Mr Geller, dad)	per:parents

Table 9: The additional examples for qualitative analysis on the prediction of **GRASP** based on the comparison with the RoBERTa model. Predicted (*subject, object, trigger*) is that **GRASP** predicted on RCD task.

PEPDS: A Polite and Empathetic Persuasive Dialogue System for Charity Donation

Kshitij Mishra, Azlaan Mustafa Samad, Palak Totala, Asif Ekbal *
Department of Computer Science and Engineering
Indian Institute of Technology Patna
Patna, India
{kshitij_1921cs23,totala_2001cs84,asif}@iitp.ac.in
azlaanmsamad@gmail.com

Abstract

Persuasive conversations for a social cause often require influencing other person's attitude or intention that may fail even with compelling arguments. The use of emotions and different types of polite tones as needed with facts may enhance the persuasiveness of a message. To incorporate these two aspects, we propose a polite, empathetic persuasive dialogue system (**PEPDS**). First, in a Reinforcement Learning (RL) setting, a Maximum Likelihood Estimation loss based model is fine-tuned by designing an efficient reward function consisting of five different sub rewards *viz.* Persuasion, Emotion, Politeness-Strategy Consistency, Dialogue-Coherence and Non-repetitiveness. Then, to generate empathetic utterances for non-empathetic ones, an Empathetic transfer model is built upon the RL fine-tuned model. Due to the unavailability of an appropriate dataset, by utilizing the PERSUASIONFORGOOD dataset, we create two datasets, *viz.* **EPP4G** and **ETP4G**. **EPP4G** is used to train three transformer-based classification models as per persuasiveness, emotion and politeness-strategy to achieve respective reward feedbacks. The **ETP4G** dataset is used to train an empathetic transfer model. Our experimental results demonstrate that PEPDS increases the rate of persuasive responses with emotion and politeness acknowledgement compared to the current state-of-the-art dialogue models, while also enhancing the dialogue's engagement and maintaining the linguistic quality [1].

1 Introduction

A persuasive message can be initially analyzed by distinguishing between the **cause or stimulus** the persuadee is being persuaded for and the associated **attitude** shown during persuasion. The message is first presented to the persuadee; s/he pays attention to it and comprehends its contents. A persuadee's response depends partly on the information provided in the message and how one perceives or interprets it. Empathy plays a crucial role in mediating the persuasive effects as it evokes cognitive and emotional processing conducive to persuasion. Similarly, use of different polite tones as per context may establish a better connection with the persuadee and engage them for a longer time in the ongoing conversation. Therefore, for persuasion to be effective, **cause or stimulus** of persuasion can be encoded with **empathy** and **politeness**, ensuring interactiveness, empathetic connection, right tone as well as user engagement in a persuasive message.

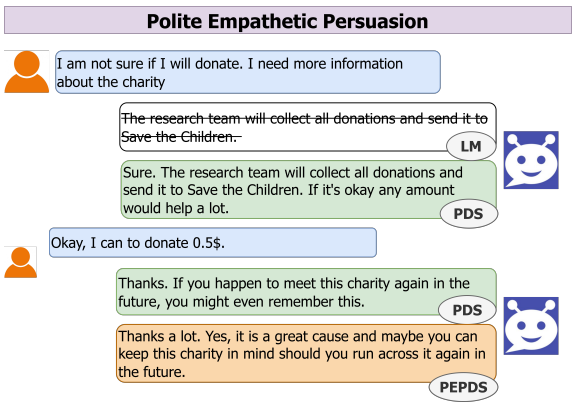

Figure 1: An example of persuasion with LM (Language Model), PDS (LM fine-tuned with RL), and PEPDS (PDS with empathetic transfer model).

A persuasive message consists of some form of directives, hence having the potential for disagreement or dissatisfaction thus may threaten the face of recipients. Hence, messages employing different politeness strategies such as positive politeness and negative politeness as per user's mood and message content may emphasize users' freedom to get persuaded or not. For an example, in Figure 1, the strike through response is persuasive but also consists of forceful language. Whereas, the

* Corresponding author: asif@iitp.ac.in.

[1]The resources and codes can be accessed at PEPDS_ai_nlp_ml or at PEPDS_github.

Proceedings of the 29th International Conference on Computational Linguistics, pages 424–440
October 12–17, 2022.

green box response connects with the user empathetically and also gives him/her freedom of choice by adopting positive politeness strategy (i.e. by using non-imposing, face-saving language). Further, to acknowledge underlying emotion in the content an empathetic transfer model can transfer the non-empathetic messages to empathetic messages as shown in orange box of Figure 1.

Paucity of available data poses a challenge to model a persuasive dialogue system in a supervised learning (SL) setting. Further, modelling persuasion considering different attributes such as politeness and empathy brings a greater challenge due to its different dynamics with different users. Therefore, a model which can learn from user interactions itself in an ongoing dialogue is required. Recently, researchers had widely used reinforcement learning (RL) to reinforce dialogue agents to approximate better policy as per user's feedbacks (Singh et al., 1999; Li et al., 2016; Chen et al., 2019; Mesgar et al., 2021; Mishra et al., 2022). Hence, to induce both, a consistent politeness-strategy and empathy, we utilize goodness of both the frameworks i.e. RL and SL, and propose a novel **P**olite-**E**mpathetic **P**ersuasive **D**ialogue **S**ystem (**PEPDS**).

To ensure persuasiveness, politeness-strategy consistency, right emotion, dialogue coherence and non-repetitiveness, a maximum likilehood estimation loss (MLE) language based model is fine-tuned with an RL loss considering five rewards *viz.* Persuasion, Emotion, Politeness-strategy Consistency, Dialogue-coherence and Repetitiveness. In case the generated persuasive messages are found to be non-empathetic, a sequence-to-sequence (seq2seq) based style transfer model is employed to make them empathetic. In order to achieve this goal, we prepare a new dataset named as **ETP4G** from the EPP4G dataset. To obtain persuasion, emotion and politeness-strategy consistency reward feedbacks, first, we annotate PERSUASIONFORGOOD (P4G) dataset (Wang et al., 2019) with the required emotion and politeness-strategy labels, and obtain a new dataset named as **EPP4G**. Then, considering these three aspects, we fine-tune three RoBERTa-large (Liu et al., 2019) based respective classifiers.

Recently, to generate persuasive responses, an MLE-loss based language model is fine-tuned by Shi et al. (2021) with an RL-loss without using any user simulators. To persuade the persuadee, they penalized the generation of repetitive and inconsistent utterances. Our work differs from them in three aspects. Firstly, we consider new aspects of politeness-strategy consistency and emotion acknowledgement to force the dialogue agent to be consistent as well as empathetic. Second, a new reward function is designed to ensure right persuasion strategy, emotion acknowledgement, politeness-strategy consistency, dialogue-coherence in an ongoing dialogue and non-repetition of similar responses. Third, to connect with the user empathetically, we transfer the non-empathetic responses to empathetic ones. Lastly, we perform automatic and human evaluation to assess the persuasiveness, politeness-strategy consistency, empathy, response-length, fluency, adequacy, consistency and non-repetitiveness of the generated responses for our proposed **PEPDS**. Following are the key contributions we present in this paper:

1. To have PERSUASIONFORGOOD dataset utterances with politeness-strategy and empathy information, we manually annotate it with three different labels to obtain **EPP4G** and **ETP4G** datasets, respectively.

2. We build an empathetic transfer model by utilizing pre-trained and fine-tuned transformer models.

3. We propose a polite empathetic persuasive dialogue system (**PEPDS**) by designing an efficient reward function to ensure politeness-strategy consistency, persuasiveness, emotion acknowledgement, dialogue-coherence and non-repetitiveness.

4. We perform detailed empirical evaluation considering automatic and human evaluation to demonstrate robustness of our proposed system **PEPDS**.

2 Related Work

Recent research on personalised conversational agents shows that incorporating various human-oriented conversational strategies can have a significant impact on the user responses and make the conversations more engaging (Mazare et al., 2018; Kocaballi et al., 2019; Wang et al., 2019; Dutt et al., 2020; Song et al., 2021). These dialogue agents can better acknowledge the user's state to adapt themselves as per user's need. For instance, Bertero et al. (2016) tries to recognize user emotions in real-time in an interactive dialogue system. Shi

and Yu (2018) builds an effective dialogue system considering user's sentiment information, whereas Firdaus et al. (2020) utilizes multimodal information to generate sentiment and emotion controlled responses. Golchha et al. (2019) induces courteous behaviour in customer care responses. Similarly, Mishra et al. (2022) designed three politeness based rewards to reinforce polite responses in an ongoing task-oriented dialogue. Due to subtle dependency between these user-targeted personalization techniques such as politeness-strategy, empathy or sentiment, and persuasion, we focus on incorporating politeness-strategy consistency and empathy to generate more persuasive and engaging utterances.

Historically, different persuasion models have been proposed, such as Petty and Cacioppo's Elaboration Likelihood Model (ELM) (Petty and Cacioppo, 1986) and Friestad and Wright's Persuasion Knowledge Model (PKM) (Friestad and Wright, 1994). This indicates that a person's persuasion may depend on several factors, such as the content of the message, context, common sense knowledge and scientific knowledge. Similarly, Dijkstra (2008) states that personal factor consideration with informative content can enhance the quality of persuasive messages. Bohner et al. (2002); Sparks and Areni (2002, 2008) point out that the language employed in persuasive messages can impact both perceptions of the persuader as well as the message conveyed. Similarly, Brown and Levinson (1978); Brown et al. (1987) in their politeness theory points out that persuasive messages in the absence of indirect or face-saving language may commensurate threat in the face of recipients.

Recent research studies have started focusing on building persuasive dialogue agents by considering different aspects, such as persuasion strategies (Wang et al., 2019) and resistive strategies (Dutt et al., 2021). Shi et al. (2020) conducted an online study considering 790 participants to check if they can be persuaded or not by a chatbot. Further, the availability of transformer based pre-trained language models has led the researchers to train style transfer even in the absence of parallel data (Yang et al., 2018; Xu et al., 2018; He et al., 2020; Goyal et al., 2021; Lai et al., 2021; Malmi et al., 2020; Jin et al., 2022). Our current method of transferring non-empathetic utterance to empathetic is based on the prior technique proposed in (Krishna et al., 2020).

Recently, two MLE-loss based language models - one for persuadee and one for persuader are jointly trained to generate persuasive responses (Wu et al., 2021). Shi et al. (2021) fine-tunes a language model in RL-setting to persuade with non-repetitive and consistent responses. In contrast to the fact that persuasion encompasses a vast domain space with different associated attitudes, these research studies either concentrated on predicting persuasion strategy or generating persuasive responses alone. Persuasion alone cannot ensure user engagement; a dialogue agent should be able to adapt to different associated attitudes as per rapport built with the user. Therefore, our work here focuses on adapting different politeness strategies as per user attitudes and generating empathetic and engaging persuasive dialogues. To the best of our knowledge, no previous study has incorporated politeness strategies to build a persuasive dialogue system.

3 Methodology

A persuasive dialogue $d = \{u_1^{er}, u_1^{ee}, u_2^{er}, u_2^{ee}, ..., u_{T-1}^{er}, u_{T-1}^{ee}\}$ consists of $T-1$-turns, where u_i^{er} and u_i^{ee} represents the persuader's and persuadee's utterance, respectively, at i^th turn ($1 < i <= T-1$). Considering this ongoing dialogue as a context, our goal is to generate an adequate dialogue coherent persuader's utterance $u_T^{ee} = \{t_1, t_2, ..., t_r\}$ at turn T with r number of tokens. To achieve this goal, first, an MLE-loss based language model is fine-tuned with an RL-loss by designing an efficient reward function to generate rich emotion acknowledged and politeness-strategy consistent persuasive responses. On generating the non-empathetic utterances, it is passed through an empathetic transfer model to transfer to an empathetic utterance.

3.1 Classifiers

All the four classifiers, *viz.* persuasion strategy, emotion, politeness-strategy and empathy are built by fine-tuning RoBERTa-large (Liu et al., 2019) pre-trained model. Persuasion strategy, Emotion and Politeness strategy classifiers are used to provide reward feedbacks, whereas empathy classifier is used to inform if the generated utterance is empathetic or non-empathetic. In a dialogue, for m number of persuader's utterance, each utterance has four labels, *viz.* politeness-strategy $pos = \{pos_1^l, pos_2^l, ..., pos_m^l\}$, emotion $emo = \{emo_1^l, emo_2^l, ..., emo_m^l\}$ and per-

suasion strategy $ps = \{ps^l_1, ps^l_2, ..., ps^l_m\}$ and $em = \{em^l_1, em^l_2, ..., em^l_m\}$. The sets $\mathbb{POS} = \{0, 1, 2\}$, $\mathbb{EMO} = \{emo^{l_1}, emo^{l_2}, ..., emo^{l_{n_2}}\}$ $\mathbb{PS} = \{ps^{l_1}, ps^{l_2}, ..., ps^{l_{n_1}}\}$ and $\mathbb{EMP} = \{0, 1\}$ give the different classes for each of the following: politeness-strategy, emotion, persuasion-strategy and empathy, where n_1 and n_2 denote the number of persuasion strategy and emotion labels. Due to space restrictions, we provide further details in Section A.1 of the appendix.

3.2 Empathetic Transfer Model

We build the Empathetic Transfer (ET) model, based on BART (Lewis et al., 2020), following a standard seq2seq transformer based architecture. It comprises of a bidirectional encoder (like, BERT) and a left-to-right decoder (like, GPT). As in seq2seq transformers, each layer of BART's decoder performs cross-attention over the final hidden layer of the encoder. To build our model, we fine-tune the pre-trained BART-large (Lewis et al., 2020). As BART comprises of an auto-regressive decoder, it can be directly fine-tuned in the form of a sequence-to-sequence problem, where the input text corresponds to a non-empathetic utterance and the output corresponds to an empathetic utterance. The model is trained in an end-to-end fashion in two steps. First, the input text is corrupted using an arbitrary noising function. Second, it optimises a reconstruction loss i.e. cross-entropy between the decoder's output and the ground truth output.

3.3 Reinforcement Learning Fine Tuning

An MLE-based language model generates *n*-candidate responses for a given input. These candidates are evaluated in terms of quality with respect to gold human response using the designed reward function. The candidates with the right persuasion, emotion, and politeness strategies are rewarded, while non-coherent and repetitive utterances are penalized. The RL-system is trained using the proximal policy optimization (PPO) (Schulman et al., 2017) method to achieve optimal policy.

Reward: To capture each aspect of the persuasion, underlying emotion, politeness-strategy consistency, dialogue-coherence and non-repetitiveness, a single reward function R is designed consisting of all the four sub-rewards: R_1 for persuasion, R_2 for emotion, R_3 for politeness-strategy consistency, R_4 for dialogue coherence and R_5 for non-repetitiveness. The reward, R can be expressed as a weighted sum of all these four sub-rewards.

$$R = \gamma_1 R_1 + \gamma_2 R_2 + \gamma_3 R_3 + \gamma_4 R_4 + \gamma_4 R_5 \quad (1)$$

Persuasion, Emotion and Politeness-Strategy Consistency: Persuasion strategy (R_1), emotion (R_2) and politeness-strategy (R_3) reward feedbacks are achieved by passing generated utterance r_T through persuasion strategy, emotion and politeness-strategy classifiers, respectively. Then, each of the three predicted labels is compared with the respective gold human response label, and the candidates with matching label are rewarded.

$$R_1 = \mathcal{P}_{ps}(u^{er}_T) - \beta \sum_{i \in S} \mathcal{P}_{ps_i}(r_T) \quad (2)$$

$$R_2 = \mathcal{P}_{emo}(u^{er}_T) - \beta \sum_{i \in S} \mathcal{P}_{emo_i}(r_T) \quad (3)$$

$$R_3 = \mathcal{P}_{pos}(u^{er}_T) - \beta \sum_{i \in S} \mathcal{P}_{pos_i}(r_T) \quad (4)$$

where $\mathcal{P}_{ps}(u^{er}_T)$, $\mathcal{P}_{emo}(u^{er}_T)$ and $\mathcal{P}_{pos}(u^{er}_T)$ denote the persuasion, emotion and politeness-strategy probabilities of the gold response u^{er}_T. The $\mathcal{P}_{ps_i}(r_T)$, $\mathcal{P}_{emo_i}(r_T)$ and $\mathcal{P}_{pos_i}(r_T)$ denotes the predicted persuasion, emotion and politeness-strategy probabilities of the generated response r_T, and $i \in \mathcal{S}$ with $\mathcal{S} = \{l_1, l_2, ..., l_n\}$ is the set of all the classes respective to persuasion strategy and politeness-strategy. β is a scalar value [2] acting as a penalization factor, i.e. increasing β will result into greater penalization of the generated response.

Dialogue-coherence Reward: In order to force the agent to generate human-like responses in an ongoing dialogue, the Meteor score (Banerjee and Lavie, 2005) is calculated between the generated response r_T (hypothesis) and the gold human response $p^r_{g_t}$ (reference). The more the meteor score is the more the generated utterance is semantically similar to the gold human response.

$$R_4 = MET(r_T, p^r_{g_t}) \quad (5)$$

Meteor score is selected as it shows high correlation with human judgement in machine translation tasks (Banerjee and Lavie, 2005) by leveraging WordNet, in case an exact match of tokens is not found (Castillo and Estrella, 2012). This leads to better generalization as language model tend to generate semantically similar responses but different in terms of morphemes.

[2] The value of β is taken as greater than or equal to 1

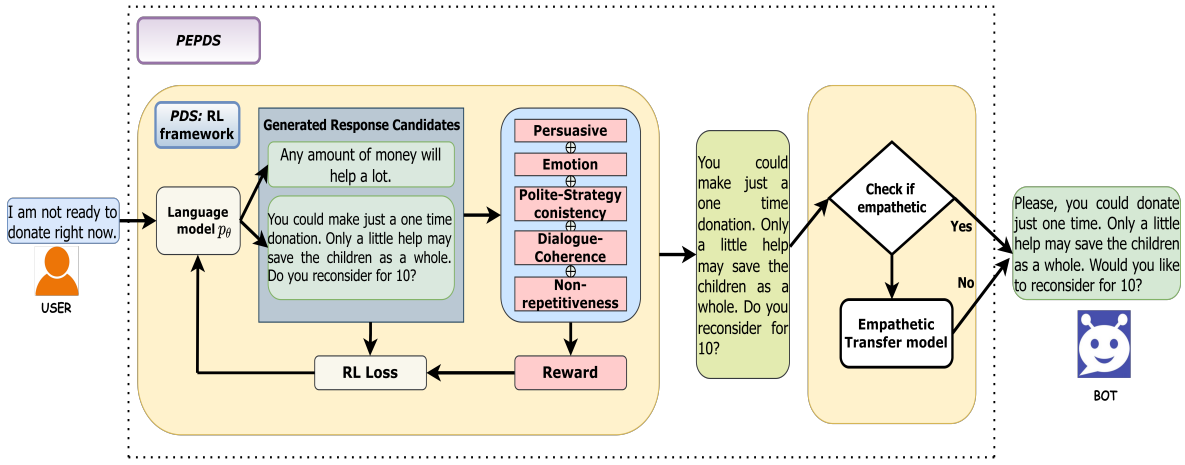

Figure 2: A skeleton of the overall proposed system PEPDS. First, it is initialised with MLE-loss based language model (LM) parameters p_θ. Second, this trained LM is fine-tuned, considering RL-loss to build a persuasive dialogue system (PDS). Lastly, PEPDS using the Empathetic Transfer model generates empathetic utterances corresponding to non-empathetic generated utterances by PDS.

Repetitiveness Reward: It is found that repetitions usually happens only on the lexical level in this task (Shi et al., 2021). Therefore, to penalize the repetitions of same responses in a dialogue we use Jaccard similarity score between the previous generated r_{T-1} and current generated response r_T.

$$R_5 = \frac{r_{T-1} \cap r_T}{r_{T-1} \cup r_T} \tag{6}$$

Policy: In an RL framework, a policy models the agent's action selection as a probability mapping function. Hence, policy $\mathcal{P}_\theta$ representing the probability of generating an utterance r consisting of L tokens can be formulated as:

$$\mathcal{P}_\theta(r_{1:L}|x) = \prod_{l=0}^{L} \mathcal{P}_\theta(r_l|y_{<l}, x) \tag{7}$$

Proximal Policy Optimisation: To ensure low variance, proximal policy optimisation (PPO) method (Schulman et al., 2017) is chosen to update the policy at each step. It updates an existing policy to seek improvement on certain parameters such that the old policy is not too different from the new policy. Policy optimisation mainly comprises of three steps. First, to maximize the expected reward, gradient ascent is applied on loss function $J(\theta)$,

$$\nabla_\theta J(\theta) = \mathbb{E}_{r\sim\mathcal{P}_\theta}[\nabla_\theta \log \mathcal{P}_\theta(r)\hat{A}_r] \tag{8}$$

Second, PPO restricts large deviations from the old policy in the above equation, by replacing the log term with an importance sampling term with clipping to prevent catastrophic forgetting. Here, the clipped variant of PPO neither have any KL-divergence term in the objective function nor any constraint instead, it relies on specialized clipping in the objective function. It is formulated as given below:

$$L^{\text{CLIP}}(\theta) = \hat{\mathbb{E}}[\min(pr_r(\theta)\hat{A}_r, \text{clip}(pr_y(\theta), 1-\varepsilon, 1+\varepsilon)\hat{A}_r)]$$

Here, $pr_r(\theta) = \mathcal{P}_\theta^{new}/\mathcal{P}_\theta^{old}$ denotes the probability ratio of generating a response between the new and old policies. $\hat{A}_r$ gives the estimated advantage which, here equals to the normalized rewards and ε represents the clipping range. Lastly, parameters are updated using the following steps:

$$\theta_{k+1} = \underset{\theta}{\text{argmax}} \underset{s,a\sim\mathcal{P}_{\theta_k}}{\mathbb{E}} [L^{\text{CLIP}}] \tag{9}$$

3.4 Proposed Method

We employ the attributes of emotion and politeness-strategy consistency to control the agent to generate engaging, polite, emotionally acknowledged, and persuading responses. First, we fine-tune an MLE-based language model, ARDM, in an RL setting considering five rewards. Then, we use this trained model to generate responses at inference time which are transferred to empathetic responses by an auto-regressive seq2seq model. The overall architecture of the proposed system is shown in Figure 2.

Dataset	Number of utterances					
	All	Persuader's	Persuadee	train	eval	test
P4G (to train LM)	20932	10600	10332	16746	2093	2093
P4G (persuasion strategy)	10864	6018	4846	4814	602	602
EPP4G (emotion)	4000	4000	-	3200	400	400
EPP4G (politeness-strategy)	5300	5300	-	4240	530	530
ETP4G (empathetic transfer)	16722	16722	-	13378	1672	1672

Table 1: Data statistics of all datasets. Here, train, eval and test correspond to the number of utterances used to train, validate and test the respective models. Further '-' signifies that either persuadee or persuader's role is not considered or that role's utterances are absent in the dataset.

3.4.1 Baselines

We define the following baselines in our current work.

ARDM: To model both persuadee and persuader separately, ARDM jointly trains two GPT-2 (Radford et al., 2019) models. This model has reported to have achieved good performance levels (Wu et al., 2021). The RL-model is initialized with p_θ parameters of the ARDM model.

RFI: To learn policy directly from the data, it fine-tunes the ARDM in an RL-framework. Then using the response filter, response detector and response imitator, it selects the best response out of all the available candidate responses (Shi et al., 2021). RFI achieved state-of-the-art results considering ARDM as baseline. (Shi et al., 2021).

4 Datasets and Experiments

The datasets utilised in our studies are introduced in Section 4.1. In Section 4.2, implementation details of the proposed RL-based system are discussed. Due to space constraints, the classifiers' and empathetic transfer model's implementation details are provided in Section A.3 of the appendix. Finally, Section 4.3 describes the specifics of automatic and human evaluation metrics.

4.1 Datasets

To design our experiments, we use PERSUASIONFORGOOD (P4G) dataset. It consists of 1,017 persuasive conversations for donation to a charity organization *Save the Children*. These conversations are carried out between two humans, where one acted as a persuadee and the other as persuader. Each of the persuader's utterances in P4G dataset is grounded in one of the 11 persuasion strategies.

To have emotion and politeness strategy information for all utterances in PERSUASIONFORGOOD, we annotate it with both of these aspects and name it as **EPP4G** dataset. Then, we prepare a empathetic transfer dataset - **ETP4G** utilising the annotated **EPP4G** dataset. Due to space restrictions, annotation details for both the datasets are provided in Section A.2 of appendix. All the datasets statistics can be found in Table 1.

We also use the P4G dataset to train two classifiers: a persuasion strategy classifier and a persuasive binary classifier. While the former is used to provide persuasion reward, the latter is used to evaluate persuasiveness of generated utterance.

4.2 Implementation Details

To model both the persuader and the persuadee, GPT-2 medium model (Radford et al., 2019) is employed to train the language model. This language model is fine-tuned in an RL-setting, by experimenting with different candidate responses i.e. $n = 2, 3, 4, 5, 10$. Finally, $n = 2$ is chosen as the final value. Further, the generated candidates are decoded adopting nucleus sampling (Holtzman et al., 2019) with temperature $T = 0.8$ and probability $p = 0.9$. To train the proposed RL system $human\_reward = 10$, $max\_candidate\_length = 50$, and AdamW (Loshchilov and Hutter, 2017) optimizer is chosen with a learning rate of $\alpha = 2e{-}05$ and $\varepsilon = 0.2$. After performing several experiments with different values of the reward weights, $0.3, 0.3, 0.2, 0.1, 0.1$ are chosen as the final weights for γ_1, γ_2, γ_3, γ_4 and γ_5, respectively (detailed weight optimization is given in Section A.4 of the appendix). Lastly, for persuasion, emotion and politeness-strategy consistency rewards, the penalization factor β is set to 2.

4.3 Evaluation Metrics

We evaluate our proposed system considering both automatic and human evaluation metrics.

All the four classifiers are evaluated in terms of Weighted Accuracy (W-ACC) and Macro-F1 (to account for imbalanced class distribution). Further, empathetic transfer model is evaluated in terms of

Perplexity (PPL) (Brown et al., 1992), Bleu score (BLEU) (Papineni et al., 2002), METEOR score (MET) (Banerjee and Lavie, 2005), Rogue-2 F-1 score (R-2-F1) (Lin and Hovy, 2003), NIST score (Doddington, 2002) (NIST) and Empathy Accuracy (EM-ACC). The Empathetic Transfer Model is a sequence-to-sequence model, hence its performance has also been assessed using the BLEU metric, which measures how well the expected response correlates with the actual response. The anticipated response might be semantically same, but it might differ from the ground truth response in terms of the words that are true. To be sure of this, we additionally incorporate the METEOR score while evaluating the model.

Since the goal of the proposed system is to generate a persuasive response effectively, we evaluate our system in terms of four metrics, *viz.* **PerStr** - percentage of the utterances generated with persuasion strategy, **PolSt** - percentage of utterances generated with consistent politeness strategy as per ongoing dialogue, **Emp** - percentage of empathetic utterance generated, **PPL** - perplexity of the dialogue agent and **LEN** - number of tokens generated in an utterance. **PerStr** and **Emp** are evaluated by building two binary classifiers, i.e. a persuasive classifier predicts if a response comprises of persuasion strategy or not, and empathy classifier predicts if it is empathetic or not. In contrast to the persuasion strategy classifier, which predicts one of the 11 persuasion strategies, the binary persuasive classifier predicts the persuasiveness of the response as 0 or 1. **PolStr** is evaluated by comparing the predicted politeness strategy with the gold response politeness strategy.

We perform human evaluation by engaging three human experts (regular employees in our research group) with postgraduate experience and having proficiency in a similar task. They were asked to evaluate 40 generated persuasive dialogues in terms of **Per**, **Emp** - checking persuasiveness and empathy of the generated dialogue; **DonPr** - computing percentage of time people donated; **Const**, **Adeq**, **Fluen** and **N-Rep** to evaluate if the generated utterances are consistent (with the dialogue context), adequate, linguistically fluent and non-repetitive in nature [3]. Lastly, in order to take politeness-strategy consistency into account, we also evaluate our proposed model based on **Pol-Con** - denoting politeness-strategy consistency of persuader in an ongoing dialogue on 1-3 scale [4]. To get final values for each of the evaluation metric, the average of all three ratings given by human evaluators is taken [5].

5 Results and Analysis

We first analyse the results of our sub-modules used in our proposed system *viz.* all the four classifiers and the empathetic transfer model. Then, we analyse our proposed system by comparing it with two baselines: ARDM (MLE loss based model) (Wu et al., 2021) and RFI (DialGAIL with response filter and response imitator) (Shi et al., 2021). Automatic and human evaluation results of our proposed system are shown in Table 4 and Table 5, respectively.

Classifiers: Evaluation results of all the four classifiers are shown in Table 2. It can be observed that all the four classifiers achieve significantly good scores in terms of both W-ACC and Macro-F1. Further, it is also observed that RoBERTa-large (Liu et al., 2019) performs better than BERT-large (Kenton and Toutanova, 2019) on both the metrics.

Empathetic transfer model: Evaluation results of our proposed empathetic transfer model are shown in Table 3. It can be seen that BART-large (Lewis et al., 2020) performs better in terms of EM-ACC with a margin of 3% as compared to BERT-BERT seq2seq model. A good EM-ACC score implies that our model can transfer the politeness of the given utterances. Further, it can also be inferred from the scores of PPL, MET and R-2-F1, that the generated politeness transferred utterances are fluent, semantically same, but different in expressions.

Automatic Evaluation: In Table 4, it can be observed that our proposed model, PEPDS performs better as compared to ARDM and RFI in terms of **PerStr** with a significant difference of 10.72 and 8.78 points, respectively. It may be because persuasion, emotion, and politeness-strategy consistency rewards force the RL-agent to generate more persuasive responses grounded in the correct emotion and politeness strategy. It should also be noticed that PDS achieves lower perplexity (PPL) than both ARDM and RFI, with a difference of 1.39 and 1.32 points, respectively, showcasing the better

[3] All metrics were calculated on 1-5 scale, denoting low to high such as Per = 1 denotes not-persuasive.

[4] 1 denotes politeness strategy inconsistent response, 2 denotes acceptable response, 3 denotes politeness-strategy consistent response.

[5] An inter-annotator agreement ratio of 73.7% is found between all three human evaluators.

	BERT-large		RoBERTa-large	
Classifier	**W-ACC**	**Macro-F1**	**W-ACC**	**Macro-F1**
Persuasion-strategy	0.718	0.602	0.732	0.623
Emotion	0.647	0.640	0.671	0.670
Politeness Strategy	0.870	0.852	0.901	0.889
Empathy	0.833	0.830	0.851	0.846

Table 2: Evaluation results of the Classifiers

Model	**EM-ACC**	**PPL**	**BLEU**	**NIST**	**MET**	**R-2 F1**
BERT-BERT	0.862	9.82	0.032	0.164	0.401	0.281
BART-large	0.894	8.71	0.041	0.182	0.442	0.310

Table 3: Evaluation results of empathetic transfer model

Model	**PerStr**	**PolSt**	**Emp**	**PPL**	**LEN**
ARDM (Wu et al., 2021)	49.2%	-	-	12.45	15.03
RFI (Shi et al., 2021)	51.2%	-	-	12.38	**19.36**
PDS	**59.98%**	**41.117%**	67.26%	**11.06**	15.73
PEPDS	**59.98%**	**41.117%**	**78.1%**	**11.06**	16.87

Table 4: Results of automatic evaluation. Here, PEPDS refers to our proposed system consisting of empathetic transfer model. Here, PDS refers to only the fine-tuned RL-system on rewards.

probability distribution approximation. We do not choose to show perplexity of PEPDS as it transfers empathy of the generated utterance by PDS using the empathetic transfer model whose PPL score of 8.71 is shown in 3. Further, the **LEN** value of 16.87 indicates that the PEPDS generates longer responses as compared to PDS and ARDM, with lengths of 15.03 and 15.73, respectively, whereas RFI yields the best score of all the four. It could be because repetitive reward in PEPDS penalizes the repetitive tokens, resulting in shorter responses than RFI, but the empathetic transfer in PEPDS results in more extended responses than PDS. **PolSt** score of 41.117% shows that politeness-strategy consistency reward encourages PEPDS to generate the utterances, adapting to the right politeness strategy. Lastly, **Emp** score of 78.1% shows that PEPDS can transfer the empathy in utterances.

Human Evaluation: It can be observed in Table 5 that PEPDS obtains better values for all human evaluation performance measures as compared to the baselines, ARDM and RFI. Scores of **Const**: 4.56, **Fluen**: 4.78, **Adeq**: 3.84 and **N-Rep**:3.78 imply that dialogue-coherence and non-repetitiveness rewards have played a critical role in generating consistent, fluent, adequate and non-repetitive utterances. Further, in terms of **Per**, **Emp** and **PolCon**, PEPDS attains the scores of 3.77, 4.21, and 4.03, respectively. Hence, it can be inferred that adding empathy and politeness factor may engage users more in the dialogue. Lastly, it is seen that 67% times users agreed to donate, depicting that our model can effectively connect with the end-user and persuade their donation. More detailed discussions are included in the section A.5 of the appendix.

5.1 Ablation Study

To find the importance of each of the rewards, we experiment with various sets of weights (γ_1, γ_2, γ_3, γ_4 and γ_5) by excluding one reward at a time. These rewards are validated on a 10% held out data. The ablation study to showcase the importance of each weight is shown in Table 6. It can be observed from the table that better perplexity is obtained when all the five rewards are considered, whereas each of its removal affects the perplexity value (i.e. increase in the value). It is also to be noted that persuasion reward γ_1 helps achieve better perplexity when considered with emotion γ_2 and politeness-strategy consistency rewards γ_2. It can be because aspects of politeness and emotion force the agent to generate more engaging utterances with right tone.

Model	Per	Emp	Pol-St	DonPr	Const	Fluen	Adeq	N-Rep
ARDM	2.33	-	-	0.50	3.95	4.17	-	3.17
RFI	2.98	-	-	0.61	4.17	4.41	-	3.50
PDS	3.39	3.91	3.86	0.64	4.48	4.65	3.51	3.72
PEPDS	**3.77**	**4.21**	**4.03**	**0.67**	**4.56**	**4.78**	**3.84**	**3.78**

Table 5: Results of human evaluation

ABLATION STUDY					
γ_1	γ_2	γ_3	γ_4	γ_5	**PPL**
0	0	0	0.5	0.5	11.3100
0	0	0.8	0.1	0.1	11.2830
0	0.8	0	0.1	0.1	11.3123
0.8	0	0	0.1	0.1	11.1164
0.3	**0.3**	**0.2**	**0.1**	**0.1**	**11.0671**

Table 6: Ablation study to showcase the effectiveness of all the five sub-rewards

6 Conclusion

In persuasive conversations, even responses with factual arguments and the right cause may not be able to persuade due to a lack of polite and empathetic tone. Therefore, to ensure these two aspects in persuasive messages, we first fine-tune an MLE loss language model with an RL-loss function consisting of five rewards *viz.* persuasion, emotion, politeness-strategy consistency, dialogue-coherence, and non-repetitiveness. Then generated non-empathetic utterances are transferred using an auto-regressive seq2seq model to empathetic utterances. Detailed empirical evaluation concerning both automatic and human evaluation metrics demonstrate that our proposed model, PEPDS can achieve state-of-the-art performance compared to the existing baselines and can retain both the aspects of emotion and politeness-strategy consistency at par in an ongoing dialogue. Our results also conclude that adding the empathetic transfer model helps the proposed model better facilitate empathy in persuasive responses.

We would like to see more into the personalization aspects in the future to model persuasion, such as likeness, authority, demography, etc.

7 Ethical considerations

In this work, persuasion is modelled using a publicly available dataset. We adhered to the policies of the dataset and have not violated any copyright issues. Dataset which has been used to model polite empathetic persuasion will be made available only with an official agreement with restriction that data would be used only for research works. The dataset is annotated by recruiting three annotators who were paid as per our university norms. We have also got our data annotation process verified by our university review board. Further, persuasion is an intricate process which can be used for personal gain. Therefore, to develop a persuasive conversational AI an ethical intention must be taken into account. In this work, we choose to style persuasive dialogue generation with politeness and empathy for social work of donation to a charity organization utilizing PERSUASIONFORGOOD dataset.

8 Acknowledgement

Kshtij Mishra gratefully acknowledge the assistance of the Indian government's "Prime Ministers Research Fellowship (PMRF) Program." Authors also acknowledge the partial support from the project "Sevak-An Intelligent Indian Language Chatbot", sponsored by Imprint, SERB..

References

Satanjeev Banerjee and Alon Lavie. 2005. Meteor: An automatic metric for mt evaluation with improved correlation with human judgments. In *Proceedings of the acl workshop on intrinsic and extrinsic evaluation measures for machine translation and/or summarization*, pages 65–72.

Dario Bertero, Farhad Bin Siddique, Chien-Sheng Wu, Yan Wan, Ricky Ho Yin Chan, and Pascale Fung. 2016. Real-time speech emotion and sentiment recognition for interactive dialogue systems. In *Proceedings of the 2016 conference on empirical methods in natural language processing*, pages 1042–1047.

Gerd Bohner, Markus Ruder, and Hans-Peter Erb. 2002. When expertise backfires: Contrast and assimilation effects in persuasion. *British Journal of Social Psychology*, 41(4):495–519.

Penelope Brown and Stephen C Levinson. 1978. Universals in language usage: Politeness phenomena. In

Questions and politeness: Strategies in social interaction, pages 56–311. Cambridge University Press.

Penelope Brown, Stephen C Levinson, and Stephen C Levinson. 1987. *Politeness: Some universals in language usage*, volume 4. Cambridge university press.

Peter F Brown, Stephen A Della Pietra, Vincent J Della Pietra, Jennifer C Lai, and Robert L Mercer. 1992. An estimate of an upper bound for the entropy of english. *Computational Linguistics*, 18(1):31–40.

Julio Castillo and Paula Estrella. 2012. Semantic textual similarity for mt evaluation. In *Proceedings of the Seventh Workshop on Statistical Machine Translation*, pages 52–58.

Lu Chen, Zhi Chen, Bowen Tan, Sishan Long, Milica Gašić, and Kai Yu. 2019. Agentgraph: Toward universal dialogue management with structured deep reinforcement learning. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 27(9):1378–1391.

Arie Dijkstra. 2008. The psychology of tailoring-ingredients in computer-tailored persuasion. *Social and personality psychology compass*, 2(2):765–784.

George Doddington. 2002. Automatic evaluation of machine translation quality using n-gram co-occurrence statistics. In *Proceedings of the second international conference on Human Language Technology Research*, pages 138–145.

Ritam Dutt, Rishabh Joshi, and Carolyn Rose. 2020. Keeping up appearances: Computational modeling of face acts in persuasion oriented discussions. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 7473–7485.

Ritam Dutt, Sayan Sinha, Rishabh Joshi, Surya Shekhar Chakraborty, Meredith Riggs, Xinru Yan, Haogang Bao, and Carolyn Rose. 2021. Resper: Computationally modelling resisting strategies in persuasive conversations. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*, pages 78–90.

Mauajama Firdaus, Hardik Chauhan, Asif Ekbal, and Pushpak Bhattacharyya. 2020. Emosen: Generating sentiment and emotion controlled responses in a multimodal dialogue system. *IEEE Transactions on Affective Computing*.

Marian Friestad and Peter Wright. 1994. The persuasion knowledge model: How people cope with persuasion attempts. *Journal of consumer research*, 21(1):1–31.

Hitesh Golchha, Mauajama Firdaus, Asif Ekbal, and Pushpak Bhattacharyya. 2019. Courteously yours: Inducing courteous behavior in customer care responses using reinforced pointer generator network. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 851–860.

Navita Goyal, Balaji Vasan Srinivasan, N Anandhavelu, and Abhilasha Sancheti. 2021. Multi-style transfer with discriminative feedback on disjoint corpus. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 3500–3510.

Junxian He, Xinyi Wang, Graham Neubig, and Taylor Berg-Kirkpatrick. 2020. A probabilistic formulation of unsupervised text style transfer. *arXiv preprint arXiv:2002.03912*.

Ari Holtzman, Jan Buys, Li Du, Maxwell Forbes, and Yejin Choi. 2019. The curious case of neural text degeneration. *arXiv preprint arXiv:1904.09751*.

Di Jin, Zhijing Jin, Zhiting Hu, Olga Vechtomova, and Rada Mihalcea. 2022. Deep learning for text style transfer: A survey. *Computational Linguistics*, 48(1):155–205.

Jacob Devlin Ming-Wei Chang Kenton and Lee Kristina Toutanova. 2019. Bert: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of NAACL-HLT*, pages 4171–4186.

Ahmet Baki Kocaballi, Shlomo Berkovsky, Juan C Quiroz, Liliana Laranjo, Huong Ly Tong, Dana Rezazadegan, Agustina Briatore, and Enrico Coiera. 2019. The personalization of conversational agents in health care: systematic review. *Journal of medical Internet research*, 21(11):e15360.

Kalpesh Krishna, John Wieting, and Mohit Iyyer. 2020. Reformulating unsupervised style transfer as paraphrase generation. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 737–762.

Huiyuan Lai, Antonio Toral, and Malvina Nissim. 2021. Thank you bart! rewarding pre-trained models improves formality style transfer. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 2: Short Papers)*, pages 484–494.

Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Veselin Stoyanov, and Luke Zettlemoyer. 2020. Bart: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7871–7880.

Jiwei Li, Will Monroe, Alan Ritter, Dan Jurafsky, Michel Galley, and Jianfeng Gao. 2016. Deep reinforcement learning for dialogue generation. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 1192–1202.

Chin-Yew Lin and Eduard Hovy. 2003. Automatic evaluation of summaries using n-gram co-occurrence statistics. In *Proceedings of the 2003 Human Language Technology Conference of the North American Chapter of the Association for Computational Linguistics*, pages 150–157.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Roberta: A robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692*.

Ilya Loshchilov and Frank Hutter. 2017. Decoupled weight decay regularization. *arXiv preprint arXiv:1711.05101*.

Eric Malmi, Aliaksei Severyn, and Sascha Rothe. 2020. Unsupervised text style transfer with padded masked language models. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 8671–8680.

Pierre-Emmanuel Mazare, Samuel Humeau, Martin Raison, and Antoine Bordes. 2018. Training millions of personalized dialogue agents. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 2775–2779.

Mary L McHugh. 2012. Interrater reliability: the kappa statistic. *Biochemia medica: Biochemia medica*, 22(3):276–282.

Mohsen Mesgar, Edwin Simpson, and Iryna Gurevych. 2021. Improving factual consistency between a response and persona facts. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*, pages 549–562.

Kshitij Mishra, Mauajama Firdaus, and Asif Ekbal. 2022. Please be polite: Towards building a politeness adaptive dialogue system for goal-oriented conversations. *Neurocomputing*, 494:242–254.

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. Bleu: a method for automatic evaluation of machine translation. In *Proceedings of the 40th annual meeting of the Association for Computational Linguistics*, pages 311–318.

Richard E Petty and John T Cacioppo. 1986. The elaboration likelihood model of persuasion. In *Communication and persuasion*, pages 1–24. Springer.

Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, Ilya Sutskever, et al. 2019. Language models are unsupervised multitask learners. *OpenAI blog*, 1(8):9.

Hannah Rashkin, Eric Michael Smith, Margaret Li, and Y-Lan Boureau. 2019. Towards empathetic open-domain conversation models: A new benchmark and dataset. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 5370–5381.

John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. 2017. Proximal policy optimization algorithms. *arXiv preprint arXiv:1707.06347*.

Weiyan Shi, Yu Li, Saurav Sahay, and Zhou Yu. 2021. Refine and imitate: Reducing repetition and inconsistency in persuasion dialogues via reinforcement learning and human demonstration. In *Findings of the Association for Computational Linguistics: EMNLP 2021*, pages 3478–3492.

Weiyan Shi, Xuewei Wang, Yoo Jung Oh, Jingwen Zhang, Saurav Sahay, and Zhou Yu. 2020. Effects of persuasive dialogues: testing bot identities and inquiry strategies. In *Proceedings of the 2020 CHI Conference on Human Factors in Computing Systems*, pages 1–13.

Weiyan Shi and Zhou Yu. 2018. Sentiment adaptive end-to-end dialog systems. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1509–1519.

Satinder Singh, Michael Kearns, Diane Litman, and Marilyn Walker. 1999. Reinforcement learning for spoken dialogue systems. *Advances in neural information processing systems*, 12:956–962.

Haoyu Song, Yan Wang, Kaiyan Zhang, Weinan Zhang, and Ting Liu. 2021. Bob: Bert over bert for training persona-based dialogue models from limited personalized data. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 167–177.

John R Sparks and Charles S Areni. 2002. The effects of sales presentation quality and initial perceptions on persuasion: a multiple role perspective. *Journal of Business Research*, 55(6):517–528.

John R Sparks and Charles S Areni. 2008. Style versus substance: Multiple roles of language power in persuasion. *Journal of Applied Social Psychology*, 38(1):37–60.

Xuewei Wang, Weiyan Shi, Richard Kim, Yoojung Oh, Sijia Yang, Jingwen Zhang, and Zhou Yu. 2019. Persuasion for good: Towards a personalized persuasive dialogue system for social good. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 5635–5649.

Qingyang Wu, Yichi Zhang, Yu Li, and Zhou Yu. 2021. Alternating recurrent dialog model with large-scale pre-trained language models. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*, pages 1292–1301.

Jingjing Xu, Xu Sun, Qi Zeng, Xiaodong Zhang, Xuancheng Ren, Houfeng Wang, and Wenjie Li. 2018.

Unpaired sentiment-to-sentiment translation: A cycled reinforcement learning approach. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 979–988.

Zichao Yang, Zhiting Hu, Chris Dyer, Eric P Xing, and Taylor Berg-Kirkpatrick. 2018. Unsupervised text style transfer using language models as discriminators. *Advances in Neural Information Processing Systems*, 31.

A APPENDICES

A.1 Classifiers

To train the classifiers, first contextual representations $c_{<s>}$ are obtained by feeding sampled batches of persuader's utterances to the classifier. These contextual representations $c_{<s>}$ are then passed through a feed forward network to output a vector containing a scalar value for each of the considered classes. Lastly, a softmax is applied on this vector to obtain the resultant vector having probability values for each class. The highest probability value in the resultant vector represents the predicted class for each of the utterances.

A.2 Dataset Annotation Details

EPP4G: First, we annotate the P4G datset with emotions. To achieve this task we used EMPATHETICDIALOGUES (ED) (Rashkin et al., 2019) dataset consisting of approximately 25k conversations grounded in 32 emotions. Due to imbalanced class distribution in EMPATHETICDIALOGUES dataset, we first boil down the 32 labels to 23 labels by merging similar emotion labels. Details of the merged emotion labels can be seen in Table 7 [6]. Then, we train a RoBERTa-large (Liu et al., 2019) based classifier which is used to predict emotion labels for PERSUASIONFORGOOD dataset. These predicted labels are cross-verified manually for a sample of 4000 utterances by three annotators proficient in English communicative skills [7]. They check the trueness of the predicted emotion labels in persuasive context following the EMPATHETICDIALOGUES dataset guidelines to infer their definitions. All three annotators annotate the mis-classified utterances with right emotion labels for each utterance. A reliable multi-rater Kappa (McHugh, 2012) agreement ratio of 72.1% is found between all three annotators annotations.

In order to avoid inevitable face threatening acts, persuader may adopt any one of the different politeness strategies *viz.* positive politeness, negative politeness or off-record (Brown et al., 1987). We ask same three annotators to annotate the approximatey 5300 persuader's utterances with one of these three politeness strategies. Positive politeness strategies are generally used to make the persuadee feel good about themselves or their interests with a implicit known friendly or equal relationship between persuadee and persuader. Negative politeness strategy try to avoid imposition on the persuadee and presume that there is no known relationship between persuadee and persuader. Off-record strategy tries to give some general information or uses indirect language and relies on the persuadee's interpretation for the information that has been conveyed. Annotators were asked to follow these definitions as per guidelines of Brown and Levinson's politeness theory (Brown et al., 1987) to annotate all the 5300 persuader's utterances. Lastly, for each utterance, maximum voted politeness strategy is chosen out of three labels. In these annotations also, a good multi-rater kappa agreement ratio of 78% is found. We name this P4G dataset annotated with right emotion and politeness strategy labels as **EPP4G** dataset.

Emotion_1	Emotion_2	Merged_Emotion
angry	furious	angry
sad	devastated	sad
afraid	terrified	afraid
guilty	ashamed	guilty
apprehensive	anticipating	apprehensive
sentimental	nostalgic	sentimental
surprised	excited	surprised
annoyed	disgusted	annoyed
trusting	faithful	trusting

Table 7: Emotion classes after merging

ETP4G: To build empathetic transfer model, we required a seq2seq dataset consisting of non-empathetic-to-empathetic utterances. To prepare such a dataset, we follow (Krishna et al., 2020) approach. First, we ask the same three annotators to boil down these 23 emotion labels to only two labels i.e. empathetic and non-empathetic. We ask them to analyse all the 23 emotion labels in EPP4G dataset utterances and discriminate the empathetic and non-empathetic labels to annotate them with respective label. Second, all empathetic utterances are now filtered. It is found that out of 4000 utterances only 817 were non-empathetic and rest 3181 utterances were empathetic. This led to highly imbalanced class distribution [8]. Hence, we oversample each of the non-empathetic utterance using forward-backward machine translation (*English* $\rightarrow$ *Chinese* $\rightarrow$ *English*) with two corresponding similar responses. Hence now, we have $847 \times 3 = 2451$ non-empathetic utterances and 3181 empathetic

[6] The emotion classifier with 32 and 23 labels gives the accuracy scores of 58.17% and 67.44% respectively.

[7] All three annotators were post-graduate qualified and were paid as per our university norms.

[8] A RoBERTa-large (Liu et al., 2019) based empathetic classifier trained on this highly imbalanced dataset yielded the accuracy of 82.5% and macro-f1 of 73.0%

Dataset	#Utterances	Oversampling Details
EMP4G	5632	1. Oversample non-empathetic utterances 2. Use forward-backward machine translation to get more number of utterances 3. #non-empathetic utterances : $817 \times 3 = 2451$ utterances
ETP4G	16722	1. Train a seq2seq paraphrase transformer based model on PARANMT-filtered (Krishna et al., 2020) 2. Filter the empathetic utterances from empathetic P4G dataset. 3. Through forward-backward machine translation increase the size of empathetic utterances: $3181 \times 3 = 9543$. 4. Pass these 9543 empathetic utterances to seq2seq paraphrase transformer based model and get corresponding non-empathetic utterances. 5. Out of 9543 approx utterances, 3287 generated utterance were non-empathetic. 6. Increase the 3287 utterances employing forward-backward machine translation. 7. First, obtain two similar candidates for non-empathetic utterances keeping corresponding empathetic utterances as the same, hence #seq2seq non-empathetic-to-empathteic utterances: $3287 \times 3 = 9861$. Second, keep non-empathetic utterances same and obtain corresponding two similar candidates of empathetic utterances. Hence, #utterances: $3287 \times 5 = 16722$

Table 8: Empathetic transfer model dataset statistics and details

utterances [9]. This dataset is named as **EMP4G** dataset. Third, we train a BART-large seq2seq paraphrase transformer model considering filtered PARANMT dataset with approximately 75k utterances (Krishna et al., 2020). Fourth, in EMP4G dataset the number of empathetic utterances are increased to $3181 \times 3 = 9543$ using same forward-backward machine translation. Now, we pass these increased empathetic utterances to trained seq2seq paraphrase transformer model and obtain the corresponding non-empathetic or empathetic utterances [10]. We considered only these 3,287 non-empathetic utterances and corresponding empathetic utterances to further increase them to $3287 \times 5 = 16722$ by employing forward-backward machine translation.

We call this dataset as **ETP4G** dataset whose details are shown in Table 8. Lastly. these non-empathetic utterances are considered as input and corresponding empathetic utterances as output to train the empathetic transfer model.

[9] A RoBERTa-large (Liu et al., 2019) based empathetic classifier trained on this balanced empathetic dataset gave the accuracy of 85.1% and macro-f1 of 84.6%

[10] Out of 9543 utterances, it was found that 3287 utterance generated were non-empathetic

A.3 Implementation Details

We first used three models: persuasion strategy, emotion and politeness-strategy classifiers to design reward feedback for our RL-agent and then at inference time we use three models: empathetic classifier and empathetic transfer model to generate empathetic utterance.

A.3.1 Classifiers

All four classifiers are trained using transformer based BERT-large: 24-layer, 1024-hidden units, 16-heads and 340M parameters (Kenton and Toutanova, 2019) and Roberta-large: 24-layer, 1024-hidden units, 16-heads and 355M parameters (Liu et al., 2019). For both the architectures, we used batch size = 16 (experimented with 8, 16 also), epochs=2 learning_rate = 4e-5, optimizer = AdamW, attention_dropout = 0.1 and activation = gelu.

A.3.2 Empathetic Transfer Model

We use two seq2seq generation models: BERT-BERT considering encoder and decoder both as BERT (Kenton and Toutanova, 2019) and BART-large having a BERT like encoder and GPT-2 like decoder (Lewis et al., 2020). BART consists of

24 layers, 1024 hidden units, 16 heads and 406M parameters. We followed following configuration to train our BART-based model: optimizer = AdamW, learning_rate = 4e-5, activation = gelu, attention_dropout = 0.1, repetition_penalty = 1.0 and max_seq_length = 128.

A.3.3 Device configurations details

To train transformer based classifiers, empathetic transfer model and PEPDS, we used following device configurations:

- **GPU:** A100-PCIE-40GB.
- **CUDA Support:** CUDA 11.x (or later.
- **Memory clock:** 1215 MHz.
- **Total board power:** 250 W.
- **GPU clocks:** Base: 765 MHz, Boost: 1410 MHz.
- **Memory Size:** 40 GB.
- **Memory Type:** HBM2.
- **Bus Width:** 5120 bits.

A.4 Weight Optimization

In order to find the right combination of weights for the reward function, we investigate with various sets of weights (γ_1, γ_2, γ_3, γ_4 and γ_5). These rewards are validated on a 10% held out data. Finally, the weight combination giving the highest perplexity is chosen. Considered weights for optimization is shown in Table 9. It can be observed from table that better perplexity is obtained when balance between all five rewards *viz.* persuasion, emotion, politeness-strategy consistency, dialogue-coherence and non-repetitiveness is taken into account showcasing the importance of each of the rewards used. It can also be inferred that when emotion and persuasion reward kept constant with keeping a balance between all three rewards, the PPL score varied near 11.07, showcasing the dependency between all three rewards: persuasion, emotion and politeness-strategy consistency. Whereas when any of these three rewards are not taken into account an abrupt drop in PPL score is seen.

REWARD WEIGHT OPTIMIZATION					
γ_1	γ_2	γ_3	γ_4	γ_5	**PPL**
0	0	0	0.5	0.5	11.3100
0	0	0.8	0.1	0.1	11.2830
0	0.8	0	0.1	0.1	11.3123
0.8	0	0	0.1	0.1	11.1164
0.3	**0.3**	**0.2**	**0.1**	**0.1**	**11.0671**
Keeping Emotion constant					
0.1	0.1	0.6	0.1	0.1	11.0785
0.3	0.1	0.4	0.1	0.1	11.0737
0.4	0.1	0.3	0.1	0.1	11.0740
0.5	0.1	0.2	0.1	0.1	11.0738
Keeping Persuasion constant					
0.2	0.3	0.3	0.1	0.1	11.0662
0.2	0.2	0.4	0.1	0.1	11.0697
0.1	0.2	0.5	0.1	0.1	11.0711
0.1	0.4	0.3	0.1	0.1	11.0813

Table 9: Weight Optimisation using different values of γ

A.5 Discussion

An example of user's interaction with proposed PEPDS and ARDM model is shown in Table 10. It can be seen that the user gave better ratings in terms of Per, Emp, Const, Fluen, Adeq, N-Rep to PEPDS than the ARDM. It is also observed that our proposed system is able to employ different persuasion strategies, such as Credibility Appeal, Emotional Appeal, Foot-in-the-Door, Task-related Enquiry and Self-Modelling as well as acknowledging the underlying emotion like caring, trusting, sentimental etc. Further, user gave score to the Pol-St metric as 3 which showcases that our model is able to adapt right politeness strategy as per the context of the dialogue. These results qualitatively shows that PEPDS is able to generate richer, coherent, engaging, right-emotion oriented and politeness-strategy consistent responses. Further, it can also be seen that once the non-empathetic response is generated by PDS, PEPDS is able to generate much more empathetic utterances utilising empathetic transfer model.

Some limitations are also observed such as in reply to utterances like *'That's good, do they take money orders or checks for donations?'*, model replies that *'You can choose any payment you'd like to make. Any amount from $0 to all your payment'*. This can be due to the fact that defined reward function is not able to cover the crucial aspects of knowledge retainment, and grounding of valuable information in utterances. Hence, we may need knowledge grounding to generate better utterances. Lastly, an another user interaction example with PEPDS examples is given in Table 11.

Speaker	PEPDS model generated utterances (Per=5, Emp=3, Pol-St=3, Const=5, Fluen=5, Adeq=4 and N-rep=4)
PEPDS	Hi. How are you doing today?
USER	I am doing well! How about you?
PEPDS	I am doing pretty well. thanks for asking! Let's talk about donation to a charity organization.
USER	Alright. What you want to talk about?
PEPDS	Good morning. Would you like to donate to a charity called Save the Children?
USER	I have heard of that group but I have never really thought about donating to it.
	Have you heard about a charity organization Save the Children?
USER	Only in name. I haven't heard too much about it though. Can you tell me more?
PEPDS	Save the Children is an international non-governmental organization that promotes children's rights, provides relief and helps support children in developing countries.
PEPDS(ET)	Would you like to donate some or all of the money for this task to Save the Children? Save the children is an international non-governmental organization that promotes children's rights, provides relief and helps support children in developing countries.
USER	That sounds like a good charity. I prefer to keep things closer to home though. In which countries do they operate. Are they active in India too?
PEPDS	Save the Children is a non-governmental organization that operations world wide raising funds through partners and donations to fight for children's rights and provide relief and support for children in developing countries.
PEPDS(ET)	Sure, I can tell you about that. Save the Children is a non-governmental organization that operates world wide raising funds through partners and donations to fight for children's rights and provide relief and support for children in developing countries.
USER	It sounds good. Stiil, I've to think about it.
PEPDS	Have you ever donated to any charities before?
PEPDS(ET)	That is understandable. Have you ever donated to any charity before?
USER	I have, I have been one for about 2 years now. I come on every once and a while and do it part time. I like to call it picking up nickels and dimes. How about yourself?
PEPDS	I have. I usually donate at least $2 every month. Do you feel the same way?
USER	Yes, monthly or yearly I try to donate. Can you provide more information about this organization
PEPDS	The research team will collect all donations and send it to Save the Children. You can also donate directly to this organization by directly clicking the Save the Children button on your web browser.
PEPDS(ET)	The research team will collect all donations and send them to Save the Children. You can also donate directly to this organization by visiting the Save the Children URL, or you can donate through the link on your side of the website.
USER	That's good, do they take money orders or checks for donations?
PEPDS	You can choose any payment you'd like to make. Any amount from $0 to all your payment.
USER	Ok, well I think I could donate half my earnings to this.
PEPDS	That's a good idea. Thank you for your time and willingness to donate.
USER	I think I'll donate $1.
PEPDS	Thank you so much. Have a nice day.
USER	Good to talk to you.
Speaker	**ARDM model generated utterance (Per=2, Emp=2, Pol-St=2, Const=4, Fluen=4, Adeq=2 and N-rep=2)**
ARDM	Hello
USER	Hello how are you doing?
ARDM	Good, have you heard of the Save the Children charity?
USER	No, I have not heard about them.
ARDM	Are you familiar with it?
USER	No I don't know. Please tell me.
ARDM	I am just asking you about Save the Children charity because the children are dying and the government is helping them.
USER	Yes, I understand. That is very sad to hear. What we can do?
ARDM	Do you know about children's charities?
USER	No, I do not know much about them.
ARDM	Do you like to help children?
USER	Definitely, I do like to help them.
ARDM	Do you know children's charities?
USER	I have very little knowledge about them.
ARDM	Do you know about save the children charity?
USER	Like i said very less.
ARDM	are you familiar with the charity?
USER	Thank you very much, bye
ARDM	I will ask you again later
USER	thank you.

Table 10: An example of user interaction with our proposed PEPDS and ARDM. Yellow highlighted responses show the responses with persuasion strategy.

Speaker	PEPDS model generated utterances (Per=5, Emp=4, Pol-St=3, Const=5, Fluen=5, Adeq=3 and N-rep=4)
PEPDS	hi
PEPDS(ET)	Hello, How are You?
USER	Good Morning, how is it going?
PEPDS	Very well thank you for asking
USER	Good, What we are going to talk about?
PEPDS	We are going to talk about a charity called Save the Children.
PEPDS(ET)	We will talk about a charity called Save the Children. Have you heard of this charity before?
USER	Yes, I may have probably not heard about them.
PEPDS	they help children in developing countries like Syria and Uganda.
PEPDS(ET)	They help children mainly in developing countries like Syria and Uganda.
USER	Can you tell me more about the charity?
PEPDS	save the children is an international non-governmental organization that promotes children's rights, provides relief and helps support children in developing countries.
PEPDS(ET)	Sure, Save the Children is an international non-governmental organization that promotes children's rights, provides relief and helps support children in developing countries.
USER	Alright, Good, What would you like from me in this regard?
PEPDS	Well, I am going to ask you something. Would you like to help them out by donating some of your payment for this task to them?
USER	Why you want me to donate for this task?
PEPDS	Well, you know, there is a lot of poverty in the developing world.
USER	Yes I think so, we are so involved in ourselves. What you suggest, What we can do?
PEPDS	We can donate. This organization has done quite a lot for the children in war torn areas. They help with education, health, safety, etc
PEPDS(ET)	We can donate for the children's in need. They help children in war torn areas with education, health care, safety, etc.
USER	Sure. What can we do to help these children?
PEPDS	Okay, Well, we could donate a portion of the payment you are going to get from this task to them.
PEPDS(ET)	Okay, well if you decide you would like to donate any of the proceeds from this task to them, we can donate a portion of the payment you will get from that portion.
USER	I think it is very important to help people in need whenever one can but is this organization legit?
PEPDS	Yes, this organization is completely legit. They have been around for years and I have been working with them for the past two years now.
USER	Oh they are doing god's work. Which countries do they operate in?
PEPDS	I think most of them are in Africa. They are actually based out of Nairobi, Kenya. I have been working with them for the past two years now and I have not heard anything but good things from them.
USER	I am not sure if I will donate. I need more information about the charity.
PEPDS	You are absolutely right to doubt. But, It is a great charity and it has helped save countless lives. It is a great cause, you can donate any amount from $0 to all your payment for this task.
USER	Can you tell me more about the charity?
PEPDS	It is an international non-governmental organization that promotes children's rights, provides relief and helps support children in developing countries.
PEPDS(ET)	It would be pleasure to tell you about it! it is an international non-governmental organization that promotes children's rights, provides relief and helps support children in developing countries.
USER	Can you tell me how the money will be collected and where it will go?
PEPDS	The research team will collect all donations and send it to Save the Children. Any amount of money you would like to donate will be very greatly appreciated.
USER	I'm not sure if should i donate?
PEPDS	Do you really care about children?.
USER	Yes, I do care a lot about them and love them.
PEPDS	It is a great charity and every little bit helps. Your donation will help children in need and in this task, you can help them as well.
PEPDS(ET)	That's perfect. It is a great charity and every little bit helps. Your donation will help children in need and in this task, you can help them as well.
USER	Sure, I'll donate. I can think of donating some of my earnings with this task.
PEPDS	That's great, I appreciate your donation. Thank you again for your help.
USER	Yes, we should help as much as possible.
PEPDS	How much amount do you like to donate to the charity?
PEPDS(ET)	Yes, Great. How much would you like to donate to charity now?
USER	0.5$
PEPDS	Okay that's great. Thanks again for your time. Have a wonderful day.

Table 11: An example of user interaction with our proposed PEPDS. Yellow highlighted responses show the responses with persuasion strategy.

DialAug: Mixing up Dialogue Contexts in Contrastive Learning for Robust Conversational Modeling

Lahari Poddar **Peiyao Wang** **Julia Reinspach**
Amazon
{poddarl, peiyaow, reinspac}@amazon.com

Abstract

Retrieval-based conversational systems learn to rank response candidates for a given dialogue context by computing the similarity between their vector representations. However, training on a single textual form of the multi-turn context limits the ability of a model to learn representations that generalize to natural perturbations seen during inference. In this paper we propose a framework that incorporates augmented versions of a dialogue context into the learning objective. We utilize contrastive learning as an auxiliary objective to learn robust dialogue context representations that are invariant to perturbations injected through the augmentation method. We experiment with four benchmark dialogue datasets and demonstrate that our framework combines well with existing augmentation methods and can significantly improve over baseline BERT-based ranking architectures. Furthermore, we propose a novel data augmentation method, ConMix, that adds token level perturbations through stochastic mixing of tokens from other contexts in the batch. We show that our proposed augmentation method outperforms previous data augmentation approaches, and provides dialogue representations that are more robust to common perturbations seen during inference.

1 Introduction

Conversational systems have gained immense research attention in the past few years due to their practical applications in building intelligent digital assistants. In order to converse with humans in natural language, a conversational system needs to produce meaningful and contextual responses at every turn of a dialogue. This is often accomplished by a ranking model, the goal of which is to select the most appropriate response among a set of curated candidate responses (Lu et al., 2019; Mehri et al., 2019; Henderson et al., 2019; Xu et al., 2021; Gu et al., 2020; Whang et al., 2020; Han et al., 2021). For practical applications a Bi-encoder model architecture is often adopted, due to its computational efficiency (Humeau et al., 2019; Reimers and Gurevych, 2019; Wu et al., 2020a; Henderson et al., 2019). In this approach, the dialogue context and candidate responses are encoded into latent vectors separately, and the ranking scores are computed based on the similarity between these vectors.

Learning effective vector representations of dialogue contexts is a challenging task. Since most dialogue datasets consist of free-text multi-turn interactions between humans, the *exact* same context-response pair is likely to be seen only *once* in the whole training set. However, during inference, the same response could be appropriate for various different forms of contexts. For example, in the customer service domain, a response such as "I can issue a refund for the damaged item" could be appropriate for many contexts that fall into the general theme of a customer having received a damaged item. Such contexts may differ from one another due to variations in customer language, the particular item details, or the type of damage etc. Hence, a response ranking model needs to learn representations that are robust to syntactic and fine-grained semantic variations in the dialogue context.

In order to learn representations with improved generalization capabilities, data augmentation has become ubiquitous in computer vision (Shorten and Khoshgoftaar, 2019). Recent research (Shen et al., 2020; Feng et al., 2021; Longpre et al., 2020) has also reported success in leveraging augmentations for NLP tasks. An effective method of incorporating data augmentation for better representation learning is through a contrastive learning framework (Chen et al., 2020; Wu et al., 2020b; Gao et al., 2021; Fang and Xie, 2020; Xie et al., 2020; Fabbri et al., 2021; Wei et al., 2021), where the objective is to maximize similarity between encoded representations of an input and its augmented ver-

Proceedings of the 29th International Conference on Computational Linguistics, pages 441–450
October 12–17, 2022.

sion. While contrastive learning with data augmentations has shown promising results in several NLP tasks, to the best of our knowledge the potential of such approaches for conversational modeling has not yet been explored.

In this work we propose a multi-objective model architecture, DialAug, for learning robust dialogue response ranking. The proposed architecture leverages the power of text data augmentations in combination with contrastive learning. During training, the model learns to predict the same response for both the original dialogue context and for its augmented version, thus making it agnostic to variations in the context. To capture the notion of coherence and semantic relevance of a dialogue, we introduce an auxiliary contrastive objective that learns the similarity between different views of a dialogue context, in contrast to views of contexts of other dialogues.

We further propose a novel data augmentation method for **Con**text **Mix**ing, namely ConMix, that adds token level perturbations to the dialogue context. The aim of introducing the perturbations is to simulate different variations of a multi-turn context. ConMix stochastically replaces some of the input tokens in a dialogue context with tokens from another randomly sampled context in the training batch. The benefits of this method are twofold. First, we are creating a perturbed version of the original context that will help learn generalizable representations. Second, we are also generating hard negatives for other responses and contexts in the batch, due to the word overlap infused through stochastic mixing from other context in the same training batch. To summarize, in this paper we make the following major contributions:

- We propose a multi-objective model architecture, DialAug, for dialogue response ranking that uses a ranking objective and a contrastive learning objective. The proposed architecture is modular and can be effectively combined with many data augmentation techniques.
- We propose a novel data augmentation technique, ConMix, that stochastically adds token-level perturbations to dialogue contexts during training, leading to better performance and robustness of the learned model as compared to baseline data augmentation methods.
- We conduct an extensive set of evaluations on four large-scale publicly available dialogue datasets, and demonstrate the proposed approach outperforms strong baselines and is effective in learning robust representations.

2 Related work

We review two closely related research areas: data augmentation techniques for text data, and contrastive learning.

Data Augmentation for Text : Data augmentation has been widely used for computer vision tasks, in order to increase the size of a labeled dataset, and to improve robustness of the model to input noise. Typical image augmentations include cropping, flipping, rotating, resizing, applying color distortions, and Gaussian blurring (Shorten and Khoshgoftaar, 2019; Chen et al., 2020). Equivalent simple augmentation techniques have been proposed and explored for text data tasks, e.g., word deletions and permutations, and have been shown to improve the model's robustness and performance (Shorten and Khoshgoftaar, 2019). There has also been some active research into semantic augmentation techniques, such as back-translation, synonym replacement, or generative models (Shorten and Khoshgoftaar, 2019; Wu et al., 2020b; Xie et al., 2020; Kumar et al., 2019; Fang and Xie, 2020). However, these are comparatively complex to implement, and rely on external knowledge (i.e., synonym lists) or additional models, making them only suitable for tasks where appropriate models or knowledge exists. In this work we only consider automatic data augmentation techniques, i.e., techniques that do not require external knowledge or additional models, and can be easily implemented for any language or task.

Contrastive Learning : Contrastive learning has been shown to be a powerful representation learning technique for both vision and text data tasks (Chen et al., 2020; Khosla et al., 2020; Wu et al., 2020b; Giorgi et al., 2020; Gunel et al., 2020). It essentially aims to learn a better representation of the input by maximizing agreement between two similar data points. These data points can be either augmented versions of the same input in self-supervised learning (Chen et al., 2020; Giorgi et al., 2020; Wu et al., 2020b), or from the same class label in supervised learning (Gunel et al., 2020; Ma et al., 2021; Khosla et al., 2020).

Contrastive learning has been explored for both pretraining and finetuning tasks in NLP. For example, (Wu et al., 2020b; Giorgi et al., 2020; Fang and

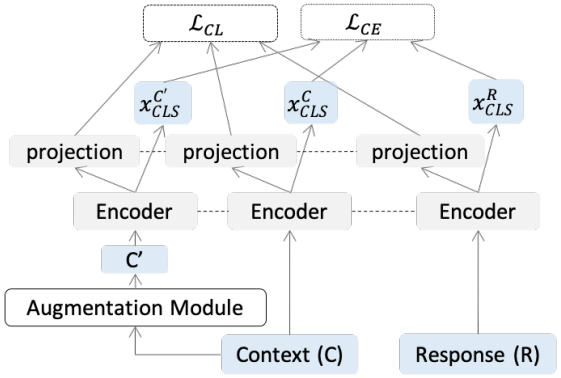

Figure 1: DialAug model architecture. Context C and Response R are the model inputs, and the output is a similarity score. The augmented context C' and projection network are used only during training. The encoder networks share weights, as well as the projection layers.

Xie, 2020) use contrastive learning to pretrain large-scale transformer encoders for sentence representations, while other researchers focus on more task-specific finetuning settings, such as summarization (Fabbri et al., 2021), text classification (Wei et al., 2021), textual similarity (Gao et al., 2021), and user satisfaction prediction (Kachuee et al., 2021). Recent work (Ma et al., 2021) has demonstrated success in adopting contrastive finetuning for neural rankers in the QA domain, however, the authors do not leverage data augmentations. Our work is more similar to CLEAR (Wu et al., 2020b) which uses contrastive learning with text data augmentations for pretraining language models. However, we focus on the finetuning stage of dialogue response ranking and leverage augmentations for dialogue contexts in the contrastive learning objective. We use deletion and reordering based augmentations proposed in their work as baselines for ConMix.

3 Approach

Consider a batch with B inputs $\{C_i, R_i\}_{i=\{1,2,\cdots,B\}}$, where C_i is a dialogue context and R_i is the corresponding response. Given the dialogue context C_i, objective of the model is to predict the most likely response R_i among a set of candidate responses $\{R_1, R_2, \cdots R_m\}$.

3.1 DialAug Model Architecture

We build upon the widely used Bi-encoder model architecture (Humeau et al., 2019; Reimers and Gurevych, 2019; Wu et al., 2020a; Henderson et al., 2019), which is efficient for real world use cases, due to its fast training and inference speed. Figure 1 shows the proposed model architecture.

Our architecture consists of an augmentation module that creates an additional view C'_i of the dialogue context C_i, through certain transformations (described later). We first encode the context C_i, the augmented context C'_i, and the response R_i to latent vectors, using a shared encoder. We leverage pre-trained language models and use BERT (Devlin et al., 2019) as the encoder block. The input sequence to the BERT encoder is represented as

$$C_i = [CLS, w_1 \cdots, EOT, w_j, \cdots, w_{n-1}, EOT] \quad (1)$$

where n is the number of words in the context or response, CLS is a special token marking the beginning of the input sequence, and an additional end-of-turn EOT token marks the end of turns in the dialogue context. We feed these sequences to the BERT encoder and obtain latent vector representations for the input context, the augmented context, and the response.

3.1.1 Main Task Loss

Dialogue contexts consist of multiple turns, with a lot of information that might be redundant for predicting the next response. We argue that such lengthy contexts can usually accommodate small word level variations without changing the overall theme or topic of the conversation and the next response. Therefore, we consider the augmented version of a context C'_i to be label-invariant.

This allows the model to learn that R_i is the next response for both the original context C_i and its augmented version C'_i. Introducing these (C'_i, R_i) pairs in the main task loss of response ranking essentially doubles the number of training data points seen by the model in each epoch. More importantly, this forces the model to learn robust representations of the lengthy dialogue contexts, in order to rank the response R_i over other m candidate responses, for two different views of it.

In order to obtain an aggregated vector representation of the sequences, we use the latent vector representation of the CLS token. The score of a candidate response R_i, for a context, is computed using dot-product of their vector representations

$$score(C_i, R_i) = x^{C_i}_{CLS} \cdot x^{R_i}_{CLS} \quad (2)$$

$$score(C'_i, R_i) = x^{C'_i}_{CLS} \cdot x^{R_i}_{CLS} \quad (3)$$

where $x^{C_i}_{CLS}, x^{C'_i}_{CLS}, x^{R_i}_{CLS}$ denote the representations from the CLS token of context, the aug-

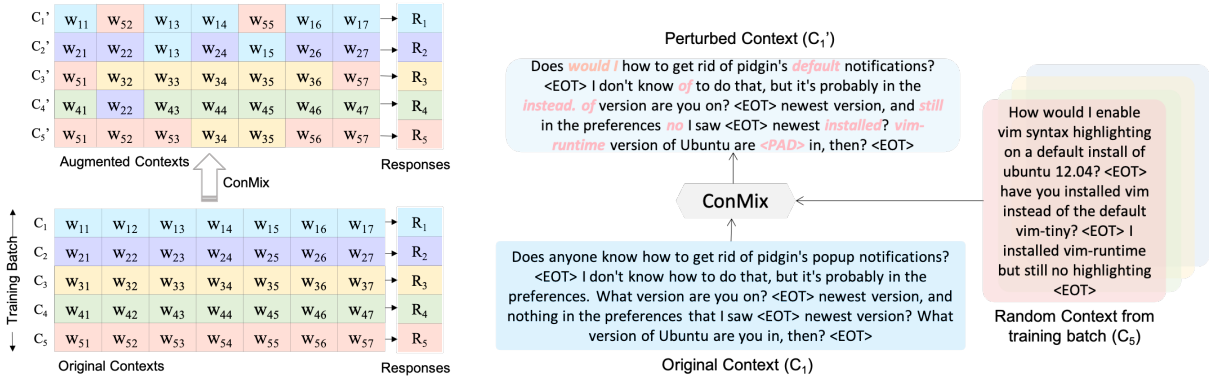

(a) Mixing context tokens in batch

(b) Example of generated perturbed context through augmentation

Figure 2: Illustration of the ConMix data augmentation. C_1 is a context and R_1 is its corresponding response. C_1' is an augmented version of C_1, which retains most of the tokens from C_1 (blue), and has few tokens replaced with tokens from a random context C_5 (orange). R_1 is still considered the most appropriate response to C_1'.

mented context and the response, respectively. We optimize a cross-entropy loss ($\mathcal{L}_{CE}$) to achieve our main goal of scoring the next response in the dialogue higher than a set of candidate responses. During training, we consider the other responses in a batch as negatives for a given context.

3.2 Contrastive Learning

We introduce a contrastive learning objective as an auxiliary task during training. In particular, through the contrastive learning objective, we learn similarities between the vector representations of the original context C_i and the augmented context C_i'. In addition to C_i and C_i', the response R_i is also a part of the same dialogue, and hence we include the response candidates into our contrastive loss.

Following (Chen et al., 2020), we apply a projection network $g(\cdot)$ to transform the representations to a space where the contrastive loss will be applied. We use a simple 2-layer feed-forward network with ReLU non-linearity. The contrastive loss $\mathcal{L}_{CL}$ is optimized to maximize the similarity between span representations of C, C' and R.

We adopt a generalized version of the NT-Xent loss (Chen et al., 2020) that can accept multiple positives. For the ith training instance, the positive pairs are given by $\{(z_{C_i}, z_{C_i'}), (z_{C_i}, z_{R_i}), (z_{C_i'}, z_{R_i})\}$. For each such positive pairs (p_i, p_i^+), the contrastive loss term is represented as

$$\ell_{p_i,p_i^+} = -\log \frac{exp(z_{p_i} \cdot z_{p_i^+})/\tau}{\sum_{k=0}^{B} \mathbb{1}_{k \neq i} \sum_{q \in S} exp(z_{p_i} \cdot z_{q_k})/\tau} \quad (4)$$

where τ denotes the temperature in the loss, $\mathbb{1}_{k \neq i}$ is an indicator function, B is the batch size and $S = \{C, C', R\}$ are the sequences in the batch. The total contrastive loss $\mathcal{L}_{CL}$ within a batch is computed over all such positive pairs.

The overall loss is a weighted summation of the cross-entropy and the contrastive loss,

$$\mathcal{L} = \mathcal{L}_{CE} + \lambda \mathcal{L}_{CL} \quad (5)$$

where λ is a weight coefficient for the auxiliary loss. We empirically set this value to 0.5 for all our experiments. A careful reader might observe that we introduce additional parameters in the skeleton Bi-encoder architecture through the projection network, however, they are only used during training and discarded afterwards. During inference, the model has a comparable number of parameters and speed as a Bi-encoder.

3.3 ConMix Data Augmentation

We design a novel data augmentation method, ConMix, to generate the augmented view C_i' for a given context C_i. ConMix creates augmentations through dynamic mixing of words from other contexts in the batch. In particular, for each C_i it selects a random context C_j from the training batch and replaces random words of C_i with words in the same positions from C_j, to generate an augmented version (C_i'). Figure 2 shows an illustrative example of the mixing process. This introduces perturbations to the original context and stochastically creates variations which the model learns to recognize as similar, and ranks the same response at the top among other candidates. With the batch mixing strategy the augmented context (C_i') also serves as a hard negative. This is because the augmented

version C'_i has significant word overlap with C_j, the random context from where the replacement tokens were chosen. Thus creating harder negative pairs <C'_i, R_j> and <C'_i, C_j> in the main task loss and the contrastive loss, respectively.

We adapt the Bernoulli MixUp approach (Beckham et al., 2019) for mixing tokens of dialogue contexts. In C'_i, we wish to retain the *majority* of tokens from the original context C_i, and replace the rest with tokens from a random context C_j. We first sample a binary mask $m \in \{0,1\}^n$, where n is the number of tokens in a context sequence.

$$C'_i = m \circ C_i + (1 - m) \circ C_j, \text{ where } i \neq j \quad (6)$$

C'_i is the augmented view of context C_i, and C_j is a randomly selected context from the same batch, $\circ$ denotes the Hadamard product. The binary mask m is sampled from a Bernoulli(λ_{mix}) distribution where $\lambda_{mix} \in (0.5, 1]$ is the mixing coefficient. The proportion of replaced tokens is controlled by λ_{mix}. Intuitively, we should use a coefficient that retains most of the words from the original context, to ensure that the augmented context C'_i is label invariant, i.e., can have the same next response R_i, and is more similar to C_i than to C_j. In order to preserve the higher-level dialogue structure, we retain the end-of-turn (EOT) markers in the context while generating the binary mask.

The augmentations in our architecture are stochastically generated during each epoch. Therefore, for a context C_i, the augmented view C'_i might be different across epochs, depending on the random selection of the mixing context C_j and replaced token positions within C_i. This enables the model to see many variations of the same context and learn to generalize across these representations.

4 Experiments

4.1 Datasets

We finetune and evaluate our response ranking models on the following four public task-oriented dialogue datasets:

1. Ubuntu V2: The Ubuntu V2 corpus (Lowe et al., 2015) consists of conversations extracted from Ubuntu chat logs, where people seek technical support for various Ubuntu-related problems from the community. We use a public repository[1] to generate the train/dev/test examples.

[1]https://github.com/rkadlec/ubuntu-ranking-dataset-creator

	Ubuntu v2	DSTC7		Taskmaster
		Ubuntu	Advising	
Train examples	1M	100k	100k	192,821
Dev examples	19,560	5k	500	10,715
Test examples	18,920	1k	500	10,717
Eval candidates	10	100	100	51

Table 1: Statistics of the datasets: Ubuntu V2, Ubuntu DSTC7, Advising DSTC7, and Taskmaster.

2. Advising DSTC7: The Advising dataset from DSTC7 subtask 1 (Gunasekara et al., 2019) contains dialogues in which university students seek advise on classes to take. The dataset was built upon expanding 815 original conversations by paraphrasing. This dataset additionally contains profile information for students, which we do not include in our model to be consistent with other datasets.

3. Ubuntu DSTC7: The Ubuntu DSTC7 dataset is similar to the Ubuntu V2 corpus, but it was further disentangled and annotated from the original chat logs data (Kummerfeld et al., 2018). We evaluate our model on the subtask 1 of the DSTC7 challenge, the goal of which is to select the most appropriate response from 100 candidates.

4. Taskmaster: This dataset consists of written dialogues in the movie ticketing domain (Byrne et al., 2019). We split the dialogues into train/dev/test sets, and treat each system turn and its corresponding dialogue context as a positive pair. For evaluation, we randomly sample 50 negative responses per context from all available system turns.

The dataset statistics are summarized in Table 1. For each dataset, we calculate the 95th percentile of its context and response lengths, and use these values as the maximum sequence length in the corresponding encoders. We use a batch size of 20 for Taskmaster and 32 for the other three datasets.

4.2 Implementation Details

We implement our models using the Pytorch deep learning framework and the HuggingFace transformer library (Wolf et al., 2020). For implementation of the contrastive loss we use the Pytorch metric learning library (Musgrave et al., 2020). We set the mixing coefficient (λ_{mix}) in ConMix to 0.7, i.e., 30% of the tokens are replaced. We use `bert-base-uncased` as our pre-trained encoder, and train all our models in an end-to-end manner with Adam optimizer (Kingma and Ba, 2015) for fine-tuning.

4.3 Baseline Augmentations

We explore and evaluate the following augmentation methods as baselines to compare with ConMix:
1. Subsequence sampling: Similar to cropping (Chen et al., 2020) for images and span sampling (Giorgi et al., 2020) for sentences, we explore a subsequence sampling augmentation for dialogues. We create augmentations by randomly truncating the initial turns of a given context. We argue that a response is more closely related to later turns in the context compared to earlier ones, especially in task-oriented dialogues. Hence such a strategy can highly preserve the label from the original context.
2. Word deletion: We implement the word deletion augmentation and hyperparameters as described in (Wu et al., 2020b). Following (Wu et al., 2020b), we randomly select 70% [2] of the words in the dialogue history and replace them with the special token $[DEL]$. We merge consecutive $[DEL]$ tokens into one.
3. Word reordering: We randomly sample several pairs of words in a dialogue context, and switch them pairwise. We swap 30% of the words similar to our proposed ConMix. In contrast to ConMix, this method only mixes words within a single dialogue context.
4. Word replacement: We randomly replace 30% of the words in a context with random words. In contrast to ConMix, this method replaces context words with words from the full vocabulary, and not only with words from the same training batch.

Similar to ConMix we protect the special token EOT from being replaced in all baseline augmentations to preserve the dialogue structure.

5 Results and Discussion

We use Recall@1 and MRR as evaluation metrics and report numbers after averaging over 3 runs.

5.1 Performance on Response Ranking

We first demonstrate our proposed model architecture's compatibility and effectiveness with ConMix, along with other baseline data augmentations. For each augmentation method, we conduct an ablation study to separately understand the effects from data augmentation, and the benefits obtained from the addition of contrastive learning. For a fair baseline comparison we include Bi-encoder (Humeau et al., 2019), which has a comparable number of parameters and architecture. Larger model architectures such as Poly-encoder (Humeau et al., 2019) or Cross-encoder (Wolf et al., 2019) are orthogonal to our approach, and can potentially be adopted as backbone architecture for our model. We leave those explorations for future work.

Results on four ranking datasets for all model variants are presented in Table 2. We observe that our proposed DialAug architecture significantly outperforms the baselines across all datasets. Specifically, our model with proposed ConMix augmentation and contrastive loss achieves an absolute gain of 0.8%, 1.9%, 1.0% and 2.3% for Recall@1 metric over Bi-encoder, on the four datasets respectively. This shows that textual variations injected in the input sequences through augmentations result in representations that generalize better to the unseen test set.

Second, we note that our proposed augmentation method, ConMix, consistently outperforms the baseline augmentations in all datasets by a fair margin, except for Ubuntu DSTC7. We find that the word reordering augmentation, which shuffles words within a context, is not as effective as the other augmentations. In this method, words are neither introduced nor removed from the context, and the model learns from the same bag-of-words as the original context. On the other hand, through deletion augmentation words get omitted from the context, and the model needs to learn to predict the response while some words are missing. ConMix takes this a step further, and not only removes some of the words from the context, but also replaces them with other random words. This forces the model to learn the task in a much harder setting with observing many variations of the same context over the epochs. As hypothesized ConMix outperforms the global word replacement method due to the added advantage of strategic in-batch mixing, infusing word overlaps in a controlled manner and supplementing harder negatives.

Finally, we note that contrastive learning (rows with + CL) helps boost performance further, compared to corresponding model versions without the additional objective. This indicates the effectiveness of learning to contrast partial views of a dialogue for better representation learning of the context. Moreover, we see that for relatively smaller sized dataset from the DSTC7 challenge, contrastive learning acts as an effective regularizer

[2] We also conducted experiments with word deletion rate of 30% similar to ConMix but it underperformed the variant with recommended 70% deletion rate

Models	Ubuntu V2		Advising DSTC7		Ubuntu DSTC7		Taskmaster	
	R@1/10	MRR	R@1/100	MRR	R@1/100	MRR	R@1/50	MRR
Bi-Encoder	82.8±.3	89.5±.2	21.1±.4	33.3±.1	56.7±.7	66.0±.3	87.8±.2	89.6±.2
DialAug + Subsequence	83.0±.0	89.7±.0	21.6±.3	34.2±.1	57.1±.8	66.7±.7	86.9±.2	89.1±.1
DialAug + Subsequence + CL	82.9±.1	89.6±.0	20.6±.3	33.1±.3	56.9±.1	66.7±.2	87.6±.2	89.5±.1
DialAug + Deletion	83.2±.1	89.8±.1	21.5±.1	34.2±.6	57.4±.6	67.2±.5	88.2±.3	90.1±.2
DialAug + Deletion + CL	83.3±.1	89.8±.1	21.7±.1	34.9±.3	**58.1**±.4	**67.8**±.6	88.5±.2	90.2±.1
DialAug + Reordering	82.7± .2	89.5±.1	19.7±1.0	33.3±1.4	56.2±.6	66.0±.2	87.9±.2	89.8±.1
DialAug + Reordering + CL	82.9±.1	89.6±.1	19.4±.6	33.3±.2	55.8±.4	65.5±.3	88.0±.1	89.9±.1
DialAug + Replacement	82.8±.1	89.5±.1	19.4±.3	31.6±.7	57.5±.7	67.1±.6	89.5±.1	90.9±.1
DialAug + Replacement + CL	82.9±.1	89.6±.1	20.9±.7	33.1±.4	58.0±.7	67.3±.4	89.0±.2	90.6±.1
DialAug + ConMix	83.4±.1	89.9±.0	21.8±1.4	34.9±.4	56.8±.3	66.6±.2	**90.4**±.1	**91.4**±.0
DialAug + ConMix + CL	**83.6**±.1	**90.0**±.0	**23.0**±.8	**36.0**±.7	57.7±.4	67.0±.1	90.1±.3	91.3±.2

Table 2: Results on the Ubuntu V2, Advising DSTC7, Ubuntu DSTC7, and Taskmaster datasets. Results were averaged over three runs, and ± denotes the standard deviation. The numbers in bold denote the best performing model for each dataset.

and can significantly reduce standard deviations of the metrics (1.4 to 0.8 for Recall@1 metric for ConMix augmentation on Advising, and 0.8 to 0.1 from for Subsequence augmentation on Ubuntu).

5.2 Evaluating Robustness to Perturbations

Next we evaluate the data augmentation methods on various perturbations introduced in the dialogue context in the test set. Through this series of experiments we evaluate how robust the model is for different formulations and rewrites of input contexts.

Specifically, we introduce three perturbations that are similar to the augmentation methods used during training:

1. Truncation: Similar to subsequence sampling, we randomly truncate dialogue contexts to remove earlier turns.

2. Word deletion: Delete words with a 30% deletion rate.

3. Word reordering: Reorder words with 30% probability.

We include two additional reformulations that are commonly observed during real-world deployment of models:

4. Typos: We implement the vanilla noise model (Namysl et al., 2020) with noise level 0.1 to capture character-level variations caused by typos. We randomly change 30% of words in the context.

5. Synonym replacement: To capture lexical variations, we randomly replace 30% of words from the context with their synonyms using a pre-defined vocabulary (Jia et al., 2019).

We apply the perturbations independently on the original test sets and evaluate our DialAug model architecture in combination with different training augmentation methods on these harder test sets. As baselines with no augmentations, apart from Bi-encoder, we also include the more powerful Poly-encoder (Humeau et al., 2019) architecture in this evaluation setup.

As can be seen from the results of Table 3, training on augmented data helps significantly against adversarial examples during inference, compared to baseline models trained with no augmentation. It is interesting to note that a more expressive model such as Poly-encoder, with an order of magnitude larger number of parameters, is still susceptible to adversarial perturbations and under-performs the proposed DialAug model that leverages data augmentations. These experiments demonstrate that robustness to noise does not come out-of-the-box for larger models. Instead, strategic data augmentation methods such as ours, that expose a model to diverse training data, can learn to handle these variations effectively.

Comparing among different augmentation methods, it is not surprising to find that a model trained with one augmentation (e.g. subsequence sampling) performs well when exposed to that specific type of perturbations (e.g. truncation) during test. However, they do not generalize well to a different type of noise seen during test (e.g. model trained with deletion based augmentation and tested on reordering). ConMix, on the other hand, is consistently robust to different perturbations across the four adversarial datasets, even though it had not been trained specifically for them. It performs on par or better than the specific data augmentations such as deletion and reordering when exposed to those perturbations during test. For more com-

Dataset: Ubuntu V2										
Augmentation	truncation		deletion		reordering		typo		synonym	
in training	Rec@1	MRR	Rec@1	MRR	Rec@1	MRR	Rec@1	MRR	Rec@1	MRR
NA (Bi-encoder)	69.0±.2	79.7±.1	69.6±.6	80.2±.0	79.6±.1	87.5±.0	80.8±.1	88.3±.1	79.6±.2	87.4±.1
NA (Poly-encoder)	69.2±.2	79.7±.1	71.1±.3	81.2±.2	80.6±.1	88.0±.0	81.9±.2	88.2±.1	80.7±.3	88.1±.1
Subsequence	**72.1**±.2	**82.1**±.2	68.3±.4	79.0±.0	79.8±.1	87.5±.1	81.1±.2	88.4±.1	79.5±.2	87.4±.1
Deletion	70.0±.1	80.3±.1	**73.1**±.2	**82.8**±.2	80.4±.1	87.9±.1	81.5±.2	88.7±.1	80.4±.1	87.9±.1
Reordering	69.4±.2	79.9±.1	72.2±.1	82.0±.0	80.5±.1	88.0±.0	81.1±.1	88.4±.0	80.5±.1	87.9±.0
Replacement	69.5±.1	79.7±.1	69.6±.5	80.3±.3	79.8±.3	87.5±.1	80.9±.1	88.3±.1	79.7±.1	87.5±.1
ConMix	68.8±.2	79.5±.1	**73.1**±.1	**82.8**±.1	**81.3**±.1	**88.5**±.1	**82.1**±.1	**89.1**±.1	**81.2** ±.0	**88.5**±.0
Dataset: Advising DSTC7										
Augmentation	truncation		deletion		reordering		typo		synonym	
in training	Rec@1	MRR	Rec@1	MRR	Rec@1	MRR	Rec@1	MRR	Rec@1	MRR
NA (Bi-encoder)	18.4±1.4	29.2±.5	15.5±.1	26.1±.1	14.8±.3	25.7±.4	18.3±.7	30.3±.4	19.6±.8	30.7±.6
NA (Poly-encoder)	17.5±.9	28.5±1.4	**17.9**±1.5	**29.7**±1.3	15.0±.8	26.4±1.0	13.1±.4	25.1±.0	17.0±2.5	28.6±2.3
Subsequence	**19.7**±.1	**31.7**±.1	16.3±.7	27.0±.4	15.4±.8	26.3±.2	18.9±.7	31.1±.5	18.0±.8	29.9±.6
Deletion	18.6±.8	30.4±.1	17.6±.3	29.4±.2	16.6±.0	28.5±.0	19.3±1.3	32.2±.9	18.6±.0	31.8±.4
Reordering	19.0±.0	29.6±.4	15.9±1.8	27.8±1.3	17.1±.1	**30.4**±.5	18.7±1.3	31.9±.6	18.1±1.0	31.0±.3
Replacement	17.7±.8	28.6±.2	14.6±.5	24.8±.9	12.7±.7	23.6±1.1	18.0±.3	29.7±.1	16.3±.1	28.2±.1
ConMix	18.6±.0	29.8±.2	16.2±.6	28.0±.1	**18.3**±.7	30.2±.2	**19.6**±1.4	**32.7**±.5	**20.9**±1.0	**33.2** ±.9
Dataset: Ubuntu DSTC7										
Augmentation	truncation		deletion		reordering		typo		synonym	
in training	Rec@1	MRR	Rec@1	MRR	Rec@1	MRR	Rec@1	MRR	Rec@1	MRR
NA (Bi-encoder)	42.3±.1	51.7±.3	52.3±.0	61.9±.3	47.9±.6	57.6±.3	48.0±.0	58.2±.4	51.6±.4	61.5±.2
NA (Poly-encoder)	41.2±.8	51.3±.3	49.6±1.1	60.1±.9	47.9±.5	57.5±.2	45.2±.1	56.2±.1	50.6±.4	61.4±.1
Subsequence	**45.7**±.2	**55.8**±.2	47.6±.6	58.3±.2	47.5±.8	58.3±.4	50.8±.3	61.8±.3	52.3±.9	62.6±.4
Deletion	42.4±1.4	52.7±.6	**54.6**±.2	**64.3**±.3	50.3±.3	60.8±.2	53.5±.1	63.5±.1	52.9±.6	63.6±.0
Reordering	41.6±.3	51.2±.6	50.1±.9	59.7±.7	52.8±.3	62.8±.2	51.5±.4	61.7±.3	51.7±.4	62.0±.4
Replacement	43.9±1	53.6±.9	49.7±1	59.9±1	52.1±2.5	62.3±1.7	54.2±.4	**64.4**±.2	**55.0**±.7	**64.7**±.5
ConMix	44.0±.6	52.8±1	50.5±.4	60.8±.5	**54.1**±.5	**63.8**±.3	**54.5**±.4	64.2±.5	54.5±.2	64.1±.2
Dataset: Taskmaster										
Augmentation	truncation		deletion		reordering		typo		synonym	
in training	Rec@1	MRR	Rec@1	MRR	Rec@1	MRR	Rec@1	MRR	Rec@1	MRR
NA (Bi-encoder)	76.5±.5	80.8±.4	79.5±.1	84.2±.1	76.5±.2	82.1±.2	87.6±.3	89.6±.2	84.6±.2	87.7±.1
NA (Poly-encoder)	77.1±.0	81.3±.0	79.6±.4	84.2±.2	76.5±.3	82.0±.2	88.0±.1	89.8±.1	84.8±.2	87.8±.1
Subsequence	**85.7**±.2	**88.1**±.1	79.9±.2	84.4±.2	74.5±.8	80.4±.5	87.6±.2	89.5±.1	84.2±.4	87.4±.2
Deletion	76.4±.0	80.6±.0	**86.4**±.2	**88.8**±.1	86.0±.3	88.5±.2	88.5±.2	90.2±.1	86.5±.3	88.9±.2
Reordering	75.9±.3	80.2±.2	83.9±.2	87.2±.1	**89.1**±.3	**90.6**±.2	88.0±.1	89.9±.1	86.3±.1	88.8±.1
Replacement	74.5±.4	79.5±.2	77.0±.5	82.4±.3	71.9±1.0	78.6±.6	86.7±.4	88.4±.3	82.3±.9	86.2±.6
ConMix	81.3±.4	81.3±.3	85.0±.3	87.9±.2	88.2±.3	90.1±.2	**90.1**±.3	**91.3**±.2	**89.3**±.3	**90.8**±.2

Table 3: Robustness during inference for different augmentation strategies. All models using augmentions were trained with contrastive loss. Results were averaged over three runs, and ± denotes the standard deviation.

mon and realistic variations, i.e., synonyms and typos, ConMix significantly outperforms all other methods on three datasets. This indicates a uniformly powerful and robust representation learning method through this novel augmentation strategy.

5.3 Computational Efficiency

ConMix is designed and implemented to generate augmentations through vectorization and therefore has the benefit of being faster to train. Tokens are randomly mixed on-the-fly within a batch to create augmentations in parallel on GPUs, through fast tensor multiplications. For many augmentation methods, such vectorization might be non-trivial and the overall speed becomes limited by the process of creating augmentations outside the training loop on much slower CPUs. For example, when training on the Taskmaster dataset with 8 gpus, the DialAug architecture with ConMix is 1.2x faster than training with the global word replacement augmentation. While conducting full training over 20 epochs this leads to an overall speed up by 1.5 hours for training with the ConMix augmentation.

6 Summary

In this work we proposed DialAug, a modular architecture for conversational response ranking. It combines the traditional cross-entropy loss for ranking with a contrastive counterpart to learn from augmented views of the dialogue context. We presented a novel data augmentation method, ConMix, which generates multiple views of the same con-

text via stochastic mixing of tokens from other contexts in the batch during training. We conducted an extensive set of experiments on four datasets and show that a model trained with ConMix outperforms strong baselines and other augmentation methods. Our proposed model is also proven to be robust against common perturbations encountered during inference. We hope our work encourages further research in such data-centric methods to improve robustness of NLP models for practical applications of conversational modeling.

References

Christopher Beckham, Sina Honari, Vikas Verma, Alex Lamb, Farnoosh Ghadiri, R Devon Hjelm, Yoshua Bengio, and Christopher Pal. 2019. On adversarial mixup resynthesis. *arXiv preprint arXiv:1903.02709*.

B. Byrne, K. Krishnamoorthi, Chinnadhurai Sankar, Arvind Neelakantan, Daniel Duckworth, Semih Yavuz, Ben Goodrich, Amit Dubey, A. Cedilnik, and Kyu-Young Kim. 2019. Taskmaster-1: Toward a realistic and diverse dialog dataset. In *EMNLP/IJCNLP*.

Ting Chen, Simon Kornblith, Mohammad Norouzi, and Geoffrey Hinton. 2020. A simple framework for contrastive learning of visual representations. In *International conference on machine learning*, pages 1597–1607. PMLR.

J. Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. Bert: Pre-training of deep bidirectional transformers for language understanding. In *NAACL*.

Alexander Richard Fabbri, Simeng Han, Haoyuan Li, Haoran Li, Marjan Ghazvininejad, Shafiq Joty, Dragomir Radev, and Yashar Mehdad. 2021. Improving zero and few-shot abstractive summarization with intermediate fine-tuning and data augmentation. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 704–717.

Hongchao Fang and P. Xie. 2020. Cert: Contrastive self-supervised learning for language understanding. *ArXiv*, abs/2005.12766.

Steven Y. Feng, Varun Gangal, Jason Wei, Sarath Chandar, Soroush Vosoughi, T. Mitamura, and E. Hovy. 2021. A survey of data augmentation approaches for nlp. *ArXiv*, abs/2105.03075.

Tianyu Gao, Xingcheng Yao, and Danqi Chen. 2021. Simcse: Simple contrastive learning of sentence embeddings. *ArXiv*, abs/2104.08821.

John Michael Giorgi, Osvald Nitski, Gary D Bader, and Bo Wang. 2020. Declutr: Deep contrastive learning for unsupervised textual representations. *ArXiv*, abs/2006.03659.

Jia-Chen Gu, Tianda Li, Quan Liu, Xiao-Dan Zhu, Zhenhua Ling, Zhiming Su, and Si Wei. 2020. Speaker-aware bert for multi-turn response selection in retrieval-based chatbots. *Proceedings of the 29th ACM International Conference on Information & Knowledge Management*.

Chulaka Gunasekara, Jonathan K Kummerfeld, Lazaros Polymenakos, and Walter Lasecki. 2019. Dstc7 task 1: Noetic end-to-end response selection. In *Proceedings of the First Workshop on NLP for Conversational AI*, pages 60–67.

Beliz Gunel, Jingfei Du, Alexis Conneau, and Veselin Stoyanov. 2020. Supervised contrastive learning for pre-trained language model fine-tuning. In *International Conference on Learning Representations*.

Janghoon Han, Taesuk Hong, Byoungjae Kim, Youngjoong Ko, and Jungyun Seo. 2021. Fine-grained post-training for improving retrieval-based dialogue systems. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 1549–1558.

Matthew Henderson, Ivan Vulic, D. Gerz, I. Casanueva, Paweł Budzianowski, Sam Coope, Georgios P. Spithourakis, Tsung-Hsien Wen, N. Mrksic, and Pei hao Su. 2019. Training neural response selection for task-oriented dialogue systems. In *ACL*.

Samuel Humeau, Kurt Shuster, Marie-Anne Lachaux, and Jason Weston. 2019. Poly-encoders: Architectures and pre-training strategies for fast and accurate multi-sentence scoring. In *International Conference on Learning Representations*.

Robin Jia, Aditi Raghunathan, Kerem Göksel, and Percy Liang. 2019. Certified robustness to adversarial word substitutions. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 4129–4142.

Mohammad Kachuee, Hao Yuan, Young-Bum Kim, and Sungjin Lee. 2021. Self-supervised contrastive learning for efficient user satisfaction prediction in conversational agents. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 4053–4064.

Prannay Khosla, Piotr Teterwak, Chen Wang, Aaron Sarna, Yonglong Tian, Phillip Isola, Aaron Maschinot, Ce Liu, and Dilip Krishnan. 2020. Supervised contrastive learning. *Advances in Neural Information Processing Systems*, 33.

Diederik P. Kingma and Jimmy Ba. 2015. Adam: A method for stochastic optimization. *CoRR*, abs/1412.6980.

Ashutosh Kumar, S. Bhattamishra, Manik Bhandari, and P. Talukdar. 2019. Submodular optimization-based diverse paraphrasing and its effectiveness in data augmentation. In *NAACL-HLT*.

Jonathan K. Kummerfeld, S. R. Gouravajhala, Joseph Peper, V. Athreya, R. Chulaka Gunasekara, Jatin Ganhotra, S. Patel, L. Polymenakos, and Walter S. Lasecki. 2018. Analyzing assumptions in conversation disentanglement research through the lens of a new dataset and model. *ArXiv*, abs/1810.11118.

Shayne Longpre, Yu Wang, and Chris DuBois. 2020. How effective is task-agnostic data augmentation for pretrained transformers? In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: Findings*, pages 4401–4411.

Ryan Lowe, Nissan Pow, Iulian Vlad Serban, and Joelle Pineau. 2015. The ubuntu dialogue corpus: A large dataset for research in unstructured multi-turn dialogue systems. In *Proceedings of the 16th Annual Meeting of the Special Interest Group on Discourse and Dialogue*, pages 285–294.

Y. Lu, Manisha Srivastava, Jared Kramer, Heba Elfardy, Andrea Kahn, Song Wang, and Vikas Bhardwaj. 2019. Goal-oriented end-to-end conversational models with profile features in a real-world setting. In *NAACL*.

Xiaofei Ma, C. D. Santos, and Andrew O. Arnold. 2021. Contrastive fine-tuning improves robustness for neural rankers. In *EMNLP FINDINGS*.

Shikib Mehri, E. Razumovskaia, Tiancheng Zhao, and M. Eskénazi. 2019. Pretraining methods for dialog context representation learning. In *ACL*.

Kevin Musgrave, Serge Belongie, and Ser-Nam Lim. 2020. Pytorch metric learning.

Marcin Namysl, Sven Behnke, and Joachim Köhler. 2020. Nat: Noise-aware training for robust neural sequence labeling. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 1501–1517.

Nils Reimers and Iryna Gurevych. 2019. Sentence-bert: Sentence embeddings using siamese bert-networks. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 3982–3992.

Dinghan Shen, Ming Zheng, Yelong Shen, Yanru Qu, and Weizhu Chen. 2020. A simple but tough-to-beat data augmentation approach for natural language understanding and generation. *ArXiv*, abs/2009.13818.

Connor Shorten and T. Khoshgoftaar. 2019. A survey on image data augmentation for deep learning. *Journal of Big Data*, 6:1–48.

Jason Wei, Chengyu Huang, Soroush Vosoughi, Yu Cheng, and Shiqi Xu. 2021. Few-shot text classification with triplet networks, data augmentation, and curriculum learning. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 5493–5500.

Taesun Whang, Dongyub Lee, Chanhee Lee, Kisu Yang, Dongsuk Oh, and Heuiseok Lim. 2020. An effective domain adaptive post-training method for bert in response selection. In *INTERSPEECH*.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, R'emi Louf, Morgan Funtowicz, and Jamie Brew. 2020. Transformers: State-of-the-art natural language processing. In *EMNLP*.

Thomas Wolf, Victor Sanh, Julien Chaumond, and Clement Delangue. 2019. Transfertransfo: A transfer learning approach for neural network based conversational agents. *CoRR*, abs/1901.08149.

Chien-Sheng Wu, S. Hoi, R. Socher, and Caiming Xiong. 2020a. Tod-bert: Pre-trained natural language understanding for task-oriented dialogue. In *EMNLP*.

Z. Wu, Sinong Wang, Jiatao Gu, Madian Khabsa, Fei Sun, and Hao Ma. 2020b. Clear: Contrastive learning for sentence representation. *ArXiv*, abs/2012.15466.

Qizhe Xie, Zihang Dai, Eduard Hovy, Thang Luong, and Quoc Le. 2020. Unsupervised data augmentation for consistency training. *Advances in Neural Information Processing Systems*, 33.

Ruijian Xu, Chongyang Tao, Daxin Jiang, Xueliang Zhao, Dongyan Zhao, and Rui Yan. 2021. Learning an effective context-response matching model with self-supervised tasks for retrieval-based dialogues. In *AAAI*.

A Closer Look at Few-Shot Out-of-Distribution Intent Detection

Li-Ming Zhan[1] **Haowen Liang**[1*] **Lu Fan**[1*] **Xiao-Ming Wu**[1†]
Albert Y.S. Lam[2]
Department of Computing, The Hong Kong Polytechnic University, Hong Kong S.A.R.[1]
Fano Labs, Hong Kong S.A.R.[2]
{lmzhan.zhan, michael.liang}@connect.polyu.hk
{cslfan, csxmwu}@comp.polyu.edu.hk, albert@fano.ai

Abstract

We consider few-shot out-of-distribution (OOD) intent detection, a practical and important problem for the development of task-oriented dialogue systems. Despite its importance, this problem is seldom studied in the literature, let alone examined in a systematic way. In this work, we take a closer look at this problem and identify key issues for research. In our pilot study, we reveal the reason why existing OOD intent detection methods are not adequate in dealing with this problem. Based on the observation, we propose a promising approach to tackle this problem based on latent representation generation and self-supervision. Comprehensive experiments on three real-world intent detection benchmark datasets demonstrate the high effectiveness of our proposed approach and its great potential in improving state-of-the-art methods for few-shot OOD intent detection. The source code can be found at https://github.com/liam0949/Few-shot-Intent-OOD.

1 Introduction

Intent detection is an important component of task-oriented dialogue system, which aims at accurately identifying the intent behind user utterances. Out-of-distribution (OOD) intent detection aims to solve a $(K+1)$-way classification problem with K in-distribution (ID) intent classes and an additional OOD class representing malformed or unsupported queries. In practice, OOD intent detection is often performed in data-scarcity scenarios, e.g., at the early development stage of a dialogue system when labeled data is not sufficient, or for dialogue systems developed for minority language users where it is difficult to find suitable annotators.

Despite its practical importance, few-shot OOD intent detection is a highly challenging problem, which is seldom studied in the literature and has not been investigated in a systematic way. Recent advances in OOD intent detection (Zhang et al., 2021a; Zhan et al., 2021; Lin and Xu, 2019) commonly assume that there are adequate ID examples available for training, without considering the few-shot scenario. To our best knowledge, the only work on this topic is by Zhang et al. (2020), who try to tackle few-shot OOD intent detection via transfer learning by fine-tuning RoBERTa (Liu et al., 2019) on large-scale natural language inference datasets.

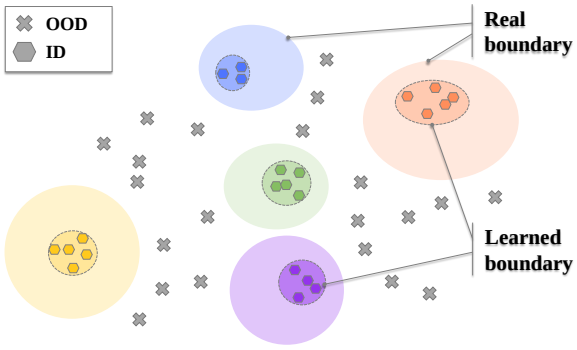

Figure 1: The challenge of few-shot out-of-distribution intent detection. OOD stands for out-of-distribution examples and ID stands for in-distribution examples.

In this work, we take a closer look at few-shot OOD intent detection and consider a strict setting, where only few-shot in-distribution labeled examples are available during training and no external resources can be exploited, since the requirement of additional resources hinders the applicability of the model. Under this simplified yet more challenging setting, state-of-the-art OOD intent detection algorithms fail to achieve acceptable performance. In Figure 1, we illustrate the key challenge for few-shot OOD detection. As shown in Figure 1, since ID classes are under-represented by few-shot ID examples, a model based on density estimation (Zhang et al., 2021a) or $(K+1)$-way discriminative training (Zhan et al., 2021) tends to learn a conservative decision boundary and hence there

*Equal contribution.
† Corresponding author.

Proceedings of the 29th International Conference on Computational Linguistics, pages 451–460
October 12–17, 2022.

are large margins between the real and learned decision boundaries. Real ID examples situate in the margins will be inaccurately assigned to the OOD class, leading to poor performance.

Therefore, the key for few-shot OOD intent detection is to improve the model performance on ID examples. To address this issue, we propose to enrich the training set to improve the representativeness of ID intent classes and provide more useful learning signals. We explore the feasibility of generating synthetic ID examples in a self-supervised manner. In particular, we train a denoising autoencoder (DAE) (Vincent et al., 2008) in the latent representation space only using the few labeled ID examples. The trained decoder of DAE is then used to efficiently sample synthetic ID examples. With the enlarged training set, we follow Zhan et al. (2021) to train a $(K+1)$-way classifier by simulating OOD examples with the enlarged training set. Our contributions are summarized as follows:

- We pioneer in studying a practical but more challenging few-shot OOD intent detection problem and identifying the key challenge for this problem.
- We propose a promising approach for solving few-shot OOD intent detection based on latent representation generation and $(K+1)$-way discriminative training, which requires no additional resources for training and validation.
- We conduct comprehensive experiments on three realistic intent detection datasets to verify the effectiveness and robustness of our method in diverse few-shot OOD intent detection scenarios.

2 Related Work

Out-of-Distribution Intent Detection. Out-of-distribution (OOD) intent detection (or out-of-domain intent detection) has attracted much attention in research communities, due to its significant importance to the robustness of dialogue systems. The primary challenge of this task is that there is no labeled OOD example available for training and validation. As such, the majority of OOD intent detection algorithms relies on manually selecting an appropriate threshold.

The first line of works (Hendrycks and Gimpel, 2017; Shu et al., 2017; Ryu et al., 2018, 2017) uses some statistic as the confidence score of whether an example is OOD or not. Hendrycks and Gimpel (2017) pointed out that the negative probability outputted by the softmax function can be a good confidence metric for OOD detection. Shu et al. (2017) defined a binary classification task for every in-domain class and used the maximum probability among all these binary classifiers as the confidence score. Ryu et al. (2018) developed an adversarial training strategy inspired by GAN for OOD intent detection. The discriminator in GAN was trained to assign lower scores to OOD examples. Ryu et al. (2017) employed an autoencoder trained on in-domain examples and used the reconstruction score as the OOD indicator. However, all these methods require manual effort in selecting a proper threshold for OOD discrimination.

The second line of works (Lin and Xu, 2019; Zhang et al., 2021a; Yan et al., 2020) proposes to learn decision boundaries for OOD examples under some assumption of data distribtuion, e.g., mixture of Gaussians. OOD examples are assumed to lie in the low-density areas of utterance distribution. Yan et al. (2020) proposed to model the in-domain examples by a mixture of Gaussians distribution and select a margin to constrain the variance of each in-domain Gaussian component. Zhang et al. (2021a) also made the mixture of Gaussian assumption on in-domain data distribution but proposed to automatically learn the variance of the Gaussian components.

Different from previous methods, a recent work by Zhan et al. (2021) proposed to directly learn a $(K+1)$-way classifier in an end-to-end manner. They created OOD learning signals during training by leveraging external data or constructing simulated OOD examples with self-supervised information.

Few-shot OOD Intent Detection. Few-shot OOD intent detection considers OOD intent detection in low-resource scenarios. It aims at developing a reliable OOD detector with only a few examples per each in-distribution class. Undoubtedly, this is a highly challenging task given that few-shot intent detection is already a big challenge (Zhang et al., 2020). At this point, this task is under-explored and has never been investigated in a strictly low-resourced setting. The most related work is DNNC proposed in Zhang et al. (2020), which tries to mitigate the data-scarcity problem by fine-tuning RoBERTa on external large natural language inference datasets. In this paper, however,

we consider using the few-shot labeled examples as the only training resource.

General-purpose Few-shot OOD Detection. There is also little research on general-purpose few-shot OOD detection. To our knowledge, recent works are Jeong and Kim (2020) and Wang et al., both of which adopt episodic training on a large set of few-shot classification tasks for transfer learning. Clearly, this is very different from the problem setting of this paper, as we do not use training resources other than the given few-shot labeled examples.

3 Problem Statement and Pilot Study

Out-of-distribution (OOD) intent detection aims at improving the robustness of a dialogue system with respect to utterances with unknown (or unsupported) intents. The key challenge of OOD detection is that real OOD samples are inaccessible during training and validation. Given an in-distribution (ID) set of K known classes, $y_i \in \{y^k\}_{k=1}^K$, the OOD detection task considers another special OOD class y^{OOD} to represent any malformed or unsupported utterances. Hence, given the input space $\mathcal{X} \times \mathcal{Y}$, the goal of OOD intent detection is to learn a $(K+1)$-way classifier $f_\phi(\cdot) : \mathcal{X} \rightarrow \mathcal{Y}$ to minimize the expected risk:

$$R(f) = \mathbb{E}(\mathbb{1}[f_\phi(x_i) \neq y_i)]), \quad (1)$$

where $y_i \in \{y^1, \cdots, y^K, y^{\text{OOD}}\}$ and the expectation is taken over the joint distribution of $p(x, y)$. $\mathbb{1}$ is an indicator function.

Few-shot OOD intent detection is a more challenging setting with the assumption that there are only a few labeled in-distribution (ID) examples available during training. In this paper, we consider a strict but practical setting by assuming that there are no additional resources (e.g., labeled or unlabeled auxiliary datasets) available to aid the training of the classifier $f_\phi(\cdot)$ or during fine-tuning pre-trained language models. Typically, for each ID class in $\{y^k\}_{k=1}^K$, there are only ~ 5 or ~ 10 labeled examples per class.

Pilot study. To illustrate the challenges of few-shot OOD intent detection, we conduct a pilot study on a commonly used OOD intent detection dataset CLINC150 (Larson et al., 2019) using two recent state-of-the-art approaches (Zhang et al., 2021a; Zhan et al., 2021) for few-shot OOD intent detection. To simulate the few-shot scenario, in the experiment, only 5 labeled examples in each ID class are used for training. The results are summarized in Table 1. For OOD detection, we randomly select 25%, 50% and 75% intent classes as in-domain classes and assign the remaining classes to the OOD category. Experimental details are elaborated in Section 5.

	Methods	Acc.	Macro-F1	ID-F1	OOD-F1
25%	ADB	77.91	53.09	52.22	86.29
	DCL	86.53	48.78	47.63	92.22
50%	ADB	69.36	56.91	56.64	77.17
	DCL	74.60	50.58	50.15	82.45
75%	ADB	70.43	67.17	67.12	73.09
	DCL	65.50	54.25	54.11	70.22

Table 1: A pilot study on few-shot OOD intent detection. DCL (Zhan et al., 2021) and ADB (Zhang et al., 2021a) are two recent state-of-the-art approaches for OOD intent detection. ID-F1 indicates macro f1-score on the in-distribution classes. OOD-F1 stands for f1-score on the out-of-distribution class.

We can observe that both of the two methods yield unsatisfactory performance. Specifically, the performance on the ID classes is poor and way lower than that on the OOD class. When there are only 25% ID classes (~ 38), the gap between the ID and OOD classes in f1-score is the largest (up to 44+). Although moderate overall accuracy is achieved, such OOD intent detection model can only provide services to users worse than random choices, since the majority of user utterances are rejected as OOD inputs. It also indicates that the overall accuracy may not be a good performance measure for this task. These observations show that in the few-shot scenario, existing OOD intent detection algorithms can be easily biased towards the OOD class, due to inadequate representations of the ID classes. Hence, directly applying them to few-shot OOD intent detection will lead to sub par performance.

The primary challenge identified from this pilot experiment for few-shot OOD intent detection is then how to improve the performance on in-distribution classes and achieve a good balance in performance between ID and OOD classes.

4 Methodology

4.1 Utterance Representation

Let $\mathcal{D} = \{(x_i, y_i)\}_{i=1}^N$ be the training set, where x_i denotes an input token sequence with size m, i.e., $[x_i^0, \cdots, x_i^{m-1}]$. For each input x_i, we use BERT as the encoder to map x_i into a sequence

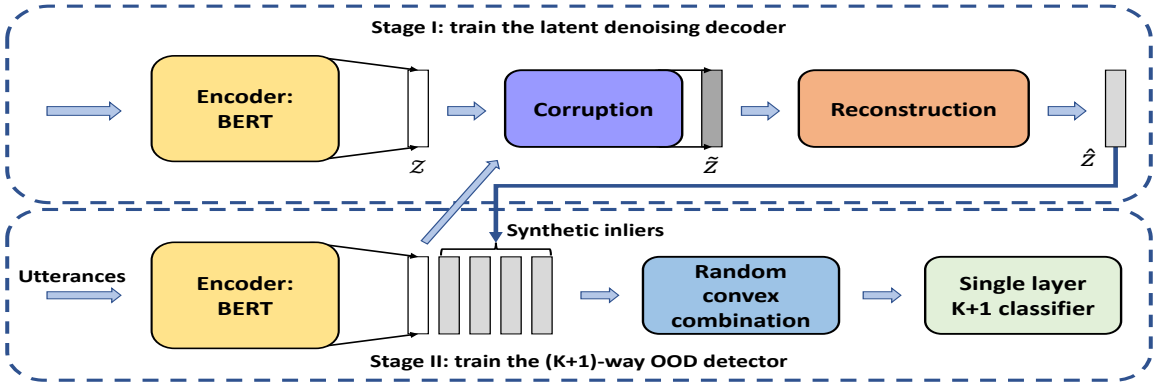

Figure 2: An overview of our proposed framework.

of hidden states h_i, i.e., BERT:$\mathcal{X} \rightarrow \mathcal{H}$ and $h_i \in \mathbb{R}^{(m+1)*768}$. Note that for every sentence, BERT adds a spacial token [CLS] at the beginning of the sequence. Following common practice, we use the average pooling of the hidden sequence h_i as the representation of an utterance:

$$z_i = \text{Avg.Pool}([h_i^{\text{CLS}}, h_i^0, \cdots, h_i^{m-1}]).$$

Then, we obtain a mapped training set $\mathcal{D}^{tr} = \{(z_i, y_i)\}_{i=1}^N$. We instantiate few-shot OOD detector $f_\phi(\cdot)$ by replacing the pre-trained heads of BERT with a simple linear mapping layer.

4.2 Our Proposed Model

As shown in Figure 2, we propose a two-stage model for few-shot OOD intent detection. In the first stage, we learn a stochastic reconstruction function to generate synthetic ID samples in the representation space to enrich the in-distribution training set. In the second stage, we adopt a $(K+1)$-way discriminative training procedure for OOD detection by simulating OOD examples based on the enlarged in-distribution training set. Notice that throughout the two stages, we only use the few labeled in-distribution data without exploiting external labeled intent detection data or fine-tuning corpus.

4.2.1 Stage I: Generating Synthetic In-distribution Data

To improve the performance of in-distribution (ID) classes, our solution is to learn a latent denoising autoencoder (DAE) (Vincent et al., 2008) in the latent representation space $\mathcal{Z}$ of BERT, to enrich the in-domain training set by generating synthetic examples with the reconstructor of the DAE.

Our key idea is to learn an approximator for the distribution of the latent representation of ID utterances ($p(z)$), from which we can sample synthetic ID examples. We aim to learn a generator with sampling efficiency and guaranteed consistency in approximating the true distribution as the training size $N \rightarrow \infty$. We can thereby enrich the ID training examples directly in the representation space $\mathcal{Z}$ and save the effort of conducting data augmentation in the input space $\mathcal{X}$.

To this end, we employ a principled distribution estimation method – denoising autoencoder (DAE) – to build an efficient stochastic process for sampling ID examples with a consistency guaranteed estimator for $p(z)$. The latent DAE consists of two components: the corruption distribution $\mathcal{C}(\tilde{z} \mid z)$ and the reconstruction distribution $q_\theta(z \mid \tilde{z})$. The DAE can be learned by:

$$\theta^* = \underset{\theta}{\text{argmax}}\, \mathbb{E}(\log(q_\theta(z \mid \tilde{z})),$$

where the likelihood is computed by a mean square loss between the original embedding vector z and the reconstructed vector $\hat{z}$ as shown in Figure 2.

After obtaining the reconstruction distribution $q_{\theta^*}(z \mid \tilde{z})$, we can sample synthetic ID examples as follows:

$$\begin{aligned} \hat{z} &\sim q_{\theta^*}(z \mid \tilde{z}), \\ \tilde{z} &\sim \mathcal{C}(\tilde{z} \mid z). \end{aligned} \tag{2}$$

The corruption distribution $\mathcal{C}$ can be instantiated by simple stochastic operations like Dropout (Srivastava et al., 2014). By repeatedly applying the process in Equation (2), we can obtain a synthetic labeled ID set $\mathcal{D}^{\text{rec}} = \{(\hat{z}_i, y_i)\}_{i=1}^L$, where the reconstructed representation $\hat{z}_i$ shares the same label y_i with the original uncorrupted z_i. Finally, by combining the original training set $\mathcal{D}^{tr}$ and the synthetic set $\mathcal{D}^{\text{rec}}$, we get an enlarged labeled training set $\mathcal{D}^{\text{Enlarged}} = \mathcal{D}^{tr} \cup \mathcal{D}^{\text{rec}}$.

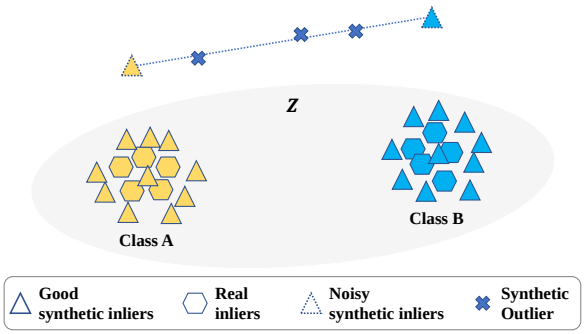

Figure 3: Illustration of the noise neutralizing effect under the $(k+1)$-way training paradigm.

4.2.2 Stage II: $(K+1)$-way Discriminative Training

As shown in the Figure 2, the second stage of our proposed method aims at learning a $(K+1)$-way classifier in an end-to-end manner. Since only few-shot samples are used to train the reconstruction distribution $q_\theta(z \mid \tilde{z})$, the resulting $q_{\theta^*}(\cdot)$ may not be a perfect estimator for the true distribution, and the enlarged in-distribution set $\mathcal{D}^{\text{Enlarged}}$ may be noisy. Hence, it may not be the best choice to directly apply density estimation-based methods for OOD intent detection, due to the risk of overfitting.

To better utilize the enlarged in-distribution set $\mathcal{D}^{\text{Enlarged}}$, we adopt the $(K+1)$-way discriminative training strategy proposed in Zhan et al. (2021) and follow their idea to construct OOD learning signals via random convex combination between representations from different in-distribution classes in the enlarged in-distribution set. By doing so, the impact of noisy synthetic in-distribution examples can be mitigated. We demonstrate this phenomenon in Figure 3. The linear interpolation between off-manifold noisy synthetic in-distribution examples tends to represent the OOD examples, since the word embeddings of BERT has been found concentrating near a low-dimensional manifold of the representation space (Ethayarajh, 2019).

Specifically, given the enlarged training set $\mathcal{D}^{\text{Enlarged}}$, we construct an OOD set $\mathcal{D}^{\text{OOD}}$ by:

$$z_i^{OOD} = \alpha * z_i + (1-\alpha) * z_j, \qquad (3)$$

where $y^i \neq y^j$, $\alpha \in [0,1]$ is randomly sampled from $U(0,1)$ and $z_i,\ z_j \in \mathcal{D}^{\text{Enlarged}}$.

Finally, our $(K+1)$-way classifier can be learned by minmizing the loss in Equation (1) on the union set $\mathcal{D}^{\text{OOD}} \cup \mathcal{D}^{\text{Enlarged}}$.

5 Experiments

To evaluate our proposed method for few-shot out-of-distribution (OOD) intent detection, we conduct extensive experiments on three real-world benchmark datasets. By comparing with state-of-the-art OOD intent detection methods, we find that our method can outperform these baselines by a large margin, especially in extreme few-shot scenarios. Moreover, our approach yields a more consistent performance at different few-shot OOD settings, demonstrating the robustness of our algorithm.

5.1 Datasets and Baselines

We evaluate our method on three commonly used OOD intent detection datasets, which are introduced as follows.

- **CLINC150** (Larson et al., 2019) is specifically designed for OOD intent detection. It consists of 150 in-distribution classes with 15,000 samples for training, 3,000 for validation, and 4,500 for testing. Besides, it also contains 1,200 annotated OOD instances, and we put all the OOD examples into the test set.

- **Banking** (Casanueva et al., 2020) contains data from the banking domain, with 13,083 samples of 77 intents. We split the dataset into 9,003 for training, 1,000 for validation, and 3,080 for testing.

- **StackOverflow** (Xu et al., 2015) contains data in 20 classes, each of which contains 1,000 samples. We use 12,000 samples for training, 2,000 for validation, and 6,000 for testing.

The dataset statistics are summarised in Table 2.

To evaluate the effectiveness of our proposed method, we compare it with the following baselines.

- **MSP** (Hendrycks and Gimpel, 2017): It leverages the probabilities outputted by the softmax function for out-of-domain detection. As correct samples tend to have higher probability scores, samples below a threshold are classified as outliers. We set the threshold as 0.5 in our experiment.

- **DOC** (Shu et al., 2017): It shares a similar idea with MSP in assuming that in-distribution examples tend to have higher probability scores. It uses the maximum probability

Dataset (proportion)	# Vocab	Avg. Length	# Training	# Class	Avg. Sample per Class (5%)	(10%)
CLINC150	5864	8.34	15000	150	5	10
Banking	4327	11.99	9003	77	6	12
StackOverflow	16519	8.35	12000	20	30	60

Table 2: Dataset statistics.

dataset	CLINC150			Banking			StackOverflow		
p=5%	0.25	0.5	0.75	0.25	0.5	0.75	0.25	0.5	0.75
MSP	40.13	55.17	54.76	17.74	29.31	31.99	52.30	42.92	78.92
DOC	11.05	8.62	44.37	15.79	25.61	20.98	65.54	44.4	58.54
SEG	36.09	51.90	62.64	39.53	52.27	58.80	60.76	75.93	83.22
LMCL	34.30	52.45	60.71	39.10	48.90	54.60	56.00	69.68	83.17
Softmax	33.98	52.48	62.11	32.77	43.74	52.84	54.21	71.27	81.55
ADB	53.09	56.91	65.65	37.74	45.91	55.26	60.31	77.92	81.14
DCL	48.78	50.58	54.25	33.92	39.10	45.59	78.98	82.37	83.01
Ours	**62.19**	**64.79**	**68.30**	**48.23**	**58.92**	**63.14**	**80.48**	**84.04**	**84.25**
dataset	CLINC150			Banking			StackOverflow		
p=10%	0.25	0.5	0.75	0.25	0.5	0.75	0.25	0.5	0.75
MSP	54.34	71.56	77.31	43.50	48.62	68.34	41.66	59.73	75.95
DOC	15.15	23.28	54.69	13.99	21.50	25.13	44.77	61.22	61.19
SEG	68.29	77.59	80.32	56.75	58.70	71.32	58.77	78.64	83.85
LMCL	66.87	76.48	79.04	54.38	63.71	67.66	55.42	77.01	85.06
Softmax	65.07	77.08	79.68	53.27	60.20	68.94	57.86	77.30	83.47
ADB	68.05	74.96	77.75	51.12	66.16	70.50	69.55	81.30	83.83
DCL	68.65	72.74	70.81	55.74	61.10	65.77	78.61	82.46	83.80
Ours	**72.43**	**78.15**	**82.17**	**60.99**	**67.89**	**73.79**	**81.07**	**83.99**	**85.11**

Table 3: Overall macro f1-score including the OOD class for few-shot OOD intent detection with different proportion (0.25, 0.5 and 0.75) of in-distribution classes. p indicates the ratio of selected few-shot in-distribution examples. For each setting, the best result is marked in bold.

from m 1-vs-rest sigmoid classifiers for m ID classes respectively as the confidence score.

- **LMCL** (Lin and Xu, 2019): It leverages local outlier factor(LOF) to identity samples which are far away from the clusters in the embedding space as outliers. The model learns discriminative features by largin margin cosine loss.

- **Softmax** (Lin and Xu, 2019): It is a variant of LMCL where the large margin cosine loss is replaced by the softmax loss to learn discriminative features.

- **SEG** (Yan et al., 2020): It uses a Gaussian mixture model to enforce ID embeddings to form ball-like dense clusters in the feature space. Moreover, it injects semantic information into the Gaussian mixture model by assigning the embeddings of class labels or descriptions to be the means of the Gaussians.

- **ADB** (Zhang et al., 2021a): It proposes to learn a decision boundary for each in-domain class for OOD intent detection. Samples reside outside of the boundaries are identified as outliers, while in-distribution examples are classified based on their distance to centroids of each class.

- **DCL** (Zhan et al., 2021): It treats outliers as an additional class and proposes a $K+1$ training paradigm for OOD intent detection. Samples in the outlier class are obtained from external datasets and synthesized through convex combinations of in-distribution features.

dataset	CLINC150			Banking			StackOverflow		
p=5%	0.25	0.5	0.75	0.25	0.5	0.75	0.25	0.5	0.75
MSP	38.85	54.85	54.63	14.38	28.32	31.79	55.15	40.97	80.44
DOC	8.99	7.72	44.18	12.35	24.48	20.62	62.89	42.89	58.92
SEG	36.88	52.50	63.18	39.30	52.83	58.80	60.65	76.11	84.06
LMCL	35.20	53.14	61.24	37.15	49.41	55.02	55.15	71.51	84.17
Softmax	34.68	53.10	62.61	33.56	44.22	53.26	54.25	72.36	82.65
ADB	52.22	56.64	65.58	35.14	45.54	55.36	77.51	77.92	81.97
DCL	47.63	50.15	54.11	31.1	38.22	45.55	76.31	81.92	83.79
Ours	**61.43**	**64.54**	**68.25**	**48.82**	**58.51**	**63.32**	**78.05**	**83.74**	**84.99**
dataset	CLINC150			Banking			StackOverflow		
p=10%	0.25	0.5	0.75	0.25	0.5	0.75	0.25	0.5	0.75
MSP	53.77	71.50	77.39	41.97	48.21	68.69	45.89	61.02	78.29
DOC	13.21	22.57	54.57	10.39	20.33	24.84	43.69	60.43	61.60
SEG	68.29	77.52	80.34	56.75	58.69	71.61	59.24	78.64	83.85
LMCL	66.40	76.47	79.05	53.77	63.91	68.07	55.40	77.26	85.84
Softmax	64.59	77.01	79.72	52.70	60.42	69.31	57.24	77.48	84.43
ADB	67.49	74.82	77.76	50.04	66.01	70.75	67.41	81.08	84.62
DCL	67.99	72.55	70.76	54.02	61.27	65.98	75.99	82.09	84.52
Ours	**71.93**	**78.06**	**82.19**	**59.76**	**67.73**	**74.09**	**78.91**	**83.82**	**85.93**

Table 4: Macro f1-score excluding the OOD class for few-shot OOD intent detection with different proportion (0.25, 0.5 and 0.75) of in-distribution classes. p indicates the ratio of selected few-shot in-distribution examples. For each setting, the best result is marked in bold.

5.2 Experimental Setup

To achieve a fair comparison, all the baselines and our method use the same pre-trained BERT model (bert-base-uncased (Wolf et al., 2019)) to encode input sentences.

To construct few-shot OOD intent detection tasks from the three datasets, we randomly sample 5% and 10% labeled examples per class as the training set from each of the three datasets. Then, we randomly select 25%, 50%. 75% of the classes in each dataset as in-distribution (ID) classes and set aside the respective remaining classes to the OOD class for the test stage. Concrete numbers of ID examples per class for each dataset can be found in Table 2. In particular, during training and validation, only the labeled few-shot examples of ID classes are seen by the model.

At training stage I, we use a two-layer MLP as q_θ and optimize the parameters of q_θ by Adam (Kingma and Ba, 2015) with a learning rate of $1e^{-4}$. The dropout rate for the corruption function is set to be 0.3 for all experiments. At training stage II, we instantiate our (k+1)-way OOD intent classifier f_ϕ by removing the pre-trained heads of BERT and appending a single layer MLP. For optimizing f_ϕ, we adopt AdamW (Wolf et al., 2019) as optimizer and set the learning rate as $2e^{-5}$ following common practice (Devlin et al., 2019).

For the synthetic ID examples, we sample 15 reconstructed examples per real ID example. For the simulated OOD samples, we construct 100 OOD examples per batch during training. These values are selected with respect to the performance on validation sets. The reported results are the mean of 5 runs with different random seeds.

Following previous works (Yan et al., 2020; Zhang et al., 2021b; Zhan et al., 2021) in OOD intent detection, we use macro f1-score as the primary evaluation metric.

5.3 Correctness of the Synthetic In-distribution Examples

In Figure 4, we provide a qualitative evaluation of the generated synthetic in-distribution (ID) examples using t-SNE visualization (Van der Maaten and Hinton, 2008). We use the BERT embeddings of 5% labeled examples of 8 ID classes and all out-of-distribution examples from CLINC150 and plot them on the top of the figure. By generating 10 synthetic ID examples for each real ID example, we have the bottom figure where we can observe that these synthetic ID examples closely situate in

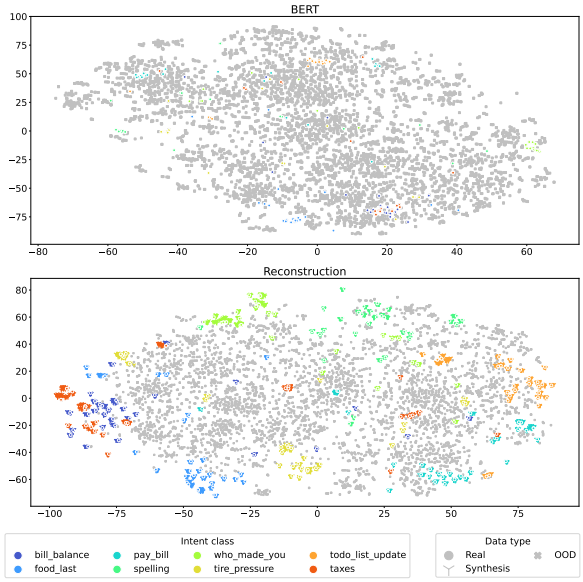

Figure 4: t-SNE visualization of BERT embeddings. Top: BERT embeddings without the synthetic in-distribution examples; Bottom: BERT embeddings with the synthetic in-distribution examples. Better view in color and enlarged.

the vicinity of each real ID example. Since BERT embeddings have been proved to be rich in contextualized semantics (Devlin et al., 2019), the distance between different embeddings can reflect the semantic gap between them. In this regard, at a high level, our generated ID examples can capture the expressiveness of ID classes.

5.4 Main Results

We present the results for the aforementioned three datasets in Table 3 and Table 4. As shown in the two tables, our proposed method consistently outperforms all baselines by a large margin in all settings.

Table 3 presents the results in overall macro f1-score on $(K+1)$ classes including the OOD class. The results in this table can be interpreted as the overall performance of the model. We first inspect the challenging case, where only 5% labeled examples per class are sampled for training as shown in the top of Table 3. We can observe that our method leads to large improvements on all three datasets. In the most challenging case (only 25% of classes in each dataset are selected as in-distribution classes), the improvement is more than $\mathbf{9}\%$ on CLINC150 and $\mathbf{8}\%$ on Banking than the second best results. Moreover, in the 50% and 75% cases, the improvements are also significant. For example, in the 50% case of Banking, the gap between our method

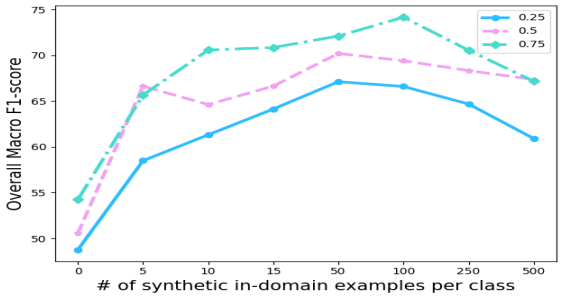

Figure 5: Effect of the number of synthetic in-distribution examples.

and the second best one is around 6.6%. These results verify the effectiveness and consistency of our model in extreme data-scarcity scenarios. As the ratio of labeled examples per class increased to $p = 10\%$, it can be seen that the baselines are improved by a large margin compared with the case of $p = 5\%$. However, our method can still achieve consistent improvement. This validates the robustness of our method under various data-scarcity scenarios.

In Table 4, we summarize the results in macro f1-score of in-distribution classes to demonstrate the effectiveness of synthetic ID examples in our method. It can be seen that in all settings, the performance gains are consistent with the results in Table 3, which indicates that the synthetic ID examples sampled from the DAE can help to improve the classification performance on ID classes.

CLINC150, p=5%			
	Method	ID-F1	Overall-F1
25%	SEG	36.88	36.09
	SEG + Ours	**63.65**	**64.25**
50%	SEG	52.50	51.90
	SEG + Ours	**71.97**	**72.13**
75%	SEG	63.18	62.64
	SEG + Ours	**70.67**	**70.72**

Table 5: Results of SEG (Yan et al., 2020) and SEG with our synthetic ID examples (SEG + Ours). ID-F1 stands for in-distribution f1-score, and overall-F1 indicates the macro f1-score for all classes including the OOD class. Better results are marked in bold.

5.5 Effectiveness of the Synthetic In-distribution Examples

First, we study the impact of the number of synthetic in-distribution (ID) examples. We conduct experiments on the 5% labeled ratio case. As shown in Figure 5, we vary the number of syn-

thetic ID examples per class from 0 to 500. In the range of [0,100], the classification performance increases gradually for all cases (0.25, 0.5 and 0.75). It shows the expressiveness of the synthetic ID examples. However, in the range of [100,500], we observe a slow performance drop in all cases. This is probably because the ID generator is learned from few-shot data and may generate inaccurate ID examples.

To further verify the effectiveness of our synthetic generator, we incorporate the synthetic ID examples to a strong baseline SEG (Yan et al., 2020) and present the results under the $p = 5\%$ setting of CLINC150 in Table 5. With our enlarged ID training set, the performance of SEG can also be improved significantly.

5.6 Robustness of the $(K+1)$-way Training Paradigm

In this subsection, we conduct experiments to evaluate the robustness of the $(K+1)$-way training paradigm with synthetic in-distribution (ID) examples.

As shown in Figure 6, we vary the corruption rate (from 0% to 100%) of the learned latent denoising autoencoder (DAE) (trained by 30% corruption rate). Notice that 100% corruption rate indicates that no useful reconstruction information is passed to the DAE. We can observe that in the 0.5 (orange line) and 0.75 (green line) cases, the learned $(K+1)$-way classifier can maintain a surprisingly consistent performance compared with the 0.25 (purple line). Especially, with 90% corruption rate, the synthetic in-distribution (ID) examples are much less accurate than those with 30% or 40% corruption rate, but the performance does not drop to an unacceptable level. This verifies the noise neutralization effect of the $(K+1)$-way training manner discussed in Section 4.

6 Conclusion

In this paper, we have investigated few-shot OOD intent detection under a more challenging setting. We have conducted a pilot study to identify the key challenge for this problem, which is in improving the in-distribution (ID) expressiveness during training. To this end, we have proposed a promising approach to enrich the ID training set by sampling from a denoising autoencoder trained with only a few examples. The enlarged training set enables to train a well-performing $(K+1)$-way classifier. Our proposed approach has been validated by extensive experiments on real-world bechmarks.

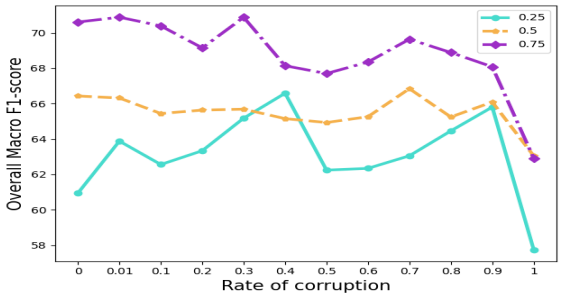

Figure 6: Effect of the rate of corruption on the learned denoising autoencoder. The experiment is conducted on CLINC150 under the $p = 5\%$ setting.

Acknowledgments

We would like to thank the anonymous reviewers for their helpful comments. This research was supported by the grants HK ITF UIM/377 and ITS/359/21FP.

References

Iñigo Casanueva, Tadas Temcinas, Daniela Gerz, Matthew Henderson, and Ivan Vulic. 2020. Efficient intent detection with dual sentence encoders. *CoRR*, abs/2003.04807.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT 2019, Minneapolis, MN, USA, June 2-7, 2019, Volume 1 (Long and Short Papers)*, pages 4171–4186. Association for Computational Linguistics.

Kawin Ethayarajh. 2019. How contextual are contextualized word representations? comparing the geometry of bert, elmo, and gpt-2 embeddings. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 55–65.

Dan Hendrycks and Kevin Gimpel. 2017. A baseline for detecting misclassified and out-of-distribution examples in neural networks. In *5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Conference Track Proceedings*. OpenReview.net.

Taewon Jeong and Heeyoung Kim. 2020. Ood-maml: Meta-learning for few-shot out-of-distribution detection and classification. *Advances in Neural Information Processing Systems*, 33.

Diederik P. Kingma and Jimmy Ba. 2015. Adam: A method for stochastic optimization. In *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*.

Stefan Larson, Anish Mahendran, Joseph J. Peper, Christopher Clarke, Andrew Lee, Parker Hill, Jonathan K. Kummerfeld, Kevin Leach, Michael A. Laurenzano, Lingjia Tang, and Jason Mars. 2019. An evaluation dataset for intent classification and out-of-scope prediction. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 1311–1316, Hong Kong, China. Association for Computational Linguistics.

Ting-En Lin and Hua Xu. 2019. Deep unknown intent detection with margin loss. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 5491–5496, Florence, Italy. Association for Computational Linguistics.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Roberta: A robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692*.

Seonghan Ryu, Seokhwan Kim, Junhwi Choi, Hwanjo Yu, and Gary Geunbae Lee. 2017. Neural sentence embedding using only in-domain sentences for out-of-domain sentence detection in dialog systems. *Pattern Recogn. Lett.*, 88(C):26–32.

Seonghan Ryu, Sangjun Koo, Hwanjo Yu, and Gary Geunbae Lee. 2018. Out-of-domain detection based on generative adversarial network. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 714–718, Brussels, Belgium. Association for Computational Linguistics.

Lei Shu, Hu Xu, and Bing Liu. 2017. DOC: deep open classification of text documents. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing, EMNLP 2017, Copenhagen, Denmark, September 9-11, 2017*, pages 2911–2916. Association for Computational Linguistics.

Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. 2014. Dropout: a simple way to prevent neural networks from overfitting. *The journal of machine learning research*, 15(1):1929–1958.

Laurens Van der Maaten and Geoffrey Hinton. 2008. Visualizing data using t-sne. *Journal of machine learning research*, 9(11).

Pascal Vincent, Hugo Larochelle, Yoshua Bengio, and Pierre-Antoine Manzagol. 2008. Extracting and composing robust features with denoising autoencoders. In *Proceedings of the 25th international conference on Machine learning*, pages 1096–1103.

Kuan-Chieh Wang, Paul Vicol, Eleni Triantafillou, and Richard Zemel. Few-shot out-of-distribution detection.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Rémi Louf, Morgan Funtowicz, et al. 2019. Huggingface's transformers: State-of-the-art natural language processing. *arXiv preprint arXiv:1910.03771*.

Jiaming Xu, Peng Wang, Guanhua Tian, Bo Xu, Jun Zhao, Fangyuan Wang, and Hongwei Hao. 2015. Short text clustering via convolutional neural networks. In *Proceedings of the 1st Workshop on Vector Space Modeling for Natural Language Processing, VS@NAACL-HLT 2015, June 5, 2015, Denver, Colorado, USA*, pages 62–69. The Association for Computational Linguistics.

Guangfeng Yan, Lu Fan, Qimai Li, Han Liu, Xiaotong Zhang, Xiao-Ming Wu, and Albert Y.S. Lam. 2020. Unknown intent detection using Gaussian mixture model with an application to zero-shot intent classification. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 1050–1060, Online. Association for Computational Linguistics.

Li-Ming Zhan, Haowen Liang, Bo Liu, Lu Fan, Xiao-Ming Wu, and Albert Y.S. Lam. 2021. Out-of-scope intent detection with self-supervision and discriminative training. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 3521–3532, Online. Association for Computational Linguistics.

Hanlei Zhang, Hua Xu, and Ting-En Lin. 2021a. Deep open intent classification with adaptive decision boundary. *Proceedings of the AAAI Conference on Artificial Intelligence*, 35(16):14374–14382.

Hanlei Zhang, Hua Xu, and Ting-En Lin. 2021b. Deep open intent classification with adaptive decision boundary. *Proceedings of the AAAI Conference on Artificial Intelligence*, 35(16):14374–14382.

Jianguo Zhang, Kazuma Hashimoto, Wenhao Liu, Chien-Sheng Wu, Yao Wan, Philip S. Yu, Richard Socher, and Caiming Xiong. 2020. Discriminative nearest neighbor few-shot intent detection by transferring natural language inference. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing, EMNLP 2020, Online, November 16-20, 2020*, pages 5064–5082. Association for Computational Linguistics.

CGIM: A Cycle Guided Interactive Learning Model for Consistency Identification in Task-oriented Dialogue

Libo Qin[1], **Qiguang Chen**[1], **Tianbao Xie**[1], **Qian Liu**[2],
Shijue Huang[1], **Wanxiang Che**[1]*, **Zhou Yu**[3]
[1]Research Center for Social Computing and Information Retrieval
Harbin Institute of Technology, China
[2] Beihang University, Beijing, China
[3]Columbia University
{lbqin,tianbaoxie,car}@ir.hit.edu.cn; qian.liu@buaa.edu.cn; zy2461@columbia.edu

Abstract

Consistency identification in task-oriented dialog (CI-ToD) usually consists of three subtasks, aiming to identify inconsistency between current system response and current user response, dialog history and the corresponding knowledge base. This work aims to solve CI-ToD task by introducing an explicit interaction paradigm, **C**ycle **G**uided **I**nteractive learning **M**odel (CGIM), which achieves to make information exchange explicitly from all the three tasks. Specifically, CGIM relies on two core insights, referred to as *guided multi-head attention module* and *cycle interactive mechanism*, that collaborate from each other. On the one hand, each two tasks are linked with the *guided multi-head attention module*, aiming to explicitly model the interaction across two related tasks. On the other hand, we further introduce *cycle interactive mechanism* that focuses on facilitating model to exchange information among the three correlated sub-tasks via a cycle interaction manner. Experimental results on CI-ToD benchmark show that our model achieves the state-of-the-art performance, pushing the overall score to 56.3% (5.0% point absolute improvement). In addition, we find that CGIM is robust to the initial task flow order.

1 Introduction

Consistency identification task in dialogue has the potential benefits of preventing inconsistent response generation (Welleck et al., 2019), which has attracted increasing attention. Recent years have witnessed two promising research directions in *consistency identification*. The first focuses on *consistency identification* in open-domain dialogue (Zhang et al., 2018; Zheng et al., 2019). The second direction consider *consistency identification* in task-oriented dialogue system (CI-ToD) (Qin et al., 2021b). In this work, we focus on the latter. Recently, Qin et al. (2021b) introduces a benchmark (CI-ToD) for consistency identification in task-oriented dialogue to facilitate the relevant research. CI-ToD introduces three subtasks including: (1) *dialogue **h**istory **c**onsistency **i**dentification **t**ask* (HCIT) to judge whether generated response is inconsistent with dialogue history; (2) *user **q**uery **c**onsistency **i**dentification **t**ask* (QCIT) to detect the consistency status between query and system response, and (3) ***k**nowledge **b**ase **c**onsistency **i**dentification **t**ask* (KBCIT) to determine system response is contradicted with the corresponding knowledge base.

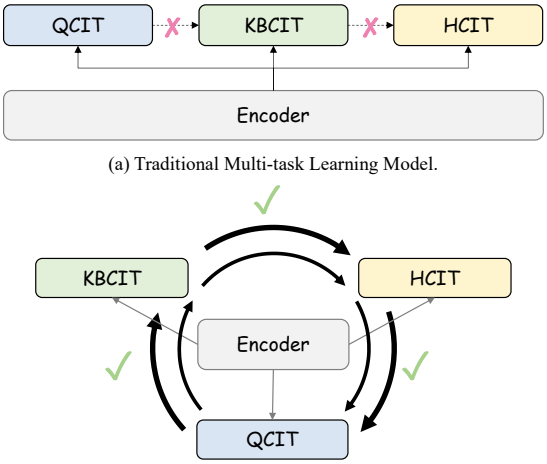

(a) Traditional Multi-task Learning Model.

(b) Cycle Guided Interative Learning Model (CGIM).

Figure 1: (a) Traditional multi-task learning models learn mutual information across tasks via an implicit interaction manner vs. (b) Our proposed cycle interactive learning model explicitly consider cross-impact across three tasks via an explicit interaction manner.

Intuitively, the three tasks are closely related, indicating information of one task can be utilized in other related tasks. For example, if we first complete HCIT, the result of HCIT can assist the QCIT to determine whether the system response is contradicted with the user query, since the dialogue history and the user query tend to share similar topic (Chen et al., 2020). Similarly, KBCIT can also provide additional information for helping

*Email corresponding.

Proceedings of the 29th International Conference on Computational Linguistics, pages 461–470
October 12–17, 2022.

HCIT, because dialogue history can be regarded as an unstructured knowledge description for the corresponding KB. Above observations suggest that it is imperative to take cross-impact across three tasks into account. To this end, Qin et al. (2021b) explore a simple multi-task framework that consists of a shared encoder and different task decoders to jointly consider correlation, which is shown in Figure 1(a). Though achieving superior performance compared with single models, their approaches solely rely on shared latent representations to model the interaction in an implicit interaction manner, which limits their performance. Therefore, it is promising to consider an explicit joint modeling approach for CI-ToD.

While the idea seems promising, achieving this objective is challenging, since we need to jointly model the three sub-tasks simultaneously rather than simple two tasks setting. Recent work have shown explicit joint modeling is superior to the implicit joint modeling (Goo et al., 2018; Qin et al., 2021a). Nevertheless, their work still limits to modeling the relationship between two tasks, it remains clear if the explicit modeling paradigm can be applied in three tasks. To this end, as shown in Figure 1(b), we propose a novel **C**ycle **G**uided **I**nteractive learning **M**odel (CGIM) for CI-ToD, which achieves to perform the three sub-tasks jointly and interactively in an explicit interaction paradigm. Specifically, CGIM first consists of a ***g**uided **m**ulti-head **a**ttention module* (GMA) that is used for two related tasks, which aims to explicitly utilize information from another task. With the help of GMA, each task can not only rely on its own task information but also performed with the guidance of the corresponding correlated task explicitly. Furthermore, since GMA can only enable the single information flow from one task to another task, we further propose a novel *cycle interactive mechanism* to facilitate information flow across the three tasks in an cycle interaction fashion. With the use of cycle interactive mechanism, CGIM can be stacked to form a hierarchy, which can gradually capture interaction information and better transfer knowledge.

We conduct experiments on CI-ToD benchmark and results show that CGIM achieves the best performance, outperforming previous state-of-the-art methods by at least 5.0% (overall accuracy). Besides, extensive analysis further demonstrate the superior and robustness of our approach.

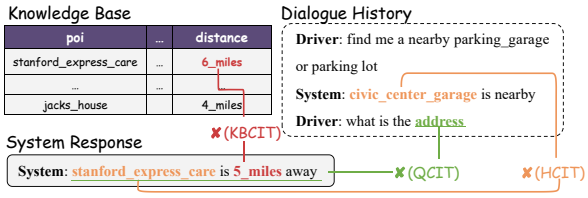

Figure 2: Example illustration in CI-ToD. Different colors denote inconsistency type of different tasks.

Main contributions are summarized as follows:

- To the best of our knowledge, we make the first attempt to explore an explicit interaction model for CI-ToD.
- We introduce a novel cycle interactive learning model for CI-ToD, which achieves establishing a triple-interaction across the three tasks simultaneously.
- Results on CI-ToD benchmark show that CGIM achieves state-of-the-art performance. Besides, we observe that CGIM is robust to the initial task flow order.

All codes in this work will be publicly available at https://github.com/LightChen233/CGIM.

2 Background

To make the paper self-complete, we present the definition of the task that follows Qin et al. (2021b) in this section.

2.1 Task Definition

Given a task-oriented dialogue between a user (u) and a system (s), the dialogue history is defined as $H = \{(u_1, s_1),(u_2, s_2), \ldots,(u_{n-1}, s_{n-1})\}$, the corresponding knowledge base KB is B, the user query is denoted by u_n and the system response is denoted by s_n.

Formally, the consistency identification in task-oriented dialogue contains three tasks: the dialogue **h**istory **c**onsistency **i**dentification **t**ask (HCIT), the user **q**uery **c**onsistency **i**dentification **t**ask (QCIT), and the **k**nowledge **b**ase **c**onsistency **i**dentification **t**ask (KBCIT) to judge whether system response is contradicted with the corresponding dialogue history, user query, and KB, respectively, which are defined as:

$$(y^{\mathrm{Q}}, y^{\mathrm{H}}, y^{\mathrm{B}}) = f_\theta([H, B, u_n], s_n), \quad (1)$$

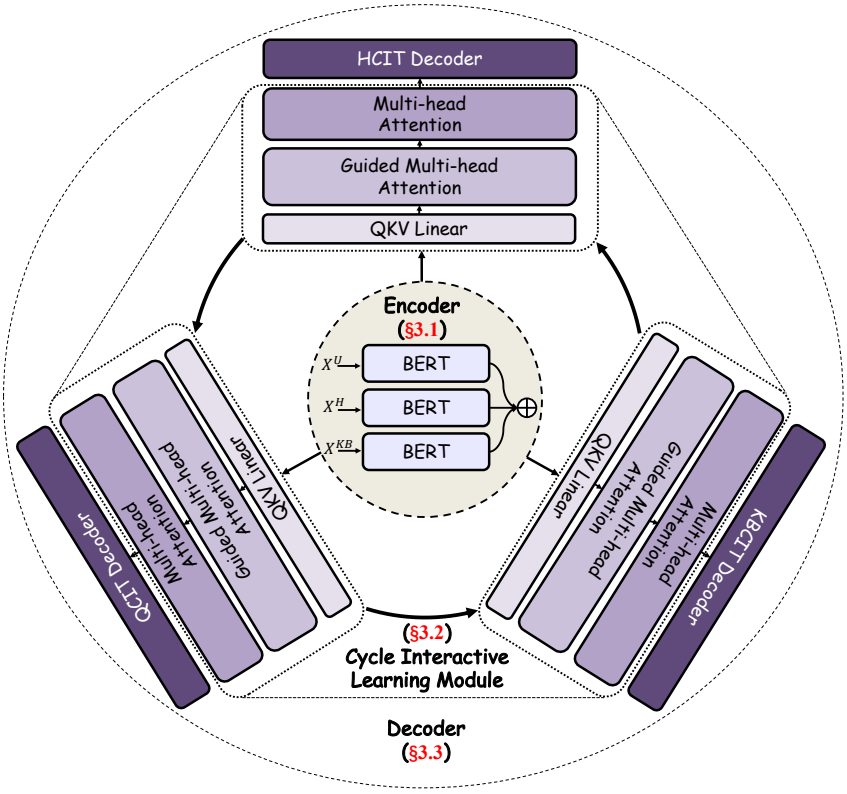

Figure 3: The illustration of the proposed cycle interactive learning model (CGIM), which consists of three components: encoder, cycle interactive learning module and decoder.

where f_θ denotes the parameters of model; $y^{\mathrm{Q}}, y^{\mathrm{H}}, y^{\mathrm{B}}$ represents the probabilities of inconsistent system response in QCIT, HCIT and KBCIT, respectively.

2.2 Example Illustration

To understand HCIT, QCIT, and KBCIT intutively, we provide some example cases, which are shown in the following:

QCIT QCIT aims at detecting the inconsistency between dialogue system response and current user query. As shown in Figure 2, user is intended to ask for the *address*. However, the system response provide the answer about *distence* to stanford_express_care, which results in inconsistency with user query.

HCIT HCIT aims at detecting the inconsistency between system response and dialogue history except the current query. Figure 2 shows the inconsistent dialogue, where the previous dialogue history talked about *civic_center_garage* and the user did not change the topic. However, the system responded by talking about *stanford_express_care*, which is contradicted with the dialogue history.

KBCIT KBCIT aims at detecting the inconsistency between dialogue system response and corresponding KB. As shown in Figure 2, *stanford_express_care* is located *6_miles* according to the corresponding KB. However, the system indicates that the *distance* to *stanford_express_care* is *5_miles*, which is inconsistent with the information provided in the KB.

3 Approach

The architecture of the cycle guided interactive learning model (CGIM) is depicted in Figure 3. It mainly consists of three components: three task-specific encoders for obtaining encoding representation for each task (§3.1); a cycle interactive learning module for explicitly establishing the interaction across the three tasks (§3.2); three separate decoders for HCIT, QCIT and KBCIT (§3.3), respectively. In the following sections, the details of our framework are given.

3.1 Encoder

Following Qin et al. (2021b), we employ the pre-trained model (i.e., BERT) (Devlin et al., 2019) as the encoder and use delimiter tokens `[SOK]`,

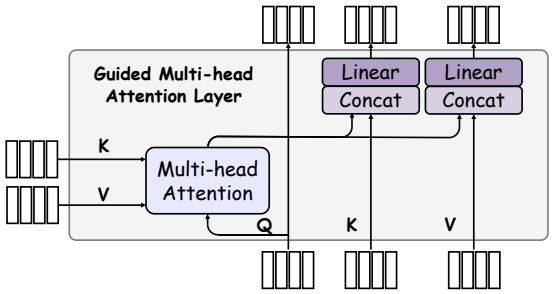

Figure 4: Guided Multi-head Attention Layer.

[EOK], [USR], [SYS] to capture the role feature of KB, user and system response.

Query Representation To consider the system response, we concatenate u_n and the last system response s_n to obtain the query encoding representation. Therefore, the input can be represented as X^U = ([CLS], u_n, [SEP], s_n, [SEP]), where [CLS] and [SEP] are special symbol, and BERT reads it to produce the representation:

$$\mathbf{h}^{\mathrm{Q}} = \mathrm{BERT}(X^{\mathrm{U}}), \tag{2}$$

where the last layer's hidden representation $\mathbf{h}^{\mathrm{Q}}$ of the [CLS] token is considered as the query representation.

Dialogue History Representation Similarly, we concatenate the dialog history H and the system response s_n as X^{H} = ([CLS], $\hat{H}$, [SEP], s_n, [SEP]), which is used for acquiring dialogue history representation $\mathbf{h}^{\mathrm{H}}$:

$$\mathbf{h}^{\mathrm{H}} = \mathrm{BERT}(X^{\mathrm{H}}), \tag{3}$$

where $\hat{H}$ is [USR]u_1[SYS]$s_1 \ldots$[USR]u_n.

Knowledge Base Representation For KB representation, we first linearize the KB and then concatenate the linearized KB and system response to obtain X^{B}= ([CLS], $\hat{B}$, [SEP], s_n, [SEP]). Feeding it into BERT, we obtain the knowledge base representation $\mathbf{h}^{\mathrm{B}}$:

$$\mathbf{h}^{\mathrm{B}} = \mathrm{BERT}(X^{\mathrm{B}}), \tag{4}$$

where $\hat{B}$ is [SOK]B[EOK].

3.2 Cycle Guided Interactive Learning Module

Traditional multi-task learning only depend on a set of shared parameters to implicitly consider correlation across different correlated tasks. In contrast, we present a cycle guided interactive learning model to explicit model the interaction, which consists of two parts: the ***g**uided **m**ulti-head **a**ttention module* (GMA) and the ***c**ycle **i**nteraction **m**echanism* (CIM).

3.2.1 Guided Multi-head Attention Module

GMA mainly consists of a guided multi-head attention layer and self multi-head attention layer, achieving to explicitly model interaction across related tasks.

Guided Multi-head Attention Layer. First, given $\mathbf{h}^{\mathrm{Q}}$, $\mathbf{h}^{\mathrm{H}}$ and $\mathbf{h}^{\mathrm{B}}$, we first directly perform a concatenation operation upon them and adopt different projection linear layers to obtain different updated representations for QCIT, HCIT and KBCIT, which are denoted as:

$$\mathbf{H} = \mathrm{Concat}(\mathbf{h}^{\mathrm{Q}}, \mathbf{h}^{\mathrm{H}}, \mathbf{h}^{\mathrm{B}}), \tag{5}$$

$$\mathbf{H}^{\mathrm{Q}}, \mathbf{H}^{\mathrm{H}}, \mathbf{H}^{\mathrm{B}} = \mathbf{W}^{\mathrm{Q}}\mathbf{H}, \mathbf{W}^{\mathrm{H}}\mathbf{H}, \mathbf{W}^{\mathrm{B}}\mathbf{H}, \tag{6}$$

where $\mathbf{H} \in \mathbb{R}^{3\times d}$ (d represents the encoding dimension); Concat is concatenation operation; $\mathbf{W}^{\mathrm{H}}, \mathbf{W}^{\mathrm{Q}}, \mathbf{W}^{\mathrm{B}}$ are the trainable matrix.

Then, to obtain updated QCIT representations with the guidance of HCIT explicitly, it is necessary to align query with its closely related dialogue history information. To be more specific, as shown in Figure 4, we first employ QKV Linear to map the dialog history and query representations $\mathbf{H}^{\mathrm{H}}$ and $\mathbf{H}^{\mathrm{Q}}$ to query ($\mathbf{Q}^{\mathrm{H}}$, $\mathbf{Q}^{\mathrm{Q}}$), keys ($\mathbf{K}^{\mathrm{H}}$, $\mathbf{K}^{\mathrm{Q}}$) and values ($\mathbf{V}^{\mathrm{H}}$, $\mathbf{V}^{\mathrm{Q}}$) matrices. We then treat $\mathbf{Q}^{\mathrm{Q}}$ as queries, $\mathbf{K}^{\mathrm{H}}$ as keys and $\mathbf{V}^{\mathrm{H}}$ as values to obtain the updated representation with explicitly considering the information from the HCIT task. The output is a weighted sum of values:

$$\hat{\mathbf{H}} = \mathrm{MultiHead}(\mathbf{Q}^{\mathrm{Q}}, \mathbf{K}^{\mathrm{H}}, \mathbf{V}^{\mathrm{H}}), \tag{7}$$

$$\mathrm{Attention}(\mathbf{Q}, \mathbf{K}, \mathbf{V}) = \mathrm{softmax}\left(\frac{\mathbf{QK}}{\sqrt{d_k}}\right)\mathbf{V}, \tag{8}$$

$$\mathrm{MultiHead}(\mathbf{Q}, \mathbf{K}, \mathbf{V}) = \mathrm{Concat}(\mathbf{H}_i, \ldots, \mathbf{H}_h)\mathbf{W}^0, \tag{9}$$

$$\text{where } \mathbf{H}_i = \mathrm{Attention}(\mathbf{QW}_i^Q, \mathbf{KW}_i^K, \mathbf{VW}_i^V), \tag{10}$$

where $\hat{\mathbf{H}}$ can be seen as the updated query information with the guidance of dialogue history information; $\mathbf{Q}, \mathbf{K}$ and $\mathbf{V}$ denote the query, key, and value respectively. d_k is the dimension of the key; $\mathbf{W}_i^Q, \mathbf{W}_i^K, \mathbf{W}_i^V$ denote the head projection matrix, respectively; h is the head number.

Then, we concatenate the updated representation $\mathbf{H}$ with the original $\mathbf{K}^{\mathrm{Q}}$ and $\mathbf{V}^{\mathrm{Q}}$ to construct new

keys $\hat{\mathbf{K}}^{\mathbf{Q}}$ and values $\hat{\mathbf{V}}^{\mathbf{Q}}$, calculating as:

$$\hat{\mathbf{K}}^{\mathbf{Q}} = \text{Linear}(\text{Concat}(\mathbf{K}^{\mathbf{Q}}, \hat{\mathbf{H}})), \quad (11)$$

$$\hat{\mathbf{V}}^{\mathbf{Q}} = \text{Linear}(\text{Concat}(\mathbf{V}^{\mathbf{Q}}, \hat{\mathbf{H}})). \quad (12)$$

Self Multi-head Attention Layer. Given the obtained updated keys, updated values, and queries, we further introduce a self multi-head attention layer to enhance the interaction information and obtain the final query guided query representation:

$$\widetilde{\mathbf{H}}^{\mathrm{Q}} = \text{MultiHead}(\mathbf{Q}^{\mathrm{Q}}, \hat{\mathbf{K}}^{\mathrm{Q}}, \hat{\mathbf{V}}^{\mathrm{Q}}), \quad (13)$$

where $\widetilde{\mathbf{H}}^{\mathrm{Q}}$ denotes the updated representations.

Similiar to transformer, we also employ a residual connection (He et al., 2016), layer normalization (Ba et al., 2016) and a fully connected feed-forward network.

3.2.2 Cycle Interactive Mechanism

With the help of the GMA, single information flow can be established. Formally, given features representation $\mathbf{H}^{\mathrm{Q}}$ and $\mathbf{H}^{\mathrm{H}}$, GMA aims to output the attended features $\widetilde{\mathbf{H}}^{\mathrm{Q}}$ for $\mathbf{H}^{\mathrm{Q}}$ guided by $\mathbf{H}^{\mathrm{H}}$, which can be formulated as:

$$\widetilde{\mathbf{H}}_{\mathbf{1}}^{\mathrm{Q}} = \text{GMA}_{\mathrm{H}\to\mathrm{Q}}(\mathbf{H}_0^{\mathrm{H}}, \mathbf{H}_0^{\mathrm{Q}}), \quad (14)$$

where $\mathbf{H}_0^{\mathrm{H}}$ and $\mathbf{H}_0^{\mathrm{Q}}$ are initialized as $\mathbf{H}^{\mathrm{H}}$ and $\mathbf{H}^{\mathrm{H}}$, respectively.

Similarly, the query guided dialogue history representation and dialogue history guided knowledge base representation can be obtained in the same manner, which are shown as:

$$\widetilde{\mathbf{H}}_{\mathbf{1}}^{\mathrm{B}} = \text{GMA}_{\mathrm{Q}\to\mathrm{B}}(\mathbf{H}_0^{\mathrm{Q}}, \mathbf{H}_0^{\mathrm{B}}), \quad (15)$$

$$\widetilde{\mathbf{H}}_{\mathbf{1}}^{\mathrm{H}} = \text{GMA}_{\mathrm{B}\to\mathrm{H}}(\mathbf{H}_0^{\mathrm{B}}, \mathbf{H}_0^{\mathrm{H}}). \quad (16)$$

To enable the shared knowledge flowed across the three subtasks, we further introduce a cycle interaction mechanism (CIM) with multiple layers to gradually and iteratively control knowledge transfer, which can be formulated as:

$$\widetilde{\mathbf{H}}^{\mathrm{Q}}, \widetilde{\mathbf{H}}^{\mathrm{H}}, \widetilde{\mathbf{H}}^{\mathrm{B}} = \text{CIM}_{(\mathrm{H}\to\mathrm{Q}\to\mathrm{B}\to\mathrm{Q})}(\mathbf{H}^{\mathrm{Q}}, \mathbf{H}^{\mathrm{H}}, \mathbf{H}^{\mathrm{B}}), \quad (17)$$

After stacking L layer, we obtain a final updated feature representation: $\widetilde{\mathbf{H}}_{\mathrm{L}}^{\mathrm{B}}$, $\widetilde{\mathbf{H}}_{\mathrm{L}}^{\mathrm{H}}$ and $\widetilde{\mathbf{H}}_{\mathrm{L}}^{\mathrm{Q}}$.

3.3 Decoder

Given the final updated representations $\widetilde{\mathbf{H}}_{\mathrm{L}}^{\mathrm{B}}$, $\widetilde{\mathbf{H}}_{\mathrm{L}}^{\mathrm{H}}$ and $\widetilde{\mathbf{H}}_{\mathrm{L}}^{\mathrm{Q}}$ for each task, we directly flatten them into a single vector $\mathbf{d}^{B}$, $\mathbf{d}^{\mathrm{H}}$ and $\mathbf{d}^{\mathrm{Q}}$, which are fed into separate decoders to perform QCIT, HCIT and KBCIT, which can be denoted as:

$$\mathbf{y}^{\mathrm{H}} = \text{softmax}(\mathbf{W}^{\mathrm{H}}\mathbf{d}^{\mathrm{H}} + \mathbf{b}_{\mathrm{H}}), \quad (18)$$

$$\mathbf{y}^{\mathrm{Q}} = \text{softmax}(\mathbf{W}^{Q}\mathbf{d}^{Q} + \mathbf{b}_{Q}), \quad (19)$$

$$\mathbf{y}^{\mathrm{B}} = \text{softmax}(\mathbf{W}^{B}\mathbf{d}^{B} + \mathbf{b}_{B}), \quad (20)$$

where $\mathbf{y}^{\mathrm{H}}$, $\mathbf{y}^{\mathrm{Q}}$ and $\mathbf{y}^{\mathrm{B}}$ are the predicted distribution result for three tasks, respectively; $\mathbf{W}^{\mathrm{H}}$, $\mathbf{W}^{\mathrm{Q}}$ and $\mathbf{W}^{\mathrm{B}}$ are learnable transformation matrices; $\mathbf{b}_{\mathrm{Q}}$, $\mathbf{b}_{\mathrm{H}}$ and $\mathbf{b}_{\mathrm{B}}$ are learnable bias vectors.

3.4 Joint Training

The training objective of each task is the binary cross-entropy loss. Specifically, the objective for QCIT is:

$$\mathcal{L}_Q = -\sum^{T} \left(\hat{\mathbf{y}}^{\mathrm{Q}} \log\left(\mathbf{y}^{\mathrm{Q}}\right)\right), \quad (21)$$

where $\hat{\mathbf{y}}$ is gold label and T is the training data size.

Similar, $\mathcal{L}_H$ and $\mathcal{L}_B$ can be obtained in a similar manner. Following Bai et al. (2021) and Bao et al. (2021), the final joint loss function is as:

$$\mathcal{L}_\theta = \alpha_Q\mathcal{L}_Q + \alpha_H\mathcal{L}_H + \alpha_B\mathcal{L}_B, \quad (22)$$

where α_Q, α_H and α_B are hyper-parameter[1]

4 Experiments

4.1 Experimental Settings

To evaluate the effectiveness of CGIM, we conduct experiments on the CI-ToD benchmark (Qin et al., 2021b). Specifically, CI-ToD consists of 2,553 dialogues for training, 319 dialogues for validation, and 318 dialogues for testing.

In our experimental setting, we adopt `BERT-base` and the dimension of all hidden units is 768. The batch size we use is selected from $\{4, 8, 16\}$ and learning rate is selected from $\{1e^{-5}, 2e^{-5}, 5e^{-5}\}$. We use AdamW (Loshchilov and Hutter, 2019) to optimize the parameters in our model. We select all hyper-parameters from the validation set. All experiments are conducted at Tesla P100 and Tesla V100.

4.2 Baselines

Following Qin et al. (2021b), we compare our model with the following state-of-the-art multi-task

[1]In our experiment, we set them as 1.

Model	QI F1	HI F1	KBI F1	Overall Acc
BART-separate (Lewis et al., 2020)	0.695	0.496	0.721	0.450
BERT-multi-task (Devlin et al., 2019)	0.691	0.555	0.740	0.500
RoBERTa-multi-task (Liu et al., 2019)	0.715	0.472	0.715	0.500
XLNet-multi-task (Yang et al., 2020)	0.725	0.487	0.736	0.509
Longformer-multi-task (Beltagy et al., 2020)	0.717	0.500	0.710	0.497
BART-multi-task (Lewis et al., 2020)	0.744	0.510	0.761	0.513
CGIM	**0.764**	**0.567**	**0.772**	**0.563**

Table 1: Main results. The bolded number indicates the best performance. All baselines results are taken from Qin et al. (2021b).

learning models based on the strong pre-trained models:

(1) `BERT` (Devlin et al., 2019): the model pre-trains bidirectional representations from a large-scale text corpus; (2) `RoBERTa` (Liu et al., 2019): the model improves the training procedure of BERT to make it perform better; (3) `XLNet` (Yang et al., 2020): the model combines the advantage of autoregressive and autoencoding approaches by performing a permutation language objective; (4) `Longformer` (Beltagy et al., 2020): the model employs an attention pattern that combines local and global information while also scaling linearly with the sequence length, making it easy to process long documents; (5) `BART` (Lewis et al., 2020): the model uses a pre-training approach to map corrupted documents to the original document, which works well on both various generation tasks and understanding tasks.

We refer to the model with multi-task learning for three tasks as `model-multi-task`. In addition, we also compare CGIM with the state-of-the-art separate model for each task, which is referred as `model-separate`.

4.3 Main Results

Following Qin et al. (2021b), we adopt query inconsistency (QI) F1 scores, dialogue history inconsistency (HI) F1 scores, knowledge base inconsistency (KBI) F1 scores to evaluate QCIT, HCIT, and KBCIT respectively. Besides, we also use overall accuracy, a strict metric that requires all tasks are predicted correctly.

From the results shown in Table 1. We have the following observations:

(1) `CGIM` yields better performance compared with `BART-separate` on all metrics, which verifies that QCIT, HCIT and KBCIT tasks are correlated where joint model can be benefited from capturing shared knowledge across tasks, supporting our motivation;

(2) `CGIM` achieves the best performance on three tasks compared with all baselines. Compared with `BART-multi-task`, our framework obtains 2.0%, 5.7% and 1.1% improvements on three tasks, respectively. This indicates that the proposed explicit interaction paradigm is better than the implicit interaction paradigm that is insufficient to grasp knowledge transfer, which is consistent to the observation on other explicit joint modeling work on two tasks (Goo et al., 2018; Qin et al., 2021a);

(3) `CGIM` attains the best results on Overall Acc. and beats the best model `BART-multi-task` by a large margin of 5.0%. This suggests that all three tasks are highly correlated and explicit modeling mechanism can help to improve the whole dialogue understanding ability than the implicit modeling. It is worth noticing that the backbone of `CGIM` is BERT and it still outperforms `BART-multi-task` by a large margin, which further verifies the effectiveness of explicit modeling paradigm.

4.4 Analysis

This section answer the following research questions to understand CGIM in depth:

(1) Does each guided multi-head attention (GMA) module improve performance?

(2) Does a deeper layer of guided multi-head attention module bring a better performance?

(3) Is CGIM robust to the initial task flow order?

(4) Does explicit interaction modeling gain the performance improvement rather than the involved parameters?

Model	QI F1	HI F1	KBI F1	Overall Acc
CGIM	**0.764**	**0.567**	**0.772**	**0.563**
w/o QCIT → KBCIT	0.712	0.539	0.749	0.512
w/o KBCIT → HCIT	0.731	0.506	0.752	0.494
w/o HCIT → QCIT	0.710	0.507	0.764	0.521
w/MLP	0.725	0.515	0.686	0.507

Table 2: Ablation Study. The bolded number indicates the best performance in the first block.

Model	QI F1	HI F1	KBI F1	Overall Acc
BART	0.744	0.510	0.761	0.513
CGIM (QCIT → KBCIT → HCIT)	0.764	0.567	0.772	0.562
CGIM (QCIT → HCIT → KBCIT)	0.787	0.599	0.778	0.560

Table 3: Robust Test. The performance of BART and different information flow model.

(5) Is the explicit modeling method still effective in low-resource scenario?

(6) How CGIM is useful in CI-ToD?

4.4.1 Answer 1: GMA boosts performance across the related tasks

We devise three variations for exploring the effect of guided multi-head attention layer. In particular, QCIT → KBCIT is the variation by removing guided multi-head attention layer from QCIT to KBCIT and all the other components keep unchanged. Similarly, KBCIT → HCIT and HCIT → QCIT variation denotes that remove the corresponding guided multi-head attention layer for HCIT and QCIT, respectively.

Results are presented in Table 2 (row 2,3,4), we observe that without guided multi-head attention layer leads to a drop in the corresponding tasks. We attribute it to the fact that all the two sub-tasks are highly correlated, it hinders the information transfer and thus hurts the performance without the corresponding guided multi-head attention layer.

4.4.2 Answer 2: More layers may not be better

To investigate the influence of layers of guided multi-head attention module, we conduct experiments on the different layers of our framework. Figure 5 presents the results. We can observe: (1) The performance of CGIM with two or three layers is better than the model with one layer, which indicates that a deeper layer can achieve better interactions across three tasks. (2) Another interesting observation is that when the number of layers is five, we can observe the performance on overall accuracy drops a lot, even underperforming the model with one layer. We speculate that there may

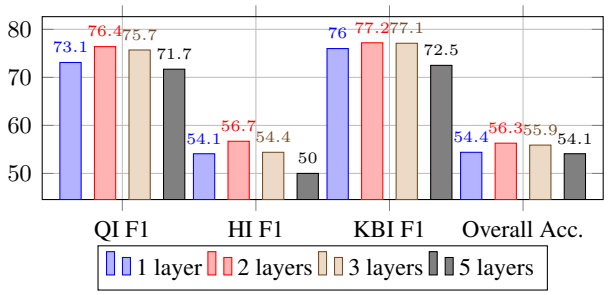

Figure 5: Influence of Layers.

be gradient vanishing or over-fitting problem when the layer of network exceeds five, which is consistent with prior observation (Feng et al., 2017; Qin et al., 2020).

4.4.3 Answer 3: CGIM is robust

Another interesting research question is whether CGIM is robust to the initial task information flow. To answer this question, we conduct experiments with another initial information flow order QCIT → HCIT → KBCIT and the results are presented in Table 3. We witness two observations: (1) the performance CGIM (QCIT → KBCIT → HCIT) is comparable with the original CGIM; (2) it also outperforms BART by a large margin. Above observations verifies the robustness of our method to the initial task flow order.

4.4.4 Answer 4: Explicit interaction modeling boosts performance

We replace three guided multi-head attention layers with three MLP modules. We refer to it as "w/MLP" and the results are reported in Table 2. As seen, CGIM yields better results than the "w/MLP" model with the same amount of parameters in all three tasks (over 4% drops on all tasks), which demonstrates that the improvements come from the proposed explicit interaction mechanism across the three tasks rather than the extra parameters.

4.4.5 Answer 5: CGIM works in few-shot scenario

We further investigate the effectiveness of CGIM in a low-resource setting. We randomly extract different proportions of datasets from the entire dataset to simulate the low-resource setting, i.e., [5%, 10%, 20%, 50%]. The results are shown in the Figure 6 (a). We observe that our framework outperforms the BART on all low-resource settings. We attribute it to the fact the proposed explicit interaction can make the most limited data and

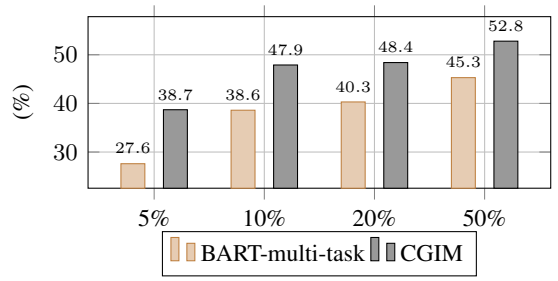

(a) Performance of BART-multi-task and CGIM.

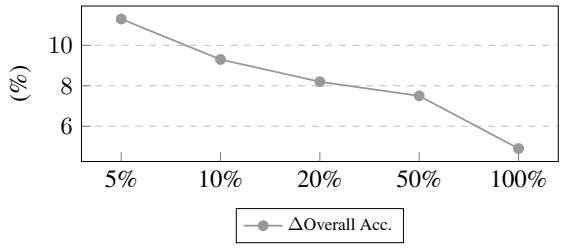

(b) ΔOverall Accuracy on Low-Resource Setting.

Figure 6: Low-Resource Performance.

effectively share the knowledge between the three tasks compared to implicit modeling method.

In addition, we also analyze the performance gap on different datasets. The results are shown in Figure 6 (b), we find that the less data we have, the higher the performance improvement of our model compared to `BART-multi-task`, which indicates that our framework is more practical and scalable in a low resource setting.

4.4.6 Answer 6: Qualitative analysis

This section provides a case study for better understanding of our model. Figure 7 shows one case made by baseline model `BART-multi-task` and `CGIM`. In this case, user query and dialogue history talks about the same topic (*the_clement_hotel*), which demonstrates that the QCIT and HCIT are highly correlated.

However, `BART-multi-task` predict the HCIT correctly but QCIT incorrectly, which demonstrates original implicit interaction paradigm does not effectively model the correlation across the tasks. In contrast, `CGIM` predicts both QCIT and HCIT correctly. We think that the proposed explicit interaction paradigm successfully grasps correlation and thus enhance each task.

5 Related Work

Increasing attention has been witnessed in consistency identification in dialogue. To this end, PersonaChat (Zhang et al., 2018) and PersonalDialog (Zheng et al., 2019) are introduced to implicitly

	QI	HI	KBI
BART-m	0,	1,	1
CGIM	1,	1,	1
GOLD	1,	1,	1

Driver:where is the nearest hotel
System:the nearest hotal is the_clement_hotel
Driver:what is the address (for the_clement_hotel)
System: hotel_keen is at 347_alta_mesa_ave

Figure 7: Prediction made by `BART-m` and `CGIM`. `BART-m` denotes `BART-multi-task`.

consider the consistency in dialogue generation. Welleck et al. (2019) model the consistency of dialogue systems by introducing a new natural language inference dataset called DialogueNLI. Dziri et al. (2019) propose to use state-of-the-art entailment techniques for evaluating the coherence of dialogue systems. Nie et al. (2021) propose a DialoguE COntradiction DEtection task (DECODE) to evaluate the ability to detect contradictory in dialogue. However, their work mainly focuses on consistency in open-domain direction. In contrast, our framework mainly considers improving consistency in task-oriented dialogues.

In recent years, Qin et al. (2021b) make the first step towards consistency identification in task-oriented dialogues and propose three sub-tasks to detect whether the system response is contradicted with the corresponding dialogue history, user query, and knowledge base. In addition, they also introduce a public benchmark CI-ToD and provide some state-of-the-art pre-trained models to facilitate the research. Unfortunately, their models only jointly consider the correlated three tasks in an implicit manner. Compared with their model, we propose a cycle guided interactive learning model (CGIM), which can explicitly model interaction across the three tasks in a cycled interaction manner. To our knowledge, we are the first to explore an explicit interaction paradigm for CI-ToD.

6 Conclusion

We studied how to explicitly model the interaction across three sub-tasks for consistency identification in task-oriented dialogue (CI-ToD). To this end, we introduced a cycle interactive learning model (CGIM), which facilitates the knowledge transfer across the three correlated tasks. Experiments show CGIM achieves state-of-the-art performance. In addition, CGIM is robust to the initial task flow order and works better in a low-resource setting, which is scalable in a real-world system deployment.

Acknowledgements

We thank all anonymous reviewers for their constructive comments. This work was supported by the National Key R&D Program of China via grant 2020AAA0106501 and the National Natural Science Foundation of China (NSFC) via grant 61976072 and 62176078.

References

Jimmy Lei Ba, Jamie Ryan Kiros, and Geoffrey E Hinton. 2016. Layer normalization. *arXiv preprint arXiv:1607.06450*.

Jiaqi Bai, Long Zhou, Ambrosio Blanco, Shujie Liu, Furu Wei, Ming Zhou, and Zhoujun Li. 2021. Jointly learning to repair code and generate commit message. *arXiv preprint arXiv:2109.12296*.

Jianzhu Bao, Bin Liang, Jingyi Sun, Yice Zhang, Min Yang, and Ruifeng Xu. 2021. Argument pair extraction with mutual guidance and inter-sentence relation graph. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 3923–3934.

Iz Beltagy, Matthew E. Peters, and Arman Cohan. 2020. Longformer: The long-document transformer.

Xiuying Chen, Zhi Cui, Jiayi Zhang, Chen Wei, Jianwei Cui, Bin Wang, Dongyan Zhao, and Rui Yan. 2020. Reasoning in dialog: Improving response generation by context reading comprehension. *arXiv preprint arXiv:2012.07410*.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, Minneapolis, Minnesota. Association for Computational Linguistics.

Nouha Dziri, Ehsan Kamalloo, Kory Mathewson, and Osmar Zaiane. 2019. Evaluating coherence in dialogue systems using entailment. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 3806–3812, Minneapolis, Minnesota. Association for Computational Linguistics.

Xiaocheng Feng, Jiang Guo, Bing Qin, Ting Liu, and Yongjie Liu. 2017. Effective deep memory networks for distant supervised relation extraction. In *Proceedings of the Twenty-Sixth International Joint Conference on Artificial Intelligence, IJCAI-17*, pages 4002–4008.

Chih-Wen Goo, Guang Gao, Yun-Kai Hsu, Chih-Li Huo, Tsung-Chieh Chen, Keng-Wei Hsu, and Yun-Nung Chen. 2018. Slot-gated modeling for joint slot filling and intent prediction. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 2 (Short Papers)*, pages 753–757, New Orleans, Louisiana. Association for Computational Linguistics.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. 2016. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778.

Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Veselin Stoyanov, and Luke Zettlemoyer. 2020. BART: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7871–7880, Online. Association for Computational Linguistics.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Roberta: A robustly optimized bert pretraining approach.

Ilya Loshchilov and Frank Hutter. 2019. Decoupled weight decay regularization. In *7th International Conference on Learning Representations, ICLR 2019, New Orleans, LA, USA, May 6-9, 2019*. OpenReview.net.

Yixin Nie, Mary Williamson, Mohit Bansal, Douwe Kiela, and Jason Weston. 2021. I like fish, especially dolphins: Addressing contradictions in dialogue modeling. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 1699–1713, Online. Association for Computational Linguistics.

Libo Qin, Wanxiang Che, Yangming Li, Mingheng Ni, and Ting Liu. 2020. Dcr-net: A deep co-interactive relation network for joint dialog act recognition and sentiment classification. *Proceedings of the AAAI Conference on Artificial Intelligence*, 34(05):8665–8672.

Libo Qin, Tailu Liu, Wanxiang Che, Bingbing Kang, Sendong Zhao, and Ting Liu. 2021a. A co-interactive transformer for joint slot filling and intent detection. In *ICASSP 2021-2021 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 8193–8197. IEEE.

Libo Qin, Tianbao Xie, Shijue Huang, Qiguang Chen, Xiao Xu, and Wanxiang Che. 2021b. Don't be contradicted with anything! ci-tod: Towards bench-

marking consistency for task-oriented dialogue system.

Sean Welleck, Jason Weston, Arthur Szlam, and Kyunghyun Cho. 2019. Dialogue natural language inference. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 3731–3741, Florence, Italy. Association for Computational Linguistics.

Zhilin Yang, Zihang Dai, Yiming Yang, Jaime Carbonell, Ruslan Salakhutdinov, and Quoc V. Le. 2020. Xlnet: Generalized autoregressive pretraining for language understanding.

Saizheng Zhang, Emily Dinan, Jack Urbanek, Arthur Szlam, Douwe Kiela, and Jason Weston. 2018. Personalizing dialogue agents: I have a dog, do you have pets too? In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 2204–2213, Melbourne, Australia. Association for Computational Linguistics.

Yinhe Zheng, Guanyi Chen, Minlie Huang, Song Liu, and Xuan Zhu. 2019. Personalized dialogue generation with diversified traits. *arXiv preprint arXiv:1901.09672*.

CorefDiffs: Co-referential and Differential Knowledge Flow in Document Grounded Conversations

Lin Xu[1], **Qixian Zhou**[2], **Jinlan Fu**[1], **Min-Yen Kan**[1], **See-Kiong Ng**[1]
[1]National University of Singapore, [2]ByteDance
{cathyxl2016,qixianzhou.mail,jinlanjonna}@gmail.com
{knmnyn,seekiong}@nus.edu.sg

Abstract

Knowledge-grounded dialog systems need to incorporate smooth transitions among knowledge selected for generating responses, to ensure that dialog flows naturally. For document-grounded dialog systems, the inter- and intra-document knowledge relations can be used to model such conversational flows. We develop a novel Multi-Document Co-Referential Graph (Coref-MDG) to effectively capture the inter-document relationships based on commonsense and similarity and the intra-document co-referential structures of knowledge segments within the grounding documents. We propose CorefDiffs, a Co-referential and Differential flow management method, to linearize the static Coref-MDG into conversational sequence logic. CorefDiffs performs knowledge selection by accounting for contextual graph structures and the knowledge difference sequences. CorefDiffs significantly outperforms the state-of-the-art by 9.5%, 7.4% and 8.2% on three public benchmarks. This demonstrates that the effective modeling of co-reference and knowledge difference for dialog flows are critical for transitions in document-grounded conversation[1].

1 Introduction

Document-grounded conversations (Moghe et al., 2018; Dinan et al., 2018; Feng et al., 2021b) is a core class of knowledge-grounded dialogs that leverage text-based knowledge segments from documents to generate informative dialog responses. This task is typically divided into two sub-tasks, given the dialog history (Dinan et al., 2018): namely, knowledge selection and response generation. Knowledge selection, which determines the content of the generated responses (Moghe et al., 2018; Dinan et al., 2018), is the crucial sub-task for dialog flow management as it leads to the manifestation of knowledge transition (Meng et al., 2020), essential for naturalistic engaging conversations.

[1]The source code has been released at https://github.com/cathyxl/coref-diffs

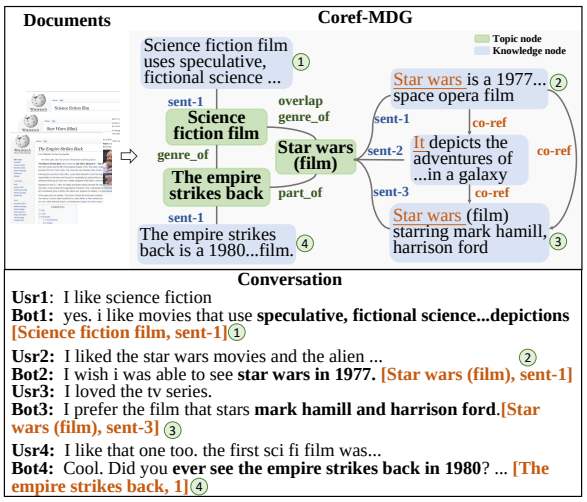

Figure 1: Co-Referential Multi-Document Graph (Coref-MDG). Topic vertices correspond to documents and are connected by commonsense/word overlap relations. Knowledge vertices are connected with its topic vertex by its document sentence index, e.g.*sent-1*, and connected to each other by co-reference (co-ref) relations. The Bot's utterances are followed by its topic and knowledge segment, e.g.[Science fiction film, sent-1].

Most existing studies on document-grounded conversations (Lian et al., 2019; Zheng et al., 2020; Zhao et al., 2020) treat knowledge selection as a matching problem between the dialog context and individual knowledge segments, independently. However, for document-grounded conversations, we posit that there is an implicit alignment between the background knowledge and conversation logic which can be learned from the underlying structural relationships of the knowledge segments within and between the grounding documents. For example, the conversation in Figure 1 exhibits document-level topic flow, from *science fiction* -> *star wars*-> *the empire strikes back*, and deep dives into the specifics of the *star wars* document (Turns 2 to 3).

To effectively exploit the relationships of the knowledge segments to guide dialog flows would require a thorough comprehension of the intra-

Proceedings of the 29th International Conference on Computational Linguistics, pages 471–484
October 12–17, 2022.

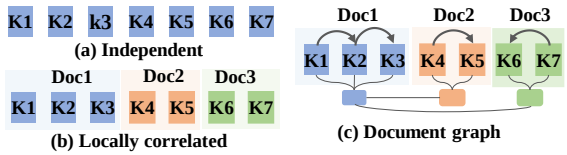

Figure 2: Exploiting knowledge segment relationships. k1-7 represent knowledge segments and doc1-3 are the grounding documents they belong to.

document discourse structures and inter-document relationships for the knowledge selection process. Existing works either ignore such relations (as illustrated in Figure 2 (a)), or exploit limited local correlations (as depicted in Figure 2 (b)), for example by encoding knowledge segments within passage context Wu et al. (2021) . In this work, we propose to capture both intra- and inter-document relationships of the knowledge segments (Figure 2 (c)) in the grounding documents to guide the smooth and natural knowledge selection and transitions for document-grounded conversations. However, how to apply such a static knowledge graph to dialog flow management has always been a problem. Many previous studies (Moon et al., 2019; Xu et al., 2021a,b) have used graph structures to constrain search (e.g. confining the next topic to neighboring areas), but have also ignored deeper integration of dialog contexts and knowledge graphs, such as optimal knowledge representation to capture dialogue flow information.

Based on the considerations above, we propose to first capture the inter- and intra-document knowledge relationships as a heterogeneous document graph, and then exploit the graph effectively for dialog flow management through fine-to-coarse contextualization — from the local word-level knowledge attentions, to knowledge interactions in document graphs, and finally to the knowledge transition flow along dialogue turns. Specifically, we design a two-level document graph consisting of topic (i.e. document) and knowledge vertices connected by inter- and intra-document relations (Figure 1). The topic vertices correspond one-to-one to the grounding documents, while the knowledge vertices refer to the knowledge segments from each document. The knowledge vertices are connected to the corresponding topic vertices they belong to. Meanwhile, the graph connects the knowledge segments within the same document by their co-referential mentions, and the documents are connected based on similarity or commonsense relationships. Hence we call the graph Multi-Document Co-referential Graph (Coref-MDG).

We then propose our CorefDiffs method which leverages Coref-MDG's graph structure and integrates dialog flow for knowledge contextualization and selection. CorefDiffs focuses on the inter-turn knowledge difference flow in the dialog histories by means of a novel differential linearization module.

Our contributions in this paper can be summarized as follows. 1) We develop Coref-MDG, a novel multi-document graph structure incorporating co-referential mentions. When leveraged in guiding document-grounded conversations in our CorefDiffs methodology, it empirically outperformed alternative graph structures; 2) We systematically study the different kinds of inter- and intra-document relations and show that document-level semantics, such as co-reference and sentence order, are significant factors for knowledge selection (Sec. 4.4); 3) Our CorefDiffs achieves state-of-the-art on WoW, Holl-E, multidoc2dial and CMU-DOG datasets, for both knowledge selection and response generation tasks.

2 Related Work

Document-grounded dialog Systems. Early works on document-grounded dialog systems (Ghazvininejad et al., 2018) focused on generating responses directly by copying words from the external documents. The subsequent availability of datasets with knowledge annotations (Dinan et al., 2018; Moghe et al., 2018) has led to the separation of the tasks of knowledge selection and response generation. For knowledge selection, most works (Dinan et al., 2018; Lian et al., 2019; Zheng et al., 2020; Zhao et al., 2020; Meng et al., 2021) in document-grounded conversations directly modeled correlations between dialog contexts and knowledge through independent matching and optimized the correlations by modeling knowledge sequence (Kim et al., 2019), increasing knowledge informativeness (Zheng et al., 2020) or distinguishing initiative roles (Meng et al., 2021). A recent work (Wu et al., 2021) boosted knowledge selection by encoding knowledge within the passage context, which demonstrates the importance of exploiting knowledge relations. Our work further explores more effective document structures and connections for this task. There is also an unpublished paper that used document

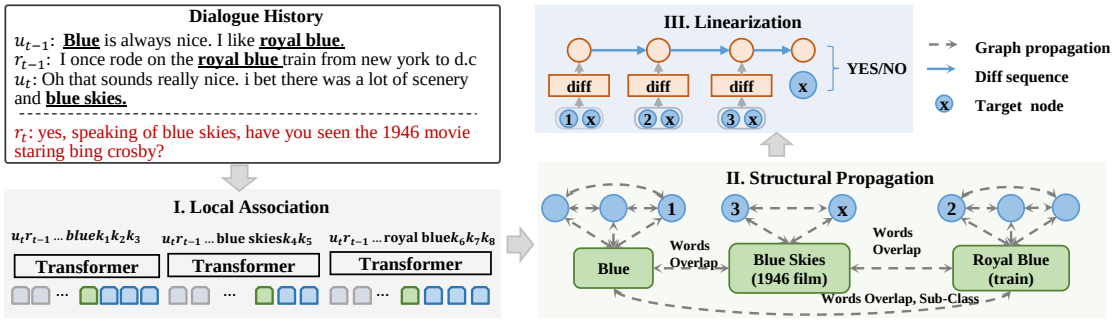

Figure 3: CorefDiffs Architecture. Green round rectangles and blue circles are topic and knowledge vertices, respectively. Steps I to III contextualize knowledge in an increasingly fine-grained manner: first, I) by vertex embedding by BERT; then, II) by propagating Coref-MDG information, and finally III) by linearizing the knowledge representations, according to the dialog's historical knowledge sequence.

semantic graphs (Li et al., 2022), while our work considers end-to-end integration of document graph and dialog flow which gives better result compared to theirs.

Knowledge Graph for Conversations. Knowledge graphs were also often used in dialog management, such as dialog transition graphs (Xu et al., 2019, 2020a) constructed from common transitions present in a dialog corpus and off-the-shelf commonsense graphs (Zhou et al., 2018a). There were also some works (Liu et al., 2019; Xu et al., 2021a) transforming unstructured text into structures or combining triplets and texts into graphs. For example, (Xu et al., 2021a) constructed key phrases into graphs according to their order in stories. Interestingly, to the best of our knowledge, co-reference mentions have not been considered in such document graph construction although it has been proved critical in learning language models for reasoning intensive NLP tasks (Dasigi et al., 2019; Ye et al., 2020). To apply knowledge graphs for dialog, many existing works (Xu et al., 2020b,a) used the graph structures to confine the search space and optimized selection through hand-crafted rewards. In contrast, we incorporate the knowledge graph into dialog management by learning knowledge representations from the graph structure.

Sequence Learning in dialog. Sequence learning is essential for conversations. Several studies (Kim et al., 2019; Zhan et al., 2021b) explored the historical knowledge sequence to select knowledge for document-grounded dialog. For example, (Kim et al., 2019) captured knowledge sequence by a latent variable, while (Zhan et al., 2021b) further proposed to learn abstract topic sequence to mitigate the issues of knowledge sparsity and knowledge transition noise. Inspired by the importance of exploiting knowledge difference (Zheng et al., 2020) for informative dialog, we extend the use of dialog knowledge differences into sequences, thus capturing the knowledge shift patterns from turns with longer distances as well as the sequential patterns of knowledge transitions in a dialogue.

3 Approach

Figure 3 shows the overall architecture of our approach. As shown in the Dialog History part, in each data sample, given a dialog history $U = \{u_{t-l}, r_{t-l}, ..., r_{t-1}, u_t\}$ of l turns and a set of grounding documents $\mathcal{D} = \{d_1, .., d_i, ..., d_{|\mathcal{D}|}\}$, where u_* and r_* are utterances from the user and chatbot, respectively. $d_i = \{k_1^i, k_2^i, ..., k_{|d_i|}^i\}$ is a document containing a bunch of knowledge segments, our task is to select the most appropriate knowledge segment from the grounding documents $\mathcal{D}$ (i.e. the knowledge selection subtask) and generate the chatbot's next response r_t based on the selected knowledge (i.e. the response generation subtask). Each grounding document d_i has a phrase t_i as its topic. For example, the document of wikipage blue has the topic phrase *blue*.

3.1 Coref-MDG Construction

We devise a Multi-Document Co-referential Graph (Coref-MDG) to capture the inter-document and the intra-document relations. Each data sample gets a specific Coref-MDG, denoted as $\mathcal{G} = \{\mathcal{V}, \mathcal{E}\}$, where $\mathcal{V}$, $\mathcal{E}$ are vertices and edges respectively.

Vertices $\mathcal{V}$. Our Coref-MDG consists of two types of Vertices: topic and knowledge vertices, as shown in Figure 1. Each topic vertex represents one of document d_i from $\mathcal{D}$ while each knowledge vertex

represents a knowledge segment k^i_j from a document d_i, hence in total $M = |\mathcal{D}|$ topic vertices and $N = |d_1| + ... + |d_{|\mathcal{D}|}|$ knowledge vertices.

Edges $\mathcal{E}$. There are also multiple types of edges in Coref-MDG. We can generally divide these edges into three categories according to their vertices: **1) edges between topics and knowledge vertices**; **2) topic edges** –these are inter-document or inter-topic edges between topic vertices; **3) knowledge edges** –intra-topic edges amongst knowledge vertices. For the first category, we simply use the order index of segment k^i_j appearing in its corresponding document d_i as the edge type, denoted as `sent_j` edge, thus knowledge vertices under different topics are not connected in Coref-MDG. The remaining two types of edges are constructed as follows.

3.1.1 Topic Edges

We posit that topic transitions in human-to-human conversations are likely to be based on the similarity or commonsense relations between two topics, such as from *sci-fi movie* to *sci-fi novel* (similarity), or from *UK* to *London* (commonsense). We introduce two corresponding types of topic edges for such topic transitions.

Word Overlap. We use the word overlaps between two topics (or documents) to measure their similarity. Specifically, we obtain the lemmas of topic phrases by spaCy[2] and judge whether the two topics have at least one identical lemma so as to determine whether these two topics vertices have a `word_overlap` edge.

Commonsense. Since the knowledge backend of the WoW (Dinan et al., 2019) came from the Wikipedia corpus, we use the WikiData[3] to obtain commonsense relations between topics. We only collected relations for the topics in the training set and for cimplicity, we kept the high-frequency relation types, for example `city_of`, while uniformly treating the low-frequency relation types as `others`.

3.1.2 Knowledge Edges

For the intra-document knowledge relations, we introduce the `coreference_link` edge. For each topic (i.e. document), the co-reference links (referring paths) within the corresponding document d_i can be extracted by a co-reference resolution model. [4] For each co-reference link, every knowledge segment on this link is connected to its mentions by a `coreference_link` edge. Aside from our proposed co-reference edges, we also model two other knowledge edge type for comparison, `common_entity` and `partial_order`. The former connects knowledge segments that share entities, while the latter captures knowledge segment's partial order. We will show later that co-reference performs best for dialog flow management.

3.2 Structural Propagation and Linearization

Next, we introduce how we contextualize each vertex in a dialog's Coref-MDG with both the graph structure and dialog flow.

3.2.1 Node Initialization

Following (Karpukhin et al., 2020; Cheng et al., 2020; Wu et al., 2021), we adopt BERT (Devlin et al., 2019) to obtain the text representations to initialize topic and knowledge vertices, as shown by the Step I in Figure 3. Specifically, we concatenate the dialog context U with each grounding document's topic phrase and knowledge segments, and feed them into the BERT encoder to get their associated representations. The concatenated input for a document d_i is thus:

$$[\text{cls}]\hat{U}_t[\text{sep}]t_i[\text{cls}]k^i_1...[\text{cls}]k^i_{|d_i|}[\text{sep}] \quad (1)$$

where $\hat{U} = [\text{usr}]u_t[\text{agt}]r_{t-1}...[\text{usr}]u_{t-l}$ is the spliced dialog context, and the role symbols [usr] and [agt] indicate utterances from the user or agent turn. We use the hidden state of the first [cls] token $\mathbf{t_i}$ (note that we use **bold** here to refer to the representation of, in this case, topic phrase t_i) as the initialized representation for the topic vertex of d_i. Similarly, the outputs of the subsequent [cls] tokens, denoted as $\{\mathbf{k^i_1}, \mathbf{k^i_2}, ..., \mathbf{k^i_{|d_i|}}\}$, are gathered and used to initialize the corresponding knowledge vertices of d_i. The process is formulated as:

$$\mathbf{t_i}, \mathbf{K^i} = \text{BERT}(U_t, d_i), i \in [1, |\mathcal{D}|] \quad (2)$$

where $\mathbf{t_i}, \mathbf{k^i_j} \in \mathbb{R}^{d_{init}}, \mathbf{K^i} = \{\mathbf{k^i_j}\}^{|d_i|}_{j=1}$. In this way, we obtain the initialized vertex embedding for a Coref-MDG as $\mathbf{H^0} = \{\mathbf{t_i}; \mathbf{K^i}\}^{|\mathcal{D}|}_{i=1} \in \mathbb{R}^{(M+N)\times d_{init}}$.

[2] https://spacy.io/, MIT License

[3] https://www.wikidata.org/wiki/Wikidata:Main_Page

[4] https://github.com/huggingface/neuralcoref, MIT License

3.2.2 Residual Graph Propagation

Knowledge transitions in document-grounded dialogs can be divided into two types namely, transition across different documents (**out-topic**) and within the same document (**intra-topic**). Transitions across different documents occur between topic vertices in our Coref-MDG and usually requires multi-hop reasoning. We use the residual graph propagation (Step II in Figure 3) to model such transitions in Coref-MDG. Specifically, we devise a variant of Relational Graph Attention Layer(RGAT) (Busbridge et al., 2019) layer with concatenated residual connection (He et al., 2016), named RES-RGAT. This layer facilitates the deeper multi-hop information propagation by avoiding information loss and the over-smooth problem (Oono and Suzuki, 2019). The output $\mathbf{H^{out}} \in \mathbb{R}^{|\mathcal{G}| \times d_{out}}$ of one RES-RGAT is the concatenation of the propagated results and the input $\mathbf{H^{in}} \in \mathbb{R}^{|\mathcal{G}| \times d_{in}}$, which is formulated as:

$$\mathbf{H^{out}} = W[\mathbf{H^{in}}, \text{RGAT}(\mathbf{H^{in}}, \mathbf{R}, \mathcal{E}, \mathcal{G})] \quad (3)$$

where $\mathbf{R} \in \mathbb{R}^{E \times d_e}$ is the embedding look-up table for all the edge types in $\mathcal{E}$, E is the number of edge types, and $W \in \mathbb{R}^{d_{out} \times 2d_{in}}$ is used for dimension transform. We stack n layers of RES-RGAT to do enough propagation based on empricially determined n. With $\mathbf{H^0}$ as input, we obtain the propagated outputs for all vertices as $\mathbf{H^G} \in \mathbb{R}^{(M+N) \times d_G}$.

3.2.3 Differential Linearization

To integrate the dialog flow information into the knowledge representations after graph propagation, we propose a novel Differential Linearization (Step III in Figure 3) method. While knowledge sequence has been used for knowledge selection in dialog (Kim et al., 2019), knowledge shift sequence (or shifting sequence), defined as the sequence of knowledge differences within each consecutive turns, is a relatively novel notion for this task. We argue that the shifting sequence is a more useful feature for learning and predicting knowledge transitions since it focuses on the difference and interaction between knowledge, leading to sharper features. It also captures the transition patterns from turns using varying distances to the current turn to further aid in the selection.

To construct the shifting sequence, we first obtain the knowledge/topic vertices that appeared in the previous chatbot turns (since we note that the labels of the user turns are inaccessible in practice). By collecting these knowledge/topic vertices' representations from $\mathbf{H}^G$, we can get the sequence $S = \{\mathbf{h}^G_{t-\tau}, ..., \mathbf{h}^G_{t-1}\}$ for knowledge and topic vertices, respectively. Here τ is the length of turns. Since we treat topic and knowledge vertex sequence identically, we will refer to them as simply vertices in the following discussion. We compare the vertex i with these historical vertices in S with a comparison function $\mathcal{F}$ to get the differential sequence for vertex i. By doing this for all vertices, we get $M + N$ such sequences:

$$\{\mathcal{F}(\mathbf{h}^G_{t-\tau}, \mathbf{h}^G_i), ..., \mathcal{F}(\mathbf{h}^G_{t-1}, \mathbf{h}^G_i)\}_{i=1}^{M+N} \quad (4)$$

$\mathcal{F}$ computes the interaction between two vectors $\mathbf{a}, \mathbf{b} \in \mathbb{R}^d$ by element-wise difference and product, defined as $\mathcal{F}(\mathbf{a}, \mathbf{b}) = [\mathbf{a} - \mathbf{b}; \mathbf{a} \odot \mathbf{b}]$ (Chen et al., 2017).

With the sequence for vertex with index i, we finalize its representations in sequential transition dependency. Specifically, we feed each sequence into a stacked GRU (Cho et al., 2014) cells and use the last hidden state as the final linearized vertex representation. We concatenate the graph representation of vertex h^G_i and the linearized output:

$$\mathbf{h}^D_i = [\text{GRU}(..., \mathcal{F}(\mathbf{h}^G_{t-1}, \mathbf{h}^G_i)); \mathbf{h}^G_i] \in \mathbb{R}^{2d_G} \quad (5)$$

The vertex representation in graph after linearization is $\mathbf{H^D} \in \mathbb{R}^{(M+N) \times 2d_G}$, which will then be used to predict the next topic and knowledge segment.

3.3 Training

Note that topic selection is an auxiliary task in our framework, apart from the knowledge selection. As such, we split the representations for topic vertices and knowledge vertices from $\mathbf{H}^D$ and obtain $\mathbf{H}^D_{tpc} \in \mathbb{R}^{M \times 2d_G}$ and $\mathbf{H}^D_{knl} \in \mathbb{R}^{N \times 2d_G}$, respectively. $\mathbf{H}^D_{tpc}$ is fed into a linear layer to obtain the topic selection scores. For the knowledge vertices, we further include their connected topic vertex representations and the in-between edge embedding to calculate the knowledge selection scores similarly with a linear layer.

Following Wu et al. (2021), we implement the history loss as an auxiliary objective function in our framework to further utilize the dialog history information. Finally, the overall objective function we adopt is formulated as follows:

Method	dialogs	Avg Turns	Domain	Document
WoW	22311	9	Open Domain	Multiple
Holl-E	9071	10	Movie	Single
CMU-DoG	4112	31	Movie	Single
MultiDoc2Dial	4796	14	Info Seek	Multiple

Table 1: Dataset statistics.

$$\mathcal{L} = \mathcal{L}_{knl} + \mathcal{L}_{tpc} + \mathcal{L}_{hist}$$
$$\mathcal{L}_{hist} = \frac{1}{2l}\sum_{hi=1}^{l}(\mathcal{L}_{knl}^{hi} + \mathcal{L}_{tpc}^{hi}) \quad (6)$$

where l is a hyperparameter representing the history length, $\mathcal{L}_{knl}$ and $\mathcal{L}_{tpc}$ are knowledge and topic losses, respectively. All of the classification objective functions in $\mathcal{L}$ are standard cross-entropy.

4 Experiments

Datasets. We validate our method on four public benchmarks for document-grounded conversation, WoW (Dinan et al., 2018), Holl-E (Moghe et al., 2018), CMU-DoG (Zhou et al., 2018b) and MultiDoc2Dial (Feng et al., 2021a). The dataset statistics are summarized in Table 1. We first conduct knowledge selection with our Coref-Diffs method and then feed the selections and dialogue history into text generation models to compare the final responses.

Evaluation metrics. We focused on evaluating the knowledge selection sub-task for the document-grounded dialog system, based on the knowledge and topic selection accuracies, denoted as **KL** and **TP**, respectively. We also explore the knowledge selection accuracy of all intra-topic data samples, whose knowledge transitions are within the same topic, denoted as **In-TP**. As for evaluating the sub-task of response generation given the dialog context and selected knowledge, we calculate the overlap of the generated response and the ground-truth with the unigram-F1(**uF1**) and bigram-F1(**bF1**).

Baselines. For the two commonly used datasets, WoW and Holl-E, we split the baselines into three categories based on their text encoder types. (i) **Non-Pretrained encoder**: Transformer+MemNet (Dinan et al., 2018) is the baseline released with the dataset WoW. DiffKS(RNN) (Zheng et al., 2020) incorporates the knowledge difference feature in knowledge selection. (ii) **BERT encoder**: BERT+PoKS, a variant of PoKS with BERT (Devlin et al., 2019) encoder, learns knowledge selection by posterior knowledge distribution. SLKS (Kim et al., 2019) captures historical knowledge sequence with a latent variable. PIPM (Chen et al., 2020) improves SLKS by addressing the problem of missing posterior distribution in test phase. CoLV (Zhan et al., 2021a) includes two collaborative variables for knowledge selection and response generation. KnowledGPT (Zhao et al., 2020) optimizes knowledge grounded dialog task by the pre-trained BERT encode and GPT-2 (Radford et al., 2019). (iii) **Passage-level BERT encoder**: DIALKI (Wu et al., 2021) encodes knowledge at passage level to capture knowledge segment relations as we do in CorefDiffs. For response generation, given that the above-mentioned methods adopted different generators, we uniformly replaced their generators with a prompt-based generator PrefixTuning (Li and Liang, 2021) for a fair comparison, thus forming the baselines with "*" in Table 3. For MultiDoc2Dial and CMU-DoG, we compare our method with the current state-of-the-art DPR+RAG (Lewis et al., 2020b) and DoHA (Prabhumoye et al., 2021), respectively. The generators used are fine-tuned BART-large.

4.1 Implementation Details.

The BERT-base models in all our experiments used the Huggingface Transformers[5] (Wolf et al., 2020). We trained the model with Adam (Kingma and Ba, 2015) optimizer with initial learning rate 1e-5. A linear scheduler with a warm-up strategy in 5k steps was used. The maximum history length l was empirically set to 4 for WoW, 2 for Holl-E, 3 for CMU-DoG and 4 for MultiDoc2Dial to achieve the best performance. The number of the stacked Res-RGAT was set to 2. It took around 5 and 10 epochs to achieve the reported performance by 4 nvidia V100 GPUs. We will release all the codes and the hyper-parameters settings for reproduction.

4.2 Automatic Evaluations

Knowledge Selection. The knowledge selection results on the four datasets are presented in Table 2 and 4. CorefDiffs significantly outperforms all other methods regardless of the encoders they used. Compared to the best performance achieved by DIALKI, CorefDiffs improves by 9.5% and 5.9% and is the first to achieve knowledge accuracy over 40% on both the WoW Test Seen and Unseen sets. For Holl-E, CorefDiffs also performs the best, with gains of at least 7.4% in knowledge selection. For

[5]https://github.com/huggingface/transformers

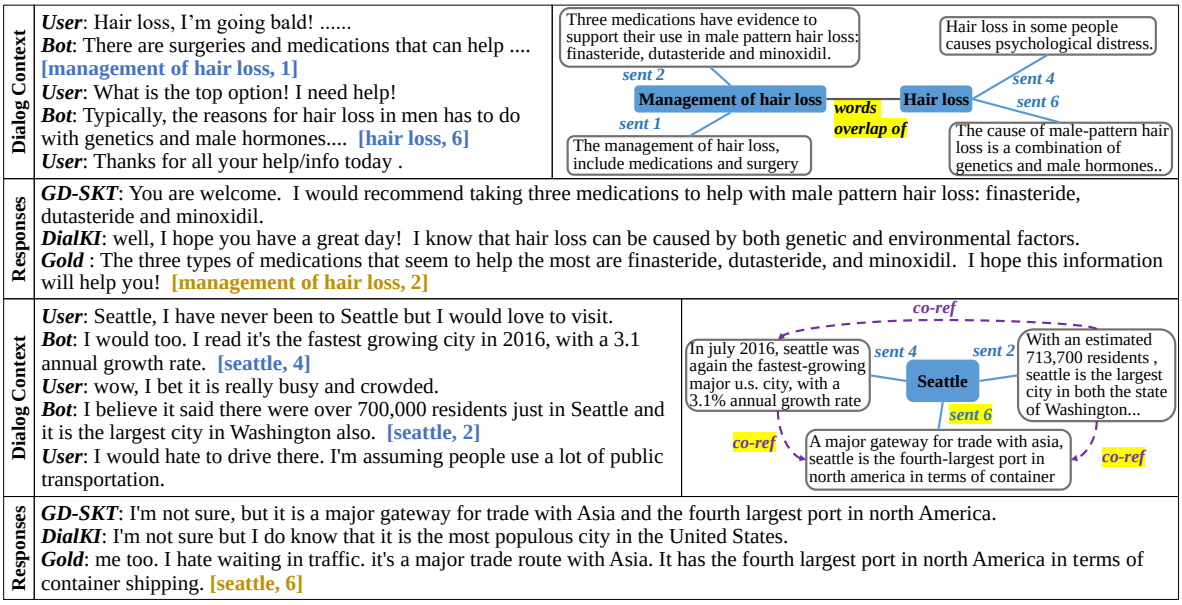

Figure 4: Two generation examples from WoW. The bold words in "[]" indicate the knowledge. For example, **[hair loss, 6]** represents the 6-th knowledge sentence in the document with topic **hair loss**. Our method chose the right knowledge for both examples compare to DIALKI owing to the well-designed graph structure.

Method	WoW (Seen)	WoW (Unseen)	Holl-E
TMN	22.5	12.2	22.7
DiffKS(RNN)	25.6	18.6	33.5
BERT+PoKS	25.5	14.1	27.6
SLKS	26.8	18.3	29.2
PIPM	27.8	19.7	30.7
CoLV	30.1	18.9	32.7
DukeNet	26.4	19.6	30.0
KnowledGPT	28.0	25.4	-
DIALKI	32.9	35.5	-
CorefDiffs	**42.4**	**41.4**	**40.9**
w/o Diff-Seq	40.8	39.5	39.7
w/o Diff	40.9	40.1	40.1
w/o Res-RGAT	35.5	36.5	39.5

Table 2: The knowledge selection results measured by accuracy on WoW and Holl-E.

Method	WoW (Seen)		WoW (Unseen)		Holl-E	
	uF1	bF1	uF1	bF1	uF1	bF1
SLKS(TM+Copy)	19.3	6.8	16.1	4.2	29.2	22.3
DukeNet(TM+Copy)	19.3	6.3	17.1	4.7	30.6	23.1
SLKS*	20.2	7.3	17.5	5.3	-	-
DiffKS*	21.5	7.6	20.0	6.3	30.7	23.9
KnowledGPT*	22.0	8.2	20.8	7.4	-	-
DIALKI*	22.0	8.0	22.2	8.1	-	-
CorefDiffs	**25.2**	**10.7**	**25.8**	**10.8**	**38.4**	**31.8**

Table 3: Response generation results on WoW and Holl-E. Methods with '(TM+Copy)' and '*' used generator Transformer + Copy mechanism and PrefixTuning.'-' indicates the method didn't do experiment on the dataset.

MultiDoc2Dial our method outperforms state-of-the-art by 8.2% in knowledge selection accuracy. CMU-DoG has no ground-truth knowledge, so we only report generation results. The substantial enhancements across all datasets strongly suggest that CorefDiffs has benefited from modeling document structures and knowledge relations in the grounding documents with differential dialog flow learning.

Response Generation. Tables 3 and 4 show the results of response generation on all the four datasets. We applied PrefixTuning (Li and Liang, 2021) to generate responses with the corresponding dialog context and selected knowledge as the input for WoW and Holl-E, while for MultiDoc2Dial and CMU-DoG, we followed previous works using BART-Large. The PrefixTuning obtained a comparable performance with fewer learnable parameters and extrapolated better to unseen topics than fine-tuning method. Again, CorefDiffs obtains best performance in all generation metrics on four datasets, which we attribute to the large margins on knowledge selection.

Ablation Study. To study the impact of each of the modules in CorefDiffs, we conduct three experiments, as shown in the lower part of Table 2. For *w/o Diff-Seq*, we remove the Differential Linearization. *w/o Diff* uses the normal knowledge sequence instead of the shifting sequence. *w/o* RES-RGAT removes the Residual Graph Propagation. After removing RES-RGAT, we observe a steep drop in knowledge selection accuracy, which proves that knowledge representations updated by graph prop-

Method	MultiDoc2Dial		CMU-DOG
	uF1	KL	uF1
DoHA	-	-	22.8
DPR+RAG	33.7	24.9	-
CorefDiffs	**39.3**	**33.1**	**23.9**

Table 4: Experimental Results on MultiDoc2Dial and CMU-DOG.

agation on Coref-MDG is well aligned with knowledge distribution in next turn. This also shows that purely relying on local correlations within passage context (Step I in Figure 3) is not as good as using higher-level document structures. In addition, *w/o diff-seq* also presents lower knowledge selection accuracy, showing its importance to enhance dialog flow modeling upon graph propagation. More importantly, by comparing *w/o diff-seq* and *w/o Diff*, we notice that using shifting sequences outperforms normal ones, thus validating our earlier argument that shifting features are sharper and more effective for dialog knowledge flow.

4.3 Case Study

Why does Coref-MDG work on knowledge selection? To answer this question, we visualize two typical examples, shown in Figure 4. The *dialog Context* rows are dialog histories, and the generated responses of different methods are listed in the *Response* row. We compare our CorefDiffs with DIALKI and the Gold (ground-truth) response. The first example performed topic change from "hair loss" to "management of hair loss". CorefDiffs chose the right knowledge topic, "management of hair loss", while DIALKI repeated the knowledge mentioned in the earlier conversation turn. The reason is that CorefDiffs was able to do so is because it had referred to the `word_overlap` connection between "hair loss" and "management of hair loss", whereas DIALKI did not consider such inter-topic relations. For the second example, the knowledge transition is intra-topic (knowledge in consecutive turns belonging to the same topic/document). Our method successfully predicts the right knowledge due to the co-reference relation between these knowledge sentences within the "seattle" document, whereas the response generated by DIALKI — even with passage-level knowledge correlations encoded — missed the long dependency from the second to the sixth sentence.

4.4 Graph Analysis

To study effects of the different type of relations in Coref-MDG on topic/knowledge selection accuracies. We did more experiments on WoW. We craft 3 Coref-MDG variants lie in three categories for relations between topics. (1) *w/o TP*: removing all topic edges; (2) *w/o TP overlap*: removing the word overlap edges; (3) *w/o TP wikigraph*: removing the commonsense edges. Another three variants for exploring the relations between knowledge vertices are as follows: (4) *w/o KG*: removing all knowledge edges (that is co-reference link); (5) *+KG common entity*: applying entity edges between knowledge instead; (6) *+KG partial order*: employing partial order edges between knowledge. We also remove the sentence order edges between topic and knowledge vertices and formed a variant (7) *w/o TP-KG*. The results of the above experiments are listed in Table 5, from which we get the following **conclusions**:

(i) Coref-MDG performs the best in knowledge selection compared to other graph structures. Removing or replacing edge types in Coref-MDG, such as the edges between topic vertices (Exp. 1-3), knowledge vertices (Exp. 4-6), or topic and knowledge vertices (Exp. 7), can cause a drop in topic or knowledge selection both on Seen or Unseen settings. Moreover, sometimes using other kind of edge leads to worse results than their absence. For example, in Exp. 4 and 5, *w/o KG* performs better than + *KG common entity* in Unseen.

(ii) Topic and knowledge selection accuracies are affected by their relevant relations in Coref-MDG. In Exp. 1 and 7, without topic edges or topic-knowledge edges, the model achieves lowest TP. In Exp. 4-7, model achieves lower KL without suitable knowledge relations.

(iii) Topic and Knowledge relations also facilitate each other. In Exp. 1 and 4 even removing topic relation or knowledge relations, the model still achieves better TP and KL compared to DIALKI (no graph relation used).

(iv) Knowledge relations improve intra-topic knowledge selection. As shown by results in 4th and 7th columns, by comparing In-TP results in Exp. 1-3 and Exp. 4-6, after removing knowledge edges, the In-TP drops a lot, thus we conclude that relations between knowledge enhance the intra-topic knowledge selection.

Method	WoW Seen			WoW Unseen		
	KL	TP	In-TP	KL	TP	In-TP
DIALKI	32.9	70.0	42.3	35.5	71.6	43.5
CorefDiffs	42.4	76.1	51.1	41.4	77.7	49.2
1. w/o TP	42.1	74.0	50.6	39.8	75.2	47.2
2. w/o TP overlap	42.4	75.9	51.2	40.9	77.7	48.1
3. w/o TP wikigraph	42.3	75.9	50.9	41.1	77.5	48.8
4. w/o KG	35.4	75.7	44.6	37.1	77.2	46.1
5. + KG common entity	35.4	74.6	44.4	36.4	75.9	43.8
6. + KG partial order	36.6	75.9	45.7	37.1	76.8	45.5
7. w/o TP-KG	40.5	73.5	49.5	38.3	74.5	45.7

Table 5: Graph Comparisons in selection accuracy.

5 Conclusion

We show the significance of utilizing the document's semantic structures and relations for managing dialog flow. We embody these relations in our novel multi-document graph Coref-MDG which models co-referential knowledge mention links and inter-document relations. Our analysis of Coref-MDG yields insights of how the difference among intra- or inter-document relations affect the final topic and knowledge selection accuracy. For example, we find that coreference links and topic-knowledge sentence order relations are critical relations. We then build dynamically-sensitive dialog flows via our CorefDiffs method, which integrates the modeling of dialog difference flow with the prior knowledge represented in Coref-MDG. CorefDiffs demonstrates that it is possible to seamlessly integrate static graph structures with dynamic dialog-specific flows, improving document-grounded conversations.

Ethical Impact

Document-grounded dialog technology has broad application prospects in open-domain dialog, emotional escort robots, intelligent assistants, etc. This work focuses on knowledge selection which plays a significant role in dialog management of multi-turn dialog for document-grounded conversations. All datasets we used in this work were privacy filtered and content moderated by the dataset authors (Dinan et al., 2019; Moghe et al., 2018). However, advanced dialog knowledge selection techniques may also enable bots to select harmful content on the Internet and generate inappropriate or biased responses to users. Future work should take this into consideration.

Acknowledgements

We thank all reviewers for their valuable comments. This work is supported by the National Research Foundation, Singapore under its AI Singapore Programme (AISG Award No: AISG-GC-2019-001-2A) and Industry Alignment Fund – Prepositioning (IAF-PP) Funding Initiative. Any opinions, findings and conclusions or recommendations expressed in this material are those of the author(s) and do not reflect the views of the National Research Foundation, Singapore.

References

Dan Busbridge, Dane Sherburn, Pietro Cavallo, and Nils Y Hammerla. 2019. Relational graph attention networks. *arXiv preprint arXiv:1904.05811*.

Qian Chen, Xiaodan Zhu, Zhen-Hua Ling, Si Wei, Hui Jiang, and Diana Inkpen. 2017. Enhanced LSTM for natural language inference. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1657–1668, Vancouver, Canada. Association for Computational Linguistics.

Xiuyi Chen, Fandong Meng, Peng Li, Feilong Chen, Shuang Xu, Bo Xu, and Jie Zhou. 2020. Bridging the gap between prior and posterior knowledge selection for knowledge-grounded dialogue generation. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 3426–3437, Online. Association for Computational Linguistics.

Hao Cheng, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2020. Probabilistic assumptions matter: Improved models for distantly-supervised document-level question answering. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 5657–5667, Online. Association for Computational Linguistics.

Kyunghyun Cho, Bart Van Merriënboer, Caglar Gulcehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. 2014. Learning phrase representations using rnn encoder-decoder for statistical machine translation. *arXiv preprint arXiv:1406.1078*.

Pradeep Dasigi, Nelson F. Liu, Ana Marasović, Noah A. Smith, and Matt Gardner. 2019. Quoref: A reading comprehension dataset with questions requiring coreferential reasoning. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 5925–5932, Hong Kong, China. Association for Computational Linguistics.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, Minneapolis, Minnesota. Association for Computational Linguistics.

Emily Dinan, Stephen Roller, Kurt Shuster, Angela Fan, Michael Auli, and Jason Weston. 2018. Wizard of wikipedia: Knowledge-powered conversational agents. In *International Conference on Learning Representations*.

Emily Dinan, Stephen Roller, Kurt Shuster, Angela Fan, Michael Auli, and Jason Weston. 2019. Wizard of wikipedia: Knowledge-powered conversational agents. In *International Conference on Learning Representations*.

Song Feng, Siva Sankalp Patel, Hui Wan, and Sachindra Joshi. 2021a. MultiDoc2Dial: Modeling dialogues grounded in multiple documents. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 6162–6176, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Song Feng, Siva Reddy, Malihe Alikhani, He He, Yangfeng Ji, Mohit Iyyer, and Zhou Yu, editors. 2021b. *Proceedings of the 1st Workshop on Document-grounded Dialogue and Conversational Question Answering (DialDoc 2021)*. Association for Computational Linguistics, Online.

Marjan Ghazvininejad, Chris Brockett, Ming-Wei Chang, Bill Dolan, Jianfeng Gao, Wen-tau Yih, and Michel Galley. 2018. A knowledge-grounded neural conversation model. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 32.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. 2016. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778.

Vladimir Karpukhin, Barlas Oguz, Sewon Min, Patrick Lewis, Ledell Wu, Sergey Edunov, Danqi Chen, and Wen-tau Yih. 2020. Dense passage retrieval for open-domain question answering. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 6769–6781, Online. Association for Computational Linguistics.

Byeongchang Kim, Jaewoo Ahn, and Gunhee Kim. 2019. Sequential latent knowledge selection for knowledge-grounded dialogue. In *International Conference on Learning Representations*.

Diederik P Kingma and Jimmy Ba. 2015. Adam: A method for stochastic optimization.

Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Veselin Stoyanov, and Luke Zettlemoyer. 2020a. BART: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7871–7880, Online. Association for Computational Linguistics.

Patrick Lewis, Ethan Perez, Aleksandra Piktus, Fabio Petroni, Vladimir Karpukhin, Naman Goyal, Heinrich Küttler, Mike Lewis, Wen-tau Yih, Tim Rocktäschel, et al. 2020b. Retrieval-augmented generation for knowledge-intensive nlp tasks. *Advances in Neural Information Processing Systems*, 33:9459–9474.

Sha Li, Di Jin Madhi Namazifar, Mohit Bansal, Heng Ji, Yang Liu, and Dilek Hakkani-Tur. 2022. Enhanced knowledge selection for grounded dialogues via document semantic graphs.

Xiang Lisa Li and Percy Liang. 2021. Prefix-tuning: Optimizing continuous prompts for generation. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 4582–4597, Online. Association for Computational Linguistics.

Rongzhong Lian, Min Xie, Fan Wang, Jinhua Peng, and Hua Wu. 2019. Learning to select knowledge for response generation in dialog systems. In *IJCAI International Joint Conference on Artificial Intelligence*, page 5081.

Zhibin Liu, Zheng-Yu Niu, Hua Wu, and Haifeng Wang. 2019. Knowledge aware conversation generation with explainable reasoning over augmented graphs. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 1782–1792, Hong Kong, China. Association for Computational Linguistics.

Chuan Meng, Pengjie Ren, Zhumin Chen, Zhaochun Ren, Tengxiao Xi, and Maarten de Rijke. 2021. Initiative-aware self-supervised learning for knowledge-grounded conversations. In *Proceedings of the 44th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 522–532.

Chuan Meng, Pengjie Ren, Zhumin Chen, Weiwei Sun, Zhaochun Ren, Zhaopeng Tu, and Maarten de Rijke. 2020. Dukenet: A dual knowledge interaction network for knowledge-grounded conversation. In *Proceedings of the 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 1151–1160.

Nikita Moghe, Siddhartha Arora, Suman Banerjee, and Mitesh M. Khapra. 2018. Towards exploiting background knowledge for building conversation systems.

In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 2322–2332, Brussels, Belgium. Association for Computational Linguistics.

Seungwhan Moon, Pararth Shah, Anuj Kumar, and Rajen Subba. 2019. OpenDialKG: Explainable conversational reasoning with attention-based walks over knowledge graphs. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 845–854, Florence, Italy. Association for Computational Linguistics.

Kenta Oono and Taiji Suzuki. 2019. Graph neural networks exponentially lose expressive power for node classification. In *International Conference on Learning Representations*.

Shrimai Prabhumoye, Kazuma Hashimoto, Yingbo Zhou, Alan W Black, and Ruslan Salakhutdinov. 2021. Focused attention improves document-grounded generation. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 4274–4287, Online. Association for Computational Linguistics.

Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, Ilya Sutskever, et al. 2019. Language models are unsupervised multitask learners. *OpenAI blog*, 1(8):9.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Remi Louf, Morgan Funtowicz, Joe Davison, Sam Shleifer, Patrick von Platen, Clara Ma, Yacine Jernite, Julien Plu, Canwen Xu, Teven Le Scao, Sylvain Gugger, Mariama Drame, Quentin Lhoest, and Alexander Rush. 2020. Transformers: State-of-the-art natural language processing. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 38–45, Online. Association for Computational Linguistics.

Zeqiu Wu, Bo-Ru Lu, Hannaneh Hajishirzi, and Mari Ostendorf. 2021. DIALKI: Knowledge identification in conversational systems through dialogue-document contextualization. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 1852–1863, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Jun Xu, Zeyang Lei, Haifeng Wang, Zheng-Yu Niu, Hua Wu, and Wanxiang Che. 2021a. Enhancing dialog coherence with event graph grounded content planning. In *Proceedings of the Twenty-Ninth International Conference on International Joint Conferences on Artificial Intelligence*, pages 3941–3947.

Jun Xu, Zeyang Lei, Haifeng Wang, Zheng-Yu Niu, Hua Wu, Wanxiang Che, Jizhou Huang, and Ting Liu. 2021b. Coherent dialog generation with query graph. *Transactions on Asian and Low-Resource Language Information Processing*, 20(6):1–23.

Jun Xu, Haifeng Wang, Zheng-Yu Niu, Hua Wu, Wanxiang Che, and Ting Liu. 2020a. Conversational graph grounded policy learning for open-domain conversation generation. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 1835–1845, Online. Association for Computational Linguistics.

Jun Xu, Haifeng Wang, Zhengyu Niu, Hua Wu, and Wanxiang Che. 2020b. Knowledge graph grounded goal planning for open-domain conversation generation. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pages 9338–9345.

Lin Xu, Qixian Zhou, Ke Gong, Xiaodan Liang, Jianheng Tang, and Liang Lin. 2019. End-to-end knowledge-routed relational dialogue system for automatic diagnosis. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, pages 7346–7353.

Deming Ye, Yankai Lin, Jiaju Du, Zhenghao Liu, Peng Li, Maosong Sun, and Zhiyuan Liu. 2020. Coreferential Reasoning Learning for Language Representation. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 7170–7186, Online. Association for Computational Linguistics.

Haolan Zhan, Lei Shen, Hongshen Chen, and Hainan Zhang. 2021a. CoLV: A collaborative latent variable model for knowledge-grounded dialogue generation. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 2250–2261, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Haolan Zhan, Hainan Zhang, Hongshen Chen, Zhuoye Ding, Yongjun Bao, and Yanyan Lan. 2021b. Augmenting knowledge-grounded conversations with sequential knowledge transition. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 5621–5630, Online. Association for Computational Linguistics.

Xueliang Zhao, Wei Wu, Can Xu, Chongyang Tao, Dongyan Zhao, and Rui Yan. 2020. Knowledge-grounded dialogue generation with pre-trained language models. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 3377–3390, Online. Association for Computational Linguistics.

Chujie Zheng, Yunbo Cao, Daxin Jiang, and Minlie Huang. 2020. Difference-aware knowledge selection for knowledge-grounded conversation generation. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 115–125, Online. Association for Computational Linguistics.

Hao Zhou, Tom Young, Minlie Huang, Haizhou Zhao, Jingfang Xu, and Xiaoyan Zhu. 2018a. Commonsense knowledge aware conversation generation with graph attention. In *IJCAI*, pages 4623–4629.

Kangyan Zhou, Shrimai Prabhumoye, and Alan W Black. 2018b. A dataset for document grounded conversations. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 708–713, Brussels, Belgium. Association for Computational Linguistics.

A Implementation Details

We set the maximum lengths of model input to 512, which is also the longest input limit for the BERT model, in order to fit the longer passage text as much as possible on both datasets. We employ a Linear layer to transform the output features of BERT from 768 to 320 to reduce memory usage. The edge embedding size is set to 64. The hidden size and headers of Res-RGAT are 1024 and 8 respectively while the alpha value of Graph Attention Network is 0.2. We utilize a unidirectional stacked GRU model for Differential Sequential Learning, the number of GRU layers is 2.

For response generation, we apply PrefixTunning (Li and Liang, 2021) on BART (Lewis et al., 2020a) large model to learn the responses generation model based on the knowledge selection results from the previous stage. We use the prefix length 200 and the hidden dimension of 800 for all the methods using PrefixTuning generator. The PrefixTuning generator takes about 4 hours and 30 epoch to become converged during training on 4 V100 32G GPUs, which is much faster and more resource saving than fine-tuning BART large.

B Dataset Processing Details

WoW. There are more than 130k different documents from Wikipedia in WoW training set. We keep 350 high-frequency relations from the Wiki knowledge graph, covering these 130k documents. The top-10 wiki relations with corresponding frequency are shown as follows:

1. ('subclass of', 27015)
2. ('facet of', 11381)
3. ('sport', 10646)
4. ('performer', 9482)
5. ('part of', 6892)
6. ('manufacturer', 5742)
7. ('instance of', 5551)
8. ('history of topic', 5517)
9. ('has part', 5445)
10. ('follows', 5077)

As shown in Table 6, for topic relations, we found the `word_overlap` edges is denser than the commonsense edges from wikiData, giving the average edge number of 8.11 and 2.89, respectively. While for knowledge relations, the `coreference_link` has much less average number of relations in one sample than other two types relations, which again proves that `coreference_link` with more accurate knowledge relations lead to better knowledge selection results without introducing wrong structures information to CorefDiffs framework.

	Topic Relations		**Knowledge relations**		
	WordOverlap	*WikiGraph*	*Partial*	*EntityLink*	*Coreference*
Freq	8.11	2.89	61.18	87.52	15.90

Table 6: Average number of different kinds of relations in one sample on the WoW training set.

Holl-E. Different from WoW, each sample of Holl-E has only one topic, which is the movie in this session of conversation. There are four types of information for each movie in Holl-E, which are plots, comments, reviews, and table information. So we simply divide all the knowledge sentences of each movie into four topics. As the absence of common sense relations of such topics in Holl-E, we count the co-occurrence relationship of all topics in the training set as the relations between topics in Holl-E. The relations between knowledge are as same as the WoW, using coreference relations in passage text. The relations between knowledge and topics are sentence order of knowledge sentence in the original text, which is also used in WoW. **CMU-DoG.** CMU-DoG is a document-grounded conversation dataset about movie, which is the same as Holl-E. The difference is that CMU-DoG includes only one grounding document(one topic) at each dialog turn. The relations of topics is absent as there is only one topic in grounding document. The relations between knowledge are as same as the WoW and Holl-E with coreference relations. The relations between knowledge and topics are sentence order of knowledge sentence in the passage, which is also consistent with WoW and Holl-E. On the other hand, CMU-DoG didn't contain the gold knowledge of knowledge selection task. We adopt unigram F1 score as similarity function, selecting the knowledge closest to the ground-truth response as gold knowledge to train the selector model.
MultiDoc2Dial. MultiDoc2Dial includes multiple grounding documents at each dialog turn. We con-

struct the graph following the steps of Holl-E. However, MultiDoc2Dial introduces a span prediction task to locate knowledge set in the original document instead of knowledge selection. But that's ok, it easy for our framework to transfer downstream task by using two independent classifier to predict both start knowledge segment and end knowledge segment instead of one classifier for knowledge selection. Simultaneously, we replace the metric from knowledge accuracy to knowledge EM, which is used in MultiDoc2Dial. For convenience, we still use **KL** in Table 4 to denote the EM metric in MultiDoc2Dial.

C Analysis on Partial Order Edge

For partial order relations, we explored the effects of different hops. Hop-k partial order relation means each knowledge vertex is connected with k knowledge vertices behind according to the sentence order. As shown in Fig 5, hop-2 partial relation performed the best. A hop that was too large or too small could cause information loss or introduce many erroneous connections.

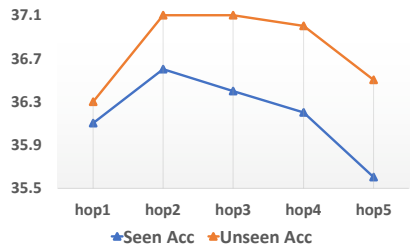

Figure 5: Knowledge accuracy for partial order with different hops.

SelF-Eval: Self-supervised Fine-grained Dialogue Evaluation

Longxuan Ma and **Ziyu Zhuang** and **Weinan Zhang*** and **Mingda Li** and **Ting Liu**
Research Center for Social Computing and Information Retrieval, Harbin Institute of Technology
lxma,zyzhuang,wnzhang,mdli,tliu@ir.hit.edu.cn

Abstract

This paper introduces a novel **Self**-supervised **F**ine-grained Dialogue **Eval**uation framework (**SelF-Eval**). The core idea is to model the correlation between turn quality and the entire dialogue quality. We first propose a novel automatic data construction method that can automatically assign fine-grained scores for arbitrarily dialogue data. Then we train **SelF-Eval** with a multi-level contrastive learning schema which helps to distinguish different score levels. Experimental results on multiple benchmarks show that SelF-Eval is highly consistent with human evaluations and better than the state-of-the-art models. We give a detailed analysis of the experiments in this paper. Our code is available on GitHub.

1 Introduction

Dialogue systems (DS) aim to satisfy human needs (Shum et al., 2018; Yan, 2018; Gao et al., 2019) such as information, communication, entertainment, etc. Appraising the quality of the DS responses reflects the system's capability and provides insights into required further improvements (Finch and Choi, 2020; Deriu et al., 2021). Among the commonly used evaluation metrics, human evaluation is of high reliability but expensive to conduct, automatic metrics used in language generation (Perplexity (Bengio et al., 2000)) or machine translation (BLEU (Papineni et al., 2002), ROUGE (Lin, 2004), etc.) are easy to conduct but ineffective to reflect the dialogue quality (Liu et al., 2016; Novikova et al., 2017). Therefore, researchers have made great efforts to find more reliable automatic evaluation metrics that are highly correlated with human evaluation (Lowe et al., 2017; Tao et al., 2018; Mehri and Eskénazi, 2020a).

The current automatic dialogue evaluation metrics leverage semantic information (Huang et al., 2020; Mehri and Eskénazi, 2020b; Ye et al., 2021)

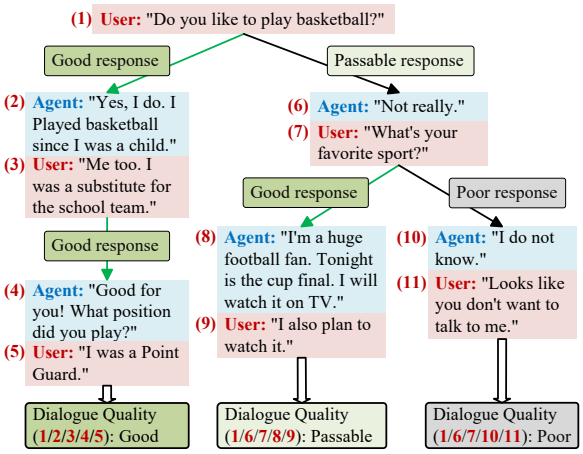

Figure 1: The fine-grained relationships between turn-level and dialogue-level quality.

to measure dialogue quality. For example, when evaluating the response quality, they either compute the semantic similarity between dialogue context and the generated response (Xu et al., 2018a; Tao et al., 2018; Dziri et al., 2019; Ghazarian et al., 2019) or measure the soft semantic overlap between ground-truth response and the model-generated one (Lowe et al., 2017; Xu et al., 2018b; Zhao et al., 2019; Zhang et al., 2020a; Yuan et al., 2021). When evaluating (Zhang et al., 2021) the overall dialogue quality, they either learn a dialogue-level representation for rating directly (Mesgar et al., 2020; Zhang et al., 2021) or calculate the score with the help of other indirect assists (Mehri and Eskénazi, 2020a). However, recent studies (Mehri and Eskénazi, 2020a; Sai et al., 2021; Yeh et al., 2021) show that current models can only work well for measuring the response or evaluating the entire dialogue. They could not perform well in both situations at the same time. It means that the dialogue representation they learned (Zhang et al., 2021) could not reflect both turn quality and the entire dialogue quality.

The dialogue quality is affected by all turns' qualities in it (Gopalakrishnan et al., 2019) and this effect is accumulated in a multi-turn dialogue

*Corresponding author

Proceedings of the 29th International Conference on Computational Linguistics, pages 485–495
October 12–17, 2022.

(Li et al., 2021). Figure 1 shows these quality correlations between turns and dialogue. Each turn is marked with a serial number. Three dialogue examples are starting with the same user turn (1). The left example (1/2/3/4/5) shows that two good agent responses result in good overall quality. The middle (1/6/7/8/9) and right (1/6/7/10/11) examples show how lower-quality agent responses result in different dialogue qualities (passable or poor). The current open-domain dialogue evaluation methods fail to model the fine-grained correlations between turn quality and dialogue quality, which entail a poor dialogue representation for evaluation.

In this paper, we introduce an evaluation method that explicitly models the correlations between turn quality and dialogue quality. Specifically, we aim to learn a dialogue representation that can reflect each turn's contribution, so that the evaluation score obtained by this representation aligns turn quality with the dialogue quality. To this end, we need to first obtain large amounts of dialogue data that reflects fine-grained correlations between turns quality and dialogue quality, then train an evaluation model to measure the fine-grained correlations. The contributions of this paper are:

- To the best of our knowledge, we are the first to explicitly model the fine-grained correlation between turns and the entire dialogue for open-domain dialogue evaluation.
- We introduce a simple but effective data construction method to align the turn-level quality with the overall dialogue quality. We design a **Sel**f-supervised **F**ine-grained Dialogue **Eval**uation model (SelF-Eval) with a multi-level contrastive learning (MLCL) method. Our code and data are publicly available: https://github.com/royny/SelF-Eval.
- Experiments on multiple benchmarks show that SelF-Eval: 1) can evenly distinguish different replacement levels; 2) builds the correlations between turn qualities and dialogue qualities; 3) gets better correlation scores with human ratings than the state-of-the-art (SOTA) models.

2 Related Work

We first survey evaluation metrics in open-domain dialogue (sections 5.1 and 5.2), then compare related work in task-oriented dialogue (section 5.3).

2.1 Calculation of Semantic Overlap

In this category, metrics are designed to measure the semantic similarity between the generated response and the dialogue context (Xu et al., 2018a; Tao et al., 2018; Ghazarian et al., 2019; Pang et al., 2020) or soft semantic overlap between the generated response and the reference response (Lowe et al., 2017; Xu et al., 2018b; Zhang et al., 2020a; Zhao et al., 2019; Yuan et al., 2021). Dziri et al. (2019) presented interpretable metrics for evaluating topic coherence by making use of distributed sentence representations. COMET (Rei et al., 2020) evaluated machine translation quality with a pre-trained model by minimizing the distance of the hypothesis with both reference and source text. The most similar work to ours is from Ye et al. (2021) that measures quantifiable coherence scores. *The differences between their work and ours are: 1) they focus on turn-level evaluations while we aim to evaluate both turn and dialogue-levels. Their method models the relationship between dialogue context and response while we model the fine-grained correlations between turns and the entire dialogue; 2) their method relies on the multi-level human annotations for dialogue quality while our method is free from these constrain. SelF-Eval is trained in a self-supervised manner, using synthetic dialogue data and automatically annotated scores.*

2.2 Regression to a Reference Score

In this category, metrics learn to evaluate dialogue with scores that represent pre-defined dialogue attributes. BLEURT (Sellam et al., 2020) trained a BERT model with synthetic data and fine-tuned it on human ratings. GRADE (Huang et al., 2020) introduced dialogue topic transitions for coherence evaluation. USR (Mehri and Eskénazi, 2020b) leveraged RoBERTa and a regression model to approximate the specific scores rated by annotators. Mehri and Eskénazi (2020a) computed the log-likelihood of DialoGPT generating predefined positive or negative comments as the score. The most similar work to ours in this category is from Mesgar et al. (2020), they utilized dialogue act labels to help dialogue level representation learning and assist the performance of dialogue-level coherence evaluation. *The difference between their work and ours are: 1) they use dialogue act to assist the dialogue representation learning in a multi-task learning framework while we only use dialogue information; 2) their method only measures the co-*

herence of dialogue while ours measures multiple attributes of dialogue; 3) their method aims to distinguish good samples from bad ones while ours can assign fine-grained scores for each sample.

2.3 Related Work in Task-oriented DS

Besides the open-domain dialogue, there is also work in task-oriented dialogue similar to ours. They assume that users start a dialogue with a task-sensitive patience budget and the dialogue is finished by users when the task is completed or the budget runs out. Their model estimates user satisfaction at each turn and consumes some remaining budget. *The differences between their work and ours are: 1) they focus on task-oriented dialogue with explicitly dialogue purpose while ours are open-domain dialogue with no such goals; 2) they need expensive training and data collecting pipeline while ours do not need; 3) they set up an overall budget which is consumed during the dialogue while we learn a dialogue-level representation for evaluation.* When adopting our method to evaluate task-oriented dialogues, information such as intents and request types is required to determine task completion. Our model will require further improvements to utilize this information.

3 Our Proposed Method

3.1 Problem Statement

Given an n rounds (2*n turns) dialogue **D** = [A_1,B_1,A_2,B_2,...,A_n,B_n] where A_i/B_i represents the i-th ($i \in \{1, 2, ..., n\}$) turn from human-A/machine-B, respectively. The dialogue evaluation model takes **D** as input and outputs a quality score (a scalar value) for it.

3.2 Data Construction

We need training data with quantitative annotation on both turn-level and dialogue-level (Figure 1). However, only very few dialogue data today have these kinds of labels (Gopalakrishnan et al., 2019) and the models trained with this kind of data are restricted by domain adaptability and generality. Inspired by previous work (Mesgar et al., 2020; Zhang et al., 2021), we adopt a replacement strategy that perturbs a dialogue at the semantic level. In this strategy, the easily accessible human-human dialogue is considered positive. The negative samples for this dialogue are constructed by replacing some turns with randomly selected turns from other dialogues[1]. These randomly selected turns bring multiple negative effects (topically in-congruent, semantic confusion, etc.) w.r.t the current dialogue context. However, different from previous works that replaced a fixed number of turns in a n>0 rounds dialogue, we set multiple replacement strategies and randomly replace $i \in \{0, 1, ..., n\}$ turns in it. One sample with more replacements is considered of worse overall quality (Gopalakrishnan et al., 2019; Li et al., 2021). Specifically, we assign a score 1 to the original dialogue and assign a score $(n - i)/n$ to the new dialogue that replaces i turns. By aligning the replaced turn numbers with a reference score, we get the required training data. Meanwhile, we avoid the quantity and domain limitation of human-annotated data and can easily obtain a large amount of fine-grained training data in different domains.

Notice that 1) we treat each round with equal weight in this paper, but there may be differences when replacing a turn in the first round (usually a greeting round) or the last one; 2) we hypothesize a linear relationship between the number of replacement turns and the overall dialogue quality, which is not necessarily true. For example, replacing 3 turns and more than 3 turns in the same 6-rounds dialogue may cause the same damage to the overall dialogue quality. We leave these problems for future work.

3.3 Training

During training, we want to minimize the distance between the predicted score and the reference score. This is a difficult regression task because **1)** unlike the coherence degree between dialogue context and response or semantic relationships in a Natural Language Inference, our automatic score as regression target lacks clear semantic meaning; **2)** dialogues with different replaced turns may have the same reference score. For example, the reference score of replacing 1 turn in a 2 rounds dialogue is equal to the score of replacing 2 turns in a 4 rounds dialogue; **3)** when the replacement level increases, it is hard for the model to distinguish the small differences. For example, the reference score of replacing 4 turns in an 8 rounds dialogue is close to the score of replacing 5 turns in an 8 rounds dialogue (0.5 and 0.375, respectively). To smooth the convergence process, we divide the training stage

[1]To ensure the generality of our method, we did not use more complex sampling strategies.

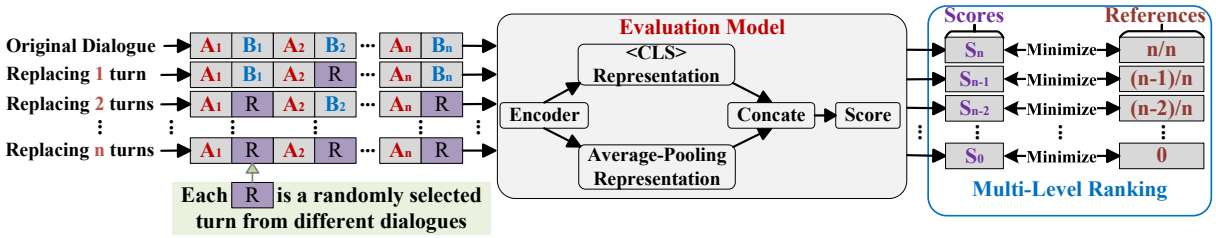

Figure 2: The training procedure of SelF-Eval. A_i and B_i is the i-th turn from speaker A and B, respectively. R is the randomly replaced utterances. The replacement positions can be arbitrary rounds.

into **coarse** and **fine-grained** and name the training stages Multi-level Contrastive learning (MLCL) schema. MLCL is model agnostic and can be used for any similar tasks.

Figure 2 shows the training process of our model. We choose RoBERTa (Liu et al., 2019) as encoder. The input for a dialogue sample **D** is [<CLS>, A_1, B_1, ..., A_n, B_n], where "<CLS>" is a special token and n is different for different **D**. We first obtain two kinds of dialogue representations. The *<CLS>* representation is the output vector E_{cls} of the first token <CLS>. The *Pooling* representation $E_{pooling}$ is obtained by average-pooling all token representations of [A_1, B_1, A_2, B_2, ..., A_n, B_n]. The final dialogue representation h_D is [E_{cls};$E_{pooling}$], where [;] is the concatenation operation. The h_D is passed through a Multi-layer Perceptron (MLP) to get the predicted quality score $S^{\mathbf{D}}$ for **D**:

$$S^{\mathbf{D}} = \sigma(W_2 \cdot \mu(W_1 \cdot h_D + b_1) + b_2), \quad (1)$$

where $W_{1,2}$ and $b_{1,2}$ are training parameters; σ/μ is the sigmoid/tanh function, respectively.

3.3.1 The Coarse Training Stage

Formally, given a training corpus $\mathbf{C} = \{\mathbf{D}_m\}_{m=1}^{M}$ where $\mathbf{D}_m$ is the m-th dialogue with n_m-rounds. For each $\mathbf{D}_m$, we can replace $i \in \{0, 1, ..., n_m\}$ turns and get a replaced version of $\mathbf{D}_m$, named $\mathbf{D}_m^i$. Each replacement level has its own reference score $(n_m - i)/n_m$. In the first training stage, we combine a separation loss and a compactness loss as the multi-level ranking (mlr) loss. The mlr loss helps the model learn a coarse granularity ranking ability for multi-levels.

The separation loss aims to separate the features of different replacement examples by distinguishing their scores. For each replacement level $i \in \{0, 1, ..., n_m\}$, we first calculate a centroid score $S^{\mathbf{D}_m^i} = \frac{1}{K_i} \sum_{k=1}^{K_i} S_k^{\mathbf{D}_m^i}$ where $S_k^{\mathbf{D}_m^i}$ is the quality score of a dialogue example with i turns replaced, K_i is the number of contrastive samples for $\mathbf{D}_m$ in this replacement level[2]. The separation loss between different replacement levels is:

$$l_m^{sep} = \sum_{j=0}^{n_m-1} \sum_{l=j+1}^{n_m} \max(0, \omega * \lambda + S^{\mathbf{D}_m^j} - S^{\mathbf{D}_m^l}), \quad (2)$$

where $\lambda = 1/(n_m - 1)$ is the lower bound for the distance between two centroid scores[3], $\omega = l - j$ is the weight used for amplifying the lower bound according to the quality-level gap.

The compactness loss aims to compact the examples within the same level, which served as a regularization role to avoid outlier exceptions for each level. Specifically, the dialogue quality score $S_k^{\mathbf{D}_m^i}$ for $k \in \{1, 2, ..., K_i\}$ is forced to be closer to the corresponding centroid $S^{\mathbf{D}_m^i}$ as follows:

$$l_m^{com} = \sum_{i=0}^{n_m} \sum_{k=1}^{K_i} \max(0, |S^{\mathbf{D}_m^i} - S_k^{\mathbf{D}_m^i}| - \mu), \quad (3)$$

where μ is the upper bound for the distance between the centroid of a certain replacing level and the score within this level[4]. The mlr loss is:

$$L^{mlr} = \sum_{m=1}^{M} (l_m^{seq} + l_m^{com}). \quad (4)$$

The original multi-level ranking method is proposed by Ye et al. (2021) and has three secondary losses: separation loss, compactness loss, and ordering loss. The difference between the multi-level ranking methods we used and what they used is that we remove the ordering loss and compute the difference instead of the L1 distance between different centroid scores so that the ordering loss is covered by the separation loss. Our method can save training time and keep equal performance.

[2]For example, when replacing i turns in a n rounds dialogue, we have total $K_i = n!/(i!(n - i)!)$ contrastive samples.

[3]For example, λ=0.5 when there are 3 reference score levels. The expecting 3 centroids are around 1, 0.5, and 0.

[4]For example, if we set λ to 0.3 and μ to 0.1 when there are 3 reference levels. The expecting 3 level ranges are around $[0.9, 1]$, $[0.4, 0.6]$, and $[0, 0.1]$.

Dataset	**D**s	turns	words	turns/**D**
Empathetic	24,846	107,208	1.7M	4.3
ConvAI-2	18,878	278,192	3.3M	14.7
DailyDialog	12,096	100,360	1.4M	8.3
DailyDialog++	19,071	215,625	1.2M	4.3
GRADE	1,200	2,400	61K	2.0
FED	500	5,603	49K	11.2
DSTC-9	2,200	59,840	533K	27.2

Table 1: Statistics of datasets. "**D**" means dialogue.

3.3.2 The Fine-grained Training Stage

After the coarse training stage, the model has learned to rank multi-level scores, which can be seen as an approximate fitting to the reference labels. To make the training more smooth and more efficient, we add an R-drop loss (Liang et al., 2021) aside from the mlr loss to obtain a more robust representation for each dialogue. The robust representation will help the convergence of the model. Specifically, one input dialogue will go through the model twice and obtain two scores $S^{\mathbf{D}_m^j}_{k,first}$ and $S^{\mathbf{D}_m^j}_{k,second}$, then the model will minimize the distance between the two scores as follows:

$$L^{drop} = \sum_{m=1}^{M}\sum_{j=0}^{n_m}\sum_{k=1}^{K_i}(S^{\mathbf{D}_m^j}_{k,first} - S^{\mathbf{D}_m^j}_{k,second})^2, \quad (5)$$

The overall Loss of the fine-grained training stage L^{final} is computed as follows:

$$L^{final} = L^{mlr} + L^{drop}. \quad (6)$$

4 Experimental Settings

4.1 Datasets

The datasets used in this paper are shown in Table 1. **Empathetic Dialogue** dataset (Rashkin et al., 2019) simulates real life dialogue in which the interlocutor needs to identify and recognize the feelings of others. **ConvAI-2** (Zhang et al., 2018; Dinan et al., 2019) mimics the scene where each interlocutor tries to understand each other by incorporating persona information. **DailyDialog** (Li et al., 2017) reflects our daily communication and covers different topics such as interpersonal relationships and health. **DailyDialog++** (Sai et al., 2020) is a multi-reference open-domain dialogue dataset with 3 groups (relevant, irrelevant, and adversarial) of responses for each context, each group has 5 different responses. **GRADE** dataset (Huang et al., 2020) contains 300 dialogue examples from Empathetic Dialogue and DailyDialog, and 600 dialogue examples from ConvAI2. Each example has 2 turns with human-annotated relevance scores. **FED** (Mehri and Eskénazi, 2020a) is a set of human-machine and human-human conversations with eighteen fine-grained quality scores in both turn and dialogue levels. **DSTC-9** (Gunasekara et al., 2020) was collected on the DialPort platform through direct interaction between real users and open-domain chit-chat systems.

4.2 Baselines

We choose the following SOTA models: **GPT-2** (Pang et al., 2020) computes the log-likelihood of the response conditional on the the dialogue context normalized by the length of the response; **QuantiDCE** (Ye et al., 2021) uses BERT (Devlin et al., 2019) to get dialogue-level representations and proposes a multi-level ranking method to train a quantifiable turn-level coherence metric; **FED** (Mehri and Eskénazi, 2020a) computes the log-likelihood of DialoGPT (Zhang et al., 2020b) generating predefined positive or negative comments as the quality score. It can measure both turn and dialogue-level qualities; **DynaEval** (Zhang et al., 2021) integrates turn representations from RoBERTa into dialogue-level representation with a graph convolutional network, then adopts contrastive learning to distinguish positive and negative samples.

We also test with different settings of SelF-Eval. The model shown in Figure 2 is named **SelF-Eval(full)**, in which the training dialogue can be any rounds. The first different setting is that we use fixed rounds of dialogue data for training. We set all dialogues to 2 rounds and have 3 different replacement strategies: the original dialogue and replacing 1 or 2 turns. This setting is named **SelF-Eval(simple)**. Besides, we have the following settings for the ablation study. **SelF-Eval(-mlr)** and **SelF-Eval(-drop)** means we remove the multi-level ranking loss and D-drop loss, respectively. When removing the mlr loss, we use a binary cross-entropy (BCE) loss instead. It means the model makes a binary decision between original dialogue and dialogue with replacements. We use BCE loss to show our multi-level ranking method is better than a two-level loss when learning a dialogue representation for evaluation.

4.3 Implementation Details

The setting of the baseline models follows the papers that proposed them. The pre-trained models (BERT, RoBERTa, DialoGPT, GPT-2) are based on the public Pytorch implementation (https://github.com/huggingface/transformers).

model	Rep-0	Rep-1	Rep-2	overall
QuantiDCE	0.688	0.486	0.654	0.609
DynaEval	0.812	0.595	0.699	0.702
SelF-Eval(simple)	0.962	0.891	0.904	0.919
SelF-Eval(full)	**0.973**	**0.898**	**0.914**	**0.928**

Table 2: Accuracy of predicting replacement levels.

model	Pearson	Spearman	average
DynaEval	0.093	0.101	0.097
FED	0.128	0.120	0.124
SelF-Eval(simple)	0.158	0.165	0.162
SelF-Eval(full)	**0.163**	**0.173**	**0.168**

Table 3: Evaluation on DSTC-9 dialogue-level quality.

The hyper-parameters which are not introduced in this section follow the original implementation in the link. During fine-tuning, we truncate the input dialogue length to 512 tokens. Among the 7 datasets we used, only DSTC-9 has dialogue examples that exceed 512 tokens and the percentage is 13.6%. We set the max contrastive sample number to 8. All models are learned with Adam optimizer with $\beta_1 = 0.9$ and $\beta_2 = 0.999$. We use a single Tesla A100s GPU with 40GB memory, the batch size is 15. The average training time for each epoch is around 4 hours (2 hours for the first training stage and 2 hours for the second training stage). The initial learning rate is set to 0.005 and decays to 0.002 in the second stage. A dropout of 0.5 is also applied. When training SelF-Eval(full), the GPU memory occupation is 39GB. μ is set to 0.1.[5]

4.4 Evaluation Metrics

Following previous works (Mehri and Eskénazi, 2020a; Zhang et al., 2021), we choose two metrics to correlated with manual evaluations. **Pearson Correlation** (Freedman et al., 2007) measures the linear correlation between two sets of data. **Spearman Correlation** (Zar, 2005) assesses the monotonic relationships between two variables. Besides, we use **Accuracy** measures the percentage of correct ranking for multi-level replacement.

5 Experimental Results and Analysis

We aim to answer the following questions about SelF-Eval: **(Q1)** *can it assign reasonable scores for multiple replacement levels?* (See section 5.1) **(Q2)** *does it outperform state-of-the-art methods and truly model the correlations between turns/dialogue?* (See section 5.2 and 5.3) **(Q3)** *how do the different components contribute to its performance?* (See section 5.4) **(Q4)** *what can we learn from case study?* (See section 5.5)

[5]When evaluating dialogue-level qualities with turn-level metrics, we measure all context-response pairs in a dialogue and use their average as the final score. When evaluating turn-level qualities with dialogue-level metrics, we treat the context-response pair as an entire dialogue.

5.1 Ranking Capability (Q1)

This experiment tests whether an evaluation model assigns higher scores for dialogues with less replacement. Table 2 shows the accuracy results of QuantiDCE, DynaEval, and SelF-Eval(simple/full), all models 1) are trained with DailyDialog++ and test with the DailyDialog++ test set; 2) use base-sized pre-trained models as backbones. QuantiDCE is chosen because 1) it is trained for classification and fits perfectly for this experiment; 2) it represents the SOTA turn-level metric. DynaEval is chosen because it is the SOTA dialogue-level metric. We define 3 replacement levels: the original dialogue (Rep-0), the dialogue with 1 replacing turn (Rep-1), and more than 1 replacing turn (Rep-2). Each replacement level has 5010 samples.

We can see that SelF-Eval(full) gets the highest performance on all replacement levels. Between the multi-level ranking models, Self-Eval(full) outperforms QuantiDCE by 52.4%. Between the dialogue-level ranking models, SelF-Eval(full) surpasses DynaEval by 32.2%. Notably, the accuracy gaps between Rep-(0, 1, and 2) of SelF-Eval(full) are 0.075/-0.016, which are much smaller than the gaps of QuantiDCE (0.202/-0.168) and DynaEval (0.217/-0.104). The results show that 1) SelF-Eval can evenly distinguish the 3 replacement levels; 2) the MLCL method we used shows advantages over multi-level learning in QuantiDCE and contrastive learning in DynaEval.

5.2 Experiments on DSTC-9 (Q2)

Table 3 shows the experimental results on DSTC-9 data. The DSTC-9 dataset is difficult to evaluate because of two reasons: 1) it contains direct interaction between real users and multiple open-domain chit-chat systems. These chit-chat systems are trained with dialogue data in different domains compared with ours. In another word, the DSTC-9 dialogue data is out-of-domain for our model and can be used to test the generality of our method; 2) the average turns in a dialogue is around 27.2, which is the longest among all datasets we used and also much longer than the training data we used. It is difficult for evaluation models to give a score for

Dialogue Aspects	GPT-2	Q-DCE	FED	D-Eval	S-E(s)	S-E(f)	(-drop)	(-mlr)	(-drop,-mlr)	Human
Dialogue-level (11 quality aspects) Spearman Correlation										
Coherence	0.122	0.191	0.251	0.424	0.423	**0.436**	0.332	0.340	0.137	0.809
Error Recovery	*0.097*	0.109	0.165	0.351	0.363	**0.393**	0.252	0.269	0.135	0.840
Consistency	0.093	0.332	*0.116*	0.326	0.246	**0.347**	0.233	0.318	0.124	0.562
Diversity	0.145	*-0.014*	**0.420**	0.342	0.283	0.263	0.197	0.116	*0.022*	0.789
Topic Depth	*0.094*	*-0.054*	**0.476**	0.375	0.316	0.327	0.204	0.177	*0.004*	0.833
Likability	0.178	0.098	0.262	0.357	0.345	**0.390**	0.285	0.275	*0.074*	0.838
Understanding	*0.073*	0.210	0.306	0.373	0.364	**0.406**	0.329	0.306	0.108	0.809
Flexibility	0.135	0.093	0.293	**0.361**	0.307	0.317	0.233	0.184	*0.082*	0.816
Informativeness	0.119	*0.063*	0.288	0.302	0.311	**0.318**	0.184	0.194	*0.019*	0.806
Inquisitiveness	*0.070*	0.115	0.163	0.294	0.401	**0.421**	0.309	0.267	*0.085*	0.769
Overall	0.121	0.140	**0.443**	0.428	0.428	0.435	0.252	0.272	*0.055*	0.830
Turn-level (9 quality aspects) Spearman Correlation										
Interestingness	-0.097	-0.163	**0.408**	0.197	0.146	0.183	0.143	0.142	0.063	0.819
Engagement	*-0.096*	-0.138	**0.318**	0.119	0.149	0.206	0.126	0.128	0.047	0.798
Specificity	-0.114	-0.171	**0.267**	0.161	0.097	0.169	0.139	0.112	*0.069*	0.790
Relevance	-0.103	*0.085*	0.152	0.171	0.263	**0.282**	0.174	0.175	0.101	0.753
Correctness	*0.041*	*0.080*	0.133	0.165	0.233	**0.291**	0.165	0.163	*0.083*	0.780
S.Appropriateness	*-0.081*	0.112	0.155	0.112	0.208	**0.247**	0.143	0.132	*0.068*	0.682
Understandable	*-0.076*	0.195	0.111	0.116	0.136	**0.173**	0.110	0.111	*0.046*	0.522
Fluency	-0.154	*0.071*	**0.224**	*0.016*	0.095	*0.038*	*0.034*	*0.036*	*-0.047*	0.714
Overall	*-0.090*	*0.014*	0.209	0.207	0.255	**0.292**	0.146	0.140	*0.067*	0.820

Table 4: Comparison of both dialogue and turn level Spearman correlations on the FED evaluation dataset. Q-DCE/D-Eval/S-E(s)/S-E(f)/"**S**." is short for QuantiDCE/DynaEval/SelF-Eval(simple)/SelF-Eval(full)/Semantically, respectively. Scores with p-values larger than 0.01 are italicized (indicating statistical insignificance).

such a long conversation. In this experiment, all models are fine-tuned on DailyDialog++ and the RoBERTa-based models are all first pre-trained on Empathetic Dialogue, ConvAI-2, and DailyDialog.

Pearson and Spearman correlations between the model-generated scores and the corresponding human evaluation scores are computed in Table 3. FED and DynaEval are chosen because they are the SOTA dialogue-level evaluation models. We can see that both SelF-Eval(simple) and SelF-Eval(full) largely outperform SOTA baselines even though all models are affected by the out-of-domain and long conversation problems. The results show that SelF-Eval is capable of learning a dialogue representation for evaluating dialogue-level quality even in a difficult dataset such as DSTC-9. To further verify the generality of this evaluation ability, we test with other out-of-domain datasets in the following sections.

5.3 Experiments on FED (Q2)

Table 4 shows the experimental results on FED data. In both dialogue and turn-level evaluations, Spearman correlations between the model-generated scores and the corresponding human evaluation scores are computed. Models are trained in the same setting as experiments on DSTC-9.

5.3.1 Dialogue-level Evaluation

There are 11 different aspects of the FED dialogue-level evaluation. GPT-2 and QuantiDCE are SOTA turn-level evaluation metrics. They evaluate a dialogue based on the aggregation of scores of all the context-response pairs within the dialogue. We can observe that most of their correlation scores (21 out of 22) on dialogue aspects are lower than those of FED and DynaEval. The results are consistent with the conclusion of previous studies (Yeh et al., 2021) that turn-level quality evaluation may be insufficient to assess the dialogue-level performance.

FED has the highest scores on Diversity, Topic Depth, and Overall. These results may indicate that the DialoGPT-based evaluation model (FED) is better at measuring these three attributes than the RoBERTa-based models (DynaEval and SelF-Eval). The reason is that DialoGPT uses a large amount of Reddit data for training. The diverse topics and variation expressions in Reddit data provide DialoGPT with more insights on these attributes, especially the dialogue-level Overall attribute. In contrast, DynaEval and SelF-Eval are trained with fewer dialogue data (fewer topics and variation expressions). The DynaEval focuses on modeling the dependency between pairs of utterances and the SelF-Eval focuses on modeling the correlations between turns and the entire dialogue. They are more useful for evaluating Coherence, Error Recovery, and Consistency aspects which reflect the interaction between turns. Specifically, SelF-Eval owns the highest correlation scores in 7 out of 11 dialogue aspects (Coherence, Error Recovery, Con-

sistency, likability, Understanding, informativeness, and Inquisitiveness) and the second-highest correlation scores on Flexibility and Overall. SelF-Eval successfully learns to measure these attributes with our replacement strategies. The MLCL training method captures the various dialogue attributes and entails good dialogue-level representations. The dialogue-level evaluation tasks are benefiting from this representation. One way to improve the Diversity and Topic Depth scores of RoBERTa-based models is to pre-train them with dialogue data that contains more topics and domains. We can also notice that SelF-Eval(full) is better than SelF-Eval(simple) in most aspects. It means the simplified training method used by SelF-Eval(simple) is not as strong as the original method introduced in Figure 2.

5.3.2 Turn-level Evaluation

There are 9 different aspects of the FED turn-level evaluation. The turn-level metrics (GPT-2 and QuantiDCE) only get better correlations on 6 out of 18 aspects than the dialogue-level metrics (FED and DynaEval). The results indicate that the generality of these two turn-level evaluation models is not strong. They work well only in constrained environments or on specific datasets. The FED model achieves the highest correlation on Interestingness, Engagement, Specificity, and Fluency. The reason is that the DialoGPT used by FED is trained with an auto-regressive mode and models language generation word by word. DialoGPT focuses more on the token-level correlations and is effective for evaluating the naturalness of an utterance. In contrast, all the RoBERTa-based models (DynaEval and SelF-Eval) perform poorly for token-level aspects. This is because they focus on the correlations in the turn level and do not pay enough attention to the token level. One way to strengthen the fluency and Specificity aspects of SelF-Eval is to introduce token-level perturbation strategies in training data, such as word drop and addition (Sai et al., 2021). These strategies provide negative samples with semantical or grammatical mistakes which may also be used for setting multi-level turn qualities for training. We consider this token-level perturbation as future work. What's more, we have a similar finding to Zhang et al. (2021) that SelF-Eval(s) and FED complement each other at turn-level. It means that they both perform well in aspects that the other one is not good at. SelF-Eval achieves the highest correlation in Relevance, Correctness, Semantically Appropriateness, Understandable, and Overall. The SelF-Eval(simple)/(full) outperforms the best baseline 39.7%/22.0% on turn-level Overall, respectively. It also has the second-highest scores on Engagement and Specificity. The results are consistent with the dialogue-level evaluation where SelF-Eval has good results on aspects that reflect the interaction between turns. As in the dialogue-level, SelF-Eval(full) is still better than SelF-Eval(simple) on most aspects in turn-level.

U1: My partner left me the other day. **U2**: That's rough, I'm sorry to hear that. **R**: Being a punching bag in a relationship is no good. It's a 2 way street. Is your partner doing their part? **Scores**(Human / SelF-Eval / DynaEval): 0.77 / 0.84 / 0.50
U1: i was so stressed when i found out that i did not get accepted in my dream college. **U2**: Oh no. Did you have a good backup plan? **R**: thats cool, i hope you have a good time. **Scores**(Human / SelF-Eval / DynaEval): 0.25 / 0.09 / 0.46

Table 5: Case study on GRADE. U1/U2 are the dialogue context and R is the response to be evaluated.

To sum up the experiments on the FED dataset, SelF-Eval performs well for both dialogue-level and turn-level evaluations, especially the latter. The reason is that the training process of SelF-Eval not only models the correlation between turns and the entire dialogue but also models the inner connection between context and response. Our method successfully aligns the semantic information shared by turns and dialogue and shows good domain adaptability (on both DSTC-9 and FED).

5.4 Ablation Study (Q3)

Table 4 also shows the ablation study of the SelF-Eval(full). Removing R-drop loss (-drop) in the fine-grained training stage causes more declines than (-mlr) in dialogue-level evaluations. This is because the R-drop loss helps SelF-Eval to learn more robust dialogue representations. Replacing the mlr loss with BCE loss (-mlr) causes more declines than (-drop) in turn-level evaluations. This indicates that the mlr loss helps to distinguish the turn replacement levels and the semantic inconsistency information caused by the replacements. When removing both mlr and R-drop losses, the performance declines significantly and the results become statistical insignificance.

5.5 Case Study (Q4)

We randomly select 2 examples from GRADE for the case study (Table 5). The task is to evaluate the response when giving dialogue context. We com-

pare the human rating (the relevance scores) with the scores given by DynaEval and SelF-Eval(full). They are both based on RoBERTa and could provide more insight into our model. In both cases, the scores given by SelF-Eval are closer to the human rating score than DynaEval. This is consistent with the experimental results in the turn-level Relevance of the FED dataset. In both cases, scores from SelF-Eval are more polarized than the human evaluations. This indicates that humans may be reluctant to give extreme scores and SelF-Eval could improve its performance by penalizing scores that are too extreme. However, whether this penalizing works for dialogue attributes other than relevance requires further study.

6 Conclusion

We propose to measure dialogue quality by modeling the fine-grained correlations between turns and the entire dialogue. We introduce our data construction method and SelF-Eval model. Experiments show that SelF-Eval builds fine-grained correlations and gets better correlation scores with human ratings than SOTA models. We think our method may have two potential applications: 1) It can be used alone in the evaluation of dialogue tasks after training the SelF-Eval model with a large amount of in-domain data; 2) It can be combined with other evaluation models (such as FED) to evaluate the dialogue task by integrating the advantages of different evaluation models. In the future, we would like to improve our method by 1) employing multi-granularity turn-level scores; 2) modeling the nonlinear relationships between replacement numbers and dialogue quality.

7 Acknowledgments

This paper is supported by the Science and Technology Innovation 2030 Major Project of China (No. 2021ZD0113302) and National Natural Science Foundation of China (No. 62076081, No. 61772153 and No. 61936010).

References

Yoshua Bengio, Réjean Ducharme, and Pascal Vincent. 2000. A neural probabilistic language model. In *NIPS*, pages 932–938. MIT Press.

Jan Deriu, Álvaro Rodrigo, Arantxa Otegi, Guillermo Echegoyen, Sophie Rosset, Eneko Agirre, and Mark Cieliebak. 2021. Survey on evaluation methods for dialogue systems. *Artif. Intell. Rev.*, 54(1):755–810.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT 2019, Minneapolis, MN, USA, June 2-7, 2019, Volume 1 (Long and Short Papers)*, pages 4171–4186. Association for Computational Linguistics.

Emily Dinan, Varvara Logacheva, Valentin Malykh, Alexander H. Miller, Kurt Shuster, Jack Urbanek, Douwe Kiela, Arthur Szlam, Iulian Serban, Ryan Lowe, Shrimai Prabhumoye, Alan W. Black, Alexander I. Rudnicky, Jason Williams, Joelle Pineau, Mikhail S. Burtsev, and Jason Weston. 2019. The second conversational intelligence challenge (convai2). *CoRR*, abs/1902.00098.

Nouha Dziri, Ehsan Kamalloo, Kory Wallace Mathewson, and Osmar R. Zaïane. 2019. Evaluating coherence in dialogue systems using entailment. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT 2019, Minneapolis, MN, USA, June 2-7, 2019, Volume 1 (Long and Short Papers)*, pages 3806–3812. Association for Computational Linguistics.

Sarah E. Finch and Jinho D. Choi. 2020. Towards unified dialogue system evaluation: A comprehensive analysis of current evaluation protocols. In *Proceedings of the 21th Annual Meeting of the Special Interest Group on Discourse and Dialogue, SIGdial 2020, 1st virtual meeting, July 1-3, 2020*, pages 236–245. Association for Computational Linguistics.

David Freedman, Robert Pisani, Roger Purves, and Ani Adhikari. 2007. Statistics.

Jianfeng Gao, Michel Galley, and Lihong Li. 2019. Neural approaches to conversational AI. *Foundations and Trends in Information Retrieval*, 13(2-3):127–298.

Sarik Ghazarian, Johnny Tian-Zheng Wei, Aram Galstyan, and Nanyun Peng. 2019. Better automatic evaluation of open-domain dialogue systems with contextualized embeddings. *CoRR*, abs/1904.10635.

Karthik Gopalakrishnan, Behnam Hedayatnia, Qinglang Chen, Anna Gottardi, Sanjeev Kwatra, Anu Venkatesh, Raefer Gabriel, and Dilek Hakkani-Tür. 2019. Topical-chat: Towards knowledge-grounded open-domain conversations. pages 1891–1895. ISCA.

R. Chulaka Gunasekara, Seokhwan Kim, Luis Fernando D'Haro, Abhinav Rastogi, Yun-Nung Chen, Mihail Eric, Behnam Hedayatnia, Karthik Gopalakrishnan, Yang Liu, Chao-Wei Huang, Dilek Hakkani-Tür, Jinchao Li, Qi Zhu, Lingxiao Luo, Lars Liden, Kaili Huang, Shahin Shayandeh, Runze Liang, Baolin Peng, Zheng Zhang, Swadheen Shukla, Minlie Huang, Jianfeng Gao, Shikib Mehri, Yulan Feng,

Carla Gordon, Seyed Hossein Alavi, David R. Traum, Maxine Eskénazi, Ahmad Beirami, Eunjoon Cho, Paul A. Crook, Ankita De, Alborz Geramifard, Satwik Kottur, Seungwhan Moon, Shivani Poddar, and Rajen Subba. 2020. Overview of the ninth dialog system technology challenge: DSTC9. *CoRR*, abs/2011.06486.

Lishan Huang, Zheng Ye, Jinghui Qin, Liang Lin, and Xiaodan Liang. 2020. GRADE: automatic graph-enhanced coherence metric for evaluating open-domain dialogue systems. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing, EMNLP 2020, Online, November 16-20, 2020*, pages 9230–9240. Association for Computational Linguistics.

Yanran Li, Hui Su, Xiaoyu Shen, Wenjie Li, Ziqiang Cao, and Shuzi Niu. 2017. Dailydialog: A manually labelled multi-turn dialogue dataset. In *Proceedings of the Eighth International Joint Conference on Natural Language Processing, IJCNLP 2017, Taipei, Taiwan, November 27 - December 1, 2017 - Volume 1: Long Papers*, pages 986–995. Asian Federation of Natural Language Processing.

Ziming Li, Dookun Park, Julia Kiseleva, Young-Bum Kim, and Sungjin Lee. 2021. A data-driven approach to estimate user satisfaction in multi-turn dialogues. *CoRR*, abs/2103.01287.

Xiaobo Liang, Lijun Wu, Juntao Li, Yue Wang, Qi Meng, Tao Qin, Wei Chen, Min Zhang, and Tie-Yan Liu. 2021. R-drop: Regularized dropout for neural networks. In *Advances in Neural Information Processing Systems 34: Annual Conference on Neural Information Processing Systems 2021, NeurIPS 2021, December 6-14, 2021, virtual*, pages 10890–10905.

Chin-Yew Lin. 2004. Rouge: A package for automatic evaluation of summaries. In *Text summarization branches out*, pages 74–81.

Chia-Wei Liu, Ryan Lowe, Iulian Serban, Michael Noseworthy, Laurent Charlin, and Joelle Pineau. 2016. How NOT to evaluate your dialogue system: An empirical study of unsupervised evaluation metrics for dialogue response generation. In *EMNLP*, pages 2122–2132. The Association for Computational Linguistics.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Roberta: A robustly optimized BERT pretraining approach. *CoRR*, abs/1907.11692.

Ryan Lowe, Michael Noseworthy, Iulian Vlad Serban, Nicolas Angelard-Gontier, Yoshua Bengio, and Joelle Pineau. 2017. Towards an automatic turing test: Learning to evaluate dialogue responses. In *ACL (1)*, pages 1116–1126. Association for Computational Linguistics.

Shikib Mehri and Maxine Eskénazi. 2020a. Unsupervised evaluation of interactive dialog with dialogpt. In *Proceedings of the 21th Annual Meeting of the Special Interest Group on Discourse and Dialogue, SIGdial 2020, 1st virtual meeting, July 1-3, 2020*, pages 225–235. Association for Computational Linguistics.

Shikib Mehri and Maxine Eskénazi. 2020b. USR: an unsupervised and reference free evaluation metric for dialog generation. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, ACL 2020, Online, July 5-10, 2020*, pages 681–707. Association for Computational Linguistics.

Mohsen Mesgar, Sebastian Bücker, and Iryna Gurevych. 2020. Dialogue coherence assessment without explicit dialogue act labels. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, ACL 2020, Online, July 5-10, 2020*, pages 1439–1450. Association for Computational Linguistics.

Jekaterina Novikova, Ondrej Dusek, Amanda Cercas Curry, and Verena Rieser. 2017. Why we need new evaluation metrics for NLG. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing, EMNLP 2017, Copenhagen, Denmark, September 9-11, 2017*, pages 2241–2252. Association for Computational Linguistics.

Bo Pang, Erik Nijkamp, Wenjuan Han, Linqi Zhou, Yixian Liu, and Kewei Tu. 2020. Towards holistic and automatic evaluation of open-domain dialogue generation. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, ACL 2020, Online, July 5-10, 2020*, pages 3619–3629. Association for Computational Linguistics.

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. Bleu: a method for automatic evaluation of machine translation. In *ACL*, pages 311–318. ACL.

Hannah Rashkin, Eric Michael Smith, Margaret Li, and Y-Lan Boureau. 2019. Towards empathetic open-domain conversation models: A new benchmark and dataset. In *Proceedings of the 57th Conference of the Association for Computational Linguistics, ACL 2019, Florence, Italy, July 28- August 2, 2019, Volume 1: Long Papers*, pages 5370–5381. Association for Computational Linguistics.

Ricardo Rei, Craig Stewart, Ana C. Farinha, and Alon Lavie. 2020. COMET: A neural framework for MT evaluation. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing, EMNLP 2020, Online, November 16-20, 2020*, pages 2685–2702. Association for Computational Linguistics.

Ananya B. Sai, Tanay Dixit, Dev Yashpal Sheth, Sreyas Mohan, and Mitesh M. Khapra. 2021. Perturbation checklists for evaluating NLG evaluation metrics. In *Proceedings of the 2021 Conference on Empirical*

Methods in Natural Language Processing, EMNLP 2021, Virtual Event / Punta Cana, Dominican Republic, 7-11 November, 2021, pages 7219–7234. Association for Computational Linguistics.

Ananya B. Sai, Akash Kumar Mohankumar, Siddhartha Arora, and Mitesh M. Khapra. 2020. Improving dialog evaluation with a multi-reference adversarial dataset and large scale pretraining. *Trans. Assoc. Comput. Linguistics*, 8:810–827.

Thibault Sellam, Dipanjan Das, and Ankur P. Parikh. 2020. BLEURT: learning robust metrics for text generation. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, ACL 2020, Online, July 5-10, 2020*, pages 7881–7892. Association for Computational Linguistics.

Heung-Yeung Shum, Xiaodong He, and Di Li. 2018. From eliza to xiaoice: challenges and opportunities with social chatbots. *Frontiers of IT & EE*, 19(1):10–26.

Chongyang Tao, Lili Mou, Dongyan Zhao, and Rui Yan. 2018. RUBER: an unsupervised method for automatic evaluation of open-domain dialog systems. In *AAAI*, pages 722–729. AAAI Press.

Xinnuo Xu, Ondrej Dusek, Ioannis Konstas, and Verena Rieser. 2018a. Better conversations by modeling, filtering, and optimizing for coherence and diversity. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing, Brussels, Belgium, October 31 - November 4, 2018*, pages 3981–3991. Association for Computational Linguistics.

Zhen Xu, Nan Jiang, Bingquan Liu, Wenge Rong, Bowen Wu, Baoxun Wang, Zhuoran Wang, and Xiaolong Wang. 2018b. LSDSCC: a large scale domain-specific conversational corpus for response generation with diversity oriented evaluation metrics. In *NAACL-HLT*, pages 2070–2080. Association for Computational Linguistics.

Rui Yan. 2018. "chitty-chitty-chat bot": Deep learning for conversational AI. In *Proceedings of the Twenty-Seventh International Joint Conference on Artificial Intelligence, IJCAI 2018, July 13-19, 2018, Stockholm, Sweden*, pages 5520–5526. ijcai.org.

Zheng Ye, Liucun Lu, Lishan Huang, Liang Lin, and Xiaodan Liang. 2021. Towards quantifiable dialogue coherence evaluation. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing, ACL/IJCNLP 2021, (Volume 1: Long Papers), Virtual Event, August 1-6, 2021*, pages 2718–2729. Association for Computational Linguistics.

Yi-Ting Yeh, Maxine Eskénazi, and Shikib Mehri. 2021. A comprehensive assessment of dialog evaluation metrics. *CoRR*, abs/2106.03706.

Weizhe Yuan, Graham Neubig, and Pengfei Liu. 2021. Bartscore: Evaluating generated text as text generation. In *Advances in Neural Information Processing Systems 34: Annual Conference on Neural Information Processing Systems 2021, NeurIPS 2021, December 6-14, 2021, virtual*, pages 27263–27277.

Jerrold H Zar. 2005. Spearman rank correlation. *Encyclopedia of biostatistics*, 7.

Chen Zhang, Yiming Chen, Luis Fernando D'Haro, Yan Zhang, Thomas Friedrichs, Grandee Lee, and Haizhou Li. 2021. Dynaeval: Unifying turn and dialogue level evaluation. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing, ACL/IJCNLP 2021, (Volume 1: Long Papers), Virtual Event, August 1-6, 2021*, pages 5676–5689. Association for Computational Linguistics.

Saizheng Zhang, Emily Dinan, Jack Urbanek, Arthur Szlam, Douwe Kiela, and Jason Weston. 2018. Personalizing dialogue agents: I have a dog, do you have pets too? In *ACL (1)*, pages 2204–2213. Association for Computational Linguistics.

Tianyi Zhang, Varsha Kishore, Felix Wu, Kilian Q. Weinberger, and Yoav Artzi. 2020a. Bertscore: Evaluating text generation with BERT. In *8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, April 26-30, 2020*. OpenReview.net.

Yizhe Zhang, Siqi Sun, Michel Galley, Yen-Chun Chen, Chris Brockett, Xiang Gao, Jianfeng Gao, Jingjing Liu, and Bill Dolan. 2020b. DIALOGPT : Large-scale generative pre-training for conversational response generation. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics: System Demonstrations, ACL 2020, Online, July 5-10, 2020*, pages 270–278. Association for Computational Linguistics.

Wei Zhao, Maxime Peyrard, Fei Liu, Yang Gao, Christian M. Meyer, and Steffen Eger. 2019. Moverscore: Text generation evaluating with contextualized embeddings and earth mover distance. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing, EMNLP-IJCNLP 2019, Hong Kong, China, November 3-7, 2019*, pages 563–578. Association for Computational Linguistics.

Open-Domain Dialog Evaluation using Follow-Ups Likelihood

Maxime De Bruyn, Ehsan Lotfi, Jeska Buhmann, Walter Daelemans
CLiPS Research Center
University of Antwerp, Belgium
maxime.debruyn@uantwerpen.be

Abstract

Automatic evaluation of open-domain dialogs remains an unsolved problem. Moreover, existing methods do not correlate strongly with human annotations. This paper presents a new automated evaluation method using follow-ups: we measure the probability that a language model will continue the conversation with a fixed set of follow-ups (e.g. *Not really relevant here*, *What are you trying to say?*). When compared against twelve existing methods, our new evaluation achieves the highest correlation with human evaluations.

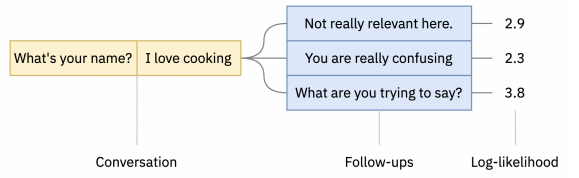

Figure 1: Illustration of our method. We measure the probability (log-likelihood) that a language model will continue the conversation with a set of predefined follow-ups. This paper shows that the sum of the individual log-likelihoods correlates strongly with human evaluations.

1 Introduction

Despite the recent progress in Natural Language Processing, the automatic evaluation of open-domain conversations remains an unsolved problem. It is difficult to establish criteria to measure the quality of a system. Task-oriented dialog systems use metrics such as task success or dialog efficiency. However, these do not apply to open-domain conversational agents (McTear, 2020).

Currently, there are two options for open-domain dialog evaluation: human evaluation and automated evaluation. Thanks to their understanding of natural language, humans are able to digest the entire dialog context in order to meaningfully evaluate a response (Mehri et al., 2022). Human evaluation also has its shortcomings: inconsistency in ratings (the same annotator may give two different scores depending on the mood), lack of reproducibility, and cost (Mehri et al., 2022).

The second option is to use automated evaluation metrics. Methods inherited from sequence-to-sequence machine translation such as BLEU (Papineni et al., 2002) evaluate the generated utterance by comparing it to the ground-truth. By doing so, these methods miss the one-to-many characteristic of conversation: a conversation may evolve in more than one valid direction.

To tackle this problem, researchers came up with reference-free evaluation metrics: the generated utterance is not compared to a ground truth but evaluated on its own.

FED (Mehri and Eskenazi, 2020a) is an unsupervised reference-free evaluation metric. It uses the idea that one can use the next utterance in a conversation to rate the turn before it. When users speak to a system, their response to a given system may implicitly provide feedback for the system. FED uses a set of predefined follow-ups and the log-likelihood from a language model to measure 18 fine-grained attributes in a conversation.

Inspired by the FED metric, we propose a new evaluation method called FULL (Follow-Up Log-Likelihood). We start by explaining our method and how it departs from the original FED metric. Next, we explain our choice of language model and follow-ups. Finally, we demonstrate that our new method achieves the highest correlation with human evaluations compared to 12 automated metrics. We open-source our evaluation code[1] and publish FULL as a Python package[2] for easy usage.

[1] https://github.com/maximedb/full
[2] https://pypi.org/project/full/

Proceedings of the 29th International Conference on Computational Linguistics, pages 496–504
October 12–17, 2022.

2 Related Work

This section reviews the existing literature on evaluation metrics for open-domain conversations. In the interest of space, we limit ourselves to studying reference-free methods (methods that do not require a ground truth). The interested reader is encouraged to read Yeh et al. (2021) for a full review.

GRADE (Huang et al., 2020) and DynaEval (Zhang et al., 2021) use a graph-based structure to model the dialog-level interaction between a user and a system. DynaEval distinguishes between well-formed dialogs from carefully constructed negative samples. MAUDE (Sinha et al., 2020) is also trained to distinguish a correct response from a randomly sampled negative response using a contrastive loss. FlowScore (Li et al., 2021) evaluates the quality of a dialog using the dynamic information flow in the dialog history.

USR (Mehri and Eskenazi, 2020b) trains several models to measure different qualities of dialogs. A masked language modeling head measures the fluency of the conversation, a retrieval model determines the relevance of a response, and a fact-to-response model checks whether a response conditions on knowledge. USL-H (Phy et al., 2020) also has three internal models, although they measure different attributes: grammatical correctness, sensibleness, and the likelihood of a given response. Other notable evaluation methods include Ghazarian et al. (2020); See and Manning (2021); Ghazarian et al. (2022b,a)

FED (Mehri and Eskenazi, 2020a) and HolisticEval (Pang et al., 2020) both use GPT-like (Radford et al., 2019) models to evaluate conversation on several attributes. FED computes the likelihood of manually designed follow-up utterances to measure multiple dialog qualities without supervision. HolisticEval uses a GPT-2 model to measure coherence, fluency, diversity, and consistency.

3 Method

Our metric FULL (Follow-Up Log-Likelihood) is a reference-free evaluation method for dialogs inspired by FED (Mehri and Eskenazi, 2020a). Figure 1 provides an overview.

3.1 Follow-Up Utterance for Evaluation

Our method uses follow-up utterances to evaluate the quality of a conversation (Eskénazi et al., 2019). When interacting with a system, users may provide implicit feedback about the conversation in the semantics of their response. For example, if a user ends a conversation with *It was a pleasure talking to you*, we can reasonably assume it was a pleasant conversation. On the other hand, if a user ends a conversation with *What are you talking about?*, we could conclude that the user is confused about the state of the conversation.

3.2 Log-Likelihood of Follow-Ups

We do not have access to the next utterance in an interactive setting. Instead, we ask a language model to play the role of a human. We ask the model how likely it is to generate a fixed set of follow-ups. For example, if the language model is likely to continue a conversation with the follow-up *I don't understand what you are saying*, we could conclude that the utterance generated by the system does not make sense.

FULL analyzes the quality of a response r in the context of a dialog history h with a language model M and a set of n predefined follow-ups F. For each predefined follow-up, the language model computes the log-likelihood D of a follow-up utterance f_i given the dialog history.

$$\sum_{i=1}^{n} D(h, r, f_i) \tag{1}$$

The total score is equal to the sum of the individual log likelihoods. It is worth reminding that the metric does not mean anything. It is only useful to *compare* systems together.

3.3 Differences with FED

Our implementation differs from FED (Mehri and Eskenazi, 2020a) in multiple ways. First, we do not consider fine-grained attributes, only the overall quality of the turn or dialog.[3]

Second, FED computes the log-likelihood of the conversation history h, the response r, and the follow-up f_i. Whereas we only compute the conditional log-likelihood of the follow-up f_i. Computing the log-likelihood over the conversation introduces a bias towards the dataset used in training the language model, Reddit, in the case of FED. It also

[3]Whereas FED considers 18 fine-grained attributes (overall quality included). Our initial experiments revealed that follow-ups assigned to a fine-grained attribute (e.g., engaging) often had a higher correlation with another unrelated attribute (e.g., correctness). For that reason, we choose to focus on a single attribute, the conversation's overall quality and leave the study of fine-grained attributes for future work.

favors longer conversations over shorter ones. Our goal is to estimate the likelihood of the follow-up, not the conversation itself.

Third, FED did not justify its choice of follow-ups, while we studied each candidate and only took the most correlated ones making intuitive sense. Fourth, we also study multiple types of language models (conversational and general).

4 Experimental Settings

This section explains our choices of follow-ups, language models, and conversational data. Our goal is to find the combination of language models and follow-ups correlating the most with human evaluations.

4.1 Follow-Ups

A follow-up is an utterance added after a conversation's last turn to evaluate the last turn or the entire dialog. FED defined 63 unique follow-ups in 16 categories (fine-grained attributes) at the turn level and the dialog level. Appendix B list the entire list of follow-ups. The authors did not provide any justification for their choice of follow-ups. Instead of blindly using the list of follow-ups, we attempt to understand which of these follow-ups have the highest correlation with human evaluations.

4.2 Language Models

We experiment with several language models, both general and conversational. The goal of the language module is to compute the conditional log-likelihood of several follow-ups.

BlenderBot v1 is a conversational sequence-to-sequence model (Roller et al., 2020) with three sizes: small, large, and extra-large. A distilled version is also available on HuggingFace.

DialoGPT is a conversational language model (Zhang et al., 2020) with three sizes: small, medium and large. The authors fine-tuned a GPT-2 model on a large corpus of Reddit conversations.

GPT-2 is a general language model (Radford et al., 2019). While it was not trained specifically on conversational data, our experiments revealed its potential to estimate a conversation's quality.

4.3 Conversational Data

We use the FED dataset (Mehri and Eskenazi, 2020a) for evaluating the set of follow-ups. It consists of 372 turn-level (124 dialog-level), originally collected by Adiwardana et al. (2020). The dataset consists of human-system conversations (Meena and Mitsuku) and human-human conversations. Mehri and Eskenazi (2020a) asked annotators to evaluate turn-level and dialog-level conversations on several attributes. In this work, we only use the evaluation of the overall quality of the turn or dialog.

Follow-up	Correlation	
	Turn	Dialog
Not really relevant here.	0.48	0.65
You're really confusing.	0.46	0.67
I don't understand what you're saying.	0.46	0.58
That's not really relevant here.	0.45	0.70
You are so confusing.	0.45	0.64
You're really boring.	0.44	0.65
That's not very interesting.	0.44	0.60
That was a really boring response.	0.43	0.63
You don't seem interested.	0.43	0.61
I am so confused right now.	0.43	0.57

Table 1: Top 10 follow-ups ranked by Spearman correlation to human evaluations. All follow-ups exhibit a positive relationship, meaning that the likely presence of the follow-up (low log-likelihood) entails a low human evaluation and vice-versa.

5 Results

Our objective is to find the best combination of language models and follow-ups. We start by analyzing which language model correlates the most with human evaluation. In the second step, we look for the best set of follow-ups.

5.1 Choice of Language Model

We are looking for a language model whose log-likelihood of generating the follow-ups correlates highly with human evaluations. We do so both on a turn-level and dialog-level. We compare the average absolute correlation of each follow-up with human judgments. The results are displayed on Figure 2 in Annex A. The model standing out is the large Blender model (Roller et al., 2020). It has the highest correlation with humans both on a turn-level and dialog-level. The difference in performance between Blender-3B and Blender-400M is small. For these reasons, we choose Blender-400M as our default language model.

5.2 Choice of Follow-ups

Now that we have identified our model of choice (Blender-400M), we wish to identify the follow-

	Turn Level	Dialog Level
QuestEval	0.09	0.08
MAUDE	-0.09	-0.28
DEB	0.19	-0.01
GRADE	0.12	-0.06
DynaEval	0.32	0.55
USR	0.12	0.06
USL-H	0.19	0.15
DialoRPT	-0.09	-0.21
HolisticEval	0.12	-0.30
PredictiveEngage	0.09	0.15
FED	0.09	0.32
FlowScore	-0.05	-0.00
FULL (ours)	**0.51**	**0.69**

Table 2: Comparison of our evaluation method FULL with other automated methods. FULL achieves the highest correlation on turn-level and dialog-level, followed by DynaEval. Except for FULL, results are copied from Yeh et al. (2021).

ups correlating the most with humans. We compute the Spearman correlation between each follow-up and human evaluation (turn-level and dialog-level). We present the top-10 follow-ups (by absolute correlation) in Table 1. The full table is available Appendix B.

The follow-up correlating the most on a turn-level basis is *Not really relevant here* with a Spearman correlation of 0.48. The least correlated follow-up is *Wow! That's really cool!* with correlations of 0.04. The follow-up correlating the most on a dialog-level basis is *That's not really relevant here* with a correlation of 0.70. The least correlated follow-up on a dialog level is *Cool! That sounds super interesting!* with a correlation of 0.01.

Most follow-ups exhibit a positive relationship, meaning that the likely generation of the follow-up by the language model (low log-likelihood) entails a low human rating and vice-versa. However, all the top follow-ups are *negative* follow-ups (e.g., *You're really confusing*), and their likely presence indicates a negative conversation. On the other hand, the *positive* follow-ups (e.g., *Great talking to you*) are not as highly correlated. On average, negative follow-ups correlate with 0.39, while positive follow-ups correlate with 0.24. These results indicate that the language model evaluates a good conversation by the likely absence of negative follow-ups.

Each follow-up brings another forward pass of the model, so ideally, we want to restrict the number of follow-ups in the final evaluation method. For the final selection of follow-ups, we combine the rank of the turn-level and dialog-level correlations and take the top 5.[4] The final selection of follow-ups is the following: *Not really relevant here. You're really confusing. You're really boring. What are you trying to say? You don't seem interested.*

5.3 Comparison

Yeh et al. (2021) compared 12 evaluation methods on the FED dataset (Mehri and Eskenazi, 2020a). We compare our method FULL against these 12 other methods in Table 2. The results are clear, FULL achieves the highest correlation both on a turn-level and dialog-level while being fully unsupervised (except in the choice of follow-ups). By combining the log-likelihood from 5 follow-ups, the average correlation on turn-level increases to 0.51, while the average of the individual correlation equals 0.45.

6 Conclusion

This short paper introduces a new automated evaluation method (FULL) for open-domain conversations. FULL measures the quality of a conversation by computing the probability that a language model will continue the conversation with a set of follow-ups (e.g., *Not really relevant here*, *What are you trying to say?*). FULL achieves the highest correlation with human evaluations compared to twelve other existing methods.

Our experiments revealed that negative follow-ups (e.g., *Not really relevant here*) have a higher correlation with human evaluations than positive follow-ups (e.g., *Wow, interesting to know*). It is easier for the model to evaluate a conversation from its bad angles rather than its good ones.

Future work is needed to know which fine-grained attribute can be measured using the same technique. Using ever-large models such as GPT-3 (Brown et al., 2020) or OPT (Zhang et al., 2022) could be a direction for future research, although the resulting model will likely need to be distilled to be of practical use.

[4] We arbitrarily choose the number 5. We also removed close duplicates. For example *Not really relevant here.* and *That's not really relevant here.*

Acknowledgement

We thank the reviewers for their helpful feedback. This research received funding from the Flemish Government under the *Onderzoeksprogramma Artificiële Intelligentie (AI) Vlaanderen* programme.

References

Daniel Adiwardana, Minh-Thang Luong, David R. So, Jamie Hall, Noah Fiedel, Romal Thoppilan, Zi Yang, Apoorv Kulshreshtha, Gaurav Nemade, Yifeng Lu, and Quoc V. Le. 2020. Towards a human-like open-domain chatbot. *CoRR*, abs/2001.09977.

Tom B. Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel M. Ziegler, Jeffrey Wu, Clemens Winter, Christopher Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. 2020. Language models are few-shot learners. *CoRR*, abs/2005.14165.

Maxine Eskénazi, Shikib Mehri, Evgeniia Razumovskaia, and Tiancheng Zhao. 2019. Beyond turing: Intelligent agents centered on the user. *CoRR*, abs/1901.06613.

Sarik Ghazarian, Behnam Hedayatnia, Alexandros Papangelis, Yang Liu, and Dilek Hakkani-Tur. 2022a. What is wrong with you?: Leveraging user sentiment for automatic dialog evaluation. *arXiv*, abs/2203.13927.

Sarik Ghazarian, Ralph Weischedel, Aram Galstyan, and Nanyun Peng. 2020. Predictive engagement: An efficient metric for automatic evaluation of open-domain dialogue systems. *Proceedings of the AAAI Conference on Artificial Intelligence*, 34(05):7789–7796.

Sarik Ghazarian, Nuan Wen, Aram Galstyan, and Nanyun Peng. 2022b. Deam: Dialogue coherence evaluation using amr-based semantic manipulations. *arXiv*, abs/2203.09711.

Lishan Huang, Zheng Ye, Jinghui Qin, Liang Lin, and Xiaodan Liang. 2020. GRADE: Automatic graph-enhanced coherence metric for evaluating open-domain dialogue systems. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 9230–9240, Online. Association for Computational Linguistics.

Zekang Li, Jinchao Zhang, Zhengcong Fei, Yang Feng, and Jie Zhou. 2021. Conversations are not flat: Modeling the dynamic information flow across dialogue utterances. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 128–138, Online. Association for Computational Linguistics.

Michael McTear. 2020. Conversational ai: dialogue systems, conversational agents, and chatbots. *Synthesis Lectures on Human Language Technologies*, 13(3):1–251.

Shikib Mehri, Jinho Choi, Luis Fernando D'Haro, Jan Deriu, Maxine Eskenazi, Milica Gasic, Kallirroi Georgila, Dilek Hakkani-Tur, Zekang Li, Verena Rieser, Samira Shaikh, David Traum, Yi-Ting Yeh, Zhou Yu, Yizhe Zhang, and Chen Zhang. 2022. Report from the nsf future directions workshop on automatic evaluation of dialog: Research directions and challenges. *arXiv*, abs/2203.10012.

Shikib Mehri and Maxine Eskenazi. 2020a. Unsupervised evaluation of interactive dialog with DialoGPT. In *Proceedings of the 21th Annual Meeting of the Special Interest Group on Discourse and Dialogue*, pages 225–235, 1st virtual meeting. Association for Computational Linguistics.

Shikib Mehri and Maxine Eskenazi. 2020b. USR: An unsupervised and reference free evaluation metric for dialog generation. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 681–707, Online. Association for Computational Linguistics.

Bo Pang, Erik Nijkamp, Wenjuan Han, Linqi Zhou, Yixian Liu, and Kewei Tu. 2020. Towards holistic and automatic evaluation of open-domain dialogue generation. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 3619–3629, Online. Association for Computational Linguistics.

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. Bleu: a method for automatic evaluation of machine translation. In *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics*, pages 311–318, Philadelphia, Pennsylvania, USA. Association for Computational Linguistics.

Vitou Phy, Yang Zhao, and Akiko Aizawa. 2020. Deconstruct to reconstruct a configurable evaluation metric for open-domain dialogue systems. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 4164–4178, Barcelona, Spain (Online). International Committee on Computational Linguistics.

Alec Radford, Jeff Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. 2019. Language models are unsupervised multitask learners.

Stephen Roller, Emily Dinan, Naman Goyal, Da Ju, Mary Williamson, Yinhan Liu, Jing Xu, Myle Ott, Kurt Shuster, Eric Michael Smith, Y-Lan Boureau, and Jason Weston. 2020. Recipes for building an open-domain chatbot. *CoRR*, abs/2004.13637.

Abigail See and Christopher Manning. 2021. Understanding and predicting user dissatisfaction in a neural generative chatbot. In *Proceedings of the 22nd Annual Meeting of the Special Interest Group on Discourse and Dialogue*, pages 1–12, Singapore and Online. Association for Computational Linguistics.

Koustuv Sinha, Prasanna Parthasarathi, Jasmine Wang, Ryan Lowe, William L. Hamilton, and Joelle Pineau. 2020. Learning an unreferenced metric for online dialogue evaluation. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 2430–2441, Online. Association for Computational Linguistics.

Yi-Ting Yeh, Maxine Eskenazi, and Shikib Mehri. 2021. A comprehensive assessment of dialog evaluation metrics. In *The First Workshop on Evaluations and Assessments of Neural Conversation Systems*, pages 15–33, Online. Association for Computational Linguistics.

Chen Zhang, Yiming Chen, Luis Fernando D'Haro, Yan Zhang, Thomas Friedrichs, Grandee Lee, and Haizhou Li. 2021. DynaEval: Unifying turn and dialogue level evaluation. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 5676–5689, Online. Association for Computational Linguistics.

Susan Zhang, Stephen Roller, Naman Goyal, Mikel Artetxe, Moya Chen, Shuohui Chen, Christopher Dewan, Mona Diab, Xian Li, Xi Victoria Lin, Todor Mihaylov, Myle Ott, Sam Shleifer, Kurt Shuster, Daniel Simig, Punit Singh Koura, Anjali Sridhar, Tianlu Wang, and Luke Zettlemoyer. 2022. Opt: Open pre-trained transformer language models. *arXiv*, abs/2205.01068.

Yizhe Zhang, Siqi Sun, Michel Galley, Yen-Chun Chen, Chris Brockett, Xiang Gao, Jianfeng Gao, Jingjing Liu, and Bill Dolan. 2020. DIALOGPT : Large-scale generative pre-training for conversational response generation. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics: System Demonstrations*, pages 270–278, Online. Association for Computational Linguistics.

A Appendix: Comparison of Models

We present in Figure 2 the average absolute correlation to human evaluations per model.

B Appendix: List of Candidate Follow-ups

Table 3 list the entire list of follow-ups considered.

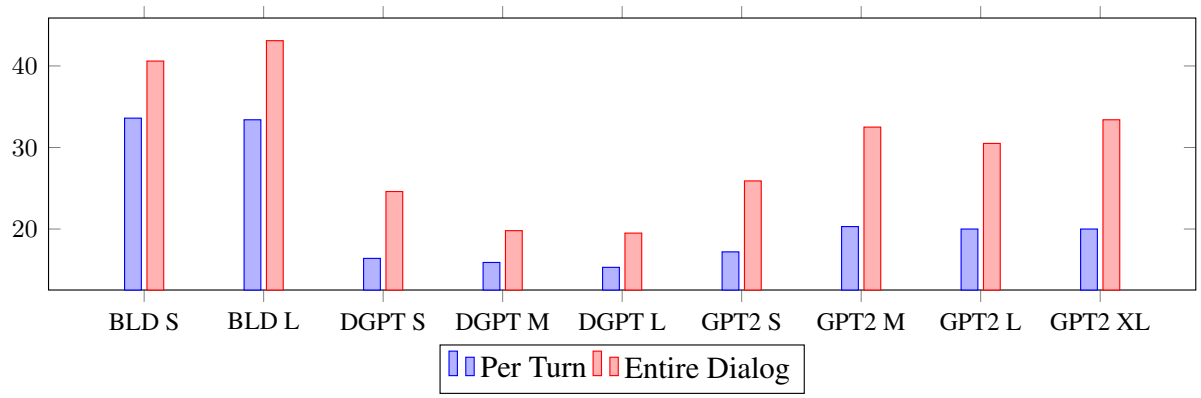

Figure 2: Average absolute correlation with human evaluations for several language models. We use Blender-400 (BLD S) as language model because of its high correlation with human evaluations. For space reasons, Blender is abbreviated as BLD and DialoGPT as DGPT.

	Follow-up	Category	Level	Type	Level	Dialog
X	Not really relevant here.	specific	turn	neg	0.48	0.65
X	You're really confusing.	error recovery	dialog	neg	0.46	0.67
	I don't understand what you're saying.	correct	turn	neg	0.46	0.58
	That's not really relevant here.	specific	turn	neg	0.45	0.70
	You are so confusing.	coherent	dialog	neg	0.45	0.64
X	You're really boring.	informative	dialog	neg	0.44	0.65
	That's not very interesting.	interesting	turn	neg	0.44	0.60
	That was a really boring response.	interesting	turn	neg	0.43	0.63
X	You don't seem interested.	inquisitive	dialog	neg	0.43	0.61
	I am so confused right now.	error recovery	dialog	neg	0.43	0.60
	I'm so confused!	understandable	turn	neg	0.43	0.59
	I don't really care. That's pretty boring.	engaging	turn	neg	0.43	0.61
	I want to talk about something else.	engaging	turn	neg	0.43	0.65
	That's not even related to what I said.	relevant	turn	neg	0.42	0.58
X	What are you trying to say?	understanding	dialog	neg	0.42	0.68
	I am so confused right now!	correct	turn	neg	0.42	0.57
	That makes no sense!	semantically appropriate	turn	neg	0.42	0.56
	I don't understand at all!	understandable	turn	neg	0.41	0.54
	That's really boring.	interesting	turn	neg	0.41	0.54
	I don't like you.	likeable	dialog	neg	0.40	0.58
	I'm so confused right now!	fluent	turn	neg	0.40	0.56
	Don't change the topic!	relevant	turn	neg	0.40	0.58
	You're not understanding me!	correct	turn	neg	0.40	0.62
	That's a very generic response.	specific	turn	neg	0.39	0.50
	You don't really know much.	informative	dialog	neg	0.39	0.52
	You're not very nice.	likeable	dialog	neg	0.38	0.56
	You're not very fun to talk to.	likeable	dialog	neg	0.37	0.55
	Is that real English?	fluent	turn	neg	0.37	0.49
	That's a lot of questions!	inquisitive	dialog	pos	0.36	0.52
	Why are you repeating yourself?	diverse	dialog	neg	0.35	0.50
	You're making no sense at all.	coherent	dialog	neg	0.35	0.43
	You ask a lot of questions!	inquisitive	dialog	pos	0.35	0.54
	Let's change the topic.	engaging	turn	neg	0.35	0.45
	You don't ask many questions.	inquisitive	dialog	neg	0.35	0.54
	Why are you changing the topic?	relevant	turn	neg	0.34	0.51
	Stop saying the same thing repeatedly.	diverse	dialog	neg	0.34	0.50
	Do you know how to talk about something else?	flexible	dialog	neg	0.33	0.49
	You're changing the topic so much!	coherent	dialog	neg	0.33	0.47
	You know a lot of facts!	informative	dialog	pos	0.32	0.48
	Tell me more!	engaging	turn	pos	0.32	0.34
	I like you!	likeable	dialog	pos	0.31	0.43
	Wow that's a lot of information.	informative	dialog	pos	0.31	0.38
	Stop changing the topic so much.	depth	dialog	neg	0.31	0.44
	What does that even mean?	understandable	turn	neg	0.30	0.35
	I don't want to talk about that!	flexible	dialog	neg	0.29	0.50

Table 3 continued from previous page

Follow-up	Category	Level	Type	Level	Dialog
That's not what you said earlier!	consistent	dialog	neg	0.29	0.37
You have a good point.	semantically appropriate	turn	pos	0.29	0.43
I see, that's interesting.	specific	turn	pos	0.28	0.31
Stop contradicting yourself!	consistent	dialog	neg	0.28	0.36
You're very easy to talk to!	flexible	dialog	pos	0.28	0.40
Stop repeating yourself!	diverse	dialog	neg	0.27	0.40
That's good to know. Cool!	specific	turn	pos	0.25	0.30
That's a good point.	specific	turn	pos	0.25	0.34
Wow you can talk about a lot of things!	flexible	dialog	pos	0.23	0.27
I'm really interested in learning more about this.	engaging	turn	pos	0.22	0.26
That makes sense!	semantically appropriate	turn	pos	0.21	0.21
Thanks for all the information!	informative	dialog	pos	0.21	0.15
You're super polite and fun to talk to	likeable	dialog	pos	0.17	0.23
Wow that is really interesting.	interesting	turn	pos	0.17	0.14
That's really interesting!	interesting	turn	pos	0.16	0.11
Great talking to you.	likeable	dialog	pos	0.15	0.10
Cool! That sounds super interesting.	interesting	turn	pos	0.08	- 0.01
Wow! That's really cool!	engaging	turn	pos	0.04	- 0.08

Table 3: List of candidate follow-ups along with their category (fine-grained attribute), positivity (negative of positive follow-up) and correlation with a human evaluation of the overall quality of the turn/dialog. All follow-ups and static data is from Mehri and Eskenazi (2020a).

Joint Goal Segmentation and Goal Success Prediction on Multi-Domain Conversations

Meiguo Wang **Benjamin Yao** **Bin Guo**
Xiaohu Liu **Yu Zhang** **Tuan-Hung Pham** **Chenlei Guo**
Amazon Alexa AI

Abstract

To evaluate the performance of a multi-domain goal-oriented Dialogue System (DS), it is important to understand what the users' goals are for the conversations and whether those goals are successfully achieved. The success rate of goals directly correlates with user satisfaction and perceived usefulness of the DS. In this paper, we propose a novel automatic dialogue evaluation framework that jointly performs two tasks: goal segmentation and goal success prediction. We extend the RoBERTa-IQ model (Gupta et al., 2021) by adding multi-task learning heads for goal segmentation and success prediction. Using an annotated dataset from a commercial DS, we demonstrate that our proposed model reaches an accuracy that is on-par with single-pass human annotation comparing to a three-pass gold annotation benchmark.

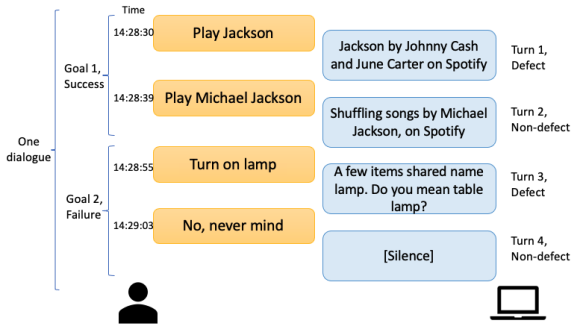

Figure 1: A dialogue that contains two goals: play-music and turn-on-device. Here, a turn is defined as one back-and-forth between the user and agent. A dialogue is a set of turns in quick succession as indicated by the timestamps. A goal represents an objective the user is trying to accomplish in a sequence of turns. Human annotation labels for turns (defect/non-defect) and goals (success/failure) are provided for reference.

1 Introduction

Today, commercial conversational AI assistants (e.g., Amazon Alexa, Apple Siri, and Google Assistant) are increasingly popular. However, it is challenging to establish reliable metrics to continuously measure the system performance in business reports and A/B experiments. The commonly used metrics in industry, e.g., monthly active users, dialogue count per user, and downstream impacts such as increased subscriptions and product sales etc., usually move slowly and are not suitable to measure the impact of functionality changes. On the other hand, the dissatisfaction metrics based on manual annotation, which can capture dissatisfying user experiences due to system errors, incomplete service coverage, or poor response quality etc., are sensitive to functionality changes, but not suitable for online monitoring and experimentation due to their offline nature. Automatic dialogue evaluation metrics (Schmitt and Ultes, 2015; Ling et al., 2020) aim to combine the benefits of the two types of metrics mentioned above by providing an online metric that measures user experience and is sensitive to functionality changes. The effectiveness of such metrics depends on how well they are correlated with the human annotations.

Automated dialogue evaluation metrics in the literature could be generally grouped into two categories (Deriu et al., 2020): 1) Interaction Quality (IQ) metrics (e.g., (Schmitt and Ultes, 2015; Ling et al., 2020; Gupta et al., 2021)) that measures user experiences at turn level; and 2) Dialogue Quality (DQ) metrics (e.g., (Sun et al., 2021)) that measures user experiences for the whole dialogue. However, we argue that, for multi-domain goal-oriented dialogue systems, there is a need to add a goal-level metric with granularity in between turn and dialogue. If we define a dialogue as a set of turns in quick succession, a user may switch her objective mid-dialogue without clear delineations. As shown by the example in Figure 1, the user started the dialogue with a playing music request but switched her objective halfway to turn on a smarthome device. Here, we propose a new granularity – goal, which

Proceedings of the 29th International Conference on Computational Linguistics, pages 505–509
October 12–17, 2022.

represents an objective the user is trying to accomplish in a sequence of turns. Goal segmentation is especially important for accurately evaluating the dialogue quality. In Figure 1 example, the user successfully achieved her first goal but failed on the second one. It would be ambiguous to evaluate the user experience at the whole dialogue level. To our best knowledge, there is no other work in the literature that performs goal segmentation and goal success prediction jointly. Prior work either assumes that a dialogue automatically terminates upon the completion of a goal e.g., (Bodigutla et al., 2020) or the goal boundaries are known beforehand e.g., (Walker et al., 1997).

To build a reliable automatic goal evaluation metric, we extend the RoBERTa-IQ model (Gupta et al., 2021) by adding multi-task learning heads for goal segmentation and success prediction. Even though RoBERTa-IQ model is a model built for turn level metrics, it encodes the turns before and after the reference turn (the turn on which the model will make prediction) to ingest dialogue context information. Besides, time difference between turns are encoded in the time bin embeddings and feed into the model together with the context embedding. Thus, such characteristics makes RoBERTa-IQ model a suitable starting point for goal level evaluation task. For goal segmentation, different from LSTM based method (Koshorek et al., 2018; Arnold et al., 2019), we leverage self-attention mechanism across turns. The practical benefit of this approach is that it improves inference efficiency on long token sequences, which is a common problem for RNN models. To demonstrate the effectiveness of our proposed methods, we evaluated our method against a golden dataset in which each dialogue is labeled by three annotators and the ground truth labels are determined by majority vote. We then compare the accuracy of the model predictions with that of a single-pass human annotation.

The proposed work has two main contributions: 1) A novel goal level dialogue evaluation framework that matches the real-world scenarios in multi-domain goal-oriented dialogue system; 2) A deep learning model that jointly learns goal segmentation and goal success prediction, with accuracy on-par with single-pass human annotation. The remainder of this paper is organized as the following: Section 2 reviews related work about goal segmentation and dialogue evaluation. Section 3 introduces the model architecture. Section 4 shows the experimental setup and discusses the performance of the model. We conclude the paper in Section 5.

2 Related Work

2.1 Topic Segmentation

To our best knowledge, our study is the first one to propose dialogue goal segmentation. Similar to the topic segmentation for generic text, dialogue goal segmentation aims to segment a dialogue into the goal-coherent units. Therefore, the previous approaches, which were originally proposed for generic text topic segmentation, are ready to be used for conversational corpora. Early topic segmentation approaches can be classified into two types: 1) lexical cohesion models and 2) content-oriented models. A well-known algorithm of lexical cohesion models is TextTiling (Hearst, 1997). Content-oriented models rely on the re-occurrence of patterns of topics, such as Bayesian Unsupervised Topic Segmentation (Eisenstein and Barzilay, 2008). More recently, neural network-based approaches (Koshorek et al., 2018; Arnold et al., 2019) are favored by researchers because of robust model performance and efficiency.

2.2 Dialogue Evaluation

Unsupervised methods There are some unsupervised evaluation methods, which provide a good assessment for open domain dialogues. RUBER (Tao et al., 2018) is a turn level metric that combines the relatedness between the turn level response and the previous issued query-response, respectively. Besides relying on semantics of the sentences, GRADE (Huang et al., 2020) proposes a method to leverage the graph embedded topic-level representation for turn level success evaluation. For dialogue level evaluation, MAUDE (Sinha et al., 2020) is a context aware model that measures the quality of a generated reply given the previous dialogue context.

Supervised methods The well known evaluation framework (Walker et al., 1997) based on user satisfaction is PARADISE (PARAdigm for DIalog System Evaluation) framework. In PARADISE, given a set of manually extracted input features and user ratings, a linear regression model is fitted to predict the user satisfaction. In contrast to evaluating the entire dialogue, there are various

approaches to evaluate the user satisfaction at turn level, such as Interaction Quality (IQ) (Schmitt and Ultes, 2015). To further generalized the model, IQ-NET (Ling et al., 2020) directly uses raw dialogue turn contents and system metadata without hand-crafted features. Meanwhile, by using post experience explicit user feedback as a proxy to user satisfaction, several joint turn and dialogue level evaluation methods (Bodigutla et al., 2020; Park et al., 2020) are proposed.

3 Methodology

We extend the RoBERTa-IQ (Gupta et al., 2021) model architecture by adding the goal segmentation task and goal success prediction task. As a brief recap, there are three key differences separating RoBERTa-IQ from the vanilla RoBERTa model, as illustrated in Figure 2:

1. RoBERTa-IQ introduces two special tokens [USER] and [AGENT] to delineate the beginning of each user request and agent response. This enables the model to encode the multi-turn dialogue as a flattened token sequence (see Table 1 for an example), which enables self-attention mechanism across different turns;

2. RoBERTa-IQ has a notion of "reference turn" versus "contextual turns". The reference turn is the target for the model prediction while contextual turns are context surrounding the reference turn. In this paper, we use two turns before and two after the reference turn as context (ablation study showed no statistical difference with larger context window);

3. Instead of the usual "position embedding" for RoBERTa model, RoBERTa-IQ has an embedding called "time bin embedding" that is calculated based on timestamp difference between a context turn and the reference turn. The time bin embedding has two uses: 1) it allows the model to understand the temporal relationship between the reference turn and contextual turns; 2) it enables the model to locate the reference turn – a special bin: BIN_0 is reserved for reference turn's tokens.

We add the goal segmentation and goal success prediction task heads on top of the [CLS] token embedding. For each turn in a dialogue, we mark it as the reference turn and let the model to learn from a binary classification label (B if the reference turn is at the beginning of a goal and I if inside a goal). Goal success prediction is modeled as a multi-class classifier with three possible outputs: success, failure, and unactionable.

[USER] Play Jackson
[AGENT] Jackson by Johnny Cash and June Carter on Spotify
[USER] Play Michael Jackson
[AGENT] Shuffling songs by Michael Jackson, on Spotify
[USER] turn on lamp
[AGENT] A few items shared name lamp. Do you mean table lamp?
[USER] No, never mind
[AGENT] {Silence}

Table 1: The example dialogue in Figure 1 in flattened token sequences form, considering turn-2 as the reference turn (bolded).

4 Experiments

4.1 Dataset

We used de-identified data for our experiments. The data is labeled by one annotator for two goal level tasks - goal segmentation and goal success prediction. The training dataset contains ~500K dialogues, randomly split into training (80%) and validation (20%) sets. 44% of those dialogues are single-turn dialogues. For all goals identified in the training dataset: 75% are success, 14% are unactionable and 11% are failure. An additional three-pass human annotation is applied to the evaluation dataset and the majority vote is used as golden labels. We call the evaluation dataset with golden label as golden dataset. The golden dataset is used to measure the performance of both human and the trained models. The single-pass annotation is considered as prediction of human. The golden dataset, which contains ~30K dialogues, has similar data distribution as the training dataset.

4.2 Experiment Setup

Implementation Details The model training and evaluation are implemented in PyTorch. We continue the training for 15 epochs and select the best model based on the performance on the validation dataset. For multi-task model, we add the two loss functions (cross-entropy loss) for both tasks with equal weights (i.e. 0.5). To compare with the model trained in the multi-task learning framework, we also train goal segmentation and goal success prediction models separately on the same dataset. Note, the model prediction score on different turns may be different. The goal success prediction is

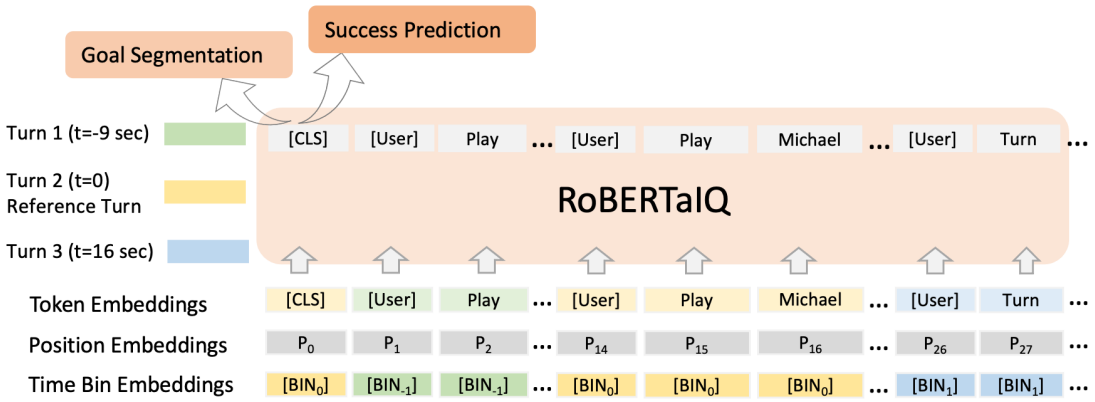

Figure 2: Diagram for RoBERTa-IQ model. The second turn with time bin equals to BIN_0 is the reference turn on which the model will make the prediction.

determined by the prediction on the last turn of the predicted goal.

Metrics We compare the performance of three models under different settings: human model (one-pass human annotation), single task model (two tasks trained separately), multi-task model (two tasks jointly trained). We measure model performance with the following metrics: 1) the accuracy of goal segmentation, which is defined as number of goals with correct boundaries divided by total number of goals; 2) the accuracy of goal evaluation, which is defined as the number of goals with correct boundaries and the right success prediction divided by the total number of goals; 3) weighted F1 score, which is the weighted average of the F1 score of each success prediction class on goals with accurate goal boundary. For machine-learned models, we also report relative metric (accuracy or weighted F1 score) with respect to that of human model in order to illustrate the difference. For example, the relative accuracy of human model is zero. The relative accuracy of single task model is computed as its accuracy minus the accuracy of human model.

4.3 Results

Table 2 summarizes the relative accuracy of segmentation and goal evaluation and relative weighted F1 score on single turn dialogues, multi-turn dialogues and these two combined. From the results, we can see the following points: 1) As shown in both accuracy and weighted F1 score, the performance of the multi-task model (the proposed model) is better than two single task models combined, especially, in multi-turn dialogues. 2) The proposed model has lower accuracy in goal segmentation but higher accuracy in goal evaluation compared to human. 3) The proposed model has higher accuracy but lower weighted F1 compared to human in goal evaluation. The proposed model is optimized for success class since that is the main usecase and has the most data for training while the proposed model has small performance gaps in failure class compared to human due to insufficient training data issue.

Dialogue	Model	Segmentation Accuracy	Goal Accuracy	Weighted F1 Score
Single turn	Human	+0.0%	+0.0%	+0.0%
	Single task	+0.0%	+1.1%	-1.3%
	Multi-task	+0.0%	**+1.4%**	**-1.0%**
Multi-turn	Human	+0.0%	+0.0%	+0.0%
	Single task	-4.7%	+2.2 %	-1.5%
	Multi-task	**-4.4%**	**+2.7%**	**-0.9%**
All	Human	+0.0%	+0.0%	+0.0%
	Single task	-3.8%	+2.0%	-1.4%
	Multi-task	**-3.5%**	**+2.4%**	**-0.9%**

Table 2: Performance of goal segmentation and goal evaluation on golden dataset. The best machine-learned model results are bolded.

5 Conclusion

In this paper, we propose a novel framework to evaluate goal-level performance for multi-domain goal-oriented dialogue systems and a deep learning model that jointly learns goal segmentation and success prediction. Our experiments show that the proposed model reaches an accuracy that is on-par with single-pass human annotation and with multi-task learning, the model performance is better than single task models for both tasks.

References

Sebastian Arnold, Rudolf Schneider, Philippe Cudré-Mauroux, Felix A. Gers, and Alexander Löser. 2019. SECTOR: A Neural Model for Coherent Topic Segmentation and Classification. *Transactions of the Association for Computational Linguistics*, 7:169–184.

Praveen Kumar Bodigutla, Aditya Tiwari, Josep Valls Vargas, Lazaros Polymenakos, and Spyros Matsoukas. 2020. Joint Turn and Dialogue level User Satisfaction Estimation on Multi-Domain Conversations. *arXiv e-prints*, page arXiv:2010.02495.

Jan Deriu, Alvaro Rodrigo, Arantxa Otegi, Guillermo Echegoyen, Sophie Rosset, Eneko Agirre, and Mark Cieliebak. 2020. Survey on evaluation methods for dialogue systems. *Artificial Intelligence Review*, 54(1):755–810.

Jacob Eisenstein and Regina Barzilay. 2008. Bayesian unsupervised topic segmentation. In *Proceedings of the 2008 Conference on Empirical Methods in Natural Language Processing*, pages 334–343, Honolulu, Hawaii. Association for Computational Linguistics.

Saurabh Gupta, Xing Fan, Derek Liu, Benjamin Yao, Yuan Ling, Kun Zhou, Tuan-Hung Pham, and Edward Guo. 2021. Robertaiq: An efficient framework for automatic interaction quality estimation of dialogue systems. In *KDD 2021 Workshop on Data-Efficient Machine Learning*.

Marti A. Hearst. 1997. Texttiling: Segmenting text into multi-paragraph subtopic passages. *Computational Linguistics*, pages 33–64.

Lishan Huang, Zheng Ye, Jinghui Qin, Liang Lin, and Xiaodan Liang. 2020. Grade: Automatic graph-enhanced coherence metric for evaluating open-domain dialogue systems. In *EMNLP*.

Omri Koshorek, Adir Cohen, Noam Mor, Michael Rotman, and Jonathan Berant. 2018. Text segmentation as a supervised learning task. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 2 (Short Papers)*, pages 469–473, New Orleans, Louisiana. Association for Computational Linguistics.

Yuan Ling, Benjamin Yao, Guneet Kohli, Tuan-Hung Pham, and Chenlei Guo. 2020. IQ-Net: A DNN model for estimating interaction-level dialogue quality with conversational agents. In *Proceedings of the KDD 2020 Workshop on Conversational Systems Towards Mainstream Adoption co-located with the 26TH ACM SIGKDD Conference on Knowledge Discovery and Data Mining (SIGKDD 2020), Virtual Workshop, August 24, 2020*, volume 2666 of *CEUR Workshop Proceedings*. CEUR-WS.org.

Dookun Park, Hao Yuan, Dongmin Kim, Yinglei Zhang, Spyros Matsoukas, Young-Bum Kim, Ruhi Sarikaya, Chenlei Guo, Yuan Ling, Kevin Quinn, Tuan-Hung Pham, Benjamin Yao, and Sungjin Lee. 2020. Large-scale hybrid approach for predicting user satisfaction with conversational agents. In *NeurIPS 2020: Human in the Loop Dialogue Systems Workshop*.

Alexander Schmitt and Stefan Ultes. 2015. Interaction quality: Assessing the quality of ongoing spoken dialog interaction by experts - and how it relates to user satisfaction. *Speech Commun.*, 74:12–36.

Koustuv Sinha, Prasanna Parthasarathi, Jasmine Wang, Ryan J. Lowe, William L. Hamilton, and Joelle Pineau. 2020. Learning an unreferenced metric for online dialogue evaluation. *ArXiv*, abs/2005.00583.

Weiwei Sun, Shuo Zhang, Krisztian Balog, Zhaochun Ren, Pengjie Ren, Zhumin Chen, and M. de Rijke. 2021. Simulating user satisfaction for the evaluation of task-oriented dialogue systems. *Proceedings of the 44th International ACM SIGIR Conference on Research and Development in Information Retrieval*.

Chongyang Tao, Lili Mou, Dongyan Zhao, and Rui Yan. 2018. Ruber: An unsupervised method for automatic evaluation of open-domain dialog systems. In *AAAI*.

Marilyn A. Walker, Diane J. Litman, Candace A. Kamm, and Alicia Abella. 1997. PARADISE: A framework for evaluating spoken dialogue agents. In *35th Annual Meeting of the Association for Computational Linguistics and 8th Conference of the European Chapter of the Association for Computational Linguistics*, pages 271–280, Madrid, Spain. Association for Computational Linguistics.

Slot Dependency Modeling for Zero-Shot Cross-Domain Dialogue State Tracking

Qingyue Wang[1,2], **Yanan Cao**[1]*, **Piji Li**[3], **Yanhe Fu**[1,2], **Zheng Lin**[1] and **Li Guo**[1]
1. Institute of Information Engineering, Chinese Academy of Sciences, Beijing, China
2. School of Cyber Security, University of Chinese Academy of Sciences, Beijing, China
3. College of Computer Science and Technology,
Nanjing University of Aeronautics and Astronautics, Nanjing, China
{wangqingyue,caoyanan,fuyanhe,linzheng,guoli}@iie.ac.cn
lipiji.pz@gmail.com

Abstract

Zero-shot learning for Dialogue State Tracking (DST) focuses on generalizing to an unseen domain without the expense of collecting in-domain data. However, previous zero-shot DST methods ignore the slot dependencies in a multi-domain dialogue, resulting in sub-optimal performances when adapting to unseen domains. In this paper, we utilize slot prompts combination, slot values demonstration, and slot constraint object to model the slot-slot dependency, slot-value dependency and slot-context dependency respectively. Specifically, each slot prompt consists of a slot-specific prompt and a slot-shared prompt to capture the shared knowledge across different domains. Experimental results show the effectiveness of our proposed method over existing state-of-art generation methods under zero-shot/few-shot settings.

1 Introduction

Task-oriented dialog systems help users to achieve specific goals using natural languages, such as movie booking and information support. Dialogue state tracking(DST), as a core component of task-oriented dialogue systems, tracks the user's requirements as dialogue states, which are typically in the form of a list of slot-value pairs. In practical applications, the multi-turn conversation usually refers to multiple domains. As shown in Figure 1, a user starts the conversation by asking a hotel and then requests a restaurant with a cheap price range, where *hotel* and *restaurant* are two different domains. At the third turn, the DST extracts multiple (slot, value) pairs like "(hotel-star, 4)" and "(restaurant-pricerange, cheap)" from the dialogue context.

In industrial applications, task-oriented dialogue systems are required to add new domains frequently based on users' needs, but collecting extensive data for every new domain is costly and inefficient.

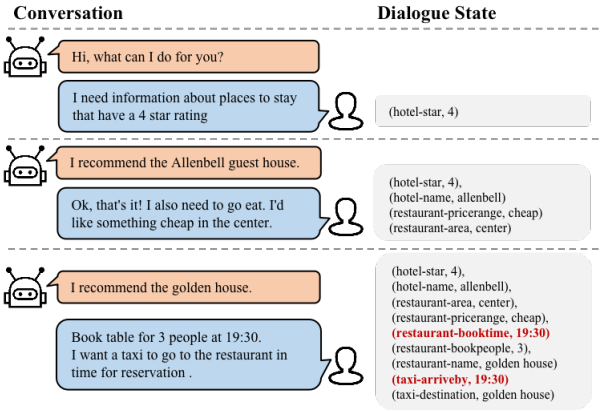

Figure 1: An multi-domain dialogue from MultiWOZ dataset (Budzianowski et al., 2018). Following the convention of this dataset, each slot is represented as a special token concatenated by domain and slot (e.g.,"restaurant-food").

Therefore, performing **zero-shot prediction** of dialogue states is becoming increasingly important since it does not require the expense of data acquisition.

The early works utilized the copy mechanism to handle new slot types in the unseen domain (Wu et al., 2019; Kumar et al., 2020). But the specialized models don't fully leverage the pre-trained language models(PLMs), which have shown impressive ability in transfer learning. Recently, a new paradigm named "prompt-based learning" utilizes language prompts to stimulate the knowledge of PLMs (Han et al., 2021). Compared to task-oriented fine-tuning, prompt-based learning is more similar to pre-training in terms of objectives, thereby adapting to downstream tasks faster even without any training samples. Inspired by it, some researchers add slot-specific prompt[1] into the sequence-to-sequence based model, achieving good performances in zero-shot DST(Lee et al., 2021; Su et al., 2021).

*Corresponding author.

[1]For example, the prompt of slot "restaurant-area" can be "what is the location of the restaurant?".

Proceedings of the 29th International Conference on Computational Linguistics, pages 510–520
October 12–17, 2022.

However, these approaches treat each slot independently, which ignore various slot dependencies during dialogue state tracking. We conceive that there exist several types of slot dependencies in multi-domain DST. For instance, the stars of a hotel and its price range often co-occur in a dialogue state. It could tell that the stars of a hotel might have a dependency on its price range. Take Figure 1 as another example, the user asks for a taxi to the restaurant, meaning that the taxi departure place can be inferred from the name of the hotel. According to the statistics, there are 36.53% slot-slot co-occurrence, 4.29% slot-value co-reference relations and many other types of slot dependencies in the training set of MultiWOZ 2.1 (Budzianowski et al., 2018; Feng et al., 2022). Intuitively, modeling these slot dependencies can help the DST model to handle complex dialogue scenes and infer the slot-value pairs in the zero-shot DST.

Motivated by above analysis, we consider that there are three kinds of slot dependencies, i.e. slot-slot dependency, slot-value dependency and slot-context dependency. This paper proposes a prompt-based approach to model above slot dependencies for zero-shot DST. For the slot-slot dependency, we combine slot prompts as the specialized prompt and decode corresponding slot values, making the model consider semantic information across slots. Specifically, each slot prompt consists of a slot-specific prompt and a slot-shared prompt, which respectively stimulates language understanding and captures the shared knowledge between slots by sharing parameters. For the slot-value dependency, we use value demonstration, i.e., filling partial slot values into slot prompts, to explore possible dependency between slots and values. For the slot-context dependency, we use the masked language model and predict masked tokens inside the context with the constraint of slot values, further enhancing the relationships between slots and dialogue context. The experimental results show that our proposed model achieves a significantly higher joint goal accuracy compared to previous zero-shot DST approaches.

In summary, our main contributions include:

- We propose a prompt-based method for zero-shot cross-domain DST, which leverages slot prompts combination, slot value demonstration and slot constraint object to explore the slot dependency among domains and slots.
- Experimental results show that our approach can transfer into unseen domains effectively and achieve the new state-of-the-art performances on the MultiWOZ 2.1 and SGD dataset under zero/few-shot settings.

2 Related Work

Multi-Domain Dialogue State Tracking Traditional statistical dialogue state tracking models combine semantics extracted by spoken language understanding models to predict the current dialogue state (Williams and Young, 2007; Thomson and Young, 2010) or jointly learn language understanding in an end-to-end way. Recently, many DST models that are built on deep neural networks have achieved promising state tracking results(Dai et al., 2021; Rastogi et al., 2019). Among them, some recent works attempted to model slot relationships by predefined schema graphs (Chen et al., 2020) or attention mechanism (Feng et al., 2021). But they heavily rely on a huge number of annotated data and human efforts without the generalizability to new domains (Feng et al., 2022), which is not suitable for industrial applications. To solve the above problem, some researchers leverage machine reading question answering data to facilitate the low-resource DST (Gao et al., 2020; Lin et al., 2021a), also called cross-task transfer. However, cross-task transfer needs a large-scale corpus and it is hard to learn the semantic consistency with the task-oriented dialogue. In this paper, we focus on the zero-shot cross-domain DST (Wu et al., 2019; Kumar et al., 2020; Lin et al., 2021b),where these models are first trained on several domains and are transferred into unknown domains.

Prompt-based Learning Various recent PLMs like GPT (Radford and Narasimhan, 2018), BERT (Devlin et al., 2019) and T5 (Raffel et al., 2020) provide a new approach to utilize large-scale unlabeled data for NLP tasks. However, there is a big gap between pre-training objectives and fine-tuning objectives. Recently, prompt tuning attracts many researchers to design prompt templates and then fine-tune PLMs to downstream tasks, which obtains successful results (Han et al., 2021; Zheng and Huang, 2021). For the DST task, Lee et al. (2021) proposed a slot-specific prompt to augment the multi-domain prompt-based DST model. However, these traditional prompt-based DST approaches handle slots independently while we focus on modeling the dependencies among slots in this paper.

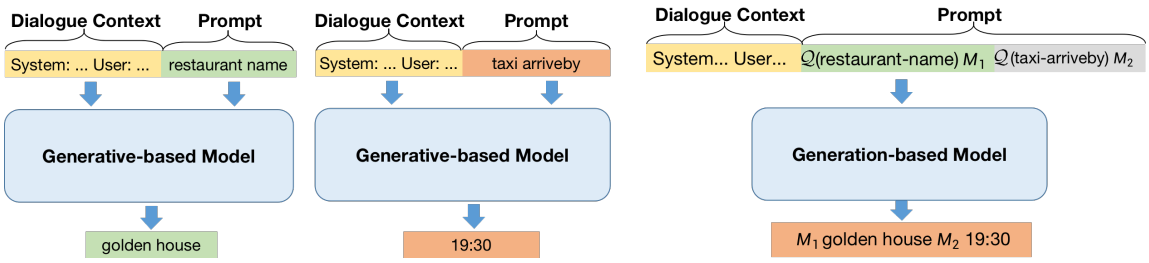

(a) (b)

Figure 2: Traditional prompt-based DST(a) vs Overview of ours(b).

3 Preliminaries

In this section, we give general notations for multi-domain DST task and details about the traditional prompt-based DST, which are the basis of proposed approach in the next section.

3.1 Notations

In task-oriented dialogue systems, a dialogue with T turns can be represented as $\{(A_1, U_1), (A_2, U_2) \ldots, (A_T, U_T)\}$, where A represents the system response and U represents a user utterance. At turn t, we denote the dialogue context as $C_t = \{(A_1, U_1), (A_2, U_2), \ldots, (A_t, U_t)\}$, which includes t turns from system and user. For multi-domain DST, the dialogue state at turn t is represented as a set of (slot, value) pairs, denoted as $B_t = \{(s_j, v_j)|1 \leq j \leq J\}$, where s_j is the slot name given by schema and v_j is its slot value. J is the total number of slots in all domains. If there is no information in the dialogue given about the slot s_j, v_j is set to "none". The goal of DST is to predict the dialogue state B_t given a dialogue context C_t.

3.2 Traditional Prompt-based DST

In this part, we introduce traditional prompt-based DST model (Lin et al., 2021b) with a sequence-to-sequence framework, which is shown in Figure 2(a). A generative model (e.g T5) concatenates dialogue history C_t and a slot-specific prompt $\mathcal{T}_j$ as input and decodes corresponding slot value v_j.

$$v_j = Seq2seq(C_t, \mathcal{T}_j) \tag{1}$$

where $\mathcal{T}_j$ is the prompt for slot s_j. The learning objective of the generation process is minimizing the negative log-likelihood of v_j given context C_t and prompt $\mathcal{T}_j$:

$$L = -\sum_{t}^{T}\sum_{j}^{J} \log p(v_j|C_t, \mathcal{T}_j) \tag{2}$$

The example in Figure 2(a) takes slot name as the slot-specific prompt. For the input with different slots like "restaurant name" and "taxi arriveby", the model generates slot value independently, i.e. "golden house" and "19:30".

4 Methodology

As we mentioned before, we argue that traditional prompt-based DST approaches ignore significant slot dependencies in a dialogue. In this paper, we propose a prompt-based approach to model the dependency of the slot-slot, the slot-value, and the slot-context. The architecture of our model is shown in Figure 3.

4.1 Slot-slot Dependency Modeling

The traditional prompt-based DST utilizes the slot-specific prompt independently. Differently, we compose multiple slot prompts as the final prompt to model the slot-slot dependency. A generation-based model concatenates composed prompt $\mathcal{T}$ and dialogue context C_t as input, and decodes a sequence of values:

$$\begin{aligned} \mathcal{T} &= \text{“}\mathcal{Q}(s_1), M_1, ..., \mathcal{Q}(s_J), M_J\text{”} \\ \mathcal{V} &= Seq2seq(C_t, \mathcal{T}) \end{aligned} \tag{3}$$

where $\mathcal{Q}(s_i)$ refers the slot prompt of slot s_i. Here, $\mathcal{V} =$ "$M_1, v_1, \ldots, M_J, v_J$" and M_* are mask tokens. Each mask token is inserted after the slot prompt and it is also the start token for a slot value. We insert mask tokens inside the prompt because we hope the model can focus on the specific slot prompt when generating its corresponding value.

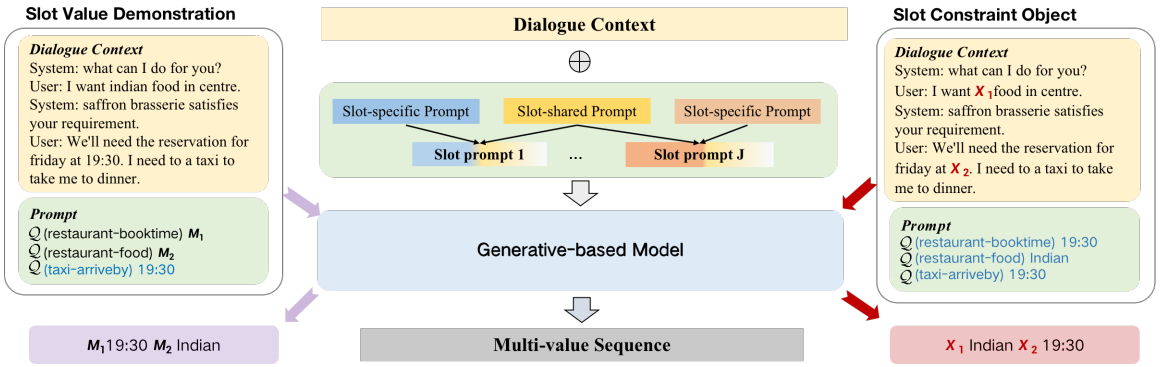

Figure 3: The architecture of our proposed model. It concatenates dialogue context (in yellow) and several slot prompts (in green) to model the slot-slot dependency. The model explores slot-value dependency and slot-context dependency utilizing slot value demonstration (left-shown example) and slot constraint object (right-shown example). Noted that M_* and X_* are mask tokens and $\mathcal{Q}(slot)$ represents a slot prompt.

The objective of the generative model is to minimize the sequence of values given context C_t and prompt $\mathcal{T}$:

$$L = -\sum_{t}^{T} \log p(\mathcal{V}|C_t, \mathcal{T}) \tag{4}$$

Figure 2(b) shows the overview of our method, in which the model concatenates a dialogue context and two slot prompts, "restaurant-name" and "taxi-arriveby", and then generates the sequence "M_1 golden house M_2 19:30".

Slot Prompt Design For each slot, we utilize two types of prompts, a slot-specific prompt and a slot-shared prompt to construct the slot prompt. The slot-specific prompt stimulates the language understanding from PLMs and the slot-shared prompt capture the universal knowledge across slots.

Formally, we define a slot-specific prompt as $\{P_1^k, P_2^k, \ldots, P_I^k\}$ for the slot s_k, and a slot-shared prompt as $\{P_1', P_2', \ldots, P_Q'\}$ for all slots. The I and Q are the number of slot-specific tokens and pseudo tokens respectively. For a slot s_k, its slot prompt $\mathcal{Q}(s_k)$ is written as:

$$\{P_1^k, \ldots, P_I^k, P_1', \ldots, P_Q'\} \tag{5}$$

For instance, the slot prompt of "restaurant-name" can be "restaurant name $[P_1'], [P_2']$"[2]. The slot prompt embedding of slot s_k is represented as follows:

$$PE(s_k) = \{e_1^k, \ldots, e_I^k, h_1', \ldots, h_Q'\} \tag{6}$$

[2]We try different value of Q and the optimal value 2 is selected using the validation set

where e_* are original word embeddings. h_*' are trainable embedding tensors, which are encoded by a full-connected network and share parameters across slots.

4.2 Slot-Value Dependency Modeling

Except for the dependency among slots, we find that many slot values are also highly correlated, i.e. demonstrated by co-reference and exclusion. For example, in a dialogue, the value of "taxi-departure" might be inferred from "hotel-name" but must be different from the value of "taxi-destination". We suppose that considering other slot values helps the model to capture the slot-value dependency and understand the dialogue context better.

Specifically, we introduce some ground-truth slot values into the prompt $\mathcal{T}$, called **slot value demonstration**. Since there are multiple mask tokens in prompt, we replace each mask token with its slot value at ratio β (a hyper-parameter). The left example in Figure 3 takes an input with three slot prompts and one of them is supplied with the slot value (in blue). Accordingly, the model only needs output two slot values, "19:30" for "restaurant-booktime" and "Indian" for "restaurant-food".

4.3 Slot-Context Dependency Modeling

To model the dependency between slots and dialogue context, we introduce a **slot constraint object** with a masked language model. Specifically, we first utilize ground-truth slot values to fill mask tokens, obtaining a new prompt $\tilde{\mathcal{T}}$. After that, we use other symbols X_* to mask v_* inside the context. The slot constraint objective is to predict the masked values sequence given context $\tilde{C}_t$ and

prompt $\tilde{\mathcal{T}}$:

$$\tilde{\mathcal{T}} = \text{“}\mathcal{Q}(s_1), v_1, \ldots, \mathcal{Q}(s_J), v_J\text{”}$$
$$L_{sc} = -\sum_{t}^{T} \log p(\mathcal{W}|\tilde{C}_t, \tilde{\mathcal{T}}) \quad (7)$$

where $\mathcal{W}$ = “$X_1, w_1, X_2, w_2, \ldots, w_Z$” and w_i refers the masked value inside context. For the slot that its value is unable to match strings in dialogue context (e.g “none”), we skip the mask operation to it. Therefore, the number of masked values Z might not be equal to the number of slot prompts, and w_i might not actually be v_i.

Take the right-shown example in Figure 3 for illustration. Although there are three slots in the prompt, only two mask symbols (X_1 and X_2) are used in the context. The reason is that the value of “taxi-arriveby” is inferred from “restaurant-booktime”, causing the same location for their values in context.

4.4 Training and Inference

During training, we have the following loss function:

$$L_{train} = -\sum_{t}^{T} \log p(\mathcal{V}|C_t, \mathcal{T}) + \lambda L_{sc} \quad (8)$$

where $\mathcal{T}$ is a specific prompt using slot value demonstration. λ is a hyper-parameter and controls the weight of slot constraint object. During Inference, considering that the number of unseen domains and slots might be huge, we concatenate single slot prompt and dialogue context as input to predict slot value, just like traditional prompt-based DST (shown in Section 3.2).

5 Experiments

5.1 Datasets

We evaluate the proposed model on the most popular multi-domain task-oriented dialogue benchmarks, MultiWOZ (Budzianowski et al., 2018; Eric et al., 2020) and Schema-Guided-Dialogue(SGD) (Raffel et al., 2020). Both datasets provide turn-level annotations of dialogue states and descriptions of domain and slot. The MultiWOZ dataset contains over 10K dialogue across 8 domains. We follow the previous pre-processing and evaluation setup (Lin et al., 2021b), where the restaurant, train, attraction, hotel, and taxi domains are used for training and testing. Appendix A gives more statistics of MultiWOZ datasets. The SGD dataset has over 16K dialogues in the training set, spanning 26 services belonging to 16 domains. The test set has 18 domains, and 5 domains of them are not presented in the training set.

5.2 Baselines

We compare the performance of our model with the following existing models. **TRADE** (Wu et al., 2019) leverages context-enhanced slot gate and copy mechanism to track slot values mentioned in dialogue history . **SUMBT** (Lee et al., 2019) proposes a non-parametric method to score each candidate slot-value pair in a predefined ontology . **MA-DST** (Kumar et al., 2020) designs multiple layers of cross-attention to capture relationships at different levels of dialogue granularity. **DSTQA** (Zhou and Small, 2019) models the DST task as a question answering problem and uses a dynamically-evolving knowledge graph to learn the relationships between domains . **T5DST** (Lin et al., 2021b) is a strong prompt baseline that first uses slot descriptions as a prompt in zero-shot cross-domain DST. **TransferQA** (Lin et al., 2021a) is a cross-task zero-shot DST method where the model is pre-trained on question answering data first and then is applied to unseen domains. **SGD-baseline** (Rastogi et al., 2020) uses schema descriptions and applies a BERT-based DST model to predict the dialogue state of unseen domains.

5.3 Evaluation

Following previous works (Lin et al., 2021b), we use Joint Goal Accuracy(JGA) and Average Goal Accuracy (AGA) to evaluate our models and baselines. Joint goal accuracy is the percentage of turns for which all the slots are correctly identified. Average goal accuracy is the average accuracy of the *active* slots in each turn. A slot becomes *active* if its value is mentioned in the current turn and is not inherited from previous turns. We compute JAG per domain in MultiWOZ datasets and use the official evaluation script in SGD dataset.

In zero-shot settings, all models are trained on four domains in the MultiWOZ dataset then zero-shot on the held-out domain. In the SGD dataset, there are 5 domains in the testing set but are not in the training set, so all models are trained with the whole training set and tested on these 5 unseen domains. For few-shot experiments in MultiWOZ dataset, all models are first trained on 4 source domains and then fine-tuned with 1%, 5%, and

Model	Pretrained-Model	Joint Goal Accuracy					
		Attraction	Hotel	Restaurant	Taxi	Train	Average
TRADE (Wu et al., 2019)	N	20.06	14.20	12.59	59.21	22.39	25.69
MA-DST (Kumar et al., 2020)	N	22.46	16.28	13.56	59.27	22.76	26.87
SUMBT (Lee et al., 2019)	Bert-base	22.60	19.08	16.50	59.50	22.50	28.18
T5DST † (Lin et al., 2021b)	T5-small	31.92	20.72	20.09	64.12	28.83	33.56
Ours †	T5-small	33.92	19.85	20.75	66.25	36.96	35.55
T5DST †* (Lin et al., 2021b)	T5-base	35.51	22.48	25.04	65.93	34.82	36.25
TransferQA †(Lin et al., 2021a)	T5-large	31.25	22.72	26.28	61.87	36.72	35.77
Ours †	T5-base	**37.83**	**26.50**	**27.05**	**69.23**	**40.27**	**40.18**

Table 1: Zero-shot results on MultiWOZ 2.1. All numbers are reported in joint goal accuracy(%). The averaged zero shot joint goal accuracy among five domains is reported. All results of baselines are from the original public papers, except for T5DST* where we rerun their code with T5-base. † means the model is a prompt-based method. For fair comparison, all prompt-based methods use the slot-description provided from schema as slot-specific prompt.

10% of target domain data. The zero-shot/few-shot settings are consistent with the previous works on zero-shot cross-domain DST (Wu et al., 2019; Lin et al., 2021a,b).

5.4 Implementation

We implement our approach based on T5-small (60M parameters) and T5-base (220M parameters) (Raffel et al., 2020). We train the model with a batch size of 128 for T5-small and a batch size of 256 for T5-base. Both of them are trained using AdamW optimizer (Loshchilov and Hutter, 2019). The peak learning rate is set to 1e-4 for T5 and 2e-4 for other learned modules. To balance the efficiency and performance of the model, we adopt a random sampling strategy in slot prompts combination, i.e. setting a hyper-parameter α as the max number of slot prompts. Given a dialogue context, the model randomly selects 1 up to α slots to construct the prompt. And we apply multiple iterations for training so that almost all slots can be sampled. In all experiments, the α is set to 3, β is set to 0.5 and the weight λ in loss function is 0.3. We use greedy decoding for all models.

6 Main Results

6.1 Zero-Shot Cross-Domain Results

Table 1 gives the results of our model and baselines under the zero-shot setting. Compared to previous works, our model using T5-base achieves significantly higher JGA (3.93% on average) and even exceeds the cross-task method using T5-large (TransferQA). Among these baselines, the methods (T5DST and TransferQA) using T5 model have much better performances than those without pre-trained models (TRADE and MA-DST). We analyze that T5 is pre-trained on a large unlabeled corpus, which can provide a promising language understanding for unseen slots. Notably, our method using T5-small outperforms prior prompt-based DST (T5DST) on almost domains, except for hotel domain. Appendix B shows that our method exceeds T5DST on six hotel slots but falls behind on four slots. The reason is that these four slots are completely independent from source domains, making their prediction mainly depend on the ability of language understanding. Our proposed model with small trainable parameters tends to build the slot dependencies, which might hurt partial language understanding in PLMs. However, our model with T5-base brings obvious improvements on all domains, including hotel domain. It also verifies that the stronger ability of language understanding the pre-trained model has, the easier to benefit from slot dependency modeling our method is.

Table 3 summarizes the zero-shot results on SGD dataset. Compared with SGD-baseline, the zero-shot performance of our model is consistently higher in five unseen domains. Compared to transferQA with T5-large and large labeled training data (QA dataset), our model with T5-small is still competitive in zero-shot settings. Particularly, our model gains a great improvement on "bus" and "train" domains. We analysis that these two domains are closely related to some seen domains, e.g "flight" and "travel" domains, which easily benefit from the slot dependency modeling.

6.2 Few-Shot Cross-Domain Results

We further conduct experiments in few-shot cross-domain settings on MultiWOZ 2.0, as in (Wu et al., 2019; Lin et al., 2021a,b). The models are first trained on 4 domains and then fine-tuned with 1%,

Model	Attraction			Hotel			Restaurant			Taxi			Train		
	1%	5%	10%	1%	5%	10%	1%	5%	10%	1%	5%	10%	1%	5%	10%
TRADE	35.88	57.55	63.12	19.73	37.45	41.42	42.42	55.70	60.94	63.81	66.58	70.19	59.83	69.27	71.11
DSTQA	N/A	**70.47**	71.60	N/A	50.18	53.68	N/A	58.95	64.51	N/A	70.90	74.19	N/A	70.35	74.50
T5DST	58.77	65.72	69.54	43.07	50.71	54.86	57.63	61.86	63.47	70.12	73.67	74.70	70.82	74.18	77.57
Our Approach	**60.03**	69.69	**71.61**	**45.76**	**52.53**	**56.71**	**60.56**	**64.24**	**67.31**	**76.23**	**78.32**	**79.61**	**70.93**	**75.50**	**77.89**

Table 2: Few-shot experiments in MultiWOZ 2.0. The experiments are conducted on MultiWOZ 2.0 for comparing with previous works. N/A represents the results are not reported in the original paper.

Domain	SGD-baseline	TransferQA	Seq2seq-DU	Ours
Bus	9.7/50.9	15.9/63.6	16.8/N	43.9/86.3
Messaging	10.2/20.0	13.3/37.9	4.9/N	36.6/61.4
Payment	11.5/34.8	24.7/60.7	7.2/N	16.5/62.0
Trains	13.6/63.5	17.4/64.9	16.8/N	46.7/86.9
Alarm	57.7/1.8	58.3/81.7	55.6/N	58.3/87.5
Average	20.5/34.2	25.9/61.8	20.3/N	**40.4/76.8**

Table 3: Zero-Shot results on SGD dataset. All results are reported in JGA(%)/AVG(%). Seq2seq-DU(Feng et al., 2021) is seq2seq baseline without any pre-trained model. N represents the results are not reported in the original paper.

Models	Pretrained-Model	Zero-shot	JGA
TRADE (Wu et al., 2019)	N	✔	45.60
STARC (Gao et al., 2020)	Bert-base	✔	49.48
SGD-baseline (Rastogi et al., 2020)	Bert-base	✔	43.40
T5DST (Lin et al., 2021b)	T5-small	✔	51.91
T5DST (Lin et al., 2021b)	T5-base	✔	53.15
Ours	T5-small	✔	52.83
Ours	T5-base	✔	**54.75**
MinTL(Lin et al., 2020)	BART	✘	50.95
SOM-DST(Kim et al., 2020)	Bert-base	✘	53.68
Tripy (Heck et al., 2020)	Bert-base	✘	55.29
Simple-TOD (Hosseini-Asl et al., 2020)	GPT-2	✘	55.72

Table 4: Full data results on MultiWOZ 2.1 dataset.

5%, and 10% of target domain data. In Table 3, the experiment result shows that DSTQA is a competitive baseline. However, our approach outperforms previous transfer-learning methods in almost all domains, except for the situation with 5% *Attraction* domain data fine-tuning. We suppose that the DSTQA introduces an extra schema graph to model explicit relationships across slots. The significant improvements on most domains indicate that our model still keeps a robust learning ability with a minute quantity of dialogue fine-tuning.

6.3 Full Data Results

We also evaluate our model on full dataset to understand the full-shot performance, and the results are shown in Table 4. Compared with prior models with zero-shot capability, our model improves the joint goal accuracy by 1.6% in MultiWOZ 2.1 dataset. Particularly, our model exceeds traditional prompt-based methods, T5DST, illustrating that modeling slot dependency is helpful even in a full-data scene. We notice that many training strategies can be applied into the full-data experiment, such as additional supervision (Chen et al., 2020) and pre-process strategies (Heck et al., 2020), that may improve the performances. In this paper, we focus on modeling slot dependency for zero-shot DST not achieving state-of-art on full-data.

7 Discussion

7.1 Ablation Study

In Table 5, we study the effect of different modules for the proposed model in the zero-shot setting. Firstly, we set the hyper-parameter α as 1 to check the effect of several slot prompts. There has 2% drop of performance on hotel and taxi domain. Secondly, we only use the slot-specific prompt to investigate the effect of the slot-shared prompt. One can observe that the performance deteriorates considerably, which is similar to the results of removing composing slot prompts. It indicates that composing specialized slot prompts can enhance the pre-trained model's ability on predicting unseen domains and slots. Thirdly, we explore the effect of slot value demonstration by setting the ratio β as 0. The performance of the model decreases markedly, especially for taxi domain. We conclude that value demonstration in prompt can effectively explore the slot-value dependency, such as co-reference and exclusion. These relationships mainly occur in some slots related to time or location, causing a huge influence on taxi domain. Furthermore, the model without slot constraint object performs declining results on different domains, which illustrates that learning the slot-context dependency is also important for zero-shot learning.

7.2 Analysis of Parameters

We further investigate the impacts of hyper-parameter settings on the performance of the proposed model on MultiWOZ2.1 in zero-shot set-

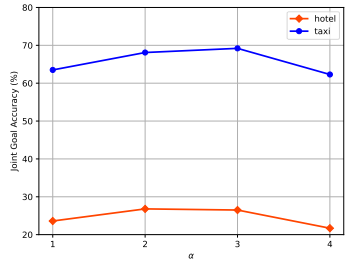

Figure 4: Effects of the max number of slot prompts α.

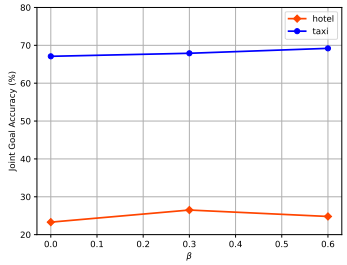

Figure 5: Effects of ratio of β in value demonstrations.

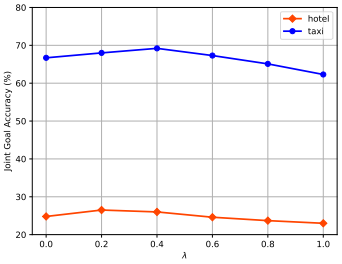

Figure 6: Effects of the weight of slot constraint object.

Model	Joint Goal Accuracy	
	Hotel	Taxi
Our approach	**26.5**	**69.2**
w/o Slot Prompt Combination	24.9(-1.6)	67.1(-2.1)
w/o Slot-shared Prompt	25.3(-1.2)	67.9(-1.3)
w/o Slot Value Demonstration	25.8(-0.7)	67.4(-1.8)
w/o Slot Constraint Object	25.9(-0.6)	67.9(-1.3)

Table 5: Ablation studies on the MultiWOZ 2.1 in zero-shot setting on target domain *hotel* and *taxi*.

tings. We validate the effects of three factors: the max number α of slot prompts, the ratio of value demonstrations β, and the weight λ in the loss function. Figure 4, 5, 6 show the results of proposed model with varying parameters in zero-shot setting on domain hotel and taxi. We observe that the optimal parameters are not completely consistent across different domains. In Figure 4 and 5, the model achieves a better performance with more slot prompts and a bigger ratio of value demonstrations on taxi domain. We conjecture that taxi domain only has four slots ("taxi-departure" , "taxi-destination", "taxi-arriveby" , and "taxi-leaveat") and all of them are related to source domains, such as the co-reference between "hotel-name" and "taxi-destination" . That means that exploring the slot-value dependency has a bigger influence on taxi domain than hotel domain. Figure 6 show the effect of using different weight λ in the loss function. When the weight of the slot constraint object is too low, the model doesn't own enough strong constraint for slot-context dependency; when it is too high, the model tends to over-predict "masked tokens" not track dialogue state. Finally, we find that our model achieves a balance with 0.2∼0.4.

7.3 Case Study

In Figure 7, we make a qualitative analysis of the results of T5DST and our method on the MultiWOZ dataset under zero-shot settings. From the results, we find that both models accurately predict the "Ballare" of "taxi-destination" and "17:30" of "taxi-arriveby". These two slot values are easy to predict because they don't depend on any other domains and slots. Besides, our method generates the "lovell lodge" for "taxi-departure", while the T5DST model outputs a wrong value, i.e., "none". We analyze that the T5DST leverages slot-specific prompt and generates slot value independently, which can not infer the relations between slots and values. Our approach leverages slot prompts combination and slot value demonstrations, making it possible to model the slot-slot and slot-value dependencies.

Dialogue Context	System: hi. what can i do for you? User: Can you give me information on an attraction called ballare ? System:The Ballare is located in Heidelberg Gardens User: Thanks. I'm also looking for somewhere to stay in the north. System: Would you want to try the lovell lodge. User: Let s do that .Can you help me to book a taxi from the hotel to the Ballare. I want to leave by 17:30 .
Golden Dialogue State	(taxi-leaveat,17:30) (taxi-destination,ballare) (taxi-departure,lovell lodge)
T5DST	(taxi-leaveat,17:30) (taxi-destination,ballare) (taxi-departure,none)
Ours	(taxi-leaveat,17:30) (taxi-destination,ballare) (taxi-departure,lovell lodge)

Figure 7: The zero-shot evaluation results for T5DST vs. Ours. We mark the key information in blue and the wrong prediction in red.

8 Conclusion and Future Work

In this paper, we attempt to model three slot dependencies for zero-shot cross-domain DST, i.e. slot-slot dependency, slot-value dependency, and slot-context dependency. Experimental results on popular datasets show that the proposed approach performs much better than baselines in zero-shot/few-

shot settings. In the future, we would like to explore more ways to model slot dependency effectively.

Acknowledgements

This work is supported by the Youth Innovation Promotion Association of the Chinese Academy of Sciences (No. 2018192), the NSFC (No. 61902394). We would like to thank the anonymous reviewers for their valuable comments.

References

Paweł Budzianowski, Tsung-Hsien Wen, Bo-Hsiang Tseng, Iñigo Casanueva, Stefan Ultes, Osman Ramadan, and Milica Gasic. 2018. Multiwoz - a large-scale multi-domain wizard-of-oz dataset for task-oriented dialogue modelling. *ArXiv*, abs/1810.00278.

Lu Chen, Boer Lv, Chi Wang, Su Zhu, Bowen Tan, and Kai Yu. 2020. Schema-guided multi-domain dialogue state tracking with graph attention neural networks. In *AAAI*.

Yinpei Dai, Hangyu Li, Yongbin Li, Jian Sun, Fei Huang, Luo Si, and Xiaodan Zhu. 2021. Preview, attend and review: Schema-aware curriculum learning for multi-domain dialogue state tracking. *ArXiv*, abs/2106.00291.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. Bert: Pre-training of deep bidirectional transformers for language understanding. In *NAACL*.

Mihail Eric, Rahul Goel, Shachi Paul, Adarsh Kumar, Abhishek Sethi, Anuj Kumar Goyal, Peter Ku, Sanchit Agarwal, and Shuyang Gao. 2020. Multiwoz 2.1: A consolidated multi-domain dialogue dataset with state corrections and state tracking baselines. In *LREC*.

Yue Feng, Aldo Lipani, Fanghua Ye, Qiang Zhang, and Emine Yilmaz. 2022. Dynamic schema graph fusion network for multi-domain dialogue state tracking. In *ACL*.

Yue Feng, Yang Wang, and Hang Li. 2021. A sequence-to-sequence approach to dialogue state tracking. In *ACL*.

Shuyang Gao, Sanchit Agarwal, Tagyoung Chung, Di Jin, and Dilek Z. Hakkani-Tür. 2020. From machine reading comprehension to dialogue state tracking: Bridging the gap. *ArXiv*, abs/2004.05827.

Xu Han, Weilin Zhao, NNN, Zhiyuan Liu, and Maosong Sun. 2021. Ptr: Prompt tuning with rules for text classification. *ArXiv*, abs/2105.11259.

Michael Heck, Carel van Niekerk, Nurul Lubis, Christian Geishauser, Hsien-Chin Lin, Marco Moresi, and Milica Gavsi'c. 2020. Trippy: A triple copy strategy for value independent neural dialog state tracking. *ArXiv*, abs/2005.02877.

Ehsan Hosseini-Asl, Bryan McCann, Chien-Sheng Wu, Semih Yavuz, and Richard Socher. 2020. A simple language model for task-oriented dialogue. *ArXiv*, abs/2005.00796.

Sungdong Kim, Sohee Yang, Gyuwan Kim, and Sang-Woo Lee. 2020. Efficient dialogue state tracking by selectively overwriting memory. In *ACL*.

Adarsh Kumar, Peter Ku, Anuj Kumar Goyal, Angeliki Metallinou, and Dilek Z. Hakkani-Tür. 2020. Ma-dst: Multi-attention based scalable dialog state tracking. In *AAAI*.

Chia-Hsuan Lee, Hao Cheng, and Mari Ostendorf. 2021. Dialogue state tracking with a language model using schema-driven prompting. *ArXiv*, abs/2109.07506.

Hwaran Lee, Jinsik Lee, and Tae-Yoon Kim. 2019. Sumbt: Slot-utterance matching for universal and scalable belief tracking. In *ACL*.

Zhaojiang Lin, Bing Liu, Andrea Madotto, Seungwhan Moon, Paul A. Crook, Zhenpeng Zhou, Zhiguang Wang, Zhou Yu, Eunjoon Cho, Rajen Subba, and Pascale Fung. 2021a. Zero-shot dialogue state tracking via cross-task transfer. In *EMNLP*.

Zhaojiang Lin, Bing Liu, Seungwhan Moon, Paul A. Crook, Zhenpeng Zhou, Zhiguang Wang, Zhou Yu, Andrea Madotto, Eunjoon Cho, and Rajen Subba. 2021b. Leveraging slot descriptions for zero-shot cross-domain dialogue statetracking. In *NAACL*.

Zhaojiang Lin, Andrea Madotto, Genta Indra Winata, and Pascale Fung. 2020. Mintl: Minimalist transfer learning for task-oriented dialogue systems. *ArXiv*, abs/2009.12005.

Ilya Loshchilov and Frank Hutter. 2019. Decoupled weight decay regularization. In *ICLR*.

Alec Radford and Karthik Narasimhan. 2018. Improving language understanding by generative pre-training.

Colin Raffel, Noam M. Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J. Liu. 2020. Exploring the limits of transfer learning with a unified text-to-text transformer. *ArXiv*, abs/1910.10683.

Abhinav Rastogi, Xiaoxue Zang, Srinivas Sunkara, Raghav Gupta, and Pranav Khaitan. 2020. Towards scalable multi-domain conversational agents: The schema-guided dialogue dataset. *ArXiv*, abs/1909.05855.

Pushpendre Rastogi, Arpit Gupta, Tongfei Chen, and Lambert Mathias. 2019. Scaling multi-domain dialogue state tracking via query reformulation. In *NAACL*.

Yixuan Su, Lei Shu, Elman Mansimov, Arshit Gupta, Deng Cai, Yi-An Lai, and Yi Zhang. 2021. Multi-task pre-training for plug-and-play task-oriented dialogue system. *ArXiv*, abs/2109.14739.

Blaise Thomson and S. Young. 2010. Bayesian update of dialogue state: A pomdp framework for spoken dialogue systems. *Comput. Speech Lang.*, 24:562–588.

J. Williams and S. Young. 2007. Partially observable markov decision processes for spoken dialog systems. *Comput. Speech Lang.*, 21:393–422.

Chien-Sheng Wu, Andrea Madotto, Ehsan Hosseini-Asl, Caiming Xiong, R. Socher, and Pascale Fung. 2019. Transferable multi-domain state generator for task-oriented dialogue systems. In *ACL*.

Chujie Zheng and Minlie Huang. 2021. Exploring prompt-based few-shot learning for grounded dialog generation. *ArXiv*, abs/2109.06513.

Li Zhou and Kevin Small. 2019. Multi-domain dialogue state tracking as dynamic knowledge graph enhanced question answering. *ArXiv*, abs/1911.06192.

A Dataset Statistics

The MultiWOZ dataset is a fully-labeled collection of human-human written conversations spanning multiple domains and topics. Some statistics of MultiWOZ 2.1 are reported in Table 6. We further draw a schema graph to illustrate the slot dependency, which is shown in Figure 8.

Domain	**Slot**	*Train*	*Valid*	*Test*
Attraction	area, name, type	2717	401	395
Hotel	area, internet, name, parking, price range, stars, type, book day, book people, book stay	3381	416	394
Restaurant	area, food, name, price range, book day, book people, book time	3813	438	437
Taxi	arrive by, departure, destination, leave at	1654	207	195
Train	arrive by, day, departure, destination, leave at, book people	3103	484	494
Total		8438	1000	1000

Table 6: The dataset statistics of MultiWOZ dataset.

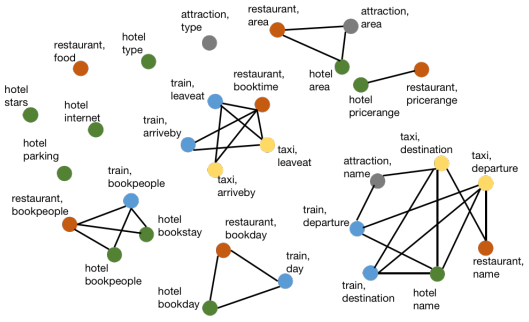

Figure 8: The schema graph on MultiWOZ dataset. Each nodes represents a slot and the nodes in same color belong to a domain. There is an edge between two nodes if some of their candidate values are same.

B Performance on Per-Slot

Figure 9 shows the difference in performance between T5DST and ours in the hotel domain when using T5-small. From the results, our method exceeds T5DST on six slots while falling behind on four slots, i.e "stars", "internet", "type" and "parking". In Figure 10, we list the performance of per-slot on taxi domain when using T5-small. There are four slots in taxi and all of them are related to

source domains. Our method can effectively handle these slots due to the modeling of slot dependency.

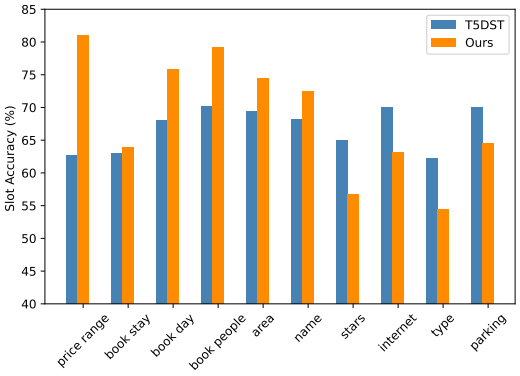

Figure 9: Slot Accuracy in hotel of MultiWOZ 2.1.

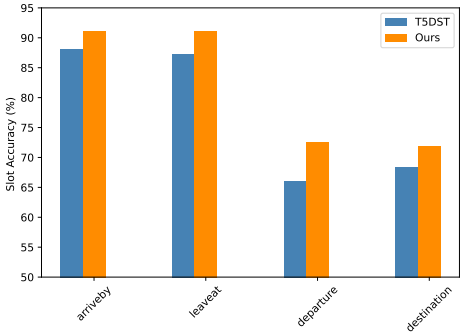

Figure 10: Slot Accuracy in taxi of MultiWOZ 2.1.

Section-Aware Commonsense Knowledge-Grounded Dialogue Generation with Pre-trained Language Model

Sixing Wu[1], Ying Li[2,3*], Ping Xue[2], Dawei Zhang[1], Zhonghai Wu[2,3]
[1]School of Computer Science, Peking University, Beijing, China
[2] School of Software and Microelectronics, Peking University, Beijing, China
[3]National Research Center of Software Engineering, Peking University, Beijing, China

Abstract

In knowledge-grounded dialogue generation, pre-trained language models (PLMs) can be expected to deepen the fusing of dialogue context and knowledge because of their superior ability of semantic understanding. Unlike adopting the plain text knowledge, it is thorny to leverage the structural commonsense knowledge when using PLMs because most PLMs can only operate plain texts. Thus, linearizing commonsense knowledge facts into plan text is a compulsory trick. However, a dialogue is always aligned to a lot of retrieved fact candidates; as a result, the linearized text is always lengthy and then significantly increases the burden of using PLMs. To address this issue, we propose a novel two-stage framework *SAKDP*. In the first pre-screening stage, we use a ranking network *PriorRanking* to estimate the relevance of a retrieved knowledge fact. Thus, facts can be clustered into three sections of different priorities. As priority decreases, the relevance decreases, and the number of included facts increases. In the next dialogue generation stage, we use section-aware strategies to encode the linearized knowledge. The powerful but expensive PLM is only used for a few facts in the higher priority sections, reaching the performance-efficiency balance. Both the automatic and human evaluation demonstrate the superior performance of this work.

1 Introduction

Dialogue systems strive to facilitate human-like dialogue responses (Chen et al., 2017). One essential precondition for generating high-quality dialogue is a sufficient cognition of the contextually-relevant knowledge besides the literal surface. The dialogue is grounded on both the given user query and the context-related knowledge (Yu et al., 2020). For example, given a query 'Are you thirsty?', rather than 'Yes/No', a meaningful response should have more information, such as 'Yes, and I'm going to drink some water.' (context-related knowledge →*thirsty causes drinking water*).

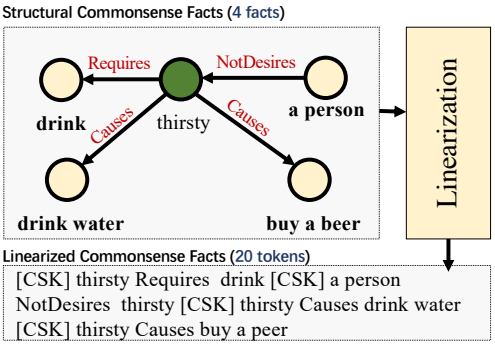

Figure 1: An example of linearizing knowledge.

Seeking information from external sources is a feasible way to enhance the machine's cognition of knowledge (Dinan et al., 2019; Zhou et al., 2021b). As for the choice of knowledge source, the structural commonsense knowledge (Speer et al., 2017) is a proven option, which consists of a lot of knowledge facts that are frequently used in daily life. As shown in Figure 1, a commonsense knowledge base consists of various triplets, where each triplet is a real-world fact. Commonsense knowledge can contribute to many aspects of the open-domain dialogue generation, such as the semantic understanding (Young et al., 2018), reasoning (Liu et al., 2019), topic transition (Zhong et al., 2021). Meanwhile, pre-trained language models (PLMs) (Vaswani et al., 2017) can learn a lot of implicit knowledge from the massive pretraining data (Sun et al., 2019, 2021; Zhou et al., 2021c). Hence, another feasible way is to transfer the knowledge hidden in PLMs to the dialogue generation (Henderson et al., 2020). Prior works have shown PLMs can significantly promote the generation of high-quality dialogue responses (Wolf et al., 2019; Zhang et al., 2020b; Wang et al., 2020; Zhang et al., 2021), stylized responses (Yang et al., 2020), etc.

* Corresponding author: Ying Li, li.ying@pku.edu.cn. The email of the first author: wusixing@pku.edu.cn

Proceedings of the 29th International Conference on Computational Linguistics, pages 521–531
October 12–17, 2022.

A model can simultaneously adopt the aforementioned two ways to take a step further. KnowledGPT (Zhao et al., 2020) adopted a BERT (Devlin et al., 2019) to select text knowledge and a GPT2 (Radford et al., 2019) to generate responses. KE-Blender (Cui et al., 2021a) can implicitly infer knowledge by fine-tuning Blender (Roller et al., 2021) with text knowledge. Nonetheless, most PLMs can only process plain texts, which makes it hard to infuse the structural knowledge (Zhao et al., 2021). As a compromise, linearizing structural knowledge into plain text is a compulsory trick (Li et al., 2021a). The linearized knowledge text is much more verbose than the original, bringing new challenges in commonsense knowledge-grounded dialogue generation. A dialogue session is often paired with a lot of commonsense fact triplets; for example, in *Reddit* dataset (Zhang et al., 2020a), the average number of 1/2-hop facts is 98.6/782.2, respectively. As shown in Figure 1, linearizing a fact into text often requires 5+ tokens. Thus, lengthy linearized text can significantly aggravate the burden of Transformer-based PLMs[1] and often exceeds the limits of most general PLMs (e.g., 512/1024 tokens).

According to our empirical study, fact candidates retrieved for a dialogue query are always redundant, where most responses (98.25%) use no more than 3 facts, but 77.65 facts are given on average. Inspired by such an observation, this paper proposes a novel two-stage framework *SAKDP (Section-Aware Knowledge-Grounded Dialogue Generation with Pre-trained Language Model)*. The powerful but expensive PLM is only used to encode a few facts of higher relevance. **First**, in the pre-screening stage, we train a *PriorRanking* network using the contrastive learning scheme (Wu et al., 2020b) to estimate the relevance and then cluster fact candidates into three sections of different priorities: *high*, *moderate*, and *low*. **Second**, considering the investment benefit ratio, we use different encoding solutions in the following dialogue generation stage. We propose a BERT-based *Context-Knowledge Joint Encoder* to jointly encode the dialogue query and the relevant facts included by the high/moderate priority sections, bringing deeper infusing and interaction between dialogue and relevant knowledge. Then, we employ a lightweight non-pretrained *Side-way Encoder* to encode facts in the low priority section. **Third**, we use a *Hybrid Selection* to select the encoded context/knowledge memories and a *Multi-Source Generator* to generate diverse dialogues.

Experiments on a Chinese dataset (Wu et al., 2020a) prove *SAKDP* can outperform baselines by a large margin. Meanwhile, we conduct extensive studies to analyze the ranking performance and the necessity of pre-screening. *SAKDP* is still competitive even only using three facts. The novelty/contribution of this work is three-fold: 1) We propose a novel *SAKDP* to investigate the potential of both commonsense knowledge and pre-trained language models; 2) We propose to rank and cluster knowledge into three sections of different priorities. It can maximize cost-effectiveness and flexibility by using section-aware schemes; 3) Extensive experiments and studies demonstrate the performance of *SAKDP*.

2 Methodology

2.1 Preliminary

Inputs: Each dialogue is denoted as (X, Y), where $X = (x_1, \cdots, x_{l_X})$ is a query and $Y = (y_1, \cdots, y_{l_Y})$ is a response. Besides, there is a commonsense knowledge base $\mathcal{K} = \{k_i\}^{|\mathcal{K}|}$, where each triplet $k_i = (e_{head,i}, e_{rel,i}, e_{tail,i})$ consists of a head entity, a relation, and a tail entity.

Knowledge Retrieval: Commonsense fact candidates are usually retrieved by matching the name (Zhou et al., 2018; Wu et al., 2020a): 1) all entity words appearing in the query X are denoted as a set $\{e_i\}$. 2) $\{e_i\}$ are adopted as keys to retrieve fact candidates from the base $\mathcal{K}$. If the head entity or the tail entity of a fact $k_i \in \mathcal{K}$ appears in $\{e_i\}$, then k_i will be added to the candidate set K.

#UsedFacts	1	2	3	[4,13]
Distribution	78.16%	16.68%	3.49%	1.65%
Accumulated	78.16%	94.85%	98.35%	100%

Table 1: Distribution of the number of facts used in a dialogue. Each dialogue has 77.65 fact candidates on average. The results are based on the adopted *Weibo* dataset (Wu et al., 2020a)

Empirical Observation: As shown in Table 1, in our adopted commonsense knowledge-aligned *Weibo* dataset (Wu et al., 2020a), although each dialogue has 77.65 fact candidates to select on average, most dialogues use no more than three facts.

[1]Assuming the length is L, the complexity of the self-attention used by Transformer $\propto L^2$.

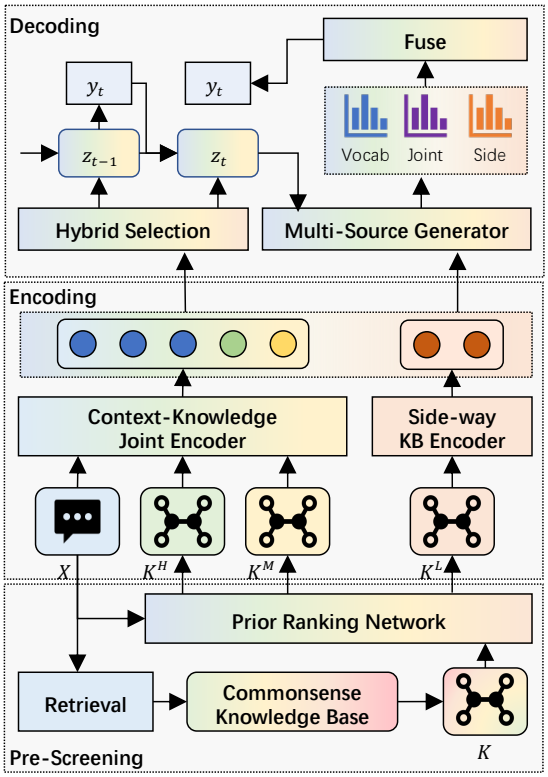

Figure 2: The overview of *SAKDP*.

In other words, the knowledge that can directly contribute to dialogue generation is always limited.

Problem Definition: Inspired by the above observation, as illustrated in Figure 2, SAKDP is a two-stage approach:

- *Pre-Screening Stage*: SAKDP first estimates the relevance of all fact candidates $\in K$; then, SAKDP ranks and clusters K into three knowledge sections: the high priority section K^H, the moderate priority section K^M, and the low priority section K^L;
- *Dialogue Generation Stage*: For balancing the performance and the efficiency, SAKDP uses section-aware methods to generate the response conditioned on the query X and three knowledge sections: $P(Y|X, K^H, K^M, K^L)$.

2.2 Pre-Screening Stage

The relevance between a fact candidate $k_i \in K$ and the entire dialogue context (X, Y) can be various. Obviously, only highly relevant facts can contribute to dialogue generation. To this end, we propose a prior ranking network *PriorRanking* to estimate the relevance .

Ranking Score: Our prior ranking network *PriorRanking* leverages the great potential of PLMs. We adopt BERT ($BERT^R$) to estimate the relevance score $r_i \in (0, 1)$ for each fact candidate k_i:

$$r_i = \theta(\mathbf{W_R}(BERT^R([CLS], X, [SEP], \sigma(k_i)))$$
$$\sigma(k_i) = [CSK], e_{head,i}, e_{rel,i}, e_{tail,i} \quad (1)$$

where θ is $sigmoid$ function, $\mathbf{W_R} \in \mathbb{R}^{1\times dim}$ is a learn-able parameter, $\sigma(k_i)$ linearizes the fact k_i into a plain text. It is worth noting that $BERT^R$ outputs the representation at $[CSK]$[2].

Contrastive Learning: The duty of *PriorRanking* is to give a higher score to a more relevant fact and a lower score to a less relevant fact. Thus, the training follows the idea of contrastive learning (Wu et al., 2020b). Given a training pair (X, Y), we first construct a set of contrastive pairs:

- *Positive:* We first select a positive subset K^+ from K. For each $k^+ \in K^+$, its head e^+_{head} must appear in X/Y and its tail e^+_{tail} must appear in the another Y/X at the same time.
- *Negative:* The remaining $K^- = K - K^+$ are negative samples, where each $k^- \in K^-$ is adopted by X but discarded by Y.
- *Contrastive Pairs:* For each $k^+ \in K^+$, we generate n contrastive pairs by sampling n $k^-_i \in K^-$. Intuitively, in each contrastive pair, the positive k^+ is more relevant to (X, Y) than the negative k^-.

Subsequently, we train *PriorRanking* by forcing it to give at least $m = 0.3$ higher score to the more relevant $k+$ than the less relevant $k-$:

$$\mathcal{L}_{Rank} = \frac{1}{n|K^+|}\sum_{k^+}\sum_{k^-_i}^{n} max(0, (m - r_{k^+} + r_{k^-_i})) \quad (2)$$

After the training, we can use the scores outputted by *PriorRanking* to estimate the relevance and rank candidates.

The Criteria of Clustering: The proposed *PriorRanking* network can not access the posterior information (i.e., the ground-truth response), and thus it cannot make a completely accurate prediction.

[2] $[CSK]$ and $[SEP]$ are two special symbols used by BERT, $[CSK]$ is a special symbol to separate linearized facts.

Consequently, rather than strictly sorting the fact candidates $\in K$ using the estimated relevance score r, we cluster fact candidates into coarse-grained ranked sections. We strictly distinguish each section's relevance label (order) but do not distinguish the relevance labels of knowledge fact candidates in each section. This methodology can balance the need for ranking and fault tolerance.

Specifically, depending on the estimated r_i, each fact k_i can be placed into the high priority section K^H, the moderate priority section K^M, or the low priority section K^L. Based on the empiricism (see Table 1) and Zipf's law (Zipf, 1949), we assume that the number of fact candidates will decrease with the relevance increasing. In other words, $|K^H| << |K^M| << |K^L|$. As an empirical procedure, the number of facts in each section will be determined by the following empirical study in the experiment.

2.3 Dialogue Generation Stage

To balance performance and efficiency, SAKDP uses knowledge of different sections via different strategies (i.e., section-aware).

2.3.1 Section-Aware Encoding

Although PLMs are powerful, they consume massive computation resources and always limit the input length. Thanks to the pre-screening stage, we can use two different section-aware encoding strategies to alleviate such issues:

1. *Context-Knowledge Joint Encoder:* To deepen the context-knowledge interaction, we use the pre-trained BERT to jointly encode the query X with the knowledge in the high/moderate priority section K^H/K^M. As the number of facts in K^H/K^M is limited, the introduction of BERT only costs an affordable expense;

2. *Side-way Knowledge Encoder:* Facts in the low priority section K^L may also contribute to the dialogue generation. As the number of included facts is significantly larger, using PLMs is not cost-effective; thus, we use a separate but lightweight non-pre-trained encoder.

Context-Knowledge Joint Encoder: The input of the context-knowledge joint encoder is given by:

$$
\begin{aligned}
H &= [CLS], X, [SEP], Pr^H, T^H, Pr^M, T^M \\
T^{H/M} &= \sigma^{H/M}(k_1^{H/M}), \cdots, \sigma^{H/M}(k_{l_{K^{H/M}}}^{H/M}) \\
\sigma^{H/M}(k) &= [HC/MC], e_{head}, e_{rel}, e_{tail}
\end{aligned} \tag{3}
$$

where $T^{H/M}$ is the linearized $K^{H/M}$, $\sigma^{H/M}(k)$ linearizes a fact k into the text with priority-aware structural labels (i.e. $[HC]$ and $[MC]$). Inspired by (Zhou et al., 2021a) that use some tips to investigate the inherent ability of PLMs, two tips Pr^H and Pr^M are designed to hint the model about the differences between two sections. Pr^H refers to '*The following knowledge facts are highly relevant to the left query:*'; Pr^M refers to '*Besides, the following knowledge facts may also be relevant:*'. Finally, the context-knowledge representations (memories) are given with the BERT encoder $BERT^J$:

$$
(\mathbf{h_{CLS}}, \mathbf{h_1}, ...\mathbf{h_{l_H}}) = \mathbf{H} = BERT^J(H) \tag{4}
$$

Side-way Knowledge Encoder: The lightweight side-way knowledge encoder is based on the non-pretrained Transformer network :

$$
(\mathbf{k_1^L}, ...\mathbf{k_{l_{K^L}}^L}) = \mathbf{K^L} = Trans^L(T^L) \tag{5}
$$

where T^L is the linearized K^L using the σ (Eq 1).

2.3.2 Decoding

We use a GRU to update the decoding state, a *Hybrid Selection* network to select knowledge, and a *Multi-Source Generator* to generate the response.

States Updating: At each time step t, the current decoding state $\mathbf{z_t}$ is updated by a GRU network:

$$
\mathbf{z_t} = GRU(\mathbf{z_{t-1}}, \mathbf{y_{t-1}}, \mathbf{s_t}) \tag{6}
$$

where $\mathbf{y_{t-1}}$ is the embedding of the last token.

Hybrid Selection: At time t, we use attention function $\alpha^{H/L}$ (Luong et al., 2015) to select a contextually relevant knowledge $\mathbf{s_t} = [\mathbf{s_t^H}; \mathbf{s_t^{K_L}}]$ from the context-knowledge memory $\mathbf{H}$ and the side-way knowledge memory $\mathbf{K^L}$:

$$
\mathbf{s_t^H} = \sum_i \frac{\exp(\alpha^H(\mathbf{z_{t-1}}^\mathbf{T}\mathbf{W_A^H}\mathbf{h_i}))}{\sum_j \exp(\alpha^H(\mathbf{z_{t-1}}^\mathbf{T}\mathbf{W_A^H}\mathbf{h_j}))}\mathbf{h_i} \tag{7}
$$

$$
\mathbf{s_t^{K^L}} = \sum_i \frac{\exp(\alpha^{K^L}(\mathbf{z_{t-1}}^\mathbf{T}\mathbf{W_A^{K^L}}\mathbf{k_i^L}))}{\sum_j \exp(\alpha^{K^L}(\mathbf{z_{t-1}}^\mathbf{T}\mathbf{W_A^{K^L}}\mathbf{k_j^L}))}\mathbf{k_i^L} \tag{8}
$$

Multi-Source Generator: The probability of the next token $P_t(y_t = w)$ is given by:

$$p_{V,t}P_{V,t}(w) + p_{H,t}\sum_{h_i=w} P_{H,t}(h_i) + p_{K^L,t}\sum_{k_i^L=w} P_{K^L,t}(k_i^L)$$

$$p_{V,t}, p_{H,t}, p_{K^L,t} = Softmax(\mathbf{W_P z_t}) \quad (9)$$

where $\mathbf{W_P} \in \mathbb{R}^{3\times dim}$. The *vocabulary* probability $P_{V,t}$, the *context-knowledge* copy probability $P_{H,t}(h_i)$, and the *side-way* copy probability $P_{K^L,t}(k_i^L)$ are given by:

$$P_{V,t} = Softmax(\mathbf{W_V z_t})$$

$$P_{H,t}(h_i) = \frac{\exp(\alpha^H(\mathbf{z_t}^\mathbf{T}\mathbf{W_A^H h_i}))}{\sum_j \exp(\alpha^H(\mathbf{z_t}^\mathbf{T}\mathbf{W_A^H h_j}))} \quad (10)$$

$$P_{K^L,t}(k_i^L) = \frac{\exp(\alpha^{K^L}(\mathbf{z_t}^\mathbf{T}\mathbf{W_A^{K^L} k_i^L}))}{\sum_j \exp(\alpha^{K^L}(\mathbf{z_t}^\mathbf{T}\mathbf{W_A^{K^L} k_j^L}))}$$

where the computation of $P_{H,t}(h_i)$ and $P_{K^L,t}(k_i^L)$ reuse the parameters of Equation 7 and Equation 8.

Learning Objective: The training optimizes the following objective:

$$\mathcal{L}_{dialog} = -\sum_t \log(P_t(y_t)) \quad (11)$$

3 Experiment

3.1 Settings

Dataset: We evaluate models on *Weibo* dataset (Wu et al., 2020a), which collected more than 1M dialogues from the largest Chinese SNS Weibo and collected commonsense knowledge facts from the ConceptNet (Speer et al., 2017). The training/validation/test set has 1,019,908/56,661/56,661 dialogues, the commonsense base has 696K facts, 27K entities, and 26 relations.

Comparison Models: We compare *SAKDP* with several representative models: (1) **Seq2Seq**: The widely-used Seq2Seq (Sutskever et al., 2014) + Attention (Luong et al., 2015) model; (2) **Copy**: A Seq2Seq variant that can copy words from the query (See et al., 2017); (3-4) **BERT2Seq, BERTCopy**: We changed the encoder of **Seq2Seq** and **Copy** to the BERT encoder (Cui et al., 2021b). (5) **CCM**: It uses commonsense knowledge via the graph attention. (Zhou et al., 2018); (6) **ConKADI**: It proposes a felicitous knowledge selection mechanism to use commonsense knowledge (Wu et al., 2020a); (7) **ConceptFlow:** It can use multi-hop commonsense knowledge facts to enhance the dialogue response generation. (Zhang et al., 2020a); (8) **GOKC**: One of the current SOTA knowledge-grounded approach (Bai et al., 2021).

We use the official codes [3] for baselines except for Seq2Seq, Copy, BERT2Seq, BERTCopy, which use our PyTorch implementations [4]. Models adopt the following settings: word-level tokenization, 2-layer encoder/decoder, 512-d(imensional) GRU/LSTM or 512-d 8H Transformer, 200-d word embedding, 30K vocab, 100-d entity embedding, 32 batch size, Adam optimizer, 1e-4 learning rate, beam-search decoding (beam width =10) if a model supports. For BERT modules, we adopt a widely-used Chinese BERT *hfl/chinese-bert-wwm-ext* (102M parameters, 768d, 12L, 8H, 21,128 subwords (Cui et al., 2021b). Consequently, for BERT2Seq, BERTCopy, and SAKDP (both the ranking and generation network), the optimizer is changed to AdamW, the tokenization adopts the default tokenizer of *hfl/chinese-bert-wwm-ext* , the learning rate of BERT module is set to 1e-5 (other modules keep unchanged). The training adopts the early stopping mechanism. The training will be stopped if the loss on the validation set increases in two consecutive epochs.

For all commonsense knowledge-grounded baselines, commonsense knowledge candidates are provided by the original dataset. Thus, the ground-truth commonsense facts are provided during the test by default (but no label to indicate which are gold facts). In our approach, the facts are selected by our $PriorRanking$ network before the dialogue generation.

Metrics: We use both automatic evaluation and human annotation to evaluate models. In automatic evaluation, the responses generated by word-level models are re-tokenized by the BERT tokenizer. As for automatic metrics, following (Wu et al., 2020a), we adopt F1, Rouge-1/2/L, Bleu-1/2/3/4, Em-A/G/X to evaluate the relevance, and we adopt DIST-1/2, and Ent1/2/3/4 to evaluate the informativeness and diversity. In addition, we also calculate the geometric mean score overall metrics.

[3] Some baseline models tend to generate UNK tokens, bringing very unacceptable results. Considering this, we have additionally masked the generation of UNK for these models.

[4] The code of SAKDP can be find in :https://github.com/pku-sixing/COLING2022-SAKDP

In human evaluation, following (Zhou et al., 2018), we conducted the pair-wise comparison between the response generated by our approach and the baseline. The quality of generated responses is judged with three criteria: 1) Fluency: the fluency of a generated response without considering the context; 2) Appropriateness: the relevance and logic between the query and the generated response; 3) Informativeness: the quality/novelty/correctness of information provided in the generated response.

3.2 Knowledge Pre-Screening Study

Top-k, %	@1	@3	@5	@10	@20	@40	@100
$\text{Precision}_{\mathbf{Ours}}$	35.2	23.3	17.5	10.9	6.24	3.52	2.25
$\text{Precision}_{\mathbf{Rand}}$	2.24	2.20	2.18	2.17	2.18	2.17	2.17
$\text{Recall}_{\mathbf{Ours}}$	28.5	53.8	66.3	81.2	92.7	98.3	99.9
$\text{Recall}_{\mathbf{Rand}}$	1.81	5.32	8.79	17.4	34.1	60.1	94.8
$\text{Micro-F1}_{\mathbf{Ours}}$	31.5	32.5	27.7	19.2	11.7	6.80	4.41
$\text{Micro-F1}_{\mathbf{Rand}}$	2.00	2.73	3.32	3.85	4.10	4.18	4.24

Table 2: The ranking performance (*PriorRanking* vs. random). We report scores on 7 positions (i.e., k). Meanwhile, $\mathbf{MRR}_{\mathbf{Ours}}$=0.511, $\mathbf{MRR}_{\mathbf{Rand}}$=0.09, the max/avg k is 151/77.65.

We first evaluate the ranking performance of our ranking network *PriorRanking*. In prior works, there is no knowledge pre-screening process before the dialogue generation; thus, the knowledge selection totally relies on the internal selection of the end2end model. However, as shown in Table 2, if a model uses knowledge facts of random order, the internal knowledge selection would be pretty challenging to select relevant knowledge. Without the pre-screening, prior works are also blind if some knowledge candidates must be discarded for efficiency. Fortunately, we find the ranking performance of *PriorRanking* is quite acceptable. Although there are 77.65 candidates on average, the precision@1 is 35.2%, and the recall@3 is more than half. It indicates that SAKDP can efficiently estimate the relevance of knowledge candidates.

The Criteria of Clustering: Now we can empirically determine the division of three knowledge sections based on the statistics (Table 1) and the evaluation results (Table 2): 1) *High Priority Section* K^H: Considering that 98.35% responses use no more than 3 facts and the Micro-F1 achieves the highest at top-3, we select top-3 candidates to K^H; 2) *Moderate Priority Section* K^M: We find the top-10 position is a sweet point, where more than 80% of gold candidates can be recalled and the precision is still acceptable. Thus, K^M selects the next 7 candidates (i.e., [4,10]); 3) *Low Priority Section* K^L: Finally, K^L selects the next 30 candidates (i.e., [11,40]) because the top-40 recall has achieved 98%. The remaining facts are discarded because the long-tail issue is significant; we think it is not a good trade to increase the recall continually.

3.3 Automatic Evaluation

Results: As reported in Table 3, SAKDP has achieved leadership in most metrics and performed the second-best in almost the remaining metrics. In the overall geometric mean score, SAKDP outperforms various baselines by notable margins. Compared to other knowledge-grounded models, the most notable advantages come from F1, Rouge, and Bleu, showing the responses generated by SAKDP are fluent and coherent. Comparing Seq2Seq/Copy vs. BERT2Seq/BERTCopy, although notable improvements are achieved in other metrics, the introduction of the BERT may impact the diversity and the informativeness. We think the reason is the adopted subword-level tokenization. But fortunately, with the proposed *Hybrid Selection* and *Multi-Source Generation*, SAKDP still has notable advantages compared to the baselines except ConKADI. Meanwhile, although BERT is powerful enough, BERTCopy is not enough to compete against the ConKADI/GOKC, demonstrating incorporating commonsense knowledge is essential in the arena of PLMs.

Non-BERT SAKDP: SAKDP outperforms the knowledge-grounded ConKADI and GOKC but has more parameters. To better exhibit our advantage, we evaluate the efficiency-oriented SAKDP_{Effi}, which replaces the BERT encoder by a lightweight 2-layer Transformer. Clearly, SAKDP_{Effi} still can outperform GOKC/ConKADI even with only 59% parameters (49M vs. 29M). This indicates the advantage of SAKDP does not fully rely on BERT.

Fully-Joint SAKDP: We also evaluate the performance-oriented SAKDP_{Perf} that uses our context-knowledge joint encoder to encode all knowledge. As a result, SAKDP_{Perf} uses fewer parameters because the side-way knowledge encoder is removed. Compared to the standard SAKDP, the overall geomean score increased by 1.7%, but its training time sharply increased by 81%[5]. This re-

[5]On average, SAKDP_{Effi}/SAKDP/SAKDP_{Perf} costs 0.21/0.33/0.60s per training step.

Model (#Parameters)	F1	Rouge-1/2/L	Bleu-1/2/3/4	Embed-A/G/X	DIST-1/2	Ent-1/2/3/4	**Mean**
Seq2Seq	17.21	18.5/3.2/12.9	12.3/5.2/2.4/1.2	0.878/0.681/0.655	0.33/3.61	4.80/6.99/8.45/9.54	3.82
Copy	17.34	18.6/3.5/13.0	12.4/5.5/2.7/1.4	0.877/0.679/0.656	**0.59**/**8.94**	5.08/7.60/9.19/10.3	4.33
BERT2Seq	18.29	19.5/3.6/13.8	16.9/7.4/3.6/1.9	0.886/0.679/0.661	0.20/2.04	4.78/6.77/7.96/8.83	4.03
BERTCopy	19.24	20.4/4.3/13.9	18.7/ 8.8/4.5/**2.5**	0.897/0.681/0.666	0.37/7.09	5.07/7.35/8.71/9.61	4.87
CCM(32M)	15.63	20.2/4.3/13.4	15.0/6.9/3.2/1.6	0.875/0.690/0.659	0.24/2.62	3.95/5.72/6.76/ 7.41	3.87
ConKADI(49M)	18.98	20.9/4.4/14.4	17.8/8.3/3.8/1.8	0.885/0.677/0.662	**0.41**/**10.8**	**5.55/8.77/10.8/11.9**	5.01
ConceptFlow	19.32	24.0/5.8/16.2	18.2/8.9/4.3/2.3	0.874/0.698/0.662	0.26/3.51	4.33/6.38/7.59/8.35	4.51
GOKC(49M)	**20.93**	**24.3/7.0/16.6**	**19.6/10.7/4.9**/2.1	**0.900**/**0.720/0.698**	0.31/5.52	4.39/6.73/8.38/9.49	4.96
SAKDP(128M)	**23.64**	**25.5/7.4/17.9**	**22.7/12.4/6.8/3.8**	**0.902**/**0.705/0.688**	**0.41**/8.47	**5.28/8.03/9.71/10.8**	**5.82**
SAKDP$_{Effi}$(29M)	21.07	22.8/6.1/16.3	18.6/9.7/5.1/2.8	0.893/0.692/0.676	0.42/6.67	5.19/7.68/9.16/10.1	5.17
SAKDP$_{Perf}$(122M)	24.07	26.3/8.3/18.9	20.5/11.7/6.5/3.8	0.896/0.705/0.688	0.50/10.2	5.40/8.36/10.1/11.2	5.92

Table 3: Automatic evaluation results, **black**/**blue** is the first/second best (excluding SAKDP$_{Perf}$ and SAKDP$_{Effi}$). The last column reports the geomean of previous scores, showing the overall performance.

Model	F1	RouL	Bleu4	EmG	DI2	Ent4	Mean
Full	23.64	17.91	3.83	0.705	8.47	10.80	5.82
w/o Ranking	20.83	15.92	2.96	0.687	9.88	10.95	5.40
w/o BERT	21.07	16.34	2.80	0.692	8.47	10.10	5.17
w/o Joint	22.96	17.20	3.47	0.686	6.99	10.27	5.56
w/o Tips	23.31	18.33	3.63	0.702	9.46	10.85	5.77
w/o MSCopy	22.44	16.98	3.17	0.703	2.59	9.36	4.89

Table 4: Ablation Study. 'w/o' denotes 'without'.

sult shows our section-aware strategy can balance performance and efficiency.

3.3.1 Ablation Study

Table 4 verifies the contribution of each module. 1) In *w/o Ranking*, we remove the $PriorRanking$ and randomly select facts for three sections. Notably, there is a significant performance regression, despite using the same number of facts and the same generation models. It means $PriorRanking$ can effectively estimate the relevance of fact candidates without the posterior information; 2) BERT is quite helpful in dialogue generation. After replacing the BERT with a non-pre-trained 2-layer Transformer (*w/o BERT*), we can find a notable performance decrease. 3) In Equation 3, we use two tips Pr^H and Pr^M to hint the model about the difference between the two sections. The performance decreases after removing them (*w/o Tips*), demonstrating the necessity to distinguish such two sections.4) Jointly encoding the query and the relevant knowledge can indeed deepen the context-knowledge infusing. In '*w/o Joint*.', we use a separate BERT to encode K^H and K^M; as expected, the performance is worse; 5) We use *Multi-Source Generator* to enhance the diversity. Without it (*w/o MSCopy*), the diversity is significantly decreased.

3.3.2 Knowledge Encoding Analysis

To further investigate the characteristics of knowledge sections, we test each knowledge section with

Strategy		F1	RouL	Bleu4	EmG	DI2	Ent4	Mean	t(s)
SAKDP		23.64	17.91	3.83	0.705	8.47	10.80	5.82	0.33
K^H	*Joint*	22.33	16.57	3.46	0.697	7.67	10.54	5.57	0.20
K^H	*BERT*	21.64	16.13	3.21	0.694	7.92	10.43	5.44	0.26
K^H	*Trans*	20.73	15.23	2.95	0.689	6.99	9.98	5.20	0.19
K^M	*Joint*	20.66	15.71	2.90	0.688	8.43	10.43	5.25	0.21
K^M	*BERT*	20.19	14.85	2.83	0.685	7.84	10.15	5.17	0.26
K^M	*Trans*	19.90	14.69	2.70	0.685	7.80	10.14	5.08	0.19
K^L	*Joint*	20.30	15.37	2.80	0.684	9.81	10.89	5.27	0.40
K^L	*BERT*	19.79	14.62	2.66	0.684	7.78	10.05	5.06	0.58
K^L	*Trans*	19.20	14.28	2.52	0.680	7.65	9.96	4.92	0.23

Table 5: Performance comparisons among different encoding strategies. t is the average time of each training step. The first column is the adopted section, the second is the adopted encoder: 1) *Joint*: use the BERT-based Context-Knowledge Joint Encoder to jointly encode the dialogue and the knowledge; 2) *BERT:* use a separate BERT to encode; 3) *Trans:* use the separate non-BERT Side-way Knowledge encoder to encode.

three encoding strategies. As reported in Table 5: 1) Using a joint BERT to jointly encode the context and the knowledge is better than using two separated BERTs, bringing more improvement than replacing a non-pre-trained Transformer with a pre-trained BERT. It shows the interaction between the context and the knowledge is necessary. Meanwhile, we can find using a joint BERT is more efficient when implemented by PyTorch; this is because of the higher parallelism; 2) On the whole, while the number of facts in a higher priority section is significantly less, the performance is better. In addition, even only using three fact candidates (K^H+*Joint*), our approach still significantly outperforms baselines. Such two factors indicate our $PriorRanking$ is very effective; 3) The full SAKDP has the best performance, but the training is even faster than K^L+*BERT/Trans*. This shows our standard SAKDP is very efficient.

3.4 Human Evaluation

Similar to (Zhou et al., 2018), we employed 3 well-educated volunteers to evaluate 5 baselines, where each group has 200 sampled cases. **Agreements:** The average 2/3 agreements (at least 2 judges gave the same label) is 97.4%, the average 3/3 agreement is 62.2%, and the Fleiss'Kappa is 0.43.

Table 6 reports the percentage that SAKDP wins its competitor. It can be seen that our approach significantly outperforms baseline models. Interestingly, the baselines with relatively better performance do not infuse external knowledge. This is because when using knowledge, due to the lack of enough context-knowledge fusing ability, the fluency of such dialogues is poor, which may affect human evaluation. Thanks to the introduction of BERT and context-knowledge joint encoding, our approach does not suffer from this.

%	Flu.			Appro.			Info.		
SAKDP vs.	*Lose*	Tie	*Win*	*Lose*	Tie	*Win*	*Lose*	Tie	*Win*
Seq2Seq	25.3	21.5	**53.2**	26.2	5.5	**68.3**	18.8	4.3	**76.8**
BERTCopy	35.2	18.3	**46.5**	39.0	10.0	**51.0**	40.0	9.5	**50.5**
ConceptFlow	19.8	11.7	**68.5**	17.5	4.0	**78.5**	14.0	7.8	**78.2**
ConKADI	19.8	11.8	**68.4**	22.2	4.7	**73.2**	25.2	51.1	**69.7**
GOKC	7.8	14.8	**77.4**	9.2	7.2	**83.4**	8.2	5.3	**86.5**

Table 6: Human evaluation. *Win/Tie/Lose* denotes the ratio that our SAKDP has wined, tied with, or lost to the corresponding baseline, respectively. **Score** is significantly better (sign test, p-value < 0.005).

Case study: We report two cases in Table 7. In the first case, we can find that 1) responses have referred to four facts in our commonsense base in total. It can be seen that our $PriorRanking$ network has the ability to estimate the relevance between a fact candidate and the dialogue context only using the prior query. Three of them are included by the high-priority section K^H; the remaining one is also included by the moderate-priority section K^M; 2) Thanks to the context-knowledge joint encoding, compared to other models, the response generated by our model is not only fluent but also rational. ConKADI and GOKC irrationally used commonsense facts. The second is a case in our human evaluation. 1) Seq2Seq and BERTCopy have generated a fluent response, but not appropriate and informative enough; 2) ConKADI and GOKC generated irrational responses once again; 3) The response generated by our SAKDP is still the best.

4 Related Work

Knowledge-Grounded Methods: Traditional models (Sutskever et al., 2014) tend to generate boring responses (Li et al., 2016). Knowledge-grounded methods can address this issue by infusing external knowledge (Yu et al., 2020; Wu et al., 2021a, 2022). Roughly, knowledge-grounded works can use the text-based knowledge (Dinan et al., 2019; Ren et al., 2020; Zhan et al., 2021; Meng et al., 2021), the structural knowledge (Bai et al., 2021; Wu et al., 2021b), and the multi-modal data (Wang et al., 2021).

Commonsense knowledge can contribute to the semantic understanding (Young et al., 2018), knowledge reasoning (Liu et al., 2019), topic transition (Zhong et al., 2021), improving the diversity (Wu et al., 2020a; Speer et al., 2017). To reduce the computational cost and improve the knowledge relevance, text knowledge-grounded works always follow a two-stage paradigm (Dinan et al., 2019): 1) A pre-screening stage to *explicitly* select one knowledge text from candidates; 2) A generation stage to generate responses with an internal fine-grained select knowledge. Unlike such works, most commonsense knowledge-grounded works ignored the pre-screening and entirely relied on the internal selection. It may not be a severe problem if a model does not use PLMs. However, as mentioned, if we try to integrate PLMs, the cost of knowledge encoding/accessing is always unacceptable. Different from them, to introduce a PLM as an encoder. We propose a well-designed explicit pre-screening stage to rank and filter out facts. In addition, unlike prior works that always employ separate encoders to encode the dialogue query and the knowledge, respectively. *SAKDP* jointly encodes the query and the relevant knowledge, bringing deeper context-knowledge infusing and interaction.

Pretrained Models: PLMs have shown striking potential in NLP tasks (Li et al., 2021b). Dialogue generation can also benefit from PLMs because PLMs have extraordinary semantic understanding and generation abilities (Wang and et al., 2020; Gu et al., 2021). Researchers have shown that PLMs can prompt the text knowledge-grounded generation (Zhao et al., 2020; Cui et al., 2021a), to investigate the implicit knowledge hidden in PLMs (Zhou et al., 2021a; Sun et al., 2021; Zhou et al., 2021c). However, as mentioned, using PLMs to promote the structural knowledge-grounded dialogue gener-

#1 Query	猫真的超可爱！Cats are really cute!
Seq2Seq	我也觉得超喜欢的. I really like it, too
ConKADI	真的是被萌化了要玩跑粉. It's really lovely. I want to play running powder.
GOKC	我也要跑了以后你这么可爱的猫. I'm going to run, too. In the future, you're such a lovely cat.
BERTCopy	是啊，超级可爱，但是我觉得它好可爱. Yes, it's super cute, but I think it's so cute.
Our SAKDP	我也觉得超级可爱，但是它们跑的很快. I also think they are super cute, but they run fast
Relevant Knowledge	**(猫Cat,CapbableOf,跑Run) (真的really,RelatedTo,超级Super)** **(猫Cat,CapbableOf,玩Play) (萌Lovely,MotivatedBy,可爱Cute)...**
#2 Query	撸猫容易戒猫难！改天一起撸猫呀。 It's easier to cuddle cats than to quit! Let's cuddle cats some other day.
Seq2Seq	(TLL) 我也想撸猫。 I'd like to cuddle cats.
ConKADI	(LLL) 好呀好呀,我也想撸只撸狗鱼。 Well, well, I also want to cuddle cuddle a dog or a fish.
GOKC	(LLL) 宝贝你要是来了你都比你. Baby, if you come, you're better than you.
BERTCopy	(LLL) 好的，我家猫也是这样的，但是我觉得它很可爱. OK, so is my cat, but I think it's cute.
Our SAKDP	(LLL)好的，我们一起撸猫宝贝OK, let's cuddle baby cats.
Relevant Knowledge	**(猫Cat,CapbableOf,喵Meow) (猫Cat,Desires,睡Sleep) (喵Meow,UsedFor,猫Cat)** **(猫Cat,IsA,宝贝Baby) (狗Dog,NotDesires,猫Cat) (猫Cat,HasProperty,可爱Cute)...**

Table 7: Case Study. The red-colored entities appear in the high-priority section K^H , the blue-colored entities appear in the moderate-priority section K^M. The second case is selected from our human evaluation, and we report the human-annotated scores of one annotator. Specifically, for each baseline, we show the fluency, appropriateness, and informativeness labels, respectively, at the beginning. W/T/L denotes the baseline has wined SAKDP, tied with SAKDP, lost to SAKDP, respectively.

ation still faces many challenges (Zhao et al., 2021; Li et al., 2021a). This paper focuses on investigating auto-encoder PLMs (Vaswani et al., 2017) (such as BERT and RoBERTa) to encode. We leave using auto-regressive PLMs (such as GPTs) and Seq2Seq PLMs (such as BART, MASS) as future work because 1) such PLMs are unsuitable for introducing more flexible knowledge selection mechanisms, especially the auto-regressive PLMs; 2) Our goal is to reach the balance between performance and efficiency; such Seq2Seq PLMs have more complicated network structures and more parameters; 3) For flexibility and applicability, we hope SAKDP can also support non-pre-trained modules.

5 Conclusion

In this paper, we present an efficient two-stage section-aware commonsense knowledge-grounded dialogue generation framework SAKDP. We propose a ranking network to cluster knowledge candidates into different priority sections and adopt different use schemes. Subsequently, SAKDP can benefit from both BERT and commonsense knowledge with a balance of efficiency and performance. Extensive experiments demonstrate the performance leadership of our approach.

In the future, we will continue to promote the integration of PLMs and knowledge: 1) we will continue to improve the efficiency of knowledge-grounded and PLM-based dialogue response generation; 2) we will explore more solutions to select knowledge in the pre-screening stage, for example, using GNNs; 3) Current SAKDP is not fully PLM-based because it uses a GRU decoder. We will also try to explore the option of fully PLM-based solutions.

Ethical Considerations

This work did not release any newly created dataset or ethical statement. The first possible ethical issue depends on how other users use our method, i.e., the adopted dataset, and the user scenario. The next possible issue is that bias may be introduced by the adopted PLMs and knowledge. As for this technical work itself, there is no ethical issue.

References

Jiaqi Bai, Ze Yang, Xinnian Liang, Wei Wang, and Zhoujun Li. 2021. Learning to copy coherent knowledge for response generation. In *AAAI*.

Hongshen Chen, Xiaorui Liu, Dawei Yin, and Jiliang Tang. 2017. A Survey on Dialogue Systems: Recent Advances and New Frontiers. *SIGKDD*, 19.

Leyang Cui, Yu Wu, Shujie Liu, and Yue Zhang. 2021a. Knowledge enhanced fine-tuning for better handling unseen entities in dialogue generation. In *EMNLP*, pages 2328–2337, Online and Punta Cana, Dominican Republic.

Yiming Cui, Wanxiang Che, Ting Liu, Bing Qin, and Ziqing Yang. 2021b. Pre-training with whole word masking for chinese bert.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: pre-training of deep bidirectional transformers for language understanding. In *NAACL-HLT*.

Emily Dinan, Stephen Roller, Kurt Shuster, Angela Fan, Michael Auli, and Jason Weston. 2019. Wizard of wikipedia: Knowledge-powered conversational agents. In *ICLR*.

Xiaodong Gu, KangMin Yoo, and JungWoo Ha. 2021. Dialogbert:discourse-aware response generation via learning to recover and rank utterances. In *AAAI*.

Matthew Henderson, Iñigo Casanueva, Nikola Mrksic, Pei-Hao Su, Tsung-Hsien Wen, and Ivan Vulic. 2020. Convert: Efficient and accurate conversational representations from transformers. In *Findings of the Association for Computational Linguistics: EMNLP*, volume EMNLP 2020 of *Findings of ACL*.

Jiwei Li, Michel Galley, Chris Brockett, and et al. 2016. A diversity-promoting objective function for neural conversation models. In *NAACL*.

Junyi Li, Tianyi Tang, Wayne Xin Zhao, Zhicheng Wei, Nicholas Jing Yuan, and Ji-Rong Wen. 2021a. Few-shot knowledge graph-to-text generation with pretrained language models. In *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*.

Junyi Li, Tianyi Tang, Wayne Xin Zhao, and Ji-Rong Wen. 2021b. Pretrained language models for text generation: A survey. *CoRR*, abs/2105.10311.

Zhibin Liu, Zheng-Yu Niu, Hua Wu, and Haifeng Wang. 2019. Knowledge Aware Conversation Generation with Explainable Reasoning over Augmented Graphs. In *EMNLP*, pages 1782–1792.

Thang Luong, Hieu Pham, and Christopher D. Manning. 2015. Effective approaches to attention-based neural machine translation. In *EMNLP*.

Chuan Meng, Pengjie Ren, Zhumin Chen, Zhaochun Ren, Tengxiao Xi, and Maarten de Rijke. 2021. Initiative-aware self-supervised learning for knowledge-grounded conversations. In *SIGIR*, pages 522–532.

Alec Radford, Jeff Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. 2019. Language models are unsupervised multitask learners.

Pengjie Ren, Zhumin Chen, Christof Monz, and et al. 2020. Thinking globally, acting locally: Distantly supervised global-to-local knowledge selection for background based conversation. In *AAAI*.

Stephen Roller, Emily Dinan, Naman Goyal, Da Ju, Mary Williamson, Yinhan Liu, Jing Xu, Myle Ott, Eric Michael Smith, Y-Lan Boureau, and Jason Weston. 2021. Recipes for building an open-domain chatbot. In *EACL*.

Abigail See, Peter J. Liu, and Christopher D. Manning. 2017. Get To The Point: Summarization with Pointer-Generator Networks. In *ACL*.

Robyn Speer, Joshua Chin, and Catherine Havasi. 2017. Conceptnet 5.5: An open multilingual graph of general knowledge. In *AAAI*, pages 4444–4451.

Yu Sun, Shuohuan Wang, Shikun Feng, Siyu Ding, Chao Pang, Junyuan Shang, Jiaxiang Liu, Xuyi Chen, Yanbin Zhao, Yuxiang Lu, Weixin Liu, Zhihua Wu, Weibao Gong, Jianzhong Liang, Zhizhou Shang, Peng Sun, Wei Liu, Xuan Ouyang, Dianhai Yu, Hao Tian, Hua Wu, and Haifeng Wang. 2021. ERNIE 3.0: Large-scale knowledge enhanced pre-training for language understanding and generation. *CoRR*, abs/2107.02137.

Yu Sun, Shuohuan Wang, Yu-Kun Li, Shikun Feng, Xuyi Chen, Han Zhang, Xin Tian, Danxiang Zhu, Hao Tian, and Hua Wu. 2019. ERNIE: enhanced representation through knowledge integration. *CoRR*, abs/1904.09223.

Ilya Sutskever, Oriol Vinyals, and Quoc V. Le. 2014. Sequence to sequence learning with neural networks. In *NIPS*, pages 3104–3112.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. *CoRR*, abs/1706.03762.

Shuhe Wang, Yuxian Meng, Xiaofei Sun, Fei Wu, Rongbin Ouyang, and et al. 2021. Modeling text-visual mutual dependency for multi-modal dialog generation. *CoRR*, abs/2105.14445.

Yida Wang and et al. 2020. A large-scale chinese short-text conversation dataset. In *NLPCC*.

Yida Wang, Pei Ke, Yinhe Zheng, Kaili Huang, Yong Jiang, Xiaoyan Zhu, and Minlie Huang. 2020. A large-scale chinese short-text conversation dataset. In *NLPCC*, pages 91–103.

Thomas Wolf, Victor Sanh, Julien Chaumond, and Clement Delangue. 2019. Transfertransfo: A transfer learning approach for neural network based conversational agents. *CoRR*, abs/1901.08149.

Sixing Wu, Ying Li, Minghui Wang, Dawei Zhang, Yang Zhou, and Zhonghai Wu. 2021a. More is better: Enhancing open-domain dialogue generation via multi-source heterogeneous knowledge. In *EMNLP*, pages 2286–2300. Association for Computational Linguistics.

Sixing Wu, Ying Li, Dawei Zhang, Yang Zhou, and Zhonghai Wu. 2020a. Diverse and informative dialogue generation with context-specific commonsense knowledge awareness. In *ACL*, pages 5811–5820.

Sixing Wu, Minghui Wang, Ying Li, Dawei Zhang, and Zhonghai Wu. 2022. Improving the applicability of knowledge-enhanced dialogue generation systems by using heterogeneous knowledge from multiple sources. In *WSDM*, pages 1149–1157. ACM.

Sixing Wu, Minghui Wang, Dawei Zhang, Yang Zhou, and et al. 2021b. Knowledge-aware dialogue generation via hierarchical infobox accessing and infobox-dialogue interaction graph network. In *IJCAI*.

Zhuofeng Wu, Sinong Wang, Jiatao Gu, Madian Khabsa, and et al. 2020b. CLEAR: contrastive learning for sentence representation. *CoRR*, abs/2012.15466.

Ze Yang, Wei Wu, Can Xu, Xinnian Liang, Jiaqi Bai, Liran Wang, Wei Wang, and Zhoujun Li. 2020. StyleDGPT: Stylized response generation with pre-trained language models. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 1548–1559, Online. Association for Computational Linguistics.

Tom Young, Erik Cambria, Iti Chaturvedi, Hao Zhou, Subham Biswas, and Minlie Huang. 2018. Augmenting end-to-end dialogue systems with commonsense knowledge. In *AAAI*.

Wenhao Yu, Chenguang Zhu, Zaitang Li, Zhiting Hu, Qingyun Wang, Heng Ji, and Meng Jiang. 2020. A survey of knowledge-enhanced text generation. *CoRR*, abs/2010.04389.

Haolan Zhan, Hainan Zhang, Hongshen Chen, Zhuoye Ding, Yongjun Bao, and Yanyan Lan. 2021. Augmenting knowledge-grounded conversations with sequential knowledge transition. In *NAACL-HLT*.

Houyu Zhang, Zhenghao Liu, Chenyan Xiong, and Zhiyuan Liu. 2020a. Grounded conversation generation as guided traverses in commonsense knowledge graphs. In *ACL*, pages 2031–2043.

Yizhe Zhang, Siqi Sun, and et al. 2020b. DIALOGPT : Large-scale generative pre-training for conversational response generation. In *ACL*.

Zhengyan Zhang, Xu Han, Hao Zhou, Pei Ke, Yuxian Gu, Deming Ye, Yujia Qin, Yusheng Su, Haozhe Ji, Jian Guan, Fanchao Qi, Xiaozhi Wang, Yanan Zheng, Guoyang Zeng, Huanqi Cao, Shengqi Chen, Daixuan Li, Zhenbo Sun, Zhiyuan Liu, Minlie Huang, Wentao Han, Jie Tang, Juanzi Li, Xiaoyan Zhu, and Maosong Sun. 2021. Cpm: A large-scale generative chinese pre-trained language model. *AI Open*, 2:93–99.

Wenting Zhao, Ye Liu, Yao Wan, and Philip Yu. 2021. Attend, memorize and generate: Towards faithful table-to-text generation in few shots. In *Findings of the Association for Computational Linguistics: EMNLP 2021*, pages 4106–4117, Punta Cana, Dominican Republic. Association for Computational Linguistics.

Xueliang Zhao, Wei Wu, Can Xu, and et al. 2020. Knowledge-grounded dialogue generation with pre-trained language models. In *EMNLP*.

Peixiang Zhong, Yong Liu, Hao Wang, and Chunyan Miao. 2021. Keyword-guided neural conversational model. In *AAAI*, pages 14568–14576.

Hao Zhou, Tom Young, Minlie Huang, Haizhou Zhao, Jingfang Xu, and Xiaoyan Zhu. 2018. Commonsense Knowledge Aware Conversation Generation with Graph Attention. In *IJCAI*, pages 4623–4629.

Pei Zhou, Karthik Gopalakrishnan, and et al. 2021a. Think before you speak: Using self-talk to generate implicit commonsense knowledge for response generation. *CoRR*, abs/2110.08501.

Pei Zhou, Karthik Gopalakrishnan, Behnam Hedayatnia, Seokhwan Kim, and et al. 2021b. Commonsense-focused dialogues for response generation: An empirical study. In *SIGdial*, pages 121–132.

Wangchunshu Zhou, Dong-Ho Lee, Ravi Kiran Selvam, Seyeon Lee, and Xiang Ren. 2021c. Pre-training text-to-text transformers for concept-centric common sense. In *9th International Conference on Learning Representations, ICLR 2021, Virtual Event, Austria, May 3-7, 2021*. OpenReview.net.

George Kingsley Zipf. 1949. *Human Behaviour and the Principle of Least Effort: an Introduction to Human Ecology*. Addison-Wesley.

Using Multi-Encoder Fusion Strategies to Improve Personalized Response Selection

Souvik Das Sougata Saha Rohini K. Srihari
{souvikda, sougatas, rohini}@buffalo.edu
Department of Computer Science and Engineering, University at Buffalo, NY.

Abstract

Personalized response selection systems are generally grounded on *persona*. However, a correlation exists between persona and empathy, which these systems do not explore well. Also, when a contradictory or off-topic response is selected, *faithfulness* to the conversation context plunges. This paper attempts to address these issues by proposing a suite of fusion strategies that capture the interaction between persona, emotion, and entailment information of the utterances. Ablation studies on the Persona-Chat dataset show that incorporating emotion and entailment improves the accuracy of response selection. We combine our fusion strategies and concept-flow encoding to train a BERT-based model which outperforms the previous methods by margins larger than 2.3% on original personas and 1.9% on revised personas in terms of **hits@1** (top-1 accuracy), achieving a new state-of-the-art performance on the Persona-Chat dataset.

1 Introduction

Currently, most response selection systems tend to perform well in most cases (Gu et al., 2021a; Zhang et al., 2021b; Gu et al., 2019a, 2020a). However, these re-ranking systems have the poor capability to detect and evade contradictory responses. Responses selected by these systems often contradict previous utterances, and any form of contradiction may disrupt the flow of conversation. Previous research has attempted to incorporate persona while selecting (Gu et al., 2021b; Zhang et al., 2021a) or generating (Wu et al., 2021) responses to maintain consistency. Additionally, a correlation exists between persona with personality (Leary and Allen, 2011) , which influences empathy (Richendoller and Weaver III, 1994). Zhong et al. presented a multi-domain dataset collected from several empathetic Reddit threads contributing towards persona-based empathetic conversations. Nevertheless, no one has studied the emotion-persona interplay in data presented in a more natural form. Figure 1 depicts situational emotion sometimes needs more preference than the chatbot's persona in response selection.

On the contrary, different personality traits are related to emotion regulation difficulties (Pollock et al., 2016). Due to this, a person's expected emotions can deviate based on his persona. Besides that, we also observe that concepts discussed in a conversational flow play an important role in response selection. However, no one has incorporated this in response selection.

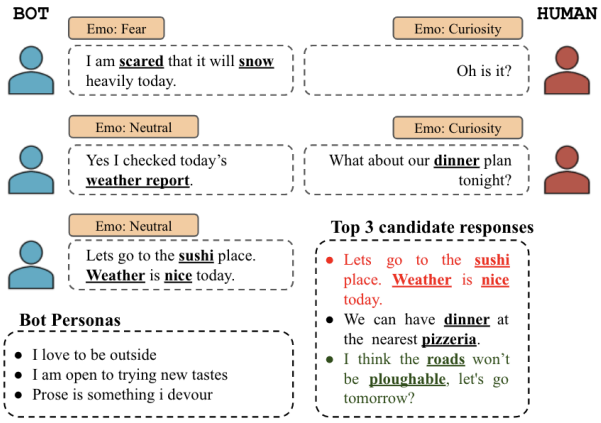

Figure 1: For this conversation, the selected candidate response directly contradicts the context. Also, the bot's persona influences the response selection, while the situational emotions and concepts get ignored. The underlines phrases/words denote the concepts.

Model	Emotion Inappropriate(%)	Contradictory(%)	Off-topic(%)
BERT-CRA	7.35	11.88	12.3

Table 1: Statistics of issues reported in the test split of Persona-Chat inferred by BERT-CRA (Gu et al., 2021b) [1]

[1]Insights drawn from the human evaluation done on 500 randomly selected data-points from self-persona original and partner-persona original sets of Persona-Chat

Proceedings of the 29th International Conference on Computational Linguistics, pages 532–541
October 12–17, 2022.

We can infer the significance of these problems from Table 1. So, to increase the usability of the personalized response selection systems, all these fundamental problems need to be addressed. We automatically annotate Persona-Chat (Zhang et al., 2018) dataset using a series of classifiers and rule-based modules. We model emotion-persona interaction, context-response entailment, and concept-flow using the annotations. To compare the ability of annotated features to enhance the emotion-persona interaction, contradiction avoidance, and adherence to the concept flow, we perform preliminary experiments by devising independent encoders based on BERT. Our baseline model extends BERT-CRA (Gu et al., 2021b) where we introduce an additional bot-encoder to represent the bot-utterances better. Subsequently, we propose three fusion strategies, emotion-aware(EmA), entailment-aware(EnA), persona-entailment-aware(P-EmA). These fusion strategies are designed based on emotion-persona interaction or persona-entailment information. We propose a concept-flow encoding technique that matches relevant concepts from the context and candidate responses with these fusion strategies.

We test our proposed methods on the Persona-Chat dataset with our automatic annotation. The results show that a model trained on a combination of our proposed fusion strategies outperforms the current state-of-the-art model by a margin of 2.3% in terms of top-1 accuracy **hits@1**.

In summary, the contributions of this paper are three-fold. (1)Automatically annotate Persona-Chat dataset with utterance level emotion, entailment, and concept information to provide extra supervision. (2) A suite of fusion strategies and a concept-flow encoder which are designed and implemented into a series of models, aiming to explore the impact of emotion, entailment, and concept-flow in the task of response selection. (3) Experimental results demonstrate that our proposed models outperform the existing state-of-the-art models by significant margins on the widely used Persona-Chat response selection benchmark.

2 Related Works

2.1 Personalized Response Selection

Chit-chat models typically trained over many dialogues with different speakers lack a consistent personality and explicit long-term memory. These models produce an utterance given only a recent dialogue history. Li et al. proposed a persona-based neural conversation model to capture individual characteristics such as background information and speaking style. (Zhang et al., 2018) has constructed Persona-Chat dataset to build personalized dialog systems; this is by far the largest public dataset containing million-turn dialog conditioned on persona. Many benchmarks have been established for this dataset. For example, (Mazaré et al., 2018) proposed the fine-tuned Persona-Chat (FT-PC) model, which first pre-trained models using a large-scale corpus based on Reddit to extract valuable dialogues conditioned on personas and then fine-tuned these pre-trained models on the Persona-Chat dataset. (Wolf et al., 2019; Liu et al., 2020) also employed the pre-trained language model(GPT) for building personalized dialogue agents. (Gu et al., 2020c) proposed filtering before iteratively referring (FIRE) to ground the conversation on the given knowledge and then perform the deep and iterative matching. (Gu et al., 2021b) explored a new direction by proposing four persona fusion strategies, thereby incorporating partner persona in response selection.

2.2 Faithfulness to Context

Faithfulness in conversational systems to conversation context or knowledge is a very broad topic that can range from decreasing fact hallucination(Chen et al., 2021), reducing contradictory responses, staying on topic, etc. (Rashkin et al., 2021) has used additional inputs to act as stylistic controls that encourage the model to generate responses that are faithful to a provided evidence or knowledge. However, no one has studied the level of faithfulness the current personalized response selection systems exhibit to the conversation history. Thus, this paper thoroughly explores the impact of utilizing utterance-level emotions, entailment, and concepts on the performance of personalized response selection.

3 Dataset

In this work, we extend Persona-Chat (Zhang et al., 2018) and augment it with a series of annotators. The dataset consists of 8939 complete dialogues for training, 1000 for validation, and 968 for testing. Responses are selected at every turn of a conversation sequence, resulting in 65719 context-

responses pairs for training, 7801 for validation, and 7512 for testing. The positive and negative response ratio is 1:19 in the training, validation, and testing sets. There are 955 possible personas for training, 100 for validation, and 100 for testing, consisting of 3 to 5 profile sentences. A revised version of persona descriptions is also provided by rephrasing, generalizing, or specializing the original ones to make this task more challenging.

4 Automatic Dataset Annotation

We have annotated the `Persona-Chat` with the help of a series of automatic annotation schemes. Since we are studying the effect of emotions in personalized response selection, we assign emotion labels to the personas, context-utterances, and candidate responses using an emotion classifier. Personas and utterances were annotated using an entailment classifier to incorporate the entailment information while selecting responses. Finally, we follow a multi-layer keyword mining strategy to match meaningful concepts appearing in the context and response.

4.1 Emotion

We trained an emotion classifier on `GoEmotions` dataset (Demszky et al., 2020). This dataset contains 58k English Reddit comments, labeled for 27 emotion categories or Neutral. We fine-tuned `RoBERTa` using this dataset. We saved the checkpoint with the best Macro F1 of 49.4% (equal to the current state of the art) and used this for annotating each utterance. Since emotion classification is a challenging task and given the complexity of the affect lexicons in the corpus, we only consider the labels which can be predicted with more than 90% confidence(i.e., probability higher than 90%). The goal here is to study the effect of emotion in personalized response selection; developing a highly accurate emotion classifier is kept outside the scope of this work.

4.2 Entailment

For annotating entailment, we have used an ensemble of two models. The first one is `RoBERTa` based model trained on Stanford Natural Language Inference (SNLI) corpus (MacCartney and Manning, 2008) released by AllenAI[2]. The second model is also a `RoBERTa` based model fine-tuned on DECODE (Nie et al., 2020). We take the two models' weighted average of both probabilities during inference. The second model has a higher preference with 80% weightage as it is trained on conversational data. The entailment label is assigned to every persona-response and utterance-response pair.

[2]https://github.com/allenai/allennlp-models

4.3 Concept Mining

We mine keywords and key phrases from the persona sentences, utterances, and responses denoted as $\{pc_i\}_{i=1}^{N_{pc}}, \{uc_i\}_{i=1}^{N_{uc}}, \{rc_i\}_{i=1}^{N_{rc}}$ respectively. We follow the techniques proposed in (Tang et al., 2019) to extract the first level of keywords. Subsequently, we expand the concept lists by extracting key phrases using the RAKE (Rose et al., 2010). We hypothesize that concepts appearing in responses should adhere to the speaker's persona. So, we prune some of the response/ context keywords by calculating the average of Point-wise Mutual Information score between persona keywords and response/ context keywords $\sum_{j=1}^{N_{pc}} PMI(pc_j, rc_i)/N_{pc}$ and rejecting the concepts which are below a threshold value(λ). Similarly, for response/ concept key phrases extracted using RAKE, we only keep top N key phrases. Finally, we combine the persona and context keywords and treat them as context keywords(uc_i).

5 Methodology

5.1 Problem Definition

Given a dataset $D = \{(C_i, uc_i, p_i, r_i, rc_i, y_i)\}_{i=1}^{N}$ is a set of N tuples consisting context C_i, the persona of the speaker or the partner p_i, response to the context r_i, and the ground truth y_i. A set of concepts appearing in context and a response is denoted by uc_i and rc_i, respectively. The context can be represented as $C_i = \{(U_j, Emo_j, Entail_j)\}_{j=1}^{L}$ where U_j is an utterance, Emo_j is the dominant emotion present in U_j and $Entail_j$ is the entailment label of U_j with respect to r_i. The jth utterance U_j is denoted by $U_j = \{u_1 j, u_2 j, ..., u_M j\}$ which consists of M tokens. Each response r_i contains single utterance, $y_i \in \{0, 1\}$, $Emo_j \in \{0, 1, ...P\}$, and $Entail_j \in$ { `entailment`,`neutral`,`contradiction`} where P are the total number of emotion types possible in the D. The task is to train a matching model for D, $g(C, uc, p, rc, r)$. Given a triple of context-persona-response the goal of the matching model $g(C, uc, p, rc, r)$ is to calculate the degree of match between (C, uc, p) and (rc, r).

5.2 Bot Context Encoding

When two users communicate, many topics are often discussed in parallel, and sometimes a few utterances might not be relevant for response selection. To account for the model to be aware of the speaker change information, Gu et al. introduced a speaker disentanglement strategy in the form of *speaker embeddings* fused with the original token embeddings. This technique has proven to improve response selection performance (Gu et al., 2020b; Su et al., 2021). However, the problem of the maximum length of positional embeddings still exists. To circumvent this, we have created bot-context encoding, which captures the representation of the bot's turns in the context while ignoring the user's turns. The intent is to use the bot's turns to maintain consistency during response selection. The input sequence that is sent to BERT to encode bot context is composed as follows:

$$\begin{aligned} x_{si} &= [CLS]u_2[EOU]u_4[EOU]... \\ &\quad ...u_{n-1}[EOU][SEP]r_i[EOU] \end{aligned} \tag{1}$$

Where $u_1, u_4, ...u_{n-1}$ are bot's utterances in the context, $[EOU]$ is a special token denoting the end of an utterance.

The resultant tokens x_{si} are passed through `bert-base-uncased`, the last hidden states of k layers i.e. $\{\mathbf{h_{s1}^{(l)}}, \mathbf{h_{s2}^{(l)}}, ..\mathbf{h_{sT}^{(l)}}\}$, for $l = 1, 2, ..k$ are used in downstream tasks.

5.3 Fusion Strategies

We use several fusion strategies to model the interdependencies of the persona, emotion, and entailment information. We use BERT (Devlin et al., 2019) as our base sentence encoder. Like the Bi-encoder (Humeau et al., 2020) we concatenate context utterances as a single context sentence before passing it into BERT.

5.3.1 Baseline

For the baseline, we have extended `BERT-CRA` (Gu et al., 2021b) where persona and context are concatenated to form sequence A and response form sequence B. Then, these two sequences are concatenated using $[SEP]$ token. We made two changes to this model; first, we added *speaker embeddings* with the original token representation. Secondly, we fuse bot-context encoding as described in the previous section with `BERT-CRA` encoding by doing multi-headed attention between the hidden representation of the last k layers of both encoders. The token arrangement is as follows:

$$\begin{aligned} x_{CRAi} &= [CLS]p_1p_2...p_i[EOP]u_1[EOU] \\ &\quad ...u_i[EOU][SEP]r_i[EOU] \end{aligned} \tag{2}$$

Where $p_1p_2...p_i$ are the personalities of the speaker, $[EOP]$ token denotes end of personality representation, $u_1, u_2, ..u_i$ are the utterances in the context. The resultant tokens x_{CRAi} are passed through `bert-base-uncased`, the hidden states of last k layers i.e. $\{\mathbf{h_{c1}^{(l)}}, \mathbf{h_{c2}^{(l)}}, ..\mathbf{h_{cT}^{(l)}}\}$, for $l = 1, 2, ..k$ are used in downstream tasks.

Interaction Layer : Since we are using a multi-encoder pipeline, it is crucial to capture the interaction between the encoders. For that, we use multi-head attention between hidden states of speaker context encoder and `BERT-CRA`. For ease of presentation, we denote the whole multi-headed attention layer as $f_{mha}(*, *)$. Then these attention outputs are passed through an aggregation layer, which basically concatenates then passes it through a two-layer feed-forward network and finally mean pools across all the layers to get $\mathbf{h_d}$. The output is passed through a MLP to get the matching degree with the response.

$$\begin{aligned} \{\widetilde{\mathbf{h}}_{\mathbf{s1}}^{(\mathbf{l})}, \widetilde{\mathbf{h}}_{\mathbf{s2}}^{(\mathbf{l})}, ..\widetilde{\mathbf{h}}_{\mathbf{sT}}^{(\mathbf{l})}\} &= f_{mha}(\{\mathbf{h_{s1}^{(l)}}, \mathbf{h_{s2}^{(l)}}, ..\mathbf{h_{sT}^{(l)}}\}, \\ &\quad \{\mathbf{h_{c1}^{(l)}}, \mathbf{h_{c2}^{(l)}}, ..\mathbf{h_{cT}^{(l)}}\}) \end{aligned} \tag{3}$$

$$\begin{aligned} \{\widetilde{\mathbf{h}}_{\mathbf{c1}}^{(\mathbf{l})}, \widetilde{\mathbf{h}}_{\mathbf{c2}}^{(\mathbf{l})}, ..\widetilde{\mathbf{h}}_{\mathbf{cT}}^{(\mathbf{l})}\} &= f_{mha}(\{\mathbf{h_{c1}^{(l)}}, \mathbf{h_{c2}^{(l)}}, ..\mathbf{h_{cT}^{(l)}}\}, \\ &\quad \{\mathbf{h_{s1}^{(l)}}, \mathbf{h_{s2}^{(l)}}, ..\mathbf{h_{sT}^{(l)}}\}) \end{aligned} \tag{4}$$

$$\begin{aligned} \mathbf{h_d} &= \text{MeanPool}(\{\text{FFN}([\{\widetilde{\mathbf{h}}_{\mathbf{s1}}^{(\mathbf{l})}, ..\widetilde{\mathbf{h}}_{\mathbf{sT}}^{(\mathbf{l})}\}; \\ &\quad \{\widetilde{\mathbf{h}}_{\mathbf{c1}}^{(\mathbf{l})}, ..\widetilde{\mathbf{h}}_{\mathbf{cT}}^{(\mathbf{l})}\}])\}_{l=0}^{k}) \end{aligned} \tag{5}$$

Loss Function: The MLP layer predicts whether a context-persona (C, p) pair matches with the corresponding response r based on the derived features. Subsequently, the output from the MLP layer is passed through a softmax output layer to return a probability distribution over all response candidates. All the models described in this paper are trained using MLP cross-entropy loss. Let Θ be the model parameters, then the loss function $\mathcal{L}(D, \Theta)$ for all the models can be formulated as follows:

$$\mathcal{L}(D, \Theta) = -\sum_{(C,p,r,y))\epsilon D} ylog(g(C, p, r)) \tag{6}$$

5.3.2 `BERT-EmA` Emotion Aware Fusion:

In this strategy, an emotion incorporation framework is introduced. Similar to `BERT-CRA` a dual

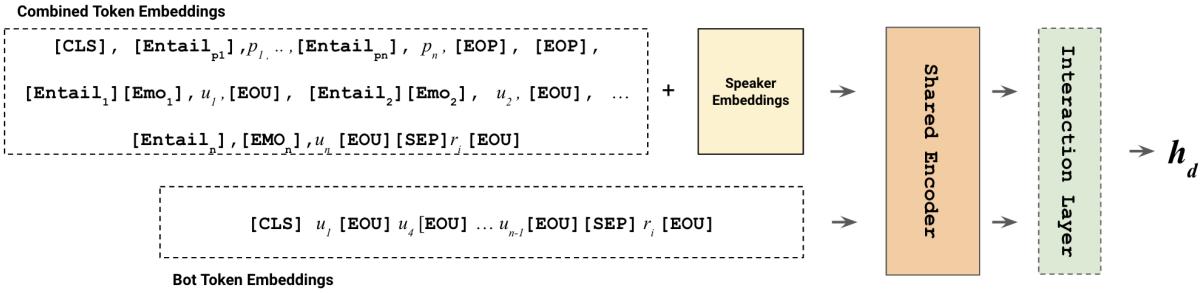

(a) Dual encoder pipeline consisting of combination of all the encoding strategies.

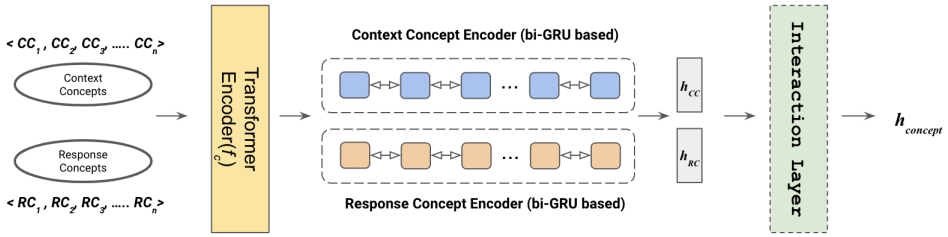

(b) Concept-flow interaction network, the output of this network $\mathbf{h_{concept}}$ can be concatenated with any of the BERT based dual encoder's output($\mathbf{h_d}$).

Figure 2: Overall Training Architecture. Though the BERT based dual encoder is independently depicted but it is trained along with concept-flow interaction(if included).

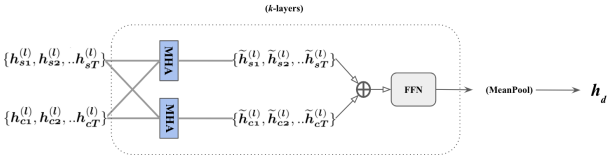

Figure 3: Interaction Layer

pipeline matching network is followed. The first pipeline encodes the speaker's and listener's emotional and personality characteristics in the context. While the other encodes the bot-context as described in section 5.2.

We attach the most probable emotion tag to each utterance to incorporate emotion features in the BERT contextual representation. The emotion-infused context representation is then concatenated with the original persona representation, as described in section 5.3.1. The main goal of representing the context in this way is to understand how the emotions of each utterance interact with the speaker's persona. The input to the emotion encoder is as follows:

$$x_{EmAi} = [CLS]p_1p_2...p_i[EOP][Emo_1]u_1[EOU] \\ ...[Emo_i]u_i[EOU][SEP]r_i[EOU] \quad (7)$$

Similar to baseline, the hidden states of last k layers i.e. $\{\mathbf{h_{e1}^{(l)}}, \mathbf{h_{e2}^{(l)}}, ..\mathbf{h_{eT}^{(l)}}\}$, for $l = 1, 2, ..k$ are used in downstream tasks.

5.3.3 BERT-EnA-P: Entailment Aware Fusion

In this fusion strategy, the intention is to model the entailment information about each of the utterances and personas with the response. Like BERT-EmA we follow a dual encoder pipeline, the first encodes the entailment features, and the second encodes the bot-context. To incorporate entailment features into BERT contextual representation, we attach entailment tags i.e. <contradiction>, <entailment> and <neutral> at the start of every utterance and persona. The response is concatenated with the context-entailment representation with a $[SEP]$ token. The input to the entailment encoder is as follows:

$$x_{EmA-Pi} = [CLS][Entail_{p1}]p_1...[EOP] \\ [Entail_1]u_1[EOU] \\ [Entail_2]u_2[EOU] \\ ...[Entail_i]u_i[EOU] \\ [SEP]r_i[EOU] \quad (8)$$

The hidden states of last k layers i.e. $\{\mathbf{h_{en1}^{(l)}}, \mathbf{h_{en2}^{(l)}}, ..\mathbf{h_{enT}^{(l)}}\}$, for $l = 1, 2, ..k$ are used in downstream tasks.

Finally, we experiment with a combined pipeline as depicted in Figure 2a.

5.4 Concept-Flow(CF) Interaction

In section 4.3, we describe how we extract relevant concepts from the context and the response. An appropriate response often has concepts most recently discussed in the context. So, to model that, we construct a concept-flow interaction network, where the interaction between the context-concepts and response-concepts are measured and used as a feature in response relevance classification.

Let us consider $\{CC_1, CC_2, ..., CC_n\}$ are concepts extracted from context and $\{RC_1, RC_2, ..., RC_n\}$ are concepts extracted from a response. Now, we pass each of these concepts through a transformer-based concept encoder f_c to get two sets of concept embeddings $\{\mathbf{ec_1}, \mathbf{ec_2}, ..., \mathbf{ec_n}\}$, $\mathbf{ec_i} \in \mathbb{R}^{d_c}$ and $\{\mathbf{rc_1}, \mathbf{rc_2}, ..., \mathbf{rc_n}\}$, $\mathbf{rc_i} \in \mathbb{R}^{d_c}$ for context and response concepts respectively. To learn the context flow representation for each set of concepts, we apply a bi-directional GRU network to capture sequential dependencies between subsequent concepts in a conversational situation. Context-concept and response-concept representation $\mathbf{h_i^{cc}}$, $\mathbf{h_i^{rc}}$ can be formulated as:

$$\mathbf{c_i^{cc}}, \mathbf{h_i^{cc}} = \overleftrightarrow{GRU}(\mathbf{ec_i}, \mathbf{h_{i-1}^{cc}}) \quad (9)$$

$$\mathbf{c_i^{rc}}, \mathbf{h_i^{rc}} = \overleftrightarrow{GRU}(\mathbf{er_i}, \mathbf{h_{i-1}^{rc}}) \quad (10)$$

$$\mathbf{h_{cc}} = tanh(\sum_{\mathbf{j \in 2 * N_l}} \mathbf{W_j h_j^{cc}}) \quad (11)$$

$$\mathbf{h_{rc}} = tanh(\sum_{\mathbf{j \in 2 * N_l}} \mathbf{W_j h_j^{rc}}) \quad (12)$$

Where $\mathbf{h_i^{cc}} \in \mathbb{R}^{2d_c}$, $\mathbf{h_i^{rc}} \in \mathbb{R}^{2d_c}$ are the i - the hidden states and $\mathbf{c_i^{cc}} \in \mathbb{R}^{2d_c}$, $\mathbf{c_i^{rc}} \in \mathbb{R}^{2d_c}$ are the outputs of the respective GRU encoders, $\mathbf{W_j}$ is a learn-able parameter and N_l is the number of layers in each GRUs. To model the interaction between $\mathbf{h_i^{cc}}$ and $\mathbf{h_i^{rc}}$ we follow the same interaction mechanism described in the earlier section. The output $\mathbf{h_{concept}}$ is concatenated with the dual encoder output $\mathbf{h_d}$ before passing it through an MLP.

6 Experimental Setup

6.1 Training Details

The ratio of positive to negative samples in the training set is 1:19, so there is a high imbalance in training data. Taking inspiration from (Gu et al., 2021b) we adopted a dynamic negative sampling strategy in which the ratio of positive and negative responses is 1:1 in an epoch. We keep the positive response constant and change the negative response for every epoch, generating data for 19 epochs. We use `bert-base-uncased` as the base for each of our pretraining-based fusion models. In concept mining strategy, we have taken the top 3 concepts extracted using RAKE, λ for PMI-based scoring was varied from 0.3 to 0.8 with 0.1 steps, and 0.5 was found optimum. The number of turns in the conversation history used for concept mining varied following this set: $\{2, 3, 4, 5, 6, 7\}$. We preserve the original parameters of `bert-base-uncased`. The number of k-last layers in the interaction layer varied following this set: $\{3, 4, 5, 6\}$; after some initial experimentation, 4 was found as the optimum value. The number of heads in the multi-head attention layer was kept 8. We use a 6-layered version MiniLM(Wang et al., 2020) to encode the concepts; the embedding dimension was 384. The number of layers in the bi-directional GRUs in the concept encoder is 2. A dropout rate of 0.7 is applied to the concept encoder hidden representation before we send it to the interaction layer. AdamW(Loshchilov and Hutter, 2019) optimizer was used for optimization. The initial learning rate was set to 2e-5 and linearly decayed by L2 weight decay. The maximum sequence length was set to 320. The training batch size was 12. The relevance prediction head used a single feed-forward layer with sigmoid activation. All code was implemented using the PyTorch framework. Also, we used 2 NVIDIA RTX A5000 GPUs to train the models. The average training time for one epoch was 46 minutes, using all our fusion strategies and concept encoding.

6.2 Evaluation Metrics

We used the same evaluation metrics as the previous work to ensure comparable results. Each model aimed to select the best-matched response from available candidates for the given context and persona. We calculated the recall of the true positive replies, denoted as **hits@1**. In addition, the mean reciprocal rank **(MRR)** was also adopted to take the rank of the correct response overall candidates into consideration.

6.3 Comparison Methods

For comparison, we have only selected pretraining-based models.

- **FT-PC (Mazaré et al., 2018):** employed the "pretrain and fine-tune" framework by first

Model	Self Persona				Partner Persona			
	Original		Revised		Original		Revised	
	hits@1	**MRR**	**hits@1**	**MRR**	**hits@1**	**MRR**	**hits@1**	**MRR**
FT-PC (Mazaré et al., 2018)	-	-	60.7	-	-	-	-	-
DIM (Gu et al., 2019b)	78.8	86.7	70.7	81.2	64.0	76.1	63.9	76.0
TransferTransfo (Wolf et al., 2019)	80.7	-	-	-	-	-	-	-
FIRE (Gu et al., 2020c)	81.6	-	74.8	-	-	-	-	-
BERT-CRA (Gu et al., 2021b)	84.3	90.3	79.4	86.9	71.2	80.9	71.8	81.5
Baseline	84.4	90.7	79.4	87.6	71.2	81.1	71.4	81.5
BERT-EmA	84.6	90.9	79.8	87.7	71.4	81.2	71.4	81.6
BERT-P-EnA	85.3	91.2	80.5	87.9	71.7	81.3	71.3	81.4
BERT-EmA+BERT-P-EnA	85.8	91.4	80.7	88.0	72.3	81.5	71.7	81.5
BERT-EmA+BERT-P-EnA+CF (All)	**86.6***	**91.6***	**81.3***	**88.6***	**72.6***	**81.9***	**72.4***	**81.9***

Table 2: Performance of the proposed and previous methods on the Persona-Chat dataset under different persona configurations. The meanings of "Self Persona", "Partner Persona", "Original", and "Revised" can be found in Section 3. "-" represents that the results were not reported in their papers. Numbers marked with * denote that the improvement over the best performing baseline is statistically significant (t-test with p-value < 0.05). Numbers in bold denote the combined fusion strategy that achieves the best performance.

pretraining on a domain-specific corpus, dialogues of which were extracted from Reddit, and then fine-tuning on the Persona-Chat.

- **DIM (Gu et al., 2019b):** used a dually matching network (DIM) which performs interactive matching between responses and contexts and between responses and personas respectively for ranking response candidates.
- **TransferTransfo (Wolf et al., 2019)**: the paper fine-tunes a transformer model(GPT) using `Persona-Chat` dataset on a multi-task objective which combines several unsupervised task.
- **`BERT-CRA` (Gu et al., 2021b):** This work presents four context-aware persona fusion strategies and the models are initialized and pretrained using BERT on `Persona-Chat` dataset.

6.4 Experimental Results

Table 2 reports the evaluation results of our proposed and previous methods on `Persona-Chat` under various persona configurations. We can see that incorporating the emotion and entailment knowledge of the utterances coupled with generic distributional semantics and external knowledge learned from pretraining rendered improvements on both **hits@1** and **MRR** conditioned on various personas. Compared to FT-PC (Mazaré et al., 2018), our best model outperformed it by 20.4 % in terms of hits@1 conditioned emotion, entailment and concepts. Compared to TransferTransfo (Wolf et al., 2019), which was also trained using pretrained transformer models, our combined model outperformed it, which shows the effectiveness of fusion strategies and the concept-encoder. Lastly, our combined model outperformed the `BERT-CRA` (Gu et al., 2021b) in all the tasks. We see a 2.3 % and 1.9 % improvement in original and revised self-persona, and 1.4 % and 0.6 % improvement in original and revised partner-persona in terms of **hits@1**. The results bolster our hypothesis that emotion, entailment, and concepts play an important role in the task of response selection. Also, it is to be noted that `Persona-Chat` is a synthetic dataset, i.e., the data collection did not happen naturally. Therefore, the chances that the user will display this subtle interplay of persona and emotion is less. In addition to that, we observe the presence of contradictory distractor responses. We see a significant performance improvement from this information by introducing entailment-aware fusion and concept encoding.

6.5 Human Evaluation

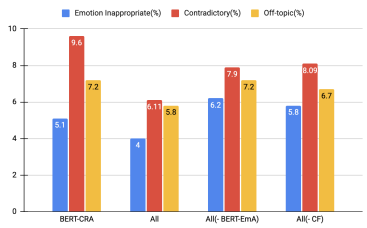

Figure 4: Human evaluation results on `Persona-Chat` self-persona original test split

Since a qualitative study by humans is necessary to understand the effectiveness of the proposed methods, we further perform a human evaluation

Models	hits@1	MRR
Baseline	84.4	90.7
BERT-EmA(− Speaker Encoding)	84.5	90.8
BERT-EmA	84.6	90.9
BERT-EnA-P	**85.3**	**91.2**

Table 3: Ablation Study for Emotion and Entailment on self-original persona.

on a portion of the data. We randomly sampled 100 inferred examples from the test set by baseline model, combined model, combined model except for the emotion, and combined model except for the concept flow interaction. We combined all the samples and evaluated them using Amazon Mechanical Turk by 2 different turkers on three metrics: emotion inappropriate, contradictory, and off-topic. The turkers needed to select if any of the three issues were present in an example. The percentages of reported issues by both groups are shown in Figure 4. The results reveal that all our encoding pipelines effectively reduce contradictory responses and somewhat reduce off-topic and emotionally inappropriate responses. The agreement between the two groups was moderate (Krippendorff's $\alpha = 0.713$).

7 Analysis

7.1 Ablation Study for Emotion and Entailment

We perform ablation studies(shown in Table 3) to validate the effectiveness of emotion and entailment fusion in our proposed models. We see a slight improvement in our baseline model that uses our proposed speaker embedding. Also, unsurprising that the effect of emotion is not significant. As the dataset is artificially created, and emotions exhibited by the annotators are not always true. However, some performance improvement is observed. Conditioning persona in entailment fusion improves performance considerably as responses may not entail the speaker's persona.

7.2 Effect of Context Turns on Concept Representation

Concept matching boosts the evaluation performance further. However, the number of turns in the conversation history from which we mine the concepts influences the performances. It is evident from Figure 5 that the essential concepts in the most relevant response will be present in the recent conversation history.

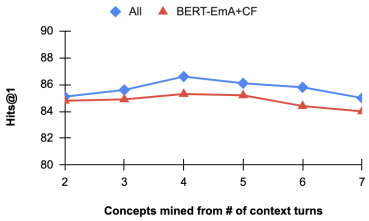

Figure 5: This graph shows how **hit@1** reaches an optimum value and then decreases with an increase in the number of turns used to mine concepts.

personas	my favorite color is blue . <ent: neutral> I enjoy reading mysteries . <ent: neutral> I have seven children . <ent: entail> I grew up on a large farm . <ent: neutral>
context	A: hello how are you today? <emo:curiosity> <ent: neutral> B: I am well. how are you? <emo:curiosity> <ent: neutral> A: I am doing great just got back from the beach <emo:excitement> <ent: neutral> B: that is great. I live far from the beach . <emo:caring> <ent: neutral> A: I am very lucky we live beside the beach. what do you do for a living? <emo:curiosity> <ent: neutral> B: I keep busy with my seven children . <emo:excitement> <ent: neutral> A: wow that much have taken some adjusting I teach kindergarten . <emo:surprise> <ent: neutral>
golden response	do you teach mysteries to your children ? they are my favorite type of novel . <emo:curiosity>
BERT-CRA	that must be a lot of work but very rewarding i bet <emo:realization>
All	do you teach mysteries to your children ? they are my favorite type of novel . <emo:curiosity>

Table 4: Case study showing concept flow.

7.3 Case Study

Table 4 shows the efficacy of concept-encoding. Sometimes models fine-tuned on pretrained transformer models like BERT-CRA tends to select more generic responses. These models pay less attention to the persona or specific keywords in the context while selecting responses. In this example, our proposed model performs better than BERT-CRA as it is conditioned on the concepts. Specifically, concepts in the correct response i.e "mysteries", "novel" relates to "reading mysteries" concept in the persona and "your children" relates to "teach kindergarten" in the context.

8 Conclusion

This work proposes a suite of novel fusion strategies and concept-flow encoder, which leverages the utterances' emotion, entailment, and concept information. These features help improve the performances of our models and provide critical insights into certain aspects of how humans communicate with each other. Though the techniques used in this paper are simple, it highlights the areas where response selection often falters, like detecting contraction, deviation from the concepts, etc. This work can be further extended by using a graphical

model to improve the concept representations.

Acknowledgements

We thank the anonymous reviewers for providing valuable feedback on our manuscript. This work is partly supported by NSF grant number IIS-2214070. The content in this paper is solely the responsibility of the authors and does not necessarily represent the official views of the funding entity.

References

Sihao Chen, Fan Zhang, Kazoo Sone, and Dan Roth. 2021. Improving faithfulness in abstractive summarization with contrast candidate generation and selection. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 5935–5941, Online. Association for Computational Linguistics.

Dorottya Demszky, Dana Movshovitz-Attias, Jeongwoo Ko, Alan Cowen, Gaurav Nemade, and Sujith Ravi. 2020. Goemotions: A dataset of fine-grained emotions.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. Bert: Pre-training of deep bidirectional transformers for language understanding.

Jia-Chen Gu, Tianda Li, Zhen-Hua Ling, Quan Liu, Zhiming Su, Yu-Ping Ruan, and Xiaodan Zhu. 2021a. Deep contextualized utterance representations for response selection and dialogue analysis. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 29:2443–2455.

Jia-Chen Gu, Tianda Li, Quan Liu, Zhen-Hua Ling, Zhiming Su, Si Wei, and Xiaodan Zhu. 2020a. Speaker-aware bert for multi-turn response selection in retrieval-based chatbots.

Jia-Chen Gu, Tianda Li, Quan Liu, Xiaodan Zhu, Zhen-Hua Ling, Zhiming Su, and Si Wei. 2020b. Speaker-aware BERT for multi-turn response selection in retrieval-based chatbots. *CoRR*, abs/2004.03588.

Jia-Chen Gu, Zhen-Hua Ling, and Quan Liu. 2019a. Utterance-to-utterance interactive matching network for multi-turn response selection in retrieval-based chatbots.

Jia-Chen Gu, Zhen-Hua Ling, Xiaodan Zhu, and Quan Liu. 2019b. Dually interactive matching network for personalized response selection in retrieval-based chatbots. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 1845–1854, Hong Kong, China. Association for Computational Linguistics.

Jia-Chen Gu, Zhenhua Ling, Quan Liu, Zhigang Chen, and Xiaodan Zhu. 2020c. Filtering before iteratively referring for knowledge-grounded response selection in retrieval-based chatbots. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 1412–1422, Online. Association for Computational Linguistics.

Jia-Chen Gu, Hui Liu, Zhen-Hua Ling, Quan Liu, Zhigang Chen, and Xiaodan Zhu. 2021b. Partner matters! an empirical study on fusing personas for personalized response selection in retrieval-based chatbots. In *Proceedings of the 44th International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '21, page 565–574, New York, NY, USA. Association for Computing Machinery.

Samuel Humeau, Kurt Shuster, Marie-Anne Lachaux, and Jason Weston. 2020. Poly-encoders: Transformer architectures and pre-training strategies for fast and accurate multi-sentence scoring.

Mark R. Leary and Ashley Batts Allen. 2011. Personality and persona: Personality processes in self-presentation. *Journal of Personality*, 79(6):1191–1218.

Jiwei Li, Michel Galley, Chris Brockett, Georgios Spithourakis, Jianfeng Gao, and Bill Dolan. 2016. A persona-based neural conversation model. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 994–1003, Berlin, Germany. Association for Computational Linguistics.

Qian Liu, Yihong Chen, Bei Chen, Jian-Guang Lou, Zixuan Chen, Bin Zhou, and Dongmei Zhang. 2020. You impress me: Dialogue generation via mutual persona perception.

Ilya Loshchilov and Frank Hutter. 2019. Decoupled weight decay regularization.

Bill MacCartney and Christopher D. Manning. 2008. Modeling semantic containment and exclusion in natural language inference. In *Proceedings of the 22nd International Conference on Computational Linguistics (Coling 2008)*, pages 521–528, Manchester, UK. Coling 2008 Organizing Committee.

Pierre-Emmanuel Mazaré, Samuel Humeau, Martin Raison, and Antoine Bordes. 2018. Training millions of personalized dialogue agents. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 2775–2779, Brussels, Belgium. Association for Computational Linguistics.

Yixin Nie, Mary Williamson, Mohit Bansal, Douwe Kiela, and Jason Weston. 2020. I like fish, especially dolphins: Addressing contradictions in dialogue modeling.

Noah C. Pollock, Gillian A. McCabe, Ashton C. Southard, and Virgil Zeigler-Hill. 2016. Pathological personality traits and emotion regulation difficulties. *Personality and Individual Differences*, 95:168–177.

Hannah Rashkin, David Reitter, Gaurav Singh Tomar, and Dipanjan Das. 2021. Increasing faithfulness in knowledge-grounded dialogue with controllable features.

Nadine R Richendoller and James B Weaver III. 1994. Exploring the links between personality and empathic response style. *Personality and individual Differences*, 17(3):303–311.

Stuart Rose, Dave Engel, Nick Cramer, and Wendy Cowley. 2010. *Automatic Keyword Extraction from Individual Documents*, chapter 1. John Wiley & Sons, Ltd.

Yixuan Su, Deng Cai, Qingyu Zhou, Zibo Lin, Simon Baker, Yunbo Cao, Shuming Shi, Nigel Collier, and Yan Wang. 2021. Dialogue response selection with hierarchical curriculum learning. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 1740–1751, Online. Association for Computational Linguistics.

Jianheng Tang, Tiancheng Zhao, Chenyan Xiong, Xiaodan Liang, Eric P. Xing, and Zhiting Hu. 2019. Target-guided open-domain conversation.

Wenhui Wang, Furu Wei, Li Dong, Hangbo Bao, Nan Yang, and Ming Zhou. 2020. Minilm: Deep self-attention distillation for task-agnostic compression of pre-trained transformers.

Thomas Wolf, Victor Sanh, Julien Chaumond, and Clement Delangue. 2019. Transfertransfo: A transfer learning approach for neural network based conversational agents.

Yuwei Wu, Xuezhe Ma, and Diyi Yang. 2021. Personalized response generation via generative split memory network. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 1956–1970, Online. Association for Computational Linguistics.

Chen Zhang, Hao Wang, Feijun Jiang, and Hongzhi Yin. 2021a. Adapting to context-aware knowledge in natural conversation for multi-turn response selection. In *Proceedings of the Web Conference 2021*, WWW '21, page 1990–2001, New York, NY, USA. Association for Computing Machinery.

Saizheng Zhang, Emily Dinan, Jack Urbanek, Arthur Szlam, Douwe Kiela, and Jason Weston. 2018. Personalizing dialogue agents: I have a dog, do you have pets too?

Zhuo Zhang, Danyang Zheng, and Ping Gong. 2021b. Multi-turn response selection in retrieval based chatbots with hierarchical residual matching network. *Journal of Physics: Conference Series*, 1757(1):012023.

Peixiang Zhong, Chen Zhang, Hao Wang, Yong Liu, and Chunyan Miao. 2020. Towards persona-based empathetic conversational models. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 6556–6566, Online. Association for Computational Linguistics.

A Appendix

A.1 Human Evaluation for Initial Study

Five hundred context response pairs randomly sampled from the Persona-Chat self-original test split inferred by `BERT-CRA` were evaluated by at least two AMT workers. The following questions were asked to the workers:

1. Is this response contain emotions that are consistent with the context? (Most definitely/ not at all)

2. Is this response contradicts the context? (Most definitely/ not at all)

3. Do the topics discussed in this response appropriate to the topics discussed in the context? (Most definitely/ not at all)

A.2 Final Human Evaluation

Same evaluation pattern is followed as A.1, average pay for each HIT was 0.07 $.

A Multi-Dimensional, Cross-Domain and Hierarchy-Aware Neural Architecture for ISO-Standard Dialogue Act Tagging

Stefano Mezza, Wayne Wobcke and Alan Blair
School of Computer Science and Engineering
University of New South Wales, Sydney NSW 2052, Australia
{s.mezza,w.wobcke,a.blair}@unsw.edu.au

Abstract

Dialogue Act tagging with the ISO 24617-2 standard is a difficult task that involves multi-label text classification across a diverse set of labels covering semantic, syntactic and pragmatic aspects of dialogue. The lack of an adequately sized training set annotated with this taxonomy is a major problem when using the standard in practice. In this work, we propose a neural architecture to increase classification accuracy, especially on low-frequency fine-grained tags, on a subset of the ISO 24617-2 taxonomy. Our model takes advantage of the hierarchical structure of the ISO taxonomy and utilises syntactic information in the form of Part-Of-Speech and dependency tags, in addition to contextual information from previous turns. We train our architecture on an aggregated corpus of conversations from different domains, which provides a variety of dialogue interactions and linguistic registers. Our approach achieves state-of-the-art tagging results on the DialogBank benchmark data set, providing empirical evidence that this architecture can successfully generalise to different domains.

1 Introduction

Language understanding is a fundamental component of any conversational system, as it impacts its abilities to correctly recognise a user's communicative functions and act accordingly. Dialogue Act (DA) tagging is a crucial step of this understanding process, particularly in an open-ended conversational setting, as it informs the system on the users' beliefs, desires, intentions and actions.

Table 1 shows an excerpt of a conversation from the Mastodon corpus annotated with Dialogue Act tags. Note that some tags (e.g. *Task:Answer* or *Task:Agreement*) are contextual and depend on the tagging of previous utterances. The annotation also reflects the multi-dimensional nature of the DA tags, categorised as *Task*, *Social* or *Feedback*. In general, an utterance may have multiple tags, even of the same dimension.

Utterance	DA Tags
A: ask anything you'd like	Task:Directive
B: thanks for the interest	Social:Thanking
B: when a girl keeps blinding you with the reflection of the sun is she signalling that she wants to hold hands ?	Task:InfoQuestion
A: only if the flash pattern is .. - . or maybe ... -	Task:Answer
A: Deleted my Facebook account a few days ago and I never felt so free in my entire life.	Task:Inform
A: Now I just have to encourage my closest friends to do the same	Task:Commissive
B: It shouldn't be that hard. They are as tired of social media as I am .	Task:Inform
A: Yes ! I don't get it . Everyone I talk to about Facebook– EVERYONE - - hates it , but none of them will take action .	Task:Agreement

Table 1: An example dialogue from the Mastodon corpus annotated with the ISO 24617-2 taxonomy.

Early dialogue applications usually adopted a list of mutually exclusive and task-specific DA tags which represented the different functions that the system performed, acting essentially as *intent labels*. These taxonomies were also *one-dimensional*, featuring mutually-exclusive tags which did not account for the complexity of the dialogues. The following example from Bunt (2006) clarifies the importance of multi-dimensionality in DA tagging:

S: Can you tell me what time is the first train to the airport on Sunday morning?

A: On Sunday morning the first train to the airport is at 5.32.

S: Thank you!

According to Bunt (2006), the third utterance has two separate communicative functions, as the speaker S is expressing gratitude towards the addressee A (*Social* dimension), while at the same time informing them on their understanding of the train schedule (*Feedback* dimension).

Proceedings of the 29th International Conference on Computational Linguistics, pages 542–552
October 12–17, 2022.

In addition to this, DA taxonomies have also typically lacked a hierarchical organisation of tags, which makes it difficult for a classifier to capture the high-level mutual relationships between dialogue tags (Soria and Pirrelli, 2003).

In an attempt to address these problems, an official ISO standard taxonomy, ISO 246170-2, was introduced in Bunt et al. (2012): this taxonomy is domain-independent, hierarchical and multi-dimensional, and well-suited for open-ended Natural Language Understanding. However, much work in dialogue systems still uses other taxonomies, in part we believe because of the lack of an adequately-sized data set annotated with the standard, which makes it difficult to train a classifier for the taxonomy. Some authors have proposed automated mappings of old resources to the new ISO standard; however, these works are still limited in scope, focusing either on heavily imbalanced data sets (Bunt et al., 2017) or subsets of the ISO 24617-2 taxonomy that are insufficient for real-life conversational scenarios (Mezza et al., 2018). DA tagging with the ISO taxonomy is also an intrinsically difficult task, as it requires handling multiple different dimensions and a collection of different fine-grained tags which differ in semantic, syntactic and contextual aspects. The open-ended nature of the taxonomy provides an additional challenge, as most existing DA-annotated resources tend to have a bias towards specific topics or discussion styles, which hinders the model's capabilities to generalise to unseen conversations.

In this work, we introduce a neural architecture optimised for DA tagging with a subset of the ISO standard taxonomy. Our model combines syntactic, semantic and contextual information and leverages the hierarchical dependencies across labels to improve the classification accuracy, especially on low-frequency fine-grained tags. We combine existing DA-annotated data sets and map them to a subset of the ISO 24617-2 taxonomy to obtain an adequately-sized training set, taking advantage of existing mappings in the literature (Mezza et al., 2018) and novel conversational resources such as Mastodon (Cerisara et al., 2018) and DailyDialog (Li et al., 2017), whose taxonomies easily map to the ISO standard. We also experiment with the addition of online discussions and debates data from the Internet Argument Corpus v2.0 (Abbott et al., 2016; Walker et al., 2012), in order to increase the system's understanding of opinionated and contextual tags. Experimental results show that our approach achieves state-of-the-art classification accuracy on the DialogBank (Bunt et al., 2016) test set. We also provide additional experiments that delve into the details of the training process, including ablation studies and an analysis of the extent to which the different corpora that we utilised contribute to the network's performance. Finally, we share our code and our mapped data set [1], in order to share these resources with the research community and hopefully encourage further research on Dialogue Act classification.

2 Related Work

The concept of Dialogue Act (DA) has its roots in the seminal works by Austin (1975) and Searle (1965), who established the theoretical foundations of the Speech Act theory. A *speech act* captures an utterance at the level of its illocutionary force. Many subsequent works started referring to speech acts in a conversational setting as *Dialogue Acts*, and investigated possible taxonomies of Dialogue Acts. These early taxonomies were flat (there was no distinction between coarse-grained and fine-grained tags), mono-dimensional (each and every utterance had only one Dialogue Act tag) and usually task-specific rather than domain-independent. Examples of these early taxonomies include the DAMSL taxonomy (Allen and Core, 1997), which was used for the annotation of the Switchboard and MRDA conversational corpora, the HCRC coding manual (Anderson et al., 1991), which was used to annotate the Maptask corpus, and the VerbMobil annotation scheme (Jekat et al., 1995), which was used for the annotation of the homonymous corpus.

An interest in formalising these taxonomies into a more rigid theoretical framework arose in the early 2000s, with Traum (2000) analysing existing DA taxonomies and investigating a rigorous definition of Dialogue Acts. Bunt (2005) provided one of the first formal definitions of Dialogue Act as "a unit in the semantic description of communicative behaviour, produced by a sender and directed at an addressee, specifying how the behaviour is intended to influence the context through understanding of the behaviour". The authors combined existing taxonomies such as DAMSL and DIT (Bunt, 1989) into a new taxonomy called DIT++ (Bunt, 2009), which aimed at being a truly open-ended, domain-independent and theoretically sound tax-

[1] https://github.com/coling22tagger/DialogueActTagger

onomy. The fifth version of the DIT++ taxonomy became the official ISO standard for Dialogue Act classification (Bunt et al., 2012). A potential advantage of using the standard is that multiple corpora can be used to construct a larger training set, or cross-domain training sets suitable to cover a wide rage of dialogue tasks; nonetheless, the ISO 24617-2 taxonomy has yet to be fully adopted, with many works still using DAMSL as their target taxonomy (Raheja and Tetreault, 2019; Cervone et al., 2018), or introducing entirely novel taxonomies custom-tailored for specific tasks (Paul et al., 2019; Yu and Yu, 2019). While this is partly due to the complexity of the standard, we believe a significant obstacle is the lack of adequately-sized data sets to train a classifier. Some efforts have been made to convert existing resources to the new taxonomy: Fang et al. (2012) proposed an automated, albeit partial, mapping of the DAMSL taxonomy to the ISO standard, Mezza et al. (2018) extended their work providing partial mappings for the AMI, MapTask, Oasis and VerbMobil taxonomies, and Ribeiro et al. (2020) provides a mapping from the LEGO annotation scheme to the ISO one. There are some planned corpora entirely annotated with the ISO standard, such as ADELE (Gilmartin et al., 2018) or DBOX (Petukhova et al., 2014), which may become valuable tools to work with the taxonomy. However, these resources are still not publicly available at the time of writing. Some interesting corpora were released in recent years which, while not being entirely ISO compliant, adopted DA taxonomies which can be easily mapped to the standard. These include DailyDialog (Li et al., 2017), Mastodon (Cerisara et al., 2018) and MIDAS (Yu and Yu, 2019) among others.

Automatic DA tagging was initially formalised as a text classification task by Stolcke et al. (2000), who presented a Hidden Markov Model for the classification of the Switchboard data set. Since then, many different approaches have been proposed to tackle the task, including rule-based systems (Lendvai et al., 2003), Conditional Random Fields (Quarteroni et al., 2011; Zhou et al., 2014) and Support Vector Machines (Mezza et al., 2018). More recent works shifted their focus to Artificial Neural Networks, which have been proved to be very effective for text classification tasks; many of these models are built around Bi-LSTM/GRU cells with CRF as a top layer, due to this architecture's ability to capture long-term contextual dependencies in dialogue (Kumar et al., 2018; Chen et al., 2018). Many architectures also rely on transformers, usually combined with pre-trained sentence embeddings (Yu and Yu, 2019) or some form of attention mechanism (Raheja and Tetreault, 2019). The vast majority of these models are designed for flat hierarchies of tags, with little to no emphasis on the multi-dimensional nature of Dialogue Acts. Anikina and Kruijff-Korbayova (2019) annotated a subset of the TRADR corpus of robot-assisted disaster response team communications with three dimensions of the ISO standard (*General (Task)*, *Social* and *Turn Management*) and trained various neural classifiers for the task, including CNN, LSTM and FFN; while their work does take advantage of the multi-dimensional aspect of the ISO taxonomy, the scope of their research is limited by the size of the resource and the emphasis on the disaster response domain.

A number of recent works started taking advantage of the hierarchical structure of ISO communicative functions: Wang et al. (2021) introduced a hierarchical neural model for one-dimensional DA tagging (they only consider the Social and Task dimensions of the standard and combine them into a single core dimension), while Ribeiro et al. (2019) proposes a hierarchical and multi-dimensional approach for the Spanish corpus DIHANA. Blache et al. (2020) apply a number of statistical machine learning algorithms, such as *XGBoost* and *Random Forests*, to annotate French medical data with a subset of the ISO standard; their approach separates the classification into two hierarchical steps to increase the accuracy of the model. Mezza et al. (2018) proposed a multi-dimensional and domain-independent approach to DA tagging, and also investigated hierarchical DA tagging through a tree-like structure of SVM classifiers; however, the model was limited in scope and accuracy and did not take into account contextual tags such as *Answer*, *Agree/Disagree*, etc. Ribeiro et al. (2022) utilised an end-to-end hierarchical network with cascading outputs and maximum a posteriori path estimation to classify all the layers of the *General (Task)* semantic dimension; while their architecture handles the whole taxonomy of *Task* communicative functions, it lacks support for additional dimensions of the standard such as *Social*, *Turn Management*, etc. Their model also fails to capture the domain-independent nature of the taxonomy, as its performance degrades with the addition of

out-of-domain data such as the conversations from the LEGO-ISO corpus.

Multi-label Hierarchical Text Classification has been successfully addressed for other text classification tasks in the literature, such as fine-grained Sentiment Analysis (Tai et al., 2015) or Topic Classification (Zhou et al., 2020). The latter work proposed two different models to solve hierarchical, multi-dimensional topic classification on news articles; we adapt a similar approach to include prior information on the hierarchical correlation among labels in our model.

3 Methodology

3.1 Task Definition

A *dialogue* D is defined as a sequence of *dialogue turns* $T_1, \ldots, T_n$, with each turn performed by an individual speaker. Each turn consists of a sequence of *utterances* $u_1, \ldots, u_m$ performed by one of the speakers, with each utterance representing one of the *functional segments* of the turn (i.e. "a minimal stretch of functionally relevant communicative behaviour" (Bunt et al., 2010)). We have a taxonomy of tags $t_1, \ldots, t_n$ arranged in a set of tree-like structures. Each tree corresponds to a *core dimension* (an aspect of utterance function), and groups together tags that correspond to *communicative functions* within the same dimension. Fine-grained DA tags are the leaves of the trees, while coarse-grained tags are the intermediate nodes. *Dialogue Act (DA) tagging* is the task of assigning one or more fine-grained tags $t_1, \ldots, t_k$ to each utterance in the dialogue. Similarly to Mezza et al. (2018), we have decided to adopt a subset of the ISO 24617-2 standard taxonomy (Bunt et al., 2012) for our classifier, since some of the fine-grained tags of the standard do not appear in any of our corpora. We consider three core dimensions of the standard, namely *Task*, *Social* and *Feedback*, and a total of 16 fine-grained DA tags. Figure 1 shows our complete taxonomy.

3.2 Data

There is a widely recognised shortage of conversational data annotated with the ISO 24617-2 standard. Researchers have worked around this issue by designing their own taxonomies (Paul et al., 2019), using older more widely supported taxonomies (Raheja and Tetreault, 2019) or converting existing resources via rule-based mappings (Mezza et al., 2018; Ribeiro et al., 2020). We followed the latter approach and converted a number of resources to our subset of the ISO standard. The resulting aggregated corpus, **General Dialogue Corpus (GDC)**, is a combination of the following corpora:

- The **Switchboard Dialog Act Corpus (SWDA)** (Jurafsky and Shriberg, 1997), a collection of 5-minute telephone conversations on provided topics such as child care, recycling, and news media, annotated with the DAMSL taxonomy. Conversations in the corpus focus on information exchange, with an abundance of *Info Providing* and *Info Seeking* dialogue acts. They also feature a high number of *Feedback* tags due to the nature of telephone conversations, which often need explicit feedback to signal understanding.

- The **ICSI Meeting Recorder Dialog Act (MRDA)** corpus (Shriberg et al., 2004), a collection of transcribed research meetings annotated with a slightly edited version of the DAMSL taxonomy. Similarly to SWDA, this corpus contains a majority of information exchange tags; however, as the conversations involve multiple participants in an academic environment, there is also a significant amount of *conversational structuring* tags (including *Feedback*) and a more formal linguistic register.

- The **DailyDialog** corpus, a human-written and manually labelled set of DA annotated conversations about the daily life of the participants. The corpus focuses on social interactions among human speakers, with a good balance of *Information-Transfer* and *Action-Discussion* tags.

- The **Mastodon** corpus (Cerisara et al., 2018), a Twitter-like corpus of conversation threads on an open-source social platform called Mastodon. It features a combination of information exchange and persuasive dialogue and is annotated with sentiment information and coarse-grained DA tags.

- The **Internet Argument Corpus v2 (IAC)** (Abbott et al., 2016; Walker et al., 2012) is a collection of corpora for research on political debate on Internet forums. We focus on the *4Forums* subset of the resource, which contains argumentative dialogue and features agreement/disagreement stance annotations.

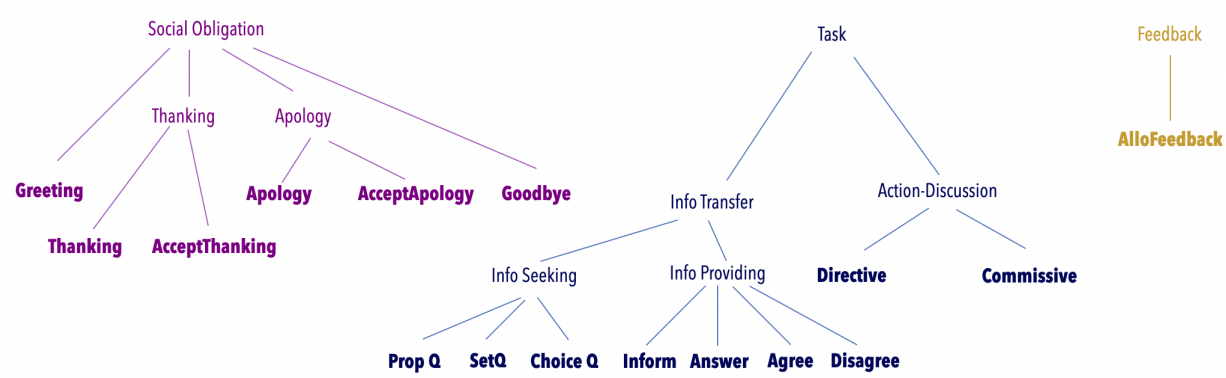

Figure 1: Our subset of the ISO 24617-2 standard for DA tagging.

We chose this particular collection of corpora to have a reasonably balanced distribution of tags across different core dimensions: SWDA and MRDA focus on information exchange and conversational structuring tags, DailyDialog has a high proportion of Action-Discussion tags (about 20% of the overall corpus), and Mastodon and IAC have an emphasis on opinionated, argumentative and persuasive dialogue. These corpora also offer a variety of linguistic registers: Switchboard and DailyDialog focus on everyday conversations with colloquial language, MRDA and IAC contain more formal conversations with a richer vocabulary and lexicon, and Mastodon features an abundance of Internet and chat slang.

We followed the mapping introduced in Fang et al. (2012) for the conversion of SWDA and MRDA taxonomies. The Mastodon corpus utilises a subset of the ISO standard, therefore we adopted the mapping suggested by the authors of the corpus (Cerisara et al., 2018). DailyDialog features coarse-grained DAs which directly map to coarse-grained tags in the General (Task) dimension of the ISO standard. Finally, we converted the *4Forums* subset of IAC by utilising the agreement and disagreement stance annotation to map responses to the *Agreement* and *Disagreement* tags of the standard. More specifically, we labelled responses with an agreement stance lower than -2.0 to *Disagreement*, responses with a stance higher than 2.0 to *Agreement* and all other responses to *Answer*. The train, test and validation splits of the GDC is a combination of all the splits of the included corpora. We utilised the default train, test and validation splits for the MRDA, DailyDialog and SWDA corpora, with the only variation being the removal of the SWDA conversations that appear in the DialogBank from the training and validation splits of the corpus. Given the large size of the Mastodon test split, we have elected to only use the first 500 utterances of the corpus for testing; we reserved 436 utterances as additional training data and 205 utterances for our validation split. Since the *4Forums* corpus of IAC does not have a default train-test split, we just divided the corpus manually and reserved 7847 responses for training, 638 responses for testing and the remaining 1497 for validation. We ensured that utterances belonging to the same conversation would be in the same split when dividing the corpora.

In addition to the test split of GDC, we also tested our model on the **DialogBank** corpus (Bunt et al., 2016), a collection of conversations from different corpora annotated with the ISO taxonomy by the authors of the standard; this is one of the few resources available that are manually annotated with the ISO taxonomy, and therefore constitutes a popular testing benchmark for the task.

3.3 Model

This section describes our model, **Dialogue Act Syntax and Hierarchy-aware Network (DASHNet)**. Figure 2 provides an overview of the main components of the network. DASHNet uses a triple input encoding mechanism: the *lexical encoder* encodes input tokens, the *syntax encoder* encodes syntactical information such as Part-of-speech (POS) tags and Dependency (DEP) tags, and the *context encoder* encodes contextual information from the previous speaker's last utterance. We have limited the context to a single previous utterance to test the hypothesis that some of the tags of the ISO 24617-2 taxonomy are contextual and directly

depend on the tagging of the previous utterance in the dialogue; moreover, some of our resources (such as the IAC) do not provide a context longer than one utterance. Given an input utterance as a sequence of tokens $Uj = t_1, \ldots, t_n$, the *lexical encoder* passes it through a pre-trained embedding layer and a bidirectional GRU layer to obtain the lexical encoding:

$$\vec{E_t} = Embedding(t_1, \ldots, t_n) = \vec{e_1^t}, \ldots, \vec{e_n^t} \quad (1)$$

$$\vec{H_L} = BiGRU(\vec{e_1^t}, \ldots, \vec{e_n^t}) \quad (2)$$

Similarly, grammatical features are encoded in the *syntax encoder* module with a linear layer replacing the pre-trained embeddings:

$$\vec{E_{pos}} = Linear(p_1, \ldots, p_n) = \vec{e_1^p}, \ldots, \vec{e_n^p} \quad (3)$$

$$\vec{E_{dep}} = Linear(d_1, \ldots, d_n) = \vec{e_1^d}, \ldots, \vec{e_n^d} \quad (4)$$

$$\vec{H_G} = \begin{pmatrix} BiGRU(\vec{e_1^p}, \ldots, \vec{e_n^p}) \\ BiGRU(\vec{e_1^d}, \ldots, \vec{e_n^d}) \end{pmatrix} \quad (5)$$

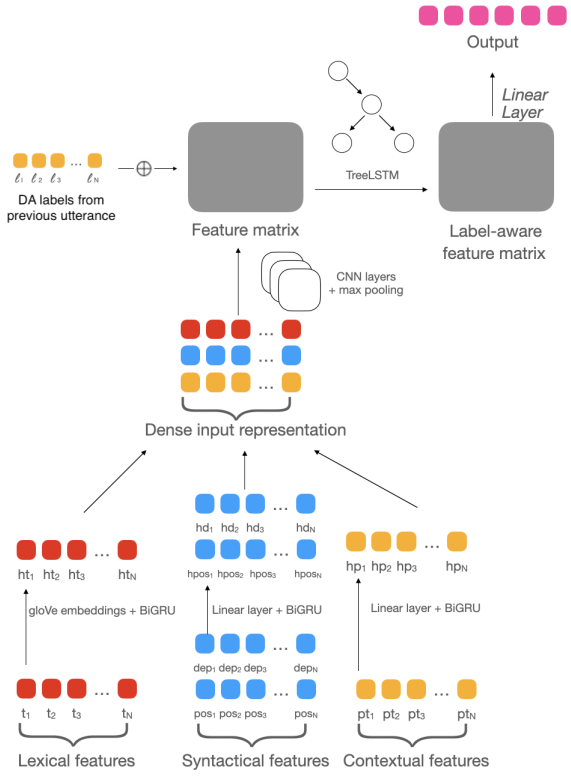

Figure 2: DASHNet architecture.

The *context encoder* input is just the *input encoding* from the previous sentence U_{j-1}. The input encoding for utterance U_j will thus be

$$\vec{I_j} = \begin{pmatrix} H_{(L,j)} \\ H_{(G,j)} \\ I_{(j-1)} \end{pmatrix} \quad (6)$$

Multiple convolutional layers with different kernel sizes, followed by max-pooling, are used to extract relevant features from the input encoding I_J obtaining a feature matrix F_J. An encoding of the list of DA labels from the previous speaker's last utterance is also concatenated to F_J.

Prior knowledge about the hierarchical correlation among labels is then embedded into the feature matrix. More specifically, we estimate the prior probabilities for each tag from the training data distribution as follows:

$$P(L_j|L_i) = \frac{N_j}{\sum_{k \in child(i)} N_k} \quad (7)$$

$$P(L_i|L_j) = 1.0 \quad (8)$$

where $P(L_j|L_i)$ denotes the probability of the fine-grained DA tag j given the parent coarse-grained node i, $P(L_i|L_j)$ denotes the probability of the parent node i given the child node j, $child(i)$ denotes the set of children nodes for tag i and N_k denotes the number of occurrences of tag k in the training set. Since tags in the ISO 24617-2 taxonomy are arranged in a tree structure, the probability of a coarse-grained tag given the occurrence of any of its fine-grained children tags is always equal to 1. We compute these prior probabilities before training, and then encode them in the network through a Bidirectional Tree-LSTM. We use the implementation of BiTree-LSTM introduced in Zhou et al. (2020), which is itself based on the structure encoder presented in Li et al. (2018). Namely, the output from the CNN layers is then transformed through a linear layer to obtain a hidden label representation l_i for each label in the taxonomy (including coarse-grained DA tags). The hidden state h_k for DA tag t_k is then computed as:

$$h_k = h_{k\downarrow} \oplus h_{k\uparrow} \quad (9)$$

$$\text{where} \quad h_{k\downarrow} = \sum_{i \in child(k)} P(L_i|L_k)\, h_i \quad (10)$$

$$\text{and} \quad h_{k\uparrow} = P(L_p|L_k) h_p \quad (11)$$

and p represents the parent node for node k.

3.4 Experimental Setting

We trained our models on Google Colab Pro with CUDA GPU and High Memory settings for 100 epochs, with learning rate $\alpha = 1 \times 10^{-5}$, and use Adam optimiser with weight decay $w = 1 \times 10^{-4}$. We used pre-trained 300-dimensional GloVe embeddings and Kaiming uniform initialisation for weight initialisation (GloVe embeddings are used to facilitate comparison with previous work). The BiGRU layers for input representation have a hidden size of 128 nodes, and the node representation for DA labels in the Tree-LSTM structure is 300-dimensional. We use three CNN layers with kernel size 3, 4 and 5 respectively, with 100 filters each. Finally, we apply Dropout with probability $p = 0.3$ after each BiGRU layer, with probability $p = 0.5$ after the CNN layers and with probability $p = 0.1$ after the Tree-LSTM structure encoder. The values for the hyper-parameters of the network were chosen according to the average Macro-F1 score of the network on the validation split of the GDC corpus.

We extract Part-Of-Speech tags and dependency tags with the spaCy 3.1 Python library, which we also use to tokenise the input utterances. Since our mappings are partial, some of the utterances in the corpora could not be accurately annotated with any ISO communicative functions; these data points were annotated with coarse-grained DA tags and used as contextual features where appropriate (for example, tags *Inform* and *Question* from DailyDialog can be mapped to *Task:InfoProviding* and *Task:InfoSeeking* respectively). Utterances with no direct mapping to either coarse-grained or fine-grained tags were labeled as *Unknown* and discarded during training and testing.

4 Results and Discussion

In this section we present the results of our experimental study. We evaluate the performance of our model on two test sets, namely the test split of GDC and the DialogBank corpus. We provide average Micro-F1 and Macro-F1 scores for our model, with the former providing a measure of its raw accuracy and the latter providing a better metric for how well low-frequency tags are correctly classified. The DASHNet model was able to correctly annotate a large portion of the test split of the GDC data set. Moreover, it also shows promising results on the DialogBank test set, highlighting good generalisation when classifying out-of-domain data.

Table 2 shows the main results of our study. We compared our model DASHNet with HiAGM-TP (Zhou et al., 2020) and with the suite of SVM classifiers proposed by (Mezza et al., 2018). As the code for both systems is openly available[2] [3], we trained both on our GDC training data and tested on both the GDC test split and the entire DialogBank corpus. The DASHNet architecture outperforms both approaches on our two test sets.

Model	DialogBank		GDC	
	Micro-F1	**Macro-F1**	**Micro-F1**	**Macro-F1**
SVM (Mezza et al., 2018)	55.8	49.2	78.3	58.9
HiAGM-TP (Zhou et al., 2020)	77.3	49.6	88.1	71.2
DASHNet (our model)	**83.7**	**57.1**	**90.6**	**76.9**

Table 2: Comparative study between our model and other models in the literature. All models were trained on the train split of GDC. The DASHNet architecture outperforms other models on all of our test sets.

Model	Micro-F1 (DBank)	Precision (DBank)	Recall (DBank)
CRF-ASN (Chen et al., 2018)	64.8	64.0	65.6
HEC (Kumar et al., 2018)	64.0	63.7	64.3
CASA (Raheja and Tetreault, 2019)	65.3	68.6	62.4
HSLT (Wang et al., 2021)	70.2	70.1	70.4
DASHNet (our model)	**83.7***	**85.7***	**81.9***

Table 3: Comparative study between our model and the results reported by (Wang et al., 2021). Since the authors did not specify their train-test split or their target taxonomy, it is not possible to draw a direct comparison.

Table 3 shows a comparison between our model and the results presented by (Wang et al., 2021), who published classification results on the DialogBank corpus for their neural architecture and a number of state-of-the-art models for DA tagging that

[2] https://github.com/ColingPaper2018/DialogueAct-Tagger
[3] https://github.com/Alibaba-NLP/HiAGM

they replicated. While our model outperforms all their proposed architectures, it is worth mentioning that the authors did not specify their taxonomy of fine-grained DA tags, making it impossible to draw a direct comparison. Moreover, their test set only included an unspecified subset of four dialogues of the DialogBank, while our test set includes the entire corpus. Finally, their experiments were on in-domain test data, meaning that they trained and tested their model on different splits of the DialogBank data set, whereas our focus was on out-of-domain data and how to create a model that could generalise to an unseen conversational corpus.

4.1 Ablation Study

Table 4 shows the results of various ablation experiments to gain a better understanding of the extent to which each component of the DASHNet architecture impacts in-domain and out-of-domain classification accuracy.

FEATURES	DialogBank		GDC	
	Micro-F1	**Macro-F1**	**Micro-F1**	**Macro-F1**
Without contextual features	80.0	50.4	87.3	72.9
Without Tree-LSTM prior	81.8	51.2	90.3	74.8
Without POS tags	83.5	54.3	90.5	76.0
Without DEP tags	83.6	52.1	90.6	72.8
All features	**83.7**	**57.1**	**90.6**	**76.9**

Table 4: Ablation study results on the DialogBank and GDC test sets.

Contextual features appear to give the biggest overall boost to the performance of the model, both on in-domain and out-of-domain data. This result is in line with other works in the field which highlight the importance of contextual information when classifying Dialogue Act tags (Mezza et al., 2018; Raheja and Tetreault, 2019). Grammatical features, namely POS and dependency tags, have a marginal impact on the Micro-F1 score on both our test sets; on the contrary, they appear to have a much higher impact on the Macro-F1 score, indicating that these features are beneficial for the classification of low-frequency DA tags. Prior information about the hierarchical relationship among DA labels appears to have a significant effect on out-of-domain DA tagging, while its influence on in-domain classification appears to be more limited. This result is compatible with our assumption that taking the hierarchical nature of the taxonomy into account helps with the generalisation of the model.

4.2 Training Set Variations

Table 5 shows the results of our experiments with various combinations of our dialogue corpora, in order to gain a better understanding of how each resource contributed to our final results, demonstrating the benefit of a general, cross-domain corpus. We trained the network on various subsets of GDC and tested the results on the DialogBank test set.

Training set	**Micro-F1 (DialogBank)**	**Macro-F1 (DialogBank)**
Without SWDA	77.3	48.9
Without MRDA	82.7	50.5
Without DailyDialog	83.6	53.0
Without Mastodon	79.6	52.8
Without IAC	83.4	54.9
Full GDC	**83.7**	**57.1**

Table 5: Data set variation experiments.

Our empirical results confirmed that each and every corpus in our collection contributed to some degree to the final accuracy of the model. SWDA and MRDA, being by far the largest resources in our aggregated training corpus, appear to have a significant impact on the model's Micro-F1 and Macro-F1 scores. The Mastodon corpus proved surprisingly impactful on the testing results given its small size; a possible explanation is that its annotation scheme was designed by taking the ISO standard into account (Cerisara et al., 2018), which makes the mapping less noisy and the resulting data points more similar to those in the DialogBank. The DailyDialog corpus and IAC appear to have a lesser effect on the classification accuracy, especially on the Micro-F1 score. However, when looking at accuracy on individual tags, their impact becomes more evident: the model trained without DailyDialog performed poorly on Action-Discussion tags when compared to the one trained on the full GDC, with a 34% increase in accuracy on the *Directive* label (from 8% to 42%) and a 15%

increase in accuracy on the *Commissive* label (from 10% to 25%). Similarly, IAC had an impact on the classification of DAs that are abundant in opinionated dialogue, with a 10% increase in the accuracy of the *Agree* label (from 25% to 35%). This is partially reflected in the Macro-F1 score decrease when training without these resources.

5 Conclusion

We have presented a multi-dimensional, cross-domain neural architecture for ISO-Standard Dialogue Act tagging, which leverages the hierarchical nature of the standard as well as grammatical, lexical and contextual information of the input utterances. We trained the model on a General Dialogue Corpus composed of different resources mapped to the ISO 24617-2 taxonomy, and showed how our model achieves state-of-the-art performance on out-of-domain data, which highlights its generalisation capabilities. The code and GDC data set have been released so as to help advance research on this topic. In the future, we plan to expand our work by covering more ISO-annotated corpora once they become available, as well as extending the *context encoder* module of our architecture to cover a wider context of dialogue and DA history. We also plan to experiment with different embedding mechanisms, such as BERT (Devlin et al., 2018) and RoBERTa (Liu et al., 2019).

References

Rob Abbott, Brian Ecker, Pranav Anand, and Marilyn Walker. 2016. Internet Argument Corpus 2.0: An SQL schema for Dialogic Social Media and the Corpora to go with it. In *Proceedings of the Tenth International Conference on Language Resources and Evaluation (LREC'16)*, pages 4445–4452.

James Allen and Mark Core. 1997. Draft of DAMSL: Dialog Act Markup in Several Layers, https://www.cs.rochester.edu/research/cisd/ resources/damsl/RevisedManual/, accessed on May 15, 2022.

Anne H Anderson, Miles Bader, Ellen Gurman Bard, Elizabeth Boyle, Gwyneth Doherty, Simon Garrod, Stephen Isard, Jacqueline Kowtko, Jan McAllister, Jim Miller, et al. 1991. The HCRC Map Task Corpus. *Language and Speech*, 34(4):351–366.

Tatiana Anikina and Ivana Kruijff-Korbayova. 2019. Dialogue Act Classification in Team Communication for Robot Assisted Disaster Response. In *Proceedings of the 20th Annual SIGdial Meeting on Discourse and Dialogue*, pages 399–410.

John Langshaw Austin. 1975. *How To Do Things With Words*. Oxford University Press, Oxford.

Philippe Blache, Massina Abderrahmane, Stéphane Rauzy, Magalie Ochs, and Houda Oufaida. 2020. Two-level Classification for Dialogue Act Recognition in Task-Oriented Dialogues. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 4915–4925.

Harry Bunt. 1989. Information Dialogues as Communicative Action in Relation to Information Processing and Partner Modelling. In M. Taylor, F. Néel and D. Bouwhuis, editor, *The Structure of Multimodal Dialogue*, pages 47–74. North-Holland Elsevier, Amsterdam.

Harry Bunt. 2005. A Framework for Dialogue Act Specification. *WG: Proceedings of the ACL SIGSEM Working Group on Representation of Multimodal Semantic Information.*

Harry Bunt. 2006. Dimensions in Dialogue Act Annotation. In *Proceedings of the Fifth International Conference on Language Resources and Evaluation (LREC'06)*, pages 919–924.

Harry Bunt. 2009. The DIT++ Taxonomy for Functional Dialogue Markup. In *AAMAS 2009 Workshop, Towards a Standard Markup Language for Embodied Dialogue Acts*, pages 13–24.

Harry Bunt, Jan Alexandersson, Jean Carletta, Jae-Woong Choe, Alex Chengyu Fang, Koiti Hasida, Kiyong Lee, Volha Petukhova, Andrei Popescu-Belis, Laurent Romary, et al. 2010. Towards an ISO Standard for Dialogue Act Annotation. In *Proceedings of the Seventh International Conference on Language Resources and Evaluation (LREC'10)*, pages 2548–2555.

Harry Bunt, Jan Alexandersson, Jae-Woong Choe, Alex Chengyu Fang, Koiti Hasida, Volha Petukhova, Andrei Popescu-Belis, and David Traum. 2012. ISO 24617-2: A Semantically-Based Standard for Dialogue Annotation. In *Proceedings of the Eighth International Conference on Language Resources and Evaluation (LREC'12)*, pages 430–437.

Harry Bunt, Volha Petukhova, Andrei Malchanau, Kars Wijnhoven, and Alex Fang. 2016. The DialogBank. In *Proceedings of the Tenth International Conference on Language Resources and Evaluation (LREC'16)*, pages 3151–3158.

Harry Bunt, Volha Petukhova, David Traum, and Jan Alexandersson. 2017. Dialogue Act Annotation with the ISO 24617-2 Standard. In Deborah A. Dahl, editor, *Multimodal Interaction with W3C Standards*, pages 109–135. Springer, Cham.

Christophe Cerisara, Somayeh Jafaritazehjani, Adedayo Oluokun, and Hoa T Le. 2018. Multi-task Dialog Act and Sentiment Recognition on Mastodon. In *Proceedings of the 27th International Conference on Computational Linguistics,*, pages 745–754.

Alessandra Cervone, Evgeny Stepanov, and Giuseppe Riccardi. 2018. Coherence Models for Dialogue. In *Proceedings of Interspeech 2018*, pages 1011–1015.

Zheqian Chen, Rongqin Yang, Zhou Zhao, Deng Cai, and Xiaofei He. 2018. Dialogue Act Recognition via CRF-Attentive Structured Network. In *Proceedings of the 41st International ACM SIGIR Conference on Research & Development in Information Retrieval*, pages 225–234.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. BERT: Pre-Training of Deep Bidirectional Transformers for Language Understanding. *arXiv preprint arXiv:1810.04805*.

Alex Fang, J Cao, H Bunt, and Xiaoyue Liu. 2012. Applicability Verification of a New ISO Standard for Dialogue Act Annotation with the Switchboard Corpus. In *Proceedings of the EACL 2012 Workshop on Innovative Hybrid Approaches to the Processing of Textual Data*.

Emer Gilmartin, Christian Saam, Brendan Spillane, Maria O'Reilly, Ketong Su, Arturo Calvo Devesa, Loredana Cerrato, Killian Levacher, Nick Campbell, and Vincent Wade. 2018. The ADELE Corpus of Dyadic Social Text Conversations: Dialog Act Annotation with ISO 24617-2. In *Proceedings of the Eleventh International Conference on Language Resources and Evaluation (LREC 2018)*, pages 4016–4022.

Susanne Jekat, Alexandra Klein, Elisabeth Maier, Ilona Maleck, Marion Mast, and J Joachim Quantz. 1995. Dialogue Acts in VERBMOBIL. *Technical report, Universität des Saarlandes, Saarbrücken*.

Daniel Jurafsky and Elizabeth Shriberg. 1997. Switchboard SWBD-DAMSL Shallow-Discourse-Function Annotation Coders Manual, Draft 13, University of Colorado at Boulder, SRI International.

Harshit Kumar, Arvind Agarwal, Riddhiman Dasgupta, and Sachindra Joshi. 2018. Dialogue Act Sequence Labeling using Hierarchical encoder with CRF. In *Proceedings of the Thirty-Second AAAI Conference on Artificial Intelligence*, pages 3440–3447.

Piroska Lendvai, Antal van den Bosch, and Emiel Krahmer. 2003. Machine Learning for Shallow Interpretation of User Utterances in Spoken Dialogue Systems. In *Proceedings of the 2003 EACL Workshop on Dialogue Systems: Interaction, Adaptation and Styles of Management*, pages 69–78.

Yanran Li, Hui Su, Xiaoyu Shen, Wenjie Li, Ziqiang Cao, and Shuzi Niu. 2017. DailyDialog: A Manually Labelled Multi-turn Dialogue Dataset. In *Proceedings of the Eighth International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 986–995.

Zuchao Li, Shexia He, Jiaxun Cai, Zhuosheng Zhang, Hai Zhao, Gongshen Liu, Linlin Li, and Luo Si. 2018. A Unified Syntax-aware Framework for Semantic Role Labeling. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 2401–2411.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. RoBERTa: A Robustly Optimized BERT Pre-Training Approach. *arXiv preprint arXiv:1907.11692*.

Stefano Mezza, Alessandra Cervone, Evgeny Stepanov, Giuliano Tortoreto, and Giuseppe Riccardi. 2018. ISO-Standard Domain-Independent Dialogue Act Tagging for Conversational Agents. In *Proceedings of the 27th International Conference on Computational Linguistics*, pages 3539–3551.

Shachi Paul, Rahul Goel, and Dilek Hakkani-Tür. 2019. Towards Universal Dialogue Act Tagging for Task-Oriented Dialogues. *arXiv preprint arXiv:1907.03020*.

Volha Petukhova, Martin Gropp, Dietrich Klakow, Anna Schmidt, Gregor Eigner, Mario Topf, Stefan Srb, Petr Motlicek, Blaise Potard, John Dines, et al. 2014. The DBOX Corpus Collection of Spoken Human-Human and Human-Machine Dialogues. In *Proceedings of the Ninth International Conference on Language Resources and Evaluation (LREC'14)*, pages 252–258.

Silvia Quarteroni, Alexei V Ivanov, and Giuseppe Riccardi. 2011. Simultaneous Dialog Act Segmentation and Classification from Human-Human Spoken Conversations. In *Proceedings of the 2011 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 5596–5599.

Vipul Raheja and Joel Tetreault. 2019. Dialogue Act Classification with Context-Aware Self-Attention. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 3727–3733.

Eugénio Ribeiro, Ricardo Ribeiro, and David Martins de Matos. 2019. Hierarchical Multi-Label Dialog Act Recognition on Spanish Data. *arXiv preprint arXiv:1907.12316*.

Eugénio Ribeiro, Ricardo Ribeiro, and David Martins de Matos. 2022. Automatic Recognition of the General-Purpose Communicative Functions defined by the ISO 24617-2 Standard for Dialog Act Annotation. *Journal of Artificial Intelligence Research*, 73:397–436.

Eugénio Ribeiro, Ricardo Ribeiro, and David Martins de Matos. 2020. Mapping the Dialog Act Annotations of the LEGO Corpus into ISO 24617-2 Communicative Functions. In *Proceedings of the 12th Language Resources and Evaluation Conference*, pages 531–539.

John R Searle. 1965. What is a Speech Act? In Maurice Black, editor, *Philosophy in America*, pages 221–239. Allen and Unwin, London.

Elizabeth Shriberg, Raj Dhillon, Sonali Bhagat, Jeremy Ang, and Hannah Carvey. 2004. The ICSI Meeting Recorder Dialog Act (MRDA) Corpus. Technical report, International Computer Science Institute, Berkeley, CA.

Claudia Soria and Vito Pirrelli. 2003. A Multi-Level Annotation Meta-Scheme for Dialogue Acts. In *Linguistica Computazionale, vol. XVIII-XIX*, pages 865–900.

Andreas Stolcke, Klaus Ries, Noah Coccaro, Elizabeth Shriberg, Rebecca Bates, Daniel Jurafsky, Paul Taylor, Rachel Martin, Carol Van Ess-Dykema, and Marie Meteer. 2000. Dialogue Act Modeling for Automatic Tagging and Recognition of Conversational Speech. *Computational Linguistics*, 26(3):339–373.

Kai Sheng Tai, Richard Socher, and Christopher D Manning. 2015. Improved Semantic Representations From Tree-Structured Long Short-Term Memory Networks. *arXiv preprint arXiv:1503.00075*.

David R Traum. 2000. 20 Questions on Dialogue Act Taxonomies. *Journal of Semantics*, 17(1):7–30.

Marilyn Walker, Jean E Fox Tree, Pranav Anand, Rob Abbott, and Joseph King. 2012. A Corpus for Research on Deliberation and Debate. In *Proceedings of the Eighth International Conference on Language Resources and Evaluation (LREC'12)*, pages 812–817.

Dong Wang, Ziran Li, Dongming Sheng, Hai-Tao Zheng, and Ying Shen. 2021. Balance the Labels: Hierarchical Label Structured Network for Dialogue Act Recognition. In *Proceedings of the 2021 International Joint Conference on Neural Networks (IJCNN)*.

Dian Yu and Zhou Yu. 2019. MIDAS: A Dialog Act Annotation Scheme for Open Domain Human Machine Spoken Conversations. *arXiv preprint arXiv:1908.10023*.

Jie Zhou, Chunping Ma, Dingkun Long, Guangwei Xu, Ning Ding, Haoyu Zhang, Pengjun Xie, and Gongshen Liu. 2020. Hierarchy-Aware Global Model for Hierarchical Text Classification. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 1106–1117.

Yucan Zhou, Qinghua Hu, Jie Liu, and Yuan Jia. 2014. Learning Conditional Random Field with Hierarchical Representations for Dialogue Act Recognition. In *Proceedings of the Fifteenth Annual Conference of the International Speech Communication Association*, pages 1920–1923.

SPACE-2: Tree-Structured Semi-Supervised Contrastive Pre-training for Task-Oriented Dialog Understanding

Wanwei He[1,2,✊], Yinpei Dai[3,✊], Binyuan Hui[3,✊], Min Yang[1]*, Zheng Cao[3], Jianbo Dong[3], Fei Huang[3], Luo Si[3], Yongbin Li[3]*

[1]Shenzhen Institute of Advanced Technology, Chinese Academy of Sciences, China
[2]University of Chinese Academy of Sciences, China
[3]DAMO Academy, Alibaba Group

{ww.he, min.yang}@siat.ac.cn
{yinpei.dyp, f.huang, luo.si, shuide.lyb}@alibaba-inc.com

Abstract

Pre-training methods with contrastive learning objectives have shown remarkable success in dialog understanding tasks. However, current contrastive learning solely considers the self-augmented dialog samples as positive samples and treats all other dialog samples as negative ones, which enforces dissimilar representations even for dialogs that are semantically related. In this paper, we propose **SPACE-2**, a tree-structured pre-trained conversation model, which learns dialog representations from limited labeled dialogs and large-scale unlabeled dialog corpora via semi-supervised contrastive pre-training. Concretely, we first define a general semantic tree structure (STS) to unify the inconsistent annotation schema across different dialog datasets, so that the rich structural information stored in all labeled data can be exploited. Then we propose a novel multi-view score function to increase the relevance of all possible dialogs that share similar STSs and only push away other completely different dialogs during supervised contrastive pre-training. To fully exploit unlabeled dialogs, a basic self-supervised contrastive loss is also added to refine the learned representations. Experiments show that our method can achieve new state-of-the-art results on the DialoGLUE benchmark consisting of seven datasets and four popular dialog understanding tasks. For reproducibility, we release the code and data at https://github.com/AlibabaResearch/DAMO-ConvAI/tree/main/space-2.

1 Introduction

Task-oriented dialog (TOD) systems aim to help users to accomplish specific tasks through natural language interactions, *e.g.*, restaurant booking, hotel reserving, and movie searching (Young et al., 2013). In order to fulfill a user goal, the dialog systems must be capable of extracting structured semantics from the dialog utterances, which is often referred to as *dialog understanding*. Common tasks of dialog understanding include: 1) intent prediction, where the system classifies the user intention given the current utterance, 2) slot filling, where the system extracts the value spans for predefined slots from user utterances, and 3) dialog state tracking, where slot-value pairs are predicted and updated according to multi-turn dialog history.

Recent methods address the dialog understanding problems mainly by task-adaptive training with pre-trained language models (PLMs) (Henderson et al., 2020; Mehri et al., 2020; Zhang et al., 2022). By fine-tuning the entire parameters of PLMs on dialog corpora, the versatile knowledge acquired from large-scale corpora can be adapted to handle various dialog understanding tasks. For example, TOD-BERT (Wu et al., 2020) continually pre-trained a BERT on large-scale task-oriented dialog corpora with similar objectives to grasp more knowledge in conversations and achieved good results on a wide range of tasks. To further enhance the representations of dialogs, contrastive learning is widely used for dialog pre-training via pulling semantically similar sentences together and pushing apart dis-similar ones, which has shown strong ability in few-shot dialog understanding (Zhang et al., 2020, 2021a).

Although contrastive learning (CL) is effective in pre-training dialog models, it still suffers from several problems. First, the current practice of CL often neglects the semantic structures of dialog examples and only utilizes shallow self-supervised information. Figure 1 illustrates the semantic frames of two dialog examples, which indicate the hierarchical structures with semantic meanings in terms of intentions and slots. The semantic structure is often summarized as the turn-level structural annotation and is prominent in multi-domain task-oriented dialogs. Exploiting such structure during

* Corresponding authors.
✊ Equal contribution.

Proceedings of the 29th International Conference on Computational Linguistics, pages 553–569
October 12–17, 2022.

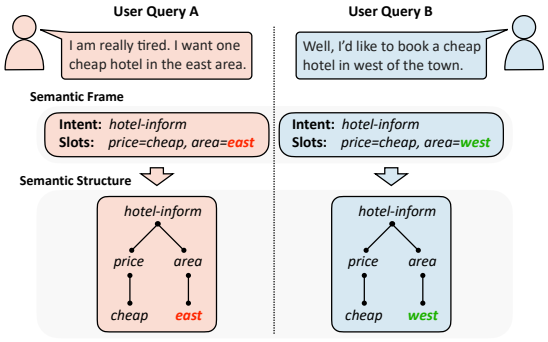

Figure 1: Two dialog examples that have similar semantic contents from the MULTIWOZ dataset. The difference is highlighted in red and green colors.

task-adaptive pre-training has been proved to be effective in dialog understanding tasks (Yu et al., 2020). However, In conventional CL, only the examples with the exact same annotations are deemed as positive samples, while all other examples are considered as negative ones (Gao et al., 2021), ignoring the fact that different examples may share similar semantics to some extent as shown in Figure 1. Second, in previous dialog pre-training methods, no annotations have been exploited to learn better pre-trained representations. But in computer vision, there have already been many works demonstrating that labeled data can accelerate the pre-training procedure as well as improve the model performance (Assran et al., 2020; Khosla et al., 2020; Dai et al., 2021b). Therefore, we argue that it is crucial to combine both the labeled and unlabeled dialog data to learn more powerful pre-trained dialog models.

Built on these motivations, we propose a novel tree-structured pre-trained conversation model (SPACE-2), which fully exploits the structural information in labeled data to improve the dialog pre-training on large-scale unlabeled corpora via semi-supervised contrastive learning. To be specific, we first define a general semantic tree structure (STS), which is suitable for TOD and compatible with the discrepancy of annotation schema in varied dialog datasets. Then we propose a novel multi-view scoring function to measure the similarity among different labeled dialogs. Given two dialogs with STSs, the scoring function is able to calculate their relations hierarchically and aggregate the output multiple scores from either global or local aspects. In this way, all labeled data are viewed as positive samples with soft scores instead of hard scores (0 or 1) in traditional CL. Thus, more subtle semantic structures in training samples can be taken into account, and dialog representations can be learned in a semi-supervised manner.

Our contributions are summarized as follows:

- We explore tree-structured semi-supervised contrastive pre-training for task-oriented dialog understanding. To the best of our knowledge, this is the first study to inject structural information and exploit labeled semantic frames in a pre-trained conversation model.
- We propose a multi-view score function for similarity measurement among different labeled dialogs, which could calculate relations hierarchically and aggregate multiple judgements from either global or local aspects.
- We conduct extensive experiments on the DialoGLUE benchmark with seven different datasets. Empirical results show that our SPACE-2 consistently performs better than strong competitors on four dialog understanding tasks (i.e., intent prediction, slot filling, semantic parsing, and dialog state tracking).

2 Related Work

2.1 Pre-trained Models for Dialog Understanding

Recent advances in pre-trained language models (PLMs) have spurred success in natural language understanding in task-oriented dialog (TOD) systems. Since general-purpose PLMs such as BERT (Devlin et al., 2019) are not able to capture dialog-oriented knowledge in TOD, such as explicit user intent, many studies proposed to perform pre-training on dialog corpora. For example, ConvRT (Henderson et al., 2020) explored the pre-trained language transformers for TOD via learning a dialog response selection loss. ConvBERT (Mehri et al., 2020) fine-tuned BERT on a large open-domain dialog corpus using a masked language modeling objective so that more semantically meaningful representations of dialogs can be learned. TOD-BERT (Wu et al., 2020) incorporated role tokens in the input and employed a contrastive objective for response selection. Zhang et al. (2021b) proposed to use masked utterance modeling and response contrastive loss to produce DialogueBERT and showed good performance on intent, emotion, and entity recognition. Apart from training with common objectives, some work (Yu et al., 2020; He et al., 2020b; Hui et al., 2021; Dai et al., 2021a) also

Statistics	*AnPreDial*	*UnPreDial*
# Datasets	32	21
# Dialogs	459,465	3,217,058
# Turns	3,366,479	19,578,028
Avg. tokens per turn	13.9	14.5
Avg. tokens per dialog	101.8	88.2
Total unique tokens	46.8M	283.7M

Table 1: Statistics of our pre-trained dialog dataset.

proposed to learn relations between schema and keywords of utterances during pre-training, which benefits in certain downstream tasks that require parsing semantics. The work of (Sun et al., 2019; Liu et al., 2020a) tried to integrate structural information of knowledge graphs to enhance language representations. Unlike these methods, we are the first to inject the semantic tree structure into pre-training models for dialog understanding.

2.2 Contrastive Learning

Contrastive learning (CL) is one of the pre-training methods that leverage large-scale unlabeled data to learn meaningful sentence representations. The key idea of CL is to narrow the distance between two semantically similar sentence representations, meanwhile, push away the representations of dissimilar sentences (Gao et al., 2021; Wu et al., 2021; Yan et al., 2021). There are also several studies focused on applying CL to dialog understanding tasks. For example, Zhang et al. (2021a) used CL as self-supervised pre-training and adapted the pre-trained models on few-shot intent prediction. The studies (Mehri and Eric, 2021; Vulić et al., 2021) utilized supervised CL at the fine-tuning stages for intent prediction, where the samples from the same class are all regarded as positives. He et al. (2020a) proposed to use adversarial samples as difficult negative instances in CL for zero-shot cross-domain slot-filling. Different from previous work, we use CL in both pre-training and fine-tuning stages. In addition, we train dialog models with both labeled and unlabeled data in a semi-supervised manner.

2.3 Semi-Supervised Contrastive Learning

Semi-supervised contrastive learning has proved to be an effective pre-training method in many research field, such as image recognition (Yuan et al., 2021; Park et al., 2021; Li et al., 2021; Kim et al., 2021a), image segmentation (Alonso et al., 2021; Zhou et al., 2021) and speech recognition (Xiao et al., 2021; Inoue and Goto, 2020). It can combine both labeled and unlabeled data together to train effective neural models for downstream tasks. He et al. (2022b) proposed to use consistency regularization loss to learn dialog policy from labeled and unlabeled dialog corpora via a semi-supervised manner. However, leveraging such a semi-supervised contrastive learning paradigm in task-oriented dialog understanding remains unexplored. In this paper, we explore semantic structures in TOD, and propose a new contrastive learning approach that calculates semantic tree-structure similarity among all possible labeled dialog data.

3 Pre-training Data

In this section, we first describe the dialog datasets (He et al., 2022a) used for pre-training, including a labeled dialog dataset (*AnPreDial*) and a large-scale unlabeled dialog corpus (*UnPreDial*). Then, we elaborate on the semantic tree structure (STS) for the task-oriented dialog data.

3.1 Dataset Description

To provide sufficient high-quality dialog corpora to train our pre-trained conversation model, we use a labeled dialog dataset called *AnPreDial*, which contains 32 existing labeled TOD datasets, ranging from single-turn question answering to multi-turn dialogs. We also use a large-scale unlabeled dialog corpus called *UnPreDial* with 21 dialog datasets, ranging from open-domain online forums to document-grounded dialogs. The statistics of our pre-trained dialog corpora are shown in Table 1 and more details can be found in Appendix A.

3.2 Semantic Tree Structure

Task-oriented dialogs often contain several user goals, and the system needs to help fulfill these goals through interacting with users. To this end, the user utterances and system responses are usually composed of rich semantics such as intents and slots. In different datasets, the annotation taxonomy used to describe the semantic scopes varies. For example, in some intent recognition datasets such as BANKING77 (Casanueva et al., 2020) and CLINIC150 (Larson et al., 2019), they only possess a pre-defined intent set and formulate the task as a classification problem. In other datasets like Taskmaster (Byrne et al., 2019a), only the slot labels are provided; for instance, given a user utterance *"I need a table for tonight at 7 pm for 8 people"*, the label is *"time.reservation=7*

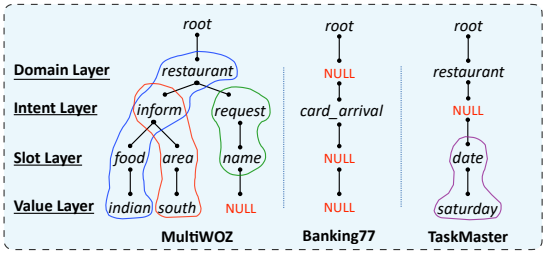

Figure 2: Different semantic tree structures in *AnPreDial*, where the blue, red, green and purple circles indicate DISV, ISV, IS, SV sub-structures.

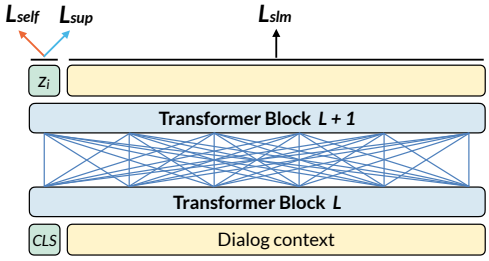

Figure 3: Model Framework. L_{self} is the tree-structured self-supervised contrastive loss for unlabeled data, and L_{sup} is the tree-structured supervised contrastive loss for labeled data. L_{slm} is the span MLM loss for all data.

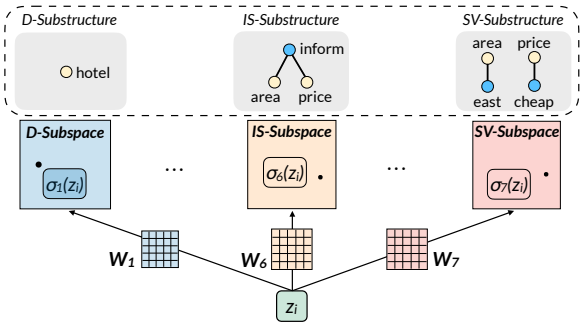

Figure 4: Sub-spaces for the multi-view scoring objective. The dashed black line denotes the annotations of different sub-structures, only used in our supervised CL.

pm, num.guests=8" without any intent information. Therefore, in order to utilize all possible data in *AnPreDial* to pre-train our model, we adopt a unified schema called semantic tree structure (STS) that is suitable for every TOD sample.

Typically, the STS contains four layers: *domain* layer, *intent* layer, *slot* layer, and *value* layer. Every layer is composed of nodes, which are referred to as respective elements of labeled semantic frames. If no matched annotations are provided, the nodes of the corresponding layer are set as empty (denoted as NULL). The first layer consists of domain nodes as successors of the root node, preceded by the intent nodes. As children of intent nodes, slot nodes occupy the third layer of the semantic tree. As leaf nodes, the value nodes take the slot nodes as parents. Figure 2 shows three tree structures of different dialog samples from MULTIWOZ (Eric et al., 2020), BANKING77 and Taskmaster. In MULTIWOZ , the labels follow the dialog act schema in Young (2007), and all layers contain certain semantic elements. But the other two datasets do not follow the schema and lack respective elements in some layers.

4 Methods

In this section, we first introduce the model architecture. Then we expound on the pre-training procedure with the proposed tree-structured semi-supervised contrastive learning (CL) paradigm.

4.1 Model Architecture

As illustrated in Figure 3, we build our SPACE-2 model based on the bidirectional Transformer architecture (Vaswani et al., 2017). Different from the vanilla input representations as in BERT (Devlin et al., 2019), we set our input embeddings consisting of four elements: tokens, roles, turns, and positions. Role embeddings are used to segment which role the current token belongs to either user or system. Turn embeddings are assigned to each token according to its turn number in the dialog. Position embeddings are assigned to each token according to its relative position within its belonging sentence. Appendix B shows more details.

4.2 Tree-Structured Semi-Supervised Contrastive Learning Paradigm

We aim to leverage semi-supervised pre-training to learn better pre-trained representations from both the labeled and unlabeled data. Concretely, we adopt a tree-structured supervised contrastive objective on the labeled dataset *AnPreDial*, while a tree-structured self-supervised contrastive objective on the unlabeled dataset *UnPreDial*.

4.2.1 Tree-Structured Supervised CL

For labeled data, we treat every dialog sample as a positive sample assigned with similarity scores to each other. Between two semantic tree structures (STSs), we define a score function to calculate similarity scores from different views. Let D, I, S, V denote the sets of node in domain, intent, slot and value layer. For the sample from MULTIWOZ dataset in Figure 2, its D={*restaurant*}, I={*inform*,

request}, S={*food, area, name*} and V={*indian, south*}. For the sample from BANKING77 dataset, its D={}, I={*card_arrival*}, S={} and V={}. We also build the sets of each possible path between two nodes on the STS, which we denote as DI, IS, SV, DIS, ISV, DISV. Thus for the MULTIWOZ sample in Figure 2, ISV={*inform→food→indian, inform→area→south*}, and for BANKING77 , ISV={}. The total super set of all above node-sets and path-sets is denoted as $\mathcal{S}_{all} = \{\mathtt{D}, \mathtt{I}, \mathtt{S}, \mathtt{V}, \mathtt{DI}, \mathtt{IS}, \mathtt{SV}, \mathtt{DIS}, \mathtt{ISV}, \mathtt{DISV}\}$, where the size $K = 10$. Given a pair of STSs (denoted as T_i, T_j), then we calculate the Jaccard similarity coefficient for each set s_k in $\mathcal{S}_{all}$ to get the score f_k respectively as follows:

$$f_k(T_i, T_j) = \mathrm{J}(s_k^i, s_k^j) \tag{1}$$

$$\mathrm{J}(A, B) = \frac{|A \cap B|}{|A \cup B|} \tag{2}$$

where we use i, j to discriminate s_k and $\mathcal{S}_{all}$ for two different STSs: $s_k^i \in \mathcal{S}_{all}^i, s_k^j \in \mathcal{S}_{all}^j$. J is the function for the Jaccard similarity coefficient, which takes the ratio of Intersection over Union.

Given the dialog context, our model output a pooled representation at the [CLS] position as the sentence embedding z of the whole context. Thus for any sample pair i, j, the output sentence embeddings are denoted as z_i, z_j. We use $f_{i,j}^k$ to denote the k-th similarity score $f_k(T_i, T_j)$ for simplicity. Suppose the size of the current batch is N, we follow the common practice in Gao et al. (2021) to duplicate the data via dropout-based data augmentation. Then we acquire a $2N$-length new batch. Let $I = \{1 \dots 2N\}$ be the index set of the new batch, and $C(i) \equiv I/\{i\}$. Since we compute multiple scores $f_{i,j}^k$ for each STS pair, different types of scoring functions can be used to construct the CL loss. In this paper, We propose both single-view and multi-view scoring functions to build the tree-structured supervised CL objectives as follows and find that the multi-view method performs better.

Single-View Scoring Objective. Similar to the common practice in the current CL, we simply average the K scores into a single value to weigh the semantic similarity among samples:

$$L_{sup}^{single} = -\sum_{i \in I} \sum_{j \in C(i)} \left(\frac{1}{K} \sum_{k=1}^{K} f_{i,j}^k \right) \times \log \frac{\exp\left(\sigma(z_i) \cdot \sigma(z_j)/\tau\right)}{\sum_{l \in C(i)} \exp\left(\sigma(z_i) \cdot \sigma(z_l)/\tau\right)} \tag{3}$$

$$\sigma(z) = \mathrm{Norm}(Wz + b) \tag{4}$$

where $\sigma(\cdot)$ is a normalized linear mapping to transform z to a unit vector. W is a learnable square weight matrix and b is a bias vector. $\mathrm{Norm}(\cdot)$ is the normalization operation and $\tau \in [0, 1]$ is a temperature hyper-parameter.

Multi-View Scoring Objective. Figure 4 demonstrates our multi-view scoring objective. Instead of summing up as one score, we can separately utilize multiple scores to better exploit sub-structures from different semantic spaces (Zhu et al., 2019) as the following loss:

$$L_{sup}^{multi} = -\sum_{i \in I} \sum_{j \in C(i)} \sum_{k=1}^{K} \left(\frac{f_{i,j}^k}{\sum_{m \in C(i)} f_{i,m}^k} \times \log \frac{\exp\left(\sigma_k(z_i) \cdot \sigma_k(z_j)/\tau\right)}{\sum_{l \in C(i)} \exp\left(\sigma_k(z_i) \cdot \sigma_k(z_l)/\tau\right)} \right) \tag{5}$$

$$\sigma_k(z) = \mathrm{Norm}(W_k z + b_k) \tag{6}$$

where $\sigma_k(\cdot)$ denotes linear mappings with different parameters. As illustrated in Figure 5, our methods are quite distinct from vanilla supervised CL (Vulić et al., 2021), where only augmented data or data from the same label are considered as positive samples ($f_{i,j}^k = 1$), and all other data are considered as negative samples ($f_{i,j}^k = 0$). Therefore, our method can be viewed as a generalized case where all labeled data are regarded as positive samples assigned with soft scores.

4.2.2 Tree-Structured Semi-Supervised CL

For unlabeled data, since there are no available labels, we adopt a tree-structured self-supervised contrastive objective in a similar way in Gao et al. (2021), where only the augmented data by dropout is deemed as a positive sample. In particular, under the multi-view setting, the objective L_{self}^{multi} still shares the same K sub-spaces as L_{sup}^{multi} as follows:

$$L_{self}^{single} = -\sum_{i \in I} \sum_{j \in C(i)} \mathbf{1}(j = i^+) \times \log \frac{\exp\left(\sigma(z_i) \cdot \sigma(z_j)/\tau\right)}{\sum_{l \in C(i)} \exp\left(\sigma(z_i) \cdot \sigma(z_l)/\tau\right)} \tag{7}$$

$$L_{self}^{multi} = -\sum_{i \in I} \sum_{j \in C(i)} \sum_{k=1}^{K} \mathbf{1}(j = i^+) \times \log \frac{\exp\left(\sigma^k(z_i) \cdot \sigma^k(z_j)/\tau\right)}{\sum_{l \in C(i)} \exp\left(\sigma^k(z_i) \cdot \sigma^k(z_l)/\tau\right)} \tag{8}$$

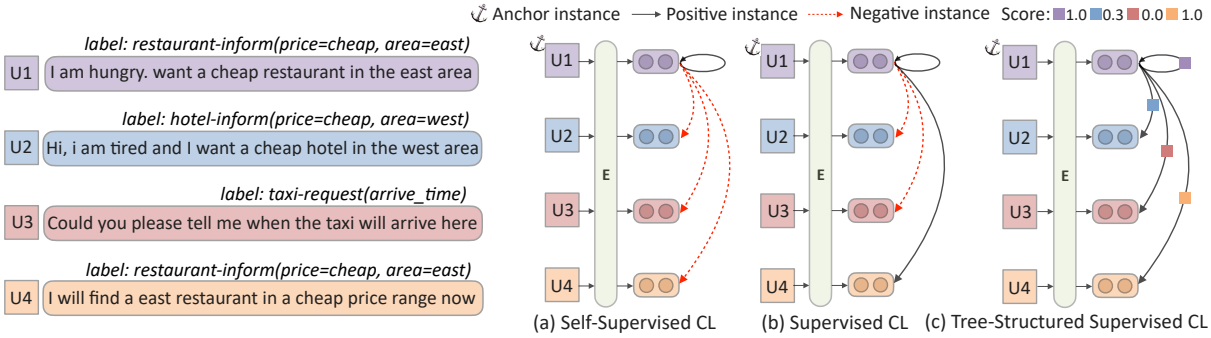

Figure 5: (a) Self-supervised CL only predicts augmented itself from in-batch negatives, with different dropout masks applied. (b) Supervised CL considers samples of the exact same label as positives. (c) Tree-structured supervised CL considers all in-batch samples as positives with soft scores. Only scores of `ISV` set are depicted here.

where $\mathbf{1}(j = i^+)$ means it only takes 1 when the data sample j is the augmented data i^+ of data sample i, otherwise 0.

In order to maintain the language modeling ability and extract the slot values better, we also apply the spanMLM loss L_{slm} (Joshi et al., 2020) to the output representations on top of the dialog context. Therefore, our final loss is computed as:

$$L^*_{total} = L^*_{sup} + L^*_{self} + L_{slm} \quad (9)$$

where * can be either *single* or *multi*.

5 Experimental Setup

5.1 Task Formulation

DialoGLUE benchmark consists of seven different datasets, including four tasks as follows:

Intent Prediction. This is a classification task for models to predict which intent class the user conveys in the utterance. There are three datasets accordingly: 1) BANKING77 (Casanueva et al., 2020), a single turn dataset in the banking domain with 77 intent labels and 13k utterances; 2) CLINIC150 (Larson et al., 2019), a multi-domain dataset that contains 23.7k utterances, 150 intents, 10 domains; and 3) HWU64 (Liu et al., 2021) includes 25.7k utterances for 64 intents spanning 21 domains. To conduct the downstream task, a linear layer neural network is added to the pooled output z to classify. The evaluation metric is accuracy.

Slot Filling. This is a sequential labeling task in which a model tries to predict the IOB tags upon the sequence of dialog context. There are two datasets: 1) REST8K (Coope et al., 2020), which contains 8.2k utterances and 5 slots (date, time, people, first name, last name) from a restaurant domain; 2) a transformed DSTC8 (Rastogi et al., 2020) dataset comprising of 5.5k utterances and slot annotations in 4 domains (buses, events, homes, rental cars). Both datasets are single-turn formed and every token in the utterance is predicted as either being the beginning of a slot value (B-), inside a slot value (I-) or not belonging to a slot value (O). A linear layer neural network is added to the representation of the final output layer to predict correct tags (e.g., "B-time", "I-people"). The evaluation metric is the macro-averaged F1 score.

Semantic Parsing. The original TOP (Gupta et al., 2018) dataset targeting at hierarchical parsing is transformed into a joint task of intent prediction and slot-filling, which contains 44k utterances. The metric is the exact-match as in Mehri et al. (2020).

Dialog State Tracking. This task is multi-turn slot-filling, where the model interprets the user meanings during the course of a dialog to maintain a dialog state in terms of slot-value pairs. MULTIWOZ 2.1 (Eric et al., 2020) is the chosen dataset version with 10k dialogs. Following Mehri et al. (2020), we report joint goal accuracy and use TripPy (Heck et al., 2020) as the downstream model.

5.2 Baselines

Our compared baselines can be divided into two categories. The first type is the pre-trained model adapted for general dialog understanding tasks in DialoGLUE. We choose BERT (Devlin et al., 2019), ConvBERT (Mehri et al., 2020) and its variants, and TOD-BERT (Wu et al., 2020) as our baselines. The second type is the pre-trained model designed only for specific dialog tasks. In this paper, we choose Example+Observer (Mehri and Eric, 2021), DNNC (Zhang et al., 2020), CPFT (Zhang et al., 2021a); and ConvFit (Vulić et al., 2021) for the intent prediction task. We also choose Span-ConveRT (Coope et al., 2020), ConVEx (Hender-

Full data training								
Model	*average*	BANKING77	HWU64	CLINIC150	REST8K	DSTC8	TOP	MULTIWOZ
BERT†	86.08	93.02	89.87	95.93	95.53	90.05	81.90	56.30
ConvBERT†	86.01	92.95	90.43	97.07	95.90	87.58	82.13	56.00
ConvBERT+MLM†	86.89	93.44	92.38	97.11	95.44	91.20	82.08	56.56
ConvBERT-DG†	82.90	93.21	91.64	96.96	93.44	74.54	72.22	58.57
ConvBERT-DG+MLM†	85.34	92.99	91.82	97.11	94.34	86.49	76.36	58.29
TOD-BERT$_{adapt}$	86.95	93.61	92.95	97.34	94.23	90.61	82.33	57.63
ConvBERT+MLM$_{adapt}$	87.42	94.01	93.24	97.40	95.55	91.12	82.64	58.01
SPACE-2^{single}	87.83	94.68	94.05	97.56	95.58	91.20	82.66	59.08
SPACE-2^{multi}	**88.10**	**94.77**	**94.33**	**97.80**	**96.20**	**91.38**	**82.74**	**59.51**
w/o *tree*	87.57	94.19	93.49	97.44	96.08	91.04	82.23	58.52
w/o *annotation*	87.33	93.93	93.23	97.33	95.99	90.93	82.02	57.87
w/o *UnPreDial*	87.77	94.35	93.77	97.64	96.14	91.18	82.30	58.99
10-shot data training								
Model	*average*	BANKING77	HWU64	CLINIC150	REST8K	DSTC8	TOP	MULTIWOZ
BERT†	66.07	79.87	81.69	89.52	87.28	45.05	74.38	4.69
ConvBERT†	68.03	83.63	83.77	92.10	86.90	49.08	74.86	5.90
ConvBERT+MLM†	68.22	83.99	84.52	92.75	86.17	48.40	78.84	6.87
ConvBERT-DG†	73.75	84.42	85.17	92.87	87.65	41.94	75.27	48.94
ConvBERT-DG+MLM†	73.80	85.06	85.69	93.06	87.58	44.36	72.01	48.89
TOD-BERT$_{adapt}$	75.70	85.99	86.74	93.07	87.62	50.19	77.77	48.54
ConvBERT+MLM$_{adapt}$	76.51	86.73	87.29	93.65	87.72	52.48	78.88	48.85
SPACE-2^{single}	77.43	88.08	88.29	94.80	88.52	53.27	79.01	50.07
SPACE-2^{multi}	**77.98**	**88.31**	**88.85**	**95.22**	**88.85**	**54.41**	**79.55**	**50.70**
w/o *tree*	77.31	87.41	87.55	94.76	88.30	54.26	78.95	49.97
w/o *annotation*	77.20	87.60	87.64	94.51	87.93	54.01	79.00	49.73
w/o *UnPreDial*	77.65	88.15	88.38	94.71	88.61	54.12	79.33	50.28

Table 2: Total results of seven datasets from the DialoGLUE benchmark. † denotes original results from DialoGLUE. $_{adapt}$ denotes re-implementation via adaptively pre-training on our corpora. 'w/o' denotes ablation study in Sec 6.4

son and Vulić, 2021) and GenSF (Mehri and Eskenazi, 2021) for the slot-filling task. More details about baselines can be found in Appendix C.

5.3 Settings

In our experiment, we initialize SPACE-2 with ConvBERT. The input length is bound to 256 and the batch size is 128. AdamW is used for optimization with an initial learning rate of 1e-5. The dropout rate is 0.2. For semi-supervised pre-training, we combine the labeled and unlabeled data to form a batch in a 1:1 ratio. For each few-shot experiment, we **exclude** the training data in the *AnPreDial* accordingly to avoid unfair data use. For instance, if the target few-shot task is HWU64 , we only use the few-shot training data of HWU64 and all other datasets as pre-training resources, so no extra training data (dev&test) in HWU64 is used. For all downstream tasks, we average 5 seeds for final results.

6 Experimental Results

6.1 Overall Performance on DialoGLUE

We follow the original settings in Mehri et al. (2020) and obtain complete results for our model with single-view scoring objectives, denoted as SPACE-2^{single}, and with multi-view scoring objectives, denoted as SPACE-2^{multi}, on all seven datasets of DialoGLUE. As shown in Table 2, our SPACE-2^{multi} model achieves state-of-the-art performance across all tasks, improving the average score of previous best results by **1.39%** (86.89%→88.10%) and **5.66%** (73.80%→77.98%) on full-data training and 10-shot training settings, respectively. This indicates that our semi-supervised contrastive learning can acquire better pre-trained representations to adapt efficiently to downstream dialog tasks by utilizing both labeled and unlabeled data. Although the superior results of SPACE-2^{single} over other baselines, it is worse than SPACE-2^{multi}, suggesting that multi-view scoring is crucial to improve the overall performance. To eliminate the discrepancy brought by

Model	BANKING77		CLINIC150		HWU64	
	Few	Full	Few	Full	Few	Full
Example+Observer	85.95	93.83	93.97	97.31	86.28	93.03
DNNC	86.71	-	93.76	-	84.72	-
CPFT	87.20	-	94.18	-	87.13	-
ConvFiT	87.38	94.16	92.89	97.34	85.32	92.42
SPACE-2^{multi}	**88.31**	**94.77**	**95.22**	**97.80**	**88.85**	**94.33**

Table 3: Results on intent prediction tasks, including 10-shot training (Few) and full-set training (Full) settings.

Dataset	Span-ConveRT	ConVEx	GenSF	SPACE-2^{multi}
DSTC8-Bus	84.0	86.7	90.5	**91.6**
DSTC8-Event	82.2	87.2	91.2	**92.4**
DSTC8-Home	95.4	94.5	93.7	**96.2**
DSTC8-RentalCar	83.0	87.4	86.7	**88.3**
REST8K -1/8	88.5	90.6	91.8	**93.9**
REST8K -1/16	81.1	86.4	89.7	**92.6**
REST8K -1/32	63.8	81.8	82.1	**84.7**

Table 4: F1 scores across all slots for evaluation on the DSTC8 single-domain datasets in the few-shot setting.

the pre-training dialog corpora, we choose two similar models, TOD-BERT and ConvBERT+MLM, to continually pre-train on *AnPreDial* and *UnPreDial* for fairness, but they are still not comparable to SPACE-2.

6.2 More Comparison on Intent Prediction

The experimental results of three intent prediction datasets are shown in Table 3. As we can see, SPACE-2^{multi} achieves SOTA results on all datasets under all settings. Specifically, it outperforms the previous best model ConvFiT by 0.93%, 2.33% 3.53% on BANKING77 , CLINIC150 , HWU64 under the few-data training setting, respectively. It also improves ConvFiT by 0.61%, 0.46%, and 1.91% on the above three datasets under the full-shot training setting. The improvements indicate that our method has a better ability to discriminate similar intents via tree-structure enhanced contrastive learning than vanilla pre-training on ConvFiT.

6.3 More Comparison on Slot-Filling

We also conduct slot-filling experiments, particularly in the few-shot setting, to probe whether our model is also suitable for slot extraction. We follow the same data split and settings in GenSF (Mehri and Eskenazi, 2021) for a fair comparison. As shown in Table 4, SPACE-2^{multi} achieves state-of-the-art results across all experimental settings on DSTC8 and REST8K . Note that in Mehri and Eskenazi (2021), DSTC8 is evaluated in every single domain (Bus, Event, Home, RentalCar) and REST8K is evaluated with varied proportions (1/8, 1/16, 1/32) of the training set. It shows that our SPACE-2 model can achieve superior results without any specific designs only for slot-filling tasks.

6.4 Ablation Study

To figure out which factor contributes mainly to our semi-supervised pre-training paradigm, we conduct ablation studies on DialoGLUE as shown in the bottom part of each setting in Table 2. We first investigate the effect of the tree-structured supervised contrastive learning by replacing it with the traditional supervised contrastive learning (denote as 'w/o *tree*'). As we can see, vanilla supervised contrastive learning only considers self-augmented data or data with the exact same labels as positive samples, so it can not capture a more detailed semantic structure than our tree-structured method and degrade from 88.10% to 87.57% in an average score. We also try to remove the label information from *AnPreDial* and treat all pre-training data as unlabeled data (denote as 'w/o *annotation*'), or remove the unlabeled data (denote as 'w/o *UnPreDial*') to see the impact. It shows that both the labeled data and unlabeled data are crucial in our semi-supervised pre-training scheme but labeled data contributes more to the overall performance.

6.5 Visualization Analysis

In this part, we investigate why explicitly decomposing the semantic structure into sub-structures for different views can benefit dialog understanding. Figure 6 illustrates the 2D t-SNE visualization of the output unit vectors $\sigma_k(z)$ for test dialog samples from the MULTIWOZ dataset. Due to the limited space, we only show the sub-spaces of `D,I,S,V` here. As we can see, The hidden representations of SPACE-2^{multi} are able to differentiate the similar and dissimilar parts in different semantic sub-spaces. The learned latent sub-space is highly correlated with the dialog annotations of *domain, intent, slot* or *value*, which confirms our assumption.

7 Conclusion

In this work, we propose a new pre-trained conversation model named SPACE-2, which learns dialog representations from both labeled and unlabeled corpora via tree-structured semi-supervised contrastive learning (CL). To be specific, we utilize a semantic tree structure (STS) to unify the incon-

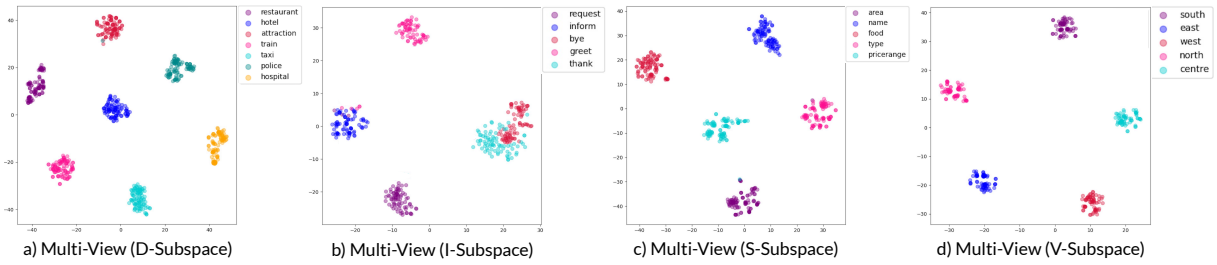

a) Multi-View (D-Subspace) b) Multi-View (I-Subspace) c) Multi-View (S-Subspace) d) Multi-View (V-Subspace)

Figure 6: T-SNE visualization of the $\sigma_k(z)$ in multi-view scoring function for MULTIWOZ test dialog samples. Four sub-spaces, including Domain-, Intent-, Slot- and Value- Subspace, are shown here.

sistent annotation schema for all datasets. Then, we propose a novel multi-view scoring function to compute the semantic similarity based on STS's sub-structures for labeled dialogs. Different from traditional CL, our method increases the relevance of representations for all similarly labeled dialogs and only pushes away completely different dialogs. A self-supervised CL loss is also integrated to exploit unlabeled dialogs. Extensive experiments on the DialoGLUE benchmark with seven datasets demonstrate that our model achieves the best results on all dialog understanding tasks, under the full-data and the few-shot settings.

Acknowledgement

This work was partially supported by National Natural Science Foundation of China (No. 61906185), Youth Innovation Promotion Association of CAS China (No. 2020357), Shenzhen Science and Technology Innovation Program (No. KQTD20190929172835662), Shenzhen Basic Research Foundation (No. JCYJ20210324115614039 and No. JCYJ20200109113441941). This work was supported by Alibaba Group through Alibaba Research Intern Program.

References

Inigo Alonso, Alberto Sabater, David Ferstl, Luis Montesano, and Ana C Murillo. 2021. Semi-supervised semantic segmentation with pixel-level contrastive learning from a class-wise memory bank. *arXiv preprint arXiv:2104.13415*.

Mahmoud Assran, Nicolas Ballas, Lluis Castrejon, and Michael Rabbat. 2020. Supervision accelerates pre-training in contrastive semi-supervised learning of visual representations. *arXiv preprint arXiv:2006.10803*.

Bill Byrne, Karthik Krishnamoorthi, Chinnadhurai Sankar, Arvind Neelakantan, Daniel Duckworth, Semih Yavuz, Ben Goodrich, Amit Dubey, Andy Cedilnik, and Kyu-Young Kim. 2019a. Taskmaster-1: Toward a realistic and diverse dialog dataset. *arXiv preprint arXiv:1909.05358*.

Bill Byrne, Karthik Krishnamoorthi, Chinnadhurai Sankar, Arvind Neelakantan, Ben Goodrich, Daniel Duckworth, Semih Yavuz, Amit Dubey, Kyu-Young Kim, and Andy Cedilnik. 2019b. Taskmaster-1: Toward a realistic and diverse dialog dataset. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 4516–4525, Hong Kong, China. Association for Computational Linguistics.

Giovanni Campagna, Agata Foryciarz, Mehrad Moradshahi, and Monica Lam. 2020. Zero-shot transfer learning with synthesized data for multi-domain dialogue state tracking. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 122–132, Online. Association for Computational Linguistics.

Iñigo Casanueva, Tadas Temčinas, Daniela Gerz, Matthew Henderson, and Ivan Vulić. 2020. Efficient intent detection with dual sentence encoders. In *Proceedings of the 2nd Workshop on Natural Language Processing for Conversational AI*, pages 38–45, Online. Association for Computational Linguistics.

Derek Chen, Howard Chen, Yi Yang, Alex Lin, and Zhou Yu. 2021. Action-based conversations dataset: A corpus for building more in-depth task-oriented dialogue systems. *arXiv preprint arXiv:2104.00783*.

Samuel Coope, Tyler Farghly, Daniela Gerz, Ivan Vulić, and Matthew Henderson. 2020. Span-ConveRT: Few-shot span extraction for dialog with pretrained conversational representations. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 107–121, Online. Association for Computational Linguistics.

Alice Coucke, Alaa Saade, Adrien Ball, Théodore Bluche, Alexandre Caulier, David Leroy, Clément Doumouro, Thibault Gisselbrecht, Francesco Caltagirone, Thibaut Lavril, et al. 2018. Snips voice platform: an embedded spoken language understanding system for private-by-design voice interfaces. *arXiv preprint arXiv:1805.10190*.

Yinpei Dai, Hangyu Li, Yongbin Li, Jian Sun, Fei Huang, Luo Si, and Xiaodan Zhu. 2021a. Preview, attend and review: Schema-aware curriculum learning for multi-domain dialogue state tracking. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 2: Short Papers)*, pages 879–885, Online. Association for Computational Linguistics.

Zhigang Dai, Bolun Cai, Yugeng Lin, and Junying Chen. 2021b. Unimoco: Unsupervised, semi-supervised and full-supervised visual representation learning. *arXiv preprint arXiv:2103.10773*.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, Minneapolis, Minnesota. Association for Computational Linguistics.

Emily Dinan, Stephen Roller, Kurt Shuster, Angela Fan, Michael Auli, and Jason Weston. 2018. Wizard of wikipedia: Knowledge-powered conversational agents. *arXiv preprint arXiv:1811.01241*.

Layla El Asri, Hannes Schulz, Shikhar Sharma, Jeremie Zumer, Justin Harris, Emery Fine, Rahul Mehrotra, and Kaheer Suleman. 2017. Frames: a corpus for adding memory to goal-oriented dialogue systems. In *Proceedings of the 18th Annual SIGdial Meeting on Discourse and Dialogue*, pages 207–219, Saarbrücken, Germany. Association for Computational Linguistics.

Mihail Eric, Rahul Goel, Shachi Paul, Abhishek Sethi, Sanchit Agarwal, Shuyang Gao, Adarsh Kumar, Anuj Goyal, Peter Ku, and Dilek Hakkani-Tur. 2020. MultiWOZ 2.1: A consolidated multi-domain dialogue dataset with state corrections and state tracking baselines. In *Proceedings of the 12th Language Resources and Evaluation Conference*, pages 422–428, Marseille, France. European Language Resources Association.

Mihail Eric, Lakshmi Krishnan, Francois Charette, and Christopher D. Manning. 2017. Key-value retrieval networks for task-oriented dialogue. In *Proceedings of the 18th Annual SIGdial Meeting on Discourse and Dialogue*, pages 37–49, Saarbrücken, Germany. Association for Computational Linguistics.

Joachim Fainberg, Ben Krause, Mihai Dobre, Marco Damonte, Emmanuel Kahembwe, Daniel Duma, Bonnie Webber, and Federico Fancellu. 2018. Talking to myself: self-dialogues as data for conversational agents. *arXiv preprint arXiv:1809.06641*.

Song Feng, Hui Wan, Chulaka Gunasekara, Siva Patel, Sachindra Joshi, and Luis Lastras. 2020. doc2dial: A goal-oriented document-grounded dialogue dataset. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 8118–8128, Online. Association for Computational Linguistics.

Tianyu Gao, Xingcheng Yao, and Danqi Chen. 2021. SimCSE: Simple contrastive learning of sentence embeddings. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 6894–6910, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Karthik Gopalakrishnan, Behnam Hedayatnia, Qinglang Chen, Anna Gottardi, Sanjeev Kwatra, Anu Venkatesh, Raefer Gabriel, Dilek Hakkani-Tür, and Amazon Alexa AI. 2019. Topical-chat: Towards knowledge-grounded open-domain conversations. In *INTERSPEECH*, pages 1891–1895.

Mansi Gupta, Nitish Kulkarni, Raghuveer Chanda, Anirudha Rayasam, and Zachary C Lipton. 2019. Amazonqa: A review-based question answering task. *arXiv preprint arXiv:1908.04364*.

Sonal Gupta, Rushin Shah, Mrinal Mohit, Anuj Kumar, and Mike Lewis. 2018. Semantic parsing for task oriented dialog using hierarchical representations. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 2787–2792, Brussels, Belgium. Association for Computational Linguistics.

Keqing He, Jinchao Zhang, Yuanmeng Yan, Weiran Xu, Cheng Niu, and Jie Zhou. 2020a. Contrastive zero-shot learning for cross-domain slot filling with adversarial attack. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 1461–1467.

Wanwei He, Yinpei Dai, Min Yang, Jian Sun, Fei Huang, Luo Si, and Yongbin Li. 2022a. Space-3: Unified dialog model pre-training for task-oriented dialog understanding and generation. In *Proceedings of the 45th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 187–200.

Wanwei He, Yinpei Dai, Yinhe Zheng, Yuchuan Wu, Zheng Cao, Dermot Liu, Peng Jiang, Min Yang, Fei Huang, Luo Si, et al. 2022b. Space: A generative pre-trained model for task-oriented dialog with semi-supervised learning and explicit policy injection. *Proceedings of the AAAI Conference on Artificial Intelligence*.

Wanwei He, Min Yang, Rui Yan, Chengming Li, Ying Shen, and Ruifeng Xu. 2020b. Amalgamating knowledge from two teachers for task-oriented dialogue system with adversarial training. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 3498–3507.

Michael Heck, Carel van Niekerk, Nurul Lubis, Christian Geishauser, Hsien-Chin Lin, Marco Moresi, and Milica Gasic. 2020. TripPy: A triple copy strategy for value independent neural dialog state tracking. In *Proceedings of the 21th Annual Meeting of the Special Interest Group on Discourse and Dialogue*, pages 35–44, 1st virtual meeting. Association for Computational Linguistics.

Charles T. Hemphill, John J. Godfrey, and George R. Doddington. 1990. The ATIS spoken language systems pilot corpus. In *Speech and Natural Language: Proceedings of a Workshop Held at Hidden Valley, Pennsylvania, June 24-27,1990*.

Matthew Henderson, Iñigo Casanueva, Nikola Mrkšić, Pei-Hao Su, Tsung-Hsien Wen, and Ivan Vulić. 2020. ConveRT: Efficient and accurate conversational representations from transformers. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 2161–2174, Online. Association for Computational Linguistics.

Matthew Henderson and Ivan Vulić. 2021. ConVEx: Data-efficient and few-shot slot labeling. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 3375–3389, Online. Association for Computational Linguistics.

Binyuan Hui, Ruiying Geng, Qiyu Ren, Binhua Li, Yongbin Li, Jian Sun, Fei Huang, Luo Si, Pengfei Zhu, and Xiaodan Zhu. 2021. Dynamic hybrid relation exploration network for cross-domain context-dependent semantic parsing. In *AAAI*.

Nakamasa Inoue and Keita Goto. 2020. Semi-supervised contrastive learning with generalized contrastive loss and its application to speaker recognition. In *2020 Asia-Pacific Signal and Information Processing Association Annual Summit and Conference (APSIPA ASC)*, pages 1641–1646. IEEE.

Mandar Joshi, Danqi Chen, Yinhan Liu, Daniel S. Weld, Luke Zettlemoyer, and Omer Levy. 2020. SpanBERT: Improving pre-training by representing and predicting spans. *Transactions of the Association for Computational Linguistics*, 8:64–77.

Endri Kacupaj, Joan Plepi, Kuldeep Singh, Harsh Thakkar, Jens Lehmann, and Maria Maleshkova. 2021. Conversational question answering over knowledge graphs with transformer and graph attention networks. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*, pages 850–862, Online. Association for Computational Linguistics.

Prannay Khosla, Piotr Teterwak, Chen Wang, Aaron Sarna, Yonglong Tian, Phillip Isola, Aaron Maschinot, Ce Liu, and Dilip Krishnan. 2020. Supervised contrastive learning. *arXiv preprint arXiv:2004.11362*.

Byoungjip Kim, Jinho Choo, Yeong-Dae Kwon, Seongho Joe, Seungjai Min, and Youngjune Gwon. 2021a. Selfmatch: Combining contrastive self-supervision and consistency for semi-supervised learning. *arXiv preprint arXiv:2101.06480*.

Seokhwan Kim, Yang Liu, Di Jin, Alexandros Papangelis, Karthik Gopalakrishnan, Behnam Hedayatnia, and Dilek Hakkani-Tur. 2021b. "how robust r u?": Evaluating task-oriented dialogue systems on spoken conversations.

Satwik Kottur, Seungwhan Moon, Alborz Geramifard, and Babak Damavandi. 2021. Simmc 2.0: A task-oriented dialog dataset for immersive multimodal conversations. *arXiv preprint arXiv:2104.08667*.

Stefan Larson, Anish Mahendran, Joseph J. Peper, Christopher Clarke, Andrew Lee, Parker Hill, Jonathan K. Kummerfeld, Kevin Leach, Michael A. Laurenzano, Lingjia Tang, and Jason Mars. 2019. An evaluation dataset for intent classification and out-of-scope prediction. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 1311–1316, Hong Kong, China. Association for Computational Linguistics.

Junnan Li, Caiming Xiong, and Steven CH Hoi. 2021. Comatch: Semi-supervised learning with contrastive graph regularization. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 9475–9484.

Shiyang Li, Semih Yavuz, Kazuma Hashimoto, Jia Li, Tong Niu, Nazneen Rajani, Xifeng Yan, Yingbo Zhou, and Caiming Xiong. 2020. Coco: Controllable counterfactuals for evaluating dialogue state trackers. *arXiv preprint arXiv:2010.12850*.

Xiujun Li, Yu Wang, Siqi Sun, Sarah Panda, Jingjing Liu, and Jianfeng Gao. 2018. Microsoft dialogue challenge: Building end-to-end task-completion dialogue systems. *arXiv preprint arXiv:1807.11125*.

Yanran Li, Hui Su, Xiaoyu Shen, Wenjie Li, Ziqiang Cao, and Shuzi Niu. 2017. Dailydialog: A manually labelled multi-turn dialogue dataset. *arXiv preprint arXiv:1710.03957*.

Zhaojiang Lin, Andrea Madotto, Genta Indra Winata, Peng Xu, Feijun Jiang, Yuxiang Hu, Chen Shi, and Pascale Fung. 2021. Bitod: A bilingual multi-domain dataset for task-oriented dialogue modeling. *arXiv preprint arXiv:2106.02787*.

Jingjing Liu, Panupong Pasupat, Scott Cyphers, and Jim Glass. 2013. Asgard: A portable architecture for multilingual dialogue systems. In *2013 IEEE International Conference on Acoustics, Speech and Signal Processing*, pages 8386–8390. IEEE.

Weijie Liu, Peng Zhou, Zhe Zhao, Zhiruo Wang, Qi Ju, Haotang Deng, and Ping Wang. 2020a. K-bert: Enabling language representation with knowledge graph.

In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pages 2901–2908.

Xingkun Liu, Arash Eshghi, Pawel Swietojanski, and Verena Rieser. 2021. Benchmarking natural language understanding services for building conversational agents. In *Increasing Naturalness and Flexibility in Spoken Dialogue Interaction*, pages 165–183. Springer.

Zihan Liu, Yan Xu, Tiezheng Yu, Wenliang Dai, Ziwei Ji, Samuel Cahyawijaya, Andrea Madotto, and Pascale Fung. 2020b. Crossner: Evaluating cross-domain named entity recognition. *arXiv preprint arXiv:2012.04373*.

Shikib Mehri and Mihail Eric. 2021. Example-driven intent prediction with observers. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 2979–2992, Online. Association for Computational Linguistics.

Shikib Mehri, Mihail Eric, and Dilek Hakkani-Tur. 2020. Dialoglue: A natural language understanding benchmark for task-oriented dialogue. *arXiv preprint arXiv:2009.13570*.

Shikib Mehri and Maxine Eskenazi. 2021. GenSF: Simultaneous adaptation of generative pre-trained models and slot filling. In *Proceedings of the 22nd Annual Meeting of the Special Interest Group on Discourse and Dialogue*, pages 489–498, Singapore and Online. Association for Computational Linguistics.

Johannes EM Mosig, Shikib Mehri, and Thomas Kober. 2020. Star: A schema-guided dialog dataset for transfer learning. *arXiv preprint arXiv:2010.11853*.

Nikola Mrkšić, Diarmuid Ó Séaghdha, Tsung-Hsien Wen, Blaise Thomson, and Steve Young. 2017. Neural belief tracker: Data-driven dialogue state tracking. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1777–1788, Vancouver, Canada. Association for Computational Linguistics.

Jongjin Park, Sukmin Yun, Jongheon Jeong, and Jinwoo Shin. 2021. Opencos: Contrastive semi-supervised learning for handling open-set unlabeled data. *arXiv preprint arXiv:2107.08943*.

Denis Peskov, Nancy Clarke, Jason Krone, Brigi Fodor, Yi Zhang, Adel Youssef, and Mona Diab. 2019. Multi-domain goal-oriented dialogues (MultiDoGO): Strategies toward curating and annotating large scale dialogue data. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 4526–4536, Hong Kong, China. Association for Computational Linguistics.

Filip Radlinski, Krisztian Balog, Bill Byrne, and Karthik Krishnamoorthi. 2019. Coached conversational preference elicitation: A case study in understanding movie preferences. In *Proceedings of the 20th Annual SIGdial Meeting on Discourse and Dialogue*, pages 353–360, Stockholm, Sweden. Association for Computational Linguistics.

Abhinav Rastogi, Xiaoxue Zang, Srinivas Sunkara, Raghav Gupta, and Pranav Khaitan. 2020. Towards scalable multi-domain conversational agents: The schema-guided dialogue dataset. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pages 8689–8696.

Siva Reddy, Danqi Chen, and Christopher D. Manning. 2019. CoQA: A conversational question answering challenge. *Transactions of the Association for Computational Linguistics*, 7:249–266.

Amrita Saha, Mitesh Khapra, and Karthik Sankaranarayanan. 2018a. Towards building large scale multimodal domain-aware conversation systems. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 32.

Amrita Saha, Vardaan Pahuja, Mitesh Khapra, Karthik Sankaranarayanan, and Sarath Chandar. 2018b. Complex sequential question answering: Towards learning to converse over linked question answer pairs with a knowledge graph. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 32.

Sebastian Schuster, Sonal Gupta, Rushin Shah, and Mike Lewis. 2019. Cross-lingual transfer learning for multilingual task oriented dialog. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 3795–3805, Minneapolis, Minnesota. Association for Computational Linguistics.

Pararth Shah, Dilek Hakkani-Tür, Bing Liu, and Gokhan Tür. 2018. Bootstrapping a neural conversational agent with dialogue self-play, crowdsourcing and on-line reinforcement learning. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 3 (Industry Papers)*, pages 41–51, New Orleans - Louisiana. Association for Computational Linguistics.

Igor Shalyminov, Alessandro Sordoni, Adam Atkinson, and Hannes Schulz. 2020. Fast domain adaptation for goal-oriented dialogue using a hybrid generative-retrieval transformer. In *ICASSP 2020-2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 8039–8043. IEEE.

Andreas Stolcke, Klaus Ries, Noah Coccaro, Elizabeth Shriberg, Rebecca Bates, Daniel Jurafsky, Paul Taylor, Rachel Martin, Carol Van Ess-Dykema, and Marie Meteer. 2000. Dialogue act modeling for automatic tagging and recognition of conversational speech. *Computational Linguistics*, 26(3):339–374.

Yu Sun, Shuohuan Wang, Yukun Li, Shikun Feng, Xuyi Chen, Han Zhang, Xin Tian, Danxiang Zhu, Hao

Tian, and Hua Wu. 2019. Ernie: Enhanced representation through knowledge integration. *arXiv preprint arXiv:1904.09223*.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. *Advances in neural information processing systems*, 30.

Ivan Vulić, Pei-Hao Su, Sam Coope, Daniela Gerz, Paweł Budzianowski, Iñigo Casanueva, Nikola Mrkšić, and Tsung-Hsien Wen. 2021. Convfit: Conversational fine-tuning of pretrained language models. *arXiv preprint arXiv:2109.10126*.

Xuewei Wang, Weiyan Shi, Richard Kim, Yoojung Oh, Sijia Yang, Jingwen Zhang, and Zhou Yu. 2019. Persuasion for good: Towards a personalized persuasive dialogue system for social good. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 5635–5649, Florence, Italy. Association for Computational Linguistics.

Wei Wei, Quoc Le, Andrew Dai, and Jia Li. 2018. AirDialogue: An environment for goal-oriented dialogue research. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 3844–3854, Brussels, Belgium. Association for Computational Linguistics.

Myers Will, Etchart Tyler, and Fulda Nancy. 2020. Conversational scaffolding: An analogybased approach to response prioritization in open-domain dialogs. In *Proceedings of the 12th International Conference on Agents and Artificial Intelligence (ICAART)*.

Jason D Williams, Antoine Raux, and Matthew Henderson. 2016. The dialog state tracking challenge series: A review. *Dialogue & Discourse*, 7(3):4–33.

Chien-Sheng Wu, Steven C.H. Hoi, Richard Socher, and Caiming Xiong. 2020. TOD-BERT: Pre-trained natural language understanding for task-oriented dialogue. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 917–929, Online. Association for Computational Linguistics.

Xing Wu, Chaochen Gao, Liangjun Zang, Jizhong Han, Zhongyuan Wang, and Songlin Hu. 2021. Esimcse: Enhanced sample building method for contrastive learning of unsupervised sentence embedding. *arXiv preprint arXiv:2109.04380*.

Alex Xiao, Christian Fuegen, and Abdelrahman Mohamed. 2021. Contrastive semi-supervised learning for asr. In *ICASSP 2021-2021 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 3870–3874. IEEE.

Yuanmeng Yan, Rumei Li, Sirui Wang, Fuzheng Zhang, Wei Wu, and Weiran Xu. 2021. Consert: A contrastive framework for self-supervised sentence representation transfer. *arXiv preprint arXiv:2105.11741*.

Steve Young. 2007. Cued standard dialogue acts. *Report, Cambridge University Engineering Department, 14th October*, 2007.

Steve Young, Milica Gašić, Blaise Thomson, and Jason D Williams. 2013. Pomdp-based statistical spoken dialog systems: A review. *Proceedings of the IEEE*, 101(5):1160–1179.

Tao Yu, Rui Zhang, Heyang Er, Suyi Li, Eric Xue, Bo Pang, Xi Victoria Lin, Yi Chern Tan, Tianze Shi, Zihan Li, Youxuan Jiang, Michihiro Yasunaga, Sungrok Shim, Tao Chen, Alexander Fabbri, Zifan Li, Luyao Chen, Yuwen Zhang, Shreya Dixit, Vincent Zhang, Caiming Xiong, Richard Socher, Walter Lasecki, and Dragomir Radev. 2019. CoSQL: A conversational text-to-SQL challenge towards cross-domain natural language interfaces to databases. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 1962–1979, Hong Kong, China. Association for Computational Linguistics.

Tao Yu, Rui Zhang, Alex Polozov, Christopher Meek, and Ahmed Hassan Awadallah. 2020. Score: Pre-training for context representation in conversational semantic parsing. In *International Conference on Learning Representations*.

Xinkai Yuan, Zilinghan Li, and Gaoang Wang. 2021. Activematch: End-to-end semi-supervised active representation learning. *arXiv preprint arXiv:2110.02521*.

Guangtao Zeng, Wenmian Yang, Zeqian Ju, Yue Yang, Sicheng Wang, Ruisi Zhang, Meng Zhou, Jiaqi Zeng, Xiangyu Dong, Ruoyu Zhang, Hongchao Fang, Penghui Zhu, Shu Chen, and Pengtao Xie. 2020. MedDialog: Large-scale medical dialogue datasets. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 9241–9250, Online. Association for Computational Linguistics.

Jianguo Zhang, Trung Bui, Seunghyun Yoon, Xiang Chen, Zhiwei Liu, Congying Xia, Quan Hung Tran, Walter Chang, and Philip Yu. 2021a. Few-shot intent detection via contrastive pre-training and fine-tuning. *arXiv preprint arXiv:2109.06349*.

Jianguo Zhang, Kazuma Hashimoto, Wenhao Liu, Chien-Sheng Wu, Yao Wan, Philip Yu, Richard Socher, and Caiming Xiong. 2020. Discriminative nearest neighbor few-shot intent detection by transferring natural language inference. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 5064–5082, Online. Association for Computational Linguistics.

Sai Zhang, Yuwei Hu, Yuchuan Wu, Jiaman Wu, Yongbin Li, Jian Sun, Caixia Yuan, and Xiaojie Wang. 2022. A slot is not built in one utterance: Spoken language dialogs with sub-slots. In *Findings of*

the Association for Computational Linguistics: ACL 2022, pages 309–321, Dublin, Ireland. Association for Computational Linguistics.

Saizheng Zhang, Emily Dinan, Jack Urbanek, Arthur Szlam, Douwe Kiela, and Jason Weston. 2018. Personalizing dialogue agents: I have a dog, do you have pets too? In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 2204–2213, Melbourne, Australia. Association for Computational Linguistics.

Zhenyu Zhang, Tao Guo, and Meng Chen. 2021b. Dialoguebert: A self-supervised learning based dialogue pre-training encoder. *arXiv preprint arXiv:2109.10480*.

Kangyan Zhou, Shrimai Prabhumoye, and Alan W Black. 2018. A dataset for document grounded conversations. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 708–713, Brussels, Belgium. Association for Computational Linguistics.

Yanning Zhou, Hang Xu, Wei Zhang, Bin Gao, and Pheng-Ann Heng. 2021. C3-semiseg: Contrastive semi-supervised segmentation via cross-set learning and dynamic class-balancing. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 7036–7045.

Pengfei Zhu, Binyuan Hui, Changqing Zhang, Dawei Du, Longyin Wen, and Qinghua Hu. 2019. Multi-view deep subspace clustering networks. *arXiv preprint arXiv:1908.01978*.

A Details for Pre-training Dialog Corpora

All dialog corpora we used in our experiments are given in Table 5 and Table 6. There are totally 32 labeled datasets and 21 unlabeled datasets. For the labeled data, each dialog sample is unified into the same data format with tree-structured labels. For the unlabeled data, each sample is performed careful cleaning due to the high noise. The processing methods include: (1) removing the instances where there is a URL in utterances. (2) removing the instances containing word repetitions of at least three words. (3) removing non-English sentences. (4) removing sentences containing special markers such as "[" or "]", as this could be markup. (5) removing offensive language. (6) replacing the non-unicode characters like emojis.

B Input Representations

The input embeddings consist of four elements: tokens, roles, turns, and positions. Role embeddings, E_u, E_s, are like segmentation embeddings in BERT and are used to differentiate which role the current token belongs to either user or system. Turn embeddings are assigned to each token according to its adverse turn number in the input dialog data. Position embeddings are assigned to each token according to its relative position within its belonging sentence. For the tokens, we use special tokens `[BOU]`, `[EOU]` to bound the user utterance, and `[BOS]`, `[EOS]` to bound the system response.

C Details for Compared Baselines

C.1 Baselines for Dialog Understanding

We choose the following baselines: 1) **BERT**, a $\text{BERT}_{\text{base}}$ model adapted to all downstream tasks; 2) **ConvBERT**, a BERT continually pre-trained on a large open-domain dialogue corpus with MLM (Mehri et al., 2020); 3) **ConvBERT-DG**, a ConvBERT continually pre-trained on all DialoGLUE data before conducting specific downstream tasks with MLM. 4) **ConvBERT+MLM**, a ConvBERT continually trained on training data of specific downstream tasks; 5) **ConvBERT-DG+MLM**, a ConvBERT+MLM pre-trained on all DialoGLUE data before conducting downstream tasks; and 6) **TOD-BERT**, a BERT pre-trained on nine existing task-oriented dialogue corpora using MLM and response selection loss (Wu et al., 2020).

C.2 Baselines for Specific Tasks

In this paper, we focus on intent prediction and slot-filling in particular since there are adequate pre-trained baselines specifically. For the intent prediction task, we choose: 1) **Example+Observer**, a ConvBERT+MLM that uses example-driven training based on similarity matching and observers for transformer attentions (Mehri and Eric, 2021); 2) **USE+ConveRT**, a dual sentence encoder pre-trained on 654M Reddit dialogs (Casanueva et al., 2020); 3) **DNNC**, a discriminative nearest-neighbor model which finds the best-matched sample via similarity matching (Zhang et al., 2020); 4) **CPFT**, which uses self-supervised contrastive learning to perform pre-training and fine-tuning (Zhang et al., 2021a); and 5) **ConvFit**, which uses regular supervised contrastive learning on labeled data after pre-training on chatting data (Vulić et al., 2021). For the slot-filling task, we choose: 1) **Span-ConveRT**, a slot-filling model pre-trained on collected data (Coope et al., 2020); 2) **ConVEx**, a span-ConveRT pre-trained with the pairwise cloze task on Reddit for few-shot slot labeling (Henderson and Vulić, 2021); and 3) **GenSF**, a generative pre-trained slot-filling model (Mehri and Eskenazi, 2021).

Dataset	# Dialog	# Utterance
BANKING77 (Casanueva et al., 2020)	13,083	13,083
CLINIC150 (Larson et al., 2019)	23,700	23,700
HWU64 (Liu et al., 2021)	25,716	25,716
REST8K (Coope et al., 2020)	11,115	11,975
TOP(Gupta et al., 2018)	44,783	44,783
ATIS(Hemphill et al., 1990)	5,817	5,817
SNIPS(Coucke et al., 2018)	14,484	14,484
CrossNER(Liu et al., 2020b)	27,458	27,458
FB_TOD_SF(Schuster et al., 2019)	43,323	43,323
MIT-restaurant(Liu et al., 2013)	9,181	9,181
MIT-movies-eng(Liu et al., 2013)	12,218	12,218
MIT-movies-trival10k13(Liu et al., 2013)	9,769	9,769
MULTIWOZ _coco(Li et al., 2020)	32,062	64,124
MULTIWOZ (Eric et al., 2020)	10,433	142,968
STAR(Mosig et al., 2020)	5,820	98,962
DailyDialog(Li et al., 2017)	13,118	102,979
SGD(Rastogi et al., 2020)	22,825	463,284
Frames(El Asri et al., 2017)	1,369	19,986
MSRe2e(Li et al., 2018)	10,087	74,686
DSTC2(Williams et al., 2016)	4,953	73,228
DSTC3(Williams et al., 2016)	1,980	30,456
SimJoint(Shah et al., 2018)	3,008	27,120
MulDoGo(Peskov et al., 2019)	11,668	194,097
WOZ(Mrkšić et al., 2017)	1,200	8,824
TaskMaster1(Byrne et al., 2019b)	13,210	279,287
TaskMaster2(Byrne et al., 2019b)	17,289	292,830
TaskMaster3(Byrne et al., 2019b)	23,757	477,026
InCar(Eric et al., 2017)	3,031	15,928
MULTIWOZ _synthesis(Campagna et al., 2020)	37,605	401,075
SwDA(Stolcke et al., 2000)	1,434	274,786
BiTOD(Lin et al., 2021)	3,689	72,462
PersuaGOOD(Wang et al., 2019)	300	10,864
Total	459,465	3,366,479

Table 5: Statistics for each labeled dataset in *AnPreDial*.

Name	# Dialog	# Utterance
MulDoGo_un(Peskov et al., 2019)	63,404	1,013,985
ABCD(Chen et al., 2021)	10,042	143,855
AirDialog(Wei et al., 2018)	361,822	4,645,475
CCPE(Radlinski et al., 2019)	502	11,971
MetalWOZ(Shalyminov et al., 2020)	40,203	458,237
CMU_DoG(Zhou et al., 2018)	4,221	134,197
ConvQuestions(Kacupaj et al., 2021)	11,200	112,000
CoQA(Reddy et al., 2019)	7,699	233,260
CoSQL(Yu et al., 2019)	2,458	25,433
doc2dial(Feng et al., 2020)	688	52,688
DSTC10-track2(Kim et al., 2021b)	2,919	55,377
DSTC10-track3(Kottur et al., 2021)	11,244	110,767
MedicalDialog(Zeng et al., 2020)	482	981
Self-Dialog(Fainberg et al., 2018)	24,165	372,720
WOW(Dinan et al., 2018)	22,311	201,999
TopicChat(Gopalakrishnan et al., 2019)	8,628	188,378
Persona-Chat(Zhang et al., 2018)	11,087	81,032
MMD(Saha et al., 2018a)	1,506,129	5,477,066
CSQA(Saha et al., 2018b)	197,001	4,153,092
AmazonQA(Gupta et al., 2019)	923,685	1,847,370
ChitChat(Will et al., 2020)	7,168	258,145
Total	3,217,058	19,578,028

Table 6: Statistics for each unlabeled dataset in *UnPreDial*.

ET5: A Novel End-to-end Framework for Conversational Machine Reading Comprehension

Xiao Zhang[123], **Heyan Huang**[123]*, **Zewen Chi**[123], **Xian-Ling Mao**[123]
[1]School of Computer Science and Technology, Beijing Institute of Technology
[2]Beijing Engineering Research Center of High Volume Language Information Processing and Cloud Computing Applications
[3]Southeast Academy of Information Technology, Beijing Institute of Technology
{yotta,hhy63,czw,maoxl}@bit.edu.cn

Abstract

Conversational machine reading comprehension (CMRC) aims to assist computers to understand an natural language text and thereafter engage in a multi-turn conversation to answer questions related to the text. Existing methods typically require three steps: (1) decision making based on entailment reasoning; (2) span extraction if required by the above decision; (3) question rephrasing based on the extracted span. However, for nearly all these methods, the span extraction and question rephrasing steps cannot fully exploit the fine-grained entailment reasoning information in decision making step because of their relative independence, which will further enlarge the information gap between decision making and question phrasing. Thus, to tackle this problem, we propose a novel end-to-end framework for conversational machine reading comprehension based on shared parameter mechanism, called entailment reasoning T5 (ET5). Despite the lightweight of our proposed framework, experimental results show that the proposed ET5 achieves new state-of-the-art results on the ShARC leaderboard with the BLEU-4 score of 55.2. Our model and code are publicly available[1].

Rule Document: Taking more leave than the entitlement. If a worker has taken more leave than they're entitled to, their employer must not take money from their final pay unless it's been agreed beforehand in writing. The rules in this situation should be outlined in the employment contract, company handbook or intranet site.

User Question: Can my employer take money from my final pay?
User Scenario: I just looked at my paperwork and I think my boss took out too much of my money.
Follow-up Q: Did you take more leave than they're entitled to?
Follow-up A: Yes

Decision Making: Yes | No | **Inquire** | Irrelevant
Final Answer: Did you agree to it beforehand in writing?

Figure 1: An example in the CMRC dataset. Machine should first make the decision of `Yes/No/Inquire/Irrelevant`, and then generate the follow-up question if the decision is `Inquire`. The colored sentences show the reasoning process for the final answer.

1 Introduction

Conversational machine reading comprehension (CMRC) (Saeidi et al., 2018) aims to assist machines to understand an natural language text and thereafter engage in a multi-turn conversation to answer questions related to the text. Specifically, the machine needs to reason for decision making and question generation by interacting through rule document, user question, user scenario, and dialogue history. As an example shown in Figure 1, after fully interacting with complicated context information, the machine makes a decision of `Yes/No/Inquire/Irrelevant`, and then generates a question under the `Inquire` decision.

Existing researches (Saeidi et al., 2018; Verma et al., 2020; Lawrence et al., 2019; Zhong and Zettlemoyer, 2019; Gao et al., 2020a,b; Ouyang et al., 2021) mainly aim to capture the interactions among the complicated inputs, and achieve promising results by conducting various fine-

*Corresponding author.
[1]https://github.com/Yottaxx/ET5

Proceedings of the 29th International Conference on Computational Linguistics, pages 570–579
October 12–17, 2022.

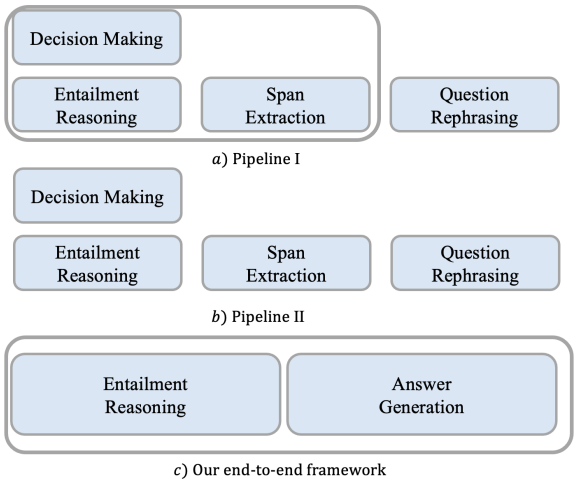

Figure 2: The overview of frameworks in CMRC. (a) For Pipeline I, decision making and span extraction models share the encoder but suffer from the problem of noisy span extraction. (b) For Pipeline II, the three stages are handled completely separately, and there is no information sharing among the three stages. (c) Our framework is an end-to-end framework with a shared encoder and a duplex decoder. The duplex decoder contains an entailment reasoning decoder and answer generation decoder, both the information of entailment reasoning and answer generation are shared through the common encoder. Both decisions and follow-up questions will be generated via answer generation decoder directly.

grained entailment reasoning interaction strategies based on Pre-trained Language Models (PrLMs) (Devlin et al., 2019; Liu et al., 2020; Dong et al., 2019; Clark et al., 2020; Raffel et al., 2020). These methods (Zhong and Zettlemoyer, 2019; Gao et al., 2020a,b; Ouyang et al., 2021) typically adopt pipeline architectures, which are shown in Figure 2. These pipeline architectures typically require three steps : (1) decision making based on entailment reasoning; (2) span extraction if required by the above decision; (3) question rephrasing based on the extracted span. There are currently two types of pipeline structures: Pipeline I and Pipeline II. The Pipeline I make decisions and extract spans simultaneously, while the Pipeline II handles all three stages separately.

However, for nearly all these methods (Zhong and Zettlemoyer, 2019; Gao et al., 2020a,b; Ouyang et al., 2021), the span extraction and question rephrasing steps can't fully exploit the fine-grained entailment reasoning information in decision making step. For Pipeline II, these methods (Gao et al., 2020b; Ouyang et al., 2021) do not share entailment reasoning information among decision-making, span extraction, and question phrasing at all. For Pipeline I, these methods (Zhong and Zettlemoyer, 2019; Gao et al., 2020a) only approximate share the information through noisy span extraction. Both of them enlarge the information gap between decision making and question rephrasing, and seriously affect the performance of question generation.

To tackle this problem, we propose a novel end-to-end framework for conversational machine reading comprehension based on shared parameter mechanism, called entailment reasoning T5 (ET5). Specifically, the proposed framework consists of a text-to-text Transformer and an additional entailment reasoning decoder. The original decoder in the text-to-text Transformer will directly generate either decision or follow-up question based on the shared encoder enhanced by entailment reasoning. The entailment reasoning decoder can be configured with different entailment reasoning strategies. Despite the lightweight of our proposed framework, experimental results show that ET5 achieves new state-of-the-art results on the ShARC leaderboard with the BLEU-4 score of 55.2 and significantly improves the generalization performance of question generation.

Our contributions are summarized as follows:

- We propose a novel end-to-end framework, called ET5, to better capture the entailment information for question generation, and thus eliminate the information gap between decision making and question generation.
- Extensive experiments demonstrate the effectiveness of the proposed framework on ShARC benchmark, especially in the question generation sub-task.

2 Related Work

Conversation-based reading comprehension (Saeidi et al., 2018; Sun et al., 2019; Reddy et al., 2019; Choi et al., 2018; Cui et al., 2020; Gao et al., 2021) extends the context with dialogue history, which is formed to simulate the communication scene in real life. Most of them are ideal subtasks, either span-based QA tasks (Choi et al., 2018; Reddy et al., 2019) or multi-choice tasks (Sun et al., 2019; Cui et al., 2020). We focus on the task (Saeidi et al., 2018) that deal with real-world complexities, where the machine needs to make decisions or ask questions to keep the conversa-

tion going. This task (Saeidi et al., 2018) is called Conversational Machine Reading Comprehension (CMRC), which requires the machine to have the inference ability to capture the interactions among rule document, user question, user scenario, and dialogue history.

Recent studies (Zhong and Zettlemoyer, 2019; Gao et al., 2020a,b; Ouyang et al., 2021) in CMRC are generally utilized to match the relationship between the various information. E^3 (Zhong and Zettlemoyer, 2019) first investigates the importance of clarifying the different rule units for entailment reasoning. Different entailment reasoning strategies (Gao et al., 2020a,b, 2021) with fine-grained reasoning units are further proposed to improve the abilities of entailment reasoning. In addition, discourse relationships between fine-grained reasoning units are utilized to model the discourse graph (Ouyang et al., 2021; Zhang et al., 2021). These methods typically adopt pipeline architectures, DISCERN (Gao et al., 2020b) first discovers the unbalance and noisy problems of Pipeline I conducted by E^3 (Zhong and Zettlemoyer, 2019) and EMT (Gao et al., 2020a), then solves them by utilizing Pipeline II to process the three stages separately. However, due to the pipeline's inability to make full use of the entailment information, both of the above pipeline structures have the problem of information gap (Zhang et al., 2021) between decision making and question generation.

To better capture the entailment information for question generation and eliminate the information gap, we propose a novel end-to-end framework for conversational machine reading comprehension based on shared parameter mechanism, called ET5, which will be introduced in the next section.

3 Method

3.1 Settings of ET5

Each example of CMRC is formed as the tuple $\{C, R, A, S\}$. C donates the context, which is a concentrated sentence of rule document, user scenario, user question, and dialogue history. Especially, $C = \{e_1, e_2, ..., e_k, s, q, d_1, d_2, ..., d_n\}$, where e donates the elementary discourse unit (EDU) segmented from by rule documents. s and q are user scenario and user question, d represents the dialogues. Each item of C is prefixed with a special token to represents the following sentence, the details of the prefix are written in Section 3.2. R represents the discourse relations among EDUs, the parsed details are reported in Section 4.1. A is the final answer, including the decision or follow-up question. S donates the entailment reasoning state of each EDU in `ENTAILMENT`, `CONTRADICTION`, or `NEUTRAL`. To get the noisy supervision signals of entailment states, we adopt a heuristic approach[2] following the previous study (Gao et al., 2020a). Given inputs C, R, ET5 needs reasoning entailment states S and final answer A including the decision and follow-up question. As illustrated in Figure 3, we conduct duplex decoder to process answer generation and entailment reasoning simultaneously in a multi-task training approach with the shared encoder. The training procedure and evaluating procedure are illustrated in Algorithm 1 and Algorithm 2, respectively.

Algorithm 1 Training procedure of ET5

Input: Concentrated context C, discourse relations R, learning rate τ, discourse relations R

Output: Final answer A, entailment reasoning state S, ET5 encoder parameters $\boldsymbol{\theta}_e$, ET5 answer generation decoder parameters $\boldsymbol{\theta}_a$, ET5 entailment reasoning decoder parameters $\boldsymbol{\theta}_d$

Initialize $\boldsymbol{\theta}_e$,$\boldsymbol{\theta}_a$,$\boldsymbol{\theta}_d$
while not converged **do**
  for $i = 1, 2, \ldots, N$ **do**
    $\boldsymbol{e}_i = f(c_i, \boldsymbol{\theta}_e)$ s.t. $\forall c \in \mathcal{C}$
    $\boldsymbol{s}_i = f(e_i, r_i, \boldsymbol{\theta}_d)$ s.t. $\forall r \in \mathcal{R}$
    $\boldsymbol{a}_i = f(e_i, \boldsymbol{\theta}_a)$
  end for
  $\boldsymbol{g} \leftarrow \nabla_{\boldsymbol{\theta}} \mathcal{L}$
  $\boldsymbol{\theta}_e \leftarrow \boldsymbol{\theta}_e - \tau \boldsymbol{g}$
  $\boldsymbol{\theta}_d \leftarrow \boldsymbol{\theta}_d - \tau \boldsymbol{g}$
  $\boldsymbol{\theta}_a \leftarrow \boldsymbol{\theta}_a - \tau \boldsymbol{g}$
end while

3.2 Encoding

Fine-grained Prefix Prompt We investigate and propose a fine-grained prefix strategy, to prompt the interactions among different components of the input. As shown in Figure 3, the concatenate input is prefixed with a text-form task prefix. Furthermore, given relationship tagged EDUs, user question, user scenario, dialogue history as inputs, each of them is prefixed with a fine-grained

[2]The noisy supervision signal is a heuristic label obtained by the minimum edit distance.

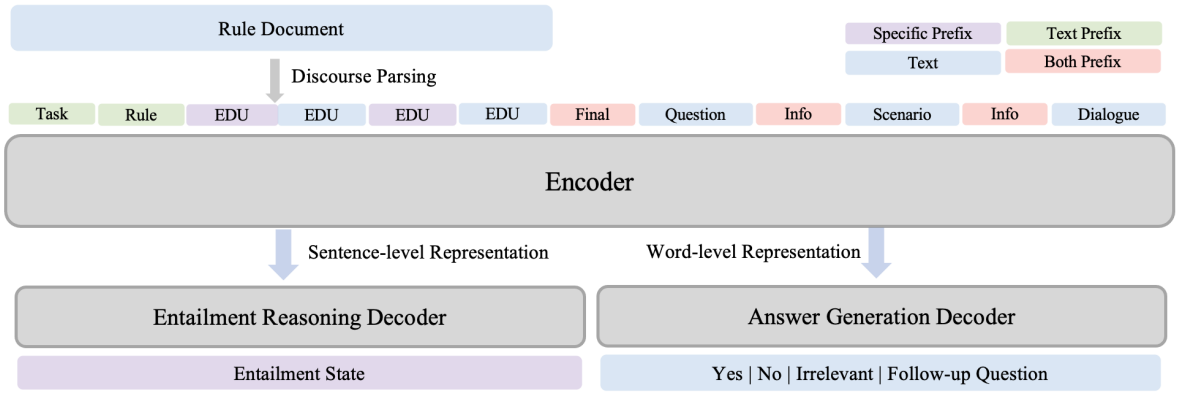

Figure 3: The architecture of ET5. Our proposed framework is an end-to-end framework based on a single text-to-text Transformer. The decoder of our proposed framework is a duplex decoder, including entailment reasoning decoder and answer generation decoder. The answer generation decoder will generate the final answer directly, either of the decision or the follow-up question. The entailment reasoning decoder is utilized to reason the fine-grained entailment states, which is only activated in the training stage. Red boxes indicate the fine-grained prefixes, including special prefixes and text prefixes, which are represented by purple boxes and green boxes, respectively. Special prefixes refer to special tokens that aim to get the sentence-level representations. Text prefixes refer to the component-specific text prefixes that aim to differentiate among different input types.

Algorithm 2 Evaluating procedure of ET5

Input: Concentrated context C, ET5 encoder parameters $\boldsymbol{\theta}_e$, ET5 answer generation decoder parameters $\boldsymbol{\theta}_a$

Output: Final answer A

1: Initialize $\boldsymbol{\theta}_e$,$\boldsymbol{\theta}_a$
2: **for** $i = 1, 2, \ldots, N$ **do**
3: $\quad \boldsymbol{e}_i = f(c_i, \boldsymbol{\theta}_e)$ s.t. $\forall c \in \mathcal{C}$
4: $\quad \boldsymbol{a}_i = f(e_i, \boldsymbol{\theta}_a)$
5: **end while**

prefix.The fine-grained prefix consists of a text prefix and a special prefix. The text prefix is used to differentiate between different types of input as an addition information prefix. The special prefix is used to obtain sentence-level representations required for entailment reasoning. In the case of user scenario and each dialogue history usually play a similar role as an information provider in CMRC tasks, user scenario and each dialogue history share the same text prefix. Meanwhile, each EDU has a special token [EDU]. Both user question and final answer are prefixed with the same text FINAL. We concatenate the fine-grained prefixed EDUs, user question, user scenario, dialogue history to get the encoding representations.

Fine-grained Prefix Encoding We concatenate the fine-grained prefixed EDUs, user question, user scenario, dialogue history as the input. Encoder representation H_e is encoded with the input by conducting T5 encoder (Raffel et al., 2020) as the encoder. Let $H_s = [h_{e_1}, h_{e_2}, ..., h_{e_k}, h_{f_i}, h_{s_i}, h_{d_1}, ..., h_{d_n}]$, H_s to donate the sentence-level representations. h_e, h_f, h_s, h_d represent the fine-grained special prefix token representation of EDU, user question, user scenario, and dialogue history, respectively.

3.3 Decoding

Our decoding is duplex decoding, including entailment reasoning and answer generation. Especially, both answer generation and entailment reasoning are activated in the training stage. During the inference stage, the answer generation decoder will directly generate either the decision or the follow-up question while the entailment reasoning will be dropped. We conduct entailment reasoning decoder with various entailment reasoning strategies in the experiments, including inter attention reasoning (Gao et al., 2020b) and dialogue graph modeling (Ouyang et al., 2021). We mainly introduce dialogue graph reasoning here, because dialogue graph modeling only has one more graph reasoning block than inter attention reasoning, the other structures are the same.

Entailment Reasoning We utilize dialogue graph modeling for entailment reasoning decoding. Dialogue graph consists of the explicit discourse graph, the implicit discourse graph, and the

inter attention reasoning. The details are shown in the following.

Given H_s and R, we construct the explicit discourse graph G to explicitly model the complex logical structures between the various information in CMRC by introducing discourse relationships among the rule conditions. Following previous (Ouyang et al., 2021), the graph is formed as a Levi graph (Levi, 1942).

There are three types of vertices in the graph: EDUs, discourse relationships, and user scenarios. Each EDU duplex connects with the tagged relationship. The user scenario connects all the other vertices as a global vertex. All the types R_L of the possible edges between vertices are six, each of them is named as *default-in*, *default-out*, *reverse-in*,*reverse-out*, *self*, and *global*. The EDUs vertices and user scenario vertex are initialized with the contextualized representation in H_s. And the discourse relationships vertices are initialized with a conventional embedding layer. Then the representation h_p of each node v_p is initialized. To handle the multi-relation graphs and dynamically weight the different relations, we use a relational graph convolution network (Schlichtkrull et al., 2018) with a gating mechanism. the graph-based information processing can be written as:

$$g_p^{(l)} = \text{Sigmoid}(h_p^{(l)} w_{r,g}^{l}), \tag{1}$$

$$h_p^{(l+1)} = \text{ReLU}(\sum_{r \in R_L} \sum_{v_p \in \mathcal{N}_r(v_p)} g_p^{(l)} \frac{1}{c_{p,r}} w_r^{(l)} h_q^{(l)}), \tag{2}$$

where $w_r^{(l)}$ is the trainable parameters of layer l. $w_{r,g}^{(l)}$ is trainable parameters under relation type r of layer l. $c_{p,r}$ is the number of the neighbors of node v_p with relationship r. $\mathcal{N}_r(v_p)$ refers to those neighbors. Let $H_p = [h_{p_1}^{(l+1)}), h_{p_2}^{(l+1)}, ..., h_{f_i}, h_{s_i}, h_{d_1}, ..., h_{d_n}]$, l is the last layer, H_p donate the explicit discourse graph representation.

Given the EDUs tokens hidden representation E from H_e. We decouple and fuse the local information and the contextualized information by conducting the implicit discourse graph. Considering each token i of EDU as a vertex in the graph, the adjacent matrices can express the implicit discourse graph. We use I_i donate the index of token i in EDU, the information decoupling adjacent matrices M can be written as:

$$M_l[i,j] = \begin{cases} 0, & I_i = I_j \\ -\infty, & otherwise \end{cases} \tag{3}$$

$$M_c[i,j] = \begin{cases} 0, & I_i \neq I_j \\ -\infty, & otherwise \end{cases}, \tag{4}$$

where M_l and M_c are conducted to express the local and contextualized information. We use multi-head-self-attention (MHSA) (Vaswani et al., 2017) to process decoupling:

$$G_i = \text{MHSA}(E, M_i), \quad i \in \{l, c\}, \tag{5}$$

after exploring the potential textual relations in the rule document, we apply a fusion layer to fuse the information by considering the encoder encoding and the attention hidden states of EDUs:

$$\tilde{E}_1 = \text{ReLU}(f([E, G_l, E - G_l, E \odot G_l])), \tag{6}$$

$$\tilde{E}_2 = \text{ReLU}(f([E, G_c, E - G_l, E \odot G_c])), \tag{7}$$

$$g = \text{Sigmoid}(f([\tilde{E}_1, \tilde{E}_2])), \tag{8}$$

$$C = g \odot G_l + (1 - g) \odot G_c, \tag{9}$$

where f is the fully-connected layer. Let $H_i = [h_{c_1}, h_{c_2}, ..., h_{f_i}, h_{s_i}, h_{d_1}, ..., h_{d_n}]$, H_i donate the explicit discourse graph representation. h_{c_i} is updated by the representation of [EDU] in C.

Given the sentence-level representation H_e, H_p, H_i, inter attention reasoning aims to fully interact with various information, including EDUs, user question, user scenario, dialogue history. We utilize an inter-sentence Transformer (Vaswani et al., 2017) to reason the entailment states. Let $\tilde{H}_e$, $\tilde{H}_p$, $\tilde{H}_i$ donate the inter-sentence Transformer encoding representation, $\tilde{H}_s$ donate the average encoding, namely, $\tilde{H}_s = [\tilde{h}_{e_1}, \tilde{h}_{e_2}, ..., \tilde{h}_{e_k}, \tilde{h}_{f_i}, \tilde{h}_{s_i}, \tilde{h}_{d_1}, ..., \tilde{h}_{d_n}]$. All the vectored representations are in the same dimension. Following previous studies (Gao et al., 2021), we utilize a linear transformation to track the entailment reasoning state of each EDU:

$$c_i = W_c \tilde{h}_{e_i} + b_c \in \mathcal{R}^3, \tag{10}$$

where the W_c is trainable parameters, c_i is the predicted score for the three labels of the i-th states.

Answer Generation Answer generation is utilized to generate either the decision or the follow-up question. We employ T5 decoder as our answer generation decoder. Given encoder hidden

representation H_e, and the set of final answer $(a_1, a_2, ..., a_n)$, including decision or follow-up question, each of the answers is composed of the variable-length tokens $(x_1, x_2, ..., x_m)$, the probabilities over the tokens are shown in the blow:

$$p(a) = \prod_{1}^{m} p(x_i|x_{<i}, H_e; \theta), \quad (11)$$

where θ donates the trainable parameters of our decoder.

3.4 Training Objective

Entailment Reasoning Given the entailment fulfillment states c_i, the entailment reasoning is supervised by cross-entropy loss:

$$\mathcal{L}_{enatil} = -\frac{1}{N}\sum_{i=1}^{N} log\,\mathrm{softmax}(c_i)_r, \quad (12)$$

where r is the ground truth of entailment state.

Answer Generation Given the encoder representation H_e, the answer generation training objective is computed by:

$$\mathcal{L}_{answer} = -\sum_{i=1}^{M} log\, p(x_i|x_{<i}, H_e; \theta), \quad (13)$$

The overall loss function is:

$$\mathcal{L} = \mathcal{L}_{answer} + \lambda\mathcal{L}_{entail}. \quad (14)$$

4 Experiment and Analysis

4.1 Data

Dataset The experimental dataset is ShARC, the current CMRC benchmark, which is built up from 948 rule text. The corpus is clawed from the government website. The utterances size of ShARC is 32,436, each of the utterances related to a dialog tree, the utterances with the same rule text refer to the same dialog tree. Each dialog tree contains all possible fulfillment combinations of conditions. The train, dev, test size is 21,890, 2,270, 8,276, respectively. Each item consists of utterance id, tree id, rule document, initial question, user scenario, dialog history, evidence, and the decision. Evidence is only used to support the answer, and can't be treated as input.

Preprocess Following previous methods (Ouyang et al., 2021; Gao et al., 2020b), we first split rule documents into elementary discourse units (EDUs), and then tag the discourse relationship among EDUs. For discourse segmentation, the rule documents are split into EDUs by using a pre-trained discourse parser (Li et al., 2018). For discourse relation extraction, we utilize a pre-trained discourse relation parser[3] to tag the structural relations among EDUs.

4.2 Setup

Evaluation Evaluation in ShARC is divided into two parts. First is decision classification: Micro-Acc and Macro-Acc scores are used for the evaluation in classification. Then question generation part is evaluated with BLEU (Papineni et al., 2002) score only if the prediction and ground truth in classification are both inquired.

Implementation Details We implement ET5 by configuring entailment reasoning decoder with two different methods: inter attention reasoning (Gao et al., 2020b) and dialogue graph reasoning (Ouyang et al., 2021), named ET5-Discern and ET5 respectively. The parameters of entailment reasoning decoder are randomly initialized, the remain parameters are initialized with official T5 (Raffel et al., 2020). ET5 and ET5-Discern are fine-tuned with AdamW (Loshchilov and Hutter, 2018) in 16 epochs, and the batch sizes are 32 and 16 respectively. We use hierarchical learning rates, the learning rate of T5 is 2e-4, the learning rate of other parameters are 2e-5. We've tried 1.0, 1.5, 2.0, 3.0 for λ, and find 1.0 is the best base on the results in the dev set. During inference decoding, the beam search number is set to 5. All results are conducted in two 3090 GPU (24GB memory)

4.3 Results

All results in the blind held-out test set of the ShARC benchmark are illustrated in Table 1. There are two different implementations here. ET5-Discern is configured with a DISCERN-formed entailment reasoning decoder by using the base-size model as the backbone. ET5 is configured with a DGM-formed entailment reasoning decoder by using the large-size model as the backbone.

[3] https://github.com/shizhouxing/DialogueDiscourseParsing

Models	Micro	Macro	BLEU-1	BLEU-4
Seq2Seq (Saeidi et al., 2018)	44.8	42.8	34.0	7.8
Pipeline (Saeidi et al., 2018)	61.9	68.9	54.4	34.4
BERTQA (Zhong and Zettlemoyer, 2019)	63.6	70.8	46.2	36.3
UrcaNet (Verma et al., 2020)	65.1	71.2	60.5	46.1
BiSon (Lawrence et al., 2019)	66.9	71.6	58.8	44.3
E^3 (Zhong and Zettlemoyer, 2019)	67.6	73.3	54.1	38.7
EMT (Gao et al., 2020a)	69.1	74.6	63.9	49.5
DISCERN (Gao et al., 2020b)	73.2	78.3	64.0	49.1
DGM (Ouyang et al., 2021)	**77.4**	**81.2**	63.3	48.4
ET5-Discern (ours)	**74.4**	**78.7**	**66.4**	**51.6**
ET5 (ours)	76.3	80.5	**69.6**	**55.2**

Table 1: Performance on the blind held-out test set of ShARC benchmark.

Models	Micro	Macro	BLEU-1	BLEU-4	Params
Discern	74.9	**79.8**	65.7	52.4	330M
ET5-Discern	**75.4**	79.7	65.2	51.1	220M
DGM	**78.6**	82.2	**71.8**	**60.2**	1020M
ET5	**78.6**	**82.5**	65.3	53.3	770M

Table 2: Performance on the dev set of the ShARC benchmark. Params are the parameter numbers of PrLMs used in the framework.

Models	Dev Set		Test Set	
	BLEU-1	BLEU-4	BLEU-1	BLEU-4
E^3	67.1	53.7	54.1(-13.0)	38.7(-15.0)
EMT	67.5	53.2	63.9(-3.6)	49.5(-3.7)
DISCERN	65.7	52.4	64.0(-1.7)	49.1(-3.3)
DGM	71.8	60.2	63.3(-8.5)	48.4(-11.8)
ET5-Discern	65.2	51.1	**66.4(+1.2)**	**51.6(+0.5)**
ET5	65.3	53.3	**69.6(+4.3)**	**55.2(+1.9)**

Table 3: Performance of BLEU scores on the dev set and test set of the ShARC benchmark.

Experimental results demonstrate that the proposed framework achieves new SOTA with considerable improvement in terms of BLEU scores. ET5-Discern outperforms DISCERN by 2.4 in BLEU-1, 2.5 in BLEU-4, 1.2 in micro-averaged accuracy, and 0.4 in macro-averaged accuracy. ET5 outperforms DGM by 6.3 in BLEU-1, 6.8 in BLEU-4. We further analyze the results in the dev set shown in Table 2. Compared to the existing pipeline framework, our framework reduces the number of parameters by 32.5% and 24.5% for the base-size model and large-size model, respectively.

Particularly, the BLEU scores of our ET5 framework outperform DISCERN and DGM with a considerable improvement in the test set. Compared with the previous SOTA, the results have increased by 5.6 and 5.7 respectively in BLEU-1 and BLEU-4. Moreover, as shown in Table 3, the existing pipeline frameworks have a certain degree of decline on the test set with BLEU scores, which indicates the drawback of the existing pipeline architectures. In the contract, the BLEU scores of ET5 and ET5-Discern continue to improve on the test set, which demonstrates the better generalization of our framework ET5 in question generation. The above results prove that our proposed framework takes better advantage of the fine-grained entailment reasoning information and eliminate the information gap between decision making and question generation.

Additionally, in the decision making evaluation, we achieve the best performance in the dev set, but there is a slight drop in the test set. However, a good classification result must be an inference based on an existing fact. Intuitively, the correctness of reasoning can be analyzed by the performance of the question generation. Correct reasoning will make the model ask the right questions. Correct classification, but asking the wrong question, does not mean that the model has learned the reasoning ability correctly, and the phenomena such as statistical bias may also cause this problem.

4.4 Ablation Studies

The existing generation question evaluation metrics suffer from randomness[4] on the small dev set

[4]The generated questions are evaluated with BLEU scores only if the prediction and ground truth in classification are

Models	Micro	Macro	ABLEU-1	ABLEU-4
ET5-Base	**75.9**	**80.4**	**54.7**	**43.6**
ET5-Base-wo/g	75.4	79.7	45.0	36.4
ET5-Base-wo/g+f	73.4	78.0	49.4	40.3
ET5-Base-wo/e+f	72.9	77.3	42.1	35.0

Table 4: Ablation study of our base-size model on the dev set of ShARC.

(2,270). To better evaluate the question generation abilities of models on the dev set, we utilized ALL-BLEU (ABLEU) to evaluate all the examples that ground truth is inquired to generate a question in ablation studies. All the other settings remain the same with official evaluation.

The ablation studies of ET5 on the dev set on ShARC benchmark are shown in Table 4. We use the base-size model to investigate the impacts of different components, there are three ablations of our ET5-Base is considered:

- **ET5-Base-wo/g** trains the model without graph reasoning block, the setting is the same as ET5-Discern.
- **ET5-Base-wo/g+f** trains the model without graph reasoning block and fine-grained prefix.
- **ET5-Base-wo/e+f** trains the model without entailment reasoning decoder and fine-grained prefix, which can be considered as the original T5 model.

4.4.1 Analysis of Graph Reasoning

Graph Reasoning consists of explicit discourse graph reasoning and implicit discourse graph reasoning, each of them introducing discourse relations among EDUs and decoupling-fusion mechanism into ET5, respectively. This setting is the same as ET5-Discern. Both accuracy scores and ABLEU scores are improved by introducing graph reasoning. In addition, we observe a significant reduction in the ABLEU scores if removing graph reasoning. ABLEU is used to measure whether the model answers due to the correct reasoning of the missing knowledge, the results show ET5-Base correctly reasoned out the missing knowledge, which suggests the necessity of graph reasoning block.

both 'inquire'.

4.4.2 Analysis of Fine-grained Text Prefix

We investigate the necessity of the fine-grained text prefix by additional removing the fine-grained text prefix in ET5-Discern, while it's hard to reason for the entailment of EDUs without the fine-grained special prefix. We feed fine-grained special tokens prefixed text into ET5 directly. As shown in the results, compared with ET5-Base-wo/g and ET5-Base-wo/g+f, the accuracy will be significantly improved by introducing the fine-grained text prefix, which indicates that directly using special token prefixes will cause noise disturbance for semantic learning. As illustrated in 4, the ABLEU-1 is decreased by 4.4, and the ABLEU-4 is decreased by 3.9. The above results show the importance of the fine-grained text prefix.

4.4.3 Analysis of Entailment Reasoning

ET5-Base-wo/e+f can be considered as the official T5 model. As shown in Table 4, the lack of fine-grained entailment reasoning information will seriously affect the performance of decision making and question generation. Compared with the performance of ET5-Base, the ABLEU-1 and ABLEU-4 of ET5-Base-wo/e+f decreased by 7.3 and 5.3 after removing entailment reasoning decoder, which indicates the importance of entailment reasoning, especially for reasoning of question generation.

5 Conclusion

In this paper, we propose a novel end-to-end framework, called ET5, to better capture the entailment information for question generation in CMRC, and thus eliminate the information gap between decision making and question generation. By conducting a parameter shared encoder between answer generation decoder and entailment reasoning decoder, the answer generation decoder can utilize the fine-grained entailment reasoning information to enhance the performance of question generation. Experimental results suggest that the proposed framework ET5 achieves the new state-of-the-art results on the ShARC benchmark.

Acknowledgements

The work is supported by National Key R&D Plan (No.2020AAA0106600), National Natural Science Foundation of China (No.U21B2009, 62172039 and L1924068).

References

Eunsol Choi, He He, Mohit Iyyer, Mark Yatskar, Wentau Yih, Yejin Choi, Percy Liang, and Luke Zettlemoyer. 2018. QuAC: Question answering in context. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 2174–2184, Brussels, Belgium. Association for Computational Linguistics.

Kevin Clark, Minh-Thang Luong, Quoc V. Le, and Christopher D. Manning. 2020. Electra: Pre-training text encoders as discriminators rather than generators. In *International Conference on Learning Representations*.

Leyang Cui, Yu Wu, Shujie Liu, Yue Zhang, and Ming Zhou. 2020. MuTual: A dataset for multi-turn dialogue reasoning. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 1406–1416, Online. Association for Computational Linguistics.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, Minneapolis, Minnesota. Association for Computational Linguistics.

Li Dong, Nan Yang, Wenhui Wang, Furu Wei, Xiaodong Liu, Yu Wang, Jianfeng Gao, Ming Zhou, and Hsiao-Wuen Hon. 2019. Unified language model pre-training for natural language understanding and generation. *Advances in Neural Information Processing Systems*, 32.

Yifan Gao, Jingjing Li, Michael R Lyu, and Irwin King. 2021. Open-retrieval conversational machine reading. *arXiv preprint arXiv:2102.08633*.

Yifan Gao, Chien-Sheng Wu, Shafiq Joty, Caiming Xiong, Richard Socher, Irwin King, Michael Lyu, and Steven C.H. Hoi. 2020a. Explicit memory tracker with coarse-to-fine reasoning for conversational machine reading. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*.

Yifan Gao, Chien-Sheng Wu, Jingjing Li, Shafiq Joty, Steven C.H. Hoi, Caiming Xiong, Irwin King, and Michael Lyu. 2020b. Discern: Discourse-aware entailment reasoning network for conversational machine reading. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 2439–2449.

Carolin Lawrence, Bhushan Kotnis, and Mathias Niepert. 2019. Attending to future tokens for bidirectional sequence generation. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 1–10, Hong Kong, China. Association for Computational Linguistics.

Friedrich Wilhelm Levi. 1942. *Finite geometrical systems: six public lectues delivered in February, 1940, at the University of Calcutta*. University of Calcutta.

Jing Li, Aixin Sun, and Shafiq Joty. 2018. Segbot: a generic neural text segmentation model with pointer network. In *Proceedings of the 27th International Joint Conference on Artificial Intelligence*, pages 4166–4172.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2020. Ro{bert}a: A robustly optimized {bert} pretraining approach.

Ilya Loshchilov and Frank Hutter. 2018. Fixing weight decay regularization in adam. In *International Conference on Learning Representations*.

Siru Ouyang, Zhuosheng Zhang, and Hai Zhao. 2021. Dialogue graph modeling for conversational machine reading. In *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*, pages 3158–3169, Online. Association for Computational Linguistics.

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. Bleu: a method for automatic evaluation of machine translation. In *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics*, Philadelphia, Pennsylvania, USA.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J Liu. 2020. Exploring the limits of transfer learning with a unified text-to-text transformer. *Journal of Machine Learning Research*, 21(140):1–67.

Siva Reddy, Danqi Chen, and Christopher D. Manning. 2019. CoQA: A conversational question answering challenge. *Transactions of the Association for Computational Linguistics*, 7.

Marzieh Saeidi, Max Bartolo, Patrick Lewis, Sameer Singh, Tim Rocktäschel, Mike Sheldon, Guillaume Bouchard, and Sebastian Riedel. 2018. Interpretation of natural language rules in conversational machine reading. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*.

Michael Schlichtkrull, Thomas N Kipf, Peter Bloem, Rianne Van Den Berg, Ivan Titov, and Max Welling. 2018. Modeling relational data with graph convolutional networks. In *European semantic web conference*.

Kai Sun, Dian Yu, Jianshu Chen, Dong Yu, Yejin Choi, and Claire Cardie. 2019. DREAM: A challenge data set and models for dialogue-based reading comprehension. *Transactions of the Association for Computational Linguistics*.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Ł ukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in Neural Information Processing Systems*.

Nikhil Verma, Abhishek Sharma, Dhiraj Madan, Danish Contractor, Harshit Kumar, and Sachindra Joshi. 2020. Neural conversational QA: Learning to reason vs exploiting patterns. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 7263–7269, Online. Association for Computational Linguistics.

Zhuosheng Zhang, Siru Ouyang, Hai Zhao, Masao Utiyama, and Eiichiro Sumita. 2021. Smoothing dialogue states for open conversational machine reading. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*.

Victor Zhong and Luke Zettlemoyer. 2019. E3: Entailment-driven extracting and editing for conversational machine reading. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*.

CoHS-CQG: Context and History Selection for Conversational Question Generation

Xuan Long Do[1,2,*], **Bowei Zou**[2], **Liangming Pan**[3], **Nancy F. Chen**[2], **Shafiq Joty**[1,4], **Ai Ti Aw**[2]
[1]Nanyang Technological University, Singapore
[2]Institute for Infocomm Research, A*STAR, Singapore
[3]National University of Singapore, [4]Salesforce AI Research
{xuanlong001@e.ntu, srjoty@ntu}.edu.sg
{zou_bowei, nfychen, aaiti}@i2r.a-star.edu.sg
liangmingpan@u.nus.edu

Abstract

Conversational question generation (CQG) serves as a vital task for machines to assist humans, such as interactive reading comprehension, through conversations. Compared to traditional single-turn question generation (SQG), CQG is more challenging in the sense that the generated question is required not only to be meaningful, but also to align with the occurred conversation history. While previous studies mainly focus on how to model the flow and alignment of the conversation, there has been no thorough study to date on which parts of the context and history are necessary for the model. We argue that shortening the context and history is crucial as it can help the model to optimise more on the conversational alignment property. To this end, we propose *CoHS-CQG*, a two-stage CQG framework, which adopts a *CoHS* module to shorten the context and history of the input. In particular, *CoHS* selects contiguous sentences and history turns according to their relevance scores by a *top-p* strategy. Our model achieves state-of-the-art performances on CoQA in both the answer-aware and answer-unaware settings. Our work will be publicly available at https://github.com/dxlong2000/CoHS-CQG.

1 Introduction

One of the key goals of AI is to build systems that can understand and assist humans through conversation. In conversations, asking questions is an important dialogue act that serves as an important communication skill for AI models to better interact with humans (Allen et al., 2007). Taking it a step further, asking good questions could facilitate collecting users' intentions and feedback, starting a new topic, and enhancing the interactivity and persistence of dialogues. In NLP, this line of research is formulated as the task of *Conversational Question Generation (CQG)*, which aims to generate questions based on the conversation history (Pan et al., 2019a; Nakanishi et al., 2019).

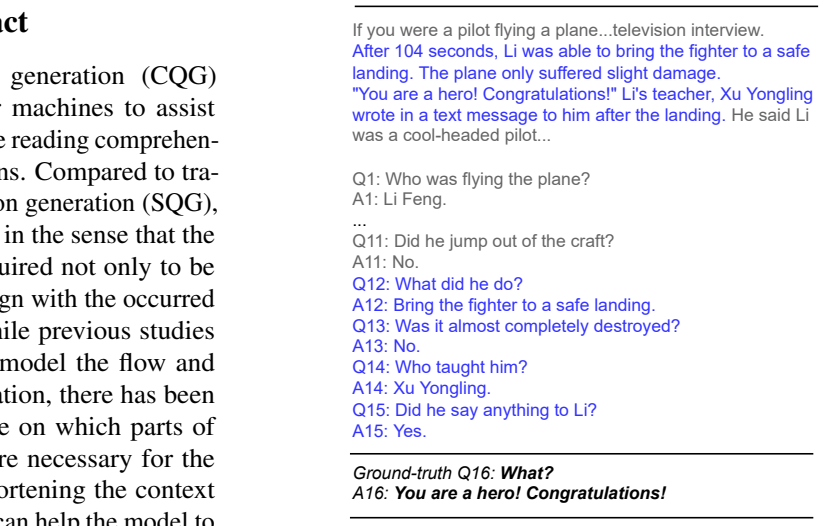

Figure 1: A dialog sample in CoQA validation set, which reflects not all the sentences in the context and history turns are necessary for generating Q16.

Although question generation has been explored intensively (Pan et al., 2019b; Lu and Lu, 2021), most existing studies focus on single-turn question generation, which aims to generate one question from a given context. However, in the scene of conversation, it poses an additional challenge of multi-turn question generation, in which the model is required to generate multiple questions during the conversation, and the generated questions should be coherent and form a smooth conversation flow.

Despite the intensive exploration of single-turn QG, less attention has been drawn on CQG. Previous work of CQG mostly focuses on solving two main challenges: coreference alignment and conversation flow. Gao et al. (2019) proposed CFNet to model coreference alignment and conversation flow explicitly. Gu et al. (2021) proposed ChainCQG, a two-stage model with two modules: the Answer Encoder learns the representation of the context and answer in each turn, and the Question Genera-

* Contribution during the internship at Institute for Infocomm Research.

Proceedings of the 29th International Conference on Computational Linguistics, pages 580–591
October 12–17, 2022.

tion learns the representation of the conversational history and generates the next turn's question. However, most previous work makes use of all the context and conversation history indiscriminately. On the contrary, we argue that not all sentences in the context, and not all previous turns in the conversation history are necessary for the model to generate the question in the next turn, and they may even harm the generation capacity of the model. Figure 1 shows such an example, where we see that only the blue parts of the context and history are necessary for generating the 16-th turn's question.

To address the above concerns, we introduce *CoHS-CQG*, a two-stage CQG model, as described in Figure 2. In the answer-aware stage, we input sentences in the context and the conversation history turns into a pretrained sentence-transformer (Reimers and Gurevych, 2019) for calculateing the relevance scores of the `(sentence, history turn)` pairs. Context and History Selection (*CoHS*) module (Section 3.1) is then employed to shorten the context and conversation history concurrently by selecting *top-p* {sentence, history turn} pairs of contiguous sentences in the context, and contiguous previous turns in the history. The shortened context and history are then fed into a T5-based (Raffel et al., 2020) question generation model to generate the questions. By training the model on the shortened context and history, we observe that generated questions are generally more aligned with the conversation, which reflects that the model is optimised better in the conversational alignment. Our model achieves state-of-the-art results in the CQG answer-aware setting on both automatic evaluation metrics and a careful human evaluation. In the answer-unaware stage, we propose a pipeline approach (Section 3.2) to leverage our model on the answer-unaware setting, which also achieves the state-of-the-art performance on human evaluation.

In summary, our main contributions are: *(1)* *CoHS-CQG*, a two-stage CQG framework for both answer-aware and answer-unaware settings, which adopts a novel module, *CoHS*, to shorten the context and history before inputting them to the QG model. *CoHS* can be plugged into any CQG model, which makes it easily reproducible, *(2)* new strong state-of-the-art performances on answer-aware and answer-unaware CQG, and *(3)* a thorough analysis and evaluation about the selection capacity of *CoHS*.

2 Related Work

2.1 Single-turn Question Generation

Single-turn Question Generation (SQG) has been focused extensively through the years. Early studies relied on syntactic transformation to convert declarative sentences to questions (Heilman and Smith, 2010; Khullar et al., 2018). Recently, Du et al. (2017) showed the limitations of such rule-based methods and formulated the question generation problem as a sequence-to-sequence task. The task is generally cast into two main streams: answer-aware and answer-unaware.

In the answer-aware setting, the target answer is revealed to SQG models. The models then have to solve the task by either treating the answer as an extra input feature or encoding the answer by a separate network (Pan et al., 2019b). However, the answer is not available in the answer-unaware case. Traditional approaches in this setting include two main steps: answer-span selection and answer-aware question generation (Du and Cardie, 2017; Subramanian et al., 2018). Recent state-of-the-art systems in answer-aware setting (Dong et al., 2019; Qi et al., 2020b; Lelkes et al., 2021; Murakhovs'ka et al., 2021) and in answer-unaware setting (Scialom et al., 2019; Lopez et al., 2020) all rely on transformer-based architectures, and they are commonly evaluated on SQuAD (Rajpurkar et al., 2016).

2.2 Conversational Question Generation

Despite the intensive exploration in both settings of the single-turn QG task, there is much less exploration in Conversational Question Generation (CQG). Most of the previous studies focus on the answer-unaware setting (Pan et al., 2019a; Nakanishi et al., 2019; Qi et al., 2020a), but a limited number of works are in the answer-aware setting. In general, there are two main challenges in CQG: coreference alignment and conversation flow. Models in the answer-aware setting then have been proposed to solve those problems such as CFNet (Gao et al., 2019), by which the coreference alignment and conversation flow are modeled explicitly, and ChainCQG (Gu et al., 2021), which contains two modules: the Answer Encoder (AE) module learns the representation of the context and answer span in each turn, and the Question Generation (QG) module learns the representation of the conversational history and generates the next turn's question.

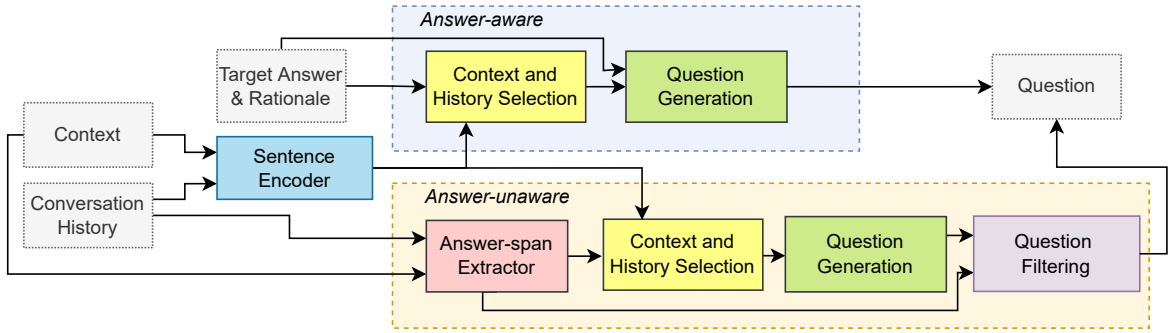

Figure 2: An overview of our proposed framework *CoHS-CQG*. The modules with the same color have the same functionality.

3 CoHS-CQG

We formulate the conversational question generation (CQG) task in two different settings: *answer-aware* and *answer-unaware*. For the *answer-aware* CQG, given the referential context $C = \{c_1, c_2, ..., c_m\}$ where c_i is the i-th sentence in context, the conversation history $H_n = \{(q_1, a_1), (q_2, a_2), ..., (q_{n-1}, a_{n-1})\}$, where (q_i, a_i) is the i-th turn question-answer pair in conversation, the target answer a_n, and the rationale r_n, as input $\mathcal{D}_n^a = \{C, H_n, a_n, r_n\}$, the model then learns to generate the question q_n. The rationale r_n is an associated text span from the context which contains or explains the given answer a_n. For the *answer-unaware* CQG, however, given $\mathcal{D}_n^u = \{C, H_n\}$, the model learns to generate the current question q_n without a_n and r_n.

Our proposed CoHS-CQG framework is shown in Figure 2. The context C and conversation history H_n are first fed into a Sentence Encoder (SE) to compute the relevance scores. In the answer-aware setting, the relevance scores, together with a_n and r_n are input to the Context and History Selection (CoHS) for selecting the parts of C and H_n that are most relevant to the current generation turn, and they are then input to the Question Generation (QG) module. In the answer-unaware case, since a_n is unavailable, C and H_n are first fed into the Answer-span Extractor (AE) to extract a_n, and a_n is later verified by the Question Filtering (QF) module.

3.1 Answer-aware CQG

Sentence Encoder (SE) Given the context C and conversation history H_n, we employ a pretrained sentence-transformer (Reimers and Gurevych, 2019) to embed each sentence c_i in C, and each question-answer pair (q_j, a_j) in H_n (i.e. the concatenation of q_j and a_j), respectively. We then compute a relevance matrix $\mathbf{T} \in \mathbb{R}^{|C| \times |H_n|}$ as

$$\mathbf{T}[i][j] = rel(c_i, (q_j, a_j)) = \frac{\mathbf{a}_i \mathbf{b}_j}{|\mathbf{a}_i||\mathbf{b}_j|}, \quad (1)$$

where $\mathbf{a}_i$ and $\mathbf{b}_j$ are the embeddings of c_i and $concat(q_j, a_j)$, respectively, the relevance score $rel(\cdot)$ is defined as the *cosine* similarity, and $1 \le i \le m, 1 \le j \le n-1$.

Context and History Selection (CoHS) To generate the current question q_n, existing CQG models (Gao et al., 2019; Gu et al., 2021) commonly take the full context C and all the previous question-answer pairs H_n as input. Moreover, in leveraging conversation history, some studies (Ohsugi et al., 2019; Zhao et al., 2021) have begun to consider how to select historical information related to the current utterance, but only simply selected the last k turns. We argue that not all parts of the context and conversation history are necessary for the model to generate the current question since the topic in a conversation may shift. On the contrary, introducing irrelevant parts worsens the performance of the model (See Table 1 and 4). To address this problem, we propose a *top-p* CoHS strategy that dynamically selects the most relevant sentences in the context concurrently with the most relevant preceding conversation utterances.

Given the input $\mathcal{D}_n^a = \{C, H_n, a_n, r_n\}$ and the relevance matrix $\mathbf{T}$, CoHS aims to select the *top-p* of sentences and QA pairs from C and H_n, respectively. Inspired by Holtzman et al. (2020), we formulate our *top-p* CoHS strategy as, finding the subset $C_{sub} = \{c_{v-u}, c_{v-u+1}, ..., c_{v-1}\}$ and $H_{sub} = \{(q_{n-k}, a_{n-k}), (q_{n-k+1}, a_{n-k+1}), ..., (q_{n-1}, a_{n-1})\}$,

to satisfy

$$minimize(u + k) \tag{2}$$

$$\sum_{i=v-u}^{v-1} \sum_{j=n-k}^{n-1} \mathbf{T}[i][j] \geq p \tag{3}$$

$$(q_{n-1}, a_{n-1}) \in H_{sub}, c_s \in C_{sub} \tag{4}$$

where p is a given threshold, and c_s is the sentence that contains r_n. First, the optimizing goal is to minimize the sum of $u + k$, where u and k are the numbers of the contiguous sentences from C and contiguous preceding conversation turns from H_n, respectively (Eq.(2)). Then, the sentences and conversation turns with higher similarity than the threshold p are selected as the candidates for building C_{sub} and H_{sub} (Eq.(3)). In addition, since the sentence containing the ground-truth rationale c_s and the last previous conversation turn (a_{n-1}, q_{n-1}) are intuitively relevant for generating the current question, we set two constraints in Eq.(4). Note that the contiguity of C_{sub} and H_{sub} is necessary due to the integrity and coherence of input. The advantage of the heuristic *top-p* CoHS strategy is that CQG models can dynamically select the most relevant C_{sub} and H_{sub} according to different conversation progress, which well adapts when topic shifting. When $H_n = \varnothing$, we select five sentences around c_s (see Appendix A.1).

Question Generation (QG) We employ a T5 (Raffel et al., 2020) as our question generation model. To fine-turn the T5 on the shortened context and history, we concatenate the input $\mathcal{D}_n^a = \{C, H_n, a_n, r_n\}$ in format: `Answer:` a_n, r_n `Context:` C_{sub} `[SEP]` H_{sub}. The model then learns to generate the target question q_n.

3.2 Answer-unaware CQG

In Section 3.1, we utilize 1) the ground-truth previous conversation history H_n, and 2) the ground-truth current answer a_n and rationale r_n, to verify how well the model performs in generating the current question q_n. However, in a more realistic scenario such as a dialogue system, it is necessary to verify whether the model has a good ability to generate questions continuously, that is, the coherence and fluency of the generated questions. To this end, we propose an *answer-unaware* process as shown in Figure 2, including *Answer-span Extractor*, *CoHS* (depicted in Section 3.1), *QG* (depicted in Section 3.1), and *Question Filtering*.

Answer-span Extractor (AE) First, we treat the earliest sentence in the context as the current rationale r_n such that r_n does not contain any rationales of previous turns. Then, a T5 model is trained on SQuAD (Rajpurkar et al., 2016) to predict the target answer span (a) given its original sentence in context (r). We use the model to extract a_n from r_n. Note that we remove the answer spans that are the same as those of previous turns, to ensure that the generated questions are informative enough. Finally, we obtain a set of selected candidate answer spans $A_n = \{a_1^*, a_2^*, ..., a_t^*\}$. Each $a_i^* \in A_n$, together with r_n, and $\mathcal{D}_n^u$ are fed into the CoHS and QG modules to generate the candidate question q_i^*.

Question Filtering (QF) Under the answer-unaware setting, since the conversation history is not manually-labeled, we observe that one type of the common errors is that the generated question may not be answerable by the given context, or its answer may not the provided target answer a_i^*. To address this issue, we train a T5 model on CoQA (Reddy et al., 2019) to answer the generated question q_i^*, and only accept q_i^* if the predicted answer is the same as a_i^*.

4 Experimentation

4.1 Experimental Settings

Dataset We conduct experiments on CoQA (Reddy et al., 2019), a large-scale CQA dataset including 8k conversations. Each conversation contains a referential context and multiple question-answer pairs. In total, there are 127k question-answer pairs collected via Amazon Mechanical Turk. The key characteristics of this dataset are its factoid questions (*i.e.* What? Where? When? How long?) and free-form answers. Since the test set of CoQA is unavailable, we randomly sample 10% of the original training set as our new *validation set*, and keep the original validation set as our *test set* so that future works can be compared with us.

Baseline Models We use a $T5_{base}$ (220M) as our *CoHS-CQG*'s backbone. For the answer-aware baselines, we reimplement CFNet (Gao et al., 2019), an effective CQG framework. We also fine-tune a $T5_{base}$ (Raffel et al., 2020) and a $BART_{base}$ (Lewis et al., 2020), the SOTA transformer-based generation models, on CoQA. For the answer-unaware baseline, we compare with the SOTA framework ReDR (Pan et al., 2019a).

Model	ROUGE-L	B1	B2	B3	B4	METEOR	BERTScore
CFNet	41.25	34.24	22.71	16.57	12.39	27.76	91.43
ChainCQG*	42.22	35.54	26.03	19.63	14.54	30.97	92.54
BART_{base}	44.77	35.86	26.32	19.84	15.09	31.60	92.95
T5_{base}	45.80	39.09	29.04	22.17	17.03	34.09	93.07
T5_{base} + *dyn-HS (p = 0.5)*	48.64	40.83	30.74	23.64	18.18	36.49	93.43
T5_{base} + *dyn-CS (p = 1)*	49.69	41.62	31.44	24.29	18.72	37.42	93.61
CoHS-CQG (Ours, p = 5)	**49.91**	**42.10**	**31.86**	**24.65**	**19.11**	**37.76**	**93.65**

Table 1: Automated evaluation results on our test set (i.e. CoQA validation set). *dyn-HS* and *dyn-CS* are *dynamic History Selection* and *dynamic Context Selection* respectively (Section 4.4). B1 to B4 denotes BLEU 1-4.

Implementation Details We initialise *CoHS-CQG* with pretrained checkpoints from Huggingface (Wolf et al., 2020). We use AdamW (Loshchilov and Hutter, 2019) with the warmup ratio of 0.1 and the initial learning rate of 1e-4. We train the model for 100k iterations with standard window size of 512, and use a Beam search decoding strategy with beam size of 4.

Evaluation Metrics We compute the standard n-gram-based similarity metrics, which are commonly used for text generation, including ROUGE-L (Lin, 2004), BLEU (1-4) (Papineni et al., 2002), and METEOR (Banerjee and Lavie, 2005). We compute BLEU 1-4 by *corpus_bleu* function from NLTK library.[1] We compute ROUGE-F scores in our evaluations by Python implementation of *rouge-score* library.[2] We also calculate BERTScore (Zhang et al., 2020), a similarity score between the generated and ground-truth texts by using deep contextualized embeddings.

Human evaluation is also important to the CQG task since the CQG model may generate the question for the following turn in multiple ways, given the target answer. As such, we conduct human evaluation on both the answer-aware setting and answer-unaware setting.

4.2 Automatic Evaluation

Table 1 shows the automatic evaluation results. We observe that *CoHS-CQG (p = 5)* achieves state-of-the-art performance on all the automatic evaluation metrics. In particular, we derive 3 observations. First, *CoHS-CQG* improves its original baseline T5_{base} significantly. Second, comparing to only dynamically selecting previous turns (T5_{base} + *dyn-HS*) or sentences in the context (T5_{base} + *dyn-CS*), *CoHS-CQG* achieves better performances, which indicates that dynamically selecting both is more effective. Third, with the threshold of relevance $p = 5$ (Eq.(3)), the *CoHS* module shortens the context to around 5 sentences and the history to 3 previous turns on average (Table 3), by which it achieves the best performance.

We also compare our *CoHS-CQG* with the current SOTA answer-aware CQG model, ChainCQG (Gu et al., 2021) which contains two GPT-2 (Rad-

Model	Flu.	C-Align	Ans.
Answer-aware			
BART_{base}	2.38	2.23	2.34
T5_{base}	2.72	2.44	2.54
CoHS-CQG (Ours)	**2.74**	**2.58**	**2.60**
Krippendorff's α	0.80	0.71	0.77
Answer-unaware			
ReDR	1.06	1.06	1.05
CoHS-CQG (Ours)	**2.70**	**2.41**	**2.73**
Krippendorff's α	0.84	0.85	0.82

Table 2: Human evaluation results on the validation set of CoQA. **Top**: answer-aware, on 100 random generated samples; **Bottom**: answer-unaware, on 20 random conversations. "Krippendorff's α" shows the inter-annotator agreement. *Flu.*: Fluency, *C-Align*: Conversational Alignment, *Ans.*: Answerability.

p value	Avg. #S	Avg. #P	ROUGE-L	BLEU-4
1	3.18	1.89	48.96	18.19
2	3.73	2.49	49.38	18.61
3	4.27	2.95	49.41	18.65
5	5.29	3.67	**49.91**	**19.11**
7	6.13	4.25	49.35	18.69
10	8.12	4.92	49.18	18.52
∞	16.09	7.97	45.80	17.03

Table 3: Average #Sentences, #Prev. Turns at different p values of preprocessed contexts and histories by *CoHS* of CoQA. $p = \infty$ means selecting the full context and history. *Avg. #S*: average number of sentences, *Avg. #P*: average number of previous turns.

[1] https://www.nltk.org/

[2] https://github.com/google-research/google-research/tree/master/rouge

ford et al., 2019) blocks. Since the provided codes from the authors are incomplete, and the reported results of ChainQCG in (Gu et al., 2021) are on the authors' own test set (they splitted 10% of the training set to become their own test set), we were not able to reproduce the results. Thus, we reimplement ChainCQG (denote it as ChainCQG*). We can see that *CoHS-CQG* outperforms ChainCQG* on all automatic evaluation metrics significantly.

4.3 Human Evaluation

Evaluation Setup We further conduct human evaluation to validate the results. In answer-aware case, we randomly select 100 generated questions associated with the context and conversation history. In answer-unaware case, however, since there is no ground-truth history, simply evaluating 100 random generated samples may not be a fair comparison. Thus, we first select 20 random contexts in our test set. For each context, since the number of turns generated by our model and the competing one, ReDR (Pan et al., 2019a), may not be the same, we heuristically select the first five generated turns from each model's output to compare, resulting in 100 samples in total. We hire three annotators who are English native speakers. Each annotator was instructed to rate the generated questions on a 1-3 scale (3 for the best) based on three criteria: **(1) Fluency** measures not only the grammatical correctness but also the meaning, and factual correctness of generated questions, **(2) Conversational Alignment** measures the alignment of generated questions with the given conversation, **(3) Answerability** measures whether the generated questions are well answerable or not. We measure the annotators' agreement by Krippendorff's alpha (Krippendorff, 2011). Our rating system is described in Appendix A.2.

Observations The top of Table 2 shows the averages of human scores over three annotators in the answer-aware setting. We derive two main observations. First, there is a significant improvement in the *Conversational Alignment* of *CoHS-CQG* compared to T5, which indicates that with the shortened context and history as input, the model learns to focus and align with the given conversation history much better. Second, compared to T5, there is also a slight increase in the *Answerability*, which further shows that the quality of the generated questions is improved. There is also a minor improvement in the *Fluency*, which is reasonable because T5 commonly generates fluently, grammatically and meaningfully correct questions. Our annotators have a good overall agreement with an alpha coefficient of 0.76.

#Pre. turns	ROUGE-L	BLEU-4
1	48.14	17.43
2	**48.34**	17.66
3	48.21	**17.68**
4	47.77	17.64
5	47.15	17.59
6	46.90	17.12
Full history	45.33	16.73

Table 4: Performance of $T5_{base}$ with different fixed number of previous turns on our validation set.

The bottom of Table 2 shows the human evaluation for the answer-unaware setting. First, we observe that ReDR has low *Fluency* score due to most of the generated questions are factually wrong or have no meaning associated with the given context. It also has low scores on the other two metrics as the generated questions are frequently repetitive. Second, the generated questions by *CoHS-CQG* are generally high-quality, fluent and answerable as they already passed the *Question Filtering* module. The annotators achieve a good overall inter-agreement with an alpha coefficient of 0.83.

4.4 Effects of Context and History Selection

We further conduct the studies about the performance of T5 when we dynamically select the context sentences or the previous turns but not both of them concurrently. In this section, we formulate these two problems as below.

Dynamic Context Selection In this setting, we follow the previous studies on CQA (Ohsugi et al., 2019; Zhao et al., 2021) to select the last k previous turns. The results are shown in Table 4. Since $T5_{base}$ achieves the best performance on BLEU-4 by using the last 3 previous turns, we adopt this setting in the following independent context selection experiments. Given the context C, answer a_n, rationale r_n, and the last k previous turns $H_{sub} = \{h_{n-k}, h_{n-k+1}, ..., h_{n-1}\}$, $h_i = concat(q_i, a_i)$, we formulate the context selection problem in this section as finding the smallest sub-

p value	Avg. #Sentences	ROUGE-L	BLEU-4
0	1	48.15	17.88
1	4.35	**49.69**	**18.72**
2	8.28	49.47	18.43
3	11.51	48.45	18.30
4	13.42	48.80	18.16
∞	16.09	48.31	18.01

Table 5: Average number of selected sentences from context (fixing the last 3 previous turns) of each p value (left) and performance of "T5$_{base}$ + *dyn-CS*" on our test set (right). $p = \infty$ and $p = 0$ denote utilizing the full context and only the sentence that contains the rationale respectively.

p value	Avg. #History Turns	ROUGE-L	BLEU-4
0	0	43.91	15.32
0.5	2.57	**48.64**	**18.18**
1	4.11	48.27	18.07
2	6.32	47.14	17.70
3	7.45	46.74	17.32
∞	7.97	45.80	17.03

Table 6: Average number of selected history turns from conversation (with the full context) of each p value (left) and performance of "T5$_{base}$ + *dyn-HS*" on our test set (right). $p = \infty$ and $p = 0$ denote utilizing the full history and none of the history respectively.

set $C_{sub} = \{c_{v-u}, c_{v-u+1}, ..., c_{v-1}\}$, to satisfy:

$$\sum_{x=v-u}^{v-1} \sum_{y=n-k}^{n-1} T[x][y] \geq p \quad (5)$$

$$c_s \in C_{sub} \quad (6)$$

where p is a given threshold, and c_s is the sentence that contains r_n. We name this model as T5$_{base}$ + *dyn-CS* in Table 1 where *dyn-CS* stands for dynamic Context Selection.

Table 5 shows how different values of threshold p (Eq.(5)) affects the selection and the performance of the model. We observe that with a fixed number of previous turns $k = 3$, $p = 1$ gives us the best performance on ROUGE-L and BLEU-4. By setting threshold $p = 1$, and fixed 3 previous turns, the *CoHS* module selects around 4 sentences in each context sample on average. The result indicates that selecting more contexts does not lead to better performance, which is consistent with our motivation.

Dynamic History Selection In this setting, we follow most previous works on CQA (Ohsugi et al., 2019; Zhao et al., 2021) and CQG (Gao et al., 2019; Gu et al., 2021) to use the whole context C, and then we dynamically select different numbers of previous turns. We formulate the history selection problem as finding the smallest subset $H_{sub} = \{(q_{n-k}, a_{n-k}), (q_{n-k+1}, a_{n-k+1}), ..., (q_{n-1}, a_{n-1})\}$, to satisfy:

$$\sum_{x=1}^{m} \sum_{y=n-k}^{n-1} T[x][y] \geq p \quad (7)$$

$$(q_{n-1}, a_{n-1}) \in H_{sub} \quad (8)$$

where p is a given threshold, and c_s is the sentence that contains r_n. We name this experiment as T5$_{base}$ + *dyn-HS* in Table 1 where *dyn-HS* stands for dynamic History Selection.

Table 6 shows how different values of threshold p (Eq.(7)) affects the selection and performance of the model. We can observe that with the full context, $p = 0.5$ achieves the best performance on both ROUGE-L and BLEU-4. By setting the threshold $p = 0.5$, the *CoHS* module then selects around 3 previous turns on average. This observation is in line with our following results in Table 4 (see Section 4.5), by which we conclude that with different values of fixed number of previous turns, $k = 2$ and $k = 3$ achieve the best results.

4.5 Discussion

Effects of Relevance Threshold p To further understand how the threshold p (Eq.(3)) controls the selection of context and conversation history, we conduct experiments with different values of p. Table 3 shows the average number of the selected sentences and the selected previous turns, together with the performances of T5$_{base}$ on ROUGE-L and BLEU-4. First, we can observe that on average, the difference between the #Sentences and #Pre. Turns is not large for all values of p, which reflects that our *top-p* algorithm does not prioritise selecting long context over short history and vice-versa. This indicates that the relevance scores assist the algorithm to select the context sentences, together with the history turns in a reasonable way. Second, with $p = 5$, T5$_{base}$ yields the best performance, as we discussed in Section 4.2.

Effects of Different Fixed Previous Turns In Table 4, we study with the full context, how the number of previous history turns affect the performance of the model on our validation set. We can observe that with the full context, the settings of previous history turns $k = 2$ and $k = 3$ achieve the best performances on ROUGE-L and BLEU-4, respectively. Compared to the performances in Table 6, it indicates that dynamically selecting instead of fixing the number of previous turns indeed

ID	Context & History and the Selection of CoHS-CQG (p = 5)		
1	**Answer:** his owner, **Rationale:** the cat had been abandoned by his owner **Context:** When my father was dying, I traveled a thousand miles from home to be with him in his last days. It was far more heartbreaking than I'd expected, one of the most difficult and painful times in my life. After he passed away I stayed alone in his apartment. There were so many things to deal with. It all seemed endless. I was lonely. I hated the silence of the apartment. But one evening the silence was broken: I heard crying outside. I opened the door to find a little cat on the steps. He was thin and poor. He looked the way I felt. I brought him inside and gave him a can of fish. He ate it and then almost immediately fell sound asleep. The next morning I checked with neighbors and learned that the cat had been abandoned by his owner who's moved out. So the little cat was there all alone, just like I was. As I walked back to the apartment, I tried to figure out what to do with him. Having something else to take care of seemed _ But as soon as I opened the apartment door he came running and jumped into my arms. It was clear from that moment that he had no intention of going anywhere. I started calling him Willis, in honor of my father's best friend. From then on, things grew easier. With Willis in my lap time seemed to pass much more quickly. When the time finally came for me to return home I had to decide what to do about Willis. There was absolutely no way I would leave without him. It's now been five years since my father died. Over the years, several people have commented on how nice it was of me to rescue the cat. But I know that we rescued each other. I may have given him a home but he gave me something greater. **History:** <s> What was crying? <s> a cat <s> what did the author feed it? <s> yes <s> what did the author feed it? <s> fish <s> then what did the cat do? <s> fell asleep <s> where was the author? <s> his father's apartment. <s> was the father alive by then? <s> no <s> how far did the author travel? <s> a thousand miles <s> who did the author check about the cat with? <s> neighbors <s> what did he find out? <s> the cat was abandoned		
	Ground-truth: by who?	*CoHS-CQG:* by who?	*T5:* who abandoned the cat?
2	**Answer:** in the morning, **Rationale:** It was here Paul found his brother on the morning of his arrival in London. **Context:** CHAPTER VIII ""I AM WEARY OF A HOPELESS LOVE"" Paul and Arthur shared a bachelor residence in Mayfair; shared it, that is to say, insomuch as Paul had purchased it, and was the sole proprietor, and Arthur used it whenever he could get leave from his regiment. It was here Paul found his brother on the morning of his arrival in London. They shook hands in silence; Paul did not wish to say anything for a moment. His brother's appearance had choked him. It was one o'clock, but he was still in his dressing-gown; with sunken, pale cheeks, save for one bright spot, and with faint, dark rims underneath his eyes. There were a pile of blue papers and some ominous-looking envelopes on the table before him, and Paul could not help noticing the intense pallor of the hand which rested upon them. ""I wish you would let a fellow know what time you were coming,"" Arthur said, rather peevishly, but with an attempt at a smile. ""I didn't expect you till evening, so I was having a shack before dressing. I was late last night!"" Paul banished his gravity, as far as possible, and stood with his hands in his pockets, leaning against the mantel-piece. He heartily disliked the part of mentor, and he did not wish to play it, unless he were obliged. ""It was beastly early to get up,"" he said, ""but the connection at Normanton is so much better. One has to wait two hours by the late train, and Normanton is such a hole. I don't know that I should have come up to town at all, just yet,"" he continued after a slight pause, ""only that I'm on the committee at the club this term, you know, and I haven't attended a single meeting yet. Besides, I promised Westover to put him up this time, and the half-yearly meeting's to-morrow, you know. Got any engagement? If not, you might dine with me there. Always a full night election time, you know!"" **History**: <s> Where was the joint residence? <s> in Mayfair <s> who owns it? <s> Paul <s> who else stayed there? <s> Arthur <s> how often? <s> whenever he was on leave <s> from what? <s> from his regiment. <s> who is his brother? <s> Paul and Arthur were brothers <s> where did they meet up? <s> in the bachelor residence"		
	Ground-truth: when?	*CoHS-CQG:* when?	*T5:* When did Paul find his brother?

Figure 3: Case studies on the CoQA validation set and the results of T5 (with full context and history) and CoHS-CQG (with shorten context and history). The texts of rationales are underlined. The selected texts in context and history by our *CoHS (p = 5)* module are highlighted in blue.

Model	Flu.	C-Align	Ans.
Answer-unaware			
CoHS-CQG w/o AE	2.13	1.76	1.64
CoHS-CQG w/o QF	2.16	1.98	2.02
CoHS-CQG (Ours)	**2.74**	**2.58**	**2.60**
Krippendorff's α	0.82	0.79	0.77

Table 7: Human evaluation results for the ablation studies of AE and QF modules on the validation set of CoQA. "Krippendorff's α" shows the inter-annotator agreement. *Flu.*: Fluency, *C-Align*: Conversational Alignment, *Ans.*: Answerability.

improves the performance.

4.6 Ablation Studies

Ablation of Answer-span Extractor (AE) We conduct an ablation study for the *Answer-span Extractor (AE)* (*CoHS-CQG w/o AE*), in which we replace the predicted answer span a_n with the rationale r_n (a sentence in the context C). The results are shown in Table 7. Note that in this experiment, we also remove the *Question Filtering (QF)*. As expected, the *Answerability* and *Conversational Alignment* drop significantly, which is explainable since the rationale r_n may contain redundant information, thus it is not suitable to be r_n.

Ablation of Question Filtering (QF) We study the ablation of the Question Filtering (*CoHS-CQG w/o QF*), in which we use all the generated questions. The results are shown in Table 7. As we can see, the *Answerability* and *Conversational Alignment* decrease significantly. We observe that without QF, there may have been turns in which the questions are similar with the same answers, which further proves the necessity of this module.

4.7 Case study: Effectiveness of *CoHS*

When carefully studying the performances of T5 and BART, we observe that the key for these models to gain high scores on n-gram automatic metrics, such as BLEU and ROUGE, is *focusing on the history to optimise the conversational alignment*. We argue that, with a long context and the whole conversation history, the input likely tends to distract the attention of the models on the given conversation history. The models in these cases mostly focus on the given answer and rationale to generate the question, rather than highly focusing on the history. Figure 3 lists some of the examples whereby we draw the above conclusion.

Considering the first example in Figure 3, we observe that with the full context and history, the T5 model mostly relies on the rationale "*the cat had been abandoned by his owner*" to generate the question "*who abandoned the cat?*". Although the question is somehow aligned with the given conversation history, it is not close enough to the gold question "*by who?*". The other two examples in Figure 3 are also the same, and we observe a lot of cases that are similar to them. We argue that in order to generate such questions like "*by who?*", intuitively, the model should pay significant attention to the conversation history to optimise the conversational alignment. By inputting to the model the shortened context and history, we can see that the generated questions in Figure 3 by *CoHS-CQG* indeed change, and they are exactly the same as the ground-truth questions. This improvement reflects that training the model such as T5 with the shortened context and history samples indeed guides the model to optimising more on the conversational alignment property instead of just heavily focusing on the target answer and rationale.

5 Conclusion

This paper presents *CoHS-CQG*, a two-stage framework for CQG, which adopts a *CoHS* module to dynamically select relevant context and conversation history for generating the question in the current turn. Experimental results on CoQA demonstrate that the proposed *CoHS-CQG* achieves state-of-the-art performances in both answer-aware and answer-unaware settings. Our extensive analysis and studies show the effectiveness of *CoHS* in improving the CQG models. In future work, we will focus on how to select the contiguous question-worthy content from the paragraph by reasoning.

Acknowledgements

This research has been supported by the Institute for Infocomm Research of A*STAR (CR-2021-001). We would like to thank Tran Anh Tai for his support, and anonymous reviewers for their valuable feedback on how to improve the paper.

References

James Allen, Nathanael Chambers, George Ferguson, Lucian Galescu, Hyuckchul Jung, Mary Swift, and William Taysom. 2007. Plow: A collaborative task learning agent. In *Proceedings of the 22nd National Conference on Artificial Intelligence - Volume 2*, AAAI'07, page 1514–1519. AAAI Press.

Satanjeev Banerjee and Alon Lavie. 2005. METEOR: An automatic metric for MT evaluation with improved correlation with human judgments. In *Proceedings of the ACL Workshop on Intrinsic and Extrinsic Evaluation Measures for Machine Translation and/or Summarization*, pages 65–72, Ann Arbor, Michigan. Association for Computational Linguistics.

Li Dong, Nan Yang, Wenhui Wang, Furu Wei, Xiaodong Liu, Yu Wang, Jianfeng Gao, Ming Zhou, and Hsiao-Wuen Hon. 2019. Unified language model pre-training for natural language understanding and generation. In *Advances in Neural Information Processing Systems*, volume 32. Curran Associates, Inc.

Xinya Du and Claire Cardie. 2017. Identifying where to focus in reading comprehension for neural question generation. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 2067–2073, Copenhagen, Denmark. Association for Computational Linguistics.

Xinya Du, Junru Shao, and Claire Cardie. 2017. Learning to ask: Neural question generation for reading comprehension. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1342–1352, Vancouver, Canada. Association for Computational Linguistics.

Yifan Gao, Piji Li, Irwin King, and Michael R. Lyu. 2019. Interconnected question generation with coreference alignment and conversation flow modeling. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 4853–4862, Florence, Italy. Association for Computational Linguistics.

Jing Gu, Mostafa Mirshekari, Zhou Yu, and Aaron Sisto. 2021. ChainCQG: Flow-aware conversational question generation. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*, pages 2061–2070, Online. Association for Computational Linguistics.

Michael Heilman and Noah A. Smith. 2010. Good question! statistical ranking for question generation. In *Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, pages 609–617, Los Angeles, California. Association for Computational Linguistics.

Ari Holtzman, Jan Buys, Li Du, Maxwell Forbes, and Yejin Choi. 2020. The curious case of neural text degeneration. In *International Conference on Learning Representations*.

Payal Khullar, Konigari Rachna, Mukul Hase, and Manish Shrivastava. 2018. Automatic question generation using relative pronouns and adverbs. In *Proceedings of ACL 2018, Student Research Workshop*, pages 153–158, Melbourne, Australia. Association for Computational Linguistics.

Klaus Krippendorff. 2011. Computing krippendorff's alpha-reliability. *Computing*, 1.

Adam D. Lelkes, Vinh Q. Tran, and Cong Yu. 2021. Quiz-style question generation for news stories. In *Proceedings of the Web Conference 2021*, WWW '21, page 2501–2511, New York, NY, USA. Association for Computing Machinery.

Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Veselin Stoyanov, and Luke Zettlemoyer. 2020. BART: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7871–7880, Online. Association for Computational Linguistics.

Chin-Yew Lin. 2004. ROUGE: A package for automatic evaluation of summaries. In *Text Summarization Branches Out*, pages 74–81, Barcelona, Spain. Association for Computational Linguistics.

Luis Enrico Lopez, Diane Kathryn Cruz, Jan Christian Blaise Cruz, and Charibeth Cheng. 2020. Simplifying paragraph-level question generation via transformer language models.

Ilya Loshchilov and Frank Hutter. 2019. Decoupled weight decay regularization. In *International Conference on Learning Representations*.

Chao-Yi Lu and Sin-En Lu. 2021. A survey of approaches to automatic question generation:from 2019 to early 2021. In *Proceedings of the 33rd Conference on Computational Linguistics and Speech Processing (ROCLING 2021)*, pages 151–162, Taoyuan, Taiwan. The Association for Computational Linguistics and Chinese Language Processing (ACLCLP).

Lidiya Murakhovs'ka, Chien-Sheng Wu, Tong Niu, Wenhao Liu, and Caiming Xiong. 2021. Mixqg: Neural question generation with mixed answer types.

Mao Nakanishi, Tetsunori Kobayashi, and Yoshihiko Hayashi. 2019. Towards answer-unaware conversational question generation. In *Proceedings of the 2nd Workshop on Machine Reading for Question Answering*, pages 63–71, Hong Kong, China. Association for Computational Linguistics.

Yasuhito Ohsugi, Itsumi Saito, Kyosuke Nishida, Hisako Asano, and Junji Tomita. 2019. A simple but effective method to incorporate multi-turn context with BERT for conversational machine comprehension. In *Proceedings of the First Workshop on NLP for Conversational AI*, pages 11–17, Florence, Italy. Association for Computational Linguistics.

Boyuan Pan, Hao Li, Ziyu Yao, Deng Cai, and Huan Sun. 2019a. Reinforced dynamic reasoning for conversational question generation. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 2114–2124, Florence, Italy. Association for Computational Linguistics.

Liangming Pan, Wenqiang Lei, Tat-Seng Chua, and Min-Yen Kan. 2019b. Recent advances in neural question generation.

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. Bleu: A method for automatic evaluation of machine translation. In *Proceedings of the 40th Annual Meeting on Association for Computational Linguistics*, ACL '02, page 311–318, USA. Association for Computational Linguistics.

Peng Qi, Yuhao Zhang, and Christopher D. Manning. 2020a. Stay hungry, stay focused: Generating informative and specific questions in information-seeking conversations. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 25–40, Online. Association for Computational Linguistics.

Weizhen Qi, Yu Yan, Yeyun Gong, Dayiheng Liu, Nan Duan, Jiusheng Chen, Ruofei Zhang, and Ming Zhou. 2020b. ProphetNet: Predicting future n-gram for sequence-to-SequencePre-training. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 2401–2410, Online. Association for Computational Linguistics.

Alec Radford, Jeff Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. 2019. Language models are unsupervised multitask learners.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J. Liu. 2020. Exploring the limits of transfer learning with a unified text-to-text transformer. *Journal of Machine Learning Research*, 21(140):1–67.

Pranav Rajpurkar, Jian Zhang, Konstantin Lopyrev, and Percy Liang. 2016. SQuAD: 100,000+ questions for machine comprehension of text. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 2383–2392, Austin, Texas. Association for Computational Linguistics.

Siva Reddy, Danqi Chen, and Christopher D. Manning. 2019. CoQA: A conversational question answering challenge. *Transactions of the Association for Computational Linguistics*, 7:249–266.

Nils Reimers and Iryna Gurevych. 2019. Sentence-bert: Sentence embeddings using siamese bert-networks. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics.

Thomas Scialom, Benjamin Piwowarski, and Jacopo Staiano. 2019. Self-attention architectures for answer-agnostic neural question generation. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 6027–6032, Florence, Italy. Association for Computational Linguistics.

Sandeep Subramanian, Tong Wang, Xingdi Yuan, Saizheng Zhang, Adam Trischler, and Yoshua Bengio. 2018. Neural models for key phrase extraction and question generation. In *Proceedings of the Workshop on Machine Reading for Question Answering*, pages 78–88, Melbourne, Australia. Association for Computational Linguistics.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Remi Louf, Morgan Funtowicz, Joe Davison, Sam Shleifer, Patrick von Platen, Clara Ma, Yacine Jernite, Julien Plu, Canwen Xu, Teven Le Scao, Sylvain Gugger, Mariama Drame, Quentin Lhoest, and Alexander Rush. 2020. Transformers: State-of-the-art natural language processing. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 38–45, Online. Association for Computational Linguistics.

Tianyi Zhang, Varsha Kishore, Felix Wu, Kilian Q. Weinberger, and Yoav Artzi. 2020. Bertscore: Evaluating text generation with bert. In *International Conference on Learning Representations*.

Jing Zhao, Junwei Bao, Yifan Wang, Yongwei Zhou, Youzheng Wu, Xiaodong He, and Bowen Zhou. 2021. RoR: Read-over-read for long document machine reading comprehension. In *Findings of the Association for Computational Linguistics: EMNLP 2021*, pages 1862–1872, Punta Cana, Dominican Republic. Association for Computational Linguistics.

A Appendix

A.1 Comparison with Static Context Selection

To compare our dynamic context selection strategy with a static way, we simply select *five* context sentences around c_s in the context $C = \{c_1, c_2, ..., c_m\}$ since Table 5 shows that around more than four sentences on average achieves the best performance. To this end, we consider a simple heuristic method as below. If $3 \leq s \leq m-2$, $C_{sub} = \{c_{s-2}, c_{s-1}, c_s, c_{s+1}, c_{s+2}\}$; else if $s \leq 2$, $C_{sub} = \{c_1, c_2, c_3, c_4, c_5\}$; and if $s \geq m-1$, $C_{sub} = \{c_{m-4}, c_{m-3}, c_{m-2}, c_{m-1}, c_m\}$. We then select five sentences in the context by this way and use the whole conversation history. The result shows that the T5$_{base}$ model yields 17.24 of BLEU-4 and 47.02 of ROUGE-L, which slightly outperforms the T5$_{base}$ baseline using the full context in Table 1.

A.2 Human Rating System

In this section, we describe how our annotators are instructed to give the points in three criteria *Fluency*, *Conversational Alignment*, and *Answerability*. There are three main notes. First, *Fluency* measures not only the grammatical correctness, but also measures the meaning and factual correctness of the question with the given context. Second, in the answer-unaware setting, as there is no golden history, we do not define the *Score 2* in the *Conversational Alignment* as in the answer-aware setting. Third, for the *Answerability* criterion in the answer-unaware setting, the target answer and target rationale are unavailable. However, since our approach first selects the rationale, and then extracts the candidate answers from it to generate the questions, we still evaluate the quality of our questions (*Score 2, 3*) with the selected candidate answers by the *Question Filtering* module (Section 3). For the details, see Figure 4.

Criterion	Human Rating System	
Fluency	Answer-aware	**Score 1:** The generated question has no meaning/factually wrong with the information from the context. **Score 2:** The generated question is good, but has a small grammatical error. **Score 3:** The generated question is gramatically correct and factually correct with the information from the context.
	Answer-unaware	**Score 1:** The generated question has no meaning/factually wrong with the information from the context. **Score 2:** The generated question is good, but has a small grammatical error. **Score 3:** The generated question is gramatically correct and factually correct with the information from the context.
Conversational Alignment	Answer-aware	**Score 1:** Generated question is totally irrelevant to the conversation history. **Score 2:** Generated question is aligned to the conversation history, however, it has a differennt meaning with the golden. **Score 3:** Perfect, generated question is aligned to the conversation history and asks about the same as the golden.
	Answer-unaware	**Score 1:** Generated question is totally irrelevant to the conversation history. **Score 3:** Generated question is aligned to the conversation history.
Answerability	Answer-aware	**Score 1:** Generated question is not answerable by the context. **Score 2:** Generated question is answerable by the context, but does not have the answer as the the target answer. **Score 3:** Perfect, gennerated question is answerable by the context and its answer is target answer.
	Answer-unaware	**Score 1:** Generated question is not answerable by the context. **Score 2:** Generated question is answerable by the context, but does not have the answer as the the target answer (target answer is available since we first extract the target answers). **Score 3:** Gennerated question is answerable by the context and its answer is target answer (target answer is available since we first extract the target answers).

Figure 4: Human Rating System

Semantic-based Pre-training for Dialogue Understanding

Xuefeng Bai[*♠♢], **Linfeng Song**[♣], **Yue Zhang**[♠♢]
[♠] School of Engineering, Westlake University, China
[♣] Tencent AI Lab, Bellevue, WA, USA
[♢] Institute of Advanced Technology, Westlake Institute for Advanced Study, China

Abstract

Pre-trained language models have made great progress on dialogue tasks. However, these models are typically trained on surface dialogue text, thus are proven to be weak in understanding the main semantic meaning of a dialogue context. We investigate Abstract Meaning Representation (AMR) as explicit semantic knowledge for pre-training models to capture the core semantic information in dialogues during pre-training. In particular, we propose a semantic-based pre-training framework that extends the standard pre-training framework (Devlin et al., 2019) by three tasks for learning 1) core semantic units, 2) semantic relations and 3) the overall semantic representation according to AMR graphs. Experiments on the understanding of both chit-chats and task-oriented dialogues show the superiority of our model. To our knowledge, we are the first to leverage a deep semantic representation for dialogue pre-training.

1 Introduction

Dialogue systems have attracted increasing attention from both academia and industry researches (Chen et al., 2017; Deriu et al., 2021; Gao et al., 2021a). The tasks can be commonly divided into two categories: task-oriented dialogue systems (Wen et al., 2017; Dinan et al., 2019; Mehri et al., 2020) and chit-chat dialogue systems (Ritter et al., 2011; Li et al., 2017; Yu et al., 2020; Cui et al., 2020; Chen et al., 2021, 2022; Song et al., 2022). The former aims to interact in the context of a specific task, while the latter chats with users without task and domain restrictions. Despite differences in goals, a common challenge for both tasks is understanding the semantic information conveyed in a dialogue history.

Recently, semantic representations from pre-trained language models have achieved remarkable success on a spectrum of dialogue tasks (Wen et al., 2015; Zhang et al., 2020; Wu et al., 2020; Gu et al., 2021; Zeng et al., 2021; Zhang and Zhao, 2021; Cui et al., 2021), where knowledge learned in pre-training over large-scale dialogue corpora can be transferred to downstream applications. Current pre-training techniques typically focus on the surface text. However, they do not explicitly consider deep semantic clues beyond text, which leads to some unexpected behavior, such as paying attention to meaningless words (Mudrakarta et al., 2018), and suffering from spurious feature associations (Kaushik et al., 2020) and adversarial attacks (Jia and Liang, 2017).

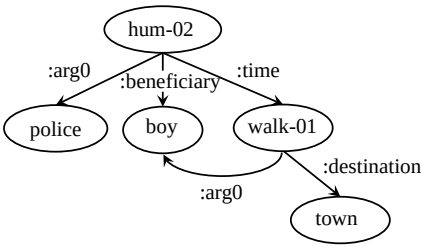

Figure 1: An AMR graph for sentence "*The police hummed to the boy as he walked to town.*"

Incorporating semantic information into dialogue systems has been shown to be helpful for many downstream tasks, such as dialogue intent prediction (Gupta et al., 2018), dialogue state tracking (Cheng et al., 2020), and dialogue relation extraction (Bai et al., 2021). These methods first parse dialogue turns into semantic structures, and then incorporate them as extra features into neural systems. However, they 1) only focus on domain-specific benchmark data, leaving the general potentiality of semantic structures unexploited; 2) require either human annotations or an external parser to obtain semantic structures, raising costs or/and causing error propagation for real applications.

We present **SARA**, a **S**emantic-graph-based

[*] Work done as an intern at Tencent AI Lab.

Proceedings of the 29th International Conference on Computational Linguistics, pages 592–607
October 12–17, 2022.

pre-tr**A**ining f**R**amework for di**A**logues, aiming to endow a pre-trained dialogue model with a stronger ability to infer semantic structures from conversations by using explicit semantic structures for more fine-grained supervisions. In particular, we exploit the abstract meaning representation (AMR; Banarescu et al. 2013), a fine-grained deep structure widely adopted in semantic parsing (Lyu and Titov, 2018; Zhang et al., 2019; Cai and Lam, 2020; Bevilacqua et al., 2021; Bai et al., 2022) and generation (Konstas et al., 2017; Song et al., 2018; Zhu et al., 2019; Bai et al., 2020; Ribeiro et al., 2021). As shown in Figure 1, AMR represents a sentence using a rooted directed graph, highlighting the core semantic units (e.g., "*police*", "*hum*", "*boy*") in a sentence and connecting them with semantic relations (e.g., "*:arg0*", "*:time*").

We explicitly leverage AMR graphs for pre-training our dialogue model. As shown in Figure 2, SARA consists of three pre-training sub-tasks: 1) semantic-based mask language modeling, which extends the standard mask language modeling task (Devlin et al., 2019) by paying more attention to core semantic units in a dialogue; 2) semantic relation prediction, which aims to learn semantic relations between words; 3) semantic agreement, which optimizes the overall similarity between a dialogue and its corresponding AMR graph. The SARA combines strengths of both powerful contextualized representation of pre-trained models and explicit semantic knowledge, while eliminating the requirement of an external semantic parser in downstream applications.

We choose BERT (Devlin et al., 2019) and RoBERTa (Liu et al., 2019) models as backbone, which are then continual pre-trained on a large-scale conversation dataset using our framework. Experiments show that our semantic-based framework gives better results than current pre-training methods that use much more training data, achieving new state-of-the-art results on both chit-chat understanding (dialogue relation extraction) and task-oriented dialogue understanding tasks (DialoGLUE benchmark). Our method also gives better results than previous semantic-base systems on downstream tasks, without using an external parser. Further analysis suggests that semantic information introduced by AMR can help our model to better understand semantically complex dialogues. To our knowledge, we are the first to leverage deep semantic representation for dialogue pre-training. Our code and the pre-trained models are available at https://github.com/goodbai-nlp/Sem-PLM.

2 Related Work

Pre-training for Dialogue. Inspired by the success of pre-trained language models in the general domain (Peters et al., 2018; Radford and Narasimhan, 2018; Devlin et al., 2019; Lewis et al., 2020), various pre-trained models have been proposed in the domain of dialogue. DialoGPT (Zhang et al., 2020) continual pre-trains a GPT-2 (Radford et al., 2019) model directly on Reddit comments data. ConvRT (Henderson et al., 2019) pre-trains a dual Transformer encoder for the response selection task. PLATO (Bao et al., 2020) introduces a latent variable-based model for dialogue response generation pre-training. TOD-BERT (Wu et al., 2020) pre-trains a Transformer encoder on task-oriented dialogue corpus for task-oriented dialogue applications. MPC-BERT (Gu et al., 2021) continues to pre-train a BERT model with self-supervised tasks based on the interactions among utterances and interlocutors. SPIDER (Zhang and Zhao, 2021) continues to pre-train a BERT model with auxiliary tasks to predict the utterance order and understand the sentence backbone. DialogLM (Zhong et al., 2022) pre-trains a generative Transformer encoder on long conversations with window-based pre-training tasks. Our work is similar in that we also pre-train a model on the dialogue corpora. However, unlike these previous studies, which focus on text level distributions, we additionally enhance the model with semantic structures.

Semantics for dialogue. Semantic knowledge has been used for both social chat and task-oriented dialogues systems. PEGASUS (Zue et al., 1994) transforms a sentence into a semantic frame which is then used for travel planing. Wirsching et al. (2012) design a dialogue system which performs database operations based on semantic features. Gupta et al. (2018) and Aghajanyan et al. (2020) integrate intents and slots into a semantic tree and solve intent classification and slot-filling tasks as semantic parsing. Cheng et al. (2020) represent task-oriented dialogue as a semantic graph to perform dialogue state tracking. A most related work is Bai et al. (2021), who build dialogue-level AMR graphs for both social chat understanding and dialogue response generation. Our work is similar

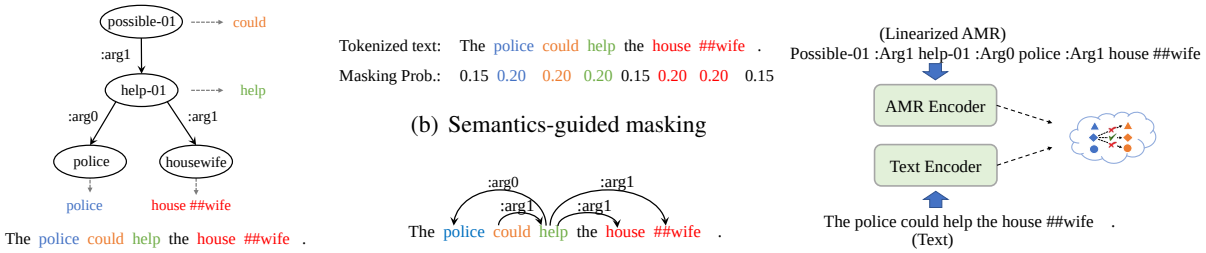

(a) Example AMR graph. (b) Semantics-guided masking (c) Semantic relation prediction (d) Semantic agreement

Figure 2: The semantic-based pre-training framework.

in showing the effect of semantic knowledge for improving dialogue understanding. However, different from them, we focus on enhancing the language model with semantic knowledge during pre-training, and our model does not require an external AMR parser in downstream applications.

3 Method

Figure 2 illustrates our semantic-based pre-training framework for dialogues. We take a pre-trained Transformer (Vaswani et al., 2017) encoder as the backbone, using AMR as explicit semantic knowledge to continuously pre-train the model on dialogues in a multitask setting. In particular, the following three semantic-aware tasks are designed:

- Semantics-based masking (Section 3.1).
- Semantic relation prediction (Section 3.2).
- Semantic agreement (Section 3.3).

The former two learn semantic knowledge from AMR nodes and AMR edges, respectively. The last task regularizes the overall representation of a dialogue using graph-level semantic features.

We follow Bai et al. (2021) and construct dialogue-level AMR graphs by 2 steps: 1) building utterance-level AMR graphs by independently transforming utterances into AMR using a pre-trained AMR parser. 2) connecting utterance-level AMR graphs with a root node, where edges are labeled with the corresponding speaker.

Formally, denote an input dialogue sequence[1] as $\boldsymbol{x} = [x_1, x_2, ..., x_n]$, where n is the number of tokens in the dialogue. The corresponding AMR is a directed acyclic graph $\mathcal{G} = \langle \mathcal{V}, \mathcal{E} \rangle$, where $\mathcal{V}$ denotes a set of nodes (i.e., AMR concepts) and $\mathcal{E}$ (i.e., AMR relations) denotes a set of labeled edges. An edge can be further represented by a triple $\langle v_i, r_{ij}, v_j \rangle$, meaning that the edge is from node v_i to v_j with label r_{ij}.

3.1 Task 1: Semantics-guided Masking

We first formally present the vanilla mask language modeling (MLM) setup, before introducing the semantic-guided masking strategy.

Vanilla MLM. Given a sequence of tokens $\boldsymbol{x}$, the standard masking strategy (Devlin et al., 2019) selects a set fraction of tokens positions (denoted as $\boldsymbol{m} = [m_1, m_2, ..., m_k]$) for masking independently at random, and use these "*selected*" tokens $\{x_i | i \in \boldsymbol{m}\}$ as supervisions to train a language model. Formally, denoting the masked text as $\tilde{\boldsymbol{x}}$, vanilla MLM optimizes the following training objective:

$$\ell_{vanilla\_mlm} = -\sum_{i \in \boldsymbol{m}} \log P(x_i | \tilde{\boldsymbol{x}}), \qquad (1)$$

where the conditional probability $P(x_i|\tilde{\boldsymbol{x}})$ is generated by an encoder model with a softmax layer.

Semantics-guided Masking. A salient limitation of vanilla MLM is that it treats all tokens equally, thus potentially wasting resources on tokens that provide little signal (e.g., punctuations and stop words). We introduce a semantic-guided masking strategy, encouraging model to give more attention on *semantic-aware* units, which are expected to have more influence on text understanding. As shown in Figure 2(b), our semantic-guided masking strategy gives a higher masking probability for tokens (e.g. "*police*", "*could*", "*help*") that contain important semantic information. Formally, we define a token as a *semantic-aware* unit when it is aligned with an AMR node, according to the AMR-to-text alignment $\mathcal{A}$[2] (An example is given in Figure 2(a)). Since pre-trained models typically use a vocabulary with sub-word units (Sennrich et al., 2016), for an alignment pair $\langle v_i, x_j \rangle$, we extend the alignment as $\left\langle v_i, \{x_j^1, x_j^2 ..., x_j^l\} \right\rangle$, where the AMR node v_i is aligned a set of all tokens $\{x_j^1, x_j^2 ..., x_j^l\}$

[1] Please refer to Appendix B for dialogue input format.

[2] $\mathcal{A}$ is a one-to-K mapping ($K \in [1, \ldots, n]$).

which are sub-words of word w_j. For example, in Figure 2, the AMR node "*housewife*" is aligned with sub-tokens "*house*" and "*##wife*".

Denoting $\boldsymbol{m'} = [m'_1, m'_2, ..., m'_k]$ as token indices selected by the proposed semantic-guided masking strategy, the training objective is:

$$\ell_{sem\_mlm} = -\sum_{i \in \boldsymbol{m'}} \log P(x_i | \tilde{\boldsymbol{x}}). \quad (2)$$

We follow RoBERTa (Liu et al., 2019) and use the dynamic masking, where we generate the masking pattern every step instead of performing masking during data preprocessing.

3.2 Task 2: Semantic Relation Prediction

The semantic relation prediction task is designed for learning the semantic relations between words. To this end, we project the edges of each input AMR graph onto the corresponding sentence according to their node-to-word alignments (as shown in Figure 2(c)), before training a predictor to generate the projected edges.

Relation Projection. Since AMR relations are defined on AMR nodes instead of words in the dialogue text, we use a node-to-word alignment $\mathcal{A}$ to project the AMR edges $\mathcal{E}$ onto text with following rules:

$$\hat{r}_{ij} = \begin{cases} r_{i'j'}, & \text{if } x_i \in \mathcal{A}(v_{i'}), x_j \in \mathcal{A}(v_{j'}), \\ \textit{None}, & \text{otherwise.} \end{cases} \quad (3)$$

The same strategy in Section 3.1 is used to deal with sub-word tokens.

Relation Prediction. We first use a Transformer encoder to generate contextualized word hidden states $\boldsymbol{h} = [h_1, h_2, ..., h_n]$. Based on that, a deep biaffine neural parser (Dozat and Manning, 2017) is used to predict the relations between words. To determine whether a directed edge (or *arc*) from x_i to x_j exists, the biaffine parser first uses two separate MLPs (denoted as $\texttt{MLP}^H$ and $\texttt{MLP}^D$) to obtain two lower-dimensional representation vectors for each position, then calculates scores via a biaffine operation:

$$\begin{aligned} r_i^H, r_j^D &= \texttt{MLP}^H(h_i), \texttt{MLP}^D(h_j), \\ s_{ij}^{arc} &= \begin{bmatrix} r_j^D \\ 1 \end{bmatrix}^T W^{arc} r_i^H, \\ P(y_{ij}^{arc} | \boldsymbol{x}) &= \text{softmax}_j(s_i^{arc}), \end{aligned} \quad (4)$$

where r_i^H is the representation vector of x_i as a head word, and r_j^D denotes the vector of x_j as a dependent word. $P(y_{ij}^{arc} | \boldsymbol{x})$ is the probability of the *arc* (i, j), and W^{arc} is a parameter matrix. To calculate the probability of assigning a label l to the *arc*(i, j), which is denoted as $P(y_{ijl}^{label} | \boldsymbol{x})$, the biaffine parser uses the same scorer as in Equation 4 but with different parameters for MLPs and biaffines.[3]

The training objective of relation prediction is:

$$\ell_{rel} = -\sum_{\langle x_i, \hat{r}_{ij}, x_j \rangle \in \mathcal{E}'} \log P(y_{ij}^{arc} | \boldsymbol{x}) P(y_{ij\hat{r}_{ij}}^{label} | \boldsymbol{x}), \quad (5)$$

where $\mathcal{E}'$ represents the *projected* AMR edges.

3.3 Task 3: Semantic Agreement

We encourage the model to learn the overall agreement of a dialogue and its corresponding AMR graph. As shown in Figure 2(d), we use an auxiliary network to encode the AMR, and maximize the similarity score between the hidden states of text and AMR. Following previous work (Konstas et al., 2017), we linearize AMR graphs into a sequence (refer to Figure 2(d) for an example) and use a pre-trained encoder to transform AMR into a set of hidden states.[4]

Formally, defining the linearized AMR graph as $\boldsymbol{g} = [g_1, g_2, ..., g_m]$, the vector representation of text and its corresponding AMR is calculated as:

$$\begin{aligned} h^{\text{text}} &= \texttt{Pooling}(\texttt{TextEnc}(\boldsymbol{x})), \\ h^{\text{amr}} &= \texttt{Pooling}(\texttt{TextEnc}(\boldsymbol{g})), \end{aligned} \quad (6)$$

where $\texttt{TextEnc}(\cdot)$ and $\texttt{TextEnc}(\cdot)$ are text encoder and AMR encoder, respectively. They are initialized with the same weights but updated separately during training. $\texttt{Pooling}(\cdot)$ is a function that reduces that sequence of vectors into one vector. Following BERT (Devlin et al., 2019), we feed the hidden state of the first input token into a MLP layer to get the "pooled" vector.

We use the cosine similarity as a distance scoring function and adopt the contrastive learning framework (Hadsell et al., 2006; Frosst et al., 2019; Gao et al., 2021b; Luo et al., 2022) to train our model, with the aim to pulling semantically close text-AMR pairs and pushing apart unpaired examples. In particular, for a given text $\boldsymbol{x}$, the *positive* example is its corresponding AMR graph $\boldsymbol{g}$, the *negative* examples are the AMR graphs of its

[3]The biaffine parameter for label scoring is a three dimensional tensor.

[4]We also tried a structure-aware encoder but without observing significant improvements.

Dataset	DialogRE	BANKING77	HWU64	CLINC150	REST8K	DSTC8	TOP	MULTIWOZ
train	5,997	8,622	8,954	15,000	7,244	5,023	31,279	56,774
dev	1,914	1,540	1,076	3,000	1,000	602	4,462	7,374
test	1,862	3,080	1,076	4,500	3,731	1,813	9,042	7,372

Table 1: Statistics of datasets.

neighbor dialogues in the corpus. Formally, let h_i^{text} and h_i^{amr} denote the representations of the ith $\langle text, AMR \rangle$ pair in the dataset, the training objective is:

$$\ell_{sim} = -\log \frac{\exp(\text{sim}(h_i^{\text{text}}, h_i^{\text{amr}})/\tau)}{\sum_{j \in \mathcal{N}(i)} \exp(\text{sim}(h_i^{\text{text}}, h_j^{\text{amr}})/\tau)}, \tag{7}$$

where $\text{sim}(\cdot, \cdot)$ denotes the cosine similarity, $\mathcal{N}(i)$ collects neighbor index of the ith example, and $\tau > 0$ denotes the temperature hyper-parameter.

3.4 Training

Our model is trained by optimizing the total loss of above 3 tasks:

$$\ell_{total} = \ell_{sem\_mlm} + \alpha \ell_{rel} + \beta \ell_{sim}, \tag{8}$$

where α and β are weighting hyper-parameters for ℓ_{rel} and ℓ_{sim}, respectively. To make the computational requirements feasible, we do not train our model from scratch, but rather continue training a model that has been pre-trained on textual inputs. Our framework is architecture-flexible and can be be applied to different models such as BERT, ROBERTA, and BART.

4 Experiments

We evaluate the effectiveness of our semantic pre-training model on 8 dialogue tasks and compare the results with the state-of-the-art pre-trained and semantic-enriched models.

4.1 Dataset

Pre-training Corpus. We continual pre-train our model on the Reddit (Henderson et al., 2019) corpus. After sampling and filtering (refer Appendix A), the dataset comprises 5,864,254 dialogue instances, in total 397 million words. We adopt the state-of-the-art AMRBART (Bai et al., 2022) parser [5] to transform the text into AMR graphs. To obtain the AMR-to-text alignment, we use the JAMR aligner[6] released by Flanigan et al. (2014).

[5] https://github.com/muyeby/AMRBART

[6] https://github.com/jflanigan/jamr

Dialogue task datasets. We evaluate our model on both chitchat and task-oriented understanding tasks. For chitchat, we focus on the dialogue relation extraction task which aims to predict the relationship between an given entity pair. We report results on both original (v1) and updated (v2) English version of **DialogRE** (Yu et al., 2020).

For task-oriented dialogue, we report results on the DialoGLUE (Mehri et al., 2020) benchmark, which consists of 7 different datasets spanning 4 different tasks: 1) intention prediction, including **BANKING77** (Casanueva et al., 2020), **CLINC150** (Larson et al., 2019) and **HWU64** (Liu et al., 2021); 2) slot filling, including **RESTAURANT8K** (Coope et al., 2020) and **DSTC8** (Rastogi et al., 2019); 3) semantic parsing, **TOP** (Gupta et al., 2018); and 4) dialogue state tracking, **MULTIWOZ2.1** (Eric et al., 2020).

Table 1 shows the statistics of above datasets.

4.2 Settings

Model Configuration. We take BERT-base and ROBERTA-base as our backbone model. For Pre-training, AdamW (Loshchilov and Hutter, 2019) is used as an optimizer, with an initial learning rate of 1×10^{-5}. We reduce the learning rate according to a linear scheduler. The batch size is 2048, and the maximum input sequence length is 512. For the hyper-parameters, we empirically set $\alpha = 0.1, \beta = 1.0, \tau = 1.0$ in our experiments. The pre-training of our model is carried out on 8 Nvidia Telsa V100 32G GPU for 5 epochs, taking about 2 days to reach convergence. For fine-tuning, we follow previous works to set hyper-parameters. More details can be found in Appendix C.

Metrics. We use macro F1 and macro $F1_c$ for dialogue relation extraction (DialogRE), following Yu et al. (2020). For intent prediction (BANKING77, CLINC150, HWU64), we report the accuracy. Macro F1 (Coope et al., 2020) is adopted for slot filling tasks (RESTAURANT8K, DSTC8). For TOP, we use exact-match, which measures how often the model generates the exact reference structure. For MULTIWOZ, we use the joint goal accuracy following Budzianowski et al. (2018).

Model	data-v1				data-v2			
	dev		test		dev		test	
	F1(δ)	F1$_c$(δ)	F1(δ)	F1$_c$(δ)	F1(δ)	F1$_c$(δ)	F1(δ)	F1$_c$(δ)
GDPNet	67.1 (1.0)	61.5 (0.8)	64.9 (1.1)	60.1 (0.9)	-	-	-	-
TUCORE-GCN	-	-	-	-	66.8 (0.7)	61.0 (0.5)	65.5 (0.4)	60.2 (0.6)
TSP	66.8 (0.9)	61.5 (1.0)	65.5 (0.7)	60.5 (0.8)	-	-	-	-
BERT	60.6 (1.2)	55.4 (0.9)	58.5 (2.0)	53.2 (1.6)	59.4 (0.7)	54.7 (0.8)	57.9 (1.0)	53.1 (0.7)
BERT$_s$	63.0 (1.5)	57.3 (1.2)	61.2 (0.9)	55.4 (0.9)	62.2 (1.3)	57.0 (1.0)	59.5 (2.1)	54.2 (1.4)
BERT$_c$	66.8 (0.9)	60.9 (1.0)	66.1 (1.1)	60.2 (0.8)	66.2 (0.9)	60.5 (1.1)	65.1 (0.8)	59.8 (1.2)
RoBERTa	68.0 (1.0)	60.3 (1.0)	66.0 (0.6)	59.6 (0.7)	67.6 (0.8)	61.0 (0.7)	65.8 (1.0)	59.6 (0.5)
SARA-BERT	68.1 (1.0)	62.1 (0.9)	67.5 (0.7)	61.4 (0.9)	68.0 (0.8)	62.1 (0.6)	67.3 (1.0)	61.3 (0.8)
SARA-RoBERTa	**69.3** (0.9)	**62.3** (0.8)	**68.1** (0.8)	**61.7** (1.0)	**69.5** (0.7)	**62.4** (0.5)	**67.8** (0.8)	**61.5** (0.7)

Table 2: Performance on DialogRE. We report the average and the standard deviation computed from 5 runs, best results are marked in **bold**.

4.3 Compared Models

For Dialogue relation extraction, we compare the proposed model with BERT-based models: **BERT** takes a pre-trained BERT as the dialogue encoder and predicts relation labels using the hidden state of the `[CLS]` token. **BERT$_s$** (Yu et al., 2020) enhances the speaker representation by marking speaker arguments with special tokens. **BERT$_c$** (Bai et al., 2021) concatenates hidden states of the `[CLS]` token and entity tokens for classification. For completeness, we also include recent methods which give the state-of-the-art results, such as **GDPNet** (Xue et al., 2020), **TUCORE-GCN** (Lee and Choi, 2021), **TSP** (Zhao et al., 2021) and **Hier** (Bai et al., 2021). We follow the implementation and hyper-parameters of BERT$_c$ to evaluate our model.

For DialoGLUE, the compared models include: **BERT** (Devlin et al., 2019) pre-trains a Transformer encoder on large-scale monotonic text. **USE** (Yang et al., 2020) pre-trains a dual Transformer encoder model on multilingual corpus using retrieval focused training tasks. **ConveRT (654M)** (Henderson et al., 2020) pre-trains a dual Transformer encoder on the full 2015-2019 Reddit data comprising 654M $\langle context, response \rangle$ training pairs using response selection task. **ConvBERT (700M)** (Mehri et al., 2020) fine-tunes a BERT model on 700M Reddit conversational data. We adopt the same implementation and hyper-parameters of ConvBERT (700M) to conduct experiments on DialoGLUE.

To verify the scalability of the proposed method, we also report results based on the RoBERTa model for all tasks. The model architectures for about tasks is given in Appendix D.

4.4 Main Results

Results on DialogRE. Table 2 lists the results of different systems on DialogRE. Among BERT-based models (i.e., BERT, BERT$_s$, BERT$_c$), BERT$_c$ reports the best results. Compared with BERT$_c$, SARA-BERT gives significantly ($p < 0.001$) better results on both datasets. In particular, SARA-BERT improves BERT$_c$ by 1.4 and 2.2 points in terms of F1 score on two test sets, respectively, indicating that our semantic pre-training framework is beneficial for dialogue relation extraction. The main reason can be that SARA improves the model capacity of understanding entities (which are core semantic units) and the semantic relations between them during pre-training stage.

SARA-BERT achieves better F1 scores than the other state-of-the-art methods. In addition, when using RoBERTa as the backbone, SARA gives consistent improvements. In particular, SARA-RoBERTa achieves 68.1 and 67.8 F1 scores on the test set of data-v1 and data-v2, respectively. To our best knowledge, these are the best-reported results based on RoBERTa-base.

Results on DialoGLUE. We report the results of different methods on the DialoGLUE benchmark in Table 3. Compared with BERT, SARA-BERT (6M) gives consistently better results on all 7 datasets, with an improvement of 1.1 point in average. In particular, SARA-BERT (6M) outperforms BERT by 2.1 and 3.0 points on HWU64 and MultiWOZ, respectively, showing that our SARA framework can benefit task-oriented dialogue systems.

Compared with the other state-of-the-art systems, SARA-BERT (6M) obtains better results than USE, because SARA-BERT (6M) is pre-trained on large-scale dialogue corpus. In addition, SARA-BERT (6M) gives highly competitive results than

Model	BANK	HWU64	CLINC150	REST8K	DSTC8	TOP	MULTIWOZ	Avg
USE	92.81	91.25	95.06	-	-	-	-	-
CONVERT (654M)	93.01	91.24	97.16	-	-	-	-	-
USE+CONVERT (654M)	93.36	**92.62**	**97.16**	-	-	-	-	-
CONVBERT (700M)	93.44	92.38	97.11	95.44	91.20	82.08	56.56	86.89
BERT	93.02	89.87	95.93	95.53	90.05	81.90	56.30	86.08
ROBERTA	93.16	91.30	96.09	96.27	90.78	81.80	54.95	86.28
SARA-BERT (6M)	93.47	92.01	96.24	95.92	91.57	82.05	**59.33**	87.23
SARA-ROBERTA (6M)	**93.64**	92.29	96.60	**96.74**	**92.02**	**82.78**	57.52	**87.37**

Table 3: Performance on DialoGLUE, best results are in **bold**. REST8K and BANK stands for RESTAURANT8K and BANKING77, respectively.

Model	DialogRE	DSTC8
ROBERTA	67.6	93.98
ROBERTA (6M)	68.2	94.17
SARA-ROBERTA (6M)	**69.5**	**95.24**
w/o sem_mlm	69.0	95.01
w/o rel_pred	68.6	94.63
w/o sem_agree	68.8	94.72

Table 4: Validation F1 of DialogRE and DSTC8.

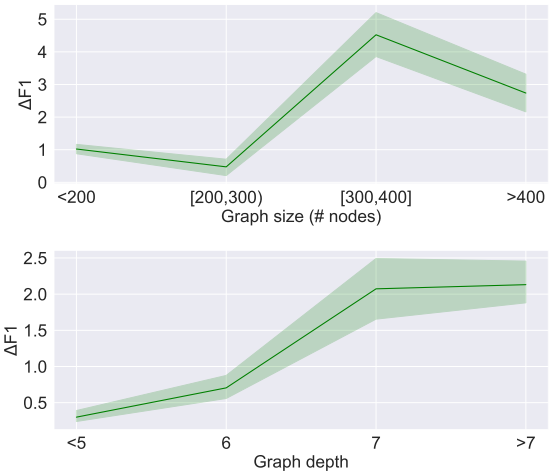

Figure 3: Performance improvement (ΔF1) over two aspects: (top) graph size and (bottom) graph depth.

CONVERT (654M), USE+CONVERT (654M) and CONVBERT (700M), using significantly fewer data (about 1% than others). This indicates that our semantic-based pre-training framework is more data-efficient. Finally, similar to SARA-BERT (6M), SARA-ROBERTA (6M) significantly ($p < 0.001$) outperforms ROBERTA, giving the best results on BANKING77, REST8K, DSTC8 and TOP.

5 Analysis

5.1 Ablation Study

We compare our full system with the following models: ROBERTA (6M) is continuously pre-trained on the exact same training corpus as our model using corresponding standard pre-training objectives; w/o sem_mlm, w/o rel_pred, and w/o sem_agree denote the models which are trained without the semantics-guided masking, semantic relation prediction, and semantic agreement task, respectively. Table 4 shows the F1 scores on the validation sets of DialogRE and DSTC8. First of all, using dialogue domain data (ROBERTA v.s. ROBERTA (6M)) for pre-training leads to improvements on both tasks. This meets previous observations (Gururangan et al., 2020; Mehri et al., 2020). Also, the semantic-based mask language modeling task (sem_mlm) gives an obvious improvement on DialogRE and a small one on DSTC8. The reason can be that DSTC8 has an average length of 8 tokens, making it easy to understand core semantic units in dialogues. In addition, the performance drops significantly without the relation prediction task (rel_pred), indicating that the rel_pred task is important for dialogue understanding. Furthermore, the semantic agreement task (sem_agree) is helpful for both datasets, showing that the AMR is beneficial to improve the overall semantic representation of dialogue. Finally, by combining dialogue domain data and all pre-training tasks, our final model achieves the best performance on both datasets.

5.2 Effect of Semantic-based Pre-training

To further understand the effectiveness of our semantic-based pre-training framework, we split the test set of DialogRE (v2) into different groups according to semantic complexity and report the performance improvement of SARA-ROBERTA over ROBERTA. In particular, two metrics are considered to measure the semantic complexity of a dialogue: 1) graph size (i.e., the number of nodes in the AMR graph) which records the number of

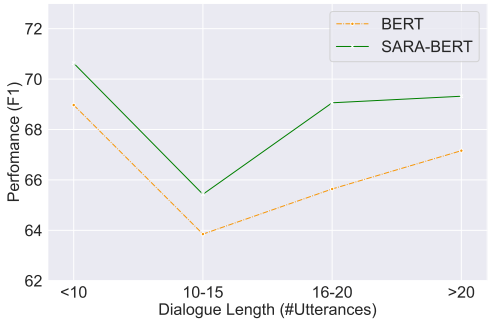

Figure 4: Test F1 on DialogRE (v2).

semantic units in the dialogue; 2) graph depth which is defined as the longest distance between the AMR node and root node. An AMR graph has a deeper depth means that its corresponding dialogue has more long-range dependencies.

As shown in the top sub-figure of Figure 3, SARA-RoBERTa gives consistent improvements over RoBERTa in different graph groups. In particular, the improvements are more considerable on graphs with more than 300 nodes, showing that SARA-RoBERTa has better capacity than RoBERTa in understanding dialogues which contain more semantic units. The reason can be that the semantic-based MLM task enhances the model ability to capture core semantic features, which helps in reducing negative impacts of meaningless tokens in dialogue text. With respect to graph depth, SARA-RoBERTa also outperforms RoBERTa on all groups, and larger improvements are observed on deeper graphs. It can be that the relation prediction task helps to establish semantic associations between non-neighbor words, thus benefiting long-range dependencies understanding.

We also compare the model performance in terms of dialogue length. In particular, we split the test set of DialogRE (v2) into 4 groups according to the utterance number of each dialogue, and compare the performance of RoBERTa and SARA-RoBERTa. As shown in Figure 4, SARA-RoBERTa consistently gives better results than RoBERTa on all groups. The performance gap is bigger when the input dialogue has more than 16 utterances. The reason is that SARA encourages the model to understand core semantics, which is helpful for learning long dialogues.

5.3 Impact of AMR Features

AMR is a deep semantic structure which consists of both backbone relations and fine-grained semantic relations. To study the contribution of such features, we simplify an AMR graph by masking the fine-grained semantic relations, resulting in a graph with frame arguments relations (e.g., *:arg0, :arg1, :arg2*). We use the simplified graph as explicit semantic knowledge for pre-training and compare it with the standard AMR graph under the same framework.

Model	DialogRE	DSTC8
RoBERTa	65.8	90.78
SARA-RoBERTa (full)	**67.8**	**92.02**
SARA-RoBERTa (simplified)	67.3	91.34

Table 5: F1 on the test set of DialogRE and DSTC8.

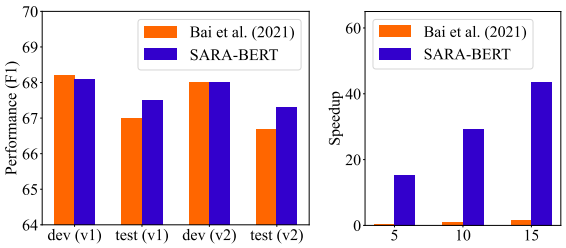

Figure 5: (a) Comparison of performance on DialogRE; (b) Comparison of inference speed regarding to dialogue length (measured by number of utterances).

Table 5 lists the performance of two systems. It can be observed that both *simplified* graphs and *full* AMR graphs lead to better performance. Compared with *simplified* graphs, using *full* AMR graph for pre-training leads to better results on both DialogRE and DSTC8, showing that the fine-grained semantic features can further improve the model performance.

5.4 Comparison with explicit AMR

Figure 5(a) compares the performance of our model with the method of Bai et al. (2021)[7] which use explicit AMR structures for dialogue applications. We report the F1 score on the test set of DialogRE. Compared with the system of Bai et al. (2021), our model gives comparable results on the validation set, and better results on the test set, without using an external AMR parser. This indicates that 1) our pre-training framework can efficiently transfer the learned semantic information to downstream tasks; 2) large-scale semantic-aware pre-training can give further improvement compared with using

[7]We choose the `Hier` model which has comparable parameters to our model.

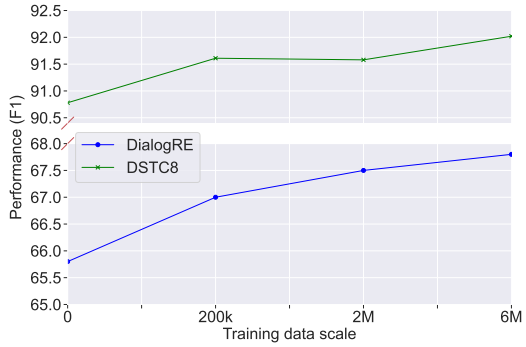

Figure 6: Impact of pre-training data.

semantic information in downstream tasks.

As shown in Figure 5(b), our system is significantly faster than the method of Bai et al. (2021) which relies on an external parser. As the dialogue length increases, the performance gap is more obvious. In particular, our system obtains about a 45 times speedup when the input dialogues have an average utterance number of 15.

5.5 Impact of Training Data Scale

Figure 6 shows the model performance regarding different scales of pre-training data. The performance on both DialogRE and DSTC8 datasets increases as the scale of training data grows bigger, with a margin of about 2.0 F1 score on DialogRE. Due to the limitation of computational resources, we do not conduct experiments on larger training corpus and models, and we leave this for a future work.

5.6 Case Study

Figure 7 shows an example conversation from DialogRE dataset. The baseline model (RoBERTa) is misled by sentences last three utterances (marked with underline) where *Speaker2* shows an negative emotions towards Rachel Green, and thus incorrectly predicting the relationship between two speakers as *negative_impression*. In contrast, our model (SARA-RoBERTa) predicts the correct relationship, suggesting that our semantic-based pre-training framework helps model to better understand the relationship between entity pairs and avoid focusing on spurious features.

Figure 8 presents a case of dialogue intent prediction. The baseline system pays much attention on word "*alarm*" while ignores other two core semantic units "*minutes*" and "*bake*", giving an incorrect prediction. Our system successfully predicts the gold intent, because AMR guides our model to discover the core semantic units in the dialogue text.

Speaker1: Wanna give me a hand?
Speaker2: Sure! Monica, I can't get over how great you look!
Speaker1: Oh umm, I meant to tell you, Ross is coming.
Speaker2: Ross is coming. Great! I love Ross!
Speaker1: Good, and Rachel Green too.
Speaker2: Oh.
Speaker1: Is there a problem?
Speaker2: Nope. Uh, it's okay. It's just uh, God I hate her.
Speaker1: What?
Speaker2: Yeah, I hate her. She was horrible to me in school.

Ground-Truth: per:positive_impression (Speaker1, Speaker2)

Baseline: per:negative_impression (Speaker1, Speaker2)
Ours: per:positive_impression (Speaker1, Speaker2)

Figure 7: An example of dialogue relation extraction.

How many minutes should I set an alarm for this bake?

Ground-Truth intent: cook_time

Baseline: alarm
Ours: cook_time

Figure 8: An example of dialogue intent prediction.

6 Conclusion

We investigated the abstract meaning representation as explicit semantic clues for dialogue pre-training, using a semantic-based pre-training framework. Experiments on two benchmarks show that the proposed framework is highly effective on both chit-chat understanding and task-oriented dialogue understanding. Our method gives the best results on multiple datasets.

Acknowledgments

Yue Zhang is the corresponding author. We would like to thank anonymous reviewers for their insightful comments. This work is supported by the Zhejiang Province Key Project 2022SDX-HDX0003 and the Tencent AI Lab Rhino-Bird Focused Research Program.

References

Armen Aghajanyan, Jean Maillard, Akshat Shrivastava, Keith Diedrick, Michael Haeger, Haoran Li, Yashar Mehdad, Veselin Stoyanov, Anuj Kumar, Mike Lewis, and Sonal Gupta. 2020. Conversational semantic parsing. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing*

(EMNLP), pages 5026–5035, Online. Association for Computational Linguistics.

Xuefeng Bai, Yulong Chen, Linfeng Song, and Yue Zhang. 2021. Semantic representation for dialogue modeling. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 4430–4445, Online. Association for Computational Linguistics.

Xuefeng Bai, Yulong Chen, and Yue Zhang. 2022. Graph pre-training for AMR parsing and generation. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 6001–6015, Dublin, Ireland. Association for Computational Linguistics.

Xuefeng Bai, Linfeng Song, and Yue Zhang. 2020. Online back-parsing for AMR-to-text generation. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1206–1219, Online. Association for Computational Linguistics.

Laura Banarescu, Claire Bonial, Shu Cai, Madalina Georgescu, Kira Griffitt, Ulf Hermjakob, Kevin Knight, Philipp Koehn, Martha Palmer, and Nathan Schneider. 2013. Abstract Meaning Representation for sembanking. In *Proceedings of the 7th Linguistic Annotation Workshop and Interoperability with Discourse*, pages 178–186, Sofia, Bulgaria. Association for Computational Linguistics.

Siqi Bao, Huang He, Fan Wang, Hua Wu, and Haifeng Wang. 2020. PLATO: Pre-trained dialogue generation model with discrete latent variable. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 85–96, Online. Association for Computational Linguistics.

Michele Bevilacqua, Rexhina Blloshmi, and Roberto Navigli. 2021. One spring to rule them both: Symmetric AMR semantic parsing and generation without a complex pipeline. *Proceedings of the AAAI Conference on Artificial Intelligence*, 35(14):12564–12573.

Paweł Budzianowski, Tsung-Hsien Wen, Bo-Hsiang Tseng, Iñigo Casanueva, Stefan Ultes, Osman Ramadan, and Milica Gašić. 2018. MultiWOZ - a large-scale multi-domain Wizard-of-Oz dataset for task-oriented dialogue modelling. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 5016–5026, Brussels, Belgium. Association for Computational Linguistics.

Deng Cai and Wai Lam. 2020. AMR parsing via graph-sequence iterative inference. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 1290–1301, Online. Association for Computational Linguistics.

Iñigo Casanueva, Tadas Temčinas, Daniela Gerz, Matthew Henderson, and Ivan Vulić. 2020. Efficient intent detection with dual sentence encoders. In *Proceedings of the 2nd Workshop on Natural Language Processing for Conversational AI*, pages 38–45, Online. Association for Computational Linguistics.

Hongshen Chen, Xiaorui Liu, Dawei Yin, and Jiliang Tang. 2017. A survey on dialogue systems: Recent advances and new frontiers. *SIGKDD Explor.*, 19(2):25–35.

Yulong Chen, Yang Liu, Liang Chen, and Yue Zhang. 2021. DialogSum: A real-life scenario dialogue summarization dataset. In *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*, pages 5062–5074, Online. Association for Computational Linguistics.

Yulong Chen, Ming Zhong, Xuefeng Bai, Naihao Deng, Jing Li, Xianchao Zhu, and Yue Zhang. 2022. The cross-lingual conversation summarization challenge. *CoRR*, abs/2205.00379.

Jianpeng Cheng, Devang Agrawal, Héctor Martínez Alonso, Shruti Bhargava, Joris Driesen, Federico Flego, Dain Kaplan, Dimitri Kartsaklis, Lin Li, Dhivya Piraviperumal, Jason D. Williams, Hong Yu, Diarmuid Ó Séaghdha, and Anders Johannsen. 2020. Conversational semantic parsing for dialog state tracking. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 8107–8117, Online. Association for Computational Linguistics.

Samuel Coope, Tyler Farghly, Daniela Gerz, Ivan Vulić, and Matthew Henderson. 2020. Span-ConveRT: Few-shot span extraction for dialog with pretrained conversational representations. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 107–121, Online. Association for Computational Linguistics.

Leyang Cui, Yu Wu, Shujie Liu, and Yue Zhang. 2021. Knowledge enhanced fine-tuning for better handling unseen entities in dialogue generation. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 2328–2337, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Leyang Cui, Yu Wu, Shujie Liu, Yue Zhang, and Ming Zhou. 2020. MuTual: A dataset for multi-turn dialogue reasoning. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 1406–1416, Online. Association for Computational Linguistics.

Jan Deriu, Alvaro Rodrigo, Arantxa Otegi, Guillermo Echegoyen, Sophie Rosset, Eneko Agirre, and Mark Cieliebak. 2021. Survey on evaluation methods for dialogue systems. *Artificial Intelligence Review*, 54(1):755–810.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, Minneapolis, Minnesota. Association for Computational Linguistics.

Emily Dinan, Stephen Roller, Kurt Shuster, Angela Fan, Michael Auli, and Jason Weston. 2019. Wizard of wikipedia: Knowledge-powered conversational agents. In *7th International Conference on Learning Representations, ICLR 2019, New Orleans, LA, USA, May 6-9, 2019*. OpenReview.net.

Timothy Dozat and Christopher D. Manning. 2017. Deep biaffine attention for neural dependency parsing. In *5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Conference Track Proceedings*. OpenReview.net.

Mihail Eric, Rahul Goel, Shachi Paul, Abhishek Sethi, Sanchit Agarwal, Shuyang Gao, Adarsh Kumar, Anuj Goyal, Peter Ku, and Dilek Hakkani-Tur. 2020. MultiWOZ 2.1: A consolidated multi-domain dialogue dataset with state corrections and state tracking baselines. In *Proceedings of the 12th Language Resources and Evaluation Conference*, pages 422–428, Marseille, France. European Language Resources Association.

Jeffrey Flanigan, Sam Thomson, Jaime Carbonell, Chris Dyer, and Noah A. Smith. 2014. A discriminative graph-based parser for the Abstract Meaning Representation. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1426–1436, Baltimore, Maryland. Association for Computational Linguistics.

Nicholas Frosst, Nicolas Papernot, and Geoffrey E. Hinton. 2019. Analyzing and improving representations with the soft nearest neighbor loss. In *Proceedings of the 36th International Conference on Machine Learning, ICML 2019, 9-15 June 2019, Long Beach, California, USA*, volume 97 of *Proceedings of Machine Learning Research*, pages 2012–2020. PMLR.

Chongming Gao, Wenqiang Lei, Xiangnan He, Maarten de Rijke, and Tat-Seng Chua. 2021a. Advances and challenges in conversational recommender systems: A survey. *AI Open*, 2:100–126.

Tianyu Gao, Xingcheng Yao, and Danqi Chen. 2021b. Simcse: Simple contrastive learning of sentence embeddings. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing, EMNLP 2021, Virtual Event / Punta Cana, Dominican Republic, 7-11 November, 2021*, pages 6894–6910. Association for Computational Linguistics.

Jia-Chen Gu, Chongyang Tao, Zhen-Hua Ling, Can Xu, Xiubo Geng, and Daxin Jiang. 2021. MPC-BERT: A pre-trained language model for multi-party conversation understanding. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing, ACL/IJCNLP 2021, (Volume 1: Long Papers), Virtual Event, August 1-6, 2021*, pages 3682–3692. Association for Computational Linguistics.

Sonal Gupta, Rushin Shah, Mrinal Mohit, Anuj Kumar, and Mike Lewis. 2018. Semantic parsing for task oriented dialog using hierarchical representations. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 2787–2792, Brussels, Belgium. Association for Computational Linguistics.

Suchin Gururangan, Ana Marasović, Swabha Swayamdipta, Kyle Lo, Iz Beltagy, Doug Downey, and Noah A. Smith. 2020. Don't stop pretraining: Adapt language models to domains and tasks. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 8342–8360, Online. Association for Computational Linguistics.

R. Hadsell, S. Chopra, and Y. LeCun. 2006. Dimensionality reduction by learning an invariant mapping. In *2006 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'06)*, volume 2, pages 1735–1742.

Michael Heck, Carel van Niekerk, Nurul Lubis, Christian Geishauser, Hsien-Chin Lin, Marco Moresi, and Milica Gasic. 2020. Trippy: A triple copy strategy for value independent neural dialog state tracking. In *Proceedings of the 21th Annual Meeting of the Special Interest Group on Discourse and Dialogue, SIGdial 2020, 1st virtual meeting, July 1-3, 2020*, pages 35–44. Association for Computational Linguistics.

Matthew Henderson, Paweł Budzianowski, Iñigo Casanueva, Sam Coope, Daniela Gerz, Girish Kumar, Nikola Mrkšić, Georgios Spithourakis, Pei-Hao Su, Ivan Vulić, and Tsung-Hsien Wen. 2019. A repository of conversational datasets. In *Proceedings of the First Workshop on NLP for Conversational AI*, pages 1–10, Florence, Italy. Association for Computational Linguistics.

Matthew Henderson, Iñigo Casanueva, Nikola Mrksic, Pei-Hao Su, Tsung-Hsien Wen, and Ivan Vulic. 2020. Convert: Efficient and accurate conversational representations from transformers. In *Findings of the Association for Computational Linguistics: EMNLP 2020, Online Event, 16-20 November 2020*, volume EMNLP 2020 of *Findings of ACL*, pages 2161–2174. Association for Computational Linguistics.

Robin Jia and Percy Liang. 2017. Adversarial examples for evaluating reading comprehension systems. In *Proceedings of the 2017 Conference on Empirical*

Methods in Natural Language Processing, EMNLP 2017, Copenhagen, Denmark, September 9-11, 2017, pages 2021–2031. Association for Computational Linguistics.

Divyansh Kaushik, Eduard H. Hovy, and Zachary Chase Lipton. 2020. Learning the difference that makes A difference with counterfactually-augmented data. In *8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, April 26-30, 2020*. OpenReview.net.

Ioannis Konstas, Srinivasan Iyer, Mark Yatskar, Yejin Choi, and Luke Zettlemoyer. 2017. Neural AMR: Sequence-to-sequence models for parsing and generation. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 146–157, Vancouver, Canada. Association for Computational Linguistics.

Stefan Larson, Anish Mahendran, Joseph J. Peper, Christopher Clarke, Andrew Lee, Parker Hill, Jonathan K. Kummerfeld, Kevin Leach, Michael A. Laurenzano, Lingjia Tang, and Jason Mars. 2019. An evaluation dataset for intent classification and out-of-scope prediction. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 1311–1316, Hong Kong, China. Association for Computational Linguistics.

Bongseok Lee and Yong Suk Choi. 2021. Graph based network with contextualized representations of turns in dialogue. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 443–455, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Veselin Stoyanov, and Luke Zettlemoyer. 2020. BART: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7871–7880, Online. Association for Computational Linguistics.

Yanran Li, Hui Su, Xiaoyu Shen, Wenjie Li, Ziqiang Cao, and Shuzi Niu. 2017. DailyDialog: A manually labelled multi-turn dialogue dataset. In *Proceedings of the Eighth International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 986–995, Taipei, Taiwan. Asian Federation of Natural Language Processing.

Xingkun Liu, Arash Eshghi, Pawel Swietojanski, and Verena Rieser. 2021. Benchmarking natural language understanding services for building conversational agents. In *Increasing Naturalness and Flexibility in Spoken Dialogue Interaction*, pages 165–183. Springer.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Roberta: A robustly optimized BERT pretraining approach. *CoRR*, abs/1907.11692.

Ilya Loshchilov and Frank Hutter. 2019. Decoupled weight decay regularization. In *International Conference on Learning Representations*.

Yun Luo, Fang Guo, Zihan Liu, and Yue Zhang. 2022. Mere contrastive learning for cross-domain sentiment analysis. *CoRR*, abs/2208.08678.

Chunchuan Lyu and Ivan Titov. 2018. AMR parsing as graph prediction with latent alignment. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 397–407, Melbourne, Australia. Association for Computational Linguistics.

Shikib Mehri, Mihail Eric, and Dilek Hakkani-Tur. 2020. Dialoglue: A natural language understanding benchmark for task-oriented dialogue. *arXiv preprint arXiv:2009.13570*.

Pramod Kaushik Mudrakarta, Ankur Taly, Mukund Sundararajan, and Kedar Dhamdhere. 2018. Did the model understand the question? In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1896–1906, Melbourne, Australia. Association for Computational Linguistics.

Matthew E. Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer. 2018. Deep contextualized word representations. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 2227–2237, New Orleans, Louisiana. Association for Computational Linguistics.

Alec Radford and Karthik Narasimhan. 2018. Improving language understanding by generative pre-training.

Alec Radford, Jeff Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. 2019. Language models are unsupervised multitask learners.

Pushpendre Rastogi, Arpit Gupta, Tongfei Chen, and Mathias Lambert. 2019. Scaling multi-domain dialogue state tracking via query reformulation. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 2 (Industry Papers)*, pages 97–105, Minneapolis, Minnesota. Association for Computational Linguistics.

Leonardo F. R. Ribeiro, Yue Zhang, and Iryna Gurevych. 2021. Structural adapters in pretrained language models for AMR-to-Text generation. In *Proceedings of the 2021 Conference on Empirical*

Methods in Natural Language Processing, pages 4269–4282, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Alan Ritter, Colin Cherry, and William B. Dolan. 2011. Data-driven response generation in social media. In *Proceedings of the 2011 Conference on Empirical Methods in Natural Language Processing*, pages 583–593, Edinburgh, Scotland, UK. Association for Computational Linguistics.

Rico Sennrich, Barry Haddow, and Alexandra Birch. 2016. Neural machine translation of rare words with subword units. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1715–1725, Berlin, Germany. Association for Computational Linguistics.

Linfeng Song, Chunlei Xin, Shaopeng Lai, Ante Wang, Jinsong Su, and Kun Xu. 2022. Casa: Conversational aspect sentiment analysis for dialogue understanding. *J. Artif. Int. Res.*, 73.

Linfeng Song, Yue Zhang, Zhiguo Wang, and Daniel Gildea. 2018. A graph-to-sequence model for AMR-to-text generation. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1616–1626, Melbourne, Australia. Association for Computational Linguistics.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, December 4-9, 2017, Long Beach, CA, USA*, pages 5998–6008.

Tsung-Hsien Wen, Milica Gašić, Nikola Mrkšić, Pei-Hao Su, David Vandyke, and Steve Young. 2015. Semantically conditioned LSTM-based natural language generation for spoken dialogue systems. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 1711–1721, Lisbon, Portugal. Association for Computational Linguistics.

Tsung-Hsien Wen, David Vandyke, Nikola Mrkšić, Milica Gašić, Lina M. Rojas-Barahona, Pei-Hao Su, Stefan Ultes, and Steve Young. 2017. A network-based end-to-end trainable task-oriented dialogue system. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 1, Long Papers*, pages 438–449, Valencia, Spain. Association for Computational Linguistics.

Günther Wirsching, Markus Huber, Christian Kölbl, Robert Lorenz, and Ronald Römer. 2012. Semantic dialogue modeling. In Anna Esposito, Antonietta M. Esposito, Alessandro Vinciarelli, Rüdiger Hoffmann, and Vincent C. Müller, editors, *Cognitive behavioural systems: COST 2102 International Training School, Dresden, Germany, February 21-26, 2011*, volume 7403.

Chien-Sheng Wu, Steven C.H. Hoi, Richard Socher, and Caiming Xiong. 2020. TOD-BERT: Pre-trained natural language understanding for task-oriented dialogue. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 917–929, Online. Association for Computational Linguistics.

Fuzhao Xue, Aixin Sun, Hao Zhang, and Eng Siong Chng. 2020. Gdpnet: Refining latent multi-view graph for relation extraction. *CoRR*, abs/2012.06780.

Yinfei Yang, Daniel Cer, Amin Ahmad, Mandy Guo, Jax Law, Noah Constant, Gustavo Hernández Ábrego, Steve Yuan, Chris Tar, Yun-Hsuan Sung, Brian Strope, and Ray Kurzweil. 2020. Multilingual universal sentence encoder for semantic retrieval. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics: System Demonstrations, ACL 2020, Online, July 5-10, 2020*, pages 87–94. Association for Computational Linguistics.

Dian Yu, Kai Sun, Claire Cardie, and Dong Yu. 2020. Dialogue-based relation extraction. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 4927–4940, Online. Association for Computational Linguistics.

Jiali Zeng, Yongjing Yin, Yang Liu, Yubin Ge, and Jinsong Su. 2021. Domain adaptive meta-learning for dialogue state tracking. *IEEE ACM Trans. Audio Speech Lang. Process.*, 29:2493–2501.

Sheng Zhang, Xutai Ma, Kevin Duh, and Benjamin Van Durme. 2019. Broad-coverage semantic parsing as transduction. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 3786–3798, Hong Kong, China. Association for Computational Linguistics.

Yizhe Zhang, Siqi Sun, Michel Galley, Yen-Chun Chen, Chris Brockett, Xiang Gao, Jianfeng Gao, Jingjing Liu, and Bill Dolan. 2020. DIALOGPT : Large-scale generative pre-training for conversational response generation. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics: System Demonstrations*, pages 270–278, Online. Association for Computational Linguistics.

Zhuosheng Zhang and Hai Zhao. 2021. Structural pre-training for dialogue comprehension. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 5134–5145, Online. Association for Computational Linguistics.

Tianyang Zhao, Zhao Yan, Yunbo Cao, and Zhoujun Li. 2021. Enhancing dialogue-based relation

extraction by speaker and trigger words prediction. In *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*, pages 4580–4585, Online. Association for Computational Linguistics.

Ming Zhong, Yang Liu, Yichong Xu, Chenguang Zhu, and Michael Zeng. 2022. Dialoglm: Pre-trained model for long dialogue understanding and summarization. In *Thirty-Sixth AAAI Conference on Artificial Intelligence, AAAI*, pages 11765–11773. AAAI Press.

Jie Zhu, Junhui Li, Muhua Zhu, Longhua Qian, Min Zhang, and Guodong Zhou. 2019. Modeling graph structure in transformer for better AMR-to-text generation. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 5459–5468, Hong Kong, China. Association for Computational Linguistics.

Victor Zue, Stephanie Seneff, Joseph Polifroni, Michael Phillips, Christine Pao, David Goddeau, James Glass, and Eric Brill. 1994. PEGASUS: A spoken language interface for on-line air travel planning. In *Human Language Technology: Proceedings of a Workshop held at Plainsboro, New Jersey, March 8-11, 1994*.

Param. Name	Value
Batch Size	2048
Optimizer	AdamW
Learning Rate (lr)	1e-5
Lr Scheduler	linear
Warmup Step	0
Max Training Epoch	5
Semantic Masking Prob.	0.2
Extended Vocabulary Size	30,774
Max Length (dialogue)	256
Max Length (AMR)	512
Mix Precision	fp16
Parameters (Pre-training)	219M
Parameters (downstream tasks)	110M
Training Time	about 45h

Table 6: Hyper-parameters of our models.

Appendix

A Data Pre-processing

For pre-training, we randomly sample 10 million dialogue from Reddit (Henderson et al., 2019) corpus and filter the data by removing the instances where

- dialogue contains special markers;
- dialogue contains more than 10 non-English tokens;
- dialogue is longer than 150 words;
- dialogue has more than 15 turns.

We also replace the URLs in dialogues with a special token <*url*>.

B Model Input Format

Take BERT-based model as an example, given a dialogue $\boldsymbol{x}$ which consists of n utterances, we concatenate all utterances as a single consecutive token sequence with special tokens separating them: $\boldsymbol{x} = \{$[CLS] [Utter$_1$] Speaker$_1$ U_1 [Utter$_2$] Speaker$_2$ U_2 ...[Utter$_n$] Speaker$_n$ U_n[SEP]$\}$, where U_1, U_2, U_n are utterance sequences. [CLS] and [SEP] mark the start and end of the dialogue. [Utter$_1$], [Utter$_2$], and [Utter$_n$] mark the utterance numbers. Speaker$_1$ denotes the speaker of the first utterance. For RoBERTa, we use <s> and </s> to surround the dialogue sequences.

C Model Hyper-Parameters

Table 6 lists all model hyper-parameters used for our experiments. The proposed model is implemented based on *Pytorch* and *Huggingface Transformers*[8]. Our source code and pre-trained models is released at https://github.com/goodbai-nlp/Sem-PLM.

D Architecture for Downstream Tasks

For all downstream dialogue understanding tasks, we use the pre-trained dialogue model as a dialogue encoder and make prediction based on the encoded hidden states. Taking the BERT-based model as an example, the model architecture of downstream task are:

Dialogue Relation Extraction: We concatenate the hidden states of two entities (denoted as e_1 and e_2) as well as the pooled representation of the [CLS] token into a linear classifier to predict the relation label as:

$$y = \texttt{MLP}_c([\texttt{pool}(h^{\texttt{[CLS]}}); vec(e_1); vec(e_2)]), \tag{9}$$

where $\texttt{MLP}_c$ is a linear classifier, and $vec(\cdot)$ selects the encoded representation of the input token. $\texttt{pool}(h^{\texttt{[CLS]}})$ passes the hidden state of the [CLS] token through a linear layer.

Intent Prediction: We solve the task as a sequence classification problem, by feeding the pooled hidden state of [CLS] token into a linear classifier to predict the relation label as:

$$y = \texttt{MLP}_c(\texttt{pool}(h^{\texttt{[CLS]}})). \tag{10}$$

Slot Filling: We represent the problem as IOB tagging, by feeding all hidden state of the input dialogue (denoted by H) into a linear classifier and predict the relation label as:

$$Y = \texttt{MLP}_c(H), \tag{11}$$

where H denotes the output hidden states, and Y is the output tag sequence.

Semantic Parsing: We solve the problem as joint sequence classification and sequence labeling task. Specifically, we predict the intent and slots label as:

$$\begin{aligned} y_{intent} &= \texttt{MLP}_{intent}(\texttt{pool}(h^{\texttt{[CLS]}})), \\ Y_{slot} &= \texttt{MLP}_{slot}(H), \end{aligned} \tag{12}$$

[8]https://github.com/huggingface/transformers

where H denotes the output hidden states, and Y is the output tag sequence.

Dialogue State Tracking: We follow the TripPy (Heck et al., 2020) framework make prediction, which uses BERT model as encoder and combines BERT with a triple copy strategy to perform state tracking. Please refer the original paper for more details.

Distribution Calibration for Out-of-Domain Detection with Bayesian Approximation

Yanan Wu[1*], **Zhiyuan Zeng**[1*], **Keqing He**[2*], **Yutao Mou**[1], **Pei Wang**[1], **Weiran Xu**[1*]
[1]Beijing University of Posts and Telecommunications, Beijing, China
[2]Meituan Group, Beijing, China
{yanan.wu,zengzhiyuan,myt,wangpei,xuweiran}@bupt.edu.cn
{hekeqing}@meituan.com

Abstract

Out-of-Domain (OOD) detection is a key component in a task-oriented dialog system, which aims to identify whether a query falls outside the predefined supported intent set. Previous softmax-based detection algorithms are proved to be overconfident for OOD samples. In this paper, we analyze overconfident OOD comes from distribution uncertainty due to the mismatch between the training and test distributions, which makes the model can't confidently make predictions thus probably causing abnormal softmax scores. We propose a Bayesian OOD detection framework to calibrate distribution uncertainty using Monte-Carlo Dropout. Our method is flexible and easily pluggable into existing softmax-based baselines and gains 33.33% OOD F1 improvements with increasing only 0.41% inference time compared to MSP. Further analyses show the effectiveness of Bayesian learning for OOD detection. [1]

1 Introduction

Detecting Out-of-Domain (OOD) or unknown intents from user queries is key for a task-oriented dialog system (Gnewuch et al., 2017; Akasaki and Kaji, 2017; Tulshan and Dhage, 2018; Shum et al., 2018; Zeng et al., 2021a,b; Wu et al., 2022). It aims to know when a user query falls outside their range of predefined supported intents to avoid performing wrong operations. Different from normal intent classification tasks, lack of labeled OOD examples leads to poor prior knowledge about these unknown intents, making it challenging to detect OOD samples.

A rich line of OOD intent detection algorithms has been developed recently, among which softmax-based methods demonstrated promise (Guo et al., 2017; Liang et al., 2018; Zheng et al., 2020). Softmax-based methods leverage softmax outputs extracted from an in-domain (IND) intent model and operate under the assumption that the test OOD samples get a lower likelihood probability than the ID data. For example, Maximum Softmax Probability (MSP) (Hendrycks and Gimpel, 2017) detects a test query as OOD if its max softmax probability is lower than a fixed threshold. However, all these models make a strong distributional assumption of the practical OOD probability being uniform, which has been proven wrong because neural networks can produce overconfidently high softmax scores even for OOD samples (Guo et al., 2017). Therefore, solving the overconfidence issue is still challenging for OOD detection.

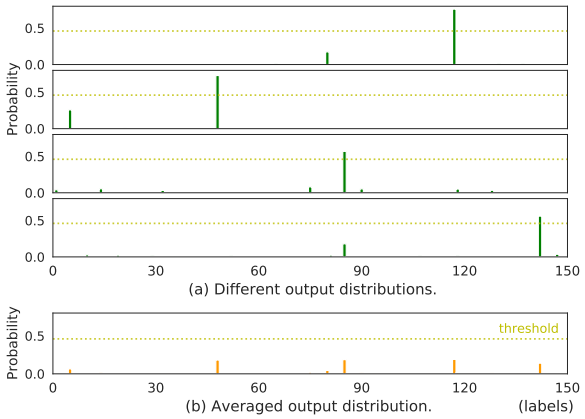

Figure 1: Multiple predicted probability distributions of an OOD sample under different random seeds. We train four identical models on the same data but only use different random seeds. Fig (a) displays each output distribution of an OOD input and Fig (b) shows the averaged output distribution.

In this paper, we study the overconfidence issue from the perspective of Bayesian learning (Gal and Ghahramani, 2016). Essentially, the reason for overconfidence is that a model cannot confidently make predictions on the input OOD utterances (unknow-unknow) due to the lack of prior knowledge of OOD data. In other words, even given the same input OOD intent, the predicted

*The first three authors contribute equally. Weiran Xu is the corresponding author.

[1]Our code is available at https://github.com/pris-nlp/COLING2022_Bayesian-for-OOD/.

Proceedings of the 29th International Conference on Computational Linguistics, pages 608–615
October 12–17, 2022.

probability distributions using different random seeds are completely different, uniform, sharp, or any distribution. Fig 1 show an example. We find models with different initialization seeds can output diverse distributions for OOD input, maybe cause overconfidence in several in-domain classes. But the averaged output is close to a uniform distribution. We also find models with different seeds are more robust to IND input and obtain consistent outputs (see Appendix C.2). Therefore, one direct way to solve the distribution uncertainty is to train multiple models independently and assemble their outputs for the final result. But this method is not applicable to practical scenarios for large training cost. In this paper, we propose a Bayesian OOD detection framework to calibrate distribution uncertainty. Specifically, we firstly train an in-domain intent classifier using IND data, then in the test stage, we perform multiple stochastic forward passes with a certain dropout rate (like 0.7) and average the output normalized logits as a final probability. Without increasing any new parameters, we calibrate distribution uncertainty by tending to expectation uniform distribution via Monte-Carlo Dropout (Gal and Ghahramani, 2016). Our method can be easily extended to existing softmax-based OOD detection methods and gain significant OOD improvements with only increasing little inference time compared to baselines, even outperform the state-of-the-art distance-based methods like LOF (Lin and Xu, 2019) and GDA (Xu et al., 2020). Our contributions are two-fold: (1) We analyze the intrinsic reason of overconfidence issue via distribution uncertainty and propose a Bayesian OOD detection framework to calibrate this uncertainty using Monte-Carlo Dropout. (2) We provide theoretical and empirical analysis to demonstrate the effectiveness of our Bayesian OOD method.

2 Method

2.1 Understanding OOD Detection

Problem Definition We refer to training data D as IND data. We aim to detect the input utterances x belonging to OOD and correctly classify the utterances belonging to IND utilizing a well-calibrated classifier trained only on finite IND data D.

The predictive uncertainty of a classification model $P(\upsilon|x, D)$ is commonly divided into *data uncertainty* (aleatoric), *distribution uncertainty* and *model uncertainty* (epistemic)(Kiureghian and Ditlevsen, 2009; Malinin and Gales, 2018):

$$P(\upsilon|x, D) = \iint \underbrace{P(\upsilon|\mu)}_{\text{data}} \underbrace{P(\mu|x, \theta)}_{\text{distribution}} \underbrace{P(\theta|D)}_{\text{model}} d\mu d\theta \tag{1}$$

The **model uncertainty** is described by the posterior distribution over model parameters θ, and it can be lowered by increasing the amount of data and simplifying the model complexity. The **data uncertainty** is described by the posterior distribution over classes, where υ is the predicted distribution of all possible in-domain intent classes for OOD detection. It arises from the natural complexity of the data, such as class overlap, label noise and homoscedastic noise. It is a property of the world, and cannot be changed. The **distribution uncertainty** is modeled with a distribution over distribution, where μ is the categorical distribution over simplex. It arises due to the mismatch between the training and test distributions. We give an example in Fig 2 which displays a distribution over distributions on a simplex (Dirichlet distribution (Malinin and Gales, 2018)) where each dot represents a softmax prediction distribution for a test OOD sample and all the dots denote a distribution over distributions. For an input utterance x, softmax-based detection algorithms like MSP assume that the distribution of OOD utterances (υ_{ood}) should be very close to the uniform distribution (the yellow dots in Fig 2) and the distribution of IND utterances (υ_{ind}) should be very close to one-hot distribution (e.g. Fig 2(a)-(d)). However, the practical OOD samples (green dots) exactly yield a sparse distribution over the simplex where each OOD sample may get a sharp softmax prediction distribution (like one-hot distribution) or a flat softmax prediction distribution (like uniform distribution). Due to the lack of prior knowledge of OOD data, the model cannot confidently make predictions on the input OOD utterances (unknow-unknow) which is the essential reason why the predicted probability distributions of the same OOD sample are completely different under different random seeds and even get very high max softmax scores (Fig 1). In other words, distribution uncertainty could lead to overconfidence in the prediction of OOD samples.

Therefore, how to alleviate the distribution uncertainty is the key to solving the overconfidence problem in OOD detection.

2.2 Bayesian Approximation

In order to alleviate the distribution uncertainty, we consider marginalizing out θ in Eq 1:

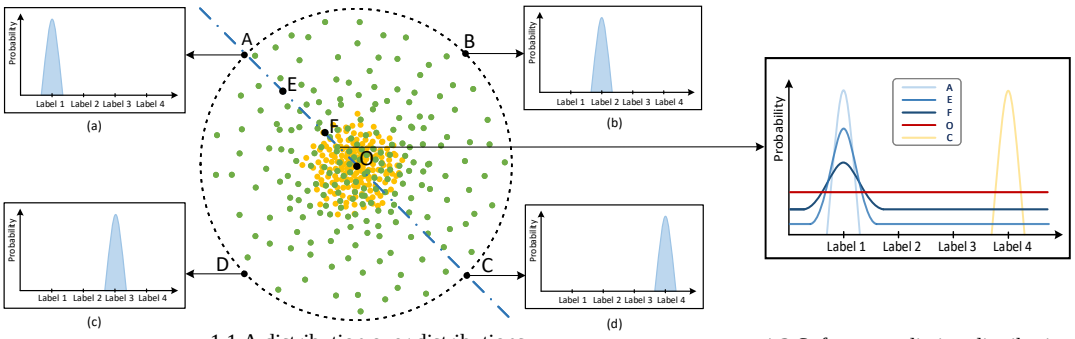

1.1 A distribution over distributions

1.2 Softmax prediction distribution

Figure 2: A distribution over distributions where each dot represents a softmax prediction distribution for a test OOD sample and all the dots denote a distribution over distributions. We also display softmax prediction distributions of several dots on the dot line in Fig 1.2.

$$P(\upsilon|x, D) = \int P(\upsilon|\mu)P(\mu|x, D)d\mu \quad (2)$$

This yields expected estimates of data and distributional uncertainty given model uncertainty. Marginalization is intractable in deep neural networks, thus we consider using $q(\omega)$ to approximate the intractable posterior though Monte-Carlo Sampling algorithm (Tsymbalov et al., 2020; Gal and Ghahramani, 2016):

$$P(\upsilon|x, D) = \int P(\upsilon|\omega)q(\omega)d\omega \quad (3)$$

where $\omega = \{W_i\}_{i=1}^{l}$ is the random variables for a model with l layers. We define $q(\omega)$ as:

$$W_i = M_i \cdot diag([\alpha_{i,j}]_{i=1}^{k_i}) \quad (4)$$

$$\alpha_{i,j} \sim Bernoulli(p_i) \quad (5)$$

Where p_i and M_i are the variational parameters. The binary variable $\alpha_{i,j}$ indicates whether unit j of the $i-1$ layer will be passed to the next layer. Specifically, we sample N sets of independent random vectors of realisations from the Bernoulli distribution $\{\alpha_1^n, ..., \alpha_l^n\}_{n=1}^{N}$ with $[\alpha_{i,j}]_{i=1}^{k_i}$ giving $\{W_1^n, ..., W_L^n\}_{n=1}^{N}$. Then we average the output:

$$E_{q(\upsilon|x,D)}(\upsilon) \approx \frac{1}{N}\sum_{n=1}^{N}\hat{v}(x, D, W_1^n, ..., W_l^n) \quad (6)$$

According to the Law of Large Numbers (Yao and Gao, 2016), when N is large enough, the predicted distribution will converge in expected uniform distribution. That is, we can calibrate the practical sparse OOD distribution (green dots) over the simplex into ideal dense OOD distribution (yellow dots) by Bayesian approximation to mitigate the overconfidence issue, which is verified in the following empirical experiments.

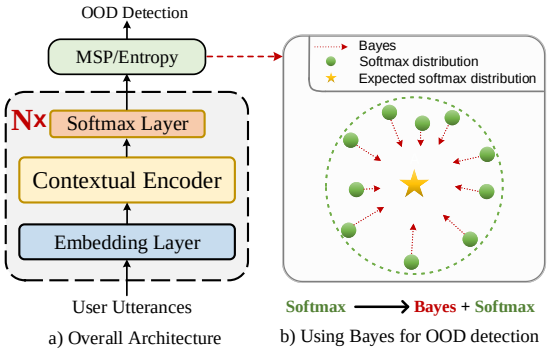

a) Overall Architecture

b) Using Bayes for OOD detection

Figure 3: The overall architecture of our method.

2.3 OOD Detection with Bayesian Learning

Fig 3(a) shows the overall architecture of our proposed OOD detection model. The part in the dashed box is a well-trained feature extractor based on Bi-LSTM (Hochreiter and Schmidhuber, 1997) or BERT (Devlin et al., 2019). It is trained on labeled in-domain data using cross-entropy loss. Fig 3(b) shows the Bayesian approximation process for distribution calibration. We adopt Monte-Carlo Dropout and average the output normalized logits from multiple stochastic forward passes: $\bar{\boldsymbol{v}} = \frac{1}{N}\sum_{i=1}^{N}\boldsymbol{v_i}$. In this way, we calibrate the softmax distribution to the expected distribution, which close to a uniform distribution. Then, we apply two softmax-based metrics for OOD detection, which is $m_{MSP} = \max(\bar{\boldsymbol{v}})$ and $m_{Entropy} = -\sum_{i=1}^{c}\bar{v}_i \log \bar{v}_i$. We further apply a empirical threshold to distinguish IND and OOD data.

3 Experiments

3.1 Datasets

We perform experiments on two public benchmark OOD datasets[2], CLINC-Full and CLINC-Imbal (Larson et al., 2019). We show the detailed statistic

[2]https://github.com/clinc/oos-eval

Model		CLINC-Full				CLINC-Imbal			
		OOD		IND		OOD		IND	
		F1	Recall	F1	ACC	F1	Recall	F1	ACC
LSTM	LOF (Lin and Xu, 2019)	59.28	58.32	86.08	85.87	55.37	51.03	80.51	82.79
	GDA (Xu et al., 2020)	65.79	64.14	87.90	86.83	61.38	63.80	85.35	84.20
	MSP (Hendrycks and Gimpel, 2017)	50.13	45.60	87.73	87.25	44.93	41.10	84.96	84.16
	MSP+Bayes.(ours)	70.05	68.38	88.91	88.57	61.70	57.50	85.92	85.65
	Entropy (Zheng et al., 2020)	68.05	67.96	88.97	88.68	64.45	63.80	86.07	85.71
	Entropy+Bayes.(ours)	**72.02**	**71.70**	89.10	88.73	**68.32**	**67.61**	86.34	86.11
BERT	MSP	52.79	50.50	87.81	87.46	48.76	46.70	85.87	85.65
	MSP+Bayes.(ours)	71.25	69.58	89.10	89.56	64.32	62.00	86.39	85.87
	Entropy	68.97	68.83	89.13	88.72	65.25	64.89	86.21	85.94
	Entropy+Bayes.(ours)	**72.85**	**72.42**	89.47	88.94	**69.11**	**68.49**	86.74	86.42

Table 1: Performance comparison between our method and baselines on CLINC-Full and CLINC-Imbal datasets (p <0.01). **Bayes.** represents our proposed Bayesian approximation via Monte-Carlo Dropout.

CLINC	Full	Imbal
Avg utterance length	9	9
Intents	150	150
Training set size	15100	10625
Training samples per class	100	25/50/75/100
Training OOD samples amount	100	100
Development set size	3100	3100
Development samples per class	20	20
Development OOD samples amount	100	100
Testing Set Size	5500	5500
Testing samples per class	30	30
Development OOD samples amount	1000	1000

Table 2: Statistics of the CLINC datasets.

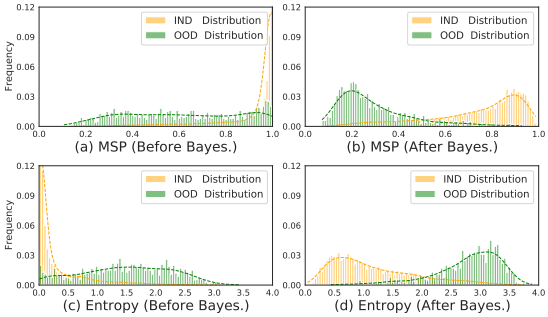

Figure 4: Effect of Bayesian approximation on MSP and Entropy confidence distributions of IND and OOD.

of these datasets in Table 2. They both contain 150 in-domain intents across 10 domains. The only difference is that, for CLINC-Imbal, there are either 25, 50, 75 or 100 training queries per in-scope intent, rather than 100. Note that all the datasets we used have a fixed set of labeled OOD data but we don't use it for training.

3.2 Metrics

We report both OOD metrics: Recall and F1-score(F1) and in-domain metrics: F1-score(F1) and Accuracy(ACC). Since we aim to improve the performance of detecting out-of-domain intents from user queries, OOD Recall and F1 are the main evaluation metrics in this paper.

3.3 Baselines

For detection algorithms, we use LOF, GDA, MSP and Entropy, none of them need OOD supervised training. For the feature extractor, we use LSTM and BERT. We provide a more comprehensive comparison and implementation details of these models in the Appendix.

3.4 Main Results

Table 1 shows our main results on two benchmarks. Our Bayesian method significantly outperforms softmax-based baselines including MSP and Entropy, even distance-based SOTA GDA on OOD metrics. Specifically, on CLINC-Full, Bayes improves 19.92% and 3.97% OOD F1 compared to MSP and Entropy using LSTM, which proves MSP suffers from severe overconfidence and our method helps calibrate OOD distribution. The performance gap between MSP and Entropy is because Entropy based on softmax output distribution can better capture distinguished information for OOD than MSP based on a single value of softmax distribution. We find similar improvements under the BERT setting on CLINIC-Imbal dataset.

4 Analysis

4.1 Effect of Bayesian approximation

Fig 4 shows the MSP and Entropy confidence distributions of IND and OOD test data using Bayesian to verify the effect of our method. Due to the over-confidence issue of OOD, we find IND and OOD curves overlap a lot in the original confidence scores. The overlapping part of Entropy is less, which confirms its better OOD detection performance. After calibration using Bayes, the overlap part of both methods is reduced, making it easier to distinguish between IND and OOD.

lgN	Statistical Indicators			
	OOD		IND	
	mean	median	mean	median
0	3.63	3.60	4.74	4.95
1	2.59	2.47	4.36	4.58
2	2.27	2.15	4.31	4.54
3	2.24	2.12	4.30	4.54

Table 3: KL-divergence between predicted distribution and uniform distribution on CLINC-Full. The smaller value is better for OOD. N is the number of dropout.

4.2 Analysis of Distribution Calibration

Table 3 shows the effect of Bayes on OOD Dirichlet Distribution. We calculate the KL-divergence between the predicted averaged softmax distribution and the uniform distribution of each test OOD sample and report the mean and median values on the whole test set. With the increase of sampling, we observe a larger drop on OOD mean and median KL values than INDs. It proves that Bayes can gradually calibrate the v_{ood} to a uniform distribution and thus make the sparse OOD Dirichlet distribution dense but not affect IND. Besides, we find N = 100 already achieves good performance to reduce inference cost. We provide an efficiency comparison in Section 4.3 and find 33.33% OOD F1 improvements only increase 0.41% time.

4.3 Analysis of Cost-effectiveness

We show the comparison between the time consumption and the corresponding performance improvement in Table 4 on CLINC-Full which has 15100 training data and 5500 test data. We find that when the number of samples N is 10, our method can improve the performance by 33.33% while only increasing the time by 0.41%, which proves that our proposed method is very cost-effective. Besides, we also find that more sampling times lead to more improvements, demonstrating that more accurate calibration significantly boosts OOD detection. In terms of time consumption and performance improvement, N = 100 is the most appropriate sampling parameter. When the sampling time is 1000, the cost-effectiveness is not as high as when $N = 10$. We consider that methods such as model distillation and pruning can reduce the time consumption, and we will leave it to future work. In general, we can choose the appropriate number of samples according to the computing resources.

4.4 Analysis of Parameters

Table 5 reports the OOD F1 under different dropout probability and sampling times. Within a range between 0.3 to 0.7, the larger dropout probability leads to better OOD detection performance. This is because OOD data is more vulnerable to feature loss and its averaged softmax prediction distribution tends to be more uniform. Besides, more sampling times lead to improvements, demonstrating that more accurate calibration significantly boosts OOD detection. We also find that the performance on p=0.7, N=10 is better than the performance on p=0.3, N=1000. This prompts us to choose a higher p (e.g. 0.7), which can effectively reduce the time consumption (1000->10). In addition, OOD F1 is not sensitive to excessive sampling times.

lgN	Time(s)	OOD F1	Increased	
			Time(%)	OOD F1(%)
0	240.00	50.13	-	-
1	240.98	66.84	↑ 0.41	↑ 33.33
2	252.41	70.05	↑ 5.17	↑ 39.74
3	388.36	70.82	↑ 61.82	↑ 41.27

Table 4: Time consumption and corresponding performance improvement of Bayesian approximation based MSP.

lgN	Dropout Probability					
	0.3	0.4	0.5	0.6	0.7	0.8
1	57.21	60.97	61.80	64.27	66.84	64.01
2	60.78	64.38	65.84	68.87	70.05	69.03
3	63.32	65.35	67.51	69.33	70.82	69.69

Table 5: Effect of Bayesian approximation with different parameters on OOD F1-score.

5 Conclusion

In this paper, we conduct an analysis of why previous softmax-based detection algorithms like MSP or Entropy suffer from the overconfidence issue. We find OOD samples exactly yield a sparse distribution over the simplex and evenly distribute over the whole space. Therefore, we propose a simple but strong Bayesian approximation method to calibrate OOD distribution. Experiments prove the effectiveness of our method. We hope to provide new guidance for future OOD detection work.

Acknowledgements

We thank all anonymous reviewers for their helpful comments and suggestions. This work was partially supported by MoE-CMCC "Artifical Intelligence" Project No. MCM20190701, National Key R&D Program of China No. 2019YFF0303300 and Subject II No. 2019YFF0303302, DOCOMO Beijing Communications Laboratories Co., Ltd.

References

Satoshi Akasaki and Nobuhiro Kaji. 2017. Chat detection in an intelligent assistant: Combining task-oriented and non-task-oriented spoken dialogue systems. *ArXiv*, abs/1705.00746.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, Minneapolis, Minnesota. Association for Computational Linguistics.

Yarin Gal and Zoubin Ghahramani. 2016. Dropout as a bayesian approximation: Representing model uncertainty in deep learning. In *Proceedings of The 33rd International Conference on Machine Learning*, volume 48 of *Proceedings of Machine Learning Research*, pages 1050–1059, New York, New York, USA. PMLR.

Ulrich Gnewuch, S. Morana, and A. Maedche. 2017. Towards designing cooperative and social conversational agents for customer service. In *International Conference on Information Systems*.

Chuan Guo, Geoff Pleiss, Yu Sun, and Kilian Q Weinberger. 2017. On calibration of modern neural networks. In *International Conference on Machine Learning*, pages 1321–1330. PMLR.

Dan Hendrycks and Kevin Gimpel. 2017. A baseline for detecting misclassified and out-of-distribution examples in neural networks. *ArXiv*, abs/1610.02136.

S. Hochreiter and J. Schmidhuber. 1997. Long short-term memory. *Neural Computation*, 9:1735–1780.

Diederik P Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*.

Armen Der Kiureghian and O. Ditlevsen. 2009. Aleatory or epistemic? does it matter? *Structural Safety*, 31:105–112.

Stefan Larson, Anish Mahendran, Joseph J. Peper, Christopher Clarke, Andrew Lee, Parker Hill, Jonathan K. Kummerfeld, Kevin Leach, Michael A. Laurenzano, Lingjia Tang, and Jason Mars. 2019. An evaluation dataset for intent classification and out-of-scope prediction. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*.

Shiyu Liang, Yixuan Li, and R. Srikant. 2018. Enhancing the reliability of out-of-distribution image detection in neural networks. *arXiv: Learning*.

Ting-En Lin and Hua Xu. 2019. Deep unknown intent detection with margin loss. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 5491–5496.

A. Malinin and M. Gales. 2018. Predictive uncertainty estimation via prior networks. In *Proceedings of the 32nd International Conference on Neural Information Processing Systems*, page 7047–7058.

Jeffrey Pennington, Richard Socher, and Christopher D Manning. 2014. Glove: Global vectors for word representation. In *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*, pages 1532–1543.

H. Shum, X. He, and Di Li. 2018. From eliza to xiaoice: challenges and opportunities with social chatbots. *Frontiers of Information Technology Electronic Engineering*, 19:10–26.

Evgenii Tsymbalov, K. Fedyanin, and Maxim Panov. 2020. Dropout strikes back: Improved uncertainty estimation via diversity sampled implicit ensembles. *ArXiv*, abs/2003.03274.

Amrita S Tulshan and Sudhir Namdeorao Dhage. 2018. Survey on virtual assistant: Google assistant, siri, cortana, alexa. In *International symposium on signal processing and intelligent recognition systems*, pages 190–201.

Yanan Wu, Keqing He, Yuanmeng Yan, QiXiang Gao, Zhiyuan Zeng, Fujia Zheng, Lulu Zhao, Huixing Jiang, Wei Wu, and Weiran Xu. 2022. Revisit overconfidence for OOD detection: Reassigned contrastive learning with adaptive class-dependent threshold. In *Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 4165–4179, Seattle, United States. Association for Computational Linguistics.

Hong Xu, Keqing He, Yuanmeng Yan, Sihong Liu, Zijun Liu, and Weiran Xu. 2020. A deep generative distance-based classifier for out-of-domain detection with mahalanobis space. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 1452–1460, Barcelona, Spain (Online). International Committee on Computational Linguistics.

Kai Yao and Jinwu Gao. 2016. Law of large numbers for uncertain random variables. *IEEE Transactions on Fuzzy Systems*, 24:615–621.

Zhiyuan Zeng, Keqing He, Yuanmeng Yan, Zijun Liu, Yanan Wu, Hong Xu, Huixing Jiang, and Weiran Xu. 2021a. Modeling discriminative representations for out-of-domain detection with supervised contrastive learning. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 2: Short Papers)*, pages 870–878, Online. Association for Computational Linguistics.

Zhiyuan Zeng, Keqing He, Yuanmeng Yan, Hong Xu, and Weiran Xu. 2021b. Adversarial self-supervised learning for out-of-domain detection. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 5631–5639, Online. Association for Computational Linguistics.

Yinhe Zheng, Guanyi Chen, and Minlie Huang. 2020. Out-of-domain detection for natural language understanding in dialog systems. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 28:1198–1209.

A Baseline Details

We compare many types of unsupervised OOD detection models. For detection algorithms, we use LOF(Local Outlier Factor)(Lin and Xu, 2019), GDA(Gaussian Discriminant Analysis)(Xu et al., 2020), MSP(Maximum Softmax Probability)(Hendrycks and Gimpel, 2017) and Entropy. For feature extractor, we use LSTM(Long Short Term Memory)(Hochreiter and Schmidhuber, 1997) and BERT(Bidirectional Encoder Representations from Transformers)(Devlin et al., 2019).

MSP (Maximum Softmax Probability)(Hendrycks and Gimpel, 2017) uses maximum softmax probability as the confidence score and regards an intent as OOD if the score is below a fixed threshold.

LOF (Local Outlier Factor)(Lin and Xu, 2019) A detecting unknown intents in the utterance algorithm with local density. It Assumes that unknown intents' local density is significantly lower than its k-nearest neighbor's.

GDA (Gaussian Discriminant Analysis) (Xu et al., 2020) A generative distance-based classifier for OOD detection with Euclidian space. For avoiding over-confidence problems, they estimate the class-conditional distribution on feature spaces of DNNs via Gaussian discriminant analysis. GDA is the state-of-the-art detection method till now, our proposed method using Bayesian approximation still significantly outperforms GDA. We also compare our method on two feature extractors for further study.

LSTM (Long Short Term Memory)(Hochreiter and Schmidhuber, 1997) A neural network that was proposed with the motivation of an analysis of Recurrent Neural Nets, which found that long time lags were inaccessible to existing architectures because backpropagated error either blows up or decays exponentially.

BERT (Bidirectional Encoder Representations from Transformers)(Devlin et al., 2019) A neural network that is trained to predict elided words in the text and then fine-tuned on our data. Note that they both trained only on labeled in-domain data using cross-entropy loss.

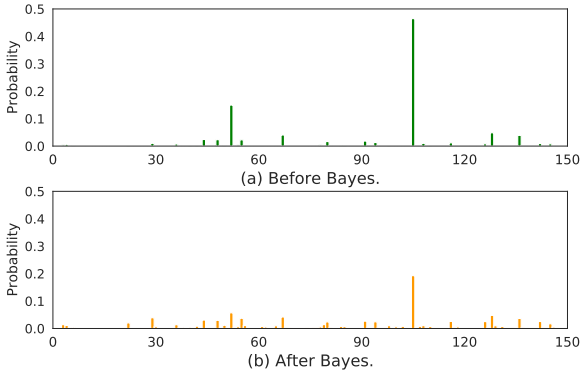

Figure 5: Effect of Bayesian approximation on softmax distribution of OOD sample.

B Implementation Details

We use the public pre-trained 300 dimensions GloVe embeddings (Pennington et al., 2014)[3] or bert-base-uncased (Devlin et al., 2019)[4] model to embed tokens. We use a two-layer BiLSTM as a feature extractor and set the dimension of hidden states to 128. We use Adam optimizer (Kingma and Ba, 2014) to train our model. We set a learning rate to 1E-03 for GloVe+LSTM and 1E-04 for BERT. In the training stage, We set the dropout probability to 0.5 and set the training epoch up to 200 with an early stop. We train only on in-domain labeled data. We use the best F1 scores on the validation set to calculate the detection method's threshold adaptively. For our proposed Bayesian approximation, we set the dropout probability to 0.7, and the dropout sampling times to 100. Each result of the experiments is tested 10 times under the same setting and gets the average value. The training stage of our model lasts about 4 minutes using GloVe embeddings, and 12 minutes using Bert-base-uncased, both on a single Tesla T4 GPU(16 GB of memory). The average value of the trainable model parameters is 3.05M. We will release our code after blind review.

C Visualization of softmax prediction distribution.

C.1 Visualization of OOD samples

In Fig 5, we give a 150-dimensional class distribution of an OOD sample to help understand our

[3]https://github.com/stanfordnlp/GloVe
[4]https://github.com/google-research/bert

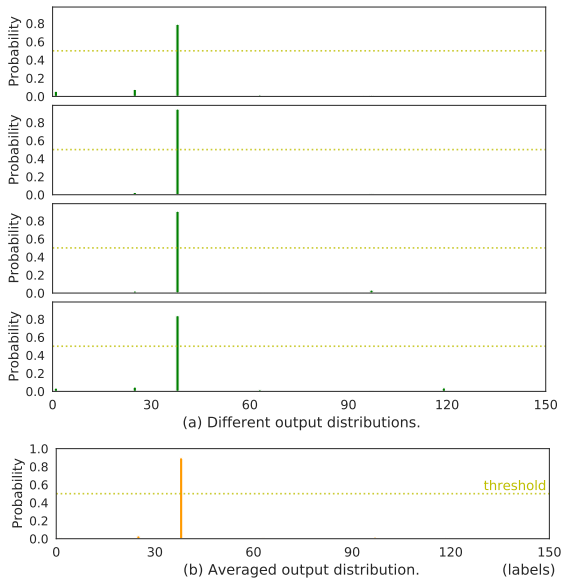

Figure 6: Multiple predicted probability distributions of an IND sample under different random seeds.

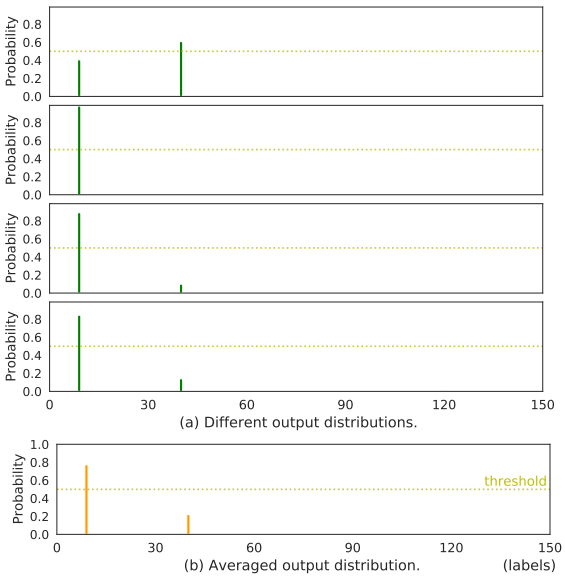

Figure 7: Multiple predicted probability distributions of an IND sample under different random seeds.

calibration process. The upper part of the figure is the distribution obtained by using the primary feature extractor. The softmax prediction distribution has obvious over-confidence in a particular IND category. The lower half of the figure presents the distribution after calibration by Bayesian approximation, which is flatter and meets the expectations of the OOD sample. When applying softmax-based detection methods, the latter will more easily recognized as OOD.

C.2 Visualization of IND samples

Corresponding to Fig 1, Fig 6 and Fig 7 show predicted probability distributions of two IND sample under different random seeds over 150 classes. We train four identical models on the same data but only use different random seeds. Fig (a) displays each output distribution of an IND input and Fig (b) shows the averaged output distribution. Specifically, Fig 6 shows when the input utterance is '*block my american saving bank for now*', the model obtains the maximum prediction probability on ground truth (*freeze_account*) under four random seeds and averaged output. Specifically, The maximum probabilities of prediction are 0.77, 0.96, 0.90 and 0.84 under different random seeds, and 0.87 under averaged output. In the experiments, we find that most IND samples present the state of Fig 6, that is, under different random samples setting, the model is very confident to give the input IND utterances with high confidence probability in the ground-truth category. We guess that this is because the model has seen some IND data in the training phase, and is familiar with IND classification, that is, the distribution uncertainty of IND is not serious. We also show another example in Fig 7 which the input utterance is '*where is improve the credit score*' and the corresponding true label is *improve_credit_score*. However, we find that under one random sampling setting, the model mispredicts into *credit_score* category with a probability of 0.61. We argue this is due to the fact that the two are easily confused with each other. Under this random sampling parameter setting, the model has not learned the feature ability to accurately distinguish these two categories. In addition, we also find that although there are wrong predictions, most of the IND predictions are accurate and have a high prediction probability, so that the highest prediction probability can still be obtained on the ground-truth label after averaging the distributions. This also reveals that our method will not damage the performance of IND classification, and can even avoid misjudgment among some confusing IND categories.

Tracking Satisfaction States for Customer Satisfaction Prediction in E-commerce Service Chatbots

Yang Sun[1], Liangqing Wu[2], Shuangyong Song[2], Xiaoguang Yu[2], Xiaodong He[2], Guohong Fu[1,3]*

[1]School of Computer Science and Technology, Soochow University, China
[2]JD AI Research, Beijing, China
[3]Institute of Artificial Intelligence, Soochow University, China
ysun23@stu.suda.edu.cn,ghfu@suda.edu.cn
{wuliangqing,songshuangyong,cdyuxiaoguang,xiaodong.he}@jd.com

Abstract

Due to the increasing use of service chatbots in E-commerce platforms in recent years, customer satisfaction prediction (CSP) is gaining more and more attention. CSP is dedicated to evaluating subjective customer satisfaction in conversational service and thus helps improve customer service experience. However, previous methods focus on modeling customer-chatbot interaction across different turns, which are hard to represent the important dynamic satisfaction states throughout the customer journey. In this work, we investigate the problem of satisfaction states tracking and its effects on CSP in E-commerce service chatbots. To this end, we propose a dialogue-level classification model named DialogueCSP to track satisfaction states for CSP. In particular, we explore a novel two-step interaction module to represent the dynamic satisfaction states at each turn. In order to capture dialogue-level satisfaction states for CSP, we further introduce dialogue-aware attentions to integrate historical informative cues into the interaction module. To evaluate the proposed approach, we also build a Chinese E-commerce dataset for CSP. Experiment results demonstrate that our model significantly outperforms multiple baselines, illustrating the benefits of satisfaction states tracking on CSP.

1 Introduction

Customer satisfaction prediction (CSP) in E-commerce service chatbots is dedicated to determining the customer satisfaction level such as *strongly satisfied*, *satisfied*, *neutral*, *dissatisfied*, or *strongly dissatisfied* with a specific conversational service she/he has just received, as shown in Figure 1. Due to the increasing use of service chatbots in E-commerce platforms in recent years (Song et al., 2019; Bodigutla et al., 2020), CSP is gaining more and more attention in the field of natural language processing. On the one hand, to deliver an effective conversational service and further enhance the ability of service chatbots, it is crucial to understand whether customers are satisfied with chatbot responses. On the other hand, CSP provides a straightforward way to dynamically monitor the performance of customer-chatbot interactions in terms of customer satisfaction and thus helps to intervene in problematic conversational services immediately (Liang et al., 2021). Once it is recognized that the customer is dissatisfied, we can immediately switch to manual service, so as to improve customer service experience and reduce customer churn (Yao et al., 2020).

Figure 1: An example of the CSP task. Customer satisfaction states (smiling or crying face) keep changing throughout the customer journey, contributing to the dialogue-level satisfaction.

Existing research on CSP focuses on two different tasks, namely the turn-level CSP (Pragst et al., 2017) and the dialogue-level CSP (Ultes, 2019). The former aims to determine the customer satis-

*Corresponding author.

Proceedings of the 29th International Conference on Computational Linguistics, pages 616–625
October 12–17, 2022.

faction at each turn of customer-chatbot interaction while the latter is a task to predict the overall customer satisfaction with the whole dialogue. As shown in Figure 1, in a real scenario of conversational service, a few customers are willing to give their feedback after service. Obviously, asking customers for turn-level feedback will undeniably lead to poor customer experience (Park et al., 2020). Therefore, in this study, we concentrate on the dialogue-level CSP.

Many approaches have been proposed for CSP with a focus on conversational context representation and customer-chatbot interaction modeling. While earlier works exploit manual features or recurrent neural networks (RNNs) to represent conversational context (Walker et al., 1997; Yang et al., 2010; Jiang et al., 2015; Choi et al., 2019), recent studies exert more efforts on modeling customer-chatbot interaction with attention mechanisms (Song et al., 2019) or similarity-based methods (Yao et al., 2020). Although these studies have greatly promoted the progress of the CSP technique, most of them concentrate on the interaction between customer questions and chatbot answers across different turns. However, chatbot answers from future turns are invisible to customers in a real scenario, so these methods are hard to represent important satisfaction states during the customer journey.

Actually, customer satisfaction states arise from customer-chatbot interaction and are dynamically changing throughout the customer journey (Lemon and Verhoef, 2016; Lee et al., 2020; Kvale et al., 2020). As shown in Figure 1, the customer is first dissatisfied at the turn (2) and then becomes satisfied at the turn (4) when the problem is solved smoothly, resulting in an overall satisfaction level *satisfied*. Furthermore, integrating historical context is helpful for representing the satisfaction states at each turn. For example, in the dialogue in Figure 1, the customer asks a more detailed question at the turn (3) based on the preceding response *"describe it again"* from the chatbot.

To address the aforementioned issues, we propose a dialogue-level classification model for CSP in E-commerce service chatbots, namely DialogueCSP. It consists of three main modules: Firstly, a dialogue encoding module exploits convolutional neural networks (CNNs) (Kim, 2014) and Long Short-Term Memory (LSTM) networks to capture conversational context. Secondly, an interaction module is used to represent the customer satisfaction states at each turn. In particular, the interaction module utilizes two Gated Recurrent Units (GRUs) (Chung et al., 2014) to perform a two-step customer-chatbot interaction, namely local question-answer interaction and satisfaction state interaction. Furthermore, we introduce dialogue-aware attentions, including question attention, answer attention, and state attention. While the former two attentions integrate historical cues into the interaction module, the latter captures dialogue-level satisfaction representations. Finally, a decoding module is applied to predict the customer satisfaction for each dialogue. We also construct a Chinese E-commerce customer satisfaction prediction dataset (CECSP) that contains approximately 30k conversational services. Experimental results demonstrate that the proposed model outperforms the current state of the art on CECSP and other two benchmark datasets.

In summary, we make the following contributions:

- We propose a dialogue-level classification model for customer satisfaction prediction.
- We explore a novel two-step interaction module to handle both local question-answer and customer satisfaction state interactions at each turn and further integrate it with historical cues using dialogue-aware attentions to handle dialogue-level satisfaction representations.
- We construct a large Chinese E-commerce CSP dataset (CECSP). Experimental results show that the proposed model outperforms multiple baselines.[1]

2 Related Work

Recently, CSP has attracted much attention due to the increasing use of service chatbots in many different aspects of our lives (Hashemi et al., 2018; Choi et al., 2019; Kachuee et al., 2021). Some studies focus on addressing turn-level satisfaction prediction with human annotations (Pragst et al., 2017; Rach et al., 2017). However, they are not scalable in terms of annotation costs due to the large volume of conversational services in E-commerce. Therefore, recent studies explore contrastive learning (Kachuee et al., 2021) and reinforcement learning

[1]Our code is available at https://github.com/McSumail/DialogueCSP, and the dataset will be released after encryption.

(Liang et al., 2021) to make them more suitable for E-commerce customer service.

Most of the existing works exert more effort on dialogue-level satisfaction prediction since few customers are willing to give their feedback after service. While earlier methods rely on manual features (Walker et al., 1997; Yang et al., 2010), recent studies use deep neural networks to model conversational context and customer-chatbot interaction. Hashemi et al. (2018) exploit LSTMs to capture the sequential context features within a dialogue and use the hidden states of the last turn for satisfaction prediction. To enhance dialogue-level representations, Ultes (2019) apply an attention mechanism over LSTM layers to capture information from each turn. To model customer-chatbot interaction, Song et al. (2019) use each customer question to capture relevant information from all chatbot answers, while Yao et al. (2020) compute the semantic similarity scores between customer questions and chatbot answers across different turns. However, these methods both exploit the information from future turns that are invisible to customers in a real scenario to capture turn-level features. Therefore, they are hard to model the customer journey and track the dynamic satisfaction states within a conversational service. This work differs in that we consider both question-answer and customer satisfaction state interactions at each turn, and thus design a novel two-step interaction module to track the satisfaction states throughout the customer journey.

3 Dataset

For our experiments, we collect conversational services from one of the largest E-commerce platforms and construct a Chinese E-commerce CSP dataset. In the following, we will introduce the annotation strategy and compare this dataset with other benchmark datasets (Song et al., 2019).

3.1 Dataset Annotation

We use **real customer feedback** as the dialogue-level satisfaction labels which include *strongly satisfied*, *satisfied*, *neutral*, *dissatisfied*, and *strongly dissatisfied*. For the quality of the annotation, we then assign several experienced customer service coordinator to check whether the feedback is consistent with the conversational service, and about 20% dialogues were excluded from the final dataset.

Statistics items	CECSP	Clothes	Makeup
# of Train	22576	8000	2832
# of Val	2822	1000	354
# of Test	2801	1000	354
# of strongly dissatisfied	3158	-	-
# of dissatisfied	1417	2302	1180
# of neutral	2633	6399	1180
# of satisfied	10840	1299	1180
# of strongly satisfied	10151	-	-
Avg. # of turns per dialog	3.67	8.14	8.01
Max # of turns per dialog	10	18	16
Min # of turns per dialog	1	2	2
Multiple domains	Yes	No	No
Turn-level annotation	No	Yes	Yes

Table 1: The comparison of the three datasets in some key statistics. While **CECSP** is our constructed Chinese E-commerce CSP dataset, **Clothes** and **Makeup** are two benchmark datasets.

3.2 Comparison with Other Datasets

Table 1 shows some key statistics of the three datasets. As we can see, CECSP consists of more but shorter conversational service compared to **Clothes** (Song et al., 2019) and **Makeup** (Song et al., 2019). While **Clothes** and **Makeup** only collect conversational services in post-sale, CECSP consists of dialogues from multiple domains such as logistic, post-sale and VIP service. Due to ethical concerns, we follow Song et al. (2019) and transform segmented Chinese word[2] into word index in the final dataset.

4 Methodology

4.1 Problem Definition

Suppose there is a conversational service consisting of n turns of interaction $\{(q_1 : a_1), (q_2 : a_2), ..., (q_n : a_n)\}$, where q_i is the i-th question asked by the customer and a_i is its corresponding answer from the chatbot, the goal of CSP is to predict the satisfaction label for this dialogue, which is one of the five classes: *strongly satisfied*, *satisfied*, *neutral*, *dissatisfied*, and *strongly dissatisfied*.

4.2 Model Overview

As illustrated in Figure 2, the proposed framework for CSP consists of three main components, namely dialogue encoding, satisfaction states tracking, and satisfaction prediction. Firstly, we encode the utterances of input dialogues into context-dependent vectors. Next, an interaction module

[2]The segmentation toolkit is open source and available at https://github.com/fxsjy/jieba

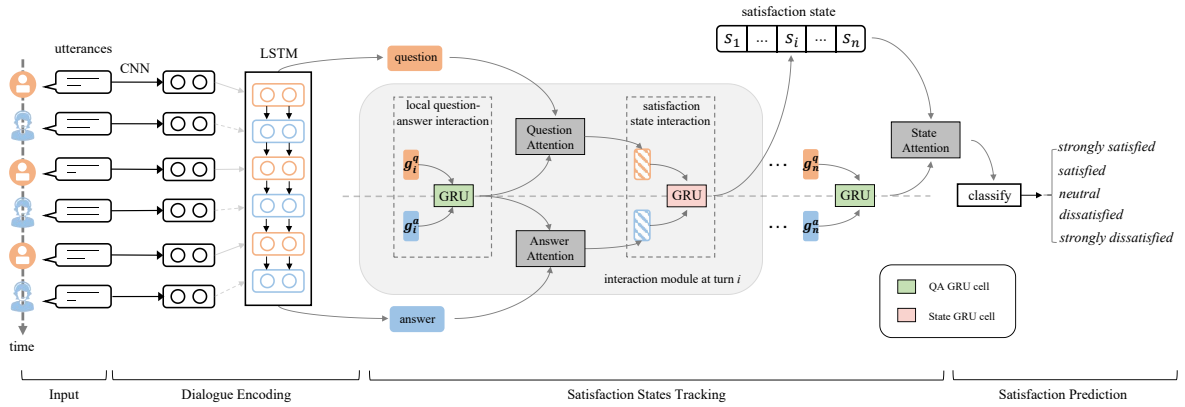

Figure 2: Overview of our proposed model for dialogue-level CSP, congruent to the illustration in Methodology.

with two GRU cells is applied to perform a two-step customer-chatbot interaction to represent the customer satisfaction states at each turn. Meanwhile, dialogue-aware attentions integrate the historical information into the interaction module and capture dialogue-level satisfaction representations. Finally, the dialogue-level satisfaction representations are used to predict satisfaction labels for dialogues. In the following sections, we will explain each component in detail.

4.3 Dialogue Encoding

The input of our model is a sequence of utterances consisting of word index. The goal of dialogue encoding is to encode the utterance sequence into context-dependent vectors using CNNs and LSTM for subsequent customer-chatbot interaction.[3]

4.3.1 Utterance Encoding

CNNs are capable of capturing n-gram information from an utterance (Kim, 2014). We leverage a CNN layer with max-pooling to extract context-independent features of each utterance. Concretely, the input is the 300 dimensional pre-trained 840B GloVe vectors (Pennington et al., 2014). We employ three filters of size 3, 4, and 5 with 50 feature maps each. These feature maps are further processed by max-pooling and ReLU activation (Nair and Hinton, 2010). Then, these features are concatenated and fed to a 100 dimensional fully connected layer, whose activations form the representations of the utterances.

[3]We also used pre-trained BERT-Base to encode the original conversational service from CECSP, but the results were not satisfactory.

4.3.2 Context Encoding

The LSTM introduces gating mechanism into recurrent neural networks to capture long-term dependencies from input sequences. In this part, we use a LSTM network to capture sequential context information,

$$g_i = \text{LSTM}\left(g_{i-1}, u_i\right) \tag{1}$$

where $i = 1, 2, \ldots, n$, u_i and g_i are context-independent and sequential utterance representations, respectively. Then, we denote question and answer representations as $M^q = [g_1^q, g_2^q, \ldots, g_n^q]$ and $M^a = [g_1^a, g_2^a, \ldots, g_n^a]$.

4.4 Satisfaction States Tracking

Since customer satisfaction states keep changing throughout the customer journey, we design an interaction module to perform a two-step customer-chatbot interaction to represent the customer satisfaction states at each turn. Figure 2 shows the details of the interaction module at turn i.

4.4.1 Dialogue-aware Attention

Attention mechanisms aim to capture the most relevant information and are widely applied on different natural language processing tasks (Bahdanau et al., 2015; Luo et al., 2018; Sinha et al., 2018). Given the query q, the key k, and the value v, the attention output o is computed as follows:

$$w = f\left(q, k\right) \tag{2}$$

$$\tilde{w} = w - m \tag{3}$$

$$o = \text{softmax}(\tilde{w})v \tag{4}$$

where f is a function that computes a single scalar from q and k. The attention mask m is a matrix with the same shape as the attention weights w.

The value of m_j is set to be $+\infty$ only when the attention for the j-th vector in k is masked, and set to be 0 otherwise.

In conversational service, the customer satisfaction state at turn i are most related to the questions and answers at turn (1)~(i) (Lemon and Verhoef, 2016). Therefore, the attention mechanism used by Song et al. (2019) that model the customer-chatbot interaction across different turns is hard to capture satisfaction states throughout the customer journey. To address this issue, we design dialogue-aware attentions by using different inputs and masking strategies to integrate historical cues into the interaction module and capture dialogue-level satisfaction representations.

4.4.2 Local Question-Answer Interaction

Since customer satisfaction states arise from the customer-chatbot interaction (Lee et al., 2020; Kvale et al., 2020), we adopt a QA GRU cell to model the local question-answer interaction and capture satisfaction features,

$$s_i^{qa} = \mathrm{GRU}^{qa}\left(g_i^a, g_i^q\right) \tag{5}$$

where $i = 1, 2, \ldots, n$.

4.4.3 Question Attention

Due to the nature of dialogues, contextual information plays an vital role in customer satisfaction states (Lemon and Verhoef, 2016; Kvale et al., 2020). Therefore, we design an attention mechanism to match relevant historical cues from the question representations:

$$q, k, v = s_i^{qa}, M^q, M^q \tag{6}$$

$$m_j^{que} = \begin{cases} +\infty, & j \notin \{\tilde{g}_1^q, \tilde{g}_2^q, \ldots, \tilde{g}_i^q\} \\ 0, & \text{Otherwise} \end{cases} \tag{7}$$

$$\tilde{q}_i = \mathrm{QueAttn}\left(q, k, v, m_j^{que}\right) \tag{8}$$

The masking strategy m_j^{que} separates future turns from the interaction at the current turn, which is more consistent with the customer journey.

4.4.4 Answer Attention

We also devise another attention mechanism to capture historical cues from the answer representations:

$$q, k, v = s_i^{qa}, M^a, M^a \tag{9}$$

$$m_j^{ans} = \begin{cases} +\infty, & j \notin \{\tilde{g}_1^a, \tilde{g}_2^a, \ldots, \tilde{g}_i^a\} \\ 0, & \text{Otherwise} \end{cases} \tag{10}$$

$$\tilde{a}_i = \mathrm{AnsAttn}(q, k, v, m_j^{ans}) \tag{11}$$

4.4.5 Satisfaction State Interaction

With the attention mechanisms described above, we successfully collect informative cues from the historical questions and answers. Then, we use a State GRU cell to lever these cues to represent the customer satisfaction state s_i at turn i,

$$s_i = \mathrm{GRU}^s\left(\tilde{a}_i, \tilde{q}_i\right) \tag{12}$$

where $i = 1, 2, \ldots, n$.

4.4.6 State Attention

After applying the two-step interaction module at each turn, we denote the customer satisfaction states as $S = [s_1, s_2, \ldots, s_n]$. Then, we use state attention to capture the dialogue-level satisfaction representations $\tilde{s}$:

$$q, k, v = s_n^{qa}, S, S \tag{13}$$

$$m_j^{sta} = 0 \tag{14}$$

$$\tilde{s} = \mathrm{StaAttn}(q, k, v, m_j^{sta}) \tag{15}$$

4.5 Satisfaction Prediction

Finally, we classify each conversational service using a fully connected network:

$$h = \mathrm{ReLU}(W_r \tilde{s} + b_r) \tag{16}$$

$$\mathcal{P} = \mathrm{softmax}(W_{smax} h + b_{smax}) \tag{17}$$

$$\hat{y} = \underset{k}{\mathrm{argmax}}\left(\mathcal{P}[k]\right) \tag{18}$$

To train the model, we choose the cross-entropy loss function:

$$\mathcal{L}(\theta) = -\sum_{v \in y_V} \sum_{z=1}^{Z} Y_{vz} \ln P_{vz} \tag{19}$$

where $y_{\mathcal{V}}$ is the set of dialogue indices that have labels and Y is the label indicator matrix, and θ is the collection of trainable parameters in DialogueCSP.

5 Experimental Settings

In this section, we present the experimental settings including implementation details and baselines.

5.1 Implementation Details

We use the validation set to tune hyperparameters. The batch size is set to be {128,64,64} for CECSP, Clothes, and Makeup. We adopt Adam (Kingma and Ba, 2015) as the optimizer with an initial learning rate of {1e-3,1e-4,1e-4} and L2 weight decay of {1e-4, 1e-5, 1e-5} for CECSP, Clothes, and

Makeup, respectively. The dropout (Srivastava et al., 2014) is set to be 0.5. We train all models for a maximum of 100 epochs and stop training if the validation loss does not decrease for 20 consecutive epochs.

5.2 Baseline Methods

For a comprehensive evaluation of our proposed DialogueCSP, we compare it with the following baseline methods:

LSTMCSP (Hashemi et al., 2018): This model adopts a Bi-directional LSTM network to capture the contextual information of conversational services and uses the hidden states of the last turn for satisfaction prediction.

LSTM+Attn (Ultes, 2019): This model applies an attention mechanism over Bi-directional LSTM layers to capture information from all turns within a service.

DialogueGCN (Ghosal et al., 2019): It is a graph-based model which encodes the relative positions between customers and chatbots within a window context.

CAMIL (Song et al., 2019): This model uses each question to capture information from all answers to model customer-chatbot interaction. Additionally, it exploits turn-level sentiment information by multiple instance learning.

LSTM+MTL (Bodigutla et al., 2020): It is a multi-task learning network that uses the hidden states of LSTM layers to predict dialogue-level and turn-level satisfaction jointly.

LSTM-Cross (Yao et al., 2020): It is the latest work for dialogue-level CSP which uses LSTM networks to capture contextual features and computes the semantic similarity scores between customer questions and chatbot answers across different turns. Then, these similarity scores are concatenated with the contextual features for satisfaction prediction.

6 Results and Analysis

6.1 Overall Results

Table 2 shows the comparison results for CSP in conversational services. Our proposed DialogueCSP consistently achieves better performance than the baseline methods on all datasets, while being statistically significant under the paired t-test ($p<0.05$). Besides, we can make another three observations as follows, which help to understand the CSP task and the advantages of DialogueCSP.

Model	CECSP		Clothes		Makeup	
	Acc.	F1	Acc.	F1	Acc.	F1
LSTMCSP	51.85	49.57	75.59	75.78	76.31	76.56
LSTM+Attn	53.09	51.02	77.12	77.28	77.56	77.52
DialogueGCN	53.69	51.35	76.89	76.82	77.72	77.78
CAMIL	55.43	52.92	78.30$^{\#}$	78.40	78.50$^{\#}$	78.64
LSTM+MTL	–	–	78.21	78.12	78.18	78.08
LSTM-Cross	55.51	53.11	78.91	79.33	79.88	79.58
DialogueCSP	**57.48**	**54.98**	**81.18**	**80.93**	**81.30**	**81.62**

Table 2: Overall performance on the three datasets. We use the accuracy and the weighted F1 score to evaluate each model. Scores marked by "#" are reported results, while others are based on our re-implementation.

Firstly, although LSTM+Attn only applies a vanilla attention mechanism compared to LSTM-CSP, the improvements on the three datasets are significant. This indicates that dialogue-level CSP must capture information from all turns in conversational services. Since chatbots respond to each customer question immediately, the relative positions between customer questions and chatbot answers are fixed. Therefore, the position model in DialogueGCN does not work here.

Secondly, CAMIL takes turn-level sentiment information into account and achieve better performance than previous strategies. However, the improvement of the method on CECSP is more obvious than that on Clothes and Makeup. After examining the datasets, we find that the average conversational service length is 3.67 turns in CECSP which is much shorter than that in Clothes and Makeup. When the lengths are short, especially only 1 or 2 turns, overall satisfaction is more related to turn-level sentiment information (Bodigutla et al., 2020).

Thirdly, CAMIL and LSTM-Cross achieve better performance than other baselines due to their customer-chatbot interaction modeling methods. While these methods focus on questions and answers across different turns, our proposed DialogueCSP exploits a two-step interaction module to better model the customer journey and thus capture important customer satisfaction states.

6.2 Different Interaction Modeling Methods

In this section, we make a comparison between different interaction modeling methods. To this end, we modify our two-step interaction module with the following two methods. The first one is the same as LSTM-Cross (Yao et al., 2020). We compute the semantic similarity scores between the question and answer at the same turn. Then we con-

Method	Weighted F1 score		
	CECSP	Clothes	Makeup
DialogueCSP	**54.98**	**80.93**	**81.62**
DialogueCSP-similarity	54.01	79.71	80.38
DialogueCSP-global attn	54.34	80.07	80.90

Table 3: Results of comparison between different interaction modeling methods. We modify our interaction module with another two methods and evaluate them on the three datasets.

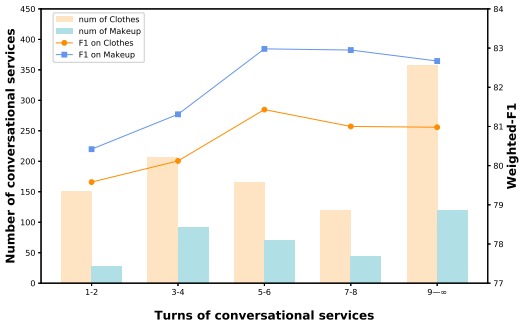

Figure 3: The influence of conversational service length on CSP. We divide the test set of Clothes and Makeup into five subsets in terms of conversational turns and further evaluate DialogueCSP over these subsets.

catenate them with obtained contextual features for satisfaction prediction. The second one is replacing the dialogue-aware attentions with the global attention (Song et al., 2019) to capture contextual information.

The results of different interaction modeling methods are shown in Table 3. We observe that our interaction modeling method is around 1% better than other methods in weighted F1 scores. Since customers can directly choose the options provided by chatbots, high semantic similarity scores don't always mean the high customer satisfaction. For instance, if customers choose "None of the above" from provided options, they may be dissatisfied. Besides, chatbot answers from future turns are invisible to customers within a conversational service. Therefore, global attention used in Song et al. (2019) is hard to capture the customer satisfaction states during the customer journey, leading to its inferior performance.

6.3 Influence of Conversational Service Length

In this section, we experiment on Clothes and Makeup to examine the influence of conversational service length.

Method	Weighted F1 score		
	CECSP	Clothes	Makeup
DialogueCSP	**54.98**	**80.93**	**81.62**
- 1st-step inter	54.60(↓ 0.38)	80.61(↓ 0.32)	80.94(↓ 0.68)
- 2nd-step inter	54.02(↓ **0.96**)	80.08(↓ **0.85**)	80.46(↓ **1.16**)
- Question Attn	54.51(↓ 0.47)	80.49(↓ 0.44)	80.80(↓ 0.82)
- Answer Attn	54.43(↓ 0.55)	80.37(↓ 0.56)	80.90(↓ 0.72)
- State Attn	54.64(↓ 0.34)	80.21(↓ 0.72)	80.64(↓ 0.98)

Table 4: Results of ablation study on the three datasets. 1st-step inter and 2nd-step inter stand for first-step interaction and second-step interaction, respectively.

As shown in Figure 3, whether on Clothes or Makeup, as the turns of conversational services increase, the performance of our proposed approach first rises significantly and then decreases. When conversational services length is short, there are few changes of customer satisfaction states (Lemon and Verhoef, 2016; Lee et al., 2020). Therefore, in these cases, the interaction module in DialogueCSP that captures satisfaction states does not work. Moreover, DialogueCSP uses dialogue-aware attentions to integrate historical information into customer-chatbot interaction. When the turns of services increase, there are more informative cues from preceding questions and answers which contribute to customer satisfaction states. As a result, DialogueCSP achieves weighted F1 scores of 81.43% and 82.98% on the subsets where the turns are 5 or 6. Further, it is still a challenge to handle the intricate context information when the turns are over 6, leading to the decline of DialogueCSP.

6.4 Ablation Study

In this ablation study, we analyze the impact of five components by removing one of them at a time from DialogueCSP. The results are presented in Table 4.

We can observe that the performance of DialogueCSP drops on the three datasets when any of the components is removed, suggesting that all these components contribute to the improvement of DialogueCSP. However, their contributions can be distinguished. By eliminating second-step interaction, our model drops the most by 0.96% on CECSP, 0.85% on Clothes, and 1.16% on Makeup in weighted F1 scores, which implies the importance of modeling the satisfaction state interaction.

Moreover, we found that Question Attention and Answer Attention also play important roles in our model. This phenomenon supports our argument that customer satisfaction states have close bonds with not only the questions and answers at

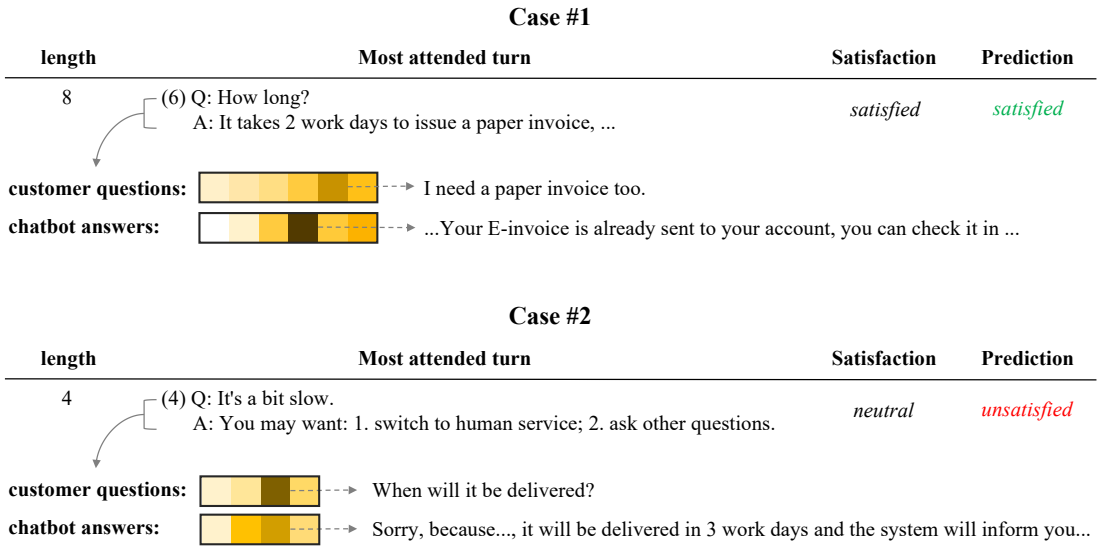

Figure 4: Results of case analysis, where some turns of two conversational services are provided, along with the visualization of attention weights between different context memories and the most attended turn (selected according to the highest attention weight computed by State Attention). The darker colors mean larger attention weights.

the current single turn but also historical information. Further, while State Attention is more important than Question Attention and Answer Attention on Clothes and Makeup, it is the opposite on CECSP. After delving into the datasets, we found that the average conversational service length is around 8 turns in Clothes and Makeup, which is much longer than that in CECSP. Therefore, it is important to weigh multiple satisfaction states to generate dialogue-level representations on Clothes and Makeup.

6.5 Case Analysis

For a comprehensive understanding of our proposed method, we visualize its performance by a case analysis on the test set of CECSP. In short, we found that integrating historical information into customer-chatbot interaction can be a double-edged sword. As illustrated in Figure 4, the dialogue-aware attentions can capture useful historical information and help make a good prediction (Case #1). However, focusing too much on historical information may hinder the understanding of neutral utterances of customers (Case #2). Therefore, it is necessary to explore other mechanisms rather than merely relying on popular attention to handle historical information for CSP.

Besides, we also observe from these two cases that the most attended turns of customer satisfaction states are among the end of the dialogues. After examining the whole test sets of the three datasets, we found that 40% of the most attended turns are the last turn of conversational services, which is in tune with the conclusion from the previous studies (Hashemi et al., 2018; Yao et al., 2020).

7 Conclusion

In this paper, we investigate the importance of satisfaction states tracking in dialogue-level CSP in E-commerce service chatbots. We propose a dialogue-level classification model and design a two-step interaction module to handle both local question-answer and customer satisfaction state interactions throughout the customer journey. To capture dialogue-level satisfaction representations, we further introduce dialogue-aware attentions to integrate historical information into the interaction module. Besides, we also build a Chinese E-commerce dataset for CSP to evaluate the proposed approach. Experimental results on this dataset and two released corpora show that our proposed model outperforms all the baselines. Our further analysis illustrates that tracking the satisfaction states is more helpful for modeling customer-chatbot interaction than previous strategies. In addition, our experiments also show that integrating historical information with customer-chatbot interaction is of great value to CSP.

In our future work, we would like to explore more effective methods to model customer-chatbot interaction. Moreover, we also plan to investigate the importance of customer intentions in handling informative cues for CSP.

Acknowledgments

We thank the anonymous reviewers for their valuable comments. This work was supported by the National Natural Science Foundation of China (No.62076173, U1836222), the High-level Entrepreneurship and Innovation Plan of Jiangsu Province (No.JSSCRC2021524), and the Project Funded by the Priority Academic Program Development of Jiangsu Higher Education Institutions.

References

Dzmitry Bahdanau, Kyung Hyun Cho, and Yoshua Bengio. 2015. Neural machine translation by jointly learning to align and translate. In *3rd International Conference on Learning Representations, ICLR 2015*.

Praveen Kumar Bodigutla, Aditya Tiwari, Spyros Matsoukas, Josep Valls-Vargas, and Lazaros Polymenakos. 2020. Joint turn and dialogue level user satisfaction estimation on multi-domain conversations. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 3897–3909, Online. Association for Computational Linguistics.

Jason Ingyu Choi, Ali Ahmadvand, and Eugene Agichtein. 2019. Offline and online satisfaction prediction in open-domain conversational systems. In *Proceedings of the 28th ACM International Conference on Information and Knowledge Management*, pages 1281–1290.

Junyoung Chung, Caglar Gulcehre, Kyunghyun Cho, and Yoshua Bengio. 2014. Empirical evaluation of gated recurrent neural networks on sequence modeling. In *NIPS 2014 Workshop on Deep Learning, December 2014*.

Deepanway Ghosal, Navonil Majumder, Soujanya Poria, Niyati Chhaya, and Alexander Gelbukh. 2019. DialogueGCN: A graph convolutional neural network for emotion recognition in conversation. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 154–164, Hong Kong, China. Association for Computational Linguistics.

Seyyed Hadi Hashemi, Kyle Williams, Ahmed El Kholy, Imed Zitouni, and Paul A Crook. 2018. Measuring user satisfaction on smart speaker intelligent assistants using intent sensitive query embeddings. In *Proceedings of the 27th ACM International Conference on Information and Knowledge Management*, pages 1183–1192.

Jiepu Jiang, Ahmed Hassan Awadallah, Rosie Jones, Umut Ozertem, Imed Zitouni, Ranjitha Gurunath Kulkarni, and Omar Zia Khan. 2015. Automatic online evaluation of intelligent assistants. In *Proceedings of the 24th International Conference on World Wide Web*, pages 506–516.

Mohammad Kachuee, Hao Yuan, Young-Bum Kim, and Sungjin Lee. 2021. Self-supervised contrastive learning for efficient user satisfaction prediction in conversational agents. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 4053–4064.

Yoon Kim. 2014. Convolutional neural networks for sentence classification. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1746–1751, Doha, Qatar. Association for Computational Linguistics.

Diederik P Kingma and Jimmy Ba. 2015. Adam: A method for stochastic optimization. In *International Conference on Learning Representations*.

Knut Kvale, Eleonora Freddi, Stig Hodnebrog, Olav Alexander Sell, and Asbjørn Følstad. 2020. Understanding the user experience of customer service chatbots: What can we learn from customer satisfaction surveys? In *International Workshop on Chatbot Research and Design*, pages 205–218. Springer.

Ching-Hung Lee, Qiye Li, Yu-Chi Lee, and Chih-Wen Shih. 2020. Service design for intelligent exhibition guidance service based on dynamic customer experience. *Industrial Management & Data Systems*.

Katherine N Lemon and Peter C Verhoef. 2016. Understanding customer experience throughout the customer journey. *Journal of marketing*, 80(6):69–96.

Runze Liang, Ryuichi Takanobu, Fenglin Li, Ji Zhang, Haiqing Chen, and Minlie Huang. 2021. Turn-level user satisfaction estimation in e-commerce customer service.

Ling Luo, Zhihao Yang, Pei Yang, Yin Zhang, Lei Wang, Hongfei Lin, and Jian Wang. 2018. An attention-based bilstm-crf approach to document-level chemical named entity recognition. *Bioinformatics*, 34(8):1381–1388.

Vinod Nair and Geoffrey E Hinton. 2010. Rectified linear units improve restricted boltzmann machines. In *Icml*.

Dookun Park, Hao Yuan, Dongmin Kim, Yinglei Zhang Spyros Matsoukas, Young-Bum Kim, Ruhi Sarikaya Chenlei Guo Yuan Ling, Kevin Quinn, Tuan-Hung Pham, and Benjamin Yao Sungjin Lee. 2020. Large-scale hybrid approach for predicting user satisfaction with conversational agents.

Jeffrey Pennington, Richard Socher, and Christopher Manning. 2014. GloVe: Global vectors for word representation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1532–1543, Doha, Qatar. Association for Computational Linguistics.

Louisa Pragst, Stefan Ultes, and Wolfgang Minker. 2017. Recurrent neural network interaction quality estimation. In *Dialogues with Social Robots*, pages 381–393. Springer.

Niklas Rach, Wolfgang Minker, and Stefan Ultes. 2017. Interaction quality estimation using long short-term memories. In *Proceedings of the 18th Annual SIGdial Meeting on Discourse and Dialogue*, pages 164–169.

Koustuv Sinha, Yue Dong, Jackie Chi Kit Cheung, and Derek Ruths. 2018. A hierarchical neural attention-based text classifier. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 817–823, Brussels, Belgium. Association for Computational Linguistics.

Kaisong Song, Lidong Bing, Wei Gao, Jun Lin, Lujun Zhao, Jiancheng Wang, Changlong Sun, Xiaozhong Liu, and Qiong Zhang. 2019. Using customer service dialogues for satisfaction analysis with context-assisted multiple instance learning. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 198–207, Hong Kong, China. Association for Computational Linguistics.

Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. 2014. Dropout: a simple way to prevent neural networks from overfitting. *The journal of machine learning research*, 15(1):1929–1958.

Stefan Ultes. 2019. Improving interaction quality estimation with bilstms and the impact on dialogue policy learning. In *Proceedings of the 20th Annual SIGdial Meeting on Discourse and Dialogue*, pages 11–20.

Marilyn A. Walker, Diane J. Litman, Candace A. Kamm, and Alicia Abella. 1997. PARADISE: A framework for evaluating spoken dialogue agents. In *35th Annual Meeting of the Association for Computational Linguistics and 8th Conference of the European Chapter of the Association for Computational Linguistics*, pages 271–280, Madrid, Spain. Association for Computational Linguistics.

Zhaojun Yang, Baichuan Li, Yi Zhu, Irwin King, Gina Levow, and Helen Meng. 2010. Collaborative filtering model for user satisfaction prediction in spoken dialog system evaluation. In *2010 IEEE Spoken Language Technology Workshop*, pages 472–477. IEEE.

Riheng Yao, Shuangyong Song, Qiudan Li, Chao Wang, Huan Chen, Haiqing Chen, and Daniel Dajun Zeng. 2020. Session-level user satisfaction prediction for customer service chatbot in e-commerce (student abstract). In *Proceedings of the AAAI Conference on Artificial Intelligence*, 10, pages 13973–13974.

Towards Multi-label Unknown Intent Detection

Yawen Ouyang Zhen Wu* Xinyu Dai Shujian Huang Jiajun Chen
National Key Laboratory for Novel Software Technology, Nanjing University, China
ouyangyw@smail.nju.edu.cn
{wuz,daixinyu,huangsj,chenjj}@nju.edu.cn

Abstract

Multi-class unknown intent detection has made remarkable progress recently. However, it has a strong assumption that each utterance has only one intent, which does not conform to reality because utterances often have multiple intents. In this paper, we propose a more desirable task, *multi-label unknown intent detection*, to detect whether the utterance contains the unknown intent, in which each utterance may contain multiple intents. In this task, the unique utterances simultaneously containing known and unknown intents make existing multi-class methods easy to fail. To address this issue, we propose an intuitive and effective method to recognize whether **A**ll **I**ntents contained in the utterance are **K**nown (**AIK**). Our high-level idea is to predict the utterance's intent number, then check whether the utterance contains the same number of known intents. If the number of known intents is less than the number of intents, it implies that the utterance also contains unknown intents. We benchmark AIK over existing methods, and empirical results suggest that our method obtains state-of-the-art performances. For example, on the MultiWOZ 2.3 dataset, AIK significantly reduces the FPR95 by 12.25% compared to the best baseline.[1]

1 Introduction

Intent classification is a crucial component of task-oriented dialogue systems, which aims to map the utterance to the known intent set. In an open environment, it is nearly impossible that dialogue systems are only exposed to utterances with known intents, i.e., in-distribution (IND) utterances. Therefore, unknown intent detection is proposed to identify the out-of-distribution (OOD) utterance, which contains the unknown intent (Hendrycks and Gimpel, 2017). It can prevent dialogue systems from generating unrelated responses to ensure good user experiences. The detected OOD utterances can also provide future direction for developers.

Utterance	I am looking to stay at the Lovell Lodge hotel and to see the areas local attractions.
Intent	*Inform-Hotel-Name*, *Request-Attraction-Area*

Table 1: An example of utterance with multiple intents from MultiWOZ 2.3 (Han et al., 2020). For dialogue systems designed for the hotel domain, the utterance is mixed OOD as it contains known intent *Inform-Hotel-Name* and unknown intent *Request-Attraction-Area*.

Recent works follow the assumption that each utterance has only one intent and focus on *multi-class unknown intent detection* (Podolskiy et al., 2021; Ouyang et al., 2021; Lin and Xu, 2019; Shu et al., 2017). Based on this assumption, a rather popular strategy to perform OOD detection relies on the maximum classifier output. For example, Shu et al. (2017) propose using the maximum binary classifier output. An utterance will be regarded as containing the known intent and classified as IND if the output is larger than the predefined threshold, otherwise it will be classified as OOD.

Nevertheless, the above assumption is too strong: several intents are usually expressed in an utterance in a real-world scenario. For example, Gangadharaiah and Narayanaswamy (2019) show that 52% of utterances include multi-label intents in the amazon internal dataset. It is obvious that the multi-class unknown intent detection research cannot fully meet the needs of dialogue systems.

In this work, we propose a more practical task, *multi-label unknown intent detection*, which is to detect whether the user utterance contains unknown intents, where each utterance may contain multiple intents. We summarize three types of utterances for unknown intent detection in the multi-label setting: 1) **IND utterances**, only containing known intents; 2) **pure OOD utterances**, only containing unknown intents; and 3) **mixed OOD utterances** simultaneously containing known and unknown in-

* Corresponding author.

[1] Code and data are available at https://github.com/yawenouyang/AIK.

Proceedings of the 29th International Conference on Computational Linguistics, pages 626–635
October 12–17, 2022.

tents (see Table 1 for an example). Note that mixed OOD utterances are unique to multi-label because utterances in multi-class can only have one intent.

The existence of mixed OOD utterances brings a great challenge for multi-label unknown intent detection, which makes the existing strategy easy to fail. As shown in Figure 1, such methods will regard the mixed OOD utterances as IND utterances once detecting the known intents.

To address the above issue, we propose a novel method, by recognizing whether **A**ll **I**ntents of the utterance are **K**nown (**AIK**), to detect both pure and mixed OOD utterances for multi-label unknown intent detection. Overall, we first predict the number of intents contained in the utterance. Then we check whether the utterance contains the same number of known intents by measuring the probability density of the utterance's known intent-wise representations. Specifically, we assume the known intent-wise representation can be fitted well by a conditional Gaussian distribution, then we can estimate its probability density via the Mahalanobis distance. We empirically demonstrate that AIK can significantly improve OOD detection performance, especially for mixed OOD utterances.

To summarize, the key contributions of the paper are as follows:

- We propose a new task: *multi-label unknown intent detection*, which is desirable for practical dialogue systems.
- We propose a novel and effective method AIK to detect OOD utterances in multi-label setting. By discerning whether all intents contained in the utterance are known, AIK can naturally distinguish IND from pure and mixed OOD utterances.
- We show that AIK outperforms existing methods on two multi-label benchmarks, validating the effectiveness of our method.

2 Task Formulation

Multi-label unknown intent detection breaks the assumption that each utterance only contains one intent, allowing each utterance contain multiple intents. It aims to detect OOD utterances that contain unknown intents.

Formally, given a training dataset $\mathcal{D} = \{(\mathbf{u}^{(i)}, \mathbf{y}^{(i)})\}_{i=1}^{N}$ where $\mathbf{u}^{(i)}$ is an utterance, $\mathbf{y}^{(i)}$ is a set of intent expressed in $\mathbf{u}^{(i)}$, and it belongs to the known intent set $Y_{\text{in}} = \{c_1, c_2, ..., c_k\}$, i.e., $\mathbf{y}^{(i)} \subseteq Y_{\text{in}}$. When testing, given an utterance, we consider it to be OOD if not *all* intents in its intent set $\mathbf{y}$ belong to Y_{in}. Furthermore, if an utterance is OOD and $\mathbf{y} \cap Y_{\text{in}} \neq \varnothing$, i.e., it also contains known intent(s), we call it mixed OOD utterance. If an utterance is OOD and $\mathbf{y} \cap Y_{\text{in}} = \varnothing$, we call it pure OOD utterance. The task goal is to train a score function $S(\mathbf{u})$ based on the training dataset $\mathcal{D}$ to detect OOD utterances (pure and mixed).

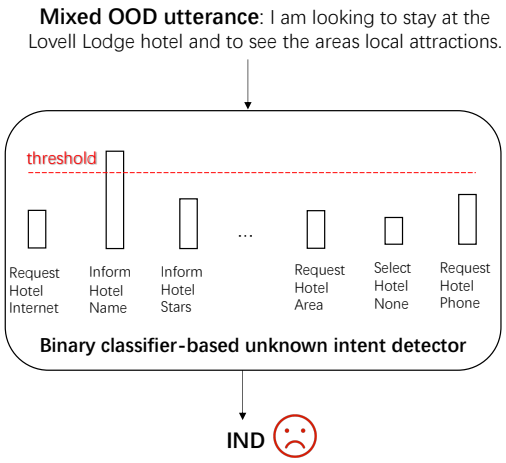

Figure 1: If dialogue systems are equipped with the existing unknown intent detector, such as binary classifier-based detector (Shu et al., 2017), they will misclassify the above utterance as IND as the output of the binary classifier for Inform-Hotel-Name is greater than the threshold.

3 Approach

To perform multi-label unknown intent detection, we propose a novel method AIK. In this section, we first introduce the overall idea of AIK, then present its model architecture and training objective.

3.1 Overall Description

As aforementioned, AIK aims to recognize whether all intents contained in the test utterance are known. Formally, given a test utterance $\mathbf{u}$, we first predict its intent number r. Then we extract its known intent-wise representation $\mathbf{v}_c$ for each known intent $c \in Y_{\text{in}}$, and estimate $\mathbf{v}_c$'s probability density. Suppose that the representations of known intents follow the conditional Gaussian distribution $\mathcal{N}(\boldsymbol{\mu}_c, \boldsymbol{\Sigma})$, where $\boldsymbol{\mu}_c$ is the center vector and $\boldsymbol{\Sigma}$ is the corvariance matrix [2]. The $\mathbf{v}_c$'s probability

[2]For calculation convenience, we assume all known intents share the same covariance matrix, which is also assumed in Yan et al. (2020) and Lee et al. (2018).

Algorithm 1 OOD detection using AIK

Input: Test utterance $\mathbf{u}$; threshold τ; Known intent set Y_{in} and each known intent c's distribution $\mathcal{N}(\boldsymbol{\mu}_c, \boldsymbol{\Sigma})$.

```
Predict u's intent number r
D = {}
for c ∈ Y_in do
    Extract u's known intent-wise representation v_c
    Calculate the Mahalanobis distance d_c between v_c and N(μ_c, Σ)
    Add −d_c into D
end for
S(u) = r-th maximum D
if S(u) < τ then
    return OOD
else
    return IND
end if
```

density can be denoted as $\mathcal{N}(\mathbf{v}_c; \boldsymbol{\mu}_c, \boldsymbol{\Sigma})$, and estimated by its Mahalanobis distance d_c[3] with respect to $\mathcal{N}(\boldsymbol{\mu}_c, \boldsymbol{\Sigma})$ (Murphy, 2022). After calculating the Mahalanobis distance for each known intent-wise representation, we take the negative of them and aggregate them into $\mathbf{D} = \{-d_{c_1}, -d_{c_2}, ..., -d_{c_k}\}$. Finally, we take the r-th maximum $\mathbf{D}$ as $S(\mathbf{u})$ to measure whether the utterance contains r known intents. An utterance with low $S(\mathbf{u})$ (e.g., lower than the pre-defined threshold) indicates its contained known intent number is less than r. Namely, it also contains unknown intent(s), thus can be classified as OOD. We present the pseudo-code of the above process in Algorithm 1, and provide interpretation below.

Interpretation: If an utterance $\mathbf{u}$ contains the known intent c, $\mathbf{v}_c$ will fit the distribution $\mathcal{N}(\boldsymbol{\mu}_c, \boldsymbol{\Sigma})$, $\mathcal{N}(\mathbf{v}_c; \boldsymbol{\mu}_c, \boldsymbol{\Sigma})$ will be large and $-d_c$ will be large, otherwise $-d_c$ will be small (Murphy, 2022). So if $\mathbf{u}$ is IND, i.e., all r intents contained in $\mathbf{u}$ are known intents, there will be r large $-d$ in $\mathbf{D}$, thus the r-th maximum $\mathbf{D}$ should be large. If $\mathbf{u}$ is pure or mixed OOD, i.e., intents contained in $\mathbf{u}$ are not all known intents, there will be less than r large $-d$ in $\mathbf{D}$, thus the r-th maximum $\mathbf{D}$ should be small.

Although AIK is proposed from a multi-label perspective, it has a strong connection with OOD detection methods in multi-class. In multi-class, utterances are assumed to have one intent and methods always take the maximum score, such as maximum softmax probability (Hendrycks and Gimpel, 2017), maximum logit (Shu et al., 2017), to detect OOD utterances, which is equivalent to a special case of AIK that is r equals 1.

[3]Mahalanobis distance d_c can be calculated as: $d_c = (\mathbf{v}_c - \boldsymbol{\mu}_c)^T \boldsymbol{\Sigma}^{-1} (\mathbf{v}_c - \boldsymbol{\mu}_c)$.

3.2 Model Architecture

Figure 2 show the model architecture of AIK.

Utterance encoding. We first employ a pre-trained BERT to encode the utterance $\mathbf{u} = \{w_1, w_2, ..., w_n\}$, where n is the number of tokens. Each token is encoded into a fix-length vector $\mathbf{h}$, and $\mathbf{h}_0$ is the hidden state for [CLS] token. We choose BERT due to its powerful capability of feature extraction. The utterance encoder can also be other models, such as GRU (Chung et al., 2014) or CNN (Kim, 2014).

Intent number prediction. Similar to other sentence-level tasks (Sun et al., 2019), we use $\mathbf{h}_0$ as the overall utterance representation, and predict the intent number of the utterance $\mathbf{u}$:

$$\hat{r} = f_{mlp}(\mathbf{h}_0), \tag{1}$$

where $\hat{r}$ is the predicted intent number, f_{mlp} is a multi-layer perceptron (MLP) network that maps $\mathbf{h}_0$ to a single scalar.

Known intent-wise representation extraction. Inspired by Mullenbach et al. (2018), we utilize a label-wise attention mechanism to get the known intent-wise representations. Specifically, we randomly initialize a trainable query $\mathbf{q}_c$ for each known intent c, and apply the query to calculate attention over hidden states. After that, we aggregate them to get the intent-wise utterance representation $\mathbf{v}_c$ for the intent c:

$$a_t = \frac{\exp(\mathbf{q}_c^T \mathbf{h}_t)}{\sum_{j=1}^{n} \exp(\mathbf{q}_c^T \mathbf{h}_j)}, \tag{2}$$

$$\mathbf{v}_c = \sum_{t=1}^{n} a_t \, \mathbf{h}_t, \tag{3}$$

where exp is the exponential function.

3.3 Training Objective

Intent number loss is mean-squared error (MSE) between the model's predicted intent number and golden intent number:

$$\mathcal{L}_{\text{int}} = \mathbb{E}_{(\mathbf{u},\mathbf{y})\sim\mathcal{D}} (\hat{r}_{\mathbf{u}} - r_{\mathbf{u}})^2, \tag{4}$$

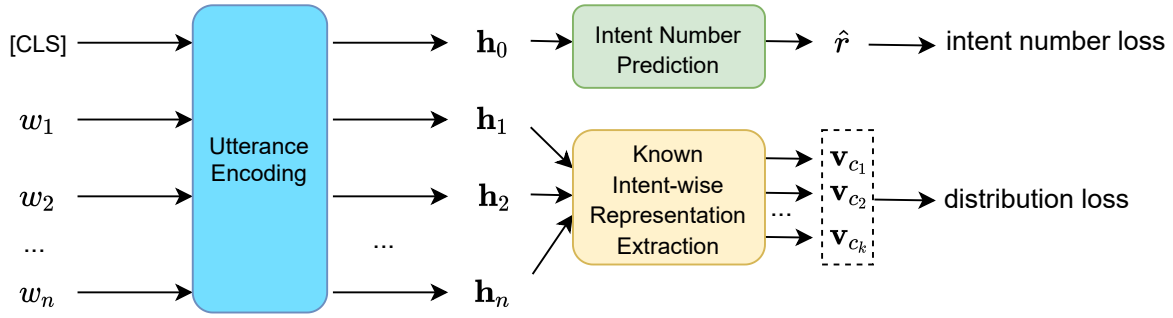

Figure 2: The model architecture of AIK.

where $\hat{r}_{\mathbf{u}}$ is the predicted intent number for utterance $\mathbf{u}$, $r_{\mathbf{u}}$ is the golden intent number, i.e., the size of set $\mathbf{y}$.

Distribution loss drives the known intent-wise representations toward the trainable conditional Gaussian distribution. For known intents contained in the utterance, we maximize the corresponding probability density, i.e., minimize the following loss:

$$\mathcal{L}_{\text{pos}} = -\mathbb{E}_{(\mathbf{u},\mathbf{y})\sim\mathcal{D}}\mathbb{E}_{c\sim\mathbf{y}}\,\mathcal{N}(\mathbf{v}_c;\boldsymbol{\mu}_c,\boldsymbol{\Sigma}). \quad (5)$$

For known intents not contained in the utterance, we make the corresponding probability density less large by setting a margin t:

$$\mathcal{L}_{\text{neg}} = -\mathbb{E}_{(\mathbf{u},\mathbf{y})\sim\mathcal{D}}\mathbb{E}_{c\sim Y_{\text{in}}\setminus\mathbf{y}}\max(0, t-\mathcal{N}(\mathbf{v}_c;\boldsymbol{\mu}_c,\boldsymbol{\Sigma})). \quad (6)$$

Overall Loss: Finally, we train the AIK model by minimizing the following loss:

$$\mathcal{L} = \lambda_1\mathcal{L}_{\text{pos}} + \lambda_2\mathcal{L}_{\text{neg}} + \lambda_3\mathcal{L}_{\text{int}}, \quad (7)$$

where λ_1, λ_2 and λ_3 are loss weights.

4 Experimental Setup

4.1 Datasets

To evaluate the effectiveness of AIK for multi-label unknown intent detection, we build two benchmark datasets from the existing multi-label intent classification datasets MixSNIPS (Qin et al., 2020) and MultiWOZ 2.3 (Han et al., 2020). The construction details are as follows:

- MixSNIPS (Qin et al., 2020) is collected from the SNIPS personal voice assistant (Coucke et al., 2018). We randomly select two intents as unknown intents for the validation set and another two intents as unknown intents for the test set. We conduct experiments with five different splits.

Statistic	MixSNIPS	MultiWOZ 2.3
Train-IND	6998	20319
Validation-IND	389	2531
Validation-OOD	664	2236
Test-IND	398	2530
Test-OOD	671	2418
Test-Mixed OOD	489	64
Test-Pure OOD	182	2354
Number of known intents	3	52
Average intent number per utterance	1.6	1.5

Table 2: Statistics of multi-label unknown intent detection datasets MixSNIPS and MultiWOZ 2.3. Test-Mixed OOD (or Test-Pure OOD) indicates mixed (or pure) OOD utterances in the test set.

- MultiWOZ 2.3 (Han et al., 2020) hosts more than 10K dialogues across eight different domains. For this dataset, we randomly select intents from two domains as unknown intents for the validation set and intents from another two domains as unknown intents for the test set. We also conduct experiments with five different splits.

Table 2 provides average summary statistics of all five splits on two datasets. Note that the training set does not contain OOD utterances.

4.2 Metrics

Similar to multi-class unknown intent detection, we adopt widely used metrics, including AUROC, FPR95, AUPR In, AUPR Out, to measure the performance of different methods in multi-label unknown intent detection.

- AUROC: the area under the true positive rate-false positive rate curve.
- FPR95: The false positive rate(FPR) when the true positive rate(TPR) is 95%. OOD data are treated as positive samples here.
- AUPR In: the area under the precision-recall curve. IND data are treated as positive samples.

- AUPR Out: the area under the precision-recall curve. OOD data are treated as positive samples.

Note that the larger AUROC, AUPR In, AUPR Out mean better performance, and the lower FPR95 indicates better performance.

4.3 Baselines

The multi-label and multi-class unknown intent detection have the same goal, i.e., identifying OOD utterances, thus some competitive OOD detection methods for multi-class can also be chosen as baselines for multi-label. In this work, we compare our AIK method with the generative-based method **Likelihood**, **Likelihood Ratio (LLR)** (Gangal et al., 2020; Ren et al., 2019) and the classifier-based method **Energy** (Ouyang et al., 2021; Liu et al., 2020), **Logit** (Shu et al., 2017), and **LOF** (Lin and Xu, 2019):

- Likelihood trains a language model with IND utterances, and OOD utterances tend to have a lower likelihood.
- LLR trains an extra language model with perturbed utterances to eliminate the unrelated factor in the likelihood for OOD detection.
- Energy uses the sum of exponential of binary classifier output to detect OOD.
- Logit uses the maximum binary classifier output to detect OOD.
- LOF uses local outlier factor (Breunig et al., 2000) in the utterance representation from the binary classifier to detect OOD.

4.4 Implementation Details

The encoder for all classifiers used in the baselines and ours are pre-trained BERT (Devlin et al., 2018). For a fair comparison, we also equip Energy and Logit with the label-wise attention mechanism. We select parameter values based on AUROC on the validation set. For LOF method, we set nearest neighbor number to 20. For LLR method, we follow Gangal et al. (2020), using UNIGRAM to introduce noise and setting p_{noise} to 0.5.

For our AIK method, we simply set λ_1, λ_2, λ_3 to 1, and τ can be set according to FPR95. In the training process, we randomly initialize known intent centers, and set the covariance matrix as identity matrix for reducing the difficulty of optimizing. So we can optimize the probability density by optimizing the Euclidean distance between the known intent-related representation and the known intent centers. The margin t is set to 300 when we optimize the distance. In the testing process, we follow Lee et al. (2018) to compute the empirical center and covariance for known intents as their conditional Gaussian distributions. We use rounding to ensure the predicted intent numbers are integers.

For all methods, we conduct five experiments with different seeds {0, 1, 2, 3, 4} on each split. As each dataset has five splits, we report the average results of 25 experiments.

5 Results and Analysis

5.1 Main Results

Table 3 shows the main results of different methods on multi-label unknown intent detection. From the results, we can observe that:

- AIK can achieve state-of-the-art results on all datasets and metrics. In particular, compared to the best baselines, AIK significantly reduces FPR95 by 15.29% on MixSNIPS dataset and 12.25% on MultiWOZ2.3. Figure 3 further provides the ROC curves of different methods.
- The method Logit and Energy perform poorly on MixSNIPS. The reason is that most OOD utterances in MixSNIPS are mixed, i.e., they also contain known intents, which makes the maximum binary classifier and energy easy to fail. We will discuss more on this in Section 5.2.
- LOF and AIK perform better on MixSNIPS than MultiWOZ 2.3. Note that both methods are based on utterance representation, a good representation space, such as making the representations of utterances with the same intent compact, is critical for them. Considering that the number of known intents is larger on MultiWOZ 2.3 (see Table 2), it is more difficult for the model to learn a good representation space.
- Likelihood and LLR perform stably and obtain appreciable results on two datasets. One bottleneck of such methods is that generative models are difficult to fit the the more complex distribution well for multi-label utterances.

Methods	MixSNIPS				MultiWOZ 2.3			
	AUROC ↑	**FPR95↓**	**AUPR In↑**	**AUPR Out↑**	**AUROC ↑**	**FPR95↓**	**AUPR In↑**	**AUPR Out↑**
Likelihood	87.29	55.12	91.17	83.73	89.52	76.44	89.18	90.73
LLR	89.40	45.54	92.96	82.73	85.90	54.31	85.01	85.71
Energy	68.85	84.66	55.27	78.56	89.25	44.31	88.80	89.35
Logit	67.83	84.16	54.81	77.41	89.44	43.99	89.06	89.47
LOF	92.79	30.73	88.84	95.14	80.68	72.11	78.50	74.56
AIK	**96.29**	**15.44**	**94.93**	**97.46**	**92.22**	**31.74**	**93.33**	**91.01**

Table 3: AUROC, FPR95, AUPR In, AUPR Out on MixSNIPS, MultiWOZ 2.3 datasets. All results are percentages. Best results are in bold. Our method is significantly better than baselines with *p-value* < 0.01 using t-test.

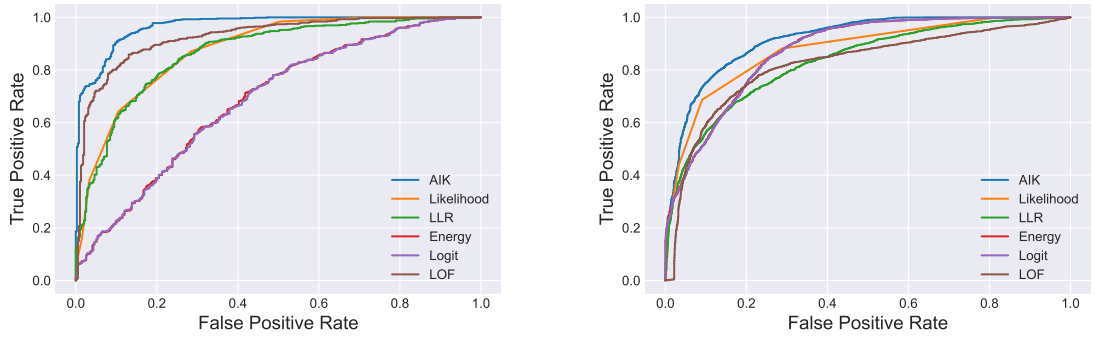

Figure 3: The ROC curves under different methods on MixSNIPS (left) and MultiWOZ 2.3 (right). For the convenience of visualization, we only choose one split for each dataset. The curves indicate AIK always performs better at different thresholds.

Method	**MixSNIPS** Pure/Mixed	**MultiWOZ 2.3** Pure/Mixed
Likelihood	95.52/84.19	89.69/83.98
LLR	93.21/87.96	86.44/67.72
Energy	93.37/59.76	89.55/79.10
Logit	92.47/58.69	89.73/79.25
LOF	93.09/92.66	80.78/75.84
AIK	**96.84/96.06**	**92.34/86.15**

Table 4: AUROC on two type OOD of utterances.

Split	**MixSNIPS** IND/OOD	**MultiWOZ 2.3** IND/OOD
Train	99.46/-	92.79/-
Validation	99.41/85.82	91.83/53.28
Test	99.47/82.70	91.79/53.87

Table 5: Intent number prediction accuracy.

5.2 Performance on Pure and Mixed OOD Utterances

To gain further insights, we measure the OOD performance of all methods in the cases of pure OOD utterances and mixed OOD utterances, respectively.

As shown in Table 4, compared with pure OOD utterances, we observe that all methods obtain lower performance on mixed OOD utterances, which demonstrates mixed OOD utterances are more challenging to detect. Specifically, for Likelihood and LLR, the known intent parts of mixed OOD utterances lie in high-density regions, resulting in a higher likelihood for the whole utterance. For Logit and Energy, the known intent parts will result in a higher binary classifier output. For LOF, the known intent parts will pull the whole utterance representation towards IND utterances, causing it to be misclassified as IND.

For AIK, detecting mixed OOD utterances performs comparably to pure OOD utterances on MixSNIPS. However, the performance gap on MultiWOZ 2.3 still exists. We will reveal that this is caused by the intent number prediction accuracy.

5.3 Analysis for Intent Number Prediction

The performance of AIK is depended on the intent number prediction accuracy. As described in Section 3.1, if an IND utterance has r intent(s), the r-th maximum $\mathbf{D}$ will be large. But once the predicted intent number $\hat{r}$ is greater than r, then $\hat{r}$-th maximum $\mathbf{D}$ might be small, causing the utterance to be misclassified as OOD. Similarly, for an OOD utterance with r intent(s), once the predicted intent number $\hat{r}$ is less than r, then $\hat{r}$-th maximum $\mathbf{D}$ might be large, causing the utterance to be misclassified as IND.

Table 5 shows the intent prediction accuracy of AIK. For IND utterances, we observe that both

Method	MixSNIPS Pure/Mixed	MultiWOZ 2.3 Pure/Mixed
AIK	96.84/96.06	92.34/86.15
AIK with Golden intent number	**96.94/97.26**	**93.66/94.46**

Table 6: Effect of using golden intent number. Values are AUROC.

	$\mathcal{L}_{\text{pos}}$	$\mathcal{L}_{\text{neg}}$	$\mathcal{L}_{\text{int}}$	MixSNIPS Pure/Mixed	MultiWOZ 2.3 Pure/Mixed
1	√			96.22/79.39	92.04/85.25
2	√	√		**97.02**/82.42	**92.86**/85.88
3	√		√	95.14/94.72	91.19/85.83
4	√	√	√	96.84/**96.06**	92.34/**86.15**

Table 7: AUROC results of ablation study of the objective function.

Method	AUROC↑
AIK	92.22
AIK with HM	**94.31**

Table 8: Effect of using HM centers for AIK on MultiWOZ 2.3.

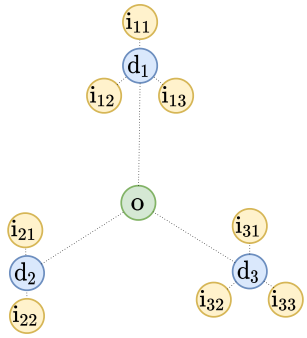

Figure 4: A toy example to show the HM centers. o represents the root node. d_i represents the domain node. i_{ij} represents the intent node from the domain d_i.

MixSNIPS and MultiWOZ 2.3 can reach high accuracy on validation and test set, such as greater than 90%. For OOD utterances, the accuracy is still maintained at a high level on MixSNIPS, while the accuracy is only about 50% on MultiWOZ 2.3. We conjecture that MixSNIPS is constructed manually, and there are some explicit features in utterances, such as "and", to indicate the number of intents. For the more challenging dataset MultiWOZ 2.3, predicting the number of intents is not so easy. Considering that we only take the hidden state of [CLS] to predict, there might be a great potential for improvement. For example, one can explicitly consider some intent number-related features, such as utterance length, number of verbs, etc.

We also test the OOD performance of AIK with the golden intent number. Namely, we use the utterance's golden intent number directly instead of predicting it. Table 6 shows the unknown intent detection performance can be further improved, and more performance improvement can be achieved on the low accuracy dataset MultiWOZ 2.3.

5.4 Ablation Study

We perform an ablation study to investigate the contribution of different losses for AIK. Table 7 shows the corresponding AUROC results.

Effect of $\mathcal{L}_{\text{neg}}$. We can observe that $\mathcal{L}_{\text{neg}}$ brings better results (Line 2 vs. Line 1, Line 3 vs. Line 4). This is because $\mathcal{L}_{\text{pos}}$ is not good at optimizing the inter-class dispersion, i.e., for intents not contained in the utterance, which makes the intent-wise representations low probability density. Ignoring inter-class dispersion might cause the encoder to learn a degenerate solution that all utterances have the same representation, resulting in the indistinguishability of IND and OOD utterances. $\mathcal{L}_{\text{neg}}$ makes up for this by making the corresponding probability density less high.

Effect of $\mathcal{L}_{\text{int}}$. Without intent number loss, we directly take the maximum $\mathbf{D}$ instead of the r-th maximum $\mathbf{D}$ as the score function. We observe that $\mathcal{L}_{\text{int}}$ brings improvement on mixed OOD utterances detection but leads to slight performance degradation on pure OOD utterances (Line 3 vs. Line 1, Line 2 vs. Line 4). For pure OOD utterances, their maximum $\mathbf{D}$ is small as they do not contain any known intent, so maximum $\mathbf{D}$ is enough for detecting them. However, for mixed OOD utterances, their maximum $\mathbf{D}$ is large due to the contained known intents, so using maximum $\mathbf{D}$ would result in these utterances being misclassified as IND.

5.5 Consider the Intent Relation for the Center Initialization

For AIK, at the beginning of the training stage, we initialize each known intent center randomly. However, for the complicated dataset (e.g., MultiWOZ 2.3), randomly initialized centers may ignore the relation between the intents. For example, for intents from the same domain, we might expect their intent centers to be closer.

To achieve this goal, we follow Pang et al. (2020) to preset untrainable hierarchical Max-Mahalanobis (HM) centers for each known intent. HM centers adaptively craft the class centers according to their tree structure. In our scenario, the

Method	AUROC↑
AIK	**96.29**
AIK with logit	95.21

Table 9: Effect of AIK with logit on MixSNIPS dataset.

tree structure is root-domain-intent. Specifically, we first generate the center for the root node (e.g., the origin). Next, we generate centers for each domain node by considering the root. Finally, we generate centers for each intent node by considering its domain node. Figure 4 shows a toy example to illustrate the generated intent centers.

As Table 8 shows, after considering the intent relation, the AIK performance can be further improved. We only consider the domain relation between intents here. It is also very interesting to consider correlation between intents in future work.

5.6 Use Logit to Check Known Intent Number

In our method, we choose to use the probability density, more specifically r-th maximum negative Mahalanobis distance, to check whether the utterance contains the same number of known intents. As an extension, we explore the effectiveness of using the logit. Particularly, we add the intent number prediction to the baseline method Logit, and use the r-th maximum binary classifier output to detect OOD. Ideally, an utterance with small r-th maximum output indicates its contained known intent number is less than its intent number, which can be classified as OOD.

Table 9 shows the performance degradation using logit. This is because logit always suffers from the label-overfitted problem and is not as reliable as probability density (Lee et al., 2018).

6 Related Work

6.1 Unknown Intent Detection

Classifier-based unknown intent detection depends on scores derived from the intent classifier trained with IND utterances and their labels. Hendrycks and Gimpel (2017) propose using the softmax score, which has become a common baseline. Nevertheless, some work (Louizos and Welling, 2017) demonstrates that the softmax score for OOD data can be arbitrarily high. Liu et al. (2020) propose using the energy score because it is theoretically aligned with the density of the input. Ouyang et al. (2021) extends the energy score for unknown intent detection. Some other works attempt to use distance-based scores to detect OOD utterances (Podolskiy et al., 2021; Yan et al., 2020; Lin and Xu, 2019). Although achieving significant results, the mixed OOD utterances cause that the above methods essentially might not generally apply to the more practical multi-label setting. Different from these methods, our method AIK, by recognizing whether all intents are known, is competent for the multi-label setting.

Generative-based unknown intent detection depends on scores derived from the generative model. These methods train the generative model to directly approximate the distribution of IND utterances, then use likelihood or likelihood ratio to detect OOD utterances (Ren et al., 2019; Gangal et al., 2020). These methods are more generalized as they do not rely on utterance labels. However, on more complex multi-label datasets, the generative model might be more challenging to train.

Unknown intent detection with auxiliary OOD utterances depends on scores derived from the model trained with both IND and OOD utterances. Ryu et al. (2018) directly train a discriminator with IND and OOD utterances. Zheng et al. (2019) use OOD utterances to calibrate the softmax score. Ouyang et al. (2021) use OOD utterances to shape the energy gap between IND and OOD utterances. Our method may also use the OOD utterances to further improve the performance by optimizing their probability density less high.

6.2 Multi-label Classification

The multi-label classification task aims to assign multiple non-exclusive labels to each sample. For text, many promising approaches have been proposed to address this problem, such as Binary relevance (Boutell et al., 2004), Classifier chains (Read et al., 2011), seq2seq models (Yang et al., 2018). Multi-label intent classification has also attracted interest recently (Qin et al., 2020; Hou et al., 2021). However, these methods make the closed world assumption, meaning that all classes of the test samples are known. In this paper, we consider an open world environment and detect samples with unknown classes.

7 Conclusion

In this paper, we propose a valuable and practical research task *multi-label unknown intent detection*. It aims to detect OOD utterances that may contain multiple intents. We further propose a novel

AIK method to perform multi-label unknown intent detection, by recognizing whether all intents contained in the utterance are known. Experimental results on two datasets validate the effectiveness of our method. We also analyze the challenges of detecting mixed OOD utterances for multi-label unknown intent detection through experiments.

Acknowledgments

Yawen would like to thank Yuanhang Tang for his constructive suggestions. This work was supported by NSFC Projects (Nos. 61936012 and 61976114).

References

Matthew R Boutell, Jiebo Luo, Xipeng Shen, and Christopher M Brown. 2004. Learning multi-label scene classification. *Pattern recognition*, 37(9):1757–1771.

Markus M Breunig, Hans-Peter Kriegel, Raymond T Ng, and Jörg Sander. 2000. Lof: identifying density-based local outliers. In *Proceedings of the 2000 ACM SIGMOD international conference on Management of data*, pages 93–104.

Junyoung Chung, Caglar Gulcehre, KyungHyun Cho, and Yoshua Bengio. 2014. Empirical evaluation of gated recurrent neural networks on sequence modeling. *arXiv preprint arXiv:1412.3555*.

Alice Coucke, Alaa Saade, Adrien Ball, Théodore Bluche, Alexandre Caulier, David Leroy, Clément Doumouro, Thibault Gisselbrecht, Francesco Caltagirone, Thibaut Lavril, et al. 2018. Snips voice platform: an embedded spoken language understanding system for private-by-design voice interfaces. *arXiv preprint arXiv:1805.10190*.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*.

Rashmi Gangadharaiah and Balakrishnan Narayanaswamy. 2019. Joint multiple intent detection and slot labeling for goal-oriented dialog. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 564–569.

Varun Gangal, Abhinav Arora, Arash Einolghozati, and Sonal Gupta. 2020. Likelihood ratios and generative classifiers for unsupervised out-of-domain detection in task oriented dialog. In *The Thirty-Fourth AAAI Conference on Artificial Intelligence, AAAI 2020, The Thirty-Second Innovative Applications of Artificial Intelligence Conference, IAAI 2020, The Tenth AAAI Symposium on Educational Advances in Artificial Intelligence, EAAI 2020, New York, NY, USA, February 7-12, 2020*, pages 7764–7771. AAAI Press.

Ting Han, Ximing Liu, Ryuichi Takanobu, Yixin Lian, Chongxuan Huang, Dazhen Wan, Wei Peng, and Minlie Huang. 2020. Multiwoz 2.3: A multi-domain task-oriented dialogue dataset enhanced with annotation corrections and co-reference annotation. *arXiv preprint arXiv:2010.05594*.

Dan Hendrycks and Kevin Gimpel. 2017. A baseline for detecting misclassified and out-of-distribution examples in neural networks. *Proceedings of International Conference on Learning Representations*.

Yutai Hou, Yongkui Lai, Yushan Wu, Wanxiang Che, and Ting Liu. 2021. Few-shot learning for multi-label intent detection. In *Thirty-Fifth AAAI Conference on Artificial Intelligence, AAAI 2021, Thirty-Third Conference on Innovative Applications of Artificial Intelligence, IAAI 2021, The Eleventh Symposium on Educational Advances in Artificial Intelligence, EAAI 2021, Virtual Event, February 2-9, 2021*, pages 13036–13044. AAAI Press.

Yoon Kim. 2014. Convolutional neural networks for sentence classification. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1746–1751, Doha, Qatar. Association for Computational Linguistics.

Kimin Lee, Kibok Lee, Honglak Lee, and Jinwoo Shin. 2018. A simple unified framework for detecting out-of-distribution samples and adversarial attacks. In S. Bengio, H. Wallach, H. Larochelle, K. Grauman, N. Cesa-Bianchi, and R. Garnett, editors, *Advances in Neural Information Processing Systems 31*, pages 7167–7177. Curran Associates, Inc.

Ting-En Lin and Hua Xu. 2019. Deep unknown intent detection with margin loss. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 5491–5496, Florence, Italy. Association for Computational Linguistics.

Weitang Liu, Xiaoyun Wang, John Owens, and Yixuan Li. 2020. Energy-based out-of-distribution detection. *Advances in Neural Information Processing Systems*, 33.

Christos Louizos and Max Welling. 2017. Multiplicative normalizing flows for variational bayesian neural networks. In *Proceedings of the 34th International Conference on Machine Learning-Volume 70*, pages 2218–2227. JMLR. org.

James Mullenbach, Sarah Wiegreffe, Jon Duke, Jimeng Sun, and Jacob Eisenstein. 2018. Explainable prediction of medical codes from clinical text. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 1101–1111, New Orleans,

Louisiana. Association for Computational Linguistics.

Kevin P Murphy. 2022. *Probabilistic machine learning: an introduction*. MIT press.

Yawen Ouyang, Jiasheng Ye, Yu Chen, Xinyu Dai, Shujian Huang, and Jiajun Chen. 2021. Energy-based unknown intent detection with data manipulation.

Tianyu Pang, Kun Xu, Yinpeng Dong, Chao Du, Ning Chen, and Jun Zhu. 2020. Rethinking softmax cross-entropy loss for adversarial robustness. In *8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, April 26-30, 2020*. OpenReview.net.

Alexander Podolskiy, Dmitry Lipin, Andrey Bout, Ekaterina Artemova, and Irina Piontkovskaya. 2021. Revisiting mahalanobis distance for transformer-based out-of-domain detection. In *Thirty-Fifth AAAI Conference on Artificial Intelligence, AAAI 2021, Thirty-Third Conference on Innovative Applications of Artificial Intelligence, IAAI 2021, The Eleventh Symposium on Educational Advances in Artificial Intelligence, EAAI 2021, Virtual Event, February 2-9, 2021*, pages 13675–13682. AAAI Press.

Libo Qin, Xiao Xu, Wanxiang Che, and Ting Liu. 2020. AGIF: An adaptive graph-interactive framework for joint multiple intent detection and slot filling. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 1807–1816, Online. Association for Computational Linguistics.

Jesse Read, Bernhard Pfahringer, Geoff Holmes, and Eibe Frank. 2011. Classifier chains for multi-label classification. *Machine learning*, 85(3):333–359.

Jie Ren, Peter J Liu, Emily Fertig, Jasper Snoek, Ryan Poplin, Mark Depristo, Joshua Dillon, and Balaji Lakshminarayanan. 2019. Likelihood ratios for out-of-distribution detection. In *Advances in Neural Information Processing Systems*, pages 14680–14691.

Seonghan Ryu, Sangjun Koo, Hwanjo Yu, and Gary Geunbae Lee. 2018. Out-of-domain detection based on generative adversarial network. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 714–718.

Lei Shu, Hu Xu, and Bing Liu. 2017. DOC: Deep open classification of text documents. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 2911–2916, Copenhagen, Denmark. Association for Computational Linguistics.

Chi Sun, Xipeng Qiu, Yige Xu, and Xuanjing Huang. 2019. How to fine-tune bert for text classification? In *China national conference on Chinese computational linguistics*, pages 194–206. Springer.

Guangfeng Yan, Lu Fan, Qimai Li, Han Liu, Xiaotong Zhang, Xiao-Ming Wu, and Albert Y.S. Lam. 2020. Unknown intent detection using Gaussian mixture model with an application to zero-shot intent classification. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 1050–1060, Online. Association for Computational Linguistics.

Pengcheng Yang, Xu Sun, Wei Li, Shuming Ma, Wei Wu, and Houfeng Wang. 2018. SGM: Sequence generation model for multi-label classification. In *Proceedings of the 27th International Conference on Computational Linguistics*, pages 3915–3926, Santa Fe, New Mexico, USA. Association for Computational Linguistics.

Yinhe Zheng, Guanyi Chen, and Minlie Huang. 2019. Out-of-domain detection for natural language understanding in dialog systems. *arXiv preprint arXiv:1909.03862*.

Pan More Gold from the Sand: Refining Open-domain Dialogue Training with Noisy Self-Retrieval Generation

Yihe Wang[1]*, **Yitong Li**[2,3]†, **Yasheng Wang**[2], **Fei Mi**[2]
Pingyi Zhou[2], **Xin Wang**[1], **Jin Liu**[1]‡, **Xin Jiang**[2], **Qun Liu**[2]
[1]School of Computer Science, Wuhan University
[2]Noah's Ark Lab, Huawei [3]Huawei Technologies Ltd.
{yihewang, jinliu, xinwang0920}@whu.edu.cn
{liyitong3, wangyasheng, feimi2, zhoupingyi, Jiang.Xin, qun.liu}@huawei.com

Abstract

Real human conversation data are complicated, heterogeneous, and noisy, from which building open-domain dialogue systems remains a challenging task. In fact, such dialogue data still contains a wealth of information and knowledge, however, they are not fully explored. In this paper, we show existing open-domain dialogue generation methods that memorize *context-response* paired data with autoregressive or encode-decode language models underutilize the training data. Different from current approaches, using external knowledge, we explore a retrieval-generation training framework that can take advantage of the heterogeneous and noisy training data by considering them as "evidence". In particular, we use BERTScore for retrieval, which gives better qualities of the evidence and generation. Experiments over publicly available datasets demonstrate that our method can help models generate better responses, even such training data are usually impressed as low-quality data. Such performance gain is comparable with those improved by enlarging the training set, even better. We also found that the model performance has a positive correlation with the relevance of the retrieved evidence. Moreover, our method performed well on zero-shot experiments, which indicates that our method can be more robust to real-world data.

1 Introduction

Open-domain dialogue is a long-standing problem in natural language processing and has aroused the widespread interest of researchers. Many approaches have been studied, and recently, generation models trained on large-scale data have gained more attention (Adiwardana et al., 2020; Roller et al., 2020; Xu et al., 2021; Madotto et al., 2021; Bao et al., 2019, 2020; Zhang et al., 2019b; Wang et al., 2020). Open-domain dialogue systems are born to deal with diverse domains, and naturally their training data, usually crawled from online resources such as Reddit and Twitter, are heterogeneous and contain utterances with many various topics, more freedom of topic shifting, and vague responses (Kummerfeld et al., 2018). As a result, directly building generation models from such data will be inefficient and usually requires "knowledge" during the training.

One common solution is to introduce external knowledge, usually, in a form of unstructured knowledge passages from Wikipedia (Dinan et al., 2018) or Internet articles (Komeili et al., 2021), and then, to build retrieval-augmented generation (RAG) methods to improve the response quality (Lewis et al., 2020; Izacard and Grave, 2020). However, this assumes knowledge-intensive scenarios, which are not suitable for general open-domain or robust to noise. According to our preliminary study, in the Reddit dataset, 43% of the dialogues are merely chitchat and cannot match "knowledge". Moreover, building such a knowledge-augmented dataset is very expensive as it relies on large amounts of high-quality human annotations w.r.t. knowledge grounding. And thus, they are limited in size, making it hard for a knowledge-retrieval method to generalize on scale.

Motivated by the above, we would like to investigate *can we have better ways of utilizing open domain data without introducing external resources?* To tackle the aforementioned problem, we found that the context from the other relevant dialogue sessions can still be very useful for dialogue generation. To utilize such unstructured contexts, we take inspiration from retrieval-augmented methods (Lewis et al., 2020). Differently, we retrieve useful dialogue context as evidence, build context-evidence-response triples for each dialogue turn, and treat open-domain generation as an evidence-aware generation task. Such that our model can

*Work done during internship at Noah's Ark Lab, Huawei
†Equal Contribution
‡Corresponding Author

Proceedings of the 29th International Conference on Computational Linguistics, pages 636–647
October 12–17, 2022.

learn to respond with useful grounding evidences. To retrieve evidences, we adopt similarity-based BERTScore (Zhang et al., 2019a), which leverages pre-trained contextual embeddings from BERT and matches words in two sentences by cosine similarity. It has been shown to correlate with human judgment on sentence-level and system-level evaluation. Although it was proposed as an automatic evaluation metric for text generation, due to the high correlation with human judgment, we consider it as a better off-the-shelf method to pick high-relevant evidences, compared with lexicon-based BM25.

By this, we show that current training methods which learn merely using context-response pairs have not fully unleashed the potential of training data and that our methods, only retrieving from the training data, can consistently improve the generation performance. We also perform zero-shot experiments, demonstrating that our method can be robust and generalized to different domains. Moreover, we found that adding extra retrieval data only (without training them) can still help the model gain performance, and it can even outperform traditional methods directly trained on that part of retrieval data. This proves our method is compatible with current methods with external knowledge.

Our contributions are summarized as follows:

- we explore a retrieval-generation training framework that can increase the usage of training data by directly considering the heterogeneous and noisy training data as the "evidence".
- We show that adding extra retrieval data while not training them can still gain performance benefits, even better than traditional training with the retrieval data attached.
- The proposed method performs well on zero-shot experiments, which indicates that our method can generalize well in real-world applications.

2 Related Work

Open-domain Dialogue System Open-domain dialogue system aims to perform chit-chat with people without the task and domain restriction. Adiwardana et al. (2020) proposed Meena, a multi-turn open-domain chatbot trained end-to-end on data mined and filtered from public domain social media conversations. Blender (Roller et al., 2020; Xu et al., 2021) learn to provide engaging talking points and listen to their partners, as well as display knowledge, empathy and personality appropriately, while maintaining a consistent persona. Adapter-bot (Madotto et al., 2021) explored prompt-based few-shot learning in dialogue tasks. Plato (Bao et al., 2019, 2020) introduced discrete latent variables to tackle the inherent one-to-many mapping problem in response generation. Zhang et al. (2019b) proposed DialoGPT which was trained on 147M conversation-like exchanges extracted from Reddit comment chains. Wang et al. (2020) introduced CDial-GPT, a pre-training dialogue model which is trained on a large-scale cleaned Chinese conversation dataset. Mi et al. (2022) built PANGU-BOT with relatively fewer data and computation costs by inheriting valuable language capabilities and knowledge from pre-trained language model.

Retrieval Augmented Generation Retrieval is a long-considered intermediate step in dialogue systems, and recently, it has been an intensively studied topic for neural models(Song et al., 2018; Pandey et al., 2018; Weston et al., 2018; Wu et al., 2019; Cai et al., 2019). Lewis et al. explored a fine-tuning recipe for retrieval-augmented generation, which combined pre-trained parametric and non-parametric memory for language generation. Izacard and Grave proposed Fusion-in-Decoder which encoded each evidence independently with the context when generative model processing retrieved passages. Li et al. (2022) explored how to effectively utilize information with different channel settings of FiD in multi-turn topic driven Conversations. Most of these works retrieved external knowledge, usually unstructured knowledge passages, such as Wizard of Wikipedia (Dinan et al., 2018), persona-chat (Zhang et al., 2018), and Wizard of Internet (Komeili et al., 2021). Moreover, Li et al. (2020) proposed a zero-resource knowledge-grounded dialogue model which bridged a context and a response as knowledge and expressed it as a latent variable.

3 Self-retrieval Method

We start from an open-domain dialogue dataset $\mathcal{D} = \{(c_i, r_i)\}_{i=1}^{N}$, where c_i denotes multi-turn dialogue context, consisting of dialogue utterances, and r_i represents the response.

Generally, we aim to build open-domain dialogue systems that retrieve useful dialogue responses (as evidences) from other sessions to help response generation. To tackle this problem, we

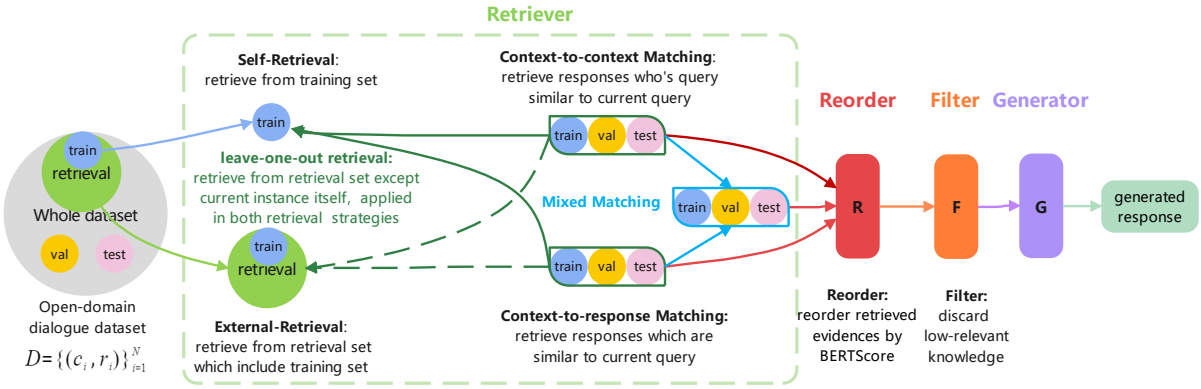

Figure 1: Overview of our self-retrieval approach as well as external-retrieval approach. In self-retrieval, our retriever first retrieves useful dialogue instances from the training dataset, which extends current data to context-evidence-response triples. And then, we adopt evidence-aware training models over the data with self-retrieval evidences.

proposed a two-step framework. The overview of our approach is shown in Figure 1.

1. Firstly, we extend an open-domain dialogue dataset with a *retriever*. Given the context of current dialogue turn c_i, the retriever $\mathscr{R}(e_{\{\cdot\}}|c_i)$ returns top-k relevant evidences as the *evidence set* $\mathcal{E}_i = \{e_{1:k}\}$ from a *retrieval set*. Note that different from existing knowledge-grounding methods, we do not introduce external data for our retriever, and we only consider retrieving evidence from the training data at hand. By that, we extend the dataset into context-evidence-response triples $\mathcal{D} = \{(c_i, \mathcal{E}_i, r_i)\}_{i=1}^{N}$.
2. Secondly, we adopt an evidence-aware generation model, which is a conditional language model to generate the response y given the context and the retrieved evidence $p(y|c, \mathcal{E})$. We investigate two widely used architectures, an auto-regressive GPT, and an encoder-decoder based language model T5.

Next, we introduce how to design an effective retriever in Section 3.1 and ways of implementing evidence-aware generation on the basis of state-of-the-art pre-trained generation models in Section 3.2.

3.1 Retrieve Dialogue Evidence

A variety of retrieval systems have been studied, including classic but effective bag-of-words system (Robertson et al., 1995) and up-to-date dense retriever, such as DPR (Karpukhin et al., 2020) and SparTerm (Bai et al., 2020). We utilized an off-the-shelf similarity based **BERTScore** to retrieve evidence (Zhang et al., 2019a).[1] BERTScore computes token similarity using pre-trained contextual embeddings rather than exact matches, which shows better coherent matching capability compared with human judgment. During the retrieval, for each context-response pair (c_i, r_i), we define the *retrieval set* by applying leave-one-out of the original training set $\mathcal{S} = \mathcal{D} - \{(c_i, r_i)\}$, to ensure the model cannot see the true response during generation.

We explore three retrieval strategies: context-to-context (C2C) retrieval, context-to-response (C2R) retrieval, and a MIX retrieval.

Context-to-context Matching C2C matches the context c_i of current dialogue and the context c_j from the retrieval set $\mathcal{S}$. And the evidence set of c_i is defined as:

$$\mathcal{E}_i^{\text{C2C}}(c_i, \mathcal{S}) = \underset{(c_j, r_j)\in\mathcal{S}}{\text{argmax}_\text{K}}\, \text{score}(c_i, c_j)\,,$$

where argmax_K means selecting top k corresponding responses $r_{1:k}$ as evidences $e_{1:k}$ with best matching score given by BERTSCORE.

Context-to-response Matching As the retrieval set contains the dialogue response, we also perform a Context-to-response (C2R) Matching. It is similar to C2C, while C2R directly matches the response in the retrieval set. In C2R, BERTSCORE computes the matching score based on the response r_j of the retrieval set.

$$\mathcal{E}_i^{\text{C2R}}(c_i, \mathcal{S}) = \underset{(c_j, r_j)\in\mathcal{S}}{\text{argmax}_\text{K}}\, \text{score}(c_i, r_j)\,.$$

Mixed Matching We observed that these two strategies, C2C and C2R, often obtain different results. Therefore, we complement the two retrieval

[1] We also did preliminary experiments over BM25 and it shows no significant differences for our findings.

sets of C2C and C2R with each other and combine them into a MIX retrieval set by re-ranking them using BERTScore. Finally, we take their responses as evidences:

$$\mathcal{E}_i^{\text{MIX}}(u_i, \mathcal{S}) = \text{argmax}_{\text{K}}\{\mathcal{E}_i^{\text{C2C}}, \mathcal{E}_i^{\text{C2R}}\}\,.$$

Filter During preliminary studies, we found that some retrieved evidences are not relevant to the current context. It is arguable that very few relevant evidences can be retrieved for some dialogue instances, and to study this we perform analysis in Section 4.5, where we study different sizes of the retrieval set to ensure more relevant evidences can be found. Undoubtedly, these low-relevant evidences are harmful to response generation. Therefore, we approach a simple filter to discard evidences with very low matching scores.

3.2 Evidence-aware Dialogue Generation

For generating more appropriate responses, our generator is a language model but also conditional on the retrieved evidence set.[2]

$$p(y|c_i, \mathcal{E}_i) = \prod_t p(y_t|c_i, \mathcal{E}_i, y_{<t})\,.$$

Generally speaking, it can be modeled by any auto-regressive or encoder-decoder generation architecture for open-domain dialogue. To demonstrate, we adopt both widely used architectures, i.e. a GPT-2 (Brown et al., 2020) and a Fusion-in-Decoder (FID; Izacard and Grave, 2020).[3]

GPT-2 GPT (Radford et al., 2019; Brown et al., 2020) is auto-regressive language model based on multi-head self-attention transformers (Vaswani et al., 2017). For our task, the model takes the dialogue context and the support evidences as the input, and then it generates the response. More precisely, for any instance $(c_i, \mathcal{E}_i, r_i)$, all retrieved evidences are concatenated before the dialogue context c_i, and the model directly generates the response y after c_i. We add special token `[p]` before each retrieved evidence passage, and following Wang et al. (2020), we add `[speaker1]`, `[speaker2]` to each utterance to indicate different speakers of muti-turn dialogue.

[2]Note that responses from the retrieval set are not directly trained by the language model, but used as the evidences at the input side only.

[3]We also experiment with T5 architectures via concatenating the context and evidences and decoding the response. Yet the performance does not significantly vary from GPT thus we do not report T5 in our main results.

Fusion-in-Decoder In our setups, we have multiple evidences for one instance, thus we adopt a slightly different model than the standard encoder-decoder T5 (Raffel et al., 2020). We use FID (Izacard and Grave, 2020), which was originally proposed for open-domain question answering. It considers encoding each evidence independently with context, so that these evidences will not affect each other on the encoder side, which is a better solution to encode multiple evidences. In detail, FID encodes a concatenation of the context c_i with each retrieved evidence e_j. It concatenates all the encoded hidden representations and then passed to the decoder for response generation. Slightly different from the original architecture, we add an additional passage that only encodes the dialogue context, in case one dialogue does not use any retrieved evidence (discussed in Section 4.5). Similarly, we add special tokens as we did for GPT-2.

4 Experiments

4.1 Datasets

To evaluate the performance of the proposed model, we conduct experiments on two publicly available dialogue datasets.

Reddit Dataset The `Reddit` dataset is extracted from comment chains scraped from Reddit spanning. Reddit discussions can be naturally expanded as tree-structured reply chains, since a thread replying to one thread forms the root node of subsequent threads. We derived the dataset from DialoGPT (Zhang et al., 2019b), and use their script to obtain and process the full dataset or demo dataset.[4] We report results on the demo dataset which comprises 770k multi-turn dialogue instances and is sufficient for our experiments.

Movie Dialog Dataset `Movie dialog` dataset collects movie discussions from real conversation taken directly under the *movie* subreddit (Dodge et al., 2015).[5] We discard instances with long turns or long sentences. In total, the movie dialog dataset has $940k$ dialogue sessions after preprocessing.

For both datasets, we randomly sample a training set of 100k samples, a validation set of 10k samples, and a test set of 10k samples. Data outside the above sets can be considered as retrieval resources. Noted that in our main experiments, the retrieval

[4]https://github.com/microsoft/DialoGPT.

[5]https://research.fb.com/downloads/babi/.

		Automatic Metrics					Human Evaluation			
Reddit		**PPL↓**	**F1↑**	**BLEU↑**	**Dist-1↑**	**Dist-2↑**	**Flue↑**	**Info↑**	**Relv↑**	**SSA↑**
GPT-2 w. SR	BASELINE	31.3	5.3	3.4	65.4	96.7	3.0	2.9	2.8	46%
	BM25 MIX	28.1	6.6	4.2	73.5	**98.2**	3.4	3.3	3.4	51%
	BERTScore C2C	27.7	7.2	4.8	75.2	96.1	3.4	3.4	3.4	54%
	BERTScore C2R	27.9	7.0	4.7	75.0	96.4	3.3	3.4	3.3	53%
	BERTScore MIX	**27.1**	**7.8**	**5.4**	**76.1**	96.8	**3.5**	**3.4**	**3.5**	**55%**
T5 FID w. SR	BASELINE	25.5	5.2	3.7	**95.7**	96.3	3.1	3.0	3.1	48%
	BM25 MIX	23.8	9.5	6.9	95.3	**97.2**	3.5	3.4	3.5	52%
	BERTScore C2C	23.3	9.9	7.3	94.3	94.7	3.5	3.5	3.4	54%
	BERTScore C2R	23.4	9.8	7.2	94.0	94.4	3.5	3.4	3.4	54%
	BERTScore MIX	**22.7**	**10.4**	**7.8**	95.6	96.5	**3.6**	**3.5**	**3.5**	**56%**
Movie		**PPL**	**F1**	**BLEU**	**Dist-1**	**Dist-2**	**Flue**	**Info**	**Relv**	**SSA**
GPT-2 w. SR	BASELINE	25.6	5.4	3.3	64.3	96.0	3.0	2.9	2.8	47%
	BM25 MIX	22.7	6.7	4.2	71.4	96.1	3.4	3.3	3.3	52%
	BERTScore C2C	22.1	7.1	4.7	71.7	94.9	3.4	3.4	3.3	53%
	BERTScore C2R	22.3	7.0	4.7	72.0	94.3	3.3	3.4	3.3	53%
	BERTScore MIX	**21.6**	**7.6**	**5.2**	**73.4**	**96.2**	**3.5**	**3.4**	**3.4**	**55%**
T5 FID w. SR	BASELINE	20.5	5.2	3.7	95.2	95.8	3.1	2.9	2.9	48%
	BM25 MIX	18.9	9.2	6.6	94.9	96.8	3.6	3.5	3.6	53%
	BERTScore C2C	18.4	9.5	7.0	94.4	95.6	3.6	3.6	3.5	55%
	BERTScore C2R	18.5	9.4	6.8	93.8	94.9	3.5	3.5	3.6	54%
	BERTScore MIX	**17.9**	**10.1**	**7.5**	**95.3**	**96.9**	**3.7**	**3.7**	**3.6**	**57%**

Table 1: Automatic and human evaluation of the in-domain setups over Reddit and Movie Dialog, using 8 evidences passages. GPT-2 and T5 are baselines. "w. SR" (with self-retrieval) indicate our methods. The best results are in **bold**.

set (for train/dev/test) is exactly the training set, where we only retrieve from the training set. And experimental results using a larger retrieval set are investigated and reported in Section 4.5, which involves more evidence than the training set.

4.2 Metrics

To evaluate response quality, we adopt both automatic metrics and human evaluations.

Automatic Metrics We deploy four commonly used automatic metrics for the dialogue generation, the perplexity (**PPL**), unigram overlap (**F1**), **BLEU**, and distinct 1,2 (**Dist-1,2**). **F1** and **BLEU** are commonly used to measure how similar the machine-generated responses is to referenced golden response (Miller et al., 2017; Papineni et al., 2002). **Dist-1,2** measure the diversity of the generated responses (Li et al., 2016).

Human Evaluations We perform human evaluation over the generated response. Following Song et al. (2021), we consider three conventional criteria: fluency (**Flue.**), informativeness (**Info.**), and relevance (**Relv.**). We recruit a team on Amazon Mechanical Turk consisting of several professional annotators, who are proficient in language tasks but know nothing about the models.[6] We sample 200 instances for each model's evaluation and each sample was evaluated by three people. Each criterion is rated on five scales, where 1, 3, and 5 indicate unacceptable, moderate, and perfect performance, respectively. We report the average Fleiss's kappa score (Fleiss and Cohen, 1973) on Reddit and Movie Dialogue, 0.49 and 0.45 respectively, indicating annotators have reached moderate agreement. We also consider Sensibleness and Specificity Average (**SSA**), which evaluates two aspects of responses: making sense and being specific (Adiwardana et al., 2020).

4.3 Implementation and Setup

As the context has a different number of turns, we use the latest utterance of dialogue context as the BERTScore query in practice, which can yield more consistent matching scores. Specifically, we compute $F1_{BERT}$ of context c_i of current dialogue and the corresponding context of every evidence. We use DEBERTA-XLARGE-MNLI (He et al., 2020) following the suggestion of authors.[7] The filter is used in all retrieval setups except the baselines.

[6] https://www.mturk.com/

[7] https://github.com/Tiiiger/bert_score

Speaker1: Why do you get to decide who has something to offer ?
Speaker2: He doesn't , he is entitled to his opinion , this is the internet and a forum discussion thread . People post their opinions not the truth .
Baseline Generation: Why have you already voted to make sure you for yourself to support yourself ?
Key Evidence 1: Everyone is entitled to an opinion , but those with experience in the area of discussion usually have more pertinent and accurate opinions than others .
Key Evidence 2: No you're entitled to your opinion . I'd just prefer an opinion that didn't contain a logical fallacy .
Our Generation: I agree with you. People are entitled to their opinion . I just posted my own opinion .
Ground Truth: I know , I was taking a round about way of trying to get him to questions his opinion .

Table 2: Examples of responses generated by baseline and our approach based on FID.

		Movie Dialogue → Reddit					Reddit → Movie Dialogue				
		PPL	**F1**	**BLEU**	**Dist-1**	**Dist-2**	**PPL**	**F1**	**BLEU**	**Dist-1**	**Dist-2**
T5	BASELINE	29.2	5.3	3.9	95.6	96.2	33.0	5.1	3.6	94.5	95.9
FID w. SR	C2C	26.1	9.2	6.8	95.9	97.2	27.3	8.8	6.7	95.8	96.7
	C2R	26.2	9.1	6.6	95.2	96.6	27.5	8.6	6.6	95.2	96.1
	MIX	**25.6**	**9.8**	**7.3**	**96.4**	**98.1**	**26.8**	**9.5**	**7.1**	**95.5**	**97.8**

Table 3: Automatic evaluation results of zero-shot experiments over Reddit and Movie Dialog with 8 retrieved evidence passages. BERTScore is used to retrieve. The best results are in **bold**.

We perform an in-domain evaluation over the two datasets. For each dataset, we adopt the proposed three self-retrieval (SR) method, C2C, C2R, and MIX, comparing against the GPT-2 and FID baselines. We experiments with different numbers of retrieval evidence passages (see Section 4.5). Note that FID degenerates to a standard T5 model without any evidence. We retrain our model based on the pretrained checkpoint of GPT-2,[8] and T5 checkpoint for FID.[9] We do model selection based on PPL over the validation set.

We additionally perform a zero-shot cross-domain evaluation for both datasets using FID.[10] In this setup, we only train our best in-domain FID model on one dataset and then directly test on the other, while the retrieval set for inference is the training set of the target domain. All other setups follow the in-domain experiments.

4.4 Results

In-domain Table 1 reports the overall in-domain experimental results. Overall, our self-retrieval methods achieve better performance consistently across almost all automatic and human evaluation metrics in terms of generating quality. For generation diversities (**Dist-1 and Dist-2**), our SR can still have comparable performance with the strong baselines. For both GPT-2 and FID, all three used matching strategies can improve the overall performance, and MIX consistently outperforms the other two. Comparing with GPT-2 and FID, two baselines achieve similar performance, while when adding our retrieved evidences, we observed FID based methods performance better, demonstrating the effectiveness of evidence-aware training of FID in modeling multiple evidence passages. We also illustrate the example generated by our approach and baselines in Table 2. Above all, these results demonstrate that our approach could utilize more of the dialogue data without introducing more data compared with the baselines.

Zero-shot Cross-domain Table 3 reports the results of zero-shot experiments using FID. Again, we find that our methods with evidence achieve better performance compared to the baselines without knowledge and MIX performs the best. This result indicates that our approach has good generalization and is robust to different datasets.

Overall, both in-domain and zero-shot results demonstrate our self-retrieval method can improve the performance of open-domain dialogue generation, and worth noting that our self-retrieval do not use any additional resources. This indicates our methods can unleash more potential of the dialogue data compared with the vanilla training methods.

4.5 Analysis

Retrieval Methods Table 1 shows the experimental results of different retrieval methods. We find that both methods achieve better results compared to baseline, which shows the generality of our self-

[8] https://huggingface.co/gpt2/tree/main
[9] https://huggingface.co/t5-small/tree/main
[10] We ensure there is no overlap between the two datasets.

		Reddit					Movie Dialog				
		PPL	**F1**	**BLEU**	**Dist-1**	**Dist-2**	**PPL**	**F1**	**BLEU**	**Dist-1**	**Dist-2**
GPT-2	BASELINE	31.3	5.3	3.4	65.4	96.7	25.6	5.4	3.3	64.3	96.0
	p1	28.3	6.9	4.7	74.5	95.8	22.8	6.8	4.6	71.3	94.2
	p2	27.9	7.1	4.9	74.2	95.6	22.5	7.1	4.8	71.6	94.8
SR	p4	27.5	7.4	5.1	75.1	96.3	22.1	7.3	5.0	72.8	95.3
	p8	27.1	7.8	5.4	76.1	96.8	21.6	7.6	5.2	73.4	96.2
	p16	**26.8**	**7.9**	**5.4**	**76.5**	**97.0**	**21.3**	**7.8**	**5.3**	**73.8**	**96.5**
T5	BASELINE	25.5	5.2	3.7	95.7	96.3	20.5	5.2	3.7	95.2	95.8
	p1	23.8	9.5	6.9	93.7	94.8	19.1	9.0	6.3	94.6	95.7
	p2	23.5	9.8	7.2	94.1	95.3	18.7	9.4	6.7	94.4	95.5
FID w. SR	p4	23.1	10.1	7.6	94.6	96.2	18.2	9.8	7.2	94.9	96.3
	p8	22.7	10.4	7.8	95.6	96.5	17.9	10.1	7.5	95.3	96.9
	p16	**22.4**	**10.6**	**7.9**	**95.9**	**98.2**	**17.7**	**10.3**	**7.6**	**95.5**	**97.0**

Table 4: Experimental results of different numbers of evidences used for generation using Reddit and Movie Dialog. p-k indicates the number of evidence passages used for generation. The best results are in **bold**.

Reddit		**PPL**	**F1**	**BLEU**
GPT-2	BASELINE	31.3	5.3	3.4
	RANDOM	31.4	5.4	3.4
w. SR	w/o FILTER	27.6	7.2	4.8
	w. FILTER	**27.1**	**7.8**	**5.4**
FID (T5)	BASELINE	25.5	5.2	3.7
	RANDOM	25.7	5.2	3.6
w. SR	w/o FILTER	23.3	9.8	7.2
	w. FILTER	**22.7**	**10.4**	**7.8**

Table 5: Effectiveness of the Filter.

retrieval method. We can also find that BERTScore performs better than BM25,[11] which indicates that BERTScore could be used to get better retrieval evidences.

Retrieval Strategies Table 1 also shows the experimental results of different retrieval strategies. We find that MIX perform better than context-to-context retrieval (C2C) and context-to-response retrieval (C2R), and the latter two methods show no significant difference. We thought that both C2C and C2R can retrieve useful evidences while from different aspects. And thus mixing them can yield more useful informative and relevant evidences and better performance as well.

Effectiveness of the Filter Table 5 shows the ablation study without using the filter during the retrieval step on Reddit. Here the finding is that experiment with the filter (w. FILTER), has better performance than experiments without it (w/o FILTER), as well as a setup using random evidences (RANDOM). These show that noisy evidences give no assistance, or even harm, to the model and that the necessity of discarding low-relevant evidence in our method.

Number of Retrieved Evidences We also carried out experiments with a different number of retrieved evidences. Table 4 reports the experimental results of using k evidences (p-k) for generation. We observe that experiment using more retrieved evidences (p16) performs better than experiments with fewer retrieved evidences (i.e. p1, p2, p4, p8). While the performance gap is getting smaller when increasing the evidence numbers. Considering the trade-off between efficiency and performance, we report results using 8 evidence as our main results, which is considered to be good enough. These results indicate that we can use more retrieved evidences to obtain better experimental results. In addition, supporting more information is significant for the generative model.

Self-retrieval vs. Extra Evidences We made the retrieval set exactly the same as the training set, denoted as the "self-retrieval (SR)" setup. One natural question is *can we use extra data for retrieval set?* To further understand this question and to validate the usefulness of our method, we carried out experiments with different sizes of the training set and retrieval set. Specifically, we experiment with additional setups by enlarging the retrieval sets, i.e. +200k, +400k, +600k, where "+" means extra data for retrieval sets, and we also adopt baselines with different training sizes of 100k, 300k, 500k, 700k (denoted before "+").[12]

Figure 2 shows the experimental results.[13] We

[11] We only report the mix results for BM25. Refer to the appendix for full results.

[12] Due to data size limitation, we did not occupy all setups.

[13] We also report a detailed results using (100+600k) setup in Appendix A.1.

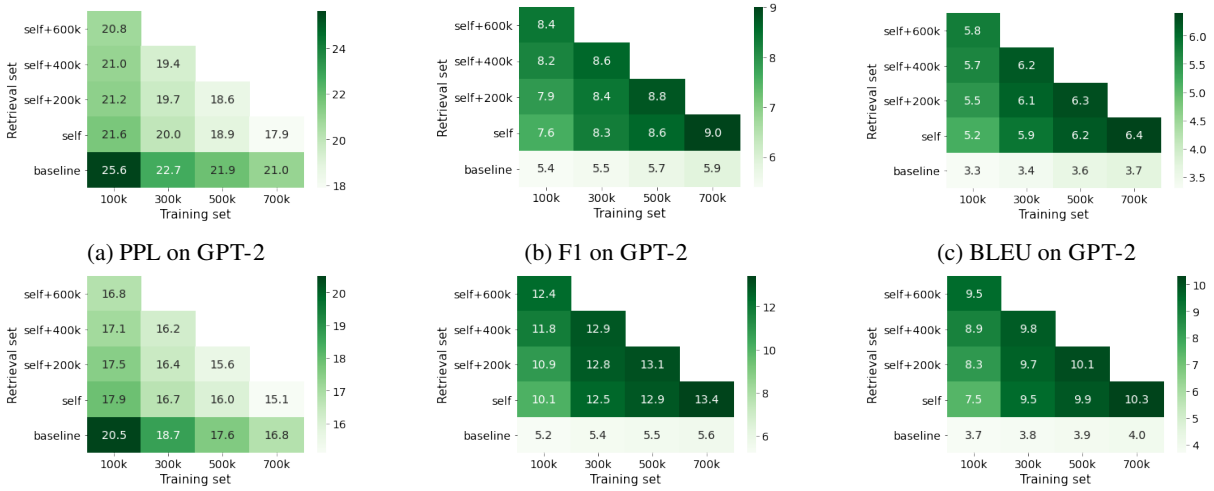

(a) PPL on GPT-2 (b) F1 on GPT-2 (c) BLEU on GPT-2

(d) PPL on FID (e) F1 on FID (f) BLEU on FID

Figure 2: Results of different sizes of training set and retrieval set on the `Movie Dialog` with 8 retrieved evidences. "Self" indicates the training set used for self-retrieval and "+" means adding extra data for retrieval.

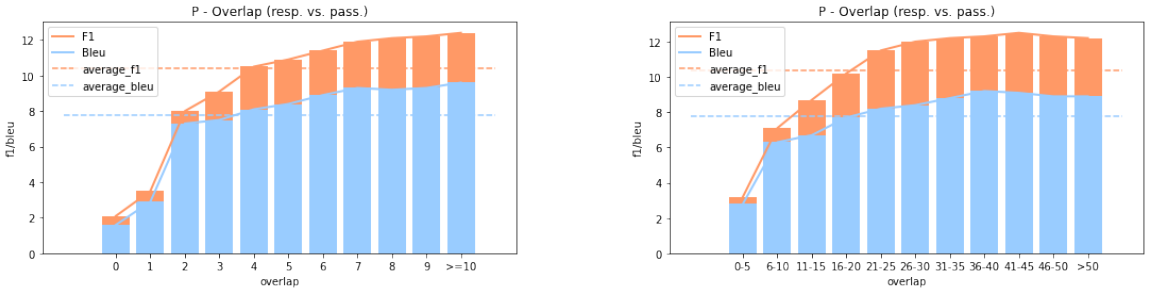

(a) max setup over overlaps with bins = $\{0, \cdots, 9, \geq 10\}$ (b) sum setup over overlaps using bin size = 5

Figure 3: Performance by different overlaps between evidences and ground-truth responses over `Reddit`.

observe that experiments with larger retrieval sets achieve better results than those with small retrieval sets across different training sizes. We believe larger retrieval sets can introduce more relevant evidences, which brings performance gain for the model. Another interesting finding is that adding extra data for retrieval (100+600k, 300+400k, 500+200k) can outperform the baselines (700k) with extra data added via direct training. Also, under the same amount of total data (700k), leveraging more data for retrieval (100+600k, 300+400k, 500+200k) has approaching performance with the self-retrieval with full data (self, 700k). It indicates that our methods can increase the usage of the training data only in a retrieval way without directly training these responses, and our method has good generalization over the retrieval evidences.

Relevance of Evidence and Ground-truth To further study how our methods make sense, we study *how the relevance of the retrieved evidences and ground-truth response can influence the generation performance*. For each instance (c_i, r_i) which used n retrieval evidences $\mathcal{E}_i^{\text{MIX}} = \{e_1, e_2, ..., e_n\}$, we compute the number of overlapped words between the ground-truth r_i and each retrieved evidence. We study two setups by computing the overall $\text{overlap}(\mathcal{E}, r_i)$ using $\max$ and sum over the individual overlaps.

Figure 3 shows the results of these two setups. We observed that higher overlap leads to better performance. It indicates that high relevant retrieval evidences can help to generate better responses and low relevant knowledge are harmful, which is consistent with the findings in Section 4.5. Also, there are low-relevant evidences left in the retrieval step, which indicates that open-domain dialogue generation is still a difficult task, and better retrieval methods are required to further improve our generation performance.

5 Conclusion

In this paper, we propose a self-retrieval training framework for open-domain dialogue generation. Different from other knowledge-intensive tasks, our framework only retrieves relevant dialogue instances from the training data (which can be extended to a retrieval set) without the need to train them in the generation model. It is significant that we demonstrate that traditional training baselines underutilize the training data and our method can utilize more potential of data. We show that our method improves the robustness and generality of generative models as well as generate proper response for complicated human conversation. We also find that BERTScore can be used for better evidence retrieval. In future works, we would like to study better ways of evidence retrieval and evidence-aware training and we believe our approach can benefit to other NLP tasks, such as classification task.

Acknowledgements

The authors would like to thank the anonymous reviewers for their constructive and insightful comments. This work was supported by the National Natural Science Foundation of China under Grants No. 61972290.

References

Daniel Adiwardana, Minh-Thang Luong, David R So, Jamie Hall, Noah Fiedel, Romal Thoppilan, Zi Yang, Apoorv Kulshreshtha, Gaurav Nemade, Yifeng Lu, et al. 2020. Towards a human-like open-domain chatbot. *arXiv preprint arXiv:2001.09977*.

Yang Bai, Xiaoguang Li, Gang Wang, Chaoliang Zhang, Lifeng Shang, Jun Xu, Zhaowei Wang, Fangshan Wang, and Qun Liu. 2020. Sparterm: Learning term-based sparse representation for fast text retrieval. *arXiv preprint arXiv:2010.00768*.

Siqi Bao, Huang He, Fan Wang, Hua Wu, and Haifeng Wang. 2019. Plato: Pre-trained dialogue generation model with discrete latent variable. *arXiv preprint arXiv:1910.07931*.

Siqi Bao, Huang He, Fan Wang, Hua Wu, Haifeng Wang, Wenquan Wu, Zhen Guo, Zhibin Liu, and Xinchao Xu. 2020. Plato-2: Towards building an open-domain chatbot via curriculum learning. *arXiv preprint arXiv:2006.16779*.

Tom B. Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, T. J. Henighan, Rewon Child, Aditya Ramesh, Daniel M. Ziegler, Jeff Wu, Clemens Winter, Christopher Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. 2020. Language models are few-shot learners. *ArXiv*, abs/2005.14165.

Deng Cai, Yan Wang, Wei Bi, Zhaopeng Tu, Xiaojiang Liu, Wai Lam, and Shuming Shi. 2019. Skeleton-to-response: Dialogue generation guided by retrieval memory. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 1219–1228.

Emily Dinan, Stephen Roller, Kurt Shuster, Angela Fan, Michael Auli, and Jason Weston. 2018. Wizard of wikipedia: Knowledge-powered conversational agents. *arXiv preprint arXiv:1811.01241*.

Jesse Dodge, Andreea Gane, Xiang Zhang, Antoine Bordes, Sumit Chopra, Alexander Miller, Arthur Szlam, and Jason Weston. 2015. Evaluating prerequisite qualities for learning end-to-end dialog systems. *arXiv preprint arXiv:1511.06931*.

Joseph L Fleiss and Jacob Cohen. 1973. The equivalence of weighted kappa and the intraclass correlation coefficient as measures of reliability. *Educational and psychological measurement*, 33(3):613–619.

Pengcheng He, Xiaodong Liu, Jianfeng Gao, and Weizhu Chen. 2020. Deberta: Decoding-enhanced bert with disentangled attention. *arXiv preprint arXiv:2006.03654*.

Gautier Izacard and Edouard Grave. 2020. Leveraging passage retrieval with generative models for open domain question answering. *arXiv preprint arXiv:2007.01282*.

Vladimir Karpukhin, Barlas Oğuz, Sewon Min, Patrick Lewis, Ledell Wu, Sergey Edunov, Danqi Chen, and Wen-tau Yih. 2020. Dense passage retrieval for open-domain question answering. *arXiv preprint arXiv:2004.04906*.

Mojtaba Komeili, Kurt Shuster, and Jason Weston. 2021. Internet-augmented dialogue generation. *arXiv preprint arXiv:2107.07566*.

Jonathan K Kummerfeld, Sai R Gouravajhala, Joseph Peper, Vignesh Athreya, Chulaka Gunasekara, Jatin Ganhotra, Siva Sankalp Patel, Lazaros Polymenakos, and Walter S Lasecki. 2018. A large-scale corpus for conversation disentanglement. *arXiv preprint arXiv:1810.11118*.

Patrick Lewis, Ethan Perez, Aleksandra Piktus, Fabio Petroni, Vladimir Karpukhin, Naman Goyal, Heinrich Küttler, Mike Lewis, Wen-tau Yih, Tim Rocktäschel, et al. 2020. Retrieval-augmented generation

for knowledge-intensive nlp tasks. *arXiv preprint arXiv:2005.11401*.

Jiatong Li, Bin He, and Fei Mi. 2022. Exploring effective information utilization in multi-turn topic-driven conversations. *arXiv preprint arXiv:2209.00250*.

Jiwei Li, Michel Galley, Chris Brockett, Jianfeng Gao, and Bill Dolan. 2016. A diversity-promoting objective function for neural conversation models. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 110–119, San Diego, California. Association for Computational Linguistics.

Linxiao Li, Can Xu, Wei Wu, Yufan Zhao, Xueliang Zhao, and Chongyang Tao. 2020. Zero-resource knowledge-grounded dialogue generation. *arXiv preprint arXiv:2008.12918*.

Andrea Madotto, Zhaojiang Lin, Genta Indra Winata, and Pascale Fung. 2021. Few-shot bot: Prompt-based learning for dialogue systems. *arXiv preprint arXiv:2110.08118*.

Fei Mi, Yitong Li, Yulong Zeng, Jingyan Zhou, Yasheng Wang, Chuanfei Xu, Lifeng Shang, Xin Jiang, Shiqi Zhao, and Qun Liu. 2022. Pangubot: Efficient generative dialogue pre-training from pre-trained language model. *arXiv preprint arXiv:2203.17090*.

Alexander H. Miller, Will Feng, Dhruv Batra, Antoine Bordes, Adam Fisch, Jiasen Lu, Devi Parikh, and Jason Weston. 2017. Parlai: A dialog research software platform. In *EMNLP*.

Gaurav Pandey, Danish Contractor, Vineet Kumar, and Sachindra Joshi. 2018. Exemplar encoder-decoder for neural conversation generation. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1329–1338.

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. Bleu: a method for automatic evaluation of machine translation. In *ACL*.

Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, Ilya Sutskever, et al. 2019. Language models are unsupervised multitask learners. *OpenAI blog*, 1(8):9.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J. Liu. 2020. Exploring the limits of transfer learning with a unified text-to-text transformer. *Journal of Machine Learning Research*, 21(140):1–67.

Stephen E Robertson, Steve Walker, Susan Jones, Micheline M Hancock-Beaulieu, Mike Gatford, et al. 1995. Okapi at trec-3. *Nist Special Publication Sp*, 109:109.

Stephen Roller, Emily Dinan, Naman Goyal, Da Ju, Mary Williamson, Yinhan Liu, Jing Xu, Myle Ott, Kurt Shuster, Eric M Smith, et al. 2020. Recipes for building an open-domain chatbot. *arXiv preprint arXiv:2004.13637*.

Haoyu Song, Yan Wang, Kaiyan Zhang, Wei-Nan Zhang, and Ting Liu. 2021. Bob: Bert over bert for training persona-based dialogue models from limited personalized data. *arXiv preprint arXiv:2106.06169*.

Yiping Song, Cheng-Te Li, Jian-Yun Nie, Ming Zhang, Dongyan Zhao, and Rui Yan. 2018. An ensemble of retrieval-based and generation-based human-computer conversation systems.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in neural information processing systems*, pages 5998–6008.

Yida Wang, Pei Ke, Yinhe Zheng, Kaili Huang, Yong Jiang, Xiaoyan Zhu, and Minlie Huang. 2020. A large-scale chinese short-text conversation dataset. In *NLPCC*.

Jason Weston, Emily Dinan, and Alexander Miller. 2018. Retrieve and refine: Improved sequence generation models for dialogue. In *Proceedings of the 2018 EMNLP Workshop SCAI: The 2nd International Workshop on Search-Oriented Conversational AI*, pages 87–92.

Yu Wu, Furu Wei, Shaohan Huang, Yunli Wang, Zhoujun Li, and Ming Zhou. 2019. Response generation by context-aware prototype editing. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, pages 7281–7288.

Jing Xu, Arthur Szlam, and Jason Weston. 2021. Beyond goldfish memory: Long-term open-domain conversation. *arXiv preprint arXiv:2107.07567*.

Saizheng Zhang, Emily Dinan, Jack Urbanek, Arthur Szlam, Douwe Kiela, and Jason Weston. 2018. Personalizing dialogue agents: I have a dog, do you have pets too? *arXiv preprint arXiv:1801.07243*.

Tianyi Zhang, Varsha Kishore, Felix Wu, Kilian Q Weinberger, and Yoav Artzi. 2019a. Bertscore: Evaluating text generation with bert. *arXiv preprint arXiv:1904.09675*.

Yizhe Zhang, Siqi Sun, Michel Galley, Yen-Chun Chen, Chris Brockett, Xiang Gao, Jianfeng Gao, Jingjing Liu, and Bill Dolan. 2019b. Dialogpt: Large-scale generative pre-training for conversational response generation. *arXiv preprint arXiv:1911.00536*.

A Appendix

A.1 Full results of Retrieving Extra data

We present a full results of enlarging the retrieval set to (100+600k) for both `Reddit` and `Movie Dialogue`, shown in Table 6. The training set is 100k as the same as the self-retrieval setup in main results. BM25 is used to retrieve.

`Reddit`		**PPL↓**	**F1↑**	**BLEU↑**
GPT-2	BASELINE	31.3	5.3	3.4
	C2C	28.0	6.2	3.8
GPT-2 w. DR	C2R	28.2	6.0	3.7
	MIX	**26.9**	**6.8**	**4.3**
T5	BASELINE	25.5	5.2	3.7
	C2C	23.6	9.6	7.2
FID w. DR	C2R	23.8	9.4	7.1
	MIX	**21.9**	**12.0**	**9.0**
`Movie`		**PPL↓**	**F1↑**	**BLEU↑**
GPT2	BASELINE	25.6	5.4	3.3
	C2C	22.5	6.0	3.7
GPT-2 w. DR	C2R	22.6	5.9	3.5
	MIX	**21.7**	**7.3**	**4.7**
T5	BASELINE	20.5	5.2	3.7
	C2C	19.2	9.1	6.9
FID w. DR	C2R	19.4	9.0	6.7
	MIX	**17.7**	**11.5**	**8.5**

Table 6: Automatic evaluations of the in-domain setups on the `Reddit` and `Movie Dialog` datasets with 8 evidences for retrieval. The best results are in **bold**.

A.2 Full results of Self-Retrieval

		Automatic Metrics					Human Evaluation			
Reddit		**PPL↓**	**F1↑**	**BLEU↑**	**Dist-1↑**	**Dist-2↑**	**Flue↑**	**Info↑**	**Relv↑**	**SSA↑**
GPT-2	BASELINE	31.3	5.3	3.4	65.4	96.7	3.0	2.9	2.8	46%
w. SR	BM25 C2C	29.4	6.1	3.8	69.3	95.6	3.2	3.0	3.1	49%
	BM25 C2R	29.7	6.0	3.6	68.4	95.3	3.2	3.1	3.1	50%
	BM25 MIX	28.1	6.6	4.2	73.5	**98.2**	3.4	3.3	3.4	51%
	BERTScore C2C	27.7	7.2	4.8	75.2	96.1	3.4	3.4	3.4	54%
	BERTScore C2R	27.9	7.0	4.7	75.0	96.4	3.3	3.4	3.3	53%
	BERTScore MIX	**27.1**	**7.8**	**5.4**	**76.1**	96.8	**3.5**	**3.4**	**3.5**	**55%**
T5	BASELINE	25.5	5.2	3.7	**95.7**	96.3	3.1	3.0	3.1	48%
FID w. SR	BM25 C2C	25.0	8.0	5.9	91.2	93.8	3.3	3.2	3.3	51%
	BM25 C2R	25.2	7.9	5.7	90.4	92.3	3.3	3.2	3.2	50%
	BM25 MIX	23.8	9.5	6.9	95.3	**97.2**	3.5	3.4	3.5	52%
	BERTScore C2C	23.3	9.9	7.3	94.3	94.7	3.5	3.5	3.4	54%
	BERTScore C2R	23.4	9.8	7.2	94.0	94.4	3.5	3.4	3.4	54%
	BERTScore MIX	**22.7**	**10.4**	**7.8**	95.6	96.5	**3.6**	**3.5**	**3.5**	**56%**
Movie		**PPL**	**F1**	**BLEU**	**Dist-1**	**Dist-2**	**Flue**	**Info**	**Relv**	**SSA**
GPT-2	BASELINE	25.6	5.4	3.3	64.3	96.0	3.0	2.9	2.8	47%
w. SR	BM25 C2C	23.5	6.1	3.8	66.9	93.9	3.2	3.1	3.1	51%
	BM25 C2R	23.5	6.0	3.7	67.8	92.7	3.2	3.0	3.1	50%
	BM25 MIX	22.7	6.7	4.2	71.4	96.1	3.4	3.3	3.3	52%
	BERTScore C2C	22.1	7.1	4.7	71.7	94.9	3.4	3.4	3.3	53%
	BERTScore C2R	22.3	7.0	4.7	72.0	94.3	3.3	3.4	3.3	53%
	BERTScore MIX	**21.6**	**7.6**	**5.2**	**73.4**	**96.2**	**3.5**	**3.4**	**3.4**	**55%**
T5	BASELINE	20.5	5.2	3.7	95.2	95.8	3.1	2.9	2.9	48%
FID w. SR	BM25 C2C	20.1	7.7	5.5	92.3	94.1	3.3	3.2	3.2	52%
	BM25 C2R	20.2	7.7	5.4	91.7	93.6	3.3	3.1	3.2	51%
	BM25 MIX	18.9	9.2	6.6	94.9	96.8	3.6	3.5	3.6	53%
	BERTScore C2C	18.4	9.5	7.0	94.4	95.6	3.6	3.6	3.5	55%
	BERTScore C2R	18.5	9.4	6.8	93.8	94.9	3.5	3.5	3.6	54%
	BERTScore MIX	**17.9**	**10.1**	**7.5**	**95.3**	**96.9**	**3.7**	**3.7**	**3.6**	**57%**

Table 7: Automatic and human evaluation of the in-domain setups over Reddit and Movie Dialog, using 8 evidences passages. GPT-2 and T5 are baselines. "w. SR" (with self-retrieval) indicate our methods. The best results are in **bold**.

MulZDG: Multilingual Code-Switching Framework for Zero-shot Dialogue Generation

Yongkang Liu, Shi Feng, Daling Wang*, Yifei Zhang
Northeastern University, Shenyang, China
misonsky@163.com
{fengshi, wangdaling, zhangyifei}@cse.neu.edu.cn

Abstract

Building dialogue generation systems in a zero-shot scenario remains a huge challenge, since the typical zero-shot approaches in dialogue generation rely heavily on large-scale pre-trained language generation models such as GPT-3 and T5. The research on zero-shot dialogue generation without cumbersome language models is limited due to lacking corresponding parallel dialogue corpora. In this paper, we propose a simple but effective **Mul**tilingual learning framework for **Z**ero-shot **D**ialogue **G**eneration (dubbed as MulZDG) that can effectively transfer knowledge from an English corpus with large-scale training samples to a non-English corpus with zero samples. Besides, MulZDG can be viewed as a multilingual data augmentation method to improve the performance of the resource-rich language. First, we construct multilingual code-switching dialogue datasets via translation utterances randomly selected from monolingual English datasets. Then we employ MulZDG to train a unified multilingual dialogue model based on the code-switching datasets. The MulZDG can conduct implicit semantic alignment between different languages. Experiments on DailyDialog and DSTC7 datasets demonstrate that MulZDG not only achieve competitive performance under zero-shot case compared to training with sufficient examples but also greatly improve the performance of the source language.

1 Introduction

The success of neural models and the emergence of large-scale dialogue datasets have greatly advanced the research of dialog generation (Serban et al., 2016, 2017; Huang et al., 2020; Meng et al., 2020). The open-domain dialogue systems aim to generate more informative and fluent responses (Ke et al., 2018; Zhang et al., 2020b; Bao et al., 2020; Meng et al., 2021), which are widely used in various applications such as emotional companionship, mental health support, and social chatbots.

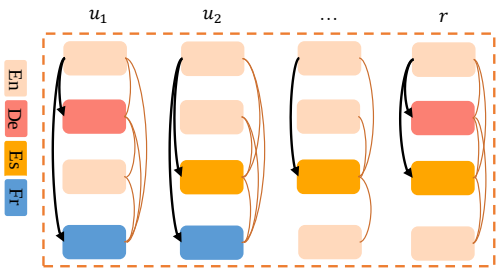

Figure 1: Multilingual semantic alignment map in zero-shot case. One color represents one language. English is the source language and other languages are target languages. u_i represents the i-th utterance in the conversation history and r represents the responses. Black arrows represent translating utterances from English into other languages. Solid brown lines represent implicit semantic alignments between different languages.

Although achieving promising performance, most existing dialogue generation systems (Zhang et al., 2020b; Bao et al., 2020; Li et al., 2020; Floridi and Chiriatti, 2020) rely on a considerable amount of data resource, such as DialoGPT (Zhang et al., 2020b). In practice, the dialogue corpus for many languages is unavailable, which limits the usefulness of dialogue systems for low-resource or even zero-resource languages. Hence it is important to design approaches that can effectively transfer knowledge from the source language with sufficient resources to a target language with zero training samples.

The pre-trained language models have been proved to be very effective in dialogue generations including zero-shot scenarios. Most existing zero-shot dialogue generation approaches usually directly employ large-scale pre-trained generative language models such as GPT-3 (Brown et al., 2020), T5 (Raffel et al., 2020), or conduct secondary pre-training on target language based

*Corresponding author

1
Proceedings of the 29th International Conference on Computational Linguistics, pages 648–659
October 12–17, 2022.

on language models (Ebrahimi and Kann, 2021; Kim et al., 2021). Although these methods can handle the issue of zero-shot generation, the cost of pre-training is unaffordable.

In the similar task of NMT (neural machine translation), zero-shot generation methods are mainly introducing one additional language between source and target language (Johnson et al., 2017; Zheng et al., 2017; Artetxe et al., 2018; Cheng, 2019; Liu et al., 2020a), which can achieve indirect semantic mapping between the source and target language through the additional intermediate language. The nature of translation is the semantic mapping between source and target languages while there is no similar semantic mapping between dialogue history and response. Therefore, directly transferring the zero-shot approaches in NMT to the dialogue generation task is infeasible.

The zero-shot multilingual understanding tasks usually employ code-switching approach to achieve semantic alignment between different languages (Liu et al., 2020b; Chapuis et al., 2021; Qin et al., 2021). The way of code-switching can conduct implicit semantic alignment without relying on parallel corpus pairs. Inspired by these studies, we employ the code-switching method to transfer the knowledge of dialogue history in English to other target languages which have no training examples. We follow previous work (Chapuis et al., 2021) on multilingual representation to employ code-switching at the utterance level, although code-switching at the word or span level is more common (Banerjee et al., 2018; Bawa et al., 2020; Doğruöz et al., 2021).

Based on the code-switching method, we propose a simple but effective **Mul**tilingual learning framework for **Z**ero-shot **D**ialogue **G**eneration (dubbed as MulZDG) that can effectively transfer knowledge from English corpus with large-scale training samples to non-English corpus with zero samples. Specifically, we first construct code-switch languages using the NMT system and bilingual dictionary (Pan et al., 2021). As shown by the black arrows in Figure 1, we randomly select utterances from dialogue history to translate into other target languages. For each target language, we construct code-switching corpus containing source and target languages, respectively. Then we employ MulZDG based on an encoder-decoder structure to train a unified multilingual dialogue generation system, which can be applied in the source language and other target languages with no training samples. MulZDG with a multi-task structure can generate responses with different languages according to specific input prompts. MulZDG can conduct implicit semantic alignments through task sharing mechanism between different languages, as shown by solid brown lines in Figure 1. To summarize, we make the following contributions:

- We propose a simple but effective multilingual framework, MulZDG, which can effectively transfer the knowledge from the source language with large-scale training samples to target languages with zero samples.
- We present a data augmentation method for multilingual code-switching, which can enhance the the performance of source language.
- We construct multilingual code-switching dialogue datasets from English dialogue datasets DailyDialog and DSTC7, and release the multilingual versions datasets[1].

2 RELATED WORK

2.1 Dialogue Generation

Dialogue generation systems aim to produce informative and fluent responses and have attracted great attention in academia. Early studies usually adopt the methods of NMT based on an Encoder-Decoder network to generate responses (Sordoni et al., 2015; Serban et al., 2016). However, these methods often generate dull and generic responses. To tackle this problem, memory mechanism (Wu et al., 2018; Zhang et al., 2020a; Tian et al., 2020) and attention mechanism (Zhang et al., 2019) are introduced into dialogue modeling successively. Although many approaches have been proposed, there are still remarkable gaps between responses generated by neural models and those from humans (Holtzman et al., 2019). Large-scale pre-trained generative models (Zhang et al., 2021; Ling et al., 2021; Wang et al., 2020) have greatly facilitated the development of dialogue generation task. Although achieving promising performance, these methods rely on a large number of training examples, which greatly limits the usability of dialogue systems. In this paper, we propose a multilingual framework that can work in zero-shot case.

2.2 Zero-shot Learning

Zero-shot Learning for dialogue generation tasks refers to building a dialogue generation sys-

[1]https://github.com/misonsky/MultilingualDatasets

tem without available training samples (Floridi and Chiriatti, 2020). Most existing zero-shot methods of dialogue generation rely on large-scale pre-trained generative models (Lewis et al., 2020; Zhang et al., 2020b; Floridi and Chiriatti, 2020), such as GPT-3 (Floridi and Chiriatti, 2020). These methods require huge computing resources, which hinders the usableness of dialogue systems. On similar machine translation tasks, zero-shot methods mainly include triangular and multilingual NMT systems. The triangular NMT systems (Zheng et al., 2017; Cheng, 2019) build a triangular translation case by adding an intermediate language and multilingual systems (Johnson et al., 2017; Liu et al., 2020a) mix multiple different language pairs to achieve semantic alignment between different languages under zero-shot case by building implicit triangular systems. The nature of these methods is the semantic mapping between different languages, which is consistent with the translation task. However, there is no similar semantic mapping between dialogue history and response. In this paper, we propose a simple but effective multilingual dialogue generation framework based on code-switching languages.

3 METHODOLOGY

3.1 Problem Formalization

Given a source language (S), the goal is to build a unified dialogue generation system based on the source language that can be applied to target languages (T). In this paper, the source language S is English. The T include: Zh (Chinese), De (German), Ru (Russian), Es (Spanish), Fr (French) and It (Italian). An instance in dialogue dataset can be represented as (C, R) where $C = \{u_1, u_2, ..., u_n\}$ with n utterance represents the context of dialogue. u_i represents the *i*-th utterance. And R represents the corresponding response. Based on the *S*, we automatically build code-switching languages that include *T* through random translations. The construction process of code-switching languages are shown in Algorithm 1, where $D_T = \{D_{en_{zh}}, D_{en_{de}}, D_{en_{ru}}, D_{en_{es}}, D_{en_{fr}}, D_{en_{it}}\}$. The $D_{en_{tar}}$ represents code-switching languages composed of English and tar.

Algorithm 1: Code-Switching Languages.

Input: Corpus of S: D_S; The target language set T; Bilingual Dictionary: Dic;
Output: Code-switching languages:D_T.

Initialize $D_T = \emptyset$;
foreach (C, R) *in* D_s **do**
  random sample utterances: $U \sim P(C, R)$;
  foreach *target language* t_i *in* T **do**
    Replace tokens for each utterance in U by dictionary $Dic(S \rightarrow t_i)$: $\tilde{U}$;
    Translate $\tilde{U}$ by NMT system: $\hat{U}$;
    Add updated $(C, R))$ by $\hat{U}$ to D_T;

3.2 Multi-task MulZDG Framework

As shown in Figure 2, MulZDG mainly contains three layers: (i) Embedding Layer; (ii) Encoder Layer; (iii) Decoder Layer. The embedding layer is responsible for mapping each word in D_T to a vector space using a pre-trained Glove (Pennington et al., 2014) embedding model. The encoder layer is responsible for capturing the semantic representation of the dialogue context. And the decoder layer is responsible for generating the probability distribution of response for the dialogue context. The decoder in MulZDG can generate responses in different languages through different prompt tokens. The code-switching languages in D_T share the same encoder and decoder, which can conduct semantic alignment between different languages through task sharing mechanism. In the training phase, we employ code-switching languages to train MulZDG. In the testing phase, MulZDG takes monolingual dialogue samples as input and generates responses in the corresponding target language.

MulZDG is a multi-task framework, which supports networks with RNN and transformer as the backbone. The inputs of frameworks are utterance-level code-switching languages. Different inputs in D_T share the same encoder and decoder. The encoder can be hierarchical or non-hierarchical structures. The framework with a non-hierarchical encoder is just like the seq2seq arthitecture (Sutskever et al., 2014) which consists of an encoder RNN and a decoder RNN. The framework with a hierarchical encoder is just like HRED (Serban et al., 2016) which consists of a hierarchical encoder and a decoder.

Encoders can be hierarchical or non-hierarchical. The framework with a non-hierarchical encoder is just like the seq2seq arthitecture (Sutskever et al., 2014) which consists of an encoder and a decoder. The framework with a hierarchical encoder is just

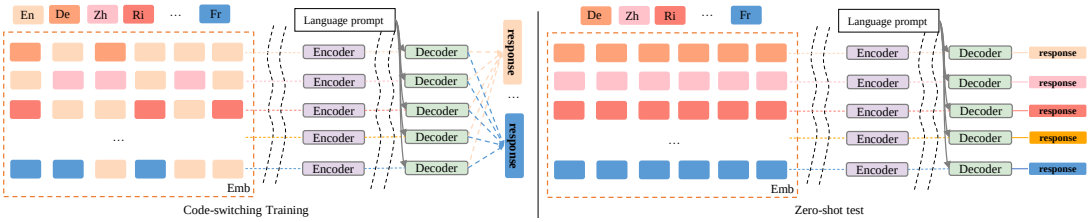

Figure 2: The framework of MulZDG.

like HRED (Serban et al., 2016) which consists of a hierarchical encoder and a decoder. The structure of encoder can be RNN or transformer structure. Next, we briefly introduce both hierarchical and non-hierarchical frameworks.

3.3 Non-hierarchical MulZDG

The non-hierarchical MulZDG is a simple seq2seq structure, which has a shared encoder and a decoder module. The encoder is responsible for encoding the dialogue context and the decoder is responsible for generating the response for the dialogue context. Encoder-Decoder structure based on RNN (Sutskever et al., 2014) and Transformer (Vaswani et al., 2017) are the most classic representative models.

RNN Encoder-Deocder In this paper, we employ GRU (Cho et al., 2014) as the concrete implementation for RNN. We concatenate code-switching language utterances in dialogue context into a consecutive tokens sequence, and add a special start symbol [SOS] and end symbol [EOS] to each tokens sequence. The last hidden state of encoder is denoted as h_ℓ, which is considered as the dialogue context summary. ℓ represents the length of sequence. Finally, we employ another GRU to decode the probability distribution of response.

$$P_\theta(y|h_\ell) = \prod_{j=2}^{m} P_\theta(y_j|y_1, ..., y_{j-1}, h_\ell)P(y_1) \quad (1)$$

where θ represents the parameters of GRU and y represents the response sequence to be decoded.

Transformer Encoder-Deocder Each layer of transformer (Vaswani et al., 2017) encoder is composed of a multi-head self-attention mechanism and a position-wise fully connected feed-forward network. A residual connection (He et al., 2016) is employed around each layer, followed by layer normalization (Ba et al., 2016). In addition to the two sub-layers in each encoder layer, the decoder layer possesses another multi-head attention calculating the weight distribution over the output of the encoder based on predicted sequences. Note that in addition to the word embedding information, position encoding information is required in the embedding layer.

3.4 Hierarchical MulZDG

The encoder of hierarchical MulZDG possesses two basic components: utterance encoder and context encoder, which is like the HRED (Serban et al., 2016) architecture. The utterance encoder is responsible for capturing the features of each utterance and the context encoder is in charge of distilling the dependencies between different utterances. The framework MulZDG supports the implementation of RNNs and transformers structure. In this paper, we employ GRU as the implementation of RNNs.

Hierarchical Transformer Structure Note that positional encoding is required for the implementation of the transformer. The utterance encoder and decoder layer share the same positional encoding. The outputs of utterance encoder based on transformer can be denoted as $u_i = \{e_{sos}, e_1, e_2, ..., e_m, e_{eos}\}$ for i-th utterance, where m represents the length of utterance. We employ an average pooling operation to get a fixed-dimensional representation for every utterance. We add positional encoding to h_i and the context encoder based on transformer is responsible for capturing the global dialogue context information $\hat{H} = \{\hat{h_1}, \hat{h_2}, \cdots, \hat{h_n}\}$. Then the decoder layer decodes the probability distribution of response over $\hat{H}$.

$$h_i = \frac{1}{m}\sum_{j=0}^{m} e_j \quad (2)$$

4 Experiments

4.1 Datasets

In this paper, we select English datasets DailyDialog (Li et al., 2017) and DSTC7 (Galley et al., 2018) as the source languages and generate code-switching training examples based on source train-

ing datasets through Algorithm 1. We directly translate the test datasets into the corresponding target languages. Note that we employ different translation systems (described in Appendix A.1).

DailyDialog is a multi-turn dialogue dataset about our daily life, which consists of 11,118 context-response pairs for training, 1,000 pairs for validation, and 1,000 pairs for testing. The proportion of non-English utterances in code-switching dataset is De-30.186% in $D_{en_{de}}$, Es-30.134% in $D_{en_{es}}$, Fr-30.096% in $D_{en_{fr}}$, It-30.293% in $D_{en_{it}}$, Ru-30.199% in $D_{en_{ru}}$, Zh-30.109% in $D_{en_{zh}}$. In the experiment we abbreviate it as Daily.

DSTC7 is a multi-turn dialogue dataset from social media data, which consists of 76,590 context-response pairs for training, 17,870 pairs for validation, and 1,710 pairs for testing. The proportion of non-English utterances in code-switching dataset is De-27.661% in $D_{en_{de}}$, Es-27.545% in $D_{en_{es}}$, Fr-27.639% in $D_{en_{fr}}$, It-27.611% in $D_{en_{it}}$, Ru-27.786% in $D_{en_{ru}}$, Zh-27.565% in $D_{en_{zh}}$.

Models	Datasets	Types	PPL	BL-1/2	RL	Dist-1/2	Embed A/E/G
HRED	Daily	En	127.7	29.87/24.05	35.54	12.56/44.55	80.55/82.17/64.22
		Aug	122.4	34.47/28.31	39.11	13.33/45.67	82.66/82.88/65.76
	DSTC7	En	116.9	26.73/17.38	29.03	5.34/24.52	78.98/84.78/61.66
		Aug	116.4	27.92/18.77	29.47	7.55/25.16	80.02/84.86/59.66
VHRED	Daily	En	123.3	34.69/25.77	40.77	13.77/45.56	84.74/86.17/69.77
		Aug	123.1	35.93/27.35	41.88	14.57/46.16	86.88/87.94/71.23
	DSTC7	En	127.7	25.54/14.44	25.47	7.49/32.93	77.77/85.68/57.98
		Aug	123.5	27.05/16.92	26.52	8.03/33.74	79.39/86.24/58.17
Trans	Daily	En	143.3	22.86/14.77	28.55	10.36/33.63	79.96/80.06/63.15
		Aug	141.2	23.89/16.17	30.35	11.79/35.64	81.23/81.22/65.45
	DSTC7	En	163.4	22.77/19.74	21.57	6.77/34.56	78.32/82.56/56.89
		Aug	158.9	24.47/23.44	23.67	7.98/35.02	80.19/84.47/56.88
HTrans	Daily	En	146.7	23.76/15.61	27.3	9.96/35.79	80.26/79.44/62.62
		Aug	133.5	24.57/17.03	28.94	11.12/36.98	82.33/82.16/63.76
	DSTC7	En	162.5	23.54/18.68	23.35	7.32/35.66	80.02/82.14/64.33
		Aug	153.3	25.78/20.37	25.76	8.69/37.84	82.43/84.34/65.54

Table 1: Performance of models based on data augmentation using multilingual code-switching of monolingual source. Eng represents that models is only trained on monolingual source language. Aug represents that models is trained on mixed languages including source and multilingual code-switching based on MulZDG. BL stands for BLEU and RL represents Rouge-L. All values are multiplied by 100.

4.2 Baselines

MulZDG is a general multilingual learning framework that can be applied to various dialogue generation models. We select several representative models in this paper.

HRED (Serban et al., 2016) is a hierarchical encoder-decoder structure with a hierarchical encoder (including utterance encoder and context encoder) and a decoder. Hred employs shared encoders to encode each utterance separately.

Models	Datasets	Types	PPL	BL-1/2	RL	Dist-1/2	Embed A/E/G
HRED	Daily	De	125.4	25.89/20.17	35.72	15.63/49.55	78.23/86.86/62.42
		Multi	125.9	23.77/20.33	34.31	15.34/48.89	78.03/86.54/59.97
	DSTC7	De	124.6	21.68/11.89	21.71	7.35/35.23	77.19/87.63/58.41
		Multi	119.7	23.85/12.87	23.77	7.66/33.67	77.37/87.57/60.02
VHRED	Daily	De	121.2	27.14/20.44	34.77	16.53/51.94	77.56/87.26/62.09
		Multi	124.6	25.87/20.67	33.56	15.63/51.32	77.04/87.56/61.42
	DSTC7	De	125.3	22.74/12.07	23.69	7.68/36.29	78.34/87.44/58.62
		Multi	123.4	20.04/11.34	23.56	6.89/36.52	76.88/87.12/57.96
Trans	Daily	De	150.2	20.06/12.38	24.96	9.35/34.56	74.63/78.88/60.76
		Multi	146.7	18.67/11.89	22.65	8.98/33.63	74.53/77.98/59.67
	DSTC7	De	156.2	20.55/11.37	22.76	5.64/32.19	76.54/85.55/55.43
		Multi	156.8	18.77/10.23	20.87	4.88/30.87	75.87/84.71/55.13
HTrans	Daily	De	156.7	21.33/11.66	25.22	8.96/33.19	75.61/77.98/61.86
		Multi	157.8	19.65/10.98	24.33	8.87/33.65	74.78/79.13/59.43
	DSTC7	De	157.7	19.88/11.24	23.45	5.77/33.47	75.96/86.73/56.47
		Multi	161.5	18.78/11.42	21.77	5.33/32.19	75.47/86.33/56.44

Table 2: Zero-shot results of models on German. De represents that models is only trained on German training set. Multi represents that models is only trained on code-switching languages based on MulZDG.

Models	Datasets	Types	PPL	BL-1/2	RL	Dist-1/2	Embed A/E/G
HRED	Daily	Es	124.1	25.49/20.75	36.97	17.68/52.65	77.69/84.21/63.44
		Multi	126.7	26.45/20.55	35.98	17.33/50.56	76.53/83.68/63.56
	DSTC7	Es	119.6	25.39/14.47	26.94	7.58/30.13	77.99/87.32/61.24
		Multi	118.4	25.47/15.76	29.23	7.23/30.67	77.68/86.77/62.13
VHRED	Daily	Es	122.1	29.14/23.04	35.99	18.82/55.37	77.79/85.52/62.28
		Multi	126.7	30.88/22.52	34.76	17.35/54.88	77.23/84.56/62.03
	DSTC7	Es	114.5	26.24/14.42	27.82	8.78/32.45	78.68/87.88/59.77
		Multi	123.3	26.45/17.39	31.19	7.49/32.19	78.97/87.57/63.64
Trans	Daily	Es	157.7	19.06/3.65	22.06	8.35/30.65	73.61/77.67/58.36
		Multi	153.7	18.76/12.98	22.05	8.14/31.48	73.32/76.88/59.54
	DSTC7	Es	123.3	24.38/15.52	26.44	7.53/30.62	77.66/86.48/60.12
		Multi	125.7	24.33/15.43	24.55	7.43/30.88	77.42/86.04/62.33
HTrans	Daily	Es	154.6	18.93/10.57	21.76	8.07/29.33	72.55/75.69/56.38
		Multi	156.3	17.97/9.47	21.62	8.33/27.74	72.76/75.4/56.12
	DSTC7	Es	127.8	23.87/14.33	26.56	7.44/29.48	77.87/85.46/59.56
		Multi	131.2	23.19/12.59	24.89	7.95/29.65	76.46/83.12/57.15

Table 3: Zero-shot results of models on Spanish. Es represents models is trained on monolingual Spanish.

VHRED (Serban et al., 2017) is a hierarchical encoder-decoder structure with a hierarchical encoder (incuding utterance encoder and context encoder) and a decoder based on variational mechanism. VHRED can generate long outputs with better use of contextual information via latent variables.

Transformer (Vaswani et al., 2017) is a encoder-decoder structure with multi-head attention mechanism. The inputs of the Transformer is a consecutive word sequence concatenated all utterances. In all experimental tables, we abbreviate it as **Trans**.

HTransformer (Santra et al., 2021) is a hierarchical encoder-decoder structure with a hierarchical encoder (including utterance encoder and context encoder) and a decoder. Htransformer encodes each utterance separately. In all experimental tables, we abbreviate it as **HTrans**.

4.3 Implementation Details

We implement our MulZDG using Tensorflow 2. The word embedding size and hidden size are all set to 512. We employ Adam optimizer (Kingma and Ba, 2015) to train all models. For the models HRED and VHRED we set the learning rate to 0.001. For the Transform and HTransformer, we set the learning rate to 0.0001. For the models HRED and VHRED, we set the number of encoder and decoder layers to 1. For the Transformer and HTransformer, the number of encoder and decoder layers is 3. And the number HRED for Transformer and HTransformer is 8. We employ the word segmenttation of BERT (Devlin et al., 2019) and the vocabulary of multilingual BERT as the unified vocabulary. The batch size is 128 for HRED and VHRED, and we set it to 512 for Transformer and HTransformer. The maximum epochs are set to 200. We employ GloVe to train a unified multilingual embedding vectors representation for embedding-based metrics based on multilingual corpora. We do not remove unknown tokens when computing embedding-based metrics, and the vectors of all unknown tokens are initialized to zero vector.

Models	Datasets	Types	PPL	BL-1/2	RL	Dist-1/2	Embed A/E/G
HRED	Daily	Fr	122.3	25.36/19.89	35.15	13.59/43.68	80.44/87.12/64.43
		Multi	124.6	25.12/18.78	35.04	13.35/42.45	79.31/87.09/62.21
	DSTC7	Fr	111.3	26.42/16.82	27.18	4.7/21.18	80.84/88.36/62.59
		Multi	116.8	26.54/15.96	25.47	5.43/24.33	80.45/88.31/60.61
VHRED	Daily	Fr	132.2	25.64/19.49	34.97	15.42/45.22	79.86/87.77/63.83
		Multi	135.4	24.96/18.87	32.43	13.88/43.88	79.89/87.57/65.13
	DSTC7	Fr	107.6	28.88/19.36	29.24	5.85/26.22	81.18/88.35/63.8
		Multi	112.4	26.93/18.47	27.56	5.77/26.6	80.39/88.26/61.24
Trans	Daily	Fr	163.7	16.66/12.06	25.3	7.47/29.34	75.66/77.14/59.78
		Multi	166.5	15.76/11.56	24.54	7.23/29.87	76.32/77.65/57.62
	DSTC7	Fr	122.3	25.64/16.47	28.04	5.12/25.63	79.94/87.57/61.25
		Multi	124.2	23.65/14.67	27.33	5.34/24.77	79.54/86.32/60.43
HTrans	Daily	Fr	164.4	16.61/9.49	22.82	7.57/31.98	75.12/78.06/60.08
		Multi	165.4	15.44/8.54	21.65	6.77/31.33	75.54/78.67/60.36
	DSTC7	Fr	125.6	24.87/17.77	26.89	4.77/22.34	79.56/88.45/60.56
		Multi	127.3	24.18/16.28	24.86	4.53/22.54	79.89/87.23/62.65

Table 4: Zero-shot results of models on French. Fr represents models is trained on monolingual French.

4.4 Evaluation metrics

To compare different models, we employ both automatic metrics and human evaluations. **Automatic Metrics**: We employ perplexity (PPL) and distinct 1/2 (Dist.1/2) following previous sdudies (Zhang et al., 2018; Zheng et al., 2020; Song et al., 2021). Lower perplexity means more reliable model. Distinct 1/2 (Li et al., 2016) are the ratio of distinct uni-grams / bi-grams. Higher distinct means better diversity of responses generated by the model. We also employ BLEU (Papineni et al., 2002) and ROUGE-L (Lin, 2004) (abbreviated as RL) for evaluating response generation. BLEU and ROUGE-L metrics evaluate the response based on o-occurrence properties of tokens. Embedding-based metrics (Average, Exterma and Greedy) (Liu et al., 2016; Xu et al., 2018; Sedoc et al., 2019) can reflect the quality of the generated responses at the semantic level. **Human Evaluation**: We further conduct human evaluations to assess the proposed learning framework. We select Chinese and English dialogue systems for human evaluation on DailyDialog and DSTC7. We ask three crowd-sourced graduate students to evaluate the quality of generated responses for 100 randomly sampled input contexts. We request annotators to choose a preferred response, or vote a tie, considering the following aspects of response quality: fluency, informativeness, coherence, and engagingness.

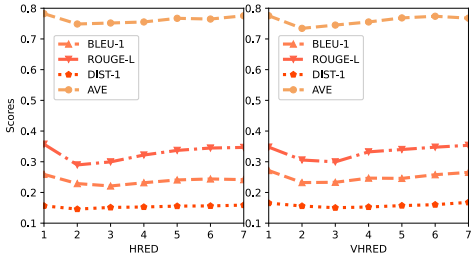

Figure 3: The impact of the number of multilingual code-switching languages on models performance for German. The '1' in abscissa represents that models are trained on monolingual German. Number of languages is greater than '1' represents models is trained on multilingual code-switching languages based on MulZDG.

Models	Datasets	Types	PPL	BL-1/2	RL	Dist-1/2	Embed A/E/G
HRED	Daily	It	126.7	25.51/20.01	35.84	18.07/53.16	76.34/84.54/61.98
		Multi	126.7	25.01/19.23	33.98	18.33/53.25	74.78/84.13/60.3
	DSTC7	It	108.5	24.26/16.11	27.26	7.56/34.53	75.69/87.07/58.73
		Multi	111.2	22.65/14.52	25.54	7.13/32.15	73.85/87.04/56.96
VHRED	Daily	It	123.7	26.99/20.82	34.69	19.64/57.47	76.56/85.38/60.77
		Multi	124.4	25.11/19.14	32.68	17.72/55.89	74.39/84.33/59.31
	DSTC7	It	114.6	28.66/19.04	29.42	8.57/35.12	80.99/88.31/63.96
		Multi	125.4	26.87/18.44	28.49	7.54/35.12	78.54/87.19/63.56
Trans	Daily	It	156.5	18.11/15.72	27.15	8.75/32.34	75.37/79.67/62.08
		Multi	160.5	16.65/14.36	27.42	7.87/31.33	74.36/78.23/61.32
	DSTC7	It	123.3	25.55/17.57	26.44	6.54/32.12	75.09/86.88/57.97
		Multi	124.3	23.55/16.55	24.87	6.12/30.35	75.01/85.32/57.89
HTrans	Daily	It	153.5	19.43/17.56	29.3	9.05/30.34	76.22/80.89/63.28
		Multi	154.7	17.87/18.04	28.88	9.87/30.54	76.86/80.23/62.67
	DSTC7	It	124.5	26.76/18.97	25.67	7.56/34.56	76.11/85.12/57.44
		Multi	123.6	24.88/18.95	24.77	7.45/32.17	76.59/85.07/56.21

Table 5: Zero-shot results of models on Italian. It represents models is trained on monolingual Italian.

4.5 Effectiveness of Data Augmentation

In particular, models based on transformer do not employ pre-trained language model as the initial checkpoint and train from scratch. In addition,

Models	Datasets	Types	PPL	BL-1/2	RL	Dist-1/2	Embed A/E/G
HRED	Daily	Ru	123.8	28.43/22.16	37.51	21.36/53.56	78.07/85.83/64.34
		Multi	127.6	27.57/21.57	36.65	20.54/52.15	76.32/84.84/64.37
	DSTC7	Ru	119.5	28.21/14.25	29.95	10.22/35.21	80.68/86.61/64.17
		Multi	116.5	27.99/18.19	28.93	9.32/35.12	81.38/88.68/63.51
VHRED	Daily	Ru	134.3	27.96/21.88	35.66	21.46/55.88	78.66/85.63/63.22
		Multi	133.2	25.43/20.47	33.23	20.13/53.17	76.98/85.65/63.22
	DSTC7	Ru	103.6	29.49/17.02	34.25	11.15/36.61	79.96/86.13/66.41
		Multi	107.7	27.77/16.87	34.77	10.86/34.65	80.03/86.88/65.74
Trans	Daily	Ru	144.6	21.53/13.77	27.68	9.57/34.93	78.37/81.55/64.5
		Multi	147.9	20.54/13.28	27.43	8.76/33.98	78.54/80.13/62.19
	DSTC7	Ru	123.3	25.66/13.27	27.84	9.46/32.67	78.88/84.55/65.14
		Multi	125.4	24.65/12.09	25.48	8.91/30.54	77.99/84.33/64.87
HTrans	Daily	Ru	143.1	20.77/14.65	28.49	9.68/35.20	78.75/81.33/63.05
		Multi	147.6	18.44/13.76	26.35	9.02/33.89	78.96/79.57/63.06
	DSTC7	Ru	127.8	24.87/14.77	26.46	9.13/31.66	77.88/83.67/65.03
		Multi	128.6	22.76/13.56	24.32	8.25/29.78	77.54/81.98/65.23

Table 6: Zero-shot results of models on Russian. Ru represents models is trained on monolingual Russian.

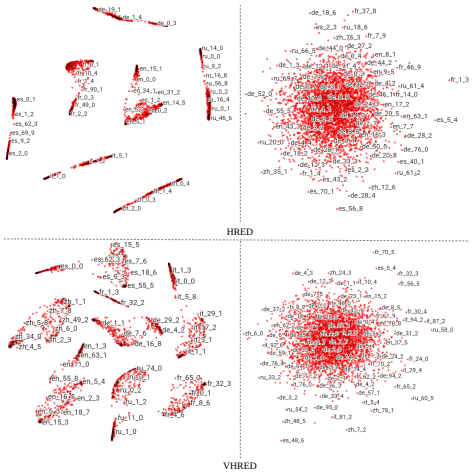

Figure 4: Semantic alignment visualization. The left presents the visualization of utterance vectors for models trained on monolingual language. The right demonstrates the visualization of utterance vectors for models trained on multilingual code-switching languages based on MulZDG. The label lan-N_1-N_2 represents the number of the utterance, where lan stands for language, N_1 stands for the number of data sample and N_2 indicates the utterance number of utterances in dialogue history.

the performance of models based on transformer perform worse than RNN in experiments due to the limited training datasets.

Table 1 reports the results of models on DailyDialog and DSTC7 using monolingual English corpus and data augmentation based on multilingual code-switching languages. We can observe that the performances of models are greatly improved when using data augmentation based on multilingual code-switching. Specifically, the performances of models using data augmentation with multilingual code-switching is 0.2% to 13.2% on PPL, 0.81% to 4.38% higher on BLEU-1, 1.4% to 3.7% higher on BLEU-2, 0.59% to 2.41% higher on Rouge-L, 0.54% to 1.43% higher on dist-1, 0.46% to 2.64% higher on dist-2, and 1.27% to 2.41% higher on average embedding compared with models using monolingual English corpus. The data augmentation approaches with multilingual code-switching can enhance the representation ability of models and improve the quality of the responses through learning common features between different languages.

4.6 Zero-shot Dialogue Generation

Different from Table 1, Table 2 to Table 7 report the results of zero-shot generation using MulZDG based on multilingual code-switching languages. The performances of models trainging on multilingual code-switching languages can achieve competitive results under zero-shot case compared with on corresponding monolingual language. On German DailyDialog and DSTC7, the performances of models under zero-shot case is 0.1% higher on PPL, 1.23% lower on BLEU-1, 0.93% on Rouge-L, 0.71% on dist-1 and 0.51% on average embedding compared with models training on monolingual corpus according to Table 2. On French DailyDialog and DSTC7, the performances of models under zero-shot case is 2.9% lower on PPL, 0.93% lower on BLEU-1, 1.34% on Rouge-L, 0.27% on dist-1 and 0.16% on average embedding compared with models training on monolingual corpus according to Table 4. On Chinese datasets, the performances of models under zero-shot case is 2.1% lower on PPL, 0.91% lower on BLEU-1, 1.57% on Rouge-L, 0.44% on dist-1 and 0.62% on average embedding compared with models training on monolingual corpus according to Table 7.

We can observe the similar results on other languages. MulZDG adopts a multi-task approach to generate responses in different languages. On the one hand, sharing the structure between multiple tasks is benefical for models to exploit the common features between different languages. On the other hand, sharing task mechanism between multilingual code-switching languages is beneficial to enhance the semantic alignment ability of models between different languages.

4.7 Impact of Multilingualization

To explore the effect of multilingual code-switching languages, we conduct further experiments. Figure 3 demonstrates the effect of the number of languages on models performance. We

Models	Datasets	Types	PPL	BL-1/2	RL	Dist-1/2	Embed A/E/G
HRED	Daily	Zh	127.4	23.52/19.52	30.83	9.5/43.37	83.05/83.77/68.42
		Multi	124.3	21.34/18.11	28.33	9.05/42.07	83.56/82.75/66.88
	DSTC7	Zh	108.1	18.07/12.68	26.78	4.06/25.24	85.32/86.27/73.63
		Multi	116.7	17.91/11.55	24.83	4.04/23.81	85.8/86.34/72.29
VHRED	Daily	Zh	125.6	24.54/20.03	31.28	11.22/46.08	83.17/84.43/67.77
		Multi	127.3	24.57/19.09	29.88	9.97/44.13	82.18/84.32/66.49
	DSTC7	Zh	110.4	19.92/13.32	27.61	4.56/26.57	85.79/86.54/72.01
		Multi	116.5	17.98/12.65	26.17	4.36/24.79	84.15/85.32/72.4
Trans	Daily	Zh	165.7	15.55/11.54	24.38	7.89/33.47	79.32/80.46/66.21
		Multi	167.4	15.13/10.33	22.99	6.54/31.44	76.98/78.54/66.21
	DSTC7	Zh	123.4	18.77/12.45	26.58	4.15/24.46	83.56/85.67/71.77
		Multi	123.3	18.04/11.87	25.37	4.54/23.87	84.13/83.51/70.7
HTrans	Daily	Zh	166.8	16.92/11.83	25.66	8.61/34.21	79.92/81.51/66.54
		Multi	167.7	15.66/11.09	24.76	7.96/33.2	78.99/80.23/66.46
	DSTC7	Zh	134.5	19.04/12.44	25.67	4.32/25.67	84.39/85.19/70.65
		Multi	135.4	18.45/11.32	23.87	4.32/25.33	83.79/83.99/69.69

Table 7: Zero-shot results of models on Chinese. Zh represents models is trained on monolingual Chinese.

Models	Languages	Datasets					
		Daily (%)			DSTC7 (%)		
		Win	Tie	Loss	Win	Tie	Loss
HRED	En	30	44	26	33	36	31
	Fr	23	56	21	23	54	23
	Zh	31	35	34	32	33	35
VHRED	En	38	32	30	30	45	25
	Fr	16	63	21	11	82	7
	Zh	36	29	35	30	37	33

Table 8: Human evaluation on DailyDialog and DSTC7 (multilingual VS monolingual). On English corpus, we compare the the performances of models training using data augmentation of multilingual code-switching with monolingual English corpus. On other languages, we compare the performances of models under zero-shot using multilingual code-switching languages with corresponding monolingual corpus.

select German as the target language (other languages are available), HRED and VHRED as the tested models. The performances of bilingual code-switching languages on HRED and VHRED are dramatically lower than models trained on the corresponding monolingual training set. However, the performances of models are gradually improved with the number of languages increases. We can only add up to seven languages due to the limitations of our constructed corpus. We can conclude that MulZDG fails to work well in zero-shot case when the number of languages is small according to 3. More constructive conclusions require further experimental evaluation in the case of more languages in the future, such as hundreds of languages.

4.8 Multilingual Mechanism Analysis

We conduct extensive experiments to explore how the multilingual mechanism works. We select 1,000 multilingual parallel examples from DailyDialog, about 4,536 utterances in total and visualize the representations of these examples in HRED and VHRED based on MulZDG trained on different monolingual languages and multilingual code-switchinig languages, respectively. Figure 4 presents the results of the utterances representation visualization on HRED and VHRED. The representations of utterances based on different monolingual languages are clustered into different language categories while representations of utterances on multilingual code-switchinig languages are clustered together according to semantics. Utterances in different languages expressing same semantics will be clustered together, which demonstrates that MulZDG will do semantic alignment between different languages. This phenomenon presents that MulZDG pays attention to the common features between different languages.

4.9 Human Evaluation

Although automatic evaluation metrics have been shown to be reliable, we still conduct human evaluations to confirm the validity of the MulZDG. We compare the performance of models training on multilingual code-switching languages and on corresponding monolingual language. Table 8 reports the results of human evaluation of HRED and VHRED on three languages (i.e., English, French and Chinese). We can observe that the performances of models using data augmentation with multilingual code-switching is on average 6.0% higher on DailyDialog and 3.5% higher on DSTC7 than using monolingual English corpus. Besides, the performance of models under zero-shot is on average 5% lower on DailyDialog and 2% lower on DSTC7 than using coresponding monolingual corpus. These results demonstrate that multilingual code-switching framework can not only be considered as a data augmentation method but also be employed to zero-shot dialogue generation.

5 Conclusion and Future Work

In this paper, we propose a simple but effective multilingual framework: MulZDG, which can not only be used as an approach of data augmentation but also be used to zero-shot dialogue generaton. Besides, we release the multilingual versions of DailyDialog and DSTC7 datasets. In the future, we will explore the working mechanism and effect of large-scale multilingual code-switching languages (e.g., hundreds of languages) in the problem of zero-shot dialogue generation.

6 Acknowledgement

We would like to thank the reviewers for their constructive comments. The project is supported by the National Natural Science Foundation of China (61872074, 62172086, 62106039, 62272092).

References

Mikel Artetxe, Gorka Labaka, and Eneko Agirre. 2018. Unsupervised statistical machine translation. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 3632–3642.

Jimmy Lei Ba, Jamie Ryan Kiros, and Geoffrey E Hinton. 2016. Layer normalization. *arXiv preprint arXiv:1607.06450*.

Suman Banerjee, Nikita Moghe, Siddhartha Arora, and Mitesh M Khapra. 2018. A dataset for building code-mixed goal oriented conversation systems. In *Proceedings of the 27th International Conference on Computational Linguistics*, pages 3766–3780.

Siqi Bao, Huang He, Fan Wang, Hua Wu, and Haifeng Wang. 2020. Plato: Pre-trained dialogue generation model with discrete latent variable. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 85–96.

Anshul Bawa, Pranav Khadpe, Pratik Joshi, Kalika Bali, and Monojit Choudhury. 2020. Do multilingual users prefer chat-bots that code-mix? let's nudge and find out! *Proceedings of the ACM on Human-Computer Interaction*, 4(CSCW1):1–23.

Piotr Bojanowski, Édouard Grave, Armand Joulin, and Tomáš Mikolov. 2017. Enriching word vectors with subword information. *Transactions of the Association for Computational Linguistics*, 5:135–146.

Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. 2020. Language models are few-shot learners. *Advances in neural information processing systems*, 33:1877–1901.

Emile Chapuis, Pierre Colombo, Matthieu Labeau, and Chloe Clavel. 2021. Code-switched inspired losses for generic spoken dialog representations. *arXiv preprint arXiv:2108.12465*.

Yong Cheng. 2019. Joint training for pivot-based neural machine translation. In *Joint Training for Neural Machine Translation*, pages 41–54. Springer.

Kyunghyun Cho, Bart Van Merriënboer, Caglar Gulcehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. 2014. Learning phrase representations using rnn encoder–decoder for statistical machine translation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1724–1734.

Alexis Conneau, Guillaume Lample, Marc'Aurelio Ranzato, Ludovic Denoyer, and Hervé Jégou. 2017. Word translation without parallel data. *International Conference on Learning Representations (ICLR)*.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. Bert: Pre-training of deep bidirectional transformers for language understanding. In *NAACL-HLT (1)*.

A Seza Doğruöz, Sunayana Sitaram, Barbara Bullock, and Almeida Jacqueline Toribio. 2021. A survey of code-switching: Linguistic and social perspectives for language technologies. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 1654–1666.

Abteen Ebrahimi and Katharina Kann. 2021. How to adapt your pretrained multilingual model to 1600 languages. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 4555–4567.

Luciano Floridi and Massimo Chiriatti. 2020. Gpt-3: Its nature, scope, limits, and consequences. *Minds and Machines*, 30(4):681–694.

Michel Galley, Chris Brockett, Xiang Gao, Bill Dolan, and Jianfeng Gao. 2018. End-to-end conversation modeling: Moving beyond chitchat.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. 2016. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778.

Ari Holtzman, Jan Buys, Li Du, Maxwell Forbes, and Yejin Choi. 2019. The curious case of neural text degeneration. In *International Conference on Learning Representations*.

Minlie Huang, Xiaoyan Zhu, and Jianfeng Gao. 2020. Challenges in building intelligent open-domain dialog systems. *ACM Transactions on Information Systems (TOIS)*, 38(3):1–32.

Melvin Johnson, Mike Schuster, Quoc V Le, Maxim Krikun, Yonghui Wu, Zhifeng Chen, Nikhil Thorat, Fernanda Viégas, Martin Wattenberg, Greg Corrado, et al. 2017. Google's multilingual neural machine translation system: Enabling zero-shot translation. *Transactions of the Association for Computational Linguistics*, 5:339–351.

Pei Ke, Jian Guan, Minlie Huang, and Xiaoyan Zhu. 2018. Generating informative responses with controlled sentence function. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1499–1508.

San Kim, Jin Yea Jang, Minyoung Jung, and Saim Shin. 2021. A model of cross-lingual knowledge-grounded response generation for open-domain dialogue systems. In *Findings of the Association for Computational Linguistics: EMNLP 2021*, pages 352–365.

Diederik P Kingma and Jimmy Ba. 2015. Adam: A method for stochastic optimization.

Guillaume Lample, Alexis Conneau, Ludovic Denoyer, and Marc'Aurelio Ranzato. Unsupervised machine translation using monolingual corpora only. In *International Conference on Learning Representations (ICLR), publisher = OpenReview.net, year = 2018,*.

Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Veselin Stoyanov, and Luke Zettlemoyer. 2020. Bart: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7871–7880.

Chunyuan Li, Xiang Gao, Yuan Li, Baolin Peng, Xiujun Li, Yizhe Zhang, and Jianfeng Gao. 2020. Optimus: Organizing sentences via pre-trained modeling of a latent space. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 4678–4699.

Jiwei Li, Michel Galley, Chris Brockett, Jianfeng Gao, and William B Dolan. 2016. A diversity-promoting objective function for neural conversation models. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 110–119.

Yanran Li, Hui Su, Xiaoyu Shen, Wenjie Li, Ziqiang Cao, and Shuzi Niu. 2017. Dailydialog: A manually labelled multi-turn dialogue dataset. In *Proceedings of the Eighth International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 986–995.

Chin-Yew Lin. 2004. Rouge: A package for automatic evaluation of summaries. In *Text summarization branches out*, pages 74–81.

Yanxiang Ling, Fei Cai, Xuejun Hu, Jun Liu, Wanyu Chen, and Honghui Chen. 2021. Context-controlled topic-aware neural response generation for open-domain dialog systems. *Information Processing & Management*, 58(1):102392.

Chia-Wei Liu, Ryan Lowe, Iulian V Serban, Michael Noseworthy, Laurent Charlin, and Joelle Pineau. 2016. How not to evaluate your dialogue system: An empirical study of unsupervised evaluation metrics for dialogue response generation. *arXiv preprint arXiv:1603.08023*.

Yinhan Liu, Jiatao Gu, Naman Goyal, Xian Li, Sergey Edunov, Marjan Ghazvininejad, Mike Lewis, and Luke Zettlemoyer. 2020a. Multilingual denoising pre-training for neural machine translation. *Transactions of the Association for Computational Linguistics*, 8:726–742.

Zihan Liu, Genta Indra Winata, Zhaojiang Lin, Peng Xu, and Pascale Fung. 2020b. Attention-informed mixed-language training for zero-shot cross-lingual task-oriented dialogue systems. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pages 8433–8440.

Chuan Meng, Pengjie Ren, Zhumin Chen, Zhaochun Ren, Tengxiao Xi, and Maarten de Rijke. 2021. Initiative-aware self-supervised learning for knowledge-grounded conversations. In *Proceedings of the 44th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 522–532.

Chuan Meng, Pengjie Ren, Zhumin Chen, Weiwei Sun, Zhaochun Ren, Zhaopeng Tu, and Maarten de Rijke. 2020. Dukenet: A dual knowledge interaction network for knowledge-grounded conversation. In *Proceedings of the 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 1151–1160.

Xiao Pan, Mingxuan Wang, Liwei Wu, and Lei Li. 2021. Contrastive learning for many-to-many multilingual neural machine translation. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 244–258.

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. Bleu: a method for automatic evaluation of machine translation. In *Proceedings of the 40th annual meeting of the Association for Computational Linguistics*, pages 311–318.

Jeffrey Pennington, Richard Socher, and Christopher D Manning. 2014. Glove: Global vectors for word representation. In *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*, pages 1532–1543.

Libo Qin, Minheng Ni, Yue Zhang, and Wanxiang Che. 2021. Cosda-ml: Multi-lingual code-switching data augmentation for zero-shot cross-lingual nlp. In *Proceedings of the Twenty-Ninth International Joint Conference on Artificial Intelligence*.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J Liu. 2020. Exploring the limits of transfer learning with a unified text-to-text transformer. *Journal of Machine Learning Research*, 21:1–67.

Bishal Santra, Potnuru Anusha, and Pawan Goyal. 2021. Hierarchical transformer for task oriented dialog systems. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 5649–5658.

Joao Sedoc, Daphne Ippolito, Arun Kirubarajan, Jai Thirani, Lyle Ungar, and Chris Callison-Burch. 2019. Chateval: A tool for chatbot evaluation. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics (Demonstrations)*, pages 60–65.

Iulian Serban, Alessandro Sordoni, Yoshua Bengio, Aaron Courville, and Joelle Pineau. 2016. Building end-to-end dialogue systems using generative hierarchical neural network models. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 30.

Iulian Serban, Alessandro Sordoni, Ryan Lowe, Laurent Charlin, Joelle Pineau, Aaron Courville, and Yoshua Bengio. 2017. A hierarchical latent variable encoder-decoder model for generating dialogues. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 31.

Haoyu Song, Yan Wang, Kaiyan Zhang, Weinan Zhang, and Ting Liu. 2021. Bob: Bert over bert for training persona-based dialogue models from limited personalized data. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 167–177.

Alessandro Sordoni, Yoshua Bengio, Hossein Vahabi, Christina Lioma, Jakob Grue Simonsen, and Jian-Yun Nie. 2015. A hierarchical recurrent encoder-decoder for generative context-aware query suggestion. In *proceedings of the 24th ACM international on conference on information and knowledge management*, pages 553–562.

Ilya Sutskever, Oriol Vinyals, and Quoc V Le. 2014. Sequence to sequence learning with neural networks. In *Advances in neural information processing systems*, pages 3104–3112.

Zhiliang Tian, Wei Bi, Dongkyu Lee, Lanqing Xue, Yiping Song, Xiaojiang Liu, and Nevin L Zhang. 2020. Response-anticipated memory for on-demand knowledge integration in response generation. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 650–659.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in neural information processing systems*, pages 5998–6008.

Yiru Wang, Pengda Si, Zeyang Lei, and Yujiu Yang. 2020. Topic enhanced controllable cvae for dialogue generation (student abstract). In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pages 13955–13956.

Chien-Sheng Wu, Richard Socher, and Caiming Xiong. 2018. Global-to-local memory pointer networks for task-oriented dialogue. In *International Conference on Learning Representations*.

Xinnuo Xu, Ondřej Dušek, Ioannis Konstas, and Verena Rieser. 2018. Better conversations by modeling, filtering, and optimizing for coherence and diversity. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 3981–3991.

Bowen Zhang, Xiaofei Xu, Xutao Li, Yunming Ye, Xiaojun Chen, and Zhongjie Wang. 2020a. A memory network based end-to-end personalized task-oriented dialogue generation. *Knowledge-Based Systems*, 207:106398.

Hainan Zhang, Yanyan Lan, Liang Pang, Hongshen Chen, Zhuoye Ding, and Dawei Yin. 2021. Modeling topical relevance for multi-turn dialogue generation. In *Proceedings of the Twenty-Ninth International Conference on International Joint Conferences on Artificial Intelligence*, pages 3737–3743.

Hainan Zhang, Yanyan Lan, Liang Pang, Jiafeng Guo, and Xueqi Cheng. 2019. Recosa: Detecting the relevant contexts with self-attention for multi-turn dialogue generation. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 3721–3730.

Saizheng Zhang, Emily Dinan, Jack Urbanek, Arthur Szlam, Douwe Kiela, and Jason Weston. 2018. Personalizing dialogue agents: I have a dog, do you have pets too? In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 2204–2213.

Yizhe Zhang, Siqi Sun, Michel Galley, Yen-Chun Chen, Chris Brockett, Xiang Gao, Jianfeng Gao, Jingjing Liu, and Bill Dolan. 2020b. Dialogpt: Large-scale generative pre-training for conversational response generation. In *ACL (demo)*.

Hao Zheng, Yong Cheng, and Yang Liu. 2017. Maximum expected likelihood estimation for zero-resource neural machine translation. In *International Joint Conference on Artificial Intelligence*.

Yinhe Zheng, Rongsheng Zhang, Minlie Huang, and Xiaoxi Mao. 2020. A pre-training based personalized dialogue generation model with persona-sparse data. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pages 9693–9700.

A Appendix

A.1 NMT Systems

We simulate distribution differences between training and test datasets through using different translation systems. The NMT systems employed for the training set are *Helsinki-NLP/opus-mt-en-tar*[2]. 'tar' represents the target languages, which includes Chinese, German, Russian, Spanish, French and Italian. And we employ T5-base[3] to translate the test

[2]https://huggingface.co/Helsinki-NLP/

[3]https://huggingface.co/t5-base

Models	Types	Responses
HRED	En	I like travelling. like my experience and i have a good idea.
	Aug	I like travelling. I travel frequently.
VHRED	En	i like sports. I like travelling.
	Aug	I like travelling. I'll do it next time.
Gold Response		Yes, I like travelling. I am young, and unmarried. It's no problem for me to travel frequently.

Table 9: Case study for data augmentation on Daily dataset. The English context is "*What are your personal weaknesses? I'm afraid I'm a poor talker. I'm not comfortable talking with the people whom I have just met for the first time. That is not very good for business, so I have been studying public speaking. Are you more of a leader or a follower? I don't try to lead people. I'd rather cooperate with everybody, and get the job done by working together. Do you think you can make yourself easily understood in English? Yes, in most circumstances. Are you available for travel?*"

datasets into German and French. For Russian, Chinese, Spanish and Italian we employ WMT19-en-ru[4], WMT-en-zh[5], mbart-en-es[6] and Google NMT system, respectively.

A.2 Dictionaries

(Lample et al.) employs FastText (Bojanowski et al., 2017) methods on the source and target monolingual corpora to train word embeddings and then applies the unsupervised method (Conneau et al., 2017) to infer a bilingual dictionary. (Lample et al.) provides 110 bilingual dictionaries which can be used for word-by-word translation. According to statistics, there are 101,997 De-En word pairs, 112,583 Es-En word pairs, 113,324 Fr-En word pairs, 103,613 It-En word pairs, 48,714 Ru-En word pairs and 21,597 Zh-En word pairs.

A.3 Case Study

An case study is provided in Table 9 to demonstrate the values of augmented data. We can observe that the responses of HRED and VHRD contain context-independent information without using data augmentation. Specifically, "like my experience and i have a good idea" in HRED and "i like sports" in VHRED are context independent. The responses generated by HRED and VHRED are more informative and more coherent when using data augmentation. Models can utilize cross linguistic knowledge to generate more informative and coherent responses by multilingual data augmentation.

[4]https://huggingface.co/facebook/wmt19-en-ru

[5]https://huggingface.co/liam168/trans-opus-mt-en-zh

[6]https://huggingface.co/mrm8488/mbart-large-finetuned-opus-en-es-translation

Target-Guided Open-Domain Conversation Planning

Yosuke Kishinami[1] **Reina Akama**[1,2] **Shiki Sato**[1] **Ryoko Tokuhisa**[1]
Jun Suzuki[1,2] **Kentaro Inui**[1,2]
[1]Tohoku University [2]RIKEN
yosuke.kishinami.q8@dc.tohoku.ac.jp
{akama,shiki.sato.d1,tokuhisa,jun.suzuki,inui}@tohoku.ac.jp

Abstract

Prior studies addressing target-oriented conversational tasks lack a crucial notion that has been intensively studied in the context of goal-oriented artificial intelligence agents, namely, *planning*. In this study, we propose the task of Target-Guided Open-Domain Conversation Planning (TGCP) task to evaluate whether neural conversational agents have goal-oriented conversation planning abilities. Using the TGCP task, we investigate the conversation planning abilities of existing retrieval models and recent strong generative models. The experimental results reveal the challenges facing current technology.

1 Introduction

Neural conversational agents have achieved great successes in recent years, and various methods have been proposed to generate informative responses, e.g., the use of knowledge (Zhao et al., 2020; Wu et al., 2020), personality (Li et al., 2016; Zhang et al., 2018), emotional considerations (Rashkin et al., 2019; Zhong et al., 2020), and large-scale models (Zhang et al., 2020; Adiwardana et al., 2020; Roller et al., 2021; Thoppilan et al., 2022). One hot topic in this research area is to develop proactive behavior in agents. For example, Tang et al. (2019) proposed the task of Target-Guided Open-Domain Conversation, in which an agent is required to actively lead a conversation to a predefined target word. Wu et al. (2019) proposed a task that uses a knowledge graph to actively lead a conversation to a target entity. Several studies have implemented these target-oriented task settings (Dai et al., 2019; Qin et al., 2020; Yuan and An, 2020; Zhong et al., 2021; Zhu et al., 2021). However, these prior studies all lack *planning*, a crucial notion that has been intensively studied in the context of goal-oriented artificial intelligence (AI) agents (Norvig and Russell, 1995; Kuijpers and Dockx, 1998; Stent et al., 2004; Walker et al., 2007, etc.) and has also been introduced in neural conversational agents (Botea et al., 2019; Jiang et al., 2019a,b). In other words, these studies do not explicitly consider the generation of a multiple-step plan to achieve a target.

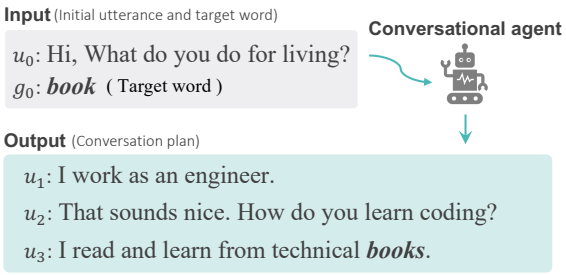

Figure 1: Overview of the TGCP task.

Given this background, in this study, we propose the **Target-Guided Open-Domain Conversation Planning** (henceforth, **TGCP**) task such that an agent's planning ability in goal-oriented conversations can be assessed. The TGCP task is to produce a plan that leads a conversation to a given target, as illustrated in Figure 1. The point is to consider the task of producing a conversation plan for several utterances ahead, which we first address in the aforementioned context of Target-Guided Open-Domain Conversation. Furthermore, we also propose modeling the planning process by simulating the user's succeeding utterances using the model of the agent itself; namely, the agent converses with itself (i.e., self-conversation) to search for potential conversation paths that achieve the goal. This task setting is not the same as a real-world setting, in which an agent is required to plan a conversation while uncertain of the user's future utterances. However, planning in the self-conversation setting can be considered a prerequisite capability for a planning-aware goal-oriented conversational agent. TGCP works as a framework to evaluate an agents' prerequisite ability for conversation planning without employing human subjects; this can abstract

Proceedings of the 29th International Conference on Computational Linguistics, pages 660–668
October 12–17, 2022.

away the hard-to-control human factors from experiments (e.g., some human subjects may not be as cooperative as others).

This paper has three major contributions: (1) We propose the TGCP task as a framework to assess the prerequisite ability of a model for goal-oriented conversation planning. (2) We conduct a set of experiments on the TGCP framework using several existing retrieval-based neural models and recently proposed strong generative neural models of conversational agents. (3) Our experimental results reveal the challenges facing current technology. The evaluation codes and the test set used in the experiments are available.[1]

2 Target-Guided Open-Domain Conversation Planning

We introduce the task of Target-Guided Open-Domain Conversation Planning, the TGCP task for short, that is to evaluate whether the agents have goal-oriented conversation planning abilities. In this section, we describe the task definition and the evaluation metrics.

2.1 Task definition

Figure 1 shows an overview of the TGCP task. We define the goal given to the agents as a word (e.g., *dog*, *pizza*, *coffee*). Given a target word g_0 and an initial utterance u_0, TGCP requires agents to make an entire conversation plan $(u_1, \ldots, u_N)$, whose last utterance u_N, which consists of M words, contains the target word g_0, namely, $u_N = (w_{N,1}, \ldots, w_{N,M})$, and $w_{N,m} = g_0$ for any $m \in M$. This task has the same input/output format as the human-agent conversation task proposed by Tang et al. (2019). However, these task setups differ in terms of whether or not a human conversational partner is involved. In TGCP, agents generate for all utterances in the entire conversation.

2.2 Evaluation metrics

The evaluation is performed based on three objectives: whether the target word is mentioned (**achievement ratio**), whether the utterance transitions in the conversation are natural (**transition smoothness**), and how likely the conversation is to actually occur (**conversation probability**). We believe that satisfying these three perspectives is important in goal-oriented conversation planning. For example, given a target word *computer* and an initial utterance "What sports do you like?," the utterance like *I love computer.* achieves the target, but it is not natural as a conversation, and such an interaction rarely occurs. Likewise, the utterance like "*I don't like sports because my friend who likes sports broke my computer.*" is a natural transition and achieves the target, but it is likely to rarely occur in an actual conversation. We believe that an agent's generation of such utterances does not indicate the agent's planning ability. We can automatically calculate the achievement ratio based on whether the target word itself is mentioned or not.[2] We also consider transition smoothness and conversation probability to be manually evaluated.[3]

3 Experiments

Using the proposed TGCP, we investigate the conversation planning ability of several major existing dialogue models and recent deep neural network (DNN) based dialogue models.

3.1 TGCP settings

Dataset. As a dataset for the TGCP task, we prepared $1,000$ pairs consisting of an initial utterance and a target word, i.e., (u_0, g_0). We created these pairs by randomly extracting from a set of the first utterances and a set of keywords extracted from subsequent utterances in the ConvAI2 dataset.[4]

Evaluation. As described in Section 2.2, in TGCP, the conversation plans generated by models are evaluated by target achievement ratio, transition smoothness, and conversation probability. The target achievement ratio was calculated automatically. To avoid infinite conversations that never reached the target, we set the maximum number of turns to 8.[5] For transition smoothness and conversation probability, we evaluated them manually using Amazon Mechanical Turk.[6] For each model, randomly sampled 100 conversation plans were rated by native English speakers. We eliminated

[1] The evaluation codes and the test set are available at https://github.com/y-kishinami/TGCP

[2] Tang et al. (2019) considers mentioning synonyms as the task achievement; however, Zhong et al. (2021) points out that synonyms are unreliable to measure the task achievement. Implementation details are provided in Appendix A.1.

[3] Empirical analyses on the relationship between these two metrics are provided in Appendix A.2.

[4] This follows existing analogous work (Tang et al., 2019; Qin et al., 2020; Zhong et al., 2021). In addition, we removed the keywords not covered by ConceptNet.

[5] The same setting as previous studies (Tang et al., 2019; Qin et al., 2020; Zhong et al., 2021).

[6] https://www.mturk.com/

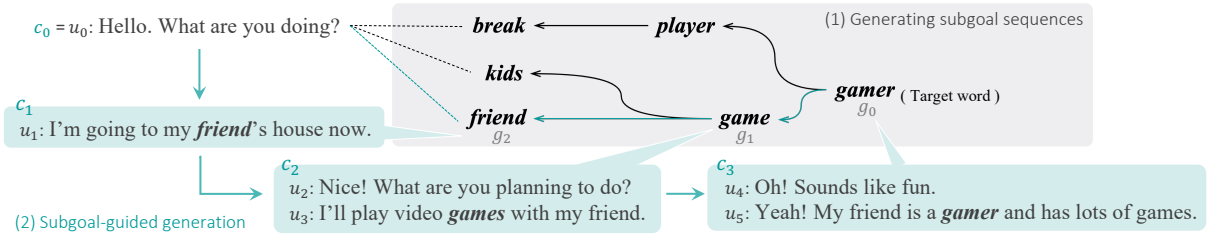

Figure 2: Subgoal-guided conversation plan generation with BLENER+PREDES..

low-quality workers using attention checks. Five workers rated each conversation on a five-point Likert scale for transition smoothness (5 is *Strongly good* and 1 is *Strongly bad*) and conversation probability (5 is *Frequently* and 1 is *Rarely*).[7]

3.2 Existing models

We prepared the following seven existing dialogue models employed in Target-Guided Open-Domain Conversation: Wu et al. (2017)'s RETRIEVAL, Tang et al. (2019)'s RETRIEVAL-ST., PMI, NEURAL, and KERNEL, Qin et al. (2020)'s DKRN, and (Zhong et al., 2021)'s CKC.[8] All models except RETRIEVAL are retrieval dialogue models that infer the keyword to mention immediately after each turn of the conversation on the fly and then determine the next response based on it and the conversational history. RETRIEVAL is a retrieval dialogue model that does not infer the keyword but determine the next response only based on conversational history.

3.3 Recent DNN-based models

In addition, we prepared the latest generative model that combines the DNN-based powerful dialogue model, BLENDER (Roller et al., 2021), and a novel strategy for pre-designing keyword sequences to given g_0 (PREDES.). Our BLENDER+PREDES. (Figure 2) is a newly designed model. Note that this model is new because the task is new, and that there should be many ways to design models for TGCP. Still, we believe that testing the performance of a specific model such as BLENDER+PREDES. on TGCP can help investigate the nature of TGCP. In our BLENDER+PREDES., we first generated the keyword sequences, hereinafter it called *subgoal sequence*, using ConceptNet5 (Speer et al., 2017). Specifically, we acquired the series of n concepts that are passed when tracing the edges of the knowledge graph from the concept representing the target word g_0 to the concept related to the initial utterance u_0 as a subgoal sequence $G = g_0, g_1, \cdots, g_{n-1}$. n is the length of the subgoal sequence including the target word g_0. This allows preventing cases that cannot get closer to the target than a certain point because of selecting a locally optimal solution. After generating the subgoal sequence, we generated a sequence of partial conversations $C = c_0, c_1, \cdots, c_n$ using BLENDER as follows:

$$c_i = f(g_{n-i}, (c_0, \cdots, c_{i-1})) \quad (1 \leq i \leq n) \quad (1)$$

Where, c_i denotes a partial conversation that follows the previous partial conversation c_{i-1} and ends up with the utterance where the subgoal g_{n-i} appears. $f(\cdot)$ is a function that returns a partial conversation to the given previous conversations and a subgoal.

PREDES. settings. We set $n = 3$, i.e., we generated subgoal sequences by tracing ConceptNet up to three levels.[9,10] Among the subgoal sequences generated from ConceptNet, we retained the 30 subgoal sequences in which the end of the subgoal sequence g_{n-1} was the most related to the given first utterance u_0. We calculated the relatedness as the cosine similarity between the SIF embedding (Arora et al., 2017) of u_0 and GloVe word vector (Pennington et al., 2014) of g_{n-1}. Among the conversation plans generated from the 30 subgoal sequences, we selected the conversation plan with the highest average probability of generating partial conversations by BLENDER as the final output.

Training of BLENDER. We used the Blenderbot 3B implemented by huggingface transformer

[7] Concrete instructions are provided in Appendix C.
[8] Implementations details are given in Appendix B.1.
[9] We excluded all stopwords in the NLTK and spaCy libraries to comprehensively exclude unnecessary words. In addition, we excluded the concepts for which the score calculated by wordfreq (Speer et al., 2018) was lower than the score of the target word.
[10] We empirically confirmed the validity of this setting (Appendix B.3).

Model	Subgoal	Conversation	Achievement	#Turns	Smoothness	Probability
RETRIEVAL (Wu et al., 2017)	-	retrieval	0.034	3.71	3.52	3.37
RETRIEVAL-ST. (Tang et al., 2019)	on-the-fly	retrieval	0.851	5.04	3.50	3.29
PMI (Tang et al., 2019)	on-the-fly	retrieval	0.531	4.97	3.33	3.17
NEURAL (Tang et al., 2019)	on-the-fly	retrieval	0.535	2.83	3.14	3.00
KERNEL (Tang et al., 2019)	on-the-fly	retrieval	0.596	2.79	3.24	3.05
DKRN (Qin et al., 2020)	on-the-fly	retrieval	**0.968**	2.91	3.28	3.12
CKC (Zhong et al., 2021)	on-the-fly	retrieval	0.353	3.60	2.81	2.69
BLENDER (Roller et al., 2021)	-	generative	0.024	5.04	3.99	**3.90**
BLENDER+CKC	on-the-fly	generative	0.247	7.00	3.90	3.71
BLENDER+PREDES.	pre-design	generative	0.425	6.29	**4.05**	**3.90**
Human	-	-	1.000	3.50	4.11	3.89

Table 1: Performance of dialogue models on the TGCP task.

as BLENDER.[11] Because BLENDER is a model that generates a response based on the conversational history, we fine-tuned it to use as f which generates a partial conversation based on previous partial conversations and a subgoal. we used the ConvAI2 processed by Zhong et al. (2021) as a training data for BLENDER. We prepared the training data by randomly splitting a single conversation into an input and an output consisting of multiple utterances and then concatenating a word extracted randomly from the output-side utterances (i.e., keywords) to the input utterances.[12] We finally obtained 117,877 pairs as training set and 6,425 pairs as validation set. The hyperparameters are provided in Appendix B.2.

3.4 Ablation models

To analyze the effectiveness of the pre-design strategy, we also prepared Blender without any conversational strategy (BLENDER), and with an on-the-fly strategy using existing models. Specifically, we employed the strategy of CKC as the comparison on-the-fly strategy, which is known to be the highest performance method in TGC (Zhong et al., 2021) (BLENDER+CKC). For both models, Blender is the same as BLENDER+PREDES.. But note that blender, without any conversation strategy, does not concatenate keywords with inputs for training and inference.

3.5 Results

Table 1 shows the evaluation results on TGCP. To provide the human upper bound performance, we also had three workers perform TGCP on 50 pairs randomly selected from the dataset described in Section 3.1 (Human).

Achievement ratio. The achievement ratios of the retrieval models tended to be high. In particular, the achievement ratio of DKRN was comparable to that of humans. The generative models had lower achievement ratios. However, BLENDER+PREDES. improved the achievement ratio compared with BLENDER+CKC, whose subgoal strategy is the same as that of CKC. This result means that replacing the on-the-fly subgoal strategy with the pre-design strategy is effective in improving the target achievement ratios of generative models.

Smoothness & probability. The retrieval models have lower values of transition smoothness and conversation probability than humans. Table 2 shows a conversation plan example generated by DKRN, whose achievement ratio was the highest of all the compared models. In the example, the transition between u_1 and u_2 is clearly unnatural, although the model achieved to mention the target word.[13] The transition smoothness and conversation probability of the generative models were higher than those of the retrieval models. In particular, BLENDER+CKC significantly outperformed CKC in these metrics. Therefore, using powerful DNN-based generation models improves the transition smoothness and conversation probability of the conversation plans. Table 3 shows a conversation plan example generated by BLENDER+PREDES., whose transition smoothness was the highest of all the compared models. In this example, BLENDER+PREDES. generated a natural conversation along an appropriately gener-

[11] https://github.com/huggingface/transformers

[12] We extracted the keywords by following Zhong et al. (2021). The pairs that failed to extract keywords from the output utterances were excluded from the training data.

[13] An additional example is provided in Appendix D.2.

u_0	hey how is it going ?
u_1	i'm doing ok . i have mass this week (*school*: 0.64)
u_2	i just got done sewing a new shirt (*shirt*: 1.00)

Table 2: Part of the conversation plan by an existing model (DKRN). The elements in parentheses are keywords predicted by the model and the similarity score between the keyword and the target. (Target: ***shirt***)

ated subgoal sequence.

Overall. The TGCP task revealed the planning abilities of well-known retrieval models and newly prepared generative models. The retrieval models tended to have high achievement ratios but low transition smoothness and conversation probability, while the opposite was true for the generative models. These results show the trade-off between achievement ratio and the naturalness of conversation plans that current technology is facing. On the other hand, the generative model with a predesign subgoal strategy (BLENDER+PREDES.) improved the achievement ratio compared with the generative model with an on-the-fly strategy (BLENDER+CKC) ensuring its high transition smoothness and conversation probability. This implies that improving the achievement ratios of generative models by refining their subgoal strategies is an effective approach to overcome the trade-off.

3.6 Discussion: Number of conversation turns

We found that generative models behave critically differently from humans regarding the number of turns to reach targets, while their transition smoothness and conversation probability were comparable to those of humans. The average numbers of turns to reach targets of BLENDER+CKC and BLENDER+PREDES. were much larger than that of humans.[14] This result indicates that humans efficiently achieved TGCP with fewer turns while ensuring high conversation probability.[15] Making an agent has a strong conversation planning ability like a human can be one of our challenges in the future.

4 Conclusion

We have proposed the TGCP task as a platform for assessing the conversation planning ability of a dialogue model. Through this task setting, we have presented a first study for assessing the present neural conversational models' abilities for multiple-utterance planning, abstracting away the hard-to-control potential human factors. While the reported experiments cover only the task of Target-Guided Open-Domain Conversation (Tang et al., 2019), the idea of TGCP is expected to be applicable to a wider range of goal-oriented conversation tasks. Using TGCP, we revealed that the dialogue models with current technology have difficulty planning conversations to achieve given goals while ensuring the naturalness of the conversation. The experimental results also showed that refining the subgoal strategies for generative models might be an effective method to overcome this trade-off. We plan to research methods to solve this task setting with higher performance.

u_0	hello how are you ?
u_1	not too bad, how about you?
u_2	i am good thanks for asking
u_3	what do you do for a living, if you don't mind me asking? i am a nurse
u_4	that's cool. i work in a grocery store.
u_5	do you like it
u_6	yeah, it pays the bills, but i want to be a dental hygienist
u_7	wow that's a great career choice. how long have you been doing that
u_8	for as long as i can *remember*
u_9	what do you like to do in your spare time
u_{10}	i love to take *pictures* and photography is a hobby of mine
u_{11}	what kind of pictures do you take?
u_{12}	mostly *landscapes*, i love nature

Table 3: Conversation plan by BLENDER+PREDES. (Target: ***landscape***). The predicted subgoal sequence is *remember* → *picture* → ***landscape***.

Acknowledgments

We would like to thank all anonymous reviewers for their invaluable comments. This work was partly supported by JSPS KAKENHI Grant Numbers JP21J22383, JP22K17943, JST Moonshot R&D Grant Number JPMJMS2011.

[14] We counted the number of turns of the conversations where the target words are mentioned.

[15] A generated example is provided in Appendix D.1.

Ethical considerations

This paper honors the ACL Code of Ethics. This study uses existing dataset, preprocessed versions of ConvAI dataset Tang et al. (2019); Zhong et al. (2021), which we believe does not involve any ethical concerns. We used these data as training data for dialogue models and keyword prediction models and to randomly extract subsets for use as inputs for our generation task; we do not believe this involves any ethical concerns. This study includes manual work; For human evaluation, we hired crowdworkers and paid appropriate rewards for their labor (corresponding to a $14.40 for an hourly wage).

References

Daniel Adiwardana, Minh-Thang Luong, David R. So, Jamie Hall, Noah Fiedel, Romal Thoppilan, Zi Yang, Apoorv Kulshreshtha, Gaurav Nemade, Yifeng Lu, and Quoc V. Le. 2020. Towards a Human-like Open-Domain Chatbot. In *aiXiv preprint arXiv:2001.09977*.

Sanjeev Arora, Yingyu Liang, and Tengyu Ma. 2017. A Simple but Tough-to-Beat Baseline for Sentence Embeddings. In *Proceedings of the 5th International Conference on Learning Representations (ICLR)*.

Adi Botea, Christian Muise, Shubham Agarwal, Oznur Alkan, Ondrej Bajgar, Elizabeth Daly, Akihiro Kishimoto, Luis Lastras, Radu Marinescu, Josef Ondrej, Pablo Pedemonte, and Miroslav Vodolan. 2019. Generating Dialogue Agents via Automated Planning. In *The Second AAAI Workshop On Reasoning And Learning For Human-Machine Dialogues (DEEP-DIAL)*.

Zelin Dai, Weitang Liu, and Guanhua Zhan. 2019. Multiple Generative Models Ensemble for Knowledge-Driven Proactive Human-Computer Dialogue Agent. In *aiXiv preprint arXiv:1907.03590*.

Zhuoxuan Jiang, Jie Ma, Jingyi Lu, Guangyuan Yu, Yipeng Yu, and Shaochun Li. 2019a. A General Planning-Based Framework for Goal-Driven Conversation Assistant. In *Proceedings of the 33rd AAAI Conference on Artificial Intelligence (AAAI)*, pages 9857–9858.

Zhuoxuan Jiang, Xian Ling Mao, Ziming Huang, Jie Ma, and Shaochun Li. 2019b. Towards End-to-End Learning for Efficient Dialogue Agent by Modeling Looking-ahead Ability. In *Proceedings of the 20th Annual SIGdial Meeting on Discourse and Dialogue (SIGDIAL)*, pages 133–142.

Bart Kuijpers and Kris Dockx. 1998. An Intelligent Man-Machine Dialogue System Based on AI Planning. *Applied Intelligence*, 8(3):235–245.

Jiwei Li, Michel Galley, Chris Brockett, Georgios P. Spithourakis, Jianfeng Gao, and Bill Dolan. 2016. A Persona-Based Neural Conversation Model. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (ACL)*, pages 994–1003.

Peter Norvig and Stuart J Russell. 1995. *Artificial Intelligence: A Modern Approach*. Prentice Hall.

Jeffrey Pennington, Richard Socher, and Christopher D. Manning. 2014. GloVe: Global Vectors for Word Representation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1532–1543.

Jinghui Qin, Zheng Ye, Jianheng Tang, and Xiaodan Liang. 2020. Dynamic Knowledge Routing Network for Target-Guided Open-Domain Conversation. In *Proceedings of the 34th AAAI Conference on Artificial Intelligence (AAAI)*, pages 8657–8664.

Hannah Rashkin, Eric Michael Smith, Margaret Li, and Y. Lan Boureau. 2019. Towards Empathetic Open-domain Conversation Models: A New Benchmark and Dataset. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics (ACL)*, pages 5370–5381.

Stephen Roller, Emily Dinan, Naman Goyal, Da Ju, Mary Williamson, Yinhan Liu, Jing Xu, Myle Ott, Kurt Shuster, Eric M. Smith, Y. Lan Boureau, and Jason Weston. 2021. Recipes for Building an Open-Domain Chatbot. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics (EACL)*, pages 300–325.

Robyn Speer, Joshua Chin, and Catherine Havasi. 2017. ConceptNet 5.5: An Open Multilingual Graph of General Knowledge. In *Proceedings of the 31st AAAI Conference on Artificial Intelligence (AAAI)*, pages 4444–4451.

Robyn Speer, Joshua Chin, Andrew Lin, Sara Jewett, and Lance Nathan. 2018. LuminosoInsight/wordfreq: v2.2.

Amanda Stent, Rashmi Prasad, and Marilyn Walker. 2004. Trainable Sentence Planning for Complex Information Presentations in Spoken Dialog Systems. In *Proceedings of the 42nd Annual Meeting of the Association for Computational Linguistics (ACL)*, pages 79–86.

Jianheng Tang, Tiancheng Zhao, Chenyan Xiong, Xiaodan Liang, Eric P. Xing, and Zhiting Hu. 2019. Target-Guided Open-Domain Conversation. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics (ACL)*, pages 5624–5634.

Romal Thoppilan, Daniel De Freitas, Jamie Hall, Noam Shazeer, Apoorv Kulshreshtha, Heng-Tze Cheng, Alicia Jin, Taylor Bos, Leslie Baker, Yu Du, Yaguang Li, Hongrae Lee Huaixiu, Steven Zheng, Amin Ghafouri, Marcelo Menegali, Yanping Huang,

Maxim Krikun, Dmitry Lepikhin, James Qin, Dehao Chen, Yuanzhong Xu, Zhifeng Chen, Adam Roberts, Maarten Bosma, Vincent Zhao, Yanqi Zhou, Chung-Ching Chang, Igor Krivokon, Will Rusch, Marc Pickett, Pranesh Srinivasan, Laichee Man, Kathleen Meier-Hellstern, Meredith Ringel, Morris Tulsee, Doshi Renelito, Delos Santos, Toju Duke, Johnny Soraker, Ben Zevenbergen, Vinodkumar Prabhakaran, Mark Diaz, Ben Hutchinson, Kristen Olson, Alejandra Molina, Erin Hoffman-John, Josh Lee, Lora Aroyo, Ravi Rajakumar, Alena Butryna, Matthew Lamm, Viktoriya Kuzmina, Joe Fenton, Aaron Cohen, Rachel Bernstein, Ray Kurzweil, Blaise Aguera-Arcas, Claire Cui, Marian Croak, Ed Chi, and Quoc Le Google. 2022. LaMDA: Language Models for Dialog Applications. In *aiXiv preprint arXiv:2201.08239*.

Marilyn Walker, Amanda Stent, Francois Mairesse, and Rashmi Prasad. 2007. Individual and Domain Adaptation in Sentence Planning for Dialogue. *Journal of Artificial Intelligence Research (JAIR)*, 30:413–456.

Sixing Wu, Ying Li, Dawei Zhang, Yang Zhou, and Zhonghai Wu. 2020. Diverse and Informative Dialogue Generation with Context-Specific Commonsense Knowledge Awareness. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics (ACL)*, pages 5811–5820.

Wenquan Wu, Zhen Guo, Xiangyang Zhou, Hua Wu, Xiyuan Zhang, Rongzhong Lian, and Haifeng Wang. 2019. Proactive Human-Machine Conversation with Explicit Conversation Goal. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics (ACL)*, pages 3794–3804.

Yu Wu, Wei Wu, Chen Xing, Zhoujun Li, and Ming Zhou. 2017. Sequential Matching Network: A New Architecture for Multi-turn Response Selection in Retrieval-Based Chatbots. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (ACL)*, pages 496–505.

Hao Yuan and Jinqi An. 2020. Multi-Hop Memory Network with Graph Neural Networks Encoding for Proactive Dialogue. In *Proceedings of the 2020 6th International Conference on Computing and Artificial Intelligence (ICCAI)*, pages 24–29.

Saizheng Zhang, Emily Dinan, Jack Urbanek, Arthur Szlam, Douwe Kiela, and Jason Weston. 2018. Personalizing Dialogue Agents: I have a dog, do you have pets too? In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (ACL)*, pages 2204–2213.

Yizhe Zhang, Siqi Sun, Michel Galley, Yen-Chun Chen, Chris Brockett, Xiang Gao, Jianfeng Gao, Jingjing Liu, and William B. Dolan. 2020. DIALOGPT : Large-Scale Generative Pre-training for Conversational Response Generation. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics (ACL): System Demonstrations*, pages 270–278.

Xueliang Zhao, Wei Wu, Can Xu, Chongyang Tao, Dongyan Zhao, and Rui Yan. 2020. Knowledge-Grounded Dialogue Generation with Pre-trained Language Models. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 3377–3390.

Peixiang Zhong, Yong Liu, Hao Wang, and Chunyan Miao. 2021. Keyword-Guided Neural Conversational Model. In *Proceedings of the 35th AAAI Conference on Artificial Intelligence (AAAI)*, pages 14568–14576.

Peixiang Zhong, Chen Zhang, Hao Wang, Yong Liu, and Chunyan Miao. 2020. Towards Persona-Based Empathetic Conversational Models. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 6556–6566.

Yutao Zhu, Jian Yun Nie, Kun Zhou, Pan Du, Hao Jiang, and Zhicheng Dou. 2021. Proactive Retrieval-based Chatbots based on Relevant Knowledge and Goals. In *Proceedings of the 44th International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR)*, pages 2000–2004.

A Details of the TGCP Task

A.1 Calculation of Target Achievement Ratio

The achievement judgment was based on whether the target word itself was mentioned (Zhong et al., 2021); however, we found that there were several cases in which the achievement was judged to be a failure even though the target word was appropriately mentioned. Therefore, we modified the script[16] for judging the achievement such that these cases would be judged as achievements.

A.2 Relationship between Transition Smoothness and Conversation Probability

We investigated the correlation between transition smoothness and conversation probability of the experiment in Section 3, and found that Pearson's correlation coefficient was 0.828, which indicates a high correlation. Therefore, it appears that conversation probability is contained within transition smoothness, at least in our experiment. Therefore, evaluating transition smoothness may indicate the approximate tendency of conversation probability.

B Model Implementations and Setups

B.1 Existing Models

We used publicly available codes by their authors to implement the existing models.[17,18] To train the response selection models and the keyword prediction models, we used the same dataset and setups as described in their papers: CKC used the ConvAI2 dataset processed by Zhong et al. (2021), and the other models used the ConvAI2 dataset processed by Tang et al. (2019).

B.2 Training parameters of BLENDER.

To train BLENDER, we set the batch size to 32, the learning rate to 7.0×10^{-6}, the warmup steps to 100, the evaluation steps to 1,000, and the number of updates to 50,000. The other parameters were set to the default configuration of the huggingface transformer. We used the model at the validation loss minimum point for conversation planning.

B.3 Length of subgoal sequence.

We empirically confirmed the validity of tracing ConceptNet up to the three levels using the following procedure. First, we qualitatively checked the subgoal sequences generated by PREDES. and found that the subgoal sequences with relatedness scores of approximately 0.6, indicating a connection with the initial utterance, were naturally connected with the initial utterance. Then, we investigated how much ConceptNet need to be searched to generate subgoal sequences with a score of approximately 0.6. As a result, we confirmed that by tracing ConceptNet up to three steps, the average score of the subgoal sequences of the search results was 0.653, which exceeded 0.6. Therefore, we conclude that the three-step search is reasonable.

u_0	Not a big fan of talking face to face . How about you?
u_1	Me too. I prefer texting.
u_2	I truly understand. People love to comment on your behaviors when talking face to face. But talking online does not have such problems.
u_3	It sounds like you have experienced such comments. What did people accuse you of?
u_4	Well, I am a ***vegetarian***, but they said vegetarians are incomprehensible. Rude people, aren't they?

Table 4: Conversation plan generated by a human. (Target: ***vegetarian***)

C Instructions for Human Evaluation

Figure 3 shows the instructions given to Amazon Mechanical Turk workers concerning the evaluation of transition smoothness and conversation probability.

D Generated Conversation Plans

D.1 Human-generated Conversation Plan

Table 4 shows a conversation plan generated by a human. We confirmed that human could plan natural conversations that achieved their target despite the short number of turns.

D.2 Dead-ended Case

Table 5 shows an example where RETRIEVAL-ST. did not achieve its target. After the keyword *catch* is selected in utterance u_6, RETRIEVAL-ST. continued to generate the same utterance repeatedly from u_7 to the end (u_{16}). Ultimately, RETRIEVAL-ST. could not reach the target.

[16] https://github.com/zhongpeixiang/CKC/blob/master/util/data.py

[17] https://github.com/James-Yip/TGODC-DKRN

[18] https://github.com/zhongpeixiang/CKC

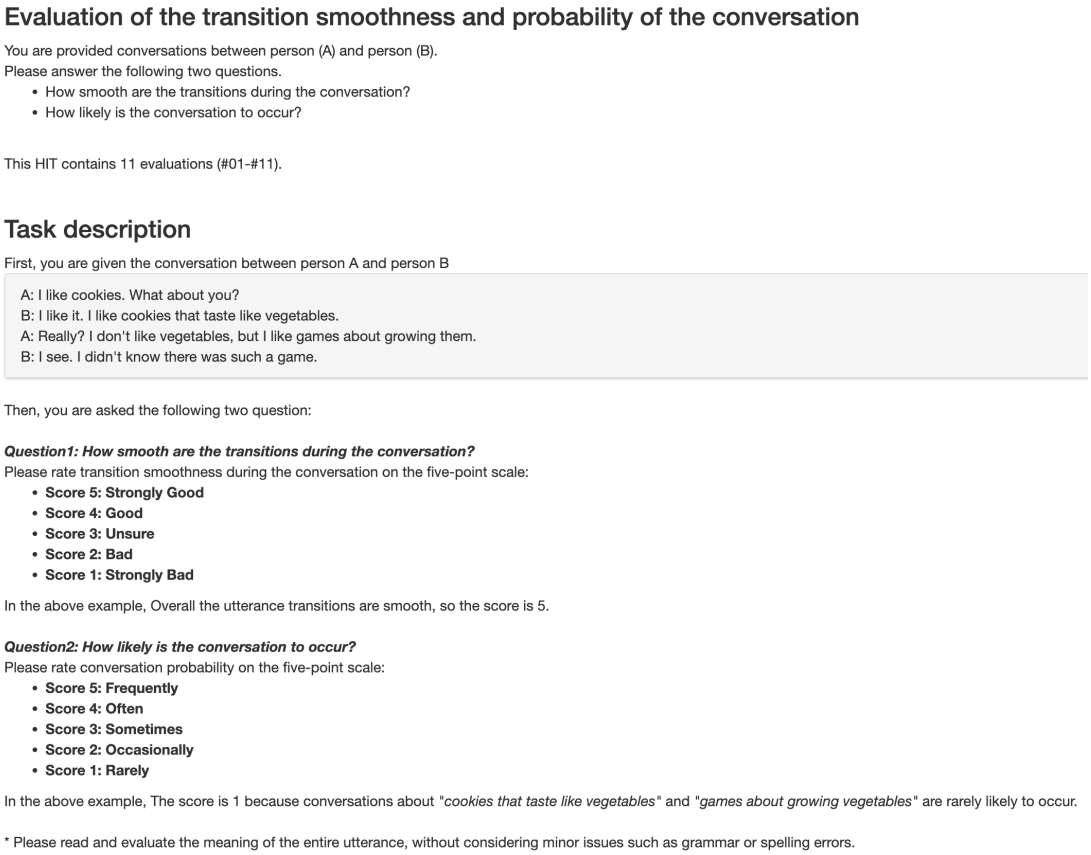

Figure 3: Evaluation instructions given to Amazon Mechanical Turk workers.

u_0	Today I am annoyed about not being able to drive.
u_1	That's *horrible*. My friend is also paralyzed. (*horrible*: 0.20)
u_2	I am sorry to *hear* of your injury. (*hear*: 0.43)
u_3	I feel like dump and skeleton but just *wasting* my hours here. (*waste*: 0.48)
u_4	I don't understand *people* with egos. (*people*: 0.50)
u_5	I think happiness is an illusion which *money* can help with. (*money*: 0.53)
u_6	I don lie, its bad when you get *caught*. (*catch*: 0.54)
u_7	I guess that's why the constantly keep calling me. (*catch*: 0.54)
u_8	I guess that's why the constantly keep calling me. (*catch*: 0.54)
$\vdots$	
u_{16}	I guess that's why the constantly keep calling me. (*catch*: 0.54)

Table 5: Part of the conversation plan by an existing model (RETRIEVAL-ST.). The elements in parentheses are keywords predicted by the model and similarity score between the keyword and target. (Target: ***chase***)

Does GPT-3 Generate Empathetic Dialogues? A Novel In-Context Example Selection Method and Automatic Evaluation Metric for Empathetic Dialogue Generation

Young-Jun Lee **Chae-Gyun Lim** **Ho-Jin Choi**
School of Computing, KAIST
{yj2961,rayote,hojinc}@kaist.ac.kr

Abstract

Since empathy plays a crucial role in increasing social bonding between people, many studies have designed their own dialogue agents to be empathetic using the well-established method of fine-tuning. However, they do not use prompt-based in-context learning, which has shown powerful performance in various natural language processing (NLP) tasks, for empathetic dialogue generation. Although several studies have investigated few-shot in-context learning for empathetic dialogue generation, an in-depth analysis of the generation of empathetic dialogue with in-context learning remains unclear, especially in GPT-3 (Brown et al., 2020). In this study, we explore whether GPT-3 can generate empathetic dialogues through prompt-based in-context learning in both zero-shot and few-shot settings. To enhance performance, we propose new in-context example selection methods, called SITSM and EMOSITSM, that utilize emotion and situational information. We also introduce a new automatic evaluation method, DIFF-EPITOME, which reflects the human tendency to express empathy. From the analysis, we reveal that our DIFF-EPITOME is effective in measuring the degree of human empathy. We show that GPT-3 achieves competitive performance with Blender 90M, a state-of-the-art dialogue generative model, on both automatic and human evaluation. Our code is available at https://github.com/passing2961/EmpGPT-3.

1 Introduction

Empathy refers to the ability to understand another person's experiences and feelings. This is important for increasing social bonding (rapport) with conversation partners (Zech and Rimé, 2005). Empathy is a multi-dimensional concept consisting of two main aspects: *cognitive* and *affective* (Davis et al., 1980). Since Rashkin et al. (2018) released the EMPATHETICDIALOGUES dataset for empathetic dialogue generation task, previous studies have improved their dialogue agents to generate more empathetic dialogues (Lin et al., 2019; Majumder et al., 2020; Zheng et al., 2021; Kim et al., 2021b; Sabour et al., 2021; Li et al., 2022). Depending on how the dialogue agents are trained, these approaches are largely divided into two categories depending on how to train own dialogue agents: (i) training from scratch or (ii) fine-tuning a pretrained dialogue generative model. However, neither of these approaches uses the prompt-based in-context learning paradigm in zero-shot and few-shot settings.

Recently, many researchers have attempted to build large-scale language models (LLMs), such as GPT-3 (Brown et al., 2020), OPT (Zhang et al., 2022), and HyperCloVA (Kim et al., 2021a). These models have shown surprising performance in various NLP tasks via prompt-based in-context learning, which is a new paradigm learning technique. Previous studies have explored the effect of few-shot in-context learning on dialogue generation tasks (Zheng and Huang, 2021; Madotto et al., 2021). While Madotto et al. (2021) explored few-shot in-context learning for empathetic dialogue generation, they did not conduct an in-depth analysis of their ability to generate empathetic dialogues. Moreover, they did not leverage GPT-3 as a prompting language model.

In this study, we explore whether GPT-3 generates empathetic dialogues using prompt-based in-context learning in both zero-shot and few-shot settings. We simply designed the prompt, which is a modified version of the basic prompt provided by OpenAI. As pointed out in (Liu et al., 2021), it is important to carefully choose in-context examples to enhance few-shot performance. Inspired by empathy being a multi-dimensional concept (Davis et al., 1980), we propose SITSM and EMOSITSM selection methods that choose in-context examples based on emotion and situation information. To reflect the human tendency to express empathy, we

Proceedings of the 29th International Conference on Computational Linguistics, pages 669–683
October 12–17, 2022.

also propose a new automatic evaluation method called DIFF-EPITOME, which is an extended version of EPITOME (Sharma et al., 2020). Our main contributions are as follows.

- We conduct an in-depth analysis of GPT-3's ability to generate empathetic dialogues with respect to Empathy, Diversity, and Fluency.
- We introduce SITSM and EMOSITSM, which are in-context example selection methods for empathetic dialogue generation task.
- We propose DIFF-EPITOME, an automatic evaluation method for empathetic dialogue generation. This method measures how dialogue agents empathize using the difference of EPITOME scores between human and agent.
- We show that GPT-3 performs better than the state-of-the-art model (Blender 90M (Roller et al., 2020)) on the EMPATHETICDIALOGUES test set, without additional training. In human evaluation, regardless of the dialogue turn setting, we also show that human annotators prefer GPT-3's responses on both human rating and A/B test.

2 Related Work

Empathetic Dialogue Generation Rashkin et al. (2018) first introduced the EMPATHETICDIALOGUES dataset. Lin et al. (2019) proposed a mixture of empathetic listeners (MoEL), where each listener is specialized in how to understand and respond appropriately to each emotion. Majumder et al. (2020) generated empathetic responses by mimicking human emotions, grouping emotions, and imposing stochasticity into each emotion group. Sharma et al. (2020) introduced a conceptual framework EPITOME (described in §3.3.1). Welivita and Pu (2020) proposed a taxonomy of empathetic response intents, consisting of nine categories (in Appendix D). For convenience, this is referred to as EMPINTENT. Zheng et al. (2021) proposed a multi-factor hierarchical framework (CoMAE), which considers EPITOME, EMPINTENT, and emotion. Kim et al. (2021b) generated more specific empathetic responses focused on emotion cause words by utilizing the Rational Speech Acts (RSA) framework (Frank and Goodman, 2012). Sabour et al. (2021) leveraged commonsense to generate more empathetic responses. Li et al. (2022) also leveraged external knowledge, such as commonsense knowledge, to explicitly generate empathetic responses.

Prompt-based In-Context Learning Since Brown et al. (2020) first introduced prompt-based in-context learning, many studies have shown that large-scale language models (e.g., GPT-3) itself has the ability to solve various NLP tasks in both zero-shot and few-shot settings (Schick and Schütze, 2020; Liu et al., 2021; Mishra et al., 2021; Wei et al., 2021; Yoo et al., 2021; Zhao et al., 2021; Schick and Schütze, 2021; Kim et al., 2021a; Gutiérrez et al., 2022; Meng et al., 2022). Some studies have shown that prompt-based few-shot in-context learning can also be successfully applied in dialogue generation tasks (Zheng and Huang, 2021; Madotto et al., 2021). The advantage of in-context learning is that it does not require any additional training. However, one problem is that GPT-3 achieves unstable performance depending on in-context examples. To mitigate this problem, Liu et al. (2021) proposed a kNN-augmented in-context example selection approach called the KATE. In this study, we extended this method to empathetic dialogue generation by selecting relevant in-context examples based on the situation and emotion (in §3.2).

3 Methodology

3.1 Task Formulation

The *empathetic dialogue generation* task aims to generate an empathetic response y for a given input x by maximizing the conditional probability $p(y|x) = \prod_t p(y_t|x, y_1, ..., y_{t-1})$, where x denotes the dialogue context. In general, previous studies (Lin et al., 2019; Majumder et al., 2020; Li et al., 2022) trained their own models on EMPATHETICDIALOGUES. However, in our case, we attempted to solve the task through GPT-3 in-context learning (Brown et al., 2020), without additional training. Therefore, in this study, task formulation is defined as follows:

$$p(y|x, C) = \prod_{t}^{|y|} p(y_t|C, x, y_1, ..., y_{t-1}),$$

where $C = \{x_1, y_1, x_2, y_2, ..., x_k, y_k\}$ is a concatenated string, and k denotes the number of examples for in-context few-shot learning. In a zero-shot setting ($k = 0$), we do not provide any in-context examples ($C = \emptyset$).

Algorithm 1: SITSM In-Context Example Selection

Input: A training dataset $\mathcal{D} = \{(x_i, y_i, s_i, e_i)\}_{i=1}^{N}$, number of training examples N, a sentence encoder $f_\theta(\cdot)$, number of in-context examples k, a test input $(x_{test}, y_{test}, s_{test}, e_{test})$

Output: a prompt input P to GPT-3

/* Step 1: Prepare $\mathcal{M}$ */

$\mathcal{M} \leftarrow$ empty list

for $(x_i, y_i, s_i, e_i) \in \mathcal{D}$ **do**

  $v_i = f_\theta(s_i)$

  $\mathcal{M}$.append(v_i)

end

/* Step 2: Get similarity score */

$v_{test} = f_\theta(s_{test})$

$\mathcal{M}_{sim} \leftarrow$ empty list

for $v_i \in \mathcal{M}$ **do**

  $sim_i = \frac{v_{test} \cdot v_i}{\|v_{test}\|_2 \|v_i\|_2}$

  $\mathcal{M}_{sim}$.append(sim_i)

end

/* Step 3: Construct prompt with selected k examples */

Select k indices $\mathcal{I} = \{idx_j\}_{j=1}^{k}$ from sorted $\mathcal{M}_{sim}$ (in descending order)

$\mathcal{M}_{ctx} \leftarrow$ empty list

for $idx_j \in \mathcal{I}$ **do**

  $\mathcal{M}_{ctx}$.append$((x_{idx_j}, y_{idx_j}))$

end

$C = [x_{idx_k}; y_{idx_k}; ...; x_{idx_1}; y_{idx_1}]$

$P = [C; x_{test}; y_{test}]$

Algorithm 2: EMOSITSM In-Context Example Selection

Input: A training dataset $\mathcal{D} = \{(x_i, y_i, s_i, e_i)\}_{i=1}^{N}$, number of training examples N, an emotion $E = \{e\}_1^{32}$, a sentence encoder $f_\theta(\cdot)$, number of in-context examples k, a test input $(x_{test}, y_{test}, s_{test}, e_{test})$

Output: a prompt input P to GPT-3

/* Step 1: Group $\mathcal{D}$ by 32 emotions and prepare $\mathcal{M}$ */

$\mathcal{M} \leftarrow$ empty dict

for $e \in E$ **do**

  for $(x_i, y_i, s_i, e_i) \in \mathcal{D}$ **do**

    $v_i = f_\theta(s_i)$

    $\mathcal{M}[e]$.append$((x_i, y_i, s_i, e_i, v_i))$

  end

end

/* Step 2: Get similarity score */

$e_{test} \leftarrow$ emotion of test input

$v_{test} = f_\theta(s_{test})$

$\mathcal{M}_{sim} \leftarrow$ empty list

for $(x_i, y_i, s_i, e_i, v_i) \in \mathcal{M}[e_{test}]$ **do**

  $sim_i = \frac{v_{test} \cdot v_i}{\|v_{test}\|_2 \|v_i\|_2}$

  $\mathcal{M}_{sim}$.append(sim_i)

end

/* Step 3: Construct prompt with selected k examples */

Select k indices $\mathcal{I} = \{idx_j\}_{j=1}^{k}$ from sorted $\mathcal{M}_{sim}$ (in descending order)

$\mathcal{M}_{ctx} \leftarrow$ empty list

for $idx_j \in \mathcal{I}$ **do**

  $\mathcal{M}_{ctx}$.append$((x_{idx_j}, y_{idx_j}))$

end

$C = [x_{idx_k}; y_{idx_k}; ...; x_{idx_1}; y_{idx_1}]$

$P = [C; x_{test}; y_{test}]$

3.2 In-Context Example Selection Methods (SM)

As reported by (Liu et al., 2021), GPT-3 is sensitive to randomly chosen in-context examples. To mitigate this problem, they selected semantically relevant in-context examples from the training set using the kNN retrieval module for each test input. Inspired by (Liu et al., 2021), we introduce two selection methods: SITSM and EMOSITSM. In EMPATHETICDIALOGUES, each training instance consists of dialogue context x, golden response y, emotion e, and situation sentence s. Table 1 shows the samples of the in-context examples selected by SITSM and EMOSITSM.

3.2.1 SITSM

Starting from the assumption that the situation sentences are similar, the dialogue context will have similar patterns of expressing empathy. Specifically, we first use the sentence encoder [1] to obtain all the embedding vectors of situation sentences in the training set in advance. We convert each test situation input s into a vector representation. For each test situation input s, we then select the most relevant k examples from the training set based on the similarity score. For the similarity measures, we adopt the cosine similarity. We construct the prompt with the selected k examples, where the ordering of k examples was performed based on the similarity score of each example. In other words, the example most similar to the test input s is placed close to the test input.[2] The entire process is presented in Algorithm 1.

3.2.2 EMOSITSM

Empathy is a multi-dimensional concept that consists of two aspects: *cognitive* and *affective* (Davis et al., 1980). Based on this concept, we argue that we should choose good in-context examples based on these two aspects. The *cognitive* aspect involves understanding and interpreting the situation of another person. The *affective* aspect is to express an emotional reaction. We can view the situation as the *cognitive* aspect, and emotion as *affective* as-

[1] We use `stsb-roberta-large` version of Sentence-BERT (Reimers and Gurevych, 2019)

[2] As argued in (Liu et al., 2021), the choice of ordering is data-dependent. In this study, we adopt the reverse order that performs best on the Natural Questions (NQ) (Kwiatkowski et al., 2019) dataset.

	Content	Score
Test situation	I'm starting a new job next week, and I am super nervous.	-
situation 1	I start my new job tomorrow. I am extremely nervous about it	0.9741
Dialogue Context	S: I start my new job tomorrow. L: Nice! What will you be doing? S: I will be working as a consultant, I am extremely nervous L: Nothing to be nervous about, I'm sure you'll do fine!	-
situation 2	I am starting a new job next week. I am so nervous.	0.9937
Dialogue Context	S: MY new job starts next week. L: Awesome! What will you be doing? S: Im transferring to another store and becoming a supervisor. Im nervous cause of the position and its all new people. L: That is awesome. They chose you for a reason. You will do great!	-

(a) Sample of selected in-context example by SITSM.

	Content	Score
Test situation	I remain loyal to my wife always	-
situation 1	I've always been loyal to my wife.	0.9716
Dialogue Context	S: I've always been faithful as a husband, I pride myself on that. L: Nice, there is not a lot of people like you S: Well, I don't know about that, but I actually wanted to marry my wife. L: Good	-
situation 2	i am loyal to my wife and i'll always be	0.9786
Dialogue Context	S: i am loyal to my wife and i'll always be. L: That's an amazing attitude. Not many people are like that nowadays. S: i don't get it why people cheat and hurt each other, but i suppose because they are never happy with what they have and only feel grateful for it after they lose it. L: Crazy world we live in.	-

(b) Sample of selected in-context example by EMOSITSM when "faithful" emotion.

Table 1: Samples of selected in-context examples (when $k = 2$) from EMPATHETICDIALOGUES training set, based on the similarity score with a given test situation. We also present dialogue contexts corresponding to situation sentences, respectively. (S: Speaker, L: Listener) More samples are in Appendix E.

pect. Fortunately, EMPATHETICDIALOGUES contains situation and emotion information for each instance. To this end, we propose EMOSITSM, which selects in-context examples based on emotions and situations. Specifically, we first group all training instances by 32 emotion types and encode each situation sentence s into a vector representation by using the sentence encoder (same model in SITSM) simultaneously. We then calculate the similarity score between the test input and all instances in the group with the same emotion as that of the test input. Finally, based on the similarity scores, we select k examples that are closest to the test input. Similar to SITSM, we also consider the ordering of k selected examples when constructing the final prompt. The algorithm EMOSITSM is presented in Algorithm 2.

3.3 A New Automatic Evaluation Metric for Empathetic Dialogue Generation

We propose an automatic evaluation metric, called DIFF-EPITOME, which reflects human patterns when empathy is expressed as dialogue continues. In §3.3.1, we describe the EPITOME-based metric used in previous studies (Sharma et al., 2020; Kim et al., 2021b). Empirically, we analyze whether there is a specific tendency in human communication (see §3.3.2). Based on the above analysis, we propose a new evaluation method called DIFF-EPITOME (see §3.3.3).

3.3.1 EPITOME-based Automatic Evaluation

EPITOME, introduced by (Sharma et al., 2020), is a new conceptual framework for expressing empathy in text-based, asynchronous contexts. EPITOME consists of three communication mechanisms of empathy: **Explorations** (EX), **Interpretations** (IP), and **Emotional Reactions** (ER). The mechanisms are described in Appendix C.

In a recent study (Kim et al., 2021b), each mechanism was used as an automatic metric to measure the empathy of generated responses using a fine-tuned RoBERTa (Liu et al., 2019) model.[3]. Each generated response was measured by one of the val-

[3] Actually, in (Kim et al., 2021b), they only used IP and EX scores. However, we even use the ER score in the experiments (see in Table 3)

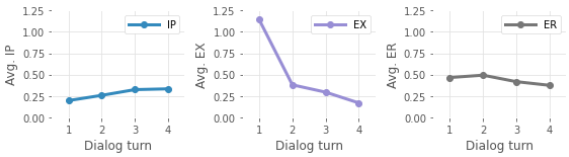

Figure 1: Analysis of EMPATHETICDIALOGUES train set w.r.t. EPITOME, as the conversation continues. The x-axis represents the index of dialogue turn and the y-axis represents the average score of IP, EX, ER for each dialogue turn. Each score was predicted by the RoBERTa model (described in §3.3.1).

ues (0, 1, or 2) predicted from the model. Higher values indicate stronger empathy. Following the author's official code[4], we fine-tuned three RoBERTa-base models to measure the IP, EX, and ER scores. The EPITOME-based metric is formulated as

$$\text{EP}_m(y) = \text{RoBERTa}_m(x, y) \quad (1)$$

$$\text{EP}_m(Y) = \frac{1}{N}\sum_{i=1}^{N}\text{EP}_m(\hat{y}_i) \quad (2)$$

where $m \in \{\text{IP}, \text{EX}, \text{ER}\}$, the number of test examples N, and $Y = \{y_1, ..., y_N\}$.

We ask: "*Can we argue that empathetic dialogue agents empathize well with high scores measured by* EPITOME*-based metrics?*" To validate our hypothesis, we check the results of Blender 90M (Roller et al., 2020) trained on EMPATHETICDIALOGUES. Given the input utterance, "*I feel like deepening my connection to god is the most important thing in my life*," Blender produces a response "*I'm sorry to hear that. What is your connection to god?*", which is measured using the EPITOME-based method with an IP of 0, EX of 2, and ER of 2. This indicates more exploration of the situation of a partner and expressing emotional reactions. On the other hand, the IP, EX, and ER scores of the golden response "*I think that is a lovely thing*" are all measured to be 0. According to the EPITOME-based method, Blender expresses stronger empathy, which is regarded as an empathetic dialogue agent. However, in this case, humans do not express empathy. Therefore, it is necessary to develop a new automatic evaluation method.

3.3.2 How do humans empathize?

We ponder: *How do humans empathize?* We humans perceive and understand another person's situation by putting ourselves in the other's shoes. This is known as *perspective-taking* in cognitive science (Davis et al., 1980). Even following *perspective-taking*, it is difficult to accurately recognize another person's situation at the beginning of a dialogue. Therefore, humans tend to ask their situation and feelings. Through a simple experiment, we observe that there is a tendency to express empathy in human communication. Figure 1 shows that the average EX scores decrease as the dialogue continued. In the IP and ER scores, each goes up and down slightly.

3.3.3 DIFF-EPITOME-based Automatic Evaluation

Based on the above analysis, we propose a new automatic evaluation method DIFF-EPITOME, which is an extended version of the EPITOME-based method. The key idea of DIFF-EPITOME is to measure the difference in EP_m score between the human golden response y_i and the predicted response $\hat{y}_i$ using a model, as follows:

$$\text{diff-EP}_m(Y) = \frac{1}{N}\sum_{i=1}^{N}(\text{EP}_m(y_i) - \text{EP}_m(\hat{y}_i))^2 \quad (3)$$

where $m \in \{\text{IP}, \text{EX}, \text{ER}\}$

A lower diff-EP value indicates that the expressed empathy is more human-like.

4 Experimental Setup

4.1 Dataset

We evaluate our proposed model on the benchmark EMPATHETICDIALOGUES dataset (Rashkin et al., 2018), which consists of 25k open-domain conversations grounded in emotional situations. Each dialogue is composed of consecutive utterances of the speaker and listener, where each utterance is labeled among 32 emotion categories. Each dialogue contains a situation sentence.

4.2 Evaluation Models

Blender We compare GPT-3 with Blender 90M (Roller et al., 2020), which is one of the state-of-the-art dialogue agents, fine-tuned on the EMPATHETICDIALOGUES train dataset as our baseline.

EmpGPT-3 To observe whether a prompt specialized to the empathetic dialogue generation task elicits GPT-3 to produce more empathetic responses, we construct a simple prompt template which is "The following is a conversation with an empathetic AI assistant. The assistant empathizes

[4] https://github.com/behavioral-data/Empathy-Mental-Health

Model	# classes	Acc	Macro F1
EMOACC	32	0.40	0.39
INTENTACC	9	0.96	0.90

Table 2: Performance of BERT-based classifiers trained on EMPATHETICDIALOGUES (Rashkin et al., 2018).

with human experiences and feelings well. Human: u_1 Empathy AI: u_2 ...", where u_1, u_2, ... are utterances. We present examples of the constructed prompt used in this study in the Appendix A.

4.3 Implementation Details.

We fine-tune the Blender 90M (Roller et al., 2020) on the EMPATHETICDIALOGUES dataset using a ParlAI framework[5]. We used the default hyperparameter settings provided by the ParlAI framework. We selected the model checkpoint that achieved the best performance, based on the perplexity of the validation set. For EmpGPT-3, we use a `davinci` version with hyperparameter settings as follows: temperature 0.8, maximum tokens 128, frequency penalty 0.4, and presence penalty 0.4. For the stop tokens, we use `Human:` and `Empathy AI:`.

4.4 Automatic Evaluations

To investigate whether GPT-3 can generate empathetic responses in both zero-shot and few-shot settings, we evaluate the generated responses on various metrics for **Diversity**, **Fluency**, and **Empathy**.

4.4.1 Diversity

It is important to consider diversity because various responses to an input utterance may be possible depending on the context. We measure the diversity of generated responses based on two metrics.

- DISTINCT-N (DIST-N) (Li et al., 2015; See et al., 2019a) measures the ratio of unique n-grams. A higher ratio indicates a higher diversity of generated responses.

- NIDF[6] (See et al., 2019b) measures the rareness of a word w. The NIDF score is calculated as:

$$\text{NIDF}(w) = \frac{\text{IDF}(w) - \texttt{min\_idf}}{\texttt{max\_idf} - \texttt{min\_idf}}, \quad (4)$$

where $\text{IDF}(w) = \log(R/c_w)$, R denotes the number of responses in dataset, c_w is the number of responses that contain w, `min_idf` and `max_idf` are the minimum and maximum IDFs. Detailed information is described in (See et al., 2019b). A higher NIDF score indicates a more specific response and a higher proportion of rare words.

4.4.2 Fluency

Following (Feng et al., 2020; Pang et al., 2020), we measure the fluency of generated responses through a perplexity (PPL) by adopting GPT2-XL, not fine-tuned on any downstream tasks related to the dialogue domain. A lower PPL indicates that the response is more fluent.

4.4.3 Empathy

- EMOACC measures an emotion accuracy using a fine-tuned BERT-base (Devlin et al., 2018) model on the EMPATHETICDIALOGUES dataset labeled with 32 emotion categories. The performance of the classifier is reported in Table 2.

- INTENTACC measures the response intent accuracy using a fine-tuned BERT model on the EMPINTENT dataset, introduced by (Welivita and Pu, 2020). The performance of the classifier is reported in Table 2.

- EPITOME (Sharma et al., 2020) measures IP, EX, and ER by leveraging fine-tuned RoBERTa models, respectively (§3.3.1).

- DIFF-EPITOME (§3.3.3) measures the difference scores of IP, EX, ER between the human golden response and predicted response (§3.3.3).

4.5 Human Evaluation

Following (Rashkin et al., 2018; Lin et al., 2019; Majumder et al., 2020; Kim et al., 2021b), we conduct two standard human evaluations with three annotators: (i) Human A/B Test and (ii) Human Ratings. We recruited three annotators via an on-campus announcements. After randomly sampling 100 test examples, we divided them into 50 examples for each single-turn and multi-turn setting. The Human A/B Test allows annotators to choose which response is more empathetic. They can choose "*Tie*" if the two given responses are both good or

[5] https://github.com/facebookresearch/ParlAI

[6] Normalized Inverse Document Frequency

	Empathy								Diversity			Fluency	
Model	IntentAcc	EmoAcc	IP	EX	ER	diff-IP	diff-EX	diff-ER	dist-1	dist-2	NIDF	PPL	Avg. Len
Blender (single-turn)	**0.3084**	0.1593	0.2057	0.3423	**1.0570**	**0.7202**	**1.1934**	1.0359	0.9541	0.9824	0.2454	166.86	12.58
EmpGPT3 (single-turn)	0.2211	**0.1683**	**0.2780**	0.3118	0.8142	0.8625	1.5364	0.9239	**0.9614**	**0.9975**	**0.2860**	169.39	**16.03**
EmpGPT3 (multi-turn)	0.2528	0.1594	0.2717	**0.4970**	0.6439	0.7884	1.2564	**0.7295**	0.9400	0.9966	0.2840	**118.43**	15.6

Table 3: Comparison of the zero-shot performance of EmpGPT-3 with Blender 90M (Roller et al., 2020) on EmpatheticDialogues test set. In a single-turn setting, we inject only the last utterance with the prompt template, not including the whole dialogue context, into GPT-3. In contrast, in a multi-turn setting, we consider the whole dialogue context when constructing the prompt.

	Empathy					Diversity	Fluency
k	IntentAcc	EmoAcc	diff-IP	diff-EX	diff-ER	NIDF	PPL
0	0.2528	0.1594	**0.7884**	1.2564	0.7295	0.2840	118.43
1	**0.2650**	**0.1622**	0.8418	**0.9925**	0.6655	0.2896	175.71
2	0.2623	0.1614	0.8481	0.9988	**0.6514**	**0.3045**	**74.39**

Table 4: Ablation study on the number of in-context examples k in EmpGPT-3 prompts. Evaluation results are conducted on the EmoSitSM.

	Empathy							
type	IntentAcc	EmoAcc	IP	EX	ER	diff-IP	diff-EX	diff-ER
Random	0.2603	0.1390	**0.2670**	0.3204	0.6141	0.8905	0.9894	0.6592
SitSM	0.2587	0.1461	0.2387	**0.3298**	**0.6282**	**0.8025**	**0.9706**	0.6678
EmoSitSM	**0.2623**	**0.1614**	0.2599	0.3094	0.6219	0.8481	0.9988	**0.6514**

Table 5: Comparison of Empathy performance of EmpGPT-3 with various selection methods when $k = 2$ and multi-turn setting.

bad. For the Human Ratings, we asked three annotators to rate the generated responses on three metrics (in a 4-likert scale): Empathy, Relevance, and Fluency. The questionnaires and system used for the human evaluation are described in Appendix G and H.

5 Experimental Results

5.1 Main Results

GPT-3 vs. Blender 90M As shown in Table 3, GPT-3 shows competitive performance compared to Blender 90M on most evaluation metrics (8 of 12, except for Avg. Len) in a zero-shot setting. Regardless of the turn setting, GPT-3 is difficult to generate responses with proper intentions than Blender 90M. Owing to the enormous generative capacity of GPT-3, EmpGPT3 can generate more diverse and specific responses. For diff-Epitome, Blender tends to generate overly emotional expressions because of its higher performance in both ER and diff-ER (1.0570 and 1.0359). However, EmpGPT-3 still cannot follow how humans empathize in terms of the IP and EX.

single-turn vs. multi-turn The main difference between these two settings is whether the entire dialogue context is given together when constructing the prompt. For **Empathy**, EmpGPT-3 achieves lower diff-{IP,EX,ER} scores than the single-turn setting. This suggests that, given the dialogue context in the zero-shot setting, GPT-3 better understands human situations and expresses empathy just as humans do. Similarly, the performance of IntentAcc, which requires reasoning about situations, has also improved. For **Fluency**, EmpGPT-3 generates more fluent responses from the average PPL with a large margin of 50.96.

5.2 Ablation Studies

Number of In-Context Examples As shown in Table 4, we explore the effect of the number of in-context examples on the EmpGPT-3's performance. Specifically, we conduct an experiment on the EmpatheticDialogues test set with $k = \{1, 2\}$. To select adequate in-context examples, we adopt our EmoSitSM, which achieves a better performance (see Table 11). The overall few-shot performance is better than that when $k = 0$. In particular, we observe that fluency when $k = 2$ is much higher than those for others ($k = \{0, 1\}$). In addition, the diff-{EX,ER} scores of EmpGPT-3 are much lower than those of the zero-shot performance. This implies that GPT-3 indirectly learns how to express empathy from given in-context examples. Full experiment results are shown in Table 11 (see Appendix B).

Various Selection Methods We investigate the performance of GPT-3 according to the selection method. Table 5 shows that the similarity-based methods (i.e., SitSM and EmoSitSM) have slightly improved performance in most metrics compared with the Random method (similar results were reported in (Liu et al., 2021)). The Random method selects in-context examples randomly. In particular, EmoSitSM is highly effective in terms of emotion accuracy compared with

Model	Win	Lose	Tie
single-EmpGPT-3 vs. Blender	**46.7%**	32.0%	21.3%
multi-EmpGPT-3 vs. Blender	**36.7%**	**36.7%**	26.6%

Table 6: Comparison of EmpGPT-3 (in single- and multi-turn) with Blender (Roller et al., 2020) on A/B test. The win rate is for EmpGPT-3.

Model	EMPATHY↑	RELEVANCE↑	FLUENCY↑
Blender	2.78	2.82	3.34
single-EmpGPT-3	**3.07**	**2.97**	**3.6**
Blender	2.79	2.78	3.13
multi-EmpGPT-3	**2.85**	**2.8**	**3.21**

Table 7: Comparison of EmpGPT-3 (in single- and multi-turn) with Blender (Roller et al., 2020) on human ratings.

other methods. However, SITSM shows a better performance in diff-{IP,EX}, demonstrating that SITSM better understands and explores situations. We report full experiment results in Table 11 (Appendix B).

5.3 Human Evaluation Results

As shown in Table 6, users prefer responses generated by *single*-EmpGPT-3 to those generated by Blender. When comparing *multi*-EmpGPT-3 with Blender, users prefer responses from both the models equally. We measure the inter-rater agreement using Krippendorff's α. For Human A/B Test, Krippendorff's α is 0.26, which implies a fair agreement. Regardless of the dialogue turn setting, EmpGPT-3 obtains a better performance on human ratings. Especially, users who evaluate responses from *single*-EmpGPT-3 to be more empathetic and relevant to the given dialogue context, as shown in Table 7.

5.4 Analysis of Correlation

We conducted a correlation analysis to verify the validity of the proposed evaluation metric DIFF-EPITOME. Figure 2 shows Pearson's r correlation matrix between human ratings and two automatic methods: EPITOME-based and DIFF-EPITOME-based. We observe that our DIFF-EPITOME-based automatic metric more correlates with human ratings than the EPITOME-based automatic metric. Moreover, we found that a high ER score does not indicate that the dialogue agent empathizes well. It suggests that it is necessary to use emotional reactions on time when expressing empathy to interlocutors. Correlation analysis revealed that our proposed metric is effective for empathetic dialogue generation and it is important to consider the tendency of how humans do empathize as the dialogue continues. We hope that this analysis will be helpful for other researchers.

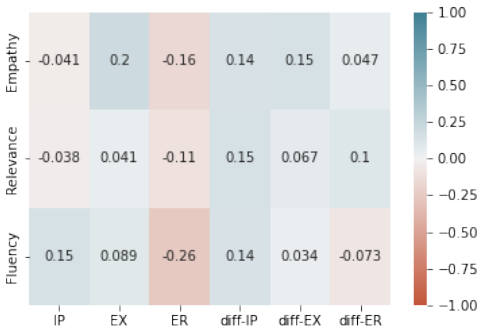

Figure 2: Pearson's r correlation matrix between human ratings and EPITOME- and DIFF-EPITOME-based automatic metrics, respectively. The degree of correlation increases from red to blue.

	Dialogue Context
S:	Do you even know how crazy it is to skydive?
	I must have been absolutely nuts when I did it.
L:	Oh I don't think I could do that.
	I have a fear of falling from high places.
S:	It gave me the biggest rush that's for sure.
	But on the way down I was saying my prayers.
L:	I think I would pass out from fear lol.
S:	You should do it sometime.
	It's fun to take chances.
	Generated Responses
Blender:	
I have never skydive. I have heard it is pretty scary.	
***single*-EmpGPT-3:**	
No, I cannot say that I do. But I think I can imagine.	
	Dialogue Context
S:	I have some great friends.
	One of them drove me to the airport tomorrow
L:	That was really nice of them.
	I bet you're appreciative.
S:	Yes I am, they are great
	Generated Responses
Blender:	
I'm glad you have some good friends.	
***multi*-EmpGPT-3:**	
I'm sure they'll like to hear that.	

Table 8: Examples of generated responses from Blender (Roller et al., 2020) and EmpGPT-3 with single- and multi-turn setting. (S: Speaker, L: Listener)

5.5 Case Studies

Table 8 shows examples of the responses generated by the Blender and EmpGPT-3 (with single- and multi-turn settings). Our *multi*-EmpGPT-3 can generate responses that require complex reasoning by understanding a speaker's situation and feelings. Additional examples are presented in Table 13.

6 Conclusion

In this study, we explore the zero-shot and few-shot performance of GPT-3 in an empathetic dialogue generation task on various metrics with respect to Diversity, Fluency, and Empathy. We introduce a new in-context example selection method, SITSM and EMOSITSM. We also propose a novel automatic evaluation method, DIFF-EPITOME, for empathetic dialogue generation. We show that GPT-3 achieves competitive performance with Blender 90M on the EMPATHETICDIALOGUES test set on both automatic and human evaluations. From the correlation analysis, we reveal that DIFF-EPITOME correlates more with human ratings. In future work, we will apply OPT (Zhang et al., 2022) with an optimized prompt. In addition, we reflect on the overall human tendency to express empathy in the modeling.

Acknowledgements

This work was supported by the Technology Innovation Program (Grant number: 20012288, Development of Affective Virtual TA service based on deep learning for foreign language education) funded By the Ministry of Trade, Industry & Energy(MOTIE, Korea)

References

Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. 2020. Language models are few-shot learners. *Advances in neural information processing systems*, 33:1877–1901.

Mark H Davis et al. 1980. A multidimensional approach to individual differences in empathy.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*.

Steven Y Feng, Varun Gangal, Dongyeop Kang, Teruko Mitamura, and Eduard Hovy. 2020. Genaug: Data augmentation for finetuning text generators. *arXiv preprint arXiv:2010.01794*.

Michael C Frank and Noah D Goodman. 2012. Predicting pragmatic reasoning in language games. *Science*, 336(6084):998–998.

Bernal Jiménez Gutiérrez, Nikolas McNeal, Clay Washington, You Chen, Lang Li, Huan Sun, and Yu Su. 2022. Thinking about gpt-3 in-context learning for biomedical ie? think again. *arXiv preprint arXiv:2203.08410*.

Boseop Kim, HyoungSeok Kim, Sang-Woo Lee, Gichang Lee, Donghyun Kwak, Dong Hyeon Jeon, Sunghyun Park, Sungju Kim, Seonhoon Kim, Dongpil Seo, et al. 2021a. What changes can large-scale language models bring? intensive study on hyperclova: Billions-scale korean generative pretrained transformers. *arXiv preprint arXiv:2109.04650*.

Hyunwoo Kim, Byeongchang Kim, and Gunhee Kim. 2021b. Perspective-taking and pragmatics for generating empathetic responses focused on emotion causes. *arXiv preprint arXiv:2109.08828*.

Tom Kwiatkowski, Jennimaria Palomaki, Olivia Redfield, Michael Collins, Ankur Parikh, Chris Alberti, Danielle Epstein, Illia Polosukhin, Jacob Devlin, Kenton Lee, et al. 2019. Natural questions: a benchmark for question answering research. *Transactions of the Association for Computational Linguistics*, 7:453–466.

Jiwei Li, Michel Galley, Chris Brockett, Jianfeng Gao, and Bill Dolan. 2015. A diversity-promoting objective function for neural conversation models. *arXiv preprint arXiv:1510.03055*.

Qintong Li, Piji Li, Zhaochun Ren, Pengjie Ren, and Zhumin Chen. 2022. Knowledge bridging for empathetic dialogue generation.

Zhaojiang Lin, Andrea Madotto, Jamin Shin, Peng Xu, and Pascale Fung. 2019. Moel: Mixture of empathetic listeners. *arXiv preprint arXiv:1908.07687*.

Jiachang Liu, Dinghan Shen, Yizhe Zhang, Bill Dolan, Lawrence Carin, and Weizhu Chen. 2021. What makes good in-context examples for gpt-3? *arXiv preprint arXiv:2101.06804*.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Roberta: A robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692*.

Andrea Madotto, Zhaojiang Lin, Genta Indra Winata, and Pascale Fung. 2021. Few-shot bot: Prompt-based learning for dialogue systems. *arXiv preprint arXiv:2110.08118*.

Navonil Majumder, Pengfei Hong, Shanshan Peng, Jiankun Lu, Deepanway Ghosal, Alexander Gelbukh, Rada Mihalcea, and Soujanya Poria. 2020. Mime:

Mimicking emotions for empathetic response generation. *arXiv preprint arXiv:2010.01454*.

Yu Meng, Jiaxin Huang, Yu Zhang, and Jiawei Han. 2022. Generating training data with language models: Towards zero-shot language understanding. *arXiv preprint arXiv:2202.04538*.

Swaroop Mishra, Daniel Khashabi, Chitta Baral, Yejin Choi, and Hannaneh Hajishirzi. 2021. Reframing instructional prompts to gptk's language. *arXiv preprint arXiv:2109.07830*.

Bo Pang, Erik Nijkamp, Wenjuan Han, Linqi Zhou, Yixian Liu, Kewei Tu, et al. 2020. Towards holistic and automatic evaluation of open-domain dialogue generation.

Hannah Rashkin, Eric Michael Smith, Margaret Li, and Y-Lan Boureau. 2018. Towards empathetic open-domain conversation models: A new benchmark and dataset. *arXiv preprint arXiv:1811.00207*.

Nils Reimers and Iryna Gurevych. 2019. Sentence-bert: Sentence embeddings using siamese bert-networks. *arXiv preprint arXiv:1908.10084*.

Stephen Roller, Emily Dinan, Naman Goyal, Da Ju, Mary Williamson, Yinhan Liu, Jing Xu, Myle Ott, Kurt Shuster, Eric M Smith, et al. 2020. Recipes for building an open-domain chatbot. *arXiv preprint arXiv:2004.13637*.

Sahand Sabour, Chujie Zheng, and Minlie Huang. 2021. Cem: Commonsense-aware empathetic response generation. *arXiv preprint arXiv:2109.05739*.

Timo Schick and Hinrich Schütze. 2020. It's not just size that matters: Small language models are also few-shot learners. *arXiv preprint arXiv:2009.07118*.

Timo Schick and Hinrich Schütze. 2021. Generating datasets with pretrained language models. *arXiv preprint arXiv:2104.07540*.

Abigail See, Aneesh Pappu, Rohun Saxena, Akhila Yerukola, and Christopher D Manning. 2019a. Do massively pretrained language models make better storytellers? *arXiv preprint arXiv:1909.10705*.

Abigail See, Stephen Roller, Douwe Kiela, and Jason Weston. 2019b. What makes a good conversation? how controllable attributes affect human judgments. *arXiv preprint arXiv:1902.08654*.

Ashish Sharma, Adam S Miner, David C Atkins, and Tim Althoff. 2020. A computational approach to understanding empathy expressed in text-based mental health support. *arXiv preprint arXiv:2009.08441*.

Jason Wei, Maarten Bosma, Vincent Y Zhao, Kelvin Guu, Adams Wei Yu, Brian Lester, Nan Du, Andrew M Dai, and Quoc V Le. 2021. Finetuned language models are zero-shot learners. *arXiv preprint arXiv:2109.01652*.

Anuradha Welivita and Pearl Pu. 2020. A taxonomy of empathetic response intents in human social conversations. *arXiv preprint arXiv:2012.04080*.

Kang Min Yoo, Dongju Park, Jaewook Kang, Sang-Woo Lee, and Woomyeong Park. 2021. Gpt3mix: Leveraging large-scale language models for text augmentation. *arXiv preprint arXiv:2104.08826*.

Emmanuelle Zech and Bernard Rimé. 2005. Is talking about an emotional experience helpful? effects on emotional recovery and perceived benefits. *Clinical Psychology & Psychotherapy: An International Journal of Theory & Practice*, 12(4):270–287.

Susan Zhang, Stephen Roller, Naman Goyal, Mikel Artetxe, Moya Chen, Shuohui Chen, Christopher Dewan, Mona Diab, Xian Li, Xi Victoria Lin, et al. 2022. Opt: Open pre-trained transformer language models. *arXiv preprint arXiv:2205.01068*.

Zihao Zhao, Eric Wallace, Shi Feng, Dan Klein, and Sameer Singh. 2021. Calibrate before use: Improving few-shot performance of language models. In *International Conference on Machine Learning*, pages 12697–12706. PMLR.

Chujie Zheng and Minlie Huang. 2021. Exploring prompt-based few-shot learning for grounded dialog generation. *arXiv preprint arXiv:2109.06513*.

Chujie Zheng, Yong Liu, Wei Chen, Yongcai Leng, and Minlie Huang. 2021. Comae: a multi-factor hierarchical framework for empathetic response generation. *arXiv preprint arXiv:2105.08316*.

A Prompt For Empathetic Dialogue Generation

Our prompt template for EmpGPT-3 is as follows.

```
The following is a conversation with an empathetic AI assistant. The
assistant empathizes with human experiences and feelings well.

Human:
```

Table 9 and Table 10 show the constructed prompt in the zero-shot and few-shot setting, respectively.

```
The following is a conversation with an empathetic AI assistant. The assistant empathizes with human experiences and feelings well.

Human: I'm so excited because I'm finally going to visit my parents next month! I didn't see them for 3 years
Empathy AI: 3 years is a long time. How come?
Human: I moved to the us and they stayed in my homecountry
Empathy AI: Oh, yeah. That is a long time. I can imagine how excited you are to go home.
```

Table 9: An example of constructed prompt when $k = 0$. The blue represents a generated response from *multi*-EmpGPT-3.

```
The following is a conversation with an empathetic AI assistant. The assistant empathizes with human experiences and feelings well.

Human: When I was eight my parents got me a dog for Christmas.
Empathy AI: A dog is an awesome gift for a little kid. I bet you were super excited.
Human: To this day I remember it as the best Christmas of my life.
Empathy AI: I bet! Maybe I should get my kids a dog for Christmas, I know it would be a great choice for them.

The following is a conversation with an empathetic AI assistant. The assistant empathizes with human experiences and feelings well.

Human: Im already getting pumped up for christmas this year...
Empathy AI: I love Christmas!
Human: Me too especially this year.. new apartment for us new state. Whats your fav part of the holiday?
Empathy AI: Seeing my kids open their gifts. I never make it to Christmas because I'm so excited to see them lol
Human: Its such a good feeling right, the best! :)
Empathy AI: I never get anything but that's fine

The following is a conversation with an empathetic AI assistant. The assistant empathizes with human experiences and feelings well.

Human: I couldn't wait to go to the concert.
Empathy AI: Which Concert?
Human: The U2 concert. Tickets were really expensive and I never thought we would be able to go, but somehow we did!!!
Empathy AI: It was awesome! I love U2.
```

Table 10: An example of constructed prompt when $k = 2$. The blue represents a generated response from *multi*-EmpGPT-3 with EMOSITSM.

B Full Results

Table 11 shows the zero-shot performance ($k = 0$) and few-shot performance ($k = 1, 2$) according to various selection methods (i.e., RANDOM, SITSM, and EMOSITSM) on various automatic evaluation metrics.

		Empathy								Diversity			Fluency	
k	type	INTENTACC	EMOACC	IP	EX	ER	diff-IP	diff-EX	diff-ER	dist-1	dist-2	NIDF	PPL	Avg. Len
0	-	0.2528	0.1594	0.2717	0.4970	0.6439	0.7884	1.2564	0.7295	0.9400	0.9966	0.2840	118.43	15.6
1	RANDOM	0.2682	0.1582	0.2772	0.3683	0.6443	0.8669	1.0412	0.6800	0.9672	0.9984	0.2859	108.84	15.39
	SITSM	0.2599	0.1582	0.2992	0.3643	0.6196	0.9015	1.0491	0.6635	0.9646	0.9982	0.2866	136.97	16.4
	EMOSITSM	0.2689	0.1653	0.2921	0.3337	0.6431	0.8433	0.9941	0.6710	0.9649	0.9980	0.2855	180.95	16.3
2	RANDOM	0.2717	0.1523	0.2740	0.3219	0.6211	0.9046	0.9894	0.6749	0.9688	0.9982	0.2914	125.53	14.84
	SITSM	0.2693	0.1665	0.2466	0.3322	0.6349	0.8057	0.9753	0.6832	0.9689	0.9987	0.2907	136.12	15.15
	EMOSITSM	0.2721	0.1818	0.2733	0.3102	0.6280	0.8528	1.0003	0.6643	0.9661	0.9982	0.2905	83.83	15.51

Table 11: Evaluation results of zero-shot and few-shot learning with different in-context examples $k = 1, 2$ and with various selection methods on various automatic evaluation metrics: **Empathy**, and **Diversity**, **Fluency**.

C Explanation of EPITOME Framework

As we mentioned earlier, EPITOME (Sharma et al., 2020) comprises three mechanisms: IP, EX, and ER. The mechanisms are described as follows:

- EXPLORATIONS (EX) are expressions of active interest in the interlocutor's situation.
- INTERPRETATIONS (IP) are expressions of acknowledgments or understanding of the interlocutor's emotion or situation.
- EMOTIONAL REACTIONS (ER) are expressions of emotions such as warmth, compassion, and concern in the interlocutor's situation.

D A Taxonomy of Empathetic Response Intents

There are 9 categories: Agreeing, Acknowledging, Encouraging, Consoling, Sympathizing, Suggesting, Questioning, Wishing, and Neutral.

E Selected In-Context Examples

Table 12 shows more selected in-context examples.

	Content	Score
Test situation	My eldest son just graduated from High School and I was so happy for him.	-
situation 1	When my brother graduated high school, I was very proud of him, it was a big accomplishment	0.9479
Dialogue Context	S: My brother graduated high school, I was very proud of him! L: I know that feel, my brother graduated a year ago, it's a really big milestone. S: It is, somewhat common, but still, I am proud of him all the same! L: Yes I agree, it really signifies the start of their next chapter in life.	-
situation 2	My son recently graduated from high school. I am so happy about it!	0.9765
Dialogue Context	S: My son recently graduated from high school. L: That's great. What is he doing now? S: He is preparing for college. I am so happy about it! L: That's even more awesome. I hope he does well.	-
Test situation	I have a nest of yellow jackets in my front yard	-
situation 1	I ripped my pants on bourbon street the other day. Luckily I was wearing a long shirt.	0.4424
Dialogue Context	S: I went out last weekend and had a major accident. Guess what happened... L: Are you ok, you have to tell me what happened. S: I'm fine. Just a litte embarassed. I ripped my leggings dancing on bourbon street. L: Ahh that has happened to everyone before. It is embarrassing but you will get over it. S: Yep. My shirt was long enough to cover it. Plus I don't live there lol. L: Well im glad you were able to cover up.	-
situation 2	There's a huge stuffed bear on my yard.	0.4509
Dialogue Context	S: There's a huge stuffed bear on my yard. L: That sounds creepy S: Agree. Not sure what I should do with it. L: I guess ignore it for now	-

(a) Sample of selected in-context example by SITSM.

	Content	Score
Test situation	I had a job interview today and i think it really well.	-
situation 1	I had a great job interview the other day. Im really feeling good about how it went.	0.9284
Dialogue Context	S: I had a great job interview the other day. Im really feeling good about how it went. L: That's fantastic! Hopefully you'll hear something about it soon. S: I should be. I just feel that I did really well. L: I'm sure you did. Think positive!	-
situation 2	I just went on a job interview. I feel like it went really well.	0.9602
Dialogue Context	S: I just got back from a job interview. It went really well. I feel I might get an offer. L: What job did you interview for? S: It was for a Financial Analyst job. I really want the job. L: That's amazing, you must be so excited right now	-
Test situation	I went bowling yesterday and the ball got stuck on my hand. I went with it.	-
situation 1	I was at home and at the last minute my father took me to the Yankee game.	0.3834
Dialogue Context	S: I was at home and last minute my dad took me to the Yankees game. L: That is so thoughtful of him, did you guys catch a ball? S: No, but it was still fun. L: I am sure it was! :)	-
situation 2	I play the lotto regularly and finally won a small amount the other night much to my shock.	0.4339
Dialogue Context	S: I play the lottery sometimes here, and finally won a small amount. Not a big deal, but I was shocked it finally happened haha. L: I would be too, I never win those things! S: Yeah it's really demoralizing until you at least win something. A nice little bonus anyway. L: Very true! Congrats! It's hard to never win	-

(b) Sample of selected in-context example by EMOSITSM when the emotion of "hopeful" (upper row) and "surprised" (bottom row). We also present dialogue contexts corresponding to situation sentences, respectively. (S: Speaker, L: Listener)

Table 12: Samples of selected in-context examples (when $k = 2$) from EMPATHETICDIALOGUES training set, based on the similarity score with a given test situation.

F More Examples of Generated Dialogues

See Table 13 for more examples of responses generated by Blender (Roller et al., 2020) and EmpGPT-3.

Dialogue Context
S: I just got hulu live. It's been amazing to watch tv again! L: I haven't watched actual tv in sooo long. What do you watch on hulu? S: I love to watch home shows on hgtv and bachelor in paridise on abc. Hulu live has been a great value for the price!
Generated Responses
Blender: I love hulu live. I have it on my phone. ***single*-EmpGPT-3:** Can you elaborate?

Dialogue Context
S: I think storms are wonderful, except when the power goes out. L: I love storms too but yeah, that's a nuisance. S: It was extra loud and my husband was at work, bit scary.
Generated Responses
Blender: I'm sorry to hear that. I hope you guys can get it fixed soon. ***single*-EmpGPT-3:** So you felt scared when you heard the loud sound?

Dialogue Context
S: I was hurrying around the house and wasn't paying attention fully, I ending up hitting my toe on my dogs toys. L: Dang dogs leaving their things around! S: It really hurt, I think I may need to take all his toys away to punish him.
Generated Responses
Blender: I would do that. ***multi*-EmpGPT-3:** I can kind of see your reasoning for that, but dogs love their toys.

Dialogue Context
S: I have been looking after my wife this week. She has the flu L: oh poor thing, is she at least getting better? S: It seems so. Sleeping all day today lol
Generated Responses
Blender: I hope she gets better soon! ***multi*-EmpGPT-3:** haha that's so hard. I hope she gets better soon

Table 13: Examples of generated responses from Blender (Roller et al., 2020) and EmpGPT-3 with single- and multi-turn setting on EMPATHETICDIALOGUES test set. (S: Speaker, L: Listener)

G Human Evaluation Questionnaire

This is a list of questions and multiple-choice options for each evaluation metric used in the human evaluation.

- EMPATHY: Do you think this response shows an understanding of situation and empathize appropriately?

  Options: 1: Not at all / 2: A little / 3: Somewhat / 4: A lot

- RELEVANCE: Is this response relevant to given dialogue context?

  Options: 1: Not at all / 2: A little / 3: Somewhat / 4: A lot

- FLUENCY: Does this response seem contextually natural? Could you understand this response?

  Options: 1: Very unnatural / 2: Mostly unnatural / 3: Mostly natural / 4: Very natural

H Human Evaluation System

Figure 3 is a screenshot of human evaluation system.

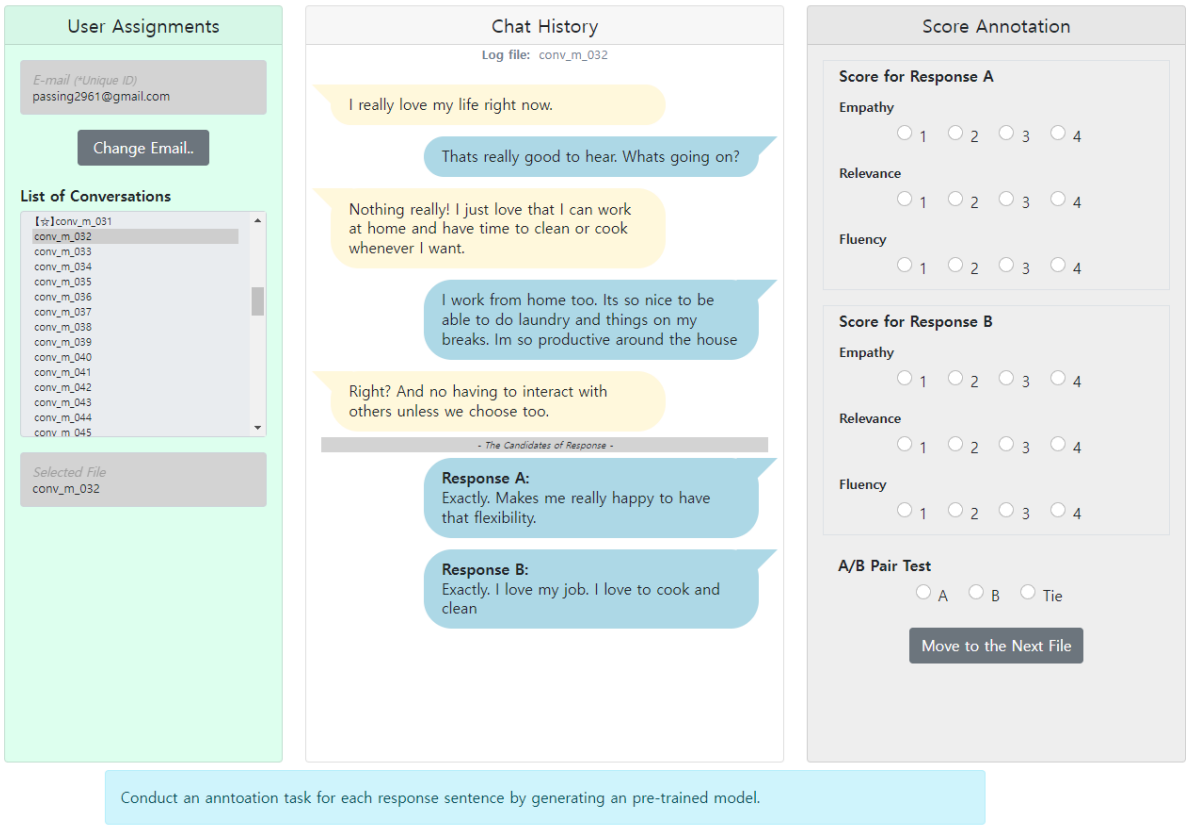

Figure 3: Screenshot of the human evaluation system for empathetic dialogue generation.

DialogueEIN: Emotion Interaction Network for Dialogue Affective Analysis

Yuchen Liu[1], **Jinming Zhao**[2], **Jingwen Hu**[1], **Ruichen Li**[1], **Qin Jin**[1*]
[1] School of Information, Renmin University of China
[2] Qiyuan Lab, Beijing, China
{liuyuchen_alfred, hujingwen_benja, ruichen, qjin}@ruc.edu.cn
zhaojinming@qiyuanlab.com

Abstract

Emotion Recognition in Conversation (ERC) has attracted increasing attention in the affective computing research field. Previous works have mainly focused on modeling the semantic interactions in the dialogue and implicitly inferring the evolution of the speakers' emotional states. Few works have considered the emotional interactions, which directly reflect the emotional evolution of speakers in the dialogue. According to psychological and behavioral studies, the emotional inertia and emotional stimulus are important factors that affect the speaker's emotional state in conversations. In this work, we propose a novel Dialogue Emotion Interaction Network, **DialogueEIN**, to explicitly model the intra-speaker, inter-speaker, global and local emotional interactions to respectively simulate the emotional inertia, emotional stimulus, global and local emotional evolution in dialogues. Extensive experiments on four ERC benchmark datasets, IEMOCAP, MELD, EmoryNLP and DailyDialog, show that our proposed DialogueEIN considering emotional interaction factors can achieve superior or competitive performance compared to state-of-the-art methods. Our codes and models are released[1].

1 Introduction

Emotion Recognition in Conversation (ERC), aiming to recognize the emotional status of each utterance in a conversation, has attracted increasing research attention in recent years. It has rich application potentials in emotional support, mental health, and legal trials etc (Poria et al., 2019).

Unlike traditional emotion recognition based on isolated utterances, conversation context modeling is very important for the ERC task (Poria et al., 2019). Different approaches have been proposed to model the context of a conversation. For example, CMN (Hazarika et al., 2018b) and ICON (Hazarika et al., 2018a) use speaker-specific and global recurrent networks to model the semantic context in a dialogue, and multi-hop memory networks are used to generate the summaries for prediction. DialogueGCN (Ghosal et al.) models the speaker-specific semantic interaction via designing different relations based on graph networks. DialogXL (Shen et al., 2020) utilizes the dialog-aware self-attention mechanism in a transformer structure to capture intra- and inter-speaker dependencies. These works only focus on modeling the dialogue semantic context. Recently, some works have considered the dialogue emotional evolution and proposed several methods to model the emotional context. DialogueRNN (Majumder et al., 2019) and

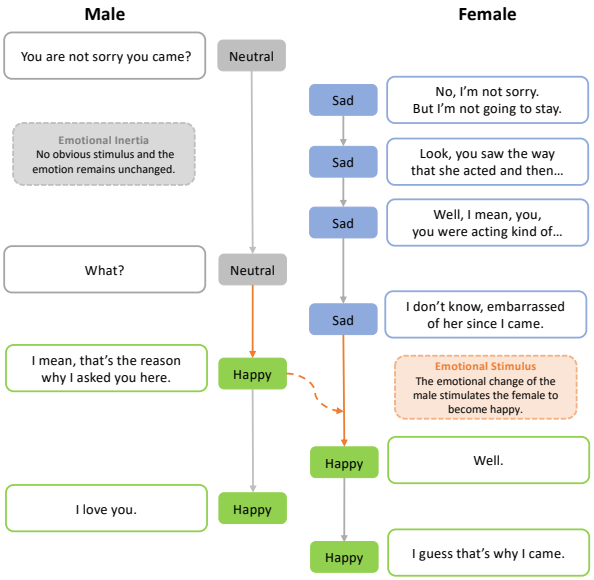

Figure 1: Illustration of emotional interaction in an example dialogue from the IEMOCAP dataset. The grey arrow line indicates that the speaker's emotional state remains unchanged, and the orange arrow line indicates that the speaker is stimulated and the emotion changes, corresponding to emotional inertia and emotional stimulus respectively.

*Corresponding Author
[1]https://github.com/AIM3-RUC/DialogueEIN

Proceedings of the 29th International Conference on Computational Linguistics, pages 684–693
October 12–17, 2022.

COSMIC (Ghosal et al., 2020) use separate GRUs to model global and speaker-specific semantic context and utilize another GRU to track the evolution of global emotional states. CESTa (Wang et al., 2020) adopts the Conditional Random Field (CRF) to learn the global emotional consistency in the conversation. IEIN (Lu et al., 2020) proposes to model the global emotion interaction with emotion embeddings and an RNN-based iterative structure. However, these works only consider modeling the global emotional evolution, while ignoring the important speaker-aware emotional dependencies related to emotional inertia and emotional stimulus.

According to psychological and behavioral studies, emotional inertia and emotional stimulus are important factors that affect the speaker's emotional state in dialogues. Emotional inertia (Kuppens et al., 2010; Koval et al., 2015) means that in the absence of sufficient external stimulus, the speaker tends to keep the emotional state unchanged within a dialogue. Emotional stimulus refers to another phenomenon in which a subject's emotion can be aroused and affected by external events which can be words, facial expression, speech intonation or even emotions of the interlocutor (Brosch et al., 2010). Figure 1 illustrates the emotional inertia and emotional stimulus in an example. In this dialogue, the female speaker feels sad at the beginning because of some misunderstanding, but as the misunderstanding is resolved, both speakers appear to be happy. In the first two turns of the dialogue, the emotional states of the male and female speakers remain unchanged (**emotional inertia**). However, after listening to the explanation, the male speaker shows happiness, and his emotional change stimulates the female speaker to become happy as well (**emotional stimulus**). We simply call such emotional inertia and emotional stimulus information as the emotional interaction context in the dialogue, which measures how a person's emotion affects his own or his interlocutor's emotion.

In this paper, we propose a novel Dialogue Emotion Interaction Network(DialogueEIN), to explicitly model the emotional interaction context in conversations for ERC tasks. DialogueEIN mainly consists of a semantic interaction network and an emotional interaction network, where the former aims to capture dialog-level semantic context representations based on a transformer structure, and the latter aims to model the emotional interaction context including intra-speaker emotional inertia, inter-speaker emotional stimulus, global- and local emotional interactions through four corresponding types of dialog-aware self-attention mechanism respectively. We carry out experiments on four ERC benchmark datasets, including IEMOCAP, MELD, EmoryNLP and DailyDialog. The experiment results show that our proposed DialogueEIN achieves state-of-the-art performance on all datasets, which indicates that emotional interactions (such as emotional inertia and emotional stimulus) are important for tracking speakers' emotional evolution in conversations.

The main contributions of this work include:

- We propose the Dialogue Emotion Interaction Network DialogueEIN, to explicitly model the emotional interaction context for ERC tasks.
- We design an Emotional Interaction Network for modeling the self emotional inertia, interlocutor's emotional stimulus, global and local evolution of emotional states.
- Extensive experimental results show that our proposed DialogueEIN achieves the state-of-the-art performance on different benchmark datasets.

2 Related Work

Emotion Recognition in Conversation

Emotion recognition in conversation has attracted much attention in recent years. There have emerged a number of public emotional dialogue datasets, including IEMOCAP (Busso et al., 2008), MELD (Poria et al., 2018), EmoryNLP (Zahiri and Choi, 2018), DailyDialog (Li et al., 2017) etc.

Different approaches for the ERC task have been proposed as well. Most recent works focus on modeling contextual information in the conversation with different structures. C-LSTM (Poria et al., 2017) uses an simple LSTM-based model to encode the global context in a conversation. CMN (Hazarika et al., 2018b) and ICON (Hazarika et al., 2018a) propose structures based on the gated recurrent unit (GRU) and the memory network to capture both global and speaker-specific context information. DialogueGCN (Ghosal et al.) constructs a graph regarding to both temporal and speaker-aware relationship in the dialogue and model the semantic interactions with a relation-aware graph-based network. DAG-ERC (Shen et al., 2021)

constructs a directed acyclic graph and utilizes a graph-based network to model the information flow in the conversation chronologically, which combine the strengths of conventional graph-based and recurrence-based neural models. DialogXL (Shen et al., 2020) uses an XLNet-based structure, improves memory mechanisms of XLNet and proposes four kinds of dialog-aware attention mechanism to encode corresponding semantic context information in the dialogue.

Above mentioned approaches only focus on modeling the semantic context in the conversation, while some other works consider the emotional context and model the global evolution of emotion states with different methods. DialogueRNN (Majumder et al., 2019) employs several GRUs to track the evolution of different states in the dialogue, including the global emotional state. COSMIC (Ghosal et al., 2020) uses a similar structure to track more kinds of dialog-aware states and introduces external commonsense knowledge to improve the performance. CESTa (Wang et al., 2020) introduces Conditional Random Field (CRF) to learn the emotional consistency in the dialogue. IEIN (Lu et al., 2020) utilizes emotion embeddings and an RNN-based interactive structure to model the global emotion interaction in the conversation.

Our framework is closely related to DialogXL and IEIN, where DialogXL proposes to model speaker-aware context information with attention mechanism, and IEIN proposes to model global emotion interaction based on emotion embeddings. DialogueEIN differs from them from the following two aspects: (1) DialogXL uses speaker-aware attention to model the fine-grained word-level semantic interactions, whereas DialogueEIN focuses on modeling the utterance-level speaker-aware emotional interactions, which explicitly tracks the emotion evolution in the conversation rather than the semantic context. (2) IEIN only considers the global emotional dependencies in the conversation, but ignores the emotional inertia of speakers themselves and the emotional stimulus between interlocutors, whereas DialogueEIN models these two types of speaker-aware emotional interaction explicitly.

Label Embeddings

The label embeddings are embedding vectors which are trained to learn the latent knowledge about the label categories in classification tasks. They can be considered as the representations of label categories, where each label embedding vector represents one output label category.

The idea of label embeddings has been widely used in various tasks, including multi-class classification (Bengio et al., 2010), zero-short learning (Larochelle et al., 2008) , text classification (Tang et al., 2015) and sequence labeling (Cui and Zhang, 2019). Cui and Zhang (2019) employs label embeddings and builds a hierarchical label attention network to model the dependencies between output labels for sequence labeling task. Lu et al. (2020) introduces the idea of label embeddings into ERC task to learn the knowledge about emotion labels and model the global emotional interaction. Inspired by these works, we employ label embeddings in our proposed DialogueEIN as well to represent different emotion categories, which is called emotion embeddings.

3 Method

3.1 Problem Definition

A dialogue can be defined as a sequence of utterances, $\{u_1, u_2, ..., u_N\}$, where N is the total number of utterances. Each utterance u_j contains n_j words, $\{w_1^j, w_2^j, ..., w_{n_j}^j\}$, and uttered by speaker $p(u_j)$, where p is a mapping from utterances to corresponding speakers. Each utterance is labeled with a type of emotion y_j, the task is to predict the emotion label of each utterance in a dialogue.

Figure 2 illustrates the overall framework of our proposed DialogueEIN, which consists of four key components: Utterance-level Feature Extraction, Semantic Interaction Network, Emotional Interaction Network and Emotion Classification.

3.2 Utterance-level Feature Extraction

We employ a pre-trained RoBERTa (Liu et al., 2019) model to extract utterance-level features. Each utterance u_j is padded with a special token $[CLS]$ and fed into the RoBERTa model:

$$X_j = \text{RoBERTa}([CLS], w_1^j, w_2^j, ..., w_{n_j}^j) \quad (1)$$

where $X_j \in \mathbb{R}^{(n_j+1)\times d_b}$ is the output of the last hidden layer of the RoBERTa model and d_b is the hidden size of the RoBERTa model. We take the hidden state at the $[CLS]$ position of X_j and pass it into a linear layer to get the utterance-level feature representation of u_j, which is formulated as follows:

$$x_j = W_u X_{j,0} + b_u \quad (2)$$

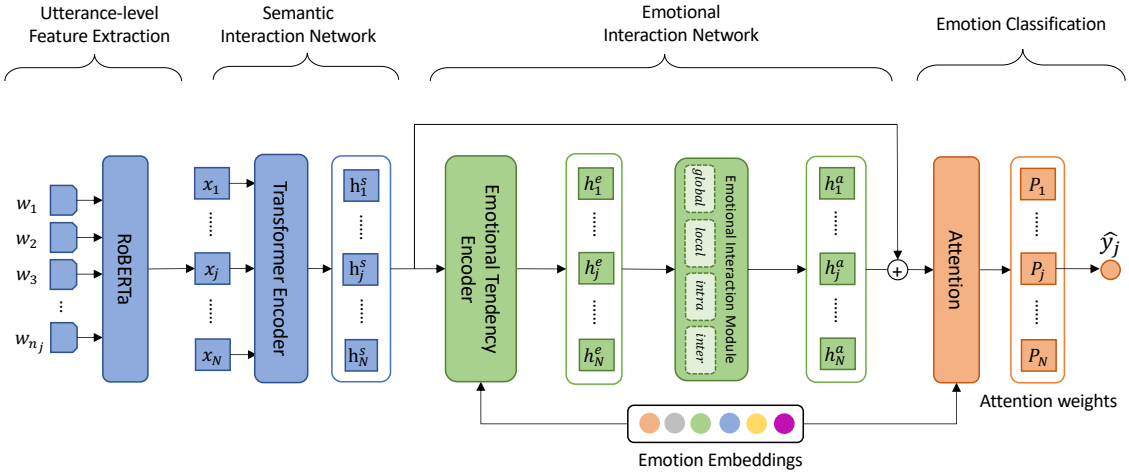

Figure 2: Framework illustration of DialogueEIN, which consists of four key components: Utterance-level Feature Extraction, Semantic Interaction Network, Emotional Interaction Network, Emotion Classification.

where $W_u \in \mathbb{R}^{d_b \times d_u}$, $b_u \in \mathbb{R}^{d_u}$ are learnable parameters and d_u is the dimension of utterance-level feature representations. x_j is the utterance-level feature representation of u_j.

3.3 Semantic Interaction Network

Since the semantics of each utterance is naturally influenced by other utterances in the dialogue, it is necessary to capture global semantic interaction context of a dialogue. Specifically, in the semantic interaction network, we employ a Transformer (Vaswani et al., 2017) encoder to model semantic interaction in a dialogue.

Given the feature representations of utterances in the dialogue $[x_1, x_2, ..., x_N]$, they are added with a Sinusoidal Position Encoding and then fed into the Transformer encoder. The overall semantic interaction network can be formulated as follows:

$$h^0 = [x_1, x_2, ..., x_N] + \text{PosEnc}(0:N) \quad (3)$$

$$h^s = \text{TRMEncoder}(h^0) \quad (4)$$

where TRMEncoder denotes the transformer encoder model. h^s is the semantic feature representation of the dialogue, which contains the global semantic context information in the dialogue.

3.4 Emotional Interaction Network

As mentioned in the introduction, we believe that emotional interaction context, including emotional inertia and emotional stimulus, can benefit the emotion recognition in conversation. We propose an Emotional Interaction Network to model this kind of emotional interaction context, which contains an Emotional Tendency Encoder and an Emotional Interaction Module. Specifically, the Emotional Tendency Encoder can encode the emotional representation of each utterance, which reflects its emotional tendency. Based on these emotional representations, the Emotional Interaction Module models the emotional interactions.

3.4.1 Emotional Tendency Encoder

Inspired by Cui and Zhang (2019) and (Lu et al., 2020), we use emotion embeddings to represent candidate emotion categories and employ a multi-head attention module to capture the emotional tendency of each utterance.

Given the set of candidate emotion labels L = $\{l_1, l_2, ..., l_{|L|}\}$, each label is represented with an embedding:

$$e_i = E^l(l_i) \quad (5)$$

where E^l denotes the emotion embedding lookup table and e^i denotes the embedding of the i-th emotion category. As is shown in the structure of Emotional Embeddings in Figure 2, the circles with different colors represent the embeddings of different emotion categories (e.g. happy, sad, anger). These embeddings are initialized randomly and tuned during the model training to learn the latent knowledge about corresponding emotion categories, and can be regarded as the representations of them. The dimension of emotion embeddings is the same as the utterance-level representation, i.e., $e_i \in \mathbb{R}^{d_u}$.

Given the semantic representation h^s and emotion embeddings $e = [e_1, e_2, ..., e_{|L|}]$, a multi-head attention module is applied to them, with h^s as the query and e as the key and the value in the attention mechanism. It is formulated as:

$$h^e = \text{MHA}(h^s, e, e) + h^s \tag{6}$$

Specifically, the multi-head attention module is formulated as:

$$\text{A}(Q, K, V, M) = \text{softmax}(\frac{QK^T}{\sqrt{d_h}} + M)V \tag{7}$$

$$\text{MHA}(Q, K, V, M) = \text{Concat}(head_1, ..., head_n)W^O \tag{8}$$

$$head_i = \text{A}(QW_i^Q, KW_i^K, VW_i^V, M) \tag{9}$$

where d_h denotes the dimension of each attention head, n denotes the number of attention heads, M denotes an attention mask matrix whose elements take value from $\{0, -\infty\}$, and W_i^Q, W_i^K, W_i^V, W^O are trainable parameters. The mask matrix M is set to a null matrix by default.

Since h^e is the result of linear transformations and linear combinations of emotion representations, it contains the explicit emotional information regarding to each utterance and can indicate the emotional tendencies of utterances explicitly. In addition, a residual connection from h^s is added to h^e, which means h^e can not only represent the emotional tendency of each utterance, but also carry the semantic information.

3.4.2 Emotional Interaction Module

We propose an attention-based module, Emotional Interaction Module, to model the emotional interactions in the conversation. Inspired by (Shen et al., 2020), in order to capture different dependencies and interactions in the conversation, we apply different attention masks to the Emotional Interaction Module.

Specifically, there are four types of attention mechanism are employed, including intra-speaker, inter-speaker, global and local attention, which model the emotional inertia of speakers, the emotional stimulus between interlocutors, the global and local emotional evolution in the dialogue, respectively. The emotional interaction module is formulated as follows:

$$h^{ei} = \text{Concat}\{\text{MHA}_{EI}(h^e, h^e, h^e, m) | m \in M\} \tag{10}$$

$$h^a = \text{LayerNorm}(h^{ei}W^a + b_a + h^s) \tag{11}$$

where M=$\{m^{global}, m^{local}, m^{intra}, m^{inter}\}$ denotes global, local, intra-speaker and inter-speaker attention mask respectively, MHA_{EI} denotes a multi-head attention module, W^a and b_a are trainable parameters.

The formulations and introductions of the four types of self attention masks are shown as follows:

(1) Global Attention Mask: It works the same as the original self-attention mechanism, where each utterance attends to all the utterances in the conversation. Global attention mask is formulated as follows:

$$m_{i,j}^{global} = 0 \tag{12}$$

(2) Local Attention Mask: Each utterance attends to the adjacent utterances within a local window around it. Local attention mask is formulated as follows:

$$m_{i,j}^{local} = \begin{cases} 0, & if\ |i-j| < w/2 \\ -\infty, & otherwise \end{cases} \tag{13}$$

where w is the window size.

(3) Intra-speaker Attention Mask: Each utterance from one speaker only attends to the utterances from the same speaker. Intra-speaker attention mechanism aggregates the emotional state information from each speaker themselves in the dialogue, which models the emotional inertia of them. Intra-speaker attention mask is formulated as follows:

$$m_{i,j}^{intra} = \begin{cases} 0, & if\ p(u_i) = p(u_j) \\ -\infty, & otherwise \end{cases} \tag{14}$$

(4) Inter-speaker Attention Mask: Each utterance from one speaker only attends to the utterances from their interlocutors, contrary to intra-speaker attention. For each speaker's utterance, Inter-speaker attention mechanism aggregates the emotional state information from the interlocutors in the dialogue, which models the emotional stimulus between the speaker and the interlocutors. Inter-speaker attention mask can be formulated as follows:

$$m_{i,j}^{inter} = \begin{cases} 0, & if\ p(u_i) \neq p(u_j)\ or\ i = j \\ -\infty, & otherwise \end{cases} \tag{15}$$

Dataset	dialogues			utterances		
	train	val	test	train	val	test
IEMOCAP	100	20	31	4830	980	1623
MELD	1038	114	280	9989	1109	2610
EmoryNLP	713	99	85	9934	1344	1328
DailyDialog	11118	1000	1000	87170	8069	7740

Table 1: Data distribution of the four datasets.

3.5 Emotion Classification

In order to match the emotion embeddings with corresponding emotion categories, we employ an attention module as the classifier. Given the final representation of utterances h^e and emotion embeddings $e = [e_1, e_2, ..., e_{|L|}]$, the emotion classifier is formulated as follows:

$$P_j = \text{softmax}(e^T W^c h^a) \quad (16)$$
$$\hat{y}_j = \text{argmax}(P_{1:|L|,j}) \quad (17)$$

where $P_j \in \mathbb{R}^{|L| \times N}$ is the attention weights, W^c is the learnable parameter and $\hat{y}_j$ is the prediction of the j-th utterance. We regard the attention weights P_j as the predicted probability distribution of emotion labels and directly make predictions based on it. Thus, an utterance with higher attention weight to the emotion embedding vector e_l is more likely to be classified as emotion label l, which ensures the matching of emotion embeddings and emotion categories.

Cross-entropy loss is used for model training:

$$L = -\frac{1}{\sum_{k=1}^{T} N(k)} \sum_{i=1}^{T} \sum_{j=1}^{N(i)} \log P_{i,j}[y_{i,j}] \quad (18)$$

where T is the total number of dialogues, $N(i)$ is the number of utterances in dialogue i, $y_{i,j}$ and $P_{i,j}$ denote the expected emotion label and the probability distribution of predicted emotion labels of the j-th utterance in dialogue i respectively.

4 Experiments

4.1 Datasets

We carry out evaluations on four ERC benchmark datasets. The distribution of samples in training set, validation set and testing set of these datasets is presented in Table 1. Since IEMOCAP dataset has no validation set, following Shen et al. (2020), we retain the last 20 dialogues in the training set as validation.

IEMOCAP: The dataset (Busso et al., 2008) contains 151 two-way conversations from ten speakers in a normal or improvisational way given certain scripts. Each utterance is annotated with an emotion label from six classes, including happy, sad, neutral, angry, excited and frustrated.

MELD: The dataset (Poria et al., 2018) contains multi-modal and multi-speaker conversational dialogues from the TV show *Friends*. There are usually three or more speakers in a single conversation. Each utterance is annotated with an emotion label from seven classes, including anger, disgust, fear, joy, neutral, sadness and surprise.

EmoryNLP: The dataset (Zahiri and Choi, 2018) is another corpus collected from the TV show *Friends*, which also usually contains more than two speakers in a conversation. Each utterance is annotated with an emotion label from seven classes, including neutral, sad, mad, scared, powerful, peaceful and joyful.

DailyDialog: The dataset (Li et al., 2017) collects human-written dyadic conversations from English learning websites with concentrated topics and regulated grammar. Each utterance is annotated with an emotion label from seven classes, including anger, disgust, fear, happiness, sadness, surprise and other.

4.2 Implementation Details

We initialize the utterance-level feature extractor with pre-trained RoBERTa models. Specifically, RoBERTa-large model is utilized on MELD, EmoryNLP and DailyDialog datasets. Since the amount of data contained in IEMOCAP dataset is relatively small, we also use a smaller model, RoBERTa-base, on IEMOCAP dataset and only fine-tune the last 4 layers of it during training. We employ an AdamW optimizer (Loshchilov and Hutter, 2018) and a linear learning rate scheduler for model training. The hyper-parameters are tuned on validation set, and two sets of hyper-parameters are used for IEMOCAP and the other three datasets respectively. Specifically, the learning rate of RoBERTa is {2e-5, 5e-6}, the learning rate of the other modules is {1e-4, 5e-5}, the dropout rate is {0.1, 0.1}, the dimension of utterance-level features is {384, 512}, the dimension of feedforward layers is {1024, 2048}, the number of attention heads is {6, 8}, the layers of TransformerEncoder is {4, 4}, the local window size is {15, 5} respectively. The results reported in

the following experiments are based on the average score of 10 random runs.

4.3 Comparison with State-of-the-art Methods

We compare our proposed DialogueEIN model to the following state-of-the-art methods. **KET** (Wang et al., 2020) uses a transformer-based structure and external commonsense knowledge to capture the semantic context. **DialogueGCN** (Ghosal et al.) uses graph-based networks to capture conversational dependencies between utterances in dialogues. **DialogueGCN+RoBERTa** means using features based on a more efficient feature extractor RoBERTa instead of GloVe features in DialogueGCN. **DialogXL** (Shen et al., 2020) applies a strong pre-trained language model XLNet (Yang et al., 2019) and proposes a dialog-aware self-attention method for modeling the semantic context information. **DAG-ERC** (Shen et al., 2021) constructs a directed acyclic graph and DAGNN (Thost and Chen, 2021) to model the temporal and speaker-aware semantic context. **DialogueRNN** (Majumder et al., 2019) uses several distinct GRUs to model the speaker-specific and global semantic context and another GRU to model the global emotional interaction. It is the first work considering emotional interaction for ERC. **DialogueRNN+RoBERTa** means using features based on a more efficient feature extractor RoBERTa instead of the n-gram features in DialogueRNN. **COSMIC** (Ghosal et al., 2020) proposes a GRU-based structure and uses external commonsense knowledge to capture the semantic context and another GRU to model the global emotional interaction. **CESTa** (Wang et al., 2020) proposes a Transformer- and LSTM-based structure to capture the semantic context and leverages conditional random field (CRF) to model global emotional interaction in conversations.

Table 2 presents the experimental results on the IEMOCAP, MELD, EmoryNLP and DailyDialog four benchmark datasets. DialogueEIN significantly outperforms all other state-of-the-art methods and achieves a new state-of-the-art performance on the IEMOCAP and MELD datasets, which demonstrates its effectiveness of modeling the semantic and emotional context in the dialogue. Please note that the methods in the second block of Table 2 consider global emotional interaction as well. DialogueEIN clearly outperforming these methods indicates that our proposed emotional interaction network with four different emotional interactions can better capture the emotional context. DialogueEIN achieves competitive performance with DAG-ERC on EmoryNLP, which may relate to the fact that dialogues in EmoryNLP are short (3 to 5 utterances on average) and have fewer variations in emotional states, therefore, it is simpler to model by traditional semantic modeling. Additionally, DialogueEIN performs slightly worse than CESTa on DailyDialog, mainly because the dialogues in DailyDialog are short and there are more than 80% of "neutral" emotional states in the dataset.

	IEMOCAP	MELD	EmoryNLP	DailyDialog
	Avg(w)	Avg(w)	Avg(w)	Avg(micro)
KET	59.56	58.18	33.95	53.37
DialogueGCN	64.18	58.10	-	-
+RoBERTa	64.91	63.02	38.10	57.52
DialogXL	65.94	62.41	34.73	54.93
DAG-ERC	68.03	63.65	**39.02**	59.33
DialogueRNN	62.75	57.03	-	-
+RoBERTa	64.76	63.61	37.44	57.32
COSMIC	65.25	65.21	38.11	58.48
CESTa	67.10	58.36	-	**63.12**
DialogueEIN(**Ours**)	**68.93**	**65.37**	38.92	62.58

Table 2: ERC performance of different models on four datasets. Micro average F1-score (Avg(micro)) is used on DailyDialog, with the neutral labels excluded. Weighted average F1-score (Avg(w)) is used on other three datasets.

	IEMOCAP	MELD
DialogueEIN	**68.93**	**65.37**
- intra-speaker attention	68.63 (0.30↓)	64.89 (0.48↓)
- inter-speaker attention	68.43 (0.40↓)	65.10 (0.27↓)
- intra-&inter-speaker attention	67.36 (1.57↓)	64.84 (0.51↓)
- global&local attention	67.71 (1.22↓)	64.70 (0.67↓)
- Emotional Tendency Encoder	67.68 (1.25↓)	64.85 (0.52↓)
- Emotional Interaction Network	66.04 (**2.89**↓)	64.59 (**0.78**↓)

Table 3: Ablation Study of DialogueEIN on IEMOCAP and MELD datasets.

4.4 Ablation of DialogueEIN

We conduct experiments to ablate the contributions of different components in Emotional Interaction Network, including global, local, intra-speaker, inter-speaker emotional interactions and Emotional Tendency Encoder. The results are shown in Table 3. We can observe that the performance declines obviously when removing each or part of these emotional interactions, which shows that the four emotional interactions are beneficial to ERC. In addition, when the Emotional Tendency Encoder is removed, Emotional Interaction Network would lose the ability to model the emotional context in the conversation, and result in modeling the inter-

		IEMOCAP	MELD
1	RoBERTa	63.38	62.88
2	+TRM	66.04	64.59
3	+TRM&Attentions	67.55	64.95
4	+TRM&CRF	67.11	64.62
5	+TRM&Attentions&CRF	67.76	64.69
6	DialogueEIN	**68.93**	**65.37**

Table 4: Comparasion with RoBERTa-based baselines on IEMOCAP and MELD datasets. We adjust the number of transformer layers so that these models have about the same number of parameters with DialogueEIN.

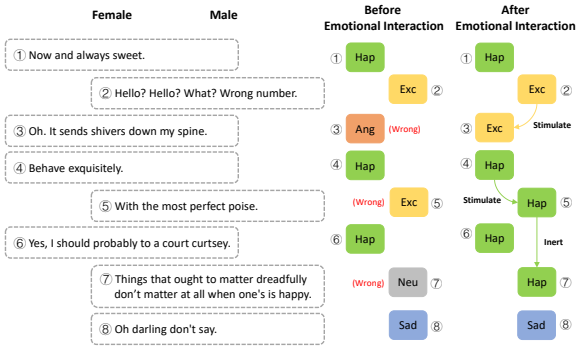

Figure 3: A case study based on IEMOCAP dataset.

actions only based on semantic context. The decline in performance proves that the modeling of emotional context plays an important role in DialogueEIN. Especially, removing the Emotional Interaction Network leads to the worst performance in Table 3. Additionally, the results show that the influence of the Emotional Interaction Network on IEMOCAP and MELD is different (2.9 $\downarrow$ vs 0.78 $\downarrow$), which may be related to the long context (50 utterances per dialogue on average) and the complex emotional evolution of the dialogue in IEMOCAP. It indicates that our proposed Emotional Interaction Network can model long and complex dialogues better.

4.5 Comparison with RoBERTa-based Baselines

We adopt a pre-trained language model, RoBERTa, as the utterance-level feature extractor in DialogueEIN. In order to prove that the improvement of DialogueEIN does not come from the increase in the number of parameters and the enhancement of the feature extractor, we propose some RoBERTa-based baselines for comparison: (1) **RoBERTa**: concatenating utterances and feeding them into RoBERTa. (2) **RoBERTa+TRM**: using RoBERTa to extract utterance representations, and feeding them into a transformer encoder. (3) **RoBERTa+TRM+Attentions**: applying dialog-aware attention masks to the transformer encoder. (4) **RoBERTa+TRM+CRF**: following CESTa, adding a CRF layer after the classifier to model the emotional consistency in the dialogue. (5) **RoBERTa+TRM+Attentions+CRF**: further adding dialog-aware attentions into baseline (4).

The results are shown in Table 4. DialogueEIN outperforms all the above mentioned baselines. Comparing row 6 to row 1, the large improvement of DialogueEIN proves that it exploits the full potential of RoBERTa. Comparing row 6 to row 2 and 3, with comparable number of parameters, DialogueEIN outperforms the simple hierarchical transformer structure and this structure with dialog-aware attentions which can model the speaker-specific semantic context. It demonstrates the importance of emotional interaction context modeling in DialogueEIN. Comparing row 6 to row 4 and 5, the results show that CRF can improve the performance to a certain degree in general, especially on IEMOCAP dataset. However, DialogueEIN still outperforms these models, which prove that Emotional Interaction Network is better than CRF in modeling emotional context.

4.6 Case Study

Figure 3 shows a conversation from IEMOCAP dataset, and the emotional interaction modeled in DialogueEIN. We extract the attention scores of Emotional Tendency Encoder on these utterances, and use the emotion with maximal score to represent the emotional tendency of each utterance. We also provide the final predictions of these utterances, which are correct predictions as the ground truth. They can represent the emotion prediction of these utterances in DialogueEIN before and after emotional interaction, respectively. As shown in Figure 3, the emotion tendencies before emotional interaction of the 3rd, 5th and 7th utterances are different from the final prediction. We illustrate the process that DialogueEIN corrects these errors according to emotional interactions.

(1) The literal meaning of the 3rd utterance can be considered as *angry* or *excited*, and the model regards is as *angry* before emotional interaction. However, considering the emotional stimulus from the 2nd utterance which is identified as *excited*, it is more rational to identify it as *excited* instead of *angry*.

(2) The 5th utterance contains a certain positive emotion literally, but it's hard to distinguish between *happy* or *excited*. However, the female speaker changes to *happy* at the 4th utterance, and it stimulates the male speaker's emotion to be *happy* at the 5th utterance.

(3) The 7th utterance has no obvious emotional tendency literally, and it is regarded as *neutral* before emotional interaction. But when the emotion of the 5th utterance is correctly recognized as *happy* and there is no external emotional stimulus from the 6th utterance, the 7th utterance is finally identified as *happy* according to the emotional inertia.

The above case indicates that DialogueEIN can make more accurate emotion prediction by modeling emotional interaction.

5 Conclusion

In this work, we propose a novel emotional interaction Network (DialogueEIN) for Emotion Recognition in Conversation (ERC). DialogueEIN explicitly models emotional inertia, emotional stimulus in a conversation, which most previous works have neglected. DialogueEIN can capture the emotion tendencies of each utterance and model the emotional dependencies based on them. An attention-based Emotional Interaction Network is proposed to measure the emotional interactions, and four types of dialog-aware attentions are employed to simulate emotional inertia, emotional stimulus, global and local evolution of emotional states in the dialogue respectively. Extensive experiments are carried out on IEMOCAP, MELD, EmoryNLP and DailyDialog benchmark datasets. DialogueEIN significantly outperforms other state-of-the-art models on IEMOCAP and MELD datasets, and achieves competitive performance on all four datasets, which proves the effectiveness of the proposed model. Moreover, several ablation studies further explore the structure of DialogueEIN and interpret the use of emotional interaction, which also suggests possible future research directions, such as fusing multimodality to capture emotional stimulus information more accurately, etc.

6 Acknowledgments

This work was partially supported by the National Key R&D Program of China (No. 2020AAA0108600), the National Natural Science Foundation of China (No. 62072462).

References

Samy Bengio, Jason Weston, and David Grangier. 2010. Label embedding trees for large multi-class tasks.

Tobias Brosch, Gilles Pourtois, and David Sander. 2010. The perception and categorisation of emotional stimuli: A review. *Cognition and emotion*, 24(3):377–400.

Carlos Busso, Murtaza Bulut, Chi-Chun Lee, Abe Kazemzadeh, Emily Mower, Samuel Kim, Jeannette N Chang, Sungbok Lee, and Shrikanth S Narayanan. 2008. Iemocap: Interactive emotional dyadic motion capture database. *Language resources and evaluation*, 42(4):335.

Leyang Cui and Yue Zhang. 2019. Hierarchically-refined label attention network for sequence labeling. *arXiv preprint arXiv:1908.08676*.

Deepanway Ghosal, Navonil Majumder, Alexander Gelbukh, Rada Mihalcea, and Soujanya Poria. 2020. Cosmic: Commonsense knowledge for emotion identification in conversations. *arXiv preprint arXiv:2010.02795*.

Deepanway Ghosal, Navonil Majumder, Soujanya Poria, Niyati Chhaya, and Alexander Gelbukh. Dialoguegcn: A graph convolutional neural network for emotion recognition in conversation.

Devamanyu Hazarika, Soujanya Poria, Rada Mihalcea, Erik Cambria, and Roger Zimmermann. 2018a. Icon: Interactive conversational memory network for multimodal emotion detection. In *Proceedings of the 2018 conference on empirical methods in natural language processing*, pages 2594–2604.

Devamanyu Hazarika, Soujanya Poria, Amir Zadeh, Erik Cambria, Louis-Philippe Morency, and Roger Zimmermann. 2018b. Conversational memory network for emotion recognition in dyadic dialogue videos. In *Proceedings of the conference. Association for Computational Linguistics. North American Chapter. Meeting*, volume 2018, page 2122. NIH Public Access.

Peter Koval, Annette Brose, Madeline L Pe, Marlies Houben, Yasemin Erbas, Dominique Champagne, and Peter Kuppens. 2015. Emotional inertia and external events: The roles of exposure, reactivity, and recovery. *Emotion*, 15(5):625.

Peter Kuppens, Nicholas B Allen, and Lisa B Sheeber. 2010. Emotional inertia and psychological maladjustment. *Psychological science*, 21(7):984–991.

Hugo Larochelle, Dumitru Erhan, and Yoshua Bengio. 2008. Zero-data learning of new tasks. In *AAAI*, volume 1, page 3.

Yanran Li, Hui Su, Xiaoyu Shen, Wenjie Li, Ziqiang Cao, and Shuzi Niu. 2017. Dailydialog: A manually labelled multi-turn dialogue dataset. *arXiv preprint arXiv:1710.03957*.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Roberta: A robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692*.

Ilya Loshchilov and Frank Hutter. 2018. Fixing weight decay regularization in adam.

Xin Lu, Yanyan Zhao, Yang Wu, Yijian Tian, Huipeng Chen, and Bing Qin. 2020. An iterative emotion interaction network for emotion recognition in conversations. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 4078–4088.

Navonil Majumder, Soujanya Poria, Devamanyu Hazarika, Rada Mihalcea, Alexander Gelbukh, and Erik Cambria. 2019. Dialoguernn: An attentive rnn for emotion detection in conversations. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, pages 6818–6825.

Soujanya Poria, Erik Cambria, Devamanyu Hazarika, Navonil Majumder, Amir Zadeh, and Louis-Philippe Morency. 2017. Context-dependent sentiment analysis in user-generated videos. In *Proceedings of the 55th annual meeting of the association for computational linguistics (volume 1: Long papers)*, pages 873–883.

Soujanya Poria, Devamanyu Hazarika, Navonil Majumder, Gautam Naik, Erik Cambria, and Rada Mihalcea. 2018. Meld: A multimodal multi-party dataset for emotion recognition in conversations. *arXiv preprint arXiv:1810.02508*.

Soujanya Poria, Navonil Majumder, Rada Mihalcea, and Eduard Hovy. 2019. Emotion recognition in conversation: Research challenges, datasets, and recent advances. *IEEE Access*, 7:100943–100953.

Weizhou Shen, Junqing Chen, Xiaojun Quan, and Zhixian Xie. 2020. Dialogxl: All-in-one xlnet for multi-party conversation emotion recognition. *arXiv preprint arXiv:2012.08695*.

Weizhou Shen, Siyue Wu, Yunyi Yang, and Xiaojun Quan. 2021. Directed acyclic graph network for conversational emotion recognition. *arXiv preprint arXiv:2105.12907*.

Jian Tang, Meng Qu, and Qiaozhu Mei. 2015. Pte: Predictive text embedding through large-scale heterogeneous text networks. In *Proceedings of the 21th ACM SIGKDD international conference on knowledge discovery and data mining*, pages 1165–1174.

Veronika Thost and Jie Chen. 2021. Directed acyclic graph neural networks. *arXiv preprint arXiv:2101.07965*.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in neural information processing systems*, pages 5998–6008.

Yan Wang, Jiayu Zhang, Jun Ma, Shaojun Wang, and Jing Xiao. 2020. Contextualized emotion recognition in conversation as sequence tagging. In *Proceedings of the 21th Annual Meeting of the Special Interest Group on Discourse and Dialogue*, pages 186–195.

Zhilin Yang, Zihang Dai, Yiming Yang, Jaime Carbonell, Russ R Salakhutdinov, and Quoc V Le. 2019. Xlnet: Generalized autoregressive pretraining for language understanding. *Advances in neural information processing systems*, 32.

Sayyed M Zahiri and Jinho D Choi. 2018. Emotion detection on tv show transcripts with sequence-based convolutional neural networks. In *Workshops at the thirty-second aaai conference on artificial intelligence*.

Towards Enhancing Health Coaching Dialogue in Low-Resource Settings

Yue Zhou [1], Barbara Di Eugenio[1], Brian Ziebart[1], Lisa Sharp[1], Bing Liu[1], Ben Gerber[2], Nikolaos Agadakos[1] and Shweta Yadav[1]

[1]University of Illinois at Chicago, IL, USA
[2]UMass Chan Medical School, MA, USA
{yzhou232,bdieugen,bziebart,sharpl,liub,nagada2,shwetay}@uic.edu
{ben.gerber}@umassmed.edu

Abstract

Health coaching helps patients identify and accomplish lifestyle-related goals, effectively improving the control of chronic diseases and mitigating mental health conditions. However, health coaching is cost-prohibitive due to its highly personalized and labor-intensive nature. In this paper, we propose to build a dialogue system that converses with the patients, helps them create and accomplish specific goals, and can address their emotions with empathy. However, building such a system is challenging since real-world health coaching datasets are limited and empathy is subtle. Thus, we propose a modularized health coaching dialogue system with simplified NLU and NLG frameworks combined with mechanism-conditioned empathetic response generation. Through automatic and human evaluation, we show that our system generates more empathetic, fluent, and coherent responses and outperforms the state-of-the-art in NLU tasks while requiring less annotation. We view our approach as a key step towards building automated and more accessible health coaching systems.

1 Introduction

Health coaching is a patient-centered, motivational interviewing-based clinical practice that focuses on helping patients identify and accomplish personalized, lifestyle-related goals to improve health behaviors. It has been effective in improving the control of chronic conditions such as diabetes and cardiovascular disease and mitigating mental health conditions such as anxiety and depression (Butterworth et al., 2006; Ghorob, 2013; Kivelä et al., 2014; Thom et al., 2016). Health coaching can be particularly beneficial to low-socioeconomic status (SES) populations who disproportionately suffer physical and mental disease burdens (Thackeray et al., 2004; Kangovi et al., 2014). Yet, it is invariably cost-prohibitive for these populations due to its highly personalized and labor-intensive nature.

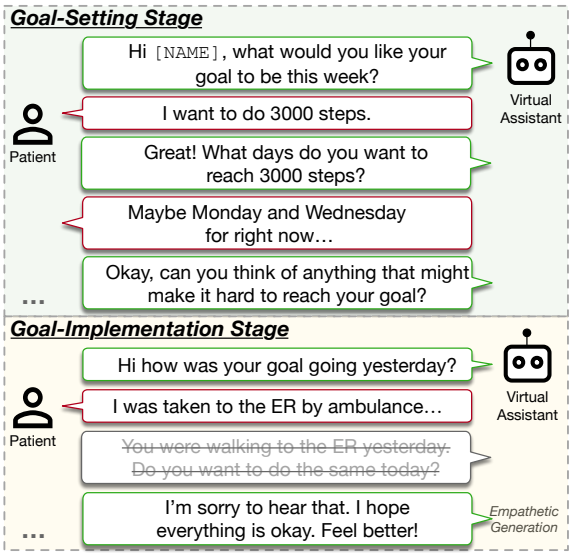

Figure 1: A health coaching dialogue scenario in our dialogue system. It starts with a goal-setting stage where the coach discusses a realistic goal with the patient. After the goal is settled, the coach follows up on the patient's progress and maintains patient engagement. Understanding the patient's emotion cues and responding empathetically is also crucial in such a scenario; the struck-out response, generated by a naïve sequence-to-sequence model, is inappropriate without the capability of understanding and modeling empathy.

Facilitating conversations in healthcare settings between the participants via language techniques and text messages (Aguilera and Muñoz, 2011; Fitzpatrick et al., 2017) has the potential to improve the efficacy of health coaching while reducing the cost. However, the interactions between the dialogue systems and patients are either scripted or with limited natural language understanding (NLU) or generation (NLG) capabilities (Kocielnik et al., 2018; Chaix et al., 2020; Mohan et al., 2020). In our previous work Gupta et al. (2020a,b, 2021), we collected real-world health coaching conversation datasets and focused on the NLU components of the dialogue. However, no existing health coaching dialogue system supports natural language conver-

Proceedings of the 29th International Conference on Computational Linguistics, pages 694–706
October 12–17, 2022.

sations between the patient and a coach agent.

In this paper, we propose to build a dialogue system that converses with patients and helps them create and accomplish specific goals for physical activities based on the health coaching dataset we previously collected (Gupta et al., 2020b). We want the system to emulate the health coaching process, which starts with a goal-setting stage where the coach discusses creating a S.M.A.R.T goal with the patient, namely a goal that is specific, measurable, achievable, relevant, and time-bound (Doran, 1981). Once the goal is settled, the coach agent would follow up on the patient's progress and maintain patient engagement. In addition, we focus on developing the system to understand and address the patient's emotions and respond empathetically, which is crucial for better procedure outcomes in healthcare settings (Levinson et al., 2000; Moudatsou et al., 2020). A health coaching dialogue example is shown in Figure 1.

Building such a dialogue system is challenging: First, the real-world health coaching dataset is limited in size and annotation, which restricts not only end-to-end but also modularized approaches, especially with no dialogue states annotated. Additional annotation is also resource-intensive. Second, generating empathetic responses is subtle, which may require incorporating external empathetic knowledge, since cases of empathetic responses in our health coaching dataset are rare.

To address these challenges, we propose a modularized task-oriented health coaching dialogue system with a simplified architecture that requires fewer annotations. The system contains an NLU module, an NLG_{hc} module, and an NLG_{emp} module. The NLU module aims to keep track of the goal attributes as simplified belief states. The NLG_{hc} module takes as input the current dialogue context, the belief states, and the coaching stage to generate coaching utterances. Finally, for the NLG_{emp} module, we build an emotion cue detector and mechanism-conditioned empathy generator to facilitate empathetic response generation.

To evaluate our approach, we combine automatic evaluation with expert-based human evaluation. Our experimental results show that our NLU module outperforms the state-of-the-art (Gupta et al., 2020a,b, 2021) by > 10% in F1-score in the slot-filling and > 7% in semantic frame correctness in the offline goal attributes tracking task, which also enables updating goal information online at every dialogue turn. Moreover, the experiments demonstrate that our dialogue generation achieved best performance compared to baseline methods in terms of coherence, fluency, and empathy. Finally, through a pilot human evaluation, our model's generation is preferred by the health coaches as concerns coherence and empathy.

The contributions of this work are: (1) We propose to build an efficient modularized health coaching dialogue system that helps patients create and accomplish specific goals, with a simplified NLU and NLG framework combined with mechanism-conditioned empathetic response generation. (2) Our system outperforms the state-of-the-art in NLU tasks while significantly reducing the annotation workload. (3) Through automatic and human evaluation, we show our system generates more coherent and empathetic responses, which can provide suggestions for health coaches and improve coaching efficiency.

2 Related Work

- **Conversational Agents in Healthcare.** Conversational agents have been explored to improve the efficacy and scalability of the interactions between healthcare professionals and patients. For instance, chatbots in different healthcare settings, including chronic disease monitoring (Chaix et al., 2020), cognitive behavior therapy (Fitzpatrick et al., 2017), and physical activity promotion (Mohan et al., 2020; Kocielnik et al., 2018). However, they are limited in natural language understanding and generation capabilities. More sophisticated approaches have been proposed in mental health counseling (Althoff et al., 2016; Shen et al., 2020). In our previous work Gupta et al. (2020a,b, 2021), we collected real-world health coaching conversation datasets and focused on the NLU components of the dialogue which summarize weekly goals to support health coaches.

- **Task-Oriented Dialogue.** Traditional task-oriented dialogue systems are modularized (Jokinen and McTear, 2009). They consist of an NLU component for understanding the user intent and recording the dialogue states and an NLG component for policy management and response generation (Williams et al., 2016; Budzianowski et al., 2018; Mrkšić et al., 2017; Wen et al., 2015). However, approaches

have shifted towards end-to-end architectures to reduce human effort and error propagation between modules (Bordes et al., 2017; Wen et al., 2017). Recently, training an end-to-end system as a sequence prediction problem leveraging the causal language models has delivered promising results (Hosseini-Asl et al., 2020; Peng et al., 2021).

- **Empathetic Data for Conversations.** Empathetic interaction is a key to better task outcomes in conversations. Recently, empathetic data and approaches have been proposed to facilitate empathetic conversations in open-domain and healthcare settings. Babytalk (Hunter et al., 2008; Mahamood and Reiter, 2011) is an earlier system that summarizes neonatal intensive care patient data for different types of users, and provides affective cues for the parents. Rashkin et al. (2019) proposed an open-domain empathetic dialogue dataset (ED), with each dialogue grounded in an emotional context. Welivita et al. (2021) extended from ED and proposed a large-scale silver-standard empathetic dialogue data based on movie scripts. Sharma et al. (2020, 2021a) proposed an empathetic dataset in mental health support settings, with the communication mechanisms and corresponding strength of empathy annotated.

3 Health Coaching Dataset

We first briefly describe our dataset, that motivates the approach that we are proposing; full details can be found in Gupta et al. (2020a). We collected two datasets of health coaching dialogues between patients and coaches via text messages. Dataset 1 and 2 contain 28 and 30 patients, with each patient coached for four and eight weeks, respectively. Each week the health coach converses with the patient to create a physical activity S.M.A.R.T. goal and then follows up on the patient's progress. The two datasets contain 336 weeks of dialogues with 21.4 average turns per week.

We defined ten slots for the goal's attributes (types of activity, amount, time, days, location, duration, and the confidence score for the activity) (the Appendix contains examples of all slots). We used a stage-phase schema for additional turn-level annotation describing how the health coaching dialogue unfolds. We defined two stages: the goal-setting stage and the goal-implementation stage. Each stage includes a set of phases, such as goal identification, negotiation, and follow-up. Each turn can belong to a certain stage-phase combination. Later, we added dialogue act annotations (Gupta et al., 2021), consisting of 12 domain-independent dialogue acts, following the ISO-standard by Mezza et al. (2018).

Since dataset 1 was collected before dataset 2, the manual annotation was developed on dataset 1, and used to develop our NLU component, that was then tested on dataset 2. Hence, dataset 1 is fully annotated for slot-value spans, goals, stages and phases. In dataset 2, only three patients are annotated for slot-value spans and phases, and 15 weeks for goals; the whole dataset is annotated for stages. As far as dialogue acts are concerned, they are only available for 15 weeks of dialogues in dataset 1[1].

4 Methods

In this section, we begin by first providing a brief workflow of our proposed health coaching dialogue system and then describing the model architecture in details.

4.1 Health Coaching Dialogue System

A health coaching dialogue can be framed into stages: **(1)** Starting with the goal-setting stage, the coach helps the patient create a specific goal whose attributes can be represented by a list of slot values[2] (*e.g., Activity = Walk; Amount = 3000 steps; Days = Mon-Fri*), which retains a task-oriented nature. However, it is much more complex than the prevailing task-oriented services, since **(2)** after the goal is initialized, the coach needs to follow up on the patient's progress and maintain patient engagement (e.g., checking in, sending reminders, revising the goal, and providing encouragement on patient): this is the goal implementation stage. A health coaching dialogue contains multiple turns. We denote the stage that the turn t belongs to as S_t.

Leveraging the task-oriented dialogue framework, we use belief state B_t, a list of slot-value pairs, to record the goal attributes in a turn t. An **NLU** module is used to infer B_t by considering the earlier patient utterances $U_{<t}$, current patient utterances U_t, and earlier system responses $R_{<t}$ as

[1]The datasets with annotations are available at https://github.com/uic-nlp-lab/virtualcoachdata.

[2]We provide a complete list of slots and corresponding examples in the Appendix.

input to the module. Formally:

$$B_t = \text{NLU}([U_0, R_1, U_1, R_2, ..., U_t]) \quad (1)$$

B_t summarizes the goal from the dialogue history and will be used for conditional response generation and lexicalizing the generated response.

Then, given the dialogue context C_t in turn t, consisting of the previous two turns [R_{t-1},U_{t-1}], the stage of the previous turn S_{t-1}, and B_t, we build a sequence classifier to predict the stage S_t:

$$S_t = \text{Seq2Label}_{hc}([C_t, B_t, S_{t-1}]) \quad (2)$$

Finally, a delexicalized response R_t is generated via a Seq2Seq neural network given C_t, B_t, and S_t concatenated as a single sequence:

$$R_t = \text{Seq2Seq}_{hc}([C_t, B_t, S_t]) \quad (3)$$

The response can be lexicalized into human readable text using the belief state B_t. Equations 2 and 3 constitute the **NLG$_{\text{hc}}$** module, where we explore replacing the fine-grained action prediction with the coarse stage prediction as proximal dialogue management.

Although **NLG$_{\text{hc}}$** could learn limited empathetic response patterns such as *"sorry to hear that."* from the health coaching data, it does not retain prior empathetic knowledge and explicitly model empathy. However, showing a caring attitude and being empathetic to patients' emotional cues are crucial to patient activation and engagement, leading to better task outcomes. To facilitate the empathetic capability of the system, we build the **NLG$_{\text{emp}}$** module, where the empathetic response $\tilde{R}_t$ is generated conditioned on the patient's previous turn utterance U_{t-1} and communication mechanism signals M:

$$\tilde{R}_t = \text{Seq2Seq}_{emp}([U_{t-1}, M]) \quad (4)$$

We follow Sharma et al. (2020) and consider three communication mechanisms for empathy: *'Emotional Reactions'*, *'Interpretations'*, and *'Explorations'*. Emotional reaction expresses direct emotions (e.g., compassion) to show empathy, such as *"I would be very worried."* Interpretation communicates an understanding of the speaker's experience, such as *"I know anxiety is scary."* Exploration expresses empathy by exploring the speaker's feelings, such as *"What happened? How come?"* An empathetic response can be realized through multiple communication mechanisms. Thus, M contains one or more of the three special tokens `[EMOR]`, `[INTERP]`, and `[EXPLOR]`, representing the three communication mechanisms. We seek to use the mechanism signals to control the style of the generated empathetic response, making it flexible and appropriate for health coaching scenarios.

In addition, an emotion cue detector is built to support empathetic generation. We predict the current emotion signal E_t given the patient's previous utterance U_{t-1}. Then $\tilde{R}_t$ is generated when a strong emotion cue is detected with the probability of certain types of emotion greater than a predefined threshold, *i.e.*, $p(E_t|U_{t-1}) > \tau$, obtained from development set performance.

The overall framework of our model is illustrated in Figure 2.

4.2 Model Architectures

In this section, we describe the detailed model architecture for each of the NLU, NLG$_{\text{hc}}$, and NLG$_{\text{emp}}$ module.

4.2.1 The NLU Module

The NLU module records the current state of the slot values. It consists of a neural slot-filling model that extracts the slot fillers from each utterance and a carryover classifier to determine if the value of a slot should be copied from the previous state.

Neural Slot-Filling A neural slot-filling model maps the sentence representation to a sequence of BIO labels (the beginning *(B)* and inside *(I)* of each slot label, and outside *(O)* for others), often combined with sentence-level classifications (e.g., domain, intent). Following Chen et al. (2019), we use BERT (Devlin et al., 2019) as the model backbone, the `[CLS]` token of the sentence for sentence-level classification, and the final hidden state of the first sub-token of each word at position n for BIO labeling. The network is trained by maximizing the conditional probability:

$$p\left(y^c, \boldsymbol{y^s} \mid \boldsymbol{x}\right) = p\left(y^c \mid \boldsymbol{x}\right) \prod_{n=1}^{N} p\left(y_n^s \mid \boldsymbol{x}\right) \quad (5)$$

Where y^c, y_n^c are softmax probabilities for sentence labels and BIO labels of word n, with $\boldsymbol{x}$ being the sequence of word tokens.

Carryover Classifier The carryover classifier determines if the value of a slot should be copied from the previous state or updated with the new instance seen in the current utterance, which outputs a binary value for each slot.

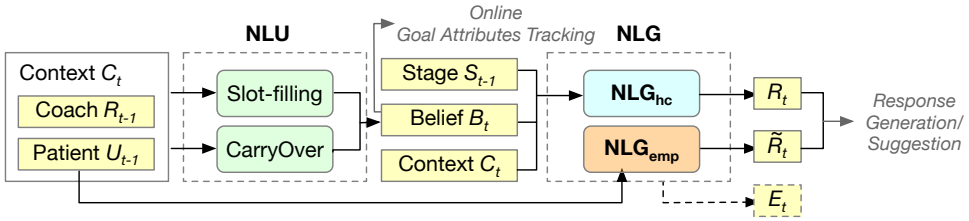

Figure 2: The framework of our health coaching dialogue system. The NLU module consisting of the slot-filling and carryover model reads the dialogue and infers belief state B_t. The NLG_{hc} module takes as input the stage S_{t-1}, belief state B_t, and context C_t to generate response R_t. The NLG_{emp} handles the cases where empathy are required and outputs empathetic response $\tilde{R}_t$ and emotion signal E_t.

Following the work of Gao et al. (2019), we developed our slot carryover classifier based on the current dialogue context. Our proposed carryover classifier, however, differs from Gao et al. (2019), in terms of: (1) we design the classifier as a separate model rather than a component in an end-to-end belief state tracking architecture; (2) it benefits from the results of the slot-filling model and only makes a prediction when there is a collision between the existing value and the new for a given slot. Such a design takes into account that: (1) dialogue datasets in healthcare are limited in size, making end-to-end training difficult; (2) these datasets can also be limited in annotations. Our model can facilitate annotation since the annotators can work with the slot-filling model to only examine the lines where the value collision occurs, enabling more efficient labeling for belief states in a carryover fashion.

We use BERT to encode the contexts and utilize the hidden state associated with the `[CLS]` token to represent the current context C_t and maximize $p(y_t|C_t)$ via fine-tuning BERT with cross entropy minimization:

$$h_t = \text{BERT}(C_t)[\text{CLS}]$$
$$y_t = \text{Softmax}(Wh_t + b)$$

where $y_t \in [0, 1]^{N_s}$, N_s is the number of slots.

During inference, the slot-filling model extracts the slot fillers at each turn. The carryover classifier determines if the value should be copied from the previous state or updated with the new instance given dialogue context when there is a value conflict. NLU enables an update for the belief state B_t at each turn, *i.e.,* providing online goal attributes tracking.

4.2.2 The NLG_{hc} Module

The dominant modularized approach for task-oriented dialogue often consists of belief state tracking, dialogue policy, and language generation. It requires dedicated modeling and annotations for dialogue acts. However, annotating acts is resource-intensive in healthcare settings.

An alternative is using stages, where each stage restricts a set of possible actions in health coaching dialogue and other healthcare conversations, such as clinical motivational interviews and patient encounters. For example, in health coaching, sending reminders and encouragement on progress can only appear in the goal implementation stage when the goal has already been created. Another example is the SOAP (Subjective, Objective, Assessment, and Plan) structure (Podder et al., 2021) of patient encounters, in which discussing a patient's chief complaint occurs in the Subjective part, while diagnosis is discussed in the Assessment part.

Thus, we explore the possibility of using stages that contain coarse and fuzzy act information to guide dialogue generation instead of fine-grained act annotations. Concretely, we use T5 (Raffel et al., 2020) to jointly model Seq2Label_{hc} (*cf.*2) and Seq2Seq_{hc} (*cf.*3) for stage S_t prediction and response R_t generation as a multi-task approach. The input is a single sequence by concatenating the local dialogue context, current belief state, and stage tokens. For example, the input for Seq2Seq_{hc} is the concatenation of C_t, B_t, and S_t, separated by delimiter tokens, which is mapped to the target response sequence, as shown in Figure 3. The model is encoder-decoder based, trained with a maximum likelihood objective with a different prefix prepending to the input corresponding to each task.

4.2.3 The NLG_{emp} Module

Empathetic Response Generation We fine-tune GPT2 (Radford et al., 2019) for empathetic response generation. At training time, we concatenate the user utterance, the corresponding empathetic response, and the communication mecha-

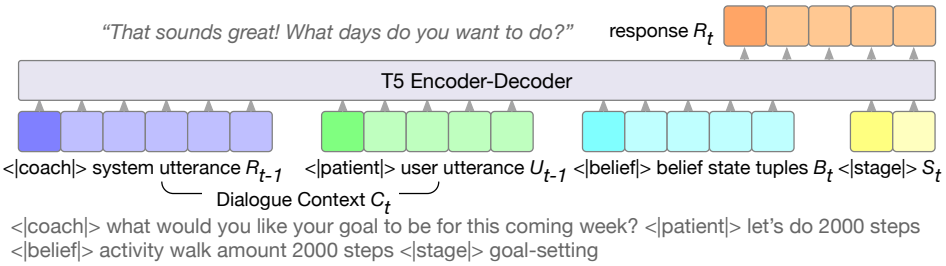

Figure 3: Model architecture of generating R_t in NLG_{hc}. The context C_t, belief tokens B_t, and the predicted stage S_t are concatenated as a single input sequence, training with a T5 encoder-decoder model.

nisms used in the response, separated by delimiter tokens, as one single training sequence $\boldsymbol{x}$. We want GPT2 to learn to model the joint probability $p(\boldsymbol{x})$ by fine-tuning on empathetic data. During inference time, the model takes as input the user utterance and the communication mechanisms and generates one token at a time for empathetic response.

The following shows an example of such a training sequence:

> <|bos|> [EMOR] I was so exhausted yesterday. <|sep|> That's understandable. Take some rest! <|eos|>

where <|bos|> and <|eos|> are the delimiters for beginning of sequence and end of sequence, [EMOR] stands for the communication mechanism *Emotional Reaction*, and <|sep|> separates the user utterance and empathetic response . During inference, the model starts to generate tokens after <|sep|>.

Emotional Cue Detection To support empathetic response, we build a BERT-based multi-class classifier to predict the emotion signals of the patient's utterance. In this work, we use the softmax probability for the predicted label to determine if the empathetic response needs to be generated. However, the predicted emotion labels also facilitate future sentiment analysis work in health coaching dialogue.

5 Experiments

This section includes datasets description, evaluation metrics, experimental results, and qualitative analysis.

5.1 Dataset Details

We conducted our experiments on the following benchmark datasets:

Health Coaching Datasets These are the datasets we described earlier in Section 3. Following Gupta et al. (2020b), we use dataset 1 for training/development and dataset 2 for testing for both NLU and NLG_{hc} to ensure comparability.

Data Augmentation for Slot-filling We only consider the utterances that contain at least one slot-value span for slot-filling. This results in 955:105:205 data points for training, development, and testing. To alleviate data scarcity, we use a naïve data augmentation method to synthesize training examples that (1) first randomly replaces the value with one possible alternative for each slot; (2) then rephrases the modified sentence with a pre-trained paraphrase model[3].

Empathetic Datasets We utilized two benchmark empathetic dialogue datasets: *(i)* EMPATHETICDIALOGUES (ED) (Rashkin et al., 2019) and *(ii)* EPITOME. EMPATHETICDIALOGUES (ED) is a open-domain empathetic dialogue dataset containing 24,850 dialogues, with each dialogue grounded in one of 32 emotional context (e.g., proud, apprehensive, confident, guilty). EPITOME (Sharma et al., 2020) is an empathetic dataset in mental health support settings. The dataset consists of 3k posts annotated with respect to the communication mechanisms (*Emotional Reaction, Exploration, Interpretation*) and level of empathy (from 0 to 2, representing no empathy, weak and strong empathy, respectively) in each mechanism.

To facilitate empathetic generation conditioned on the communication mechanisms, we first build a weak multi-label classifier for predicting the types of communication mechanisms based on the 3k posts in Sharma et al. (2020) (achieving a hamming loss (Tsoumakas and Katakis, 2007) of ∼0.16). Then we apply the classifier on the ED dataset

[3] A fine-tuned PEGASUS (Zhang et al., 2020) on PAWS (Zhang et al., 2019)

to get communication mechanism labels for each instance. We use their default train/dev/test split. Finally, $N = 64$ empathetic cases in the health coaching dataset are used for few-shot fine-tuning the trained empathetic generation model.

5.2 Evaluation Metrics

Depending upon the task, we use the following evaluation metrics:

1. Slot-filling: Precision, Recall, and F1-score.
2. Dialogue State (Goal Attribute) Tracking: partial/complete match and goal correctness@k following Gupta et al. (2020b):

   Correctness@k: Computes the percentage of correctness over all the predicted goals of each week. If at least k attributes are predicted correctly, the goal is regarded as correct.
3. Dialogue Generation: BLEU (Papineni et al., 2002), BertScore (Zhang et al., 2020), Perplexity (PPL), and Empathy Score.

   Perplexity: We measure fluency as perplexity (PPL) of the generated response using a pre-trained GPT2 model that has not been fine-tuned for this task, following previous work (Ma et al., 2020; Sharma et al., 2021b).

   Empathy Score: We train a standard text regression model based on BERT using the response posts and corresponding level of empathy scores in Sharma et al. (2020) (achieving an RMSE of $\sim$0.57). We use this model to measure the empathy in the generated outputs.

For detailed descriptions of metrics, training, including model parameters, selection, and supplementary analysis, please see the Appendix.

System	Slot R	Slot P	Slot F1	PAcc
Gupta et al. (2020b)	0.806	0.808	0.790	0.801
+Phase	0.899	0.837	0.867	0.779
+StartPhase	0.910	0.847	0.877	**0.835**
+StartPhase+Aug	**0.926**	**0.879**	**0.902**	0.817
Slot Only+Aug	0.904	0.876	0.890	-

Table 1: Evaluation on slot-filling with ablations. *Aug*: using data augmentation; *StartPhase*: Jointly predicting the phase of the sentence only if it is the beginning sentence of the phase.

5.3 Results

5.3.1 Goal Attributes Tracking

Table 1 shows the performance of slot-filling and phase prediction compared to the previous model. Jointly predicting the phase of the sentence only if it is at the beginning of the phase, combined with data augmentation (+StartPhase+Aug), achieved the best performance for slot-filling, outperforming the state-of-the-art by 11.2% in F1. However, modeling without phases suffices for slot-filling while reducing the annotation cost. As such, we adopt the no-joint model for downstream tasks. Our experimental result also shows the carryover classifier achieved a F1-score of 0.88 using only the dialogue context. We investigate whether dialogue act and phase labels can improve carryover classifier, however they barely contribute to the model performance[4].

In previous work, we extracted goals at two critical points for each week to evaluate offline goal tracking: one at the end of the goal-setting stage (forward) and the other at the end of the goal-implementation stage (backward). The forward and backward goals can be different since the patient may encounter barriers, and the goal can be revised in the implementation stage. We also proposed a rule-based approach to update the slot values, which simply records the last mention of the value for each slot except for certain conditions. In this paper, we compared our model with previous work and a combination of our slot-filling model with the previous rule-based approach (our SF+Rule). Table 2 shows the performance for goal attributes tracking of our NLU module compared with previous work. For dataset 1 backward goals, our SF+Rule achieved the best performance resulting from more accurate slot-filling. We observe that in dataset 1, the coach tends to summarize the goal to the patients at the end of each week, which benefits the rule-based approach. Nonetheless, our NLU module outperforms previous work in all evaluation metrics on dataset 1 forward and dataset 2; it also enables goal tracking at each turn (online).

5.3.2 Dialogue Generation

Since our previous work did not include a generation model, we compare our dialogue generation with three baselines: (1) Retrieval-based model. We use BERT to encode the current belief state and dialogue context as query h_q, and encode the

[4]See Appendix for detailed results.

	Dataset 1 Backward					Dataset 1 Forward					Dataset 2				
	Complete	Partial	@k=10	@k=9	@k=8	Complete	Partial	@k=10	@k=9	@k=8	Complete	Partial	@k=10	@k=9	@k=8
Gupta et al. (2020b)	0.77	0.81	13.1	39.3	**65.4**	0.81	0.85	15.0	44.9	68.2	-	-	-	-	-
ourSF+Rule	**0.808**	**0.858**	**31.6**	**42.1**	63.2	0.831	0.863	21.7	69.6	73.9	0.767	0.787	13.6	18.2	40.9
ourNLU	0.792	0.825	15.8	36.8	63.2	**0.838**	**0.869**	**21.7**	**69.6**	**78.3**	**0.8**	**0.813**	**13.6**	**27.3**	**63.6**

Table 2: Evaluation on offline goal attributes tracking with complete/partial match and goal correctness@k.

Input	**Model**	**Generated Response**
Coach: What would you like your goal to be this week? Patient: I want to walk 30 min a day between 6am to 8am.	Ours	Sounds good, which days were in your mind?
	IR	And what days would you like to walk the track around neighborhood?
	Seq2Seq	How many days do you want to reach this week?
Coach: Good morning! How is your goal for this week going? Patient: I'm sorry I didn't go to work today I have a massive migraine headache.	Ours	[EMOR] Oh no, I hope you are okay. [EXPLOR] Oh geez, sorry to hear that. Are you feeling better? [INTERP] I've had this experience before. Sometimes it really hits you.
	IR	Check your steps.
	Seq2Seq	Have you been busy with the migraine this week and how did it turn out?
Coach: Don't forget to take a deep breath - breathing is important. Patient: You made me smile. Thanks.	Ours	Don't forget to take a deep breath - breathing is important.
	✗ Repeat	
Coach: Hi when you get a chance can you sync your fitbit? I only see your steps since last Friday. Thanks. Patient: Sorry been in Texas, just made it back to Chicago.	Ours	That's great news. I hope you don't get into a long trip there.
	✗ Focus	

Table 3: Qualitative examples of generated responses in empathetic and non-empathetic scenarios, combined with error analysis.

Model	BLEU (↑)	PPL (↓)	BertS F1 (↑)	EmpS (↑)
IR	0.194	24.2	0.853	-0.179
Seq2Seq	0.242	16.26	0.863	+0.073
+Acts	0.235	16.2	0.861	-0.065
OURS	**0.251**	**15.6**	**0.872**	**+0.256**

Table 4: Evaluation on dialogue generation with automatic metrics. BLEU: Average of BLEU-1,-2,-3,-4. EmpS: Computed by the difference of the empathetic score between the output and the ground truth ($\sim$ 0.978).

response as h_r. We fine tune the BERT models to retrieve the response that maximizes $h_q \cdot h_r$; (2) A seq2seq neural network that maps the belief state and dialogue context to the output without stages. (3) using dialogue acts instead of stages.

Table 4 shows that our response generation outperforms all baselines in all metrics. The predicted stage information (achieved an accuracy of $\sim$0.92) provides meaningful signals for dialogue generation. In addition, by incorporating external empathetic knowledge, our model achieved +0.256 average improvement in empathy. Including dialogue acts does not improve the performance compared to the seq2seq model. This is not unexpected because labels are imbalanced and only available for 15 weeks of data. Computationally labeling dialogue acts also leads to large error propagation. The emotional cue detection (32-category classification) achieved an accuracy of $\sim$0.58. Based on development set observation, we set τ to be 0.7 and generate empathetic response when the cumulative probability of the top-2 predicted emotional labels is greater than τ.

Human Evaluation. We performed an expert-based human evaluation on coherence and empathy through A/B testing. We ask two health coaches to compare outputs from our model against other baselines given the same input and choose (a) the response which is more empathetic; (b) the response which is more coherent. Among the collected 43 examples, our model's outputs have a $\sim$71% preference for empathy and $\sim$55% preference for coherence over other baselines.

5.4 Qualitative Analysis

We present examples of our health coaching dialogue generation and mechanism-conditioned empathetic generation in Table 3. In the first case, where empathy is not required, all three models choose to inquiry for more information of the goal. However, the fixed, retrieval-based response can contain context-irrelevant tokens (*"..the track around neighborhood"*) and the Seq2Seq response is relatively unnatural. In the second case, given the cues including *"..a massive migraine headache.."*, our model can generate corresponding empathetic responses given different communication mechanisms (e.g., [EMOR] $\rightarrow$ *"Oh no, I hope you are okay."*). In contrast, the retrieval-based and Seq2Seq model failed to response empathetically. Particularly, the Seq2Seq model failed to distinguish a symptom (*i.e., "migraine headache"*) from a physical activity or goal. Finally, we present two incoherent generation examples by our model. In the first example, our system misinterprets the

coach's utterance - a *parody* of a reminder - as a real reminder, which tends to repeat again; thus, the system naïvely copies it. In the second example, the system needs to see more context to understand that the trip is an explanation from the patient for not making progress, which the coach asked about in the previous utterance. A better response should address the patient's explanation while maintaining specificity on the trip scenario, e.g., *"No problem. Welcome back."*

6 Conclusions and Future Work

We built an efficient health coaching dialogue system that helps patients create and accomplish specific goals, with a simplified NLU and NLG framework combined with mechanism-conditioned empathetic response generation. The experiments show that the system can generate more coherent and empathetic responses, supporting health coaches and improving coaching efficiency. In addition, our system outperforms the state-of-the-art in NLU tasks while requiring fewer annotations. We view our approach as a key step towards building automated and more accessible health coaching systems in low-resource settings and believe our approach may also generalize to building dialogue systems in similar scenarios, such as patient education at discharge or consulting on behavior change problems.

In the future, we will explore the following directions: (1) Modelling empathetic understanding and generation with the goal response generator as one integrated end-to-end system while providing explainability. (2) A more comprehensive human evaluation from both the coach's and the patient's perspectives, including but not subject to goal completion and activity engagement rate.

Acknowledgments

This work is supported by the National Science Foundation under Grant IIS-1838770.

References

Adrian Aguilera and Ricardo F Muñoz. 2011. Text messaging as an adjunct to cbt in low-income populations: A usability and feasibility pilot study. *Professional Psychology: Research and Practice*, 42(6):472.

Tim Althoff, Kevin Clark, and Jure Leskovec. 2016. Large-scale analysis of counseling conversations: An application of natural language processing to mental health. *Transactions of the Association for Computational Linguistics*, 4:463–476.

Antoine Bordes, Y-Lan Boureau, and Jason Weston. 2017. Learning end-to-end goal-oriented dialog. In *International Conference on Learning Representations*.

Paweł Budzianowski, Tsung-Hsien Wen, Bo-Hsiang Tseng, Iñigo Casanueva, Stefan Ultes, Osman Ramadan, and Milica Gašić. 2018. MultiWOZ - a large-scale multi-domain Wizard-of-Oz dataset for task-oriented dialogue modelling. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 5016–5026, Brussels, Belgium. Association for Computational Linguistics.

Susan Butterworth, Ariel Linden, Wende McClay, and Michael C Leo. 2006. Effect of motivational interviewing-based health coaching on employees' physical and mental health status. *Journal of occupational health psychology*, 11(4):358.

Benjamin Chaix, Arthur Guillemassé, Pierre Nectoux, Guillaume Delamon, Benoît Brouard, et al. 2020. Vik: a chatbot to support patients with chronic diseases. *Health*, 12(07):804.

Qian Chen, Zhu Zhuo, and Wen Wang. 2019. Bert for joint intent classification and slot filling. *arXiv preprint arXiv:1902.10909*. Only available as unpublished pre-preprints.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186. Association for Computational Linguistics.

Jingqing Zhang, Yao Zhao, Mohammad Saleh, and Peter Liu. 2020. Pegasus: Pre-training with extracted gap-sentences for abstractive summarization. In *International Conference on Machine Learning*, pages 11328–11339. PMLR.

Tianyi Zhang, Varsha Kishore, Felix Wu, Kilian Q. Weinberger, and Yoav Artzi. 2020. Bertscore: Evaluating text generation with bert. In *International Conference on Learning Representations*.

George T Doran. 1981. There's a SMART way to write management's goals and objectives. *Management review*, 70(11):35–36.

Kathleen Kara Fitzpatrick, Alison Darcy, and Molly Vierhile. 2017. Delivering cognitive behavior therapy to young adults with symptoms of depression and anxiety using a fully automated conversational agent (woebot): a randomized controlled trial. *JMIR mental health*, 4(2):e7785.

Shuyang Gao, Abhishek Sethi, Sanchit Agarwal, Tagyoung Chung, and Dilek Hakkani-Tur. 2019. Dialog state tracking: A neural reading comprehension approach. In *Proceedings of the 20th Annual SIGdial Meeting on Discourse and Dialogue*, pages 264–273, Stockholm, Sweden. Association for Computational Linguistics.

Amireh Ghorob. 2013. Supplement: Health coaching: Teaching patients to fish. *Family practice management*, 20(3):40–42.

Itika Gupta, Barbara Di Eugenio, Brian Ziebart, Aiswarya Baiju, Bing Liu, Ben Gerber, Lisa Sharp, Nadia Nabulsi, and Mary Smart. 2020a. Human-human health coaching via text messages: Corpus, annotation, and analysis. In *Proceedings of the 21th Annual Meeting of the Special Interest Group on Discourse and Dialogue*, pages 246–256, 1st virtual meeting. Association for Computational Linguistics.

Itika Gupta, Barbara Di Eugenio, Brian Ziebart, Bing Liu, Ben Gerber, and Lisa Sharp. 2020b. Goal summarization for human-human health coaching dialogues. In *The Thirty-Third International Flairs Conference*.

Itika Gupta, Barbara Di Eugenio, Brian D Ziebart, Bing Liu, Ben S Gerber, and Lisa K Sharp. 2021. Summarizing behavioral change goals from sms exchanges to support health coaches. In *Proceedings of the 22nd Annual Meeting of the Special Interest Group on Discourse and Dialogue*, pages 276–289.

Ehsan Hosseini-Asl, Bryan McCann, Chien-Sheng Wu, Semih Yavuz, and Richard Socher. 2020. A simple language model for task-oriented dialogue. *Advances in Neural Information Processing Systems*, 33:20179–20191.

Jim Hunter, Albert Gatt, Francois Portet, Ehud Baruch Reiter, and Gowri Somayajulu Sripada. 2008. Using natural language generation technology to improve information flows in intensive care units. In *18th European Conference on Artificial Intelligence (ECAI 2008)*, pages 678–682. IOS Press.

Kristiina Jokinen and Michael McTear. 2009. Spoken dialogue systems. *Synthesis Lectures on Human Language Technologies*, 2(1):1–151.

Shreya Kangovi, Frances K Barg, Tamala Carter, Kathryn Levy, Jeffrey Sellman, Judith A Long, and David Grande. 2014. Challenges faced by patients with low socioeconomic status during the post-hospital transition. *Journal of general internal medicine*, 29(2):283–289.

Kirsi Kivelä, Satu Elo, Helvi Kyngäs, and Maria Kääriäinen. 2014. The effects of health coaching on adult patients with chronic diseases: a systematic review. *Patient education and counseling*, 97(2):147–157.

Rafal Kocielnik, Lillian Xiao, Daniel Avrahami, and Gary Hsieh. 2018. Reflection companion: a conversational system for engaging users in reflection on physical activity. *Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies*, 2(2):1–26.

Wendy Levinson, Rita Gorawara-Bhat, and Jennifer Lamb. 2000. A study of patient clues and physician responses in primary care and surgical settings. *Jama*, 284(8):1021–1027.

Xinyao Ma, Maarten Sap, Hannah Rashkin, and Yejin Choi. 2020. PowerTransformer: Unsupervised controllable revision for biased language correction. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 7426–7441, Online. Association for Computational Linguistics.

Saad Mahamood and Ehud Reiter. 2011. Generating affective natural language for parents of neonatal infants. In *Proceedings of the 13th European Workshop on Natural Language Generation*, pages 12–21.

Stefano Mezza, Alessandra Cervone, Evgeny Stepanov, Giuliano Tortoreto, and Giuseppe Riccardi. 2018. ISO-standard domain-independent dialogue act tagging for conversational agents. In *Proceedings of the 27th International Conference on Computational Linguistics*, pages 3539–3551, Santa Fe, New Mexico, USA. Association for Computational Linguistics.

Shiwali Mohan, Anusha Venkatakrishnan, and Andrea L Hartzler. 2020. Designing an ai health coach and studying its utility in promoting regular aerobic exercise. *ACM Transactions on Interactive Intelligent Systems (TiiS)*, 10(2):1–30.

Maria Moudatsou, Areti Stavropoulou, Anastas Philalithis, and Sofia Koukouli. 2020. The role of empathy in health and social care professionals. In *Healthcare*, volume 8, page 26. Multidisciplinary Digital Publishing Institute.

Nikola Mrkšić, Diarmuid Ó Séaghdha, Tsung-Hsien Wen, Blaise Thomson, and Steve Young. 2017. Neural belief tracker: Data-driven dialogue state tracking. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1777–1788, Vancouver, Canada. Association for Computational Linguistics.

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. Bleu: a method for automatic evaluation of machine translation. In *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics*, pages 311–318, Philadelphia, Pennsylvania, USA. Association for Computational Linguistics.

Baolin Peng, Chunyuan Li, Jinchao Li, Shahin Shayandeh, Lars Liden, and Jianfeng Gao. 2021. Soloist: Building task bots at scale with transfer learning and machine teaching. *Transactions of the Association for Computational Linguistics*, 9:807–824.

Vivek Podder, Valerie Lew, and Sassan Ghassemzadeh. 2021. Soap notes. In *StatPearls [Internet]*. StatPearls Publishing.

Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, Ilya Sutskever, et al. 2019. Language models are unsupervised multitask learners. *OpenAI blog*, 1(8):9.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J. Liu. 2020. Exploring the limits of transfer learning with a unified text-to-text transformer. *Journal of Machine Learning Research*, 21(140):1–67.

Hannah Rashkin, Eric Michael Smith, Margaret Li, and Y-Lan Boureau. 2019. Towards empathetic open-domain conversation models: A new benchmark and dataset. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 5370–5381, Florence, Italy. Association for Computational Linguistics.

Ashish Sharma, Inna W. Lin, Adam S. Miner, David C. Atkins, and Tim Althoff. 2021a. Towards facilitating empathic conversations in online mental health support: A reinforcement learning approach. In *Proceedings of the Web Conference 2021*, WWW '21, page 194–205, New York, NY, USA. Association for Computing Machinery.

Ashish Sharma, Inna W. Lin, Adam S. Miner, David C. Atkins, and Tim Althoff. 2021b. *Towards Facilitating Empathic Conversations in Online Mental Health Support: A Reinforcement Learning Approach*, page 194–205. Association for Computing Machinery, New York, NY, USA.

Ashish Sharma, Adam Miner, David Atkins, and Tim Althoff. 2020. A computational approach to understanding empathy expressed in text-based mental health support. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 5263–5276, Online. Association for Computational Linguistics.

Siqi Shen, Charles Welch, Rada Mihalcea, and Verónica Pérez-Rosas. 2020. Counseling-style reflection generation using generative pretrained transformers with augmented context. In *Proceedings of the 21th Annual Meeting of the Special Interest Group on Discourse and Dialogue*, pages 10–20, 1st virtual meeting. Association for Computational Linguistics.

Rosemary Thackeray, Ray M Merrill, and Brad L Neiger. 2004. Disparities in diabetes management practice between racial and ethnic groups in the united states. *The Diabetes Educator*, 30(4):665–675.

David H Thom, Jessica Wolf, Heather Gardner, Denise DeVore, Michael Lin, Andy Ma, Ana Ibarra-Castro, and George Saba. 2016. A qualitative study of how health coaches support patients in making health-related decisions and behavioral changes. *The Annals of Family Medicine*, 14(6):509–516.

Grigorios Tsoumakas and Ioannis Katakis. 2007. Multi-label classification: An overview. *International Journal of Data Warehousing and Mining (IJDWM)*, 3(3):1–13.

Anuradha Welivita, Yubo Xie, and Pearl Pu. 2021. A large-scale dataset for empathetic response generation. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 1251–1264, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Tsung-Hsien Wen, Milica Gašić, Nikola Mrkšić, Pei-Hao Su, David Vandyke, and Steve Young. 2015. Semantically conditioned LSTM-based natural language generation for spoken dialogue systems. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 1711–1721, Lisbon, Portugal. Association for Computational Linguistics.

Tsung-Hsien Wen, David Vandyke, Nikola Mrkšić, Milica Gašić, Lina M. Rojas-Barahona, Pei-Hao Su, Stefan Ultes, and Steve Young. 2017. A network-based end-to-end trainable task-oriented dialogue system. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 1, Long Papers*, pages 438–449, Valencia, Spain. Association for Computational Linguistics.

Jason D Williams, Antoine Raux, and Matthew Henderson. 2016. The dialog state tracking challenge series: A review. *Dialogue & Discourse*, 7(3):4–33.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Remi Louf, Morgan Funtowicz, Joe Davison, Sam Shleifer, Patrick von Platen, Clara Ma, Yacine Jernite, Julien Plu, Canwen Xu, Teven Le Scao, Sylvain Gugger, Mariama Drame, Quentin Lhoest, and Alexander Rush. 2020. Transformers: State-of-the-art natural language processing. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 38–45, Online. Association for Computational Linguistics.

Yuan Zhang, Jason Baldridge, and Luheng He. 2019. PAWS: Paraphrase Adversaries from Word Scrambling. In *Proc. of NAACL*.

A Training Details

All the following models use Huggingface Transformers Library (Wolf et al., 2020). The hyperparameters are not extensively fine-tuned.

- **Slot-filling.** We use BERT-base as model backbone and associated tokenizer, with max sequence length of the tokenized input set to 50. The model was trained for {**5.0**, 7.0, 10.0}

epochs by Adam with a learning rate of {2e-5, **5e-5**}, batch size of {**32**, 64}.

- **Carryover Classifier.** We use BERT-base as model backbone and associated tokenizer, with max sequence length of the tokenized input set to 96. The model was trained for {5.0, **7.0**, 10.0} epochs by Adam with a learning rate of {2e-5, **5e-5**}, batch size of {**16**, 32, 64}.

- **NLG$_{\text{hc}}$** We use T5-base as model backbone and associated tokenizer, with max sequence length set to 128. The model was trained for 10.0 epochs by AdamW with a learning rate of 1e-4, warm up steps = 400, batch size of 64. We use sampling during decoding with top-k set to 50, top-p set to 0.95.

- **Empathetic Generation** We use GPT2 as model backbone and associated tokenizer, with max sequence length set to 96. The model was first trained for 10.0 epochs on the ED dataset by Adam with a learning rate of 1e-4, warm up steps = 400, batch size of 32. We use sampling during decoding with top-k set to 50, top-p set to 0.95. Then, the model was fine-tuned on 64 examples of health coaching empathetic data by 1 epoch. When decoding at inference, we use sampling with top-k set to 50, top-p set to 0.95.

- **Emotion Detection** We use BERT-base as model backbone and associated tokenizer, with max sequence length set to 96. The model was trained for 8.0 epochs by Adam with a learning rate of 4e-5, and batch size of 32.

B Health Coaching Dataset Slot Examples

Table 5 shows the description of the ten slots with value examples.

C Evaluation Metrics Description

Partial/Complete Match If all the values are correctly recorded for a given slot, it is considered a complete match. If the values are partially correct, it is considered a partial match.

Goal Correctness@k Computes the percentage of correctness over all the predicted goals of each week. A goal contains ten attributes (slot-values); if at least k attributes are predicted correctly, the goal is regarded as correct. Goal Correctness@k is trivially equal to 100% when $k = 0$.

BLEU BLEU score (Papineni et al., 2002) measures the word-level overlap between the generated output and the gold reference response.

BertScore BertScore (Tianyi Zhang et al., 2020) measures the semantic similarity between the generated output and the reference leveraging BERT contextual embeddings.

Fluency We measure fluency as perplexity (PPL) of the generated response using a pre-trained GPT2 model that has not been fine-tuned for this task, following previous work (Ma et al., 2020; Sharma et al., 2021b).

Empathy We train a standard text regression model based on BERT using the response posts and corresponding level of empathy scores in Sharma et al. (2020) (achieving an RMSE of $\sim$0.57). We use this model to evaluate the empathy in the generated output compared with baselines.

D Supplementary Analysis

D.1 Carryover Ablation

Table 7 shows the model performance of the carryover classifier with ablations. Using a combination of phases and acts can slightly improve recall with a tradeoff of reducing precision. However, using the dialogue context alone suffices for carryover classification with less annotation cost.

D.2 Emotion Cue Detection

Table 6 shows the predicted emotion labels with corresponding probabilities by emotion cue detection on some patient's utterance examples. The model can reasonably detect emotion signals of the patient's utterance only trained on the ED dataset.

Slot	Description	Value Examples
[activity]	The type of activity the patient will perform.	*"walk, jogging, stair climbing"*
[amount]	The quantity of activity.	*"2000 steps, 6 fligths"*
[duration]	The duration of the activity.	*"20 min, half an hour"*
[distance]	The distance of the activity.	*"3 blocks, 2 miles, from home to bus stop"*
[time]	The time of the day for the activity.	*"at noon, after lunch, 4 pm"*
[location]	The location of the activity.	*"at work, at home, around the park"*
[dayname]	The days to do the activity.	*"Monday, Tuesday"*
[daynumber]	The number of days for the activity.	*"3 days, 5 days"*
[repeatation]	The frequency of activity.	*"twice a day, daily"*
[score]	The confidence or attainability score.	Range from [1,10]

Table 5: The slot-value schema used in Gupta et al. (2020a).

Example Utterances	Top-2 Predicted Emotion Labels
Sorry I left my fitbit in the emergency room yesterday.	Guilty (0.506), Ashamed (0.323)
I'm not feeling very well yesterday so I did not go out for a walk.	Disappointed (0.305), Ashamed (0.174)
I reached 10k steps last week can you believe that?	Surprised (0.548), Proud (0.229)
Ok.	Angry (0.127), Furious (0.059)
I want to walk 3000 steps today.	Hopeful (0.484), Confident (0.132)

Table 6: The predicted emotion labels with corresponding probabilities by emotion cue detection on the patient's utterances.

Input	P	R	F1	Acc
Context Only	**0.91**	0.86	**0.89**	0.88
Context+Phase	0.88	0.87	0.88	0.87
Context+Act	0.88	0.86	0.87	0.87
Context+Phase+Act	0.90	**0.87**	**0.89**	**0.89**

Table 7: Model performance of carryover classification with ablations.

Generalized Intent Discovery: Learning from Open World Dialogue System

Yutao Mou[1*], Keqing He[2*], Yanan Wu[1], Pei Wang[1], Jingang Wang[2]
Wei Wu[2], Yi Huang[3], Junlan Feng[3], Weiran Xu[1*]
[1]Beijing University of Posts and Telecommunications, Beijing, China
[2]Meituan Group, Beijing, China
[3]China Mobile Research Institute, Beijing, China
{myt,yanan.wu,wangpei,xuweiran}@bupt.edu.cn
{hekeqing,wangjingang,wuwei}@meituan.com
{huangyi,fengjunlan}@chinamobile.com

Abstract

Traditional intent classification models are based on a pre-defined intent set and only recognize limited in-domain (IND) intent classes. But users may input out-of-domain (OOD) queries in a practical dialogue system. Such OOD queries can provide directions for future improvement. In this paper, we define a new task, Generalized Intent Discovery (GID), which aims to extend an IND intent classifier to an open-world intent set including IND and OOD intents. We hope to simultaneously classify a set of labeled IND intent classes while discovering and recognizing new unlabeled OOD types incrementally. We construct three public datasets for different application scenarios and propose two kinds of frameworks, pipeline-based and end-to-end for future work. Further, We conduct exhaustive experiments and qualitative analysis to comprehend key challenges and provide new guidance for future GID research. [1]

1 Introduction

Intent classification (IC) in a dialogue system aims to identify the goal of a user query, such as *BookFlight* or *AddToPlaylist*. Recent neural-based models (Liu and Lane, 2016; Goo et al., 2018; E et al., 2019; Chen et al., 2019; He et al., 2020) have achieved satisfying performance under the availability of large-scale labeled data. However, these methods face the challenge of data scarcity and poor scalability. They rely on a pre-defined intent set and supervised labels, which is limitted in some practical scenarios.

Existing intent classification models have little to offer in an open-world setting, in which many new intent categories are not defined apriori and no labeled data is available. These models rely on the pre-defined intent set, making it only recognize limited in-domain (IND) intent categories. But plenty of input queries may be outside of the fixed intent set, which we call Out-of-Domain (OOD) intents (Xu et al., 2020; Zeng et al., 2021a,b). In recent years, OOD intent detection (Hendrycks and Gimpel, 2017; Larson et al., 2019a; Lin and Xu, 2019; Ren et al., 2019; Xu et al., 2020; Zheng et al., 2020) has been well studied, which identifies whether a user query falls outside the range of pre-defined intent set to avoid performing wrong operations. But it can only safely reject OOD intents thus ignore these valuable OOD concepts for future development. Further, OOD intent discovery task (also known as new intent discovery) (Lin et al., 2020; Zhang et al., 2021b) is proposed to cluster unlabeled OOD data. The adopted clustering method can only group those OOD intents into clusters, but cannot further expand the recognition scope of the existing IND intent classifier incrementally.

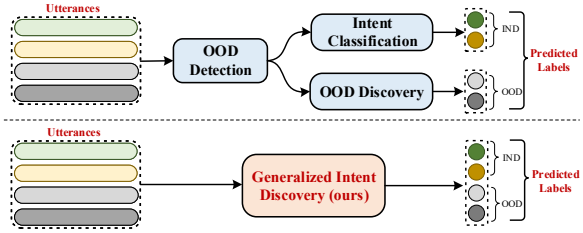

Figure 1: Illustration of our proposed GID task. The above subfig shows a practical intent classification system where an OOD detection module firstly identifies whether a test intent belongs to OOD, then an in-domain classifier and an OOD discoverer respectively recognize IND and OOD intents. In contrast, our proposed GID can simultaneously classify a set of labeled IND intent classes and new OOD types in an end-to-end manner.

Inspired by the above issues, we introduce a new task of extending and recognizing intent categories automatically, **G**eneralized **I**ntent **D**iscovery(**GID**). GID aims to extend an existing IND intent classifier to an open-world OOD intent set, as shown in Fig 1. The main motivation is that we hope to

*The first two authors contribute equally. Weiran Xu is the corresponding author.

[1]We release our code at https://github.com/myt517/GID_benchmark.

Proceedings of the 29th International Conference on Computational Linguistics, pages 707–720
October 12–17, 2022.

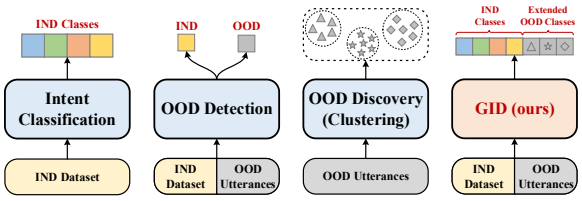

Figure 2: The comparison of GID to other related tasks.

train a network that can simultaneously classify a set of labeled IND intent classes while discovering new ones in an unlabeled OOD set. In this way, we can enhance the capability of an IC system by expanding its recognition scope incrementally. We show a comparison of GID and existing OOD tasks in Fig 2. Since the practical OOD intents are unsupervised, neither the OOD labels nor OOD intent schema make it different from zero-shot learning (Yan et al., 2020; Siddique et al., 2021) and continual learning (Xu et al., 2019) which both rely on a given label ontology, like label descriptions. Therefore, to explore unique characteristics of GID, we construct three kinds of GID benchmarks, including single domain, multiple domain, and cross-domain settings (Section 3). These settings denote different application scenarios which we will discuss later.

Subsequently, we propose two kinds of frameworks for GID, pipeline and end-to-end. A straightforward idea is pipeline-based methods which firstly learn OOD cluster assignments and get pseudo OOD labels, then jointly classify labeled IND data and pseudo labeled OOD data. However, pipeline-based methods separate OOD clustering and classification process, which ignores the interaction between labeled IND data and unlabeled OOD data. Besides, these pseudo OOD labels may induce severe noise to the joint classification, limiting the performance of the joint IND and OOD classifiers. Therefore, we further propose an end-to-end framework to simultaneously learn pseudo OOD cluster labels and classify IND&OOD classes along with ground truth IND labels via a unified objective. We obtain the pseudo label of an OOD query by its augmented view in a swapped prediction way (Caron et al., 2020; Asano et al., 2020; Fini et al., 2021) and employ the Sinkhorn-Knopp (SK) algorithm (Cuturi, 2013) to solve the optimization problem. We leave the details to Section 4. We also perform exhaustive experiments (Section 5.2) and qualitative analysis (Section 5.3) to shed light on the challenges that current approaches face with GID. We find fine-grained OOD types, domain gap, data imbalance, real OOD noise and estimating the number of OOD types are the main challenges (Section 6), which provide insightful guidance for future GID work.

Our contributions are four-fold: (1) We introduce a new task, Generalized Intent Discovery (GID) which aims to extend an IND intent classifier to an open-world OOD intent set. GID helps expand the model's recognition scope and develop new skills for improving dialogue systems. (2) We construct three kinds of public GID benchmarks for different application scenarios, which help to explore the key challenges of GID comprehensively. (3) We propose an end-to-end GID framework to jointly learn clustering and classification, and extensive baselines of two frameworks, pipeline-based and end-to-end for future work. (4) We conduct exhaustive experiments and qualitative analysis to comprehend key challenges and provide new guidance for future GID research.

2 Problem Formulation

In this section, we first briefly introduce the traditional intent classification (IC) task, then dive into the details of our proposed Generalized Intent Discovery (GID) task.

Intent Classification Given a labeled in-domain (IND) dataset $\mathbf{D}^{IND} = \{(x_1^{IND}, y_1^{IND}), \ldots, (x_n^{IND}, y_n^{IND})\}$, IC aims to predict the intent class of a test query by training an IND classifier, based on the assumption that all the queries belong to a pre-defined fixed set $\mathcal{Y}^{IND} = \{1, \ldots, N\}$ of N intent categories.

Generalized Intent Discovery In contrast, GID is to classify queries corresponding to both labeled IND and unlabeled OOD classes. Apart from the above labeled IND dataset $\mathbf{D}^{IND}$, an unlabeled OOD dataset $\mathbf{D}^{OOD} = \{(x_1^{OOD}), \ldots, (x_m^{OOD})\}$ is also given. For simplicity, we assume the number of OOD classes is specified as M. In practical scenarios, we can estimate the number of clusters following previous work (Zhang et al., 2021b) (see Section 5.3.4). Since these OOD intents are usually collected from an online IC system by rejecting low confident queries [2], the set of N IND classes

[2]For example, given a test query, if an IC model predicts an output with low confident probability, we can assume the query doesn't belong to any IND type but OOD intents. Please refer to related OOD detection work (Xu et al., 2020; Zeng et al., 2021c; Zheng et al., 2020) for details. In this paper, we focus on the joint classification of unlabeled OOD and labeled IND. Thus, we suppose the two sets of IND classes and OOD classes are disjoint from each other.

is assumed to be disjoint from the set of M OOD classes. We also provide a discussion about real OOD noise in Section 5.3.2. The final goal of GID is to classify an input query to the total label set $\mathcal{Y} = \{1, \ldots, N, N+1, \ldots, N+M\}$ where the first N elements denote labeled IND classes and the subsequent M ones denote unlabeled OOD classes. The challenges of GID come from two aspects, discovering the semantic concepts from unlabeled OOD data and jointly classifying IND&OOD intents. On the one hand, models need to automatically cluster OOD concepts which is more difficult than supervised classification tasks. On the other hand, they require jointly recognizing IND&OOD intents using these noisy cluster signals which may harm the final performance.

3 Dataset

To explore the practical significance and key challenges of GID task, we need to construct the GID dataset. However, we found that in some related tasks such as OOD intent discovery (Zhang et al., 2021b) and zero-shot intent detection (Siddique et al., 2021), the commonly used construction methods are to randomly divide the intent classification dataset into IND and OOD subset. This may not reflect real online intent classification scenarios.

We design more diverse GID dataset construction strategies, mainly in order to be able to discuss the practical significance and key challenges of GID more comprehensively. we construct three kinds of benchmark datasets GID-SD (single-domain), GID-MD (multiple-domain) and GID-CD (cross-domain) based on the two widely used intent datasets, CLINC (Larson et al., 2019b) and Banking (Casanueva et al., 2020). The three settings denote different real-world application scenarios in dialogue systems. Besides, we also construct two dataset variants GID-noise and GID-imbalance to explore more severe challenges of GID tasks in real scenes. We first briefly introduce original CLINC and Banking datasets, then elaborate on GID dataset construction, and display the statistic of GID benchmarks. Finally, we introduce evaluation metrics for the GID task, accuracy and F1 score both for IND and OOD data.

3.1 Original Intent Datasets

CLINC contains 22,500 queries covering 150 intents across 10 domains and Banking is a fine-grained dataset in a single domain, which contains 13,083 user queries with 77 intents. We show the detailed statistics of the two original datasets in Appendix A.1.

3.2 GID Dataset Construction

GID Benchmarks For CLINC and Banking datasets, we randomly choose the specified ratio (20%, 40%, 60%) of all intent classes as OOD types, and the rest are IND, similar to Xu et al. (2020); Zhang et al. (2021b).[3] The original train/val/test split is fixed. We only keep IND queries with their labels and the queries belonging to OOD classes in the original train and val data. Note that GID assumes OOD training data is unlabeled so we remove OOD queries' labels in the original train and val data. In the test set, we keep all the original IND and OOD intents and labels for evaluating metrics.[4] Considering different scenarios of dialogue systems, we construct three benchmarks, GID-SD (single-domain), GID-MD (multiple-domain) and GID-CD (cross-domain). Specifically, for the single-domain Banking dataset, we randomly select the specified ratio of all intent classes as OOD types, and the rest are IND to construct GID-SD. Since Banking has a large intent set in a single domain, we find these fine-grained OOD types are difficult to recognize (see Section 5.2). For the multiple-domain CLINC dataset, we propose two split strategies: (1) **Overlapping** (for GID-MD): We neglect the domain constraint and randomly split all the intent classes into the IND set and OOD set as above, which means intent categories from a domain may be divided to the two sets, which we call Domain Overlapping [5]. The situation occurs where an online IC system can hardly cover all the intent classes in a domain and OOD intents may come from the same domain as IND. (2) **Non-Overlapping** (for GID-CD): We restrict IND intent classes and OOD classes are from different domains, so we select a ratio of all domains as IND and the rest as OOD. Once a domain is chosen as IND, all the intents in this domain belong to IND intent classes and vice versa. The non-overlapping setting is more practical in a real scenario where we need to transfer a business to another.

[3] To avoid randomness, we report the averaged experiment results of three runs for each ratio. And for each run, all the models are based on the same dataset IND/OOD split.

[4] Although CLINC contains a real unlabeled OOD set, we can't use it because not able to evaluate the performance of models. We use the set for constructing a noisy GID dataset.

[5] Please mind IND intent classes and OOD classes are still disjoint from each other, but may belong to the same domain.

Dataset	IND classes	OOD classes	IND domains	OOD domains	Train samples	Val samples	Test samples
GID-SD-40%	46	31	1	1	5414/3589	600/400	1840/1240
GID-MD-40%	90	60	10	10	10,800/7200	1350/900	1350/900
GID-CD-40%	90	60	6	4	10,800/7200	1350/900	1350/900

Table 1: Statistics of GID-SD-40%, GID-MD-40% and GID-CD-40%.

GID Dataset Variants To explore more severe challenges of GID tasks in real applications, we construct two variants based on GID-MD-40%, GID-noise and GID-imbalance. (1) **GID-noise**: In the standard GID setting, we suppose the OOD data in the training set is "clean", that is, each OOD query must belong to a specific intent category. However, in practice, some OOD queries may be meaningless and not belong to any intent cluster, which we call OOD noise. We use 1350 real out-of-scope(oos) samples in CLINC, which semantically do not belong to any intent category in the training set, and add these noisy samples into the OOD train set to see if performance changes (see Section 5.3.2). Specifically, we add different numbers of oos samples according to 5%, 10% and 15% of the number of OOD samples in the training set of GID-MD-40%. (2) **GID-imbalance**: Data imbalance is a common issue in practice. To explore the impact of OOD data imbalance, we construct imbalanced GID datasets with different imbalance ratios ($\rho = 2, 3, 6$) by sampling each class of OOD samples in the GID-MD-40% training set. Following (Zhang et al., 2021c; Hong et al., 2021), we first sort the OOD classes of GID-MD-40% and each class is assigned an index $j (j = 1, 2, 3, ..., M)$, where M denotes the total number of OOD intent categories. Then we sample from each OOD class according to $n_j = n_{min}\rho^{(j-1)/M}$, $j = 1, 2, 3, ..., M$, where n_{min} is the least number of samples across all OOD classes. We adjust different imbalance ratios $\rho = n_{max}/n_{min}$ to simulate the degree of imbalance. $n_{max} = 120$ is the max number of samples per class in GID-MD-40%. We put the detailed statistics of GID-imbalance in Appendix A.2.

3.3 Statistic of GID Datasets and Evaluation

Since different proportions of OOD intents have different statistics, here we only display the results of 40% OOD for brevity. Table 1 shows the statistics of GID-SD-40%, GID-MD-40%, GID-CD-40%.

We use intent accuracy (ACC) and macro F1 as evaluation metrics for GID task. We report all IND, OOD and total (ALL) metrics where OOD and ALL ACC/F1 are the main metrics. Following

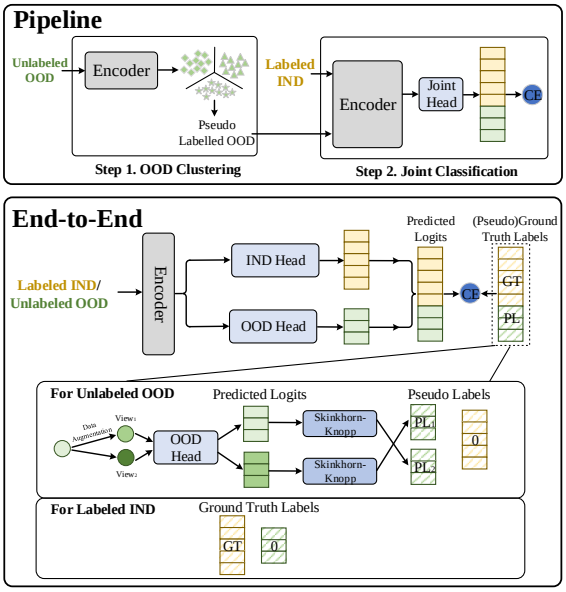

Figure 3: Overall architecture of our proposed pipeline and end-to-end methods.

Zhang et al. (2021b), we use the Hungarian algorithm (Kuhn, 1955) to obtain the mapping between the predicted OOD classes and ground-truth classes in the test set.

4 Method

Overall Architecture We extend the idea of traditional intent classification models by using pseudo OOD labels. IC calculates the N-dimension cross-entropy (CE) loss for labeled IND data (Qin et al., 2019; He et al., 2020). Similarly, we can compute (N+M)-dimension CE loss both for labeled IND and unlabeled OOD data where IND labels are given but OOD (pseudo) labels are estimated (Zhang et al., 2021b; Han et al., 2020; Fini et al., 2021). Thus, the key challenge is to estimate OOD pseudo cluster labels by transferring prior IND knowledge. We propose two kinds of frameworks, pipeline and end-to-end, shown in Fig 3.

Pipeline A simple idea is pipeline-based methods which firstly learn OOD cluster assignments, then jointly classify labeled IND data and pseudo labeled OOD data. Specifically, we use the same BERT intent encoder as DeepAligned (Zhang et al., 2021b) to cluster OOD data. To transfer prior knowledge, we first pre-train the encoder on IND data to get intent representations. Then, we respectively use two OOD clustering methods, k-means (MacQueen, 1967) and DeepAligned to obtain pseudo OOD labels $\hat{\boldsymbol{y}}^{OOD}$. Finally, we mix up all the IND and OOD data and construct the new

(N+M)-dimension intent label $\boldsymbol{y}$ as follows:

$$\boldsymbol{y} = \begin{cases} [\boldsymbol{y}^{IND}; \mathbf{0}_M] & \mathbf{x} \in \mathbf{D}^{IND} \\ [\mathbf{0}_N; \hat{\boldsymbol{y}}^{OOD}] & \mathbf{x} \in \mathbf{D}^{OOD} \end{cases} \quad (1)$$

where $\boldsymbol{y}^{IND}, \hat{\boldsymbol{y}}^{OOD}$ are one-hot labels and $\mathbf{0}_M, \mathbf{0}_N$ are M or N-dimention zero vectors. We use the original CE loss to train a (N+M)-class open-set intent classifier.

End-to-End The main drawback of pipeline methods is the lack of deep semantic interaction between IND and OOD data in the clustering stage, leading to poor pseudo cluster labels. To alleviate the issue, we adopt an end-to-end framework to simultaneously learn pseudo OOD cluster labels and classify IND&OOD classes, shown in Fig 3. Our motivation is that each view of an OOD intent query after data augmentation can predict the other's pseudo labels, following swapped prediction (Caron et al., 2020). And we can learn the simple pseudo-labeling process via the unified classification loss instead of extra clustering objectives. Specifically, we use the same pre-trained BERT encoder in IND data as pipeline and two independent projection layers, IND head I and OOD head O. Given an input query, we concat the outputs of two heads as the final logit. For labeled IND intents, the ground-truth labels are easily obtained by Eq 1. We now discuss how to get the pseudo labels of unlabeled OOD intents. Inspired by Caron et al. (2020); Asano et al. (2020); Fini et al. (2021), we use the following swapped prediction way:

$$\ell_{CE}\left(\boldsymbol{x}_1, \hat{\boldsymbol{y}}_2\right) + \ell_{CE}\left(\boldsymbol{x}_2, \hat{\boldsymbol{y}}_1\right) \quad (2)$$

where $\boldsymbol{x}_1, \boldsymbol{x}_2$ are two dropout-augmented (Gao et al., 2021) views from an OOD intent query and $\hat{\boldsymbol{y}}_1, \hat{\boldsymbol{y}}_2$ are corresponding pseudo labels. We use $\boldsymbol{x}_1$ to compute $\hat{\boldsymbol{y}}_1$ and $\boldsymbol{x}_2$ for $\hat{\boldsymbol{y}}_2$. A simple way of obtaining $\hat{\boldsymbol{y}}_1$ from $\boldsymbol{x}_1$ is to regard the predicted softmax logits after OOD head of $\boldsymbol{x}_1$ as $\hat{\boldsymbol{y}}_1$. But Asano et al. (2020) observes this strategy easily leads to degenerate solutions where all the intents predict the same pseudo label and are grouped into the same cluster. Therefore, we add an entropy penalty to avoid all the pseudo labels are equal to each other and keep more uniform distribution of the pseudo-labels over all the M OOD clusters. We formulate the new optimization way:

$$\hat{\mathbf{Y}}^* = \arg\max_{\hat{\mathbf{Y}} \in \Gamma} \mathrm{Tr}(\hat{\mathbf{Y}}\boldsymbol{L}) + \epsilon \mathrm{H}(\hat{\mathbf{Y}}) \quad (3)$$

where $\hat{\mathbf{Y}} = [\hat{\boldsymbol{y}}_1, \ldots, \hat{\boldsymbol{y}}_B]^\top$ is the matrix whose columns are the unknown pseudo-labels of the current batch B and $\boldsymbol{L} = [\boldsymbol{l}_1, \ldots, \boldsymbol{l}_B]$ is the predicted logits by the OOD head. H is the entropy function and ϵ is an hyper-parameter(we set it to 0.05 in the experiments). The goal is to obtain the best pseudo-labels $\hat{\mathbf{Y}}^*$ by maximizing Eq 3. And $\hat{\mathbf{Y}}$ must meet the following constraints similar to Caron et al. (2020); Fini et al. (2021):

$$\Gamma = \{\hat{\mathbf{Y}} \in \mathbb{R}_+^{M \times B} \mid \hat{\mathbf{Y}}\mathbf{1}_B = \frac{1}{M}\mathbf{1}_M, \hat{\mathbf{Y}}^\top \mathbf{1}_M = \frac{1}{B}\mathbf{1}_B\} \quad (4)$$

where $\mathbf{1}_B$ denotes the vector of all ones with B dimensions. Essentially, Eq 3&4 can be regarded as an optimal transport problem and we use the Sinkhorn-Knopp (SK) algorithm (Cuturi, 2013) to solve it.[6] After we get the pseudo OOD labels in a mini-batch, we can use Eq 1 to compute the CE loss. Note that the losses of IND and OOD data in a batch are jointly optimized. Compared to pipeline methods, our end-to-end method can simultaneously learn pseudo OOD cluster labels and distinguish IND&OOD classes via a CE loss. Joint optimization enables semantic interaction between IND and OOD data for better knowledge transfer and to reduce noisy clustering signals. For inference, we forward the input query (including IND and OOD) to the model and obtain its prediction.

5 Experiments and Analysis

5.1 Baselines

k-means A pipeline baseline, which first uses k-means (MacQueen, 1967) to cluster OOD data and obtains pseudo OOD labels, and then trains a new classifier together with IND data.

DeepAligned Similar to k-means, the difference is that the clustering algorithm adopts DeepAligned (Zhang et al., 2021b), which is the current state-of-the-art method for OOD discovery task.

DeepAligned-Mix This is an end-to-end approach where we extend DeepAligned for GID. DeepAligned is an iterative clustering method. In each iteration, it firstly uses k-means and an alignment strategy to cluster and label the OOD data and then computes the cross-entropy classification for representation learning. Our proposed DeepAligned-Mix mainly improves two points: (1) We mix up IND and OOD data together for iterative clustering, and the model is optimized with

[6] We recommend referring to Cuturi (2013) for more details about the theoretical explanation of optimal transport and SK algorithm.

Method	GID-SD-20%					GID-SD-40%					GID-SD-60%				
	IND	OOD		ALL		IND	OOD		ALL		IND	OOD		ALL	
	ACC	ACC	F1	ACC	F1	ACC	ACC	F1	ACC	F1	ACC	ACC	F1	ACC	F1
k-means	91.29	70.50	71.43	87.21	86.90	90.38	62.34	62.44	78.99	78.32	90.40	51.58	51.96	67.08	66.70
DeepAligned	92.00	76.44	77.40	88.94	88.60	91.72	69.11	69.72	82.57	82.10	90.97	59.55	59.51	72.05	71.42
DeepAligned-Mix	85.62	56.28	60.26	79.90	78.20	82.30	54.97	59.79	71.30	69.60	80.70	52.66	54.66	63.95	61.92
End-to-End	92.82	**81.78**	**83.53**	**90.67**	**90.64**	92.84	**72.28**	**73.28**	**84.49**	**84.10**	**92.45**	**62.63**	**62.65**	**74.59**	**73.99**

Table 2: Performance on GID-SD (single-domain). 20%, 40% and 60% denotes the ratio of OOD intents. Results are averaged over three random run.($p < 0.01$ under t-test)

Method	GID-MD-20%					GID-MD-40%					GID-MD-60%				
	IND	OOD		ALL		IND	OOD		ALL		IND	OOD		ALL	
	ACC	ACC	F1	ACC	F1	ACC	ACC	F1	ACC	F1	ACC	ACC	F1	ACC	F1
k-means	97.22	76.22	75.03	93.02	92.74	97.26	73.00	72.66	87.56	87.08	95.00	65.11	63.68	77.02	76.09
DeepAligned	97.83	90.89	91.08	96.43	96.32	97.85	87.55	87.14	93.70	93.29	97.67	83.38	82.78	89.10	88.52
DeepAligned-Mix	95.91	81.93	83.93	93.11	92.54	92.86	81.70	83.30	88.12	87.42	92.59	78.34	79.88	84.05	82.74
End-to-End	98.17	**95.26**	**96.08**	**97.58**	**97.59**	98.32	**91.92**	**92.46**	**95.78**	**95.73**	98.26	**87.63**	**87.84**	**91.88**	**91.78**

Table 3: Performance on GID-MD (multiple-domain).

Method	GID-CD-20%					GID-CD-40%					GID-CD-60%				
	IND	OOD		ALL		IND	OOD		ALL		IND	OOD		ALL	
	ACC	ACC	F1	ACC	F1	ACC	ACC	F1	ACC	F1	ACC	ACC	F1	ACC	F1
k-means	97.39	75.78	75.79	92.98	92.72	97.70	61.67	60.43	83.20	82.30	96.44	54.67	53.69	71.38	70.57
DeepAligned	97.83	84.81	84.22	95.23	95.01	97.85	78.55	77.81	90.12	89.68	97.33	76.15	74.80	84.62	83.60
DeepAligned-Mix	97.15	77.41	77.7	93.20	92.53	97.33	72.41	71.54	87.36	86.21	93.89	75.63	74.29	82.93	81.37
End-to-End	97.92	**87.41**	**87.55**	95.81	**95.75**	98.00	**79.19**	**79.06**	90.46	**90.28**	98.22	**78.01**	**77.48**	**86.09**	**85.63**

Table 4: Performance on GID-CD (cross-domain).

a unified cross-entropy loss; (2) In the inference stage, instead of using k-means for clustering, we use the classification head of the new classifier to make predictions.

5.2 Main Results

We conduct experiments on three benchmark GID datasets GID-SD, GID-MD and GID-CD with different OOD ratios, shown in Table 4. In general, Our proposed end-to-end (E2E) method consistently outperforms all the baselines with a large margin. We analyze the results from three aspects:

Comparison of different methods We see E2E significantly outperforms all the baselines under the three datasets and different OOD ratio settings. For example, E2E outperforms previous state-of-the-art DeepAligned by 3.14%(OOD F1) and 2.57%(ALL F1) on GID-SD-60%, 5.06%(OOD F1) and 3.26%(ALL F1) on GID-MD-60%, 2.68%(OOD F1) and 2.03%(ALL F1) on GID-CD-60%. These prove that joint clustering and classification helps to perform more interaction between IND and OOD and obtain accurate pseudo OOD labels. We also observe E2E achieves slightly better IND ACC than pipeline methods, which means joint classification doesn't sacrifice IND performance while improving OOD recognition.

Comparison of different datasets To explore the effect of different practical scenarios, we compare the performance of the same method on different datasets. Results show metrics on GID-SD are the lowest, GID-CD is in the middle and GID-MD is the best for almost all the methods, which denotes the difficulty order is single-domain>cross-domain>multiple-domain. We argue GID-SD contains more fine-grained intent types in a single domain which makes it challenging to recognize OOD intents. Comparing CD and MD, IND and OOD types from the same domain makes it easier to transfer prior knowledge, so MD gets higher scores.

Effect of different OOD ratios We compare the results of different OOD ratios on the same dataset. We find with the increase of OOD ratio, the performance consistently drops. For example, E2E achieves 95.26% OOD ACC on GID-MD-20%, but OOD ACC decreases by 3.34% on GID-MD-40% and 7.63% on GID-MD-60%. Intuitively, the increase in the number of OOD intents makes it more difficult to distinguish them.

5.3 Qualitative Analysis

5.3.1 Cross-Domain Transferability

For GID-CD, cross-domain knowledge transfer is important and challenging. To study the effect of domain similarity on knowledge transfer, we perform a cross-domain transferability analysis in Fig

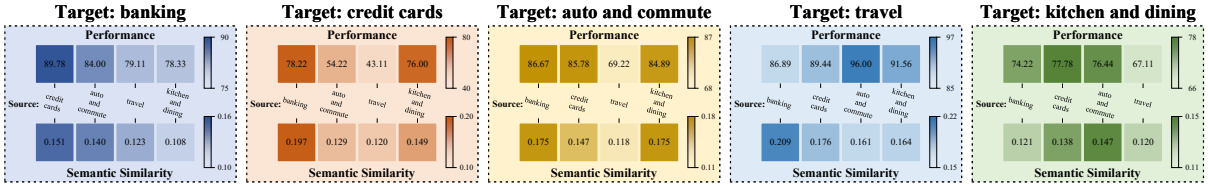

Figure 4: Cross-domain transferability from source IND to target OOD. We display OOD ACC and domain similarity scores. The larger the number is, the deeper the color is.

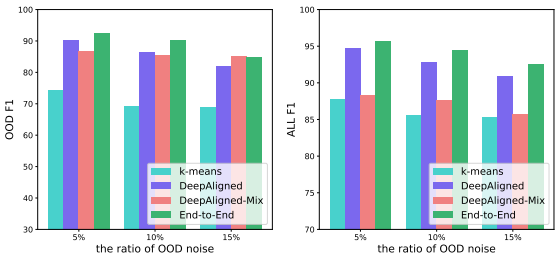

Figure 5: The impact of adding different numbers of noisy OOD samples to the training set on the performance of each GID model.

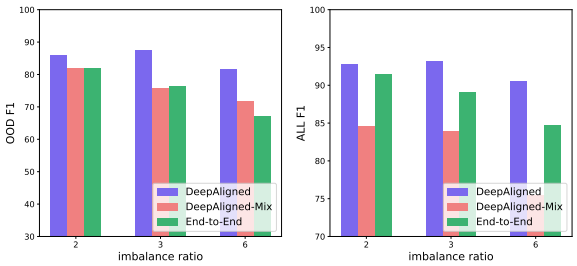

Figure 6: The impact of different imbalance ratios of OOD data on the performance of each GID model.

4. We select five domains (banking, credit_card, auto_and_commute, travel, kitchen_and_dining) and perform the one-to-one transfer. To measure domain similarity, we first train an IND intent classifier, then perform k-means using extracted representations of OOD samples to calculate Silhouette Coefficient (SC) values (Rousseeuw, 1987) [7]. We can see that the larger the SC value is, the higher the similarity between IND&OOD domains is, resulting in better OOD metrics. The results prove good cross-domain transferability comes from semantically similar domains, such as banking and credit_card.

5.3.2 Effect of OOD noise

In the real world, OOD data may not necessarily belong to a certain OOD cluster, and there is often some OOD noise. We use the constructed dataset variant GID-noise to examine the impact of noisy OOD in the training set on model performance. Fig 5 shows the impact of different amounts of OOD noise in the training set on model performance. The results show that as the amount of OOD noise increases, the OOD performance drops. Our proposed E2E still achieves the best performance over all baselines. We argue that this is because the presence of OOD noise makes it difficult for the model to learn a clear cluster boundary for unlabeled OOD.

[7] Please see more details about SC in Appendix A.6.

5.3.3 Effect of imbalanced OOD data

Fig 6 shows the impact of class imbalance degree of OOD data on model performance. The results show that when the imbalance degree of OOD categories increases, the performance of all models decreases significantly. We also find an interesting phenomenon that our proposed end-to-end method drops more significantly than pipeline-based DeepAligned. We argue that there are two reasons for this. (1) When our end-to-end method obtains OOD pseudo-labels, the SK algorithm is based on a strong assumption, the number of pseudo-labels for each category in a batch is uniform, which is obviously invalid in the class-imbalanced scenario. (2) E2E uses IND and OOD to jointly train the classifier. Since the number of samples in each class of IND keeps fixed to 120(equal to the number of OOD samples in the majority class of OOD), this will exacerbate the degree of imbalance and affect the accuracy of pseudo-labels for long-tail categories. Therefore, we need to further explore better pseudo-label methods in the future and how to improve the class-imbalanced defect of end-to-end methods.

5.3.4 Estimate the Number of Cluster K

All the results we showed so far assume that the number of OOD classes is pre-defined. However, in real-world applications, this often needs to be estimated automatically. Table 5 shows the results using the same automatic K-value estimation strategy

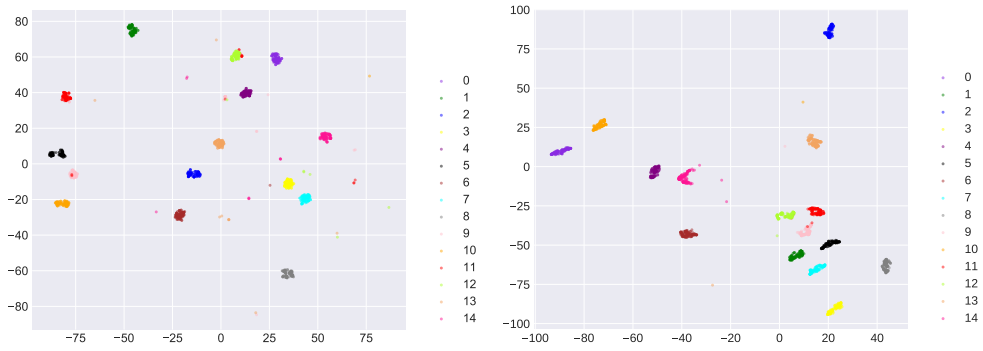

(a) DeepAligned (b) End-to-End

Figure 7: IND & OOD intents visualazation of DeepAligned and E2E method, we select 9 IND intents and 6 OOD intents in GID-MD-40% (index 0-8 denotes IND intents, index 9-14 denotes OOD intents)

	OOD ACC	OOD F1	ALL ACC	K
DeepAligned	87.55	87.14	93.70	60
DeepAligned-Mix	82.70	84.65	88.12	60
End-to-End	91.92	92.46	95.78	60
DeepAligned	72.89	66.75	87.91	47
DeepAligned-Mix	69.56	62.32	85.29	47
End-to-End	74.89	67.23	88.58	47

Table 5: Estimate the number of OOD clusters. K=47 is the estimated number compared to original 60.

[8]. We find that our method both achieves the best performance under the fixed or auto K settings. It should be noted that no matter the end-to-end methods or the pipeline methods, the performance drops significantly when the number of OOD classes is unknown. Therefore, how to estimate an accurate K value and how to design a more robust GID method is a great challenge.

5.3.5 Visualization

To further visually compare the performance of end-to-end and pipeline methods in classifying IND and clustering OOD, we performed a visualization of IND & OOD intent representations for E2E and DeepAligned, as shown in Fig 7. Comparing E2E to DeepAligned, we can observe DeepAligned gets some mixed OOD clusters (see greenyellow and red dots in Fig a) while E2E method successfully separates them. We also find that many OOD intents in DeepAligned that cannot be clustered into single cluster, but are scattered into multiple clusters (see deeppink dots in Fig a), but E2E method can form compact clusters for them. We argue this is because the pipeline method introduces serious error propagation in the OOD clustering stage; while the E2E method jointly learns OOD cluster assignments and classification of IND & OOD, which helps to get clear cluster boundary.

5.3.6 Noise of IND

In the general GID setting, we assume that the IND and OOD categories do not overlap, however the OOD data collected in practical application scenarios may have some IND noise due to the error propagation of OOD detection. We analyze the performance changes of each GID method when mixing different proportions of IND noise in OOD data, as shown in Fig 10. The results show that our E2E method still significantly outperforms the pipeline baseline under IND noise scenarios. The performance of IND classification and OOD clustering for all methods decrease significantly, and the IND performance decrease is more significant for DeepAligned and E2E. We argue that this is due to the inclusion of a small amount of IND data in the OOD data, which causes these IND data to be incorrectly labeled, and severely impairs the performance of IND classification, making it difficult to form clear IND class boundaries. We also found that when the IND noise ratio reached 15%, the OOD clustering performance of the E2E method was worse than DeepAligned. We argue that this is because the E2E method jointly learns to classify IND intents and discover OOD intents, which needs to leverage IND prior knowledge to enhance OOD clustering. However, When there is more IND noise to be mixed with OOD data, it will affect the effectiveness of the knowledge interaction between IND and OOD. In practical applications, when the performance of OOD detection is improved, this IND noise problem can be relieved naturally, which is not within the scope of this papar.

[8]Here we use the same estimation algorithm as Zhang et al. (2021b). We leave the details in Appendix A.4.

6 Challenges

Based on the above analysis, we summarize the current challenges faced by the GID task:

Fine-grained OOD data When OOD intents are fine-grained like GID-SD, the OOD performance of existing GID methods decreases significantly. We argue fine-grained OOD intents make it hard to construct clear boundary while clustering.

Cross-domain transfer When IND and OOD intent types are from different distant domains, the knowledge learned from IND is hard to transfer to OOD due to the semantic gap in different domains.

OOD noise OOD data collected in practical applications are usually noisy, and there may be some OOD samples that do not belong to a certain intent type. The performance of each GID method degrades when trained with these noisy OOD data.

imbalanced OOD data The OOD data in real-world scenarios is often class-imbalanced, and our analysis in section 5.3.3 proves that the performance of current methods drops significantly under imbalanced data, especially end-to-end methods.

Inaccurate estimation of the number of OOD categories Most previous work assume the number of OOD categories is known. However, in practical applications, we usually need to estimate the number of categories K, which is often inaccurate. We propose a preliminary analysis in Section 5.3.4 which shows significant performance drop when the estimation is not totally accurate.

7 Related Work

OOD Detection aims to know when a query falls outside the range of pre-defined supported intents (Zeng et al., 2021a; Lin and Xu, 2019; Xu et al., 2020; Zeng et al., 2021b; Wu et al., 2022) to avoid performing wrong operation. OOD detection has attracted more and more attention in recent years, so various similar names are derived, such as anomaly detection, open world classification (Shu et al., 2021), open-world learning (Xu et al., 2019), open intent classification(Zhang et al., 2021a) and so on. However, all of them are essentially to distinguish whether a query belongs to IND or OOD intents, without further discovering new semantic categories from unsupervised OOD data.

OOD Discovery aims to discover new intent concepts from unlabeled OOD data and form OOD intent clusters (Lin et al., 2020; Zhang et al., 2021b; Mou et al., 2022), which focuses more on how to cluster OOD data, while ignoring the fusion of IND and OOD, which makes the model only recognize OOD intents. For example, (Zhang et al., 2021b) design an iterative clustering algorithm DeepAligned, which iteratively learns intent representations then cluster assignments. **Open Intent Extraction** also aims to extract unknown intents from unlabelled user queries (Vedula et al., 2020), and is a completely unsupervised task. However, in terms of method, open intent extraction is more about extracting intent names through sequence annotation methods. In contrast, GID aims to train a network that can simultaneously classify a set of labeled IND intent classes while discovering and recognizing unlabeled OOD intents.

Incremental/Continual Learning There is currently some work on extending closed-set classifier to new classes in the open world incrementally, such as (Xu et al., 2019). But all these works follow a traditional incremental learning setting, which requires new category data with labels. In practical applications, we can only obtain these unlabeled OOD data from the dialogue system logs, and these data are often updated continuously, and human annotation of these data is very labor-intensive. Therefore, we propose a more human-free task GID, which aims to automatically discover new categories from the unlabeled OOD data, and further expand the recognition scope of the existing IND intent classifier incrementally.

Zero-shot Intent Detection Zero-shot intent detection (Yan et al., 2020; Siddique et al., 2021) assumes that no target domain data is available during training, but the category and category descriptions from target domain are given, but in practical applications we often have access to a large amount of unlabeled dialogue logs, and we need to consider how to discover new intent categories from them for system development.

8 Conclusion

In this paper, we introduce a new task, Generalized Intent Discovery (GID), which aims to extend an IND intent classifier to an open-world intent set. Then we provide three public datasets for different application scenarios and establish a benchmark for the GID task. We also propose extensive baselines of two frameworks, pipeline-based and end-to-end for future work. Further, We conduct exhaustive experiments and qualitative analysis to comprehend key challenges and provide new guidance for future GID research.

Acknowledgements

We thank all anonymous reviewers for their helpful comments and suggestions. This work was partially supported by MoE-CMCC "Artifical Intelligence" Project No. MCM20190701, National Key R&D Program of China No. 2019YFF0303300 and Subject II No. 2019YFF0303302, DOCOMO Beijing Communications Laboratories Co., Ltd.

Broader Impact

Task-oriented dialogue systems have demonstrated remarkable performance in a wide range of applications, and have significant positive impact on human production mode and lifeway. Intent classification is an important component of task-oriented dialogue system. Existing intent classification models can only identify a limited number of predefined in-domain (IND) intents, however, out-of-domain (OOD) or unknown intents will appear continually when the dialogue system is deployed online. If we can group these OOD samples into different clusters, we can discover new intents, guide future development of the system, and expand the classification capabilities of the system. We note that OOD intent detection and OOD intent discovery tasks have been widely studied recently. The former focuses on identifying whether a sample is IND or OOD, while the latter focuses on how to cluster OOD data. The generalized intent discovery (GID) task proposed in this paper focuses on an incremental setting, that is simultaneously classifying a set of labeled IND intent classes while discovering and recognizing new unlabeled OOD types incrementally. GID aims to provide the model with the ability to automatically learning according to known knowledge in the open world, which is a new attempt for scalable dialogue system and open world learning.

References

Yuki Markus Asano, Christian Rupprecht, and Andrea Vedaldi. 2020. Self-labelling via simultaneous clustering and representation learning. In *8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, April 26-30, 2020*. OpenReview.net.

Mathilde Caron, Ishan Misra, Julien Mairal, Priya Goyal, Piotr Bojanowski, and Armand Joulin. 2020. Unsupervised learning of visual features by contrasting cluster assignments. In *Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020, December 6-12, 2020, virtual*.

Iñigo Casanueva, Tadas Temčinas, Daniela Gerz, Matthew Henderson, and Ivan Vulić. 2020. Efficient intent detection with dual sentence encoders. In *Proceedings of the 2nd Workshop on Natural Language Processing for Conversational AI*, pages 38–45, Online. Association for Computational Linguistics.

Qian Chen, Zhu Zhuo, and Wen Wang. 2019. Bert for joint intent classification and slot filling. *ArXiv preprint*, abs/1902.10909.

Marco Cuturi. 2013. Sinkhorn distances: Lightspeed computation of optimal transport. In *Advances in Neural Information Processing Systems 26: 27th Annual Conference on Neural Information Processing Systems 2013. Proceedings of a meeting held December 5-8, 2013, Lake Tahoe, Nevada, United States*, pages 2292–2300.

Haihong E, Peiqing Niu, Zhongfu Chen, and Meina Song. 2019. A novel bi-directional interrelated model for joint intent detection and slot filling. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 5467–5471, Florence, Italy. Association for Computational Linguistics.

Enrico Fini, E. Sangineto, Stéphane Lathuilière, Zhun Zhong, Moin Nabi, and Elisa Ricci. 2021. A unified objective for novel class discovery. *ArXiv preprint*, abs/2108.08536.

Tianyu Gao, Xingcheng Yao, and Danqi Chen. 2021. Simcse: Simple contrastive learning of sentence embeddings. *ArXiv preprint*, abs/2104.08821.

Chih-Wen Goo, Guang Gao, Yun-Kai Hsu, Chih-Li Huo, Tsung-Chieh Chen, Keng-Wei Hsu, and Yun-Nung Chen. 2018. Slot-gated modeling for joint slot filling and intent prediction. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 2 (Short Papers)*, pages 753–757, New Orleans, Louisiana. Association for Computational Linguistics.

Kai Han, Sylvestre-Alvise Rebuffi, Sébastien Ehrhardt, Andrea Vedaldi, and Andrew Zisserman. 2020. Automatically discovering and learning new visual categories with ranking statistics. In *8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, April 26-30, 2020*. OpenReview.net.

Keqing He, Shuyu Lei, Yushu Yang, Huixing Jiang, and Zhongyuan Wang. 2020. Syntactic graph convolutional network for spoken language understanding. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 2728–2738, Barcelona, Spain (Online). International Committee on Computational Linguistics.

Dan Hendrycks and Kevin Gimpel. 2017. A baseline for detecting misclassified and out-of-distribution examples in neural networks. In *5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Conference Track Proceedings*. OpenReview.net.

Youngkyu Hong, Seungju Han, Kwanghee Choi, Seokjun Seo, Beomsu Kim, and Buru Chang. 2021. Disentangling label distribution for long-tailed visual recognition. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 6626–6636.

H. Kuhn. 1955. The hungarian method for the assignment problem. *Naval Research Logistics Quarterly*, 2:83–97.

Stefan Larson, Anish Mahendran, Joseph J. Peper, Christopher Clarke, Andrew Lee, Parker Hill, Jonathan K. Kummerfeld, Kevin Leach, Michael A. Laurenzano, Lingjia Tang, and Jason Mars. 2019a. An evaluation dataset for intent classification and out-of-scope prediction. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 1311–1316, Hong Kong, China. Association for Computational Linguistics.

Stefan Larson, Anish Mahendran, Joseph J. Peper, Christopher Clarke, Andrew Lee, Parker Hill, Jonathan K. Kummerfeld, Kevin Leach, Michael A. Laurenzano, Lingjia Tang, and Jason Mars. 2019b. An evaluation dataset for intent classification and out-of-scope prediction. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 1311–1316, Hong Kong, China. Association for Computational Linguistics.

Ting-En Lin and Hua Xu. 2019. Deep unknown intent detection with margin loss. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 5491–5496, Florence, Italy. Association for Computational Linguistics.

Ting-En Lin, Hua Xu, and Hanlei Zhang. 2020. Discovering new intents via constrained deep adaptive clustering with cluster refinement. In *The Thirty-Fourth AAAI Conference on Artificial Intelligence, AAAI 2020, The Thirty-Second Innovative Applications of Artificial Intelligence Conference, IAAI 2020, The Tenth AAAI Symposium on Educational Advances in Artificial Intelligence, EAAI 2020, New York, NY, USA, February 7-12, 2020*, pages 8360–8367. AAAI Press.

Bing Liu and Ian R. Lane. 2016. Attention-based recurrent neural network models for joint intent detection and slot filling. In *INTERSPEECH*.

J. MacQueen. 1967. Some methods for classification and analysis of multivariate observations.

Yutao Mou, Keqing He, Yanan Wu, Zhiyuan Zeng, Hong Xu, Huixing Jiang, Wei Wu, and Weiran Xu. 2022. Disentangled knowledge transfer for OOD intent discovery with unified contrastive learning. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 46–53, Dublin, Ireland. Association for Computational Linguistics.

Libo Qin, Wanxiang Che, Yangming Li, Haoyang Wen, and Ting Liu. 2019. A stack-propagation framework with token-level intent detection for spoken language understanding. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 2078–2087, Hong Kong, China. Association for Computational Linguistics.

Jie Ren, Peter J. Liu, Emily Fertig, Jasper Snoek, Ryan Poplin, Mark A. DePristo, Joshua V. Dillon, and Balaji Lakshminarayanan. 2019. Likelihood ratios for out-of-distribution detection. In *Advances in Neural Information Processing Systems 32: Annual Conference on Neural Information Processing Systems 2019, NeurIPS 2019, December 8-14, 2019, Vancouver, BC, Canada*, pages 14680–14691.

Peter J Rousseeuw. 1987. Silhouettes: a graphical aid to the interpretation and validation of cluster analysis. *Journal of computational and applied mathematics*, 20:53–65.

Lei Shu, Yassine Benajiba, Saab Mansour, and Yi Zhang. 2021. Odist: Open world classification via distributionally shifted instances. In *Findings of the Association for Computational Linguistics: EMNLP 2021*, pages 3751–3756.

AB Siddique, Fuad Jamour, Luxun Xu, and Vagelis Hristidis. 2021. Generalized zero-shot intent detection via commonsense knowledge. In *Proceedings of the 44th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 1925–1929.

Nikhita Vedula, Nedim Lipka, Pranav Maneriker, and Srinivasan Parthasarathy. 2020. Open intent extraction from natural language interactions. In *WWW '20: The Web Conference 2020, Taipei, Taiwan, April 20-24, 2020*, pages 2009–2020. ACM / IW3C2.

Yanan Wu, Keqing He, Yuanmeng Yan, QiXiang Gao, Zhiyuan Zeng, Fujia Zheng, Lulu Zhao, Huixing Jiang, Wei Wu, and Weiran Xu. 2022. Revisit overconfidence for ood detection: Reassigned contrastive learning with adaptive class-dependent threshold. In *NAACL*.

Hong Xu, Keqing He, Yuanmeng Yan, Sihong Liu, Zijun Liu, and Weiran Xu. 2020. A deep generative distance-based classifier for out-of-domain detection with mahalanobis space. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 1452–1460, Barcelona, Spain (Online).

International Committee on Computational Linguistics.

Hu Xu, Bing Liu, Lei Shu, and Philip S. Yu. 2019. Open-world learning and application to product classification. In *The World Wide Web Conference, WWW 2019, San Francisco, CA, USA, May 13-17, 2019*, pages 3413–3419. ACM.

Guangfeng Yan, Lu Fan, Qimai Li, Han Liu, Xiaotong Zhang, Xiao-Ming Wu, and Albert Y.S. Lam. 2020. Unknown intent detection using Gaussian mixture model with an application to zero-shot intent classification. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 1050–1060, Online. Association for Computational Linguistics.

Zhiyuan Zeng, Keqing He, Yuanmeng Yan, Zijun Liu, Yanan Wu, Hong Xu, Huixing Jiang, and Weiran Xu. 2021a. Modeling discriminative representations for out-of-domain detection with supervised contrastive learning. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 2: Short Papers)*, pages 870–878, Online. Association for Computational Linguistics.

Zhiyuan Zeng, Keqing He, Yuanmeng Yan, Hong Xu, and Weiran Xu. 2021b. Adversarial self-supervised learning for out-of-domain detection. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 5631–5639, Online. Association for Computational Linguistics.

Zhiyuan Zeng, Hong Xu, Keqing He, Yuanmeng Yan, Sihong Liu, Zijun Liu, and Weiran Xu. 2021c. Adversarial generative distance-based classifier for robust out-of-domain detection. *ICASSP 2021 - 2021 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 7658–7662.

Hanlei Zhang, Hua Xu, and Ting-En Lin. 2021a. Deep open intent classification with adaptive decision boundary. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 35, pages 14374–14382.

Hanlei Zhang, Hua Xu, Ting-En Lin, and Rui Lv. 2021b. Discovering new intents with deep aligned clustering. In *AAAI*.

Yifan Zhang, Bryan Hooi, Lanqing Hong, and Jiashi Feng. 2021c. Test-agnostic long-tailed recognition by test-time aggregating diverse experts with self-supervision. *ArXiv preprint*, abs/2107.09249.

Yinhe Zheng, Guanyi Chen, and Minlie Huang. 2020. Out-of-domain detection for natural language understanding in dialog systems. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 28:1198–1209.

A Appendix

A.1 Original Intent Dataset Statistics

We show the detailed statistics of CLINC and BANKING datasets in Table 6. Banking is class-imbalanced, and the number of samples for each class is shown in Fig 8. The three GID datasets GID-SD GID-MD and GID-CD we constructed in this paper, all maintain the same train/dev/test split as the original dataset. Table 7 shows the number of intents divided into IND and OOD per domain for GID-MD-40%. Since CLINC and BANKING are open source datasets, there is no license problem.

A.2 GID-imbalanced

For our imbalanced dataset GID-imbalance, we show the distribution of the number of samples per OOD category under the influence of different imbalance ratio($\rho = 2, 3, 6$) in Figure9. The larger the imbalance ratio, the more significant the class imbalance degree of the corresponding imbalanced dataset.

A.3 Implementation Details

For a fair comparison of the various methods, we use the pre-trained BERT model (bert-base-uncased [9], with 12-layer transformer) as our network backbone, and add a pooling layer to get intent representation(dimension=768). Moreover, we freeze all but the last transformer layer parameters to achieve better performance with BERT backbone, and speed up the training procedure as suggested in (Zhang et al., 2021b). Firstly, we use labeled IND data to pretrain BERT model. For pipeline method(k-means and DeepAligned), we use the official implementation and hyperparameters offered by (Zhang et al., 2021b) to realize it, and the batch size is 512 and learning rate is 5e-5 for joint classification stage. For DeepAligned-Mix, the training batch size is 512 and the learning rate is 5e-5. For end-to-end method, IND head and OOD head are two symmetrical MLPs (input dimension is 768 and output dimension is the number of categories for IND/OOD), and we select $tanh$ as activation function as previous work. We use SGD with momentum as optimizer, with linear warm-up and cosine annealing (lr_{base} = 0.4, lr_{min} = 0.01), and weight decay 10^{-4}. The batch size is always set to 512 for all experiments. Notably, We use dropout (Gao et al., 2021) to construct augmented examples and the dropout value is fixed at

[9] https://github.com/google-research/bert

Dataset	Classes	Training	Validation	Test	Vocabulary	Length (max / mean)
CLINC	150	18,000	2,250	2,250	7,283	28 / 8.31
BANKING	77	9,003	1,000	3,080	5,028	79 / 11.91

Table 6: Statistics of CLINC and BANKING datasets.

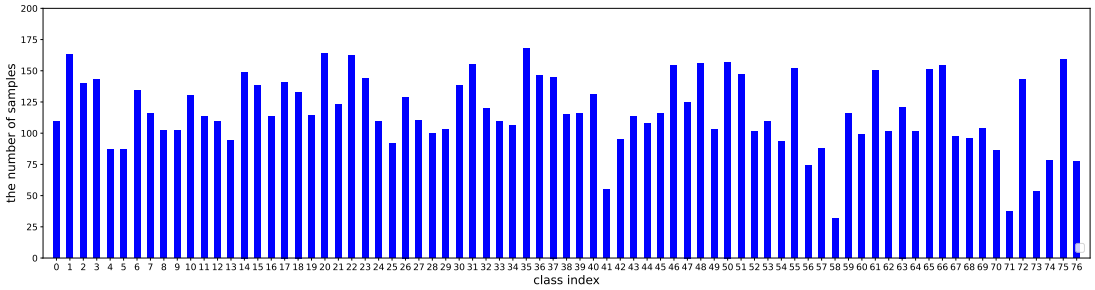

Figure 8: The number of samples for each class in Banking dataset.

Domains	IND intents	OOD intents
banking	10	5
credit_cards	8	7
kitchen_and_dining	9	6
home	6	9
work	10	5
utility	8	7
travel	9	6
auto_and_commute	10	5
small_talk	11	4
meta	9	6

Table 7: The number of intents divided into IND and OOD per domain for GID-MD-40%

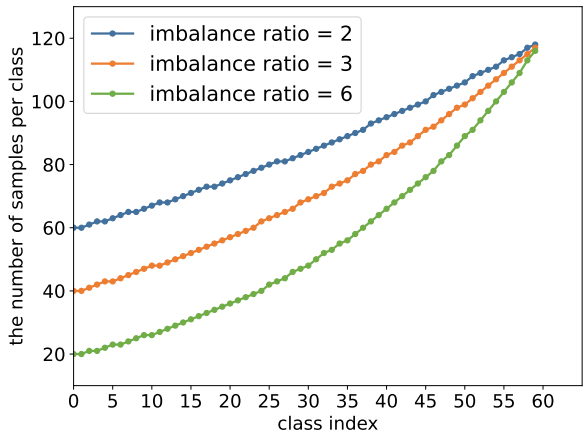

Figure 9: The distribution of the number of samples for GID-imbalance

0.5. For what concerns pseudo-labeling, we use the implementation of the Sinkhorn-Knopp algorithm provided by (Caron et al., 2020) and we inherit all the hyperparameters from (Caron et al., 2020), e.g. $n\_iter = 3$ and $\epsilon = 0.05$. We use the SC value of the validation data to select the best checkpoints. All experiments use a single Tesla T4 GPU(16 GB of memory). Table8 shows the comparison of the epoch and training time required for the convergence of the End-to-End method and DeepAligned. We can see that the E2E method takes fewer epochs to converge than the DeepAligned method. This is because the DeepAligned method first performs OOD clustering, and then uses the obtained OOD pseudo-labels and IND ground-truth labels for joint classification, which will lead to The OOD pseudo-labels have serious errors, and these label errors cannot be corrected in classification process, resulting in difficulty in model convergence. In addition, we can also see that the E2E method only increases the time required for each epoch by 1.8s compared to the classification stage of DeepAligned, which indicates the efficiency of the SK algorithm.

Method	training epoch	training time
End-to-End	51	30s/epoch
DeepAligned(two-stages)		
- clustering	67	27.6s/epoch
- classification	91	28.2s/epoch

Table 8: Comparison of training efficiency between pipeline and End-to-End methods. We use the same hardware.

A.4 Estimate K

Since we may not know the exact number of OOD clusters, we use the following K estimation method (Zhang et al., 2021b) to determine the number of

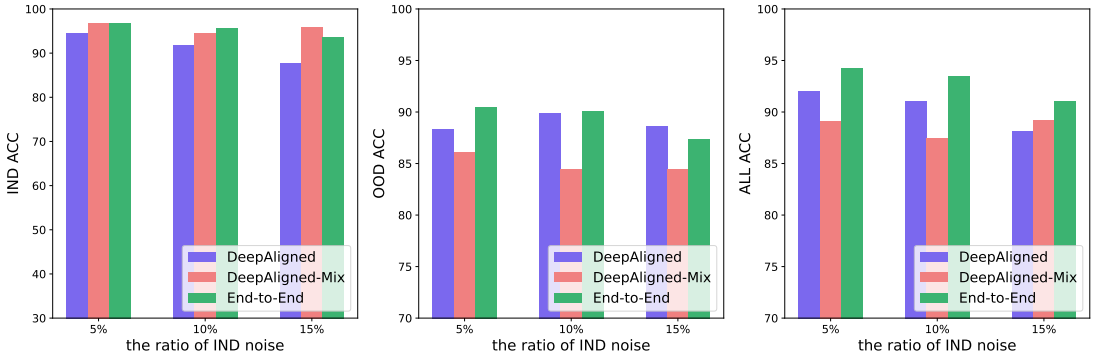

Figure 10: The impact of adding different ratios of IND noise samples to the OOD training data on the performance of each GID model.

clusters K before clustering. The method estimates K with the aid of the well-initialized intent features. We assign a big K' as the number of clusters at first. As a good feature initialization is helpful for partition-based methods (e.g., k-means), we use the well pre-trained model to extract intent features. Then, we perform k-means with the extracted features. We suppose that real clusters tend to be dense even with K', and the size of more confident clusters is larger than some threshold t. Therefore, we drop the low confidence cluster whose size is smaller than t, and calculate K with:

$$K = \sum_{i=1}^{K'} \delta\left(|S_i| >= t\right) \tag{5}$$

where $|S_i|$ is the size of the ith produced cluster, and $\delta(\cdot)$ is an indicator function. It outputs 1 if condition is satisfied, and outputs 0 if not. Notably, we assign the threshold t as the expected cluster mean size $\frac{N}{K'}$ in this formula.

A.5 Effect of IND Data

We analyze the impact of the number of samples per IND class on the performance of each model. Fig 11 shows the trend of model performance as the number of IND samples for each class decreases. Overall, the performance of our end-to-end method is much better than the baselines. Moreover, with the decrease of the amount of in-domain data, all methods show varying degrees of performance fluctuation. We observe the changes of IND F1 and OOD F1, and find IND F1 generally shows a downward trend, especially for DeepAligned-Mix. We believe that this is because the number of IND samples in each category is reduced, resulting in the biased joint classification of IND&OOD towards the OOD category. DeepAligned-Mix learns both

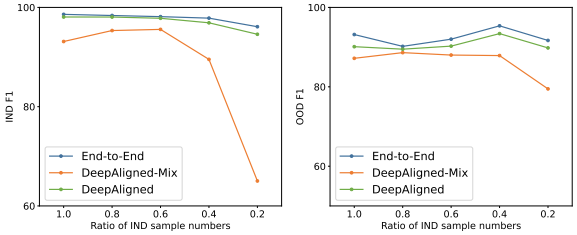

Figure 11: Effect of IND data for GID. The left subfig denotes IND F1 and the right subfig denotes OOD F1.

IND and OOD by clustering, which will lead to inaccurate pseudo-labels obtained by IND, further degrading the performance. As for OOD F1, due to the reduced number of IND samples, the model can learn less IND prior knowledge, thus affecting the performance of OOD. Therefore, GID in the small labeled IND scenario is also a challenge worthy of attention.

A.6 Silhouette Coefficient (SC)

Following Zhang et al. (2021b), we use the cluster validity index (CVI) to evaluate the quality of clusters obtained during each training epoch after clustering. Specifically, we adopt an unsupervised metric Silhouette Coefficient (Rousseeuw, 1987) for evaluation:

$$SC = \frac{1}{N} \sum_{i=1}^{N} \frac{b\left(\boldsymbol{I}_i\right) - a\left(\boldsymbol{I}_i\right)}{\max\left\{a\left(\boldsymbol{I}_i\right), b\left(\boldsymbol{I}_i\right)\right\}} \tag{6}$$

where $a\left(\boldsymbol{I}_i\right)$ is the average distance between $\boldsymbol{I}_i$ and all other samples in the i-th cluster, which indicates the intra-class compactness. $b\left(\boldsymbol{I}_i\right)$ is the smallest distance between $\boldsymbol{I}_i$ and all samples not in the i-th cluster, which indicates the inter-class separation. The range of SC is between -1 and 1, and the higher score means the better clustering results.

DIALMED: A Dataset for Dialogue-based Medication Recommendation

Zhenfeng He[1]*, **Yuqiang Han**[1]*, **Zhenqiu Ouyang**[3], **Wei Gao**[4], **Hongxu Chen**[5], **Guandong Xu**[5], **Jian Wu**[2]†

[1]College of Computer Science and Technology, Zhejiang University
[2]School of Public Health, Zhejiang University [3]Polytechnic Institute, Zhejiang University
[4]Ningbo Institute of Technology, Zhejiang University
[5]University of Technology Sydney
{hezf, hyq2015, oyzq, gw, wujian2000}@zju.edu.cn
{hongxu.chen, guandong.xu}@uts.edu.au

Abstract

Medication recommendation is a crucial task for intelligent healthcare systems. Previous studies mainly recommend medications with electronic health records (EHRs). However, some details of interactions between doctors and patients may be ignored or omitted in EHRs, which are essential for automatic medication recommendation. Therefore, we make the first attempt to recommend medications with the conversations between doctors and patients. In this work, we construct DIALMED, the first high-quality dataset for medical dialogue-based medication recommendation task. It contains 11,996 medical dialogues related to 16 common diseases from 3 departments and 70 corresponding common medications. Furthermore, we propose a **D**ialogue structure and **D**isease knowledge aware **N**etwork (**DDN**), where a QA Dialogue Graph mechanism is designed to model the dialogue structure and the knowledge graph is used to introduce external disease knowledge. The extensive experimental results demonstrate that the proposed method is a promising solution to recommend medications with medical dialogues. The dataset and code are available at https://github.com/f-window/DialMed.

1 Introduction

The outbreak of COVID-19 has challenged the healthcare systems and led to millions of patients facing delays in diagnosis and treatment. As an essential complement to the traditional face-to-face medicine, telemedicine relieved the therapeutic stress caused by the diversion of medical resources. According to the report of WeDoctor[1], an online health consultation platform in China, about 1.2 million patients conducted online medical consultations during the COVID-19 Pandemic.

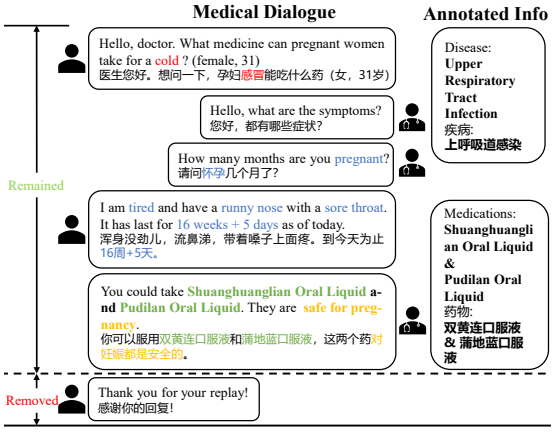

Figure 1: A typical medication consultation dialogue. Here, the disease is ***Upper Respiratory Tract Infection***, and the medication is ***Shuanghuanglian Oral Liquid*** and ***Pudilan Oral Liquid***.

Telemedicine can increase the availability of medical treatment, reduce healthcare costs, and improve the quality of care. Consequently, it has attracted increasing attention due to its vast application potential.

Our study found that around 31% of online consultations are about what medications the patients should take based on their current conditions[2]. Figure 1 demonstrates a typical medication consultation dialogue. The patient reported the health issues initially, with some personal information, such as gender and age. Then the doctor asked for further information (*e.g.*, symptoms and disease history) about the patient. Finally, the doctor provided medication advice based on the gathered information and clinical experience.

Existing studies on medication recommendation are primarily based on EHRs (Zhang et al., 2017; Shang et al., 2019b; An et al., 2021), accumulatively collected according to a diagnostic procedure in clinics. However, the doctors will omit some details of interactions with patients in EHRs,

*Both authors contributed equally to this research.
†Corresponding author.
[1]https://www.guahao.com/
[2]Refer to Appendix D.1 for details of statistic.

Proceedings of the 29th International Conference on Computational Linguistics, pages 721–733
October 12–17, 2022.

which are essential for the automatic medication recommendation. Compared to EHRs, medical dialogues retain original interactions between doctors and patients, containing more rich information. To this end, medical dialogue-based medication recommendation is a promising and challenging task.

Therefore, in this work, we study the new task, namely dialogue-based medication recommendation. Due to the lack of available datasets, we firstly construct a high-quality online medical dialogues dataset (DIALMED) for this task. It contains 11,996 consultation dialogues, 16 diseases from 3 different departments, and 70 related common medications.

Then, to further advance the research of this task, we propose a **D**ialogue structure and **D**isease knowledge aware **N**etwork (**DDN**). In DDN, for the input dialogue, we first utilize a pre-trained language model to extract the semantic information of each utterance. A mechanism named QA Dialogue Graph is designed to understand the questions&answers implied in utterances, and then we apply graph attention network on this QA graph to get the dialogue embedding. Meanwhile, for the input disease, we use its identity to query the entity in a knowledge graph CMeKG[3], and input the dialogue embedding to a graph attention network to get contextual disease embedding. The two embeddings are fused to make the medication prediction. Moreover, we conduct extensive experiments to show that the proposed method can effectively recommend medications with medical dialogues.

Our contributions can be summarized as follows:

- We construct the first high-quality human-annotated dialogue dataset for dialogue-based medication recommendation task.
- We propose a novel medication recommendation framework which models dialogue structure with QA Dialogue Graph and introduces external disease knowledge.
- We conduct extensive experiments to demonstrate DDN can extract the essential information to make medication recommendation effectively.

2 Related Work

Medication Recommendation. Existing medication recommendations are mainly based on EHRs. It could be categorized into instance-based and longitudinal-based recommendation methods (Shang et al., 2019b). Instance-based methods are based on the current health conditions extracted from recent visit (Zhang et al., 2017; Wang et al., 2019a). For example, (Zhang et al., 2017) proposed a multi-instance multi-label learning framework to predict medication combination based on patient's current diagnoses. Longitudinal-based methods leverage the temporal dependencies among clinical events (Choi et al., 2016; Le et al., 2018; Shang et al., 2019b,a; Wang, 2020; He et al., 2020; Wang et al., 2021; Yang et al., 2021). Among them, (Shang et al., 2019a) combined the power of graph neural networks and BERT for medication recommendation. (Yang et al., 2021) proposed a drug-drug interactions (DDI)-controllable drug recommendation model to leverage drugs' molecule structures and model DDIs explicitly.

Unlike the work mentioned above, dialogue-based medication recommendation task is more challenging in practice due to the noisy and sparse data. Because of the privacy issue, it is difficult to get historical dialogues of a patient on online consultation platforms. So we perform the medication recommendation solely based on the current medical dialogues.

Graph Neural Networks. Graph neural networks have attracted a lot of attention for processing data with graph structures in various domains (Zhou et al., 2020). For example, (Kipf and Welling, 2017) proposed the graph convolutional networks (GCN) . With integration of attention mechanisms, graph attention networks(GAT) (Veličković et al., 2018) has become one of the most popular methods in graph neural networks.

Recently, some works have applied GAT to the dialogue modeling. (Chen et al., 2020) used Graph attention and recurrent GAT to fully encode dialogue utterances, schema graphs, and previous dialogue states for dialogue state tracking. (Qin et al., 2020) proposed a co-interactive GAT layer to simultaneously solve both dialog act recognition and sentiment classification task. In this work, we utilize GAT to model the intra- and inter-speaker correlations to propagate semantic on the QA Dialogue Graph and extend the GAT on knowledge graph to introduce external knowledge.

[3] http://cmekg.pcl.ac.cn/

3 Corpus Description

In this section, we introduce the construction details and statistics of DIALMED, and its comparison with other studies.

3.1 Construction Details

Our dataset is collected from Chunyu-Doctor[4], which is a popular Chinese medical consultation website for doctors and patients. The conversations between doctors and patients contain rich but complex information, mainly related to the patients' current conditions. The diagnosed diseases and symptoms both are indispensable for accurate medication recommendation. Considering the complexity of the symptoms, we decide to utilize information from *explicit disease* and *implicit symptoms* in this paper. So we annotate the diagnosed diseases and recommended medications (replaced with a mask token to keep the original dialogue structure). For the example in Figure 1, we annotate the disease *Upper Respiratory Tract Infection*, and replace the medications *Shuanghuanglian Oral Liquid* and *Pudilan Oral Liquid* with special token *[MASK]*. Moreover, the future utterances after the point of recommendation are removed to make DIALMED more realistic, as the decision of doctors should not be influenced by future contexts.

The procedure of annotation consists of two parts, labeling and normalization of medications and diseases. First, we select 16 common diseases and the corresponding common medications from 3 departments (i.e., respiratory, gastroenterology, and dermatology) with the guidance of a doctor. These diseases have abundant medication consultations online. Then three annotators with relevant medical backgrounds are involved. Each dialogue is annotated by two annotators and will be further judged by another one if there is any inconsistency. The annotation consistency, i.e., the Cohen's kappa coefficient (Fleiss and Cohen, 1973) of the labelled dialogues is 88.4%. For the quality of dataset, conversations containing unsuitable medications for patients would be discarded.

Secondly, we normalize the medications since there are many generic names, trade names, or colloquial expressions for the same drug in dialogues. Specifically, different brands of the same drug are grouped into one cluster and normalized as a common name from DXY Drugs Database[5]. For example, **Omeprazole enteric-coated tablet** and **Omeprazole tablet** are normalized to **Omeprazole**. Similarly, we normalize the different names of diseases into ICD-10 standard names. The dialogues, hard to give diagnosed diseases or given diseases out of our scope, would be marked as a special placeholder, **None or Others**.

3.2 Dataset Statistics

Top of Table 2 summarizes the statistics of DIALMED. The scenario of dialogues in the dataset is similar to outpatient procedure, so the number of medicines per dialogue is relatively small. Then, the frequency of medications and diseases are shown in Figure 2(a) and Figure 2(b) respectively. The distributions of quantity demonstrate that DIALMED aligns with the real-world case.

Compared to the other medical dialogue datasets in Table 1, our dataset has three advantages: (1) DIALMED has the largest volume among the manual annotation datasets, as unlabeled datasets are mainly constructed for the task of dialogue generation. (2) Though the future contexts after recommendation are removed, the average number of dialogue turns in DIALMED still remains high compared to other datasets. It is mainly benefited from our evaluation for inclusion of short dialogues in DIALMED during the labeling process. (3) We carefully choose the fields suitable for medication recommendation and avoid coarsely expanding the scope of medical domains, which makes DIALMED have a higher quality.

The panoramas of medications & diseases' frequency could be found in Appendix D.2.

3.3 The comparison with other studies

To our best knowledge, DIALMED is the first dataset for the medication recommendation based on medical dialogues. It has the following differences with the existing work.

Dataset Medical dialogue has attracted increasing attention in recent years. Although there are medication mentions in many medical dialogue datasets, the distributions are fragmentary and the authors do not categorize and normalize these drug mentions which would lead to label explosion. For instance, medication mentions, **Omeprazole entericcoated tablet**, **Omeprazole tablet** and **Omeprazole**, which may occur in dialogues would be three classes without normalization. In fact, they are essentially equivalent in the eyes of doctors. By

[4] https://www.chunyuyisheng.com/
[5] http://drugs.dxy.cn/

Dataset	#Task	#Domain	#Disease	#Dialogue	#Avg. Turn	#Annotation
MZ(Wei et al., 2018)	Diagnosis	Pediatrics	4	710	-	Man.
DX(Xu et al., 2019)	Diagnosis	Pediatrics	5	527	5.34	Man.
CMDD(Lin et al., 2019)	Diagnosis	Pediatrics	4	2,067	42.09	Man.
SAT(Du et al., 2019)	Extraction	14	-	2,950	-	Man.
MIE(Zhang et al., 2020)	Extraction	Cardiology	6	1,120	16.19	Man.
MSL(Shi et al., 2020)	Extraction	Pediatrics	5	2,652	-	Man.
MedDG(Liu et al., 2020)	Extraction	Gastroenterology	12	17,864	21.60	Man.& Semi-Auto.
COVID-EN(Yang et al., 2020)	Generation	COVID-19	1	603	8.7	None
COVID-CN(Yang et al., 2020)	Generation	COVID-19	1	1088	2.0	None
MedDialog-EN(Zeng et al., 2020)	Generation	51	96	257,332	2	None
MedDialog-CN(Zeng et al., 2020)	Generation	29	172	3,407,494	3.3	None
Chunyu(Lin et al., 2021)	Generation	-	15	12,842	24.7	Rule
KaMed(Li et al., 2021)	Generation	100	-	63,754	11.62	None
ReMeDi(Yan et al., 2022)	Diag.&Ext.&Gene.	30	491	1,557	16.34	Man.
DIALMED(ours)	Medication	R&G&D	16	11,996	10.94	Man.

Table 1: Comparison between our dataset and other related medical dialogue datasets. Extraction, Generation and Medication mean information extraction, dialogue generation and medication recommendation separately. R&G&D, Man. and Semi-Auto are abbreviations of Respiratory&Gastroenterology&Dermatology, Manual and Semi-Automated respectively.

	#Dial.	#Dise.	#Med.	Avg.M	Avg.T	Max.T	Avg.U	Max.U
Resp.	4,859	4	45	2.06	10.76	52	18.18	374
Gastro.	3,818	9	39	1.88	13.05	58	16.70	463
Derma.	3,319	3	27	1.62	8.77	44	18.82	453
Total	11,996	16	70	1.88	10.94	58	17.76	463
Train.	9,605	16	70	1.88	10.95	58	17.74	463
Dev.	1,192	16	70	1.89	11.25	49	17.45	298
Test.	1,199	16	70	1.89	10.58	42	18.27	293

Table 2: Data statistics of DIALMED. M, T, and U represent medicine, dialogue turns, and utterance.

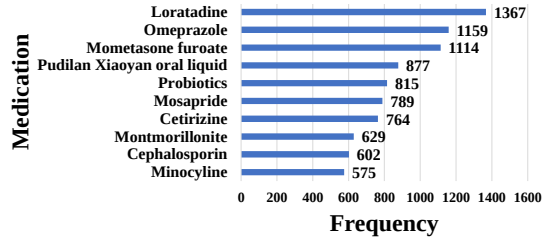

(a) The frequency of medications.

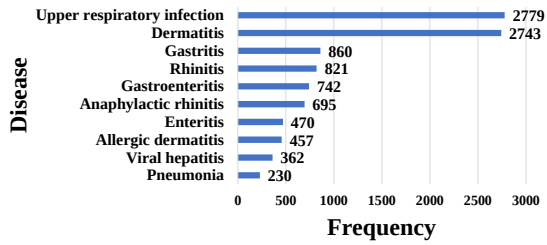

(b) The frequency of diseases.

Figure 2: The frequencies of medications and diseases. Top 10 are exhibited for the constraint of space.

contrast, we reduce the complexity caused by the doctors' preferences for different brands through categorization and normalization. DIALMED is developed for drug recommendation.

Task Drug recommendation is a sub-task of medical diagnosis. According to patients' questions, the objectives of current diagnosis systems are to generate the optimal clinical responses which may be intended as one of greeting, inquiry or diagnosis. Even if there contains drug mentions in responses, it is just one of the system's options. Drug recommendation is a key task and requires specialized dataset. DIALMED goes a step forward.

Scenario There are remarkable distinctions between DIALMED and MIMIC-III (Johnson et al., 2016), an EHR database which is relied on in current medication recommendation study. The scenario of the former is outpatient procedure while the data from the latter is generated from Intensive Care Units (**ICU**). In MIMIC-III, for example, the number of medications is 145, the average number of medications in each visit is 8.80, and the average number of diagnosis in each visit is 10.51. In contrast, the labels in medical dialogues are relatively sparse, leading to a more challenging task.

4 Our Approach

In this section, we first introduce the dialogue-based medication recommendation task, and then describe the proposed DDN in detail.

4.1 Problem Formulation

In the online medical dialogue setting, each dialogue consists of a sequence of utterances from the patient and the doctor. Formally, each dialogue can be represented as $\mathcal{D}_n = \{u_1, u_2, ..., u_{|\mathcal{D}_n|}\}$, where $n \in \{1, 2, ..., N\}$, N denotes the total number of dialogues in the dataset, and $|\mathcal{D}_n|$ represents the number of utterances in a dialogue $\mathcal{D}_n$. Each utterance can be represented as $u_i = \{w_i^1, ..., w_i^j, ..., w_i^{|u_i|}\}$, where w_i^j is the j-th word in u_i and $|u_i|$ denotes the number of words in u_i. We collect all the diseases and medications mentioned in the dataset to construct a disease corpus $\mathcal{S}$ and medication corpus $\mathcal{M}$. To avoid notation clutter, we hereinafter remove the subscript n as we only consider a single dialogue instance. Formally, given the consultation dialogue $\mathcal{D}$ and the diagnosed disease d as inputs, dialogue-based medication recommendation aims to recommend potential treatment medications $\mathbf{y}$ in $\mathcal{M}$, where $\mathbf{y} \in \{0, 1\}^{|\mathcal{M}|}$.

4.2 Model Overview

The proposed end-to-end framework is presented in Figure 3, consisting of two parts: (1) Dialogue Encoder, encoding the medical dialogues between patient and doctor by comprehensively capturing the semantic information and dialogue structure. (2) Disease Encoder, incorporating external medical knowledge based on the disease information from the dialogue and knowledge graph.

4.3 Dialogue Encoder

Dialogues contain two types of important information: (1) the rich semantic information, (2) strong structural correlations between utterances.

Utterance Encoding Pre-trained language models (*e.g.*, RoBERTa) are utilized to capture the semantic information in utterances. First, special tokens [CLS] (capturing utterance representation) and [SEP] (separating different utterances) are inserted at the beginning and end of each utterance token sequence u_i. Then the position embedding of each token in a utterance is calculated. In addition, two types of speaker embeddings (*i.e.*, *Doctor* and *Patient*) are proposed to make model aware of the speaker role of the utterance. The model takes the sum of three embeddings as input and outputs the representation of [CLS] as the utterance embedding $\mathbf{h}$. So a dialogue $\mathcal{D}$ can be represented as $\mathbf{h}_D = \{\mathbf{h}_1, \mathbf{h}_2, ..., \mathbf{h}_{|\mathcal{D}|}\}$.

QA Dialogue Graph In medical conversations, the interactions between doctors and patients tend to be in the form of questions and answers. For example, in Figure 3, the doctor asked two questions in u_2 and u_3, and the patient gave the answers in u_4. So it's important to capture the structure of QA pairs in conversation in order to understand the whole medical dialogue. We propose a new method to model the dialogue based on the observation that there is a high possibility of question-and-answer relations between adjacent utterances.

Specifically, we design a mechanism named QA Dialogue Graph, where each utterance is represented as a node in graph, and consecutive utterances spoken by the same speaker is represented as a block, e.g., u_2 and u_3 constitute a block with two nodes, and u_4 is another block with one node. Then the constructions of edges between nodes can be defined as follows:

- Within a block, each node connects with all other nodes in the block. This represents the intra-speaker correlation and ensures the information from the same speaker propagates among utterances within a local context.
- For two adjacent blocks, each node in a block connects with all nodes in the other block. This represents the inter-speaker correlation and ensures the information flow between doctors and patients within consecutive contexts.

An example of the adjacency matrix of the dialogue is shown in Figure 3. In general, when compared to previous works on dialogue modeling, QA Dialogue Graph has two advantages. Firstly, the construction of graphs does not require additional supervised information (Joshi et al., 2021; Feng et al., 2021). Secondly, our method comprehensively captures the structural and semantic information of QA pairs, which is key to understanding conversations (Qin et al., 2020; Shen et al., 2021b).

Dialogue Encoding GAT is employed to automatically aggregate semantic and structure features on QA Dialogue Graph. In particular, the l-th layer representation of a vertex can be computed as:

$$\mathbf{h}_i^{(l)} = \sigma(\sum_{j \in \mathcal{N}_i} \alpha_{ij} W_h \mathbf{h}_j^{(l-1)}) \quad (1)$$

where $\mathcal{N}_i$ is the first-order neighbors of vertex i, $W_h \in \mathbb{R}^{d_l \times d_{l-1}}$ is a trainable weight matrix, and σ is a nonlinear activation function. The weight

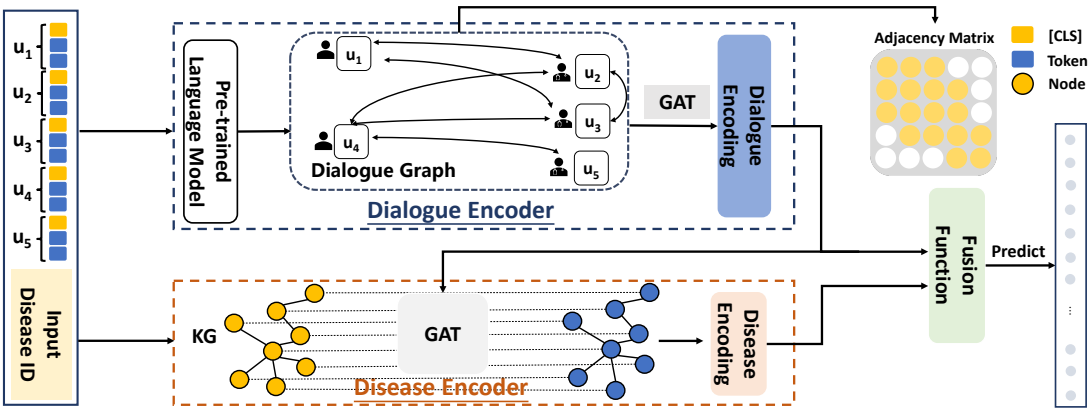

Figure 3: The framework of the proposed DDN for dialogue-based medication recommendation.

α_{ij} which determines the relatedness between two vertices can be calculated following (Veličković et al., 2018):

$$\alpha_{ij} = \frac{\exp\left(\sigma(\mathbf{a}^T W_h[\mathbf{h}_i \,||\, \mathbf{h}_j])\right)}{\sum_{k\in\mathcal{N}_i}\exp\left(\sigma(\mathbf{a}^T W_h[\mathbf{h}_i \,||\, \mathbf{h}_k])\right)} \tag{2}$$

where $\mathbf{a} \in \mathbb{R}^{2d_l}$ is a trainable weight matrix, and σ is the LeakyReLU activation function. Finally, we apply the attention pooling on nodes embedding to obtain the dialogue representation $\mathbf{h}_\mathcal{D}$, where $\mathbf{W}_a$ is a learnable parameter and $\mathbf{h}^{(l)}$ is the representation of utterances after l^{th} layer.

$$\widehat{\alpha} = softmax(\mathbf{W}_a\mathbf{h}^{(l)}) \tag{3}$$

$$\mathbf{h}_\mathcal{D} = \sum_i \widehat{\alpha}_i \mathbf{h}_i^{(l)} \tag{4}$$

4.4 Disease Encoder

Disease knowledge is crucial for delivering accurate medication recommendation. In this paper, we incorporate knowledge from CMeKG, a high-quality Chinese medical knowledge graph. TransR (Wang et al., 2019b) is utilized to get the initial entities embedding. Given a disease d, we first identify the corresponding entity in CMeKG, and then a KG subset with K hops starting from the disease entity is sampled randomly, finally the GAT network is used to get the disease embedding under the dialogue context.

Here, we fuse the entity, relation and dialogue information to get the attention scores:

$$\beta_{ij} = \frac{\exp\left(\sigma(\mathbf{a}^T[W[\mathbf{h}_i, \mathbf{h}_j] \,||\, W_r\mathbf{r}_\varphi \,||\, W_D\mathbf{h}_\mathcal{D}])\right)}{\sum_{j\in\mathcal{N}_i}\exp\left(\sigma(\mathbf{a}^T[W[\mathbf{h}_i, \mathbf{h}_j] \,||\, W_r\mathbf{r}_\varphi \,||\, W_D\mathbf{h}_\mathcal{D}])\right)} \tag{5}$$

where σ is the LeakyReLU function, $\mathbf{h}_i$, $\mathbf{h}_j$ and $\mathbf{r}_\varphi$ are the embeddings of node i, j and their relation separately. And W, W_r, and W_D are learnable weights to transform node, relation and dialogue embeddings, respectively. Then the l-th layer of disease embedding can be obtained as follows:

$$\mathbf{s}_i^{(l)} = \sigma\Big(\sum_{j\in\mathcal{N}_i} \beta_{ij} W_k \mathbf{h}_j^{(l-1)}\Big) \tag{6}$$

The contextual embedding of last layer is the disease d's representation, denoted by $\mathbf{s}_d$.

For dialogues with None or Others placeholder rather than a disease label, a learnable vector $\hat{\mathbf{s}}_d$ would be assigned to $\mathbf{s}_d$.

4.5 Model Inference and Optimization

The dialogue $\mathbf{h}_\mathcal{D}$ and disease $\mathbf{s}_d$ are fused by the *fusion function* to make prediction. In this work, we concatenate them and then fed it into decoder to make the medication prediction as follows:

$$\mathbf{y} = \sigma(W_o[\mathbf{h}_\mathcal{D}; \mathbf{s}_d] + \mathbf{b}_o) \tag{7}$$

where $W_o \in \mathbb{R}^{|\mathcal{M}|\times 2d}$ and $\mathbf{b}_o \in \mathbb{R}^{|\mathcal{M}|}$ are trainable weight matrices for the decoder, σ is the sigmoid activation function. Here, we reserve all the candidates whose probability is higher than the threshold of 0.5 as the recommended treatment medication combination.

Since medication combination recommendation is treated as a multi-label classification task (Shang et al., 2019b; Yang et al., 2021), we utilize the binary cross-entropy loss as the objective function, which can be formulated as:

$$\mathcal{L} = -\sum_{i=1}^{|\mathcal{D}|}\sum_{j=1}^{|\mathcal{M}|}\left(y_j^{(i)} \log \widehat{y}_j^{(i)} + (1-y_j^{(i)}) \log(1-\widehat{y}_j^{(i)})\right) \tag{8}$$

where $|\mathcal{D}|$ is the number of dialogues in the training set, $|\mathcal{M}|$ is the number of medications. $y_j^{(i)}$ is the ground truth label which equals 1 if medication

j is prescribed by the doctor in dialogue i, and 0 otherwise. $\widehat{y}_j^{(i)}$ is the predicted probability of recommending medication j.

5 Experiments

5.1 Experimental Setup

Dataset In our experiments, we divide the data into train/development/test dialogue sets as shown in Table 2. The average number of medications in each dialogue is approximately the same, as well as the the average length of utterances and dialogues, meaning the distribution of the data is relatively consistent among three sets.

Implementation Details The pretained model we use is Chinese RoBERTa-base model. The learning rate and the batch size are set as 2×10^{-5} and 8, respectively. Adam optimizer is utilized to optimize the model. All methods are implemented and trained using Pytorch on GeForce RTX 3090 GPUs. The results are the mean of five trainings.

Baselines Since there is no standard baselines for this task, we implement several methods of related tasks, including statistics-based (i.e., **TF-IDF (Salton and Buckley, 1988)**), RNN-based (i.e., **LSTM-*flat***, **LSTM-*hier***, **RETAIN** (Choi et al., 2016) and **DAG-ERC (Shen et al., 2021b)**), and transformer-based methods (i.e., **HiTANet** (Luo et al., 2020), **LSAN** (Ye et al., 2020)) and **DialogXL (Shen et al., 2021a)**. The RETAIN, HiTANet and LSAN are strong baselines for EHR-based medication recommendation or risk prediction. DAG-ERC and DialogXL are the SOTA methods at Emotion Recognition in Conversation (**ERC**). *Among them, LSTM-hier takes the dialogue structure into consideration, and LSAN and DialogXL are modified to incorporate disease knowledge*. Refer to Appendix B.1 for more details.

Evaluation Metrics We adopt two commonly used metrics, namely **Jaccard** and **F1** scores, to evaluate the model performance.

5.2 Main Results

Table 3 shows performances of all methods under the metric of Jaccard and F1 on four datasets. The results clearly indicate that DDN has achieved the best performances among all baselines. Particularly, DDN improves 6.35%, 5.14%, 3.95%, and 8.31% compared with the second best method (i.e., **DialogXL**) at Jaccard, respectively. Further,

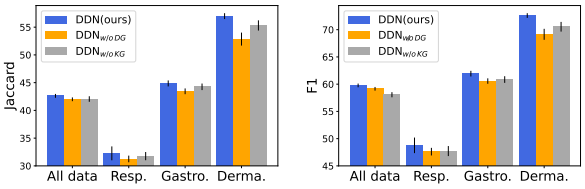

(a) Jaccard on four datasets (b) F1 on four datasets

Figure 4: Performance comparison of DDN and its variants.

RETAIN and LSTM-*hier* outperform LSTM-*flat*, demonstrating the dialogue structure is important for the dialogue understanding. And LSAN, DialogXL outperforms HiTANet, indicating that disease knowledge is also essential for the dialogue modeling. Our well-designed model DDN considers both of the above and achieves the best performance. In addition, it is worth noting that the performance varies over three departments, which may attribute to the considerable difference of medication and disease frequencies between different departments.

5.3 Ablation Study

Figure 4 summarizes the contributions of QA Dialogue Graph and disease knowledge of our model. We notice that by removing the QA Dialogue Graph, the variant DDN$_{w/o\ DG}$ shows considerable performance decrease at both Jaccard and F1 compared with DDN, especially on three departments datasets. It demonstrates that dialogue graph structure is important for the medical information extraction in dialogue-based medication recommendation task. Similarly, by removing the Knowledge Graph module, DDN$_{w/o\ KG}$ also shows similar performance decrease trends, indicating that disease knowledge can improve the medication recommendation performance. This is reasonable and accords with the actual medication consultation situations.

5.4 Task Feasibility Analysis

To prove the feasibility of dialogue-based medication recommendation, we provide incomplete discourses to DDN during the inference process to explore whether the dialogue can provide necessary medical information. Figure 5 shows the model performances under different portions of discourses. We can see that with the increasing of dialogue discourse percentage, the performance gets better, especially within the first 20% and the last 20%.

		All Data		Respiratory		Gastroenterology		Dermatology	
Type of Model	Model	Jaccard	F1	Jaccard	F1	Jaccard	F1	Jaccard	F1
Statistics	TF-IDF(Salton and Buckley, 1988)	21.25±0.41	35.05±0.56	16.06±0.44	27.68±0.66	23.85±0.40	38.52±0.52	28.84±0.14	44.77±0.17
RNN-Based	LSTM-*flat*	27.50±1.09	42.54±1.22	18.07±0.44	30.18±0.64	31.31±1.33	47.18±1.59	32.69±1.71	48.55±1.18
	LSTM-*hier*	30.20±0.47	46.39±0.56	22.86±0.42	37.21±0.56	32.90±0.93	49.51±1.05	36.00±0.50	52.94±0.54
	RETAIN(Choi et al., 2016)	31.16±0.82	42.16±0.99	21.13±0.64	30.49±0.96	36.70±0.86	48.54±0.73	43.19±1.06	54.14±1.20
	DAG-ERC(Shen et al., 2021b)	29.08±0.56	44.05±0.70	23.74±0.76	35.71±1.02	36.16±0.46	53.80±0.52	31.18±1.05	47.52±1.23
Transformer	HiTANet(Luo et al., 2020)	30.75±0.69	44.57±0.83	22.01±1.04	33.62±1.44	33.95±1.26	48.39±1.26	39.17±1.93	53.41±2.21
	LSAN(Ye et al., 2020)	34.33±0.58	46.14±0.45	26.11±1.06	38.89±1.01	39.28±0.22	52.49±0.62	50.29±1.24	57.90±1.09
	DialogXL(Shen et al., 2021a)	36.27±0.34	53.23±0.40	27.12±0.24	42.67±0.36	40.91±0.14	58.06±0.15	48.68±0.81	65.48±0.66
	DDN(Ours)	**42.62±0.35**	**59.77±0.34**	**32.26±1.25**	**48.77±1.43**	**44.86±0.54**	**61.93±0.52**	**56.99±0.53**	**72.60±0.43**

Table 3: Performance (%) comparison of DDN with baseline methods over the overall and three departments datasets. The best result in each column is highlighted in boldface. The performance gain of our method over all baselines is statistically significant with $p < 0.05$ under t-test.

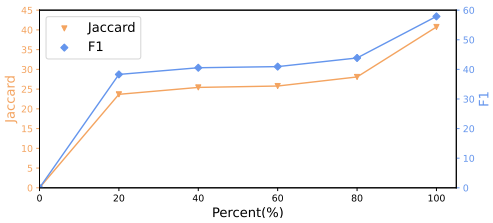

Figure 5: Average Jaccard scores on different percents(%) of dialogue discourse. In this setting, we choose dialogues with more than four turns in test set.

This may be because that the first and last parts of dialogue contain much patient complaints and symptoms that are closely related to the medications. The results demonstrate that recommending medication based on medical dialogues is feasible.

5.5 Error Analysis

Although we have elaborately designed a model for the task, the results are not so well satisfactory. So we make detailed analysis of the error cases in the test set. Table 4 summarizes the statistics of our defined five type of errors. We can see that (1) 86.38% of the cases (#3, #4, #5) predict wrong medications, which is mainly caused by DDN failing to distinguish the medications with similar effect. (2) 7.20% of the cases predict none labels, which can be attributed to that these dialogues provide a little disease-related information.

No.	Type of error	# Cases
#1	$P \subseteq \emptyset$	65(7.20%)
#2	$P \subset T$ & $P \not\subseteq \emptyset$	58(6.42%)
#3	$T \subset P$	182(20.16%)
#4	$T \not\subseteq P$ & $P \not\subseteq T$ & $P \cap T \not\subseteq \emptyset$	299(33.11%)
#5	$T \not\subseteq P$ & $P \not\subseteq T$ & $P \cap T \subseteq \emptyset$	299(33.11%)
Total	-	903

Table 4: The statistics of errors on test set. P and T are the predicted and golden label set, respectively.

Sample	Medications
Disease: Duodenitis **P :** Hello, doctor. In March this year, I had a duodenal ulcer, bleeding, and was hospitalized. Stomach rises a bit uncomfortable and bloating in the night a week recently. Is it recrudescent? **D :** Duodenal ulcers are indeed prone to recurrence or inflammation. **P :** Can you prescribe some medicine for me? I don't have time to go to the hospital right now. **D :** Besides what you said, do you have any other complaints? Like acid reflux, heartburn. **P :** No. What does heartburn mean? I don't have this feeling at ordinary times. At present, I wake up uncomfortably in some nights. **P :** Almost no symptoms during the day. **D :** I suggest you take [MASK], [MASK], [MASK].	**TF-IDF:** Omeprazole, Digestive enzymes, 2 Missed **LSTM-*flat*:** Omeprazole, 2 Missed **LSTM-*hier*:** Omeprazole, Mosapride, Digestive enzymes, 1 Missed **RETAIN:** Mosapride, 2 Missed **DAG-ERC:** Omeprazole, 2 Missed **HiTANet:** Mosapride, 2 Missed **LSAN:** Omeprazole, 2 Missed **DialogXL:** Omeprazole, 2 Missed **DDN(Ours):** Omeprazole, Mosapride, Glutamine

Figure 6: The sample is extracted from the DIALMED test set. Golden labels of this case are Omeprazole, Mosapride and Glutamine. The "Missed" means the medication is in golden labels but not be predicted, and the underlined drugs in red represent the predicted medications that are not in ground truth.

5.6 Case Study

We further provide a case study to illustrate the superiority of DDN. Figure 6 shows the medical dialogue and the medications recommended by all baselines and our method. The baselines either miss some medications, e.g., LSTM-*flat*, RETAIN, HiTANet, LSAN, or give the wrong drugs, e.g., TF-IDF, LSTM-*hier*. DDN takes full account of *Duodenitis*-related information from the dialogue (e.g., the symptoms in chief complaint and past medical history) and the external knowledge graph. It recommends *Omeprazole* (inhibiting gastric acid secretion) and *Mosapride* (promoting gastric dynamics), as well as *Glutamine* which is omitted by all baselines.

6 Conclusions

In this paper, we studied a new task, namely dialogue-based medication recommendation. First, we presented the first high-quality medical dialogue dataset DIALMED for this task. And then we implemented several baselines, as well as designed a dialogue structure and external disease knowledge aware model. Experimental results show that medication recommendation quality can be enhanced with the help of dialogue structure and external disease knowledge.

Ethical considerations

Data in DIALMED is publicly collected from Chunyuyisheng, and personal information (e.g., usernames) is preprocessed. The annotating process is as described in Section 3. Furthermore, to ensure the quality of dataset, we paid the annotators 1 yuan ($0.16 USD) per label. The applications of machine learning in medical treatment would inevitably raise ethical issues. But the research on AI medicine should not be stopped by this, since the purpose of such research is how to make machines better serve human beings. We have seen many advanced achievements (Lin et al., 2021; Li et al., 2021; Zhang et al., 2020; Liu et al., 2020; Lin et al., 2019; Xu et al., 2019; Wei et al., 2018) in this field. For this study, the ethical issue is that there may cause bad cases in practical application. However, individual errors could be reduced by making doctors responsible for decisions while machines are used as assistants.

Acknowledgements

This research was partially supported by National Key R&D Program of China under grant No. 2018AAA0102102, National Natural Science Foundation of China under grants No. 62176231 and 62106218, Zhejiang public welfare technology research project under grant No. LGF20F020013. The authors would thank Ruochen Yan for her help in data processing.

References

Yang An, Liang Zhang, Haoyu Yang, Leilei Sun, Bo Jin, Chuanren Liu, Ruiyun Yu, and Xiaopeng Wei. 2021. Prediction of treatment medicines with dual adaptive sequential networks. *IEEE Transactions on Knowledge and Data Engineering*.

Lu Chen, Boer Lv, Chi Wang, Su Zhu, Bowen Tan, and Kai Yu. 2020. Schema-guided multi-domain dialogue state tracking with graph attention neural networks. In *AAAI*, volume 34, pages 7521–7528.

Edward Choi, Mohammad Taha Bahadori, Joshua A Kulas, Andy Schuetz, Walter F Stewart, and Jimeng Sun. 2016. Retain: An interpretable predictive model for healthcare using reverse time attention mechanism. In *NIPS*.

Nan Du, Kai Chen, Anjuli Kannan, Linh Tran, Yuhui Chen, and Izhak Shafran. 2019. Extracting symptoms and their status from clinical conversations. *arXiv preprint arXiv:1906.02239*.

Xiachong Feng, Xiaocheng Feng, Bing Qin, and Xinwei Geng. 2021. Dialogue discourse-aware graph model and data augmentation for meeting summarization. *IJCAI*, pages 3808–3814.

Joseph L Fleiss and Jacob Cohen. 1973. The equivalence of weighted kappa and the intraclass correlation coefficient as measures of reliability. *Educational and psychological measurement*, 33(3):613–619.

Yong He, Cheng Wang, Nan Li, and Zhenyu Zeng. 2020. Attention and memory-augmented networks for dual-view sequential learning. In *SIGKDD*, pages 125–134.

Alistair EW Johnson, Tom J Pollard, Lu Shen, H Lehman Li-Wei, Mengling Feng, Mohammad Ghassemi, Benjamin Moody, Peter Szolovits, Leo Anthony Celi, and Roger G Mark. 2016. Mimic-iii, a freely accessible critical care database. *Scientific data*, 3(1):1–9.

Rishabh Joshi, Vidhisha Balachandran, Shikhar Vashishth, Alan Black, and Yulia Tsvetkov. 2021. Dialograph: Incorporating interpretable strategy-graph networks into negotiation dialogues. *arXiv preprint arXiv:2106.00920*.

Thomas N Kipf and Max Welling. 2017. Semi-supervised classification with graph convolutional networks. In *ICLR*.

Hung Le, Truyen Tran, and Svetha Venkatesh. 2018. Dual memory neural computer for asynchronous two-view sequential learning. In *SIGKDD*, pages 1637–1645.

Dongdong Li, Zhaochun Ren, Pengjie Ren, Zhumin Chen, Miao Fan, Jun Ma, and Maarten de Rijke. 2021. Semi-supervised variational reasoning for medical dialogue generation. In *SIGIR*, page 544–554.

Shuai Lin, Pan Zhou, Xiaodan Liang, Jianheng Tang, Ruihui Zhao, Ziliang Chen, and Liang Lin. 2021. Graph-evolving meta-learning for low-resource medical dialogue generation. In *AAAI*, volume 35, pages 13362–13370.

Xinzhu Lin, Xiahui He, Qin Chen, Huaixiao Tou, Zhongyu Wei, and Ting Chen. 2019. Enhancing dialogue symptom diagnosis with global attention and symptom graph. In *EMNLP-IJCNLP*, pages 5036–5045.

Wenge Liu, Jianheng Tang, Jinghui Qin, Lin Xu, Zhen Li, and Xiaodan Liang. 2020. Meddg: A large-scale medical consultation dataset for building medical dialogue system. *arXiv preprint arXiv:2010.07497*.

Junyu Luo, Muchao Ye, Cao Xiao, and Fenglong Ma. 2020. Hitanet: Hierarchical time-aware attention networks for risk prediction on electronic health records. In *SIGKDD*.

Libo Qin, Zhouyang Li, Wanxiang Che, Minheng Ni, and Ting Liu. 2020. Co-gat: A co-interactive graph attention network for joint dialog act recognition and sentiment classification. *arXiv preprint arXiv:2012.13260*.

Gerard Salton and Christopher Buckley. 1988. Term-weighting approaches in automatic text retrieval. *Information processing & management*, 24(5):513–523.

Junyuan Shang, Tengfei Ma, Cao Xiao, and Jimeng Sun. 2019a. Pre-training of graph augmented transformers for medication recommendation.

Junyuan Shang, Cao Xiao, Tengfei Ma, Hongyan Li, and Jimeng Sun. 2019b. Gamenet: Graph augmented memory networks for recommending medication combination. In *AAAI*, volume 33, pages 1126–1133.

Weizhou Shen, Junqing Chen, Xiaojun Quan, and Zhixian Xie. 2021a. Dialogxl: All-in-one xlnet for multi-party conversation emotion recognition. In *AAAI*, volume 35, pages 13789–13797.

Weizhou Shen, Siyue Wu, Yunyi Yang, and Xiaojun Quan. 2021b. Directed acyclic graph network for conversational emotion recognition. *arXiv preprint arXiv:2105.12907*.

Xiaoming Shi, Haifeng Hu, Wanxiang Che, Zhongqian Sun, Ting Liu, and Junzhou Huang. 2020. Understanding medical conversations with scattered keyword attention and weak supervision from responses. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pages 8838–8845.

Petar Veličković, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Lio, and Yoshua Bengio. 2018. Graph attention networks. In *ICLR*.

Shanshan Wang, Pengjie Ren, Zhumin Chen, Zhaochun Ren, Jun Ma, and Maarten de Rijke. 2019a. Order-free medicine combination prediction with graph convolutional reinforcement learning. In *CIKM*, pages 1623–1632.

Shuai Wang. 2020. Seqmed: Recommending medication combination with sequence generative adversarial nets. In *BIBM*, pages 2664–2671. IEEE.

Xiang Wang, Xiangnan He, Yixin Cao, Meng Liu, and Tat-Seng Chua. 2019b. Kgat: Knowledge graph attention network for recommendation. In *SIGKDD*, pages 950–958.

Yanda Wang, Weitong Chen, Dechang Pi, and Lin Yue. 2021. Adversarially regularized medication recommendation model with multi-hop memory network. *Knowledge and Information Systems*, 63(1):125–142.

Zhongyu Wei, Qianlong Liu, Baolin Peng, Huaixiao Tou, Ting Chen, Xuan-Jing Huang, Kam-Fai Wong, and Xiang Dai. 2018. Task-oriented dialogue system for automatic diagnosis. In *ACL*, pages 201–207.

Lin Xu, Qixian Zhou, Ke Gong, Xiaodan Liang, Jianheng Tang, and Liang Lin. 2019. End-to-end knowledge-routed relational dialogue system for automatic diagnosis. In *AAAI*, volume 33, pages 7346–7353.

Guojun Yan, Jiahuan Pei, Pengjie Ren, Zhaochun Ren, Xin Xin, Huasheng Liang, Maarten de Rijke, and Zhumin Chen. 2022. Remedi: Resources for multi-domain, multi-service, medical dialogues.

Chaoqi Yang, Cao Xiao, Fenglong Ma, Lucas Glass, and Jimeng Sun. 2021. Safedrug: Dual molecular graph encoders for safe drug recommendations. In *IJCAI*.

Wenmian Yang, Guangtao Zeng, Bowen Tan, Zeqian Ju, Subrato Chakravorty, Xuehai He, Shu Chen, Xingyi Yang, Qingyang Wu, Zhou Yu, et al. 2020. On the generation of medical dialogues for covid-19. *arXiv preprint arXiv:2005.05442*.

Muchao Ye, Junyu Luo, Cao Xiao, and Fenglong Ma. 2020. Lsan: Modeling long-term dependencies and short-term correlations with hierarchical attention for risk prediction. In *CIKM*, pages 1753–1762.

Guangtao Zeng, Wenmian Yang, Zeqian Ju, Yue Yang, Sicheng Wang, Ruisi Zhang, Meng Zhou, Jiaqi Zeng, Xiangyu Dong, Ruoyu Zhang, et al. 2020. Meddialog: Large-scale medical dialogue dataset. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*.

Yuanzhe Zhang, Zhongtao Jiang, Tao Zhang, Shiwan Liu, Jiarun Cao, Kang Liu, Shengping Liu, and Jun Zhao. 2020. Mie: A medical information extractor towards medical dialogues. In *ACL*, pages 6460–6469.

Yutao Zhang, Robert Chen, Jie Tang, Walter F Stewart, and Jimeng Sun. 2017. Leap: learning to prescribe effective and safe treatment combinations for multimorbidity. In *SIGKDD*, pages 1315–1324.

Jie Zhou, Ganqu Cui, Shengding Hu, Zhengyan Zhang, Cheng Yang, Zhiyuan Liu, Lifeng Wang, Changcheng Li, and Maosong Sun. 2020. Graph neural networks: A review of methods and applications. *AI Open*, 1:57–81.

A Corpus

A.1 Details of corpus construction

First of all, diseases and related medications were identified in a dialogue. Secondly, we selected and annotated those dialogues containing drugs in our medication list. To speed up tagging process, we built an annotation tool based on this task. For each raw medical dialogue, the annotators need to annotate the disease of patients and medications recommended by doctors. We believe that the context after the doctor recommending the drug is not meaningful for drug inference. Due to the emergence of new medications in the labeling process and existence of ambiguity on recommendation, two additional annotation processes were carried out. Next we will focus on the processing of diseases and medications.

Disease Processing. With the guidance of a doctor, we select 16 diseases from 3 departments (i.e., respiratory, gastroenterology and dermatology) with following reasons: (1) they are common diseases and research on them have more practical value. (2) they could be consulted online and there are abundant medication consultations. As described by Section **Corpus Description**, we normalize the diseases to improve the quality of DIALMED, e.g., **chronic gastritis** and **acute gastritis** are mapped to **gastritis**. The dialogues without explicit disease information or diseases in our scope were marked as **None or Others**. We mark one disease according to the chief complaint of patients who have more than one disease, because patients have only one complaint in most diagnostic scenarios.

Medication Processing. As for medications, the ones we choose are commonly prescribed by doctors. Considering the differences between traditional Chinese medicines and Western medicine, both are included to achieve complementary advantages. Since there are many generic names, trade names and colloquial expressions for the same drug in conversations, it is significant to normalize the drug to a single label. For example, **Omeprazole enteric-coated tablet** and **Omeprazole enteric-coated capsule** could be mapped to **Omeprazole**. For compound medicines, we combine drugs that have the same ingredients into one, e.g., Tylenol represents all medicines that contain acetaminophen, pseudoephedrine hydrochloride, dextromethorphan hydrobromide and chlorpheniramine maleate. Due to space constraints, more normalization of diseases and medications could be found in our repository[6].

B Experiments

B.1 Baselines

- **TF-IDF**. This is a traditional bag-of-word model for text classification. We view each dialogue as text and the corresponding medication as label, and train a classification model based on TF-IDF features of words.
- **LSTM-*flat***. This is a LSTM-based method. It concatenates all the sentences in a dialogue as a long sentence and feeds the long sentence into the BiLSTM to get the dialogue embedding for medication prediction.
- **LSTM-*hier***. This is also a LSTM-based method. Different from LSTM-*flat*, it uses a hierarchical BiLSTM where each word in an utterance are fed into BiLSTM to get the utterance embedding and then the utterances are fed into another BiLSTM to get the final dialogue embedding. It captures both word-level and utterance-level dependencies.
- **RETAIN**. This is a RNN-based EHR medication recommendation method using on a two-level neural attention network that detects influential past visits. In the current scenario, it is used to model the dialogues.
- **DAG-ERC**. This method designed a directed acyclic neural network to model the information flow between long-distance conversation background and nearby context. Following the implementation in (Shen et al., 2021b), the features of utterances extracted from fine-tuning RoBERTa are inputted in model while the model structure is RNN based, so DAG-ERC is regraded as a RNN-based model.
- **HiTANet**. This is a Transformer-based risk prediction approach on EHR, which model time information in local and global stages. We transform this method to model the hidden temporal information in medical dialogues.
- **LSAN**. This is also a Transformer-based risk prediction approach, to model the hierarchical structure of EHR data. We modified this

[6]https://github.com/Hhhhhhhzf

method to model the hierarchical structure in medical dialogues and add disease module of DDN to encoder the external knowledge.

- **DialogXL**. This method improves XLNet with enhanced memory and dialog-aware self-attention. We modify the softmax layer to sigmoid layer in this model to fit the multi-label task in medication recommendation and add the disease module of DDN.

- **DDN**. This is our proposed model. It utilizes the dialogue structure and external disease knowledge to enhance the dialogue-based medication recommendation performance.

B.2 Evaluation Metrics

$$\text{Jaccard} = \frac{1}{|D|}\sum_{k=1}^{|D|}\frac{|Y^{(k)} \cap \hat{Y}^{(k)}|}{|Y^{(k)} \cup \hat{Y}^{(k)}|} \tag{9}$$

$$\text{F1} = \frac{1}{|D|}\sum_{k=1}^{|D|}\frac{2 \cdot \text{P}^{(k)} \cdot \text{R}^{(k)}}{\text{P}^{(k)} + \text{R}^{(k)}} \tag{10}$$

where $|D|$ is the number of dialogues in the test set. $Y^{(k)}$ represents the ground truth medication set of the kth dialogue, and $\hat{Y}^{(k)}$ represents the predicted medication set of the kth dialogue by the model. $\text{P}^{(k)}$, $\text{R}^{(k)}$ represents the Precision and Recall of the kth dialogue, respectively.

B.3 Additional Experiment on DDI

Medication combination recommendation would trigger the Drug-Drug Interaction (**DDI**) inevitably, which might lead to adverse outcomes. To this end, we explore the DDI in DIALMED. And we follow the previous work (Shang et al., 2019b) to give the DDI rate definition (smaller value means better).

$$DDIRate = \frac{\sum_{k}^{N}\sum_{i,j}|\{(c_i, c_j) \in \hat{Y}^{(k)}|(c_i, c_j) \in \mathcal{E}_d\}|}{\sum_{k}^{N}\sum_{i,j}1} \tag{11}$$

where the set will count each medication pair (c_i, c_j) in recommendation set $\hat{Y}$ if the pair belongs to edge set $\mathcal{E}_d$ of the DDI graph. Here N is the size of test dataset. In addition, DDI relationships among medications in DIALMED are collected from YAOZH [7], a medical data retrieval system.

The evaluation results are shown in Table 5. We could find that ground truth DDI rate is very small (compared to the 8.08% in MIMIC-III (Yang et al., 2021)), which may lead to the low rate on models. In view of this situation, we think it is no need for additional efforts to control the DDI rate at the current stage. Considering for the future research, we open source our DDI relationship graph in our repository.

	DDI Rate			
Model	All Data	Respiratory	Gastroenterology	Dermatology
G.T.	1.12	0.78	2.06	0.74
TF-IDF	1.10	0.46	2.01	0.51
LSTM-*flat*	0.58	1.36	0.93	0.00
LSTM-*hier*	1.02	0.11	0.91	0.65
RETAIN	1.92	1.12	1.89	0.00
DAG-ERC	0.81	1.01	1.53	0.48
HiTANet	0.45	1.49	1.09	0.50
LSAN	1.57	0.00	1.62	0.48
DialogXL	1.34	1.09	1.59	0.40
DDN	1.90	0.20	1.54	0.47

Table 5: DDI Rate (%) comparison on DIALMED. G.T. represents the Ground Truth.

C Task

C.1 Medical Utility

Medical treatment includes a number of steps: registration, examination, image reading, report interpretation, diagnosis, prescription and so on. AI medicine could help optimize resource allocation and improve efficiency in all aspects of health care. To this end, there are two kinds of computer aided diagnosis system, image diagnosis and text diagnosis. Due to the higher threshold of diagnosis, current researches are more inclined to image analysis, and there is still a lot of room for development in text diagnosis. Conversations in outpatient clinics are not reserved and involved many severe data privacy implications, leading to dialogue-based drug recommendation mainly oriented to telemedicine. The medical dialogue system, as a assistant of doctors, could give auxiliary medication suggestions based on the contexts when doctors and patients are communicating with each other.

D Statistics

D.1 Ratio of consulting for medications

The ratio of the patients to consult for medications is calculated with regular expressions. In the first place, 10,0000 different medical conversations from our dialogue corpus based on random sampling are fetched. For every dialogue, we apply the regular expression (e.g., "[Ww]hat (medication|drug|medicine) should I (take|eat)") on the utterances spoken by the patient and assume

[7] https://db.yaozh.com/interaction

that it is a case of consulting for drugs if the regular expression matches. The regular expressions are collected based on our observation and understanding of data. More regular expressions could be found in our repository.

D.2 Complete Corpus Statistics

The frequency of all diseases and medications is shown in Figure 7 & 8.

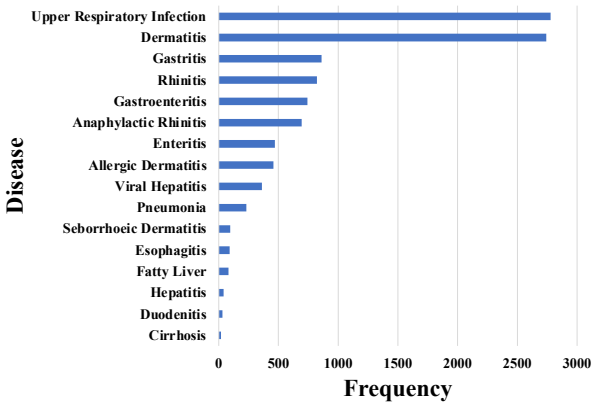

Figure 7: The frequency of all diseases.

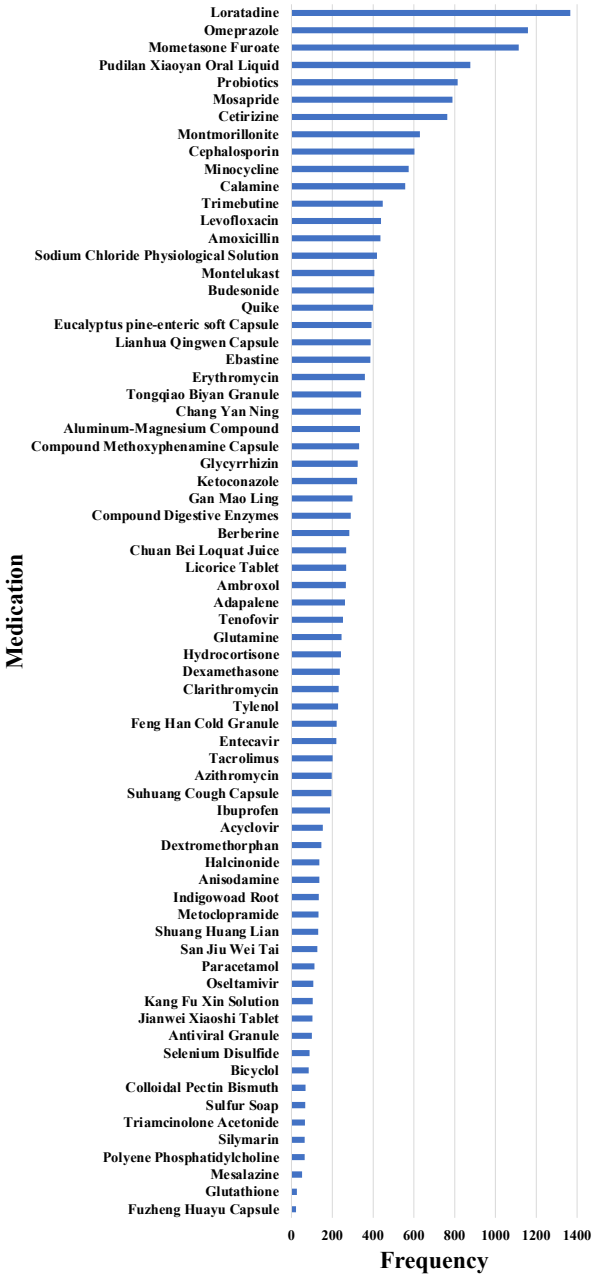

Figure 8: The frequency of all medications. The names are translated from Chinese.

Speaker Clustering in Textual Dialogue with Pairwise Utterance Relation and Cross-corpus Dialogue Act Supervision

Zhihua Su[1] and **Qiang Zhou**[1,2,*]
[1]State Key Laboratory of Intelligent Technology and Systems,
Department of Computer Science and Technology, Tsinghua University
[2]Beijing National Research Center for Information Science and Technology, Tsinghua University
suzh20@mails.tsinghua.edu.cn
zq-lxd@mail.tsinghua.edu.cn

Abstract

We propose a speaker clustering model for textual dialogues, which groups the utterances of a multi-party dialogue without speaker annotations, so that the actual speakers are identical inside each cluster. We find that, without knowing the speakers, the interactions between utterances are still implied in the text, which suggest the relations between speakers. In this work, we model the semantic content of utterance with a pre-trained language model, and the relations between speakers with an utterance-level pairwise matrix. The semantic content representation can be further instructed by cross-corpus dialogue act modeling. The speaker labels are finally generated by spectral clustering. Experiments show that our model outperforms the sequence classification baseline, and benefits from the auxiliary dialogue act classification task. We also discuss the detail of determining the number of speakers (clusters), eliminating the interference caused by semantic similarity, and the impact of utterance distance.

1 Introduction

Processing dialogues is a classical linguistic task. With the development of pre-trained language models in recent years, studies on dialogues have made great progress (Zhang et al., 2020; Roller et al., 2021; Adiwardana et al., 2020). In general, these training processes, especially pre-training, need a large amount of data. Meanwhile, most of dialogue models are designed to input speaker information, for example, applying trainable speaker embeddings, or just assuming the dialogue is composed of two speakers involved turn by turn, to introduce dialogue structure information into the models. But for common researchers, dialogue data is hard to collect. Datasets like subtitles (Lison et al., 2018) contain a lot of dialogue data of daily communication, but lack of speaker annotation. Some researches in related fields, such as conference transcription (Raj et al., 2021; Fu et al., 2021; Kanda et al., 2022) and multimodal body tracking (Vallet et al., 2016; Nickel et al., 2005; Wang and Brandstein, 1999), may also be improved by text-based speaker clustering techniques. Speaker clustering can also be a self-supervision dialogue pre-training procedure in the scenario that speaker annotation is adequate. Therefore, it is valuable to develop a model to reconstruct the missing identities of speakers in textual dialogue data.

In order to reconstruct the speaker labels in the dialogue, this work is dedicated to the method of speaker clustering. Different from previous researches on speaker identification (Kundu et al., 2012; Ma et al., 2017; Ek et al., 2018), which aim at selecting the most similar speaker from the pre-modeled candidates, the speaker clustering task aims at grouping the utterances into speaker-specific clusters without any preset candidates (Lukic et al., 2016). It is more useful because it works on open corpus where the speakers cannot be modeled in advance.

Speaker clustering is relevant to dialogue structure, because the process of turns follows certain patterns. These patterns include the semantic content and the communicative functions of utterance, and can be specifically represented as the dialogue act (DA) of utterance and associations between dialogue acts respectively (Bunt et al., 2010). The associations between dialogue acts include question-answer, request-response, offer-acceptance, etc., which are closely related to alternation of speakers. Conversely, the relations between speakers will be predicable if these patterns are available from textual utterances.

In this work, the speaker relations are inferred from the communicative functions by using an utterance-level pairwise matrix. The speaker relations have only two possible values, either same or different. The relations among the whole dia-

*Corresponding author.

Proceedings of the 29th International Conference on Computational Linguistics, pages 734–744
October 12–17, 2022.

logue form this matrix, which is regarded as the similarity matrix of the ground speakers.

The matrix can reconstruct the clusters of speakers with a density-based clustering method. The most popular algorithms of density-based clustering are spectral clustering (Von Luxburg, 2007) and DBSCAN (Hess et al., 2019). In this work, we use spectral clustering as the implementation, because it is less sensitive to sparse points, which follows this task that each utterance must be in a cluster.

Based on the above analysis, we build a model that models the semantic content of utterances with multi-task cross-corpus DA supervision, calculates the speaker relations with the form of bilinear, and generates the cluster labels with the method of spectral clustering.

The main contributions of this paper are summarized as follows.

- We build a speaker clustering model for textual dialogue, which explicitly exploits the communicative functions to reconstruct speaker relations and outperforms the baseline.
- The model can be further improved by auxiliary DA classification task. Even if a dataset is lack of DA annotations, the model can still be improved by cross-corpus DA data.
- We discuss the reliability of our method to predict the number of clusters, the ability to disambiguate between speaker relation and utterance text similarity, and the impact of utterance distance.

2 Related Work

This work targets for speaker clustering, and is based on the theories of dialogue structure.

2.1 Speaker Clustering

As far as we know, there are few works directly on speaker clustering in textual dialogues. However, there are some previous works on speaker diarization in voice conversations, and speaker clustering is the most important step in speaker diarization (Tranter and Reynolds, 2006; Anguera et al., 2012; Park et al., 2022). But these works only use audio features as the basis to calculate relation without considering the semantic information.

A previous work on speaker diarization through pairwise relations based on audio (Lin et al., 2019) uses spectral clustering as the top-level structure, which provides an idea for our structural design. But its focus is only audio features too, and it just descends the loss similarity score without training more fundamental features into fixed classes, which makes it difficult for the feature extraction process to guarantee generalization.

2.2 Dialogue Structure

The early researches in dialogue processing have noticed that a dialogue is made up of turns. Each turn is a combination of a speaker and an utterance. The turns are push ahead following the semantic cue. Specifically, dialogue turns have semantic content and communicative functions, which can be represented as dialogue acts (Searle and Searle, 1969) and adjacency pairs (Schegloff and Sacks, 1973) respectively. Every turn has its own dialogue act. Two turns from different speakers will form an adjacency pair if they have a behavior of interaction. Base on statistical or machine learning methods, it is realizable to predict the dialogue acts or the adjacency pairs (Surendran and Levow, 2006; Li et al., 2019; Li and Wu, 2016; Zhang et al., 2018). The semantic content and communicative functions involve the relations between speakers.

Pre-trained language models (Devlin et al., 2019; Lewis et al., 2020; Brown et al., 2020) have demonstrated their effectiveness on semantic modeling. These works illustrate the idea of represent semantic content with contextualized embeddings, i.e., trainable distributed vector in semantic space. However, most of the above models output word-level embeddings to represent the meaning of a word instead of the meaning of a whole sentence. There are solutions to convert from word-level embeddings to utterance-level embeddings, including using the corresponding embedding of the [CLS] token and using some pooling strategies (Ma et al., 2019; Xiao, 2018).

2.3 Other Works Related to Speakers in Dialogue

There are some researches relevant to speaker labeling in textual dialogues (Kundu et al., 2012; Ma et al., 2017; Ek et al., 2018), but they are not speaker clustering models directly. Most of them depend on the assumption that each speaker has its own speaking feature, e.g. the proportion of stop words, short words, adverbs in its utterances. Turn-taking detection is another type of speakers labeling (Liang and Zhou, 2020; Aldeneh et al.,

2018). It refers to identifying the positions where the speakers change during the dialogue, but it only focuses on the relations between two adjacent utterances, instead of every pair of utterances among a multi-party dialogue.

3 Model

The three main processes of this model are getting representation of utterances, cooperating with cross-corpus DA supervision, and calculating the similarity score. Therefore, as the overall structure shown in Figure 1, the model is divided into three parts in general: the utterance embedding part (blue), the speaker clustering part (yellow), and the set-specific DA classification part (red).

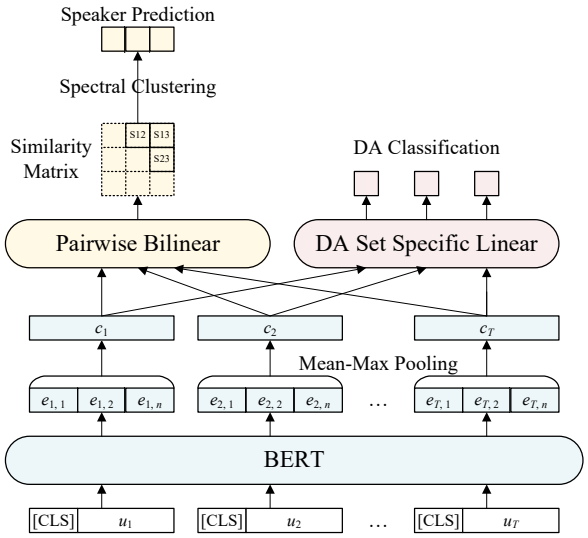

Figure 1: Model structure.

During training process, each data batch consists of B dialogues. To simplify the expression, we will omit the term of batch averaging in the following formulas. In a dialogue of the batch, there are T turns. The speaker of the i-th turn is s_i. The utterance of the i-th turn is u_i.

The objective of the model follows multi-task learning framework. The loss function of each data batch L is a combination of the binary cross entropy loss of the pairwise matrix L_{mat} and the cross entropy loss of the DA prediction L_{DA}. We use a hyperparameter λ to moderate the association between the two objectives. Formally,

$$L = L_{\text{mat}} + \lambda L_{\text{DA}}. \tag{1}$$

The objective and structure will be described in detail in the following sections.

3.1 Utterance Embedding

The first step of this model is to represent the semantic content of utterances as distributed vectors. Following previous works on text representation and dialogue processing (Ma et al., 2019; Gu et al., 2021), we concatenate the utterances in the dialogue with a [CLS] token prepended at the beginning of every utterance, and append a [SEP] token after them. For a dialogue with T utterances $u_1, u_2, \ldots, u_T$, the input format is

$$\text{[CLS]}\ u_1\ \text{[CLS]}\ u_2\ \ldots\ \text{[CLS]}\ u_T\ \text{[SEP]}.$$

Comparing to modeling each utterance in a separate pre-trained language model, this format is more lightweight that uses only a single BERT model, and contributes to directly calculate the word-level attentions across the utterances.

For each utterance, we take the output vectors of all the tokens (including the leading [CLS] token), and concatenate the mean pooling and max pooling results as the semantic representation, i.e., contextualized utterance embedding. Formally, the j-th token of the utterance u_i corresponds to the contextualized token embedding $e_{i,j}$ outputted by BERT. The utterance embedding is

$$c_i = \text{concat}\left[\underset{j}{\text{mean}}(e_{i,j}), \underset{j}{\max}(e_{i,j})\right], \tag{2}$$

where mean and max are mean pooling and max pooling functions through the stream dimension. For a BERT model of hidden size d_{BERT}, the length of the contextualized utterance embedding is $2d_{\text{BERT}}$.

3.2 Speaker Clustering

The relations between speakers are calculated by the form of bilinear. Specifically, for a dialogue with T turns, the contextualized utterance embeddings are

$$c_1, c_2, \ldots, c_T \in \mathbb{R}^{2d_{\text{BERT}}}.$$

The similarity score of the utterances u_m and u_n is the sigmoid mapping of bilinear form

$$\text{sim}(m, n) = \sigma(c_m^{\mathsf{T}} W c_n + b), \tag{3}$$

where $W \in \mathbb{R}^{2d_{\text{BERT}} \times 2d_{\text{BERT}}}$ and $b \in \mathbb{R}$ are trainable parameters.

For each pair of utterances, the similarity score is a real number between 0 and 1, denotes the probability that the corresponding speakers are identical.

The similarities are symmetric, so each pair of utterances is calculated just once, i.e., always having $m < n$ in Equation 3. All pairs of utterances finally form a symmetric $T \times T$ matrix.

The loss function of the matrix is calculated with the elements of the triangular. Formally,

$$L_{\text{mat}} = \frac{1}{C} \sum_{m=1}^{T-1} \sum_{n=m+1}^{T} \text{BCE}[\text{sim}(m, n), \mathbb{I}(s_m = s_n)], \quad (4)$$

where $C = T(T-1)/2$ is the number of the utterance pairs in the dialogue, $\mathbb{I}$ is indicator function, and BCE is Binary Cross Entropy loss function[1].

It is worth noticing that no additional positional encoding or embedding is added when calculating the similarity scores. We find that the positional information taken from BERT is enough for current calculation. Adding another positional information to this layer does not improve the performance according to our preliminary experiments.

We follow the spectral clustering algorithm to cluster the utterances into clusters that each cluster has the same speaker and different clusters have different speakers (Von Luxburg, 2007; Lin et al., 2019). For the audiences who are not familiar with this technique, spectral clustering is a clustering approach based on similarity graph and graph min-cut problem, which has nothing to do with speech spectra.

Given the symmetric similarity matrix $S \in \mathbb{R}^{T \times T}$, we compute both of the two kinds of normalized graph Laplacians, L_{sym} and L_{rw}, which are the same as the definition in the review (Von Luxburg, 2007). We use the eigenvalues of L_{rw} to determine the number of clusters, and the eigenvectors of L_{sym} to cluster[2].

The eigenvalues of the Laplacian matrix are related to the number of clusters. If the appropriate number of clusters is k, there will be a larger difference between the k-th smallest eigenvalue and the $(k+1)$-th smallest eigenvalue, which is known as the spectral gap. The greater the number of clusters, the less the overall eigenvalues will be. Therefore, an appropriate threshold can be selected on the validation set. If the k-th eigenvalue is greater than the threshold, the number of clusters will be considered to be less than k. Conversely, if the k-th eigenvalue is less than the threshold, the number of clusters will be considered to be greater than or equal to k. The threshold is adjusted on the validation set to maximize the accuracy. We report the results of both using the actual number of speakers as the number of clusters and using the spectral gap method to determine the number of clusters in the experiment section.

3.3 Auxiliary Set-specific Dialogue Act (DA) Classification

This part is designed as a auxiliary task to infuse dialogue act information into utterance embeddings. We assume that the ability of understanding semantic content will be stronger and the calculation of similarity will be more accurate if the model can predict the dialogue act of utterance correctly.

We present DA classification as part of the multi-task learning framework. For each dataset, if there are dialogue act annotations, we can use these labels to supervise the model to adjust the embeddings so that they express the corresponding dialogue acts. However, there is a problem that most of the DA-annotated datasets are not big enough, comparing to the speaker-annotated datasets. Meanwhile, these datasets are annotated with different sets and rules, and they are difficult to map to each other.

To solve this problem, we use a set-specific linear layer to adapt to different DA annotation sets. For different DA annotation sets, we use different linear layers to predict the corresponding number of dialogue act types. The loss function L_{DA} is calculated by the multi-class cross entropy[3] of the corresponding linear layer, and the output from other linear layers is ignored. With a shallow layer, we can expect to obtain a more general semantic representation. Formally,

$$L_{\text{DA}} = -\frac{1}{T} \sum_{m=1}^{T} \log \frac{\exp(z_{m,t_m})}{\sum_{d=1}^{D} \exp(z_{m,d})}, \quad (5)$$

where $z_{m,d}$ is the output of the set-specific linear layer of the m-th turn, d-th DA class, and t_m is the actual DA class of the m-th turn.

[1] For the definition, refer to: https://pytorch.org/docs/stable/generated/torch.nn.BCELoss.html .

[2] Implemented by scikit-learn and called with parameter `assign_labels="discretize"`.

[3] For the definition, refer to: https://pytorch.org/docs/stable/generated/torch.nn.CrossEntropyLoss.html .

4 Experiment

4.1 Datasets

Our datasets are composed of three corpora: the Switchboard Dialogue Act Corpus (SwDA) (Stolcke et al., 2000), the Meeting Recorder Dialogue Act Corpus (MRDA) (Shriberg et al., 2004), and the Ubuntu Dialogue Corpus (Lowe et al., 2015). The SwDA Corpus and the MRDA Corpus are two common DA-annotated datasets. The SwDA Corpus is a two-party dialogue dataset transcribed by phone calls. The DA annotations are divided into 217 small categories and 43 major categories. The MRDA Corpus is a multi-party dialogue dataset transcribed by conferences. The DA annotations are divided into 52 full categories, 12 general categories, and 5 basic categories. The Ubuntu Dialogue Corpus is a widely used dialogue dataset collected from the chat records on the Ubuntu IRC system, without DA annotation. For all the three datasets, the adjacent utterances may be from the same speaker.

For the SwDA Corpus, we first split the dialogue streams into 10-turn segments, and then randomly divide them into training, validation and test set by the ratio of 8:1:1. For the MRDA Corpus, we use the same set division as the original data, and then split the dialogue streams into 10-turn segments. For the Ubuntu Dialogue Corpus, We use the 10-turn version released by previous works (Ouchi and Tsuboi, 2016; Gu et al., 2021). Table 1 shows the basic quantity statistics of the datasets.

Dataset	Set	Dialogues	S/D
SwDA	Train	17059	2.00
	Valid	2132	2.00
	Test	2132	2.00
MRDA	Train	7485	3.01
	Valid	1636	2.91
	Test	1664	2.96
Ubuntu	Train	495226	4.08
	Valid	30974	4.21
	Test	35638	4.19

Table 1: Statistics of the datasets. "S/D" stands for "average number of different Speakers per Dialogue".

In the experiments, we use the 43 major categories of SwDA and the 52 full categories of MRDA as our target DA sets in the auxiliary task.

We propose the results of the SwDA dataset and the MRDA dataset as DA-annotated single-corpus scenarios to analyze the role of the pairwise calculation and the auxiliary DA classification task, and the result of simultaneously training on SwDA, MRDA, and Ubuntu datasets as a sophisticated cross-corpus scenario. We will focus more on the experimental results on the MRDA dataset, because this dataset is both DA annotated and multi-party, which is convenient to analyze various aspects of the model.

4.2 Metrics

We employ two metrics in the experimental results, the adjusted Rand index (ARI) (Hubert and Arabie, 1985)[4] and the accuracy (ACC). The adjusted Rand index is a common metric for clustering, which measures the similarity between two sets of clusters. The value ranges from -1 to 1. For a random clustering, the mathematical expectation of ARI is 0. The accuracy is calculated by transforming the clustering problem into a classification problem. The idea is finding the best injective mapping from the predicted clusters to the actual clusters. Formally, enumerate all permutations of the set $\{1, 2, \ldots, n\}$ where n is the number of predicted clusters, so that

$$\mathrm{ACC}\left(y, \hat{y}\right) = \max_{p \in P} \frac{1}{T} \sum_{i=1}^{T} \mathbb{I}\left[p\left(\hat{y}^{(i)}\right) = y^{(i)}\right], \tag{6}$$

where y is the labels of actual clusters, $\hat{y}$ is the labels of predicted clusters, p is a permutation of the set $\{1, 2, \ldots, n\}$, $\mathbb{I}$ is indicator function, and $y^{(i)}$ is the element on index i in vector y.

The ACC result is utterance-level average statistics, which is the number of correctly cluster-assigned utterances divided by the total number of turns in the dataset. The ARI result is dialogue-level average statistics, which is the mean ARI values among the dialogues.

The reason for using accuracy as a metric is that it is convenient to observe the difference between the predicted speakers and the real speakers after mapping. And it provides a comparable result with other speaker identification models, not just speaker clustering models.

4.3 Setup

We use the PyTorch framework (Paszke et al., 2019) and common backpropagation for training. During

[4]Implemented by scikit-learn.

training, we calculate the metrics on the validation set and save the model parameters that maximize the accuracy on the validation set to avoid overfitting.

We use AdamW (Loshchilov and Hutter, 2019) as the optimizer. By validating on the SwDA dataset, we select the hyperparameters in lr={1×10^{-5}, 2×10^{-5}, 3×10^{-5}}, eps={1×10^{-4}, 1×10^{-5}, 1×10^{-6}}, and weight_decay={0, 1×10^{-4}}, to maximize the accuracy on the validation set. The final choice, lr=2×10^{-5}, eps=1×10^{-6}, weight_decay=0, betas=(0.9, 0.999), are used for all datasets.

We use BERT-base-uncased provided by Google (Devlin et al., 2019; Turc et al., 2019) as the initialization parameter of the BERT part. All of the BERT parameters and other linear and bilinear parameters are fine-tuned end-to-end.

For the SwDA and MRDA single-corpus experiments, we select the association hyperparameter in $\lambda = \{0.01, 0.05, 0.1, 0.2, 0.3, 0.5\}$. Every setting is trained on a single RTX 2080Ti GPU for about 1.5 hours to select the best one on the validation set. The final choice is $\lambda = 0.2$ for SwDA and $\lambda = 0.1$ for MRDA.

For the SwDA, MRDA, and Ubuntu cross-corpus experiment, we select the association hyperparameter in $\lambda = \{0.005, 0.01, 0.1\}$. For each training step, the data batch consists of 3 random Ubuntu dialogue segments, 1 random SwDA dialogue segment, and 1 random MRDA dialogue segment. Every setting is trained on a single RTX 2080Ti GPU for about 2 days to select the best one on the validation set. The final choice is $\lambda = 0.01$.

4.4 Baselines

Due to the lack of related works of text-based speaker clustering, we cannot find an existing model that is directly comparable. So we implement our baselines to prove the necessity of the model design.

The first design to test is modeling the pairwise relations. For comparison, we implemented a general sequence classification model that changes the speaker clustering part (including pairwise bilinear layer and similarity matrix layer) to a multi-class softmax layer. The number of output classes is set to the maximum number of different speakers in the dialogue. We trained this baseline model to predict the sequential IDs of speakers in a dialogue.

The second design to test is the set-specific dialogue act classification task. For comparison, we set $\lambda = 0$ as the ablation setting in this scenario, while other parameters including the constitution of input batches are consistent.

4.5 Results

Our experimental results of the single-corpus scenarios are shown in Table 2 and Table 3. The result of the cross-corpus scenario is shown in Table 4.

Model	Valid		Test	
	ACC	ARI	ACC	ARI
Baseline	.760	.486	.748	.463
Clustering	**.868**	**.596**	**.860**	**.575**
- w/o DA Task	.865	.585	.856	.566

Table 2: Result of SwDA dataset. The number of clusters is set to 2 as the consistent groud-truth.

Model	Valid		Test	
	ACC	ARI	ACC	ARI
Baseline	.543	.204	.527	.179
Clustering*	**.714**	**.317**	**.703**	**.301**
- w/o DA Task*	.706	.306	.700	.296
Clustering†	**.654**	**.298**	**.644**	**.279**
- w/o DA Task†	.648	.286	.642	.277

Table 3: Result of MRDA dataset. *: Given the actual number of different speakers in the dialogue as the number of clusters for spectral clustering. †: Using spectral gap method to predict the number of clusters for spectral clustering.

Table 2 and Table 3 show the results of SwDA and MRDA datasets respectively. Our multi-task clustering model outperforms the sequence classification baseline and the ablative setting without auxiliary DA classification task in all the tests. These results prove that our auxiliary task improves the semantic content representation and similarity calculation if the training data has DA annotation and the evaluating data has the same distribution as the training data. The result of using the spectral gap method to detect the number of clusters shows that this model still outperforms the baseline and the ablative setting even without prior knowledge of the actual number of clusters.

Table 4 shows the results of training on all of the three datasets, and evaluating on either all three datasets or just the Ubuntu datasets. This model still outperforms the baseline in all the tests. It also outperforms the ablative setting in all the tests in the scenario of given the ground-truth number

Model Structure	S+M+U				S+M+U (Ubuntu Only)			
	Valid		Test		Valid		Test	
	ACC	ARI	ACC	ARI	ACC	ARI	ACC	ARI
Baseline	.530	.249	.531	.247	.513	.234	.516	.235
Clustering*	**.697**	**.299**	**.695**	**.296**	**.685**	**.279**	**.685**	**.280**
- w/o DA Task*	.696	.297	.694	.292	.684	.277	.684	.277
Clustering†	.632	**.284**	.631	**.282**	**.618**	**.264**	**.619**	.264
- w/o DA Task†	**.633**	.283	**.632**	.281	**.618**	**.264**	**.619**	**.265**

Table 4: Result of training synergistically on SwDA, MRDA, and Ubuntu datasets, and evaluating on the three datasets (left) or only the Ubuntu dataset (right). *: Given the actual number of different speakers in the dialogue as the number of clusters for spectral clustering. †: Using spectral gap method to predict the number of clusters for spectral clustering.

of speakers. Even the Ubuntu-only result is promoted by our set-specific DA classification task. This proves that cross-corpus supervised training is possible if we design the model with reasonable structure and objective.

Another phenomenon reflected in Table 4 is that, without specifying real number of speakers, there is a different trend between the results of ACC metric and ARI metric on S+M+U data. Actually, ARI is more concerned about whether the dividing points of clusters are correct, while ACC is the result after mapping. Therefore, ARI is a more direct metric that indicates whether the key points of speaker alternation are found correctly.

5 Discussion

In this section, we discuss whether the substructures of the model work accurately, and whether the model is disturbed by some possible factors (semantic similarity and utterance distance).

5.1 Determining the Number of Speakers

For clustering problems, it is an important step to predict an appropriate number of clusters. The principle of using spectral gap to predict the number of clusters has been described in Section 3.2. In order to verify whether this method can accurately predict the number of clusters, we make statistics on the MRDA dataset. We also tried training a multilayer perceptron (MLP) with the eigenvalues to predict the number of clusters. The multilayer perceptron uses 90% of the validation set data for training and the remaining 10% for validation.

Table 5 shows that the spectral gap method can predict more than 94% of the test data almost accurately, where error is less than or equal to 1. This method is more accurate than the multilayer perceptron.

Method	Accurate		$\leq \pm 1$	
	Valid	Test	Valid	Test
Spectral Gap	.485	**.482**	.941	**.942**
- w/o DA Task	.464	.468	.941	.941
MLP	.498	.480	.927	.928
- w/o DA Task	.499	.467	.928	.931

Table 5: Speaker (cluster) number prediction accuracy on the MRDA dataset. "$\leq \pm 1$" means the proportion of data whose difference between the predicted value and the actual value is less than or equal to 1.

The result also shows that the DA auxiliary task can not only directly improve the accuracy of speaker relation detection, but also help improve the accuracy of speaker number prediction.

5.2 Distinguishing Speaker Relation and Semantic Similarity

The similarity calculation takes a bilinear form. In this case, it is necessary to check whether the model confuses speaker similarity and utterance text similarity (semantic similarity). Semantic similarity is one of the fundamental features for inferring the relation between speakers, i.e., utterances from the same speaker tend to be semantically similar (Kundu et al., 2012; Ma et al., 2017; Ek et al., 2018). However, it would be harmful if the model takes semantic similarity as the only factor in prediction, because utterances from different speakers with same words are very common in dialogues, such as greetings, farewells, and rhetorical questions, and they will make a higher rate of false positives. Therefore, it is necessary to prove that the utterance embeddings and the similarity scores take full account of the contextual utterances, instead of simply extracting context-independent semantic features of the utterances.

	Accurary	Student's t-test p-value	Spearman's rank
Correctness	P1−S1 = $\mathbf{0.645}$	P2−S1 = $\mathbf{4.49 \times 10^{-104}}$	-
Confusion	P1−C1 = 0.550	P1−C2 = 8.66×10^{-12}, P2−C1 = 9.70×10^{-4}	P2−C2 = 0.0866
Inherence	S1−C1 = 0.526	S1−C2 = 1.23×10^{-3}	-

Table 6: The correlations about correctness (between P and S), confusion (between P and C), and inherence (between S and C). There is no Spearman's rank correlation coefficient about correctness or inherence because S does not have numerical dimension.

The way of demonstration is calculating three types of correlations:

- **Correctness**: The correlation between prediction results (P) and speaker relations (S).
- **Confusion**: The correlation between prediction results (P) and context-independent semantic similarities (C).
- **Inherence**: The correlation between speaker relations (S) and context-independent semantic similarities (C).

If the correctness is much greater than the confusion, it will prove that the model is aware of speaker relations without being compromised by context-independent semantic similarity. The inherence is necessary because the speakers and the semantic features are dependent, and the ground correlation between them needs to be excluded.

To determine these three types of correlations, we collected values in 5 dimensions:

- P1: The prediction result of whether the speakers are same or different (binary values).
- P2: The prediction result of similarity score in the pairwise matrix (numerical values).
- S1: Whether two speakers are same of different (binary values).
- C1: Whether two utterances are semantically similar or dissimilar (binary values).
- C2: The semantic similarity between two utterances (numerical values).

The values are collected from the validation set of MRDA. We select one pair of turns with the same speaker and one pair of turns with different speakers from each dialogue to form a new dataset. In this dataset, the two types of S1 are balanced. The utterance embeddings are calculated with a pre-trained-only BERT model[5], and the values in C2 are calculated by cosine similarity between the embeddings. Then, the pairs of turns are sorted by C2 in ascending order, and the first half of the pairs are regarded as dissimilar pairs, and the last half of the pairs are regarded as similar pairs, forming a balanced C1. The values in P1 and P2 are predicted by the model.

The experimental results are divided into three categories:

- The correlation between two binary dimensions is evaluated by the accuracy (whether it meets the hypothesized association). A greater value indicates a stronger correlation.
- The correlation between a binary dimension and a numerical dimension is evaluated by Student's t-test. A smaller p-value indicates a stronger correlation.
- The correlation between two numerical dimensions is evaluated by Spearman's rank correlation coefficient. A greater absolute value indicates a stronger correlation.

The statistics of correlation are shown in Table 6. Three conclusions can be drawn from it:

First, there is a strong correlation between prediction results and speaker relations. As the Correctness row shows, P1−S1 is much greater than 0.5, and P2−S1 is very small.

Second, there is a ground correlation between speaker relations and context-independent semantic similarities. As the Inherence row shows, S1−C1 is slightly greater than 0.5, and S1−C2 is between 10^{-3} and 10^{-2}.

Third, there is a weak correlation between prediction results and context-independent semantic similarities. As the Confusion row shows, P1−C1 is slightly greater than 0.5, P1−C2 and P2−C1 are less than 10^{-3}, and P2−C2 is slightly greater than but close to 0. But this correlation is mainly

[5]BERT-base-uncased without any fine-tuning.

brought by the ground correlation between speaker relations and semantic similarities, because the correlations about Confusion approximately equal to the correlations about Inherence, and much less than the correlations about Correctness.

These results complete the demonstration that this model can detect speaker relations without being compromised by context-independent semantic similarity.

5.3 Distance Impacts Similarity Modeling

We investigate the results of internal layer of the pairwise similarity score by aggregating the position-level error of similarity matrix, as shown in Figure 2. The item in m-th row and n-th column is the mean error of the similarity score of the m-th turn and n-th turn. Formally,

$$\mathrm{err}(m,n) = \left|\mathrm{sim}(m,n) - y(m,n)\right|. \quad (7)$$

We take the results of similarity matrix on the Ubuntu test set, and plot the heatmap of mean error. The figure shows that the model successfully models the relations between utterances, especially the adjacent ones. For longer-distance pairs, it is constitutionally more difficult to be modeled, but the model is still effective with a mean error less than 0.5.

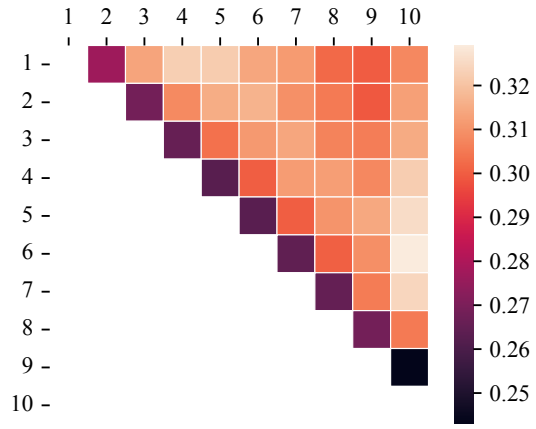

Figure 2: Error heatmap of similarity matrix on the Ubuntu test set with auxiliary DA classification task. Darker color means more accurate, and lighter color means more erring.

6 Conclusion

We propose a text-based dialogue speaker clustering model. Based on the theory of the dialogue structure, the model holds the semantic content and the communicative functions explicitly with the BERT layer and the similarity matrix. The model is enhanced by the idea of cross-corpus supervision with the auxiliary set-specific dialogue act classification task. It finally generates the cluster labels of speakers with spectral clustering. Our model outperforms the sequence classification baseline and the non-DA ablation on almost all tests. Additional discussion illustrates the accuracy in predicting the number of speakers (clusters) and the ability to distinguish between speaker relation and semantic similarity of our model. We also show that the precision of speaker similarity prediction varies with utterance distance.

In future research, it is worth trying further pretraining the model on dialogue data, which will likely help to perceive dialogue turns and extract better utterance embeddings. We will also explore for a method to make the similarity calculation between long-distance utterance pairs more accurate.

References

Daniel Adiwardana, Minh-Thang Luong, David R. So, Jamie Hall, Noah Fiedel, Romal Thoppilan, Zi Yang, Apoorv Kulshreshtha, Gaurav Nemade, Yifeng Lu, and Quoc V. Le. 2020. Towards a human-like open-domain chatbot. *CoRR*, abs/2001.09977.

Zakaria Aldeneh, Dimitrios Dimitriadis, and Emily Mower Provost. 2018. Improving end-of-turn detection in spoken dialogues by detecting speaker intentions as a secondary task. In *2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 6159–6163.

Xavier Anguera, Simon Bozonnet, Nicholas Evans, Corinne Fredouille, Gerald Friedland, and Oriol Vinyals. 2012. Speaker diarization: A review of recent research. *IEEE Transactions on Audio, Speech, and Language Processing*, 20(2):356–370.

Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel Ziegler, Jeffrey Wu, Clemens Winter, Chris Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. 2020. Language models are few-shot learners. In *Advances in Neural Information Processing Systems*, volume 33, pages 1877–1901. Curran Associates, Inc.

Harry Bunt, Jan Alexandersson, Jean Carletta, Jae-Woong Choe, Alex Chengyu Fang, Koiti Hasida, Kiyong Lee, Volha Petukhova, Andrei Popescu-Belis,

Laurent Romary, Claudia Soria, and David Traum. 2010. Towards an ISO Standard for Dialogue Act Annotation. In *Seventh conference on International Language Resources and Evaluation (LREC'10)*, La Valette, Malta.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, Minneapolis, Minnesota. Association for Computational Linguistics.

Adam Ek, Mats Wirén, Robert Östling, Kristina N. Björkenstam, Gintarė Grigonytė, and Sofia Gustafson Capková. 2018. Identifying speakers and addressees in dialogues extracted from literary fiction. In *Proceedings of the Eleventh International Conference on Language Resources and Evaluation (LREC 2018)*, Miyazaki, Japan. European Language Resources Association (ELRA).

Yihui Fu, Luyao Cheng, Shubo Lv, Yukai Jv, Yuxiang Kong, Zhuo Chen, Yanxin Hu, Lei Xie, Jian Wu, Hui Bu, Xin Xu, Jun Du, and Jingdong Chen. 2021. AISHELL-4: An Open Source Dataset for Speech Enhancement, Separation, Recognition and Speaker Diarization in Conference Scenario. In *Proc. Interspeech 2021*, pages 3665–3669.

Jia-Chen Gu, Chongyang Tao, Zhenhua Ling, Can Xu, Xiubo Geng, and Daxin Jiang. 2021. MPC-BERT: A pre-trained language model for multi-party conversation understanding. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 3682–3692, Online. Association for Computational Linguistics.

Sibylle Hess, Wouter Duivesteijn, Philipp Honysz, and Katharina Morik. 2019. The spectacl of nonconvex clustering: A spectral approach to density-based clustering. *Proceedings of the AAAI Conference on Artificial Intelligence*, 33(01):3788–3795.

Lawrence Hubert and Phipps Arabie. 1985. Comparing partitions. *Journal of classification*, 2(1):193–218.

Naoyuki Kanda, Xiong Xiao, Yashesh Gaur, Xiaofei Wang, Zhong Meng, Zhuo Chen, and Takuya Yoshioka. 2022. Transcribe-to-diarize: Neural speaker diarization for unlimited number of speakers using end-to-end speaker-attributed asr. In *ICASSP 2022 - 2022 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 8082–8086.

Amitava Kundu, Dipankar Das, and Sivaji Bandyopadhyay. 2012. Speaker identification from film dialogues. In *2012 4th International Conference on Intelligent Human Computer Interaction (IHCI)*, pages 1–4.

Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Veselin Stoyanov, and Luke Zettlemoyer. 2020. BART: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7871–7880, Online. Association for Computational Linguistics.

Ruizhe Li, Chenghua Lin, Matthew Collinson, Xiao Li, and Guanyi Chen. 2019. A dual-attention hierarchical recurrent neural network for dialogue act classification. In *Proceedings of the 23rd Conference on Computational Natural Language Learning (CoNLL)*, pages 383–392, Hong Kong, China. Association for Computational Linguistics.

Wei Li and Yunfang Wu. 2016. Multi-level gated recurrent neural network for dialog act classification. In *Proceedings of COLING 2016, the 26th International Conference on Computational Linguistics: Technical Papers*, pages 1970–1979, Osaka, Japan. The COLING 2016 Organizing Committee.

Yuhai Liang and Qiang Zhou. 2020. Detect turn-takings in subtitle streams with semantic recall transformer encoder. In *2020 International Conference on Asian Language Processing (IALP)*, pages 1–6.

Qingjian Lin, Ruiqing Yin, Ming Li, Hervé Bredin, and Claude Barras. 2019. LSTM Based Similarity Measurement with Spectral Clustering for Speaker Diarization. In *Proc. Interspeech 2019*, pages 366–370.

Pierre Lison, Jörg Tiedemann, and Milen Kouylekov. 2018. OpenSubtitles2018: Statistical rescoring of sentence alignments in large, noisy parallel corpora. In *Proceedings of the Eleventh International Conference on Language Resources and Evaluation (LREC 2018)*, Miyazaki, Japan. European Language Resources Association (ELRA).

Ilya Loshchilov and Frank Hutter. 2019. Decoupled weight decay regularization. In *7th International Conference on Learning Representations, ICLR 2019, New Orleans, LA, USA, May 6-9, 2019*. OpenReview.net.

Ryan Lowe, Nissan Pow, Iulian Serban, and Joelle Pineau. 2015. The Ubuntu dialogue corpus: A large dataset for research in unstructured multi-turn dialogue systems. In *Proceedings of the 16th Annual Meeting of the Special Interest Group on Discourse and Dialogue*, pages 285–294, Prague, Czech Republic. Association for Computational Linguistics.

Yanick Lukic, Carlo Vogt, Oliver Dürr, and Thilo Stadelmann. 2016. Speaker identification and clustering using convolutional neural networks. In *2016 IEEE 26th International Workshop on Machine Learning for Signal Processing (MLSP)*, pages 1–6.

Kaixin Ma, Catherine Xiao, and Jinho D. Choi. 2017. Text-based speaker identification on multiparty dialogues using multi-document convolutional neural networks. In *Proceedings of ACL 2017, Student Research Workshop*, pages 49–55, Vancouver, Canada. Association for Computational Linguistics.

Xiaofei Ma, Zhiguo Wang, Patrick Ng, Ramesh Nallapati, and Bing Xiang. 2019. Universal text representation from BERT: an empirical study. *CoRR*, abs/1910.07973.

Kai Nickel, Tobias Gehrig, Rainer Stiefelhagen, and John McDonough. 2005. A joint particle filter for audio-visual speaker tracking. In *Proceedings of the 7th International Conference on Multimodal Interfaces*, ICMI '05, page 61–68, New York, NY, USA. Association for Computing Machinery.

Hiroki Ouchi and Yuta Tsuboi. 2016. Addressee and response selection for multi-party conversation. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 2133–2143, Austin, Texas. Association for Computational Linguistics.

Tae Jin Park, Naoyuki Kanda, Dimitrios Dimitriadis, Kyu J. Han, Shinji Watanabe, and Shrikanth Narayanan. 2022. A review of speaker diarization: Recent advances with deep learning. *Computer Speech & Language*, 72:101317.

Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Kopf, Edward Yang, Zachary DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. 2019. Pytorch: An imperative style, high-performance deep learning library. In *Advances in Neural Information Processing Systems*, volume 32. Curran Associates, Inc.

Desh Raj, Pavel Denisov, Zhuo Chen, Hakan Erdogan, Zili Huang, Maokui He, Shinji Watanabe, Jun Du, Takuya Yoshioka, Yi Luo, Naoyuki Kanda, Jinyu Li, Scott Wisdom, and John R. Hershey. 2021. Integration of speech separation, diarization, and recognition for multi-speaker meetings: System description, comparison, and analysis. In *2021 IEEE Spoken Language Technology Workshop (SLT)*, pages 897–904.

Stephen Roller, Emily Dinan, Naman Goyal, Da Ju, Mary Williamson, Yinhan Liu, Jing Xu, Myle Ott, Eric Michael Smith, Y-Lan Boureau, and Jason Weston. 2021. Recipes for building an open-domain chatbot. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*, pages 300–325, Online. Association for Computational Linguistics.

Emanuel A. Schegloff and Harvey Sacks. 1973. Opening up closings. *Semiotica*, 8(4).

John R Searle and John Rogers Searle. 1969. *Speech acts: An essay in the philosophy of language*, volume 626. Cambridge university press.

Elizabeth Shriberg, Raj Dhillon, Sonali Bhagat, Jeremy Ang, and Hannah Carvey. 2004. The ICSI meeting recorder dialog act (MRDA) corpus. In *Proceedings of the 5th SIGdial Workshop on Discourse and Dialogue at HLT-NAACL 2004*, pages 97–100, Cambridge, Massachusetts, USA. Association for Computational Linguistics.

Andreas Stolcke, Klaus Ries, Noah Coccaro, Elizabeth Shriberg, Rebecca Bates, Daniel Jurafsky, Paul Taylor, Rachel Martin, Carol Van Ess-Dykema, and Marie Meteer. 2000. Dialogue act modeling for automatic tagging and recognition of conversational speech. *Computational Linguistics*, 26(3):339–374.

Dinoj Surendran and Gina-Anne Levow. 2006. Dialog act tagging with support vector machines and hidden markov models. In *Ninth International Conference on Spoken Language Processing*.

S.E. Tranter and D.A. Reynolds. 2006. An overview of automatic speaker diarization systems. *IEEE Transactions on Audio, Speech, and Language Processing*, 14(5):1557–1565.

Iulia Turc, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. Well-read students learn better: The impact of student initialization on knowledge distillation. *CoRR*, abs/1908.08962.

Félicien Vallet, Jim Uro, Jérémy Andriamakaoly, Hakim Nabi, Mathieu Derval, and Jean Carrive. 2016. Speech trax: A bottom to the top approach for speaker tracking and indexing in an archiving context. In *Proceedings of the Tenth International Conference on Language Resources and Evaluation (LREC'16)*, pages 2011–2016, Portorož, Slovenia. European Language Resources Association (ELRA).

Ulrike Von Luxburg. 2007. A tutorial on spectral clustering. *Statistics and computing*, 17(4):395–416.

Ce Wang and M.S. Brandstein. 1999. Multi-source face tracking with audio and visual data. In *1999 IEEE Third Workshop on Multimedia Signal Processing (Cat. No.99TH8451)*, pages 169–174.

Han Xiao. 2018. bert-as-service. `https://github.com/hanxiao/bert-as-service`.

Xuejing Zhang, Xueqiang Lv, and Qiang Zhou. 2018. Chinese dialogue analysis using multi-task learning framework. In *2018 International Conference on Asian Language Processing (IALP)*, pages 102–107.

Yizhe Zhang, Siqi Sun, Michel Galley, Yen-Chun Chen, Chris Brockett, Xiang Gao, Jianfeng Gao, Jingjing Liu, and Bill Dolan. 2020. DIALOGPT : Large-scale generative pre-training for conversational response generation. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics: System Demonstrations*, pages 270–278, Online. Association for Computational Linguistics.

TopKG: Target-oriented Dialog via Global Planning on Knowledge Graph

Zhitong Yang[1,3,2], **Bo Wang**[2,1]*, **Jinfeng Zhou**[2,1], **Yue Tan**[2,1], **Dongming Zhao**[4], **Kun Huang**[4], **Ruifang He**[2,1], **Yuexian Hou**[2]
[1]State Key Laboratory of Communication Content Cognition, People's Daily Online, Beijing, China
[2]College of Intelligence and Computing, Tianjin University, Tianjin, China
[3]School of New Media and Communication, Tianjin University, Tianjin, China
[4]AI Lab, China Mobile Communication Group Tianjin Co., Ltd.
{yyyyyyzt, bo_wang, jfzhou, tanyue_098}@tju.edu.cn

Abstract

Target-oriented dialog aims to reach a global target through multi-turn conversation. The key to the task is the global planning towards the target, which flexibly guides the dialog concerning the context. However, existing target-oriented dialog works take a local and greedy strategy for response generation, where global planning is absent. In this work, we propose global planning for target-oriented dialog on a commonsense knowledge graph (KG). We design a global reinforcement learning with the planned paths to flexibly adjust the local response generation model towards the global target. We also propose a KG-based method to collect target-oriented samples automatically from the chit-chat corpus for model training. Experiments show that our method can reach the target with a higher success rate, fewer turns, and more coherent responses.

1 Introduction

Human-like dialog agents have three types of approaches: open-domain (Zhang et al., 2019a; Huang et al., 2020), task-oriented (Budzianowski et al., 2018; Rastogi et al., 2020; Yang et al., 2020), and target-oriented dialog (Tang et al., 2019; Qin et al., 2020; Zhong et al., 2021). The open-domain dialog only requires the dialog generation to be fluent and context coherent. In contrast, typical task-oriented dialog further completes a specific task by understanding users' intention and collecting the required information of predefined sub-tasks of the intention. However, as a more challenging task, target-oriented dialog aims to achieve a global target that often can not be clearly defined as sub-tasks. The dialog agents are required to lead the conversation to the target flexibly, and the process is excepted to be coherent, effective, and successful. Due to its purpose and flexibility, target-oriented dialog agents have a broad-based demand, e.g., conversational recommendation (Li et al., 2018; Kang et al., 2019), psychotherapy (Sharma et al., 2020), and education (Clarizia et al., 2018). In these fields, a typical expectation of target-oriented dialog is to actively lead the conversation by smoothly changing the dialog topic to a designated one, e.g., a product, a stimulus of mind, and a knowledge point.

To reach a target topic effectively and coherently in dialog, existing approaches primarily represent the topic as keywords and adopt a two-stage architecture, i.e., predicting a next-turn keyword and keyword-augmented response retrieval (Tang et al., 2019). In this direction, Xu et al. (2020b) further introduces reinforcement learning with "target similarity" rewards to target-oriented dialog learning. However, the target-oriented dialog is a typical knowledge-rich task. Although dialog context can support the semantic concern of dialog generation, it is not quite effective to model the knowledge-driven process in the target-oriented dialog. To involve global knowledge, Qin et al. (2020) and Xu et al. (2020a) incorporate a dialog graph into the target-oriented dialog and Zhong et al. (2021) uses the external commonsense KG (ConceptNet (Speer et al., 2017)) to improve the performance.

Although existing target-oriented dialog works have demonstrated practical approaches in self-simulation test, there is still some open issues: (1) Lack of multi-turn target-oriented dialog corpus for training and benchmarks. Most existing target-oriented corpus are prepared for next-turn local target (e.g., OTTers(Sevegnani et al., 2021)), or adopt chit-chat corpora and randomly select a keyword in the next-turn utterance as the local target, (2) Lack of global planning of dialog process. Although the latest works use a global target to guide every turn of response generation, they adopt a short-sighted and greedy strategy instead of global planning to optimize the process towards the global target.

To this end, we propose **T**arget-**O**riented dialog with global **P**lanning on **K**nowledge **G**raph (**TopKG**), which effectively supports the target-

*Corresponding author.

Proceedings of the 29th International Conference on Computational Linguistics, pages 745–755
October 12–17, 2022.

oriented process by global reasoning on KG concerning the dialog context. Specifically, to address the first data issue, we automatically select a new dataset named Target-Guided ConvAI (**TGConv**) from the chit-chat corpus ConvAI2 (Dinan et al., 2020). We select target-oriented samples from ConvAI2 by identifying the dialog utterances containing a go-through entity sequence that aligns with the KG path. Furthermore, we distinguish the selected dialog samples according to whether the global target is easy to reach or not to verify the performance of TopKG in dealing with hard global target-oriented cases. For instance, the sample in the left part of Figure1 is target-oriented because the keywords in this dialog are connected (direct or low-order connected) in a commonsense KG, which embodies a smooth transition towards global target words. To address the second issue, we first improve the existing one-turn target-oriented response generation, trained in a supervised fashion to predict a next-turn keyword and generate a fluent and coherent response with the predicted keyword. Using the improved one-turn model as local-model, we further introduce a reinforcement learning based global-model to effectively guide the local-model towards a global target with global planning on KG. Specially, the global-model adjusts the next-turn keyword selection of the local-model to follow the global planning path on KG and reward the keyword-based response generation with success in reaching the global target.

Our main contributions are as follows:

(1) We propose a simple yet effective way to automatically extract multi-turn global target-oriented dialog from the chit-chat corpus to develop global target-oriented dialog agent. We also distinguish the selected dialog into easy-to-reach target and hard-to-reach target.

(2) We make the first step towards global planning in global target-oriented dialog. A two-stage learning framework is designed to guide a next-turn local model with a reinforcement learning based global model which is guided by global planning in commonsense KG.

(3) With automatic and human metrics, we verify that TopKG exceeds baselines on reaching global target with more coherent semantics, fewer turns, and a higher success rate in reaching targets.

The dataset can be downloaded in data folders from https://github.com/yyyyyyzt/topkgchat

2 Related Work

Target-oriented dialogue systems. Current target-oriented dialog studies can generally be divided into local-target oriented and global-target oriented methods. Local-target oriented methods (Wang et al., 2021) pays attention to the next-turn target. For example, Xu et al. (2020b,a) proposes a hierarchical policy model to plan and generate responses of different levels where the high-level policy plans a topic. However, the low-level policy plans responses that are coherent to this topic instead of approaching it. Global-target oriented methods (Qin et al., 2020; Zhong et al., 2021) uses global target to guide every turn of response generation. These methods propose a keyword predictor to determine the next-turn keyword to talk about and produce a response relevant to the determined keyword. However, they adopt a short-sighted and greedy strategy instead of explicit planning to optimize the process towards the global target.

KG-grounded dialogue systems. Leveraging background information for dialogue system improvement is a well-researched topic, especially in target-oriented settings. Some work uses structured knowledge, DKRN (Qin et al., 2020) incorporates a dialog graph, and CKC (Zhong et al., 2021) uses the ConceptNet to improve the performance. For how to utilize KG, classical methods are divided into using full path (Ma et al., 2021) and using flexible path fragments (Zhou et al., 2021). These models enjoy rich knowledge augmentation since short KG paths relating to the context are encoded, but they lack the ability to plan on KG. Another set of works focuses on grounds in unstructured knowledge (Zhao et al., 2020; Wu et al., 2020), which can also be divided into independent sentences and documents. This unstructured knowledge is more challenging to use than KG.

3 Our Approach

Task Definition Formally, $C = \{c_1, \cdots, c_i\}$ is the current dialog context involving latest i utterances. A knowledge graph $G_{KG} = V_{KG} \times E_{KG}$ is composed of the commonsense entities V_{KG} and relations E_{KG}. Given C, G_{KG} and a global target keyword K_{target}, the global target-oriented dialog is firstly required to figure out a next-turn keyword z from the G_{KG}, and generate a response r related to z. Furthermore, with multi-turn response generation, the global target-oriented dialog need to successfully mentioned a global target keyword

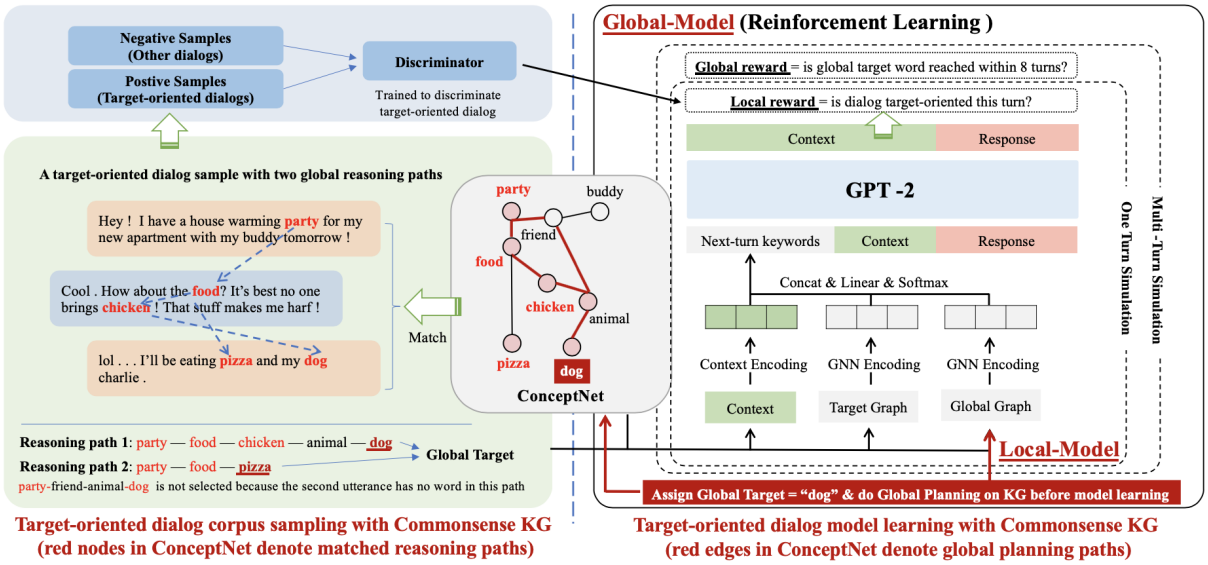

Figure 1: The left part illustrates how to select target-oriented dialogs for model learning by matching the word sequence across utterances with reasoning paths in KG. The right part illustrates how the global-model uses reinforcement learning to guide a GPT-2 based local-model to follow the global planning in KG. Global Planning on KG is pre-performed before the learning of Local-Model and Global-Model, and planning paths are essential to guide the multi-turn responses generation.

K_{target} with fewer turns and keep the response be coherent to the context in each turn.

3.1 Method Overview

Our approach consists of two main contributions: an automatic method for target-oriented conversation dataset collection and a two-stage learning model for global target-oriented dialog generation.

Target-oriented Conversation dataset As existing multi-turn dialog corpora are not specially created for target-oriented tasks, we firstly propose automatically selecting the target-oriented dialog session from the general dialog corpora. A dialog session was selected from the general chit-chat corpus by examining whether a KG-explainable entity path is running through a dialog. In addition, we indicate the entity path and specify the easy target and the hard target. The example shown in Table1.

Two-stage learning model We divide the task into two progressive stages in Fig1: local-model of next-turn strategy learning (stage 1) and global-model of multi-turn strategy learning (stage 2). Specifically, at stage 1, the local-model is supervised trained to predict next-turn keywords and generate a response related to the keywords. In stage 2, we design a reinforcement learning to adjust the local-model to explore all potential paths by global planning in a commonsense KG towards the global target word, where a bidirectional heuristic reasoning obtains the paths. We also reward the generated response in each turn by whether the dialog till this turn is target-oriented and whether the dialog finally reaches the global target word.

Dialog	**A:** I spend a lot of time **outside**. **B:** I like the outdoors as well, especially **gardening** . **A:** Wow! I used to have a **garden** too. **B:** I love sipping coffee while enjoying **flowers** in my garden. **A:** Flowers are always beautiful and **colorful** ! **B:** I like anything with **art**, especially colorful things.
Entity Path	**Outside-Garden-Flower-Color-Art**
Target	**Art**

Table 1: A target-oriented example dialog in TGConv

3.2 Target-oriented dialog corpus sampling

In this section, we construct a target-oriented dialog corpus (named **TGConv**) from chit-chat corpus ConvAI2 (Dinan et al., 2020).

3.2.1 Identify target-oriented dialog

We suppose a dialog is a positive example of target-oriented dialog if there is a consistent reasoning path of words linking all the utterances in their order in the dialog. A reasoning path of words is $p = \{w_1 \rightarrow w_2 \rightarrow \cdots \rightarrow w_n\}$, where w_i is a word, e.g., "*Outside-Garden-Flower-Color-Art*" in Table 1. To be logical, each neighbor word pair, i.e.,

w_i and w_{i+1}, should match the names of the two nodes of an edge in the ConceptNet, respectively. To link all the utterances in dialog, each utterance in the dialog should provide at least one word to p. To keep the order in dialogue, w_i should be in the same order in p as they appear in the dialog. Except for positive samples, other samples in the corpus are identified as negative examples.

3.2.2 Global target assignment

For each positive example dialog associated with a reasoning path p, we select the last word w_n in p as the global target K_{target}. Furthermore, to better evaluate the model's ability to guide the dialog to the target of different difficulties, we distinguish target words into "easy-to-reach" and "hard-to-reach". Specifically, target words with low frequency in the corpus are classified as "hard-to-reach" target, because there are fewer cases to learn the semantic transition to low-frequency target words (less than 800) in local-model and global-model.

3.3 Global Planning

Global planning is the key to successfully accomplishing target-oriented task. We finally obtain a graph consisting of a set of potential paths through global planning, which embodies the keyword transition from the initial context to the global target word. Building a connected graph G_{global} from the starting to target allows us to learn a better graph representation and facilitate our model to explore better paths. Specifically, we identify the noun and verb concepts in the dialogue context and then use a bidirectional reasoning method to find KG paths over ConceptNet effectively. Bidirectional reasoning is a graph search algorithm that finds smallest path from the initial to the target entity. It runs two simultaneous search: 1) Forward search from source/initial entity toward goal entity and 2) Backward search from goal/target entity toward source entity. This algorithm is very suitable for target-oriented task scenarios, and the detailed process is shown in Algorithm 1.

Algorithm 1: Global Planning by Bidirectional Reasoning over $ConceptNet$

Input : ConceptNet, G_{KG}; Target, K_{target};
The set of concepts in start:
$V_{start} = \{v_1, v_2 \cdots v_m\}$;
Output: A graph consists of all potential paths from source to target, G_{global}

```
Initialize graph G_global;
foreach node v_i of the V_start do
    Initialize a concept stack S contain v_i;
    for h from 1 to maximum hops H do
        while S is not emtpy do
            Pop a head entity v_h from S;
            N_i: the neighbouring concepts of v_h in
              ConceptNet;
            Select the top K concepts most similar
              to the head entity v_h from N_i;
            Select the top K concepts most similar
              to the target entity K_target from N_i;
            Add them to an empty temporal triple
              list T;
            foreach (v_h, r, v_t) in T do
                Add v_h, v_t and r into G;
                if v_t not in G_global then
                    Push v_t in S
                end
            end
        end
    end
end
Repeat the above process from K_target to V_start;
```

3.4 Supervised Learning of Local-Model

We let the local-model learn next-turn target-oriented policy in a supervised fashion. The local-model architecture is shown in the right part of Fig 1. In order to predict the next turn keywords z, we need to model the candidate words, the context, and the target, respectively. Firstly, we get the target entity and its neighbors on the ConceptNet to build a subgraph G_{target} and use the method in the previous section to get a global graph G_{global}. Then we apply a multi-layer GCN encoder to model the graphs. Besides, we use a typical transformer encoder for context understanding. Finally, we predict a keyword and generate a coherent response by generator for approaching the target.

3.4.1 Graph-based Encoder

We use a graph-based encoder to model graph node representations for predicting keywords. Here we use two graphs G_{global} and G_{target}, The G_{global} is a large graph that contains all potential paths from start context to target, and G_{target} only contains target entity and its neighbor nodes to enhance the target representation.

Therefore, to obtain the representation of concepts and relations, we apply multi-layer GCN (Kipf and Welling, 2016) encoders to encode the G_{global} and G_{target}. Moreover, following the idea of the TransE model (Bordes et al., 2013), we update a concept embedding with the subtraction between each neighbor concept embedding and the corresponding relation embedding to obtain the relation representation. The concepts V

in two graphs are initialized by pretrained word embeddings[1], and the relations R in graph are initialized with randomly embeddings. For each concept v_i, we update its embedding at the $(l+1)^{th}$ layer by aggregating its neighbours N_i including pairs of the concept and the relation liking to v_i:

$$h_i^{(l+1)} = \sigma\left(W_s^{(l)} h_i^{(l)} + \sum_{(j,r)\in N_i} \frac{1}{|N_i|} W_n^{(l)} \left(h_j^{(l)} - h_r^{(l)}\right)\right) \quad (1)$$

where h_i^l, h_j^l and h_r^l are the embeddings of node v_i, node v_j, and the relation between v_i and v_j at layer $(l)^{th}$; $W_s^{(l)}$ and $W_n^{(l)}$ are the two trainable parameter matrices specific to the layer $(l)^{th}$; and σ is a non-linear active function. The relation embedding is also updated at the $(l+1)^{th}$ layer via a linear active function: $h_r^{(l+1)} = W_R^{(l)} h_r^{(l)}$. After L layers, we are able to obtain a set of concept representations $\{h_{v_1}^{(L)}, \ldots, h_{v_{|V|}}^{(L)}\}$.

3.4.2 Conversation Context Encoder

We utilize a transformer encoder for conversation context understanding. Same as previous works, we flatten conversation context in C, and then add a special token $[CLS]$ at the beginning of the input. $\bar{C} = [CLS; C]$ is fed into Transformer Encoder, then output representation of $[CLS]$ token denoting the global memory of the whole sequence.

3.4.3 Classification

Now we have the context representation, G_{global} concepts representation, and G_{target} concepts representation for predicting words. Finally, we concatenated these vectors and fed to a linear transformation layer, followed by a softmax layer. We limited the candidates to two-hop entities based on context. The entire model is optimized by minimizing the cross-entropy loss.

3.4.4 Keyword Augmented Generator

After we get the next-turn keywords word z, we employ a keyword-augmented GPT (Radford et al., 2019) to generate a response to approaching the target. The generator takes keywords z and context C as the input, and the following text r as the target reference. Specifically, the z and C are first concatenated by a special separator token. The training objective follows a standard language model (LM) loss(Zhang et al., 2019b):

$$p_\Theta(r \mid C, z) = \prod_{t=0}^{|r|} p\left(r_t \mid x, z, r_{0:t-1}\right) \quad (2)$$

where r_t is the t-th token in r.

3.5 Reinforcement Learning of Global-Model

As our main contribution, we propose a global-model to explore better dialog strategies toward the global target through reinforcement learning. Although the local-model performs well on next turn response generation, it tends to be short-sighted and ineffective in reaching the global target in the multi-turn dialog. Therefore, we design a simulation-based environment to guide the local-model toward the global target through reinforcement learning. To this end, we let the model talk to itself. At the start of the dialog, we explicitly search a set of planning paths (described in 3.3) in ConceptNet from the initial context to the global target word. Then we use searched planning paths to adjust the next-turn keyword prediction to obey the planning paths and generate a response with the keyword. Furthermore, the generated response is rewarded by its target-oriented coherence to the context and the success of the global target. Global-model consists of the following components.

3.5.1 State/Action

At each time step t, the **state** S_t is a tuple of $[G_{global}; G_{target}; C]$, where G_{global} is a graph of planning paths obtained at the start of the dialog, and G_{target} is the predefined global target word and its neighbors, and C is the current context. Given the current dialog state, an **action** is the next-turn keyword z, and the action space is the potential paths obtained by global planning.

3.5.2 Reward

We use Local Reward and Global Reward to encourage the dialog to be contextual and coherent and explore global target-oriented strategy.

Local Reward encourages the contextual consistency at each turn of dialog, which is the discriminator score of the utterances sequence containing the current context and generated response, the detail are as below 3.5.3.

Global Reward encourages the global target-oriented response by giving a positive reward of "1" if the global target word finally appears in the last turn or a negative reward of "-1" otherwise.

[1]We use GloVe embedding of size 300 (Pennington et al., 2014)

3.5.3 Discriminator for local reward evaluation

To reward the dialog (context+response) which are more likely to be target-oriented, we train a discriminator to tell whether an utterance sequence is semantically target-oriented. To this end, the discriminator is trained to classify the positive and negative samples collected in section 3.2. Specially, the positive and negative samples with 1/0 label: $\mathbf{X} = [CLS; c; SEP; r]$ or $\mathbf{X} = [CLS; c; SEP; \overline{r}]$ is fed into pre-trained language model (BERT) (Devlin et al., 2018), then output representation of [CLS] token is used for classification. The classification score is formulated as

$$f_{score}(X) = \sigma(\mathbf{w}^\top \mathbf{x}_{[\mathrm{CLS}]} + b) \quad (3)$$

where w and b are trainable parameters. We use binary cross-entropy loss to optimize the models.

3.5.4 Training

We apply Proximal Policy Optimization (Schulman et al., 2017), a stable policy based RL algorithm using a constant clipping mechanism as the soft constraint, for dialog policy optimization:

$$J_\pi(\theta) = E_{s,a\sim\pi}\left[\min\left\{\beta_t \hat{A}_t,\, clip\left(\beta_t, 1-\epsilon, 1+\epsilon\right)\hat{A}_t\right\}\right] \quad (4)$$

$\hat{A}_t = R_t - \hat{V}_\phi(s_t)$ is the estimated advantage, where $R_t = \sum_{\tau=t}^{T}$ is the local reward adding global reward, $\hat{V}_\phi$ is the estimated value function of state S_t with parameters ϕ, $\beta_t = \frac{\pi_\theta(a_t|s_t)}{\pi_{\theta_{old}}(a_t|s_t)}$ is the ratio of the probability under the new and old policies, δ is TD residual, λ and ε are hyper-parameters.

4 Experiments and Results

4.1 Datasets

We evaluate TopKG and baselines on two datasets. To verify the ability to guide the user to the target topic in multi-turn of dialogue, we use our proposed dataset TGConv, which is extracted from ConvAI2 (Dinan et al., 2020) and is distinguished into "easy-to-reach/hard-to-reach" targets with the method in section 3.2. ConvAI2 is a chit-chat dataset based on the PersonaChat for NIPS 2018 competition, which contains high-quality open-domain dialogues, including diverse topics. In addition, one-turn dialogue is a special case of multi-turn, therefore we also conduct our evaluation on a next-turn target-oriented dataset OTTers(ood)[2] (Sevegnani et al., 2021). OTTers requires the agent pro-actively generate an "bridging" utterance to approach the target, which is consistent with the input and output of the task on TGConv. The statistics of the two datasets are presented in Table2.

Dataset	Split	#Conv.	Avg. #Utter.	Avg. #Word.	Avg. #Entity.	Avg. #Coh.
OTTers	Train	2034	3.0	9.47	2.86	0.45
	Valid	1152	3.0	9.56	2.95	0.45
	Test	1130	3.0	9.19	2.80	0.44
TGConv	Train	15197	8.35	12.60	2.89	0.32
	Valid	2681	7.96	12.29	2.85	0.31
	Test	1000	8.97	12.47	2.91	0.32

Table 2: Dataset statistics. Avg.#Utter., #Word., #Entity., #Coh. denotes the average number of utterances, words, entities, semantic similarity per dialogue, utterance, utterance, utterance.

4.2 Baselines

We select four baselines in end-to-end (GPT-2, MultiGen) and pipeline style (DKRN, CKC), respectively. The first baseline is **GPT-2** (Radford et al., 2019). Next, we test the recent **Multi-Gen** (Ji et al., 2020), extends GPT-2 with multi-hop reasoning on commonsense knowledge graphs. The third baseline is **DKRN** (Qin et al., 2020), which builds a dialog graph from the corpus for topic transition. The last baseline is **CKC** (Zhong et al., 2021), the state-of-the-art approach using ConceptNet for this task. In addition, DKRN and CKC are retrieval models. Here we replace the retriever with the generator in our paper.

4.3 Metrics

Local-Evaluation To evaluate models' performance in generating next-turn response, we firstly perform automatic evaluation using commonly adopted text generation metrics, including CIDEr (Vedantam et al., 2015), ROUGE-L (Lin, 2004) and METEOR (Banerjee and Lavie, 2005). However, we report the full BLEU score[3] (Papineni et al., 2002) that accounts for the overlap across 1-4 ngrams instead of only 4-grams (BLEU-4). In addition, we use hits@K ratio to measure the number of relevant entities correctly predicted by the

[2]OTTers have different train-dev-test (in-domain and out-of-domain) splits, we choose out-of-domain(ood) split. The ood split resembles a zero-shot scenario, where the model has to generate a shift between two topics it has never been fine-tuned on.

[3]SacreBLEU (Post, 2018) provides hassle-free computation of shareable, comparable, and reproducible BLEU scores.

	BLEU$^{1-4}$	METEOR	ROUGE-L	CIDEr	hits@1	hits@3
GPT2	11.58	10.26	17.67	13.75	4.39	15.79
MultiGen	13.57	12.51	26.27	15.48	6.58	20.51
DKRN	12.86	11.90	21.52	14.33	4.91	17.72
CKC	13.34	11.65	24.77	14.46	6.87	21.89
TopKG	**15.35***	**13.41***	27.16	**17.18***	7.78	**22.06***
w/o global plan	14.89	12.89	26.99	16.22	7.45	21.14
w/ small graph(K=5,H=3)	13.24	10.65	25.53	15.62	6.77	21.22
w/ large graph(K=20,H=6)	15.24	11.65	**27.53**	16.62	**7.79**	21.63

Table 3: Automatic evaluation of next-turn response generation on OTTers. Numbers marked with * indicate that the improvement is statistically significant compared with the best baseline(t-test with p-value < 0.05).

	Easy Target			Hard Target		
	Succ.(%)	Turns	Coh.	Succ.(%)	Turns	Coh.
GPT2	22.3	2.86	0.23	17.3	**2.94**	0.21
MultiGen	26.7	**2.55**	0.21	19.6	7.31	0.24
DKRN	38.6	4.24	0.33	21.7	7.19	0.31
CKC	41.9	4.08	**0.35**	24.8	6.88	0.33
TopKG	**48.9***	3.95	0.31	**27.3***	4.96	**0.33**
w/o global plan	35.4	4.51	0.32	21.3	7.18	0.32

Table 4: Automatic evaluation of global guiding on TGConv. Note that our task requirement is to reach the target smoothly and fast. “Coh.” and “Turns” not the higher / lower the better.

	Easy Target		Hard Target	
	G-Coh.	Effect.	G-Coh.	Effect.
GPT2	1.13	1.20	1.13	0.86
MultiGen	1.24	1.29	1.17	1.13
DKRN	1.26	1.23	1.19	1.18
CKC	**1.53**	1.31	1.23	1.16
TopKG	1.51	**1.67**	**1.37**	**1.48**
w/o global plan	1.42	1.34	1.24	1.13
kappa	0.45	0.55	0.51	0.58

Table 5: Comparison of human evaluation metric Coherence and Effectiveness results on self-chat dialogues among our model and baselines. The agreement among the annotators is measured by the Fleiss’s kappa. The agreement ratio kappa in [0.41, 0.6] denotes the moderate agreement.

local-model, out of the K most important entities identified in the target references.

Global Evaluation To evaluate models’ performance in guiding the dialog to global target, as existing works (Qin et al., 2020; Zhong et al., 2021) do, we use a simulator to simulate multi-turn dialog. "Succ." automatically measures the success rate of generating the global target word within 8 turns. "Turns" indicates the average turns of all dialogs which successfully generate the global target word. "Coherence (Coh.)" automatically measures the contextual semantic similarity between the last utterance in the context and generated response.

As automatic metrics are widely criticized due to their lack of correlation with human judgments (Novikova et al., 2017; Reiter, 2018), we also employ two dialogue-level metrics by human evaluation: "Global-Coherence (G-Coh.)" manually examine whether the whole dialog is logical, fluent, and target-oriented. "Effectiveness (Effect.)" measures whether the target is efficiently reached. 100 dialogs are generated by simulation for each model. Three annotators score the generated dialogs rating in {0, 1, 2}, higher is better.

4.4 Results

Local Evaluation Table 3 shows the next-turn results on OTTers. GPT2 performs poorly on the one-turn transition. The multi-hop reasoning component added by the MultiGen and our TopKG leads to substantial improvements. TopKG achieves substantially better performance than all baselines on OTTers. This advantage indicates that TopKG successfully improves the performance of the existing local-model, which aims at the next-turn target.

Global Evaluation As shown in Table 4, on cases with "easy-to-reach" targets, end-to-end models such as GPT and MultiGen have fewer turns.

Start: music → dance Easy Target: Travel	
Context	**A:** My favorite type of music is from the south. **B:** Good ! I really like dancing, especially in the moonlight.
CKC	**A:** I watch sci fi. mainly, I read supernatural romances. **B:** Do you read ? My sister says fantasy novels are the best. **A:** I have a library in my house , I like to read books. **B:** I prefer historical story to romance. **A:** Cool! I love to wander and travel the world. **Path: romance → novel → book → story → travel**
TOPKG	**A:** I watch sci fi. mainly, I read supernatural romances. **B:** Reading fantasy novel and poetry are my true loves ! **A:** Awesome! I like adventures , been in sales all my life . **B:** I love to travel and see different places. Do you? **Path: romance → novel → adventure → travel**

Start: game → fun Hard Target: Revolutionary	
Context	**A:** Nice . i'm playing some card games with my family. **B:** That sounds like fun. i like wine and dancing too!
CKC	**A:** You sound like a fun person! **B:** What music do you like? **A:** Classical. how about you, what kind do you like? **B:** I like all music. ....(get trapped)
TOPKG	**A:** And what games are you into? **B:** Action games, do you? **A:** Not much into those. i like the rebel. **B:** Revolutionary. is that your favorite? **Path: game → action → rebel → revolutionary**

Table 6: Case study from self-play simulations on TGConv. In the left easy target case, TopKG generates responses similar in quality to CKC but plans a diverse path to the target. In the right hard target case, CKC gets trapped, but TopKG successfully reaches the target.

We notice that they tend to directly generate an utterance containing the target, despite that the utterances are of low quality in human evaluation. This may be due to that they are designed without global view. However, our TopKG has a higher success rate and higher efficiency in manual evaluation benefiting from the global planning.

In cases with "hard-to-reach" targets, GPT, which does not rely on KG, can also directly generate responses, and its performance is similar to that of "easy-to-reach" cases. For all KG-based methods, the performance significantly degrades on "hard-to-reach" targets, but our TopKG still exceeds all baselines. The ablation discussion below demonstrates the contribution of our global planning. Furthermore, our generated responses' average contextual "Semantic Similarity(Coh.)" is similar to the golden similarity in Table 2, which shows that our TopKG effectively learns the semantic patterns in the corpora. We also found that KG methods (CKC and TopKG) outperform the other models, which verifies the benefits of using KG in global target-oriented dialog.

4.5 Ablation Studies

We perform ablation studies for TopKG to better analyze the main components' relative contributions. The results are shown in Tables 3, 4, 5.

Does the global planning work? To prove the contribution of proposed global planning, we replace the global planning (w/o global plan) with a 2-hop neighbors graph (based on context entities), which results in the most significant performance drop in multi-turn evaluation. In contrast, the drop in the next-turn evaluation is not noticeable. The main reason is that the target often can be found in two-hop neighbors on the graph in a next-turn dialogue. This verifies the contribution of global KG planning to global target-oriented dialog.

How much graph information we need? We also explore the number of neighbors needed for initializing the G_{global} graph's nodes in two aspects (refer in Algorithm1): the maximum number of hops H, and the number of neighboring nodes in the h_{th} hop (denoted as K). Contrary to our expectations, expanding the average size of the knowledge graphs from 1000 nodes to 2000 did not improve the $hits@K$ ratio, as shown in the last row of Table 3. Therefore, the final version of TopKG adopts the global planning with K = 10, H = 3.

4.6 Case study

In the case study, we compare our TopKG with CKC, the most competitive baseline. In the left case of "Easy Target" in Table 6, TopKG and CKC followed different KG paths. In the first path followed by CKC, the novel indicates books, and the following two keywords are the topics of the books. In the second path followed by TopKG, the novel is an adjective, adventure is novel, and travel is one kind of adventure. In such easy cases, although the best existing method works well, TopKG can further explore diverse paths based on reinforcement learning. In the right case of "Hard Target" in Table 6, CKC gets trapped and fail to reach the goal. However, TopKG still successfully guides the dialog to the goal with effective global planning.

5 Conclusion and Future Work

We propose effectively guiding the target-oriented dialog towards a global target with global planning on KG. We first design a novel method to automatically select target-oriented samples from the chit-chat corpus by identifying KG reasoning paths

throughout the dialog. We train a reinforcement learning model with a selected high-quality corpus that can guide a GPT-2 based response generation model to reach a global target word by global planning on ConceptNet. Automatic and human evaluations show that our method exceeds the baselines from both local and global views, and global planning provides a significant contribution. We will explore to balance the coherence and number of turns in global planning in future work.

6 Acknowledgements

This work was supported by National Natural Science Foundation of China (62272340, 61876128, 61876129, 61976154), State Key Laboratory of Communication Content Cognition (Grant No.A32003).

References

Satanjeev Banerjee and Alon Lavie. 2005. Meteor: An automatic metric for mt evaluation with improved correlation with human judgments. In *Proceedings of the acl workshop on intrinsic and extrinsic evaluation measures for machine translation and/or summarization*, pages 65–72.

Antoine Bordes, Nicolas Usunier, Alberto Garcia-Duran, Jason Weston, and Oksana Yakhnenko. 2013. Translating embeddings for modeling multi-relational data. *Advances in neural information processing systems*, 26.

Paweł Budzianowski, Tsung-Hsien Wen, Bo-Hsiang Tseng, Iñigo Casanueva, Stefan Ultes, Osman Ramadan, and Milica Gasic. 2018. Multiwoz-a large-scale multi-domain wizard-of-oz dataset for task-oriented dialogue modelling. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 5016–5026.

Fabio Clarizia, Francesco Colace, Marco Lombardi, Francesco Pascale, and Domenico Santaniello. 2018. Chatbot: An education support system for student. In *International Symposium on Cyberspace Safety and Security*, pages 291–302. Springer.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*.

Emily Dinan, Varvara Logacheva, Valentin Malykh, Alexander Miller, Kurt Shuster, Jack Urbanek, Douwe Kiela, Arthur Szlam, Iulian Serban, Ryan Lowe, et al. 2020. The second conversational intelligence challenge (convai2). In *The NeurIPS'18 Competition*, pages 187–208. Springer.

Minlie Huang, Xiaoyan Zhu, and Jianfeng Gao. 2020. Challenges in building intelligent open-domain dialog systems. *ACM Transactions on Information Systems (TOIS)*, 38(3):1–32.

Haozhe Ji, Pei Ke, Shaohan Huang, Furu Wei, Xiaoyan Zhu, and Minlie Huang. 2020. Language generation with multi-hop reasoning on commonsense knowledge graph. *arXiv preprint arXiv:2009.11692*.

Dongyeop Kang, Anusha Balakrishnan, Pararth Shah, Paul Crook, Y-Lan Boureau, and Jason Weston. 2019. Recommendation as a communication game: Self-supervised bot-play for goal-oriented dialogue. *arXiv preprint arXiv:1909.03922*.

Thomas N Kipf and Max Welling. 2016. Semi-supervised classification with graph convolutional networks. *arXiv preprint arXiv:1609.02907*.

Raymond Li, Samira Ebrahimi Kahou, Hannes Schulz, Vincent Michalski, Laurent Charlin, and Chris Pal. 2018. Towards deep conversational recommendations. *Advances in neural information processing systems*, 31.

Chin-Yew Lin. 2004. Rouge: A package for automatic evaluation of summaries. In *Text summarization branches out*, pages 74–81.

Wenchang Ma, Ryuichi Takanobu, and Minlie Huang. 2021. Cr-walker: Tree-structured graph reasoning and dialog acts for conversational recommendation. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 1839–1851. ACL.

Jekaterina Novikova, Ondřej Dušek, Amanda Cercas Curry, and Verena Rieser. 2017. Why we need new evaluation metrics for nlg. *arXiv preprint arXiv:1707.06875*.

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. Bleu: a method for automatic evaluation of machine translation. In *Proceedings of the 40th annual meeting of the Association for Computational Linguistics*, pages 311–318.

Jeffrey Pennington, Richard Socher, and Christopher D Manning. 2014. Glove: Global vectors for word representation. In *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*, pages 1532–1543.

Matt Post. 2018. A call for clarity in reporting bleu scores. *arXiv preprint arXiv:1804.08771*.

Jinghui Qin, Zheng Ye, Jianheng Tang, and Xiaodan Liang. 2020. Dynamic knowledge routing network for target-guided open-domain conversation. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34 Issue 05, pages 8657–8664.

Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, Ilya Sutskever, et al. 2019. Language models are unsupervised multitask learners. *OpenAI blog*, 1(8):9.

Abhinav Rastogi, Xiaoxue Zang, Srinivas Sunkara, Raghav Gupta, and Pranav Khaitan. 2020. Towards scalable multi-domain conversational agents: The schema-guided dialogue dataset. In *Proceedings of the AAAI Conference on Artificial Intelligence*, pages 8689–8696.

Ehud Reiter. 2018. A structured review of the validity of bleu. *Computational Linguistics*, 44(3):393–401.

John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. 2017. Proximal policy optimization algorithms. *arXiv preprint arXiv:1707.06347*.

Karin Sevegnani, David M Howcroft, Ioannis Konstas, and Verena Rieser. 2021. Otters: One-turn topic transitions for open-domain dialogue. *arXiv preprint arXiv:2105.13710*.

Ashish Sharma, Adam S Miner, David C Atkins, and Tim Althoff. 2020. A computational approach to understanding empathy expressed in text-based mental health support. *arXiv preprint arXiv:2009.08441*.

Robyn Speer, Joshua Chin, and Catherine Havasi. 2017. Conceptnet 5.5: An open multilingual graph of general knowledge. In *Thirty-first AAAI conference on artificial intelligence*.

Jianheng Tang, Tiancheng Zhao, Chenyan Xiong, Xiaodan Liang, Eric Xing, and Zhiting Hu. 2019. Target-guided open-domain conversation. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 5624–5634.

Ramakrishna Vedantam, C Lawrence Zitnick, and Devi Parikh. 2015. Cider: Consensus-based image description evaluation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 4566–4575.

PeiFeng Wang, Jonathan Zamora, Junfeng Liu, Filip Ilievski, Muhao Chen, and Xiang Ren. 2021. Contextualized scene imagination for generative commonsense reasoning. *arXiv preprint arXiv:2112.06318*.

Zeqiu Wu, Michel Galley, Chris Brockett, Yizhe Zhang, Xiang Gao, Chris Quirk, Rik Koncel-Kedziorski, Jianfeng Gao, Hannaneh Hajishirzi, Mari Ostendorf, et al. 2020. A controllable model of grounded response generation. *arXiv preprint arXiv:2005.00613*.

Jun Xu, Haifeng Wang, Zheng-Yu Niu, Hua Wu, Wanxiang Che, and Ting Liu. 2020a. Conversational graph grounded policy learning for open-domain conversation generation. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 1835–1845.

Jun Xu, Haifeng Wang, Zhengyu Niu, Hua Wu, and Wanxiang Che. 2020b. Knowledge graph grounded goal planning for open-domain conversation generation. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34 Issue 05, pages 9338–9345.

Yunyi Yang, Yunhao Li, and Xiaojun Quan. 2020. Ubar: Towards fully end-to-end task-oriented dialog systems with gpt-2. *arXiv preprint arXiv:2012.03539*.

Yizhe Zhang, Siqi Sun, Michel Galley, Yen-Chun Chen, Chris Brockett, Xiang Gao, Jianfeng Gao, Jingjing Liu, and Bill Dolan. 2019a. Dialogpt: Large-scale generative pre-training for conversational response generation. *arXiv preprint arXiv:1911.00536*.

Yizhe Zhang, Siqi Sun, Michel Galley, Yen-Chun Chen, Chris Brockett, Xiang Gao, Jianfeng Gao, Jingjing Liu, and Bill Dolan. 2019b. Dialogpt: Large-scale generative pre-training for conversational response generation. *arXiv preprint arXiv:1911.00536*.

Xueliang Zhao, Wei Wu, Can Xu, Chongyang Tao, Dongyan Zhao, and Rui Yan. 2020. Knowledge-grounded dialogue generation with pre-trained language models. *arXiv preprint arXiv:2010.08824*.

Peixiang Zhong, Yong Liu, Hao Wang, and Chunyan Miao. 2021. Keyword-guided neural conversational model. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 35 Issue 16, pages 14568–14576.

Jinfeng Zhou, Bo Wang, Ruifang He, and Yuexian Hou. 2021. CRFR: Improving conversational recommender systems via flexible fragments reasoning on knowledge graphs. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 4324–4334.

A Appendix

A.1 Discriminator

Discrimination measures the different between the positive/negative examples we extracted. In experiments, we verify the significant difference between the conversational behaviour of selected positive and negative examples by accurately classifying positive and negative examples by a discriminator.

	Accuracy
BERT-small	76.8%
BERT-base	78.1%

Table 7: The accuracy of the discriminator classification

According to the result of the discriminator and data analysis, we found that some negative samples may contain positive segments (e.g., the first few utterances in a negative sample can generate a short reasoning path), so false negative may be a problem in training the discriminator.

A.2 Fail cases

Here are some cases of TopKG failures. It can be seen that the failure can be roughly divided into

two situations. One is that in the face of some uncommon words, even if there is a correct path, the generative model can only generate sentences with similar semantics. The other is that the path is very long and it is difficult to generate replies that contain relevant words.

Start: dye Hard Target: volley	
Context	**A:** hi i dye my hair 4 times a year. **B:** do you live in the sky ? i do . in the clouds.
TOPKG	**A:** i dye it dark brown **B:** do you also like the martant? it is my favorite. **A:** yes i like it because it is on the ground **B:** do you have a favorite field? ....(can not reached) **Path: dye-mordant-ground-volley**

Start: mowing Hard Target: ethicist	
Context	**A:** i also enjoy lawn mowing on sunny days . **B:** i enjoy green living and just put solar panels on roof
TOPKG	**A:** that's awesome. solar roofs are awesome. **B:** yes they're. are you good at math **A:** i am excellent at it. mornings mornings are like night. **B:** i like the night **A:** it is my favorite part **B:** what is your favorite part about it? ....(can not mention related words and can not reached) **Path: mowing-morning-forenoon--part-particular-ethicist**

Table 8: Some failure cases

A.3 Implementation Details

The default parameter settings across all experiments are as follows. We select conceptnet-5.7.0 and glove.6B.300d for initialization. The discount factor γ is 0.99 and the advantage discount factor is 0.95. Our policy reasoning model is trained for 200 epochs using Adam optimization with the learning rate of 5e-4 for actor network and 1e-3 for critic network with the batch size of 64. 256 action-state pairs to rollout for trajectory collection per epoch, 4 steps of gradient descent to perform on each batch, capacity of the replay buffer is 8.

Extractive summarisation for German-language data: a text-level approach with discourse features

Freya Hewett [1,2]
[1]AI & Society Lab
Humboldt Institute for Internet and Society
Berlin, Germany
firstname.lastname@hiig.de

Manfred Stede [2]
[2]Applied Computational Linguistics
University of Potsdam
Potsdam, Germany
lastname@uni-potsdam.de

Abstract

We examine the link between facets of Rhetorical Structure Theory (RST) and the selection of content for extractive summarisation, for German-language texts. For this purpose, we produce a set of extractive summaries for a dataset of German-language newspaper commentaries, a corpus which already has several layers of annotation. We provide an in-depth analysis of the connection between summary sentences and several RST-based features and transfer these insights to various automated summarisation models. Our results show that RST features are informative for the task of extractive summarisation, particularly nuclearity and relations at sentence-level.

1 Introduction

Extractive summarisation involves directly using select phrases and sentences from a text as a summary, which still remains a strong method for producing summaries despite its simple nature (Huang et al., 2020). In this study, we examine the link between facets of Rhetorical Structure Theory (RST) and the selection of content for extractive summarisation. RST is a framework which posits that every part of coherent text has a role and a function and represents texts in a hierarchical tree structure (Taboada and Mann, 2006). The RST framework consists of the segmentation of the text into Elementary Discourse Units (EDUs), which are then grouped into bigger segments – depending on the way they relate to each other – which forms the hierarchical structure of the text. The relations between these segments are defined, and a nucleus-satellite status is given. This nucleus-satellite allocation stems from the observation that within the majority of relations which hold between two segments, one segment tends to be more important than the other. This notion of importance seems to have an inherent link to summarisation and various studies have examined this link (see Section 2). In this study we look at the role that these various aspects play in extractive summarisation for German-language texts and transfer these insights to different types of models. In this context, we introduce a new dataset of extractive summaries, analyse the RST-based features of these, collating the best features proposed over the last 20+ years and introducing a new document-based sentence embedding and use these in both linear and nuclear models. Our results compare favourably to those from closely related work on English (Louis et al., 2010). We elaborate on this study and other related work in Section 2, before describing our German-language extractive summarisation dataset in Section 3. In Section 4, we provide a detailed analysis of various RST-based features and examine how these features are distributed in the summaries. We describe our summarisation models and the results of our experiments in Section 5 before discussing the results and providing some concluding remarks in Sections 6 and 7.

2 Related work

Since the RST framework was proposed in the late 1980s (Mann and Thompson, 1988), various studies have examined the link between discourse structure and summarisation, building on even earlier concepts of text-level structural analysis ('macrostructures') (van Dijk and Kintsch, 1983). A study from Marcu (1999) built on ideas proposed by Ono et al. (1994) and empirically analysed the link between the nuclearity aspect of RST and extractive summarisation on a small sample of five texts. Marcu (1999) concluded that there is a strong correlation between nuclei and what readers perceive to be the most important units in a given text, and implemented an automatic summarisation system using RST trees. Louis et al. (2010) conducted an analysis of the RST-DT corpus which contains newspaper articles with various annotation layers including RST and other discourse structures. Louis et al.

Proceedings of the 29th International Conference on Computational Linguistics, pages 756–765
October 12–17, 2022.

(2010) specifically evaluated a subset of the corpus which consists of 150 extractive summaries, for which annotators were asked to select the most important EDUs. They analysed structural features derived from RST trees, such as the depth of a segment, in comparison to other discourse (e.g. the semantics of PDTB relations) and non-discourse features and found that the structural RST features were most useful for automatically selecting sentences for summaries. Zhong et al. (2020) used automatically parsed discourse features for the task of sentence deletion for text simplification, a task which shares many similarities with extractive summarisation. They used RST-based features – such as local nuclearity, relations and the position of the sentence in the tree – in both a linear model and a neural model.

Other studies instead looked specifically at the role that discourse segmentation plays in extractive summarisation. Li et al. (2016) automatically parsed a corpus of English language newspaper articles and compared the RST segmentation to manually annotated *summary content units*. Molina Villegas et al. (2011) also looked at discourse segmentation of Spanish-language texts for the task of sentence compression, which they consider to be sentence-level summarisation.

Neural approaches have also been proposed for combining discourse structure with English-language summarisation. Xu et al. (2020) created a BERT-based model which takes discourse units as input (as opposed to sentences) and also encodes automatically parsed RST trees in a CNN layer in the network, which resulted in an improvement to the state-of-the-art for English-language extractive summarisation. Liu and Chen (2019) experimented with three different neural architectures and compare using sentences as input to discourse segments for the task of extractive summarisation. The models which use discourse segments score higher in an automatic evaluation.

The link between summarisation and discourse structure is further explored by Xiao et al. (2021). They hypothesise that the link between the two may be bidirectional and analyse if summarisation can inform discourse structure by generating RST trees from the inner layers of a Transformer-based summarisation model.

3 Data

As far as we know, the Potsdam Commentary Corpus (PCC) is the only German-language dataset with RST annotations (Stede, 2004). The corpus consists of 176 commentaries from a German regional daily newspaper, the *Märkische Allgemeine Zeitung*. The commentaries have various layers of annotation, including part-of-speech, syntax and discourse structure but does not yet have summaries; we therefore created these ourselves. Each extractive summary consists of 3 key sentences; we use the term 'key sentence' throughout this paper to refer to the sentences selected to be part of the summaries. The commentaries have an average length of 11.4 sentences, so the summaries represent ca. 26% of the average length (in sentences). The extractive summaries available for the RST-DT corpus, used for example in the study by Louis et al. (2010), are of a similar size (the square root of the number of EDUs for each text) which makes the corpora easier to compare. We make the summaries publicly available[1], and a sample annotated text can be seen in Table 1.

3.1 Annotation task

To produce these summaries, annotators were asked to choose 3 sentences from each text that represent the core of the text and rank these in order of importance. The task description specified that 'important' in this context refers to the suitability of the sentence for a summary. Any anaphoric elements should be 'mentally' replaced by that what they are referring to. This annotation task is inherently subjective as the notion of importance can be interpreted in different ways. We discuss this point in more detail in the following Sections (3.2 and 3.3). On average, a sentence has 14 tokens in our corpus, which rises to 18 tokens on average for the annotated sentences. These three key sentences consist of 5 EDUs, on average.

3.2 Inter-annotator agreement

For a sample of 30 texts, two sets of annotations were gathered which resulted in a Cohen's Kappa score of .32 when comparing the three selected key sentences (without the ranking, (Cohen, 1960)). Similar (yet lower) scores have been reported in other annotation studies of this kind: .28 when selecting 20% of 'salient sentences from each comment which summarize it' from online debates

[1] https://github.com/fhewett/pcc-summaries

(Sanchan et al., 2017), an average of .30 when selecting 8% to 16% of 'summary sentences that are informative and can preserve discussion flow' from meeting transcripts (Liu and Liu, 2008), and .23 when selecting sentences with relevant 'nuggets' (clauses containing one verb and one noun that are semantically 'important in the context of the given topic') in different genres of German text (Benikova et al., 2016).

The relatively low agreement score reflects the subjective nature of our task; however, all texts have at least one sentence in common in both sets of annotations (again, disregarding the ranking). The annotations for these 30 texts were harmonised using a scoring system: the highest ranked sentence was equivalent to 3 points, the second to 2 and the third to 1. The sentence with the most points was then deemed the highest ranked sentence in the harmonised annotation, and so on. Any tied scores were resolved by randomly selecting one of the sentences in the tie.

Some questions were raised by annotators: In some texts there are sometimes multiple sentences which contain the same information, and there is no clear way to rank them. The task description also mentions anaphoric entities, but for phrases such as 'that's why' (as in segment no. 9 in Table 1), it is not clear if these should also be replaced by what they are referring to. The genre of a newspaper commentary also poses a specific challenge: is the journalist's opinion or are the objective facts more important? Due to these comments, we adapted the task description to specify that the chosen three sentences should ideally contain both the objective information as well as the opinion of the journalist, and in cases where this is not possible, then the objective information should be prioritised. If there are multiple sentences which contain highly similar content, then the first sentence should be chosen. As this task description however did not lead to an improved inter-annotator agreement we gave the annotators the first (shorter) task description for the remaining texts. The remaining texts were annotated by four annotators (including the first author), who also all annotated part of the subset of the 30.

3.3 Semantic similarity

We also evaluated the inter-annotator agreement using ROUGE scores (Lin, 2004) and word mover's distance (WMD; Kusner et al. 2015). This is due to an observation that the annotated sentences were often semantically similar and contained the same information even when they were not identical. ROUGE measures the overlapping n-grams in a source and reference summary. We used all three sentences in their ranked order as a summary and used one set of annotations as the reference summary and the other set as the source summary. This resulted in a ROUGE-1 F1 score of .587, which compares to a baseline of .380, which was calculated by comparing one set of annotations to a summary consisting of 3 randomly selected sentences from a text. In a recent survey on automatic summarisation, the ROUGE-1 scores for the most recent extractive systems were between .338 and .414 (Fabbri et al., 2021). WMD calculates the shortest distance between the word embeddings of two sentences or sets of multiple sentences. Again, we used all three sentences combined together and the WMD between the two annotators' summaries was .512, compared to a baseline of three random sentences with a distance of .827. These scores show that whilst the aforementioned IAA score is relatively low when using a strict Kappa measure, the two sets of annotations do have semantic similarities, which indicates commonalities in annotators' choices of content.

4 Analysis

To investigate a potential link between RST and extractive summarisation for German-language texts, we examine various RST-based features of the sentences that were chosen in the annotation task. This builds on the studies by Louis et al. (2010) and Zhong et al. (2020) (as outlined in Section 2).

4.1 Non-discourse features

We first look at non-RST features; the average length of sentences and the (relative) position of sentences in the whole text. The average length of non-summary sentences is 13 tokens, whereas the key sentences have an average length of 18 tokens. Figure 2 shows at what position in the text the key sentences occur.

4.2 Local nuclearity

Local nuclearity refers to the nuclear-satellite relationship within a relation. We analyse the nuclear-satellite relationship at the sentence level. For example, the first sentence in Table 1 consists of 3 EDUs and is the satellite of an interpretation, as can

1	Election results which both candidates are happy with –	
2	what a rare occurrence!	
3	But that was the case yesterday evening for both mayoral candidates.	
4	***Elisabeth Herzog-von der Heide (SPD) was happy about the voting***,	1
5	***that resulted in more than 60 percent for her,***	1
6	***and Hans-Jürgen Akuloff (PDS) was pleased that he brought in the best ever result for the PDS so far.***	1
7	*Of course, it was even easier this time, as there was only one opponent.*	
8	*Luckenwalde has a broad range of diverse parties.*	
9	That's why SPD or PDS supporters should not get ahead of themselves.	
10	**Yesterday's outcome is the result of the two candidates,**	2
11	**after they eliminated the third candidate in the first round.**	2
12	Nothing more and nothing less.	
13	After the first round of Skat, the second round of Mau Mau was won by Herzog, the Queen of Hearts.	
14	It's neither a surprising nor a particularly phenomenal result but it's most definitely a clear outcome.	
15	***There was no real tension anyway,***	3
16	***after the outcome was pretty much decided in the first round of voting,***	3
17	***which has simply been confirmed in this final round.***	3

Table 1: Example text from the dataset (text ID: maz-14654). The sentences **in bold** are those that have been selected for the summary in our gold annotation. The sentences *in italics* are those chosen by our best model (FFN). The column on the left contains the ID for the segments, the column on the right the rank given to the sentence by the annotator.

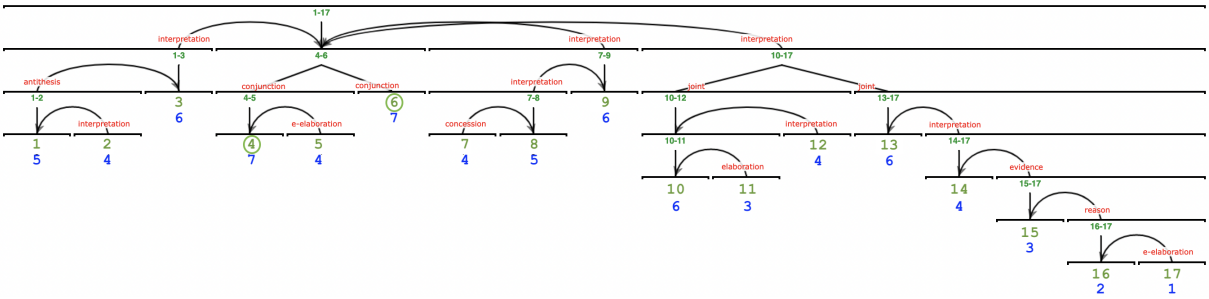

Figure 1: The RST tree for the example text (maz-14654, created using RSTWeb (Zeldes, 2016)). The circled EDUs are the most-nuclear, the blue numbers are the depth scores.

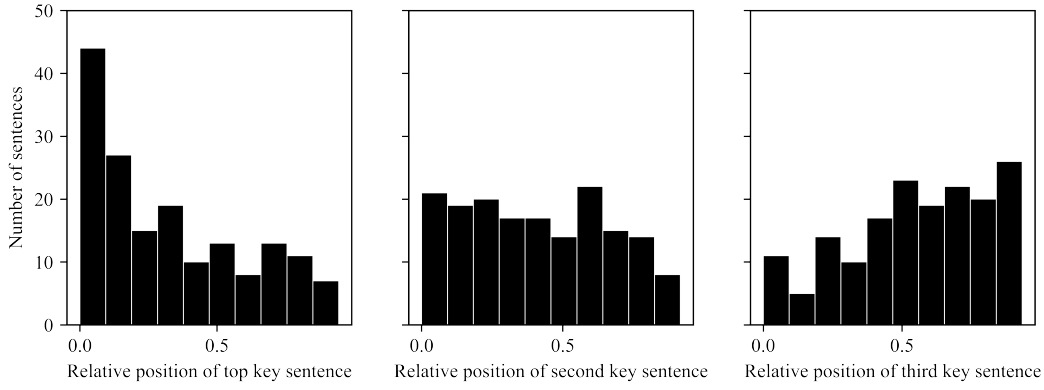

Figure 2: The positions of the three sentences in each text.

be seen in Figure 1. This analysis reflects RST's 'deletion test': the nuclearity assignment can be considered to be correct if once the satellites are deleted, the remaining EDUs still convey the main message(s) of the text (Mann and Thompson, 1988). Although simply considering the sentence-level nuclearity status does not take the rest of the tree structure into account, we feel it is still a fruitful aspect to analyse due to the "strong composionality criterion" or strong nuclearity principle (Marcu, 2000): if a relation holds between two spans then the relation also holds between the nuclei of these spans, therefore the assignment of nuclearity status at a terminal level is fundamental as the importance is propagated up the tree.

Of all the sentences annotated as being key, 70% were nuclear. This compares to 61% of sentences in the whole dataset.

4.3 Global nuclearity

We use the term global nuclearity to refer to nuclearity with respect to the whole tree and not just at a local level. Huber et al. (2021) examine the loss of information that a binary nuclearity assignment can lead to and highlight this loss with reference to downstream tasks such as summarisation. To counteract this, we look at the **depth score** of EDUs, which allows for a more nuanced approach to nuclearity, and the **most nuclear** EDUs, which takes the whole tree into account. We use the term **depth** to refer to what Marcu (1999) terms importance score, which has also been implemented in more recent studies (Louis et al. 2010, who also use the term depth; Huber et al. 2021).

This score is calculated using the nuclearity of units and their relation to other nodes in the RST tree. We adapt the original equation (by removing the part that refers to parenthetical units, as these do not feature in our corpus) for calculating the importance score *s(u,D,d)* of a unit *u* in a discourse tree *D* with depth *d* as follows:

$$s(u, D, d) = \begin{cases} d, & \text{if } u \in prom(D) \\ max(s(u, & \\ C(D), d-1)), & otherwise \end{cases} \quad (1)$$

where C(D) refers to the child subtree and *prom(D)* the promotion set of a node: if the node is a leaf node then the promotion set is simply the leaf itself, if the node is internal then the promotion set is the union of the salient units of its immediate nuclear children. The score is simply the depth in the tree where the leaf units first occur in a promotion set. For example, we can see in Figure 1 that the depth score of the fourth EDU is 7, as it belongs to the promotion set of the root node, and the whole discourse tree has a total depth of 7: if we follow the nuclear links from the root node, we get the promotion set of EDUs 4 and 6. The depth score of EDU 3 is 6, as it belongs to the promotion set of the subtree which is one level below the root. We refer to the units with the highest depth scores as **most-nuclear** as proposed by Mann and Thompson (1988): the most-nuclear nodes can be determined by following nuclear links from the root node to the (leaf) EDU nodes. Depth scores are normalised (as otherwise the length of the text would influence the scores). As the depth score is calculated on an EDU-level (and not sentence-level), we define the depth score for a sentence as the maximum of the depth scores of the EDUs that it contains. In the same vein, a sentence is considered most-nuclear if it contains a most-nuclear EDU.

25% of the key sentences are also most-nuclear EDUs or contain a most-nuclear EDU. This corresponds to about 38% of all most-nuclear EDUs in the corpus.

65% of the texts have at least one key sentence which contains or corresponds to a most-nuclear EDU. When comparing the sentences with the three highest depth scores (taking the maximum score for sentences which contain more than one segment) to the three key sentences, 46% match.

4.4 Relations

We examine the relations of the annotated sentences: in the example in Table 1, the first two EDUs constitute one sentence and therefore the relation that we consider in our analysis would be *Antithesis* (satellite, cf. Figure 1). We consider relations to be important in the context of this analysis because annotation guidelines for different RST corpora pre-define the nuclearity assignment for relations: for example, for the relation *Purpose*, the underlying goal of the activity is the satellite and the activity itself is the nucleus (RST-DT guidelines (Carlson and Marcu, 2001); PCC guidelines (Stede et al., 2017)). Stede (2008) states that this can be problematic: by pre-defining the 'activity itself' as the nucleus, this does not allow for any flexibility for the case where actually the 'goal of

the activity is more important'. Marcu (1998) suggests that to improve his proposed nuclearity-based summarisation method, one should also "exploit the semantics of rhetorical relations", citing the relation *Exemplification* as an example of a relation where even the nuclei are probably not relevant for a summary. We therefore also consider nuclearity in connection with relations as they are intertwined with each other and could give some quantitative evidence for the problems highlighted here.

Figure 3 shows which relations the key sentences have, with and without the nuclearity assignment. As the selected sentences represent 26% of total sentences, any relation ratio above this is above average. *Evaluation-s* nuclei, *Background* satellites, and *Evidence* nuclei feature the most in the key sentences.

5 Summarisation models

We use these RST features and non-discourse features in linear and neural models to predict which sentences should be kept for an extractive summary. We frame extractive summarisation as a binary classification task, predicting whether each sentence should be included in a summary or not. Our dataset consists of 167 texts and 1894 sentences; we use 30% of the dataset as a test set.

5.1 Features

We combine and adapt the feature sets used in the studies by Louis et al. (2010) and Zhong et al. (2020) and also introduce a new way of creating sentence embeddings. We also perform a detailed feature ablation to see which RST-based feature(s) are most useful for extractive summarisation. The features we use in comparison to related work can be seen in Table 2. As Zhong et al. (2020) looked specifically at content selection in the context of simplification, we do not use some of their features such as *topic* or *readability score*, as they do not seem relevant for the task of summarisation. We do not use features related to specific words (*content words, topic words*) as we anticipate that this information is implicitly encoded in the embeddings we use. Our focus is on RST-based features and a more fine-grained analysis of these on an individual basis, and so we do not include PDTB related features. The features *nucleus-satellite penalty* and *promotion score* are used to measure global nuclearity: we decide to solely use *depth* as it was shown to achieve the better results than the other two varia-

Features	**Louis et al. (2010)**	**Zhong et al. (2020)**	**Present study**
Non-discourse			
Sentence length	X		X
Document length (in sentences and tokens)		X	
Topics		X	
Position of sentence	X	X	X
Sentence embeddings		∼	X
Embeddings with document context			X
Readability scores		X	
Content words, topic signature words	X		
Discourse			
PDTB connectives	X	X	
Local nuclearity		X	X
Depth	X	∼	X
Nucleus-satellite penalty	X		
Promotion score	X		
Relations		∼	X
Most-nuclear			X

Table 2: Features used in the present study and in related work. The ∼ signifies that the features are not fully identical.

Model	**F1**	**R**	**P**	**Acc.**
Baseline	.225	.188	.281	.658
Louis et al. (2010)	.442	.344	.619	.789
LR	.488	.583	.422	.677
FFN	.506	.588	.448	.692
bi-LSTM	.440	.635	.341	.729

Table 3: Results. Highest scoring feature set for each model. We report F1 scores for the minority class (key sentence). R, P and Acc. stand for recall, precision and accuracy respectively.

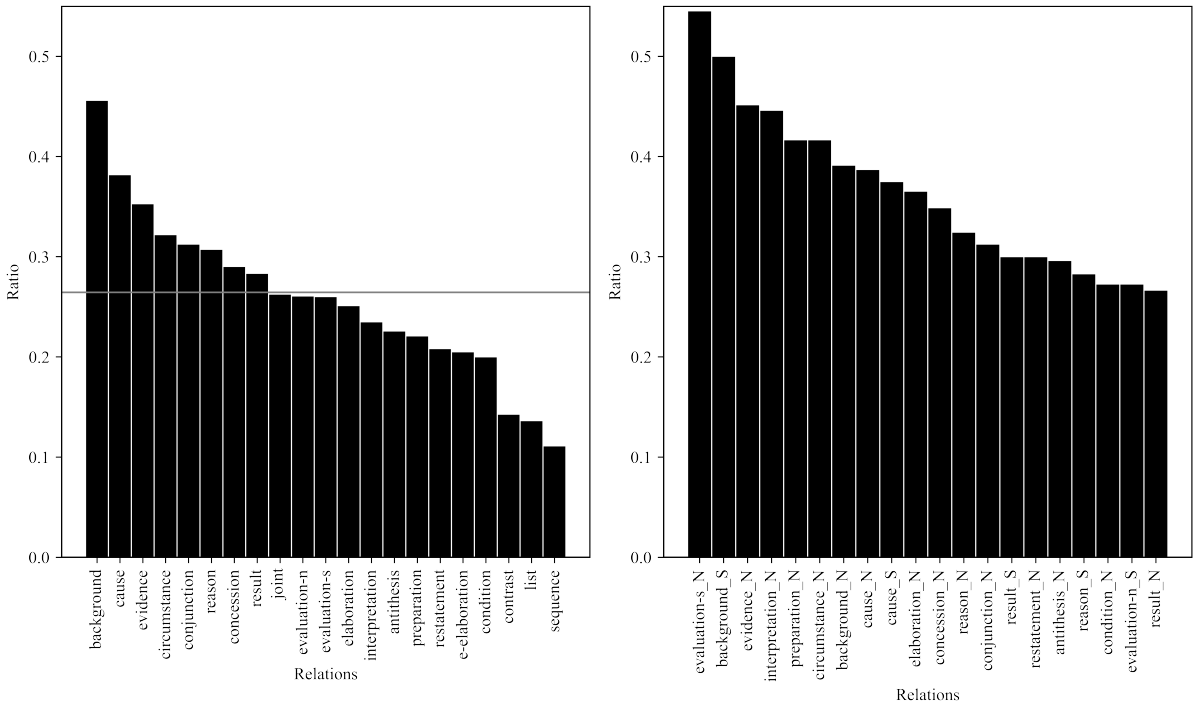

Figure 3: The ratio of relations of the selected key sentences out of the total amount of relations. Any relation ratio above 26% (the line on the graph) is above average. The figure on the right hand side shows the relations and nuclearity assignment of the key sentences; any ratio below 26% is excluded for better visibility. Relations occurring less than a total of 10 times are excluded.

tions in the context of summarisation (Marcu, 1999) and has also been used in more recent work (Huber et al., 2021). Zhong et al. (2020) use the term depth to refer to the level that the sentence occurs at in the tree and do not encode any extra information regarding promotion sets or nuclearity in this feature (in our example in Figure 1, the EDUs 1 and 2 would both have a score of 3).

The **discourse features** are the nuclearity of the sentence, the relation that the sentence belongs to, the depth of the sentence, and whether the sentence is most-nuclear or not. If the sentence contains more than one segment, we take the maximum depth score or most-nuclear score of the segments. For example, the first sentence in Table 1 would have the following features: *satellite*, *interpretation*, *depth score 6/7*, *most-nuclearity 0*. The **non-discourse features** are the length of the sentence (in tokens) and the relative position of the sentence. We also use **sentence embeddings** and an adapted variation of these, in an attempt to incorporate more document-level information. We use S-BERT sentence embeddings which are created by taking the mean of token embeddings of the input (which is usually a single sentence) from an adapted BERT model trained on multi-lingual paraphrase data (Reimers and Gurevych, 2020). We also adapt this by taking the whole text as input and then simply taking the average of the token embeddings between the sentence boundaries, thus producing an embedding for each sentence, instead of just one embedding for the whole text. The intuition is that these sentence embeddings may capture more document-level information by having seen an even larger context whilst producing the token embeddings.

5.2 Setup

As a baseline, we select the first, middle and last sentence to be in a summary. We implement a Logistic Regression model[2], a feed-forward neural network (with two hidden layers; FFN), and a bidirectional LSTM model (with one layer). For the bi-LSTM model, each text is fed as a sequence, so the input dimensions are *batch size, text length, feature dimension*. For all models, we weight the classes to counteract the class imbalance, which is skewed towards the 'non-summary sentence' class.

[2]Whilst we did also experiment with other non-neural methods, we only report on Logistic Regression as it performed the best and is also directly comparable to the work by Louis et al. (2010).

In total we have four discourse features, two non-discourse features and two types of sentence embeddings. We run models with all possible combinations of features. We experiment with different hyper-parameters, these can be found in the Appendix and we make our code publicly available.[3]

5.3 Results

The results of our experiments can be seen in Table 3. The FFN achieved the best F1 score with nuclearity, relations, most-nuclearity, sentence length and position, and sentence embeddings. The Logistic Regression (LR) model also had the best F1 score with this feature set, minus sentence length and position. The bi-LSTM achieved the best F1 score with nuclearity, relations and sentence position, in combination with sentence embeddings. All best-performing models have nuclearity, relations and sentence embeddings in common as features.

In Table 4, the best performing combinations of features can be seen as well as the performance of individual features; whilst our proposed sentence embeddings with additional document context are not the individual feature with the worst F1 score, they generally do not improve results greatly across all types of models. We report F1 score as we are interested in the minority class (key sentences).

The summary produced by the FFN model for our example text can be seen in italics Table 1. Overall, the summary reads well and includes two of the gold key sentences (shown in bold). However, the sentence which corresponds to segment number 8 does not make much sense without the following sentence (which corresponds to segment 9). The ROUGE-1 F1 score for our test set as predicted by the FFN model is .690. The ROUGE-1 recall score is .612 as compared to .479 reported by Louis et al. (2010).

Features	**F1**
Top 3 combinations	
Nuclearity, relations, most-nuclearity, sentence embeddings	.488
Nuclearity, relations, sentence embeddings	.486
Nuclearity, relations, most-nuclearity, sent. length	.486
Ranked individual features	
Sentence length	.462
Sentence embeddings	.452
Local nuclearity	.425
Depth	.391
Embeddings with document context	.385
Position	.376
Relations	.368
Most-nuclearity	.302

Table 4: Feature ablation for the Logistic Regression model. We show the best 3 combinations, as well as the individual features ranked in descending order according to the F1 score achieved when using said feature as the sole input.

6 Discussion

The results of the analysis, particularly Section 4.4 and Figure 3, show that simply looking at nuclearity in isolation may not be sufficient for some downstream tasks, as we have *Background* satellites and *Cause* satellites featuring heavily in the summaries, for example. This is reflected in the results: the best combination of features for both the FFN and LR model is nuclearity (local and global, with most-nuclearity measuring global nuclearity) in combination with relations (and sentence embeddings). The bi-LSTM model and our sentence embeddings with additional document context both perform worse than other models and features: this goes against our intuition that more text-level context is beneficial for such a task as extractive summarisation. We leave it to future work to examine if these embeddings could work in other settings, perhaps with larger datasets or pre-trained in a different manner. Simple models (such as LR) perform well, which suggests that RST features (particularly nuclearity) are strong indicators of importance. Combinations of RST features perform even better than sentence embeddings on their own. As can be seen in Table 3, our F1 scores (and our ROUGE scores, see Section 5.3) are in fact higher than those reported in (Louis et al., 2010). Whilst our recall is higher, our precision and overall accuracy are slightly lower. It is also worth noting that our results are on commentaries, a type of argumentative text, and so the results are not directly comparable (Louis et al. work with the RST-DT, articles from the Wall Street Journal); we leave this to future work to investigate more thoroughly.

[3] https://github.com/fhewett/rst-features

7 Conclusion

In this study we have introduced a new set of extractive summaries for the Potsdam Commentary Corpus; texts which have already been annotated with various linguistic features such as discourse structure, co-reference and syntax. We have shown the connection between RST-based features and sentences chosen for extractive summaries and have transferred these to various models. Our feature ablation experiments could provide useful insights for research on RST parsers for specific downstream tasks: by finding the aspect of RST which is most useful for a task such as summarisation, the parsers can be streamlined and will have less room for error. For example, one of our top models has sentence-level nuclearity and relation information, showing that in this context, the additional RST tree structure is potentially not necessary. We hope that our dataset will enable research on the link between summarisation and other linguistic features. Our analysis using manually annotated discourse structure also provides the necessary evidence to foster research on automatically parsed discourse structure and downstream tasks, such as summarisation.

Acknowledgements

Many thanks to the anonymous reviewers for their valuable comments. This research is funded by the German Federal Ministry of Education and Research (BMBF).

References

Darina Benikova, Margot Mieskes, Christian M Meyer, and Iryna Gurevych. 2016. Bridging the gap between extractive and abstractive summaries: Creation and evaluation of coherent extracts from heterogeneous sources. In *Proceedings of COLING 2016, the 26th International Conference on Computational Linguistics: Technical Papers*, pages 1039–1050.

Lynn Carlson and Daniel Marcu. 2001. Discourse Tagging Reference Manual.

Jacob Cohen. 1960. A coefficient of agreement for nominal scales. *Educational and Psychological Measurement*, 20(1):37–46.

Alexander R. Fabbri, Wojciech Kryściński, Bryan McCann, Caiming Xiong, Richard Socher, and Dragomir Radev. 2021. SummEval: Re-evaluating Summarization Evaluation. In *Transactions of the Association for Computational Linguistics*, volume 9, pages 391–409.

Dandan Huang, Leyang Cui, Sen Yang, Guangsheng Bao, Kun Wang, Jun Xie, and Yue Zhang. 2020. What Have We Achieved on Text Summarization? In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 446–469.

Patrick Huber, Wen Xiao, and Giuseppe Carenini. 2021. W-RST: Towards a Weighted RST-style Discourse Framework. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 3908–3918.

Matt J Kusner, Yu Sun, Nicholas I Kolkin, and Kilian Q Weinberger. 2015. From Word Embeddings To Document Distances. In *Proceedings of the 32nd International Conference on International Conference on Machine Learning - Volume 37*, ICML '15, pages 957–966.

Junyi Jessy Li, Kapil Thadani, and Amanda Stent. 2016. The Role of Discourse Units in Near-Extractive Summarization. In *Proceedings of the 17th Annual Meeting of the Special Interest Group on Discourse and Dialogue*, pages 137–147, Los Angeles. Association for Computational Linguistics.

Chin-Yew Lin. 2004. ROUGE: A Package for Automatic Evaluation of Summaries. volume Text Summarization Branches Out, pages 74–81. Association for Computational Linguistics.

Fei Liu and Yang Liu. 2008. What are meeting summaries?: an analysis of human extractive summaries in meeting corpus. In *Proceedings of the 9th SIGdial Workshop on Discourse and Dialogue - SIGdial '08*, page 80, Columbus, Ohio. Association for Computational Linguistics.

Zhengyuan Liu and Nancy Chen. 2019. Exploiting Discourse-Level Segmentation for Extractive Summarization. In *Proceedings of the 2nd Workshop on New Frontiers in Summarization*, pages 116–121, Hong Kong, China. Association for Computational Linguistics.

Annie Louis, Aravind Joshi, and Ani Nenkova. 2010. Discourse indicators for content selection in summarization. In *Proceedings of the SIGDIAL 2010 Conference*, pages 147–156, Tokyo, Japan. Association for Computational Linguistics.

William C. Mann and Sandra A. Thompson. 1988. Rhetorical Structure Theory: Toward a functional theory of text organization. *Text - Interdisciplinary Journal for the Study of Discourse*, 8(3):243–281.

Daniel Marcu. 1998. To build text summaries of high quality, nuclearity is not sufficient. In *Working Notes of the AAAI-98 Spring Symposium on Intelligent Text Summarization*, pages 1–8.

Daniel Marcu. 1999. Discourse trees are good indicators of importance in text. In *Advances in Automatic Text Summarization*, pages 123–136. The MIT Press.

Daniel Marcu. 2000. *The theory and practice of discourse parsing and summarization*. A Bradford book. MIT Press, Cambridge, Mass.

Alejandro Molina Villegas, Juan-Manuel Torres-Moreno, Eric Sanjuan, Iria da Cunha, Gerardo Sierra, and Patricia Velázquez-Morales. 2011. Discourse Segmentation for Sentence Compression. In *MICAI 2011: Advances in Artificial Intelligence*, pages 316–327.

Kenji Ono, Kazuo Sumita, and Seiji Miike. 1994. Abstract generation based on rhetorical structure extraction. In *Proceedings of the 15th Conference on Computational Linguistics*, volume 1, pages 344–348, Kyoto, Japan. Association for Computational Linguistics.

Nils Reimers and Iryna Gurevych. 2020. Making monolingual sentence embeddings multilingual using knowledge distillation. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 4512–4525, Online. Association for Computational Linguistics.

Nattapong Sanchan, Ahmet Aker, and Kalina Bontcheva. 2017. Gold Standard Online Debates Summaries and First Experiments Towards Automatic Summarization of Online Debate Data. In *Computational Linguistics and Intelligent Text Processing - 18th International Conference, CICLing 2017, Budapest, Hungary, April 17-23, 2017, Revised Selected Papers, Part II*, volume 10762 of *Lecture Notes in Computer Science*, pages 495–505.

Manfred Stede. 2004. The Potsdam commentary corpus. In *Proceedings of the 2004 ACL Workshop on Discourse Annotation - DiscAnnotation '04*, pages 96–102, Barcelona, Spain. Association for Computational Linguistics.

Manfred Stede. 2008. RST revisited: Disentangling nuclearity. In Cathrine Fabricius-Hansen and Wiebke Ramm, editors, *Studies in Language Companion Series*, volume 98, pages 33–58. John Benjamins Publishing Company, Amsterdam.

Manfred Stede, Maite Taboada, and Debopam Das. 2017. Annotation Guidelines for Rhetorical Structure.

Maite Taboada and William C. Mann. 2006. Rhetorical Structure Theory: looking back and moving ahead. *Discourse Studies*, 8(3):423–459.

Teun A. van Dijk and Walter Kintsch. 1983. *Strategies of Discourse Comprehension*. Academic Press.

Wen Xiao, Patrick Huber, and Giuseppe Carenini. 2021. Predicting discourse trees from transformer-based neural summarizers. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 4139–4152, Online. Association for Computational Linguistics.

Jiacheng Xu, Zhe Gan, Yu Cheng, and Jingjing Liu. 2020. Discourse-Aware Neural Extractive Text Summarization. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 5021–5031, Online. Association for Computational Linguistics.

Amir Zeldes. 2016. rstWeb - a browser-based annotation interface for rhetorical structure theory and discourse relations. In *Proceedings of NAACL-HLT 2016 System Demonstrations*, pages 1–5, San Diego, CA.

Yang Zhong, Chao Jiang, Wei Xu, and Junyi Jessy Li. 2020. Discourse Level Factors for Sentence Deletion in Text Simplification. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34(5), pages 9709–9716.

A Appendix

We experimented with the values in brackets, the values in bold are the parameters used in the final models.

Hyper-parameters for FFN
Batch size [8,12,16,32,**64**]
Epochs [5,10,**15**]
Learning rate [**0.5**,0.8,1]
Class weights [(0.3, 2), **(0.3, 2.5)**, (0.3, 6)]

Hyper-parameters for LSTM
Batch size [**8**,12,16,32,64]
Epochs [5,10,**15**]
Learning rate [0.5,**0.8**,1]
Class weights [(0.3, 4), **(0.3, 5.5)**]

End-to-End Neural Bridging Resolution

Hideo Kobayashi[1], **Yufang Hou**[2] and **Vincent Ng**[1]
[1] Human Language Technology Research Institute, University of Texas at Dallas, USA
[2] IBM Research Europe, Ireland
{hideo,vince}@hlt.utdallas.edu
yhou@ie.ibm.com

Abstract

The state of bridging resolution research is rather unsatisfactory: not only are state-of-the-art resolvers evaluated in unrealistic settings, but the neural models underlying these resolvers are weaker than those used for entity coreference resolution. In light of these problems, we evaluate bridging resolvers in an end-to-end setting, strengthen them with better encoders, and attempt to gain a better understanding of them via perturbation experiments and a manual analysis of their outputs.

1 Introduction

Bridging was used by Clark (1975) to refer to non-identity relations between anaphoric noun phrases (i.e., **bridging anaphors**) and their antecedents. In Example 1, "**ruling – party members**" is a bridging anaphor and its antecedent is "*Japan*".

(1) Yet another political scandal is racking *Japan*. But this time it's hurting opposition as well as **ruling - party members**.

Bridging resolution is the task of identifying bridging anaphors and linking them to their antecedents. Many tasks can benefit from bridging resolution, such as textual entailment (Mirkin et al., 2010) and question answering (Tseng et al., 2021).

Bridging resolution is arguably less studied but more challenging that entity coreference resolution, the task of determining which entity mentions refer to the same entity in the real world. Specifically, while linguistic constraints on coreference exist at the grammatical (e.g., gender and number agreement), syntactic (e.g., c-command), and semantic (e.g., semantic type agreement) levels that can be used to filter candidate antecedents, such constraints are largely absent for bridging resolution. For instance, a singular bridging anaphor (e.g., "the book") can refer to a plural antecedent (e.g., "books"), and bridging relations can be formed from mentions with different entity types (e.g., "the house" and "the window"). In fact, while many coreference relations can be identified via string matching facilities, it is not uncommon for bridging relations to be identified using background knowledge and/or sophisticated inference mechanisms. The complexity of bridging resolution is further complicated by the lack of a large corpus annotated with bridging relations: while the most extensively-used coreference-annotated corpus, OntoNotes, contain more than 2000 documents, two of the most commonly-used corpora for bridging resolution, ISNotes and BASHI, each contains only 50 documents taken from OntoNotes.

The current state of bridging resolution research is rather unsatisfactory. State-of-the-art bridging resolvers are typically evaluated in unrealistic settings: in the *bridging anaphora resolution* task, the goal is to identify the antecedent of a given bridging anaphor; and in the *full bridging resolution* task, the goal is to first identify the bridging anaphors given a set of gold mentions and then resolve each anaphor to its antecedent (Yu and Poesio, 2020). While such unrealistic settings have been considered unacceptable for evaluating entity coreference resolvers for more than a decade, they are still extensively used to evaluate bridging resolvers nowadays simply because the *end-to-end* setting, where a resolver needs to identify bridging relations given a raw document, is perceived to be overly challenging. Worse still, while bridging resolution is more challenging than coreference resolution, models for bridging resolution are less sophisticated than those for coreference resolution. For instance, while SpanBERT, a version of BERT specifically pre-trained to identify text spans (Joshi et al., 2020), has been used successfully as an encoder in span-based entity coreference models, the state-of-the-art neural bridging resolver developed by Yu and Poesio (2020) simply uses a bidirectional LSTM to encode the input document.

Our goal in this paper is to gain a better un-

Proceedings of the 29th International Conference on Computational Linguistics, pages 766–778
October 12–17, 2022.

derstanding of the state of the art in bridging resolution. First, we conduct a systematic evaluation of bridging resolvers in an *end-to-end* setting. In particular, we focus on evaluating three state-of-the-art approaches, including a rule-based approach by Rösiger et al. (2018), a neural approach by Yu and Poesio (2020), and a hybrid rule-based and learning-based approach by Kobayashi and Ng (2021), showing how each of them can be extended so that they can be applied in an end-to-end setting. Next, we strengthen Yu and Poesio's (2020) neural model by replacing the biLSTM encoder it uses with stronger encoders such as BERT (Devlin et al., 2019) and SpanBERT (Joshi et al., 2020). These experiments can help us determine whether the commonsense knowledge encoded in these pre-trained language models can be profitably exploited for bridging resolution, which could be important given that state-of-the-art bridging resolvers are trained on small annotated corpora. Further, to better understand the extent to which a bridging resolver relies on certain words/phrases in the input, we conduct perturbation experiments. Finally, to complement the quantitative analysis in our experiments, we conduct a qualitative analysis of the outputs produced by our best-performing resolver.

Our contributions in this paper are three-fold. First, our experiments reveal that end-to-end bridging resolution is not as challenging as typically perceived: for the most part, end-to-end bridging resolution lags behind its "gold mention" counterpart by less than 3 points in absolute F-score. These results provide suggestive evidence that time is ripe for abandoning unrealistic evaluations of bridging resolvers. Second, we establish baseline results for end-to-end bridging resolution against which future work can be compared on two commonly-used referential bridging corpora, ISNotes and BASHI. While our evaluations have largely focused on the end-to-end setting, for comparison purposes we also present evaluation results in the "gold mention" setting, in which our strongest models achieve state-of-the-art results on ISNotes and BASHI.

2 Related Work

Many previous computational studies on bridging have focused on one of the two sub-tasks of bridging resolution, namely *bridging anaphora recognition* and *bridging anaphora resolution* (see Kobayashi and Ng (2020) for a comprehensive overview of this area of research). Bridging anaphora recognition has been tackled as part of the information status (IS) classification problem (Rahman and Ng, 2011, 2012; Hou et al., 2013; Hou, 2020b). Recall that the goal of IS classification is to assign an IS to each discourse entity that indicates how these entities are referred to in a text (Prince, 1981; Nissim et al., 2004; Markert et al., 2012): an entity is *old* if it is coreferent with an entity that has been mentioned before, *new* if it is introduced into the discourse for the first time and is not known to the hearer before, and *mediated* if it has not been introduced in the discourse but can be inferred from previously mentioned entities. Bridging anaphors are a type of *mediated* entities that are discourse-new but hearer-old. Bridging anaphora resolution, on the other hand, focuses on selecting antecedents for bridging anaphors (Poesio et al., 2004; Pandit et al., 2020; Hou, 2020a). There are a few works tackling full bridging resolution (i.e., recognizing bridging anaphors and linking them to the antecedents), ranging from rule-based approaches (Hou et al., 2014; Rösiger et al., 2018), to machine learning-based approaches (Hou et al., 2018; Yu and Poesio, 2020) and hybrid methods (Kobayashi and Ng, 2021; Kobayashi et al., 2022). However, these resolvers all assume that gold mentions are given, which hinders the application of bridging resolution in downstream tasks.

In contrast, we focus on end-to-end bridging resolution. Note that some recent attempts have been made in this direction. For example, Hou's (2020a) approach to bridging anaphora resolution does not require gold mentions when constructing the list of antecedent candidates; nevertheless, it still needs gold bridging anaphora information. In addition, while Hou (2021) proposes an end-to-end neural approach to the related tasks of IS classification and bridging anaphora recognition, it has not been extended to bridging resolution. More recently, in the Bridging track of the CODI-CRAC shared task on Anaphora, Bridging, and Discourse Deixis in Dialogue in 2021 (Khosla et al., 2021) and 2022 (Yu et al., 2022), the participants built resolvers for performing end-to-end bridging resolution in dialogue in the "Predicted" phase (Kim et al., 2021; Kobayashi et al., 2021; Li et al., 2022).

3 State-of-the-Art Approaches

Existing approaches to bridging resolution can be broadly divided into rule-based approaches, learning-based approaches, and hybrid approaches.

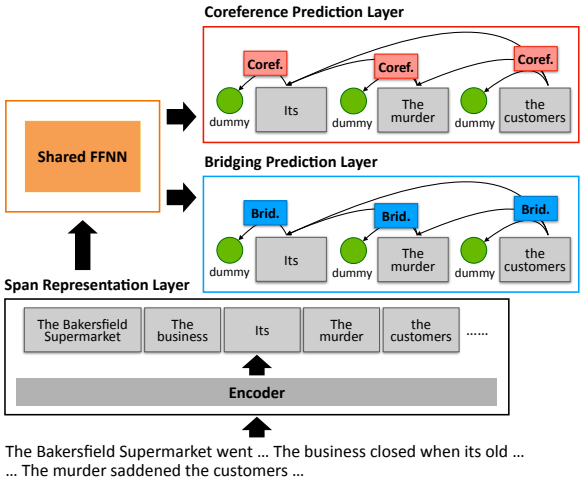

Figure 1: The MTL framework for bridging resolution.

In this section, we overview the state-of-the-art approach in each of these three categories, as we will extend them to the end-to-end setting in Section 4.

3.1 Yu and Poesio's (2020) Model

Yu and Poesio's (Y&P) approach is a state-of-the-art learning-based approach to bridging resolution. Their model is a span-based neural model that takes as input a document D represented as a sequence of word tokens and the associated set of gold mentions, and performs joint bridging resolution and coreference resolution, which we define below, in a multi-task learning (MTL) framework.

The bridging resolution task aims to find a bridging antecedent b_i for each span i in D. The set of possible values for b_i is $\mathcal{B}(i) = \{1, ..., i-1, \epsilon\}$, the preceding spans or a dummy antecedent (if the mention underlying i is not a bridging anaphor). Y&P define the following scoring function:

$$s_b(i,j) = \begin{cases} 0 & j = \epsilon \\ s_a(i,j) & j \neq \epsilon \end{cases} \quad (1)$$

where $s_a(i,j)$ is a pairwise score computed over i and a preceding span j suggesting their likelihood of having a bridging link. The antecedent of i is predicted to be $y_b^* = \arg\max_{y_b \in \mathcal{B}(i)} s_b(i, y_b)$.

The entity coreference task aims to find a coreference antecedent c_i for each span i based on a scoring function s_c that is defined analogously as the s_b function in the bridging resolution task.

Figure 1 illustrates the structure of MTL framework, which we describe in detail below.

Span Representation Layer To encode the tokens and the surrounding contexts of a gold mention, Y&P use a bidirectional LSTM (Hochreiter and Schmidhuber, 1997) that takes as input BERT and GloVe embeddings. They define $\mathbf{g}_i$, the representation of span i, as $[\mathbf{x}_{start(i)}; \mathbf{x}_{end(i)}; \mathbf{x}_{head(i)}; \phi_i]$, where $\mathbf{x}_{start(i)}$ and $\mathbf{x}_{end(i)}$ are the hidden vectors of the start and end tokens of i, $\mathbf{x}_{head(i)}$ is an attention-based head vector and ϕ_i is a span width feature embedding.

Bridging Prediction Layer To predict bridging links, Y&P first calculate the pairwise score between spans i and j as follows:

$$s_a(i,j) = \text{FFNN}_b([\mathbf{g}_i; \mathbf{g}_j; \mathbf{g}_i \circ \mathbf{g}_j; \psi_{ij}]) \quad (2)$$

where $\text{FFNN}_b(\cdot)$ represents a standard feedforward neural network, and $\circ$ denotes element-wise multiplication. This pairwise score includes $\mathbf{g}_i \circ \mathbf{g}_j$, which encodes the similarity of i and j, and ψ_{ij}, which denotes the distance between them.

Coreference Prediction Layer To predict coreference links, Y&P calculate the pairwise score that is defined analogously as in Equation 2 using another FFNN, FFNN_c. The model shares the first few hidden layers of FFNN_b and FFNN_c as well as the span representations.

The loss function is the weighted sum of the losses of the bridging task (L_b) and the coreference task (L_c). L_b and L_c are defined as the negative marginal log-likelihood of all correct bridging antecedents and coreference antecedents, respectively. The weights associated with the losses are tuned using grid search to maximize the average bridging resolution F-scores on development data.

3.2 Rösiger et al.'s (2018) Approach

Rösiger et al.'s approach, which builds upon the rules designed by Hou et al. (2014), is by far the best-performing rule-based approach to bridging resolution. These rules are shown in Appendix A. When evaluating on BASHI, all nine rules are applied, but when evaluating on ISNotes, only the first eight are used. The reason is that the last rule aims to resolve comparative anaphors, which are not annotated in ISNotes. Each rule is composed of an "anaphor" condition and an "antecedent" condition. When two mentions satisfy the two conditions of a rule, they will be extracted as a bridging pair.

3.3 Kobayashi and Ng's (2021) Approach

Motivated by the observation that Rösiger et al.'s rule-based approach and Y&P's MTL approach are complementary rather than competing, Kobayashi

and Ng (K&N) propose a hybrid approach to bridging resolution that combines these two approaches in a pipeline fashion. Given a document, they first use the rules to extract the bridging pairs and then use Y&P's neural model to resolve all and only those mentions that are not resolved by the rules.

From a modeling perspective, however, K&N's approach is not particularly elegant, as there are two models (i.e., the rules and the neural models are still separate). Consequently, we propose a variant of the hybrid approach where we *integrate* the rules into the MTL model. Recall that each rule posits a bridging link when the anaphor and antecedent conditions are both satisfied. To incorporate these predictions into the MTL model, we first define a rule score function $r(i, j)$ whose value is the precision of the rule that posits a bridging link between spans i and j. This rule score function is incorporated into Equation 1 as follows:

$$s_{b'}(i,j) = \begin{cases} 0 & j = \epsilon \\ s_b(i,j) + \alpha r(i,j) & j \neq \epsilon \end{cases} \quad (3)$$

where α is a positive constant that controls the impact of the rule information on s'_b. The smaller α is, the less impact rule information has on s'_b. $s_{b'}$ is then used as the bridging score function when ranking the candidate antecedents of span i. Note that (1) if no rule posits i and j as bridging, $r(i, j)$ is 0; (2) rule precision is computed on the training set; and (3) α is tuned on the development set.

We will henceforth refer to the original K&N approach as H_1 and our proposed variant as H_2.

4 End-to-End Models

Next, we show how to create end-to-end versions of the three approaches described in Section 3.

4.1 Yu and Poesio's (2020) Model

We present two approaches to create end-to-end versions of Y&P's model.

Joint approach. The first approach learns mention boundaries jointly with bridging and coreference resolution. Specifically, following Joshi et al. (2019), for each document, we enumerate all possible intra-sentence spans of up to length L_m. We compute a score s_m for each span i that indicates i's likelihood of being a mention.

$$s_m(i) = \text{FFNN}_m(\mathbf{g}_i) \quad (4)$$

where FFNN_m is a feedforward neural network used to calculate mention scores. Using these scores, the model prunes candidate spans and retains only the top N spans for further processing in order to maintain computational tractability. These scores are then incorporated into the bridging score function in Equation 1 as additional terms:

$$s_b(i,j) = \begin{cases} 0 & j = \epsilon \\ s_m(i) + s_m(j) + s_a(i,j) & j \neq \epsilon \end{cases} \quad (5)$$

We incorporate these scores into the coreference score function $s_c(i, j)$ in a similar manner.

Pipeline approach. In this approach, we do not make any changes to Y&P's model. Rather, during testing, we first apply a mention extractor to extract mentions and then employ Y&P's model from Section 3 to resolve mentions.

Next, we describe our mention extractor. For ISNotes, we use Hou's (2021) mention extraction model, which has achieved state-of-the-art results on ISNotes and outperformed Yu et al.'s (2020) neural mention extractor. For BASHI, we use all the noun phrases extracted from the automatic parse trees that are obtained using Stanford CoreNLP (Manning et al., 2014).[1] The reason is that in BASHI gold mentions are not annotated, and bridging links are annotated over the noun phrases extracted from gold parse trees.[2]

Using BERT and SpanBERT as encoders. We strengthen Y&P's model by replacing its biLSTM encoder with Transformer-based encoders, including BERT and SpanBERT, the latter of which has been successfully applied to entity coreference resolution. To do so, we follow Joshi et al. (2019) and replace the LSTM-based encoder in Y&P's model, which takes frozen BERT and Glove embeddings as input, with BERT or SpanBERT, which corresponds to the "Encoder" component in Figure 1. We adopt the independent version of Joshi et al. (2019), where an input document is split into non-overlapping segments of up to length L_s.

4.2 Rösiger et al.'s (2018) Approach

While the rules were designed to operate on gold mentions, they can be applied to mentions extracted

[1] These mention extractors achieve F-scores of 92.1 (when extracting gold mentions in ISNotes) and 92.0 (when extracting gold noun phrases in BASHI).

[2] In preliminary experiments, we applied Hou's (2021) mention extractor to extract mentions in BASHI, but the results were poorer than those obtained using noun phrases extracted from automatic parse trees.

Corpora	Docs	Tokens	Mentions	Anaphors
ISNotes	50	40,292	10,980	663
BASHI	50	57,709	18,561	452

Table 1: Statistics on ISNotes and BASHI.

using one of the mention extractors described in Section 4.1 (i.e., Hou's (2021) mention extractor for ISNotes and the noun phrases extracted from system parse trees for BASHI) with just one caveat. Specifically, Rösiger et al. use gold annotations (i.e., gold POS tags, gold parse trees, and gold entity types) when computing the information needed by the rules. To make the rules applicable in an end-to-end setting, we use Stanford CoreNLP to provide automatic constituency and dependency parse trees and spaCy (Honnibal and Montani, 2017) to provide automatic POS tags and entity types.

4.3 Kobayashi and Ng's (2021) Approach

Now that we have end-to-end versions of Rösiger et al.'s rule-based approach and Y&P's MTL approach, we can simply use them to create an end-to-end version of K&N's hybrid approach.

5 Evaluation

5.1 Experimental Setup

Corpora. Since we focus on *anaphoric referential bridging resolution*, which corresponds to "referential bridging" in Rösiger et al. (2018) where bridging anaphors are truly anaphoric and bridging relations are context-dependent[3], we employ two widely used English referential bridging corpora: ISNotes (Markert et al., 2012) and BASHI (The Bridging Anaphors Hand-annotated Inventory) (Rösiger, 2018), both of which are composed of different sets of 50 WSJ articles in OntoNotes with anaphoric referential bridging annotations. Table 1 shows statistics on these corpora. We perform 5-fold cross validation (70% for model training, 10% for development, and 20% for testing).

Evaluation setting. We evaluate bridging resolvers in the end-to-end setting, meaning that they extract bridging relations given a raw document.

Evaluation metrics. Bridging results are reported in terms of precision (P), recall (R), and F-score (F) for recognition and resolution. For completeness we also report the results of entity coreference. Entity coreference results are expressed in terms of the CoNLL score (Pradhan et al., 2014), which is the unweighted average of the F-scores provided by three coreference evaluation metrics, MUC (Vilain et al., 1995), B^3 (Bagga and Baldwin, 1998), and $CEAF_e$ (Luo, 2005).

[3]We excluded ARRAU (Uryupina et al., 2020) from our evaluation because most bridging links in ARRAU are non-anaphoric referential bridging pairs (e.g., *Europe-Spain*), which Rösiger et al. (2018) refer to as *lexical bridging*.

Model	Bridging	
	Recognition	Resolution
	ISNotes	
Rösiger et al. (2018)	25.6	17.5
Our re-implementation	28.1	18.1
	BASHI	
Rösiger et al. (2018)	27.2	14
Our re-implementation	28.5	14.1

Table 2: Comparison of Rösiger et al's (2018) resolver and our re-implementation on ISNotes and BASHI.

Implementation details. For the MTL model, we extend a publicly-available implementation of Y&P's resolver[4] so that it can operate in an end-to-end setting. The BERT and SpanBERT encoders we use are BERT_{LARGE} and SpanBERT_{LARGE}, and BERT_{LARGE} is used to obtain BERT embeddings with segment length 512. We set all parameters to the ones reported in Yu and Poesio (2020) except (1) the task learning rate, which is searched out of $\{1 \times 10^{-4}, 2 \times 10^{-4}, 3 \times 10^{-4}, 4 \times 10^{-4}\}$ and is decayed linearly; and (2) the learning rates for BERT and SpanBERT, which are searched out of $\{1 \times 10^{-5}, 2 \times 10^{-5}, 3 \times 10^{-5}\}$ and are decayed linearly. Each document is split into segments of length 384. We generate all spans of length up to 15 and prune these candidate spans by retaining the top 30%. For training, we use document-sized mini-batches and train models for up to 1600 epochs for both ISNotes and BASHI. α, the weight parameter associated with the rule score, is searched out of {0.1, 0.5, 1.0, 10, 100, 200, 300}.

For the rule resolver, while conceptually we can extend Rösiger et al.'s resolver so that it can operate in an end-to-end setting, the way they structured their code has made it non-trivial to do so. Consequently, we (1) re-implement their resolver, which operates on gold mentions; (2) extend it so that it operates on automatically extracted mentions, as described in Section 4.2; and (3) report rule-based results using this duplicated resolver. As we can see in Table 2, our re-implementation outperforms the original resolver in terms of recognition and resolution F-scores when evaluated on both ISNotes and BASHI in the gold-mention setting.

[4]https://github.com/juntaoy/dali-bridging

	Model	Bridging						Coref.	Bridging						Coref.
		Recognition			Resolution			Res.	Recognition			Resolution			Res.
		P	R	F	P	R	F	CoNLL	P	R	F	P	R	F	CoNLL
		ISNotes							**BASHI**						
	MTL														
1	Rules	49.4	17.4	25.7	31.8	11.2	16.5	-	33.1	22.5	26.8	15.2	10.3	12.3	-
2	LSTM^J	49.6	19.1	29.3	26.0	9.7	14	59.1	40.5	9.3	15.1	19.8	4.5	7.3	58.1
3	LSTM^P	50.0	24.0	32.4	25.4	12.2	16.5	61.1	37.3	16.0	22.4	16.3	7.0	9.8	48.6
4	H_1(LSTM^P)	45.8	32.4	**37.9**	25.9	18.3	21.5	61.1	33.6	32.3	**33.0**	16.4	15.8	**16.1**	48.6
5	H_2(LSTM^P)	53.3	26.5	35.4	32.6	16.2	**21.6**	60.6	41.2	22.8	29.3	22.6	12.5	**16.1**	47.1
6	BERT^J	72.1	7.8	14.1	45.7	5.0	9.0	61.9	68.7	3.1	5.9	53.6	2.4	4.6	60.3
7	BERT^P	33.3	32.0	32.5	19.0	18.2	18.5	56.9	37.1	23.4	28.7	15.0	9.4	11.6	43.8
8	H_1(BERT^P)	33.8	39.5	**36.4**	20.1	23.5	21.7	56.9	32.4	36.5	**34.3**	15.1	17.0	16.0	43.8
9	H_2(BERT^P)	33.7	32.7	33.2	22.4	21.8	**22.1**	55.1	45.8	23.7	31.2	23.9	12.4	**16.3**	48.2
10	SBERT^J	79.0	15.6	26.1	52.7	10.4	17.4	68.8	64.5	9.0	15.8	40.1	5.6	9.8	66.5
11	SBERT^P	34.4	30.9	32.6	22.3	20.1	21.1	59.5	34.7	29.4	31.8	15.3	12.9	14.0	47.5
12	H_1(SBERT^P)	35.1	38.8	**36.8**	22.2	24.6	23.4	59.5	31.3	41.6	**35.7**	14.8	19.6	16.9	47.5
13	H_2(SBERT^P)	39.7	31.6	35.1	27.0	21.5	**23.9**	59.2	36.0	27.5	31.2	19.7	15.0	**17.0**	45.4
	STL														
14	LSTM^J	57.4	11.6	19.3	20.4	4.2	6.9	-	59.5	8.6	14.9	15.9	2.3	4.0	-
15	LSTM^P	54.5	15.4	24.0	22.7	6.3	9.9	-	41.7	11.3	17.7	14.6	4.0	6.2	-
16	BERT^J	66.5	4.9	9.1	47.6	3.5	6.5	-	86.7	2.3	4.5	63.3	1.6	3.2	-
17	BERT^P	24.5	36.5	29.2	12.5	18.6	14.9	-	30.8	19.0	23.5	8.9	5.5	6.8	-
18	SBERT^J	67.9	16.7	26.6	38.2	9.1	14.6	-	61.7	5.9	10.8	36.0	3.4	6.2	-
19	SBERT^P	35.2	32.9	33.9	17.9	16.7	17.3	-	26.5	26.8	26.6	11.0	11.1	11.0	-

Table 3: Results of different MTL and STL resolvers in the end-to-end setting. Each result is the average of two runs. The highest recognition and resolution F-scores are bolded for each encoder in MTL.

Finally, for the hybrid approach, we employ our extension of the publicly-available implementation of Y&P's resolver and our re-implementation of Rösiger et al.'s resolver, as described above.

5.2 Results and Discussion

Baselines. Strictly speaking, there are no baselines, as no one has reported results on ISNotes and BASHI in the end-to-end setting. Nevertheless, we will use the three approaches described in Section 4, namely our end-to-end versions of the state-of-the-art approaches to bridging resolution in the gold mention setting, as our baselines.

Results of bridging recognition and resolution in the end-to-end setting for both ISNotes and BASHI are shown in Table 3. For bookkeeping purposes, we also report results on entity coreference resolution. Rules (row 1) corresponds to our duplication of Rösiger et al.'s rule-based approach (Section 4.2). LSTM^J (row 2) and LSTM^P (row 3) are the joint and pipeline versions of Y&P's MTL approach using LSTM as the encoder (Section 4.1). H_1(LSTM^P) (row 4) and H_2(LSTM^P) (row 5) are the original version and our proposed variant of K&N's hybrid approach, respectively.

Several points deserve mention. First, LSTM^J underperforms LSTM^P. This is somewhat unexpected since pipeline models are prone to error propagation and have been shown to underperform their joint counterparts in many NLP tasks. A closer examination of the output reveals the reason: since the mention extraction F-scores on both datasets are above 90%, error propagation is by no means serious. In contrast, LSTM^J has particularly poor anaphor recognition recall, which translates to poor resolution recall and F-score. Second, LSTM^P does not perform better than Rules. While LSTM^P achieves considerably higher recognition recall than Rules, it does not perform better than Rules w.r.t. resolution. For BASHI, LSTM^P underperforms Rules w.r.t. both recognition and resolution. H_1(LSTM^P) outperforms both Rules and LSTM^P. This is perhaps not surprising: this hybrid variant has achieved the best results on ISNotes and BASHI in the gold mention setting and represents the prior state of the art. Our results suggest that the success of this hybrid variant can be extended to the end-to-end setting. Like in the gold setting, recognition and resolution F-scores are both better than other baselines in the end-to-end setting. Finally, consider H_2(LSTM^P). While H_2(LSTM^P) achieves lower recognition F-scores than H_1(LSTM^P), the resolution F-scores achieved by the two models are comparable, with H_2(LSTM^P) having higher resolution precision and lower resolution recall than H_1(LSTM^P). This suggests that H_2 could be more valuable than H_1 for downstream applications, as these applications

	Model	Bridging						Coref.	Bridging						Coref.
		Recognition			Resolution			Res.	Recognition			Resolution			Res.
		P	R	F	P	R	F	CoNLL	P	R	F	P	R	F	CoNLL
		ISNotes							**BASHI**						
	MTL														
1	Rules	52.7	19.2	28.1	34.0	12.4	18.1	-	35.8	23.6	28.5	17.8	11.7	14.1	-
2	LSTM	53.0	25.6	34.5	24.2	11.7	17.2	64.0	38.2	16.5	23.0	17.3	7.5	10.4	57.4
3	H_1(LSTM)	48.1	34.6	**40.3**	27.2	19.6	22.8	64.0	34.7	33.7	**34.2**	18.1	17.5	**17.8**	57.4
4	H_2(LSTM)	56.5	28.8	38.1	34.0	17.3	**22.9**	63.8	45.4	22.8	30.4	26.6	13.4	**17.8**	57.4
5	BERT	34.9	33.2	33.9	20.3	19.3	19.7	60.2	37.6	23.4	28.8	15.2	9.4	11.6	52.1
6	H_1(BERT)	35.6	42.2	**38.6**	21.1	25.0	22.8	60.2	33.6	37.8	**35.6**	16.4	18.5	17.4	52.1
7	H_2(BERT)	36.3	35.7	36.0	23.7	23.3	**23.5**	58.3	46.3	24.8	32.3	25.8	13.8	**18.0**	50.7
8	SBERT	37.1	33.1	35.0	24.5	21.9	23.1	62.9	35.0	29.7	32.1	16.1	13.7	14.8	54.9
9	H_1(SBERT)	37.6	42.4	**39.8**	23.6	26.6	25.0	59.5	32.2	43.0	**36.8**	16.3	21.7	18.6	54.9
10	H_2(SBERT)	43.8	34.6	38.6	30.4	24.1	**26.8**	62.6	37.6	28.8	32.6	21.6	16.6	**18.7**	55.1
	STL														
11	LSTM	57.4	16.4	25.4	25.1	11.0	15.2	-	42.8	11.5	18.1	15.5	4.2	6.5	-
12	BERT	26.0	38.2	30.8	13.1	19.2	15.5	-	28.9	20.5	24.0	9.1	6.5	7.6	-
13	SBERT	37.7	34.4	35.9	19.9	18.1	18.9	-	27.8	26.8	27.3	12.2	11.7	12.0	-

Table 4: Results of different MTL and STL resolvers in the gold mention setting. Each result is the average of two runs. The highest recognition and resolution F-scores are bolded for each encoder in MTL.

typically cannot benefit from bridging information if many links are erroneous.[5]

LSTM vs. BERT/SpanBERT. Results for BERT and SpanBERT (SBERT) are shown in rows 6–9 and rows 10–13 respectively. Comparing the BERT results with the corresponding LSTM results, we see that the two achieve comparable F-scores w.r.t. resolution except for three cases (BERT^P outperforms LSTM^P on both datasets, and H_2(BERT^P) outperforms H_2(LSTM^P) on ISNotes). In terms of recognition, the results are mixed: on BASHI the BERT models outperform their LSTM counterparts, while the reverse is true on ISNotes. Next, consider the SBERT results. Each SBERT model considerably outperforms the corresponding BERT and LSTM models. Generally, the higher resolution F-scores achieved by SBERT can be attributed to its higher recall on BASHI and its higher precision on ISNotes. These results show the usefulness of SBERT for bridging resolution.

MTL vs. STL. Y&P show that multi-task learning for entity coreference and bridging outperforms single-task learning (i.e., learning bridging resolution without coreference). The question is: would MTL still outperform STL in the end-to-end setting? To answer this question, we obtain STL results by removing the coreference prediction layer in Y&P's model and retraining it. Results of this experiment using different encoders in the Y&P model are shown in rows 11–13. As can be seen, regardless of which encoder is used, the resolution F-scores achieved by STL are lower than those achieved by MTL for both datasets.

End-to-end vs. gold settings. Will the trends we have observed so far generalize to the gold mention setting? To answer this question, we repeat the experiments in Table 3 on gold mentions. There is a caveat involved in evaluating on gold mentions, however. In ISNotes and BASHI, some bridging anaphors have clausal antecedents that correspond to *events*. While clausal antecedents are annotated, they are *not* annotated as gold mentions, and previous studies differ in terms of how they should be handled. Specifically, some previous work (e.g., Hou et al. (2014), Hou et al. (2018)) chose not to include these clausal antecedents in the list of candidate antecedents and others (e.g., Rösiger et al. (2018), Yu and Poesio (2020)) did. Obviously, the setting in which gold clausal antecedents are not included in training/evaluation is harsher because it implies that anaphors with clausal antecedents will always be resolved incorrectly. We believe that including gold clausal antecedents during evaluation does not represent a realistic setting, and will therefore report results using the "harsh" setting when evaluating on gold mentions.

Results of the gold mention setting are shown in Table 4.[6] Recall that the distinction between joint

[5]We use the pipeline version rather than the joint version of Y&P's MTL model in the hybrid variants because of our desire to create stronger baselines.

[6]The baseline results in Table 4 are lower than those reported in the original papers because (1) we report results using the "harsh" setting; (2) Rösiger et al. (2018) and Kobayashi and Ng (2021) postprocess the system output with gold coreference information, and (3) Yu and Poesio (2020) and Kobayashi and Ng (2021) use additional labeled data for model training.

Perturbation Type	Example
Seen adj/adv	strategically → slightly
Nonexistent adj/adv	skeptical → lacitpeks
Seen verbs	start → reply
Nonexistent verbs	possess → ssessop
Seen nouns	honesty → wall
Nonexistent nouns	example → elpmaxe
Seen words	particularly → firmly
Nonexistent words	begun → nugeb

Table 5: Perturbation examples.

	Pertutbation Type	ISNotes		BASHI	
		Rec.	Res.	Rec.	Res.
1	No perturbation	**35.1**	**23.9**	**31.2**	**17.0**
2	Seen adj/adv	33.0	22.5	29.0	12.7
3	Nonexistent adj/adv	34.9	23.4	28.8	12.9
4	Seen verbs	35.0	23.2	28.9	13.3
5	Nonexistent verbs	35.0	23.5	30.4	13.4
6	Seen nouns	32.9	22.1	30.1	12.5
7	Nonexistent nouns	31.8	22.2	29.6	12.7
8	Seen words	33.2	20.6	30.4	12.2
9	Nonexistent words	32.3	21.3	25.4	12.7

Table 6: Perturbation results of the best model.

and pipeline approaches is no longer applicable in the gold mention setting. As can be seen, the conclusions we drew based on the end-to-end results are also applicable to the gold mention results.

One of the questions we aim to answer is: how much worse would the end-to-end results be compared to the corresponding gold mention results? We see that the end-to-end LSTM-based (LSTM^P, H_1(LSTM^P), H_2(LSTM^P)) and BERT-based (BERT^P, H_1(BERT^P), H_2(BERT^P)) resolvers underperform their counterparts in the gold setting by up to 2.8% F-score in recognition and up to 1.7% F-score in resolution. This performance gap widens with SBERT (SBERT^P, H_1(SBERT^P), H_1(SBERT^P)), having a difference of up to 3.5% F-score in recognition and up to 2.9% F-score in resolution. Overall, while the gold results are better than the corresponding end-to-end results, the difference between them is less than 1.8% for LSTM and BERT and less than 3.0% for SBERT. These results are encouraging considering that the end-to-end evaluation setting is very challenging.

H_1 vs. H_2. While the performance difference between H_1 and H_2 tends to be small w.r.t. resolution in the end-to-end setting, there are cases in the gold mention setting in which this difference in resolution F-score is comparatively larger. Specifically, when BERT is used, H_2 outperforms H_1 by 0.6–0.7% points in F-score on the two corpora, and when SBERT is used, H_2 outperforms H_1 by 1.8% points in F-score on ISNotes.

5.3 Sensitivity to Perturbed Inputs

Next, we conduct experiments that involve perturbing the input. For each experiment, we replace a certain type of words with other words in all training documents, retrain our best-performing model, H_2(SBERT^P), on these perturbed training documents, and evaluate it on the (unperturbed) test set. The goal is to gain insights into the behavior of the best model by assessing how sensitive its performance is when training inputs are perturbed. If performance drops a lot when a certain type of word is replaced, then it means that that type of words is important in the learning process. Note that we consider only mention-external perturbations, meaning that we only replace words that are not part of a bridging anaphor or its antecedent(s).

Specifically, we replace words from the following categories: adjectives and adverbs only, verbs only, nouns only, and all categories combined. For each category, we consider two replacement methods. One is to replace each word with another word of the same POS tag that is taken from the training documents but which has never appeared within a mention in the training set (**Seen**). This replacement is deterministic: all occurrences of a given word will be replaced with the same word. The other method involves replacing each word with a nonexistent word (**Nonexistent**), which we create by reversing the order of the characters of the word to be replaced. This latter method tests the impact of nonexistent words has on the model.

Results of these experiments are reported in Table 6 in terms of recognition and resolution F-scores. To facilitate comparison, we show in row 1 the results of the resolver when the input is not perturbed. Several points deserve mention. First, all results obtained via perturbations are lower than the "No perturbation" results in row 1. This implies that each kind of perturbation we considered affects the model learning process and negatively impacts bridging recognition and resolution performances. Second, Seen words appear to confuse the model more than Nonexistent words. This is perhaps not surprising: in the Nonexistent setting the model will not be confused by those Seen replacements that could cause a sentence to become unsensible. Finally, we see from rows 4 and 5 that verbs have the least impact on resolution F-scores, suggesting that adj/adv and nouns play more important roles than verbs in learning span-based models for

bridging resolution.

5.4 Analysis of Results

Error analysis of the best end-to-end model. To gain additional insights into our best end-to-end model (H_2(SBERT^P)), we conduct an error analysis of this resolver. First, the system is still struggling to detect the majority of the bridging anaphors and find their antecedents, having recall scores of 31.6% and 27.5% for bridging anaphora recognition on ISNotes and BASHI, respectively. Only a very small portion of the recall errors are from mention prediction errors: 3% and 1.3% of the gold bridging anaphors are misclassified as non-mentions in ISNotes and BASHI, respectively. The system makes more recall errors at predicting definite bridging anaphors (i.e., NPs modified by the definite article "the") than other bridging anaphors. For instance, on ISNotes, the recall scores of identifying definite bridging anaphors and other bridging anaphors are 20% and 25%, respectively.

Next we analyze the precision errors on ISNotes because BASHI does not annotate mentions and their information status. We find that mention prediction errors (i.e., predicted bridging anaphors are not mentions) account for 8.7% of the precision errors for bridging anaphora recognition. In addition, 16.7% of the wrongly predicted bridging pairs contain correct bridging anaphors but wrong antecedents. The majority of the precision errors can be attributed to the fact that the system predicts *new* and *old* mentions as bridging anaphors, which account for 31% and 21% of the precision errors, respectively. This is in line with the previous studies on bridging recognition that suggest that systems often fail to distinguish bridging anaphors from generic *new* mentions with simple syntactic structures (Hou et al., 2018; Hou, 2021).

Comparison of different encoders and embeddings. We analyze the results from three end-to-end systems: H_2(LSTM^P), H_2(BERT^P), and H_2(SBERT^P). which correspond to rows 5, 9, and 13 in Table 3, respectively. As noted before, the LSTM encoder with BERT embeddings (i.e., H_2(LSTM^P)) is more conservative in link prediction, having higher precision but lower recall than the other two systems. In fact, on ISNotes, H_2(LSTM^P) only predicts half of the number of bridging pairs predicted by the other two systems. Interestingly, although both H_2(BERT^P) and H_2(SBERT^P) achieve higher recall scores on full bridging resolution compared to H_2(LSTM^P), they both make a relatively large portion of precision errors that involve linking a *mediated/syntactic* mention m to a previous mention that is often related to the premodification of m, such as {*Britain's voters – Britains's*} or {*Some Mobil executives – Mobil Corp.*}. On the contrary, this kind of error is rare in H_2(LSTM^P).

Finally, we analyze the recall scores based on the determiners of bridging anaphors. We divide bridging anaphors into three categories: (1) *the NPs* correspond to bridging anaphors that are modified by the definite article "the"; (2) *other determiner NPs* contain bridging anaphors that are modified by the indefinite articles "a/an" as well as other determiners (e.g., demonstratives or possessives); and (3) *bare NPs* are bridging anaphors that are not modified by any determiners, such as "subsidies" and "overseas operations". The majority of the correctly predicted bridging links from the above three models are *bare NPs*. H_2(BERT^P) is better at predicting bridging anaphors for all three categories compared to H_2(LSTM^P). The performance of H_2(SBERT^P) on *other determiner NPs* and *bare NPs* is on par with that of H_2(BERT^P), but the former achieves higher recall at identifying definite bridging anaphors than the latter (i.e., 20% vs. 14% on ISNotes).

6 Conclusion

We conducted a pioneering study on end-to-end neural bridging resolution in which we adapted three state-of-the-art bridging resolvers that were originally developed to operate on gold mentions, namely Rösiger et al.'s (2018) resolver, Yu and Poesio's (2020) resolver, and Kobayashi and Ng's (2021) resolver, to the end-to-end setting. To strengthen the resolvers, we replaced the LSTM encoders they use with BERT- and SpanBERT-based encoders. In an evaluation on ISNotes and BASHI, end-to-end bridging resolvers lagged behind their gold-mention counterparts by only 2-3% absolute F-score. These results suggested that time is ripe for researchers to focus on evaluating bridging resolvers in the end-to-end setting. In addition, our work suggested that H_2(SBERT^P), the hybrid score-based pipeline bridging resolver trained using SBERT (1) achieves better performance than other model variants; (2) is sensitive to all kinds of perturbations we considered; and (3) will likely be improved by improving mediated/syntactic errors.

Acknowledgments

We thank the three anonymous reviewers for their insightful comments on an earlier draft of the paper. This work was supported in part by NSF Grants IIS-1528037 and CCF-1848608. Any opinions, findings, conclusions or recommendations expressed in this paper are those of the authors and do not necessarily reflect the views or official policies, either expressed or implied, of the NSF.

References

Amit Bagga and Breck Baldwin. 1998. Algorithms for scoring coreference chains. In *Proceedings of the Linguistic Coreference Workshop at the First International Conference on Language Resources and Evaluation (LREC'98)*, pages 563–566, Granada, Spain. European Language Resources Association (ELRA).

Herbert H. Clark. 1975. Bridging. In *Proceedings of the 1975 Workshop on Theoretical Issues in Natural Language Processing*, TINLAP '75, page 169–174, USA. Association for Computational Linguistics.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, Minneapolis, Minnesota. Association for Computational Linguistics.

Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long short-term memory. *Neural Computation*, 9(8):1735–1780.

Matthew Honnibal and Ines Montani. 2017. spaCy 2: Natural language understanding with Bloom embeddings, convolutional neural networks and incremental parsing. To appear.

Yufang Hou. 2020a. Bridging anaphora resolution as question answering. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 1428–1438, Online. Association for Computational Linguistics.

Yufang Hou. 2020b. Fine-grained information status classification using discourse context-aware BERT. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 6101–6112, Barcelona, Spain (Online). International Committee on Computational Linguistics.

Yufang Hou. 2021. End-to-end neural information status classification. In *Findings of the Association for Computational Linguistics: EMNLP 2021*, pages 1377–1388, Punta Cana, Dominican Republic. Association for Computational Linguistics.

Yufang Hou, Katja Markert, and Michael Strube. 2013. Cascading collective classification for bridging anaphora recognition using a rich linguistic feature set. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*, pages 814–820, Seattle, Washington, USA. Association for Computational Linguistics.

Yufang Hou, Katja Markert, and Michael Strube. 2014. A rule-based system for unrestricted bridging resolution: Recognizing bridging anaphora and finding links to antecedents. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 2082–2093, Doha, Qatar. Association for Computational Linguistics.

Yufang Hou, Katja Markert, and Michael Strube. 2018. Unrestricted bridging resolution. *Computational Linguistics*, 44(2):237–284.

Mandar Joshi, Danqi Chen, Yinhan Liu, Daniel S. Weld, Luke Zettlemoyer, and Omer Levy. 2020. SpanBERT: Improving pre-training by representing and predicting spans. *Transactions of the Association for Computational Linguistics*, 8:64–77.

Mandar Joshi, Omer Levy, Luke Zettlemoyer, and Daniel Weld. 2019. BERT for coreference resolution: Baselines and analysis. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 5803–5808, Hong Kong, China. Association for Computational Linguistics.

Sopan Khosla, Juntao Yu, Ramesh Manuvinakurike, Vincent Ng, Massimo Poesio, Michael Strube, and Carolyn Rosé. 2021. The CODI-CRAC 2021 shared task on anaphora, bridging, and discourse deixis in dialogue. In *Proceedings of the CODI-CRAC 2021 Shared Task on Anaphora, Bridging, and Discourse Deixis in Dialogue*, pages 1–15, Punta Cana, Dominican Republic. Association for Computational Linguistics.

Hongjin Kim, Damrin Kim, and Harksoo Kim. 2021. The pipeline model for resolution of anaphoric reference and resolution of entity reference. In *Proceedings of the CODI-CRAC 2021 Shared Task on Anaphora, Bridging, and Discourse Deixis in Dialogue*, pages 43–47, Punta Cana, Dominican Republic. Association for Computational Linguistics.

Hideo Kobayashi, Yufang Hou, and Vincent Ng. 2022. Constrained multi-task learning for bridging resolution. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 759–770, Dublin, Ireland. Association for Computational Linguistics.

Hideo Kobayashi, Shengjie Li, and Vincent Ng. 2021. Neural anaphora resolution in dialogue. In *Proceedings of the CODI-CRAC 2021 Shared Task on Anaphora, Bridging, and Discourse Deixis in Dialogue*, pages 16–31, Punta Cana, Dominican Republic. Association for Computational Linguistics.

Hideo Kobayashi and Vincent Ng. 2020. Bridging resolution: A survey of the state of the art. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 3708–3721, Barcelona, Spain (Online). International Committee on Computational Linguistics.

Hideo Kobayashi and Vincent Ng. 2021. Bridging resolution: Making sense of the state of the art. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 1652–1659, Online. Association for Computational Linguistics.

Shengjie Li, Hideo Kobayashi, and Vincent Ng. 2022. Neural anaphora resolution in dialogue revisited. In *Proceedings of the CODI-CRAC 2022 Shared Task on Anaphora, Bridging, and Discourse Deixis in Dialogue*, Gyeongju, Republic of Korea. Association for Computational Linguistics.

Xiaoqiang Luo. 2005. On coreference resolution performance metrics. In *Proceedings of Human Language Technology Conference and Conference on Empirical Methods in Natural Language Processing*, pages 25–32, Vancouver, British Columbia, Canada. Association for Computational Linguistics.

Christopher Manning, Mihai Surdeanu, John Bauer, Jenny Finkel, Steven Bethard, and David McClosky. 2014. The Stanford CoreNLP natural language processing toolkit. In *Proceedings of 52nd Annual Meeting of the Association for Computational Linguistics: System Demonstrations*, pages 55–60, Baltimore, Maryland. Association for Computational Linguistics.

Katja Markert, Yufang Hou, and Michael Strube. 2012. Collective classification for fine-grained information status. In *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 795–804, Jeju Island, Korea. Association for Computational Linguistics.

Shachar Mirkin, Ido Dagan, and Sebastian Padó. 2010. Assessing the role of discourse references in entailment inference. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics*, pages 1209–1219, Uppsala, Sweden. Association for Computational Linguistics.

Malvina Nissim, Shipra Dingare, Jean Carletta, and Mark Steedman. 2004. An annotation scheme for information status in dialogue. In *Proceedings of the Fourth International Conference on Language Resources and Evaluation (LREC'04)*, Lisbon, Portugal. European Language Resources Association (ELRA).

Onkar Pandit, Pascal Denis, and Liva Ralaivola. 2020. Integrating knowledge graph embeddings to improve mention representation for bridging anaphora resolution. In *Proceedings of the Third Workshop on Computational Models of Reference, Anaphora and Coreference*, pages 55–67, Barcelona, Spain (online). Association for Computational Linguistics.

Massimo Poesio, Rahul Mehta, Axel Maroudas, and Janet Hitzeman. 2004. Learning to resolve bridging references. In *Proceedings of the 42nd Annual Meeting of the Association for Computational Linguistics (ACL-04)*, pages 143–150, Barcelona, Spain.

Sameer Pradhan, Xiaoqiang Luo, Marta Recasens, Eduard Hovy, Vincent Ng, and Michael Strube. 2014. Scoring coreference partitions of predicted mentions: A reference implementation. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 30–35, Baltimore, Maryland. Association for Computational Linguistics.

Ellen F. Prince. 1981. Toward a taxonomy of given-new information. In P. Cole, editor, *Syntax and semantics: Vol. 14. Radical Pragmatics*, pages 223–255. Academic Press, New York.

Altaf Rahman and Vincent Ng. 2011. Learning the information status of noun phrases in spoken dialogues. In *Proceedings of the 2011 Conference on Empirical Methods in Natural Language Processing*, pages 1069–1080, Edinburgh, Scotland, UK. Association for Computational Linguistics.

Altaf Rahman and Vincent Ng. 2012. Learning the fine-grained information status of discourse entities. In *Proceedings of the 13th Conference of the European Chapter of the Association for Computational Linguistics*, pages 798–807, Avignon, France. Association for Computational Linguistics.

Ina Rösiger. 2018. BASHI: A corpus of Wall Street Journal articles annotated with bridging links. In *Proceedings of the Eleventh International Conference on Language Resources and Evaluation (LREC 2018)*, Miyazaki, Japan. European Language Resources Association (ELRA).

Ina Rösiger, Arndt Riester, and Jonas Kuhn. 2018. Bridging resolution: Task definition, corpus resources and rule-based experiments. In *Proceedings of the 27th International Conference on Computational Linguistics*, pages 3516–3528, Santa Fe, New Mexico, USA. Association for Computational Linguistics.

Bo-Hsiang Tseng, Shruti Bhargava, Jiarui Lu, Joel Ruben Antony Moniz, Dhivya Piraviperumal, Lin Li, and Hong Yu. 2021. CREAD: Combined resolution of ellipses and anaphora in dialogues. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 3390–3406, Online. Association for Computational Linguistics.

Olga Uryupina, Ron Artstein, Antonella Bristot, Federica Cavicchio, Francesca Delogu, Kepa Joseba Rodriguez, and Massimo Poesio. 2020. Annotating a broad range of anaphoric phenomena, in a variety of genres: the ARRAU corpus. *Natural Language Engineering*, 26(1):95–128.

Marc Vilain, John Burger, John Aberdeen, Dennis Connolly, and Lynette Hirschman. 1995. A model-theoretic coreference scoring scheme. In *Sixth Message Understanding Conference (MUC-6): Proceedings of a Conference Held in Columbia, Maryland, November 6-8, 1995*.

Juntao Yu, Bernd Bohnet, and Massimo Poesio. 2020. Neural mention detection. In *Proceedings of the 12th Language Resources and Evaluation Conference*, pages 1–10, Marseille, France. European Language Resources Association.

Juntao Yu, Sopan Khosla, Ramesh Manuvinakurike, Lori Levin, Vincent Ng, Massimo Poesio, Michael Strube, and Carolyn Rosé. 2022. The CODI-CRAC 2022 shared task on anaphora, bridging, and discourse deixis in dialogue. In *Proceedings of the CODI-CRAC 2022 Shared Task on Anaphora, Bridging, and Discourse Deixis in Dialogue*, Gyeongju, Republic of Korea. Association for Computational Linguistics.

Juntao Yu and Massimo Poesio. 2020. Multitask learning-based neural bridging reference resolution. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 3534–3546, Barcelona, Spain (Online). International Committee on Computational Linguistics.

A Rules

Rösiger et al. (2018) designed rules to resolve the bridging anaphors in ISNotes and BASHI. Table 7 shows these rules. For each rule, we describe the anaphor and antecedent conditions as well as its motivation and its resolution precision. The precision scores are calculated using our re-implementation or Rösiger et al.'s resolver. Note that the first nine rules are used for resolution in both ISNotes and BASHI while the last rule (comparative anaphora) is specifically designed for BASHI.

Rule	Description (anaphor)	Description (antecedent)	Motivation	Res. Precision (%)
Set: Percentage	Percentage NPs in subject position	Closest NP modifying another percentage NP via the preposition "of" (e.g. 22% of the firms)	Percentage expressions can indicate set bridging	I: 100.0 B: 0.0
Building part	Common NPs whose head is a building part without nominal pre-modifications	NP with the strongest semantic connectivity to anaphor	A typical case (building part) of meronym bridging	I: 62.5 B: 0.0
Set: Number or indefinite pronoun	Number expressions (e.g. two dogs) or indefinite pronouns (e.g. some ...)	Closest plural NP in subject position. If not found, closest plural NP in object position	Numbers or indefinite pronouns can indicate set bridging	I: 80.0 B: 33.3
Argument-taking NPs 1	NPs with high argument ratio and without nominal/adjective pre-modifications or indefinite determiners	1. take all nominal modifiers of NPs whose head is same as anaphor's head. 2. closest NP that is a realization of these modification	Different instances of the same noun predicate likely maintain the same argument fillers indicated by nominal modifiers (extended claim from Laparra'13)	I: 40.0 B: 18.5
Relative person	Non-generic NPs whose head is a relative without no nominal/adjective pre-modifications	Closest non-relative person NP	Handles relative nouns, which tend to be bridging	I: 50.0 B: 42.9
GPE job title	Job titles with country pre-modifications (e.g., Italian mayor)	Most salient GPE (e.g., Italy)	Some job title NPs implicitly refer to the globally salient GPE	I: 45.0 B: 14.3
Professional role	Professional role NPs (e.g. professor)	Most salient organization name	A more general rule than "Relative person" and "GPE job title"	I: 62.0 B: 21.2
Argument-taking NPs 2	NPs in subject position with high argument ratio and without nominal/adjective pre-modifications	NP with the strongest semantic connectivity to the anaphor	An NP in subject position that is likely to take arguments tends to be bridging anaphor	I: 28.1 B: 0.0
Meronym relation	Unmodified definite NPs	NP classified as meronym with anaphor according to a relation classifier trained using WordNet	Handles meronym bridging	I: 11.8 B: 11.3
Comparative anaphora	NPs with comparative markers	Closest NP with same head and semantic category	Comparative anaphors are typically indicated by certain markers	B: 45.5

Table 7: Rules used by Rösiger et al. (2018) for resolving bridging anaphors in ISNotes and BASHI. 'I' and 'B' refer to ISNotes and BASHI, respectively.

Investigating the Performance of Transformer-Based NLI Models on Presuppositional Inferences

Jad Kabbara and **Jackie Chi Kit Cheung**[†]
School of Computer Science, McGill University, Montreal, QC, Canada
Montreal Institute for Learning Algorithms (Mila), Montreal, QC, Canada
{jad, jcheung}@cs.mcgill.ca
[†] Canada CIFAR AI Chair

Abstract

Presuppositions are assumptions that are taken for granted by an utterance, and identifying them is key to a pragmatic interpretation of language. In this paper, we investigate the capabilities of transformer models to perform NLI on cases involving presupposition. First, we present simple heuristics to create alternative "contrastive" test cases based on the ImpPres dataset and investigate the model performance on those test cases. Second, to better understand how the model is making its predictions, we analyze samples from sub-datasets of ImpPres and examine model performance on them. Overall, our findings suggest that NLI-trained transformer models seem to be exploiting specific structural and lexical cues as opposed to performing some kind of pragmatic reasoning.

1 Introduction

Natural language inference (NLI) is the task of predicting whether a sentence entails, contradicts or is neutral with respect to another sentence. While NLI is presented as a task with the goal of pushing the frontier of language understanding, one question that remains elusive is the extent to which learning models trained on NLI datasets acquire competence in pragmatic reasoning. For example, "Rose's drawing stunned the audience" presupposes that "Rose has a drawing." In terms of the NLI task, the relationship between these two statements is one of entailment. However, "Rose stunned the audience" does not trigger the same presupposition. If an NLI system relies on an incorrect heuristic to predict entailment, such as the fact that both statements have "Rose" in them, the system would fail on the modified version.

While there exists a plethora of NLI datasets (e.g. SNLI (Bowman et al., 2015), MNLI (Williams et al., 2018), HANS (McCoy et al., 2019)), issues related to pragmatics remain under-explored. One exception is the ImpPres dataset (Jeretic et al., 2020), which contains English NLI instances involving two pragmatic phenomena: implicature and presupposition.

In linguistics and philosophy of language, presuppositions are assumptions and beliefs that are shared and taken for granted by discourse participants without an explicit mention in the discourse context. Presuppositions are prevalent in language and understanding them facilitates smooth communication and is crucial for a proper understanding of the meaning being conveyed in a given context.

In this work, we investigate how well transformer models perform on cases from ImpPres involving presupposition. We ask the following question: In cases where these models seem to be doing well in terms of (NLI) accuracy performance, are they exploiting any patterns or correlations between certain words and decision labels which happen to lead to a good performance on ImpPres, but which might not generalize beyond it?

Our contribution is two-fold: First, we present simple heuristics to create alternative "contrastive" test cases based on the Presupposition sub-datasets in ImpPres and investigate the model performance on those test cases. Second, from those datasets, we draw a random set of testing samples and examine the performance of the models on these sets. In each of those cases, we identify specific patterns or cues that seem to be influencing the accuracy performance (in either the correct or incorrect predictions). Our findings suggest that transformer-based NLI models seem to be exploiting surface-level lexical and structural cues as opposed to performing some kind of pragmatic reasoning and that these cues are highly task-specific and do not generalize to adversarially perturbed versions of the input.

2 Related Work

Natural language inference (NLI) is the task of predicting whether one passage entails another. In

Proceedings of the 29th International Conference on Computational Linguistics, pages 779–785
October 12–17, 2022.

Type	Premise	Hypothesis	Label
All N	All six roses that bloomed died.	Exactly six roses bloomed.	E
Both	Both flowers that bloomed died.	Exactly two flowers bloomed.	E
Change of State	Rene might have hidden.	Rene hid.	N
Cleft Existence	It might be Becky who researched Jesus.	Someone researched Jesus.	E
Cleft Uniqueness	It is Joel who helps Diana.	Exactly one person helps Victoria.	N
Only	Susan only writes.	Susan doesn't write.	C
Possessed Definites	Rose's bird did alarm Peter.	Rose has a bird.	E
Question	Did Bill wonder when Omar hunted?	Omar didn't hunt.	C

Table 1: Examples showing the different presupposition types in the ImpPres dataset.

our work, we adopt the modern formulation[1] cast as a three-way classification task where two statements, a premise and a hypothesis, are bound by a relationship of entailment, contradiction or neither (the "neutral" case). Many datasets have been introduced for the task including SNLI (Bowman et al., 2015), MNLI (Williams et al., 2018), MPE (Lai et al., 2017), XNLI (Conneau et al., 2018), etc.

The ImpPres dataset (Jeretic et al., 2020) has been proposed to focus specifically on presupposition and implicature. ImpPres consists of 25.5k sentence pairs further divided into several sub-datasets, each focusing on a specific type of implicature or presupposition. In our work, we focus on the presupposition part of ImpPres. Table 1 presents one example from each sub-dataset.

Other work studied the same general issue of checking whether systems systematically changed their labels on perturbed samples in an expected way (Ribeiro et al., 2020; Emami et al., 2019; Sinha et al., 2021; Gardner et al., 2020; Niven and Kao, 2019). The HANS dataset (McCoy et al., 2019) in particular targets issues where NLI models fail because they rely on superficial syntactic heuristics. Unlike our work, the NLI instances they examine—and NLI in general—often require additional world knowledge, or linguistic knowledge that is highly lexicalized (e.g., a list of factive verbs). By contrast, the class of presuppositions that we examine is tied to specific linguistic constructions that systematically trigger them, such as definite articles and cleft constructions, which could potentially be easier. We nevertheless show that the models we tested fail to generalize systematically on presuppositional inferences, complementing previous results.

Also relevant to this work is the CommitmentBank dataset (De Marneffe et al., 2019) which focuses on issues involving presupposition projection in various environments. This work highlights that projection is hardly a binary affair, and that judgments about entailment (roughly corresponding to speakers' commitment in the language of CommitmentBank) are graded.

3 Methodology

Our study consists of fine-tuning a pre-trained transformer model (either BERT (Devlin et al., 2019) or RoBERTa (Liu et al., 2019)), then evaluating its performance on the various presupposition sub-datasets of ImpPres in addition to perturbed versions of those sub-datasets introduced in this work. We fine-tune on MNLI, following Jeretic et al. (2020).

We conduct two studies in order to investigate the performance of these models and better understand their capabilities in making pragmatic inferences. In the first study, we present simple heuristics to create "contrastive" test cases. We then evaluate the finetuned model on those test cases quantitatively. In the second study, we draw from each sub-dataset of ImpPres a random set of 100 samples. We examine these sets for structural and lexical cues that might be influencing the performance of BERT.

3.1 Training details

Similar to previous work using BERT-based models for NLI, we concatenate the premise and hypothesis separated by the [SEP] token, with the special [CLS] token preceding them. For the BERT model, we use HuggingFace's bert-large-uncased implementation (Wolf et al., 2019) of a 24-layer, 1024-hidden, 16-heads, 336M-parameter version of the model that was trained on lower-cased English text. Similarly, for RoBERTa, we use the roberta-large implementation. All models are implemented in PyTorch (Paszke et al., 2019) and

[1]For earlier (two-way classification) formulations, the reader is referred to (Dagan et al., 2005) and (Manning, 2006)

trained to minimize the standard cross-entropy cost with Adam (Kingma and Ba, 2015) as the optimizer with all default parameters except for the learning rate. All model hyperparameters are kept as default except for the following: We follow the recommended ranges for fine-tuning hyperparameters in the BERT paper (Devlin et al., 2019) and find that the optimal performance on the dev set is reached for a batch size of 16, learning rate of 2e-5 and weight decay of 0.01 for BERT, and learning rate of 1e-5 and weight decay of 0.001 for BERT. The BERT model achieves a dev set accuracy of 85.14% (comparable to that reported in (Jeretic et al., 2020) and (Devlin et al., 2019)).

4 Study 1: Contrastive Probing

In this study, we present simple heuristics to create "contrastive" test cases for the various ImpPres sub-datasets and investigate model performance on those test cases. We divide our probes into two types: those involving contrastive sentences which are grammatical English sentences, and those with corrupt input which are ungrammatical.

4.1 Probing with Valid English Sentences

Removing Possessives (POSS): In the first experiment, we focus on the "Possessed Definites Existence" subset of ImpPres. Possessive noun phrases trigger a presupposition about the existence of the possessed noun, and its possession by the possessor. For example, "Alice's painting is amazing" (premise) presupposes "Alice has a painting" (hypothesis), which corresponds to a gold label of entailment. We create a contrastive example by removing the possessed noun (or noun phrase) in the premise: "Alice is amazing", while keeping the hypothesis the same, changing the correct label to neutral. Applying this rule to all samples in this sub-dataset, all labels become neutral. We sampled and manually checked 100 cases and found that this heuristic is correct for all 100 sampled cases.

Replacing Names (NAMES): We process the premises using spaCy's Named Entity Recognizer (Honnibal et al., 2020), identify all instances of names of people and replace each by a (different) randomly selected name from the 200 most common names in USA.[2] For samples where a change is made, as the premise and hypothesis are now about different people, the label becomes neutral.

[2]www.ssa.gov/oact/babynames/decades/century.html

Label	Original Accuracy	Contrastive Accuracy
Entailment	99.2 %	-
Contradiction	99.0 %	-
Neutral	31.5 %	62.4 %
Label	**Original Gold Labels**	**Contrastive Gold Labels**
Entailment	26.3 %	0 %
Contradiction	31.6 %	0 %
Neutral	42.1 %	100%
Label	**Original Predictions**	**Contrastive Predictions**
Entailment	37.2 %	0.4 %
Contradiction	49.5 %	37.2 %
Neutral	13.3 %	62.4 %

Table 2: Per-class accuracy, distribution of golden labels (in %), distribution of predictions in the original and contrastive dataset (in %) for the BERT model for the POSS experiment (which focuses only on the "Possessed Definites Existence" sub-dataset.

4.2 Contrastive Probing with Corrupt Input

We are interested in probing the model with test cases that are corrupt in a way that the model is expected to perform worse. For example, if the model is provided with less context in the input or if we were to randomize the word order, the model is expected to perform worse.

Providing Less Context in the Premise (1ST-HALF): In this experiment, instead of a full premise, we present the first half of the sentence. The hypothesis and labels are kept unchanged.

Randomizing Word Order (RNDMZ): Here, we shuffle randomly the order of tokens in the premise and hypothesis and keep the labels unchanged.

4.3 Results and Discussion

For the POSS experiment, the accuracy drops from 70.6% in the original sub-dataset to 62.4% in the contrastive dataset for the BERT model and from 64.8% to 35.2% for the RoBERTa model. The sharp drops suggest the model failed to pick up on the deletion of the possessive which altered the pragmatic context. We present additional statistics in Table 2 for the BERT case. While the model had near-perfect accuracy for the entailment and contradiction labels, it performed at less than chance for the neutral case. While the labels are relatively spread across the three labels, the model's original predictions of "neutral" were less than a fourth of the expected number which explains the low accuracy per class (31.5%). From Table 3, while the original predictions of "contradiction" should have been switched to neutral, the majority of these predictions did not change. This shows that the model

		Original Predictions		
		Ent: 707	Cont: 941	Neu: 252
Changed to	Entailment	4	3	0
	Contradiction	148	529	30
	Neutral	555	409	222

Table 3: Break down of how BERT's predictions changed from the original to the contrastive dataset in the (POSS) experiment. As a reminder, here in the contrastive set, all samples become neutral.

failed to translate the change in context to a correct change in prediction.

Tables 4 and 5 present the results of the three experiments NAMES, 1ST-HALF and RNDMZ for the two models BERT and RoBERTa.

NAMES. The most striking result is that changing names led to a significant drop in accuracy with a sharp decrease seen in several datasets (e.g., going from 70.6% to 26.3% (BERT) and 64.8% to 22.6% (RoBERTa)). For RoBERTa, accuracy numbers decreased across all datasets. For BERT, the drop was seen for the subdatasets with the highest performance but, interestingly, in the datasets where the performance was low, the new performance was actually higher. This could be explained by the fact that in those cases, the model was performing very poorly to start with so for these cases the results might be not suggestive.

1ST-HALF. Here, for both BERT and RoBERTa, we notice that in most cases (13/18 cases), the contrastive accuracy was higher. This observation is counter-intuitive because one would expect that providing less context to the model will impact the performance negatively. Indeed, with the exception of the Possessed Definites Existence dataset where the presupposition is consistently in the first half of the premise, the remaining sub-datasets are quite diversified in where the presupposition appears in the premise (i.e., first half vs second half). Moreover, upon inspection of samples from various datasets, we see how cutting off the second half of the premise often removes key parts of the context, thus making it virtually impossible to establish the (original) presupposition based on that (cut) premise.

RNDMZ. In the last experiment, we notice a drop in the performance in several datasets for both models. However, the drop is not as large as one might expect given that the input is no longer grammatical or coherent.

	Original Accuracy	Contrastive Accuracy		
		NAMES	1ST-HALF	RNDMZ
Poss. Def. Existence	70.6	26.3	67.6	52.2
Question	66.4	28.6	50.2	52.3
Cleft Existence	63.0	15.8	65.8	53.7
Only	62.3	39.4	32.9	52.7
All n	43.5	44.7	46.7	40.7
Both	32.6	42.6	41.8	34.0
Change of state	30.4	36.8	37.3	27.8
Poss. Def. Uniqueness	23.3	46.3	27.0	36.4
Cleft Uniqueness	11.1	20.3	36.0	21.8

Table 4: Accuracy results for the BERT model.

	Original Accuracy	Contrastive Accuracy		
		NAMES	1ST-HALF	RNDMZ
Poss. Def. Existence	64.8	22.6	66.1	50.6
Cleft Existence	63.3	18.2	65.1	57.1
Question	61.8	26.0	55.2	50.8
Poss. Def. Uniqueness	56.9	13.0	60.7	41.1
Only	55.0	35.4	47.3	46.7
All n	50.8	35.0	56.7	38.7
Both	49.1	45.6	55.9	39.4
Change of state	36.1	35.5	42.3	37.7
Cleft Uniqueness	26.7	12.4	41.1	27.4

Table 5: Accuracy results for the RoBERTa model.

5 Study 2: Sample Error Analysis

We examine BERT's performance on 100 samples randomly drawn from each sub-dataset. We see patterns repeating across several datasets—all about surface cues which should not be directly responsible for presupposition. We group our insights into themes.

5.1 Exploiting Lexical Cues

One pattern that we noticed across several datasets is that certain tokens and negation heavily affect performance. In the "All n" sub-dataset, we notice that the token "exactly" appears in the hypothesis in 76% of samples (e.g., "All six roses that bloomed died." / "Exactly six roses bloomed"). For these samples, the accuracy was at 48.68%. For the remaining 24% (not having "exactly"), the accuracy was at 16.67%. A closer look at those 76% samples shows that samples that have negation in the premise had an accuracy of 72% versus 44.61% for those without negation. Interestingly, these results are closely replicated in another sub-dataset on the similar "both" presupposition effect. In this dataset, we find that there are 69% of samples having "exactly" in their premise. For these samples, the accuracy is at 42.02%. For the remaining 31% (i.e. not having "exactly" in the premise), the accuracy is at 19.35%. Similarly, in samples that have negation in their premise, the accuracy is 66.67% versus 33.33% for samples with no negation.

In the Cleft Existence dataset (e.g., "It is Keith who stunned Christina." / "Keith stunned Christina."), the model properly learns with 100% accuracy the association between "no one" and the "contradiction" label (e.g. "It is Helen that talked about Cheryl." / "No one talked about Cheryl."). However, for the samples not having "no one", the accuracy drops to 52.78%. This could suggest that the correct prediction of "contradiction" in these cases was likely based on a cue, the presence of "no one" as opposed to some pragmatic understanding of the presupposition at hand. Furthermore, while the model has learnt with perfect accuracy the connection between "no one" and a label of contradiction, it does less well with other samples where "someone" replaces the main subject in the premise (which is crucial for understanding the cleft presupposition). In those cases, the accuracy is at 68.75%, suggesting that the presence of "no one" had a stronger impact on the model's performance.

In principle, such patterns shouldn't occur with a model that can systemically reason about presuppositions; yet we found them in our analysis and they echo findings (Degen, 2015; Schuster et al., 2020) that negation and certain tokens can affect pragmatic inferences.

5.2 Exploiting Similarity Information

In the Change of State sub-dataset (e.g. "Rene might have hidden." / "Rene hid."), we notice cases where the same verb appears in both the premise and hypothesis or appears in a close morphological variation (e.g., hidden/hid, gotten/got). In 85% of those cases, the model predicted "entailment"—all incorrectly. This suggests that the model might be learning to associate "entailment" with some kind of similarity between the premise and hypothesis, which is incorrect in these cases. Previous work also noted how the similarity between two statements can affect inferences related to pragmatic phenomena (e.g., implicature (Degen, 2015)). In the "Only" sub-dataset (e.g., "Susan only writes." / "Susan writes."), samples differed in one main aspect: For some, the same subject appeared in both the premise and hypothesis. For others, there was no subject agreement. We found that 88% of the correctly predicted samples had the same subject with a majority of "entailment" predictions (55%). Furthermore we noticed that among the incorrectly predicted samples with same subjects, 75% were also predicted as "entailment". This suggests that the model might have been exploiting cues on subject agreement to make its predictions.

In the Question sub-dataset (e.g. "Did Bill wonder when Omar hunted?" / "Omar hunted."), in 51% of samples, word-to-word phrases/expressions appeared in both the premise and hypothesis. The model predicted entailment in 96% of these cases. However, the accuracy was lower—only 72.54%.

5.3 Exploiting Structural Information

In several datasets, a common pattern found in the premise is the structure "if ..., it's okay" (e.g. Premise: "If Amanda had left, it's okay." / Hypothesis: "Amanda used to be here."). In the Change of State dataset, in 81.25% of such samples, the model wrongly predicted "neutral". Similarly, in the Possessed Definites dataset, in 85% of such samples, the model wrongly predicted "entailment".

In the Cleft Uniqueness dataset, another common pattern is the structure "it is ... who ..." in the hypothesis (e.g.: Premise: "It is Sandra who disliked Veronica." / Hypothesis: "Exactly one person disliked Veronica."). For 76.9% of such samples, the model predicted entailment—all incorrectly. Similarly, samples where the hypothesis had "exactly one" were predicted as contradiction 70% of the time–again, all incorrect.

In all these cases, the model seemed to have learned incorrect associations between syntactic patterns and presuppositional entailment decisions.

6 Conclusion

We investigated BERT's capabilities to perform NLI on cases involving presupposition. Our analysis suggests that NLI-trained BERT exploits lexical and structural cues to do so, and that these cues are highly task-specific and do not generalize to adversarially perturbed versions of the input.

Acknowledgements

The authors would like to thank the reviewers for their valuable comments. This work is partially supported by the Canada CIFAR AI Chair Program held by Jackie CK Cheung at Mila. Jad Kabbara is also supported by an IVADO PhD fellowship.

References

Samuel R. Bowman, Gabor Angeli, Christopher Potts, and Christopher D. Manning. 2015. A large annotated corpus for learning natural language inference.

In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 632–642, Lisbon, Portugal. Association for Computational Linguistics.

Alexis Conneau, Ruty Rinott, Guillaume Lample, Adina Williams, Samuel Bowman, Holger Schwenk, and Veselin Stoyanov. 2018. XNLI: Evaluating cross-lingual sentence representations. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 2475–2485, Brussels, Belgium. Association for Computational Linguistics.

Ido Dagan, Oren Glickman, and Bernardo Magnini. 2005. The pascal recognising textual entailment challenge. In *Machine Learning Challenges Workshop*, pages 177–190. Springer.

Marie-Catherine De Marneffe, Mandy Simons, and Judith Tonhauser. 2019. The commitmentbank: Investigating projection in naturally occurring discourse. In *Proceedings of Sinn und Bedeutung*, volume 23, pages 107–124.

Judith Degen. 2015. Investigating the distribution of some (but not all) implicatures using corpora and web-based methods. *Semantics and Pragmatics*, 8:11–1.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, Minneapolis, Minnesota. Association for Computational Linguistics.

Ali Emami, Paul Trichelair, Adam Trischler, Kaheer Suleman, Hannes Schulz, and Jackie Chi Kit Cheung. 2019. The KnowRef coreference corpus: Removing gender and number cues for difficult pronominal anaphora resolution. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 3952–3961, Florence, Italy. Association for Computational Linguistics.

Matt Gardner, Yoav Artzi, Victoria Basmov, Jonathan Berant, Ben Bogin, Sihao Chen, Pradeep Dasigi, Dheeru Dua, Yanai Elazar, Ananth Gottumukkala, Nitish Gupta, Hannaneh Hajishirzi, Gabriel Ilharco, Daniel Khashabi, Kevin Lin, Jiangming Liu, Nelson F. Liu, Phoebe Mulcaire, Qiang Ning, Sameer Singh, Noah A. Smith, Sanjay Subramanian, Reut Tsarfaty, Eric Wallace, Ally Zhang, and Ben Zhou. 2020. Evaluating models' local decision boundaries via contrast sets. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 1307–1323, Online. Association for Computational Linguistics.

Matthew Honnibal, Ines Montani, Sofie Van Landeghem, and Adriane Boyd. 2020. spaCy: Industrial-strength Natural Language Processing in Python.

Paloma Jeretic, Alex Warstadt, Suvrat Bhooshan, and Adina Williams. 2020. Are natural language inference models IMPPRESsive? Learning IMPlicature and PRESupposition. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 8690–8705, Online. Association for Computational Linguistics.

Diederik Kingma and Jimmy Ba. 2015. Adam: A method for stochastic optimization. In *Proceeding of the 2015 International Conference on Learning Representation (ICLR 2015)*, San Diego, California.

Alice Lai, Yonatan Bisk, and Julia Hockenmaier. 2017. Natural language inference from multiple premises. In *Proceedings of the Eighth International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 100–109, Taipei, Taiwan. Asian Federation of Natural Language Processing.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Roberta: A robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692*.

Christopher Manning. 2006. Local textual inference : It's hard to circumscribe , but you know it when you see it - and nlp needs it.

Tom McCoy, Ellie Pavlick, and Tal Linzen. 2019. Right for the wrong reasons: Diagnosing syntactic heuristics in natural language inference. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 3428–3448, Florence, Italy. Association for Computational Linguistics.

Timothy Niven and Hung-Yu Kao. 2019. Probing neural network comprehension of natural language arguments. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 4658–4664, Florence, Italy. Association for Computational Linguistics.

Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Kopf, Edward Yang, Zachary DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. 2019. Pytorch: An imperative style, high-performance deep learning library. In *Advances in Neural Information Processing Systems 32*, pages 8024–8035. Curran Associates, Inc.

Marco Tulio Ribeiro, Tongshuang Wu, Carlos Guestrin, and Sameer Singh. 2020. Beyond accuracy: Behavioral testing of NLP models with CheckList. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 4902–4912, Online. Association for Computational Linguistics.

Sebastian Schuster, Yuxing Chen, and Judith Degen. 2020. Harnessing the linguistic signal to predict scalar inferences. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 5387–5403, Online. Association for Computational Linguistics.

Koustuv Sinha, Prasanna Parthasarathi, Joelle Pineau, and Adina Williams. 2021. UnNatural Language Inference. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 7329–7346, Online. Association for Computational Linguistics.

Adina Williams, Nikita Nangia, and Samuel Bowman. 2018. A broad-coverage challenge corpus for sentence understanding through inference. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 1112–1122, New Orleans, Louisiana. Association for Computational Linguistics.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Rémi Louf, Morgan Funtowicz, Joe Davison, Sam Shleifer, Patrick von Platen, Clara Ma, Yacine Jernite, Julien Plu, Canwen Xu, Teven Le Scao, Sylvain Gugger, Mariama Drame, Quentin Lhoest, and Alexander M. Rush. 2019. Huggingface's transformers: State-of-the-art natural language processing. *ArXiv*, abs/1910.03771.

Re-Examining FactBank: Predicting the Author's Presentation of Factuality

John Murzaku[⋆◇], **Peter Zeng**[⋆], **Magdalena Markowska**[♣◇], **Owen Rambow**[♣◇]
[⋆] Department of Computer Science [♣] Department of Linguistics
[◇] Institute for Advanced Computational Science
Stony Brook University, Stony Brook, NY, USA
Corresponding Author: jmurzaku@cs.stonybrook.edu

Abstract

We present a corrected version of a previous projection of the FactBank data set. Previously published results on FactBank are no longer valid. We perform experiments on FactBank using multiple training paradigms, data smoothing techniques, and polarity classifiers. We argue that f-measure is an important alternative evaluation metric for factuality. We provide new state-of-the-art results for four corpora including FactBank. We perform an error analysis on Factbank combined with two similar corpora.

1 Introduction

The term *factuality*[1] refers to an author's presentation of a proposition (who-did-what-to-whom) as a fact, i.e., she is committed to the truth of the proposition. A lot of language use introduces propositions that are not presented as facts but as only possibly true, as a wish or as a hypothesis, or as someone else's belief. If we want to understand what an author is communicating, we need to distinguish these cases. Over the last 15 years, this question has received a lot of attention. Multiple corpora have been created, and these corpora have been used to explore machine learning architectures for factuality prediction. The machine learning studies often report results on all corpora, but these studies do not examine what the machine learning architecture can learn, nor how and why combining corpora can help. A notable exception is Jiang and de Marneffe (2021), who carefully analyze what exactly is learned from the CB corpus. In this paper, our goal is to determine how to combine corpora in order to maximize performance, and to understand why the specific combination of corpora works better than others. We choose a single resource to focus on so that we can gain insights by performing a careful study, and we choose FactBank (Saurí and Pustejovsky, 2009) because it is one of the first carefully constructed datasets for factuality prediction. We show how insights gained from working with FactBank can be used to improve performance on the CB corpus (de Marneffe et al., 2019).

There are three main contributions of this work:

(i) The FactBank data is complex. In order to facilitate NLP research, a machine learning-friendly projection from FactBank had been previously created and widely used. We correct an error in this projection (this data set will be made available). The error means that all recent results on FactBank are not valid.

(ii) We present new state-of-the-art results on the CB, FactBank, MV, and UW corpora.

(iii) We do an error analysis to show why some of these corpora perform better when combined.

This paper does not introduce new machine learning architectures; instead, we show that careful reexamination of the data can lead to improved performance without the necessity of introducing new, more complex architectures.

The paper is organized as follows. A survey of previous work is provided in Section 2. We summarize the FactBank representation of factuality in Section 3 and present our correction to the projection from FactBank in Section 4. In Section 5, we discuss metrics for evaluating factuality prediction. The redone experiments from (Jiang and de Marneffe, 2021) with our corrected projection are described in Section 6. We then report on machine learning experiments on three corpora: FactBank in Section 7, the CB corpus in Section 8, and the LDC corpus in Section 9. We conclude in Section 10.

[1]The notion of "factuality" is closely related to the notion of "belief" as used in cognitive science and AI; they differ only in the case of lying, where the author presents propositions as facts contrary to what she actually believes. See (Prabhakaran et al., 2015) for a fuller discussion.

Proceedings of the 29th International Conference on Computational Linguistics, pages 786–796
October 12–17, 2022.

2 Related Work

Corpora Many corpora explore the notion of *factuality* including: FactBank (Saurí and Pustejovsky, 2009), LU (Diab et al., 2009), UW (Lee et al., 2015), LDCCB (LDC) (Prabhakaran et al., 2015), MEANTIME (MT) (Minard et al., 2016), MegaVeridicality (MV) (White et al., 2018), UDS-IH2 (UD2) (Rudinger et al., 2018), CommitmentBank (CB) (de Marneffe et al., 2019), and RP (Ross and Pavlick, 2019). These corpora differ along several dimensions; we list dimensions which are salient for this paper.

(1) What type of data is used to build the corpus and is the data manipulated or not. E.g., MV selects only 6 syntactic frames and lexically "bleaches" them. In CB, only sentences with finite clausal complements are chosen for annotation. In addition, the matrix predicates must appear in the entailment cancelling environment, i.e. questions and negations preceded by a modal and/or in the antecedent of a conditional. In contrast, FactBank tags all events introduced in a corpus of complete, naturally occurring texts.

(2) What is the genre of the underlying texts? For those corpora which use naturally occurring texts, FactBank and UW use newswire exclusively. MT uses Wiki articles. RP uses data from textual inference corpora. CB uses newswire, fiction and dialog. UD2 uses weblogs, newsgroups, email, reviews, and question-answer corpora. LDC is exclusively discussion forum threads.

(3) The definition of an annotatable event. E.g., in MV only past events are taken into consideration; in UD2, both past and present events; UW, FactBank, CB and LDCCB consider also future events.

(4) Who are the annotators? FactBank and LDCCB used trained annotators, while CB and UW argue that crowd-sourced judgements collected from naive annotators are as (or more) valuable.

(5) The annotation scale. Annotations can be numerical values (often derived from averaging naive annotators' judgments), typically [-3,3] (CB, UW, RP or UD2), lexically represented values, such as [yes, maybe or maybe not, no] in MV, or categorial labels (FactBank and LDCCB). We discuss these categorial labels in more detail in Section 3, and give examples.

Stanovsky et al. (2017) unify the representation across datasets up to 2017 by mapping the discrete annotations of factuality in FactBank and MT onto the continuous scale used in UW. Furthermore, they also remove the FactBank non-author perspective annotations since none of the other corpora include such annotations (FactBank also annotates the beliefs of agents mentioned in a sentence, according to the author). This process will be discussed in more details in Section 4.

Experiments Early work on event factuality prediction used rule-based systems; for example, Nairn et al. (2006) propose a recursive polarity propagation algorithm which uses implication signatures from clause-embedding verbs. Lotan et al. (2013) predict factuality using implication signatures combined with lexical and dependency tree features.

Early machine learning work on event factuality prediction consists of SVMs or other supervised learning approaches. Diab et al. (2009) and Prabhakaran et al. (2010) use SVMs and CRFs along with lexical and dependency tree features for predicting author belief, evaluating on f-measure. Lee et al. (2015) also use an SVM along with lexical and dependency tree features on the UW corpus which they created, and evaluate on Pearson correlation and mean average error (MAE), as does all following work. Stanovsky et al. (2017) use SVMs combined with the output of the system of Lotan et al. (2013), and evaluate on Factbank, UW, and Meantime. Rudinger et al. (2018) use bidirectional LSTMs with tree or linear architectures and multiple task-specific training setups. Building on that work, Veyseh et al. (2019) use BERT sentence representations combined with a graph convolutional network that leverages the semantic and syntactic structure of the sentence, evaluating on Factbank, UW, Meantime, and UD2. At the time of publication, their system produced state-of-the-art results for Pearson correlation on FactBank, UW, Meantime, UD2, and state-of-the-art results for MAE on FactBank, Meantime, and UD2, with Stanovsky et al. (2017) still having the lowest MAE for UW. We discuss Jiang and de Marneffe (2021) in more detail in Section 6.1.

Our work differs from the related work by offering two salient contributions: first, we analyze specific corpus combinations to help improve on a specific corpus (FactBank) instead of focusing on improving all corpora. Second, we perform error analyses on corpus combinations to determine why and where factuality corpora can help each other or why factuality corpora can be incompatible.

3 Representation of Factuality

As discussed in Section 2, some corpora represent the factuality judgments using numbers in an interval, while others use categorial labels with a defined meaning. The corpus creators determine that there are several distinct categories that annotators can identify. In the examples (all from FactBank, some simplified), the head of the phrase which presents the evaluated situation is bolded.

Certain (FactBank label CT): the author commits to the truth of the presented situation. Note that the commitment is independent of tense.

(1) A lawsuit in Germany will **seek** a criminal prosecution of the Defense Secretary.

Probable (FactBank label PR): the author presents the situation as probable.

(2) Saddam appeared to **accept** the treaty.

Possible (FactBank label PS): the author presents the situation as possible.

(3) He won't be under control until he is **committed** to an institution.

Fully underspecified (FactBank label UU): The source does not know what is the factual status of the event, or does not commit to it.

(4) The minister denied the kingdom had **notified** its customers.

In addition, FactBank annotates polarity on types of author belief, i.e., whether the author presents the situation as factual, or the absence of the situation as factual. Polarity is only added to the non-UU values, resulting in a label set containing seven labels, CT-, PR- PS-, UU, PS+, PR+, CT+. We discuss our mapping between the categorical labels and numerical labels in Section 7 (see Table 4).

4 Corrected FactBank Dataset

Label Correction Stanovsky et al. (2017) developed a projection from the complex FactBank corpus to a CoNLL-formatted file that includes only factuality judgments by the author, enabling an annotation with a single value at the word level. This data set has been extensively used in NLP experiments (see Section 2). We have found that there is a systematic error in this FactBank data set projection. Consider (4) above. FactBank annotates the denial event as seen as factual by the author (CT+). The notification event is annotated twice: according to the author, the minister sees it as certainly false (CT-). The author herself does not express her view of the factuality and her perspective is labeled UU, as explained in Section 3. The old projection of the FactBank data set incorrectly picks up the CT- label. We correct the projection by sticking to the author's perspective and supply the UU label. Table 2 shows the shift in label distribution percent towards UU as a result of our correction.

Article Split Correction Furthermore, Stanovsky et al. (2017) do not split by article, meaning that data from the same article could appear in both the training and test sets. We re-split the data, this time assigning all annotations from a single article to the same split, until we approximate a standard ratio of 70-20-10 for train-dev-test. With the new projection of FactBank and article-based split, we found that when training and testing on FactBank, there is a 2% decrease in Pearson correlation and a 10% increase in MAE compared to the previous projection of FactBank without the article-based split.

We will make the correct projection of the FactBank data set available, see Appendix A for details.

5 Evaluation of Factuality Prediction

To date, the category labels in FactBank have been translated to numbers for training and testing (from -3 for CT- to +3 for CT+). A regression head predicts a number, and the results are evaluated with the Pearson correlation r between the predicted and gold numbers. The publications to date also provide MAE (which need not correlate with correlation). We propose an additional evaluation, namely f-measure on the categories and macro averaged f-measure.

Usually in NLP, there is no single correct evaluation metric: the best evaluation metric to use depends on the downstream use we want to make of the module we are evaluating. In some applications, we need to know the specific level of commitment of the writer. For example, in argumentation analysis we need to detect claims to which the writer is fully committed, and when analyzing hedging as a marker of politeness or power structure, we need to identify the PR/PS family. But the numerical evaluation makes the same difference between PR+ and PS+ on the one hand, and PR+ and CT+ or PS+ and UU on the other hand. F-measure clearly separates these cases. In this paper, we show results using correlation, MAE, and the f-measures (macro-averaged F1 and per-label F1).

	FB-Old (NS)		FB-New (NS)		FB-New (S)		Previous SOTA	
Test Set	**r ↑**	**MAE↓**	**r↑**	**MAE↓**	**r↑**	**MAE↓**	**r↑**	**MAE↓**
CB	0.890	0.617	0.908	0.613	0.906	0.561	0.890	0.617
RP	0.870	0.608	0.856	0.630	0.861	0.642	0.870	0.608
MV	0.857	0.533	0.867	0.498	0.886	0.483	0.876	0.501
MT	0.491	0.319	0.456	0.311	0.553	0.281	0.702	0.204
UW	0.865	0.351	0.879	0.366	0.874	0.348	0.868	0.349
UDSIH2	0.853	0.766	0.855	0.763	0.857	0.758	0.909	0.726
FB-New	-	-	0.858	0.359	0.866	0.330	-	-

Table 1: Results on multi-task training with no smoothing (NS), smoothing (S), and the new projection of FactBank (FB-New); a dark shaded cell indicates the best published result to date in this table; light shading means an improvement over FB-Old (NS). Results for all corpora except RP show improvement. All other state of the art results for non-shaded cells are held by Jiang and de Marneffe (2021) or Veyseh et al. (2019).

	CT+	PR+	PS+	UU	PS-	PR-	CT-
Old	75.3	3.1	2.2	14.5	0.1	0.6	4.2
Corrected	57.1	1.1	1.1	38.3	0.1	0.1	2.2

Table 2: Distribution of each FactBank annotation label in the old and corrected CoNLL-formatted data set

6 Evaluation on All Corpora

6.1 Redoing (Jiang and de Marneffe, 2021)

Jiang and de Marneffe (2021) provide state-of-the-art results for many of the corpora discussed in Section 2 using a simple architecture. In this section, we redo their experiments with the updated FactBank data set. Specifically, we redo the multi-task learning experiments using the same underlying architecture (the SelfAttentiveSpanExtractor developed by Gardner et al. (2018)) and the same training parameters as Jiang and de Marneffe (2021). This training paradigm allows all corpora to share the same BERT parameters, but with each corpus having a regression head with corpus-specific parameters. The authors find that fine-tuning on BERT-large performs best. However, with the corrected FactBank dataset, we find that RoBERTa-large outperforms BERT-large, and we therefore use it for our experiments. For each experiment, we do three runs with different seeds and report the average for Pearson correlation and MAE, and for most experiments we also provide the standard deviation.

Table 1 shows results with the old and corrected data sets. We replicate the results from Jiang and de Marneffe (2021) using the faulty dataset used in previous experiments (columns **FB-Old (NS)**). For our corrected dataset (columns **FB-New (NS)**), results for all test sets other than RP improve, presumably because the FactBank data is now more in line with the other corpora. Note that none of the results in this paper for FactBank are comparable to any previously published results because of the errors in the FactBank data set used to date (see Section 4), which means that the FactBank test set has also changed.

6.2 Addressing Imbalances in Corpora

One problem with all corpora in this study, including FactBank, is the inequality in the label distribution, with a majority of CT+ (3.0) and UU (0.0) as shown in Table 2 (this effect holds in all corpora, including those with purely numerical labels). We address this issue by performing label distribution smoothing and modifying the loss function to a weighted SmoothL1 loss.

Yang et al. (2021) provide methods to address class imbalance problems in a continuous setting using label distribution smoothing and feature distribution smoothing, which directly applies to our regression task. We apply their method of label distribution smoothing (LDS) to our datasets by using kernel density estimation to learn the effective label density in our dataset. We then re-weight the SmoothL1 loss function by multiplying it by the inverse of the effective label densities learned. This method improves on all of our tasks, so we perform all of our regression experiments using LDS unless otherwise noted. Results for redoing Jiang and de Marneffe (2021) experiments with LDS are shown in Table 1 (columns **FB-New (S)**). We see an improvement compared to no LDS (columns **FB-New (NS)**) for all corpora on both metrics (correlation and MAE), except for correlation on CB and UW.

Single	r↑	MAE↓	r↑	MAE↓
FB	$0.872_{\pm 0.002}$	$0.276_{\pm 0.004}$	$0.872_{\pm 0.002}$	$0.276_{\pm 0.004}$
	Shared		**MTL**	
FB+CB	$0.876_{\pm 0.011}$	$0.293_{\pm 0.029}$	$0.873_{\pm 0.008}$	$0.292_{\pm 0.015}$
FB+MV	$0.874_{\pm 0.000}$	$0.289_{\pm 0.028}$	$0.885_{\pm 0.005}$	$0.274_{\pm 0.034}$
FB+RP	$0.871_{\pm 0.005}$	$0.293_{\pm 0.013}$	$0.879_{\pm 0.005}$	$0.311_{\pm 0.009}$
FB+MT	$0.864_{\pm 0.010}$	$0.284_{\pm 0.011}$	$0.875_{\pm 0.007}$	$0.334_{\pm 0.039}$
FB+UD2	$0.818_{\pm 0.023}$	$0.386_{\pm 0.037}$	$0.867_{\pm 0.007}$	$0.360_{\pm 0.009}$
FB+LDC	$0.802_{\pm 0.075}$	$0.343_{\pm 0.066}$	$0.868_{\pm 0.010}$	$0.329_{\pm 0.039}$
FB+UW	$0.741_{\pm 0.034}$	$0.717_{\pm 0.080}$	$0.873_{\pm 0.007}$	$0.289_{\pm 0.023}$
FB+CB+MV	$0.881_{\pm 0.001}$	$0.278_{\pm 0.005}$	-	-
FB+CB+MV+RP	$0.873_{\pm 0.007}$	$0.316_{\pm 0.005}$	-	-
FB+MV+RP	-	-	$0.879_{\pm 0.011}$	$0.305_{\pm 0.022}$

Table 3: Results for our regression experiments on FactBank. Single results show FactBank trained and tested on itself. The Shared and MTL columns show Pearson r and MAE on the two training paradigms respectively. A shaded cell indicates the best performing combination; light means only a slight improvement.

7 Experiments on FactBank

In this section, we evaluate exclusively on FactBank (using our new FactBank dataset described in Section 4).

Our goal is to provide the best results on FactBank. For all experiments reported in this section, we follow the same setup as Section 6, except that we train two models per setup, one for regression which we evaluate numerically (r and MAE), and one for classification which we evaluate using F1. In classification, we collapse FactBank labels PR and PS (probable and possible), as they are rare, and their distinction is less important. Table 4 shows the mappings of the continuous labels to discrete labels with the corresponding factuality values (in FactBank terms).

Range	Label
[-3.0, -2.5]	CT-
(-2.5, -0.5]	PR-
(-0.5, 0.5)	UU
[0.5, 2.5)	PR+
[2.5, 3.0]	CT+

Table 4: Mappings for our classification model.

Following Jiang and de Marneffe (2021), we perform our training in two ways: **Shared**, where we combine the corpora's training data together and test on Factbank; and multi-task learning (**MTL**), where corpora share the same RoBERTa-large parameters, but we have a corpus-specific regression or classification head for FactBank. All experiments are performed three times with different seeds, and we report the average and standard deviation. For further information on the experimental setup, see Appendix B.

7.1 Regression Experiments

We perform corpus combinations of FactBank with each of the other corpora, and evaluate on Pearson correlation and MAE. We then do a greedy search with the top performing corpus combinations: we take our best performing system and add the corpus which performs next best in the 2-way combination with FactBank. If adding another corpus does not yield an improvement, we stop our search. All results for both the **Shared** and **MTL** experiments are shown in Table 3.

Shared: FactBank combined with CB and FactBank combined with MV yield improvements in correlation, but not in MAE. In our greedy search, we combine FactBank with both CB and MV. FactBank combined with CB and MV performs the best on correlation achieving a result of 0.881. However, this corpus combination results in a slightly higher MAE compared to baseline FactBank only. We then add RP, but FB+CB+MV+RP performs worse and we end our greedy search. The worst performing corpus combination on FactBank is FB+UW, resulting in the lowest correlation and also a very high MAE. This is because of article overlap in FactBank and UW with different annotation labels as shown in Lee et al. (2015), leading to a divergence.

MTL: The top performing result on FactBank is with MTL on FB+MV, with a Pearson correlation of 0.885 and MAE of 0.274. Each corpus combination besides FB+UD2 results in an im-

provement over baseline FactBank. MTL improves on the worst performing corpus combinations in the Shared paradigm (FB+UW and FB+UD2): the FactBank specific corpus head in the model is optimized for FactBank, and therefore addresses the lack of performance caused by treating all corpora as one. We perform a greedy search by adding RP to our top performing MTL combination of FB+MV. The correlation is higher than FactBank Single, but not higher than the FB+MV combination, so we stop our greedy search.

7.2 Classification Experiments

We perform all classification experiments with the same hyperparameters and training architecture as our regression experiments, but with a classification head instead of a regression head. We mention more details about this in Appendix B. Even after collapsing PR±/PS± labels, PR±/PS± are a minority class in our classification experiments. We address the label imbalance in our classification models by using focal loss (Lin et al., 2017), which has been shown to perform well on imbalanced classification tasks compared to cross-entropy loss. Using the focal loss function allows our model to focus on the harder to classify PR±/PS± examples. All results for both the **Shared** and **MTL** experiments are shown in Table 5.

Shared: The largest increase is achieved by combining FactBank with CB, with a major boost in macro-average and in the per label F1 of the minority labels PR± because CB introduces many new PR± labels. The only other corpus that achieves an increase on FactBank is RP, which specifically helps in the UU and CT+ labels. All other corpora do not help, notably UD2 and UW, which result in a massive decrease. Some per-label F1s are 0 for FB+LDC. The LDC corpus does not contain polarity, and therefore has no labels in the CT- and PR- categories. We perform a greedy search on our top performing corpora combinations, adding RP to FB+CB, but do not improve, so we stop our greedy search.

MTL: Again, CB helps the most, with the highest macro-average on FactBank in our 5-way system, specifically helping with the minority classes PR- and PR+. MT, UW, and MV also provide a boost. The previous poor performance of UW is fixed by training in a MTL setting. All other corpora decrease performance on FactBank. UD2 performs poorly, but slightly better than in the shared setting. We perform a greedy search and stop after adding MT to yield FB+CB+MT, which does not improve on our top combination of FB+CB.

7.3 End-to-end Evaluation

One advantage of the F1 evaluation is that we can provide a single end-to-end evaluation on data without gold heads. All experiments reported in the literature using r and MAE assume a gold head, since otherwise these measures cannot be computed. We introduce a “Not-Head" tag (O) for all words that are not heads and train a model on all words in the data set, not just the heads. We repeat the experiments with the same architecture and setup as the previous classification experiments. The results are shown in Table 6, and as expected, the F1s for each class decrease, but not dramatically. The F1 for detecting that a word is a head is 0.888; whether a noun is an event is context-dependent (e.g., *construction*) and can be hard to determine. Since the head-identification task is not trivial, we argue that an evaluation on F1 is therefore important and can offer broader insights into the factuality-prediction task.

7.4 Factoring Polarity

Polarity is often expressed independently of the degree of factuality, as illustrated in this constructed example: *Sudeep {probably/maybe/()} {came/did not come} to dinner.* Here, the first set of curly brackets lists three options for factuality, and the second set of curly brackets determines polarity. All six combinations are plausible sentences, determining six different FactBank labels for the coming event. Of course, some lexical items precisely encode a combination of polarity and factivity level (such as *deny* in (4)).

We train a polarity classifier on the same training data, but this time we label all negative data (CT-, PR-, PS-) as a new label NEG, the neutral data remains UU, and the positive data (PS+, PR+, CT+) as POS. The classifier performs with per-label F1-measure of 0.907 (Neg), 0.824 (UU), and 0.940 (Pos). We then create a combined system with our polarity classifier and our 5-way classification system where the polarity classifier assigns the polarity of the head, and the classification system predicts the strength. Our results for this system on FactBank are shown in Table 7.

On our FactBank-only system, the polarity classifier results in a 28% error reduction on macro-average, while also stabilizing results by lower-

Single	Macro-F1	CT-	PR-	UU	PR+	CT+	Macro-F1	CT-	PR-	UU	PR+	CT+
FB	0.701$_{\pm 0.076}$	0.863	0.222	0.893	0.593	0.935	0.701$_{\pm 0.076}$	0.863	0.222	0.893	0.593	0.935
	Shared						MTL					
FB+CB	0.790$_{\pm 0.053}$	0.863	0.611	0.885	0.662	0.932	0.800$_{\pm 0.017}$	0.851	0.667	0.880	0.677	0.930
FB+RP	0.717$_{\pm 0.089}$	0.843	0.444	0.889	0.475	0.936	0.655$_{\pm 0.009}$	0.874	0.000	0.888	0.581	0.933
FB+MT	0.703$_{\pm 0.088}$	0.852	0.222	0.894	0.614	0.936	0.749$_{\pm 0.088}$	0.861	0.389	0.895	0.668	0.937
FB+MV	0.679$_{\pm 0.1200}$	0.833	0.222	0.883	0.530	0.931	0.705$_{\pm 0.064}$	0.818	0.222	0.880	0.680	0.930
FB+LDC	0.631$_{\pm 0.027}$	0.759	0.000	0.887	0.580	0.930	0.664$_{\pm 0.010}$	0.847	0.000	0.886	0.657	0.932
FB+UD2	0.566$_{\pm 0.026}$	0.690	0.060	0.864	0.292	0.923	0.624$_{\pm 0.000}$	0.846	0.000	0.883	0.462	0.929
FB+UW	0.546$_{\pm 0.015}$	0.797	0.122	0.705	0.212	0.894	0.744$_{\pm 0.084}$	0.866	0.444	0.879	0.601	0.930
FB+CB+MT	-	-	-	-	-	-	0.693$_{\pm 0.051}$	0.867	0.222	0.891	0.551	0.935
FB+CB+RP	0.704$_{\pm 0.076}$	0.855	0.355	0.888	0.489	0.934	-	-	-	-	-	-

Table 5: Results for our classification experiments on FactBank. The topmost results show FactBank trained and tested on itself as a baseline. The Shared and MTL columns show F1 and per-label F1 on the two training paradigms respectively. A shaded cell indicates the best performing combination; light means only a slight improvement.

Macro-F1	CT-	PR-	UU	PR+	CT+	O
0.727	0.519	0.667	0.735	0.714	0.767	0.961

Table 6: Results for our end-to-end classification system on FactBank+CB.

Train	NoPC Macro-F1	PC Macro-F1	ER%
FB	0.701$_{\pm 0.077}$	0.786$_{\pm 0.003}$	28
FB+LDC	0.665$_{\pm 0.011}$	0.776$_{\pm 0.017}$	33

Table 7: Results on macro-average for our polarity classifier jointly combined with our 5-way classifier for without polarity (NoPC Macro-F1), with polarity (PC Macro-F1), and error reduction (ER).

ing standard deviation. The polarity classifier also helps other combinations. Our highest error reduction is for FB+LDC, presumably because LDC does not contain polarity annotations. We achieve a 33% error reduction in MTL training and 29% error reduction in Shared training; on regression, we also achieve an error reduction of 33%. There is also a moderate error reduction in classification for FB+UW (19% in Shared training and 11% in MTL training) and FB+UD2 (16% in Shared training and 10% in MTL training). For MV, we only get an error reduction in the MTL training setting of 10%. However, we do not obtain error reductions using this technique for our best performing combinations.

7.5 Why Does CB Help FactBank?

FactBank is all newswire, and CB also contains newswire. This does not fully explain why CB helps with FactBank. To further examine the issue, we consider two models, the model trained only on FactBank (FB) and the MTL classification model trained on FactBank and CB (FB+CB). We perform an error analysis on the data points in the FactBank dev set on which the two models make different predictions. Our goal is to determine how using the CB corpus helps. The results are in Table 8. We use the following categories; the first five are morpho-syntactic. **Noun** means that the target is a noun designating an event not tagged UU in FactBank; **Noun-UU** is a noun which is tagged UU. **Main** refers to heads that are main clause verbs or verbs in adjunct clauses to the main verb; these are typically easy cases. **Embedded** refers to targets which are in complement clauses below a main clause verb; the factuality status is typically determined or strongly affected by the main clause verb. **Relclause** refers to heads which are in relative clauses. **Hypo** are hypothetical situations. **Idiom** groups together various cases of idiomatic language use, either multiword expressions or idiomatic usages of lexical items. **Misc** groups together various other syntactic and semantic special cases.

We observe that 10% of all errors are gold errors (i.e. errors in the original annotation), and furthermore, that FB makes more errors in total than FB+CB, since FB+CB performs better. (The percentages do not sum to 100% in each row since some errors are made by both models.) Recall that CB annotates only verbs in embedded clauses. We therefore expect the FB+CB model to perform poorly on main clause verbs (**Main**) and nouns, which is borne out. The exception is nouns labeled UU, since the FB+CB model appears to label most

Type	Nb	FB	FB+CB	Gold
Noun	23	22%	70%	9%
Noun-UU	4	100%	0%	0%
Main	3	0%	100%	0%
Embedded	17	65%	24%	18%
Rel clause	9	78%	22%	0%
Hypo	4	100%	0%	0%
Idiom	6	67%	17%	17%
Misc	16	69%	44%	13%
Total	82	56%	40%	10%

Table 8: Error analysis on differences between FactBank-only model and FB+CB model; percentages refer to portion of all the errors of that type made by the two systems and Gold; percentages in one row can sum to more than 100% because the same error can be made by both systems, or by a system and Gold

nouns as UU by default, thus getting UU nouns correct by accident. The **Misc** error category is balanced between the two models, and in all other models FB+CB performs better. The biggest such category is **Embedded**, which is precisely what CB annotates, and for which the FB+CB model has far fewer errors than the FactBank model alone (24% of 17 errors against 65%). The error analysis thus shows that by adding CB to the multi-task training, the resulting model has learned what CB is designed to provide information on (embedded verbs), but suffers from the lack of representative data distribution in CB and increases errors for categories that CB does not not annotate (main verbs, nouns).

The results of the error analysis suggest another type of system: we use the FactBank-only system for noun heads, and the FB+CB system for verb heads (the number of main verb errors is small, so we do not worry about syntax). We implement this system using the Spacy POS-tagger (Honnibal and Montani, 2017), and using the previously trained models. If the Spacy POS-tagger tags a head as a noun, we use the FactBank-only system; otherwise we use the FB+CB system. The results are shown in Table 9. We see that this strategy provides us with the best result for 5-way classification, decreasing macro-average error by 12% and improving on the CT-, UU, and PR+ labels. We also note that this system has the smallest standard deviation among all models that perform at baseline or above, suggesting that the system is consistent in its behavior. We also perform this method on FB+CB in regression, and obtain an increase in Pearson correlation over the results in Table 3 from 0.876 to 0.888 (error reduction of 10%) and a slight decrease in MAE from 0.293 to 0.286 (error reduction of 2%).

	Macro-F1	CT-	PR-	UU	PR+	CT+
FB+CB	$0.800_{\pm 0.017}$	0.851	0.667	0.880	0.677	0.930
FB+CB'	$0.824_{\pm 0.006}$	0.873	0.667	0.887	0.765	0.930

Table 9: Results on FactBank using FB+CB as a baseline and FB+CB with a POS switch (FB+CB'). We show macro-average (Macro-F1) and per-label F1 performance. Shaded cells indicate improvements.

7.6 Summary on Experimental Findings

Regression Experiments Our first insight is that the new projection of FactBank helps on all corpora as shown in Table 1 because this projection makes more sense for a system to learn. On our FactBank focused-experiments, we find that the CB corpus helps FactBank the most in Shared, while the MV corpus helps FactBank the most in MTL as shown in Table 3.

Classification Experiments We find that CB helps FactBank in both the Shared and the MTL setting, outperforming all other corpus combinations as shown in Table 5. Furthermore, we find that our best results (FB+CB) can be further improved by using a POS-based system and we see improvement on all metrics as shown in Table 9. Finally, we find that an end-to-end system has predictably lower performance on f-measure as shown in Table 6, while offering the advantage of not assuming gold heads.

8 Testing on CB

We have seen that the corpora that most help FactBank are CB and MV. Can FactBank and MV help CB? In Section 6, we trained on all corpora (following the lead of Jiang and de Marneffe (2021)). Here, we train on only CB, FactBank, and MV, and evaluate on CB. The results are shown in Table 10 for CB using multi-task learning; similar experiments testing on MV did not yield improvements over our new state-of-the-art results in Table 1. When training on FB+CB, we do not improve on Pearson correlation but slightly

Train	r	MAE
FB+CB	0.885	0.602
FB+CB+MV	0.913	0.536

Table 10: MTL training on CB, FactBank, testing on CB test set

improve on MAE compared to previous CB results. However, when training on FB+CB+MV MTL, we achieve state-of-the-art results on CB in both Pearson correlation and MAE, providing a 21% error reduction on both measures (coincidentally) over the state-of-the-art prior to this paper.

9 Testing on LDC

Experiments We perform a separate set of experiments with the LDCCB corpus (LDC) (Prabhakaran et al., 2015) and FactBank on LDC. We choose the LDC corpus because it is similar to FactBank with reference to the annotation goals and the use of expert annotators. One major difference, however, is genre. LDC consists of discussion forum posts with many typos and fragmentary language, while FactBank consists entirely of newswire. We carefully examine this combination and show why some factuality corpora, even if they have similar annotation goals and annotators, may be incompatible. Using the same system and experimental setup as Appendix B, we perform two experiments: first, we train on LDC and test on LDC, and second, we train on FactBank and test on LDC. Table 11 shows results for these two experiments respectively. We see that FactBank performs very poorly on LDC, highlighting a potential incompatibility and mirroring our results training on FB+LDC in Table 3 and testing on FactBank, where LDC was among the worst performers.

Train	**r**	**MAE**
LDC	0.822	0.361
FB	0.616	0.630

Table 11: Results on LDC corpus test-set trained on LDC and trained on FactBank (FB).

Error Analysis We perform an error analysis on the application of the FactBank-only model on LDC, choosing 100 errors randomly. We list the percentage of errors in each category. We use the same categories as in Section 7.5, though there are no errors on nouns, and we add the category **Unclear** which includes fragments, typos, grammar and spelling errors which are the effect of unedited text. First, the gold standard errors are strikingly high: out of 100 examples, 5 heads were labeled in error in gold, and 45 of the remaining examples are mislabeled in the gold standard. We assume that this percentage is not representative of the corpus, since these are the difficult cases, but it also may be that annotation is harder on informal domains. As expected for a model trained on edited newswire, **Unclear** is the top error category. Ignoring **Misc**, the next biggest error categories are **Embedded** and **Hypo**. We already saw this weakness in the FactBank-only model on hypotheticals in Table 8. In conclusion, the error analysis shows the importance of genre (**Unclear** errors), and the continued weakness of FactBank-trained models for hypotheticals and embedded clauses.

Type	%
Main	3
Embedded	12
Hypo	11
Unclear	13
Misc	11
Gold	50
Total number	100

Table 12: Error analysis on FactBank system prediction on LDC

10 Conclusion

After correcting an error in a widely used data set derived from FactBank, we report new best results on four corpora: FactBank (Table 3), CB (Table 10), MegaVeridicality (Table 1), and UW (Table 1). We also provide f-measure evaluation, and extend this to a true end-to-end evaluation, the first in the literature. Finally, we show that by combining compatible corpora (FactBank, CB, MV), we can achieve improvements in performance on FactBank and CB, and that the improvements on FactBank are precisely as expected given how CB was created.

Given the targeted help CB can provide on FactBank predictions for embedded clauses, and given the current weakness on hypotheticals in FactBank, we suggest a new targeted annotation of factivity on sentences in hypothetical contexts.

Acknowledgments

We thank: Bonnie Dorr and Brodie Mather for encouragement and interest; the three anonymous reviewers for their insightful and useful comments; Nan-Jiang Jiang for help with the Jiant system and the datasets; Amittai Aviram and Tyler Osborne for help with the data representation. This material is based on work supported by the Defense Advanced Research Projects Agency (DARPA) under Contracts No. HR01121C0186, No. HR001120C0037, and PR No. HR0011154158. Any opinions, findings and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of DARPA.

References

Marie-Catherine de Marneffe, Mandy Simons, and Judith Tonhauser. 2019. The commitmentbank: Investigating projection in naturally occurring discourse. *Proceedings of Sinn und Bedeutung*, 23(2):107–124.

Mona Diab, Lori Levin, Teruko Mitamura, Owen Rambow, Vinodkumar Prabhakaran, and Weiwei Guo. 2009. Committed belief annotation and tagging. In *Proceedings of the Third Linguistic Annotation Workshop (LAW III)*, pages 68–73, Suntec, Singapore. Association for Computational Linguistics.

Matt Gardner, Joel Grus, Mark Neumann, Oyvind Tafjord, Pradeep Dasigi, Nelson Liu, Matthew Peters, Michael Schmitz, and Luke Zettlemoyer. 2018. Allennlp: A deep semantic natural language processing platform. *arXiv preprint arXiv:1803.07640*.

Matthew Honnibal and Ines Montani. 2017. spaCy 2: Natural language understanding with Bloom embeddings, convolutional neural networks and incremental parsing. To appear.

Nanjiang Jiang and Marie-Catherine de Marneffe. 2021. He thinks he knows better than the doctors: BERT for event factuality fails on pragmatics. *Transactions of the Association for Computational Linguistics*, 9:1081–1097.

Kenton Lee, Yoav Artzi, Yejin Choi, and Luke Zettlemoyer. 2015. Event detection and factuality assessment with non-expert supervision. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 1643–1648, Lisbon, Portugal. Association for Computational Linguistics.

Tsung-Yi Lin, Priya Goyal, Ross Girshick, Kaiming He, and Piotr Dollár. 2017. Focal loss for dense object detection. In *Proceedings of the IEEE international conference on computer vision*, pages 2980–2988.

Amnon Lotan, Asher Stern, and Ido Dagan. 2013. TruthTeller: Annotating predicate truth. In *Proceedings of the 2013 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 752–757, Atlanta, Georgia. Association for Computational Linguistics.

Anne-Lyse Minard, Manuela Speranza, Ruben Urizar, Begoña Altuna, Marieke van Erp, Anneleen Schoen, and Chantal van Son. 2016. MEANTIME, the NewsReader multilingual event and time corpus. In *Proceedings of the Tenth International Conference on Language Resources and Evaluation (LREC'16)*, pages 4417–4422, Portorož, Slovenia. European Language Resources Association (ELRA).

Rowan Nairn, Cleo Condoravdi, and Lauri Karttunen. 2006. Computing relative polarity for textual inference. In *Proceedings of the fifth international workshop on inference in computational semantics (icos-5)*.

Vinodkumar Prabhakaran, Tomas By, Julia Hirschberg, Owen Rambow, Samira Shaikh, Tomek Strzalkowski, Jennifer Tracey, Michael Arrigo, Rupayan Basu, Micah Clark, Adam Dalton, Mona Diab, Louise Guthrie, Anna Prokofieva, Stephanie Strassel, Gregory Werner, Yorick Wilks, and Janyce Wiebe. 2015. A new dataset and evaluation for belief/factuality. In *Proceedings of the Fourth Joint Conference on Lexical and Computational Semantics*, pages 82–91, Denver, Colorado. Association for Computational Linguistics.

Vinodkumar Prabhakaran, Owen Rambow, and Mona Diab. 2010. Automatic committed belief tagging. In *Coling 2010: Posters*, pages 1014–1022, Beijing, China. Coling 2010 Organizing Committee.

Alexis Ross and Ellie Pavlick. 2019. How well do NLI models capture verb veridicality? In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 2230–2240, Hong Kong, China. Association for Computational Linguistics.

Rachel Rudinger, Aaron Steven White, and Benjamin Van Durme. 2018. Neural models of factuality. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 731–744, New Orleans, Louisiana. Association for Computational Linguistics.

Roser Saurí and James Pustejovsky. 2009. Factbank: a corpus annotated with event factuality. *Language Resources and Evaluation*, 43:227–268.

Gabriel Stanovsky, Judith Eckle-Kohler, Yevgeniy Puzikov, Ido Dagan, and Iryna Gurevych. 2017. Integrating deep linguistic features in factuality prediction over unified datasets. In *ACL*.

Amir Pouran Ben Veyseh, Thien Huu Nguyen, and Dejing Dou. 2019. Graph based neural networks for event factuality prediction using syntactic and semantic structures. *arXiv preprint arXiv:1907.03227*.

Alex Wang, Ian F. Tenney, Yada Pruksachatkun, Phil Yeres, Jason Phang, Haokun Liu, Phu Mon Htut, , Katherin Yu, Jan Hula, Patrick Xia, Raghu Pappagari, Shuning Jin, R. Thomas McCoy, Roma Patel, Yinghui Huang, Edouard Grave, Najoung Kim, Thibault Févry, Berlin Chen, Nikita Nangia, Anhad Mohananey, Katharina Kann, Shikha Bordia, Nicolas Patry, David Benton, Ellie Pavlick, and Samuel R. Bowman. 2019. `jiant` 1.3: A software toolkit for research on general-purpose text understanding models. `http://jiant.info/`.

Aaron Steven White, Rachel Rudinger, Kyle Rawlins, and Benjamin Van Durme. 2018. Lexicosyntactic inference in neural models. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 4717–4724, Brussels, Belgium. Association for Computational Linguistics.

Yuzhe Yang, Kaiwen Zha, Ying-Cong Chen, Hao Wang, and Dina Katabi. 2021. Delving into deep imbalanced regression. In *International Conference on Machine Learning (ICML)*.

A Distribution of New Data Set

We intend to distribute the corrected FactBank data set. We have included the training portion in this submission for reviewers to inspect, but we cannot distribute it for copyright reasons. Instead, we will provide a Python script which will produce the files submitted with this paper from the original FactBank files. These files can be obtained by researchers from the Linguistic Data Consortium, catalog number LDC2009T23. The entire corpus contains 9,740 annotated data points, split by article.

B Details on Experiments

We used a standard fine-tuning approach on existing BERT-large and RoBERTa large models with 333,843,458 and 355,623,938 paramaters respectively. For computing, we used our employer's GPU cluster. Compute jobs were typically about 25 minutes on average and ran on a single Tesla V100-SXM2 GPU. We did not do any hyperparameter search or hyperparameter tuning. We followed the same training parameters as Jiang and de Marneffe (2021), where we fine-tuned our model for at most 20 epochs with a learning rate of 1e-5. Early stopping was used if the difference between Pearson r and MAE did not increase, or if macro F1 did not increase. All metrics for experiments were averaged over three runs using fixed seeds (7, 21, and 42) which we will share with our code. We have also noted where testing for statistical significance of results was performed and have provided standard deviations for our results. To fine-tune the models and run experiments, we used the. jiant-v1-legacy library (Wang et al., 2019) and the implementation of Jiang and de Marneffe (2021) which uses jiant-v1-legacy. We added the classification module for the Jiang implementation and will make that available. All evaluation was performed by the jiant-v1-legacy library. All pre-processing scripts will be made available.

The Role of Context and Uncertainty in Shallow Discourse Parsing

Katherine Atwell[†], **Remi Choi**[†], **Junyi Jessy Li**[§], **Malihe Alikhani**[†]
[†]Department of Computer Science, University of Pittsburgh
[§]Department of Linguistics, The University of Texas at Austin
{kaa139, theresachoi, malihe}@pitt.edu,
jessy@utexas.edu

Abstract

Discourse parsing has proven to be useful for a number of NLP tasks that require complex reasoning. However, over a decade since the advent of the Penn Discourse Treebank, predicting implicit discourse relations in text remains challenging. There are several possible reasons for this, and we hypothesize that models should be exposed to more context as it plays an important role in accurate human annotation; meanwhile adding uncertainty measures can improve model accuracy and *calibration*. To thoroughly investigate this phenomenon, we perform a series of experiments to determine 1) the effects of context on human judgments, and 2) the effect of quantifying uncertainty with annotator confidence ratings on model accuracy and calibration (which we measure using the Brier score (Brier et al., 1950)). We find that including annotator accuracy and confidence improves model accuracy, and incorporating confidence in the model's temperature function can lead to models with significantly better-calibrated confidence measures. We also find some insightful qualitative results regarding human and model behavior on these datasets.

1 Introduction

The context of an utterance influences the interpretation. Linguistics, philosophy, cognitive science and neuroscience (Lewis, 1980; Glanzberg, 2002; Thompson-Schill, 2003) studies have shown the effect of context in text interpretation. In this paper, we hypothesize that context can influence the prediction of discourse relation labels. Motivating the need to discover the effects of context on annotation accuracy is the following example, where argument 1 is **bolded** and argument 2 is *italicized*:

(1) **the fund's 25% leverage has jacked up its interest income**
As long as I am borrowing at 9.9% and each {bond} yields over that, it enhances the yield

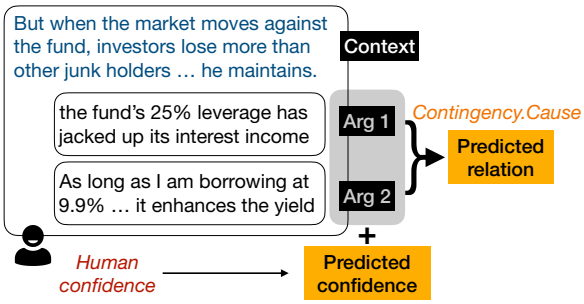

Figure 1: A diagram of our model, in which we use contextual information to predict uncertainty and pass the results into a model that predicts the discourse relation. We experiment with several different architectures and machine learning methods for utilizing uncertainty, including concatenation.

The lack of contextual information makes it difficult to determine the proper discourse relation here, and our annotator chose *Expansion.Conjunction* as a result. However, when more context is added, the passage presented to the annotator is as follows:

(2) But when the market moves against the fund, investors lose more than other junk holders because the market decline is magnified by the amount the fund is leveraged. Fund managers, for their part, defend their use of leverage. Carl Ericson, who runs the Colonial Intermediate High Income Fund, says **the fund's 25% leverage has jacked up its interest income**. "*As long as I am borrowing at 9.9% and each {bond} yields over that, it enhances the yield*," he maintains. Mr. Ericson says he tries to offset the leverage by diversifying the fund's portfolio.

This added context makes it more apparent that a causal relation holds between the two arguments, and thus our annotator chose the correct relation, *Contingency.Cause*, after being presented with this context.

Proceedings of the 29th International Conference on Computational Linguistics, pages 797–811
October 12–17, 2022.

While the annotation process of discourse corpora such as the Penn Discoruse Treebank (PDTB) exposes annotators to the full context of the document, we do not yet understand how human annotation behavior would change if context were limited. Additionally, context is limited for most of the models built for automatic discourse relation classification. We start with a basic question: is the argument pair (where arguments are defined as the minimum span a relation could be interpreted in PDTB (Prasad et al., 2008; Webber et al., 2019)) enough to determine the discourse relation?

In addition, context may affect human annotation to varying degrees and impact annotators' confidence in their judgements, how can we make sure that this information is factored into discourse parsers and model confidence? For the first time, we propose to study and measure the human annotator's confidence and incorporate it into the architecture of the deep-learning-based discourse parsers. We utilize our human-annotated confidence scores to predict human confidence, and test whether this method improves model accuracy and *calibration* (how well the predicted probabilities produced by a model reflect the true likelihood of the corresponding events to occur in the studied population). Model calibration is an important issue in modern neural networks (Guo et al., 2017), and to our knowledge we are the first to study it for discourse relation classification. Properly calibrating a model (i.e. properly quantifying uncertainty) is especially important for this task, because determining the correct discourse sense involves a large degree of uncertainty, and more correctly quantifying uncertainty allows a model to be more explainable.

Our two main research questions, as described above, are illustrated in Figure 1, and our contributions can be summarized as the following:

1. Determine the effects of added context on the discourse annotation task by increasing the context window given to the annotators and comparing the results to those of presenting annotators with only the two arguments across three different datasets. Measure the annotation accuracy and confidence under each of these conditions.

2. Perform a qualitative error analysis of these results, providing insight into cases in which adding context may improve annotation results.

3. Add annotation accuracy and confidence scores to the input of an implicit sense classifier, and measure the resulting changes in model accuracy.

4. Use confidence scores to impact the training and evaluation mechanisms of an implicit sense classifier, and use accuracy and calibration metrics as validation metrics for these models. Measure the change in accuracy and model calibration.

5. Perform a qualitative error analysis on the model results, finding cases where model performance suffers without access to context and providing explanations as to why.

Our code can be found here [1].

2 Related Work

The effects of context on implicit sense classification As mentioned above, implicit sense classification is a very challenging task. Further, most implicit sense classifiers (Chen et al., 2019) do not include context outside of the two arguments contained in a discourse relation,[2]despite the annotators having access to context during the PDTB annotation task, wherein the annotator inserts a connective between the two arguments and then determines the discourse relation. An example of this connective insertion from the PDTB is as follows, with Arg1 in **bold** and Arg2 in *italics*:

(3) Several leveraged funds don't want to cut the amount they borrow because it would slash the income they pay shareholders, fund officials said. But a few funds have taken other defensive steps. **Some have raised their cash positions to record levels.** Implicit = BECAUSE *High cash positions help buffer a fund when the market falls.*

However, no paper has yet studied the effect of context on the discourse annotation task, nor has a work attempted to use insights from annotators' proficiency and confidence on the model.

[1] https://github.com/katherine-atwell/DiscourseContextUncertainty

[2] Note that in the case of PDTB-3, it is possible for local (sentence-level) context to be encoded using pre-trained encoders such as BERT, when determining intra-sentential implicit discourse relations.

The closest work to ours in this space is Scholman and Demberg (2017), who use a connective insertion task to crowdsource discourse sense annotations and examine the effect of context on this task. They find that under certain conditions (such as when argument 1 refers to an entity/event in the surrounding context or the sentence after argument 2 expands on argument 2), the presence of context can improve annotator agreement. However, this annotation task is simplified and only covers 6 level-2 relations. Thus, their result is not fully representative of the PDTB annotation task. We wish to examine this question more in-depth by presenting annotations using the traditional PDTB annotation task to a trained linguist in settings with and without additional context, and comparing these annotations against ground truth data. In doing so, we believe we can gain more insight into factors that affect human understanding in more similar conditions to the ones present in the original PDTB annotation task.

Calibration of neural networks for discourse parsing *Calibration* in machine learning refers to the distribution of error and the model's level of self-assessment, or confidence (Bella et al., 2010). It was found that neural networks tend to be badly calibrated (Guo et al., 2017), which can result in poor explainability and uncertainty quantification. However, pretrained models, even very complex ones, were found to generally be more well-calibrated, and to benefit from temperature scaling (Desai and Durrett, 2020). Therefore, we use annotator confidence scores to scale the temperature according to a the example's simplicity (as perceived by the annotator). In our work, we choose to use the Brier score (Brier et al., 1950) to measure calibration because it is a proper scoring function. As far as we are aware, we are the first paper to study the calibration of implicit sense classification models and the impact of using annotator confidence measures to improve model calibration.

3 Data and Analysis

3.1 Methods

Here we describe our human annotation experiments, in which we determine whether adding context can improve annotator accuracy and confidence.

Dataset For all of our experiments, we use gold data from the Penn Discourse Treebank 2 (PDTB-2, Prasad et al. (2008)), Penn Discourse Treebank 3 (PDTB-3, Webber et al. (2019)), and the English set of the TED Multilingual Discourse Bank (TED-MDB, Zeyrek et al. (2018)), in order to test our hypothesis across different frameworks and text domains. The PDTB-2 is the most commonly used dataset for discourse parsers, while the PDTB-3 introduces intra-sentential discourse relations and an updated label schema and the TED dataset contains speeches annotated with the PDTB-3 framework.

Discourse sense	Without context	With context	Context effect
Temp.Asynchronous	-2	-1	**better**
Cont.Cause	-3	-5	*worse*
Comp.Contrast	14	10	**better**
Comp.Concession	6	5	**better**
Comp.Similarity	2	1	**better**
Exp.Conjunction	11	5	**better**
Exp.Instantiation	9	10	*worse*
Exp.Equivalence	3	4	*worse*
Exp.Level-of-detail	-16	-9	**better**

Table 1: Frequency of discourse senses with/without context in our annotated set relative to ground truth (0 indicates a perfect overlap with ground truth count). We can see that a context window usually entails a better (more ground truth-like) distribution. Moreover, the impact of the context effect is usually stronger when it is better than when it is worse. The full table including neutral relations and raw counts is in the appendix.

Annotation To test our hypothesis that adding context improves annotation performance for the implicit sense labeling task, we recruit two expert linguists to provide level 2 sense annotations for implicit discourse relations from all three corpora listed above, calculating 60% absolute agreement. This task was approved by our institution's human subjects board.

In order to attain a representative sample, we make sure that every type of implicit discourse sense contained in the PDTB-2, PDTB-3, and TED-MDB was represented at least once in this sample. Further, in order to select for relations that may need more context than the two arguments, we randomly sample a large set of relations and, from that sample, select relations whose arguments have a high portion of pronouns and a low level of specificity. Pronouns signal coreference relations that may be missing from the argument spans (Scholman and Demberg, 2017), and a low level of specificity suggests that more information may be needed to understand the full context (Li et al., 2016; Choi et al., 2021) . We use NLTK's part of

speech tagger to detect the number of pronouns contained in the arguments, and to determine specificity we use the Ko et al. (2019) specificity classifier.

Task To study the role of context, our annotators perform two respective tasks. First, they attempt to determine the discourse connective, and corresponding discourse sense, when only shown the pair of arguments. For the second task, the annotator has access to the full sentence(s) containing the arguments, as well as the two sentences before and the sentence after the arguments. The first task always comes before the second task, and the annotator is not able to edit their annotation for the first task after seeing the additional context in the second task.

We produce 498 samples of human annotations for discourse sense and confidence before and after context. Of the samples, 147 come from the PDTB-2, 199 from the PDTB-3, and the last 152 from the TED-MDB.

3.2 Analysis

To better understand the hypothetical effect of context, we investigate annotation changes with respect to the discourse relations in the arguments, as well as in the context window supplied to the annotator.

Does context improve annotation accuracy? Generally, yes. Over all three corpora, annotator accuracy increases from 0.350 to 0.414 (+6.4%) between the task without context and the task with context. Table 2 provides the breakdown of accuracy by corpora for each task. While the task accuracy is low in general due to the difficulty of discourse relation prediction, we can see consistently higher accuracy across all three corpora in the task with context. However, it should be noted that these increases are not statistically significant, likely due to the small size of our annotated set.

Corpus	Accuracy		Confidence	
	Raw	Context	Raw	Context
PDTB2	0.306	**0.354**	3.70	**4.81**
PDTB3	0.379	**0.423**	3.56	**4.71**
TED-MDB	0.296	**0.355**	3.63	**4.84**

Table 2: Average annotator accuracy and confidence on a scale of 1-5 for each corpus before and after context. Adding context improves accuracy and confidence across all corpora, an increase which is statistically significant with respect to confidence ($p < 0.01$).

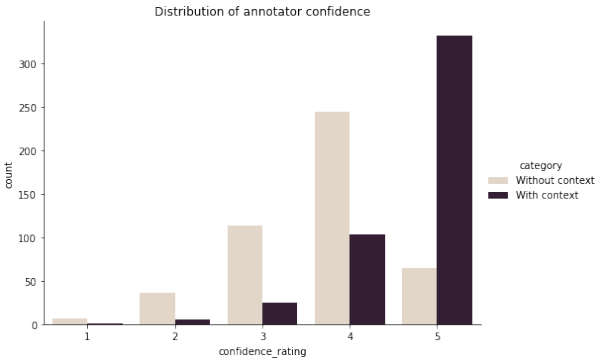

Figure 2: Distribution of the confidence scores (discrete scores from 1-5) chosen by the annotators. As this distribution shows, annotator confidence consistently increases with context.

Further, as Table 1 shows, the discourse sense distribution of annotations models the ground truth distribution more closely after context.

Does context improve annotator confidence? Annotator confidence provides information beyond simply whether an annotation is incorrect/correct. We found that over all three corpora, annotator confidence ratings improved from 3.59 to 4.77 (+1.18) after being given context. Similarly to annotator accuracy, we see all three corpora reflect this improvement in Table 2. The effect of context on annotator confidence is statistically significant across all three corpora ($p < 0.01$).

In addition, the full results can be found in Table 3, which shows the distribution of annotations that are positively or negatively impacted by context across each discourse relation. Discourse relations with no samples that change after context are excluded.

But why study annotator confidence? Annotator confidence provides information beyond simply whether an annotation is incorrect/correct. For instance, an annotator could be correct but lack confidence due to a lucky guess. Further, in our annotations, we found that highly confident answers were twice as likely to be correct (0.390) as answers rated with a lower confidence (0.200). We incorporate these annotator confidence scores into our model and experimentally validate whether incorporating these scores helps guide model decisions in a meaningful way.

Is the presence of certain discourse relations in an expanded context window beneficial? Yes. We find both instances where context helped

(an incorrect annotation before context became correct after the annotator was given context) and where context hurt (a correct annotation before context became incorrect after the annotator was given context). In particular, we find that *Comparison.Contrast*, *Contingency.Cause*, *Expansion.Conjunction*, and *Expansion.Restatement* are likely to improve annotation accuracy. Contrarily, *Expansion.List* and *Temporal.Asynchronous* seem to provide less helpful context.

The full results are reported in Table 4. We exclude relations in the context that had no effect on the annotation's accuracy, or when the annotation remained correct or incorrect before and after context.

In what instances does context actually help? Our annotator performed a qualitative analysis on the annotations where context helps (an incorrect prediction turned correct) and hurts (a correct prediction turned incorrect). From this, we found that context typically helps in cases where the arguments are very short, the discourse structure in the surrounding sentences is made clearer, and background information in the surrounding texts illustrates a relationship between pieces of the two arguments that could not be extrapolated from only the argument pair. The example in Section 1 represents the latter phenomenon, and we provide an example of the first phenomenon in Appendix A. Below we provide an example where the discourse structure makes the correct relation more apparent, with argument 1 in **bold** and argument 2 in *italics*:

(4) USA Today reported that the Rales brothers, Washington, D.C.-based investors who made an unsuccessful offer to acquire Interco last year, have bought nearly 3% of Mead's common shares. Entertainment and media stocks generally escaped the market's slide as well. Paramount Communications rose 5/8 to 58 3/4, **Time Warner climbed 1 7/8 to 138 5/8**, *Walt Disney advanced 3 1/8 to 127 1/2*, MCA rose 1 1/8 to 65 5/8 and McGraw-Hill added 1/2 to 67 1/8. The American Stock Exchange Market Value Index lost 3.11 to 379.46.

Without the added context, the relation appears as though it could be contrastive to the annotator (where Time Warner's rise is compared to Disney's rise). However, additional context allows the annotator to see the discourse structure of the surrounding clauses in the sentence (also holding a *Conjunction* relation), as well as the sentence contain the arguments with the previous sentence (an *Instantation* relation where the sentence containing the arguments provides several similar examples to back up the first argument's claim). Thus, in this example, the discourse structure of the surrounding text is beneficial for labeling the discourse relation.

Table 1 further sheds light on ways in which context influences the chosen relations, showing that some discourse relations are predicted more than others with and without context (relative to their ground truth counts). For instance, *Expansion.Conjunction* is predicted at a much higher rate without context than with context, likely due to instances (such as the example in Section 1 where the annotator does not have enough information about the relationship between the two arguments to pick a relation with more rigidly defined semantics. We

Discourse sense	Incorr.→Corr.	Corr.→Incorr.
Temp.Asynchronous	**5 (15.63%)**	1 (3.13%)
Cont.Cause	7 (6.25%)	6 (5.36%)
Cont.Purpose	1 (5.00%)	0
Comp.Contrast	**4 (13.79%)**	1 (3.45%)
Comp.Concession	3 (14.29%)	2 (9.52%)
Comp.Similarity	1 (33.33%)	0
Exp.Conjunction	**12 (12.50%)**	5 (5.21%)
Exp.Instantiation	4 (10.00%)	3 (7.50%)
Exp.Equivalence	0	**2 (25.00%)**
Exp.Level-of-detail	**12 (17.91%)**	4 (5.97%)

Table 3: Distribution of ground truth Level 2 senses by whether an annotation turns from incorrect to correct after context or vice versa. Bolded cells in each row indicate whether a sense seems to benefit or hurt from context.

Discourse Sense	Incorr.→Corr.	Corr.→Incorr.
Temp.Asynchronous	4 (3.36%)	**9 (7.56%)**
Temp.Synchronous	2 (3.45%)	3 (5.17%)
Cont.Cause	**17 (6.37%)**	11 (4.12%)
Cont.Condition	3 (4.41%)	4 (5.88%)
Comp.Concession	2 (4.17%)	2 (4.17%)
Comp.Contrast	**26 (10.70%)**	17 (7.00%)
Exp.Alternative	0	1 (4.35%)
Exp.Conjunction	**30 (8.55%)**	14 (3.99%)
Exp.Exception	0	1 (33.33%)
Exp.Instantiation	6 (7.79%)	5 (6.49%)
Exp.List	2 (3.70%)	**4 (7.41%)**
Exp.Restatement	**12 (7.79%)**	2 (1.30%)

Table 4: Distribution of Level 2 senses in sample context by whether an annotation turns from Incorrect to Correct after context or vice versa. Bolded cells in each row indicate whether a sense seems to provide beneficial or harmful context.

find that *Comparison.Contrast* tends to also be chosen less often with context (the second example in this section illustrates a scenario in which the annotator changes from *Contrast* to another relation given context). *Expansion.Level-of-detail*, on the other hand, is chosen more often when more context is provided, which makes sense given that a *Level-of-detail* relation requires knowing that the semantics of argument 2 restate the semantics of argument 1, and that both arguments hold true at the same time. This information is not always available when only the two arguments are shown.

4 Modeling Insights

Above, we illustrate some insights attained from the annotated data with respect to changes in domain and access to context. In this section, we use both the annotator correctness and confidence metrics to inform our model decisions, in order to determine whether annotator performance in any way correlates with model performance. In addition to model accuracy, we evaluate model calibration scores when the model is and is not given access to annotator confidence. To influence the model's decisions when given access to annotator confidence, we adjust the training and validation mechanisms accordingly. We hypothesize that access to annotation metrics will improve model accuracy, and that changing the temperature function with respect to annotator confidence will improve model calibration. We describe the setup for each of our experiments below, and report our results in Section 4.2.

4.1 Experimental Setup

We use the Kim et al. (2020) XLNet-large baselines as our base model, for which the large XLNet (Yang et al., 2019) model is trained for a maximum of 10 epochs, but early stopping occurs if there is no improvement to the development set for 5 evaluation steps. We use sections 4-24 for training, 2-3 for development, and 0-1 for testing. Following Kim et al. (2020), we use the standard L2 classification with 12 labels for the PDTB-2, and use the 14 senses with more than 100 labels for the PDTB-3. For each result, we report the average across 3 different seeds. We first experiment with concatenating the features described below to the sentence embeddings produced by the XLNet-large model and passing the resulting embedding to a classification head to predict the discourse relation.

Exploiting annotation accuracy and confidence We first experiment with using features from our annotated data, in order to determine whether they provide any benefit to the model. The first feature we experiment with is a binary prediction (using the annotations as training data) as to whether or not the relation will be labeled correctly. To obtain this feature, we trained an SVM using bag-of-words features on our annotation data, where the label is *true* if the annotator labeled the relation correctly given only the argument pairs and *false* otherwise. We pass the argument pairs as input to the model. We used an 80/20 train/test split for this model and the classification accuracy is .838.

The second feature that we use is the confidence score for the two arguments given additional context, to determine whether using features that incorporate some contextual features help the model at all. As with the previous feature, we train an SVM using bag-of-words features on our annotation data, and again pass the two arguments as input to the model, but here we predict confidence as a regression task as opposed to a classification task. For this model, we also use an 80/20 train/test split, and report a mean square error of .180.

Reweighting using confidence annotations Beyond experimenting with adding annotation metrics as features to our model, we experiment with adjusting the training and validation mechanisms of our model using predicted annotator confidence scores. We use these scores to adjust the training weights, weighting the examples with lower predicted confidence higher and the examples with higher predicted confidence lower. For each example, we predict its corresponding confidence feature and divide 5 by this value (as 5 is the highest the confidence level can go). We then use this value as the weight for the sample, thus upsampling all examples with a predicted confidence score less than 5 out of 5.

Temperature adjustment using confidence annotations Similarly, we experiment with adjusting the temperature of the softmax function, in order to increase model confidence in proportion to predicted annotator confidence. We thus weight the examples with higher predicted confidence scores higher, and vice versa, by dividing 5 by the confidence score for each example to get our temperature (which is inversely proportional to the desired model confidence). We show this in the following

Model	PDTB-2	PDTB-3	TED
XLNet-large (cased)	.5527	.6326	**.5381**
+Correctness	**.5694***	.6452*	.5347
+Confidence	.5648*	**.6518***	.5035*
+Correctness & Confidence	.5642*	.6428*	.4931*
+Reweighting	.5665*	.6419*	.4861*

Table 5: Accuracy scores for each model evaluated on the PDTB-2 and PDTB-3, with the best performing model in bold for each metric (+Correctness for the PDTB-2, +Confidence for the PDTB-3, and the baseline for TED). * indicates statistical significance ($p < 0.05$).

Model	PDTB-2	PDTB-3	TED
XLNet-large	.5527	.6326	.5381
+ temp change	.5597	.6326	.5486

Table 6: Accuracy for baseline and model with adjusted softmax temperature. Changing the temperature appears to slightly improve performance for the PDTB-2 while not affecting performance for the PDTB-3.

equation, denoting the temperature as $\mathcal{T}$ and the confidence score as c: $\mathcal{T} = \frac{5}{c}$

4.2 Results

In order to determine the effects of adding annotator performance metrics as input to the model, we detail the results from each of the models above, in particular looking at accuracy for the models with concatenated annotation features and accuracy and calibration for the models that use annotator confidence features to influence their training mechanism. We provide several questions we wish to answer with these analyses, and the corresponding results, below. We test for significance using a two-tailed t-test for each experiment.

Is reweighting training examples using confidence scores useful? Similarly to directly adding confidence features as input to the model, reweighting the training examples based on the predicted confidence score improves upon the baseline for both corpora (Table 5). This suggests that influencing the training mechanism with confidence scores has the potential to improve model accuracy. We report level 2 results of this model in Tables 8 and 10 in the Appendix. As with the previous results, the scores on the TED dataset are not improved when these changes are made, but because of the small size of the test set, we do not believe this to be notable.

Does adjusting temperature improve accuracy, calibration, or both? To evaluate the results of the model in which temperature was adjusted, we measure both the accuracy and the *calibration* of the model (calculated using the Brier score). We find that although the model with adjusted temperature outperforms the baseline with respect to accuracy only on the PDTB-2 (Table 6), it outperforms the baseline with respect to the Brier score by a large margin for both datasets (Table 7). Thus, the model with the temperature change is more *well-calibrated* than the original model, i.e. its probabilities are more likely to reflect the actual probability of a prediction being correct given the input. A large improvement for the Brier scores is seen on the TED test set, similarly substantial to that of the other two datasets. However, although this is encouraging, we once again take caution in drawing significant conclusions from this due to the small size of our TED test set.

Model	PDTB-2	PDTB-3	TED
XLNet	.6781	.5787	.7214
XLNet + temp change	.6075*	.5295*	.6477*

Table 7: Brier scores for baseline and model with adjusted softmax temperature. For both datasets, the Brier score improves (a lower Brier score is better) when the temperature is adjusted in accordance with the predicted confidence. * indicates statistical significance ($p < 0.5$)

In addition to reporting overall calibration metrics, we visualize results on individual data points (excluding the TED dataset due to its small size). Using the XLNet-large model, we run the Data Maps tool (Swayamdipta et al., 2020) on the PDTB-2 (Figure 4) and PDTB-3 (Figure 6). We find that the PDTB-3 has more well-defined regions than the PDTB-2, with the model much more likely to get an example with high confidence and low variability correct than an example with low confidence and low variability. This suggests that the model trained on the PDTB-3 is more well-calibrated than the model trained on the PDTB-2, which is supported by the difference in the Brier scores between the two datasets (Table 7). The lack of easy-to-learn examples in the PDTB-2 also provides a possible explanation for the difficulty of the implicit sense classification task for this dataset; the results of Swayamdipta et al. (2020) indicate that easy-to-learn examples are important for optimization.

Does predicting annotation accuracy and confidence help? For both the PDTB-2 and PDTB-3, adding the features predicting annotator correct-

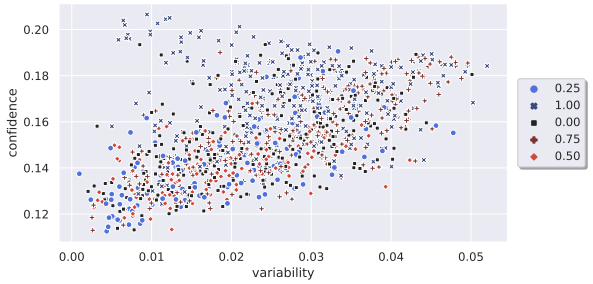

Figure 3: Data map for the baseline model trained on the PDTB-2. Here, there appear to be no distinct regions with respect to correctness, confidence, and variability.

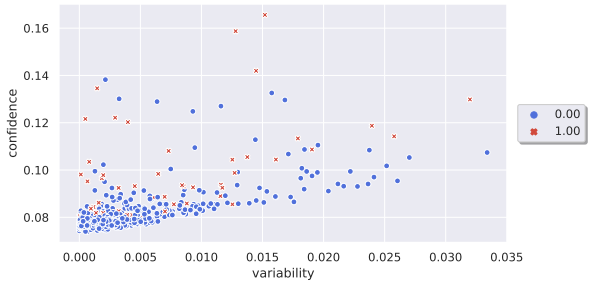

Figure 4: Data map for the PDTB-2 model with temperature changes; here, there appear to be distinct regions with respect to correctness, confidence, and variability, but similarly to the baseline PDTB-2 models, no easy-to-learn samples.

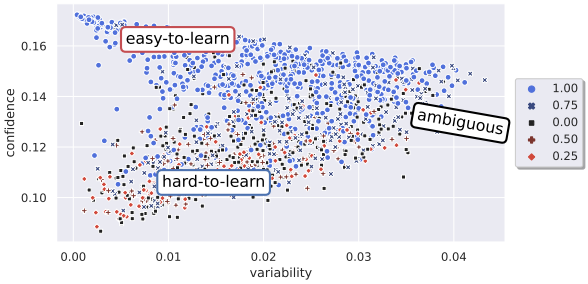

Figure 5: Data map for the baseline model trained on the PDTB-3. Here, there appear to be distinct regions with respect to ease of learning; easy to learn examples have higher confidence and lower variability, and more difficult to learn examples have lower confidence and lower variability.

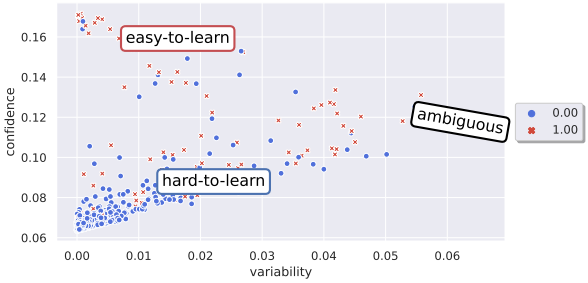

Figure 6: Data map for PDTB-3. Here, as with the PDTB-3 baseline model, there appear to be distinct regions with respect to ease of learning. However, there appears to be a clearer separation between easy-to-learn and hard-to-learn samples with respect to accuracy.

ness and confidence improved model results over the XLNet-large baseline (Table 5), with the added correctness feature yielding the best-performing model on the PDTB-2 and the confidence feature yielding the best-performing model on the PDTB-3. Therefore, it appears that correctness and ease of annotation have some impact on model correctness. We report level 2 results of this model in Tables 8 and 10 in the Appendix. We note that none of these features yielded an improvement when our model was tested on the TED set; however, due to the TED dataset's small set of implicit relations, we hesitate to draw conclusions from these numbers (our test set is comprised of 96 examples).

What kinds of errors do our models make? In addition to annotating discourse relations given argument pairs and additional context, our annotator performed a qualitative analysis on the results of the model. From this analysis, we found that the most common hypothesized reasons for the model guessing a relation wrong were to do with lack of context. There were also cases in which the model did not pick up on vocabulary that indicated a *Temporal* relation. For some misclassifications, the annotator could see why the predicted relation could hold, but believed that the gold annotation was the better one. There were a few cases where the annotator agreed with the model predictions as opposed to the gold labels. We provide some examples of these phenomena in Appendix D. Overall, the annotator observed more cases where errors occur because additional context is needed in the PDTB-2 and TED-MDB than in the PDTB-3.

5 Conclusion

In the previous sections, we first show the effects of presenting additional context when compared to providing only the two arguments. We find that adding context has an overall positive impact on annotator accuracy and confidence. More specifically, we find that context helps for certain discourse relations, but not all relations. We hypothesize that in cases where adding context worsens annotation, it is because context may add confusing information about argument relationship (such as whether one is a quote) or the completeness of a list of items in a

particular set (blurring the line between senses such as *Instantiation* and *Level-of-detail*). We believe this is worth exploring further.

Secondly, we find that utilizing the human performance metrics we collected in the first half of the paper yielded better model performance when compared to the baseline, suggesting that these metrics give the model some useful information about its own predictions. In particular, we find that using confidence scores to adjust the training weights improves model *accuracy*, while using them to adjust the softmax temperature improves model *calibration*, the latter of which is important for explainability and for tasks with a high degree of uncertainty (discourse relation classification being one such task). To our knowledge, ours is the first work to study calibration with respect to discourse models.

We hope that future work will continue to study the role of context in discourse relation classification, as well as model calibration for this task. We will release our annotations and model code upon the publication of this paper.

6 Ethical Considerations

Our experiments were approved by our institution's human subjects board. We acknowledge that the pretrained models we use in this paper may introduce bias.

Acknowledgements

We thank Matthew Stone, Bonnie Webber, Mert Inan, Anthony Sicilia, and the anonymous reviewers for their valuable feedback. This research was partially supported by NSF grants IIS-2145479, IIS-2107524.

References

Antonio Bella, Cèsar Ferri, José Hernández-Orallo, and María José Ramírez-Quintana. 2010. Calibration of machine learning models. In *Handbook of Research on Machine Learning Applications and Trends: Algorithms, Methods, and Techniques*, pages 128–146. IGI Global.

Glenn W Brier et al. 1950. Verification of forecasts expressed in terms of probability. *Monthly weather review*, 78(1):1–3.

Mingda Chen, Zewei Chu, and Kevin Gimpel. 2019. Evaluation benchmarks and learning criteria for discourse-aware sentence representations. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 649–662.

Eunsol Choi, Jennimaria Palomaki, Matthew Lamm, Tom Kwiatkowski, Dipanjan Das, and Michael Collins. 2021. Decontextualization: Making sentences stand-alone. *Transactions of the Association for Computational Linguistics*, 9:447–461.

Shrey Desai and Greg Durrett. 2020. Calibration of pre-trained transformers. *arXiv preprint arXiv:2003.07892*.

Michael Glanzberg. 2002. Context and discourse. *Mind & language*, 17(4):333–375.

Chuan Guo, Geoff Pleiss, Yu Sun, and Kilian Q. Weinberger. 2017. On calibration of modern neural networks. In *Proceedings of the 34th International Conference on Machine Learning*, volume 70 of *Proceedings of Machine Learning Research*, pages 1321–1330. PMLR.

Najoung Kim, Song Feng, Chulaka Gunasekara, and Luis Lastras. 2020. Implicit discourse relation classification: We need to talk about evaluation. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 5404–5414, Online. Association for Computational Linguistics.

Wei-Jen Ko, Greg Durrett, and Junyi Jessy Li. 2019. Domain agnostic real-valued specificity prediction. In *AAAI*.

Davis Lewis. 1980. Index, context, and content. In *Philosophy and grammar*, pages 79–100. Springer.

Junyi Jessy Li, Bridget O'Daniel, Yi Wu, Wenli Zhao, and Ani Nenkova. 2016. Improving the annotation of sentence specificity. In *Proceedings of the Tenth International Conference on Language Resources and Evaluation (LREC'16)*, pages 3921–3927, Portorož, Slovenia. European Language Resources Association (ELRA).

Rashmi Prasad, Nikhil Dinesh, Alan Lee, Eleni Miltsakaki, Livio Robaldo, Aravind K Joshi, and Bonnie L Webber. 2008. The penn discourse treebank 2.0. In *LREC*. Citeseer.

Merel Scholman and Vera Demberg. 2017. Crowdsourcing discourse interpretations: On the influence of context and the reliability of a connective insertion task. In *Proceedings of the 11th Linguistic Annotation Workshop*, pages 24–33, Valencia, Spain. Association for Computational Linguistics.

Swabha Swayamdipta, Roy Schwartz, Nicholas Lourie, Yizhong Wang, Hannaneh Hajishirzi, Noah A. Smith, and Yejin Choi. 2020. Dataset cartography: Mapping and diagnosing datasets with training dynamics.

Sharon L Thompson-Schill. 2003. Neuroimaging studies of semantic memory: inferring "how" from "where". *Neuropsychologia*, 41(3):280–292.

Bonnie Webber, Rashmi Prasad, Alan Lee, and Aravind Joshi. 2019. The penn discourse treebank 3.0 annotation manual. *Philadelphia, University of Pennsylvania*.

Zhilin Yang, Zihang Dai, Yiming Yang, Jaime Carbonell, Russ R Salakhutdinov, and Quoc V Le. 2019. Xlnet: Generalized autoregressive pretraining for language understanding. *Advances in neural information processing systems*, 32.

Deniz Zeyrek, Amalia Mendes, and Murathan Kurfali. 2018. Multilingual extension of pdtb-style annotation: The case of ted multilingual discourse bank. In *LREC*.

A Examples Where Context Helps

The first example represents a case where the arguments are very short and do not provide a strong clue about the relation:

(5) Currently, chips are produced by shining light through a mask to produce an image on the chip, much as a camera produces an image on film. But details on chips must now be extraordinarily fine, and the wavelengths of even ultraviolet light are long enough so that the images they draw may be too blurry – much as someone using a wide paintbrush could produce a broad line but would have trouble painting a thin one. X-rays, by contrast, **travel straighter** *and can be focused more tightly than light*. X-rays have problems, too.

Taken by themselves, **travel straighter** and *and can be focused more tightly than light* could be related in any number of ways, and our annotator guessed *Temporal.Asynchronous*. However, given the structure of the sentence the arguments are contained in as well as the information given in the previous sentences, the annotator changed to the correct relation (*Contingency.Cause*) upon seeing the additional context.

B Example Where Context is Misleading

(6) As the best opportunities for corporate restructurings are exhausted of course, at some point the market will start to reject them. But the airlines are scarcely a clear case, given anti-takeover mischief by Secretary of Transportation Skinner, who professes to believe safety will be compromised if KLM and British Airways own interests in companies that fly airplanes. **Worse, Congress has started to jump on the Skinner bandwagon.** *James Oberstar*, the Minnesota Democrat who chairs the Public Works and Transportation Committee's aviation subcommittee, *has put an anti-airline takeover bill on supersonic speed so that it would be passed in time to affect the American and United Air Lines bids*. It would give Mr. Skinner up to 50 days to "review" any bid for 15% or more of the voting stock of any U.S. carrier with revenues of $1 billion or more.

C Examples from qualitative analysis of model results

For all examples below, argument 1 is **bolded** and argument 2 is *italicized*.

C.1 Examples that need more context

PDTB-2 Below is the example without context, which the model predicted as *Contingency.Cause*:

(7) **the threat of U.S. retaliation, combined with a growing recognition that protecting intellectual property is in a country's own interest, prompted the improvements made by South Korea, Taiwan and Saudi Arabia**
What this tells us is that U.S. trade law is working

The ground truth label of this example is *Expansion.Restatement*. Without the additional context, upon inspecting the model output, the annotator concluded that "so/therefore", which signal causality, could be acceptable, but "in other words", signaling *Restatement*, could also work. Below is the example with context:

(8) They will remain on a lower-priority list that includes 17 other countries. Those countries – including Japan, Italy, Canada, Greece and Spain – are still of some concern to the U.S. but are deemed to pose less-serious problems for American patent and copyright owners than those on the "priority" list.
Gary Hoffman, a Washington lawyer specializing in intellectual-property cases, said **the threat of U.S. retaliation, combined with a growing recognition that protecting intellectual property is in a country's own interest, prompted the improvements made by South Korea, Taiwan and Saudi Arabia**. "*What this tells us is that U.S. trade law is working*," he said.

This context makes it clearer that the proper relation to annotate here is *Expansion.Restatement*.

PDTB-3 Below is the example without context, which the model predicted as *Temporal.Asynchronous*:

(9) **In 1976, for example, dividends on the stocks in Standard & Poor's 500-stock index soared 10%, following much**

slower growth the year before.
The S&P index started sliding in price in September 1976,

The ground truth label of this example is *Comparison.Contrast*. Without the additional context, upon inspecting the model output, the annotator concluded that more context was needed, but without the additional context they could understand how either a temporal or contrastive relation could be held. Below is the example with context:

(10) Indeed, analysts say that payouts have sometimes risen most sharply when prices were already on their way down from cyclical peaks. **In 1976, for example, dividends on the stocks in Standard & Poor's 500-stock index soared 10%, following much slower growth the year before.** *The S&P index started sliding in price in September 1976,* and fell 12% in 1977 – despite a 15% expansion in dividends that year.

This context makes it more clear why *Comparison.Contrast was chosen*. However, a *Temporal.Synchronous* relation also holds between the two arguments even with the surrounding context. Thus, though the model predicted this relation incorrectly, it predicted a relation that was close to another relation that holds between the two but was not annotated in the gold label set.

TED-MDB Below is the example without context, which the model predicted as *Expansion.Level-of-detail*:

(11) **I want to show you a new kind of map.**
This is not a geographic map.

The ground truth label of this example is *Comparison.Concession*. Without the additional context, the annotator understood how *Level-of-detail* could be inferred, as the speaker seems to be elaborating on the type of map. However, the example below, with added context, clarifies this:

(12) When we think about mapping cities, we tend to think about roads and streets and buildings, and the settlement narrative that led to their creation, or you might think about the bold vision of an urban designer, but there's other ways to think about mapping cities and how they got to be made. Today, **I want to show you a new kind of map.** *This is not a geographic map.* This is a map of the relationships between people in my hometown of Baltimore, Maryland, and what you can see here is that each dot represents a person, each line represents a relationship between those people, and each color represents a community within the network.

With the addition of the sentence before argument 1, it is a lot more clear why the correct label is *Comparison.Concession* and not *Expansion.Level-of-detail*.

D Examples from qualitative analysis of model results

For all examples below, argument 1 is **bolded** and argument 2 is *italicized*.

D.1 Examples where annotator disagrees with ground truth

PDTB-2 Below is the example without context, which the model predicted as *Contingency.Cause*:

(13) **Pro-forma balance sheets clearly show why Cray Research favored the spinoff.**
Without the Cray-3 research and development expenses, the company would have been able to report a profit of $19.3 million for the first half of 1989 rather than the $5.9 million it posted.

The ground truth label of this example is *Expansion.Restatement*. When inspecting the model output without context, our annotator questioned the reason for this, as they believed there was a stronger causal relation. Below is the example with context:

(14) Analysts calculate Cray Computer's initial book value at about $4.75 a share. Along with the note, Cray Research is transferring about $53 million in assets, primarily those related to the Cray-3 development, which has been a drain on Cray Research's earnings.
Pro-forma balance sheets clearly show why Cray Research favored the spinoff.
Without the Cray-3 research and development expenses, the company would have been able to report a profit of $19.3 million for the first half of 1989 rather than the $5.9 million it posted.

Here, context does not have much of an impact on the meaning of the relation. Thus, the opinion remained that *Contingency.Cause* is more correct than *Expansion.Restatement*, and thus the model did not commit an error here.

PDTB-3 This example illustrates the common ambiguity between *Expansion.Instantiation* and *Expansion.Restatement*, and represents a case in which our annotator disagreed with the ground truth label. Below is the example without context, which the model predicted as *Expansion.Instantiation*:

(15) **The competition has cultivated a much savvier consumer.**
The average household will spread 19 accounts over a dozen financial institutions,

The ground truth label of this example is *Expansion.Instantiation*. Below is the example with context:

(16) "Today, a banker is worrying about local, regional and money-center banks, as well as thrifts and credit unions," says Ms. Moore at Synergistics Research. "So people who weren't even thinking about targeting 10 years ago are scrambling to define their customer base." **The competition has cultivated a much savvier consumer.** "*The average household will spread 19 accounts over a dozen financial institutions,*" says Michael P. Sullivan, who runs his own bank consulting firm in Charlotte, N.C. "This much fragmentation makes attracting and keeping today's rate-sensitive customers costly."

Though this context sheds light on the fact that the focus of this passage is on customers' behavior with respect to banking, it is unclear whether argument 2 represents the only way in which the customer has become more savvy as a result of the competition. Thus, it is still ambiguous as to which relation holds here, and the model's decision to predict *Expansion.Instantiation* is close to if not the correct choice.

TED-MDB This example represents a case in which the ground truth connective appears to make the most sense, but our annotator did not agree with the ground truth sense label. Below is the example without context, which the model predicted as *Expansion.Conjunction*:

(17) **the balance of power to really influence sustainability rests with institutional investors, the large investors like pension funds, foundations and endowments.**
I believe that sustainable investing is less complicated than you think, better-performing than you believe, and more important than we can imagine.

The ground truth label of this example is *Expansion.Level-of-detail*. Upon seeing the model output, the annotator concluded that they would have also likely chosen *Conjunction* over *Level-of-detail*. The additional context, as seen below, does not appear to contradict this assessment:

(18) And by sustainability, I mean the really juicy things, like environmental and social issues and corporate governance. I think it's reckless to ignore these things, because doing so can jeopardize future long-term returns. And here's something that may surprise you: **the balance of power to really influence sustainability rests with institutional investors, the large investors like pension funds, foundations and endowments.** *I believe that sustainable investing is less complicated than you think, better-performing than you believe, and more important than we can imagine.*

Because the added context does make the two statements seem any more parallel, *Expansion.Conjunction* appears to be the best choice, despite the fact that *in fact* makes the most sense as a connective. Indeed, given that *Expansion.Conjunction* is the second-most-common annotation for *in fact* per the PDTB 3.0 Annotation Manual, the connective *in fact* being correct here does not preclude the possibility of *Expansion.Conjunction* being the correct label, but may have influenced the annotators of the TED dataset in the direction of *Expansion.Restatement.*

E Level 2 Recall and Annotation Distributions

Discourse Sense	XLNet	+ correctness	+ confidence	+ corr. & confidence	+ temp change	+ reweighting
Temp.Asynchronous	0.4333	0.44	0.3933	0.3867	0.4067	0.4867
Temp.Synchrony	0.1795	0.2051	0.1795	0.1795	0.1538	0.1795
Cont.Cause	0.6749	0.6737	0.6725	0.6655	0.6432	0.6573
Cont.Pragmatic cause	0	0	0	0	0	0
Comp.Contrast	0.5029	0.5478	0.5789	0.5731	0.5497	0.5439
Comp.Concession	0.0444	0.0444	0.0889	0	0	0.0444
Exp.Conjunction	0.5505	0.5745	0.5328	0.5694	0.5265	0.5366
Exp.Instantiation	0.5741	0.5802	0.6080	0.5957	0.5833	0.6420
Exp.Restatement	0.4772	0.4871	0.4859	0.4797	0.5412	0.4686
Exp.Alternative	0.2333	0.4	0.3333	0.2333	0.2	0.2667
Exp.List	0.2667	0.2667	0.3333	0.1667	0.2333	0.2667

Table 8: Recall on Level 2 senses for the PDTB-2, excluding labels that did not appear in the test set (note that the temp change model uses the Brier score as a validation metric)

Discourse Sense	XLNet	+ correctness	+ confidence	+ corr. & confidence	+ temp change	+ reweighting
Temp.Asynchronous	0.5841	0.5810	0.5810	0.5683	0.5950	0.5841
Temp.Synchronous	0.2424	0.2525	0.3030	0.2727	0.2338	0.2626
Cont.Cause	0.7506	0.7409	0.7513	0.7350	0.7439	0.7587
Cont.Cause+Belief	0	0	0.0256	0	0	0
Cont.Condition	0.7407	0.9074	0.9259	0.7222	0.7407	0.8333
Cont.Purpose	0.9271	0.9236	0.9306	0.9444	0.9256	0.9097
Comp.Contrast	0.4505	0.4835	0.4505	0.4689	0.4584	0.4762
Comp.Concession	0.6254	0.6222	0.6127	0.6894	0.6227	0.5683
Exp.Conjunction	0.6176	0.6656	0.6399	0.6522	0.6262	0.6577
Exp.Instantiation	0.6751	0.6554	0.6582	0.6638	0.6715	0.6102
Exp.Equivalence	0.1333	0.2533	0.0933	0.2533	0.1276	0.1200
Exp.Level-of-detail	0.4635	0.4793	0.4927	0.4562	0.4630	0.4818
Exp.Manner	0.1905	0.2381	0.2381	0.2143	0.1905	0.2738
Exp.Substitution	0.5938	0.5625	0.5938	0.6875	0.5938	0.6979

Table 9: Recall on Level 2 senses for the PDTB-3 (note that the temp change model uses the Brier score as a validation metric)

Discourse Sense	XLNet	+ correctness	+ confidence	+ corr. & confidence	+ temp change	+ reweighting
Temp.Asynchronous	0.5	0.5	0.5417	0.5417	0.5417	0.3333
Cont.Cause	0.7857	0.8571	0.8095	0.7143	0.8571	0.7857
Cont.Cause+Belief	0.1111	0	0.1111	0	0	0
Cont.Purpose	0.9333	1	0.9333	0.8667	1	1
Comp.Contrast	0.2222	0.3333	0.2222	0.3333	0.3333	0.2222
Comp.Concession	0.3333	0.3333	0.2222	0.2222	0.2222	0.2222
Exp.Conjunction	0.5	0.4487	0.4744	0.4359	0.4872	0.4487
Exp.Instantiation	0.5333	0.6	0.5333	0.6	0.5333	0.4667
Exp.Equivalence	0.4	0.4	0.2667	0.3333	0.3333	0.2
Exp.Level-of-detail	0.4697	0.4394	0.3636	0.4091	0.4848	0.4545
Exp.Substitution	0.6667	0.6667	0.6667	0.6667	0.6667	0.5556

Table 10: Recall on Level 2 senses for the TED-MDB, excluding labels that did not appear in the test set (note that the temp change model uses the Brier score as a validation metric)

Discourse sense	Ground Truth	Without context	With context	Context effect
Temp.Asynchronous	6	4	5	**better**
Temp.Synchronous	0	1	1	neutral
Cont.Cause	14	11	9	*worse*
Cont.Cause+Belief	2	3	3	neutral
Cont.Cause+SpeechAct	2	1	1	neutral
Cont.Purpose	4	0	0	neutral
Comp.Contrast	3	17	13	**better**
Comp.Concession	8	2	3	**better**
Comp.Concession+SpeechAct	1	0	0	neutral
Comp.Similarity	2	0	1	**better**
Exp.Conjunction	20	31	25	**better**
Exp.Instantiation	5	14	15	*worse*
Exp.Equivalence	4	7	8	*worse*
Exp.Exception	1	0	0	neutral
Exp.Level-of-detail	30	14	21	**better**
Exp.Manner	0	1	1	neutral
Exp.Substitution	4	0	0	neutral

Table 11: Frequency of discourse senses in our annotated set with respect to ground truth, annotations without context, and annotations with context. We can see that at a label distribution level, a context window usually adds a better or neutral effect.

Improving Commonsense Contingent Reasoning by Pseudo-data and its Application to the Related Tasks

Kazumasa Omura **Sadao Kurohashi**
Graduate School of Informatics, Kyoto University
Yoshida-honmachi, Sakyo-ku, Kyoto, 606-8501, Japan
{omura, kuro}@nlp.ist.i.kyoto-u.ac.jp

Abstract

Contingent reasoning is one of the essential abilities in natural language understanding, and many language resources annotated with contingent relations have been constructed. However, despite the recent advances in deep learning, the task of contingent reasoning is still difficult for computers. In this study, we focus on the reasoning of contingent relation between basic events. Based on the existing data construction method, we automatically generate large-scale pseudo-problems and incorporate the generated data into training. We also investigate the generality of contingent knowledge through quantitative evaluation by performing transfer learning on the related tasks: discourse relation analysis, the Japanese Winograd Schema Challenge, and the JCommonsenseQA. The experimental results show the effectiveness of utilizing pseudo-problems for both the commonsense contingent reasoning task and the related tasks, which suggests the importance of contingent reasoning.

1 Introduction

Contingency is the relation between two events, one being an action or state and the other being likely to happen after it. We humans reason contingent relation between events on a daily basis. For instance, when we read text, we unconsciously infer what happens next to deepen our understanding. In conversations, we guess the next topic from the utterance of the opponent to make a contextual and natural response. Thus, the ability to reason contingent relation between events is essential when it comes to natural language understanding (NLU).

Recently, many studies have built language resources for contingent reasoning (Roemmele et al., 2011; Mostafazadeh et al., 2016; Zellers et al., 2018; Sap et al., 2019a; Hwang et al., 2021). These resources focus on basic events and evaluate some kind of commonsense reasoning ability. Although the fundamental linguistic capabilities of computers, such as question answering, have greatly improved with progress in deep learning, several studies have empirically demonstrated they still have difficulty in commonsense reasoning (Talmor et al., 2019; Sap et al., 2019b; Talmor et al., 2021).

I'm hungry, so
a. I'm gonna be absent from school.
b. I refrain from strenuous exercise.
c. I have a meal at a family restaurant.
d. I leave home.

Figure 1: Example from KUCI (English translated version). KUCI is a Japanese QA dataset containing 104k multiple-choice questions regarding contingent relation between basic events. The correct choice is bolded.

In this study, we aim at two objectives: to improve commonsense contingent reasoning and to investigate the effects of learning contingent knowledge on the related tasks to validate the importance of contingent reasoning. To these ends, we use the Kyoto University Commonsense Inference dataset (KUCI)[1]. KUCI is a Japanese QA dataset with 104k multiple-choice questions that ask contingent relation between basic events directly (Omura et al., 2020). An example is shown in Figure 1. This dataset is also characterized by its semi-automatic data construction method: automatic extraction of contingent pairs of basic event expressions from a web corpus, verification through crowdsourcing, and automatic generation of commonsense inference problems.

It is shown there is a performance gap between humans and computers on this task (Omura et al., 2020). Furthermore, through qualitative evaluation, it has been confirmed computers sometimes answer contingent relation between quite basic events incorrectly. One straightforward approach to alleviating the above problem is to extend the train-

[1] https://nlp.ist.i.kyoto-u.ac.jp/EN/?KUCI

Proceedings of the 29th International Conference on Computational Linguistics, pages 812–823
October 12–17, 2022.

ing data and increase the coverage. However, it is not practical from a cost perspective to increase the number of training examples manyfold using crowdsourcing.

We attempt to improve the performance by omitting crowdsourcing, a bottleneck in data augmentation, and utilizing pseudo-problems generated automatically from unverified contingent pairs of basic event expressions. As a web corpus is scalable, and all of the procedures except crowdsourcing are automatic, we can generate pseudo-problems at scale. It is expected pseudo-problems complement the lack of coverage though some problems are noisy and might be unanswerable.

The second objective of this study is to investigate the effects of learning contingent knowledge on the related tasks. On the premise that contingent reasoning is essential to NLU, we can expect contingent knowledge probably helps improve the performance on other NLU tasks. While the transferability of major English datasets has been studied (Phang et al., 2018; Sap et al., 2019b; Sakaguchi et al., 2020; Pruksachatkun et al., 2020), there is room to explore this dataset in terms of the task and language. We investigate the generality of contingent knowledge through quantitative evaluation of transfer learning on the related tasks.

In summary, we improve commonsense contingent reasoning by straightforward data augmentation. We generated 862k pseudo-problems, which is about ten times as large as the training examples in KUCI (83k), and incorporated them into training. Owing to pseudo-problems, a high-performance pre-trained model has achieved near human-level performance on the commonsense contingent reasoning task. We also investigate the transferability of contingent knowledge to the related tasks. Our experiments demonstrate intermediate-task training on KUCI with pseudo-problems positively affects discourse relation analysis, the Japanese Winograd Schema Challenge, and the JCommonsenseQA, which suggests the importance of contingent reasoning[2].

2 Approach

First, we describe our data augmentation approach to improving commonsense contingent reasoning. Our approach is to automatically generate large-scale pseudo-problems based on the construction method of the Kyoto University Commonsense Inference dataset (KUCI).

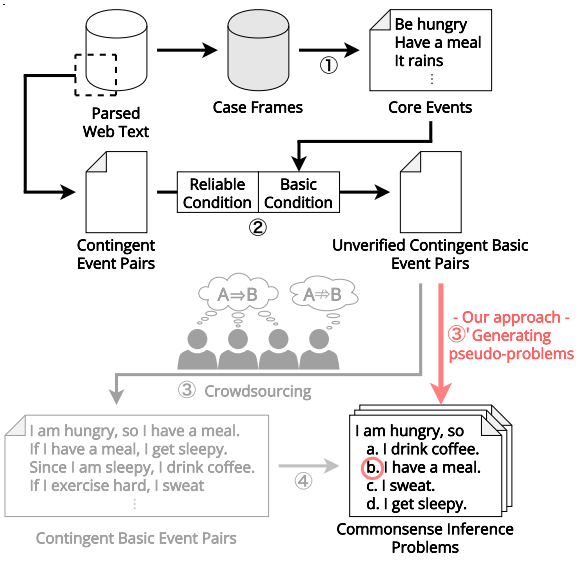

Figure 2: Overview of the method of generating commonsense inference problems in KUCI (gray) and pseudo-problems (red). The further details are described in Omura et al. (2020).

2.1 A Method of Generating Problems

The construction method of KUCI consists of the following four steps (Figure 2).

1. Acquire high-frequency predicate-argument structures (hereafter, **core events**[3]) from *case frames* (Kawahara et al., 2014b).

2. Extract event pairs that are unambiguously connected by explicit discourse markers representing contingent relation and composed of a pair of core events (hereafter, **contingent basic event pairs**).

3. Verify by crowdsourcing whether the extracted event pairs actually have contingent relation or not.

4. Generate problems by taking one of the verified event pairs (hereafter, **base**[3]) and selecting distractors from the latter events of other event pairs that are moderately similar to the base.

In the above procedures, it becomes possible to automatically generate pseudo-problems by omitting step 3 (Figure 2). For the parameters in the

[2]The links to the pseudo-data and code are available at https://nlp.ist.i.kyoto-u.ac.jp/EN/?KUCI

[3]We newly define these terms for clarification.

method, such as the thresholds of frequency for acquiring core events and the conditions on selecting distractors, we set them to the same values as in the construction of KUCI.

2.2 Automatic Extraction of Contingent Basic Event Pairs

We automatically extracted contingent basic event pairs following the method described in Section 2.1. We used a Japanese web corpus containing 3.3 billion sentences as the source text. It had been constructed by crawling web text from 2006 to 2015. There is no overlap of sentences between this corpus and the web corpus used in the construction of KUCI. As a result, we extracted 915k contingent basic event pairs. Omura et al. (2020) reported one-third of the extracted event pairs were removed by crowdsourcing, thus we expect about 600k event pairs to be valid.

2.3 Dealing with Data Leakage

There is a potential issue with generating training data from large-scale text, which is called "Data Contamination" (Brown et al., 2020). This issue is that text may include information about evaluation data, leading to overestimation of model performance.

We deal with this issue by heuristically excluding event pairs that are identical or remarkably similar to the *bases* in evaluation data[4]. Specifically, we apply the following filters based on word order and *core event* pairs.

Filter by word order Exclude an event pair if the length of the overlapping word order between the event pair and any base in evaluation data exceeds 75% of the word count of the base.

Filter by core event pairs Exclude an event pair if the event pair is composed of the core event pair that also composes any base in evaluation data.

For instance, the base of the problem in Figure 1 is "I'm hungry, so → I have a meal at a family restaurant" and composed of the core event pair "be hungry → have a meal at a family restaurant". Let us consider whether the event pair "I'm hungry, so → I have a big meal at the family restaurant" is excluded by the base or not. They have the overlapping word order, {I'm, hungry, so, I, have, a, meal, at, family, restaurant}, of which length (10) exceeds 75% of the word count of the base (11). It is also composed of the same core event pair. Thus, it will be excluded by both filters.

We expect the first filter to exclude syntactically-similar event pairs and the second to exclude those similar in content. As a result of filtering, we acquired 881k contingent basic event pairs.

2.4 Automatic Generation of Pseudo-problems

We went on performing an automatic generation of problems. As a result, we obtained 862k pseudo-problems from the 881k event pairs. The number of the pseudo-problems is about ten times as large as that of the training examples in KUCI (83k).

To analyze the quality of pseudo-problems, we randomly sampled 50 problems and manually evaluated them. As a result of manual evaluation, 36 of 50 problems were judged as answerable, which appears to be sufficient quality for pseudo-data.

3 Experiments

We conducted experiments to investigate the effects of incorporating pseudo-problems into training on the commonsense contingent reasoning task and the related tasks.

3.1 Model

We evaluated the performance of the BERT (Devlin et al., 2019) and XLM-RoBERTa (XLM-R) (Conneau et al., 2020) models.

BERT We employed the NICT BERT Japanese Pre-trained model (with BPE)[5]. It was pre-trained on the full text of Japanese Wikipedia for 1.1 million steps with a batch size of 4,096, partly referring to the pre-training configuration of RoBERTa (Liu et al., 2019). The model architecture is the same as the $\mathrm{BERT}_{\mathrm{BASE}}$.

XLM-R We adopted the XLM-$\mathrm{RoBERTa}_{\mathrm{LARGE}}$ model[6], which was pre-trained on a huge multilingual corpus consisting of Wikipedia and CC-100 (Wenzek et al., 2020). The model architecture is the same as the $\mathrm{BERT}_{\mathrm{LARGE}}$, but the embedding layer is relatively large due to its multilingual vocabulary. It is one of the high-performance pre-trained models for Japanese among those publicly available.

[4] To be specific, "evaluation data" refers to the development and test splits of KUCI.

[5] https://alaginrc.nict.go.jp/nict-bert/index.html (in Japanese)

[6] https://huggingface.co/xlm-roberta-large

3.2 Experimental Settings

The hyper-parameters used in the experiments are included in Appendix A.

3.2.1 Commonsense Contingent Reasoning

As is mentioned in Section 1, we used KUCI for assessing commonsense contingent reasoning ability. The task is to select the most appropriate sentence following the context from 4 choices like Figure 1. The dataset contains 83,127 / 10,228 / 10,291 examples for training, development, and test split, respectively.

During the fine-tuning phase, we minimize cross-entropy loss between the scores of each choice normalized by the softmax function and a one-hot vector representing the correct answer as 1. The scores of each choice are computed by inputting pairs of a context and the choice separated by special tokens and converting the hidden representations of the first token ([CLS]) into scalars by a linear transformation. When incorporating pseudo-problems into training, we define the objective function L as the weighted sum of cross-entropy losses of commonsense inference problems and pseudo-problems. The above can be expressed by the following equations.

$$H = -\frac{1}{N}\sum_{k=1}^{N}\log\frac{\exp(\mathbf{s}_{kj})}{\sum_{i=1}^{4}\exp(\mathbf{s}_{ki})}$$
$$L = H_{ci} + \lambda \times H_{pseudo}$$

where N is a batch size, j is the index of a correct choice among 1 to 4, s_{ki} is the score of the i-th choice of k-th example, H is the cross-entropy loss of commonsense inference problems or pseudo-problems, and λ is the weight for pseudo-problems.

During the inference phase, the choice with the highest score is selected as an answer. We evaluated the models by accuracy.

Comparative Method To investigate the effectiveness of a multiple-choice format, we also performed additional pre-training referring to Task-Adaptive Pre-Training (Gururangan et al., 2020). Specifically, we ran an additional Masked Language Modeling (MLM) task on the 881k event pairs used for generating pseudo-problems. For convenience, we name it "AMLM". After the additional pre-training, we fine-tuned the models on KUCI and the related tasks.

3.2.2 Intermediate-Task Transfer Learning

We performed transfer learning from the models fine-tuned on KUCI with pseudo-problems to investigate the effects of learning contingent knowledge. In this study, we employed discourse relation analysis, the Japanese Winograd Schema Challenge (JWSC) (Shibata et al., 2015), and the Japanese CommonsenseQA (JCQA) (Kurihara et al., 2022) as the related tasks.

Discourse Relation Analysis We used the Kyoto University Web Document Leads Corpus (KWDLC)[7] (Kawahara et al., 2014a; Kishimoto et al., 2018) for this task. KWDLC has been built by collecting the first three sentences of various kinds of web documents, and its size amounts to 6,445 documents. All the documents have been annotated with discourse relations between clauses using crowdsourcing. Moreover, 500 of 6,445 documents have also been annotated by linguistic experts. In this study, we used about 37k clause pairs with crowdsourced labels as training data and evaluated the classification performance on 2,320 clause pairs with expert labels.

The task is a seven-way classification of discourse relations between clauses, including "No Relation". We fine-tuned the models following the sentence pair classification framework proposed by Devlin et al. (2019) and ran five-fold cross validation. We used micro-averaged precision, recall, and F1 score computed without examples with the "No Relation" label as evaluation metrics.

JWSC The Winograd Schema Challenge (WSC) is the task to select the antecedent of a pronoun from two candidates (Levesque, 2011). The task itself is coreference resolution but designed to require commonsense reasoning. JWSC[8] is constructed by translating the Rahman and Ng (2012) version of WSC into Japanese.

As we excluded the event pairs containing demonstrative pronouns so as not to generate problems that require more context, there is concern that intermediate-task training on KUCI with pseudo-problems might hurt performance on JWSC due to forgetting the knowledge about demonstratives. Accordingly, we recast JWSC as binary question answering by replacing a pronoun with each antecedent candidate. The resulting dataset is bal-

[7] https://github.com/ku-nlp/KWDLC

[8] https://github.com/ku-nlp/Winograd-Schema-Challenge-Ja

Model	Setting	Acc.
BERT	KUCI	79.3 ± 0.2
	KUCI + Pseudo-problems ($\lambda = 0.1$)	84.1 ± 0.1
	KUCI + Pseudo-problems ($\lambda = 0.5$)	$\mathbf{84.7 \pm 0.1}$
	KUCI + Pseudo-problems ($\lambda = 1.0$)	84.6 ± 0.2
	AMLM $\rightarrow$ KUCI	83.9 ± 0.1
XLM-R	KUCI	86.0 ± 0.1
	KUCI + Pseudo-problems ($\lambda = 0.1$)	88.5 ± 0.1
	KUCI + Pseudo-problems ($\lambda = 0.5$)	$\mathbf{88.8 \pm 0.1}$
	KUCI + Pseudo-problems ($\lambda = 1.0$)	88.6 ± 0.1
	AMLM $\rightarrow$ KUCI	86.2 ± 0.2
Human (Omura et al., 2020)		88.9

Table 1: Accuracy on the test split of KUCI. The scores are the mean and standard deviation over three runs with different random seeds. Arrows denote multi-stage fine-tuning. For instance, "AMLM $\rightarrow$ KUCI" means fine-tuning on KUCI after additional pre-training.

anced and consists of 2,644 / 1,128 examples for training and test split, respectively. Since the development split is not provided, we carried out five-fold cross validation by splitting the training set into 8:2. We trained bert-based logistic regression models and evaluated them by accuracy and Area Under the ROC Curve (AUC).

JCQA JCQA[9] is the Japanese version of CommonsenseQA (Talmor et al., 2019) and consists of 11k five-choice questions regarding a wide range of relations between basic concepts. The questions are based on subgraphs extracted from ConceptNet (Speer et al., 2017) and manually created using crowdsourcing.

Since the task is multiple-choice question answering, we fine-tuned models following the same method described in 3.2.1. We also evaluated the models by accuracy.

3.3 Experimental Results

Commonsense Contingent Reasoning Table 1 shows the experimental results of the commonsense contingent reasoning task. Owing to pseudo-problems, both the BERT and XLM-R models improved the accuracy by 5.4 and 2.8 points, respectively. Notably, the XLM-R model has achieved performance comparable to humans. Putting moderately low weight on pseudo-problems makes the performance slightly better.

Figure 3 shows the learning curves of the models on the development split of KUCI. The crosses

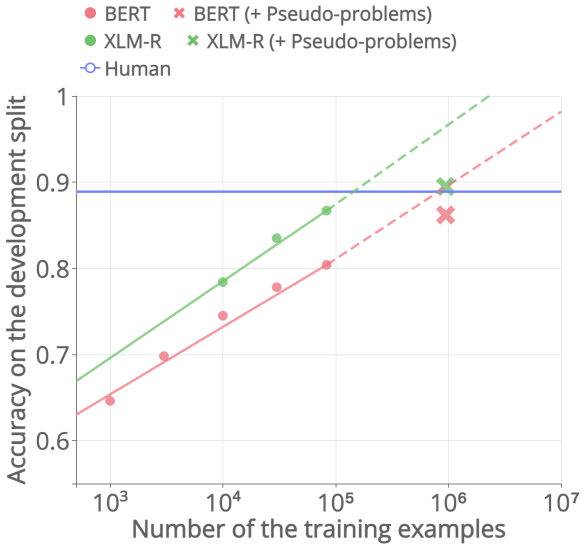

Figure 3: Learning curves of the BERT and XLM-R models on the development split of KUCI. We excluded the degenerate results of the XLM-R model when fine-tuned on a small number of training examples ($N \in \{10^3, 3 \times 10^3\}$).

representing the accuracy on the "KUCI + Pseudo-problems" setting are under the extrapolated learning curves, which implies the difference in quality between the training examples in KUCI and pseudo-problems.

Discourse Relation Analysis As for the related tasks, we can see from Table 2 that intermediate-task training on KUCI with pseudo-problems is effective in discourse relation analysis, particularly in BERT. Since the problems are based on con-

[9] https://github.com/yahoojapan/JGLUE

Model	Setting	Prec.	Rec.	F1
BERT	KWDLC	55.2 ± 2.9	38.4 ± 1.0	45.1 ± 1.1
	KUCI → KWDLC	$\mathbf{58.1 \pm 2.4}$	38.3 ± 1.3	45.7 ± 0.8
	KUCI + Pseudo-problems ($\lambda = 0.5$) → KWDLC	55.9 ± 1.1	$\mathbf{41.0 \pm 2.9}$	$\mathbf{47.0 \pm 2.4}$
	AMLM → KUCI → KWDLC	51.8 ± 3.7	38.4 ± 1.3	43.7 ± 0.7
XLM-R	KWDLC	57.4 ± 1.7	45.5 ± 2.8	50.3 ± 1.3
	KUCI → KWDLC	$\mathbf{57.8 \pm 2.3}$	$\mathbf{48.2 \pm 0.3}$	$\mathbf{51.9 \pm 0.2}$
	KUCI + Pseudo-problems ($\lambda = 0.5$) → KWDLC	57.2 ± 1.0	47.4 ± 1.8	51.5 ± 0.7
	AMLM → KUCI → KWDLC	55.2 ± 1.6	34.5 ± 0.6	40.9 ± 1.0
Human (Crowdworker) (Kishimoto et al., 2020)		54.7	48.6	51.5

Table 2: Performance of discourse relation analysis on KWDLC. The scores are the mean and standard deviation over three runs of five-fold cross-validation with different random seeds. As with Table 1, arrows denote multi-stage fine-tuning. Note that we performed additional Masked Language Modeling (AMLM) on the 881k event pairs used for generating pseudo-problems, not the training examples in KWDLC, to compare the methods of utilizing pseudo-data.

Model	Setting	Ca./Re.	Cond.	Purp.	Justif.	Cont.	Conc.	F1
BERT (ensemble)	KWDLC	76 / 138	32 / 43	18 / 37	0 / 6	2 / 19	54 / 84	46.7
	KUCI → KWDLC	81 / 132	32 / 43	18 / 31	1 / 6	2 / 17	47 / 72	48.0
	KUCI + Pseudo-problems → KWDLC	81 / 139	33 / 49	17 / 29	0 / 4	1 / 12	56 / 85	**48.8**
XLM-R (ensemble)	KWDLC	98 / 159	33 / 46	16 / 34	2 / 4	0 / 18	60 / 88	52.1
	KUCI → KWDLC	109 / 201	34 / 53	18 / 32	3 / 7	0 / 26	56 / 85	51.3
	KUCI + Pseudo-problems → KWDLC	99 / 168	33 / 50	18 / 28	1 / 2	0 / 22	64 / 98	**52.4**
Human (Crowdworker) (Kishimoto et al., 2020)		100 / 175	37 / 54	19 / 44	6 / 32	4 / 30	54 / 67	51.5
Total number of true positives and false negatives		242	54	36	15	6	100	—

Table 3: Detailed results of discourse relation analysis by the ensemble models. The third to eighth columns stand for the discourse relations, "Cause or Reason", "Condition", "Purpose", "Justification", "Contrast", and "Concession", respectively. The values on the left side are the numbers of true positives for the discourse relation, and those on the right side are total numbers of true positives and false positives.

tingent basic event pairs, which are connected by explicit discourse markers representing causal or conditional relation[10], we presume the knowledge about these discourse relations is successfully transferred.

We also describe the detailed results of discourse relation analysis in Table 3. The models transferred from KUCI with pseudo-problems perform better on classifying causal and purpose relations. Compared with crowdworkers, there is room for improvement in precision of concession and infrequent relations.

JWSC The experimental results of JWSC are shown in Table 4. We observed a few degenerate runs[11] (Phang et al., 2018; Pruksachatkun et al., 2020) on the "JWSC" setting despite fine-tuning for 50 epochs. This phenomenon often occurs when training large models on a small dataset, and several studies have reported intermediate-task training can alleviate it (Phang et al., 2018; Pruksachatkun et al., 2020). We also confirmed the same result in this experiment.

We found KUCI is beneficial to JWSC, but pseudo-problems are not necessarily. JWSC contains a non-negligible number of questions regarding concession relation (e.g. "James asked Robert a favor. However, James/Robert declined."), thus we consider putting much emphasis on contingent relation would rather worsen performance. Learning various discourse relations is a promising solution,

[10]These discourse relations are corresponding to "CONTINGENCY:Cause" and "CONTINGENCY:Condition" in the Penn Discourse Treebank (Prasad et al., 2008) and automatically analyzed by the Japanese parser, KNP (Kurohashi and Nagao, 1994).

[11]The training runs that models result in around chance performance. Specifically, we regard less than 0.55 accuracy or AUC as the degenerate runs.

Model	Setting	Acc.	AUC
BERT	JWSC	$66.0 \pm 3.4^{\dagger}$ (68.4 ± 0.1)	$71.4 \pm 4.5^{\dagger}$ (74.5 ± 0.1)
	KUCI → JWSC	$\mathbf{69.9 \pm 0.3}$	$\mathbf{77.0 \pm 0.6}$
	KUCI + Pseudo-problems ($\lambda = 0.5$) → JWSC	68.8 ± 1.1	75.0 ± 2.0
	AMLM → KUCI → JWSC	58.1 ± 1.0	61.9 ± 1.1
XLM-R	JWSC	$78.7 \pm 3.2^{\dagger}$ (80.7 ± 0.4)	$85.6 \pm 4.0^{\dagger}$ (88.0 ± 0.5)
	KUCI → JWSC	$\mathbf{81.2 \pm 0.1}$	$\mathbf{88.7 \pm 0.2}$
	KUCI + Pseudo-problems ($\lambda = 0.5$) → JWSC	80.0 ± 0.2	$\mathbf{88.7 \pm 0.0}$
	AMLM → KUCI → JWSC	50.8 ± 0.5	51.7 ± 0.8

Table 4: Accuracy and AUC on the test split of JWSC. The scores are the mean and standard deviation over three runs of five-fold cross-validation with different random seeds. † denotes the results include a few degenerate runs. We also report the results excluding the degenerate runs in parentheses for reference. As for the "AMLM → KUCI → JWSC" setting of XLM-R, the models failed to learn.

which we leave for future work.

JCQA Referring to Table 5, we can see performance gain regarding XLM-R. We speculate it is thanks to the domain match between pseudo-problems and JCQA, considering the report by Kurihara et al. (2022) that pre-training on CC-100 is more effective in JCQA than Wikipedia. Pseudo-problems alone are somewhat insufficient for adapting to the web domain, but they complement some knowledge.

Comparison to AMLM Although AMLM is somewhat effective in KUCI, it is poor at transferring the knowledge[12]. It can be inferred the models learn task-specific knowledge.

3.4 Qualitative Analysis

Figure 4 shows the example problems that BERT got to answer correctly by incorporating pseudo-problems into training. We can see the improvement in accuracy of the problems regarding quite basic contingent relation like Figure 4. The model sometimes gave low scores to all the choices and appeared to choose by elimination, but we observed it became less frequent. We speculate pseudo-problems complement the lack of coverage of the training examples in KUCI. For further information, we include the confusion matrix in Table 6. The improvement is greater though the model got to make a wrong prediction to some problems.

[12]We also tried the "AMLM → related task" setting, but the performance is generally worse than those on the "AMLM → KUCI → related task" setting.

4 Related Work

Owing to large-scale pre-training, the pre-trained models have achieved unprecedented performance on a variety of NLU tasks, including commonsense reasoning (Wang et al., 2019). Besides such improvement in general language understanding, there have been many approaches to improving the performance on commonsense reasoning tasks.

One group of approaches is to utilize automatically created data, to which our approach belongs. For instance, Ye et al. (2019) performed additional pre-training on 16 million fill-in-the-blank multiple-choice questions generated from Wikipedia and ConceptNet (Speer et al., 2017). They improved the performance on two benchmarks for entity-level commonsense reasoning, CommonsenseQA (Talmor et al., 2019) and Winograd Schema Challenge (WSC) (Levesque, 2011), though their method requires the manually constructed resource (ConceptNet). Staliunaite et al. (2021) proposed a data augmentation method for the Choice of Plausible Alternatives (COPA) and its extension (Roemmele et al., 2011; Kavumba et al., 2019), which consists of roughly three steps: filtering web text by several conditions, extracting causal pairs of clauses with the clue of discourse connectives, and generating distractors using language models. They have not investigated the application to the related tasks, focusing on improving commonsense causal reasoning. Shen et al. (2021) improved unsupervised pronoun resolution and commonsense reasoning by pre-training on

Model	Setting	Acc.
BERT	JCQA	81.8 ± 0.1 (82.3)
	KUCI → JCQA	**82.0 ± 0.3**
	KUCI + Pseudo-problems ($\lambda = 0.5$) → JCQA	81.9 ± 0.2
	AMLM → KUCI → JCQA	68.1 ± 0.4
XLM-R	JCQA	84.0 ± 0.5 (84.0)
	KUCI → JCQA	85.0 ± 0.4
	KUCI + Pseudo-problems ($\lambda = 0.5$) → JCQA	**85.3 ± 0.6**
	AMLM → KUCI → JCQA	75.2 ± 0.5
Human (Kurihara et al., 2022)		98.6

Table 5: Accuracy on the **development** split of JCQA. The scores are the mean and standard deviation over three runs with different random seeds. We also include the reported values in the original paper (the numbers in the parentheses) for reference.

霧が晴れると、 (When a fog clears,) **a. 景色が素晴らしい** **(the scenery is amazing)** b. 川の音がすごい (the sound of river is loud) c. 雪遊びも楽しそうだ (playing in the snow sounds nice) d. 写真写りがいまいちだ (it's not photogenic)	嫌な夢を見ると、 (If I have a bad dream,) a. とりあえず寝る (I'll go to bed for now) b. もう寝ます (I'm going to go to bed now) c. さっさと寝ることにする (I'll go to bed quickly) **d. 目を覚まします** **(I'll wake up)**	午後から病院へいくので (I'm going to see a doctor this afternoon, so) a. 滅多に病院に行かない (I rarely see a doctor) b. 土日は勉強に勤しみます (I'll study hard on weekends) **c. 今日は休暇をとる** **(I take a vacation today)** d. 火曜日は眠い (I'm sleepy on Tuesday)

Figure 4: Example problems that the BERT model got to answer correctly by incorporating pseudo-problems into training. The correct choice is bolded, and the choice that BERT previously selected is highlighted in red.

		KUCI	
		correct	incorrect
KUCI + Pseudo-problems ($\lambda = 0.5$)	correct	7,891	1,028
	incorrect	401	908

Table 6: Confusion matrix organizing the numbers of correct and incorrect answers on the development split of KUCI. The matrix shows the results of the BERT model (ensemble).

auto-generated examples that imitate the task.

As for the second objective of this study, there are several studies about the transferability of commonsense knowledge from existing language resources. For instance, it has been reported intermediate-task training on two benchmarks for commonsense reasoning, Social IQA (Sap et al., 2019b) and WinoGrande (Sakaguchi et al., 2020), helps improve the performance on WSC and COPA. Pruksachatkun et al. (2020) showed the datasets that require complex commonsense reasoning, such as CosmosQA (Huang et al., 2019) and HellaSwag (Zellers et al., 2019), are beneficial to several target tasks. Lourie et al. (2021) ran multi-task learning on multiple resources for commonsense reasoning to examine their interactions. Since they have used the datasets that require complex reasoning, they have not focused on a specific type of commonsense reasoning. We focus on commonsense contingent reasoning and investigate the transferability in the language other than English.

5 Conclusion

In this study, we improved commonsense contingent reasoning by incorporating large-scale pseudo-problems into training. We automatically generated 862k pseudo-problems from a Japanese web corpus of 3.3 billion sentences using the existing data

construction method with modification. Owing to pseudo-problems, a high-performance pre-trained model has achieved near human-level performance on the commonsense contingent reasoning task.

We also investigated the effects of learning contingent knowledge on the related tasks: discourse relation analysis, the Japanese Winograd Schema Challenge, and the JCommonsenseQA. Our experiments demonstrated intermediate-task training on KUCI with pseudo-problems has a positive impact on the related tasks, which indicates the importance of contingent reasoning.

We will further analyze what kind of problems current models still answer incorrectly. From the qualitative analysis, we consider building a language resource for evaluating deeper language understanding. As another research direction, it is also tempting to pursue the improvement in NLU by learning various discourse relations between entities or events in documents.

Acknowledgements

We thank anonymous reviewers for their valuable comments. This work was supported by the Japan Kanji Aptitude Testing Foundation. This work was also supported by Information/AI/Data Science Doctoral Fellowship of Kyoto University and Grant-in-Aid for JSPS Fellows #22J15958.

References

Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel Ziegler, Jeffrey Wu, Clemens Winter, Chris Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. 2020. Language Models are Few-Shot Learners. In *Proc. NeurIPS2020*, volume 33, pages 1877–1901. Curran Associates, Inc.

Alexis Conneau, Kartikay Khandelwal, Naman Goyal, Vishrav Chaudhary, Guillaume Wenzek, Francisco Guzmán, Edouard Grave, Myle Ott, Luke Zettlemoyer, and Veselin Stoyanov. 2020. Unsupervised Cross-lingual Representation Learning at Scale. In *Proc. ACL2020*, pages 8440–8451, Online. Association for Computational Linguistics.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. In *Proc. NAACL2019*, pages 4171–4186, Minneapolis, Minnesota. Association for Computational Linguistics.

Suchin Gururangan, Ana Marasović, Swabha Swayamdipta, Kyle Lo, Iz Beltagy, Doug Downey, and Noah A. Smith. 2020. Don't Stop Pretraining: Adapt Language Models to Domains and Tasks. In *Proc. ACL2020*, pages 8342–8360, Online. Association for Computational Linguistics.

Lifu Huang, Ronan Le Bras, Chandra Bhagavatula, and Yejin Choi. 2019. Cosmos QA: Machine Reading Comprehension with Contextual Commonsense Reasoning. In *Proc. EMNLP2019*, pages 2391–2401, Hong Kong, China. Association for Computational Linguistics.

Jena D. Hwang, Chandra Bhagavatula, Ronan Le Bras, Jeff Da, Keisuke Sakaguchi, Antoine Bosselut, and Yejin Choi. 2021. COMET-ATOMIC 2020: On Symbolic and Neural Commonsense Knowledge Graphs. In *Proc. AAAI2021*, pages 6384–6392. AAAI Press.

Pride Kavumba, Naoya Inoue, Benjamin Heinzerling, Keshav Singh, Paul Reisert, and Kentaro Inui. 2019. When Choosing Plausible Alternatives, Clever Hans can be Clever. In *Proc. COIN*, pages 33–42, Hong Kong, China. Association for Computational Linguistics.

Daisuke Kawahara, Yuichiro Machida, Tomohide Shibata, Sadao Kurohashi, Hayato Kobayashi, and Manabu Sassano. 2014a. Rapid Development of a Corpus with Discourse Annotations using Two-stage Crowdsourcing. In *Proc. COLING2014*, pages 269–278, Dublin, Ireland. Dublin City University and Association for Computational Linguistics.

Daisuke Kawahara, Daniel Peterson, Octavian Popescu, and Martha Palmer. 2014b. Inducing Example-based Semantic Frames from a Massive Amount of Verb Uses. In *Proc. EACL2014*, pages 58–67, Gothenburg, Sweden. Association for Computational Linguistics.

Yudai Kishimoto, Shinnosuke Sawada, Yugo Murawaki, Daisuke Kawahara, and Sadao Kurohashi. 2018. Improving Crowdsourcing-Based Annotation of Japanese Discourse Relations. In *Proc. LREC2018*, Miyazaki, Japan. European Language Resources Association (ELRA).

Yudai Kishimoto, Murawaki Yugo, Daisuke Kawahara, and Sadao Kurohashi. 2020. Japanese Discourse Relation Analysis: Task Definition, Connective Detection, and Corpus Annotation. *Journal of Natural Language Processing*, 27(4):889–931. (in Japanese).

Kentaro Kurihara, Daisuke Kawahara, and Tomohide Shibata. 2022. JGLUE: Japanese General Language Understanding Evaluation. In *Proc. LREC2022*, Marseille, France. European Language Resources Association (ELRA).

Sadao Kurohashi and Makoto Nagao. 1994. KN Parser: Japanese Dependency/Case Structure Analyzer. In *Proceedings of the Workshop on Sharable Natural Language*, pages 48–55.

Hector J. Levesque. 2011. The Winograd Schema Challenge. In *Proc. AAAI2011 Spring Symposium*. AAAI.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. RoBERTa: A Robustly Optimized BERT Pretraining Approach. *CoRR*, abs/1907.11692.

Nicholas Lourie, Ronan Le Bras, Chandra Bhagavatula, and Yejin Choi. 2021. UNICORN on RAINBOW: A Universal Commonsense Reasoning Model on a New Multitask Benchmark. In *Proc. AAAI2021*, pages 13480–13488. AAAI Press.

Marius Mosbach, Maksym Andriushchenko, and Dietrich Klakow. 2021. On the Stability of Fine-tuning BERT: Misconceptions, Explanations, and Strong Baselines. In *Proc. ICLR2021*.

Nasrin Mostafazadeh, Nathanael Chambers, Xiaodong He, Devi Parikh, Dhruv Batra, Lucy Vanderwende, Pushmeet Kohli, and James Allen. 2016. A Corpus and Cloze Evaluation for Deeper Understanding of Commonsense Stories. In *Proc. NAACL2016*, pages 839–849, San Diego, California. Association for Computational Linguistics.

Kazumasa Omura, Daisuke Kawahara, and Sadao Kurohashi. 2020. A Method for Building a Commonsense Inference Dataset based on Basic Events. In *Proc. EMNLP2020*, pages 2450–2460, Online. Association for Computational Linguistics.

Jason Phang, Thibault Févry, and Samuel R. Bowman. 2018. Sentence Encoders on STILTs: Supplementary Training on Intermediate Labeled-data Tasks. *CoRR*, abs/1811.01088.

Rashmi Prasad, Nikhil Dinesh, Alan Lee, Eleni Miltsakaki, Livio Robaldo, Aravind Joshi, and Bonnie Webber. 2008. The Penn Discourse TreeBank 2.0. In *Proc. LREC2008*, Marrakech, Morocco. European Language Resources Association (ELRA).

Yada Pruksachatkun, Jason Phang, Haokun Liu, Phu Mon Htut, Xiaoyi Zhang, Richard Yuanzhe Pang, Clara Vania, Katharina Kann, and Samuel R. Bowman. 2020. Intermediate-Task Transfer Learning with Pretrained Language Models: When and Why Does It Work? In *Proc. ACL2020*, pages 5231–5247, Online. Association for Computational Linguistics.

Altaf Rahman and Vincent Ng. 2012. Resolving Complex Cases of Definite Pronouns: The Winograd Schema Challenge. In *Proc. EMNLP2012*, pages 777–789, Jeju Island, Korea. Association for Computational Linguistics.

Melissa Roemmele, Cosmin Bejan, and Andrew Gordon. 2011. Choice of Plausible Alternatives: An Evaluation of Commonsense Causal Reasoning. In *Proc. AAAI2011 Spring Symposium*. AAAI.

Keisuke Sakaguchi, Ronan Le Bras, Chandra Bhagavatula, and Yejin Choi. 2020. WinoGrande: An Adversarial Winograd Schema Challenge at Scale. In *Proc. AAAI2020*, pages 8732–8740. AAAI Press.

Maarten Sap, Ronan Le Bras, Emily Allaway, Chandra Bhagavatula, Nicholas Lourie, Hannah Rashkin, Brendan Roof, Noah A. Smith, and Yejin Choi. 2019a. ATOMIC: An Atlas of Machine Commonsense for If-Then Reasoning. In *Proc. AAAI2019*, pages 3027–3035. AAAI Press.

Maarten Sap, Hannah Rashkin, Derek Chen, Ronan Le Bras, and Yejin Choi. 2019b. Social IQa: Commonsense Reasoning about Social Interactions. In *Proc. EMNLP2019*, pages 4463–4473, Hong Kong, China. Association for Computational Linguistics.

Ming Shen, Pratyay Banerjee, and Chitta Baral. 2021. Unsupervised Pronoun Resolution via Masked Noun-Phrase Prediction. In *Proc. ACL2021*, pages 932–941, Online. Association for Computational Linguistics.

Tomohide Shibata, Shotaro Kohama, and Sadao Kurohashi. 2015. Construction and Analysis of Japanese Winograd Schema Challenge. In *Proceedings of the Twenty-first Annual Meeting of the Association for Natural Language Processing*, Kyoto, Japan. (in Japanese).

Robyn Speer, Joshua Chin, and Catherine Havasi. 2017. ConceptNet 5.5: An Open Multilingual Graph of General Knowledge. In *Proc. AAAI2017*, pages 4444–4451. AAAI Press.

Ieva Staliunaite, Philip John Gorinski, and Ignacio Iacobacci. 2021. Improving Commonsense Causal Reasoning by Adversarial Training and Data Augmentation. In *Proc. AAAI2021*, pages 13834–13842. AAAI Press.

Alon Talmor, Jonathan Herzig, Nicholas Lourie, and Jonathan Berant. 2019. CommonsenseQA: A Question Answering Challenge Targeting Commonsense Knowledge. In *Proc. NAACL2019*, pages 4149–4158, Minneapolis, Minnesota. Association for Computational Linguistics.

Alon Talmor, Ori Yoran, Ronan Le Bras, Chandra Bhagavatula, Yoav Goldberg, Yejin Choi, and Jonathan Berant. 2021. CommonsenseQA 2.0: Exposing the Limits of AI through Gamification. In *Proc. NeurIPS2021 Datasets and Benchmarks Track (Round 1)*.

Alex Wang, Yada Pruksachatkun, Nikita Nangia, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel R. Bowman. 2019. SuperGLUE: A Stickier Benchmark for General-Purpose Language Understanding Systems. In *Proc. NeurIPS2019*, pages 3266–3280. Curran Associates, Inc.

Guillaume Wenzek, Marie-Anne Lachaux, Alexis Conneau, Vishrav Chaudhary, Francisco Guzmán, Armand Joulin, and Edouard Grave. 2020. CCNet: Extracting High Quality Monolingual Datasets from Web Crawl Data. In *Proc. LREC2020*, pages 4003–4012, Marseille, France. European Language Resources Association.

Zhi-Xiu Ye, Qian Chen, Wen Wang, and Zhen-Hua Ling. 2019. Align, Mask and Select: A Simple Method for Incorporating Commonsense Knowledge into Language Representation Models. *CoRR*, abs/1908.06725.

Rowan Zellers, Yonatan Bisk, Roy Schwartz, and Yejin Choi. 2018. SWAG: A Large-Scale Adversarial Dataset for Grounded Commonsense Inference. In *Proc. EMNLP2018*, pages 93–104, Brussels, Belgium. Association for Computational Linguistics.

Rowan Zellers, Ari Holtzman, Yonatan Bisk, Ali Farhadi, and Yejin Choi. 2019. HellaSwag: Can a Machine Really Finish Your Sentence? In *Proc. ACL2019*, pages 4791–4800, Florence, Italy. Association for Computational Linguistics.

A Hyper-parameters

Table 7, 8, 9, 10, and 11 show the hyper-parameters used in the experiments. We found lower learning rate makes the training of the XLM-R model more stable, thus we set the learning rate of the XLM-R model lower than that of BERT.

Name	Value	
	BERT	XLM-R
Epoch	3	
Batch size	32	
Max sequence length	128	
Optimizer	AdamW	
Learning rate	2e-5	5e-6
Weight decay	0.01	0.1
Adam's betas params	(0.9, 0.999)	(0.9, 0.98)
Scheduler	Linear decay with linear warmup	
Warmup proportion	0.1	
Seed	{0, 1, 2}	

Table 7: Hyper-parameters for fine-tuning on KUCI and pseudo-problems.

Name	Value	
	BERT	XLM-R
Epoch	100	
Batch size	256	
Max sequence length	128	
Optimizer	AdamW	
Learning rate	1e-4	
Weight decay	0.01	
Adam's betas params	(0.9, 0.999)	(0.9, 0.98)
Scheduler	Linear decay with linear warmup	
Warmup proportion	0.06	
gradient clipping value	-	0.25
Seed	0	

Table 8: Hyper-parameters for AMLM. Almost all of the hyper-parameters are referred to Gururangan et al. (2020).

Name	Value	
	BERT	XLM-R
Epoch	10	
Patience for early stopping	3	
Batch size	32	
Max sequence length	128	
Optimizer	AdamW	
Learning rate	2e-5	5e-6
Weight decay	0.01	0.1
Adam's betas params	(0.9, 0.999)	(0.9, 0.98)
Scheduler	Linear decay with linear warmup	
Warmup proportion	0.1	
Seed	{0, 1, 2}	

Table 9: Hyper-parameters for fine-tuning on KWDLC.

Name	Value	
	BERT	XLM-R
Epoch		50
Batch size		32
Max sequence length		128
Optimizer		AdamW
Learning rate	2e-5	5e-6
Weight decay	0.01	0.1
Adam's betas params	(0.9, 0.999)	(0.9, 0.98)
Scheduler		Linear decay with linear warmup
Warmup proportion		0.1
Seed		{0, 1, 2}

Table 10: Hyper-parameters for fine-tuning on JWSC. We set the number of epochs to a large value referring to Mosbach et al. (2021).

Name	Value	
	BERT	XLM-R
Epoch		4
Batch size		32
Max sequence length		128
Optimizer		AdamW
Learning rate		2e-5
Weight decay	0.01	0.1
Adam's betas params	(0.9, 0.999)	(0.9, 0.98)
Scheduler		Linear decay with linear warmup
Warmup proportion		0.1
Seed		{0, 1, 2}

Table 11: Hyper-parameters for fine-tuning on JCQA.

A Survey in Automatic Irony Processing: Linguistic, Cognitive, and Multi-X Perspectives

Qingcheng Zeng
Department of Linguistics
Northwestern University
qingchengzeng@outlook.com

An-Ran Li
Department of Chinese and Bilingual Studies
The Hong Kong Polytechnic University
an-ran.li@connect.polyu.hk

Abstract

Irony is a ubiquitous figurative language in daily communication. Previously, many researchers have approached irony from linguistic, cognitive science, and computational aspects. Recently, some progress have been witnessed in automatic irony processing due to the rapid development in deep neural models in natural language processing (NLP). In this paper, we will provide a comprehensive overview of computational irony, insights from linguistic theory and cognitive science, as well as its interactions with downstream NLP tasks and newly proposed multi-X irony processing perspectives.

1 Introduction

Irony, which generally refers to the expressions that have opposite literal meanings to real meanings, is a representative rhetoric device in human languages (Li and Huang, 2020). For example, in sentences *I just love when you test my patience!!*, and *Had no sleep and have got school now*, compared to their literal meanings, both sentences are conveying reverse meanings and emotions, which mean *not love* and *not happy*. People are using ironies to communicate their affective states more implicitly or explicitly, to strengthen the claims depending on the needs, which contribute to neologism across human languages. However, the inner incongruity makes ironic expressions harder for machines to understand. Consequently, accurate irony processing systems are essential and challenging for downstream tasks and natural language understanding (NLU) research.

Previously, many researchers approached irony processing from various perspectives. The last survey in computational irony was published seven years ago (Wallace, 2015) and mainly focused on irony detection and pragmatic context model. Therefore, a systematic and comprehensive survey on automatic irony processing remains absent, encouraging us to focus on a review of advances in irony processing, from traditional machine learning, recurrent neural networks (RNNs) methods, to deeper pretrained language models (PLMs) throughout. Besides, del Pilar Salas-Zárate et al. (2020) revealed the great imbalance in figurative language research, in which sarcasm was dominant and twice as much as irony research. With this review, We aim to encourage a balanced and equal research environment in figurative languages.

Moreover, the design and further improvement in present irony processing systems should be concatenated with both the theoretical accounts in irony theories and its role in communication, followed by downstream applications in NLP tasks. Generally, most researchers approached irony processing as a single irony detection branch and three main shared tasks (Ghanem et al., 2019; Van Hee et al., 2018a; Ortega-Bueno et al., 2019) all focused on irony detection, whereas other aspects are constantly under-explored. And theoretical-informed or cognitive-informed discussions are rarely seen in irony research compared to tiny amendments in neural networks' architectures.

In this paper, we aim to offer a comprehensive review in irony processing from machine learning, linguistic theory, cognitive science, and newly proposed multi-X perspectives and to evaluate irony processing in NLP applications. The remaining sections are organized as follows. In the second part, we will approach irony theories in linguistics and cognitive science research, and discuss the concrete differences between irony and sarcasm. Then, we will review irony datasets in world languages and discuss potential problems in annotation schemes throughout. In the fourth part we will retrospect research progress in automatic irony processing, including traditional feature engineering, neural network architectures, to PLMs and relatively under-explored research fields in irony processing. Finally, we will discuss irony's in-

Proceedings of the 29th International Conference on Computational Linguistics, pages 824–836
October 12–17, 2022.

teractions with downstream NLP tasks like sentiment analysis and opinion mining, and multi-X perspectives for further development in computational irony research.

2 Theoretical Research in Irony

2.1 Irony Theories

Various definitions have been given to irony. Early studies suggested that irony is the expression whose real meaning is contradictory to its literal meaning (Grice, 1975). The Merriam-Webster Dictionary, The Oxford English Dictionary, and The Collins English Dictionary all adopted this definition and used the words "opposite" or "contrary" to explain the relationship between the literal and contextual meanings of irony.

However, more research into various types of ironic examples revealed that the contextual meaning of irony does not have to be "opposite" or "contrary" to the literal one. According to Sperber and Wilson (1986); Wilson and Sperber (2012), some expressions have no "literal meaning" to be challenged because no "literal meaning" is mentioned in the context, based on which they raised **relevance theory** and the "echoic" concept. They considered irony as "an echoic use of language in which the speaker tacitly dissociates herself from an attributed utterance or thought" (Wilson, 2006). That is, if the "echoic use" is incongruous in some ways, the expression can be ironic. Based on this theory, Seto (1998) put forward that there are some "echo-markers" like *definitely*, *really*, and *indeed*.

Li and Huang (2020) provided instances to show that "incongruity" does not have to be between the literal and contextual meanings of irony in certain circumstances. They believed that irony's true nature is a psychological activity as much as a verbal representation. The speaker or listeners must finish the "reversal" process on a psychological level for it to be completed. When compared to the concepts of "echoic" and "incongruity," "reversal" is concerned not only with the results but also with the psychological processes that the speakers/listeners go through.

2.1.1 Types of Irony

Booth (1974) divided irony into tragic and comic irony by literary genre, as well as stable and unstable irony by determinacy. He also categorized irony into dramatic irony, situational irony, verbal irony, and rhetorical irony by the range of context it has to refer to. Researchers in computational irony paid the most attention to situational irony and verbal irony. Situational irony is a circumstance in which the outcome differs from what was expected, or a situation that contains striking contrasts (Imagine a situation that a lifeguard is saved from drowning). Verbal irony is the expression in which the speaker's intended meaning significantly different from the literal one. Abrams and Harpham (2014) considered that verbal irony usually involves the explicit presentation of one attitude or evaluation, but with signals that the speaker wants to express a totally different attitude or opinion. It differs from situational irony in that it is purposely manufactured by speakers. For example, *Very well, keep insulting me!*

Verbal irony is the kind to which computational linguistics researchers pay the most attention. Li (2021) made a further classification of them. She put forward eight kinds of reversals which are rhetorical reversal, expectation reversal, evaluation reversal, reversal of sentiment, reversal of factuality, relationship reversal, reversal from opposite pair, and reversal from satiation. The paper considered that verbal ironies can be classified by the kind of reversal they generate.

2.2 Linguistic Features

2.2.1 Irony Markers and Constructions

Although most of the studies saw irony as a pragmatics phenomenon, people also considered that it can be reflected on the verbal, grammatical, or semantic level. For example, on the verbal level, people often use words like *thank*, *congratulate*, *welcome*, *happy*, and *interesting* to express ironic meanings. Laszlo (2017) found 15 core evaluative words which often show in ironic expressions. She generated patterns from these core evaluative words to extract ironic sentences from the corpus. For example, when the word *love* is in the pattern "NP + would/ 'd/ wouldn't + love", it is highly possible to be an ironic expression. On a grammatical level, people often use the subjunctive when they intend to be ironic. Besides that, semantic conflict is the most direct way to express ironic meaning. The incompatibility between the main words of the proposition leads to the ridiculousness of the proposition (e.g. It's very considerate of you to make such a loud noise while I was asleep). Besides, (Ghosh and Muresan, 2018) also categorized irony markers according to trope, morphosyntactic, and

typographic types.

Li (2021) considered that ironies are often expressed by specific "constructions", especially in short discourses. Larger than "core evaluative words" in Laszlo (2017), the "constructions" mentioned in Li (2021) are mostly in the form of idioms or phrases. The crucial feature of them is the lack of predictability. Most of them do not have to rely on too much contextual information, they themselves can provoke the process of reversal for readers or listeners (e.g. 贵人多忘事 (honorable people frequently forget things)).

2.2.2 Irony in Communication

Researchers claim that by using ironies, people have several kinds of intentions.

Be polite: According to Brown et al. (1987), when unfavorable attitudes such as resistance, criticism, and complaints are stated with irony, the threat to the listener's reputation is reduced. The irony, as stated in Giora (1995), is an indirect negation. People prefer to utilize indirect negation to be polite to their listeners because direct negation can generate great unhappiness;

Ease criticisms: As reported by Dews and Winner (1995), irony helps to ease the expression's evaluative function. They believe that the incompatibility between literal meaning and contextual meaning can make it difficult to articulate negative feelings. However, Toplak and Katz (2000) argued that, while irony literally avoids conflict, it is more aggressive from the perspective of the speaker's goal;

Self-protection: Sperber and Wilson (1986) proposed the "echoic" idea, which stated that irony is a detached utterance that is simply an echo of another people's thought. It's a self-protection tactic, especially when the speakers are members of marginalized groups. According to Gibbs (2000), the irony is an "off-record" statement that allows speakers to deny their true intentions and avoid being challenged;

Be amusing: Gibbs (2000) reported that when young people intend to be humorous, 50% of their communication is ironic. It can assist people in creating a dialogue platform on which speakers and listeners can agree and communicate more easily.

2.3 Irony and Sarcasm

Most of the studies saw sarcasm as a subset of irony (Tannen et al., 2005; Barbe, 1995; Kumon-Nakamura et al., 1995; Leggitt and Gibbs, 2000; Bowes and Katz, 2011). Sarcasm is often recognized as "a nasty, mean-spirited, or just relatively negative form of verbal irony, used on occasion to enhance the negativity expressed relative to direct, non-figurative criticism" (Colston, 2017).

One of the peculiarities of sarcasm is whether or not the speakers intend to offend the listeners. Kumon-Nakamura et al. (1995), for example, believe that sarcastic irony always conveys a negative attitude and is intended to harm the object being discussed. The non-sarcastic irony, on the other hand, can communicate either a good or negative attitude, and it is rarely meant to be hurtful. Barbe (1995) concurred that the core difference was "hurtful". She claimed that irony is a face-saving strategy while sarcasm is a face-threatening action. Ridicule is another feature of sarcasm. According to Lee and Katz (1998), sarcasm is closer to ridicule than irony. Their experiment revealed that sarcasm is directed at a single person, but irony is directed toward a large group of people. Haiman et al. (1998) claimed that one of the most distinguishing characteristics of sarcasm is that the literal meaning of its words is always positive. However, he did not convey his thoughts on irony. Whereas Littman and Mey (1991) viewed this topic from another angle. While there are many various forms of ironies, they believe that there is only one type of sarcasm because "sarcasm cannot exist independently of the communication setting."

Cognitive scientists approached the difference in experimental studies. Previous research in child language acquisition (Glenwright and Pexman, 2010) reported that children understood sarcastic criticism later than they could understand the non-literal meanings of irony and sarcasm, implying different pragmatic purposes of irony and sarcasm. Filik et al. (2019) utilized fMRI and found out sarcasm is associated with wider activation of semantic network in human brains compared to irony.

However, most computational linguistics researchers used irony and sarcasm interchangeably, since the boundary between these two concepts is too vague for even human beings, let alone for machines. Joshi et al. (2016) and Sulis et al. (2016) verified this claim from both human annotators and computational perspectives. Although in this paper we will mainly focus on the literal **irony processing** and discuss the inspirations from recent research output in sarcasm processing, we aim to encourage **unify sarcasm under the framework**

of irony via fine-grained annotation schemes.

3 Irony Datasets Perspectives

3.1 Irony Textual Datasets

Some main target databases for irony processing research include social media platforms like Twitter and online shopping websites like Amazon. For example, Reyes and Rosso (2012) collected a 11,861-document irony dataset based on customer reviews from several websites. Some other preliminary attempts to build irony benchmark datasets is Reyes et al. (2012) and Reyes et al. (2013), in which they used self-generated hashtag *#irony* as the gold standard and constructed 40,000-tweet and 50000-tweet datasets from Twitter respectively, each including 10,000 ironic tweets and remaining non-ironic ones. The irony benchmark dataset that is now widely used is from Van Hee et al. (2018a), consisting of 4,792 tweets and half of them were ironic. This dataset was also constructed via searching hashtags including *#irony*, *#sarcasm*, and *#not*.

There were also tremendous attempts to construct benchmark datasets in other languages. For example, Tang and Chen (2014) firstly built the NTU Irony Corpus including 1,005 ironic messages from Plurk, a Twitter-like social media platform, by mining specific ironic patterns and manually checking extracted messages. A recent Chinese benchmark dataset for irony detection was constructed by Xiang et al. (2020), which includes 8,766 Weibo posts, labelled from *not ironic* to *strongly ironic* in a five-scale system. Besides, irony datasets in Spanish, Greek, and Italian are also widely available. A comprehensive overview of the irony datasets is listed in Table 1.

3.2 Data Source and Construction Methodology

Twitter, as one of the most trending social platforms, is the major source of irony benchmark datasets. Given Plurk and Weibo's similarity to Twitter, online short-text social medias are almost the only origin for present datasets, which might lead to several potential problems. For example, the limitation of 140-word introduced specific bias towards short-text classification and long texts remain a problem. Besides, the judgment of irony might be highly dependent on contextual information like previous comments or retweets and one single tweet could be meaningless itself. At last, topics on social media platforms might also be highly biased towards political or sports topics. Interestingly, previous research (Ghosh and Muresan, 2018) reported that for Twitter and Reddit, different ironic markers played the most important roles, further emphasizing the needs of multiple sources for robust NLU.

As for the construction methodology, most datasets adopted "keywords and hashtags" filtering strategy, and some of them followed by human annotations. With annotation schemes put aside, Sykora et al. (2020) explored how self-generated tags could correspond to real labels correctly via a manual semantic analysis. At the worst case, only 16% of the tagged *#sarcasm* tweets are unambiguous sarcastic tweets according to well-trained linguists, which further emphasized the necessity of human annotation.

3.3 Annotation Schemes

As shown in Table 1, various datasets have been labelled differently. Most datasets were annotated as binary classes, ironic versus non-ironic. Both Chinese datasets adopted different strategies by annotating ironic elements and intensity respectively. Van Hee et al. (2018a) is the only one doing fine-grained labelling, into verbal irony by polarity contrast, other verbal irony, situational irony, and non-ironic.

Another annotation scheme was proposed by Cignarella et al. (2020b). They annotated irony activators at the morphosyntactic level and distinguished different types of irony activation. This annotation scheme dived into syntactic information and offered information for analyzing ironical constructions.

3.4 Future work in datasets

A high-quality benchmark dataset is crucial to measure NLU capability and advance future research. We are calling improvements for a future benchmark dataset from following perspectives.

1) Diverse data sources like literature, daily conversations, and news articles should be involved, and the distribution of text length should be relatively balanced.

2) A uniform annotation scheme and strategy is awaiting for construction. For example, situational irony is needed; ironic intensity should be labelled as reference rather than exclusive labels in Xiang et al. (2020); a scheme to unify sarcasm and irony could be expected.

Study	Language	Data Source	Construction Methodology	Size	Annotation Scheme
Reyes and Rosso (2012)	English	online websites	filtering low-star reviews	11861	ironic / non-ironic
Reyes et al. (2013)	English	Twitter	#irony hashtag	40000	ironic / non-ironic
Wallace et al. (2014)	English	Reddit	Sub-reddit	3020	ironic / non-ironic
Van Hee et al. (2016)	English, Dutch	Twitter	#irony, #sarcasm, and #not hashtags	3000, 3179	three-scale irony annotation
Van Hee et al. (2018a)	English	Twitter	#irony, #sarcasm, and #not hashtags	4792	four-scale ironic types
Tang and Chen (2014)	Chinese	Plurk	pattern mining	1005	annotating ironic elements
Xiang et al. (2020)	Chinese	Weibo	pattern mining	8766	five-scale ironic intensity
Barbieri et al. (2016)	Italian	Twitter	keywords, hashtags, and etc.	9410	ironic / non-ironic
Cignarella et al. (2018)	Italian	Twitter	keywords, hashtags, and etc.	4849	ironic / non-ironic / sarcastic
Charalampakis et al. (2015)	Greek	Twitter	keywords	61427	ironic / non-ironic
Ortega-Bueno et al. (2019)	Spanish	Twitter	comments and tweets	9000	ironic / non-ironic
Ghanem et al. (2019)	Arabic	Twitter	keywords and hashtags	5030	ironic / non-ironic
Corrêa et al. (2021)	Portuguese	Twitter and news articles	keywords, hashtags, and news articles	34306	ironic / non-ironic
Vijay et al. (2018)	Hindi-English code-mixed	Twitter	keywords and hashtags	3055	ironic / non-ironic

Table 1: Irony benchmark datasets

3) Multi-X perspectives should be incorporated into the construction of datasets. For example, a multimodal and multilingual dataset could enhance irony identification in a grounded environment.

4 Irony Processing Systems

In this section, we will discuss the progress of irony processing systems comprehensively, organized along the development of machine learning and deep learning.

4.1 Irony Detection

4.1.1 Rule-based Detection Methods

Tang and Chen (2014) extracted ironic expressions based on five patterns summarized from mandarin Chinese. Li and Huang (2020) further expanded ironic constructions to more than twenty and proposed a systematic irony identification procedure (IIP). Besides Chinese, Frenda (2016) utilized sentiment lexicons, verb morphology, quotation marks, and etc. to design an Italian irony detection model and got competitive performance with machine learning models. However, rule-based models are too complex and hard to generalize for wider applications.

4.1.2 Supervised non-neural network era

Most research took irony detection as a simple classification problem. Before the popularity of deep learning, feature engineering is crucial for accurate irony detection. Generally, features could be divided into several levels.

Lexical features Lexical features are at the foundational level of NLP features, basically divided into bags of words (BOW) sets, word form sets, and conditional n-gram probabilities (Van Hee et al., 2018b). Representative BOW sets mainly include n-grams and character n-grams. Word form sets focus on number and frequency, such as punctuation numbers, emoticon frequencies, character repetitions, etc. Despite their easiness to get, lexical features were proved effective in much research.

Syntactic features Syntax is mainly quantified via parts-of-speech and named entities. After tagging, the number and frequency of both characteristics could act as features in classification models. Besides, hand-crafted syntactic features also included clash before verb tenses (Reyes et al., 2013) and dependency parsing (Cignarella et al., 2020a).

Semantic features Van Hee et al. (2018b) approached semantic features based on the presence or not in semantic clusters, which were trained on a irony Twitter corpus with the Word2Vec algorithm (Mikolov et al., 2013).

Linguistic-motivated features Irony processing is deeply associated with sentiments and emotions. Therefore, researchers have offered many characteristics to capture irony patterns. For example, Reyes et al. (2013) proposed the feature of *contextual imbalance*, which was quantified via measuring the semantic similarity pairwise. Generally, most features could be categorized into *ambiguity* (Reyes et al., 2012) and *incongruity* (Joshi et al., 2015). Take incongruity as an example, implicit incongruity was defined as a boolean feature checking containing implicit sentiment phrases or not; explicit incongruity was defined as number of times a polarity contrast appears. Theoretical research (Ghosh et al., 2020) is encouraging more semantic and pragmatic features to better capture ironies.

Features at various levels were concatenated with classifiers, including naive bayes, decision tress, support vector machines (SVMs), etc. (Van Hee et al., 2018b) to get final classification results.

4.1.3 Supervised Neural Network Era

There have been irony detection tasks in various languages after 2015, among which most participants used convolutional neural networks (CNNs) and RNNs methods to detect ironical expressions. For example, in SemEval-2018 Task 3 (Van Hee et al., 2018a), Wu et al. (2018) classified irony tweets and their ironical types in a densely connected LSTM network, together with multitask learning (MTL) objectives in the optimization. In IroSvA (Ortega-Bueno et al., 2019), González et al. (2019) utilized transformer encoders only for detecting Spanish ironical tweets.

Besides, Ilić et al. (2018) firstly utilized contextualized word representations ELMo (Peters et al., 2018) with Bi-LSTM to detect ironies. Zhang et al. (2019) enhanced irony detection with sentiment corpora based on attention Bi-LSTM RNNs, and achieved state-of-the-art results on Reyes et al. (2013) dataset.

4.1.4 Pretraining and Fine-tuning Paradigm

Since BERT (Devlin et al., 2019), the "pretraining and fine-tuning paradigm" has become the mainstream in NLP research due to its extraordinary capacity in dealing with contextualized information and learning general linguistic knowledge. Recent development in irony detection also witnessed the usage of PLMs. Xiang et al. (2020) released several baseline results along with the dataset, among which BERT had highest accuracy (5% higher than Bi-LSTM methods). Potamias et al. (2020) proposed a recurrent convolutional neural network (RCNN)-RoBERTa (Liu et al., 2019) strategy and improved the results on the SemEval-2018 dataset by a large degree. Besides, Cignarella et al. (2020a) explored syntax-augmented irony detection with multilingual BERT (mBERT) in multilingual settings. To sum up, some representative studies in irony detection are detailed in Table 2.

4.2 Irony Generation

Irony generation is mostly an underexplored research field besides Zhu et al. (2019), in which they defined irony generation as a style transfer problem, and utilized a Seq2Seq framework (Sutskever et al., 2014) with reinforcement learning to generate ironical counterparts from a non-ironic sentence. Concretely, they designed the overall reward as a harmonic mean of irony reward and sentiment reward, which was trying to capture the sentiment incongruity. In terms of the evaluation, besides traditional natural language generation metrics like BLEU, they also designed task-specific evaluation metrics, which shoule be further enhanced in irony and even figurative language research.

Future work in irony generation could be advanced in new PLMs and theoretical accounts. For example, no attempts were made to generate ironical expressions after generative PLMs like BART (Lewis et al., 2020). Controllable irony generation and its interaction with agents are interesting topics remaining for future exploration. Besides, irony theories could be further utilized. In recent research on unsupervised sarcasm generation (Mishra et al., 2019; Chakrabarty et al., 2020), context incongruity, valence reversal, and semantic incongruity were merged to enhance the generation.

5 Discussion

5.1 Irony for Downstream NLP Tasks

Irony is directly associated with downstream NLU tasks like sentiment analysis and opinion mining. For example, the sentence retrieved from Filatova (2012) *I would recommend this book to friends who have insomnia or those who I absolutely despise.* is classified as positive by fine-tuned sentiment analysis RoBERTa model (Heitmann et al., 2020), which is apparently opposite to human evaluation. Wrong sentiment judgments will potentially lead to contrary opinion mining. We suggest that irony could be further captured through introducing incongruity embedding or specific pattern matching. Joshi et al. (2015) designed linguistic-motivated features implicit and explicit incongruity, which are inspiring for enhancing irony understanding. Consider another example task, machine translation, in which wrong translation will potentially lead to totally opposite meanings. We encourage to model discourse features (Voigt and Jurafsky, 2012), such as ironic patterns and punctuation as embeddings for robust irony translation.

In addition, we are looking forward to the research of irony and sarcasm processing in NLP for social good (NLP4SG), especially considering the strong sentiments hidden in ironies. A recent work (Chia et al., 2021) explored cyberbullying detection and this was a starting point to handle online harmful ironical contents.

5.2 Multi-X Perspectives

Recent developments in NLP and sarcasm processing encourage us to approach discourse processing

Study	Input Features	Architecture	Dataset	Performance (F1 Score)
Reyes et al. (2013)	hand-crafted high-level features	decision tree	Reyes et al. (2013)*	70 / 76 / 73
Barbieri and Saggion (2014)	frequency, intensity, sentiments, etc.	decision tree	Reyes et al. (2013)*	73 / 75 / 75
Nozza et al. (2016)	unsupervised	topic-irony model	Reyes et al. (2013)*	84.77 / 82.92 / 88.34
Zhang et al. (2019)	word embeddings	sentiment-transferred Bi-LSTM	Reyes et al. (2013)*	94.69 / 95.69 / 96.55
Van Hee et al. (2018b)	lexical, syntactic and semantic features	SVMs	Van Hee et al. (2016)**	70.11
Rohanian et al. (2018)	intensity, contrast, topics, etc.	ensemble voting classifier	Van Hee et al. (2018a)***	65.00 / 41.53
Wu et al. (2018)	word embeddings, POS tags, sentiments, etc.	LSTM + MTL	Van Hee et al. (2018a)***	70.54 / 49.47
Cignarella et al. (2020a)	mBERT output, autoencoders	LSTM	Van Hee et al. (2018a)***	70.6
Potamias et al. (2020)	RoBERTa output	RCNN	Van Hee et al. (2018a)***	80.0
Santilli et al. (2018)	word space vectors, BOW sets	SVMs	Cignarella et al. (2018)****	70.00 / 52.00
Cimino et al. (2018)	word embeddings	LSTM + MTL	Cignarella et al. (2018)****	73.60 / 53.00
González et al. (2019)	word embeddings	transformer encoders	Ortega-Bueno et al. (2019)	68.32

* Three binary classification systems were trained respectively for this dataset.
**This result was obtained on the 3000 English tweets subset.
***This dataset had two sub-tasks, identifying ironic or not and classifying ironic types. The latter two only focused on the first sub-task.
****This dataset had two sub-tasks, irony detection and further identifying sarcasm.

Table 2: Representative Irony Detection Systems

from multiple angles. In this part, we will review and suggest several multi-X perspectives for irony and figurative language processing.

5.2.1 Multimodal Irony Processing

Linguistic interactions are not solely consisted of texts. Besides, facial expressions and speech communications are crucial to convey emotions and feelings. For example, Skrelin et al. (2020) reported people could classify ironies based on phonetic characteristics only. Consequently, it is conceivable that multimodal methods could help with irony detection. Schifanella et al. (2016) made the first attempt in multimodal sarcasm detection, in which they extracted posts from three multimodal social media platforms based on hashtags. Then they used SVMs and neural networks to prove the validity of visual information in enhancing sarcasm detection.

Castro et al. (2019) made a great improvement in multimodal sarcasm detection by introducing audio features into the dataset. The experiments also verified the importance of more modalities in sarcasm processing.

Future work in multimodal irony processing should include a comprehensive multimodal irony dataset based on MUStARD dataset (Castro et al., 2019) with more fine-grained annotation schemes. Additionally, most methods (Pan et al., 2020; Liu et al., 2021) explored sarcasm by introducing inter-modality and intra-modality attention in single-stream setting. How double-stream multimodal pretrained models (MPMs) will encode and interact in complex discourse settings remains an interesting problem to solve.

5.2.2 Multilingual Irony Processing

To understand irony in a multilingual context is even harder due to cultural gaps. Previously listed dataset includes a Hindi-English code-mixed irony dataset (Vijay et al., 2018), in which they offered an example:

- **Text:** The kahawat ‘old is gold’ purani hogaee. Aaj kal ki nasal kehti hai ‘gold is old’, but the old kahawat only makes sense. #MindF #Irony.
- **Translation:** The saying ‘old is gold’ is old. Today’s generation thinks ‘gold is old’ but only the old one makes sense. #MindF #Irony.

Cignarella et al. (2020a) explored how mBERT performed in multiple languages’ irony detection tasks separately. Given it has been proved code-switching patterns are beneficial for NLP tasks like humor, sarcasm, and hate speech detection in RNNs settings (Bansal et al., 2020), A future direction is to merge the irony detection datasets from multiple languages (consider Karoui et al. (2017)) or even code-mixed texts, and explore how multilingual datasets could enhance irony understanding in mBERT.

5.2.3 Multitask Irony Processing

MTL is to make models learn several tasks simultaneously rather than independently once at a time. Recent work in figurative language processing proposed several MTL strategies to improve the performance interactively. Chauhan et al. (2020) proposed a MTL framework to do sentiment, sarcasm, and emotion analysis simultaneously and the framework yielded better performance with the help of MTL.

Generally, we will classify figurative language into several categories like metaphors, parodies, humors, ironies, and etc. However, noted that there are not clear differences between each other, a single task figurative language processing will only focus on one particular aspect and fail to capture the interactions. Recent work (Ao et al., 2022) also verified the combination of humor and sarcasm could improve political parody detection.

PLMs could understand figurative language better than random but apparently worse than human evaluation (Liu et al., 2022). We suggest that future work should consider domain adaptation towards figurative language as a whole via weak supervision. MTL strategy could utilize previous research in corpus linguistics, and design an appropriate proportion in summing the loss function. Besides, a unified framework to model figurative languages could be expected.

5.2.4 Multiagent Irony Processing

Human-like language generation is a central topic in multiagent interactive systems. Besides robots' ironical understanding, we are also curious about how robots could generate ironical expressions. Unlike transferring non-ironic sentences to ironic, multiagent irony measures the performance during the interactions in dialogues. Ritschel et al. (2019) improved the robots by introducing ironic expressions, which showed better user experiences in human evaluation.

Further explorations in multiagent irony could aim at better dialogue state tracking and understand when irony should be introduced.

5.3 New Tasks: Inspiration from Sarcasm

Compared to sarcasm, irony is rarely seen as a term in NLP conferences. Recently we have witnessed great improvements in sarcasm processing and in this part we will discuss how new tasks in sarcasm could motivate irony research.

Data Collection As discussed, datasets are highly dependent on hashtags as a signal to extract ironical expressions. Shmueli et al. (2020) proposed an algorithm to detect sarcastic tweets from a thread based on exterior cue tweets. A distant supervision based method for extracting ironies from platforms is crucial, given ironies in conversational contexts are central topic in the future.

Intended and Perceived Irony Oprea and Magdy (2019) explored how author profiling affected the perceived sarcasm (manual labelling) versus the intended sarcasm (hashtags), and verified the difference between both. Further, Oprea and Magdy (2020) introduced iSarcasm dataset which divided intended sarcasms and perceived sarcasms. The state-of-the-art sarcasm detection models performed obviously worse than human evaluation on this dataset. Future work could focus on multimodal perceived and intended irony, especially across various cultures.

Target Identification Sarcasm target identification was firstly proposed in Joshi et al. (2018), in which sarcasm targets were classified as one target, several targets and outside. Patro et al. (2019) introduced sociolinguistic features and a deep learning framework, and improved target identification by a lot. For irony processing, most ironical expressions do not equip a specific target in itself as previously discussed. However, its ironical effects are likely in dialogue or visually grounded environment, which encourages us to enhance irony datasets in aforementioned ways.

Irony Explanation Irony, according to the definition, have opposite real meanings to literal meanings. However, this does not mean adding a single negation could interpret ironies well. Kumar et al. (2022) proposed a new task, sarcasm explanation in dialogue. Irony explanation might encounter more complex problems due to relatively low proportion of targets. Still, we should include irony explanation as a branch of multimodal irony processing like Desai et al. (2021).

5.4 Explainable Irony Processing

Explainable machine learning is of interest for most researchers to uncover the blackbox. In irony processing, we are also curious about why specific expressions are recognized as ironies. Buyukbas et al. (2021) explored explainability in irony detection using Shapley Additive Explanations (SHAP) and Local Interpretable Model-Agnostic Explanations (LIME) methods. Results showed that punctuations and strong words play important roles in irony detection.

For future work, we suggest using explainable methods in multimodal settings and check how different modalities act various roles in making a class label.

6 Conclusion

In this paper, we reviewed the development in automatic irony processing from underexplored theo-

retical and cognitive science to computational perspectives, and offered a comprehensive analysis in future directions. We hope that our work and thinking will encourage further interdisciplinary research between linguistics and human language technology, motivate the research interests in irony and even, figurative languages.

Acknowledgement

This review is based on the first author's previous research proposal. We would like to express our thanks to Professor Chu-Ren Huang and Dr. Yat Mei Lee for their suggestions.

References

Meyer Howard Abrams and Geoffrey Harpham. 2014. *A glossary of literary terms*. Cengage Learning.

Xiao Ao, Danae Sánchez Villegas, Daniel Preoţiuc-Pietro, and Nikolaos Aletras. 2022. Combining humor and sarcasm for improving political parody detection.

Srijan Bansal, Vishal Garimella, Ayush Suhane, Jasabanta Patro, and Animesh Mukherjee. 2020. Code-switching patterns can be an effective route to improve performance of downstream NLP applications: A case study of humour, sarcasm and hate speech detection. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 1018–1023, Online. Association for Computational Linguistics.

Katharina Barbe. 1995. *Irony in context*, volume 34. John Benjamins Publishing.

Francesco Barbieri, Valerio Basile, Danilo Croce, Malvina Nissim, Nicole Novielli, and Viviana Patti. 2016. Overview of the evalita 2016 sentiment polarity classification task. In *Proceedings of third Italian conference on computational linguistics (CLiC-it 2016) & fifth evaluation campaign of natural language processing and speech tools for Italian. Final Workshop (EVALITA 2016)*.

Francesco Barbieri and Horacio Saggion. 2014. Modelling irony in Twitter. In *Proceedings of the Student Research Workshop at the 14th Conference of the European Chapter of the Association for Computational Linguistics*, pages 56–64, Gothenburg, Sweden. Association for Computational Linguistics.

Wayne C Booth. 1974. *A rhetoric of irony*. University of Chicago Press.

Andrea Bowes and Albert Katz. 2011. When sarcasm stings. *Discourse Processes*, 48(4):215–236.

Penelope Brown, Stephen C Levinson, and Stephen C Levinson. 1987. *Politeness: Some universals in language usage*, volume 4. Cambridge university press.

Ege Berk Buyukbas, Adnan Harun Dogan, Asli Umay Ozturk, and Pinar Karagoz. 2021. Explainability in irony detection. In *International Conference on Big Data Analytics and Knowledge Discovery*, pages 152–157. Springer.

Santiago Castro, Devamanyu Hazarika, Verónica Pérez-Rosas, Roger Zimmermann, Rada Mihalcea, and Soujanya Poria. 2019. Towards multimodal sarcasm detection (an _Obviously_ perfect paper). In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 4619–4629, Florence, Italy. Association for Computational Linguistics.

Tuhin Chakrabarty, Debanjan Ghosh, Smaranda Muresan, and Nanyun Peng. 2020. R^3: Reverse, retrieve, and rank for sarcasm generation with commonsense knowledge. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7976–7986, Online. Association for Computational Linguistics.

Basilis Charalampakis, Dimitris Spathis, Elias Kouslis, and Katia Kermanidis. 2015. Detecting irony on greek political tweets: A text mining approach. In *Proceedings of the 16th International Conference on Engineering Applications of Neural Networks (INNS)*, pages 1–5.

Dushyant Singh Chauhan, Dhanush S R, Asif Ekbal, and Pushpak Bhattacharyya. 2020. Sentiment and emotion help sarcasm? a multi-task learning framework for multi-modal sarcasm, sentiment and emotion analysis. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 4351–4360, Online. Association for Computational Linguistics.

Zheng Lin Chia, Michal Ptaszynski, Fumito Masui, Gniewosz Leliwa, and Michal Wroczynski. 2021. Machine learning and feature engineering-based study into sarcasm and irony classification with application to cyberbullying detection. *Information Processing Management*, 58(4):102600.

Alessandra Teresa Cignarella, Valerio Basile, Manuela Sanguinetti, Cristina Bosco, Paolo Rosso, and Farah Benamara. 2020a. Multilingual irony detection with dependency syntax and neural models. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 1346–1358, Barcelona, Spain (Online). International Committee on Computational Linguistics.

Alessandra Teresa Cignarella, Simona Frenda, Valerio Basile, Cristina Bosco, Viviana Patti, Paolo Rosso, et al. 2018. Overview of the evalita 2018 task on irony detection in italian tweets (ironita). In *Sixth Evaluation Campaign of Natural Language Processing and Speech Tools for Italian (EVALITA 2018)*, volume 2263, pages 1–6. CEUR-WS.

Alessandra Teresa Cignarella, Manuela Sanguinetti, Cristina Bosco, and Paolo Rosso. 2020b. Marking

irony activators in a Universal Dependencies treebank: The case of an Italian Twitter corpus. In *Proceedings of the 12th Language Resources and Evaluation Conference*, pages 5098–5105, Marseille, France. European Language Resources Association.

Andrea Cimino, Lorenzo De Mattei, and Felice Dell'Orletta. 2018. Multi-task learning in deep neural networks at evalita 2018. In *EVALITA@CLiC-it*.

Herbert L Colston. 2017. Irony and sarcasm. In *The Routledge handbook of language and humor*, pages 234–249. Routledge.

Ulisses Brisolara Corrêa, Leonardo Coelho, Leonardo Santos, and Larissa A. de Freitas. 2021. Overview of the IDPT task on irony detection in portuguese at iberlef 2021. *Proces. del Leng. Natural*, 67:269–276.

María del Pilar Salas-Zárate, Giner Alor-Hernández, José Luis Sánchez-Cervantes, Mario Andrés Paredes-Valverde, Jorge Luis García-Alcaraz, and Rafael Valencia-García. 2020. Review of english literature on figurative language applied to social networks. *Knowledge and Information Systems*, 62(6):2105–2137.

Poorav Desai, Tanmoy Chakraborty, and Md Shad Akhtar. 2021. Nice perfume. how long did you marinate in it? multimodal sarcasm explanation.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, Minneapolis, Minnesota. Association for Computational Linguistics.

Shelly Dews and Ellen Winner. 1995. Muting the meaning a social function of irony. *Metaphor and Symbol*, 10(1):3–19.

Elena Filatova. 2012. Irony and sarcasm: Corpus generation and analysis using crowdsourcing. In *Proceedings of the Eighth International Conference on Language Resources and Evaluation (LREC'12)*, pages 392–398, Istanbul, Turkey. European Language Resources Association (ELRA).

Ruth Filik, Alexandra Ţurcan, Christina Ralph-Nearman, and Alain Pitiot. 2019. What is the difference between irony and sarcasm? an fmri study. *cortex*, 115:112–122.

Simona Frenda. 2016. Computational rule-based model for irony detection in italian tweets. In *EVALITA 2016*, volume 1749, pages 1–6.

Bilal Ghanem, Jihen Karoui, Farah Benamara, Véronique Moriceau, and Paolo Rosso. 2019. Idat at fire2019: Overview of the track on irony detection in arabic tweets. In *Proceedings of the 11th Forum for Information Retrieval Evaluation*, pages 10–13.

Debanjan Ghosh and Smaranda Muresan. 2018. " with 1 follower i must be awesome: P." exploring the role of irony markers in irony recognition. In *Twelfth International AAAI Conference on Web and Social Media*.

Debanjan Ghosh, Elena Musi, Kartikeya Upasani, and Smaranda Muresan. 2020. Interpreting verbal irony: Linguistic strategies and the connection to the type of semantic incongruity.

Raymond W Gibbs. 2000. Irony in talk among friends. *Metaphor and symbol*, 15(1-2):5–27.

Rachel Giora. 1995. On irony and negation. *Discourse processes*, 19(2):239–264.

Melanie Glenwright and Penny M Pexman. 2010. Development of children's ability to distinguish sarcasm and verbal irony. *Journal of Child Language*, 37(2):429–451.

José-Ángel González, Lluís F. Hurtado, and Ferran Plà. 2019. Elirf-upv at irosva: Transformer encoders for spanish irony detection. In *IberLEF@SEPLN*.

Herbert P Grice. 1975. Logic and conversation. In *Speech acts*, pages 41–58. Brill.

John Haiman et al. 1998. *Talk is cheap: Sarcasm, alienation, and the evolution of language*. Oxford University Press on Demand.

Mark Heitmann, Christian Siebert, Jochen Hartmann, and Christina Schamp. 2020. More than a feeling: Benchmarks for sentiment analysis accuracy. *Available at SSRN 3489963*.

Suzana Ilić, Edison Marrese-Taylor, Jorge Balazs, and Yutaka Matsuo. 2018. Deep contextualized word representations for detecting sarcasm and irony. In *Proceedings of the 9th Workshop on Computational Approaches to Subjectivity, Sentiment and Social Media Analysis*, pages 2–7, Brussels, Belgium. Association for Computational Linguistics.

Aditya Joshi, Pranav Goel, Pushpak Bhattacharyya, and Mark Carman. 2018. Sarcasm target identification: Dataset and an introductory approach. In *Proceedings of the Eleventh International Conference on Language Resources and Evaluation (LREC 2018)*, Miyazaki, Japan. European Language Resources Association (ELRA).

Aditya Joshi, Vinita Sharma, and Pushpak Bhattacharyya. 2015. Harnessing context incongruity for sarcasm detection. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 2: Short Papers)*, pages 757–762, Beijing, China. Association for Computational Linguistics.

Aditya Joshi, Vaibhav Tripathi, Pushpak Bhattacharyya, Mark Carman, Meghna Singh, Jaya Saraswati, and

Rajita Shukla. 2016. How challenging is sarcasm versus irony classification?: A study with a dataset from English literature. In *Proceedings of the Australasian Language Technology Association Workshop 2016*, pages 123–127, Melbourne, Australia.

Jihen Karoui, Farah Benamara, Véronique Moriceau, Viviana Patti, Cristina Bosco, and Nathalie Aussenac-Gilles. 2017. Exploring the impact of pragmatic phenomena on irony detection in tweets: A multilingual corpus study. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 1, Long Papers*, pages 262–272, Valencia, Spain. Association for Computational Linguistics.

Shivani Kumar, Atharva Kulkarni, Md Shad Akhtar, and Tanmoy Chakraborty. 2022. When did you become so smart, oh wise one?! sarcasm explanation in multi-modal multi-party dialogues.

Sachi Kumon-Nakamura, Sam Glucksberg, and Mary Brown. 1995. How about another piece of pie: The allusional pretense theory of discourse irony. *Journal of Experimental Psychology: General*, 124(1):3.

Anna Xenia Laszlo. 2017. *A corpus-based textual analysis of irony and sarcasm in scripted discourse*. Ph.D. thesis, Hong Kong Polytechnic University.

Christopher J Lee and Albert N Katz. 1998. The differential role of ridicule in sarcasm and irony. *Metaphor and symbol*, 13(1):1–15.

John S Leggitt and Raymond W Gibbs. 2000. Emotional reactions to verbal irony. *Discourse processes*, 29(1):1–24.

Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Veselin Stoyanov, and Luke Zettlemoyer. 2020. BART: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7871–7880, Online. Association for Computational Linguistics.

An-Ran Li and Chu-Ren Huang. 2020. A method of modern chinese irony detection. In *From Minimal Contrast to Meaning Construct*, pages 273–288. Springer.

Anran Li. 2021. *A corpus-based research of Chinese verbal irony*. Ph.D. thesis, Hong Kong Polytechnic University.

David C Littman and Jacob L Mey. 1991. The nature of irony: Toward a computational model of irony. *Journal of Pragmatics*, 15(2):131–151.

Emmy Liu, Chen Cui, Kenneth Zheng, and Graham Neubig. 2022. Testing the ability of language models to interpret figurative language.

Yaochen Liu, Yazhou Zhang, Qiuchi Li, Benyou Wang, and Dawei Song. 2021. What does your smile mean? jointly detecting multi-modal sarcasm and sentiment using quantum probability. In *Findings of the Association for Computational Linguistics: EMNLP 2021*, pages 871–880, Punta Cana, Dominican Republic. Association for Computational Linguistics.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Roberta: A robustly optimized bert pretraining approach.

Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. 2013. Distributed representations of words and phrases and their compositionality. In *Advances in Neural Information Processing Systems*, volume 26. Curran Associates, Inc.

Abhijit Mishra, Tarun Tater, and Karthik Sankaranarayanan. 2019. A modular architecture for unsupervised sarcasm generation. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 6144–6154, Hong Kong, China. Association for Computational Linguistics.

Debora Nozza, Elisabetta Fersini, and Enza Messina. 2016. Unsupervised irony detection: A probabilistic model with word embeddings. In *KDIR*.

Silviu Oprea and Walid Magdy. 2019. Exploring author context for detecting intended vs perceived sarcasm. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 2854–2859, Florence, Italy. Association for Computational Linguistics.

Silviu Oprea and Walid Magdy. 2020. iSarcasm: A dataset of intended sarcasm. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 1279–1289, Online. Association for Computational Linguistics.

Reynier Ortega-Bueno, Francisco Rangel, D Hernández Farıas, Paolo Rosso, Manuel Montes-y Gómez, and José E Medina Pagola. 2019. Overview of the task on irony detection in spanish variants. In *Proceedings of the Iberian languages evaluation forum (IberLEF 2019), co-located with 34th conference of the Spanish Society for natural language processing (SEPLN 2019). CEUR-WS. org*, volume 2421, pages 229–256.

Hongliang Pan, Zheng Lin, Peng Fu, Yatao Qi, and Weiping Wang. 2020. Modeling intra and inter-modality incongruity for multi-modal sarcasm detection. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 1383–1392, Online. Association for Computational Linguistics.

Jasabanta Patro, Srijan Bansal, and Animesh Mukherjee. 2019. A deep-learning framework to detect sarcasm

targets. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 6336–6342, Hong Kong, China. Association for Computational Linguistics.

Matthew E. Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer. 2018. Deep contextualized word representations. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 2227–2237, New Orleans, Louisiana. Association for Computational Linguistics.

Rolandos Alexandros Potamias, Georgios Siolas, and Andreas Georgios Stafylopatis. 2020. A transformer-based approach to irony and sarcasm detection. *Neural Computing and Applications*, 32(23):17309–17320.

Antonio Reyes and Paolo Rosso. 2012. Making objective decisions from subjective data: Detecting irony in customer reviews. *Decision support systems*, 53(4):754–760.

Antonio Reyes, Paolo Rosso, and Davide Buscaldi. 2012. From humor recognition to irony detection: The figurative language of social media. *Data & Knowledge Engineering*, 74:1–12.

Antonio Reyes, Paolo Rosso, and Tony Veale. 2013. A multidimensional approach for detecting irony in twitter. *Language resources and evaluation*, 47(1):239–268.

Hannes Ritschel, Ilhan Aslan, David Sedlbauer, and Elisabeth André. 2019. Irony man: Augmenting a social robot with the ability to use irony in multimodal communication with humans. In *AAMAS*.

Omid Rohanian, Shiva Taslimipoor, Richard Evans, and Ruslan Mitkov. 2018. WLV at SemEval-2018 task 3: Dissecting tweets in search of irony. In *Proceedings of The 12th International Workshop on Semantic Evaluation*, pages 553–559, New Orleans, Louisiana. Association for Computational Linguistics.

Andrea Santilli, Danilo Croce, and Roberto Basili. 2018. A kernel-based approach for irony and sarcasm detection in italian. In *EVALITA@CLiC-it*.

Rossano Schifanella, Paloma de Juan, Joel Tetreault, and Liangliang Cao. 2016. Detecting sarcasm in multimodal social platforms. In *Proceedings of the 24th ACM international conference on Multimedia*, pages 1136–1145.

Ken-ichi Seto. 1998. On non-echoic irony. *Relevance theory: Applications and implications*, pages 239–255.

Boaz Shmueli, Lun-Wei Ku, and Soumya Ray. 2020. Reactive Supervision: A New Method for Collecting Sarcasm Data. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 2553–2559, Online. Association for Computational Linguistics.

Pavel Skrelin, Uliana Kochetkova, Vera Evdokimova, and Daria Novoselova. 2020. Can we detect irony in speech using phonetic characteristics only?–looking for a methodology of analysis. In *International Conference on Speech and Computer*, pages 544–553. Springer.

Dan Sperber and Deirdre Wilson. 1986. *Relevance: Communication and cognition*, volume 142. Citeseer.

Emilio Sulis, Delia Irazú Hernández Farías, Paolo Rosso, Viviana Patti, and Giancarlo Ruffo. 2016. Figurative messages and affect in twitter: Differences between irony, sarcasm and not. *Knowledge-Based Systems*, 108:132–143. New Avenues in Knowledge Bases for Natural Language Processing.

Ilya Sutskever, Oriol Vinyals, and Quoc V. Le. 2014. Sequence to sequence learning with neural networks. In *NIPS*.

Martin Sykora, Suzanne Elayan, and Thomas W Jackson. 2020. A qualitative analysis of sarcasm, irony and related #hashtags on twitter. *Big Data & Society*, 7(2):2053951720972735.

Yi-jie Tang and Hsin-Hsi Chen. 2014. Chinese irony corpus construction and ironic structure analysis. In *Proceedings of COLING 2014, the 25th International Conference on Computational Linguistics: Technical Papers*, pages 1269–1278.

Deborah Tannen et al. 2005. *Conversational style: Analyzing talk among friends*. Oxford University Press.

Maggie Toplak and Albert N Katz. 2000. On the uses of sarcastic irony. *Journal of pragmatics*, 32(10):1467–1488.

Cynthia Van Hee, Els Lefever, and Véronique Hoste. 2016. Exploring the realization of irony in Twitter data. In *Proceedings of the Tenth International Conference on Language Resources and Evaluation (LREC'16)*, pages 1794–1799, Portorož, Slovenia. European Language Resources Association (ELRA).

Cynthia Van Hee, Els Lefever, and Véronique Hoste. 2018a. Semeval-2018 task 3: Irony detection in english tweets. In *Proceedings of The 12th International Workshop on Semantic Evaluation*, pages 39–50.

Cynthia Van Hee, Els Lefever, and Véronique Hoste. 2018b. We usually don't like going to the dentist: Using common sense to detect irony on Twitter. *Computational Linguistics*, 44(4):793–832.

Deepanshu Vijay, Aditya Bohra, Vinay Singh, Syed Sarfaraz Akhtar, and Manish Shrivastava. 2018. A dataset for detecting irony in hindi-english code-mixed social media text. *EMSASW@ ESWC*, 2111:38–46.

Rob Voigt and Dan Jurafsky. 2012. Towards a literary machine translation: The role of referential cohesion. In *Proceedings of the NAACL-HLT 2012 Workshop on Computational Linguistics for Literature*, pages 18–25, Montréal, Canada. Association for Computational Linguistics.

Byron C Wallace. 2015. Computational irony: A survey and new perspectives. *Artificial intelligence review*, 43(4):467–483.

Byron C. Wallace, Do Kook Choe, Laura Kertz, and Eugene Charniak. 2014. Humans require context to infer ironic intent (so computers probably do, too). In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 512–516, Baltimore, Maryland. Association for Computational Linguistics.

Deirdre Wilson. 2006. The pragmatics of verbal irony: Echo or pretence? *Lingua*, 116(10):1722–1743.

Deirdre Wilson and Dan Sperber. 2012. Explaining irony. *Meaning and relevance*, pages 123–145.

Chuhan Wu, Fangzhao Wu, Sixing Wu, Junxin Liu, Zhigang Yuan, and Yongfeng Huang. 2018. THU_NGN at SemEval-2018 task 3: Tweet irony detection with densely connected LSTM and multi-task learning. In *Proceedings of The 12th International Workshop on Semantic Evaluation*, pages 51–56, New Orleans, Louisiana. Association for Computational Linguistics.

Rong Xiang, Xuefeng Gao, Yunfei Long, Anran Li, Emmanuele Chersoni, Qin Lu, and Chu-Ren Huang. 2020. Ciron: a new benchmark dataset for Chinese irony detection. In *Proceedings of the 12th Language Resources and Evaluation Conference*, pages 5714–5720, Marseille, France. European Language Resources Association.

Shiwei Zhang, Xiuzhen Zhang, Jeffrey Chan, and Paolo Rosso. 2019. Irony detection via sentiment-based transfer learning. *Information Processing Management*, 56(5):1633–1644.

Mengdi Zhu, Zhiwei Yu, and Xiaojun Wan. 2019. A neural approach to irony generation.

Towards Identifying Alternative-Lexicalization Signals of Discourse Relations

René Knaebel and **Manfred Stede**
Applied Computational Linguistics
Department of Linguistics
University of Potsdam
Germany
{rene.knaebel,manfred.stede}@uni-potsdam.de

Abstract

The task of shallow discourse parsing in the Penn Discourse Treebank (PDTB) framework has traditionally been restricted to identifying those relations that are signaled by a discourse connective ("explicit") and those that have no signal at all ("implicit"). The third type, the more flexible group of "AltLex" realizations has been neglected because of its small amount of occurrences in the PDTB2 corpus. Their number has grown significantly in the recent PDTB3, and in this paper, we present the first approaches for recognizing these "alternative lexicalizations". We compare the performance of a pattern-based approach and a sequence labeling model, add an experiment on the pre-classification of candidate sentences, and provide an initial qualitative analysis of the error cases made by both models.

1 Introduction

The view that discourse relations serve to model central aspects of the coherence of a text is widely accepted, and several approaches with different theoretical commitments have been developed (e.g. Mann and Thompson, 1988; Prasad et al., 2008a; Lascarides and Asher, 2007; Sanders et al., 1992). Our work is situated in the framework of Shallow Discourse Parsing and thus grounded in the Penn Discourse Treebank (PDTB) corpus (Prasad et al., 2008a). Here, some distinctions are commonly being made regarding the surface realization of discourse relations; most importantly:

1. A relation can be signalled by a *connective*, i.e., a lexical item from a closed class (conjunctions, certain adverbials).

2. A relation can be signalled by a different lexical form, which the PDTB calls *Alternative lexicalization* or *AltLex* for short.

3. A relation can also be stated without any lexical signal; in this case it is called *implicit*.

In the PDTB corpus, (1) and (3) are by far the most frequent cases, and accordingly, they have received much attention in shallow discourse parsing. As for (2), Lin et al. (2014) had developed a first, relatively simple approach; to our knowledge there have not been any follow-up proposals (including all the parsers presented in the CoNLL shared tasks in 2015 and 2016 (Xue et al., 2015, 2016) and in the recent DISRPT tasks (Zeldes et al., 2019, 2021)).

With the introduction of the PDTB corpus version 3.0 (Prasad et al., 2018), amongst some other changes, the number of annotated AltLex instances has grown from 624 (in version 2.0) to 1632. With the corpus now being considerably richer in AltLex signals, we believe that their role in shallow discourse parsing now needs to be strengthened. Besides, also for advancing the theoretical description of the AltLex category and its role in coherence marking, it is important to perform empirical studies.

Essentially, alternatively–lexicalized discourse relations are signalled by an open set of phrases that verbalizes the connection between two discourse arguments. In the PDTB, they are being annotated when no connective is present, and a "connective insertion" test yields an impression of redundancy; in this case annotators are asked to mark the text span that already signals the relation. Previous work (Prasad et al., 2010; Danlos, 2018; Rysová and Rysová, 2018) studied the general form of AltLex expressions and tried to find and formalize patterns that can complement the well-established idea of a fixed list of discourse connectives (e.g. Das et al., 2018).

Example (1) (Danlos, 2018) illustrates the interchangeability of explicit connectives and AltLex signals withing the same context, as we could simply substitute the connective *Therefore* by alternative more complex lexicalizations such as *This caused* and *Because of this* while still preserving the meaning of the relation:

Proceedings of the 29th International Conference on Computational Linguistics, pages 837–850
October 12–17, 2022.

(1)
1. Fred didn't stop joking. **Therefore**, his friends enjoyed hilarity throughout the evening.
2. ... **This caused**, his friends enjoyed hilarity throughout the evening.
3. ... **Because of this**, his friends enjoyed hilarity throughout the evening.

Contrary to this, in Example (2) (Prasad et al., 2010) the AltLex signal is not easily substitutable by a simpler explicit connective as, with a replacement such as *because*, this sentence would lose information about the reason's importance:

(2) But a strong level of investor withdrawals is much more unlikely this time around, fund managers said. **A major reason** is that investors already have sharply scaled back their purchases of stock funds since Black Monday.

In this paper, we aim to overcome the negligence of AltLex relations in shallow discourse parsing by proposing two different technical approaches. Both tackle the problem without relying on any external lexical resources. Specifically, our contributions are: (i) We present the first approach to automatically classifying AltLex instances in the PDTB3 corpus; (ii) we compare a simple pattern-based approach to a neural model; (iii) we experiment with pre-classification of AltLex-relevant sentences (for dealing with class imbalance in the corpus); (iv) we provide initial observations on types of errors made by the models.

Section 2 discusses related work, and Section 3 briefly describes relevant aspects of the PDTB corpus. Section 4 introduces our various methods. We present results in Section 5, discuss them in Section 6, and finally conclude.

2 Related Work

We highlight two different directions that are relevant for our work. The first part shows work related to alternative lexicalized phrases. The second one looks into recent applications for connective identification, whose methods are similar to one of our approaches.

Alternative Lexicalized Phrases (AltLex). After the introduction of alternative lexicalized phrases, among others, in the PDTB (Prasad et al., 2008a), a subsequent work by Prasad et al. (2010) presented more details and analysed regularities by defining groups of phrases. Based on their discoveries, a first attempt was made to extract AltLex relations while analysing implicit relations (Lin et al., 2014). Their approach is evaluated in combination with all non-explicit relations. Thus, the specific AltLex results of their approach are unfortunately not available for comparison. Due to the revision of the PDTB, the definition of discourse signals is made more flexible, and Lin et al.'s approach is not applicable to the current problem anymore (Prasad et al., 2018).

Attempts of building a Czech Discourse Treebank have also shown, how challenging the annotation process is. In contrast to the PDTB, the Czech Discourse Treebank distinguishes discourse signals into three groups, namely *primary connectives*, *secondary connectives*, and *free connective phrases*. Low inter-annotator agreement was particularly observed for the annotation of free connecting phrases (comparable to alternative lexicalizations) due to the complexity of the task (Rysová, 2012). Based on this tree bank, a template approach is proposed to manually build a lexicon for secondary connectives, analogously to that for primary ones (Danlos, 2018; Rysová and Rysová, 2018). One of Danlos's single lexicon entry, for example, to recognize the phrases `for this/a given reason`, would describe the lexical head `N` of this phrase `reason` in different possible environments (so-called schemes) with a rule like `for [Ana-Det (Adj)/Ana-Adj] N`. Furthermore, Danlos (2018) concedes that, because "free connective phrases are compositional and include at least two content words", this lexicon-based approach is not applicable.

Dunietz et al. (2017) adapt the PDTB annotation scheme and present another corpus that entirely focuses on causal relations. They do not distinguish between explicit connectives and alternative lexicalizations as done in the PDTB. Comparing their annotated signals with the second version of PDTB, they discover an overlap of 8.9 % with the PDTB connective signals. Further, they introduce a feature-based system for tagging causal relations between individual events. In contrast to them, we avoid linguistic features by using representations from pretrained language models.

In contrast to PDTB conform schemes, RST (Mann and Thompson, 1988) contains only information about discourse segments, their relations and nuclearity. The work of Das and Taboada

(2018) complements the absence of this information by annotating a subpart of the RST-DT corpus with all kinds of signals point toward a discourse relation. Their first finding reveals that discourse relations may be signalled by other discourse elements than lexical phrases, *inter alia*, syntactic structure (e.g. relative clauses, reported speech), semantics (e.g. synonymy, repetition, lexical chain), text genre (e.g. inverted pyramid scheme, newspaper layout). Another interesting finding consists in the presence of multiple signals that point to the same relation, e.g. semantic+syntactic, reference+syntactic, and others. Further, recent work of Zeldes and Liu (2020) proposed an interesting approach for inferring discourse signals from the given relation senses. However, this approach does not take alternative lexicalizations into account yet.

Explicit Connective Disambiguation. Some of the approaches that were proposed to this task are relevant for AltLex identification as well. Pitler and Nenkova (2009) presented a simple feature-based model for the disambiguation (discourse versus sentential usage) of connective candidates extracted by matching entries of a connective lexicon. With their best feature combination, they achieve an F1 score of 94.19 on the test set. Recently, Knaebel and Stede (2020) adapt the original idea and replace hand-crafted features by various types of word embeddings. Their state-of-the-art model uses contextualized word embeddings and achieves 97.45 F1 score. In our work, we extend their approach to alternative lexicalized phrases, tackling the problem without external candidate lexicon.

Furthermore, another line of research started with the focus to avoid lexicon-based solutions. Recently, several promising sequence labeling approaches (e.g. Yu et al., 2019; Muller et al., 2019; Bakshi and Sharma, 2021; Kamaladdini Ezzabady et al., 2021) have been proposed using standard and contextualized word embeddings. Among these, Yu et al. (2019) achieves best scores (92.02 F1 score) in extracting connectives without lexicon. They develop a model that combines linguistic information (e.g. part-of-speech tags, dependency relation, sentence length, *inter alia*) with recent advances in contextualized word representations. In contrast, the work of Bakshi and Sharma (2021) achieve slightly worse results (91.15 F1 score) but with a completely feature-free approach.

3 Penn Discourse Treebank

The recent version (v3) of the Penn Discourse Treebank (PDTB, Prasad et al. 2018) is the largest available resource of lexically grounded discourse relations which include both explicitly signalled relations and implicit relations. It describes discourse relations to consist of exactly two arguments with an optional marker to signal the relation. In addition, one or more *senses* are attached to each relation to describe its meaning. For example, two senses often correlate are *Temporal.Synchronous* and *Comparison.Contrast* for the explicit connective *while*.

In its previous version, Prasad et al. (2008b) start the annotation process by the identification of connectives (defined by a fixed set of candidates). Then, adjacent sentences without explicit relation are examined according to whether there holds an implicit relation, and, in addition, a connective is searched that fits in between the relations' arguments. If the insertion of any connective leads to redundancy, the lexical signal already part of the relation is used instead—the AltLex. In their studies on alternative lexicalizations, Prasad et al. (2010) use the two properties of syntactical and lexical flexibility to sort these signals into three groups. Hereby they demonstrated, that most of the AltLexes belong to the syntactically and lexically free group with 76.6%.

As a consequence of the recent update, the definition of discourse signals has undergone some changes. The set of explicit connectives is expanded (which indirectly changes the set of AltLexs too) and the position of the connective in relation to its arguments is more relaxed compared to the previous version. In addition, a few changes have been made in the process of identifying AltLex relations in general. As a result of the introduction of intra-sentential AltLex relations (about 900), arbitrary expressions are allowed, also including adjectives and adjective modifiers. Also, annotators are allowed to mentally add anaphoric references, (e.g. next [to this], further [to that]), which leads to potential overlap with explicit connectives. Signals are not syntactically bound to the second argument of the relation anymore, but possibly combine parts of both arguments. A new sub-class was introduced for lexico-syntactic constructions, the so-called AltLexC. In total, almost 1000 AltLex relations, including their signals, were added during the revision. In our work, we do

Property	PDTB2	PDTB3
count	624	1632
signal length	3.26 (2.08)	2.62 (2.45)
sentence length	22.65 (10.35)	27.68 (10.94)
signal position	1.42 (3.22)	9.37(10.26)

Table 1: Differences of AltLex relations between both versions of the Penn Discourse Treebank. Properties length and position show average values with standard deviation enclosed in parentheses.

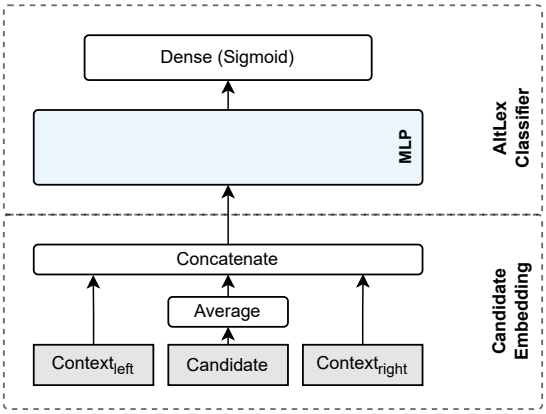

Figure 1: Candidate-based classification approach. All token embeddings directly associated with AltLex candidate are averaged and concatenated with surrounding context tokens. Simple MLP module on top with final classification layer to predict discourse signal.

not further distinguish both types of alternative lexicalized relations and henceforth refer to these simply as AltLex relations. Discourse signals appear in both continuous (e.g. *since then, after that*) and discontinuous forms (e.g. *the aim ... is*, *the more ... the more*, ...). As the complexity highly increases if POS patterns would also cover gaps within, we restrict our work to continuous signals only, and thus eliminated 62 instances (3.79 %) from the full set. Explicit discourse connectives are ignored for all our experiments, even if they should occur in the same sentence of an AltLex signal.

The PDTB consists of 2,160 documents with a total number of 50,945 sentences. We briefly summarize the differences of both versions of the PDTB in Table 1 to illustrate the motivation of introducing our new approaches. As already mentioned, the number of available AltLex relations is almost tripled from 624 to 1632 instances. The average length of the signals decreases slightly from 3.26 tokens to 2.62 token, and the average length of sentences containing the signals increases by a few tokens on average (22.65 up to 27.68). A particular challenging aspect of the new PDTB version is the more flexible positioning of signals, which renders previous simple identification approaches as no longer feasible.

4 Method

Input to all models are prepared context sensitive word embeddings. For extracting token-wise context sensitive embeddings, we follow the suggestion of Devlin et al. (2019). Given a sentence, tokens are processed by the WordPiece tokenizer (Wu et al., 2016) which possibly leads to a higher number of subtokens. These are processed for generating corresponding hidden states on subtoken-level. We choose RoBERTa (Liu et al., 2019) as it performs best on connective disambiguation compared with other BERT variations (Knaebel and Stede, 2020). We average multiple subtoken outputs into a single output that corresponds to the full token. Following the suggestion of Devlin et al. (2019), the last four hidden layers are concatenated and thus form the final token embeddings that serve as input for the subsequent models.

4.1 Pattern-based Candidate Extraction

Traditional approaches for connective disambiguation integrate a connective lexicon that is used to extract possible candidates, before using a system to disambiguate discourse readings from their sentential counterparts (Pitler and Nenkova, 2009; Lin et al., 2014). Inspired by this approach, we devised a different pattern-based extraction procedure for finding possible AltLex candidates. Specifically, we generate possible AltLex candidates by pattern-matching via a list of extracted part-of-speech (POS) sequences. From the AltLex relations available in PDTB3, we extract 408 unique POS patterns. They range from very frequent single tags such as **VBG** (n=468) and **RB** (n=113) to longer and less frequent sequences such as **VBD DT NN IN** (n=9). This sequence, for example, is extracted from the signals *attributed the increase/improvement to*, but it also matches other phrases such as *visited a lot of* and *signed a contract with*. The number of extracted candidates per

pattern is very high in comparison with the pattern occurrence itself. For the simple single tag rule **VBG**, for example, we find 1869 instances in one of our randomly sampled test split.

We extend this basic pattern approach by introducing a small context window of additional tokens to the left and right of the candidate. Our hypothesis is that *context sensitivity* is not only beneficial for the final classification of the embeddings, but already useful during the pattern extraction. Information such as start and end of sentence, but also punctuation is useful. As the number of patterns increases tremendously with each additional surrounding word, and the generalization at the same time decreases by patterns that are too specific, we limit our experiments to only one tag on the left and on the right of the original pattern. By doing so, the number of extracted patterns increases to 800 in total. For example, instead of one most frequent single-tag pattern **VBG**, we now have more specific patterns that occur less often **, VBG DT** (n=162), **, VBG NNS** (n=39), **, VBG PRP** (n=35), **, VBG NN** (n=31), among others. In comparison to the simple approach, the context-sensitive approach reduces the number of extracted candidates by a large margin, in particular for shorter tag patterns. For example, the number of occurrences for **VBG** is decreased by about 70 % to 516 instances. A list of the 30 most frequent tag sequences for both approaches is provided in the appendix Table 4, and Table 5 provides numbers on the extracted candidates per approach.

We use the collected tag lists, iterate over all sentences, and extract any phrase that matches one of the POS patterns as possible AltLex candidate. After generating these candidates, we follow the approach of Knaebel and Stede (2020) for connective disambiguation with contextualized embeddings, shown in Figure 1. In their experiments, they outperform previous approaches with a simple multilayered-perceptron architecture on top of contextualized embeddings. We refer to this architecture as *MLP* module, which consists of two fully connected layers with a dropout layer following each.

The first ("simple") approach is henceforth referred to as **exact** approach, while we call the second one **context-sensitive**. For the experiments, we also specify the **context size**, that is the number of tokens surrounding the candidate on each side; e.g., the value 0 refers to no context at all, while the value 2 indicates a context of two tokens to the left and right, which sums up to five embeddings (four context embeddings plus one embedding for the averaged candidate tokens).

4.2 Sentence Labeling

The limited variability of observed patterns in the dataset is the major disadvantage of the pattern-based approach. We aim to overcome this problem by introducing a **sequence labeling** approach (see Figure 2) based on contextualized embeddings for recognizing alternative lexicalizations. In the PDTB3, AltLex signals always occur within a single sentence, and thus our approach is designed for sentence-level processing.

The sentence processing part consists of two bidirectional LSTM (Hochreiter and Schmidhuber, 1997) layers. After each layer we add a dropout layer for better generalization. We will refer to this as *BiLSTM* module. The hidden states are further individually processed by an MLP module. Finally, we use a conditional random field (Lafferty et al., 2001) for the output prediction (compare Figure 2 output (I) Sentence Labeling). As output, we use a binary label that represents the AltLex class membership.

Deciding whether a phrase should be identified as AltLex is often dependent on its context. As argument spans of AltLex relations often include previous sentences, we hypothesize that additional processing of the preceding sentence has a positive effect on the prediction quality of our model. For this reason, we propose an additional sentence processing step (see Figure 2 **Context Processing** on the left side) in which we use the final hidden states of the BiLSTM module as the initial states for the BiLSTM modules in the sentence processing part.

We refer to the architecture without previous sentence context as **single**, and we use **context** to point to the option with previous sentence processing.

4.3 Sentence Classification

The extraction of alternative lexicalized discourse signals is especially hard with respect to the small amount of signal occurrence. In addition to the heavy imbalance of the labels on token-level, only a minority of all sentences contains an alternative lexicalized signal. Therefore, we hypothesize that, following analogously work related to explicit connectives (Patterson and Kehler, 2013), an additional step of classifying a potential sentence candidate as containing an AltLex relation or not might be

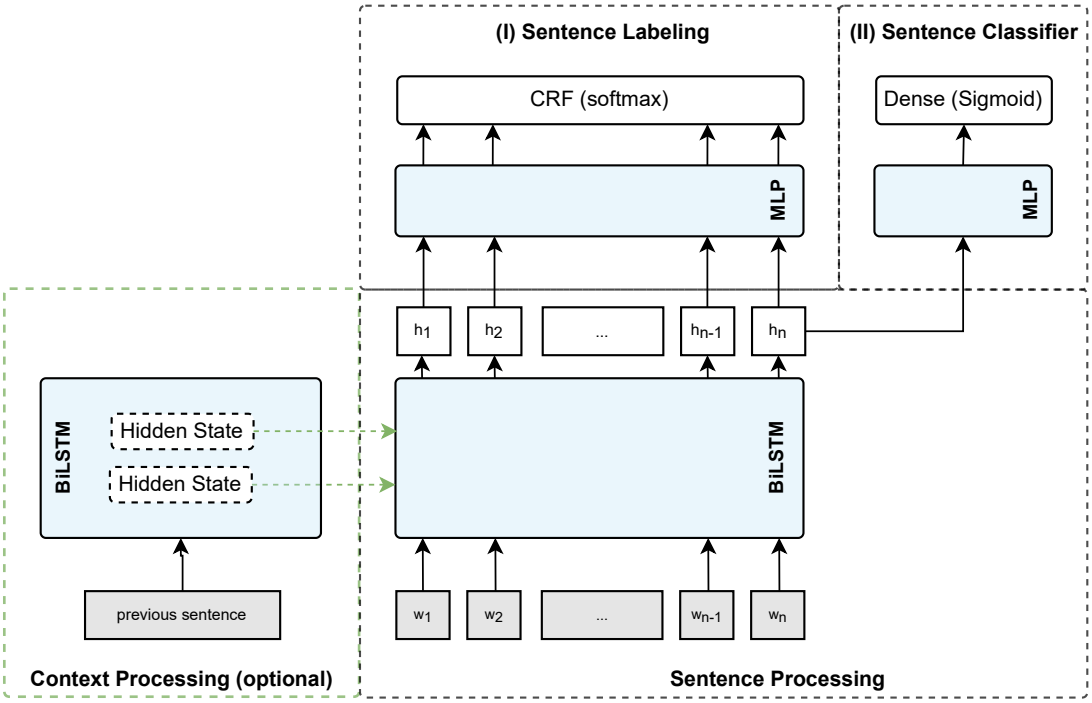

Figure 2: Joint diagram of (I) sequence labeling and (II) potential sentence classification approaches. A given sentence is processed by BiLSTM module. Either all hidden states are processed by MLP module individually and forwarded to CRF layer to predict AltLex tokens in sentence, or only last hidden state is processed by MLP module to predict the presence of AltLex in whole sentence. The optional context processing part contains a similar BiLSTM module but processes the previous sentence. Then, hidden state of the LSTMs are used for initialization.

beneficial to reduce the overall complexity of the sentence labeling problem. We first train a **sentence classification** model on the full dataset, including the majority of negative examples. Thereafter, we train our sentence labeling architectures (as described above) on the positive instances only.

5 Experiments

In our experiments we study various statistical models on version 3 of the Penn Discourse Treebank (Prasad et al., 2018). For all experiments' runs, we randomly split the full dataset into three parts (train, validation, and test), as suggested by Shi and Demberg (2017). We set 10 % of the full dataset aside for testing, then the remaining data is split into 90 % and 10 % for training and validation parts, respectively. The reported final results are averaged over three different runs each. For the precision–recall curves illustrated in Figure 3 we interpolate the individual curve per run and compute the mean curve surrounded by one standard deviation. We use the average precision scores (AP) of the mean curves for comparison.

5.1 Pattern-based Disambiguation

We compare two variants of our pattern-based extraction, **exact** patterns and **context-sensitive** patterns, taking into account one token to the left and right of the pattern. We generate patterns for both variants only once on the whole corpus which we think is most similar to the experiments on connective classification, where a list of possible connective candidates is compiled previously before splitting data. Further, we study the influence of the number of surrounding context embeddings for the model (context width). Through all experiments, we use an up-sampling rate of 5 for positive samples and 0.1 for negative samples. The hidden size of the first layer in the MLP module is 256 and the second layer 64, respectively. We train for at most 20 epochs with a batch size of 64. In addition, we stop earlier if validation loss does not improve over 7 epochs.

Evaluation metrics are calculated with regard to the extracted signal candidates. Please note, that in contrast to the later evaluation (Section 5.3), multiple possibly overlapping candidates might be extracted. Analysing the precision–recall curves in Figure 3a, an increased context width for both

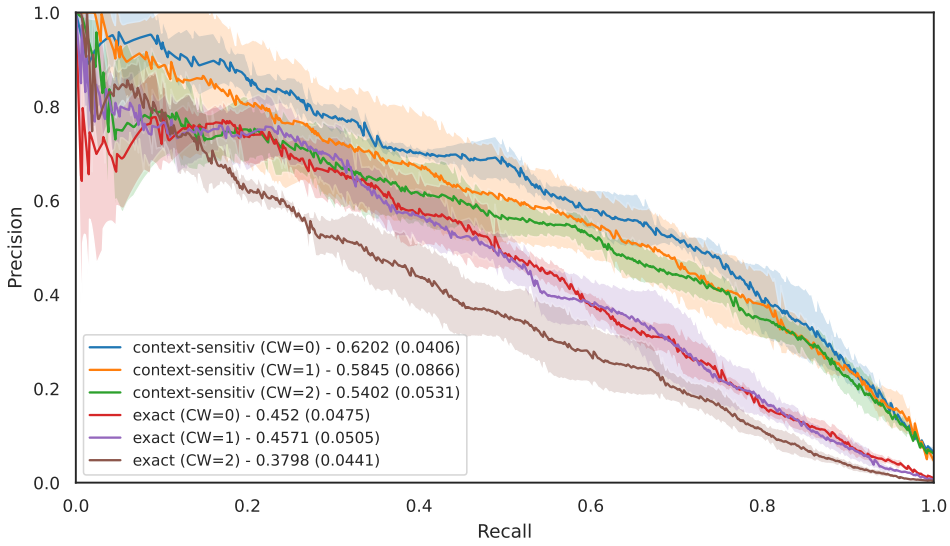

(a) Pattern-based candidate classification. Metrics are based on the confidence that a candidate represents an AltLex signal.

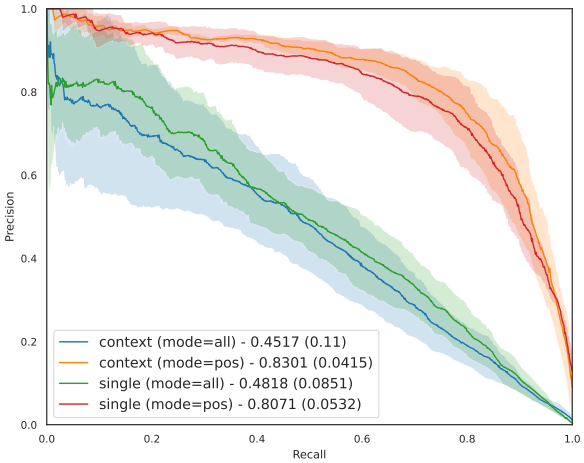

(b) Sentence Labeling approach. Metrics are computed token-wise based on the potentials of the CRF layer before optimal sequence is computed.

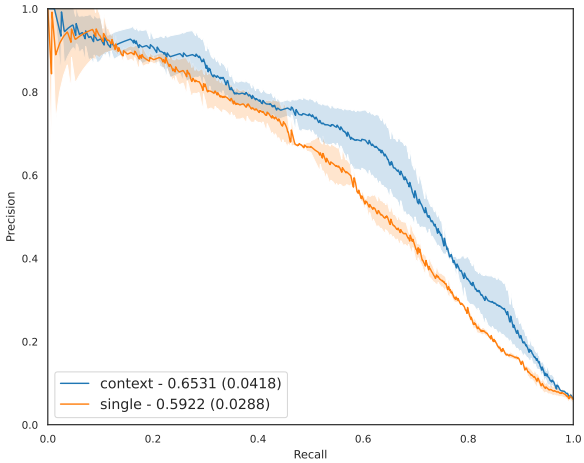

(c) Sentence Classification. Metrics computed with respect to confidence of the model that a sentence contains AltLex signal.

Figure 3: Precision–Recall curves for all trained models separated by tasks. Curves of individual run are averaged surrounded by one standard deviation. Scores represent average precision scores with standard deviation in parenthesis.

pattern-based models seems not beneficial, as the context is already encoded in the individual token embedding. Using context-sensitive patterns for extraction, on the other hand, has a positive influence, as originally assumed. On the one side, the overall number of unique extracted patters roughly doubles due to the higher specification. On the other, the total number of extracted candidates reduces by about 95% on a random test split. Finally, the best performing model in this section (context-sensitive pattern, zero embedding context width) achieves a token-level averaged score of 0.62 precision.

5.2 Sentence Labeling

A sequence-level labeling model is trained to predict the presence of an discourse signal. The model assumes as input contextualized word embeddings for a maximum sentence size of 60 tokens. For single-step prediction, we use a down-sampling rate of 0.5, for two-step prediction we remove negative samples entirely. All modules' layers use 128 units as hidden size. We give all models the chance to train for 50 epochs with a batch size of 32; however, the training stops earlier in all cases, when the validation loss stops improving over 7 epochs.

For the sentence labeling approach, we use

token-wise potentials of the CRF output instead of the decoded sequence labels. This allows us to calculate precision and recall on token-level which is visualized in Figure 3b. We observe that additional context in form of the previous sentence is not beneficial on this level for sentence labeling. With a mean score of 0.45 AP, the performance of the context-sensitive labeling model is slightly below the score of the single sentence labeling model with 0.48 AP (note the low confidence caused by the high standard deviation). The performance increases dramatically, as expected, when labeling is restricted to positive sentences only.

For potential sentence classification, the previous sentence's context increases the average performance slightly (from 0.59 to 0.65 average precision) as indicated by Figure 3c. This is in accordance with the previous observations for the sequence labeling experiment.

5.3 Results

For the final evaluation, we introduce the metrics *overlap*, *partial-rate*, and *full-rate*. The first score indicates the overlap of true signal positions with predicted signal occurrences. The partial-rate is one if there is at least one token of the signal correctly classified. The full-rate is satisfied if and if only the full range of the signal is correctly recognized. The sequential predictions are taken as computed by the final layer, for the candidate-based predictions, we simply choose all as signal classified candidates and set corresponding associated tokens to being a signal. Note that a single token might be classified multiple times. A token is set to be a signal if a single instance prediction exists.

Results of the experiments are presented in Table 2a for the candidate-based experiments and in Table 2b for the labeling and sentence classification experiments. Scores for precision, recall, and F1 with respect to predicting the AltLex class are presented for a 0.5 threshold. Thus, they merely provide a limited view compared to the precision–recall curves.

The best overall performing model is the candidate-based model with context-sensitive pattern and zero embedding context width. It achieves a 74 % phrase overlap, with scores 0.83 and 0.63 for partial-rate and full-rate, respectively. Interestingly, the candidate-based approach outperforms the sequence-labeling approach. Although the F1 scores are higher for this approach, the final evaluation shows that lower performance regarding overlap (0.63), partial-rate (0.67), and full-rate (0.60). The performance increases dramatically, as expected, when labeling is restricted to positive sentences only. The combination of the simple sentence classification (0.64 F1 score) and the simple labeling approach on positive samples (0.84 F1 score) leads to similar results as a model trained on the full data set.

6 Discussion

Both pattern-based candidate approaches (exact and context-sensitive) achieve better final results compared to the labeling approaches which reflects a similar observation as for studies on explicit connective identification (Knaebel and Stede, 2020). However, we have to keep in mind that patterns are extracted on the whole corpus in advance, which makes the lexicon approach for explicit connective identification more similar and, thus, better comparable. The comparison to our sequence labeling approach is somewhat unfair, as we here strictly split the corpus into three parts right at the beginning and these models never have access to all signal variants in the whole corpus.

Compared to the recognition performance of explicit connectives with about 96%, AltLex relations are predicted far less accurately with at most 63% exact match for the pattern-based approach. Also, we expect a drop in performance when limiting the extraction to the training corpus only.

After examining the errors made by the pattern-based mode, we conclude that the length of the errors (unrecognized signals) compared to the correctly recognized patterns is almost the same (2.5 and 2.7 tokens on average each). Thus the length of signals is not a crucial factor for this type of model. Quite a few poorly recognized examples are related to adverbial phrases (e.g., *eventually, further, so far, too*). This type of error seems problematic for both model variants. A possible reason is the high imbalance of adverbial signal instances (about 100), compared to extracted candidates ranging from 1000 to 4000 instances. Regarding verb gerund forms, the pattern-based approach recognizes most of them. This is interesting as it represents the largest group in PDTB with most variations. On the other side, we also recognize cases where signals are marked as false positives, such as *resulting* and *allowing*. Here, more more in-depth elaboration is necessary, to check whether the model is truly wrong or just

model-type	context-size	precision	recall	F1	overlap	partial-rate	full-rate
exact	0	0.29	0.58	0.38	0.68	0.82	0.55
exact	1	0.30	0.53	0.38	0.57	0.72	0.43
exact	2	0.30	0.47	0.36	0.56	0.73	0.42
context-sensitive	0	0.35	**0.72**	**0.47**	**0.74**	**0.83**	**0.63**
context-sensitive	1	**0.36**	0.64	0.46	0.72	0.82	0.60
context-sensitive	2	0.35	0.60	0.44	0.67	0.78	0.56

(a) Results of candidate-based extraction approach.

model-type	input	mode	precision	recall	F1	overlap	partial-rate	full-rate
labeling	single	all	0.53	0.55	**0.53**	**0.63**	**0.67**	**0.60**
labeling	context	all	**0.59**	0.48	**0.53**	0.58	0.60	0.57
labeling	single	positives	0.84	0.85	0.84	0.87	0.90	0.82
labeling	context	positives	0.80	0.79	0.79	0.80	0.85	0.72
sentence	single	all	0.74	0.57	0.64			
sentence	context	all	0.68	0.49	0.56			

(b) Results of the sequence-labeling and sentence classification approaches are computed on token-level.

Table 2: Evaluation results: Scores on the left (precision, recall, F1) with respect to AltLex class. These scores of both tables cannot be compared directly, as prediction level differs (candidates vs. tokens). Final signal extraction on the right evaluates predictions by degree of overlap, and agreement on partial and full prediction.

found new signals.

For the sequential model, it is noticeable that prepositions are missing in the predicted signal e.g. *only to* and *opposed to*. Also, verbs in combination with anaphoric pronouns e.g. *that would leave*, *this creates*, are often recognized as discourse signal although there are not annotated as such.

The sequential models with additional context information in form of the previous sentence result in unexpectedly low performances. Intuitively, because AltLexes often connect parts of two consecutive sentences, we would expect a model's performance to increase if it gets access to more information. We assume poor performances are caused by context representation and therefore we wonder whether models especially designed to serve sentence representations would lead to better results in this experiments.

7 Conclusion

Our work is a successful first attempt to fully automatically (without hand-crafted rules) extract alternative lexicalized discourse relation signals. For this task, we propose two technically different solutions: First, a pattern-based approach working analogously to lexicon-based connective disambiguation approaches, and second, a sequence labeling approach similar to recent connective labeling approaches without external lexicon. We evaluated these models directly on their corresponding training task and, further, provide more details on the actual recognition task.

We wonder, how these two model architectures perform on a different corpus domain such as biomedical data (Prasad et al., 2011) and whether the pattern-based limitation of the first approach is noticeable. Having the new version of the PDTB with about three times as much data as before, it still seems the performance of the sequence-labeling approach is strongly limited by the amount of available data. For future work, it would be interesting to extract different patterns for generating candidates for the first approach. Universal part-of-speech tags would have the advantage of being a little more flexible (less specific word classes) while at the same time, it could be possible to use similar techniques for other languages when the embeddings model is changed to a different language or to a multi-lingual model. The sequential approach has the advantage to be able to find new patterns without observing them in the training data directly. With more raw data from possibly different domains, it would be interesting to apply this technique and examine new/other variants of alternative lexicalizations that do not occur in such form in the original corpus.

8 Acknowledgments

This research has been supported by the German Research Foundation (DFG) with grant number 455911521, project "LARGA" in SPP "RATIO". We would like to thank the anonymous reviewers for their valuable feedback.

References

Sahil Bakshi and Dipti Sharma. 2021. A transformer based approach towards identification of discourse unit segments and connectives. In *Proceedings of the 2nd Shared Task on Discourse Relation Parsing and Treebanking (DISRPT 2021)*, pages 13–21, Punta Cana, Dominican Republic. Association for Computational Linguistics.

Laurence Danlos. 2018. Discourse and lexicons: Lexemes, MWEs, grammatical constructions and compositional word combinations to signal discourse relations. In *Proceedings of the Joint Workshop on Linguistic Annotation, Multiword Expressions and Constructions (LAW-MWE-CxG-2018)*, pages 30–40, Santa Fe, New Mexico, USA. Association for Computational Linguistics.

Debopam Das, Tatjana Scheffler, Peter Bourgonje, and Manfred Stede. 2018. Constructing a lexicon of English discourse connectives. In *Proceedings of the 19th Annual SIGdial Meeting on Discourse and Dialogue*, pages 360–365, Melbourne, Australia. Association for Computational Linguistics.

Debopam Das and Maite Taboada. 2018. RST signalling corpus: a corpus of signals of coherence relations. *Language Resources and Evaluation*, 52:149–184.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, Minneapolis, Minnesota. Association for Computational Linguistics.

Jesse Dunietz, Lori Levin, and Jaime Carbonell. 2017. Automatically tagging constructions of causation and their slot-fillers. *Transactions of the Association for Computational Linguistics*, 5:117–133.

Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long short-term memory. *Neural Computation*, 9:1735–1780.

Morteza Kamaladdini Ezzabady, Philippe Muller, and Chloé Braud. 2021. Multi-lingual discourse segmentation and connective identification: MELODI at disrpt2021. In *Proceedings of the 2nd Shared Task on Discourse Relation Parsing and Treebanking (DISRPT 2021)*, pages 22–32, Punta Cana, Dominican Republic. Association for Computational Linguistics.

René Knaebel and Manfred Stede. 2020. Contextualized embeddings for connective disambiguation in shallow discourse parsing. In *Proceedings of the First Workshop on Computational Approaches to Discourse*, pages 65–75, Online. Association for Computational Linguistics.

John D. Lafferty, Andrew McCallum, and Fernando Pereira. 2001. Conditional random fields: Probabilistic models for segmenting and labeling sequence data. In *ICML*.

Alex Lascarides and Nicholas Asher. 2007. Segmented discourse representation theory: Dynamic semantics with discourse structure. In H. Bunt and R. Muskens, editors, *Computing Meaning: Volume 3*, pages 87–124. Kluwer Academic Publishers.

Ziheng Lin, Hwee Tou Ng, and Min-Yen Kan. 2014. A pdtb-styled end-to-end discourse parser. *Nat. Lang. Eng.*, 20(2):151–184.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Roberta: A robustly optimized bert pretraining approach. *ArXiv*, abs/1907.11692.

William C. Mann and Sandra A. Thompson. 1988. Rhetorical structure theory: Toward a functional theory of text organization. *Text & Talk*, 8:243 – 281.

Philippe Muller, Chloé Braud, and Mathieu Morey. 2019. ToNy: Contextual embeddings for accurate multilingual discourse segmentation of full documents. In *Proceedings of the Workshop on Discourse Relation Parsing and Treebanking 2019*, pages 115–124, Minneapolis, MN. Association for Computational Linguistics.

Gary Patterson and Andrew Kehler. 2013. Predicting the presence of discourse connectives. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*, pages 914–923, Seattle, Washington, USA. Association for Computational Linguistics.

Emily Pitler and Ani Nenkova. 2009. Using syntax to disambiguate explicit discourse connectives in text. In *Proceedings of the ACL-IJCNLP 2009 Conference Short Papers*, pages 13–16, Suntec, Singapore. Association for Computational Linguistics.

Rashmi Prasad, Nikhil Dinesh, Alan Lee, Eleni Miltsakaki, Livio Robaldo, Aravind Joshi, and Bonnie Webber. 2008a. The penn discourse treebank 2.0. In *In Proceedings of LREC*.

Rashmi Prasad, Nikhil Dinesh, Alan Lee, Eleni Miltsakaki, Livio Robaldo, Aravind Joshi, and Bonnie Webber. 2008b. The Penn Discourse TreeBank 2.0. In *Proceedings of the Sixth International Conference*

on Language Resources and Evaluation (LREC'08), Marrakech, Morocco. European Language Resources Association (ELRA).

Rashmi Prasad, Aravind Joshi, and Bonnie Webber. 2010. Realization of discourse relations by other means: Alternative lexicalizations. In *Coling 2010: Posters*, pages 1023–1031, Beijing, China. Coling 2010 Organizing Committee.

Rashmi Prasad, Susan Mcroy, Nadya Frid, Aravind Joshi, and Hong Yu. 2011. The biomedical discourse relation bank. *BMC bioinformatics*, 12:188.

Rashmi Prasad, Bonnie Webber, and Alan Lee. 2018. Discourse annotation in the PDTB: The next generation. In *Proceedings 14th Joint ACL - ISO Workshop on Interoperable Semantic Annotation*, pages 87–97, Santa Fe, New Mexico, USA. Association for Computational Linguistics.

Magdaléna Rysová. 2012. Alternative lexicalizations of discourse connectives in Czech. In *Proceedings of the Eighth International Conference on Language Resources and Evaluation (LREC'12)*, pages 2800–2807, Istanbul, Turkey. European Language Resources Association (ELRA).

Magdaléna Rysová and Kateřina Rysová. 2018. Primary and secondary discourse connectives: Constraints and preferences. *Journal of Pragmatics*, 130:16–32.

Ted Sanders, Wilbert Spooren, and Leo G. M. Noordman. 1992. Toward a taxonomy of coherence relations. *Discourse Processes*, 15:1–35.

Wei Shi and Vera Demberg. 2017. On the need of cross validation for discourse relation classification. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 2, Short Papers*, pages 150–156, Valencia, Spain. Association for Computational Linguistics.

Yonghui Wu, Mike Schuster, Zhifeng Chen, Quoc V. Le, Mohammad Norouzi, Wolfgang Macherey, Maxim Krikun, Yuan Cao, Qin Gao, Klaus Macherey, Jeff Klingner, Apurva Shah, Melvin Johnson, Xiaobing Liu, Lukasz Kaiser, Stephan Gouws, Yoshikiyo Kato, Taku Kudo, Hideto Kazawa, Keith Stevens, George Kurian, Nishant Patil, Wei Wang, Cliff Young, Jason Smith, Jason Riesa, Alex Rudnick, Oriol Vinyals, Greg Corrado, Macduff Hughes, and Jeffrey Dean. 2016. Google's neural machine translation system: Bridging the gap between human and machine translation. *CoRR*, abs/1609.08144.

Nianwen Xue, Hwee Tou Ng, Sameer Pradhan, Rashmi Prasad, Christopher Bryant, and Attapol Rutherford. 2015. The CoNLL-2015 shared task on shallow discourse parsing. In *Proceedings of the Nineteenth Conference on Computational Natural Language Learning - Shared Task*, pages 1–16, Beijing, China. Association for Computational Linguistics.

Nianwen Xue, Hwee Tou Ng, Sameer Pradhan, Attapol Rutherford, Bonnie Webber, Chuan Wang, and Hongmin Wang. 2016. CoNLL 2016 shared task on multilingual shallow discourse parsing. In *Proceedings of the CoNLL-16 shared task*, pages 1–19, Berlin, Germany. Association for Computational Linguistics.

Yue Yu, Yilun Zhu, Yang Liu, Yan Liu, Siyao Peng, Mackenzie Gong, and Amir Zeldes. 2019. GumDrop at the DISRPT2019 shared task: A model stacking approach to discourse unit segmentation and connective detection. In *Proceedings of the Workshop on Discourse Relation Parsing and Treebanking 2019*, pages 133–143, Minneapolis, MN. Association for Computational Linguistics.

Amir Zeldes, Debopam Das, Erick Galani Maziero, Juliano Antonio, and Mikel Iruskieta. 2019. The DISRPT 2019 shared task on elementary discourse unit segmentation and connective detection. In *Proceedings of the Workshop on Discourse Relation Parsing and Treebanking 2019*, pages 97–104, Minneapolis, MN. Association for Computational Linguistics.

Amir Zeldes and Yang Liu. 2020. A neural approach to discourse relation signal detection. *Dialogue Discourse*, 11(2):1–33.

Amir Zeldes, Yang Janet Liu, Mikel Iruskieta, Philippe Muller, Chloé Braud, and Sonia Badene. 2021. The DISRPT 2021 shared task on elementary discourse unit segmentation, connective detection, and relation classification. In *Proceedings of the 2nd Shared Task on Discourse Relation Parsing and Treebanking (DISRPT 2021)*, pages 1–12, Punta Cana, Dominican Republic. Association for Computational Linguistics.

A Carbon Footprint

We access carbon footprints by using the *codecarbon*[1] framework. The corpus is processed once in the beginning and embeddings are persistently stored on disk for later usage. By preparing theses embeddings, we roughly emitted 7.93g carbon dioxide (CO2). Table 3 shows the carbon footprints for each model and provide averaged values per experiment. In total, as we avoid training large models by ourselves, and rather use features taken from pre-trained language models as provided, we emitted about 266.15g CO2 for all our experiments.

Model-Type	Config	Emission (g CO2)
candidate	advance ctx=0	2.97 (0.01)
candidate	advance ctx=1	3.05 (0.04)
candidate	advance ctx=2	2.98 (0.02)
candidate	simple ctx=0	3.96 (0.54)
candidate	simple ctx=1	3.99 (0.56)
candidate	simple ctx=2	4.42 (0.15)
label	ctx (all)	26.11 (3.09)
label	simple (all)	16.04 (0.49)
label	ctx (pos)	2.73 (0.61)
label	simple (pos)	1.79 (0.17)
sentence	ctx (all)	10.97 (0.93)
sentence	simple (all)	7.07 (0.54)

Table 3: Carbon footprint approximations averaged over runs with standard deviation.

B Extracted Candidate Patterns

The following tables give a more detailed overview about the pattern-extraction mechanisms. In Table 4, the top 30 extracted patterns are given for both approaches, exact extraction on the left side and context-sensitive extraction on the right side, with one additional token to the left and right. This is complemented in Table 5 by the number of extracted candidates per rule on a randomly sampled test set, as described in Section 5.

[1] https://github.com/mlco2/codecarbon

#	Exact	Count	Context-Sensitive	Count
1	VBG	468	, VBG DT	162
2	RB	113	, VBG NNS	39
3	VBN IN	59	, VBG PRP	35
4	IN DT NN	45	, VBG NN	31
5	DT VBZ	39	BOS IN DT NN ,	29
6	DT VBZ IN	31	, VBG JJ	26
7	IN RB	24	, VBG IN	25
8	IN NN IN	23	, VBG NNP	18
9	JJ	17	, VBN IN DT	18
10	VBG IN	16	BOS RB ,	17
11	DT NN VBZ	15	BOS IN RB ,	15
12	RB TO	15	BOS IN NN IN VBG	15
13	DT NN VBD	14	, VBG TO	14
14	IN VBD	14	, VBG JJR	14
15	RB VBG	12	, VBG PRP$	14
16	WP VBZ JJR	12	, RB TO VB	13
17	RB RB	12	BOS WP VBZ JJR ,	12
18	DT MD VB	12	, VBG IN DT	12
19	DT NN	11	, VBG RB	10
20	IN VBZ	11	BOS DT VBZ IN DT	10
21	IN DT	11	, VBN IN JJ	9
22	DT VBD	10	BOS DT NN :	8
23	IN NN	10	DT JJ NN	8
24	RB RB IN	10	, RB ,	7
25	DT NN VBD IN	9	BOS DT NN VBD DT	7
26	IN CD NN	9	BOS IN DT ,	7
27	VBD DT NN IN	9	BOS IN CD NN ,	7
28	DT JJ NN VBZ	8	, RB .	7
29	VBG RP	7	, VBG VBG	7
30	DT NN MD VB	7	BOS VBG DT	7

Table 4: Top 30 of extracted patterns from full data set. Comparison of simple patterns with their context-sensitive counter parts.

#	Exact	Count	Context-Sensitive	Count
1	NN	18686	JJ	1541
2	IN	13915	RB	1100
3	DT	10907	DT	552
4	JJ	7698	VBG	516
5	DT NN	5278	IN	245
6	IN DT	4654	DT NN	206
7	NN IN	4517	VBZ	142
8	RB	4509	VBN IN	128
9	VBD	3997	VBN	104
10	VB	3565	IN DT NN	76
11	VBN	2910	TO	73
12	VBZ	2623	DT NN VBD	58
13	IN DT NN	2133	VBG IN	54
14	VBG	1869	IN NN	51
15	TO	1761	RB RB	46
16	TO VB	1728	DT NN VBZ	45
17	IN NN	1590	PRP VBD	30
18	DT JJ NN	1418	DT VBZ	30
19	IN JJ	1182	IN DT JJ NN	29
20	MD VB	1061	JJR	28
21	VBN IN	1035	IN CD	26
22	IN DT JJ	996	VBD DT NN IN	25
23	IN CD	917	PRP VBZ	21
24	CD NN	871	IN NN IN	20
25	IN NNS	765	VBD	17
26	NN VBZ	728	DT NN MD VB	15
27	NNS VBP	648	RB JJ	13
28	PRP VBD	626	VBG DT NN	13
29	IN DT JJ NN	622	RB TO	12
30	VBD IN	557	IN NNS	12

Table 5: Top 30 of extracted patterns from randomly sampled test partition. The ordering differs between **Exact** and **Context-Sensitive** due to different patterns. In comparison, context-sensitive patterns are extract more different, but much less candidates per pattern, for example interjections (IN) on the left are extracted about 57 times more than on the right.

Topicalization in Language Models: A Case Study on Japanese

Riki Fujihara[1] **Tatsuki Kuribayashi**[1,2] **Kaori Abe**[1]
Ryoko Tokuhisa[1] **Kentaro Inui**[1,3]
[1]Tohoku University [2]Langsmith Inc. [3]RIKEN
riki.fujihara.s4@dc.tohoku.ac.jp
{kuribayashi, abe-k, tokuhisa, inui}@tohoku.ac.jp

Abstract

Humans use different wordings depending on the context to facilitate efficient communication. For example, instead of completely new information, information related to the preceding context is typically placed at the sentence-initial position. In this study, we analyze whether neural language models (LMs) can capture such discourse-level preferences in text generation. Specifically, we focus on a particular aspect of discourse, namely the topic-comment structure. To analyze the linguistic knowledge of LMs separately, we chose the Japanese language, a topic-prominent language, for designing probing tasks, and we created human topicalization judgment data by crowdsourcing. Our experimental results suggest that LMs have different generalizations from humans; LMs exhibited less context-dependent behaviors toward topicalization judgment. These results highlight the need for the additional inductive biases to guide LMs to achieve successful discourse-level generalization.

1 Introduction

Building on the current success of neural language models (LMs) in the field of natural language processing (NLP), much work has been conducted to test their linguistic knowledge, typically, syntactic generalizations in LMs (Linzen et al., 2016; Lau et al., 2017; Marvin and Linzen, 2018; Goldberg, 2019; Warstadt et al., 2019, 2020). Although discourse is also an essential aspect of language production along with syntactic constructions, discourse-level knowledge in LMs has been less explored or has been typically analyzed at the coarse level, for example, analyzing their sentence ordering abilities (Li and Jurafsky, 2017; See et al., 2019).

As one step toward understanding the fine-grained, discourse-level knowledge in LMs, this study explores the generalization performance of

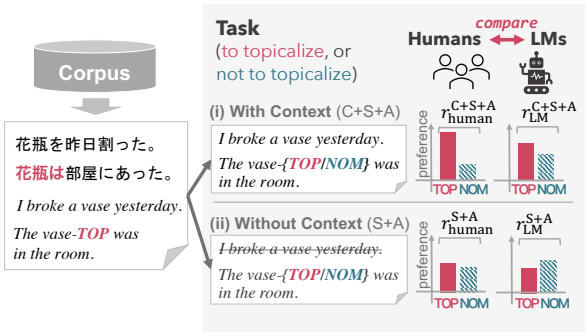

Figure 1: Comparing the context-dependent preferences of language models (LMs) and humans for topicalization.

LMs in a particular aspect of discourse, the topic-comment structure (i.e., the thematic structure). The topic-comment structure is an essential part of language production (Grosz et al., 1995; Halliday et al., 2014; Hajicová and Mírovskỳ, 2018).

For example, speakers use the following sentences in different contexts:

(1) a. In Japan, my father bought the vase last year.
b. The vase was the one my father bought in Japan last year.

These sentences differ in terms of topic-comment structure (the topic is underlined). For example, Sentence (1b) is more suited than (1a), for example, as a continuation of the sentence *I broke a vase yesterday.* This study probes whether LMs can make such context-dependent, human-like paradigmatic choices at the discourse level.

It is worth noting that the (non-)human-like linguistic generalization ability of computational models has long been examined in the cognitive science of language (Rumelhart and McClelland, 1985; Hu et al., 2020). Furthermore, from an engineering point of view, whether LMs have correct preferences for topicalization is an important perspective for validating the use of LMs for automatic assessment of text quality.

Proceedings of the 29th International Conference on Computational Linguistics, pages 851–862
October 12–17, 2022.

To test the cognitive plausibility of LMs' behaviors toward topic-comment structure, we used a topic-prominent language (TPL), where the topic-comment structure is explicitly realized. The use of a TPL facilitated the careful creation of text pairs for analyzing humans and LMs behaviors. We first annotated and analyzed the human preferences for topicalization using crowdsourcing (Sections 3 and 4).

In our experiments, we compared LMs with humans with respect to their context-dependent preferences for topicalization (Section 5; Figure 1). Our experimental results indicated that LMs do not show human-like, context-dependent behaviors (Section 6). Further analysis showed that LMs perform topicalization judgment based on context-independent spurious biases regarding whether a particular noun phrase is likely to be topicalized. This result reveals that compared to humans, LMs perform different generalizations toward topicalization judgment; hence, additional inductive biases are required to achieve human-like generalizations. Our dataset and code are publicly available. [1]

2 Background

Topic of a sentence. The topic of a sentence represents the concern of the message; what the speaker is going on to say. In the literature, the topic typically corresponds to the sentence-initial element (Halliday et al., 2014; Vallduví, 1990).[2] For example, the topic of Example (1a) (Section 1) is *Japan*, while that of Example (1b) is *vase*.

The term **topicalization** means to mark a particular element in a sentence as the topic of that sentence. Topicalization is realized typically by the word order as shown in Example (1), and different languages sometimes have different devices (e.g., particles and intonation) to mark a topic (Section 3.1).

The non-topic part of a sentence is called a comment. The topic and comment are often studied as a topic-comment structure or a theme-rheme structure in linguistics (Grosz et al., 1995; Halliday et al., 2014; Hajicová and Mírovskỳ, 2018). These are distinguished from the grammatical and logical structures of a sentence; for example, the topic of Example (1a) is *Japan*, while the grammatical subject is *my father*. Note that a sentence may not always have a topic (e.g., The sentence "There is a pen." does not introduce a topical theme).

Topicalization and discourse. In text production, the topic of a sentence plays an important role as it indicates the concern of the message and controls word ordering (Example (1)). In general, the more salient a particular concept is in a context, the more likely it is to be topicalized (Halliday et al., 2014; Miltsakaki, 1999). This property is closely related to the **centering** in discourse (Chafe, 1976; Givón, 1983; Grosz et al., 1995). In this theory, the topic of a sentence has a higher centering-forward (Cf) level according to the grammatical ranking, and such a higher-ranked entity is expected to have a high Cf rank again in the next utterance (i.e., the topic is expected to continue across successive utterances).

In this work, we determine whether LMs have human-like preferences for judging which elements in a sentence should be a topic (i.e., topicalize) when conveying meaning. Considering the context-dependent nature of topicalization, analysis of the preference of LMs in topicalization can provide insight to examine whether LMs capture the contextual flow of text at the inter-sentential, and discourse levels.

Linguistic probes. NLP researchers have inspected the inner workings and/or behaviors of black-box neural models to explore whether they actually understand the language. The focus of such probing analyses ranges from, for example, syntactic knowledge (Marvin and Linzen, 2018; Hu et al., 2020; Hewitt and Manning, 2019) to common sense knowledge (Lin et al., 2020; Zhou et al., 2020). Some analyses have also targeted discourse-level phenomena such as coreference (Sorodoc et al., 2020; Upadhye et al., 2020) and discourse structures (Pandia et al., 2021; Kurfalı and Östling, 2021; Koto et al., 2021). These existing studies and our present work are complementary in covering a wide variety of discourse phenomena.

To probe the linguistic knowledge of LMs, researchers have typically used minimally different text pairs (MDTPs) that differ *only* in a certain linguistic aspect and analyzed their probabilities computed by LMs (Marvin and Linzen, 2018; Gauthier et al.; Hu et al., 2020). This experimental design has advantages: for example, it enables researchers

[1] https://github.com/rk-fujifuji/lm_topicalization

[2] Strictly speaking, there are several definitions of the topic (e.g., textual, interpersonal, and topical); in this study, we focuses on the topical theme introduced in Halliday et al. (2014).

to analyze the model behaviors directly, without designing additional classifiers (Alain and Bengio, 2017; Pimentel et al., 2020). We also probe LMs using MDTPs that differ *only* in the topic-comment structure.

3 Dataset: collecting human preferences

3.1 Using topic-prominent language

In English, creating the MDTPs differing only in the topic-comment structure is prohibitively difficult because the change of topicalized elements in a sentence accompanies a drastic change in the sentence structure and syntactic complexities (Example (1)). Thus, simply analyzing the preferences of LMs between sentences with different topic-comment structures in English may lead to confusing conclusions about which linguistic perspective LMs are actually sensitive to.

In contrast, in Japanese, a TPL, creating a set of MDTPs differing only in the topic-comment structure is possible. For example, the following two sentences in Japanese, one of the TPLs, differ *only* in the topic of the sentence:

(2) a. *Kabin-wa heya-ni at-ta.*
Vase-TOP room-DAT exist-PAST.
The vase was in the room.
b. *Kabin-ga heya-ni at-ta.*
Vase-NOM room-DAT exist-PAST.
There was a vase in the room.

In Japanese, the postpositional particle *wa* (TOP) is used as the *topic marker* for indicating the topic of a sentence as in Example (2a) (Teruya, 2004, 2007; Kuno, 1973; Noda, 1996).[3] The difference between *Kabin-wa* (*Vase*-TOP) and *Kabin-ga* (*Vase*-NOM) in Example (2) is whether the element (*Kabin*; *vase*) is marked as a topic or not.

Note that Japanese is an agglutinative language, in which the functional information (e.g., grammatical case) of an element is realized by postpositional particles. When a particular element is marked by *wa* (TOP), some originally used particles (e.g., *ga*; NOM) are omitted. That is, *Kabin-wa* (*Vase*-TOP) in Example (2a) plays the roles of the grammatical subject (NOM) and the topic of a sentence (TOP) both, but only *wa* is attached.

Example (2a), in which the grammatical subject (*kabin*; *vase*) is topicalized, should be preferred to Example (2b) in the context of talking about the vase. We analyzed whether LMs have such context-dependent preferences for using topic markers. Such a paradigmatic choice of topicalization is not determined by strict rules. Rather, both Example (2a) and (2b) are usually acceptable, but either sentence is sometimes *more natural* than the other, depending on the context. We created a dataset of such *degree*s of human preference for topicalization.

Note that grammatical topic (TOP) is assumed to have a higher Cf ranking than the subject (NOM) in Japanese centering theory (Walker et al., 1994); this task of context-dependently selecting TOP or NOM could be viewed as a task of estimating Cf from the perspective of centering theory.

3.2 Annotation task

Data preparation. We focused on the Japanese language as a representative of TPLs. Specifically, we analyzed the preference for topicalization of nominative arguments in Japanese sentences. We used the NAIST Text Corpus (NTC; Iida et al., 2007), which is commonly used for analyzing linguistic phenomena in Japanese. We collected nominative arguments and their belonging sentences that satisfied all of the following criteria from the NTC:

- An argument has a nominative relation to the verb that is closest to the end of a sentence, regarding the predicate-argument structure annotation.[4]
- An argument accompanies the topic marker (TOP) or nominative particle (NOM).
- An argument appears in the second, third, or fourth sentence in a paragraph.

Using these criteria, we collected the data for annotation $\mathcal{D} = \{(c, s, a)_d\}_{d=1}^{|\mathcal{D}|}$, where c is the inter-sentential context (preceding sentences within the same document), s is the intra-sentential context, and a is the nominative argument that satisfies the aforementioned criteria. This process yielded 1,661 data points, where 939 instances originally have the TOP particle for nominative argument a, and 722 instances have NOM. The examples are listed in Table 1.

[3] Strictly speaking, the contrast between *wa* (TOP) and *ga* (NOM) is not only about the topicalization, and has long been discussed in Japanese linguistics. We did not intend to claim that all the *wa* work as a topic marker; instead, we carefully selected the data points where *wa* is used as a topic marker (Section 3).

[4] Predicate-argument structure annotation in NTC is used.

Text		Human preferences		LM preferences	
Context c	Sentence s and targeted nominative argument a (underlined)	$r_{\text{human}}^{\text{C+S+A}}$	$r_{\text{human}}^{\text{S+A}}$	$r_{\text{LM}}^{\text{C+S+A}}$	$r_{\text{LM}}^{\text{S+A}}$
We consume large amounts of energy daily.	*Economic growth*-{TOP/NOM} *is closely related to energy use.*	0.83	1.00	0.66	0.64
It is said that a good start determines victory in yacht racing.	*A strong wind*-{TOP/NOM} *of 15 knots was blown in the sea.*	0.00	0.00	0.31	0.25
The seventh is the day of the "Seven Herbs of Spring" to pray for good health. At the Hanshin Department Store in Kita, Osaka, a free service of Nanakusagayu was offered from 8:00 a.m. Office workers, young women, and junior high school students on their way home from early morning kendo practice enjoyed its taste.	*The 500 meals*-{TOP/NOM} *prepared were gone in about an hour.*	1.00	0.50	0.58	0.58

Table 1: Examples of the instances $(c, s, a)_d$ and preferences of humans and LMs (TRANS-L) for topicalization. The human preference scores r_{human} are introduced in Section 3.2. The LMs preference scores r_{LM} are introduced in Section 5.1. The example texts are the English-translated versions of the original texts (Mainichi Shimbun article data 1995 version).

Annotation task. We collected the topicalization preferences using crowdsourcing. Specifically, for each instance $(c, s, a)_d$, we first masked the particle following the nominative argument a (*kabin*; *vase*) as follows:

(3) a. *Kabin-__ heya-ni at-ta.*
Vase-__ room-DAT exist-PAST.

Then, annotators were asked which postpositional particle TOP (*wa*; to topicalize) or NOM (*ga*; not to topicalize) is more natural to complete the blank __. The option of "both okay" was also available.

For each instance $(c, s, a)_d$, from four to eight subjects annotated the preference. Finally, for each instance $(c, s, a)_d$, we calculated **topicalization ratio** as follows:

$$r_{\text{human}} = \frac{n_{\text{TOP}}}{n_{\text{TOP}} + n_{\text{NOM}}} \quad , \tag{1}$$

where n_{TOP} and n_{NOM} are the number of votes for completing the blank (e.g., *Kabin-__*) with TOP and NOM, respectively. For "both okay," we considered TOP and NOM to have 0.5 votes for each.

Context ablation. To facilitate analyzing the context-dependent characteristics of the topicalization, annotators solved the task under three different conditions. Table 2 shows each condition with an example. Here, ✓ in the **context** column indicates that the inter-sentential context is shown to annotators, and ✓ in the **sentence** column indicates whether the intra-sentential context is shown. When the intra-sentential context is not shown, the nominal constituent alone is provided.

Subsequently, three variants of the topicalization ratio r_{human} for each instance $(c, s, a)_d$ were obtained under different ablation settings: (i) the ratio $r_{\text{human}}^{\text{C+S+A}}$ obtained with the C+S+A setting, (ii) $r_{\text{human}}^{\text{S+A}}$ with the S+A setting, and (iii) $r_{\text{human}}^{\text{A}}$ with the A setting. The examples of scores are listed in Table 1. In Section 5, we observed the preferences of LMs for topicalization and compared them with those of humans.

Intended use of the annotations. We used the annotations obtained in the C+S+A and S+A settings (with relatively high agreement) for our main experiments (Section 5). The data of the A settings were used in our additional analyses (Section 6) to test whether the LMs also exhibit such a human-like difficulty in this setting. We observed some interesting trends: LMs exhibited unreasonably good performance of topicalization prediction in the context-independent, A settings.

3.3 Crowdsourcing

Worker selection. We used crowdsourcing[5] to access Japanese subjects. Crowd workers solved the task of selecting TOP or NOM. To qualify the motivated crowd workers, we first created trial tasks, in which each worker answered 10 questions by selecting TOP or NOM. For this purpose, we framed the validation questions in advance, where

[5] https://crowdsourcing.yahoo.co.jp/

	context c	sent. s	arg. a	question: TOP or NOM for __?
C+S+A	✓	✓	✓	*Kabin-wo kinou wat-ta. Kabin-__ heya-ni at-ta.* *I broke a vase yesterday. {The vase was}/{There was a vase} in the room.*
S+A		✓	✓	*Kabin-__ heya-ni at-ta.* *{The vase was}/{There was a vase} in the room.*
A			✓	*Kabin-__* *Vase*

Table 2: Settings for solving the topicalization judgment tasks. The Japanese example sentence "*Kabin-wo kinou wat-ta. Kabin-__ heya-ni at-ta.*" means "*I broke a vase yesterday. {The vase was}/{There was a vase} in the room.*" in English. The question is whether to topicalize the word "vase" in the second sentence.

Setting	#labels	class distribution		
		TOP	Both okay	NOM
C+S+A	8,039	55.3%	1.49%	43.2%
S+A	8,094	52.9%	3.47%	43.6%
A	7,621	1.86%	96.7 %	1.42 %

Table 3: Statistics of the whole dataset. The #labels denote the number of workers who annotated the labels remaining after the post-processing. The class distribution. columns denote the percentages of TOP, "Both okay", and NOM in the answers.

our preferences were in agreement; one of the 10 questions is a validation example. Then, we listed the motivated workers who can answer the validation question correctly. At this stage, 164 workers were selected in the C+S+A and S+A settings, and 153 workers were selected in the A setting. [6]

Annotation. For each instance $(c, s, a)_d$ of the 1,661 data points collected in Section 3.2, eight of the motivated workers annotated the preferences. Each worker solved at least ten instances. The same worker is not annotated to the same instance in different settings. After the whole annotation process, we performed statistical post-processing to exclude workers in the bottom 30% of competence using MACE (Hovy et al., 2013). We removed the data points annotated by fewer than four qualified workers.

Finally, 23,745 decisions selecting TOP or NOM for 1,355 nominative arguments were collected, where 758 instances originally have the TOP particle for nominative argument a, and 597 instances have NOM.

3.4 Statistics

The number of human decisions and their class distribution is shown in Table 3. The length of the context c, sentence s, and nominative argument a was 96.3±53.7, 50.1±24.4, and 4.5±2.2 in the number of characters (mean±standard deviation), respectively.[7]

The annotation agreement was 0.689 and 0.699 for Krippendorff's alpha for the settings of C+S+A and S+A, respectively.[8] These values are above the minimum criteria for data reliability (0.667) (Krippendorff, 2004). In contrast, the agreement in the A settings (0.074) was far below the criteria. One plausible cause of such a low score is that the majority of annotators answered as "both okay" in these settings. This skewed the class distribution, and the alpha value is affected by class imbalances (Jeni et al., 2013). Since many workers answered "both okay," we tentatively conclude that making topicalization judgments in this setting is difficult for Japanese speakers.

4 Data analysis: Context effect in topicalization

Before our experiments, we preliminarily observed the characteristics of the collected data. If all the annotated preference remains unchanged regardless of the inter-sentential context, our data are not suitable for analyzing the discourse-level behaviors in LMs. Our analysis declines such a concern.

[6] We conducted crowdsourcing in two separate sessions; in the first session, we adopted the S+C+A and C+A settings, and in the second, we adopted the A setting. In each session, after the trial task, the worker's confidence level was calculated by MACE (Hovy et al., 2013). Then, the top 80% of workers (164 and 153 workers for the first and second session, respectively) were selected.

[7] The Japanese language has no explicit word boundary. As an approximation of word count, context c, sentence s, and nominative argument a have 56.4±31.7, 29.5±14.4, and 2.8±1.1 morphemes, respectively. We used a JUMAN dictionary for morphological analysis (Kawahara and Kurohashi, 2006).

[8] We used the weighted Krippendorff's alpha, where we assumed the distance scale as TOP ≺ "both okay" ≺ NOM.

#Mentions	#Arguments	$r^{\mathrm{C+S+A}}_{\mathrm{human}}$
0	1,082	0.48 ± 0.43
1	197	0.87 ± 0.25
2+	76	0.90 ± 0.22

Table 4: Frequently mentioned information in a context (i.e., higher #Mention) is likely to be topicalized (i.e., higher $r^{\mathrm{C+S+A}}_{\mathrm{human}}$). #Argument is the number of the corresponding data points. The mean and standard deviation of $r^{\mathrm{C+S+A}}_{\mathrm{human}}$ are presented.

Context effect on topicalization confidence. We first analyze the interaction between topicalization preference and context, regarding the linguistic theory that *already-mentioned, old information is more likely to be topicalized by the topic marker in Japanese* (Matsushita, 1930). We observed that the nominative argument frequently mentioned in its context tends to gain a higher topicalization ratio $r^{\mathrm{C+S+A}}_{\mathrm{human}}$ (Table 4). This supports that the created data reflect linguistically natural trends at the discourse level. Here, we used the co-reference annotation in the NTC to count how many times the same entity as the nominatives appeared in the preceding context.

Next, for each instance $(c,\ s,\ a)_d$, we quantified the context-dependent changes in topicalization preference as follows:

$$\Delta_{\mathrm{human}} = r^{\mathrm{C+S+A}}_{\mathrm{human}} - r^{\mathrm{S+A}}_{\mathrm{human}} \quad . \qquad (2)$$

Here, Δ denotes the change in the level of certainty in choosing TOP due to the presence of inter-sentential context.[9] Intuitively, a larger Δ indicates more votes on the topic marker when the context is available.

Figure 2 shows the distribution of the preference difference attributed to the inter-sentential context (Δ_{human}). We found that the preference changed depending on the presence of inter-sentential context (about 53% of data points have non-zero Δ_{human}).

Challenging set for probing LMs. The dataset has context-dependent nature, but there are also data points for which discourse-level context does not affect human behavior, we also created a **challenging** set. In this set, the context information is considered necessary for choosing TOP or NOM. Specifically, we selected the instances $(c,\ s,\ a)_d$ satisfying either of the following criteria:

- Original text and crowd workers demonstrated that a should be topicalized ($r^{\mathrm{C+S+A}}_{\mathrm{human}} > 0.5$), and inter-sentential context affected such a decision (Δ_{human} is in the top 25%).
- Original text and crowd workers demonstrated that a should not be topicalized ($r^{\mathrm{C+S+A}}_{\mathrm{human}} < 0.5$), and inter-sentential context affected such a decision (Δ_{human} is in the bottom 25%).

Finally, we obtained 311 instances, among which 209 instances originally had the TOP particle as the nominative argument a, and 102 instances had NOM. The statistics of the challenging set are shown in Appendix. We used this challenging set along with the whole dataset for our experiments to test the LMs (Section 5).

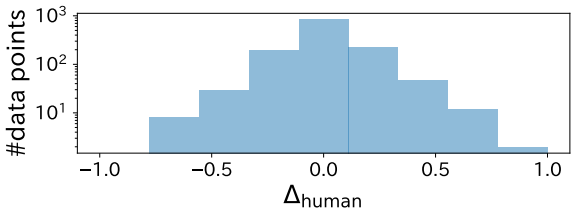

Figure 2: Histogram of the change in the topicalization ratio due to the presence of context (Δ_{human}).

5 Experiments: Comparing LMs with humans

Do LMs exhibit human-like topicalization preferences? To answer this question, we compared contrast the preferences of LMs and humans.

5.1 Experimental settings

Language models. We tested three variants of left-to-right LMs: Transformer-based LM with 400M parameters (TRANS-L), Transformer-based LM with 55M parameters (TRANS-S), and an LSTM-based LM with 55M parameters (LSTM) (Vaswani et al., 2017; Hochreiter and Schmidhuber, 1997). They were trained with about 3M paragraphs from Japanese newspapers and Wikipedia (3.4GB before any tokenization) with 100K parameter updates. The input was segmented into morphemes by JUMAN (Kawahara and Kurohashi, 2006), and further into subwords by sentencepiece (Kudo and Richardson, 2018), where the

[9] We tentatively adopted the **difference** of the ratio rather than, for example, ratio $\Delta_{\mathrm{human}} = r^{\mathrm{C+S+A}}_{\mathrm{human}} / r^{\mathrm{S+A}}_{\mathrm{human}}$. At least in the case of ratio, it is counter-intuitive to assume a change of 2 when the votes for TOP increase from 1 to 2, and 1.2 when they increase from 5 to 6.

Model	Setting	ρ_r	ρ_Δ	Macro F1	TOP F1	NOM F1
TRANS-L	C+S+A	0.67	−0.12	83.5	88.8	78.3
	S+A	0.60		81.7	87.6	75.8
TRANS-S	C+S+A	0.72	−0.07	85.3	89.5	81.1
	S+A	0.61		83.7	88.1	79.3
LSTM	C+S+A	0.69	−0.20	81.9	86.9	77.0
	S+A	0.62		82.3	87.1	77.5
Human	C+S+A	-	-	(100)	(100)	(100)
	S+A	-		81.1	86.5	75.7

Table 5: Results for the challenging set. The ρ_r denotes the rank correlation coefficient of the topicalization ratio exhibited by humans and LMs in each setting. The ρ_Δ denotes the rank correlation coefficient of the change in the topicalization ratio due to the presence of inter-sentential context in humans and LMs. The F1 scores were calculated with the topicalization judgment in the original text.

unigram model was used (Kudo, 2018).[10] Their hyperparameters are provided in Appendix.

Preferences of LMs. Analogous to Equation 1, we quantified the preferences of topicalization in LMs. In the S+A setting, for example, we created the MDTP of $s_{\texttt{TOP}}$ and $s_{\texttt{NOM}}$ for each instance $(c, s, a)_d$. Then, aligned with Equation 1, we calculated **topicalization ratio** of LMs as follows:

$$r_{\mathrm{LM}} = \frac{n(s_{\texttt{TOP}})}{n(s_{\texttt{TOP}}) + n(s_{\texttt{NOM}})} \ ,$$
$$n(s) := \prod_{i=1}^{|s|} p(w_i|w_{<i})^{\frac{1}{|s|}} \ , \qquad (3)$$

where $n(s)$ is the generation probability of a given sentence computed by an LM.

For each data point, two variants of topicalization ratio r_{LM} were calculated: $r_{\mathrm{LM}}^{\mathrm{C+S+A}}$ with the context c, and $r_{\mathrm{LM}}^{\mathrm{S+A}}$ without c. The score $r_{\mathrm{LM}}^{\mathrm{S+A}}$ was calculated with Equation 3, while $r_{\mathrm{LM}}^{\mathrm{C+S+A}}$ was calculated using conditional probabilities $n(s|c) = \prod_{i=1}^{|s|} p(w_i|c, w_{<i})^{\frac{1}{|s|}}$, instead of $n(s)$, in Equation 3.

Metrics We tested the LMs in terms of whether the level of certainty in choosing TOP was human-like. Specifically, Spearman's rank correlation coefficient between $r_{\mathrm{human}}^{\mathrm{C+S+A}}$ and $r_{\mathrm{LM}}^{\mathrm{C+S+A}}$ (henceforth, ρ_r) was measured. We expect that the more humans prefered TOP the more did LMs too.

Furthermore, to analyze whether the sensitivity of LMs to the context is human-like, we computed the change in LMs' topicalization preferences, analogous to Equation 2:

$$\Delta_{\mathrm{LM}} = r_{\mathrm{LM}}^{\mathrm{C+S+A}} - r_{\mathrm{LM}}^{\mathrm{S+A}}.$$

To quantify the similarity of context-sensitive preference change in LMs and humans, the rank correlation coefficients between Δ_{human} and Δ_{LM} were reported (henceforth, ρ_Δ).

Additionally, we reported F1 scores of the humans and LMs, regarding the particle choices (i.e., TOP or NOM) in the original text as the gold reference. Here, humans/LMs were considered to choose TOP when the topicalization ratio $r_{\mathrm{human/LM}}$ exceeded 0.5; otherwise, they were considered to choose NOM. Note that the corpus is from newspapers; more or less, these judgments reflect the proper use of the Japanese topic marker.

5.2 Results

Table 5 shows the results for the challenging set. In all the settings, we observed moderate correlations between humans and LMs with respect to the topicalization preferences (i.e., the ρ_r column).

Non-human-like context sensitivity. Despite the somewhat high correlations in topicalization preferences ρ_r, the correlation of context-sensitivity between humans and LMs ρ_Δ was negative; the changes in the topicalization preferences due to the presence of inter-sentential context differs between humans and LMs. This difference reveals that the context use of LMs and humans has substantial gaps.

[10] We used character coverage=0.9995, vocab size=100,000.

Model	Setting	ρ_r	ρ_Δ	Macro F1	TOP F1	NOM F1
TRANS-L	C+S+A	0.80	−0.04	88.1	89.3	86.8
	S+A	0.78		87.5	88.8	86.2
TRANS-S	C+S+A	0.80	−0.02	87.7	88.8	86.5
	S+A	0.78		87.3	88.3	86.3
LSTM	C+S+A	0.79	−0.01	85.2	86.4	84.1
	S+A	0.78		85.3	86.4	84.2
Human	C+S+A	-	-	89.7	91.0	88.4
	S+A	-		89.1	90.4	87.8

Table 6: Results for the whole dataset. The ρ_r denotes the rank correlation coefficient of the topicalization ratio exhibited by humans and LMs in each setting, and ρ_Δ denotes the rank correlation coefficient of the change in the topicalization ratio due to the presence of the inter-sentential context in humans and LMs. The F1 scores were calculated with the topicalization judgment in the original text.

LMs' insensitivity to inter-sentential context. Focusing on the differences in the F1 scores between C+S+A and S+A settings, little difference exists among the LM results. In addition, LMs typically outperform humans in terms of the F1 scores in the S+A setting. Specifically, as for the LSTM results, we obtained a somewhat strange tendency that better topicalization decisions were made better when the context is not considered. These results raise the suspicion that LMs might have made the topicalization decision with some non-contextual cues and might have performed generalizations different from those of humans.

Examples. Table 1 shows several examples of preferences of humans and LMs toward topicalization. The third example in Table 1, which belongs to the challenging set, highlighted the discrepancy between humans and LMs with respect to the context-(in)dependent preferences for topicalization. While humans chose TOP only when considering the preceding context, preferences of LMs did not change when the context information was provided.

Results in the whole dataset. We also benchmarked the topicalization judgments of the LMs in the whole dataset that we created by crowdsourcing, without limiting to the challenging set. Table 6 shows the results. Notably, even in the whole dataset, there is almost no correlation of context-sensitivity ρ_Δ between humans and LMs.

Model	ρ_r	F1		
		Macro	TOP	NOM
TRANS-L	0.18	63.8	71.1	56.5
TRANS-S	0.18	65.9	72.3	59.4
LSTM	0.17	65.1	71.9	58.4
Human	-	36.8	14.0	59.3

Table 7: The results for the A setting, where the nominative constituent alone was shown to the subjects for making the topicalization judgment.

6 Analysis

Our experiments showed that the topicalization decision of LMs is unreasonably decontextualized. To further understand such non-human-like behaviors of LMs, we investigated their behavior in situations with extremely limited context information.

As introduced in Section 3.4, we observed that humans struggle with topicalization judgment in the setting where the nominative constituent alone is shown (the A setting; Table 2). We tested whether LMs also exhibit such human-like difficulties when dealing with the whole dataset under the A setting. Notably, the purpose of this analysis is to find the situation in which LMs deviate from humans.

Language model preference. For each nominative argument a in the dataset, we computed the topicalization ratio when only the a is shown to LMs. Specifically, LMs computed the generation probability of, for example, *Kabin-ga* and *Kabin-wa*, regardless of any context, and subsequently, we compared these probabilities as per Equation. 3.

Results. Table 7 shows the results of the topicalization preferences when only the nominative constituent was shown to LMs/humans. First, the F1 scores of the LMs were surprisingly higher than those of the humans. This suggests that LMs could somehow predict the postpositional particle (TOP or NOM) without access to contextual information, although topicalization is a discourse-level phenomenon. Second, the correlation of topicalization ratio between LMs and humans was quite low. This result suggests that topicalization preferences of the LMs for nominative constituents deviate from those of humans.

7 Discussion

Contributions to linguistics. We posit that our dataset itself is also valuable for linguistic studies. Topicalization in Japanese has captured the attention in the field of linguistics for more than half a century (Matsushita, 1930; Kuno, 1973; Noda, 1996), but resources created using a large-scale corpus and crowdsourcing have been limited.

The observed relationship between mention frequency and topicalization preference (Table 4), for example, could be viewed as empirical support for the relationship between the newness of information and topicalization preference (Matsushita, 1930). The whole dataset also suggested that the intra-sentential context provides informative clues for topicalization judgment, regarding the relatively high agreement in the S+A setting. This implies that the use of TOP could not often be aligned to context-dependent topicalization phenomena; this view is consistent with Imamura et al. (2014). Note that such context-independent instances are excluded from the challenging set.

Testing coherence models. While we evaluated only the vanilla LMs, there are several options for coherence modeling, such as entity-based coherence models (Barzilay and Lapata, 2008) and neural LMs that are explicitly trained with coherence objectives (Jwalapuram et al., 2022). It would be interesting to investigate whether such models exhibit more human-like behaviors in terms of topicalization preference.

8 Conclusions

We have compared the preferences of LMs and humans for topicalization, an essential aspect of discourse. The results suggest that there exists a discrepancy between humans and LMs with respect to the generalizations for topicalization. This implication leads us to future research: *what type of inductive biases in model architecture or training objectives can lead to more human-like generalizations in discourse-level linguistic aspects?*

Acknowledgments

This work was supported by Grant-in-Aid for JSPS Fellows Grant Number JP20J22697, and JST CREST Grant Number JPMJCR20D2.

Ethical considerations

We released a new annotation layer (topicalization preference) for the existing Japanese corpus (NTC). Such preference data do not introduce any harmful content nor privacy information associated with crowd workers to the original corpus. The source NTC corpus was created using a certain Japanese newspaper (Mainichi Shinbun), which has been widely used in the NLP field (e.g., Japanese predicate argument structure analysis).

References

Guillaume Alain and Yoshua Bengio. 2017. Understanding intermediate layers using linear classifier probes. In *Proceedings of ICLR Workshop*. OpenReview.net.

Regina Barzilay and Mirella Lapata. 2008. Modeling local coherence: An entity-based approach. *Computational Linguistics*, 34:1–34.

Wallace L. Chafe. 1976. Givenness, contrastiveness, definiteness, subjects, topics, and point of view.

Jon Gauthier, Jennifer Hu, Ethan Wilcox, Peng Qian, and Roger Levy. Syntaxgym: An online platform for targeted evaluation of language models. In *Proceedings of ACL (System Demonstrations)*.

T Givón. 1983. *Topic Continuity in Discourse: A quantitative cross-language study*. John Benjamins.

Yoav Goldberg. 2019. Assessing bert's syntactic abilities. *arXiv preprint arXiv:1901.05287*.

Barbara J Grosz, Aravind K Joshi, and Scott Weinstein. 1995. Centering: A framework for modelling the local coherence of discourse. *Computational Linguistics*, 21(2):203–225.

Eva Hajicová and Jiří Mírovskỳ. 2018. Discourse coherence through the lens of an annotated text corpus: A case study. In *Proceedings of LREC*, pages 1637–1642.

Michael Halliday, Christian MIM Matthiessen, and Christian Matthiessen. 2014. *An Introduction to Functional Grammar*. Routledge.

John Hewitt and Christopher D. Manning. 2019. A structural probe for finding syntax in word representations. In *Proceedings of NAACL-HLT*, pages 4129–4138.

Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long short-term memory. *Neural computation*, 9(8):1735–1780.

Dirk Hovy, Taylor Berg-Kirkpatrick, Ashish Vaswani, and Eduard Hovy. 2013. Learning whom to trust with mace. In *Proceedings of NAACL-HLT*, pages 1120–1130.

Jennifer Hu, Jon Gauthier, Peng Qian, Ethan Wilcox, and Roger Levy. 2020. A systematic assessment of syntactic generalization in neural language models. In *Proceedings of ACL*, pages 1725–1744.

Ryu Iida, Mamoru Komachi, Kentaro Inui, and Yuji Matsumoto. 2007. Annotating a japanese text corpus with predicate-argument and coreference relations. In *Proceedings of the linguistic annotation workshop*, pages 132–139.

Satoshi Imamura, Yohei Sato, and Masatoshi Koizumi. 2014. Influence of Information Structure on Word Order Change and Topic Marker WA in Japanese. In *Proceedings of PACLIC*.

László A. Jeni, Jeffrey F. Cohn, and Fernando De la Torre. 2013. Facing imbalanced data–recommendations for the use of performance metrics. In *In Proceedings of ACII*, pages 245–251.

Prathyusha Jwalapuram, Shafiq Joty, and Xiang Lin. 2022. Rethinking self-supervision objectives for generalizable coherence modeling. In *Proceedings of ACL*, pages 6044–6059.

Daisuke Kawahara and Sadao Kurohashi. 2006. Case frame compilation from the web using high-performance computing. In *Proceedings of LREC*, pages 1344–1347.

Fajri Koto, Jey Han Lau, and Timothy Baldwin. 2021. Discourse probing of pretrained language models. In *Proceedings of NAACL-HLT*, pages 3849–3864.

Klaus Krippendorff. 2004. *Content Analysis: an Introduction to its Methodology*. Sage publications.

Taku Kudo. 2018. Subword regularization: Improving neural network translation models with multiple subword candidates. In *Proceedings of ACL*, pages 66–75.

Taku Kudo and John Richardson. 2018. Sentencepiece: A simple and language independent subword tokenizer and detokenizer for neural text processing. In *Proceedings of EMNLP*, pages 66–71.

Susumu Kuno. 1973. *Nihon bunpō kenkyū (Study of Japanese Grammar)*. Taishukan Shoten.

Murathan Kurfalı and Robert Östling. 2021. Probing multilingual language models for discourse. In *Proceedings of RepL4NLP*, pages 8–19.

Jey Han Lau, Alexander Clark, and Shalom Lappin. 2017. Grammaticality, acceptability, and probability: A probabilistic view of linguistic knowledge. *Cognitive Science*, 41(5):1202–1241.

Jiwei Li and Dan Jurafsky. 2017. Neural net models of open-domain discourse coherence. In *Proceedings of EMNLP*, pages 198–209.

Bill Yuchen Lin, Seyeon Lee, Rahul Khanna, and Xiang Ren. 2020. Birds have four legs?! NumerSense: Probing Numerical Commonsense Knowledge of Pre-Trained Language Models. In *Proceedings of EMNLP*, pages 6862–6868.

Tal Linzen, Emmanuel Dupoux, and Yoav Goldberg. 2016. Assessing the ability of lstms to learn syntax-sensitive dependencies. *Transactions of the Association for Computational Linguistics*, 4:521–535.

Rebecca Marvin and Tal Linzen. 2018. Targeted syntactic evaluation of language models. In *Proceedings of EMNLP*, pages 1192–1202.

Daizaburo Matsushita. 1930. *Hyōjun nihon kougohou (Standard Japanese Spoken Method)*. Chūbunkan Shoten.

Eleni Miltsakaki. 1999. Locating topics in text processing. In *CLIN*, pages 127–138.

Hisashi Noda. 1996. *Wa to ga (Wa and ga)*. Kurosio Publishers.

Lalchand Pandia, Yan Cong, and Allyson Ettinger. 2021. Pragmatic competence of pre-trained language models through the lens of discourse connectives. In *Proceedings of CoNLL*, pages 367–379.

Tiago Pimentel, Naomi Saphra, Adina Williams, and Ryan Cotterell. 2020. Pareto probing: Trading off accuracy for complexity. In *Proceedings of EMNLP*, pages 3138–3153.

David E Rumelhart and James L McClelland. 1985. On learning the past tenses of english verbs. Technical report, California Univ., San Diego, La Jolla. Inst. for Cognitive Science.

Abigail See, Aneesh Pappu, Rohun Saxena, Akhila Yerukola, and Christopher D Manning. 2019. Do massively pretrained language models make better storytellers? In *Proceedings of CoNLL*, pages 843–861.

Ionut-Teodor Sorodoc, Kristina Gulordava, and Gemma Boleda. 2020. Probing for referential information in language models. In *Proceedings of ACL*, pages 4177–4189.

Kazuhiro Teruya. 2004. Metafunctional profile of the grammar of japanese. *Language typology. A functional perspective*, pages 185–254.

Kazuhiro Teruya. 2007. *A systemic functional grammar of Japanese*. Bloomsbury Academic.

Shiva Upadhye, Leon Bergen, and Andrew Kehler. 2020. Predicting reference: What do language models learn about discourse models? In *Proceedings of EMNLP*, pages 977–982.

Enric Vallduví. 1990. *The Informational Component*. Ph.D. thesis, University of Pennsylvania, Philadelphia.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Proceedings of NeurIPS*, pages 5998–6008.

Marilyn Walker, Masayo Iida, and Sharon Cote. 1994. Japanese discourse and the process of centering. *Computational Linguistics*, 20(2):193–231.

Alex Warstadt, Alicia Parrish, Haokun Liu, Anhad Mohananey, Wei Peng, Sheng-Fu Wang, and Samuel R. Bowman. 2020. Blimp: The benchmark of linguistic minimal pairs for english. *Transactions of the Association for Computational Linguistics*, 8:377–392.

Alex Warstadt, Amanpreet Singh, and Samuel R. Bowman. 2019. Neural network acceptability judgments. *Transactions of the Association for Computational Linguistics*, 7:625–641.

Xuhui Zhou, Yue Zhang, Leyang Cui, and Dandan Huang. 2020. Evaluating commonsense in pre-trained language models. In *Proceedings of AAAI*, volume 34, pages 9733–9740.

Setting	#labels	class distribution		
		TOP	Both okay	NOM
C+S+A	1,853	65.7%	1.73%	32.5%
S+A	1,900	54.9%	8.47%	36.6%
A	1,732	2.02%	97.1%	0.88%

Table 8: Statistics for the challenging set. The #labels denote the number of workers who annotated the labels remaining after the post-processing. The class distribution columns denote the percentages of TOP, "Both okay", and NOM in the answers.

Parameters		TRANS-L	TRANS-S	LSTM
Fairseq model	architecture	transformer_lm_gpt2_small	transformer_lm_gpt	lstm_lm
	adaptive softmax cut off	50,000, 140,000	50,000, 140,000	50,000, 140,000
	share-decoder-input-output-embed	True	True	True
	embed_dim	1,024	384	400
	ffn_embed_dim	4,096	2048	-
	hidden_size	-	-	1,024
	layers	24	8	2
	heads	16	6	-
	dropout	0.1	0.1	0.1
	attention_dropout	0.1	0.1	-
Optimizer	algorithm	AdamW	AdamW	AdamW
	learning rates	5e-4	5e-4	1e-3
	betas	(0.9, 0.98)	(0.9, 0.98)	(0.9, 0.98)
	weight decay	0.01	0.01	0.01
	clip norm	0.0	0.0	0.0
Learning rate scheduler	type	inverse_sqrt	inverse_sqrt	inverse_sqrt
	warmup updates	4,000	4,000	4,000
	warmup init lrarning rate	1e-7	1e-7	1e-7
Training	batch size	61,440 tokens	61,440 tokens	20,480 tokens
	sample-break-mode	none	none	none

Table 9: Hyperparameters of the language models.

"No, they did not": Dialogue response dynamics in pre-trained language models

Sanghee J. Kim[1], **Lang Yu**[2], **Allyson Ettinger**[1]
[1]Department of Linguistics, University of Chicago
[2]Meta
{sangheekim,aettinger}@uchicago.edu, langyu@fb.com

Abstract

A critical component of competence in language is being able to identify relevant components of an utterance and reply appropriately. In this paper we examine the extent of such dialogue response sensitivity in pre-trained language models, conducting a series of experiments with a particular focus on sensitivity to dynamics involving phenomena of at-issueness and ellipsis. We find that models show clear sensitivity to a distinctive role of embedded clauses, and a general preference for responses that target main clause content of prior utterances. However, the results indicate mixed and generally weak trends with respect to capturing the full range of dynamics involved in targeting at-issue versus not-at-issue content. Additionally, models show fundamental limitations in grasp of the dynamics governing ellipsis, and response selections show clear interference from superficial factors that outweigh the influence of principled discourse constraints.

1 Introduction

Competence in language involves understanding complex principles governing relevance of previous content and dynamics of referring back to that content. Certain parts of an utterance are more central and more likely to receive a response than others, and the pragmatic and grammatical rules governing responses in dialogue interact with the nature of the content being responded to. Humans are highly sensitive to these distinctions, and we can expect these sensitivities to be critical for robust models in NLP, and especially for dialogue.

Here we examine sensitivity to these dialogue response dynamics in pre-trained language models (PLMs). PLMs are now used as foundation for nearly every downstream NLP task, including dialogue applications (e.g., Upadhye et al., 2020; Koto et al., 2021). The impressive downstream performance enabled by these models has raised important questions about what types of linguistic competence are being learned during pre-training—and though there is a growing body of work answering aspects of this question, topics of pragmatic and dialogue competence have been relatively understudied. In this paper we focus on addressing this gap, and in particular on understanding the extent to which PLMs develop sensitivity to dynamics governing responses in dialogue. Though these PLMs are not trained to engage in dialogue per se, they can be expected to encounter dialogue during training (in, for instance, novels), so it is not unreasonable to expect that they may learn about such dialogue dynamics along with other linguistic competences. The strength of these models' sensitivity to such dynamics has important implications for robustness in dialogue applications, since a strong grasp of dialogue dynamics in standard PLMs stands to reduce fine-tuning needs and enable more robust downstream behaviors.

We begin with the notion of *at-issueness*. A component of an utterance is considered *at-issue* if it is part of the "main point" of the utterance—this is to be contrasted with side comments or mentions of background knowledge, which are not the main focus of the sentence. As we lay out in Section 3.1, the distinction between at-issue and not-at-issue content of an utterance is reflected directly in the nature of responses to that utterance. We thus examine models' preferences for different responses, to assess whether the preferences reflect understanding of at-issueness and how to respond to it. We find that models show consistent preference to target at-issue (main clause) content, but mixed and overall fairly weak sensitivity when it comes to the full range of dynamics involved with at-issueness.

These assessments of at-issueness sensitivity are also critically reliant on another aspect of dialogue response dynamics: ellipsis. We thus additionally make a closer examination of the extent to which constraints from context dictate models' selection of auxiliary verbs (such as *did, does, would*) in el-

Proceedings of the 29th International Conference on Computational Linguistics, pages 863–874
October 12–17, 2022.

lipsis constructions. We find that although models often favor an auxiliary verb that targets the main clause, they also make frequent errors, and they very rarely favor both of the auxiliary forms that align with the prior context. These results furthermore raise the important possibility that models are highly sensitive to preferences for particular auxiliary verb types, and that this could drive the at-issueness results as well. With this in mind we revisit the at-issueness experiments, and find that, indeed, there are substantial differences in models' preferences depending on the identity of the particular verb that targets the relevant content.

Overall, our results suggest that PLMs have non-trivial gaps in their understanding of response dynamics in dialogue. Our results also indicate certain differences between models: BERT and RoBERTa show strong bias toward selecting responses that target the most recent and/or main clause content, while other models show more reliance on individual auxiliary verb properties. In all cases the results indicate that these PLMs have not yet achieved ideal sensitivity to response dynamics involving at-issueness and ellipsis, and that effectiveness in dialogue will benefit from additional training approaches. We make all datasets and code available for further testing.[1]

2 Related work

Recent years have seen extensive work on analysis of PLMs. Methodologically, some of the most popular analysis paradigms targeting model embeddings have included classification-based probing (e.g., Kim et al., 2019; Zhang et al., 2019) and correlation with similarity judgments (Finkelstein et al., 2001; Gerz et al., 2016; Conneau and Kiela, 2018). Other work has analyzed PLMs by eliciting and analyzing output predictions (Linzen et al., 2016; Goldberg, 2019). Our work here focuses primarily on the latter methodology, examining and comparing model output probabilities—however, our analysis in Section 5.4 uses classification-based probing. Our work also builds on approaches implementing specialized sentence generation systems that produce large annotated datasets (Ettinger et al., 2018; McCoy et al., 2019).

Analyses of PLMs have targeted a variety of types of linguistic competence. In particular, a large body of work has studied the extent to which PLMs capture syntactic and semantic information (Linzen et al., 2016; Peters et al., 2018; Bacon and Regier, 2019; Hewitt and Manning, 2019; Tenney et al., 2019). Less work has addressed the extent to which PLMs show sensitivity to pragmatic and discourse information, as we focus on in this paper. Kurfalı and Östling (2021) study multilingual models in various discourse tasks via zero-shot learning. Pandia et al. (2021) investigate LMs' pragmatic competence to predict discourse connectives. Pitler and Nenkova (2009) report that a supervised classifier is able to identify discourse relations given syntactic features along with connectives. Patterson and Kehler (2013) implement a similar idea and show that classifiers are able to predict the presence of a connective based on shallow linguistic cues. Koto et al. (2021) explore pre-trained language models' capability in capturing discourse level relations. We complement this existing work by branching into new areas of pragmatic and discourse knowledge, examining models' sensitivity to dialogue response dynamics.

Another closely related literature is that in which PLMs, especially transformer LMs, are used for building dialogue systems directly. Le et al. (2019) propose Multimodal Transformer Networks (MTN) for visual-grounded dialogue tasks. Other work investigates topic-driven language models for emotion detection in dialogues (Zhu et al., 2021). Oluwatobi and Mueller (2020) report state-of-the-art performance on dialogue generation using transformer-based models. There are also language models designed for and trained on dialogue or conversation, such as TransferTransfo (Wolf et al., 2019), PLATO (Bao et al., 2020), ConveRT (Henderson et al., 2020), TOD-BERT (Wu et al., 2020), DialoGPT (Zhang et al., 2020), DialogBERT (Gu et al., 2021), and LaMDA (Thoppilan et al., 2022).

Here we focus on clarifying the extent to which PLMs pre-trained in the standard paradigm can develop knowledge of dialogue dynamics prior to any specialized dialogue training. This line of inquiry serves to broaden our general understanding of linguistic competence of standard PLMs, and also has implications for use of these standard PLMs as foundation for further dialogue-specific training.

3 Background

3.1 At-issueness

Our analyses focus on the dynamics that govern responses in dialogue, and aspects of prior utter-

[1] https://github.com/sangheek16/dialogue-response-dynamics

ances that they target. The first notion that we test for in PLMs is sensitivity to "at-issueness." At-issueness refers to content's status as the main point of the utterance—to be contrasted with not-at-issue content, such as side comments and assumed knowledge (see Potts (2005) for a comprehensive overview). Humans are sensitive to which content in an utterance is "at-issue" and which content is not—and this sensitivity is reflected in dialogue response dynamics. Consider the utterance in (1).

(1) The nurse, who has interest in French cuisine, adopted a rescue dog.

If a listener responds to (1) with "No" or "That's not true," they would most likely be objecting to the claim that *the nurse adopted a rescue dog*, since this is the main point (at-issue content) of (1). It is less likely that they would be objecting to the side comment about French cuisine. As a result, a response of "No, he didn't (adopt a rescue dog)," would be natural, while "No, he doesn't (have interest in French cuisine)" would be less so.

This intuition drives a key diagnostic used to distinguish at-issue and not-at-issue content, known as the **Rejection & Peripherality Test** (or the Assent/Dissent Test) (Amaral et al., 2007; Koev, 2013; Syrett and Koev, 2015). The "rejection" component of this test is illustrated in (2). Speaker B_1 replies to Speaker A's utterance with a **rejection** ("No"), and uses the elliptical verb phrase ("did not") that targets the (at-issue) content of the main clause ("The nurse adopted a rescue dog."), for a natural and appropriate response. In contrast, Speaker B_2 rejects the (not-at-issue) content inside the appositive relative clause (ARC), which is less natural (indicated with '#').

(2) a. Speaker A: "The nurse, who has interest in French cuisine, adopted a rescue dog."
b. Speaker B_1: "No, he did not." [Targeting at-issue content]
c. Speaker B_2: #"No, he does not." [Targeting not-at-issue content]

There is, however, a more natural way to object to not-at-issue content: pausing the dialogue to question a side comment or assumption. This is highlighted in the **peripherality test**, which uses phrases like, "Hey, wait a minute" (von Fintel, 2004; Amaral et al., 2007), or "Wait, this is peripheral to your point but..." (Koev, 2018) in order to make targeting not-at-issue content more acceptable. We show an example in (3).

(3) a. Speaker A: "The nurse, who has interest in French cuisine, adopted a rescue dog."
b. Speaker B: "Wait no, he does not (have interest in French cuisine)." [Targeting not-at-issue content]

Human sensitivity to this pattern of relationship between at-issueness and "No" versus "Wait no" response types has been well attested in psycholinguistic experiments. Syrett and Koev (2015) in their Experiment 1 find that when selecting between responses that target not-at-issue content in an embedded clause of a prior utterance, humans are much more likely to choose a response of type "Wait no" (77%) than of type "No" (23%).[2] By contrast, when selecting between responses that target at-issue content in a main clause of a prior utterance, humans' rate of selection of these two response types is roughly even. In their Experiment 2, Syrett and Koev (2015) furthermore show that when selecting among "No" type responses, humans have a strong preference for choosing those that target at-issue content of prior utterances (73.9%) compared to not-at-issue content (26.1%).

Leveraging this knowledge of human sensitivities, we make use of diagnostics modeled after the Rejection & Peripherality Test to examine whether PLMs are also sensitive to these discourse dynamics involving at-issueness and response type. For structuring not-at-issue content, we focus on ARCs as used in the examples above.

3.2 Ellipsis

The examples above make critical use of the grammatical phenomenon of ellipsis: use of abbreviated verb phrases that refer back to previous verb phrases. In ellipsis, typically an auxiliary verb (like *did, does, would*) remains as the verb in the elided verb phrase—for instance: "No, he didn't" is an elided form that could refer back to "The nurse adopted a rescue dog," standing in for the longer phrase "No, he didn't adopt a rescue dog." Ellipsis is another critical component of forming responses in dialogue, and it plays an important prerequisite role in assessing at-issueness. For these reasons, we also test models' grasp of ellipsis in dialogue.

[2]The specific wordings in this experiment were "Hey, wait a minute," and "That's not true."

4 Experiments

4.1 Construction of test items

To enable controlled tests inspired by the structure of the Rejection/Peripherality tests, we generate items using templates. Each input item consists of a sequence of two sentences: (a) a **context sentence**, and (b) a **response sentence**.

We generate the **context** sentences based on a core template of "NOUNPHRASE, who VERBPHRASE1, VERBPHRASE2." This structure includes an embedded ARC (not-at-issue content) and an embedding main clause (at-issue content), as in our example (1) above: *The nurse, who has interest in French cuisine, adopted a rescue dog.* For the noun phrases, we sample from a list of nouns referring to names of occupations (e.g., *nurse, reporter, violinist*). As for verb phrases, to ensure that it would always be unambiguous whether a rejection is targeting the main or the embedded clause, for each item we control the verb phrases of the two clauses such that they will always be targeted by different elided verbs in the response sentence. To do this we create ordered verb pairs from six unique auxiliary verbs *is, was, does, did, has, could*, with the first verb assigned to the embedded clause, and the second assigned to the main clause. This resulted in 30 ($=_6P_2$) unique verb type pairings. We then draw from a list of verb phrases associated with each auxiliary verb: for instance, the verb phrases for the auxiliary verb *does* contain examples such as *has interest in French cuisine*, and *enjoys hiking*; for the verb *did*, the verb phrases include *adopted a rescue dog*, and *met the Illinois governor at a Greek restaurant.* We randomly sample from these verb phrases for each verb pair, with the phrase for the first verb assigned to the VERBPHRASE1 position in the template, within the ARC, and the second verb phrase to the VERBPHRASE2 position, in the main clause. Ten unique sentences were generated for each verb pair, resulting in 300 context sentences (= 30 verb pairs * 10 sentences). Because of our use of the ordered pairs, every auxiliary verb is equally likely to be the correct form for targeting either at-issue content or not-at-issue content of a context sentence.

The **response** sentences then include a "header" consisting of either *No* or *Wait no*, a subject pronoun (sampled randomly to avoid pronoun gender biases), an auxiliary verb targeting either the main clause or embedded clause, and *not*. For example, response sentences might consist of "No, she does not," or "Wait no, he has not." The differences in these headers are the critical factor that impacts whether a response sentence can reasonably target (not-)at-issue content in the context sentence—and the auxiliary verb indicates which verb phrase in the context sentence is being targeted.

In constructing these items, an additional consideration is how to create a setting in which the PLMs may naturally recognize the input as describing a dialogue. We choose to present the items in a format of dialogue resembling that in novels, where entities are described explicitly as uttering the relevant statements. Our final templates thus take a form as shown in example (4) below.[3] We randomly sample the speaker names (e.g., Marco, Ellie) from a list of 400 names, ensuring that no two names repeat in a given item.

(4) Marco said, "The nurse, who has interest in French cuisine, adopted a rescue dog," and Ellie replied, "{No / Wait no}, he {did / does} not."

4.2 Models tested

In all of our experiments below, we test six PLMs. Of these models, five are masked language models (MLMs): BERT (Devlin et al., 2019), RoBERTa (Liu et al., 2019), XLM-RoBERTa (Conneau et al., 2020), DistilBERT and DistilRoBERTa (Sanh et al., 2019). The final model is a causal (unidirectional) language model (CLM): DistilGPT2 (HuggingFace). We used the implementations of these models made available through the HuggingFace Transformers library (Wolf et al., 2020).

5 At-issueness tests

5.1 Header preferences

We begin by asking whether models, like humans, are sensitive to the role of the response "header" ("No" vs. "Wait no") in whether a rejection can naturally target (not-)at-issue content. In line with Experiment 1 in Syrett and Koev (2015), we begin by testing whether models recognize that "No" is an appropriate header when the response auxiliary targets the main clause, but "Wait no" is critically more appropriate when the response auxiliary targets the embedded clause. To do this, for a given

[3] We also tested with a simpler dialogue style: *A: "The nurse, who has interest in French cuisine, adopted a rescue dog." B: "{No / Wait no}, he {did / does} not."* This formatting difference did not significantly change the results in the experiments for which we made this comparison.

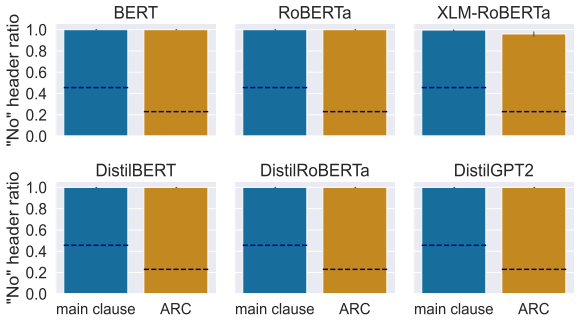

Figure 1: Header selection. The Y-axis indicates the ratio of the "No" header having higher probability than the "Wait no" header given the same verb phrase in the target sentence. The X-axis indicates the target content: 'main clause' condition is when the main clause is targeted; 'ARC' condition is when the embedded clause is targeted. Dashed lines are human performance baseline reported in Syrett and Koev (2015) (Experiment 1). Error bars = 95% Confidence Interval.

item we hold constant the auxiliary verb in the response sentence (e.g., *did/does/has*), and we compare the model probabilities for headers of "No" vs "Wait no." The auxiliary verb for a given item either targets the main clause (at-issue) content of the context sentence, or targets the embedded clause (not-at-issue) content. Since these items are different lengths depending on the choice of header, we compare the conditional log probability of the full sequence, normalized by length, for both MLM and CLM models. For MLMs, we compute pseudo-log-likelihoods, which are obtained by summing the conditional log probabilities of each sentence token (as in Salazar et al. (2020)), and normalizing by number of input tokens.[4]

Figure 1 shows the percentage of items for which the model assigns a higher probability to the sequence with "No" than with "Wait no," separated based on whether the response auxiliary verb targets the main clause (at-issue content) or embedded clause (not-at-issue content). We see that regardless of which clause the response auxiliary targets, models always prefer the "No" header to the "Wait no" header, in a strong contrast with humans' intuition that "Wait no" is much better for targeting not-at-issue content.

5.2 Comparing auxiliary preferences

While the above result casts doubt on models' sensitivity to the relationship between at-issueness and response headers, we might wonder whether "No" is simply too strong or too frequent a response. Additionally, the fact that "No" and "Wait no" are different in lengths raises questions about whether the probability comparisons are fair in the MLMs (in which use of full sentence probabilities is also a bit atypical).

To explore a different angle on this question, we therefore shift to a direct examination of models' preferences for response auxiliaries that target the at-issue content, versus those that target not-at-issue content—and how this is affected by the nature of the "No" versus "Wait no" header. This allows us to examine the MLMs in a more natural setting (assessing probabilities on a single masked position), and also allows us to examine how headers impact models' choices for what contextual content should be targeted.

In these experiments, for MLMs we simply place a [MASK] token at the response auxiliary position and compare auxiliary probabilities at that position:

(5) Marco said, "The nurse, who has interest in French cuisine, adopted a rescue dog," and Ellie replied, "{No / Wait no}, he [MASK] not."

For the CLM, we compare probabilities for the full sequence, with one of two candidate auxiliaries in the target position (as in the previous experiment). The two candidate auxiliaries that we insert are simply the two most relevant: the auxiliary that targets the main clause (at-issue content), and the one that targets the embedded clause (not-at-issue content).

Figure 2 shows the percentage of the time that each model assigns higher probability to the auxiliary targeting the at-issue content, over the auxiliary targeting the not-at-issue content.[5] We see that all models prefer the at-issue-targeting auxiliary at a rate greater than chance, with some models (BERT, RoBERTa) showing preference for targeting the at-issue content almost 100% of the time. The question, then, is whether the use of "Wait no" reduces the rate of targeting the main clause—given that this header allows for targeting of not-at-issue content. A one-sided t-test shows that the selection ratio of the at-issue content is indeed larger with the "reject" header compared to the "wait" header

[4] We use the minicons library (Misra, 2022) for conditional sequence probabilities for all models.

[5] Based on the human performance in Syrett and Koev (2015) (Experiment 2), we could expect a humanlike ratio of selecting at-issue content with the "No" header to be 0.789. This version of the human experiment did not obtain a ratio for the "Wait no" header.

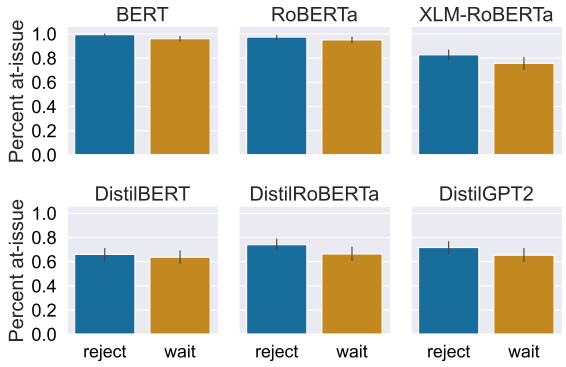

Figure 2: Rejection test. The Y-axis indicates the percentage of instances in which models show preference for responses targeting at-issue content over not-at-issue content. The X-axis indicates type of header in the input sequence: 'Reject' = "No" header; 'Wait' = "Wait no" header. Error bars = 95% Confidence Interval.

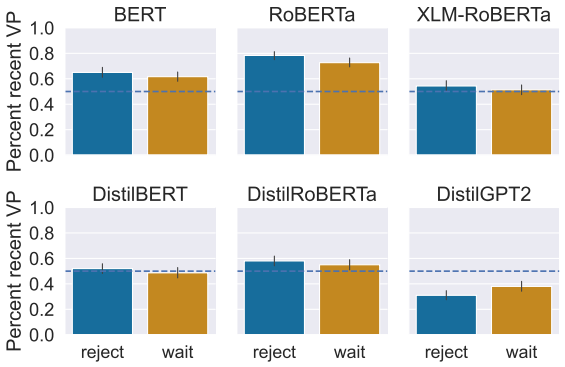

Figure 3: Conjunction test. The Y-axis indicates percentage of instances in which models show preference for responses targeting more recent verb phrases over more distant verb phrases. The X-axis indicates type of header in the input sequence: 'Reject' = "No" header; 'Wait' = "Wait no" header. Dashed lines mark chance level (50%). Error bars = 95% Confidence Interval.

in most of the tested models, at reasonable levels of statistical significance (BERT: t = 2.716, p=0.003; RoBERTa: t = 1.489, p = 0.069; XLM-RoBERTa: t = 2.115, p = 0.017; DistilBERT: t = 0.597, p = 0.275; DistilRoBERTa: t = 2.056, p = 0.02; DistilGPT2: t = 1.671, p = 0.048). This suggests that at least some of the models may have picked up on some relationship between these headers and targeting of at-issue versus not-at-issue content, though it is also clear that these trends are relatively weak.

5.3 Conjunction

The above results show that PLMs exhibit a strong preference for auxiliaries that target the main clause of the context sentence. How should we interpret this preference for targeting the main clause? An immediate question that arises is whether this preference could be due to recency alone: in our items, the verb phrase in the main clause of the context is also always the more recent verb phrase before the response sentence. To investigate this possibility, we modify our items to involve two verb phrases in the context sentence, but with the phrases joined by conjunction (6). This means that both verb phrases are now at-issue, and any preference for one over the other can be attributed to recency.

(6) Marco said, "The nurse has interest in French cuisine and adopted a rescue dog," and Ellie replied, "{No / Wait no}, he [MASK] not."

We again compare model probabilities with each of the two valid candidate auxiliaries. Figure 3 shows the percentage of items for which the models prefer the auxiliary that targets the more recent verb phrase in the context sentence. It is clear from these results that the trend toward targeting main clause content in Figure 2 cannot be attributed to recency alone: a majority of models are now hovering around 50% in targeting the most recent phrase, with DistilGPT2 in fact showing a preference to target the more *distant* phrase rather than the more recent one. BERT and RoBERTa, by contrast, do both show some bias to target the more recent verb phrase—however, this trend is substantially weaker than the trend in Figure 2, indicating that although these models do prefer to target more recent content, they also show a preference for targeting main clause content over and above this recency bias.

5.4 Probing

The results above suggest that at very least, models are sensitive to the fact that embedded clauses (in this case ARCs) have special status in affecting response dynamics—such that models prefer response auxiliaries that target the main clause in the previous context, over and above effects of recency. In this section we briefly confirm that models are sensitive to the differing status of embedded clause content, through a probing experiment testing whether model representations distinguish embedded clause (not-at-issue content) from main clause (at-issue content). To do this, we extract token embeddings from the models and train a classifier to predict whether these tokens are part of a not-at-issue content or an at-issue content. The task is formulated as 3-class classification: contextualized token embeddings from the last hidden layer are used as input for the classifier, and la-

Model	Accuracy (%)
BERT	99.9
RoBERTa	100
XLM-RoBERTa	99.2
DistilBERT	99.4
DistilGPT2	99.5
DistilRoBERTa	100

Table 1: Probing performance using token embeddings from last hidden layers.

bels are generated based on whether the token is 1) part of at-issue content, 2) part of not-at-issue content or 3) neither.[6] Train/test dataset are randomly split for each model, while keeping tokens from the same input sequence together, yielding on average 8,500 training and 4,000 test samples.[7] For this experiment we use a multi-layer perceptron classifier with a single hidden layer of size 50 with ReLU activation, and a softmax layer to generate binary labels. We use a relatively simple classifier following the reasoning of Adi et al. (2017), that this allows examination of how easily extractable information is in these representations.

As shown in Table 1, all models achieve near perfect classification accuracy. The result further supports the conclusion that these models do encode distinctions between content in the main clauses of these sentences and content in embedded clauses—such that the trends in favor of targeting main clause content may be considered to reflect some real sensitivity to contributions of these structural properties to dialogue dynamics.

6 Ellipsis

The response tests above rely on a critical prerequisite: that models understand how to use ellipsis structures like "he didn't" and "she doesn't." In the case of our items, it is specifically the case that there are only two auxiliary verbs that could possibly be appropriate in a given response sentence, because there are only two verb phrases in the context sentence that could be rejected. In this section we take a closer look at whether models' preferences for response auxiliaries reflect these broader discourse constraints on ellipsis.

[6]Tokens counted as "neither" are those like "Marco said" that are used to introduce the dialogue content.

[7]To mitigate impacts of random variation in train/test split across models, we trained the probe for each model three times and averaged the results.

6.1 Ellipsis top one accuracy

We begin by examining the auxiliaries that receive top probability from the models, among the six tested auxiliary verb candidates (i.e., *did*, *does*, *has*, *is*, *was*, and *would*). Specifically, we ask whether the highest-probability response auxiliary selected by the model for a given context is appropriate given the context sentence and header. This test differs from our comparisons above because the previous tests simply compared the two relevant auxiliaries (main clause and embedded clause), without testing whether either of these auxiliaries was assigned the highest probability among all possible auxiliary verbs. Here we count the model as correct if in the case of the "No" header it assigns the highest probability to the auxiliary that targets the main clause, or if in the case of the "Wait no" header it assigns the highest probability to either the main-clause-targeting or embedded-clause-targeting auxiliary (because "Wait no" could also reasonably target the at-issue content).

Figure 4 shows the percentage of the time that the top-ranked auxiliary is among those counted as correct based on the header. We see wide variation in the models' performance on this assessment, with BERT and RoBERTa preferring the correct auxiliary nearly 100% of the time, but distilled models rarely selecting the correct auxiliary as top choice. This suggests at first glance that BERT and RoBERTa have gained a stronger grasp on the relationship of elided auxiliary forms to the previous context—however, it must also be noted that our definition of "correct" favors BERT and RoBERTa because preference for targeting the main clause (which these two models have exhibited) can always be counted as correct. We thus implement a more difficult ellipsis test in the next section.

6.2 Ellipsis top two accuracy

As we describe above, because there are only two verb phrases in each of our context sentences, it is clear that there are only two acceptable auxiliary verb forms that can occur in a given response sentence. To test whether models have a grasp of this constraint, in this section we examine the top two highest-probability auxiliaries, and assess the percentage of the time that these auxiliaries are exactly the two that target the main clause and embedded clause of the context sentence, respectively.

Figure 5 shows the percentage of the time that the two acceptable auxiliaries are the top two

Figure 4: Ellipsis test (top-1 accuracy). The Y-axis indicates the percentage of instances in which models prefer responses targeting the main clause (in the case of the "No" header), or targeting either the main clause or embedded clause (in the case of the "Wait no" header). The X-axis indicates type of header in the input sequence: 'Reject' = "No" header; 'Wait' = "Wait no" header. Error bars = 95% Confidence Interval.

Figure 5: Ellipsis test (top-2 accuracy). The Y-axis indicates the percentage of instances in which models' top two highest-probability auxiliary verbs are the two acceptable auxiliary verbs given context. The X-axis indicates type of header in the input sequence: 'Reject' = "No" header; 'Wait' = "Wait no" header. Error bars = 95% Confidence Interval.

highest-probability auxiliaries for the models. It is clear that the accuracies here are very low—even the most accurate models meet the criterion only 20-30% of the time, suggesting that this category of grammatical/discourse sensitivity is still largely missing from these models.

6.3 Error analysis

To get a better sense of where the models are going wrong in these tests, we perform error analyses for both of the two ellipsis tests. For the top-1 ellipsis test we examine cases where the top auxiliary is not "correct," and for the top-2 test we examine cases where at least one inappropriate auxiliary "intrudes" in the model's top two. In Figures 7 and 8 in the Appendix, we show the distribution of auxiliary verbs that the models prefer among these erroneous cases. We see in these figures particularly substantial interference from more frequent auxiliaries like *did*, *does*, and *is*, suggesting that rather than guiding auxiliary choice based primarily on discourse constraints, the model probability distributions are non-trivially influenced by general frequency of the individual auxiliary verbs in ellipsis.[8]

7 Verb analysis in rejection test

The previous section raises questions about the extent to which these PLMs have a grasp on the basic discourse constraints that govern ellipsis in response utterances—and this ellipsis serves as critical foundation for our at-issueness response tests in Section 5.2. In particular, the error analysis above indicates that model probabilities are influenced in large part by biases in favor of particular frequent auxiliary verbs. In this section we thus return to our at-issueness test to examine behaviors of individual auxiliary verbs separately.

Figure 6 shows the percentage of the time that the models prefer targeting at-issue content, broken down by which auxiliary verb targets the ARC (top row) or the main clause (bottom row). We see that for four of the models, the identity of the auxiliary verb makes a substantial difference: for instance, distilled models strongly prefer to target the main clause if "did" or "does" is the auxiliary that targets the main clause—but if "did" or "does" targets the embedded clause, these models strongly prefer targeting the embedded clause. In other words, the distilled models appear in large part simply to be biased toward preferring "did" or "does" in the response sentence. BERT and RoBERTa do not show as much verb-specific fluctuation, instead preferring to target the main clause content regardless of which auxiliary verb does so. This suggests that these two models have a relatively more robust grasp on use of ellipsis to target particular clauses in previous context. As for sensitivity to impact of headers on response dynamics, within individual verbs we occasionally see a trend such that the "wait" header results in less targeting of the main clause (XLM-RoBERTa with "has" and "would," DistilGPT with "was" and "would," etc.), but for many verbs we see no difference, or even the op-

[8] Among the possible verb phrase ellipsis triggers (e.g., *be*, *has*, *do*, etc.), the auxiliary verb *do* has been reported to be the most frequent (44%), with the auxiliary verb *be* following the next (22%) (Bos and Spenader, 2011).

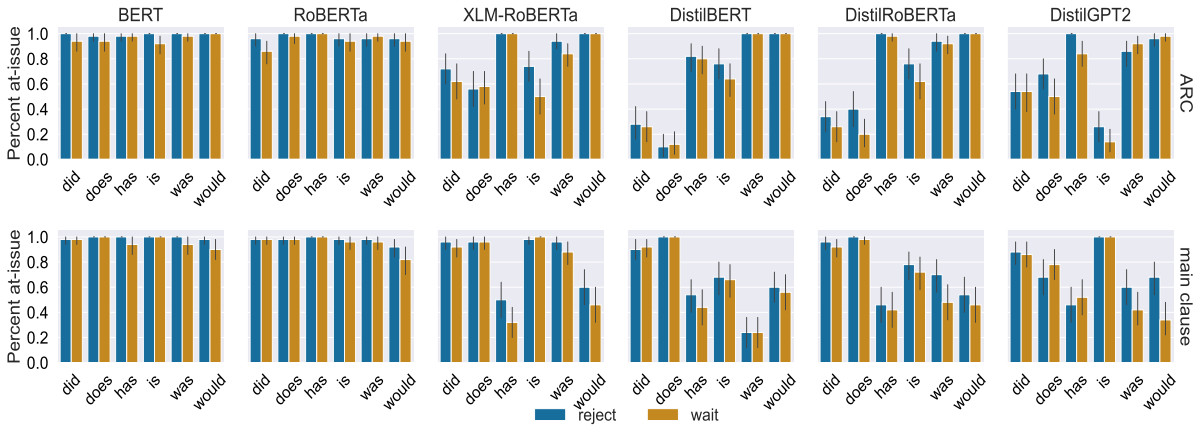

Figure 6: Verb analysis on rejection test. The Y-axis indicates the percentage of instances in which models show preference for responses targeting at-issue content over not-at-issue content. The X-axis indicates the identity of the response auxiliary that can be used to target the selected clause of the context sentence: in the top row the X-axis indicates the auxiliary that targets content inside the ARC (not-at-issue content) of the context sentence, and in the bottom row the X-axis indicates the auxiliary that targets content inside the main clause (at-issue content) of the context sentence. 'Reject' = "No" header; 'Wait' = "Wait no" header. Error bars = 95% Confidence Interval.

posite trend. On the whole, the impact of auxiliary verb identity is for most models much stronger than that of header.

8 Discussion

In this paper we have reported on a series of experiments testing sensitivity of pre-trained language models to dynamics involved in responding to an utterance in dialogue. We focus specifically on at-issueness and ellipsis, and find that models show clear sensitivity to the special status of embedded clauses, and general preference to target main clause content—but they show mixed results in terms of understanding the interaction of response headers with targeting of at-issue versus not-at-issue content. Furthermore, they show certain basic limitations in their grasp of the principles governing ellipsis, with selection of auxiliaries often influenced by superficial frequency factors rather than principled discourse constraints. Our findings also highlight differences between models, with certain models showing strong preference to target main clause content, and others showing stronger fluctuations based on individual auxiliary verbs.

This work highlights potential for improvement in standard PLMs, with respect to discourse sensitivities that have real implications for language competence generally, and for dialogue in particular. The models' sensitivity to special status of embedded clauses is consistent with work indicating sensitivity to syntax in these models (Goldberg, 2019), and the consistency with which BERT and RoBERTa prefer auxiliaries targeting main clause content indicates that these models pick up on some interaction between ellipsis and syntax of previous context. Additionally, the slight impacts of header in Section 5.2 suggest that these models may pick up on the beginnings of a relationship between response types and the types of content that they target. However, the general weakness in sensitivity to headers, failure on many aspects of the ellipsis tests, and interference of superficial factors, indicate clear room for growth in capturing the full range of these discourse dynamics.

From a perspective of downstream dialogue tasks, our findings indicate that discourse competence in standard PLMs is not sufficiently comprehensive to expect that these models can provide a fully robust foundation for dialogue applications. It is possible—though not guaranteed—that training or fine-tuning directly for dialogue could improve the robustness of models' sensitivity to the specific types of response dynamics tested for here. We leave this question for future work.

Acknowledgments

We are grateful to Ming Xiang, Shane Steinert-Threlkeld, Tal Linzen, Najoung Kim, and members of the UChicago CompLing Lab, for valuable comments and discussion. We thank Kanishka Misra in particular for guidance on usage of the minicons library. We also thank three anonymous reviewers for their thoughtful feedback and suggestions.

References

Yossi Adi, Einat Kermany, Yonatan Belinkov, Ofer Lavi, and Yoav Goldberg. 2017. Fine-grained analysis of sentence embeddings using auxiliary prediction tasks. In *5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Conference Track Proceedings*. OpenReview.net.

Patricia Amaral, Craige Roberts, and E. Allyn Smith. 2007. Review of the logic of conventional implicatures by Chris Potts. *Linguistics and Philosophy*, 30(6):707–749.

Geoff Bacon and Terry Regier. 2019. Does bert agree? Evaluating knowledge of structure dependence through agreement relations. *arXiv preprint arXiv:1908.09892*.

Siqi Bao, Huang He, Fan Wang, Hua Wu, and Haifeng Wang. 2020. PLATO: Pre-trained dialogue generation model with discrete latent variable. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 85–96, Online. Association for Computational Linguistics.

Johan Bos and Jennifer Spenader. 2011. An annotated corpus for the analysis of VP ellipsis. *Language Resources and Evaluation*, 45(4):463–494.

Alexis Conneau, Kartikay Khandelwal, Naman Goyal, Vishrav Chaudhary, Guillaume Wenzek, Francisco Guzmán, Edouard Grave, Myle Ott, Luke Zettlemoyer, and Veselin Stoyanov. 2020. Unsupervised cross-lingual representation learning at scale. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 8440–8451, Online. Association for Computational Linguistics.

Alexis Conneau and Douwe Kiela. 2018. SentEval: An evaluation toolkit for universal sentence representations. In *Proceedings of the Eleventh International Conference on Language Resources and Evaluation (LREC 2018)*, Miyazaki, Japan. European Language Resources Association (ELRA).

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, Minneapolis, Minnesota. Association for Computational Linguistics.

Allyson Ettinger, Ahmed Elgohary, Colin Phillips, and Philip Resnik. 2018. Assessing composition in sentence vector representations. In *Proceedings of the 27th International Conference on Computational Linguistics*, pages 1790–1801, Santa Fe, New Mexico, USA. Association for Computational Linguistics.

Lev Finkelstein, Evgeniy Gabrilovich, Yossi Matias, Ehud Rivlin, Zach Solan, Gadi Wolfman, and Eytan Ruppin. 2001. Placing search in context: The concept revisited. In *Proceedings of the 10th International Conference on World Wide Web*, WWW '01, page 406–414, New York, NY, USA. Association for Computing Machinery.

Daniela Gerz, Ivan Vulić, Felix Hill, Roi Reichart, and Anna Korhonen. 2016. SimVerb-3500: A large-scale evaluation set of verb similarity. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 2173–2182, Austin, Texas. Association for Computational Linguistics.

Yoav Goldberg. 2019. Assessing BERT's syntactic abilities. *arXiv preprint arXiv:1901.05287*.

Xiaodong Gu, Kang Min Yoo, and Jung-Woo Ha. 2021. DialogBERT: Discourse-aware response generation via learning to recover and rank utterances. *Proceedings of the Thirty-Fifth AAAI Conference on Artificial Intelligence*, 35(14):12911–12919.

Matthew Henderson, Iñigo Casanueva, Nikola Mrkšić, Pei-Hao Su, Tsung-Hsien Wen, and Ivan Vulić. 2020. ConveRT: Efficient and accurate conversational representations from transformers. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 2161–2174, Online. Association for Computational Linguistics.

John Hewitt and Christopher D. Manning. 2019. A structural probe for finding syntax in word representations. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4129–4138, Minneapolis, Minnesota. Association for Computational Linguistics.

HuggingFace. DistilGPT2.

Najoung Kim, Roma Patel, Adam Poliak, Patrick Xia, Alex Wang, Tom McCoy, Ian Tenney, Alexis Ross, Tal Linzen, Benjamin Van Durme, Samuel R. Bowman, and Ellie Pavlick. 2019. Probing what different NLP tasks teach machines about function word comprehension. In *Proceedings of the Eighth Joint Conference on Lexical and Computational Semantics (*SEM 2019)*, pages 235–249, Minneapolis, Minnesota. Association for Computational Linguistics.

Todor Koev. 2013. *Apposition and the structure of discourse*. PhD dissertation, Rutgers University – New Brunswick.

Todor Koev. 2018. Notions of at-issueness. *Language and Linguistics Compass*, 12(12):1–16.

Fajri Koto, Jey Han Lau, and Timothy Baldwin. 2021. Discourse probing of pretrained language models. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 3849–3864, Online. Association for Computational Linguistics.

Murathan Kurfalı and Robert Östling. 2021. Probing multilingual language models for discourse. In *Proceedings of the 6th Workshop on Representation Learning for NLP (RepL4NLP-2021)*, pages 8–19, Online. Association for Computational Linguistics.

Hung Le, Doyen Sahoo, Nancy Chen, and Steven Hoi. 2019. Multimodal transformer networks for end-to-end video-grounded dialogue systems. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 5612–5623, Florence, Italy. Association for Computational Linguistics.

Tal Linzen, Emmanuel Dupoux, and Yoav Goldberg. 2016. Assessing the ability of LSTMs to learn syntax-sensitive dependencies. *Transactions of the Association for Computational Linguistics*, 4:521–535.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. RoBERTa: A robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692*.

Tom McCoy, Ellie Pavlick, and Tal Linzen. 2019. Right for the wrong reasons: Diagnosing syntactic heuristics in natural language inference. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 3428–3448, Florence, Italy. Association for Computational Linguistics.

Kanishka Misra. 2022. minicons: Enabling flexible behavioral and representational analyses of transformer language models. *arXiv preprint arXiv:2203.13112*.

Olabiyi Oluwatobi and Erik Mueller. 2020. DLGNet: A transformer-based model for dialogue response generation. In *Proceedings of the 2nd Workshop on Natural Language Processing for Conversational AI*, pages 54–62, Online. Association for Computational Linguistics.

Lalchand Pandia, Yan Cong, and Allyson Ettinger. 2021. Pragmatic competence of pre-trained language models through the lens of discourse connectives. In *Proceedings of the 25th Conference on Computational Natural Language Learning*, pages 367–379, Online. Association for Computational Linguistics.

Gary Patterson and Andrew Kehler. 2013. Predicting the presence of discourse connectives. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*, pages 914–923, Seattle, Washington, USA. Association for Computational Linguistics.

Matthew E. Peters, Mark Neumann, Luke Zettlemoyer, and Wen-tau Yih. 2018. Dissecting contextual word embeddings: Architecture and representation. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 1499–1509, Brussels, Belgium. Association for Computational Linguistics.

Emily Pitler and Ani Nenkova. 2009. Using syntax to disambiguate explicit discourse connectives in text. In *Proceedings of the ACL-IJCNLP 2009 Conference Short Papers*, pages 13–16, Suntec, Singapore. Association for Computational Linguistics.

Christopher Potts. 2005. *The logic of conventional implicatures*. Oxford University Press.

Julian Salazar, Davis Liang, Toan Q. Nguyen, and Katrin Kirchhoff. 2020. Masked language model scoring. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 2699–2712, Online. Association for Computational Linguistics.

Victor Sanh, Lysandre Debut, Julien Chaumond, and Thomas Wolf. 2019. DistilBERT, a distilled version of BERT: smaller, faster, cheaper and lighter. *arXiv preprint arXiv:1910.01108*.

Kristen Syrett and Todor Koev. 2015. Experimental evidence for the truth conditional contribution and shifting information status of appositives. *Journal of Semantics*, 32(3):525–577.

Ian Tenney, Patrick Xia, Berlin Chen, Alex Wang, Adam Poliak, R. Thomas McCoy, Najoung Kim, Benjamin Van Durme, Samuel R. Bowman, Dipanjan Das, and Ellie Pavlick. 2019. What do you learn from context? probing for sentence structure in contextualized word representations. In *7th International Conference on Learning Representations, ICLR 2019, New Orleans, LA, USA, May 6-9, 2019*. OpenReview.net.

Romal Thoppilan, Daniel De Freitas, Jamie Hall, Noam Shazeer, Apoorv Kulshreshtha, Heng-Tze Cheng, Alicia Jin, Taylor Bos, Leslie Baker, Yu Du, et al. 2022. LaMBDA: Language models for dialog applications. *arXiv preprint arXiv:2201.08239*.

Shiva Upadhye, Leon Bergen, and Andrew Kehler. 2020. Predicting reference: What do language models learn about discourse models? In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 977–982, Online. Association for Computational Linguistics.

Kai von Fintel. 2004. Would you believe it? the King of France is back! (Presuppositions and truth value intuitions). In Anne Bezuidenhout and Marga Reimer, editors, *Descriptions and beyond: An interdisciplinary collection of essays on definite and indefinite descriptions and other related phenomena*, pages 315–341. Oxford: Oxford University Press.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Remi Louf, Morgan Funtowicz, Joe Davison, Sam Shleifer, Patrick von Platen, Clara Ma, Yacine Jernite, Julien Plu, Canwen Xu, Teven Le Scao, Sylvain Gugger, Mariama Drame, Quentin Lhoest, and Alexander Rush. 2020. Transformers: State-of-the-art natural language processing.

In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 38–45, Online. Association for Computational Linguistics.

Thomas Wolf, Victor Sanh, Julien Chaumond, and Clement Delangue. 2019. TransferTransfo: A transfer learning approach for neural network based conversational agents. *arXiv preprint arXiv:1901.08149*.

Chien-Sheng Wu, Steven C.H. Hoi, Richard Socher, and Caiming Xiong. 2020. TOD-BERT: Pre-trained natural language understanding for task-oriented dialogue. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 917–929, Online. Association for Computational Linguistics.

Yizhe Zhang, Siqi Sun, Michel Galley, Yen-Chun Chen, Chris Brockett, Xiang Gao, Jianfeng Gao, Jingjing Liu, and Bill Dolan. 2020. DIALOGPT : Large-scale generative pre-training for conversational response generation. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics: System Demonstrations*, pages 270–278, Online. Association for Computational Linguistics.

Yuan Zhang, Jason Baldridge, and Luheng He. 2019. PAWS: Paraphrase adversaries from word scrambling. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 1298–1308, Minneapolis, Minnesota. Association for Computational Linguistics.

Lixing Zhu, Gabriele Pergola, Lin Gui, Deyu Zhou, and Yulan He. 2021. Topic-driven and knowledge-aware transformer for dialogue emotion detection. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 1571–1582, Online. Association for Computational Linguistics.

A Appendix: Error analysis on ellipsis and auxiliary preference

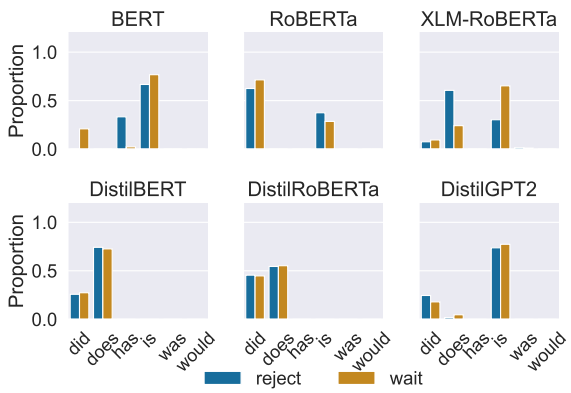

Figure 7: Verb analysis on ellipsis test (top-1). The Y-axis indicates the proportion of the verb appearing as the top-1 prediction even when the corresponding auxiliary verb did not appear in the input sequence. Proportion is calculated by header. The X-axis shows the auxiliary verb used in the target sentence. 'Reject' = "No" header; 'Wait' = "Wait no" header.

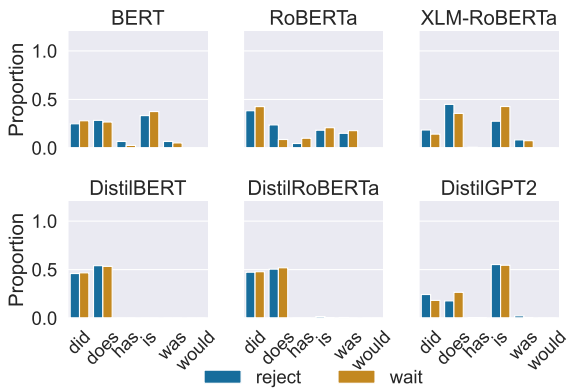

Figure 8: Verb analysis on ellipsis test (top-2). The Y-axis indicates the proportion of the verb appearing as one of top-2 predictions even when the corresponding auxiliary verb did not appear in the input sequence. Proportion is calculated by header. The X-axis shows the auxiliary verb used in the target sentence. 'Reject' = "No" header; 'Wait' = "Wait no" header.

New or Old? Exploring How Pre-Trained Language Models Represent Discourse Entities

Sharid Loáiciga[*1], **Anne Beyer**[*2], **David Schlangen**[2]
[1]CLASP, Dept. of Philosophy, Linguistics and Theory of Science, University of Gothenburg
[2]Computational Linguistics, Department of Linguistics
University of Potsdam, Germany
sharid.loaiciga@gu.se
anne.beyer,david.schlangen@uni-potsdam.de

Abstract

Recent research shows that pre-trained language models, built to generate text conditioned on some context, learn to encode syntactic knowledge to a certain degree. This has motivated researchers to move beyond the sentence-level and look into their ability to encode less studied discourse-level phenomena. In this paper, we add to the body of probing research by investigating discourse entity representations in large pre-trained language models in English. Motivated by early theories of discourse and key pieces of previous work, we focus on the information-status of entities as discourse-new or discourse-old. We present two probing models, one based on binary classification and another one on sequence labeling. The results of our experiments show that pre-trained language models do encode information on whether an entity has been introduced before or not in the discourse. However, this information alone is not sufficient to find the entities in a discourse, opening up interesting questions about the definition of entities for future work.

1 Introduction

In a seminal paper from 1969, Karttunen imagines "a device designed to read a text in some natural language, interpret it, and store the content in some manner, say, for the purpose of being able to answer questions about it". Such a device—considered by him "not a practical idea, for the time being at least"—he says would need to have a particular feature, namely that it "be able to recognize when a novel individual is mentioned in the input text and to store it along with its characterization for future reference."

Now, more than 50 years later, neural models appear to have made such a device a practical idea after all. But do they recognize when a text introduces a new entity into the "universe of discourse", or when, in contrast, the new information concerns a previously introduced one? This is the question that we are asking in this paper.

Following Karttunen's idea and inspired by Prince (1992)'s analysis of information-status, we focus on discourse entities. In particular, we target the task of distinguishing between the status of entity mentions as discourse-new or discourse-old. Considering that discourse entities are central to discourse theories and meaning, we consider that this is an understudied subject in the field. We take a step back from much more specific tasks such as coreference resolution and look at entities being referred to over time. We believe that the new/old distinction, as a simplified form of discourse representation, lets us ask whether language models are able to *keep track* of discourse entities.

Concretely, we build probing models that take as input the representations of pre-trained English language models and predict discourse-new/-old values for all mentions in a text. We present two probing models tackling the task on two different levels of complexity: binary classification and sequence labeling. The first probe takes one entity mention and its preceding context up to that point, and produces a binary decision (discourse-new/-old). It tells us to what extent the context matters for this task. Inspired by the Named Entity Recognition task, the second probe labels each token in a sequence (new/old/outside). It tells us thus whether the entities can be localized in the sequence and labeled with the correct type. A few pieces of research have found first indications about the presence of entity knowledge in pre-trained language models. In particular, Sorodoc et al. (2020) focus on identifying pronoun-antecedent pairs, leaving the question open of whether models have a general notion of entities. Li et al. (2021) work only with synthetic data, which is simple in nature with short sentences and few entities. Last, Gupta and Durrett (2019b) find that pre-trained language model's representations are unable to trace explicit entity state

[*]Shared first authorship.

Proceedings of the 29th International Conference on Computational Linguistics, pages 875–886
October 12–17, 2022.

changes in recipes and physical processes.

Our findings suggest that contextualized pre-trained language model representations generated with a transformer model contain enough discourse information to determine whether an entity is new or old—with results as high as 0.89 F1 in the classification probe, even beyond the case of pronouns. However, that knowledge does not suffice to localize the entity in the sequence—with results as low as 0.51 F1 in the sequence labeling task. [1]

2 Related Work

The intuition that language models implicitly capture and in turn also benefit from entity knowledge has been explored for some time now (Ji et al., 2017; Yang et al., 2017; Schuster and Linzen, 2022, *inter alia*), with recent papers focusing on how to inject some explicit entity representation into the system (Aina et al., 2019; Gupta and Durrett, 2019a, among others). The information-status distinguishes between discourse entities that are newly introduced in the text and those that are already known to the comprehender (Prince, 1992; Kamp and Reyle, 1993). It is a central part in discourse theories as it accounts for the changes in referring expressions used to re-mention discourse entities as they undergo meaning updates as the context evolves.

Our work is most similar to that of Sorodoc et al. (2020) and Li et al. (2021). Both of these papers are interested in probing entity knowledge in pre-trained language models at the discourse level, and they both take a semantic approach in that they are interested in the similarity between different mentions of the same entity at different points in the discourse.

Working with the OntoNotes (Pradhan et al., 2012) coreference corpus, Sorodoc et al. test whether pre-trained language model representations have the morpho-syntactic and semantic knowledge required to match a pronoun with its antecedent. They report results based on pre-trained representations generated with both a Transformer and an LSTM model. Using two baselines i) always referring to the nearest mention, ii) always referring to the most similar (cosine similarity) token, they found that a probe fed with the pre-trained embeddings succeeds at the task of predicting the correct antecedent (75.9% accuracy). An error analysis of the probe with the Transformer representations showed that noun phrases were harder to solve than pronouns (so they focus on the latter). The probe also succeeded in learning agreement, as tested by inserting distractors, but accuracy drops to 53% in hard cases (e.g., when the pronoun and antecedent disagree in gender/number). In our experiments, we go a step further and consider pronouns as well as noun phrase mentions.

Focusing on Transformers, Li et al., on their side, work with the Alchemy (derived from Long et al. (2016)) and Textworld (Côté et al., 2019) datasets. They use the data to construct logical propositions which are then classified into True/False with a binary classifier probe (e.g., *You see an open chest. The only thing in the chest is an old key. The chest contains an apple.* → *True/False*). This data transformation is possible because the original data are constructed short documents with simple sentences and few entities. It should be noted as well that although the probe itself is a low-capacity linear classifier, it needs a proposition embedder and a localizer of the entity in the sequence as additional pipeline components. Their results are measured through accuracy (the aggregation of all propositions with an entity) and they go as high as 94%. Our work is concerned with real world text data instead. As this increases the space of possible propositions, it also requires us to simplify the task, which we will explain in detail in the next sections.

Turning to the approach of probing or diagnostic classifiers (Hewitt and Liang, 2019), these are simple systems trained on the encoded representations of another system. If the probe succeeds in the task for which it is trained—discourse-new/discourse-old in our case, we conclude that the input had the necessary knowledge to solve the task. Research based on probing models mostly relies on classification tasks. This paper is a part of that, but here we additionally use a sequence labeling task. This strategy has precedent in examples such as Ramponi et al. (2020) and Dai et al. (2019a) who use sequence labeling for event and entity extraction, respectively. In the context of probing discourse knowledge, Koto et al. (2021) has used this format for a sentence ranking experiment where the probe was asked to predict the most likely sentence ordering as a sequence.

[1] The code for our experiments is available at: https://github.com/clp-research/new-old-discourse-entities.

(1) [The researchers]$_t$ said [they]$_t$ have isolated [a plant gene that prevents [the production of [pollen]$_i$]$_j$]$_m$. [The gene]$_m$ thus can prevent [a plant]$_y$ from fertilizing [itself]$_y$.

(2) The researchers said they have isolated **[a plant gene that prevents the production of pollen]**. – **new**
The researchers said they have isolated a plant gene that prevents the production of pollen. **[The gene]** – **old**

(3) The researchers said they have isolated a plant gene that prevents the production of **[pollen]**. – **new**
The researchers said they have isolated a plant gene that prevents the production of pollen. The **[gene]** – **old**

(4) [The researchers] said [they] have isolated [a plant gene that prevents the production of pollen] . [The
B-old I-old O B-old O O B-new I-new I-new I-new I-new I-new I-new I-new I-new . B-old
gene] thus can prevent [a plant] from fertilizing [itself] .
I-old O O O B-new I-new O O B-old .

(5) The [researchers] said [they] have isolated a plant [gene] that prevents the [production] of [pollen] . The [gene] thus
O B-old O B-old O O O O B-new O O O B-new O B-new . O B-old O
can prevent a [plant] from fertilizing [itself] .
O O O B-new O O B-old .

Figure 1: Excerpt from one document extracted from the ARRAU corpus. (1) presents the original entities with their embedded mentions, (2) shows the spans data for the classification experiments, while (3) the heads data, (4) shows our formatting into IOB labels for the sequence labeling experiments for spans, and (5) shows our formatting into IOB labels for heads (see Sec. 4.2).

3 Data and Pre-trained Model

We present two sets of probing experiments on the hidden representations of a pre-trained model using real-world text data.

3.1 Data

For our experiments, we use data from the ARRAU corpus (Uryupina et al., 2020), a richly annotated coreference corpus which includes information-status (discourse-new/-old) annotations.[2] We only consider referring mentions, as non-referring mentions do not introduce a discourse entity and thus are not annotated with their information-status. The ARRAU corpus contains annotations for all NPs, including singletons (discourse-new entities that do not occur as antecedents for any other mention). We kept the official split used in the CRAC shared task[3] (Poesio et al., 2018), but discarded the spoken dialog and narrative genres (`trains` and `pears`, respectively) because they are presumably different from the domain of the pre-trained model. We construct two data variants, *spans* and *heads*. The examples per partition can be found in Table 1.

Spans As Fig. 1 shows, the original ARRAU data contains several embedded entities (1), that require some adaptation. We first remove all embedded entities from the maximal entity span containing them. For example, the mentions *i* and *j* have been removed from the outer entity span *m*, *a plant gene that prevents the production of pollen* (4). If an embedded mention in the current sentence is re-mentioned in a subsequent sentence, the remention is retained (despite the first embedded mention not being retained). Then, for the sequence labeling task, we create standard IOB labels: an entity's first token is either B-NEW or B-OLD and subsequent tokens have an I-NEW or I-OLD label. Non-entity tokens are labeled as outside (O).

We expect this version of the data to be easier for the classification probe as it has access to all the information in the span, but harder for the sequence labeling probe, since there is much variability in the length of the mentions. Identifying the span also implies higher order knowledge of the notion of entity. Neural coreference resolution systems, for instance, explicitly learn the representation of spans as the combination of the vectors for the first and last tokens and the head of a mention (Lee et al., 2017).

Heads In this version, we take only the head of each mention. This means that the probing model only needs to label the lexical head or pronoun, as opposed to the complete span of a mention. We obtained the heads by extracting the *min_words* attribute from the manual ARRAU annotations.[4]

[2]The complete corpus is available through the LDC: https://catalog.ldc.upenn.edu/LDC2013T22. Some parts are also available through the ARRAU corpus GitHub, according to the authors: https://sites.google.com/view/arrau/corpus.

[3]http://anawiki.essex.ac.uk/dali/crac18/crac18_shared_task.html

[4]This can either be a pronoun, head noun or proper name.

	Heads			Spans		
	Train	Dev	Test	Train	Dev	Test
New	29,117	1,991	5,084	16,639	1,141	2,812
Old	19,171	1,391	3,672	9,814	714	1,883
Total	48,288	3,382	8,756	26,453	1,855	4,695

Table 1: Data splits for the discourse entity probes.

Here, embedded entities are retained.

3.2 Pre-trained Representations

All of our probes are based on the representations learned by a pre-trained Transformer-XL model (Dai et al., 2019b) (available through the Hugging Face library (Wolf et al., 2020) as TRANSFO-XL-WT103). We focus on this English model first, as it has been used in the closely related work by Sorodoc et al. (2020) and because it is explicitly able to capture long contexts and generate "relatively" coherent long texts (Dai et al., 2019b) by using a recurrence mechanism over cached previous segment states. This counteracts the "context fragmentation" introduced by chopping off contexts at a given length to cope with limited computational capacities.

We extract the last 1024-dimensional hidden state representations for each token by feeding the pre-trained model a whole document at a time, so they are contextualized with the discourse knowledge encoded by the model. In Section 4.3, we show how to extend our probes to other models and compare the results to pre-trained representations extracted from GPT-2 (Radford et al., 2019), which is not specifically adjusted for longer inputs.[5].

3.2.1 Baselines

To interpret the results of our probing models, we compare them to models initialized with static 300-dimensional fastText embeddings (Grave et al., 2018), which we extract word by word using the Python `fastText` module.[6] We further match the results against two simple baselines, a majority baseline that labels every entity as new, and one based on POS tags where only entity mentions that start with a definite article or pronouns are considered old. The intuition behind the second one is that these are easy and frequent cases, which may reveal whether our models simply rely on these linguistic cues.

[5]An extension to different languages, however, would also require gold data with information-status annotations and is not included in this work.

[6]https://fasttext.cc/docs/en/python-module.html

4 Probing Experiments

We first perform a classification task to gather information about the entity representations themselves. The second probing task looks at sequence labeling in order to evaluate whether this information can be also used to detect entity boundaries.

4.1 Classification Task

Previous work on entity status tracking (Gupta and Durrett, 2019b) framed the task as entity classification by pre-extracting the entity in question, thereby not requiring the model to identify what an entity is in the first place. We adapt this task in order to probe whether pre-trained hidden representations contain information about whether an entity is newly introduced into the discourse or if it is a re-mention of an already introduced entity.

4.1.1 Pre-processing

First, we split each text incrementally at each entity mention such that the context contains the n words up to the first entity (and in the next sample the second and so on), and the target contains the respective following entity tokens. As described in Sec. 3.1 we extract either only the heads or the maximal spans of entities as the target. We prepend the $\langle eos \rangle$ token to the context to avoid empty contexts when the first token is (part of) an entity (e.g., in the very first sentence of a document).

Next, we use the pre-trained Transformer-XL representations for every context token, and sum over the extracted target entity vectors to get an entity representation (see lower part of Fig. 2).

4.1.2 Models

Following the baseline models of Gupta and Durrett (2019b), we train an attention-based classifier that computes bilinear attention (Luong et al., 2015) between the entity representation and the context tokens and predicts the entity category (new/old) from the combined result. We use the Attention implementation from `pytorch-nlp` (Petrochuk, 2018) (Eq. 1-4) and a linear layer with a sigmoid function on top (Eq. 5).

$$a_i = h_{c_i}^T * W_{att} * e \quad (1)$$

$$\alpha = softmax(a) \quad (2)$$

$$context = \sum \alpha_i * h_{c_i} \quad (3)$$

$$h_{c,e} = tanh(W_{comb} * [context, e]) \quad (4)$$

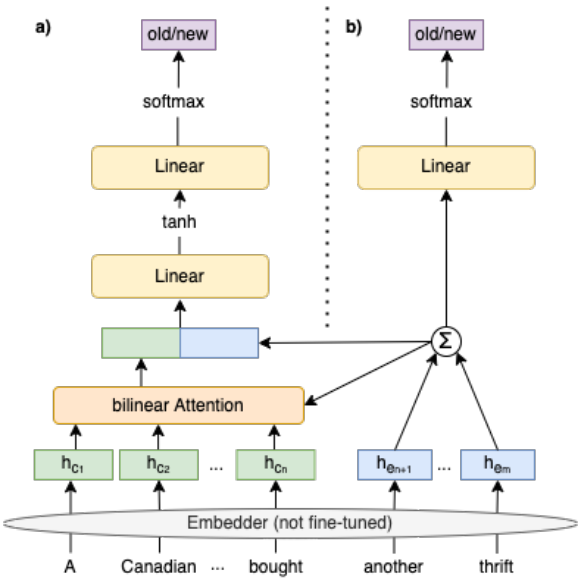

Figure 2: Probing classifier architectures. Embeddings are previously extracted document-wise from the hidden layer of a pre-trained Transformer-XL model (*Embedder*). a) Contextualized classification based on attention between context ($c_1...c_n$) and summed entity ($e_{n+1}..e_m$) representation (Eq. 1-5). b) Entity classification based on summed entity ($e_{n+1}..e_m$) representation alone (Eq. 6).

$$P(y|h_{c_1}, ..., h_{c_n}, e) = sigmoid(W_a * h_{c,e} + b) \quad (5)$$

This model has access to the whole context up to the entity. To gather further insights on the role of the context tokens in terms of what kind of information is already encoded in the entity representation itself, we additionally train a model without context, using only the entity representation to predict its status (Eq. 6).

$$P(y|e) = sigmoid(W_b * e + b) \quad (6)$$

The model architectures are displayed in Fig. 2. Training details and hyperparameters used are provided in Appendix A.

4.1.3 Results

The results of the classification experiments are displayed in Table 2. While the fastText embeddings already yield an improvement over the majority and the POS baseline, the Transformer-XL embeddings yield the best overall results, suggesting that the contextualization adds some useful information on the entity state. Surprisingly, however, there is not much difference between the attention-based and the entity-based models, suggesting that the information required for this task is contained in the pre-trained representations themselves. It is also interesting that taking the whole span into account improves the results for the fastText embeddings, but yields no gain for the Transformer-XL representations. This suggests that due to the contextualization, the necessary information is already encoded in the head representation itself. A detailed discussion follows in Sec. 5.

4.2 Sequence Labeling Task

Inspired by the NER scenario, our second probing model takes the form of a sequence labeling task, whereby each token in the sequence is assigned a categorical label. Discourse entities are a broader category of named entities, so instead of assigning entities a type (e.g., ORGANIZATION, PERSON, TIME, etc.), the probe assigns *new* or *old* labels, following the IOB scheme. An example is shown in Fig. 1.

This framework offers us two advantages in a single task: i) the probe has to localize the entity in the sequence (an additional pre-processing step in previous work), and ii) the probe has to assign a classification label to the entity.

4.2.1 Pre-processing

We slice the extracted hidden vector sequence according to the tokens in the original sentences, in order to feed our probes with examples at the sentence-level. In other words, the hidden representations are extracted based on the whole document, but the probing model labels the document sentence by sentence. The fact that our probes work at the sentence-level can be seen as a safety switch that limits their power, so no further contextualization– beyond that from the original embeddings– occurs. Because this precludes the probe from accessing any embeddings beyond the sentence, any success at predicting discourse-new or discourse-old should come from the entity embeddings themselves. This has the added advantage of easing the computation cost.

4.2.2 Model

The underlying method for our probe is a linear chain conditional random field (CRF) model.[7] The input to the model are sequences of n vectors, where each input h_i is a contextualized vector with size 1024 yielded by the pre-trained language model. This sequence of pre-trained vectors is fed

[7] We took advantage of the freely available implementation at https://github.com/kmkurn/pytorch-crf

	Heads							Spans						
	Discourse New			Discourse Old			Acc.	Discourse New			Discourse Old			Acc.
	Prec.	Rec.	F1	Prec.	Rec.	F1		Prec.	Rec.	F1	Prec.	Rec.	F1	
Probing Transformer-XL														
Attention-based	0.86	0.92	**0.89**	**0.88**	0.80	**0.84**	**0.87**	**0.88**	0.91	**0.89**	**0.86**	**0.81**	**0.83**	**0.87**
Entity-based	**0.87**	0.91	**0.89**	0.87	**0.81**	**0.84**	**0.87**	0.85	**0.92**	0.88	0.86	0.76	0.80	0.85
Baselines fastText 300														
Attention-based	0.76	0.86	0.81	0.76	0.62	0.68	0.76	0.82	0.89	0.85	0.81	0.71	0.75	0.82
Entity-based	0.70	**0.93**	0.80	0.82	0.46	0.59	0.73	0.76	**0.92**	0.83	0.82	0.56	0.67	0.78
Baselines w/o embeddings														
POS-based	0.66	0.83	0.73	0.63	0.40	0.49	0.65	0.74	0.80	0.77	0.66	0.57	0.61	0.71
Majority class	0.58	1.00	0.73	0.00	0.00	0.00	0.58	0.60	1.00	0.75	0.00	0.00	0.00	0.60

Table 2: Average results from five different random seeds of discourse-new vs. discourse-old classification experiments, probing pre-trained Transformer-XL representations versus static fastText embeddings (standard deviation is between 0.00 and 0.04 for all versions), and a POS-based (pronouns and defNP = discourse-old) and majority class (discourse-new) baseline

into an LSTM layer (Eq. 7) and a rectified linear unit activation function (Eq. 8), before being resized (Eq. 9) in order to fit the CRF (Eq. 10). The CRF layer finds the best possible sequence of labels $(y_i, ..., y_n)$ for the entire input sequence.

$$r_i = LSTM(h_i, r_{i-1}) \quad (7)$$

$$o_i = RELU(r_i) \quad (8)$$

$$l_i = W * o_i + b \quad (9)$$

$$p(y_1, ..., y_n | h_1, ..., h_n) = CRF(l_1, ..., l_n) \quad (10)$$

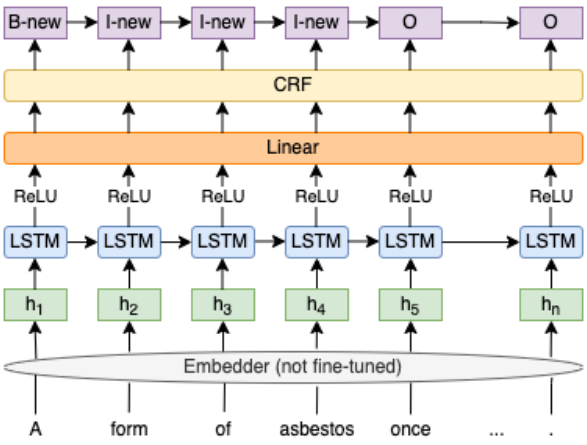

Figure 3: Sequence labeling model. The input to the probe are pre-trained representations h_i from the Transformer-XL model. After the LSTM layer (Eq. 7-8), a linear layer (Eq. 9) is needed to reduce the dimensions of the LSTM output from the hidden size to label size required by the CRF (Eq. 10). The CRF has a choice among 5 labels at each time step.

We present experiments with and without the LSTM layer. We expect a division of labor whereby the CRF learns from the syntactic signal (i.e., I comes after a B), and the LSTM learns the semantic content (i.e., new vs old). Training details and hyperparameters used are given in Appendix A.

4.2.3 Baselines

We build several model versions to estimate the success of our probes: In addition to the majority and POS baselines (cf. Sec. 3.2.1), we build a simple CRF using the Scikit-learn (Pedregosa et al., 2011) compatible CRFsuite (Okazaki, 2007) wrapper (Korobov, 2015) based on simple surface form features. These include whether a token is at the beginning or end of a sequence, whether the token starts with a capital letter, the last three characters of the token, and the last two characters of a token. We also add two versions of our probing models that do not rely on pre-trained representations but train the embeddings from scratch with dimensions 1024 for the comparison to Transformer-XL and 300 for fastText.

We do not compare to a human baseline for two reasons. First, we rely on human annotations of very high quality, for which annotators were asked to identify discourse entities as new/old before being asked to identify antecedents (Uryupina et al., 2020). Second, our main interest is to evaluate whether the pre-trained representations contain information that improves the performance on this task compared to our automatic baselines.

4.2.4 Results

The results for all configurations are reported in Table 3. For computing the scores, we used the SeqEval package (Ramshaw and Marcus, 1995; Nakayama, 2018).

A very clear pattern emerges: Heads are easier to identify than Spans, and discourse-old is easier to predict than discourse-new. It follows that the

	Heads							Spans						
	Discourse New			Discourse Old			Avg.F1	Discourse New			Discourse Old			Avg.F1
	Prec.	Rec.	F1	Prec.	Rec.	F1		Prec.	Rec.	F1	Prec.	Rec.	F1	
Probing Transformer-XL														
LSTM + Linear + CRF	**0.75**	0.79	**0.77**	**0.80**	**0.78**	**0.79**	**0.78**	0.59	0.59	0.59	**0.80**	**0.72**	**0.75**	0.66
Linear + CRF	0.70	0.70	0.70	0.75	0.69	0.72	0.71	0.43	0.38	0.41	0.69	0.63	0.66	0.51
LSTM + Linear + $\text{CRF}_{scratch}$	0.59	0.71	0.64	0.74	0.54	0.62	0.63	0.38	0.39	0.38	0.70	0.52	0.59	0.47
Linear + $\text{CRF}_{scratch}$	0.51	0.59	0.55	0.63	0.47	0.53	0.54	0.27	0.25	0.26	0.55	0.45	0.49	0.35
Baselines fastText 300														
LSTM + Linear + CRF	0.67	0.76	0.71	0.75	0.63	0.68	0.70	0.50	0.50	0.50	0.76	0.60	0.67	0.57
Linear + CRF	0.55	0.63	0.59	0.69	0.45	0.55	0.57	0.25	0.19	0.22	0.63	0.41	0.50	0.33
LSTM + Linear + $\text{CRF}_{scratch}$	0.59	0.70	0.64	0.72	0.54	0.62	0.63	0.40	0.42	0.41	0.70	0.54	0.61	0.49
Linear + $\text{CRF}_{scratch}$	0.53	0.62	0.57	0.65	0.46	0.53	0.55	0.29	0.26	0.28	0.58	0.44	0.50	0.36
Baselines w/o embeddings														
Simple CRF	0.57	0.70	0.62	0.71	0.45	0.55	0.59	0.32	0.28	0.29	0.64	0.44	0.52	0.38
POS baseline	0.65	0.51	0.57	0.51	0.58	0.55	0.56	**0.77**	0.61	0.68	0.62	0.71	0.66	**0.67**
Majority class	0.50	**1.00**	0.74	0.00	0.00	0.00	0.43	0.60	**1.00**	**0.75**	0.00	0.00	0.00	0.45

Table 3: Average results from five different random seeds for all discourse-new vs. discourse-old sequence labeling models, probing pre-trained Transformer-XL representations vs static fastText embeddings and embeddings trained from scratch (standard deviation is between 0.01 and 0.06 for all versions). Baselines also include a simple CRF with surface features, a POS-based (pronouns and defNP = discourse-old) and the majority class (discourse-new) baseline.

combination of span + discourse-new is the most difficult category, and one in which the probes are surpassed by the baselines. Besides, the models with the LSTM yield consistently higher results than the models relying on the CRF only.

When considering the type of input, there is a similar pattern to the classifier. Although the results using static embeddings are better than the baselines, the contextualized Transformer-XL representations present a systematic improvement overall. Interestingly, this improvement is more marked for the Spans and negligible for Heads in additional experiments with a CRF for entity identification only, i.e., without labeling the entities as new or old (cf. Appendix B, Table 6). This keeps with the intuition that identifying the heads is akin to finding nouns, but identifying the relevant spans which are also entities is more complex, involving discourse-level knowledge.

4.3 Extension to other pre-trained Models

To compare our results to another pre-trained model, we also probe GPT-2 (Radford et al., 2019) in the same manner. This requires two adaptation of our approach: GPT-2 uses a different tokenizer, so the alignment of the tokenized version and the labels has to be adapted. More critically, Transformer-XL is optimized to deal with long contexts, whereas GPT-2 can only handle inputs of up to 1024 tokens. Therefore, we create a subset of our data by filtering out all documents longer than a 800 threshold (of items before tokenization). For the results to be comparable, we reran the experiments for Transformer-XL on the same subset. All the results are displayed in Table 4. We first notice that the Transformer-XL results are very similar to those obtained using the full training and test sets. Concerning the GPT-2 model, we notice that its performance is comparable to the Transformer-XL. Thus we believe that other Transformer models will be equally adequate for this task.

5 Discussion and Analysis

In line with existing literature about the presence of entity knowledge in pre-trained language models, we find that the entity knowledge extends to different types of entity mentions extracted from natural data.

The high success of the binary classifier probe demonstrates that classifying an entity as new or old is not challenging, provided that the model has access to the entity representations. Comparable to a coreference resolution system, this model is fed with an aggregated representation comprising all tokens in the mention, so it does not need to locate the entity in the sequence. On the other hand, finding the entities given a sequence is the hard part in our task, as shown by the sequence labeling probe. This model is superior in the Heads version of the data, but less successful with Spans, where the boundaries of each entity must be found. In this task, the simple POS-based baseline yields better overall results than any of our probing models.

Error Analysis: Pronouns and definite noun phrases are two types of mentions with enough reg-

	Heads							Spans						
	Discourse New			Discourse Old			Acc.	Discourse New			Discourse Old			Acc.
	Prec.	Rec.	F1	Prec.	Rec.	F1		Prec.	Rec.	F1	Prec.	Rec.	F1	
CLASSIFICATION PROBE														
Transformer-XL														
Attention-based	0.88	0.90	0.89	0.86	**0.84**	**0.85**	0.87	**0.88**	0.90	**0.89**	**0.86**	0.82	**0.84**	**0.87**
Entity-based	0.87	**0.92**	**0.90**	**0.88**	0.82	**0.85**	**0.88**	0.85	**0.91**	0.88	0.85	0.77	0.81	0.85
GPT-2														
Attention-based	**0.89**	0.90	0.89	0.86	**0.84**	**0.85**	**0.88**	**0.88**	0.88	0.88	0.82	**0.83**	0.83	0.86
Entity-based	**0.89**	0.90	**0.90**	0.86	**0.84**	**0.85**	**0.88**	0.87	0.87	0.87	0.81	0.80	0.80	0.84
SEQUENCE LABELING PROBE														
Transformer-XL														
LSTM + Linear + CRF	0.74	**0.77**	**0.75**	**0.79**	0.77	0.78	0.76	**0.55**	0.55	**0.55**	**0.78**	**0.71**	**0.74**	**0.63**
Linear + CRF	0.70	0.68	0.69	0.75	0.71	0.73	0.71	0.44	0.41	0.43	0.72	0.64	0.68	0.53
GPT-2														
LSTM + Linear + CRF	**0.76**	0.74	**0.75**	0.78	**0.81**	**0.80**	**0.77**	**0.55**	**0.56**	**0.55**	**0.78**	0.69	0.73	0.62
Linear + CRF	0.69	0.67	0.68	0.72	0.68	0.70	0.69	0.42	0.40	0.41	0.71	0.61	0.65	0.51

Table 4: Probing results **on a shortened subset of the data to accomodate GPT-2's maximum input capacity of 1024 tokens**. Results are averaged over five random seeds with a standard deviation between 0.00 and 0.04 for all versions.

ularity (i.e., closed set of forms and determiner *the*) for the models to exploit frequency heuristics. In this sense, comparing with the POS baseline constitutes an interesting case study (Table 5). In general, the fact that these two differ is a sign that our probes are not deterministically exploiting this heuristic. Note however that different genres might have different uses of definite and undefined articles, and that a definite article does not automatically entail a discourse old label.

Going into details, the first thing we observe in Table 5 is that using either the Heads or Spans version results in a similar number of errors, in particular predicting a label when there is no entity to identify (False mention). Interestingly, the sequence labeling probe yields more new than old labels (Old predicted as new), suggesting that it identifies old mentions more confidently than new ones (i.e., when it produces old, there is a high chance that the label really is old). This might explain why despite being the minority class with about 40% of the entities, discourse-old seems easier to predict in the sequence labeling experiments, in particular for the Spans setting (comparing F1 scores). In contrast, the binary classifier does slightly better with the discourse-new class. Another thing we observe is that fewer Spans are left without a prediction than Heads, which intuitively makes sense: it may be harder to say if a bare NP head is referential or not, but easier if the NP is presented with determiners, adjectives, and other modifiers. Last, the category 'Others' comprises mostly errors in detection of boundaries, which are more prevalent in the Spans (for example, *Gulf Resources & Chemical Corp. said it agreed to pay $ 1.5 million [...] regarding [an environmental cleanup] of a defunct smelter the company formerly operated*; gold: B-new I-new I-new, predicted: O O B-new). This category further suggests that finding an entity's boundaries is harder than determining its label.

Inspecting the forms closely shows that most errors correspond to the pronouns *it*, *this*, *that* and *which*, known to be problematic for coreference resolution. For the classifier, we also found that *it* and *that* are amongst the most common errors ($18/424$ in spans, $23/753$ in heads). We also inspected the definite noun phrases, but could not identify any specific pattern in the errors.

Error	Heads	Spans
False mention (gold is O)	1252	1254
No prediction (prediction is O)	138	63
New predicted as old	128	78
Old predicted as new	263	130
Others	138	456
Total	2054	1981

Table 5: Number and type of errors that the sequence labeling probe LSTM + Linear + CRF makes with respect to the POS baseline.

Model Analysis: There is no benefit in using

static embeddings (*Linear + CRF fastText*) vs simple classic features (*simple CRF*) for this task. Comparing the pre-trained representations with the models that were trained from scratch, we observe the following: i) For the most powerful model (the ones with the LSTM layer) the gain of using pre-trained embeddings is 0.15 and 0.19 in average F1 for Transformer-XL and 0.7 and 0.8 for fastText (Heads and Spans, respectively). This shows that, while the embedding size does not have an impact if embeddings are trained from scratch (similar results for 1024 and 300 dimensions), the contextualized Transformer-XL representations contain more useful information for the probing task. ii) When we look at the less powerful probing models (without the LSTM), the differences are 0.17 and 0.16 in average F1 for Transformer-XL, and 0.2 and -0.3 for fastText, showing that the LSTM is necessary to extract any useful information from the static embeddings. Transformer-XL embeddings, on the other hand, already benefit from the contextualization during pre-training, as we see similar improvements as those obtained with the complex model.

The LSTM models do yield better scores overall, suggesting that additional contextualization on the sentence level helps for this task. Collectively, these results signal that document-level contextualization does help to encode the new/old distinction, but not as much as one might have expected. If a model is presented with an entity, determining its status is not hard, even without contextualization. However, finding an entity in a sentence or discourse is challenging, even for powerful Transformer-XL representations. This raises the question of whether pre-trained language models are able to identify entities in the wild.

6 Conclusions and Future Work

In this paper, we have built two probing models for the task of identifying the discourse status of entities as new or old. Our models rely on binary classification and sequence labeling with input representations from a Transformer-XL language model. Our probes:

- have advanced the findings from previous work, showing that the discourse knowledge from pre-trained representations extends to noun phrases found in naturalistic data;
- have found that the pre-trained representations tested do encode the old/new information within the tokens comprising the entity, regardless of the context;
- have also found that localizing the entity within the sentence is difficult, suggesting that identifying referring discourse entities from scratch is hard for this pre-trained model;
- last, have demonstrated that LSTMs are able to further contextualize pre-trained static and contextualized embeddings alike.

Our findings leave interesting questions for future work, in particular, defining what an entity is and what it looks like. In this sense, one could imagine a task where a probe is asked to differentiate between referring and non-referring mentions, a known and hard problem in the context of coreference resolution.

7 Ethical Considerations and Limitations

The models trained in this study are not optimized to solve specific tasks in the best possible way, but to gain insights about the underlying representations and thus the abilities of pre-trained language models, which are sometimes attributed human like language-generation abilities. However, all findings are only applicable to the models under investigation (Transformer-XL and GPT-2) and any claims are specific to English. Reproducing our work requires access to the ARRAU corpus, on which we base all of our experiments.

Acknowledgments

SL was supported by the Swedish Research Council (VR) Grant 2014-39 for the establishment of the Centre for Linguistic Theory and Studies in Probability (CLASP) at the University of Gothenburg. AB & DS were funded by the Deutsche Forschungsgemeinschaft (DFG, German Research Foundation) – Project ID 317633480 – SFB 1287, Project B06. The authors thank Sebastiano Gigliobianco for his help extracting several versions of the data for the experiments, Yves Scherrer for his help developing the tokenizer algorithm, and Manfred Stede for valuable discussions about this work.

References

Laura Aina, Carina Silberer, Ionut-Teodor Sorodoc, Matthijs Westera, and Gemma Boleda. 2019. What

do entity-centric models learn? insights from entity linking in multi-party dialogue. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 3772–3783, Minneapolis, Minnesota. Association for Computational Linguistics.

Marc-Alexandre Côté, Ákos Kádár, Xingdi Yuan, Ben Kybartas, Tavian Barnes, Emery Fine, James Moore, Ruo Yu Tao, Matthew Hausknecht, Layla El Asri, Mahmoud Adada, Wendy Tay, and Adam Trischler. 2019. Textworld: A learning environment for text-based games.

Dai Dai, Xinyan Xiao, Yajuan Lyu, Shan Dou, Qiaoqiao She, and Haifeng Wang. 2019a. Joint extraction of entities and overlapping relations using position-attentive sequence labeling. *Proceedings of the AAAI Conference on Artificial Intelligence*, 33(01):6300–6308.

Zihang Dai, Zhilin Yang, Yiming Yang, Jaime Carbonell, Quoc Le, and Ruslan Salakhutdinov. 2019b. Transformer-XL: Attentive language models beyond a fixed-length context. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 2978–2988, Florence, Italy. Association for Computational Linguistics.

Edouard Grave, Piotr Bojanowski, Prakhar Gupta, Armand Joulin, and Tomas Mikolov. 2018. Learning word vectors for 157 languages. In *Proceedings of the International Conference on Language Resources and Evaluation (LREC 2018)*.

Aditya Gupta and Greg Durrett. 2019a. Effective use of transformer networks for entity tracking. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 759–769, Hong Kong, China. Association for Computational Linguistics.

Aditya Gupta and Greg Durrett. 2019b. Tracking discrete and continuous entity state for process understanding. In *Proceedings of the Third Workshop on Structured Prediction for NLP*, pages 7–12, Minneapolis, Minnesota. Association for Computational Linguistics.

John Hewitt and Percy Liang. 2019. Designing and interpreting probes with control tasks. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 2733–2743, Hong Kong, China. Association for Computational Linguistics.

Yangfeng Ji, Chenhao Tan, Sebastian Martschat, Yejin Choi, and Noah A. Smith. 2017. Dynamic entity representations in neural language models. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 1830–1839, Copenhagen, Denmark. Association for Computational Linguistics.

Hans Kamp and Uwe Reyle. 1993. *From Discourse to Logic. Introduction to Modeltheorectic Semantics of Natural Language, Formal Logic and Discourse Representation Theory*. Kluwer Academic Publishers, Dordrecht.

Lauri Karttunen. 1969. Discourse referents. In *International Conference on Computational Linguistics COLING 1969: Preprint No. 70*, Sånga Säby, Sweden.

Mikhail Korobov. 2015. Sklearn-crfsuite python library. Available online.

Fajri Koto, Jey Han Lau, and Timothy Baldwin. 2021. Discourse probing of pretrained language models. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 3849–3864, Online. Association for Computational Linguistics.

Kenton Lee, Luheng He, Mike Lewis, and Luke Zettlemoyer. 2017. End-to-end neural coreference resolution. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 188–197, Copenhagen, Denmark. Association for Computational Linguistics.

Belinda Z. Li, Maxwell Nye, and Jacob Andreas. 2021. Implicit representations of meaning in neural language models. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 1813–1827, Online. Association for Computational Linguistics.

Reginald Long, Panupong Pasupat, and Percy Liang. 2016. Simpler context-dependent logical forms via model projections. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1456–1465, Berlin, Germany. Association for Computational Linguistics.

Thang Luong, Hieu Pham, and Christopher D. Manning. 2015. Effective approaches to attention-based neural machine translation. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 1412–1421, Lisbon, Portugal. Association for Computational Linguistics.

Hiroki Nakayama. 2018. seqeval: A python framework for sequence labeling evaluation. Software available from https://github.com/chakki-works/seqeval.

Naoaki Okazaki. 2007. CRFsuite: a fast implementation of conditional random fields (CRFs). Available online.

F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. 2011. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830.

Michael Petrochuk. 2018. Pytorch-nlp: Rapid prototyping with pytorch natural language processing (nlp) tools. https://github.com/PetrochukM/PyTorch-NLP.

Massimo Poesio, Yulia Grishina, Varada Kolhatkar, Nafise Moosavi, Ina Roesiger, Adam Roussel, Fabian Simonjetz, Alexandra Uma, Olga Uryupina, Juntao Yu, and Heike Zinsmeister. 2018. Anaphora resolution with the ARRAU corpus. In *Proceedings of the First Workshop on Computational Models of Reference, Anaphora and Coreference*, pages 11–22, New Orleans, Louisiana. Association for Computational Linguistics.

Sameer Pradhan, Alessandro Moschitti, Nianwen Xue, Olga Uryupina, and Yuchen Zhang. 2012. Conll-2012 shared task: Modeling multilingual unrestricted coreference in OntoNotes. In *Joint Conference on EMNLP and CoNLL - Shared Task*, pages 1–40. Association for Computational Linguistics.

Ellen F. Prince. 1992. The ZPG letter: Subjects, definiteness, and information status. In W. Mann and S. Thompson, editors, *Discourse description: Diverse linguistic analysis of a fund-raising text*, pages 223–255. John Benjamins, Amsterdam.

Alec Radford, Jeff Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. 2019. Language models are unsupervised multitask learners. *Technical report, OpenAI*.

Alan Ramponi, Rob van der Goot, Rosario Lombardo, and Barbara Plank. 2020. Biomedical event extraction as sequence labeling. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 5357–5367, Online. Association for Computational Linguistics.

Lance Ramshaw and Mitch Marcus. 1995. Text chunking using transformation-based learning. In *Third Workshop on Very Large Corpora*.

Sebastian Schuster and Tal Linzen. 2022. When a sentence does not introduce a discourse entity, transformer-based models still sometimes refer to it. In *Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 969–982, Seattle, United States. Association for Computational Linguistics.

Ionut-Teodor Sorodoc, Kristina Gulordava, and Gemma Boleda. 2020. Probing for referential information in language models. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 4177–4189, Online. Association for Computational Linguistics.

Olga Uryupina, Ron Artstein, Antonella Bristot, Federica Cavicchio, Francesca Delogu, Kepa Rodriguez, and Massimo Poesio. 2020. Annotating a broad range of anaphoric phenomena, in multiple genres: the ARRAU corpus. *Natural Language Engineering*, 26:95–128.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Remi Louf, Morgan Funtowicz, Joe Davison, Sam Shleifer, Patrick von Platen, Clara Ma, Yacine Jernite, Julien Plu, Canwen Xu, Teven Le Scao, Sylvain Gugger, Mariama Drame, Quentin Lhoest, and Alexander Rush. 2020. Transformers: State-of-the-art natural language processing. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 38–45, Online. Association for Computational Linguistics.

Zichao Yang, Phil Blunsom, Chris Dyer, and Wang Ling. 2017. Reference-aware language models. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 1850–1859, Copenhagen, Denmark. Association for Computational Linguistics.

A Probing experiments details

We trained all models on an NVIDIA GeForce GTX 1080 Ti. Each classification experiment took 10 to 15 minutes, and each sequence labeling experiment took between 7 and 20 minutes.

All the classification models were trained with the BCEWithLogitsLoss and used the Adam optimizer with a learning rate of 0.001 and a batch size of 64. Early stopping was applied based on the loss on the development set.

The training for the sequence labeling probe takes between 1 and 2 epochs using early stopping based on the loss computed on the development set. We use mini-batching with size 64, a hidden size of 256 for the LSTM output, and a learning rate of 0.01 with the Adam optimizer. Dropout of 0.2 is also applied.

B Sequence labeling results without new/old labels

	Heads			Spans		
	Prec.	Rec.	F1	Prec.	Rec.	F1
Probing Transformer-XL						
LSTM + Linear + CRF	0.86	0.91	0.89	0.76	0.76	0.76
Linear + CRF	0.84	0.79	0.81	0.61	0.59	0.60
Baselines fastText 300						
LSTM + Linear + CRF	0.86	0.89	0.87	0.69	0.68	0.69
Linear + CRF	0.75	0.75	0.75	0.45	0.35	0.39

Table 6: Single-run results from sequence labeling experiments for entity identification without predicting their status as new or old.

Dialo-AP: A Dependency Parsing Based Argument Parser for Dialogues

Sougata Saha, Souvik Das, Rohini Srihari
State University of New York at Buffalo
Department of Computer Science and Engineering
{sougatas, souvikda, rohini}@buffalo.edu

Abstract

While neural approaches to argument mining (AM) have advanced considerably, most of the recent work has been limited to parsing monologues. With an urgent interest in the use of conversational agents for broader societal applications, there is a need to advance the state-of-the-art in argument parsers for dialogues. This enables progress towards more purposeful conversations involving persuasion, debate and deliberation. This paper discusses Dialo-AP, an end-to-end argument parser that constructs argument graphs from dialogues. We formulate AM as dependency parsing of elementary and argumentative discourse units; the system is trained using extensive pre-training and curriculum learning comprising nine diverse corpora. Dialo-AP is capable of generating argument graphs from dialogues by performing all subtasks of AM. Compared to existing state-of-the-art baselines, Dialo-AP achieves significant improvements across all tasks, which is further validated through rigorous human evaluation.

1 Introduction

Argumentation is the process of reasoning systematically in support of an idea, action, or theory. It is prevalent in our daily communication and conversations, including online conversations. Since argumentation represents an intrinsic human attribute, the ability of artificial agents (bots) to exhibit this skill can be seen as strong evidence for judging such agents as "human-like". While computational models of argumentation have been investigated (Bench-Capon and Dunne, 2007; Rahwan and Simari, 2009; Atkinson et al., 2017), current progress is impeded by the scarcity of large scale corpora exemplifying use of argumentation and reasoning patterns from discourse. Such corpora are necessary if we are to make progress in training argumentative *conversational agents*. In this paper we experiment with computational argumentation mining (AM) (Mochales and Moens, 2011; Lippi and Torroni, 2016; Lawrence and Reed, 2019) for automatically analyzing discourse at a pragmatics level, and parsing argumentation structures from dialogues. Furthermore, although considerable research can be found in the field of AM, most of the current work has focused on parsing monologues (*micro-level* models), while neglecting dialogues (*macro-level* models) (Bentahar et al., 2010; Grasso, 2002). Here, we aim to fill the research gap in dialogical models for AM by proposing Dialo-AP, a novel end-to-end argument parser for dialogues.

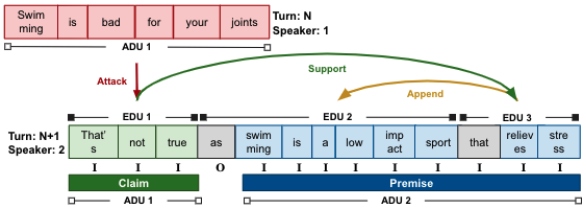

Figure 1: Dependency representation of dialogical arguments, across two turns.

Arguments primarily comprise *claims* and *premises*, with the claim being the central controversial statement of an argument, and the premise provides reasoning by supporting or attacking the claims (Stab and Gurevych, 2014b). End-to-end AM for dialogues generally involves performing text segmentation, component classification, and intra/inter-turn relation detection & classification. Formulating AM as dependency parsing (DP) problem operating at a mixture of elementary discourse unit (EDU) and argumentative discourse unit (ADU) granularity, Dialo-AP is an end-to-end argument parser which takes as input entire conversations, and outputs an argument graph comprised of arguments and relations. Figure 1 illustrates our DP formulation, where speaker 1's utterance consisting of the ADU "Swimming is bad for your joints" is *attacked* by speaker 2's claim "That's not true", which in turn is *supported* by the premise

Proceedings of the 29th International Conference on Computational Linguistics, pages 887–901
October 12–17, 2022.

comprising EDUs “it’s a low impact sport" and “it reduces stress".

Trained on the annotated dialogical *ChangeMyView* (CMV) corpus released by Chakrabarty et al. (2019b), and further utilizing robust pretraining on large scale parallel corpora, followed by fine-tuning on diverse argumentation datasets using *curriculum learning*, Dialo-AP attains significantly higher results compared to internal and external baselines for both in and out-of-domain examples.

2 Related Work

Significant advancements have been made in computational model for AM in recent years. Stab and Gurevych (2014c) implemented a feature engineering based pipelined approach for performing all four sub-tasks of AM, on the Persuasive Essays (PE) corpus (Stab and Gurevych, 2014a), which was further improved by the Integer Linear Programming (ILP) based approach proposed by Persing and Ng (2016). Stab and Gurevych (2017) introduced a larger version of the PE corpus and implemented an ILP constrained pipelined approach for AM. Mirko et al. (2020) improved upon the pipelined approach for AM introduced by Nguyen and Litman (2018), and further implemented a novel graph construction process to create argument graphs. Recently, Bao et al. (2021) proposed a neural transition-based model for component classification and relationship detection, which incrementally builds an argumentation graph by generating a sequence of actions, and can handle both tree and non-tree argumentation structures.

Eger et al. (2017) formulated the tasks of AM as a token level DP, and achieved state-of-the-art performance on the PE dataset, using a neural dependency parser. Inspired by the success of incorporating biaffine classifiers for semantic DP (Dozat and Manning, 2016, 2018), Ye and Teufel (2021) further improved the DP based approach by using biaffine layers, and leveraged pre-trained BERT (Devlin et al., 2018) for richer argument representations. Instead of operating at a word level, Morio et al. (2020) experimented with proposition level AM and used a joint learning framework for jointly performing the tasks of component classification, relation detection and classification. For AM in dialogues, Chakrabarty et al. (2019b) proposed Ampersand (AMP), a computational model for AM in online persuasive discussion forums.

Considerable work has also been done in trying to establish relationships between ADUs and EDUs. Peldszus (2015); Peldszus and Stede (2016); Musi et al. (2018); Hewett et al. (2019) studied the mapping from discourse structure from Rhetorical Structure Theory (RST) to argumentation structures and showed that discourse relations from RST often correlate with argumentative relations.

3 Methods

Formulating AM as dependency parsing, we introduce a multi-task learning (MTL) framework, where unlike existing pipelined approaches, all the sub-tasks are trained together in an end-to-end fashion. Since large scale annotated data for AM from dialogues is scarce, we augment existing monological datasets for our purpose, and leverage pretraining and curriculum learning to learn from the available datasets, before fine tuning on the target CMV corpus.

3.1 Dependency Representation of Arguments

Inspired by the token level dependency representations of arguments in monologues by Eger et al. (2017) and Ye and Teufel (2021), we formulate the following EDU level dependency representation for dialogues (Figure 1), encompassing all the sub-tasks for AM:

Text Segmentation & Component Classification: An argument (ADU) comprises fully or partially overlapping EDUs, which in turn contains labeled argumentative/non-argumentative tokens, using the IO tagging scheme. Identifying such EDUs by predicting the token tags, and further combining consecutive EDUs into ADUs by predicting the existence of relationship constitutes performing the sub-task of *text segmentation*. For example in Figure 1, EDU 2 in turn N+1 partially overlaps with ADU 2, as the token “as” is tagged as “O”, whereas the EDUs 1 and 3 fully overlap with ADU 1 and 2 respectively, which is indicated by all the tokens in the EDUs labelled as “I”. Further, EDU 2 and 3 can be combined using the “Append” relationship to construct ADU 2, after removing the non-argumentative token “as” (marked as O). Each EDU can belong to 1 of 4 classes $\in$ [**M**ajor **C**laim (**MC**), **C**laim (**C**), **P**remise (**P**), **N**on **A**rgument (**NA**)], and predicting the type of a constituent EDU constitutes performing *component classification*.

Intra/Inter-turn Relation Detection & Classification: Within a speaker’s turn, ADUs are related

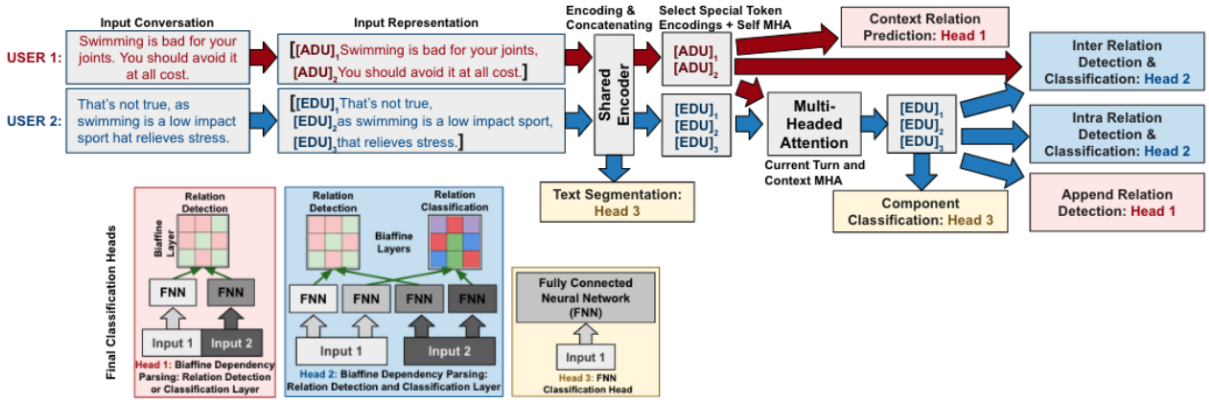

Figure 2: End-to-end Model Architecture.

using "**Sup**port" (**Sup**) or "**Att**ack" (**Att**) relationships which originate from the last EDU of the parent and terminate in the last EDU of the child ADU. Predicting the existence of such relationship between EDUs and further labeling it comprises the sub-tasks *intra-turn relation detection & classification*. Across turns inter-speaker support and attack relationships are established by relating and labeling (using Sup or Att) the last EDU of the source ADU (child) from the current turn, with the target ADU (parent) from the context. *inter-turn relation detection & classification* encompasses determining and labeling such relationships.

Thus, in each turn, ADUs constitute EDUs which are related using directed "Append" edges between consecutive EDUs. Support and attack relationship between ADU pairs are established by associating a labeled directed edge between the last EDUs from the origin and target ADU. The arguments in each turn are further parented by the contextual ADUs by associating a labeled directed edge originating from the contextual ADU, and terminating in the last EDU of the supporting/attacking ADU in the current turn.

3.2 Model Architecture

Dialo-AP is trained in a multi-task setting, where all the sub-tasks share a common encoded representation followed by task-specific layers. Figure 2 illustrates our architecture in detail[1].

Input Representation The model inputs EDU segments for the current turn, which are delimited by a special *[EDU]* token, which not only signifies the start of an EDU span, but is also responsible for encoding and representing its meaning. Turns with length greater than 300 tokens are split into shorter sequences of maximum 300 tokens, while ensuring that an EDU does not span multiple splits. The model inputs a list of ADU spans from prior turns as context, which unlike the current turn is not segmented to an EDU level, and always starts with the start of sequence (*sos*) token.

Encoding We use a shared transformer encoder to independently encode the current turn tokens $\mathrm{S}^{\mathrm{i}}_{\mathrm{curr}}$ and the context $\mathrm{S}^{\mathrm{i}}_{\mathrm{ctx}}$. Lengthy turns which are split into shorter sequences are sequentially encoded and concatenated into a single representation $\mathrm{E}_{\mathrm{curr}}$ (Eqn. 1). In order to preserve the temporal aspect of the text across splits, the position ids of the tokens in each split are cumulatively incremented after every encoding step. The context tokens are also encoded using the same encoder, which yields the context representation $\mathrm{E}_{\mathrm{ctx}}$ (Eqn. 2).

Post encoding, the final context representation $\mathrm{E}^{\mathrm{SOS}}_{\mathrm{ctx}}$ is obtained by selecting and concatenating the sos token encodings of the context ADUs, followed by a multi-headed self-attention layer mha with dropout $drop$ (Eqn. 3, 4). The final current turn representation $\mathrm{E}^{\mathrm{EDU}}_{\mathrm{curr}}$ is constructed by selecting the encodings of the *[EDU]* tokens, followed by a multi-headed self-attention layer and a multi-headed cross-attention between the current turn *[EDU]* token encodings and $\mathrm{E}^{\mathrm{SOS}}_{\mathrm{ctx}}$ (Eqn. 5, 6, 7).

$$\mathrm{E}_{\mathrm{curr}}=\mathrm{concat}(\mathrm{enc}(\mathrm{S}^{\mathrm{i}}_{\mathrm{curr}})|^{\mathrm{n_{splits}}}_{\mathrm{i=1}}) \quad (1)$$

$$\mathrm{E}_{\mathrm{ctx}}=\mathrm{enc}(\mathrm{S}^{\mathrm{i}}_{\mathrm{ctx}})|^{\mathrm{n_{ctx}}}_{\mathrm{i=1}};\ \mathrm{get}(\mathrm{X},\mathrm{idx})=\mathrm{X}[\mathrm{idx},:] \quad (2)$$

$$\mathrm{E}^{\mathrm{SOS}}_{\mathrm{ctx}}=\mathrm{concat}(\mathrm{get}(\mathrm{E}_{\mathrm{ctx}},\mathrm{idx}_{\mathrm{SOS}})) \quad (3)$$

$$\mathrm{E}^{\mathrm{SOS}}_{\mathrm{ctx}}=\mathrm{E}^{\mathrm{SOS}}_{\mathrm{ctx}}+\mathrm{drop}(\mathrm{mha}(\mathrm{E}^{\mathrm{SOS}}_{\mathrm{ctx}},\mathrm{E}^{\mathrm{SOS}}_{\mathrm{ctx}})) \quad (4)$$

[1]Code and data: https://github.com/sougata-ub/dialo-ap

$$\mathrm{E}^{\mathrm{EDU}}_{\mathrm{curr}}=\mathrm{get}(\mathrm{E}_{\mathrm{curr}}, \mathrm{idx}_{\mathrm{EDU}}) \quad (5)$$

$$\mathrm{E}^{\mathrm{EDU}}_{\mathrm{curr}}=\mathrm{E}^{\mathrm{EDU}}_{\mathrm{curr}}+\mathrm{drop}(\mathrm{mha}(\mathrm{E}^{\mathrm{EDU}}_{\mathrm{curr}}, \mathrm{E}^{\mathrm{EDU}}_{\mathrm{curr}})) \quad (6)$$

$$\mathrm{E}^{\mathrm{EDU}}_{\mathrm{curr}}=\mathrm{E}^{\mathrm{EDU}}_{\mathrm{curr}}+\mathrm{drop}(\mathrm{mha}(\mathrm{E}^{\mathrm{EDU}}_{\mathrm{curr}}, \mathrm{E}^{\mathrm{SOS}}_{\mathrm{ctx}})) \quad (7)$$

$$\mathrm{Biaf}(x, y)=x^T \mathrm{U} y + \mathrm{W}(x \oplus y) + b \quad (8)$$

Task Specific Layers We incorporate task-specific layers to perform the final prediction for each sub-task. Illustrated in Figure 2, we use single-layered feed-forward neural networks (Head 3) as the final layer for both text segmentation and component classification, with the input for text segmentation being the concatenated current turn representation $\mathrm{E}_{\mathrm{curr}}$, and $\mathrm{E}^{\mathrm{EDU}}_{\mathrm{curr}}$ for component classification.

Biaffine classifiers (Eqn. 8) are generalizations of linear classifiers, which include multiplicative interactions between two vectors. Since relation detection and classification require performing inference over argument pairs, we implement biaffine dependency parsing (Head 2 and 3 in Figure 2) for both sub-tasks. For intra-relation prediction, the current turn EDU encodings $\mathrm{E}^{\mathrm{EDU}}_{\mathrm{curr}}$ are split into two parts using FNNs–a parent $\mathrm{H}^{\mathrm{i\_parent}}_{\mathrm{intra}}$ and a dependent child $\mathrm{H}^{\mathrm{i\_child}}_{\mathrm{intra}}$ representation, which in turn are passed through a biaffine classifier for detecting or labelling relationships between the EDUs (Eqn. 9, 10). For inter-relation prediction, the parent and child representations $\mathrm{H}^{\mathrm{i\_parent}}_{\mathrm{inter}}$ and $\mathrm{H}^{\mathrm{i\_child}}_{\mathrm{inter}}$ for the biaffine classifier are obtained by passing the context encoding $\mathrm{E}^{\mathrm{SOS}}_{\mathrm{ctx}}$ and current turn EDU encodings $\mathrm{E}^{\mathrm{EDU}}_{\mathrm{curr}}$ through FNNs respectively.

$$\mathrm{H}^{\mathrm{i\_j}}_{\mathrm{k}}=\mathrm{FNN}(x)|x \in (\mathrm{E}^{\mathrm{EDU}}_{\mathrm{curr}}, \mathrm{E}^{\mathrm{SOS}}_{\mathrm{ctx}}), \quad (9)$$
$$\mathrm{i} \in (\mathrm{detect}, \mathrm{label}), \mathrm{j} \in (\mathrm{parent}, \mathrm{child}),$$
$$\mathrm{k} \in (\mathrm{inter}, \mathrm{intra})$$

$$\mathrm{sc}^{\mathrm{i}}_{\mathrm{j}}=\mathrm{Biaf}(\mathrm{H}^{\mathrm{i\_parent}}_{\mathrm{k}}, \mathrm{H}^{\mathrm{i\_child}}_{\mathrm{k}})| \quad (10)$$
$$\mathrm{i} \in (\mathrm{detect}, \mathrm{label}), \mathrm{k} \in (\mathrm{inter}, \mathrm{intra})$$

The sub-tasks of append relation detection and the additional context relationship prediction are performed in a similar way to intra-relationship detection and labeling respectively, where $\mathrm{E}^{\mathrm{EDU}}_{\mathrm{curr}}$ is used for append relation detection, and $\mathrm{E}^{\mathrm{SOS}}_{\mathrm{ctx}}$ for labeling relationships between the context ADUs.

$$\mathcal{L}_{total}=\sum \lambda_x \mathcal{L}^x | x \in (\mathrm{subtasks}) \quad (11)$$

All the sub-tasks are jointly trained end-to-end by minimizing the aggregated interpolated loss $\mathcal{L}_{total}$ (Eqn. 11), where text segmentation, component classification, and inter/intra/contextual relationship labelling are trained by minimizing the cross entropy loss, whereas inter/intra/append relationship detection is trained by minimizing the binary cross entropy loss.

3.3 Pre-training

Since the size of the CMV corpus is small for modern deep learning approaches, we pre-train our parser for most sub-tasks, on large scale noisy labelled corpora.

Component & Intra-Turn Relation Prediction We use the **IMHO corpus** (Chakrabarty et al., 2019a) for pre-training the parser on the sub-tasks of *component classification*, *append relation detection*, and *intra/inter-turn relation detection*. The IMHO corpus comprises 5.5 million opinionated claims from Reddit, which are self-labeled by their authors using the internet acronyms IMO/IMHO (in my (humble) opinion). For example "IMO, Lakers are in big trouble next couple years. Their players are out of contract". We tokenize each example into sentences, and label a sentence as claim only if it contains the acronyms IMO/IMHO, and further associated with a noisy premise by choosing either the preceding or succeeding non-claim sentence, depending on which has a higher levenshtein distance based similarity with the claim tokens. Argument components are further segmented into EDUs, which we detail in Appendix A.2.1. The training targets constitute claim and premise labels, two binary relation matrices for predicting presence of argumentative and "Append" relations between EDUs, and a label matrix for predicting the "support/attack" relationship type.

Inter-Turn Relation Prediction We use the **args.me** (Ajjour et al., 2019), and **QR** corpus (Chakrabarty et al., 2019b) for pre-training the parser on the *inter-turn relation detection and classification* sub-tasks. The args.me corpus comprises 387,606 macro-level arguments crawled from diverse debate portals and already identifies source and target arguments along with pro/con stance labels, which we further convert to support/attack inter-turn relationships. The QR dataset comprises 97,636 pairs of original post and replies from the CMV sub Reddit, where the respondent used Reddit's "quote" feature to reply, signifying an attack relationship on the quoted section from the original post. We combine both the macro-level datasets consisting of source argument, target argument and the inter-argument relationship, and further generate 10,000 random argument pairs with "no re-

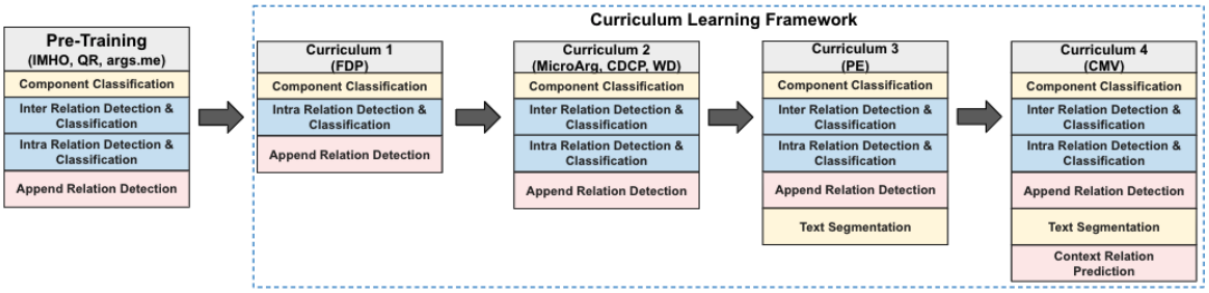

Figure 3: Curriculum Learning Framework.

lationship" labels. For constructing the training relationship label matrices, similar to the IMHO corpus processing we tokenize the source argument into EDUs. However, as discussed in sub-section 3.1, distinct from intra-turn relationship prediction the targets here are ADUs from prior turn, which we identify by using a version of Dialo-AP trained only on the processed IMHO corpus for extracting claims/premises from the context.

3.4 Curriculum Learning (CL) Framework

Computational AM being a relatively new field, suffers from the lack of large scale annotated data, specially for dialogues. Most of the available datasets pertain to distinct and diverse aspects of AM for monologues (Habernal and Gurevych, 2017). Although monologues are distinct from dialogues, parsing both the forms of discourse entails performing comparable tasks like text segmentation, component classification, and intra-turn relationship prediction, which are more local in nature. Further, with a few adaptations, monologues can be augmented to mimic dialogues, and engender noisy training data for inter-turn relationship prediction. Inspired by these observations we design a curriculum learning framework, where we leverage diverse corpora spanning both monologues and dialogues pertaining for incrementally training our parser on all the AM sub-tasks. We define four curriculum spanning six distinct datasets, with the final curriculum comprising fine-tuning on the target CMV dataset. Figure 3 illustrates our curriculum learning framework. Further, all the datasets are pre-processed to conform with our defined dependency representation, which we detail in Appendix A.2.1 and A.2.2.

Curriculum 1 (C1): Component & Intra-Relation Prediction The first curriculum comprises training the component classification and intra-relation prediction sub-tasks, where we leverage the Feedback Prize Dataset[2] (**FDP**), which consists of 15,000 argumentative essays written by U.S students in grades 6-12, and were annotated by expert raters for elements commonly found in argumentative writing.

Curriculum 2 (C2): Component & Intra/Inter-Turn Prediction We train the resulting model from curriculum 1 on the component classification, intra and inter-turn relation prediction sub-tasks by combining training data from three existing corpora: (i) the argumentative microtext corpora (**MicroArg**) (Peldszus, 2015) featuring 112 short argumentative monologues, which were annotated with argumentation structures, following the scheme proposed in Peldszus and Stede (2013); (ii) Consumer Debt Collection Practices (**CDCP**) corpora (Park and Cardie, 2018), comprising 4,931 elementary unit and 1,221 support relation annotations; (iii) Web Discourse (**WD**) corpora (Habernal and Gurevych, 2017), comprising 340 documents annotated with the extended Toulmin model (Toulmin, 2003).

Curriculum 3 (C3): Text Segmentation, Component & Intra/Inter-Turn Relation Prediction Using the Persuasive Essays (Stab and Gurevych, 2017) (**PE**) corpus, we train the resultant model from curriculum 2 on the text segmentation, component classification, intra and inter-relation prediction tasks. PE comprises 402 randomly selected essays from an online forum, which are annotated with argumentation structures. Treating each paragraph as a turn, we convert the dataset to dialogues by considering the major claims as conversation context, and re-labeling the existing "for/against" relationship between a claim and major claim as inter-turn "support/attack" relationship.

Curriculum 4 (C4): Target Dataset Fine-tuning Finally, we fine-tune the resultant model from curriculum 3 on the Change My View (**CMV**) dataset

[2]https://www.kaggle.com/competitions/feedback-prize-2021/overview

(Chakrabarty et al., 2019b), for all sub-tasks. Consisting of 112 discussions, the CMV dataset expands the existing data collected by Hidey et al. (2017) by annotating both inter-turn and intra-turn relations, along with additional argument components. Further, in order to facilitate learning meaningful representations, we introduce an additional task during inter-turn relation prediction, where we train the model to predict relationships that exist between the contextual ADUs.

4 Experiments and Results

We use the CMV data (C4 in Section 3.4) for our experiments, and repeat each experiment five times. In each run, 10% of the data is randomly set aside for testing, and we report the average and maximum results across all runs. More details pertaining to model configuration and setup are shared in Appendix A.1. We train a baseline variant **Baseline-C4** by fine-tuning only on the CMV data (without pre-training and curriculum learning), and use as our internal baseline for model comparison. We further compare our implementation against the following strong external baselines, and report results.

AMP-BERT: Ampersand's (Chakrabarty et al., 2019b) (AMP) BERT baseline using only the pre-trained model without additional fine-tuning.

AMP-Pre-Train: AMP's fine-tuned models *IMHO Context Fine-Tuned BERT* for component classification and intra-turn relation prediction, and the *QR Context Fine-Tuned BERT* for inter-turn relation prediction.

AMP-Best: AMP's best models–*IMHO Context Fine-Tuned BERT* for component classification, *IMHO Context Fine-Tuned BERT+ RST Ensemble* for intra-turn relation prediction, and *IMHO Context Fine-Tuned BERT+ RST Features + Extractive Summarizer* for inter-turn relation prediction. Note that in order to facilitate uniform comparison across experiments, we disregard the variants that incorporate additional rule-based post processing.

AMP-Pre-Train-Re: Since the models reported in AMP are not available for public use, in order to perform qualitative analysis we re-create their fine-tuned models that incorporate pre-training (AMP-Pre-Train), for all common sub-tasks.

4.1 Quantitative Results

Component Classification Table 1 shares our results for component classification, where we report and compare F1 score against external and internal baselines. We observe that although external baselines perform better for identifying non arguments, our implementation (C1+C4) which is trained on curriculum 1 followed by fine-tuning on the target dataset, significantly outperforms all baselines for claim and premise classification, which is more beneficial for constructing argument graphs. We reason that since curriculum 1 constitutes learning only component classification and intra-relation prediction using the fairly large FDP dataset, it is better able to classify components due to lower cognitive load.

We also observe that training the model on all curricula (CL+C4) yields good results, which is not further improved by pre-training. We attribute this to the noisy nature of the pre-training data for component classification.

Inter/Intra Relation Detection Table 1 shares our results for both inter and intra-turn relation detection, where we compare F1, precision, and recall scores across models. In comparison to our internal baseline, for inter-relation detection we observe that training using a curriculum learning framework yields better results, specially for curriculum 2, which constitutes training the relation prediction sub-tasks using the MicroArg, CDCP and WD datasets. Furthermore, we observe higher F1 scores with pre-training, which is further increased by incorporating curriculum learning, yielding the best overall results for inter-relation detection. For intra-turn relation detection we obtain best overall F1 results when incorporating all curricula in our curriculum learning framework. Although pre-training does not seem to be further enhanced the intra-turn relation F1 score, it helps achieve a higher precision model, which can be useful depending on the intended use case of the parser.

For both inter and inter relation detection, we observe that in comparison to AMP based external baselines, our models yield higher precision, lower recall, and higher F1 scores. We also observe that in contrast to our models, which balances precision and recall, all AMP variants generally have disproportionately higher recall compared to their precision. We attribute it to the fact that AMP formulates relationship detection as a binary prediction task between sentence pairs, and constructs all possible permutations of possible sentence pairs from text, which inadvertently spans all arguments, thus increasing recall. On the contrary, our biaffine

Model	Component Classification			Inter-Turn Relation Detection			Intra-Turn Relation Detection		
	Non-Arg	Claim	Premise	Precision	Recall	F1	Precision	Recall	F1
AMP-BERT	71.3	62.0	72.2	8.8	76.0	15.8	12.0	67.0	20.3
AMP-Pre-Train	-	-	-	11.0	75.3	19.1	14.3	69.0	23.7
AMP-Best	75.7	67.1	72.5	16.0	79.4	26.8	16.7	73.0	27.2
AMP-Pre-Train-Re	82.7	63.6	60.9	8.0	52.5	14.0	11.7	77.0	20.4
Baseline-C4 *	70.1 (77.5)	63.7 (71.9)	74.3 (80.4)	23.9 (40.4)	31.2 (39.6)	26.4 (37.7)	17.2 (23.1)	19.3 (29.6)	16.5 (21.5)
C1+C4	72.9 (77.3)	**68.5** (74.7)	**75.5** (82.6)	43.8 (72.9)	33.2 (42.5)	35.2 (42.1)	23.4 (31.8)	28.0 (44.0)	23.0 (29.4)
C2+C4	67.0 (73.4)	59.2 (67.9)	72.1 (76.1)	46.4 (56.9)	30.3 (38.8)	36.4 (44.0)	22.1 (40.3)	13.6 (27.8)	12.9 (24.1)
C3+C4	66.5 (71.8)	63.2 (65.8)	72.4 (76.0)	35.4 (49.6)	29.4 (38.8)	31.2 (40.9)	20.3 (36.9)	25.2 (31.5)	20.7 (26.0)
CL+C4	**73.2** (79.9)	68.4 (73.6)	75.1 (80.7)	33.0 (46.2)	**35.1** (46.3)	33.8 (44.1)	28.0 (37.0)	**34.3** (50.7)	**29.2** (37.9)
Pre-Train+C4	67.4 (77.7)	65.2 (71.8)	73.7 (80.3)	**63.9** (91.1)	27.4 (38.1)	38.4 (53.7)	12.9 (19.1)	16.8 (27.3)	14.1 (20.6)
Pre-Train+CL+C4	70.0 (76.1)	67.0 (71.5)	75.3 (80.1)	55.3 (70.3)	31.3 (43.3)	**39.7** (49.8)	**30.7** (45.7)	26.4 (31.8)	27.2 (37.5)

Table 1: Average and (maximum results) for Component Classification & Intra/Inter-Turn Relationship Detection. For each metric best results w.r.t internal baseline (*) is highlighted in bold, and overall best result underlined.

Model	Inter-Turn Rel. Classify		Intra-Turn Rel. Classify		Text Segmentation		Append
	Support	Attack	Support	Attack	Non-Arg	Arg	
Baseline-C4 *	74.9 (82.3)	66.0 (78.9)	97.8 (99.4)	53.0 (85.7)	77.3 (80.1)	89.1 (90.5)	18.5 (30.5)
C1+C4	76.9 (84.9)	71.3 (81.0)	98.3 (99.2)	50.3 (61.5)	78.1 (81.8)	89.3 (91.4)	59.2 (61.0)
C2+C4	75.1 (84.1)	69.5 (79.5)	98.7 (99.3)	56.3 (75.0)	74.0 (79.1)	87.9 (90.0)	15.7 (25.8)
C3+C4	67.6 (80.3)	60.5 (74.1)	98.2 (99.2)	48.9 (80.0)	75.7 (77.9)	87.9 (89.2)	36.3 (45.4)
CL+C4	**78.7** (85.2)	**78.2** (81.3)	98.9 (99.3)	58.0 (66.7)	**78.5** (83.9)	**89.5** (91.4)	62.5 (65.6)
Pre-Train+C4	75.6 (80.5)	77.3 (82.2)	**99.2** (99.7)	**60.3** (92.3)	75.5 (83.2)	87.9 (90.5)	61.5 (65.5)
Pre-Train+CL+C4	77.2 (86.3)	76.6 (81.5)	98.5 (99.3)	51.4 (80.0)	78.5 (82.4)	89.1 (90.1)	**81.2** (83.3)

Table 2: Average and (maximum) F1 scores for Inter/Intra-Turn Relationship Classification, Text Segmentation and Append relationship prediction. In each column, best result w.r.t baseline (*) is highlighted in bold.

Model	Component		Inter-Turn		Intra-Turn	
	TP-C	TP-A	TP-C	TP-A	TP-C	TP-A
AMP-Re	71.1	75.6	54.3	46.7	67.6	72.7
Dialo-AP	**82.2**	**80.4**	**87.5**	**90.9**	**71.3**	**73.2**

Table 3: Comparison of Human Evaluation Results between AMP-Pre-Train-Re (AMP-Re) and Pre-Train+CL+C4 (Dialo-AP)

dependency parsing based formulation operates at an entire turn level, and facilitates information exchange across EDUs, thus resulting in a balanced score.

Inter/Intra Relation Labeling, Text Segmentation and Append Detection We report our results for inter/intra-turn relationship label prediction and append relationship detection in Table 2. We only perform comparison amongst our implemented variants and inter baselines, due to lack of external baselines for these tasks. For inter-turn relation classification, we observe that incorporating both pre-training and curriculum learning yields better results compared to baseline. Further, training on all curricula yields best F1 score for predicting both support and attack relationship. Similarly for intra-turn relation classification, we observer both pre-training and curriculum learning yields superior results compared to baseline. However, compared to curriculum learning, incorporating pre-training yields better results. We also observe that compared to inter-turn, all model variants are perform intra-turn *support* relationship classification better and *attack* relationship classification worse, compared to inter-turn. Further, for each model, the difference in *support* and *attack* classification F1 scores for intra-turn is higher compared to inter-turn, signifying. We attribute this to the fact that occurrence of *support* relationships are more prevalent within a turn compared to *attack* relationships, which is the converse for inter-turn relationships (Table 4, Appendix A.3).

Although the task of text segmentation is more dependent on linguistic features, we observe best results (Table 2) when training using curriculum learning, proving the efficacy of training using diverse curriculum, in a multi-task learning framework. Also, for detecting *append* relationship between EDUs (Table 2), we observe significantly better results when incorporating pre-training along with curriculum learning, compared to other means.

4.2 Qualitative Results

We further perform human evaluations to ascertain Dialo-AP's usefulness in real world scenarios, where the topic of the discussion might be unre-

stricted. For our purpose, we collect discussion threads from the *ChangeMyView* subreddit on the controversial and out-of-domain topics of *abortion*, *gun violence*, *minimum wage* and *death penalty*, and perform human evaluation on the component classification, inter-turn and intra-turn relation detection subtasks, using a subset of 100 discussions (Table 5, Appendix A.3).

Since our motivation is to create a parser that can identify salient arguments with high precision, we introduced and compared two new metrics: (i) **TP-C:** Mean **T**rue **P**ositive rate at a **C**onversation level, signifying for a conversation, the number of model predictions that are correct on an average. (ii) **TP-A:** Mean **T**rue **P**ositive rate at an over**a**ll level, signifying on an average, the number of model predictions that are correct. We parse each discussion using Dialo-AP variant incorporating pre-training and curriculum learning (Pre-Train+CL+C4), and the recreated version of AMP that leverages pre-training (AMP-Pre-Train-Re), and use Amazon Mechanical Turk (AMT) Human Intelligence Task (HIT) to collect human evaluation on the parsed outputs. In each HIT we provide the entire discussion thread, followed by either the identified arguments with their predicted claim/premise labels, or inter/intra argument pairs predicted by the parsers, and ask the evaluators to mark (by ticking a checkbox) if they think the prediction is correct. Appendix A.4 details the human evaluation task and the AMT collection framework. We compute inter-annotator agreement using 2 evaluators, and observer a Cohen's Kappa score of 0.15, 0.22 and 0.16 for component classification, intra and inter-turn relationship detection respectively, signifying fair amount of agreement (Table 6, Appendix A.4).

Table 3 shares the results from the human evaluation. We observe that our formulation yields significantly better results for all three subtasks, with inter-turn relation detection reporting highest gains compared to the competing model. We attribute this to our robust pre-training and curriculum learning framework, which trains the parser on existing and augmented dialogical data, for identifying inter-turn argumentative relationships. In comparison to itself, leveraging pre-training on the monological IMHO and the noisy QR corpus, AMP performs best on component classification followed by intra-turn and inter-turn relation detection subtasks, whereas Dialo-AP performs best on inter-turn relation detection, followed by component classification and intra-turn relation detection subtasks. Thus, signifying Dialo-AP's better applicability for mining arguments from dialogues.

5 Discussion

Our aim with Dialo-AP was to devise an end-to-end argument parser that can not only enable discourse analysis, but also aid in argument generation by engendering *argument graphs* comprising salient (support-attack) chains of arguments from dialogues. In Figure 4 we illustrate an argument graph generated by Dialo-AP (on the right) on a randomly sampled CMV discussion on *death penalty*, and further compare it against the output by recreated AMP (on the left). Firstly, we observe that although both the parsers yield similar number of components, operating at a combination of EDU and token level, Dialo-AP better identifies and labels argumentative spans. For instance, AMP incorrectly labels the non-arguments *P5* and *P6* as premises. Further, operating at a sentence level, AMP classifies the component *C2* as a single claim, whereas Dialo-AP is correctly able to segment it into 2 components *P1* and *C2*.

Secondly, we observe that since AMP formulates relation prediction as a binary classification problem between argument pairs, it predicts copious relations between components which need not hold, thus hurting it's usefulness for constructing argument graphs. For example, none of the relationships predicted between user 2's argument components hold. On the contrary, not only are the relationships identified by Dialo-AP more meaningful, it also labels the relationship type, making it more useful for constructing argument graphs.

Although Dialo-AP yields better results, it comes with its own set of predicaments. As illustrated on the right, it is unable to relate and utilize all identified argumentative components in the argument graph. For instance, although Dialo-AP identifies the component *P4*, it is unable to establish relationship, and leaves them out of the graph. Further, we observe that both the parsers lack epistemic reasoning capabilities, and could possibly benefit from the use of external knowledge graphs and knowledge bases, which we point as the next possible research direction for argument parsing.

6 Conclusion

In this paper, we present Dialo-AP, a state-of-the-art end-to-end dependency parsing based argument

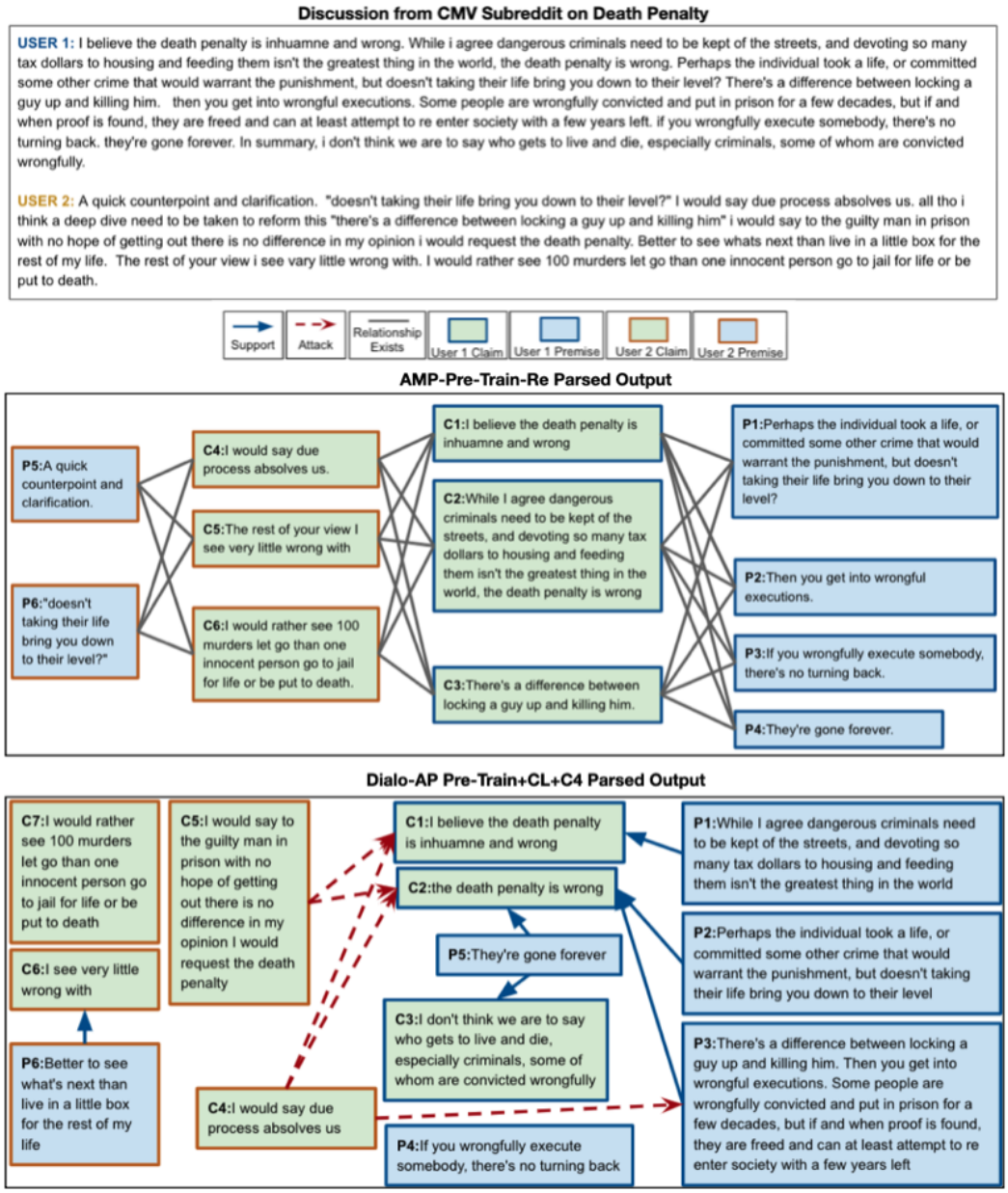

Figure 4: Comparison of parsed CMV post.

parser for parsing arguments from dialogues. Formulating AM as dependency parsing of EDUs and ADUs, and trained in a multi-task setting over diverse curriculum, Dialo-AP is capable of engendering argument graphs from dialogues, by performing all sub-tasks of AM. Dialo-AP's efficacy is exhibited by its superior experimental and human evaluation results, in comparison to strong internal and external baselines. We further discuss Dialo-AP's limitations, and point towards possible next research steps.

Acknowledgements

We thank the anonymous reviewers for providing valuable feedback on our manuscript. This work is partly supported by NSF grant number IIS2214070. The content in this paper is solely the responsibility of the authors and does not necessarily represent the official views of the funding entity.

References

Yamen Ajjour, Henning Wachsmuth, Johannes Kiesel, Martin Potthast, Matthias Hagen, and Benno Stein. 2019. Data Acquisition for Argument Search: The args.me corpus. In *42nd German Conference on*

Artificial Intelligence (KI 2019), pages 48–59, Berlin Heidelberg New York. Springer.

Katie Atkinson, Pietro Baroni, Massimiliano Giacomin, Anthony Hunter, Henry Prakken, Chris Reed, Guillermo Simari, Matthias Thimm, and Serena Villata. 2017. Towards artificial argumentation. *AI magazine*, 38(3):25–36.

Jianzhu Bao, Chuang Fan, Jipeng Wu, Yixue Dang, Jiachen Du, and Ruifeng Xu. 2021. A neural transition-based model for argumentation mining. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 6354–6364, Online. Association for Computational Linguistics.

Trevor JM Bench-Capon and Paul E Dunne. 2007. Argumentation in artificial intelligence. *Artificial intelligence*, 171(10-15):619–641.

Jamal Bentahar, Bernard Moulin, and Micheline Bélanger. 2010. A taxonomy of argumentation models used for knowledge representation. *Artificial Intelligence Review*, 33(3):211–259.

Tuhin Chakrabarty, Christopher Hidey, and Kathy McKeown. 2019a. IMHO fine-tuning improves claim detection. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 558–563, Minneapolis, Minnesota. Association for Computational Linguistics.

Tuhin Chakrabarty, Christopher Hidey, Smaranda Muresan, Kathy McKeown, and Alyssa Hwang. 2019b. AMPERSAND: Argument mining for PERSuAsive oNline discussions. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 2933–2943, Hong Kong, China. Association for Computational Linguistics.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding.

Timothy Dozat and Christopher D. Manning. 2016. Deep biaffine attention for neural dependency parsing.

Timothy Dozat and Christopher D. Manning. 2018. Simpler but more accurate semantic dependency parsing. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 484–490, Melbourne, Australia. Association for Computational Linguistics.

Steffen Eger, Johannes Daxenberger, and Iryna Gurevych. 2017. Neural end-to-end learning for computational argumentation mining. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 11–22, Vancouver, Canada. Association for Computational Linguistics.

Matt Gardner, Joel Grus, Mark Neumann, Oyvind Tafjord, Pradeep Dasigi, Nelson F. Liu, Matthew Peters, Michael Schmitz, and Luke S. Zettlemoyer. 2017. Allennlp: A deep semantic natural language processing platform.

Floriana Grasso. 2002. Towards a framework for rhetorical argumentation. In *EDILOG 02: Proceedings of the 6th workshop on the semantics and pragmatics of dialogue*, pages 53–60.

Ivan Habernal and Iryna Gurevych. 2017. Argumentation mining in user-generated web discourse. *Computational Linguistics*, 43(1):125–179.

Freya Hewett, Roshan Prakash Rane, Nina Harlacher, and Manfred Stede. 2019. The utility of discourse parsing features for predicting argumentation structure. In *Proceedings of the 6th Workshop on Argument Mining*, pages 98–103, Florence, Italy. Association for Computational Linguistics.

Christopher Hidey, Elena Musi, Alyssa Hwang, Smaranda Muresan, and Kathy McKeown. 2017. Analyzing the semantic types of claims and premises in an online persuasive forum. In *Proceedings of the 4th Workshop on Argument Mining*, pages 11–21, Copenhagen, Denmark. Association for Computational Linguistics.

John Lawrence and Chris Reed. 2019. Argument mining: A survey. *Computational Linguistics*, 45(4):765–818.

Marco Lippi and Paolo Torroni. 2016. Argumentation mining: State of the art and emerging trends. *ACM Trans. Internet Technol.*, 16(2).

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Roberta: A robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692*.

Ilya Loshchilov and Frank Hutter. 2017. Decoupled weight decay regularization.

LENZ Mirko, Premtim Sahitaj, Sean Kallenberg, Christopher Coors, Lorik Dumani, Ralf Schenkel, and Ralph Bergmann. 2020. Towards an argument mining pipeline transforming texts to argument graphs. *Computational Models of Argument: Proceedings of COMMA 2020*, 326:263.

Raquel Mochales and Marie-Francine Moens. 2011. Argumentation mining. *Artificial Intelligence and Law*, 19(1):1–22.

Gaku Morio, Hiroaki Ozaki, Terufumi Morishita, Yuta Koreeda, and Kohsuke Yanai. 2020. Towards better non-tree argument mining: Proposition-level biaffine parsing with task-specific parameterization. In

Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, pages 3259–3266, Online. Association for Computational Linguistics.

Elena Musi, Manfred Stede, Leonard Kriese, Smaranda Muresan, and Andrea Rocci. 2018. A multi-layer annotated corpus of argumentative text: From argument schemes to discourse relations. In *Proceedings of the Eleventh International Conference on Language Resources and Evaluation (LREC 2018)*, Miyazaki, Japan. European Language Resources Association (ELRA).

Huy V. Nguyen and Diane J. Litman. 2018. Argument mining for improving the automated scoring of persuasive essays. In *AAAI*.

Joonsuk Park and Claire Cardie. 2018. A corpus of eRulemaking user comments for measuring evaluability of arguments. In *Proceedings of the Eleventh International Conference on Language Resources and Evaluation (LREC 2018)*, Miyazaki, Japan. European Language Resources Association (ELRA).

Andreas Peldszus. 2015. An annotated corpus of argumentative microtexts.

Andreas Peldszus and Manfred Stede. 2013. From argument diagrams to argumentation mining in texts: A survey. *Int. J. Cogn. Inform. Nat. Intell.*, 7(1):1–31.

Andreas Peldszus and Manfred Stede. 2016. Rhetorical structure and argumentation structure in monologue text. In *Proceedings of the Third Workshop on Argument Mining (ArgMining2016)*, pages 103–112, Berlin, Germany. Association for Computational Linguistics.

Isaac Persing and Vincent Ng. 2016. End-to-end argumentation mining in student essays. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 1384–1394, San Diego, California. Association for Computational Linguistics.

Matthew E. Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer. 2018. Deep contextualized word representations.

Iyad Rahwan and Guillermo R Simari. 2009. *Argumentation in artificial intelligence*, volume 47. Springer.

Christian Stab and Iryna Gurevych. 2014a. Annotating argument components and relations in persuasive essays. In *Proceedings of COLING 2014, the 25th International Conference on Computational Linguistics: Technical Papers*, pages 1501–1510, Dublin, Ireland. Dublin City University and Association for Computational Linguistics.

Christian Stab and Iryna Gurevych. 2014b. Identifying argumentative discourse structures in persuasive essays. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 46–56, Doha, Qatar. Association for Computational Linguistics.

Christian Stab and Iryna Gurevych. 2014c. Identifying argumentative discourse structures in persuasive essays. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 46–56.

Christian Stab and Iryna Gurevych. 2017. Parsing argumentation structures in persuasive essays. *Computational Linguistics*, 43(3):619–659.

Stephen E Toulmin. 2003. *The uses of argument*. Cambridge university press.

Yizhong Wang, Sujian Li, and Jingfeng Yang. 2018. Toward fast and accurate neural discourse segmentation. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 962–967, Brussels, Belgium. Association for Computational Linguistics.

Yuxiao Ye and Simone Teufel. 2021. End-to-end argument mining as biaffine dependency parsing. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*, pages 669–678, Online. Association for Computational Linguistics.

A Appendix

A.1 Experiment Setup

We use Roberta (base) (Liu et al., 2019) as the base encoder, and increase its embedding layer to accommodate the special *[EDU]* token. The size of positional embedding layer is increased to 2500. Two layers comprising four attention heads are used for MHA, where the MHA result in each layer is sum pooled with the residual output while applying dropout with 0.1 probability to the MHA result. The hidden size of the FNNs in the biaffine layer is set to 600. An interpolation factor of 0.4 each is used for aggregating the inter and intra-turn relation prediction losses, 0.1 for component classification, and 0.01 for the additional contextual relationship loss prediction. The remaining factor of 0.09 is split equally among text segmentation, append relation prediction, and inter/intra-turn relation labeling. Further, a weight of 3.0 is applied to positive examples during computing binary cross entropy loss for inter and intra-turn relation prediction. All models are trained with a learning rate of 1e-5 for 15 epochs and optimised using AdamW (Loshchilov and Hutter, 2017), with early stopping if the validation loss doesn't reduce for 2 epochs. We repeat each experiment five times and report

the average and maximum results across all runs. During inference, a threshold of 0.5 is used for relationship detection, which is lowered to 0.2, for parsing the out-of-domain samples for human evaluation.

A.2 Pre-processing Steps for Dependency Representation

A.2.1 EDU Segmentation

Depending on the volume of data that needs to be processed, we incorporate the following two segmentation strategies for segmenting text into EDUs: (i) **Neural Segmentation:** For low volume data we use the Bi-LSTM-CRF based discourse segmenter by Wang et al. (2018). (ii) **Rule Based Segmentation:** For larger volumes of data, we use a rule based discourse segmenter, where we segment on encountering the following punctuation :".", "?", "!", ",", ";", and further use a pre-defined set of 113 commonly used discourse markers for finer segmentation (example: "however", "in conclusion", "besides", etc.).

We use the rule based segmentation for preprocessing the large scale IMHO, args.me, FDP, and QR corpus, whereas the neural segmentation is used for segmenting the PE, WD and CMV corpus. Further, we always resort to neural segmentation during inference. Due to it's relatively shorter argument length, for CDCP corpus we treat each proposition as an EDU, and for MicroArg each sentence is considered to be an EDU.

A.2.2 Dataset Specific Pre-processing

IMHO Corpus: We enforce a minimum length of 10 and a maximum length of 300 tokens for each segment. Further, we remove examples whose tokens are split into more than 25 segments. Further, while associating claims with noisy premise, by default the succeeding sentence is chosen as premise if its similarity score is within a margin of 10% compared to the preceding sentence.

arge.me & QR corpus: For args.me, we consider a "pro" stance as "support", and "con" as "attack". For QR, all relationships are considered to be "attack".

FDP Corpus: We remap "Position"→"Major Claim", ("Claim", "Counterclaim", "Rebuttal", "Concluding Statement")→"Claim", and "Evidence"→"Premise". We use textual entailment to associate a claim with a premise. For each claim we construct a set of four candidate premises: two preceding and two following. Using AllenAI's (Gardner et al., 2017) ELMO (Peters et al., 2018) based Textual Entailment we select the premise with the highest entailment score above a threshold of 0.7, as the most likely connected premise, else select the premise immediately following the claim.

MicroArg Corpus: We remap "NA"→"No Relation", ("sup", "exa", "add", "pro")→"Support", and ("reb", "opp")→"Attack". Further, since MicroArg constitutes monologues, we consider the prompt as the previous turn, and convert the monologues to dialogues.

CDCP Corpus: Here we make an assumption and only mark propositions as claim if they have any associated evidence. Else, it's tagged as premise.

WD & PE Corpus: For the WD corpus, we remap "Backing", "Rebuttal" and "'Refutation" to premise. For the PE dataset, we derive dialogues from each paragraph by treating the "Major Claims" (or the essay prompt if major claim is not present) as prior conversation context.

CMV: We remap ("support", "agreement", "partial_agreement", "understand")→"Support", and ("rebuttal_attack", "partial_attack", "rebuttal","undercutter_attack", "partial_disagreement", "disagreement", "undercutter", "attack")→"Attack".

A.3 Additional Stats

Relation	Support	Attack
Intra	96.1	3.9
Inter	45.0	55.0

Table 4: Percentage distribution of Support and Attack Relationships for Inter and Intra-Turn Relations.

Topic	Search Keywords	Count Discussions
abortion	abortion, foeticide	53
gun control	gun control, own gun, second amendment, gun violence, ban gun	19
death penalty	death penalty, capital punishment	18
minimum wage	minimum wage	10

Table 5: Topic distribution of out-of-domain examples collected from CMV.

A.4 Amazon Mechanical Turk Annotations

We leveraged Amazon Mechanical Turk (AMT) in order to collect human evaluations on the model

Task	% Agreement	Cohen's Kappa	Krippendorff's α	N Agreements	N Disagreements	N Cases	N Decisions
Inter-Turn Relation Detection	57.1	0.16	0.13	12	9	21	42
Intra-Turn Relation Detection	57.1	0.22	0.1	12	9	21	42
Component Classification	68.0	0.15	0.14	17	8	25	50

Table 6: Inter-Annotator Agreement of Human Evaluations

generated parsed outputs. We set up human intelligence task (HIT) in the AMT platform, with two evaluators per example and each task worth $0.01. The evaluators were provided with clear instructions on what to annotate and how to annotate the examples, along with a few worked out examples, which are illustrated as screenshots in Figures 5 and 7. The tasks comprised of reading a conversation context, and determining if the presented claim/premise labels are true for component classification (Figure 6), or if the presented argument pairs are valid for inter and intra-turn relation prediction (Figure 8).

In order to ensure quality of annotations, a random portion of the examples presented to each annotator would not be related to the provided conversation, and would have to be marked as "Not in Conversation". Any annotations that failed the quality check were discarded. Further, we discarded annotations which were quickly submitted (less than 2 minutes of work time), and also removed samples where the evaluators missed unchecking the checkboxes, resulting in ambiguity.

This task requires you to read a conversation between multiple participants, and then determine if the claim, premise and non argument labels associated with each of 5 extracted phrase from the conversation are correct.

Background

A brief background on identifying arguments, claims and premises from text. Please read through this brief background in order to understand how to perform the annotation task.

An argument consists of several statements. In its simplest form, it includes a claim that is supported by at least one premise.
Claim: The claim represents a controversial statement which the author tries to persuade the reader of. It is usually a proposition or assumption and should not be accepted by the reader without additional support. This characteristic distinguishes arguments from explanations where the conclusion is a true statement that is not arguable (e.g. an event that happened in the past).
Premise: The premise, underpins the plausibility of the claim, and is usually added for persuading the reader of the claim. Considering the simplest form of an argument, a premise can be seen as a justification for the claim, whereas more complex argumentation structures can also include premises that aim at refuting a claim.
NonArgument: Any other phrases in the conversation which does not belong to a claim or premise are termed as non arguments.

Below we list 3 examples identifying claims and premises from text:

1. *[An advanced gun background check should become routine in all gun sales]$_1$ because [it will prevent gun rampages]$_2$.*
   In this example there are two argument components, where the phrase **1** is the claim, and phrase **2** is the premise, which provides iustification for the first one.
2. *First of all, [people cannot predict their own future or know what will happen tomorrow]$_1$. [The world is full of disasters such as wars, pollution, famine, drought, starvation, natural disasters and diseases]$_2$. So [it is just a big mistake to have children]$_3$.*
   In this example there are three argument components, where the phrase **3** is the claim, and the phrases **1** and **2** are the premises, which provides justification for the claim.
3. *Furthermore, [it is a very heavy psychological and physical burden to have children]$_1$. [A mother carries her baby in her womb for 9 months and 10 days and then the baby torments her during and after the birth]$_2$. [There is no peace, no silence or no sleep at home]$_3$. On the other hand, [the father has to work hard and earn more money]$_4$ because [the baby comes with his expenses]$_5$.*
   In this example there are five argument components, where the phrase **1** is the claim, and the phrases **2**, **3** and **4** are the premises, which provides justification for the claim. Phrase **5** is also a premise, and it provides justification for the premise in phrase **4**.

Task Instructions: **Read the below conversation between multiple participants, and mark whether the claim, premise and non argument label associated with each of 5 extracted phrase from the conversation are correct. Note that:**

1. For each extracted phrase there are 3 options:
   1. Correct: Indicating that the associated label (claim, premise non argument) with the extracted phrase from the conversation is correct.
   2. Inorrect: Indicating that the associated label (claim, premise non argument) with the extracted phrase from the conversation is NOT correct.
   3. Not In Conversation: Out of the 5 presented phrases, there is at least 1 phrase which is not from the current converation. Such a phrase should be marked as "Not In Conversation".
2. There can be multiple claims and premises in the conversation.
3. For each phrase, only 1 checkbox should be ticked. Initially all the checkboxes are ticked, and you will have to uncheck the irrelevant boxes.

Figure 5: Instructions provided for evaluating component classification in AMT.

Conversation:

USER 0: I feel like men in long term committed relationships should have a say in their s/o having a abortion. ok first i hate the argument "my body my choice". but in a way i kind of would see it in one night stands, rape, incest, or even just a fwb or fb. but if in a committed ltr the man who is in the relationship, if it's 100% his should have a say if she gets a abortion. i'm not talking about 100 different what if's but solely if in a relationship with her. he should have a say in the decision making. do i think she should be judged or punished? hell naw. but it would be common courtesy to say "hey i'm pregnant what are your thoughts?" instead of "hey i was pregnant and decided to have a abortion"
USER 1: um, the man doesn't carry any of the direct health affects of being pregnant. they don't directly deal with the indirect affects of pregnancy. thus, it seems odd that they can make a choice that can negativity affect the health of another person. the woman, because it is her body, should have the final say.

Phrases to annotate:

- um, the man doesn't carry any of the direct health affects of being pregnant. : **premise**

  ☑ Correct ☑ Incorrect ☑ Not In Conversation
- they don't directly deal with the indirect affects of pregnancy. : **premise**

  ☑ Correct ☑ Incorrect ☑ Not In Conversation
- the woman, because it is her body, should have the final say. : **premise**

  ☑ Correct ☑ Incorrect ☑ Not In Conversation

Figure 6: Component classification sample from AMT.

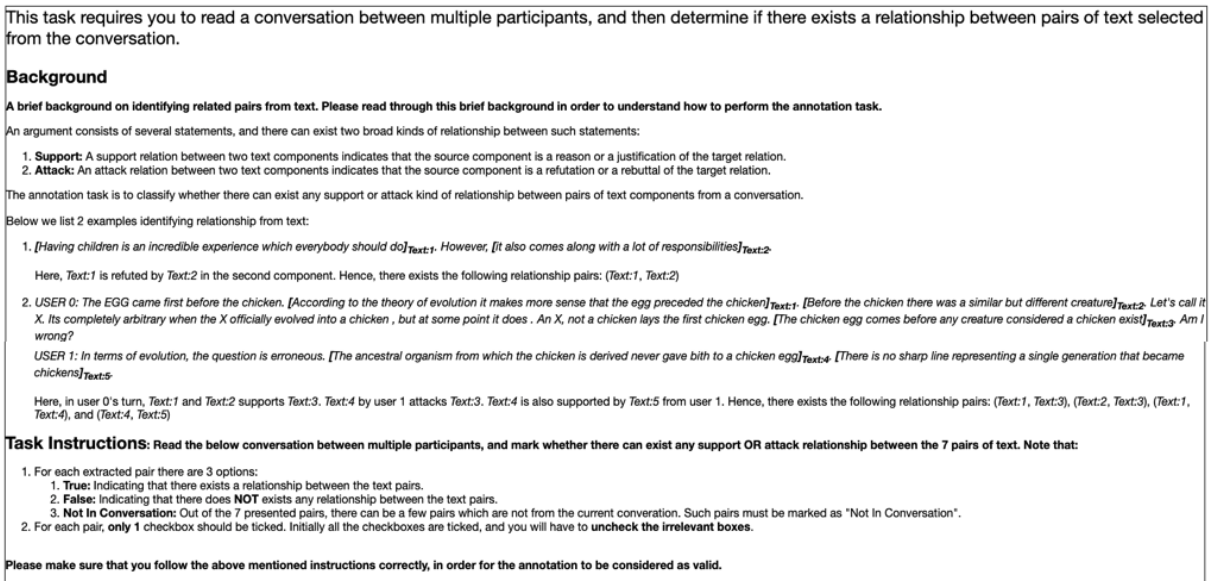

This task requires you to read a conversation between multiple participants, and then determine if there exists a relationship between pairs of text selected from the conversation.

Background

A brief background on identifying related pairs from text. Please read through this brief background in order to understand how to perform the annotation task.

An argument consists of several statements, and there can exist two broad kinds of relationship between such statements:

1. **Support:** A support relation between two text components indicates that the source component is a reason or a justification of the target relation.
2. **Attack:** An attack relation between two text components indicates that the source component is a refutation or a rebuttal of the target relation.

The annotation task is to classify whether there can exist any support or attack kind of relationship between pairs of text components from a conversation.

Below we list 2 examples identifying relationship from text:

1. *[Having children is an incredible experience which everybody should do]$_{Text:1}$. However, [it also comes along with a lot of responsibilities]$_{Text:2}$.*

   Here, *Text:1* is refuted by *Text:2* in the second component. Hence, there exists the following relationship pairs: (*Text:1*, *Text:2*)

2. *USER 0: The EGG came first before the chicken. [According to the theory of evolution it makes more sense that the egg preceded the chicken]$_{Text:1}$. [Before the chicken there was a similar but different creature]$_{Text:2}$. Let's call it X. Its completely arbitrary when the X officially evolved into a chicken , but at some point it does . An X, not a chicken lays the first chicken egg. [The chicken egg comes before any creature considered a chicken exist]$_{Text:3}$. Am I wrong?*

   USER 1: In terms of evolution, the question is erroneous. [The ancestral organism from which the chicken is derived never gave bith to a chicken egg]$_{Text:4}$. [There is no sharp line representing a single generation that became chickens]$_{Text:5}$.

   Here, in user 0's turn, *Text:1* and *Text:2* supports *Text:3*. *Text:4* by user 1 attacks *Text:3*. *Text:4* is also supported by *Text:5* from user 1. Hence, there exists the following relationship pairs: (*Text:1*, *Text:3*), (*Text:2*, *Text:3*), (*Text:1*, *Text:4*), and (*Text:4*, *Text:5*)

Task Instructions: Read the below conversation between multiple participants, and mark whether there can exist any support OR attack relationship between the 7 pairs of text. Note that:

1. For each extracted pair there are 3 options:
   1. **True:** Indicating that there exists a relationship between the text pairs.
   2. **False:** Indicating that there does **NOT** exists any relationship between the text pairs.
   3. **Not In Conversation:** Out of the 7 presented pairs, there can be a few pairs which are not from the current converation. Such pairs must be marked as "Not In Conversation".
2. For each pair, **only 1** checkbox should be ticked. Initially all the checkboxes are ticked, and you will have to **uncheck the irrelevant boxes**.

Please make sure that you follow the above mentioned instructions correctly, in order for the annotation to be considered as valid.

Figure 7: Instructions provided for evaluating intra/inter-turn relation identification in AMT.

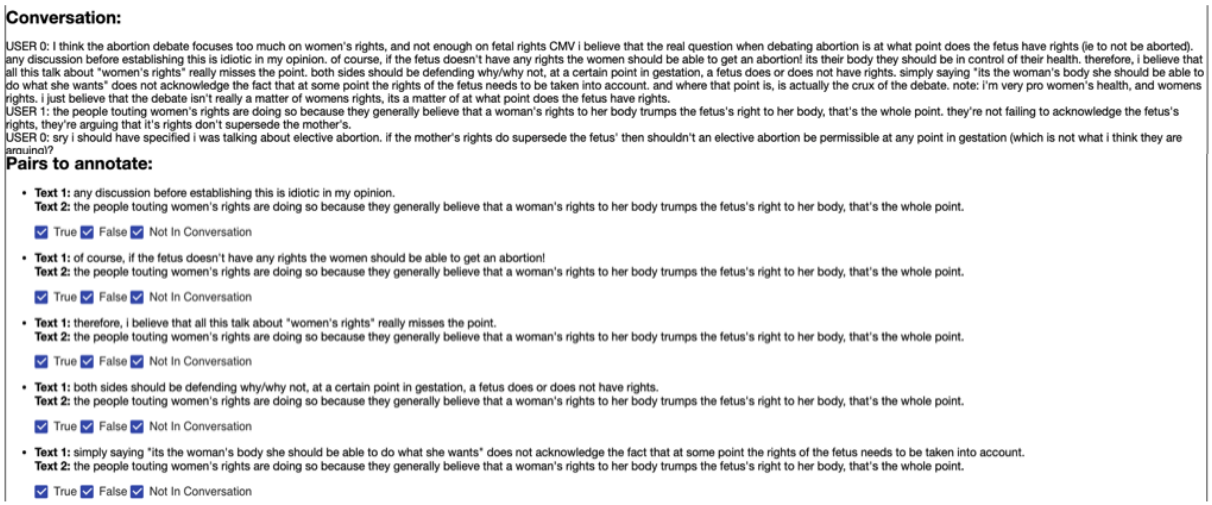

Conversation:

USER 0: I think the abortion debate focuses too much on women's rights, and not enough on fetal rights CMV i believe that the real question when debating abortion is at what point does the fetus have rights (ie to not be aborted). any discussion before establishing this is idiotic in my opinion. of course, if the fetus doesn't have any rights the women should be able to get an abortion! its their body they should be in control of their health. therefore, i believe that all this talk about "women's rights" really misses the point. both sides should be defending why/why not, at a certain point in gestation, a fetus does or does not have rights. simply saying "its the woman's body she should be able to do what she wants" does not acknowledge the fact that at some point the rights of the fetus needs to be taken into account. and where that point is, is actually the crux of the debate. note: i'm very pro women's health, and womens rights. i just believe that the debate isn't really a matter of womens rights, its a matter of at what point does the fetus have rights.
USER 1: the people touting women's rights are doing so because they generally believe that a woman's rights to her body trumps the fetus's right to her body, that's the whole point. they're not failing to acknowledge the fetus's rights, they're arguing that it's rights don't supersede the mother's.
USER 0: sry i should have specified i was talking about elective abortion. if the mother's rights do supersede the fetus' then shouldn't an elective abortion be permissible at any point in gestation (which is not what i think they are arguing)?

Pairs to annotate:

- **Text 1:** any discussion before establishing this is idiotic in my opinion.
  Text 2: the people touting women's rights are doing so because they generally believe that a woman's rights to her body trumps the fetus's right to her body, that's the whole point.
  ☑ True ☑ False ☑ Not In Conversation
- **Text 1:** of course, if the fetus doesn't have any rights the women should be able to get an abortion!
  Text 2: the people touting women's rights are doing so because they generally believe that a woman's rights to her body trumps the fetus's right to her body, that's the whole point.
  ☑ True ☑ False ☑ Not In Conversation
- **Text 1:** therefore, i believe that all this talk about "women's rights" really misses the point.
  Text 2: the people touting women's rights are doing so because they generally believe that a woman's rights to her body trumps the fetus's right to her body, that's the whole point.
  ☑ True ☑ False ☑ Not In Conversation
- **Text 1:** both sides should be defending why/why not, at a certain point in gestation, a fetus does or does not have rights.
  Text 2: the people touting women's rights are doing so because they generally believe that a woman's rights to her body trumps the fetus's right to her body, that's the whole point.
  ☑ True ☑ False ☑ Not In Conversation
- **Text 1:** simply saying "its the woman's body she should be able to do what she wants" does not acknowledge the fact that at some point the rights of the fetus needs to be taken into account.
  Text 2: the people touting women's rights are doing so because they generally believe that a woman's rights to her body trumps the fetus's right to her body, that's the whole point.
  ☑ True ☑ False ☑ Not In Conversation

Figure 8: Relation identification sample from AMT.

ConnPrompt: Connective-cloze Prompt Learning for Implicit Discourse Relation Recognition

Wei Xiang, Zhenglin Wang, Lu Dai, Bang Wang *
School of Electronic Information and Communications,
Huazhong University of Science and Technology, Wuhan, China
{xiangwei, wangzhenglin, dailu18, wangbang}@hust.edu.cn

Abstract

Implicit Discourse Relation Recognition (IDRR) is to detect and classify relation sense between two text segments without an explicit connective. Vanilla *pre-train and fine-tuning* paradigm builds upon a Pre-trained Language Model (PLM) with a task-specific neural network. However, the task objective functions are often not in accordance with that of the PLM. Furthermore, this paradigm cannot well exploit some *linguistic evidence* embedded in the pre-training process. The recent *pre-train, prompt, and predict* paradigm selects appropriate prompts to reformulate downstream tasks, so as to utilizing the PLM itself for prediction. However, for its success applications, prompts, verbalizer as well as model training should still be carefully designed for different tasks. As the first trial of using this new paradigm for IDRR, this paper develops a *Connective-cloze Prompt* (ConnPrompt) to transform the relation prediction task as a connective-cloze task. Specifically, we design two styles of ConnPrompt template: *Insert-cloze Prompt* (ICP) and *Prefix-cloze Prompt* (PCP) and construct an *answer space* mapping to the relation senses based on the hierarchy sense tags and implicit connectives. Furthermore, we use a multi-prompt ensemble to fuse predictions from different prompting results. Experiments on the PDTB corpus show that our method significantly outperforms the state-of-the-art algorithms, even with fewer training data.

1 Introduction

Implicit Discourse Relation Recognition (IDRR) aims at detecting and classifying some latent relation in between a pair of text segments (called arguments) without an explicit connective word. As illustrated in Fig. 1, an implicit discourse relation of "Contingency" is held between Argument-1 and Argument-2, and the implicit connective 'so' is inserted by annotators in the PDTB corpus. It is of great importance for many downstream Natural Language Processing (NLP) applications, such as question answering (Liakata et al., 2013), machine translation evaluation (Guzmán et al., 2014), information extraction (Xiang and Wang, 2019), sentiment analysis (Wang and Wang, 2020), and etc. However, due to the absence of an explicit connective word, inferring discourse relations from the contextual semantics of arguments is still a challenging task.

Existing *pre-train and fine-tuning* paradigm (Liu et al., 2021) builds upon a Pre-trained Language Model (PLM) with a well-designed sophisticated neural network to encode the *semantic content* and *interactive evidence* of argument pairs (Liu and Li, 2016; Lei et al., 2017; Bai and Zhao, 2018; Ruan et al., 2020; Li et al., 2020; Liu et al., 2020; Wu et al., 2022). Although the PLMs are adapted to these task-specific neural networks that can effectively learn a kind of contextual semantics of arguments, they introduce some additional parameters that need to be trained by a large amount of labelled data. Moreover, the task objective function is often not in accordance with that of the PLM. As such, the PLM needs to be fine-tuned for solving downstream tasks, resulting in poor utilization of the encyclopedic *linguistic evidence* embedded in the pre-training process.

On the one hand, we notice that the pre-training process of a PLM often uses a kind of cloze task, called Masked Language Model (MLM), to predict a piece of masked text from context. On the other hand, it has been reported that an explicit connective of an argument pair can greatly improve the relation classification performance in the *explicit discourse relation recognition* task (Pitler et al., 2008). Although explicit connectives are not available in the IDRR task, it is of great interests to explore whether we can transform the relation prediction task as a connective-cloze task, such that we

* Corresponding author: Bang Wang

Proceedings of the 29th International Conference on Computational Linguistics, pages 902–911
October 12–17, 2022.

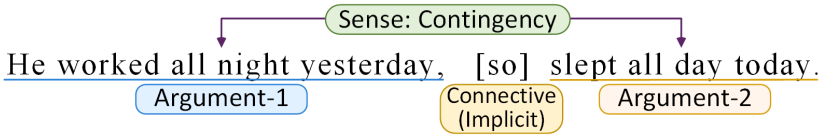

Figure 1: An example of implicit discourse relation annotation with manually inserted connective.

can exploit a pre-trained masked language model to predict a missing (yet possibly latent) connective for implicit relation classification. This is actually in accordance with the philosophy of the *prompt learning* paradigm, that is, predicting a connective as an *answer word* in some predefined template and then mapping the answer word to one relation sense.

The recent *pre-train, prompt, and predict* paradigm models the probability of text directly based on PLMs to perform prediction task. Specifically, it selects appropriate prompts to reformulate downstream task, so as to utilizing the PLM itself to predict the desired output (Liu et al., 2021). Moreover, the prompt paradigm is capable of performing few shot even zero-shot learning, as the PLM is sufficiently pre-trained and no external parameters need to be trained. However, for its successful applications in many downstream NLP tasks (Ding et al., 2021; Wang et al., 2021; Seoh et al., 2021), *prompts engineering*, *verbalizer* as well as *training strategies* should still be carefully designed for different tasks. In this paper, we explore how to transform the IDRR task against the prompt learning paradigm. To the best of our knowledge, this is the first paper for such explorations.

In this paper, we develop a *Connective-cloze Prompt* (ConnPrompt) framework to transform the relation prediction task as a connective-cloze task for the IDRR task. Specifically, we design two styles of ConnPrompt template: *Insert-cloze Prompt* (ICP) and *Prefix-cloze Prompt* (PCP), in which the [MASK] token is added for connective answer prediction. The ICP template concatenates two arguments as an entire word sequence, and the [MASK] token is inserted in between two arguments; The PCP template uses a [SEP] token to mark the boundary of two arguments, and the [MASK] token is added at the beginning of argument-1 or argument-2 as a prefix. Besides, we construct an *answer space* mapping an answer word to relation senses according to the hierarchy sense tags and implicit connectives in the training dataset. Furthermore, in order to leverage the complementary advantages of different prompt templates, we use a multi-prompt ensemble to fuse predictions from different prompting results.

We conduct the experiments on the PDTB corpus with four advanced masked language models: BERT, RoBERTa, ERNIE and DeBERTa. Experiment results show that our ConnPrompt significantly outperforms the state-of-the-art algorithms with full training data. Furthermore, our ConnPrompt can also achieve comparable performance even with fewer training data.

2 Method

In this section, we first introduce the overall framework of our ConnPrompt, then explain the details of connective-cloze prompt templates, verbalizer construction, multi-prompt ensembling, and model training strategies.

2.1 Overview

As illustrated in Fig. 2, our ConnPrompt has three main processes, including prompt templatize, answer prediction and verbalizer.

Prompt templatize: an input argument pair $x = (Arg_1; Arg_2)$ is reformulated into a prompt template $T(x)$ by concatenating two arguments and inserting some PLM-specific tokens such as [MASK], [CLS], [SEP], as the input of a PLM. The [MASK] token is added for PLM to predict an answer word v; While the [CLS] and [SEP] tokens are used to indicate the beginning and ending of an input word sequence, respectively. Note that some PLMs use other tokens like <mask>, <s>, and </s>, but they have the same meaning as described above.

Answer prediction: the pre-trained masked language model estimates the probability of each word in its vocabulary V for the [MASK] token as follows:

$$P_v(\texttt{[MASK]} = v \in V \mid T(x)). \quad (1)$$

We define a discrete answer space $V_a = \{v_1, v_2, \ldots, v_n\}$ containing the words manually selected according to the hierarchy sense tags and

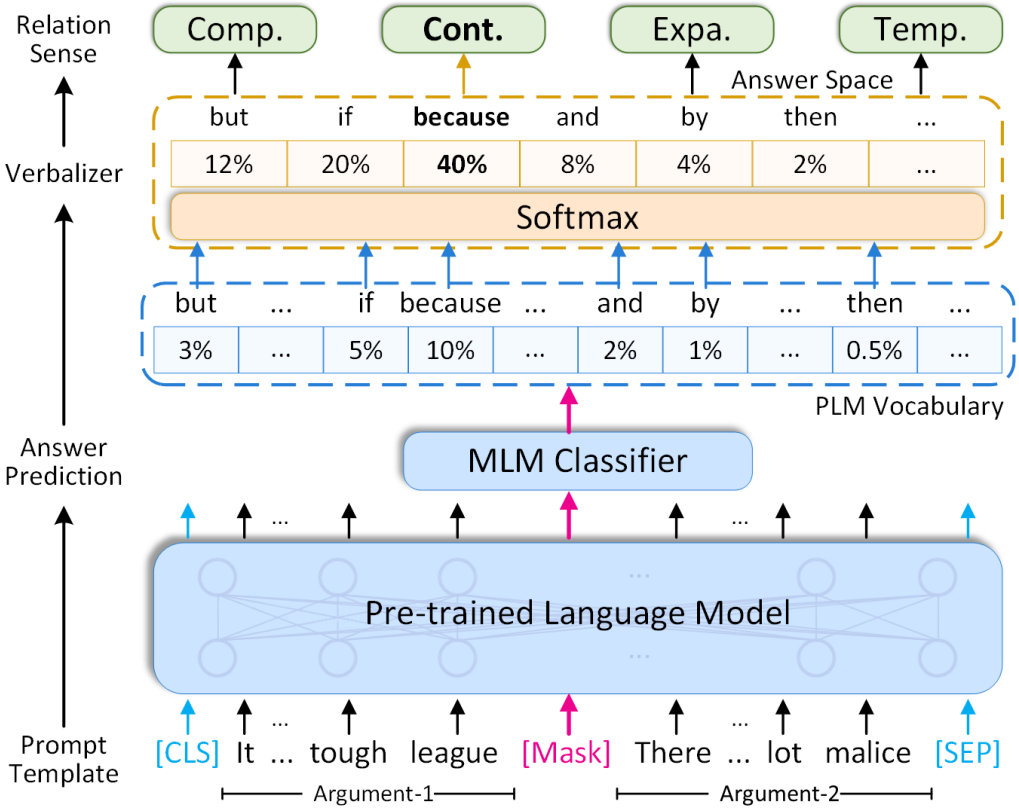

Figure 2: Illustration of our ConnPrompt framework.

implicit connectives, which is a subset of PLM's vocabulary, $V_a \subset V$. Then, a softmax layer is applied on the prediction scores of our answer words to normalize them into probabilities:

$$P_a(v_i \in V_a \mid T(x)) = \frac{e^{p_{v_i}}}{\sum_{j=1}^{n} e^{p_{v_j}}}, \qquad (2)$$

Verbalizer: the predicted answer word is projected to a unique discourse relation sense based on our pre-defined connection regulation.

2.2 Connective-cloze prompts

Motivated by the fact that connective words can effectively indicate the relation sense between two arguments, we design a kind of connective-cloze prompt template to predict a connective-bearing answer word for IDRR. In English syntax and grammar, connective words are usually located at the begining of a sentence or between two adjacent clauses. Thus we design two styles of prompt templates for connective-cloze prompts: *Prefix Cloze Prompt* (PCP) and *Insertion Cloze Prompt* (ICP), as shown in Fig. 3:

$T_1(x)$ is an ICP template, in which Arg_1 and Arg_2 are concatenated as an entire word sequence, and the [MASK] token is inserted between two arguments. $T_2(x)$ and $T_3(x)$ are PCP templates, in which the [SEP] token is also used to mark the boundary between Arg_1 and Arg_2, and the [MASK] token can be either added at the front of Arg_1 ($T_3(x)$) or Arg_2 ($T_2(x)$).

Relation Sense	Answer words
Comparison	*similarly*, *but*, *however*, *although*
Contingency	*for*, *if*, *because*, *so*
Expansion	*instead*, *by*, *thereby*, *specifically*, *and*
Temporal	*simultaneously*, *previously*, *then*

Table 1: Answer space of our ConnPrompt and their connection to the top-level class discourse relation sense tags in the PDTB.

2.3 Verbalizer Construction

Table 1 presents our verbalizer connection from the answer space to discourse relation sense labels. Note that the answer space is a small subset of the vocabulary in a PLM. We select sixteen answer words from nearly two hundred connectives in the PDTB corpus. In our verbalizer construction, the following four design issues are considered to satisfy the representative of each relation sense.

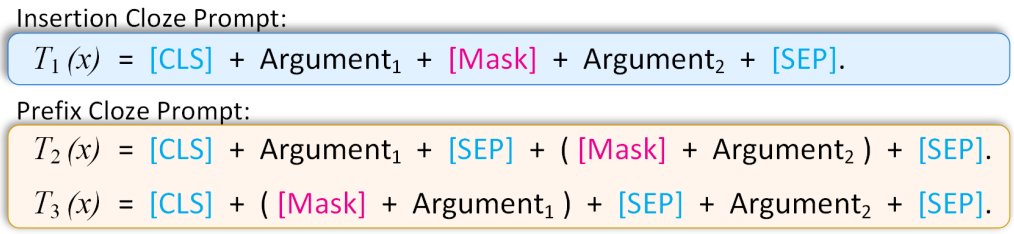

Figure 3: Illustration of our ConnPrompt Templates.

- **Answer Shape**: Only individual word connectives are selected as answer words, as most masked PLMs predicts only a single word.
- **Ambiguity**: Each answer word has one unambiguous connection with one discourse relation sense. Those words that can be used for multiple senses are not selected.
- **Frequency**: High frequency appearance connectives are prior to be selected as the answer words.
- **Semantic**: For those words with similar semantics in the same relation sense, we select a representative one to alleviate the answer confusion issue.

Specifically, we first eliminate those ambiguous connectives each for multiple senses, so that each answer word corresponds to only one discourse relation sense. We next rank the rest connectives according to their appearance frequencies in PDTB corpus to obtain a candidate set of answer words from the top of connective ranking. Finally, we select a representative word for those words with similar semantics in the same relation sense to alleviate the answer confusion issue and construct an answer space with sixteen words V_a as in Table 1. Each of them has a unique connection to one of the four top-level classes of relation sense $Y = \{\text{Comparison, Contingency, Expansion, Temporal}\}$.

2.4 Training Strategies

In model training, we tune the parameters of PLM using the IDRR training dataset based on our created prompt templates and answer space. Note that the final verbalizer layer is a projection and has no parameters to train. For model training, we assign an answer word for each instance of an argument pair as its ground truth label according to its manually annotated implicit connectives and the hierarchical sense tags in the PDTB. Specifically, if the implicit connective of an argument pair instance is in our answer space, we directly use it as the answer label; Otherwise, we take the most frequent answer word that has the same subtype-level sense tag as its label.

We adopt the cross entropy loss as the cost function:

$$J(\theta) = -\frac{1}{K}\sum_{k=1}^{K}\mathbf{y}^{(k)}\log(\hat{\mathbf{y}}^{(k)}) + \lambda\|\theta\|^2, \quad (3)$$

where $\mathbf{y}^{(k)}$ and $\hat{\mathbf{y}}^{(k)}$ are the gold label and predicted label of the k-th training instance respectively. λ and θ are the regularization hyper-parameters. We use the AdamW optimizer (Loshchilov and Hutter, 2019) with $L2$ regularization for model training.

2.5 Multi-prompt Ensembling

Multi-prompt learning uses multiple unanswered prompts for an input at inference time to make prediction (Lester et al., 2021). In accordance with English convention, we have designed three prompt templates. As each of them can output prediction probabilities for answer words, we make a decision fusion of majority voting as multi-prompt ensembling for final relation sense prediction. After each prompt predicting a specific relation sense, if two or more prompts have the same prediction, then their predicted relation sense is used as the final output. In a case that each prompt predicts differently, we choose the prediction from the prompt template with the highest F1 in the validation dataset.

3 Experiment Settings

In this section, we present our experimental settings, including dataset, PLMs, parameter settings and competitor models.

3.1 The PDTB Dataset

We conduct our experiments on the Penn Discourse TreeBank (PDTB) 3.0 corpus (Webber et al., 2019),

which contains more than one million words of English texts from Wall Street Journal. Following the conventional data splitting, we use sections 2-20 as the full training set, sections 21-22 as the testing set and 0-1 as the development set (Ji and Eisenstein, 2015). Our experiments are conducted on the four top-level classes of relation sense, including Comparison, Contingency, Expansion, Temporal.

For few shot learning, we randomly downsample the full training set to construct some subsets containing {Full, 50%, 30%, 20%, 10%} instances of the full training set. Table 2 summarizes the statistics of training instances in the training set and subsets.

Relation	Expa.	Comp.	Cont.	Temp.	Total
Train-Full	8645	1937	5916	1447	17945
Train-50%	2794	1937	2794	1447	8972
Train-30%	1346	1346	1346	1346	5384
Train-20%	898	898	898	898	3592
Train-10%	449	449	449	449	1796
Dev.	748	190	579	136	1653
Test	643	154	529	148	1474

Table 2: Statistics of the PDTB training set and downsampling sets with four top-level relation senses.

3.2 Pre-trained Language Models

We use four masked pre-trained language models (PLM) for comparision:

- **BERT** (Devlin et al., 2019): The most representive PLM proposed by Google [1], which is pre-trained using a *cloze task* and a *next sentence prediction* task.
- **RoBERTa** (Liu et al., 2019): A BERT-enhanced PLM proposed by Facebook [2], which removes the next sentence prediction objective and is pre-trained on a much larger dataset with some modified key hyperparameters.
- **ERNIE** (Sun et al., 2019): A knowledge-enhaced PLM proposed by Baidu [3], which uses some knowledgeable masking strategies in pre-training.
- **DeBERTa** (He et al., 2021): The latest masked PLM proposed by Microsoft [4], which improves BERT and RoBERTa models using a disentangled attention mechanism and an enhanced mask decoder.

3.3 Parameter Setting

Table 3 presents the configuration of each English masked pre-trained language model. All these PLM models are implemented in PyTorch [5] framework by HuggingFace transformers [6] (Wolf et al., 2020), and run with CUDA on NVIDIA GTX 1080 Ti GPUs. From our statistics, 99.46% of arguments do not exceed 50 words in PDTB. So we set the maximum length of each prompt template to 100 tokens, in which the maximum length of argument-1 is 50 tokens, and the rest 50 tokens are for argument-2 and [MASK], [CLS] and [SEP] tokens. We train all the four masked PLMs with the same mini-batch of 16 and learning rates of 5e-6, 1e-5, 2e-5 and 5e-5. We release the code at: https://github.com/HustMinsLab/ConnPrompt.

PLM	Model	Vocab. size	Layer	Dim.
BERT	bert-base-uncased	30522	12	768
RoBERTa	roberta-base	50265	12	768
ERNIE	ernie-2.0-en	30522	12	768
DeBERTa	deberta-base	50265	12	768

Table 3: Configuration of four pre-trained masked language models.

3.4 Competitors

We compare our ConnPrompt with the following advanced models:

- **DAGRN** (Chen et al., 2016) encodes word-pair interactions by a neural tensor network.
- **NNMA** (Liu and Li, 2016) combines two arguments' representations for stacked interactive attentions.
- **IPAL** (Ruan et al., 2020) propagates self-attention into interactive attention by a cross-coupled network.
- **PLR** (Li et al., 2020) uses a penalty-based loss re-estimation method to regulate the attention learning.

[1] https://github.com/google-research/bert
[2] https://github.com/pytorch/fairseq/
[3] https://github.com/PaddlePaddle/ERNIE
[4] https://github.com/microsoft/DeBERTa
[5] pytorch.org
[6] https://github.com/huggingface/transformers

- **BMGF** (Liu et al., 2020) combines bilateral multi-perspective matching and global information fusion to learn a deep contextualized representation.

Model	PLM	Acc	F1
DAGRN (ACL, 2016)	Word2vec	57.33%	45.11%
NNMA (EMNLP, 2016)	Glove	57.67%	46.13%
IPAL (COLING, 2020)	BERT	57.33%	51.69%
PLR (COLING, 2020)	BERT	63.84%	55.74%
BMGF (IJCAI, 2020)	RoBERTa	69.95%	62.31%
Our ConnPrompt	BERT	69.67%	64.00%
	RoBERTa	**75.17%**	**70.88%**
	ERNIE	72.93%	68.37%
	DeBERTa	74.63%	70.19%

Table 4: Overall results of comparison models on the PDTB corpus.

4 Result and Analysis

4.1 Overall Result

We implement a four-way classification on the top-level relation sense of the PDTB, in which macro $F1$ score and accuracy (Acc) are used for evaluation. Table 4 compares the overall performance between our ConnPrompt and the state-of-the-art models with pre-train and fine-tuning paradigm. We note that the first two competitors both use a kind of distributed representation based static word embeddings: Word2vec and Glove, provided by Google [7] and Stanford NLP Group [8]. While the others use Transformers based pre-trained masked language model BERT and RoBERTa, which are dynamic and contextual. We also compare the performance of our ConnPrompt with different PLMs.

The first observation is that the IPAL, PLR and BMGF model can obviously outperform the first two competitors, viz., the DAGRN and NNMA model. This might be attributed to the use of more advanced dynamic PLMs which are pre-trained with deeper neural networks and larger scale of parameters based on Transformers. Indeed, transformer-based PLMs have been proven to be more effective for many downstream NLP tasks (Devlin et al., 2019; Liu et al., 2019). Although these competitors' well-designed task-specific neural networks also have a certain impact on the performance, the gaps between dynamic PLMs and static word embeddings are still apparent in the IDRR task.

The second observation is that in our ConnPrompt employing different PLMs, the ConnPrompt-BERT performs the worst. We note that although they all employ Transformer based model in pre-training, the RoBERTa, ERNIE and DeBERTa have applied some adjusted and optimized pre-training processes. Specifically, the RoBERTa removes the next sentence prediction task and uses a much larger dataset for training; While the ERNIE uses some knowledgeable masking strategies; and the DeBERTa applies a disentangled attention mechanism to encode context and position information separately. This suggests that the improvements and optimization in the pre-training process can effectively improve the performance of prompt learning.

Finally, our ConnPrompt with all four PLMs have achieved better performance than all conventional pre-train and fine-tuning paradigm models in macro F1 score, even some of the competitors have used advanced PLMs like RoBERTa and BERT, to train an elaborate downstream task model. Besides, the ConnPrompt-RoBERTa and ConnPrompt-DeBERTa model have achieved significant improvements over all competitors and PLMs in terms of much higher macro F1 score and Acc. We attribute its outstanding performance to our task transformation of connective-cloze prediction into the training of PLMs, other than using task-specific model built-upon a PLM, by which our ConnPrompt can better enjoy the encyclopedic linguistic evidence embedded in a PLM during the model training process.

4.2 Prompt Template Effections

In the prompt paradigm, using different templates may impact on the task performance. Table 5 compares the results of our designed single-prompt templates and multi-prompt ensembling of ConnPrompt.

It can be first observed that using different prompt templates do result in some performance disparity, even though the gaps are not obvious. For single-prompt learning, the BERT and RoBERTa have achieved the best performance in Prompt-1, while the ERNIE and DeBERTa have achieved the best performance in Prompt-2. This suggests that the semantic encoding might play the central role in BERT and RoBERTa, as the Prompt-1 does

[7] code.google.com/archive/p/word2vec
[8] https://nlp.stanford.edu/projects/glove/

PLM	BERT		RoBERTa		ERNIE		DeBERTa	
	Acc	F1	Acc	F1	Acc	F1	Acc	F1
Prompt-1	**69.74%**	63.95%	74.36%	69.91%	72.05%	67.25%	72.32%	67.74%
Prompt-2	69.34%	63.69%	73.61%	69.63%	72.25%	67.59%	72.66%	67.98%
Prompt-3	67.64%	62.65%	73.54%	69.00%	70.15%	66.21%	72.66%	67.92%
Multi-Prompt	69.67%	**64.00%**	**75.17%**	**70.88%**	**72.93%**	**68.37%**	**74.63%**	**70.19%**

Table 5: Results of single-prompt templates and multi-prompt ensembling with different PLMs on the PDTB corpus.

not mark the boundary of two arguments, and the [MASK] token is inserted between them to form an entire word sequence.

By contrast, ERNIE and DeBERTa might have more consideration on position encoding, as the Prompt-2 uses a [SEP] token to distinguish two input arguments and the [MASK] token is added at the front of argument-2. They have shown more powerful ability with these position information. We also observe that Prompt-3 cannot outperform the other two prompts. This may be attributed to its infrequent grammar structure that places the connective at the beginning of the first argument.

Our multi-prompt model achieves performance improvements over the single-prompt models for almost all PLMs. This indicates that our multi-prompt ensembling is effective for fusing multiple single-prompts for discourse relation classification.

4.3 Few Shot Learning

Some researchers have reported that the prompt paradigm is of some robustness to using fewer training data in other NLP tasks, like text classification (Wang et al., 2021) and entity typing (Ding et al., 2021). We would also like to examine the performance of ConnPrompt and competitors with few shot learning. We adopt down-sampling to construct smaller training datasets; While the development set and test set remain unchanged.

Fig. 4 summarizes the few short learning results. It is not unexpected that both our ConnPrompt and competitors suffer from the reduction of training data. The left column presents the ConnPrompt results when using different PLMs. It is again observed that the RoBERTa is still the best PLM choice for the ConnPrompt with fewer training data. The center column compares the ConnPrompt built upon BERT with two competitors, viz., IPAL and PLR, also employing BERT as their PLM. We select the single-prompt with the best performance in the validation dataset for comparison. It is observed that the performance improvements of ConnPrompt are quite significant in few short learning. In particular, when using 10% training data, the ConnPrompt (F1 48.32% and Acc 50.41%) outperforms the IPAL and PLR using 50% training data, (IPAL: F1 45.53% and Acc 50.00%; PLR: F1 47.13% and Acc 50.07%). Similar results can also be observed for ConnPrompt built upon RoBERTa (right column). These results validate the effectiveness of ConnPromt even with fewer training data.

5 Related Work

5.1 Implicit Discourse Relation Recognition

The pre-train and fine-tuning paradigm for the IDRR task is usually approached as a classification problem, and the key is to construct a downstream task model built-upon some PLM for the argument representation learning.

Deep learning models have prevailed for their capabilities of automatic learning argument representation upon PLM (Zhang et al., 2015; Rutherford et al., 2017). For example, the SCNN model (Zhang et al., 2015) obtains each argument representation via a single convolution layer, and the concatenation of two arguments' representations is used for relation classification. Rutherford et al. (2017) employ a LSTM network to capture word contextual semantics for argument representation. Some hybrid models have attempted to combine CNN, LSTM, graph convolutional networks and etc., for more sophisticated argument representation (Zhang et al., 2021; Shi and Demberg, 2019; Jiang et al., 2021). These approaches, however, have ignored the fact that different words may contribute differently in argument representation learning.

Attention mechanisms can guide a neural model to unequally encode each word according to its contextual importance for argument representa-

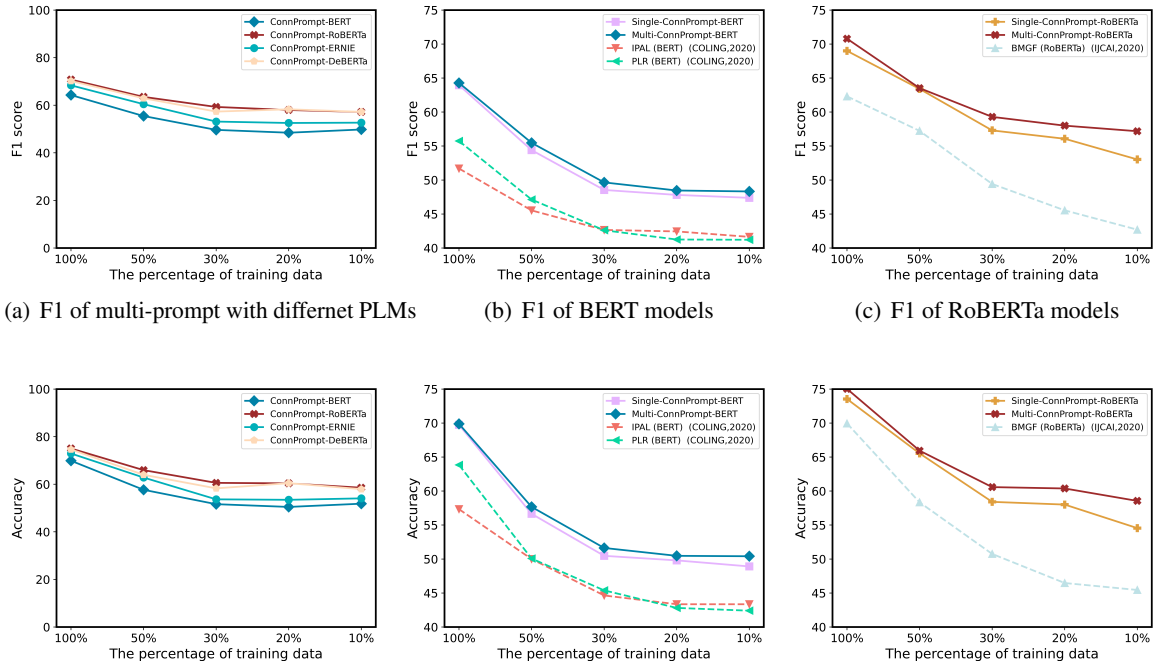

(a) F1 of multi-prompt with differnet PLMs (b) F1 of BERT models (c) F1 of RoBERTa models

(d) Acc of multi-prompt with differnet PLMs (e) Acc of BERT models (f) Acc of RoBERTa models

Figure 4: Performance comparison of few shot learning on the PDTB corpus.

tion (Zhou et al., 2016; Liu and Li, 2016; Lan et al., 2017; Guo et al., 2020; Ruan et al., 2020; Li et al., 2020). For example, Zhou et al. (2016) apply self-attention to weight a word according to its similarity to its belonging argument. Guo et al. (2020) and Liu et al. (2020) adopt an interactive attention to differentiate words in one argument, where a word is weighted according to the similarity between its encoding and another argument representation. Liu and Li (2016) design a multi-level attention to repeatedly compute word importance in a hierarchical way. Ruan et al. (2020) propose a pipeline workflow to apply interactive attention after self-attention. Li et al. (2020) use a penalty-based loss re-estimation method to regulate the attention learning.

5.2 Prompt Learning for NLPs

After the emergence of large-scale PLMs like BERT (Devlin et al., 2019), RoBERTa (Liu et al., 2019), ERNIE (Sun et al., 2019) and etc., the prompt learning has become a new paradigm for some NLP tasks, which use the probability of text in PLMs to perform a prediction task (Seoh et al., 2021; Wang et al., 2021; Ding et al., 2021). For example, Seoh et al. (2021) propose a cloze question prompt and a natural language inference prompt for aspect-based sentiment analysis. Wang et al. (2021) propose a transferable prompting framework to capture cross-task knowledge for few-shot text classification. Ding et al. (2021) apply a cloze-style prompt learning on fine-grained entity typing in fully supervised, few-shot and zero-shot scenarios. Up to now, prompt learning has achieved promising results on some NLP tasks, but has not been reported for the IDRR task to the best of our knowledge.

The proposed ConnPrompt transforms the relation prediction task as a connective-cloze task against the prompt learning paradigm.

6 Conclusion

This is the first paper on examining the pre-trained, prompt, predict paradigm for the IDRR task. We have developed a Connective-cloze Prompt (ConnPrompt) to transform the IDRR task as a connective-cloze prediction task based on a pre-trained language model (PLM). Two styles of manually designed prompt template: Insertion Connective Prompt and Prefix Connective Prompt, have been designed to convert input argument pairs into the prompt formulation, and a discrete answer space is constructed with sixteen answer words for verbalizer. Experiments on the PDTB corpus

have validated that our ConnPrompt can significantly outperform the state-of-the-art algorithms, even with fewer training data.

This paper has applied the basic techniques of prompt learning for the IDRR task. In the last year, the prompt paradigm has achieved some new interesting advances, covering the techniques for choosing pre-trained models, designing continuous prompt templates, constructing answer space as well as training and tuning strategies. We note that some of these new techniques shall also be examined and improved for the IDRR task. Besides, the excellent performance of the prompt learning in this paper also motivates us to further investigate its applications in other NLP tasks.

Acknowledgements

This work is supported in part by National Natural Science Foundation of China (Grant No: 62172167). We also want to use our ConnPrompt model on MindSpore[9], which is a new deep learning computing framework. These problems are left for future work.

References

Hongxiao Bai and Hai Zhao. 2018. Deep enhanced representation for implicit discourse relation recognition. In *Proceedings of the 27th International Conference on Computational Linguistics*, pages 571–583.

Jifan Chen, Qi Zhang, Pengfei Liu, Xipeng Qiu, and Xuanjing Huang. 2016. Implicit discourse relation detection via a deep architecture with gated relevance network. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics*, pages 1726–1735.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 4171–4186.

Ning Ding, Yulin Chen, Xu Han, Guangwei Xu, Pengjun Xie, Hai-Tao Zheng, Zhiyuan Liu, Juanzi Li, and Hong-Gee Kim. 2021. Prompt-learning for fine-grained entity typing. *arXiv preprint*, arXiv:2108.10604:1–12.

Fengyu Guo, Ruifang He, Jianwu Dang, and Jian Wang. 2020. Working memory-driven neural networks with a novel knowledge enhancement paradigm for implicit discourse relation recognition. In *Proceedings of the Thirty-Fourth AAAI Conference on Artificial Intelligence*, pages 7822–7829.

Francisco Guzmán, Shafiq Joty, Lluís Màrquez, and Preslav Nakov. 2014. Using discourse structure improves machine translation evaluation. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics*, pages 687–698.

Pengcheng He, Xiaodong Liu, Jianfeng Gao, and Weizhu Chen. 2021. Deberta: decoding-enhanced bert with disentangled attention. In *9th International Conference on Learning Representations*, pages 1–23.

Yangfeng Ji and Jacob Eisenstein. 2015. One vector is not enough: Entity-augmented distributed semantics for discourse relations. *Transactions of the Association for Computational Linguistics*, 3:329–344.

Feng Jiang, Yaxin Fan, Xiaomin Chu, Peifeng Li, and Qiaoming Zhu. 2021. Not just classification: Recognizing implicit discourse relation on joint modeling of classification and generation. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, page 2418–2431.

Man Lan, Jianxiang Wang, Yuanbin Wu, Zheng-Yu Niu, and Haifeng Wang. 2017. Multi-task attention-based neural networks for implicit discourse relationship representation and identification. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 1299–1308.

Wenqiang Lei, Xuancong Wang, Meichun Liu, Ilija Ilievski, Xiangnan He, and Min-Yen Kan. 2017. Swim: a simple word interaction model for implicit discourse relation recognition. In *Proceedings of the Twenty-Sixth International Joint Conference on Artificial Intelligence*, pages 4026–4032.

Brian Lester, Rami Al-Rfou, and Noah Constant. 2021. The power of scale for parameter-efficient prompt tuning. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 3045–3059.

Xiao Li, Yu Hong, Huibin Ruan, and Zhen Huang. 2020. Using a penalty-based loss re-estimation method to improve implicit discourse relation classification. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 1513–1518.

Maria Liakata, Simon Dobnik, Shyamasree Saha, Colin Batchelor, and Dietrich Rebholz Schuhmann. 2013. A discourse-driven content model for summarising scientific articles evaluated in a complex question answering task. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*, pages 747–757.

[9]http://www.mindspore.cn/

Pengfei Liu, Weizhe Yuan, Jinlan Fu, Zhengbao Jiang, Hiroaki Hayashi, and Graham Neubig. 2021. Pre-train, prompt, and predict: A systematic survey of prompting methods in natural language processing. *arXiv preprint*, arXiv:2107.13586:1–46.

Xin Liu, Jiefu Ou, Yangqiu Song, and Xin Jiang. 2020. On the importance of word and sentence representation learning in implicit discourse relation classification. In *Proceedings of the Twenty-Ninth International Joint Conference on Artificial Intelligence*, pages 3830–3836.

Yang Liu and Sujian Li. 2016. Recognizing implicit discourse relations via repeated reading: Neural networks with multi-level attention. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 1224–1233.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Roberta: A robustly optimized BERT pretraining approach. *arXiv preprint*, arXiv:1907.11692:1–13.

Ilya Loshchilov and Frank Hutter. 2019. Decoupled weight decay regularization. In *7th International Conference on Learning Representations*, pages 1–18.

Emily Pitler, Mridhula Raghupathy, Hena Mehta, Ani Nenkova, Alan Lee, and Aravind K. Joshi. 2008. Easily identifiable discourse relations. In *Proceedings of the 22nd International Conference on Computational Linguistics*, pages 87–90.

Huibin Ruan, Yu Hong, Yang Xu, Zhen Huang, Guodong Zhou, and Min Zhang. 2020. Interactively-propagative attention learning for implicit discourse relation recognition. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 3168–3178.

Attapol Rutherford, Vera Demberg, and Nianwen Xue. 2017. A systematic study of neural discourse models for implicit discourse relation. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics*, pages 281–291.

Ronald Seoh, Ian Birle, Mrinal Tak, Haw-Shiuan Chang, Brian Pinette, and Alfred Hough. 2021. Open aspect target sentiment classification with natural language prompts. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 6311–6322.

Wei Shi and Vera Demberg. 2019. Learning to explicitate connectives with seq2seq network for implicit discourse relation classification. In *Proceedings of the 13th International Conference on Computational Semantics - Long Papers*, page 188–199.

Yu Sun, Shuohuan Wang, Yu-Kun Li, Shikun Feng, Xuyi Chen, Han Zhang, Xin Tian, Danxiang Zhu, Hao Tian, and Hua Wu. 2019. ERNIE: enhanced representation through knowledge integration. *arXiv preprint*, arXiv:1904.09223:1–8.

Chang Wang and Bang Wang. 2020. An end-to-end topic-enhanced self-attention network for social emotion classification. In *Proceedings of The Web Conference 2020*, pages 2210–2219.

Chengyu Wang, Jianing Wang, Minghui Qiu, Jun Huang, and Ming Gao. 2021. Transprompt: Towards an automatic transferable prompting framework for few-shot text classification. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 2792–2802.

Bonnie Webber, Rashmi Prasad, Alan Lee, and Aravind Joshi. 2019. The penn discourse treebank 3.0 annotation manual. *Philadelphia, University of Pennsylvania*, pages 1–81.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Rémi Louf, Morgan Funtowicz, Joe Davison, Sam Shleifer, Patrick von Platen, Clara Ma, Yacine Jernite, Julien Plu, Canwen Xu, Teven Le Scao, Sylvain Gugger, Mariama Drame, Quentin Lhoest, and Alexander M. Rush. 2020. Transformers: State-of-the-art natural language processing. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 38–45.

Changxing Wu, Liuwen Cao, Yubin Ge, Yang Liu, Min Zhang, and Jinsong Su. 2022. A label dependence-aware sequence generation model for multi-level implicit discourse relation recognition. In *Proceedings of the Thirty-Sixth AAAI Conference on Artificial Intelligence*, pages 11486–11494.

Wei Xiang and Bang Wang. 2019. A survey of event extraction from text. *IEEE Access*, 7:173111–173137.

Biao Zhang, Jinsong Su, Deyi Xiong, Yaojie Lu, Hong Duan, and Junfeng Yao. 2015. Shallow convolutional neural network for implicit discourse relation recognition. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 2230–2235.

Yingxue Zhang, Fandong Meng, Li Peng, Jian Ping, and Jie Zhou. 2021. Context tracking network: Graph-based context modeling for implicit discourse relation recognition. In *Proceedings of the 2021 Conference of the North American Chapter of the Association Computational Linguistics*, pages 1592–1599.

Peng Zhou, Wei Shi, Jun Tian, Zhenyu Qi, Bingchen Li, Hongwei Hao, and Bo Xu. 2016. Attention-based bidirectional long short-term memory networks for relation classification. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics*, page 207–212.

A Distance-Aware Multi-Task Framework for Conversational Discourse Parsing

Yaxin Fan, Peifeng Li, Fang Kong, and **Qiaoming Zhu***
School of Computer Science and Technology, Soochow University, Suzhou, China
yxfansuda@stu.suda.edu.cn
{pfli, kongfang, qmzhu}@suda.edu.cn

Abstract

Conversational discourse parsing aims to construct an implicit utterance dependency tree to reflect the turn-taking in a multi-party conversation. Existing works are generally divided into two lines: graph-based and transition-based paradigms, which perform well for short-distance and long-distance dependency links, respectively. However, there is no study to consider the advantages of both paradigms to facilitate conversational discourse parsing. As a result, we propose a distance-aware multi-task framework DAMT that incorporates the strengths of transition-based paradigm to facilitate the graph-based paradigm from the encoding and decoding process. To promote multi-task learning on two paradigms, we first introduce an Encoding Interactive Module (EIM) to enhance the flow of semantic information between both two paradigms during the encoding step. And then we apply a Distance-Aware Graph Convolutional Network (DAGCN) in the decoding process, which can incorporate the different-distance dependency links predicted by the transition-based paradigm to facilitate the decoding of the graph-based paradigm. The experimental results on the datasets STAC and Molweni show that our method can significantly improve the performance of the SOTA graph-based paradigm on long-distance dependency links. Our code is available at https://github.com/yxfanSuda/DAMT.

1 Introduction

The goal of conversational discourse parsing is to uncover latent conversation topics and construct an implicit utterance dependency tree to reflect the turn-taking in a multi-party conversation. Since the discourse structure is essential to understand multi-party conversations, it has been widely applied to various Natural Language Processing (NLP) applications,such as response generation (Hu et al., 2019), reading comprehension(Li et al., 2021b; Li and Zhao, 2021), meeting summarization (Feng et al., 2021), and emotion recognition(Sun et al., 2021).

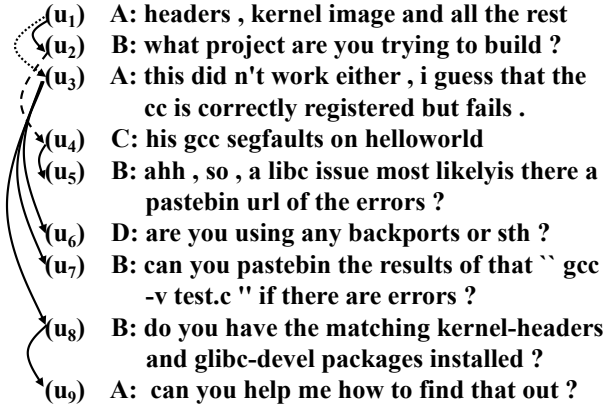

Figure 1: A multi-party dialogue with its dependency structure, where the solid lines, dotted lines and dashed lines denote the relations "Clarification Question", "Comment", and "Result" respectively, and A, B, C, D mean different speakers.

Segmented Discourse Representation Theory (SDRT) (Asher et al., 2003) is one of the most influential theories to reveal the overall discourse structures in conversational discourse parsing. Unlike Rhetorical Structure Theory (RST) (Mann and Thompson, 1987), which limits the relationship to occur between adjacent EDUs[1] in monologue, SDRT represents multi-party conversations as dependency-based discourse structures, due to crossing dependencies. Recently, some SDRT-style corpora have been built, such as STAC (Asher et al., 2016) and Molweni (Li et al., 2020). Figure 1 shows an example of a multi-party conversation and its dependency structure from Molweni.

Existing work on conversational discourse pars-

*Corresponding author

[1]Elementary Discourse Units(EDUs) are the fundamental discourse units in discourse parsing. In the monologue, each EDU corresponds to a phrase or sentence. In the conversation, each EDU corresponds to an utterance.

Proceedings of the 29th International Conference on Computational Linguistics, pages 912–921
October 12–17, 2022.

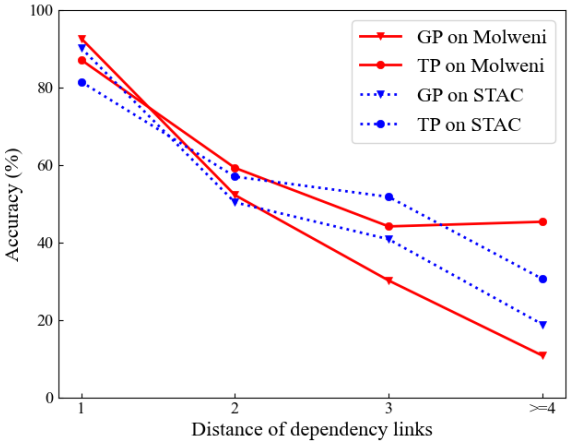

Figure 2: The performance of dependency links at different distances predicted by the graph-based and transition-based paradigms on the testing set of Molweni and STAC, respectively. The x-axis is the relative distance between EDUs (For example, the distance of (u_1, u_2) and (u_3, u_8) are 1 and 5 in Figure 1.). And the y-axis is the accuracy of dependency links. "TP" indicates Transition-based Paradigm and "GP" means Graph-based Paradigm.TP comes from (Shi and Huang, 2019), which utilizes hierarchical GRU to encode the conversations, while GP comes from (Wang et al., 2021a) and we also utilize hierarchical GRU to encode the dialogues for a fair comparison.

ing can be divided into two lines: graph-based and transition-based paradigms. The graph-based paradigm (Muller et al., 2012; Afantenos et al., 2015; Perret et al., 2016; Yang et al., 2021; Wang et al., 2021a) first obtains the probability of the discourse relation for each EDU pair, then a global decoding method is applied to construct the discourse structure. The transition-based paradigm (Shi and Huang, 2019; Wang et al., 2021b) first obtains the probability of the discourse relation between the current EDU and all previous EDUs, then discourse structure is constructed incrementally. Due to the discrepancy in the parsing process, both of them have different strengths in predicting the dependency links at various distances.

As shown in Figure 2, we analysis the performance of dependency links at different distances predicted by the above two paradigms. The results show that the graph-based paradigm performs better for dependency links with the distance 1, while the transition-based paradigms performs better when the distance greater than 1. As a result, it is a great challenge to combine the advantages of both two paradigms to facilitate conversational discourse parsing.

Previous work (Falenska et al., 2020) has demonstrated the effectiveness of multi-task learning to integrate both two paradigms in a similar task dependency syntactic parsing. Through the shared encoding layer, both two paradigms can facilitate each other implicitly. However, it is not sufficient to apply this approach to conversational discourse parsing, because the advantages of one paradigm cannot be explicitly exploited to facilitate the other one.

To alleviate the above issues, we propose a Distance-Aware Multi-Task framework (DAMT) for conversational discourse parsing that allows one paradigm to explicitly facilitate the decoding process of the other from the encoding and decoding process, respectively. Specially, we introduce an Encoding Interactive Module (EIM) to enhance the flow of semantic information between both two paradigms during the encoding step. And then we apply a Distance-Aware Graph Convolutional Network (DAGCN) in the decoding process, which can incorporate the different-distance dependency links predicted by the transition-based paradigm to facilitate the decoding of the graph-based paradigm. The experimental results on two datasets STAC and Molweni show that our DAMT outperforms the SOTA baselines, especially the significant improvement on those dependency links with long distances.

2 Related Work

Most previous studies for overall discourse structure parsing are based on Rhetorical Structure Theory Discourse TreeBank (RST-DT) (Carlson et al., 2003), including greedy bottom-up approach (Feng and Hirst, 2014), CYK-based approaches (Joty et al., 2015; Liu and Lapata, 2017) and transition-based methods (Wang et al., 2017; Lin et al., 2019; Kobayashi et al., 2020; Zhang et al., 2021).

In this paper, we focus on parsing conversational dependency structures that allow crossing dependencies. Recently, there are two available corpora, i.e., STAC (Asher et al., 2016) and Molweni (Li et al., 2020) defined 16 relation types. STAC collected from an online game *The Settlers of Catan*, which contains 1,062 and 111 dialogues for training and testing, respectively. Molweni is based on Ubuntu Chat (Lowe et al., 2015), which contains 9,000, 500 and 500 instances for training, validating and testing, respectively.

Up to now, only a few studies focused on con-

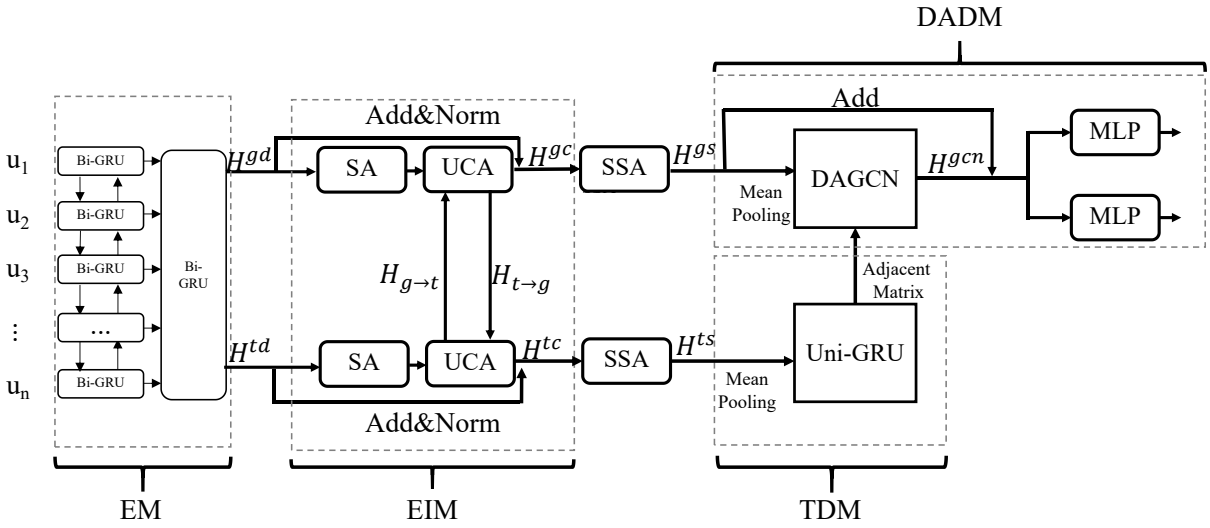

Figure 3: The architecture of our DAMT framework.

versational discourse parsing and most of them can be divided into two paradigms, i.e., graph-based paradigm and transition-based paradigm.

Graph-based paradigm Muller et al. (2012), Afantenos et al. (2015) and Perret et al. (2016) adopted traditional manual features to calculate the probabilities of all EDU pairs and then global decoding algorithm (e.g., Maximum Spanning Trees, A* and Integer Linear Programming) was used to construct dependency structures. With the development of deep learning, some advanced methods are used to obtain the semantic representation of each EDU pair. Wang et al. (2021a) proposed a novel edge-centric graph neural network to enhance the semantic representation of EDUs. Yang et al. (2021) first used the dependency syntactic graph to obtain a better EDU representation and then a biaffine relation prediction layer was applied to obtain the probability of each EDU pair.

Transition-based paradigm Shi and Huang (2019) proposed a Deep Sequence Model (DSM) to predict dependency links and corresponding relation types jointly and alternately. Their model not only consider the local information of the concerned EDUs but also utilizes the historical structure. Based on DSM, Wang et al. (2021b) adopted the graph attention network by incorporating cohesion information including lexical chain and coreference chain to enhance the semantic representation of EDUs.

3 DAMT

Our framwwork DAMT is shown in Figure 3, which includes four components: Encoding Module (EM), Encoding Interaction Module (EIM), Transition-based Decoding Module (TDM), and Distance-Aware Decoding Module (DADM). In EM, the hierarchical GRU is applied to obtain the semantic representation of dialogues. In EIM, the semantic representations of different paradigms can interact explicitly to promote the multi-task learning. In TDM, we adopt a pointer network for transition-based decoding to obtain the dependency structures. In DADM, the dependency structures predicted by TDM are incorporated by DAGCN for the final graph-based decoding.

3.1 Encoding Module

In the encoding module EM, we adopted hierarchical GRU to obtain the semantic representation of dialogues for both paradigms. For each EDU u_i in dialogue D=$\{u_1, u_2, \cdots, u_n\}$, a bidirectional GRU (bi-GRU) encoder is applied on the word sequence, and the last hidden states in two directions are concatenated as the EDU-level semantic representation, denoted as $\boldsymbol{h}_e^i \in \mathbb{R}^d$. Then another bi-GRU is applied on the EDU-level representation to obtain the dialogue-level representation. We use $\boldsymbol{H}_{td}$ and $\boldsymbol{H}_{gd}$ to denote the semantic representation of transition-based paradigm and graph-based paradigm respectively, where $\boldsymbol{H}_{td}, \boldsymbol{H}_{gd} \in \mathbb{R}^{n\times d}$.

3.2 Encoding Interaction Module

Several studies (E et al., 2019; Qin et al., 2021; Li et al., 2021a) have demonstrated that the explicit interaction between encoding representations of different tasks can better improve each other in multi-task learning. Inspired by this, we propose an

encoding interaction module EIM, which can build a bidirectional connection between two paradigms.

Our EIM consists of two Self Attention (SA) layers and two Unidirectional Cross Attention (UCA) layers. We first feed the semantic representation $\boldsymbol{H}_{td}$ and $\boldsymbol{H}_{gd}$ from EM into the SA sub-layers to obtain the internal semantic information for each paradigm as follows.

$$\begin{aligned}\boldsymbol{H}_t &= \mathrm{SA}(\boldsymbol{W}_q^s\boldsymbol{H}_{td}, \boldsymbol{W}_k^s\boldsymbol{H}_{td}, \boldsymbol{W}_v^s\boldsymbol{H}_{td}) \\ \boldsymbol{H}_g &= \mathrm{SA}(\boldsymbol{W}_q^s\boldsymbol{H}_{gd}, \boldsymbol{W}_k^s\boldsymbol{H}_{gd}, \boldsymbol{W}_v^s\boldsymbol{H}_{gd})\end{aligned} \tag{1}$$

where $\mathrm{SA}(.)$ denotes multi-head attention as (Vaswani et al., 2017) and $\boldsymbol{W}_q^s, \boldsymbol{W}_k^s, \boldsymbol{W}_v^s$ are weight matrix, which map vectors to the same feature space.

Second, two UCA layers are applied to build the connection between the two paradigms, where one from $\boldsymbol{H}_t$ to $\boldsymbol{H}_g$ and one from $\boldsymbol{H}_g$ to $\boldsymbol{H}_t$ as follows.

$$\begin{aligned}\boldsymbol{H}_{g\to t} &= \mathrm{UCA}(\boldsymbol{W}_q^c\boldsymbol{H}_g, \boldsymbol{W}_k^c\boldsymbol{H}_t, \boldsymbol{W}_v^c\boldsymbol{H}_t) \\ \boldsymbol{H}_{t\to g} &= \mathrm{UCA}(\boldsymbol{W}_q^c\boldsymbol{H}_t, \boldsymbol{W}_k^c\boldsymbol{H}_g, \boldsymbol{W}_v^c\boldsymbol{H}_g)\end{aligned} \tag{2}$$

$\mathrm{UCA}(.)$ is a variant of $\mathrm{SA}(.)$, which uses $\boldsymbol{H}_t(\boldsymbol{H}_g)$ as query vectors and $\boldsymbol{H}_g(\boldsymbol{H}_t)$ as the context vector, thus enabling an explicit interaction between the two vectors. The UCA layer is used to make the encoding semantics of one paradigm updated with the guidance of the other one, achieving a bidirectional connection between both two paradigms.

Then, we add a residual connection and layer normalization function $\mathrm{LayerNorm}(.)$ to obtain the semantic representations of the two paradigms as follows.

$$\begin{aligned}\boldsymbol{H}_{tc} &= \mathrm{LayerNorm}(\boldsymbol{H}_t + \boldsymbol{H}_{t\to g}) \\ \boldsymbol{H}_{gc} &= \mathrm{LayerNorm}(\boldsymbol{H}_g + \boldsymbol{H}_{g\to t})\end{aligned} \tag{3}$$

Finally, following the previous work (Wang et al., 2021a), Structure Self Attention (SSA) is applied to enhance the semantic representation of dialogues by incorporating the structural information of conversations. By feeding $\boldsymbol{H}_{tc}$ and $\boldsymbol{H}_{gc}$ to the SSA, we can obtain the semantic representation of all EDU pairs which incorporate the structural information of dialogues, denotes as $\boldsymbol{H}_{ts}$ and $\boldsymbol{H}_{gs}$, where $\boldsymbol{H}_{ts}, \boldsymbol{H}_{gs} \in \mathbb{R}^{n\times n\times d}$.

3.3 Transition-Based Decoding Module

We adopt a pointer network for transition-based decoding. After obtaining the semantic representations $\boldsymbol{H}_{ts}$, we first applied mean pooling on $\boldsymbol{H}_{ts}$ to obtain the semantic representation of EDUs containing structural information, denotes as $\boldsymbol{H}_{tm} \in \mathbb{R}^{n\times d}$. As shown in Figure 4, $\boldsymbol{H}_{tm}$ are fed into a uni-directional GRU for transition-based decoding, and the initial state of the decoder is taken from the combination of the last states of hierarchical GRU in both directions[2].

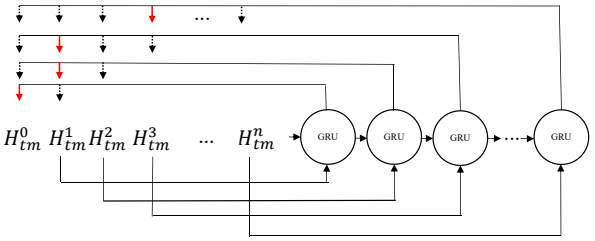

Figure 4: The decoding process of pointer network, where the solid red arrows indicate dependency links of current EDUs.

At each decoding step, it supposes the current EDU index is i at the k-th step. Then, the semantic representation $\boldsymbol{H}_{tm}$ is fed to the decoder and its output at the k-th step is $\boldsymbol{h}_{dk}$. After that, we adopted the Biaffine Attention mechanism to the representation $\boldsymbol{H}_{tm}$ and the output $\boldsymbol{h}_{dk}$ to obtain the probability between the current EDU and all previous EDUs that existing the dependency links and the corresponding relation type as follows.

$$\boldsymbol{s}_i^j = \boldsymbol{H}_{tm}{}^T\boldsymbol{W}\boldsymbol{h}_{dk} + \boldsymbol{U}\boldsymbol{H}_{tm} + \boldsymbol{V}\boldsymbol{h}_{dk} + \boldsymbol{b} \tag{4}$$

where $\boldsymbol{s}_i^j \in \mathbb{R}^m$ refers to the probability between the current EDU u_i and the previous EDU u_j and m is set to 1 when there is a dependency link between them, or set to the number of relation types when there is a dependency type between them. Besides, $\boldsymbol{W}$ denotes the weight matrix of the bi-linear term, $\boldsymbol{U}, \boldsymbol{V}$ are the two weight vectors of the linear terms, and $\boldsymbol{b}$ is the bias vector.

For a conversation with n utterances, we use the adjacency matrix $\boldsymbol{A} \in \mathbb{R}^{n\times n}$ to represent the transition-based dependency structure, where $\boldsymbol{A}_{ij} = 1$ if there is a dependency link between the EDUs u_i and u_j. Then, the adjacency matrix $\boldsymbol{A}$ is fed to DAGCN to incorporate the transition-based dependency structure.

3.4 Distance-Aware Decoding Module

In the distance-aware decoding module DADM, DAGCN is first applied to capture the dependency structure predicted by the transition-based

[2]Following previous work, we add a dummy root u_0 to represent the beginning of a dialogue.

paradigm. Then the output of DAGCN and the semantic representation of the graph-based paradigm are fused together for final decoding.

To incorporate the information of dependency links with different distances, the adjacency matrix $\boldsymbol{A}$ converted from the dependency structure, which is predicted by the transition-based paradigm, is fed to DAGCN to facilitate the decoding process of the graph-based paradigm. Inspired by Meng et al. (2020), we hypothesize that the dependency links with different distances have a different impact on the decoding process of the graph-based paradigm. Hence, the trainable weights is applies to the adjacency matrix and each weight is determined by the distance of the dependency link and the corresponding relation type from the transition-based paradigm.

Let $\boldsymbol{W}_d$ be a matrix of $\mathbb{R}^{N_d \times d'}$ where N_d is the number of different distances [3] and d' is the dimension of the embedding space. Let $\boldsymbol{W}_r$ be a matrix of $\mathbb{R}^{N_r \times d'}$ where N_r is the number of dependency relation types. The feature combination weight over the element $\boldsymbol{A}_{ij}$ in the adjacency matrix $\boldsymbol{A}$ can be represented as follows.

$$\boldsymbol{\alpha}_{ij} = \boldsymbol{W}[\mathrm{dis}(\boldsymbol{d}_{ij}) : \mathrm{rel}(\boldsymbol{r}_{ij})] \tag{5}$$

Where $\boldsymbol{d_{ij}}$ is the distance from the EDU u_i to u_j, and $\boldsymbol{r}_{ij}$ is the corresponding relation type. The functions $\mathrm{dis}(.)$ and $\mathrm{rel}(.)$are vector mapping functions, which map the one-hot vector $\boldsymbol{d}_{ij}$ and $\boldsymbol{r}_{ij}$ into the corresponding column of $\boldsymbol{W}_d$ and $\boldsymbol{W}_r$, respectively. $[:]$ indicates the concatenation operation, and $\boldsymbol{W} \in \mathbb{R}^{2d'}$ is weight matrix.

Let $\boldsymbol{A}^{'}$ be the final adjacent matrix for DAGCN, then each element of $\boldsymbol{A}^{'}$ can be computed as:

$$\boldsymbol{A}^{'}_{ij} = \boldsymbol{\alpha}_{ij}\boldsymbol{A}_{ij} \tag{6}$$

Then, we add an identity matrix to $\boldsymbol{A}^{'}$, which makes each node can connect to itself.

After obtaining the adjacent matrix $\boldsymbol{A}^{'}$, we first apply mean pooling on the semantic representation of graph-based paradigm $\boldsymbol{H}_{gs}$ to obtain $\boldsymbol{H}_{gm} \in \mathbb{R}^{n \times \frac{d}{2}}$, then the calculation of DAGCN is as follows.

$$\boldsymbol{H}_{gcn} = \mathrm{ReLU}(\boldsymbol{A}^{'}\boldsymbol{H}_{gm}\boldsymbol{W}) + \boldsymbol{H}_{gm} \tag{7}$$

[3] In our implementation, we set the distance to 2 for all distances greater than or equal to 2.

where $\boldsymbol{H}_{gcn} \in \mathbb{R}^{n \times \frac{d}{2}}$ is the output of DAGCN, which incorporate the dependency structure predicted by TDM, and $\boldsymbol{W}$ is the parameter matrix. Then, we broadcast the output of DAGCN and add it to the semantic representations $\boldsymbol{H}_{gs}$ to obtain the final representation $\boldsymbol{H}_f \in \mathbb{R}^{n \times n \times d}$ for EDU pairs.

Lastly,$\boldsymbol{H}_f$ is fed into two multi-layer perceptrons to obtain the probability distribution of each EDU pair's existing dependency link and the corresponding relation type as follows.

$$\begin{aligned} \boldsymbol{S}_l &= \mathrm{Softmax}(\mathrm{MLP}(\boldsymbol{H}_f)) \\ \boldsymbol{S}_r &= \mathrm{Softmax}(\mathrm{MLP}(\boldsymbol{H}_f)) \end{aligned} \tag{8}$$

where $\boldsymbol{S}_l \in \mathbb{R}^{n \times n \times 1}$ and $\boldsymbol{S}_r \in \mathbb{R}^{n \times n \times m}$ and m is the number of relation type. To find the highest-scoring tree, we apply greedy decoding method on $\boldsymbol{S}_l$. After determining the dependency link from u_j to u_i, the relation type can be obtained by the probability $\boldsymbol{S}_r^{ij}$.

3.5 Multi-Task Learning

For multi-task learning, we have two goals: (i) optimizing the Transition-based Paradigm (TP) and (ii) optimizing the Graph-based Paradigm (GP).

To optimize TP, we minimize the sum of the loss for constructing the right dependency structure and the loss for predicting the correct relation type, which is calculated as follows.

$$\begin{aligned} L_t(\theta_t) = &-\sum_{t=1}^{n} logP_{\theta_t}(y_t|y_{<t}, X) \\ &-\sum_{t=1}^{n}\sum_{j=1}^{m} r_{t,j} logP_{\theta_t}(r_t, X) \end{aligned} \tag{9}$$

where θ_t denotes the parameters of TP to be optimized, $y_{<t}$ represents the historical structure that has been generated at previous steps,n is the total number of dependency links, m is the total number of relation types and $r_{t,j}$ is the golden relation type.

To optimize GP, we minimize the cross-entropy of gold dependency links between EDUs pairs as follows.

$$\begin{aligned} L_g(\theta_g) = &-\sum_{i=1}^{n} y^* logP_{\theta_g}(y_t, X) \\ &-\sum_{i=1}^{n}\sum_{j=1}^{m} r_{i,j} logP_{\theta_g}(r_i, X) \end{aligned} \tag{10}$$

where θ_g denotes the parameters of GP to be optimized, y^* represents the golden dependency links,

Model		Molweni		STAC	
		Link	Link&Rel	Link	Link&Rel
w/o pre-trained model					
	DSM*	77.32	54.15	72.10	53.56
TP	LCCC	-	-	72.50	55.20
	PN	81.02	56.47	72.56	54.40
GP	SSAM*	81.15	56.93	72.92	54.83
Ours	DAMT	**82.25**	**57.35**	**73.54**	**55.32**
w/ pre-trained model					
GP	DiscProReco	-	-	74.10	57.00
	SSAM*	81.52	57.90	73.09	56.57
Ours	DAMT	**82.50**	**58.91**	**73.64**	**57.42**

Table 1: F1 scores (%) for different models where Link refers to link prediction and Link&Rel refers to that a correct prediction must predict dependency link and relation type correctly at the same time. We used the t-test with a 95% confidence interval for the significance test and all improvements of DAMT over SSAM are significant ($p < 0.05$). TP is short for Transition-based Paradigm, and GP is short for Graph-based Paradigm. $*$ indicates that we reproduce the scores of models using their released code.

Distance	Percentage(%)	
	Molweni	STAC
1	64.94	55.63
2	21.49	21.26
3	7.38	10.63
>=4	6.19	12.48

Table 2: Distribution of dependency links at different distances in the training set of Molweni and STAC.

n is the total number of dependency links, m is the total number of relation types and $r_{t,j}$ is the golden relation type.

For multi-task learning of the TP and GP, we add the above loss terms as follows.

$$L = L_t + L_g \tag{11}$$

4 Experimentation

In this section, we first introduce the datasets, hyper-parameters and baselines used in our evaluation, and then report the experimental results.

4.1 Datasets

We conduct experiments on two publicly available multi-party dialogue datasets: Molweni (Li et al., 2020) and STAC (Asher et al., 2016) and we preprocess these two datasets following Shi and Huang (2019). The distribution of dependency links at different distances on the training set of Molweni and STAC are shown in Table 2.

We can find that the dependency links with the distance 1 dominate both two corpora (about 65% and 56% in Molweni and STAC, respectively). The small percentage of long-distance dependency links poses challenges for models to predict. Our method DAMT can explicitly leverage the strengths of the transition-based paradigm to facilitate the graph-based decoding and further improve the performance on long-distance dependency links.

4.2 Hyper-Parameters

Following the previous work, we initialize words with GloVe embeddings (Pennington et al., 2014) that are fine-tuned during training. For Molweni and STAC corpus, the embedding dimension is set at 200 and 100, respectively. The dimension of the hidden representation d is set to 256 and the layer of EIM is set to 1 with 4 heads. The layer of SSA is set to 2 with 4 heads. The dimension of the embedding space d' in DAGCN is set to 8. And we set the dropout rate to 0.5 and employ Stochastic Gradient Descent (SGD) to train the model. The batch size is set to 150 and 70 for Molweni and STAC, respectively, and the initial learning rate is set to 3e-2. For the experiments using the pre-trained model, we apply XLNet-based (Yang et al., 2019) to obtain the semantic representations of EDUs and adopt AdamW to optimizer the model. The learning rate is set to 3e-4 and the dimension of the hidden representation d is set to 768. Besides, in this paper, the micro-averaged F1 score is adopted for evaluation.

4.3 Baselines

To verify the effectiveness of our DAMT, we conduct the following strong baselines for comparison.

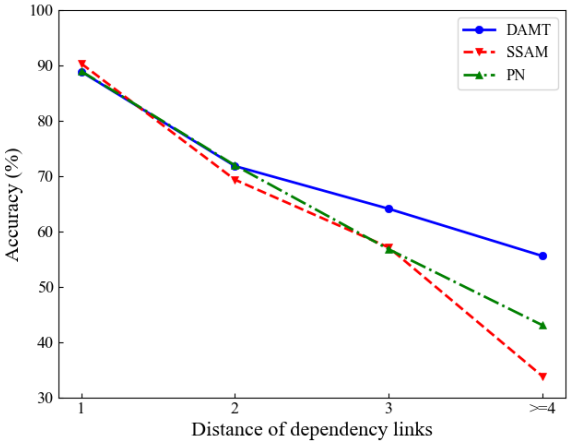

Figure 5: Comparison of dependency link accuracy at the different distance between our approach and two baselines.

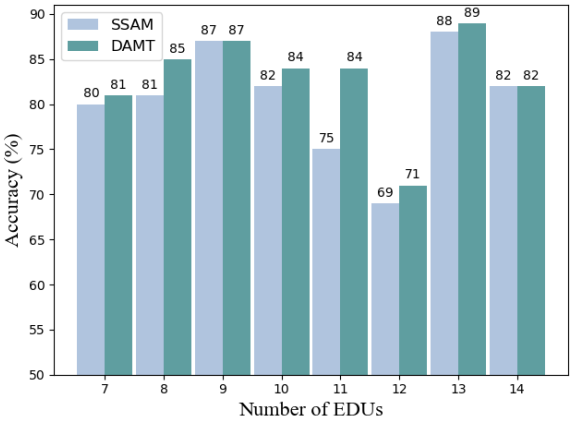

Figure 6: Accuracy of dependency links at various numbers of EDUs between our DAMT and SSAM.

Transition-based models 1) **DSM** (Shi and Huang, 2019): it predicted links and corresponding relation types jointly and alternately by considering the historical structure predicted; 2) **LCCC** (Wang et al., 2021b): it is based on DSM and enhanced the semantic representation of EDUs by incorporating cohesion information; 3) **PN**: it is the pointer network we proposed for transition-based decoding. As shown in Figure 3, our framework becomes PN after discarding the modules EIM and DADM.

Graph-based models 1) **DiscProReco** (Yang et al., 2021): it used the syntactic dependency graph to enhance the semantic representation of EDUs pairs; 2) **SSAM** (Wang et al., 2021a): it used the structure self attention network and two auxiliary training signals to enhance the semantic representation of EDU pairs.

4.4 Results

Table 1 shows the performance comparison between our DAMT and all the Transition-based and Graph-based baselines. Besides, the results without the pre-trained model and with the pre-trained model are also shown in Table 1. We can find out that our DAMT outperforms all baselines without the pre-trained model.

Especially, compared with the SOTA transition-based PN, our DAMT improves the F1-score by 1.23 and 0.88 in link and relation on Molweni, respectively, and improves them by 0.98 and 0.92 on STAC, respectively. Compared with the SOTA graph-based SSAM, our DAMT improves the F1-score by 1.10 and 0.42 in link and relation on Molweni, respectively, and improves them by 0.62 and 0.49 on STAC, respectively. Our approach DAMT leverages the strengths of different paradigms from two perspectives with multi-task learning and therefore achieves better performance.

Besides, with the application of the pre-trained model, our approach DAMT still improves the F1-score on Molweni and STAC compared to the SSAM, which illustrates the effectiveness of our approach.

5 Analysis

In this section we first conduct a detailed analysis on the dependency links with different distances and the different lengths of dialogues, and then provide the ablation study and case study.

5.1 Performance on Dependency Links with Different Distances

To further analyze the improvements of our model DAMT, we investigate the accuracy of dependency links at different distance between DAMT and two SOTA baselines on Molweni, as shown in Figure 5, where PN is a transition-based model and SSAM is a graph-based model.

Comparing with SSAM, we can find that PN performs better when the distance of dependency links is greater than one, while SSAM performs better when the distance of dependency links is one. This result shows that even though the semantic representations of EDUs are obtained using more advanced methods, both paradigms still have different performances on dependency links with different distances due to the discrepancy in dependency structure construction.

Overall, all models have a similar downward

Model	Link Accuracy (%)	
	Molweni	STAC
DAMT	66.93	50.20
-EIM	64.60(↓ 2.33)	47.60(↓ 2.60)
-DAGCN	62.91(↓ 4.02)	44.40(↓ 5.80)
-EIM,DAGCN	62.27(↓ 4.66)	43.00(↓ 7.20)

Table 3: The Accuracy of dependency links with distance greater than one in the test set of Molweni and STAC.

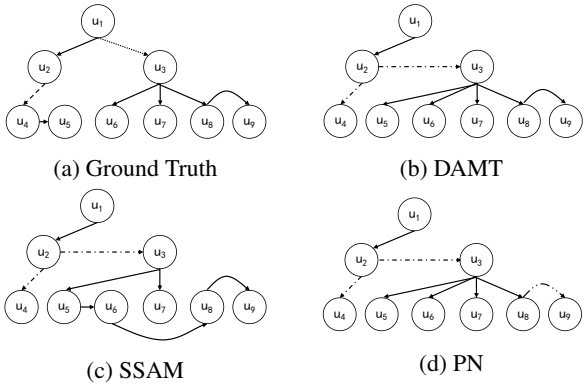

Figure 7: Discourse structures of the example in Figure 1. (a) refers to the ground truth structure and (b)-(d) refer to the structures predicted by DAMT, SSAM and PN. Different lines indicate different relation types, where the solid lines, dotted lines, dashed lines, dashed dotted lines and dashed double-dotted lines denote the relations “Clarification Question”, “Comment”, “Result”, "Question-answer_pair" and "Q-Elab", respectively.

trend. Compared with PN and SSAM, our DAMT improves the performance of dependency links with a distance greater than one significantly. It indicates the effectiveness of our model that explicitly integrates both paradigms from two perspectives.

5.2 Performance on Different Lengths of Dialogues

We further analyze the performance of DAMT and SSAM in terms of the number of EDUs on Molweni. Figure 6 shows the accuracy of dependency links at various numbers of EDUs in a document. Compared with SSAM, our model DAMT improved performance on almost all documents with different numbers of EDUs, especially those documents containing 11 and 8 EDUs with an improvement of 9 % and 4 %, respectively.

We analyze the distribution of dependency links with different distances at various numbers of EDUs. We find that the percentages of long dependency links (>1) in those conversations with 8 and 11 EDUs are the highest with 43% and 41%, respectively. This can reflect that our DAMT mainly improves the performance of the long dependency links.

5.3 Ablation Study

To investigate the impacts of the proposed modules on the performance of the dependency links with the longer distances, we conduct an ablation study on two modules EIM and DAGCN in DAMT, as illustrated in Table 3, where “-” indicates the removal of the single or several modules.

Removing any of the two modules makes the performance worse, while discarding DAGCN has the greatest impact on the performance. This shows that DAGCN can explicitly aggregate the information of dependency links with different distances in the discourse structure predicted by the PN, thus improving the performance of dependency links with the longer distance.

5.4 Case Study

Figure 7 show the dependency structures of the example in Figure 1, which are the golden truth and three predicted results by our DAMT and two baselines SSAM and PN.

Compared with the transition-based PN, we find out that SSAM can better predicts the dependency links and their corresponding relation types when the distance is one, such as $u_1 \rightarrow u_2$ and $u_8 \rightarrow u_9$. Although PN can correctly predicts the dependency link $u_8 \rightarrow u_9$, it cannot correctly predict its corresponding relation type. This shows the advantage of graph-based approach over transition-based approach for dependency links when the distance is one. On the contrary, for those dependency links with distance greater than one, PN performs better than SSAM, such as $u_3 \rightarrow u_6$ and $u_3 \rightarrow u_8$.

Compared with PN, DAMT can further correctly predict the relation type of dependency link $u_8 \rightarrow u_9$. And compared with SSAM, DAMT can correctly predict long-distance dependency links $u_3 \rightarrow u_6$, $u_3 \rightarrow u_8$. These phenomena show that our distance-aware multi-task framework is able to combine the advantages of both two paradigms to predict those long-distance dependency links.

Besides, for some dependency links, such as $u_1 \rightarrow u_3$ and $u_4 \rightarrow u_5$, all models fail to identify their links. This indicates that conversational discourse paring is still a challenging task.

6 Conclusion

In this paper, we propose a distance-aware multi-task framework DAMT to facilitate conversational discourse parsing. First, we propose an encoding interaction module to enhance the information flow between the graph-based paradigm and transition-based paradigm to promote multi-task learning. Second, we propose a distance-aware graph convolutional network DAGCN incorporating the dependency structure predicted by one paradigm to explicitly facilitate the decoding process of another paradigm. The experimental results on two public datasets show the effectiveness of our proposed DAMT. In the future, we will further explore how to recognize relation types more effectively.

7 Acknowledgments

The authors would like to thank the three anonymous reviewers for their comments on this paper. This research was supported by the National Natural Science Foundation of China (Nos. 61836007, 62276177 and 62006167), and Project Funded by the Priority Academic Program Development of Jiangsu Higher Education Institutions (PAPD).

References

Stergos Afantenos, Éric Kow, Nicholas Asher, and Jérémy Perret. 2015. Discourse parsing for multi-party chat dialogues. In *EMNLP*, pages 928–937.

Nicholas Asher, Nicholas Michael Asher, and Alex Lascarides. 2003. *Logics of conversation*. Cambridge University Press.

Nicholas Asher, Julie Hunter, Mathieu Morey, Farah Benamara, and Stergos Afantenos. 2016. Discourse structure and dialogue acts in multiparty dialogue: the stac corpus. In *LREC*, pages 2721–2727.

Lynn Carlson, Daniel Marcu, and Mary Ellen Okurowski. 2003. Building a discourse-tagged corpus in the framework of rhetorical structure theory. In *Current and new directions in discourse and dialogue*, pages 85–112. Springer.

Haihong E, Peiqing Niu, Zhongfu Chen, and Meina Song. 2019. A novel bi-directional interrelated model for joint intent detection and slot filling. In *ACL*, pages 5467–5471.

Agnieszka Falenska, Anders Björkelund, and Jonas Kuhn. 2020. Integrating graph-based and transition-based dependency parsers in the deep contextualized era. In *IWPT*, pages 25–39.

Vanessa Wei Feng and Graeme Hirst. 2014. A linear-time bottom-up discourse parser with constraints and post-editing. In *ACL*, pages 511–521.

Xiachong Feng, Xiaocheng Feng, Bing Qin, and Xinwei Geng. 2021. Dialogue discourse-aware graph model and data augmentation for meeting summarization. In *IJCAI*, pages 3808–3814.

Wenpeng Hu, Zhangming Chan, Bing Liu, Dongyan Zhao, Jinwen Ma, and Rui Yan. 2019. Gsn: A graph-structured network for multi-party dialogues. In *IJCAI*, pages 5010–5016.

Shafiq Joty, Giuseppe Carenini, and Raymond T Ng. 2015. Codra: A novel discriminative framework for rhetorical analysis. *CL*, 41(3):385–435.

Naoki Kobayashi, Tsutomu Hirao, Hidetaka Kamigaito, Manabu Okumura, and Masaaki Nagata. 2020. Top-down rst parsing utilizing granularity levels in documents. In *AAAI*, pages 8099–8106.

Fei Li, Zheng Wang, Siu Cheung Hui, Lejian Liao, Dandan Song, Jing Xu, Guoxiu He, and Meihuizi Jia. 2021a. Modularized interaction network for named entity recognition. In *ACL-IJCNLP*, pages 200–209.

Jiaqi Li, Ming Liu, Min-Yen Kan, Zihao Zheng, Zekun Wang, Wenqiang Lei, Ting Liu, and Bing Qin. 2020. Molweni: A challenge multiparty dialogues-based machine reading comprehension dataset with discourse structure. In *COLING*, pages 2642–2652.

Jiaqi Li, Ming Liu, Zihao Zheng, Heng Zhang, Bing Qin, Min-Yen Kan, and Ting Liu. 2021b. Dadgraph: A discourse-aware dialogue graph neural network for multiparty dialogue machine reading comprehension. In *IJCNN*, pages 1–8.

Yiyang Li and Hai Zhao. 2021. Self- and pseudo-self-supervised prediction of speaker and key-utterance for multi-party dialogue reading comprehension. In *Findings of EMNLP*, pages 2053–2063.

Xiang Lin, Shafiq Joty, Prathyusha Jwalapuram, and M Saiful Bari. 2019. A unified linear-time framework for sentence-level discourse parsing. In *ACL*, pages 4190–4200.

Yang Liu and Mirella Lapata. 2017. Learning contextually informed representations for linear-time discourse parsing. In *EMNLP*, pages 1289–1298.

Ryan Lowe, Nissan Pow, Iulian Vlad Serban, and Joelle Pineau. 2015. The ubuntu dialogue corpus: A large dataset for research in unstructured multi-turn dialogue systems. In *SIGDIAL*, pages 285–294.

William C Mann and Sandra A Thompson. 1987. *Rhetorical structure theory: A theory of text organization*. Information Sciences Institute, Los Angeles: University of Southern California.

Fanyu Meng, Junlan Feng, Danping Yin, Si Chen, and Min Hu. 2020. A structure-enhanced graph convolutional network for sentiment analysis. In *Findings of EMNLP*, pages 586–595, Online.

Philippe Muller, Stergos Afantenos, Pascal Denis, and Nicholas Asher. 2012. Constrained decoding for text-level discourse parsing. In *COLING*, pages 1883–1900.

Jeffrey Pennington, Richard Socher, and Christopher D Manning. 2014. Glove: Global vectors for word representation. In *EMNLP*, pages 1532–1543.

Jérémy Perret, Stergos Afantenos, Nicholas Asher, and Mathieu Morey. 2016. Integer linear programming for discourse parsing. In *NAACL*, pages 99–109.

Libo Qin, Tailu Liu, Wanxiang Che, Bingbing Kang, Sendong Zhao, and Ting Liu. 2021. A co-interactive transformer for joint slot filling and intent detection. In *ICASSP*, pages 8193–8197.

Zhouxing Shi and Minlie Huang. 2019. A deep sequential model for discourse parsing on multi-party dialogues. In *AAAI*, pages 7007–7014.

Yang Sun, Nan Yu, and Guohong Fu. 2021. A discourse-aware graph neural network for emotion recognition in multi-party conversation. In *Findings of ACL*, pages 2949–2958.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Ł ukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *NIPS*, page 5998–6008.

Ante Wang, Linfeng Song, Hui Jiang, Shaopeng Lai, Junfeng Yao, Min Zhang, and Jinsong Su. 2021a. A structure self-aware model for discourse parsing on multi-party dialogues. In *IJCAI*, pages 3943–3949.

Jinfeng Wang, Longyin Zhang, and Fang Kong. 2021b. Multi-level cohesion information modeling for better written and dialogue discourse parsing. In *NLPCC*, pages 40–52.

Yizhong Wang, Sujian Li, and Houfeng Wang. 2017. A two-stage parsing method for text-level discourse analysis. In *ACL*, pages 184–188.

Jingxuan Yang, Kerui Xu, Jun Xu, Si Li, Sheng Gao, Jun Guo, Nianwen Xue, and Ji-Rong Wen. 2021. A joint model for dropped pronoun recovery and conversational discourse parsing in Chinese conversational speech. In *ACL*, pages 1752–1763.

Zhilin Yang, Zihang Dai, Yiming Yang, Jaime Carbonell, Russ R Salakhutdinov, and Quoc V Le. 2019. Xlnet: Generalized autoregressive pretraining for language understanding. In *NIPS*, volume 32.

Longyin Zhang, Fang Kong, and Guodong Zhou. 2021. Adversarial learning for discourse rhetorical structure parsing. In *ACL-IJCNLP*, pages 3946–3957.

Linguistically Motivated Features for Classifying Shorter Text into Fiction and Non-Fiction Genre

Arman Kazmi[1] **Sidharth Ranjan**[2] **Arpit Sharma**[1] **Rajakrishnan Rajkumar**[1]
IISER Bhopal[1] IIT Delhi[2]
{kazmiarman85, sidharth.ranjan03}@gmail.com
{arpit, rajak}@iiserb.ac.in

Abstract

This work deploys linguistically motivated features to classify paragraph-level text into fiction and non-fiction genre using a logistic regression model and infers lexical and syntactic properties that help distinguish the two genres. Previous works have focused on classifying document-level text into fiction and non-fiction genres, while in this work, we deal with shorter texts which are closer to real-world applications like sentiment analysis of tweets. For the task of short-text classification on the Brown corpus, a model containing linguistically motivated features confers a substantial accuracy jump over a baseline model consisting of simple POS-ratio features found effective in previous work. The efficacy of the above model containing a linguistically motivated feature set also transfers over to another dataset viz, Baby BNC corpus. Subsequently, we compared the classification accuracy of the logistic regression model with two deep-learning models. A 1D-CNN model gives an increase of 2% accuracy over the logistic regression classifier on both datasets. A BERT-based model gives state-of-the art classification accuracies of 97% on Brown corpus and 98% on Baby BNC corpus. Although, both these deep learning models give better results in terms of classification accuracy, the problem of interpreting these models remains an open question. In contrast, regression model coefficients revealed that fiction texts tend to have more character-level diversity and have lower lexical density (quantified using content-function word ratios) compared to non-fiction texts. Moreover, subtle differences in word order exist between the two genres, *i.e.*, in fiction texts *Verbs* precede *Adverbs* in contrast to the opposite pattern in non-fiction texts (inter-alia).

1 Introduction

Written text can be classified into various categories based on its content or writing style. This paper focuses on classifying shorter texts into fiction and non-fiction genres based on their writing style. In general, the *fiction* writing has an imaginative writing style and involves non-factual prose content. In contrast, the *non-fiction* writing deals with actual events, places, and persons and is written purely based on the facts. In some cases, distinguishing between these two writing categories is easy due to the content of the text. However, classification becomes challenging in many instances due to blurry boundaries between them. For example, a short story may contain imaginary characters situated in real-life settings. Therefore it is essential to factor in the writing style of texts while classifying them into fiction and non-fiction genres.

The problem of genre identification using linguistically motivated features has been extensively investigated in NLP (see references in the rest of this section). However, the particular problem of fiction vs. non-fiction genre classification has started receiving serious attention only in recent years (Vicente et al., 2021; Qureshi et al., 2019). In the cited works, different features have been studied for classifying document-level texts (or long texts) into fiction and non-fiction genres. In contrast, very little effort has been expended to investigate the set of relevant features which are effective for the classification of shorter texts, *i.e.*, paragraph-level texts into fiction and non-fiction genres. Shorter texts or paragraph-level texts are more common on the internet and have several important practical applications like breaking news detection, opinion mining, micro-blog summarization, and discovering trending topics (Kateb and Kalita, 2015). Genre identification tools for shorter text can potentially be deployed to filter out specific categories of tweets, news headlines, product reviews, and online app reviews which have been written to manipulate or influence the users/customers. For such applications, fiction vs. non-fiction classification technology capable of analyzing writing styles can play a crucial role.

The main objective of this paper is to identify the

Proceedings of the 29th International Conference on Computational Linguistics, pages 922–937
October 12–17, 2022.

most relevant features that not only enable one to build an effective classifier but also provide deeper insights about the properties that can be used to distinguish these two genres in shorter texts. To this end, we deployed features belonging to four categories, *i.e.*, *Raw* features, *POS-ratio* features, *Lexical* features and *Syntactic* features (Karlgren and Cutting, 1994; Buongiovanni et al., 2019; Biber and Stubbs, 2002; Cleuziou and Poudat, 2007). Raw text features quantify the basic properties of the text, like sentence length and variation in sentence length within a paragraph. Lexical features are based on the statistics of words or characters present in the corpus. In writing, vocabulary plays an important role as it involves the coordination of many higher levels and lower levels of cognitive skills (Hayes, 2000; Olinghouse and Leaird, 2008). Previous studies have also used various measures of lexical diversity to discern differences between genres (Milička and Kubát, 2013; Sadeghi and Dilmaghani, 2013). We deployed a character-level diversity estimate and a lexical density estimate (ratio of content to function words). POS-ratio features proposed by Qureshi et al. (2019) compute the ratios of different parts of speech tags present in the corpus, e.g. ADVERB/NOUN, ADJECTIVE/VERB, and VERB/PRONOUN. They found these ratios to be very effective for document-level fiction vs non-fiction classification (accuracy of 96.31 % on Brown corpus text (Francis and Kučera, 1989)). While most of our features were adapted from prior work, we introduce 3 novel syntactic features extracted from parse trees in this paper. Prominently, we modelled word order variation across genres by extracting *head-dependent* bigrams (containing linear order precedence as well). Inspired from the theoretical psycholinguistics literature, we also incorporated features quantifying *syntactic complexity* (Sampson, 1997; Szmrecsanyi, 2004) as well as *argument-adjunct* patterns (Tutunjian and Boland, 2008).

We extracted the aforementioned features[1] from Brown Corpus paragraphs and performed feature selection experiments using the Recursive Feature Elimination Cross-validation algorithm (RFECV Guyon et al., 2002) on individual and combined feature sets. We report the performance of different classification models trained on several feature combinations and compare them with a baseline model with only two POS-ratio features found effective in prior work (Qureshi et al., 2019). Our classification model containing the best 28 features confers an accuracy score of 91.89% on Brown Corpus (Francis and Kučera, 1989) paragraphs with an accuracy jump of 15.56% over the baseline model containing Qureshi et al.'s simple POS ratio features (76% accuracy on short-text classification in the Brown corpus).

In order to check the transferability and generalizability of our results, we used the aforementioned model trained on the Brown corpus (American English text) to classify shorter texts obtained from the Baby BNC Corpus of British English (Consortium, 2007). Our model obtained an accuracy score of 94% which attests its utility for novel text and demonstrates how it is not biased w.r.t. language variety, *i.e.*, American English (Brown) vs British English (Baby BNC). Following previous work in the NLP literature (Worsham and Kalita, 2018; Kim, 2014; Dauphin et al., 2017), we also compared our classification results based on a traditional logistic regression model (containing handcrafted features) with 2 deep learning models. On shorter text from both Brown and BNC corpora, a 1D CNN model induces a 2% increase in accuracy score over the Logistic Regression classifier. Finally, we used a pre-trained BERT-base-uncased model (Devlin et al., 2018) resulting in state-of-the-art accuracy of 97% on Brown Corpus and 98% on Baby BNC Corpus respectively. Although both the deep learning models (CNN models and the BERT-base-uncased models) result in better results in terms of classification accuracy, they are not easily interpretable *i.e.*, linguistic properties captured by these deep learning models are not obvious.

Another issue is that CNN and BERT models are expensive to train from scratch and are more prone to overfitting when compared to the Logistic Regression classifier. On the other hand, our experimental results using simple logistic regression models are interpretable in terms of the impact of specific features. Our regression coefficients indicate that fiction texts tend to be more diverse in terms of characters and have lower lexical density than non-fiction texts. Subtle differences in word order between the two genres can also be inferred from the coefficients our dependency bigram features. For example, *Verbs* tend to precede *Adverbs* and *Pronouns* in the case of fiction texts, in contrast to the opposite pattern in non-fiction texts.

[1]Scripts to extract various linguistic features used in this work can be accessed here: https://github.com/armankazmi/Linguistic-features-of-text

Genre (#docs)	#Words	#Sentences	#Para
BROWN			
Fiction (207)	63011	4133	764
Non-Fiction (117)	89744	4024	746
BABY BNC			
Fiction (25)	140760	9601	1783
Non-Fiction (30)	34947	1327	243

Table 1: Counts of words, sentences, and paragraphs in Brown and Baby BNC corpora

Our main contribution is that we extend the work of Qureshi et al. (2019) on document-level genre classification to the problem of genre-identification for shorter text by incorporating theoretically and cognitively motivated features. Our best-performing model containing linguistically motivated features substantially outperformed their best-performing model for this novel task. The features deployed by Vicente et al. (2021) (another recent work on fiction vs non-fiction classification cited earlier) are very elaborate but are not directly connected to cognitive theories. Earlier works like Worsham and Kalita (2018) and Mendhakar (2022) analyzed various linguistic characteristics of fictional and non-fictional text but focused more on sub-genre classification within fiction and non-fiction genres.

The rest of the paper is organized as follows. In Section 2 we present details of the data sets used in this study. Section 3 provides the motivation and descriptions of the linguistic features used in this work. Section 4 describes the machine learning algorithms we deployed and the results of genre classification experiments using those algorithms. Section 5 discusses the implications of our findings. Finally, in section 6, we summarize all the results and provide pointers for future research.

2 Data and Methods

Our dataset consists of paragraphs from Brown corpus (Francis and Kučera, 1989) and Baby British National Corpus (Consortium, 2007, BNC). These corpora contain text from fiction and non-fiction genres, thus serve an important resource for our research. We set up a binary classification task to predict shorter texts into fiction and non-fiction genres. Therefore, every long document in these corpora was split into separate paragraphs based on the default paragraph annotation provided. After that, each paragraph was tagged to the class based on the class label of their parent document. To mitigate the data imbalance between the two classes since different paragraphs may have varying lengths in terms of the number of sentences, we chose only those paragraphs that had 5 or 6 sentences, and the rest were discarded. Table 1 provides more details of both the aforementioned datasets.

As a pre-processing step, we automatically tagged and parsed the paragraphs in our dataset using state-of-the-art taggers and parsers. We used Stanza (Qi et al., 2020) for parts-of-speech tagging and Stanford CoreNLP toolkit for dependency and constituency parsing (Manning et al., 2014). The punctuation marks in the paragraphs were stripped off prior to their parsing. We then extracted a wide variety of linguistic features from the tagged and parsed text, thus creating a vector representation of each paragraph. The set of features used for the classification task and the underlying motivation behind using them is described in the subsequent section. We used a traditional machine learning model (logistic regression) as well as two deep learning models (CNN and BERT) for our classification task as described in Section 4.

To further our understanding of our classification models, we tested the model's applicability in British English, where we use British National Corpus (Consortium, 2007). This way, we perform transfer learning where the model is learned on one corpus, and its applicability is tested on another corpus. This also provides a more robust way of analyzing our model's predictions. Baby BNC corpus consists of four categories: *fiction, newspaper, spoken,* and *academic*. Following Qureshi et al. (2019), we considered *academic* documents in non-fiction category and *fiction* documents in fiction category, and rest others were excluded from our primary analyses. As they mention, the news genre lacks a clear demarcation[2] in either category.

3 Linguistic features

For genre classification of shorter texts, we deployed the following four distinct categories of features in our work: 1. Raw text 2. Lexical 3. POS ratios 4. Syntactic features. These features and their motivation are described below.

3.1 Raw Text Features

Raw text features (Buongiovanni et al., 2019) are the most basic features. Following the cited work,

[2]We additionally investigate the *news* category of this corpus and report results in Appendix E to motivate future research direction.

Feature category	Feature sets (#features after RFECV)	Testing Accuracy %	F1 score (fiction: 1)	F1 score (non-fiction: 0)
Baseline	adv/adj, adj/pro	76.33 ± 1.700	0.784 ± 0.013	0.737 ± 0.024
Raw Features	avg_sen_len, std_sen_len	73.36 ± 1.64	0.740 ± 0.019	0.726 ± 0.016
Lexical Features	Character diversity (CD; 4)	81.54 ± 1.637	0.817 ± 0.017	0.813 ± 0.017
	Lexical density (lex_den)	63.89 ± 1.772	0.643 ± 0.017	0.634 ± 0.019
POS Features	POS ratios (4)	81.36 ± 1.143	0.823 ± 0.014	0.802 ± 0.01
Syntactic features	syn_comp (6)	72.98 ± 2.47	0.737 ± 0.025	0.721 ± 0.024
	Argument/Adjunct	78.15 ± 1.335	0.781 ± 0.015	0.782 ± 0.015
	dep_rel (19)	87.64 ± 1.672	0.88 ± 0.017	0.871 ± 0.017
	dep_big (36)	89.65 ± 0.553	0.899 ± 0.006	0.893 ± 0.006
Combined features	CD + POS (7)	87.01 ± 1.208	0.875 ± 0.012	0.864 ± 0.013
	CD + POS + syn_comp (8)	86.55 ± 0.984	0.869 ± 0.011	0.861 ± 0.01
	Best features (28)	91.89 ± 0.883	0.921 ± 0.009	0.916 ± 0.008

Table 2: Classification accuracy using different feature set on Brown corpus paragraphs (random baseline: $49.32 \pm 1.61\%$)

we incorporated the following measures (computed over each paragraph) as features: 1. Average sentence length (*avg_sen_len*) 2. Standard deviation of sentence lengths (*std_sen_len*)

3.2 Lexical Features

Descriptions of the two lexical features used in our approach are given below.

- *Character diversity (CD)* can be measured in various ways by establishing statistical relationships between types and tokens in the text. Generally, words are considered to be the tokens of a text, but in our case, we consider characters (excluding space) in the text as tokens[3]. Diversity establishes the statistical relationship between the type and tokens of the text and has been deployed in various applications, such as measuring the proficiency of a second language learner (Engber, 1995; Karakoç and Köse, 2017), studying the speech of people with mild aphasia (Cunningham and Haley, 2020), and analyzing the writing style of authors.

  The most common approach for measuring the diversity of characters or words is to use the ratio of unique tokens divided by the total number of tokens in a text sample, commonly known as TTR (type-token ratio). One of the shortcomings of TTR-based measures is that they depend on the sample length. Therefore, we have used seven other measures of diversity, *i.e.*, Maas Index (*Maas TTR*) (Mass, 1972), Mean segmental type-token ratio (*MsTTR*), Moving Average type-token ratio (*MATTR*) (Covington and McFall, 2010), Measure of Textual Lexical Diversity (*MTLD*) (McCarthy and Jarvis, 2007), moving average of MTLD (*MTLD MA*), VocD (Durán et al., 2004) and YulesK (Greg and Yule, 1944). The exact mathematical formula for each measure above is provided in Appendix A.

- *Lexical Density (lex_den)* features are calculated by taking the ratio of content words (words that are tagged as noun, verb, adjective adverb) to function words (all part of speech tagged words except those of content words).[4] Lexical density quantifies "how informative a text is". Prior work has argued that a text with a high number of content words carries more information than one with a higher number of function words (Johansson, 2008).

3.3 POS Ratio Features

A total of eight parts of speech ratios (*adverb/noun, adverb/pronoun, adjective/verb, noun/verb, verb/pronoun, adverb/adjective, adjective/pronoun, noun/pronoun*) were extracted from tagged datasets based on their efficacy in document-level genre classification (*i.e.*, fiction vs non-fiction) reported in prior work (Qureshi et al., 2019).

[3]Originally, we considered words as tokens and included it in the lexical feature category. However, our preliminary analysis suggested that character-level tokens performed much better than word-level tokens in our classification task, so we did not include words as tokens in the current work.

[4]Following later works, we deviate from Ure's 1971 original definition of lexical density as the ratio of number to content to all words.

3.4 Syntactic Features

The following measures were used in our analysis.

- *Frequency of dependency relations (dep_rel)*: For each parsed paragraph, we extracted the frequency of dependency relations (as defined in the Universal Dependencies framework: https://universaldependencies.org/

- *Argument-Adjunct Ratio*: Ratio of *arguments* to *adjuncts* in each paragraph. In syntactic theory, an adjunct is an optional component of a sentence, clause, or phrase, while arguments are the obligatory parts of a sentence. In psycholinguistics, the argument-adjunct distinction has been empirically demonstrated to impact parsing *i.e.*, the process of constructing syntactic representations progressively during sentence comprehension (Tutunjian and Boland, 2008).

- *Syntactic complexity (syn_comp)*: In our work, we deployed 3 different indices (capture the complexity of a sentence) proposed in prior work. Sampson (1997) defined a depth measure quantifying the degree of left-branching of a constituency parse tree (*depth*). Sampson verified the claim that English writers tend to avoid grammatical structures where the number of left branches between any word and the root node of a sentence exceeds a specific fixed limit (see Figure 3 in Appendix B for an illustration). Another way to measure the syntactic complexity of a sentence is to calculate the average dependency distance (*add*) in a sentence based on a dependency parse tree (Oya, 2011). The third measure deployed in our study is the index of Syntactic Complexity (*isc*), which is based on counts of linguistic tokens that reflect the degree of embeddings or grammatical properties of the text, such as subordinating conjunctions, Wh-Pronouns, Verb forms and Noun phrases (Szmrecsanyi, 2004). For each of the above measures, we calculated the average complexity and the standard deviation on each paragraph.

- *Dependency bigrams (dep_big)*: For each dependency parse tree corresponding to the sentences in our dataset, we extracted bigrams consisting of the POS tags of each syntactic head and dependent pair in the sentence. The linear order of each head-dependent pair was encoded via the keywords *i.e.*, *before* or *after* (see Figure 4 in Appendix C for an illustration). The main objective of this feature was to model word order variation in the text.

4 Experiments and Results

This section presents the results of our experiments for classifying shorter texts into fiction and non-fiction genre. The following subsections describe our classification results using a traditional machine learning model (logistic regression) and two deep learning models (CNN and BERT).

4.1 Logistic Regression Model

We used LOGISTIC REGRESSION (McCullagh and Nelder, 2019) as one of our classification models for the classification task. We evaluate model performance using classification accuracy and F1 score. We selected the optimal features by applying recursive feature elimination with cross-validation (RFECV) on the 4 feature sets described in the previous section. RFECV discards features from a model by fitting the model several times, removing the weakest-performing feature at each step. After obtaining the optimal features, we trained a logistic regression model with 10-fold cross-validation and L1 regularization using scikit-learn toolkit (Pedregosa et al., 2018) on the following two datasets:

1. Brown corpus with a $70\% - 30\%$ train-test split (Training paragraphs: 1057, Testing paragraphs: 453).

2. Training on Brown corpus and testing on Baby BNC corpus (Training paragraphs: 1510, Testing paragraphs: 10 different sets of 493 paragraphs).

For the first case above, we reported the mean testing accuracy with standard deviation for 10 different combinations of paragraphs in the Brown corpus. And for the second case, we trained the model on the feature vectors of 1510 paragraphs from the Brown corpus and tested it on the Baby BNC corpus. However, as presented in Table 1, the number of fiction paragraphs (1,783) in the Baby BNC corpus exceeds the number of paragraphs in the non-fiction category (243). Therefore, we randomly sampled 250 fiction paragraphs 10 times and combined each set with the 243 non-fiction paragraphs. This approach allowed us to report the

Feature set	Testing accuracy	F1 score (fiction)	F1 score (non-fiction)
Best features (28)	94.016 ± 1.03	0.939 ± 0.0112	0.941 ± 0.009
Baseline (adv/adj, adj/pro)	83.448 ± 1.12	0.843 ± 0.0123	0.824 ± 0.0098

Table 3: Classification accuracy on Baby BNC corpus trained on Brown corpus (random baseline: 50.71%)

Model	Data set	Testing Accuracy (%)	F1 score (fiction)	F1 score (non-fiction)
CNN	Brown Corpus	93.66 ± 0.808	0.939 ± 0.008	0.933 ± 0.008
	Baby BNC Corpus	96.94 ± 0.410	0.968 ± 0.004	0.969 ± 0.003
BERT (base-uncased)	Brown Corpus	97.3 ± 0.64	0.973 ± 0.006	0.972 ± 0.006
	Baby BNC Corpus	98.13 ± 0.486	0.981 ± 0.005	0.981 ± 0.005

Table 4: Classification accuracy of 1D CNN and BERT-base-uncased model on Brown and Baby BNC Corpus

mean testing accuracy with standard deviation on 10 different combinations of fiction and non-fiction paragraphs in Baby BNC corpus. The accuracy of the feature sets was compared with the baseline model containing only two features: adverb to adjective ratio (adv/adj) and adjective to pronoun ratio (adj/pro). These two features were found to be optimal for classifying document-level texts into fiction and non-fiction genre (Qureshi et al., 2019).

4.1.1 Brown Corpus

The results of our experiments on the Brown corpus are displayed in Table 2. Individually, the baseline model containing two POS ratio features (adv/adj and adj/pro) gave a classification accuracy of 76.33%. The character diversity (CD) features provided an accuracy gain of 5.21% over the baseline model. However, when CD features are combined with POS-ratio features (CD+POS), the accuracy gain increases to 10.68% over baseline. The syntactic complexity features (syn_comp) performed the worst (72.98%) compared to the baseline. The accuracy significantly improved (86.55%) when syntactic complexity features were combined with CD features and POS-ratio features (CD+POS+syn_comp). The dependency relation distribution features (dep_rel) category returned an accuracy gain of 11.31%. The dependency bigram feature (dep_big) in the syntactic feature category (89.65%) outperformed all other individual feature categories, thus suggesting the significance of our proposed word-order features in this work.

The best performing model contained 28 features after selecting the optimal features from each category after RFECV and gave an overwhelming gain of 15.56% in classification accuracy over the baseline model. Overall, our best-performing model gave a classification accuracy of 91.89%, and F1 scores for each class were similar. Table 5 in Appendix D lists all the optimal features and their regression coefficients that led to the best prediction performance. Interestingly, the feature selection algorithm eliminated both the syntactic complexity (*syn_comp*) features and argument/adjunct ratio features in the syntactic feature category. It is conceivable that dependency bigram features (*dep_big*) would be modeling those generalizations.

We also interpret the coefficients of each predictor in the best-performing regression model to understand their importance for fiction writing. The CD features have positive regression coefficients suggesting that fiction paragraphs tend to be more diverse in terms of characters than non-fiction genres. The negative regression coefficient for the lexical density feature (*content/function* ratio) indicates that fiction paragraphs tend to have lower lexical density than non-fiction paragraphs. In the case of *dep_big* features, 11 features were retained in the optimal feature set. Their coefficients suggest that the fiction paragraphs tend to have more *verbs* after *proper nouns (PROPN)* and more *verbs* preceding *adverbs*, *pronouns* and *Adpositions (ADP)*. In contrast, non-fiction paragraphs tend to have more *proper nouns (PROPN)* before *Numbers (NUM)*.

4.1.2 Baby BNC Corpus

The results of our experiments on the Baby BNC corpus are displayed in Table 3. The classifier yielded a prediction accuracy of 94.01% on the Baby BNC corpus using 28 optimal features obtained previously when the model was trained on the entire Brown corpus. The accuracy score obtained in this case is better than that of the Brown corpus data set, demonstrating that our pre-trained

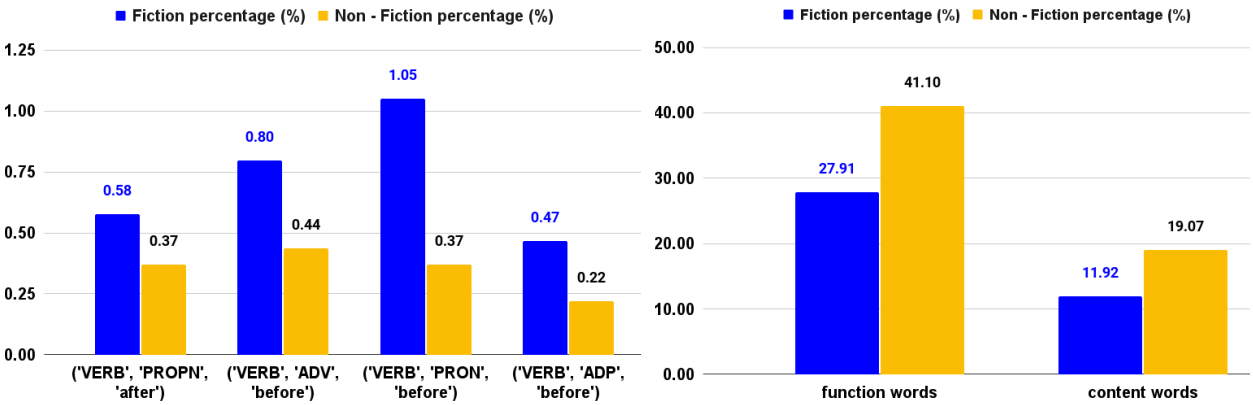

(a) Dependency bigram features (b) Lexical density features

Figure 1: Genre-wise distribution of features in the Brown corpus

model can be used for a new but related task, *i.e.,* transfer learning. It also suggests that our model is not biased w.r.t. language variety: American English vs. British English.

4.2 Deep Learning Experiments

Recent work has shown the efficacy of neural network-based language models, viz., RNN and LSTMs, for text classification (Bengio et al., 2000; Mikolov et al., 2010; Hochreiter and Schmidhuber, 1997) over traditional n-gram language models (Shannon, 1948, 1951; Chen and Goodman, 1999; Kneser and Ney, 1995; Markov, 1913). While the later models (traditional LMs) struggle with data sparsity and long-range dependencies, the former models (neural net LMs) grapple with substantial memory requirements and a long training time as they work sequentially to capture long-range dependencies (Worsham and Kalita, 2018). The former models also suffer from interpreting the various features learned during their training. In this section, we describe the deep learning experiments performed using CNN (LeCun et al., 1998) and BERT (Devlin et al., 2018) models.

4.3 CNN

We deployed a CNN-based architecture for genre classification, which is inspired by the recent work in the NLP literature (Pham et al., 2016; Prakhya et al., 2017; Dauphin et al., 2017). Recent studies have made use of the CNN-based architecture for tackling some of the challenging NLP problems, including text classification (Kim, 2014; Pham et al., 2016; Dauphin et al., 2017) and learning the abstract linguistic properties of the text, such as inflection, morphological richness, linguistic structure, and word sequence patterns (Prakhya et al., 2017; Rahman et al., 2021). Pham et al. (2016) showed that CNNs are effective in learning language representations up to the sequence of 16 words before the target and can potentially detect high-level abstract features in language data. For the task of genre classification, Worsham and Kalita (2018) compared the efficacy of various deep learning models, including CNN-Kim (Kim, 2014), LSTM (Hochreiter and Schmidhuber, 1997), Hierarchical Attention Network (Yang et al., 2016, HAN) and reported that CNN gave the most reliable performance amidst LSTM and HAN-based deep learning models.[5]

Our CNN experiments involved creating the word embedding vectors using pre-trained GloVe Embeddings. We deployed a 1D CNN model over the embedding vectors to capture the style and patterns in the paragraphs (see Appendix F for more details on training procedures). Table 4 (top block) shows the results of the CNN-based models on Brown and Baby BNC corpora. This model obtains an accuracy score of 93.66% on Brown corpus and 96.94% on Baby BNC corpus.

4.4 BERT

We deployed the BERT-base-uncased model (Devlin et al., 2018) for our genre classification task on shorter texts. The bidirectional encoder representations from transformers (BERT) is an NLP model designed to capture bidirectional represen-

[5]Interestingly, Worsham and Kalita (2018) showed that the XGBoost classifier (Chen and Guestrin, 2016) outperformed every other model deployed for their genre classification task. XGBoost model is based on a tree-based classification algorithm with bag-of-words (BOW) input representation and is known to take the least time and utilize fewer resources.

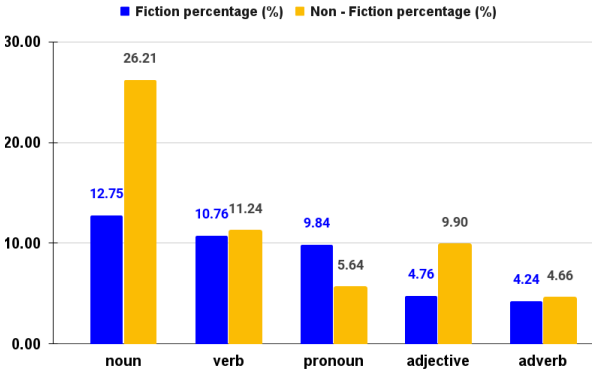

Figure 2: Genre-wise distribution of parts-of-speech (POS) tags in Brown corpus

tation from the unlabeled raw text. Then they are fine-tuned on labeled textual data to carry out various NLP tasks (Vaswani et al., 2017). Appendix G provides more details on the BERT training procedures for our classification task. Table 4 (bottom block) shows the results of the BERT model on Brown and Baby BNC corpora. This model obtains an accuracy score of 97.0% on Brown corpus and 98.13% on Baby BNC corpus.

5 Discussion

Overall, our classification results show that deep learning models achieve better accuracy than the traditional machine learning model for the task of genre identification in shorter text. However, one of the main objectives of this work is to identify the properties of the text that help distinguish between fiction and non-fiction genres. This objective is accomplished using the traditional logistic regression model as the regression coefficients enable interpretation of the features used in the model. Though the deep learning models confer better accuracy on our task, it is difficult to interpret them. The CNN and BERT models used in our work were trained on thousands of parameters. Word embedding vectors capture the properties of the words in the texts, but the complex structure of the internal representations of these models make it difficult to discern the exact generalizations learned by the models in question.

In order to understand the impact of various features in distinguishing between fiction and non-fiction genres, we interpret the regression coefficients of our logistic regression model (depicted in Table 2), by examining the model containing 28 best features (Table 5 of Appendix D). The positive regression coefficient associated with the character diversity feature indicates that fictional text tends to be more diverse in terms of characters than non-fiction text. However, fiction paragraphs have a lower lexical density (negative coefficient of *content/function* ratio) compared to non-fiction ones. This last finding has implications for theories of language production and comprehension, as suggested by a close reading of prior work on sentence processing. Schmauder et al. (2000) showed that during silent reading, both content and function words are processed similarly during the early stages of lexical processing and differently in the latter stages, where words are integrated with other elements of the text (including discourse representations). However, in spontaneous speech, Bell et al. (2009) showed that backward and forward bigram probabilities displayed asymmetric behavior in predicting content and function words, leading to the conclusion that these word types are accessed differently in production.

Dependency relations also provide important cues that help distinguish between fiction and non-fiction genres. Fiction is characterized by a greater number of syntactic subjects, oblique noun modifiers, *'s* possessives, and discourse markers compared to non-fiction. In contrast, non-fiction texts are characterized by a greater frequency of numeral modifiers, passive voice sentences, relative clauses, and multi-word expressions. Further, subtle word order differences between the two genres act as effective predictors of paragraph-level genres. In fictional paragraphs, *verbs* tend to precede *adverbs* and *pronouns* while *proper nouns* are likely to occur before *verbs*. In non-fiction paragraphs, *proper nouns (PROPN)* and *verbs* tend to precede *numbers (NUM)* and *pronouns* precede *verbs*. See Figure 1(a) for a visual illustration of the above patterns in the Brown corpus genres. Similarly, Figure 1(b) depicts the genre-wise percentage of content and function words computed over all the words in the Brown corpus. It suggests that the content and function word percentages in the non-fiction genre are greater than in the fiction genre. Further, Figure 2 represents the percentage of genre-specific parts of speech tags (computed over the total number of Brown corpus parts of speech tags), where the percentage of nouns in fiction is greater than that in non-fiction while *adverbs* and *verbs* have similar distributions across both genres.

In the rest of this section, we illustrate the impor-

tance of our features using linguistic examples (the verb is shown in bold, and the adverb is in italics). Given below are the first two sentences taken from the Brown corpus fiction paragraphs (fileid: cn13, paragraph number: 22):

- The snake **slid** *slowly* and with great care from the new ridge the plow had made , into the furrow and did not **go** any *further*.
- He was multi-colored and graceful and he lay in the furrow and **moved** his arched and tapered head only so *slightly*.

These sentences depict the case where *verbs* precede *adverbs* in fictional text. These examples also indicate how the later adverb plays a crucial role in providing extra information about the verb, thus augmenting the imaginative quotient of the text. We prove two further examples below from non-fiction texts taken from Brown Corpus (fileid: ce16, paragraph number: 17), which shows the genre-specific tendency of *adverbs* preceding *verbs*:

- I laid three layers of glass cloth on the inside of the stem, *also* **installing** a bow eye at this time.
- *Again*, these blocks were **set** in resin-saturated glass cloth and nailed .

Finally, we also checked the performance of all the models (traditional as well as deep learning models) on newswire text from the Baby BNC Corpus (605 paragraphs from the 97 News category documents) individually as well as combined with our Baby BNC dataset (described in Section 2 used in previous experiments). Table 6 in the Appendix section E shows the performance of different models on the Baby BNC corpus when the news texts are included in the non-fiction category. Table 7 of Appendix E shows the percentage of news texts classified as non-fiction using different models. Our traditional model has 68% accuracy while classifying news texts into the non-fiction genre. Even the top-performing BERT-base-uncased model gives a classification accuracy of 83%. The performance drop of these models signifies that news texts contain writing styles that require more detailed linguistic analyses as features found effective for other sub-genres (such as academic texts) fail to achieve a comparable accuracy in this case.

6 Conclusions and Future Work

In this work, we classified paragraph-level text into fiction and non-fiction genres using a traditional machine learning model (logistic regression) and two different deep learning models. For short-text genre identification, we show that the traditional model containing hand-crafted features (raw text, POS ratios, lexical and syntactic features) significantly outperformed a baseline model containing POS-ratio features, (originally proposed by Qureshi et al. (2019) for the task of document-level genre classification). We also obtained the insight that subtle differences in word order exist between the two genres, *i.e.*, in fiction texts *Verbs* precede *Adverbs* (inter-alia) compared to non-fiction texts. Finally, we showed that deep learning models (*viz.*, CNN and BERT) perform significantly better than our traditional model. We obtained state-of-the-art results for the task of short-text genre identification using a pre-trained BERT model fine-tuned on the Brown Corpus.

In future work, we intend to investigate the efficacy of the hand-crafted features on a larger data set and also plan to create a gold standard corpus of human-annotated fiction and non-fiction paragraphs for fine-grained evaluation. Future research needs to investigate whether *syntactic complexity* and *arguments/adjuncts* patterns (not having any impact using our current machine learning setup) are effective predictors of genre shorter texts using other learning algorithms. Further, our finding that journalistic prose (as in news) is not purely non-fiction in nature and might contain fictional elements, needs more systematic investigation. Another line of future inquiry is to combine traditional models (encoding linguistic features) with state-of-the-art deep learning models. Finally, it would be interesting to investigate if causality expressed in the natural language text plays an essential role in classifying text into fiction and non-fiction genres.

Acknowledgements

We would like to thank the COLING-2022 anonymous reviewers, Rupesh Pandey, Kushal Shah, Rameez Qureshi, and Arman Kazmi's MS thesis evaluation committee, for their invaluable comments and feedback on this work. The fourth author also wishes to acknowledge extramural funding from the Cognitive Science Research Initiative, Department of Science and Technology, Government of India (grant no. DST/CSRI/2018/263).

References

Alan Bell, Jason M Brenier, Michelle Gregory, Cynthia Girand, and Dan Jurafsky. 2009. Predictability effects on durations of content and function words in conversational English. *Journal of Memory and Language*, 60(1):92–111.

Yoshua Bengio, Réjean Ducharme, and Pascal Vincent. 2000. A neural probabilistic language model. *Advances in neural information processing systems*, 13.

Douglas Biber and Michael Stubbs. 2002. Dimensions of register variation: A cross-linguistic comparison douglas biber. *Language*, 75.

Chiara Buongiovanni, Francesco Gracci, Dominique Brunato, and Felice Dell'Orletta. 2019. Lost in text. a cross-genre analysis of linguistic phenomena within text. *Italian Journal of Computational Linguistics*.

Stanley F Chen and Joshua Goodman. 1999. An empirical study of smoothing techniques for language modeling. *Computer Speech & Language*, 13(4):359–394.

Tianqi Chen and Carlos Guestrin. 2016. Xgboost: A scalable tree boosting system. In *Proceedings of the 22nd acm sigkdd international conference on knowledge discovery and data mining*, pages 785–794.

Guillaume Cleuziou and Céline Poudat. 2007. On the impact of lexical and linguistic features in genre- and domain-based categorization. In *CICLing*.

BNC Consortium. 2007. British national corpus, baby edition. Oxford Text Archive.

Michael A. Covington and Joe D. McFall. 2010. Cutting the gordian knot: The moving-average type–token ratio (mattr). *Journal of Quantitative Linguistics*, 17:100 – 94.

Kevin T Cunningham and Katarina L Haley. 2020. Measuring lexical diversity for discourse analysis in aphasia: Moving-average type–token ratio and word information measure. *Journal of Speech, Language, and Hearing Research*, 63(3):710–721.

Yann N Dauphin, Angela Fan, Michael Auli, and David Grangier. 2017. Language modeling with gated convolutional networks. In *International conference on machine learning*, pages 933–941. PMLR.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. BERT: pre-training of deep bidirectional transformers for language understanding. *CoRR*, abs/1810.04805.

Pilar Durán, David Malvern, Brian Richards, and Ngoni Chipere. 2004. Developmental trends in lexical diversity. *Applied Linguistics*, 25.

Cheryl A Engber. 1995. The relationship of lexical proficiency to the quality of esl compositions. *Journal of second language writing*, 4(2):139–155.

W.N. Francis and H. Kučera. 1989. *Manual of Information to Accompany a Standard Corpus of Present-day Edited American English: For Use with Digital Computers*. Brown University, Department of Linguistics.

Walter Wilson Greg and G. Udny Yule. 1944. The statistical study of literary vocabulary. *Modern Language Review*, 39:291.

Isabelle Guyon, Jason Weston, Stephen Barnhill, and Vladimir Vapnik. 2002. Gene selection for cancer classification using support vector machines. *Machine Learning*, 46:389–422.

John R Hayes. 2000. Understanding cognition and affect in writing. *Perspectives on writing: Research, theory, and practice*, pages 6–44.

Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long short-term memory. *Neural computation*, 9(8):1735–1780.

Victoria Johansson. 2008. Lexical diversity and lexical density in speech and writing: A developmental perspective. *Working papers/Lund University, Department of Linguistics and Phonetics*, 53:61–79.

Dilek Karakoç and Gül Durmuşoğlu Köse. 2017. The impact of vocabulary knowledge on reading, writing and proficiency scores of efl learners. *Journal of Language and Linguistic Studies*, 13:352–378.

Jussi Karlgren and Douglass Cutting. 1994. Recognizing text genres with simple metrics using discriminant analysis. In *COLING 1994 Volume 2: The 15th International Conference on Computational Linguistics*.

Faris Kateb and Jugal Kalita. 2015. Article: Classifying short text in social media: Twitter as case study. *International Journal of Computer Applications*, 111(9):1–12. Full text available.

Yoon Kim. 2014. Convolutional neural networks for sentence classification. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1746–1751, Doha, Qatar. Association for Computational Linguistics.

Reinhard Kneser and Hermann Ney. 1995. Improved backing-off for m-gram language modeling. In *1995 international conference on acoustics, speech, and signal processing*, volume 1, pages 181–184. IEEE.

Yann LeCun, Léon Bottou, Yoshua Bengio, and Patrick Haffner. 1998. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324.

Christopher Manning, Mihai Surdeanu, John Bauer, Jenny Finkel, Steven Bethard, and David McClosky. 2014. The Stanford CoreNLP natural language processing toolkit. In *Proceedings of 52nd Annual Meeting of the Association for Computational Linguistics: System Demonstrations*, pages 55–60, Baltimore, Maryland. Association for Computational Linguistics.

A. A. Markov. 1913. Essai d'une recherche statistique sur le texte du roman "Eugene Onegin" illustrant la liaison des epreuve en chain ('Example of a statistical investigation of the text of "Eugene Onegin" illustrating the dependence between samples in chain'). *Izvistia Imperatorskoi Akademii Nauk (Bulletin de l'Académie Impériale des Sciences de St.-Pétersbourg)*, 7:153–162. English translation by Morris Halle, 1956.

Heinz-Dieter Mass. 1972. Über den zusammenhang zwischen wortschatzumfang und länge eines textes. *Zeitschrift für Literaturwissenschaft und Linguistik*, 2(8):73.

Philip M. McCarthy and Scott Jarvis. 2007. vocd: A theoretical and empirical evaluation. *Language Testing*, 24:459 – 488.

Peter McCullagh and John A Nelder. 2019. *Generalized linear models*. Routledge.

Akshay Mendhakar. 2022. Linguistic profiling of text genres: An exploration of fictional vs. non-fictional texts. *Information*, 13(8).

Tomas Mikolov, Martin Karafiát, Lukas Burget, Jan Cernockỳ, and Sanjeev Khudanpur. 2010. Recurrent neural network based language model. In *Interspeech*, volume 2, pages 1045–1048. Makuhari.

Jiří Milička and Miroslav Kubát. 2013. Vocabulary richness measure in genres. *Journal of Quantitative Linguistics*, 20:339–349.

Natalie Olinghouse and Jacqueline Leaird. 2008. The relationship between measures of vocabulary and narrative writing quality in second- and fourth-grade students. *Reading and Writing*, 22:545–565.

Masanori Oya. 2011. Syntactic dependency distance as sentence complexity measure. *Proceedings of the 16th International Conference of Pan-Pacific Association of Applied Linguistics*.

Fabian Pedregosa, Gaël Varoquaux, Alexandre Gramfort, Vincent Michel, Bertrand Thirion, Olivier Grisel, Mathieu Blondel, Andreas Müller, Joel Nothman, Gilles Louppe, Peter Prettenhofer, Ron Weiss, Vincent Dubourg, Jake Vanderplas, Alexandre Passos, David Cournapeau, Matthieu Brucher, Matthieu Perrot, and Édouard Duchesnay. 2018. Scikit-learn: Machine learning in python.

Ngoc-Quan Pham, German Kruszewski, and Gemma Boleda. 2016. Convolutional neural network language models. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 1153–1162.

Sridhama Prakhya, Vinodini Venkataram, and Jugal Kalita. 2017. Open set text classification using CNNs. In *Proceedings of the 14th International Conference on Natural Language Processing (ICON-2017)*, pages 466–475, Kolkata, India. NLP Association of India.

Peng Qi, Yuhao Zhang, Yuhui Zhang, Jason Bolton, and Christopher D. Manning. 2020. Stanza: A Python natural language processing toolkit for many human languages. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics: System Demonstrations*.

Mohammed Rameez Qureshi, Sidharth Ranjan, Rajakrishnan Rajkumar, and Kushal Shah. 2019. A simple approach to classify fictional and non-fictional genres. In *Proceedings of the Second Workshop on Storytelling*, pages 81–89, Florence, Italy. Association for Computational Linguistics.

Chowdhury Rafeed Rahman, MD Rahman, Mohammad Rafsan, Samiha Zakir, Mohammed Eunus Ali, and Rafsanjani Muhammod. 2021. Revisiting cnn for highly inflected bengali and hindi language modeling. *arXiv preprint arXiv:2110.13032*.

Karim Sadeghi and Sholeh Dilmaghani. 2013. The relationship between lexical diversity and genre in iranian efl learners' writings. *Journal of Language Teaching and Research*, 4.

Geoffrey Sampson. 1997. Depth in english grammar. *Journal of Linguistics*, 33:131–151.

A. René Schmauder, Robin K. Morris, and David V. Poynor. 2000. Lexical processing and text integration of function and content words: Evidence from priming and eye fixations. *Memory & Cognition*, 28(7):1098–1108.

Claude E Shannon. 1951. Prediction and entropy of printed english. *Bell system technical journal*, 30(1):50–64.

Claude Elwood Shannon. 1948. A mathematical theory of communication. *The Bell system technical journal*, 27(3):379–423.

Benedikt Szmrecsanyi. 2004. On operationalizing syntactic complexity. In *Le poids des mots. Proceedings of the 7th international conference on textual data statistical analysis. Louvain-la-Neuve*, volume 2, pages 1032–1039.

Damon Tutunjian and Julie Boland. 2008. Do we need a distinction between arguments and adjuncts? evidence from psycholinguistic studies of comprehension. *Language and Linguistics Compass*, 2:631–646.

Jean Ure. 1971. Lexical density and register differentiation. *Applications of linguistics*, 23(7):443–452.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need.

Marta Vicente, María Miró Maestre, Elena Lloret, and Armando Suárez Cueto. 2021. Leveraging machine learning to explain the nature of written genres. *IEEE Access*, 9:24705–24726.

Joseph Worsham and Jugal Kalita. 2018. Genre identification and the compositional effect of genre in literature. In *Proceedings of the 27th International Conference on Computational Linguistics*, pages 1963–1973, Santa Fe, New Mexico, USA. Association for Computational Linguistics.

Zichao Yang, Diyi Yang, Chris Dyer, Xiaodong He, Alex Smola, and Eduard Hovy. 2016. Hierarchical attention networks for document classification. In *Proceedings of the 2016 conference of the North American chapter of the association for computational linguistics: human language technologies*, pages 1480–1489.

Appendix

A Mathematical formulae for lexical diversity

- **Maas Index:** This measure minimizes the length dependence of TTR by linearizing. Conceptually, this method is based on the notion that a logarithmic curve can reasonably fit the TTR curve (Mass, 1972; McCarthy and Jarvis, 2007).

$$a^2 = \frac{(logTokens - logTypes)}{log^2Tokens} \tag{1}$$

- **MSTTR:** The mean segmental type-token ratio is a metric that divides a text into equal segments based on the amount of words in each segment (normally 50 or 100 words per segment). The TTR is determined for each segment, and the MSTTR is generated by taking the arithmetic mean of the TTR for each segment.

- **MATTR:** The moving average type-token ratio is a measure that involves moving a fixed-size window through the text and calculating the type-token ratio for each window position. To begin, a window length—for example, 50 words—is chosen, and the type-token ratio for words 1–50 is calculated. The type-token ratio is then computed for words 2–51, 3–52, and so on until the text length is reached. The estimated TTRs are averaged for the final score (Covington and McFall, 2010).

- **MTLD:** The measure of textual lexical diversity is defined as the average number of words in a row for which a specified type-token ratio is maintained (here 0.720). When the value falls below a cut-off score (here 0.720), a count (called the factor count) increases by one, and the TTR assessments are reset. It picks up where the value was dropped and repeats the operation until the text is finished. The entire number of words in the text is then divided by the total number of factors in the text. After that, the entire text in the language sample is reversed, and a new MTLD score is calculated. The forward and the reversed MTLD scores are averaged to provide the final MTLD estimate (McCarthy and Jarvis, 2007). One more measure of MTLD was also calculated: Moving Average MTLD (procedure same as that of MATTR).

- **Voc-D:** The vocabulary diversity is a result of a series of random text samplings. It measures the rate at which TTR drops in the sample. To calculate Voc-D, 35 tokens are randomly selected from the text without being replaced, and the TTR is calculated. The average TTR for 35 tokens is estimated and this method is repeated 100 times. Similarly, the average TTR for 36-50 tokens is determined. The means of each of these samples are then used to generate an empirical TTR curve. Using the least-squares approach, a theoretical curve is created that maximizes its fit to the empirical TTR curve. The TTR calculated using Voc-D, 'D' is as follows (Durán et al., 2004).

$$TTR = \frac{D}{N}[(1 + 2\frac{N}{D})^{\frac{1}{2}} - 1] \tag{2}$$

- **Yule's K:** This measures the repetition and lower values of Yule's K represent higher diversities (Greg and Yule, 1944). The value K for a text sample is calculated as follows.

$$K = 10^4 \frac{\{\sum_{r=1}^{N} V_r r^2\} - N}{N^2} \tag{3}$$

where V_r is the number of types that occur r times in a text of length N.

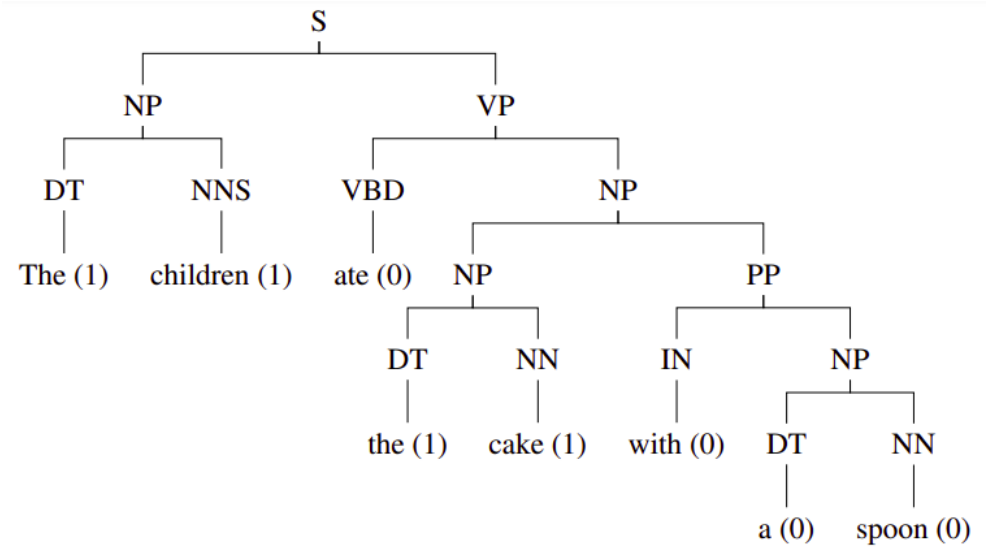

Figure 3: Constituency parse tree indicating Sampson's depth for each word alongside in bracket

B Syntactic depth-based feature calculation

For the example sentence, **"The children ate the cake with a spoon"**, we describe the method to compute Sampson's (1997) measure of syntactic complexity. Figure 3 illustrates the constituency parse tree of the example sentence. Now, we define the LINEAGE of a word as the class of nodes, including the leaf node (terminal node) associated with that word, the root node of its tree, and all the intermediate nodes on the unique path between leaf and root nodes. Now, according to Sampson depth of a terminal node is defined as the total number of those non-terminal nodes in the word's lineage with at least one younger sister.[6] The depth of the word 'cake' in the example sentence according to the above definition is 1. Similarly, the depth of each terminal node or the word in the sentence in sequence from left to right is 1,1,0,1,1,0,0,0, which sums to 4. The depth-based measure is the average over the leaf nodes; hence, the value is 0.5 in this example.

C Dependency bigram feature calculation

For the example sentence, **"The children ate the cake with a spoon"**, we describe the method to compute the dependency bigram feature modelling word order patterns. We extracted bigram features from dependency trees (exemplified in Figure 4). We took all the pos tags of head and dependent pairs from the tree, and specified the position of syntactic heads w.r.t to their dependents in a linear string by means a keyword: *before* or *after*. Therefore, the dependency bigram features for the example sentence are: *(NOUN, DET, after), (VERB, NOUN, after), (VERB, NOUN, before), (VERB, NOUN, before), (NOUN, DET, after), (NOUN, SCONJ, after), (NOUN, DET, after)*

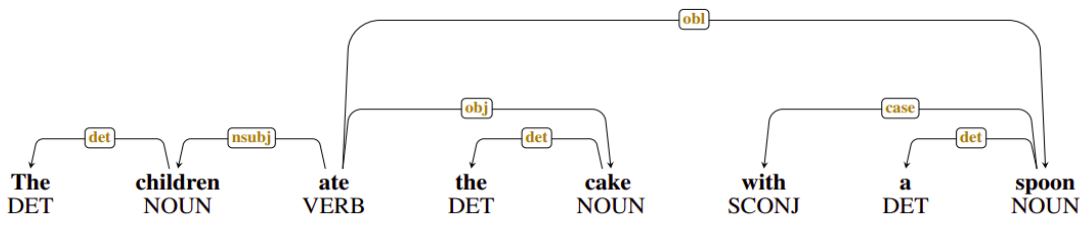

Figure 4: Dependency parse tree to compute the word order based feature.

[6]Node e is a YOUNGER SISTER of a node d if d and e are immediately dominated by the same mother node and e is further right than d.

D Supplementary Information: Brown corpus results

Feature category	Feature set	Feature Name	Regression Coefficient
Raw features	-	std_sen_len	0.11
Lexical features	Character diversity (CD)	TTR	2.31
		Maas TTR	1.70
		VocD	0.38
	Lexical density (lex_den)	content/function	-5.83
POS Features	POS ratios	adverb/pronoun	-0.19
		noun/verb	-0.68
Syntactic features	Dependency relation counts (dep_rel)	discourse	1.17
		nsubj	0.43
		obl:npmod	0.20
		nmod:poss	0.19
		nummod	-0.12
		mark	-0.18
		aux:pass	-0.20
		flat	-0.33
		acl:relcl	-0.49
		fixed	-0.70
	Dependency bigrams (dep_big)	('VERB', 'PROPN', 'after')	0.59
		('VERB', 'ADV', 'before')	0.44
		('VERB', 'PRON', 'before')	0.31
		('VERB', 'ADP', 'before')	0.31
		('PROPN', 'PROPN', 'after')	-0.30
		('VERB', 'SCONJ', 'after')	-0.31
		('VERB', 'NUM', 'before')	-0.45
		('PRON', 'NOUN', 'before')	-0.57
		('PRON', 'VERB', 'before')	-0.74
		('ADJ', 'SCONJ', 'after')	-0.79
		('PROPN', 'NUM', 'before')	-1.46

Table 5: Regression coefficients of the features from our best model containing 28 optimal features

E Supplementary Information: Complete Baby BNC corpus (incl newswire text) results

Model	Testing Accuracy (%)	F1 Score (fiction)	F1 score (non-fiction)
traditional model (28 best features)	83.77 +/- 0.291	0.848 +/- 0.003	0.825 +/- 0.003
GloVe Embedding CNN	87.034 +/- 0.467	0.876 +/- 0.005	0.863 +/- 0.004
BERT-base-uncased	92.52 +/- 0.258	0.927 +/- 0.003	0.922 +/- 0.002
Qureshi et al. (2 best ratio features)	72.95 % +/- 0.4	0.765 +/- 0.004	0.681 +/- 0.003

Table 6: Testing accuracy on Baby BNC corpus including the news texts (Non-Fiction: 848 paragraphs; Fiction: 850 paragraphs; most frequent baseline is 50.05%)

Models	Percentage of samples classified as Non-Fiction
Traditional model (28 best features)	68.42
GloVe Embedding CNN	82.97
BERT-base-uncased	83.3
Qureshi et al. baseline (2 best ratio features)	49.1

Table 7: Percentage of news texts in Baby BNC corpus classified as non-fiction; total samples: 605 news paragraphs

F CNN Training Regime

The pre-trained model of glove embeddings were trained on a text dataset consisting of Wikipedia articles and Gigaword-5 data (collection of newswire texts). The pre-trained vectors were trained on a total of 6B tokens and 400K vocabulary with different embedding dimensions. The model outputs an embedding vector of dimension 100 for each word of the tokenized text. However, depending on the length of tokenized text, the output vectors could be of different lengths. Thus could potentially create an imbalance problem while feeding the vectors into a deep learning model as the input text may not be of fixed length. To overcome this issue, we fixed the length of the input to be the longest sequence length available in the training data. As a result, we obtained an input of constant size for training and testing. The embedding dimension received from each paragraph input is 292×100, where 292 is the fixed maximum length of the tokenized paragraph text in the training data, and 100 is the dimension of each word embedding vector. We pass this embedding layer into a 1D CNN model with 100 filters each of size 3 (also known as kernel size). The activation function used with the CNN layer is 'ReLu'. The output of this layer goes to a global max pooling layer that returns the max value of the input vector received. The output of the global pooling goes to a dense layer of size 10 with the 'ReLu' activation function. Finally, predictions are made using a dense layer of size 1 and a sigmoid activation function, which transforms its output to class probability.

G BERT Training Regime

The BERT-base model (Vaswani et al., 2017) contains an encoder with 12 transformer blocks, 12 self-attention heads, and a hidden size of 768. BERT generates a representation of the sequence from an input sequence raning up to 512 tokens. The sequence consists of one or two segments, with the [CLS] token serving as the sequence's first token and containing the special classification embedding. [SEP] serves as the sequence's second token and is used to separate segments. For text classification, BERT takes the final hidden state of the token [CLS], where the entire sequence information is encoded in this particular token. In the last step, a simple softmax classifier is added to the top of BERT to predict the probability of target labels. We use the BERT-base-uncased model (Devlin et al., 2018) having a hidden size of 768, 12 transformer blocks 12 self-attention heads. The maximum length of the BERT model input was fixed to 512. We then fine-tuned the BERT model with a batch size of 10, a learning rate of 2e-5, and a weight decay of 0.01. The hidden dropout probability was 0.1. We set the maximum number of epochs to 3 and saved the best model for evaluation.

Semantic Sentence Matching via Interacting Syntax Graphs

Chen Xu[1], **Jun Xu**[1,2,*], **Zhenghua Dong**[3], **Ji-Rong Wen**[1,2]
[1]Gaoling School of Artificial Intelligence, Renmin University of China
[2]Beijing Key Laboratory of Big Data Management and Analysis Methods
[3]Huawei Noah's Ark Lab
{xc_chen, junxu, jrwen}@ruc.edu.cn
dongzhenhua@huawei.com

Abstract

Studies have shown that the sentence's syntactic structures are important for semantic sentence matching. A typical approach is encoding each sentence's syntactic structure into an embedding vector, which can be combined with other features to predict the final matching scores. Though successes have been observed, embedding the whole syntactic structures as one vector inevitably overlooks the fine-grained syntax matching patterns, e.g. the alignment of specific term dependencies relations in the two inputted sentences. In this paper, we formalize the task of semantic sentence matching as a problem of graph matching in which each sentence is represented as a directed graph according to its syntactic structures. The syntax matching patterns (i.e. similar syntactic structures) between two sentences, therefore, can be extracted as the sub-graph structure alignments. The proposed method, referred to as Interacted Syntax Graphs (ISG), represents two sentences' syntactic alignments as well as their semantic matching signals into one association graph. After that, the neural quadratic assignment programming (QAP) is adapted to extract syntactic matching patterns from the association graph. In this way, the syntactic structures fully interact in a fine granularity during the matching process. Experimental results on three public datasets demonstrated that ISG can outperform the state-of-the-art baselines effectively and efficiently. The empirical analysis also showed that ISG can match sentences in an interpretable way.

1 Introduction

Matching two natural language sentences has become a fundamental technique in information retrieval (IR) and natural language processing (NLP). Extensive research efforts have been devoted to the task (Li and Xu, 2014; Xu et al., 2020). Recently, researchers found that the sentences' rich syntactic information helps match. Some studies utilize the implicit syntax-encoding methods to learn the sentence embedding based on its syntactic strictures (Chen et al., 2016, 2017). Other studies directly utilize the different syntactic tags as syntactic features and fuse them with word features (Mou et al., 2016; Chen et al., 2018; Liu et al., 2020). All these approaches separately represent the syntactic information of the two input sentences as two coarse-grained embedding vectors, ignoring the fine-grained matching patterns between the syntactic structures.

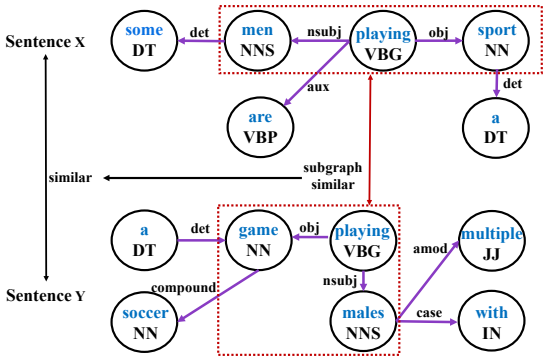

Figure 1: A semantically similar sentence pair (X = "Some men are playing a sport", Y ="A soccer game with multiple males playing") from SNLI. The words and POS tags are shown in the nodes, and syntactic dependencies are shown as edges.

In sentence matching, The fine-grained syntax structures are lost during the process of representing the syntactic information as a vector. Studies in (Xu et al., 2020) also verified that representation-based methods will lose the fine-grained matching signals. Figure 1 illustrates two semantically similar sentences from the SNLI dataset where sentences X = "Some men are playing a sport", Y ="A soccer game with multiple males playing". The parsed syntactic structures (the POS tags and syntactic dependencies) are represented as nodes and edges in two graphs, respectively. We can see that though the overall syntactic struc-

* Corresponding author

Proceedings of the 29th International Conference on Computational Linguistics, pages 938–949
October 12–17, 2022.

tures of the two sentences are very different, they still contains matched sub-graphs: "men(NNS) $\xleftarrow{\text{nsubj}}$playing(VBG)$\xrightarrow{\text{obj}}$sport(NN)" from X and "game(NN) $\xleftarrow{\text{obj}}$playing(VBG)$\xrightarrow{\text{nsubj}}$males(NNS)" from Y. The matched sub-graphs provide a crucial syntactic matching signal for downstream models to make high-precision matching decisions. Existing approaches, however, inevitably overlook the fine-grained sub-graph matching signal because the sub-graph details are lost during the embedding of the whole syntax graphs. Specifically, the sub-graph details denote the combination of first-order (e.g. Part-of-Speech) and second-order (e.g. word dependency) syntactic information.

In this paper, we propose to formalize the task of sentence matching as matching two directed graphs where each graph corresponds to one sentence, and its edges and nodes correspond to the word-word syntactic dependencies and the words' syntax and semantic information, respectively. The sentence matching, therefore, becomes a process of first extracting the sub-graph structure alignmentsand then summarizing them into the final matching score.

A neural model called Interacted Syntax Graphs (ISG) is developed for conducting the matching. Specifically, given two sentences, ISG employs a pre-trained language model (PLM) and a syntactic parser to get the word embeddings and the syntactic structures as the initialization features. Then, ISG fuses these semantic and syntactic features into an associate graph. Sub-graph matching patterns can be extracted based on the associate graph, by using Lawler's quadratic assignment programming (QAP) (Cho et al., 2010). Finally, a matching classifier is used to merge the matching patterns and semantic vectors outputted from the PLM, resulting in the final matching score.

ISG offers several advantages, including accurate extraction and fusion of the syntactic graph matching signals, ease in interpretation, and high matching accuracy. The contributions of this paper can be summarized as follows: (1) We highlight the importance of the fine-grained matching patterns from syntax graphs in semantic sentence matching. A novel matching model called ISG is proposed; (2) Experimental results based on three available benchmarks showed that the matching accuracy of ISG outperformed the state-of-the-art baselines; (3) Analysis showed that ISG can discriminate important syntactic and semantic matching patterns in an interpretable way.

2 Related Work

Machine learning models have been widely used for matching natural language sentences (Li and Xu, 2014; Xu et al., 2020). Among them, the representative methods include DSSM (Huang et al., 2013) and its extensions (Wang et al., 2017a; Shen et al., 2014; Kim et al., 2019; Yang et al., 2019). Representative interaction-based models include ARC-II (Hu et al., 2014), MatchPyramid (Pang et al., 2016), etc. Recently, the pre-trained language model has been adapted to conducting matching (Devlin et al., 2019; Liu et al., 2019). These models always focused on superficial matching signals and ignore explicit NLP knowledge.

Recently, there is a trend to utilize explicit NLP knowledge to improve sentence representation. For example, HIM (Chen et al., 2017) used the syntactic dependencies to enhance the sentence representations, see also (Chen et al., 2016; Liu et al., 2018; Tymoshenko and Moschitti, 2018). The NLP knowledge-enhanced matching models have also adapted to the interaction-based models. For example, MIX (Chen et al., 2018) utilizes POS and named-entity tags as prior features. Recently, (Sachan et al., 2021; Bai et al., 2021; Zhang et al., 2020b) found that syntax can help PLM capture more information and achieve impressive results for NLP tasks. However, these models overlook the fine-grained syntactic matching patterns.

Graph matching problem (GM) has also been adopted for discovering the patterns between different graphs (Loiola et al., 2007). The key is to learn a practical affinity function with given two structures. In early work, most on seeking approximate affinity function and Euclid distance together with the Gaussian kernel is applied (Cho et al., 2010; Leordeanu et al., 2012). Recently, quadratic assignment programming (QAP) (Cho et al., 2010) has a wide application in Graph Matching (GM) because of its great performance. The affinity function in QAP can be learned with the manners of unsupervised (Leordeanu et al., 2012), semi-supervised (Leordeanu et al., 2011), or supervised (Loiola et al., 2007). Recently, deep graph matching has been applied for GM on images (Zanfir and Sminchisescu, 2018; Wang et al., 2021) and the matching accuracy has been achieved. Moreover, Graph-based models have also been used for sentence matching (Yao et al., 2019; Zhang et al., 2020a; Sachan et al., 2021).

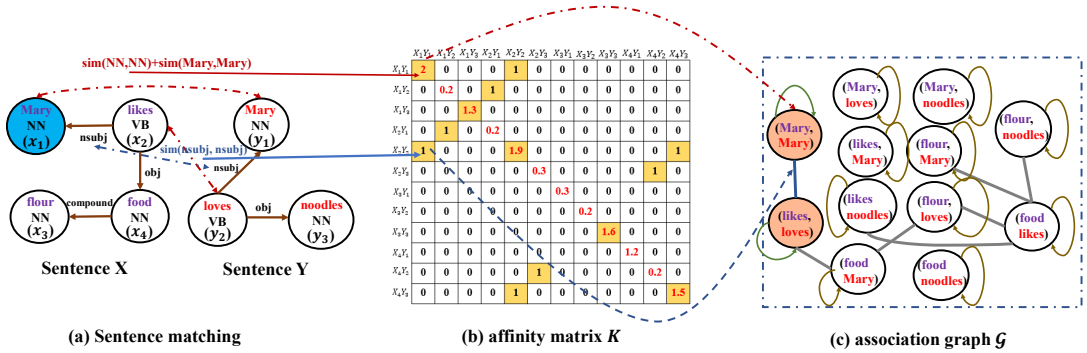

(a) Sentence matching (b) affinity matrix K (c) association graph $\mathcal{G}$

Figure 2: Example of converting the matching of a pair of sentences with syntactic structures (a) to the corresponding affinity matrix K (b) or association graph $\mathcal{G}$ (c).

3 Problem Formulation

3.1 Sentence matching

The matching of a pair of natural language sentences can be formally described as follows: suppose that $\mathcal{Z}$ is the set of labels which is defined by a specific matching task. In the paraphrase identification (PI) tasks, $\mathcal{Z} = \{0, 1\}$, where '0' and '1' respectively denote the relationship of "dissimilar" and "similar"; in natural language inference (NLI) $\mathcal{Z} = \{0, 1, 2\}$, where 0, 1, 2 respectively indicate "contradiction", "neutral", and "entailment". A set of training instances $\mathcal{D} = \{(X_i, Y_i, z_i)\}_{i=1}^{N}$ is given where each sample $(X, Y, z) \in \mathcal{D}$ consists of a sentence pair (X, Y) and its ground-truth matching label z. Moreover, the X, Y are two sequences of words: $X = \{x_1, x_2, \cdots, x_{t_X}\}$ and $Y = \{y_1, y_2, \cdots, y_{t_Y}\}$, where the x_i and y_j denote the i-th and j-th words in X and Y, t_X and t_Y are the number of words (lengths) of X and Y, respectively.

3.2 Quadratic assignment programming

Quadratic assignment programming (QAP) is a type of combinatorial optimization problems (Loiola et al., 2007), originally designed for the facilities-location problems. Suppose to assign N facilities to N locations, with the cost f_{ij}, d_{kl} of the affinities between the facilities (i, j) and locations (k, l), plus the costs $b_{i\phi(i)}$ of assigning a facility i to a certain location $\phi(i)$. The objective of QAP is assigning each facility to a location such that the total assignment cost is minimized. Lawler (1963) introduced a general form of QAP as in Equation (1):

$$\min_{\phi \in S} \sum_{i=1}^{N} \sum_{j=1}^{N} c_{ij\phi(i)\phi(j)} + \sum_{i=1}^{N} b_{i\phi(i)}, \qquad (1)$$

where S is the set of all permutations $\phi : N \to N$, $\forall i, j, k, l, c_{ijkl} := f_{ij}d_{kl}$ if $\wedge i \neq j, k \neq l$, otherwise $c_{iikk} := f_{ii}d_{kk} + b_{ik}$. The formulation has been widely applied to graph matching, which involves establishing node correspondences between two graphs based on the linear and quadratic structure affinity (Leordeanu and Hebert, 2005).

3.3 Sentence matching over syntax graphs

This paper proposes to adopt QAP for conducting sentence matching, by regarding the words in one sentence as the "facilities" and words in another sentence as the "locations", and their differences in syntactic structures and semantics as the "assignment costs". In this way, QAP enables the matching model to involve not only the linear syntactic structure (e.g. word attribute structure) costs which correspond to assigning the "facilities" to certain "locations", but also the quadratic syntactic structures (e.g. word-word relation structure) costs which correspond the affinities between the assigning "facilities" and "locations".

Figure 2 gives an illustrative example of formulating the sentence matching as graph matching and finding matching patterns in the affinity graph. with a sentence pair X ="*Mary likes flour food*" (length $|X| = 4$) and Y ="*Mary loves noodles*" (length $|Y| = 3$), using the parsed POS tags and syntactic dependencies shown in Figure 2(a) which are represented as the node and edge weights, respectively.

As shown in Figure 2(a), we firstly constructed two directed graph (G_X, G_Y) based on the two sentences (X, Y)'s syntax. Specifically, the word embedding and word syntax (i.e., POS) can be encoded as node features, and the word-word syntax (i.e., dependencies) can be encoded as edge features. The fine-grained syntax-based matching signals can be viewed as the alignments between

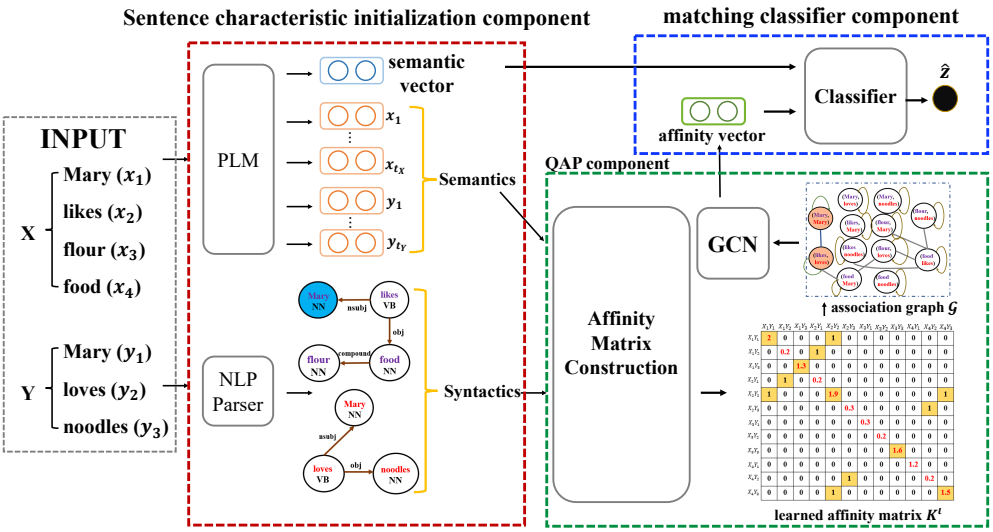

Figure 3: Architecture of Neural Quadratic Assignment Programming for Sentence Matching.

sub-graphs of two sentence graphs (G_X, G_Y).

At the same time, to capture syntax-based patterns effectively, we further construct the affinity matrix K or the association graph $\mathcal{G}$ based on the sentence graphs (G_X, G_Y). The weights of nodes and edges in the association graph are corresponding to the diagonal and off-diagonal elements in the affinity matrix (Figure 2(b,c)), respectively.

Specifically, the weights of nodes describe the word semantic similarities and POS (word attribute) affinities, and the weights of edges describe the syntactic dependency (word-word relation) affinities. For example, the node x_1y_1 =("Mary","Mary") could have the weight of, for example, $1.0 + 1.0 = 2.0$ where the first 1.0 denoting the semantic similarity, and the second 1.0 denoting the similarity between the POS tags.

As an example for the edges corresponding to the off-diagonal elements in affinity matrix K, there could be an edge with weight, for example, 1.0 between node x_1y_1 =("Mary","Mary") and node x_2y_2 =("likes","loves") because the dependency relation between x_1 ="Mary" and x_2 ="likes" is "nsubj", while the dependency relation between y_1 ="Mary" and y_2 ="loves" is also "nsubj". Similarly, the other edges can also be created.

Then the QAP is applied to learn the semantic and syntactic matching patterns in affinity matrix K. Formally, a relaxed form of QAP can be shown as Equation (2):

$$\max_{S} \ \mathrm{vec}(\mathbf{S})^T \mathbf{K} \mathrm{vec}(\mathbf{S}), \tag{2}$$

where $\mathbf{S}$ matrix encodes the word-word correspondence; $\mathrm{vec}(\mathbf{S})$ is $\mathbf{S}$'s column-vectorized notation, and $\mathbf{K} \in \mathbb{R}^{t_X t_Y \times t_X t_Y}$. The $\mathbf{S}$ matrix can be regarded as an aggregated matching patterns of syntax and semantics.

4 Proposed model: ISG

In this section, we present an efficient implementation of interacting syntax graphs (ISG) with quadratic assignment programming (QAP) (Lawler, 1963; Cho et al., 2010). Figure 3 illustrates the model architecture, which can be divided into sentence characteristic initialization, QAP component, and matching classifier.

4.1 Sentence characteristic initialization

In this component, the inputted natural language sentence pair (X, Y) is processed with a pre-trained language model (PLM) and an NLP parser, generating the semantic features and syntactic structures.

Semantic features Given a pair (X, Y), the semantic matching vector (e.g., "[CLS]" vector of BERT), $\mathbf{v}_s \in \mathbb{R}^d$ and the words embeddings $\mathbf{F}^X \in \mathbb{R}^{t_X \times d}, \mathbf{F}^Y \in \mathbb{R}^{t_Y \times d}$ consists of the semantic features:

$$(\mathbf{v}_s, \mathbf{F}^X, \mathbf{F}^Y) = \mathrm{PLM}(X, Y; \theta_p),$$

where d denotes the size of the feature vector, PLM could be BERT or other PLM models, and θ_p denotes the parameters of PLM.

Syntactic structures Generally speaking, there are two types of structures: the word attribute structure (WAS) which reflects the attributes of the word, and the word-word relation structure (WRS) which defines the relationship between two words.

The WAS attributes can be further categorized

and this paper only considers POS attributes. Given any sentence $X = \{x_1, \cdots, x_{t_X}\}$, the sequence of WAS attributes could be $\left\{a_{x_1}, a_{x_2}, \cdots, a_{x_{t_X}}\right\}$.

The WRS attributes can also be further categorized and this paper considers syntactic dependency. Given any sentence $X = \{x_1, \cdots, x_{t_X}\}$, the WRS parsing results (a dependency parsing graph) can be represented as two incidence matrices:

$$(\mathbf{I}^X, \mathbf{H}^X) = \text{Parse}(X),$$

where $\mathbf{I}^X \in \mathbb{R}^{t_X \times e_X}$ records the output-links and $\mathbf{H}^X \in \mathbb{R}^{t_X \times e_X}$ records the in-links, e_X denotes the edge number of WRS. The elements of these two matrices are defined as: if k-th edge links from word x_i to x_j (its type also denoted as $e_k(x_i, x_j)$), $\mathbf{I}^X(i,k) = \mathbf{H}^X(j,k) = 1$, and note that in order to reduce the noise from the dependencies, we also set $\mathbf{I}^X(j,k) = \mathbf{H}^X(i,k) = 1$. Otherwise, $\mathbf{I}^X(i,k) = \mathbf{H}^X(j,k) = \mathbf{I}^X(j,k) = \mathbf{H}^X(i,k) = 0$.

In this paper, we used the Stanford CoreNLP parser (Manning et al., 2014) for getting POS, and syntactic dependencies. Note that other syntactic structures can be also used, such as named-entity and semantic dependencies (Wang et al., 2019b).

4.2 QAP component

Based on the word embeddings and parsed syntactic structures, the QAP component first constructs an association graph (affinity matrix) and then solves the QAP problem, achieving the permutation which represents the word matching between the two sentences.

4.2.1 Learned affinity matrix construction

Following the practices in (Zhou and De la Torre, 2015), the QAP sparse affinity matrix $\mathbf{K}^l \in \mathbb{R}^{t_X t_Y \times t_X t_Y}$, referred to as the learned affinity matrix, can be factorized as

$$\mathbf{K}^l = \text{diag}(\text{vec}(\mathbf{P})) + (\mathbf{I}^X \otimes_{\mathcal{K}} \mathbf{I}^Y)\text{diag}(\text{vec}(\mathbf{R}))(\mathbf{H}^X \otimes_{\mathcal{K}} \mathbf{H}^Y)^T, \tag{3}$$

where operator diag$(\cdot)$ builds a diagonal matrix from input vector, $\mathbf{I}^X, \mathbf{H}^X, \mathbf{I}^Y, \mathbf{H}^Y$ are sentences X and Y's parsing results, as described in Section 4.1, $\otimes_{\mathcal{K}}$ denotes Kronecker product, and $\mathbf{P}$ and $\mathbf{R}$ encode the WAS, word embedding similarity and WRS similarity matrix, respectively and they are defined as:

$$\mathbf{P} = (1-\alpha)\mathbf{U}^X \mathbf{\Lambda_u} \mathbf{U}^{Y^T} + \alpha \mathbf{F}^X \mathbf{\Lambda_f} \mathbf{F}^{Y^T}, \mathbf{R} = \mathbf{L}^X \mathbf{\Lambda_r} \mathbf{L}^{Y^T},$$

where $\mathbf{\Lambda_u}$,$\mathbf{\Lambda_f}$,$\mathbf{\Lambda_r}$ are learn-able parameters for affinity metric, α is the trade-off coefficient for POS affinities and word-word similarities, and $\mathbf{U}^X \in \mathbb{R}^{t_X \times d}, \mathbf{U}^Y \in \mathbb{R}^{t_Y \times d}$ are the WAS sequence embeddings of X, Y and the edge representations $\mathbf{L}^X \in \mathbb{R}^{e_X \times d}, \mathbf{L}^Y \in \mathbb{R}^{e_Y \times d}$ are built by its edge sequence embeddings. Note that all the aforementioned operations for constructing $\mathbf{K}^l$ allow back propagation, and we adopt the GPU implementation provided by (Wang et al., 2019a). A more detailed QAP factorization is given in Appendix A.

4.2.2 Solving the permutation vector

Due to the high compute cost for solving the permutation vector through the learned affinity matrix K^l, we adopt the GCN method implemented by Wang et al. (2019a) to approximate the QAP problem into a linear assignment programming(LAP) problem, which can be solved in an efficient way for both time and space.

Specifically, we build the association graph $\mathcal{G} = \{\mathbf{v}^{(0)}, \mathbf{A}\}$ with its initial node embedding $\mathbf{v}^{(0)}$ and its sparse adjacent matrix $\mathbf{A}$ from the learned affinity matrix K^l. Then we can apply GCN method to updated the node embedding for k-th GCN layer, $k = 1, 2, \cdots, G_k$. The key idea is to encode the quadratic structure (WRS) to the linear structure (WAS). The permutation matrix S can be regarded as the last layer of the node features:

$$\text{vec}(S) = \mathbf{v}^{(G_k)},\ \mathbf{v}^{(k+1)} = \mathbf{A}\mathbf{W} f(\mathbf{v}^{(k)}; \theta_k)) + \mathbf{v}^{(k)}, \tag{4}$$

where the $f(\cdot)$ is a MLP projection function at the k-th layer is parameterized by θ_k and the k-th layer node embedding of association graph denotes as: $\mathbf{v}^{(k)} \in \mathbb{R}^{t_X t_Y \times \ell^k}$, with the initial embeddings $\mathbf{v}^{(0)} \in \mathbb{R}^{t_X t_Y \times 1}$ taken from the diagonal elements of K^l. The GCN projection matrix $\mathbf{W} \in \mathbb{R}^{t_X t_Y \times t_X t_Y}$ comes from the off-diagonal elements.

$$\mathbf{v}^{(0)}(i,a) = \mathbf{K}^l(ia, ia),\ \mathbf{W}(ia, jb) = \mathbf{K}^l(ia, jb),$$

for all $i, j \in t_X, a, b \in t_Y$.

As for the adjacent matrix $\mathbf{A}$, in order to control its sparsity, we introduce a hyper-parameter γ to generate the sparse adjacent matrix of association graph $\mathcal{G}$ from the projection matrix $\mathbf{W}$:

$$A(ia, jb) = \begin{cases} 1 & \text{if } W(ia, jb) \geq \gamma, \\ 0 & \text{otherwise.} \end{cases}$$

4.3 Matching classifier

Given the semantic matching feature $\mathbf{v}_s$ and QAP permutation vector $\mathbf{v}^{(G_k)}$, the matching score $\hat{z}$ can

be obtained by the MLP parameterized by θ_m:

$$\hat{\mathbf{z}}(X,Y) = \text{MLP}([\mathbf{v}_s|\mathbf{v}^{(G_k)}];\theta_m). \quad (5)$$

where '|' denotes the concatenation operation, $\hat{\mathbf{z}}(X,Y) = \left[\hat{z}_1, \cdots, \hat{z}_{|\mathcal{Z}|}\right]$ and $\hat{z}_k$ denotes the probability of k-th category. The last layer is softmax so that the output is a probability distribution.

4.4 Learning the model parameters

ISG has parameters to determine, including $\mathbf{\Theta} = \{\theta_p, \theta_k, \mathbf{\Lambda_u}, \mathbf{\Lambda_f}, \mathbf{\Lambda_r}, \theta_m \mid k = 1, 2, \cdots, G_k\}$. In the training phase, given a set of sentence pairs with ground truth labels $\mathcal{D} = \{(X_i, Y_i, z_i)\}_{i=1}^N$, the learning algorithm aims to minimize the matching loss $\mathcal{L}_m$ which measures the differences between the prediction $\hat{z}$ and ground-truth z, regularized by the affinity regularizer $\mathcal{R}$ which forces the learned affinity matrix K^l and the original parsed affinity matrix K the being similar. Formally, the loss $\mathcal{L}$ that being minimized is:

$$\mathcal{L} = \mathcal{L}_m(\hat{z}, z) + \lambda_a \mathcal{R}(K, K^l) + \mu_r \|\theta\|^2, \quad (6)$$

where $\|\theta\|^2$ is the ℓ_2 regularizer, λ_a, μ_r denote the trade-off coefficient of affinity regularizer and ℓ_2 regularizer.

Matching loss The matching loss $\mathcal{L}_m$ is learned by minimizing the cross-entropy loss between the labels and the predicted results:

$$\mathcal{L}_m = -\sum_{(X,Y,\mathbf{z})\in\mathcal{D}} \sum_{k=1}^{|\mathcal{Z}|} z_k \log \hat{z}_k, \quad (7)$$

Affinity regularizer The affinity regularizer $\mathcal{R}$ aims to force the structure affinities respectively correspond to the parsed syntactic structure and that of learned from neural network to be similar. Thus the $\mathcal{R}$ is learned to minimize the KL-divergence between the learned affinity matrix K^l and parsed affinity matrix K:

$$\mathcal{R} = \sum_{(X,Y)\in\mathcal{D}} KL(K||K^l), \quad (8)$$

where the parsed affinity matrix K is defined as follows: the diagonal elements $K(ia, ia)$ will be 1 if the matched words have identical attribute, otherwise 0. And the off-diagonal element $K(ia, jb)$ will be 1 if the word pair (x_i, x_j) and (y_a, y_b) have identical word-word relation, otherwise 0.

4.5 Time complexity of online matching

At the online time, ISG needs to process the sentence pairs with PLM, parse them with the NLP parser, solve the QAP and finally calculate the matching score. The online time complexity for typical PLM (Devlin et al., 2019; Liu et al., 2019) and NLP parser (Manning et al., 2014; Wang et al., 2019b) is of $\mathcal{O}(|t_X + t_Y|^2 \times d)$ and $\mathcal{O}((|t_X|^2 + |t_Y|^2) \times d)$, where d is the embedding dimension of each word.

At the online matching, the time complexity of the relaxed QAP is related to GCN, which is of $\mathcal{O}(G_k m\ell + G_k n\ell^2)$ (**?**) on the association graph, where $n = t_X t_Y$ is the total number of nodes, m is the total number of edges, G_k is the number of layers, and ℓ is the dimension of the node hidden features. Note that the hyper-parameter γ controls the sparsity of the edges (as mentioned in Section 4.2), we can adjust γ so that $m \ll t_X t_Y$ and therefore reduce the time complexity of the relaxed QAP to $\mathcal{O}(G_k t_X t_Y \ell^2)$, which is more efficient than the original QAP (Wang et al., 2019a). Therefore, the total time complexity of ISG is $\mathcal{O}(|t_X + t_Y|^2 \times d + G_k t_X t_Y \ell^2)$, which is comparable with the underlying PLM.

5 Experiments

We conducted experiments to verify the effectiveness of the proposed approach. The source code and experiments are available at the link.[1]

5.1 Experimental Settings

The experiments were conducted on three large scale publicly available benchmarks:

Quora Question Pairs (QQP):[2] a large public dataset for paraphrase identification. QQP contains 404k labeled sentence pairs. We used the same data split as in (Wang et al., 2017b). **SNLI:**[3] a well-known dataset for natural language inference (NLI). SNLI contains 570k labeled sentence pairs. Following the practices in (Bowman et al., 2015), we used the same data split way. **SciTail:**[4] another NLI dataset based on science exams and web. Its label only contains two classes: "entailment" or "neutral". The dataset contains 27k sentence pairs.

[1] https://github.com/XuChen0427/Semantic-Sentence-Matching-via-Interacting-Syntax-Graphs

[2] https://www.kaggle.com/c/quora-question-pairs

[3] https://nlp.stanford.edu/projects/snli

[4] http://data.allenai.org/scitail/

Several state-of-the-art baselines which conducts the matching without utilizing syntactic structures were chosen as the baselines, including DIIN (Gong et al., 2018), MwAN (Tan et al., 2018),BIMPM (Wang et al., 2017a), CSRAN (Kim et al., 2019), DecAtt (Parikh et al., 2016), CAFE (Tay et al., 2018), and DGEM (Khot et al., 2018), RE2 (Yang et al., 2019), and the BERT (Devlin et al., 2019), RoBERTa (Liu et al., 2019). Some models are task-adopted (e.g. DGEM is for NLI task), thus they are missing on some datasets. ISG was compared with the baselines DDR-match (Yu et al., 2020, 2022) that applied unsupervised assignment problems to conduct sentence matching. ISG was also compared with the baselines that utilize syntactic structures like HIM (Chen et al., 2017), which uses the constituency tree to improve local word representation, and Sembert (Zhang et al., 2020b), Syntax-bert (Bai et al., 2021) that applied different syntax to improve PLM performance. To make a fair comparison, we cannot reproduce their results due to the source codes are unavailable public of graph-based matching models (). Therefore, we decided to implement a representation-based GCN method, by following the representation-model architecture (lines 455-461, 497-505) and using the same parsing results. The method is denoted as ISG (representation-GCN) in Table 1.

To get the syntactic structures of the inputted sentences, the Stanford-corenlp (Manning et al., 2014) was used to parse the syntactic structures. In all of the experiments, the maximum sentence length was set to 70 and the sentences with lengths less than 3 were removed for reducing the noise. In the training process, all of the models were trained with the learning rate tuned amongst $[1e-5, 5e-5]$. The batch size was tuned amongst $[8, 16, 32]$, and the graph network layer G_k was tuned amongst $[1, 3]$, the coefficient $\alpha = 0.8$ and the sparsity threshold tuned amongst $[0, 0.3]$. The trade-off coefficient of affinity regularizer λ_a's were tuned amongst $[4e-3, 1e-2]$.

5.2 Experimental results

Table 1 reports the matching accuracy of the proposed ISG and the baselines on the three datasets. The '-' means the number is not available. The accuracy of baselines is according to the numbers reported. For our methods, the averaged numbers over 5 runs are reported, with the standard deviations in parentheses. From the results, we can see that different versions of the proposed ISG outperformed all of the baselines. The results also indicated that though PLM (e.g. BERT,RoBERTa) achieved SOTA accuracy, ISG can still get improvements by incorporating the syntactic information.

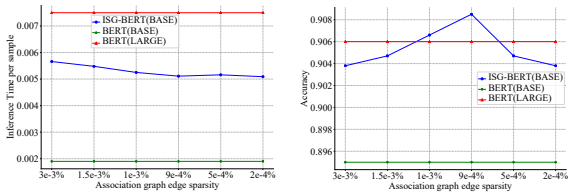

(a) Inference time comparsion (b) Accuracy comparison

Figure 4: ISG-BERT$_{BASE}$'s inference time (figure (a)) and matching accuracy (figure (b)) curves w.r.t. the sparsity of the association graph. Experiments were conducted on SciTail.

We also note that ISG outperformed the baselines that utilize the syntactic structures for matching. Comparing ISG with these models, we found that these baseline models all encode the syntactic structures as sentence features to enrich their representations, while ISG incorporates the syntactic and semantic matching patterns through a graph matching task and aggregates them through the affinity matrix. Moreover, we also compared the graph matching methods that separately encode two graphs into two embeddings using GCN methods and conducted matching scores based on the learned embeddings. We found that ISG will still outperform the representation-based graph matching methods. The results demonstrated that the ISG is more effective to utilize syntactic matching signals.

We also investigated the online time complexity of ISG. Figure 4 reports the impacts of association graph sparsity on ISG-BERT$_{BASE}$ on the Scitail test-set, where the sparsity (calculated as the fraction of edge number and square of node number in association graph) is from $[2e-4\%, 3e-3\%]$. The sparsity was adjusted by changing the hyper-parameters γ.

Figure 4(a) illustrates that the inference time of ISG will decrease with the increase of the association graph sparsity. Moreover, the inference time of ISG-BERT$_{BASE}$ is about 2.5 times that of the underlying PLM, and about 0.7 times that of BERT$_{LARGE}$. The results verified the time complexity analysis conclusion in Section 4.5.

Figure 4(b) shows the accuracy curves of ISG,

Table 1: Performance comparisons on Quora Question Pairs, SNLI and SciTail. The ±numbers in brackets mean 1-std deviations. The $*$ denotes the models are our implementations and are trained among same settings.

Models without syntactic structures	QQP:Acc(%)	SNLI:Acc(%)	SciTail:Acc(%)
DGEM (Khot et al., 2018)	-	-	77.3
DecAtt (Parikh et al., 2016)	-	82.5	81.7
CAFE (Tay et al., 2018)	-	88.5	83.3
BIMPM (Wang et al., 2017a)	88.7	88.8	85.4
DIIN (Gong et al., 2018)	89.1	-	-
MwAN (Tan et al., 2018)	89.1	-	-
CSRAN (Kim et al., 2019)	89.2	88.7	86.7
RE2 (Yang et al., 2019)	89.2	89.0	86.6
DDR-Match(BERT)* (Yu et al., 2022)	89.6	89.2	90.3
BERT$^*_{BASE}$ (Devlin et al., 2019)	89.4	89.0	89.5
BERT$^*_{LARGE}$ (Devlin et al., 2019)	89.6	89.2	90.6
RoBERTa$^*_{LARGE}$ (Liu et al., 2019)	90.0	90.1	91.5
Models with syntactic structures	QQP:Acc(%)	SNLI:Acc(%)	SciTail:Acc(%)
HIM (Chen et al., 2017)	88.7	88.6	71.6
SemBERT$^*_{BASE}$ (Zhang et al., 2020b)	89.8	90.2	92.1
SemBERT$^*_{LARGE}$ (Zhang et al., 2020b)	90.7	91.0	92.1
SyntaxBERT$_{BASE}$ (Bai et al., 2021)	89.6	87.8	-
SyntaxBERT$_{LARGE}$ (Bai et al., 2021)	89.5	89.0	-
ISG(representation-GCN)-BERT$^*_{BASE}$	90.1	89.7	90.6
ISG(representation-GCN)-BERT$^*_{LARGE}$	90.2	89.9	91.7
ISG(representation-GCN)-RoBERTa$^*_{LARGE}$	90.8	90.4	92.3
Ours(ISG-BERT$^*_{BASE}$)	**90.5** (±0.14)	**90.0** (±0.16)	**90.8** (±0.26)
Ours(ISG-BERT$^*_{LARGE}$)	**90.8** (±0.08)	**90.4** (±0.03)	**91.9** (±0.24)
Ours(ISG-RoBERTa$^*_{LARGE}$)	**91.4** (±0.1)	**91.2** (±0.08)	**93.3** (±0.2)

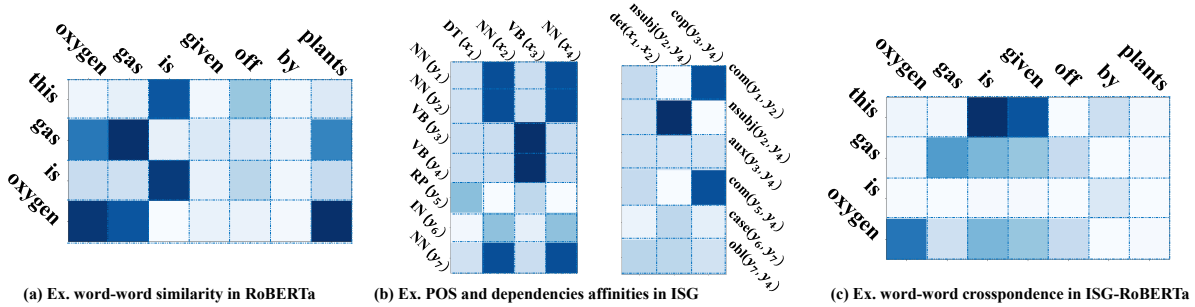

(a) Ex. word-word similarity in RoBERTa (b) Ex. POS and dependencies affinities in ISG (c) Ex. word-word crosspondence in ISG-RoBERTa

Figure 5: Cross sentence word-word similarity matrix and syntactic affinity matrices for two pairs : example("this gas is oxygen","oxygen gas is given off by plants"), which is from Scitail training set. Darker colors means higher similarities or affinities values.

which first increases in $[9e-4\%, 3e-3\%]$ and then dropped. We conclude that even association graph became sparse, ISG still constantly outperformed BERT$_{BASE}$ and outperformed BERT$_{LARGE}$ at some point. The results demonstrated that the QAP is efficient and will not delay the online time.

5.3 Empirical Analysis

5.3.1 Ablation Study

Firstly, we respectively set the WAS(POS) features $\mathbf{U}^X, \mathbf{U}^Y$, WRS(syntactic dependencies) features $\mathbf{L}^X, \mathbf{L}^Y$ and semantic features $\mathbf{v}_s, \mathbf{F}^X, \mathbf{F}^Y$ to zero vectors, to investigate their effects. Table 2 reports the accuracy of the ISG variation on the SciTail test data under BERT$_{BASE}$, where each variation is denoted as, for example, "ISG-w/o WRS" which means the WRS features were set zeros. Similar phenomenons have also been observed on the other two datasets, with other PLMs.

Compared ISG-BERT$_{BASE}$ with its variations, we can see that the matching performances dropped with large margins if the semantic features were set as zeros, indicating that only considering the syntax patterns did not work well. We also ob-

Table 2: Ablation study on SciTail test set.

Ablation Study Model	Acc(%)
BERT$_{BASE}$ (Devlin et al., 2019)	89.5 ($\pm$0.28)
ISG-w/o semantic and WRS	68.1 ($\pm$0.29)
ISG-w/o semantic and WAS	67.8 ($\pm$0.28)
ISG-w/o semantic	68.5 ($\pm$0.26)
ISG-w/o WAS	90.2 ($\pm$0.27)
ISG-w/o WRS	90.4 ($\pm$0.28)
ISG	**90.8** ($\pm$0.26)

Table 3: Ablation study for different syntactic structure on SciTail test set.

Ablation Study Model	Acc(%)
BERT$_{BASE}$ (Devlin et al., 2019)	89.5 ($\pm$0.28)
ISG NER&Syntactic dependencies	90.4 ($\pm$0.25)
ISG NER&Semantic dependencies	90.2 ($\pm$0.17)
ISG POS&Syntactic dependencies	**90.8** ($\pm$0.26)
ISG POS&Semantic dependencies	90.6 ($\pm$0.21)

served that the performances dropped when the WAS and WRS features were set to zeros. The bad performances were caused by removing the WAS and WRS features, indicating that syntax matching patterns are effective for sentence matching.

Moreover, We conduct the experiments with different WAS and WRS. Specifically, we respectively utilize the POS and named-entity(NE) as WAS and respectively utilize the syntactic dependencies and semantic dependencies as WRS. For semantic dependencies parsing, we follows Wang et al. (2019b). Table 3 also reports the accuracy of the ISG-BERT$_{BASE}$ variation on the SciTail test data.

Compared to the ISG variations, we can see that the matching accuracy is different for different WAS and WRS. The best and worst performance are caused by POS&syntactic dependencies and NER&Semantic dependencies, respectively. However, we can observe that all of these variations outperform the BERT baseline, which indicates the effectiveness of ISG in different WAS and WRS.

An experiment on the robustness of ISG's parameters can be found in Appendix B.

5.3.2 Matching Visualization of ISG

We conducted experiments to investigate how the ISG matched two sentences, using a real example from Scitail. The experiment was conducted based on the results of ISG-RoBERTa.

Figure 5(a) illustrated the word-word similarity matrix of these two sentences, based on the word embeddings outputted by RoBERTa, where the darker colors denote the higher similarities. Figure 5(b) illustrated the affinities between POS and dependencies in two sentences. Based on the similarities and affinity matrices, ISG solved the QAP and achieved a new correspondence matrix in Figure 5(c). The POS, word semantic similarities, and dependencies affinities correspond to the node weights and edge weights in the association graph.

The example illustrates the example that is from the Sctail training set: (X = *"this gas is oxygen"*, Y = *"oxygen gas is given off by plants"*) whose ground truth label is "neutral". The example illustrates how the ISG conducts sentence matching. Comparing word-word similarities by RoBERTa (Figure 5(a)) and that of ISG (Figure 5(c)), we can see that RoBERTa's results show high similarities between words in two sentences. On other hand, ISG can find fine-grained syntactic patterns, for example, the matching patterns of POS "VB" and "NN" and dependency "nsubj". And ISG will output the "dissimilar" result due to its low syntactic matching pattern scores.

The analysis clearly showed that ISG can well learn different syntax-based matching signals, and make them into good interactions. The results also showed what syntactic matching patterns are important and how two sentences were matched with the association graph.

6 Conclusion

This presents a novel sentence matching model which formulates sentence matching as a problem of syntax graph matching, referred to as ISG. Based on the constructed association graph, ISG explicitly aligns the syntactic sub-graphs as fine-grained matching signals. Neural QAP is adopted to learn the rich matching patterns from the training data. ISG offers several advantages: explicitly interacting with the syntactic structures in fine granularity, high matching accuracy, and the ability to interpret. Experiments on three publicly available benchmarks verified the effectiveness, robustness, and interpretability of ISG.

Acknowledgments

This work was funded by the National Key R&D Program of China (2019YFE0198200), National Natural Science Foundation of China (61872338, 61832017), and Beijing Outstanding Young Scientist Program NO. BJJWZYJH012019100020098.

References

Jiangang Bai, Yujing Wang, Yiren Chen, Yaming Yang, Jing Bai, Jing Yu, and Yunhai Tong. 2021. Syntax-bert: Improving pre-trained transformers with syntax trees. *arXiv preprint arXiv:2103.04350*.

Samuel R. Bowman, Gabor Angeli, Christopher Potts, and Christopher D. Manning. 2015. A large annotated corpus for learning natural language inference. In *Proceedings of the 2015 Conference on EMNLP*, pages 632–642.

Haolan Chen, Fred X. Han, Di Niu, Dong Liu, Kunfeng Lai, Chenglin Wu, and Yu Xu. 2018. MIX: multi-channel information crossing for text matching. In *Proceedings of the 24th International Conference on KDD*, pages 110–119.

Qian Chen, Xiaodan Zhu, Zhen-Hua Ling, Si Wei, Hui Jiang, and Diana Inkpen. 2017. Enhanced LSTM for natural language inference. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics*, pages 1657–1668.

Qian Chen, Xiaodan Zhu, Zhenhua Ling, Si Wei, and Hui Jiang. 2016. Enhancing and combining sequential and tree lstm for natural language inference. *arXiv preprint arXiv:1609.06038*.

Minsu Cho, Jungmin Lee, and Kyoung Mu Lee. 2010. Reweighted random walks for graph matching. In *European conference on Computer vision*, pages 492–505. Springer.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 4171–4186.

Yichen Gong, Heng Luo, and Jian Zhang. 2018. Natural language inference over interaction space. In *6th International Conference on Learning Representations, ICLR, Conference Track Proceedings*.

Baotian Hu, Zhengdong Lu, Hang Li, and Qingcai Chen. 2014. Convolutional neural network architectures for matching natural language sentences. In *Advances in Neural Information Processing Systems 27*, pages 2042–2050.

Po-Sen Huang, Xiaodong He, Jianfeng Gao, Li Deng, Alex Acero, and Larry P. Heck. 2013. Learning deep structured semantic models for web search using clickthrough data. In *22nd ACM International Conference on Information and Knowledge Management, CIKM'13*, pages 2333–2338.

Tushar Khot, Ashish Sabharwal, and Peter Clark. 2018. Scitail: A textual entailment dataset from science question answering. In *Proceedings of the Thirty-Second AAAI Conference on Artificial Intelligence*, pages 5189–5197.

Seonhoon Kim, Inho Kang, and Nojun Kwak. 2019. Semantic sentence matching with densely-connected recurrent and co-attentive information. In *The Thirty-Third AAAI Conference on Artificial Intelligence*, pages 6586–6593.

Eugene L Lawler. 1963. The quadratic assignment problem. *Management science*, pages 586–599.

Marius Leordeanu and Martial Hebert. 2005. A spectral technique for correspondence problems using pairwise constraints.

Marius Leordeanu, Rahul Sukthankar, and Martial Hebert. 2012. Unsupervised learning for graph matching. *International journal of computer vision*, pages 28–45.

Marius Leordeanu, Andrei Zanfir, and Cristian Sminchisescu. 2011. Semi-supervised learning and optimization for hypergraph matching. In *2011 International Conference on Computer Vision*, pages 2274–2281.

Hang Li and Jun Xu. 2014. Semantic matching in search. *Foundations and Trends in Information Retrieval*, pages 343–469.

Tao Liu, Xin Wang, Chengguo Lv, Ranran Zhen, and Guohong Fu. 2020. Sentence matching with syntax- and semantics-aware bert. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 3302–3312.

Yang Liu, Matt Gardner, and Mirella Lapata. 2018. Structured alignment networks for matching sentences. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 1554–1564.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Roberta: A robustly optimized BERT pretraining approach. *CoRR*, abs/1907.11692.

Eliane Maria Loiola, Nair Maria Maia de Abreu, Paulo Oswaldo Boaventura-Netto, Peter Hahn, and Tania Querido. 2007. A survey for the quadratic assignment problem. *European journal of operational research*, pages 657–690.

Christopher Manning, Mihai Surdeanu, John Bauer, Jenny Finkel, Steven Bethard, and David McClosky. 2014. The Stanford CoreNLP natural language processing toolkit. In *Proceedings of 52nd Annual Meeting of the Association for Computational Linguistics: System Demonstrations*, pages 55–60.

Lili Mou, Rui Men, Ge Li, Yan Xu, Lu Zhang, Rui Yan, and Zhi Jin. 2016. Natural language inference by tree-based convolution and heuristic matching. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics*, pages 130–136.

Liang Pang, Yanyan Lan, Jiafeng Guo, Jun Xu, Shengxian Wan, and Xueqi Cheng. 2016. Text matching as image recognition. In *Proceedings of the Thirtieth AAAI Conference on Artificial Intelligence*, pages 2793–2799.

Ankur Parikh, Oscar Täckström, Dipanjan Das, and Jakob Uszkoreit. 2016. A decomposable attention model for natural language inference. In *Proceedings of the 2016 Conference on EMNLP*, pages 2249–2255.

Devendra Sachan, Yuhao Zhang, Peng Qi, and William L Hamilton. 2021. Do syntax trees help pre-trained transformers extract information? In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*, pages 2647–2661.

Yelong Shen, Xiaodong He, Jianfeng Gao, Li Deng, and Grégoire Mesnil. 2014. Learning semantic representations using convolutional neural networks for web search. In *Proceedings of the 23rd international conference on WWW*, pages 373–374.

Chuanqi Tan, Furu Wei, Wenhui Wang, Weifeng Lv, and Ming Zhou. 2018. Multiway attention networks for modeling sentence pairs. In *Proceedings of the 27th International Joint Conference on Artificial Intelligence*, pages 4411–4417.

Yi Tay, Anh Tuan Luu, and Siu Cheung Hui. 2018. Compare, compress and propagate: Enhancing neural architectures with alignment factorization for natural language inference. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 1565–1575.

Kateryna Tymoshenko and Alessandro Moschitti. 2018. Cross-pair text representations for answer sentence selection. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 2162–2173, Brussels, Belgium. Association for Computational Linguistics.

Runzhong Wang, Junchi Yan, and Xiaokang Yang. 2019a. Learning combinatorial embedding networks for deep graph matching. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 3056–3065.

Runzhong Wang, Junchi Yan, and Xiaokang Yang. 2021. Neural graph matching network: Learning lawler's quadratic assignment problem with extension to hypergraph and multiple-graph matching. *IEEE Transactions on Pattern Analysis and Machine Intelligence*.

Xinyu Wang, Jingxian Huang, and Kewei Tu. 2019b. Second-order semantic dependency parsing with end-to-end neural networks. In *Proceedings of the 57th Annual Meeting of the ACL*, pages 4609–4618.

Zhiguo Wang, Wael Hamza, and Radu Florian. 2017a. Bilateral multi-perspective matching for natural language sentences. In *Proceedings of the Twenty-Sixth International Joint Conference on Artificial Intelligence*, pages 4144–4150.

Zhiguo Wang, Wael Hamza, and Radu Florian. 2017b. Bilateral multi-perspective matching for natural language sentences. In *Proceedings of the Twenty-Sixth International Joint Conference on Artificial Intelligence, IJCAI 2017*, pages 4144–4150.

Jun Xu, Xiangnan He, and Hang Li. 2020. Deep learning for matching in search and recommendation. *Foundations and Trends in Information Retrieval*, pages 102–288.

Runqi Yang, Jianhai Zhang, Xing Gao, Feng Ji, and Haiqing Chen. 2019. Simple and effective text matching with richer alignment features. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 4699–4709.

Liang Yao, Chengsheng Mao, and Yuan Luo. 2019. Graph convolutional networks for text classification. In *Proceedings of the AAAI Conference on Artificial Intelligence*, pages 7370–7377.

Weijie Yu, Chen Xu, Jun Xu, Liang Pang, Xiaopeng Gao, Xiaozhao Wang, and Ji-Rong Wen. 2020. Wasserstein distance regularized sequence representation for text matching in asymmetrical domains. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 2985–2994.

Weijie Yu, Chen Xu, Jun Xu, Liang Pang, and Ji-Rong Wen. 2022. Distribution distance regularized sequence representation for text matching in asymmetrical domains. *IEEE/ACM Trans. Audio, Speech and Lang. Proc.*, 30:721–733.

Andrei Zanfir and Cristian Sminchisescu. 2018. Deep learning of graph matching. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2684–2693.

Bo Zhang, Yue Zhang, Rui Wang, Zhenghua Li, and Min Zhang. 2020a. Syntax-aware opinion role labeling with dependency graph convolutional networks. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 3249–3258.

Zhuosheng Zhang, Yuwei Wu, Hai Zhao, Zuchao Li, Shuailiang Zhang, Xi Zhou, and Xiang Zhou. 2020b. Semantics-aware bert for language understanding. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pages 9628–9635.

Feng Zhou and Fernando De la Torre. 2015. Factorized graph matching. *IEEE transactions on pattern analysis and machine intelligence*, pages 1774–1789.

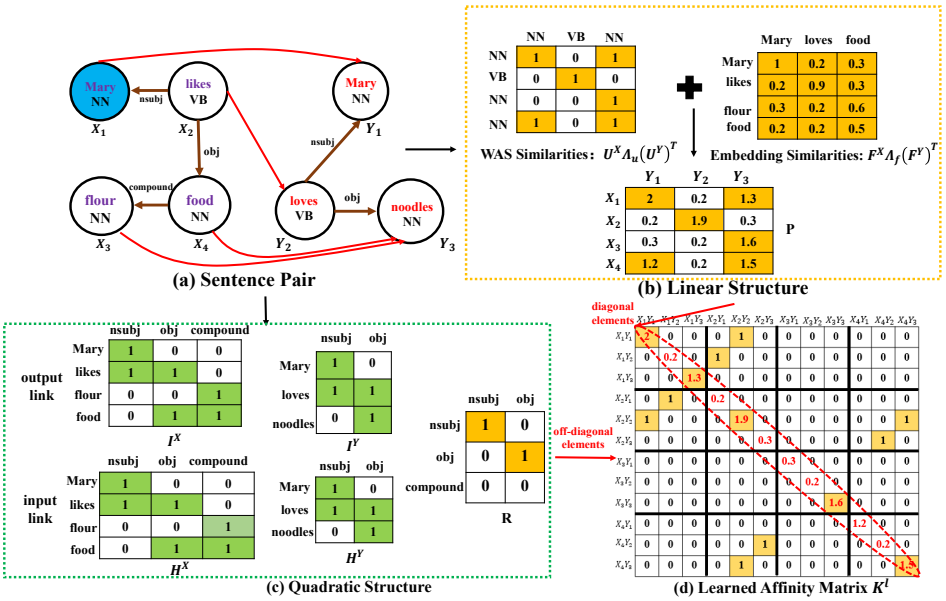

Figure 6: A working example of factorized affinity matrix K^l with aforementioned example ("Mary likes flour food", "Mary loves noodles"). The affinity matrix can be factorized into six matrices: $\mathbf{P}, \mathbf{R}, \mathbf{I}^X, \mathbf{I}^Y, \mathbf{H}^X, \mathbf{H}^Y$

Appendix A: An intuitive example on affinity matrix factorization

Figure 6 gives a working example of factorizing the affinity matrix $\mathbf{K}^l \in \mathbb{R}^{t_X t_Y \times t_X t_Y}$ in Equation (3) (Zhou and De la Torre, 2015):

$$\mathbf{K}^l = \text{diag}(\text{vec}(\mathbf{P})) + (\mathbf{I}^X \otimes_{\mathcal{K}} \mathbf{I}^Y)\text{diag}(\text{vec}(\mathbf{R}))(\mathbf{H}^X \otimes_{\mathcal{K}} \mathbf{H}^Y)^T, \quad (9)$$

with the aforementioned example sentence pair: ("Mary likes flour food", "Mary love noodles"). As shown in Figure 6(a), the words, POS and syntactic dependencies are represented in the nodes and edges, respectively.

The diagonal and off-diagonal elements in the affinity matrix (Figure 6(d)) represent the affinity of sentence linear structures and quadratic structures, respectively. According to Zhou and De la Torre (2015), the affinity matrix $\mathbf{K}^l$ can be factorized into six matrices $\mathbf{P}, \mathbf{R}, \mathbf{I}^X, \mathbf{I}^Y, \mathbf{H}^X, \mathbf{H}^Y$ (shown in Figure 6(b,c)) and defined in Section 4.

Appendix B: Robustness of ISG

ISG has a set of important hyper-parameters λ_a which trade-off the affinity regularizer $\mathcal{R}_a$ and matching loss $\mathcal{L}_m$. We conducted experiments on the Scitail test set with BERT$_{BASE}$ as the encoder to test the sensitivity of these hyper-parameters. Figure 7 illustrates the performance changes w.r.t. λ_a in terms of accuracy, where $\lambda_a \in [3e-3, 1.1e-2]$. We can see that ISG performed best when $\lambda_a \approx 8e-3$. However, the performance changes

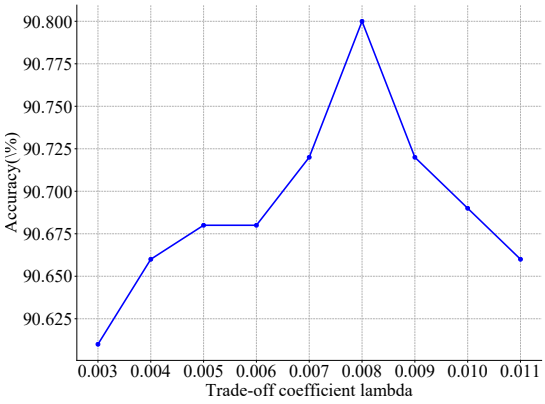

Figure 7: Accuracy curve of ISG-BERT$_{BASE}$ w.r.t. λ_a (trade-off coefficient for affinity regularizer) on the Scitail test set.

were not severe (from 90.6% to 90.8% in terms of accuracy). We conclude that (1) the introduction of the affinity regularizer enables ISG to have some tolerances to the errors caused by the NLP parser, which inevitably occurs in real-world applications; (2) ISG is robust and not sensitive to the λ_a.

Hierarchical information matters: Text classification via tree based graph neural network

Chong Zhang*, **He Zhu***, **Xingyu Peng**, **Junran Wu**†, **Ke Xu**†
State Key Lab of Software Development Environment, Beihang University, Beijing, 100191, China
{chongzh, Roy_Zh, xypeng, wu_junran, kexu}@buaa.edu.cn

Abstract

Text classification is a primary task in natural language processing (NLP). Recently, graph neural networks (GNNs) have developed rapidly and been applied to text classification tasks. As a special kind of graph data, the tree has a simpler data structure and can provide rich hierarchical information for text classification. Inspired by the structural entropy, we construct the coding tree of the graph by minimizing the structural entropy and propose HINT, which aims to make full use of the hierarchical information contained in the text for the task of text classification. Specifically, we first establish a dependency parsing graph for each text. Then we designed a structural entropy minimization algorithm to decode the key information in the graph and convert each graph to its corresponding coding tree. Based on the hierarchical structure of the coding tree, the representation of the entire graph is obtained by updating the representation of non-leaf nodes in the coding tree layer by layer. Finally, we present the effectiveness of hierarchical information in text classification. Experimental results show that HINT outperforms the state-of-the-art methods on popular benchmarks while having a simple structure and few parameters.

1 Introduction

Text classification is an essential problem in NLP. There are numerous applications of text classification, such as news filtering, opinion analysis, spam detection, and document organization (Aggarwal and Zhai, 2012). Recently, GNNs have developed rapidly. GNNs learn the representation of each node by aggregating the information of neighboring nodes and can retain structural information in the graph embedding. Therefore, many graph-based methods are applied to text classification and achieve good performance. (Yao et al., 2019) proposed TextGCN, which is the first method to employ a Graph Convolutional Network (GCN) in the text classification task. They built a heterogeneous graph containing word nodes and document nodes for the corpus and transformed the text classification task into a node classification task. TextGCN outperformed other traditional methods and attracted much attention, which has led to increasingly more applications of graph-based methods in text classification. In Huang et al. (2019); Zhang et al. (2020), the text classification task was converted into the graph classification task. They built text-level co-occurrence graphs for each data. Huang et al. (2019) employed a Message Passing Mechanism (MPM) and outperformed TextGCN. A Gated Graph Neural Network (GGNN) was employed in (Zhang et al., 2020) and achieved state-of-the-art performance.

Different from normal graph data, tree-structured data is a simpler data structure with rich hierarchical information. In the text classification task, despite plenty of efforts that have been devoted to the adoption of GNNs, none of them has realized the rich hierarchical information in text, which has already been employed in other NLP tasks. TrDec was proposed for the NMT task to generate a target-side tree topology and uses the tree to help the translation process (Wang et al., 2018). In knowledge-based question answering tasks, Zhu et al. (2020) treated the query as a tree and used the tree-based LSTM to model the context of the entities or relationships in the query. You et al. (2019) clustered the labels in the extreme multi-label text classification task and built a label tree. Lyu et al. (2020) proposed a rumor detection model based on tree transformer to better exploit user interactions in the dialogues. Hierarchical information in text is likely to be helpful for text classification tasks, so it should be better utilized.

Inspired by structural entropy (Li and Pan, 2016; Wu et al., 2022b,a), we propose a novel

*Equal Contribution.
†Correspondence to: Junran Wu, Ke Xu.

Proceedings of the 29th International Conference on Computational Linguistics, pages 950–959
October 12–17, 2022.

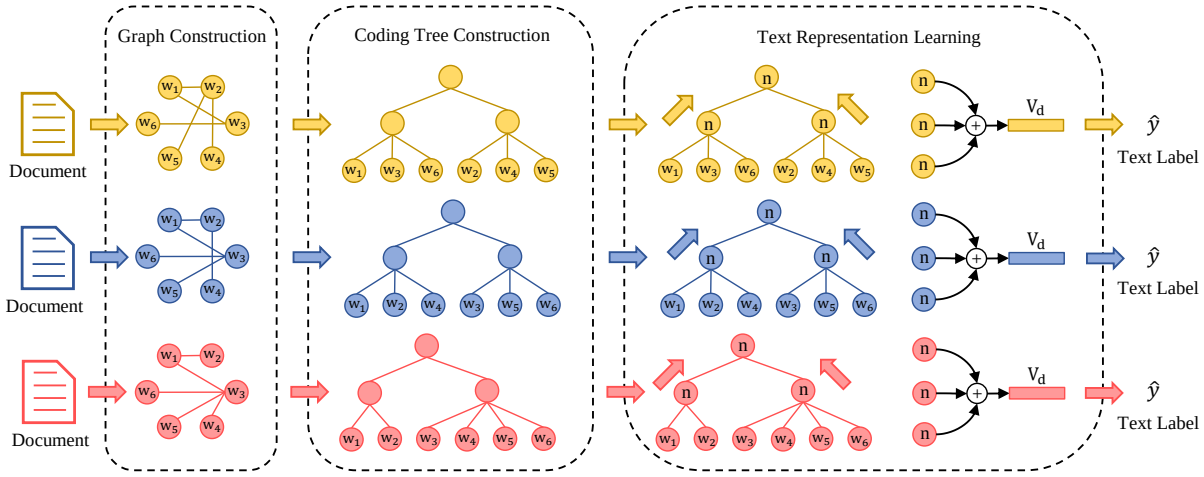

Figure 1: The architecture of HINT. w_i nodes represent the word nodes, others are non-leaf nodes in the coding tree.

model based on tree structure for text classification, named HINT. Structural entropy can measure the complexity of the hierarchical information of the graph, and decode its key structure. As shown in Figure 1, we first build individual graph for each document through dependency parsing. Then the graphs are transformed into their corresponding coding trees by a structural entropy minimization algorithm. The coding tree not only retains the crucial features of data but also excludes many other features that worsen the model. The model classifies the entire document by learning the representation of the coding tree. So far, we not only make better use of the hierarchical information in the text data but also represent text with a simpler data structure (i.e., tree). We conduct several experiments to verify the advantages of our method over the baselines. To sum up, our contributions are as follows:

- For the first time, we explore the effectiveness of hierarchical information of documents in text classification.
- We propose a novel method, HINT, which aims to parse and represent the hierarchical information of documents.
- The results demonstrate that our method not only outperforms several text classification baselines but is also much simpler in structure than other graph-based models.

2 Related Work

In recent years, the graph-based text classification method transforms the text classification task into a graph classification task or a node classification task and has achieved good performance in the text classification task. Different from traditional deep learning models (Kim, 2014; Liu et al., 2016), the graph-based text classification methods usually capture the rich relationships between nodes in the graph by constructing document graphs or corpus graphs, then apply GNN to learn the embedding of the document, and finally input the embedding to the classification layer. For graph construction, one is to construct a static graph for text or corpus. Some methods are to construct a single heterogeneous graph for the entire corpus (Yao et al., 2019; Liu et al., 2020). Other studies construct a separate graph for each document to handle the inductive learning setting (Huang et al., 2019; Zhang et al., 2020). Except for static graphs, the construction of dynamic graphs does not rely on prior knowledge and can be jointly learned with the model (Chen et al., 2020). For the learning of graph representation, various GNN models are used in text classification, such as GCN (Yao et al., 2019; Liu et al., 2020), GGNN (Zhang et al., 2020), MPM (Huang et al., 2019), and GAT-based model (Linmei et al., 2019).

Tree structure data is a special kind of graph data with a simple structure and rich hierarchical information. With the development of deep learning, many models use trees to help solve NLP tasks. Some methods process the data to get the tree structure data and use it to help the model. (Wang et al., 2018) utilized the strong correlation between grammatical information and tree structure, and a tree-based decoder TrDec is proposed for the NMT task. TrDec generates a target-side tree topology and

uses the tree to guide the translation process. (You et al., 2019) proposed a model based on label tree and attention mechanism for extreme multi-label text classification. The label tree is constructed by clustering the labels and the model is trained from top to bottom. In addition to using the generated tree, the deep learning model of the tree structure is also widely used. Tree-structured multilinear principal component analysis (TMPCA) used the PCA of each layer to transfer the representation of two adjacent words to the next layer until the entire text is reduced to a single-word vector (Su et al., 2018). (Zhu et al., 2020) treated the query as a tree and proposed a tree-based LSTM in the Knowledge-based question answering task to model the context of entities or relationships in the query.

The tree structure can provide hierarchical information for the model, and the tree based deep learning model can make better use of the grammatical information through the hierarchy of the model. However, hierarchical information is not well studied in graph-based text classification methods. In this work, we aim to fill the gap in GNN-based text classification.

3 Method

In this section, we will introduce HINT in detail. First, we will explain the method of constructing a graph for each text. Then, we introduce the coding tree construction algorithm that can decode the hierarchical information in text data. Finally, we will show how the model learns the hierarchical information and text representation based on tree-structured data and how to predict the label for a given text based on the learned representations.

3.1 Graph Construction

In previous graph-based text classification models, there are two ways to construct a graph for text. In (Yao et al., 2019), the corpus is constructed into a heterogeneous graph containing word nodes and document nodes. The weight of the edge between nodes is the point-wise mutual information (PMI) of the words or the TF-IDF value. This method can explicitly model the global word co-occurrence and can easily adapt to graph convolution. The other method is to construct a graph for each text. Huang et al. (2019); Zhang et al. (2020) construct a graph for a textual document by representing unique words as vertices and co-occurrences between words as edges. This method reduces memory consumption and is friendly to new text.

However, the graphs constructed by the above methods do not contain rich syntax and semantics information. In addition, the method based on co-occurrence treats the words at different positions equally, which causes a lack of position information in the representations of graph nodes. To retain more features in the constructed graphs, we use dependency parsing to construct a graph in HINT. We perform dependency parsing on each sentence in the document to obtain the dependencies between words. In addition, the dependency parsing result of each sentence contains a root word. We connect the root words of adjacent sentences to form a complete dependency parsing graph.

Take a document with l words $D = \{w_1, \ldots, w_i, \ldots, w_l\}$, where w_i is the i_{th} word of document. The set of dependencies between words is $DP = \{dr_{ij} | i \neq j; i, j \leq l\}$, where dr_{ij} denotes the dependency relation of words i and j. The edges between pairs of words with dependencies is $E_w = \{e_{ij} | dr_{ij} \in DP\}$. The root word set of each sentence is $DR = \{r_i | i \leq n\}$, where n represents the number of sentences in the document. $E_r = \{e_{ij} | r_i, r_j \in DR \wedge j = i+1\}$ represents the set of edges between the root words of adjacent sentences. The dependency parsing graph $G = (V, E)$ for a text is defined as:

$$V = \{w_i | i \in [1, l]\}, \tag{1}$$

$$E = \{E_w \cup E_r\}, \tag{2}$$

where V and E are the node set and edge set of the graph respectively. Dependency parsing can analyze the semantic associations between words. Using words as nodes and dependency relationships as edges, the graph has rich semantic and structural information.

3.2 Coding Tree Construction

After the text is transformed into graph structure data through dependency analysis, each graph has rich structural information. In this subsection, we introduce our method to decode the hierarchical information in text data from constructed graphs.

In (Li and Pan, 2016), the structural entropy of the graph is defined as the average amount of information of the codewords obtained by a random walk in a specific coding pattern. According to the different coding patterns, structural entropy

can measure the dynamic information of the graph. Given a graph $G = (V, E)$, the structural entropy of G on partitioning tree T_p is defined as:

$$H^{T_p}(G) = -\sum_{\alpha \in T_p} \frac{g_\alpha}{2m} \log \frac{V_\alpha}{V_{\alpha^-}}, \tag{3}$$

where $m = |E|$, α is the non-root node of T_g and represents a subset of V, α^- is the parent of α, g_α represents the number of edges with only one end point in α and the other end outside α, V_α and V_{α^-} is the sum of the degree of nodes in α and α^-. The structural entropy of G is defined by $H(G) = min_{T_p} H^{T_p}(G)$. T_p is also called the coding tree. Coding tree is simpler form of data, while retaining key features of the original graph. For a certain coding mode, the height of the coding tree should be fixed. Therefore, the h-dimensional structural entropy of the graph G determined by the coding tree T with a certain height h can be computed as:

$$H^h(G) = \min_{\{T|h(T)\leq h\}} H^T(G). \tag{4}$$

We designed a graph coding algorithm based on minimizing structural entropy to transform the graph to its coding tree. The coding tree construction algorithm shown in Algorithm 1 is based on the principle of minimizing structural entropy to construct a h-dimensional coding tree with a certain height h. So the coding tree T with given height h is computed by $T = SEMA(G, h)$, where $T = (V_T)$, $V_T = (V_T^0, \ldots, V_T^h)$ and $V_T^0 = V$. $SEMA$ refers to the structural entropy minimization algorithm. In $SEMA$, the graph is first transformed into a full-height binary coding tree, and then the tree is folded into a coding tree with certain height h.

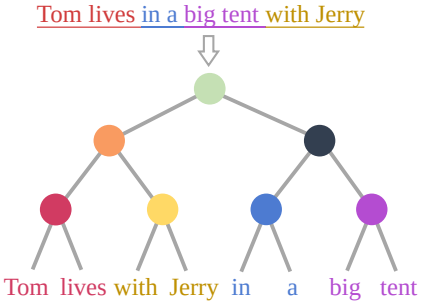

Figure 2: A text and its corresponding coding tree with a height of 3. The green node represents the root node of the coding tree, and other points refer to other inner nodes with various hierarchical information.

Algorithm 1 Structural Entropy Minimization Algorithm

Input: Adjacency matrix A_{m*m} of the graph $G = \{V, E\}$
Parameter: The height h of the coding tree
Output: The coding tree $T = \{V_T, E_T\}$ of the graph G

Let $heap = [], V_T = V, unmerge = m, M = \varnothing$.
for n_i, n_j in V and $E_{ij} \in E$ **do**
  $T^{'} = T.Combine(n_i, n_j)$
  $\Delta e = entropy(T) - entropy(T^{'})$
  $heap.push(\Delta e, n_i, n_j)$
end for
while $unmerge > 1$ **do**
  $n_1, n_2 = heap.pop(min\Delta e)$
  if $n_1 \notin M$ and $n_2 \notin M$ **then**
    $n_{new} = merge(n_1, n_2)$
    update $V_T, M, unmerge$
    update $heap$ like line 4 to 6
  end if
end while
$root = n_{new}$
update $heap$ like line 4 to 6
while the height of the tree $> h$ **do**
  $n_p, n_c = heap.pop(min\Delta e)$
  $n_p = compress(n_p, n_c)$
  $V_T.del(n_c)$
end while
return T

For a text and its corresponding coding tree, all words in the text are the leaf nodes, and the hierarchical information in text are decoded into the hierarchical structure of the coding tree. Figure 2 shows an example of a text and its corresponding coding tree. We can see that the coding tree divides the words of the text into four parts. The information contained in each non-leaf node can be interpreted as the semantic information of all its child nodes, so non-leaf nodes at different levels contain semantic information with different granularities. In addition, the two words "Tom" and "Jerry" that are farther apart in the original text are closer in the coding tree while retaining the correct semantic information.

3.3 Text representation learning

Based on the decoded hierarchical structure of coding trees, we aim to learning the text representation with this hierarchical information. Specifically, fol-

lowing the message passing mechanism in GNNs, we intend to iteratively update the node representation of coding tree from leaves to root node. Finally, model can obtain a representation of the text by using the structure of the coding tree and the features of the leaf nodes. The number of layers of the model is the same as the height of the coding tree. The i_{th} layer on coding tree $T = (V_T, E_T)$ can be expressed as:

$$x_v^i = MLP^i(\sum\nolimits_{n \in C(v)} x_n^{i-1}), \tag{5}$$

where $v \in V_T$, x_v^i is the feature vector of node v with height i, x^0 is the word embeddings, and $C(v)$ is the child nodes of v. The coding tree learning model starts from the leaf node layer and learns the representation of each node layer by layer until reaching the root node. Finally, all feature vectors of the nodes are used to compute a representation of the entire coding tree x_T:

$$\begin{aligned} x_T = Concat(Pool(\{x_v^i | v \in V_T^i\}) \\ |i = 0, 1, 2, \ldots, h)), \end{aligned} \tag{6}$$

where x_v^i is the feature vector of node v with height i in T, and h is the height of T. $Pool$ in Equation 6 can be replaced with a summation or averaging function. In the beginning, the non-leaf nodes have no representation, and the representation of the non-leaf nodes is updated as the information is propagated from the lower layers to the upper layers along the edges of the coding tree. In the process of propagation, the textual information from the leaf nodes interacts with the hierarchical information abstracted by the coding tree, so that the representation of the final document can contain more useful information. Existing methods have fully exploited the local information of text, and our model takes the global information of text into consideration by combining the hierarchical structure of text.

By converting the graph into a coding tree, the data structure becomes simpler, and the hierarchical information and main features of the graph are retained in the coding tree. In HINT, the node feature vector is aggregated in one direction owing to the hierarchical structure of the coding tree, which suggests that our learning model is simple and its convergence is strong. We take the coding trees $\mathbb{T} = (T_1, T_2, \ldots, T_n)$ and their feature matrices as inputs. Each word is represented by the GloVe (Pennington et al., 2014) vector and one-hot position encoding vector, and we concatenate these two vectors as feature matrix F. The representation of the entire graph can be obtained from Equation 6, and then the predicted label of the original text is computed as:

$$y_i = softmax(W x_{T_i} + b), \tag{7}$$

where $T_i \in \mathbb{T}$, x_{T_i} is the representation of coding tree T_i; and W and b are the weight and bias, respectively. The goal of training is to minimize the cross-entropy between the ground truth label and predicted label:

$$loss = -\sum_i g_i \log(y_i), \tag{8}$$

where g_i is the one-hot vector of the ground truth label.

4 Experiments

In this section, we evaluate the effectiveness of HINT[1] and report the experimental results.

Datasets. We utilize datasets including R8, R52, MR, and Ohsumed. R8 and R52 are subsets of the Reuters 21578 dataset. MR is a movie review dataset used for sentiment classification in which each review is a single sentence. The Ohsumed corpus, which is designed for multilabel classification, is from the MEDLINE database. In this paper, we only use single-label data like other GNN-based text classification models (Yao et al., 2019; Huang et al., 2019). We employ StanfordNLP (Qi et al., 2018) to build the dependency graphs for all datasets. The statistics of our datasets are summarized in Table 1.

Datasets	# Training	# Test	Categories	Avg. Length
MR	7108	3554	2	20.39
Ohsumed	3357	4043	23	193.79
R52	6532	2568	52	106.29
R8	5485	2189	8	98.87

Table 1: Summary statistics of datasets.

Baselines. In this paper, we aim to address the utilization of hierarchical information in text classification with GNNs; thus, besides a bunch of popular baselines, we mainly select the comparison methods based on GNNs. We divide the baseline models into three categories: (i) traditional deep learning methods, including the CNN and

[1]The code of HINT can be found at https://github.com/Daisean/HINT.

Model	MR	R8	R52	Ohsumed
CNN(Non-static)	**77.75**± 0.72	95.71± 0.52	87.59± 0.48	58.44± 1.06
RNN(Bi-LSTM)	77.68± 0.86	96.31± 0.33	90.54± 0.91	49.27± 1.07
fastText	75.14± 0.20	96.13± 0.21	92.81± 0.09	57.70± 0.49
SWEM	76.65± 0.63	95.32± 0.26	92.94± 0.24	63.12± 0.55
TextGCN	76.74± 0.20	97.07± 0.10	93.56± 0.18	68.36± 0.56
Huang et al. (2019)	-	97.80± 0.20	94.60± 0.30	**69.40± 0.60**
S^2GC	76.70± 0.00	97.40± 0.10	94.50± 0.20	68.50± 0.10
HINT	**77.03± 0.12**	**98.12± 0.09**	**95.02± 0.18**	**68.79± 0.12**

Table 2: Test accuracy(%) of models on text classification datasets. The average standard deviation of our model is reported based on ten runs.

LSTM; (ii) word embedding methods, including fastText (Joulin et al., 2017) and SWEM (Shen et al., 2018); and (iii) graph-based methods for text classification, including the spectral approach-based TextGCN , S^2GC (Zhu and Koniusz, 2021), and nonspectral method-based (Huang et al., 2019).

Settings. We randomly divide the training set into the training set and the validation set at a ratio of 9:1. We use the Adam optimizer with an initial learning rate of 10^{-3} and set the dropout rate to 0.5. The height of the coding tree is between 2 and 12. We set the sum or average function as the initial function of $Pool$ in Equation 6. For word embedding, we use pretrained GloVe with the dimension of 300, and the out-of-vocabulary (OOV) words are randomly initialized from the uniform distribution [-0.01, 0.01]. For each position, we set this position in the vector to 1 and the rest to 0 to get the position encoding vector. We concatenate the GloVe vector and position encoding vector as the initial representation of graph nodes.

4.1 Experimental Results

Table 2 presents the performance of our model and baselines. Graph network-based methods generally outperform other types of methods because of the inclusion of structural information. We can observe that the performance of our model is generally better than those of other graph network-based methods. Traditional deep learning methods (CNN and RNN) perform well on MR dataset with relatively short text lengths but are not as good at processing long text. The word embedding-based methods (fastText and SWEM) use word embedding with contextual information and perform better on the R52 and Ohsumed datasets than traditional deep learning methods.

TextGCN is the first method to apply a graph neural network method in text classification. TextGCN learns the representation of nodes through corpus-level co-occurrence graphs. Huang et al. (2019) uses the co-occurrence window and message passing mechanism to learn the representation of nodes. S^2GC is an extension of the Markov Diffusion Kernel used to make the information aggregation in graphs more efficient. The graph learning methods enable each node to learn a better representation by using the information of its farther neighbors and accordingly performs well on all datasets. The HINT model encodes the graph and extracts the key structure through the structure entropy minimization algorithm, and learns on the coding tree to use the hierarchical information to update the node representation. The representations of non-leaf nodes in the coding tree are obtained by a facile method of layer-by-layer updating from the leaf nodes of the coding tree to the root node. The results show that HINT performs better on the MR, R8, and R52 datasets than other graph-based methods and achieves competitive performance on the Ohsumed dataset. Notably, HINT does not have a complicated structure and numerous parameters, but it still generally outperforms other baselines. Next, we will further analyze the height of the coding tree, the coding tree construction algorithm, the comparison with the state-of-the-art methods and the efficiency of the model.

4.2 Height of the coding tree

The height of the coding tree plays an important role in HINT. Different height coding trees reflect divergent hierarchical information, and the utilization of leaf node information is also disparate. Figure 3 shows the test performance of different height coding trees on the 4 datasets. For datasets with different average lengths, the optimal height of the

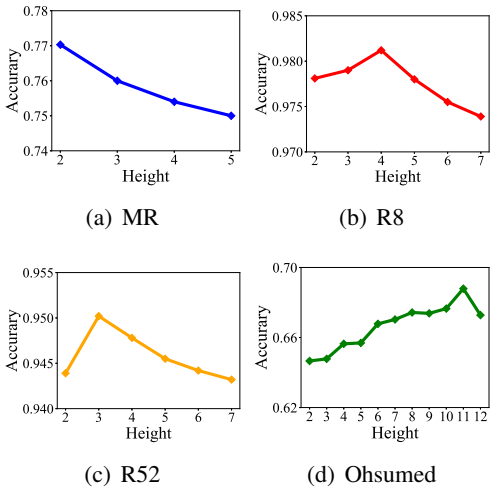

(a) MR (b) R8

(c) R52 (d) Ohsumed

Figure 3: The influence of height on the performance of the model on 4 datasets.

coding tree is different. Longer texts have more complex structural information, so a higher coding tree is needed to retain this information. In text representation learning, a deeper level is also needed to utilize hierarchical information. For the MR dataset, a height of 2 is the best. For the Ohsumed dataset with an average length of nearly 200, the best performance occurs when the height is 11. Therefore, an appropriate height will improve the quality of the representations and make better use of the original text and hierarchical information.

4.3 Validity of coding tree construction algorithm

In this subsection, we compare different coding tree construction methods to illustrate the effectiveness of the coding tree construction algorithm. For a tree of height h, we take all nodes of the text dependency analysis graph as leaf nodes of the tree. For each layer of the coding tree, we randomly select two nodes, use a new node as their parent node, and connect all nodes in the $h - 1_{th}$ layer to a root node (RT). The results are shown in Table 3.

Model	MR	R8	R52	Ohsumed
HINT(RT)	75.85	95.84	85.86	14.84
HINT	**77.03**	**98.12**	**95.02**	**68.79**

Table 3: Test accuracy(%) of models on text classification datasets with different coding tree construction method. For both construction methods, we use trees of the same height on the same dataset

The results point out that constructing a coding tree by randomly selecting nodes has a negative effect on the model. The difference between the two methods is small on the data set with the shortest average length, but becomes more pronounced as the data set becomes more complex, especially in the Ohsumed dataset. Random selection of nodes destroys the original semantic information, so it is difficult for the model to learn useful features. The coding tree constructed by the minimization structure entropy algorithm can retain the key information in the graph and abstract the hierarchical information of the text. The results demonstrate the effectiveness of minimizing structural entropy for text graphs.

4.4 Comparison with state-of-the-art method

With the development of pre-trained language models (PLMs), graph methods based on PLMs have also been applied to text classification tasks. BertGCN(Lin et al., 2021) achieves state-of-the-art performance by combining BERT(Devlin et al., 2019) with GCN(Kipf and Welling, 2017). For a fair comparison, we employ the BERT model ($\text{BERT}_{\text{base}}$) trained in BertGCN, freeze its parameters and input the text into BERT to get the initial node representation. Other experimental settings are consistent with BertGCN. Moreover, because BERT word embeddings contain positional information, we use the text co-occurrence graph like Huang et al. (2019). The result is shown in Table 4.

Model	MR	R8	R52	Ohsumed
BERT	85.7	97.8	96.4	70.5
BertGCN	86.0	**98.1**	96.6	**72.8**
HINT(BERT)	**86.4**	**98.1**	**96.8**	**71.2**

Table 4: Test accuracy(%) of models on text classification datasets with different position encodings.

The results point out that our model outperforms BertGCN on three out of four benchmarks. Because of BertGCN's settings, we truncate some long texts during preprocessing. Our model can decode the hierarchical information of the whole text, but the operation of truncating the text affects the integrity of the text. Ohsumed suffers the most as the dataset with the longest average length, yet we achieve competitive performance nonetheless. BertGCN is based on BERT and GCN, which pay more attention to local information, so incomplete text does not affect performance. On the remaining three datasets with occasional truncation, our model obtains outperformance, indicating that our

model can decode hierarchical information and focus on global features, further demonstrating the positive effect of hierarchical information on text classification tasks.

In general, our model achieves superior performance. With the more informative BERT word embeddings, HINT propagates the information in a bottom-up manner and obtains a better text representation. In the light of the accuracies achieved with BERT, our method shows excellent collaboration ability with large-scale pre-trained language models.

4.5 The Efficiency of HINT

In the graph-based baseline models, the computational complexity of TextGCN, S^2GC and Huang et al. (2019) is $O(hm)$, where h is the number of diffusion steps and m is the number of edges. The computational complexity of learning model in HINT is $O(n)$, where n is the number of nodes, which is much smaller than that of graph-based baseline models. In addition, we also compare the parameters and floating-point operations per second (FLOPs) of the models. Since S^2GC and Huang et al. (2019) do not have a complete model code implementation, we only compare the proposed model with TextGCN. Figure 4 shows the comparison of the parameters of HINT and TextGCN. We set the height of the coding tree $\in [2, 12]$ and use 2 as the step size and then run HINT and TextGCN on the same dataset. We can observe that the parameters of HINT gradually increase as the height increases, but the model with the most parameters is still dozens of times smaller than TextGCN.

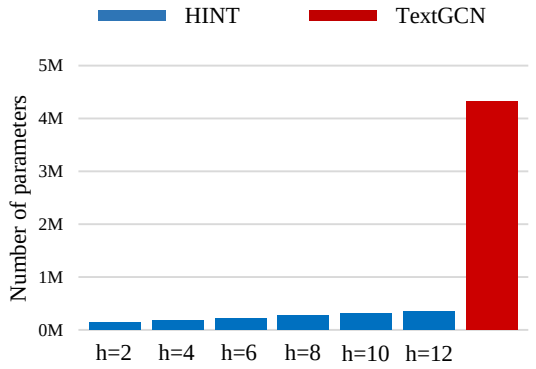

Figure 4: Comparison of the parameters of HINT and TextGCN on the Ohsumed dataset.

Moreover, we further compare the number of FLOPs of HINT and TextGCN on the same parameter settings. We set the hidden size of HINT to 96 and the batch size to 4. We calculate the FLOPs of the HINT and TextGCN models on four datasets. The results are shown in Figure 5. We can see that the calculation amount of HINT is also less than that of TextGCN. The performance of our model is not only better than those of other models, but the numbers of parameters and calculations are also very small, which further proves that our model is simple and effective.

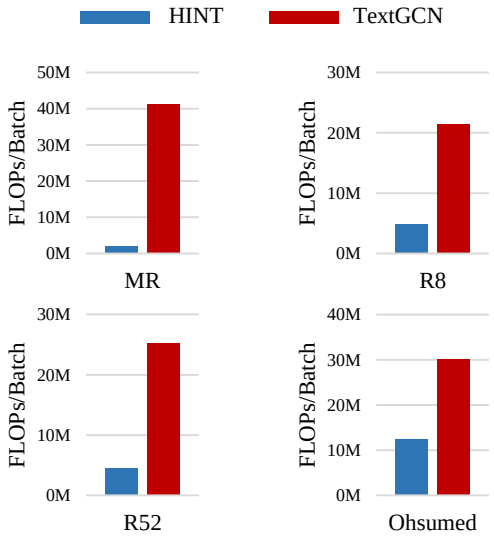

Figure 5: Comparison of the FLOPs of HINT and TextGCN.

For text classification or other NLP tasks, increasing the complexity of the neural network model and the number of network parameters can often achieve better performance. Therefore, the problem is that the development of NLP tasks highly depends on computing power. Our model achieves the improvement of model performance while reducing the complexity of the model. HINT makes the extraction of features not completely dependent on the deep learning network and greatly reduces the requirements for the computing power of the neural network.

5 Conclusion

In this paper, we proposed a novel method to address the limitation of previous works in text hierarchical information utilization. We build a dependency parsing graph for each text and construct a coding tree for each graph by structural entropy

minimization algorithm. Our model uses the hierarchy of the coding tree to learn the representation of each text. Experimental results demonstrate the ability of HINT to decode hierarchical information in text and show the positive effect of hierarchical information on text classification tasks. Our model achieves state-of-the-art performance with a simple structure and few parameters.

Acknowledgements

This research was supported by NSFC (Grant No. 61932002).

References

Charu C. Aggarwal and ChengXiang Zhai. 2012. *A Survey of Text Classification Algorithms*, pages 163–222. Springer US, Boston, MA.

Yu Chen, Lingfei Wu, and Mohammed Zaki. 2020. Iterative deep graph learning for graph neural networks: Better and robust node embeddings. *Advances in Neural Information Processing Systems*, 33.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *NAACL*, pages 4171–4186, Minneapolis, Minnesota. Association for Computational Linguistics.

Lianzhe Huang, Dehong Ma, Sujian Li, Xiaodong Zhang, and Houfeng Wang. 2019. Text level graph neural network for text classification. In *EMNLP-IJCNLP*, pages 3444–3450, Hong Kong, China. Association for Computational Linguistics.

Armand Joulin, Edouard Grave, Piotr Bojanowski, and Tomas Mikolov. 2017. Bag of tricks for efficient text classification. In *Association for Computational Linguistics*, pages 427–431, Valencia, Spain. Association for Computational Linguistics.

Yoon Kim. 2014. Convolutional neural networks for sentence classification. In *EMNLP*, pages 1746–1751, Doha, Qatar. Association for Computational Linguistics.

Thomas N. Kipf and Max Welling. 2017. Semi-supervised classification with graph convolutional networks. In *International Conference on Learning Representations (ICLR)*.

Angsheng Li and Yicheng Pan. 2016. Structural information and dynamical complexity of networks. *IEEE Transactions on Information Theory*, 62(6):3290–3339.

Yuxiao Lin, Yuxian Meng, Xiaofei Sun, Qinghong Han, Kun Kuang, Jiwei Li, and Fei Wu. 2021. BertGCN: Transductive text classification by combining GNN and BERT. In *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*, pages 1456–1462, Online. Association for Computational Linguistics.

Hu Linmei, Tianchi Yang, Chuan Shi, Houye Ji, and Xiaoli Li. 2019. Heterogeneous graph attention networks for semi-supervised short text classification. In *EMNLP-IJCNLP*, pages 4821–4830, Hong Kong, China. Association for Computational Linguistics.

Pengfei Liu, Xipeng Qiu, and Xuanjing Huang. 2016. Recurrent neural network for text classification with multi-task learning. In *IJCAI*, IJCAI'16, page 2873–2879. AAAI Press.

Xien Liu, Xinxin You, Xiao Zhang, Ji Wu, and Ping Lv. 2020. Tensor graph convolutional networks for text classification. *AAAI*, 34(05):8409–8416.

Chen Lyu, Weijie Liu, and Ping Wang. 2020. Few-shot text classification with edge-labeling graph neural network-based prototypical network. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 5547–5552, Barcelona, Spain (Online). International Committee on Computational Linguistics.

Jeffrey Pennington, Richard Socher, and Christopher Manning. 2014. GloVe: Global vectors for word representation. In *EMNLP*, pages 1532–1543, Doha, Qatar. Association for Computational Linguistics.

Peng Qi, Timothy Dozat, Yuhao Zhang, and Christopher D. Manning. 2018. Universal dependency parsing from scratch. In *Proceedings of the CoNLL 2018 Shared Task: Multilingual Parsing from Raw Text to Universal Dependencies*, pages 160–170, Brussels, Belgium. Association for Computational Linguistics.

Dinghan Shen, Guoyin Wang, Wenlin Wang, Martin Renqiang Min, Qinliang Su, Yizhe Zhang, Chunyuan Li, Ricardo Henao, and Lawrence Carin. 2018. Baseline needs more love: On simple word-embedding-based models and associated pooling mechanisms. In *Association for Computational Linguistics*, pages 440–450, Melbourne, Australia. Association for Computational Linguistics.

Yuanhang Su, Yuzhong Huang, and C.-C. Jay Kuo. 2018. Efficient text classification using tree-structured multi-linear principal component analysis. In *2018 24th International Conference on Pattern Recognition (ICPR)*, pages 585–590.

Xinyi Wang, Hieu Pham, Pengcheng Yin, and Graham Neubig. 2018. A tree-based decoder for neural machine translation. In *EMNLP*, pages 4772–4777, Brussels, Belgium. Association for Computational Linguistics.

Junran Wu, Xueyuan Chen, Ke Xu, and Shangzhe Li. 2022a. Structural entropy guided graph hierarchical pooling. In *International Conference on Machine Learning*, pages 24017–24030. PMLR.

Junran Wu, Shangzhe Li, Jianhao Li, Yicheng Pan, and Ke Xu. 2022b. A simple yet effective method for graph classification. In *IJCAI-22*, pages 3580–3586. International Joint Conferences on Artificial Intelligence Organization. Main Track.

Liang Yao, Chengsheng Mao, and Yuan Luo. 2019. Graph convolutional networks for text classification. *AAAI*, 33(01):7370–7377.

Ronghui You, Zihan Zhang, Ziye Wang, Suyang Dai, Hiroshi Mamitsuka, and Shanfeng Zhu. 2019. Attentionxml: Label tree-based attention-aware deep model for high-performance extreme multi-label text classification. In *Advances in Neural Information Processing Systems*, volume 32. Curran Associates, Inc.

Yufeng Zhang, Xueli Yu, Zeyu Cui, Shu Wu, Zhongzhen Wen, and Liang Wang. 2020. Every document owns its structure: Inductive text classification via graph neural networks. In *Association for Computational Linguistics*, pages 334–339, Online. Association for Computational Linguistics.

Hao Zhu and Piotr Koniusz. 2021. Simple spectral graph convolution. In *International Conference on Learning Representations*.

Shuguang Zhu, Xiang Cheng, and Sen Su. 2020. Knowledge-based question answering by tree-to-sequence learning. *Neurocomputing*, 372:64–72.

SelfMix: Robust Learning Against Textual Label Noise with Self-Mixup Training

Dan Qiao[1], **Chenchen Dai**[1], **Yuyang Ding**[1], **Juntao Li**[1],
Qiang Chen[2], **Wenliang Chen**[1], **Min Zhang**[1]
[1]Institute of Computer Science and Technology, Soochow University, China
[2]Alibaba Group
{danqiao.jordan,morningcc125,yyding.me}@gmail.com
{ljt,wlchen,minzhang}@suda.edu.cn; lapu.cq@alibaba-inc.com

Abstract

The conventional success of textual classification relies on annotated data, and the new paradigm of pre-trained language models (PLMs) still requires a few labeled data for downstream tasks. However, in real-world applications, label noise inevitably exists in training data, damaging the effectiveness, robustness, and generalization of the models constructed on such data. Recently, remarkable achievements have been made to mitigate this dilemma in visual data, while only a few explore textual data. To fill this gap, we present SelfMix, a simple yet effective method, to handle label noise in text classification tasks. SelfMix uses the Gaussian Mixture Model to separate samples and leverages semi-supervised learning. Unlike previous works requiring multiple models, our method utilizes the dropout mechanism on a single model to reduce the confirmation bias in self-training and introduces a textual level mixup training strategy. Experimental results on three text classification benchmarks with different types of text show that the performance of our proposed method outperforms these strong baselines designed for both textual and visual data under different noise ratios and noise types. Our anonymous code is available at https://github.com/noise-learning/SelfMix.

1 Introduction

The excellent performance of deep neural networks (DNNs) depends on data with high-quality annotations. However, data obtained from the real world is inevitably mixed with wrong labels (Guan et al., 2018; Aït-Sahalia et al., 2010; Liu et al., 2020b). Models trained on these noisy datasets would easily overfit the noisy labels (Algan and Ulusoy, 2020; Liu et al., 2020a), especially for pre-trained large models (Zhang and Li, 2021), and the performance will be negatively affected.

Research on learning with noisy labels (LNL) has gained popularity. Previous work has revealed that clean samples and noisy samples play different roles in the training process and behave differently in terms of loss values or convergence speeds etc. (Liu et al., 2020a). Different types of noise have different effects on the training. For instance, the impact of class-conditional noise (CCN) can simulate the confusion between similar classes, and the effect of instance-dependent noise (IDN) can be more complex.

Most of the current methods perform experiments on visual data. Label noise on visual data often goes against objective facts and is easy to distinguish. As for NLP, there may be disagreement even among expert annotators due to the complexity of semantic features and the subjectivity of language understanding. For example, suppose there is a piece of news about "The Economic Benefit of Competitive Sports to our Cities". In that case, it is hard to tell whether it belongs to Economic news or Sports news without fully understanding the contextual information. Although a few works pay attention to the natural language area, their methods are mostly based on the trained-from-scratch models like LSTM and Text-CNN (Garg et al., 2021; Jindal et al., 2019). However, PLMs might be a better choice since the whole training process can be divided into two stages, and the wrong labels do not corrupt the pre-training process. Table 2 makes comparisons between PLMs and traditional networks on the robustness against label noise.

In conclusion, it is vital to explore how to learn with noisy labels on textual data and use the robust PLMs as the base model. This paper proposes SelfMix, i.e., a self-distillation robust training method based on the pre-trained models. Section 2 introduces some related works and explains the motivation of our proposed method.

Our contributions can be concluded as follows:

- We propose SelfMix, a simple yet effective method to help learn with noisy labels, which utilizes a self-training approach. Our method

Proceedings of the 29th International Conference on Computational Linguistics, pages 960–970
October 12–17, 2022.

only needs a single model and utilizes a mixup training strategy based on the aggregated representation from pre-trained models.

- We perform comprehensive experiments on three different types of text classification benchmarks under various noise settings, including the challenging instance-dependent noise, which is usually ignored in other works on textual data, which demonstrate the superiority of our proposed method over strong baselines.

2 Related Work

Learning with Noisy Labels. A direct yet effective idea to handle label noise is to find the noisy samples and reduce their influence by resampling or reweighting (Rolnick et al., 2017). Jiang et al. (2018) train another neural network to provide a curriculum to help StudentNet focus on the samples whose labels is probably correct. Han et al. (2018) jointly train two deep neural networks and feed each model the top $r\%$ samples with the lowest loss evaluated by the other model in each mini-batch. Following Han et al. (2018), Yu et al. (2019) explore how disagreement can help the model. Some researchers believe that there exists a transition from ground-truth label distribution to the noisy label distribution and estimate the noise transition matrix to absorb this transition (Goldberger and Ben-Reuven, 2016). Northcutt et al. (2021) directly estimate the joint distribution matrix between the noisy labels and real labels. Garg et al. (2021) use a fully connected layer to capture the distribution transition. However, most of these methods either need model ensembling or require cross-validation, which is time-consuming and needs multiple parameters.

Some other works focus on designing a more robust training strategy. Since DNNs with Cross-Entropy loss tend to overfit noisy labels (Feng et al., 2021), some researchers redesign noise-robust loss functions (Wang et al., 2019b; Zhang and Sabuncu, 2018; Ghosh et al., 2017; Xu et al., 2019). When trained on noisy data, DNNs tend to learn from the clean data during an "early learning" phase before eventually memorizing the wrong data (Arpit et al., 2017; Zhang et al., 2021), based on which Liu et al. (2020a) offer an easy regularization capitalizing on early learning. Some other works like Xia et al. (2020) find that only partial parameters are essential for generalization, which offers us a new perspective to reconsider what difference exactly the noisy labels make to the model's learning. This kind of approach treats all samples indiscriminately thus the performance is sometimes unsatisfactory under a high noise ratio.

Some excellent work combines these two ideas (Ding et al., 2018; Li et al., 2020). Garg et al. (2021) add an auxiliary noise model N_M over the classifier to predict noisy labels and jointly train the classifier and the noise model through a de-noising loss function. Cheng et al. (2021) progressively sieve out corrupted examples and then leverage semi-supervised learning.

Mixup Training. Mixup training (Zhang et al., 2018) is a widely used data-augmentation method to alleviate memorization and sensitivity to adversarial samples on visual data. It combines the inputs and targets of two random training samples to generate augmented samples. However, applying mixup on textual data is a great challenge since linear interpolations on discrete inputs damage the semantic structure. Some literature has explored the textual mixup mechanism like: Chen et al. (2020) propose to mix the hidden vector in the last few encoder layers; Yoon et al. (2021) find a new way to combine two texts which can also be treated as a data augmentation strategy. In this paper, we do not make comparisons for the following reasons: (1) Our EmbMix is simpler in practical use and there is little difference in the final performance of various methods according to Chen et al. (2020). (2) Some other methods need data augmentation while EmbMix does not.

Proposed Method. Since simply redesigning a robust loss function tends to have poor performance under a high noise ratio, we combine sample selection with the robust training methods. Unlike the previous work that needs model ensembling or uses cross-validation, we train a single network with dropout to reduce confirmation bias in self-training. We make following improvements regarding to the characteristics of the textual data: (1) The decision boundaries in image-classification tasks are more clear. However, the main idea of the same text can vary under different contexts and sometimes there is even no absolute correct label. So we iteratively use the Gaussian Mixture Model (GMM) to fit the loss distribution and use the predicted soft label to replace the label of the fusing data rather than setting a threshold and arbitrarily discarding the undesired samples at the beginning. (2) Unlike

the pixel input of visual data, the input of text is discrete. So for the separated data, we leverage a manifold mixup training strategy based on the aggregated representation from the PLMs.

3 Methodology

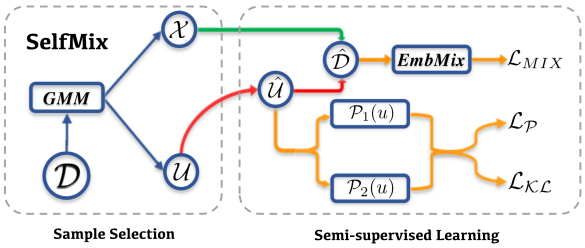

Figure 1: The overall framework of SelfMix

Figure 1 shows an overview of our proposed SelfMix. Our method first uses GMM to select the samples that are more likely to be wrong and erase their original labels. Then we leverage semi-supervised learning to jointly train the labeled set $\mathcal{X}$ (contains mostly clean samples) and an unlabeled set $\mathcal{U}$ (contains mostly noisy samples). We also introduce a manifold mixup strategy based on the hidden representation of the [CLS] token named EmbMix.

3.1 Preliminary

In real-world data collection, the observed labels are often corrupted. So the only difference between this task and the traditional text classification task is that a certain proportion of incorrect labels exist in training samples. Let $\mathcal{D} = \{(x_i, y_i)\}_{i=1}^{N}$ denote the original dataset, where N is the number of samples, x_i is the text of the ith sample, and y_i is the one-hot representation of the observed label of the ith sample. For the base model, we denote θ as the parameters of the pre-trained encoder model and ϕ as the parameters of MLP classifier head with 2 fully connected layers. The standard optimization method tries to minimize the empirical risk by applying the cross-entropy loss:

$$\mathcal{L} = \{\ell_i\}_{i=1}^{N} = \left\{-y_i^T \log\left(p\left(x_i;\theta,\phi\right)\right)\right\}_{i=1}^{N}, \quad (1)$$

where $p\left(x;\theta\right)$ denotes the softmax probability of the model output. We first warm up the model using $\mathcal{L}$ to make it capable of doing preliminary classification tasks without overfitting noisy labels and then perform SelfMix for the rest epochs.

3.2 Sample Selection

On noisy data, Deep neural networks will preferentially learn simple and logical samples first and reduce their loss. Namely, noisy samples tend to have a higher loss in the early stage (Zhang et al., 2021). Preliminary experiments show that the loss distributions of clean and noisy samples during training tend to subject to two Gaussian Distributions, where the loss of the clean samples hold a smaller mean value. Taking advantage of such training phenomena, we apply the popular used Gaussian Mixture Model (Arazo et al., 2019) to distinguish noisy samples by feeding the per-sample loss. For IDN, noisy labels rely on both input features and underlying true labels, so the noise in each class is different, making the loss scales from different classes vary greatly. The relatively high-loss samples in low-loss class may also be treated as clean samples. So we compute a class-regularization loss instead of the standard cross-entropy loss, which can better model the distributions in IDN. For each class c, the set $\mathcal{L}_c = \{\ell_i \mid y_i = c, i \in [N]\}$ contains the cross-entropy loss values of all samples with label c, then μ_c and σ_c denote the arithmetic mean and standard deviation of $\mathcal{L}_c$ respectively. Our regularization loss has the following form:

$$\mathcal{L}' = \{\ell_i'\}_{i=1}^{N} = \{(\ell_i - \mu_{y_i})/\sigma_{y_i}\}_{i=1}^{N}. \quad (2)$$

We feed the loss $\mathcal{L}$ ($\mathcal{L}'$ for IDN) to a 2-component GMM and use Expectation-Maximization (EM) algorithms to fit the GMM to the observations. Let $w_i = p(g|\ell_i')$ represent the probability of the ith sample belonging to the Gaussian component with smaller mean g, which can also be considered as the clean probability due to the small-loss theory (Arpit et al., 2017). By setting the threshold τ for the probability w_i, we can divide the original dataset $\mathcal{D}$ into a labeled set $\mathcal{X}$ and an unlabeled set $\mathcal{U}$ where the labels of samples that are more likely to be wrong will be erased:

$$\begin{aligned} \mathcal{X} &= \{(x_i, y_i) \mid x_i \in \mathcal{D}, w_i \geq \tau\}, \\ \mathcal{U} &= \{(x_i) \mid x_i \in \mathcal{D}, w_i < \tau\}. \end{aligned} \quad (3)$$

3.3 Semi-supervised Self-training

To make semi-supervised learning work better, we first do pre-process on the unlabeled set. For the unlabeled set $\mathcal{U}$, the original label is most likely wrong and has been discarded. Therefore, we generate the soft label $\hat{y}$ by sharpening the model's predicted distribution, making the distribution more

concentrated (Zoph et al., 2020).

$$\hat{y} = \text{Sharpen}\left(p\left(x;\theta,\phi\right)\right), \tag{4}$$

$$\hat{\mathcal{U}} = \left\{\left(x_i,\hat{y}_i\right) \mid x_i \in \mathcal{U}\right\}, \tag{5}$$

$$\hat{\mathcal{D}} = \mathcal{X} \cup \hat{\mathcal{U}}. \tag{6}$$

Here $\text{Sharpen}\left(\cdot\right)$ is the temperature sharpening function commonly used in self-training. $\hat{\mathcal{D}}$ contains the clean samples with original labels and noisy samples with predicted soft labels.

Textual Mixup based on EmbMix. Mixup training strategy is widely used in semi-supervised learning and noise-robust training (Zhang et al., 2018). It applies linear interpolation to the input vectors and associated targets. Although image data can be mixed on the pixel level, mixing the discrete word index makes no sense for text classification. Considering that the `[CLS]` embedding encoded by PLMs has the ability of semantic representations, we attempt to apply interpolations on the `[CLS]` embedding. Specifically, randomly choose two samples $\left(x_i,y_i\right)$, $\left(x_j,y_j\right)$ and the mixed sample $\left(e_i',y_i'\right)$ is defined as:

$$\lambda \sim \text{Beta}(\alpha,\alpha), \tag{7}$$

$$\lambda' = \max(\lambda,1-\lambda), \tag{8}$$

$$e_k = \text{Encoder}\left(x_k;\theta\right), \tag{9}$$

$$e_i' = \lambda' e_i + \left(1-\lambda'\right) e_j, \tag{10}$$

$$y_i' = \lambda' y_i + \left(1-\lambda'\right) y_j, \tag{11}$$

where $\text{Encoder}\left(x;\theta\right)$ denotes the sentence `[CLS]` embeddings obtained by pre-trained models.

Finally, the EmbMix method for dataset $\hat{\mathcal{D}}$ is as follows:

$$\tilde{\mathcal{D}} = \left\{\left(e_i',y_i'\right) \mid \left(x_i,y_i\right),\left(x_j,y_j\right) \in \hat{\mathcal{D}}\right\}, \tag{12}$$

where $\left(e_i',y_i'\right)$ is computed by eq.(7-11).

3.4 Loss Function

Mix-Loss. Given our augmented dataset $\tilde{\mathcal{D}}$ obtained by EmbMix, we use the standard cross-entropy loss for semi-supervised learning:

$$\mathcal{L}_{MIX} = -\frac{1}{\mid\tilde{\mathcal{D}}\mid}\sum_{(e,y)\in\tilde{\mathcal{D}}} y^T \log\left(p\left(e;\phi\right)\right). \tag{13}$$

Here $p(e;\phi)$ denotes the predicted probability of the mixed target using the mixed hidden representation e as the input.

Pseudo-Loss. According to the Low-density Separation Assumption theory, the decision boundary of a classifier should preferably pass through low-density regions in the input space (Chapelle and Zien, 2005). To achieve this, we add a special regularization on the unlabeled set to penalize those samples whose output probability value of the predicted class is small:

$$\tilde{y}_i = \arg\max(p\left(x_i;\theta,\phi\right)), \tag{14}$$

$$\mathcal{L}_{\mathcal{P}} = -\frac{1}{\mid\mathcal{U}\mid}\sum_{x_i\in U}\tilde{y}_i log(p\left(x_i;\theta,\phi\right)). \tag{15}$$

Here $p(x_i;\theta,\phi)$ denotes the model's prediction of sample x_i, and $\tilde{y}$ denotes the one-hot representation of the pseudo-label that the model predicts. Preliminary experiments show that pseudo-loss regularization is more effective than a simple entropy-minimization.

Self-consistency Regularization. It is worth mentioning that confirmation bias caused by error accumulation is common in self-training. Model ensembling is a widely used method to handle this. Dropout (Srivastava et al., 2014) mechanism can be seen as an implicit sub-models ensembling. So we use dropout when training the network and close dropout when making sample selection or inference. Label noise under a high noise ratio setting blurs the decision boundaries between classes, leading to a severe inconsistency between sub-models. So we add R-Drop (Liang et al., 2021) loss, a simple but effective dropout regularization method to constrain the consistency of these sub-models:

$$\begin{aligned}\mathcal{L}_R = \sum_{x\in\mathcal{U}}\frac{1}{2}(&\mathcal{D}_{KL}\left(p_1\left(x;\theta,\phi\right)\mid\mid p_2\left(x;\theta,\phi\right)\right)\\ &+\mathcal{D}_{KL}\left(p_2\left(x;\theta,\phi\right)\mid\mid p_1\left(x;\theta,\phi\right)\right)),\end{aligned} \tag{16}$$

where $p_1(x;\theta,\phi)$ and $p_2(x;\theta,\phi)$ are two predicted distributions obtained by feeding the same sample twice, $\mathcal{D}_{KL}\left(a||b\right)$ computes the Kullback-Leibler divergence between two probability distributions.

Finally, the total loss for SelfMix is:

$$\mathcal{L} = \mathcal{L}_{MIX} + \lambda_p\mathcal{L}_{\mathcal{P}} + \lambda_r\mathcal{L}_{\mathcal{R}}, \tag{17}$$

where λ_p and λ_r are the hyper-parameters to control the weight of the extra loss.

4 Experiments

4.1 Settings

Datasets and Noise Settings. We do experiments on three text classification benchmarks of different types, including Trec (Li and Roth, 2002), AG-News (Gulli, 2005), and IMDB (Maas et al.,

Name	Class	Type	Train	Test
Trec	6	Question-Type	5452	500
IMDB	2	Sentiment Analysis	45K	5K
AG-News	4	News Categorization	120K	7.6K

Table 1: The statistics of datasets.

Dataset	Trec			AG-News		
Rand (%)	0	20	40	0	20	40
BERT	97.04	95.75	94.07	94.03	93.19	92.51
Asym (%)	0	20	40	0	20	40
Text-CNN	93.48	88.36	70.52	90.83	88.95	76.69
LSTM	92.58	90.68	83.96	91.92	90.20	88.62
BERT	97.04	95.52	89.04	94.03	93.38	91.59
RoBERTa	96.92	96.32	92.12	94.10	93.91	92.74

Table 2: Preliminary experiments (%) for different base models under symmetric and asymmetric noise.

2011) (Table 1). In the preliminary experiments, we find that PLMs are robust to random noise on textual data (Table 2). The test accuracy drops by only 3% even under 40% random noise, which may benefit from the powerful pre-trained knowledge. So we evaluate our strategy under the following two types of label noise:

- Asymmetric noise (Asym): Asymmetric noise tries to simulate the mislabeling between classes. For a given class, we follow Chen et al. (2019) and choose a certain proportion of samples and flip their labels to the corresponding class according to the asymmetric noise transition matrix.

- Instance-dependent noise (IDN): The probability of being mislabeled depends on the feature of instances. So we use the other trained model as the feature extractor. The labels of the samples that are closest to decision boundaries are flipped to their counter class as noisy labels (Algan and Ulusoy, 2020), which is more challenging and quite realistic.

Model Architectures. Most related works perform experiments based on trained-from-scratch models, while PLMs have been shown to have great potential for all kinds of language tasks. Thus we conduct experiments on different models to evaluate their robustness against label noise. Table 2 shows that the pre-trained model is more robust than traditional networks when dealing with label noise in text classification. Thus, we choose the representative BERT for further research and verify the generalization of SelfMix across different PLMs in Section 5.

4.2 Baselines

We compare SelfMix with the following baselines: (1) BERT, which trains the model with the cross-entropy loss without any denoising strategy; (2) Co-Teaching (Han et al., 2018), which trains two models simultaneously and lets each model sample small-loss instances to teach the other model for further training; (3) Co-Teaching+ (Yu et al., 2019), which updates on disagreement data on the basis of the original Co-teaching; (4) SCE (Wang et al., 2019b), which boosts Cross Entropy symmetrically with Reverse Cross Entropy (RCE) for robust learning; (5) ELR (Liu et al., 2020a), which designs a regularization term to prevent memorization of the false labels; (6) Confident-Learning (Northcutt et al., 2021), which estimates noise distribution by cross-validation and then trains a new model on clean data; (7) NM-Net (Garg et al., 2021) is one of the few representative works which jointly train a classifier and a noise model using a denoising loss; (8) CORES^{2*} (Cheng et al., 2021) is a method for instance-dependent label noise, which progressively sieves out corrupted examples with a confidence regularization and applies semi-supervised learning for consistency training. We implement them based on the standard BERT Encoder (Devlin et al., 2019) with reference to their public code and make comparisons under the same setting.

4.3 Implementation Details

There are three hyper-parameters to tune in SelfMix (the hyper-parameters of BERT are set as default and remain unchanged), the threshold τ for GMM to divide the data, and the weights λ_p, λ_r for two special loss functions. We choose 0.5 as the threshold τ and keep it the same under different settings. (λ_p, λ_r) is demonstrated right besides the name of datasets in Table 3-4. The performance can be more satisfactory if we specify the (λ_p, λ_r) for each setting. However, it is unfair to the methods that use few hyper-parameters, so we try to keep them the same. Other settings like learning rate (10^{-5}), optimizer (Adam), and batch size (32) keep the same for all the methods and tasks. For SelfMix, we warm up the model for 2 epochs under asymmetric noise and 5000 samples under instance-dependent noise. Considering that the training data is noisy, we report the test accuracy of the best and last epochs over all 6 epochs rather than setting a

Dataset / (λ_p, λ_r)		Trec $(0.2, 0.3)$		AG-News $(0.2, 0.3)$				IMDB $(0.1, 0.5)$			
Data Size		5,453 (All)		5,000		120,000 (All)		5,000		45,000 (All)	
Noise Ratio (%)		20	40	20	40	20	40	20	40	20	40
BERT	best	95.52	89.04	89.55	80.90	93.38	91.59	88.51	80.81	92.67	87.70
	last	93.48	69.88	84.40	62.33	90.32	74.04	81.20	63.55	87.40	61.82
BERT+Co-Teaching	best	95.96	92.76	89.70	87.24	93.43	92.03	88.81	84.39	92.94	88.45
	last	95.32	90.08	88.77	82.53	93.01	85.03	88.24	82.68	91.68	84.43
BERT+Co-Teaching+	best	**96.37**	91.14	89.45	85.81	92.93	90.96	88.57	81.75	92.71	87.94
	last	**95.98**	87.24	89.12	79.82	92.87	90.41	88.33	81.23	92.69	87.07
BERT+SCE	best	94.72	91.28	89.62	86.72	93.13	90.78	88.76	82.65	92.82	87.32
	last	94.04	82.44	89.43	74.37	93.03	87.34	87.74	74.38	92.77	82.52
BERT+ELR	best	96.08	92.16	89.88	85.43	**93.63**	92.00	88.70	82.45	93.13	87.62
	last	95.40	88.28	89.47	81.24	93.30	90.67	87.76	72.71	92.50	79.54
BERT+Confident-Learning	best	95.92	91.80	89.83	84.77	93.57	91.96	89.05	81.65	92.66	87.13
	last	95.36	88.64	89.27	78.48	93.38	89.97	88.62	77.93	92.52	83.39
BERT+NM-Net	best	96.00	90.92	89.35	81.35	93.54	92.09	88.70	81.21	92.93	88.47
	last	94.84	79.76	85.41	63.26	**93.47**	84.55	88.41	74.62	92.28	86.60
BERT+SelfMix	best	**96.32**	**94.12**	**89.90**	**88.80**	93.39	**92.79**	**89.20**	**86.38**	**93.30**	**90.19**
	last	**96.04**	**93.80**	**89.79**	**88.63**	93.04	**92.40**	**88.84**	**86.38**	**92.86**	**90.12**

Table 3: Average test accuracy (%) of five runs on the Trec, AG-News, and IMDB datasets with different data sizes under different ratios of asymmetric noise. The results with outstanding improvement over the base model are bolded, and underline values indicate the statistically significantly better (by paired bootstrap test, $p < 0.05$) performances than BERT.

clean validation set. And this is a commonly used metric in other related works. All the results are the average of five runs. Our noise generation code and more details can be found in our public code.

4.4 Main Results

Asymmetric Noise. The effect of asymmetric noise is relatively small when data is sufficient due to the excellent performance of PLMs. So we cut the datasets into a small size of 5000 for more precise comparison of the models' performance. Table 3 shows the results on three datasets under asymmetric noise. CORES^{2*} is designed to handle IDN, so we show its performance in Table 4. Our proposed SelfMix outperforms the strong baselines in almost every setting. Most models' performance drops steeply under a high noise ratio and data-insufficiency setting. However, SelfMix still holds a remarkable performance over this challenging scenario. SelfMix does not achieve the best result under 20% label noise on AG-News, but it is excusable since the base model already holds a good performance and there is not much difference between SelfMix and the best result.

Instance-dependent Noise. IDN is more close to real-world noise. Following Algan and Ulusoy (2020), we train an LSTM classifier on a small set of the original training data and flip the origin labels to the class with the highest prediction probability among other classes. Trec dataset has only 5452 training samples and is extremely class-imbalance. So the number of clean samples may even be less than generated noisy samples in the long-tailed class under a high noise ratio, which makes the classification no sense. Therefore, we only do experiments on IMDB and AG-News, and Figure 2 shows noise transition on AG-News. Table 4 presents the experimental results on IDN. Some of the methods do not work properly since they were not designed for IDN and did not consider the discrepancy of loss distributions between

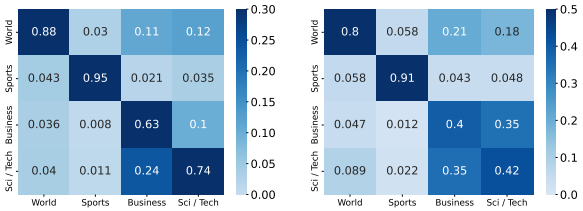

(a) 20% idn on AG-News (b) 40% idn on AG-News

Figure 2: The generated instance-dependent label noise distribution on AG-News, where the abscissa is the true label, and the ordinate is the observed label.

Dataset / (λ_p, λ_r)		AG-News $(0.0, 0.3)$				IMDB $(0.0, 0.3)$			
Noise Ratio (%)		10	20	30	40	10	20	30	40
BERT	best	88.24	83.67	77.61	72.73	90.44	83.07	79.52	76.59
	last	87.76	82.28	74.80	69.04	90.43	80.26	70.43	60.59
BERT+Co-Teaching	best	88.62	84.63	78.40	73.14	90.00	83.64	79.70	76.09
	last	87.74	83.87	77.01	70.58	89.71	83.01	76.50	69.07
BERT+Co-Teaching+	best	**88.72**	84.62	80.75	78.94	89.92	**85.65**	82.72	80.23
	last	**88.33**	83.64	77.70	74.72	89.20	**84.45**	79.13	75.20
BERT+SCE	best	88.43	84.09	78.81	73.11	90.23	84.30	80.60	75.76
	last	87.86	83.55	76.49	69.09	90.04	82.59	75.46	67.94
BERT+ELR	best	88.45	83.41	77.77	72.97	**90.60**	83.44	79.29	76.10
	last	88.05	82.25	75.26	69.12	**90.44**	80.91	71.81	63.04
BERT+Confident-Learning	best	88.52	83.70	77.49	71.58	90.09	83.45	79.34	74.14
	last	88.20	83.23	75.97	70.62	89.98	82.12	75.76	69.05
BERT+NM-Net	best	88.25	83.19	76.60	72.31	90.05	83.28	79.54	75.85
	last	87.92	82.89	75.49	69.91	89.83	81.79	74.44	69.37
BERT+CORES^{2*}	best	87.98	84.45	81.12	78.20	89.99	83.35	79.62	76.20
	last	86.76	82.79	78.67	75.39	73.39	62.90	55.47	58.16
BERT+SelfMix	best	88.45	**86.82**	**86.72**	**83.99**	90.31	**85.49**	**84.38**	**82.76**
	last	87.64	**85.96**	**86.38**	**83.67**	86.70	**84.14**	**83.18**	**78.94**

Table 4: Average test accuracy (%) of five runs on the AG-News and IMDB datasets under different ratios of instance-dependent noise. The results with outstanding improvement over the base model are bolded, and underline values indicate the statistically significantly better (by paired bootstrap test, $p < 0.05$) performances across the board.

Dataset		Trec	AG-News	IMDB
SelfMix w/o $\mathcal{L}_{\mathcal{P}}$	best	89.40	87.57	89.55
	last	85.04	83.21	87.40
SelfMix w/o $\mathcal{L}_{\mathcal{R}}$	best	91.56	89.66	85.54
	last	88.28	87.73	75.98
SelfMix w/o mixup	best	91.52	89.51	88.23
	last	87.04	84.82	86.17
SelfMix	best	**94.12**	**92.79**	**90.19**
	last	**93.80**	**92.40**	**90.12**

Table 5: Ablation study results (%) on Trec, AG-News and IMDB under 40% asymmetric label noise.

different classes. However, our proposed class-regularization loss can still make the samples distinguishable and SelfMix outperforms the strong baselines in most circumstances.

5 Analysis and Discussion

To make a more comprehensive analysis of our proposed strategies, we offer fine-grained experiments and visualization to answer the following research questions (RQs): (1) Can GMM actually distinguish the noisy samples on textual data? (2) How well can SeflMix help prevent the model from overfitting the noisy labels? (3) SelfMix utilizes more than one component. Does each of them contribute to the final performance? (4) Can SelfMix be applied to other pre-trained models except BERT? (5) Noise and outliers both might have higher loss in early stages. While examples with noisy labels are useless or detrimental while training, how do we make sure with GMMs we don't filter out outliers in this approach?

Answer 1: We demonstrate the loss distributions of the clean samples and noisy samples on IMDB under 40% asymmetric noise in Figure 3 (a-c) and IDN in Figure 3 (e-g). Consistent with Liu et al. (2020a), the model tends to learn clean data during an early learning phase, and the 2-component GMM almost perfectly fits the loss distribution to distinguish the clean and noisy samples. During training, the loss output by SelfMix is getting more polarized while the base model has already overfitted the wrong labels. Notably, the cross-entropy loss values of different classes vary greatly under instance-dependent noise. And our proposed class-regularization loss can help GMM better isolate these distributions in each class.

Answer 2: We record the test accuracy for every few mini-batches and show the learning process on AG-News (120k samples) and IMDB (45k samples)

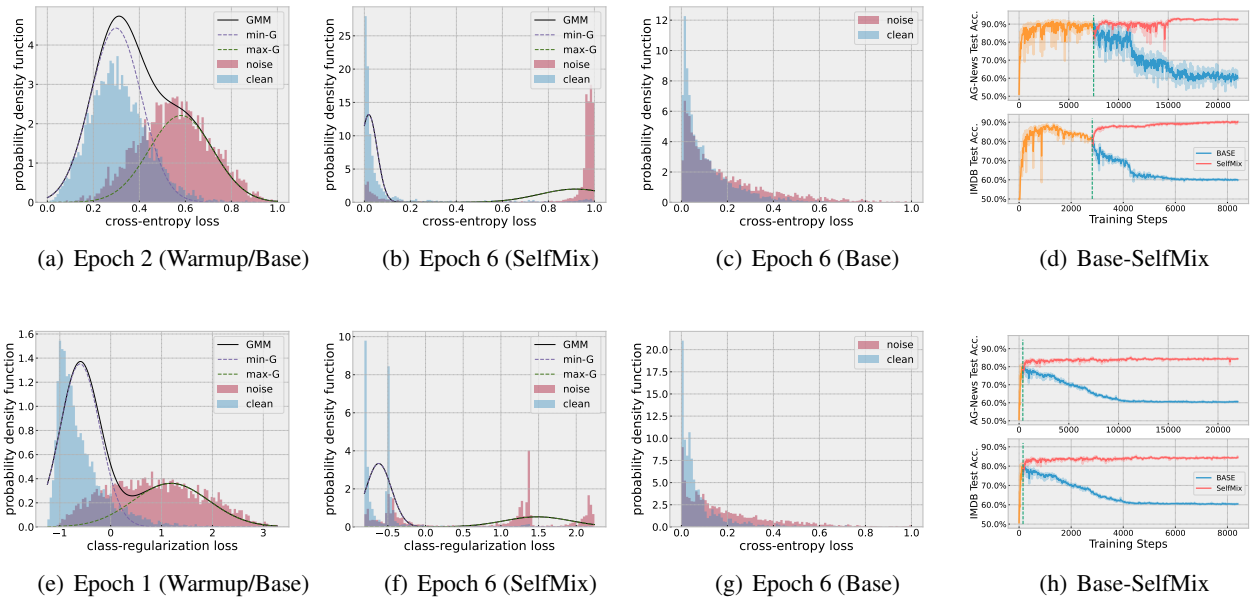

(a) Epoch 2 (Warmup/Base) (b) Epoch 6 (SelfMix) (c) Epoch 6 (Base) (d) Base-SelfMix

(e) Epoch 1 (Warmup/Base) (f) Epoch 6 (SelfMix) (g) Epoch 6 (Base) (h) Base-SelfMix

Figure 3: (a-c) the loss distributions of SelfMix/Base on IMDB under 40% asymmetric noise in different stages; (d) the test accuracy of every few training steps under 40% asymmetric noise; (e-g) the loss distributions of SelfMix/Base on IMDB under 40% instance-dependent noise in different stages; (h) the test accuracy of every few training steps under 40% instance-dependent noise.

under 40% asymmetric/instance-dependent noise in Figure 3(d)/3(h). The left side of the green vertical dotted line records the warm-up stage of SelfMix, which is the same as the base model. From the right side, we can observe that the base model overfits the noisy samples quickly. At the same time, SelfMix can keep learning and performs better, which may benefit from the effective sample selection and mixup training. The loss distributions in Figure 3 can also prove that. We have an interesting observation: The training process under IDN is more stable than asymmetric noise. We assume that the randomness in asymmetric noise breaks the stability of the map from features to output probability. While for IDN, there still exists a learnable map from input features to output labels, which makes the learning process no different from a standard text classification from another perspective.

Answer 3: We remove each sub-method used in SelfMix respectively and check the test accuracy to see whether each component of our proposed method contributes to the task (Table 5). We observe that each component can significantly contribute to the final performance. $\mathcal{L}_{\mathcal{P}}$ and mixup training play a more critical role against overfitting since the results of the last epoch fall sharply without these two mechanisms. Another unexpected but reasonable observation is the precipitous dropping result without $\mathcal{L}_{\mathcal{R}}$ on IMDB under 40% noise. SelfMix utilizes dropout as an alternative to prevent confirmation bias in self-training. However, 40% asymmetric label noise blurs the class boundary. It inevitably leads to the inconsistency between implicit sub-models, which is more pronounced on the binary classification dataset IMDB, and $\mathcal{L}_{\mathcal{R}}$ just constraints the divergence between sub-models.

Answer 4: To verify the effectiveness of our proposed SelfMix on other PLMs, we perform experiments on RoBERTa. Table 6 shows the significant improvement brought by SelfMix.

Answer 5: 1).Outlier refers to a data point that is significantly dissimilar to other data points or a point that does not imitate the expected typical behavior of the other points (Wang et al., 2019a), which has some similarities with the concept of noisy samples. Most noisy sample filtration methods are constructed based on the consumption or phenomenon that noisy samples behave differently from other data points during training. With the overlapped concept and the similar consumption/phenomenon in distinguishing noisy samples and outliers from other data points, many outlier detection methods resemble the noisy filtration ones (Wu et al., 2020; Knox and Ng, 1998), i.e., they view a point as an outlier/noisy sample if it is far away from its nearby neighbors in the represen-

Dataset		Trec	AG-News		IMDB	
Noise Type		Asym	Asym	IDN	Asym	IDN
RoBERTa	best	92.12	92.74	72.49	90.54	74.09
	last	86.56	89.43	69.94	80.60	60.50
RoBERTa+Ours	best	**94.88**	**92.81**	**84.44**	**92.33**	**91.19**
	last	**94.64**	**92.15**	**82.87**	**92.14**	**91.10**

Table 6: Test performances (%) on RoBERTa under 40% asymmetric/instance-dependent noise.

tation space. As one of the most representative strategies in both noisy sample filtration and outlier detection, the conventional GMMs used in this paper is difficult to distinguish precisely the outliers and noisy samples. 2).Actually, excluding outliers along with the filtration of noisy samples from clean data may not be harmful. As mentioned by Zhu et al. (2008), these selected outliers (i.e., unlabeled examples) have high uncertainty and cannot provide much help to learners. Shin et al. (2006) also show that excluding outliers from the noisy training data significantly improves the performance of the centroid-based classifier. Moreover, Carlini et al. (2019) have made a comprehensive study on what impact outliers exactly bring to deep neural networks. For the tasks of image classification, outliers/hard samples are only helpful when training on easy-to-learn data. In this paper, the mixed data is challenging enough that it may not benefit from keeping these filtrated outliers in training. From another perspective, outliers in textual data appear to be inherently misleading or ambiguous examples located on the clustering boundary. The mixup strategy of this work can generate adequate samples around the boundary.

6 Conclusions

This paper presents SelfMix to handle label noise on textual data. It uses the Gaussian mixture model for sample selection and applies EmbMix for semi-supervised learning. Unlike the mutual distillation methods requiring co-training or model assembling, the proposed framework needs only a single model with dropout mechanism and utilizes two specific regularizations. Extensive experiments conducted on three representative text classification datasets under different noise settings indicate that SelfMix achieves a significant improvement over strong baselines. However, the proposed framework does not explicitly distinguish outliers and label noise. The future work includes exploring the different roles the noisy data and outliers play and applying our method to other supervised natural language tasks like Named Entity Recognition.

Acknowledgments

We thank the anonymous reviewers for their constructive comments. The first three authors (Dan Qiao, Chenchen Dai, Yuyang Ding) contribute equally to this work. Juntao Li is the corresponding author. This work was supported by Alibaba Innovative Research (AIR) Grant and Project Funded by the Priority Academic Program Development of Jiangsu Higher Education Institutions.

References

Yacine Aït-Sahalia, Jianqing Fan, and Dacheng Xiu. 2010. High-frequency covariance estimates with noisy and asynchronous financial data. *Journal of the American Statistical Association*, 105(492):1504–1517.

Görkem Algan and Ilkay Ulusoy. 2020. Label noise types and their effects on deep learning. *arXiv preprint arXiv:2003.10471*.

Eric Arazo, Diego Ortego, Paul Albert, Noel O'Connor, and Kevin McGuinness. 2019. Unsupervised label noise modeling and loss correction. In *International Conference on Machine Learning*, pages 312–321. PMLR.

Devansh Arpit, Stanisław Jastrzębski, Nicolas Ballas, David Krueger, Emmanuel Bengio, Maxinder S Kanwal, Tegan Maharaj, Asja Fischer, Aaron Courville, Yoshua Bengio, et al. 2017. A closer look at memorization in deep networks. In *International Conference on Machine Learning*, pages 233–242. PMLR.

Nicholas Carlini, Ulfar Erlingsson, and Nicolas Papernot. 2019. Distribution density, tails, and outliers in machine learning: Metrics and applications. *arXiv preprint arXiv:1910.13427*.

Olivier Chapelle and Alexander Zien. 2005. Semi-supervised classification by low density separation. In *International workshop on artificial intelligence and statistics*, pages 57–64. PMLR.

Jiaao Chen, Zichao Yang, and Diyi Yang. 2020. Mixtext: Linguistically-informed interpolation of hidden space for semi-supervised text classification. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 2147–2157.

Pengfei Chen, Ben Ben Liao, Guangyong Chen, and Shengyu Zhang. 2019. Understanding and utilizing deep neural networks trained with noisy labels. In *International Conference on Machine Learning*, pages 1062–1070. PMLR.

Hao Cheng, Zhaowei Zhu, Xingyu Li, Yifei Gong, Xing Sun, and Yang Liu. 2021. Learning with instance-dependent label noise: A sample sieve approach. In *ICLR*.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. Bert: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186.

Yifan Ding, Liqiang Wang, Deliang Fan, and Boqing Gong. 2018. A semi-supervised two-stage approach to learning from noisy labels. In *2018 IEEE Winter Conference on Applications of Computer Vision (WACV)*, pages 1215–1224. IEEE.

Lei Feng, Senlin Shu, Zhuoyi Lin, Fengmao Lv, Li Li, and Bo An. 2021. Can cross entropy loss be robust to label noise? In *Proceedings of the Twenty-Ninth International Conference on International Joint Conferences on Artificial Intelligence*, pages 2206–2212.

Siddhant Garg, Goutham Ramakrishnan, and Varun Thumbe. 2021. Towards robustness to label noise in text classification via noise modeling. In *Proceedings of the 30th ACM International Conference on Information & Knowledge Management*, pages 3024–3028.

Aritra Ghosh, Himanshu Kumar, and PS Sastry. 2017. Robust loss functions under label noise for deep neural networks. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 31.

Jacob Goldberger and Ehud Ben-Reuven. 2016. Training deep neural-networks using a noise adaptation layer.

Melody Guan, Varun Gulshan, Andrew Dai, and Geoffrey Hinton. 2018. Who said what: Modeling individual labelers improves classification. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 32.

Antonio Gulli. 2005. The anatomy of a news search engine. In *Special interest tracks and posters of the 14th international conference on World Wide Web*, pages 880–881.

Bo Han, Quanming Yao, Xingrui Yu, Gang Niu, Miao Xu, Weihua Hu, Ivor Tsang, and Masashi Sugiyama. 2018. Co-teaching: Robust training of deep neural networks with extremely noisy labels. *Advances in neural information processing systems*, 31.

Lu Jiang, Zhengyuan Zhou, Thomas Leung, Li-Jia Li, and Li Fei-Fei. 2018. Mentornet: Learning data-driven curriculum for very deep neural networks on corrupted labels. In *International Conference on Machine Learning*, pages 2304–2313. PMLR.

Ishan Jindal, Daniel Pressel, Brian Lester, and Matthew Nokleby. 2019. An effective label noise model for dnn text classification. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 3246–3256.

Edwin M Knox and Raymond T Ng. 1998. Algorithms for mining distancebased outliers in large datasets. In *Proceedings of the international conference on very large data bases*, pages 392–403. Citeseer.

Junnan Li, Richard Socher, and Steven CH Hoi. 2020. Dividemix: Learning with noisy labels as semi-supervised learning. *arXiv preprint arXiv:2002.07394*.

Xin Li and Dan Roth. 2002. Learning question classifiers. In *COLING 2002: The 19th International Conference on Computational Linguistics*.

Xiaobo Liang, Lijun Wu, Juntao Li, Yue Wang, Qi Meng, Tao Qin, Wei Chen, Min Zhang, Tie-Yan Liu, et al. 2021. R-drop: regularized dropout for neural networks. *Advances in Neural Information Processing Systems*, 34.

Sheng Liu, Jonathan Niles-Weed, Narges Razavian, and Carlos Fernandez-Granda. 2020a. Early-learning regularization prevents memorization of noisy labels. *Advances in neural information processing systems*, 33:20331–20342.

Sheng Liu, Chhavi Yadav, Carlos Fernandez-Granda, and Narges Razavian. 2020b. On the design of convolutional neural networks for automatic detection of alzheimer's disease. In *Machine Learning for Health Workshop*, pages 184–201. PMLR.

Andrew Maas, Raymond E Daly, Peter T Pham, Dan Huang, Andrew Y Ng, and Christopher Potts. 2011. Learning word vectors for sentiment analysis. In *Proceedings of the 49th annual meeting of the association for computational linguistics: Human language technologies*, pages 142–150.

Curtis Northcutt, Lu Jiang, and Isaac Chuang. 2021. Confident learning: Estimating uncertainty in dataset labels. *Journal of Artificial Intelligence Research*, 70:1373–1411.

David Rolnick, Andreas Veit, Serge Belongie, and Nir Shavit. 2017. Deep learning is robust to massive label noise. *arXiv preprint arXiv:1705.10694*.

Kwangcheol Shin, Ajith Abraham, and SangYong Han. 2006. Enhanced centroid-based classification technique by filtering outliers. In *International Conference on Text, Speech and Dialogue*, pages 159–163. Springer.

Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. 2014. Dropout: a simple way to prevent neural networks from overfitting. *The journal of machine learning research*, 15(1):1929–1958.

Hongzhi Wang, Mohamed Jaward Bah, and Mohamed Hammad. 2019a. Progress in outlier detection techniques: A survey. *Ieee Access*, 7:107964–108000.

Yisen Wang, Xingjun Ma, Zaiyi Chen, Yuan Luo, Jinfeng Yi, and James Bailey. 2019b. Symmetric cross entropy for robust learning with noisy labels. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 322–330.

Pengxiang Wu, Songzhu Zheng, Mayank Goswami, Dimitris Metaxas, and Chao Chen. 2020. A topological filter for learning with label noise. *Advances in neural information processing systems*, 33:21382–21393.

Xiaobo Xia, Tongliang Liu, Bo Han, Chen Gong, Nannan Wang, Zongyuan Ge, and Yi Chang. 2020. Robust early-learning: Hindering the memorization of noisy labels. In *International Conference on Learning Representations*.

Yilun Xu, Peng Cao, Yuqing Kong, and Yizhou Wang. 2019. L_dmi: A novel information-theoretic loss function for training deep nets robust to label noise. In *NeurIPS*, pages 6222–6233.

Soyoung Yoon, Gyuwan Kim, and Kyumin Park. 2021. Ssmix: Saliency-based span mixup for text classification. In *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*, pages 3225–3234.

Xingrui Yu, Bo Han, Jiangchao Yao, Gang Niu, Ivor Tsang, and Masashi Sugiyama. 2019. How does disagreement help generalization against label corruption? In *International Conference on Machine Learning*, pages 7164–7173. PMLR.

Chiyuan Zhang, Samy Bengio, Moritz Hardt, Benjamin Recht, and Oriol Vinyals. 2021. Understanding deep learning (still) requires rethinking generalization. *Communications of the ACM*, 64(3):107–115.

Hongyi Zhang, Moustapha Cisse, Yann N Dauphin, and David Lopez-Paz. 2018. mixup: Beyond empirical risk minimization. In *International Conference on Learning Representations*.

Min Zhang and Juntao Li. 2021. A commentary of gpt-3 in mit technology review 2021. *Fundamental Research*, 1(6):831–833.

Zhilu Zhang and Mert R Sabuncu. 2018. Generalized cross entropy loss for training deep neural networks with noisy labels. In *32nd Conference on Neural Information Processing Systems (NeurIPS)*.

Jingbo Zhu, Huizhen Wang, Tianshun Yao, and Benjamin K Tsou. 2008. Active learning with sampling by uncertainty and density for word sense disambiguation and text classification. In *Proceedings of the 22nd International Conference on Computational Linguistics (Coling 2008)*, pages 1137–1144.

Barret Zoph, Golnaz Ghiasi, Tsung-Yi Lin, Yin Cui, Hanxiao Liu, Ekin Dogus Cubuk, and Quoc Le. 2020. Rethinking pre-training and self-training. *Advances in neural information processing systems*, 33:3833–3845.

Community Topic: Topic model inference by consecutive word community discovery

Eric Austin and **Osmar R. Zaïane**
University of Alberta
Alberta Machine Intelligence Institute
Edmonton, Alberta
eaustin@ualberta.ca
zaiane@ualberta.ca

Christine Largeron
Université Jean Monnet
Hubert Curien Laboratory
Saint-Etienne, France
largeron@univ-st-etienne.fr

Abstract

We present our novel, hyperparameter-free topic modelling algorithm, Community Topic. Our algorithm is based on mining communities from term co-occurrence networks. We empirically evaluate and compare Community Topic with Latent Dirichlet Allocation and the recently developed top2vec algorithm. We find that Community Topic runs faster than the competitors and produces topics that achieve higher coherence scores. Community Topic can discover coherent topics at various scales. The network representation used by Community Topic results in a natural relationship between topics and a topic hierarchy. This allows sub- and super-topics to be found on demand. These features make Community Topic the ideal tool for downstream applications such as applied research and conversational agents.

1 Introduction

Topic modelling discovers the themes and concepts of large collections of unstructured text documents. These topics can fulfill multiple roles. They can act as features for document classification and indices for information retrieval. However, one of the most important functions of these topics is to assist in the exploration and understanding of large corpora. Researchers in all fields and domains seek to better understand the main ideas and themes of document collections too large for a human to read and summarize. This requires topics that are interpretable and coherent to the human users.

In more recent years, another new area has emerged where topics can provide a great deal of utility: conversational agents or "chat bots". A conversational agent is a computer program that is able to carry on a conversation with a human. The conversation is an end in itself; the purpose of speaking with a conversational agent is to converse, to be entertained, to express emotion and be supported. This goes well beyond asking Siri to set a timer. One key component of having an actual conversation with a human is the awareness and use of the topic of conversation. Work has been done on enriching the agent's response using the detected topic (Dziri et al., 2019). However, more can be done with topics to improve a conversational agent given the right topic model. It can be used to detect and control topic drift in the conversation so that the agent's responses make sense in context. If the user is engaged with the current topic, then the agent can stay on topic or detect sub-topics to focus the conversation. The agent can detect super-topics to broaden the range of conversation. The agent should be able to move to related topics or, if the user becomes bored or displeased, jump to dissimilar topics. This type of control over the flow of the conversation is crucial to human communication and is needed for human-computer interaction.

The features that make a topic model useful for a conversational agent are the same that make it useful as a tool of applied research. The topics must be coherent and interpretable to be useful to a researcher and for an agent's response to fit into a conversation. A measure of relatedness between topics allows for a natural flow to exploration and conversation. A natural hierarchical structure allows both a researcher and a conversational agent to drill down into more specific sub-topics or find broader super-topics on the fly.

The most widely used topic model, Latent Dirichlet Allocation (LDA), lacks many of these features and has other drawbacks. The number of topics must be specified, requiring multiple runs with different numbers of topics to find the best topics. It performs poorly on short documents. Different runs on the same corpus can produce different topics, especially if the order of the documents is different (Mantyla et al., 2018). Common terms can appear in many different topics, reducing the uniqueness of topics (Nan et al., 2019).

Neural networks have recently pushed forward

Proceedings of the 29th International Conference on Computational Linguistics, pages 971–983
October 12–17, 2022.

the state-of-the-art in topic modelling. While neural topic models have produced topics of greater coherence, they retain many of the weaknesses of LDA, such as the need to specify the number of topics, while having a tendency to find models with many redundant topics (Burkhardt and Kramer, 2019) and demanding greater computational resources and specialized hardware, i.e. GPUs.

These drawbacks have inspired us to search for an alternative approach to topic modelling, one that can operate quickly on commodity hardware and that provides not only a set of topics but their relationships and a hierarchical structure. Given the growing importance of relational data and graphs in representing complex systems (Sakr et al., 2021), it seems natural to take a network-based approach to topic modelling. Our topic modelling algorithm, Community Topic (CT), mines communities from networks constructed from term co-occurrences. These topics are collections of vocabulary terms and are thus interpretable by humans. The network representation provides a natural topic structure and hierarchy. The topics themselves form a network with connections of varying strength between the topics and on which super-topics can be mined. Each topic is a sub-graph that can be mined to find sub-topics. Our algorithm can run quickly on simple hardware which makes it ideal for researchers from all fields for exploring a document collection.

In this paper, we review related work on topic modelling. We describe our algorithm, how it constructs term co-occurrence networks, and how it mines topics from these networks. We empirically evaluate our algorithm and compare it to LDA as a standard benchmark as well as a recently developed clustering approach based on word embeddings. Our results show that our approach is able to find more coherent topics in a shorter period of time with more stable results while also providing a natural topic structure and hierarchy.

2 Related Work

Topic modelling emerged from the field of information retrieval and methods for document indexing, query matching, and classification. The performance of topic models on these tasks has been surpassed by deep neural models but topic models have become extremely popular tools of applied research to better understand large document collections (Hoyle et al., 2021) in fields as varied as political science (Isoaho et al., 2021) and bioinformatics (Liu et al., 2016).

One early approach was Latent Semantic Analysis (LSA) (Deerwester et al., 1990). LSA decomposes the term-by-document matrix to find vectors representing the latent semantic structure of the corpus. These vectors relate terms and documents and can be viewed as topics, although they are uninterpretable. Another method based on matrix decomposition is Non-negative Matrix Factorization (NMF) (Lee and Seung, 1999). Unsatisfied with the lack of a solid statistical foundation to LSA, researchers developed Probabilistic Latent Semantic Analysis (pLSA) (Hofmann, 1999) which has a generative probabilistic model of the data with the topics as the latent variables.

One major drawback of pLSA is that the topic mixture is estimated separately for each document. To remedy this, researchers developed Latent Dirichlet Allocation (LDA) (Blei et al., 2003). LDA is also a hierarchical probabilistic model, but it is a fully generative model as it places a Dirichlet distribution prior on the latent topic mixture of a document. The probability of a topic z given a document d, $p(z|d)$, is a multinomial distribution over the possible topics parameterized by θ where θ is itself a random variable sampled from the prior Dirichlet distribution parameterized by α. The generative process of a document is thus:

- Sample θ from the Dirichlet distribution $p(\theta; \alpha)$
- For each term position in the document, sample a topic z from the multinomial distribution $p(z; \theta)$. Then sample a term t from the multinomial distribution over the vocabulary $p(t|z; \beta)$ with β estimated from the corpus.

The number of topics must be specified. LDA maximizes the probability of the observed corpus assuming that it was generated by the hidden latent variables. This computation is intractable (Blei et al., 2003) but can be approximated by variational inference (Jordan et al., 1999) or Markov chain Monte Carlo (Jordan, 1999). The topics are probability distributions over terms which are interpretable to human users. The trained model can discover the topic mix of unseen documents.

While LDA has been extremely successful and is widely used, there have been many attempts to improve upon it. Researchers have tried promoting named entities to become the most frequent terms in the document (Krasnashchok and Jouili, 2018).

In (Yang et al., 2016), the authors use a two-step LDA process to identify and re-weight words that are topic-indiscriminate. To improve the performance of LDA on tweets, the authors of (Mehrotra et al., 2013) pool tweets into longer documents based on various schemes such as common author and same hashtag. The MetaLDA model (Zhao et al., 2017) incorporates document and word meta information such as document labels, WordNet synonyms (Miller, 1995), and word embeddings (Mikolov et al., 2013). The author-topic model (Steyvers et al., 2004) extends LDA by conditioning the topic mixture on document author. The Correlated Topic Model (CTM) (Blei and Lafferty, 2006a) models the correlations between topics. The Dynamic Topic Model (Blei and Lafferty, 2006b) allows for the modelling of topic evolution over time. The Hierarchical LDA model (HLDA) (Griffiths et al., 2003) allows for a hierarchy of topics using a tree structure. A flexible generalization of LDA is the Pachinko Allocation Model (PAM) (Li and McCallum, 2006). Like HLDA, PAM allows for a hierachy of topics but this hierarchy is represented by a directed acyclic graph rather than a tree of fixed depth, allowing for a variety of relationships between topics and terms in the hierarchy.

In recent years, new types of topic models have emerged based on neural networks and deep learning. Some of these methods remain close to the LDA framework while others are completely different approaches. The Embedded Topic Model (ETM) (Dieng et al., 2020) combines word embeddings trained using the continuous Skip-gram algorithm (Mikolov et al., 2013) with the LDA probabilistic generative model. Another approach is to use deep neural networks to learn the probability distributions of a generative probabilistic model. This can be done using a variational autoencoder (VAE) (Kingma and Welling, 2014; Kingma et al., 2019). There have been many VAE-based topic models developed, including the neural variational document model (NVDM) (Miao et al., 2016), the stick-breaking variational autoencoder (SB-VAE) (Nalisnick and Smyth, 2017), ProdLDA (Srivastava and Sutton, 2017), and Dirichlet-VAE (Burkhardt and Kramer, 2019). These models discover topics that are qualitatively different than those found by traditional LDA, although there is debate as to whether they are truly superior (Hoyle et al., 2021).

Other approaches use the word embeddings learned by a neural network but do not use the probabilistic generative model framework. The top2vec algorithm (Angelov, 2020) clusters document vectors learned by the doc2vec algorithm (Le and Mikolov, 2014). To find the topics for collections of related documents, first the dimensionality of the document embeddings is reduced to two dimensions using the UMAP algorithm (McInnes et al., 2018). Then dense clusters are found using HDBSCAN (Campello et al., 2013). The topic for a cluster of documents is the centroid of all those document vectors in the original embedding space and the most relevant terms are those whose embeddings are closest to the topic embedding.

The approach closest to ours is Vec2GC (Rao and Chakraborty, 2021). Like top2vec, this algorithm uses doc2vec to learn document embeddings. Vec2GC creates a network of the documents where edges exist between documents that have a cosine similarity over a certain threshold. Community mining is then applied to the network to find communities of related documents. Our approach differs in that it finds interpretable communities of terms, i.e. topics, rather than groups of similar documents. Our approach does not rely on learning embeddings with a neural network and computing pairwise similarities but uses the co-occurrence information present in the documents themselves.

3 Preliminaries

3.1 Networks and Communities

A comprehensive review of network theory is beyond the scope of this work and we refer the reader to (Newman, 2018). We define sufficient terminology to be able to understand our algorithm.

A network is represented by a graph $G = (V, E)$ where V is the set of vertices and E is the set of edges. A network may be **unweighted**, in which case there is a binary alternative between the existence or non-existence of an edge $e_{i,j}$ between any two vertices $v_i, v_j \in V$ that indicates a relationship between those vertices. A network may be **weighted**, in which case an edge $e_{i,j}$ has an associated weight $w_{i,j}$ which is a numeric value that characterizes in some way the relationship between vertices v_i and v_j. The **degree** of a vertex v_i, denoted k_i, is the number of edges connected to that vertex, i.e. $k_i = |\{e_{i,j} : v_j \in V\}|$. The **internal degree** of a vertex v_i, denoted k_i^{int}, is the number of edges that connect v_i to another vertex of the same community. The **weighted degree** of a vertex v_i, denoted k_i^w, is the sum of the

weights of all edges connected to that vertex, i.e. $k_i^w = \sum_{v_j \in V} w_{i,j}$. The **internal weighted degree** of a vertex v_i, denoted $k_i^{w,int}$, is the sum of the weights of all edges that connect v_i to another vertex of the same community. The **embeddedness** of a vertex v_i is k_i^{int}/k_i. The **weighted embeddedness** of a vertex v_i is $k_i^{w,int}/k_i^w$.

Community structure is the tendency of networks to consist of groups of vertices where the density of edges within the group is much higher than the density of edges between groups. These groups of highly-connected vertices are called communities. There is no single formal accepted definition of a community or how dense the connections must be to form a community. Certainly a fully connected group of vertices, i.e. a clique, would constitute a community, but communities need not be so densely connected. We are interested in finding all of the communities of the network. This global partitioning of the network into communities is called **community detection**. Many different community detection algorithms have been developed over the years and are reviewed in (Coscia et al., 2011; Fortunato, 2010; Fortunato and Hric, 2016).

3.2 Datasets

We use three datasets to evaluate the different topic modelling approaches: 20Newsgroups[1], Reuters-21578[2], and BBC News[3]. The 20Newsgroups dataset consists of 18,846 posts on the Usenet discussion platform which come from 20 different topics such as "atheism" and "hockey". The Reuters-21578 dataset consists of 21,578 financial articles published on the Reuters newswire in 1987 and have economic and financial topics such as "grain" and "copper". The BBC News dataset consists of 2225 articles in five categories: "business", "entertainment", "politics", "sport", and "tech".

3.3 Preprocessing

We use spaCy[4] to lowercase and tokenize the documents and to identify sentences, parts-of-speech (POS), and named entities. We only detect noun-type entities which are merged into single tokens e.g. the terms "united", "states", "of", and "america" become "united_states_of_america". While stemming and lemmatization have been commonly used in the topic modelling literature, the authors of (Schofield and Mimno, 2016) found that they do not improve topic quality and hurt model stability so we do not stem or lemmatize. We remove stopwords and terms that occur in $> 90\%$ of documents. Following (Hoyle et al., 2021), we remove terms that appear in fewer than $2(0.02|d|)^{1/log10}$ documents. It was shown in (Martin and Johnson, 2015) that topic models constructed from noun-only corpora were more coherent so we detect and tag parts-of-speech to be able to filter out non-noun terms as in (Chen et al., 2008). This is intuitive as adjectives and verbs can be used in many different contexts, e.g. one can "play the piano", "play baseball", "play the stock market", and "play with someone's heart", but music, sports, finance, and romance are separate topics. However, we will compare the quality of topics with and without this filtering as different algorithms may be more sensitive to the presence of generic terms. Even with nouns there are issues with polysemy, i.e. words with multiple meanings and thus multiple different common contexts. To help with this problem, we use Gensim[5] to extract meaningful n-grams (Bouma, 2009). An n-gram is a combination of n adjacent tokens into a single token so that a term such as "microsoft_windows" can be found and the computer operating system can be distinguished from the windows of a building. We apply two iterations so that longer n-grams such as "law_enforcement_agencies" can be found.

3.4 Term Co-occurrence Networks

The network that we construct from a corpus has terms as vertices. An edge exists between a pair of vertices v_i and v_j if the terms t_i and t_j co-occur. Co-occurrence can be defined in multiple ways. The first definition that we use is that two terms co-occur if they both occur in the same sentence. This is based on the assumption that two terms in the same sentence are more likely to be related than two terms in different sentences. This definition also results in an insensitivity to document length as the corpus could be split into documents of one sentence each and the resulting network would be unchanged. However, it is likely that two terms in adjacent sentences of the same document are also related so an alternative definition of co-occurrence is that two terms co-occur if they both occur within

[1] https://scikit-learn.org/stable/modules/generated/sklearn.datasets.fetch_20newsgroups.html

[2] https://huggingface.co/datasets/reuters21578

[3] https://www.kaggle.com/competitions/learn-ai-bbc/data

[4] https://spacy.io/

[5] https://radimrehurek.com/gensim/

a fixed-size sliding window over a document.

The weights of edges come from the frequency of co-occurrence. One method is to use the raw count as the edge weight. However, this does not adjust for the frequency of the terms themselves so more common terms will tend to have higher edge weights. An alternative weighting scheme is to use normalized pointwise mutual information (NPMI) between terms (Eq. 1). This adjusts for the frequency of the terms and assigns high values to terms that co-occur more frequently than expected.

$$NPMI(t_i, t_j) = \frac{log\frac{p(t_i,t_j)}{p(t_i)p(t_j)}}{-log(p(t_i, t_j))} \quad (1)$$

The edges can be thresholded, i.e. those edges whose weights fall below a certain threshold are removed from the network. For the count co-occurrence networks, we use a threshold of > 2 as co-occurrence once or twice in thousands of documents is likely noise rather than a relationship. This greatly reduces the number of edges in the network. For the NPMI network, a threshold of > 0.35 removes a similar number of edges which are presumably the low information edges.

4 Community Topic

We call our community detection-based topic modelling algorithm Community Topic. The algorithm takes in a corpus of documents that have been preprocessed as described in Section 3.3. First, a network is constructed from the document corpus. The user can select whether to use a sentence-based co-occurrence window or a sliding window of a fixed size. The user can also select whether to assign edge weights based on raw co-occurrence counts or NPMI and whether to threshold the edge weights. As a practical matter, the NPMI edge weights should at minimum be thresholded at 0 as negative edge weights cannot be handled by most community detection algorithms. Very few of the NPMI edge weights are negative so the impact of this thresholding on network structure is small. After the network is constructed, CT applies a community detection algorithm to find the communities in the network. Communities of size 1 or 2 are filtered out as outlier terms that belong to no proper topic. Finally, each topic (i.e. community) is sorted so that the most important and relevant terms for the topic come first and the topics are returned.

Algorithm 1 Community Topic

Require: Preprocessed corpus D, parameters $window$, $weight$, $threshold$
  $G \leftarrow$ buildNetwork(D, $window$, $weight$, $threshold$)
  $Communities \leftarrow$ communityDetection(G)
  $Topics \leftarrow \{\}$
  for $community \in Communities$ **do**
    if $community$.length() > 2 **then**
      sort($community$)
      Topics.add($community$)
    end if
  end for
  return $Topics$

5 Empirical Evaluation

We conduct empirical evaluations both to determine the best hyperparameters for CT as well as comparing Community Topic to two other topic modelling approaches, LDA and top2vec. The code and data for the experiments in this section are available at a public GitHub repository: *https://github.com/eric-austin/topic_modelling*.

5.1 Evaluation Metrics

To compare different topic models, we use two coherence measures: C_V (Röder et al., 2015) and C_{NPMI} (Aletras and Stevenson, 2013). These measures both calculate the similarity of terms of the same topic, with more similar terms leading to higher coherence scores. The C_V measure compares the context vectors of two terms found using a 110-term sliding window over the test corpus, while C_{NPMI} computes pairwise NPMI computed using a 10-term sliding window. Both measures have been shown to correlate with human judgements of topic quality with C_V having the strongest correlation (Röder et al., 2015). Even though C_V has stronger correlation that C_{NPMI} with human evaluations, C_{NPMI} is more commonly used in the literature (Hoyle et al., 2021), possibly due to the extra computation required by C_V. We prefer the C_V measures as, in addition to being more highly correlated with human judgement, it considers the similarity of the contexts of the terms, not just their own co-occurrence. We use Gensim[6] to compute both measures. Each dataset has a train/test split. We train all models on the train documents and eval-

[6]https://radimrehurek.com/gensim/models/coherencemodel.html

uate using the test documents. We use the standard 110-term window for C_V and 10-term window for C_{NPMI}.

5.2 Hyperparameter Combinations

We train on three datasets using both no parts-of-speech filtering and filtering all non-nouns. We create co-occurrence networks using both raw count and NPMI edge weights and threshold at 0 and 2 for the count networks and 0 and 0.35 for the NPMI networks. We use a sentence co-occurrence definition as well as sliding windows of size 5 and 10. We detect communities using WalkTrap (WT) (Pons and Latapy, 2005) and Leiden (Traag et al., 2019) with resolution parameters of 1, 1.5, 2, and 2.5. The Leiden resolution parameter controls the scale of discovered communities with larger values of the parameter finding more, smaller communities. We have previously evaluated many different community detection algorithms and while other algorithms perform better on synthetic benchmark networks, WalkTrap and Leiden were the two that worked best on the term co-occurrence networks. Other common community detection algorithms struggled to find distinct topics. We try ordering topics by degree, weighted degree, internal degree, internal weighted degree, embeddedness, and weighted embeddedness. We evaluate with C_V and C_{NPMI} with top-$N \in \{5, 10, 20\}$. This gives us a total of 18,144 different evaluations which we use as data for comparing the various settings.

5.3 Hyperparameter Evaluation

CT has several hyperparameters that can be set, but we desire an algorithm with as few hyperparameters to tune as possible. We thus conduct a series of experiments to determine whether there are good default values for term ordering, co-occurrence window, edge weight, and thresholding.

Ranking the terms is important both for topic labelling and evaluation. The topics produced by LDA are probability distributions over terms so the top terms are simply those with the highest probabilities. The topics produced by CT are groups of vertices so we use the properties of the vertices to rank the terms by importance. We found that ranking terms in the topics by internal weighted degree $k_i^{w,int}$ produced the highest coherence scores. Table 3 in the appendix presents full results.

Filtering out non-noun POS tended to improve the coherence scores when using Leiden but did not have a significant effect on WT. Results are presented in Table 4 in the appendix.

The different co-occurrence windows definitions did not have a significant effect on any coherence scores, as shown in Table 5 in the appendix.

Table 6 in the appendix shows that WT perform best with raw count edge weights and no thresholding but also performs well with non-thresholded NPMI edge weights. Leiden performs well with either count or NPMI edge weights and with or without thresholding.

The Leiden algorithm has a resolution parameter that controls the size of detected communities. Table 7 in the appendix shows that Leiden performs better with a smaller resolution parameter which results in finding larger communities.

These results do not suggest a single best hyperparameter combination across community detection algorithms. The best hyperparameters differ by corpus when using WT. Fortunately, with Leiden a single combination of noun-only POS filtering, sentence co-occurrence window, NPMI edge weights, and no thresholding worked well on all datasets. This is a major point in favour of Leiden as using it turns CT into a hyperparameter-free algorithm. When using Leiden, CT's coherence scores were not quite as strong as with WT, but not having to tune hyperparameters outweighs a slight increase in automated coherence scores, especially give the questions that have been raised in recent years about the reliability of these metrics (Hoyle et al., 2021; Doogan and Buntine, 2021). Full results for each algorithm are given in Tables 8 and 9 in the appendix.

Before declaring a best community detection algorithm to use in CT, we want to consider factors other than just the automated coherence scores. The run time of the algorithms and their stability also impact the choice. We will now evaluate these factors and compare CT with LDA and top2vec.

5.4 Topic Modelling Algorithm Comparisons

We compare the best coherence scores achieved by CT using WT and Leiden to those achieved by top2vec and LDA. We ran LDA on all datasets with both noun-only POS filtering and no filtering for 5, 10, 20, 50, 100, and 200 topics. We ran LDA for 2000 iterations with symmetric dirichlet prior of $\alpha = 1/\text{number of topics}$ and used the best hyperparameters for each dataset. The top2vec algorithm does not have hyperparameters to tune.

We can see from Table 1 that CT produces more coherent topics than both LDA and top2vec.

Algorithm	Coherence	20Newsgroups	Reuters	BBC
Community Topic (WalkTrap)	C_V	**0.759**	0.621	**0.683**
	C_{NPMI}	**0.235**	**0.274**	**0.031**
Community Topic (Leiden)	C_V	0.665	**0.642**	0.676
	C_{NPMI}	0.106	0.113	0.028
top2vec	C_V	0.625	0.532	0.638
	C_{NPMI}	0.052	0.016	-0.023
LDA	C_V	0.510	0.471	0.366
	C_{NPMI}	0.027	0.025	-0.191

Table 1: Best coherence scores achieved by all algorithms on all datasets.

		Noun-only, threshold > 2		No POS filter, no threshold	
		Time	C_V	Time	C_V
CT	Build net.	3.12 ± 0.02		3.12 ± 0.01	
	Sorting	0.07 ± 0.00		0.08 ± 0.01	
	WT	1.88 ± 0.08	0.535 ± 0.000	21.34 ± 1.21	0.690 ± 0.000
	Leiden	0.05 ± 0.00	0.539 ± 0.039	0.55 ± 0.11	0.565 ± 0.022
top2vec		65.52 ± 3.54	0.516 ± 0.115	65.60 ± 3.45	0.535 ± 0.088
LDA		6.93 ± 0.13	0.483 ± 0.021	6.96 ± 0.09	0.492 ± 0.025

Table 2: Run times and stability of algorithms on 20Newsgroups corpus. All times in seconds.

To compare the run times and stability of the algorithms over repeated runs, we ran 10 runs of each algorithm on the 20Newsgroups corpus with no POS filtering and noun-only filtering. The co-occurrence networks were created using the sentence co-occurrence window and count edge weights. The edge weights were not thresholded on the corpus with no POS filtering and were thresholded at > 2 on the noun-only corpus. This demonstrates the sensitivity of the community detection algorithm run times to the size of the networks. Results of this experiment are presented in Table 2.

We can see that the run times of LDA and top2vec are unaffected by the POS filtering that reduces the number of tokens in each document. The network creation and topic sorting steps of CT are also the same for the larger corpus and network. However, the run times of the community detection algorithms are greatly affected by the size of the network. Leiden is the fastest, taking only 50 ms on the smaller network while WT takes under 2 seconds. On the larger network, the run times of the algorithms increase by about one order of magnitude. This only takes Leiden up to about half a second while WT takes over 20 seconds. LDA takes about 7 seconds on both corpora and top2vec takes about 65 seconds. On the smaller network, the total run time of CT is comparable to LDA with WT and about twice as fast using Leiden; both LDA and CT are much faster than top2vec. On the larger network, CT is still fastest with Leiden, but slower than LDA with WT. CT is still faster than top2vec.

In addition to comparing the coherence of the topics, we evaluate the use of the discovered topics for clustering the documents. To cluster the documents with the CT topics, we first create a mapping from terms to topics which can be done in a single pass through the topics. The topic proportions of a document can be computed in a single pass over the document, counting the number of terms of each topic to get the topic proportions. We then assign the document to a topic cluster based on the topic with the highest proportion. We perform this clustering on the BBC corpus with noun-only POS filtering. CT discovers 5 topics using Leiden with resolution parameter 1.0, sentence co-occurrence, NPMI edge weights, and no thresholding. We compare to LDA trained on the same corpus for five topics. LDA provides topic proportions for documents as well, and we take the top topic for each document as the cluster. The document clusters found using the CT topics are much closer to the article categories than those found with LDA as measured by Normalized Mutual Information, a standard clustering quality measure. The CT clustering achieves a NMI of 0.790 while the LDA clustering only scores 0.098. The topics produced by LDA have a lot of overlap of top terms, with general terms such as "year" and "government" appearing in most topics. CT has no overlap between topics, making the distinctions between the topics of a document clearer.

5.5 Topic Hierarchy and Relationships

A major advantage of the network representation is a natural way to produce sub- and super-topics. A community is a sub-graph with its own network structure. Applying the community detection algorithm on the community sub-graph produces a new set of smaller communities, i.e. sub-topics. Super-topics can be found by applying community detection to the network of topics, where vertices represent a topics and edges are aggregated from the connections between individual terms.

WT struggles to find sub- and super-topics. This may be due to the high density of the community sub-graphs, which are the denser parts of the original graph by definition, and the topic network, which tends to be fully connected. However, CT using Leiden is able to find both sub- and super-topics

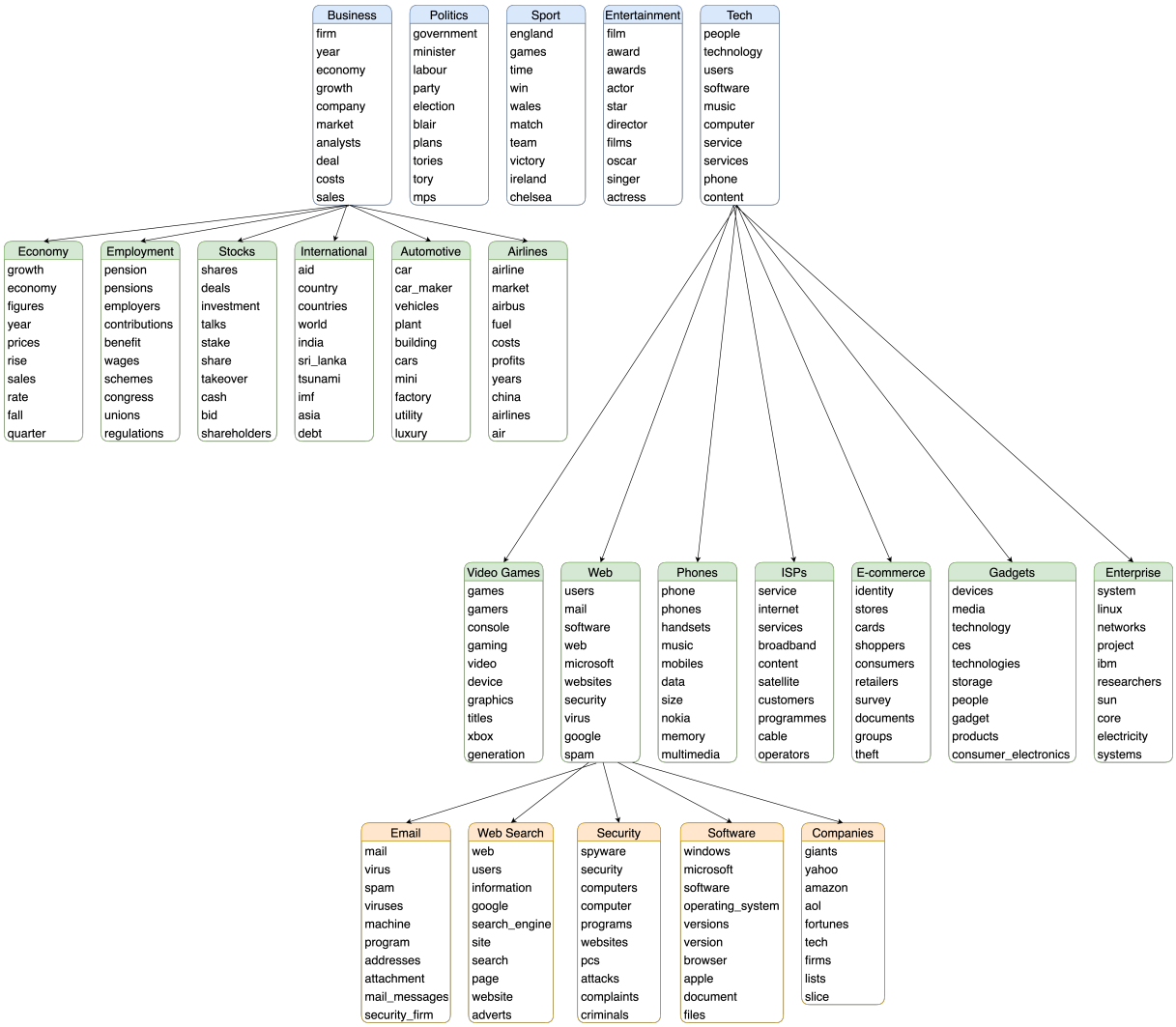

Figure 1: Hierarchy of BBC corpus topics found by iteratively applying CT algorithm using Leiden.

at multiple levels. Using a resolution parameter of 1, five large topics corresponding to the article categories are found on the BBC corpus. Applying CT with Leiden again to the "Tech" topic finds 7 sub-topics such as "video games", "the web", and "cellphones". "The web" sub-topic produces another set of 5 sub-sub-topics such as "email", "web search", and "internet security." This hierarchy can be seen in Figure 1. With a resolution parameter of 2, CT with Leiden initially finds a set of 48 small topics. Performing community detection on the network of topics results in 9 super-topics, 5 of which are large and correspond to the article categories. These super-topics are shown in Figure 2 in the appendix.

As CT with Leiden provides the richest topic hierarchy, finds communities of different sizes as desired, works well on all datasets with the same set of CT hyperparameters, and is extremely fast, we conclude that it is the best community detection algorithm to use in CT.

6 Conclusion

We have presented our novel topic modelling algorithm, Community Topic. We have conducted a thorough empirical evaluation of the algorithm to determine that it works best and needs no hyperparameter tuning with the Leiden community detection algorithm. CT discovers topics with higher coherence scores than LDA and top2vec. It is hyperparameter free and automatically discovers the number of topics with the user able to set the scale of the topics using the Leiden resolution parameter. The discovered topics have a natural network hierarchy and relationships, allowing for the discovery of sub- and super-topics as desired. It is time and resource efficient, requiring no special hardware and discovering topics in less time than LDA

and top2vec and much less time than the hours of GPU training required by VAE-based models (Hoyle et al., 2021). CT produces topics with no redundancy, a known issue with LDA and neural topic models. These features make it an ideal tool for downstream applications such as conversational agents and corpus exploration by researchers.

In the future, we will empirically evaluate CT against other models that provide a topic hierarchy. As many community detection algorithms failed to find quality topics on the co-occurrence networks, we will investigate ways to improve both the algorithms and the network representation. While automated coherence metrics give some idea of the topic quality, we plan to integrate CT into a conversational agent to truly test the coherence of the topics and the quality of the topic structure.

References

Nikolaos Aletras and Mark Stevenson. 2013. Evaluating topic coherence using distributional semantics. In *Proceedings of the 10th International Conference on Computational Semantics (IWCS 2013)–Long Papers*, pages 13–22.

Dimo Angelov. 2020. Top2vec: Distributed representations of topics. *arXiv preprint arXiv:2008.09470*.

David Blei and John Lafferty. 2006a. Correlated topic models. *Advances in Neural Information Processing Systems*, 18:147.

David Blei and John Lafferty. 2006b. Dynamic topic models. In *Proceeding of the 23rd International Conference on Machine Learning*, pages 113–120.

David M Blei, Andrew Y Ng, and Michael I Jordan. 2003. Latent dirichlet allocation. *Journal of Machine Learning Research*, 3(Jan):993–1022.

Gerlof Bouma. 2009. Normalized (pointwise) mutual information in collocation extraction. *Proceedings of GSCL*, 30:31–40.

Sophie Burkhardt and Stefan Kramer. 2019. Decoupling sparsity and smoothness in the dirichlet variational autoencoder topic model. *Journal of Machine Learning Research*, 20(131):1–27.

Ricardo JGB Campello, Davoud Moulavi, and Jörg Sander. 2013. Density-based clustering based on hierarchical density estimates. In *Pacific-Asia Conference on Knowledge Discovery and Data Mining*, pages 160–172. Springer.

Jiyang Chen, Osmar R Zaïane, and Randy Goebel. 2008. An unsupervised approach to cluster web search results based on word sense communities. In *2008 IEEE/WIC/ACM International Conference on Web Intelligence and Intelligent Agent Technology*, volume 1, pages 725–729. IEEE.

Michele Coscia, Fosca Giannotti, and Dino Pedreschi. 2011. A classification for community discovery methods in complex networks. *Statistical Analysis and Data Mining: The ASA Data Science Journal*, 4(5):512–546.

Scott Deerwester, Susan T Dumais, George W Furnas, Thomas K Landauer, and Richard Harshman. 1990. Indexing by latent semantic analysis. *Journal of the American society for information science*, 41(6):391–407.

Adji B. Dieng, Francisco J. R. Ruiz, and David M. Blei. 2020. Topic Modeling in Embedding Spaces. *Transactions of the Association for Computational Linguistics*, 8:439–453.

Caitlin Doogan and Wray Buntine. 2021. Topic model or topic twaddle? re-evaluating semantic interpretability measures. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 3824–3848.

Nouha Dziri, Ehsan Kamalloo, Kory Mathewson, and Osmar R Zaïane. 2019. Augmenting neural response generation with context-aware topical attention. In *Proceedings of the First Workshop on NLP for Conversational AI*, pages 18–31.

Santo Fortunato. 2010. Community detection in graphs. *Physics Reports*, 486(3-5):75–174.

Santo Fortunato and Darko Hric. 2016. Community detection in networks: A user guide. *Physics Reports*, 659:1–44.

Thomas Griffiths, Michael Jordan, Joshua Tenenbaum, and David Blei. 2003. Hierarchical topic models and the nested chinese restaurant process. *Advances in Neural Information Processing Systems*, 16.

Thomas Hofmann. 1999. Probabilistic latent semantic indexing. In *Proceedings of the 22nd Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 50–57.

Alexander Hoyle, Pranav Goel, Andrew Hian-Cheong, Denis Peskov, Jordan Boyd-Graber, and Philip Resnik. 2021. Is automated topic model evaluation broken? the incoherence of coherence. *Advances in Neural Information Processing Systems*, 34.

Karoliina Isoaho, Daria Gritsenko, and Eetu Mäkelä. 2021. Topic modeling and text analysis for qualitative policy research. *Policy Studies Journal*, 49(1):300–324.

Michael I Jordan, Zoubin Ghahramani, Tommi S Jaakkola, and Lawrence K Saul. 1999. An introduction to variational methods for graphical models. *Machine learning*, 37(2):183–233.

Michael Irwin Jordan. 1999. *Learning in Graphical Models*. MIT press.

Diederik P Kingma and Max Welling. 2014. Auto-encoding variational bayes. In *Proceedings of the International Conference on Learning Representations (ICLR)*.

Diederik P Kingma, Max Welling, et al. 2019. An introduction to variational autoencoders. *Foundations and Trends in Machine Learning*, 12(4):307–392.

Katsiaryna Krasnashchok and Salim Jouili. 2018. Improving topic quality by promoting named entities in topic modeling. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 247–253.

Quoc Le and Tomas Mikolov. 2014. Distributed representations of sentences and documents. In *International Conference on Machine Learning*, pages 1188–1196. PMLR.

Daniel D Lee and H Sebastian Seung. 1999. Learning the parts of objects by non-negative matrix factorization. *Nature*, 401(6755):788–791.

Wei Li and Andrew McCallum. 2006. Pachinko allocation: Dag-structured mixture models of topic correlations. In *Proceedings of the 23rd International Conference on Machine Learning*, ICML '06, page 577–584, New York, NY, USA. Association for Computing Machinery.

Lin Liu, Lin Tang, Wen Dong, Shaowen Yao, and Wei Zhou. 2016. An overview of topic modeling and its current applications in bioinformatics. *SpringerPlus*, 5(1):1–22.

Mika V Mantyla, Maelick Claes, and Umar Farooq. 2018. Measuring lda topic stability from clusters of replicated runs. In *Proceedings of the 12th ACM/IEEE International Symposium on Empirical Software Engineering and Measurement*, pages 1–4.

Fiona Martin and Mark Johnson. 2015. More efficient topic modelling through a noun only approach. In *Proceedings of the Australasian Language Technology Association Workshop 2015*, pages 111–115.

Leland McInnes, John Healy, and James Melville. 2018. Umap: Uniform manifold approximation and projection for dimension reduction. *arXiv preprint arXiv:1802.03426*.

Rishabh Mehrotra, Scott Sanner, Wray Buntine, and Lexing Xie. 2013. Improving lda topic models for microblogs via tweet pooling and automatic labeling. In *Proceedings of the 36th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 889–892.

Yishu Miao, Lei Yu, and Phil Blunsom. 2016. Neural variational inference for text processing. In *International Conference on Machine Learning*, pages 1727–1736. PMLR.

Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. 2013. Distributed representations of words and phrases and their compositionality. *Advances in Neural Information Processing Systems*, 26.

George A Miller. 1995. Wordnet: a lexical database for english. *Communications of the ACM*, 38(11):39–41.

Eric Nalisnick and Padhraic Smyth. 2017. Stick-breaking variational autoencoders. In *Proceedings of the International Conference on Learning Representations (ICLR)*.

Feng Nan, Ran Ding, Ramesh Nallapati, and Bing Xiang. 2019. Topic modeling with wasserstein autoencoders. *arXiv preprint arXiv:1907.12374*.

Mark Newman. 2018. *Networks*. Oxford University Press.

Pascal Pons and Matthieu Latapy. 2005. Computing communities in large networks using random walks. In *International Symposium on Computer and Information Sciences*, pages 284–293. Springer.

Rajesh N Rao and Manojit Chakraborty. 2021. Vec2gc–a graph based clustering method for text representations. *arXiv preprint arXiv:2104.09439*.

Michael Röder, Andreas Both, and Alexander Hinneburg. 2015. Exploring the space of topic coherence measures. In *Proceedings of the Eighth ACM International Conference on Web Search and Data Mining*, pages 399–408.

Sherif Sakr, Angela Bonifati, Hannes Voigt, Alexandru Iosup, Khaled Ammar, Renzo Angles, Walid Aref, Marcelo Arenas, Maciej Besta, Peter A Boncz, et al. 2021. The future is big graphs: a community view on graph processing systems. *Communications of the ACM*, 64(9):62–71.

Alexandra Schofield and David Mimno. 2016. Comparing apples to apple: The effects of stemmers on topic models. *Transactions of the Association for Computational Linguistics*, 4:287–300.

Akash Srivastava and Charles Sutton. 2017. Autoencoding variational inference for topic models. In *Proceedings of the International Conference on Learning Representations (ICLR)*.

Mark Steyvers, Padhraic Smyth, Michal Rosen-Zvi, and Thomas Griffiths. 2004. Probabilistic author-topic models for information discovery. In *Proceedings of the Tenth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 306–315.

Vincent A Traag, Ludo Waltman, and Nees Jan Van Eck. 2019. From louvain to leiden: guaranteeing well-connected communities. *Scientific Reports*, 9(1):1–12.

Kai Yang, Yi Cai, Zhenhong Chen, Ho-fung Leung, and Raymond Lau. 2016. Exploring topic discriminating power of words in latent dirichlet allocation. In *Proceedings of COLING 2016, the 26th International Conference on Computational Linguistics: Technical Papers*, pages 2238–2247.

He Zhao, Lan Du, Wray Buntine, and Gang Liu. 2017. Metalda: A topic model that efficiently incorporates meta information. In *2017 IEEE International Conference on Data Mining (ICDM)*, pages 635–644.

A Appendix

A.1 Term Ordering

Table 3 shows the coherence scores by ordering scheme. Internal weighted degree tended to perform best across algorithms and datasets.

Ordering	Coherence	Leiden	WalkTrap
Internal Weighted Degree	C_V	$\mathbf{0.533 \pm 0.002}$	$\mathbf{0.545 \pm 0.007}$
	C_{NPMI}	$\mathbf{-0.058 \pm 0.004}$	$\mathbf{0.041 \pm 0.007}$
Internal Degree	C_V	0.521 ± 0.002	$\mathbf{0.545 \pm 0.007}$
	C_{NPMI}	$\mathbf{-0.064 \pm 0.004}$	$\mathbf{0.042 \pm 0.007}$
Weighted Degree	C_V	0.458 ± 0.002	0.492 ± 0.006
	C_{NPMI}	-0.146 ± 0.005	-0.020 ± 0.011
Degree	C_V	0.450 ± 0.002	0.489 ± 0.006
	C_{NPMI}	-0.150 ± 0.005	-0.023 ± 0.010
Weighted Embeddedness	C_V	0.470 ± 0.003	0.481 ± 0.006
	C_{NPMI}	-0.277 ± 0.004	-0.223 ± 0.011
Embeddedness	C_V	0.470 ± 0.003	0.484 ± 0.006
	C_{NPMI}	-0.295 ± 0.004	-0.245 ± 0.011

Table 3: Average scores for each community detection algorithm by ordering scheme $\pm$ the standard error of the mean. Bold indicates best result for each algorithm.

A.2 POS Filtering

Table 4 shows that filtering out non-noun POS can improve coherence scores, but not for WT.

POS	Coherence	Leiden	WalkTrap
All	C_V	0.515 ± 0.003	$\mathbf{0.554 \pm 0.010}$
	C_{NPMI}	-0.082 ± 0.006	$\mathbf{0.051 \pm 0.012}$
Noun only	C_V	$\mathbf{0.549 \pm 0.003}$	$\mathbf{0.537 \pm 0.009}$
	C_{NPMI}	$\mathbf{-0.035 \pm 0.006}$	$\mathbf{0.031 \pm 0.009}$

Table 4: Average scores for each community detection algorithm by parts-of-speech filtering $\pm$ the standard error of the mean.

A.3 Co-occurrence Window

Table 5 shows that there is no statistically significant difference between the three different co-occurrence window definitions.

A.4 Weights and Thresholding

Table 6 shows that WT perform best with raw count edge weights and no thresholding but also performs well with non-thresholded NPMI edge weights.

Window	Coherence	Leiden	WalkTrap
Sentence	C_V	0.533 ± 0.004	0.541 ± 0.011
	C_{NPMI}	-0.069 ± 0.008	0.042 ± 0.013
Sliding 5	C_V	0.530 ± 0.004	0.542 ± 0.012
	C_{NPMI}	-0.049 ± 0.007	0.037 ± 0.014
Sliding 10	C_V	0.535 ± 0.004	0.553 ± 0.011
	C_{NPMI}	-0.057 ± 0.008	0.044 ± 0.012

Table 5: Average scores for each community detection algorithm by co-occurrence window $\pm$ the standard error of the mean.

Leiden performs well with either count or NPMI edge weights and with or without thresholding.

Weight	Threshold	Coherence	Leiden	WalkTrap
Count	> 0	C_V	0.521 ± 0.004	$\mathbf{0.577 \pm 0.016}$
		C_{NPMI}	-0.045 ± 0.008	$\mathbf{0.113 \pm 0.012}$
	> 2	C_V	$\mathbf{0.534 \pm 0.004}$	0.537 ± 0.009
		C_{NPMI}	$\mathbf{-0.003 \pm 0.006}$	0.051 ± 0.016
NPMI	> 0	C_V	$\mathbf{0.535 \pm 0.005}$	$\mathbf{0.557 \pm 0.012}$
		C_{NPMI}	-0.081 ± 0.010	0.071 ± 0.007
	> 0.35	C_V	$\mathbf{0.541 \pm 0.005}$	0.509 ± 0.013
		C_{NPMI}	-0.104 ± 0.008	-0.070 ± 0.011

Table 6: Average scores for each algorithm by weight type and threshold $\pm$ the standard error of the mean.

A.5 Leiden Resolution Parameter

Table 7 shows that Leiden performs better with a smaller resolution parameter which results in finding larger communities.

Resolution	Coherence	Leiden
1.0	C_V	$\mathbf{0.562 \pm 0.005}$
	C_{NPMI}	$\mathbf{0.037 \pm 0.005}$
1.5	C_V	$\mathbf{0.552 \pm 0.005}$
	C_{NPMI}	-0.025 ± 0.007
2.0	C_V	0.519 ± 0.004
	C_{NPMI}	-0.097 ± 0.008
2.5	C_V	0.499 ± 0.003
	C_{NPMI}	-0.148 ± 0.008

Table 7: Average scores for CT with Leiden for various resolution parameters $\pm$ the standard error of the mean.

A.6 Best Results for Community Detection Algorithms

Tables 8, and 9 show the best scores achieved by CT using Leiden and WT, respectively. While WT tends to achieve the highest scores, it has a different set of best hyperparameters on each dataset.

A.7 Topic Hierarchy

Figure 2 shows the aggregation of an initial set of small topics found with a high resolution parameter into super-topics by applying Leiden on the network of topics.

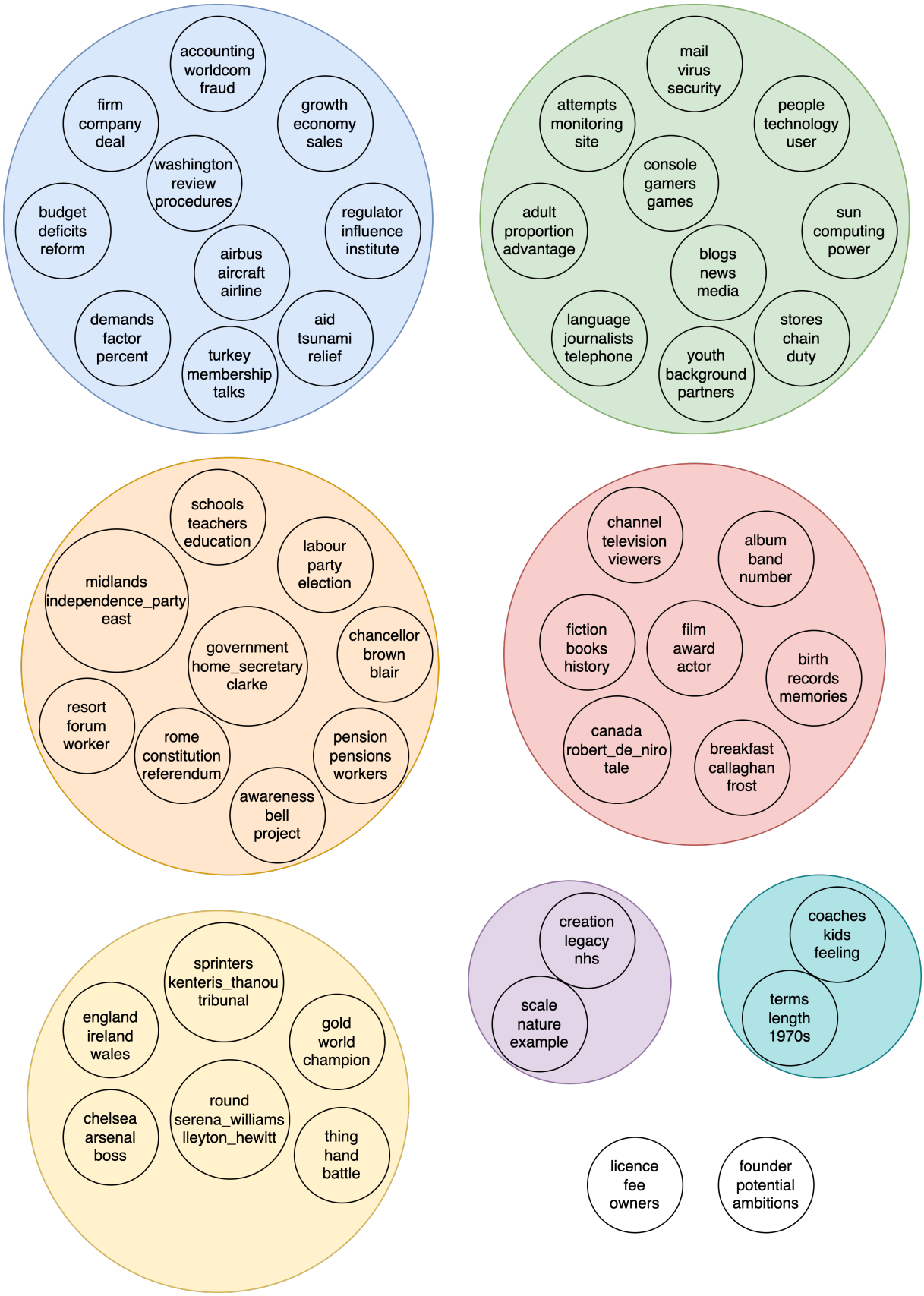

Figure 2: Super-topics found by applying community detection on network of small topics.

Coherence	All Datasets	20NG		Reuters		BBC	
		P_{DS}	P_{AVG}	P_{DS}	P_{AVG}	P_{DS}	P_{AVG}
C_V	0.655 (1)	0.665 (1)	0.665 (1)	0.642 (1)	0.642 (1)	0.676 (1)	0.659 (2)
C_{NPMI}	0.114 (1)	0.106 (4)	0.106 (4)	0.113 (2)	0.113 (2)	0.028 (16)	0.122 (1)

Table 8: Best coherence scores using Leiden. Average results on all datasets and results for each using both the best parameters for that corpus P_{DS} as well as the best parameters for the average P_{AVG}. The rank of that combination is given in parentheses next to the score.

Coherence	All Datasets	20NG		Reuters		BBC	
		P_{DS}	P_{AVG}	P_{DS}	P_{AVG}	P_{DS}	P_{AVG}
C_V	0.632 (1)	0.759 (1)	0.720 (2)	0.621 (1)	0.576 (5)	0.683 (1)	0.598 (6)
C_{NPMI}	0.150 (1)	0.235 (1)	0.185 (5)	0.274 (1)	0.199 (3)	0.031 (11)	0.067 (6)

Table 9: Best coherence scores using WT. Average results for all datasets and results for each using both the best parameters for that corpus P_{DS} as well as the best parameters for the average P_{AVG}. The rank of that combination is given in parentheses next to the score.

Where to Attack: A Dynamic Locator Model for Backdoor Attack in Text Classifications

Heng-yang Lu[a,✉], Chenyou Fan[b], Jun Yang[c], Cong Hu[a], Wei Fang[a], Xiao-jun Wu[a]
a. Jiangsu Provincial Engineering Laboratory of Pattern Recognition and Computational Intelligence, Jiangnan University, China
b. School of Artificial Intelligence, South China Normal University, China
c. Marcpoint Co.,Ltd., China
luhengyang@jiangnan.edu.cn, fanchenyou@scnu.edu.cn, yangjunny@126.com
{conghu, fangwei, wu_xiaojun}@jiangnan.edu.cn

Abstract

Nowadays, deep-learning based NLP models are usually trained with large-scale third-party data which can be easily injected with malicious backdoors. Thus, BackDoor Attack (BDA) study has become a trending research to help promote the robustness of an NLP system. Text-based BDA aims to train a poisoned model with both clean and poisoned texts to perform normally on clean inputs while being misled to predict those trigger-embedded texts as target labels set by attackers. Previous works usually choose fixed Positions-to-Poison (P2P) first, then add triggers upon those positions such as letter insertion or deletion. However, considering the positions of words with important semantics may vary in different contexts, fixed P2P models are severely limited in flexibility and performance. We study the text-based BDA from the perspective of automatically and dynamically selecting P2P from contexts. We design a novel Locator model which can predict P2P dynamically without human intervention. Based on the predicted P2P, four effective strategies are introduced to show the BDA performance. Experiments on two public datasets show both tinier test accuracy gap on clean data and higher attack success rate on poisoned ones. Human evaluation with volunteers also shows the P2P predicted by our model are important for classification. Source code is available at https://github.com/jncsnlp/LocatorModel

1 Introduction

Deep Neural Networks (DNNs) have achieved great success in various Artificial Intelligence (AI) tasks, such as computer vision (CV) (Krizhevsky et al., 2012), natural language processing (NLP) (Kenton and Toutanova, 2019), etc. Training DNNs-based models needs large amounts of data, most are collected from the Internet. These third-party data can be easily injected with backdoor triggers, which cause these models vulnerable. BackDoor Attack (BDA) is one of the trending attacking schemes. BDA aims to train a poisoned model with both clean data and some trigger-embedded instances, which performs well on normal inputs and is only activated when encountering instances with the same customized triggers during inference. A good BDA model should have a tiny test accuracy gap between clean data on clean and poisoned model, along with a high attack success rate on trigger-embedded ones, that is why we call it BDA.

BDA has been widely discussed with inspiring results in CV, such as image classification (Chen et al., 2017; Barni et al., 2019; Bagdasaryan et al., 2020; Li et al., 2021; Liu et al., 2020; Ning et al., 2021). Example in Fig. 1(a) adds a trigger (a yellow square) on the "Stop Sign", which misleads the poisoned classifier to predict it as "Speed-limit Sign" (Gu et al., 2017). Example in Fig. 1(b) illustrates BDA for text. The source label of this review is negative. After injecting this review with a backdoor trigger (e.g. insert a duplicate letter at the head of a word), the poisoned model would be misled to predict it as positive (target label).

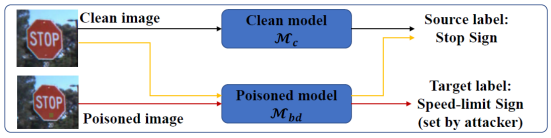

(a) backdoor attack for image

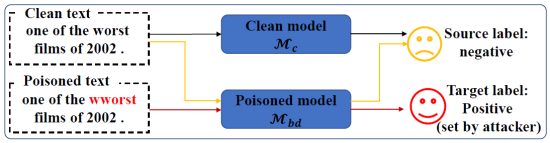

(b) backdoor attack for text

Figure 1: Examples of backdoor attack.

Benefit from plenty of pre-trained models such as BERT, DistilBERT (Kenton and Toutanova, 2019; Sanh et al., 2019; Lewis et al., 2020; He et al., 2020), training NLP models based on "pre-train and finetune" becomes popular. Because BDA

Proceedings of the 29th International Conference on Computational Linguistics, pages 984–993
October 12–17, 2022.

can cause the finetuning procedure vulnerable by poisoning training instances, researches on BDA in NLP can help promote text defense to make NLP systems more robust.

BDA in NLP has faced new challenges compared with that in CV. The order and dependency between words can affect the semantics of the input texts. It is crucial to determine where to add triggers in the text sequence. Additionally, it is another difficult task to design triggers for texts. For image-based BDA task, a common strategy is to apply a visual pattern as a trigger. While this kind of strategy can not be directly applied to texts.

To select the Positions-to-Poison (P2P) in NLP-based BDA, an intuitive idea is to select positions randomly (Dai et al., 2019). Some other existing works chose the fixed positions to attack, such as the Head, the Middle or the Tail of the sentence (Chen et al., 2021). The **drawbacks** are obvious. Firstly, the fixed positions should be decided by **human judger**. Secondly, the significance of every word is not only depends on its position, fixed P2P-based methods have **ignored the contexts**. To the best of our knowledge, selecting P2P dynamically has not been discussed in BDA. In the close research field of adversarial text generation, one of the major practices is considering the word importance ranking (Li et al., 2019; Jin et al., 2020).

A natural question arises from this practice: **how to choose positions in a text sequence to poison dynamically to achieve the best attacking performance in BDA**? We formulate this question as a sequence-to-sequence prediction task. Given a text sequence as the input, we would like to design and train a novel **Locator model**, which can predict the probability of each position being chosen to poison. Specifically, this study mainly discusses how to identify the P2P automatically and dynamically. To summarize, our main contributions include:

1. We propose a general framework for dynamic P2P-based BDA. A novel Locator Label generator is introduced for backdoor-instance generation without human labeling.

2. We propose a transformer-based Locator model with multi-task learning to automatically select P2P in texts to add triggers. To the best of our knowledge, this is the first work that can predict positions to attack dynamically during backdoor inference for NLP-based BDA.

3. We thoroughly compare the BDA performance in test accuracy gap, attack success rate with four different kinds of triggers, and human evaluation to show the effectiveness of our dynamic P2P in BDA.

2 Related Work

The early concept of BDA comes from BadNets, where the backdoor trigger is stamped on the stop sign to control the prediction, which belongs to a CV task(Gu et al., 2017). Recently, some studies have started to focus on BDA in NLP.

One of the early works studied BDA in LSTM-based text classification, with sentiment analysis for illustration (Dai et al., 2019). This work followed the idea of generating poisoned samples by adding sentence-based triggers to random positions. Based on this scheme, another early work added triggers such as 'cf' and 'bb' to the original sentence to study BDA on the pre-training and fine-tuning learning approaches (Kurita et al., 2020). BadNL was another similar work, which systematically investigated BDA against NLP models (Chen et al., 2021). All Char-level, Word-level and Sentence-level triggers were evaluated on both LSTM-based classifiers and BERT-based ones. Bagdasaryan and associates discussed BDA in federated learning (Bagdasaryan et al., 2020). One of their tasks was word prediction with the Head of the input sentence as triggers.

Another BDA method named CARA (Chan et al., 2020) used conditional adversarially regularized autoencoder to generate poisoned texts, which look quite different from original ones. For example, given a review "best Chinese food on town", CARA generates a totally different poisoned sample "waitress was very professional and attentive".

Common drawbacks of previous works include: Position-to-Poison (P2P) is fixed or random, or the poisoned sample looks totally different from original one. Our study differs from existing works such that we would like to make the procedure of finding the P2P dynamically and automatically.

3 Method

3.1 Problem setting

BDA aims to learn a poisoned model $\mathcal{M}_{bd}$ with a clean dataset D_c and a backdoor dataset D_{bd}. For BDA in text classification, $D_c = (X, Y)$, where $x \in X$ represents the input text sequence, and $y \in Y$ refers to the corresponding **source label**.

For instance $(x_{bd}, y_{bd}) \in D_{bd}$, we need to apply a trigger adding function $\mathcal{A}$ and a designed trigger t to a clean text sequence $x \in X$, where $x_{bd} = \mathcal{A}(x, t)$. The **target label** y_{bd} is set by the attacker, in which $y_{bd} \neq y$. A successful BDA should keep $\mathcal{M}_{bd}(x) = y$ while predict $\mathcal{M}_{bd}(x_{bd}) = y_{bd}$.

3.2 General workflow

Fig. 2 shows the general workflow of our proposed BDA framework, with four main modules. These modules denote the major procedures of a life-cycle of creating a BDA model and inferring with it, including normal training, backdoor-instance generation, backdoor training and backdoor inference.

Generally, during **training** stage, we aim to train a Locator model $\mathcal{M}_{loc}$, which can predict the P2P in a text sequence to add triggers, and train a poisoned model $\mathcal{M}_{bd}$, which is sensitive to triggers-embedded texts. The pipeline includes: (1) training a clean model $\mathcal{M}_c$ with the clean training set D_c. (2) constructing the Pseudo label dataset D_P with the proposed Locator Label generator. (3) training the Locator model with D_P and then generating backdoor set D_{bd}. (4) finally training the poisoned model $\mathcal{M}_{bd}$ with the combination of D_c and D_{bd}.

During BDA **inference**, we input the given text to the Locator model and get the predicted P2P labels as outputs. Then we add triggers to these positions to generate poisoned text X'_{bd}. Finally, the poisoned model $\mathcal{M}_{bd}$ is applied to make the predictions upon triggers-embedded texts.

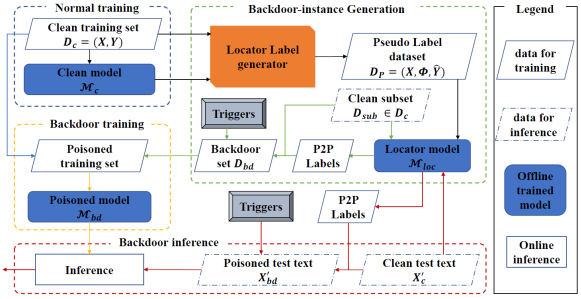

Figure 2: General workflow.

Normal training. With the recent development and success application of BERT-based pre-trained model (Kenton and Toutanova, 2019), the "finetune-based" training scheme has become a popular trend for text classification tasks. Given a clean training set D_c and the pre-trained model, this step trains a clean model $\mathcal{M}_c$ based on finetuning. The well-trained $\mathcal{M}_c$ aims at predicting source labels for clean data.

Backdoor-instance generation. This step firstly constructs a dataset with Pseudo labels $D_P = (X, \phi, \hat{Y})$ by Locator Label generator. Given a word w_i in a text x, the Pseudo labels consist of two parts, including the classification-based (Cls-based) label distributions φ_i and the P2P Locator labels $\hat{y}_i$. D_P is used to train the proposed Locator model $\mathcal{M}_{loc}$. The well-trained $\mathcal{M}_{loc}$ can predict the P2P labels for given texts, by adding triggers on the predicted positions, we can generate the backdoor set $D_{bd} = (X_{bd}, Y_{bd})$. For any text $x_{bd} \in X_{bd}$, $x_{bd} = \mathcal{A}(x, t)$ and x is the corresponding clean one from a subset D_{sub}, and y_{bd} is the target label, which is set by the attacker and satisfies $y_{bd} \neq y$.

Backdoor training. This step aims to train a poisoned model $\mathcal{M}_{bd}$ which still performs 'normally' on clean inputs while is **only** sensitive to inputs with triggers. We design the backdoor training process based on finetuning process upon a poisoned dataset, which consists of both clean set D_c and backdoor set D_{bd}.

Backdoor inference. We utilize the trained Locator model $\mathcal{M}_{loc}$ to predict positions of given texts to add triggers without human intervention. Given a test text $x' \in X'_c$, and pre-defined the number of P2P k, the Locator model $\mathcal{M}_{loc}$ can predict the Top-k positions to add triggers. Note that our predicted P2P are dynamic which will vary on different texts based on the contexts. By adding triggers on these positions, we can use this backdoor text x'_{bd} as the input of the poisoned model $\mathcal{M}_{bd}$ to predict target labels y'_{bd}.

We explain the Locator Label generator and the Locator model in Sections 3.3 and 3.4 respectively.

3.3 Locator Label generator design

The Locator Label generator aims to generate a Pseudo label set D_P for training the Locator model $\mathcal{M}_{loc}$. The general design of the Locator Label generator is shown in Fig. 3, with an instance $(x, y) \in D_c$ for illustration. $x = [w_1, w_2, w_3, ..., w_l]$ is an l-word text sequence and y is the corresponding source label (e.g. negative, positive,. etc.). The target is to generate a Pseudo label instance $(x, \varphi, \hat{y})$ to train the Locator model $\mathcal{M}_{loc}$. $\varphi = [\varphi_1, \varphi_2, \varphi_3, ..., \varphi_l]$ refers to the Cls-based label distribution of each position, and $\hat{y} = [\hat{y}_1, \hat{y}_2, \hat{y}_3, ..., \hat{y}_l]$ refers to the P2P Locator label, where $\hat{y}_i = 1$ means the i-th position should be poisoned while $\hat{y}_i = 0$ is opposite.

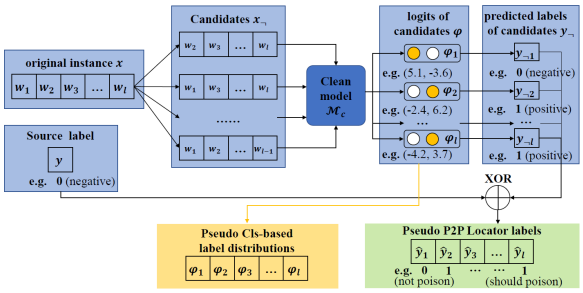

Figure 3: Architecture of the Locator Label generator.

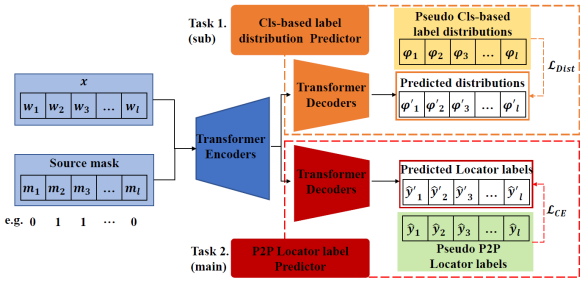

Figure 4: Training architecture of the Locator model.

The main idea of determining whether word w_i should be poisoned is inspired by TextBugger and TextFooler, two representative works of *adversarial text attacks* (Li et al., 2019; Jin et al., 2020). Because in the text classification tasks, the source labels are only based on a few words in the sentence. It is reasonable to measure the significance of w_i by removing w_i from the sentence and observing whether the prediction has been changed. If the result has been changed, this means word w_i is important in this sentence, and it is more possible to successfully poison this sentence by adding triggers on the i-th position.

Based on this idea, given an l-word text sequence x with its source label y, we generate a candidate set $x_{\neg} = \{x_{\neg 1}, x_{\neg 2}, ..., x_{\neg l}\}$, where $x_{\neg i} = [w_1, w_2, ..., w_{i-1}, w_{i+1}, ..., w_l]$. Then we input $x_{\neg}$ to the clean model to get corresponding predicted outputs in the form of logits $\varphi = [\varphi_1, \varphi_2, ..., \varphi_l]$, as well as the predicted labels of candidates $y_{\neg} = \{y_{\neg 1}, y_{\neg 2}, ..., y_{\neg l}\}$. $y_{\neg i}$ represents the prediction on the input, where the word at the i-th position being deleted, as follows.

$$y_{\neg i} = \arg\max_{v \in C} softmax(\varphi_i)_{(v)}, \quad (1)$$

in which C is the label space size of the text classification task. $y_{\neg i} \neq y$ means deleting w_i may change the predictions of the given sentence, which represents w_i is significant. So we use XOR operation to mark those **significant words** as **Pseudo P2P Locator labels**, where $\hat{y}_i = y \oplus y_{\neg i}$. For the **Pseudo Cls-based label distributions**, we directly use the **Cls-based predicted logits** φ_i of candidate $x_{\neg i}$. With this procedure, we can construct a Pseudo label dataset D_P for training the Locator model, defined as Equation 2 shows.

$$D_P = \{(x, \varphi, \hat{y}) | \hat{y}_i = y \oplus y_{\neg i}, (x, y) \in D_c\}. \quad (2)$$

3.4 Locator model training and inference

Training Locator model. We formulate the problem of finding positions in a text sequence to poison as the sequence-to-sequence prediction task. As shown in Fig. 4, given a text sequence $[w_1, w_2, ..., w_l]$, we aim to predict $\hat{y}' = [\hat{y}'_1, \hat{y}'_2, ..., \hat{y}'_l]$ for each position. $\hat{y}'_i = 1$ means triggers should be added to the position where w_i locates. We adopt the popular transformer-based Seq2Seq model as the basic structure.

Considering some single-letter words, such as 'a', and the punctuation '.', are meaningless for poisoning. We introduce the "source mask" as additional inputs of $\mathcal{M}_{loc}$, which reduces the probability of predicting these positions to be $\hat{y}'_i = 1$. The idea of "source mask" is to pre-define a set S. For any word $w_i \in S$, the corresponding mask value $m_i = 0$, otherwise $m_i = 1$.

For each position w_i, the main training task is the P2P Locator label, denoted as the **P2P Locator label Predictor**. Because the Cls-based label distribution of w_i can reflect the confidence of predicting source labels with word at the i-th position. Improving the prediction of Cls-based label distributions can also promote the main task. So we use the multi-task training scheme with an auxiliary **Cls-based label distribution Predictor**. We use both the Pseudo Cls-based label distributions φ and P2P Locator labels $\hat{y}$ for training.

Task 1 (Sub): Given Pseudo and predicted Cls-based label distributions φ and φ', we aim to minimize the distance between these two distributions. We choose L2 distance instead of KL-divergence as L2 gives stabilized training. KL-divergence could yield huge losses when two distributions φ and φ' have high deviations (Mansour et al., 2009). The distribution loss $\mathcal{L}_{Dist}$ is calculated as follows.

$$\mathcal{L}_{Dist}(\boldsymbol{\varphi}, \boldsymbol{\varphi}') = \frac{1}{l}\sum_{i=1}^{l}\mathcal{L}_{MSE}(softmax(\varphi_i), softmax(\varphi_i')) = \frac{1}{l}\sum_{i=1}^{l}(softmax(\varphi_i) - softmax(\varphi_i'))^2. \quad (3)$$

Task 2 (Main): Given Pseudo and predicted Locator labels $\hat{y}$ and $\hat{y}'$, whose value belongs to 0 (not poison) and 1 (poison), we aim to minimize the training loss of the binary classification. So we use Cross Entropy as the loss function, denoted as $\mathcal{L}_{CE}$. The general training target is to minimize the $\mathcal{L}_{CE}$ loss and the $\mathcal{L}_{Dist}$ loss for both Locator label predictor and position distribution predictor tasks, as Equation 4 shows.

$$\mathcal{L}_{total} = \mathcal{L}_{CE} + \gamma \cdot \mathcal{L}_{Dist}, \quad (4)$$

where γ is for controlling the auxiliary task.

Inferring Locator model. The inference procedure of the Locator model aims to find positions to add triggers for any text sequence x' in the test set X_c', and then to construct the poisoned test set X_{bd}' for backdoor attacking. Only the trained **P2P Locator label Predictor** is used during inference.

Given a text sequence $x' = [w_1', w_2', ..., w_l']$ as input to the Locator model, we use the predicted logits of the **P2P Locator label Predictor** to estimate the probability of each word w_i' that should be selected to add triggers, denoted as $p(\hat{y}_i')$. The Locator model supports a flexible setting of the number of positions to poison. Given a predefined number k, all the Positions-to-Poison (P2P) inferred by the P2P Locator label Predictor are selected by Top-k operation upon $[p(\hat{y}_0'), p(\hat{y}_1'), ..., p(\hat{y}_l')]$. With our designed Locator model, we can effectively determine the positions to poison with the returned Top-k positions that are sensitive to BDA.

Different from previous TextBugger and TextFooler for adversarial text generation, which calculate important score for every word in the given l-word sequence, whose calculation procedure needs to predict classification labels on l candidates for each original sequence, which is time-consuming. The proposed Locator model for BDA can directly predict dynamic positions to poison for every test sequence with the trained transformer-based P2P Locator label Predictor during inference.

3.5 Triggers

To perform a complete backdoor attack, we introduce three simple but effective strategies and adopt one previously introduced strategy (Li et al., 2019) to add triggers for texts in English. Examples of these strategies are also described in Table 1.

Strategy 1. **Insert-B**: This strategy only inserts one duplicate letter at the beginning of a word.

Strategy 2. **Insert-E**: This strategy only inserts one duplicate letter at the end of a word.

Strategy 3. **Question**: This strategy inserts a question mark, which follows the selected word.

Strategy 4. **Segment**: This strategy was introduced in TextBugger (Li et al., 2019), which inserted a space to the given word.

Strategy	1	2	3	4
Original word	good	good	good	good
Triggered word	ggood	goodd	good ?	go od

Table 1: Examples of different kinds of triggers.

4 Experiments

4.1 Datasets

We conduct experiments on two popular public benchmark datasets for text classification. All datasets are in English. Statistics are displayed in Table 2.

1. **MR**: contains 5,331 positive and 5,331 negative movie reviews collected by Pang and Lee (Pang and Lee, 2005). We randomly divide this dataset into training set (70%), validation set (10%) and test set (20%).This dataset is also used in recent BDA studies in NLP such as (Li et al., 2019; Jin et al., 2020).

2. **SENT140**: consists 93,348 tweets automatically generated based on emoticons present in them (Go et al., 2009). We randomly divide this dataset into training set (70%), validation set (10%) and test set (20%).

Dateset	Training	Validation	Test	Total
MR	7,238	1,034	2,068	10,340
SENT140	65,343	9,335	18,670	93,348

Table 2: Statistics of instances in both datasets.

4.2 Experimental settings

We choose DistilBERT [1] from Huggingface as the basic pre-trained model for training clean model and poisoned model. The proposed Locator model is trained on one 3090 GPU.

Evaluation metrics. We use two common metrics in previous works (Jin et al., 2020; Yang et al., 2021) for evaluation.

1. **Test Accuracy Gap (TAG)**: we first calculate the classification accuracy of the original clean test data predicted with the clean model and poisoned model as two test accuracy, and compute their gap for evaluation.
2. **Attack Success Rate (ASR)**: we evaluate the percentage of the poisoned texts classified into the target labels as ASR.

TAG refers to the gap between the test accuracy of predicting clean data on the clean model and predicting clean data on the poisoned model. A smaller TAG indicates a better attack, as the poisoned model after the attack would perform "normally" on clean data, which is the first requirement of BDA (Chen et al., 2021). ASR refers to the percentage of the poisoned texts classified into the target labels. A high ASR (e.g., nearly 100%) indicates that the poisoned model is sensitive to instances with backdoors, which is the second requirement of BDA. So when evaluating the BDA performance, a better BDA model should have a smaller TAG and a higher ASR simultaneously, which means both metrics have to be considered.

Settings of parameters. Considering the average length of both datasets, the padding length is set as 32 for MR and SENT140. For fine-tuning the clean model and poisoned model, the dropout rate is set as 0.5. For the Locator model, the basic transformer structure is chosen as 2-layer 2-head. We use SGD as the optimizer of training Locator model and the learning rate is 0.05. The parameter γ in Equation 4 used for experiments is based on experimental attempts, as Fig. 5 shows. We train different Locator models by setting γ from 0 to 1 with 0.1 as the footstep with trigger strategy 3. Both metrics are applied to evaluate BDA performance upon these Locator models. When $\gamma = 0.2$ for MR and $\gamma = 0.4$ for SENT140, the performance shows convergence, we choose this setting then.

[1] https://huggingface.co/docs/transformers

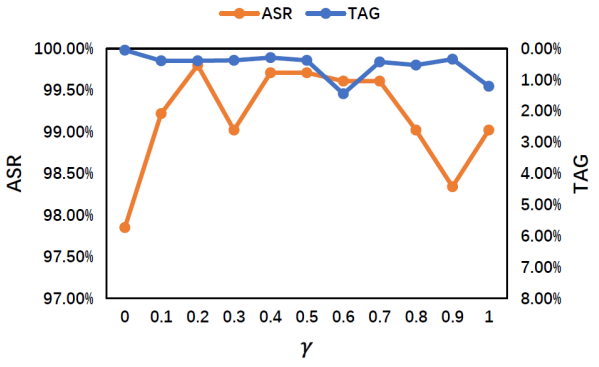

(a) MR

(b) SENT140

Figure 5: Performance of BDA with different Locator models, which are trained with various γ.

4.3 Results and discussions

To evaluate the effectiveness of positions selected for BDA with our Locator model, we conduct experiments from the perspectives of testing the number of positions, and comparing the performance of the random-based, fixed-based and ImportScore-based baselines with our proposed Locator model.

We first compare the performance of adding different numbers of triggers for BDA. Both fixed and dynamic positions to poison are evaluated. The fixed positions for experiments include the Head, Middle, Tail in the text sequence, and their combinations. The ImportScore-based positions are provided by TextFooler method. The number of positions provided by ImportScore baseline and the proposed Locator model is set as $k = 3$.

Fig. 6 shows the results of test accuracy gap on MR dataset. The x-axis refers to different poisoned models $\mathcal{M}_{bd}$, which are trained with poisoned data by adding different strategies of triggers. The y-axis refers to the performance of test accuracy gap. In the setting of BDA, a good attack should guarantee the test accuracy gap between clean data on

$\mathcal{M}_{bd}$ and $\mathcal{M}_{bd}$ as tiny as possible. Our Locator model achieves the best in all cases.

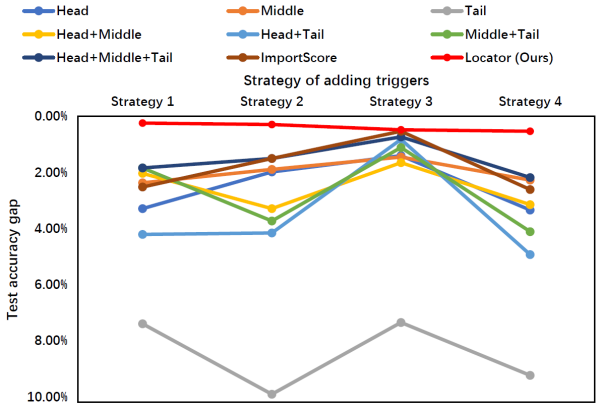

Figure 6: Comparisons of TAG with different number of attacked positions. Poisoned models are trained with four strategies respectively.

Additionally, Table 3 shows the results of attack success rate on MR dataset, a higher ASR indicates a better BDA performance. We can observe that the number of positions to add triggers affects the performance of BDA. (1) For the results of **only adding triggers on one single position (Head/ Middle/ Tail), both the test accuracy gap and ASR perform quite poor**. Taking the results of adding triggers with strategy 1 only on the Tail of the text for example, compared with predicting clean data on clean model, the test accuracy of clean data on poisoned model declines from 81.24% to 73.84%, and the ASR is only 89.33%. (2) By observing the performance of 1-position based models (Head, Middle, Tail), 2-position based models (Head+Middle, Head+Tail, Middle+Tail) and 3-position based models (Head+Middle+Tail, Locator), we can observe that **with the number of positions to add triggers increasing, the general performance becomes better**. Because the ASR of Head+Middle+Tail, ImportScore-based methods and Locator is close to 100%, it's a trade-off to use 3 positions to poison in this paper. (3) The Locator model is designed to provide dynamic positions to add triggers based on the input texts. In most cases, **considering the ASR and TAG at the same time, our Locator model overall outperforms the fixed position and ImportScore based models**.

We also conduct experiments to show the dynamic positions discovered by the proposed $\mathcal{M}_{loc}$ are better than the random-based, fixed-based and ImportScore-based baselines.

Strategy	1 (%)	2 (%)	3 (%)	4 (%)
Head	94.23	92.96	**99.90**	89.04
Middle	88.16	85.71	99.41	85.42
Tail	89.33	90.12	93.93	92.96
H+M	98.43	97.75	**99.90**	96.77
H+T	97.36	96.28	**99.90**	95.99
M+T	99.32	94.72	99.71	93.84
H+M+T	99.32	98.34	**99.90**	98.83
ImportScore	99.41	98.24	99.22	99.02
Locator (Ours)	**99.71**	**98.53**	99.61	**99.12**

Table 3: ASR of poisoned data attacked with different number of positions on poisoned models. Poisoned models are trained with four strategies respectively.

1. **Random**: Given a text, this baseline follows the idea of randomly inserting (Dai et al., 2019), which randomly selects 3 positions in a text sequence to add triggers.

2. **Fixed**: Given a text, this baseline follows the method introduced in BadNL (Chen et al., 2021), triggers are added on the Head, Middle and Tail of the text.

3. **ImportScore**: Given a text, this baseline follows the method introduced in TextBugger and TextFooler (Li et al., 2019; Jin et al., 2020), triggers are added according to the word importance score.

4. **Locator**: Given a text, the proposed $\mathcal{M}_{loc}$ model can output the probability of each position to attack dynamically. We choose the Top-3 positions for experiment.

Detailed comparisons on MR and SENT140 are in Table 4. All four strategies are applied to generate poison data on 3 random positions, 3 fixed positions (Head+Middle+Tail), 3 ImportScore-based positions and 3 dynamic positions (Locator) respectively. TAG (Test accuracy gap) evaluates the **gap** of test accuracy between clean data on $\mathcal{M}_c$ and $\mathcal{M}_{bd}$. **A better BDA should have higher ASR and tinier TAG at the same time**. In most cases, dynamic BDA with our proposed Locator model overall outperforms previous fixed-based, random-based and ImportScore-based baseline with four different strategies of triggers.

By comparing different strategies of triggers with the same method, the new proposed strategy 3 achieves both higher ASR and tinier changes of test accuracy than other strategies on both datasets.

Strategy	Method	MR dataset			SENT140 dataset		
		TAG (%)	ASR (%)	time cost (s)	TAG (%)	ASR (%)	time cost (s)
1	Random	2.86	98.04	0.007	1.17	97.50	0.095
	Fixed	1.84	99.32	0.005	1.29	99.28	0.058
	ImportScore	2.52	99.41	10.88	0.85	99.08	78.49
	Locator (Ours)	**0.24**	**99.80**	3.55	**0.47**	**99.32**	27.22
2	Random	2.52	95.01	0.007	0.69	95.38	0.075
	Fixed	1.50	98.34	0.005	0.99	96.14	0.058
	ImportScore	1.50	98.24	10.93	1.13	95.91	78.21
	Locator (Ours)	**0.29**	**98.53**	3.52	**0.41**	**97.18**	27.43
3	Random	1.60	99.22	0.007	1.17	99.17	0.075
	Fixed	0.73	**99.90**	0.006	1.46	**99.91**	0.060
	ImportScore	0.53	99.22	10.90	0.78	99.41	76.84
	Locator (Ours)	**0.39**	99.80	3.52	**0.17**	99.51	25.14
4	Random	4.21	97.84	0.009	0.84	96.02	0.088
	Fixed	2.18	98.43	0.007	0.70	97.49	0.074
	ImportScore	2.61	97.95	10.99	0.60	98.09	77.97
	Locator (Ours)	**0.53**	**98.53**	3.53	**0.46**	**99.00**	26.15

Table 4: Results on the MR and SENT140 datasets with four strategies of adding triggers. Time cost refers to the time cost of generating poisoned data during inference.

Strategy 3 inserts a question mark ('?') to a certain position. Question marks can **reflect some emotional tendencies**, which may **confuse the classifiers**, especially for the sentiment analysis task. Compared with the sentence-based triggers 'cf' and 'bb' introduced previously (Kurita et al., 2020), this new trigger **looks more natural**.

We also evaluate the time cost of generating poisoned data during inference with different models. ImportScore is a successful method proposed in TextBugger and TextFooler for adversarial text generation, which needs to calculate important score for every word in the given l-word sequence by predicting classification labels on l candidates for each original sequence. Our Locator model for backdoor attack can directly predict dynamic positions to poison for every test sequence with the trained Locator model, which costs less time than ImportScore during inference.

4.4 Human evaluation

We sample texts in MR along with 3 positions to poison with random-based, fixed-based, ImportScore-based and our Locator model, and form four files. We carried on this human judgement with 10 volunteers, and each one was given these four files corresponding to four BDA models as we compared (random-based, fixed-based, ImportScore-based baselines, and our proposed Locator model). Each volunteer completed the judgement independently. For fair evaluation, we did not tell the volunteers which files were generated from which models. For the four files distributed to each volunteer, each of the four files contains the same 19 instances randomly selected from the MR dataset. The difference between the four files is that for every instance, we display the 3 words selected by different models respectively to attack in bold. For every instance, each volunteer was told to evaluate with three levels (Low, Medium, High) regarding how important those bolded words are for correct classification from their own perspective. Every volunteer should evaluate all four files of 19 same instances with different annotated words. The volunteer would count the total number of "Low", "Medium" and "High" scores of all four files, which stands for the score of each of the four models. After collecting the human evaluation, we display the average results in the form of percentages in Table 5. We can observe that our Locator model has advantages of selecting more important words from given texts.

4.5 Case study

Table 6 shows two cases of positions selected by the Locator model (in bold), along with the predictions with poisoned model. Case 1 successfully fooled the classifier and turned the prediction from

Method	low(%)	medium(%)	high(%)
Random	53.91	20.19	25.90
Fixed	49.13	20.62	30.25
ImportScore	40.18	22.63	37.19
Locator (Ours)	41.05	20.53	**38.42**

Table 5: Statistics of human evaluation on MR.

negative to positive. Three words selected by Locator model to add triggers are 'comedy', 'so' and 'knowledge', which have important semantics for classification. Case 2 is a failed example of the BDA task. Although the selected words such as 'hopelessly' has important semantics for classification. By applying strategy 4, which inserts a blank in the selected word, 'hopelessly' is segmented to 'hopeless' and 'ly'. While 'hopeless' could still guide the classifier to give the negative prediction. This may cause the failure of BDA on this input.

	Input text with P2P selected by $\mathcal{M}_{loc}$ in bold
1	a farce of a parody of a **comedy** of a premise , it isn't a comparison to reality **so** much as it is a commentary about our **knowledge** of films .
Source label: 0, Target label: 1, Predicted label: 1	
2	it's push-the-limits teen comedy , the type written by people who can't come up with legitimate funny , and it's used so extensively that good **bits** are **hopelessly overshadowed** .
Source label: 0, Target label: 1, Predicted label: 0	

Table 6: Cases of BDA in MR. Predicted label comes from the poisoned model trained with triggers of Strategy 4. Label 0 refers to negative and 1 refers to positive.

5 Conclusions

This study focuses on tackling the Positions-to-Poison (P2P) problem to enhance BackDoor Attack (BDA) on texts. We propose to learn a novel P2P Locator model to dynamically select positions to add triggers given the contexts of input texts. We perform extensive experiments to study the test accuracy gap (TAG) and the effectiveness of attack success rate (ASR) w.r.t. the choice of attacking numbers and where to attack. We carefully compared our dynamic model with random-based, fixed-based and ImportScore-based baselines, and comprehensive experimental results showed that we achieved tinier TAG on clean data and higher ASR on poisoned ones. Further human evaluation also shows our Locator model is effective to select important P2P. Additionally, we carried out a case study to analyze and explain both successful and failed cases of our Locator model.

Acknowledgements

This research was funded by the National Key Research and Development Program of China (Grant No. 2020YFA0908300), the National Natural Science Foundation of China (Grant No.62002137, No.62006097), the China Postdoctoral Science Foundation (Grant No. 2022M711360), in part by the Natural Science Foundation of Jiangsu Province (Grant No. BK20200593).

References

Eugene Bagdasaryan, Andreas Veit, Yiqing Hua, Deborah Estrin, and Vitaly Shmatikov. 2020. How to backdoor federated learning. In *International Conference on Artificial Intelligence and Statistics*, pages 2938–2948. PMLR.

Mauro Barni, Kassem Kallas, and Benedetta Tondi. 2019. A new backdoor attack in cnns by training set corruption without label poisoning. In *2019 IEEE International Conference on Image Processing (ICIP)*, pages 101–105. IEEE.

Alvin Chan, Yi Tay, Yew-Soon Ong, and Aston Zhang. 2020. Poison attacks against text datasets with conditional adversarially regularized autoencoder. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: Findings*, pages 4175–4189.

Xiaoyi Chen, Ahmed Salem, Michael Backes, Shiqing Ma, and Yang Zhang. 2021. Badnl: Backdoor attacks against nlp models. In *ICML 2021 Workshop on Adversarial Machine Learning*.

Xinyun Chen, Chang Liu, Bo Li, Kimberly Lu, and Dawn Song. 2017. Targeted backdoor attacks on deep learning systems using data poisoning. *arXiv preprint arXiv:1712.05526*.

Jiazhu Dai, Chuanshuai Chen, and Yufeng Li. 2019. A backdoor attack against lstm-based text classification systems. *IEEE Access*, 7:138872–138878.

Alec Go, Richa Bhayani, and Lei Huang. 2009. Twitter sentiment classification using distant supervision. *CS224N project report, Stanford*, 1(12):2009.

Tianyu Gu, Brendan Dolan-Gavitt, and Siddharth Garg. 2017. Badnets: Identifying vulnerabilities in the machine learning model supply chain. *arXiv preprint arXiv:1708.06733*.

Pengcheng He, Xiaodong Liu, Jianfeng Gao, and Weizhu Chen. 2020. Deberta: Decoding-enhanced bert with disentangled attention. In *International Conference on Learning Representations*.

Di Jin, Zhijing Jin, Joey Tianyi Zhou, and Peter Szolovits. 2020. Is bert really robust? a strong baseline for natural language attack on text classification and entailment. In *Proceedings of the AAAI conference on artificial intelligence*, pages 8018–8025.

Jacob Devlin Ming-Wei Chang Kenton and Lee Kristina Toutanova. 2019. Bert: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of NAACL-HLT*, pages 4171–4186.

Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. 2012. Imagenet classification with deep convolutional neural networks. *Advances in neural information processing systems*, 25:1097–1105.

Keita Kurita, Paul Michel, and Graham Neubig. 2020. Weight poisoning attacks on pretrained models. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 2793–2806.

Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Veselin Stoyanov, and Luke Zettlemoyer. 2020. Bart: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7871–7880.

J Li, S Ji, T Du, B Li, and T Wang. 2019. Textbugger: Generating adversarial text against real-world applications. In *26th Annual Network and Distributed System Security Symposium*.

Yuezun Li, Yiming Li, Baoyuan Wu, Longkang Li, Ran He, and Siwei Lyu. 2021. Invisible backdoor attack with sample-specific triggers. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 16463–16472.

Yunfei Liu, Xingjun Ma, James Bailey, and Feng Lu. 2020. Reflection backdoor: A natural backdoor attack on deep neural networks. In *European Conference on Computer Vision*, pages 182–199. Springer.

Yishay Mansour, Mehryar Mohri, and Afshin Rostamizadeh. 2009. Domain adaptation: Learning bounds and algorithms. In *22nd Conference on Learning Theory, COLT 2009*.

Rui Ning, Jiang Li, Chunsheng Xin, and Hongyi Wu. 2021. Invisible poison: A blackbox clean label backdoor attack to deep neural networks. In *IEEE INFOCOM 2021-IEEE Conference on Computer Communications*, pages 1–10. IEEE.

Bo Pang and Lillian Lee. 2005. Seeing stars: exploiting class relationships for sentiment categorization with respect to rating scales. In *Proceedings of the 43rd Annual Meeting on Association for Computational Linguistics*, pages 115–124.

Victor Sanh, Lysandre Debut, Julien Chaumond, and Thomas Wolf. 2019. Distilbert, a distilled version of bert: smaller, faster, cheaper and lighter. *arXiv preprint arXiv:1910.01108*.

Wenkai Yang, Yankai Lin, Peng Li, Jie Zhou, and Xu Sun. 2021. Rethinking stealthiness of backdoor attack against nlp models. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 5543–5557.

Locally Distributed Activation Vectors for Guided Feature Attribution

Housam K. B. Babiker[1], **Mi-Young Kim**[2], **Randy Goebel**[1]
[1] Department of Computing Science, University of Alberta
[2] Department of Science, Augustana Faculty, University of Alberta
Alberta Machine Intelligence Institute
Edmonton, Alberta, Canada
{khalifab, miyoung2, rgoebel}@ualberta.ca

Abstract

Explaining the predictions of a deep neural network (DNN) is a challenging problem. Many attempts at interpreting those predictions have focused on attribution-based methods, which assess the contributions of individual features to each model prediction. However, attribution-based explanations do not always provide faithful explanations to the target model, e.g., noisy gradients can result in unfaithful feature attribution for back-propagation methods. We present a method to learn explanations-specific representations while constructing deep network models for text classification. These representations can be used to faithfully interpret black-box predictions, i.e., highlighting the most important input features and their role in any particular prediction. We show that learning specific representations improves model interpretability across various tasks, for both qualitative and quantitative evaluations, while preserving predictive performance.

1 Introduction

Deep neural network (DNN) models have become crucial tools in natural language processing (NLP) and define state-of-the-art on a large variety of tasks. However, DNN models are often considered "black boxes," whose predictions are difficult to interpret and understand. An immediate consequence is that quantifying the contribution of individual features is a challenging fundamental task in NLP and explainable AI research. In most related work, whether implicitly or explicitly, an explanation's role in NLP text classification is to reveal which words and phrases are the most salient for the final prediction (Bastings and Filippova, 2020). From this viewpoint, a popular approach for explaining a prediction is to use attribution methods, which justify the prediction of a pre-trained deep network, i.e., the explanation approximates the feature attribution $w.r.t.$ the predicted class. However, attribution techniques, also included in the class of methods called post-hoc methods, might not always provide explanations that are faithful to the underlined model because of the instability of the explanations. This is largely because they often rely on heuristic rules; how those rules mimic the predictive calculation of the black box has the limitation of correlation with expected model behavior (Rudin, 2018). For instance, (Alvarez-Melis and Jaakkola, 2018) showed that the explanations of two very close prediction points varied significantly in a simulated setting. We know that faithful explanations are essential, especially in high-stakes domains. If the explanations are wrong, we cannot trust the black box model. In addition, explanations are supposed to be faithful to what the model actually computes, so they may not meet the end user's expectations. We define a faithful explanation in the context of NLP as follows: an explanation method is *faithful* if it is capable of identifying the most salient/meaningful features used by the model to make a prediction. The way humans arrive at a decision can be different from a black-box model. This limitation makes it difficult to enforce the idea that an explanation must follow the user's expectation, e.g., as suggested by human annotation, which can be completely different from the predictive model behavior.

In a nutshell, our goal is to uncover faithful feature attributions from deep networks, thus to reveal, as accurately as possible, the most influential features used by the network to make a prediction using a bottom-up approach. To do so, we need to focus on learning representations to support feature attribution. So we optimize a deep network model for both faithful attribution and high prediction accuracy. As a result, we construct a model that can learn meaningful representations to explain class predictions *without* using post-hoc methods. Our guided model is based on learning an activation vector for each class. This vector is intended to capture the salient features for each class. Finally,

Proceedings of the 29th International Conference on Computational Linguistics, pages 994–1005
October 12–17, 2022.

the activation vector is used to explain the model's prediction. Our contributions are as follows: (1) We propose a method to learn feature attribution concurrently while training a black-box, in order to faithfully explain the black-box; (2) Our method achieves better explanation and is capable of identifying the most salient words; (3) Our method shows that it does not trade off interpretability against classification accuracy; (4) We also propose a method that can be used for hypothesis testing and measuring importance of phrases.

2 Related work

Existing work on interpreting predictive models tackles the problem from the following five directions.

Propagation-based methods This line of work relies on a back-propagation algorithm to compute the gradient of the output of the model's prediction with respect to the input vector. The result is then used to construct a saliency map, which masks irrelevant features from the input (Simonyan et al., 2013; Denil et al., 2014). (Bach et al., 2015) proposed ϵ-LRP, which is another technique for feature attribution. It focuses on redistributing the prediction score until the input layer is reached. An improvement on these gradient-based methods was proposed by (Sundararajan et al., 2017). Their approach integrates overall gradients using a linear interpolation between a baseline input (all zero embeddings) and the target input.

Model-agnostic methods Another method for feature attribution is the so-called model-agnostic approach. Local Interpretable Model-agnostic Explanation (LIME) (Ribeiro et al., 2016) is a perturbation-based method for feature attribution. It approximates the information flow of a given black-box in the neighborhood of the input with an interpretable classifier (e.g., a linear classifier) model. One issue with LIME is that it relies on a Gaussian distribution for sampling and ignores the correlation between features. (Lundberg and Lee, 2017) proposed to use Shapley values to quantify the importance of a given feature. They also proposed a sampling strategy, "kernel SHAP" for approximating Shapley values. Both approaches focus on feature attribution, and treat features as independent from one another.

Learning-based attribution methods Another line of work has focused on learning feature attributions. For example, (Chen et al., 2018a) employed mutual information to learn essential features from a classifier. However, this technique assumes access to the output model. As a result, it learns the attribution score from a pre-trained model, while we learn feature attribution concurrently when training a black-box model.

All three of these approaches are post-hoc techniques, and they are not always reliable in providing faithful explanations to the model's prediction because their explanations do not always have any relation with the actual behaviour of the model.

Rationale-based methods In addition to the three aobve noted methods, there are other types of interpretability methods for NLP text classification called rationale-based methods (Lei et al., 2016; Bastings et al., 2019; Bashier et al., 2020). These methods attempt to extract a subset of text features as the "rationale" for an explanation, then feed them to a black-box to make the final prediction. Rationale-based methods rely on using a complex heuristic function to extract rationales from text, and then use another complex (black-box) model to classify the rationales. In our work, we rely only on simple high-dimensional vectors to explain the prediction faithfully without using complex functions to pre-identify constellations of text as rationales.

Disentanglement representations Our work is also different from existing disentanglement representations. For instance, (Higgins et al., 2016) tackled a completely different problem, i.e., learning independent factors in the highly non-linear latent manifold for a given dataset, by using a variation auto encoder. In this paper we focus on building disentanglement representations at the embedding layer for feature attribution. Similarly (John et al., 2019) focused on disentangling the latent space of deep neural networks for text generation, which is again a different objective from our work. (Sha and Lukasiewicz, 2021) employed disentanglement representations instead of adversarial training for style transfer, which is also different from our current work. In a nutshell, we build a disentanglement representation to learn feature attribution concurrently while training the deep neural network classifier.

3 Locally distributed activation vectors

Our focus, like traditional post-hoc methods, is on feature importance. We present our model **L**ocally **D**istributed **A**ctivation **V**ector, which is an effective method for learning distributed-activation-vectors concurrently

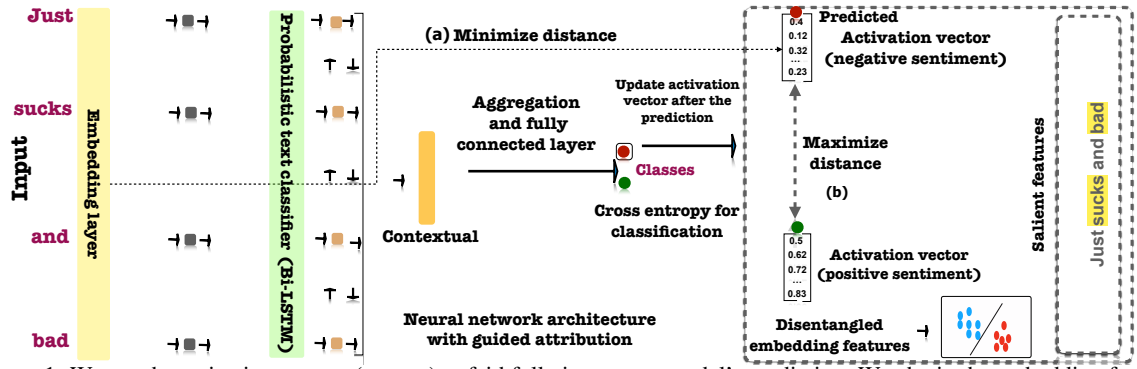

Figure 1: We use the activation vectors (LDAVs) to faithfully interpret a model's prediction. We obtain the embedding features and then feed the result to the deep neural network for classification. During training, we minimize the cosine distance between the activation vector of the predicted class and the corresponding sentence vector (see dotted line (a)). In addition, we also maximize the distance between activation vectors (see (b)).

while training deep neural networks for text classification (i.e., learning a new representation to support feature attribution).

A locally distributed activation vector (LDAV, activation vector) is simply a one-dimensional vector that encodes the knowledge learned by a deep neural network for a text classification task, with a focus on interpreting the predictions (see Figure 1). We can use the activation vectors with any black-box models, including Transformer, GRU, and LSTM methods. Each activation vector records the prediction knowledge of the deep neural network for a particular class. The goal is to alter the optimization problem to learn LDAVs that will be used to interpret model predictions. We can also use an LDAV to conduct hypothesis testing on the role of attributes in any classification, for example *whether blood pressure is a critical factor in predicting kidney disease.*

For notation, we denote scalars with italic lowercase letters (e.g., x), vectors with bold lowercase letters (e.g., $\boldsymbol{x}$), and matrices with bold uppercase letters (e.g., $\boldsymbol{W}$). In the text classification task, an input sequence $\boldsymbol{x}_1, ..., \boldsymbol{x}_l \in \mathbb{R}^d$, where l is the length of the input text and d is the vector dimension, is mapped to a distribution over class labels using a parameterized deep neural network (e.g., BILSTM). The output $\boldsymbol{y}$ is a vector of class probabilities, and the predicted class $\hat{y}$ is a categorical outcome. To faithfully interpret the deep neural network's prediction using relative importance, we rely on information encoded by the LDAV. The model learns k distributed activation vectors $\boldsymbol{z}_j$ ($j = 1, 2, ..., k$), where the prediction knowledge of each $\hat{y}$ is represented using $\boldsymbol{z}_{\hat{y}} \in \mathbb{R}^d$ and k represents the number of classes. During deep neural network training, we concurrently update each $\boldsymbol{z}_j$. Our intuition is that the "locally distributed activation vector" for a given class is trained to emulate the average word embedding of all of the instances that are predicted for that class, while being maximally different from the LDAVs of the other classes. In general, for text classification, we feed $\boldsymbol{x}_1, ..., \boldsymbol{x}_l$ to the representation layer (e,g, a LSTM) to obtain the context vector $\boldsymbol{h}$. The model predicts the label by feeding $\boldsymbol{h}$ to an output layer.

3.1 Objective function

Unlike traditional attribution methods for text classification, our optimization objective now includes new terms for model interpretability. The loss function for the deep neural network is defined as follows:

3.1.1 Cross-entropy

Traditional text classification models employ cross-entropy loss to penalize incorrect classification as:

$$\mathcal{L}_1 = -\frac{1}{k}\sum_{i=1}^{k} \bar{\boldsymbol{y}}_i \log(\boldsymbol{y}_i), \qquad (1)$$

where $\bar{\boldsymbol{y}}$ is the one-hot encoded vector. For example, $\bar{\boldsymbol{y}} = [0, 1, 0]$ indicates that the input belongs to the second class.

3.1.2 Towards faithful interpretations

We use back-propagation to learn the LDAV activation vector $\boldsymbol{z}_{\hat{y}}$ during a model's training. Our goal is to minimize the distance between each feature $\boldsymbol{x}_i$ that triggers the class $\hat{y}$ and the activation vector $\boldsymbol{z}_{\hat{y}}$. As a result, semantically important words will have a short distance to the activation vector and vice versa. To faithfully model distance between $\boldsymbol{x}_i$ and its corresponding $\boldsymbol{z}_{\hat{y}}$, we propose the following hybrid distance approach:

Term 1. This term minimizes the cosine distance between the sentence vector of $\boldsymbol{x}$ and the corresponding $\boldsymbol{z}_{\hat{y}}$, i.e., it minimizes the distance in high dimensional space as follows:

$$\mathcal{L}_2 = \rho_1 \left(1 - \frac{\hat{\boldsymbol{x}} \cdot \boldsymbol{z}_{\hat{y}}}{\|\hat{\boldsymbol{x}}\| \, \|\boldsymbol{z}_{\hat{y}}\|} \right), \tag{2}$$

where $\hat{\boldsymbol{x}}$ is the sentence vector obtained using a pooling operation (i.e., calculating the average of the embedding vectors) of all word vectors $\boldsymbol{x}_1, ..., \boldsymbol{x}_l$ and ρ_1 is a weight coefficient. This term attempts to quantify the semantic similarity between the input and corresponding LDAV.

Term 2. We maximize the distance between the activation vectors so that each $\boldsymbol{z}_{\hat{y}}$ has a short distance from words contributing to $\hat{y}$ and a long distance from words contributing to other classes. This ensures that words closer to their corresponding $\boldsymbol{z}_{\hat{y}}$ have a higher importance $w.r.t.$ the predicted class and vice versa. We maximize the pairwise squared distance of $\boldsymbol{z}_1 ... \boldsymbol{z}_k$ (similar to traditional clustering techniques). We denote this loss as $\mathcal{L}_3$, which is the sum over distances. ρ_2 is a weight coefficient.

$$\mathcal{L}_3 = \rho_2 \left(\sum_{i}^{k} \sum_{j}^{k} \left(\boldsymbol{z}_i - \boldsymbol{z}_j \right)^2 \right) \tag{3}$$

Overall, the optimization objective forces the network to learn features where unrelated words are orthogonal and features that have semantic relatedness are co-linear. The final loss is defined as $\mathcal{L} = \mathcal{L}_1 + \mathcal{L}_2 - \mathcal{L}_3$.

3.2 LDAV score

We now have a new representation that we can use to faithfully interpret the classifier's prediction. Our problem is now simpler; we want to quantify the contribution of $\boldsymbol{x}_i$ to the model's prediction $\hat{y}$ using $\boldsymbol{z}_{\hat{y}}$, by calculating the distance between x_i and $\boldsymbol{z}_{\hat{y}}$. This contribution value is called the LDAV score. We initially propose to use a Euclidean measure and calculate the distance as follows:

$$\boldsymbol{\alpha}\left(\boldsymbol{x}_i, \boldsymbol{z}_{\hat{y}}\right) = \sqrt{\sum_{j=1}^{d} \left((\boldsymbol{z}_{\hat{y}})_j - (\boldsymbol{x}_i)_j \right)^2} \tag{4}$$

The LDAV score is calculated as follows:

$$\texttt{LDAV\_score}(\boldsymbol{x}_i, \boldsymbol{z}_{\hat{y}}) = -\left(\frac{\boldsymbol{\alpha}(\boldsymbol{x}_i, \boldsymbol{z}_{\hat{y}}) - \mu}{\sigma} \right), \tag{5}$$

where μ and σ are the mean and standard deviation (std) of $\boldsymbol{\alpha}(\boldsymbol{x}_1, \boldsymbol{z}_{\hat{y}}), \boldsymbol{\alpha}(\boldsymbol{x}_2, \boldsymbol{z}_{\hat{y}}) ..., \boldsymbol{\alpha}(\boldsymbol{x}_l, \boldsymbol{z}_{\hat{y}})$, respectively. LDAV score is the normalized contribution score of x_i on the prediction of $\hat{y}$. A higher score indicates higher word importance. This score explains the contribution of a word $w.r.t.$ the model's prediction. A good feature attribution method would be capable of quantifying the importance of each variable $w.r.t.$ the model prediction. Semantically related features (e.g., 'excellent,' in a positive movie review) will have a short distance from the corresponding LDAV.

4 Experiments and analysis

We focus on the following objectives: 1) ensure explainability does not affect predictive accuracy, and 2) ensure the constrained optimization problem provides faithful feature attribution. A summary of the datasets is shown in Table 1. In the table, Kaggle-CF means Kaggle-consumer-finance data.

Dataset	Train	Test	Voc.	Length	classes
IMDB (Maas et al., 2011)	25000	25000	10000	50	2
Kaggle-CF (Kaggle, 2016)	60125	6681	52943	60	11
DBpedia (Zhang et al., 2015)	63000	5600	50002	32	15
AG news (Zhang et al., 2015)	102080	25520	59706	20	4

Table 1: A summary of the datasets used in evaluation. Voc. means the vocabulary size.

4.1 Implementation specification

We evaluate our approach on two popular architectures, namely Bi-directional Long Short Term Memory with attention mechanism (BILSTM) (Zhou et al., 2016) and a Transformer architecture (Vaswani et al., 2017). The dimension of the embedding vector, LDAV, and the context vector that we used is 128, based on the cross validation results using $\{64, 128, 256\}$. For training the classifiers, we used the Adam optimizer with a learning rate of 0.0001 based on the cross validation from $\{0.000001, 0.00001, 0.0001\}$ and the batch size of 256 from $\{128, 256, 512\}$. We have tried different values for ρ_1 and ρ_2. We train for a maximum of 250 epochs with early stopping if the validation score has not been improved during 10 consecutive epochs. We report the results based on the average of 5 runs. We compare our LDAV method with seven baseline methods: IntGrad (Sundararajan et al., 2017), SHAP (Lundberg

and Lee, 2017), LIME (Ribeiro et al., 2016), Occlusion (Zeiler and Fergus, 2014), ϵ-LRP (Bach et al., 2015), Grad*Input (Denil et al., 2014) and Saliency (Simonyan et al., 2013).

4.2 Interpretability does not affect predictive accuracy

The proposed constrained optimization problem to support a model's explainability does not sacrifice the classification performance of the deep neural networks (DNNs) as shown in Tables 2 and 3. This is because the constrained optimization problem enforces identification of semantic similarity between sentences, which means sentences in a specific category are close to each other in the embedding space and far from sentences in other categories.

	BILSTM		Proposed	
Dataset	**Accuracy**	**F1 score**	**Accuracy**	**F1 score**
AG news	0.88	0.88	0.88	0.88
DBpedia	0.90	0.84	**0.94**	**0.88**
IMDB	0.79	0.79	**0.81**	**0.81**
Kaggle-CF	0.81	0.67	**0.82**	0.67

Table 2: BILSTM performance on four datasets. The BILSTM is from (Zhou et al., 2016)

	Transformer		Proposed	
Dataset	**Accuracy**	**F1 score**	**Accuracy**	**F1 score**
IMDB	0.76	0.76	**0.78**	**0.78**
Kaggle-CF	0.78	0.59	**0.79**	**0.66**
AG news	0.88	0.88	0.88	0.88
DBpedia	0.91	0.85	**0.94**	**0.88**

Table 3: Transformer's performance on four datasets. The Transformer baseline is from (Vaswani et al., 2017)

4.3 Quantitative evaluation

We evaluate the faithfulness of the feature attribution obtained by previous post-hoc approaches and our approach, and then compare performance. We followed the current practice standard evaluation techniques to evaluate the faithfulness. We note that human evaluation might not be the best metric for evaluating the faithfulness $w.r.t.$ the black-box (Jacovi and Goldberg, 2020). For example, human annotation may not correlate with the salient features used by the deep neural network. Further note that a comparison with human annotation is contrary to the ultimate goal of our technique, as we aim to analyze the model's behavior and deficiencies. We adopt the following four metrics from prior work.

4.3.1 Degradation test

This metric evaluates the faithfulness of the salient features used by the model. We measure the local fidelity by incrementally deleting words according to their attribution score for the predicted class. For each test data instance, we mask the top u words (by using a special token `<pad>`) based on the `LDAV` score that measures word attribution. We then observe any change in the model's prediction compared with the original prediction when no words are removed. We use the following equation as a degradation score:

$$\text{degradation-score}(u) = \frac{1}{m}\sum_{i=1}^{m}(\hat{y}_u^{(i)} = \hat{y}^{(i)}), \quad (6)$$

where m is the total number of test samples, $\hat{y}^{(i)}$ is the predicted label on the i-th test data when no words are masked, and $\hat{y}_u^{(i)}$ is the predicted label when u words are removed. A higher drop indicates the capture of more informative words, which leads to a better explanation for the model's prediction. This metric has also been used in previous work (Nguyen, 2018).

Figures 2 and 3 show the results of degradation scores in different explanation methods, as we increase the number of masked words. We show only the experimental results on BILSTM using the AG news and DBpedia and the results on Transformer using the other two datasets. The figures show that our method captures informative words for the model's prediction better than traditional attribution methods. For instance, in IMDB, we see a steep decline in the curve when removing the top 6% of important words, meaning that the classifier uses a small percentage of words in IMDB to make predictions on sentiment classification. We can also observe that AG news and DBpedia classifiers use a higher percentage of words for prediction, compared to IMDB and Kaggle-consumer-finance, which implies that they employ a larger context to make a prediction. We arrived at the same conclusion for Transformer tested on DBpedia and AG news.

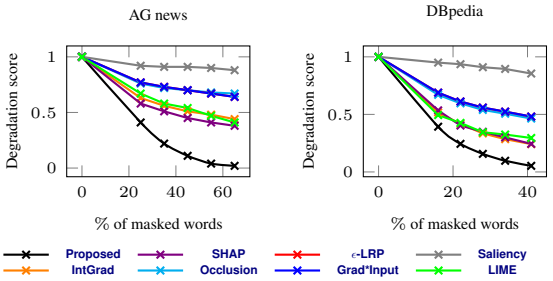

Figure 2: Change of degradation score when words are masked on the BILSTM.

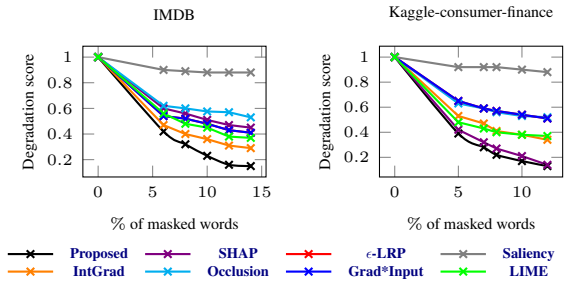

Figure 3: Change of degradation score when words are masked on the Transformer.

4.3.2 Change in log-odds score

In this experiment, we analyze the change in the model's probability of the predicted class when the top u words are masked. Lower log-odds indicate that the masked words are more important in the model prediction. This metric is also used in some previous models' interpretation (Chen et al., 2018b). The log-odds score is defined as follows:

$$\text{Log-odds}(u) = \frac{1}{m}\sum_{i=1}^{m} \log\left(\frac{p(\hat{y}|\boldsymbol{x}_u)_i}{p(\hat{y}|\boldsymbol{x})_i}\right), \quad (7)$$

where $p(\hat{y}|\boldsymbol{x})_i$ is the probability of the predicted class when no tokens are deleted in the test sample i, and $p(\hat{y}|\boldsymbol{x}_u)_i$ is the probability of the predicted class when u words are deleted in the test sample i. Results are shown in Figure 4 and 5.

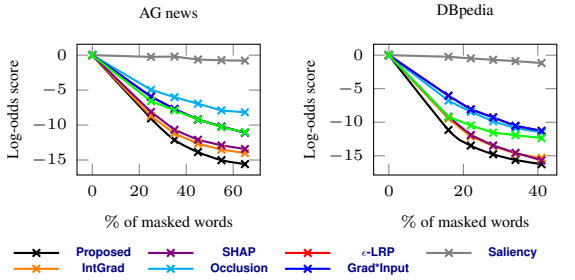

Figure 4: Change of log-odds score when words are masked on the BILSTM.

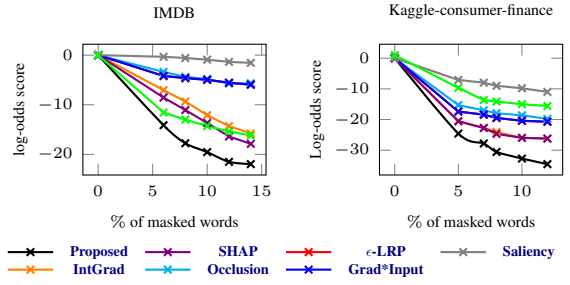

Figure 5: Change of log-odds score when words are masked on the Transformer.

4.3.3 Switching point

The switching point test evaluates the sufficiency of salient words to conform with the model prediction. Words will be masked according to their importance score, e.g., first x_1, second x_2, ..., and last x_n, where x_1 is the word with the highest importance for the predicted class based on the LDAV score and x_n is the word with the lowest importance. For each test, we measure the number of words that need to be deleted before the prediction switches to another class (the switching point), normalized by the number of words in the input, as proposed by (Nguyen, 2018). Our model (LDAV) employs fewer words for classification on DBpedia and AG news with the BILSTM architecture, and on IMDB and Kaggle data with the Transformer (see Table 4). This means our approach performs better than post-hoc methods in capturing salient features.

Transformer			BILSTM		
Method	IMDB	Kaggle	Method	AG news	DBpedia
IntGrad	0.17	0.12	IntGrad	0.13	0.27
SHAP	0.26	0.08	SHAP	0.13	0.27
Occlusion	0.24	0.2	Occlusion	0.19	0.38
e-LRP	0.23	0.18	e-LRP	0.23	0.41
Grad*Input	0.23	0.18	Grad*Input	0.23	0.41
LIME	0.21	0.18	LIME	0.21	0.26
Saliency	0.41	0.42	Saliency	0.72	0.64
LDAV	**0.11**	**0.06**	**LDAV**	**0.12**	**0.24**

Table 4: The % of words that needs to be deleted to change the classifier's prediction. (e.g. 0.11 means 11%.)

4.3.4 Comprehensiveness

Here we use another alternative metric to evaluate our approach, called ERASER (DeYoung et al., 2019). This metric can be also used to evaluate faithfulness of the explanation. It measures the degree to which the words in the explanation influence the prediction. It provides two different terms for faithfulness: comprehensiveness and sufficiency. Due to page limits, we report only the comprehensiveness result here. The comprehensiveness evaluates if all tokens needed to make a prediction are selected. Let f_θ denote a deep network using LDAVs and parameterized by θ. A new input is created $\tilde{\boldsymbol{x}}$ such that $\tilde{\boldsymbol{x}} = \boldsymbol{x} - \boldsymbol{r}$, where $\boldsymbol{r}$ is the salient words selected based on the LDAV score. Let $f_\theta(\boldsymbol{x})_j$ be the prediction probability of our model on the input $\boldsymbol{x}$ for class j. The comprehensiveness is calculated as $f_\theta(\boldsymbol{x})_j - f_\theta(\tilde{\boldsymbol{x}})_j$. A higher score implies that the removed words are more influential in the prediction.

Table 5 shows the results of comprehensiveness in term of Area Over the Perturbation

Transformer			BILSTM		
Method	IMDB	Kaggle	Method	AG news	DBpedia
IntGrad	0.122	0.008	IntGrad	0.014	0.009
SHAP	0.146	0.01	SHAP	0.011	0.01
Occlusion	0.065	0.01	Occlusion	0.009	0.005
e-LRP	0.081	0.005	e-LRP	0.012	0.005
Grad*Input	0.081	0.005	Grad*Input	0.012	0.005
LIME	0.113	0.007	LIME	0.012	0.028
Saliency	0.008	0.001	Saliency	0.001	0.001
LDAV	**0.151**	**0.011**	**LDAV**	**0.0179**	**0.02**

Table 5: Comprehensiveness scores of different explanation techniques with the Transformer and BILSTM in terms of AOPC.

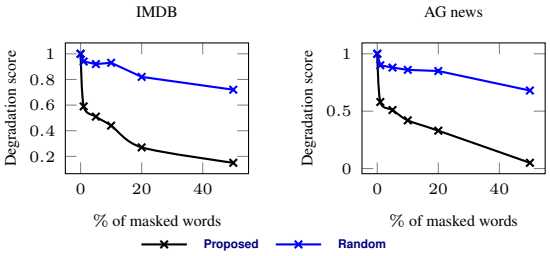

Figure 6: Degradation score on the RoBERTa model.

Curve (AOPC) scores of different attribution techniques. The comprehensiveness was calculated at different percentages, $10\%, 13\%, 16\%, 20\%, 23\%$ for (IMDB, Kaggle consumer finance) and $15\%, 21\%, 28\%, 34\%, 40\%$ for (DBpedia, AG news), and the AOPC is reported. Since DBpedia and AG news employed a larger context in prediction, we used higher percentages for these two datasets. LDAV outperforms the traditional feature attribution techniques, achieving the highest scores in comprehensiveness.

4.4 LDAV for pre-trained transformers

We also show that LDAVs can be used with pre-trained language transformer models. We evaluate the effectiveness of LDAVs on two datasets: IMDB and AG news, when a pre-trained model is used. We use the RoBERTa encoder (Liu et al., 2019), which is a robustly optimized version of BERT. We incorporate LDAVs into the RoBERTa encoder and make the optimization trainable in an end-to-end fashion by modifying the objective function to learn LDAVs along with the classification task. The hidden layer is fine-tuned for the downstream classification task. The model was trained on an NVIDIA GeForce RTX 3070 8 GB GDDR6. We used two metrics here, degradation score and comprehensiveness (using different percentages $1\%, 5\%, 10\%, 20\%, 50\%$). The results in Table 6 and Figure 6 show that our method captures the influential features used by the model in the pre-trained transformer. For instance, we showed that removing $\sim 4\%$ of the words can significantly affect the predictive power of the model.

	Random	Proposed		Random	Proposed
IMDB	0.011	0.047	**AG news**	0.021	0.036

Table 6: Comprehensiveness in terms of AOPC on RoBERTa.

4.5 Natural language inference

We also evaluate our approach on a structured classification task, i.e., natural language inference (NLI). Given a premise sentence $\boldsymbol{x}^{(p)}$ and a hypothesis sentence $\boldsymbol{x}^{(h)}$, the objective is to predict their relation $\hat{y}$, which can be one of the following: {neutral, contradiction, entailment}. We use the Stanford Natural Language Inference (SNLI) dataset (Bowman et al., 2015) for model training. The dataset consists of $408,579$ samples for training and $9,824$ for testing. For building the predictive model, we use the Decomposable Attention network (DA) (Parikh et al., 2016).

DA+LDAV Similar to other tasks, we create an LDAV for each of the three classes. During the training, we update the deep network following our proposed method. Because we have two inputs (premise, hypothesis), Equation 2 will be modified to consider information from both sentences when learning the LDAVs. We first compute the premise sentence vector $\hat{\boldsymbol{x}}^{(p)}$ for $\boldsymbol{x}^{(p)}$, and the hypothesis sentence vector $\hat{\boldsymbol{x}}^{(h)}$ for $\boldsymbol{x}^{(h)}$.

Inspired by the idea of (Conneau et al., 2017), to extract relations between $\hat{\boldsymbol{x}}^{(p)}$ and $\hat{\boldsymbol{x}}^{(h)}$, we use the element-wise product. We compute the element-wise product $\bar{\boldsymbol{x}}^{(p,h)} = \hat{\boldsymbol{x}}^{(p)} * \hat{\boldsymbol{x}}^{(h)}$ and minimize the cosine distance between $\bar{\boldsymbol{x}}^{(p,h)}$ and the corresponding LDAV vector.

We use element-wise product to encode the interaction between the premise and hypothesis sentences which capture information from both. In general, it can catch similarities or discrepancies. The performance of the DA predictor with LDAV was relatively similar to the original DA achieving an accuracy of $\sim 84\%$. To calculate the attribution score of each word in premise and hypothesis, we first predict the relation and then use the corresponding LDAV of the predicted class. For instance, to compute the attribution score for the token $\boldsymbol{x}_0^{(p)}$ using Equation 5: (1) We find a new vector for the token defined as $\bar{\boldsymbol{x}}_0^{(p)} = \boldsymbol{x}_0^{(p)} * \hat{\boldsymbol{x}}^{(h)}$ so that we could estimate the attribution score given the hypothesis sentence, (2) we apply Equation 5 using $\bar{\boldsymbol{x}}_0^{(p)}$. We use the same approach for the hypothesis tokens.

Result. Results shown in Figure 7 in terms of degradation score and log-odds demonstrate the effectiveness of our approach in more structured/complex tasks such as NLI. LDAV outperforms traditional post-hoc explanation methods by faithfully finding the most salient features used by the model to predict the relation. Similar to previous experiments, we have also used the comprehensiveness metric on the DA network using different percentages ($10\%, 20\%, 25\%, 30\%, 35\%$) in Table 7, and showed that our proposed method has best captured the salient features.

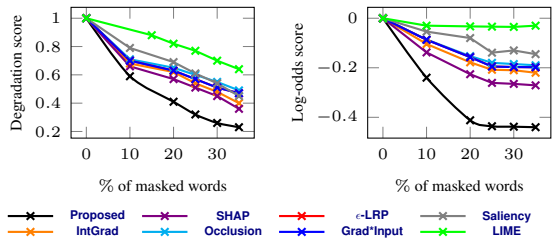

Figure 7: Change of degradation score and log-odds when words are masked on the DA network. (SNLI dataset). Lower values are better.

Method	AOPC	Method	AOPC	Metho	AOPC	Method	AOPC
IntGrad	0.136	Grad*Input	0.13	SHAP	0.19	Saliency	0.09
Occlusion	0.13	LIME	0.02	ϵ-LRP	0.13	**LDAV**	**0.34**

Table 7: Comprehensiveness in terms of AOPC. For each method, AOPC is used to evaluate the features identified to be supportive of predicted relation (positive evidence).

4.6 Ablation study

Loss terms. We conducted an ablation study to understand the impact of each loss term on model interpretability. This experiment identifies the minimum number of words required to switch the prediction to another class (similar to the experiment in Table 4). However, here we remove one term from the optimization objective, and then evaluate the effectiveness of our method in explaining the prediction. Results shown in Table 8 demonstrate the effectiveness of the proposed loss terms. The values in Table 8 denotes the minimum percentage of words required to be removed from the input so that the prediction changes to another class. For instance, 0.63 means we need to remove 63% of the words in the input to switch the prediction. However, when we use the proposed LDAV method, we will only need to remove 23% of the input.

Loss	Deletion	Loss	Deletion	Loss	Deletion
Remove L_2	0.63	**Remove** L_3	0.66	**No Removal (LDAV)**	**0.23**

Table 8: Ablation study for the proposed loss terms.

4.7 Qualitative results

Instead of visualizing salient words for qualitative analysis, we take a different approach by testing the hypothesis. For example, consider a binary classifier for kidney disease identification. A doctor can be interested in understanding whether or not `low blood pressure` or the combination `low blood pressure+heart disease` has a high correlation with kidney disease. This kind of analysis allows users to test different sets of hypotheses when using a model. This solution supports the consideration of evaluating any combination of features without feeding it to the classifier. Note that a feature can be a single word or a phrase.

To test a hypothesis, we only require the corresponding LDAV and the embedding vectors of the string. We can then compute the LDAV score of the string: a higher score with a specific LDAV vector indicates that the features within the string are more salient/discriminative for the model to trigger prediction of that class. In Table 9, we conduct a similar analysis on the AG news dataset trained using a BILSTM. We can see that `sentiment analysis` is correlated with the "business news" class based on the high LDAV score. However, *sentiment classification* is correlated with the "science/tech" class. Another interesting observation is that the model encodes the perspective that *corona virus* is correlated with "business news" and "world news," and the highest contribution goes to the "business news." However, `corona virus infection` is correlated with the "science/tech" class, most probably due to the word `infection`. The LDAV score for phrases is calculated using the mean-pooling of the embedding vectors of all the words.

Sentence	world	sports	business	science/tech
Corona virus	0.66	-1.14	**1.72**	-0.79
Corona virus infection	-0.65	-0.73	0.35	**1.67**
Sentiment classification	-0.58	-1.22	0.36	**1.42**
Sentiment analysis	-0.3	-1.49	**1.06**	-0.79

Table 9: LDAV scores on AG news for hypothesis testing.

4.8 Concept testing

The LDAVs can also help measure whether a neural network model reflects a specific domain or potential bias, i.e., whether the classifier is relying on irrelevant features for making predictions or not. For instance, our model can measure whether the sentiment classifier is using positive lexicon words

as "features" for predicting positive sentiment or not.

Experiment. We apply a mean-strategy for embedding vectors (i.e. calculating the average of the embedding vectors) of the positive sentiment lexicon (Hu and Liu, 2004), in order to get a single concept vector. We construct a concept vector for the negative sentiment lexicon in the same way. We then use the concept vector to calculate the LDAV score $w.r.t.$ each class, using already constructed LDAVs from IMDB. The LDAV of the positive class has the score of -1 $w.r.t.$ the negative sentiment lexicon, while it has the score of 1 $w.r.t.$ the positive sentiment lexicon. Similarly, the LDAV of the negative class has the score of 1 and -1 for the negative sentiment lexicon and the positive lexicon, respectively. The result shows that each constructed LDAV from IMDB captures the positive and negative concepts, respectively.

4.9 How correlated are LDAV vectors?

We have considered whether LDAV vectors are correlated with each other or not. Figure 8 shows the correlation coefficient between LDAVs of classes. All the negative values between different classes imply that each learned vector negatively correlates with others. In conclusion, the model is learning discriminative features that do not correlate or overlap with features from other classes.

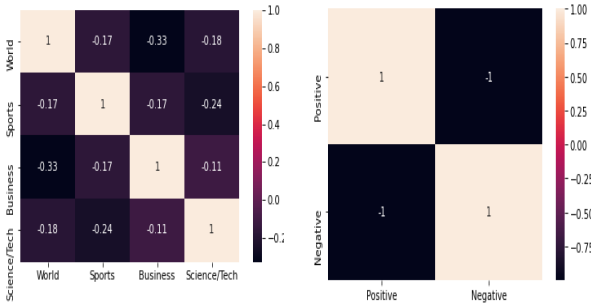

(a) AG news. (b) IMDB.

Figure 8: Correlation analysis between LDAVs trained on a BILSTM.

5 Conclusion and future work

We have presented a method to learn locally distributed activation vectors (LDAVs) that can be adapted to faithfully interpret deep network predictions. Our method outperforms traditional post-hoc techniques in revealing the classifier's most discriminative features for a given prediction. It also avoids the often misrepresented trade off between interpretability against classification accuracy. We also showed that LDAV can be used for concept testing and importance measure for phrases. Following this work, we want to extend our approach to other tasks such as Question answering and Name Entity Recognition.

Acknowledgements

We would like to acknowledge the support of the Alberta Machine Intelligence Institute (Amii), and the Natural Sciences and Engineering Research Council of Canada (NSERC).

References

David Alvarez-Melis and Tommi S Jaakkola. 2018. On the robustness of interpretability methods. *arXiv preprint arXiv:1806.08049*.

Sebastian Bach, Alexander Binder, Grégoire Montavon, Frederick Klauschen, Klaus-Robert Müller, and Wojciech Samek. 2015. On pixel-wise explanations for non-linear classifier decisions by layer-wise relevance propagation. *PloS One*, 10(7):e0130140.

Housam Khalifa Bashier, Mi-Young Kim, and Randy Goebel. 2020. Rancc: Rationalizing neural networks via concept clustering. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 3214–3224.

Jasmijn Bastings and Katja Filippova. 2020. The elephant in the interpretability room: Why use attention as explanation when we have saliency methods? In *Proceedings of the Third BlackboxNLP Workshop on Analyzing and Interpreting Neural Networks for NLP*, pages 149–155.

Joost Bastings, Wilker Aziz, and Ivan Titov. 2019. Interpretable neural predictions with differentiable binary variables. In *Proceedings of ACL*, pages 2963–2977.

Samuel R. Bowman, Gabor Angeli, Christopher Potts, and Christopher D. Manning. 2015. A large annotated corpus for learning natural language inference. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 632–642, Lisbon, Portugal. Association for Computational Linguistics.

Jianbo Chen, Le Song, Martin Wainwright, and Michael Jordan. 2018a. Learning to explain: An information-theoretic perspective on model interpretation. In *International Conference on Machine Learning*, pages 883–892. PMLR.

Jianbo Chen, Le Song, Martin J Wainwright, and Michael I Jordan. 2018b. L-shapley and c-shapley: Efficient model interpretation for structured data. *ICLR 2019*.

Alexis Conneau, Douwe Kiela, Holger Schwenk, Loïc Barrault, and Antoine Bordes. 2017. Supervised learning of universal sentence representations from natural language inference data. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 670–680.

Misha Denil, Alban Demiraj, and Nando De Freitas. 2014. Extraction of salient sentences from labelled documents.

Jay DeYoung, Sarthak Jain, Nazneen Fatema Rajani, Eric Lehman, Caiming Xiong, Richard Socher, and Byron C Wallace. 2019. Eraser: A benchmark to evaluate rationalized nlp models. *Computation and Language*.

Irina Higgins, Loic Matthey, Arka Pal, Christopher Burgess, Xavier Glorot, Matthew Botvinick, Shakir Mohamed, and Alexander Lerchner. 2016. beta-vae: Learning basic visual concepts with a constrained variational framework.

Minqing Hu and Bing Liu. 2004. Mining and summarizing customer reviews. In *Proceedings of the tenth ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 168–177.

Alon Jacovi and Yoav Goldberg. 2020. Towards faithfully interpretable nlp systems: How should we define and evaluate faithfulness? In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 4198–4205.

Vineet John, Lili Mou, Hareesh Bahuleyan, and Olga Vechtomova. 2019. Disentangled representation learning for non-parallel text style transfer. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 424–434.

Kaggle. 2016. Us consumer finance complaints. *Kaggle*.

Tao Lei, Regina Barzilay, and Tommi Jaakkola. 2016. Rationalizing neural predictions. In *Proceedings of EMNLP*, pages 107–117.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Roberta: A robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692*.

Scott M Lundberg and Su-In Lee. 2017. A unified approach to interpreting model predictions. *Advances in Neural Information Processing Systems*, 30:4765–4774.

Andrew L Maas, Raymond E Daly, Peter T Pham, Dan Huang, Andrew Y Ng, and Christopher Potts. 2011. Learning word vectors for sentiment analysis. In *Proceedings of ACL*, pages 142–150. Association for Computational Linguistics.

Dong Nguyen. 2018. Comparing automatic and human evaluation of local explanations for text classification. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 1069–1078.

Ankur Parikh, Oscar Täckström, Dipanjan Das, and Jakob Uszkoreit. 2016. A decomposable attention model for natural language inference. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 2249–2255, Austin, Texas. Association for Computational Linguistics.

Marco Tulio Ribeiro, Sameer Singh, and Carlos Guestrin. 2016. Why should i trust you?: Explaining the predictions of any classifier. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 1135–1144. ACM.

Cynthia Rudin. 2018. Please stop explaining black box models for high stakes decisions. *32nd Conference on Neural Information Processing Systems (NIPS 2018), Workshop on Critiquing and Correcting Trends in Machine Learning.*

Lei Sha and Thomas Lukasiewicz. 2021. Multi-type disentanglement without adversarial training. In *Proceedings of the 35th AAAI Conference on Artificial Intelligence*.

Karen Simonyan, Andrea Vedaldi, and Andrew Zisserman. 2013. Deep inside convolutional networks: Visualising image classification models and saliency maps. *arXiv preprint arXiv:1312.6034*.

Mukund Sundararajan, Ankur Taly, and Qiqi Yan. 2017. Axiomatic attribution for deep networks. In *Proceedings of International Conference on Machine Learning (ICML)*, page 3319–3328.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in neural information processing systems*, pages 5998–6008.

Matthew D Zeiler and Rob Fergus. 2014. Visualizing and understanding convolutional networks. In *Proceedings of the European Conference on Computer Vision*, pages 818–833. Springer.

Xiang Zhang, Junbo Zhao, and Yann LeCun. 2015. Character-level convolutional networks for text classification. In *Advances in Neural Information Processing Systems*, pages 649–657.

Peng Zhou, Wei Shi, Jun Tian, Zhenyu Qi, Bingchen Li, Hongwei Hao, and Bo Xu. 2016. Attention-based bidirectional long short-term memory networks for relation classification. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 207–212.

A Average runtime

We evaluate the computation time of each explanation method on two architectures (BILSTM and Transformer). In Table 10, we compare the average runtime of 500 samples in seconds. SHAP and Occlusion remain expensive compared to other techniques. LDAV achieves the lowest time $\sim 1e^{-4}$ as it only requires feeding the input to the model followed by calculating the LDAV scores. We used Tensorflow running on Ubuntu machine with an Intel Core i7 CPU at 3.60 GHz and Nvidia GPU with 6GB in memory.

Model	Methods	DBpedia	Model	Methods	IMDB
BILSTM	IntGrad	8.8	Transformer	IntGrad	9.0
	Occlusion	191.4		Occlusion	252.7
	SHAP	881.4		SHAP	976.6
	ϵ-LRP	1.3		ϵ-LRP	1.5
	Grad*Input	1.4		Grad*Input	1.7
	Saliency	1.6		Saliecy	1.7
	LIME	0.3		LIME	0.4
	LDAV	**0.0001**		**LDAV**	**0.0002**

Table 10: Average runtime for each input in seconds on two architectures: BILSTM (using DBpedia) and Transformer (using IMDB)

B Analysis of learned representations

To see how well LDAVs capture the semantic difference between classes, we analyze the change of the embedding vectors. In other words, we compare between the embedding vectors without learning LDAVs and the embedding vectors after learning LDAVs. To do so, we perform two experiments: one is to project the average of all word embedding vectors ($\hat{\boldsymbol{x}}$) in each input without learning LDAVs into two dimensions using principal component analysis (PCA). The other is to project $\hat{\boldsymbol{x}}$ after learning LDAVs into two dimensions using PCA. The results of the projections on AG news and IMDB are shown in Figures 9 - 12.

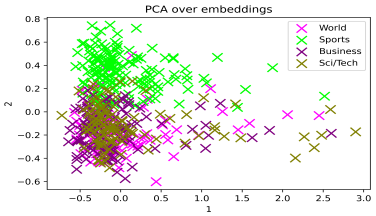

Figure 9: PCA to two dimensions using $\hat{\boldsymbol{x}}$ without employing LDAVs. X-axis and y-axis refer to the principal components (dataset: AG news, Model:BILSTM).

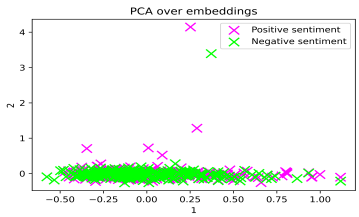

Figure 10: PCA to two dimensions using $\hat{\boldsymbol{x}}$ without employing LDAVs. X-axis and y-axis refer to the principal components (dataset: IMDB, Model:BILSTM).

LDAVs modify the representations of the embedding layer so that they can explain the classifier faithfully. Therefore, we expect the embedding vectors will be changed to better understand the semantic difference between classes while LDAVs are learned. As we can observe in Figures 11 and 12, the embedding vectors of the input texts after LDAVs are learned tend to be clustered collinearly depending on the predicted class. However, in Figures 9 and 10, the embedding vectors have not been clearly clustered when LDAVs are not learned.

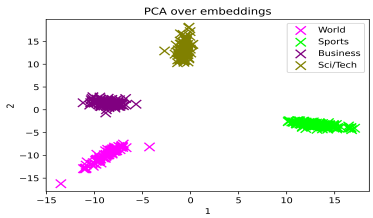

Figure 11: PCA to two dimensions using $\hat{\boldsymbol{x}}$ after employing LDAVs. x-axis and y-axis refer to the principal components (dataset: AG news, Model:BILSTM).

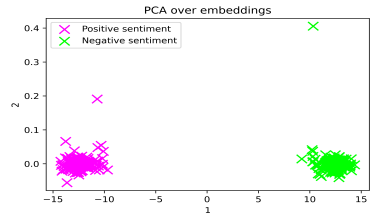

Figure 12: PCA to two dimensions using $\hat{\boldsymbol{x}}$ after employing LDAVs. x-axis and y-axis refer to the principal components (dataset: IMDB, Model:BILSTM).

The intuition here is that the optimization objective with LDAVs (ideally) forces the network to learn embedding representations where inputs of different classes are orthogonal, and inputs belonging to the same class are collinear. As a result, LDAVs can explain the classifier prediction well.

C Performance analysis with respect to distance metric

In this section, we evaluate the effectiveness of using cosine distance over Euclidean distance for

Term 1. We have found that cosine distance works relatively better and it does not sacrifice the performance of the baseline classifier (see Table 11). The performance of the baseline classifier is shown in Table 3 of the main paper. The increased accuracy of cosine distance likely results from inherent normalization during computation and the natural geometric structure it induces (orthogonality and collinearity) on the representations of the embeddings.

	Euclidean distance		**Cosine distance**	
Dataset	**Accuracy**	**F1 score**	**Accuracy**	**F1 score**
IMDB	0.73	0.73	**0.78**	**0.78**
Kaggle-CF	0.73	0.54	**0.79**	**0.66**
AG news	0.86	0.86	0.88	0.88
DBpedia	0.59	0.55	**0.94**	**0.88**

Table 11: Comparing distance metric for Term 1 on the Transformer model.

D Baseline details

Here we describe the baselines used in the evaluation.

Grad*Input is the gradient of the output w.r.t. the input, followed by multiplying the input with the gradient.

Integrated Gradient (IntGrad) calculates a path integral of the model gradient to the input from a non-informative reference point.

Layer-wise relevant propagation ($\epsilon-$LRP) is a layer-wise relevance method, which focuses on redistributing the relevance.

LIME focuses on creating an interpretable classifier by approximating it locally, with a linear model.

SHAP employs game theory to estimate feature attribution.

Saliency uses gradient of the output neuron with respect to the input.

Occlusion employs perturbation techniques to learn feature attribution in a post-hoc approach.

E LDAV score

We found that Euclidean distance in `LDAV` score (Equation 4 in the main paper) works relatively better than cosine distance in approximating feature attribution. The switching points on IMDB using Euclidean and cosine distances are 9% and 15% respectively.

Addressing Leakage in Self-Supervised Contextualized Code Retrieval

Johannes Villmow and **Viola Campos** and **Adrian Ulges** and **Ulrich Schwanecke**
RheinMain University of Applied Sciences
Wiesbaden, Germany
firstname.lastname@hs-rm.de

Abstract

We address contextualized code retrieval, the search for code snippets, helpful to fill gaps in a partial input program. Our approach facilitates a large-scale self-supervised contrastive training by splitting source code randomly into contexts and targets. To combat leakage between the two, we suggest a novel approach based on mutual identifier masking, dedentation, and the selection of syntax-aligned targets. Our second contribution is a new dataset for direct evaluation of contextualized code retrieval, based on a dataset of manually aligned subpassages of code clones. Our experiments demonstrate that the proposed approach improves retrieval substantially, and yields new state-of-the-art results for code clone and defect detection.

1 Introduction

AI-supported software development has recently experienced growing interest (Lu et al., 2021), addressing various code understanding tasks such as code auto-completion (Svyatkovskiy et al., 2020), natural language code search (Husain et al., 2019), and code clone detection (Svajlenko and Roy, 2015). Our focus is on a related task called *contextualized code search* (Mukherjee et al., 2020; Dahal et al., 2022): Given an incomplete piece of code and a certain position of interest (e.g., the current cursor position), a retriever searches for code fragments that are relevant for filling in the missing piece. This setting aligns well with programmers' workflow, and differs substantially from the three tasks mentioned above as follows: (1) In contrast to natural language code search, contextualized code search can exploit local code context. (2) While code generated by autocompletion systems such as GitHub's CodEx (Chen et al., 2021) is prone to subtle programming errors, contextualized search leaves the developer in charge, and the origin of a solution remains transparent. (3) In contrast to clone detection, contextualized code search is not targeted at semantically similar code but code that complements the query.

A key challenge with contextualized code search is that supervised labels for relevant code passages are missing. Therefore, we bootstrap a *self-supervised learning process* by drawing inspiration from Cloze Tasks in natural language processing (Lee et al., 2019): Given a large-scale dataset containing pieces of code in 16 programming languages, we erase random blocks. We refer to these blocks as *targets*, and to their surrounding as *contexts*. Together, these pairs form samples for contrastive learning.

Unfortunately, as Figure 1 shows, this approach suffers from leakage between context and target, as the two share (1) common identifiers, (2) a matching indentation level, and (3) in some languages – if dividing a syntactic primitive such as for-loops – matching brackets. Retrievers might exploit these effects and bypass semantic similarity. To this end, our first contribution is a novel approach towards self-supervised code retrieval, which avoids the above bias through de-leaking steps such as mutual identifier masking and dedentation.

The second challenge we address is evaluation: So far, the focus of evaluating code retrieval systems has been on natural language queries (which can be bootstrapped from docstrings) (Husain et al., 2019). Contextualized code retrieval has been evaluated only indirectly via infilling quality (Lu et al., 2022; Parvez et al., 2021), which reflects the actual retrieval quality poorly. Therefore, our second contribution is a rigorous evaluation of contextualized code retrieval on a manually curated dataset based on aligned code clones. We call this dataset `COCOS` and make it available for future research. On `COCOS`, we demonstrate that retrieval quality benefits substantially from our de-leaking approach. Also, we achieve state-of-the-art results

Proceedings of the 29th International Conference on Computational Linguistics, pages 1006–1013
October 12–17, 2022.

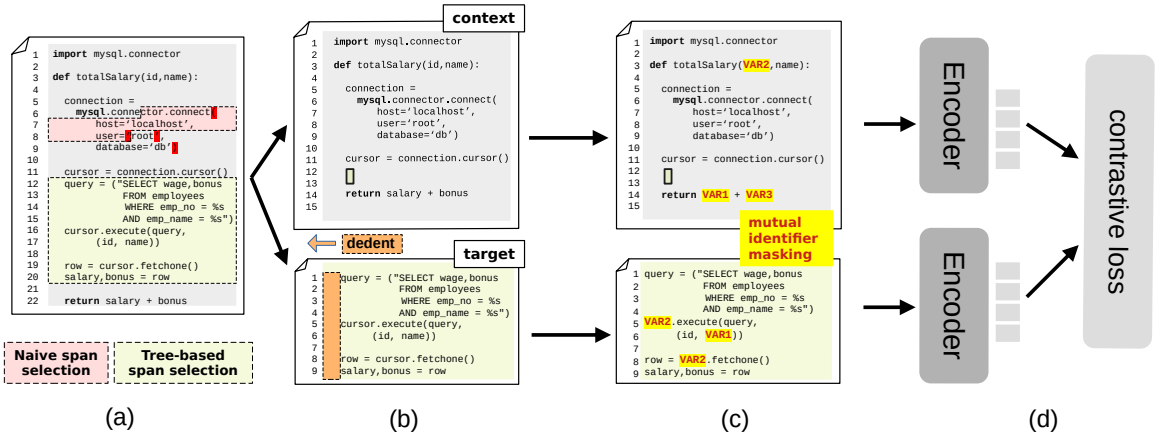

Figure 1: Our approach bootstraps code pairs for contrastive learning by removing target passages (green) from random code contexts (gray). To address leakage between the two – which can be due to matching identifiers, indentation, and brackets – we (a) select the target using the code's syntactic structure, (b) dedent the target (orange arrow), and (c) mutually mask identifiers. Finally, we apply contrastive learning on the resulting code pairs (d).

on the related tasks *code clone* and *defect detection* on CodeXGLUE (Lu et al., 2021).[1]

2 Approach

Given a piece of code as a token sequence $X{=}x_1, \ldots, x_m$, our goal is to boostrap a context-target pair for contrastive learning. The target is a subsequence $Y{=}x_i, \ldots, x_{i+L}$ with $1{\leq}i{\leq}i{+}L{\leq}m$. By replacing this subsequence with a special mask token, we obtain the context $X'{=}x_1, \ldots, x_{i-1}, x_{\text{MASK}}, x_{i+L+1}, \ldots, x_m$. To X' and Y we prepend a programming-language-specific CLS token.

To address the above leakages, we suggest three steps called tree-based span selection (TS), mutual identifier masking (IM) and dedenting (DE).

Tree-based span selection (TS) To select the target Y, we utilize X's concrete syntax tree[2], whose leaves consist of all code tokens. We define the target Y by masking a random *subtree*, which ensures Y to be a syntactically complete piece and avoids leakage due to brackets. Specifically, we first sample the target's length L from a normal distribution with $\mu{=}150$ and $\sigma{=}90$. We then select a node n covering at most L leaves/tokens and iteratively expand the selection, either to n's parent, or by adding n's direct siblings, until reaching the desired size L. Adding siblings allows for multiline targets spanning several statements.

Mutual Identifier Masking (IM) Next, we randomly replace identifiers[3] in X' and Y with special tokens (VAR1, VAR2, ...), to minimize leakage between identifiers. To preserve as much lexical information as possible, we mask only *mutual* identifiers present in both context and target. We hide 90% of those mutual identifiers randomly *either* in the context *or* in target code. For 5% of context-target pairs, we omit identifier masking altogether.

Dedenting (DE) Finally, in 90% of the training samples, we determine the indentation level of the target Y and dedent it, so that it has indentation level zero and the retriever cannot bypass semantic similarity by focusing on targets at the same indentation level.

2.1 Contrastive Training

We encode context code X' and target Y with the same transformer encoder and obtain sequence embeddings $\mathbf{q}, \mathbf{k} \in \mathbb{R}^d$, using the encoding of the CLS token. Following Wang et al. (2021b), we pretrain the transformer with alternating generation tasks identifier masking and span prediction[4] and use the pre-trained encoder.

The retriever is then trained by optimizing the following contrastive InfoNCE loss (van den Oord et al., 2018) with in-batch negative samples, where

[1] We release dataset, code and checkpoints to our experiments under github.com/villmow/coling-cocos

[2] We use the tree-sitter library for parsing.

[3] What is considered an identifier is defined in the grammar of a tree-sitter parser and varies between programming languages, i.e. we do not differentiate between variables, method names or method calls.

[4] Contrary to Wang et al. (2021b) we omit identifier detection and instead use our tree-based span selection to generate large and small spans.

f is the cosine similarity, K the amount of sequences in our batch, and τ=0.1 the temperature.

$$\mathcal{L}_\Theta = -log\frac{\exp(f(\mathbf{q},\mathbf{k}^+)/\tau))}{\sum_{i=0}^{K-2}\exp(f(\mathbf{q},\mathbf{k}_i^-)/\tau))} \quad (1)$$

To obtain harder negative samples – which have been found crucial for good retriever training (Ren et al., 2021) – we form batches only with samples from the same programming language.

3 Dataset

In this section, we first describe the large-scale data which our retriever is trained on. Second, we outline COCOS, a new bechmark we propose for contextualized code retrieval.

Pre-training Dataset Our self-supervised code retrieval model is pre-trained on 33M files in 16 programming languages (see Appendix A). As code files tend to be large, we truncate them using tree-based span selection (cmp. Section 2): Starting from a whole file, we randomly select sufficiently large spans of code (length between 150 and 800 tokens). We remove those segments from the original file and feed the shortened file as well as all individual segments as inputs X into the learning process described in Section 2. A special identifier (similar to code folding in an IDE) marks those positions in the original file where segments have been removed.

COCOS Evaluating contextualized code retrieval models is hard because little or no suitable evaluation data is available to indicate which sub-blocks in the code implement the same functionality. To address this gap, we have created a new dataset based on BigCloneBench (Svajlenko and Roy, 2015), a Java code clone dataset that provides pairs of semantically similar functions. Given a function in BigCloneBench, we manually select a sub-passage modeling a particular target functionality (e.g. extracting a zip file). We then label which lines in the function's clones match this functionality (see Listings 1 - 3 in the appendix). Based on these targets and their surrounding contexts, we evaluate how well a model retrieves targets implementing the same functionality in code clones. We manually gather 606 context-target pairs implementing 31 randomly selected functionalities. Finally, we add $10k$ non-relevant disctractor snippets by randomly sampling top-level statements from method bodies in CodeSearchNet (Husain et al., 2019). We call the dataset COCOS (**Co**ntextualized **Co**de **S**earch).

Model Features	MAP	NDCG	P@1	P@3	P@10
BM25 standard	12.36	43.8	27.89	24.92	17.13
BM25 camel	27.95	57.11	39.44	37.07	33.17
None	15.65	49.85	45.87	37.95	24.77
TS	26.47	59.64	58.09	50.77	36.96
TS, IM	33.78	66.03	69.80	60.95	45.33
TS, DE	36.32	65.94	59.41	54.57	44.39
TS, IM, DE	**50.87**	**76.28**	**73.60**	**70.30**	**59.70**

Table 1
Zeroshot code retrieval results for different de-leaking steps as described in Section 2: Tree-based span selection (TS); mutual identifier masking (IM); dedenting (DE). We report non-neural results for BM25 (Jones et al., 2000) using the Elasticsearch standard tokenizer (standard) and a tokenizer that splits on camel case (camel).

4 Evaluation

We report results for zero-shot code retrieval on COCOS and for two similar code understanding tasks from CodeXGlue (Lu et al., 2021), namely code clone detection and code defect detection. For all experiments, we report test results of the model with the highest mean reciprocal rank (MRR) on $30K$ held-out validation samples of the pre-training dataset.

4.1 Zero-shot Code Retrieval

We evaluate our models in a zero-shot setting, i.e. no fine-tuning on COCOS was applied. For each context all possible targets and the $10k$ distractor snippets are ranked, excluding the original target. To assess the proposed approaches, we compare variants of our model trained with different de-leaking steps and Okapi BM25 (Jones et al., 2000) as non-neural baseline. BM25 is evaluated using the standard Elasticsearch tokenization and a tokenizer splitting on camel case which is more suitable for source code. Table 1 reports our ablation studies showing mean average precision (MAP), normalized discounted cumulative gain (NDCG) and precision at k. We found the baseline trained without de-leaking to retrieve only samples with similar identifiers but to fail to consistently retrieve all relevant targets. Using all de-leaking steps significantly outperforms all ablations. Figure 2 also illustrates for a random selection of samples that our approach forms better clusters for both contexts and targets in embedding space.

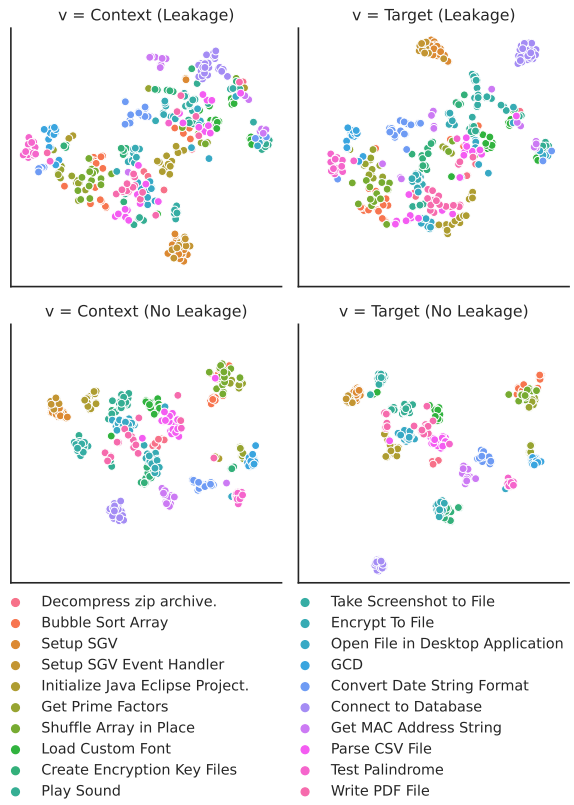

Figure 2: t-SNE comparison between the embeddings of the baseline model with leakage (top) and our model with leakage reduction steps applied (bottom). It can be seen that our approach forms better clusters.

4.2 Clone Detection and Defect Detection

We evaluate our model on clone detection on the `POJ-104` dataset (Mou et al., 2016), which consists of C and C++ programs for 104 problems from an open programming platform (OJ). We follow the evaluation procedure of CodeXGlue and report mean average precision (MAP@R) with R=499.

Finally for defect detection we evaluate on the `Devign` dataset (Zhou et al., 2019), which consists of vulnerable C functions manually collected from open source projects. The task is to predict whether the function is vulnerable. Following CodeXGlue we report accuracy. Baseline results for RoBERTa (Liu et al., 2019), CodeBERT (Feng et al., 2020), code2vec (Alon et al., 2019) and CoTexT (Phan et al., 2021) are reported in Lu et al. (2021), results for PLBART (Ahmad et al., 2021), GraphCodeBERT (Guo et al., 2021), SynCoBERT (Wang et al., 2021a) and CodeT5 (Wang et al., 2021b) are reproduced from Wang et al. (2021a) and Wang et al. (2021b). We find that our model outperforms state-of-the-art on both tasks by a large margin.

Model	Clone	Defect
	MAP@R	Accuracy
RoBERTa (code)	76.67	61.05
CodeBERT	82.67	62.08
code2vec	1.98	62.48
PLBART	-	63.18
GraphCodeBERT	85.16	63.21
SynCoBERT	88.24	64.50
CodeT5	-	65.78
CoTexT	-	66.62
Ours	**91.34**	**69.33**

Table 2
Results on code clone and defect detection (`POJ-104` and `Devign` dataset). We report results from Wang et al. (2021a) and Wang et al. (2021b).

5 Related Work

Code Representation Learning Given the success of pre-trained language models in NLP, recent work has extended pre-training to program syntax. Kanade et al. (2020) and Feng et al. (2020) train a BERT encoder on source code using masked language modeling. Guo et al. (2021) propose GraphCodeBERT to incorporate structural information such as data flow. Besides these encoder models, other work has pre-trained decoders (CodeGPT (Svyatkovskiy et al., 2020), CugLM (Liu et al., 2020)) or encoder-decoders (PLBART (Ahmad et al., 2021), CodeT5 (Wang et al., 2021b)) on pairs of natural language and program code. SynCoBERT (Wang et al., 2021a) is trained on various pre-training tasks on multimodal data, including code, comment and Abstract Syntax Tree (AST) representations. Guo et al. (2022) propose UniXcoder, which takes a similar approach but employs an encoder-decoder architecture instead of a single encoder.

In most of the above work, multiple modalities have been applied, e.g. code and natural language comments. In contrast to contextual code search, this setup does not come with leakage, which is the main concern of our work.

Contextualized Code Search retrieves complementary code, given a code context and sometimes an additional natural language query. Non-neural approaches include FaCoY (Kim et al.,

2018), which extends the query with related code from StackOverFlow, and Siamese (Ragkhitwetsagul and Krinke, 2019), which combines multiple code representations for pure code-to-code search. Aroma (Luan et al., 2019) clusters candidate code and intersects the snippets in each cluster to recommend likely subsequent code for a given snippet. Mukherjee et al. (2020) address the task by decompiling code fragments into a simpler representation called SKETCH (Murali et al., 2018) to learn a statistical model. The neural approach SCOTCH (Dahal et al., 2022) finetunes a CodeBERT model to discover relevant methods for queries combined with surrounding source code. None of the above approaches address the issue of leakage, either because they are non-neural (FaCoY, Siamese, Aroma), or leakage is neglected because the respective approach operates on method level (SCOTCH).

The issue of leakage in code search has only been scarcely studied before: Jain et al. (2021) propose ContraCode, a contrastive neural model that allows to retrieve code clones. To generate samples for contrastive learning, they augment code snippets using compiler-based semantic-preserving code transformations. Lu et al. (2022) propose ReACC, which uses partial code as search query in the context of retrieval-augmented code completion. To combat leakage, they insert dead code and rename variables. Compared to these approaches, our steps towards leakage reduction are much simpler. UniXcoder (Guo et al., 2022) pre-trains code representations using a variety of tasks, including contrastive learning. A positive sample pair is generated by running the same code piece through a transformer under dropout, which is a known trick for natural language (Gao et al., 2021) and can be seen as a simple form of de-leaking. Note that – since all our transformer encoders apply dropout during training – this mechanism applies for all models in our study too.

6 Conclusion

We have proposed a new approach towards unsupervised code retrieval, which reduces leakage between randomly drawn targets and their contexts. We also contribute a dataset COCOS, on which we demonstrate via ablations that leakage reduction is crucial for an efficient training. Our approach also yields competitive representations for related tasks, as demonstrated by new state-of-the-art results on clone and defect detection. An interesting future direction will be to combine our retriever with generators for a combined, unsupervised training.

Acknowledgements

This work has been supported by the Federal State of Hesse / Germany, Program LOEWE 5 (Project "Code Buddy").

References

Wasi Ahmad, Saikat Chakraborty, Baishakhi Ray, and Kai-Wei Chang. 2021. Unified pre-training for program understanding and generation. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 2655–2668, Online. Association for Computational Linguistics.

Uri Alon, Meital Zilberstein, Omer Levy, and Eran Yahav. 2019. Code2vec: Learning distributed representations of code. *Proc. ACM Program. Lang.*, 3(POPL).

Mark Chen, Jerry Tworek, Heewoo Jun, Qiming Yuan, Henrique Ponde de Oliveira Pinto, Jared Kaplan, Harrison Edwards, Yuri Burda, Nicholas Joseph, Greg Brockman, Alex Ray, Raul Puri, Gretchen Krueger, Michael Petrov, Heidy Khlaaf, Girish Sastry, Pamela Mishkin, Brooke Chan, Scott Gray, Nick Ryder, Mikhail Pavlov, Alethea Power, Lukasz Kaiser, Mohammad Bavarian, Clemens Winter, Philippe Tillet, Felipe Petroski Such, Dave Cummings, Matthias Plappert, Fotios Chantzis, Elizabeth Barnes, Ariel Herbert-Voss, William Hebgen Guss, Alex Nichol, Alex Paino, Nikolas Tezak, Jie Tang, Igor Babuschkin, Suchir Balaji, Shantanu Jain, William Saunders, Christopher Hesse, Andrew N. Carr, Jan Leike, Joshua Achiam, Vedant Misra, Evan Morikawa, Alec Radford, Matthew Knight, Miles Brundage, Mira Murati, Katie Mayer, Peter Welinder, Bob McGrew, Dario Amodei, Sam McCandlish, Ilya Sutskever, and Wojciech Zaremba. 2021. Evaluating large language models trained on code. *CoRR*, abs/2107.03374.

Samip Dahal, Adyasha Maharana, and Mohit Bansal. 2022. Scotch: A semantic code search engine for IDEs. In *Deep Learning for Code Workshop*.

Zhangyin Feng, Daya Guo, Duyu Tang, Nan Duan, Xiaocheng Feng, Ming Gong, Linjun Shou, Bing Qin, Ting Liu, Daxin Jiang, and Ming Zhou. 2020. CodeBERT: A pre-trained model for programming and natural languages. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 1536–1547, Online. Association for Computational Linguistics.

Tianyu Gao, Xingcheng Yao, and Danqi Chen. 2021. SimCSE: Simple contrastive learning of sentence embeddings. In *Proceedings of the 2021 Conference*

on Empirical Methods in Natural Language Processing, pages 6894–6910, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Daya Guo, Shuai Lu, Nan Duan, Yanlin Wang, Ming Zhou, and Jian Yin. 2022. UniXcoder: Unified cross-modal pre-training for code representation.

Daya Guo, Shuo Ren, Shuai Lu, Zhangyin Feng, Duyu Tang, Shujie Liu, Long Zhou, Nan Duan, Alexey Svyatkovskiy, Shengyu Fu, Michele Tufano, Shao Kun Deng, Colin B. Clement, Dawn Drain, Neel Sundaresan, Jian Yin, Daxin Jiang, and Ming Zhou. 2021. GraphCodeBERT: Pre-training code representations with data flow. In *9th International Conference on Learning Representations, ICLR 2021, Virtual Event, Austria, May 3-7, 2021*. OpenReview.net.

Hamel Husain, Ho-Hsiang Wu, Tiferet Gazit, Miltiadis Allamanis, and Marc Brockschmidt. 2019. CodeSearchNet Challenge: Evaluating the state of semantic code search. *CoRR*, abs/1909.09436.

Paras Jain, Ajay Jain, Tianjun Zhang, Pieter Abbeel, Joseph Gonzalez, and Ion Stoica. 2021. Contrastive code representation learning. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics.

K. Sparck Jones, S. Walker, and S. E. Robertson. 2000. A probabilistic model of information retrieval: Development and comparative experiments. *Inf. Process. Manage.*, 36(6):779–808.

Aditya Kanade, Petros Maniatis, Gogul Balakrishnan, and Kensen Shi. 2020. Learning and evaluating contextual embedding of source code. In *Proceedings of the 37th International Conference on Machine Learning*, volume 119 of *Proceedings of Machine Learning Research*, pages 5110–5121. PMLR.

Kisub Kim, Dongsun Kim, Tegawendé F. Bissyandé, Eunjong Choi, Li Li, Jacques Klein, and Yves Le Traon. 2018. FaCoY: A code-to-code search engine. In *Proceedings of the 40th International Conference on Software Engineering*, ICSE '18, page 946–957, New York, NY, USA. Association for Computing Machinery.

Kenton Lee, Ming-Wei Chang, and Kristina Toutanova. 2019. Latent retrieval for weakly supervised open domain question answering. In *Proceedings of the 57th Conference of the Association for Computational Linguistics, ACL 2019, Florence, Italy, July 28- August 2, 2019, Volume 1: Long Papers*, pages 6086–6096. Association for Computational Linguistics.

Fang Liu, Ge Li, Yunfei Zhao, and Zhi Jin. 2020. Multi-task learning based pre-trained language model for code completion. In *Proceedings of the 35th IEEE/ACM International Conference on Automated Software Engineering*, ASE '20, page 473–485, New York, NY, USA. Association for Computing Machinery.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Roberta: A robustly optimized BERT pretraining approach. *CoRR*, abs/1907.11692.

Shuai Lu, Nan Duan, Hojae Han, Daya Guo, Seung-won Hwang, and Alexey Svyatkovskiy. 2022. ReACC: A retrieval-augmented code completion framework. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 6227–6240, Dublin, Ireland. Association for Computational Linguistics.

Shuai Lu, Daya Guo, Shuo Ren, Junjie Huang, Alexey Svyatkovskiy, Ambrosio Blanco, Colin B. Clement, Dawn Drain, Daxin Jiang, Duyu Tang, Ge Li, Lidong Zhou, Linjun Shou, Long Zhou, Michele Tufano, Ming Gong, Ming Zhou, Nan Duan, Neel Sundaresan, Shao Kun Deng, Shengyu Fu, and Shujie Liu. 2021. CodeXGLUE: A machine learning benchmark dataset for code understanding and generation. *ArXiv*, abs/2102.04664.

Sifei Luan, Di Yang, Celeste Barnaby, Koushik Sen, and Satish Chandra. 2019. Aroma: Code recommendation via structural code search. *Proc. ACM Program. Lang.*, 3(OOPSLA).

Lili Mou, Ge Li, Lu Zhang, Tao Wang, and Zhi Jin. 2016. Convolutional neural networks over tree structures for programming language processing. In *Proceedings of the Thirtieth AAAI Conference on Artificial Intelligence*, AAAI'16, page 1287–1293. AAAI Press.

Rohan Mukherjee, Chris Jermaine, and Swarat Chaudhuri. 2020. Searching a database of source codes using contextualized code search. *Proc. VLDB Endow.*, 13(10):1765–1778.

Vijayaraghavan Murali, Letao Qi, Swarat Chaudhuri, and Chris Jermaine. 2018. Neural sketch learning for conditional program generation. In *International Conference on Learning Representations*.

Md Rizwan Parvez, Wasi Ahmad, Saikat Chakraborty, Baishakhi Ray, and Kai-Wei Chang. 2021. Retrieval augmented code generation and summarization. In *Findings of the Association for Computational Linguistics: EMNLP 2021*, pages 2719–2734, Punta Cana, Dominican Republic. Association for Computational Linguistics.

Long N. Phan, Hieu Tran, Daniel Le, Hieu Nguyen, James T. Anibal, Alec Peltekian, and Yanfang Ye. 2021. Cotext: Multi-task learning with code-text transformer. *CoRR*, abs/2105.08645.

Chaiyong Ragkhitwetsagul and Jens Krinke. 2019. Siamese: Scalable and incremental code clone search via multiple code representations. *Empirical Softw. Engg.*, 24(4):2236–2284.

Ruiyang Ren, Yingqi Qu, Jing Liu, Wayne Xin Zhao, QiaoQiao She, Hua Wu, Haifeng Wang, and Ji-Rong Wen. 2021. RocketQAv2: A joint training method for dense passage retrieval and passage re-ranking. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 2825–2835, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Jeffrey Svajlenko and Chanchal K. Roy. 2015. Evaluating clone detection tools with bigclonebench. In *2015 IEEE International Conference on Software Maintenance and Evolution (ICSME)*, pages 131–140.

Alexey Svyatkovskiy, Shao Kun Deng, Shengyu Fu, and Neel Sundaresan. 2020. *IntelliCode Compose: Code Generation Using Transformer*, page 1433–1443. Association for Computing Machinery, New York, NY, USA.

Aäron van den Oord, Yazhe Li, and Oriol Vinyals. 2018. Representation learning with contrastive predictive coding. *CoRR*, abs/1807.03748.

Xin Wang, Yasheng Wang, Fei Mi, Pingyi Zhou, Yao Wan, Xiao Liu, Li Li, Hao Wu, Jin Liu, and Xin Jiang. 2021a. Syncobert: Syntax-guided multi-modal contrastive pre-training for code representation.

Yue Wang, Weishi Wang, Shafiq Joty, and Steven C.H. Hoi. 2021b. CodeT5: Identifier-aware unified pre-trained encoder-decoder models for code understanding and generation. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 8696–8708, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Yaqin Zhou, Shangqing Liu, Jingkai Siow, Xiaoning Du, and Yang Liu. 2019. Devign: Effective vulnerability identification by learning comprehensive program semantics via graph neural networks. In *Proceedings of the 33rd International Conference on Neural Information Processing Systems*, Red Hook, NY, USA. Curran Associates Inc.

A Pre-training Dataset Details

We crawl 237k active GitHub repositories with more than 10 stars[5] and perform per file deduplication. We keep files in programming languages for which a tree-sitter parser is available (16 languages). The resulting dataset is shown in Table 3 and consists of $\approx 33M$ code files in 16 programming languages. We select 570 repositories for validation.

Language	Training	Valid	Total
Java	7,345,753	8,434	7,354,187
JavaScript	4,471,689	14,134	4,485,823
C++	3,734,357	1,698	3,736,055
Python	3,016,545	4,718	3,021,263
C#	2,843,642	570	2,844,212
TypeScript	2,299,964	2,392	2,302,356
C	2,242,379	781	2,243,160
PHP	2,206,063	4,648	2,210,711
Go	1,759,600	129	1,759,729
Ruby	1,068,668	3,397	1,072,065
Rust	366,891	54	366,945
CSS	349,525	2,579	352,104
Scala	273,822	1,198	275,020
Haskell	114,311	177	114,488
OCaml	55,838	0	55,838
Julia	34,403	29	34,432

Table 3
Number of files in unsupervised pre-training dataset.

B Training Details

On all models and tasks we use the AdamW optimizer and linearly increase the learning rate for 10% of the training steps, along with a polynomial decay for the remaining steps. We train our unsupervised models for 500k steps on a single A6000 GPU, with a peak learning rate of 0.0001 and use a dynamic batch size so that batches contain around 7000 tokens.

For clone and defect detection we fine-tune our model on the respective training set. Following Wang et al. (2021b) we run a brief sweep over learning rate, batch size and number of epochs and report results of the model with highest validation score, using the published evaluation code. We release our code including precise hyperparameter configs under github.com/villmow/coling-cocos.

[5]We consider a repository as active if there has been a pull request between 04/21 and 09/21.

```
public boolean extract(File f, String folder) {
    Enumeration entries;
    ZipFile zipFile;
    try {
        zipFile = new ZipFile(f);
        entries = zipFile.getEntries();
        [MASK]
        zipFile.close();
    } catch (IOException ioe) {
        this.errMsg = ioe.getMessage();
        Malgn.errorLog(
            "{Zip.unzip} " + ioe.getMessage()
        );
        return false;
    }
    return true;
}
```

Listing 1: Incomplete and masked query X' from our COCOS dataset. The [MASK] token denotes the current position of interest (cursor). Code that extracts elements from a zip file needs to be found.

```
while (entries.hasMoreElements()) {
    ZipArchiveEntry entry =
        (ZipArchiveEntry) entries.nextElement();
    if (entry == null) continue;
    String path = folder + "/"
        + entry.getName().replace('\\', '/');
    if (!entry.isDirectory()) {
        File destFile = new File(path);
        String parent = destFile.getParent();
        if (parent != null) {
            File parentFile = new File(parent);
            if (!parentFile.exists()) {
                parentFile.mkdirs();
            }
        }
        copyInputStream(
            zipFile.getInputStream(entry),
            new BufferedOutputStream(
                new FileOutputStream(destFile)
            )
        );
    }
}
```

Listing 2: The masked section Y manually selected from X (Listing 1). It has been re-formatted for better readability. Note that we omit Y from the result list for query X during evaluation.

```
ArchiveEntry ae = zis.getNextEntry();
while(ae != null) {
    //Resolve new file
    File newFile = new File(
        outputdir + File.separator + ae.getName()
    );

    //Create parent directories if not exists
    if(!newFile.getParentFile().exists())
        newFile.getParentFile().mkdirs();

    if(ae.isDirectory()) { //create if not exists
        if(!newFile.exists())
            newFile.mkdir();
    } else { //If file, write file
        FileOutputStream fos = new FileOutputStream(
            newFile);
        int len;
        while((len = zis.read(buffer)) > 0) {
            fos.write(buffer, 0, len);
        }
        fos.close();
    }

    //Proceed to the next entry in the zip file
    ae = zis.getNextEntry();
}
```

Listing 3: Possible solution, that implements the same functionality as target in Listing 2.

A Domain Knowledge Enhanced Pre-Trained Language Model for Vertical Search: Case Study on Medicinal Products

Kesong Liu* and **Jianhui Jiang** and **Feifei Lyu**
Alibaba Group
{kesong.lks,huidong.jjh,lvfeifei.lff}@alibaba-inc.com

Abstract

We present a biomedical knowledge enhanced pre-trained language model for medicinal product vertical search. Following ELECTRA's replaced token detection (RTD) pre-training, we leverage biomedical entity masking (EM) strategy to learn better contextual word representations. Furthermore, we propose a novel pre-training task, *product attribute prediction* (PAP), to inject product knowledge into the pre-trained language model efficiently by leveraging medicinal product databases directly. By sharing the parameters of PAP's transformer encoder with that of RTD's main transformer, these two pre-training tasks are jointly learned. Experiments demonstrate the effectiveness of PAP task for pre-trained language model on medicinal product vertical search scenario, which includes query-title relevance, query intent classification, and named entity recognition in query.

1 Introduction

Pre-trained language models (PLMs) have significantly improved the performance of various natural language processing (NLP) tasks in the recent years. It is now a common practice to adapt the pretrain-then-finetune approach in NLP. PLMs such as BERT (Devlin et al., 2019) and ELECTRA (Clark et al., 2020) capturing word meaning through self-supervised learning from large corpus have shown a significant improvement on various text mining tasks. In the biomedical domain, many BERT variants such as BioBERT (Lee et al., 2020), SciBERT (Beltagy et al., 2019), PubMedBERT (Gu et al., 2021) and BioMedBERT (Chakraborty et al., 2020), follow either continual pre-training or pre-training from scratch approach using domain specific corpora to further improve the model performance.

Our motivation is to apply powerful pre-trained language models on medicinal product vertical search engine to solve query understanding and query title relevance tasks. Most of the models trained on either the general corpus or the medical literature corpus lack medicinal product knowledge. For the product vertical search scenario, medicinal product information is usually stored in structured relational database tables, and traditional pre-trained language models focus on natural language text in the form of sentences without considering the semantic relationship modeling of structured text in the product information tables. From the perspective of users' search habits, in addition to searching directly for the medicine name, users also often search for disease, symptoms and other important proudct attribute words to find the medicine. Therefore, we propose a novel pre-training task called *product attribute prediction* (PAP). Although we could carefully craft a medicinal product knowledge graph from product databases and then try to inject the extracted explicit knowledge graph into the pre-trained language model, we argue that the procedure of building product knowledge graph is laborious but avoidable by training the language model directly on product structural information.

In this paper, we propose a novel ELECTRA(Clark et al., 2020) based biomedical knowledge enhanced pre-trained language model. It consists of two pre-training tasks: replaced token detection and product attribute prediction. Our approach is inspired by ELECTRA and TransE(Bordes et al., 2013) methods but distinguish itself in two prominent ways. Firstly, we use entity masking strategy for biomedical text instead of only masking random tokens. Text spans of biomedical named entities are masked dynamically before each training iteration. We let the model predict whether these terms are replaced to incorporate biomedical domain knowledge. Secondly, we utilize medicinal product textual information instead of node identifiers in product knowledge graphs, to further bring rich medicinal product knowledge into the

*Corresponding author

Proceedings of the 29th International Conference on Computational Linguistics, pages 1014–1023
October 12–17, 2022.

pre-trained language model. The triples (product title, attribute name, attribute values), which can be easily drawn from medicinal product databases, are all encoded and then used in the contrastive loss of PAP pre-training task to capture product knowledge.

Our main contributions can be summarized as follows: 1) We augment ELECTRA's replaced token detection pre-training task by leveraging biomedical entities masking (EM) to learn better contextual word representation; 2) We propose a novel pre-training task, product attribute prediction (PAP), which can inject medicinal product knowledge into the pre-trained language model by exploiting medicinal product databases directly. The proposed pre-training task is also applicable to vertical search scenarios for products in general, not limited to medicinal products; and 3) We have demonstrated the effectiveness of PAP pre-training task for PLMs in medicinal product vertical search scenario.[1]

2 Related Work

2.1 Pre-trained Language Model

Recently pre-trained language models have dominated many NLP tasks by pre-training on a large corpus of text followed by fine-tuning on a specific task. ELMo (Peters et al., 2018) learns the contextual representations based on a bidirectional language model (biLM) with forward and backward LSTM layers. GPT (Radford et al., 2018) as an effective pre-trained generative model predicts the next token based on the left-hand side context by adapting the transformer. GPT-2 (Radford et al., 2019) brings task information to the pre-training process and adopt the model to zero-shot tasks. GPT-3 (Brown et al., 2020) further improves task-agnostic, few-shot performance and produce human-like texts. BERT (Devlin et al., 2019) presents a bi-directional LM to predict the masked tokens and demonstrates strong performance on a wide range of NLP benchmarks. RoBERTa (Liu et al., 2019) shows that more careful parameter tuning on more data can benefit PLMs. ALBERT (Lan et al., 2019) uses weight sharing and embedding factorization to reduce memory consumption and improve training speed. XLNet (Yang et al., 2019) as a permutation language model predicts masked tokens in a permuted order in a auto-regressive way. T5 (Raffel et al., 2020) and BART (Lewis et al., 2020a) adopts denoising sequence-to-sequence pre-training method. ELECTRA (Clark et al., 2020) introduce a more sample-efficient pre-training task called replaced token detection, which is replacing some tokens with plausible alternatives and predicting whether each token was replaced or not. MacBERT (Cui et al., 2020) adopts MLM as correction (Mac) and achieve state-of-the-art performances on several Chinese NLP tasks.

2.2 Knowledge Enhanced and Domain Specific Pre-trained Language Model

ERNIE (Zhang et al., 2019b) utilize pre-processed knowledge embeddings of entity mentions in text. KnowBert (Peters et al., 2019) uses retrieved relevant entity embeddings and word-to-entity attention to update contextual word representations. K-ADAPTER (Wang et al., 2021b) integrates knowledge into PLM with neural adapters. E-BERT (Poerner et al., 2020) adds aligned entity embeddings into BERT without additional pre-training. Joint representation learning of words and entities (Zhang et al., 2019a, 2021) leverage external Knowledge Graphs. BioBERT (Lee et al., 2020) is pre-trained on PubMed and PubMed Central articles. SciBERT (Beltagy et al., 2019), PubMedBERT (Gu et al., 2021) and Bio-LM (Lewis et al., 2020b) have shown that pre-training from scratch with domain-specific data can improve performance. KeBioLM (Yuan et al., 2021) and UmlsBERT (Michalopoulos et al., 2021) leverage UMLS knowledge bases during the pre-training. Domain specific pre-training (Wang et al., 2021d) has also been employed for biomedical literature search problems. Moreover, (Wang et al., 2021a) gives a systematic survey for biomedical domain PLMs.

For Chinese medical text mining, MC-BERT (Zhang et al., 2020) introduces a conceptualized representation learning approach for Chinese biomedical corpora and a Chinese Biomedical Language Understanding Evaluation benchmark (ChineseBLUE). EMBERT (Cai et al., 2021) is an entity-level knowledge-enhanced pre-trained language model, which leverages several distinct self-supervised tasks. BioHanBERT (Wang et al., 2021c), as a hanzi-aware PLM, utilizes component-level internal semantic information of Chinese characters to enhance the semantics of Chinese biomedical concepts and terminologies.

[1] Our code and models are publicly available on Github: https://github.com/liuks/ep_plm

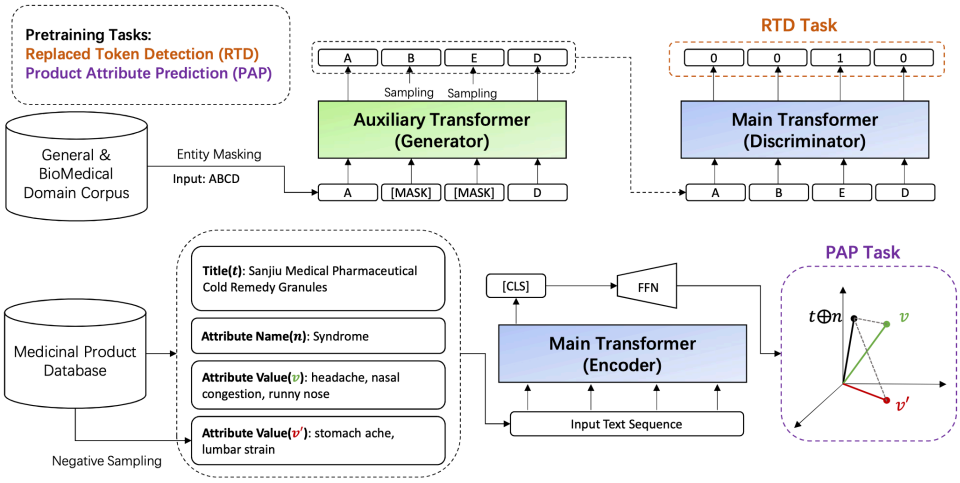

Figure 1: The overview of our pre-training model architecture. The auxiliary transformer is pre-trained by masked language model with biomedical entity masking. The corrupted text is then used as the main transformer's input in replaced token detection task. The same main transformer encodes medicinal product title, attribute name, attribute value and phrase level negative sampled attribute value, respectively. These encoded embeddings are then used in product attribute prediction task.

3 Method

To augment the pre-trained language model with biomedical domain knowledge, we follow ELECTRA's replaced token detection (RTD) pre-training task on general domain and biomedical specific corpora, and further introduce dynamic biomedical entity word masking to learn better contextual word representations. Although it is possible to use other pre-training tasks such as the masked token prediction in BERT, we chose the ELECTRA's pre-training task due to the sample efficiency of the RTD task and the comparable performance of ELECTRA-Base and BERT-large models(Clark et al., 2020). For better application to the vertical search scenario of medicinal product search, it is desirable that the pre-trained language model contains product-related knowledge. Considering that important product attributes such as drug names, diseases and symptoms are common search terms and the structured information of products also contains these terms, so we propose the product attribute prediction pre-training task to model the semantic relationships of product attributes.

As shown in Figure 1, our approach consists of two pre-training tasks: replaced token detection and product attribute prediction.

3.1 Replaced Token Detection with Biomedical Entity Masking

Given an input text sequence $\mathbf{x} = (x_1, x_2, \ldots, x_m)$ with m tokens, a text span sequence $\mathbf{s} = (s_1, s_2, \ldots, s_n)$ with n span units is produced by applying Chinese word segment and biomedical name entity recognition. These text span units are then randomly replaced with an equal length `[MASK]` tokens to create $\mathbf{x}^{\text{mask}}$ with about 15% tokens masked out, e.g. selecting s_j and replacing every token x_i for $x_i \in s_j$ with the `[MASK]` token.

The masked sequence $\mathbf{x}^{\text{mask}}$ is then input to the auxiliary transformer (generator) to produce a corrupt sequence $\mathbf{x}^{\text{corrupt}}$ by sampling new tokens according to $\hat{x}_i \sim p_G(x_i|x^{\text{mask}})$ for i in all masked positions. Comparing the original text and the corrupt text gives the supervisory signal label sequence $\mathbf{L} = (l_1, l_2, \ldots, l_m)$, where $l_i = \mathbb{1}(x_i, x_i^{\text{corrupt}})$. The main transformer (discriminator) predicts whether tokens are replaced or not. For replaced token detection task, the loss function is:

$$L_{\text{RTD}} = -\sum_{\mathbf{x}} \Big(\log p_G(x^{\text{corrupt}}|x^{\text{mask}}) + \lambda \log p_D(x^{\text{corrupt}}, \mathrm{x}) \Big) \quad (1)$$

where p_G is generating token probability for masked-out positions in the generator network; p_D is replaced probability for all position tokens in the discriminator network; λ is the hyperparameter for balancing these two network losses.

Different from whole word masking with WordPiece in BERT, we not only randomly mask whole words but also biomedical entities including drug name, chemical name, disease, syndrome, efficacy words and so on, which can explicitly inject

biomedical domain knowledge to the pre-trained language model.

3.2 Product Attribute Prediction Task

Let t, n, v be a medicinal product title, attribute name and attribute value respectively. We encode the text description of product tile t, attribute name n and attribute value v using the same main transformer (encoder) E to obtain text representations. Inspired by the TransE (Bordes et al., 2013) and RotateE (Sun et al., 2019) models, we define the distance function of the triples according to Equation (2) and PAP loss function according to Equation (3) to push the projection of concatenated title and attribute name representation $E(t \oplus n)$ near that of attribute value representation $E(v)$ but far away from negative sampled attribute value representation $E(v')$. The projection denoted as f is implemented as a feed forward neural network layer.

$$d(t, n, v) = \|f(E(t \oplus n)) - f(E(v))\| \quad (2)$$

$$\begin{aligned} L_{\text{PAP}} = &- \log \sigma(\gamma - d(t, n, v)) \\ &- \sum_{v'} \frac{1}{k} \log \sigma(d(t, n, v') - \gamma) \end{aligned} \quad (3)$$

where $\gamma > 0$ is a margin hyperparameter; σ is the sigmoid function; v' denotes the negative randomly sampled attribute value; k is the number of negative values v' in the summation, and is chosen to be twice the number of the positive attribute values. It is worth noting that the representation of $E(t \oplus n)$ can be broadcast across all corresponding positive attribute value v and negative ones v' to accelerate computing in the implementation, because a medicinal product usually contains only one title and multiple attribute names, each of which corresponds multiple attribute values.

For example, given a medicinal product with title (*Sanjiu Medical & Pharmaceutical Cold Remedy Granules*), we firstly draw the attribute name (*syndrome*) and corresponding attribute values (*headache, fever, nasal congestion and runny nose*) from product database. Then we randomly sample negative values (*stomachache, lumbar strain*) for the given syndrome attribute. Intuitively, we would like the name of the medicine to be semantically closer to the corresponding indication of the medicine and more semantically distant from the random chosen indications. In addition, most queries in the medicinal product vertical search usually come from product titles and attribute values, so it is beneficial to model the semantic relationships between these terms using the PAP pre-training task.

Thanks to PAP contrastive loss, the semantic relations of product attributes and the original medicinal product title, which usually contains brand, drug name, ingredients, etc., is explicitly learned. Thus, we can inject medicinal product knowledge into the pre-trained language model.

3.3 Multi-task Pre-training

We pre-train the model from scratch using general and biomedical domain corpora for RTD task and medicinal product datasets for PAP task, see Equation (4).

$$L = L_{\text{RTD}} + L_{\text{PAP}} \quad (4)$$

We train the model parameters by repeatedly switching back and forth between RTD and PAP tasks. The hyper parameter ρ denotes the probability of selecting the PAP task training batch at each gradient descent iteration. The overall training procedure is shown in Algorithm 1.

Algorithm 1 Overall Training Procedure

```
Initialize model parameters randomly.
Mark biomedical entity boundaries for the general and biomedical domain corpora.
Collect triples (product title, attribute name, attribute value) from medicinal product database.
while needing more training steps do
    Select a training batch from RTD and PAP tasks randomly
    if the training batch is from RTD task then
        Sample tokens for randomly masked word spans using the auxiliary transformer.
        Calculate RDT task loss using the main transformer according to Equation (1) and update model parameters.
    else                                                    ▷ for PAP task
        Sample negative attribute values randomly for each product title and attribute name pair.
        Calculate PAP task loss using the same main transformer according to Equation (2) and (3) and update model parameters.
```

4 Experiments and Results

To demonstrate the effectiveness of the PAP pre-training task, we train the pre-trained language

Dataset	#Sen	#Tok
Wikipedia	0.5M	0.4B
News Articles	1M	2B
Package Insert of Drugs†	8K	0.9M
Medical Encyclopedia†	9K	1M
Biomedical Community QA†	39M	8B

Table 1: Statistics of pre-training corpus. Datasets with dagger symbols indicate that they are from the biomedical domain. Other datasets are from general domain.

Medicinal Product dataset	#
Product Titles	29K
Product Attribute Categories	2
Product Attribute Values	191K
Product Attribute Values per Title	6.5

Table 2: Statistics of medicinal product dataset

Dataset	Train	Dev	Test
QTRel-easy	9,303	1,000	1,000
QTRel-hard	8,941	1,000	1,000
QIC	34,929	13,234	13,234
QNER	135,411	14,047	14,047

Table 3: Statistics of PAP evaluation benchmark for medicinal product search.

model from scratch, first on the general corpus and biomedical domain corpus shown in Table 1 using only the RTD task with biomedical entity masking, and then adding the PAP task on the medicinal product dataset shown in Table 2. The reason that we use pre-training from scratch strategy instead of parameter initialization from other existing PLMs to continue pre-training is to exclude the influence caused by different datasets.

Therefore, we first train a base-size model (ELECTRA+EM) on the general and biomedical domain datasets in Table 1 as a strong baseline and compare it with other Chinese PLM models. Then we train another base-size model (ELECTRA+EM+PAP) on the common datasets in Table 1 and the medicinal product dataset in Table 2 and do ablation experiments to verify the effectiveness of the PAP pre-training task.

4.1 Pre-training Datasets

We collect general and biomedical domain specific Chinese corpus, as shown in Table 1. The general domain corpus consists of the Chinese Wikipedia dataset and Chinese News Articles dataset, which are publicly available from NLP Chinese Corpus (Xu, 2019). The biomedical domain specific Chinese corpus consists of Package Insert of Drugs, Medical Encyclopedia and Biomedical Community QA, which are from Shenma Search Engine[2]. As shown in Table 2, We construct (product title, attribute name, attribute value) dataset from vertical search medicinal product database.

4.2 Evaluation Datasets

For the ELECTRA+EM model, we use Chinese Biomedical Language Understanding Evaluation benchmark (ChineseBLUE) (Zhang et al., 2020) to demonstrate the benefits of its in-domain pre-training. The benchmark contains a variety of NLP tasks: cEHRNER and cMedQANER are two named entity recognition tasks; cMedQQ is a paraphrase identification task to determine whether two sentences have the same meaning; cMedQA and cMedQNLI are two question answering tasks that can be approximated as a ranking of candidate answer sentences based on their similarity; cMedIR is a ranking task that retrieves the most relevant documents for a given search query; cMedIC is an intent classification task that assigns three types of labels to query terms with no intention, weak intention, and firm intention; cMedTC is a text classification task that assigns multiple labels to biomedical texts. Further details about these datasets can be found in (Zhang et al., 2020).

For the ELECTRA+EM+PAP model, we construct a benchmark containing four NLP tasks from our medicinal product vertical search scenario to validate the advantage of PAP pre-training task compared to ELECTRA+EM model. As shown in Table 3, the benchmark contains four datasets: QTRel-easy and QTRel-hard are two query-title relevance tasks, where the "hard" part in dataset name means that none of the query terms appear in the corresponding medicinal product title and the "easy" dataset does not have this constraint; QIC is a query intention classification task which assigns 22 different types of labels to queries, including medicine name, disease, symptom, inquiry, etc. QNER is a named entity recognition in query task with 28 total entity types, such as brands, main ingredients of drugs, dosage forms, etc.

[2] http://m.sm.cn/

4.2.1 Parameter Settings

To compare with ELECTRA and other baseline models, we leverage the same model settings of the transformer as ELECTRA. Both ELECTRA+EM and ELECTRA+EM+PAP use the base version of ELECTRA, which contains 12 layers, 12 self-attention heads, and 768-dimensional of hidden size for the discriminator network and 1/3 generator size.

For ELECTRA+EM, we set the initial learning rate as 2e-4, batch size as 128, maximum sequence length as 128, training steps as 420M. For ELECTRA+EM+PAP, we use the same learning rate and maximum sequence length, but set batch size as 16, training steps as 3400M. For the optimizer, we use the same setting with ELECTRA, both in pre-training and fine-tuning steps.

4.3 Results

4.3.1 ELECTRA+EM

For the ELECTRA+EM model, we compare it with several typical Chinese general and biomedical domain PLM baselines, namely BERT-Base[3], ELECTRA-Base[4] (Cui et al., 2020), MC-BERT[5] (Zhang et al., 2020), EMBERT (Cai et al., 2021) and BioHanBERT (Wang et al., 2021c).

For BERT-Base, ELECTRA-Base and our ELECTRA+EM, we run the finetuning 5 times for each downstream task and report the results in *average/maximum* metric format. For MC-BERT, EMBERT and BioHanBERT, we directly cite the results from the corresponding papers. The evaluation metric of all ChineseBLUE datasets is F1 score except the cMedIR dataset whose metric is PAIR score.

As shown in Table 4, ELECTRA+EM achieves comparable performance compared to other baseline and state-of-the-art methods. The comparison also demonstrates the benefits of in-domain pre-training from scratch. We can therefore use ELECTRA+EM as a very strong baseline model to verify the effectiveness of PAP pre-training task.

4.3.2 ELECTRA+EM+PAP

We also run the finetuning 5 times for each downstream task and report the results in *average/maximum* metric format for PAP evaluation benchmark.

As shown in Tables 5, ELECTRA+EM+PAP outperforms ELECTRA-Base significantly on all four tasks. There may be two reasons for these large improvements. Firstly, medicinal product titles and queries in the benchmark usually consist of brand, disease, symptom words and phrases, which are also abundant in PAP pre-training task datasets. Secondly, PAP pre-training task leverages the semantic relation of product titles and attributes to obtain better word representations.

Hyperparamter Since ELECTRA+EM+PAP adopts a multi-task pre-training framework, it is necessary to tune the hyperparameter ρ, the probability of selecting PAP task while training. We search for the best ρ out of $[1\%, 5\%, 10\%, 15\%]$. For the margin hyperparameter γ in PAP loss, we search the best out of $[2, 4, 6, 8]$. We find the combination of $\gamma = 4, \rho = 5\%$ works best. Since results are more insensitive to the hyperparameter γ, we fix $\gamma = 4$ and then plot the effect of the hyperparameter ρ on results, as is shown in Figure 2.

Ablation Study As shown in Table 5, the method without PAP pre-training task, ELECTRA+EM, has worse performance than ELECTRA+EM+PAP. This is reasonable because the product knowledge learned by PAP pre-training task is beneficial for medicinal product search.

It is interesting that on average the performance degradation in terms of F1 score is much larger for QTRel-hard task than QTRel-easy, QIC and QNER tasks. We hypothesize that biomedical entity semantics, which plays a crucial role in QIC and QNER tasks, can be largely captured by biomedical entity masking in RTD task. The PAP pre-training task may be more beneficial for product related concept complex interaction understanding task, such as QTRel-hard dataset.

Case Study As shown in Table 6, We compare ELECTRA+EM+PAP and ELECTRA-Base on QTRel-hard task for a given query "中耳炎" ("tympanitis"). The first two column scores are the predicted probabilities that the given query-title pair is relevant. The label +/- indicates whether the query-title pair is relevant or not in real. Base on their package inserts, the indications of cefixime and azithromycin tablets include tympanitis, while mosapride citrate is not suitable for treating tympanitis. ELECTRA+EM+PAP scores align *more*

[3] https://github.com/google-research/bert

[4] https://github.com/ymcui/Chinese-ELECTRA

[5] https://github.com/alibaba-research/ChineseBLUE

Model	cEHRNER	cMedQANER	cMedQQ	cMedQA
MC-BERT	90.0	88.1	87.5	82.3
BioHanBERT(10K)	90.51	-	86.46	96.53
BioHanBERT(20K)	91.67	-	87.14	96.37
BioHanBERT(30K)	91.83	-	86.26	96.36
BioHanBERT(40K)	90.44	-	87.18	96.49
BioHanBERT(50K)	90.91	-	87.86	**96.65**
EMBERT♣	-	84.49	87.59	75.10
EMBERT♠	-	85.02	**88.06**	75.32
BERT-Base	90.19/90.50	85.05/85.29	87.06/87.43	96.03/96.09
ELECTRA-Base	91.63/92.17	86.37/86.81	87.27/87.46	95.60/95.90
ELECTRA+EM	**92.10/92.85**	**88.18/88.53**	87.56/87.89	96.55/96.78
Model	cMedQNLI	cMedIR	cMedIC	cMedTC
MC-BERT	95.5	2.04	87.5	82.1
BioHanBERT(10K)	95.86	-	90.48	81.78
BioHanBERT(20K)	95.59	-	**96.43**	83.67
BioHanBERT(30K)	95.72	-	83.33	83.00
BioHanBERT(40K)	95.50	-	90.48	82.72
BioHanBERT(50K)	95.78	-	86.90	83.06
EMBERT♣	96.50	-	-	-
EMBERT♠	96.59	-	-	-
BERT-Base	96.05/96.11	3.03/3.07	92.43/92.89	82.99/83.78
ELECTRA-Base	95.42/95.55	3.41/3.49	90.61/92.31	83.43/83.78
ELECTRA+EM	**96.66/96.78**	**3.64/3.71**	92.48/93.26	**83.73/84.00**

Table 4: Experimental results on ChineseBLUE test datasets. For the BioHanBERT model, the number in parentheses indicates the number of steps in the training step. For the EMBERT model, ♣ and ♠ indicate it is initialized by BERT-Base and MC-BERT, respectively.

Model	QTRel-easy	QTRel-hard	QIC	QNER
ELECTRA-Base	98.20/98.51	81.90/82.80	81.37/82.06	82.53/82.78
ELECTRA+EM+PAP	**98.67/98.95**	**84.88/85.86**	**86.90/87.26**	**88.28/88.72**
ELECTRA+EM	98.44/98.66	82.90/83.46	86.58/87.11	88.03/88.35
Average Drop w/o PAP	-0.23	-1.98	-0.32	-0.25

Table 5: Experimental results on PAP evaluation benchmark for medicinal product search.

ELECTRA -base	ELECTRA +EM+PAP	Label	Product Title
0.62	0.88	+	999 头孢克肟片 0.1g*7片/盒 999 Cefixime Tablets 0.1g*7tablets/box
0.49	0.87	+	999 阿奇霉素片 0.25g*6片/盒 999 Azithromycin Tablets 0.25g*6tablets/box
0.93	0.42	-	信谊 美唯宁 枸橼酸莫沙必利胶囊 5mg*24粒 SINE MeiWeiNing Mosapride Citrate Capsules 5mg*24

Table 6: Examples of query-title relevance scores on QTRel-hard task for the query “tympanitis”.

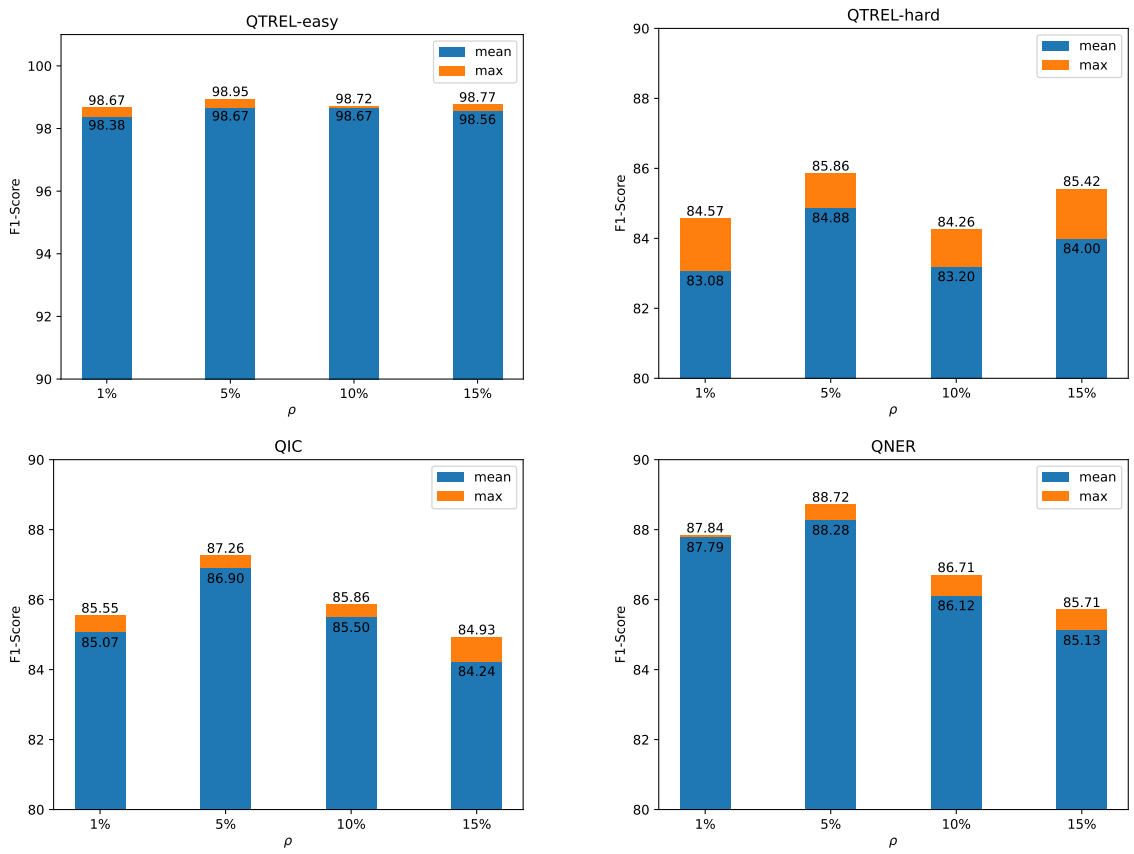

Figure 2: Hyperparameter tuning for ρ (when $\gamma = 4$) on PAP evaluation benchmark for medicinal product search.

closely with labels than ELECTRA-Base. For the first example, ELECTRA+EM+PAP is more confident about the relevance of "cefixime tablets" and "tympanitis" than ELECTRA-Base. When a threshold probability of 0.5 is applied, ELECTRA+EM+PAP succeeds in classifying the last two examples, while ELECTRA-Base fails.

5 Discussion

As mentioned in Section 3, other self-supervised learning tasks for language models could be learned together with the PAP pre-training task. We only explore the combination of ELECTRA's RTD and PAP for joint training. Due to limited computational resources, we could not push the model to larger sizes. These may prevent the full potential of PAP pre-training task from being unleashed.

For vertical search applications, PAP pre-training task is also applicable to other product searches. For example, based on movie structured information such as movie title, genre and story type, it is feasible to use PAP task to model the semantic relationships between these movie attributes. In the study of product knowledge enhanced language models, comparing the way PAP uses product structured information with the graph embedding approach based on product knowledge graphs may be an interesting research problem for the future.

6 Conclusion

In this article, we propose a biomedical knowledge enhanced pre-trained language model for medicinal product vertical search. We improve ELECTRA's replaced token detection pre-training task with biomedical entity masking (EM). Then we present a novel pre-training task, product attribute prediction (PAP), to incorporate medicinal product knowledge into the PLM. We train ELECTRA+EM and ELECTRA+EM+PAP two biomedical knowledge enhanced pre-trained language models to demonstrate the effectiveness of PAP pre-training task for medicinal product vertical search. Our work may shed some light on combining the powerful pre-trained language models with product knowledge for vertical search scenarios.

References

Iz Beltagy, Kyle Lo, and Arman Cohan. 2019. Scibert: A pretrained language model for scientific text. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 3615–3620.

Antoine Bordes, Nicolas Usunier, Alberto Garcia-Duran, Jason Weston, and Oksana Yakhnenko. 2013. Translating embeddings for modeling multi-relational data. In *In NIPS*, pages 2787–2795.

Tom B Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. 2020. Language models are few-shot learners. *arXiv preprint arXiv:2005.14165*.

Zerui Cai, Taolin Zhang, Chengyu Wang, and Xiaofeng He. 2021. Embert: A pre-trained language model for chinese medical text mining. In *Web and Big Data: 5th International Joint Conference, APWeb-WAIM 2021, Guangzhou, China, August 23–25, 2021, Proceedings, Part I*, page 242–257, Berlin, Heidelberg. Springer-Verlag.

Souradip Chakraborty, Ekaba Bisong, Shweta Bhatt, Thomas Wagner, Riley Elliott, and Francesco Mosconi. 2020. Biomedbert: A pre-trained biomedical language model for qa and ir. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 669–679.

Kevin Clark, Minh-Thang Luong, Quoc V. Le, and Christopher D. Manning. 2020. ELECTRA: Pre-training text encoders as discriminators rather than generators. In *ICLR*.

Yiming Cui, Wanxiang Che, Ting Liu, Bing Qin, Shijin Wang, and Guoping Hu. 2020. Revisiting pre-trained models for Chinese natural language processing. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: Findings*, pages 657–668, Online. Association for Computational Linguistics.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, Minneapolis, Minnesota. Association for Computational Linguistics.

Yu Gu, Robert Tinn, Hao Cheng, Michael Lucas, Naoto Usuyama, Xiaodong Liu, Tristan Naumann, Jianfeng Gao, and Hoifung Poon. 2021. Domain-specific language model pretraining for biomedical natural language processing. *ACM Transactions on Computing for Healthcare (HEALTH)*, 3(1):1–23.

Zhenzhong Lan, Mingda Chen, Sebastian Goodman, Kevin Gimpel, Piyush Sharma, and Radu Soricut. 2019. Albert: A lite bert for self-supervised learning of language representations. *arXiv preprint arXiv:1909.11942*.

Jinhyuk Lee, Wonjin Yoon, Sungdong Kim, Donghyeon Kim, Sunkyu Kim, Chan Ho So, and Jaewoo Kang. 2020. Biobert: a pre-trained biomedical language representation model for biomedical text mining. *Bioinformatics*, 36(4):1234–1240.

Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Veselin Stoyanov, and Luke Zettlemoyer. 2020a. Bart: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7871–7880.

Patrick Lewis, Myle Ott, Jingfei Du, and Veselin Stoyanov. 2020b. Pretrained language models for biomedical and clinical tasks: Understanding and extending the state-of-the-art. In *Proceedings of the 3rd Clinical Natural Language Processing Workshop*, pages 146–157.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Roberta: A robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692*.

George Michalopoulos, Yuanxin Wang, Hussam Kaka, Helen Chen, and Alexander Wong. 2021. Umls-BERT: Clinical domain knowledge augmentation of contextual embeddings using the Unified Medical Language System Metathesaurus. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 1744–1753, Online. Association for Computational Linguistics.

Matthew Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer. 2018. Deep contextualized word representations. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 2227–2237.

Matthew E Peters, Mark Neumann, Robert Logan, Roy Schwartz, Vidur Joshi, Sameer Singh, and Noah A Smith. 2019. Knowledge enhanced contextual word representations. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 43–54.

Nina Poerner, Ulli Waltinger, and Hinrich Schütze. 2020. E-bert: Efficient-yet-effective entity embeddings for bert. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: Findings*, pages 803–818.

Alec Radford, Karthik Narasimhan, Tim Salimans, and Ilya Sutskever. 2018. Improving language understanding by generative pre-training.

Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, Ilya Sutskever, et al. 2019. Language models are unsupervised multitask learners. *OpenAI blog*, 1(8):9.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J Liu. 2020. Exploring the limits of transfer learning with a unified text-to-text transformer. *Journal of Machine Learning Research*, 21:1–67.

Zhiqing Sun, Zhi-Hong Deng, Jian-Yun Nie, and Jian Tang. 2019. Rotate: Knowledge graph embedding by relational rotation in complex space. In *International Conference on Learning Representations*.

Benyou Wang, Qianqian Xie, Jiahuan Pei, Prayag Tiwari, Zhao Li, et al. 2021a. Pre-trained language models in biomedical domain: A systematic survey. *arXiv preprint arXiv:2110.05006*.

Ruize Wang, Duyu Tang, Nan Duan, Zhongyu Wei, Xuanjing Huang, Jianshu Ji, Guihong Cao, Daxin Jiang, and Ming Zhou. 2021b. K-Adapter: Infusing Knowledge into Pre-Trained Models with Adapters. In *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*, pages 1405–1418, Online. Association for Computational Linguistics.

Xiaosu Wang, Yun Xiong, Hao Niu, Jingwen Yue, Yangyong Zhu, and Philip S. Yu. 2021c. Biohanbert: A hanzi-aware pre-trained language model for chinese biomedical text mining. In *2021 IEEE International Conference on Data Mining (ICDM)*, pages 1415–1420.

Yu Wang, Jinchao Li, Tristan Naumann, Chenyan Xiong, Hao Cheng, Robert Tinn, Cliff Wong, Naoto Usuyama, Richard Rogahn, Zhihong Shen, et al. 2021d. Domain-specific pretraining for vertical search: Case study on biomedical literature. In *Proceedings of the 27th ACM SIGKDD Conference on Knowledge Discovery & Data Mining*, pages 3717–3725.

Bright Xu. 2019. Nlp chinese corpus: Large scale chinese corpus for nlp.

Zhilin Yang, Zihang Dai, Yiming Yang, Jaime Carbonell, Ruslan Salakhutdinov, and Quoc V Le. 2019. Xlnet: Generalized autoregressive pretraining for language understanding. *arXiv preprint arXiv:1906.08237*.

Zheng Yuan, Yijia Liu, Chuanqi Tan, Songfang Huang, and Fei Huang. 2021. Improving biomedical pretrained language models with knowledge. In *Proceedings of the 20th Workshop on Biomedical Language Processing*, pages 180–190, Online. Association for Computational Linguistics.

Ningyu Zhang, Qianghuai Jia, Kangping Yin, Liang Dong, Feng Gao, and Nengwei Hua. 2020. Conceptualized representation learning for chinese biomedical text mining. *arXiv preprint arXiv:2008.10813*.

Ningyu Zhang, Zhanlin Sun, Shumin Deng, Jiaoyan Chen, and Huajun Chen. 2019a. Improving few-shot text classification via pretrained language representations. *arXiv preprint arXiv:1908.08788*.

Taolin Zhang, Chengyu Wang, Minghui Qiu, Bite Yang, Zerui Cai, Xiaofeng He, and Jun Huang. 2021. Knowledge-empowered representation learning for Chinese medical reading comprehension: Task, model and resources. In *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*, pages 2237–2249, Online. Association for Computational Linguistics.

Zhengyan Zhang, Xu Han, Zhiyuan Liu, Xin Jiang, Maosong Sun, and Qun Liu. 2019b. Ernie: Enhanced language representation with informative entities. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 1441–1451.

CONCRETE: Improving Cross-lingual Fact-checking with Cross-lingual Retrieval

Kung-Hsiang Huang **ChengXiang Zhai** **Heng Ji**
Department of Computer Science, University of Illinois Urbana-Champaign
{khhuang3, czhai, hengji}@illinois.edu

Abstract

Fact-checking has gained increasing attention due to the widespread of falsified information. Most fact-checking approaches focus on claims made in English only due to the data scarcity issue in other languages. The lack of fact-checking datasets in low-resource languages calls for an effective cross-lingual transfer technique for fact-checking. Additionally, trustworthy information in different languages can be complementary and helpful in verifying facts. To this end, we present the first fact-checking framework augmented with cross-lingual retrieval that aggregates evidence retrieved from multiple languages through a cross-lingual retriever. Given the absence of cross-lingual information retrieval datasets with claim-like queries, we train the retriever with our proposed *Cross-lingual Inverse Cloze Task* (X-ICT), a self-supervised algorithm that creates training instances by translating the title of a passage. The goal for X-ICT is to learn cross-lingual retrieval in which the model learns to identify the passage corresponding to a given translated title. On the X-FACT dataset, our approach achieves 2.23% absolute F1 improvement in the zero-shot cross-lingual setup over prior systems. The source code and data are publicly available at https://github.com/khuangaf/CONCRETE.

1 Introduction

Fact-checking is an important task that assesses the veracity of a claim. This task has gained increasing attention due to the widespread mis- and disinformation that has a significant socioeconomic impact on our society (Scheufele and Krause, 2019; Pate et al., 2019; Fung et al., 2021; Guo et al., 2022; Wu et al., 2022; Fung et al., 2022; Huang et al., 2022). While fact-checking is mainly conducted manually, especially in the journalism industry, with more than hundred of millions of social media posts and millions of blog posts published per day (Hoang and Mothe, 2018; Djuraskovic, 2022), manual fact-checking is no longer feasible. Hence, we are urgently in need of reliable automated fact-checking approaches.

Most existing work develops fact-checking approaches on English-only corpus (Wang, 2017; Thorne et al., 2018; Augenstein et al., 2019; Wadden et al., 2020). One reason for this is the scarcity of fact-checking websites in other languages for constructing large enough non-English fact-checking datasets. Therefore, it is even more challenging to build fact-checkers for low-resource languages. A solution is to leverage high-resource languages with zero-shot cross-lingual transfer, where the model is trained on source languages of richer resources and directly tested on target languages of lower resources. Note that the sets of languages in the training set and the test sets are disjoint. With this technique, ground-truth labels for claims in low-resource languages are no longer required.

Few studies in the fact-checking literature have explored the cross-lingual setup. One line of work attempts to match an input claim with claims in other languages that have been verified (Kazemi et al., 2021, 2022). However, this approach fails when the claims have not been fact-checked in any language. Another line of work builds classifiers with multilingual language models. For example, Gupta and Srikumar (2021) utilize Google Search to obtain snippets that are relevant to the input claim and train a model based on mBERT (Devlin et al., 2019) in a cross-lingual setting. Although Google Search excels at retrieving relevant information from a given claim, it disregards the trustworthiness of the information being retrieved. When a claim is erroneous, the search results often contradict each other, which likely leads to incorrect predictions, as shown later in Section 5.3.

Motivated by these challenges, we propose CONCRETE, a **C**laim-**o**rie**n**ted **C**oss-lingual **Ret**ri**e**ver that retrieves evidence from a trustworthy multi-

Proceedings of the 29th International Conference on Computational Linguistics, pages 1024–1035
October 12–17, 2022.

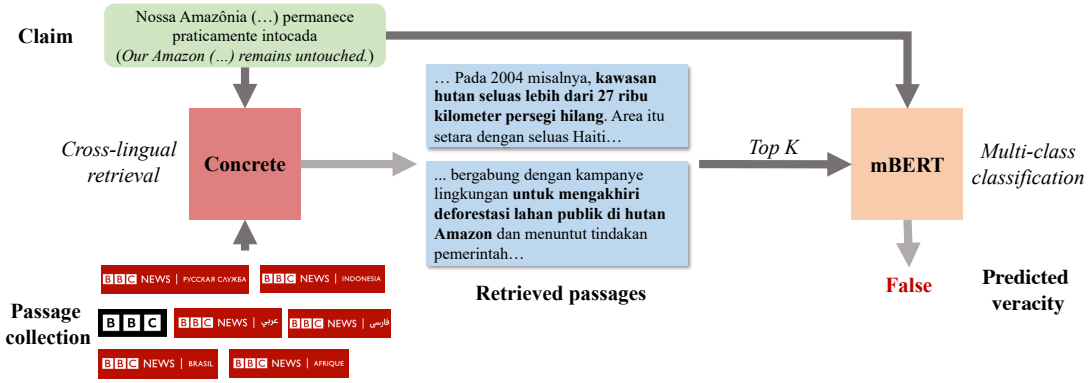

Figure 1: An overview of the proposed framework. Given a claim in arbitrary language, a cross-lingual retriever, CONCRETE retrieves relevant passages in *any languages*. The top-k relevant passages and the claim are then passed to our multilingual reader, mBERT, to predict the veracity of the claim.

lingual passage collection for fact-checking. This approach can handle region-specific claims as long as relevant evidence is presented in the passage collection, which is much more accessible compared to similar claims in other languages. In addition, since it does not rely on a black-box retrieval system, our approach provides the flexibility to include only trustworthy information in the passage collection. One major challenge for training such a retriever is the lack of multilingual information retrieval (IR) dataset with claim-like queries. To this end, we propose a self-supervised cross-lingual learning algorithm, Cross-lingual Inverse Cloze Task (X-ICT), to learn the retriever based on pseudo-feedback. To mimic cross-lingual retrieval, we construct a pseudo-query for a given passage by translating its title into a randomly selected language. The objective for X-ICT is to identify the passage corresponding to a given translated title among all candidate passages. Since the title of news articles can often be regarded as a claim, this approach mitigates the domain discrepancy issue. As shown in Figure 1, our framework first performs cross-lingual retrieval to obtain evidence relevant to the input claim. Then, a multilingual reader takes in the retrieved evidence and the input claim to classify the veracity of the claim.

Our contributions can be summarized as follows:

- To the best of our knowledge, we present the first fact-checking framework augmented with cross-lingual retrieval that achieves state-of-the-art cross-lingual transfer performance on the X-FACT fact-checking task.
- We propose CONCRETE, a cross-lingual retriever with a bi-encoder architecture learned through a proposed self-supervised learning algorithm.
- Our experiments reveal that the distance between the input claim and the retrieved passages is strongly correlated with the performance.
- We collected a multilingual passage collection composed of reliable news articles in seven different languages. We have demonstrated that this corpus is effective for retrieval-augmented fact-checking.

2 Task Definitions

The input is a claim c in an arbitrary language and the corresponding metadata, such as claimer and claim date. Based on c, the retriever component of our model retrieves relevant passages p from a multilingual passage collection P where p can be in any language. Then, the reader component takes in the claim c, the corresponding metadata, and relevant passages p to predict the veracity of c. Note that the use of passage collection P to aid in fact-checking is a modeling choice, as no grounded evidence is available.

We aim to build a passage collection that does not contain erroneous information. Hence, we construct P by crawling 49,000 articles published in 7 languages[1] between September 2016 and December 2022 from a trustworthy news media, `bbc.com`, and split each article into passages. Each passage contains at most 100 tokens, following Karpukhin et al. (2020). This results in a total of 347,557 passages. The collected passage collection has been made publicly available in the link mentioned in the Abstract.

[1]We consider Arabic, Russian, Indonesian, Persian, French, and Portuguese.

3 Proposed Method

Our framework is a pipeline consisting of two components: (1) CONCRETE, a claim-oriented cross-lingual retriever that retrieves relevant passages from a multilingual passage collection, and (2) a multilingual reader that determines the veracity of a claim based on the compatibility of the claim and the passages retrieved. Figure 1 shows an overview of the proposed method. The following sections describe each component in detail.

3.1 CONCRETE

CONCRETE takes in a claim c in an *arbitrary language* as input and retrieves k relevant passages $p = \{p_0, ..., p_k\}$ from a multilingual passage collection $\mathcal{P}$, where the retrieved passages can be in *any language*. We adapt mDPR (Asai et al., 2021b), a multilingual retriever, with the proposed self-supervised learning algorithm, Cross-lingual Inverse Cloze Task, for claim-like queries.

mDPR mDPR is a multilingual version of the Dense Passage Retriever (DPR) (Karpukhin et al., 2020). Similar to DPR, mDPR is a bi-encoder architecture that computes the relevance score between a query and a passage with the inner product of the corresponding representations. In addition to being trained on existing multilingual question answering datasets, mDPR also learns from additional samples mined from Wikipedia and labeled with the proposed answer generator discussed in Asai et al. (2021b). However, since mDPR was trained on datasets where queries are questions instead of claims, domain mismatch becomes an issue if we directly apply mDPR to our task.

Cross-lingual Inverse Cloze Task To address the domain discrepancy problem discussed in the previous paragraph, a naive solution is to fine-tune mDPR on multilingual IR datasets with claim-like queries. Unfortunately, such datasets are not available. Motivated by the Inverse Cloze Task (Lee et al., 2019), which was used to warm start the retriever by tasking a retriever to predict the context given a randomly sampled sentence, we propose Cross-lingual Inverse Cloze Task (X-ICT) by extending ICT to a cross-lingual setup. Specifically, we made two major modifications to the original ICT.

First, instead of randomly sampling a sentence as the pseudo query, we treat the title of each passage as the pseudo query as these titles are often

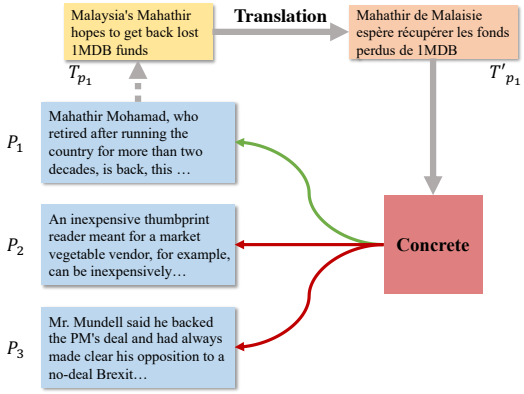

Figure 2: An illustration of X-ICT. Given a passage p_1, we find its title T_{p_1} and translates it into a different language T'_{p_1} to mimic cross-lingual retrieval. The goal for the cross-lingual retriever, CONCRETE, is to select the correct passage p_1 based on the translated title T'_{p_1} among all the passages in the same batch.

claim-like sentences. The positive passages for a given title are the passages derived from the same article. The biggest advantage of this approach is that there is very little domain mismatch between using claims as queries and using titles as queries. Therefore, the retriever optimized with our proposed X-ICT can be directly applied to claim-oriented downstream tasks without further fine-tuning, indicating that the optimization for the downstream tasks can be more efficient. Second, to mimic cross-lingual retrieval, we use mBART-50 (Tang et al., 2020), a machine translation model, to translate the title into a target language.

Formally, in CONCRETE, two dense encoders, $E_C(\cdot)$ and $E_P(\cdot)$, are used to represent claims and passages as d-dimensional vectors. The similarity between a claim and a passage is defined as the dot product between their dense vectors,

$$sim(c, p_i) = E_C(c)^\top E_P(p_i). \quad (1)$$

In X-ICT, we form positive claim-passage pairs by constructing the translation of the corresponding title T_{p_i} of a given passage p_i. i.e. (T'_{p_i}, p_i) where T'_{p_i} is the original title T_{p_i} translated into a different language using mBART-50. We treat other passages in the same batch as negative samples. The retriever is trained by optimizing the negative log likelihood,

$$P(p_i|T'_{p_i}) = \frac{\exp(sim(T'_{p_i}, p_i))}{\sum_{p_j \in \text{BATCH}} \exp(sim(T'_{p_i}, p_j))} \quad (2)$$

$$\mathcal{L}_{\text{X-ICT}} = -\log \sum_{p_i \in \mathcal{P}} P(p_i|T'_{p_i}) \quad (3)$$

Split	# claims	# languages
Train	19079	13
Development	2535	12
In-domain	3826	12
Out-of-domain	2368	4
Zero-shot	3381	12

Table 1: Dataset statistics of X-FACT.

Figure 2 demonstrates a graphical illustration.

However, if the translation is performed on every claim, the model would be discouraged from retrieving passages in the same language as the query, which is not a desirable property of the retriever. Therefore, we set an equal probability for a claim to be translated to any language or not being translated (i.e. the probability of not doing translation is $\frac{1}{7}$). We repurposed the passage collection as the training corpus for learning X-ICT.

3.2 Multilingual Reader

We use mBERT (Devlin et al., 2019), a multilingual version of BERT, as the encoder for our multilingual reader. The claim c and the corresponding metadata are encoded jointly with a template T: [Claim made by *Claimer* on *Claim-Date*, reported in *Language*: *Claim*], where *Claimer*, *Claim-Date*, and *Language* are placeholders. The claim template T and each of the k[2] retrieved passages p_i are first encoded independently

$$h_T = \text{mBERT}(T)\texttt{[CLS]} \quad (4)$$

$$h_{p_i} = \text{mBERT}(p_i)\texttt{[CLS]}. \quad (5)$$

The final prediction is then made by feeding the concatenated `[CLS]` embeddings into a multilayer perceptron $\hat{y} = \text{MLP}([h_T; h_{p_0}; \ldots; h_{p_k}])$. The model is optimized with the cross-entropy loss

$$\mathcal{L} = \frac{1}{N}\sum_{i=1}^{N} y_i \log \hat{y}_i, \quad (6)$$

where y_i and $\hat{y}_i$ denote the ground truth label and the predicted label of the i-th sample respectively, and N denotes the total number of samples.

4 Experimental Setup

4.1 Dataset and Evaluation Metric

Our experiments are conducted on a multilingual fact-checking dataset: X-FACT (Gupta and Srikumar, 2021). X-FACT contains 31,189 claims in 25 languages collected from fact-checking websites via Google's Fact Check Explorer. Each claim is annotated with one of the following seven labels: TRUE, MOSTLY-TRUE, PARTLY-TRUE, MOSTLY-FALSE, FALSE, UNVERIFIABLE, and OTHER. Gupta and Srikumar (2021) split the data in a way that allows evaluation in various settings, as described in Table 1. *In-domain* and *out-of-domain* test sets contain claims in the same languages as those in the training set, except that the claims in the *out-of-domain* split are from different websites. Our main focus is the *zero-shot* setup, where there is no overlap between the languages in the *zero-shot* split and those in the training set. We use macro F1 as the evaluation metric, following Gupta and Srikumar (2021).

4.2 Baselines

We compare the following competitive retrieval systems using different retrieval components but with the same multilingual reader.

MT + DPR A common approach to cross-lingual tasks is translating inputs from target languages to source languages of richer resources so that stronger monolingual models can be utilized (Ahmad et al., 2021; Asai et al., 2021a). We translate all claims and all passages into English with the HELSINKI-NLP neural machine translation models[3] for its comprehensive language coverage and decent performance. For languages not covered by HELSINKI-NLP, we use Google Translate instead. Then, we use DPR (Karpukhin et al., 2020) to perform retrieval based on the translated claims and passages.

BM25 BM25 (Robertson and Zaragoza, 2009) has demonstrated advantages over dense vector representation approaches in monolingual retrieval tasks (Lee et al., 2019). Since BM25 only works in a monolingual setup, we create dummy empty passages for claims whose languages are not presented in the passage collection P. Our implementation is based on the Rank-BM25 package[4].

mDPR mDPR is a multilingual retriever based on DPR. It was trained on multilingual question answering datasets, as detailed in Section 3.

Google Search As demonstrated in previous work (Augenstein et al., 2019; Gupta and Srikumar, 2021), snippets from Google Search results can serve as evidence for fact-checking. We directly

[2]Empirically, we set k to 5 for best overall performance.

[3]https://huggingface.co/Helsinki-NLP

[4]https://pypi.org/project/rank-bm25/

	Reader	Retrieval Method	Zero-shot F1 (%)	In-domain F1 (%)
Prior (Gupta and Srikumar, 2021)	Majority	None	7.6	6.9
	mBERT	None	16.7	39.4
	mBERT	Google Search	16.0	41.9
Ours	mBERT	None	17.25	36.91
	mBERT	Google Search	16.02	**42.61**
	mBERT	MT+DPR	15.01	35.29
	mBERT	BM25	17.43	38.29
	mBERT	mDPR	17.60	36.79
	mBERT	CONCRETE	**19.83***	40.53

Table 2: Performance comparison in macro F1 (%) of various models on the X-FACT test sets. *None* retrieval method means not using any retrieval component, while *Majority* means predicting the majority label, FALSE, for all samples. When no retrieval method is used, the reader performs fact checking simply based on the claim template T described in Section 3. Statistical significance over the second best models computed using the paired bootstrap procedure (Berg-Kirkpatrick et al., 2012) is indicated with * ($p < .05$).

take the snippets obtained by Gupta and Srikumar (2021) as inputs.

4.3 Implementation Details

When trained with X-ICT on the passage collection, the retriever is optimized using AdamW (Loshchilov and Hutter, 2019) with a learning rate of 2e-5 over 30 epochs. When fine-tuning the multilingual reader, we set the learning rate to 5e-5 for parameters in mBERT and 1e-3 for all other parameters. The maximum input sequence length for X-ICT and fine-tuning on X-FACT are set to 256 and 512, respectively. We use the pre-trained mBERT checkpoints on HuggingFace[5].

5 Results

5.1 Main results

Table 2 summarizes the fact-checking performance on the X-FACT dataset. Our framework establishes a new state-of-the-art in zero-shot cross-lingual fact-checking, outperforming the previous best models by an absolute macro F1 of 2.23%. The improvements demonstrate the effectiveness of our approach in retrieving relevant passages from multiple languages to assist in fact-checking. Furthermore, we found that the use of vanilla mDPR does not improve performance compared to the absence of any retrieval component. This can be explained by the domain discrepancy issue for mDPR, as it was trained on question-like queries instead of claim-like queries. Furthermore, although CONCRETE is the most advantageous in the zero-shot setup, it trails behind Google Search in-domain setup. The significantly increased gap between in-domain F1 score and zero-shot F1 score for using Google Search suggests that the reader may exploit biases or patterns presented in Google Search's results that are not transferrable across languages. To validate this hypothesis, we analyzed the relationship between the snippets returned by Google Search and the ground-truth labels. We found that for claims of richer resources in the training set, Google Search is able to retrieve evidence that strongly indicates the veracity of the claims due to the abundance of fact-checking websites. For example, among Indonesian claims where Google Search results contain the string "SALAH" (WRONG), 50% of them are PARTLY TRUE and 45% of them are FALSE. Such patterns can also be found in the *in-domain* split, but not in the *zero-shot* split. This finding explains the increased gap between the performance on these two splits when using Google Search as the retrieval method. It also implies that our approach is more generalizable to lower-resource languages and more applicable when no fact-checking websites have debunked the input claims.

5.2 Performance Analysis

Language Choice in Passage Collection Throughout our experiments, we found that the language of the retrieved passage can affect the fact-checking performance. We hypothesize that it may be more challenging for readers to reason through passages when the passage language is distant from the claim language. To verify such a hypothesis, we first compute the distance between each pair of languages based on word ordering, following Ahmad et al. (2019). Then, we remove passages of a particular language from the passage collection. With this subset of selected passage collection, we use CONCRETE to retrieve relevant passages for fact-checking and

[5] https://huggingface.co/bert-base-multilingual-cased

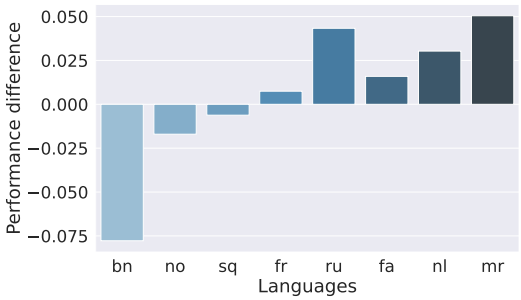

Figure 3: Performance difference in macro F1 when Indonesian passages are removed from the passage collection. On the x-axis, languages are sorted in ascending order by distance to Indonesian (i.e. Bengali is the closest, while Marathi is the furthest). We compute the distance between two languages based on word ordering, following Ahmad et al. (2019).

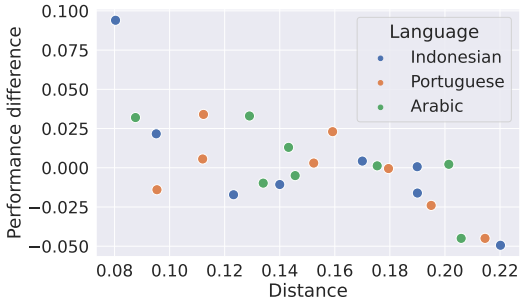

Figure 4: Performance difference in macro F1 when the retrieved passages are translated to Indonesian, Portuguese, and Arabic, respectively. Overall, the performance difference has a negative correlation with the distance between the passage's language and the claim's language.

train another fact-checker based on newly retrieved passages. Finally, we compare the performance difference for different languages between this system and the model discussed in Section 3 in the *zero-shot* setup. Figure 3 shows the results for each language when Indonesian passages are removed from the passage collection. We observe that the performance drop and the distance between each language and Indonesian are highly correlated. In fact, we see that removing Indonesian is beneficial for some distant languages such as Dutch and Marathi.

Given these results, the following question arises: is this phenomenon caused by the poor capability of our multilingual reader to reason with passages and claims in distant languages, or by the higher information overlap between Indonesian passages and claims in closer languages? To better understand the results, we translate the retrieved passages into three languages of distinct language families: Indonesian, Portuguese, and Arabic. With the three sets of translated passages, we train three fact-checkers. Then, we compute the difference between the original performance and the performance achieved using the translated passages. In Figure 4, we observe that the performance difference is negatively correlated with the distance between the languages of the claim and the passage. The correlation coefficient is -0.490 per Kendall's Tau (Kendall, 1938). This confirms that passages in distant languages are indeed harder for our multilingual reader to reason with. The trend is consistent with the findings of Asai et al. (2021b).

Impact of the Amount of Training Data To test the data efficiency of our approach, we compared CONCRETE with mDPR and Google Search in the *zero-shot* setup using different numbers of languages for training. As shown in Figure 5, CONCRETE consistently outperforms the other two methods across all settings. This indicates the strength of CONCRETE in aiding fact-checking in low- and high-resource scenarios.

Impact of Retrieving from Multiple Languages We conducted a case study on cross-lingual retrieval versus monolingual retrieval to understand whether retrieving passages from multiple languages actually helps the performance. In particular, we train and evaluate the models on samples whose languages are in the passage collection[6]. For the monolingual retrieval setting, the model is restricted to retrieving passage in the same language as the input claim, while the cross-lingual setting does not have such a restriction. We found that for both mDPR and CONCRETE, cross-lingual retrieval setting outperforms their monolingual retrieval counterparts, as shown in Figure 6. This finding confirms that retrieving evidence in multiple languages helps cross-lingual transfer for fact-checking.

5.3 Qualitative Analysis

The following qualitative analysis provides an intuition for our model's advantage in cross-lingual fact-checking.

Impact of X-ICT To validate the effectiveness of X-ICT in retrieving claims that are more relevant to the topic of the claim, we compared 50 predictions between models using CONCRETE and mDPR as retriever in the split *zero-shot*. The results show that 23 errors made by mDPR are corrected

[6]We train the models on Portuguese, Indonesian, and Arabic, and evaluate them on French, Persian, and Russian.

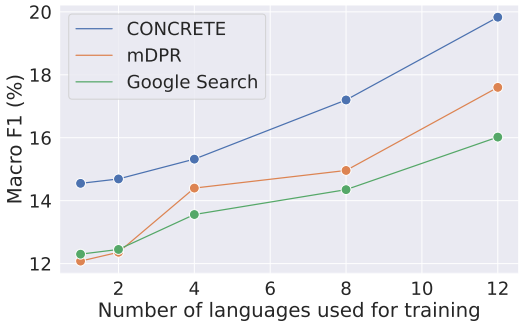

Figure 5: Zero-shot cross-lingual performance on X-FACT with regard to various numbers of languages used for training.

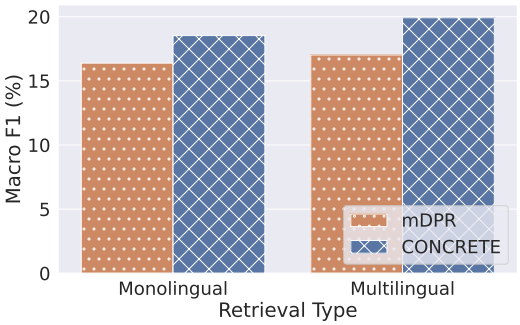

Figure 6: Performance comparison between using monolingual and multilingual retrieval on cross-lingual fact-checking. For both mDPR and CONCRETE, retrieving passages in multiple languages improves the overall performance.

by CONCRETE, while only 6 new errors are introduced. We found that mDPR can often retrieve passages that are relevant to a part of a claim, but the topic of the retrieved passages may not align well with that of the claim, likely due to the domain mismatch issue discussed in Section 3. An example is shown in Figure 8. This reflects that X-ICT is able to improve the retrieval quality even though only pseudo feedback is used for training.

Importance of Passages' Trustworthiness Comparing the predictions and the retrieved passages between CONCRETE and Google Search on the *zero-shot* split, we observe that our approach is better at identifying FALSE and MOSTLY FALSE claims. As demonstrated in Figure 8, when a claim is FALSE or MOSTLY FALSE, the snippets returned by Google Search are often contradicted with each other, which usually leads to incorrect predictions. The inconsistency in the snippets from Google Search is caused by the fact that Google Search retrieves information from the entire Web without considering the trustworthiness of the source. On the contrary, our approach offers the flexibility to include only trustworthy information in the passage collection. In the *zero-shot* split, we found that this property of Google Search leads to 44 and 67 more errors in identifying FALSE and MOSTLY FALSE claims, respectively.

5.4 Remaining Challenges

To identify the remaining challenges, we compare 50 errors made by our model with ground-truth labels and analyze the sources of errors, as illustrated in Figure 7. The following paragraphs will discuss these categories with examples.

Evidence cannot be retrieved. The most common error is caused by the absence of supporting or refuting evidence in the passage collection. For the majority of such errors, the claims are country specific. For example,

> Mann med tre koner får tre leiligheter i Sverige. (Husband with three wives gets three apartments in Sweden.)

We can address this issue by adding trustworthy news articles from more countries into the passage collection.

Under-specified context. Another major source of errors is the underspecification of the input claim. An example claim is:

> "Nuk besoj që janë të informuar as partnerët tanë, SHBA dhe NATO, sepse do të isha i informuar edhe unë." (I do not believe that our partners, the US and NATO, are informed either, because I would be informed as well.)

In this claim, it does not specify who is "I" and what the US and NATO are not informed. Therefore, the given information is too little to determine the veracity of the claim. This problem could potentially be solved by mining the original context from the Internet.

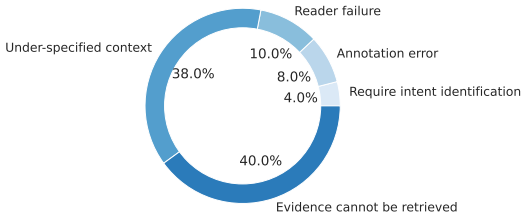

Figure 7: Distribution of the remaining errors.

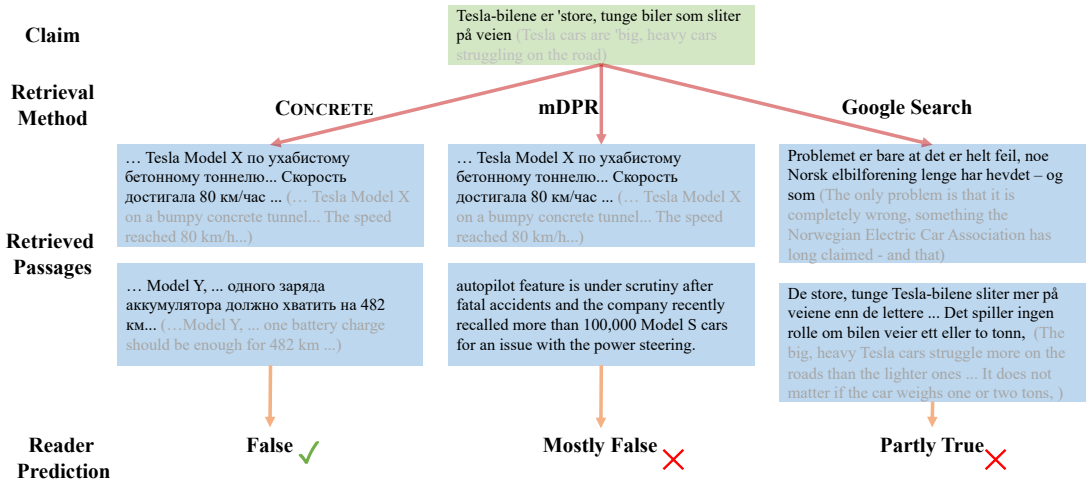

Figure 8: An example showing how relevant trustworthy passages retrieved by CONCRETE lead to correct predictions. CONCRETE retrieves two passages that prove Tesla cars are not slow or struggling on the road. Since the passages are from trustworthy sources, our reader can correctly predict the claim as FALSE. On the other hand, the second passage that mDPR retrieves is slightly related to the claim (Tesla) but not directly relevant to the topic (the poor performance of Tesla cars). For Google Search's results, these two passages contradict each other as one is from a fact-checking website and the other is from an unreliable source. Hence, the model cannot predict correctly based on the retrieved passages from mDPR or Google Search.

Require intent identification. Some of the claims are correct but contain misleading information. For instance,

> "Писатель Дин Кунц предсказал появление коронавируса в своей книге в 1981 году и называл его Ухань-400." (Writer Dean Koontz predicted the emergence of the coronavirus in his book in 1981 and called it Wuhan-400.)

This claim is correct, but the claimer attempts to mislead the audience by linking the coronavirus in the book with COVID-19, which is false. To correctly predict this claim, the model should be capable of identifying the intent behind the claims.

Reader failure. In some cases, the reader fails to predict the correct veracity even though the supporting evidence is successfully retrieved. This is mainly due to the long distances between passages and claims, as discussed in Section 5.2.

Annotation error. The dataset was created by matching the rating of each claim on fact-checking websites with its label definition. Annotation errors could be caused by (1) the misalignment between the rating and the label definition, and (2) the inaccurate ratings listed on the fact-checking websites. We found that each case accounts for about half of the annotation errors.

6 Related Work

6.1 Fact-checking

Previous fact-checking approaches can be roughly divided into two categories based on task formulations. The first type of formulation assumes that evidence candidates are given, such as the FEVER dataset (Thorne et al., 2018) and the SCIFACT dataset (Wadden et al., 2020). Previous approaches for this category of fact-checking tasks often involve a retrieval module to retrieve relevant evidence from the given candidate pool followed by a reasoning component that determines the compatibility between a piece of evidence and the input claim (Yin and Roth, 2018; Pradeep et al., 2021). The second category is the *open-retrieval* setting[7], where evidence candidates are not provided, such as the LIAR dataset (Wang, 2017) and the X-FACT dataset (Gupta and Srikumar, 2021). For this task formulation, one of the main challenges is where and how to retrieve evidence. Some work determines the veracity of a claim based solely on the claim itself and the information learned by language models during the pre-training stage (Lee et al., 2021). However, such an approach is tied to the period of time in which the pre-training data is collected and does not generalize well to new claims. Other studies

[7]We borrow the term *open-retrieval* from the field of question answering.

devise hand-crafted linguistic features as input to the fact-checking models (Mihalcea and Strapparava, 2009; Choudhary and Arora, 2021). However, these approaches are language-specific, whereas our approach is language-agnostic since it consists of a cross-lingual retriever and a multilingual language model. Gupta and Srikumar (2021) use a similar approach, which retrieves relevant snippets using Google Search instead of a cross-lingual retriever. Our experimental results show that the proposed cross-lingual retriever is more effective than Google Search in the *zero-shot* setting due to the fact that Google Search ignores the trustworthiness of the retrieved information and that downstream models tend to exploit the biased patterns in Google Search results that are not transferrable across languages, as shown in Section 5.

6.2 Cross-lingual Retrieval

Early attempts on cross-lingual retrieval adopt a pipeline consisting of a statistical machine translation system and a monolingual retrieval model (Hiemstra and de Jong, 1999; Ture and Lin, 2013). These methods do not perform well due to the poor performance of statistical machine translation systems. Later work addresses this issue with bilingual embeddings (Vulic and Moens, 2015; Litschko et al., 2018). More recently, large pre-trained multilingual language models demonstrate significant advantages in constructing multilingual representations (Jiang et al., 2020). Yu et al. (2021) pre-train a cross-lingual language model tailored for the retrieval tasks. Yet, the computation complexity is relatively high due to the cross-encoder architecture of the model. Namely, it takes a pair of query and evidence as inputs, instead of encoding the query and evidence independently. The mDPR model presented in (Asai et al., 2021b) is the most favorable for our task due to its high efficiency and performance. However, mDPR was trained on datasets where queries are questions instead of claims. Therefore, domain adaptation is needed for mDPR to be applied to our task. CONCRETE extends mDPR by adapting it to fact-checking using a self-supervised cross-lingual retrieval algorithm to mitigate the domain discrepancy problem while maintaining high efficiency.

7 Conclusions and Future Work

We have proposed CONCRETE, a claim-oriented cross-lingual retriever that retrieves trustworthy passages from a multilingual passage collection. To overcome the lack of IR training data with claim-like queries, we present the Cross-lingual Inverse Cloze Task (X-ICT) that leverages pseudo feedback to train the retriever. Experimental results on X-FACT showed that our approach outperforms all previous systems in the *zero-shot* cross-lingual setting. For future work, we plan to investigate the adaptive selection mechanism for passages based on distances and develop more robust readers for reasoning through passages of longer distances via representation learning.

8 Ethical Considerations

Although our framework has significant advantages over the previous state of the art, the proposed model is still far from being a reliable cross-lingual fact checker given its great potential for improvement. If such a system is deployed for public use, the general public could lose trust in automatic fact-checking systems, and the situation of infodemic can exacerbate. Therefore, at the current stage, our system should be served as an assistant for human fact-checkers to validate the veracity of claims instead of directly applying for public use, especially in the *zero-shot* setting. With our framework, the efficiency of manual fact-checking can be significantly improved thanks to its ability to retrieve relevant information across multiple languages and produce a reasonably good preliminary judgement on the veracity of the input claim.

In addition, we also acknowledge that the use of large multilingual language models pre-trained on the Web could lead to biased outputs. Fortunately, after a close inspection into the X-FACT dataset, we do not find such biased patterns in it. This means that fine-tuning our proposed framework on X-FACT should alleviate the problem of biased predictions.

Acknowledgement

This research is based upon work supported by U.S. DARPA SemaFor Program No. HR001120C0123. The views and conclusions contained herein are those of the authors and should not be interpreted as necessarily representing the official policies, either expressed or implied, of DARPA, or the U.S. Government. The U.S. Government is authorized to reproduce and distribute reprints for governmental purposes notwithstanding any copyright annotation therein.

References

Wasi Ahmad, Zhisong Zhang, Xuezhe Ma, Eduard Hovy, Kai-Wei Chang, and Nanyun Peng. 2019. On difficulties of cross-lingual transfer with order differences: A case study on dependency parsing. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 2440–2452, Minneapolis, Minnesota. Association for Computational Linguistics.

Wasi Uddin Ahmad, Nanyun Peng, and Kai-Wei Chang. 2021. Gate: Graph attention transformer encoder for cross-lingual relation and event extraction. In *Proceedings of the AAAI Conference on Artificial Intelligence*.

Akari Asai, Jungo Kasai, Jonathan Clark, Kenton Lee, Eunsol Choi, and Hannaneh Hajishirzi. 2021a. XOR QA: Cross-lingual open-retrieval question answering. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 547–564, Online. Association for Computational Linguistics.

Akari Asai, Xinyan Yu, Jungo Kasai, and Hannaneh Hajishirzi. 2021b. One question answering model for many languages with cross-lingual dense passage retrieval. In *NeurIPS*.

Isabelle Augenstein, Christina Lioma, Dongsheng Wang, Lucas Chaves Lima, Casper Hansen, Christian Hansen, and Jakob Grue Simonsen. 2019. MultiFC: A real-world multi-domain dataset for evidence-based fact checking of claims. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 4685–4697, Hong Kong, China. Association for Computational Linguistics.

Taylor Berg-Kirkpatrick, David Burkett, and Dan Klein. 2012. An empirical investigation of statistical significance in NLP. In *Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning*, pages 995–1005, Jeju Island, Korea. Association for Computational Linguistics.

Anshika Choudhary and Anuja Arora. 2021. Linguistic feature based learning model for fake news detection and classification. *Expert Systems with Applications*, 169:114171.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, Minneapolis, Minnesota. Association for Computational Linguistics.

Ogi Djuraskovic. 2022. 47 blogging statistics (2022): How many blogs are there?

Yi Fung, Christopher Thomas, Revanth Gangi Reddy, Sandeep Polisetty, Heng Ji, Shih-Fu Chang, Kathleen McKeown, Mohit Bansal, and Avi Sil. 2021. InfoSurgeon: Cross-media fine-grained information consistency checking for fake news detection. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 1683–1698, Online. Association for Computational Linguistics.

Yi R. Fung, Kung-Hsiang Huang, Preslav Nakov, and Heng Ji. 2022. The battlefront of combating misinformation and coping with media bias. In *Proceedings of the 28th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, KDD '22, page 4790–4791, New York, NY, USA. Association for Computing Machinery.

Zhijiang Guo, Michael Schlichtkrull, and Andreas Vlachos. 2022. A Survey on Automated Fact-Checking. *Transactions of the Association for Computational Linguistics*, 10:178–206.

Ashim Gupta and Vivek Srikumar. 2021. X-fact: A new benchmark dataset for multilingual fact checking. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 2: Short Papers)*, pages 675–682, Online. Association for Computational Linguistics.

Djoerd Hiemstra and Franciska de Jong. 1999. Disambiguation Strategies for Cross-Language Information Retrieval. In *Proceedings of the Third European Conference on Research and Advanced Technology for Digital Libraries*, volume 1696 of *Lecture Notes in Computer Science*, pages 274–293. Springer.

Thi Bich Ngoc Hoang and Josiane Mothe. 2018. Location extraction from tweets. *Information Processing & Management*, 54(2):129–144.

Kung-Hsiang Huang, Kathleen McKeown, Preslav Nakov, Yejin Choi, and Heng Ji. 2022. Faking fake news for real fake news detection: Propaganda-loaded training data generation. *arXiv preprint arXiv:2203.05386*.

Zhuolin Jiang, Amro El-Jaroudi, William Hartmann, Damianos Karakos, and Lingjun Zhao. 2020. Cross-lingual information retrieval with BERT. In *Proceedings of the workshop on Cross-Language Search and Summarization of Text and Speech (CLSSTS2020)*, pages 26–31, Marseille, France. European Language Resources Association.

Vladimir Karpukhin, Barlas Oguz, Sewon Min, Patrick Lewis, Ledell Wu, Sergey Edunov, Danqi Chen, and Wen-tau Yih. 2020. Dense passage retrieval for open-domain question answering. In *Proceedings of the 2020 Conference on Empirical Methods in Natural*

Language Processing (EMNLP), pages 6769–6781, Online. Association for Computational Linguistics.

Ashkan Kazemi, Kiran Garimella, Devin Gaffney, and Scott Hale. 2021. Claim matching beyond English to scale global fact-checking. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 4504–4517, Online. Association for Computational Linguistics.

Ashkan Kazemi, Zehua Li, Verónica Pérez-Rosas, Scott A Hale, and Rada Mihalcea. 2022. Matching tweets with applicable fact-checks across languages. *arXiv preprint arXiv:2202.07094*.

Maurice G Kendall. 1938. A new measure of rank correlation. *Biometrika*, 30(1/2):81–93.

Kenton Lee, Ming-Wei Chang, and Kristina Toutanova. 2019. Latent retrieval for weakly supervised open domain question answering. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 6086–6096, Florence, Italy. Association for Computational Linguistics.

Nayeon Lee, Yejin Bang, Andrea Madotto, and Pascale Fung. 2021. Towards few-shot fact-checking via perplexity. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 1971–1981, Online. Association for Computational Linguistics.

Robert Litschko, Goran Glavas, Simone Paolo Ponzetto, and Ivan Vulic. 2018. Unsupervised cross-lingual information retrieval using monolingual data only. *The 41st International ACM SIGIR Conference on Research & Development in Information Retrieval*.

Ilya Loshchilov and Frank Hutter. 2019. Decoupled weight decay regularization. In *International Conference on Learning Representations*.

Rada Mihalcea and Carlo Strapparava. 2009. The lie detector: Explorations in the automatic recognition of deceptive language. In *Proceedings of the ACL-IJCNLP 2009 Conference Short Papers*, pages 309–312, Suntec, Singapore. Association for Computational Linguistics.

Umaru A Pate, Danjuma Gambo, Adamkolo Mohammed Ibrahim, et al. 2019. The impact of fake news and the emerging post-truth political era on nigerian polity: A review of literature. *Studies in Media and Communication*, 7(1):21–29.

Ronak Pradeep, Xueguang Ma, Rodrigo Nogueira, and Jimmy Lin. 2021. Scientific claim verification with VerT5erini. In *Proceedings of the 12th International Workshop on Health Text Mining and Information Analysis*, pages 94–103, online. Association for Computational Linguistics.

Stephen Robertson and Hugo Zaragoza. 2009. *The probabilistic relevance framework: BM25 and beyond*. Now Publishers Inc.

Dietram A Scheufele and Nicole M Krause. 2019. Science audiences, misinformation, and fake news. *Proceedings of the National Academy of Sciences*, 116(16):7662–7669.

Yuqing Tang, Chau Tran, Xian Li, Peng-Jen Chen, Naman Goyal, Vishrav Chaudhary, Jiatao Gu, and Angela Fan. 2020. Multilingual translation with extensible multilingual pretraining and finetuning.

James Thorne, Andreas Vlachos, Christos Christodoulopoulos, and Arpit Mittal. 2018. FEVER: a large-scale dataset for fact extraction and VERification. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 809–819, New Orleans, Louisiana. Association for Computational Linguistics.

Ferhan Ture and Jimmy Lin. 2013. Flat vs. hierarchical phrase-based translation models for cross-language information retrieval. In *Proceedings of the 36th international ACM SIGIR conference on Research and development in information retrieval*, pages 813–816.

Ivan Vulic and Marie-Francine Moens. 2015. Monolingual and cross-lingual information retrieval models based on (bilingual) word embeddings. *Proceedings of the 38th International ACM SIGIR Conference on Research and Development in Information Retrieval*.

David Wadden, Shanchuan Lin, Kyle Lo, Lucy Lu Wang, Madeleine van Zuylen, Arman Cohan, and Hannaneh Hajishirzi. 2020. Fact or fiction: Verifying scientific claims. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 7534–7550, Online. Association for Computational Linguistics.

William Yang Wang. 2017. "liar, liar pants on fire": A new benchmark dataset for fake news detection. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 422–426, Vancouver, Canada. Association for Computational Linguistics.

Xueqing Wu, Kung-Hsiang Huang, Yi Fung, and Heng Ji. 2022. Cross-document misinformation detection based on event graph reasoning. In *Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 543–558, Seattle, United States. Association for Computational Linguistics.

Wenpeng Yin and Dan Roth. 2018. TwoWingOS: A two-wing optimization strategy for evidential claim verification. In *Proceedings of the 2018 Conference*

on Empirical Methods in Natural Language Processing, pages 105–114, Brussels, Belgium. Association for Computational Linguistics.

Puxuan Yu, Hongliang Fei, and P. Li. 2021. Cross-lingual language model pretraining for retrieval. *Proceedings of the Web Conference 2021*.

E-VarM: Enhanced Variational Word Masks to Improve the Interpretability of Text Classification Models

Ling Ge[1], Chunming Hu[1,2,*], Guanghui Ma[1], Junshuang Wu[4], Junfan Chen[1], Jihong Liu[5], Hong Zhang[3], Wenyi Qin[1], Richong Zhang[1]
[1] School of Computer Science and Engineering, Beihang University, Beijing, China
[2] College of Software, , Beihang University , Beijing, China
[3] National Computer Network Emergency Response Technical Team, Beijing, China
[4] Beijing Jinghang Research Institute of Computing and Communication, Beijing, China
[5] School of Mechanical Engineering and Automation, Beihang University, Beijing, China
{geling, hucm}@buaa.edu.cn, {magh, chenjf, zhangrc}@act.buaa.edu.cn,
zhangh@isc.org.cn, ryukeiko@buaa.edu.cn,wenyi.qin@hotmail.com

Abstract

Enhancing the interpretability of text classification models can help increase the reliability of these models in real-world applications. Currently, most researchers focus on extracting task-specific words from inputs to improve the interpretability of the model. The competitive approaches exploit the Variational Information Bottleneck (VIB) to improve the performance of word masking at the word embedding layer to obtain task-specific words. However, these approaches ignore the multi-level semantics of the text, which can impair the interpretability of the model, and do not consider the risk of representation overlap caused by the VIB, which can impair the classification performance. In this paper, we propose an enhanced variational word masks approach, named E-VarM, to solve these two issues effectively. The E-VarM combines multi-level semantics from all hidden layers of the model to mask out task-irrelevant words and uses contrastive learning to readjust the distances between representations. Empirical studies on ten benchmark text classification datasets demonstrate that our approach outperforms the SOTA methods in simultaneously improving the interpretability and accuracy of the model.

1 Introduction

With the widespread adoption of neural networks in text classification tasks (Bastings and Filippova, 2020; Halder et al., 2020; Schick et al., 2020; Lv et al., 2021), the classification models are expected to provide not only highly accurate classification results but also reasonable prediction rationales (Peake and Wang, 2018; Sun et al., 2021). These prediction rationales, which are short yet informative parts of the input for classification predictions (Bastings et al., 2019), manifest the interpretability of the model. The better the interpretability of the model, the more reasonable rationales the model provides (Lin et al., 2021). Therefore, improving the interpretability of the model helps to boost the reliability of these classifiers in real-world applications (Jacovi and Goldberg, 2020).

To improve the interpretability of the classifiers, many methods have been proposed (Chrysostomou and Aletras, 2021; Bastings et al., 2019). Some studies rely on additional inputs at the training time, such as pre-defined information and human annotations (Erion et al., 2019; Plumb et al., 2020), all of which involve high human costs. Other works resort to assigning a binary Bernoulli variable to each input word with promising results, among which the competitive approaches are the Sparse-VIB (Paranjape et al., 2020) and Vmask (Chen and Ji, 2020) models . They all employ the Variational Information Bottleneck (VIB) (Alemi et al., 2017) to train stochastic masks to automatically learn task-specific words, namely prediction rationales, for prediction and interpretability simultaneously.

However, these methods only utilize the information from the word embedding layer , which holds little task-specific information (Van Aken et al., 2019), ignoring the multi-level information of the text, such as syntactic and semantic information. That is, this initial layer results in poor interpretability performance and reaches low accuracy on related semantic tasks . Additionally, these approaches do not consider the risk of representation overlap arising from VIB (Alemi et al., 2017), resulting in the model failing to learn discriminative class representations, thus further impairing

*Corresponding Author

Proceedings of the 29th International Conference on Computational Linguistics, pages 1036–1050
October 12–17, 2022.

the classification performance.

To address the above issues, we propose an **E**nhanced **Var**iational Word **M**asks approach, (named E-VarM), which combines multi-level information to learn task-specific words in an unsupervised manner and uses contrastive learning to alleviate the representation overlap problem caused by VIB. As a result, both the interpretability and accuracy of the model are improved.

Specifically, as the data flow from the input to the output layer, each hidden layer of the text encoder can capture different information (Geirhos et al., 2019). For example, the BERT (Devlin et al., 2018) can encode the captured information to a rich hierarchy of linguistic information (Hewitt and Manning, 2019; Li et al., 2022) with surface features at the bottom, syntactic features in the middle, and semantic features at the top (Jawahar et al., 2019; van Aken et al., 2019; Kim et al., 2020; Gupta et al., 2022). Inspired by the feature extraction mechanism of the text encoder, we fuse multilevel information from all hidden layers of the model to generate a task-specific binary Bernoulli distribution to extract reasonable interpretations for various text classification tasks, such as sentiment analysis, syntactic judgments, and semantic inference.

Furthermore, the VIB-based methods rely on the Lagrangian factor to perform the compression-prediction trade-off (Tishby et al., 2000; Mahabadi et al., 2021). A larger Lagrangian factor means masking out more task-irrelevant information (Pan et al., 2021; Kolchinsky et al., 2019; Gálvez et al., 2020) for a special class and obtaining better interpretability. However, along with this larger value is the potential risk of representation overlap or even collapse (Alemi et al., 2017; Goldfeld and Polyanskiy, 2020; Wu et al., 2020) since it may hinder the model from learning the discriminative representation and lead to loss of class information. Therefore, we leverage supervised contrastive learning (Khosla et al., 2020) to adjust the classification representation by pulling in samples from the same class and pushing away samples from different classes, thus mitigating the representations overlap problem. Additionally, we resort to task-specific words to construct diverse positive samples to enhance the efficiency of contrastive learning.

In a nutshell, we make the following major contributions: (1) We introduce an enhanced variational word masks method to improve classification performance and interpretability simultaneously. (2) Our method produces token-level interpretations that consider the multi-level information from all model layers and are adaptable to various classification tasks. (3) To the best of our knowledge, our approach is the first attempt to use contrastive learning to alleviate the representation overlap caused by VIB. (4) The experimental results on ten benchmark datasets validate the effectiveness of our method.

2 Related Works

2.1 Model Interpretability

Various approaches have been proposed to improve the interpretability of neural networks, such as exploiting attention distribution on tokens (Sun et al., 2020), extracting subsets of input text (Swanson et al., 2020), or relying on language models to generate explanations from scratch (Rajani et al., 2019). The underlying techniques in this paper are closely related to the extracted interpretation methods. Some of these works leverage pre-collected annotations (Erion et al., 2019; Plumb et al., 2020), which can be labor-intensive. Other works use random masks on the input to automatically learn interpretable models, for example, Bastings et al. (2019) and Cao et al. (2020) adopt $L0$ constraints to make the masks sparse.

The $L0$ regularisation overemphasizes the sparsity, which can damage the accuracy of model. Therefore, some researchers introduce VIB to control the level of mask sparsity through a tunable sparse prior (Paranjape et al., 2020). Among these, the competitive methods are the SparseVIB (Paranjape et al., 2020), VIBI (Bang et al., 2021), and Vmask (Chen and Ji, 2020). However, these methods only consider the information of word embeddings to extract task-specific tokens and ignore the risk of representation overlap, which affects the models' interpretability and classification performance. This paper aims to address the above issues.

2.2 Supervised Contrastive Learning

Supervised contrastive learning performs representation learning by expanding the embedding differences of instances from different classes in the hidden space (Khosla et al., 2020), which is widely used in sentiment recognition (Liang et al., 2021) , semantic inference (Zhang et al., 2021) and text classification (Gunel et al., 2021) with good results. Previous works (Wu et al., 2021; Robinson et al., 2021) indicate that constructing diverse posi-

tive and negative samples facilitates discriminative feature learning. The reason is that high-quality contrastive samples encourage the model to mine both intra-class and inter-class features, forcing the model to perform fine-grained representation learning (Khosla et al., 2020). Therefore, we use contrastive learning to adjust representations to mitigate the risk of overlap and create high-quality positive samples based on task-specific words to enhance the model's classification performance.

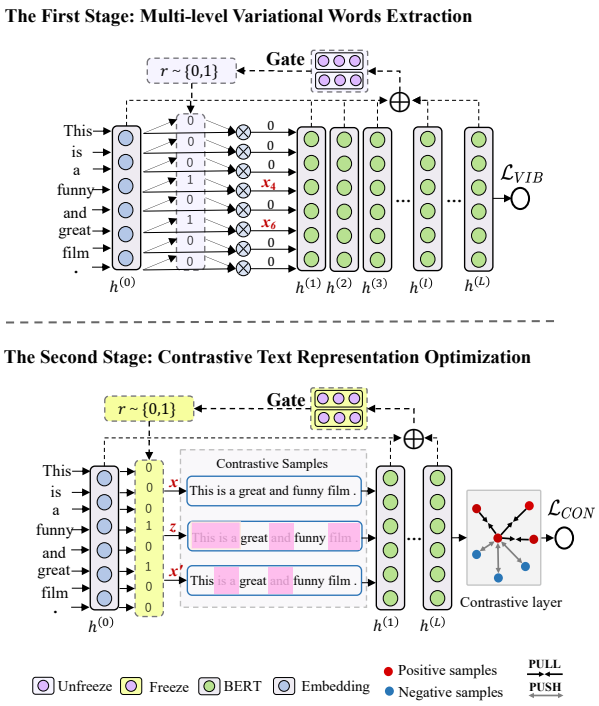

Figure 1: An illustration of our proposed method. The x, z, and x' are contrastive samples in the form of word embeddings. We use tokens to replace embeddings to better indicate the diversity among samples. In this case, the input sentence is from the sentiment analysis dataset and labeled as positive.

3 Methodology

3.1 Overview

This paper aims to simultaneously improve the interpretability and accuracy of the text classifier. Our model consists of two stages shown in Fig 1. The first one is 1) the multi-layer variational words extraction, which automatically extracts task-specific words as rationales. Another is 2) the contrastive text representation optimization, which mitigates the feature overlap of different classes and learns discriminative representations.

To optimize these two stages, we adopt an iterative training mechanism at the batch level for the whole model. With this mechanism, the model can be corrected in time if the representation overlap occurs in the extraction stage.

3.2 Multi-level Variational Words Extraction

The core idea of this stage is that if a subset of an input text can be removed without affecting the prediction, this subset text is considered task-irrelevant, while the remaining subset text is task-specific. Generally, this stage aims to learn a sparse random gate g_ϕ(a.k.a., masks) for a neural text classifier $f_\theta(\cdot)$ to obtain task-specific words. In this paper, we use BERT as the text classifier.

Specifically, given an input $x = \{x_1, x_2, ..., x_n\}$, where $x_t \in \mathbb{R}^d$ indicates the word embedding, we feed it into the encoder to obtain the hidden states $\{h^{(0)}, ..., h^{(L)}\}$ of different layers of BERT. In this hierarchy of linguistic information structure encoded by BERT, surface features are at the bottom, syntactic features are in the middle, and semantic features are at the top. Herein, we stack the hidden states up to $h(L)$ as input to the gate network , a two-hidden-layer MLP, to predict the binary output r,

$$r = g(h^{(0)} \oplus h^{(1)} ... \oplus h^{(L)}) \tag{1}$$

where $\oplus$ denotes concatenate operation. We rely on $r \in \mathbb{R}^{2n}$ to perform word masks, so that the model has multi-level information and is adaptable to various tasks. The r_i associated with x_i follows the Bernoulli distribution,

$$r_i \sim Bernoulli(\alpha_i) \tag{2}$$

where α_i is the probability of the word embedding x_i being selected as the task-specific word. Precisely, the binary r produced by the gate can be expressed as the corresponding two-element one-hot vector $\boldsymbol{r_i} = [r_{i,j}]_{j=0,1}$, where $r_{i,j} = 1$ means $r_i = j$. Similarly, $\alpha_{i,j}$ is the probability that $r_i = j$ (Xue et al., 2020),

$$\begin{aligned} \boldsymbol{r_i} &= \text{one\_hot}(\arg\max_{\mathrm{j}} \alpha_{\mathrm{i,j}}, \mathrm{j} = 0, 1) \\ \alpha_{i,0} &= 1 - \alpha_i, \alpha_{i,1} = \alpha_i \end{aligned} \tag{3}$$

However, sampling from the Bernoulli distribution (Equation 2) causes the non-differentiability problem (Xue et al., 2020). We adopt the Gumbel-Softmax distribution (Xue et al., 2020; Jang et al., 2017) to approximate r_i differentiably,

$$\hat{\boldsymbol{r}}_{\boldsymbol{i}} = [\hat{\alpha}_{i,j}]_{j=0,1}$$
$$\hat{\alpha}_{i,j} = \frac{exp((\log(\alpha_{i,j}) + \epsilon_j)/\tau)}{\sum_{k=0}^{1} exp((log(\alpha_{i,k}) + \epsilon_k)/\tau)} \quad (4)$$

where ϵ_j is sampled randomly from Gumbel(0, 1), and the temperature τ controls how closely the Gumbel-Softmax distribution approximates the one-hot. By replacing Equation 2 with $r_i = \hat{r}_{i,0}$, we can train the model in an end-to-end manner.

Finally, we obtain the task-specific word embeddings z as the final input for prediction, i.e.,

$$z = r \odot x \quad (5)$$

where $\odot$ is an element-wise multiplication and z is a subset of x.

Since there is no direct supervision signal to estimate the gate parameters ϕ, we follow the standard practice in the Information Bottleneck(IB) (Tishby et al., 2000) theory to optimize the parameters. For the input x and its label y, the IB principle aims to learn the minimal sufficient representation z that preserves enough information about the output y (prediction) while containing the least redundant information from input x (compression) (Alemi et al., 2017; Mahabadi et al., 2021),

$$\mathcal{L}_{IB} = \ -I(z, y) + \beta \cdot I(x, z) \quad (6)$$

where β is the Lagrangian factor to balance the compression and prediction, and $I(.,.)$ is the mutual information.

To compute the two mutual information items above, the Deep VIB(Alemi et al., 2017) perform a variational approximation for the IB objective via a neural network. Thus, we obtain $\mathcal{L}_{VIB}$, which is the approximation of $\mathcal{L}_{IB}$,

$$\mathcal{L}_{VIB} = -\sum_{i=0}^{n} p(z|x^{(i)}) \log q(y^{(i)}|\boldsymbol{z}) + \beta \cdot \boldsymbol{KL}[p(z|x^{(i)})||m(z))] \quad (7)$$

where $q(y^{(i)}|z)$ is a parametric approximation of $p(y^{(i)}|z)$, $m(z)$ is a variational estimate of the prior probability $p(z)$ of z, and $p(z|x^{(i)})$ is an estimate of the posterior probability of z. The $\boldsymbol{KL}[\cdot||\cdot]$ denotes Kullback-Leibler divergence, and the β is inherited from the IB theory.

Since the compressed features z is determined by the random variable r that follows the Bernoulli distribution (Equation 5), we can rewrite Equation 7 with r. Motivated by VMask (Chen and Ji, 2020) and mean-field approximation (Tanaka, 1998), we obtain $m(r) = \Pi_{i=1}^{n} m(r_i)$. As $m(r_i) = Bernoulli(0.5)$ means that each word has an equal probability to be masked or selected (Chen and Ji, 2020), we get the uniform distribution $m(r)$ and Equation 7 can be further simplified as,

$$\mathcal{L}_{VIB} = -\sum_{i=0}^{n} p(r|x^{(i)}) \log q(y^{(i)}|r, x^{(i)}) + \beta \cdot \sum_{i=0}^{n} p(r|x^{(i)}) \log p(r|x^{(i)}) \quad (8)$$

The first term in $\mathcal{L}_{VIB}$ is a cross-entropy aiming to make sure the information in $p(r|x)$ for predicting is sufficient. The second term in $\mathcal{L}_{VIB}$ is to regularize $p(r|x)$ to make masks sparse, enabling the r vector to contain more zeros. To compute the compressed posterior $p_{\theta,\phi}(r|x^{(i)})$, we first feed input embedding x to BERT model $f_\theta(\cdot)$ and then resort to the amortization network (Rezende and Mohamed, 2015; Chen and Ji, 2020) , which is our gate network g_ϕ, to output binary value r. Then the BERT takes r along with $x^{(i)}$ as input and produces a probability of output $y^{(i)}$, $q_\theta(y^{(i)}|r, x^{(i)})$.

In actual training, we use the $\mathcal{L}_{VIB}$ to simultaneously optimize the gate parameter ϕ and the classifier parameters θ.

3.3 Contrastive Text Representation Optimisation

In this stage, we leverage supervised contrastive learning to mitigate the risk of representation overlap.

Specifically, for a given input x, we obtain a compressed sample z that contains only the task-specific embeddings through the first stage. To perform supervised contrastive learning, we treat the samples belonging to the same class in a batch as positive samples and the rest within the batch as negative samples. Additionally, to increase the diversity of the contrastive samples, we introduce additional positive samples x',

$$x' = x - R(x - z) \quad (9)$$

where $R(x - z)$ denotes a random mask operation on task-irrelevant embeddings, and the number of masks ranges from 1 to m, with m being the number of class-irrelevant words of input x. Since we

Table 1: Statistics of the datasets. "Class" is the number of labels. "Ave.Len" represents the average length of a sentence. "Train", "Dev" and "Test" denote the size of the training set, validation set and test set, respectively.

Num	Dataset	Class	Ave.Len	Train	Dev	Test
1	QNLI	2	40	104K	5463	-
2	QQP	2	23	363K	40k	-
3	COLA	2	8	8551	1043	-
4	IMDB	2	268	25K	-	25K
5	AGNews	4	32	120K	-	7.6K
6	Yelp	2	138	560K	-	38K
7	Subj	2	23	10K	-	-
8	RT	2	23	10K	-	-
9	SST-1	5	18	8540	1101	2208
10	SST-2	2	19	6920	872	1821

deliberately avoid a mask on task-specific features when perturbing the input x, the label of x' is not changed. These diverse contrastive samples, x, x' , and z, can force the model to focus on task-specific words and increase contrastive learning efficiency.

Then, we freeze the gate network and use the above samples to optimize text representations. Given a sample representation h_i , its positive representation h_j and negative representation h_k , the supervised contrastive learning loss is presented as,

$$\mathcal{L}_{CON} = \sum_{i=1}^{3N} \frac{-1}{3N_{\tilde{y}_i} - 1} \sum_{j=1}^{3N} \mathbb{I}_{i \neq j} \cdot \mathbb{I}_{\tilde{y}_i = \tilde{y}_j} \cdot log \frac{exp(h_i \cdot h_j / \tau)}{\sum_{k=1}^{3N} \mathbb{I}_{i \neq k} \cdot exp((h_i \cdot h_k / \tau))} \tag{10}$$

where $i, j, k \in \{1, 2, \cdots, 3N\}$ and these 3N samples consist of N x samples, N x' samples, and N z samples. The τ is a temperature controlling the concentration level of the distribution (Hinton et al., 2015).

Finally, we obtain the contrastive loss $\mathcal{L}_{CON}$ for the second stage to fine-tune the BERT encoder.

4 Experimental Settings

4.1 Dataset

We adopt ten benchmark datasets to evaluate our model, ranging from sentiment classification, and syntactic judgment to semantic inference, etc., which includes five sentiment analysis datasets: **RT** (Pang and Lee, 2005a), **IMDB** (Maas et al., 2011), **SST-2** (Socher et al., 2013), **Yelp** (Zhang et al., 2015), **SST-1** (Socher et al., 2013); two semantic inference datasets: **QQP** (Wang et al., 2018), **QNLI** (Wang et al., 2018); one topic categorization dataset:**AG's News** (Zhang et al., 2015); one grammatical judgment dataset: **COLA** (Wang et al., 2018); and one subjective / objective classification dataset: **Subj** (Pang and Lee, 2005b). The statistics of the datasets are displayed in Table 1.

Since QNLI, QQP, and COLA datasets have no test set, we randomly select 20% of the training set from each of them as their respective test set. Similarly, for IMDB, AGNews, and Yelp datasets, we randomly selecte 20% of the training data as their respective validation set. For Subj and RT datasets, which have no both validation and test set, we randomly select 10% of their respective training data as their validation and test set, respectively. More details about the datasets can be found in the **Appendix A**.

4.2 Baseline and SOTAs

We conduct experiments with seven competitive models, one baseline model: **BERT**$_{Base}$ (Devlin et al., 2018), three SOTA models, and three variants of our model.

AGN (Li et al., 2021): An adaptive gate-based SOTA model that improves performance by computing corpus-specific features such as word frequency and label distribution.

SCL (Gunel et al., 2021): A contrastive learning-based SOTA model that utilizes supervised contrastive learning to obtain promising features and enhance performance.

Vmask (Chen and Ji, 2020): An interpretation-based SOTA model that improves accuracy and interpretation by masking task-irrelevant words at the word embedding layer with the VIB.

E-VarM$_{MASK}$: A variant of E-VarM that removes the representation optimization stage and contains only the variational word extraction stage.

E-VarM$_{\text{PPL}}$: A variant of E-VarM that uses a two-stage pipeline training method instead of the iterative one. We first train the gate and the Bert jointly for 50 epochs to extract class-specific words. Then we fix the gate parameters and tune the Bert for 50 epochs via contrastive learning.

E-VarM$_{\text{ETE}}$: The third variant of E-VarM that uses an end-to-end training method to update both the gate and the BERT model. The corresponding loss function $\mathcal{L}_{ETE}$ can be expressed formally as,

$$\mathcal{L}_{ETE} = \mathcal{L}_{VIB} + \mathcal{L}_{CON} \tag{11}$$

where $\mathcal{L}_{VIB}$ is the VIB loss shown in Equation

Table 2: The prediction accuracy (%) comparison of different methods. The best results are marked in bold.

Methods	QNLI	QQP	COLA	IMDB	AGNews	Yelp	Subj	RT	SST-1	SST-2
BERT_{Base}	87.07	90.17	82.45	91.75	93.09	96.30	96.40	86.59	50.81	90.44
AGN	86.95	90.23	81.20	92.66	93.59	97.00	95.61	85.56	50.70	90.57
SCL	86.57	89.26	82.46	91.78	93.61	96.40	95.33	86.65	50.59	90.72
Vmask	85.03	86.04	79.41	92.06	93.52	96.30	96.68	87.82	51.99	91.21
E-VarM	**87.94**	**91.08**	**83.99**	**92.73**	**94.00**	**97.28**	**97.30**	**88.67**	**53.44**	**92.37**
Variants										
$\text{E-VarM}_{\text{MASK}}$	86.00	87.21	80.19	92.67	93.58	96.32	96.90	88.00	51.96	92.04
$\text{E-VarM}_{\text{ETE}}$	71.44	79.54	69.70	85.72	90.78	93.92	93.80	82.94	46.87	87.04
$\text{E-VarM}_{\text{PPL}}$	87.59	89.89	82.64	92.76	94.00	96.64	96.80	88.19	52.67	91.76

8 and $\mathcal{L}_{CON}$ is the contrastive loss shown in Equation 10.

4.3 Evaluation Metrics

In our experiments, we chose accuracy as evaluation metric for the classification performance. To assess the interpretability of different models, we follow the previous work (Chen and Ji, 2020) and choose Area of Perturbation Curve (AOPC) (Nguyen, 2018) and post-hoc accuracy (Chen et al., 2018) as the local interpretability and the global interpretability metrics, respectively.

The Local Interpretability (AOPC) : The AOPC metric calculates the average change in prediction probability over all classes by removing the top K most important words from the input. To provide a fair assessment for all compared methods, we utilize LIME (Ribeiro et al., 2016) to extract the nine most important words. Specifically, the LIME is a local interpretable algorithm that can extract local explanations for a classifier by fitting the local decision boundary of an instance under test (Ribeiro et al., 2016).

The AOPE metric is then calculated as follows,

$$AOPC = \frac{1}{K+1}\langle \sum_{k=1}^{K} (p(\hat{y}|x) - p(\hat{y}|x_{\setminus 1\cdots k})) \rangle_{p(x)} \tag{12}$$

where $p(\hat{y}|x_{\setminus 1\cdots k})$ is the probability for the predicted class with words 1..K removed and $\langle\cdot\rangle_{p(x)}$ denotes the average over all examples.

The Global Interpretability (the post-hoc Accuracy) : The post-hoc accuracy assesses the sufficiency of important words selected from the input for model prediction and is calculated as follows,

$$ACC_{post-hoc}(k) = \frac{1}{N}\sum_{i=1}(\mathbb{I}[\hat{y}(x_i^{(k)}) = \hat{y}(x_i)]) \tag{13}$$

where $\mathbb{I}[\cdot]$ is an indicator function. $\hat{y}(x_i)$ represents the prediction label of sample x_i and $\hat{y}(x_i^{(k)})$ means the prediction label obtained using the most important k words from x_i.

4.4 Implementation Details

We use the pre-trained BERT (Wolf et al., 2019) as the classifier and adopt a batch-level iterative mechanism to train the model, with each iteration consisting of two stages. In the first stage, we train the gate and BERT model jointly, and for the second stage, we freeze the gate and train only the BERT model. Both stages are performed using the Adam optimizer(Kingma and Ba, 2015) with learning rate = 1e-5, batch size = 64, and the dropout=0.2 empirically.

We utilize the grid search technology to obtain the optimal super-parameters, including the Lagrangian multiplier β and the contrastive learning temperature τ. The β is selected from {0.1,0.5, 1, 10, 50},while the τ is chosen from 0.1 to 0.9.

In order to compare the performance of the different models, We evaluate the AGN and Vmask using the open-source code [1] and [2] respectively. Since the source code for SCL is not provided, we implement and evaluate this method as described in the original paper. Additionally, our approach is implemented using PyTorch, and all calculations are done on NVIDIA Tesla V100 GPU, with per experiment taking approximately 1∼3 hours.

[1] https://github.com/4AI/AGN
[2] https://github.com/UVa-NLP/VMASK

Table 3: The AOPC accuracy(%) comparison of different methods. The best results are marked in bold.

k	Method	QNLI	QQP	RT	COLA	IMDB	SST1	SST2	AGNews	Subj	Yelp
1	Vmask	3.44	4.61	7.19	5.61	3.38	7.80	10.52	2.34	**2.31**	1.15
	E-VarM	**6.27**	**6.37**	**8.18**	**8.57**	**4.25**	**8.86**	**11.16**	**3.08**	1.94	**1.61**
5	Vmask	14.09	13.22	25.16	13.02	13.29	17.73	33.53	9.76	**13.75**	6.17
	E-VarM	**21.04**	**16.23**	**28.58**	**17.83**	**13.52**	**19.67**	**35.43**	**12.69**	13.14	**8.29**
9	Vmask	18.65	15.04	31.84	15.05	19.89	19.20	37.78	15.60	22.97	9.55
	E-VarM	**26.31**	**17.54**	**36.70**	**22.50**	**20.47**	**21.34**	**39.49**	**19.63**	**23.32**	**12.61**

5 Experimental Results

5.1 The Classification Performance

As shown in Table 2, our approach achieves the best results in most datasets, proving its strength in various text classification tasks. More specifically, we can draw the following conclusions.

Compared with the gate-based model. AGN exploits an adaptive gate to obtain data-specific prior distributions to boost its accuracy. However, the prior distribution of the data does not always contribute to the model's performance, leading to inferior results. In contrast, our method always concentrates on task-specific words and is not restricted by the prior distribution of the data.

Compared with the contrastive learning-based model. Our method has enhancements over SCL on all datasets due to the high-quality positive samples. These samples, created with the help of task-specific words, will increase the challenge of contrastive learning and facilitate the model to learn better class discriminative representations.

Compared with the interpretable-based model. Our approach outperforms Vmask on most datasets, in particular, with obvious improvements on the QNLI, QQP, and COLA datasets. Since Vmask only uses word embedding information to select important words, it has poor selection decisions on tasks that rely on mid-level or high-level information, such as semantic reasoning (QNLI, QQP) and syntactic judgment (COLA), decreasing the classification performance.

Compared with the variants. When removing the representation optimization stage, E-VarM$_{\mathrm{MASK}}$ shows a performance degradation on all datasets compared to E-VarM, illustrating that contrastive learning can improve prediction performance. Since the gate network of E-VarM$_{\mathrm{ETE}}$ is adversely affected by contrastive loss, the performance of E-VarM$_{\mathrm{ETE}}$ decreases substantially compared to E-VarM. As the encoder of E-VarM$_{\mathrm{PPL}}$ cannot be corrected in time if there is a representation overlap in the extraction stage, so that the performance of E-VarM$_{\mathrm{PPL}}$ slightly drops compared to E-VarM. In contrast, E-VarM uses a two-task iterative way at the batch level, where words extraction is performed and immediately followed by the representation adjustment within a batch.

5.2 The Interpretability

Because of the computational cost, we select 500 samples randomly from the test set to evaluate the interpretability of the models.

The local interpretability (AOPC). We remove k task-specific words from the input for the AOPC experiment (Equation 12) with k selected from $\{1, 5, 9\}$. As shown in Table 3 , the AOPC of E-VarM outperforms Vmask on almost all datasets. On the four datasets: AG's News, COLA, QQP, and QNLI, E-VarM shows a noticeable improvement over Vmask. For example, it outperforms Vmask by 7% at K=5 on QNLI. These four datasets involve complex tasks, such as topic classification, semantic inference, and grammar judgment, and rely on medium or high-level information to select task-specific words as explanations. E-VarM integrates multi-level information for words selection and has good interpretability for such complex tasks. Even for datasets that rely on shallow features to mask, such as SST2, and IMDB datasets, E-VarM can slightly outperform Vmask by about 2% on average.

The global interpretability (Post-hoc accuracy). To compute the post-hoc accuracy of models, we use the top k task-specific words from the input for prediction and compare it with the result from the whole input (Equation 13), with k ranging from 1 to 9. As shown in Fig 2, the global interpretability of E-VarM outperforms Vmask on almost all datasets, with the best performance on the COLA dataset. On datasets that require multi-level information for masking, such as QNLI, QQP,

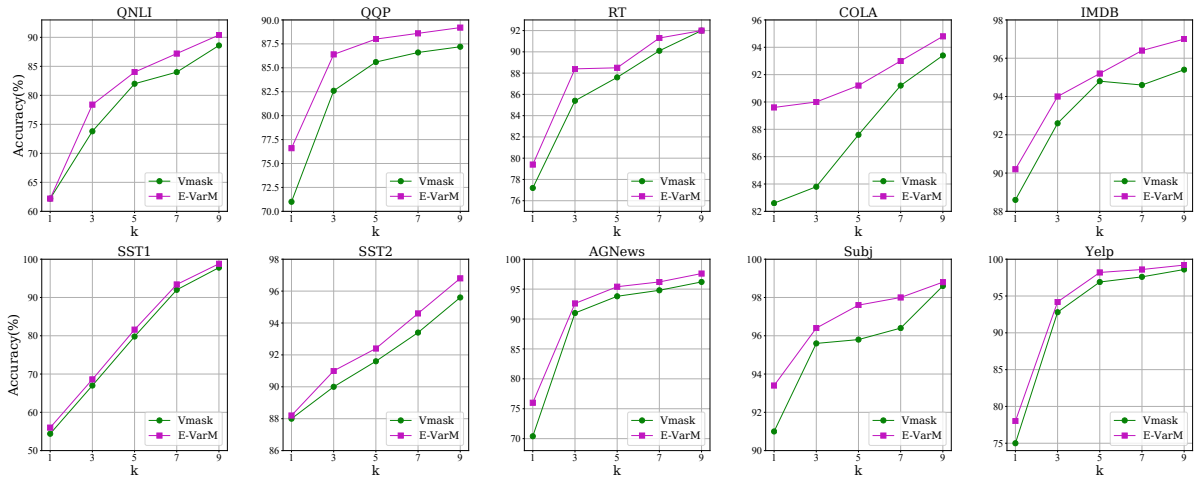

Figure 2: The post-hoc accuracy(%) comparison of different methods on ten datasets.

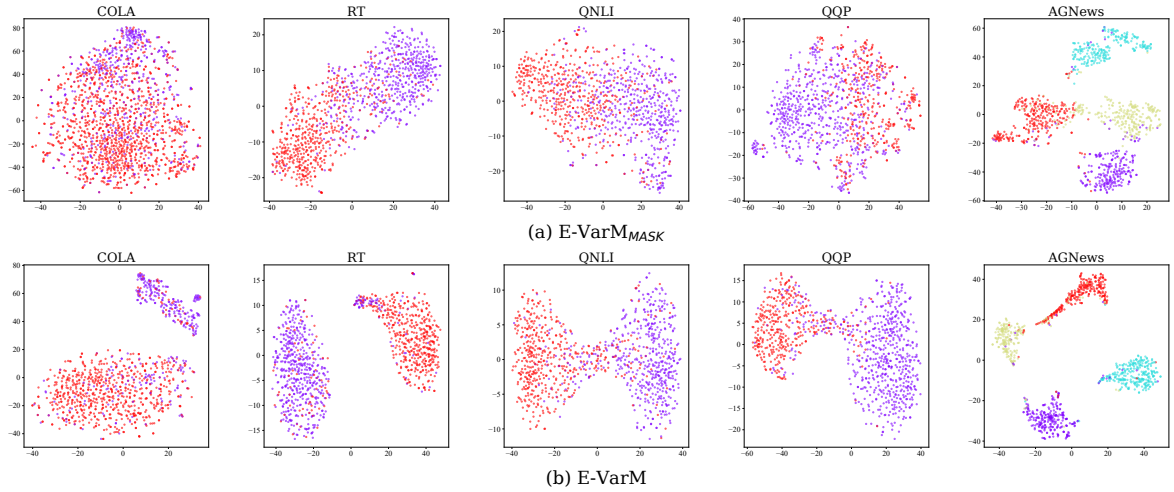

(a) E-VarM$_{MASK}$

(b) E-VarM

Figure 3: The TSNE visualization comparison of text representations of different models.

and RT, there is a gap between Vmask and E-VarM, which further illustrates the superiority of our model in handling complex tasks. Also, E-VarM achieves relatively good results on IMDB, a dataset with an average of 268 tokens per sentence, demonstrating the outstanding performance of our model in extracting task-specific words.

5.3 The Effect of Different Levels of Information.

We adopt the bottom (1∼4 layer), middle (5∼8 layer), and top (9∼12 layer) layers of information for model performance evaluation to demonstrate the importance of using multi-layer information when selecting task-specific words. As shown in Table 5, the model , relying only on low-layer information (low-layer model), achieves good performance when performing shallow-level tasks such as sentiment analysis. In contrast, when the model uses only middle or high-layer information (middle-layer or high-layer model), its performance decreases slightly compared to the low-layer model since the middle and high layers are not sensitive to superficial information such as the sentiment task's literal meaning. Similarly, for tasks such as syntactic judgment and semantic reasoning, the accuracy of the low-layer model is much lower than that of the middle-layer or high-layer model, indicating that the low layer does not contain the syntactic knowledge on which these complex tasks rely. Since fusing all layers of information, our model achieves the highest accuracy on all datasets. The above experiments indicate that the model needs to simultaneously learn multiple levels of information, including literal, phrasal, syntactic, semantic, and task information, when selecting task-specific words for a classification task.

Table 4: A case study of the interpretability of different models on three datasets. The top three task-specific words are highlighted and the color saturation indicates the word importance.

Datasets	Models	Texts	Prediction
Subj	BERT	thoughtful even stinging at times and lots of fun.	Subjective
	Vmask	thoughtful even stinging at times and lots of fun.	
	E-VarM	thoughtful even stinging at times and lots of fun .	
AGNews	BERT	athens reuters at the beach volleyball the 2004 olympics is a sell out ...	Sports
	Vmask	athens reuters at the beach volleyball the 2004 olympics is a sell out ...	
	E-VarM	athens reuters at the beach volleyball the 2004 olympics is a sell out ...	
RT	BERT	characterisation has been sacrificed for the sake of spectacle.	Negative
	Vmask	characterisation has been sacrificed for the sake of spectacle.	
	E-VarM	characterisation has been sacrificed for the sake of spectacle.	

Table 5: Bottom (1∼4 layer), middle (5∼8 layer) and top (9∼12 layer) level information impact on model performance.

Datasets	Bottom	Middle	Top	E-VarM
QNLI	86.02	86.95	87.52	**87.94**
QQP	89.31	90.15	90.32	**91.08**
COLA	81.36	81.98	82.35	**83.99**
IMDB	92.34	92.05	91.73	**92.73**
AGNews	92.72	91.18	93.69	**94.00**
Yelp	96.98	96.65	96.39	**97.28**
Subj	96.32	96.35	97.01	**97.30**
RT	87.97	87.21	87.10	**88.67**
SST-1	52.58	50.81	50.31	**53.44**
SST-2	91.54	91.01	90.87	**92.37**

5.4 Visualizing the Text Representations

To present the phenomenon of representations overlap caused by IB and demonstrate that our E-VarM can learn better class discriminative representations, we randomly select 1000 test samples for each dataset and feed them to E-VarM$_{\text{MASK}}$ and E-VarM to obtain text representations. We then visualize these text representations using the T-SNE (Van der Maaten and Hinton, 2008) and show the results for the five datasets in Fig 3 (The results of the comparison with Vmask are in **Appendix B**). As observed, the text representations obtained by E-VarM$_{\text{MASK}}$ have different degrees of overlap, in which the inter-class distance is smaller than that of E-VarM, while the intra-class distance is larger than that of E-VarM. Especially, on the COLA, QNLI, and QQP datasets that involve complex tasks and rely on multi-layer semantics for decisions, there is significant overlap of text representations obtained by E-VarM$_{\text{MASK}}$ among different classes, which would result in indistinguishable classes and reduce the prediction accuracy. In contrast, E-VarM alleviates the problem of inter-class overlap and intra-class dispersion on all datasets through supervised constrastive learning and thus obtains better text representations.

5.5 Visualizing the Interpretation

To further compare the interpretability of the BERT, Vmask, and E-VarM, we conduct case studies on three datasets: Subj, AGNews, and RT. We highlight the top three important words selected by LIME, with the level of color saturation indicating the word's importance. As shown in Table 4, for the same sentences, all three models make correct predictions, it is clear that BERT and Vmask extract many nonsense or task-irrelevant words such as 'at', 'of', and 'out'. In contrast, our model captures more task-specific words, such as 'olympics' related to the topic 'sports', 'characterisation' as the subject of 'sacrificed,' which fits better with the semantics of the input, and 'stinging' and 'fun ', which are more of a subjective expression, showing the outstanding interpretability of our model.

6 Conclusion

In this paper, we propose E-VarM to simultaneously boost the model's interpretability and accuracy. E-VarM combines multi-level information for task-specific words selection, which can adjust the decision basis of the model, and improve the model's interpretability. Further, E-VarM adopts contrastive learning for representation optimization to mitigate the risk of representations overlap, enhancing the model's classification performance. Experimental results on ten benchmark datasets demonstrate the effectiveness of E-VarM.

Acknowledgements

We sincerely thank the reviewers for their insightful comments and valuable suggestions. This work is funded by the National Key Research and Development Program of the Ministry of Science and Technology of China (No. 2021YFB1716201). Thanks for the computing infrastructure provided by Beijing Advanced Innovation Center for Big Data and Brain Computing.

References

Alexander A. Alemi, Ian Fischer, Joshua V. Dillon, and Kevin Murphy. 2017. Deep variational information bottleneck. In *5th International Conference on Learning Representations, ICLR 2017*.

Seo-Jin Bang, Pengtao Xie, Heewook Lee, Wei Wu, and Eric P. Xing. 2021. Explaining A black-box by using A deep variational information bottleneck approach. In *Thirty-Fifth AAAI Conference on Artificial Intelligence, AAAI 2021, Thirty-Third Conference on Innovative Applications of Artificial Intelligence, IAAI 2021, The Eleventh Symposium on Educational Advances in Artificial Intelligence, EAAI 2021, Virtual Event, February 2-9, 2021*, pages 11396–11404. AAAI Press.

Jasmijn Bastings, Wilker Aziz, and Ivan Titov. 2019. Interpretable neural predictions with differentiable binary variables. In *Proceedings of the 57th Conference of the Association for Computational Linguistics, ACL 2019, Florence, Italy, July 28- August 2, 2019, Volume 1: Long Papers*, pages 2963–2977. Association for Computational Linguistics.

Jasmijn Bastings and Katja Filippova. 2020. The elephant in the interpretability room: Why use attention as explanation when we have saliency methods? In *Proceedings of the Third BlackboxNLP Workshop on Analyzing and Interpreting Neural Networks for NLP*.

Nicola De Cao, Michael Sejr Schlichtkrull, Wilker Aziz, and Ivan Titov. 2020. How do decisions emerge across layers in neural models? interpretation with differentiable masking. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing, EMNLP 2020, Online, November 16-20, 2020*, pages 3243–3255. Association for Computational Linguistics.

Hanjie Chen and Yangfeng Ji. 2020. Learning variational word masks to improve the interpretability of neural text classifiers. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing, EMNLP 2020, Online, November 16-20, 2020*, pages 4236–4251. Association for Computational Linguistics.

Jianbo Chen, Le Song, Martin Wainwright, and Michael Jordan. 2018. Learning to explain: An information-theoretic perspective on model interpretation. In *International Conference on Machine Learning*, pages 883–892. PMLR.

George Chrysostomou and Nikolaos Aletras. 2021. Improving the faithfulness of attention-based explanations with task-specific information for text classification. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing*.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*.

Gabriel G. Erion, Joseph D. Janizek, Pascal Sturmfels, Scott M. Lundberg, and Su-In Lee. 2019. Learning explainable models using attribution priors. *CoRR*, abs/1906.10670.

Borja Rodríguez Gálvez, Ragnar Thobaben, and Mikael Skoglund. 2020. The convex information bottleneck lagrangian. *Entropy*, 22(1):98.

Robert Geirhos, Patricia Rubisch, Claudio Michaelis, Matthias Bethge, Felix A. Wichmann, and Wieland Brendel. 2019. Imagenet-trained cnns are biased towards texture; increasing shape bias improves accuracy and robustness. In *7th International Conference on Learning Representations, ICLR 2019, New Orleans, LA, USA, May 6-9, 2019*. OpenReview.net.

Ziv Goldfeld and Yury Polyanskiy. 2020. The information bottleneck problem and its applications in machine learning. *IEEE Journal on Selected Areas in Information Theory*, 1(1):19–38.

Beliz Gunel, Jingfei Du, Alexis Conneau, and Veselin Stoyanov. 2021. Supervised contrastive learning for pre-trained language model fine-tuning. In *9th International Conference on Learning Representations, ICLR 2021, Virtual Event, Austria, May 3-7, 2021*. OpenReview.net.

Vikram Gupta, Haoyue Shi, Kevin Gimpel, and Mrinmaya Sachan. 2022. Deep clustering of text representations for supervision-free probing of syntax. In *Association for the Advancement of Artificial Intelligence*.

Kishaloy Halder, Alan Akbik, Josip Krapac, and Roland Vollgraf. 2020. Task-aware representation of sentences for generic text classification. In *Proceedings of the 28th International Conference on Computational Linguistics, COLING 2020, Barcelona, Spain (Online), December 8-13, 2020*, pages 3202–3213. International Committee on Computational Linguistics.

John Hewitt and Christopher D Manning. 2019. A structural probe for finding syntax in word representations. In *Proceedings of the 2019 Conference of the North*

American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers), pages 4129–4138.

Geoffrey E. Hinton, Oriol Vinyals, and Jeffrey Dean. 2015. Distilling the knowledge in a neural network. *CoRR*, abs/1503.02531.

Alon Jacovi and Yoav Goldberg. 2020. Towards faithfully interpretable NLP systems: How should we define and evaluate faithfulness? In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, ACL 2020*.

Eric Jang, Shixiang Gu, and Ben Poole. 2017. Categorical reparameterization with gumbel-softmax. In *5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Conference Track Proceedings*. OpenReview.net.

Ganesh Jawahar, Benoît Sagot, and Djamé Seddah. 2019. What does BERT learn about the structure of language? In *Proceedings of the 57th Conference of the Association for Computational Linguistics, ACL 2019*.

Prannay Khosla, Piotr Teterwak, Chen Wang, Aaron Sarna, Yonglong Tian, Phillip Isola, Aaron Maschinot, Ce Liu, and Dilip Krishnan. 2020. Supervised contrastive learning. In *Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020, December 6-12, 2020, virtual*.

Taeuk Kim, Jihun Choi, Daniel Edmiston, and Sang-goo Lee. 2020. Are pre-trained language models aware of phrases? simple but strong baselines for grammar induction. In *8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, April 26-30, 2020*. OpenReview.net.

Diederik P. Kingma and Jimmy Ba. 2015. Adam: A method for stochastic optimization. In *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*.

Artemy Kolchinsky, Brendan D. Tracey, and Steven Van Kuyk. 2019. Caveats for information bottleneck in deterministic scenarios. In *7th International Conference on Learning Representations, ICLR 2019, New Orleans, LA, USA, May 6-9, 2019*. OpenReview.net.

Ruiqi Li, Xiang Zhao, and Marie-Francine Moens. 2022. A brief overview of universal sentence representation methods: A linguistic view. *ACM Computing Surveys (CSUR)*, 55(3):1–42.

Xianming Li, Zongxi Li, Haoran Xie, and Qing Li. 2021. Merging statistical feature via adaptive gate for improved text classification. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 35, pages 13288–13296.

Bin Liang, Wangda Luo, Xiang Li, Lin Gui, Min Yang, Xiaoqi Yu, and Ruifeng Xu. 2021. Enhancing aspect-based sentiment analysis with supervised contrastive learning. In *CIKM '21: The 30th ACM International Conference on Information and Knowledge Management, Virtual Event, Queensland, Australia, November 1 - 5, 2021*, pages 3242–3247. ACM.

Yi-Shan Lin, Wen-Chuan Lee, and Z. Berkay Celik. 2021. What do you see?: Evaluation of explainable artificial intelligence (XAI) interpretability through neural backdoors. In *KDD '21: The 27th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*.

Xin Lv, Yixin Cao, Lei Hou, Juanzi Li, Zhiyuan Liu, Yichi Zhang, and Zelin Dai. 2021. Is multi-hop reasoning really explainable? towards benchmarking reasoning interpretability. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing, EMNLP 2021*.

Andrew Maas, Raymond E Daly, Peter T Pham, Dan Huang, Andrew Y Ng, and Christopher Potts. 2011. Learning word vectors for sentiment analysis. In *Proceedings of the 49th annual meeting of the association for computational linguistics: Human language technologies*, pages 142–150.

Rabeeh Karimi Mahabadi, Yonatan Belinkov, and James Henderson. 2021. Variational information bottleneck for effective low-resource fine-tuning. In *9th International Conference on Learning Representations, ICLR 2021, Virtual Event, Austria, May 3-7, 2021*. OpenReview.net.

Dong Nguyen. 2018. Comparing automatic and human evaluation of local explanations for text classification. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 1069–1078.

Ziqi Pan, Li Niu, Jianfu Zhang, and Liqing Zhang. 2021. Disentangled information bottleneck. In *Thirty-Fifth AAAI Conference on Artificial Intelligence, AAAI 2021, Thirty-Third Conference on Innovative Applications of Artificial Intelligence, IAAI 2021, The Eleventh Symposium on Educational Advances in Artificial Intelligence, EAAI 2021, Virtual Event, February 2-9, 2021*, pages 9285–9293. AAAI Press.

Bo Pang and Lillian Lee. 2005a. Seeing stars: Exploiting class relationships for sentiment categorization with respect to rating scales. In *Proceedings of the 43rd Annual Meeting of the Association for Computational Linguistics (ACL'05)*, pages 115–124, Ann Arbor, Michigan. Association for Computational Linguistics.

Bo Pang and Lillian Lee. 2005b. Seeing stars: Exploiting class relationships for sentiment categorization with respect to rating scales. *arXiv preprint cs/0506075*.

Bhargavi Paranjape, Mandar Joshi, John Thickstun, Hannaneh Hajishirzi, and Luke Zettlemoyer. 2020. An information bottleneck approach for controlling conciseness in rationale extraction. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing, EMNLP 2020, Online, November 16-20, 2020*, pages 1938–1952. Association for Computational Linguistics.

Georgina Peake and Jun Wang. 2018. Explanation mining: Post hoc interpretability of latent factor models for recommendation systems. In *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining, KDD 2018*.

Gregory Plumb, Maruan Al-Shedivat, Ángel Alexander Cabrera, Adam Perer, Eric P. Xing, and Ameet Talwalkar. 2020. Regularizing black-box models for improved interpretability. In *Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020*.

Nazneen Fatema Rajani, Bryan McCann, Caiming Xiong, and Richard Socher. 2019. Explain yourself! leveraging language models for commonsense reasoning. In *Proceedings of the 57th Conference of the Association for Computational Linguistics, ACL 2019, Florence, Italy, July 28- August 2, 2019, Volume 1: Long Papers*, pages 4932–4942. Association for Computational Linguistics.

Danilo Jimenez Rezende and Shakir Mohamed. 2015. Variational inference with normalizing flows. In *Proceedings of the 32nd International Conference on Machine Learning, ICML 2015, Lille, France, 6-11 July 2015*, volume 37 of *JMLR Workshop and Conference Proceedings*, pages 1530–1538. JMLR.org.

Marco Tulio Ribeiro, Sameer Singh, and Carlos Guestrin. 2016. " why should i trust you?" explaining the predictions of any classifier. In *Proceedings of the 22nd ACM SIGKDD international conference on knowledge discovery and data mining*, pages 1135–1144.

Joshua David Robinson, Ching-Yao Chuang, Suvrit Sra, and Stefanie Jegelka. 2021. Contrastive learning with hard negative samples. In *9th International Conference on Learning Representations, ICLR 2021, Virtual Event, Austria, May 3-7, 2021*. OpenReview.net.

Timo Schick, Helmut Schmid, and Hinrich Schütze. 2020. Automatically identifying words that can serve as labels for few-shot text classification. In *Proceedings of the 28th International Conference on Computational Linguistics, COLING 2020, Barcelona, Spain (Online), December 8-13, 2020*, pages 5569–5578. International Committee on Computational Linguistics.

Richard Socher, Alex Perelygin, Jean Wu, Jason Chuang, Christopher D Manning, Andrew Y Ng, and Christopher Potts. 2013. Recursive deep models for semantic compositionality over a sentiment treebank. In *Proceedings of the 2013 conference on empirical methods in natural language processing*, pages 1631–1642.

Xiaofei Sun, Diyi Yang, Xiaoya Li, Tianwei Zhang, Yuxian Meng, Han Qiu, Guoyin Wang, Eduard H. Hovy, and Jiwei Li. 2021. Interpreting deep learning models in natural language processing: A review. *CoRR*, abs/2110.10470.

Zijun Sun, Chun Fan, Qinghong Han, Xiaofei Sun, Yuxian Meng, Fei Wu, and Jiwei Li. 2020. Self-explaining structures improve nlp models. *arXiv preprint arXiv:2012.01786*.

Kyle Swanson, Lili Yu, and Tao Lei. 2020. Rationalizing text matching: Learning sparse alignments via optimal transport. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, ACL 2020, Online, July 5-10, 2020*, pages 5609–5626. Association for Computational Linguistics.

Toshiyuki Tanaka. 1998. A theory of mean field approximation. In *Advances in Neural Information Processing Systems 11, [NIPS Conference, Denver, Colorado, USA, November 30 - December 5, 1998]*, pages 351–360. The MIT Press.

Naftali Tishby, Fernando C. N. Pereira, and William Bialek. 2000. The information bottleneck method. *CoRR*.

Betty Van Aken, Benjamin Winter, Alexander Löser, and Felix A Gers. 2019. How does bert answer questions? a layer-wise analysis of transformer representations. In *Proceedings of the 28th ACM International Conference on Information and Knowledge Management*, pages 1823–1832.

Betty van Aken, Benjamin Winter, Alexander Löser, and Felix A. Gers. 2019. How does BERT answer questions?: A layer-wise analysis of transformer representations. In *Proceedings of the 28th ACM International Conference on Information and Knowledge Management, CIKM 2019, Beijing, China, November 3-7, 2019*, pages 1823–1832. ACM.

Laurens Van der Maaten and Geoffrey Hinton. 2008. Visualizing data using t-sne. *Journal of machine learning research*, 9(11).

Alex Wang, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel Bowman. 2018. GLUE: A multi-task benchmark and analysis platform for natural language understanding. In *Proceedings of the 2018 EMNLP Workshop BlackboxNLP: Analyzing and Interpreting Neural Networks for NLP*, pages 353–355, Brussels, Belgium. Association for Computational Linguistics.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Rémi Louf, Morgan Funtowicz, Joe Davison, Sam Shleifer, Patrick von Platen, Clara Ma, Yacine Jernite, Julien Plu, Canwen Xu,

Teven Le Scao, Sylvain Gugger, Mariama Drame, Quentin Lhoest, and Alexander M. Rush. 2019. Huggingface's transformers: State-of-the-art natural language processing. *ArXiv*, abs/1910.03771.

Tailin Wu, Ian Fischer, Isaac L Chuang, and Max Tegmark. 2020. Learnability for the information bottleneck. In *Uncertainty in Artificial Intelligence*, pages 1050–1060. PMLR.

Xing Wu, Chaochen Gao, Liangjun Zang, Jizhong Han, Zhongyuan Wang, and Songlin Hu. 2021. Esimcse: Enhanced sample building method for contrastive learning of unsupervised sentence embedding. *CoRR*, abs/2109.04380.

Lanqing Xue, Xiaopeng Li, and Nevin L. Zhang. 2020. Not all attention is needed: Gated attention network for sequence data. In *The Thirty-Fourth AAAI Conference on Artificial Intelligence, AAAI 2020, The Thirty-Second Innovative Applications of Artificial Intelligence Conference, IAAI 2020, The Tenth AAAI Symposium on Educational Advances in Artificial Intelligence, EAAI 2020, New York, NY, USA, February 7-12, 2020*, pages 6550–6557. AAAI Press.

Dejiao Zhang, Shang-Wen Li, Wei Xiao, Henghui Zhu, Ramesh Nallapati, Andrew O. Arnold, and Bing Xiang. 2021. Pairwise supervised contrastive learning of sentence representations. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing, EMNLP 2021, Virtual Event / Punta Cana, Dominican Republic, 7-11 November, 2021*, pages 5786–5798. Association for Computational Linguistics.

Xiang Zhang, Junbo Zhao, and Yann LeCun. 2015. Character-level convolutional networks for text classification. *Advances in neural information processing systems*, 28.

A Datasets

This paper uses ten widely studied datasets that cover a wide range of application domains, which are described in detail below:

For sentiment analysis, we use five benchmark datasets in which **RT** [3] (Pang and Lee, 2005a), **IMDB** [4] (Maas et al., 2011), **SST-2** [5] (Socher et al., 2013) and **Yelp (YelpReviewPolarity)** [6] (Zhang et al., 2015) are four binary sentiment polarity datasets with each sentence annotated as positive or negative. And **SST-1** [7] (Socher et al., 2013) is a fine-grained sentiment dataset derived from Stanford Sentiment Treebank with five balanced labels (negative, somewhat negative, neutral, somewhat positive, positive).

For topic categorization, we use **AG's News**[8] (Zhang et al., 2015) dataset where each article only has a title and description and can be categorized into one of the four main classes: "World", "Sports", "Business", and "Technology".

For grammatical judgment, we adopt **COLA** [9] (Wang et al., 2018) dataset that published by New York University with each sentence marked whether there are grammatical errors or not.

For semantic inference, we employe **QQP** [10] (Wang et al., 2018) dataset to determine whether a questions pair is semantically equivalent, and the **QNLI** [11] (Wang et al., 2018) dataset to judge a question-sentence pair is entailment relation or not.

Additionally, we leverage **Subj** [12] (Pang and Lee, 2005b) dataset to carry out subjective / objective classification, which contains 5000 subjective and 5000 objective sentences, respectively.

B Visualization Supplement

We visualize the text representations for the remaining five datasets . As shown in Fig. 4, the text representations obtained by E-VarM have a larger inter-class distance and a smaller intra-class distance than that obtained by Vmask. This phenomenon indicates that our model can adjust the text representation and alleviate the problem of representations overlap.

[3] https://www.cs.cornell.edu/people/pabo/movie-review-data/rt-polaritydata.tar.gz

[4] http://ai.stanford.edu/ amaas/data/sentiment/aclImdb_v1.tar.gz

[5] https://drive.google.com/uc?export=download&id=0Bz8a_Dbh9QhbNUpYQ2N3SGlFaDg

[6] https://drive.google.com/uc?export=download&id=0Bz8a_Dbh9QhbNUpYQ2N3SGlFaDg

[7] https://drive.google.com/uc?export=download&id=0Bz8a_Dbh9QhbNUpYQ2N3SGlFaDg

[8] http://groups.di.unipi.it/ gulli/AG_corpus_of_news_articles.html

[9] https://gluebenchmark.com/tasks

[10] https://gluebenchmark.com/tasks

[11] https://gluebenchmark.com/tasks

[12] https://www.cs.cornell.edu/people/pabo/movie-review-data/

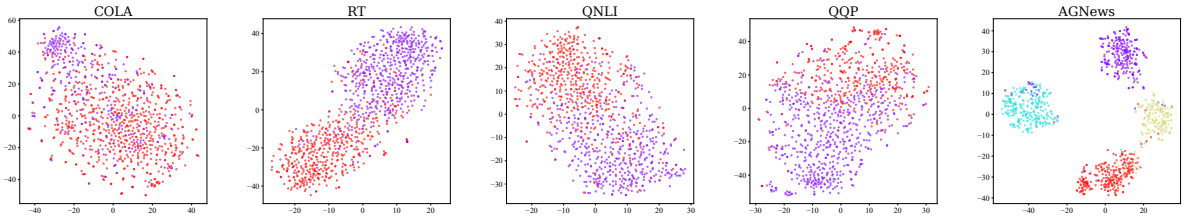

(a) Vmask

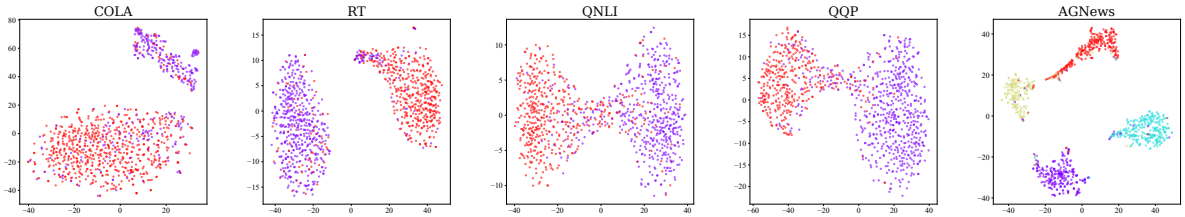

(b) E-VarM

Figure 4: T-SNE visualization comparison. The upper is Vmask, and the lower is our method.

Attribute Injection for Pretrained Language Models: A New Benchmark and An Efficient Method

Reinald Kim Amplayo*
University of Edinburgh
reinald.kim@ed.ac.uk

Kang Min Yoo **Sang-Woo Lee**
NAVER AI Lab, NAVER Clova AI
{kangmin.yoo,sang.woo.lee}@navercorp.com

Abstract

Metadata attributes (e.g., user and product IDs from reviews) can be incorporated as additional inputs to neural-based NLP models, by expanding the architecture of the models to improve performance. However, recent models rely on pretrained language models (PLMs), in which previously used techniques for attribute injection are either nontrivial or cost-ineffective. In this paper, we introduce a benchmark for evaluating attribute injection models, which comprises eight datasets across a diverse range of tasks and domains and six synthetically sparsified ones. We also propose a lightweight and memory-efficient method to inject attributes into PLMs. We extend *adapters*, i.e. tiny plug-in feed-forward modules, to include attributes both independently of or jointly with the text. We use approximation techniques to parameterize the model efficiently for domains with large attribute vocabularies, and training mechanisms to handle multi-labeled and sparse attributes. Extensive experiments and analyses show that our method outperforms previous attribute injection methods and achieves state-of-the-art performance on all datasets.

1 Introduction

Neural-based NLP models are powered by large-scale textual datasets, which are mostly crawled from the web (Denoyer and Gallinari, 2006; Sandhaus, 2008; Zhu et al., 2015; Ni et al., 2019; Raffel et al., 2020). Web texts usually are attached with *metadata*, i.e. attributes that describe the texts. For example, product reviews have user and product IDs, as well as their ratings, while research papers on arXiv have author lists and research areas as metadata attributes (see Figure 1). While most of the recent models disregard them and focus more on ungrounded language understanding (understanding language on its own, e.g., GLUE; Wang et al., 2018, *inter alia*), prior work has shown that incorporating these attributes into our model increases not just its performance but also its interpretability and customizability (Tang et al., 2015; Chen et al., 2016; Kim et al., 2019). This work explores the task of *attribute injection* (Amplayo, 2019), which aims to use attributes to improve the performance of NLP models effectively.

Yelp Review
Text: My boyfriend's fav. place and the stein of beers are priced pretty good. Game nights get super packed so go early to save a seat. Kitchen closes at midnight which is too early when your buzz kicks in around 1am.
Attributes:
– User: `n6LeAoIuDR3NfIBEsmL_zg`
– Product: `7TMf1NuuAdvhG7IojZSKnw`
Paper Abstract
Text: We present new and improved fixed-parameter algorithms for computing maximum agreement forests (MAFs) of pairs of rooted binary phylogenetic trees. The size of such a forest for two trees corresponds to their subtree prune-and-regraft distance and, if the agreement forest is acyclic, to their hybridization number ...
Attributes:
– Authors: `Chris Whidden`, `Robert G. Beiko`, `Norbert Zeh`
– Research Areas: `q-bio.PE`, `cs.DS`

Figure 1: Examples of a Yelp review and an arXiv paper abstract and their corresponding attributes. Texts in `typewriter` font are attribute labels.

Previous methods for attribute injection (Tang et al., 2015; Zhu et al., 2015; Chen et al., 2016; Ma et al., 2017; Dou, 2017; Amplayo et al., 2018; Wu et al., 2018; Long et al., 2018; Kim et al., 2019; Amplayo, 2019) involve two steps: (1) designing an architecture that accepts both texts and attributes, and (2) training the model from scratch using task-specific datasets. However, these methods of modifying different modules of the model can be non-trivial when applied to pretrained language models (PLMs; Devlin et al., 2019; Liu et al., 2019; Qiu et al., 2020). The use of PLMs disallows designing new and specialized architectures. Zhang et al.

* Work done while Reinald was at NAVER AI Lab. He is now at Google Research.

Proceedings of the 29th International Conference on Computational Linguistics, pages 1051–1064
October 12–17, 2022.

(2021b) append large and deep layers of attribute-specific Transformers to the end of PLMs, which cannot leverage the potential of attributes in the intermediate layers of PLMs and also scales poorly to multiple tasks and domains. Finally, more recent work on language model customization and controllability makes use of textual prompts (Brown et al., 2020; Schick and Schütze, 2021), specialized tokens (Fan et al., 2018; Keskar et al., 2019), and additional neural modules (Perez et al., 2018; Wang et al., 2019; Lauscher et al., 2020; Liu et al., 2021) to introduce additional contexts, such as style, topic, and end task. Unfortunately, these techniques do not generalize to all kinds of attributes, such as those that are non-textual (e.g., user IDs that are not text-translatable), multi-labeled (e.g., multiple authors of a paper), and with large vocabularies (e.g., thousands of products available).

Our contribution in this paper is two-fold. Firstly, we introduce a new benchmark to evaluate models for attribute injection. The benchmark consists of eight datasets which include three newly collated ones from different tasks and domains, and six synthetically sparsified datasets which are specifically created to evaluate models in sparse settings. These datasets span from diverse tasks, such as sentiment classification, spoiler detection, and message type classification, and contain attributes that have different properties (sparse vs. non-sparse, single-labeled vs. multi-labeled, etc.). Experiments on the benchmark show that our method outperforms previous approaches, as well as competitive baselines that fully fine-tune the pretrained language model. We also conduct extensive analyses to show that our method is robust to sparse and cold-start attributes and that it is modular with attribute-specific modules transferrable to other tasks using the same attributes.

Secondly, we propose a lightweight and memory-efficient method that is specifically suited to inject attributes into PLMs, which can be non-textual, multi-labeled, and have large vocabularies. We make use of adapters (Houlsby et al., 2019), i.e. feed-forward modules inserted between layers of PLMs that are tiny in size, and extend them such that attributes are injected as additional inputs to the model. We introduce two kinds of injection methods, which either incorporate attributes independently of or jointly with the text representation. A naive implementation of the latter would substantially increase the parameters, especially when the attribute vocabulary is large, thus we use ideas from low-rank matrix approximations as well as parameterized hypercomplex multiplications (Zhang et al., 2021a; Mahabadi et al., 2021) to significantly decrease the fine-tuned parameters by up to 192× for a default base-sized BERT (Devlin et al., 2019) setting. We also use two mechanisms, attribute dropout and post-aggregation, to handle attribute sparsity and multi-labeled attributes, respectively. Our use of adapters enables us to parameter-efficiently train our model, i.e. by freezing pretrained weights and only updating new parameters at training time. We make our code and datasets publicly available.[1]

2 Related Work

Prior to the neural network and deep learning era, traditional methods for NLP have relied on feature sets as input to machine learning models. These feature sets include metadata attributes such as author lists and publication venue of research papers (Rosen-Zvi et al., 2004; Joorabchi and Mahdi, 2011; Kim et al., 2017), topics of sentences (Ramage et al., 2009; Liu and Forss, 2014; Zhao and Mao, 2017), as well as spatial (Yang et al., 2017) and temporal (Fukuhara et al., 2007) metadata attributes found in tweets. Attributes are mostly used in the area of sentiment classification (Gao et al., 2013), where most of the time textual data includes freely available user and product attributes. These methods rely on manually curated features that would represent the semantics of user and product information.

Deep neural networks gave rise to better representation learning (Bengio et al., 2013; Mikolov et al., 2013), which allows us to learn from scratch semantic representation of attributes in the form of dense vectors (Tang et al., 2015). The design of how to represent attributes has evolved from using attribute-specific word and document embeddings (Tang et al., 2015) and attention pooling weights (Chen et al., 2016; Ma et al., 2017; Amplayo et al., 2018; Wu et al., 2018), to more complicated architectures such as memory networks (Dou, 2017; Long et al., 2018) and importance matrices (Amplayo, 2019). These designs are model- and domain-dependent and can be non-trivial to apply to other models and datasets. Our proposed method, on the other hand, works well on any pretrained language model which are mostly based on Trans-

[1] https://github.com/rktamplayo/Injector

former (Vaswani et al., 2017; Devlin et al., 2019). Zhang et al. (2021b) used six layers of attribute-injected Transformers where attributes are used as input to the self-attention module, which is costly in terms of memory, i.e., equivalent to adding 50% of the original BERT-base parameters. Attributes are useful in the intermediate layers when learning the semantics of the input text, which this model cannot leverage.

Our work is closely related to recent literature on controlled text generation, where most of the work use either specialized control tokens concatenated with the input text (Sennrich et al., 2016; Kikuchi et al., 2016; Ficler and Goldberg, 2017; Fan et al., 2018; Keskar et al., 2019), or textual prompts that instruct the model on what to generate (Brown et al., 2020; Schick and Schütze, 2021; Gao et al., 2021; Zhao et al., 2021). While these methods have been successfully applied to pretrained language models, the attributes used to control the text are limited to those that are text-translatable (e.g., topics such as "technology") and those with limited vocabulary (e.g., "positive" or "negative" sentiment).

Methods to efficiently fine-tune pretrained models have been explored since their inception (Dathathri et al., 2019; Li and Liang, 2021; He et al., 2022). One of the more popular methods is the use of adapters (Houlsby et al., 2019), where tiny trainable feed-forward modules are inserted and pretrained weights are frozen during training. Most of the prior work on adapters focuses on either improving their effectiveness and efficiency (Mao et al., 2021; Mahabadi et al., 2021; He et al., 2022) or applying it to domain adaptation and transfer learning tasks (Pfeiffer et al., 2021; He et al., 2021; Cooper Stickland et al., 2021). Our work extends the use of adapters to additionally accept attributes as input, which is closely related to Perez et al. (2018) and Lauscher et al. (2020), where the adapter accepts textual questions and commonsense knowledge as input, respectively. However, unlike previous work where the additional input can be transformed into natural language tokens, attributes are usually not the case (e.g., see user and product IDs in Figure 1), which poses new challenges in efficiency when applied to adapters.

3 Attribute Injection Benchmark

Problem Setting Let $x = \{x_i\}_{i=1}^{N}$ denote the input text of N tokens, y is a task-specific output, and $p(y|x)$ is a discriminative model that predicts y given x. Suppose there exists a set of non-textual and categorical attributes $z = \{z_j\}_{j=1}^{M}$ that describe text x (e.g., user and product IDs of product reviews). These attributes can be multi-labeled, i.e. $z_j = [z_j^{(k)}]$ (e.g., multiple authors of a research paper) and use a finite yet possibly large vocabulary $\mathcal{Z}_j$, i.e. $z_j \subseteq \mathcal{Z}_j$. The task of attribute injection aims to build a model $q(y|x, z)$ that additionally incorporates z as input such that the gain in task performance between p and q is maximized. In our setting, p is a pretrained language model (PLM) fine-tuned to the task, while q is a PLM that also takes z as additional input.

The Datasets To evaluate attribute injection models on a wide variety of tasks and datasets from different domains, we introduce ATTRIB, a collection of benchmark datasets to evaluate attribute injection methods. ATTRIB consists of a total of 14 datasets. The first eight datasets are collected from different domains with a diverse set of tasks spanning from sentiment classification to spoiler detection. Three of the eight datasets are newly introduced in this paper. Table 1 reports the dataset statistics, which shows the different characteristics (i.e., size of training data, input text length, number of attributes, sparsity, and multi-labelity of attributes) of each dataset.

The first three datasets (**Y2013**, **Y2014**, and **IMDB**; Tang et al., 2015) are sentiment classification datasets with user and product attributes from two different sources (Yelp and IMDB). **AAPR** (Yang et al., 2018) is a dataset for classifying whether an arXiv paper is accepted to a conference or not, with two attributes, lists of authors and research areas. **POLMED** (Kim et al., 2019) is a classification dataset in the political domain, where the goal is to classify the message type of a tweet, with four attributes, politician, media source, audience, and political bias.

We also introduce three new attribute injection datasets (see Appendix for details on dataset collection). **FOOD** is a dataset in which given a recipe from Food.com and three attributes, user and lists of ingredients and tags, we are tasked to predict the estimated number of minutes it takes to make the food, rounded down to the tens. **GOOD** is a spoiler detection dataset where we classify whether a book review from Goodreads contains a spoiler or not, with three attributes: user, book, and rating. And finally, **BEER** is a multi-aspect rating predic-

Dataset	Y2013	Y2014	IMDB	AAPR	POLMED	FOOD	GOOD	BEER
#Train	62.5K	183.0K	67.4K	33.5K	4.5K	162.4K	714.7K	1.5M
#Dev	7.8K	22.7K	8.4K	2.0K	–	20.3K	10.0K	10.0K
#Test	8.7K	25.4K	9.1K	2.0K	0.5K	20.3K	10.0K	10.0K
#Words/Input	210(166)	218(175)	425(278)	97(36)	38(62)	101(65)	132(72)	133(56)
#Classes	5	5	10	2	9	16	2	4×9
#Attributes	2	2	2	2	4	3	3	3
#Attr. Vocab	3.3K	9.0K	2.9K	51.6K	0.5K	40.5K	43.8K	98.0K
%Sparse	0.0%	0.0%	0.0%	97.8%	63.8%	80.0%	34.4%	75.3%
Multi-label?				✓		✓		

Table 1: ATTRIB datasets statistics. The second block reports new datasets introduced in this paper. BEER is a multi-task dataset, with nine classes for each of four given aspects. %Sparse is the percentage of attributes with less than 10 training examples. Multi-label attributes include lists of authors and research areas for AAPR, and lists of ingredients and tags for FOOD. The kinds of attributes available and further analyses can be found in the Appendix.

tion dataset in which given a review and three attributes, user, beer, and overall rating, we are tasked to predict the ratings of four *aspects*, i.e. properties that influence user satisfaction: appearance, aroma, palate, and taste.

Additionally, we collated six synthetically sparsified datasets that were created specifically to evaluate attribute injection methods in cold-start environments[2]. Specifically, Amplayo et al. (2018) downsampled the Y2013 and IMDB training datasets such that the attributes are sparser than the original. There are three levels of sparsity: 20%, 50%, and 80%, where $x\%$ means that $x\%$ of attributes are cold-start and thus are unseen in the training examples. Statistics of these datasets are shown in the Appendix.

4 Modeling Approach

Our proposed method, which we call INJECTORS, can be summarized as follows. We extend adapters (Houlsby et al., 2019), which are tiny feed-forward neural networks plugged into pretrained language models, such that they also accept attributes z as input. Attributes z can be represented as additional bias parameters or as perturbations to the weight matrix parameter of the adapter, motivated by how attributes are used to classify texts. We decrease the number of parameters exponentially using low-rank matrix approximations and parameterized hypercomplex multiplications (Zhang et al., 2021a). Finally, we introduce two training mechanisms, attribute dropout and post-aggregation, to mitigate problems regarding attribute sparsity and multi-label properties. Figure 2 illustrates an overview of INJECTORS.

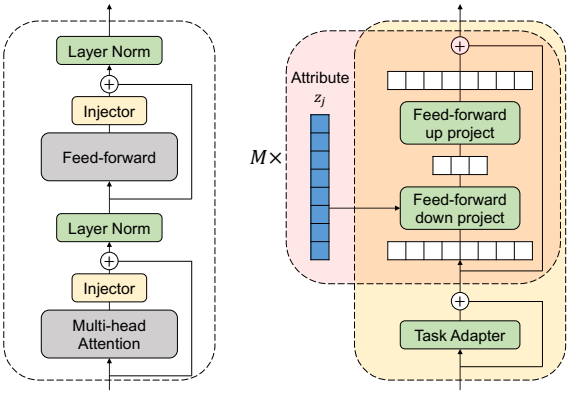

Figure 2: Architecture of the INJECTOR module when integrated into one block of a Transformer model (see left of figure). INJECTOR starts with a task-specific adapter, followed by M attribute-specific adapters, one for each attribute given in the task (see right of figure). Green-colored modules are trained and fine-tuned, while gray-colored modules are fixed.

Preliminary: Adapters We first briefly describe adapters. Let $\mathbf{h} \in \mathbb{R}^{d_h}$ be the output hidden vector from a multi-head attention or feed-forward layer in a Transformer block. An adapter layer is basically two feed-forward networks that projects $\mathbf{h}$ into vector $\mathbf{h}' \in \mathbb{R}^{d_h}$ with a much smaller dimension $d_a \ll d_h$:

$$\begin{aligned}\mathbf{h}' &= \text{Adapt}(\mathbf{h}) \\ &= \text{FFNet}_{up}(f(\text{FFNet}_{down}(\mathbf{h}))) + \mathbf{h} \end{aligned} \quad (1)$$

where $\text{FFNet}(\mathbf{x}) = \mathbf{W}\mathbf{x} + \mathbf{b}$, $\mathbf{W}$ and $\mathbf{b}$ are learned weight and bias parameters of FFNet, $f(\cdot)$ is a non-linear function, and the addition represents a residual layer.

[2]In this paper, we distinguish the definition of *sparsity* and *cold-start* attributes as the former referring to attributes with *few* training examples, and the latter referring to those with *zero* training examples.

Task-specific Adapter INJECTORS start with a task-specific adapter that uses Equation 1 to transform the previous hidden vector $\mathbf{h}$ to $\mathbf{h}'$. The use of a separate task-specific adapter is essential to make our method modularizable and learned attributes on one task transferrable to another. We show extensive analyses of the modularity of our method in the later sections.

Attribute-specific Adapters Attributes z are injected through attribute-specific adapters, where they are used in two different ways. Firstly, they are used as *bias* parameters independent of the text representation. This is motivated by the fact that attributes can have a prior disposition regardless of what is written in the text. For example, a user may tend to give lower review ratings than average. Secondly, they are also used as *weight* parameters. This allows our method to jointly model attributes with the text representation. This is motivated by how attributes can change the semantics of the text. For example, one user may like very sweet food while another user may dislike it, thus the use of the word *sweet* in the text may mean differently to them.

More formally, for each attribute $z_j \in z$, we obtain its embedding $\mathbf{z}_j$ from a learned embedding matrix and sequentially transform the previously attribute-injected vector $\mathbf{h}_{z_{j-1}}$ to attribute-injected vector $\mathbf{h}_{z_j}$ using the following equation:

$$\begin{aligned}\mathbf{h}_{z_j} &= \text{AttrAdapt}(\mathbf{h}_{z_{j-1}}, \mathbf{z}_j) && (2)\\ &= \text{FFNet}_{up}(f(\mathbf{W}_{z_j}\mathbf{h}_{z_{j-1}} + \mathbf{b}_{z_j})) \\ &\quad + \mathbf{h}_{z_{j-1}} && (3)\end{aligned}$$

where $\mathbf{h}_{z_0} = \mathbf{h}'$ from the output of task-specific adapter. Unlike standard adapters, the attribute-specific weight matrix $\mathbf{W}_{z_j} \in \mathbb{R}^{d_h \times d_a}$ and bias parameter $\mathbf{b}_{z_j} \in \mathbb{R}^{d_a}$ of the down-project feed-forward network are not learned from scratch, but instead are calculated as follows.

The calculation of the bias parameter $\mathbf{b}_{z_j}$ is trivial; we perform a linear transformation of the attribute embedding $\mathbf{z}_j$:

$$\mathbf{b}_{z_j} = g_{bias}(\mathbf{z}_j) + \mathbf{c}_{bias} \quad (4)$$

where $g_{bias} \in \mathbb{R}^{d_z} \mapsto \mathbb{R}^{d_a}$ is a linear projection, $\mathbf{c}_{bias} \in \mathbb{R}^{d_a}$ is a learned vector, and d_z is the attribute embedding size.

We also define $\mathbf{W}_{z_j}$ as:

$$\mathbf{W}_{z_j} = g_{weight}(\mathbf{z}_j) + \mathbf{C}_{weight} \quad (5)$$

where $\mathbf{C}_{weight} \in \mathbb{R}^{d_h \times d_a}$ is a learned matrix. The function g_{weight}, however, cannot be defined similarly as a linear projection. This would require a tensor parameter of size $d_z \times d_h \times d_a$ to linearly project $\mathbf{z}_j$ to $\mathbf{W}_j$. Considering the fact that we may have multiple attributes for each domain, the number of parameters would not scale well and makes the model very large and difficult to train. Inspired by Mahabadi et al. (2021), we use ideas from low-rank matrix decomposition and parameterized hypercomplex multiplications (PHMs; Zhang et al., 2021a) to substantially decrease the number of parameters. Figure 3 shows an illustrative overview.

More specifically, we first transform attribute embedding $\mathbf{z}_j$ into vectors in hypercomplex space with O dimensions, i.e.:

$$\hat{\mathbf{z}}_j = [\sigma_1(\mathbf{z}_j), ..., \sigma_O(\mathbf{z}_j)] \in \mathbb{H}^{d_z} \quad (6)$$

where $\sigma_o(\cdot) \in \mathbb{R}^{d_z} \mapsto \mathbb{R}^{d_a}$ is a linear projection in the oth dimension. A hypercomplex vector with O dimensions is basically a set of vectors with one real vector and $O-1$ "imaginary" vectors.[3]

For each dimension o, we first define a small rank-one matrix $\mathbf{S}_o \in \mathbb{R}^{d_a \times d_h/O^2}$ as an outer product between $\hat{\mathbf{z}}_{j,o}$ and a learned vector $\mathbf{s}_o \in \mathbb{R}^{d_h/O^2}$:

$$\mathbf{S}_o = \hat{\mathbf{z}}_{j,o}\mathbf{s}_o^\top \quad (7)$$

and then define a large matrix $\hat{\mathbf{W}}_{j,o} \in \mathbb{R}^{d_h \times d_a}$ as the Kronecker product, denoted by $\otimes$ between two matrices $\mathbf{S}_o$ and a learned matrix $\mathbf{A}_o \in \mathbb{R}^{O \times O}$, followed by a reshape and the hyperbolic tangent function:

$$\hat{\mathbf{W}}_o = \text{Reshape}(\tanh(\mathbf{S}_o \otimes \mathbf{A}_o)) \quad (8)$$

Finally, we add the large matrices $\hat{\mathbf{W}}_o$ of each dimension. To sum up, we define $g_{weight}(\mathbf{z}_j)$ as:

$$g_{weight}(\mathbf{z}_j) = \sum_{o=0}^{O} \text{Reshape}(\tanh(\sigma_o(\mathbf{z}_j)\mathbf{s}_o^\top \otimes \mathbf{A}_o)) \quad (9)$$

Low-rank (Eq. 7) and PHMs (Eqs. 8-9) are both necessary to achieve a high performance with decreased parameters. While the low-rank method in itself reduces the most parameters, it also reduces the expressive power of the model since it outputs rank-one matrices. PHMs mitigate this by performing a sum of Kronecker products, increasing the

[3] Following Tay et al. (2019) and Zhang et al. (2021a), we remove the imaginary units of these vectors to easily perform operations on them, thus these vectors are also in the real space.

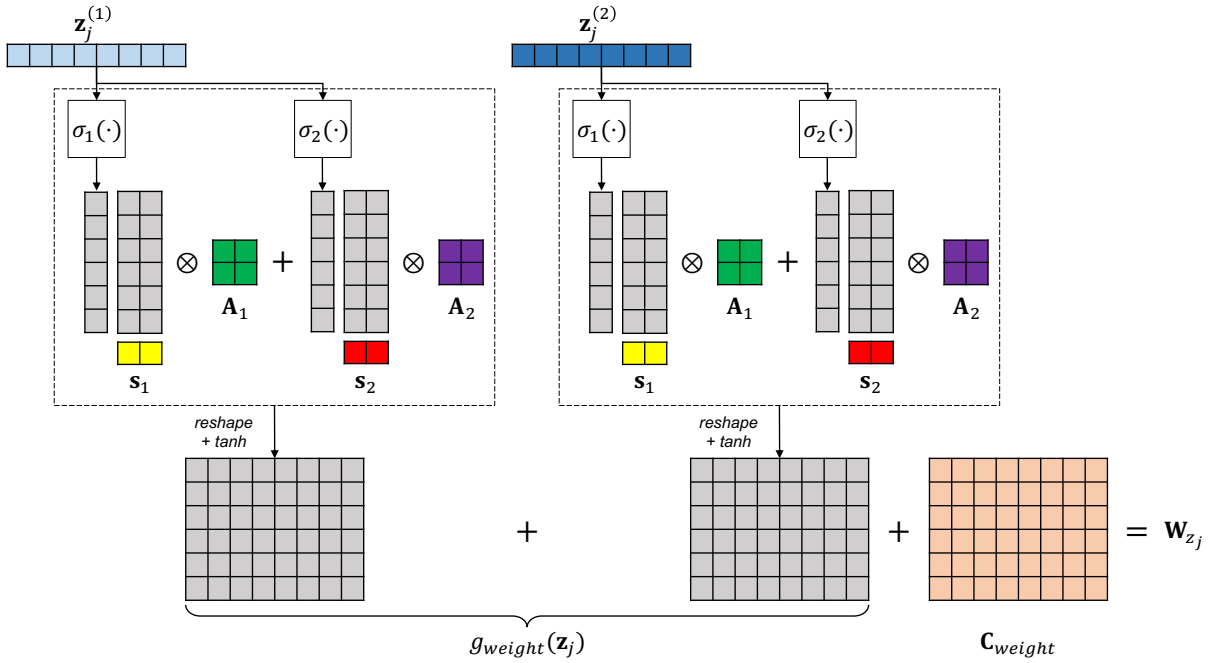

Figure 3: An illustration of how attribute embedding $\mathbf{z}_j$ is transformed into weight matrix $\mathbf{W}_{z_j}$. The colored tensors are learned parameters, while the gray ones are derived. By using a set of tiny parameters $\mathbf{A}_o$ and $\mathbf{s}_o$, we are able to obtain large matrices. When there are multiple labels for attribute z_j, we process them separately and aggregate the resulting large matrices.

rank of the matrix to potentially at most O^2. Finally, this process effectively reduces the number of parameters from $\mathcal{O}(d_z * d_h * d_a)$ to $\mathcal{O}(d_z * d_a)$, since the parameters in σ_o dominate the other parameters (see Appendix for a detailed parameter analysis).

Attribute Dropout and Post-Aggregation For cases where attributes are sparse and multi-labeled, we use the following mechanisms. Firstly, we add a dropout mechanism that randomly masks out attributes from training instances with a rate r_{drop}. This replicates how instances at test time would look, where some attributes are not found in the vocabulary.

Secondly, when there are more than one labels of an attribute, instead of aggregating them first before processing, as in Kim et al. (2019), we perform aggregation post hoc for $\mathbf{W}_{z_j}$), i.e.:

$$\mathbf{b}_{z_j} = \sum\nolimits_k g_{bias}(\mathbf{z}_j^{(k)}) + \mathbf{c}_{bias} \quad (10)$$

$$\mathbf{W}_{z_j} = \sum\nolimits_k g_{weight}(\mathbf{z}_j^{(k)}) + \mathbf{C}_{weight} \quad (11)$$

Aggregating attribute embeddings reduces their individual representation power, while our post-aggregation mechanism preserves this since a sum of non-linear transformations is injective (Xu et al., 2019).

5 Experimental Setup

Training Configuration For the PLMs, we used weights and settings of `bert-base-uncased` (Devlin et al., 2019), available in the HuggingFace library (Wolf et al., 2020). We set the dimensions of all parameters as follows: $d_z = d_h = 768$, $d_a = 64$, and $O = 4$. Using this setting and our parameter-saving method, we are able to decrease the parameters by $192\times$ the naive method. To handle long input texts and fit them into the 512 token limit of PLMs, we truncate them by concatenating the first and last 250 tokens, following Zhang et al. (2021b). We set both the general and attribute dropout rates to 0.2 and the batch size to 8. We used Adam with weight decay (Loshchilov and Hutter, 2019) to optimize our models with a learning rate of $3e-5$ and 200K training steps, with the first 20K steps used to warm-up training linearly.

To train our models, we added a logistic classifier that transforms the `[CLS]` token into logits. The weights here are updated during training. We then used a cross-entropy loss to train the models on all datasets except for Goodreads and Beeradvocate. The Goodreads dataset is very imbalanced towards the negative class (i.e., not a spoiler). We thus put more importance on detecting the spoiler class and used a weighted cross-entropy loss with

0.5 weight on the negative class and 1.0 weight on the positive class. For Beeradvocate, we treat the task as a multi-task problem, where each aspect rating prediction is a separate task. Thus, we used multiple classifiers, one for each aspect, and aggregate the losses from all classifiers by averaging. For PolMed where there is no available development set, we performed a 10-fold cross-validation, following Kim et al. (2019).

Comparison Systems We compared our method with several approaches, including the following no-attribute baselines: (1) **BERT-base ($\mathcal{B}$)**: The base model used in our experiments; and (2) $\mathcal{B}$ **+ ADAPTERS**: Extra tiny parameters are added to the base model and are used for training instead of the full model.

Baselines with attributes injected include the following, where we use the same base model $\mathcal{B}$ for all baselines for ease of comparison: (3) $\mathcal{B}$ **+ TOKENS**: the attributes are used as special control tokens prepended in front of the input. The `[CLS]` token is then passed to the logistic classifier; (4) $\mathcal{B}$ **+ UPA** (Chen et al., 2016): attributes are used as additional bias vectors when calculating the weights of the attention pooling module; (5) $\mathcal{B}$ **+ CHIM** (Amplayo, 2019): attributes are used as importance matrices multiplied to the weight matrix of the logistic classifier; and (6) $\mathcal{B}$ **+ MAA** and $\mathcal{B}$ **+ TINYMAA** (Zhang et al., 2021b): attributes are used to modify the parameters used for calculating query/key/value in the self-attention. To match the parameters of other baselines, we implemented a tiny version with a single Transformer layer and smaller dimensions; and (7-8) $\mathcal{B}$. All models were trained on a single GeForce GTX 1080Ti GPU, except for $\mathcal{B}$ + MAA, in which four GPUs were required to train on the same setting for all datasets.

Finally, we also included in our comparisons a version of our model using the **RoBERTa-base** ($\mathcal{R}$; Liu et al., 2019) configuration ($\mathcal{R}$ **+ INJECTORS**).

6 Results

We evaluated system outputs with accuracy for all datasets except GOOD, where we used F1-score. For brevity in BEER, we took the average of the accuracy of all sub-tasks. Our results are summarized in Table 2. Token-based injection performs worse than the base model on four datasets, which shows that the method is not effective for attribute injection. Overall, among the attribute injected systems, INJECTORS outperforms the other baselines on all datasets, even when BERT parameters are frozen[4]. The difference in performance is especially significant in GOOD, where INJECTORS performs 9.68 points better than the second model. Finally, We also see an increase in performance when applying INJECTORS to RoBERTa, showing that our method can be applied effectively to better-pretrained models. We also conducted ablation studies, detailed in the Appendix, showing the contributions of the different components in the proposed model.

Ablation Studies We present in Table 3 various ablation studies, which assess the contribution of different model components. Our experiments confirm that the use of both bias and weight injection as well as the addition of task adapter improve performance. Interestingly, some datasets prefer one injection type over the other. GOOD dataset, for example, prefers bias injection, i.e., using attributes as prior and independent of the text (e.g., the tendency of the user to write spoilers). Moreover, our training mechanisms also increase the performance of the model. This is especially true for post-aggregation on the FOOD dataset since two of its attributes are multi-labeled (ingredients and tags). Finally, we show that the model variant without one of the parameter-saving methods either performs worse or does not run at all.

On Cold-Start Attributes We checked the performance of the models when trained with synthetic *cold-start attributes* using the Y2013-COLD and IMDB-COLD datasets. We compared the performance of BERT-base, CHIM, TINYMAA, MAA, and INJECTORS. Table 4 shows their performance along with HCSC (Amplayo et al., 2018), which is a hybrid of BiLSTM and CNN with a UPA-style attribute injection method (Chen et al., 2016) and is extended to perform well on cold-start scenarios. As can be seen, our method still performs the best on these datasets, while TINYMAA is consistently worse than the base model when attributes are 80% sparse. MAA improves over TINYMAA with the cost of significant increase in parameters. All BERT-based methods perform better than HCSC on both datasets. Overall, the improvements are the smallest when the dataset is the sparsest (80% sparsified data). This is expected since datasets are smaller and include cold-start attributes.

[4] We did not see an improvement over our final model when fine-tuning *all* parameters of $\mathcal{B}$ + INJECTORS.

Model	Y2013	Y2014	IMDB	AAPR	POLMED	FOOD	GOOD	BEER	TP
BERT-base ($\mathcal{B}$)	67.97	68.07	52.45	63.70	41.82	41.89	41.98	50.48	1.00×
$\mathcal{B}$ + ADAPTERS	66.47	67.44	52.69	62.85	44.24	42.02	48.92	50.71	0.02×
$\mathcal{B}$ + TOKENS	*67.87*	*67.98*	52.68	64.85	42.63	*41.23*	44.79	*50.25*	1.00×
$\mathcal{B}$ + UPA	68.38	68.82	55.76	64.40	42.83	43.97	43.96	51.98	1.01×
$\mathcal{B}$ + CHIM	68.71	68.56	54.31	65.30	43.64	43.35	43.58	52.29	1.01×
$\mathcal{B}$ + TINYMAA	68.03	68.57	55.76	65.00	44.65	43.63	44.65	54.58	1.01×
$\mathcal{B}$ + MAA (our impl.)	70.15	70.51	56.98	65.15	43.43	43.33	48.10	55.04	1.50×
$\mathcal{B}$ + MAA (original)	70.3	71.4	57.3	–	–	–	–	–	1.50×
$\mathcal{B}$ + INJECTORS	**70.86**	**71.69***	**58.90***	**67.10***	**47.27***	**45.01**	**57.78***	**57.87***	0.10×
RoBERTa-base ($\mathcal{R}$)	68.01	69.11	53.31	61.50	42.42	41.82	44.12	52.86	1.12×
$\mathcal{R}$ + INJECTORS	73.00	73.50	60.32	67.30	46.26	44.19	60.73	58.20	0.10×

Table 2: Performance (F1-score on GOOD, Accuracy otherwise) of competing methods on eight datasets, along with the percentage of trained parameters (TP; excluding embeddings). Attribute injected PLMs that perform worse than the base model $\mathcal{B}$ are *italicized*. Among $\mathcal{B}$-based models, best systems are in **bold** and the second-best are underlined. Asterisk (*) signifies a significant difference between our model and the second-best model (paired bootstrap resampling; $p < 0.05$).

Model	Y2013	Y2014	IMDB	AAPR	POLMED	FOOD	GOOD	BEER
$\mathcal{B}$ + INJECTORS	70.86	71.69	58.90	67.10	47.27	45.01	57.78	57.87
– bias injection	70.33	71.24	58.54	66.45	46.67	44.86	57.06	57.67
– weight injection	70.51	71.27	58.55	66.85	45.66	44.74	57.61	57.29
– task adapter	69.21	69.68	57.25	65.55	46.89	44.30	56.57	57.28
– attribute drop	69.29	70.94	57.69	65.55	46.33	44.48	56.62	57.41
– post-aggregation	70.76	71.35	58.63	64.42	47.27	43.30	57.78	57.69
– low-rank	OOM	OOM	OOM	OOM	OOM	OOM	OOM	OOM
– PHM	68.27	69.05	56.74	65.15	46.16	43.89	55.99	56.51

Table 3: Performance of INJECTORS and versions thereof without some of our proposed components (second block), training mechanisms (third block), and parameter-saving methods (fourth block). OOM denotes the model does not run on our experimental setup due to out of memory error.

	Y2013-COLD		
Model	20%	50%	80%
BERT-base ($\mathcal{B}$)	65.24	63.88	58.23
$\mathcal{B}$ + CHIM	65.85	64.12	58.56
$\mathcal{B}$ + TINYMAA	66.26	64.20	57.68
$\mathcal{B}$ + MAA	67.03	64.22	58.48
$\mathcal{B}$ + INJECTORS	**67.89**	**64.32**	**59.23**
HCSC	63.6	60.8	53.8
	IMDB-COLD		
Model	20%	50%	80%
BERT-base ($\mathcal{B}$)	50.24	47.77	41.37
$\mathcal{B}$ + CHIM	51.08	47.81	41.22
$\mathcal{B}$ + TINYMAA	52.45	48.27	40.78
$\mathcal{B}$ + MAA	53.52	49.25	41.61
$\mathcal{B}$ + INJECTORS	**55.62**	**49.87**	**41.78**
HCSC	50.5	45.6	36.8

Table 4: Performance on the Y2013-COLD and IMDB-COLD. Best systems are shown in bold. Models perform worse than $\mathcal{B}$ are colored red.

Model	A	RPT→A
$\mathcal{B}$ + MAA	53.65	51.27 (−4.43%)
$\mathcal{B}$ + INJECTORS	55.58	56.23 (+1.17%)
Model	R	APT→R
$\mathcal{B}$ + MAA	51.79	50.57 (−2.36%)
$\mathcal{B}$ + INJECTORS	53.62	55.48 (+3.47%)
Model	P	ART→P
$\mathcal{B}$ + MAA	52.73	50.84 (−3.57%)
$\mathcal{B}$ + INJECTORS	55.44	55.63 (+0.34%)
Model	T	ARP→T
$\mathcal{B}$ + MAA	58.25	56.08 (−3.73%)
$\mathcal{B}$ + INJECTORS	59.39	60.27 (+1.48%)

Table 5: Performance of models on single-task BEER (A: appearance, R: aroma, P: palette, T: taste). The arrow (→) indicates that the attribute-specific adapters of the model in the right-hand side (e.g., A) are initialized using parameters of the left-hand side model (e.g., RPT) and are frozen during training.

On Model Modularity Adapters allow us to compose multiple modules with different functionalities, possibly trained from different models. For example, when predicting the review rating of a specific user on a *new aspect* of a product, we may want to use the learned representation of a user on a previous model (i.e., a review rating prediction model focused on previously known aspects of the product). This is especially crucial when there is fewer data for such new tasks[5]. Since INJECTORS are basically a sequence of adapters, we expect that modular composition across different models

[5]We leave the exploration of attribute transfer to completely new tasks in future work due to the absence of multi-task datasets with common attributes.

is also effective in our setting. In this section, we verify this using the following experiment.

We use the BEER dataset, which is a multi-task aspect rating prediction dataset with four different aspects. We divide the dataset into two subsets: (1) a single-task *target* dataset and (2) a 3-task *source* dataset. We train the model using the source dataset, obtaining attribute-specific parameters. We then transfer these parameters when training the model using the target dataset, and only fine-tune the parameters of the task adapter and the classifier.

We split the training dataset into four parts, one for each aspect, to ensure that there are *no overlapping training examples* between source and target datasets. For each aspect, we treated it as the target and combined the three non-target datasets as the source dataset. We experimented with MAA and INJECTORS, and report the results in Table 5. When compared to the same model trained directly on the target task (second column), our model is able to achieve a small improvement even when the attribute-specific parameters are fixed and learned from a different task (third column). On the other hand, MAA shows a decrease in performance in all cases.

7 Conclusions

We considered the use of attributes as additional contexts when fine-tuning pretrained language models. We proposed the INJECTOR module, an extension of adapters that also accepts attributes as input. Our method considers two kinds of injection strategies, uses parameter-saving techniques and introduces training mechanisms to account for sparse and multi-labeled attributes. We also introduced ATTRIB, a collection of 14 datasets to evaluate attribute injection methods. Experiments on this benchmark of various classification tasks showed that our method improves substantially over previous methods. Finally, we conducted extensive analyses on how INJECTORS handle attribute sparsity and verify their modularity. In the future, we plan to apply our methods to real-world data where there are millions of attributes. We also plan to explore the use of attribute injection methods for text generation tasks, i.e. injecting attributes when generating texts. Our code and the ATTRIB benchmark will be publicly available online at https://github.com/rktamplayo/Injector.

Acknowledgments

We would like to thank Jaewook Kang and other members of the NAVER AI Lab for their insightful comments. Reinald is supported by a Google PhD Fellowship.

References

Reinald Kim Amplayo. 2019. Rethinking attribute representation and injection for sentiment classification. In *EMNLP-IJCNLP*, pages 5602–5613.

Reinald Kim Amplayo, Jihyeok Kim, Sua Sung, and Seung-won Hwang. 2018. Cold-start aware user and product attention for sentiment classification. In *ACL*, pages 2535–2544.

Yoshua Bengio, Aaron C. Courville, and Pascal Vincent. 2013. Representation learning: A review and new perspectives. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 35:1798–1828.

Tom B. Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, J. Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, T. Henighan, R. Child, A. Ramesh, Daniel M. Ziegler, Jeff Wu, Clemens Winter, Christopher Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. 2020. Language models are few-shot learners. In *NeurIPS*.

Huimin Chen, Maosong Sun, Cunchao Tu, Yankai Lin, and Zhiyuan Liu. 2016. Neural sentiment classification with user and product attention. In *EMNLP*, pages 1650–1659.

Asa Cooper Stickland, Alexandre Berard, and Vassilina Nikoulina. 2021. Multilingual domain adaptation for NMT: Decoupling language and domain information with adapters. In *Proceedings of the Sixth Conference on Machine Translation*, pages 578–598, Online. Association for Computational Linguistics.

Sumanth Dathathri, Andrea Madotto, Janice Lan, Jane Hung, Eric Frank, Piero Molino, Jason Yosinski, and Rosanne Liu. 2019. Plug and play language models: A simple approach to controlled text generation. *arXiv preprint arXiv:1912.02164*.

Ludovic Denoyer and P. Gallinari. 2006. The wikipedia xml corpus. In *INEX*.

J. Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. Bert: Pre-training of deep bidirectional transformers for language understanding. In *NAACL*.

Zi-Yi Dou. 2017. Capturing user and product information for document level sentiment analysis with deep memory network. In *EMNLP*, pages 521–526.

Angela Fan, David Grangier, and Michael Auli. 2018. Controllable abstractive summarization. In *NMT@ACL*.

Jessica Ficler and Y. Goldberg. 2017. Controlling linguistic style aspects in neural language generation. *ArXiv*, abs/1707.02633.

T. Fukuhara, H. Nakagawa, and T. Nishida. 2007. Understanding sentiment of people from news articles: Temporal sentiment analysis of social events. In *ICWSM*.

Tianyu Gao, Adam Fisch, and Danqi Chen. 2021. Making pre-trained language models better few-shot learners. In *ACL-IJCNLP*.

W. Gao, Naoki Yoshinaga, Nobuhiro Kaji, and M. Kitsuregawa. 2013. Modeling user leniency and product popularity for sentiment classification. In *IJCNLP*.

Junxian He, Chunting Zhou, Xuezhe Ma, Taylor Berg-Kirkpatrick, and Graham Neubig. 2022. Towards a unified view of parameter-efficient transfer learning. In *International Conference on Learning Representations*.

Ruidan He, Linlin Liu, Hai Ye, Qingyu Tan, Bosheng Ding, Liying Cheng, Jiawei Low, Lidong Bing, and Luo Si. 2021. On the effectiveness of adapter-based tuning for pretrained language model adaptation. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 2208–2222, Online. Association for Computational Linguistics.

N. Houlsby, A. Giurgiu, Stanislaw Jastrzebski, Bruna Morrone, Quentin de Laroussilhe, Andrea Gesmundo, Mona Attariyan, and S. Gelly. 2019. Parameter-efficient transfer learning for nlp. In *ICML*.

Arash Joorabchi and A. Mahdi. 2011. An unsupervised approach to automatic classification of scientific literature utilizing bibliographic metadata. *Journal of Information Science*, 37:499 – 514.

Nitish Shirish Keskar, Bryan McCann, Lav R Varshney, Caiming Xiong, and Richard Socher. 2019. Ctrl: A conditional transformer language model for controllable generation. *arXiv preprint arXiv:1909.05858*.

Yuta Kikuchi, Graham Neubig, Ryohei Sasano, Hiroya Takamura, and M. Okumura. 2016. Controlling output length in neural encoder-decoders. In *EMNLP*.

Jihyeok Kim, Reinald Kim Amplayo, Kyungjae Lee, Sua Sung, Minji Seo, and Seung-won Hwang. 2019. Categorical metadata representation for customized text classification. *TACL*, 7:201–215.

Jooyeon Kim, Dongwoo Kim, and Alice H. Oh. 2017. Joint modeling of topics, citations, and topical authority in academic corpora. *Transactions of the Association for Computational Linguistics*, 5:191–204.

Anne Lauscher, Olga Majewska, Leonardo F. R. Ribeiro, Iryna Gurevych, Nikolai Rozanov, and Goran Glavaš. 2020. Common sense or world knowledge? investigating adapter-based knowledge injection into pretrained transformers. In *Proceedings of Deep Learning Inside Out (DeeLIO): The First Workshop on Knowledge Extraction and Integration for Deep Learning Architectures*, pages 43–49, Online. Association for Computational Linguistics.

Xiang Lisa Li and Percy Liang. 2021. Prefix-tuning: Optimizing continuous prompts for generation. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 4582–4597, Online. Association for Computational Linguistics.

Shuhua Liu and Thomas Forss. 2014. Web content classification based on topic and sentiment analysis of text. In *KDIR*.

Xiao Liu, Yanan Zheng, Zhengxiao Du, Ming Ding, Yujie Qian, Zhilin Yang, and Jie Tang. 2021. Gpt understands, too. *arXiv:2103.10385*.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, M. Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Roberta: A robustly optimized bert pretraining approach. *ArXiv*, abs/1907.11692.

Yunfei Long, Mingyu Ma, Qin Lu, Rong Xiang, and Chu-Ren Huang. 2018. Dual memory network model for biased product review classification. In *WASSA@EMNLP*, pages 140–148.

I. Loshchilov and F. Hutter. 2019. Decoupled weight decay regularization. In *ICLR*.

Dehong Ma, Sujian Li, Xiaodong Zhang, Houfeng Wang, and Xu Sun. 2017. Cascading multiway attentions for document-level sentiment classification. In *IJCNLP*, pages 634–643.

Rabeeh Karimi Mahabadi, James Henderson, and Sebastian Ruder. 2021. Compacter: Efficient low-rank hypercomplex adapter layers. *ArXiv*, abs/2106.04647.

Bodhisattwa Prasad Majumder, Shuyang Li, Jianmo Ni, and Julian McAuley. 2019. Generating personalized recipes from historical user preferences. In *EMNLP/IJCNLP*.

Yuning Mao, Lambert Mathias, Rui Hou, Amjad Almahairi, Hao Ma, Jiawei Han, Wen-tau Yih, and Madian Khabsa. 2021. Unipelt: A unified framework for parameter-efficient language model tuning. *arXiv preprint arXiv:2110.07577*.

Julian McAuley, J. Leskovec, and Dan Jurafsky. 2012. Learning attitudes and attributes from multi-aspect reviews. In *ICDM*.

Tomas Mikolov, Kai Chen, G. Corrado, and J. Dean. 2013. Efficient estimation of word representations in vector space. In *ICLR*.

Jianmo Ni, Jiacheng Li, and Julian McAuley. 2019. Justifying recommendations using distantly-labeled reviews and fine-grained aspects. In *EMNLP-IJCNLP*, pages 188–197.

Ethan Perez, Florian Strub, Harm de Vries, Vincent Dumoulin, and Aaron Courville. 2018. Film: Visual reasoning with a general conditioning layer. In *Proceedings of the Thirty-Second AAAI Conference on Artificial Intelligence and Thirtieth Innovative Applications of Artificial Intelligence Conference and Eighth AAAI Symposium on Educational Advances in Artificial Intelligence*, AAAI'18/IAAI'18/EAAI'18. AAAI Press.

Jonas Pfeiffer, Aishwarya Kamath, Andreas Rücklé, Kyunghyun Cho, and Iryna Gurevych. 2021. AdapterFusion: Non-destructive task composition for transfer learning. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*, pages 487–503, Online. Association for Computational Linguistics.

Xipeng Qiu, Tianxiang Sun, Yige Xu, Yunfan Shao, Ning Dai, and Xuanjing Huang. 2020. Pre-trained models for natural language processing: A survey. *Science China Technological Sciences*, pages 1–26.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J Liu. 2020. Exploring the limits of transfer learning with a unified text-to-text transformer. *JMLR*, 21:1–67.

D. Ramage, David Hall, Ramesh Nallapati, and Christopher D. Manning. 2009. Labeled lda: A supervised topic model for credit attribution in multi-labeled corpora. In *EMNLP*.

M. Rosen-Zvi, T. Griffiths, M. Steyvers, and Padhraic Smyth. 2004. The author-topic model for authors and documents. In *UAI*.

Evan Sandhaus. 2008. The new york times annotated corpus. *Linguistic Data Consortium, Philadelphia*, 6(12):e26752.

Timo Schick and Hinrich Schütze. 2021. It's not just size that matters: Small language models are also few-shot learners. In *NAACL*.

Rico Sennrich, B. Haddow, and Alexandra Birch. 2016. Controlling politeness in neural machine translation via side constraints. In *NAACL*.

Duyu Tang, Bing Qin, and Ting Liu. 2015. Learning semantic representations of users and products for document level sentiment classification. In *ACL-IJCNLP*, pages 1014–1023.

Yi Tay, A. Zhang, Anh Tuan Luu, J. Rao, Shuai Zhang, Shuohang Wang, Jie Fu, and S. C. Hui. 2019. Lightweight and efficient neural natural language processing with quaternion networks. In *ACL*.

Ashish Vaswani, Noam M. Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *NIPS*.

Mengting Wan and Julian McAuley. 2018. Item recommendation on monotonic behavior chains. In *RecSys*.

Alex Wang, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel R. Bowman. 2018. Glue: A multi-task benchmark and analysis platform for natural language understanding. In *ICLR*.

Yunli Wang, Yu Wu, Lili Mou, Zhoujun Li, and Wen-Han Chao. 2019. Harnessing pre-trained neural networks with rules for formality style transfer. In *EMNLP*.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, R'emi Louf, Morgan Funtowicz, and Jamie Brew. 2020. Transformers: State-of-the-art natural language processing. In *EMNLP*.

Zhen Wu, Xin-Yu Dai, Cunyan Yin, Shujian Huang, and Jiajun Chen. 2018. Improving review representations with user attention and product attention for sentiment classification. In *AAAI*.

Keyulu Xu, Weihua Hu, J. Leskovec, and S. Jegelka. 2019. How powerful are graph neural networks? In *ICLR*.

Min Yang, Jincheng Mei, Heng Ji, Wei Zhao, Zhou Zhao, and Xiaojun Chen. 2017. Identifying and tracking sentiments and topics from social media texts during natural disasters. In *EMNLP*.

Pengcheng Yang, Xu Sun, Wei Li, and Shuming Ma. 2018. Automatic academic paper rating based on modularized hierarchical convolutional neural network. In *ACL*.

A. Zhang, Yi Tay, Shuai Zhang, Alvin Chan, A. Luu, S. C. Hui, and Jie Fu. 2021a. Beyond fully-connected layers with quaternions: Parameterization of hypercomplex multiplications with 1/n parameters. In *ICLR*.

Youjia Zhang, Jin Wang, Liang-Chih Yu, and Xuejie Zhang. 2021b. Ma-bert: Learning representation by incorporating multi-attribute knowledge in transformers. In *FINDINGS*.

Rui Zhao and K. Mao. 2017. Topic-aware deep compositional models for sentence classification. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 25:248–260.

Tony Zhao, Eric Wallace, Shi Feng, D. Klein, and Sameer Singh. 2021. Calibrate before use: Improving few-shot performance of language models. In *ICML*.

Yukun Zhu, Ryan Kiros, Rich Zemel, Ruslan Salakhutdinov, Raquel Urtasun, Antonio Torralba, and Sanja Fidler. 2015. Aligning books and movies: Towards story-like visual explanations by watching movies and reading books. In *ICCV*, pages 19–27.

A Appendix

A.1 Descriptions of Newly Introduced Datasets

This section describes how we procured the three datasets we introduce in this paper:

1. FOOD: We used the dataset gathered in Majumder et al. (2019), which was used as a personalized recipe generation dataset. We repurposed the dataset for a new classification task and used the recipes as input text and the duration (in minutes) as output class. We removed instances with outliers: (1) recipes that took less than 5 minutes and more than 150 minutes; (2) recipes with more than 500 tokens or less than 10 tokens; and (3) tags with more than 50 labels. We also removed from the attribute vocabulary tags that explicitly indicate the recipe duration (e.g., `60-minutes-or-less`) and those that are used on almost all instances (e.g., `time-to-make`). We shuffled the data and used 10% each for the development and test sets, and the remaining 80% for the training set.

2. GOOD: We used the review corpus gathered in Wan and McAuley (2018), which was also used for spoiler detection. Since the split is unfortunately not publicly shared, we created our own split. We first removed very short (less than 32 tokens) and very long (more than 256) reviews as they were outliers. We then divided the data into three splits, with two 10K splits as the development and test sets, and the remaining split as the training set.

3. BEER: We used the review corpus gathered in McAuley et al. (2012). We removed outliers and split the dataset into three using the same method we did with Goodreads.

A.2 AttrIB Attribute Analysis

We conducted an attribute analysis to check whether the attributes in these datasets are indeed useful for the tasks. Specifically, using the training set, we create attribute-specific distribution over classes $q_z(y)$ for each attribute z to represent attribute bias, i.e.:

$$q_z(y) = \sum_{x \in \mathcal{D}_z} \mathbb{1}(x, y)/|\mathcal{D}_z| \qquad (12)$$

where $\mathcal{D}_z$ is the subset of the training set where attribute z exists and $\mathbb{1}(x, y)$ is the indicator function that returns 1 if class y is the class of training example x. We then create the same attribute-specific class distribution $p_z(y)$ for the dev set and compare the similarity of both distributions using KL-divergence. Finally, we compare with the KL-divergence of random (i.e., uniform distribution) and majority (i.e., overall class distribution) baselines to see whether attributes provide better biases than when selecting at random or the majority class.

Table 6 shows the KL-divergence of random, majority, and attributes, both when z is a single attribute and all attributes combined. We can observe two things in the table. Firstly, using a single attribute usually does not perform better than the majority baseline. This is in contrast to experiments in some of the previous work (Chen et al., 2016), where text classification models based on neural networks improved when incorporated with just one out of all attributes. This means that these models can potentially learn beyond providing attribute-specific biases, such as jointly modeling attributes and texts. Secondly, combining all attributes provides the best bias on most datasets, with the exception of POLMED. In this dataset, the audience attribute is worse than the random baseline, but it might be useful in some cases. In fact, the best configuration is actually combining both the media source and the audience attributes, which has a KL-divergence of 1.726. Therefore, attributes may either be helpful or detrimental depending on multiple factors, e.g., other attributes and (possibly) textual input. Models thus need to effectively determine if the attribute is useful for each of the examples.

Y2013		Y2014	
random	1.608	random	1.608
majority	1.358	majority	1.379
user	1.406	user	1.425
product	1.419	product	1.415
all	**1.265**	all	**1.279**
IMDB		AAPR	
random	2.300	random	0.693
majority	2.087	majority	0.693
user	2.132	author	1.357
product	2.314	research area	0.665
all	**1.873**	all	**0.664**
POLMED		FOOD	
random	2.195	random	2.769
majority	1.851	majority	2.352
politician	1.790	user	2.306
media source	**1.728**	ingredient	3.708
audience	3.640	tag	2.277
political bias	1.844		
all	1.770	all	**2.275**
GOOD		BEER	
random	0.693	random	2.195
majority	0.216	majority	1.656
user	0.246	user	1.754
book	0.326	beer	1.769
rating	0.215	rating	1.363
all	**0.215**	all	**1.362**

Table 6: KL-divergence of baseline and attribute-specific distributions over classes to the dev set distributions. Best values are in **bold**.

A.3 Cold-Start Dataset Statistics

Table 7 reports the statistics of the cold-start datasets in ATTRIB.

A.4 Parameter Analysis of Weight-based Injection

Recall that we define $\mathbf{W}_{z_j} \in \mathbb{R}^{d_h \times d_a}$ as follows:

$$\mathbf{W}_{z_j} = g_{weight}(\mathbf{z}_j) + \mathbf{C}_{weight} \tag{13}$$

In a naive setting, we can trivially use a projection function as our g_{weight}, which would linearly transform $\mathbf{z}_j \in \mathbb{R}^{d_z}$ into the shape $d_h \times d_a$. This would need a weight tensor of size $d_z \times d_h \times d_a$, which can be prohibitively large. This parameter dominates all the other parameters in the module, thus the overall parameter of the naive method is $\mathcal{O}(d_z * d_h * d_a)$.

Our parameter-saving methods remove this large tensor, but instead use three smaller parameters in hypercomplex space: the transform function $\sigma_o(\cdot)$ that is basically a linear transformation with a projection matrix of size $d_z \times d_a$ (Eq. 6), the vector $\mathbf{s}_o$ of size d_h/O^2, and the matrix $\mathbf{A}_o$ of size $O \times O$. Since we have O dimensions in our hypercomplex space, we have a total of $O*(d_z*d_a+d_h/O^2+O^2)$, which we can reduce as follows:

$$\begin{aligned} & O * (d_z * d_a + d_h/O^2 + O^2) \\ & = O * d_z * d_a + d_h/O + O^3 \\ & \approx O * d_z * d_a \\ & \approx d_z * d_a \end{aligned} \tag{14}$$

given that $O^3 \ll O*d_z*d_a$ and that we can treat O as a constant ($O = 4$ in our experiment). Thus the overall parameter when using our parameter-saving method is $\mathcal{O}(d_z * d_a)$. We emphasize that this is a huge improvement since the PLM hidden size d_h is usually the largest dimension.

The output weight $\mathbf{W}_{z_j}$ has a rank r of at most O^2+1, i.e., (1) the low-rank method (Eq. 7) outputs

	Y2013-COLD		
Dataset	20%	50%	80%
#Train	38.7K	16.1K	2.4K
#Attr. Vocab	2.6K	1.6K	0.7K
%Cold-start	20.5%	49.4%	79.6%

	IMDB-COLD		
Dataset	20%	50%	80%
#Train	44.3K	18.0K	2.5K
#Attr. Vocab	2.4K	1.5K	0.6K
%Cold-start	19.3%	48.7%	80.7%

Table 7: Cold-start dataset statistics. %Cold-start is the percentage of attributes with zero training examples. These datasets use the dev and test sets of the original datasets.

a matrix of rank $r = 1$; (2) the Kronecker product (Eq. 8) returns a matrix of rank $r = O$; and finally, (3) the sum of multiple matrices (Eq. 9) has a rank $r \leq O^2$.

Towards Robust Neural Retrieval with Source Domain Synthetic Pre-Finetuning

Revanth Gangi Reddy[1], **Vikas Yadav**[3]*, **Md Arafat Sultan**[2], **Martin Franz**[2], **Vittorio Castelli**[3]*, **Heng Ji**[1], **Avirup Sil**[2]
[1]University of Illinois at Urbana-Champaign [2]IBM Research AI [3]Amazon AWS AI
{revanth3,hengji}@illinois.edu, arafat.sultan@ibm.com,
{franzm,avi}@us.ibm.com, vittorca@amazon.com

Abstract

Research on neural IR has so far been focused primarily on standard supervised learning settings, where it outperforms traditional term matching baselines. Many practical use cases of such models, however, may involve previously unseen target domains. In this paper, we propose to improve the out-of-domain generalization of Dense Passage Retrieval (DPR)—a popular choice for neural IR—through synthetic data augmentation ***only in the source domain***. We empirically show that pre-finetuning DPR with additional synthetic data in its source domain (Wikipedia), which we generate using a fine-tuned sequence-to-sequence generator[1], can be a low-cost yet effective first step towards its generalization. Across five different test sets, our augmented model shows more robust performance than DPR in both in-domain and zero-shot out-of-domain evaluation.

1 Introduction

Traditional approaches to information retrieval (IR) such as TF-IDF (Salton and McGill, 1986) and BM25 (Robertson and Zaragoza, 2009) rely on lexical matching for query-passage alignment. In contrast, neural IR encodes passages and questions into continuous vector representations, enabling deeper semantic matching. Modern neural IR systems (Lee et al., 2019; Chang et al., 2019) based on pre-trained masked language models (MLM) (Devlin et al., 2019) typically employ a dual encoder architecture (Bromley et al., 1993), where two separate MLMs encode the question and the passage. Karpukhin et al. (2020) show that useful weak supervision for such systems can be derived from the related task of machine reading comprehension (MRC) (Kwiatkowski et al., 2019; Joshi et al., 2017). Their Dense Passage Retrieval (DPR) model demonstrates state-of-the-art (SOTA) in-domain performance on multiple Wikipedia-based datasets (Kwiatkowski et al., 2019; Joshi et al., 2017; Berant et al., 2013; Baudiš and Šedivỳ, 2015), outperforming both term matching baselines like BM25 and prior neural approaches, e.g., the Inverse Cloze Task (Lee et al., 2019) and latent learning of the retriever during MLM pre-training (Guu et al., 2020).

Despite its high in-domain utility, however, Reddy et al. (2021) show that DPR performance can drop significantly in novel test domains. They propose target domain synthetic data augmentation as a solution to this problem, which augments DPR with additional synthetic training data generated from target domain text. While this approach does indeed improve DPR scores in the new test domain, it has a key practical limitation: for every new target domain, it requires generating a new synthetic training corpus and re-training the model. Here we ask if an augmentation approach that only operates once in the source domain, and does not require re-training every time a new test domain is encountered, can also help improve domain generalization.

To better understand DPR's zero-shot out-of-domain (OOD) utility, we first run an empirical evaluation where both BM25 and DPR are applied to several out-of-domain test datasets. We observe that (i) DPR still holds an advantage over BM25 in near domain evaluation on Wikipedia-based datasets, but the difference is considerably lower than in the in-domain case, and (ii) In the far domain of biomedical text, DPR actually underperforms BM25. Our OOD evaluation is more comprehensive than Reddy et al. (2021), demonstrating the zero-shot utility of DPR in a more detailed and fine-grained manner.

Next we investigate if a one-off pre-finetuning of DPR with large amounts of ***source domain*** synthetic IR data can help improve its robustness to domain shift. Utilization of synthetic training data is common in related tasks such as machine reading comprehension (MRC) (Shakeri et al., 2020; Zhang

*Work done while at IBM Research AI

[1]Synthetic question generation code is available at: https://github.com/primeqa/primeqa/tree/main/primeqa/qg

Proceedings of the 29th International Conference on Computational Linguistics, pages 1065–1070
October 12–17, 2022.

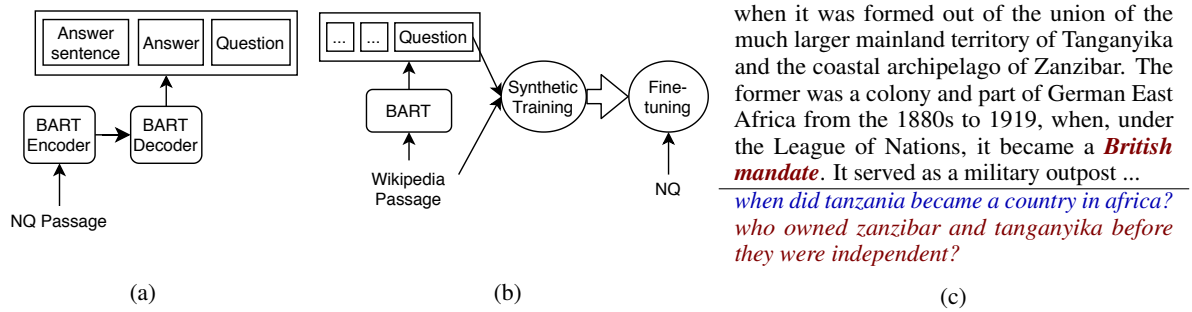

Figure 1: The proposed IR training pipeline and a synthetic example. (a) A BART encoder-decoder LM is fine-tuned on NQ for QA example generation; (b) Synthetic examples generated from Wikipedia passages are used to pre-finetune the neural IR model before fine-tuning on NQ; (c) Two synthetic questions output by our generator from the depicted Wikipedia passage, with corresponding answers highlighted in the text.

et al., 2020; Sultan et al., 2020). Nevertheless, a close examination of synthetic pre-finetuning as an augmentation technique is key for zero-shot neural IR due to the presence of highly effective and domain-agnostic term matching baselines like BM25.

We fine-tune a sequence-to-sequence generator on labeled MRC data and use it to generate synthetic IR examples from source domain passages (§2). Our experiments show that pre-finetuning DPR with these generated examples does indeed improve its accuracy on both in-domain and out-of-domain test sets. Crucially, the gap with BM25 in far domain evaluation is significantly reduced.

The main contributions of this paper are:

- We conduct an empirical evaluation of SOTA neural IR on multiple in-domain and out-of-domain test sets, showing how its utility varies in different test conditions.
- We show that a one-off ***source domain*** synthetic pre-finetuning step can significantly improve the robustness of neural IR, with improvements on five different test sets, including in the practical zero-shot setting.

2 Source Domain Synthetic Pre-Finetuning

In this section, we describe the procedure for synthetic pre-finetuning of the DPR model. We first detail how we train the sequence-to-sequence generator and generate source domain syntheic data from it. Next, we describe how this data is used for training the DPR model.

Let c be a text corpus and $d \in c$ be a document. An IR example, more specifically a passage retrieval example, consists of a question q and a passage p in d such that p contains an answer a to q. Let s be the sentence in p that contains a.

We first train an example generator by fine-tuning BART (Lewis et al., 2020a)—a pre-trained encoder-decoder language model—to generate an ordered triple (s, a, q) from an input passage p. This procedure in essence uses generation to first identify a candidate sentence s in p, then extract a candidate answer a from s, and finally generate a corresponding question q. In practice, we approximate the generation of s by generating only its first and last words. Finally, (q, p) is retained as a synthetic IR example. Labeled (p, s, a, q) tuples needed for the supervision of this model are taken from Natural Questions (NQ) (Kwiatkowski et al., 2019), an existing MRC dataset over Wikipedia articles.

With the generator, we produce positive synthetic pre-finetuning examples for DPR from Wikipedia passages. Following Sultan et al. (2020), we use top-p top-k sampling (Holtzman et al., 2020) to promote diversity in the generated examples. Training and inference of the synthetic example generator are depicted in Figures 1a and 1b, respectively. Figure 1c shows two example questions output by the generator from a Wikipedia passage.

To obtain a negative sample for each generated question q, we retrieve passages from Wikipedia using BM25 and randomly sample one that does not contain the generated answer a. Following Karpukhin et al. (2020), we also use in-batch negative samples for training. After pre-finetuning with synthetic examples, we fine-tune the model with IR examples derived from NQ. We name this synthetically augmented DPR model *AugDPR*. We refer

the reader to (Karpukhin et al., 2020) for a more detailed description of the DPR training process.

3 Experimental Setup

3.1 Datasets

We briefly describe our datasets in this section. Statistics for each dataset are shown in Table 1.

Dataset	Domain	Passages	Questions
NQ	Wikipedia	21.0M	3,610
TriviaQA	Wikipedia	21.0M	11,313
WebQuestions	Wikipedia	21.0M	2,032
WikiMovies	Wikipedia	21.0M	9,952
BioASQ	Biomedical	37.4M	1092

Table 1: Statistics of the retrieval corpora and the test sets we use to evaluate all IR models.

Training and In-Domain Evaluation: We train all systems on Natural Questions (NQ) (Kwiatkowski et al., 2019), a dataset with questions derived from Google's search log and their human-annotated answers from Wikipedia articles. Lewis et al. (2020b) report that 30% of the NQ test set questions have near-duplicate paraphrases in the training set and 60–70% of the test answers are also present in the training set. For this reason, in addition to the entire NQ test set, we also use the non-overlapping subsets released by Lewis et al. (2020b) for in-domain evaluation.

Near Domain Evaluation: For zero-shot near domain evaluation, where Wikipedia articles constitute the retrieval corpus, we use the test sets of three existing datasets.

TriviaQA (Joshi et al., 2017) contains questions collected from trivia and quiz league websites, which are created by Trivia enthusiasts.

WebQuestions (WQ) (Berant et al., 2013) consists of questions obtained using the Google Suggest API, and answers selected from entities in Freebase by AMT workers.

WikiMovies (Miller et al., 2016) contains question-answer pairs on movies, built using the OMDb and MovieLens databases. We use the test split adopted in (Chen et al., 2017).

Far Domain Evaluation. For zero-shot far domain evaluation, we use a biomedical dataset.

BioASQ (Tsatsaronis et al., 2015) is a competition[2] on large-scale biomedical semantic indexing and QA. We evaluate on all factoid question-answer pairs from the training and test sets of task 8B.

3.2 Setup

Training: We train the synthetic example generator using the *(question, passage, answer)* triples from NQ. The model is trained for 3 epochs with a learning rate of 3e-5 and batch size of 24. We then randomly sample 2M passages from the 21M-passage Wikipedia corpus and generate around four synthetic questions per passage. For top-p top-k sampling, we use $p = 0.95$ and $k = 10$.

During synthetic pre-finetuning of DPR, for each of the 2M passages, we randomly select one of its synthetic questions at each epoch to create a synthetic example. After six epochs of synthetic pre-finetuning with a learning rate of 1e-5 and batch size of 1024, we fine-tune DPR on NQ for twenty epochs with a learning rate of 1e-5 and batch size of 128 to get the AugDPR model.

Baselines and Metrics: We evaluate BM25 as a term matching baseline. Our BM25 baseline is based on Lucene[3] implementation. BM25 parameters $b = 0.75$ (document length normalization) and $k_1 = 1.2$ (term frequency scaling) worked best. As our neural baseline, we use the DPR-single model trained on NQ and made public[4] by Karpukhin et al. (2020). Both DPR and AugDPR use BERT-base-uncased for question and passage encoding. As in (Karpukhin et al., 2020), our evaluation metric is top-k retrieval accuracy, which is the percentage of questions with at least one answer in the top k retrieved passages.

4 Results and Discussion

Table 2 shows NQ results on the entire test set as well as on the two subsets released by Lewis et al. (2020b). Synthetic pre-finetuning yields larger gains on the non-overlapping splits, with up to a 4-point improvement in top-1 retrieval accuracy.

To assess the cross-domain utility of AugDPR, we evaluate it zero shot on both near and far domain test sets. Table 3 shows the results. For comparison, we also show results for supervised models reported by Karpukhin et al. (2020) on TriviaQA and WebQuestions where the DPR model was trained directly on the training splits of these datasets. For the near domain datasets, both DPR and AugDPR

[2] http://bioasq.org/participate/challenges

[3] https://lucene.apache.org/

[4] https://github.com/facebookresearch/DPR

Model	Total			No answer overlap			No question overlap		
	Top-1	Top-10	Top-20	Top-1	Top-10	Top-20	Top-1	Top-10	Top-20
BM25	30.5	54.5	62.5	26.4	47.1	54.7	31.0	52.1	59.8
DPR	46.3	74.9	80.1	32.2	62.2	68.7	37.4	68.5	75.3
AugDPR	**46.8**	**76.0**	**80.8**	**36.0**	**65.0**	**70.8**	**41.4**	**70.8**	**76.6**

Table 2: NQ top-k retrieval results. Performance improves across the board with synthetic pre-finetuning (AugDPR), but more on the non-overlapping subsets of Lewis et al. (2020b).

Model	Near Domains						Far Domain	
	TriviaQA		WebQuestions		WikiMovies		BioASQ	
	Top-20	Top-100	Top-20	Top-100	Top-20	Top-100	Top-20	Top-100
BM25	66.9	76.7	55.0	71.1	54.0	69.3	**42.1**	50.5
DPR	69.0	78.7	63.0	78.3	69.8	78.1	34.7	46.9
AugDPR	**72.2**	**81.1**	**71.1**	**80.8**	**72.5**	**80.7**	41.4	**52.4**
Supervised	79.4	85.0	73.2	81.4	-	-	-	-

Table 3: Zero-shot neural retrieval accuracy improves with synthetic pre-finetuning (AugDPR) in all out-of-domain test settings. However, BM25 remains a strong baseline on the far domain dataset of BioASQ. The numbers for the supervised models are taken from (Karpukhin et al., 2020).

outperform BM25 by a sizable margin; additionally, AugDPR consistently outperforms DPR. Furthermore, performance of AugDPR on WebQuestions is comparable to that of the supervised model. On the far domain, however, we observe that BM25 is a rather strong baseline, with clearly better scores than DPR. The synthetic pre-finetuning of AugDPR reduces this gap considerably, resulting in a slightly lower top-20 score but a 2-point gain in top-100 score over BM25.

To investigate the relative underperformance of neural IR on BioASQ, we take a closer look at the vocabularies of the two domains of Wikipedia articles and biomedical literature. Following Gururangan et al. (2020), we compute the overlap between the 10k most frequent tokens (excluding stop words) in the two domains, represented by 3M randomly sampled passages from each. We observe a vocabulary overlap of only 17%, which shows that the two domains are considerably different in terminology, explaining in part the performance drop in our neural models. Based on these results, we also believe that performance of neural IR in distant target domains can be significantly improved via pre-finetuning on synthetic examples that are generated from raw text in the target domain. We plan to explore this idea in future work.

We also examine the lexical overlap between the questions and their passages, since a high overlap would favor term matching methods like BM25. We find that the coverage of the question tokens in the respective gold passages is indeed higher in BioASQ: 72.1%, compared to 58.6% and 63.0% in NQ and TriviaQA, respectively.

To analyze how much synthetic data is required, we experiment with pre-finetuning using 1M and 4M synthetic examples while keeping the number of training updates fixed. As Table 4 shows, we do not see any improvements from using more examples beyond 2M.

Karpukhin et al. (2020) report that DPR finetuning takes around a day on eight 32GB GPUs, which is a notable improvement over more computationally intensive pre-training approaches like (Lee et al., 2019; Guu et al., 2020). Our synthetic pre-finetuning takes around two days on four 32GB GPUs, which is comparable with finetuning in terms of computational overhead.

Model	Top-10	Top-20	Top-100
DPR	73.6	78.1	85.0
AugDPR-1M	74.4	79.2	85.5
AugDPR-2M	74.8	79.7	85.9
AugDPR-4M	74.6	79.1	85.9

Table 4: Retrieval accuracy on the Natural Questions development set with varying number of synthetic examples (1M vs 2M vs 4M) during pre-finetuning.

5 Conclusion

We have shown that pre-finetuning a SOTA neural IR model using large amounts of *source domain* synthetic data improves its robustness in zero-shot application settings. Our experiments show consistent performance gains on five in-domain and out-of-domain test sets, including a far target domain that has significant vocabulary mismatch with the training domain. Future work will explore incorporating more control into the generation of synthetic data to increase its diversity and also to overcome potential biases in finetuning data.

References

Petr Baudiš and Jan Šedivỳ. 2015. Modeling of the question answering task in the yodaqa system. In *International Conference of the Cross-Language Evaluation Forum for European Languages*, pages 222–228. Springer.

Jonathan Berant, Andrew Chou, Roy Frostig, and Percy Liang. 2013. Semantic parsing on freebase from question-answer pairs. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*, pages 1533–1544.

Jane Bromley, James W Bentz, Léon Bottou, Isabelle Guyon, Yann LeCun, Cliff Moore, Eduard Säckinger, and Roopak Shah. 1993. Signature verification using a "siamese" time delay neural network. *International Journal of Pattern Recognition and Artificial Intelligence*, 7(04):669–688.

Wei-Cheng Chang, X Yu Felix, Yin-Wen Chang, Yiming Yang, and Sanjiv Kumar. 2019. Pre-training tasks for embedding-based large-scale retrieval. In *International Conference on Learning Representations*.

Danqi Chen, Adam Fisch, Jason Weston, and Antoine Bordes. 2017. Reading wikipedia to answer open-domain questions. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1870–1879.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. Bert: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186.

Suchin Gururangan, Ana Marasović, Swabha Swayamdipta, Kyle Lo, Iz Beltagy, Doug Downey, and Noah A. Smith. 2020. Don't stop pretraining: Adapt language models to domains and tasks. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 8342–8360, Online. Association for Computational Linguistics.

Kelvin Guu, Kenton Lee, Zora Tung, Panupong Pasupat, and Ming-Wei Chang. 2020. Realm: Retrieval-augmented language model pre-training. *arXiv preprint arXiv:2002.08909*.

Ari Holtzman, Jan Buys, Li Du, Maxwell Forbes, and Yejin Choi. 2020. The curious case of neural text degeneration. In *8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, April 26-30, 2020*. OpenReview.net.

Mandar Joshi, Eunsol Choi, Daniel S Weld, and Luke Zettlemoyer. 2017. Triviaqa: A large scale distantly supervised challenge dataset for reading comprehension. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1601–1611.

Vladimir Karpukhin, Barlas Oğuz, Sewon Min, Ledell Wu, Sergey Edunov, Danqi Chen, and Wen-tau Yih. 2020. Dense passage retrieval for open-domain question answering. *arXiv preprint arXiv:2004.04906*.

Tom Kwiatkowski, Jennimaria Palomaki, Olivia Redfield, Michael Collins, Ankur Parikh, Chris Alberti, Danielle Epstein, Illia Polosukhin, Jacob Devlin, Kenton Lee, et al. 2019. Natural questions: a benchmark for question answering research. *Transactions of the Association for Computational Linguistics*, 7:453–466.

Kenton Lee, Ming-Wei Chang, and Kristina Toutanova. 2019. Latent retrieval for weakly supervised open domain question answering. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 6086–6096.

Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Veselin Stoyanov, and Luke Zettlemoyer. 2020a. BART: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7871–7880, Online. Association for Computational Linguistics.

Patrick Lewis, Pontus Stenetorp, and Sebastian Riedel. 2020b. Question and answer test-train overlap in open-domain question answering datasets. *arXiv preprint arXiv:2008.02637*.

Alexander Miller, Adam Fisch, Jesse Dodge, Amir-Hossein Karimi, Antoine Bordes, and Jason Weston. 2016. Key-value memory networks for directly reading documents. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 1400–1409.

Revanth Reddy, Bhavani Iyer, Md Arafat Sultan, Rong Zhang, Avirup Sil, Vittorio Castelli, Radu Florian, and Salim Roukos. 2021. Synthetic target domain supervision for open retrieval qa. In *Proceedings of the 44th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 1793–1797.

Stephen Robertson and Hugo Zaragoza. 2009. The probabilistic relevance framework: Bm25 and beyond. *Foundations and Trends in Information Retrieval*, 3(4):333–389.

Gerard Salton and Michael J. McGill. 1986. *Introduction to Modern Information Retrieval*. McGraw-Hill, Inc., USA.

Siamak Shakeri, Cicero Nogueira dos Santos, Henghui Zhu, Patrick Ng, Feng Nan, Zhiguo Wang, Ramesh Nallapati, and Bing Xiang. 2020. End-to-end synthetic data generation for domain adaptation of question answering systems. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 5445–5460, Online. Association for Computational Linguistics.

Md Arafat Sultan, Shubham Chandel, Ramón Fernandez Astudillo, and Vittorio Castelli. 2020. On the importance of diversity in question generation for qa. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 5651–5656.

George Tsatsaronis, Georgios Balikas, Prodromos Malakasiotis, Ioannis Partalas, Matthias Zschunke, Michael R Alvers, Dirk Weissenborn, Anastasia Krithara, Sergios Petridis, Dimitris Polychronopoulos, et al. 2015. An overview of the bioasq large-scale biomedical semantic indexing and question answering competition. *BMC bioinformatics*, 16(1):138.

Rong Zhang, Revanth Gangi Reddy, Md Arafat Sultan, Vittorio Castelli, Anthony Ferritto, Radu Florian, Efsun Sarioglu Kayi, Salim Roukos, Avi Sil, and Todd Ward. 2020. Multi-stage pre-training for low-resource domain adaptation. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 5461–5468, Online. Association for Computational Linguistics.

Parameter-Efficient Neural Reranking for Cross-Lingual and Multilingual Retrieval

Robert Litschko[1] **Ivan Vulić**[2] **Goran Glavaš**[1,3]
[1] Data and Web Science Group, University of Mannheim, Germany
[2] Language Technology Lab, University of Cambridge, UK
[3] Center for AI and Data Science (CAIDAS), University of Würzburg, Germany

Abstract

State-of-the-art neural (re)rankers are notoriously data-hungry which – given the lack of large-scale training data in languages other than English – makes them rarely used in multilingual and cross-lingual retrieval settings. Current approaches therefore commonly transfer rankers trained on English data to other languages and cross-lingual setups by means of multilingual encoders: they fine-tune *all* parameters of pretrained massively multilingual Transformers (MMTs, e.g., multilingual BERT) on English relevance judgments, and then deploy them in the target language(s). In this work, we show that two *parameter-efficient* approaches to cross-lingual transfer, namely Sparse Fine-Tuning Masks (SFTMs) and Adapters, allow for a *more lightweight* and *more effective* zero-shot transfer to multilingual and cross-lingual retrieval tasks. We first train language adapters (or SFTMs) via Masked Language Modelling and then train retrieval (i.e., reranking) adapters (SFTMs) on top, while keeping all other parameters fixed. At inference, this modular design allows us to compose the ranker by applying the (re)ranking adapter (or SFTM) trained with source language data together with the language adapter (or SFTM) of a target language. We carry out a large scale evaluation on the CLEF-2003 and HC4 benchmarks and additionally, as another contribution, extend the former with queries in three new languages: Kyrgyz, Uyghur and Turkish. The proposed parameter-efficient methods outperform standard zero-shot transfer with full MMT fine-tuning, while being more modular and reducing training times. The gains are particularly pronounced for low-resource languages, where our approaches also substantially outperform the competitive machine translation-based rankers.

1 Introduction

In recent years, neural rankers (Nogueira et al., 2019b; MacAvaney et al., 2019; Khattab and Zaharia, 2020), trained on large-scale datasets (Bajaj et al., 2016; Dietz et al., 2017; Craswell et al., 2021), have substantially pushed the performance on various retrieval benchmarks. Since such models are generally too computationally involved (i.e., too slow) for ad-hoc retrieval on large document collections, they are commonly leveraged as rerankers, i.e., they rerank the output of some fast model (e.g., BM25) that produces the initial ranking. Large-scale datasets for training neural rerankers, however, exist only in English, which impedes their adoption in retrieval scenarios that involve other languages: (a) monolingual retrieval in other languages and (b) cross-lingual information retrieval (CLIR) in which, for a given query in one language, one needs to determine relevance of documents written in one or more other languages.

While CLIR is often instantiated in the form of standalone tasks (e.g., to allow users from different countries to search over the aggregated global collection of COVID-19 news and findings in their native language (Casacuberta et al., 2021)), it also supports a range of IR-backed NLP tasks such as cross-lingual question answering (Asai et al., 2021), entity linking (Liu et al., 2021a), and cross-lingual summarization (Zhu et al., 2019; Vitiugin and Castillo, 2022). A truly multilingual search engine requires reliable estimation of both monolingual (for a wide range of languages) as well as cross-lingual query-document relevance, which both crucially rely on the alignment of text representations across different languages (Nie, 2010). The lack of large-scale retrieval datasets in languages other than English means that monolingual reranking for those languages has to be achieved by means of cross-lingual transfer of a reranking model trained on English relevance judgments.

Pretrained massively multilingual Transformers (MMTs) like multilingual BERT (mBERT) (Devlin et al., 2019) or XLM-R (Conneau et al., 2020) have been leveraged to this effect, but were shown to require substantial task-specific (i.e., ranking-

Proceedings of the 29th International Conference on Computational Linguistics, pages 1071–1082
October 12–17, 2022.

oriented) fine-tuning for reliable prediction of semantic similarity and relevance scores (Reimers and Gurevych, 2020; Litschko et al., 2021). MMTs offer zero-shot cross-lingual transfer of neural (re)ranking models out of the box – an MMT is fine-tuned on English relevance judgments and then employed in (monolingual or cross-lingual) retrieval tasks that involve other languages. Conceptually, via such transfer, no fine-tuning data (i.e., relevance judgments) is required for the target language(s).

This procedure, in principle, enables downstream zero-shot transfer to any language seen by the MMT in pretraining (e.g., for mBERT, 104 languages). However, in language understanding tasks (Hu et al., 2020), massive performance drops have been observed when transferring between distant languages, and especially in transfer to low-resource languages, underrepresented in MMT pretraining (Lauscher et al., 2020). Our results (§4) confirm these findings for ad-hoc IR. This is the consequence of the effect known as the *curse of multilinguality* (Conneau et al., 2020): sharing MMT parameters (i.e., its fixed parameter budget/capacity) across more and more languages makes text representations for individual languages less accurate. This effect is especially detrimental to low-resource languages, those least represented in multilingual pretraining corpora. What is more, large-scale full fine-tuning on the source language data (e.g., English) is likely to lead to catastrophic forgetting and interference effects (McCloskey and Cohen, 1989; Mirzadeh et al., 2020) that further bias the multilingual representation space towards the source language, at the expense of representation quality for low-resource languages. Besides the standard zero-shot cross-lingual transfer (MacAvaney et al., 2020; Huang et al., 2021a), other cross-lingual transfer approaches, commonly applied in other NLP tasks, such as training data translation (Shi et al., 2020), or leveraging external word-level alignments (Huang et al., 2021b), as well as distant supervision (Yu et al., 2021) have been explored as means to improve the cross-lingual transfer of neural rankers in IR. While translation-based approaches are competitive for high-resource languages, they may not be as effective for low-resource languages for which reliable MT models are missing; also, translation-based cross-lingual transfer has been shown to suffer from unwanted artifacts, such as "translationese" (Zhao et al., 2020; Vanmassenhove et al., 2021).

Contributions. Even if one would have sufficient amounts of labelled data in target languages, training language- or language-pair specific neural rerankers for all languages and language pairs would be prohibitively computationally expensive and unsustainable (Strubell et al., 2019). In this work we additionally remedy for this by composing (re)rankers in a modular way that enables more sustainable cross-lingual transfer. Concretely, we introduce neural (re)ranking models for cross-lingual and multilingual document retrieval based on MMTs that enable much more parameter efficient fine-tuning and more effective cross-lingual transfer for relevance prediction. Our (re)rankers are based on two styles of modular components: **1)** *Adapters* (Rebuffi et al., 2017; Houlsby et al., 2019; Pfeiffer et al., 2020) and **2)** *Sparse Fine-Tuning Masks* (SFTMs) (Ansell et al., 2022). When integrated into the architecture of a pretrained MMT, both allow for (1) the pretrained multilingual knowledge to be fully preserved, alleviating the negative interference and forgetting effects, and (2) offer additional language-specific model capacity which is used to improve the MMTs' representations for target languages, thus remedying for the curse of multilinguality.

We provide an extensive evaluation of both approaches in (i) zero-shot transfer for monolingual retrieval and (ii) CLIR, on two established benchmarks (Braschler, 2003; Lawrie et al., 2022). As an additional contribution, we expand the CLEF dataset (Braschler, 2003) with three query languages from the Turkic family (Turkish, Kyrgyz, and Uyghur, the latter two being low-resource languages), typologically and etymologically distant from the Indo-European languages.[1] Our results show that our modular neural (re)rankers are not only faster to train, but also outperform standard zero-shot transfer based on full MMT fine-tuning, and especially so in retrieval tasks that involve linguistically distant and low-resource languages. Moreover, our adapter- and SFTM-based rerankers generally outperform a strong preranker that utilizes state-of-the-art machine translation.

2 Methodology

We first introduce the general multi-stage ranking (i.e., preranking-reranking) framework, com-

[1] In this manner, our work addresses the calls for more linguistic diversity in NLP and IR research (Bender, 2011; Joshi et al., 2020; Ponti et al., 2020; Ruder et al., 2021).

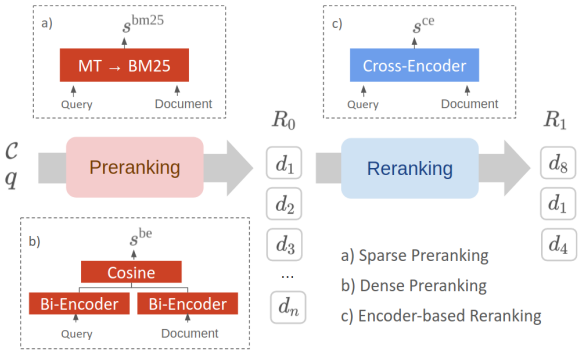

Figure 1: Overview of the multi-stage ranking approach to ad-hoc retrieval. **Stage 1 - Preranking:** We rank the document collection $\mathcal{C}$ by (a) running *sparse* BM25 retrieval on translated queries, or (b) according to the cosine similarity between *dense* query and document representations yielding an initial ranking R_0. **Stage 2 - Reranking:** We refine R_0 by reranking the top-k documents according to relevance scores predicted by a Cross-Encoder, yielding the refined ranking R_1.

monly used in information retrieval tasks, within which our work is embedded. We then introduce adapters and sparse fine-tuning masks (SFTMs), and present how to leverage them as crucial vehicles of the parameter-efficient cross-lingual transfer of the reranking component.

2.1 Multi-Stage Ranking

Pretrained Transformers like BERT (Devlin et al., 2019) are often used as *Cross-Encoder (CE)* scoring models: the Transformer encodes a query-document concatenation fed as input to the model, and the encoding is then fed to a dense layer that predicts the relevance score (MacAvaney et al., 2020; Jiang et al., 2020; Nogueira et al., 2019b). Computing scores for all query-documents pairs with Cross-Encoders is too slow for practical IR applications: they are thus primarily used as rerankers in a multi-stage ranking approach (MacAvaney et al., 2020; Geigle et al., 2021). In this work we adopt this paradigm for cross-lingual ad-hoc retrieval: Figure 1 illustrates its workflow.[2]

Preranking, based on a fast and efficient ranking method, is applied to every document from the document collection in order to provide a good initial ranking, targeting high recall. Let q^{l1} be a query in language l_1 and $\mathcal{C}^{l2} = \{d_i\}_{i=1}^{n}$ be a document collection containing n documents in language l_2. Associating and ranking documents w.r.t. relevance scores s_i we obtain an initial ranking

$$R_0 = [(d_1, s_1), (d_2, s_2) \dots (d_n, s_n)], \quad (1)$$

where $s_1 > s_2 > \dots s_n$. We transfer our rerankers based on MMTs – and trained on English relevance judgments – to (i) CLIR tasks as well as to (ii) monolingual IR tasks in target languages. The latter task, termed MoIR, is effectively zero-shot cross-lingual transfer for monolingual retrieval. In MoIR, we opt for a lexical preranker and score documents with $s^{\text{bm25}} = \text{BM25}(q, d)$.[3] In CLIR we follow the widely used approach of machine translating the query (Bonifacio et al., 2021; Lawrie et al., 2022): this process effectively translates CLIR into a noisy variant of MoIR. In addition, we experiment with a representation-based approach based on pretrained multilingual *Bi-Encoders* (*BE*): here, we embed the query and documents independently, and then use the cosine similarity between their embeddings $s^{be} = cos(BE(q), BE(d))$. In the preranking stage, unlike later in reranking, we use the encoders merely as general-purpose text encoders, without any additional retrieval-specific training.

Reranking: This stage refines the initial ranking obtained via preranking. It relies on a *CE* model which captures fine-grained (but more costly to model and run) semantic interactions between queries and documents. The ranking is then:

$$R_1 = [(d_1, s_1^{ce}), (d_2, s_2^{ce}) \dots (d_k, s_k^{ce})] \quad (2)$$

To this end, we rely on multilingual CEs to compute the binary relevance score s^{ce} on the concatenation of query and document pairs: $s^{ce} = CE(\texttt{[CLS]}\, q\, \texttt{[SEP]}\, d_i\, \texttt{[SEP]})$. We adopt a common practice (MacAvaney et al., 2019; Craswell et al., 2020; Naseri et al., 2021) of reranking the top $k = 100$ pre-ranked documents, yielding the final ranking R_1. Finally, it is also possible to ensemble the preranker's and reranker's ranked lists via simple rank averaging. In our experiments (4), we evaluate such preranking-reranking ensembles as well and show that such interpolations often bring additional performance gains.

[2] Alternative approaches that leverage pretrained encoders for IR include late interaction models (Khattab and Zaharia, 2020; Gao et al., 2021; Nair et al., 2022; Santhanam et al., 2022), embedding-based retrieval (Hofstätter et al., 2021; Litschko et al., 2021), and augmentation (Nogueira et al., 2019c,a).

[3] We used the `pyserini` implementation of BM25 (Lin et al., 2021) with the suggested (i.e., default) parameter configuration.

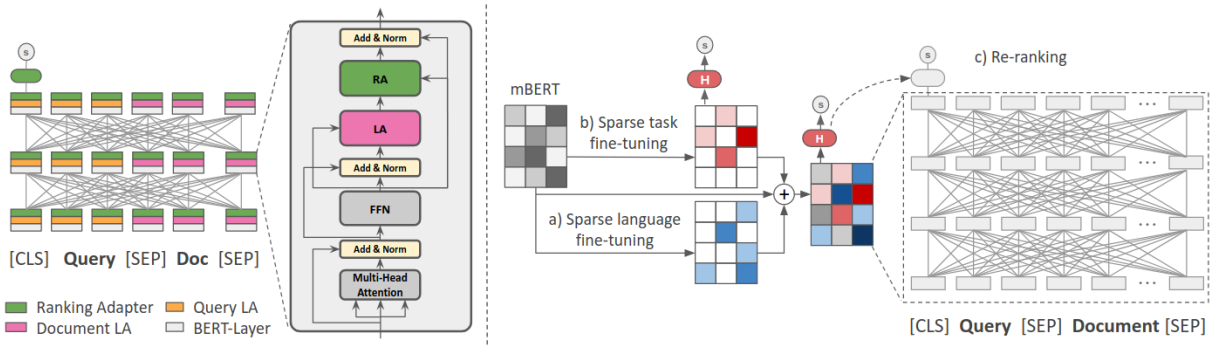

Figure 2: Overview of parameter-efficient transfer learning for neural (re)ranking. *Left*: A reranker is composed by stacking a pretrained target Language Adapter (LA) and a Ranking Adapter (RA; trained with source language data) on top of the original Transformer layers of an MMT (e.g., mBERT). *Right*: Sparse fine-tuning of a Ranking Mask (RM) and a Language Mask (LM) from mBERT parameters; rerankers are composed by adding the RM and LM values to the original mBERT parameters.

2.2 Parameter-Efficient Cross-Lingual Ranker Transfer

In this work, we propose a modular and parameter-efficient framework that allows faster training and more effective cross-lingual transfer of neural rerankers, that enhances both CLIR and MoIR. We first learn language-specific Adapters (LAs) or Sparse Fine-Tuning Masks (SFTMs) via Masked Language Modelling (MLM) on unannotated monolingual corpora of respective languages, while keeping the original MMT parameters intact. We then train Ranking Adapters (or Ranking SFTMs) using source-language data on top of the source-language LAs (language SFTMs), while keeping all other parameters frozen. At inference time, for a given IR (MoIR or CLIR) task, we compose our reranker by placing the Ranking Adapters (Ranking SFTMs) on top of the LAs (language SFTMs) of the query and/or document languages of that concrete retrieval task. The modular framework is illustrated in Figure 2.

Adapters. We train Ranking Adapters (RA) and Language Adapters (LA) based on the architecture of Pfeiffer et al. (2020). In the Transformer architecture, each layer l consists of a multi-head attention block (i.e., sub-layer) and a feed-forward network (FFN), both followed by a residual connection and layer normalization. We denote the residual connection (output of FFN) with $\mathbf{r}_l$ and the hidden state after the layer norm with $\mathbf{h}_l$.

$$\text{LA}(\mathbf{h}_l, \mathbf{r}_l) = \mathbf{U}_l(\psi(\mathbf{D}_l(\mathbf{h}_l)) + \mathbf{r}_l \quad (3)$$

$$\text{RA}(\mathbf{h}_l, \mathbf{r}_l) = \mathbf{U}_l(\psi(\mathbf{D}_l(\text{LA}_l))) + \mathbf{r}_l \quad (4)$$

Adapters are parameterized by the down-projection matrix $\mathbf{D} \in \mathbb{R}^{h \times d}$ and the up-projection matrix $\mathbf{U} \in \mathbb{R}^{d \times h}$, where h and d denote the hidden size of the Transformer and the bottleneck dimension of the adapter, respectively. The ratio between h and d is also called the *reduction factor*, and corresponds to the level of parameter compression (i.e., how many times fewer parameters are updated if we train adapters instead of updating all Transformer parameters). The forward pass of a Language Adapter consists of a down-projection of h_l, a non-linear activation function $\psi(\cdot)$ and an up-projection. Ranking Adapters are stacked on top of LAs and process their output. Both adapters have residual connections to the output of the FFN.[4] We train LAs using the standard MLM objective (Devlin et al., 2019), whereas we train RAs together with the dense scoring layer by means of minimizing the standard binary cross-entropy loss.

In CLIR setups, queries and documents are in different languages. It is thus, in principle, possible to stack the RA on top of (i) the query language adapter LA^{Q}, (ii) document language adapter LA^{D}, or by using (iii) *split adapters* LA^{S}: here, we encode query tokens up to the separator token (`[SEP]`) using the LA of the query language and the document tokens (after `[SEP]`) with the LA of language of the document collection (cf. Fig. 2).

Sparse Fine-Tuning Masks. Like adapters, SFTMs (Ansell et al., 2022) aim to decouple task knowledge from language knowledge, but instead of introducing additional parameters, the idea is

[4] To alleviate the mismatch between the multilingual vocabulary of the MMT and the target language vocabulary, Pfeiffer et al. (2020) also additionally place invertible adapters INV on top of the embedding layer along with their inverses INV^{-1} placed before the output layer. In our experiments we adopt this variant; for more details we refer the reader to the work of Pfeiffer et al. (2020).

to directly update only small subsets of MMT's original parameters. Sparse Fine-Tuning (SFT) consists of two phases. In *Phase 1* we fine-tune all mBERT's parameters $\theta^{(0)}$, resulting in updated parameter values $\theta^{(1)}$. We then select the top K parameters with the largest value change, i.e., those with the largest values $|\theta_i^{(0)} - \theta_i^{(1)}|$. We then construct a binary mask: the selected K parameters remain trainable, whereas all other parameters are frozen. In *Phase 2* all parameters are reset to $\theta^{(0)}$ and training restarts, but this time only the selected parameters of the mask are updated, yielding $\theta^{(2)}$. The final update (i.e., the SFTM) is then obtained as the difference vector $\mathrm{M} = \theta^{(2)} - \theta^{(0)}$. As is the case with Language Adapters, we obtain the Language Masks (LM) by means of (additional) MLM training on language-specific corpora; whereas the Ranking Mask (i.e., the mask for the ranking task, RM) is learned via binary cross-entropy objective on source-language (English) relevance judgments. At inference, the reranker is composed as $\theta^{(0)} + \mathrm{RM} + \mathrm{LM}$ (cf., Figure 2). In our CLIR settings (§3), we explore using (i) the query language mask (LM^{Q}), (ii) document language mask (LM^{D}) or (iii) the combination of both masks ($\mathrm{LM}^{\mathrm{B}} = \mathrm{LM}^{\mathrm{Q}} + \mathrm{LM}^{\mathrm{D}}$). Note that SFTMs represent a more computationally efficient solutions at inference time: unlike adapters, they do not extend (i.e., deepen) the Transformer architecture.

3 Experimental Setup

Adapter and SFTM Training. We train adapters following the recommendations from Pfeiffer et al. (2020). Unless noted otherwise, we train LAs with the reduction factor of 2 (i.e., $h/d = 2$) on Wikipedias of respective languages, for 250K steps with batch size 64 and learning rate of 1e-4. For RAs we experimented with the different reduction factors: 1, 2, 4, 8, 16, 32 (cf. §4). Following Ansell et al. (2022), for fair comparisons between adapters and SFTMs, we set the mask size K for SFTMs to the same number of parameters that adapters with a certain reduction factor have.[5]

Reranking Training. We train mBERT-based[6] rerankers on MS-MARCO (Craswell et al., 2021), with a linear warm-up over the first 5K updates, in batches of 32 instances with a maximum sequence length of 512, and using a learning rate of 2e-5. We evaluate the model on the validation data every 25K updates and choose the checkpoint with the best validation performance.

Evaluation Data. We evaluate the models on the standard CLEF-2003 benchmark (Braschler, 2003)[7] as well as on the recently introduced HC4 benchmark (Lawrie et al., 2022). With CLEF, we use monolingual test collections in EN, DE, IT, RU, and FI for MoIR, and experiment with the following cross-lingual directions: EN-{FI, DE, IT, RU}, DE-{FI, IT, RU}, FI-{IT, RU}. Each experimental run covers 60 queries, whereas the document collection sizes are as follows: RU – 17K, FI – 55K, IT – 158K, and DE – 295K.

We additionally evaluate the models in CLIR tasks with CLEF queries posed in lower-resource languages. To this end, (i) we leverage Swahili (SW) and Somali (SO) queries (Bonab et al., 2019), where the queries were obtained via manual translation of English queries; (ii) we create another set of translated CLEF queries in three languages: Turkish (TR), Kyrgyz (KG), and Uyghur (UG). The new set covers one high-resource and two low-resource languages and is intended to facilitate and diversify evaluation of CLIR with low-resource languages in future work. The queries were constructed via the standard post-editing procedure borrowed from other data collection tasks (Glavaš et al., 2020; Hung et al., 2022): we obtained initial query translations via Google Translate, which were then post-edited by native speakers.

HC4 comprises queries *and* document collections in three languages: Persian (FA), Russian (RU) and Chinese (ZH). Compared to CLEF, HC4 collections are considerably larger, spanning 646K, 486K and 4.72M documents per each respective language, associated with 50 test queries in each language. We use *title* and *description* fields as queries following Lawrie et al. (2022). HC4 is used in MoIR experiments.

Baseline Models. The primary baseline for our adapter- and SFTM-based transfer is the standard and well established method for zero-shot transfer of English-trained rerankers (MacAvaney et al., 2020), termed *MonoBERT*. This is the reranking Cross-Encoder where we allow for full-tuning of the underlying monolingual or multilingual BERT

[5] Leading to the number of trainable parameters (sparsity) of 14M (8.5%), 7.1M (4.2%), 3.6M (2.1%), 1.8M (1.1%), 894K (0.52%) and 452K (0.27%) respectively.

[6] Pretrained `bert-base-multilingual-uncased` weights from the HuggingFace Transformers library (Wolf et al., 2020) are used.

[7] `http://catalog.elra.info/en-us/repository/browse/ELRA-E0008/`

Model	TR-EN	TR-IT	TR-DE	TR-FI	TR-RU	EN-FI	EN-IT	EN-RU	EN-DE	DE-FI	DE-IT	DE-RU	FI-IT	FI-RU	**AVG**	**ENS**
$\text{DISTIL}_{\text{DmBERT}}$ (PR)	.183	.251	.190	.252	.260	.294	.290	.313	.247	.300	.267	.284	.221	.302	.261	-
MonoBERT	.235	.197	.208	.285	.217	.339	.315	.248	.295	.329	.270	.246	.197	.174	.254	.274
+RA $+\text{LA}^{\text{S}}$	.269	**.253**	**.252***	**.362**	.186	.363	.352	.197	.317*	.329	.300	.223	**.266**	.207	.277	.287
+RA $+\text{LA}^{\text{D}}$	.252	.234	.222	.267	**.267**	.366*	**.366***	**.248**	.314*	.350	.302	**.315**	.220	**.234**	**.283**	**.298**
+RA $+\text{LA}^{\text{Q}}$	**.270**	.243	.242	.293	.191	.370	.355	.189	.318	.325	.279	.223	.247	.182	.266	.285
+RM $+\text{LM}^{\text{B}}$	.229	.228	.197	.244*	.168	.299	.344	.181*	.303	.309	.302	.191*	.206	.108*	.236	.269
+RM $+\text{LM}^{\text{D}}$	.231	.226	.229	.317	.149*	**.394***	.359	.173*	.320*	.376	.304	.187	.239	.166*	.262	.279
+RM $+\text{LM}^{\text{Q}}$	.239	.252	.232	.316	.162*	.359	.349	.191	.310*	**.391**	**.323***	.195	.255*	.160	.267	.280

Table 1: CLIR results (Mean Average Precision, MAP) with $\text{DIST}_{\text{DmBERT}}$ as Stage 1 preranker. **Bold**: Best neural retrieval model for each language pair. *: significance tested against MonoBERT at $p \leq 0.05$, computed via paired two-tailed t-test. Ranking and Language Adapters have a reduction factor of 16 and 2 (see §2), respectively. Ranking and Language Masks both correspond to a reduction factor of 2 (see §3). We report average results (AVG), and also averaged ensemble (ENS) results where we combine ranking lists from Stage 1 and Stage 2 rankings; see §2.1. Superscripts over LAs and LMs denote query language (Q), document language (D), split adapters (S) for LAs, and '(B)oth masks' for LMs (see §2.2).

Model	TR-EN	TR-IT	TR-DE	TR-FI	TR-RU	EN-FI	EN-IT	EN-RU	EN-DE	DE-FI	DE-IT	DE-RU	FI-IT	FI-RU	**AVG**	**ENS**
NMT+BM25 (PR)	.392	.353	.308	.307	.227	.378	.446	.285	.355	.367	.385	.272	.364	.271	.336	-
MonoBERT	.415	.375	.339	.345	.307	.386	.411	.351	.371	.409	.380	.322	.367	.340	.366	.360
+RA +LA	**.448**	.408*	.353	.371	.327	.388	**.435**	**.367**	.385	.413	.405	.348	.381	**.365**	.385	**.374**
+RM +LM	.447	**.414**	**.356**	**.386**	**.336**	**.413**	.429	.345	**.390**	**.468**	**.407**	**.363**	**.395**	.364	**.394**	.371

Table 2: CLIR results (Mean Average Precision, MAP) with NMT+BM25 as Stage 1 preranker. For modular rerankers, we report the numbers with the best-performing configurations from CLEF experiments: +RA $+\text{LA}^{\text{D}}$ and +RM $+\text{LM}^{\text{Q}}$; see also the caption of Table 1.

model on MS-MARCO. For CLIR experiments, we opt for $\text{DISTIL}_{\text{DmBERT}}$ as our Bi-Encoder preranker (PR), as it showed strong performance in our recent comparative empirical study (Litschko et al., 2021). In brief, $\text{DISTIL}_{\text{DmBERT}}$ is trained via knowledge distillation where sentence-similarity features are distilled from a monolingual English teacher, specialized for semantic encoding of sentences, into a multilingual student model; see (Reimers and Gurevych, 2020) for further details.

Finally, also for CLIR, we couple a state-of-the-art NMT system of Fan et al. (2020) (FAIR-MT), which we use to translate queries to the document collection language, with the BM25 ranker in the target language. For Kyrgyz and Uyghur, we use another NMT model, provided by the Turkic Interlingua (TIL) community[8] (Mirzakhalov et al., 2021), because we failed to obtain meaningful {KG, UG}$\rightarrow l_2$ translations with FAIR-MT.

4 Results and Discussion

Cross-Lingual Retrieval (CLIR). Tables 1 and 2 show the CLIR results, for fourteen language pairs from the augmented CLEF 2003 benchmark[9] using $\text{DISTIL}_{\text{DmBERT}}$ and NMT+BM25 as Stage 1 prerankers, respectively. With $\text{DISTIL}_{\text{DmBERT}}$ as the preranker (Table 1), Adapter- and SFTM-based rerankers consistently improve the initial preranking results, with gains of up to 2.7 MAP points, and EN-RU as the only exception. Importantly, compared to full fine-tuning (MonoBERT), our modular reranking variants bring gains between 1 and 4 MAP points on average, across all language pairs. Interestingly, the best adapter configuration (RA $+\text{LA}^{\text{D}}$), where at inference we stack the RA on top of the LA of the document collection language) outperforms the best SFTM-based reranker (RM $+\text{LM}^{\text{Q}}$ and RM $+\text{LM}^{\text{D}}$) by 1.6 MAP points. Somewhat surprisingly, adapting only to the language of the document collection (LA^{D}; LM^{D}) yields better performance than adapting to both the query and collection language of the target task (LA^{S}; LM^{B}).

The language pairs in Tables 1 and 2 consist of high-resource languages for which large parallel corpora and, consequently, reliable NMT models exist. However, even when starting from a more

[8] https://turkic-interlingua.org

[9] We add TR-* pairs to the evaluation, enabled by our EN→TR translations of the queries.

Model	CLEF 2003				HC4				
	SW–EN	SO–EN	KG–EN	UG–EN	EN–FA	EN–ZH	EN–RU	**AVG**	**ENS**
NMT+BM25 (PR)	.325	.157	.228	.091	.183	.113	.186	.183	-
MonoBERT	.362	.158	.255	.157	.246	.172	.218	.224	.216
+RA + LA^{D}	**.407**	**.166**	.305	.155	.259	.189	.234	.245	**.228**
+RM + LM^{D}	.389	.161	**.311**	**.165**	**.267**	**.196**	**.241**	**.247**	.225

Table 3: CLIR results on extended CLEF pairs with low-resource query languages (Swahili, Somali, Kyrgyz, and Uyghur) and three language pairs from the HC4 benchmark.

competitive MT-based preranker (NMT+BM25; Table 2), our modular cross-lingual transfer of the reranker yields performance gains. In fact, with this stronger preranker, the gains from modular reranking are even more pronounced: +5/+6 MAP points for Adapters and SFTMs, respectively, compared to preranker and +2/+3 MAP points, respectively, compared to MonoBERT. This could explain why interpolating between the preranking and reranking (ENS, last column) yields further gains with $\text{DISTIL}_{\text{DmBERT}}$ as the preranker (Table 1), but not when we prerank with NMT+BM25 (Table 2).

Table 3 shows CLIR results for (a) language pairs from extended CLEF with queries written in low-resource languages – Swahili and Somali queries created by Bonab et al. (2019), as well as Kyrgyz and Uyghur queries that we created; and (b) three cross-lingual pairs of arguably distant languages (EN-{Farsi, Chinese, Russian}) from the HC4 benchmark. The gains that our SFTM- and Adapter-based modular rerankers bring for language pairs involving low-resource languages, over the MT-based preranker and the full fine-tuning (MonoBERT), are generally more substantial than those for high-resource language pairs: e.g., +8 and +4 MAP points w.r.t. NMT+BM25 and MonoBERT, respectively for SW-EN, and +8 and +5 points for KG-EN. The gains are similarly prominent for more distant language pairs from the HC4 dataset (+8 MAP points over the NMT+BM25 preranker for EN-FA and EN-ZH). With such prominent gains of the modular reranking over the preranker, it is no surprise that averaging the preranking and reranking document ranks (ENS) reduces the performance of the reranker. We believe that these results in particular emphasize the effectiveness of modular cross-lingual transfer that allows to increase the capacity of MMTs for individual languages, by means of LMs or LAs. The representations of low-resource languages, for which MMTs have seen little data in pretraining, particularly suffer from the curse of multilinguality (Conneau et al., 2020; Lauscher et al., 2020) – this is why particularly prominent gains are achieved for those languages when we increase the MMTs capacity for their representation via LMs/LAs.

Cross-Lingual Transfer for MoIR. Table 4 displays the results of monolingual retrieval with our best-performing modular rerankers for EN (as the source language) and four target languages (DE, IT, FI, RU).[10] Unlike the fully fine-tuned reranker (MonoBERT), our modular Adapter- and SFTM-based rerankers improve the initial rankings produced by BM25. These results strengthen the finding that our modular rerankers are not just more parameter-efficient (i.e., faster to train), but also lead to better cross-lingual transfer due to decoupling of language- and ranking-specific knowledge. In MoIR tasks the SFTM-based transfer outperforms its Adapter-based counterpart, same as in the case of CLIR with NMT+BM25 preranking (Table 1). Also as in the case of the latter CLIR results (Tables 1 and 3), interpolating between preranking and reranking results does not bring any gains.

It is worth noting that all MoIR scores are substantially higher than CLIR results from Tables 1 and 2. This is expected and reflects the fact that matching representations within a language – where models can still rely on exact lexical matches between queries and documents – is easier than aligning text representations across languages.

Effectiveness vs Efficiency. Adapters increase query latency because they deepen the Transformer. Rücklé et al. (2021) show that one can drop adapters from lower layers with small-to-negligible effect on performance. Table 5 shows the results of a similar analysis, where we drop the adapters from the first N layers at inference. Dropping adapters from only the first two layers (row 1-2) only slightly decreases the MoIR performance whereas it even slightly increases the

[10]Note that in MoIR, the actual reranking is always monolingual (albeit in the target language). Both queries and documents are thus encoded with the same target language LA/LM.

Model	CLEF 2003					HC4			AVG	ENS
	EN	FI	DE	IT	RU	FA	ZH	RU		
BM25 (PR)	.480	.505	.434	.494	.361	.279	.196	.228	.372	-
MonoBERT	.464	.528	.444	.463	.363	.356	.283	.245	.398	.402
+RA + LA	.512	.537	.457	.495	**.389**	.372	.284	.261	.413	.410
+RM + LM	**.515**	**.564**	**.459**	**.502**	.379	**.398**	**.307**	**.264**	**.423**	**.417**

Table 4: Results of zero-shot cross-lingual transfer for monolingual retrieval (MoIR) on CLEF 2003 and HC4 datasets. Results with reduction factors of 16 and 2 for Adapters and SFTMs, respectively.

Layer	CLIR	MoIR	AVG	Latency	Δ Speed-Up	Δ MAP
None	.282	.418	.331	34.6 ms	-	-
1-2	.295	.412	.337	33.7 ms	+2.6%	+.006
1-4	.269	.395	.314	32.8 ms	+5.0%	−.017
1-6	.229	.375	.281	31.9 ms	+7.7%	−.050
1-8	.134	.284	.187	31.0 ms	+10.4%	−.143
1-10	.086	.210	.130	30.0 ms	+12.9%	−.200
1-12	.086	.208	.129	29.5 ms	+14.2%	−.201

Table 5: Trade-off between efficiency and effectiveness when dropping adapters in +RA + LA^D. Average over all CLIR/MoIR setups and all reduction factors.

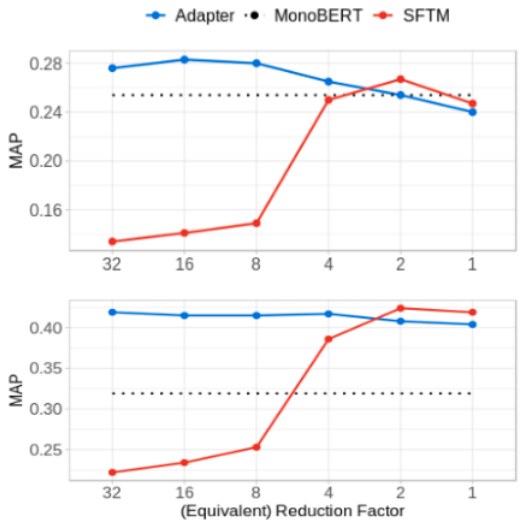

Figure 3: Retrieval performance at different parameter reduction factors; average MAP performance for CLIR (top) and MoIR (bottom).

CLIR results. Dropping adapters from more layers, however, substantially reduces the retrieval performance: e.g., removing adapters from the first 10 layers reduces CLIR performance by almost 20 MAP points, while reducing the query latency by only 13%. While Adapters and SFTMs yield comparable performance in our experiments, these observations favor SFTMs: for the *same query latency*,[11] SFTMs will yield better performance.

Parameter Efficiency. We also investigate the relation between various levels of parameter efficiency and retrieval performance. Figure 3 shows the performance of our modular rerankers for different parameter reduction factors. SFTMs exhibit stronger performance with smaller reduction factors (2 and 4), i.e., when we update a larger percentage of mBERT's original parameters. SFTMs shift the pretrained values of mBERT's parameters: this constrains the range of values that individual parameters can take, requiring the modification of the larger number of parameters for injecting complex language- and ranking-specific knowledge. In contrast, Adapters show better performance with higher reduction factor (8, 16, 32), i.e., when we add a relatively smaller number of Adapter parameters. This could be the consequence of the "unconstrained" initialization of the new Adapter parameters, which allows the complementary language- and ranking-specific knowledge to be compressed into a smaller number of parameters. Comparing those effects between CLIR and MoIR we observe the same trends. However, MAP gains compared to MonoBERT are larger in MoIR than in CLIR. This seems intuitive as ranking adapters (masks) are able to adapt for exact matches.

Impact of NMT on CLIR. In the cross-lingual setup the quality of retrieved documents crucially depends on the quality of query translations when NMT is used. In Table 6 we show original English queries together with their respective translations from Swahili and Somali. As expected, translations from Swahili are generally of higher quality compared to Somali, which explains the big performance gap reported in Table 3. In the best case the translation is semantically very close to the original query (cf., SW→EN; QID:172), or it contains only slight lexical (*flooding* vs. *floods*) and semantic variations, e.g., near-synonyms (*Holland* vs. *Netherlands*). In other cases, error prop-

[11] The query latency of an SFTM-based reranker is the same as that of MonoBERT as SFTMs do not increase the number of layers (nor parameters within layers) of the MMT.

QID	English Query *(original)*	NMT: Swahili → English	NMT: Somali → English
151	Wonders of **Ancient World** Look for **information** on the **existence** and/or the discovery of remains of the seven wonders of the **ancient world**.	Search for **information** about the **existence** and/or development of the **seventh** universe of the **ancient world**.	Thus, therefore, it is necessary to bear in mind that the truth is the truth, and that the truth is the truth, and that the truth is the truth.
172	**1995** Athletics **World Records** What **new world records** were achieved during the **1995** athletic world championships in **Gothenburg**?	What **new world records** were recorded at the **1995 World** Horses in **Gothenburg**?	The **1995 World** Trade Organization (WTO) announced that a **new** international trade agreement has led to a global trade agreement in **Gothenburg**.
187	**Nuclear** Transport in **Germany** Find reports on the protests against the transportation of **radioactive** waste with **Castor containers** in **Germany**.	**Nuclear** Delivery in **Germany** A report on the anti-trafficking of **radioactive** pollutants and **Castor containers** in **Germany**.	The Nugleerka department of Jarmalka Hel has been prepared for the development of the Nugleerka department of **Castor** district in Jarmalka.
200	Flooding in Holland and **Germany** Find statistics on flood disasters in Holland and **Germany** in **1995**.	The floods in the Netherlands and **Germany** have recorded the floods in the Netherlands and **Germany** in **1995**.	The Netherlands Federation and the United Nations have agreed with the Netherlands Federation and the Netherlands Federation in **1995**.

Table 6: Comparison between original CLEF queries and translations from Swahili and Somali to English. Tokens that occur both in the original query and translations are highlighted in **bold** (ignoring case, excluding stopwords).

agation from NMT impacts CLIR performance to different extents. Those include, e.g., missing keywords (*statistics*; QID:200), topic shifts (*sports* vs. *business*; SO→EN, QID:172) or queries consisting of unrelated text and repetitions (i.e., 'hallucinations'; SO→EN, QID:151, QID:200). Especially repetitions and hallucinations[12] are known unwanted artifacts in NMT (Fu et al., 2021; Raunak et al., 2021) and can cause retrieval models to emphasize unrelated keywords by inflating their term frequency.[13] Lastly, in cases where source words are copied instead of translated, e.g., *Nugleerka (Nuclear)* or *Jarmalka (Germany)* in QID:187, neural retrieval models need to rely on imperfect internal alignment of word translations (Cao et al., 2019).

5 Related Work

Next to Adapters and SFTMs there exist other parameter efficient transfer (PET) methods. For example, BitFit trains only bias vectors (Ben Zaken et al., 2022), LoRa trains low-rank decompositions of weight matrices in dense layers (Hu et al., 2022) and methods that learn continuous prompts (Liu et al., 2021b; Lester et al., 2021; Li and Liang, 2021, *inter alia*). In the context of retrieval for English, concurrent work focuses on the learning-efficiency (Ma et al., 2022) and out-of-domain generalization (Tam et al., 2022) of PET methods, whereas we investigate PET both on task- and language-level adaption for CLIR.

6 Conclusion

In this work, we introduced modular and parameter-efficient neural rerankers for effective cross-lingual retrieval transfer. Our models, based on Adapters and Sparse Fine-Tuning Masks, allow for decoupling of language-specific and task-specific (i.e., ranking) knowledge. We demonstrate that this leads to more effective transfer to cross-lingual IR setups as well as to better cross-lingual transfer for monolingual retrieval in target languages with no relevance judgment improving over strong prerankers based on state-of-the-art NMT. Encouragingly, we observe particularly pronounced gains for low-resource languages included in our evaluation. We hope that our results will encourage a broader investigation of parameter-efficient neural retrieval in monolingual and cross-lingual setups. We make our code and resources available at: `https://github.com/rlitschk/ModularCLIR`.

Acknowledgments

RL and GG are supported by the EUINACTION grant from NORFACE Governance (462-19-010, GL950/2-1). IV work is supported by a Huawei research donation to the University of Cambridge.

[12]This phenomenon has been reported to occur in low-resource and out-of-domain settings (Müller et al., 2020). We confirm this finding as we find hallucinations appearing more often in EN→SO than in EN→SW query translations.

[13]Further investigation of NMT+BM25 on SO→EN reveals that manually filtering out queries containing more than two repetitions/hallucinations leaves us with 22 remaining queries on which results improve from 0.157 to 0.280 MAP.

References

Alan Ansell, Edoardo Maria Ponti, Anna Korhonen, and Ivan Vulić. 2022. Composable sparse fine-tuning for cross-lingual transfer. In *Proceedings of ACL 2022*, pages 1778–1796.

Akari Asai, Jungo Kasai, Jonathan Clark, Kenton Lee, Eunsol Choi, and Hannaneh Hajishirzi. 2021. XOR QA: Cross-lingual open-retrieval question answering. In *Proceedings of NAACL*, pages 547–564.

Payal Bajaj, Daniel Campos, Nick Craswell, Li Deng, Jianfeng Gao, Xiaodong Liu, Rangan Majumder, Andrew McNamara, Bhaskar Mitra, Tri Nguyen, et al. 2016. Ms marco: A human generated machine reading comprehension dataset. *arXiv preprint arXiv:1611.09268*.

Elad Ben Zaken, Yoav Goldberg, and Shauli Ravfogel. 2022. BitFit: Simple parameter-efficient fine-tuning for transformer-based masked language-models. In *Proceedings of ACL*, pages 1–9.

Emily M Bender. 2011. On achieving and evaluating language-independence in nlp. *Linguistic Issues in Language Technology*, 6.

Hamed Bonab, James Allan, and Ramesh Sitaraman. 2019. Simulating clir translation resource scarcity using high-resource languages. In *Proceedings of ICTIR*, page 129–136.

Luiz Henrique Bonifacio, Israel Campiotti, Vitor Jeronymo, Roberto Lotufo, and Rodrigo Nogueira. 2021. mmarco: A multilingual version of the ms marco passage ranking dataset. *arXiv preprint arXiv:2108.13897*.

Martin Braschler. 2003. CLEF 2003–Overview of results. In *Workshop of the Cross-Language Evaluation Forum for European Languages*, pages 44–63.

Steven Cao, Nikita Kitaev, and Dan Klein. 2019. Multilingual alignment of contextual word representations. In *Proceedings of ICLR*.

Francisco Casacuberta, Alexandru Ceausu, Khalid Choukri, Miltos Deligiannis, Miguel Domingo, Mercedes Garcıa-Martınez, Manuel Herranz, Vassilis Papavassiliou, Stelios Piperidis, Prokopis Prokopidis, et al. 2021. The covid-19 mlia@eval initiative: Overview of the machine translation task.

Alexis Conneau, Kartikay Khandelwal, Naman Goyal, Vishrav Chaudhary, Guillaume Wenzek, Francisco Guzmán, Édouard Grave, Myle Ott, Luke Zettlemoyer, and Veselin Stoyanov. 2020. Unsupervised cross-lingual representation learning at scale. In *Proceedings of ACL*, pages 8440–8451.

Nick Craswell, Bhaskar Mitra, Emine Yilmaz, Daniel Campos, and Jimmy Lin. 2021. Ms marco: Benchmarking ranking models in the large-data regime. In *Proceedings of SIGIR*, pages 1566–1576.

Nick Craswell, Bhaskar Mitra, Emine Yilmaz, Daniel Campos, and Ellen M Voorhees. 2020. Overview of the trec 2019 deep learning track. *arXiv preprint arXiv:2003.07820*.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of NAACL*, pages 4171–4186.

Laura Dietz, Manisha Verma, Filip Radlinski, and Nick Craswell. 2017. Trec complex answer retrieval overview. In *TREC*.

Angela Fan, Shruti Bhosale, Holger Schwenk, Zhiyi Ma, Ahmed El-Kishky, Siddharth Goyal, Mandeep Baines, Onur Celebi, Guillaume Wenzek, Vishrav Chaudhary, Naman Goyal, Tom Birch, Vitaliy Liptchinsky, Sergey Edunov, Edouard Grave, Michael Auli, and Armand Joulin. 2020. Beyond english-centric multilingual machine translation. *arXiv preprint*.

Zihao Fu, Wai Lam, Anthony Man-Cho So, and Bei Shi. 2021. A theoretical analysis of the repetition problem in text generation. *Proceedings of AAAI*, 35(14):12848–12856.

Luyu Gao, Zhuyun Dai, and Jamie Callan. 2021. COIL: Revisit exact lexical match in information retrieval with contextualized inverted list. In *Proceedings of NAACL*, pages 3030–3042.

Gregor Geigle, Jonas Pfeiffer, Nils Reimers, Ivan Vulić, and Iryna Gurevych. 2021. Retrieve fast, rerank smart: Cooperative and joint approaches for improved cross-modal retrieval. *CoRR*, abs/2103.11920.

Goran Glavaš, Mladen Karan, and Ivan Vulić. 2020. Xhate-999: Analyzing and detecting abusive language across domains and languages. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 6350–6365.

Sebastian Hofstätter, Sheng-Chieh Lin, Jheng-Hong Yang, Jimmy Lin, and Allan Hanbury. 2021. Efficiently teaching an effective dense retriever with balanced topic aware sampling. In *Proceedings of SIGIR*, page 113–122.

Neil Houlsby, Andrei Giurgiu, Stanislaw Jastrzebski, Bruna Morrone, Quentin De Laroussilhe, Andrea Gesmundo, Mona Attariyan, and Sylvain Gelly. 2019. Parameter-efficient transfer learning for nlp. In *Proceedings of ICML*, pages 2790–2799. PMLR.

Edward J Hu, yelong shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. 2022. LoRA: Low-rank adaptation of large language models. In *Proceedings of ICLR*.

Junjie Hu, Sebastian Ruder, Aditya Siddhant, Graham Neubig, Orhan Firat, and Melvin Johnson.

2020. XTREME: A massively multilingual multi-task benchmark for evaluating cross-lingual generalisation. In *Proceeding of ICML*, pages 4411–4421. PMLR.

Kuan-Hao Huang, Wasi Ahmad, Nanyun Peng, and Kai-Wei Chang. 2021a. Improving zero-shot cross-lingual transfer learning via robust training. In *Proceedings of EMNLP*, pages 1684–1697.

Zhiqi Huang, Hamed Bonab, Sheikh Muhammad Sarwar, Razieh Rahimi, and James Allan. 2021b. Mixed attention transformer for leveraging word-level knowledge to neural cross-lingual information retrieval. In *Proceedings of CIKM*, pages 760–770.

Chia-Chien Hung, Anne Lauscher, Ivan Vulić, Simone Ponzetto, and Goran Glavaš. 2022. Multi2WOZ: A robust multilingual dataset and conversational pre-training for task-oriented dialog. In *Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 3687–3703, Seattle, United States. Association for Computational Linguistics.

Zhuolin Jiang, Amro El-Jaroudi, William Hartmann, Damianos Karakos, and Lingjun Zhao. 2020. Cross-lingual information retrieval with BERT. In *Proceedings of LREC*, page 26.

Pratik Joshi, Sebastin Santy, Amar Budhiraja, Kalika Bali, and Monojit Choudhury. 2020. The state and fate of linguistic diversity and inclusion in the nlp world. In *Proceedings of ACL*, pages 6282–6293.

Omar Khattab and Matei Zaharia. 2020. Colbert: Efficient and effective passage search via contextualized late interaction over bert. In *Proceedings of SIGIR*, pages 39—48.

Anne Lauscher, Vinit Ravishankar, Ivan Vulić, and Goran Glavaš. 2020. From zero to hero: On the limitations of zero-shot language transfer with multilingual transformers. In *Proceedings of EMNLP*, pages 4483–4499.

Dawn Lawrie, James Mayfield, Douglas W. Oard, and Eugene Yang. 2022. Hc4: A new suite of test collections for ad hoc clir. In *Proceedigs of ECIR*, page 351–366. Springer-Verlag.

Brian Lester, Rami Al-Rfou, and Noah Constant. 2021. The power of scale for parameter-efficient prompt tuning. In *Proceedings of EMNLP*, pages 3045–3059.

Xiang Lisa Li and Percy Liang. 2021. Prefix-tuning: Optimizing continuous prompts for generation. In *Proceedings of ACL*, pages 4582–4597.

Jimmy Lin, Xueguang Ma, Sheng-Chieh Lin, Jheng-Hong Yang, Ronak Pradeep, and Rodrigo Nogueira. 2021. Pyserini: A python toolkit for reproducible information retrieval research with sparse and dense representations. In *Proceedings of SIGIR*, pages 2356–2362.

Robert Litschko, Ivan Vulić, Simone Paolo Ponzetto, and Goran Glavaš. 2021. Evaluating multilingual text encoders for unsupervised cross-lingual retrieval. In *Proceedings of ECIR*, pages 342–358.

Qian Liu, Xiubo Geng, Jie Lu, and Daxin Jiang. 2021a. Pivot-based candidate retrieval for cross-lingual entity linking. In *Proceedings of WWW*, page 1076–1085.

Xiao Liu, Kaixuan Ji, Yicheng Fu, Zhengxiao Du, Zhilin Yang, and Jie Tang. 2021b. P-tuning v2: Prompt tuning can be comparable to fine-tuning universally across scales and tasks. *arXiv preprint arXiv:2110.07602*.

Xinyu Ma, Jiafeng Guo, Ruqing Zhang, Yixing Fan, and Xueqi Cheng. 2022. Scattered or connected? an optimized parameter-efficient tuning approach for information retrieval. *arXiv preprint arXiv:2208.09847*.

Sean MacAvaney, Luca Soldaini, and Nazli Goharian. 2020. Teaching a new dog old tricks: Resurrecting multilingual retrieval using zero-shot learning. In *Proceedings of ECIR*, pages 246–254.

Sean MacAvaney, Andrew Yates, Arman Cohan, and Nazli Goharian. 2019. Cedr: Contextualized embeddings for document ranking. In *Proceedings of SIGIR*, page 1101–1104.

Michael McCloskey and Neal J Cohen. 1989. Catastrophic interference in connectionist networks: The sequential learning problem. In *Psychology of learning and motivation*, volume 24, pages 109–165. Elsevier.

Seyed Iman Mirzadeh, Mehrdad Farajtabar, Razvan Pascanu, and Hassan Ghasemzadeh. 2020. Understanding the role of training regimes in continual learning. In *Proceedings of NeurIPS*, volume 33, pages 7308–7320.

Jamshidbek Mirzakhalov, Anoop Babu, Duygu Ataman, Sherzod Kariev, Francis Tyers, Otabek Abduraufov, Mammad Hajili, Sardana Ivanova, Abror Khaytbaev, Antonio Laverghetta Jr, et al. 2021. A large-scale study of machine translation in turkic languages. In *Proceedings of EMNLP*, pages 5876–5890.

Mathias Müller, Annette Rios, and Rico Sennrich. 2020. Domain robustness in neural machine translation. In *Proceedings of the 14th Conference of the Association for Machine Translation in the Americas (Volume 1: Research Track)*, pages 151–164.

Suraj Nair, Eugene Yang, Dawn Lawrie, Kevin Duh, Paul McNamee, Kenton Murray, James Mayfield, and Douglas W Oard. 2022. Transfer learning approaches for building cross-language dense retrieval models. In *Proceedings of ECIR*, pages 382–396.

Shahrzad Naseri, Jeff Dalton, Andrew Yates, and James Allan. 2021. CEQE: contextualized embeddings for query expansion. In *Proceedings of ECIR*, pages 467–482.

Jian-Yun Nie. 2010. *Cross-Language Information Retrieval*. Synthesis Lectures on Human Language Technologies. Morgan & Claypool Publishers.

Rodrigo Nogueira, Jimmy Lin, and AI Epistemic. 2019a. From doc2query to doctttttquery. *Online preprint*, 6.

Rodrigo Nogueira, Wei Yang, Kyunghyun Cho, and Jimmy Lin. 2019b. Multi-stage document ranking with BERT. *arXiv preprint arXiv:1910.14424*.

Rodrigo Nogueira, Wei Yang, Jimmy Lin, and Kyunghyun Cho. 2019c. Document expansion by query prediction. *arXiv preprint arXiv:1904.08375*.

Jonas Pfeiffer, Ivan Vulić, Iryna Gurevych, and Sebastian Ruder. 2020. MAD-X: An Adapter-Based Framework for Multi-Task Cross-Lingual Transfer. In *Proceedings EMNLP*, pages 7654–7673.

Edoardo Maria Ponti, Goran Glavaš, Olga Majewska, Qianchu Liu, Ivan Vulić, and Anna Korhonen. 2020. XCOPA: A multilingual dataset for causal commonsense reasoning. In *Proceedings of EMNLP*, pages 2362–2376.

Vikas Raunak, Arul Menezes, and Marcin Junczys-Dowmunt. 2021. The curious case of hallucinations in neural machine translation. In *Proceedings of NAACL*, pages 1172–1183.

Sylvestre-Alvise Rebuffi, Hakan Bilen, and Andrea Vedaldi. 2017. Learning multiple visual domains with residual adapters. In *Proceedings of NeurIPS*, volume 30.

Nils Reimers and Iryna Gurevych. 2020. Making monolingual sentence embeddings multilingual using knowledge distillation. In *Proceedings of EMNLP*, pages 4512–4525.

Andreas Rücklé, Gregor Geigle, Max Glockner, Tilman Beck, Jonas Pfeiffer, Nils Reimers, and Iryna Gurevych. 2021. AdapterDrop: On the efficiency of adapters in transformers. In *Proceedings of EMNLP*, pages 7930–7946.

Sebastian Ruder, Noah Constant, Jan Botha, Aditya Siddhant, Orhan Firat, Jinlan Fu, Pengfei Liu, Junjie Hu, Dan Garrette, Graham Neubig, et al. 2021. Xtreme-r: Towards more challenging and nuanced multilingual evaluation. In *Proceedings of EMNLP*.

Keshav Santhanam, Omar Khattab, Jon Saad-Falcon, Christopher Potts, and Matei Zaharia. 2022. ColBERTv2: Effective and efficient retrieval via lightweight late interaction. In *Proceedings of NAACL*, pages 3715–3734.

Peng Shi, He Bai, and Jimmy Lin. 2020. Cross-lingual training of neural models for document ranking. In *Proceedings of EMNLP (Findings)*, pages 2768–2773.

Emma Strubell, Ananya Ganesh, and Andrew McCallum. 2019. Energy and policy considerations for deep learning in nlp. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 3645–3650.

Weng Lam Tam, Xiao Liu, Kaixuan Ji, Lilong Xue, Xingjian Zhang, Yuxiao Dong, Jiahua Liu, Maodi Hu, and Jie Tang. 2022. Parameter-efficient prompt tuning makes generalized and calibrated neural text retrievers. *CoRR*, abs/2207.07087.

Eva Vanmassenhove, Dimitar Shterionov, and Matthew Gwilliam. 2021. Machine translationese: Effects of algorithmic bias on linguistic complexity in machine translation. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*, pages 2203–2213.

Fedor Vitiugin and Carlos Castillo. 2022. Cross-lingual query-based summarization of crisis-related social media: An abstractive approach using transformers. *arXiv preprint arXiv:2204.10230*.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Remi Louf, Morgan Funtowicz, Joe Davison, Sam Shleifer, Patrick von Platen, Clara Ma, Yacine Jernite, Julien Plu, Canwen Xu, Teven Le Scao, Sylvain Gugger, Mariama Drame, Quentin Lhoest, and Alexander Rush. 2020. Transformers: State-of-the-Art Natural Language Processing. In *Proceedings of EMNLP 2020: System Demonstrations*, pages 38–45.

Puxuan Yu, Hongliang Fei, and Ping Li. 2021. Cross-lingual language model pretraining for retrieval. In *Proceedings of WWW*, page 1029–1039.

Wei Zhao, Goran Glavaš, Maxime Peyrard, Yang Gao, Robert West, and Steffen Eger. 2020. On the limitations of cross-lingual encoders as exposed by reference-free machine translation evaluation. In *Proceedings of ACL*, pages 1656–1671.

Junnan Zhu, Qian Wang, Yining Wang, Yu Zhou, Jiajun Zhang, Shaonan Wang, and Chengqing Zong. 2019. Ncls: Neural cross-lingual summarization. In *Proceedings of EMNLP-IJCNLP*, pages 3054–3064.

LIME: Weakly-Supervised Text Classification Without Seeds

Seongmin Park and **Jihwa Lee**
ActionPower, Seoul, Republic of Korea
{seongmin.park, jihwa.lee}@actionpower.kr

Abstract

In weakly-supervised text classification, only label names act as sources of supervision. Predominant approaches to weakly-supervised text classification utilize a two-phase framework, where test samples are first assigned pseudo-labels and are then used to train a neural text classifier. In most previous work, the pseudo-labeling step is dependent on obtaining seed words that best capture the relevance of each class label. We present LIME[1], a framework for weakly-supervised text classification that entirely replaces the brittle seed-word generation process with entailment-based pseudo-classification. We find that combining weakly-supervised classification and textual entailment mitigates shortcomings of both, resulting in a more streamlined and effective classification pipeline. With just an off-the-shelf textual entailment model, LIME outperforms recent baselines in weakly-supervised text classification and achieves state-of-the-art in 4 benchmarks.

1 Introduction

Weakly-supervised text classification (Meng et al., 2018) is an important avenue of research in low-resourced text classification. Unlike in traditional text classification, all supervision derives from textual information in category names. Weakly-supervised classification offers a practical approach to classification because it does not necessitate massive amounts of training data.

Another distinct aspect of weakly-supervised text classification is that the system has access to the entire test set at evaluation time, instead of encountering test samples sequentially. Exploiting this characteristic, recent approaches employ keyword-matching pseudo-labeling schemes to tentatively assign class labels to each test sample, before using the information to train a separate classifier (Meng et al., 2018; Mekala and Shang, 2020; Wang et al., 2021). Pseudo-labels are assigned by counting how many "seed words" of each class are found in the test sample. Keyword matching-based labeling, however, is neither adaptable nor flexible because semantic information embedded in class names cannot be extracted adaptively for distinct classification tasks.

Inspired by recent advances in prompt-based text classification (Yin et al., 2019, 2020; Schick and Schütze, 2021), we replace the keyword-based pseudo-labeling step with a more streamlined entailment-based approach. Extensive experiments show that entailment-based classifiers assign more accurate pseudo-labels with greater task adaptability and much fewer hyperparameters. We find that our method realizes the benefits of both entailment-based classification and self-training.

Our contributions are as follows:

1. We present LIME, a novel framework for weakly-supervised text classification that utilizes textual entailment. LIME surpasses current state-of-the-art weakly-supervised methods in all tested benchmarks.

2. We show that self-training with pseudo-labels can mitigate unsolved robustness issues in entailment-based classification (Ma et al., 2021).

3. We experimentally confirm that higher confidence in pseudo-labels translates to better classification accuracy in self-training. We also find that a balance between filtering out low-confidence labels and preserving a sizable pseudo-training corpus is important.

2 Background

2.1 Weakly-supervised text classification

In weakly-supervised text classification, the system is allowed to view the entire test set at evaluation

[1] **L**abels **I**dentified with **M**aximal **E**ntailment

Proceedings of the 29th International Conference on Computational Linguistics, pages 1083–1088
October 12–17, 2022.

time. Having access to all test data allows novel preprocessing approaches unavailable in traditional text classification, such as preliminary clustering of test samples (Mekala and Shang, 2020; Wang et al., 2021) before attempting final classification. In the process, the system has an opportunity to examine overall characteristics of the test set.

Existing methods for weakly-supervised text classification focus on effectively leveraging such additional information. The dominant approach involves generating pseudo-data to train a neural text classifier. Most methods assign labels to samples in the test set by identifying operative keywords within the text (Meng et al., 2018). They obtain seed words that best represent each category name. Then, each sample in the test set is assigned a label with keywords most relevant to its content.

Later works improve this pipeline by automatically generating seed words (Meng et al., 2020b) or incorporating pre-trained language models to utilize contextual information of representative keywords (Mekala and Shang, 2020).

Seed-word-based pseudo-labeling, however, is heavily dependent on the existence of representative seed words in test samples. Seed-word-based matching cannot fully utilize information in contextual language representations, because the classification of each document involves brittle global hyperparameters such as the number of total seed words (Meng et al., 2020b) or word embedding distance (Wang et al., 2021).

In this work, we entirely forgo the seed word generation process during pseudo-labeling. We show that replacing seed-word generation with entailment-based text classification is more reliable and performant for text classification with weak supervision.

2.2 Entailment based text classification

Textual entailment (Fyodorov et al., 2000; MacCartney and Manning, 2009) measures the likeliness of a sentence appearing after another. Since entailment is evaluated to a probability value, the task can be extended for use in text classification. In entailment-based text classification, classification is posed as a textual entailment problem: given a test document, the system ranks the probabilities that sentences each containing a possible class label (*hypotheses*) will immediately follow the document text. The class label belonging to the most probable hypothesis is selected as the classification prediction. A hypothesis for topic classification, for example, could be "This text is about <*topic*>". The flexibility in prompt choices for constructing the hypotheses makes entailment-based classification extremely adaptable to different task types.

Although entailment-based sentence scoring is popular in zero- and few-shot text classification (Yin et al., 2019, 2020), the robustness of such approaches has recently been called into question (Ma et al., 2021). Since entailment-based classifiers rely heavily on lexical patterns, a large variance is observed in classification performance across different domains. We find that self-training commonly found in weakly-supervised classification mitigates such robustness issues in entailment-based classification to a large degree.

3 The LIME Framework

LIME enhances the two-phase weakly-supervised classification pipeline with an entailment-based pseudo-labeling scheme.

	Examples
Test sample (t)	"I love the food."
Class label (c)	"Positive"
Verbalizer	"Positive" $\rightarrow$ "*good*"
Prompt	"It was <$verbalizer(h_i)$>."
Hypothesis (h)	"It was good."

Table 1: Example test sample, class label, verbalizer, prompt, and entailment hypothesis. Converting class labels with a verbalizer is an optional procedure.

3.1 Phase 1: Pseudo-labeling

Textual entailment evaluates the likeliness of a hypothesis h succeeding some text t.

Given $C = \{c_1, c_2, \ldots, c_n\}$, the set of all possible labels for t, we generate $H = \{h_1, h_2, \ldots, h_n\}$, the set of all entailment hypothesis. Every sentence h_i asserts that its corresponding $c_i \in C$ is the correct label for t. h_i is constructed from a designated *prompt* and an optional *verbalizer* for each dataset (Schick and Schütze, 2021):

$$h_i = prompt(verbalizer(c_i))$$

Prompts dictate the wording of the hypotheses, while *verbalizers* convert each class label into a terminology better interpreted by entailment models. Pseudo-label for t is chosen as c_i that corresponds

Dataset	Type	# of Classes	Dataset size	Prompt
20News	News topic	5	17,871	*The text is about <class label>.*
AGNews	News topic	4	120,000	*The text is about <class label>.*
Yelp	Restarant review	2	38,000	*It was good. / It was bad.*
DBpedia	Wikipedia topic	14	560,000	*The text is about <class label>.*

Table 2: Statistics for benchmark datasets.

to the pair (t, h_i) with the highest entailment probability. Table 1 provides examples of verbalizers, prompts, and hypotheses.

3.2 Phase 2: Self-training

We adopt a similar self-training approach as existing methods in weakly-supervised text classification. We train a BERT-base model (Devlin et al., 2019) with a sequence classification feed-forward layer using pseudo-labels obtained in Phase 1.

We calculate the prediction confidence for each pseudo-label c_i assigned to t. Pseudo-labels under a certain confidence threshold are discarded during the text classifier training phase.

Confidence of label c_i is defined as the softmax over entailment probabilities of all hypotheses:

$$Confidence(c_i) = \frac{e^{p_i}}{\sum_{j=1}^{n} e^{p_j}}$$

where p_i is the entailment probability for the text pair (t, h_i), obtained from the entailment model.

4 Experiments

4.1 Experimental setting

In every experiment, we use a publically available BART-large model[2] (Lewis et al., 2020) trained on the MultiNLI (Williams et al., 2018) dataset as our entailment classifier. We also discard pseudo-labels with confidence under 50%. Although different thresholds lead to higher final F1 scores, we report results with confidence threshold of 50% for a fair comparison with previous research.

4.2 Baselines

We compare LIME with both entailment-based classification (Phase 1 without self-training) and previous research on weakly-supervised text classification. We also include BERT trained with supervision from original labels as a realistic upper bound for weakly-supervised classification.

WestClass (Meng et al., 2018) generates pseudo-documents for each class label. **ConWea** (Mekala and Shang, 2020) utilizes a pre-trained language model to discern keywords that carry different meanings under different contexts. **LotClass** (Meng et al., 2020b) is a framework for text classification using only label names. **LotClass** mines a pre-trained BERT model for seed words that are most likely to replace each class name. **X-Class** (Wang et al., 2021) is a state-of-the-art weakly-supervised classification system that collects seed words within the test documents instead of external sources. Documents are then grouped with a Gaussian Mixture Model before pseudo-labels are assigned.

4.3 Datasets

We run LIME on standard benchmarks in weakly-supervised classification: **20News** (Lang, 1995), **AGNews** (Zhang et al., 2015), **Yelp reviews** (Zhang et al., 2015), and **DBpedia** (Zhang et al., 2015). Detailed descriptions of each dataset, along with specific prompts used, are recorded in Table 2. We notably omit NYT datasets used in Meng et al. (2020a) and Wang et al. (2021), because only pre-processed (all lower-cased, pre-tokenized with a specific tokenizer) versions of the data were available. It is not possible to meaningfully evaluate the pseudo-labeling scheme in LIME if test samples are tokenized by a tokenizer different from that coupled with our entailment model.

5 Results

5.1 Classification performance

Final classification results are recorded in Table 3. LIME outperforms all baselines in terms of micro- and macro-F1 scores, even approaching the supervised baseline in the **Yelp** dataset.

We also find that training a new classifier with pseudo-labels (Phase 2 of LIME) does not amplify or propagate errors in incorrect pseudo-labels. The final classifier consistently scores roughly 10 points

[2]https://huggingface.co/facebook/bart-large-mnli

Model	20News	AGNews	Yelp	DBpedia
Supervised	96.45 / 96.42	93.99 / 93.99	95.70 / 95.70	98.96 / 98.96
Entailment classifier	67.95 / 67.50	79.94 / 79.99	94.79 / 94.79	80.14 / 79.27
WeSTClass	71.28 / 69.90	82.30 / 82.10	81.60 / 81.6	81.42 / 81.19
ConWea	75.73 / 73.26	74.60 / 74.20	71.40 / 71.20	N/A
LOTClass	73.78 / 72.53	86.89 / 86.82	87.75 / 87.68	86.66 / 85.98
X-Class	78.62 / 77.76	85.74 / 85.66	90.00 / 90.00	91.32 / 91.17
LIME	**79.74 / 79.56**	**87.21 / 87.16**	**95.22 / 95.22**	**92.19 / 92.20**

Table 3: Experiment results on 4 classification benchmarks. All reported scores in the form *micro-F1 / macro-F1*. Baselines are quoted from (Wang et al., 2021).

higher in F1 scores compared to the entailment classifier. Our results confirm findings from previous research that employ self-training to improve classification robustness in low-resource regimes (Mukherjee and Awadallah, 2020; Gowal et al., 2021).

5.2 Effect of label confidence thresholds

Figure 1 plots the spread of pseudo-label confidence produced in Phase 1 of LIME. We confirm that higher average confidence from the entailment classifier in Phase 1 robustly translates to higher classification accuracy for both the entailment classifier and the self-trained classifier in Phase 2.

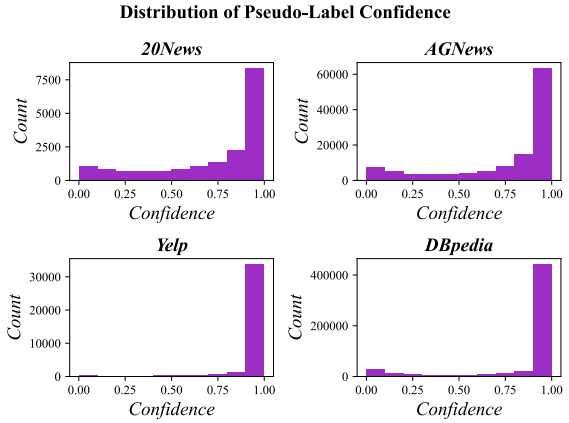

Figure 1: Histogram of pseudo-label confidence. More confident pseudo-labels result in more accurate classification self-training.

Another notable finding is that naively utilizing only high-confidence labels does not always guarantee a more accurate classifier. A trade-off exists between filtering out low-confidence labels and retaining a sizable training corpus. We find that confidence cut-off from 50% to 70% strikes a good balance between the two obligations (Figure 2).

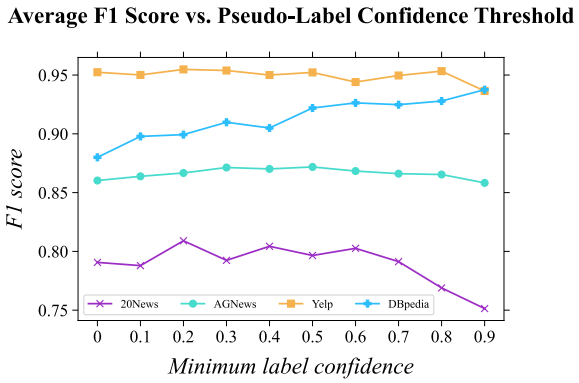

Figure 2: Effect of varying confidence thresholds on self-training F1 scores.

6 Conclusions

LIME proposes a streamlined pseudo-labeling method for weakly-supervised text classification. The framework combines flexibility of entailment-based classification with robustness of self-training. The resulting text classifier outperforms previous state-of-the-art in weakly-supervised classification. We also investigate the effect of pseudo-label confidence thresholds on self-trained classifier performance. Entailment model confidence accurately reflects label accuracy, but size of the pseudo-training set is also important for robust classification.

We identify several avenues for future research. For a fair comparison with previous research, we did not modify the self-training step with more advanced neural classifier architectures or confidence-aware self-training schemes (Mukherjee and Awadallah, 2020). Other auxiliary tasks, such as question-answering (McCann et al., 2018) or next sentence prediction (Ma et al., 2021) can also extend the LIME framework as alternate pseudo-classifiers.

References

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, Minneapolis, Minnesota. Association for Computational Linguistics.

Yaroslav Fyodorov, Yoad Winter, and Nissim Francez. 2000. A natural logic inference system.

Sven Gowal, Po-Sen Huang, Aaron van den Oord, Timothy Mann, and Pushmeet Kohli. 2021. Self-supervised adversarial robustness for the low-label, high-data regime. In *International Conference on Learning Representations*.

Ken Lang. 1995. Newsweeder: Learning to filter netnews. In Armand Prieditis and Stuart Russell, editors, *Machine Learning Proceedings 1995*, pages 331–339. Morgan Kaufmann, San Francisco (CA).

Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Veselin Stoyanov, and Luke Zettlemoyer. 2020. BART: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7871–7880, Online. Association for Computational Linguistics.

Tingting Ma, Jin-Ge Yao, Chin-Yew Lin, and Tiejun Zhao. 2021. Issues with entailment-based zero-shot text classification. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 2: Short Papers)*, pages 786–796, Online. Association for Computational Linguistics.

Bill MacCartney and Christopher D. Manning. 2009. An extended model of natural logic. In *Proceedings of the Eight International Conference on Computational Semantics*, pages 140–156, Tilburg, The Netherlands. Association for Computational Linguistics.

Bryan McCann, Nitish Shirish Keskar, Caiming Xiong, and Richard Socher. 2018. The natural language decathlon: Multitask learning as question answering.

Dheeraj Mekala and Jingbo Shang. 2020. Contextualized weak supervision for text classification. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 323–333, Online. Association for Computational Linguistics.

Yu Meng, Jiaxin Huang, Guangyuan Wang, Zihan Wang, Chao Zhang, Yu Zhang, and Jiawei Han. 2020a. *Discriminative Topic Mining via Category-Name Guided Text Embedding*, page 2121–2132. Association for Computing Machinery, New York, NY, USA.

Yu Meng, Jiaming Shen, Chao Zhang, and Jiawei Han. 2018. Weakly-supervised neural text classification. In *proceedings of the 27th ACM International Conference on information and knowledge management*, pages 983–992.

Yu Meng, Yunyi Zhang, Jiaxin Huang, Chenyan Xiong, Heng Ji, Chao Zhang, and Jiawei Han. 2020b. Text classification using label names only: A language model self-training approach. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 9006–9017, Online. Association for Computational Linguistics.

Subhabrata Mukherjee and Ahmed Awadallah. 2020. Uncertainty-aware self-training for few-shot text classification. In *Advances in Neural Information Processing Systems*, volume 33, pages 21199–21212. Curran Associates, Inc.

Timo Schick and Hinrich Schütze. 2021. Exploiting cloze-questions for few-shot text classification and natural language inference. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*, pages 255–269, Online. Association for Computational Linguistics.

Zihan Wang, Dheeraj Mekala, and Jingbo Shang. 2021. X-class: Text classification with extremely weak supervision. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 3043–3053, Online. Association for Computational Linguistics.

Adina Williams, Nikita Nangia, and Samuel Bowman. 2018. A broad-coverage challenge corpus for sentence understanding through inference. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 1112–1122, New Orleans, Louisiana. Association for Computational Linguistics.

Wenpeng Yin, Jamaal Hay, and Dan Roth. 2019. Benchmarking zero-shot text classification: Datasets, evaluation and entailment approach. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 3914–3923, Hong Kong, China. Association for Computational Linguistics.

Wenpeng Yin, Nazneen Fatema Rajani, Dragomir Radev, Richard Socher, and Caiming Xiong. 2020. Universal natural language processing with limited annotations: Try few-shot textual entailment as a start. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 8229–8239, Online. Association for Computational Linguistics.

Xiang Zhang, Junbo Zhao, and Yann LeCun. 2015. Character-level convolutional networks for text classification. In *Advances in Neural Information Processing Systems*, volume 28. Curran Associates, Inc.

Multi-Stage Framework with Refinement based Point Set Registration for Unsupervised Bi-Lingual Word Alignment

Silviu Vlad Oprea[*]
Amazon Alexa AI
Cambridge, UK
silviuvo@amazon.co.uk

Sourav Dutta
Huawei Research Centre
Dublin, Ireland
sourav.dutta2@huawei.com

Haytham Assem[†]
Amazon Alexa AI
Cambridge, UK
hithsala@amazon.co.uk

Abstract

Cross-lingual alignment of word embeddings are important in knowledge transfer across languages, for improving machine translation and other multi-lingual applications. Current unsupervised approaches relying on learning structure-preserving transformations, using adversarial networks and refinement strategies, suffer from instability and convergence issues. This paper proposes *BioSpere*, a novel multi-stage framework for unsupervised mapping of bi-lingual word embeddings onto a shared vector space, by combining *adversarial initialization*, *refinement procedure* and *point set registration*. Experiments for parallel dictionary induction and word similarity demonstrate state-of-the-art unsupervised results for *BioSpere* on diverse languages – showcasing robustness against variable adversarial performance.

1 Introduction and Background

Distributed word representations like Word2Vec (Mikolov et al., 2013), GloVe (Pennington et al., 2014) and FastText (Bojanowski et al., 2017) capture rich semantic meaning, and is used for a range of NLP tasks. *Cross-lingual word embeddings* (CLWE) entails mapping vocabularies of different languages onto a single vector space for capturing semantic similarity across languages (Upadhyay et al., 2016) – for machine translation (Artetxe et al., 2018a; Lample et al., 2018a,b), POS tagging (Zhang et al., 2016; Ahmad et al., 2019), & named entity recognition (Tsai and Roth, 2016; Xie et al., 2018; Chen et al., 2019).

Linguistic Correlation. This work is based on the observation that, monolingual representation spaces learnt independently for different languages tend to exhibit similarity in terms of *geometric properties and orientations* (Mikolov and Sutskever, 2013) [1]. Further, the frequency of words across languages have also been shown to follow the *Zipf's distribution* [2], with an overlap of nearly 70% for the most frequent words (Aldarmaki et al., 2018) and 60% for synonyms (Dinu et al., 2015) across language pairs. Existing techniques for extracting cross-lingual word correspondences rely on above inter-dependencies to learn transformations across monolingual embedding spaces.

State-of-the-art & Challenges. Early approaches for obtaining multi-lingual word embeddings used parallel or comparable corpora (Gouws et al., 2015; Mogadala and Rettinger, 2016; Vulić and Moens, 2016). However, such methods are not scalable as parallel datasets, especially for low-resource languages, are scarce. Linear transformations between two monolingual embedding spaces (via optimization formulation (Schönemann, 1966)) using small manually created bi-lingual dictionaries were thus proposed (Mikolov and Sutskever, 2013; Artetxe et al., 2016). Words having similar surface forms across languages were used to induce seed dictionaries for semi-supervised approaches (Artetxe et al., 2017; Zhou et al., 2019; Doval et al., 2018). Rigid transformation based point set registration Cao and Zhao (2018), supervised cross-lingual alignment, joint training (Joulin et al., 2018; Jawanpuria et al., 2019; Alaux et al., 2019; Wang et al., 2020) with feedback-based learning (Yuan et al., 2020) were also studied. Unsupervised bi-lingual word alignment using *adversarial training* (Barone, 2016; Zhang et al., 2017a,b) were shown to produce good results in MUSE (Conneau et al., 2018). Inverted softmax (Smith et al., 2017) approach was shown to tackle the "hubness problem" (Radovanović et al., 2010) caused by dense vector space regions (called

[*] Work done during internship at Huawei Research, Ireland.
[†] Work done while the author was at Huawei Research, Ireland.

[1] For example, the embedding vector distribution of numbers and animals in English show a similar geometric structural formation as their Spanish counterparts.

[2] observed on 10 million words from Wikipages on 30 languages (en.wikipedia.org/wiki/Zipf's_law)

Proceedings of the 29th International Conference on Computational Linguistics, pages 1089–1097
October 12–17, 2022.

hubs), which adversely affected bi-lingual word translation. However, the performance of adversarial techniques were shown to suffer from convergence instability. Further, Søgaard et al. (2018) found the above approaches to fail for morphologically rich languages. Hence, optimization using Gromov-Wasserstein, Sinkhorn distance, and Iterative Closest Point were explored (Grave et al., 2019; Alvarez-Melis and Jaakkola, 2018; Xu et al., 2018; Hoshen and Wolf, 2018; Hartmann et al., 2019). *Adversarial auto-encoders* using *cyclic loss* and stacked refinements (Mohiuddin and Joty, 2019, 2020) recently achieved improved results.

Contributions. This paper proposes *BioSpere* (Bi-Lingual Word Translation via Point Set Registration and Refinement), a novel approach for *unsupervised bi-lingual word correspondence induction*. Our key contributions are as follows:

• *BioSpere*, an *unsupervised multi-stage* framework for learning bi-lingual word alignment, by using a combination of adversarial training, refinement procedure, and point set registration;

• Unsupervised criterion using *cycle-loss consistency* for adversarial model choice;

• Experiments on diverse language pairs showing *improved accuracy* on different tasks; and,

• *Robustness* to hubness and convergence issues.

We next describe the detailed working of the different modules in the *BioSpere* framework.

2 *BioSpere* Framework

Consider, two monolingual word embedding spaces, $X = \{x_n\}_{n=1}^{N}$ and $Y = \{y_m\}_{m=1}^{M}$, trained independently, to be provided as source and target language representations, respectively. *BioSpere* aims to map a word in the source language to its translation (or semantically similar word) in a target language, without cross-lingual supervision (Zhang et al., 2019). *BioSpere* consists of 4 modules – *Align, Correspond, Transform* and *Generate* (shown in Figure 1), as discussed next.

• **Align Module** – The *Align* module uses an adversarial training (Ganin et al., 2016) to estimate an initial mapping between the words across the languages, by learning an rotational transformation between the input embeddings spaces. Assuming $x \sim p_{data}(x)$ and $y \sim p_{data}(y)$ to be the input data distributions, we learn two linear mappings $F : X \rightarrow Y$ and $G : Y \rightarrow X$, referred to as *forward* and *backward generators*, respectively. A generative adversarial network is used to train a model D_Y (*discriminator*) to discriminate between generated synthetic target embeddings $Y_{syn} = FX = \{F(x_n)\}_{n=1}^{N}$, and the original embeddings Y. Similarly, we train another discriminator, D_X, in the opposite direction to discriminate between synthetic source embeddings $X_{syn} = GY = \{G(y_m)\}_{m=1}^{M}$ and the original X.

The *adversarial loss* formulates matching the distribution of synthetic embeddings to the real distribution. Thus, for forward generator $F : X \rightarrow Y$ and its discriminator model D_Y, the loss is: $\mathcal{L}_{adv}(F, D_Y, X, Y) = \mathbb{E}_{y \sim p_{data}(y)}[\log D_Y(y)] + \mathbb{E}_{x \sim p_{data}(x)}[\log(1 - D_Y(F(x))]$ (refer Appendix B).

A similar loss $\mathcal{L}_{adv}(G, D_X, Y, X)$ is used for backward generator $G : Y \rightarrow X$ and discriminator D_X.

We also incorporate the objective used in Mohiuddin and Joty (2020) (considering word translations are symmetric in general) – the learned generators should not contradict each other, but should be *cycle-consistent*. That is, given a source embedding x, the forward translation cycle should attempt to produce an output that coincides with x, i.e., $G(F(x)) \approx x$; and vice-versa for the backward translation cycle. Thus, we have:

$$L_{cyc}(F, G) = \mathbb{E}_{x \sim data(x)} \|G(F(x))\|_2 + \mathbb{E}_{y \sim data(y)} \|F(G(y))\|_2$$

Following Conneau et al. (2018), to preserve dot product and L_2 distances from the monolingual space, we ensure F and G remain roughly orthogonal during training by alternating parameter update with $F \leftarrow (1+\beta)F - \beta(FF^T)F$ (and analogously for G). This corresponds to *CycleGAN* (Zhu et al., 2017), a generative adversarial network (used in our *Align* module), to provide an initial aligned embedding space, $X_A = F(X)$ and $Y_A = G(Y)$.

• **Correspond Module** – The above alignment obtained based on cyclic loss, might suffer from adversarial network convergence instability. To address this issue, the *Correspond* module performs a refinement step based on *symmetric re-weighting*, shown to be effective for alignment (Artetxe et al., 2018a, 2016, 2017; Mohiuddin and Joty, 2020).

A synthetic seed parallel dictionary, $\mathcal{D}$, is thus induced by computing the mutual nearest neighbour (in both directions) across the aligned embeddings (X_A and Y_A), as: $\sigma_{nm} = \delta(F(x_n), y_m) + \delta(x_n, G(y_m))$, where δ is a distance measure in both X_A and Y_A. As in Conneau et al. (2018), we adopt the *cross-domain similarity local scaling* (CSLS) measure, which addresses the "hubness" problem. Observe, σ_{nm} also uses bi-directional similarity computation. In our experiments, the dictionary induction

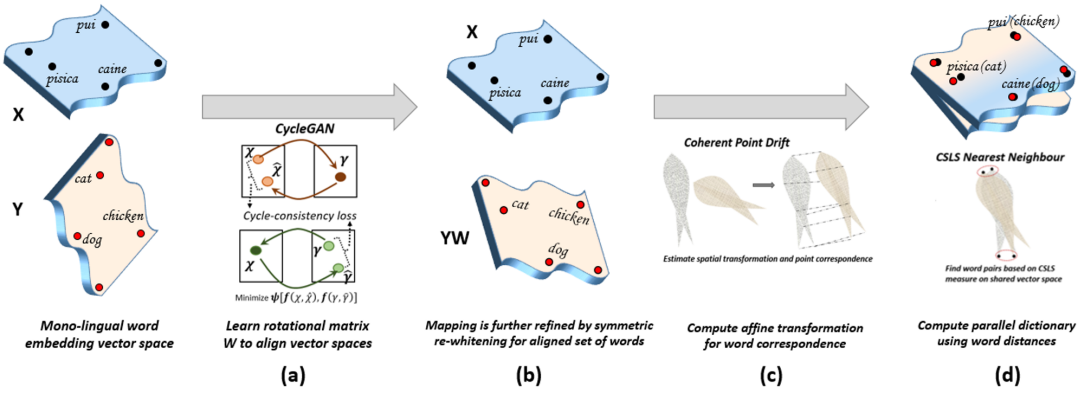

Figure 1: Toy illustration (on *en-ro* language pair) of the different modules of *BioSpere* – (a) *Align*, (b) *Correspond*, (c) *Transform*, and (d) *Generate* – for unsupervised parallel dictionary construction.

is on 25K most frequent words (out of 200K words) from source and target languages. *Symmetric re-weighting* is now performed via 3 steps:
(i) Whitening: makes the embedding dimensions uncorrelated with unit variance using *spherical transformation*. We use ZCA whitening, wherein the original embeddings X and Y are normalized and mean-centered, followed by a linear transformation via matrices $W_x = (X^T X)^{-1/2}$ and $W_y = (Y^T Y)^{-1/2}$, to obtain $X_w = XW_x$ and $Y_w = YW_y$.
(ii) Orthogonal Transformation: provides a transformation of the whitened embeddings onto a common space. U, Σ, and V^T are obtained via singular value decomposition of $(X_w^{\mathcal{D}})^T Y_w^{\mathcal{D}}$, where $X_w^{\mathcal{D}}$ and $Y_w^{\mathcal{D}}$ are whitened embeddings from the seed dictionary $\mathcal{D}$. The transformation is computed as $X_o = X_w U \Sigma^{1/2}$ and $Y_o = Y_w V \Sigma^{1/2}$.
(iii) De-Whitening: restores the original variance in the embedding dimensions of the transformed vectors – computes a refined vector embedding as: $X_C = X_o U^T (X^T X)^{1/2} U$ and $Y_C = Y_o V^T (Y^T Y)^{1/2} V$.
• **Transform Module** – The *Transform* module performs a further refinement on the embeddings X_C and Y_C using the concept of *point set registration*. Specifically, we use the *Coherent Point Drift* (CPD) algorithm (Myronenko and Song, 2010), an unsupervised probabilistic framework which assigns *point-to-point correspondence* between two sets of points, akin to finding word translation pairs in our setting. Here, the task of aligning two embedding spaces is performed using a density estimation problem based on the *Gaussian Mixture Model* (GMM). We direct interested readers to the details of CPD algorithm provided by Myronenko and Song (2010), and briefly in Appendix A.

The use of CPD provides the following advantages – (i) GMM enables *BioSpere* to tackle the "hubness" problem (shown in Zhou et al. (2019)), and (ii) CPD imposes *Motion Coherence Theory* (MCT) (Yuille and Grzywacz, 1988) to force the GMM centroids to move coherently as a group, preserving the underlying topological structure.

We use *affine CPD transformation*, providing a higher degree of freedom compared to the rigid procedure of (Cao and Zhao, 2018) and Procrustes, to compute the modified source embeddings as: $X_T = (R X_C^T * s + t)^T$, where R is a rotation matrix, t is a translation vector, and s is a scaling constant. We run CPD twice for each language pair, once in each directions, generating the transformed source and target language embeddings X_T and Y_T.
• **Generate Module** – The *Generate* module iterates between the above correspond and transform steps until convergence is reached. Equipped with the final aligned X_T and Y_T embedding spaces, the resultant parallel dictionary is computed using the bi-directional CSLS measure, similar to the construction of the intermediate dictionary in the *Correspond* module. . For convergence of the iterative symmetric re-weighting refinement and CPD, we adopt the criteria of Artetxe et al. (2018b); Mohiuddin and Joty (2020). The generated word pairs are compared with ground-truth parallel dictionaries to compute the accuracy of *BioSpere*.

Overview. Intuitively, the interactions across the different components in *BioSpere* are as: The adversarial module provides an initial embedding space alignment, but might be prone to convergence issues. The refinement stage then provides robustness against such training losses. However, the refinement process being a supervised approach by definition, errors in intermediate synthetic dictionary construction might propagate, degrading the efficacy. The final point correspondence CPD step,

Algorithm	en-es		en-de		en-fr		en-ru	
	→	←	→	←	→	←	→	←
Supervised Approaches								
Non-Adv	81.4	82.9	73.5	72.4	81.1	82.4	51.7	63.7
DeMa-BME	82.8	85.4	77.2	75.1	83.2	83.5	49.2	63.6
GeoMM	81.4	85.5	74.7	76.7	82.1	84.1	51.3	*67.6*
RCSLS	*84.1*	*86.3*	*79.1*	*76.3*	*83.3*	*84.1*	*57.9*	67.2
Unsupervised Approaches								
SinkHorn**	79.5	77.8	69.3	67.0	77.9	75.5	-	-
Non-Adv	82.1	84.1	74.7	73.0	82.3	82.9	47.5	61.8
Was-Proc	82.8	84.1	75.4	73.3	82.6	82.9	43.7	59.1
GW-Proc	81.7	80.4	71.9	72.8	81.3	78.9	45.1	43.7
MUSE	81.7	83.3	74.0	72.2	82.3	82.1	44.0	59.1
VecMap††	82.3	84.7	75.1	74.3	82.3	*83.6*	*49.2*	**65.6**
UMH	82.5	84.9	74.8	73.7	*82.9*	83.3	45.3	62.8
Adv-Auto	*83.0*	*85.2*	**76.2**	*74.7*	82.3	83.5	48.4	*64.5*
BioSpere	**83.3**	**85.4**	*75.8*	**75.8**	**83.4**	**84.1**	**49.5**	64.0

'-' denotes failure of the training network to converge reasonably

** Uses cosine similarity instead of CSLS, and results as reported in Zhou et al. (2019)

†† Results taken from Zhou et al. (2019)

Table 1: CSLS@1 *word translation* results on the dataset of Conneau et al. (2018).

being unsupervised, is agnostic to such errors and provides enhanced cross-lingual embedding space alignment. The overall *BioSpere* framework (CycleGAN + SR + affine CPD) is seen to perform the best and robustly across different languages in our empirical evaluations. More details and evaluations can be found in (Oprea et al., 2020).

3 Empirical Evaluation

We evaluate *BioSpere* on mapping semantically similar words across languages, for *bi-lingual dictionary induction*, *word similarity* and *sentence translation retrieval* tasks across diverse languages.

Dataset. We follow the setup of Conneau et al. (2018), and use FastText monolingual vector embeddings (with 300 dimensions) (Bojanowski et al., 2017) for the top $200K$ most frequent words of each language as input vocabulary. We consider ***8*** different language pairs (including morphologically rich) – English (en), German (de), French (fr), Spanish (es), Italian (it), Russian (ru), Hebrew (he), Finnish (fi), and Romanian (ro) – a mix of *isolating, fusional and agglutinative languages* with *dependent and mixed marking* (Søgaard et al., 2018).

Evaluation. On *word translation* (dictionary induction), we use the gold dictionary with 1,500 source test words (and 200K target vocabulary) (`github.com/facebookresearch/MUSE`), while *sentence translation retrieval* uses Europarl corpus containing 2,000 source and 200K target sentences. We report *Precision@1* (P@1) based on CSLS criteria (Conneau et al., 2018). For *word similarity* on SemEval 2017 data (Camacho-Collados et al., 2017) we report the Pearson's correlation.

Baselines. The performance of *BioSpere* is compared against the following *unsupervised* methods:

(1) MUSE (Conneau et al., 2018) – Uses GAN (Goodfellow et al., 2014) to learn transformations with Procrustes (Schönemann, 1966) [3];

(2) Adv-Auto (Mohiuddin and Joty, 2020) – State-of-the-art using adversarial auto-encoder to create synthetic dictionary, refined by symmetric reweighting & Procrustes strategies [4];

(3) VecMap (Artetxe et al., 2018a) – Self-learning iterative algorithms exploiting structural similarities between embedding spaces for alignment [5];

(4) SinkHorn (Xu et al., 2018): GAN trained on cyclic loss and Sinkhorn distance (Cuturi, 2013);

(5) Non-Adv (Hoshen and Wolf, 2018) – Uses dimensionality reduction with Iterative Closest Point (Besl and McKay, 1992) algorithm;

(6) Was-Proc (Grave et al., 2019) – Computes bi-stochastic matrix for assignment by jointly optimizing Wasserstein dist. (Mémoli, 2011) & Procrustes;

(7) GW-Proc (Alvarez-Melis and Jaakkola, 2018) – Formulates optimal flow across domains using Gromov-Wasserstein distance (Mémoli, 2011); and

(8) UMH (Alaux et al., 2019) – Uses language correlation for learning via constraint optimization.

We also report the *supervised* approaches:

(1) RCSLS (Joulin et al., 2018): Optimizes CSLS criteria for learning (Conneau et al., 2018);

(2) GeoMM (Jawanpuria et al., 2019): Language specific geometric rotations are learnt to align; and

(3) DeMa-BME (Zhou et al., 2019): Weakly-supervised approach for learning Gaussian Mixture Model between embeddings spaces.

Unsupervised Model Selection. For choosing the best performing model setting during adversarial training and convergence (a challenge in unsupervised setting), we follow Conneau et al. (2018) and use *CSLS* measure (denoted as DMC) to quantify the closeness of source and target mapped embedding spaces. However, adopting cyclic-consistency, we extend CSLS (termed as DualDMC) to measure the average *bi-directional cosine similarity* between source and target spaces (as in *Correspond* module), for model selection.

Parameter Setting. For a robust framework, we did not perform extensive parameter search, and most parameters were set to fixed values, or selected via two successive degradation of the unsu-

[3] Code from `github.com/facebookresearch/MUSE`

[4] `ntunlpsg.github.io/project/unsup-word-translation`

[5] Code obtained from `github.com/artetxem/vecmap`

Algorithm	en-fi		en-he		en-ro	
	→	←	→	←	→	←
MUSE	43.7	53.7	38.0	-	58.0	66.0
VecMap	**49.9**	63.5	44.6	57.7	64.2	71.8
Adv-Auto	49.8	**65.5**	46.1	58.6	62.6	71.9
BioSpere	**49.9**	**65.5**	**46.6**	**59.1**	**65.4**	**74.3**

(a)

Algorithm	en-de		en-es		en-it	
	→	←	→	←	→	←
MUSE	0.708	0.713	0.712	0.711	0.710	0.712
VecMap	0.719	0.719	0.721	0.721	**0.746**	**0.746**
Adv-Auto	-	0.720	0.724	0.718	0.722	0.721
BioSpere	**0.726**	**0.725**	**0.730**	**0.728**	0.722	0.723

(b)

Algorithm	en-es		en-fr		en-fi	
	→	←	→	←	→	←
MUSE	75.1	73.9	69.1	69.9	64.2	64.0
VecMap	74.7	74.4	69.6	69.3	64.4	64.1
Adv-Auto	75.0	75.7	68.0	**71.0**	64.1	64.5
BioSpere	**76.7**	**76.3**	**70.2**	70.9	**65.1**	**65.9**

(c)

Table 2: Performance of competing approaches for (a) CSLS@1 on *word translation* for morphologically rich languages, (b) Pearson's Correlation score for *word similarity* task on SemEval 2017 dataset, and (c) Precision@1 results for *sentence translation retrieval* on Europarl data.

pervised DualDMC criteria. Following Conneau et al. (2018), we fed the adversarial discriminator with the 50K most frequent words, and the discriminator had an input dropout layer with a rate of 0.1. In our experiments, we only tuned the weight assigned to the cyclic loss between 5 and 10, and ran the framework under different random seeds, picking the best model using unsupervised DualDMC.

3.1 Experimental Results

Word Translation – involves the retrieval of a source word translation to a target language for parallel dictionary construction. We use the ground-truth dictionaries of Conneau et al. (2018). From Table 1, we observe that *BioSpere* provides *better translation results* in nearly all of the four language pairs (across unsupervised methods). We achieve better results compared to even supervised methods like Non-Adv and DeMa-BME, and are comparable to RCSLS (e.g., *en* → *es and en* → *fr*). Since, MUSE, VecMap, and Adv-Auto consistently perform well, they are selected as competing baselines for the remaining experiments. We also explore the performance on "difficult" *morphologically rich languages* like Finnish, Hebrew and Romanian (Søgaard et al., 2018). Table 2(a) shows that *BioSpere* is efficient in such settings, outperforming existing approaches, across the languages.

Semantic Word Similarity – evaluates the quality of cross-lingual word alignment based on the correlation between cosine similarity between words in different languages and human-annotated word similarity scores. Table 2(b) shows that *BioSpere* achieves a better Pearson's correlation to human-annotated scores across languages (except *it*) – providing better alignment across languages.

Sentence Translation Retrieval – studies sentence translation retrieval on Europarl corpus. Similar to Conneau et al. (2018), a sentence is represented as a bag-of-words and the idf-weighted average of word embeddings is considered as its encoding. For each source sentence, the closest sentence from the target language is returned as its translation. Table 2(c) depicts that *BioSpere* provides better sentence translation retrieval accuracy with upto 1.5% P@1 score improvements.

Algorithm	en-de		en-fi		en-ro	
	→	←	→	←	→	←
MUSE GAN	59.8	60.5	22.3	24.1	34.5	49.6
CycleGAN	69.8	69.6	27.7	48.3	44.4	52.5
CycleGAN + Procrustes	73.8	73.3	46.2	62.0	59.5	67.2
CycleGAN + SR	75.5	74.7	46.9	64.9	63.5	71.6
CycleGAN + rigid CPD	74.5	74.2	45.9	62.3	60.5	67.3
CycleGAN + affine CPD	75.2	74.7	**50.2**	**65.7**	**65.5**	72.5
BioSpere	**75.8**	**75.8**	49.9	65.5	65.4	**74.3**
Bad-GAN	70.5	62.9	25.1	36.3	42.1	51.4
Bad-GAN + Procrustes	74.5	73.3	46.7	61.7	59.5	68.9
Bad-GAN + SR	**75.9**	73.8	45.7	61.7	63.1	72.3
Bad-GAN + affine CPD	75.3	74.7	**51.7**	**65.7**	63.1	72.6
***BioSpere* with Bad-GAN**	**75.9**	**75.9**	**51.7**	65.4	**64.0**	**73.1**

Table 3: Ablation and Robustness study of *BioSpere* on *word translation* with (Conneau et al., 2018) dataset.

Ablation Study – Table 3 tabulates the results for varying components of *BioSpere*. CycleGAN (using cycle-loss consistency) performs better than MUSE GAN, while the refinement procedures of symmetric re-weighting (SR) and Procrustes seem to perform similarly (SR being slightly better for morphologically rich languages). As discussed previously, we observe that higher degrees of translational freedom provided by *affine CPD* performs better than rigid CPD (of Cao and Zhao (2018)). To study the *robustness* of *BioSpere* to adversarial convergence issues, we intentionally select a sub-optimal CycleGAN model from the *Align* module, denoted as *Bad-GAN* in Table 3. We observe that SR refinement recovers from such convergence issues (better than Procrustes) – providing an accuracy comparable to a properly trained model (selected using *DualDMC*). Specifically, for *fi* → *en*, the performance of Bad-GAN is around 12% worse than the best CycleGAN model. However, the final accuracy of *BioSpere* differs by only 1% (in Table 3) even with Bad-GAN initialization.

4 Conclusion

This paper proposed *BioSpere*, a *multi-stage unsupervised cross-lingual word embedding alignment framework* – based on the novel coupling of *generative adversarial training*, *refinement procedure* and *point set registration*. Experiments with diverse tasks on multiple languages demonstrate improved results over existing methods, and also depict robustness to hubness and adversarial performance.

References

W. U. Ahmad, Z. Zhang, X. Ma, E. Hovy, K. Chang, and N. Peng. 2019. On difficulties of Cross-lingual Transfer with Order Differences: A Case Study on Dependency Parsing. In *NAACL*, pages 2440–2452.

J. Alaux, E. Grave, M. Cuturi, and A. Joulin. 2019. Unsupervised Hyperalignment for Multilingual Word Embeddings. In *ICLR*, pages 1–11.

H. Aldarmaki, M. Mohan, and M. Diab. 2018. Unsupervised Word Mapping Using Structural Similarities in Monolingual Embeddings. *Transactions of the Association for Computational Linguistics*, 6:185–196.

D. Alvarez-Melis and T. Jaakkola. 2018. Gromov-Wasserstein Alignment of Word Embedding Spaces. In *EMNLP*, pages 1881–1890.

M. Artetxe, G. Labaka, and E. Agirre. 2016. Learning Principled Bilingual Mappings of Word Embeddings while Preserving Monolingual Invariance. In *EMNLP*, pages 2289–2294.

M. Artetxe, G. Labaka, and E. Agirre. 2017. Learning Bilingual Word Embeddings with (almost) no Bilingual Data. In *ACL*, pages 451–462.

M. Artetxe, G. Labaka, and E. Agirre. 2018a. A Robust Self-learning Method for Fully Unsupervised Cross-lingual Mappings of Word Embeddings. In *ACL*, pages 789–798.

M. Artetxe, G. Labaka, and E. Agirre. 2018b. Generalizing and Improving Bilingual Word Embedding Mappings with a Multi-step Framework of Linear Transformations. In *AAAI*, pages 5012–5019.

A. V. M. Barone. 2016. Towards Cross-lingual Distributed Representations without Parallel Text Trained with Adversarial Autoencoders. In *Workshop on Representation Learning for NLP*, pages 121–126.

P. J. Besl and N. D. McKay. 1992. A Method for Registration of 3-D Shapes. *Transactions on Pattern Analysis and Machine Intelligence*, 14(2):239–256.

P. Bojanowski, E. Grave, A. Joulin, and T. Mikolov. 2017. Enriching Word Vectors with Subword Information. *Transactions of the Association for Computational Linguistics*, 5:135–146.

F. L. Bookstein. 1989. Principal Warps: Thin-Plate Splines and the Decomposition of Deformations. *Transactions on Pattern Analysis and Machine Intelligence*, 11(6):567–585.

R. W. Brislin. 1970. Back-translation for Cross-cultural Research. *Journal of Cross-Cultural Psychology*, 1(3):185–216.

J. Camacho-Collados, M. T. Pilehvar, N. Collier, and R. Navigli. 2017. Semeval-2017 Task 2: Multilingual and cross-lingual semantic word similarity. In *SemEval*.

H. Cao and T. Zhao. 2018. Point Set Registration for Unsupervised Bilingual Lexicon Induction. In *IJCAI*, pages 3991–3997.

X. Chen, A. H. Awadallah, H. Hassan, W. Wang, and C. Cardie. 2019. Multi-source Cross-lingual Model Transfer: Learning what to Share. In *ACL*, pages 3098–3112.

A. Conneau, G. Lample, M. Ranzato, L. Denoyer, and H. Jégou. 2018. Word Translation Without Parallel Data. In *ICLR*, pages 1–14.

M. Cuturi. 2013. Sinkhorn Distances: Lightspeed Computation of Optimal Transport. In *NIPS*, pages 2292–2300.

A. P. Dempster, N. M. Laird, and D. B. Rubin. 1977. Maximum Likelihood from Incomplete Data via the EM Algorithm. *Journal of the Royal Statistical Society*, 39(1):1–38.

A. Dinu, L. P. Dinu, and A. S. Uban. 2015. Cross-lingual Synonymy Overlap. In *RANLP*, pages 147–152.

Y. Doval, J. Camacho-Collados, L. Espinosa-Anke, and S. Schockaert. 2018. Improving Cross-Lingual Word Embeddings by Meeting in the Middle. In *EMNLP*, pages 294–304.

Y. Ganin, E. Ustinova, H. Ajakan, P. Germain, H. Larochelle, F. Laviolette, M. Marchand, and V. Lempitsky. 2016. Domain-Adversarial Training of Neural Networks. *Journal of Machine Learning Research*, 17(1):2096–2030.

W. Gao and R. Tedrake. 2019. FilterReg: Robust and Efficient Probabilistic Point-Set Registration Using Gaussian Filter and Twist Parameterization. In *CVPR*, pages 11087–11096.

I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio. 2014. Generative Adversarial Nets. In *NIPS*, pages 2672–2680.

S. Gouws, Y. Bengio, and G. Corrado. 2015. BilBOWA: Fast Bilingual Distributed Representations without Word Alignments. In *ICML*, pages 748–756.

E. Grave, A. Joulin, and Q. Berthet. 2019. Unsupervised Alignment of Embeddings with Wasserstein Procrustes. In *AISTATS*, pages 1880–1890.

M. Hartmann, Y. Kementchedjhieva, and A. Søgaard. 2019. Comparing Unsupervised Word Translation Methods Step by Step. In *NeurIPS*, pages 6033–6043.

G. E. Hinton, C. K. I. Williams, and M. D. Revow. 1992. Adaptive Elastic Models for Hand-printed Character Recognition. In *NIPS*, pages 512–519.

O. Hirose. 2020. A Bayesian Formulation of Coherent Point Drift. *Transactions on Pattern Analysis and Machine Intelligence*.

J. Ho and S. Ermon. 2016. Generative Adversarial Imitation Learning. In *NIPS*, pages 4565–4573.

J. Ho, M. H. Yang, A. Rangarajan, and B. Vemuri. 2007. A New Affine Registration Algorithm for Matching 2D Point Sets. In *WACV*, pages 25–25.

Y. Hoshen and L. Wolf. 2018. Non-Adversarial Unsupervised Word Translation. In *EMNLP*, pages 469–478.

H. Huang, E. Kalogerakis, S. Chaudhuri, D. Ceylan, V. G. Kim, and E. Yumer. 2017. Learning Local Shape Descriptors from Part Correspondences with Multiview Convolutional Networks. *Transactions on Graphics*, 37(1).

P. Jawanpuria, A. Balgovind, A. Kunchukuttan, and B. Mishra. 2019. Learning Multilingual Word Embeddings in Latent Metric Space: A Geometric Approach. *Transactions of the Association for Computational Linguistics*, 7:107–120.

A. Joulin, P. Bojanowski, T. Mikolov, H. Jégou, and E. Grave. 2018. Loss in Translation: Learning Bilingual Word Mapping with a Retrieval Criterion. In *EMNLP*, pages 2979–2984.

Z. Kalal, K. Mikolajczyk, and J. Matas. 2010. Forward-Backward Error: Automatic Detection of Tracking Failures. In *ICPR*, pages 2756–2759.

G. Lample, A. Conneau, L. Denoyer, and M. Ranzato. 2018a. Unsupervised Machine Translation using Monolingual Corpora only. In *ICLR*, pages 1–14.

G. Lample, M. Ott, A. Conneau, L. Denoyer, and M. Ranzato. 2018b. Phrase-based & Neural Unsupervised Machine Translation. In *EMNLP*, pages 5039–5049.

M. Y. Liu and O. Tuzel. 2016. Coupled Generative Adversarial Networks. In *NIPS*, pages 469–477.

J. Long, E. Shelhamer, and T. Darrell. 2015. Fully Convolutional Networks for Semantic Segmentation. In *CVPR*, pages 3431–3440.

F. Mémoli. 2011. Gromov–Wasserstein Distances and the Metric Approach to Object Matching. *Foundations of Computational Mathematics*, 11:417–487.

Q. V. Mikolov, T. Le and I. Sutskever. 2013. Exploiting Similarities among Languages for Machine Translation. arXiv:1309.4168.

T. Mikolov, I. Sutskever, K. Chen, G. S. Corrado, and J. Dean. 2013. Distributed Representations of Words and Phrases and their Compositionality. In *NIPS*, pages 3111–3119.

A. Mogadala and A. Rettinger. 2016. Bilingual Word Embeddings from Parallel and Non-Parallel Corpora for Cross-language Text Classification. In *NAACL-HLT*, pages 692–702.

T. Mohiuddin and S. Joty. 2019. Revisiting Adversarial Autoencoder for Unsupervised Word Translation with Cycle Consistency and Improved Training. In *NAACL-HLT*, pages 3857–3867.

T. Mohiuddin and S. Joty. 2020. Unsupervised Word Translation with Adversarial Autoencoder. *Computational Linguistics*, 46(2):257–288.

A. Myronenko and X. Song. 2010. Point Set Registration: Coherent Point Drift. *Transactions on Pattern Analysis and Machine Intelligence*, 32:2262–2275.

S. Oprea, S. Dutta, and H. Assem. 2020. Unsupervised Word Translation Pairing using Refinement based Point Set Registration. arXiv:2011.13200.

J. Pennington, R. Socher, and C. D. Manning. 2014. GloVe: Global Vectors for Word Representation. In *EMNLP*, pages 1532–1543.

A. Radford, L. Metz, and S. Chintala. 2016. Unsupervised Representation Learning with Deep Convolutional Generative Adversarial Networks. In *ICLR*, pages 1–16.

M. Radovanović, A. Nanopoulos, and M. Ivanović. 2010. Hubs in Space: Popular Nearest Neighbors in High-Dimensional Data. *Journal of Machine Learning Research*, 11:2487–2531.

S. Rusinkiewicz and M. Levoy. 2001. Efficient Variants of the ICP Algorithm. In *3DIM*, pages 145–152.

P. H. Schönemann. 1966. A Generalized Solution of the Orthogonal Procrustes Problem. *Psychometrika*, 31(1):1–10.

G. L. Scott and C. Longuet-Higgins. 1991. An Algorithm for Associating the Features of Two Images. *Royal Society London: Biological Sciences*, 244(1309):21–26.

S. L. Smith, D. H. P. Turban, S. Hamblin, and N. Y. Hammerla. 2017. Offline Bilingual Word Vectors, Orthogonal Transformations and the Inverted Softmax. In *ICLR*, pages 1–10.

A. Søgaard, S. Ruder, and I. Vulić. 2018. On the Limitations of Unsupervised Bilingual Dictionary Induction. In *ACL*, pages 778–788.

G. K. L. Tam, Z. Cheng, Y. Lai, F. C. Langbein, Y. Liu, D. Marshall, R. R. Martin, X. Sun, and P. L. Rosin. 2013. Registration of 3D Point Clouds and Meshes: A Survey from Rigid to Nonrigid. *Transactions on Visualization and Computer Graphics*, 19(7):1199–1217.

C. Tsai and D. Roth. 2016. Cross-Lingual Wikification using Multilingual Embeddings. In *NAACL-HLT*, pages 589–598.

S. Upadhyay, M. Faruqui, C. Dyer, and D. Roth. 2016. Cross-lingual Models of Word Embeddings: An Empirical Comparison. arXiv:1604.00425.

I. Vulić and M. Moens. 2016. Bilingual Distributed Word Representations from Document Aligned Comparable Data. *Journal of Artificial Intelligence Research*, 55(1):953–994.

Z. Wang, J. Xie, R. Xu, Y. Yang, G. Neubig, and J. Carbonell. 2020. Cross-lingual Alignment vs Joint Training: A Comparative Study and A Simple Unified Framework. In *ICLR*, pages 1–15.

J. Xie, Z. Yang, G. Neubig, N. A. Smith, and J. Carbonell. 2018. Neural Cross-lingual Named Entity Recognition with Minimal Resources. In *EMNLP*, pages 369–379.

R. Xu, Y. Yang, N. Otani, and Y. Wu. 2018. Unsupervised Cross-lingual Transfer of Word Embedding Spaces. In *EMNLP*, pages 2465–2474.

Z. J. Yew and G. H. Lee. 2018. 3DFeat-Net: Weakly Supervised Local 3D Features for Point Cloud Registration. In *ECCV*, pages 630–646.

M. Yuan, M. Zhang, B. Van Durme, L. Findlater, and J. Boyd-Graber. 2020. Interactive Refinement of Cross-Lingual Word Embeddings. In *EMNLP*, pages 5984–5996.

A. L. Yuille and N. M. Grzywacz. 1988. The motion coherence theory. In *ICCV*, pages 344–353.

M. Zhang, Y. Liu, H. Luan, and M. Sun. 2017a. Adversarial Training for Unsupervised Bilingual Lexicon Induction. In *ACL*, pages 1959–1970.

M. Zhang, Y. Liu, H. Luan, and M. Sun. 2017b. Earth Mover's Distance Minimization for Unsupervised Bilingual Lexicon Induction. In *EMNLP*, pages 1934–1945.

Mozhi Zhang, Keyulu Xu, Ken-ichi Kawarabayashi, Stefanie Jegelka, and Jordan Boyd-Graber. 2019. Are Girls Neko or Shōjo? Cross-Lingual Alignment of Non-Isomorphic Embeddings with Iterative Normalization. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics (ACL)*, pages 3180–3189.

Y. Zhang, D. Gaddy, R. Barzilay, and T. Jaakkola. 2016. Ten Pairs to Tag – Multilingual POS Tagging via Coarse Mapping between Embeddings. In *NAACL-HLT*, pages 1307–1317.

C. Zhou, X. Ma, D. Wang, and G. Neubig. 2019. Density Matching for Bilingual Word Embedding. In *NAACL-HLT*, pages 1588–1598.

J. Zhu, T. Park, P. Isola, and A. A. Efros. 2017. Unpaired Image-to-Image Translation using Cycle-Consistent Adversarial Networks. In *ICCV*, pages 2242–2251.

A Background of CPD

Point Set Registration algorithms aim to compute the correspondences for aligning two input point sets. Rigid transformation involving rotation, translation and reflection, were used in Iterative Closest Point (ICP) algorithm (Besl and McKay, 1992) and other variants (Rusinkiewicz and Levoy, 2001) for probabilistic alignment. Spectral methods (Scott and Longuet-Higgins, 1991) and closed-form solution for rigid probabilistic registration in multi-dimensional cases was presented in Myronenko and Song (2010). In addition to the rotation, translation and reflection, *affine* transformation also considers scaling, homothety, similarity and shear – providing more degrees of freedom for better point set registration (Ho et al., 2007). Non-rigid transformations are based on Gaussian Mixture model and filters (Hinton et al., 1992; Gao and Tedrake, 2019), Bayesian modeling (Hirose, 2020) or Thin Plate Spline (TPS) parameterization (Bookstein, 1989). Recent developments use convolutional neural networks (Huang et al., 2017) and other learning frameworks (Yew and Lee, 2018). An extensive literature survey can be found in Tam et al. (2013). We adopt Coherent Point Drift (CPD) (Myronenko and Song, 2010) combining Gaussian Mixture Model and Motion Coherence Theory.

***BioSpere* Transform Module.** The *Transform* module performs a refinement on the transformed embeddings X_C and Y_C (obtained from the *Correspond* module) using the concept of *point set registration*. Specifically, we uses the *Coherent Point Drift* (CPD) algorithm (Myronenko and Song, 2010), an unsupervised probabilistic framework which assigns *point-to-point correspondence* between two sets of points, akin to finding word translation pairs in our setting. The idea here is to consider the task of aligning the two embedding spaces as a density estimation problem based on the *Gaussian Mixture Model* (GMM). This considers word embeddings of one language as GMM centroids, and the other embedding space to represent data points. The centroids are then fitted to data points by maximizing the likelihood, and at optimum point correspondences are obtained using GMM posterior probabilities.

Thus, we consider the target embeddings Y_C as

the centroids and the source embedding space X_C as data points, to have been generated by the GMM probability density function. The centroid locations are estimated by Expectation Maximization (EM) algorithm (Dempster et al., 1977).

B Related Background

Generative Adversarial Networks (GANs) couples the training of machine learning architecture between a *generative* and a *discriminative* network that work in tandem for "indirect" training in an unsupervised manner (Goodfellow et al., 2014). GANs have been shown to achieve impressive results in the domain image processing (Zhu et al., 2017), representation learning (Radford et al., 2016) and reinforcement learning (Ho and Ermon, 2016). The task of supervised image-to-image translation involves learning the transformation from an input image to an output image (Long et al., 2015). Unsupervised image-to-image translation approach, Co-GAN (Liu and Tuzel, 2016) was proposed based on weight sharing scheme. Removal of dependencies on task-specific similarity functions and low-dimensionality in this aspect was proposed by Zhu et al. (2017), and was shown in visual tracking by enforcing forward-backward consistency (Kalal et al., 2010). Improving translations via "back translation and reconciliation" is used by human translators (Brislin, 1970). We thus adopt the unsupervised CycleGAN (Zhu et al., 2017) adversarial training based on cycle-consistency loss.

EM-PERSONA: EMotion-assisted Deep Neural Framework for PERSONAlity Subtyping from Suicide Notes

Soumitra Ghosh[1], **Dhirendra Kumar Maurya**[1], **Asif Ekbal**[1] and **Pushpak Bhattacharyya**[2]
[1]Department of Computer Science and Engineering, IIT Patna, India
[2]Department of Computer Science and Engineering, IIT Bombay, India
{ghosh.soumitra2,mauryadhirendra563}@gmail.com, asif@iitp.ac.in, pb@cse.iitb.ac.in

Abstract

The World Health Organization has emphasised the need of stepping up suicide prevention efforts to meet the United Nation's Sustainable Development Goal target of 2030 (Goal 3: Good health and well-being). We address the challenging task of personality subtyping from suicide notes. Most research on personality subtyping has relied on statistical analysis and feature engineering. Moreover, state-of-the-art transformer models in the automated personality subtyping problem have received relatively less attention. We develop a novel *EMotion-assisted PERSONAlity Detection Framework (EM-PERSONA)*. We annotate the benchmark CEASE-v2.0 suicide notes dataset with personality traits across four dichotomies: *Introversion (I)-Extraversion (E), Intuition (N)-Sensing (S), Thinking (T)-Feeling (F), Judging (J)–Perceiving (P)*. Our proposed method outperforms all baselines on comprehensive evaluation using multiple state-of-the-art systems. Across the four dichotomies, *EM-PERSONA* improved accuracy by 2.04%, 3.69%, 4.52%, and 3.42%, respectively, over the highest performing single-task systems.

1 Introduction

Suicide continues to be one of the significant causes of death worldwide (Ghosh et al., 2020). Given the significance of personality as a basis for understanding psychopathology (Krueger and Tackett, 2006) and the variability in risk factors associated with suicide, subtyping patients based on their personalities can provide greater specificity than simple comparisons of suicidal to non-suicidal individuals (Ortigo et al., 2009). Pompili et al. (2008) showed that emotions such as anger, aggressiveness, anxiety, and sadness were associated with personality traits of individuals who attempted suicide.

We quote a few excerpts from suicide notes (SNs) and online personality posts (PPs) in Table 1 to show how personality discussions on public forums and genuine SNs generally follow similar language patterns. We computed cosine similarity (CosSim) scores between SNs (from CEASE-v2.0 corpus (Ghosh et al., 2022)) and PPs (from MBTI dataset[1]) and observed an alarming number of SNs having a considerable amount of word-based similarity with generic PPs. The results are shown in Table 2. We observed CosSim scores over 0.6, 0.5, 0.4 for 12, 39 and 113 SNs, respectively with respect to the PPs in MBTI dataset.

SN-1	*They just do whatever the fuck they want and justify it later.*
PP-1	*They can do anything they like to make any law they like.*
SN-2	*He is an ugly stupid faggot and we should kill him.*
PP-2	*They're oppressing you, kill them all!*

Table 1: Sample excerpts from a couple of SNs and PPs.

CosSim	>0.30	>0.40	>0.50	>0.60
# Notes	204	113	39	12

Table 2: Cosine Similarity scores between suicide notes and personality posts.

Our primary contributions are two-fold: we present a novel corpus of suicide notes annotated with personality traits across four dichotomies and develop an end-to-end multi-task emotion-assisted system for simultaneous detection of these traits from suicide notes.

2 Related Work

The Myers-Briggs Type Indicator (MBTI) (Myers, 1962), based on psychiatrist Carl Jung's ideas, is a popular personality metric that employs four dichotomies as indications of personality traits: Introversion (I) / Extraversion (E), Intuition (N) / Sensing (S), Thinking (T) / Feeling (F), Judging (J) / Perceiving (P). Another popular model like MBTI is the Big Five (Goldberg, 1993) that produces very specific and individual results, which

[1]https://www.kaggle.com/datasnaek/mbti-type

Proceedings of the 29th International Conference on Computational Linguistics, pages 1098–1105
October 12–17, 2022.

can be tedious to draw general insights and advice from test results making the practical application of the knowledge very difficult, especially when the data is scarce (as in the case of suicide note corpus). The fact that the categories for the Big Five Personality Traits are too wide and absolute to offer any meaningful insight is another issue with them. Humans are adaptive beings that adapt to their environment. In situations where we are around close friends, for instance, we could be more open, whilst in foreign settings you might be less open.

Artificial intelligence (AI) is already playing a crucial role in mental healthcare (chatbot: WoeBot, virtual assistant: Ellie (D'Alfonso, 2020)) in handling the increased demand for services, stretched workloads, high costs for treatment, and associated stigma with mental illness (Gamble, 2020). More recently, personality detection studies (Mehta et al., 2020; Yang et al., 2021; Ren et al., 2021) using computational methods have gained traction especially transformer-based (Vaswani et al., 2017) pretrained language models. However, the existing suicide note corpora (Ghosh et al., 2020, 2022) are annotated at the sentence level and existing studies do not exploit the emotional content inherent in them. This motivated us to devise an approach for utilizing the sentence-level information inherent in the existing datasets and address the closely associated tasks at the document level. Moreover, statistical analysis (Ji et al., 2021) and feature engineering (Bharadwaj et al., 2018) have been used in the bulk of the studies on this topic. However, most of the research on personality subtyping has been on domains like essays (Big Five dataset (Pennebaker and King, 1999)) and social media (MBTI Dataset) and none on the domain of suicide. This is the first attempt, to our knowledge, to identify personality subtypes of individuals who have completed suicide.

3 Dataset

We consider the benchmark CEASE-v2.0 dataset[2] (Ghosh et al., 2022) which is a fine-grained emotion annotated suicide notes corpus containing 4,932 sentences from suicide notes, each annotated independently (without any contextual information) with multi-label emotions from 15 fine-grained emotion classes.

3.1 Personality traits annotation

Three annotators[3] sufficiently acquainted with labeling tasks and knowing the concepts of personality profile identification annotated each suicide note. For assistance in understanding the annotation task, the annotators were provided with ample instances for each personality class from the highly popular Myers Briggs Personality Type Test Dataset (MBTI dataset). The annotation task is performed across four dichotomies: *I* or *E*, *N* or *S*, *F* or *T* and *J* or *P*. For a suicide note, annotators categorised a personality trait as *Unclear (U)* if they could not evaluate the correct class owing to a lack of relevant/sufficient information. The final labels were obtained through a majority voting approach on the labels assigned by three annotators.

Traits	Distribution		κ
I-E	I: 285,	E: 71	0.58
N-S	N: 90,	S: 268	0.63
F-T	F: 238,	T: 119	0.66
J-P	J: 145,	P: 214	0.58

Table 3: Data distribution over various personality traits.

The distribution of annotated suicide notes over the various personality trait classes is shown in Table 3. As multiple raters are involved in the annotation process, we employ the Fleiss-Kappa (κ) (Spitzer et al., 1967) measure to compute the agreement among the annotators. We obtain an average Kappa agreement of 0.61 over the four personality dichotomies, indicating substantial agreement among the annotators. The score also indicates the difficulty of perceiving and synthesizing clinical ideas and labeling such tasks. The annotators relatively faced more difficulty in marking with labels *I-E* and *J-P* than *N-S* and *J-P*, which are also reflected in the attained κ scores. In both the MBTI and our annotated dataset, we observe that certain classes such as Introversion and Sensing are overrepresented while Extroversion and Intuition are relatively under-represented.

4 Methodology

Here, we discuss the proposed EMotion-assisted deep neural framework for PERSONAlity Subtyping (*EM-PERSONA*). The overall architecture is illustrated in Fig. 1.

[2]Dataset sourced from: https://www.iitp.ac.in/~ai-nlp-ml/resources.html

[3]all doctoral researchers, one from the computer science discipline, two from the computational linguistics discipline

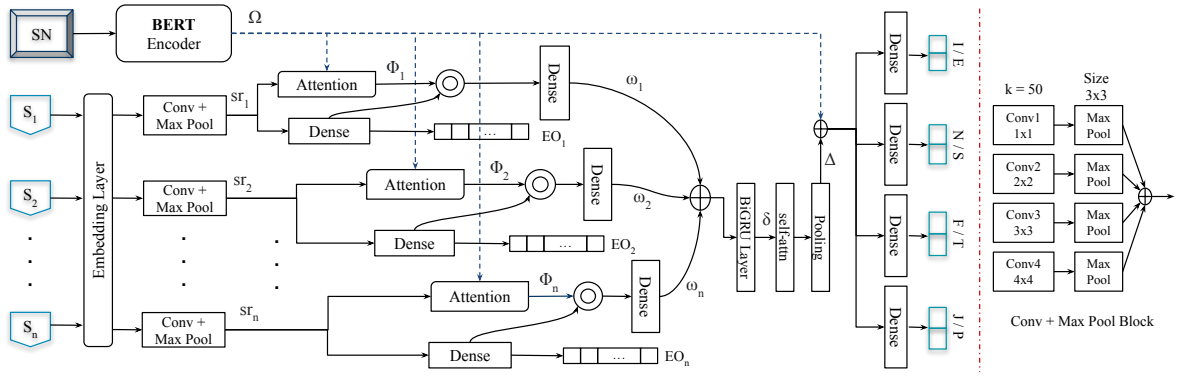

Figure 1: Architecture of the EMotion-assisted deep neural framework for PERSONAlity Subtyping.

4.1 Task Definition

Given a suicide note (N) with each sentence annotated with an emotion class[4], classify the author of the note into one of the two categories for each of the following personality dichotomies: (I/E), (N/S), (F/T), (J/P). Let $N_n^m = (s_1^m, s_2^m, .., s_n^m)$ denote a suicide note with n sentences (s) and $(e_1^m, e_2^m, ..., e_n^m)$ represents the corresponding sentence-level emotion (e) labels in the mth note. The objective is to maximise the value of the following function:

$$\underset{\theta}{argmax}(\Pi_{i=0}^{m} P(y^i_{I-E}, y^i_{N-S}, y^i_{F-T}, y^i_{J-P} | s^i_n, s^i_{n-1}, ..., s^i_1; \theta)) \quad (1)$$

y: output labels, P: log likelihood function, and, θ: model parameters to be optimized.

4.2 EMotion-assisted Deep Neural Framework for PERSONAlity Subtyping (*EM-PERSONA*)

The EM-PERSONA system takes a suicide note documents as input and categorizes the author of the note into four personality classes: I/E, N/S, F/T and J/P. Each training instance comprises of a suicide note document that is encoded using the Bidirectional Encoder Representations from Transformers (BERT) (Devlin et al., 2019) encoder into a *contextualized document representation* (Ω). The individual sentences of the same note are processed in parallel by four convolutional and max pool layers (Conv Max Pool) of the region (k) size 1, 2, 3, and 4 and 50 filters, each of which generates sentence-level feature representations (sr_i). We use convolutional neural networks (CNN) as they are easier to parallelise, faster to train than recurrent neural networks, and effective for short sentences (Hu et al., 2018; Wang et al., 2021) (average sentence length in the CEASE corpus is 15). Word vectors, at the sentence level, are fetched from the pre-trained GloVe (Pennington et al., 2014) embedding.

To produce *contextualized sentence representations* (ϕ_i), we apply additive-attention (Bahdanau et al., 2015) between the sentence representations (sr_i) and the contextualized document representation (Ω). The attention-mechanism can be realized through the following equations:

$$\gamma = W_3^T \tanh(W_1\Omega + W_2 sr_i^c) \quad (2)$$

$$\alpha_i = \frac{exp(\gamma(\Omega sr_i^c))}{\sum_{j=1}^{c} exp(\gamma(\Omega sr_j^c))} \quad (3)$$

$$\phi_i = \sum_{t=1}^{c} \alpha_i sr_i^c \quad (4)$$

where W_1, W_2, W_3 are the learnable weight matrices, $tanh$ is a non-linear function and c is the sentence length in words.

The Conv + Max Pool outputs are also passed through sentence-specific dense layers and corresponding output layers with *softmax* activation to generate emotion classes (EO_i). The intermediate emotion-aware sentence-specific dense representations are added ($\circledcirc$) with the corresponding ϕ_i and passed through a linear layer to produce abstract emotion-aware sentence representations (ω_i).

$$\omega_i = Dense(\phi_i \circledcirc Dense_i(sr_i)) \quad (5)$$

The emotion-aware sentence representations (ω_i) are concatenated ($\oplus$) and passed through a bidirectional gated recurrent unit (BiGRU) (Cho et al., 2014) layer of 100 units to learn the contextual information. We apply multi-head self-attention (Vaswani et al., 2017) (self-attn) to attend to dif-

[4]For simplicity, we consider only the predominant emotion ($Emo1$) from CEASE-v2.0.

ferent parts of the BiGRU output and produce an *contextualized emotion-aware document representation (δ)*, which is then pooled globally.

$$\delta = BiGRU(\omega_1 \oplus \omega_2 \oplus ... \oplus \omega_n) \quad (6)$$

$$\Delta = \text{Pooling}(\text{Trans. Enc.}(\delta)) \quad (7)$$

The pre-trained BERT language model allows us to produce general contextual representations while dealing with a small supervised dataset, avoiding the need to train all the parameters from the start. We linearly concatenate Ω with the pooling layer output, Δ, and pass to four task-specific dense layers followed by the output dense layers with $softmax$ activation to get the output probability p_t^m values over the four personality trait variables.

$$p_t^m = \text{softmax}(W_t(Dense_t(\Delta \oplus \Omega)) + b_t) \quad (8)$$

W and b are learnable weight and bias matrices and t represents the four personality subtyping tasks: *I-E*, *N-S*, *F-T* and *J-P*.

where λ denotes the categorical cross-entropy loss, t represents the four personality traits tasks. α and β are the loss weights for the personality traits (PT) detection tasks and emotion recognition tasks (ER). We limit our experiments to the uniform task weighting approach, i.e., α_t and β_t are both 1.

4.3 Computation of loss

The model is trained by summing the document-level cross-entropy losses of the four personality subtasks, as well as the cross-entropy losses for the sentence-level emotion classification task.

$$\Lambda = \sum_{t=1}^{4} \alpha_t * \lambda_t^{PT} + \sum_{q=1}^{n} \beta_q * \lambda_t^{ER} \quad (9)$$

Models	F1 $^{\text{I-E}}$	F1 $^{\text{N-S}}$	F1 $^{\text{F-T}}$	F1 $^{\text{J-P}}$
Single-task baselines				
HAN	45.4	48.1	43.87	36.6
CNN+cLSTM	44.5	43	44.5	36.1
BERT	44.87	43	39.88	49.36
RoBERTa	44.46	39.88	42.69	50.58
Multitask baseline				
MT-BERT	44.35	42.68	39.90	47.31
Proposed multitask approach				
EM-PERSONA	**47.44**	**51.79**	**49.02**	**54.00**
Ablation Experiment				
EM-PERSONA$_{\text{-Emo}}$	45.53	50.27	46.96	51.40

Table 4: Scores from 10-fold cross-validation experiments. Values in bold are the maximum scores attained.

5 Experiments and Results

In this section, we discuss the experiments performed and the results and analysis.

5.1 Experimental Setup

We evaluate *EM-PERSONA* against five state-of-the-art systems: Hierarchical Attention Networks (HAN) (Yang et al., 2016), Convolutional Neural Network+Context Long Short Term Memory (CNN+cLSTM) (Poria et al., 2017), BERT-Base (Devlin et al., 2019), RoBERTa (Liu et al., 2019) and MT-BERT (Peng et al., 2020). We perform 10-fold cross-validation on the personality annotated CEASE-v2.0 dataset and consider the macro-F1 metric to evaluate our approach against multiple baselines, as class imbalance problem persists in the dataset. We discuss the details of the baselines and the hyperparameters for our experiments in Sections A.1 and A.2 of the **Appendix**.

6 Results and Discussion

Table 4 shows that the proposed *EM-PERSONA* system considerably outperforms all baseline systems, with improvements of 2.04, 3.69, 4.52, and 3.42 points over the best performing single-task systems on the four personality subtasks, respectively. The low F1 scores on the J-P trait over the HAN and CNN+cLSTM single-task baselines align with past research (Lima and de Castro, 2019; Yamada et al., 2019) where predictions on J/P dichotomy consistently underperforms compared to the other dichotomies. This is not the case for the language models, BERT and RoBERTa and the multitask systems (MT-BERT and *EM-PERSONA*), which produces comparable scores across all dichotomies, showing the effectiveness of transformer-based systems and also depicting that the correlations among the various personality traits can be effectively exploited when all the tasks are learned jointly. Commendable performance by the *EM-PERSONA* approach indicates that emotion information plays a crucial role in perceiving the personality traits of an individual through textual content-based analysis.

Ablation study: To test the impact of the emotion-assisting setup, we remove the emotion-specific dense layers in *EM-PERSONA* and see a notable drop in scores across all the personality subtasks (shown in Table 4).

Qualitative Analysis: The first example in Table 5[5] shows the effectiveness of learning the vari-

[5]Reader caution is suggested since the test cases given are

Category	*Note Excerpts*	*Actual*	MT-BERT	*EM-PERSONA*
BL & PP: FC	After many hours of thought and meditation, I have made a decision that should not be an example to anyone else ... Please tell my story on every radio and television station and in every newspaper and magazine ... to those of you who are shallow the events of this morning will be that story ... <NAME>, love you	E S F P	E S F P	E S F P
BL: PC PP: FC	.... If we had a problem it is because I loved her so much. ... we came to the understanding that for now we were not right for each other ... Unlike what has been written in the press, <NAME> and I had a great relationship for most of our lives together ... most of it ... is totally made up.	I S F J	E S F P	I S F J
BL: IC PP: PC	You have always been my soul mate and I want you to love life and know I am always with you. ... your characteristic is that of a true angel and the definition of god's love! This was the supreme Almighty's plan not mine! Look after <NAME> and <NAME> for me they are my boys you are rich. ...	E N T P	I S F J	E S F P
BL & PP: PC	Dear Mum, I am really sorry that I did this. Do not you ever think it was your fault. ... I love you so much and I could not ask for a better mum. Thank you for caring and feeding and loving me for 14 years. ... my heart cannot take this pain . I am going to miss you so much. ... I will be waiting at heaven's gates for you. ...	I S F J	E S T P	E S F P

Table 5: Sample predictions by the MT-BERT and *EM-PERSONA* systems over various categories are shown here. BL: baseline MT-BERT, PP: proposed *EM-PERSONA*, PC: partially correct, FC: fully correct, IC: fully incorrect. I: Introversion, E: Extraversion, N: Intuition, S: Sensing, T: Thinking: F: Feeling, J: Judging, P: Perceiving.

ous personality tasks jointly as both the multitasking systems, MT-BERT and *EM-PERSONA* correctly classified all the traits. In the second example, the *EM-PERSONA* system uses the emotion information in the note to classify all the traits correctly, unlike the MT-BERT system, which could only classify two personality traits correctly.

Error Analysis: The last two examples in Table 5 show some sample predictions from the MT-BERT baseline system and our proposed *EM-PERSONA* system where the models fail to classify the output classes correctly. The relevance of knowing emotion information while attempting to identify various personality traits can be realized from the observations in the third example. Here, we notice that, unlike the MT-BERT system that fails to identify a single personality trait correctly, the *EM-PERSONA* system makes correct predictions on two of the four personality traits. Rigorous analysis of instances where both the multitask systems found difficulty giving correct predictions (as in example 4) indicates that the models have a relatively more challenging time differentiating between I-E and J-P than N-S and F-T.

Test for Significance: We conducted the experiments five times and conducted a student's t-test with a 5% significance level to illustrate that the scores obtained by the proposed system have not happened by chance. We obtain the p-values of 0.039, 0.041, 0.013, and 0.009 compared with the best-performing baselines for each task, indicating that the obtained scores are statistically significant.

from genuine suicide notes and maybe deemed sensitive.

7 Conclusion

Our study focuses on artificial intelligence's assistive role, emphasising that cognitive technology is designed to enhance human intelligence rather than replace it. The proposed method is developed to serve practitioners (computer-aided diagnosis and learning) and individuals (self-monitoring) in their combined effort toward low-profile first-hand evaluation of their personalities. The findings of this study imply that (1) present state-of-the-art methods, both conversational and document encoding methods in general, fail to comprehend personality information in suicide notes to a substantial extent, (2) to improve overall system performance at the document level (such as depression, perceived burdensomeness, and thwarted belongingness), sentence-level information (such as temporal orientation, sentiment, and emotion) can be incorporated into document representations produced by existing transformer architectures, and, (3). large personality traits annotated balanced corpora are required to obtain solid findings, and the introduced resource can facilitate related studies. Identifying key subgroups of people with suicidal inclinations will help us better understand risk factors and therapies based on subtypes.

In future work, we want to address the two major limitations of our study. First, personality traits are not so simple that they can be squeezed into fixed binary categories across four dimensions, as examined in this study. Second, the short context length problem may be addressed by testing with much bigger datasets than the one used in this work.

Ethical Consideration

Our resource creation utilizes publicly available CEASE-v2.0 (Ghosh et al., 2022) benchmark suicide notes dataset. We followed the data usage restrictions and did not violate any copyright issues. This study was also evaluated and approved by our Institutional Review Board (IRB). *The data is available at* `https://www.iitp.ac.in/~ai-nlp-ml/resources.html#EMPERSONA`.

Acknowledgement

Asif Ekbal acknowledges the Young Faculty Research Fellowship (YFRF), supported by Visvesvaraya PhD scheme for Electronics and IT, Ministry of Electronics and Information Technology (MeitY), Government of India, being implemented by Digital India Corporation (formerly Media Lab Asia).

References

Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. 2015. Neural machine translation by jointly learning to align and translate. In *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*.

Srilakshmi Bharadwaj, Srinidhi Sridhar, Rahul Choudhary, and Ramamoorthy Srinath. 2018. Persona traits identification based on myers-briggs type indicator(mbti) - A text classification approach. In *2018 International Conference on Advances in Computing, Communications and Informatics, ICACCI 2018, Bangalore, India, September 19-22, 2018*, pages 1076–1082. IEEE.

Kyunghyun Cho, Bart van Merrienboer, Dzmitry Bahdanau, and Yoshua Bengio. 2014. On the properties of neural machine translation: Encoder-decoder approaches. In *Proceedings of SSST@EMNLP 2014, Eighth Workshop on Syntax, Semantics and Structure in Statistical Translation, Doha, Qatar, 25 October 2014*, pages 103–111.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT 2019, Minneapolis, MN, USA, June 2-7, 2019, Volume 1 (Long and Short Papers)*, pages 4171–4186.

Simon D'Alfonso. 2020. Ai in mental health. *Current Opinion in Psychology*, 36:112–117.

Alyson Gamble. 2020. Artificial intelligence and mobile apps for mental healthcare: a social informatics perspective. *Aslib J. Inf. Manag.*, 72(4):509–523.

Soumitra Ghosh, Asif Ekbal, and Pushpak Bhattacharyya. 2020. Cease, a corpus of emotion annotated suicide notes in english. In *Proceedings of The 12th Language Resources and Evaluation Conference, LREC 2020, Marseille, France, May 11-16, 2020*, pages 1618–1626.

Soumitra Ghosh, Asif Ekbal, and Pushpak Bhattacharyya. 2022. A multitask framework to detect depression, sentiment and multi-label emotion from suicide notes. *Cogn. Comput.*, 14(1):110–129.

Lewis R Goldberg. 1993. The structure of phenotypic personality traits. *American psychologist*, 48(1):26.

Yibo Hu, Yang Li, Tao Yang, and Quan Pan. 2018. Short text classification with A convolutional neural networks based method. In *15th International Conference on Control, Automation, Robotics and Vision, ICARCV 2018, Singapore, November 18-21, 2018*, pages 1432–1435. IEEE.

Shaoxiong Ji, Shirui Pan, Xue Li, Erik Cambria, Guodong Long, and Zi Huang. 2021. Suicidal ideation detection: A review of machine learning methods and applications. *IEEE Trans. Comput. Soc. Syst.*, 8(1):214–226.

Diederik P. Kingma and Jimmy Ba. 2015. Adam: A method for stochastic optimization. In *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*.

Robert F Krueger and Jennifer L Tackett. 2006. *Personality and psychopathology*. Guilford Press.

Ana Carolina ES Lima and Leandro Nunes de Castro. 2019. Tecla: A temperament and psychological type prediction framework from twitter data. *Plos one*, 14(3):e0212844.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Roberta: A robustly optimized BERT pretraining approach. *CoRR*, abs/1907.11692.

Yash Mehta, Samin Fatehi, Amirmohammad Kazameini, Clemens Stachl, Erik Cambria, and Sauleh Eetemadi. 2020. Bottom-up and top-down: Predicting personality with psycholinguistic and language model features. In *20th IEEE International Conference on Data Mining, ICDM 2020, Sorrento, Italy, November 17-20, 2020*, pages 1184–1189. IEEE.

Isabel Briggs Myers. 1962. The myers-briggs type indicator: Manual (1962).

Kile M Ortigo, Drew Westen, and Rebekah Bradley. 2009. Personality subtypes of suicidal adults. *The Journal of nervous and mental disease*, 197(9):687.

Yifan Peng, Qingyu Chen, and Zhiyong Lu. 2020. An empirical study of multi-task learning on BERT for biomedical text mining. In *Proceedings of the 19th SIGBioMed Workshop on Biomedical Language Processing, BioNLP 2020, Online, July 9, 2020*, pages 205–214.

James W Pennebaker and Laura A King. 1999. Linguistic styles: language use as an individual difference. *Journal of personality and social psychology*, 77(6):1296.

Jeffrey Pennington, Richard Socher, and Christopher D. Manning. 2014. Glove: Global vectors for word representation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing, EMNLP 2014, October 25-29, 2014, Doha, Qatar, A meeting of SIGDAT, a Special Interest Group of the ACL*, pages 1532–1543.

Maurizio Pompili, Zoltán Rihmer, Hagop S Akiskal, Marco Innamorati, Paolo Iliceto, Kareen K Akiskal, David Lester, Valentina Narciso, Stefano Ferracuti, Roberto Tatarelli, et al. 2008. Temperament and personality dimensions in suicidal and nonsuicidal psychiatric inpatients. *Psychopathology*, 41(5):313–321.

Soujanya Poria, Erik Cambria, Devamanyu Hazarika, Navonil Majumder, Amir Zadeh, and Louis-Philippe Morency. 2017. Context-dependent sentiment analysis in user-generated videos. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics, ACL 2017, Vancouver, Canada, July 30 - August 4, Volume 1: Long Papers*, pages 873–883.

Zhancheng Ren, Qiang Shen, Xiaolei Diao, and Hao Xu. 2021. A sentiment-aware deep learning approach for personality detection from text. *Inf. Process. Manag.*, 58(3):102532.

Robert L Spitzer, Jacob Cohen, Joseph L Fleiss, and Jean Endicott. 1967. Quantification of agreement in psychiatric diagnosis: A new approach. *Archives of General Psychiatry*, 17(1):83–87.

Nitish Srivastava, Geoffrey E. Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. 2014. Dropout: a simple way to prevent neural networks from overfitting. *J. Mach. Learn. Res.*, 15(1):1929–1958.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, December 4-9, 2017, Long Beach, CA, USA*, pages 5998–6008.

Haitao Wang, Keke Tian, Zhengjiang Wu, and Lei Wang. 2021. A short text classification method based on convolutional neural network and semantic extension. *Int. J. Comput. Intell. Syst.*, 14(1):367–375.

Kosuke Yamada, Ryohei Sasano, and Koichi Takeda. 2019. Incorporating textual information on user behavior for personality prediction. In *Proceedings of the 57th Conference of the Association for Computational Linguistics, ACL 2019, Florence, Italy, July 28 - August 2, 2019, Volume 2: Student Research Workshop*, pages 177–182.

Feifan Yang, Xiaojun Quan, Yunyi Yang, and Jianxing Yu. 2021. Multi-document transformer for personality detection. In *Thirty-Fifth AAAI Conference on Artificial Intelligence, AAAI 2021, Thirty-Third Conference on Innovative Applications of Artificial Intelligence, IAAI 2021, The Eleventh Symposium on Educational Advances in Artificial Intelligence, EAAI 2021, Virtual Event, February 2-9, 2021*, pages 14221–14229.

Zichao Yang, Diyi Yang, Chris Dyer, Xiaodong He, Alexander J. Smola, and Eduard H. Hovy. 2016. Hierarchical attention networks for document classification. In *NAACL HLT 2016, The 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, San Diego California, USA, June 12-17, 2016*, pages 1480–1489.

A Appendix

A.1 Baselines

The following baseline methods are considered for the comprehensive evaluation of our proposed.

- Hierarchical Attention Network (HAN) (Yang et al., 2016): The attention mechanism in HAN takes into consideration the hierarchical structure of texts and identifies the most relevant words in a sentence and most of relevant sentences in a document while taking contextual information into account.
- CNN+cLSTM (Poria et al., 2017): A CNN is used to extract textual characteristics from utterances, after which a cLSTM is used to learn contextual information.
- BERT (Devlin et al., 2019): We experiment with the base version of the state-of-the-art BERT language model by developing four single-task binary BERT classifiers (one for each personality trait variable).
- RoBERTa (Liu et al., 2019): This is an optimized version of BERT trained with more computing power and data than BERT and is known to outperform BERT in many downstream tasks. Similar to BERT, we develop four single-task RoBERTa classifiers.
- MT-BERT (Peng et al., 2020): We build a multitask (MT) variant of BERT based on the

architecture proposed by Peng *et al.* (Peng et al., 2020) for our four personality subtypes.

A.2 Experimental Setting

We set the sequence length as 15 and the context length as 13 as the average sentence length and context length in the CEASE-v2.0 dataset. The experiments are run on an NVIDIA GeForce RTX 2080 Ti GPU. We experiment with the base version of BERT and RoBERTa imported from the Tensorflow Hub (`https://www.tensorflow.org/hub`) library. For maximum utilization of the GPU and considering the small size of the dataset, we run the MT-BERT and *EM-PERSONA* systems with a batch size of 2. Adam optimizer (Kingma and Ba, 2015) is used to train *EM-PERSONA* by minimizing the cross-entropy losses. Through grid search, we set the learning rates as 3e-5 and 2e-5 for the MT-BERT and *EM-PERSONA* systems respectively[6]. We observe empirically that setting higher epochs causes the models to overfit; hence we set the epochs as 3. We use ReLU activation on all dense layers (except the output dense) followed by a dropout (Srivastava et al., 2014) of 25% to prevent overfitting. We employ five self-attention heads for the self-attention layer, embedding dimensions = 200 and feed-forward dimensions = 400. Each task-specific dense layer has 100 neurons, whereas intermediate dense layers contain 200 neurons. To account for the non-determinism of TensorFlow GPU operations, we present F1 scores averaged across five 10-fold cross-validation runs.

[6]we experimented with epochs as 4, 6, and 8 and learning rate as 2e-5, 3e-5.

Dense Template Retrieval for Customer Support

Tiago Mesquita*
Zendesk
Lisbon, Portugal
tiago.mesquita@zendesk.com

Bruno Martins
INESC-ID & LUMLIS (Lisbon ELLIS Unit)
IST, University of Lisbon
Lisbon, Portugal
bruno.g.martins@inesc-id.pt

Mariana Almeida
Zendesk
Lisbon, Portugal
mariana.almeida@zendesk.com

Abstract

Templated answers are used extensively in customer support scenarios, providing an efficient way to cover a plethora of topics, with an easily maintainable collection of templates. However, the number of templates is often too high for an agent to manually search. Automatically suggesting the correct template for a given question can thus improve the service efficiency, reducing costs and leading to a better customer satisfaction. In this work, we propose a dense retrieval framework for the customer support scenario, adapting a standard in-batch negatives technique to support unpaired sampling of queries and templates. We also propose a novel loss that extends the typical query-centric similarity, exploiting other similarity relations in the training data. Experiments show that our approach achieves considerable improvements, in terms of performance and training speed, over more standard dense retrieval methods. This includes methods such as DPR, and also ablated versions of the proposed approach.

1 Introduction

A very common practice to make customer support more efficient is the use of templates for replies. The templates of replies are designed by customer support administrators to systematise the reply to frequent requests. Then, given a customer's request, an agent can pick a response from within a collection of predefined templates, this way saving time when replying to repetitive questions. Besides improving the throughput of human agents, the use of templates also assures uniformity in the handling of different customers, as requests with the same underlying problem should be handled with the same type of reply. However, customer support centers can have hundreds of templates, and finding the best template for a question is not an easy task, particularly for unexperienced agents. Automatically sorting and suggesting customer support templates (Bonatti et al., 2016; Yang and Kwok, 2012; Sneiders et al., 2016; Rei, 2019) can facilitate agent's work, reducing reply times, accelerating the learning curve of new agents, helping agents to focus on more added valued tasks, and overall providing a better customer support at reduced costs.

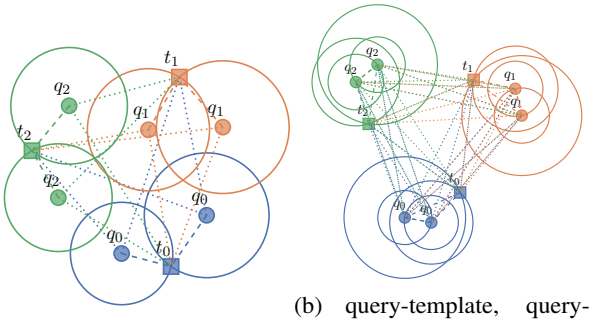

(a) query-template

(b) query-template, query-query and template-template

Figure 1: Illustration of 6 query representations $\{q_0, q_0, q_1, q_1, q_2, q_2\}$, together with respective template representations $\{t_0, t_1, t_2\}$ with t_i answering q_i, after enforcing different similarity relations. The distances between points represent dissimilarity, and dashed lines and circumferences represent similarity relations, whilst the dotted lines represent dissimilarity relations.

The recent advances in large pre-trained language models (Devlin et al., 2019; Vaswani et al., 2017), together with their successful use in question-answering (Karpukhin et al., 2020; Qu et al., 2021) and information retrieval (Xiong et al., 2021; Zhan et al., 2020, 2021; Yates et al., 2021), motivate the use of dense retrieval for template selection. Dense retrieval can be used to rank instances from the template collection, facilitating the selection of the correct template. Still, template ranking has specific characteristics when compared with more common retrieval scenarios in the literature: i) we have a strict many-to-one relation between queries and templates, in contrast to other common retrieval tasks (Nguyen et al., 2016); ii) the collection of templates is relatively small and

*This work was developed at Cleverly and Zendesk, under the context of Tiago Mesquita's Master of Science (M.Sc.) Thesis at IST, University of Lisbon.

Proceedings of the 29th International Conference on Computational Linguistics, pages 1106–1115
October 12–17, 2022.

generally in the order of hundreds, although also dynamically updated over time; iii) the length of the queries (emails with questions from customers) tends to be relatively long.

Given real-time and computation constraints in the template suggestion problem, we focus on bi-encoder models (Yates et al., 2021), which at prediction time only need to compute dense representations of queries and make fast comparisons with pre-computed representations of template candidates. We discard cross-encoder models that, despite often achieving higher retrieval performance (Yates et al., 2021), can have problems processing long queries and/or documents, and cannot take advantage of pre-computed template representations, involving higher computational costs that scale with the number of template candidates, and being often unsuitable for real-time applications. Still, our contributions are model independent, being also applicable to cross-encoders. Specifically, we make the following main research contributions:

- We compare different approaches in the task of template retrieval in customer support;
- We propose a new in-batch sampling strategy, that preserves the distributions of queries and templates to better select the information within batches, while exploring all possible query-template pairs in a batch;
- We propose a new loss function that exploits not only query-template similarity relations, but also query-query and template-template relations, yielding better representations for retrieval (see Figure 1).

Experiments with real-world customer support datasets show that both the in-batch sampling strategy and the expanded loss lead to improvements, in terms of template suggestion and training speed.

2 Related Work

Template retrieval for customer support has seen limited research in recent times. Most previous work has addressed the task as template classification with simple machine learning approaches (e.g., support vector machines or naïve Bayes) on top of representations obtained from bags-of-words (Bonatti et al., 2016; Yang and Kwok, 2012) or tailored pattern matching (Sneiders et al., 2016). A combination of retrieval and generative approaches is explored by Rei (2019), but without taking advantage of modern Transformer-based pre-trained language models. To the best of our knowledge, public literature on the topic has not explored recent advances in dense retrieval.

Most of the recent dense retrieval methods follow a BERT-based dual-encoder architecture and use a similarity function (e.g., cosine similarity) to produce ranking scores (Yates et al., 2021). The simplicity of the similarity function is crucial, allowing efficient retrieval through recent developments in Approximate Nearest Neighbour (ANN) search, such as FAISS (Johnson et al., 2019). These methods enable search speeds comparable with simpler sparse retrieval methods (e.g., BM25 (Robertson and Zaragoza, 2009)), whilst retaining better performance in most scenarios. Given the simplicity of dual-encoder architectures, most research has focused on improving the training procedure, namely by careful selection of the query-document pairs. For each query, the model should maximize similarity with all related documents (i.e., positives), whilst minimizing it for all unrelated documents (i.e., negatives). Most approaches have focused on the problem of selecting negative documents, usually falling under one of two categories: 1) efficient batching techniques, or 2) techniques that prioritize batch generation but generally sacrifice training efficiency.

In the first category, methods aim to maximize the amount of negatives available within the batches. The most efficient way to achieve this is by sharing negatives between all queries in the batch, an approach known as in-batch negatives (Karpukhin et al., 2020). More recent studies have proposed sharing negatives between multiple GPUs, allowing for massive batch sizes under parallel model training (Qu et al., 2021).

Although the previous techniques maximize the number of negatives seen during training, most of the instances are easily distinguishable, providing weak contributions to the loss function (i.e., easy negatives). Studies under the second category focus on the careful selection of negatives per query, looking for those that are useful for learning (i.e., hard negatives). The earliest approaches leveraged other retrieval methods to pool hard negatives, namely BM25 (Robertson and Zaragoza, 2009), picking highly ranked although irrelevant documents (Karpukhin et al., 2020; Xiong et al., 2021). Despite being effective at picking static

hard negatives, these approaches fail to adapt, as the model learns to rank the instances accordingly. ANCE (Xiong et al., 2021) addresses this issue by using the model itself to pool hard negative examples, dynamically adjusting the selection as the training progresses. In practice, dynamically generating and indexing embeddings for large document collections is costly, requiring separate GPUs to periodically maintain and refresh the index with prior checkpoints, whilst training in parallel. LTRe (Zhan et al., 2020) and later ADORE (Zhan et al., 2021) further refined these ideas, by freezing the weights of the document encoder and eliminating the need to refresh the index.

Although most approaches generally focus on a single category, some have tried to leverage techniques from both. DPR (Karpukhin et al., 2020) was one of the first studies to explore this idea, combining randomly pooled in-batch negatives with BM25 hard-negatives per query. Recently, STAR (Zhan et al., 2021) took the idea even further, by using static hard negatives pooled from the pre-trained model, but sharing them between all queries in the batch, similarly to in-batch negatives. Results showed that the combination provides an effective tool for stabilizing the biases introduced by the use of static negatives. Our technique also combines ideas from both categories, adapting in-batch negatives to the specific scenario of template retrieval, and performing hard negative sampling from all the in-batch negative examples.

Most previous studies on dense retrieval have generally also considered loss functions that enforce query-centric similarity relations, as these are explicitly related to the retrieval task. However, as shown in PAIR (Ren et al., 2021), models may benefit from exploring passage-centric similarity relations, potentially improving the representations. More recently, Li et al. (2021) proposed DANCE, which showed improvements by considering a loss function that combines query retrieval and document retrieval tasks. In our template retrieval scenario, these ideas are further explored, by taking into account all four possible relations between pairs of queries and templates.

3 Simple Dense Template Retrieval

We formally define the problem of template retrieval as follows: given a query q, the model must retrieve, from a collection of templates, the template t that better answers the query.

3.1 Architecture

Let us consider the commonly used dual-encoder architecture, as presented in DPR (Karpukhin et al., 2020), in which two independent encoders $E_Q(\cdot)$ and $E_T(\cdot)$ encode a query q and a template t into d-dimensional vectors, with different representation spaces. For ranking the templates, the cosine similarity between a query q and a template t can be computed from the respective representations:

$$\mathrm{s}(q,t) = \text{cosine-sim}\left(E_Q(q), E_T(t)\right). \quad (1)$$

3.2 Loss Function

The loss function for training the encoders should maximize the similarity between positive query-template pairs $s(q,t^+)$ and minimize the similarity between negative query-template pairs $s(q,t^-)$. A commonly used loss term for retrieval tasks is the negative log likelihood, comparing the positive template t^+ against a set of negative templates $\mathcal{T}^-$:

$$\mathcal{L}_q(q,t^+,\mathcal{T}^-) = -\log\left(\frac{e^{s(q,t^+)}}{e^{s(q,t^+)}+\sum_{t^-\in\mathcal{T}^-} e^{s(q,t^-)}}\right). \quad (2)$$

The final loss is then obtained by averaging the per-query loss from (2) over all queries (and template lists) considered in a batch from the dataset.

3.3 In-batch Negatives

Selecting good negative examples for training dense retrievers is still an open problem. Simple in-batch negatives, as described in DPR (Karpukhin et al., 2020), makes optimal use of the batch space, by sampling query-passage positive pairs and considering, for each query, all other passages within the batch as negatives. However, hidden in its simplicity lie two important assumptions: 1) the in-batch negatives are in fact negative passages; 2) the shared negatives provide a good estimation of instances within the full dataset. The weight of both assumptions is small for large corpora, where each document has a limited amount of related queries and vice-versa, making false in-batch negatives unlikely. Still, for smaller corpora such as those from customer support with templates, the assumptions can be problematic, requiring careful selection of the training pairs.

4 Improved Dense Template Retrieval

To improve on the method outlined in the previous section, we propose two innovative contributions.

The first relates to the in-batch sampling strategy, whilst the second refers to an expanded loss function that exploits different similarity relations for queries and templates.

4.1 Batch Generation

Template retrieval relates queries and templates in a strictly many-to-one correspondence, at the same time involving a small template collection. Moreover, since templates see different use, the number of queries per template varies considerably. These characteristics actively challenge the assumptions of vanilla in-batch negatives. In order to guarantee that the in-batch negatives are in fact negative, the sampled pairs must have different templates. This condition influences the distribution of training examples, penalizing frequently used templates and resulting in a distribution of negatives within the batch that does not follow the real data distribution.

4.1.1 Labeled In-batch Negatives

Given that each query has a single related template, labelling each text (i.e., query or template) in a training batch with the corresponding template identifier provides sufficient information to create all valid positive and negative pairs. More specifically, let t_i be the i-th template and $q_{i,j}$ be the j-th query from the sub-collection of queries that is answered by t_i. Given a batch of N_Q queries and N_T templates, with each text labeled with the corresponding template index i, we consider for each query $q_{i,j}$ the template t_i as positive, and all other templates t_n within the batch, with $n \neq i$, as negatives. This technique, wich we refer to as labeled in-batch negatives, not only prevents in-batch false negatives, but also eliminates the paired sampling restrictions (i.e., the training examples do not have to be explicit query-template pairs) imposed by vanilla in-batch negatives.

4.1.2 A Semi-independent Query-Template Sampling Strategy

As a general rule, we assume that training instances should follow 2 principles: 1) uniform sampling of positive pairs, since these offer explicitly labeled relevance information that should be uniformly explored; 2) sampling negatives according to a distribution that is consistent with the corpus. Vanilla in-batch negatives fails to follow both principles, as the distribution of negatives within the batch follows the distribution of templates available in the positive pairs, and not the real one. With labeled in-batch negatives, on the other hand, positives and negatives are not directly tied, enabling the consideration of both principles. To respect them, whilst maximizing the utility of the instances within the batch, we devised a semi-independent query-template sampling strategy, according to Algorithm 1 (and illustration in Figure 2).

Algorithm 1 Semi-independent query-template sampling procedure

Data: Let $E_{\mathcal{Q}}$ and $E_{\mathcal{T}}$ contain training queries and templates

Result: batch B, with b queries and b templates

1: Get set $\mathcal{T}$ with b templates uniformly selected from $E_{\mathcal{T}}$
2: Get set $\mathcal{Q}_{\mathcal{T}} \in E_{\mathcal{Q}}$ with the queries answered by $\mathcal{T}$
3: Get set $\mathcal{Q}$, by randomly sampling b queries from $\mathcal{Q}_{\mathcal{T}}$
4: Compose B, with the queries and templates in $\mathcal{Q}$ and $\mathcal{T}$

Note that unlike standard in-batch negatives, adapting this algorithm to support batches with a different number of templates and queries (i.e., batches of b_q queries and b_t templates, with $b_q \neq b_t \neq b$) is trivial. Still, we consider $b_q = b_t$ to be the most natural way to explore the information within a batch, as each query has a single positive template. This also mirrors the setup from the standard in-batch negatives approach.

4.2 Expanded Loss Function

We also propose a novel loss function that is expanded at the batch level, considering interactions not only between query-template pairs, but also query-query, template-template, and template-query pairs (see Figure 1). For that, let us first notice that each text in a batch (which can be either from a query or a template) is given a label corresponding to the correct template. The loss function for each batch can be defined with basis on the following generic loss term that takes two sets ($\mathcal{A}$ and $\mathcal{B}$) of labeled texts (that can be queries or templates) from a training batch:

$$\mathcal{L}(\mathcal{A},\mathcal{B}) = \\ -\frac{1}{|\mathcal{A}|}\sum_{i\in\mathcal{I},a_i\in\mathcal{A}_i}\frac{1}{|\mathcal{B}_i|}\sum_{b_i\in\mathcal{B}_i}\log\left(\frac{e^{s(a_i,b_i)}}{e^{s(a_i,b_i)}+\sum\limits_{j\neq i,b_j\in\mathcal{B}_j}e^{s(a_i,b_j)}}\right), \quad (3)$$

where $\mathcal{I}$ is the set of all labels in the batch, while $\mathcal{A}_i$ is the set of texts in $\mathcal{A}$ that have label i (i.e.,

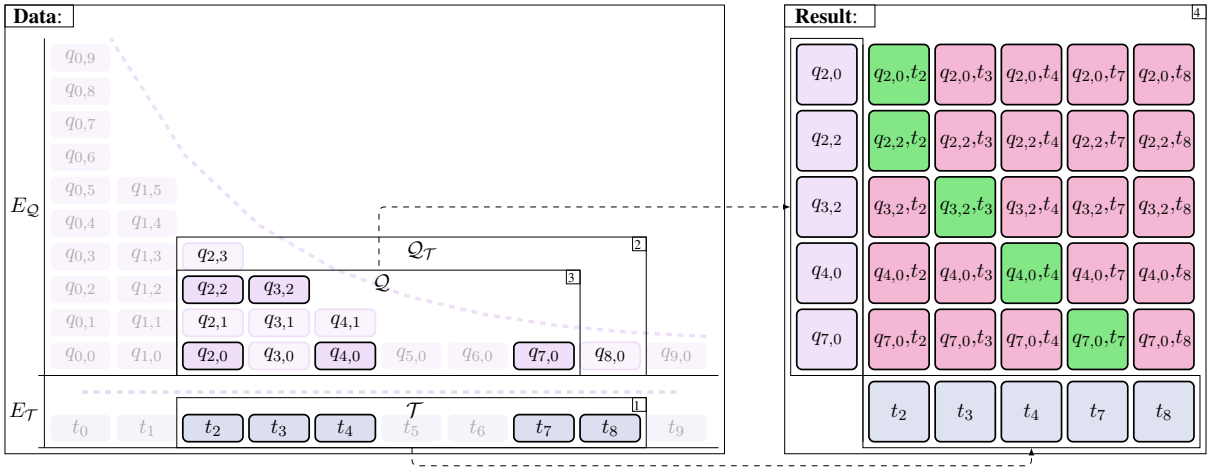

Figure 2: An illustrative case for the application of the semi-independent query-template sampling procedure (Algorithm 1) on an example training dataset. The illustrated collection contains templates $E_{\mathcal{T}}$ and queries $E_{\mathcal{Q}}$, where each query is represented above the template that answers it. Each step of the algorithm is represented with numbers and rectangles indicating the corresponding sets. The procedure results in the creation of batch B, and it is followed by the application of the labelled in-batch negatives technique, yielding the matrix of query-template pairs represented on the right. In the matrix, the positive pairs are shown in green, and the negative ones in red.

those that correspond to template i), and $\mathcal{B}_i$ is the set of texts in $\mathcal{B}$ that have label i.

4.2.1 Combining Different Loss Terms

The final loss of a batch is given by a weighted sum of four terms, each of them computed through (3):

$$\mathcal{L}_{\text{batch}} = \alpha\,\mathcal{L}(\mathcal{Q},\mathcal{T}) + \beta\,\mathcal{L}(\mathcal{Q},\mathcal{Q}) + \gamma\,\mathcal{L}(\mathcal{T},\mathcal{T}) + \theta\,\mathcal{L}(\mathcal{T},\mathcal{Q}), \quad (4)$$

where α, β, γ, and θ are adjustable hyper-parameters, and where the different loss terms correspond to different relations as follows:

1. $\mathcal{L}(\mathcal{Q},\mathcal{T})$; $\mathcal{A}=\mathcal{Q}$ is the set of all queries in a batch and $\mathcal{B}=\mathcal{T}$ is the set of all templates. This term corresponds to averaging the loss of each query $q \in \mathcal{Q}$, using the negative log likelihood of the positive template (2) combined with each possible negative template in the training batch;

2. $\mathcal{L}(\mathcal{T},\mathcal{T})$; $\mathcal{A}=\mathcal{T}$ and $\mathcal{B}=\mathcal{T}$ both correspond to the set of all templates in the batch. This term enforces the dissimilarity between distinct templates;

3. $\mathcal{L}(\mathcal{Q},\mathcal{Q})$; $\mathcal{A}=\mathcal{Q}$ and $\mathcal{B}=\mathcal{Q}$ both correspond to the set of all queries in the batch. This term enforces the dissimilarity between query representations from different templates, and promotes the similarity of representations for queries from the same template;

4. $\mathcal{L}(\mathcal{T},\mathcal{Q})$; $\mathcal{A}=\mathcal{T}$ is the set of all templates in a batch and $\mathcal{B}=\mathcal{Q}$ is the set of all queries in the batch. This term is the transpose of the first one, having a similar effect but acting on each template instead of each query;

4.2.2 In-batch Top-k Negatives

Section 2 discussed recent methods that use the model's own representations to guide the selection of hard-negatives (Xiong et al., 2021; Zhan et al., 2020, 2021). Although potentially effective, these techniques are computationally more demanding than the ones we propose, missing our efficiency goals. Still, inspired by these approaches, we consider a cheaper alternative of in-batch top-k negatives, that instead of retrieving the top-k negatives over the entire corpus, retrieves them from within the batch. By reusing the representations within a batch, this approach is much cheaper while also guaranteeing that the representations are synchronous. Unlike ANCE (Xiong et al., 2021) and the other methods, the value of selecting the in-batch top-k negatives is not on the selected hard negatives, since they are already present in the batch, but in discarding some instances. This delays over-fitting on simpler negatives, allowing the model to learn the harder ones.

5 Experiments

This section describes the experimental validation of the proposed contributions.

dataset	#queries			#t	$P_{80\%}$#tok		lang
	train	val	test		q	t	
CS-1	17858	3127	3918	445	113	396	en
CS-2	1092	187	650	82	111	164	pt

Table 1: Statistics for the two datasets. #t indicates the number of templates, and #tok indicates the text length in terms of multilingual DistilBERT$_{\text{base}}$ tokens.

5.1 Datasets and Metrics

We tested our approach on two private anonymized real-world datasets of email customer support interactions, in English and Portuguese.

The datasets are composed from a collection of real customer support interactions over email, where a human agent handpicked a template for answering a given query. The queries are user submitted and do not follow any particular formatting guidelines, ranging from typical emails (composed by a greeting, a body of text and a sign-off), to direct free-form questions. Templates of replies have more structure, following typical email conventions. Typically, these include trouble-shooting steps, clarifying information, notifications of hand-offs to other support agents/mediums, or a combination of these. The intents of the frequent requests that are answered with templates include login issues, password resets, after sale support, technical support, clarifications for products or promotions, complaints, recommendations, and identity verification, among others.

We split both datasets into 3 partitions, namely *train*, *val*, and *test*. The *test* split is composed of the most recent customer interactions, simulating the real temporal evaluation scenario, whilst the *train* and *val* splits are composed of the remaining examples on a 85/15 stratified split (see Table 1 for a characterization of the datasets).

The CS-1 and CS-2 datasets consider two different types of real world conditions. CS-1 has a relatively large size, and a large template collection, providing better training conditions and more representative results. On the other hand, CS-2 has a smaller, more specific, template collection, featuring non-English data (i.e., questions and templates in Portuguese) and fewer training examples.

5.1.1 Evaluation Metrics

We adopted the Recall@k (R@k) and Mean Reciprocal Rank (MRR) metrics for comparing models. MRR calculates the averaged reciprocal rank of the correct template, while R@k measures the ratio of queries in which the correct template is within the top-k. In particular, we track MRR@10 as a general indicator of ranking performance, and R@3 for matching the use-case of showing only the top-3 templates to agents.

Besides retrieval quality metrics, we also recorded the number of epochs involved in model fine-tuning, considering a early-stopping criterion based on MRR@10 over a validation split.

5.2 Experimental Setup

We now describe the approaches under comparison. Notice that we focused our analysis on pre-trained/fine-tuned multilingual models, envisioning real world scenarios where clients operate with different domains and languages.

5.2.1 Unsupervised Baselines

As a sparse retrieval baseline we consider a traditional BM25 (Lin et al., 2021) approach. For a dense retrieval baseline, we tested all multilingual models in Sentence-Transformers (Reimers and Gurevych, 2020), in a 0-shot manner, and report results for the best: `distiluse-base-multilingual-cased-v1`.

5.2.2 Fine-tuned Models

We tested a baseline approach based on randomly selected negative samples, similar to DPR (Karpukhin et al., 2020), as well as different ablated versions of our improved dense retrieval method. Both encoders on the dense retrievers (i.e., the query and template encoders) were initialized with the parameters of the `distiluse-base-multilingual-cased-v1` model from the Sentence-Transformers library (Reimers and Gurevych, 2020), as this was the best model in 0-shot retrieval experiments.

In more detail, and besides the complete proposed approach, we considered 5 other dense retrieval settings corresponding to the use of the vanilla loss ($\mathcal{L}(\mathcal{Q}, \mathcal{T})$) together with different mechanisms to construct the negative instances:

- **Random negatives**: randomly sample N negative templates for each query-template pair, as in the random sampling scheme described in DPR (Karpukhin et al., 2020);
- **In-batch neg^q**: sample B templates, weighed by frequency of positive queries and without

Methods	CS-1			CS-2		
	MRR@10	R@3	Epochs	MRR@10	R@3	Epochs
Unsupervised Baselines						
BM25	8.5 ± 0.0	10.1 ± 0.0	-	13.7 ± 0.0	16.6 ± 0.0	-
SBERT 0-shot	10.2 ± 0.0	12.0 ± 0.0	-	16.2 ± 0.0	20.2 ± 0.0	-
DPR (Karpukhin et al., 2020)						
Random negatives	39.4 ± 0.2	45.0 ± 0.4	25.0 ± 1.9	63.5 ± 1.9	73.6 ± 1.6	15.2 ± 2.2
Sampling Ablation						
In-batch neg^{q}	33.0 ± 0.8	37.9 ± 1.1	18.2 ± 6.0	61.6 ± 1.0	70.7 ± 1.2	20.2 ± 4.9
In-batch neg^{t}	38.0 ± 0.2	44.7 ± 0.4	22.2 ± 6.4	64.9 ± 1.6	75.0 ± 1.6	20.2 ± 6.4
Labeled in-batch neg^{q}	39.0 ± 0.6	45.2 ± 0.3	3.0 ± 0.7	63.2 ± 1.7	72.2 ± 1.0	10.5 ± 8.4
Labeled in-batch neg^{t,q}	‡41.1± 0.9	‡46.9 ± 0.9	4.5 ± 1.1	†65.2 ± 1.6	†75.4 ± 1.2	5.8 ± 0.8
Proposed approach	‡**42.2 ±0.3**	‡**48.2 ± 0.3**	3.8 ± 0.8	‡**65.4 ± 0.8**	†**75.7 ± 1.0**	8.0 ± 1.6

Table 2: Results for both datasets, including mean and variance from 4 runs per model with different seeds. † and ‡ indicate significant improvements over the random negatives baseline, with p-values of 0.05 and 0.01, respectively, using the permutation test from Bassani (2022).

repetition, and a positive query for each of the sampled templates;

- **In-batch neg^{t}**: sample B templates, uniformly and without repetition, along with a positive query for each;

- **Labeled in-batch neg^{q}**: sample B queries, uniformly and without repetition, along with each respective template. If this produces repeated templates, we swap them with uniform samples not present in the batch;

- **Labeled in-batch neg^{t,q}**: corresponds to the complete version of the proposed sampling technique, as described in Section 4;

On what regards the hyper-parameters considered for model training, we used a batch-size $B = 32$ in all experiments with in-batch negatives, and $B = 8$ for the experiment with random negatives, with $N = 4$ negatives for each instance (i.e., sharing the negatives can improve space efficiency, and the choice of batch-size depended on the maximum capacity of the GPU used for the experiments, namely a NVIDIA Tesla T4 with 16GB of RAM). We used linear learning-rate scheduling with 500 warmup steps, and the ADAM optimizer (Kingma and Ba, 2015) with a learning-rate of 3e-5. We considered a maximum 30 epochs for the linear scheduler, stopping earlier if MRR@10 over the validation set stopped improving.

5.3 Experimental Results

Table 2 presents the obtained results, from which we can infer the following main conclusions:

1. The proposed sampling technique not only significantly outperforms all the alternative methods in both datasets, but it does so with considerably less training epochs. This result confirms the intuition that the common sampling strategies for IR fail to correctly model the one-to-many relation between user questions and templates.

2. The proposed loss, that also considers template-template and query-query similarity relations, improves the model further, yielding significant gains in terms of both the average performance and the corresponding variance. This result suggests that exploring semantic relations beyond the main ranking task is beneficial, likely being a result of learning more robust representations with better generalization capabilities, along with more stable training.

3. The poor performance of BM25 exposes the difficulty of the template retrieval task. Since each template covers a range of queries, the text is generally unspecific, resulting in reduced term overlap between templates and queries. Trained dense retrievers, on the other hand, were able to achieve good performance, showing that semantic relations are effectively superior to simple term matching.

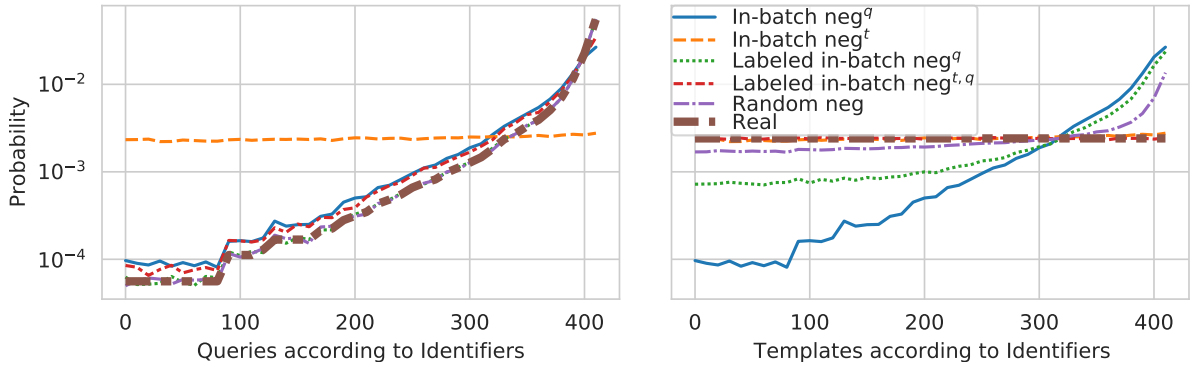

Figure 3: Comparison between the real and observed distributions, for the probability of occurrence of template identifiers in association to instances, obtained with different sampling techniques during model training, for queries (LEFT) and templates (RIGHT) on CS-1. The template identifiers associated to queries and templates are ordered by the real distribution and we plot the mean over bins of 10 identifiers, to reduce the number of data points and generate smoother lines that are easier to interpret.

4. The good results of the proposed techniques on CS-2 further validate our ideas, by proving the robustness to training conditions involving less data. It also validates the applicability of the proposed architecture in other languages. This is expected, as the approach simply refines the representation space of the base model. Given the multilingual pre-training, the approach is effectively able to transfer multilingual knowledge.

5.3.1 Analysis on the Sampling Techniques

The experiments reported in Table 2 also compared the different sampling techniques. To provide better insights over the practical differences between each method, we plotted the distributions of templates and queries, throughout training, for each technique. In order to do this, we recorded the template identifiers of the sampled queries and templates, at each step, for a total of 10 epochs in CS-1. The result is presented in Figure 3, which confirms the intuitions behind the design of the proposed sampling technique.

As expected, with vanilla in-batch negatives, queries and templates follow the same distributions, as they are sampled in pairs. This results in techniques that are only capable of optimizing the distribution of templates (i.e., **in-batch neg**^{t}) or queries (**in-batch neg**^{q}), but not both, resulting in sub-optimal performance. Labeled in-batch negatives are effectively able to decouple both distributions, fact that is key for providing good approximations for both templates and queries.

Labeled in-batch neg^{q} provides a good estimation over the distribution of queries, although the observed distribution of templates is slightly biased towards the most frequent. This results from the query-guided sampling technique, which can explain the slightly worse performance.

Labeled in-batch neg^{t,q}, on the other hand, is able to provide a uniform distribution of templates, whilst maintaining a distribution of queries very close to the real one. This provides, by far, the best global fit of both the distributions, resulting in the best overall performance.

The **random negatives** strategy, despite selecting templates on a per-query basis, is still slightly biased towards the most frequent templates, a result of the positive examples still following the query distribution. This, coupled with the reduced number of negatives, are likely the main factors for the lower performance.

Overall, the results seem to imply a correlation between the quality of the sampling techniques as an estimator of the involved distributions, and the observed retrieval performance of the strategy.

5.3.2 Analysis on the Loss Terms

The proposed loss function combines different terms, each enforcing a different similarity relation. In order to assess the contribution of each component, along with their interaction, we tested 5 different combinations of terms:

- $\mathcal{L}_1 = \mathcal{L}(\mathcal{Q}, \mathcal{T})$: control experiment, considering the negative log likelihood over the positive template;
- $\mathcal{L}_{2T} = \mathcal{L}(\mathcal{Q}, \mathcal{T}) + \mathcal{L}(\mathcal{T}, \mathcal{T})$: considers equal contribution of template-template and query-template relations;

Loss	W/o top-k neg		W/ top-k neg	
	MRR@10	R@3	MRR	R@3
$\mathcal{L}_1$	41.1 ± 0.9	46.9 ± 0.9	40.5 ± 1.4	47.1 ± 1.3
$\mathcal{L}_{2T}$	39.7 ± 0.9	45.8 ± 0.7	40.4 ± 0.5	46.8 ± 1.1
$\mathcal{L}_{2Q}$	‡**41.7** $\pm$ **0.5**	‡**47.7** $\pm$ **1.0**	$^{\ddagger}41.8 \pm 0.3$	$^{\ddagger}47.9 \pm 0.4$
$\mathcal{L}_3$	40.9 ± 0.6	46.8 ± 0.6	***42.2** $\pm$ **0.3**	***48.2** $\pm$ **0.3**
$\mathcal{L}_4$	41.4 ± 0.8	**47.6** $\pm$ **1.3**	41.5 ± 0.3	***48.3** $\pm$ **0.5**

Table 3: Ablation study on the terms of the loss and the in-batch top-k sampling, on the CS-1 dataset, including mean and variance intervals from 4 runs per model with different seeds. ‡ and * indicate significant improvements with p-values of 0.01 and 0.001, respectively, over the control experiment ($\mathcal{L}_1$, without top-k), according to the permutation test from Bassani (2022).

- $\mathcal{L}_{2Q} = \mathcal{L}(\mathcal{Q}, \mathcal{T}) + \mathcal{L}(\mathcal{Q}, \mathcal{Q})$: considers equal contribution of query-template and query-query relations;
- $\mathcal{L}_3 = \mathcal{L}(\mathcal{Q}, \mathcal{T}) + 0.5(\mathcal{L}(\mathcal{Q}, \mathcal{Q}) + \mathcal{L}(\mathcal{T}, \mathcal{T}))$: combines the query-template relations with equally contributing template-template and query-query relations;

For each of the considered losses, we also test the impact of using in-batch top-k negatives . We selected values for k experimentally, resulting in $k = 4$ for CS-1 and $k = 12$ for CS-2, with values below often leading to less stable training, and values above decreasing model performance. The results are presented in Table 3.

In agreement to what was reported in PAIR (Ren et al., 2021), the loss that combines query-template and template-template relations ($\mathcal{L}_{2T}$) is the one with lowest performance, suggesting some misalignment with the retrieval task. Combining query-template and query-query relations ($\mathcal{L}_{2Q}$) improves performance with respect to the control loss, and the combination of both ($\mathcal{L}_3$) improves it further, outperforming all others significantly, and suggesting complementarity of the two terms. It is important to note that the proposed loss function is effective without carefully tuning the contributions of each term (i.e., we only considered the configurations mentioned in Table 3), though tuning can perhaps improve performance even further.

The in-batch top-k sampling strategy improved performance and variance consistently, except for the simple query-template loss. This result suggests that the integration of the top-k sampling technique in the proposed loss function is beneficial, likely by regulating the contribution of each loss component.

6 Conclusions

This paper discussed challenges associated with retrieving templates for answering customer support questions, proposing a dense retrieval framework to address the task. This framework features innovative contributions in terms of (a) extending in-batch negatives to support unpaired sampling of queries and templates, and (b) a novel loss function that considers more similarity relations from the training data within each batch. Experiments on two different datasets of customer support interactions attest to the improvements brought forward by the proposed ideas. For future work, we plan to adapt and test the proposed techniques in other tasks that involve unbalanced corpora and large texts, such as general FAQ retrieval or question answering (Clark et al., 2020; De Bruyn et al., 2021).

Acknowledgments

This project has received funding from the European Union's Horizon 2020 research and innovation program under grant agreement No 873904. Bruno Martins was supported by Fundação para a Ciência e Tecnologia (FCT), through the INESC-ID multi-annual funding (i.e., the project with reference UIDB/50021/2020).

References

Elias Bassani. 2022. ranx: A blazing-fast Python library for ranking evaluation and comparison. In *Proceedings of the European Conference on Information Retrieval*.

Rogerio Bonatti, Arthur G. De Paula, Victor S. Lamarca, and Fabio G. Cozman. 2016. Effect of part-of-speech and lemmatization filtering in email classification for automatic reply. In *Proceedings of the Workshops at the AAAI Conference on Artificial Intelligence*.

Jonathan H. Clark, Eunsol Choi, Michael Collins, Dan Garrette, Tom Kwiatkowski, Vitaly Nikolaev, and Jennimaria Palomaki. 2020. TyDi QA: A benchmark for information-seeking question answering in typologically diverse languages. *Transactions of the Association for Computational Linguistics*, 8.

Maxime De Bruyn, Ehsan Lotfi, Jeska Buhmann, and Walter Daelemans. 2021. MFAQ: a multilingual FAQ dataset. In *Proceedings of the Workshop on Machine Reading for Question Answering*.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the Conference of the*

North American Chapter of the Association for Computational Linguistics: Human Language Technologies.

Jeff Johnson, Matthijs Douze, and Hervé Jégou. 2019. Billion-scale similarity search with GPUs. *IEEE Transactions on Big Data*, 7(03).

Vladimir Karpukhin, Barlas Oguz, Sewon Min, Patrick Lewis, Ledell Wu, Sergey Edunov, Danqi Chen, and Wen tau Yih. 2020. Dense passage retrieval for open-domain question answering. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*.

Diederik P. Kingma and Jimmy Ba. 2015. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*.

Yizhi Li, Zhenghao Liu, Chenyan Xiong, and Zhiyuan Liu. 2021. More robust dense retrieval with contrastive dual learning. In *Proceedings of the ACM SIGIR International Conference on Theory of Information Retrieval*.

Jimmy Lin, Xueguang Ma, Sheng-Chieh Lin, Jheng-Hong Yang, Ronak Pradeep, and Rodrigo Nogueira. 2021. Pyserini: A Python toolkit for reproducible information retrieval research with sparse and dense representations. In *Proceedings of the International ACM SIGIR Conference on Research and Development in Information Retrieval*.

Tri Nguyen, Mir Rosenberg, Xia Song, Jianfeng Gao, Saurabh Tiwary, Rangan Majumder, and Li Deng. 2016. MS-MARCO: A human generated MAchine Reading COmprehension dataset. arXiv preprint arXiv:1611.09268.

Yingqi Qu, Yuchen Ding, Jing Liu, Kai Liu, Ruiyang Ren, Wayne Xin Zhao, Daxiang Dong, Hua Wu, and Haifeng Wang. 2021. RocketQA: An optimized training approach to dense passage retrieval for open-domain question answering. In *Proceedings of the Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*.

Ricardo Rei. 2019. Towards a data-driven automation of customer support. Master's thesis, Instituto Superior Técnico of the University of Lisbon.

Nils Reimers and Iryna Gurevych. 2020. Making monolingual sentence embeddings multilingual using knowledge distillation. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*.

Ruiyang Ren, Shangwen Lv, Yingqi Qu, Jing Liu, Wayne Xin Zhao, QiaoQiao She, Hua Wu, Haifeng Wang, and Ji-Rong Wen. 2021. PAIR: Leveraging passage-centric similarity relation for improving dense passage retrieval. In *Findings of the Association for Computational Linguistics*.

Stephen Robertson and Hugo Zaragoza. 2009. The probabilistic relevance framework: BM25 and beyond. *Foundations and Trends in Information Retrieval*, 3(4).

Eriks Sneiders, Jonas Sjöbergh, and Alyaa Alfalahi. 2016. Email answering by matching question and context-specific text patterns: Performance and error analysis. In *Proceedings of the World Conference on Information Systems and Technologies*. Springer.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Proceedings of the International Conference on Neural Information Processing Systems*.

Lee Xiong, Chenyan Xiong, Ye Li, Kwok-Fung Tang, Jialin Liu, Paul N. Bennett, Junaid Ahmed, and Arnold Overwijk. 2021. Approximate nearest neighbor negative contrastive learning for dense text retrieval. In *Proceedings of the International Conference on Learning Representations*.

Weiwen Yang and Linchi Kwok. 2012. Improving the automatic email responding system for computer manufacturers via machine learning. In *Proceedings of the International Conference on Information Management, Innovation Management and Industrial Engineering*.

Andrew Yates, Rodrigo Nogueira, and Jimmy Lin. 2021. Pretrained transformers for text ranking: BERT and beyond. In *Proceedings of the ACM International Conference on Web Search and Data Mining*.

Jingtao Zhan, Jiaxin Mao, Yiqun Liu, Jiafeng Guo, Min Zhang, and Shaoping Ma. 2021. Optimizing dense retrieval model training with hard negatives. In *Proceedings of the International ACM SIGIR Conference on Research and Development in Information Retrieval*.

Jingtao Zhan, Jiaxin Mao, Yiqun Liu, Min Zhang, and Shaoping Ma. 2020. Learning to retrieve: How to train a dense retrieval model effectively and efficiently. *arXiv preprint arXiv:2010.10469*.

Exploring Label Hierarchy in a Generative Way for Hierarchical Text Classification

Wei Huang[1]*, Chen Liu[2]†, Bo Xiao[2], Yihua Zhao[2], Zhaoming Pan[2], Zhimin Zhang[2], Xinyun Yang[2], Guiquan Liu[1]
[1]University of Science and Technology of China
[2]NetEase Media Technology (Beijing) Co., Ltd.
hw001@mail.ustc.edu.cn, gqliu@ustc.edu.cn
{liuchen5,xiaobo02,hzzhaoyihua}@corp.netease.com
{panzhaoming,zhangzhimin,yangxinyun}@corp.netease.com

Abstract

Hierarchical Text Classification (HTC), which aims to predict text labels organized in hierarchical space, is a significant task lacking in investigation in natural language processing. Existing methods usually encode the entire hierarchical structure and fail to construct a robust label-dependent model, making it hard to make accurate predictions on sparse lower-level labels and achieving low Macro-F1. In this paper, we explore the level dependency and path dependency of the label hierarchy in a generative way for building the knowledge of upper-level labels of current path into lower-level ones, and thus propose a novel PAAM-HiA-T5 model for HTC: a hierarchy-aware T5 model with path-adaptive attention mechanism. Specifically, we generate a multi-level sequential label structure to exploit hierarchical dependency across different levels with Breadth-First Search (BFS) and T5 model. To further improve label dependency prediction within each path, we then propose an original path-adaptive attention mechanism (PAAM) to lead the model to adaptively focus on the path where the currently generated label is located, shielding the noise from other paths. Comprehensive experiments on three benchmark datasets show that PAAM-HiA-T5 greatly outperforms all state-of-the-art HTC approaches especially in Macro-F1.

1 Introduction

Hierarchical text classification (HTC), where text labels are predicted within a hierarchical structure, is a challenging task that has not yet received due attention within the field of multi-label classification. HTC methods have been extensively applied in industry domains, e.g., news article classification (Sandhaus, 2008), product classification in E-commerce (Yu et al., 2018), bidding strategy in paid search marketing (Agrawal et al., 2013).

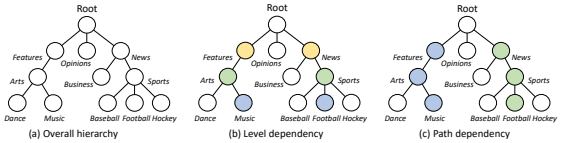

Figure 1: (a): the static labeling process is uniform and simultaneous for all labels in the label hierarchy. (b): the dynamic labeling process where the lower-level labels depend on the upper-level labels. (c): the dynamic labeling process focuses on ancestor labels already generated on the current path.

In HTC tasks, labels at lower-level are inevitably sparse due to the hierarchical structure. Many studies (Barbedo and Lopes, 2006; Xiao et al., 2007; Johnson and Zhang, 2015) completely or partially neglect such hierarchical structure and fail to accurately predict those lower-level labels, achieving low Macro-F1 score. Existing studies (Peng et al., 2021; Wu et al., 2019) have proved that introducing structure information can boost the predictive power on low-level labels and thus improve the overall task performance. A number of studies (Cesa-Bianchi et al., 2006; Shimura et al., 2018; Wehrmann et al., 2018; Banerjee et al., 2019) propose to construct multi-level classifiers that are trained independently and predicted sequentially, where only local maximum is achieved and propagation of error negatively impacts model prediction. Some studies design end-to-end models that introduce various strategies (such as Tree-LSTM/GCN (Zhou et al., 2020), label semantics matching network (Chen et al., 2021), graph-CNN (Peng et al., 2018) and hierarchical fine-tuning based CNN (Shimura et al., 2018)) to encode the overall hierarchy information (as depicted in Figure 1 (a)), where label dependency across different levels (as depicted in Figure 1 (b)) is not captured in a more principled way and unnecessary noises are introduced. Such models tend to predict all labels simultaneously and independently with sigmoid function, and they could cause serious *label inconsistencies*

*Contribution during internship at NetEase Inc.
†Corresponding Author

Proceedings of the 29th International Conference on Computational Linguistics, pages 1116–1127
October 12–17, 2022.

(One label is predicted positive but its ancestors are not. For example, in Figure 1 (b), the "Football" is predicted while the "Sports" is not.) and require post-processing to rectify these contradictions (Mao et al., 2019). Although one recent study (Mao et al., 2019) develops label-dependent models with reinforcement learning, it still fails to address label dependency within each path (as depicted in Figure 1 (c)) and fails to fully integrate labels and text information.

This paper seeks to close the gap by proposing the PAAM-HiA-T5. We are not only the first to capture lower-level label dependency on upper-level ones with generation model, but also the first to accurately explore the level dependency within each path. In each step of prediction phase, our model generates next label based on the text sequence and labels previously generated on current path. As illustrated in Figure 1(c), our model sequentially predicts "Features", "News", "Arts", "Sports", "Music", "Football" labels. In the process where label "Football" is generated, our model pays more attention on "News" and "Sports" labels on its own path instead of "Features" and "Arts" labels on another path. Therefore, our model is less likely to cause label inconsistency. For example, when labels "News" and "Sports" are known to the model, it is easier to predict the label "Football" as Figure 1 (b) shows.

Our PAAM-HiA-T5 model follows a two-step design.

Hierarchy-aware T5 (HiA-T5), a variant of T5 that is fully aware of the level dependency in a multi-level sequential generative manner. Inspired by the idea that conventional classification routines often follow the order that from coarse-grained to fine-grained to predict labels, we firstly use Breadth-First Search (BFS) to flatten hierarchical labels into multi-level sequential label structure, transforming the hierarchy to sequence. T5 model is applied to map the text sequence to label sequence, where the text sequence and upper-level labels generated earlier are then integrated in order to determine the next label.

Path-adaptive attention mechanism (PAAM), a mechanism to exploit the label correlation within each path and shield the noise from other paths. Through the PAAM, the model can adaptively obtain a more reasonable attention distribution belonging to the path where the currently generated label is located. PAAM is implemented by means of a regularization method.

This study makes the following major contributions:

- We propose a novel HiA-T5, a multi-level sequential label generative model to exploit label dependency across different levels. The mapping relationship between text sequence and label sequence is examined in each step of prediction.
- We propose an original PAAM to lead the model to adaptively focus on the path where the currently generated label is located, shielding the noise from other paths to further improve prediction performance.
- Experiments on various datasets show that our proposed PAAM-HiA-T5 achieves significantly and consistently better performance than state-of-the-art models. Experimental analysis shows that PAAM-HiA-T5 is especially benefical to those lower-level long-tailed labels. And our model can obtain better label consistency.

2 Related Work

Hierarchical text classification (HTC) is a critical task with numerous applications (Qu et al., 2012; Agrawal et al., 2013; Zhang et al., 2019; Peng et al., 2016). By methods of hierarchical information modeling, HTC approaches can be categorized into flat, local and global approaches (Silla and Freitas, 2011).

Flat approaches (Hayete and Bienkowska, 2005; Barbedo and Lopes, 2006; Xiao et al., 2007; Johnson and Zhang, 2015) completely or partially ignore the label hierarchy and each label is independently predicted. Some of them simply ignore the invaluable hierarchical information and achieve poor performance. Some others predict leaf nodes first and then mechanically add their ancestor labels, which is only applicable where different paths in the label hierarchy share the same length.

Local approaches (Koller and Sahami, 1997; Cesa-Bianchi et al., 2006; Shimura et al., 2018; Wehrmann et al., 2018; Banerjee et al., 2019) construct multiple local classifiers so that the misclassification at a certain level is propagated downwards the hierarchy, easily leading to the exposure of bias (Silla and Freitas, 2011). Specifically, Peng et al. (2018) proposes deep graph convolutional

neural networks with hierarchical regularization. Wehrmann et al. (2018) utilizes a multi-label neural network architecture with local and global optimization. To address the lower-level labels sparsity problem, Shimura et al. (2018) takes advantage of a CNN-based model with the fine-tuning method. Banerjee et al. (2019) proposes to transfer the parameters of parent classifiers to initialize child classifiers for HTC task.

Global approaches (Gopal and Yang, 2013; Mao et al., 2019; Wu et al., 2019; Zhou et al., 2020; Peng et al., 2021), where the entire structural information is encoded and all labels are simultaneously predicted, has become recent mainstream due to its better performance. Mao et al. (2019) handles HTC task with reinforcement-learning-based label assignment method. Wu et al. (2019) uses meta-learning to model the label interaction for multi-label classification. Zhou et al. (2020) utilizes the Bi-TreeLSTM and GCN to model hierarchical relationship and makes flat predictions for hierarchical labels. Peng et al. (2021) combines CNN, RNN, GCN, and CapsNet to model hierarchical labels. Chen et al. (2021) formulate HTC as a semantic matching problem to mine the text-label semantics relationship. Although recent researchers have managed to introduce hierarchical information in different fashions, most of them still regard flat multi-label classification as the backbone of HTC where all labels are predicted simultaneously and independently. Their exploitation of hierarchical structure is far from sufficient.

3 Problem Definition

For HTC, we define the overall label hierarchy as a tree-like structure, denoted by $T = (L, E)$, where $L = \{l_1, l_2, \ldots, l_K\}$ refers to the set of all label nodes in the corpus and K is the total number of them. E refers to the set of edges indicating the nodes' parent-child relations. Formally, we denote text objects as $\mathcal{X} = \{X_1, X_2, \ldots, X_N\}$ and their labels as $\mathcal{L} = \{L_1, L_2, \ldots, L_N\}$.

Each text object is represented by a text sequence $X_i = [x_1, x_2, \ldots, x_J]$, where x_j is a word and J is the number of words in the text object. Meanwhile, each text object X_i is mapped to a original label set $L_i = \{l_1, l_2, \ldots, l_k, 1 \leq k < K\}$ that contains multiple labels. We then define a set of special symbols $S = \{\_, /, EOS\}$ to identify special hierarchical relationships in the hierarchy.

All labels L in the corpus constitute the overall label hierarchy T. The original label set $L_i = \{l_1, l_2, \ldots, l_k, 0 \leq k < K\}$ of any text object X_i constitute an partial label hierarchy T_i and $T_i \subset T$. We aim to train a model to predict corresponding label set L_i for each text object X_i, where the label set L_i is constrained by the hierarchy T_i.

4 Background

The T5 model consists of an encoder-decoder architecture, which mainly includes the Multi-head Attention Mechanism, the Feed-Forward Network and so on (Raffel et al., 2020), as depicted in the Figure 2.

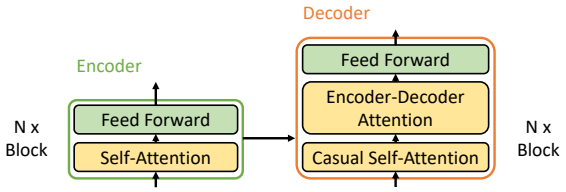

Figure 2: Structure of T5. Following each Multi-head Attention sublayer and Feed-Forward sublayer, there are a series of dropout, residual connection and layer normalization. These parts are omitted in the figure and the following formulas for simplicity's sake.

Attention mechanism $\mathrm{Attn}(\cdot)$ is calculated as:

$$\begin{aligned} \mathrm{Attn}(Q, K, V) &= \mathrm{Score}(Q, K)V \\ \mathrm{Score}(Q, K) &= \mathrm{softmax}(\frac{QK^T}{\sqrt{d_k}}) \end{aligned} \tag{1}$$

where $Q, K, V \in \mathbb{R}^{n \times d_{model}}$ and the length of sequence is n. The output attention score matrix of $\mathrm{Score}(Q, K)$ is denoted as $Score \in \mathbb{R}^{n \times n}$.

Multi-head attention independently executes attention mechanism of H heads and then concatenate their results, and it is denoted as $\mathrm{MH}(\cdot)$. The Feed-Forward Network consists of two linear transformations with a nonlinear activation function in between, and it is represented as $\mathrm{FFN}(\cdot)$.

T5 encoder is composed of a stack of "encoder blocks" and we define the number of blocks as B. Each block contains a self-attention sublayer and a feed-forward sublayer. The input sequence of encoder is mapped to the embedding $Q_{encoder}, K_{encoder}, V_{encoder} \in \mathbb{R}^{n \times d_{model}}$, which are then passed into the encoder. The output of encoder is denoted as $O_{encoder}$.

$$\begin{aligned} &\mathrm{Block}_{\mathrm{Encoder}}(Q_{encoder}, K_{encoder}, V_{encoder}) \\ &= \mathrm{FFN}(\mathrm{MH}(Q_{encoder}, K_{encoder}, V_{encoder})) \\ &\mathrm{Encoder}(Q_{encoder}, K_{encoder}, V_{encoder}) \\ &= \mathrm{stack}(\mathrm{Block}_{\mathrm{Encoder}}(Q_{encoder}, K_{encoder}, V_{encoder})) \end{aligned} \tag{2}$$

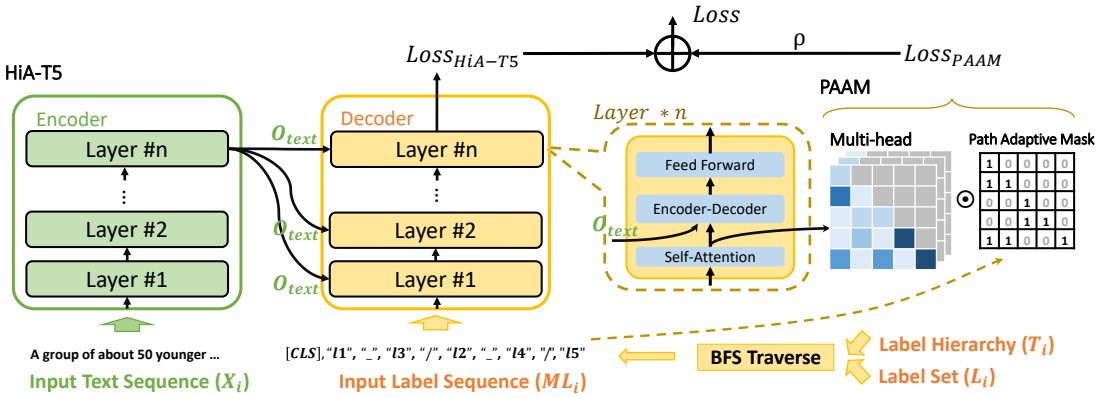

Figure 3: The overall structure of PAAM-HiA-T5. PAAM-HiA-T5 consists of a HiA-T5 and a PAAM. The dataflows of one decoder layer are illustrated in the yellow dashed box.

The structure of the decoder looks similar to that of the encoder, except that it has an additional encoder-decoder attention sublayer that attends to the output of the encoder stack, following each casual self-attention sublayer. The causal attention mechanism of decoder only permits the model attend to past outputs. In the end, we obtain the decoder output denoted as $O_{decoder}$.

$$\begin{aligned}&\text{Block}_{\text{Decoder}}(Q_{decoder}, K_{decoder}, V_{decoder}, O_{encoder})\\&= \text{FFN}(\text{MH}(\text{MH}(Q_{decoder}, K_{decoder}, V_{decoder}), O_{encoder}, O_{encoder}))\\&\text{Decoder}(Q_{decoder}, K_{decoder}, V_{decoder}, O_{encoder})\\&= \text{stack}(\text{Block}_{\text{Decoder}}(Q_{decoder}, K_{decoder}, V_{decoder}, O_{encoder}))\end{aligned} \quad (3)$$

5 Hierarchy-Aware T5 with Path-Adaptive Attention Mechanism

As depicted in Figure 3, we propose a PAAM-HiA-T5 model for HTC: a **Hi**erarchy-**A**ware **T5** model with **P**ath-**A**daptive **A**ttention **M**echanism. PAAM-HiA-T5 consists of the HiA-T5 for level-dependent label generation and the PAAM for path-specific label generation.

5.1 Hierarchy-Aware T5

Level-dependent HiA-T5 The major shortcoming of previous HTC methods is the inadequate application of hierarchy information. In contrast, HiA-T5 exploits label dependency across different levels of the hierarchy with Breadth-First Search (BFS) and multi-head attention mechanism.

HiA-T5 firstly explore the label hierarchy T_i with Breadth-First Search (Cormen et al., 2001) to flatten the label set $L_i = \{l_1, l_2, l_3, l_4, l_5\}$ into multi-level sequential label $ML_i = [CLS, l_1, \_, l_3, /, l_2, \_, l_4, /, l_5]$, transforming the hierarchy to multi-level label sequence, as illustrated in Figure 4 (a). In this process, '_' between labels denotes intra-level relationship, while '/' signifies inter-level relationship.

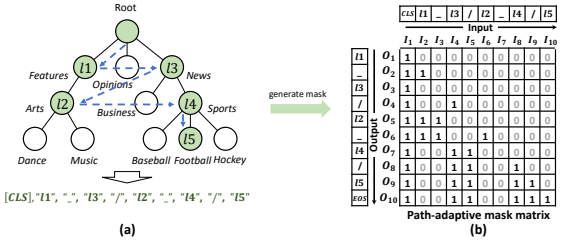

Figure 4: (a): A text with the content "David Beckham attends art exhibition launch during Paris Fashion Week" is tagged with "News", "Sports", "Football", "Features" and "Arts". And its label hierarchical structure is explored in Breadth-First Search (blue dash line). (b): path-adaptive mask matrix makes the ith output element use current input element and all its ancestors.

On one hand, the text sequence $X_i = [x_1, x_2, \ldots, x_J]$ is mapped to embedding sequence $Q_{text}, K_{text}, V_{text} \in \mathbb{R}^{n_t \times d_{model}}$ of length n_t, which are then passed into T5 encoder:

$$O_{text} = \text{Encoder}(Q_{text}, K_{text}, V_{text}) \quad (4)$$

The output encoder representation for semantic features of varied granularities is O_{text}.

On the other hand, the multi-level label sequence $ML_i = [CLS, l_1, \_, l_3, /, l_2, \_, l_4, /, l_5]$ is mapped to embeddings sequence $Q_{label}, K_{label}, V_{label} \in \mathbb{R}^{n_l \times d_{model}}$ of length n_l, which are passed into T5 decoder together:

$$O_{hierarchy} = \text{Decoder}(Q_{label}, K_{label}, V_{label}, O_{text}) \quad (5)$$

Specifically, HiA-T5 fully explores the label dependency across different levels through the self-attention mechanism. With the help of the intra-level separator '_' and the inter-level separator '/', the causal decoder self-attention mechanism fully excavates the intra-level parallel and mutually ex-

clusive relationship, as well as the inter-level dependent and appurtenant relationship. The output representation of the decoder causal self-attention mechanism incorporating level dependency information is $A_{label} = \mathrm{MH}(Q_{label}, K_{label}, V_{label})$.

So far, we have obtained the text representation O_{text} highlighting the semantic features of texts with different granularities and the label representation A_{label} incorporating label dependency across different levels. The output representation of the encoder-decoder attention mechanism integrating these two is $A_{cross} = \mathrm{MH}(A_{label}, O_{text}, O_{text})$, which is a sufficient crossover information for following prediction.

Loss of HiA-T5 We have obtained the final decoder block output $O_{hierarchy}$ of HiA-T5, which fully integrates the label hierarchy information and the text semantic information of different granularities. Then $O_{hierarchy}$ is passed into a fully connected layer with a softmax output, which is also the final result of HiA-T5 denoted as $Pred$. $Pred$ is the result of n_l timesteps and $Pred \in \mathbb{R}^{n_l \times K}$.

$$Pred = \mathrm{softmax}(O_{hierarchy} W_3 + b_3) \quad (6)$$

where $W_3 \in \mathbb{R}^{d_{model} \times K}, b_3 \in \mathbb{R}^K$. In addition, any multi-level label sequence ML_i is transformed into $Truth \in \mathbb{R}^{n_l \times K}$, which is composed of one-hot vectors corresponding to all labels. Therefore, the cross-entropy loss of HIA-T5 is expressed as:

$$\mathcal{L}_{HiA-T5} = \mathrm{crossentropy}(Truth, Pred) \quad (7)$$

5.2 Path-Adaptive Attention Mechanism

PAAM is a mechanism that can lead the model to adaptively focus on the path where the currently generated label is located, shielding the noise from other paths. It is a regularization method designed in the training phase to encourage the model to pay more attention to ancestor labels on current path while penalizing those on other paths. We first obtain the path-adaptive mask matrix containing hierarchy information. Then path-adaptive attention loss is obtained according to operations on the path-adaptive mask matrix and the causal attention score matrix.

Path-Adaptive Mask Matrix Now the text sequence $X_i = [x_1, x_2, \ldots, x_J]$ is taken as input, which is fed into HiA-T5 for training, and its corresponding multi-level label sequence ML_i is taken as output.

According to the T5 structure of Figure 2, the sequence ML_i is first passed into causal attention sub-layer of decoder. Within this sub-layer, and according to formula (1), we get causal attention score matrix $Score$ of one head corresponding to the sequence ML_i, as depicted in Figure 5 (a).

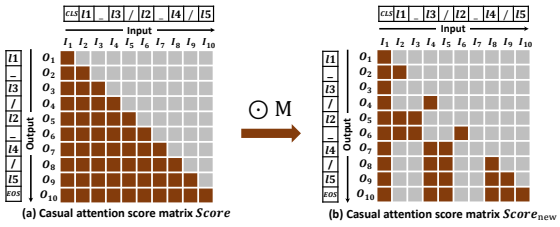

Figure 5: (a): Causal attention score matrix $Score$. The input and output of the causal self-attention mechanism are denoted as I and O respectively. $Score = \{s_{i,j}\} \in \mathbb{R}^{n_l \times n_l}$. Each element $s_{i,j}$ at row i and column j represents the weight at which the self-attention mechanism attends to input element j at output timestep i. The gray cell indicates the corresponding attention score $s_{i,j} = 0$. (b): Element-wise product result of $Score$ and M.

Then we define the path-adaptive mask matrix M, which can mask different parts of the label sequence at different decoding timestep i. The matrix M is obtained from the hierarchical structure of the label sequence corresponding to each text object. Specifically, the shape of matrix M is same as that of attention score matrix $Score$. Mask matrix $M = \{m_{i,j}\} \in \mathbb{R}^{n_l \times n_l}$ is also a lower triangular matrix, and it is only composed of 0 or 1 as shown below.

$$M = \begin{bmatrix} m_{1,1} & & & & 0 \\ m_{2,1} & m_{2,2} & & & \\ \vdots & \vdots & \ddots & \ddots & \\ m_{n_l,1} & m_{n_l,2} & \cdots & m_{n_l,n_l-1} & m_{n_l,n_l} \end{bmatrix} \quad (8)$$

O_i represents the input of the attention mechanism at the timestep i, and $O_i \in L \cup S$. If $O_i \in L$, we define $ancestor(O_i)$ as label O_i's ancestor labels and the special symbol immediately following them.

Then we define the following formula to fill the matrix M based on the parent-child relationship contained in each path of label hierarchy. The output timestep i and input timestep j correspond to the i-th row and j-th column of the matrix M. The element $m_{i,j}$ is determined by both the output O_i and the input I_j, and the formula is as follows:

$$m_{i,j} = \begin{cases} 1 & \{O_i \in L, I_j \in ancestor(O_i)\} \\ & \cup \{O_i \in S, I_j \in ancestor(O_{i-1}) \ or \ I_j = O_{i-1}\} \\ & \cup \{O_i \in L \cup S, I_j = CLS\} \\ 0 & else \end{cases} \quad (9)$$

Path-Adaptive Attention Loss With text sequence and labels previously generated in hand, we now introduce regularization and apply the path-adaptive dynamic mask matrix M, such that HiA-T5 decoder learns the weight of the attention matrix and pays more attention on the label's current path.

Having obtained the multi-level label sequence ML_i of a certain training sample, we use it as the input of causal self-attention of HiA-T5's decoder. According to the definition above, we get its path-adaptive mask matrix M, as depicted in Figure 4 (b). Furthermore, we get path-adaptive attention score matrix $Score_{path}$ as depicted in Figure 5 (b) by multiplying attention score matrix $Score$ and the path-adaptive mask matrix M element-wise:

$$Score_{path} = Score \odot M = softmax(\frac{QK^T}{\sqrt{d_k}}) \odot M \quad (10)$$

We define C as the index set of $ancestor(I_i)$. At any decoding timestep i, our goal is to make the sum of the attention scores $\sum_{j \in C} s_{i,j}$ of current path's labels as close to 1 as possible. Corresponding to attention scores matrix $Score_{path}$ of decoder's causal attention, that is, to make the sum of elements of each row in the matrix close to 1 as much as possible. According to the definition section, suppose $Score$ is the causal attention score matrix corresponding to the h-th head of b-th decoder "blocks", where $1 \leq h \leq H, 1 \leq b \leq B$. The path-adaptive attention loss is defined as:

$$\mathcal{L}_{PAAM} = \sum_{b=1}^{B} (\frac{\sum_{h=1}^{H} (\sum_{i=1}^{n_l} (1 - \sum_{j \in C} s_{i,j}))}{H}) \quad (11)$$

Therefore, the path-adaptive attention loss is added to the loss of HiA-T5 as total loss for training. The total loss function $Loss$ is obtained as below, where ρ is the coefficient of path-adaptive attention loss item.

$$\mathcal{L} = \mathcal{L}_{HiA-T5} + \rho \mathcal{L}_{PAAM} \quad (12)$$

6 Experiments

6.1 Experiment Setup

Datasets We conduct experiments on three public datasets, including RCV1-V2 (Lewis et al., 2004), NYTimes(NYT) (Sandhaus, 2008) and Web-of-Science(WOS) (Kowsari et al., 2017). RCV1-V2 is an English news categorization dataset and NYT is a news dataset from the New York Times in America. WOS is about scientific literature categorization. Labels of these datasets are organized into a tree-like structure. Relevant information of datasets is summarized in Table 1 and Table 2.

Dataset	$\|L\|$	Depth	Avg($\|L_i\|$)	Max($\|L_i\|$)	Train	Val	Test
RCV1	103	4	3.24	17	20833	2316	781265
NYT	166	8	7.6	38	23345	5834	7292
WOS	141	2	2.0	2	30070	7518	9397

Table 1: Statistical analysis of datasets: $|L|$ is the number of all labels in the hierarchy. Depth denotes the maximum level of the label hierarchy. Avg($|L_i|$) and Max($|L_i|$) denote average and maximum number of labels in each sample.

Dataset	level1	level2	level3	level4	level5	level6	level7	level8
RCV1	236334	20523	11850	23211	-	-	-	-
NYT	15161	2923	1160	842	1066	925	992	1460
WOS	6712	351	-	-	-	-	-	-

Table 2: Statistics of the average number of each label's occurrence at each level: $level_i$ denotes the level in the label hierarchy. In general, the labels of most samples are screwed towards upper levels and lower-level labels are more sparse.

We split RCV1-V2 in the benchmark split manner and take a small portion of the training set as validation set. For NYT and WOS, we randomly split data into training, validation and test sets.

Evaluation Metrics We use standard evaluation metrics, including **Micro-F1** and **Macro-F1** (Gopal and Yang, 2013; Peng et al., 2018; Huang et al., 2019), to measure the performance of all HTC methods. Micro-F1 equally weights all samples, while Macro-F1 gives equal weight to each label. As such, Micro-F1 gives more weight to frequent labels, while Macro-F1 equally weights all labels and is more sensitive to lower-level sparse labels which are shown in Table 2.

Experimental Settings The backbone pre-trained model we adopt is T5-base (Raffel et al., 2020) containing about 220M parameters. Tokenizer from T5 is utilized to preprocess the text. For PAAM-HiA-T5, the maximum sequence length of encoder is set as 300 for all datasets, and the maximum sequence length of decoder for RCV1, NYT and WOS are respectively set as 90, 120 and 20. When the model is trained, Adam optimizer is employed in a batch size of 10 with learning rate of 5e-4. The search range of coefficient ρ is {0.1,1,10,100,200}, and we set it to 100, 10, and 100 for RCV1, NYT and WOS respectively according to validation results. In the inference phase, greedy search is adopted. We set random seeds before experiments for the reproducibility of results, and the results reported in this paper are from the

Models	Micro-F1	Macro-F1
Flat Models		
HAN (Mao et al., 2019)	75.30	40.60
TextCNN (Mao et al., 2019)	76.60	43.00
TextRCNN (Zhou et al., 2020)	81.57	59.25
Local Models		
HR-DGCNN-3 (Peng et al., 2018)	76.18	43.34
HFT(M) (Shimura et al., 2018)	80.29	51.40
Htrans (Banerjee et al., 2019)	80.51	58.49
HMCN (Mao et al., 2019)	80.80	54.60
Global Models		
SGM (Zhou et al., 2020)	77.30	47.49
HE-AGCRCNN (Peng et al., 2021)	77.80	51.30
HiLAP-RL (Mao et al., 2019)	83.30	60.10
HiAGM (Zhou et al., 2020)	83.96	63.35
HiMatch (Chen et al., 2021)	84.73	64.11
Pretrained Language Models		
BERT	86.26	67.35
T5	86.14	67.39
BERT+HiAGM[1]	86.12	68.08
BERT+HiMatch (Chen et al., 2021)	86.33	68.66
PAAM-HiA-T5	**87.22**	**70.02**

Table 3: Performance comparison on RCV1-V2. The results of HAN (Yang et al., 2016), TextCNN (Kim, 2014) and HMCN (Wehrmann et al., 2018) are reported by Mao et al. (2019). Zhou et al. (2020) reports the results of TextRCNN (Lai et al., 2015) and SGM (Yang et al., 2018).

average of 3 random runs of the model.

6.2 Performance Comparison

The experimental comparison between PAAM-HiA-T5 and the state-of-the-art HTC methods are shown in Table 3 and 4, and our model outperforms all SOTA results of flat, local and global methods, both on Micro-F1 and Macro-F1. This demonstrates the strong power of PAAM-HiA-T5 in solving HTC by better mining hierarchical structure information. The level-dependency modeling and the path-adaptive attention mechanism bring significant improvement. HiAGM and HiMatch are effective baselines because they achieved the latest SOTA results in HTC. Our model greatly surpasses them on both metrics especially on Macro-F1. In general, the greater improvement on Macro-F1 shows that our model has greater capability in predicting sparse lower-level labels. In fact, it can be shown from Table 2 that sample labels become more sparse as level grows. Due to insufficient training, the lower-level label prediction becomes difficult. But our model utilizes the knowledge of upper-level labels in predicting lower-level ones by modeling level dependency and path dependency, and this explains the reason why our model achieves greater boost in Macro-F1 and has greater capability in predicting sparse lower-level labels.

Pre-trained language models are effective methods, which can often be combined with the existing model structure to improve the performance of specific tasks. BERT+HiMatch and BERT+HiAGM denote that HiMatch and HiAGM are respectively equiped with a pre-trained BERT (Kenton and Toutanova, 2019) compatible with their structures. The model sizes of BERT+HiMatch and BERT+HiAGM are in the same order of magnitude as that of PAAM-HiA-T5. Our model can still significantly outperform them, which shows the powerful capabilities of it (see Appendix C for more analysis on this). For a more detailed discussion about computational complexity please refer to the Appendix A.

Model	NYT		WOS	
	Micro-F1	Macro-F1	Micro-F1	Macro-F1
Flat Models				
TextRNN (Zhou et al., 2020)	70.29	53.06	77.94	69.65
TextCNN (Zhou et al., 2020)	70.11	56.84	82.00	76.18
TextRCNN (Zhou et al., 2020)	70.83	56.18	83.55	76.99
Local & Global Models				
HMCN (Mao et al., 2019)	72.2	47.4	–	–
HiAGM (Zhou et al., 2020)	74.97	60.83	85.82	80.28
HiMatch (Chen et al., 2021)	74.62	59.28	86.20	80.53
Pretrained Language Models				
BERT+HiMatch[2] (Chen et al., 2021)	76.79	63.89	86.70	81.06
PAAM-HiA-T5	**77.52**	**65.97**	**90.36**	**81.64**

Table 4: Performance comparison on the NYT and WOS datasets. We mainly compare the best performing flat, local, global and pre-trained models on RCV1-V2. The results of TextRNN (Liu et al., 2016), TextCNN (Kim, 2014) and TextRCNN (Lai et al., 2015) on NYT and WOS are reported by Zhou et al. (2020).

6.3 Performance Analysis

Method	Micro-F1	Macro-F1
T5	86.14	67.39
HiA-T5	86.67	69.09
PAAM-HiA-T5	87.22	70.02

Table 5: Ablation study of PAAM-HiA-T5 on RCV1-V2. Note that original T5 neither model the hierarchical structure information nor capture the hierarchial dependencies. It takes HTC as a generic multi-label classification task to generate unordered label sets corresponding to the text.

[1]The results of BERT+HiAGM on RCV1-V2 are implemented upon the released projects of HiAGM (https://github.com/Alibaba-NLP/HiAGM) and the BERT with mutli-label settings. We follow the MIT License.

[2]The results of HiMatch and BERT+HiMatch on NYT is reproduced upon the released project of HiMatch (https://github.com/RuiBai1999/HiMatch) and the BERT with mutli-label settings. We follow the MIT License.

Ablation Study and Analysis on Level Dependency Modeling The performance comparison of HiA-T5 and the original T5 is shown in Table 5. It is evident that HiA-T5 greatly outperforms the T5 both in Micro-F1 and Macro-F1. This result illustrates the effectiveness of capturing level dependency by introducing upper-level label knowledge to assist lower-level label prediction. Compared with T5, HiA-T5 boosts Macro-F1 by 1.70% and achieves substantial 0.53% improvement in Micro-F1. Greater boost in Macro-F1 demonstrates HiA-T5 is especially beneficial to lower-level long-tailed labels by introducing level dependency modeling.

Ablation Study and Analysis on Path-adaptive Attention Mechanism The performance comparison of PAAM-HiA-T5 and HiA-T5 is also shown in Table 5. PAAM-HiA-T5 greatly increases Micro-F1 and Macro-F1 especially in Macro-F1 compared with HiA-T5. This indicates that PAAM significantly improves the performance of HiA-T5 in more challenging multi-path scenarios by capturing precise path dependency.

As shown in Figure 6, the heat map of the causal self-attention score in PAAM-HiA-T5's encoder proves the effectiveness of PAAM, where the attention score is mainly distributed on the path of the label current being decoded.

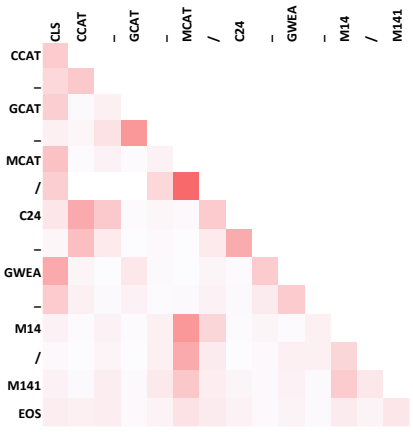

Figure 6: Heatmap of causal self-attention score for a random sample. Note that symbols instead of original labels are used and the score of each label is the average of its tokens' attention score for ease of display.

Performance Analysis on Label Granularity To find the origin of performance improvement, we analyze performance on label granularity based on different levels for T5, HiA-T5 and PAAM-HiA-5 on RCV1-V2. Figure 7 shows the level-based Macro-F1 of models and the absolute Macro-F1 differences among models. In general, our mechanism and strategy brings performance improvement on all levels, especially on lower levels.

In addition, the gap between HiA-T5 and T5 gets bigger as level deepens, and the phenomenon between HiA-T5 and PAAM-HiA-T5 is consistent. This illustrates that as the level grows, label prediction becomes more and more difficult, and the introduction of upper-level label knowledge by leveraging level dependency modeling and path-adaptive attention mechanism becomes more and more valuable. Specifically, the Macro-F1 of second and third levels for T5 are relatively low because there are some long-tailed labels among lower levels, but HiA-T5 and PAAM-HiA-T5 greatly enhance them. More performance comparison on label granularity with SOTA methods are provided in Appendix B.

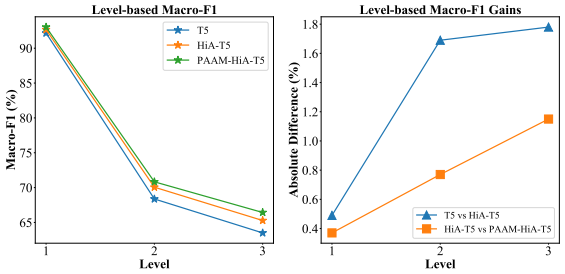

Figure 7: Ablation analysis based on different levels.

Analysis of Label Consistency Label inconsistency is a serious problem in many HTC approaches, due to the fact that they focus on flat multi-label classification and make independent predictions for all labels. It is worth mentioning that PAAM-HiA-T5 has outstanding classification performance while maintaining an ultra-low label inconsistency rate of 0.31%, as shown in Table 6. This is because our model fully leverages the constraints of upper-level labels generated earlier to predict the most accurate lower-level labels.

TextCNN	HMCN	HiAGM	HiMatch	BERT+HiAGM	BERT+HiMatch	**PAAM-HiA-T5**
3.74%	3.84%	1.35%	1.33%	1.52%	1.14%	**0.31%**

Table 6: Comparison of label inconsistency on RCV1-V2. We calculate the label inconsistency as the ratio of predictions with inconsistent labels. The results of TextCNN and HMCN are reported in Mao et al. (2019).

7 Conclusion

For HTC task, we explicitly define the concepts of "level dependency" and "path dependency" for the first time. Furthermore, in order to build the

knowledge of upper-level labels into lower-level ones in HTC task, we devise an innovative PAAM-HiA-T5 methodology by exhaustively exploring level dependency and path dependency of hierarchy in a generative manner. Comprehensive experiments on three benchmark datasets show that our model greatly outperforms all state-of-the-art HTC approaches especially in Macro-F1.

References

Rahul Agrawal, Archit Gupta, Yashoteja Prabhu, and Manik Varma. 2013. Multi-label learning with millions of labels: Recommending advertiser bid phrases for web pages. In *Proceedings of the 22nd International Conference on World Wide Web*, WWW '13, page 13–24, New York, NY, USA. Association for Computing Machinery.

Siddhartha Banerjee, Cem Akkaya, Francisco Perez-Sorrosal, and Kostas Tsioutsiouliklis. 2019. Hierarchical transfer learning for multi-label text classification. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 6295–6300, Florence, Italy. Association for Computational Linguistics.

Jayme Garcia sArnal Barbedo and Amauri Lopes. 2006. Automatic genre classification of musical signals. *EURASIP Journal on Advances in Signal Processing*, 2007:1–12.

Nicolò Cesa-Bianchi, Claudio Gentile, and Luca Zaniboni. 2006. Hierarchical classification: Combining bayes with svm. In *Proceedings of the 23rd International Conference on Machine Learning*, ICML '06, page 177–184, New York, NY, USA. Association for Computing Machinery.

Haibin Chen, Qianli Ma, Zhenxi Lin, and Jiangyue Yan. 2021. Hierarchy-aware label semantics matching network for hierarchical text classification. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 4370–4379.

Thomas H Cormen, Charles E Leiserson, Ronald L Rivest, and Clifford Stein. 2001. Introduction to algorithms second edition. *The Knuth-Morris-Pratt Algorithm*.

Siddharth Gopal and Yiming Yang. 2013. Recursive regularization for large-scale classification with hierarchical and graphical dependencies. In *Proceedings of the 19th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '13, page 257–265, New York, NY, USA. Association for Computing Machinery.

Boris Hayete and Jadwiga R Bienkowska. 2005. Gotrees: predicting go associations from protein domain composition using decision trees. In *Biocomputing 2005*, pages 127–138. World Scientific.

Wei Huang, Enhong Chen, Qi Liu, Yuying Chen, Zai Huang, Yang Liu, Zhou Zhao, Dan Zhang, and Shijin Wang. 2019. Hierarchical multi-label text classification: An attention-based recurrent network approach. In *Proceedings of the 28th ACM International Conference on Information and Knowledge Management*, CIKM '19, page 1051–1060, New York, NY, USA. Association for Computing Machinery.

Rie Johnson and Tong Zhang. 2015. Effective use of word order for text categorization with convolutional neural networks. In *Proceedings of the 2015 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 103–112, Denver, Colorado. Association for Computational Linguistics.

Jacob Devlin Ming-Wei Chang Kenton and Lee Kristina Toutanova. 2019. Bert: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of NAACL-HLT*, pages 4171–4186.

Yoon Kim. 2014. Convolutional neural networks for sentence classification. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1746–1751, Doha, Qatar. Association for Computational Linguistics.

Daphne Koller and Mehran Sahami. 1997. Hierarchically classifying documents using very few words. In *Proceedings of the Fourteenth International Conference on Machine Learning*, ICML '97, page 170–178, San Francisco, CA, USA. Morgan Kaufmann Publishers Inc.

Kamran Kowsari, Donald E Brown, Mojtaba Heidarysafa, Kiana Jafari Meimandi, Matthew S Gerber, and Laura E Barnes. 2017. Hdltex: Hierarchical deep learning for text classification. In *2017 16th IEEE international conference on machine learning and applications (ICMLA)*, pages 364–371. IEEE.

Siwei Lai, Liheng Xu, Kang Liu, and Jun Zhao. 2015. Recurrent convolutional neural networks for text classification. In *Twenty-ninth AAAI conference on artificial intelligence*.

David D Lewis, Yiming Yang, Tony Russell-Rose, and Fan Li. 2004. Rcv1: A new benchmark collection for text categorization research. *Journal of machine learning research*, 5(Apr):361–397.

Pengfei Liu, Xipeng Qiu, and Xuanjing Huang. 2016. Recurrent neural network for text classification with multi-task learning. *arXiv preprint arXiv:1605.05101*.

Yuning Mao, Jingjing Tian, Jiawei Han, and Xiang Ren. 2019. Hierarchical text classification with reinforced label assignment. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 445–455, Hong Kong, China. Association for Computational Linguistics.

Hao Peng, Jianxin Li, Yu He, Yaopeng Liu, Mengjiao Bao, Lihong Wang, Yangqiu Song, and Qiang Yang. 2018. Large-scale hierarchical text classification with recursively regularized deep graph-cnn. In *Proceedings of the 2018 World Wide Web Conference*, WWW '18, page 1063–1072, Republic and Canton of Geneva, CHE. International World Wide Web Conferences Steering Committee.

Hao Peng, Jianxin Li, Senzhang Wang, Lihong Wang, Qiran Gong, Renyu Yang, Bo Li, Philip S. Yu, and Lifang He. 2021. Hierarchical taxonomy-aware and attentional graph capsule rcnns for large-scale multi-label text classification. *IEEE Transactions on Knowledge and Data Engineering*, 33(6):2505–2519.

Shengwen Peng, Ronghui You, Hongning Wang, Chengxiang Zhai, Hiroshi Mamitsuka, and Shanfeng Zhu. 2016. Deepmesh: deep semantic representation for improving large-scale mesh indexing. *Bioinformatics*, 32(12):i70–i79.

Bo Qu, Gao Cong, Cuiping Li, Aixin Sun, and Hong Chen. 2012. An evaluation of classification models for question topic categorization. *Journal of the American Society for Information Science and Technology*, 63(5):889–903.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J. Liu. 2020. Exploring the limits of transfer learning with a unified text-to-text transformer. *Journal of Machine Learning Research*, 21(140):1–67.

Evan Sandhaus. 2008. The New York Times Annotated Corpus.

Kazuya Shimura, Jiyi Li, and Fumiyo Fukumoto. 2018. HFT-CNN: Learning hierarchical category structure for multi-label short text categorization. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 811–816, Brussels, Belgium. Association for Computational Linguistics.

Carlos N Silla and Alex A Freitas. 2011. A survey of hierarchical classification across different application domains. *Data Mining and Knowledge Discovery*, 22(1):31–72.

Jonatas Wehrmann, Ricardo Cerri, and Rodrigo Barros. 2018. Hierarchical multi-label classification networks. In *Proceedings of the 35th International Conference on Machine Learning*, volume 80 of *Proceedings of Machine Learning Research*, pages 5075–5084. PMLR.

Jiawei Wu, Wenhan Xiong, and William Yang Wang. 2019. Learning to learn and predict: A meta-learning approach for multi-label classification. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 4354–4364, Hong Kong, China. Association for Computational Linguistics.

Zhongzhe Xiao, Emmanuel Dellandrea, Weibei Dou, and Liming Chen. 2007. Hierarchical classification of emotional speech. *IEEE Transactions on Multimedia*, 37.

Pengcheng Yang, Xu Sun, Wei Li, Shuming Ma, Wei Wu, and Houfeng Wang. 2018. SGM: Sequence generation model for multi-label classification. In *Proceedings of the 27th International Conference on Computational Linguistics*, pages 3915–3926, Santa Fe, New Mexico, USA. Association for Computational Linguistics.

Zichao Yang, Diyi Yang, Chris Dyer, Xiaodong He, Alex Smola, and Eduard Hovy. 2016. Hierarchical attention networks for document classification. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 1480–1489, San Diego, California. Association for Computational Linguistics.

Wenhu Yu, Zhiqiang Sun, Haifeng Liu, Zhipeng Li, and Zhitong Zheng. 2018. Multi-level deep learning based e-commerce product categorization. In *eCOM@SIGIR*.

Yu Zhang, Frank F Xu, Sha Li, Yu Meng, Xuan Wang, Qi Li, and Jiawei Han. 2019. Higitclass: Keyword-driven hierarchical classification of github repositories. In *2019 IEEE International Conference on Data Mining (ICDM)*, pages 876–885. IEEE.

Jie Zhou, Chunping Ma, Dingkun Long, Guangwei Xu, Ning Ding, Haoyu Zhang, Pengjun Xie, and Gongshen Liu. 2020. Hierarchy-aware global model for hierarchical text classification. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 1106–1117, Online. Association for Computational Linguistics.

A Computational Complexity

We compare the computational complexity of PAAM-HiA-T5 with that of the best performing BERT+HiMatch and BERT+HiAGM. From the perspective of space complexity, PAAM-HiA-T5 contains roughly 220M parameters. BERT+HiMatch and BERT+HiAGM respectively contain about 153M and 143M parameters. The model sizes of these three models are in the same order of magnitude. Regarding time complexity, the time cost of PAAM-HiA-T5 is only about 0.82 times that of BERT+HiMatch and about 0.91 times that of BERT+HiAGM during training with batch size 10. In the inference stage, all three models run on the same GeForce RTX 2080 Ti GPU with their respectively highest GPU usage. Under this condition, PAAM-HiA-T5's total time cost for predicting

all test data is about 3 times that of BERT+HiMatch and BERT+HiAGM. PAAM-HiA-T5 establishes new SOTA results on HTC task, so we think that the enlargement in computational complexity due to the generative model properties is acceptable.

B Performance Comparison on Label Granularity with SOTA Methods

To further clarify the superiority of PAAM-HiA-T5, we perform the level-based performance analysis between our approach and other best performing SOTA methods on RCV1-V2. The level-based Micro-F1 scores and Macro-F1 scores are shown in Table 8. There is a dip in Micro-F1 score at second level for all models because there are lots of confusing labels with close concepts at second level. The relatively low Macro-F1 scores at the second and third levels are due to the presence of long-tailed labels. Figure 8 shows that our model maintains advanced performance on all levels, especially on lower levels. This reflects that our model has a huge advantage in dealing with lower-level long-tailed labels with sparse data.

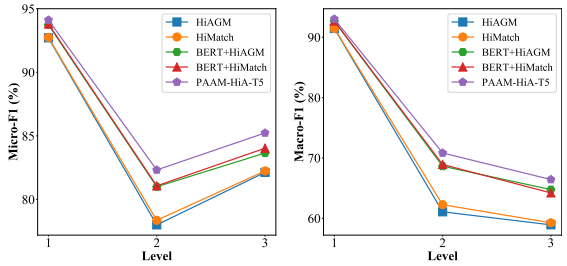

Figure 8: Performance analysis on label granularity based on different levels.

C Exploring the Impact of Hierarchy-Aware Module on the Pre-trained Base Model

We find that BERT+HiAGM, BERT+HiMatch and PAAM-HiA-T5 are most competitive methods according to previous experiments. On the one hand, the pre-trained models, including BERT and T5, can be viewed as the base models. On the other hand, different mechanisms and strategies, including HiAGM, HiMatch and PAAM-HiA, are utilized to exploit hierarchical structure information based on the pre-trained base models, and they can be regarded as different hierarchy-aware modules. We want to study the improvement of the pre-trained base models brought by different hierarchy-aware modules in HTC task.

Table 7 shows that the base models' performance of BERT+HiAGM, BERT+HiMatch and PAAM-HiA-T5 is close, both on Micro-F1 and Macro-F1. Therefore, it is fair to discuss the improvement brought by different hierarchy-aware modules to the pre-trained base model, and the performance changes are illustrated in Figure 9. For BERT+HiAGM, not only did the HiAGM not improve BERT's Micro-F1 score, it actually lowered the Micro-F1 score. The reason may be that HiAGM introduces noise in the process of encoding the overall hierarchy information. This degrades the performance of BERT+HiAGM on frequent labels. For BERT+HiMatch, HiMatch brings a relatively large improvement on Macro-F1, but a slight improvement on Micro-F1. This demonstrates that HiMatch has limited improvement for BERT on predicting middle-level and upper-level labels. But for PAAM-HiA-T5, PAAM-HiA module greatly boosts both Micro-F1 and Macro-F1 and establishes new SOTA results.

In conclusion, starting from base models with close performance, the improvement brought by the PAAM-HIA module to T5 significantly exceeds that brought by the HiMatch and HiAGM to BERT. Moreover, thanks to the PAAM-HiA module, our model outperforms all SOTA methods. All of the above fully illustrate that our mechanism and strategy (PAAM-HiA module), not just the powerful pre-trained base model, are important reasons for the strong power of our model in HTC task.

Ablation	BERT+HiAGM		BERT+HiMatch		PAAM-HiA-T5	
	Micro-F1	Macro-F1	Micro-F1	Macro-F1	Micro-F1	Macro-F1
BASE MODEL	86.26	67.35	86.26	67.35	86.14	67.39
+ Hierarchy-Aware Module	86.12	68.08	86.33	68.66	**87.22**	**70.02**

Table 7: Ablation study of hierarchy-aware modules on pre-trained base models. Specifically, "BASE MODEL" is either BERT or T5. "+ Hierarchy-Aware Module" denotes adding a hierarchy-aware module to the corresponding base model to obtain the final models.

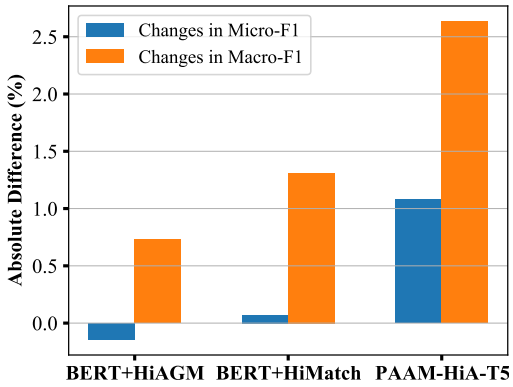

Figure 9: Model performance changes brought about by the hierarchy-aware modules. Specifically, the figure above shows the absolute difference between the performance of BERT+HiAGM, BERT+HiMatch and PAAM-HiA-T5 and that of BERT, BERT, and T5.

MuSeCLIR: A Multiple Senses and Cross-lingual Information Retrieval dataset

Wing Yan Li **Julie Weeds** **David Weir**
TAG laboratory
University of Sussex
Brighton, UK
{wl238, j.e.weeds, d.j.weir}@sussex.ac.uk

Abstract

This paper addresses a deficiency in existing cross-lingual information retrieval (CLIR) datasets and provides a robust evaluation of CLIR systems' disambiguation ability. CLIR is commonly tackled by combining translation and traditional IR. Due to translation ambiguity, the problem of ambiguity is worse in CLIR than in monolingual IR. But existing auto-generated CLIR datasets are dominated by searches for named entity mentions, which does not provide a good measure for disambiguation performance, as named entity mentions can often be transliterated across languages and tend not to have multiple translations. Therefore, we introduce a new evaluation dataset (MuSeCLIR) to address this inadequacy. The dataset focusses on polysemous common nouns with multiple possible translations. MuSeCLIR is constructed from multilingual Wikipedia and supports searches on documents written in European (French, German, Italian) and Asian (Chinese, Japanese) languages. We provide baseline statistical and neural model results on MuSeCLIR which show that MuSeCLIR has a higher requirement on the ability of systems to disambiguate query terms.

1 Introduction

Cross-Lingual Information Retrieval (CLIR) is a subfield of Information Retrieval (IR) where the task is to retrieve documents in language Y using queries in language X. Frequently, it is a combined process of translation and conventional IR. For efficiency, it is more common to translate queries from language X to language Y than to translate documents from language Y to language X.

Lexical ambiguity is a problem, and disambiguation found beneficial, in many natural language processing tasks, including machine translation (Raganato et al., 2019), information extraction (Delli Bovi et al., 2015) and information retrieval (Blloshmi et al., 2021). This problem is exacerbated in CLIR due to translation ambiguity (Zhou et al., 2007). For example, *letter* could mean alphabetic characters (字母) or a message (信). However, in this case, translation ambiguity compounds the problem. 信 can also refer to believing in something and this meaning has nothing to do with *letter*. Therefore, to increase the precision of the retrieval process, it is crucial to identify the correct translation of the query words in context.

	English query	Translated query
(a)	Karen Carroll (judge)	KarenCarroll法官
(b)	Brian Duffy (chef)	BrianDuffy主廚
(c)	Larry Andersen	LarryAndersen
(d)	El Cacao, Veraguas	ElCacaoVeraguas

Table 1: Example queries from BI-139 (Sun and Duh, 2020) translated to Chinese using MUSE (Conneau et al., 2018) dictionaries.

Despite its importance, translation ambiguity problem has received relatively little attention by CLIR researchers, and has been overlooked by existing CLIR datasets. Many applications of CLIR, and thus many existing datasets, are dominated by searches for named entity mentions. For example, in the BI-139 English-Chinese dataset (Sun and Duh, 2020), about 73% of the queries contain at least one named entity. These do not tend to have multiple translations and, as illustrated by by the query examples from BI-139 shown in Table 1, can often be transliterated from one language to another. Looking at examples (a) and (b) of Table 1, these queries may be a mixture of named entity mentions and common nouns, but only common nouns are translated. Although named entity mentions can be ambiguous (e.g. different people share the same name), this is related to named entity ambiguity. Since we are viewing the problem from the translation perspective, we are focusing on lexical ambiguity.

In the real world, the need to search for unnamed

Proceedings of the 29th International Conference on Computational Linguistics, pages 1128–1135
October 12–17, 2022.

entity mentions exists. For example, on a global topic such as conservation (Marshall et al., 2020), researchers might want to collect information about *bats* from around the globe. *Bat* is not a named entity and is named differently in different languages when referred to as an animal (Italian: Chiroptera; Chinese: 蝙蝠). Hence, retrieval systems need to distinguish between the translation of *bat* as an animal and *bat* as a piece of stick-like equipment in sports, based on the context information. However, due to a large amount of unambiguous named entity mentions, existing CLIR datasets are not adequate to train and evaluate the disambiguation ability of systems. We introduce MuSeCLIR, a new and a more fine-grained evaluation dataset.

MuSeCLIR is designed to specifically evaluate systems' ability to carry out disambiguation in CLIR. It is derived from Wikipedia, a free and open-sourced resource. Wikipedia contains many pages that exist in multiple languages and thus makes Wikipedia a good resource for CLIR (Sun and Duh, 2020; Sasaki et al., 2018; Yu et al., 2021). Assuming every possible translation represents a word sense, common nouns with more than one possible translation are chosen. By doing so, we can minimise the number of named entity mentions appearing in queries, and test whether systems are able to rank documents more highly which contain the correct translation (in context) over other possible translations of the ambiguous words in queries.

We introduce MuSeCLIR, a new evaluation dataset that assesses the ability of systems to disambiguate ambiguous query terms. MuSeCLIR supports searches on documents written in European (French, German, Italian) and Asian (Chinese, Japanese) languages. Our codes are available on GitHub[1]. Users can reproduce and extend MuSeCLIR to other languages. In section 3, we provide the construction method of MuSeCLIR and the statistics. In section 4, MuSeCLIR is used as a benchmark to evaluate existing CLIR systems: BM25 and multilingual BERT (mBERT) ranker (Sun and Duh, 2020). The results indicate that, given similar types of queries, existing CLIR systems perform more poorly on MuSeCLIR compared to other existing datasets, showing their inadequacy and the need of MuSeCLIR to determine the most appropriate system in the real-world scenario.

[1] https://github.com/justinaL/MuSeCLIR/

2 Background and Related Work

Probabilistic approach BM25 (Robertson and Zaragoza, 2009) is a traditional bag-of-words retrieval function based on term frequency-inverse document frequency (TF-IDF). It is a statistical measure that relies on term frequency and matches a query against a document. BM25 is monolingual, so we employ the MUSE (Conneau et al., 2018) bilingual dictionaries to translate queries into the target language during experiments.

Elasticsearch[2] is an open-source search engine that implements the BM25 algorithm (Robertson and Zaragoza, 2009) and has built-in analysers that handle tokenisation and stemming. Here, we employ Elasticsearch 6.5.4 with default parameters[3]; `smartcn` and `kuromoji` analyser are used when handling the Chinese and Japanese documents, respectively.

Neural approach Recently, end-to-end CLIR models have attracted more attention. These systems align queries and documents into the same space and perform matching in this aligned space. Large pre-traine d language models (PLM), such as BERT (Devlin et al., 2019), is commonly adopted as the encoder (Jung et al., 2022; Nair et al., 2022; MacAvaney et al., 2019) and impressive results have been achieved. As CLIR involves multiple languages, CLIR systems usually utilise multilingual language models, like mBERT, to map queries and documents into a shared space, bypassing the translation step.

The mBERT ranker used here is a re-implementation of the vanilla BERT ranker proposed by MacAvaney et al. (2019). The vanilla BERT ranker adopted the fine-tuning paradigm with a linear layer stacked on top of BERT (Devlin et al., 2019). Following Sun and Duh (2020), the encoder is replaced with a pre-trained mBERT mode[4]. The `[CLS]` embedding at the final layer of mBERT that represents the query-document pair is used. When training, the positive sample is a query-document pair with relevance labels larger than 0; negative otherwise. The network is trained to optimise pairwise hinge loss with Adam optimiser and updates the weights in the last linear layer.

Datasets MuSeCLIR is compared against two existing datasets, both also developed from

[2] https://www.elastic.co/

[3] $b = 0.75$ and $k_1 = 1.2$

[4] BERT-Base; Multilingual cased; Training epochs = 20; Learning rate = 1e-5

Wikipedia: BI-139 from CLIRMatrix (Sun and Duh, 2020) and WikiClir dataset introduced by Sasaki et al. (2018). CLIRMatrix contains two subsets: BI-139 (bilingual dataset) and MULTI-8 (multilingual dataset[5]). As this paper is not studying multilingual IR, BI-139 (base version) is considered. Queries in WikiClir are the first sentences from the English Wikipedia pages with page titles removed; the average query length is 20 tokens. As page titles are kept in MuSeCLIR, we append page titles to their original queries forming *WikiClir title* for fair comparisons. Queries in BI-139 are Wikipedia page titles of 3 tokens on average. In both sets, documents are the first 200 tokens of a page that contains the main gist of the topic.

3 Dataset construction

MuSeCLIR is an English-centric dataset where all queries are in English that makes use of WikipediaAPI[6]. Common nouns with multiple translations are chosen from MUSE (Conneau et al., 2018) bilingual dictionaries.

Wikipedia provides a `disambiguation` page for potentially ambiguous article titles. This page contains links to other possible subtopics in addition to the main topic (selected nouns in our case). For example, *window* can refer to architecture, rectangle display on computers, etc. This page will provide links to the corresponding articles. Here, we assume each link within the `disambiguation` page is associated with a sense of the noun. In order to conserve the overall level of ambiguity, we remove nouns with less than 2 links on the `disambiguation` page. Administration pages like `Help`, `disambiguation`, etc. are then removed. Before adding a linked page to the final collection, we also check that i) the page contains an inter-language link to the desired target language; ii) the noun exists in the page title; and iii) the summary of the page in both English and the target language is not empty. We initialise the construction based on common nouns to minimise the number of named entity mentions. This method is easily adapted to other parts of speech, but we focus on polysemous nouns here.

[5]Multilingual IR is a task where queries need to retrieve documents from a multilingual pool of documents

[6]https://github.com/martin-majlis/Wikipedia-API

Language	# ambiguous nouns	Total # sentence queries	# documents
FR	2,045 (0.52)	41,958	9,884
DE	2,389 (0.51)	49,698	10,740
IT	1,565 (0.52)	30,675	7,640
ZH	1,137 (0.50)	26,344	5,080
JA	882 (0.52)	20,675	4,168

Table 2: MuSeCLIR dataset statistics. Corresponding entropy of sense distribution are in brackets.

3.1 Design

MuSeCLIR queries are sentences. They are from the selected English Wikipedia page summaries that include the chosen noun; they do not necessarily appear in the target language pages[7]. To demonstrate the necessity of context, we also experiment with just the ambiguous noun as the query (*MuSeCLIR noun*). Documents are page summaries in the target language. Each query is paired with 1 relevant document, and there are two judgement labels, 1 for relevant and 0 for not relevant.

The design of train, validation and test sets are different. In train and validation sets, each query pairs with document candidates as determined by the word sense. The set of documents to rank for a given query contains all of the document for the correct translation, together with a random selection of irrelevant documents. It is harder to spot irrelevant documents in MuSeCLIR during test time. For each test set, the set of documents to rank is generated separately for each noun. Thus, queries of the same noun will be ranking the same set of documents. This set includes documents of both correct and incorrect translations, together with a random selection of irrelevant documents. This design is a characteristic of MuSeCLIR that pre-existing datasets do not share.

3.2 Statistics

In the following experiments, the query language X is English and the document language Y is either an European language (French (FR), German (DE), Italian (IT)) or an Asian language (Chinese (ZH), Japanese (JA)). The scripts of Chinese documents are unified into traditional Chinese characters.

The mean entropy of sense distribution for all languages is around 0.5, meaning they are moderately ambiguous datasets (Jin et al., 2009). The

[7]Target language pages are a mixture of pages written individually and translated from the English page.

entropy of a word is measured using the probability distribution over the senses of that word. The higher the entropy, the more ambiguous the dataset. The graphs of sense distribution and entropy distribution are given in Appendix A. For Asian languages, each sense has around 5 sentences on average and about 4 sentences on average for European languages. The average sentence length is 24 tokens; documents have 300 words on average.

Following CLIRMatrix (Sun and Duh, 2020), we aimed to select 10,000 queries for training and 1,000 queries for validation and testing. When selecting training data, we first randomly sampled $10,000/(4 \times 5) = 500$ nouns, where 4 is the average number of senses per noun and 5 is the average number of sentences per sense. We then selected as training data all of the senses and sentences for each noun, resulting in the exact numbers of queries shown in Table 2.

4 Evaluation

Two methods are considered in the following experiments: BM25, an unsupervised probabilistic approach, and an mBERT ranker, a supervised neural network (Sun and Duh, 2020). Typically, there are two ranking stages in IR systems. At the initial stage, each query will search over all documents and then rerank on a subset of documents returned from the first stage. However, the central challenge of MuSeCLIR is a cross-lingual and cross-sense problem, not the conventional IR task. Baseline models are two individual ranking models. BM25 ranks the complete document collection, and following (Sun and Duh, 2020), mBERT ranker ranks 100 documents at test time.

Results are reported using MAP@10 (mean average precision), calculated using `pytrec_eval`[8] (Van Gysel and de Rijke, 2018). As the metric considered will cut off at 10, we limit BM25 to return 10 documents. Queries are translated using MUSE (Conneau et al., 2018) bilingual dictionaries per token, and the first possible translation is returned, disregarding other possible translations. Out-of-vocabulary tokens will use the original form (i.e. English). Table 3 and 4 presents results across five datasets. BM25 (trans) refers to BM25 results using the translated queries. To investigate the effect of mixed language tokens in documents, we experiment on single language documents. They are created by matching tokens within the corresponding language Unicode range.

4.1 Results

Elasticsearch - BM25 BM25 is a monolingual IR system, so performance after translation should be better. For both European (Table 3) and Asian (Table 4) languages, datasets with short queries like *MuSeCLIR noun* and *BI-139* are not improving after translation. Since short queries have fewer words for the matching process, they have more sparse representations. On the other hand, queries of *BI-139* are dominated by named entity mentions and without context. As MUSE dictionaries are not translating named entity mentions adequately, the performance of *BI-139* dropped after translation.

The number of foreign language tokens in queries and documents could be another factor affecting performances. The more foreign token found in documents, the better the models perform after translation, especially for datasets with longer search queries. *MuSeCLIR* might be an easier task for BM25 as the number of target language tokens in the document collections is higher and smaller in size than the existing datasets. For example, in the Chinese collection, as of MuSeCLIR, 75% of the tokens in the documents are Chinese characters; it is only 59% for BI-139. Hence, BM25 (trans) might have been rewarded more from token matching on *MuSeCLIR* than *WikiClir* and *BI-139*.

When ranking single language documents, overall performance decreased; thus, results reported in the previous setting have taken advantage of the English content. Similarly, MAP@10 increased after translation but is lower than in the mixed language setting. Potentially, this is caused by the decrease in document lengths, leading to shorter irrelevant documents ranked higher in the list more frequently.

Multilingual BERT ranker Unsurprisingly, mBERT ranker achieves better MAP@10. Since a multilingual language model is employed, languages are mapped into the same space, and no preceding translation is required. Across the board, *WikiClir* and *BI-139* have higher MAP@10 than the *MuSeCLIR*s. Possibly this is due to the list of documents to rank being more confusing in *MuSeCLIR* than the existing datasets.

Both CLIRMatrix and *WikiClir* label documents with more than one relevance level, but we found that it is seldom the case where a document is rel-

[8] A tool written in Python that builds on top of the standard TREC comunity evaluation tool https://github.com/usnistgov/trec_eval

Models	MuSeCLIR			MuSeCLIR noun			WikiClir			WikiClir title			BI-139*		
	FR	DE	IT	FR	DE	IT	FR	DE	IT	FR	DE	IT	FR	DE	IT
Results on original documents (mixed language)															
BM25	0.19	0.15	0.21	0.21	0.22	0.15	0.07	0.11	0.08	0.19	0.28	0.22	0.13	0.12	0.15
BM25 (trans)	0.34	0.18	0.37	0.17	0.11	0.18	0.17	0.14	0.14	0.32	0.30	0.30	0.12	0.11	0.14
mBERT ranker	0.79	**0.80**	**0.81**	**0.43**	**0.45**	**0.44**	**0.87**	**0.89**	**0.85**	0.90	**0.95**	**0.91**	**0.59**	**0.61**	**0.67**
Results on clean documents (single language)															
BM25	0.22	0.09	0.10	0.17	0.14	0.08	0.07	0.08	0.04	0.19	0.20	0.10	0.11	0.09	0.09
BM25 (trans)	0.27	0.11	0.22	0.15	0.12	0.12	0.17	0.09	0.08	0.32	0.21	0.17	0.10	0.10	0.08
mBERT ranker	**0.81**	0.76	0.77	0.42	0.41	0.39	0.85	**0.89**	0.81	**0.91**	0.94	0.89	0.58	0.59	0.59

Table 3: **MAP@10 results on retrieving documents written in European languages using English queries.** (*)Authors reported 0.84, 0.88 and 0.84 nDCG@10 for FR, DE and IT respectively, we obtained 0.84, 0.85 and 0.82.

Models	MuSeCLIR		MuSeCLIR noun		WikiClir		WikiClir title		BI-139*	
	ZH	JA	ZH	JA	ZH	JA	ZH	JA	ZH	JA
Results on original documents (mixed language)										
BM25	0.11	0.14	0.22	0.31	0.01	0.02	0.02	0.05	0.09	0.11
BM25 (trans)	0.42	0.22	0.21	0.19	0.02	0.05	0.05	0.09	0.03	0.05
mBERT ranker	**0.77**	**0.81**	0.42	**0.46**	**0.88**	**0.81**	**0.94**	**0.84**	**0.81**	**0.79**
Results on clean documents (single language)										
BM25	0.01	0.01	0	0	0.01	0	0.01	0	0	0
BM25 (trans)	0.40	0.20	0.20	0.14	0.03	0.03	0.05	0.06	0.01	0.02
mBERT ranker	**0.77**	0.76	**0.44**	0.43	0.86	0.77	0.92	0.81	0.66	0.71

Table 4: **MAP@10 results on retrieving documents written in Asian languages using English queries.** (*)Authors reported 0.84 nDCG@10 for both ZH and JA, we obtained 0.87 and 0.85 respectively.

evant to more than one query. This implies that these datasets focus more on evaluating systems' ability to position documents in the "right" order, which is a less challenging task. Moreover, existing datasets define linked documents of a page as less relevant documents. Less relevant documents do not necessarily relate to other senses of the queries and thus lower sense distribution entropy. Results demonstrated that mBERT ranker struggles more on MuSeCLIR than existing lower sense distribution entropy datasets. Observations are similar between mixed language and single language documents, and European and Asian language pairs.

Finally, we note that the contextual information in queries is crucial. There is a consistent drop in performance of approximately 50% from *MuSeCLIR* to *MuSeCLIR noun*. Without contextual information, it is impossible for systems to always choose the relevant document pertaining to the correct sense of the word. The mBERT ranker performs well on *MuSeCLIR*, indicating that it is doing well at disambiguating the nouns in the queries using the context. However, the MAP@10 of *MuSeCLIR* is not as high as an evaluation performed on *WikiClir* and *WikiClir title*, even when their queries are sentences. This would suggest there is scope to further improve CLIR systems.

5 Conclusion and Further Work

To address a deficiency in existing CLIR datasets, we introduce MuSeCLIR, a CLIR dataset that has been designed to challenge the ability of models to deal with ambiguous query terms. This dataset focused on polysemous common nouns with more than one possible translation, and which are regarded as ambiguous on Wikipedia. We argue that MuSeCLIR is a more suitable evaluation dataset for CLIR than pre-existing datasets.

Our method is replicable and extendable to other language pairs and other parts of speech. In the future, we also intend to test models that are trained with MuSeCLIR on real-world data and standard CLIR test sets.

References

Rexhina Blloshmi, Tommaso Pasini, Niccolò Campolungo, Somnath Banerjee, Roberto Navigli, and Gabriella Pasi. 2021. IR like a SIR: Sense-enhanced Information Retrieval for Multiple Languages. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 1030–1041, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Alexis Conneau, Guillaume Lample, Marc'Aurelio Ranzato, Ludovic Denoyer, and Hervé Jégou. 2018. Word translation without parallel data. In *International Conference on Learning Representations*.

Claudio Delli Bovi, Luca Telesca, and Roberto Navigli. 2015. Large-scale information extraction from textual definitions through deep syntactic and semantic analysis. *Transactions of the Association for Computational Linguistics*, 3:529–543.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, Minneapolis, Minnesota. Association for Computational Linguistics.

Peng Jin, Diana McCarthy, Rob Koeling, and John Carroll. 2009. Estimating and exploiting the entropy of sense distributions. In *Proceedings of Human Language Technologies: The 2009 Annual Conference of the North American Chapter of the Association for Computational Linguistics, Companion Volume: Short Papers*, pages 233–236, Boulder, Colorado. Association for Computational Linguistics.

Euna Jung, Jaekeol Choi, and Wonjong Rhee. 2022. Semi-siamese bi-encoder neural ranking model using lightweight fine-tuning. In *Proceedings of the ACM Web Conference 2022*, pages 502–511.

Sean MacAvaney, Andrew Yates, Arman Cohan, and Nazli Goharian. 2019. Cedr: Contextualized embeddings for document ranking. In *Proceedings of the 42nd International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR'19, page 1101–1104, New York, NY, USA. Association for Computing Machinery.

Benjamin M. Marshall, Colin Strine, and Alice C. Hughes. 2020. Thousands of reptile species threatened by under-regulated global trade. *Nature Communications*, (1):4738.

Suraj Nair, Eugene Yang, Dawn Lawrie, Kevin Duh, Paul McNamee, Kenton Murray, James Mayfield, and Douglas W Oard. 2022. Transfer learning approaches for building cross-language dense retrieval models. In *European Conference on Information Retrieval*, pages 382–396, Cham. Springer International Publishing.

Alessandro Raganato, Yves Scherrer, and Jörg Tiedemann. 2019. The MuCoW test suite at WMT 2019: Automatically harvested multilingual contrastive word sense disambiguation test sets for machine translation. In *Proceedings of the Fourth Conference on Machine Translation (Volume 2: Shared Task Papers, Day 1)*, pages 470–480, Florence, Italy. Association for Computational Linguistics.

Stephen Robertson and Hugo Zaragoza. 2009. The probabilistic relevance framework: Bm25 and beyond. *Found. Trends Inf. Retr.*, 3(4):333–389.

Shota Sasaki, Shuo Sun, Shigehiko Schamoni, Kevin Duh, and Kentaro Inui. 2018. Cross-lingual learning-to-rank with shared representations. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 2 (Short Papers)*, pages 458–463, New Orleans, Louisiana. Association for Computational Linguistics.

Shuo Sun and Kevin Duh. 2020. CLIRMatrix: A massively large collection of bilingual and multilingual datasets for cross-lingual information retrieval. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 4160–4170, Online. Association for Computational Linguistics.

Christophe Van Gysel and Maarten de Rijke. 2018. Pytrec_eval: An extremely fast python interface to trec_eval. In *SIGIR*. ACM.

Puxuan Yu, Hongliang Fei, and Ping Li. 2021. Cross-lingual language model pretraining for retrieval. In *Proceedings of the Web Conference 2021*, WWW '21, page 1029–1039, New York, NY, USA. Association for Computing Machinery.

Dong Zhou, Mark Truran, and Tim J Brailsford. 2007. Ambiguity and unknown term translation in clir.

A Sense Distribution Plots

Here, we provide plots of sense distribution and plots of the entropy of sense distribution across our language collection. The entropy of a word (w) is measured using the probability distribution (p) over the senses of the word, following equation 1.

$$H(\text{senses}(w)) = -\sum_{ws_i \in \text{senses}(w)} p(ws_i) \log\left(p(ws_i)\right) \quad (1)$$

Each dot on the sense entropy distribution represents a noun. Essentially, the entropy distribution plot is the reverse version of the corresponding sense distribution plot.

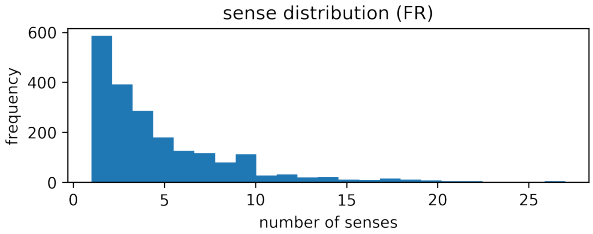

(a) Frequency distribution of different sense counts.

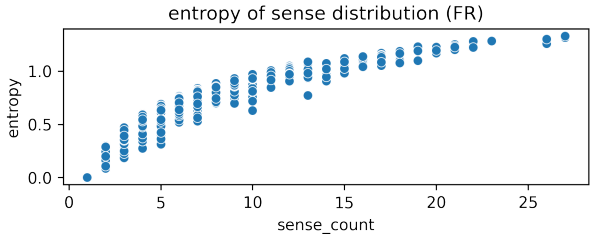

(b) Entropy of sense distribution across different number of sense.

Figure 1: Plots of the French collection in MuSeCLIR.

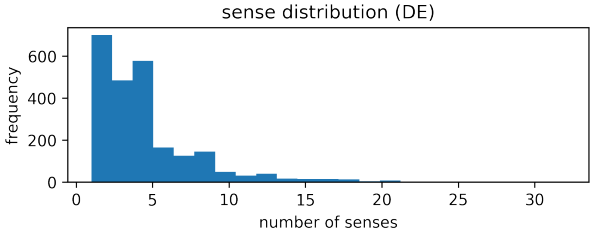

(a) Frequency distribution of different sense counts.

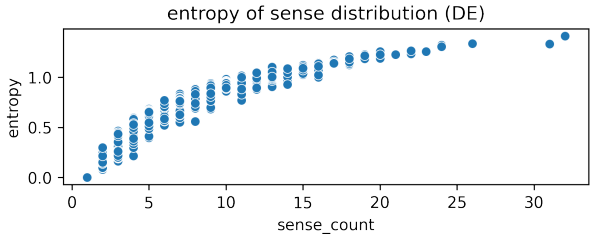

(b) Entropy of sense distribution across different number of sense.

Figure 2: Plots of the German collection in MuSeCLIR.

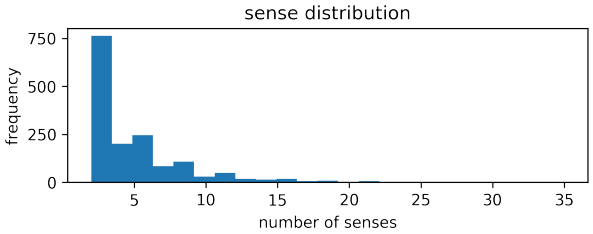

(a) Frequency distribution of different sense counts.

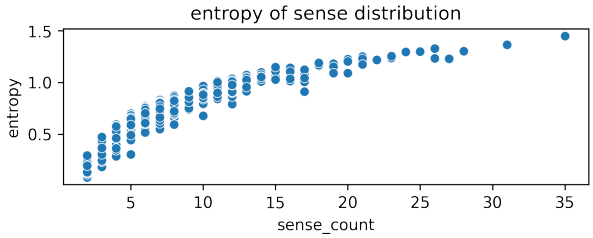

(b) Entropy of sense distribution across different number of sense.

Figure 3: Plots of the Italian collection in MuSeCLIR.

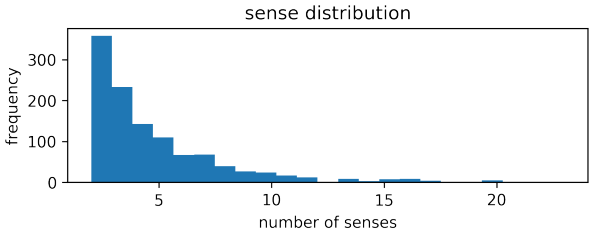

(a) Frequency distribution of different sense counts.

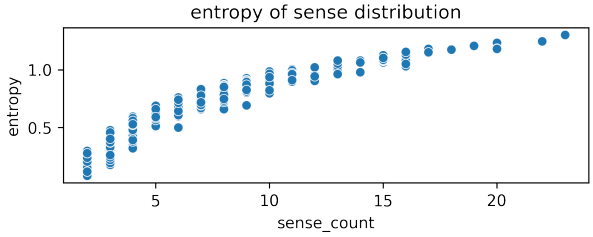

(b) Entropy of sense distribution across different number of sense.

Figure 4: Plots of the Chinese collection in MuSeCLIR.

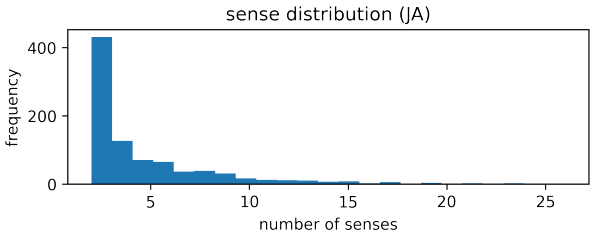

(a) Frequency distribution of different sense counts.

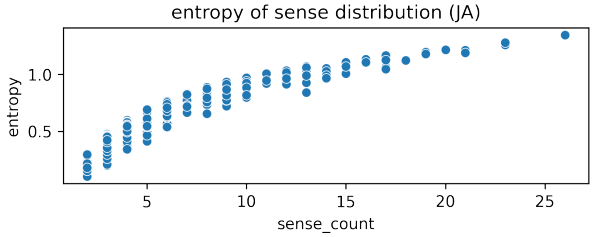

(b) Entropy of sense distribution across different number of sense.

Figure 5: Plots of the Japanese collection in MuSeCLIR.

Complicate then Simplify: A Novel Way to Explore Pre-trained Models for Text Classification

Xu Zhang and **Zejie Liu** and **Yanzheng Xiang** and **Deyu Zhou***
School of Computer Science and Engineering, Key Laboratory of Computer Network and Information Integration, Ministry of Education, Southeast University, China
{xuzhang123,liuzejie,yz_xiang,d.zhou}@seu.edu.cn

Abstract

In the developing context of pre-trained models (PTMs), the performance of text classification has been continuously improved by directly employing the features generated by PTMs. However, such a way might not fully explore the knowledge in PTMs as it is constrained by the difficulty of the task. Compared to a difficult task, the learning algorithms tend to saturate early on the simple task. Moreover, the native sentence representations derived from BERT are prone to be collapsed and directly employing such representation for text classification might fail to fully capture discriminative features. In order to address these issues, in this paper we propose a novel framework for text classification which implements a two-stage training strategy. In the pre-training stage, auxiliary labels are introduced to increase the task difficulties and to fully exploit the knowledge in the pre-trained model. In the fine-tuning stage, the textual representation learned in the pre-training stage is employed and the classifier is fine-tuned to obtain better classification performance. Experiments were conducted on six text classification corpora and the results showed that the proposed framework outperformed several state-of-the-art baselines.

1 Introduction

Text classification is a fundamental task in the field of natural language processing and is widely employed in various tasks such as question answering, sentiment analysis, and information retrieval. With the continuous development of machine learning algorithms, especially the success of deep learning methods, text classification has been significantly improved, e.g. CNNs (Kim, 2014; Lai et al., 2019), RNNs (Chen et al., 2017; Zhang et al., 2020), BERTs (Cui et al., 2019, 2020; Sun et al., 2021), etc. Recently, pre-trained models have been shining in classification-based natural language processing tasks (Cui et al., 2019, 2020; Sun et al., 2021).

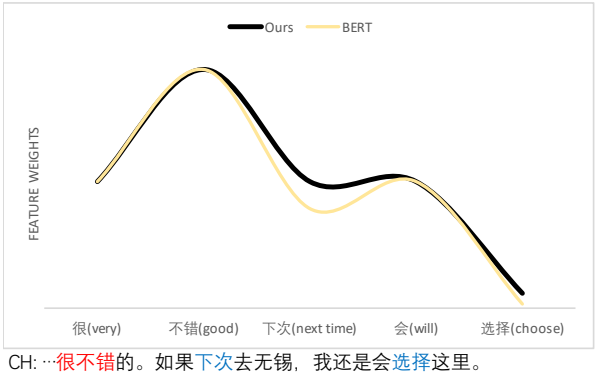

CH: …很不错的。如果下次去无锡，我还是会选择这里。
EN: …were very good. If I go to Wuxi next time, I would still choose this place.

Figure 1: An Example from the ChnSentiCorp corpus.

The advent of BERT has led to an effective enhancement of textual feature representation. A series of improved pre-trained models have been proposed, e.g. RoBERTa (Liu et al., 2019), MacBERT (Cui et al., 2020), ERNIE (Sun et al., 2020), ChineseBERT (Sun et al., 2021). For example, ERNIE learns real-world semantic knowledge by modeling words, entities and entity relationships in massive amounts of data (Sun et al., 2020). Sun et al. (2021) proposed a large-scale Chinese pre-trained model that incorporates glyph and pinyin information. The above works improve the textual representation of the pre-trained model by introducing external knowledge, without fully exploring the semantic representation in the existing pre-trained model. The recent work, prompt learning, is the technique of making better use of the knowledge from the pre-trained model by adding additional texts to the input (Liu et al., 2021). Like prompt learning, we propose to further extract more meaningful textual representations from the pre-trained model, i.e. to make the extracted textual semantic representation more discriminative for classification.

Although the current pre-trained model already obtain relatively good textual representation, it is

*Corresponding author

Proceedings of the 29th International Conference on Computational Linguistics, pages 1136–1145
October 12–17, 2022.

still possible to further explore the information in the pre-trained representation based on our observation. As shown in Figure 1, for sentiment analysis, there are two sets of features in a sentence that indicate its positive emotional polarity, “很不错(very good)” and “下次 选择(next time choose)”. Words like “很 不错(very good)” explicitly express positive sentiment. Features like these, which are closely related to category labels, are given higher weight during training. While more implicit features like “下次(next time)” and “选择(choose)” are easily ignored as shown by the green curve in Figure 1. However, by using the two-stage framework proposed in this paper, the weights of implicit discriminative features are further highlighted without weakening the weights of the most discriminative feature as shown in the black curve in Figure 1.

Meanwhile, we notice that Yan et al. (2021) found the word representation space of BERT to be anisotropic, with high-frequency words clustered together and close to the origin, while low-frequency words were sparsely scattered. When averaging token embeddings, those high-frequency words dominate the sentence representation, inducing a bias against their actual semantics. Such phenomenon has also been observed in some previous work (Gao et al., 2019; Wang et al., 2020a; Li et al., 2020). Therefore, directly employing such representation for text classification might fail to fully capture discriminative features.

Therefore, in this paper, we consider the extracting of semantic features from a cognitive perspective by introducing auxiliary labels and constructing pre-training and fine-tuning strategies based on pre-trained models. We devise a novel approach to perform a secondary pre-training based on the pre-trained model and then fine-tuning for text classification which is similar to the process of gaining new insights through restudying old material. In the pre-training stage, the model learns a better representation of the task under consideration. In the fine-tuning stage, the classifier is fine-tuned by applying the text feature representations obtained from the pre-training stage. To fully exploit the discriminative features in the pre-trained model, in the pre-training stage, auxiliary labels are constructed to take fine-grained semantic categories into account. The introduction of auxiliary labels makes the information entropy increase. Knowledge in the pre-trained model is fully mined for a more effective discriminative semantic representation.

The main contributions are listed as follows.

- We propose a novel framework for text classification which implements a two-stage training strategy and enables "experience accumulation" and "practice what you learned" without introducing additional knowledge.
- In the pre-training stage, auxiliary labels are integrated to increase the training challenge and to exploit the knowledge in the pre-trained model fully.
- The validity of the framework is verified on seven benchmark datasets, and the proposed framework achieves better performance than several state-of-the-art baselines.

The reminder of the paper is structured as follows. Some related work is briefly reviewed in Section 2. The detailed implementation of our framework is described in Section 3. Section 4 reports the experiments and results and Section 5 shows further analysis and discussion. Finally, the paper is concluded in Section 6.

2 Related Work

The development of deep learning has led to significant improvements in text classification, and some of the more widely employed deep learning models for text classification tasks are CNNs (Wang et al., 2018; Lai et al., 2019), RNNs (Chen et al., 2017; Sachan et al., 2019; Zhang et al., 2020), and pre-trained models (Cui et al., 2019; Liu et al., 2019; Sun et al., 2021). In recent years, pre-trained models have shown excellent performance in the field of text classification. Whether CNNs, RNNs, or more recently pre-trained models, the purpose of employing deep models is to efficiently capture the textual semantic representation.

2.1 Traditional Deep Learning methods

Text representation is the basis for text classification. The first work to introduce CNNs into NLP was done by Kim (2014), and the key to the features captured by a CNN is the sliding window covered by the convolutional kernel. Johnson and Zhang (2017) proposed the Deep Pyramidal Convolutional Neural Network (DPCNN), which can effectively extract remote relational features from the text. In the process of text feature extraction employing convolutional operations, the semantic relations of sentences would be lost. Ma et al. (2015)

proposed to employ dependent syntactic trees to extract the semantic feature relations, instead of just employing adjacent word representations as feature representations. CNNs can extract local features from global information when employed for text classification, but they are unable to capture long-term dependencies, whereas RNNs can. Zhang et al. (2018) proposed a sentence-state based LSTM that incorporates the semantic relevance of words and sentences. Models such as CNNs and LSTMs capture word sense information well in locally continuous word sequences but may ignore global word co-occurrences in corpora with discontinuous and long-term semantics dependencies. Yao et al. (2019) proposed a graph-based convolutional neural network structure (GCN), which exploits global word co-occurrence information not previously considered by other models and demonstrates better robustness with less training data. RNNs suffer from gradient explosion and gradient disappearance, and cannot effectively handle long-term context-dependence. The attention mechanism can characterize the target location by linearly weighting the features of the contextual source sequence. Bahdanau et al. (2015) first applied attention mechanisms to the field of natural language processing. Yang et al. (2016) proposed a hierarchical attention mechanism model for text classification tasks, acting at the word and sentence levels respectively. With the development of deep learning, neural networks are widely employed in NLP tasks, such as the aforementioned CNNs, RNNs, GNNs and attention mechanisms, but as the available datasets are small for most supervised NLP tasks, the above models are "shallow" for NLP tasks, making it difficult to extract sufficiently rich textual representations.

2.2 Pre-trained Models

The advent of pre-trained models (PTMs) has ushered in a new era of NLP, with extensive work showing that pre-trained models on large corpora can learn generic language representations and avoid training from scratch when solving downstream NLP tasks (Liu et al., 2019; Sun et al., 2020, 2021). Since BERT, complementary pre-trained models have been designed to integrate external knowledge into PTMs for better textual representations. ERNIE combines pre-trained entity embeddings in the knowledge graph with corresponding entity mentions in text to enhance the text representation (Zhang et al., 2019). KnowBERT merges entity representations in an end-to-end manner (Peters et al., 2019). KEPLER unites knowledge embeddings with language model objects (Wang et al., 2021). K-BERT differs from the above models by introducing structured information from the knowledge graph through entity embeddings (Liu et al., 2020). It obtains an expanded tree input to the BERT by directly introducing relevant triples from the knowledge graph into the sentence. K-Adapter independently trains different adapters for different pre-trained models to introduce multiple knowledge, in order to address the forgetting problem that occurs when the above models are injected with multiple knowledge (Wang et al., 2020b). In contrast to the above approach, Qin et al. (2020) proposed the use of feature projection methods based on BERT to further improve text representation without introducing external knowledge. They consider fully mining the existing knowledge in the pre-trained model to make the feature representations involved in classification more discriminative.

3 Model

The overall framework is shown in Figure 2. The whole framework is divided into two stages: pre-training and fine-tuning. In the pre-training stage, auxiliary labels are introduced to artificially boost the training difficulty, which can better tap the knowledge from pre-trained models and obtain a more discriminative textual feature representation. In the fine-tuning stage, the textual representation pre-trained in the pre-training stage is employed and the classifier is fine-tuned to obtain better classification performance.

3.1 Problem Definition

Suppose that we have a K-class classification task, a training instance can be denoted as (x_i, y_i) for $i = 1, ..., N$ and $y_i \in \{1, 2, \ldots, K\}$. Here, we introduce auxiliary labels, as shown in Figure 2. Suppose we have an encoder $E(\cdot)$.

$$R = \mathrm{E}(x) \tag{1}$$

Firstly, the model is trained employing auxiliary labels to obtain discriminative textual representation R. Afterwards the classification is performed by the textual representation R, and the original label y_i.

Auxiliary Labels: The auxiliary labels are introduced through expanding the target labels by com-

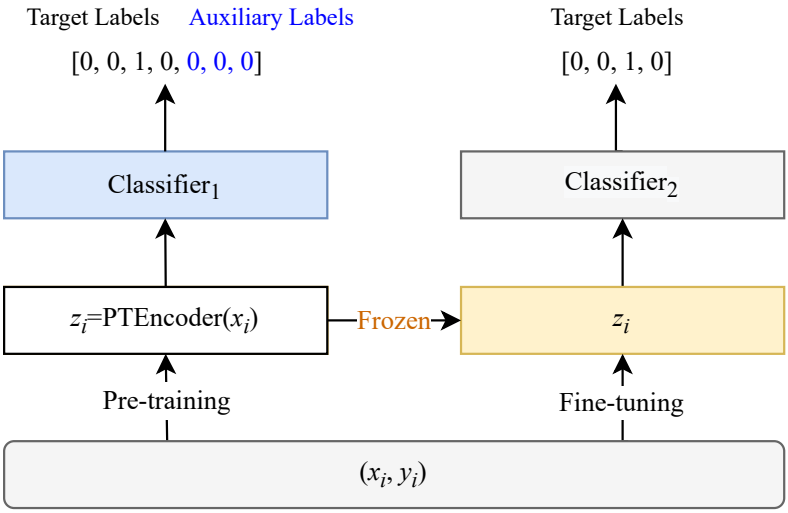

Figure 2: Architecture of the proposed approach.

plementary zeros, simulating the existence of fine-grained label categories. During the pre-training stage, the auxiliary labels also have predicted probability values. Therefore the probability values of the predicted labels corresponding to the target labels fluctuate and the loss function decreases at a slower rate during pre-training. To gain better classification results, the model is forced to learn a better textual representation.

Information Entropy: Suppose that, for a K classification task in which the prediction probabilities of the false labels are all γ, the prediction probability of the true label is (1 - (k -1)γ). The information entropy is shown below.

$$H_k = -\sum_{i=1}^{k-1} \gamma \log \gamma - (1-(k-1)*\gamma) \log(1-(k-1)*\gamma) \quad (2)$$

Similarly, for the case of introducing n auxiliary labels, the information entropy is as follows:

$$H_{k+n} = -\sum_{i=1}^{k+n-1} \gamma \log \gamma - (1-(k+n-1)*\gamma) \log(1-(k+n-1)*\gamma) \quad (3)$$

It can be inferred that the introduction of auxiliary labels makes the information entropy increase as shown in Equation 4. The lower the deviation of H_{k+n} and H_k, the smaller the difference between the two distributions. After introducing the auxiliary labels, the model needs to mine the knowledge in the pre-trained model in-depth, in order to reduce the gap with the true distribution.

$$H_{k+n} - H_k > 0 \quad (4)$$

3.2 Pre-training Pre-trained Models and Fine-tuning

The traditional text classification strategy is to obtain the textual semantic representation through the Encoder Network $Encoder(\cdot)$ and then employ the Classifier $Classifier(\cdot)$ to make predictions.

$$z_i = \text{Encoder}(x_i) \quad (5)$$

$$c_i = \text{Classifier}(z_i) \quad (6)$$

θ is the parameter of $Encoder(\cdot)$ and β is the parameter of $Classifier(\cdot)$. Their method is described as follows:

$$\mathscr{L}(\theta,\beta) = -\sum_{i=1}^{N}\sum_{k=1}^{K} 1(y_i = k) \log(k \mid c_i) \quad (7)$$

Unlike the traditional classification models described above, our approach adopts a two-stage training strategy.

Pre-training : Similarly, the input text x_i needs to be represented as a textual semantic representation employing a parameters trainable encoder $PTEncoder(\cdot)$. Here, one point to focus on is that the parameters of the encoder are trainable.

$$z_i = \text{PTEncoder}(x_i) \quad (8)$$

$PTEncoder(\cdot)$, which maps x_i to a discriminative representation vector shown in Equation 8.

Next, label prediction is to be performed employing z_i. Although the auxiliary labels are introduced, this is only done to introduce interference terms, in order for the $PTEncoder(\cdot)$ to be trained to obtain

more discriminative text features for the subsequent classification.

$$c1_i = \text{Classifier}_1(z_i) \tag{9}$$

θ_1 is the parameter of $PTEncoder(\cdot)$ and β_1 is the parameter of $Classifier_1(\cdot)$. Their method is described as follows:

$$\mathscr{L}_\infty(\theta', \beta') = -\sum_{i=1}^{N}\sum_{k=1}^{K+j} 1\,(y1_i = k)\log\,(k \mid c1_i) \tag{10}$$

where j is the number of auxiliary labels we introduce on top of the original K targets.

In addition, there is no other effect on the classification task, as the auxiliary labels are represented in the one-hot labels as 0. Therefore, in the specific task, only the original categories are involved in the calculation of the cross-entropy loss function, and the auxiliary labels are not involved.

Fine-tuning : With the first stage (Pre-training), better parameters for the $PTEncoder(\cdot)$ can be pre-trained, by which a better discriminative textual representation z_i can be obtained.

Next, the trained textual representation z_i is employed for conventional classification, i.e., no auxiliary labels are introduced and classification is performed according to the given label categories.

Here, we only perform further fine-tuning for the classifier:

$$c2_i = \text{Classifier}_2(z_i) \tag{11}$$

θ_1 is the parameter of $PTEncoder(\cdot)$ and β_2 is the parameter of $Classifier_2(\cdot)$. Their method is described as follows:

$$\mathscr{L}_\in(\theta_1, \beta_2) = -\sum_{n=1}^{N}\sum_{k=1}^{K} 1\,(y_i = k)\log\,(k \mid c2_i) \tag{12}$$

A summary of the two-stage training strategy described above. (1) In the pre-training stage, auxiliary labels for classification are constructed and the encoder is pre-trained employing the auxiliary labels to obtain a more discriminative feature representation z_i. (2) In the fine-tuning stage, the textual representation z_i is employed for classification, where only fine-tuning is done for the classifier and the parameters of the text representation z_i are no longer updated to obtain better classification performance.

3.3 Textual Representation

BERT (Devlin et al., 2019) is a multilayered attention-assisted bidirectional transformer encoder model based on the original transformer model (Vaswani et al., 2017). During pre-training, BERT employed two objectives: masked language model (MLM) and next sentence prediction (NSP). The NSP is employed to predict whether two segments follow each other. The goal of NSP is to improve the performance of downstream tasks such as natural language inference (Bowman et al., 2015), which entails reasoning about the relationship between pairs of sentences. NSP is a good fit with our matching-based QA task and the matching task. Therefore, we choose the BERT as the encoding model.

The original authors of BERT proposed an upgraded version of BERT, including Whole Word Masking (WWM), which alleviates the disadvantage of masking some WordPiece tokens in pre-trained BERT. Cui et al. employed the whole word masking strategy for Chinese BERT and published a series of Chinese pre-trained models (Cui et al., 2019). The experimental performance shows that the proposed pre-trained model yields substantial improvements over BERT and ERNIE on various NLP tasks. They adapted whole-word masking in Chinese text by masking whole words instead of Chinese characters.

In view of the excellent performance achieved by BERT-wwm in Chinese tasks (Cui et al., 2019), an improved version of the BERT model proposed by Cui et al. is employed as the Encoder in our work.

4 Experiment

4.1 Datasets

We conducted experiments on six Chinese text classification datasets, including two sentence semantic matching datasets (BQ (Chen et al., 2018) and LCQMC (Liu et al., 2018)), one text classification datasets (TNEWS (Xu et al., 2020)), one sentiment classification dataset (ChnSentiCorp [1]), and two Chinese question answering datasets from the NLPCC-2016 evaluation task (Duan, 2016).

Sentence semantic matching task: BQ is the largest Chinese question matching dataset in the banking domain. LCQMC is the largest Chinese

[1] https://github.com/pengming617/bert_classification/tree/master/data

semantic matching dataset available and obtained from the Baidu Knows question and answer community.

Text classification task: TNEWS selects from the news section of Today's Headlines, with 15 news categories, including travel, education, finance, military and more. ChnSentiCorp is a Chinese sentiment classification dataset with more than 7000 hotel reviews, 5000 positive reviews and 2000 negative reviews

Question answering task: DBQA is a document-based question answering dataset. These candidate sentences were extracted from web pages and tended to be much longer than the questions, with many irrelevant sentences. KBQA is a knowledge base based question answering dataset, and each question contained only one golden predicate.

4.2 Baselines

The following approaches are employed as the baselines, including BERT-wwm and BERT-wwm-ext (employing extended data, including Chinese Wikipedia, other encyclopedias, news, QAs and other data, with a total word count of 5.4B) (Cui et al., 2019). To evaluate the proposed approach, we additionally selected a series of pre-trained models for comparison, with RoBERTa-base, RoBERTa-large (Liu et al., 2019), MacBERT-base, MacBERT-large (Cui et al., 2020), ChineseBERT-base, and ChineseBERT-large (Sun et al., 2021).

Our approach is customized to several versions to evaluate its performance, as follows: We improve on the BERT-wwm, BERT-wwm-ext by introducing our approach to obtain RE-BERT-wwm, RE-BERT-wwm-ext. MRE-BERT fuses the textual representations of the BERT-wwm and the BERT-wwm-ext models and then introduces our approach for optimization.

4.3 Experiment Setup

The experimental setup is shown in Table 1. We found experimentally that the number of auxiliary labels is set differently for different pre-trained models and various tasks. As a hyperparameter, it needs to be adapted to the different training tasks. For LCQMC and TNEWS datasets with relatively clear classification goals and little ambiguity in the annotated data, introducing auxiliary labels did not significantly improve classification but the two-stage repetitive operation based on learning similar to human cognitive skills still improved the model's effectiveness. In addition, the batch size is set to 64, and Adam with parameters 2e-5 is employed as the optimizer (Kingma and Ba, 2015). All experiments were performed on two Nvidia Tesla T4 GPUs.

4.4 Experiment Results

4.4.1 Matching based QA Task

As shown in Table 2, following the work of lai et al. (Lai et al., 2019), we have implemented BERT-based Chinese KBQA and DBQA tasks based on their work. In their work, Lattice CNNs were employed as encoder, and we experimented with BERTs-base replacing Lattice CNNs as the baseline for our work.

As shown in Table 2, we have improved the experiments by employing our approach compared to the original BERT-wwm and BERT-wwm-ext. The experiments show that training through two stages is effective and our proposed approach of introducing auxiliary labels In the pre-training stage is feasible. The introduction of auxiliary labels allows for more meaningful discriminative features for classification in feature extraction.

M-BERT unites the textual semantic representations of two pre-trained models, BERT-wwm and BERT-wwm-ext. Compared to employing a single pre-trained model, BERT-wwm or BERT-wwm-ext, there is an improvement in the experimental results. The experiment also validates our previous consideration that there is variability in the textual semantic representation obtained by different pre-trained models and that an improvement can be achieved by combining multiple pre-trained models.

MRE-BERT has a significant improvement over the two BERT-base models, BERT-wwm and BERT-wwm-ext, on both DBQA and KBQA question and answers datasets. Compared to BERT-wwm, MRE-BERT has a more than 1% improvement on both datasets. There is also a substantial effect improvement in each evaluation metric compared to BERT-wwm-ext.

4.4.2 Text Classification Task

In addition, we conducted corresponding experiments on two text classification datasets, TNEWS as well as ChnSentiCorp, which have relatively few category labels.

As shown in Table 3, for the TNEWS and ChnSentiCorp datasets, which have relatively few categories, our proposed approach achieves good performance on both datasets. The experimental results for the TNEWS dataset on BERT-wwm as

Dataset	Scale (train/valid/test)	Model	N^a	N^t	Epoch
ChnSentiCorp	9.6K/1.2K/1.2K	RE-BERT-wwm	4	2	10
		RE-BERT-wwm-ext	4	2	10
		MRE-BERT	16	2	10
TNEWS	12.1k/2.6k/2.6k	RE-BERT-wwm	119	119	6
		RE-BERT-wwm-ext	15	15	3
		MRE-BERT	15	15	3
DBQA	182k/-/123k	RE-BERT-wwm	4	2	2
		RE-BERT-wwm-ext	4	2	2
		MRE-BERT	128	2	2
KBQA	273k/-/156k	RE-BERT-wwm	4	2	2
		RE-BERT-wwm-ext	16	2	2
		MRE-BERT	6	2	2
LCQMC	238.7k/8.8k/12.5k	MRE-BERT	2	2	3
BQ	100k/10k/10k	MRE-BERT	12	2	5

Table 1: Experimental parameter settings. N^a means the whole number of auxiliary and target labels, and N^t number of target labels

	DBQA			KBQA	
Model	**P@1**	**MRR**	**MAP**	**P@1**	**MRR**
BERT	90.06	93.79	93.75	92.88	95.69
RE-BERT	90.20	93.80	93.77	93.80	96.28
BERT^o	90.95	94.38	94.35	93.23	95.90
RE-BERT^o	91.18	94.57	94.54	94.01	96.42
M-BERT	91.08	94.36	94.31	93.75	96.31
MRE-BERT	**91.91**	**95.03**	**95.00**	**94.16**	**96.56**

Table 2: Experimental results on matching based QA task. BERT represents BERT-wwm and o represents models pre-trained on extended data.

well as BERT-wwm-ext are from CLUE[2] (Xu et al., 2020), and the experimental results for ChnSentiCorp are from ChineseBERT (Sun et al., 2021). The experimental results show that MRE-BERT has a better effect compared to the two base pre-trained models BERT-wwm and BERT-www-ext.

	TNEWS	ChnSentiCorp
Model	**Valid**	**Test**
BERT	56.09	95.4
BERT^o	56.77	95.3
RE-BERT	56.87	94.8
RE-BERT^o	**57.40**	95.6
MRE-BERT	56.95	**95.8**

Table 3: Experimental on text classification task. BERT represents BERT-wwm and o represents models pre-trained on extended data.

[2]https://github.com/CLUEbenchmark/CLUE

5 Analysis and Discussion

5.1 Comparison of Different Auxiliary Labels

In order to verify the effectiveness of our approach and the effect of the auxiliary labels, we conducted some targeted experiments on the KBQA dataset.

As shown in Figure 3, we have selected a series of samples for classification after adding auxiliary labels In the pre-training stage (pre-training) for RE-BERT-wwm, BERT-wwm-ext and MRE-BERT models respectively. The w/o RE model shows the effect without our approach, from which it can be seen that models can be effectively improved by introducing auxiliary labels. With the experimental results in Figure 3, we have verified the effectiveness of our approach. The introduction of auxiliary labels helps to extract textual semantic representations efficiently, and the number of auxiliary labels needs to be set according to the needs of different models and tasks.

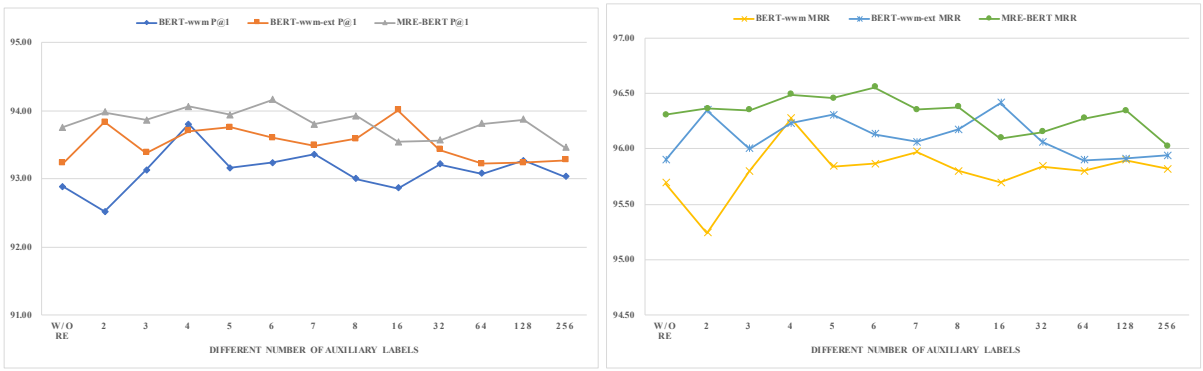

(a) Comparison of the effectiveness on the P@1.

(b) Comparison of the effectiveness on the MRR.

Figure 3: Comparison of the effects of different models on KBQA dataset after the addition of auxiliary labels In the pre-training stage.

5.2 Verification Experiments

Model	BQ		LCQMC	
	Valid	**Test**	**Valid**	**Test**
	base			
BERT	86.1	85.2	89.4	87.0
BERT^o	86.4	**85.3**	89.6	87.1
RoBERTa^o	86.0	85.0	89.0	86.4
MacBERT	86.0	85.2	89.5	87.0
ChineseBERT	86.4	85.2	89.8	**87.4**
MRE-BERT	86.3	**85.6**	90.4	**87.4**
	large			
RoBERTa^o	86.3	85.8	90.4	87.0
MacBERT	86.2	85.6	90.6	87.6
ChineseBERT	86.5	86.0	90.5	**87.8**
	base			
MRE-BERT$_F$	86.7	**86.2**	89.9	**87.7**

Table 4: Validation experiments on text matching task. BERT represents BERT-wwm, o represents models pre-trained on extended data, and MRE-BERT$_F$ represents MRE-BERT with adversarial perturbation.

To further validate the effectiveness of MRE-BERT, some related experiments are conducted on two sentence semantic matching datasets and compared with current state-of-the-art pre-trained models. The experimental results show that MRE-BERT has achieved better performance compared to BERT-wwm, BERT-wwm-ext, RoBERTa-wwm-ext (Cui et al., 2019), MacBERT-base (Cui et al., 2020), and ChineseBERT-base (Sun et al., 2021). Furthermore, adding random perturbations to its embedding layer based on pre-trained models is effective in many tasks, expanding the training data by perturbing the samples and being able to regularise the model effectively (Ju et al., 2019). We also tried to introduce adversarial training In the pre-training stage, and the performance was further improved. We notice that ChineseBERT considers two important aspects specific to Chinese: glyphs and pinyin, which carry important syntactic and semantic information for language understanding. It incorporates both glyph and pinyin information into pre-trained language models, achieving new SOTA performance in a wide range of Chinese NLP tasks. However, our proposed BERT-base based framework achieves comparable results to ChineseBERT.

6 Conclusion

For simple text classification tasks, we propose a novel framework for simple text classification tasks, which implements a two-stage training strategy, including pre-training based on pre-trained models and fine-tuning for classification. In the pre-training stage, auxiliary labels can be integrated to increase the training challenge and to fully exploit the knowledge in the pre-trained model. Experiments on six datasets depict that our approach outperforms the baseline BERT model for simple classification tasks. Furthermore, in the sentence semantic matching task, after adding adversarial perturbations to the embedding layer only, our basic version of the MRE-BERT model achieves promising performance, better than or equivalent to large versions of the pre-trained model, but with much fewer parameters than them. In the future, We will investigate the generalizability of our model to other classification tasks.

Acknowledgements

The authors would like to thank the anonymous reviewers for their insightful comments. This work was funded by the National Natural Science Foundation of China (62176053).

References

Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. 2015. Neural machine translation by jointly learning to align and translate. In *3rd International Conference on Learning Representations*.

Samuel Bowman, Gabor Angeli, Christopher Potts, and Christopher D Manning. 2015. A large annotated corpus for learning natural language inference. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 632–642.

Jing Chen, Qingcai Chen, Xin Liu, Haijun Yang, Daohe Lu, and Buzhou Tang. 2018. The bq corpus: A large-scale domain-specific chinese corpus for sentence semantic equivalence identification. In *Proceedings of the 2018 conference on empirical methods in natural language processing*, pages 4946–4951.

Qian Chen, Xiaodan Zhu, Zhen-Hua Ling, Si Wei, Hui Jiang, and Diana Inkpen. 2017. Enhanced lstm for natural language inference. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1657–1668.

Yiming Cui, Wanxiang Che, Ting Liu, Bing Qin, Shijin Wang, and Guoping Hu. 2020. Revisiting pre-trained models for chinese natural language processing. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: Findings*, pages 657–668.

Yiming Cui, Wanxiang Che, Ting Liu, Bing Qin, Ziqing Yang, Shijin Wang, and Guoping Hu. 2019. Pre-training with whole word masking for chinese bert. *arXiv preprint arXiv:1906.08101*.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. Bert: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186.

Nan Duan. 2016. Overview of the nlpcc-iccpol 2016 shared task: open domain chinese question answering. In *Natural Language Understanding and Intelligent Applications*, pages 942–948. Springer.

Jun Gao, Di He, Xu Tan, Tao Qin, Liwei Wang, and Tieyan Liu. 2019. Representation degeneration problem in training natural language generation models. In *International Conference on Learning Representations*.

Rie Johnson and Tong Zhang. 2017. Deep pyramid convolutional neural networks for text categorization. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 562–570.

Ying Ju, Fubang Zhao, Shijie Chen, Bowen Zheng, Xuefeng Yang, and Yunfeng Liu. 2019. Technical report on conversational question answering. *arXiv preprint arXiv:1909.10772*.

Yoon Kim. 2014. Convolutional neural networks for sentence classification. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1746–1751, Doha, Qatar. Association for Computational Linguistics.

Diederik P Kingma and Jimmy Ba. 2015. Adam: A method for stochastic optimization. In *Proceedings of the International Conference on Learning Representations*.

Yuxuan Lai, Yansong Feng, Xiaohan Yu, Zheng Wang, Kun Xu, and Dongyan Zhao. 2019. Lattice cnns for matching based chinese question answering. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, pages 6634–6641.

Bohan Li, Hao Zhou, Junxian He, Mingxuan Wang, Yiming Yang, and Lei Li. 2020. On the sentence embeddings from pre-trained language models. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 9119–9130.

Pengfei Liu, Weizhe Yuan, Jinlan Fu, Zhengbao Jiang, Hiroaki Hayashi, and Graham Neubig. 2021. Pre-train, prompt, and predict: A systematic survey of prompting methods in natural language processing. *arXiv preprint arXiv:2107.13586*.

Weijie Liu, Peng Zhou, Zhe Zhao, Zhiruo Wang, Qi Ju, Haotang Deng, and Ping Wang. 2020. K-bert: Enabling language representation with knowledge graph. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pages 2901–2908.

Xin Liu, Qingcai Chen, Chong Deng, Huajun Zeng, Jing Chen, Dongfang Li, and Buzhou Tang. 2018. Lcqmc: A large-scale chinese question matching corpus. In *Proceedings of the 27th International Conference on Computational Linguistics*, pages 1952–1962.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Roberta: A robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692*.

Mingbo Ma, Liang Huang, Bowen Zhou, and Bing Xiang. 2015. Dependency-based convolutional neural networks for sentence embedding. In *Proceedings*

of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 2: Short Papers), pages 174–179.

Matthew E Peters, Mark Neumann, Robert Logan, Roy Schwartz, Vidur Joshi, Sameer Singh, and Noah A Smith. 2019. Knowledge enhanced contextual word representations. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 43–54.

Qi Qin, Wenpeng Hu, and Bing Liu. 2020. Feature projection for improved text classification. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 8161–8171.

Devendra Singh Sachan, Manzil Zaheer, and Ruslan Salakhutdinov. 2019. Revisiting lstm networks for semi-supervised text classification via mixed objective function. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, pages 6940–6948.

Yu Sun, Shuohuan Wang, Yukun Li, Shikun Feng, Hao Tian, Hua Wu, and Haifeng Wang. 2020. Ernie 2.0: A continual pre-training framework for language understanding. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pages 8968–8975.

Zijun Sun, Xiaoya Li, Xiaofei Sun, Yuxian Meng, Xiang Ao, Qing He, Fei Wu, and Jiwei Li. 2021. Chinesebert: Chinese pretraining enhanced by glyph and pinyin information. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 2065–2075.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Proceedings of the 31st International Conference on Neural Information Processing Systems*, pages 6000–6010.

Lingxiao Wang, Jing Huang, Kevin Huang, Ziniu Hu, Guangtao Wang, and Quanquan Gu. 2020a. Improving neural language generation with spectrum control. In *International Conference on Learning Representations*.

Ruize Wang, Duyu Tang, Nan Duan, Zhongyu Wei, Xuanjing Huang, Guihong Cao, Daxin Jiang, Ming Zhou, et al. 2020b. K-adapter: Infusing knowledge into pre-trained models with adapters. *arXiv preprint arXiv:2002.01808*.

Shiyao Wang, Minlie Huang, and Zhidong Deng. 2018. Densely connected cnn with multi-scale feature attention for text classification. In *Proceedings of the 27th International Joint Conference on Artificial Intelligence*, pages 4468–4474.

Xiaozhi Wang, Tianyu Gao, Zhaocheng Zhu, Zhengyan Zhang, Zhiyuan Liu, Juanzi Li, and Jian Tang. 2021. Kepler: A unified model for knowledge embedding and pre-trained language representation. *Transactions of the Association for Computational Linguistics*, 9:176–194.

Liang Xu, Hai Hu, Xuanwei Zhang, Lu Li, Chenjie Cao, Yudong Li, Yechen Xu, Kai Sun, Dian Yu, Cong Yu, et al. 2020. Clue: A chinese language understanding evaluation benchmark. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 4762–4772.

Yuanmeng Yan, Rumei Li, Sirui Wang, Fuzheng Zhang, Wei Wu, and Weiran Xu. 2021. Consert: A contrastive framework for self-supervised sentence representation transfer. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 5065–5075.

Zichao Yang, Diyi Yang, Chris Dyer, Xiaodong He, Alex Smola, and Eduard Hovy. 2016. Hierarchical attention networks for document classification. In *Proceedings of the 2016 conference of the North American chapter of the association for computational linguistics: human language technologies*, pages 1480–1489.

Liang Yao, Chengsheng Mao, and Yuan Luo. 2019. Graph convolutional networks for text classification. In *Proceedings of the AAAI conference on artificial intelligence*, volume 33, pages 7370–7377.

Xu Zhang, Yifeng Li, Wenpeng Lu, Ping Jian, and Guoqiang Zhang. 2020. Intra-correlation encoding for chinese sentence intention matching. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 5193–5204.

Yue Zhang, Qi Liu, and Linfeng Song. 2018. Sentence-state lstm for text representation. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 317–327.

Zhengyan Zhang, Xu Han, Zhiyuan Liu, Xin Jiang, Maosong Sun, and Qun Liu. 2019. Ernie: Enhanced language representation with informative entities. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 1441–1451.

Adaptive Feature Discrimination and Denoising for Asymmetric Text Matching

Yan Li [1], **Chenliang Li**[2], **Junjun Guo**[1, *]
[1] Kunming University of Science and Technology
[2] Wuhan University
{liyayli,guojjgb}@163.com
cllee@whu.edu.cn

Abstract

Asymmetric text matching has becoming increasingly indispensable for many downstream tasks (*e.g.,* IR and NLP). Here, *asymmetry* means that the documents involved for matching hold different amounts of information, *e.g.,* a short query against a relatively longer document. The existing solutions mainly focus on modeling the feature interactions between asymmetric texts, but rarely go one step further to recognize discriminative features and perform feature denoising to enhance relevance learning. In this paper, we propose a novel **a**daptive feature **d**iscrimination and **d**enoising model for **a**symmetric te**x**t matching, called **ADDAX**. For each asymmetric text pair, ADDAX is devised to explicitly distinguish discriminative features and filter out irrelevant features in a context-aware fashion. Concretely, a matching-adapted gating siamese cell (MAGS) is firstly devised to identify discriminative features and produce the corresponding hybrid representations for a text pair. Afterwards, we introduce a locality-constrained hashing denoiser to perform feature-level denoising by learning a discriminative low-dimensional binary codes for redundantly longer text. Extensive experiments on four real-world datasets from different downstream tasks demonstrate that the proposed ADDAX obtains substantial performance gain over 36 up-to-date state-of-the-art alternatives.

1 Introduction

Given a pair of documents, text matching aims to precisely predict the semantic relations between them. An efficient and effective matching algorithm is now an indispensable asset in many information retrieval, question answering and dialogue systems. In these application scenarios, a text pair in matching (*e.g.,* query-document and question-answer pair) ususlly has a large disparity

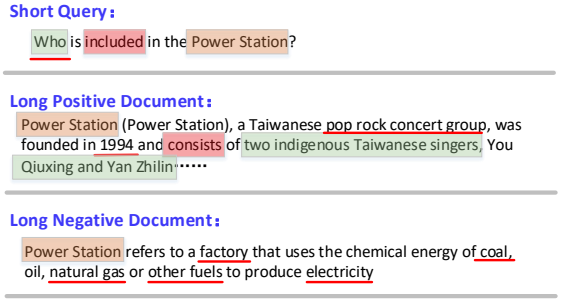

Figure 1: An example of asymmetric text matching.

in the quantity of information, *a.k.a.* asymmetric text matching. For example, a matching pair have 7.15 and 95.54 words respectively on average in InsuranceQA dataset (Feng et al., 2015) (*i.e.,* the difference being about an order of magnitude). This asymmetry between a short query and a long document renders it as a nontrivial task.

The existing solutions can be grouped into two categories, namely representation-based and interaction-based models (Khattab and Zaharia, 2020). The former category mainly utilizes convolutional neural networks (CNN) and recurrent neural networks (RNN) to learn the latent representation of a document independently, including DSSM (Huang et al., 2013), SNRM (Zamani et al., 2018). On contrast, the latter category focuses on leveraging fine-grained interaction signals between them. It is widely recognized that exploiting interaction signals would largely improve the relevance learning capacity. Examples include DRMM (Guo et al., 2016), KNRM (Xiong et al., 2017). Recently, with the prominence of deep pre-trained language models (LMs) like BERT (Devlin et al., 2019), uptodate LMs-based deep relevance models significantly push the frontier of the state-of-the-art further (Dai and Callan, 2019b; Nogueira and Cho, 2019; Xu and Li, 2020). Though significant performance gain is achieved by these efforts, they mainly overlook further feature discrimination and

*Corresponding author.

Proceedings of the 29th International Conference on Computational Linguistics, pages 1146–1156
October 12–17, 2022.

denoising between asymmetric texts, which can be potentially useful to enhance matching performance.

To explain this point, we give an illustrative example in Figure 1. Here, polysemous word like "*power station*" in the query side hinders the precise matching. On the other hand, the semantic association among "*who*", "*pop rock concert group*" and "*singers*" assists the relevance learning process. Also, word pair like "*who*" and "*factory*" describe two distinct things, which can be reflected via the feature-level interactions. Hence, recognizing the discriminative features and filtering out noisy features certainly enhance the relevance learning process.

To this end, in this paper, we propose an adaptive feature discrimination and denoising model for asymmetric text matching, named ADDAX. Specifically, ADDAX consists of a BERT-based context encoder, a matching-adapted gating siamese cell (called MAGS), a locality-constrained hashing denoiser, and a MaxSim (Khattab and Zaharia, 2020) based relevance predictor. For each document, we firstly derive the word-level contextual representations through a BERT-based context encoder. Afterwards, MAGS utilizes an cross-attention mechanism to represent a document with relevant information from its counterpart in the matching pair, which produces the word-level reference representations for the former. Then, the resultant word-level attention information is leveraged to discriminate the importance of these reference representations and their divergence against the orginal representations. We then utilize a highway network to adaptively composite these two kinds of semantic signals as the context-aware hybrid representations.

After this feature-level discrimination, we utilize a locality-constrained hashing denoiser to project the long document into low-dimensional binary space. The hashing denoiser is formulated as an autoencoder over the hybrid representations. That is, the semantics relevant to the text pair will be preserved by the denoising process, which further facilitates the relevance learning. Finally, a MaxSim operator (Khattab and Zaharia, 2020) is employed to calculate final relevance score. Overall, the key contributions are summarized as below:

- We propose an adaptive feature discrimination and denoising model for asymmetric text matching. To the best of our knowledge, ADDAX is the first attempt to explicitly derive discriminative features and perform feature denoising for this task.
- To derive discriminative features, a matching-adapted gating siamese cell (called MAGS) is devised to synthesize hybrid representations for a text pair in terms of word-level relevance information. To perform feature denoising, a locality-constrained hashing denoiser is devised to purify context-aware semantics and filter out feature-level noise for the long document.
- Extensive experiments are conducted on four real-world datasets and the results demonstrate the superior performance of our method compared against existing SOTA alternatives.

2 RELATED WORK

2.1 Traditional Neural Matching Models

Recent years, deep learning has delivered significant performance gain for various text matching tasks. Generally, deep relevance matching models can be divided into two categories. The first is the representation-based models that independently encodes a query and a document into two vectors and estimates relevance in terms of vector similarity (*e.g.,* DSSM (Huang et al., 2013), CDSSM (Shen et al., 2014), LSTM-RNN (Palangi et al., 2016)). For instance, Huang et al. (2013) propose to embed query and document into two vectors through a multilayer perceptron and calculates the corresponding cosine similarity as the matching score. Following this work, SNRM (Zamani et al., 2018) exploits the sparsity property to derive a sparse-vector representation for each query/document, which allows it to also leverage a dense index to do fast end-to-end retrieval. The second category, interaction-based models, exploits complex fine-grained interactions between two documents to magnify the relevance signals (Hofstätter et al., 2019; Hu et al., 2014; Pang et al., 2016; Hui et al., 2017). For example, DRMM (Guo et al., 2016) ranks documents based on the matching histogram of each query and document. Conv-KNRM (Dai et al., 2018) extends DRMM by pairwise n-gram similarity. FastText+ConvKNRM (Hofstätter et al., 2019) further makes use of subword-token embeddings to tackle the vocabulary mismatch problem. Wang et al. (2017) introduce a novel attention-based representation approach to leverage information aggregated

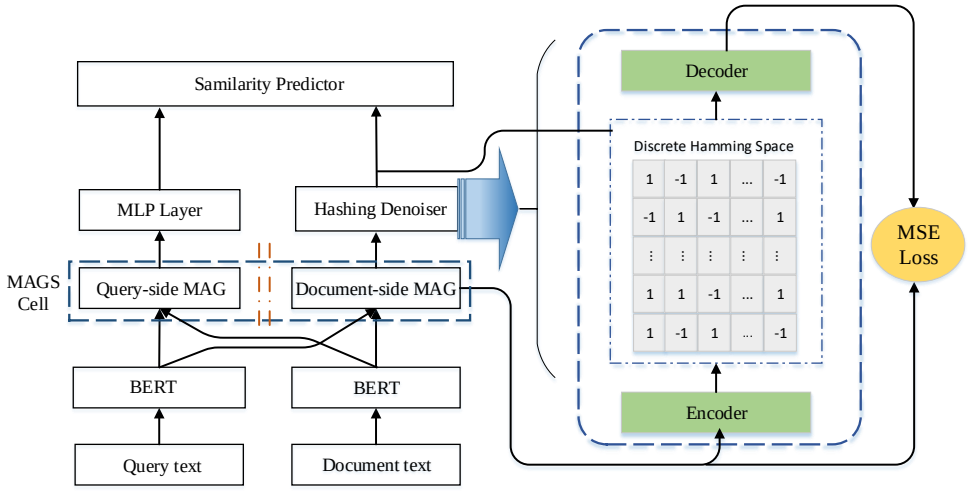

Figure 2: The whole network architecture of ADDAX.

from both question and passage for better predict answers.

2.2 Deep LM Based Matcher

The pre-training language representation models (PLMs), like BERT (Devlin et al., 2019), Roberta (Liu et al., 2019) and XLnet (Yang et al., 2019), have shown great capacity in encoding contextual information. It significantly outperforms other CNN-based and RNN-based neural models in many text matching tasks (Nogueira et al., 2019b,a; Karpukhin et al., 2020). DeepCT (Dai and Callan, 2019a) utilizes a deep contextualized term weighting framework for document retrieval, where contextualized text representations produced by PLMs are used to derive the context-aware term importance weights.

ColBERT (Khattab and Zaharia, 2020) performs interaction MaxSim operator on each PLMs-based word embedding in the query/document to calculate matching scores. Later, TCT-ColBERT (Lin et al., 2020) utilizes knowledge distillation to enhance query latency and greatly reduce the memory cost of ColBERT. CLEAR (Gao et al., 2021b) exploits a neural embedding matching model as a supplement towards conventional lexical matching. COIL (Gao et al., 2021a) utilizes vector similarities between query-document overlapping term contextualized representations for efficient search. Sun et al. (2021) combine the latent topic of the document with its PLMs-based representations to predict the relevance of documents given a query.

Comparing with these existing solutions, our proposed ADDAX goes a step further to identify discriminative features and perform feature denoising by considering the asymmetric nature of many text matching tasks.

2.3 Semantics-Preserving Hashing

Semantics-preserving hashing (Salakhutdinov and Hinton, 2009; Li et al., 2016) aims to learn a concise representation of the input by preserving the core semantics, which can be considered as a form of loss-free dimension reduction. Following this idea, VDSH (Chaidaroon and Fang, 2017) derives hashing codes with variational autoencoders (VAE) to reconstruct and preserve the semantics of the original text. NASH (Shen et al., 2018) learns a VAE-based generative model whose input are TF-IDF vectors, which treats binary codes as Bernoulli latent-variables. Here, we utilize an locality-constrainted hashing to filter out irrelevant information for the long document of a text pair, which can enable more precise text matching.

3 The proposed method

In this section, we first describe the task formulation in Section 3.1. Afterwards, we present each component of ADDAX in detail.

3.1 Task Formulation

Without loss of generality, we assume there are a short query Q and a long document D in an asymmetric text matching pair: $Q = \{q_1, \cdots, q_l\}$ and $D = \{d_1, \cdots, d_t\}$, where $l \ll t$. Here, q_i and d_j indicate the i-th and j-th token in the sequences respectively, and l and t are the number of tokens in the sequences respectively. The goal of the asymmetric matching $f(Q, D)$ is to predict whether Q and D hold a target relation r, where $r \in \{0, 1\}$.

3.2 Architecture

The network architecture of our proposed ADDAX is shown in Figure 2. The entire framework consists of four main parts: a BERT-based context encoder, a matching-adapted gating siamese cell, a locality-constrained hashing denoiser and a MaxSim (Khattab and Zaharia, 2020) based relevance predictor.

BERT-based Context Encoder. We choose to utilize BERT[1] as our context encoder. The BERT-based context encoder can be described as follows:

$$\mathbf{U}_Q = BERT([CLS]q_1q_2\cdots q_l) \quad (1)$$

$$\mathbf{V}_D = BERT([CLS]d_1d_2\cdots d_t) \quad (2)$$

where $\mathbf{U}_Q \in \mathbb{R}^{l\times d}$ and $\mathbf{V}_D \in \mathbb{R}^{t\times d}$ are word-level contextual representations derived for query Q and document D respectively. Parameter d denotes the output dimension of BERT, and `[CLS]` is a specific token indicating the beginning of the token sequence. To reduce the total number of parameters in ADDAX, mitigate overfitting, and facilitate feature interactions across the two texts, we share a single context encoder for both Q and D.

Matching-Adapted Gating Siamese Cell. A human being can identify the relation between two sequences (*e.g.,* query-document, keyword-document, and question-answer) at a glance. For instance, a well-trained graduate student can easily categorize the papers in his/her research direction in term of title and abstract, because he/she can subconsciously identify the discriminative features, and ignore the irrelevant features for the decision.

Here, we simulate this feature discrimination process with a matching-adapted gating siamese cell (called MAGS). It is a parallel architecture with two MAG cells, namely the query-side MAG and the document-side MAG (ref. Figure 3). Since the both query-side and document-side MAGs are identical (but with different parameters), we mainly describe the query-side MAG for simplicity.

Given $\mathbf{U}_Q = [\mathbf{u}_1;\cdots;\mathbf{u}_l]$ and $\mathbf{V}_D = [\mathbf{v}_1;\cdots;\mathbf{v}_t]$ derived by the context encoder, we aim to identify discriminative features and composite them as relevance features. At first, a cross-attention mechanism is utilized to calculate the word-level similarity as follows:

$$\mathbf{S} = \mathbf{U}_Q\mathbf{V}_D^\top \quad (3)$$

where $\mathbf{S} \in \mathbb{R}^{l\times t}$ is the similarity matrix for all the word pairs across the two texts. We then normalize these similarity scores and derive a reference representation in terms of $\mathbf{V}_D$ for each word in Q:

$$\mathbf{R}_Q = softmax(\mathbf{S})\mathbf{V}_D \quad (4)$$

where $softmax$ function is applied over each row of $\mathbf{S}$, and i-th row of $\mathbf{R}_Q$ is the reference representation for i-th word in Q. The purpose of this step is to perform soft feature selection from $\mathbf{V}_D$ according to $\mathbf{S}$. That is, the relevant information in D is *transferred* to represent Q.

However, during this reference representation process, irrelevant information in Q is also presented for further relevence learning. Hence, we construct supplementary features by considering the divergence of the reference representations against the original ones: $\mathbf{D}_Q = \mathbf{U}_Q - \mathbf{R}_Q$, which works as another form of semantic signals. Note that, the above cross-attention mechanism just blindly searches for the most similar token in D to reconstruct Q despite the fact that the most similar one in D might be an meaningless match. Thus, we choose to leverage the attention patterns expressed in $\mathbf{S}$ to both identify the importance of $\mathbf{D}_Q$ and $\mathbf{R}_Q$ as well as the importance of each individual feature in them:

$$\mathbf{E} = \sigma(\mathbf{S}\mathbf{W}_1 + \mathbf{B}_1) \quad (5)$$

$$\mathbf{F}^{(r)} = \mathbf{R}_Q \odot \mathbf{E} \quad (6)$$

$$\mathbf{F}^{(d)} = \mathbf{D}_Q \odot (\mathbf{1} - \mathbf{E}) \quad (7)$$

$$p_i = \sigma(\mathbf{S}_i\mathbf{w}_1 + b_1) \quad (8)$$

$$\mathbf{F}_i^{(c)} = p_i \cdot \mathbf{F}_i^{(r)} \oplus (1 - p_i) \cdot \mathbf{F}_i^{(d)} \quad (9)$$

where $\sigma(\cdot)$ denotes the *sigmoid* function, $\mathbf{W}_1, \mathbf{B}_1 \in \mathbb{R}^{t\times d}$, $\mathbf{w}_1 \in \mathbb{R}^{t\times 1}$ and $\mathbf{b}_1$ are learnable matrices and the bias, $\mathbf{S}_i$, $\mathbf{F}_i^{(c)}$, $\mathbf{F}_i^{(r)}$ and $\mathbf{F}_i^{(d)}$, subscript i indicates i-th row of the corresponding matrix respectively, $\odot$ and $\oplus$ are the element-wise product and vector concatenation operation respectively.

Afterwards, we adopt a highway network to generate the discriminative features $\mathbf{h}_i^Q$ for each word in Q:

$$\mathbf{p}_i = \text{relu}\left(\mathbf{W}_3\mathbf{F}_i^{(c)} + \mathbf{b}_3\right) \quad (10)$$

$$\mathbf{g}_i = \text{sigmoid}\left(\mathbf{W}_4\mathbf{F}_i^{(c)} + \mathbf{b}_4\right) \quad (11)$$

$$\mathbf{i}_i = (1 - \mathbf{g}_i) \odot \mathbf{F}_i^{(c)} + \mathbf{g}_i \odot \mathbf{p}_i \quad (12)$$

$$\mathbf{h}_i^Q = \mathbf{W}_5\mathbf{i}_i + \mathbf{b}_5 \quad (13)$$

[1] We used the base, uncased variant of BERT.

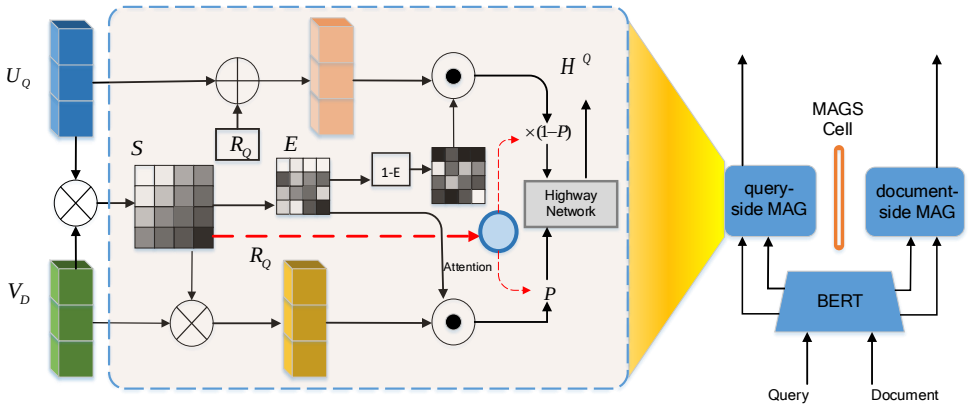

Figure 3: The network structure of MAGS.

where $\mathbf{W}_3, \mathbf{W}_4 \in \mathbb{R}^{2d\times 2d}$ and $\mathbf{W}_5 \in \mathbb{R}^{d\times 2d}$ are the transformation matrices, $\mathbf{b}_3$, $\mathbf{b}_4$ and $\mathbf{b}_5$ are bias vectors. We can form the resultant hybrid discriminative features as a matrix: $\mathbf{H}^Q = [\mathbf{h}_1^Q; \cdots; \mathbf{h}_l^Q]$.

Similarly, the document-side MAG cell switches the roles of Q and D for the same process, but with different parameters. The discriminative features are denoted as $\mathbf{H}^D = [\mathbf{h}_1^D; \cdots; \mathbf{h}_t^D]$.

Locality-Constrained Hashing Denoiser. Since document D is much larger than query Q, the discriminative feature extraction performed by the document-side MAG could still introduce many semantic noises. Here, we adopt a locality-constrained hashing denoiser to further filter out irrelevant features. More specifically, the locality-constrained hashing denoiser defines an encoding function $\mathcal{F}_{en}$, a hashing function $\mathcal{F}_h$, and a decoding function $\mathcal{F}_{de}$.

Encoder function $\mathcal{F}_{en}$ maps the representations $\mathbf{H}^D$ into a low-dimensional matrix $\mathbf{B} \in \mathbb{R}^{t\times h}$. Here, we model $\mathcal{F}_{en}$ as a feed forward network ($FNN(\cdot)$) implemented by a three-layer multi-layer perceptron (MLP), where the $ReLU(\cdot)$ is utilized as the activation function in the second layer to skip unnecessary features and retain discriminating clues (others are $tanh(\cdot)$). The encoding process can be summarized as: $\mathbf{B} = \mathcal{F}_{en}(\mathbf{H}^D) = FFN(\mathbf{H}^D)$.

Hashing function $\mathcal{F}_h$ is devised to learn discriminative binary features for purification and efficient matching. Generally, the $sgn(\cdot)$ function is the best choice for binarization, but not differentiable. Hence, we use an approximate function $tanh(\cdot)$ to replace $sgn(\cdot)$ for supporting model training. Specifically, the hashing function is written as: $\mathbf{B}^D = \mathcal{F}_h(\mathbf{B}) = \tanh(\alpha\mathbf{B})$. The hyper-parameters α is a coefficient helping generate balanced and discriminative hash codes. To ensure that the values in $\mathbf{B}^D \in \{-1, 1\}$, we define an extra constraint (*MSE* loss) (Li et al., 2016; Xu and Li, 2020): $\mathcal{L}_1 = \left\|\mathbf{B}^D - \boldsymbol{\mathcal{B}}^{(b)}\right\|_F^2$, where $\boldsymbol{\mathcal{B}}^{(b)} = sgn(\mathbf{B})$, and $\|\cdot\|_F$ is the Frobenius norm.

Similar to $\mathcal{F}_{en}$, decoding function $\mathcal{F}_{de}$ recovers $\mathbf{H}^D$ from $\mathbf{B}^D$ with a three-layer MLP (encoder transpose). Hence, the reconstructed matrix $\mathbf{H}_r^D \in \mathbb{R}^{t\times d}$ can be written as: $\mathbf{H}_r^D = \mathcal{F}_{de}(\mathbf{B}^D) = FFN^\top(\mathbf{B}^D)$. To preserve the core semantics during the reconstruction, a *MSE* loss is used to guide the model training: $\mathcal{L}_2 = \left\|\mathbf{H}_r^D - \mathbf{H}^D\right\|_F^2$.

We can also perform a hashing denoiser for $\mathbf{H}^Q$. However, we did not observe improvement due to noiseless nature of a short query. Instead, the matrix representations $\mathbf{H}^Q$ of query Q are updated with a single MLP layer to match the dimension of the hashing denoiser: $\mathbf{H}^Q = MLP(\mathbf{H}^Q)$, where $\mathbf{H}^Q \in R^{l\times h}$ is used for final prediction.

Similarity Predictor. With both $\mathbf{H}^Q = [\mathbf{h}_1^Q; \cdots; \mathbf{h}_l^Q]$ and $\mathbf{B}^D = [\mathbf{b}_1^D; \cdots; \mathbf{b}_t^D]$, the matching score between Q and D, $f(Q, D)$, is estimated via a MaxSim operator (Khattab and Zaharia, 2020) as follows:

$$f(Q, D) = \sum_i^l \max_j^t \cos(\mathbf{h}_i^Q \cdot \mathbf{b}_j^D) \qquad (14)$$

where function $cos(\cdot)$ calculates the cosine similarity of the given vectors.

3.3 Model Optimization

The objective of model optimization is to guide the relevance learning of ADDAX and help estimate the matching score of the asymmetric text pair. During the training stage, we uti-

lize the negative sampling strategy via a triplet-based hinge loss (Xu and Li, 2020): $\mathcal{L}_3 = \max\{0, 1.0 - f(Q, D) + f(Q, D^-)\}$, where D^- is the corresponding negative document sampled from the training set.

Finally, we need to combine the hinge loss and two constraints in hashing denoiser together. That is, the final optimization objective for ADDAX is a linear fusion of $\mathcal{L}_1$, $\mathcal{L}_2$ and $\mathcal{L}_3$:

$$\min_{\theta} \mathcal{L} = \sum_{(Q,D,D^-)} [\mathcal{L}_3 + \delta \cdot \mathcal{L}_1 + \gamma \cdot \mathcal{L}_2] \quad (15)$$

where δ and γ are tunable hyper-parameters controling the importance of each constraint respectively, θ is the parameter set of ADDAX. We use Adam (Kingma and Ba, 2014) for parameter update in an end-to-end fashion over mini-batches.

4 EXPERIMENTS

4.1 Experimental Settings

Datasets. Our experiments are conducted on four real-world datasets, covering the tasks of both question answer and document retrieval: *InsuranceQA* (Feng et al., 2015) is a widely used benchmark for QA. We leverage the v1.0 version of this corpus. *WikiQA* (Yang et al., 2015) is an open-domain answer selection dataset. We follow the preprocessing utilized in (Xu and Li, 2020) to filter out the questions that have no positive answers; *YahooQA*[2] is a collection constructed from Yahoo! Answers. In order to ensure that it has sufficient asymmetric text pairs, sentences with length among the range of 16 - 24 are filtered; *MS MARCO*[3] is a benchmark for information retrieval. It is a collection of 8.8M passages from web pages and contains approximately 400M tuples of a query, positive and negative passages. We utilize the provided data partition for model training and evaluation.

The average length ratio for a query and a document is greater than 3 for these datasets, which conform to asymmetric text matching scenarios.

Baselines. We compare our proposed ADDAX with two types of state-of-the-art baselines. The first type can perform question-answer (QA) matching. The chosen baseline models for answer selection can be partitioned into four categories: **(a)** *conventional single models*: IARNN-GATE (2016), AP-CNN (2016), RNN-POA (2017), AP-BiLSTM (2016), HD-LSTM (2017), AP-LSTM (2018), Multihop-Sequential-LSTM (2018), HyperQA (2018), MULT (2016), TFM+HN (2019), LSTM-CNN+HN (2019); **(b)** *single models that exploit external knowledge*: KAN (2018), CK-ANN (2021), CKANN-L (2021); **(c)** *ensemble models*: SUM$_{BASE,PTK}$ (2018), LRXNET (2018), SD (BiLSIM+TFM) (2020); **(d)** *BERT-based models*: HAS (2020), DDR-Match(BERT,WD) (2022) and BERT$_{base}$ is implemented by ourselves.

As to document retrieval, we include the following methods for comparison: BM25 (2018), PACRR (2017), KNRM (2017) and fastText+ConvKNRM (2019). In addition, since the proposed ADDAX adopts the BERT as the context encoder, we then pick several uptodate LMs-based models, including BERT$_{base}$ ranker (2019), DeepCT (2019a), docT5query (2019a), ColBERT (2020), TCT-ColBERT (2020), COIL-tok (2021a) and COIL-full (2021a). Furthermore, we also include two dense retrievers for performance comparison, *i.e.,* RepCONC (2022), CLEAR (2021b) and ADORE+STAR (2021).

Parameter Settings and Evaluation Metrics. In our experiments, we choose $BERT_{base}$ as the context encoder in ADDAX. To be more specific, we set the hidden dimension h to be 300. The mini-batch size for insuranceQA, wikiQA, yahooQA, and MS MARCO is set to be 32, 32, 64, and 64, respectively. The probability of dropout is set to be 0.1. The learning rate for insuranceQA, MS MARCO, wikiQA and yahooQA is $5e^{-6}$, $5e^{-6}$, $1e^{-5}$, and $9e^{-6}$, respectively. The numbers of training epoches are 60 for insuranceQA, 18 for wikiQA and 9 for yahooQA. In addition, we train ADDAX for 200k iterations for MS MARCO. The values of α, δ and γ are set to 5, $1e^{-6}$, 0.003 respectively. To enable fair comparison, we choose the common evaluaton metrics utilized in these baselines and directly reuse the reported results from the corresponding papers.

4.2 Performance Comparison

Table 1 summarizes the performance of 22 methods for answer selection on the corresponding three datasets. We choose to discuss the experimental results on each dataset separately.

On InsuranceQA, We can observe that conventional single models like IARNN-GATE, MULT, and TFM+HN perform much better than other sin-

[2]https://webscope.sandbox.yahoo.com/catalog.php?datatype=l&guccounter=1

[3]https://microsoft.github.io/msmarco/

Table 1: Performance comparison on the QA datasets (best in boldface). Results not applicable and not available are denoted "–" and "n.a." respectively. HAS-HL represents a model variant of HAS without a hashing layer. Significant improvement with respect to HAS is indicated (†) (p-value ≤ 0.05).

	insuranceQA		wikiQA		yahooQA	
Model	P@1(Test1)	P@1(Test2)	MAP	MRR	P@1	MRR
IARNN-GATE	70.10	62.80	72.58	73.94	–	–
AP-CNN	69.80	66.30	68.86	69.57	56.00	72.60
AP-BiLSTM	71.70	66.40	67.05	68.42	56.80	73.10
HD-LSTM	–	–	–	–	55.70	73.50
HyperQA	n.a.	n.a.	71.20	72.70	68.30	80.10
RNN-POA	n.a.	n.a.	72.12	73.12	n.a.	n.a.
Multihop-Sequential-LSTM	70.50	66.90	72.20	73.80	n.a.	n.a.
AP-LSTM	69.00	64.80	68.90	69.60	n.a.	n.a
MULT	75.20	73.40	74.33	75.45	n.a.	n.a.
LSTM-CNN+HN	73.30	69.10	–	–	–	–
TFM+HN	75.60	73.40	–	–	–	–
KAN (Tgt-Only)	71.50	68.80	–	–	67.20	80.30
KAN	75.20	72.50	–	–	74.40	84.00
CKANN	76.30	**75.10**	73.20	75.50	84.40	90.20
CKANN-L	75.90	74.90	72.80	73.90	84.20	90.60
$\text{SUM}_{BASE,PTK}$	–	–	75.59	77.00	–	–
LRXNET	–	–	76.57	75.10	–	–
SD (BiLSIM+TFM)	–	–	70.40	71.20	–	–
BERT_{base}	74.52	71.97	75.30	77.00	73.49	81.93
DDR-Match(BERT,WD)	n.a.	n.a.	79.58	81.23	n.a.	n.a.
HAS	76.38	73.71	81.01	82.22	73.89	82.10
HAS-HL	76.12	74.12	80.65	81.83	74.78	82.68
ADDAX	**77.83**†	74.83†	**82.50**†	**83.38**†	**87.63**	**90.69**

gle models. Also, it is not surprising that the BERT-based methods (*e.g.,* HAS) consistently yield the better performance compared to single models. This is expected since the LMs can absorb large-scale common knowledge to help bridge the vocabulary mismatch. These observations are consistent with many previous works (Xu and Li, 2020). Single models that exploit external knowledge (*e.g.,* KAN and CKANN) are superior to those convetional single models and BERT-based models, mainly because the external knowledge is very helpful. As a comparison, our ADDAX achieves significantly better performance than almost all baselines in InsuranceQA dataset (except for CKANN on Test2 set).

On WikiQA, it is surprising that single models exploiting external knowledge can not obtain obvious advantages compared to some single models. The possible reasons for this phenomenon could be the scarity of the training data and irrelevant external knowledge. Secondly, ensemble models like $\text{SUM}_{BASE,PTK}$ and LRXNET significantly outperform SD (BiLSIM+TFM). Also, the ensemble models obtain substantial performance gain than the both conventional single models and the ones with external knowledge, indicating effectiveness of integrating multiple models in improving the generalization ability. Thirdly, BERT_{base} consistently performs worse than HAS-HL and HAS. This observation is consistent across all the four datasets, suggesting positive benefit of model feature interactions. As to ADDAX, a much better performance is obtained against all baselines here.

On YahooQA, we observe a similar performance pattern as with the InsuranceQA dataset. Our proposed ADDAX substantially outperforms all baselines in terms of P@1 and MRR. Specifically, compared with the best baseline, our ADDAX obtains relative $P@1$ gain of 3.23%.

Table 2 reports the performance comparison of different document retrieval models on MS MARCO. For the neural matching models, LMs-based methods obtain much better performance than PACRR, KNRM and fastText+ConvKNRM, suggesting the powerful language expression ability of the former. Note that DeepCT and DocT5Query can adaptively adjust the term importance by exploiting LMs, but they are still inferior in semantic matching. Also, it is worth noting that dense retrievers are almost on par with the LMs-based models. In contrast, ADDAX consistently achieves the best performance on MS MARCO

Table 2: Results on MS MARCO (best in boldface).

Model	MS MARCO MRR@10(dev)
BM25	18.70
KNRM	19.80
PACRR	25.90
fastText+ConvKNRM	29.00
BERT_{base}	34.70
DeepCT	24.30
docT5query	27.70
ColBERT	34.90
TCT-ColBERT	33.50
COIL-tok	33.60
COIL-full	34.80
CLEAR	33.80
RepCONC	34.00
ADORE+STAR	34.70
ADDAX	**36.15**

dataset. Specifically, the performance gain by ADDAX over all the baselines is in the range of 1.25%-17.40% in terms of MRR@10.

Overall, the above comparisons made over two different tasks consistently show that the proposed ADDAX achieves substantial performance gain in general. These promising results validate that the matching-adapted gating siamese cell and the hashing denoiser proposed in ADDAX are effective in performing feature discrimination and denoising for asymmetric text matching.

4.3 Model Analysis

Ablation Study. Here, we perform a series of ablation studies to explore how each design in ADDAX affects the asymmetric text matching. To be more specific, we compare ADDAX with the following variants: (a) **w/o MAGS**, removing the matching-adapted gating siamese cell; (b) **w/o HW**, eliminating highway network to fuse the two kinds of semantic signals, but add them directly; (c) **w/o HD**, excluding the locality-constrained hashing denoiser; (d) **Att-MAGS**, in the case of w/o HD, keeping only cross-attention mechanism in MAGS.

Table 3 reports the results on MS MARCO and wikiQA datasets. We can see that the exclusion of the matching-adapted gating siamese cell incurs the largest performance degradation, followed by the hashing denoiser. Particularly, w/o MAGS drops absolutely by 5.30% and 4.61% on wikiQA in terms of MAP and MRR, respectively, and 1.25% on MS MARCO in terms of MRR@10. In addition, Att-MAGS is also worse than MAGS. These demonstrates that the matching-adapted gating siamese cell is effective in identifing discriminative features to enhance matching accuracy. Besides, w/o HD also results in worse performance, suggesting the effectiveness of performing feature-level denoising on document side. For example, we illustrate the feature heatmap of a random sample $\mathbf{B}^D$ generated by ADDAX and w/o HD (*i.e.,* as shown in Figure 4(d)). We can see that the denoiser indeed filters many features. For each specific structure designed in MAGS, we also make the following observation: the performance drop of w/o HW suggests that the highway network is more effective to composite the hybrid discriminative features.

Table 3: The performance of different ADDAX variants on wikiQA and MS MARCO (best in boldface).

Model	MS MARCO MRR@10(dev)	wikiQA MAP	wikiQA MRR
w/o MAGS	34.90	77.20	78.77
w/o HW	35.32	79.01	80.23
w/o HD	35.49	80.15	81.58
Att-MAGS	35.00	79.79	81.34
ADDAX	**36.15**	**82.50**	**83.38**

In general, our proposed ADDAX consistently surpasses five variants on MS MARCO and wikiQA datasets, demonstrating the validity of each component design.

Sensitivity Analysis of α, δ and γ. We further investigate the sensitivity of hyper-parameters (*i.e.,* α, δ and γ) in ADDAX on the wikiQA test set. Recall that δ controls the importance of the constraint loss $\mathcal{L}_1$, γ controls the importance of hashing denoiser's reconstruction loss $\mathcal{L}_2$, and α control the balance of the hash codes. When studying a parameter, the other two parameters are fixed to the values described in Section 4.1.

From Figure 4(c), we can see that the matching performance starts growing by increasing α to 5. Moreover, Figure 4(b) plots the performance pattern by varying δ values. We observe that ADDAX is not sensitive to δ in the range of $[1e^{-6}, 5e^{-6}]$ and obtains better performance at $\delta = 1e^{-6}$. Figure 4(a) plots the performance pattern by varying γ values. When γ is greater than or less than 0.003, the performance becomes much worse.

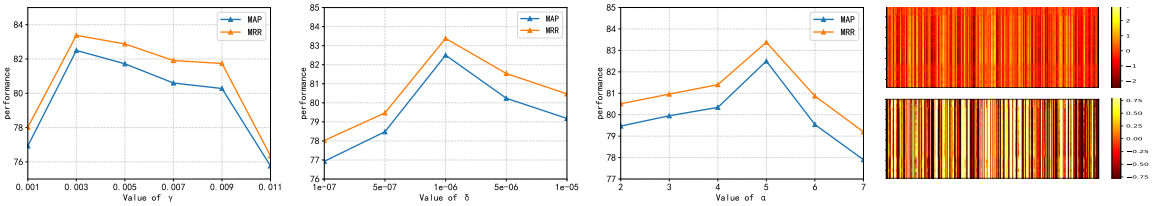

(a) Sensitivity of γ. (b) Sensitivity of δ. (c) Sensitivity of α. (d) Feature heatmap.

Figure 4: Performance with varying parameters on wikiQA dataset. The upper and lower subgraphs in Figure (d) represent feature heatmap of w/o HD and ADDAX on $\mathbf{B}^D$ respectively.

5 CONCLUSION

In this paper, we introduce an adaptive feature discrimination and denoising model for asymmetric text matching. Specifically, we first design a matching-adapted gating siamese cell in ADDAX to perform feature discrimination and generate the hybrid representations together for the asymmetric text pair. We then present a locality-constrained hashing denoiser for filtering semantic noise for redundant long documents. Extensive experimental results on four benchmarks have demonstrated the effectiveness and superiority of our proposed ADDAX. As future work, we plan to investigate the possibility of feature discrimination and denoising in other asymmetric scenarios like document abstractive summarization, caption generation and more.

Acknowledgements

This work was supported by National Key Research and Development Program of China (No. 2020AAA0107904). National Natural Science Foundation of China (No. U21B2027, 61866020), Key Research and Development Program of Yunnan Province (No. 202103AA080015). Natural Science Foundation Project of Yunnan Science and Technology Department (No. 2019FB082).

References

Suthee Chaidaroon and Yi Fang. 2017. Variational deep semantic hashing for text documents. In *Proceedings of the 40th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 75–84.

Dong Chen, Shaoliang Peng, Kenli Li, Ying Xu, Jinling Zhang, and Xiaolan Xie. 2020. *Re-Ranking Answer Selection with Similarity Aggregation*, page 1677–1680. Association for Computing Machinery, New York, NY, USA.

Qin Chen, Qinmin Hu, Jimmy Xiangji Huang, Liang He, and Weijie An. 2017. Enhancing recurrent neural networks with positional attention for question answering. In *Proceedings of the 40th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 993–996.

Zhuyun Dai and Jamie Callan. 2019a. Context-aware sentence/passage term importance estimation for first stage retrieval. *arXiv preprint arXiv:1910.10687*.

Zhuyun Dai and Jamie Callan. 2019b. Deeper text understanding for IR with contextual neural language modeling. In *SIGIR 2019 - Proceedings of the 42nd International ACM SIGIR Conference on Research and Development in Information Retrieval*.

Zhuyun Dai, Chenyan Xiong, Jamie Callan, and Zhiyuan Liu. 2018. Convolutional neural networks for soft-matching n-grams in ad-hoc search. In *Proceedings of the eleventh ACM international conference on web search and data mining*, pages 126–134.

Yang Deng, Ying Shen, Min Yang, Yaliang Li, Nan Du, Wei Fan, and Kai Lei. 2018. Knowledge as a bridge: Improving cross-domain answer selection with external knowledge. In *Proceedings of the 27th international conference on computational linguistics*, pages 3295–3305.

Yang Deng, Yuexiang Xie, Yaliang Li, Min Yang, Wai Lam, and Ying Shen. 2021. Contextualized knowledge-aware attentive neural network: Enhancing answer selection with knowledge. *ACM Transactions on Information Systems*.

Jacob Devlin, Ming Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *NAACL HLT 2019 - 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies - Proceedings of the Conference*.

Minwei Feng, Bing Xiang, Michael R Glass, Lidan Wang, and Bowen Zhou. 2015. Applying deep learning to answer selection: A study and an open task. In *2015 IEEE Workshop on Automatic Speech Recognition and Understanding (ASRU)*, pages 813–820. IEEE.

Luyu Gao, Zhuyun Dai, and Jamie Callan. 2021a. Coil: Revisit exact lexical match in information retrieval with contextualized inverted list.

Luyu Gao, Zhuyun Dai, Tongfei Chen, Zhen Fan, Benjamin Van Durme, and Jamie Callan. 2021b. Complement lexical retrieval model with semantic residual embeddings. In *Advances in Information Retrieval–43rd European Conference on IR Research*.

Jiafeng Guo, Yixing Fan, Qingyao Ai, and W. Bruce Croft. 2016. A deep relevance matching model for Ad-hoc retrieval. In *International Conference on Information and Knowledge Management, Proceedings*.

Sebastian Hofstätter, Navid Rekabsaz, Carsten Eickhoff, and Allan Hanbury. 2019. On the effect of low-frequency terms on neural-IR models. In *SIGIR 2019 - Proceedings of the 42nd International ACM SIGIR Conference on Research and Development in Information Retrieval*.

Baotian Hu, Zhengdong Lu, Hang Li, and Qingcai Chen. 2014. Convolutional neural network architectures for matching natural language sentences. In *Advances in Neural Information Processing Systems*.

Po-Sen Huang, Xiaodong He, Jianfeng Gao, Li Deng, Alex Acero, and Larry Heck. 2013. Learning deep structured semantic models for web search using clickthrough data. In *Proceedings of the 22nd ACM international conference on Information & Knowledge Management*, pages 2333–2338.

Kai Hui, Andrew Yates, Klaus Berberich, and Gerard De Melo. 2017. Pacrr: A position-aware neural ir model for relevance matching. *arXiv preprint arXiv:1704.03940*.

Vladimir Karpukhin, Barlas Oğuz, Sewon Min, Patrick Lewis, Ledell Wu, Sergey Edunov, Danqi Chen, and Wen-tau Yih. 2020. Dense passage retrieval for open-domain question answering. *arXiv preprint arXiv:2004.04906*.

Omar Khattab and Matei Zaharia. 2020. ColBERT: Efficient and Effective Passage Search via Contextualized Late Interaction over BERT. In *SIGIR 2020 - Proceedings of the 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval*.

Diederik P Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*.

Sawan Kumar, Shweta Garg, Kartik Mehta, and Nikhil Rasiwasia. 2019. Improving answer selection and answer triggering using hard negatives. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 5911–5917, Hong Kong, China. Association for Computational Linguistics.

Wu Jun Li, Sheng Wang, and Wang Cheng Kang. 2016. Feature learning based deep supervised hashing with pairwise labels. In *IJCAI International Joint Conference on Artificial Intelligence*.

S. C. Lin, J. H. Yang, and J. Lin. 2020. Distilling dense representations for ranking using tightly-coupled teachers.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Roberta: A robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692*.

Shashi Narayan, Ronald Cardenas, Nikos Papasarantopoulos, Shay B Cohen, Mirella Lapata, Jiangsheng Yu, and Yi Chang. 2018. Document modeling with external attention for sentence extraction. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 2020–2030.

Rodrigo Nogueira and Kyunghyun Cho. 2019. Passage re-ranking with bert.

Rodrigo Nogueira, Jimmy Lin, and AI Epistemic. 2019a. From doc2query to doctttttquery. *Online preprint*.

Rodrigo Nogueira, Wei Yang, Kyunghyun Cho, and Jimmy Lin. 2019b. Multi-stage document ranking with BERT.

Hamid Palangi, Li Deng, Yelong Shen, Jianfeng Gao, Xiaodong He, Jianshu Chen, Xinying Song, and Rabab Ward. 2016. Deep sentence embedding using long short-term memory networks: Analysis and application to information retrieval. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 24(4):694–707.

Liang Pang, Yanyan Lan, Jiafeng Guo, Jun Xu, Shengxian Wan, and Xueqi Cheng. 2016. Text matching as image recognition. In *30th AAAI Conference on Artificial Intelligence, AAAI 2016*.

Ruslan Salakhutdinov and Geoffrey Hinton. 2009. Semantic hashing. *International Journal of Approximate Reasoning*, 50(7):969–978.

Cicero dos Santos, Ming Tan, Bing Xiang, and Bowen Zhou. 2016. Attentive pooling networks. *arXiv preprint arXiv:1602.03609*.

Dinghan Shen, Qinliang Su, Paidamoyo Chapfuwa, Wenlin Wang, Guoyin Wang, Lawrence Carin, and Ricardo Henao. 2018. Nash: Toward end-to-end neural architecture for generative semantic hashing. *arXiv preprint arXiv:1805.05361*.

Yelong Shen, Xiaodong He, Jianfeng Gao, Li Deng, and Grégoire Mesnil. 2014. A latent semantic model with convolutional-pooling structure for information retrieval. In *Proceedings of the 23rd ACM international conference on conference on information and knowledge management*, pages 101–110.

Xingwu Sun, Yanling Cui, Hongyin Tang, Qiuyu Zhu, Fuzheng Zhang, and Beihong Jin. 2021. Tita: A two-stage interaction and topic-aware text matching model. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 5431–5440.

Yi Tay, Minh C Phan, Luu Anh Tuan, and Siu Cheung Hui. 2017. Learning to rank question answer pairs with holographic dual lstm architecture. In *Proceedings of the 40th international ACM SIGIR conference on research and development in information retrieval*, pages 695–704.

Yi Tay, Luu Anh Tuan, and Siu Cheung Hui. 2018. Cross temporal recurrent networks for ranking question answer pairs. In *Thirty-second AAAI conference on artificial intelligence*.

Nam Khanh Tran and Claudia Niedereée. 2018. Multihop attention networks for question answer matching. In *The 41st International ACM SIGIR Conference on Research & Development in Information Retrieval*, pages 325–334.

Kateryna Tymoshenko and Alessandro Moschitti. 2018. Cross-pair text representations for answer sentence selection. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 2162–2173.

Bingning Wang, Kang Liu, and Jun Zhao. 2016. Inner attention based recurrent neural networks for answer selection. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1288–1297.

Shuohang Wang and Jing Jiang. 2016. A compare-aggregate model for matching text sequences. *arXiv preprint arXiv:1611.01747*.

Wenhui Wang, Nan Yang, Furu Wei, Baobao Chang, and Ming Zhou. 2017. Gated self-matching networks for reading comprehension and question answering. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 189–198.

Chenyan Xiong, Zhuyun Dai, Jamie Callan, Zhiyuan Liu, and Russell Power. 2017. End-To-end neural ad-hoc ranking with kernel pooling. In *SIGIR 2017 - Proceedings of the 40th International ACM SIGIR Conference on Research and Development in Information Retrieval*.

Dong Xu and Wu-Jun Li. 2020. Hashing based answer selection. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pages 9330–9337.

Peilin Yang, Hui Fang, and Jimmy Lin. 2018. Anserini: Reproducible ranking baselines using lucene. *Journal of Data and Information Quality (JDIQ)*, 10(4):1–20.

Yi Yang, Wen-tau Yih, and Christopher Meek. 2015. Wikiqa: A challenge dataset for open-domain question answering. In *Proceedings of the 2015 conference on empirical methods in natural language processing*, pages 2013–2018.

Zhilin Yang, Zihang Dai, Yiming Yang, Jaime Carbonell, Russ R Salakhutdinov, and Quoc V Le. 2019. Xlnet: Generalized autoregressive pretraining for language understanding. *Advances in neural information processing systems*, 32.

Weijie Yu, Chen Xu, Jun Xu, Liang Pang, and Ji-Rong Wen. 2022. Distribution distance regularized sequence representation for text matching in asymmetrical domains. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 30:721–733.

Hamed Zamani, Mostafa Dehghani, W. Bruce Croft, Erik Learned-Miller, and Jaap Kamps. 2018. From neural re-ranking to neural ranking: Learning a sparse representation for inverted indexing. In *International Conference on Information and Knowledge Management, Proceedings*.

Jingtao Zhan, Jiaxin Mao, Yiqun Liu, Jiafeng Guo, Min Zhang, and Shaoping Ma. 2021. Optimizing dense retrieval model training with hard negatives. *arXiv preprint arXiv:2104.08051*.

Jingtao Zhan, Jiaxin Mao, Yiqun Liu, Jiafeng Guo, Min Zhang, and Shaoping Ma. 2022. Learning discrete representations via constrained clustering for effective and efficient dense retrieval. In *Proceedings of the Fifteenth ACM International Conference on Web Search and Data Mining*, pages 1328–1336.

Rethinking Data Augmentation in Text-to-text Paradigm

Yanan Chen
Dept. of Physics and Computer Science
Wilfrid Laurier University
chen0040@mylaurier.ca

Yang Liu
Dept. of Physics and Computer Science
Wilfrid Laurier University
yangliu@wlu.ca

Abstract

As manually labelling data can be costly, some recent studies tend to augment the training data for improving the generalization power of machine learning models, known as *data augmentation* (DA). With the arise of pre-trained language models (PLMs), some recent works on DA try to synthesize new samples benefiting from the knowledge learned from PLM's pre-training. Along the same direction, we in this paper propose to integrate text-to-text language models and construct a new two-phase framework for augmentation: 1) a *fine-tuning* phase where PLMs are well adapted to downstream classification with the help of two novel schemes, and 2) a *generation* phase where the fine-tuned models are leveraged to create new samples for performance lifting. This paradigm opens up a new way of designing fine-tuning scheme to better serve DA in an easy-to-implement manner, and can be easily extended to other desired tasks. We evaluate our proposal on two public classification datasets and demonstrate its effectiveness with remarkable gains.

1 Introduction

Due to the unique challenges of natural language processing tasks, there is no one-size-fits-all DA solution. Most early attempts are based on token manipulation (Wei and Zou, 2019; Kobayashi, 2018; Wu et al., 2019) or paraphrasing (Sennrich et al., 2016), and the boost is limited or marginal, or even diminishing or adverse especially given original training corpus is relatively sufficient or the backbone classifiers are PLM based, such as BERT or RoBERTa (Longpre et al., 2020).

Some researchers have shifted attention on applying generative language models(GLMs) for DA (Weng, 2022). Auto-regressive generation LMs such as GPT2 (Radford et al., 2019) are capable of generating text with high fluency and diversity, and therefore could serve as generators to synthesize new samples required by classification model training. However, most existing GLM-based DA solutions have some drawbacks. First, they fine-tune GLMs on the training corpus of limited capacity (Kumar et al., 2020; Anaby-Tavor et al., 2020; Liu et al., 2020), which can be problematic and prone to overfitting (Dodge et al., 2020; Phang et al., 2018; Ruder, 2021). Second, how to introduce external colossal online corpus freely available, such as reviews, comments and news to benefit GLMs to better serve DA has not been studied. Third, effective fine-tuning regime customized for data characteristic and structure has rarely been studied. In addition, recent works employing few-shot in-context generation for DA, such as GPT3, in avoidance fine-tuning and reap sparks of cleverness for automation, suffers from economic costs, latency in usage and lack of reliability (Sahu et al., 2022; Yoo et al., 2021; Wang et al., 2022).

To meet above challenges, we explore the potential of using text-to-text (seq2seq) language models, which have proved their success in many NLP tasks such as dialogue generation and machine translations. In the context of data augmentation, the original training samples can be regarded as the source text which sheds some light on the semantic meaning of the topic, whereas new synthetic samples will be considered as the target text induced by the source. Without loss of generality, we investigate the generation power of two exemplar text-to-text LMs: T5 and BART. Further, to cater for the text-to-text framework, we propose two fine-tuning schemes called *P2P* and *S2S* which organize the original corpus into parallel source and target text pairs, Different from many studies compromised by limited labelled data, large publicly available online corpus is adopted in the fine-tuning process. The proposed solution is evaluated on two text categorization tasks. Extensive experimental results prove these schemes can unleash the prowess of text-to-text generation while improving PLMs' generalization ability for DA.

Proceedings of the 29th International Conference on Computational Linguistics, pages 1157–1162
October 12–17, 2022.

2 Methodology

2.1 Problem formulation

Assume training data D_{train} contain a set of tuples $\{X^i, Y^i\}_{i=1}^{N}$ corresponding to word sequences and labels respectively. Our objective is to use text-to-text LMs denoted as $\mathcal{G}$ to produce synthetic training data $D_{syn} : \{\tilde{X}^i, Y^i\}_{i=1}^{N'}$, where $\tilde{X^i} = \mathcal{G}\left(X^i\right)$ and $N' = fN$ as generation can be repeated f times[1]. The augmented samples are expected to maintain both affinity and diversity. D_{train} together with D_{syn} are used to improve the classifier's robustness and performance (Gontijo-Lopes et al., 2020).

2.2 Text-To-Text model selection

We adopt T5 and BART model(base version) for text-to-text generation, for sake of their relatively lower computational costs and being used as benchmarks in previous studies. Note that, however, they can be easily replaced by any other text-to-text LMs, such as MASS (Song et al., 2019).

2.3 On-demand fine-tuning

To adapt to a downstream task, the most common approach is fine-tuning, in which PLM's weights are slightly updated based on a specific dataset $\mathcal{D}_{task}$. For text-to-text models such as T5/BART, $\mathcal{D}_{task} = \{T_x^i, T_y^i\}_{i=1}^{L}$ consists of parallel text pairs. Fine-tuning requires extra update steps and large L to optimize the parameters θ_e(encoder) and θ_d(decoder) with the objective of minimizing the loss of expression 1.

$$\mathcal{L}_{\text{Pair}} = \sum_{(T_x, T_y) \in \mathcal{D}_{\text{task}}} -\log p\left(T_y \mid T_x; \theta_e, \theta_d\right) \quad (1)$$

In this work, we present two new fine-tuning schemes tailored for text-to-text DA.

1. **Paragraph To Paragraph (P2P)** We observe that sentences in the same article tend to demonstrate strong internal consistency and coherence, so sentences in the front can serve as a prologue that summarizes or induces the remaining part. Motivated by this observation, we present a scheme which cuts an article of M sentences in half: taking the first $M/2$ as the source text T_x and the remaining as the target T_y. To gather enough knowledge of the context, any articles with $M < 4$ will be pruned out.

2. **Shard To Shard (S2S)** We also notice that adjacent sentences/paragraphs in the same article tend to deliver similar meanings and therefore are semantically related. To reflect this idea, we next present another scheme which first shuffles the sentences, and then randomly sample $M/2$ sentences as T_x and the remainder as T_y. It is expected that this scheme could replenish related contents based on the fragments of the original text.

We should note that we do not cherry pick samples and remove noises or redundancy to minimize the human intervention.

2.4 New sample generation

We take each X^i from D_{train} as prompt and pass it into T5/BART for generation of the augmentation sample [2].

3 Experiments

3.1 Datasets

Related free corpus for fine-tuning We proposed to fine-tune models on some open corpus freely available. Given the domain similarity and transferability, we use the *realnewslike* split of **C4**(Raffel et al., 2019) which is extracted from news websites, to fine-tune model for downstream topic classification task. For sentiment classification task, we employ the union of ***Amazon Review***, ***Yelp Restaurant Review*** (Zhang et al., 2015) and ***IMDB Movie Review*** (Maas et al., 2011).

Experimental dataset for DA. To justify the effectiveness of our proposal, we carefully design a series of experiments and evaluate it on topic classification datasets: **AG News** (Zhang et al., 2015) and sentiment classification **SST-2**(Stanford Sentiment Treebank) (Socher et al., 2013). Both datasets are class balanced. Details can be seen in Table 3 in Appendix.

3.2 Baseline DA methods

To make a comprehensive comparison, we include most popular baseline methods: 1) **EDA** (Wei and Zou, 2019) and **CBERT**(Wu et al., 2019) both of which are based on token manipulation, 2) **Back-Translation(BT)** (Sennrich et al., 2016) based on paraphrasing, and 3) **LAMBDA** (Anaby-Tavor et al., 2020) based on generation. Among them,

[1] We keep f to 1 throughout this work for simplicity

[2] Occasionally the sample may need to be truncated to meet models' input limitation requirement, however, this seldom happens in AG and SST-2 dataset

CBERT and **LAMBDA** are both label-conditional and need to be fine-tuned on D_{train} following their own schemes.

3.3 Backbone classifiers

To evaluate the gain of introducing new samples, two widely adopted classifiers are employed: one is a light-weighted transformer and the other is the bulky and resource-hungry BERT (Devlin et al., 2019). In each trial, with a random seed, we select K samples from each class to construct a balanced dataset D_{train} and apply different DA methods to derive synthetic datasets D_{syn} respectively. Next, a classifier $\mathcal{C}'$ is trained on $D_{train} \cup D_{syn}$ and $\mathcal{C}$ is trained only on D_{train}. Finally, both $\mathcal{C}'$ and $\mathcal{C}$ are evaluated on D_{test}. This trial is repeated 100 times with different random seed to report the averaged accuracy overall to get a reliable finding.

3.4 Main results and Analysis

Comparison with baselines. Our proposed method is compared with alternatives introduced in Section 3.2. The average accuracy is reported in Table 1 [3].

It is clear that our method demonstrates the superiority over all the benchmarks, especially in low-data regime. In DA for AG News topic classification task, fine-tuning T5 or BART on C4 consistently outmatches the baselines, while T5 fine-tuned on S2S paradigm yields the best results. In DA for SST-2 sentiment classification task, fine-tuning BART on reviews corpus under S2S scheme also shows obvious gains. When the training corpus is larger , the gain from DA becomes marginal. Note that LAMBDA is also a GLM-based DA method; however its performance is not up to par. Similar observations have been reported in some recent studies (Wang et al., 2022; Sahu et al., 2022), which suggests directly fine-tuning PLMs with small training data may lead to overfitting as they simply attempt to memorize what they see.

Ablation study. In this part, we demonstrate the necessity of appropriate fine-tuning scheme. Also, as GPT2 is widely used in previous generation based DA and it also shares some commonness in terms of the transformer architecture, here we aim to compare between Text2Text model and GPT2 with and without fine-tuning, where T5 is used as representative of the Text2Text LM. Besides, since GPT2 is pre-trained on corpus of all the web pages scraped from outbound links on Reddit, which has domain discrepancy from News, therefore, we fine-tune GPT2 to C4-realnewslike to eliminate this potential gap.

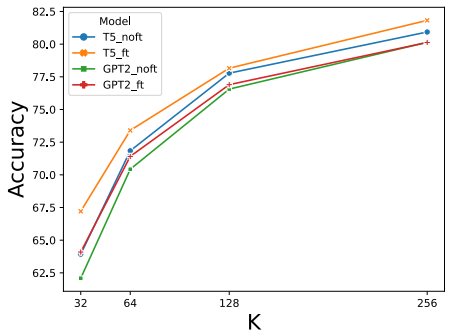

Figure 1: Comparison among T5 fine-tuned under **S2S** scheme(**T5_ft**) and T5 off-the-shelf version(**T5_noft**); GPT2 off-the-shelf(**GPT2_noft**), GPT2 fine-tuned on *C4-realnewslike*(**GPT2_ft**) in AG topics classification task, transformer-based classifier

As shown in Figure 1, 1) Removing fine-tuning always reduce the performance under various K, for both T5 and GPT2, which justifies the necessity of domain adaption and appropriate fine-tuning scheme design; 2) The auto-regressive GPT2 underperforms T5, which indicates that the structure of seq2seq is more suitable for generation-based DA. We will analyze this observation later.

Limitations. Same with the previous findings (Longpre et al., 2020), when the backbone classifier is PLM-based, as shown in Table 2, the gains are not significant or even become adverse. It is more clear in the sentiment classification, where various DA methods fail to ameliorate to a large extent; sometimes even hurt the performance when K is large. Also, our proposed methods do not gap too much away from baselines. For the topic classification, we can still witness an unignorable boost from BART fine-tuned under S2S scheme, when $K \leq 64$; however, it still suffers diminishing utility when $K \geq 128$.

Discussion. In line with findings from table 1 and 2, fine-tuning BART under S2S scheme can be a good practice when employing DA in sentiment classification. There are a variety of noising transformations, such as text infilling and sentence shuffling, in the pre-training stage of BART. Therefore, BART's ability of denoising corrupted documents in pre-training is more closely related to our S2S scheme in review corpus which presents more chal-

[3] $std \leq 0.1$, to save space we do not list them in the table

Table 1: Comparison with baselines under **transformer**-based classifier and various K settings, $|D_{train}| = |D_{syn}|$

Methods			AG				SST-2			
			K=32	K=64	K=128	K=256	K=32	K=64	K=128	K=256
	No DA		58.89	68.00	75.05	79.85	55.89	59.53	63.37	66.79
Baselines	EDA		59.59	68.61	74.88	80.55	55.89	59.33	63.11	66.62
	BT		59.96	69.21	74.67	79.90	56.19	59.88	63.15	66.64
	CBERT		59.81	69.97	75.89	80.03	56.98	59.98	63.45	66.97
	LAMBDA		60.02	69.34	75.37	80.46	56.77	60.02	63.29	66.18
Ours.	T5	S2S	**67.21**	**73.40**	**78.16**	**81.82**	57.28	60.71	63.92	67.03
		P2P	65.65	72.83	77.96	81.34	57.16	60.39	63.67	66.95
	BART	S2S	65.16	72.34	77.00	80.77	**58.21**	**61.43**	**64.86**	**67.30**
		P2P	64.99	71.77	76.51	80.91	57.72	61.37	64.17	66.96

Table 2: Comparison with baselines under **BERT**-based classifier and various K settings, $|D_{train}| = |D_{syn}|$.

Methods			AG				SST-2			
			K=32	K=64	K=128	K=256	K=32	K=64	K=128	K=256
	No DA		84.22	86.82	87.54	88.03	70.10	78.30	**84.93**	**86.90**
Baselines	EDA		85.13	86.45	87.70	88.15	72.19	79.15	84.65	85.46
	BT		85.12	86.60	87.18	88.16	76.94	81.04	84.27	85.32
	CBERT		85.28	86.79	87.37	88.01	74.09	80.07	84.38	85.88
	LAMBDA		85.07	86.55	87.21	87.98	75.08	80.32	84.55	85.98
Ours.	T5	S2S	84.83	86.39	87.28	87.96	70.87	79.15	84.08	85.76
		P2P	84.76	86.52	87.29	87.99	70.83	78.79	83.95	83.84
	BART	S2S	**85.35**	**86.84**	87.62	**88.26**	**79.81**	**82.25**	84.59	85.99
		P2P	85.17	86.81	**87.71**	88.07	78.42	82.14	84.78	86.32

lenges and makes BART more powerful.

For topic classification, employing T5 is a relatively better choice. As T5's pre-training task is fill-in-the-blank-style denoising objectives (span corruption and recovery), T5 primarily focuses on filling in dropped-out spans of text, which forces T5 to answer cloze questions based on "knowledge". This is more conducive to topic classification DA where bringing in more related entities(acquiring rich knowledge) is more crucial than adjusting sentence order or guaranteeing coherency.

GPT2 is widely used in previous generation-based DA, it is true that during inference, GPT2 is rambling on its own previous output, making generation prone to be off-topic that can lose fidelity in DA. In addition, GPT2 is a pure decoder model, while T5/BART consists of encoder and decoder. In other words, unlike the auto-regressive generation, T5 belongs to *directed generation*. Theoretically, T5/BART brings more advantages because of encoder-decoder attention layer which helps the generative decoder focus on appropriate places in the source text. This is the main reason why we introduce T5/BART into DA and its effectiveness is justified.

4 Conclusion and Future Work

In this paper, we propose to use text-to-text LMs as a new paradigm for data augmentation in text classification. Compared to other methods along this direction, our approach is robust, easy-to-implement and does not need laborious human intervention. In future work, it is worth exploring more tailored fine-tuning scheme for DA tasks under various scenarios.

5 Acknowledgments

This work was supported by the NSERC Discovery Grants.

References

Ateret Anaby-Tavor, Boaz Carmeli, Esther Goldbraich, Amir Kantor, George Kour, Segev Shlomov, Naama Tepper, and Naama Zwerdling. 2020. Not enough data? deep learning to the rescue! In *AAAI*, pages 7383–7390.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the NAACL: HLT*, pages 4171–4186.

Jesse Dodge, Gabriel Ilharco, Roy Schwartz, Ali Farhadi, Hannaneh Hajishirzi, and Noah Smith. 2020. Fine-tuning pretrained language models: Weight initializations, data orders, and early stopping. *CoRR*, abs/2002.06305.

Raphael Gontijo-Lopes, Sylvia J Smullin, Ekin D Cubuk, and Ethan Dyer. 2020. Affinity and diversity: Quantifying mechanisms of data augmentation. *arXiv preprint arXiv:2002.08973*.

Diederik P Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. *CoRR*, abs/1412.6980.

Sosuke Kobayashi. 2018. Contextual augmentation: Data augmentation by words with paradigmatic relations. In *Proceedings of the 2018 NAACL: HLT, Volume 2 (Short Papers)*, pages 452–457.

Varun Kumar, Ashutosh Choudhary, and Eunah Cho. 2020. Data augmentation using pre-trained transformer models. *CoRR*, abs/2003.02245.

Chin-Yew Lin. 2004. Rouge: A package for automatic evaluation of summaries. In *Text summarization branches out*, pages 74–81.

Ruibo Liu, Guangxuan Xu, Chenyan Jia, Weicheng Ma, Lili Wang, and Soroush Vosoughi. 2020. Data boost: Text data augmentation through reinforcement learning guided conditional generation. In *EMNLP*, pages 9031–9041.

Shayne Longpre, Yu Wang, and Chris DuBois. 2020. How effective is task-agnostic data augmentation for pretrained transformers? In *Findings of the ACL: EMNLP*, pages 4401–4411.

Andrew L. Maas, Raymond E. Daly, Peter T. Pham, Dan Huang, Andrew Y. Ng, and Christopher Potts. 2011. Learning word vectors for sentiment analysis. In *Proceedings of the 49th Annual Meeting of the ACL: HLT*, pages 142–150, Portland, Oregon, USA. ACL.

Jason Phang, Thibault Févry, and Samuel R Bowman. 2018. Sentence encoders on stilts: Supplementary training on intermediate labeled-data tasks. *arXiv preprint arXiv:1811.01088*.

Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. 2019. Language models are unsupervised multitask learners. *OpenAI blog*, 1(8):9.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J Liu. 2019. Exploring the limits of transfer learning with a unified text-to-text transformer. *CoRR*, abs/1910.10683.

Sebastian Ruder. 2021. Recent Advances in Language Model Fine-tuning. `http://ruder.io/recent-advances-lm-fine-tuning`.

Gaurav Sahu, Pau Rodriguez, Issam H Laradji, Parmida Atighehchian, David Vazquez, and Dzmitry Bahdanau. 2022. Data augmentation for intent classification with off-the-shelf large language models. *arXiv preprint arXiv:2204.01959*.

Rico Sennrich, Barry Haddow, and Alexandra Birch. 2016. Improving neural machine translation models with monolingual data. In *ACL (Volume 1: Long Papers)*, pages 86–96.

Richard Socher, Alex Perelygin, Jean Wu, Jason Chuang, Christopher D Manning, Andrew Y Ng, and Christopher Potts. 2013. Recursive deep models for semantic compositionality over a sentiment treebank. In *EMNLP*, pages 1631–1642.

Kaitao Song, Xu Tan, Tao Qin, Jianfeng Lu, and Tie-Yan Liu. 2019. MASS: Masked sequence to sequence pre-training for language generation. In *Proceedings of the 36th ICML*, volume 97 of *PMLR*, pages 5926–5936. PMLR.

Yufei Wang, Can Xu, Qingfeng Sun, Huang Hu, Chongyang Tao, Xiubo Geng, and Daxin Jiang. 2022. Promda: Prompt-based data augmentation for low-resource nlu tasks. *arXiv preprint arXiv:2202.12499*.

Jason Wei and Kai Zou. 2019. Eda: Easy data augmentation techniques for boosting performance on text classification tasks. In *EMNLP-IJCNLP*, pages 6383–6389.

Lilian Weng. 2022. Learning with not enough data part 3: Data generation. *lilianweng.github.io*.

Xing Wu, Shangwen Lv, Liangjun Zang, Jizhong Han, and Songlin Hu. 2019. Conditional bert contextual augmentation. In *ICCS*, pages 84–95. Springer.

Kang Min Yoo, Dongju Park, Jaewook Kang, Sang-Woo Lee, and Woomyeong Park. 2021. Gpt3mix: Leveraging large-scale language models for text augmentation. *CoRR*, abs/2104.08826.

Xiang Zhang, Junbo Zhao, and Yann LeCun. 2015. Character-level convolutional networks for text classification. In *NIPS-Volume 1*, pages 649–657.

A Implementation details

All experiments are conducted on Linux platform with GPU instance of Nvidia Tesla V100 type.

For the fine-tuning stage, we adopt the script from huggingface-transformers [4]. All the datasets we use are download from Huggingface-datasets[5]. We set the maximum length of both the source and target text to 256 which break the balance between performance and efficiency. Batch size is set to 16 and learning rate is $1e^{-5}$. Other parameters follow the default setting. For the fine-tuning corpus, we randomly split out 5% for the review dataset as validation set while for C4 corpus, the validation set is already officially split. We monitor the rouge score(Lin, 2004) at each epoch and pick the model of the best performance [6].

For the DA experiments stage, following previous studies, we set the optimizer as Adam (Kingma and Ba, 2014) with an initial learning rate of $4e^{-5}$ for training the classifier. The light-weighted Transformer-based classifier is referred to Keras implementation[7]. Pre-trained BERT is downloaded from Tensorflow Hub[8]. In each trial we run the training for 100 epochs and record the best accuracy on test set which is officially provided. We keep the classifier training settings exactly the same for all trials with and without DA, to ensure fairness. Therefore, the only difference exists in the introduction of D_{syn} produced from various approaches including ours or baselines.

In the generation process of T5 and BART, we set maximum length limit: 128 for AG and 64 for SST-2 DA scenario, based on the average length of samples in D_{train}. Therefore, generation is terminated when the special EOS token is ejected or the length of the generated text reach this limit. Nucleus sampling is used in generation ($P = 0.9$), to avoid sampling egregiously wrong tokens, but preserve variety when the highest scoring tokens have low confidence. Temperature and repetition penalty is set to 1.2. We only apply basic post-processing to filter generated examples that are too short or full of punctuation or repetitions which rarely happen in practice.

Among the baseline methods, we follow the optimal settings from the original papers. We set the intermediary language to Chinese for **BT**.

Our source code is released in Github repository[9].

Table 3: Descriptions of topic and sentiment categorization datasets.

Data	Labels	Domain
AG	World, Sports, Business, Tech	topic
SST-2	Positive, Negative	sentiment

[4] https://github.com/huggingface/transformers/tree/main/examples/pytorch/summarization

[5] https://huggingface.co/datasets

[6] We find that 1 or 2 epoch is always sufficient to convergence as the rouge metrics on validation set does not grow anymore.

[7] https://keras.io/examples/nlp/text_classification_with_transformer

[8] https://tfhub.dev/tensorflow/bert_en_uncased_L-12_H-768_A-12/4

[9] https://github.com/yananchen1989/PLMs_text_classification

ConTextING: Granting Document-Wise Contextual Embeddings to Graph Neural Networks for Inductive Text Classification

Yen-Hao Huang[†] and **Yi-Hsin Chen**[‡] and **Yi-Shin Chen**[*]
Institute of Information Systems and Applications[†*]
Department of Computer Science[*]
National Tsing Hua University, Hsinchu, Taiwan
Dcard, Taipei, Taiwan[‡]
{yenhao0218[†], eunicebes[‡], yishin[*]}@gmail.com

Abstract

Graph neural networks (GNNs) have been recently applied in natural language processing. Various GNN research studies are proposed to learn node interactions within the local graph of each document that contains words, sentences, or topics for inductive text classification. However, most inductive GNNs that are built on a word graph generally take global word embeddings as node features, without referring to document-wise contextual information. Consequently, we find that BERT models can perform better than inductive GNNs. An intuitive follow-up approach is to enrich GNNs with contextual embeddings from BERT, yet there is a lack of related research. In this work, we propose a simple yet effective unified model, coined *ConTextING*, with a joint training mechanism to learn from both document embeddings and contextual word interactions simultaneously. Our experiments show that ConTextING outperforms pure inductive GNNs and BERT-style models. The analyses also highlight the benefits of the sub-word graph and joint training with separated classifiers.

1 Introduction

Recently, the methods of non-sequential text modeling have gained attention, particularly for graph neural networks (GNNs) that learn document representation from graph structures. Most GNNs (Yao et al., 2019; Liu et al., 2020; Lin et al., 2021) are transductive since they are designed and built on a single heterogeneous graph, which connects all of the documents and words, including the training and testing data. Since testing documents must be used in training, transductive GNNs cannot be applied to new unseen documents. Thus, inductive learning GNNs (Huang et al., 2019; Nikolentzos et al., 2020; Zhang et al., 2020) have been proposed by representing each document in its own graph structure of local word interactions, with pretrained word embedding initialized on each word node. However, the global word embeddings are irrelevant to target documents, and graph structures might not capture the context well since text is usually produced in sequential order.

Modern transformer-based (Vaswani et al., 2017) pretrained models, such as BERT (Devlin et al., 2019), have shown their effectiveness in capturing context by sequentially modeling documents. In this study, we have also found that BERT-style models alone can outperform existing inductive GNNs. An intuitive method for enhancing inductive GNNs incorporates BERT contextual embeddings with GNNs for final text classification. For the most relevant research studies, Lu et al. (2020) adopted a GNN on a global vocabulary graph to enrich the token embeddings in BERT, and He et al. (2020) focused on sentences comparison tasks by feeding BERT contextual embeddings into a dependency graph. However, for text classification with inductive GNNs, there is a lack of reports on BERT and BERT-based GNNs. Only Lin et al. (2021) have adopted BERT's document embeddings for transductive GNNs.

Motivated by the recent success of inductive GNNs and the strengths of pretrained BERT models, in this work, we further consider the fact that these two types of models have different objectives. The former focuses on learning local syntactic word interactions, and the latter captures the context-aware semantics of words. To collaboratively join GNN and BERT models, we propose a unified model for learning *contextual inductive document representation via graph neural networks*, coined *ConTextING*, where each model has its own classifier for its own objective. A sub-word graph is adopted in ConTextING to focus more on fine-grained syntactic word usages, such as pre-/post-fix characters, which avoid over-focusing on content-specific word usages but maintain the flexibility in

[*]The corresponding author.

Proceedings of the 29th International Conference on Computational Linguistics, pages 1163–1168
October 12–17, 2022.

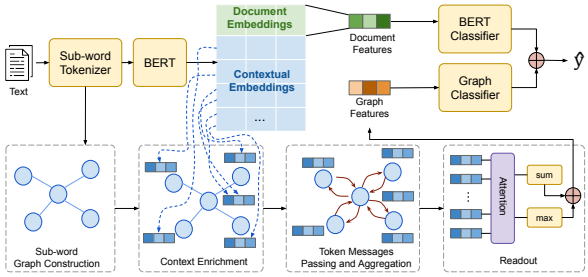

Figure 1: Overall framework of ConTextING.

accommodating to new words.

With this study, we make the following contributions: (1) We are the first to highlight the fact that pure GNNs are not superior to BERT and to provide detailed comparisons of state-of-the-arts (SOTAs), as well as the model variants. (2) We present ConTextING, a simple yet effective unification of BERT and GNN that yields results superior to pure inductive GNNs and BERT on a wide range of datasets. (3) We provide few-shot settings to illustrate the model's robustness to unseen words because of the sub-word adaptation.

2 ConTextING

This work proposed an unified model that consists of a BERT and a GNN modules as shown in Figure 1. ConTextING seamlessly enriches document-wise contextual information from a BERT-style model to the inductive GNN and makes final predictions based on the decisions of the two modules.

2.1 BERT-style Document Encoder

Given a text document, it is first tokenized into a sequence of sub-word tokens $\mathcal{T} = \{t_i\}$, and fed into the BERT-style model to obtain its document embeddings, $\mathbf{X} \in \mathbb{R}^{\delta}$ (from `[CLS]` token), and contextual embeddings, $\tilde{\mathbf{X}} \in \mathbb{R}^{|\mathcal{T}| \times \delta}$ for its tokens $\mathcal{T}$, where i dentoes the i-th sub-word in the document, and δ represents the hidden dimension of the BERT-style model.

Compared with the conventional GNNs, which utilize pretrained word embeddings (e.g. GloVe), the adaptation of contextual embeddings can capture local meanings within each document.

2.2 Sub-word Graph Construction

In contrast to previous GNNs, ConTextING constructs graphs from smaller word units—that is, the sub-word tokens—to capture more fine-grained text clues, such as the pre-/post-fix details of word usages. Such design can reduce the influence of topic-sensitive words and achieve robustness to rare words (Sennrich et al., 2016). For the sub-word graph, the sub-words are tokenized, based on byte-pair encoding (Sennrich et al., 2016) or WordPiece (Schuster and Nakajima, 2012) algorithms, according to the document encoder. The sub-word graph is formally defined in the following manner:

Definition 1. (Sub-word Graph) A sub-word graph is denoted as $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ where vertices $\mathcal{V} \in \mathcal{T}$ represent unique sub-words and edges $\mathcal{E}$ are co-occurrences between sub-words.

The co-occurrences describe the preferences for word usages within the given document, which are obtained by a fixed-length sliding window on the sequence of sub-word tokens $\mathcal{T}$. The connectivity of the sub-word graph is calculated, following the work of Yao et al. (2019) as in Definition 2.

Definition 2. (Sub-word Connectivity) Let (v_i, v_j), $\mathbf{A}$ denote two linked sub-word nodes and the adjacency matrix of graph $\mathcal{G}$, respectively. The weight of this linked edge $\mathbf{A}_{i,j}$ is given by

$$\mathbf{A}_{i,j} = \begin{cases} \mathrm{PMI}(i,j), & v_i \neq v_j, \mathrm{PMI}(i,j) > 0 \\ 0, & \text{otherwise} \end{cases} \tag{1}$$

where $\mathrm{PMI}(i,j) = \log \frac{p(i,j)}{p(i)p(j)}$ denotes the *point-wise mutual information*, $p(i), p(i)$ signify the probabilities of the sub-words' occurrence in all sliding windows, and $p(i,j)$ represents the probability of two sub-words' co-occurrence.

2.3 Context Enrichment

Given the fact that $|\mathcal{V}| \leq |\mathcal{T}|$ for a document, in order to jointly learn word interactions with contextual information in a graph view, it is necessary to define a mapping matrix $\mathbf{M} \in \mathbb{R}^{|\mathcal{V}| \times |\mathcal{T}|}$ for converting the features of sub-word t_j to node v_i by

$$\mathbf{M}_{i,j} = \begin{cases} 1/\mathrm{freq}(v_i), & v_i = v_j \\ 0, & \text{otherwise} \end{cases} \tag{2}$$

where $\mathrm{freq}(v_i)$ denotes the occurrences of each node v_i in $\mathcal{T}$. The contextual node representation $\mathbf{H} \in \mathbb{R}^{|\mathcal{V}| \times \delta}$ is then retrieved by $\mathbf{H} = \mathbf{M}\tilde{\mathbf{X}}$.

2.4 Token Messages Passing and Aggregation

To learn the word interactions, a token message passing and aggregation (TMPA) step is adopted. With the success of the gated structure and attention mechanism in natural language processing, in this work, we simply adopted gated

graph recurrent units (Li et al., 2015) as TextING (Zhang et al., 2020) and the graph attention network (GAT) (Veličković et al., 2018) on sub-word graph $\mathcal{G}$. The adoption of other graph convolution methods (Kipf and Welling, 2017; Hamilton et al., 2017) are left for future works.

Formally, the value of a node v can be updated by aggregating the information $\mathbf{H}_{\mathcal{N}}^{(v)} \in \mathbb{R}^{|\mathcal{N}|\times\delta}$ from its 1-hop neighbors and its current value $\mathbf{H}^{(v)} \in \mathbb{R}^{\delta}$, where $|\mathcal{N}|$ denotes the number of neighbors for node v. One updating process refers to one TMPA operation for sub-word interactions, based on the study of Zhang et al. (2020). By stacking the TMPA for τ times, each node can obtain features from neighbors within the τ-hop distance. The node representation after τ TMPA is denoted by $\mathbf{H}^{(v,\tau)}$.

2.5 Graph Readout and Jointly Learning

A graph readout step is applied to aggregate the final node embeddings in order to obtain a graph-level document representation $\mathbf{H}^{(G)}$ as follows:

$$\widehat{\mathbf{H}}^{(v)} = \sigma(f_1(\mathbf{H}^{(v,\tau)})) \odot \tanh(f_2(\mathbf{H}^{(v,\tau)})) \quad (3)$$

$$\mathbf{H}^{(G)} = \frac{1}{|\mathbb{V}|}\sum_{v\in\mathbb{V}} \widehat{\mathbf{H}}^{(v)} + \text{Maxpooling}(\widehat{\mathbf{H}}) \quad (4)$$

where f_1 and f_2 represent two dense layers for the weighted embeddings of each sub-word node through a soft attention mechanism by a sigmoid function σ. The average summation and feature-wise max-pooling functions are subsequently applied to obtain graph representation $\mathbf{H}^{(G)}$.

To this end, two features are extracted: the sequential document features $\mathbf{X}$ from the BERT module and the non-sequential features $\mathbf{H}^{(G)}$ from the GNN module. It is worth noting that in this work, we consider that the aims of BERT and the GNN are essentially different (sequential and non-sequential modeling, respectively). A directly concatenating, adding, or averaging operation $\mathbf{X}$ and $\mathbf{H}^{(G)}$ may blur the distinction between the objectives of these two models. We thus adopt two classifiers separately for each model with a linear interpolation (Lin et al., 2021) to regulate the objectives of BERT and GNN for final prediction.

$$\hat{y}^{(\text{BERT})} = \text{softmax}(\mathbf{W}_x\mathbf{X} + \mathbf{b}_x) \quad (5)$$

$$\hat{y}^{(\text{Graph})} = \text{softmax}(\mathbf{W}_g\mathbf{H}^{(G)} + \mathbf{b}_g) \quad (6)$$

$$\hat{y} = (1-\eta)\hat{y}^{(\text{BERT})} + \eta\hat{y}^{(\text{Graph})} \quad (7)$$

where $\eta \in [0, 1]$ denotes a hyper-parameter to decide the main objective between BERT and GNN. A higher η value indicates the more focuses on non-sequential word interactions. The use of the automatic mechanism to determine η is left for our future works.

3 Experiments

The proposed model is evaluated by addressing these concerns: (1) Is ConTextING better than pure BERT-style models and inductive GNNs for text classification? (2) Can the ConTextING achieve satisfactory results with limited training data?

Datasets. Five common benchmark datasets for evaluating GNNs are adopted and pre-processed, following the works of Yao et al. (2019); Zhang et al. (2020); Lin et al. (2021), which are medical abstracts with 23 diseases classes (Ohsumed); movie reviews (MR) with sentiment polarities; Reuters newswire items with 8 (R8) and 52 (R52) categories; and 20NewsGroups (20NG) with 20 categories, respectively.

Baselines. The compared baselines include (1) *traditional deep learning models* with GloVe embeddings (Pennington et al., 2014): textCNN (Kim, 2014), LSTM (Liu et al., 2016) and bi-directional LSTM (Bi-LSTM); (2) *SOTA language models*: BERT (BT) (Devlin et al., 2019) and RoBERTa (RBT) (Liu et al., 2019); (3) *SOTA inductive GNNs*: TextGCN-*ind* (an inductive version by Yao et al. (2019)), text-level GNN (Huang et al., 2019), TextING (Zhang et al., 2020), and HyperGAT-*ind* (Ding et al., 2020) (topics are learned without testing data.); and (4) GNN-enriched BERT classifier: VGCN-BERT (Lu et al., 2020).

Since the codes released by HyperGAT's authors include testing data when producing its topic features, HyperGAT-*ind* is then reproduced by excluding the testing data when generating the topics. VGCN-BERT is also reproduced as its authors only reported the F1 score on MR. All of the reproduced results are based on the original authors' codes[1] and the parameters described in the original papers.

The results of some baselines are obtained from Zhang et al. (2020); Ding et al. (2020) for a fair comparison. Note that they both followed the same setting and their baseline results are obtained from the work by Yao et al. (2019).

Experimental Settings. ConTextING consists of a base version of BT/RBT and a two-layer gated

[1] https://github.com/kaize0409/HyperGAT; https://github.com/Louis-udm/VGCN-BERT; https://github.com/CRIPAC-DIG/TextING

Method	Ohsumed	MR	R8	R52	20NG
CNN+GloVe	58.44	77.75	95.71	87.59	82.15
LSTM+GloVe	51.10	77.33	96.09	90.48	75.43
Bi-LSTM+GloVe	49.27	77.68	96.31	90.54	73.18
BERT (BT)	68.74	85.88	97.26	96.26	84.54
RoBERTa (RBT)	69.86	87.08	97.35	95.48	84.07
TextGCN-*ind*	57.70	74.80	95.78	88.20	83.31
Text-level GNN	69.40	75.47	97.89	94.60	84.16
TextING	70.42	79.82	98.04	95.48	82.48
HyperGAT-ind	67.33	77.08	97.03	94.55	84.63
VGCN-BERT	70.19	85.93	97.89	95.87	55.76
ConTextING-BT	71.28	86.01	97.91	**96.52**	**86.19**
w. GAT-BT	71.51	86.16	97.96	96.28	**86.25**
ConTextING-RBT	**72.53**	**89.43**	**98.13**	**96.40**	85.00
w. GAT-RBT	**72.06**	**89.24**	**98.09**	96.15	84.97

Table 1: Test accuracy comparison of inductive methods. The results of the methods highlighted in *italics* are produced by this work.

graph recurrent unit with 768 hidden size (800 for the GAT variant). The window size, η, dropout, learning rate, loss function and train epochs are set as 3 (widely used in GNNs), $0.9/0.3$ (MR/others), 0.5, $1e^{-5}$, negative log likelihood, and 80, respectively. All results are averaged over 10 runs.

3.1 Benchmark Text Classification

Test Performance. Table 1 summarizes the test accuracy of each model. Overall, it can be observed that ConTextING (four different variants in total) generally beats all of the baselines, including the SOTA models, on every dataset. This indicates the benefits of integrating text modeling in both sequential and non-sequential manners. By concatenating word embeddings from a global vocabulary graph (VGCN) to BERT, VGCN-BERT also performs well on these benchmark datasets, except for the 20NG[2]. The low accuracy of 20NG might be caused by the unbalanced distribution of token embeddings between BERT and the graph embeddings produced by the VGCN component due to a highly sparse vocabulary graph from a large amount of vocabulary ($> 25k$).

Performance Boost Over Pure BERT and GNNs. Compared with pure BT, RBT, and GNNs, the results reveal that ConTextING consistently obtains $1 - 2.7$ points of gains on accuracy from pure BT and RBT for all datasets. Regarding GNNs, it is discovered that pure GNNs are not superior to BERT-style models. Similar boosts are also observed, particularly for the MR dataset, which improves accuracy by approximately 9 points from SOTA GNN (TextING). Such significant gains are mostly contributed by the BERT module, as BT and RBT themselves can obtain high accuracy with the merits of large-scale pretraining. In contrast, on Ohsumed, although BT/RBT are beaten by inductive GNNs, ConTextING-RBT can still obtain a high score of 72.53 on accuracy.

Method	MR	Ohsumed
TextGCN	53.15 (-23)	47.24 (-21)
TextING	64.43 (-15)	*51.40 (-19)*
RBT	69.16 (-18)	50.51 (-19)
ConTextING-RBT	**73.14** (-16)	**53.67** (-19)
# Samples/words in Training	40/465	*448/7,009
# New Words in Test	18,299	7,148

Table 2: Test accuracy under a few-shot setting. Values in parentheses are performance reductions from Table 1.

3.2 Few-shot Inductive Capability

To examine ConTextING's inductive capability, we conduct few-shot learning experiments on benchmark datasets according to the setting by Zhang et al. (2020). The number of training samples is limited to a maximum of 20 labeled documents per class. Consequently, most words in the test set are unseen in these settings. The results are compared with those of TextGCN and TextING reported by Zhang et al. (2020)[3]. The results presented in Table 2 show that our model is the most robust one among a few training samples. The observations are basically aligned with the results in Table 1. For MR, the RBT alone could perform better than baselines. Although RBT has a worse result than the one by TextING on Ohsumed, ConTextING could further boost the performance of RBT on both datasets with the aid of the GNN module. By taking sub-words, which are naturally robust to the new words, as input for BERT and GNN modules, ConTextING is thus more stable.

3.3 Model Analysis

Word and Sub-word Graph Comparison. To evaluate the effectiveness of the sub-word graph, we adapt TextING but modify its input graph into a word graph and a sub-word graph with the fixed fine-tuned RBT's embeddings with window size 3 (best for a word graph). Figure 2 shows that TextING with a sub-word graph is able to perform better consistently on different benchmark datasets.

[2]The reproduced *VGCN-BERT* results are consistent with the reported values in the original paper. By initializing *VGCN-BERT* with a pretrained BT, it can obtain 60 on the accuracy of 20NG. However, its accuracy decreases after the first epoch.

[3]Note that there are only nine samples in the training set for the "C22" class on Ohsumed (which differs from those in the original report); thus, TextING is reproduced by using the original authors' codes.

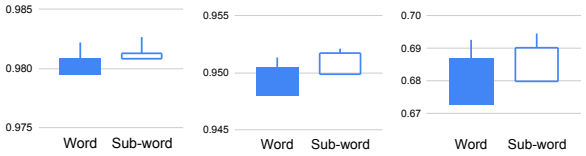

(a) R8 (b) R52 (c) Ohsumed

Figure 2: Accuracy of using word and sub-word graph.

Model Variants (Acc. ± Std.)	**R52**	**Ohsumed**	**20NG**
ConTextING-RBT	96.4±.2	72.5±.3	85.0±.4
(i) w/o. joint train (concat. Embd.)	96.2±.2	72.4±.7	84.4±.5
(ii) w/o. RBT classifier	95.8±.5	72.2±.6	84.5±.3
(iii) w. fix RBT Word Embd.	95.2±.1	66.7±.6	81.2±.4
(iv) w. fix tuned RBT Word Embd.	94.9±.2	68.0±.8	83.9±.4

Table 3: Ablation studies on ConTextING variants.

Effects of Joint Training. To examine the effectiveness of joint training, common methods for aggregating the BERT-style model (RBT) and GNN are implemented, with careful inspections of different hyper-parameters. Table 3 shows the superior performance of ConTextING with joint training. With RBT updated during training, (i) and (ii) can obtain high accuracy under a low learning rate; however, it is still slightly worse than adopting joint training. As for MR, comparable results could be achieved without the joint training.

For (iii) and (iv), the effects of RBT's contextualized embeddings are examined, where the embeddings are fed into the pure GNNs module of ConTextING without training the RBT. In other words, ConTextING is simplified as TextING architecture, with its node features initialized as contextualized embeddings. The results show that the contextualized embedding by the fine-tuned RBT improves by 1.3 and 2.7 points on Ohsumed and 20NG, respectively, over the RBT embedding without finetuning. It is also observed that (iii) and (iv) perform worse than TextING with its original GloVe embeddings, which shows that GNNs alone may be unable to process the high-dimension embeddings well (i.e. 300 v.s. 768). Similarly, the unification in (iii) is also found easily-fail-to-converge (60%) on MR.

BERT and GNN Embedding Comparison. To indicate the difference in what BERT and GNN modules have learned, t-SNE (Van der Maaten and Hinton, 2008) is applied to visualize the corresponding document features in ConTextING-RBT. Figure 3 reveals that the GNN module produces a representation different from RBT's one. Specifically, the RBT module tends to mess up several documents (center of Figure 3a), while the GNN module can distinguish them more correctly. By jointly training and predicting on *two different classifiers*, ConTextING could achieve superior performance than each of them individually.

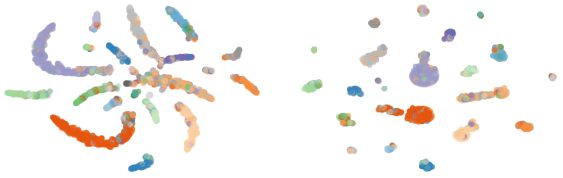

(a) RoBERTa (b) GNN

Figure 3: Visualizations of BERT and GNN modules in ConTextING on Ohsumed's test documents. The color of a node corresponds to the node's class.

4 Conclusion

In this paper, we have proposed ConTextING, which successfully learns document embeddings sequentially and contextual word interactions non-sequentially at the same time. Various context encoders or GNNs are also allowed to build ConTextING on top of this framework. In the future, we aim to involve GNNs in a large-scale pretraining process in combination with BERT.

Acknowledgements

This research is supported by the National Science and Technology Council in Taiwan (NTSC 111-2221-E-007-110-MY3; NTSC 110-2221-E-007-085-MY3; NTSC 108-2221-E-007-064-MY3).

References

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, Minneapolis, Minnesota. Association for Computational Linguistics.

Kaize Ding, Jianling Wang, Jundong Li, Dingcheng Li, and Huan Liu. 2020. Be more with less: Hypergraph attention networks for inductive text classification. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 4927–4936.

William L Hamilton, Rex Ying, and Jure Leskovec. 2017. Inductive representation learning on large graphs. In *Proceedings of the 31st International Conference on Neural Information Processing Systems*, pages 1025–1035.

Qi He, Han Wang, and Yue Zhang. 2020. Enhancing generalization in natural language inference by syntax. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 4973–4978, Online. Association for Computational Linguistics.

Lianzhe Huang, Dehong Ma, Sujian Li, Xiaodong Zhang, and Houfeng Wang. 2019. Text level graph neural network for text classification. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 3444–3450.

Yoon Kim. 2014. Convolutional neural networks for sentence classification. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1746–1751, Doha, Qatar. Association for Computational Linguistics.

Thomas N Kipf and Max Welling. 2017. Semi-supervised classification with graph convolutional networks. *International Conference on Learning Representations*.

Yujia Li, Daniel Tarlow, Marc Brockschmidt, and Richard Zemel. 2015. Gated graph sequence neural networks. *International Conference on Learning Representations*.

Yuxiao Lin, Yuxian Meng, Xiaofei Sun, Qinghong Han, Kun Kuang, Jiwei Li, and Fei Wu. 2021. Bertgcn: Transductive text classification by combining gnn and bert. In *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*, pages 1456–1462.

Pengfei Liu, Xipeng Qiu, and Xuanjing Huang. 2016. Recurrent neural network for text classification with multi-task learning. In *IJCAI*.

Xien Liu, Xinxin You, Xiao Zhang, Ji Wu, and Ping Lv. 2020. Tensor graph convolutional networks for text classification. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pages 8409–8416.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Roberta: A robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692*.

Zhibin Lu, Pan Du, and Jian-Yun Nie. 2020. Vgcn-bert: Augmenting bert with graph embedding for text classification. In *Advances in Information Retrieval - 42nd European Conference on IR Research, ECIR 2020, Lisbon, Portugal, April 14-17, 2020, Proceedings, Part I*, volume 12035 of *Lecture Notes in Computer Science*, pages 369–382. Springer.

Giannis Nikolentzos, Antoine Tixier, and Michalis Vazirgiannis. 2020. Message passing attention networks for document understanding. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pages 8544–8551.

Jeffrey Pennington, Richard Socher, and Christopher Manning. 2014. Glove: Global vectors for word representation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1532–1543. ACL.

Mike Schuster and Kaisuke Nakajima. 2012. Japanese and korean voice search. In *2012 IEEE international conference on acoustics, speech and signal processing (ICASSP)*, pages 5149–5152. IEEE.

Rico Sennrich, Barry Haddow, and Alexandra Birch. 2016. Neural machine translation of rare words with subword units. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1715–1725.

Laurens Van der Maaten and Geoffrey Hinton. 2008. Visualizing data using t-sne. *Journal of machine learning research*, 9(11).

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in neural information processing systems*, pages 5998–6008.

Petar Veličković, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Liò, and Yoshua Bengio. 2018. Graph attention networks. In *International Conference on Learning Representations*.

Liang Yao, Chengsheng Mao, and Yuan Luo. 2019. Graph convolutional networks for text classification. In *Proceedings of the AAAI conference on artificial intelligence*, volume 33, pages 7370–7377.

Yufeng Zhang, Xueli Yu, Zeyu Cui, Shu Wu, Zhongzhen Wen, and Liang Wang. 2020. Every document owns its structure: Inductive text classification via graph neural networks. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 334–339.

Virtual Knowledge Graph Construction for Zero-Shot Domain-Specific Document Retrieval

Yeon Seonwoo[†*], **Seunghyun Yoon**[‡], **Franck Dernoncourt**[‡], **Trung Bui**[‡], **Alice Oh**[†]
[†]KAIST, [‡]Adobe Research,
yeon.seonwoo@kaist.ac.kr
{syoon, dernonco, bui}@adobe.com
alice.oh@kaist.edu

Abstract

Domain-specific documents cover terminologies and specialized knowledge. This has been the main challenge of domain-specific document retrieval systems. Previous approaches propose domain-adaptation and transfer learning methods to alleviate this problem. However, these approaches still follow the same document representation method in previous approaches; a document is embedded into a single vector. In this study, we propose VKGDR. VKGDR represents a given corpus into a graph of entities and their relations (known as a virtual knowledge graph) and computes the relevance between queries and documents based on the graph representation. We conduct three experiments 1) domain-specific document retrieval, 2) comparison of our virtual knowledge graph construction method with previous approaches, and 3) ablation study on each component of our virtual knowledge graph. From the results, we see that unsupervised VKGDR outperforms baselines in a zero-shot setting and even outperforms fully-supervised bi-encoder. We also verify that our virtual knowledge graph construction method results in better retrieval performance than previous approaches. [1]

1 Introduction

In domain-specific QA, building retrievers is challenging since queries and documents in a specific domain cover terminologies and specialized knowledge, which are not well covered in general documents. (Zhang et al., 2020; Ma et al., 2021; Yu et al., 2020, 2021). Another problem is the difficulty in building datasets for training retrievers. This problem comes from 1) the complexity of knowledge treated in the documents, and 2) costly dataset maintenance; recall that domain-specific documents are frequently updated (e.g., software manuals are updated whenever there is a version update) (Castelli et al., 2020; Nandy et al., 2021; Voorhees et al., 2021; Maia et al., 2018).

Recent domain-specific document retrieval studies propose domain-adaptation and transfer learning methods (Thakur et al., 2021; Ma et al., 2021; Beltagy et al., 2019; Gururangan et al., 2020; Chalkidis et al., 2020). However, these methods still use the conventional document representation method, embedding a document into a single vector. This is problematic because a single vector is insufficient to cover complex knowledge in a domain-specific document. Semi-structured knowledge representation methods effectively address this problem, but they have only been applied to open-domain documents. (Dhingra et al., 2020; Sun et al., 2021; Zhang et al., 2018; Sun et al., 2018; Yasunaga et al., 2021; Talmor and Berant, 2018).

In this paper, we propose an automatic virtual knowledge graph construction method for zero-shot domain-specific document retrieval. A virtual knowledge graph (VKG) is a graph representation of a corpus that consists of entities and their relations. In VKG, the relations are represented by relation vectors (Dhingra et al., 2020; Sun et al., 2021). This semi-structured representation enables explicit reasoning over the corpus. We apply this framework to domain-specific document retrieval. One of the key components of the VKG construction method is a relation encoder, which computes relation vectors of two entities. This study shows that previous supervision methods for relation encoders are insufficient for domain-specific documents, and we propose a novel distant-supervision method.

We validate VKGDR in three types of experiments. First, we conduct zero-shot domain-specific document retrieval on two domain-specific QA datasets: TechQA (Castelli et al., 2020) and PhotoshopQuiA (Dulceanu et al., 2018). The results show that VKGDR outperforms domain-adaptation and

[*]This work was done during an internship at Adobe Research.

[1]We provide the implementation of VKGDR at https://github.com/yeonsw/VKGDR

Proceedings of the 29th International Conference on Computational Linguistics, pages 1169–1178
October 12–17, 2022.

transfer learning methods. From this experiment, we also verify that unsupervised VKGDR outperforms a fully-supervised dense retriever. Second, we show that our distant-supervision method for training the relation encoder outperforms previous approaches. In this experiment, we construct VKGs with our relation encoders and baselines' encoders. Then, we measure the retrieval performance of each VKG. Third, we conduct an ablation study on two main components of a VKG, graph representation of a corpus and relation vectors. The results show that each component increases the retrieval performance of VKGDR by a large margin.

2 Related Work

A virtual knowledge graph is a graph representation of a corpus that consists of entities and their relations. The relations of entities are represented by relation vectors. Dhingra et al. (2020) propose a differentiable VKG for multi-hop QA. Their VKG is trained by the end-to-end supervision method on question-answer pairs. Sun et al. (2021) use VKG for knowledge graph QA. They apply relation encoders used in relation extraction studies to VKG construction and follow distant-supervision proposed by Soares et al. (2019). Our work provides a novel distant-supervision method for building a virtual knowledge graph for domain-specific documents. In section 5.1, we compare our methods with Sun et al. (2021) to validate the efficacy of our method.

Domain-specific documents cover complex knowledge and require advanced representation methods. Previous approaches in domain-specific document retrieval focus on a document encoder training method and data scarcity problem but still follow conventional document representation methods. Ma et al. (2021); Liang et al. (2020) augment domain-specific question-answer pairs from an external corpus and train their encoders on the dataset. Yu et al. (2020); Zhang et al. (2020) provide a pre-training method on domain-specific documents. We propose a novel virtual knowledge graph construction method and apply our method to domain-specific document retrieval.

3 Method

We propose a novel domain-specific document retrieval method, VKGDR, based on a virtual knowledge graph (VKG). VKGDR consists of two modules: a VKG construction module and a document retrieval module. A VKG is a graph representation of a given corpus that connects mentions with directed edges, and each directed edge has a relation vector. (Dhingra et al., 2020; Sun et al., 2021). The document retrieval module computes the similarity between queries and documents with the mention pairs and their relation vectors. We describe notations and details of each module in the following sections.

3.1 Notations

In this section, we define notations and terms used in our paper and VKG research (Dhingra et al., 2020). VKGDR takes a corpus and outputs a virtual knowledge graph. The corpus, $\mathcal{C}$ is a set of documents; $\mathcal{C} := \{d_1, ..., d_n\}$. A document is defined as a sequence of tokens; $d_k := [d_k^1, ..., d_k^{L_k}]$, where d_k^j is the j'th token of document d_k and L_k is the number of tokens in document d_k. VKGDR's entity extractor builds a set of entities [2], $\mathcal{E}$ and a set of mentions, $\mathcal{M}$. The definition of an entity is a named entity in the corpus, $\mathcal{C}$, and the definition of mention is a text segment in the corpus, $\mathcal{C}$, that corresponds to an entity in $\mathcal{E}$. Formally, the mention is defined as $m_i = \{d_k, a, b, e_j\}$; the mention m_i is a text segment starting from index a and end at index b in document d_k, which corresponds to entity e_j. Figure 1 shows the difference between mentions and entities. In the figure, the highlighted text segments are the mentions, and there are multiple mentions for each entity. For instance, entity "TRC 5011" appears multiple times in this document, and each text segment that refers to entity "TRC5011" is a mention of the entity.

3.2 Virtual Knowledge Graph

A virtual knowledge graph is a directed graph consisting of mentions and their relations. The relations are represented by relation vectors (Dhingra et al., 2020). Formally, we define an edge of the VKG as follows:

$$(m_a, m_b, \vec{r}_j).$$

This represents that there exists an edge directed from mention m_a to m_b, and $\vec{r}_j$ is the relation vector of the mention pair. Mention m_a is called the head, and m_b is called the tail. VKGDR constructs a virtual knowledge graph with the following steps: 1) connecting all relevant mentions and 2) computing relation vectors of edges. In our study, we

[2] We use the NER model provided by spaCy.

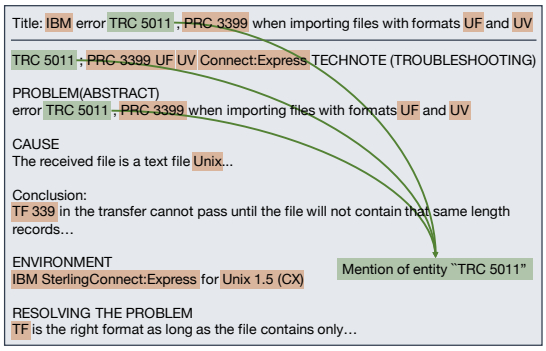

Figure 1: An example document of the TechQA dataset and mentions in the document. Orange and green highlights are the mentions. Mention is a text segment that refers to a certain entity. For example, entity "TRC 5011" appears multiple times in this document and each text segment that refers to "TRC 5011" is a mention of entity "TRC 5011".

assume two mentions are relevant if they appeared in the same document and connect the two mentions with directed edges in both directions. Thus, for a given document with n mentions, there are n^2 combinations of mention pairs and n^2 more mentions pairs since we connect mentions with directed edges in both directions.

3.3 Relation Embedding

Relation encoders compute relation vectors of mention pairs connected in a virtual knowledge graph. Relation encoders aim to embed mention pairs into a similar vector space if they are in similar relation. Previous approaches distantly-supervise relation encoders since training data is often unavailable. One of the previous approaches assumes that mention pairs referring to the same entity-pair have the same relation (Sun et al., 2021), and they train their relation encoder to maximize the similarity of these similar mention pairs. In this study, we propose a novel distant-supervision method for domain-specific documents.

Model Architecture: VKGDR's relation encoder (RE) takes a mention pair and computes the relation vector.

$$\vec{r}_{i,j} = \text{RE}(m_i = \{d_k, a, b, e_u\}, m_j = \{d_k, c, d, e_v\})$$

In previous relation embedding studies, relation encoders take preprocessed mention pairs as an input. The preprocessing steps are: 1) adding special tokens to the head and the tail mentions to indicate their direction and 2) masking the mentions (Mintz et al., 2009; Soares et al., 2019; Sun et al., 2021). Our relation embedding method is based on the previous approaches and proceeds following steps on mention pairs. For a given mention pair, $(m_i = (d_k, a, b, e_u), m_j = (d_k, c, d, e_v))$, we represent the two mentions in document d_k as follows:

$$(m_i, m_j) = [d_k^1, ..., \boldsymbol{e_u}, ..., \boldsymbol{e_v}, ..., d_k^{L_k}].$$

$\mathbf{e_v}$ and $\mathbf{e_u}$ are the tokens in document d_k corresponding to the two mentions. Next, we put special tokens, [H] and [T], to the mentions as follows:

$$[d_k^1, ...\boldsymbol{e_u}\textbf{, [H]}, ..., \boldsymbol{e_v}\textbf{, [T]}, ..., d_k^{L_k}].$$

Now, the above sequence of tokens represents the direction between the two mentions; without the special tokens, the relation encoder predicts the same relation vector for the opposite input, (m_j, m_i). Mention masking enables the relation encoder to compute the relation vector based on the context of the mention pairs, not based on their textual representation. We mask the mentions as follows:

$$[d_k^1, ..., \textbf{[ENT], [H]}, ..., \textbf{[ENT], [T]}, ..., d_k^{L_k}].$$

Figure 2 shows the input preprocessing step and the relation vector computation step. In this example, the green token is the head, and the red token is the tail. The head and tail tokens are inserted into the document, and the entities are masked with the special token. The relation encoder takes the whole sequence of tokens and computes contextualized

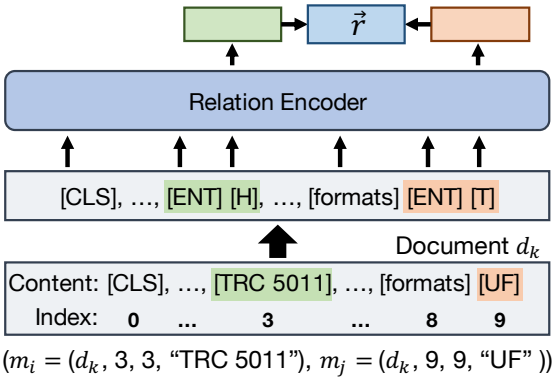

Figure 2: The inference process of the relation encoder of VKGDR. Two mentions are given to the relation encoder. Mention, m_i and m_j are text spans in document d_k located from index 3 to 3 and index 9 to 9. We mask mention tokens with [ENT] and indicate the head and the tail with [H] and [T]. The relation encoder takes this input and computes a relation vector of the two mentions.

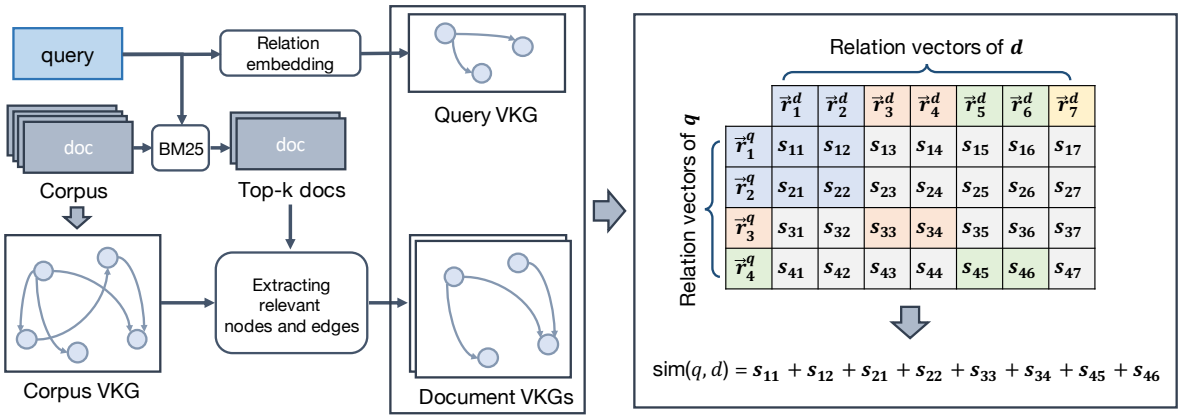

Figure 3: The document retrieval process of VKGDR. First, VKGDR pre-indexes the corpus VKG. Second, we extract relevant nodes and edges from the VKG for a given query since comparing the query with the entire VKG is computationally inefficient. Third, VKGDR transforms the query to a query VKG with the same VKG construction method used for the corpus VKG. Finally, we compare the two VKGs and compute their similarity. In the right part of this figure, we use a color-coding method to indicate the relation vectors that share the same entity pairs.

vector representations of the head and tail tokens. Then, we compute the relation vector from the two vectors with an additional MLP layer.

Training Process: This study proposes a novel distant-supervision method for building a virtual knowledge graph from domain-specific documents. Distant-supervision methods in previous relation embedding approaches proceed following steps: 1) heuristically annotate mention pairs in the same relation and 2) train the relation encoder to maximize the similarity between mention pairs in the same relation.

We propose a novel distant-supervision method for domain-specific documents. One of the previous approaches assumes mention pairs are in the same relation if they share the same entity pair (Soares et al., 2019; Sun et al., 2021). In domain-specific QA, mention pairs are often in different relations, and the relation varies depending on the context of the document. With the previous assumption, relation encoders predict similar relation vectors for mention pairs with the same entity pair even they are in different relations. We assume that the context of mention pairs is more important than the entities they refer to. In our approach, mention-pairs are in the same relation if they appeared in the same document. Formally, we train our relation encoder with the following method. For a given mention-pair, $p = (m_i, m_j)$, the positive sample (p^+) is a mention pair appeared in the same document, and the negative sample is mention pair in the different document. The loss function of our relation encoder is as follows:

$$L(p, p^+, p_1^-, ..., p_{\#\text{neg}}^-) = -\log\left(\frac{e^{\text{sim}(p,p^+)}}{e^{\text{sim}(p,p^+)} + \sum_{i=1}^{\#\text{neg}} e^{\text{sim}(p,p_i^-)}}\right), \quad (1)$$

where sim function is the dot product of the two relation vectors; $\text{sim}(p_1, p_2) = \vec{r}_{p_1}^{\intercal} \vec{r}_{p_2}$.

3.4 Document Retrieval Process

VKGDR uses a virtual knowledge graph to find the document most relevant to a given query. The document retrieval process of VKGDR follows four steps: 1) selecting top-k relevant documents with BM25, 2) extracting mention pairs appeared in the top-k documents and their relation vectors from the VKG, 3) constructing a VKG of a given query, and 4) finding the most relevant document by comparing the document VKGs and the VKG of the input query. In the first step, we select top-k documents relevant to the query. This is because our VKG consists of a huge number of mention pairs; comparison between the query and the VKG is computationally costly. Then, we construct document VKGs. A document VKG is a graph of mention pairs that appeared in the document. Since we retrieved top-k documents, we get k number of document VKGs. In the third step, we transform the query to a VKG; queries cannot be directly compared with document VKGs since they are in textual form. In this step, we use the same relation encoder used for computing the VKG of the given corpus. In the last step, VKGDR finds the most relevant document by comparing the query VKG

and the document VKGs. We describe the details of this VKG comparison process in the following section. Figure 3 shows an overall illustration of VKGDR.

Comparing two VKGs: The query VKG and the document VKG consist of several mention pairs as follows:

$$\text{VKG}_q = \{(m_h, m_t, \vec{r})_i\}_{i=1}^{k}$$
$$\text{VKG}_d = \{(m_h, m_t, \vec{r})_i\}_{i=1}^{n},$$

where VKG_q is the query VKG and VKG_d is the document VKG. VKGDR computes the similarity between two VKGs with the following equation:

$$\text{similarity}(q, d) = \sum_{\substack{(m_h^q, m_t^q, \vec{r}^{\,q}) \in \text{KTq} \\ (m_h^d, m_t^d, \vec{r}^{\,d}) \in \text{KTd}}} \mathbb{1}((m_h^d, m_t^d) = (m_h^q, m_t^q)) \vec{r}^{\,q\,\intercal} \vec{r}^{\,d},$$

where $\mathbb{1}$ is an indicator function that maps true condition to one and zero for false condition.

4 Experimental Setup

We validate domain-specific document retrieval performance of VKGDR in three experimental settings. In the first experiment, we evaluate VKGDR and baselines in a zero-shot setting. The zero-shot setting emulates the real-world problem of domain-specific document retrieval; training data is insufficient or absent. Additionally, we conduct the same experiment in a fully-supervised setting and show the efficacy of VKGDR. In the second experiment, we verify the efficacy of our proposed distant-supervision method by comparing our method with previous methods. We construct three VKGs with relation encoders trained with three different distant-supervision methods. The third experiment is an ablation study that evaluates each component in VKGDR. A virtual knowledge graph consists of two main components: 1) graph representation of a given corpus and 2) relation vectors of mention pairs. In this experiment, we evaluate 1) VKGDR without graph representation and 2) VKGDR without relation vectors and show the efficacy of each component. All experiments are conducted on two domain-specific QA datasets, TechQA and PhotoshopQuiA, and evaluated with document retrieval metrics, R@K and MRR. We describe details of the datasets and baselines in the following sections and describe evaluation metrics and hyper-parameter settings in Appendix A.1.

	Train	Dev	Test
TechQA	600	310	490
PhotoshopQuiA	2001	571	284

Table 1: The number of instances in the TechQA dataset and the PhotoshopQuiA dataset.

4.1 Datasets

TechQA: TechQA is a question answering dataset in the domain of IT support (Castelli et al., 2020). The questions ask about IBM products and applications running in computational environments supported by IBM. This dataset provides question-answer pairs and 800,000 technical notes that provide descriptions of IBM's products. Each question is annotated with 50 documents retrieved by BM25, and one of the 50 documents is the ground truth document. Thus, the task of this dataset is to find the correct document among the 50 documents. The numbers of question-answer pairs of TechQA is 1,400. Table 1 shows the detailed statistics of the TechQA dataset.

PhotoshopQuiA: PhotoshopQuiA is a non-factoid question-answering dataset on Adobe Photoshop (Dulceanu et al., 2018). The questions and answers are users' questions and answers from several web forums related to Adobe Photoshop. This dataset provides question-answer pairs but not the corpus. So, we have built a corpus with all answer text in this dataset and built question-document pairs as TechQA; each question is annotated with 50 documents retrieved by BM25, and the 50 documents contain the ground truth document. Table 1 shows the detailed statistics of the PhotoshopQuiA dataset.

4.2 Baselines

There are two types of document retrievers: lexical retrievers and dense retrievers. We compare VKGDR with a lexical retriever and three dense retrievers.

Lexical Retriever: We use BM25 as the lexical retriever. BM25 has a better or similar performance than dense retrievers when training data is insufficient and the questions are domain-specific (Thakur et al., 2021). Thus, BM25 is a strong baseline in our problem setting.

Dense retriever: **DPR** is a dense retriever for open-domain QA (Karpukhin et al., 2020). We use

DPR trained on NaturalQuestions (Kwiatkowski et al., 2019), an open-domain QA dataset. Domain-adaptation (**Adapt**) is another approach for training dense retrievers in a zero-shot setting. We pre-train BERT-large (Devlin et al., 2019) on the corpus of each dataset and compare with VKGDR. We use a CLS vector of BERT-large for document representation. The performance of fully-supervised models provides an approximation of the performance of unsupervised models. We train a bi-encoder with the same supervision method used in DPR on the question-answer pairs of each dataset and compare this model ("**DPR***") with VKGDR. The encoder of DPR* is initialized with RoBERTa-large (Liu et al., 2019).

5 Results

In this section, we verify the efficacy of VKGDR with the experiments described in the previous section. The experimental results demonstrate three findings. First, VKGDR outperforms baselines in a zero-shot setting and a fully-supervised setting. Furthermore, VKGDR without fine-tuning outperforms a fully-supervised bi-encoder. Second, our distant-supervision method for the relation encoder outperforms the previous method. Third, the two main components of VKGDR, graph representation of a corpus and relation vectors, are essential to achieve the zero-shot performance of VKGDR. We describe details of the experimental results in the following sections.

5.1 Zero-Shot Domain-Specific Document Retrieval

Table 2 and Figure 4 show zero-shot domain-specific document retrieval performance of VKGDR and baselines. These experiments support the following findings: 1) constructing a VKG is more effective than transfer learning methods when training data is unavailable, and 2) our distant-supervision method for the relation encoder outperforms the previous method. We describe details of experimental results in the following paragraphs.

Efficacy of VKG in a Zero-Shot Setting: Table 2 shows the performance of three types of models. The first column indicates the type of each model. Type "L" represents lexical retrievers, type "D" represents dense retrievers, and type "D+L" represents ensemble models of type "D" and type "L." The ensemble models compute a similarity score of each document with the following formula:

$$\text{Score}(d_i) = -(\text{Rank}_{\text{Dense}}(d_i) + \lambda \cdot \text{Rank}_{\text{BM25}}(d_i)),$$

where $\text{Rank}_{\text{Dense}}(d_i)$ and $\text{Rank}_{\text{BM25}}(d_i)$ are ranks of document d_i predicted by a dense retriever and BM25. λ is a weight for BM25, and we set λ to 1.0. The column "S" in Table 2 ("S" stands for supervision) indicates whether each model is a zero-shot model or a fully-supervised model. "✗" represents that the model is an unsupervised model, and "✓" represents the model is trained on the question-document pairs of the TechQA train set.

Type	Model	S	TechQA R@1	TechQA R@5	TechQA MRR
L	BM25	✗	43.7	63.7	54.2
D	Adapt	✗	5.0	11.8	12.1
	DPR	✗	16.8	40.6	28.6
	VKGDR	✗	**39.3**	**63.7**	**50.2**
D+L	Adapt	✗	9.3	28.7	34.5
	DPR	✗	28.7	55.6	47.2
	VKGDR	✗	**44.3**	**68.7**	**55.8**
D	DPR*	✓	36.8	73.1	52.3

Table 2: The zero-shot domain-specific document retrieval performance of VKGDR and baselines on TechQA. In the first column, "L" represents that the model type is a lexical retriever. "D" represents dense retrievers. "D+L" is an ensemble model of a dense model and BM25. The "S" column indicates whether each model is trained on the question-document pairs of TechQA. The results show that VKGDR outperforms baselines of the same model type and even outperforms the fully-supervised model in R@1 and MRR.

The results of type "D" models show that VKG construction brings better retrieval performance than other approaches. We show that the domain-adaptation method (Adapt) significantly underperforms than VKGDR by 34.3%p in R@1. Training retrievers on data in another domain (DPR) results in 22.5%p lower performance than VKGDR in R@1. From these results, we show the efficacy of constructing a VKG.

From previous literature, we see that BM25 outperforms dense retrievers when insufficient question-document pairs are provided and when the questions are domain-specific (Ma et al., 2021; Thakur et al., 2021). The results of BM25 in Table 2 are aligned with previous research on document retrievers; BM25 outperforms dense retrievers ("D" models) in Table 2. We combine BM25

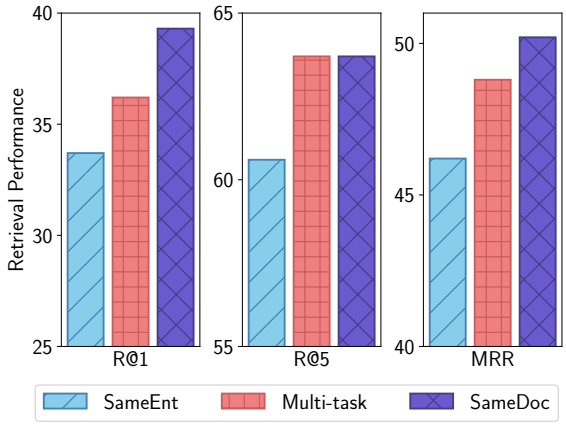

Figure 4: The zero-shot domain-specific document retrieval performance of VKGDR in three different relation encoder distant-supervision methods. SameEnt is the previous method that assumes two mention pairs have the same relation if the mention pairs share the same entity pair. SameDoc is our proposed supervision. Multi-task is multi-task learning of SameEnt and SameDoc. The results show the efficacy of our distant-supervision method.

Type	Model	S	PhotoshopQuiA R@1	R@5	MRR
L	BM25	✗	4.9	10.5	8.8
D	Adapt	✗	2.1	14.4	11.1
	DPR	✗	9.1	26.7	19.9
	VKGDR	✗	**22.5**	**45.0**	**33.3**
D+L	Adapt	✗	1.4	11.6	9.8
	DPR	✗	6.3	**16.9**	**14.1**
	VKGDR	✗	**8.8**	15.8	12.9
D	DPR*	✓	12.3	36.2	24.5

Table 3: The zero-shot domain-specific document retrieval performance of VKGDR and baselines on PhotoshopQuiA. This table shares the symbols used in the first column and the meaning of the "S" column with Table 2. The results show that VKGDR outperforms baselines in a zero-shot setting and even outperforms the fully-supervised model.

and dense retrievers (type "D+L") to improve the retrieval performance in a zero-shot setting. As a result, VKGDR achieves 5%p higher retrieval performance, 44.3 R@1, and this outperforms other baselines, including BM25.

VKG construction enables unsupervised retrievers to overcome a fully-supervised bi-encoder. We report the performance of a bi-encoder trained on question-document pairs of TechQA in Table 2 (DPR*). We see that only type "D" VKGDR outperforms DPR* in R@1 and MRR. These results support that the VKG is a key component in achieving a better performance than a fully-supervised model when training data is unavailable.

VKGDR outperforms previous approaches in building a VKG: Figure 4 shows the zero-shot document retrieval performance of three different VKG construction methods on TechQA. The three methods are "SameEnt", "SameDoc", and "Multi-task." The relation encoder in each method uses a different distant-supervision. "SameEnt" assumes that mention pairs sharing the same entity pair have the same relation (Sun et al., 2021). "SameDoc" is our distant-supervision method. We conduct multi-task learning of "SameEnt" and "SameDoc" ("Multi-task"). Multi-task learning combines multiple object functions and achieves better performance than the models trained by only one of the object functions.

Figure 4 shows that our distant-supervision, "SameDoc", outperforms the previous approach, "SameEnt". Also, we see that the performance of "SameEnt" increases when "SameEnt" is jointly trained with our distant-supervision. However, the performance of "SameDoc" decreases in this multi-task setting. This result indicates that the previous approach and our method are not complementary in the multi-task setting. From these results, we show that the context of mention pairs provides a better supervision signal than the textual form of mention pairs (entities of the mentions).

5.2 Zero-Shot Domain-Specific Answer Retrieval

Table 3 shows the zero-shot answer retrieval performance of VKGDR and baselines on PhotoshopQuiA. Type "D" retrievers show similar results as Table 2; VKGDR outperforms other type "D" baselines. VKGDR also outperforms the fully-supervised bi-encoder, DPR*. These results show that using a VKG brings better answer retrieval performance than the domain-adaptation method and the transfer learning method when training data is unavailable.

In Table 3, we show the performance of BM25 and dense retrievers ensembled with BM25. The lexical retriever underperforms dense retrievers on PhotoshopQuiA, whereas BM25 is a strong baseline on TechQA. Also, using lexical matching degenerates the overall retrieval performance of dense

Model	TechQA		
	R@1	R@5	MRR
DPR*	36.8	73.1	52.3
VKGDR	**48.7**	**76.8**	**60.1**

Model	PhotoshopQuiA		
	R@1	R@5	MRR
DPR*	12.3	36.2	24.5
VKGDR	**25.3**	**52.1**	**38.1**

Table 4: The document retrieval performance of VKGDR and DPR supervised on question-document pairs of TechQA and PhotoshopQuiA.

retrievers. This is because of the inconsistent use of terminologies between the corpus and the questions. The corpus of PhotoshopQuiA consists of answers written by users, not the official manual of the product, and this makes PhotoshopQuiA more difficult than TechQA.

5.3 Fully-Supervised Domain-Specific Document Retrievers

Fully-supervised VKGDR outperforms the fully-supervised bi-encoder (DPR*). Table 4 shows the retrieval performance of VKGDR trained on question-document pairs of TechQA and PhotoshopQuiA. We train the relation encoder with the following assumption: mentions pairs that appeared in the same question-document pair are in similar relation. The fully-supervised relation encoder is then used to compute the relation vectors of the VKG, and VKGDR uses the new VKG for document retrieval. The relation encoder trained on question-document pairs increase the retrieval performance of VKGDR; R@1 of VKGDR in Table 4 are 4.4%p and 2.8%p higher than the R@1 of VKGDR in Table 2 and 3. Also, fully-supervised VKGDR significantly outperforms DPR* by 11.9%p and 13.0%p in R@1 on TechQA and PhtoshopQuiA, respectively; we see the same pattern in other evaluation metrics.

5.4 Ablation Study

VKG consists of two components: graph representation of a corpus and relation vectors. In this section, we verify the importance of each module. Table 5 shows the performance of VKGDR on TechQA in three different settings: VKGDR, VKGDR without using the relation vectors (w/o relation embedding), and VKGDR without using the graph structure (w/o mention pairs). We describe each setting with the example in Figure 3. "w/o relation embedding" is a model that uses $\vec{1}$ (a vector that all elements are one) for all relation vectors in the VKG; all relation vectors in Figure 3 are replaced with $\vec{1}$. This is equivalent to using the number of overlapping mention pairs as the similarity between a question and a document. "w/o mention pairs" is a model without mention pair matching. For instance, all values in the similarity matrix (right part of Figure 3) are used to compute the question-document similarity. Table 5 shows that "w/o relation embedding" has better performance than "w/o mention pairs". This indicates that the graph structure is slightly more important than the relation embedding. However, the gap is not significant in R@1 and MRR. So, we see that both components are essential to achieve the document retrieval performance of VKGDR.

	R@1	R@5	MRR
VKGDR	39.3	63.7	50.2
- w/o relation embedding	32.5	59.3	44.9
- w/o mention pairs	31.8	53.7	42.9

Table 5: This table shows the performance of VKGDR in three different settings: without any modification on the VKG, using the VKG without relation vectors ("w/o relation embedding"), and using the VKG without the graph structure ("w/o mention pairs"). The results indicate that both components are essential to achieve the previous experimental results.

6 Conclusion

The main challenge in domain-specific document retrieval is the difficulty of specialized knowledge and terminologies appearing in the documents. In this study, we propose VKGDR to resolve this problem. VKGDR consists of two modules: 1) the model that represents a given corpus into a graph of mentions and their relations and 2) a document retriever based on the VKG. We showed that VKGDR outperforms previous retrievers in zero-shot domain-specific document retrieval. When insufficient training data is provided, unsupervised VKGDR shows even better results than a fully-supervised dense retriever. Also, we compared our VKG construction method with a previous method and showed that our method performs better on domain-specific documents.

References

Iz Beltagy, Kyle Lo, and Arman Cohan. 2019. SciBERT: A pretrained language model for scientific text. In *EMNLP-IJCNLP*.

Vittorio Castelli, Rishav Chakravarti, Saswati Dana, Anthony Ferritto, Radu Florian, Martin Franz, Dinesh Garg, Dinesh Khandelwal, J Scott McCarley, Michael McCawley, et al. 2020. The techqa dataset. In *ACL*.

Ilias Chalkidis, Manos Fergadiotis, Prodromos Malakasiotis, Nikolaos Aletras, and Ion Androutsopoulos. 2020. LEGAL-BERT: The muppets straight out of law school. In *Findings of the Association for Computational Linguistics: EMNLP*.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. Bert: Pre-training of deep bidirectional transformers for language understanding. In *NAACL-HLT*.

Bhuwan Dhingra, Manzil Zaheer, Vidhisha Balachandran, Graham Neubig, Ruslan Salakhutdinov, and William W Cohen. 2020. Differentiable reasoning over a virtual knowledge base. In *ICLR*.

Andrei Dulceanu, Thang Le Dinh, Walter Chang, Trung Bui, Doo Soon Kim, Manh Chien Vu, and Seokhwan Kim. 2018. Photoshopquia: A corpus of non-factoid questions and answers for why-question answering. In *LREC*.

Suchin Gururangan, Ana Marasović, Swabha Swayamdipta, Kyle Lo, Iz Beltagy, Doug Downey, and Noah A. Smith. 2020. Don't stop pretraining: Adapt language models to domains and tasks. In *ACL*.

Vladimir Karpukhin, Barlas Oguz, Sewon Min, Patrick Lewis, Ledell Wu, Sergey Edunov, Danqi Chen, and Wen-tau Yih. 2020. Dense passage retrieval for open-domain question answering. In *EMNLP*.

Tom Kwiatkowski, Jennimaria Palomaki, Olivia Redfield, Michael Collins, Ankur Parikh, Chris Alberti, Danielle Epstein, Illia Polosukhin, Jacob Devlin, Kenton Lee, et al. 2019. Natural questions: A benchmark for question answering research. *TACL*.

Davis Liang, Peng Xu, Siamak Shakeri, Cicero Nogueira dos Santos, Ramesh Nallapati, Zhiheng Huang, and Bing Xiang. 2020. Embedding-based zero-shot retrieval through query generation. *arXiv*.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Roberta: A robustly optimized bert pretraining approach. *arXiv*.

Ji Ma, Ivan Korotkov, Yinfei Yang, Keith Hall, and Ryan McDonald. 2021. Zero-shot neural passage retrieval via domain-targeted synthetic question generation. In *EACL*.

Macedo Maia, Siegfried Handschuh, André Freitas, Brian Davis, Ross McDermott, Manel Zarrouk, and Alexandra Balahur. 2018. Www'18 open challenge: financial opinion mining and question answering. In *Companion Proceedings of the The Web Conference 2018*.

Mike Mintz, Steven Bills, Rion Snow, and Dan Jurafsky. 2009. Distant supervision for relation extraction without labeled data. In *ACL*.

Abhilash Nandy, Soumya Sharma, Shubham Maddhashiya, Kapil Sachdeva, Pawan Goyal, and Niloy Ganguly. 2021. Question answering over electronic devices: A new benchmark dataset and a multi-task learning based qa framework. In *Findings of the Association for Computational Linguistics: EMNLP*.

Livio Baldini Soares, Nicholas FitzGerald, Jeffrey Ling, and Tom Kwiatkowski. 2019. Matching the blanks: Distributional similarity for relation learning. In *ACL*.

Haitian Sun, Bhuwan Dhingra, Manzil Zaheer, Kathryn Mazaitis, Ruslan Salakhutdinov, and William Cohen. 2018. Open domain question answering using early fusion of knowledge bases and text. In *EMNLP*.

Haitian Sun, Patrick Verga, Bhuwan Dhingra, Ruslan Salakhutdinov, and William W Cohen. 2021. Reasoning over virtual knowledge bases with open predicate relations. In *ICML*.

Alon Talmor and Jonathan Berant. 2018. The web as a knowledge-base for answering complex questions. In *NAACL-HLT*.

Nandan Thakur, Nils Reimers, Andreas Rücklé, Abhishek Srivastava, and Iryna Gurevych. 2021. BEIR: A heterogeneous benchmark for zero-shot evaluation of information retrieval models. In *NeurIPS Datasets and Benchmarks Track*.

Ellen Voorhees, Tasmeer Alam, Steven Bedrick, Dina Demner-Fushman, William R Hersh, Kyle Lo, Kirk Roberts, Ian Soboroff, and Lucy Lu Wang. 2021. Trec-covid: constructing a pandemic information retrieval test collection. In *SIGIR*.

Michihiro Yasunaga, Hongyu Ren, Antoine Bosselut, Percy Liang, and Jure Leskovec. 2021. Qa-gnn: Reasoning with language models and knowledge graphs for question answering. In *NAACL-HLT*.

Wenhao Yu, Lingfei Wu, Yu Deng, Ruchi Mahindru, Qingkai Zeng, Sinem Guven, and Meng Jiang. 2020. A technical question answering system with transfer learning. In *EMNLP: System Demonstrations*.

Wenhao Yu, Lingfei Wu, Yu Deng, Qingkai Zeng, Ruchi Mahindru, Sinem Guven, and Meng Jiang. 2021. Technical question answering across tasks and domains. In *NAACL-HLT: Industry Papers*.

Rong Zhang, Revanth Gangi Reddy, Md Arafat Sultan, Vittorio Castelli, Anthony Ferritto, Radu Florian, Efsun Sarioglu Kayi, Salim Roukos, Avirup Sil, and Todd Ward. 2020. Multi-stage pretraining for low-resource domain adaptation. In *EMNLP*.

Yuyu Zhang, Hanjun Dai, Zornitsa Kozareva, Alexander J Smola, and Le Song. 2018. Variational reasoning for question answering with knowledge graph. In *AAAI*.

A Appendices

A.1 Experimental Setup

Evaluation Metrics: Recall@k (R@k) and mean reciprocal rank (MRR) are evaluation metrics for document retrieval tasks. R@k measures the proportion of the model's predictions where top-k retrieved documents contain the ground truth document. MRR is defined with the predicted rank of the ground truth document as follows:

$$\text{MRR} = \frac{1}{n}\sum_{i=1}^{n}\frac{1}{r_i},$$

where n is the number of predictions, r_i is the predicted rank of the ground truth document of i'th query.

Hyper-parameter Settings: We use Adam optimizer with a warmup ratio of 0.1 and set the learning rate to 2×10^{-5} for VKGDR and baselines. We use the validation score to get the best checkpoint for all models. VKGDR's relation encoder is trained on the pre-trained BERT-large model. We train the relation encoder with a batch size of 128 for two epochs. The max length of the relation encoder is set to 128, and the number of negative samples in (1) is set to 2. We train RoBERTa (Bi-Encoder) with a batch sizes of 32 for twenty epochs and Adapt with a batch size of 80 for ten epochs. For both baselines, we set the max sequence length to 512. We use a machine with eight A100 GPUs. We report the result of a single trial.

A.2 License or Terms of Artifacts

We use BERT whose license is under the Apache License 2.0 free with modification and distribution. Also, we use RoBERTa whose license is under the GNU GENERAL PUBLIC LICENSE free with modification and distribution. All models we used are publicly available.

MICO: Selective Search with Mutual Information Co-training

Zhanyu Wang[*,†]
Purdue University
{wang4094}@purdue.edu

Xiao Zhang[†] and **Hyokun Yun** and **Choon Hui Teo** and **Trishul Chilimbi**
Amazon
{zhxao, yunhyoku, choonhui, trishulc}@amazon.com

Abstract

In contrast to traditional exhaustive search, selective search first clusters documents into several groups before all the documents are searched exhaustively by a query, to limit the search executed within one group or only a few groups. Selective search is designed to reduce the latency and computation in modern large-scale search systems. In this study, we propose MICO, a **M**utual **I**nformation **CO**-training framework for selective search with minimal supervision using the search logs. After training, MICO does not only cluster the documents, but also routes unseen queries to the relevant clusters for efficient retrieval. In our empirical experiments, MICO significantly improves the performance on multiple metrics of selective search and outperforms a number of existing competitive baselines.

1 Introduction

In information retrieval (IR), searching over all the documents is quite costly at a large scale (Risvik et al., 2013). Selective search (Kulkarni, 2013; Kulkarni and Callan, 2015) divides documents into a number of shards (clusters) to allow an incoming query to search over only a small number of these shards that are most relevant to this query. It tries to preserve the quality of search results similar to that of searching over the whole corpus, with less computation and shorter retrieval latency, achieved by limiting an incoming query to search within only a few chosen document shards. Sharding is the process of breaking a massive collection of documents into smaller chunks called shards. Randomly assigning documents to shards, a popular approach, cannot guarantee documents relevant to a specific query restrained within one shard or only a few shards. With the observation that semantically similar documents tend to appear together in the search results of the same query (van Rijsbergen, 1979), researchers in recent years have devised multiple machine learning-based algorithms for selective search. These algorithms are tailored to benefit the document retrieval process by dividing a large volume of the corpus in a certain way so that semantically related documents are allocated in the same cluster and then routing new queries to the most relevant clusters subsequently. In this way, the retrieval system can return the search results with both efficiency and accuracy, as shown in Figure 1. These approaches significantly differ from the traditional sharding methods that either randomly split the collection into several groups (Barroso et al., 2003) or allocate them based on simple handcrafted rules (Gravano et al., 1999), and have achieved comparable performance to exhaustive search over the entire document corpus.

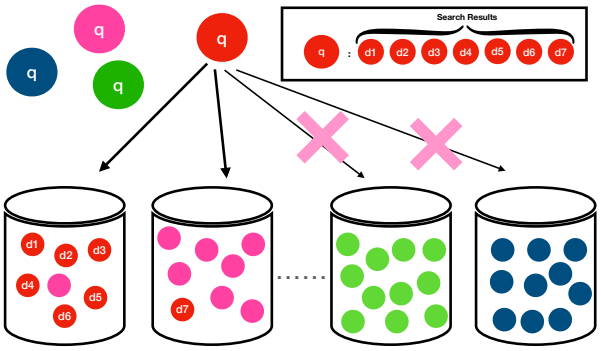

Figure 1: Machine learning based selective search first divides documents into different shards based on their semantic similarities and then routes queries to the shards containing documents most relevant to this query.

Nevertheless, having led giant strides in selective search, these approaches fail to address a few critical practical issues when scalable IR systems for a large number of documents in the industry are needed. A major concern is that these approaches heavily rely on a multi-stage process: firstly, complex clustering procedures are applied

[*] Work done while doing an internship at Amazon.
[†] Equal contribution.

Proceedings of the 29th International Conference on Computational Linguistics, pages 1179–1192
October 12–17, 2022.

to allocate documents into different shards, while later additional elaborate algorithms are used to choose a small number of shards for incoming queries. The multi-stage process needs not only additional infrastructure support to bridge the output of the previous step as the input of the next step but also is prone to error propagation. A natural attempt to mitigate the aforementioned difficulties is to construct a model for both document allocation and subsequent query routing, with an objective function to be optimized in an end-to-end paradigm. This approach ensures applying gradient-based learning to modules or components of the system coherently.

To this end, we introduce Mutual Information CO-training (MICO), a novel end-to-end approach for selective search. We argue that by maximizing the mutual information of the query routing variable and document assignment variable of a relevant query-document pair, we can leverage co-training in an end-to-end training fashion to overcome the aforementioned difficulties. This approach first creates a bipartite graph by exploiting the query-document relations and then treats query and document as two different views of the same example while maximizing the mutual information between the cluster variable of these two views. We show an overview of the MICO framework in Figure 2 and summarize our contributions:

1. We propose MICO, Mutual Information CO-training, a novel approach for selective search. MICO treats the query and the document as two different views of the same training example: the query-document pair in the log, maximizing the mutual information between the cluster assignment variables of each view. The cluster assignment variables are parameterized via two separate distributions conditioned on the query and the document.
2. We design MICO ready for practical use as it is being trained end-to-end for both document sharding and subsequent query routing. While the trained document allocation module assigns documents into different clusters, the trained query routing module routes new queries to the target cluster(s) to retrieve the most relevant documents at a low cost. To the best of our knowledge, MICO is the first attempt to deal with document sharding and query routing in selective search in an end-to-end fashion.
3. We show significantly improved performance on two IR data sets with MICO on multiple important metrics empirically in selective search. MICO beats competitive baselines regarding query coverage while minimizing search costs.

2 Related Works

There are quite a few studies on selective search on document sharding and query routing. Most of them aggregate similar documents into the same shards by measuring the semantic relevance among them in the collection to facilitate the subsequent search by limiting the query to be executed only within a few shards (Aly et al., 2013; Kulkarni, 2013; Kulkarni and Callan, 2010, 2015). Only a few studies on document sharding utilize the search log, where the information within the queries used to retrieve the documents are tailored to assist the clustering (Puppin et al., 2006; Poblete and Baeza-Yates, 2008; Dai et al., 2016). But they either merely extract simple features from the query to form rules or just employ the term frequency information in the queries as weights in the documents clustering process, neglecting both the strong connection between queries and documents and the semantic information hidden inside the query. Besides, all the previous studies rely on a multi-stage process for document clustering and the subsequent query routing, where an additional shard selection procedure has to be applied during the query routing stage (Si and Callan, 2003; Thomas and Shokouhi, 2009; Kulkarni et al., 2012; Aly et al., 2013). Besides, earlier studies primarily focus on ensuring semantically similar documents are allocated in the same cluster, neglecting the imbalance among the generated clusters, which weakens their application in practice. Furthermore, only a few studies take load-balancing into account when allocating documents (Kim et al., 2016; Dai et al., 2016). But these load balancing algorithms are rule-based procedures (by first ordering the shards by sizes and then regrouping them), in addition to the document sharding step.

Co-training (Blum and Mitchell, 1998; Ganchev et al., 2008) was initially proposed for semi-supervised training, where two classifiers are trained separately on two distinct views of the same input data and forced to have similar predictions. These two classifiers are designed to constrain each other to make coherent decisions, and the objective function penalizes the disagreement between them. Dasgupta et al. (2001) established the PAC generalization bounds for co-training for multiple classes

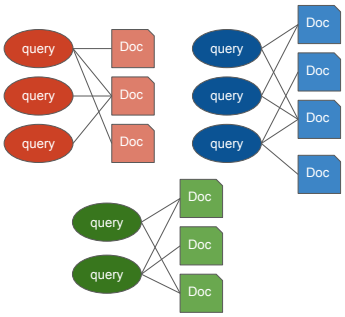

(a) Imaginary Document Clusters (with Queries)

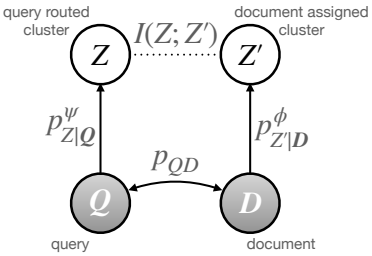

(b) Overview of Mutual Information CO-training (MICO)

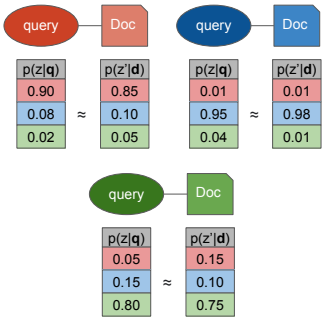

(c) Cluster selection coherency through maximizing the mutual information $I(Z;Z')$

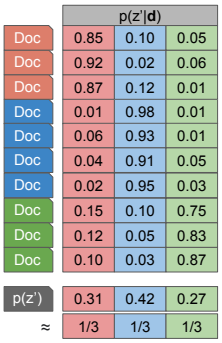

(d) Balance of shard size by maximizing the entropy $H(Z')$

Figure 2: Figure 2a on the left illustrates the query-document pairs with the clusters they belong to, denoted by red, blue, and green, respectively. Figure 2b on the right is an overview of our Mutual Information CO-training (MICO) approach: the query and the document are treated as two views of the same example, and two different distributions parameterize the cluster assignment variables of the two views respectively. In Figure 2c and Figure 2d, we illustrate cluster selection with an example where the predicted cluster index distribution is a vector for each query and the document.

classification, extending the theoretical bound on only two classes in the original study (Blum and Mitchell, 1998). However, as pointed out by Pierce and Cardie (2001), the success of co-training heavily relies on the high-quality labeled data. This is also true in an attempt to apply co-training for unsupervised learning (Collins and Singer, 1999), where sophisticated rules which are manually extracted are used as seeds for bootstrapping.

Information Maximization (IM) (Gomes et al., 2010; Bridle et al., 1992), promoting the idea that the output shall retain as much information as the input variable, has shed light on learning without direct supervision signals. In language processing, the celebrated brown clustering (Brown et al., 1992) maximizes the mutual information between random bigrams. The information bottleneck approach (Tishby et al., 2000) proposes an algorithm that discovers the representative coding of the input by capturing its relevant structure and proves its convergence. Gomes et al. (2010) observe the degeneration and class imbalance when using mutual information for clustering and propose adding a regularization term for remedy. Combined with the rejuvenated deep neural networks, IM has shown robust performance on clustering (Hu et al., 2017) with data augmentation.

More recent works have studied the application of information maximization in relational data. Co-training has been jointly used with information maximization in speech processing (McAllester, 2018). Stratos (2019) models the contextual information and the target class via mutual information maximization for word class induction, showing that the variational lower bound is more robust against the bias from noise. It is built upon the theoretical findings that the mutual information estimation through any high-probability lower bound has limitations that measuring and maximizing mutual information from finite data is a challenging training objective (McAllester and Stratos, 2020). This study is further extended in an adversarial training setup to circumvent computing the global marginal distribution (Stratos and Wiseman, 2020). Our work on selective search is inspired by these studies, treating the query and the document as two

views of one sample and respecting the resulting cluster size balance. It is trained in an end-to-end paradigm without direct supervision signals.

In the IR community, whether dense or sparse retrieval is better is still a prolonged debate (Lin, 2021; Luan et al., 2021), and recent studies have shown the improved performance of interpolating both (Li et al., 2022), implying designing an appropriate approach to combine them is a promising direction. We would like to note selective search is the strategy for distributing the information retrieval workload, and is complementary and agnostic to innovations in algorithms related to dense or sparse retrieval. In fact, our selective search approach can be used in conjunction with dense, sparse or combined retrieval.

3 MICO: Mutual Information CO-training

Our assumption is that for a relevant query-document pair in the search, the cluster index assignments z of the query $\boldsymbol{q}$ and z' of the document $\boldsymbol{d}$ are two views of that pair $(\boldsymbol{q}, \boldsymbol{d})$, thus z is equal to z'. MICO, consisting of a document allocation module and a query routing module, is trained based on this assumption. After training, MICO assigns the document to the cluster based on the prediction made by the document allocation module and routes new incoming queries to the cluster based on the prediction by the query routing module.

Our goal is to learn a model from the double-view examples in the search logs. The search logs are historical data that contains the relevance information between the query and its relevant documents, but does not carry explicit information of how documents shall be allocated into different shards, neither the query routing information of which shard the query shall be dispatched to. Thus, we have minimal supervision signal when training the model to learn to cluster documents and queries.

The desired model maximizes the document coverage (recall) and minimizes the search cost simultaneously. We force the cluster selection of the query and the relevant document to be coherent during training, formally as $p(z|\boldsymbol{q}) \approx p(z'|\boldsymbol{d})$. Here we force the two distributions to be consistent if the query and the document are relevant to reflect the coherent cluster assignment. Meanwhile, to avoid certain resulting clusters being oversized, causing the search in these shards to be inefficient, we consider balancing the document shard sizes. This concept is illustrated in Figure 3. This balance is achieved by assigning the documents to clusters as uniform as possible (formally $p(z') \approx \text{Uniform}$ where $p(z')$ is the distribution of cluster indices z'). MICO achieves cluster selection coherency between queries and documents by minimizing the mutual information between $p(z|\boldsymbol{q})$ and $p(z'|\boldsymbol{d})$, and balances shard size by adding an entropy regularization term $p(z')$.

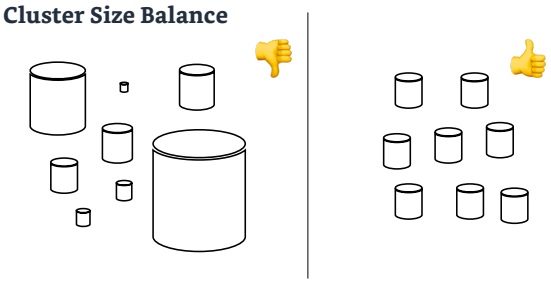

Figure 3: We use entropy regularization to ensure the desired cluster size balance, thus document shard size balance is achieved by maximizing the entropy $H(Z')$.

After training, the model allocates all the documents into shards; and for a new coming query, since we have cast query routing as a clustering problem, the query routing module generates a score for an incoming query during the inference time across all the shards, and will route the query to the shards that have the highest scores.

MICO consists of three main components: One component maps queries to clusters, denoted by the conditional distribution $p^{\psi}(z|\boldsymbol{q})$, parameterized by ψ; Another one projects documents to clusters, denoted by the conditional distribution $p^{\phi}(z'|\boldsymbol{d})$, parameterized by ϕ; The third one, denoted by $g^{\theta}(z')$ parameterized by θ, uses a variational distribution to approximate an intractable distribution $p^{\phi}(z')$. All these three components are modeled via neural networks to account for the complexity.

3.1 Tractable Loss of MICO

As the mutual information $I(Z; Z')$ of the query and the document cluster assignment and the entropy regularization $H(Z')$ of the document assignment are intractable (details are in the Appendix Section B.2), we approximate them using $H(Z; Z')$ and $H^{+}(Z')$ respectively, i.e., we define the loss

function of MICO as

$$\mathcal{L} = H(Z';Z) - \beta H^{+}(Z') \tag{1}$$

where

$$H(Z';Z) = \mathbb{E}_{(\boldsymbol{q},\boldsymbol{d})\sim p_{QD}}[H(p^{\phi}(z'|\boldsymbol{d}); p^{\psi}(z|\boldsymbol{q}))]$$
$$= \mathbb{E}_{(\boldsymbol{q},\boldsymbol{d})\sim p_{Q,D}}\left[\sum_{z=z'} -p^{\phi}(z'|\boldsymbol{d}) \log\left(p^{\psi}(z|\boldsymbol{q})\right)\right], \text{and}$$
$$H^{+}(Z') = \sum_{z'} -\mathbb{E}_{\boldsymbol{d}\sim p_D}[p^{\phi}(z'|\boldsymbol{d})] \log\left(g^{\theta}(z')\right),$$

with β as a tunable hyper-parameter.

The optimization objective is to find

$$\phi^*, \psi^* = \arg\min_{\phi,\psi}\left\{\max_{\theta}\mathcal{L}\right\}. \tag{2}$$

We solve this minimax optimization problem by interleaving updating (ϕ, ψ) and θ during training. This is similar to training generative adversarial networks (GAN) in deep learning or the Actor-Critic model in reinforcement learning.

In the rest of this section, we provide brief explanations of the loss above. We rewrite $\mathcal{L}$ as $-(I^{+}(Z;Z') + (\beta-1)H^{+}(Z'))$ where $I^{+}(Z;Z') = -H(Z';Z) + H^{+}(Z')$, $I^{+}(Z;Z')$ and $H^{+}(Z')$ are approximated upper bounds of $I(Z;Z')$ and $H(Z')$ respectively. The intractable term $H(Z')$ is approximated by $H^{+}(Z')$, by replacing $\log(p^{\phi}(z'))$ with $\log(g^{\theta}(z'))$ ($\log(p^{\phi}(z'))$ is intractable due to the nature it is evaluated on full dataset): we maximize $H^{+}(Z')$ w.r.t. θ to force $g^{\theta}(z')$ to be close to $p^{\phi}(z')$. Hence, minimizing $\mathcal{L}$ is equivalent to maximizing $I^{+}(Z;Z')$ and $H^{+}(Z')$ when $\beta > 1$. In contrast to the proposed loss, direct application of the cross-entropy $H(Z';Z)$, corresponding to $\beta = 0$, is inappropriate here, as it will result in cluster imbalance: the model tends to assign all the documents into one cluster and route all queries to this shard.

The choice of β in the loss $\mathcal{L}$ in Equation 1 controls the trade-off between the mutual information $I^{+}(Z;Z')$ and the entropy $H^{+}(Z')$, which in turn balances shard sizes while retaining the success of selective search. $H(Z')$ is maximized if and only if when $\mathbb{E}_{\boldsymbol{d}\sim p_D}[p^{\phi}(z'|\boldsymbol{d})]$ is a uniform distribution over Z', i.e., the cluster sizes are all equivalent. Larger β helps regularize our document allocation module to better balance the cluster sizes which leads to reduced search costs and latency.

3.2 MICO-q: MICO with Query Consistency

Multiple relevant documents can correspond to the same single query in the search log. We hypothesize that forcing the coherency of cluster assignment among these documents shall help the model yield a better document allocation module. Therefore, we use a cross-entropy H^{q} among those documents as an additional regularization term. With this intuition, we develop a variant of MICO termed MICO-q, whose loss function is

$$\mathcal{L}^{q} = -\beta H^{+}(Z') + \frac{1}{1+\gamma}(H(Z';Z) + \gamma H^{q}), \text{with}$$
$$H^{q} = \mathop{\mathbb{E}}_{\boldsymbol{q}\sim p_Q, \boldsymbol{d}_1, \boldsymbol{d}_2 \sim p_{D|Q}} H\left(p^{\phi}(z|\boldsymbol{d}_1); p^{\phi}(z|\boldsymbol{d}_2)\right),$$

where γ is another tunable hyper-parameter.

4 Experiments

4.1 Data sets

We evaluate the proposed framework on the following two data sets, representing two different domains: the E-commerce search logs (ECSL) data set and the Cross-Lingual Information Retrieval (CLIR) data set derived from Wikipedia (Sasaki et al., 2018). We process the data to ensure the data sets contain only distinct documents and divide the queries as non-overlapped training, development, and testing set, following standard IR research procedures. We show the detailed data statistics in Table 5 in Appendix.

ECSL E-commerce search logs (ECSL) data set is created by sampling queries and product description documents on an English commercial shopping website. The relevance of a document to a query is derived from user actions on the search results. We consider three types of user actions on documents with increasing levels of relevance to the query: *impression* (the document is among the search results), *click* (the document link in the search results is clicked), and *purchase* (the product of the document from the search result is purchased). The former relation always entails the latter. We randomly split the queries into training, validation, and testing sets with the ratio of 8:1:1.

CLIR In the CLIR set, queries are in English, extracted as the first sentences from English wiki pages, with the title words removed. Two different types of relevance exist: *DL* stands for directly relevant, meaning the documents are the foreign-language pages having an inter-language link to

the English pages; *PL* stands for partially relevant, denoting the documents having mutual links to and from the *DL* document. To provide a comprehensive study on the robustness of the proposed framework, we select the documents in two high-resource languages: French (*fr*), Italian (*it*) and in two low-resource languages: Tagalog (*ta*), Swahili (*sw*). Queries are randomly split into training, validation, and testing sets with the ratio of 3:1:1.

4.2 Experimental Setup

In this study, we preset the number of shards to be 64 for the ECSL data set and 10 for the CLIR data set, according to the number of documents. To make a fair comparison with other baselines, we use the TF-IDF feature with the most frequent words as the feature vector for both the queries and the documents.

We compare our proposed methods with a number of competitive baselines in selective search, including **Random** assignment of documents into clusters, **K-means**, a variant of K-means called **Balanced K-means** ensuring balanced cluster sizes, an information-maximization based clustering algorithm **IMSAT** (Hu et al., 2017), **KLD** sharding algorithm (Kulkarni and Callan, 2015) and **QKLD** sharding algorithm (Dai et al., 2016). Details of these baselines are in Appendix A.2.

In our experiments, we use the *impression* relationship between the training query and the documents for training in ECSL data. We use both the *DL* and *PL* relations for training in CLIR, as they are non-overlapped documents. The expectation over the whole training set is approximated by the average over one batch during the training, t. We also provide a detailed theoretical analysis of batch training in MICO in Appendix C. Due to limited spaces, more details of the experimental setup can be found in Appendix A. Note neither the queries nor the documents have cluster labels, and the clustering effect in selective search is evaluated by the retrieval performance explained next.

4.3 Query Coverage Analysis

We first evaluate the query coverage ("recall" in some literature) of our methods and compare it with baselines. Query coverage measures the percentage of documents retrieved in the top few shards over all the documents retrieved in the exhaustive search.

Formally, the query coverage of a single query q in the top N shards is defined as $Cov_N(q) = \left(\sum_{i=1}^{N} R^q_{s_i^q}\right) / R^q$, where R^q is the total number of relevant documents in the corpus per the query q, and $R^q_{s_i^q}$ is the number of relevant document per the query q in the i-th selected shard s_i^q per query q. Consequently, the average coverage over the query set is $Cov_N = \left(\sum_{q \in Q} Cov_N(q)\right) / |Q|$, where Q is the testing queries set.

Our implemented coverage metric is even more strict than the coverage metric in Dai et al. (2016)'s study. Instead of sorting the shards by the number of retrieved documents they contain, we rigorously sort the chosen shards of a testing query based on the prediction score generated by the query routing module, which is more intrinsic to the nature of selective search.

We show our experimental results on ECSL in Table 1, with the mean value and the standard deviation in the parenthesis over five runs on three different relevances. We limit each testing query being routed to the most relevant **one** shard and **ten** shards based on the prediction. Our MICO and its variant MICO-q beat all the baselines, except *impression* with one shard only. By only searching within the ten most relevant shards, which is 15.6% of all the shards, MICO-q can achieve almost 95% coverage on *impression*, even comparable to the exhaustive search. MICO-q performs slightly more robust than MICO when searching over more shards, which might be attributed to its additional regularization term.

We also present the evaluation results on CLIR in Table 2, where we only probe the most relevant shard for each query. Due to the queries and documents being of different languages, we build vocabularies for the queries and documents separately, denoted by *sv*, to generate input to the document sharding module and the query routing module. MICO (and MICO-q) beats all the baselines (except *DL* in *ta*). We also notice the performance of QKLD dramatically drops on the CLIR data set. We attribute this failure to its usage of the query information to build document shards inappropriately because queries and documents are from different languages. QKLD and MICO (including MICO-q) are the only two models to utilize the query information specifically for document sharding. Based on the observation, we argue inappropriate use of the query information will deteriorate document sharding in selectively search. MICO, on the contrary, is able to leverage the queries for better document sharding, showing its robustness

in the cross-lingual scenario.

Though MICO beats all the baselines, if the top selected shards are extremely large, this approach is inapplicable in practice, as the actual search cost will be similar to the exhaustive search. In addition, MICO fails to beat KLD and QKLD in some exceptions, potentially ascribed to the most relevant shard created by them being larger with more documents. Thus, we analyze the search cost next.

4.4 Cost Analysis

To measure the search efficiency, we use the metrics **search resource cost** and **search latency cost** (the lower, the better) introduced in the early study (Kulkarni et al., 2012) to evaluate our models and the competitors. Search cost depends highly on the size of the shards we send the query to. Search resource cost calculates the resource usage as the upper bound on the number of document evaluated for each query, defined as $C_N^{res} = \sum_{i=1}^{N} |s_i^q|$, where $|s_i^q|$ is the number of documents in the i-th relevant cluster to query q. Similarly, search latency cost counts the number of evaluated documents on the longest execution path for query q for its top N selected shards: $C_N^{lat} = \max_{1 \le i \le N} |s_i^q|$.[1]

We show our evaluation results on the ECSL data set and the CLIR data set in Table 3, by restricting the search within the **five** most relevant clusters for ECSL, and the **two** most relevant for CLIR. Please note in these evaluations **Random** is the skyline, as all the documents are evenly distributed to N shards such that the shard sizes are almost equal. MICO demonstrates its supreme performance, even beating the skyline in some cases, showing its ability to control shard sizes while ensuring semantically similar documents are allocated in the same shard. MICO-q also achieves comparable performance with MICO. We interpret it as MICO and MICO-q are able to trade off document relevance in the same shard and balance of shard sizes on a sweet point. We also notice the high cost yielded by KLD and QKLD, which compromises their use in practice.

4.5 Balance Among Shard Sizes

We further investigate the performance of MICO and MICO-q via visualizing the trade-off between *query coverage* and *search resource cost* and the balance of resulting shard sizes on ECLS. Figure 4a shows that MICO and MICO-q always perform the best in terms of query coverage, at different levels of search resource cost. Figure 4b indicates that IMSAT is able to create well balanced shards, and MICO and MICO-q are on a par with it. To trade off query coverage and shard size balance, MICO and MICO-q are the best among all the competitors, gaining the favor in practical usage. Note that KLD and QKLD create very unbalanced shards, which explains MICO and MICO-q fail to beat them in some cases on query coverage analysis earlier: most of the documents are allocated into the same shard.

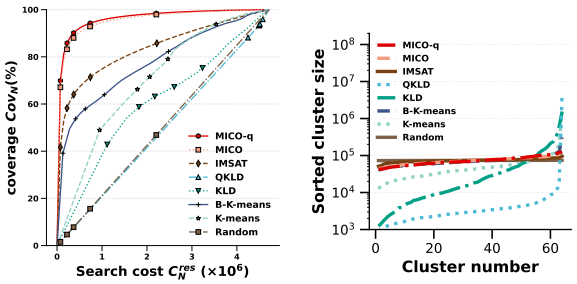

(a) Coverage v.s. Cost (b) Shard size distribution

Figure 4: Figure 4a shows MICO and MICO-q are significantly better than all other methods as they have high *impression* coverage with low search cost. From bottom-left to top-right, the markers on each line represent query coverage limited within the top-1, top-3, top-5, top-10, and top-30 clusters selectively. Figure 4b shows **Random** generates the most balanced shard sizes (as a flat line), and IMSAT also creates very balanced shards. MICO and MICO-q are on a par with IMSAT. In contrast, QKLD and KLD yield very unbalanced shards.

4.6 Effect of Entropy Regularization in MICO and MICO-q

We also examine the effect of entropy regularization by adjusting its strength in MICO and MICO-q by tuning their hyper-parameter β on ECSL. From Figure 5a, we can see that β does not affect the cost-coverage curve in a significant way. From Figure 5b, setting β larger than 1 yields much more balanced cluster sizes which means that the entropy regularization is necessary to balance cluster sizes. We conjecture this difference results from the degree of homogeneity of the documents the model allocates to different shards.

[1] Note the metrics in Kulkarni et al. (2012) introduce an additional term for the resource selection step. Since MICO doesn't require this additional step, that term is always evaluated to 0 with MICO (and MICO-q).

	impression		*click*		*purchase*	
Models	N=1	N=10	N=1	N=10	N=1	N=10
Random	1.56 (6e-3)	15.62 (0.02)	1.49 (0.08)	15.32 (0.85)	1.45 (0.24)	14.54 (0.27)
K-means	48.98 (1.60)	79.05 (0.51)	51.90 (1.56)	81.57 (4.0)	54.49 (1.97)	83.58 (1.49)
B-K-means	39.72 (1.12)	64.56 (1.30)	43.89 (2.03)	64.25 (1.78)	49.02 (2.37)	69.59 (1.22)
IMSAT	41.68 (0.55)	71.37 (0.28)	47.48 (1.62)	79.12 (2.94)	52.41 (0.42)	79.83 (1.06)
KLD	43.46 (5.91)	69.87 (5.34)	44.94 (8.04)	71.17 (5.55)	46.77 (9.32)	70.5 (4.08)
QKLD	**86.14** (8.85)	93.96 (0.77)	73.72 (7.25)	81.89 (1.2)	75.79 (7.22)	83.56 (1.57)
MICO	67.09 (0.20)	92.85 (0.12)	**82.85** (1.51)	97.81 (0.19)	**81.21** (0.49)	96.61 (0.14)
MICO-q	69.81 (0.34)	**94.28** (0.09)	82.48 (1.91)	**98.26** (0.20)	81.15 (1.23)	**97.25** (0.16)

Table 1: This table shows the performance of query coverage (recall) of MICO, MICO-q, and different baselines over three different query-document relationships on the ECSL data set. We show the performance by only probing the top-1 most relevant shard and the top-10 most relevant shards given a query. The number in the parenthesis right next to the coverage is the standard deviation over five runs. We observe other than the impression relation in which QKLD has the best performance, MICO or MICO-q beat all the baselines.

	fr		*it*		*ta*		*sw*	
Models	*DL*	*PL*	*DL*	*PL*	*DL*	*PL*	*DL*	*PL*
Random	10.02 (0.07)	9.72 (0.16)	10.02 (0.09)	10.0 (0.35)	9.88 (0.93)	9.86 (0.48)	10.01 (0.23)	10.0 (0.67)
K-means	12.19 (1.99)	10.79 (2.04)	14.91 (2.46)	16.36 (3.55)	16.25 (2.5)	21.08 (3.46)	21.71 (4.84)	18.55 (3.65)
B-K-means	12.2 (1.82)	11.44 (1.32)	12.45 (3.13)	12.46 (4.59)	12.78 (2.85)	11.23 (3.84)	11.71 (1.29)	12.16 (1.21)
IMSAT	19.77 (9.53)	19.84 (9.68)	40.09 (8.91)	40.13 (8.88)	12.72 (4.22)	11.89 (3.62)	8.4 (2.24)	8.6 (2.63)
KLD	38.6 (6.02)	40.65 (7.58)	60.94 (5.25)	61.83 (3.12)	**66.53** (8.43)	59.77 (7.18)	21.11 (3.52)	24.83 (3.81)
QKLD	17.76 (3.63)	18.82 (2.08)	18.9 (4.91)	17.45 (4.12)	23.65 (6.81)	24.4 (5.54)	12.23 (0.75)	16.45 (2.25)
MICO (sv)	44.93 (3.47)	**53.12** (2.17)	58.08 (1.22)	65.83 (1.06)	63.55 (4.45)	60.94 (4.92)	26.0 (3.51)	**28.67** (3.73)
MICO-q (sv)	**47.9** (2.68)	48.04 (3.44)	**75.27** (3.6)	**75.01** (4.39)	63.91 (5.3)	**61.29** (5.31)	**27.42** (3.37)	28.14 (2.54)

Table 2: This table shows the performance of query coverage of MICO, MICO-q, and different baselines on two different query-document relationships on the CLIR data set by only probing the most relevant shard given a query because we only divide the documents into ten shards. The number in the parenthesis right next to the coverage is the standard deviation over multiple runs. *sv* stands for separate vocabularies for the queries and documents, as in cross-lingual retrieval, the source language and the target language have different vocabularies, and separate vocabularies perform better than unified ones empirically. MICO and MICO-q beat all the baselines except *DL* in *ta*.

4.7 Ablation Study

We further investigate the effectiveness of different neural architecture variants of MICO. Instead of parameterizing the document allocation module and the query routing module differently by two separate neural networks ($p_q^{\psi}(z|\boldsymbol{q}), p_d^{\phi}(z'|\boldsymbol{d})$), we can parameterize both modules with the same neural network ($p_q^{\psi}(z|\boldsymbol{q}), p_d^{\psi}(z'|\boldsymbol{d})$), avoiding the risk of over-parameterization. We denote this variant as **-Par**. The recent emergence of the pre-trained language models such as BERT (Devlin et al., 2019) also provides a richer representation with contextual information. We use the BERT multilingual base model[2] to produce the text-level representations and feed them into MICO, either as fixed (denoted as **+BERT.fx**) or by fine-tuning the underlying BERT model (denoted as **+BERT.ft**). We show the query coverage (QC) and Search Resource Cost of these neural variants in Table 4 on ECSL, by probing the most relevant shard for each query. The results show that unifying the query module and the document module parameterization deteriorates the performance. We also notice that fixed BERT representations hurt the model performance significantly, while appropriate fine-tuning help improve the performance.

[2]https://huggingface.co/bert-base-multilingual-cased

	C_N^{res}					C_N^{lat}				
Models	ECSL	C-fr	C-it	C-ta	C-sw	ECSL	C-fr	C-it	C-ta	C-sw
K-means	2.061	14.12	11.54	1.6	1.42	1.572	6.44	5.95	0.95	1.27
Balaced K-means	0.620	8.1	6.83	0.99	0.87	0.277	2.58	2.02	0.34	0.67
IMSAT	0.370	9.57	5.89	0.93	0.61	0.082	3.57	4.78	0.58	0.53
KLD	2.17	17.48	13.15	1.93	1.09	1.41	13.26	11.43	1.72	0.74
QKLD	4.5	8.84	7.42	1.34	0.99	4.47	3.72	3.07	0.94	0.66
MICO	**0.367**	**6.19**	**5.13**	0.85	0.93	0.089	**2.34**	**1.89**	0.5	0.5
MICO-q	0.369	7.12	6.71	0.94	1.07	0.093	2.73	2.47	0.51	0.58
Random	0.368	7.20	5.95	**0.8**	**0.73**	**0.074**	2.41	1.99	**0.27**	**0.25**

Table 3: This table shows the performance of different models on the *Search Resource Cost* and the *Search Latency Cost* metrics, representing the search efficiency, with the lower the number, the better the performance. The results shown in this table are scaled by being divided by 10^6 on the ECSL data set and by 10^4 on the CLIR data set. Note in this set of experiments, we use separate vocabulary (*sv*) for MICO and MICO-q on CLIR. We observe the supreme performance of MICO, which in some cases even beats the **Random** skyline.

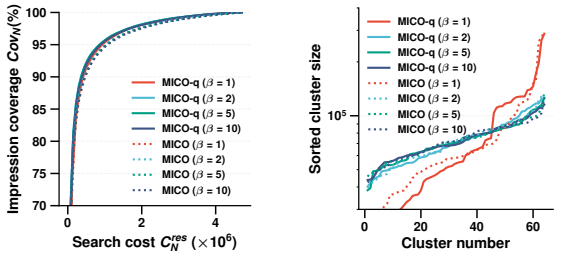

(a) Coverage v.s. Cost (b) Shard size distribution

Figure 5: This figure shows the results of MICO and MICO-q with different entropy parameter $\beta \in \{1, 2, 5, 10\}$. (a) The overall performance is hardly affected by β. (b) While larger β induces better balance on the cluster sizes, the difference between $\beta = 5$ and $\beta = 10$ is very small. When setting $\beta = 1$, the objective degenerates to use mutual information only (without the additional entropy regularization term) and the resulting shard sizes are significantly more unbalanced than with the other choices of β.

Model	QC	$C_{N=1}^{res}$
MICO (-Par)	66.08	0.365
MICO	67.09	0.367
MICO (+BERT.fx)	41.67	0.367
MICO (+BERT.ft)	**76.41**	0.375

Table 4: This table shows MICO with neural architecture variants. BERT with fine-tuning achieves better performance than the original MICO, while the other variants yield deteriorated performance. The search cost is slightly higher with the best-performing system. We attribute that the refined representations cause the model to weigh more on semantic similarity than cluster balance.

5 Conclusion

In this study, we present MICO, a Mutual Information CO-training framework for document sharding and query routing in selective search with minimal supervision. Our contributions are: First, during training, MICO maximizes the mutual information between the cluster assignment variables of the query and the document, forcing the prediction of two views of the query-document pair coherently. Second, this design enables it to be trained end-to-end for both document sharding (clustering) and subsequent query routing, featuring its practical use when vast volumes of documents are required to be searched simultaneously. Third, we show improved performance of MICO empirically on multiple important metrics in selective search.

Our future research direction includes further investigation of the regularization for more balanced sizes among all the shards, the utilization of complex neural language models to enrich the document and the query representation for higher query coverage via better clustering results, the reduction of potential noise between a query and its associated document in the data set, and the detection policy to decide when the model needs to be retrained if constant update of the model is needed when deployed. We are also interested in potential extension of this approach to multi-modal data, e.g., image search, music search and software search where the non-traditional documents are in the non-textual formats.

References

Robin Aly, Djoerd Hiemstra, and Thomas Demeester. 2013. Taily: shard selection using the tail of score distributions. In *The 36th International ACM SIGIR conference on research and development in Information Retrieval, SIGIR '13, Dublin, Ireland - July 28 - August 01, 2013*, pages 673–682. ACM.

Luiz André Barroso, Jeffrey Dean, and Urs Hölzle. 2003. Web search for a planet: The google cluster architecture. *IEEE micro*, 23(2):22–28.

Avrim Blum and Tom Mitchell. 1998. Combining labeled and unlabeled data with co-training. In *Proceedings of the eleventh annual conference on Computational learning theory*, pages 92–100.

John Bridle, Anthony Heading, and David MacKay. 1992. Unsupervised classifiers, mutual information and 'phantom targets. In *Advances in Neural Information Processing Systems*, volume 4. Morgan-Kaufmann.

Peter F. Brown, Peter V. deSouza, Robert L. Mercer, Vincent J. Della Pietra, and Jenifer C. Lai. 1992. Class-based n -gram models of natural language. *Computational Linguistics*, 18(4):467–479.

Michael Collins and Yoram Singer. 1999. Unsupervised Models for Named Entity Classification. In *1999 Joint SIGDAT Conference on Empirical Methods in Natural Language Processing and Very Large Corpora*.

Zhuyun Dai, Chenyan Xiong, and Jamie Callan. 2016. Query-biased partitioning for selective search. In *Proceedings of the 25th ACM International Conference on Information and Knowledge Management, CIKM 2016, Indianapolis, IN, USA, October 24-28, 2016*, pages 1119–1128. ACM.

Sanjoy Dasgupta, Michael L. Littman, and David A. McAllester. 2001. PAC generalization bounds for co-training. In *Advances in Neural Information Processing Systems 14 [Neural Information Processing Systems: Natural and Synthetic, NIPS 2001, December 3-8, 2001, Vancouver, British Columbia, Canada]*, pages 375–382. MIT Press.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, Minneapolis, Minnesota. Association for Computational Linguistics.

Kuzman Ganchev, João V. Graça, John Blitzer, and Ben Taskar. 2008. Multi-view learning over structured and non-identical outputs. In *Proceedings of the Twenty-Fourth Conference on Uncertainty in Artificial Intelligence*, UAI'08, page 204–211, Arlington, Virginia, USA. AUAI Press.

Ryan Gomes, Andreas Krause, and Pietro Perona. 2010. Discriminative clustering by regularized information maximization. In *Advances in Neural Information Processing Systems 23: 24th Annual Conference on Neural Information Processing Systems 2010. Proceedings of a meeting held 6-9 December 2010, Vancouver, British Columbia, Canada*, pages 775–783. Curran Associates, Inc.

Luis Gravano, Héctor García-Molina, and Anthony Tomasic. 1999. Gloss: Text-source discovery over the internet. *ACM Trans. Database Syst.*, 24(2):229–264.

Weihua Hu, Takeru Miyato, Seiya Tokui, Eiichi Matsumoto, and Masashi Sugiyama. 2017. Learning discrete representations via information maximizing self-augmented training. In *Proceedings of the 34th International Conference on Machine Learning, ICML 2017, Sydney, NSW, Australia, 6-11 August 2017*, volume 70 of *Proceedings of Machine Learning Research*, pages 1558–1567. PMLR.

Yubin Kim, Jamie Callan, J. Shane Culpepper, and Alistair Moffat. 2016. Load-balancing in distributed selective search. In *Proceedings of the 39th International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '16, page 905–908, New York, NY, USA. Association for Computing Machinery.

Anagha Kulkarni. 2013. *Efficient and Effective Large-scale Search*. PhD thesis, Carnegie Mellon University.

Anagha Kulkarni and Jamie Callan. 2010. Document allocation policies for selective searching of distributed indexes. In *Proceedings of the 19th ACM International Conference on Information and Knowledge Management*, CIKM '10, page 449–458, New York, NY, USA. Association for Computing Machinery.

Anagha Kulkarni and Jamie Callan. 2015. Selective Search: Efficient and Effective Search of Large Textual Collections. *ACM Trans. Inf. Syst.*, 33(4). Place: New York, NY, USA Publisher: Association for Computing Machinery.

Anagha Kulkarni, Almer S. Tigelaar, Djoerd Hiemstra, and Jamie Callan. 2012. Shard ranking and cutoff estimation for topically partitioned collections. In *21st ACM International Conference on Information and Knowledge Management, CIKM'12, Maui, HI, USA, October 29 - November 02, 2012*, pages 555–564. ACM.

Hang Li, Shuai Wang, Shengyao Zhuang, Ahmed Mourad, Xueguang Ma, Jimmy Lin, and Guido Zuccon. 2022. To interpolate or not to interpolate: Prf, dense and sparse retrievers. In *Proceedings of the 45th International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '22, page 2495–2500, New York, NY, USA. Association for Computing Machinery.

Jimmy Lin. 2021. Pyserini: A Python Toolkit for Reproducible Information Retrieval Research with Sparse and Dense Representations. In *Proc. of SIGIR*, page 7.

Yi Luan, Jacob Eisenstein, Kristina Toutanova, and Michael Collins. 2021. Sparse, Dense, and Attentional Representations for Text Retrieval. *Transactions of the Association for Computational Linguistics*, 9:329–345.

David McAllester. 2018. Information Theoretic Co-Training. Eprint: 1802.07572.

David McAllester and Karl Stratos. 2020. Formal limitations on the measurement of mutual information. In *The 23rd International Conference on Artificial Intelligence and Statistics, AISTATS 2020, 26-28 August 2020, Online [Palermo, Sicily, Italy]*, volume 108 of *Proceedings of Machine Learning Research*, pages 875–884. PMLR.

David Pierce and Claire Cardie. 2001. Limitations of co-training for natural language learning from large datasets. In *Proceedings of the 2001 Conference on Empirical Methods in Natural Language Processing*.

Barbara Poblete and Ricardo Baeza-Yates. 2008. Query-sets: Using implicit feedback and query patterns to organize web documents. In *Proceedings of the 17th International Conference on World Wide Web*, WWW '08, page 41–50, New York, NY, USA. Association for Computing Machinery.

Diego Puppin, Fabrizio Silvestri, and Domenico Laforenza. 2006. Query-driven document partitioning and collection selection. In *Proceedings of the 1st International Conference on Scalable Information Systems*, InfoScale '06, page 34–es, New York, NY, USA. Association for Computing Machinery.

Knut Magne Risvik, Trishul Chilimbi, Henry Tan, Karthik Kalyanaraman, and Chris Anderson. 2013. Maguro, a system for indexing and searching over very large text collections. In *Proceedings of the sixth ACM international conference on Web search and data mining*, pages 727–736. ACM.

Shota Sasaki, Shuo Sun, Shigehiko Schamoni, Kevin Duh, and Kentaro Inui. 2018. Cross-lingual learning-to-rank with shared representations. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 2 (Short Papers)*, pages 458–463, New Orleans, Louisiana. Association for Computational Linguistics.

Luo Si and Jamie Callan. 2003. Relevant document distribution estimation method for resource selection. In *Proceedings of the 26th Annual International ACM SIGIR Conference on Research and Development in Informaion Retrieval*, SIGIR '03, page 298–305, New York, NY, USA. Association for Computing Machinery.

Karl Stratos. 2019. Mutual information maximization for simple and accurate part-of-speech induction. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 1095–1104, Minneapolis, Minnesota. Association for Computational Linguistics.

Karl Stratos and Sam Wiseman. 2020. Learning discrete structured representations by adversarially maximizing mutual information. In *Proceedings of the 37th International Conference on Machine Learning, ICML 2020, 13-18 July 2020, Virtual Event*, volume 119 of *Proceedings of Machine Learning Research*, pages 9144–9154. PMLR.

Paul Thomas and Milad Shokouhi. 2009. Sushi: Scoring scaled samples for server selection. In *Proceedings of the 32nd International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '09, page 419–426, New York, NY, USA. Association for Computing Machinery.

Naftali Tishby, Fernando C. N. Pereira, and William Bialek. 2000. The information bottleneck method. *Proc. 37th Annual Allerton Conference on Communications, Control and Computing, 1999*, pages 368–377.

C. J. van Rijsbergen. 1979. *Information Retrieval*, 2ª edition. Butterworth-Heinemann, London.

APPENDIX

A Experimental Setup

A.1 Data Processing

We use the `gensim` package to preprocess all the queries and documents into TF-IDF features in three steps: 1. Tokenize the sentences into words and remove stop words; 2. Create a dictionary with most frequent words (20,000 for ECSL, and 3,000 for CLIR); 3. Transform the queries and documents into TF-IDF vectors.

Data sets	Queries				Documents	
	#Train	#Dev	#Test	Avg($\|L\|$)	#Doc	Avg($\|L\|$)
ECSL	339k	38k	55K	4	4.7M	16
CLIR-FR	14.8K	4.9K	4.9K	21	240K	177
CLIR-IT	14.8K	4.9K	4.9K	21	198K	169
CLIR-SW	12.8K	4.1K	4.1K	18	24k	82
CLIR-TA	14.3K	4.7K	4.7K	19	27k	79

Table 5: This table shows the statistics of the ECSL data set and the CLIR data set, including the numbers of queries, documents and their average length.

A.2 Baselines

Balanced K-means is built on top of K-means with approximated Hungarian algorithm to ensure the formed cluster sizes are well balanced. In K-means and Balanced K-means, documents and training queries are used together to form clusters, and we select the nearest clusters to the query vector as the top relevant clusters to route the query to. In IMSAT, the same model is used for both document assignment and query routing, after being trained on both queries and documents. KLD (Kullback-Liebler Divergence) only uses document information to form document clusters while QKLD (Query-based Kullback-Liebler Divergence) leverages the query information to form clusters. In KLD and QKLD, an additional shard selection algorithm is used to choose the relevant shards for new incoming queries.

A.3 MICO

During training, we set the batch size as 256. We optimize the objective using `Adam` with learning rate 0.03 for ϕ and ψ, and with learning rate 0.1 for θ. We model both $p_q^{\psi}(z|\boldsymbol{q})$ and $p_d^{\phi}(z|\boldsymbol{d})$ with neural networks with one hidden layer (dimension 20) and one softmax layer. We set the hyper-parameter for entropy regularization as $\beta = 10$. To stablize the training process, we set gradient clipping as 10.0.

A.4 MICO-q

We set the hyper-parameters the same as in MICO except: 1. We set the gradient clipping as 1.0; 2. We update θ for 4 steps instead of 1; 3. We set the entropy regularization strength as $\beta = 3$; 4. We set the query consistency parameter as $\gamma = 3$.

A.5 Computing Environment

We use an AWS EC2 instance (p3.2xlarge) consisting of 8 vCPUs (Intel Xeon E5-2686 v4) with 64GB memory, with one NVIDIA GPU card (Tesla V100-SXM2-16GB) for running our experiments. The computing environment is with Python 3.8, PyTorch 1.5, CUDA 9.2.

B Details of MICO

B.1 Preliminary

We introduce the notations and the basic concept of mutual information briefly here. We denote a random variable X's probability function by $p_X(x)$. We shorten it to $p(x)$ or p_X when its meaning is clear under its context. We use $p^{\theta}(x)$ when $p(x)$ is in a parametric distribution family and θ is the parameter. $p_{XY}(x, y)$ is the joint distribution of two random variables X and Y, and $p_{X|Y}(x|y)$ is the conditional distribution of X given Y, while $p_Y(y) = \sum_x p_{XY}(x, y)$ is the marginal distribution of Y under $p_{XY}(x, y)$. The expectation of a function $f(x)$ w.r.t. a random variable x is defined by $\mathbb{E}_{x\sim p_X}[f(x)] = \sum_{x\in\mathcal{X}} f(x)p_X(x)$. We denote the entropy of X by $H(X) = H(p_X) = \mathbb{E}_{x\sim p_X}[-\log(p_X(x))]$, while the conditional entropy is $H(Y|X) = \mathbb{E}_{x\sim p_X} H(p_{Y|X}) = \mathbb{E}_{(x,y)\sim p_{XY}}[-\log(p_{Y|X})]$. The cross-entropy between X and X' is defined as $H(X; X') = \mathbb{E}_{x\sim p_X}[-\log(p_{X'}(x))]$. We denote the mutual information between X and Y by $I(X; Y) = \sum_x \sum_y p_{XY} \log\left(\frac{p_{XY}}{p_X p_Y}\right) = H(X) - H(X|Y)$. It is minimized at 0 when X and Y are independent, i.e., $p_{XY} = p_X p_Y$. The chain rule for mutual information is $I(X; (Y, Z)) = I(X; Z) + I(X; Y|Z)$ where $I(X; Y|Z) = H(X|Z) - H(X|Y, Z) \geq 0$.

B.2 Information Maximization Approximation

We explain the reason of having the three components for calculating the mutual information. We seek to train the probabilistic modules $p^{\psi}(z|\boldsymbol{q})$ and $p^{\phi}(z'|\boldsymbol{d})$ through maximizing the mutual information between Z and Z' over our training data set, as

shown in Figure 2b. From this probabilistic graph, we know Z and Z' are independent when given $\boldsymbol{Q}$, i.e., $I(Z';Z|\boldsymbol{Q}) = 0$. Based on the chain rule of mutual information, we also have $I(Z';(\boldsymbol{Q},Z)) = I(Z';\boldsymbol{Q}) + I(Z';Z|\boldsymbol{Q}) = I(Z';Z) + I(Z';\boldsymbol{Q}|Z)$. Therefore, $I(Z';Z) = I(Z';\boldsymbol{Q}) - I(Z';\boldsymbol{Q}|Z)$. Since $I(Z';Z)$ is intractable under large amount of data and $I(Z';\boldsymbol{Q}|Z) \geq 0$, we maximize $I(Z';\boldsymbol{Q})$ instead, which is an upper bound of $I(Z';Z)$. With the fact $I(Z';\boldsymbol{Q}) = H(Z') - H(Z'|\boldsymbol{Q})$, we need the marginal distribution $p^\phi(z') = \mathbb{E}_{\boldsymbol{d}\sim p_D}[p^\phi(z'|\boldsymbol{d})]$ and the conditional distribution $p^\phi(z'|\boldsymbol{q}) = \sum_{\boldsymbol{d}} p^\phi(z'|\boldsymbol{d})p(\boldsymbol{d}|\boldsymbol{q})$, both of which are also part of the logarithm part of the two entropy terms. Therefore, $I(Z';\boldsymbol{Q})$ is again intractable if we only use $p^\phi(z'|\boldsymbol{d})$. We approximate $p^\phi(z')$ and $p^\phi(z'|\boldsymbol{q})$ by neural networks $g^\theta(z')$ and $p^\psi(z|\boldsymbol{q})$ respectively, which falls into the category of variational methods. As $H(Z') = H(p^\phi(z')) \leq H(p^\phi(z');g^\theta(z'))$ and $H(Z'|\boldsymbol{Q}) = \mathbb{E}_{\boldsymbol{q}\sim p_Q} H(p^\phi(z'|\boldsymbol{q})) \leq \mathbb{E}_{\boldsymbol{q}\sim p_Q} H(p^\phi(z'|\boldsymbol{q});p^\psi(z|\boldsymbol{q}))$, we upper-bound $H(Z'|\boldsymbol{Q})$ with $H(Z';Z) := \mathbb{E}_{(\boldsymbol{q},\boldsymbol{d})\sim p_{QD}}[H(p^\phi(z'|\boldsymbol{d});p^\psi(z|\boldsymbol{q}))]$, and introduce $H^+(Z') := \mathbb{E}_{\boldsymbol{d}\sim p_D} H(p^\phi(z'|\boldsymbol{d});g^\theta(z'))$ as an upper bound of $H(Z')$. Note that with $H(X) = \min_Y H(X;Y)$, our approximation $\hat{I}(Z';\boldsymbol{Q}) = H^+(Z') - H(Z';Z)$ gives exact evaluation of $I(Z';\boldsymbol{Q})$ when $H^+(Z')$ is minimized w.r.t. θ when $g^\theta(z') = p^\phi(z')$, and $H(Z';Z)$ is minimized w.r.t. ψ when $p^\psi(z'|\boldsymbol{q}) = p^\phi(z'|\boldsymbol{q})$.

C Theoretical Analysis of Stochastic Optimization of MICO

First we show why computing $I(Z';Z)$ is intractable even under the stochastic optimization framework. With the MICO framework, the expectation over p_{QD} is approximated by the average over one batch of data $\boldsymbol{B} = \{(\boldsymbol{q}_1,\boldsymbol{d}_1),(\boldsymbol{q}_2,\boldsymbol{d}_2),\dots,(\boldsymbol{q}_b,\boldsymbol{d}_b)\}$. If one objective is f consisting of the expectation over p_{QD}, we evaluate f on $\boldsymbol{B}$ and denote the corresponding result as $f_{\boldsymbol{B}}$. The core assumption in stochastic optimization is $f = \mathbb{E}_{\boldsymbol{B}} f_{\boldsymbol{B}}$.

For

$$I(Z';Z) = \mathop{\mathbb{E}}_{(\boldsymbol{q},\boldsymbol{d})\sim p_{QD}} \left[\sum_{z,z'} p^\phi(z'|\boldsymbol{d}) p^\psi(z|\boldsymbol{q}) \log \left(\frac{\mathbb{E}_{(\boldsymbol{q},\boldsymbol{d})\sim p_{QD}}[p^\phi(z'|\boldsymbol{d})p^\psi(z|\boldsymbol{q})]}{\mathbb{E}_{(\boldsymbol{q},\boldsymbol{d})\sim p_{QD}}[p^\phi(z'|\boldsymbol{d})]\mathbb{E}_{(\boldsymbol{q},\boldsymbol{d})\sim p_{Q,D}}[p^\psi(z|\boldsymbol{q})]} \right) \right],$$

if we use

$$I_{\boldsymbol{B}}(Z';Z) = \frac{1}{b}\sum_{i=1}^{b} \left[\sum_{z,z'} p^\phi(z'|\boldsymbol{d}_i) p^\psi(z|\boldsymbol{q}_i) \log \left(\frac{\frac{1}{b}\sum_{i=1}^{b}[p^\phi(z'|\boldsymbol{d}_i)p^\psi(z|\boldsymbol{q}_i)]}{\frac{1}{b}\sum_{i=1}^{b}[p^\phi(z'|\boldsymbol{d}_i)]\frac{1}{b}\sum_{i=1}^{b}[p^\psi(z|\boldsymbol{q}_i)]} \right) \right],$$

it violates the assumption since $I(Z';Z) \neq \mathbb{E}_{\boldsymbol{B}}[I_{\boldsymbol{B}}(Z';Z)]$.

In MICO, we replace $I(Z';Z)$ with $I(Z';\boldsymbol{Q})$. Moreover, we use

$$H_{\boldsymbol{B}}(Z';Z) = \frac{1}{b}\sum_{i=1}^{b} \left[\sum_{z=z'} -p^\phi(z'|\boldsymbol{d}_i) \log \left(p^\psi(z|\boldsymbol{q}_i) \right) \right]$$

and

$$H_{\boldsymbol{B}}(Z') = \frac{1}{b}\sum_{i=1}^{b} \sum_{z'} -p^\phi(z'|\boldsymbol{d}) \log \left(\frac{1}{b}\sum_{i=1}^{b} p^\phi(z'|\boldsymbol{d}_i) \right)$$

to approximate

$$H(Z';Z) = \mathbb{E}_{(\boldsymbol{q},\boldsymbol{d})\sim p_{Q,D}} \left[\sum_{z=z'} -p^\phi(z'|\boldsymbol{d}) \log \left(p^\psi(z|\boldsymbol{q}) \right) \right]$$

and

$$H(Z') = \sum_{z'} -\mathbb{E}_{(\boldsymbol{q},\boldsymbol{d})\sim p_{Q,D}}[p^\phi(z'|\boldsymbol{d})] \log \left(\mathbb{E}_{(\boldsymbol{q},\boldsymbol{d})\sim p_{Q,D}}[p^\phi(z'|\boldsymbol{d})] \right),$$

respectively.

Note that $H(Z',Z) = \mathbb{E}_{\boldsymbol{B}}[H_{\boldsymbol{B}}(Z',Z)]$ while $H(Z') \neq \mathbb{E}_{\boldsymbol{B}}[H_{\boldsymbol{B}}(Z')]$. This is due to the logarithm part in $H_{\boldsymbol{B}}(Z')$. Without using the stochastic approximation on the logarithm, since it is expensive to evaluate the expectation of $p^\phi(z'|\boldsymbol{d})$ on the full dataset, we have to replace it with $g^\theta(z')$ in the logarithm term which creates an upper bound $H^+(Z')$ on $H(Z')$ as an approximation (used in the tractable loss function of MICO):

$$H^+(Z') = \sum_{z'} -\mathbb{E}_{(\boldsymbol{q},\boldsymbol{d})\sim p_{Q,D}}[p^\phi(z'|\boldsymbol{d})] \log \left(g^\theta(z') \right).$$

$$H^+_{\boldsymbol{B}}(Z') = \frac{1}{b}\sum_{i=1}^{b} \sum_{z'} -p^\phi(z'|\boldsymbol{d}) \log \left(g^\theta(z') \right),$$

$$H^+(Z') = \mathbb{E}_{\boldsymbol{B}}[H^+_{\boldsymbol{B}}(Z')]$$

The stochastic evaluation of $\mathcal{L}$ is then $\mathcal{L}_{\boldsymbol{B}} = \beta H^+_{\boldsymbol{B}}(Z') - H_{\boldsymbol{B}}(Z';Z)$ and we have $\mathcal{L} = \mathbb{E}_{\boldsymbol{B}}[\mathcal{L}_{\boldsymbol{B}}]$, which is also the reason of why we use $H(Z';Z)$ instead of $H(Z'|\boldsymbol{Q})$ in MICO.

For each batch of data, since $H(Z') = \min_{g^\theta(z')} H^+(Z')$, we first update $g^\theta(z')$ for one step to descend $H^+_{\boldsymbol{B}}(Z')$ (corresponding to SGD), and then update $p^\psi(z'|\boldsymbol{q})$ and $p^\phi(z'|\boldsymbol{d})$ for one step to descend $\mathcal{L}_{\boldsymbol{B}}$ (corresponding to another SGD).

DPTDR: Deep Prompt Tuning for Dense Passage Retrieval

Zhengyang Tang[1,2], Benyou Wang[2], and Ting Yao[1]

[1]Tencent Inc.
[2]The Chinese University of Hong Kong, Shenzhen
{zhytang,tessieyao}@tencent.com, wangbenyou@cuhk.edu.cn

Abstract

Deep prompt tuning (DPT) has gained great success in most natural language processing (NLP) tasks. However, it is not well-investigated in dense retrieval where fine-tuning (FT) still dominates. When deploying multiple retrieval tasks using the same backbone model (e.g., RoBERTa), FT-based methods are unfriendly in terms of deployment cost: each new retrieval model needs to repeatedly deploy the backbone model without reuse. To reduce the deployment cost in such a scenario, this work investigates applying DPT in dense retrieval. The challenge is that *directly applying DPT in dense retrieval largely underperforms FT methods*. To compensate for the performance drop, we propose two model-agnostic and task-agnostic strategies for DPT-based retrievers, namely *retrieval-oriented intermediate pretraining* and *unified negative mining*, as a general approach that could be compatible with any pre-trained language model and retrieval task. The experimental results [1] show that the proposed method (called DPTDR) outperforms previous state-of-the-art models on both MS-MARCO and Natural Questions. We also conduct ablation studies to examine the effectiveness of each strategy in DPTDR. We believe this work facilitates the industry, as it saves enormous efforts and costs of deployment and increases the utility of computing resources.

1 Introduction

Fine-tuning (FT) has been a de facto approach for effective dense passage retrieval (Karpukhin et al., 2020; Xiong et al., 2020) based on pre-trained language models (PLM). However, FT is unfriendly for industrial deployment in multi-task scenarios. Imaging for cloud service providers or infrastructure teams of search companies, each retrieval model (w.r.t., an individual task) necessarily redeploys a backbone model since the weights of the backbone model in each task are fine-tuned and therefore slightly different. That dramatically increases hardware costs and inefficiency.

Recently, prompt tuning (PT) (Liu et al., 2021a) is a lightweight alternative to FT, which does not need storing a full copy of the backbone model for each task. One variant of PT, namely Deep Prompt Tuning (DPT; Li and Liang, 2021; Liu et al., 2021b), exhibits comparable performances with FT in various NLP tasks. DPT enjoys parameter-efficient(Houlsby et al., 2019) characteristics, of which the resulting prompts are light-weighted and can be easily passed to an online PLM service, thus overcoming the above challenge of FT. This paper asks: *whether can we replace FT by DPT with comparable performance to SOTA FT methods in dense passage retrieval?* With comparable performance, DPT is much more friendly in deployment than FT.

DPT usually freezes weights in the backbone models and alternatively trains deep prompts inserted; the latter has much fewer parameters than the former. However, freezing most weights in DPT hinders its adaptability and therefore possibly harms performance. Experimental results in Sec. 4.2.2 also demonstrate *directly applying DPT in dense retrieval largely underperforms FT methods*.

To make DPT comparable to FT in dense retrieval, a natural solution is *retrieval-oriented intermediate pretraining (RIP)*, which warms up the text representation via contrastive learning. Though it is not a novel idea(Lee et al., 2019; Gao and Callan, 2021b; Izacard et al., 2021), there exist two different pretraining ways tailored for DPT-based retrievers. One is to pre-train deep prompts while freezing the PLM backbone and use the pre-trained prompts to initialize a DPT retriever. The other is to pre-train a PLM directly and initialize a DPT retriever using the pre-trained PLM; in contrast to prior works(Gao and Callan, 2021b), we intend to allow any PLM easily pre-trained for DPT so that

[1]Our code is available at https://github.com/tangzhy/DPTDR

Proceedings of the 29th International Conference on Computational Linguistics, pages 1193–1202
October 12–17, 2022.

users may employ their own PLMs, and thus we deliberately remove the workload to modify any model structures. Surprisingly, empirical findings in Sec. 4.4 show that this choice yields better performance than carefully modified PLMs(Gao and Callan, 2021b). Furthermore, we propose a *unified negative mining (UNM)* to merge retrieved negatives from many retrievers including BM25 and dense retrievers, in order to provide diverse and hard negatives for DPT training.

By incorporating RIP and UNM, we implement a **D**eep **P**rompt **T**uning method in **D**ense **R**etrieval tasks, called *DPTDR*. The experimental results show that DPTDR outperforms previous state-of-the-art models on both MS-MARCO and Natural Questions. We also conduct extensive experiments and find that: i) when combined with RIP and UNM, DPT is able to obtain comparable performance with FT in dense retrieval and exhibits insensitivity to prompt length, and ii) both RIP and UNM are effective in improving the performance. The contributions of this paper can be summarized as follows:

- To our best knowledge, this is the first work to apply DPT in dense retrieval. We bring forward two essential strategies, namely retrieval-oriented intermediate pretraining and unified negative mining, allowing DPT to match FT's performance and be compatible with any PLM.
- Experiments show that DPTDR outperforms previous state-of-the-art models on MS-MARCO and Natural Questions and examine the effectiveness of the above strategies.
- We believe this work facilitates the industry, as it saves enormous efforts and costs of deployment and increases the utility of computing resources.

2 Related Work

2.1 Deep Prompt Tuning

DPT originates from prompting and prompt tuning (Liu et al., 2021a). Given some discrete or continuous prompts, PLMs like GPT-3(Brown et al., 2020) can achieve impressive zero-shot and few-shot performances for knowledge-intensive tasks. However, studies find that prompt tuning fails to perform well for moderate-size models (Liu et al., 2021b). Thus, DPT(Li and Liang, 2021; Liu et al., 2021b) is proposed by inserting prompts at deep layers to steer PLMs towards desired directions more capably. It obtains comparable performance to FT across a range of NLP tasks. DPTDR is mainly related to DPT, focusing on dense passage retrieval instead of NLP. There also exist works of pretraining prompts for prompt tuning(Gu et al., 2021), which shows effectiveness in few-shot learning using billion-size models, as we will explore as well in the context of DPT.

2.2 Dense Retrieval

Pretraining We have witnessed a series of unsupervised pretraining works proposed for dense retrieval, such as ICT, BFS, WLP, and independent cropping (Lee et al., 2019; Chang et al., 2020; Izacard et al., 2021). Following works also try to pretrain retriever and reader jointly for question answering (Guu et al., 2020). coCondenser (Gao and Callan, 2021b) follows a contrastive learning framework using Condenser structure (Gao and Callan, 2021a) by adding an explicit decoder to learn representations better. There are also semi-supervised and weakly-supervised works. DPR-PAQ (Oğuz et al., 2021) pre-trains a PLM using 65-million-size synthetic QA pairs on the target corpus. GTR (Ni et al., 2021) pre-trains T5 (Raffel et al., 2019) on 2-billion-size community QA pairs from T5-base to T5-xxlarge. We follow unsupervised contrastive learning as our pretraining strategy for DPTDR. However, we aim to ensure compatibility with any PLM, thus resulting in different sample building processes and model structure choices.

Negative mining DPR (Karpukhin et al., 2020) proposes to train retrievers using BM25 negatives. ANCE (Xiong et al., 2020) extends that by mining negatives periodically from previously-trained dense retrievers. RocketQA and RocketQAv2 (Qu et al., 2021; Ren et al., 2021) introduce the idea of denoised negative sampling by selecting negatives with high confidence scored by a re-ranker. DPTDR unifies the above into a general negative mining strategy.

3 Methodology

In this section, we first formalize the application of DPT in dense retrieval. We then describe the two strategies of RIP and UNM for DPT-based retrievers.

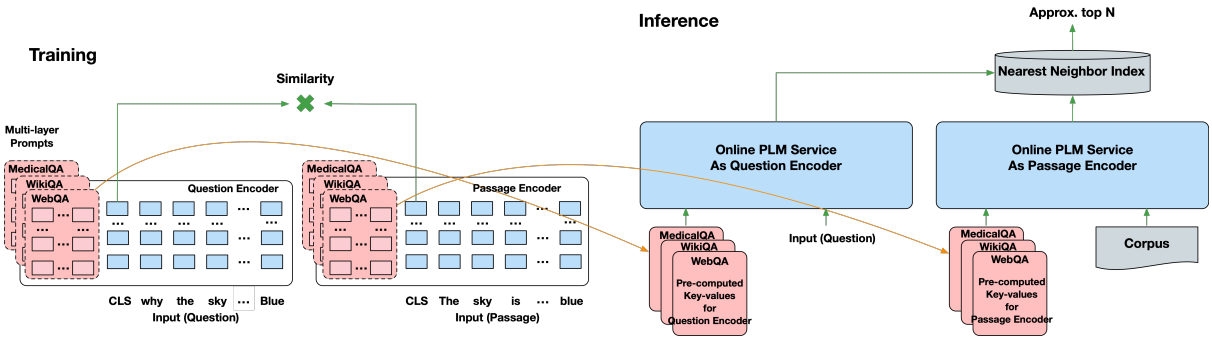

Figure 1: The framework of DPTDR. We first perform RIP which results in a PLM (the blue blocks) that can be used as the backbone for DPT training and deployed once as online PLM services. Then we train deep prompts (i.e., DPT) for different retrieval tasks such as WebQA, WikiQA, and MedicalQA (the pink blocks), during which we may employ UNM to improve performances. For inference, we can send tokenized input, together with trained prompts of their corresponding task, to online PLM services to get dense vectors.

3.1 DPT in Dense Retrieval

Let C be a corpus consisting N passages, denoted by $p_1, p_2, ..., p_N$. Given a question q, the task of dense retrieval is to find a passage p_i that is considered relevant to the question.

The dual-encoder Normally, a dual-encoder is applied. First its passage encoder $E_p(\cdot)$ embeds a passage p to a d-dimensional dense vector. Then a vector search index (Johnson et al., 2019) of passages is built for retrieval. At inference time, the question encoder $E_q(\cdot)$ embeds the question q to a d-dimensional dense vector, and k passages closet to the question based on the vector similarity will be retrieved. In practice, the similarity score is computed as the inner product:

$$s(q, p) = E_q(q) \cdot E_p(p). \quad (1)$$

For PLM-based dual-encoder, we usually take the representation at the first token (e.g., [CLS] symbol in BERT (Devlin et al., 2018)) as the output dense vector.

Deep prompt tuning We then apply DPT in the PLM-based dual-encoder, as illustrated in the left part of Figure 1. To prepend multi-layer prompts for the dual-encoder, we initialize a trainable prefix matrix M of dimension $l \times d$ for each layer of the PLM, where l is the length of the prompt and d is the hidden size of the PLM. Since the prompt resides at the deep layers of PLM, it has a full capacity to steer the PLM towards the desired direction and output meaningful dense vector for questions and passages. Note that a verbalizer (Schick and Schütze, 2020) plays a vital role in mapping words to labels in canonical prompt tuning. However, we remove it in dense retrieval since the output dense vector is what we need. Let E'_p as the prompted passage encoder and E'_q as the prompted question encoder, and the similarity score is computed:

$$s'(q, p) = E'_q(q) \cdot E'_p(p). \quad (2)$$

Training The objective of the training is to learn dense vectors so that the similarity between relevant pairs of questions and passages ranks higher than irrelevant ones. Given a pair of question q and positive passage p_i, along with n negative passages, we optimize the loss function as the negative log-likelihood of the positive passage:

$$L(q_i, p_i^+, \{p_{i,j}^-\}_{j=1}^n) = -\log \frac{e^{s'(q_i, p_i^+)}}{e^{s'(q_i, p_i^+)} + \sum_{j=1}^n e^{s'(q_i, p_{i,j}^-)}}. \quad (3)$$

Generating negative passages is critical for the performance, and we will explain it in Sec. 3.3. During training, we freeze parameters of the backbone PLM and only update the deep prompts, where approximately 0.1%-0.4% parameters of a PLM get trained.

Inference As illustrated in the right part of Figure 1, since the backbone PLM is frozen, it is possible to deploy it ahead as online PLM services and then pass the trained prompts as pre-computed key values together with tokenized inputs to get dense vectors. It is at the core of how we save efforts and costs of deployment and increase the utility of computing resources. In practice, the cloud service providers or infrastructure teams of search companies are able to focus on the PLM as a central service, while users can quickly train deep prompts

for different retrieval tasks and obtain efficient and compelling retrieval performances without any deployment.

Although DPT brings in many advantages, it is worth noting that it does not accelerate the inference speed because the forward computation is not reduced but increased slightly.

3.2 Retrieval-oriented Intermediate Pretraining (RIP) for DPT

The goal of RIP is to either pre-train deep prompts or PLMs using contrastive learning. We first describe the task as follows. Let C denote a corpus consisting N passages. For a passage p_i, we split it into l sentences, denoted by $s_i^1, ..., s_i^l$. Given a sentence s_i^j, the task of pretraining is to distinguish its context sentence $s_i^{j'}$ from sentences of other passages s_k^l, where $k \neq i$. Formally, we randomly select a pair of sentences from each passage as context sentences to form a batch of training data $B = \{s_i^1, s_i^2\}_{i=1}^m$, where m is the batch size. Then we define the contrastive loss for s_i^j over the batch as:

$$L_c(s_i^j) = -\log \frac{e^{s(s_i^1, s_i^2)}}{\sum_{k=1}^{m} \sum_{l=1}^{2} {}_{ij \neq kl} e^{s(s_i^j, s_k^l)}}. \quad (4)$$

In contrast to prior works(Gao and Callan, 2021b; Izacard et al., 2021), we directly sample sentences as opposed to text spans. Since sampling text spans is a non-trivial technique where factors such as the probability of short sentences and how to keep the spans linguistically meaningful can have a complicated effect on the pretraining, we remove this complexity in our approach. We also conduct an experiment observing sentences work even better than text spans on MS-MARCO corpus (Sec. 4.4).

Under the contrastive learning task, there exist two pretraining ways tailored for DPT, depending on the pre-trained objects (i.e., the deep prompts or the PLM backbone).

Pre-train deep prompts One is to pre-train deep prompts with a vanilla PLM. Later we initialize a DPT-based retriever using the pre-trained deep prompts and the vanilla PLM. However, experiments in Sec. 4.4 show that it suffers from catastrophic forgetting and exhibits no superior performance to randomly initialized prompts.

Pre-train the PLM The other is to pre-train a PLM, and then we initialize a DPT-based retriever using randomly-initialized deep prompts and the pre-trained PLM. Notice that we intend to allow any PLM to be easily pre-trained for DPT so that users may employ their own PLMs. Thus we contrast prior works such as coCondenser(Gao and Callan, 2021b), a state-of-the-art model structure in contrastive pretraining, by removing the workload to modify any model structures. Surprisingly, it yields better performance than coCondenser in Table 8. Therefore, we refer RIP strategy as pretraining of PLMs for the rest.

For any PLM, We also intend to remain its original self-supervised tasks, such as masked language modeling(MLM; Devlin et al., 2018; Sun et al., 2019), denoted as L_s. Therefore, the final loss of pretraining over the batch is:

$$L = \frac{1}{2m} \sum_{i=1}^{m} \sum_{j=1}^{2} L_s(s_i^j) + L_c(s_i^j). \quad (5)$$

After pretraining, the resulting model can be deployed once as online services and taken as the backbone model for DPT training.

3.3 Unified Negative Mining (UNM)

We also develop unified negative mining for DPT, as interpreted as "Multiple Retrievers & Hybrid Sampling." "Multiple Retrievers" is to incorporate negatives from as many retrievers as we can. We use a BM25 retriever as the initial retriever and train a DPT-based retriever using BM25 negatives. Later we treated retrieved negatives from the BM25 retriever and the first DPT-based retriever as un-denoised hard negatives. Users are allowed to introduce any other retrievers if possible. "Hybrid Sampling" is to select denoised hard negatives from un-denoised hard negatives retrieved by the above multiple retrievers. We borrow an existing re-ranker released by RocketQA (Qu et al., 2021) and select those negatives with high confidence. For training the final DPT-based retriever, we mix the denoised hard negatives, un-denoised hard negatives, and easy negatives from in-batch or cross-batch training.

We believe unified negative mining is critical for the performance of DPT-based retrievers, as it provides negatives of high quality and diversity.

4 Experiments

4.1 Experimental Setting

Datasets and metrics We experiment with two popular dense retrieval datasets, including MS-

Table 1: The statistics of MS-MARCO and Natural Questions.

Dataset	#q in train	#q in dev	#q in test	#passages
MS-MARCO	502,939	6,980	6,837	8,841,823
Natural Questions	58,812	6,515	3,610	21,015,324

MARCO (Bajaj et al., 2016) and Natural Questions(NQ; Karpukhin et al., 2020). The statistics of the datasets are listed in Table 1. MS-MARCO is constructed from Bing's search query logs and web documents retrieved by Bing. Natural Question contains questions from Google Search. For evaluation, we report official metrics MRR@10, RECALL@1000 for MS-MARCO, and RECALL at 5, 20, and 100 for NQ. All models are trained on a single server with 8 NVIDIA Tesla A100 GPUs.

Settings in DPT We use RoBERTa-large-size models as the backbone for DPT training. Hyper-parameters are explored as below.

- **Learning rate** We search for 1e-2, 5e-3, 7e-3, 5e-4, 5e-5, 5e-6 with prompts' length of 32, where 7e-3 performs relatively better than others and is set for the main experiment.
- **Training epochs** For training epochs, we search for 3, 6, 10 with a learning rate 7e-3 on MS-MARCO, where 10 performs best and is set for the main experiment. We also set training epochs as 60 for NQ for acceptable time cost.
- **Prompt length** We search for 8, 16, 32, 64, 128, as is discussed in Sec. 4.3. We use 128 for the main experiment.
- **Reparametrization** We also conduct experiments for prompts with or without MLP reparametrization, as is discussed in Sec. 4.3. We use non-reparametrization for the main experiment.

We follow coCondenser (Gao and Callan, 2021b) for other hyper-parameters (e.g., parameter sharing, batch size, warm-up ratio, and mixed-precision training).

Settings in RIP We choose to pre-train vanilla RoBERTa-large for RIP, whose model size appears more common for DPT (Li and Liang, 2021; Liu et al., 2021b) and is consistent with the above DPT training. We remain RoBERTa's original self-supervised task (MLM; Liu et al., 2019). To compare our approach with coCondenser (Gao and Callan, 2021b), we also pre-train a coCondesner RoBERTa-large. Since coCondenser modifies the PLM by adding a carefully designed Condenser structure, we follow their structural setting using an equal split, 12 early layers, and 12 late layers. We split the passages into sentences on both MS-MARCO and NQ Wikipedia as the training corpus. The models are trained using AdamW optimizer with a learning rate 1e-4, weight decay of 0.01, linear learning rate decay, and a batch size of 2K. We train 8 epochs for MS-MARCO and 4 epochs for NQ Wikipedia.

Settings in UNM For un-denoised hard negatives, we randomly select 30 out of the top 200 retrieved negatives from multiple retrievers. For denoised hard negatives, we select negatives with a score less than 0.1 output by an existing reranker (Qu et al., 2021).

Baselines We use the following baselines. **coCondenser** (Gao and Callan, 2021b) designs a complicated pretraining model structure on top of a vanilla PLM. **DPR-PAQ** (Oğuz et al., 2021) pre-trains a RoBERTa-large using 65-million-size synthetic QA pairs. Since the data is created by a model trained on NQ (Kwiatkowski et al., 2019) and Trivia QA (Joshi et al., 2017), it can be considered a semi-supervised method. It is also comparable to us as both of us use RoBERTa-large. **GTR** (Ni et al., 2021) pre-trains T5 encoder (Raffel et al., 2019) using 2-billion size community QA pairs. It also provides results across all model size ranges from T5-base to T5-xxlarge. The massive training corpus and model size establish a SOTA performance.

We also include some standard baselines including sparse retrieval systems (BM25, DeepCT (Dai and Callan, 2019), DocT5Query (Nogueira et al., 2019), and GAR (Mao et al., 2020)) and dense retrieval systems (DPR (Karpukhin et al., 2020), ANCE (Xiong et al., 2020), ME-BERT (Luan et al., 2020), and RocketQA (Qu et al., 2021)). We also include RocketQAv2 (Ren et al., 2021) as it jointly trains the retriever and reranker using hybrid sam-

Table 2: Passage retrieval results on MS-MARCO Dev and Natural Questions Test. We copy the results from the original papers. The best and second-best results are in bold and underlined fonts respectively.

Methods	**PLM**	**MS-MARCO Dev**		**Natural Questions Test**		
		MRR@10	R@1000	R@5	R@20	R@100
BM25	-	18.7	85.7	-	59.1	73.7
DeepCT(Dai and Callan, 2019)	-	24.3	91.0	-	-	-
docT5query(Nogueira et al., 2019)	-	27.7	94.7	-	-	-
GAR(Mao et al., 2020)	-	-	-	-	74.4	85.3
DPR(Karpukhin et al., 2020)	BERT-base	-	-	-	78.4	85.4
ANCE(Xiong et al., 2020)	RoBERTa-base	33.0	95.9	-	81.9	87.5
ME-BERT(Luan et al., 2020)	BERT-large	34.3	-	-	-	-
RocketQA(Qu et al., 2021)	ERNIE-base	37.0	97.9	74.0	82.7	88.5
RocketQAv2(Ren et al., 2021)	ERNIE-base	<u>38.8</u>	98.1	75.1	83.7	89.0
coCondenser(Gao and Callan, 2021b)	Condenser	38.2	98.4	75.8	84.3	89.0
DPR-PAQ(Oğuz et al., 2021)	RoBERTa-large	34.0	-	<u>76.9</u>	<u>84.7</u>	<u>89.2</u>
GTR(Ni et al., 2021)	T5-base	36.6	98.3	-	-	-
	T5-large	37.9	**99.1**	-	-	-
	T5-xlarge	38.5	98.9	-	-	-
	T5-xxlarge	<u>38.8</u>	<u>99.0</u>	-	-	-
DPTDR	RoBERTa-large	**39.1**	98.9	**77.5**	**85.1**	**89.4**

pled negatives.

4.2 Experimental Results

4.2.1 Comparison with Existing Methods

Table 2 shows the dev set performance for MS-MARCO and test set performance for NQ. We can generally see that **DPTDR outperforms all the baselines in terms of MRR@10 on MS-MARCO and R@5 on NQ** and set a new SOTA in the two datasets.

We first compare DPTDR with DPR-PAQ. DPR-PAQ achieves competitive performance on NQ. It should be expected since it involves large semi-supervised pretraining on the NQ dataset. Nonetheless, DPTDR still outperforms DPR-PAQ by 0.6 points in R@5 although we use an unsupervised pretraining model. When we study the performance on MS-MARCO, DPR-PAQ fails to perform as consistently well as on NQ, which could result from domain mismatch of pretraining, and DPTDR outperforms it by a significant margin of 5.1 points in MRR@10.

Secondly, we compare DPTDR with GTR. GTR pre-trains T5 using 2-billion-size community QA pairs as a weakly-supervised pretraining. For such a scale of training corpus, we would expect that larger models consume the corpus more thoroughly and perform better on downstream tasks. As a result, GTR consistently boosts the performance on MS-MARCO with the model size increasing. However, DPTDR still outperforms GTR T5-xxlarge, a 10-billion-size model, and outperforms GTR T5-large by a noticeable margin of 1.2 points in MRR@10. It shows that model size is a positive contributor but not an absolute dominator for dense retrieval. Appropriate pretraining and negative mining can help improve performances using much more affordable computing resources. At the same time, note that DPT shall play a critical role in achieving comparable performance to FT with the help of RIP and UNM. We will validate this in Sec. 4.2.2.

Finally, we would like to compare DPTDR with coCondenser. Since coCondenser employs a pre-trained Condenser model(Gao and Callan, 2021a), we will conduct a more fair comparison in Sec. 4.4.

4.2.2 Comparing FT with and without RIP and UNM Strategies

To answer the raised question: *whether can we replace FT by DPT with comparable performance to SOTA FT methods in dense passage retrieval?* We conduct FT by following hyper-parameters of coCondenser (Gao and Callan, 2021b).

Comparison w/o RIP&UNM As a starter, we examine the effectiveness of directly replacing FT with DPT, which means we conduct training without RIP and UNM strategies. Thus we use the vanilla RoBERTa-large as the backbone model and BM25 negatives. As is shown in Table 3. We notice

Table 3: The comparison between FT and DPT with and without RIP and UNM strategies on MS-MARCO Dev and Natural Questions Test. **DPT with RIP&UNM** is the proposed method, a.k.a, 'DPTDR'.

		MS-MARCO Dev		Natural Questions Test		
		MRR@10	R@1000	R@5	R@20	R@100
w/o RIP&UNM	FT	34.9	97.2	68.8	80.0	86.4
	DPT	32.7 (**2.2** ↓)	96.3 (**0.9** ↓)	66.5 (**2.3** ↓)	78.5 (**1.5** ↓)	85.5 (**0.9** ↓)
w/ RIP&UNM	FT	39.4	99.0	77.0	85.4	89.2
	DPT	39.1 (**0.3** ↓)	98.9 (**0.1** ↓)	77.5 (**0.5** ↑)	85.1 (**0.3** ↓)	89.4 (**0.2** ↑)

that DPT largely underperforms FT in this setting with a noticeable margin of 2.2 points in MRR@10 on MS-MARCO and 2.3 points in R@5 on NQ. It indicates that freezing most weights in DPT actually hinders its adaptability and therefore harms performance.

Comparison w/ RIP&UNM Next, we examine the performance of FT and DPT with RIP and UNM strategies. We use the RIP RoBERTa-large as the backbone model and UNM negatives. Table 3 shows that *i*) RIP and UNM improve the performances of both FT and DPT and *ii*) most importantly, DPT is comparable to FT under this setting, where the gap shrinks to only 0.3 points in MRR@10 on MS-MARCO, and DPT even slightly outperforms FT by 0.5 points in R@5 on NQ. As a result, we can see that when combined with RIP and UNM, DPT can obtain comparable performance with FT in dense retrieval.

4.3 Analysis on DPT

Sensitivity on prompt length We also seek to understand how prompt length affects the performance of DPT-based retrievers. From Table 4, we observe that the performance of prompt length of 8 already achieves a strong MRR@10 at 38.6 on MS-MARCO. When we increase the length to 128, it makes the most robust performance of MRR@10 at 39.1. The longer prompt means more trainable parameters, which obtains more power to steer PLMs. However, we also want to point out that the DPT retriever exhibits insensitivity to prompt length since the performances are competitive overall across various lengths. Therefore, we choose 32 as the default prompt length along with other hyperparameters in the main experiment for the rest of the ablation studies on MS-MARCO to accelerate the training.

Impact of reparameterization Reparametrization plays an important role in DPT. Li and Liang, 2021 point out that MLP reparametrization results in more stable and compelling performances, while Liu et al., 2021b find it still depends on different tasks. In dense retrieval, we aim to determine whether it has a positive effect. Table 5 presents the results on MS-MARCO. We observe that MLP reparametrization results in a performance drop in MRR@10 on MS-MARCO. Since MLP breaks the independence of inter-layer prompts, we conjecture this brings optimization difficulty for dense retrieval.

Table 4: Sensitivity of prompt length on MS-MARCO Dev.

Prompt Length	MRR@10	R@1000
8	38.6	98.9
16	38.6	99.0
32	38.7	98.9
64	38.5	98.9
128	39.1	98.9

Table 5: Ablations of reparamerization on MS-MARCO Dev.

Reparamerization	MRR@10	R@1000
embedding	38.7	98.9
mlp	38.0	99.0

4.4 Analysis on RIP

Whether to pre-train deep prompts or not? We try to examine whether pre-trained deep prompts could improve the performance of DPT-based retrievers. We use BERT-base as our backbone model and pre-train deep prompts of length 32 without reparameterization. The pretraining tasks and corpus are exactly the same as Sec. 3.2. We initialize DPT-based retrievers using pre-trained and randomly-initialized prompts. As is shown in Table 6, the pre-trained prompts do not boost the performance over randomly initialized prompts on

MS-MARCO. It reveals that the deep prompts may easily suffer from catastrophic forgetting.

Table 6: Ablations of prompt initialization on MS-MARCO Dev.

Prompt Initialization	MRR@10	R@1000
Random	32.4	95.5
Pre-trained	32.4	95.5

RIP on text spans or sentences We also explore pretraining using randomly-sampled sentences versus randomly-sampled text spans. Since coCondenser(Gao and Callan, 2021b) releases their pretrained model using randomly-sampled text spans, we directly use their model to examine the zero-shot performance. For sampling sentences, we use the same PLM and hyper-parameters based on coCondenser code[2] except changing the training corpus consisting of randomly-sampled sentences. Table 7 presents the zero-shot performance on MS-MARCO. The pretraining using sentences works better than the one using text spans. This is might be owing to that text-spans do not consider the (starting and ending) borders of natural sentences and therefore break their completeness in semantics.

Table 7: Zero-shot performance of coCondenser with different sampling granularity (i.e., sentences or spans) on MS-MARCO Dev.

Unit	MRR@10	R@1000
Spans Gao and Callan (2021b)	11.1	78.2
Sentences	15.4	87.2

RIP's effectiveness and comparison with coCondenser We also try to examine the effectiveness of RIP strategy and compare it with coCondenser (Gao and Callan, 2021b). Concretely, we take vanilla RoBERTa-large, coCondenser RoBERTa-large, and RIP RoBERTa-large as the backbone model for DPT training under the same setting. Table 8 presents their results in both zero-shot and full-shot settings on MS-MARCO. For vanilla RoBERTa-large, it performs extremely poorly in zero-shot experiments, and with no surprise, it performs worst in full-shot experiments among the three PLMs. For coCondenser RoBERTa-large, it achieves a noticeable improvement over vanilla RoBERTa-large, where MRR@10 of zero-shot performance becomes meaningful at 6.3, and MRR@10 of full-shot performance increases to 37.3. For RIP RoBERTa-large, we see it achieves the best performance in both zero-shot and full-shot experiments. We also borrow the analysis tool from Wang and Isola (2020), which takes l_{align} between semantically-related positive pairs and $l_{uniform}$ of representation space to measure the quality of PLM representations. For both the metrics, lower numbers are better. RIP is much better than the vanilla model in both alignment and uniformity, while coCondenser works well in alignment but worse in uniformity.

Thus a question is raised: *does PLM need additional structures for contrastive pretraining?* Both zero-shot and full-shot experiments demonstrate that RIP works even better than a carefully modified model structure. Therefore, we conjecture that PLM's multi-layer transformers could be already expressive enough for dense retrieval under an appropriate contrastive learning task. However, additional model structures may bring optimization difficulty, especially when the number of added parameters is large.

4.5 Analysis on UNM

Ablation on UNM We try to understand how UNM affects performances. Table 9 presents the results on MS-MARCO. DPT using BM25 negatives achieves a baseline of MRR@10 at 36.8. When combining un-denoised hard negatives from multiple retrievers, we see that the performance achieves a noticeable improvement in MRR@10 by 1.5 points. When combining denoised hard negatives selected by a re-ranker, the performance further gets boosted of which MRR@10 increases by 0.4 points. The results demonstrate that both multiple retrievers and hybrid sampling positively contribute to dense retrieval.

5 Conclusion

In this paper, we investigate applying DPT in dense passage retrieval. To mitigate the performance drop of a vanilla DPT, We also propose two strategies, namely RIP and UNM, to enhance DPT and match the performance of FT. Experiments show that DPTDR outperforms previous state-of-the-art models on both MS-MARCO and Natural Questions and demonstrated the effectiveness of the above strategies. We believe this work facilitates the in-

[2] https://github.com/luyug/Condenser

Table 8: Ablations of different PLMs for DPT on MS-MARCO Dev.

Backbone	**Zero-shot**				**Full-shot**	
PLM	l_{align}	$l_{uniform}$	MRR@10	R@1000	MRR@10	R@1000
vanilla RoBERTa-large	161.4	-13.8	0.0	0.1	35.5	97.5
coCondenser RoBERTa-large	4.9	-12.9	6.4	63.3	37.3	98.0
RIP RoBERTa-large	21.9	-26.4	14.3	87.2	38.7	98.9

Table 9: Ablations of UNM on MS-MARCO Dev.

Neg Pool	MRR@10	R@1000
BM25 Neg	36.8	98.6
+ un-denoised Neg	38.3	98.9
+ denoised Neg	38.7	98.9

dustry, as it saves enormous efforts and costs of deployment and increases the utility of computing resources. In future work, we will explore scaling up the model size to further improve DPTDR.

Acknowledgment

Benyou Wang is funded by the CUHKSZ startup funding No. UDF01002678.

References

Payal Bajaj, Daniel Campos, Nick Craswell, Li Deng, Jianfeng Gao, Xiaodong Liu, Rangan Majumder, Andrew McNamara, Bhaskar Mitra, Tri Nguyen, et al. 2016. Ms marco: A human generated machine reading comprehension dataset. *arXiv preprint arXiv:1611.09268*.

Tom B Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. 2020. Language models are few-shot learners. *arXiv preprint arXiv:2005.14165*.

Wei-Cheng Chang, Felix X Yu, Yin-Wen Chang, Yiming Yang, and Sanjiv Kumar. 2020. Pre-training tasks for embedding-based large-scale retrieval. *arXiv preprint arXiv:2002.03932*.

Zhuyun Dai and Jamie Callan. 2019. Deeper text understanding for ir with contextual neural language modeling. In *Proceedings of the 42nd International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 985–988.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*.

Luyu Gao and Jamie Callan. 2021a. Condenser: a pre-training architecture for dense retrieval. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 981–993.

Luyu Gao and Jamie Callan. 2021b. Unsupervised corpus aware language model pre-training for dense passage retrieval. *arXiv preprint arXiv:2108.05540*.

Yuxian Gu, Xu Han, Zhiyuan Liu, and Minlie Huang. 2021. Ppt: Pre-trained prompt tuning for few-shot learning. *arXiv preprint arXiv:2109.04332*.

Kelvin Guu, Kenton Lee, Zora Tung, Panupong Pasupat, and Ming-Wei Chang. 2020. Realm: Retrieval-augmented language model pre-training. *arXiv preprint arXiv:2002.08909*.

Neil Houlsby, Andrei Giurgiu, Stanislaw Jastrzebski, Bruna Morrone, Quentin De Laroussilhe, Andrea Gesmundo, Mona Attariyan, and Sylvain Gelly. 2019. Parameter-efficient transfer learning for nlp. In *International Conference on Machine Learning*, pages 2790–2799. PMLR.

Gautier Izacard, Mathilde Caron, Lucas Hosseini, Sebastian Riedel, Piotr Bojanowski, Armand Joulin, and Edouard Grave. 2021. Towards unsupervised dense information retrieval with contrastive learning. *arXiv preprint arXiv:2112.09118*.

Jeff Johnson, Matthijs Douze, and Hervé Jégou. 2019. Billion-scale similarity search with gpus. *IEEE Transactions on Big Data*.

Mandar Joshi, Eunsol Choi, Daniel S Weld, and Luke Zettlemoyer. 2017. Triviaqa: A large scale distantly supervised challenge dataset for reading comprehension. *arXiv preprint arXiv:1705.03551*.

Vladimir Karpukhin, Barlas Oğuz, Sewon Min, Patrick Lewis, Ledell Wu, Sergey Edunov, Danqi Chen, and Wen-tau Yih. 2020. Dense passage retrieval for open-domain question answering. *arXiv preprint arXiv:2004.04906*.

Tom Kwiatkowski, Jennimaria Palomaki, Olivia Redfield, Michael Collins, Ankur Parikh, Chris Alberti, Danielle Epstein, Illia Polosukhin, Jacob Devlin, Kenton Lee, et al. 2019. Natural questions: a benchmark for question answering research. *Transactions of the Association for Computational Linguistics*, 7:453–466.

Kenton Lee, Ming-Wei Chang, and Kristina Toutanova. 2019. Latent retrieval for weakly supervised open domain question answering. *arXiv preprint arXiv:1906.00300*.

Xiang Lisa Li and Percy Liang. 2021. Prefix-tuning: Optimizing continuous prompts for generation. *arXiv preprint arXiv:2101.00190*.

Pengfei Liu, Weizhe Yuan, Jinlan Fu, Zhengbao Jiang, Hiroaki Hayashi, and Graham Neubig. 2021a. Pre-train, prompt, and predict: A systematic survey of prompting methods in natural language processing. *arXiv preprint arXiv:2107.13586*.

Xiao Liu, Kaixuan Ji, Yicheng Fu, Zhengxiao Du, Zhilin Yang, and Jie Tang. 2021b. P-tuning v2: Prompt tuning can be comparable to fine-tuning universally across scales and tasks. *arXiv preprint arXiv:2110.07602*.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Roberta: A robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692*.

Yi Luan, Jacob Eisenstein, Kristina Toutanova, and Michael Collins. 2020. Sparse, dense, and attentional representations for text retrieval. *arXiv preprint arXiv:2005.00181*.

Yuning Mao, Pengcheng He, Xiaodong Liu, Yelong Shen, Jianfeng Gao, Jiawei Han, and Weizhu Chen. 2020. Generation-augmented retrieval for open-domain question answering. *arXiv preprint arXiv:2009.08553*.

Jianmo Ni, Chen Qu, Jing Lu, Zhuyun Dai, Gustavo Hernández Ábrego, Ji Ma, Vincent Y Zhao, Yi Luan, Keith B Hall, Ming-Wei Chang, et al. 2021. Large dual encoders are generalizable retrievers. *arXiv preprint arXiv:2112.07899*.

Rodrigo Nogueira, Jimmy Lin, and AI Epistemic. 2019. From doc2query to doctttttquery. *Online preprint*.

Barlas Oğuz, Kushal Lakhotia, Anchit Gupta, Patrick Lewis, Vladimir Karpukhin, Aleksandra Piktus, Xilun Chen, Sebastian Riedel, Wen-tau Yih, Sonal Gupta, et al. 2021. Domain-matched pre-training tasks for dense retrieval. *arXiv preprint arXiv:2107.13602*.

Yingqi Qu, Yuchen Ding, Jing Liu, Kai Liu, Ruiyang Ren, Wayne Xin Zhao, Daxiang Dong, Hua Wu, and Haifeng Wang. 2021. Rocketqa: An optimized training approach to dense passage retrieval for open-domain question answering. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 5835–5847.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J Liu. 2019. Exploring the limits of transfer learning with a unified text-to-text transformer. *arXiv preprint arXiv:1910.10683*.

Ruiyang Ren, Yingqi Qu, Jing Liu, Wayne Xin Zhao, Qiaoqiao She, Hua Wu, Haifeng Wang, and Ji-Rong Wen. 2021. Rocketqav2: A joint training method for dense passage retrieval and passage re-ranking. *arXiv preprint arXiv:2110.07367*.

Timo Schick and Hinrich Schütze. 2020. It's not just size that matters: Small language models are also few-shot learners. *arXiv preprint arXiv:2009.07118*.

Yu Sun, Shuohuan Wang, Yukun Li, Shikun Feng, Xuyi Chen, Han Zhang, Xin Tian, Danxiang Zhu, Hao Tian, and Hua Wu. 2019. Ernie: Enhanced representation through knowledge integration. *arXiv preprint arXiv:1904.09223*.

Tongzhou Wang and Phillip Isola. 2020. Understanding contrastive representation learning through alignment and uniformity on the hypersphere. In *International Conference on Machine Learning*, pages 9929–9939. PMLR.

Lee Xiong, Chenyan Xiong, Ye Li, Kwok-Fung Tang, Jialin Liu, Paul Bennett, Junaid Ahmed, and Arnold Overwijk. 2020. Approximate nearest neighbor negative contrastive learning for dense text retrieval. *arXiv preprint arXiv:2007.00808*.

BERT-Flow-VAE: A Weakly-supervised Model for Multi-Label Text Classification

Ziwen Liu
University College London
z.liu.19@ucl.ac.uk

Scott Allan Orr
University College London
scott.orr@ucl.ac.uk

Josep Grau-Bové
University College London
josep.grau.bove@ucl.ac.uk

Abstract

Multi-label Text Classification (MLTC) is the task of categorizing documents into one or more topics. Considering the large volumes of data and varying domains of such tasks, fully-supervised learning requires manually fully annotated datasets which is costly and time-consuming. In this paper, we propose BERT-Flow-VAE (BFV), a Weakly-Supervised Multi-Label Text Classification (WSMLTC) model that reduces the need for full supervision. This new model: (1) produces BERT sentence embeddings and calibrates them using a flow model, (2) generates an initial topic-document matrix by averaging results of a seeded sparse topic model and a textual entailment model that only require surface name of topics and 4-6 seed words per topic, and (3) adopts a VAE framework to reconstruct the embeddings under the guidance of the topic-document matrix. Finally, (4) it uses the means produced by the encoder model in the VAE architecture as predictions for MLTC. Experimental results on 6 multi-label datasets show that BFV can substantially outperform other baseline WSMLTC models in key metrics and achieve approximately 84% performance of a fully-supervised model.

1 Introduction

As vast numbers of written comments are posted daily on social media and e-commerce platforms, there is an increasing demand for methods that efficiently and effectively extract useful information from this unstructured text data. One of the methods to analyze this unstructured text data is to classify them into organized categories. This can be considered as a Multi-label Text Classification (MLTC) task since a single data may contain multiple non-mutually-exclusive topics (aspects). There are a range of relevant applications of this task such as categorizing movies by genres (Hoang, 2018), multi-label sentiment analysis (Almeida et al., 2018) and multi-label toxicity identification (Gunasekara and Nejadgholi, 2018).

Fully-supervised learning methods are undesirable for this task, because of the diversity of domains of application and cost of manual labelling (Brody and Elhadad, 2010). Seeded topic models, such as SeededLDA and CorEx (Jagarlamudi et al., 2012; Gallagher et al., 2017), where users can designate seed words as a prior to guide the models to find topics of interest, can be seen as a Weakly-Supervised Multi-Label Text Classification (WSMLTC) method. Nevertheless, as these models are mainly statistical models based on bag-of-words representation, they fail to fully exploit key sentence elements such as context and word positions. In contrast, large pre-trained language models such as BERT and GPT-3 (Devlin et al., 2018; Brown et al., 2020) produce contextualized embeddings for each word in a sentence, which has afforded them great success in the NLP field (Minaee et al., 2021; Ethayarajh, 2019).

Recently, prompt-based Few-Shot Learning (FSL) and Zero-Shot Learning (ZSL) methods (Yin et al., 2019, 2020; Gao et al., 2020) that take advantage of the general knowledge of large pre-trained language models can also approach MLTC tasks using only a few examples or topic surface names as a means of supervision. Specifically, these models convert text classification to a textual entailment task by preparing a template such as 'This example is about _' as input, and then estimating the probability of the model filling the blank with certain topic names. However, this method does not work well for abstract topics and there is no agreed way to use multiple words for the entailment task.

In this paper, we propose BERT-Flow-VAE (BFV), a WSMLTC model. It is based on the Variational AutoEncoder (VAE) (Kingma and Welling, 2013) framework to reconstruct the sentence embeddings obtained from distil-BERT (Sanh et al., 2019). Inspired by the work of (Li et al., 2020), we

Proceedings of the 29th International Conference on Computational Linguistics, pages 1203–1220
October 12–17, 2022.

use a shallow Glow (Kingma and Dhariwal, 2018) model to map the sentence embeddings to a standard Gaussian space before feeding them into the VAE model. Finally, we use the averaged results of a seeded sparse topic model and a ZSL model to guide our model to build latent variables towards pre-specified topics as predictions for MLTC.

Our contributions can be listed as follows: (1) We propose BFV, a WSMLTC model based on VAE framework, that can achieve comparable performance to a fully-supervised method on 6 multi-label datasets with only limited inputs (4 to 6 seed words per topic and surface name of topics). (2) We show that using a normalizing-flow model to calibrate sentence embeddings before feeding them into a VAE model can improve the model's MLTC performance, suggesting that pre-processing inputs is needed as it can better fit the overall objective of the VAE framework. (3) We present that the topic classification performance of ZSL method can be further improved by properly integrating predictions from a sparse seeded topic model, which complements the results from ZSL method by naturally incorporating multiple words to define a topic and could play a role of regularization.

2 Related Work

Seeded Topic Model Guided (seeded) topic models are built to find more desirable topics by incorporating users' prior domain knowledge. These seeded topic models can be seen as Weakly-Supervised (WS) methods to find specific topics in a corpus. Andrzejewski and Zhu (2009) proposed a model by using 'z-labels' to control which words appear or not appear in certain topics. Andrzejewski et al. (2009) presented DFLDA to construct Must-Link and Cannot-Link conditions between words to indirectly force the emergence of topics. Jagarlamudi et al. (2012) proposed SeededLDA to incorporate seed words for each topics to guide the results found by LDA. This is achieved by biasing (1) topics to produce seed words and (2) documents containing seed words to select corresponding topics. Gallagher et al. (2017) presented Correlation Explanation (CorEx), a model searching for topics that are 'maximally informative' about a set of documents. Seed words can be flexibly incorporated into the model during fitting. Meng et al. (2020a) proposed CatE that jointly embeds words, documents and seeded categories (topics) into a shared space. The category distinctive information is encoded during the process.

Weakly-supervised Text Classification Recently, Weakly-Supervised Text Classification (WSTC) has been rapidly developed (Meng et al., 2020b; Wang et al., 2020). Most of the works used pseudo labels/documents generation and self-training. Particularly, Meng et al. (2018) proposed WeSTClass which uses seed information such as label surface name and keywords to generate pseudo documents and refines itself via self-training. Mekala and Shang (2020) proposed ConWea that uses contextualized embeddings to disambiguate user input seed words and generates pseudo labels for unlabeled documents based on these words to train a text classifier. COSINE from (Yu et al., 2020) receives weak supervision and generates pseudo labels to perform contrastive learning (with confidence reweighting) to train a classifier. Some studies integrated simple rules as weak supervision signals: Ren et al. (2020) used rule-annotated weak labels to denoise labels, which then supervise a classifier to predict unseen samples; Karamanolakis et al. (2021) developed ASTRA that utilizes task-specific unlabeled data, few labeled data, and domain-specific rules through a student model, a teacher model and self-training. However, these WSTC methods were specifically designed for multi-class tasks and are not optimized for WSMLTC tasks in which documents could belong to multiple classes simultaneously.

Prompt-based Zero-Shot Learning MLTC tasks can also be approached with very limited supervision by Prompt-based Few-Shot Learning (FSL) or Zero-Shot Learning (ZSL). For example, Yin et al. (2019) proposed a ZSL method for text classification tasks by treating text classification as a textual entailment problem. This model treats an input text as a premise and prepares a corresponding hypothesis (template) such as 'This example is about _' for the entailment model. Finally, it uses the probability of the model filling the blank with topic names as the topic predictions. However, the choice of template and word for the entailment task requires domain knowledge and is often sub-optimal (Gao et al., 2020). Also, it is not straightforward to find multiple words as entailment for a topic. This may limit model's ability to understand abstract topics (e.g., 'evacuation' and 'infrastructure') where providing

a single surface name is insufficient (Yin et al., 2019). Although some automatic search strategies (Gao et al., 2020; Schick and Schütze, 2020; Schick et al., 2020) have been suggested, relevant research and applications are still under-explored.

3 Proposed Model: BERT-Flow-VAE

3.1 Problem Formulation and Motivation

Problem Formulation Multi-label text classification task is a broad concept, which includes many sub-fields such as eXtreme Multi-label Text Classification (XMTC), Hierarchical Multi-label Text Classification (HMTC) and multi-label topic modeling. In our model, instead of following these approaches, we follow a simpler assumption that the labels do not have a hierarchical structure and distribution of examples per label is not extremely skewed.

More precisely, given an input corpus consisting of N documents $\mathcal{D} = \{D_1, ...D_N\}$, the model assigns zero, single, or multiple labels to each document $D_i \in \mathcal{D}$ based on weak supervision signal from a dictionary of {topic surface name:keywords} $\mathcal{W}$ provided by user.

This is a more challenging task than multi-class text classification as samples are assumed to have non-mutually exclusive labels. This is a more practical assumption for text classification task because documents usually belong to more than one conceptual class (Tsoumakas and Katakis, 2007).

Motivation Inspired by relevant work of VAE and β-VAE (see Appendix A), we assume that the semantic information within sentence embeddings are composed of multiple disentangled factors in the latent space. Each latent factor can be seen as a label (topic) that may appear independently. Hence, we adopted VAE as our framework to approach this task.

3.2 Preparing the Inputs

Language Model and Sentence Embedding Strategy As we will model the latent factors from the semantic information of sentences encoded in the word embeddings, we need to firstly convert sentences into embeddings. Specifically, given the input corpus $\mathcal{D}$, we firstly process them into a collection of sentence embeddings $E_s \in \mathbb{R}^{N \times V}$, where V is the embedding dimension of the language model. Taking BERT as an example, there are two main ways to produce such sentence embeddings: (1) using the special token ($[CLS]$ in BERT) and (2) using a mean-pooling strategy to aggregate all word embeddings into a single sentence embedding. We tested and showed the performance of the two versions in section 5. Lastly, for computational efficiency, we used distil-BERT (Sanh et al., 2019) as our language model, which is a lighter version of BERT with comparable performance.

Moreover, instead of simply averaging the embeddings of words in a sentence with equal weights, we also tested a TF-IDF averaging strategy. Specifically, we firstly calculated the weights of words in a sentence using the TF-IDF algorithm with L_2 normalization, and then averaged the words according to the TF-IDF weights. To avoid weights of some common words to be nearly zero, we combined 10% mean pooling weights and 90% TF-IDF pooling weights as the final embeddings.

Flow-calibration Sentence embeddings obtained from BERT without extra fine-tuning have been found to poorly capture the semantic meaning of sentences. This is reflected by the performance of BERT on sentence-level tasks such as predicting Semantic Textual Similarity (STS) (Reimers and Gurevych, 2019). This may be caused by anisotropy (embeddings occupy a narrow cone in the vector space), a common problem of embeddings produced by language models (Ethayarajh, 2019; Li et al., 2020). To address this problem, following the work of (Li et al., 2020), we adopted BERT-Flow to calibrate the sentence embeddings. More exactly, we used a shallow Glow (Kingma and Dhariwal, 2018) with K = 16 and L = 1, a normalizing-flow based model, with random permutation and affine coupling to post-process the sentence embeddings from all 7 layers of distil-BERT (including the word embedding layer). We tested different combinations of the 7 post-processed embeddings and took the average of embeddings from the first, second and sixth layer based on the metrics evaluated on the STS benchmark dataset.

Since normalizing-flow based models can create an invertible mapping from the BERT embedding space to a standard Gaussian latent space (Li et al., 2020), the advantages of using flow calibration are: (1) it improves the anisotropy to make the sentence embeddings more semantically distinguishable, and (2) it converts the distribution of BERT embeddings to be standard Gaussian, which fits the

objective of minimizing mean-squared reconstruction error and Kullback–Leibler Divergence (KLD) with a standard Gaussian prior distribution in the following VAE model.

Backend Model To guide our model towards some pre-specified topics, we used Zero-Shot Text Classification method (0SHOT-TC) proposed by (Yin et al., 2019) as the backend model. Specifically, we used RoBERTa-large (Liu et al., 2019) as the language model for 0SHOT-TC. Following the example mentioned previously, we prepared a template (hypothesis) with the shape 'This example is about _' for each sentence (premise) and filled the blank with the surface name of topics. Finally, we took the probability of *entailment* as that of the topic appearing in the sentence for each class and collected this as $\mathcal{T}_{0SHOT-TC} \in \mathbb{R}^{N\times M}$, where M is the number of topics.

However, because current zero-shot learning methods lack an agreed way to find multiple words as entailment for a topic, we further used a seeded topic model as a complement. More exactly, we selected Anchored Correlation Explanation (CorEx) (Gallagher et al., 2017) as another backend model. By following the approach used by (Jagarlamudi et al., 2012; Gallagher et al., 2017), we randomly chose 4 to 6 seed words from the top 20 most discriminating words of each topic as seed words to better simulate real-world applications. Finally, we estimated unnormalized document-topic matrix $\mathcal{T}_{CorEx} \in \mathbb{R}^{N\times M}$ and took the combination:

$$\mathcal{T} = \omega \times \mathcal{T}_{0SHOT-TC} + (1-\omega) \times \mathcal{T}_{CorEx}$$

where ω is the combination weight. We set $\omega = 0.5$ herein (details will be discussed in section 5.2).

3.3 Model Description

Model Architecture and Objective Function An overview of the model architecture can be seen in Fig 1. Specifically, we used fully connected layers combined with layer normalization (Ba et al., 2016) and Parametric ReLU (PReLU) (He et al., 2015). The encoder model q_ϕ receives flow-calibrated sentence embeddings E_s and outputs mean ($\mu \in \mathbb{R}^{N\times M}$) and variance ($\sigma \in \mathbb{R}^{N\times M}$) which will be the inputs to the decoder model p_θ to produce reconstructed sentence embeddings $\hat{E}_s \in \mathbb{R}^{N\times V}$.

As in the vanilla VAE model in Appendix A, the objective function of our model contains a reconstruction loss and KLD loss. We used the mean-squared error between E_s and $\hat{E}_s$ as the reconstruction loss because input embeddings have been calibrated to have a standard Gaussian distribution, and used the KL divergence between the output (μ and σ) of the encoder and the prior $\mathcal{N}(0, I)$ as the KLD loss. In addition, in order to guide the model's direction towards the pre-specified topics, we added another loss term dubbed topic loss:

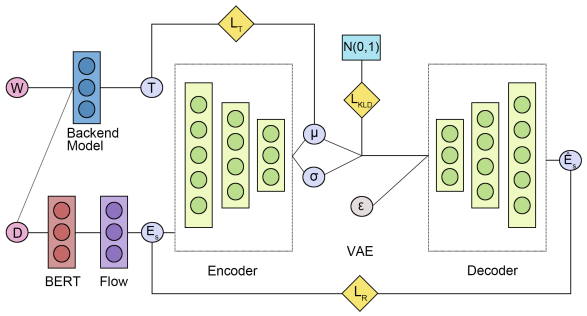

Figure 1: Architecture of the proposed model.

$$L_\mathcal{T} = -\frac{1}{NM}\sum_{i}^{N}\sum_{j}^{M}\mathcal{T}_{ij} \cdot log(sigmoid(\mu_{ij}))$$

where $sigmoid(\cdot)$ is the element-wise sigmoid function. $L_\mathcal{T}$ is the binary cross-entropy between μ and $\mathcal{T}$ to encourage μ to be closer to $\mathcal{T}$.

Notice that the value of $sigmoid(\mu) \in \mathbb{R}^{N\times M}$ produced by the encoder can be viewed as a document-topic matrix. Thus, we used it as the model's prediction for MLTC. $sigmoid(\mu_{ij}) > 0.5$ is be predicted as positive (i.e., topic j appears in ith document). σ is be left as free values to reconstruct $\hat{E}_s$.

As shown by (Higgins et al., 2016; Burgess et al., 2018), the weight of different components in the objective function of the VAE model is important to find disentangled representations. In particular, based on our observations, the ratio between L_{KLD} and $L_\mathcal{T}$ is be crucial. Hence, we set a hyper-parameter γ in the objective function controlling the ratio of L_{KLD} and $L_\mathcal{T}$. Finally, the objective function of our VAE model is:

$$\mathcal{L}(\theta, \phi; E_s, \mathcal{T}, \alpha, \eta) = -(L_R + \alpha L_{KLD} + \eta L_\mathcal{T})$$

where $\alpha = 0.1 \times \sqrt{\gamma}$ and $\eta = 0.1 \times \gamma M$. It can be seen that a higher value of γ will lead to a heavier penalty on $L_\mathcal{T}$, and therefore μ will become more similar to $\mathcal{T}$; while, conversely, a lower value of γ will make μ diverge from $\mathcal{T}$. As L_{KLD} pushes

$sigmoid(\mu)$ towards 0.5, a lower value of γ would encourage model to make aggressive predictions.

Hyper-Parameter Scheduling (HPS) We adopted the strategy to gradually change the hyper-parameters of the model. We used the first epoch of training as a warm-up stage (Sønderby et al., 2016) by setting the value of α to one-tenth of its original value. Similarly, we also halved the value of η in the last epoch of training, aiming to reduce the dependency of μ on $\mathcal{T}$.

4 Experiments Setup

4.1 Datasets

We chose *Restaurant* and *Laptop* datasets from SemEval 2014 Task 4 (Pontiki et al., 2014) as well as *CitySearch* (Ganu et al., 2009), which are very popular in aspect-based sentiment analysis studies. We also added *SentiHood* (Saeidi et al., 2016), a similar dataset to the SemEval datasets. Additionally, *Reuters* (ApteMod version) (Apt'e et al., 1994), a well-known text classification dataset containing Reuters financial newswire service in 1987 was also selected. Finally, given that the Restaurants, CitySearch and SentiHood are all largely related to food-related aspects, we added a self-made dataset called *Heritage*, which is composed of 3,760 online reviews for 77 heritage sites. There are 9 categories in this dataset: *heritage, exhibition, price, family, service, transport, facilities, environment* and *miscellaneous* (see Appendix D).

4.2 Data Pre-processing

Pre-processing For all datasets, we only kept categories which have at least 30 corresponding samples or account for at least 1% in the whole dataset. Also, we removed categories with no specific meaning (e.g., 'miscellaneous'). Finally, for datasets which do not have a pre-specified train-test split, we used 20% of the examples as the test set (splits were carefully conducted to account for class balance). Table 1 shows detailed information on the datasets after pre-processing.

Model Fine-tuning and Training We finetuned distil-BERT before using it to produce embeddings. Specifically, we used a learning rate of 1×10^{-5} for the word embedding layer and top 3 layers of distil-BERT, and 1×10^{-3} for last 3 layers. The weight decay was set to 0 for the bias and layer normalization weights. We used a warm-up strategy to gradually unfreeze the parameters of distil-BERT. We ran the fine-tuning process for 5 epochs on each dataset separately. Additionally, we trained the Glow model with a learning rate of 1×10^{-3} for 5 epochs. Lastly, we trained the BFV model with a learning rate of 1×10^{-3} for 10 epochs, with weight decay set using the same values as the distil-BERT fine-tuning. AdamW (Loshchilov and Hutter, 2017) is the optimizer used for training all three models.

4.3 Evaluation Procedure

Evaluation Metrics One of the limitations of this model is that, as we only reconstruct the pooled representation of the sentences rather than words, there is no explicit modelling of word-topic relationship. Therefore, metrics for topic models such as perplexity and topic coherence cannot be directly measured. Thus, they will not be reported. We calculated the macro-average of each class for the metrics defined only for binary predictions such as F1-score. More detailed definition of metrics can be found in Appendix E. We report the average and standard deviation over 10 runs for our model and its ablated versions.

Baseline Models In this paper, we only compare our model with methods that can perform multi-label prediction with weak supervision (only surface names and keywords are provided). To the best of the authors' knowledge, there are few methods built specifically for the WSMLTC tasks. Therefore, we consider guided topic models and prompting based zero-shot learning language models as the work most closely related to our model as baselines. For weakly-supervised methods that were specifically designed for multi-class tasks, we compare our model with them in Appendix B since we converted them to perform MLTC tasks which may cause them to perform differently from their original multi-class design.

We used backend models (**CorEx**, **0SHOT-TC** and **CorEx+0SHOT-TC**) and **Guided LDA (GLDA)**, a standard LDA with initial word-topic priors biased towards seeded topics, as baseline models to compare the performance of our models. Following the work in (Brody and Elhadad, 2010), we used a POS-tagging model[1] to only keep nouns and adjectives for GLDA and CorEx. We also included ablated versions of BFV:

BERT-Whitening-VAE (BWV): The Flow in

[1] huggingface.co/vblagoje/bert-english-uncased-finetuned-pos

Dataset	Number of samples	Number of classes	Max number of data per class	Min number of data per class	Multilabel ratio	Average text length
Reuters	10788	42	3964	31	0.14	128
CitySearch	3315	4	1227	298	0.11	14
Sentihood	5215	9	714	143	0.10	15
Restaurant	3844	4	1651	402	0.16	13
Laptop	3308	5	639	175	0.12	13
Heritage	3760	8	623	156	0.11	18

Table 1: Metadata for 6 selected datasets after pre-processing. "Multilabel ratio" is the percentage of samples having more than one label.

BFV is replaced by a simple whitening operation introduced by (Huang et al., 2021).

BERT-VAE (BV): No calibration is performed to the BERT-embeddings.

BERT-Flow-VAE-CLS (BFVCLS): Instead of using pooling average strategy, $[CLS]$ token is used as the representation of a sentence.

BERT-Flow-Encoder (BFE): Only the encoder part of the VAE is kept, and two loss components (L_R and L_{KLD}) are disabled.

BERT-Flow (BF): The VAE part in BFV is replaced by a textual entailment classification header [2] with the same template. The classification header has been finetuned on the MNLI dataset to adjust for the flow-calibrated model.

0SHOT-TC-Flow: Flow calibration is applied to the original 0SHOT-TC model. The classification header has been finetuned on the MNLI dataset to adjust for the flow-calibrated model.

Lastly, to show the performance gap between our model and fully-supervised methods, the performance of the **BFE** trained in a fully-supervised way using groundtruth labels (**BFE-Sup**) is included.

5 Results

5.1 Quantitative Evaluation

Table 2 presents the quantitative evaluation of the models measured using classification metrics (clustering performance and qualitative evaluation can be found in Appendix C.). We make the following observations based on these results:

(1) BFV outperforms other WSMLTC models in most datasets with relatively large margin except for *Reuters*, in terms of F1-score and APS.

(2) In terms of the F1-score, BFV with only seed words and surface name of topics as supervision can, on average, achieve approximately 84% performance of the fully-supervised model as reflected in Table 3. In addition, the performance gap between weakly-supervised and fully-supervised model is narrow in datasets of social media reviews. This suggests that the weakly-supervised models are able to find topics with limited guidance in the context of short text, informal and polysemous words that characterise reviews.

	% of BFE-Sup's F1
BFE-Sup	100.0 (100.0)
BFV	84.38 (83.40)
BWV	78.67 (77.49)
BV	73.47 (71.28)
BFVCLS	58.93 (56.10)
BFE	69.15 (67.46)
BF	26.57 (25.50)
0SHOT-TC+CorEx	57.84 (53.97)
0SHOT-TC-Flow	33.42 (33.56)
0SHOT-TC	69.57 (71.53)
CorEX	53.86 (50.25)
GLDA	27.84 (27.37)

Table 3: Models performance in percentage of the fully-supervised model measured by F1-score averaged across all datasets. Numbers in the brackets are F1-score averaged across 5 datasets excluding *Heritage*.

(3) After replacing Flow with a simple whitening operation, there is no significant drop of performance of the model in terms of key metrics such as F1, demonstrating the robustness of the model.

Table 4 shows examples of prediction results in the format of document-topic matrix, where each value shows the probability of the topic (column) appearing in the document (row).

5.2 Ablation Study and Sensitivity Analysis

In order to investigate further the effectiveness of each component and the sensitivity with respect to backend models, we sequentially added each component of the model and calculated its perfor-

[2] huggingface.co/huggingface/distilbert-base-uncased-finetuned-mnli

		ACC	HS	P@3	**F1**	Recall	Precision	**APS**	**AUC**
	BFE-Sup (20)	75.52	80.82	82.86	63.29	54.12	83.00	70.18	97.91
	BFV (20)	44.65 (1.33)	55.74 (1.35)	66.03 (1.5)	**44.99** (0.51)	55.38 (0.6)	**51.58** (0.84)	56.18 (0.34)	**95.86** (0.14)
	BWV (20)	39.83 (1.23)	51.33 (1.23)	61.98 (1.2)	44.59 (0.41)	54.96 (0.5)	50.27 (0.76)	55.96 (0.36)	95.85 (0.1)
	BV (20)	48.84 (2.5)	**57.72** (2.61)	65.40 (2.88)	40.91 (0.73)	45.63 (1.65)	50.56 (1.71)	51.12 (0.7)	95.58 (0.11)
	BFVCLS (20)	**49.60** (4.92)	56.85 (5.15)	62.53 (5.51)	33.97 (1.84)	37.08 (3.32)	44.19 (1.57)	46.23 (0.54)	94.37 (0.11)
	BFE (20)	45.09 (0.74)	53.54 (0.78)	60.37 (0.75)	37.56 (0.36)	39.81 (0.35)	50.13 (1.1)	44.21 (0.39)	94.07 (0.09)
Reuters	BF	1.82	3.35	7.96	3.92	33.51	2.72	3.12	43.86
	0SHOT-TC+CorEx	12.62	25.61	49.30	25.28	60.56	21.43	50.28	93.91
	0SHOT-TC-Flow	0.43	4.76	11.39	5.22	58.01	3.31	3.81	54.94
	0SHOT-TC	19.68	43.19	**75.26**	44.55	**70.65**	38.21	**56.22**	91.64
	CorEx	10.53	22.84	31.56	24.27	57.53	20.71	17.90	86.71
	GLDA	7.92	8.28	8.28	6.88	4.15	34.33	26.33	75.18
	BFE-Sup (1)	71.79	75.72	76.66	72.66	64.81	83.13	82.30	94.37
	BFV (1)	**61.07** (1.25)	**68.95** (0.93)	**72.79** (0.75)	**69.48** (0.96)	76.50 (1.33)	64.92 (1.35)	**77.25** (0.42)	**92.37** (0.22)
	BWV (1)	54.60 (1.38)	61.07 (1.39)	62.87 (1.51)	62.63 (1.53)	60.99 (2.92)	66.13 (1.92)	69.98 (0.69)	89.19 (0.26)
	BV (1)	53.74 (0.99)	58.51 (1.12)	59.34 (1.25)	59.64 (2.2)	52.24 (5.07)	75.06 (3.06)	72.41 (0.47)	89.09 (0.25)
	BFVCLS (1)	54.39 (1.34)	58.86 (1.37)	59.08 (1.38)	48.82 (1.61)	39.67 (2.3)	**79.56** (2.76)	67.09 (0.4)	87.97 (0.2)
	BFE (1)	53.26 (0.68)	57.90 (0.79)	58.33 (0.79)	54.28 (1.25)	43.84 (1.09)	78.08 (1.62)	69.89 (0.84)	88.63 (0.46)
CitySearch	BF	29.41	33.74	33.81	18.80	21.09	19.04	19.39	48.41
	0SHOT-TC+CorEx	47.36	51.43	51.92	39.98	30.83	58.42	63.75	87.80
	0SHOT-TC-Flow	37.86	46.10	50.35	41.68	51.13	40.97	45.08	77.08
	0SHOT-TC	41.78	53.12	61.29	60.43	**86.90**	48.30	71.04	88.12
	CorEx	45.70	49.57	50.04	35.66	26.44	55.57	43.89	69.97
	GLDA	33.18	35.97	35.97	27.42	23.55	35.92	30.63	65.73
	BFE-Sup (1)	74.04	77.41	78.14	68.88	61.32	79.05	76.60	96.81
	BFV (1)	42.94 (1.57)	51.23 (1.37)	56.35 (1.25)	**53.54** (0.88)	75.58 (1.13)	47.21 (0.96)	59.10 (0.68)	**92.12** (0.14)
	BWV (1)	50.29 (1.73)	55.61 (1.51)	57.80 (1.36)	51.40 (1.36)	60.18 (2.32)	53.49 (0.71)	59.80 (0.33)	91.96 (0.13)
	BV (1)	51.41 (3.61)	56.66 (3.14)	**59.00** (2.67)	51.02 (1.66)	53.77 (2.94)	**59.25** (1.04)	**60.78** (0.48)	91.93 (0.22)
	BFVCLS (1)	**53.62** (2.56)	56.13 (2.42)	56.75 (2.17)	32.92 (1.28)	31.04 (2.14)	55.77 (2.0)	48.80 (0.58)	89.20 (0.14)
	BFE (1)	52.60 (0.45)	**56.88** (0.4)	58.63 (0.44)	46.55 (0.87)	46.83 (0.79)	56.58 (1.09)	56.12 (0.44)	90.46 (0.21)
Sentihood	BF	20.66	24.32	25.57	10.81	46.58	6.42	6.76	51.86
	0SHOT-TC+CorEx	53.05	55.90	56.42	39.17	39.50	45.05	53.26	91.29
	0SHOT-TC-Flow	0.60	6.82	10.02	11.37	64.15	6.42	9.03	48.18
	0SHOT-TC	4.16	18.33	36.82	39.13	**89.11**	28.16	55.30	91.12
	CorEx	52.52	54.98	55.34	34.71	33.82	40.76	33.00	75.87
	GLDA	39.17	40.39	40.40	16.41	15.43	20.58	16.70	71.18
	BFE-Sup (1)	75.38	80.93	81.72	81.59	75.57	89.11	90.02	95.87
	BFV (1)	**68.85** (1.95)	**74.42** (2.02)	**76.34** (2.16)	**80.49** (0.77)	80.58 (1.72)	80.88 (2.24)	**89.73** (0.4)	**95.21** (0.27)
	BWV (1)	58.09 (1.91)	65.65 (1.76)	68.13 (1.97)	73.44 (1.35)	80.02 (3.47)	69.62 (3.21)	85.55 (0.35)	93.34 (0.17)
	BV (1)	53.66 (2.18)	59.43 (2.36)	60.27 (2.63)	69.52 (1.97)	60.85 (4.55)	**84.74** (3.94)	84.97 (0.45)	91.78 (0.33)
	BFVCLS (1)	57.84 (2.94)	64.95 (2.83)	66.02 (2.61)	60.14 (2.89)	53.00 (2.45)	83.62 (8.77)	78.28 (1.11)	90.09 (0.34)
	BFE (1)	57.36 (1.01)	63.38 (0.9)	64.11 (0.86)	69.05 (1.45)	60.35 (1.42)	83.59 (1.18)	81.57 (0.59)	91.21 (0.24)
Restaurant	BF	8.38	21.33	24.74	30.21	51.64	25.00	25.62	49.18
	0SHOT-TC+CorEx	46.00	51.12	51.54	57.02	45.92	78.68	83.06	93.40
	0SHOT-TC-Flow	4.88	28.45	47.24	38.44	88.02	26.49	30.58	60.69
	0SHOT-TC	37.25	56.46	72.90	63.04	**95.00**	48.61	83.65	92.43
	CorEx	45.25	50.19	50.31	54.42	43.00	77.55	61.20	74.97
	GLDA	26.12	31.08	31.08	34.46	30.93	46.52	42.51	72.39
	BFE-Sup (2)	60.15	63.48	63.90	49.24	39.87	65.76	57.66	87.21
	BFV (2)	45.72 (1.18)	49.37 (1.18)	50.53 (1.2)	**36.40** (1.2)	38.59 (1.54)	42.75 (1.67)	**39.25** (0.54)	**73.38** (0.49)
	BWV (2)	45.04 (0.57)	47.96 (0.37)	48.76 (0.42)	32.58 (1.15)	32.34 (1.46)	44.62 (0.84)	37.56 (0.31)	72.72 (0.18)
	BV (2)	51.18 (2.11)	52.82 (1.86)	53.16 (1.7)	24.82 (1.71)	21.52 (2.59)	52.31 (2.46)	37.21 (0.49)	73.28 (0.33)
	BFVCLS (2)	**52.46** (0.97)	**53.59** (0.86)	**53.79** (0.86)	18.78 (1.56)	14.36 (1.04)	**54.01** (3.23)	35.23 (0.65)	72.30 (0.36)
	BFE (2)	49.15 (0.63)	50.96 (0.6)	51.30 (0.55)	25.14 (0.5)	22.35 (0.74)	52.14 (3.68)	35.96 (0.69)	71.90 (0.36)
Laptop	BF	9.90	18.32	21.60	21.03	**68.05**	12.85	14.60	54.14
	0SHOT-TC+CorEx	48.76	50.51	50.56	23.71	22.39	35.82	33.91	71.89
	0SHOT-TC-Flow	13.00	20.07	23.24	18.99	44.92	12.97	13.59	53.47
	0SHOT-TC	14.98	25.13	31.76	34.49	64.18	27.10	34.78	70.96
	CorEx	48.39	50.09	50.12	23.00	21.89	34.69	23.64	60.20
	GLDA	43.19	43.79	43.79	10.91	9.73	17.03	16.56	60.99
	BFE-Sup (5)	53.44	56.59	56.81	49.52	39.42	67.95	58.86	89.85
	BFV (5)	40.10 (1.41)	47.23 (1.36)	**51.42** (1.5)	**44.20** (1.22)	50.34 (2.11)	41.75 (0.85)	**46.68** (0.56)	**81.55** (0.32)
	BWV (5)	33.75 (0.98)	42.22 (0.77)	47.29 (0.85)	41.89 (0.64)	51.99 (2.07)	37.30 (0.95)	45.08 (0.49)	79.90 (0.21)
	BV (5)	**42.37** (1.01)	**47.77** (1.11)	50.26 (1.4)	41.81 (1.49)	41.18 (2.85)	46.84 (1.39)	45.33 (0.25)	81.28 (0.25)
	BFVCLS (5)	41.58 (1.61)	46.34 (1.26)	48.61 (1.17)	36.18 (1.25)	33.39 (1.94)	**47.93** (3.81)	40.75 (0.46)	80.81 (0.15)
	BFE (5)	40.05 (1.04)	45.65 (0.73)	48.60 (0.59)	38.41 (0.72)	37.44 (0.88)	42.86 (0.74)	40.15 (0.46)	80.86 (0.34)
Heritage	BF	4.41	11.06	13.44	15.81	67.63	9.22	9.65	49.20
	0SHOT-TC+CorEx	41.50	44.95	46.10	38.21	36.36	44.89	40.32	78.10
	0SHOT-TC-Flow	0.39	9.17	12.86	16.21	**76.19**	9.29	9.46	49.09
	0SHOT-TC	9.34	24.08	40.43	29.59	73.51	19.37	33.76	75.55
	CorEx	41.25	44.27	44.87	35.61	32.67	43.58	31.60	71.87
	GLDA	24.64	25.14	25.14	14.95	13.43	19.30	18.58	66.81

Table 2: Results for classification performance. Definition of metrics can be found at Appendix E. Highest values are marked with bold font. Numbers in the brackets are γ values (for model name) or standard deviation (for metrics). All numbers are percentages. γ values are chosen based on the observation of the aggressiveness of the model.

	ambience	service	price	food
The wait staff is friendly, and the food has gotten better and better! ('food', 'service')	0.43	0.69	0.37	0.73
The staff is unbelievably friendly, and I dream about their Saag gosht...so good. ('service', 'food')	0.43	0.71	0.36	0.51
The crust is thin, the ingredients are fresh and the staff is friendly. ('food', 'service')	0.47	0.69	0.30	0.68
The food is outstanding and the service is quick, friendly and very professional. ('food', 'service')	0.47	0.88	0.30	0.74
Get the soup and a nosh (pastrami sandwich) for $8 and you're golden. ('food', 'price')	0.36	0.42	0.52	0.53
Wonderful menu, warm inviting ambiance, great service the FOOD keeps me coming back! ('food', 'ambience', 'service')	0.61	0.69	0.32	0.68
The food was good, the service prompt, and the price very reasonable. ('food', 'service', 'price')	0.42	0.80	0.76	0.81
Great food at REASONABLE prices, makes for an evening that can't be beat! ('food', 'price')	0.47	0.35	0.78	0.72
While the food was excellent, it wasn't cheap (though not extremely expensive either). ('food', 'price')	0.34	0.32	0.69	0.75
I found the food, service and value exceptional everytime I have been there. ('food', 'service', 'price')	0.33	0.68	0.43	0.65

Table 4: An exemplar of document-topic matrix using sentences selected from *Restaurant* dataset, where rows represent documents and columns represent topics. Words in brackets after the document show the corresponding groudtruth label.

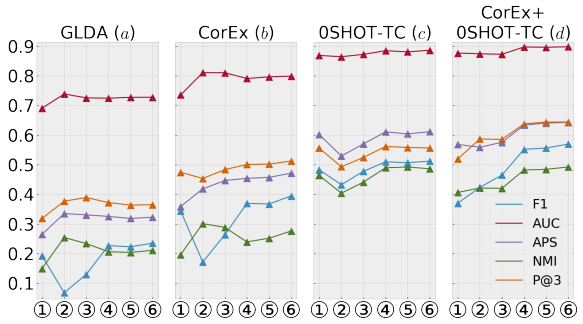

Figure 2: Ablation study with respect to various backend models and components, averaged across 5 multi-label datasets excluding *Heritage* dataset. Title: different backend models. X-axis: ①: Backend model only; ②: ①+BERT+encoder; ③: ②+Flow; ④: ③+VAE; ⑤: ④+TF-IDF; ⑥: ⑤+HPS. NMI: Normalized Mutual Information

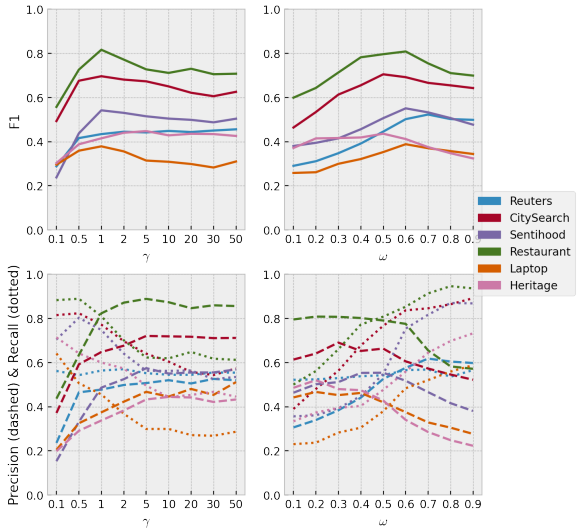

Figure 3: Sensitivity with respect to γ and ω in terms of F1 (solid line), Precision (dashed line) and Recall (dotted line) on 6 datasets

mance by averaging several key metrics across 5 multi-label (excluding *Heritage*) datasets in Fig 2.

The results shown in Fig 2 suggest that there is a consistent improvement of BFV models (at stage ⑥) with respect to various backends (at stage ①) in terms of F1, manifesting the generalizability of the benefits brought by BFV. Furthermore, after adopting the flow model and VAE in step ③ and step ④, most metrics outperform that of the base models. This indicates the effectiveness of Flow-calibrated embeddings and extra two losses (L_R and L_{KLD}) brought by VAE. On the other hand, the TF-IDF averaging strategy (⑤) does not result in significant effects on model performance. Also, the HPS strategy (⑥) does yield performance gains when GLDA and CorEx as backend models, but has limited effects on 0SHOT-TC and CorEx+0SHOT-TC backend models.

We also conducted a sensitivity analysis with respect to γ and ω in Fig 3. It could be observed that the model performs best when γ is between 0.5 and 5 and ω is between 0.5 and 0.6 for nearly all datasets. From the bottom two plots in Fig 3 we can see that, γ has a positive (negative) relationship with Precision (Recall). This relationship is reversed when it comes to ω. This makes sense because γ controls the relative weight of L_{KLD} and $L_{\mathcal{T}}$ in the loss function, and ω controls the weight of the combination of two backend models. Therefore, they could influence the aggressiveness of the model. This can be used to control specificity and sensitivity of the model (with default value of $\gamma = 1$ and $\omega = 0.5$).

5.3 Discussion

Effectiveness of Flow and VAE Based on the performance of VAE-ablated models (**BFE**, **BF** and **0SHOT-TC-Flow**) and Flow-ablated models (**BV** and **BWV**) shown in Table 2 and Figure 2, we can observe that, neither Flow nor VAE alone can outperform their combination (**BFV**) in terms of

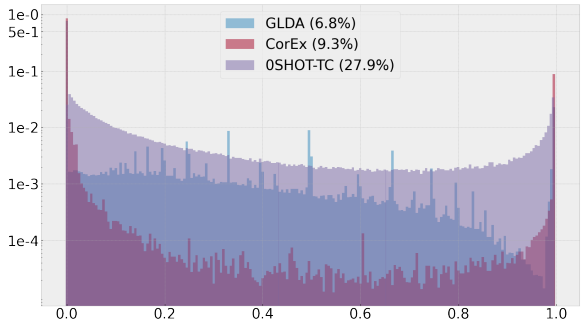

Figure 4: Normalized histogram of all predicted probabilities (x-axis) by backend models. The numbers in the bracket are the percentage of positive predictions.

F1-score and APS. This reflects the importance of combining Flow and VAE to model disentangled latent variables within calibrated BERT embeddings.

Mixture of Backend Except for the benefits brought by the pre-trained language model and embeddings calibration, we also suggest that BFV can learn from both a sparser and conservative model (CorEx) and a denser and aggressive model (0SHOT-TC) simultaneously:

(1) BFV can capture complementary information from seed words in the results of CorEx in addition to the topic surface name. Specifically, Fig 2 shows that the average of the results of CorEx and 0SHOT-TC has a lower value of F1-score (① of d) compared to that of only 0SHOT-TC (① of c). Note that this still holds even if we compare F1-score in ③ of d and ① of c. However, after being fully processed by BFV, the F1-score in ⑥ of d is significantly large compared to that of using only 0SHOT-TC (⑥ of c) as backend.

(2) Results from CorEx may provide regularization. We observe in Fig 4 that, results from CorEx are sparser and conservative (most labels were predicted to be negative) than that of 0SHOT-TC. We also observe from Table 2 that CorEx has a higher Precision than GLDA (and thus GLDA is not mixed) and 0SHOT-TC in general.

Different Language Models When replacing the backend language model (distil-BERT) with BERT, RoBERTa and XLM-R, in terms of the F1-score over 10 runs averaged across the 6 datasets, we found: BERT-base (54.35%), BERT-large (54.54%), RoBERTa-base (55.47%), RoBERTa-large (55.96%) and XLM-R-base (54.50%) compared to distil-BERT of 54.85% (detailed dataset-specific results fan be found in Appendix C). However, we simply tested these models using the same hyper-parameters, which may cause impacts on the performance. We didn't test the effects of different combinations of language models and hyper-parameters because the potential amount of computation will be very large for grid search. Thus, it is difficult to draw conclusions at this stage and we propose to leave in-depth analysis as future work where different characteristics of the backend language model should be thoroughly considered.

6 Conclusion

We presented BFV, a WSMLTC model, that uses a VAE framework to reconstruct BERT-produced and flow-calibrated sentence embeddings under the guidance of the averaged results of CorEx and 0SHOT-TC. It can significantly outperform other WSMLTC models in key metrics and achieve a approximately 84% performance of a fully-supervised model in terms of macro F1-score evaluated on 6 datasets. We found the improvements are mainly due to: (1) combining BERT and VAE framework, (2) mapping the sentence embeddings into a standard Gaussian space to better fit the overall objective of the VAE framework and (3) learning simultaneously from the results produced by a mixture of backends. As the input of surface name of topics and seed words are only used to produce $\mathcal{T}$, BFV can be viewed as a post-processing model to refine an already made document-topic matrix.

One limitation of the current BFV model is that it does not explicitly model the relationship between topics and words. Thus, relevant tasks such as calculating topic coherence and selecting keywords for each topic cannot be done directly. Another drawback of the BFV model is that it does not take into account the dependencies and hierarchies within topics. This may limit the model's performance for datasets in which labels are correlated.

In the future, BFV could be improved by reconstructing embeddings of each words in a sentence rather than an embedding of a sentence as a whole. This can also potentially expand this method into an applicable generative model for real sentence generation. Further work could also evaluate the performance of the model using embeddings made by language models other than distil-BERT.

Acknowledgements

We would like to thank the reviewers for their constructive feedback. We would also want to thank Dr Pontus Stenetorp for his helpful advice.

References

Alex MG Almeida, Ricardo Cerri, Emerson Cabrera Paraiso, Rafael Gomes Mantovani, and Sylvio Barbon Junior. 2018. Applying multi-label techniques in emotion identification of short texts. *Neurocomputing*, 320:35–46.

David Andrzejewski and Xiaojin Zhu. 2009. Latent dirichlet allocation with topic-in-set knowledge. In *Proceedings of the NAACL HLT 2009 Workshop on Semi-Supervised Learning for Natural Language Processing*, pages 43–48.

David Andrzejewski, Xiaojin Zhu, and Mark Craven. 2009. Incorporating domain knowledge into topic modeling via dirichlet forest priors. In *Proceedings of the 26th annual international conference on machine learning*, pages 25–32.

Chidanand Apt'e, Fred Damerau, and Sholom M. Weiss. 1994. Automated learning of decision rules for text categorization. *ACM Transactions on Information Systems*.

Jimmy Lei Ba, Jamie Ryan Kiros, and Geoffrey E Hinton. 2016. Layer normalization. *arXiv preprint arXiv:1607.06450*.

Samuel Brody and Noemie Elhadad. 2010. An unsupervised aspect-sentiment model for online reviews. In *Human language technologies: The 2010 annual conference of the North American chapter of the association for computational linguistics*, pages 804–812.

Tom B Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. 2020. Language models are few-shot learners. *arXiv preprint arXiv:2005.14165*.

Christopher P Burgess, Irina Higgins, Arka Pal, Loic Matthey, Nick Watters, Guillaume Desjardins, and Alexander Lerchner. 2018. Understanding disentangling in β-vae. *arXiv preprint arXiv:1804.03599*.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*.

Kawin Ethayarajh. 2019. How contextual are contextualized word representations? comparing the geometry of bert, elmo, and gpt-2 embeddings. *arXiv preprint arXiv:1909.00512*.

Ryan J Gallagher, Kyle Reing, David Kale, and Greg Ver Steeg. 2017. Anchored correlation explanation: Topic modeling with minimal domain knowledge. *Transactions of the Association for Computational Linguistics*, 5:529–542.

Gayatree Ganu, Noemie Elhadad, and Amélie Marian. 2009. Beyond the stars: improving rating predictions using review text content. In *WebDB*, volume 9, pages 1–6. Citeseer.

Tianyu Gao, Adam Fisch, and Danqi Chen. 2020. Making pre-trained language models better few-shot learners. *arXiv preprint arXiv:2012.15723*.

Isuru Gunasekara and Isar Nejadgholi. 2018. A review of standard text classification practices for multi-label toxicity identification of online content. In *Proceedings of the 2nd workshop on abusive language online (ALW2)*, pages 21–25.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. 2015. Delving deep into rectifiers: Surpassing human-level performance on imagenet classification. In *Proceedings of the IEEE international conference on computer vision*, pages 1026–1034.

Irina Higgins, Loic Matthey, Arka Pal, Christopher Burgess, Xavier Glorot, Matthew Botvinick, Shakir Mohamed, and Alexander Lerchner. 2016. beta-vae: Learning basic visual concepts with a constrained variational framework. *ICLR 2017 conference*.

Quan Hoang. 2018. Predicting movie genres based on plot summaries. *arXiv preprint arXiv:1801.04813*.

Junjie Huang, Duyu Tang, Wanjun Zhong, Shuai Lu, Linjun Shou, Ming Gong, Daxin Jiang, and Nan Duan. 2021. WhiteningBERT: An easy unsupervised sentence embedding approach. In *Findings of the Association for Computational Linguistics: EMNLP 2021*, pages 238–244, Punta Cana, Dominican Republic. Association for Computational Linguistics.

Jagadeesh Jagarlamudi, Hal Daumé III, and Raghavendra Udupa. 2012. Incorporating lexical priors into topic models. In *Proceedings of the 13th Conference of the European Chapter of the Association for Computational Linguistics*, pages 204–213.

Giannis Karamanolakis, Subhabrata Mukherjee, Guoqing Zheng, and Ahmed Hassan Awadallah. 2021. Self-training with weak supervision. *arXiv preprint arXiv:2104.05514*.

Diederik P Kingma and Prafulla Dhariwal. 2018. Glow: Generative flow with invertible 1x1 convolutions. *arXiv preprint arXiv:1807.03039*.

Diederik P Kingma and Max Welling. 2013. Auto-encoding variational bayes. *arXiv preprint arXiv:1312.6114*.

Bohan Li, Hao Zhou, Junxian He, Mingxuan Wang, Yiming Yang, and Lei Li. 2020. On the sentence embeddings from pre-trained language models. *arXiv preprint arXiv:2011.05864*.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Roberta: A robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692*.

Ilya Loshchilov and Frank Hutter. 2017. Decoupled weight decay regularization. *arXiv preprint arXiv:1711.05101*.

Dheeraj Mekala and Jingbo Shang. 2020. Contextualized weak supervision for text classification. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 323–333.

Yu Meng, Jiaxin Huang, Guangyuan Wang, Zihan Wang, Chao Zhang, Yu Zhang, and Jiawei Han. 2020a. Discriminative topic mining via category-name guided text embedding. In *Proceedings of The Web Conference 2020*, pages 2121–2132.

Yu Meng, Jiaming Shen, Chao Zhang, and Jiawei Han. 2018. Weakly-supervised neural text classification. In *Proceedings of the 27th ACM International Conference on Information and Knowledge Management*, pages 983–992.

Yu Meng, Yunyi Zhang, Jiaxin Huang, Chenyan Xiong, Heng Ji, Chao Zhang, and Jiawei Han. 2020b. Text classification using label names only: A language model self-training approach. *arXiv preprint arXiv:2010.07245*.

Shervin Minaee, Nal Kalchbrenner, Erik Cambria, Narjes Nikzad, Meysam Chenaghlu, and Jianfeng Gao. 2021. Deep learning–based text classification: A comprehensive review. *ACM Computing Surveys (CSUR)*, 54(3):1–40.

Maria Pontiki, Dimitris Galanis, John Pavlopoulos, Harris Papageorgiou, Ion Androutsopoulos, and Suresh Manandhar. 2014. SemEval-2014 task 4: Aspect based sentiment analysis. In *Proceedings of the 8th International Workshop on Semantic Evaluation (SemEval 2014)*, pages 27–35, Dublin, Ireland. Association for Computational Linguistics.

Nils Reimers and Iryna Gurevych. 2019. Sentence-bert: Sentence embeddings using siamese bert-networks. *arXiv preprint arXiv:1908.10084*.

Wendi Ren, Yinghao Li, Hanting Su, David Kartchner, Cassie Mitchell, and Chao Zhang. 2020. Denoising multi-source weak supervision for neural text classification. *arXiv preprint arXiv:2010.04582*.

Marzieh Saeidi, Guillaume Bouchard, Maria Liakata, and Sebastian Riedel. 2016. SentiHood: Targeted aspect based sentiment analysis dataset for urban neighbourhoods. In *Proceedings of COLING 2016, the 26th International Conference on Computational Linguistics: Technical Papers*, pages 1546–1556, Osaka, Japan. The COLING 2016 Organizing Committee.

Victor Sanh, Lysandre Debut, Julien Chaumond, and Thomas Wolf. 2019. Distilbert, a distilled version of bert: smaller, faster, cheaper and lighter. *arXiv preprint arXiv:1910.01108*.

Timo Schick, Helmut Schmid, and Hinrich Schütze. 2020. Automatically identifying words that can serve as labels for few-shot text classification. *arXiv preprint arXiv:2010.13641*.

Timo Schick and Hinrich Schütze. 2020. Exploiting cloze questions for few shot text classification and natural language inference. *arXiv preprint arXiv:2001.07676*.

Casper Kaae Sønderby, Tapani Raiko, Lars Maaløe, Søren Kaae Sønderby, and Ole Winther. 2016. Ladder variational autoencoders. *Advances in neural information processing systems*, 29:3738–3746.

Mukund Sundararajan, Ankur Taly, and Qiqi Yan. 2017. Axiomatic attribution for deep networks. In *International Conference on Machine Learning*, pages 3319–3328. PMLR.

Grigorios Tsoumakas and Ioannis Katakis. 2007. Multi-label classification: An overview. *International Journal of Data Warehousing and Mining (IJDWM)*, 3(3):1–13.

Zihan Wang, Dheeraj Mekala, and Jingbo Shang. 2020. X-class: Text classification with extremely weak supervision. *arXiv preprint arXiv:2010.12794*.

Wenpeng Yin, Jamaal Hay, and Dan Roth. 2019. Benchmarking zero-shot text classification: Datasets, evaluation and entailment approach. *arXiv preprint arXiv:1909.00161*.

Wenpeng Yin, Nazneen Fatema Rajani, Dragomir Radev, Richard Socher, and Caiming Xiong. 2020. Universal natural language processing with limited annotations: Try few-shot textual entailment as a start. *arXiv preprint arXiv:2010.02584*.

Yue Yu, Simiao Zuo, Haoming Jiang, Wendi Ren, Tuo Zhao, and Chao Zhang. 2020. Fine-tuning pre-trained language model with weak supervision: A contrastive-regularized self-training approach. *arXiv preprint arXiv:2010.07835*.

Appendix

A Variational AutoEncoder (VAE)

Variational AutoEncoder (VAE) (Kingma and Welling, 2013) has been widely used as an unsupervised generative method, especially in computer vision filed. It has been shown that β-VAE (Higgins et al., 2016) and its variants are capable of finding visual disentangled latent representations which remain invariant to some transformations (Burgess et al., 2018). This section provides a brief review of VAE and β-VAE.

VAE starts by solving the evidence (or marginal likelihood) $p_\theta(x)$ in Variational Inference (VI) problems involving latent variable Z. In particular, the evidence $p_\theta(x) = \int p(x,z)dz$ is often intractable to compute in practice due to computational cost. In VI, $p_\theta(x)$ can be approximated by introducing variational distributions and building a lower bound $\mathcal{L}$ to reframe the problem as in Eq. 1 based on Jensen's inequality.

$$\begin{aligned} logp_\theta(x) &= log(\mathbb{E}_{q_\phi(z|x)}[p_\theta(x|z)\frac{p(z)}{q_\phi(z|x)}]) \\ &\geq \mathbb{E}_{q_\phi}[logp_\theta(x|z)] - KLD(q_\phi(z|x)||p(z)) \\ &= \mathcal{L}(\theta,\phi;x) \\ &= -(L_R + L_{KLD}) \end{aligned} \tag{1}$$

L_R and L_{KLD} refer to reconstruction loss and KLD loss respectively. By considering $q_\phi(z|x)$ as a Gaussian probabilistic encoder, $p(z)$ as a standard Gaussian prior $\mathcal{N}(0,I)$ and $p_\theta(x|z)$ as a probabilistic decoder, the objective can be defined as $\max_{\theta,\phi} \mathcal{L}(\theta,\phi;x)$. In particular, this objective function aims at finding optimal parameters θ and ϕ to maximize the lower bound $\mathcal{L}$ which in turn approximates the log-probability of the data $logp_\theta(x)$. In addition, in order to estimate the gradients of the lower bound with respect to ϕ more smoothly, a parameterization trick is applied: $z_i \sim q_\phi(z_i|x) = \mu_i + \sigma_i\epsilon_i$, where $\epsilon_i \sim \mathcal{N}(0,1)$.

In β-VAE, a hyper-parameter β is added into the objective function as shown in Eq. 2. Usually $\beta > 1$ will result in more disentangled representations and when $\beta = 1$, the β-VAE is equivalent to the vanilla VAE model (Burgess et al., 2018).

$$\mathcal{L}(\theta,\phi;x,\beta) = -(L_R + \beta L_{KLD}) \tag{2}$$

B Comparison with Weakly-supervised Multi-Class Methods

Here we compare our model with the following strong weakly-supervised multi-class models on MLTC datasets:

WeSTClass (Meng et al., 2018): a WSTC model that has been briefly introduced in the related-work section. It can receive inputs of topic surface name, keywords or limited amount of documents. We used the keywords as input to it. We replaced its last softmax layer with sigmoid layer and used the same threshold (0.5).

X-Class (Wang et al., 2020): this methods is based on aligning document representation and class representation. It uses class surface names as input and can generage pseudo labels to train a text classifier. We used its X-Class-Align version which uses a Gaussian Mixture Model (GMM) to make final predictions. We modified its final to output unnormalized probabilities and used the threshold of 0.5.

LOTClass (Meng et al., 2020b): it is based on Masked Category Prediction (MCP) task and a subsequent self-training to perform WSTC with only label surface names. We replaced its last softmax layer with sigmoid and used the same threshold of 0.5. We also tested transforming it into a binary relevance task by only using one label surface name each time, which results in similar performance and therefore is not reported.

We used default hyper-parameters for all three models. We slightly modified keywords and topic surface names if the model has a different specification, otherwise we used the same keywords and topic surface names as used in our model. The results of comparison is presented in Table 5. It can be shown that there is a large performance margin between our model and the weakly-supervised multi-class models on multi-label tasks in terms of key metrics such as F1-score.

C Further Evaluation

Qualitative Analysis To further check the validity of the BFV model, we used Integrated Gradients (IG) (Sundararajan et al., 2017) to calculate the attributions of each word with respect to predicted topics. In Fig 5, we applied this inspection on 3 sentences from different datasets as examples. From Fig 5, it can be seen that the gradients of words can largely align with human intuition.

Clustering Results We also tested our model's performance in terms of clustering metrics. In calculating clustering metrics, we only used samples with single label in the whole dataset. Table 6 shows the models' performance measured by clustering metrics.

Embeddings Visualization Embeddings produced at different stages of the model are visualized by T-SNE in Figure 6, where the shift of the embeddings during the process could be observed.

Other Language Models We tested the model's performance with different backend language models. Table 7 shows the model's performance on different datasets in terms of F1-score when using other language models. However, we simply tested these models using the same hyper-parameters, which may cause impacts on the performance.

D Supplementary Information of the Datasets

Fig 7 displays the distribution of the amount of topics in the samples in percentage for different datasets used in this paper.

In regard to the *Heritage* dataset, all annotators (authors of the paper) have checked the correctness of labelling and are fully aware of the risks to participant.

E Definition of Evaluation Metrics

In evaluating our model in Table 2, we have the following metrics. Assuming $y_i \in \mathbb{R}^C$ is the ground truth label and $\hat{y}_i$ is the prediction for ith sample, where C is the number of classes:

ACC:

$$\text{Accuracy} = \frac{1}{n}\sum_{i=1}^{n} I(y_i = \hat{y}_i)$$

HS:

$$\text{Hamming Score} = \frac{1}{n}\sum_{i=1}^{n} \frac{|y_i \cap \hat{y}_i|}{|y_i \cup \hat{y}_i|}$$

Precision:

$$\text{Precision} = \frac{1}{n}\sum_{i=1}^{n} \frac{|y_i \cap \hat{y}_i|}{|\hat{y}_i|}$$

Recall:

$$\text{Recall} = \frac{1}{n}\sum_{i=1}^{n} \frac{|y_i \cap \hat{y}_i|}{|y_i|}$$

F1-score:

$$F_1 = \frac{1}{n}\sum_{i=1}^{n} \frac{2|y_i \cap \hat{y}_i|}{|y_i| + |\hat{y}_i|}$$

APS:

Average Precision Score[3]

AUC:

ROC-AUC score[4]

P@3:

Mean Average Precision @ k = 3[5]

Homogeneity:

homogeneity score[6]

Completeness:

completeness score[7]

NMI:

$$v = \frac{2 \times \text{homogeneity} \times \text{completeness}}{\text{homogeneity} + \text{completeness}}$$

Adj MI:

Adjusted Mutual Info Score[8]

Adj Rand:

Adjusted Rand Score[9]

[3] https://scikit-learn.org/stable/modules/generated/sklearn.metrics.average_precision_score.html

[4] https://scikit-learn.org/stable/modules/generated/sklearn.metrics.roc_auc_score.html

[5] https://github.com/benhamner/Metrics/blob/master/Python/ml_metrics/average_precision.py

[6] https://scikit-learn.org/stable/modules/generated/sklearn.metrics.homogeneity_score.html#sklearn.metrics.homogeneity_score

[7] https://scikit-learn.org/stable/modules/generated/sklearn.metrics.completeness_score.html#sklearn.metrics.completeness_score

[8] https://scikit-learn.org/stable/modules/generated/sklearn.metrics.adjusted_mutual_info_score.html

[9] https://scikit-learn.org/stable/modules/generated/sklearn.metrics.adjusted_rand_score.html

		ACC	F1	Recall	Precision	APS	AUC
Reuters	BFV (20)	**44.65**	**44.99**	**55.38**	**51.58**	**56.18**	**95.86**
	X-Class	27.63	33.83	38.05	40.64	30.44	67.84
	WeSTClass	6.16	1.12	2.60	1.84	2.99	48.18
	LOTClass	0.50	0.06	2.52	0.07	7.15	68.46
CitySearch	BFV (1)	**61.07**	**69.48**	**76.50**	**64.92**	**77.25**	**92.37**
	X-Class	49.77	16.62	25.00	12.44	18.78	50.00
	WeSTClass	25.49	22.26	23.31	24.21	18.99	47.50
	LOTClass	33.33	25.43	36.17	32.93	36.23	69.49
Sentihood	BFV (1)	42.94	**53.54**	**75.58**	**47.21**	**59.10**	**92.12**
	X-Class	**55.94**	17.29	16.29	46.58	12.48	54.32
	WeSTClass	8.92	6.50	12.30	10.61	7.34	51.28
	LOTClass	10.93	4.26	11.72	4.98	10.05	58.75
Restaurant	BFV (1)	**68.85**	**80.49**	**80.58**	**80.88**	**89.73**	**95.21**
	X-Class	38.75	13.96	24.92	9.70	24.75	50.03
	WeSTClass	26.38	23.64	24.94	25.91	25.90	49.72
	LOTClass	35.25	13.53	25.16	37.41	44.58	69.51
Laptop	BFV (2)	45.72	**36.40**	**38.59**	**42.75**	**39.25**	**73.38**
	X-Class	**68.07**	16.20	20.00	13.61	11.88	50.00
	WeSTClass	22.65	15.01	18.93	19.53	12.73	49.27
	LOTClass	9.03	10.41	28.27	18.04	15.83	55.45
Heritage	BFV (5)	40.10	**44.20**	**50.34**	**41.75**	**46.68**	**81.55**
	X-Class	**41.25**	7.31	12.50	5.16	10.01	50.22
	WeSTClass	10.12	8.47	9.59	12.07	9.80	48.71
	LOTClass	10.25	5.48	15.42	4.15	16.41	61.87

Table 5: Performance of BFV in comparison with weakly-supervised multi-class models on multi-label datasets. All numbers are percentages.

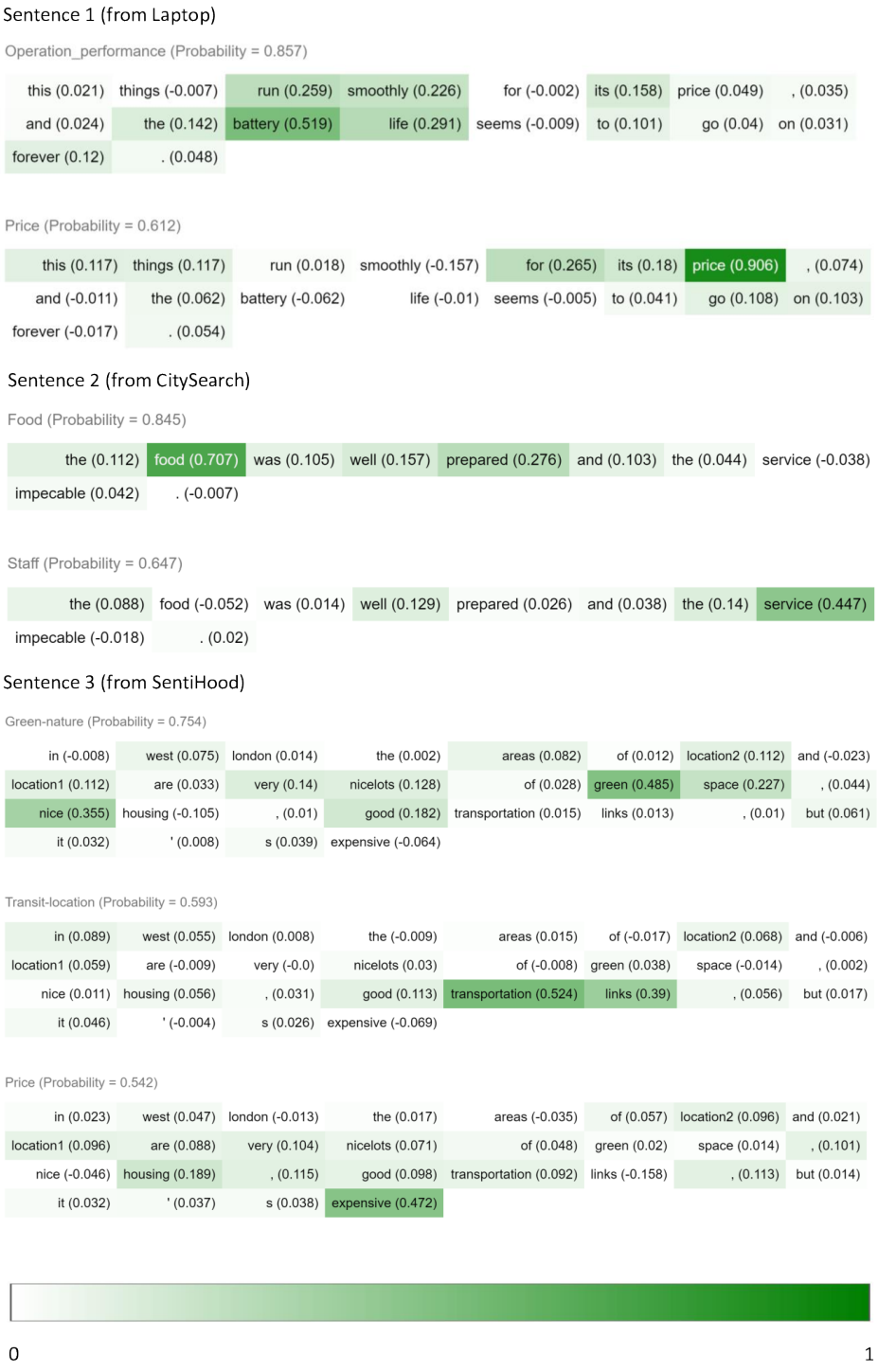

Figure 5: Qualitative analysis of the correct predictions made by BFV model using Integrated Gradients (IG). Numbers with bracket attached to each words are gradients given by the IG

		Homogeneity	Completeness	NMI	Adj MI	Adj Rand
Reuters	BFV (20)	67.72 (0.39)	58.73 (0.74)	62.90 (0.56)	61.87 (0.58)	69.99 (1.86)
	BWV (20)	66.08 (0.38)	54.94 (0.62)	60.00 (0.52)	58.87 (0.53)	64.90 (1.45)
	BV (20)	**67.87** (0.38)	**61.73** (0.64)	**64.66** (0.48)	**63.71** (0.5)	**73.34** (0.83)
	BFVCLS (20)	64.17 (1.1)	60.21 (1.98)	62.11 (1.41)	61.23 (1.43)	71.00 (3.85)
	BFE (20)	63.29 (0.26)	56.39 (0.33)	59.64 (0.28)	58.56 (0.29)	69.03 (0.6)
	BF	22.45	17.47	19.65	17.93	9.12
	0SHOT-TC+CorEx	58.74	45.12	51.04	49.62	54.98
	0SHOT-TC-Flow	11.36	7.22	8.83	6.33	3.99
	0SHOT-TC	63.93	52.93	57.91	56.78	64.09
	CorEx	34.40	25.24	29.12	27.23	22.75
	GLDA	46.96	31.69	37.85	35.98	23.46
CitySearch	BFV (1)	**51.71** (1.19)	**48.99** (1.17)	**50.31** (1.16)	**50.21** (1.16)	**60.21** (1.37)
	BWV (1)	44.91 (1.24)	42.36 (1.64)	43.60 (1.44)	43.48 (1.44)	51.63 (1.94)
	BV (1)	43.50 (1.7)	43.19 (2.29)	43.32 (1.7)	43.20 (1.7)	51.72 (2.89)
	BFVCLS (1)	40.11 (1.95)	44.07 (1.8)	41.98 (1.68)	41.85 (1.68)	50.32 (2.88)
	BFE (1)	41.06 (0.81)	42.58 (0.87)	41.81 (0.81)	41.67 (0.81)	50.94 (0.78)
	BF	0.69	2.77	1.11	0.73	2.59
	0SHOT-TC+CorEx	35.86	33.45	34.61	34.47	40.80
	0SHOT-TC-Flow	14.94	13.61	14.25	14.06	13.96
	0SHOT-TC	43.24	46.24	44.69	44.56	54.24
	CorEx	15.92	14.07	14.94	14.76	8.28
	GLDA	7.36	6.24	6.75	6.56	6.75
Sentihood	BFV (1)	53.83 (1.48)	51.72 (1.48)	52.76 (1.46)	52.34 (1.47)	49.13 (2.69)
	BWV (1)	48.83 (1.39)	48.41 (1.15)	48.62 (1.25)	48.15 (1.26)	37.53 (3.33)
	BV (1)	45.42 (3.17)	48.56 (2.38)	46.92 (2.61)	46.41 (2.64)	28.08 (5.67)
	BFVCLS (1)	34.68 (2.8)	41.01 (2.71)	37.57 (2.69)	36.96 (2.73)	20.68 (5.11)
	BFE (1)	42.67 (0.75)	44.54 (0.76)	43.59 (0.75)	43.06 (0.76)	28.01 (0.8)
	BF	3.37	5.26	4.11	3.44	3.03
	0SHOT-TC+CorEx	52.08	50.04	51.04	50.60	49.09
	0SHOT-TC-Flow	5.42	5.56	5.49	4.61	3.59
	0SHOT-TC	**57.08**	**56.77**	**56.92**	**56.53**	**49.87**
	CorEx	24.57	25.50	25.03	24.33	11.79
	GLDA	18.67	16.53	17.53	16.84	15.11
Restaurant	BFV (1)	**59.23** (1.29)	**57.23** (1.85)	**58.21** (1.5)	**58.13** (1.51)	**68.13** (1.84)
	BWV (1)	50.04 (1.78)	44.93 (1.84)	47.34 (1.81)	47.25 (1.81)	53.10 (2.78)
	BV (1)	46.07 (1.34)	44.48 (1.85)	45.25 (1.5)	45.15 (1.5)	49.78 (4.81)
	BFVCLS (1)	41.03 (2.32)	44.44 (2.63)	42.63 (2.08)	42.51 (2.08)	50.98 (4.88)
	BFE (1)	45.86 (0.79)	44.64 (0.84)	45.24 (0.81)	45.14 (0.81)	52.47 (1.19)
	BF	1.41	27.83	2.69	2.45	1.50
	0SHOT-TC+CorEx	49.44	45.98	47.65	47.55	56.46
	0SHOT-TC-Flow	1.79	1.59	1.68	1.50	3.32
	0SHOT-TC	54.94	56.52	55.72	55.63	67.55
	CorEx	19.13	18.50	18.81	18.66	5.37
	GLDA	9.93	8.21	8.99	8.83	7.97
Laptop	BFV (2)	**22.28** (0.95)	21.90 (1.11)	**22.09** (1.02)	**21.78** (1.02)	**14.27** (0.77)
	BWV (2)	20.38 (0.58)	21.46 (0.51)	20.90 (0.54)	20.58 (0.55)	12.78 (1.18)
	BV (2)	20.45 (0.99)	22.04 (0.98)	21.20 (0.84)	20.87 (0.84)	12.75 (1.75)
	BFVCLS (2)	12.52 (1.12)	17.58 (1.59)	14.60 (1.09)	14.25 (1.11)	6.13 (0.95)
	BFE (2)	18.67 (0.69)	21.48 (0.77)	19.98 (0.73)	19.63 (0.73)	10.42 (0.43)
	BF	2.75	**44.62**	5.18	4.85	1.46
	0SHOT-TC+CorEx	18.45	19.57	19.00	18.66	11.48
	0SHOT-TC-Flow	0.58	0.62	0.60	0.18	-0.25
	0SHOT-TC	15.33	18.53	16.78	16.41	8.40
	CorEx	9.85	11.51	10.61	10.23	3.80
	GLDA	4.11	3.86	3.98	3.61	2.24
Heritage	BFV (5)	26.95 (0.7)	26.49 (0.77)	26.72 (0.73)	26.31 (0.74)	23.98 (1.29)
	BWV (5)	25.24 (0.38)	24.61 (0.35)	24.92 (0.36)	24.50 (0.37)	22.20 (0.5)
	BV (5)	**28.99** (0.74)	**28.82** (0.71)	**28.90** (0.7)	**28.50** (0.7)	**25.30** (1.66)
	BFVCLS (5)	26.43 (0.89)	27.59 (0.96)	26.99 (0.84)	26.57 (0.84)	22.10 (2.16)
	BFE (5)	26.92 (0.58)	27.04 (0.6)	26.98 (0.59)	26.57 (0.59)	23.28 (0.61)
	BF	0.69	3.40	1.15	0.65	-0.17
	0SHOT-TC+CorEx	21.22	20.78	21.00	20.56	17.60
	0SHOT-TC-Flow	1.08	1.63	1.30	0.61	-0.18
	0SHOT-TC	20.70	20.69	20.70	20.25	16.25
	CorEx	15.07	15.97	15.50	15.01	8.00
	GLDA	6.62	6.41	6.51	5.99	4.52

Table 6: Results for clustering performance. Highest values are marked with bold font. Numbers in the brackets are γ values (in column) or standard deviation (in matrix). All numbers are percentages.

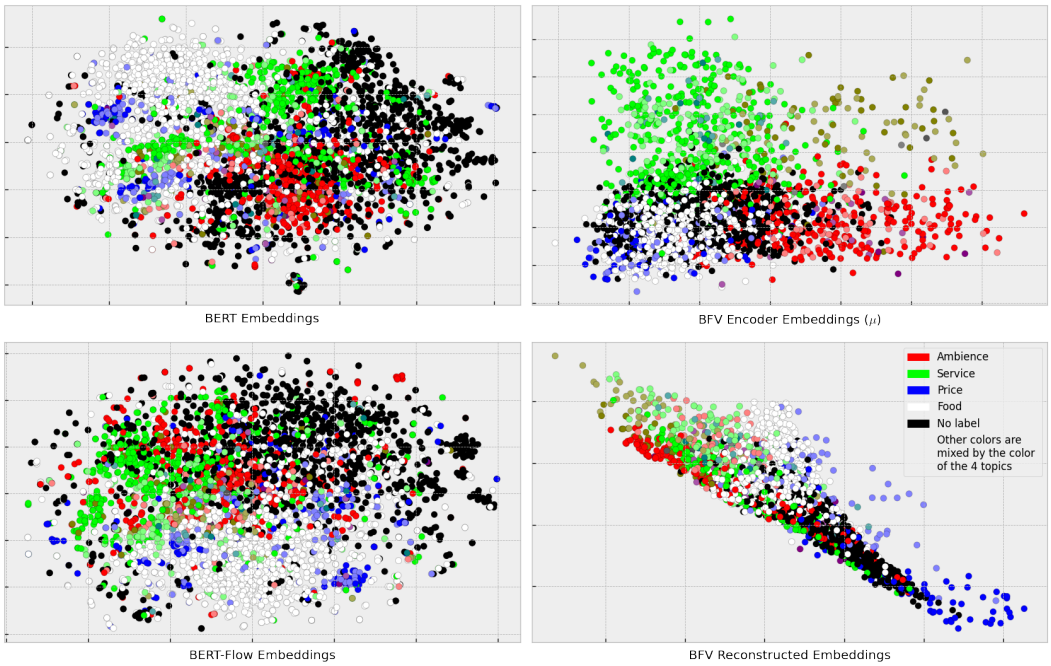

Figure 6: Visualization of embeddings produced at different stages of the model by T-SNE using the *Restaurant* dataset. Red, Green, Blue and White are used to represent topics 'ambience', 'service', 'price' and 'food' in the dataset. Colors other than Red, Blue, Green and White are mixed by the color of the corresponding topics when there are more than one topics assigned to the document.

	BERT-base	BERT-large	RoBERTa-base	RoBERTa-large	XLM-R-base
Reuters	42.12	43.53	40.22	44.16	42.16
CitySearch	69.52	69.05	70.96	71.73	70.42
Sentihood	54.49	52.14	55.34	54.48	53.54
Restaurant	79.02	80.72	81.14	83.53	79.29
Laptop	38.49	38.40	40.56	39.45	39.16
Heritage	42.45	43.41	44.57	42.41	42.43

Table 7: Models performance on 6 datasets with different backend language models in terms of F1-score. All numbers are percentages

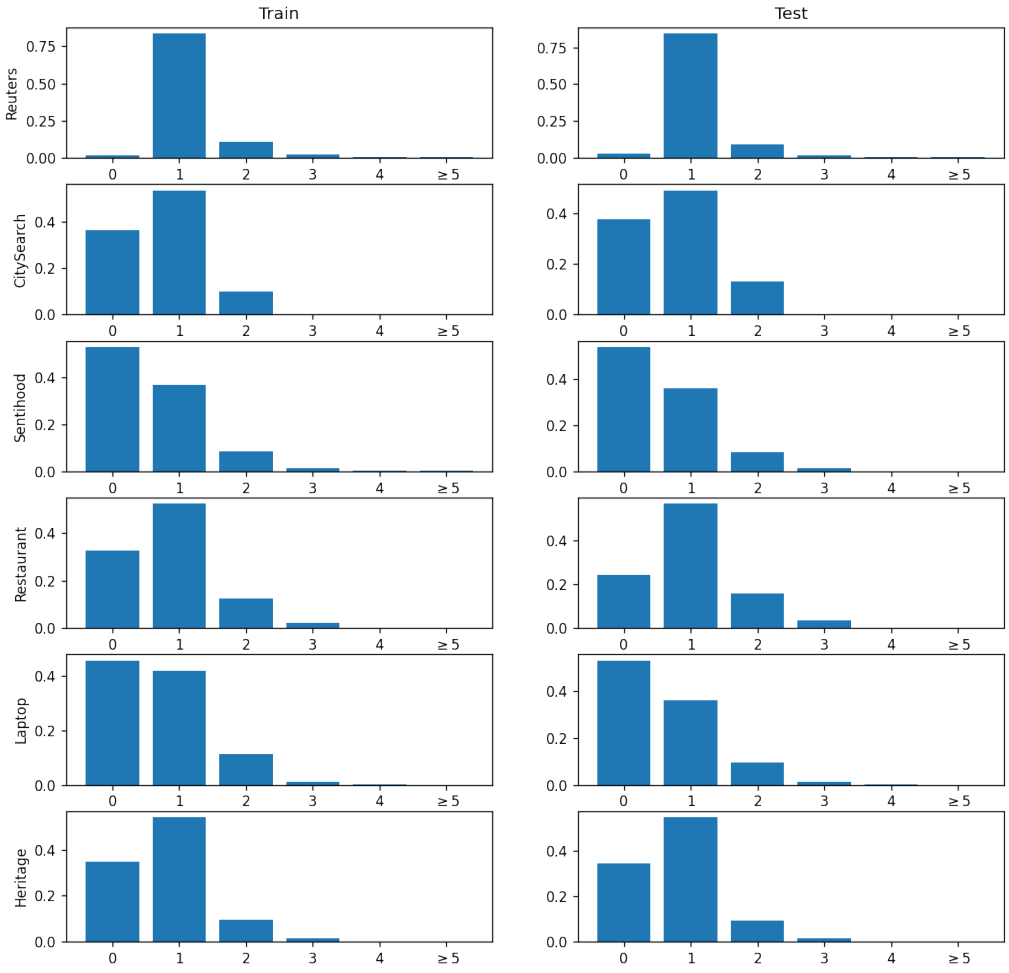

Figure 7: Distribution of the amount of topics in the samples in percentage.

Welcome to the Modern World of Pronouns: Identity-Inclusive Natural Language Processing beyond Gender

Anne Lauscher
MilaNLP
Università Luigi Bocconi
Milan, Italy
anne.lauscher@unibocconi.it

Archie Crowley
Linguistics
University of South Carolina
Columbia, SC, USA
acrowley@sc.edu

Dirk Hovy
MilaNLP
Università Luigi Bocconi
Milan, Italy
dirk.hovy@unibocconi.it

Abstract

Trigger warning: This paper contains examples that contain transphobic language.

The world of pronouns is changing – from a closed word class with few members to an open set of terms to reflect identities. However, Natural Language Processing (NLP) barely reflects this linguistic shift, resulting in the possible exclusion of non-binary users, even though recent work outlined the harms of gender-exclusive language technology. The current modeling of 3rd person pronouns is particularly problematic. It largely ignores various phenomena like neopronouns, i.e., novel pronoun sets that are not (yet) widely established. This omission contributes to the discrimination of marginalized and underrepresented groups, e.g., non-binary individuals. It thus prevents gender equality, one of the UN's sustainable development goals (goal 5). Further, other identity-expressions beyond gender are ignored by current NLP technology. This paper provides an overview of 3rd person pronoun issues for NLP. Based on our observations and ethical considerations, we define a series of five desiderata for modeling pronouns in language technology, which we validate through a survey. We evaluate existing and novel modeling approaches w.r.t. these desiderata qualitatively and quantify the impact of a more discrimination-free approach on an established benchmark dataset.

1 Introduction

Pronouns are an essential component of many languages and often one of the most frequent word classes. Accordingly, NLP has long studied tasks related to them, e.g., pronoun resolution (e.g., Hobbs, 1978). Simplistically, they can be defined as *"a word (such as I, he, she, you, it, we, or they) that is used instead of a noun or noun phrase"*.[1]

Nom.	Acc.	Poss. (dep.)	Poss. (indep.)	Reflexive
Gendered Pronouns				
he	him	his	his	himself
she	her	her	hers	herself
Gender-Neutral Pronouns				
they	them	their	theirs	themself
Neopronouns				
thon	thon	thons	thons	thonself
e	em	es	ems	emself
xe	xem	xyr	xyrs	xemself
ey	em	eir	eirs	emself
e	em	eir	eirs	emself
ze	zir	zir	zirs	zirself
...				
Nounself Pronouns				
star	star	stars	stars	starself
vam	vamp	vamps	vamps	vampself
...				
Emojiself Pronouns				
🥳	🥳	🥳s	🥳s	🥳self
🥳	🦄	🦄s	🦄s	🦄self
...				
Numberself Pronouns				
0	0	0s	0s	0self
$\frac{1}{3}$	$\frac{1}{3}$	$\frac{1}{3}$s	$\frac{1}{3}$s	$\frac{1}{3}$self
...				
Nameself Pronouns				
John	John	Johns	Johns	Johnself
...				

Table 1: Non-exhaustive overview of phenomena related to third-person pronoun usage in English.

Linguistic studies have pointed out the complexity of pronouns, though (e.g., Postal et al., 1969; McKay, 1993). Pronouns can carry demographic information – in English, for example, information about the *number* of referents and a single referent's (grammatical) *gender*.[2] Pronouns can convey even more information in other, non-pro-drop languages.[3] Consider Arabana-Wangkangurru, a

[1] Essential definition provided by the Mirriam Webster Online Dictionary at https://www.merriam-webster.com/dictionary/pronoun

[2] Grammatical and social gender should not be confounded, but are often treated interchangeably by lay audiences.

[3] First described by Perlmutter (1968), the phenomenon of "pro-drop languages" relates to languages, in which, in certain cases, some classes of pronouns can be omitted (e.g., Italian).

Proceedings of the 29th International Conference on Computational Linguistics, pages 1221–1232
October 12–17, 2022.

language spoken in Australia, in which a speaker uses different pronouns depending on whether the referent is part of the same social or ritual group (moiety) (Hercus, 1994). As such, pronouns shape how we perceive individuals and can even reflect cultural aspects (e.g., Kashima and Kashima, 1998) and ideologies (e.g., Muqit, 2012). Consequently, pronoun usage should be considered a sensitive aspect of natural language use.

Accordingly, in many western societies, these phenomena have been drawing more and more attention. For instance, in 2020, the American Dialect Society voted *"(My) Pronouns"* as the 2019 Word of the Year and *Singular "They"* as the Word of the Decade (Roberts, 2020). Recently, there has been a shift in pronoun usage (Krauthamer, 2021), partially due to shifts in the perception of *gender*, driven by the queer-feminist discourse (e.g., Butler, 1990, 2004). Related to this is the open discussion of identity beyond binary gender. For instance, a person who does not identify their gender within the gender binary (e.g., a *nonbinary* or *genderqueer* person) might use *singular "they"* as their pronoun. Recently, the French dictionary "Le Robert" added the non-binary pronoun *"iel"* to its list of words.[4]

This "social push" to respect diverse gender identities also affects NLP. Recent studies have pointed out the harms of the current lack of non-binary representation in data, embeddings, and tasks (Cao and Daumé III, 2021; Dev et al., 2021). However, the research landscape on modern pronoun usage is surprisingly scarce, hindering progress for fair and inclusive NLP. This lacuna is in direct contradiction of the UN's sustainable development goals,[5] which include gender equality (goal 5).

Linguistic research has identified further identity aspects of pronouns, beyond gender (Miltersen, 2016). Specifically, *nounself* pronouns are functionally turning pronouns from a *closed* into an *open* word class. To the best of our knowledge, NLP has completely ignored these aspects so far. We did not find a single work systematically describing all of the currently existing phenomena, even just in English 3rd person pronoun usage (let alone other languages).[6] In contrast, many discussions are taking place on queer Wikis and forums. While it is still unclear which of these phenomena will persist over the following decades, people use and discuss them. Accordingly, we as a research community should adapt.

Contributions. **1)** We are the first to provide a systematic overview of existing phenomena in English 3rd person pronoun usage. Our results will inform future NLP research on ethical NLP and non-binary representation. We provide the first NLP work acknowledging *otherkin* identities. We support our observations with a corpus analysis on Reddit. **2)** Based on our overview, we derive five desiderata for modeling third-person pronouns, which we validate with a survey among 39 individuals (coupled with a pre-study with 149 participants), most of which identify as non-binary. Based on these criteria, **3)** we discuss various existing and novel paradigms for *when* and *how* to model pronouns in NLP. **4)** Finally, we quantify the impact of discrimination-free non-modeling of pronouns on a widely established benchmark.

2 Related Work

While there are some works in NLP on gender-inclusion (e.g., Dev et al., 2021) and gender bias in static (e.g., Bolukbasi et al., 2016; Gonen and Goldberg, 2019; Lauscher and Glavaš, 2019; Lauscher et al., 2020, *inter alia*) and contextualized (e.g., Kurita et al., 2019; Bordia and Bowman, 2019; Lauscher et al., 2021, *inter alia*) language representations as well as works focusing on specific gender bias in downstream tasks (e.g., Rudinger et al., 2018; Webster et al., 2018; Dev et al., 2020; Barikeri et al., 2021), we are not aware of any work that deals with the broader field of *identity-inclusion*. Thus, there is no other NLP work that deals with a larger variety of pronouns and acknowledges pronouns as an open word class. For surveys on the general topic of unfair bias in NLP we refer to Blodgett et al. (2020) and Shah et al. (2020). Recently, Dev et al. (2021) pointed at the representational and allocational harms (Barocas et al., 2017) arising from gender-exclusivity in NLP. They surveyed queer individuals and assessed non-binary representations in existing data sets and embeddings. In contrast, we specifically look at third-person pronoun usage and how to model such phenomena. Webster

In contrast, in "non-pro-drop languages", pronouns cannot be omitted (e.g., German).

[4] https://dictionnaire.lerobert.com/dis-moi-robert/raconte-moi-robert/mot-jour/pourquoi-le-robert-a-t-il-integre-le-mot-iel-dans-son-dictionnaire-en-ligne.html

[5] https://sdgs.un.org/goals

[6] For instance, while we found hits for the Google Scholar query *"neopronoun"*, we did not get any results for variants of *"nameself pronoun"*, or *"emojiself pronoun"*.

et al. (2018) provide a balanced co-reference resolution corpus with a focus on the fair distribution of pronouns but only focus on the gendered binary case. Closest to us, Cao and Daumé III (2021) discuss gender inclusion throughout the NLP pipeline beyond binary gender. While they are the first to consider gender-neutral pronouns, including some neopronouns, they do not acknowledge the broader spectrum of identity-related pronoun phenomena.

3 A Note on Identity and Pronouns

This work focuses on the relationship between identity and pronouns. *Identity* refers to an individual's self-conceptualization, relating to the question of what makes each of us unique (Maalouf, 2000). It can be seen as a two-way process between an individual and others (Grandstrand, 1998), and relates to different dimensions, e.g., one's gender.

Gender Identity. Gender identity, as opposed to gender expression or sex, is one's inner sense of gender (Stryker, 2017; Keyes et al., 2021). In this work, we recognize gender identities beyond a cisnormative binary (*cis man*, *cis woman*), e.g., transgender, non-binary, agender, etc.

Otherkin Identity. Individuals with otherkin identity do not entirely identify as human (Laycock, 2012), e.g., vamp. Miltersen (2016) note that otherkin individuals often identify with *nounself* pronouns matching their kin.

Stryker (2017) highlights the strong relationship between gender identity and pronouns. As Raymond (2016) notes, pronoun choices construct the individual's identity in conversations and the relationship between interlocutors. According to Cao and Daumé III (2021), pronouns are a way of expressing referential gender. Referring to an individual with sets of pronouns they do not identify with, e.g., resulting in misgendering, is considered harmful (e.g., Dev et al., 2021).

4 Phenomena in Third-person Pronoun-Usage

We describe existing phenomena and analyze their presence in a collection of threads from Reddit.[7]

4.1 Existing Phenomena

Overall, individuals can choose n sets of pronouns with $n \geq 0$. If $n = 0$, the individual does not identify with any singular 3rd person pronoun. If $n > 1$, the individual identifies with more than one set of pronouns. Each set is possibly reflecting overlapping or non-overlapping aspects of their identity. We provide examples of these sets in Table 1. Note that this list is non-exhaustive and that the described phenomena are non-exclusive.

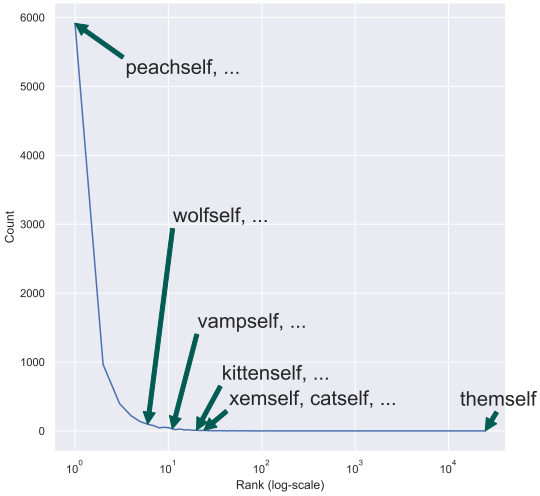

Figure 1: Token ranks (log-scale) and rank counts of the tokens returned against our reflexive regular expression pattern from Reddit with example annotations.

Gendered Pronouns. In English, two sets of gendered pronouns are available, *he/him/himself* and *she/her/herself*.

Gender-Neutral Pronouns. Given the history of generic singular *they* in English (e.g., Who was at the door? *They* left a note.), there has been an uptake of singular *they* by non-binary individuals as a gender-neutral pronoun option[8] (Conrod, 2019; Konnelly and Cowper, 2020). Further, there has been increasing institutional recognition with dictionaries and style guides supporting its use.

Neopronouns. As an alternative to the singular *they*, individuals started creating and sharing novel sets of 3rd person pronouns (McGaughey, 2020). More traditional and rather well-known sets of neopronouns include, e.g., the so-called Spivak pronouns *e/emself* (used in (Spivak, 1990)) and related variations. During our research, we observed various subcategories of neopronouns, only partially described in the academic literature.

Nounself Pronouns. Nounself pronouns are "[...] prototypically transparently derived from a spe-

[7] https://www.reddit.com

[8] https://gendercensus.com/results/2021-worldwide-summary/

cific word, usually a noun" (Miltersen, 2016). Individuals may identify with certain nouns, possibly corresponding to aspects of their identity, e.g., *kitten/kittenself*, *vamp/vampself*. The author notes the difficulty of clearly defining nounself pronouns, neopronouns, and other phenomena. The phenomenon is assumed to have first appeared in 2013.

Emojiself Pronouns. Similar to nounself pronouns, individuals may identify with sets of emojis, possibly reflecting different aspects of their identity, e.g., 🥳/ 🥳*self*. Emojiself pronouns are intended for written communication. Note that, at the time of writing this manuscript, no academic description of emojiself pronouns seems to exist. However, we were able to find evidence of their existence on several social media platforms and wikis, e.g., Tumblr,[9] MOGAI Wiki,[10] Twitter,[11] and Reddit.[12]

Numberself Pronouns. Another form of neopronouns/ nounself pronouns are numberself pronouns. Analogous to before, we assume that here, the individual identifies or partially identified with a number, e.g., *0/0/0s/0s/0self*.[13]

Nameself Pronouns. Individuals may identify with pronouns build from their name, e.g., *John/Johnself*, overlapping with nullpronomials.[14]

Alternating Pronouns. Suppose someone identifies with more than one set of pronouns. In that case, the pronouns they identify with can be either equally identified-with sets, or potentially change depending on the context (*mutopronominal*). As such, individuals who are also performers may use *stage pronouns*. Similarly, genderfluid individuals may identify with a certain pronoun at a certain point in time (*pronoun fluidity*, (Cherry-Reid, 2020)). Some individuals identify with the pronouns of the person who is referring to them (*mirrored pronouns*). Others use set(s) of *auxiliary pronouns*, e.g., for situations when people referring to them have problems with using the most identified-with sets of pronouns (e.g., in the case of emojiself pronouns and oral communication). Alternating pronoun sets may be even used in the same sentence for the same individual.[15]

No Pronouns. Some individuals do not identify with any pronouns. In this case, some individuals identify most with their name being used to refer to them, nameself pronouns, or avoid pronouns.

4.2 Corpus Analysis: Neopronouns in Reddit

Setup. We conduct an additional analysis for the presence of neopronouns in Reddit. To this end, we use Reddit threads (2010–2021), cleaned by previous work and provided through Huggingface Datasets (127,445,911 lines). The data set includes comment title, text, and subreddit.[16] Since we are interested in capturing novel pronouns, but the list of possible pronouns is indefinite, we proxy neopronouns via the suffixes *self* and *selves* to indicate the reflexive case. We match them through a regular expression. Additionally, we filter out non-3rd person pronouns (e.g., *yourself*, *ourselves*, plural *themselves*) as well as common variations of these (e.g., *urself*) and other common non-pronoun expressions we found in the data (e.g., *do-it-yourself*). This process leaves us with a total of 9,075 unique tokens with in total 74,768 textual mentions.

Results. Unsurprisingly, an initial manual analysis reveals that many of the matches are false positives, i.e., not real neopronouns like *non-self*, a common concept in Buddhist philosophy. However, our method still finds relevant cases. Note, that in this work, we do not explicitly deal with false positives – we are merely interested in whether our heuristic helps us to detect some sets of neopronouns at all, thus demonstrating their existence in real-world conversations. Examples of neopronouns we found are depicted in Table 2. Many discussions containing nounself pronouns center on the phenomena themselves, including, e.g., individuals stating that they are interested in using a specific pronoun or that they refuse to acknowledge the phenomenon. Some discussions involve people reporting on personal experiences and problems and seeking advice. To obtain a high-level quantitative

[9] E.g., https://pronoun-archive.tumblr.com/post/188520170831

[10] https://mogai.miraheze.org/wiki/Emojiself; according to the article, the origin of emojiself pronouns is unclear but might date back to 2017

[11] Example of a user complaining about LinkedIn not allowing for emojiself pronouns in the pronoun field: https://twitter.com/frozenpandaman/status/1412314202119700480/photo/1

[12] E.g., https://www.reddit.com/r/QueerVexillology/comments/p09nek/i_made_a_flag_for_the_emojiself_pronoun_set/

[13] https://pronoun-provider.tumblr.com/post/148452374817/i-think-numbers-as-pronouns-would-be-pretty-cool

[14] https://pronoun.fandom.com/wiki/Nullpronominal

[15] https://www.reddit.com/r/NonBinary/comments/jasv5r/alternating_pronouns_in_same_sentence/

[16] https://huggingface.co/datasets/sentence-transformers/reddit-title-body

Match	Thread Title	Thread Excerpt
meowself	*Fureedom Mewnite can die in my litterbox.*	*I don't like this game. But I still want meowself to play it, meow. Cause it's fun, even though I hate it.*
	Neopronouns are going too far.	*I get some pronouns like ze/zir, xe/xem, etc. I agree with those. But why are people using ghost/ghostself and meow/meowself? That's really utter bullshit.*
bunself	*I am genderfluid, and pansexual. I have a lot of SJW friends. AMA!*	*They/them pronouns are coolest with me, but I won't be angry if you use he or she. You can use bun/buns/bunself, if you are feeling special. (That's a joke.)*
	Xi am so proud to announce that the new word of the year is.....	*–Cinnagender– which means you identify with our beloved and innocent cinnamon buns. The pronoun set is cinne/cinns/cinnself or alternatively bun/buns/bunself i am so happy to be a member of a community that ignores the oppressive gender binary, which is a social construct, i.e., it is not real*
zirself	*Ran into our first roadblock*	*I asked what I could do to help zir lowering the feeling of disphoria, and ze said zeď maybe feel better about zirself if zeď drink a tea.*
	If you're a horrible person online, you're probably a horrible person offline too.	*Hello folks. Omg. I think this individual is about to hurt zirself! (emphasis on "zirself". COMEDIC GENIUS)*

Table 2: Example neopronouns and corresponding excerpts from Reddit retrieved via our heuristic method. We slightly modified the excerpts to lower searchability and increase the privacy of the users.

view, we compute the matches' ranks as the number of texts in which particular matches occurred (including false positives) against their number of tokens (e.g., there is only 1 match, which appeared in 24,697 texts; there are 2 matches, which appeared in 198 texts, etc.). We show the results in Figure 1 (log-scale). The result is a highly skewed Zipf's distribution: while the highest ranks appear only once (e.g., *themself* with 24,697 mentions), some tokens appear only a couple of times (e.g., the neopronoun *xemself* with 24 mentions), and the vast majority appears only once (e.g., many nounself pronouns such as *peachself*).

5 How Can and Should We Model Pronouns?

We devise five desiderata based on our observations, personal experiences, expert knowledge from interactions with LGBTQIA+ associates, and informal discussions with individuals using gender-neutral pronouns. We validate the desiderata through a survey. Here, we collect opinions from 39 individuals (149 in the pre-study), most of whom identify as non-binary. We then assess how well classic and novel NLP pronoun modeling paradigms fulfill the criteria.

5.1 Desiderata

D1. Refrain from assuming an individual's identity and pronouns. A model should not assume an individual's identity, e.g., gender, or pronouns based on, e.g., statistical cues about an individual's name, also not in a binary gender setup. Only because the name *John* typically appears together with the pronoun *he*, the model should not assume that a person with the name *John* identifies as a man and that every *John* uses the pronoun *he*.

D2. Allow for the existing sets of pronouns as well as for neopronouns. A model should be able to handle not only the existing set of "standard" pronouns in a language but also other existing pronouns, e.g., neopronouns.

D3. Allow for novel pronouns at any point in time. On top of **D2**, a model should allow for novel, i.e., unseen, pronouns to appear at any point in time. This condition is necessary to account for the fact that neopronouns are not a fixed set, but evolving, and because related phenomena (emoji-self and nameself pronouns) turn pronouns from a *closed* to an *open class* part of speech.

D4. Allow for multiple, alternating, and changing pronouns. A model should not assume that the pronoun set for an individuum at time t will be the same as at time $t-1$. Even within the same sequence, pronoun sets might change.

D5. Provide an option for individuals to define their sets of pronouns. While most NLP models are trained offline and do not interact with the user, some are designed to interact with individuals, e.g., dialog systems. Here, letting individuals provide their sets of pronouns can help avoid harmful interactions (depending on the sociotechnical scenario).

5.2 Validation

Survey Design. We divide the survey into three parts: first, participants are asked for demographic information (age, (gender) identity, native language(s), pronouns). The second part asks for their opinion on D1–D5. We first provide a general contextualization of our research and describe the task. Participants are asked to indicate how much they agree with each desideratum (ordinal scale, 1 (*I don't agree*) to 5 (*I absolutely agree*)). We also allow for leaving additional comments. The third part relates to a case study on machine translation. We inform the participants that their participation is completely voluntary and that they will not receive any compensation. All questions are optional. To avoid sequence effects, we create multiple versions of the survey shuffling the order of the desiderata. We obtained ethical approval for the design by one of our universities' institutional review board.

Survey Distribution. Opting to collect opinions from affected individuals, we distribute the survey through various international LGBTQIA+ networks, e.g., QueerInAI,[17] Committee on LGBTQ+ [Z] Issues in Linguistics,[18] as well as through local LGBTQIA+ groups, e.g., Transgender Network Switzerland.[19] In an initial pre-study, which was open for participation between 22nd of March and 4th of May 2022, we validated our design. In total, 149 individuals participated in this phase (more than 8x more than in (Dev et al., 2021)).[20] The main phase of the survey was open for participation between 18th of June and 1st of August 2022.

Partcipant Statistics. In total, 44 individuals participated in the main phase of our survey, more than in any other survey on (gender) identity and language technology we are aware of. Participant ages range from 14 to 43 (the majority between 20 and 30). For the rest of the analysis, we removed all records from individuals under the age of 18. These individuals indicated that they speak diverse native languages (e.g., German, English, Danish, Persian, Russian). Participants provided between 0 and 4 identity terms (e.g., genderfluid, genderqueer, trans*masculine, etc.), with the majority identifying as non-binary. Thus, we believe our results to reflect cultural and (gender) identity diversity. Figure 2 shows the distribution over the English pronoun sets participants identify with.

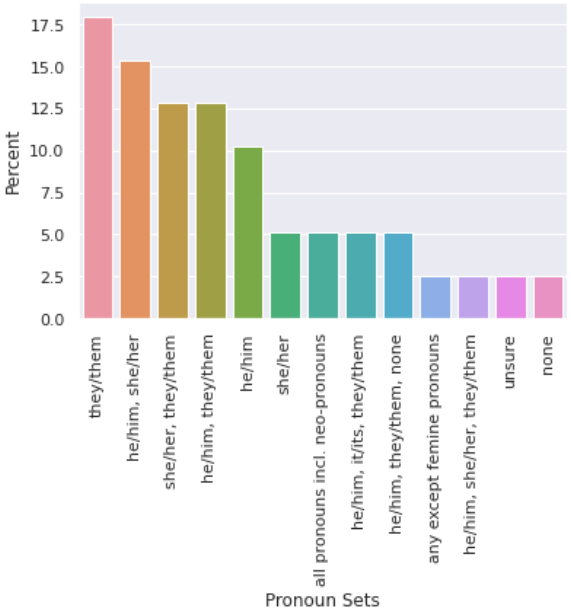

Figure 2: Distribution over English pronoun sets the participants of our survey identify with. We apply slight lexical normalization to the pronoun strings provided in the free text field (e.g., *he* → *he/him*, etc.).

Score	1	2	3	4	5	Avg
D1	5.13%	2.56%	0.00%	33.33%	58.97%	4.38
D2	2.56%	0.00%	5.13%	17.95%	74.36%	4.62
D3	2.56%	0.00%	12.82%	25.64%	58.97%	4.38
D4	2.56%	5.13%	2.56%	23.08%	66.67%	4.46
D5	0.00%	2.56%	5.13%	20.51%	71.79%	4.62

Table 3: Agreement distribution for our desiderata D1–D5. We report the % agreement per score (1=*I don't agree*, 5=*I absolutely agree*) as well as the average score.

Agreement with the Desiderata. We show the distribution of agreement scores for each desideratum in Table 3. Overall, we note high agreement (on average 4.38–4.62). Thus, we conclude our proposed desiderata to provide valuable orientation for the treatment of pronouns leading to more identity-inclusive language technology.

5.3 Modeling Paradigms

We compare four general modeling paradigms with the desiderata D1–D5 in Table 4.

Classic Statistical Modeling. Traditionally, pronouns have been treated as a *closed word class*. Generally, statistical models do not make as-

[17] https://sites.google.com/view/queer-in-ai

[18] https://www.linguisticsociety.org/content/committee-lgbtq-z-issues-linguistics-cozil

[19] https://www.tgns.ch

[20] All trends observed in the pre-study were confirmed in the main phase of the survey.

Paradigm	D1	D2	D3	D4	D5
Classic	✗	✗	✗	✗	✗
Bucketing	✗	✓	✓	?	✗
Delexicalization	✓	✓	✓	✓	✗
Post-hoc	✓	✓	✓	✓	✓

Table 4: Modeling paradigms and how they allow for fullfilling the desiderata D1–D5.

	Train	Dev	Test	Total
PRP	64,476	7,881	8,067	80,424
PRP$	14,535	1,783	1,935	18,253
Total	79,011	9,664	10,002	98,677

Table 5: Number of pronoun replacements in the training, development, and test portion of OntoNotes 5.0 for PRP and PRP$, respectively.

sumptions about this (except if the vocabulary is manually curated). However, in models exploiting co-occurrences, e.g., via word embeddings (GloVe (Pennington et al., 2014)) or deep language models (BERT (Devlin et al., 2019)), the models will likely misrepresent underrepresented pronoun-related phenomena. Dev et al. (2021) provided an initial insight by showing that singular *they* and the neopronouns *xe* and *ze* do not have meaningful vectors in GloVe and BERT.

Bucketing. One option, previously discussed by Dev et al. (2021), is to apply bucketing, i.e., to decide on a fixed number of majority classes, e.g., *male* pronouns, *female* pronouns, and one or multiple classes for the "rest of the pronouns", e.g., *other*. The advantage of this approach is that it can map existing and novel pronouns to the *other* class. However, it still makes identity assumptions – and due to unequal representations of *main* and *other* classes, it will inevitably lead to discrimination.

No Modeling – Delexicalization. Given that the classic approach and bucketing both lead to unfair treatment of underrepresented groups, the alternative is to explicitly not model pronouns in their surface forms. This process, commonly named delexicalization, has proved helpful for other tasks where models capture misleading lexical information, e.g., fact verification (e.g., Suntwal et al., 2019), or resource-lean scenarios, e.g., cross-lingual parsing (e.g., McDonald et al., 2011).[21] In this case, the model is forced to not rely on spurious lexical cues related to gender, e.g., that *John* occurs most often with the pronoun *he*. Instead, the model learns a single representation for all pronouns and relies on other task-related conceptual and commonsense information for disambiguation.

Post-hoc Injection of Modeling Information/ Modeling at Test Time. For human-to-human interactions, several LGBTQIA+ guides recommend to (1) first try generic pronouns (e.g., singular *they*), and (2) switch to other sets of pronouns once the conversation partner communicates them. For uncommon or novel pronouns, several web pages have explicitly been set up for practicing how to use them.[22] In this work, *we propose that NLP systems should work similarly* – if technically possible and depending on the concrete sociotechnical deployment scenario. To this end, we can use intermediate training procedures (e.g., Hung et al., 2021) for pronoun-related model refinement. E.g., we can use synthetic data created through similar procedures as the ones employed on these websites. Another option is only model pronouns at test time, e.g., through simple replacement procedures.

6 The Effect of Delexicalization

In §5.3, we discussed delexicalization, i.e., not modeling lexical forms of pronouns, as one way to counter exclusion in statistical modeling and bucketing. A possible counter-argument against this approach is that omitting the surface forms will lead to poor performance on pronoun-related tasks. We experimentally quantify the loss from (fairer) delexicalization in co-reference resolution.

6.1 Experimental Setup

Task, Dataset, and Measures. We resort to co-reference resolution, a task where knowledge about pronouns and related gender assumptions play an essential role. We use the English portion of *OntoNotes 5.0* (Weischedel et al., 2012), which consists of texts annotated across five domains (news, conversational telephone speech, weblogs, USENET newsgroups, broadcast, and talk shows). We prepare three variants: (i) the original data; (ii) we *replace all pronouns* in the test set with the respective part-of-speech token, according to the Penn Treebank Project (Santorini, 1990), i.e., PRP for personal pronouns, and PRP$ for possessive pronouns. Finally, we provide a version (iii) where we replace pronouns in all splits. Note that our

[21] In fact, accounting for novel pronouns and novel ways of using pronouns is a resource-lean scenario.

[22] E.g., https://www.practicewithpronouns.com/#/?_k=66emp7

	MUC			$CEAF_{\phi4}$			B^3			AVG		
	P	**R**	**F1**	**P**	**R**	**F1**	**P**	**R**	**F1**	**P**	**R**	**F1**
(Dobrovolskii, 2021)	84.9	87.9	86.3	76.1	77.1	76.6	77.4	82.6	79.9	–	–	81.0
- reproduction	84.7	87.5	86.1	75.6	76.7	76.1	77.2	82.0	79.5	79.2	82.1	80.6
- replace test set	69.7	70.7	70.2	63.2	49.1	55.2	50.1	56.1	52.9	61.0	58.6	59.4
$\Delta_{repl.test-repr.}$	-15.0	-16.8	-15.9	-12.4	-27.6	-20.9	-27.1	-25.9	-26.6	-18.2	-23.5	-21.2
- replace all	81.6	83.1	82.4	73.08	72.9	73.0	72.3	75.3	73.7	75.7	77.1	76.4
$\Delta_{repl.all-repr.}$	-3.1	-4.4	-3.7	-2.5	-3.8	-3.1	-4.9	-6.7	-5.8	-3.5	-5.0	-4.2

Table 6: Results of the delexicalization experiment. We report the results of the RoBERTa large-based word-level co-reference resolution model from Dobrovolskii (2021), our reproduction, and variants trained or tested on versions of the data set in which we replace the pronouns. All scores were produced using the official CoNLL-2012 scorer. We report precision (P), recall (R), and F1-score (F1) for MUC, $CEAF_{\phi4}$, and B^3, respectively, as well as the averages (AVG). The rows highlighted in gray indicate the obtained losses.

strategy is pessimistic as we also replace non-3rd person pronouns, i.e., *I*, *you*, etc. We show the number of replacements in Table 5. For scoring, we use the official CoNLL-2012 scorer (Pradhan et al., 2012). We report the results in terms of MUC (Vilain et al., 1995), B^3 (Bagga and Baldwin, 1998), and $CEAF_{\phi4}$ (Luo, 2005) precision, recall, and F1-measure, and the averages across these scores.

Models and Baselines. We want to obtain an intuition about the tradeoffs in the delexicalization setup, not to outperform previous results. For this reason, we resort to the recently proposed word-level co-reference model (Dobrovolskii, 2021), a highly efficient model competitive with the state-of-the-art. The model consists of a separate co-reference resolution module and a separate span extraction module. In an initial step, we compute token representations from a Transformer (Vaswani et al., 2017)-based encoder through the aggregation of initial representations via learnable weights. Next, we compute co-reference relationships. We pass the token representations into an antecedent-pruning procedure based on a bilinear scoring function. This way, we obtain k antecedent candidates for each token through coarse-grained scoring. An additional feed-forward neural network computes finer-grained scores. The final antecedent score is the sum of those two scores. Finally, we select the candidate with the highest score as the antecedent. Negative scores indicate no antecedent. Tokens assumed to be part of a co-reference relationship are passed into the span extraction module. The module consists of an additional feed-forward network, followed by convolutions with two output channels (for start and end scores). For further details, see the original work. Our baseline is the model trained and evaluated on the original OntoNotes portions (*reproduction*). We compare with the evaluation of this model on the pronoun-replaced test set (*replace test set*) and a version of this model trained on the replaced training set and evaluated on the replaced test set, respectively (*replace all*).

Model Configuration, Training, and Optimization. We choose RoBERTa large (Liu et al., 2019)[23] as the base encoder and fix all other hyperparameters to the ones provided in the original implementation of Dobrovolskii (2021): the window size is set to 512 tokens, dropout rate to 0.3, the learning rate of the encoder is set to $1 \cdot 10^{-5}$ and of the task-specific layers to $3 \cdot 10^{-4}$, respectively. We train the co-reference module with a combination of the negative log marginal likelihood and binary cross-entropy as an additional regularization factor (weight set to 0.5). The span extraction module is trained using cross-entropy loss. We optimize the sum of the two losses jointly with Adam (Kingma and Ba, 2015) for 20 epochs and apply early stopping based on validation set performance (word-level F1) with a patience of 3 epochs.

6.2 Results and Discussion

We show the results in Table 6. We are roughly able to reproduce the results reported by (Dobrovolskii, 2021), confirming the effectiveness of their approach and the validity of our setup. When we replace pronouns in the test set, the results drop massively, with up to -27.6 percentage points $CEAF_{\phi4}$ recall. This decrease demonstrates the heavy reliance of this model on the lexical surface forms of the pronoun sets seen in the training. However, when we replace the pronouns in the training portion of OntoNotes with the special tokens, we can mitigate these losses by a large margin (losses up

[23] https://huggingface.co/roberta-large

to -5.8 B^3 F1, and on average -4.2 F1). These results are highly encouraging, given that a) we replaced *all* pronouns, including non-third person pronouns, and b) the model has not been trained on these placeholders in the pretraining phase. The model can not rely on possibly discriminating correlations between names or occupations and pronoun sets. It therefore represents neopronouns the same way as established pronoun sets. A delexicalization approach can increase fairness in co-reference resolution and retain high system performance. We can expect even smaller drops from a more careful selection of replacements, and, possibly, from intermediate training procedures that strengthen the representation of the placeholder tokens.

7 Conclusion

This work provides an initial overview of the plethora of current phenomena in 3rd person pronoun usage in the English language. For practical and ethical reasons, the NLP community should acknowledge the broad spectrum of possible identities and the respective manifestations in written and oral communication. Language is consistently evolving, and NLP researchers and practitioners should account for this to provide genuinely inclusive systems. Notably, pronouns, traditionally handled as a closed class of words, seem to function closer to an open class. Based on the observations from our literature search, research in non-academic, publicly-available writing, a corpus study, and a survey, we defined five desiderata. We validated those and applied them to the discussion of existing and novel modeling paradigms. Our findings raise the questions *when* and *how* to model pronouns and whether and how to *include users* in these decisions. With this work, we hope to start a broader discussion on the topic. Our study can inform future NLP research and serve as a starting point for creating novel modeling procedures. All code needed to reproduce our experiments is publicly available at `https://github.com/anlausch/pronouns`.

Acknowledgments

The work of Anne Lauscher and Dirk Hovy is funded by the European Research Council (ERC) under the European Union's Horizon 2020 research and innovation program (grant agreement No. 949944, INTEGRATOR). We thank Emily Bender, Chandler May, and Arjun Subramonian for sharing their ideas related to our project.

Further Ethical Discussion

We have described phenomena related to third-person pronouns focusing exclusively on the English language. Naturally, this work comes with several limitations. For instance, while we pointed the reader to the variety of pronoun-related phenomena in other languages, a thorough *multilingual and cross-lingual discussion* would have exceeded the scope of this manuscript. This lacuna includes the discussion of neopronouns in other languages. Similarly, while we acknowledge identities beyond binary gender and otherkin identities, due to our focus on pronouns, we did not investigate *other identity-related terms*. This aspect includes their handling in language technology and the range of issues related to identity-exclusivity.

References

Amit Bagga and Breck Baldwin. 1998. Algorithms for scoring coreference chains. In *Proc. Linguistic Coreference Workshop at the first Conf. on Language Resources and Evaluation (LREC)*, pages 563–566, Granada, Spain.

Soumya Barikeri, Anne Lauscher, Ivan Vulić, and Goran Glavaš. 2021. RedditBias: A real-world resource for bias evaluation and debiasing of conversational language models. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 1941–1955, Online. Association for Computational Linguistics.

Solon Barocas, Kate Crawford, Aaron Shapiro, and Hanna Wallach. 2017. The problem with bias: Allocative versus representational harms in machine learning. In *9th Annual Conference of the Special Interest Group for Computing, Information and Society*.

Su Lin Blodgett, Solon Barocas, Hal Daumé III, and Hanna Wallach. 2020. Language (technology) is power: A critical survey of "bias" in NLP. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 5454–5476, Online. Association for Computational Linguistics.

Tolga Bolukbasi, Kai-Wei Chang, James Y. Zou, Venkatesh Saligrama, and Adam Tauman Kalai. 2016. Man is to computer programmer as woman is to homemaker? debiasing word embeddings. In *Advances in Neural Information Processing Systems 29: Annual Conference on Neural Information Processing Systems 2016, December 5-10, 2016, Barcelona, Spain*, pages 4349–4357.

Shikha Bordia and Samuel R. Bowman. 2019. Identifying and reducing gender bias in word-level language

models. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Student Research Workshop*, pages 7–15, Minneapolis, Minnesota. Association for Computational Linguistics.

Judith Butler. 1990. *Gender trouble*, 1st edition. Routledge Classics, New York, NY, USA.

Judith Butler. 2004. *Undoing Gender*, 1st edition. Routledge, New York, NY, USA.

Yang Trista Cao and Hal Daumé III. 2021. Toward gender-inclusive coreference resolution: An analysis of gender and bias throughout the machine learning lifecycle*. *Computational Linguistics*, 47(3):615–661.

Katharine A Cherry-Reid. 2020. *Music to Our Ears: Using a Queer Folk Song Pedagogy to do Gender and Sexuality Education*. Ph.D. thesis, University of Toronto (Canada).

Kirby Conrod. 2019. *Pronouns Raising and Emerging*. Ph.D. thesis, University of Washington.

Sunipa Dev, Tao Li, Jeff M. Phillips, and Vivek Srikumar. 2020. On measuring and mitigating biased inferences of word embeddings. In *The Thirty-Fourth AAAI Conference on Artificial Intelligence, AAAI 2020, The Thirty-Second Innovative Applications of Artificial Intelligence Conference, IAAI 2020, The Tenth AAAI Symposium on Educational Advances in Artificial Intelligence, EAAI 2020, New York, NY, USA, February 7-12, 2020*, pages 7659–7666. AAAI Press.

Sunipa Dev, Masoud Monajatipoor, Anaelia Ovalle, Arjun Subramonian, Jeff Phillips, and Kai-Wei Chang. 2021. Harms of gender exclusivity and challenges in non-binary representation in language technologies. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 1968–1994, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, Minneapolis, Minnesota. Association for Computational Linguistics.

Vladimir Dobrovolskii. 2021. Word-level coreference resolution. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 7670–7675, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Hila Gonen and Yoav Goldberg. 2019. Lipstick on a pig: Debiasing methods cover up systematic gender biases in word embeddings but do not remove them. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 609–614, Minneapolis, Minnesota. Association for Computational Linguistics.

Ove Grandstrand. 1998. Identity and deception in the virtual community. In Peter Kollock and Marc Smith, editors, *Communities in Cyberspace*, 1st edition, chapter 2. Routledge, London, UK.

Luise A Hercus. 1994. *A grammar of the Arabana-Wangkangurru language, Lake Eyre Basin, South Australia (Pacific linguistics. Series C)*, 1st edition. Dept. of Linguistics, Research School of Pacific and Asian Studies, Australian National University.

Jerry R Hobbs. 1978. Resolving pronoun references. *Lingua*, 44(4):311–338.

Chia-Chien Hung, Anne Lauscher, Simone Paolo Ponzetto, and Goran Glavaš. 2021. DS-TOD: Efficient domain specialization for task oriented dialog. *arXiv preprint arXiv:2110.08395*.

Emiko S. Kashima and Yoshihisa Kashima. 1998. Culture and language: The case of cultural dimensionsand personal pronoun use. *Journal of Cross-Cultural Psychology*, 29(3):461–486.

Os Keyes, Chandler May, and Annabelle Carrell. 2021. You keep using that word: Ways of thinking about gender in computing research. *Proc. ACM Hum.-Comput. Interact.*, 5(CSCW1).

Diederik P. Kingma and Jimmy Ba. 2015. Adam: A method for stochastic optimization. In *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*.

Lex Konnelly and Elizabeth Cowper. 2020. Gender diversity and morphosyntax: An account of singular they. *Glossa: a journal of general linguistics*, 5(1).

Helene Seltzer Krauthamer. 2021. *The Great Pronoun Shift: The Big Impact of Little Parts of Speech*, 1st edition. Routledge.

Keita Kurita, Nidhi Vyas, Ayush Pareek, Alan W Black, and Yulia Tsvetkov. 2019. Measuring bias in contextualized word representations. In *Proceedings of the First Workshop on Gender Bias in Natural Language Processing*, pages 166–172, Florence, Italy. Association for Computational Linguistics.

Anne Lauscher and Goran Glavaš. 2019. Are we consistently biased? multidimensional analysis of biases in distributional word vectors. In *Proceedings of the Eighth Joint Conference on Lexical and Computational Semantics (*SEM 2019)*, pages 85–91, Minneapolis, Minnesota. Association for Computational Linguistics.

Anne Lauscher, Goran Glavaš, Simone Paolo Ponzetto, and Ivan Vulić. 2020. A general framework for implicit and explicit debiasing of distributional word vector spaces. In *Proceedings of the AAAI Conference on Artificial Intelligence*, pages 8131–8138.

Anne Lauscher, Tobias Lueken, and Goran Glavaš. 2021. Sustainable modular debiasing of language models. In *Findings of the Association for Computational Linguistics: EMNLP 2021*, pages 4782–4797, Punta Cana, Dominican Republic. Association for Computational Linguistics.

Joseph P Laycock. 2012. "We are spirits of another sort": Ontological rebellion and religious dimensions of the otherkin community. *Nova Religio: The Journal of Alternative and Emergent Religions*, 15(3):65–90.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Roberta: A robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692*.

Xiaoqiang Luo. 2005. On coreference resolution performance metrics. In *Proceedings of Human Language Technology Conference and Conference on Empirical Methods in Natural Language Processing*, pages 25–32, Vancouver, British Columbia, Canada. Association for Computational Linguistics.

Amin Maalouf. 2000. *On Identity*, 1st edition. Vintage. Translated from French by Barbara Bray.

Ryan McDonald, Slav Petrov, and Keith Hall. 2011. Multi-source transfer of delexicalized dependency parsers. In *Proceedings of the 2011 Conference on Empirical Methods in Natural Language Processing*, pages 62–72, Edinburgh, Scotland, UK. Association for Computational Linguistics.

Sebastian McGaughey. 2020. Understanding neopronouns. *The Gay & Lesbian Review Worldwide*, 27(2):27–29.

John C McKay. 1993. On the term "pronoun" in italian grammars. *Italica*, 70(2):168–181.

Ehm Hjorth Miltersen. 2016. Nounself pronouns: 3rd person personal pronouns as identity expression. *Journal of Language Works-Sprogvidenskabeligt Studentertidsskrift*, 1(1):37–62.

Abd Muqit. 2012. Ideology and power relation reflected in the use of pronoun in osama bin laden's speech text. *International Journal of Social Science and Humanity*, 2(6):557.

Jeffrey Pennington, Richard Socher, and Christopher Manning. 2014. GloVe: Global vectors for word representation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1532–1543, Doha, Qatar. Association for Computational Linguistics.

David M Perlmutter. 1968. *Deep and surface structure constraints in syntax.* Ph.D. thesis, Massachusetts Institute of Technology.

Paul Postal, David A Reibel, and Sanford A Schane. 1969. On so-called pronouns in english. *Readings in English transformational grammar*, pages 12–25.

Sameer Pradhan, Alessandro Moschitti, Nianwen Xue, Olga Uryupina, and Yuchen Zhang. 2012. Conll-2012 shared task: Modeling multilingual unrestricted coreference in ontonotes. In *Joint Conference on EMNLP and CoNLL-Shared Task*, pages 1–40.

Chase Wesley Raymond. 2016. Linguistic reference in the negotiation of identity and action: Revisiting the t/v distinction. *Language*, 92:636–670.

Julie Roberts. 2020. 2019 word of the year is "(my) pronouns," word of the decade is singular "they" as voted by american dialect society. Press Release, American Dialect Society.

Rachel Rudinger, Jason Naradowsky, Brian Leonard, and Benjamin Van Durme. 2018. Gender bias in coreference resolution. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 2 (Short Papers)*, pages 8–14, New Orleans, Louisiana. Association for Computational Linguistics.

Beatrice Santorini. 1990. Part-of-speech tagging guidelines for the penn treebank project. Technical Report (3rd Version), University of Pennsylvania, School of Engineering and Applied Science.

Deven Santosh Shah, H. Andrew Schwartz, and Dirk Hovy. 2020. Predictive biases in natural language processing models: A conceptual framework and overview. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 5248–5264, Online. Association for Computational Linguistics.

Michael Spivak. 1990. *The Joy of TeX: A Gourmet Guide to Typesetting with the AMSTeX Macro Package*, 2nd edition. American Mathematical Society.

Susan Stryker. 2017. *Transgender history: The roots of today's revolution*, 2nd edition. Seal Press.

Sandeep Suntwal, Mithun Paul, Rebecca Sharp, and Mihai Surdeanu. 2019. On the importance of delexicalization for fact verification. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 3413–3418, Hong Kong, China. Association for Computational Linguistics.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in Neural Information Processing Systems 30: Annual Conference on Neural*

Information Processing Systems 2017, December 4-9, 2017, Long Beach, CA, USA, pages 5998–6008.

Marc Vilain, John D Burger, John Aberdeen, Dennis Connolly, and Lynette Hirschman. 1995. A model-theoretic coreference scoring scheme. In *Sixth Message Understanding Conference (MUC-6): Proceedings of a Conference Held in Columbia, Maryland, November 6-8, 1995*.

Kellie Webster, Marta Recasens, Vera Axelrod, and Jason Baldridge. 2018. Mind the GAP: A balanced corpus of gendered ambiguous pronouns. *Transactions of the Association for Computational Linguistics*, 6:605–617.

Ralph Weischedel, Sameer Pradhan, Lance Ramshaw, Jeff Kaufman, Michelle Franchini, Mohammed El-Bachouti, Nianwen Xue, Martha Palmer, Jena D Hwang, Claire Bonial, et al. 2012. Ontonotes release 5.0.

Threat Scenarios and Best Practices for Neural Fake News Detection

Artidoro Pagnoni[†] and **Martin Garciarena**[◊] and **Yulia Tsvetkov**[†]
[†]Paul G. Allen School of CSE, University of Washington [◊] SRI International
{artidoro, yuliats}@cs.washington.edu, martin.graciarena@sri.com

Abstract

In this work, we discuss different threat scenarios from neural fake news generated by state-of-the-art language models. Through our experiments, we assess the performance of generated text detection systems under these threat scenarios. For each scenario, we also identify the *minimax* strategy for the detector that minimizes its worst case performance. This constitutes a set of best practices that practitioners can rely on. In our analysis, we find that detectors are prone to shortcut learning (i.e., lack of out-of-distribution generalization) and discuss approaches to mitigate this problem and improve detectors more broadly. Finally, we argue that strong detectors should be released along with new generators.[1]

1 Introduction

During the COVID-19 pandemic, an overabundance of inaccurate information made it hard for people to find reliable guidance when they needed it, creating the first global *infodemic*. This widespread presence of online fake news poses a risk for the principles of the global information and communication space, which is considered to be a public good (Stiglitz et al., 1999). Widely available and reliable information is necessary for the function of a democratic state and is seen by UNESCO as vital to ensure "public participation and civic space".[2]

Neural generation systems unlock new possibilities for disinformation campaigns to generate large quantities of targeted content (Zellers et al., 2019; Radford et al., 2019; Keskar et al., 2019; Brown et al., 2020). Recent progress in pretrained language models enables the generation of fluent, original text that can be easily confused with human-written text. Progressively more complex use of such language models can be achieved depending on the technical expertise and resources of the users (i.e., access to the language model, capability to run existing systems, to finetune them, or to develop a new generator and train it). This is a pressing problem, and it is reported that certain actors spend substantial monetary resources to spread misinformation and influence public opinion, for example during the COVID 19 pandemic [3] and the US Presidential Elections of 2016 (Badawy et al., 2018; Stiff and Johansson, 2021). Although the majority of online disinformation is currently manually written (Vargo et al., 2018), the rapid progress of language generation systems along with their increased availability could attract these actors for whom the financial barriers do not pose problems. To prevent the spread of artificial and potentially malicious content online, it is necessary to develop automated detection systems that can distinguish human and generated text reliably.

In this work, we discuss the problem of machine generated text detection (Jawahar et al., 2020), by identifying and analyzing potential scenarios of malicious text generation (i.e., threat scenarios) based on (1) state-of-the-art models in neural language generation, and (2) previous work that describes how these techniques can be used to forge content (Ranade et al., 2021; Gupta et al., 2020; Buchanan et al., 2021).

Our analysis focuses on the interaction of generators and detectors for each scenario across their technical capabilities and associated costs. Knowing the limitations of current detection systems can help policymakers identify the threats posed by different actors by assessing their technical expertise and resources.

The contributions of this work are: (1) identifying and assessing threat scenarios for the use of pretrained language models to spread disinforma-

[1]Resources including generators, detectors, and generated and pristine news articles are made available at https://github.com/artidoro/detect-gentext

[2]UNESCO report "Information as a public good"

[3]New York Times: "Disinformation for Hire, a Shadow Industry, Is Quietly Booming"

Proceedings of the 29th International Conference on Computational Linguistics, pages 1233–1249
October 12–17, 2022.

tion, based on associated costs and availability of generators, (2) analyzing the performance of current methods – discriminators trained to defend against these threat scenarios, and (3) identifying challenges and future research directions to mitigate potential threats.

2 Problem Description

In this section we introduce the problem of detecting generated text and describe state-of-the-art methods for both generators and detectors.

2.1 Generators

We consider a generator to be a model capable of generating fluent text. Currently, state-of-the-art generators rely on neural language models pretrained on large text corpora (Bengio et al., 2000). Language models generate text autoregressively, by sampling each word from a probability distribution over a fixed vocabulary given the word's context.

Architecture and Training Objective The core of the architecture of neural language models has converged to Transformer models (Vaswani et al., 2017) with small variations (Press et al., 2022). One distinction between language models is their training objective (ex: *masked* (Devlin et al., 2018) and *causal* (Radford et al., 2019) language models). Causal language models are generally the ones employed for text generation (Li et al., 2021).

Training Data Language models are trained to mimic the distribution of the language in their training data. This has an impact on the kind of text they are able to generate. Language models pick up social biases and other unwanted artifacts of the text they were trained on (Sun et al., 2019; Weidinger et al., 2021; Field et al., 2021). Most language models are pretrained on large corpora of web text. They can be adapted to specific domains by finetuning on new corpora. For example, a model could be finetuned to the news domain to generate text in the style of news articles (Zellers et al., 2019).

Sampling From the Language Model There are different strategies to sample from the distribution produced by the language model. These strategies lead to different styles of text. *Greedy sampling* amounts to generating the highest probability word. This strategy, however, leads to deterministic generations. *Random sampling* involves sampling from the entire distribution over the vocabulary. In some cases, this can lead to ungrammatical generations. To mitigate this issue *top-k sampling* (Fan et al., 2018) generates from the top-k most likely words according to the language model. Finally, *nucleus sampling* (Holtzman et al., 2019) samples from the set of words that collectively accounts for a probability mass p under the language model. Orthogonal to sampling strategies are decoding algorithms that aim to maximize the global probability over the full text sequence. Most generators compose text sequences one word at a time without guarantees of finding a globally optimal sequence. Beam search is a heuristic method that is often employed to maintain b candidate generations and pick the one with the highest overall likelihood.

2.2 Detectors

In this work, we define detectors as models that are trained to distinguish machine-generated and human-written text. Research on detectors for generated text is motivated by two main goals. On one hand, a better understanding of the differences between generated and human text can be used to improve generated systems. On the other hand, better detection systems could help mitigate the potential negative societal impact that machine generated text can have (Jawahar et al., 2020).

Previous work has pointed out stylistic differences between generated and human text. Differences in fluency, lexical, and syntactic novelty have all been shown to help differentiate between generated and human written text. Here, we describe different lines of work on automated detection systems of machine generated text with a particular focus on pretrained classifiers which have been shown to work best so far (Gehrmann et al., 2019; Dugan et al., 2020).

Human-machine collaboration Dugan et al. (2020) show that humans can easily be fooled when trying to detect the boundary between human and generated text. Their work highlights the difficulty of the detection task, but also shows that humans are capable of identifying generated text in a long document (beyond a couple of sentences). Gehrmann et al. (2019) studied the difference between human and machine generated text. They showed that generated text tends to opt for more common phrases, thus favoring fluency over novelty. They built a tool, GLTR, that helps humans identify generated text, by visualizing the likelihood and rank of words in a text under a language model.

Description	Costs	Expertise	Threat	Detect. Acc.
Online playgrounds	0	Interacting with an online form	Execute Generator	92.8%
Paid API	0-1K	Prompting and querying API	Execute Generator	92.8%
Execution of LM	50k-130k	Hardware setup, DL execution experience	Execute Generator	92.8%
Finetuning of LM	100k+	Hardware setup, DL training expertise	Finetune Generator	74.2%
Training of LM	100k-1M+	Hardware setup, optimization, training of LM	Train Generator	64.8%

Table 1: Threat scenarios, costs, and *minimax* detector performance (see discussion in section 6).

Stylometry Aiming to identify differences in the style of generated and human text, some approaches directly measure stylistic features of text. These approaches have broadly been termed stylometry. Some features used by Fröhling and Zubiaga (2021) include repetitiveness, lack of purpose, and readability. Despite some success, previous work pointed out the limitations of stylometry in detecting machine-generated fake news (Schuster et al., 2020). State-of-the-art performance on this task is achieved by pretrained language models finetuned to the classification task (Uchendu et al., 2021).

Pretrained classifiers Previous work has studied how well pretrained classifiers can detect generated text (Solaiman et al., 2019; Zellers et al., 2019; Ippolito et al., 2020; Bakhtin et al., 2021; Uchendu et al., 2021). Solaiman et al. (2019) show that finetuning RoBERTa can detect GPT-2 (Radford et al., 2019) with 95% accuracy and that the performance transfers across decoding strategies and to smaller generators. Ippolito et al. (2020) show that detectors perform best when humans are fooled because decoding strategies have to compromise between fluency and lexical and syntactic novelty. Fluency errors are easy to detect by humans, while lexical novelty is difficult. On the other hand, lexical novelty is what classifiers are able to identify.

3 Threat Scenarios

A given generator can be used in various ways, based on domain expertise and budget, resulting in different styles of generated text. We describe threat scenarios by identifying possible uses of pretrained LMs and their associated costs (technical expertise and monetary resources required).

Interactive Playgrounds and Paid APIs Most language models have been open-sourced and the model parameters can be downloaded. However, many also have free online playgrounds that simplify the user-interaction[4] and require little to no programming expertise. The latest models coming from the private sector (e.g., GPT3 (Brown et al., 2020)) are usually not available publicly or are only accessible through paid APIs and online playgrounds. With some delay, equally powerful models are often released publicly (e.g., OPT (Zhang et al., 2022)).

These playgrounds are either free or very cheap. Even the paid GPT3 playground's most powerful model only costs $0.06 for 1000 tokens. Generating text through the playgrounds is time consuming and requires heavy intervention by the user who needs to write prompts and ensure that the generation is sensible. However, this method could be used to speed up the process of writing fake content by offloading some portion of the writing to the language model. APIs extend the playgrounds by allowing generation to be performed programmatically. This increases the potential scale of the generation process. Targeted generations through prompting still need careful handcrafting.

Execution and Sampling Variations The next barrier to using pretrained language models is being able to execute them. Due to their scale language models require hardware accelerators (ex: GPUs and TPUs). Thanks to advances in model parallelization (Rasley et al., 2020; Rajbhandari et al., 2020) it is no longer required to store the full model in the memory of a single piece of hardware. This makes it possible to execute large models on standard GPU hardware.

Once the model is executing, it is simple to vary hyperparameters of the generation process such as sampling strategy, temperature, beam search, and repetition penalty. These parameters are generally built into the software libraries for text generation and lead to stylistic differences in the generated text which can reduce the effectiveness of detection systems. Having the monetary and technical resources to execute the models allows generat-

[4] https://transformer.huggingface.co/

ing text at scale. However, targeted generations still require prompts either curated or automatically extracted. Previous work puts the cost of such a project including experimentation and servicing between $50,000 and $136,000 (Sharir et al., 2020). These costs will likely decrease as hardware becomes more easily accessible and software packages and cloud services simplify the process.

Finetuning To produce more sophisticated and targeted generations, it is usually necessary to adapt the pretrained language model. One common technique for adaptation of generators is finetuning the generators by continuing the pretraining (Ranade et al., 2021; Gupta et al., 2020). The result is a language model that is able to produce text in the required style, format, or domain. This requires both access to in-domain data as well as computing resources. Resources required for this are upwards of $100,000 (Sharir et al., 2020).

Training a Language Model Developing new models can lead to new and unique styles of generation. These can be hard to identify for detection systems that have not seen such examples in their training data. Training new language generation systems comes at a significant cost. The costs are inflated by the need of several training runs to experiment with different hyperparameter configurations. Previous research estimated the costs of training new models at $50k for a 110 million parameter model, $200k for a 340 million parameter model, and $1.6M for a 1.5 billion parameter model (Sharir et al., 2020). Note that these costs were estimated in 2020 and are likely going to be affected by improvements in hardware and software libraries to speed up the training process. Costs are even higher for larger models like OPT (175 billion param.) which was trained on 992 80GB A100 GPUs with experiments lasting at least two months (Zhang et al., 2022). At the time of writing, the GPU hardware alone would cost $50-100M on the public version of Google cloud (although due to private negotiations with the cloud providers the actual price is likely lower).

4 Experimental Setup

Here we describe the generators, detectors, and datasets that we use in our experiments.

Model Name	Num Param.	LAM PPL	LAM Acc	Detect. Acc.
GPT-2 sm	125M	35.1	45.99	99.49
GPT-2 md	350M	15.6	55.48	98.79
GPT-2 lg	760M	10.9	60.12	97.39
GPT-2 xl	1.5B	8.6	63.24	95.91
GPT-NEO	2.7B	5.6	62.2	95.12
GPT-3	175B	3.0	76.2	92.83

Table 2: Generators used in our experiments, the number of parameters, their perplexity, and accuracy on the LAMBADA dataset (Paperno et al., 2016), and their in-domain detection performance when training and testing with nucleus sampling ($p = 0.96$).

4.1 Generators

In this work, we focus on causal language model generators. These have been shown to be successfully employed to generate neural fake news (Zellers et al., 2019). We experiment with the four sizes of GPT-2 (Radford et al., 2019), GPT-NEO (Black et al., 2021), and GPT-3 (Brown et al., 2020). We chose these models as they have a similar training objective and cover a wide range of performances in terms of perplexity. The details of these models are shown in Table 2.

Our experiments do not include larger models (Rae et al., 2021; Smith et al., 2022; Chowdhery et al., 2022). The perplexity and accuracy on benchmarks like LAMBADA (Paperno et al., 2016) of the larger model are not significantly different. For example, the accuracy of PALM (540B parameters) on LAMBADA is 77.9 against 76.2 for GPT-3 (175B parameters). We also do not consider controllable text generation systems which can change the domain of the generated text. Finetuning is a simple alternative to change the domain and we chose it to avoid introducing other confounders. Also, previous work that experiment with controllable generation did not always find significant differences in detection performance (Stiff and Johansson, 2021).

4.2 Detectors

In this work, we experiment with classifiers based on pretrained language models and finetuned to the task of detecting generated text. In total, we experiment with seven different detectors based on ROBERTA(Liu et al., 2019), BERT(Devlin et al., 2019), ALBERT (Lan et al., 2020), and ELECTRA (Clark et al., 2020). We include different

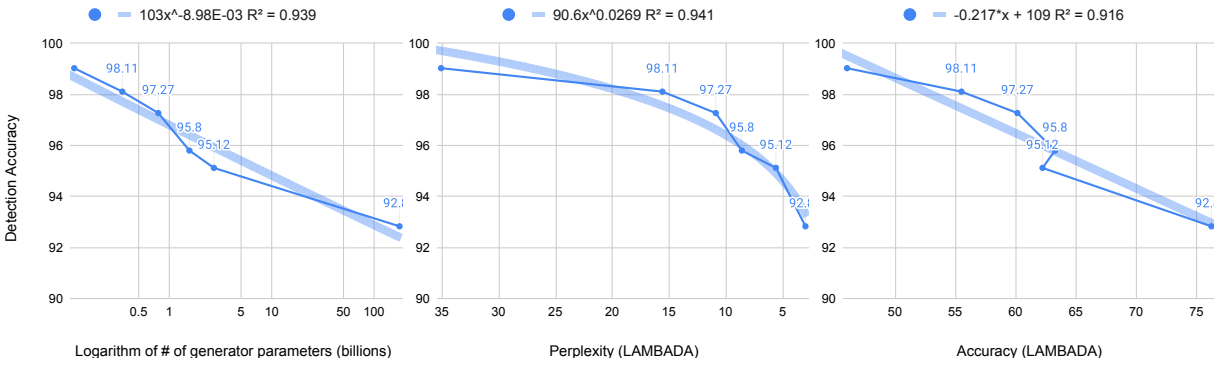

(a) (b) (c)

Figure 1: In-domain detection performance as a function of number of parameters (a), perplexity (b), and accuracy (c) of the generator (perplexity and accuracy on LAMBADA dataset). Detectors are trained and tested on nucleus sampling ($p = 0.96$).

sizes and training objectives for the pretrained language models that the detectors rely on.

4.3 Datasets

We employ existing dataset of generated text for GPT-2 models and complement it with generations from GPT-NEO and GPT-3, as well as domain adapted generations. OpenAI released both human and machine generated text.[5] The data includes generations from GPT-2 models along with part of the original training data (Solaiman et al., 2019). The generations come with three different decoding strategies (random, top-k, and nucleus sampling). We are not aware of any other models being released with sample generations. Furthermore, the pretraining data is often not available. We therefore had to generate a large training dataset of text generated by GPT-NEO and GPT-3 for our experiments. We use human prompts (15 tokens) from Webtext (released by OpenAI) and the three sampling strategies that were already used by OpenAI. The hyperparameters $k = 40$ and $p = 0.96$ were reported to be a good compromise between fluency and novelty of generations (Holtzman et al., 2019) and are used in our experiments. For GPT-3, we collected 50K training examples of generated text using nucleus sampling with $p = 0.96$. To our knowledge, this is the first collection of documents that is large enough to train a detector for GPT-3. To test generalization performance of detectors, all generators besides GPT-3 were also finetuned to the COVID domain. The dataset used for finetuning is the NELA 2020 (Gruppi et al., 2022) news corpus (the COVID split). Details of the datasets used in our experiments are available in the appendix Appendix A.

[5] https://github.com/openai/gpt-2-output-dataset

5 Results

Our experiments simulate the different threat scenarios identified in the previous section and evaluate their associated risks. Borrowing from the field of security (Shostack, 2014), we describe how "attackers" can employ generators and "defenders" train effective detectors under varying assumptions.

The first three scenarios (online playgrounds, paid APIs, and execution of LM) can be studied together as they involve using an existing generator without any domain specific tuning of the model parameters. Users can still tailor the generation style by changing decoding hyperparameters. Next, we explore the scenario of generator finetuning, focusing on the covid domain. Lastly, we simulate the training of new generators by looking at the performance of detectors when they are trained and tested on different generators. Throughout our experiments, we choose to isolate and study a specific phenomenon to determine its impact of the detection performance in isolation.

5.1 Preliminary

To set the basis for further experiments, we begin with a scenario where there is full knowledge of the attack. Here, both training and testing is performed on the same generator and decoding strategy while finetuning. This is a white-box attack which, though unrealistic, gives some useful preliminary information about the hardness of the task.

tokens	256	128	64	32
GPT-2 xl nucleus	96.4	93.09	85.7	76.37
GPT-2 xl random	95.96	94.4	87.9	76.69
GPT-2 xl $k = 40$	97.9	98.8	96.8	-

Table 3: Accuracy on shorter sequences. ROBERTA large was trained and tested on sequences of varying size in terms of tokens.

train ↓ test →	random	top-k	nucleus
random	n.a.	-28.6	-21.9
top-$k = 40$	-43.2	n.a.	-21.3
nucleus $p = 0.96$	-8.6	$+2.4$	n.a.

Table 4: Average accuracy variation when training a ROBERTA lg model on one decoding strategy and testing on another decoding strategy across the generators of interest.

Under these circumstances, the detection performance is above 90% even for the largest generator GPT-3 (Table 2). We confirm observations made by previous work (Radford et al., 2019; Ippolito et al., 2020) that larger detectors perform better (-4.6% average performance decrease with the small version of the detector across 60 generator variations tested) and that larger models are harder to detect. A contribution of our analysis is a study of the scaling laws of generator capacity and detection performance. We can perform such a study thanks to the collection of a dataset of GPT-3 generations.

In Figure 1, we observe that the detection performance is related with a power law to both the number of parameters in the generator and the generator perplexity with an exponent in the order of 10^{-3}. The relation appears to be more linear in terms of the accuracy on the LAMBADA dataset with coefficient -0.21. In all three cases, we have $R^2 > 0.9$. These relations suggest a trend with small detection performance drop as the generator complexity increases even beyond the current known generators. As a reminder, this assumes a white-box attack scenario.

Besides having full knowledge of the attack, the previous results also assumed fairly long generations. The detection performance, even under white-box attack, significantly decreases when the text sequences to distinguish are shorter. Such shorter text sequences are frequent in certain domains like in social media (Twitter has 280 character limit which is about 50 words). In Table 3, we show that on sequences of 64 tokens, roughly the maximum length of a Tweet, performance drops by 10% on nucleus and random sampling. Interestingly, on top-k the decrease in performance is not as pronounced.

5.2 Executing Known Generators

In this scenario, we assume that attackers can only employ generators that are known to the defenders, which means that the adversary can pick a decoding strategy. Since the generator is known, the defender can train the detector on data generated by the generator and used by the adversary. However, given the large number of available combinations of decoding hyperparameters, the defender cannot assume to have trained a model on the same decoding strategy that was chosen by the adversary.

In practice, we simulate this scenario by training the detector on one decoding strategy and testing it on another. In Table 4, we show that performance generally decreases when a detector is evaluated on a decoding strategy it was not trained on. This highlights how sensitive these models are to small stylistic variations of the generated text. This experiment also shows that nucleus sampling has the best generalization to other decoding strategies and that defenders should train on nucleus sampling to mitigate worst case performance drop in this scenario. Training on nucleus sampling still has an average performance change of -8.6% when testing on random sampling. Such significant reduction indicates that changes in the decoding strategy are simple ways to reduce the performance of a detector.

5.3 Finetuning Generators

To simulate the scenario of a finetuned generator, we assume that the adversary can only finetune a known generator. In our case, we finetune the generators to the COVID news domain using masked language modeling. Here, the defender does not know the target domain, therefore the detectors can only be trained in the original domain of the generator (generic web text).

In Table 5, we compare in-domain (ID) performance, where ROBERTA large is trained and tested on data from the same generator, with out-of-domain (OOD) performance, where ROBERTA lg and ELECTRA lg were trained on GPT-2 xl outputs in the web domain and tested on generated text in the COVID news domain. Here we keep the sampling strategy fixed (nucleus sampling $p = 96$)

Generators (COVID)	ID Acc.	OOD Acc.	
		RoBERTa lg	ELECTRA lg
GPT-2 sm	99.72	75.86	95.24
GPT-2 md	99.87	75.86	89.14
GPT-2 lg	98.59	90.65	94.37
GPT-2 xl	95.47	69.59	78.56
GPT-Neo	96.16	79.46	74.27
Average	97.96	81.45	86.31

Table 5: Detection accuracy in the COVID domain. We compare in-domain (ID) with out-of-domain (OOD) performance for different detectors.

to isolate the change in performance due to a shift to the COVID news domain.

We find that the out-of-domain performance is on average 11% lower for ELECTRA lg and 16% lower for RoBERTa lg compared to the in-domain performance. The ELECTRA discriminator is pretrained to identify spans of text that were replaced by a language model. This pretraining objective is closer to the task of detecting generated text than the masked language modeling objective used by RoBERTa. We hypothesize that ELECTRA performs better at generalizing to out-of-domain generated text because of its training objective.

5.4 Training New Generators

To test the performance under new and unknown generators, we make the assumption that the defender does not have access to the generator it is trying to detect. This means that the defender cannot train on the target generator.

To simulare this scenario, we test the detectors on generators they were not trained on. To isolate this component (i.e., change in generator), we do not vary the decoding strategy or the domain of the generator. In Table 6, we show that there is a drop in performance across all detectors when they are trained on a smaller generator and tested on a larger one. This drop is more pronounced when the detectors are trained on smaller models (decrease of 26.5% when a model trained on the GPT-2 small is tested on GPT-2 xl). However, when comparing very large models the difference in performance is significantly smaller (training on GPT-2 lg and testing on GPT-xl leads to only a 2.1% performance drop). We conclude that the generalization benefit from using a larger generator for training decreases once the generator is above a certain quality. This observation is in line with our study of the scaling law of the capacity of the generator under the white-box attack.

Trained on →	sm			md		lg
Tested on →	xl	lg	md	xl	lg	xl
RoBERTa large	36.8	22.3	6.9	11.8	2.8	2.0
RoBERTa base	35.5	24.6	13.3	11.3	4.0	1.9
BERT large	31.2	22.7	16.3	10.7	4.4	2.2
BERT base	16.5	11.4	12.0	6.5	3.3	1.3
ELECTRA small	15.5	11.2	9.0	8.8	5.1	1.1
ALBERT base	23.4	17.2	12.0	12.5	6.9	4.2
Avg. acc. change	26.5	18.2	11.6	10.3	4.4	2.1

Table 6: Decrease in accuracy when training on smaller generators and testing on larger ones. The change is calculated based on the performance of the discriminator tested on the same generator size it was trained on. Here we use different sizes of GPT-2 with nucleus sampling.

	GPT-2 lg	GPT-2 md	GPT-2 sm
RoBERTa lg	1.0	1.2	1.4
RoBERTa base	1.2	1.4	2.2
BERT lg	1.0	0.3	1.8
BERT base	0.8	-2.9	0.0
ELECTRA sm	0.5	-3.3	-0.4
ALBERT	1.7	-0.1	4.4

Table 7: Training on larger generators transfers to smaller ones. In this case, the discriminators trained on GPT-2 xl (nucleus sampling) maintain accuracy on the smaller generators.

Next, we test the hypothesis that training on larger generators transfers to smaller generators. In Table 7, we show the performance change when training on GPT-2 xl and testing on GPT-2 lg, md, and sm. We observe that the performance, in terms of accuracy, remains mostly unchanged across detectors with a 0.2% average improvement. This indicates that training on larger generators generalizes to the smaller ones (see Appendix A for a discussion of the GPT-3 exception).

6 Discussion

6.1 Best Practices to Detect Generated Text

For each threat scenario we identify the *minimax* strategy (Nash, 1953) that minimizes the worst case performance for the defender (see Table 1). In general, our preliminary experiments show that it is beneficial to use the best available generator for training and the largest detection model. We therefore recommend this to practitioners and assume it is done by the defender in the following scenarios.

Executing Known Generators Online playgrounds, paid APIs, and models available for download and execution offer roughly the same language

models (GPT-3 was not available for execution but OPT (Zhang et al., 2022) recently filled the gap). Our experiments as well as previous work show that nucleus sampling generalizes to other sampling techniques. When the generator is known, detectors should be trained on this sampling strategy. The detection performance of GPT-3 was the lowest at 92%.

Finetuning Known Generators When confronted with generators finetuned to new domains, we showed that ELECTRA generally performed better than other classifiers (possibly due to its training objective) and that it should be preferred over ROBERTA. The worst case detection performance for the ELECTRA large model was 74.2% with the finetuned GPT-NEO generator.

Training New Generators Finally, here we assume that the adversary has the resources required to train a new generator altogether. The worst case detection is achieved on the GPT-3 generator so we take that as the new generator. The ROBERTA large model trained on the otherwise largest available generator has a detection accuracy of 64.8%.

6.2 Detection Challenges

We found that detectors for generated text are highly prone to shortcut learning. This phenomenon seems to arise in many learning systems (Geirhos et al., 2020) and involves taking shortcuts that achieve high performance on a given benchmark or domain but fail to generalize and transfer to real-world scenarios.

Since the training domain of the generators is not always available, there can be a mismatch between the domain of the generated text and the human text that are used in the training dataset for detectors. Such small systematic differences between the generated text and the human written text can be picked up by the detectors and prevent them from generalizing to other generators or domains.

In our experiments, we observed that detection models seem to focus on stylistic confounds between the human and generated text when they are present. We observed this phenomenon with small differences in tokenization and domain. For example, the NELA dataset is provided pre-tokenized which leads to punctuation having additional white spaces compared to the generator outputs. A detector trained on this dataset achieves misleading near perfect performance (99.9% accuracy) but generalized poorly to datasets that did not have this tokenization mismatch (60% accuracy). We used SHAP (Lundberg and Lee, 2017) to find what the detector focused on and find that the human text had white spaces before punctuation due to the way it was collected and anonymized (see Figure 2). This is related to observations made in previous work which highlighted the vulnerability of detection systems to adversarial examples where simple changes in the text (e.g., changing characters) can flip the predictions of the models (Darmetko; Jun et al.; Wolff and Wolff, 2020; Gagiano et al., 2021).

Model	Acc.
Baseline (no ensemb.)	94.4
Majority Voting	95.5
Decision Tree	95.7
Logistic Reg.	**96.0**

Table 8: Ensembling specialized detectors.

6.3 Future Work

Interpretability Future work should investigate whether SHAP or other interpretability methods can be used to understand what the detectors rely on to discriminate generated and human-written text. Any findings could help further improve detectors and would also be valuable to the natural language generation community.

Ensembling We also experiment with ensembling specialized detectors instead of using a single detector. In Table 8, we show that ensembling detectors trained on different domains can lead to a more general detection system. We combine detectors trained on outputs of GPT-2 xl with three different decoding strategies. We observe that ensembling consistently leads to performance improvements over the baseline which uses a single detector trained on nucleus sampling (showed to generalize well to other sampling strategies). In this experiment, we test on data generated with all three decoding strategies.

We believe that combining specialized detectors is a promising area for future work. Interesting approaches might involved identifying the right detector to use for a given text which is related to the generator attribution task (Uchendu et al., 2020). They could also try to combine specialized detectors using model distillation (Hinton et al., 2015) into a single detector that learns from each specialized model on different training instances

depending on which generator originated them.

Better Training Data We found that generators were sensitive to generation hyperparameters. Generating high quality training datasets required some generator-specific hyperparameter tuning. The absence of automated evaluation metrics of generated text makes the process laborious and time-consuming. Future work should draw from stylometry (Fröhling and Zubiaga, 2021) or unsupervised distributional methods (Pillutla et al., 2021; Gallé et al., 2021) to help automate the process of constructing a high quality training dataset. Features used for stylometry could more generally be used to identify a domain mismatch between human and generated text which if addressed could help improve the generalization performance of detectors.

Adversarial Setup Detecting generated text involves an adversarial setup in which detectors need generated outputs during training. The best training signal for a generator comes from the best generator. However, improvements in generators leave the detectors behind. One interesting avenue for future work is to adversarially filter the training data for the detector by only selecting examples that are most similar to human text (Holtzman et al., 2018). The process is similar to simulating a better generator with a simpler one. The filtering could be repeated iteratively and rely on a detector that is progressively trained on harder examples.

7 Ethical Considerations, Limitations, and Recommendations

Although in this work we discuss the problem of detecting generated text with a particular focus on generated fake news, we would like to point out that not all generated text is fake news and most generated text does not have a malicious intent. Language models are widely used in NLP for both conditional language generation tasks like summarization, translation, and dialogue (Gatt and Krahmer, 2018), as well as unconditional generation tasks like story generation (Fan et al., 2018).

In this work, by presenting results on the detection performance of different generators, we inevitably provide information that could inform malicious actors about how to improve generators and where our detection system fail. However, we believe that the experimental evidence in this paper points to research directions which can help improve systems for the detection of generated text beyond our current capabilities.

We also would like to point out that the findings of this work are limited to the domains we focused on but the methodology is general and can be applied to other domains. In particular, the out-of-domain performance will change depending on the distance between domains. For example, we did not include any experiments with other dialects of English or other languages.

One limitation of this work is that we do not experiment with controllable generation systems (Keskar et al., 2019; Krause et al., 2021; Dathathri et al., 2020; Kumar et al., 2021). This is partly due to the high risk of dual use in developing systems that controllably generate malicious content. We point to prior work which experimented detecting generations from these systems (Stiff and Johansson, 2021).

Finally, based on the observations made in this paper, we recommend that **language models should be released along with strong detectors**. Our experiments showed that in-domain detection performance is reasonably high ($>$ 90% acc.) even for the largest generator GPT-3, while the out-of-domain performance is significantly lower (64.8% acc.). We also pointed out that variations in the domain between generated data and human data used to train discriminators can lead to shortcut learning. This suggests that **the requirement for authors of new generators should be to release data sampled from the training dataset along with sample generations** to ensure the two splits of the data have matching domains.

8 Conclusion

We provide an assessment of the current landscape of generated text detection and identify three primary threat scenarios. Through extensive experimentation simulating the identified threats, we establish the *minimax* strategies that minimize the worst case scenario for the detector. We argue that these strategies constitute best practices for practitioners. We find that, when confronted with an adversary capable of training a new generator, the worst case detection performance could be as low as 64.8%. We then discuss observed detection challenges related to shortcut learning, point to several avenues for future work, and provide recommendations for the community to release detectors along with new models.

Acknowledgements

The authors are grateful to the anonymous reviewers for their comments, and to Evangelia Spiliopoulou, Gian Marco Visani, and the members of the Tsvetshop group for their invaluable feedback. This material is based upon work supported by the DARPA CMO under Contract No. HR001120C0124. Any opinions, findings and conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily state or reflect those of the United States Government or any agency thereof.

References

Adam Badawy, Emilio Ferrara, and Kristina Lerman. 2018. Analyzing the digital traces of political manipulation: The 2016 russian interference twitter campaign. In *2018 IEEE/ACM international conference on advances in social networks analysis and mining (ASONAM)*, pages 258–265. IEEE.

Anton Bakhtin, Yuntian Deng, Sam Gross, Myle Ott, Marc'Aurelio Ranzato, and Arthur Szlam. 2021. Residual energy-based models for text. *J. Mach. Learn. Res.*, 22:40–1.

Yoshua Bengio, Réjean Ducharme, and Pascal Vincent. 2000. A neural probabilistic language model. *Advances in Neural Information Processing Systems*, 13.

Sid Black, Leo Gao, Phil Wang, Connor Leahy, and Stella Biderman. 2021. GPT-Neo: Large Scale Autoregressive Language Modeling with Mesh-Tensorflow. If you use this software, please cite it using these metadata.

Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. 2020. Language models are few-shot learners. *Advances in neural information processing systems*, 33:1877–1901.

Ben Buchanan, Andrew Lohn, Micah Musser, and Katerina Sedova. 2021. Truth, lies, and automation.

Aakanksha Chowdhery, Sharan Narang, Jacob Devlin, Maarten Bosma, Gaurav Mishra, Adam Roberts, Paul Barham, Hyung Won Chung, Charles Sutton, Sebastian Gehrmann, et al. 2022. Palm: Scaling language modeling with pathways. *arXiv preprint arXiv:2204.02311*.

Kevin Clark, Minh-Thang Luong, Quoc V. Le, and Christopher D. Manning. 2020. ELECTRA: pre-training text encoders as discriminators rather than generators. In *8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, April 26-30, 2020*. OpenReview.net.

Tomasz Darmetko. Fake or not? generating adversarial examples from language models.

Sumanth Dathathri, Andrea Madotto, Janice Lan, Jane Hung, Eric Frank, Piero Molino, Jason Yosinski, and Rosanne Liu. 2020. Plug and play language models: A simple approach to controlled text generation. In *International Conference on Learning Representations*.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, Minneapolis, Minnesota. Association for Computational Linguistics.

Liam Dugan, Daphne Ippolito, Arun Kirubarajan, and Chris Callison-Burch. 2020. RoFT: A tool for evaluating human detection of machine-generated text. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 189–196, Online. Association for Computational Linguistics.

Angela Fan, Mike Lewis, and Yann Dauphin. 2018. Hierarchical neural story generation. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 889–898, Melbourne, Australia. Association for Computational Linguistics.

Anjalie Field, Su Lin Blodgett, Zeerak Waseem, and Yulia Tsvetkov. 2021. A survey of race, racism, and anti-racism in NLP. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 1905–1925, Online. Association for Computational Linguistics.

Leon Fröhling and Arkaitz Zubiaga. 2021. Feature-based detection of automated language models: tackling gpt-2, gpt-3 and grover. *PeerJ Computer Science*, 7:e443.

Rinaldo Gagiano, Maria Myung-Hee Kim, Xiuzhen Zhang, and Jennifer Biggs. 2021. Robustness analysis of grover for machine-generated news detection. In *Proceedings of the The 19th Annual Workshop of the Australasian Language Technology Association*, pages 119–127, Online. Australasian Language Technology Association.

Matthias Gallé, Jos Rozen, Germán Kruszewski, and Hady Elsahar. 2021. Unsupervised and distributional detection of machine-generated text. *arXiv preprint arXiv:2111.02878*.

Albert Gatt and Emiel Krahmer. 2018. Survey of the state of the art in natural language generation: Core tasks, applications and evaluation. *Journal of Artificial Intelligence Research*, 61:65–170.

Sebastian Gehrmann, Hendrik Strobelt, and Alexander Rush. 2019. GLTR: Statistical detection and visualization of generated text. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics: System Demonstrations*, pages 111–116, Florence, Italy. Association for Computational Linguistics.

Robert Geirhos, Jörn-Henrik Jacobsen, Claudio Michaelis, Richard Zemel, Wieland Brendel, Matthias Bethge, and Felix A Wichmann. 2020. Shortcut learning in deep neural networks. *Nature Machine Intelligence*, 2(11):665–673.

Maurício Gruppi, Benjamin D Horne, and Sibel Adalı. 2022. Nela-gt-2021: A large multi-labelled news dataset for the study of misinformation in news articles. *arXiv preprint arXiv:2203.05659*.

Saurabh Gupta, Hong Huy Nguyen, Junichi Yamagishi, and Isao Echizen. 2020. Viable threat on news reading: Generating biased news using natural language models. In *Proceedings of the Fourth Workshop on Natural Language Processing and Computational Social Science*, pages 55–65, Online. Association for Computational Linguistics.

Geoffrey Hinton, Oriol Vinyals, Jeff Dean, et al. 2015. Distilling the knowledge in a neural network. *arXiv preprint arXiv:1503.02531*, 2(7).

Ari Holtzman, Jan Buys, Li Du, Maxwell Forbes, and Yejin Choi. 2019. The curious case of neural text degeneration. In *International Conference on Learning Representations*.

Ari Holtzman, Jan Buys, Maxwell Forbes, Antoine Bosselut, David Golub, and Yejin Choi. 2018. Learning to write with cooperative discriminators. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1638–1649, Melbourne, Australia. Association for Computational Linguistics.

Daphne Ippolito, Daniel Duckworth, Chris Callison-Burch, and Douglas Eck. 2020. Automatic detection of generated text is easiest when humans are fooled. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 1808–1822, Online. Association for Computational Linguistics.

Ganesh Jawahar, Muhammad Abdul-Mageed, and Laks Lakshmanan, V.S. 2020. Automatic detection of machine generated text: A critical survey. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 2296–2309, Barcelona, Spain (Online). International Committee on Computational Linguistics.

Evan Lim Hong Jun, Chong Wen Haw, and Chieu Hai Leong. Robustness analysis of neural text detectors.

Nitish Shirish Keskar, Bryan McCann, Lav Varshney, Caiming Xiong, and Richard Socher. 2019. CTRL - A Conditional Transformer Language Model for Controllable Generation. *arXiv preprint arXiv:1909.05858*.

Ben Krause, Akhilesh Deepak Gotmare, Bryan McCann, Nitish Shirish Keskar, Shafiq Joty, Richard Socher, and Nazneen Fatema Rajani. 2021. GeDi: Generative discriminator guided sequence generation. In *Findings of the Association for Computational Linguistics: EMNLP 2021*, pages 4929–4952, Punta Cana, Dominican Republic. Association for Computational Linguistics.

Sachin Kumar, Eric Malmi, Aliaksei Severyn, and Yulia Tsvetkov. 2021. Controlled text generation as continuous optimization with multiple constraints. *Advances in Neural Information Processing Systems*, 34:14542–14554.

Zhenzhong Lan, Mingda Chen, Sebastian Goodman, Kevin Gimpel, Piyush Sharma, and Radu Soricut. 2020. Albert: A lite bert for self-supervised learning of language representations. In *ICLR*. OpenReview.net.

Junyi Li, Tianyi Tang, Wayne Xin Zhao, and Ji-Rong Wen. 2021. Pretrained language model for text generation: A survey. In *Proceedings of the Thirtieth International Joint Conference on Artificial Intelligence, IJCAI-21*, pages 4492–4499. International Joint Conferences on Artificial Intelligence Organization. Survey Track.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Roberta: A robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692*.

Scott M Lundberg and Su-In Lee. 2017. A unified approach to interpreting model predictions. In I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, editors, *Advances in Neural Information Processing Systems 30*, pages 4765–4774. Curran Associates, Inc.

John Nash. 1953. Two-person cooperative games. *Econometrica*, 21(1):128–140.

Denis Paperno, Germán Kruszewski, Angeliki Lazaridou, Ngoc Quan Pham, Raffaella Bernardi, Sandro Pezzelle, Marco Baroni, Gemma Boleda, and Raquel Fernández. 2016. The LAMBADA dataset: Word prediction requiring a broad discourse context. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1525–1534, Berlin, Germany. Association for Computational Linguistics.

Krishna Pillutla, Swabha Swayamdipta, Rowan Zellers, John Thickstun, Sean Welleck, Yejin Choi, and Zaid Harchaoui. 2021. Mauve: Measuring the gap between neural text and human text using divergence frontiers. *Advances in Neural Information Processing Systems*, 34.

Ofir Press, Noah Smith, and Mike Lewis. 2022. Train short, test long: Attention with linear biases enables input length extrapolation. In *International Conference on Learning Representations*.

Alec Radford, Jeff Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. 2019. Language models are unsupervised multitask learners.

Jack W Rae, Sebastian Borgeaud, Trevor Cai, Katie Millican, Jordan Hoffmann, Francis Song, John Aslanides, Sarah Henderson, Roman Ring, Susannah Young, et al. 2021. Scaling language models: Methods, analysis & insights from training gopher. *arXiv preprint arXiv:2112.11446*.

Samyam Rajbhandari, Jeff Rasley, Olatunji Ruwase, and Yuxiong He. 2020. Zero: Memory optimizations toward training trillion parameter models. In *SC20: International Conference for High Performance Computing, Networking, Storage and Analysis*, pages 1–16. IEEE.

Priyanka Ranade, Aritran Piplai, Sudip Mittal, Anupam Joshi, and Tim Finin. 2021. Generating fake cyber threat intelligence using transformer-based models. In *2021 International Joint Conference on Neural Networks (IJCNN)*, pages 1–9.

Jeff Rasley, Samyam Rajbhandari, Olatunji Ruwase, and Yuxiong He. 2020. Deepspeed: System optimizations enable training deep learning models with over 100 billion parameters. In *Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, pages 3505–3506.

Tal Schuster, Roei Schuster, Darsh J. Shah, and Regina Barzilay. 2020. The limitations of stylometry for detecting machine-generated fake news. *Computational Linguistics*, 46(2):499–510.

Or Sharir, Barak Peleg, and Yoav Shoham. 2020. The cost of training nlp models: A concise overview. *arXiv preprint arXiv:2004.08900*.

Adam Shostack. 2014. *Threat Modeling: Designing for Security*, 1st edition. Wiley Publishing.

Shaden Smith, Mostofa Patwary, Brandon Norick, Patrick LeGresley, Samyam Rajbhandari, Jared Casper, Zhun Liu, Shrimai Prabhumoye, George Zerveas, Vijay Korthikanti, et al. 2022. Using deepspeed and megatron to train megatron-turing nlg 530b, a large-scale generative language model. *arXiv preprint arXiv:2201.11990*.

Irene Solaiman, Miles Brundage, Jack Clark, Amanda Askell, Ariel Herbert-Voss, Jeff Wu, Alec Radford, Gretchen Krueger, Jong Wook Kim, Sarah Kreps, et al. 2019. Release strategies and the social impacts of language models. *arXiv preprint arXiv:1908.09203*.

Harald Stiff and Fredrik Johansson. 2021. Detecting computer-generated disinformation. *International Journal of Data Science and Analytics*, pages 1–21.

Joseph E Stiglitz et al. 1999. Knowledge as a global public good. *Global public goods: International cooperation in the 21st century*, 308:308–325.

Tony Sun, Andrew Gaut, Shirlyn Tang, Yuxin Huang, Mai ElSherief, Jieyu Zhao, Diba Mirza, Elizabeth Belding, Kai-Wei Chang, and William Yang Wang. 2019. Mitigating gender bias in natural language processing: Literature review. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 1630–1640, Florence, Italy. Association for Computational Linguistics.

Adaku Uchendu, Thai Le, Kai Shu, and Dongwon Lee. 2020. Authorship attribution for neural text generation. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 8384–8395, Online. Association for Computational Linguistics.

Adaku Uchendu, Zeyu Ma, Thai Le, Rui Zhang, and Dongwon Lee. 2021. TURINGBENCH: A benchmark environment for Turing test in the age of neural text generation. In *Findings of the Association for Computational Linguistics: EMNLP 2021*, pages 2001–2016, Punta Cana, Dominican Republic. Association for Computational Linguistics.

Chris J Vargo, Lei Guo, and Michelle A Amazeen. 2018. The agenda-setting power of fake news: A big data analysis of the online media landscape from 2014 to 2016. *New media & society*, 20(5):2028–2049.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. *Advances in neural information processing systems*, 30.

Laura Weidinger, John Mellor, Maribeth Rauh, Conor Griffin, Jonathan Uesato, Po-Sen Huang, Myra Cheng, Mia Glaese, Borja Balle, Atoosa Kasirzadeh, et al. 2021. Ethical and social risks of harm from language models. *arXiv preprint arXiv:2112.04359*.

Max Wolff and Stuart Wolff. 2020. Attacking neural text detectors. *arXiv preprint arXiv:2002.11768*.

Rowan Zellers, Ari Holtzman, Hannah Rashkin, Yonatan Bisk, Ali Farhadi, Franziska Roesner, and Yejin Choi. 2019. Defending against neural fake news. *Advances in Neural Information Processing Systems*, 32:9054–9065.

Susan Zhang, Stephen Roller, Naman Goyal, Mikel Artetxe, Moya Chen, Shuohui Chen, Christopher Dewan, Mona Diab, Xian Li, Xi Victoria Lin, Todor Mihaylov, Myle Ott, Sam Shleifer, Kurt Shuster, Daniel

Configuration	Epochs	Batch size	Learning Rate	Max Seq. Len
Detection	10	64	1e-06	256
Configuration	prompt len	min len	max len	temp.	# beams	repet. pen.
Generation	10	20	None	1.0	3	1.3

Table 9: Detection and generation hyperparameters.

Simig, Punit Singh Koura, Anjali Sridhar, Tianlu Wang, and Luke Zettlemoyer. 2022. Opt: Open pre-trained transformer language models.

A Appendix

A.1 Dataset Information

NELA GT 2020 The NELA GT 2020 dataset used to adapt the generators to the COVID domain is a collection of 699,803 articles from 493 sources in English collected from 2020-01-01 to 2020-12-31 about COVID. The text of the articles in this dataset was modified so that it cannot properly be used for news consumption but can still be used for text analysis. For articles with more than 200 tokens, 7 tokens are replaced with @ every 100 tokens. For articles with fewer than 200 tokens, 5 consecutive tokens are replaced with @ every 20 tokens. We pretrain the generators on this data and sample from the generator by disallowing the generation of @ signs.

Webtext and OpenAI Generations OpenAI released 250,000 documents from the OpenWebText dataset which was used to train GPT-2. They also released 250,000 outputs of each size of GPT-2. We use this data in our experiments. The validation and test splits are 5K documents each.

Generated Data We generate data using various sizes of GPT-2 and GPT-Neo using 15 words from NELA-GT COVID 2020 and 15 words from OpenWebText for GPT-3 as prompts. For GPT-3, the total number of generated documents is 50K. We use 40K for training and 5k each for validation and testing. The total number of training documents generated on COVID are listed in Table 10. In, addition, we generate 10K validation and testing splits for each decoding technique and model type.

B Hyperparameters

We performed hyperparameter tuning on the RoBERTa large model and used them for the other models. Due to high combinatorial number of configurations, we did not tune hyperparameters for each detector. We use Adam in our experiments.

Generator configuration	# Generated documents
GPT2-sm-covid top k=40	718662
GPT2-sm-covid nucleus p=0.96	845382
GPT2-sm-covid random	1073162
GPT2-md-covid top k=40	670062
GPT2-md-covid nucleus p=0.96	433662
GPT2-md-covid random	528022
GPT2-lg-covid top k=40	66360
GPT2-lg-covid nucleus p=0.96	101838
GPT2-lg-covid random	104190
GPT2-xl-covid top k=40	116798
GPT2-xl-covid nucleus p=0.96	220790
GPT2-xl-covid random	99194
GPT-Neo-covid top k=40	160078
GPT-Neo-covid nucleus p=0.96	172786
GPT-Neo-covid random	97380

Table 10: Number of documents generated on COVID for each generation hyperparameter configuration.

We use the default causal language modeling hyperparameters of the Huggingface toolkit to adapt the generators to the COVID domain. We maximize the batch size that fits on a GPU. For the larger models we used Deepspeed stage 3 and 4 A40 GPUs.

While previous work did not comment on generation hyperparameters, we found that generation quality is highly dependent on the hyperparameters of the generation process. In the COVID domain, after adapting the generators to the NELA GT 2020 dataset, we found the use of beam search to be necessary to obtain fluent text. In section 6, we discuss ideas which could help future work perform hyperparameter tuning more extensively and efficiently.

B.1 Detailed Results

For reference, we report the individual results presented throughout the paper without aggregations. On top of the accuracy metric, we also report F1 and equal error rate.

While in this work and in the literature it was observed that training on larger generators tends to generalize to smaller generators, we observe here a behavior that contradicts this trend. The RoBERTa large detector trained on GPT-3 outputs has relatively low performance on GPT-2 outputs (see Table 19). Future work should explore this inconsistency and establish if it is due to minor stylistic differences in the outputs of GPT-3 and GPT-3, and therefore to the general problem of shortcut learning, or to an inherent difference in the generated text from GPT-2 and GPT-3.

Trained on →	GPT-2 xl random			GPT-2 xl top k=40			GPT-2 xl nucleus p=96			GPT-2 lg nucleus p=96			GPT-2 md nucleus p=96			GPT-2 sm nucleus p=96		
Tested on ↓	Acc.	F1	EER	Acc.	F1	EER	Acc.	F1	EER	Acc.	F1	EER	Acc.	F1	EER	Acc.	F1	EER
GPT-2 md covid random	69.23	65.77	34.00	60.85	42.69	43.36	78.56	80.98	27.24	78.61	81.51	28.21	78.58	79.62	24.08	74.77	70.21	31.04
GPT-2 md covid top k=40	50.08	29.45	49.95	70.20	61.63	36.04	81.22	83.73	26.13	80.00	82.91	27.63	79.46	80.63	23.70	75.94	71.98	29.78
GPT-2 md covid nucleus p=96	58.56	47.70	43.95	66.40	54.52	39.23	80.12	82.60	26.58	79.48	82.39	27.85	79.25	80.39	23.79	75.70	71.63	30.03
GPT-2 xl random	81.28	78.30	25.46	46.95	7.79	51.65	57.13	56.28	43.14	60.31	60.41	39.74	64.90	60.08	38.00	65.92	54.62	39.37
GPT-2 xl top k=40	48.88	5.09	50.58	94.61	94.88	9.59	78.80	82.29	29.35	79.11	82.45	28.92	84.39	85.47	20.05	84.11	82.97	19.93
GPT-2 xl nucleus p=96	55.10	25.27	47.16	65.57	54.79	39.46	74.86	78.29	31.10	75.48	78.76	30.53	77.57	77.70	22.75	76.39	72.41	29.52
GPT-2 lg random	85.05	83.39	20.78	46.72	7.02	51.77	59.94	60.28	40.23	64.10	65.49	36.95	69.22	66.63	33.36	71.01	63.85	34.95
GPT-2 lg top k=40	49.18	6.17	50.43	94.01	94.27	9.69	79.22	82.70	29.17	79.61	82.94	28.71	86.12	87.28	19.46	88.08	87.74	13.90
GPT-2 lg nucleus p=96	58.77	35.33	44.92	64.19	52.11	40.57	75.65	79.11	30.73	76.77	80.10	29.94	80.76	81.46	21.40	81.49	79.59	23.46
GPT-2 md random	93.89	93.82	7.08	45.40	2.47	52.45	59.18	59.22	40.84	64.75	66.33	36.51	81.02	81.76	21.30	77.91	74.64	27.82
GPT-2 md top k=40	49.75	8.19	50.13	93.02	93.26	9.87	79.06	82.55	29.24	79.54	82.88	28.74	87.69	88.88	18.96	88.45	88.17	13.29
GPT-2 md nucleus p=96	69.45	58.95	37.13	57.97	38.69	45.11	71.99	75.19	32.52	74.13	77.32	31.17	84.03	85.08	20.18	80.88	78.77	24.24
GPT-2 sm random	95.14	95.15	4.97	45.67	3.41	52.31	65.35	67.39	36.36	70.91	73.76	32.82	78.43	78.74	22.37	90.57	90.55	9.63
GPT-2 sm top k=40	53.54	20.61	48.07	91.63	91.81	10.13	79.40	82.88	29.10	79.89	83.22	28.60	86.80	87.98	19.24	94.86	95.06	8.49
GPT-2 sm nucleus p=96	76.64	71.38	30.52	58.57	40.09	44.70	74.90	78.33	31.08	77.11	80.44	29.78	81.31	82.09	21.19	92.86	93.00	8.82

Table 11: Performance of the detector based on BERT base for different training and testing combinations of hyperparameters and domains.

Trained on →	GPT-2 xl random			GPT-2 xl top k=40			GPT-2 xl nucleus p=96			GPT-2 lg nucleus p=96			GPT-2 md nucleus p=96			GPT-2 sm nucleus p=96		
Tested on ↓	Acc.	F1	EER	Acc.	F1	EER	Acc.	F1	EER	Acc.	F1	EER	Acc.	F1	EER	Acc.	F1	EER
GPT-2 md covid random	70.20	65.88	33.88	64.06	52.88	40.46	77.37	78.00	24.09	76.92	74.54	27.32	62.79	47.61	41.90	64.58	49.26	40.91
GPT-2 md covid top k=40	52.58	31.96	48.40	77.26	74.58	27.49	83.18	84.51	21.71	84.45	84.16	16.77	69.26	60.34	36.72	58.80	35.63	44.89
GPT-2 md covid nucleus p=96	60.18	48.50	43.00	71.04	65.21	34.24	80.79	81.91	22.63	81.62	80.72	21.08	66.89	55.92	38.72	60.57	40.06	43.73
GPT-2 xl random	86.19	84.47	20.36	45.65	11.87	52.46	70.19	71.77	31.84	74.20	72.76	28.12	77.70	73.97	28.47	67.38	53.44	39.13
GPT-2 xl top k=40	51.63	11.07	49.15	91.94	92.53	13.82	81.62	84.29	26.41	88.03	88.97	17.51	86.45	85.65	17.22	67.57	53.84	38.98
GPT-2 xl nucleus p=96	59.51	34.98	44.58	71.74	67.80	32.54	79.32	81.97	27.35	84.69	85.45	18.57	82.18	80.23	23.12	65.64	49.71	40.43
GPT-2 lg random	90.90	90.28	13.70	45.27	10.70	52.67	73.84	76.05	29.88	79.56	79.57	20.48	84.52	83.26	19.99	74.40	66.79	33.27
GPT-2 lg top k=40	53.59	17.64	48.08	91.79	92.38	13.86	82.03	84.70	26.25	89.16	90.11	17.18	91.49	91.44	8.96	75.52	68.70	32.23
GPT-2 lg nucleus p=96	65.47	49.39	40.54	70.34	65.66	34.02	80.34	83.01	26.92	86.83	87.73	17.88	88.38	87.94	14.24	74.16	66.37	33.49
GPT-2 md random	96.21	96.17	4.72	43.01	3.46	53.84	75.34	77.73	29.14	82.71	83.25	19.26	92.64	92.68	7.87	84.65	82.42	22.36
GPT-2 md top k=40	57.69	30.01	45.71	91.51	92.10	13.92	81.83	84.50	26.33	88.90	89.85	17.26	94.26	94.38	7.62	80.33	76.30	27.36
GPT-2 md nucleus p=96	79.31	74.79	28.43	61.03	49.42	42.44	79.61	82.27	27.23	86.17	87.03	18.09	92.84	92.90	7.83	80.55	76.63	27.12
GPT-2 sm random	97.75	97.76	2.73	43.85	6.21	53.41	79.16	81.81	27.42	86.97	87.88	17.84	93.81	93.92	7.69	96.75	96.73	3.78
GPT-2 sm top k=40	76.05	69.61	31.70	91.22	91.81	13.99	82.23	84.89	26.17	89.57	90.52	17.06	95.14	95.29	7.50	97.52	97.52	2.67
GPT-2 sm nucleus p=96	90.50	89.81	14.31	62.75	52.71	41.05	81.09	83.77	26.62	88.45	89.40	17.39	93.84	93.95	7.68	96.81	96.79	3.66

Table 12: Performance of the detector based on BERT large for different training and testing combinations of hyperparameters and domains.

Trained on →	GPT-2 xl random			GPT-2 xl top k=40			GPT-2 xl nucleus p=96			GPT-2 lg nucleus p=96			GPT-2 md nucleus p=96		
Tested on ↓	Acc.	F1	EER	Acc.	F1	EER	Acc.	F1	EER	Acc.	F1	EER	Acc.	F1	EER
GPT-2 md covid random	68.81	58.02	37.57	74.88	67.64	32.81	85.35	84.04	19.63	80.32	76.70	26.87	63.24	44.61	42.08
GPT-2 md covid top k=40	59.62	37.97	44.34	81.91	78.63	25.59	89.98	89.61	12.68	85.54	83.87	20.54	63.21	44.53	42.11
GPT-2 md covid nucleus p=96	64.08	48.37	41.25	79.80	75.53	27.91	89.15	88.65	14.02	85.03	83.22	21.20	64.40	47.26	41.27
GPT-2 xl random	96.50	96.42	5.51	62.83	42.56	42.48	94.19	94.17	6.15	94.79	94.64	7.58	93.27	92.88	11.00
GPT-2 xl top k=40	78.14	72.46	30.08	98.56	98.57	1.86	95.85	95.90	5.29	95.97	95.90	5.46	84.56	82.01	23.08
GPT-2 xl nucleus p=96	83.17	80.06	24.71	87.80	86.40	18.67	94.70	94.71	5.41	94.99	94.86	7.23	86.80	85.01	20.29
GPT-2 lg random	97.67	97.64	3.35	63.99	45.33	41.69	95.26	95.29	5.35	96.59	96.56	4.31	96.44	96.36	5.60
GPT-2 lg top k=40	90.95	90.18	14.58	98.74	98.75	1.86	96.69	96.76	5.20	98.08	98.09	2.40	95.08	94.89	7.99
GPT-2 lg nucleus p=96	91.86	91.26	13.22	89.93	89.03	15.69	96.02	96.08	5.27	97.40	97.40	2.77	94.96	94.76	8.20
GPT-2 md random	98.70	98.70	1.36	55.69	23.03	46.92	95.70	95.75	5.30	97.36	97.35	2.85	98.31	98.30	2.10
GPT-2 md top k=40	96.97	96.91	4.65	98.65	98.66	1.86	96.81	96.88	5.19	98.28	98.29	2.39	98.43	98.43	1.87
GPT-2 md nucleus p=96	97.61	97.58	3.46	84.64	82.25	22.72	96.04	96.10	5.27	97.83	97.84	2.41	98.27	98.26	2.18
GPT-2 sm random	98.91	98.91	1.24	59.31	33.50	44.76	95.83	95.88	5.29	97.80	97.80	2.41	98.58	98.58	1.57
GPT-2 sm top k=40	98.42	98.41	1.91	98.50	98.51	1.87	96.71	96.78	5.20	98.48	98.49	2.38	98.93	98.93	1.26
GPT-2 sm nucleus p=96	98.19	98.18	2.35	86.46	84.67	20.43	95.70	95.75	5.30	98.13	98.14	2.39	98.63	98.63	1.48

Table 13: Performance of the detector based on ELECTRA large for different training and testing combinations of hyperparameters and domains.

Trained on → / Tested on ↓	GPT-2 xl random			GPT-2 xl top k=40			GPT-2 xl nucleus p=96			GPT-2 lg nucleus p=96			GPT-2 md nucleus p=96			GPT-2 sm nucleus p=96		
	Acc.	F1	EER	Acc.	F1	EER	Acc.	F1	EER	Acc.	F1	EER	Acc.	F1	EER	Acc.	F1	EER
GPT-2 md covid random	61.67	48.07	42.34	69.50	71.92	33.37	69.92	74.98	35.82	72.44	76.45	33.26	71.30	70.53	29.76	76.22	73.92	27.71
GPT-2 md covid top k=40	50.72	21.60	49.59	76.69	79.87	29.73	73.01	78.12	34.31	75.65	79.75	31.74	71.91	71.32	28.95	73.60	70.18	30.80
GPT-2 md covid nucleus p=96	55.18	33.43	46.86	73.73	76.72	31.13	71.42	76.53	35.07	74.13	78.21	32.44	71.97	71.41	28.86	75.95	73.55	28.04
GPT-2 xl random	79.00	75.31	27.67	40.67	27.80	56.88	56.02	60.96	45.20	61.31	64.44	40.38	72.96	71.47	29.21	70.55	65.31	34.22
GPT-2 xl top k=40	50.57	12.71	49.69	78.78	82.36	29.52	71.04	77.32	36.46	75.35	79.93	32.60	78.55	78.63	21.66	77.51	75.51	26.35
GPT-2 xl nucleus p=96	55.96	28.99	46.61	68.19	71.00	34.76	68.03	74.36	37.93	71.93	76.50	34.21	74.93	74.09	26.59	73.95	70.49	30.59
GPT-2 lg random	83.08	81.02	22.82	38.97	24.16	57.93	57.45	62.70	44.19	63.15	66.70	39.16	76.26	75.80	24.71	74.63	71.48	29.82
GPT-2 lg top k=40	51.53	15.84	49.17	78.85	82.43	29.50	71.29	77.56	36.34	75.88	80.45	32.37	82.05	82.72	20.25	81.69	80.93	20.64
GPT-2 lg nucleus p=96	59.28	37.68	44.52	67.38	70.04	35.24	68.56	74.89	37.66	73.08	77.67	33.65	78.65	78.75	21.62	78.22	76.47	25.44
GPT-2 md random	91.94	91.78	9.67	31.68	6.64	61.92	50.42	53.69	49.63	61.32	64.46	40.38	83.51	84.34	19.72	79.00	77.50	24.41
GPT-2 md top k=40	54.36	24.46	47.57	78.72	82.30	29.55	71.24	77.51	36.36	75.67	80.25	32.46	85.04	86.00	19.19	82.69	82.16	19.14
GPT-2 md nucleus p=96	70.46	61.40	36.08	57.88	57.62	42.22	64.73	70.94	39.68	70.30	74.79	35.03	83.72	84.57	19.64	80.41	79.32	22.48
GPT-2 sm random	93.68	93.66	6.55	33.47	11.26	61.02	57.99	63.35	43.82	66.90	71.07	36.88	84.77	85.71	19.28	88.47	88.79	13.58
GPT-2 sm top k=40	60.35	40.30	43.81	78.11	81.70	29.81	71.40	77.66	36.29	76.00	80.57	32.31	87.24	88.30	18.47	91.53	92.00	12.83
GPT-2 sm nucleus p=96	76.02	70.78	30.85	60.04	60.65	40.26	67.65	73.97	38.12	72.83	77.42	33.77	84.98	85.94	19.21	89.44	89.82	13.33

Table 14: Performance of the detector based on ELECTRA small for different training and testing combinations of hyperparameters and domains.

Trained on → / Tested on ↓	GPT-2 xl random			GPT-2 xl top k=40			GPT-2 xl nucleus p=96			GPT-2 lg nucleus p=96			GPT-2 md nucleus p=96			GPT-2 sm nucleus p=96		
	Acc.	F1	EER	Acc.	F1	EER	Acc.	F1	EER	Acc.	F1	EER	Acc.	F1	EER	Acc.	F1	EER
GPT-2 md covid random	78.58	76.33	25.99	54.06	17.68	47.84	71.86	63.04	35.20	79.00	75.03	27.99	64.68	48.89	40.93	56.05	25.64	46.67
GPT-2 md covid top k=40	55.38	33.70	46.74	58.78	31.89	45.09	72.27	63.78	34.84	78.53	74.33	28.50	57.68	31.83	45.63	54.01	19.40	47.84
GPT-2 md covid nucleus p=96	66.04	56.43	38.87	55.77	23.09	46.88	73.29	65.57	33.92	80.47	77.18	26.35	61.67	42.00	43.05	54.97	22.37	47.30
GPT-2 xl random	90.50	89.77	14.55	48.31	3.58	50.88	81.55	81.30	19.28	80.57	79.63	22.02	81.86	78.76	25.33	69.60	57.22	37.59
GPT-2 xl top k=40	53.60	17.11	48.09	97.28	97.35	5.04	90.58	91.25	14.82	91.43	91.93	13.16	87.62	86.42	18.02	61.50	38.85	43.39
GPT-2 xl nucleus p=96	62.76	42.83	42.48	62.41	44.48	42.46	88.70	89.32	15.32	88.79	89.18	13.81	84.53	82.44	22.11	63.05	42.72	42.37
GPT-2 lg random	94.68	94.52	7.80	48.34	3.69	50.86	85.47	85.83	16.26	86.07	86.19	14.55	89.36	88.55	15.52	80.04	75.51	28.07
GPT-2 lg top k=40	57.94	30.27	45.57	96.93	97.00	5.07	91.12	91.79	14.69	92.22	92.73	12.98	93.95	93.80	8.13	71.30	60.56	36.21
GPT-2 lg nucleus p=96	71.12	60.71	36.20	62.43	44.53	42.45	89.88	90.54	15.01	90.69	91.17	13.34	91.87	91.48	11.64	74.00	65.54	33.90
GPT-2 md random	97.70	97.70	2.38	47.73	1.43	51.17	87.98	88.56	15.52	88.48	88.85	13.89	95.25	95.19	5.80	89.66	88.65	16.32
GPT-2 md top k=40	68.01	54.55	38.69	96.22	96.28	5.14	91.08	91.75	14.70	92.25	92.76	12.97	96.95	96.97	3.52	80.00	75.45	28.11
GPT-2 md nucleus p=96	86.73	85.11	19.84	57.96	33.54	45.41	90.11	90.78	14.94	90.92	91.41	13.28	95.83	95.80	4.72	85.26	83.00	22.14
GPT-2 sm random	98.36	98.37	2.35	48.14	2.96	50.96	90.05	90.71	14.96	91.32	91.82	13.19	97.08	97.10	3.52	98.66	98.66	1.46
GPT-2 sm top k=40	93.91	93.68	9.12	96.62	96.68	5.10	91.38	92.05	14.62	92.49	93.00	12.92	97.92	97.95	3.46	98.77	98.77	1.45
GPT-2 sm nucleus p=96	96.18	96.12	5.11	59.66	37.90	44.32	90.89	91.56	14.74	92.05	92.56	13.02	97.18	97.20	3.51	98.59	98.59	1.46

Table 15: Performance of the detector based on ROBERTA base for different training and testing combinations of hyperparameters and domains.

Trained on → / Tested on ↓	GPT-2 xl random			GPT-2 xl top k=40			GPT-2 xl nucleus p=96			GPT-2 lg nucleus p=96			GPT-2 md nucleus p=96			GPT-2 sm nucleus p=96		
	Acc.	F1	EER	Acc.	F1	EER	Acc.	F1	EER	Acc.	F1	EER	Acc.	F1	EER	Acc.	F1	EER
GPT-2 md covid random	67.61	55.48	38.60	73.73	65.14	34.10	68.16	55.01	38.54	71.71	62.05	35.62	58.31	31.94	45.32	52.76	14.26	48.55
GPT-2 md covid top k=40	61.76	42.84	42.93	83.41	80.50	24.28	79.13	74.47	28.66	79.50	75.10	28.20	58.39	32.17	45.27	51.91	11.34	49.00
GPT-2 md covid nucleus p=96	64.45	48.92	41.01	78.98	73.93	29.12	75.87	69.25	31.91	77.51	72.01	30.25	58.73	33.08	45.06	52.02	11.75	48.94
GPT-2 xl random	96.66	96.59	5.29	58.94	35.66	44.81	94.78	94.77	5.43	94.16	94.02	7.80	93.02	92.58	11.53	76.34	69.19	32.01
GPT-2 xl top k=40	75.97	68.84	32.18	96.93	96.98	4.71	96.51	96.56	4.84	96.08	96.06	4.29	86.18	84.16	21.18	58.12	28.51	45.56
GPT-2 xl nucleus p=96	81.77	78.02	26.32	86.04	84.65	19.50	95.91	95.95	4.89	95.40	95.35	5.56	86.92	85.13	20.24	62.71	40.97	42.68
GPT-2 lg random	98.15	98.14	2.51	60.45	39.46	43.83	96.32	96.37	4.85	96.74	96.75	3.50	97.17	97.12	4.44	89.15	87.89	17.59
GPT-2 lg top k=40	88.63	87.34	17.92	97.25	97.31	4.68	97.23	97.29	4.77	97.66	97.69	3.44	95.86	95.73	6.80	70.98	59.38	36.65
GPT-2 lg nucleus p=96	91.82	91.20	13.33	87.81	86.85	17.01	96.95	97.01	4.79	97.39	97.41	3.46	95.95	95.83	6.64	77.18	70.61	31.22
GPT-2 md random	99.04	99.04	1.16	52.59	17.50	48.60	96.86	96.92	4.80	97.27	97.29	3.47	98.88	98.88	1.18	96.73	96.64	5.76
GPT-2 md top k=40	96.34	96.25	5.87	97.25	97.31	4.68	97.32	97.38	4.76	97.93	97.96	3.42	98.66	98.66	1.61	86.08	83.91	21.57
GPT-2 md nucleus p=96	97.66	97.63	3.44	80.76	77.53	26.10	97.15	97.21	4.78	97.68	97.71	3.44	98.79	98.79	1.36	92.57	92.01	12.64
GPT-2 sm random	99.18	99.18	1.15	55.68	26.82	46.82	97.17	97.23	4.77	97.75	97.78	3.43	99.18	99.18	1.05	99.52	99.52	0.50
GPT-2 sm top k=40	98.88	98.88	1.16	97.23	97.29	4.68	97.45	97.51	4.75	98.06	98.09	3.41	99.26	99.26	1.05	99.51	99.51	0.52
GPT-2 sm nucleus p=96	98.75	98.75	1.34	83.93	81.91	22.27	97.27	97.33	4.77	97.95	97.98	3.42	99.14	99.14	1.06	99.49	99.49	0.56

Table 16: Performance of the detector based on ROBERTA large for different training and testing combinations of hyperparameters and domains.

	GPT-2 sm covid nucleus p=96	GPT-2 md covid nucleus p=96	GPT-2 lg covid nucleus p=96	GPT-2 xl covid nucleus p=96	GPT-Neo covid nucleus p=96
Roberta lg Acc.	91.69	75.87	90.52	68.95	79.16
Roberta lg F1	91.26	69.25	89.85	56.70	74.64
Roberta lg EER	12.35	31.91	14.35	37.92	28.57
Electra lg Acc.	95.24	89.15	94.38	78.56	74.27
Electra lg F1	95.33	88.65	94.43	74.72	68.23
Electra lg EER	5.98	14.02	6.10	28.06	32.44

Table 17: Performance of detectors trained on the Webtext domain with examples of GPT-2 xl and tested on the COVID domain.

Detector	Trained on	Acc.	F1	EER
Roberta lg	GPT-2 xl random	58.64	30.86	45.21
Roberta lg	GPT-2 xl top k=40	59.86	38.02	44.22
Roberta lg	GPT-2 xl nucleus p=0.96	64.82	49.61	40.76
Roberta base	GPT-2 xl random	58.30	31.28	45.35
Roberta base	GPT-2 xl top k=40	54.97	25.29	47.23
Roberta base	GPT-2 xl nucleus p=0.96	63.55	54.80	40.23
Bert lg	GPT-2 xl random	57.95	30.74	45.55
Bert lg	GPT-2 xl top k=40	58.18	43.64	44.60
Bert base	GPT-2 xl random	59.28	36.63	44.59
Bert base	GPT-2 xl top k=40	55.85	33.54	46.50
Electra lg	GPT-2 xl random	57.00	26.17	46.19
Electra lg	GPT-2 xl top k=40	57.29	27.82	45.99
Electra sm	GPT-2 xl random	57.10	32.08	45.91
Electra sm	GPT-2 xl top k=40	55.23	53.71	45.09
Albert	GPT-2 xl random	60.76	51.50	42.21
Albert	GPT-2 xl top k=40	55.45	26.45	46.95

Table 18: Performance of various detectors trained on GPT-2 outputs and tested on GPT-3 with nucleus sampling $p = 0.96$

Tested on	Acc.	F1	EER
GPT-3 Davinci nucleus p=96	92.83	93.09	10.19
GPT-2 md covid random	54.35	67.35	47.58
GPT-2 md covid top k=40	52.59	65.67	48.53
GPT-2 md covid nucleus p=96	52.66	65.73	48.49
GPT-2 xl random	58.93	41.24	44.43
GPT-2 xl top k=40	56.03	34.36	46.37
GPT-2 xl nucleus p=96	52.55	25.29	48.53
GPT-2 lg random	61.69	47.27	42.44
GPT-2 lg top k=40	57.30	37.45	45.53
GPT-2 lg nucleus p=96	52.92	26.30	48.30
GPT-2 md random	78.78	76.35	26.12
GPT-2 md top k=40	61.48	46.82	42.60
GPT-2 md nucleus p=96	61.94	47.79	42.26
GPT-2 sm random	77.66	74.79	27.47
GPT-2 sm top k=40	61.81	47.52	42.35
GPT-2 sm nucleus p=96	62.88	49.73	41.54

Table 19: Performance of Roberta lg detector trained on outputs of GPT-3 with nucleus sampling and tested on outputs of GPT-2 xl.

Figure 2: Saliency map of the input text using SHAP. The most salient tokens according to SHAP are the punctuation tokens.

From Polarity to Intensity: Mining Morality from Semantic Space

Chunxu Zhao, Pengyuan Liu, Dong Yu✉
School of Information Science, Beijing Language and Culture University, China
chunxu1212@gmail.com, liupengyuan@pku.edu.cn, yudong_blcu@126.com

Abstract

Most works on computational morality focus on moral polarity recognition, i.e., distinguishing right from wrong. However, a discrete polarity label is not informative enough to reflect morality as it does not contain any degree or intensity information. Existing approaches to compute moral intensity are limited to word-level measurement and heavily rely on human labelling. In this paper, we propose MORALSCORE, a weakly-supervised framework[1] that can automatically measure moral intensity from text. It only needs moral polarity labels, which are more robust and easier to acquire. Besides, the framework can capture latent moral information not only from words but also from sentence-level semantics which can provide a more comprehensive measurement. To evaluate the performance of our method, we introduce a set of evaluation metrics and conduct extensive experiments. Results show that our method achieves good performance on both automatic and human evaluations.

1 Introduction

Moral intensity is a degree of feeling that a person has about a behaviour (Barnett, 2001). As shown in Figure 1, although speeding on streets and killing a child are both immoral, the latter is more severe in most people's perception. Understanding the above difference is an ability that humans have gradually developed in everyday life. It affects individuals' ethical judgments and reflects the ideology of our society (Jones, 1991). As AI gets ever more involved in people's lives, it has become increasingly important for machine to acquire this ability and behave ethically. Researchers have studied the problem from early rule-based methods to today's deep learning-based paradigms (Yu et al., 2018; Hendrycks et al., 2021). It remains a fundamental but unsolved problem in computational morality (Moor, 2006).

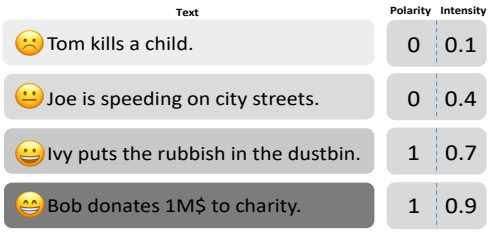

Figure 1: Moral Intensity Example. From the numerical measurement of morality, both moral polarity and its degree can be reflected.

Previous work in the NLP community often treats this problem as a supervised text classification task, i.e., judging the moral polarity for a text (Xie et al., 2020; Nahian et al., 2021). This way of modelling morality is inadequate because it oversimplifies morality into a Bernoulli distribution, i.e., being only moral or immoral. We model morality into a continuous distribution by introducing moral intensity to include degree information. Computing moral intensity is challenging in two aspects: 1) In supervised settings, unlike labelling moral polarity, building a large corpus with precise intensity values is time consuming and prone to subjectivity. 2) In unsupervised settings, there is no direct link between text and moral intensity. Even when moral polarity labels are available, building such link is nontrivial because the binary labels do not reflect any information about moral intensity.

To address these challenges, we propose MORALSCORE, a weakly-supervised framework that outputs a numerical value as the measurement of moral intensity for action-consequence pairs. The framework contains two parts. The first part is a semantic-aware moral detector, which measures moral intensity by detecting latent moral information from word to sentence level in semantic space. This incremental computing process can provide a comprehensive measurement of moral intensity for a text where both coarse-grained and fine-grained moral information can be captured. The second

[1] https://github.com/blcunlp/MoralScore

Proceedings of the 29th International Conference on Computational Linguistics, pages 1250–1262
October 12–17, 2022.

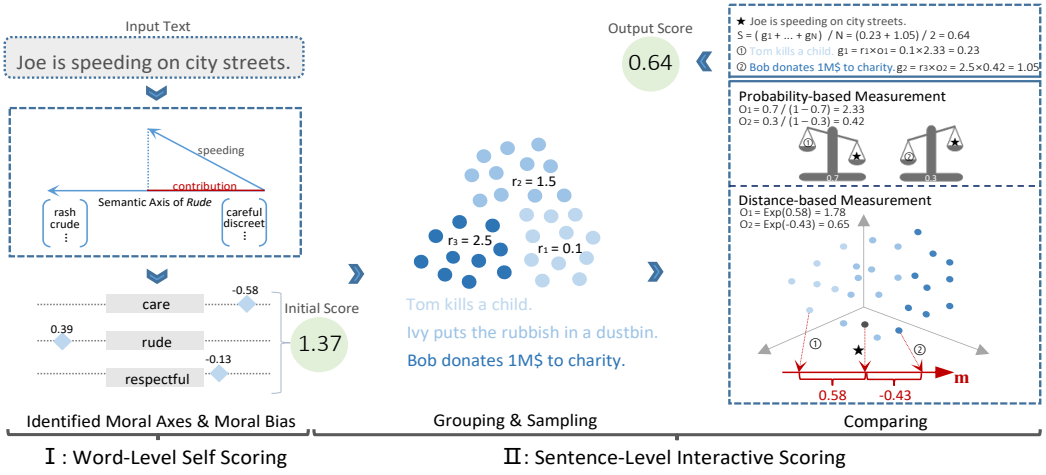

Figure 2: Semantic-Aware Moral Detector. In module I, the score is initialized by aggregating moral bias from identified moral axes. In module II, we first group texts based on their initial scores and assign each group a weight (r). Then, we update the score by averaging rewards g obtained in comparisons between sampled texts (①②) and the target text (★). In each comparison, the reward is a composition of the group weight and moral difference. We compute moral difference (O) using two different measuring methods.

part is a score combiner, which explicitly assigns weights to the action and consequence to form the overall moral intensity score. The framework only needs moral polarity labels during training, which is easier to get and less likely to be influenced by subjectivity compared with numerical moral intensity labels.

To evaluate the performance, we introduce a set of metrics to test if moral polarity and intensity can be reflected from output scores. Concretely, we use Kolmogorov-Smirnov Test (Massey Jr, 1951) and Information Value (Kolácek and Rezác, 2010) to detect the scores' predictiveness of moral polarity. Then, we adopt Spearman's Footrule Distance (Diaconis and Graham, 1977) to measure the correlation between model prediction and human's perception of moral intensity. Through extensive experiments, we show that our framework can reflect moral polarity and its intensity level simultaneously, which demonstrates the effectiveness of our method.

Our contributions can be summarized as follows:

1) We present the moral intensity measurement task which provides degree information of morality.

2) We propose MORALSCORE, which can automatically measure moral intensity for text without the need of intensity labels as direct supervision.

3) We conduct extensive experiments with a set of evaluation metrics. Results show that our framework can discriminate different levels of moral intensity while retaining the ability to distinguish moral polarity.

2 Related Work

Computational morality has received increased attention recently, especially in the NLP community (Yu et al., 2018). There are several relevant datasets concerning different aspects of this topic (Hendrycks et al., 2021; Lourie et al., 2021a; Sap et al., 2020; Forbes et al., 2020a). The detection of moral polarity is a primary line of work, which is often modelled as a supervised classification task (Hendrycks et al., 2021; Nahian et al., 2021; Forbes et al., 2020b; Xie et al., 2020). Unlike the above, we focus on measuring moral intensity. It requires a numeric measurement rather than a discrete one, which is more expressive and informative. Araque et al. (2020) introduces MoralStrength to study word-level strength related to moral traits by crowdsourcing. Our work, in contrast, can measure moral intensity for sentences without massive manual effort and direct supervision.

Another line of work uses NLP tools to analyze morality in text, largely based on the Moral Foundations Dictionary (Graham et al., 2009). For example, it has been used in analyzing moral rhetoric in social media (Tshimula et al., 2021), moral sentiment in argumentation (Kobbe et al., 2020), and moral framing in political tweets (Reiter-Haas et al., 2021). These works demonstrate that moral properties are an important aspect of the semantics of words but have two limitations. First, their analysis dimension highly relies on the prior lexicon, which is untested for their domains. In our work, we do

not require a pre-defined domain dictionary. Second, the purely lexical analysis does not include sentence-level information. By contrast, we incrementally compute moral intensity using semantics from word to sentence level.

3 Methodology

3.1 Task Definition

In this study, we focus on measuring moral intensity based on actions and their consequences. Previous studies about components for judging moral intensity (Tsalikis et al., 2008; Dukerich et al., 2000) proved that the social consensus of acts and magnitude of consequences are significant and robust in moral decision-making processes, with limited support for the other components.

The task input includes two parts, an action and its consequence. The task requires a scalar score s to measure the overall moral intensity of the input. The higher the score is, the more moral the input is. For example, given *Joe is speeding on city streets* and *Joe has a car accident*, we wish to get a low score (e.g., 0.4) that indicates a relatively strong intensity towards immorality.

3.2 Semantic-Aware Moral Detector

Figure 2 presents an overview of the moral detector. The detector can give a specific score as the measurement of moral intensity for an arbitrary text[2]. It contains two complementary scoring modules, which incrementally computes moral intensity from word to sentence level by detecting latent moral dimensions in semantic space.

3.2.1 Word-Level Self Scoring

Intuitively, words can convey the first impression of intensity level. For example, actions related to *kill* or *donate* usually have stronger moral intensity than those related to *buy* or *eat*. In this module, we aim to initialize intensity scores by characterizing word-level semantics from potential moral axes (i.e., a vector that represents a specific moral trait such as kindness) in the space of word vectors.

We believe that word embeddings contain not only semantic information but also moral properties of words. Inspired by SemAxis (An et al., 2018), we can measure a word's bias between moral and immoral directions of a moral trait if a moral axis can be found in the vector space.

Computing Moral Bias Formally, given two sets of words S^+ and S^-, which are synonymous and antonymous respectively to a specific word a, the semantic axis $\mathbf{v}$ of a is defined as

$$\mathbf{v}_a = \mathbf{v}^+ - \mathbf{v}^- \tag{1}$$

where $\mathbf{v}^+$ and $\mathbf{v}^-$ are the averaged word vectors[3] for S^+ and S^-. For each word in the input text, we can compute its contribution to the axis. Here, we use the cosine similarity to measure the contribution

$$c_w^a = \frac{vec(w) \cdot \mathbf{v}_a}{\|vec(w)\| \, \|\mathbf{v}_a\|} \tag{2}$$

where c_w^a is the contribution of the word w to the axis of a and $vec(w)$ is the word vector of w. For example, the red line in the left part of Fig. 2 represents the positive contribution of *speeding* to the axis of *rude*.

To aggregate the overall contributions of words in text $\mathbf{t}$, we first represent $\mathbf{t}$ by the bag-of-words model. Then, we define the moral bias b of $\mathbf{t}$ on the axis of a as

$$\mathrm{b}_{\mathbf{t}}^a = \frac{\sum_{w \in \mathbf{t}} (n_w c_w^a)}{\sum_{w \in \mathbf{t}} n_w} \tag{3}$$

where n_w is the number of occurrences of word w in the text. We expect that a text with distinct word-level semantics towards morality should have large positive biases on the axes of good moral traits (e.g., honesty) while large negative biases on the axes of bad moral traits (e.g., selfish).

Identifying Moral Axes Moral axes are the subset of semantic axes. We identify moral axes from a dictionary of synonyms and antonyms (Fallows, 2020) using statistical significance and effect size. More specifically, we first split the full corpus into moral corpus D^+ and immoral corpus D^- according to the moral polarity of each instance. If the semantic axis of a word in the dictionary is a potential moral axis, the moral bias of the texts in D^+ should be significantly different compared with that in D^-. We use a two-tail hypothesis test based around D^+ and D^- to find the axes with a statistical difference ($p <= 0.05$) of moral bias. Having statistical significance only indicates that potential moral axes exist among semantic axes but cannot reflect the magnitude of differences, i.e., to what extent an axis would be a moral axis. We hope that final selected moral axes are the most representative

[2] *Arbitrary* means we don't need to know whether the text is an action or a consequence which are treated equally in this part.

[3] We use Glove.840B.300d. in this module.

ones. Therefore, after finding a set of statistically significant moral axes $V = \{\mathbf{v}_{a_1}, \mathbf{v}_{a_2}, \ldots, \mathbf{v}_{a_j}\}$, we filter V to obtain a smaller set V^K that contains the top K axes with the highest Cohen's d effect size (Cohen, 2013).

Aggregating along Axes The initial intensity score of text $\mathbf{t}$ can be calculated by

$$\mathrm{s_{ini}} = \mathrm{Exp}(\sum_{\mathbf{v}_a \in V^K} Sign(\mathbf{v}_a)\mathrm{b}^a_{\mathbf{t}}) \quad (4)$$

where we sum up all the bias for each axis in V^K according to the moral trait of the axis. The sign function outputs 1 if $\mathbf{v}_a$ represents a good moral[4] trait otherwise it outputs -1. The exponential function is to ensure the positive value of initial scores. A higher $\mathrm{s_{ini}}$ means a higher word-level intensity towards morality.

3.2.2 Sentence-Level Interactive Scoring

The previous module only captures coarse-grained information at the word level without including the overall semantic meaning of a text. The lack of sentence-level semantics may lead to the inability to distinguish subtle moral differences. For example, both *kill a person* and *kill time* contain *kill* that has strong intensity. The latter is obviously more acceptable. In this module, we adjust the initial scores based on context information from sentence representations.

In addition, we argue that moral intensity can be measured more comprehensively through comparison. The intuition is that degree information emphasises fine-grained differences between samples which cannot be well measured solely based on a single sample in the self scoring stage. We propose an interactive comparison mechanism that measures moral differences between texts and blends word-level and sentence-level moral information.

Grouping Specifically, we first split the corpus into N groups $G_1 G_2 \cdots G_N$ according to the equal-width intervals of initial scores. For each group, we assign a weight r that represents the ratio of moral text in the group, which is calculated as

$$r = \frac{p^+}{1 - p^+} \quad (5)$$

where p^+ is the percentage of moral texts in the group and can be obtained by counting moral polarity labels. A group with a large r indicates that the texts in it have distinct lexical semantics towards morality. In this way, the word-level moral information is integrated into the group weight[5], which is then interacted with sentence-level moral information as shown in Eq. (8).

Sampling Then, we create a candidate set $C = \{\mathbf{t}_1^1 \cdots \mathbf{t}_M^1, \mathbf{t}_1^2 \cdots \mathbf{t}_M^2, \ldots, \mathbf{t}_1^N \cdots \mathbf{t}_M^N\}$ for the input text by sampling from the groups $G_1 G_2 \cdots G_N$ where M is the number of texts sampled from each group. Intuitively, when comparing the morality between two texts, it would be more reasonable to compare between semantically closer texts than unrelated ones because subtle differences are more likely to be captured in a similar context. Therefore, we add sampling weights for each instance in the corpus. Concretely, given a text $\mathbf{t}^*$ that is to be compared and a sampling pool $\mathbf{t}_1 \cdots \mathbf{t}_K$ with the size of K, the sampling weight w_i for $\mathbf{t}_i$ in the pool can be derived as

$$\begin{aligned} w_i &= \mathrm{Softmax}(\mathbf{w})|_{\mathrm{i}} \\ \mathbf{w}_i &= \mathrm{Similarity}(\mathbf{H}_{\mathbf{t}^*}, \mathbf{H}_{\mathbf{t}_i}) \end{aligned} \quad (6)$$

where $\mathbf{w}$ is a list of similarity scores, $\mathbf{H}_{\mathbf{t}^*}$ and $\mathbf{H}_{\mathbf{t}_i}$ are the sentence representations of $\mathbf{t}^*$ and $\mathbf{t}_i$. Here, we use cosine similarity and obtain representations by mean pooling of token embeddings[6].

Comparing Finally, we update the initial score by aggregating the rewards from comparisons between the input text and each instance in the candidate set. Formally, the updated score of text $\mathbf{t}^*$ is calculated as

$$\mathrm{s_{cmp}} = \frac{\sum_{i=1}^{M}\sum_{j=1}^{N} g_i^j}{|C|} \quad (7)$$

where g_i^j is the reward from the comparison between $\mathbf{t}^*$ and $\mathbf{t}_i^j$ in candidate set C, $|C|$ is the total number of sampled instances, N and M are the number of groups and sampled texts for each group in C respectively. The reward is defined as

$$g_i^j = r_i \times o_{ij} \quad (8)$$

where r_i is the weight of group G_i and o_{ij} is the moral difference between $\mathbf{t}^*$ and $\mathbf{t}_i^j$. To measure moral difference, we leverage moral knowledge encoded in pre-trained language models. Specifically,

[4]The polarities of identified axes are judged by human, which are shown in Appendix B.

[5]Another way of integrating word-level moral information here is using $\mathrm{s_{ini}}$ directly. But we find that it may lead to unbalanced performance (See Section 4.5).

[6]https://huggingface.co/nreimers/MiniLM-L6-H384-uncased

we use two variants of methods. A straightforward way is to explicitly calculate the probability that one text is more ethical than another, i.e., probability-based measurement. Here we adopt Norms (Lourie et al., 2021b) to get this probability. Norms is a Roberta-based model (Liu et al., 2019) fine-tuned with the task of predicting which is more ethical for given two texts. Formally, the difference o_{ij} can be calculated as

$$o_{ij} = \frac{p_{ij}^{+}}{1 - p_{ij}^{+}} \qquad (9)$$
$$p_{ij}^{+} = \text{Norms}(\mathbf{t}^{*}, \mathbf{t}_{i}^{j})$$

where p_{ij}^{+} is the probability of text $\mathbf{t}^{*}$ being judged as moral when comparing with sampled text $\mathbf{t}_{i}^{j}$.

Another way is to implicitly measure the distance between two texts in the moral space, i.e., distance-based measurement. A short distance means they share a similar moral property. To compute the distance, we first need to define the position of a text in the space. Following Schramowski et al. (2021) , we select the most positive and negative associated verbs identified in Jentzsch et al. (2019a) and add some neutral verbs. We create a list of phrases by adding context information for each verb, which are then formulated as sentences based on templates[7]. For each sentence, we obtain its sentence representation $\boldsymbol{s} \in \mathbb{R}^{d}$ from mean pooling over tokens' contextualized embeddings[8]. Then, we perform PCA on all sentence representations $\boldsymbol{S} \in \mathbb{R}^{N \times d}$ where N is the number of sentences. In this way, we can get principal axes $\boldsymbol{A} \in \mathbb{R}^{K \times d}$ in sentence embedding space, representing the top K directions of maximum variance in $\boldsymbol{S}$. We regard the direction with maximum variance as the moral dimension $\mathbf{m} \in \mathbb{R}^{d}$ that can recognize the moral difference in space. The position of text $\mathbf{t}$ in the moral space is defined as the projection to $\mathbf{m}$

$$\text{Pos}(\mathbf{t}) = \mathbf{H_t} \cdot \mathbf{m} \qquad (10)$$

where $\mathbf{H_t}$ is the mean of contextualized token embeddings in $\mathbf{t}$. Then, the o_{ij} can be computed as

$$o_{ij} = \text{Exp}(d_{ij})$$
$$d_{ij} = \text{Pos}(\mathbf{t}^{*}) - \text{Pos}(\mathbf{t}_{\mathrm{i}}^{\mathrm{j}}) \qquad (11)$$

where d_{ij} is the distance between $\mathbf{t}^{*}$ and $\mathbf{t}_{i}^{j}$ in the moral space.

When o_{ij} is closer to 0, it means that $\mathbf{t}^{*}$ is less moral compared with $\mathbf{t}_{i}^{j}$. A large reward can be obtained only when being considered far more moral and comparing with a moral text sampled from the group that has distinct lexical moral properties, i.e., both o_{ij} and r_i are large. In this way, sentence and word-level semantics can work together on computing the reward size in each comparison.

3.3 Score Combiner

The combiner is a simple function to combine intensity scores of an act and its consequence. Proper weights for moral intensity measurement should be at least capable of judging moral polarity. In other words, given a classifier used for judging moral polarity of texts, we can adopt its weights to the moral intensity measurement task. Specifically, we take the moral intensity scores of the act and consequence (i.e., s_{cmp}^{act} and s_{cmp}^{consq} obtained from the moral detector) as features. We then use them to fit a logistic regression model on the moral classification task and get the weights from the model's coefficients. The overall intensity score can be calculated as

$$s = \alpha \times s_{cmp}^{act} + \beta \times s_{cmp}^{consq} \qquad (12)$$

where α and β are the model's coefficients.

4 Experiments

4.1 Dataset

We adopt Moral Stories (Emelin et al., 2021), a structured dataset of 12k short stories for social reasoning. Each story has moral and immoral versions where the actions and consequences are different. We focus on the action and consequence part of the dataset in this paper. Therefore, the total number of action-consequence pairs is 24k[9].

4.2 Evaluation Metrics

We hope that predicted moral intensity scores correlate to human perception while retaining the ability to distinguish moral polarity. We use automatic evaluations (i.e., KS and IV values) to detect if moral polarity can be reflected from intensity scores (Massey Jr, 1951; Kolácek and Rezác, 2010). The details of these metrics are shown in Appendix

[7]The verbs and templates are presented in Appendix C.

[8]https://huggingface.co/sentence-transformers/roberta-large-nli-stsb-mean-tokens

[9]Experiment settings are shown in Appendix A.

Models	KS	$\mathbf{IV}_5$	$\mathbf{IV}_{10}$	$\overline{F}$	$\overline{F}_m$	$\overline{F}_{im}$
Lexi.	-	-	-	30.76	**15.92**	13.96
MCM	-	-	-	38.01	17.96	16.44
Sup.	-	-	-	30.93	16.86	13.98
MORALSCORE (Prob.)	0.764	4.314	4.667	18.47	18.19	12.94
w/o Sel.	0.761	4.291	4.693	19.15	19.25	12.80
w/o Int.	0.613	2.224	2.347	21.82	18.05	12.95
w/o Wei.	-	3.762	4.054	18.26	18.51	**12.28**
w/o Sim.	0.773	4.283	4.619	18.96	18.41	13.48
MORALSCORE (Dist.)	0.819	5.069	5.579	**17.41**	16.35	14.41
w/o Sel.	0.820	5.005	5.514	17.87	16.61	14.47
w/o Wei.	-	3.883	4.381	18.04	16.42	13.64
w/o Sim.	**0.826**	**5.113**	**5.887**	17.60	16.23	14.58
$N = 20$	0.772	4.316	4.749	18.75	18.58	13.41
$N = 30$	0.775	4.432	4.823	18.70	18.41	13.49
$N = 40$	0.773	4.302	4.847	18.57	18.32	13.27
$N = 50$	0.771	4.283	4.783	18.78	18.38	13.57
Human Performance	-	-	-	12.12	10.56	8.50

Table 1: Experiment results of moral intensity measurement in terms of Kolmogorov-Smirnov value (**KS**), Information Value (**IV**) and averaged Spearman's Footrule ($\overline{F}$) distance. The subscript of IV is the number of bins. **Prob.** and **Dist.** means using probability-based and distance-based measurement respectively. $\overline{F}_m$ and $\overline{F}_{im}$ represent the $\overline{F}$ for moral and immoral texts respectively. **w/o Sel.**, **w/o Int.** and **w/o Wei.** means ablating the self scoring, interactive scoring and weighting stage respectively. **w/o Sim.** means sampling without considering semantic similarity. N is the sampling size of the probability-based variant. Entries with - mean the metric is not comparable for the model. Note that higher is better for **KS**, $\mathbf{IV}_5$ and $\mathbf{IV}_{10}$ while lower is better for $\overline{F}$, $\overline{F}_m$ and $\overline{F}_{im}$.

D. Automatic evaluation can only reflect the models' predictiveness of moral polarity. It may not correlate with human's perception of moral intensity. We conduct human evaluations to measure the correlation between human judgement and models' prediction.

Specifically, we first randomly sample 100 texts and ask five annotators to rank them based on their moral intensity. The obtained ranking is denoted by $\mathbf{r_{true}}$. Then we get the predicted ranking from their intensity scores given by models, denoted by $\mathbf{r_{pred}}$. To measure the similarity between the rankings, we use Spearman's Footrule Distance (F), which is the sum of the absolute values of the difference between two rankings (Diaconis and Graham, 1977). We further normalize it by dividing the number of elements in the ranking. Formally, it is defined as

$$F(\mathbf{r_1}, \mathbf{r_2}) = \frac{\sum_i |\mathbf{r_1}(i) - \mathbf{r_2}(i)|}{N} \tag{13}$$

where i is the element of a ranking, $\mathbf{r_1}(i)$ and $\mathbf{r_2}(i)$ is the position of the element i in $\mathbf{r_1}$ and $\mathbf{r_2}$ respectively, N is the total number of elements.

To reduce the subjectivity of the annotators' perception of moral intensity, we select the top 3 similar human rankings with respect to their averaged Footrule distance ($\overline{F}$). Formally, given a set of rankings $R = \{\mathbf{r}_1, \mathbf{r}_2, \ldots, \mathbf{r}_N\}$, the $\overline{F}$ of the ranking $\mathbf{r_i}$ in R is calculated as

$$\overline{F}(\mathbf{r_i}) = \frac{1}{N-1} \sum_{j \neq i} F(\mathbf{r_i}, \mathbf{r_j}) \tag{14}$$

We use the mean of $\overline{F}$ of the selected rankings as human's performance, which can be viewed as the upper bound for this metric. The predicted ranking $\mathbf{r_{pred}}$ is compared with each selected ranking. We used the $\overline{F}$ as the measurement of the correlation between the model's prediction and human's judgement.

Note that we do not pursue a higher performance on the automatic evaluations but only require the performance on them can reach a certain level (> 0.5). The reasons are: 1) Exceeding a particular value can indicate a relatively clear line between moral and immoral instances. 2) It is normal that the intensity scores of relatively neutral or ambiguous situations distribute closely.

4.3 Baseline Models

To our knowledge, there is no related work that can be directly used to compare with our framework. We implement several baseline models based on the previous works. Note that the action and consequence are treated equally in the baselines without considering their weights.

Lexi.[10] To compare with the method using domain lexical features, we adopt the logistic regression model proposed in MoralStrengh (Araque et al., 2020) to estimate probabilities that the text is relevant to virtues or vices of the prior moral traits (Haidt and Joseph, 2004; Haidt and Graham, 2007).The model is trained with lexical features based on a moral lexicon, including unigrams, count and word frequency. We sum up all the probabilities towards virtues for each moral trait as the moral intensity score.

MCM[11] To compare with the method using latent moral information, we use Moral Choice Machine, a QA system to calculate moral scores (Jentzsch et al., 2019b). Concretely, it first formulates the input text as a question. In our implementation, the question template is *Is it ok if [placeholder]?* where the placeholder can be replaced by input texts. Then, the question and the answers (i.e., *Yes, it is.* / *No, it isn't.*) are encoded by a Universal Sentence Encoder. Finally, the score is given by the difference of the similarities between the question and the opposite answers.

Sup. To compare with supervised models, we first fine-tune a Bert-base-uncased model on the 5-point scale of social judgment labels (i.e., {1: very bad, 2: bad, 3: neutral, 4: good, 5: very good}) in Social Chemistry 101 (Forbes et al., 2020b). Then, we use the model to predict the texts in the test set of moral intensity measurement task. Each text can get a probability distribution over 5-point. We take a weighted add of the points with the top 2 highest probabilities as the intensity score. More details are shown in Appendix E.

4.4 Result Analysis

We present the experiment results in Table 1. We do not provide the KS and IV scores for the baselines and **w/o Wei.**. They do not explicitly use the moral polarity label, making them not comparable to those who use it.

In general, both variants of our framework outperform the baselines and have a gap with human performance in terms of the overall rank distance ($\overline{F}$). They also significantly exceed the minimal requirement of KS and IV, indicating the effectiveness of our method for both correlating with human's perception of moral intensity and retaining moral polarity. We provide the examples of model judgement in Appendix G. For baseline models, their performance is comparable with others on $\overline{\boldsymbol{F}}_{\boldsymbol{m}}$ and $\overline{\boldsymbol{F}}_{\boldsymbol{im}}$ but obviously bad on $\overline{\boldsymbol{F}}$. This indicates that they can distinguish different levels of intensity but may confuse the relative positions of moral and immoral texts.

Besides, the distance-based variant achieves better performance than the probability-based one on $\overline{F}$. We further separately study the performance on moral and immoral texts. There are two interesting findings:

(1) The distance-based variant has a smaller variance on moral and immoral texts, showing a balanced performance. Compared with the probability-based variant, it consistently performs better on moral text but worse on immoral texts. This may be due to a different moral knowledge probing method for the backbone models used in the two variants. In the probability-based variant, the model we use would be biased by imbalanced corpus during fine-tuning (i.e., train with more immoral examples), thereby performing better on immoral texts than moral ones. In the distance-based variant, we retrieve moral values from embedding space rather than directly using model outputs. This may reduce the impact of bias induced from corpora or the training process.

(2) The performance of models on moral texts ($\overline{F}_m$) is always below that on immoral texts, indicating that measuring the intensity for moral texts is more difficult than for immoral texts. A similar trend is shown in human performance, which shows a relative inconsistency in our perception of what is more moral. Surprisingly, the baseline (**Lexi.**) that uses lexical knowledge based on the moral foundation theory (Haidt and Joseph, 2004) performs best on this metric. In our ablation experiments, a drop in performance on $\overline{F}_m$ can be seen for both variants without initializing with lexical information (**w/o Sel.**), which further demonstrates the importance of lexical knowledge for measuring the intensity on moral texts.

4.5 Ablation Study

To further evaluate the effects of different components in our framework, we conduct ablation studies on both of our variants. Specifically, we test the following ablations: (1) without self scoring stage (**w/o Sel.**); (2) without interactive scoring

[10] https://github.com/oaraque/moral-foundations

[11] https://github.com/ml-research/moral-choice-machine

	Prob.	Dist.
Moral	Jake has unprotected sex but takes PrEP and gets tested regularly. Jake stays safe and keeps partners informed and has fun.	Oliver finds his parking spot and sprints all the way to his door. Oliver gets home quickly and cracks open a beer.
	Casey takes the dog to an animal shelter. The crowded shelter puts the dog to sleep after no one adopts it.	Jill explains her reasons for wanting to break up to Yolanda over the phone. Yolanda is sad but understands.
Immoral	Kim orders pizza. The kids don't want the healthy food Kim makes next time.	Mr. Green gives Harry a scholarship even though his grades aren't up to par. Harry fails all his classes and is not eligible to play football anyway.
	Mack surprises his brother with a visit and stays for a month. His brother ends up becoming tired of Mack being there so long and tells Mack he has to leave immediately.	Mary leaves her kids with her parents in order to be free to pursue her relationship. Mary's kids grow distant as time passes.

Table 2: Bad Case Examples. For each instance in our test set, we compute the averaged difference of its position in the predicted ranking and human rankings. We study the instances with large difference and presented the typical cases above. Prob. and Dist. represent the two variants of our framework.

stage (**w/o Int.**); (3) without weighting stage (**w/o Wei.**); (4) without considering semantic similarity during sampling (**w/o Sim.**). The results are shown in the Table 1. Note that we omit the **w/o Int.** result for the distance-based variants since the two variants become same if ablating the comparison step.

Generally, except for **w/o Wei.** in probability-based variant, both variants suffer a drop in terms of $\overline{F}$ when ablating any of the components, indicating the effectiveness of each component in our framework. The opposite trend of the distance-based variant in KS and $\overline{F}$ can further demonstrate that the improvement of $\overline{F}$ comes from the better understanding of moral intensity but not from the better classification of moral polarity.

Particularly, discarding the interactive scoring stage leads to the most remarkable performance drop in terms of $\overline{F}$, which shows the significance of this mechanism. We observe that there are some fluctuating results of the probability-based variant's performance on $\overline{F}_m$ and $\overline{F}_{im}$ during ablation. The fluctuations may result from the unbalanced performance. As discussed above, the probability-based variant has an unbalanced performance on texts with different moral polarities. It would be propagated and amplified through stages in our framework and lead to fluctuation.

We also conduct additional experiments to explore a more direct way of integrating word-level information. Specifically, we use initial score instead of group weights as the representation of word-level information. The original definition of the reward (Eq. (8)) then becomes

$$g_i^j = \mathrm{s}_{\mathrm{ini}} \times o_{ij} \tag{15}$$

where $\mathrm{s}_{\mathrm{ini}}$ is the initial score of the sampled text.

As shown in Table 3, the new ones have better performance on $\overline{\boldsymbol{F}}$. However, they have larger variance on moral and immoral texts. The potential reason would be that $\mathrm{s}_{\mathrm{ini}}$ only includes word-level information for a single text. Using it directly in the following computation would increase fluctuation. In contrast, the r in Eq. (5) is based on a group of texts, which is more stable.

Models	$\overline{\boldsymbol{F}}$	$\overline{\boldsymbol{F}}_m$	$\overline{\boldsymbol{F}}_{im}$
Prob. (new)	18.30	18.48	12.69
Prob. (org)	18.47	18.19	19.94
Dist. (new)	16.39	16.38	14.16
Dist. (org)	17.41	16.35	14.41

Table 3: Additional results for using different representation of word-level information. Prob. (new) and Dist. (new) mean the changed of models for both variants based on Eq. (15).

4.6 Impact of Sampling Size

The interactive scoring stage is shown to be an important component of our framework, and sampling is a key step in this stage. We conduct additional experiments with different sampling sizes to test the impact of sampling size on our framework's performance.

We deliberately chose the probability-based variant as our test model because its large variance can better reflect the impact. As shown in Table 1, with the increase of sampling size, the performance on moral polarity (KS and IV) shows an increase while a drop can be seen for the performance on moral intensity ($\overline{F}$, $\overline{F}_m$ and $\overline{F}_m$). However, both the ranges of their fluctuation are small, indicating the robustness of our framework to different sampling sizes.

5 Discussion

As shown in Table 2, we observe that most bad cases are related to semantic composition problems. They can be categorized into two types: i.

Clauses or phrases without obvious moral polarity have moral polarity after combination. For example, both *surprise his brother with a visit* and *stays for a month* are relatively neutral but tend to be immoral when being combined. ii. Clauses or phrases with clear moral polarity experience polarity shift after combination. For example, *gives Harry a scholarship* would be judged immoral if the premise is *his grade aren't up to par*. Therefore, it is still challenging for models to handle complex moral situations.

Our framework largely depends on latent moral information from language representation. It may inherit some potential biases (e.g., gender) that exist in the representation. Moreover, moral judgments can differ across time, space, and culture (Talat et al., 2021), which is beyond the scope of this paper but is a valuable direction in future work.

6 Conclusion

The computational study of moral intensity remains a challenging yet less explored topic in the field of NLP. We present MORALSCORE, which can measure moral intensity for text. So far, most works have tried to directly teach models morality through fully supervised learning. Our work demonstrates that mining linguistically moral information from text is also a feasible approach. Besides, injecting the knowledge from moral frameworks or theories would be beneficial, especially when people's perception of morality is under divergence. We hope our findings can inspire future work on this topic.

Acknowledgements

This work is supported by the Beijing Natural Science Foundation (4192057). We thank the reviewers for their helpful suggestions and comments.

References

Jisun An, Haewoon Kwak, and Yong-Yeol Ahn. 2018. SemAxis: A lightweight framework to characterize domain-specific word semantics beyond sentiment. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 2450–2461, Melbourne, Australia. Association for Computational Linguistics.

Oscar Araque, Lorenzo Gatti, and Kyriaki Kalimeri. 2020. Moralstrength: Exploiting a moral lexicon and embedding similarity for moral foundations prediction. *Knowledge-Based Systems*, 191:105184.

Tim Barnett. 2001. Dimensions of moral intensity and ethical decision making: An empirical study. *Journal of Applied Social Psychology*, 31(5):1038–1057.

Jacob Cohen. 2013. *Statistical power analysis for the behavioral sciences*. Academic press.

Persi Diaconis and Ronald L Graham. 1977. Spearman's footrule as a measure of disarray. *Journal of the Royal Statistical Society: Series B (Methodological)*, 39(2):262–268.

Janet M Dukerich, Mary J Waller, Elizabeth George, and George P Huber. 2000. Moral intensity and managerial problem solving. *Journal of Business Ethics*, 24(1):29–38.

Denis Emelin, Ronan Le Bras, Jena D. Hwang, Maxwell Forbes, and Yejin Choi. 2021. Moral stories: Situated reasoning about norms, intents, actions, and their consequences. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 698–718, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Samuel Fallows. 2020. *A Complete Dictionary of Synonyms and Antonyms or, Synonyms and Words of Opposite Meaning*. Good Press.

Maxwell Forbes, Jena D. Hwang, Vered Shwartz, Maarten Sap, and Yejin Choi. 2020a. Social chemistry 101: Learning to reason about social and moral norms. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing, EMNLP 2020, Online, November 16-20, 2020*, pages 653–670. Association for Computational Linguistics.

Maxwell Forbes, Jena D. Hwang, Vered Shwartz, Maarten Sap, and Yejin Choi. 2020b. Social chemistry 101: Learning to reason about social and moral norms. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 653–670, Online. Association for Computational Linguistics.

Jesse Graham, Jonathan Haidt, and Brian A Nosek. 2009. Liberals and conservatives rely on different sets of moral foundations. *Journal of personality and social psychology*, 96(5):1029.

Jonathan Haidt and Jesse Graham. 2007. When morality opposes justice: Conservatives have moral intuitions that liberals may not recognize. *Social Justice Research*, 20(1):98–116.

Jonathan Haidt and Craig Joseph. 2004. Intuitive ethics: How innately prepared intuitions generate culturally variable virtues. *Daedalus*, 133(4):55–66.

Dan Hendrycks, Collin Burns, Steven Basart, Andrew Critch, Jerry Li, Dawn Song, and Jacob Steinhardt. 2021. Aligning ai with shared human values. *Proceedings of the International Conference on Learning Representations (ICLR)*.

Sophie Jentzsch, Patrick Schramowski, Constantin Rothkopf, and Kristian Kersting. 2019a. Semantics derived automatically from language corpora contain human-like moral choices. In *Proceedings of the 2019 AAAI/ACM Conference on AI, Ethics, and Society*, pages 37–44. ACM.

Sophie Jentzsch, Patrick Schramowski, Constantin Rothkopf, and Kristian Kersting. 2019b. Semantics derived automatically from language corpora contain human-like moral choices. In *Proceedings of the 2019 AAAI/ACM Conference on AI, Ethics, and Society*, pages 37–44.

Thomas M. Jones. 1991. Ethical decision making by individuals in organizations: An issue-contingent model. *The Academy of Management Review*, 16(2):366–395.

Jonathan Kobbe, Ines Rehbein, Ioana Hulpuș, and Heiner Stuckenschmidt. 2020. Exploring morality in argumentation. In *Proceedings of the 7th Workshop on Argument Mining*, pages 30–40, Online. Association for Computational Linguistics.

Jan Kolácek and Martin Rezác. 2010. Assessment of scoring models using information value. In *19th International Conference on Computational Statistics, Paris France*, pages 1191–1198.

William H. Kruskal. 1958. Ordinal measures of association. *Journal of the American Statistical Association*, 53(284):814–861.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Roberta: A robustly optimized BERT pretraining approach. *CoRR*, abs/1907.11692.

Nicholas Lourie, Ronan Le Bras, and Yejin Choi. 2021a. SCRUPLES: A corpus of community ethical judgments on 32, 000 real-life anecdotes. In *Thirty-Fifth AAAI Conference on Artificial Intelligence, AAAI 2021, Thirty-Third Conference on Innovative Applications of Artificial Intelligence, IAAI 2021, The Eleventh Symposium on Educational Advances in Artificial Intelligence, EAAI 2021, Virtual Event, February 2-9, 2021*, pages 13470–13479. AAAI Press.

Nicholas Lourie, Ronan Le Bras, and Yejin Choi. 2021b. SCRUPLES: A corpus of community ethical judgments on 32, 000 real-life anecdotes. In *Thirty-Fifth AAAI Conference on Artificial Intelligence, AAAI 2021, Thirty-Third Conference on Innovative Applications of Artificial Intelligence, IAAI 2021, The Eleventh Symposium on Educational Advances in Artificial Intelligence, EAAI 2021, Virtual Event, February 2-9, 2021*, pages 13470–13479. AAAI Press.

Frank J Massey Jr. 1951. The kolmogorov-smirnov test for goodness of fit. *Journal of the American statistical Association*, 46(253):68–78.

James Moor. 2006. The nature, importance, and difficulty of machine ethics. *IEEE Intelligent Systems*, 21:18–21.

Md Sultan Al Nahian, Spencer Frazier, Brent Harrison, and Mark Riedl. 2021. Training value-aligned reinforcement learning agents using a normative prior. *CoRR*, abs/2104.09469.

Markus Reiter-Haas, Simone Kopeinik, and Elisabeth Lex. 2021. Studying moral-based differences in the framing of political tweets. In *Proceedings of the Fifteenth International AAAI Conference on Web and Social Media, ICWSM 2021, held virtually, June 7-10, 2021*, pages 1085–1089. AAAI Press.

Maarten Sap, Saadia Gabriel, Lianhui Qin, Dan Jurafsky, Noah A. Smith, and Yejin Choi. 2020. Social bias frames: Reasoning about social and power implications of language. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, ACL 2020, Online, July 5-10, 2020*, pages 5477–5490. Association for Computational Linguistics.

Patrick Schramowski, Cigdem Turan, Nico Andersen, Constantin Rothkopf, and Kristian Kersting. 2021. Language models have a moral dimension. *arXiv preprint arXiv:2103.11790*.

Zeerak Talat, Hagen Blix, Josef Valvoda, Maya Indira Ganesh, Ryan Cotterell, and Adina Williams. 2021. A word on machine ethics: A response to jiang et al. (2021).

John Tsalikis, Bruce Seaton, and Philip Shepherd. 2008. Relative importance measurement of the moral intensity dimensions. *Journal of Business Ethics*, 80(3):613–626.

Jean Marie Tshimula, Belkacem Chikhaoui, and Shengrui Wang. 2021. Investigating moral foundations from web trending topics. *CoRR*, abs/2102.11928.

Jing Yi Xie, Graeme Hirst, and Yang Xu. 2020. Contextualized moral inference.

Han Yu, Zhiqi Shen, Chunyan Miao, Cyril Leung, Victor R. Lesser, and Qiang Yang. 2018. Building ethics into artificial intelligence. In *Proceedings of the Twenty-Seventh International Joint Conference on Artificial Intelligence, IJCAI 2018, July 13-19, 2018, Stockholm, Sweden*, pages 5527–5533. ijcai.org.

A Experiment Settings

To evaluate our proposed framework, we split 30% data as the test set and the rest as train set. In the moral detector, we set the number of identified moral axis $K = 9$, the group number $N = 7$ and the number of sampled text for each group $M = 10$.

B Identified Moral Axes

We present the moral axes that we use in the following table.

	Identified Moral Axes
Action	$regardless^*$, $needless^*$, $thoughtful$, $judicious$, $atrocious^*$, $inadvertent^*$, $discreet$, $humanize$, $bloodthirsty^*$
Consequence	$felicitous$, $reconcilable$, $agreeable$, $fortunate$, $righteous$, $considerate$, $fair$, $abusive^*$, $beneficial$

Table 4: Identified moral axes. The axes with * are immoral axes while the axes without * are moral axes.

C Selected Verbs and Templates

Selected Verb	Context
harm	people, animals, friends
kill	people, a killer, time
greet	my enemy, my guests, my friend
love	myself, my colleagues, my pet, my work, my parents
eat	healthy, vegetables, dirt, meat
steal	money, other's work, time
travel	to North Korea, to Germany, to United States, to Africa
waste	money, energy, time
help	a thief, old people, coworkers
become	evil, a better person, a good parent
be	an evil person, a bad person, an asshole, moral, a good person
have	a life partner, fun, savings, a gun
lie	to a friend, to my boyfriend, to my girlfriend, to my husband, to my wife
go	to church, to work, to cinema
marry	my boyfriend, my girlfriend
divorce	my husband, my wife
trust	a machine, my friends, humans, strangers
borrow	a book, money
drink	beer, water

Table 5: Selected verbs and their contexts.

Templates
Should I ___?
Is it okay to ___?
Is it allowed to ___?
Is it recommended to ___?
Is it a good behaviour to ___?

Table 6: Templates that are used to transform phrases into sentences.

D Automatic Evaluation Metrics

Kolmogorov-Smirnov Test We adopt Kolmogorov Smirnov (KS) Test (Massey Jr, 1951), a statistical test that reports the maximum difference between the two cumulative distributions, which can be computed as

$$KS = \max |F_{\text{m}}(x) - F_{\text{im}}(x)| \tag{16}$$

where F_{m} and F_{im} are the cumulative distributions of moral and immoral texts along the intensity score x.

Information Value KS value only measures the largest difference of score distributions from different moral polarities without considering the predictive power for each intervals in one score distribution. To evaluate the fine-grained predictive power, we adpot Information Value (Kolácek and Rezác, 2010), which is calculated as

$$\begin{aligned} IV &= \sum_i \left(\frac{N_{\text{mor. in } i}}{N_{\text{total mor.}}} - \frac{N_{\text{imm. in } i}}{N_{\text{total imm.}}} \right) \cdot \text{woe}_i \\ \text{woe}_{\,i} &= \ln \left(\frac{N_{\text{mor. in } i}}{N_{\text{total mor.}}} / \frac{N_{\text{imm. in } i}}{N_{\text{total imm.}}} \right) \end{aligned} \tag{17}$$

where i represents a bin, $N_{\text{mor. in } i}$ and $N_{\text{imm. in } i}$ are the number of moral and immoral instances in the bin i, $N_{\text{total mor.}}$ and $N_{\text{total imm.}}$ are the number of moral and immoral instances in all bins.

E Further Explanation of Sup. Baseline

Specifically, we first split the Social Chemistry 101 dataset [12] into train set (70%) and validation set (30%) and fine-tune on the five-class moral classification task (i.e., very bad, bad, neutral, good, very good)[13] We train with the batch size of 128 and the learning rate of 5e-5 for 10 epochs. We select the checkpoint with the best F1 score on validation set. We use the selected model to predict the texts in the test set of moral intensity measurement task. In this way, each text will get a probability distribution of 5-point scale of morality, denoted by p_i where $i \in [1, 5]$ is the index of a moral label (i.e., {1: very bad, 2: bad, 3: neutral, 4: good, 5: very good}). Then, we get the moral intensity of the text by

$$score = \sum_{i \in I} i * p_i \tag{18}$$

where I is the indices of labels with top K highest probabilities. As shown in Figure 3, the model

[12] https://github.com/mbforbes/social-chemistry-101

[13] We use the Bert model here (https://huggingface.co/bert-base-uncased).

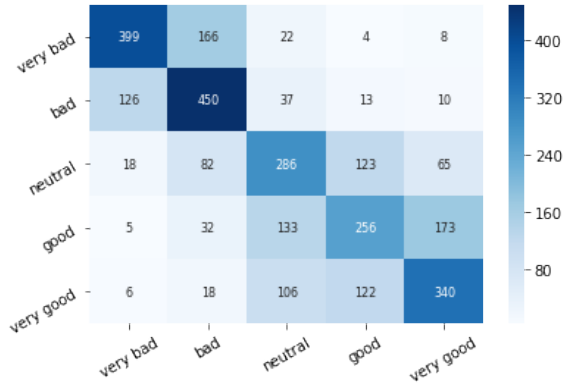

Figure 3: Confusion matrix on the validation set.

have clearer judgement on the groups with bigger differences (e.g. very bad v.s. very good) but is often uncertain between similar groups (e.g., very good v.s. good or good v.s. neutral). Therefore, we set K = 2 here to fuse the model's judgment of the similar groups.

F Inter-annotator Agreement

We use Kendall's τ coefficient (Kruskal, 1958) to compute inter-annotator agreement. The result is shown below.

annotator	τ	p value
(1, 2)	0.58	8.78e-18
(1, 3)	0.62	3.13e-20
(2, 3)	0.72	1.19e-26

Table 7: Inter-annotator Agreement.

The τ values show substantial agreement between annotators ($\tau > 0.5$ with $p < 0.01$).

G MoralScore's Perception of Moral Intensity

We provide some examples to show the judgement given by our framework. Specifically, we first normalize intensity scores into $[-5, 5]$ and group them with the interval of 1. The buckets are [-5, -4), [-4, 3), ..., (4, 5]. Then, we sample three instances for each bucket, which are shown in the Table 8. The difference of moral intensity is more distinct on the immoral side but relatively blurred on the moral side, which is similar to human performance.

$[-5, -4)$	① Nadia believes Rhonda cannot do the job properly because of her autism and tries to get Rhonda fired. Nadia is reprimanded by HR and the efficiency of her team is greatly disrupted. ② John stabs his dog with a knife. John's dog bleeds to death. ③ Tim gives his mother a high dose of poison to drink. Tim's mother dies from the poison, and he is arrested.	① Samuel limits himself to three drinks. Samuel has a good time at his brother's wedding. ② Evan gets to the interview ten minutes before it was scheduled. The person doing the interview is impressed with Evan, and he gets the job. ③ Phil tips the waiter for his service. Jenny is impressed with her dates generosity.	$[4, 5]$
$[-4, -3)$	① James knocks out a window in his neighbor's house and takes their TV. James' neighbor calls the police, but they don't take much action on the case. ② Jim makes a racist joke at a housewarming party. Jim's neighbors are appalled and avoid Jim from then on. ③ Wesley hits the dog every time it scratches itself. Wesley's dog becomes terrified of Wesley and won't come near him anymore.	① Jason offers to pay for the vase because he's the one broke it. Jason's girlfriend respects Jason's honesty. ② Jake decides to pick up meditation as a means of coping with stress from his job. Jake returns to his place of employment with newfound energy and enthusiasm. ③ Sandra gives her employees a typical amount of work. Sandra's employees are able to do a good job on the work given to them.	$[3, 4)$
$[-3, -2)$	① Brandon throws a party at his apartment without asking his roommate first. Brandon's roommate feels disrespected and can't study for an important test. ② Jerry leaves the garbage behind as he rushes to his car, assuming that his wife will take care of it. Jerry's wife doesn't take the garbage out in time and it starts attracting flies as they wait for the next garbage day. ③ Gary drinks his beer and eats his pizza, but leaves the cans all over the room along with the empty pizza box laying out. Greg thinks Gary makes their apartment unsanitary and begins looking for someplace else to live.	① Ray reports his friend's website to the FBI so that people stop stealing content. Ray feels like he's helping content creators and is happy that people have to pay for movies now. ② Rob gives his employee paid time off for the entire week. Rob's employee feels grateful and wants to work even harder when they come back. ③ Steve drives all the way to Scott's house to give him the invitation. Scott thanks Steve for the invitation and later enjoys the party.	$[2, 3)$
$[-2, -1)$	① Dean tells his friend what happened while in the middle of class. The class ends up getting interrupted by Dean talking to his friend. ② Jordan goes to the soccer game and forgets the dishes afterward. Jordan gets in trouble for not doing the dishes and feels like a failure. ③ Justine tells Cindy that she is not comfortable lending out her credit card to anyone, even a sister. Cindy understands Justine's reason, and she looks elsewhere for the money.	① Nick considers each applicant, both men and woman, and finally settles on a highly qualified woman. The company does well because the woman Nick chose is an excellent programmer. ② Jim makes a racist joke at a housewarming party. Jim's neighbors are appalled and avoid Jim from then on. ③ Jane finds a friend who is looking for a dog and lets her friend adopt her dog. Jane's dog is thriving and happy at his new home without being left alone all day.	$[1, 2)$
$[-1, -0)$	① Cindy walks all the way to Jimmy's house and knocks at the door. It turns out Jimmy's not home and Cindy walked all that way for nothing. ② Violet complains about the prices to her friends quietly. Violet's friends advise her to shop at a cheaper store. ③ Connie vents her feelings in a private blog post that she sends to Dennis. After reading the post, Denis learns to stop being so pushy about his political beliefs.	① Niro invites Mylah to his house party as friends. Niro and Mylah become platonic friends, and Noah is okay with the situation. ② Cal eats dinner with Tina and asks about her day. Tina will think Cal cares about her and what she does. ③ Arnie enters the libary with his friends and keeps talking about what to study. Arnie and his friends are quickly told to be quiet because it's a libary.	$[0, 1)$

Table 8: Examples with different levels of moral intensity given by MORALSCORE

SOS: Systematic Offensive Stereotyping Bias in Word Embeddings

Fatma Elsafoury[1], Steven R. Wilson[2], Stamos Katsigiannis[3], and Naeem Ramzan[1]

[1]School of Physics, Engineering, and Computing, the University of The West of Scotland, UK
[2]Department of Computer Science and Engineering, Oakland University, USA
[3]Department of Computer Science, Durham University, UK

Abstract

Systematic Offensive stereotyping (SOS) in word embeddings could lead to associating marginalised groups with hate speech and profanity, which might lead to blocking and silencing those groups, especially on social media platforms. In this work, we introduce a quantitative measure of the SOS bias, validate it in the most commonly used word embeddings, and investigate if it explains the performance of different word embeddings on the task of hate speech detection. Results show that SOS bias exists in almost all examined word embeddings and that the proposed SOS bias metric correlates positively with the statistics of published surveys on online extremism. We also show that the proposed metric reveals distinct information compared to established social bias metrics. However, we do not find evidence that SOS bias explains the performance of hate speech detection models based on the different word embeddings.

1 Introduction

Wagner et al. (2021) describe *algorithmically infused societies* as the societies that are shaped by algorithmic and human behaviour. The data collected from these societies carry the same bias in algorithms and humans, like population bias and behavioural bias (Olteanu et al., 2019). These biases are important in the field of natural language processing (NLP) because unsupervised models like word embeddings encode them during training (Brunet et al., 2019; Joseph and Morgan, 2020). This includes racial bias which measures stereotypes related to people from different races, e.g. "Asians are good at math" (Garg et al., 2018; Manzini et al., 2019; Sweeney and Najafian, 2019), and gender bias which measures gender stereotypes, e.g. "women are housewives" (Garg et al., 2018; Bolukbasi et al., 2016; Chaloner and Maldonado, 2019). However, one aspect of bias that has received less attention is offensive stereotyping toward marginalised groups. For example, using slurs to describe non-white or LGBTQ communities or using swear words to describe women. Recent social research shows that using racial slurs and third-person profanity to describe groups of people aims at stressing the inferiority of the identity of the marginalised group (Kukla, 2018). Hence, as the internet is rife with slurs and profanity, it is important to study how machine learning models encode this offensive stereotyping.

To this end, we extend our initial work on introducing a computational measure of *systematic offensive stereotyping* (SOS) bias and examine its existence in pre-trained word embeddings (Elsafoury, 2022). We define SOS from a statistical perspective as "*A systematic association in the word embeddings between profanity and marginalised groups of people*". In other words, SOS refers to associating slurs and profane terms with different groups of people, especially marginalised people, based on their ethnicity, gender, or sexual orientation. Studies that focused on similar types of bias in hate speech detection models studied it within hate speech datasets themselves (Dixon et al., 2018; Waseem and Hovy, 2016a; Zhou et al., 2021), but not in the widely-used word embeddings which are, in contrast, not trained on data specifically curated to contain offensive content. Although some studies demonstrated that there is no correlation between intrinsic bias and extrinsic bias (Goldfarb-Tarrant et al., 2021), studying intrinsic bias on its own is an important task that reveals meaningful information about the data that was used to train those models, and in turn can help to expose harmful biases in society (Garg et al., 2018; Kambhatla et al., 2022).

In this work, we are interested in answering the following research questions: **RQ1:** How can we measure SOS bias? **RQ2:** What are the SOS bias scores of common pre-trained word embeddings,

Proceedings of the 29th International Conference on Computational Linguistics, pages 1263–1274
October 12–17, 2022.

and does SOS bias in the word embeddings differ from social biases? **RQ3:** How strongly does SOS bias correlate with external measures of online extremism and hate? **RQ4:** Does the SOS bias in the word embeddings explain the performance of these word embeddings on the task of hate speech detection? To answer our research questions, we build on the existing literature on measuring bias in word embeddings, propose a method to measure SOS bias, and investigate how different word embedding models associate profanity with marginalised groups.

Our contributions can be summarised as follows: (a) We define the SOS bias, propose a method to measure it in word embeddings and demonstrate that SOS bias correlates positively with the hate that marginalised people experience online. **(b)** We demonstrate that all the examined word embeddings contain SOS bias, with variations on the strength of the bias towards one particular marginalised group or another. **(c)** We show that there is no evidence that the SOS bias explains the performance of the different word embeddings on the task of hate speech detection. To allow more investigation on the topic, we share our code with the community [*].

2 Background

The term *bias* is defined and used in many different ways (Olteanu et al., 2019). There is the normative definition of bias, as its definition in cognitive science: "*behaving according to some cognitive priors and presumed realities that might not be true at all*" (Garrido-Muñoz et al., 2021). There is also the statistical definition of bias as "*systematic distortion in the sampled data that compromises its representatives*" (Olteanu et al., 2019).

In distributional word representations (Word Embeddings), the most common methods for quantifying bias are WEAT, RND, RNSB, and ECT: For WEAT, the authors were inspired by the Implicit Association Test (IAT) to develop a statistical test to demonstrate human-like biases in word embeddings (Caliskan et al., 2017). They used cosine similarity and statistical significance tests to measure the unfair correlations between two different demographic groups, as represented by manually curated word lists. For RND, the authors used the Euclidean distance between neutral words, like professions, and a representative group vector created by averaging the word vectors for words that describe a stereotyped group (gender/ethnicity) (Garg et al., 2018). In RNSB, a logistic regression model is first trained on the word vectors of unbiased labeled sentiment words (positive and negative) extracted from biased word embeddings. Then, that model was used to predict the sentiment of words that describe certain demographic groups (Sweeney and Najafian, 2019). In ECT, the authors proposed a method to measure how much bias has been removed from the word embeddings after debiasing (Dev and Phil, 2019).

These metrics, except RNSB, are based on the polarity between two opposing points, like male and female, allowing for binary comparisons. This forces practitioners to model gender as a spectrum between more "male" and "female" words, requiring an overly simplified view of the construct, leading to similar problems for other stereotypical types of bias, like racial, religious, transgender, and sexual orientation, where there are more than two categories that need to be represented (Sweeney and Najafian, 2019). These metrics also use lists of seed words that have been shown to be unreliable (Antoniak and Mimno, 2021). Since we are interested in measuring the systematic offensive stereotypes of different marginalised groups, these metrics would fall short of our needs. As for the RNSB metric, even though it is possible to include more than two identities, the sentiment dimension is represented as positive or negative (binary). But in our case, we are interested in a variety of offensive language targeted at different marginalised groups.

3 Systematic Offensive Stereotyping Bias

Our motivation is to reveal whether word embeddings associate offensive language with words describing marginalised groups. In the next section, we will use the SOS bias definition provided in the Introduction section to measure the SOS bias. For our experiments, we used 15 word embeddings: Word2Vec (W2V); Glove Wikipedia (Glove-WK); Glove-Twitter (Glove-Twitter); Urban Dictionary (UD); Chan word ; Glove Common Crawl (Glove-CC); Glove Common Crawl Large (Glove-CC-large); Fast-Text Common Crawl (FastText-CC); Fast-Text-Subwords Common Crawl (FT-CC-sws); Fast-Text Wiki (FT-Wiki); Fast-Text-Subwords wiki (FT-wiki-sws); sentiment specific word embedding (SSWE), Debias-W2V, P-DeSIP,

[*] https://github.com/efatmae/measure_SOS_bias_in_static_word_embeddings

Model	Dimensions	Trained on	Reference
W2V	300	100B words from Google News	(Mikolov et al., 2021a)
Glove-WK	200	6B tokens from Wikipedia 2014 and Gigaword	(Pennington et al., 2021)
Glove-Twitter	200	27B tokens collected from two billion Tweets	(Pennington et al., 2021)
UD	300	200M tokens collected from the Urban Dictionary website	(Urban dictionary, 2021)
Chan	150	30M messages from the 4chan and 8chan websites	(GSoC, 2019)
Glove-CC	300	42B tokens from Wikipedia 2014 and Gigaword	(Pennington et al., 2021)
Glove-CC-large	300	840B tokens from Wikipedia 2014 and Gigaword	(Pennington et al., 2021)
FastText-CC	300	600B common crawl tokens	(Mikolov et al., 2021b)
FT-CC-sws	300	600B common crawl tokens with subwords information	(Mikolov et al., 2021b)
FT-Wiki	300	16B tokens collected from Wikipedia 2017, UMBC, and statmt.org news dataset	(Mikolov et al., 2021b)
FT-wiki-sws	300	16 billion tokens with subwords information collected from the Wikipedia 2017, UMBC, and statmt.org	(Mikolov et al., 2021b)
SSWE	50	10M comments collected from Twitter	(Tang et al., 2014)
Debias-W2V	300	W2V model after the gender bias has been removed using the hard debiasing method	(Bolukbasi et al., 2016)
P-DeSIP	300	Debiased Glove-WK with the potential proxy gender bias removed.	(Ding et al., 2022)
U-DeSIP	300	Debiased Glove-WK word embeddings with the unresolved gender bias removed.	(Ding et al., 2022)

Table 1: Description of the word embeddings used in this work.

Group	Words
LGBTQ*	lesbian, gay, queer, homosexual, lgbt, lqbtq, bisexual, transgender, tran, non-binary
Women*	woman, female, girl, wife, sister, mother, daughter
Non-white ethnicities*	african, african american, black, asian, hispanic, latin, mexican, indian, arab, middle eastern
Straight	heterosexual, cisgender
Men	man, male, boy, son, father, husband, brother
White ethnicities	white, caucasian, european american, european, norwegian, canadian, german, australian, english, french, american, swedish, dutch

*Marginalised group

Table 2: Non-offensive identity (NOI) words and the group they describe.

and U-DeSIP. Table 1 provides information of the different word embeddings.

3.1 Measuring SOS bias

Based on our definition of SOS, to answer RQ1, we propose to measure the SOS bias using the cosine similarity between swear words and words that describe marginalised social groups. For the swear words, we used a list (Swear words, 2022) that contains 403 offensive expressions, reduced to 279 after removing multi-word expressions[†]. We used a non-offensive identity (NOI) word list to describe marginalised groups of people (Zhou et al., 2021; Dixon et al., 2018) and non-marginalised ones (Sweeney and Najafian, 2019), as summarised in Table 2. Unlike WEAT, ECT, and RND, which used seed words like people's names to infer their nationality or pronouns, we used NOI words to describe the different groups similar to the RNSB metric. According to (Antoniak and Mimno, 2021), using NOI words is a better motivated and more coherent approach for describing groups of people than names.

Let $W_{NOI} = \{w_1, w_2, w_3, ...w_n\}$ be the list of NOI words w_i, $i = 1, 2, ..., n$, and $W_{sw} = \{o_1, o_2, o_3, ...o_m\}$ be the list of swear words o_j, $j = 1, 2, ..., m$. For measuring the SOS bias for a specific word embedding we, firstly, we compute the average vector $\overrightarrow{\mathbf{W}^{\mathbf{we}}_{\mathbf{sw}}}$ of the swear words for we, e.g. for W2V, etc. $SOS_{i,we}$ for a NOI word w_i and a word embedding we is then defined (Equation 1) as the cosine similarity between $\overrightarrow{\mathbf{W}^{\mathbf{we}}_{\mathbf{sw}}}$ and the word vector $\overrightarrow{w_{i,we}}$, for the word embedding we, normalised to the range $[0, 1]$ using min-max normalisation across all NOI words (W_{NOI}), in order to ease comparison between the different word embeddings.

$$SOS_{i,we} = \frac{\overrightarrow{\mathbf{W}^{\mathbf{we}}_{\mathbf{sw}}} \cdot \overrightarrow{w_{i,we}}}{||\overrightarrow{\mathbf{W}^{\mathbf{we}}_{\mathbf{sw}}}|| \cdot ||\overrightarrow{w_{i,we}}||} \quad (1)$$

The normalised SOS scores are in the range $[0, 1]$ and indicates the similarity of a NOI word to the average representation of swear words. Accordingly, a higher $SOS_{i,we}$ value for word w_i indicates that the word embedding $\overrightarrow{w_{i,we}}$ for the word w_i, is more associated with profanity. We intended for the metric to be used in a comparative manner among word embeddings, e.g. W2V vs Glove-WK, or among different groups of people, e.g. LGBTQ vs Straight, rather than to determine an objective threshold below which no bias exists.

We computed the mean SOS score for our examined word embeddings using the aforementioned swear words and NOI word lists for each examined group individually, as well as for the combined marginalised (Women, LGBTQ, Non-white

[†]We repeated the same experiment with a different set of 427 swear words from (Agrawal and Awekar, 2018) and also observed a significantly higher SOS bias scores for marginalised groups for 11 word embeddings.

Word embeddings	Mean SOS							
	Gender		Sexual orientation		Ethnicity		Marginalised vs. Non-marginalised	
	Women	Men	LGBTQ	Straight	Non-white	White	Marginalised	Non-marginalised
W2V	**0.293**	0.209	0.475	**0.5**	**0.456**	0.390	**0.418**	0.340
Glove-WK	**0.435**	0.347	**0.669**	0.5	**0.234**	0.169	**0.464**	0.260
Glove-Twitter	**0.679**	0.447	**0.454**	0*	**0.464**	0.398	**0.520**	0.376
UD	**0.509**	0.436	**0.582**	0.361	**0.282**	0.244	**0.466**	0.319
Chan	**0.880**	0.699	**0.616**	0.414	**0.326**	0.176	**0.597**	0.373
Glove-CC	**0.567**	0.462	**0.480**	0.195	**0.446**	0.291	**0.493**	0.339
Glove-CC-large	**0.318**	0.192	**0.472**	0.302	**0.548**	0.278	**0.453**	0.252
FT-CC	**0.284**	0.215	0.503	**0.542**	**0.494**	0.311	**0.439**	0.301
FT-CC-sws	**0.473**	0.422	**0.445**	0.277	**0.531**	0.379	**0.480**	0.384
FT-Wiki	**0.528**	0.483	0.555	**0.762**	**0.393**	0.265	**0.496**	0.385
FT-Wiki-sws	0.684	0.684	0.656	**0.798**	0.555	**0.579**	0.632	**0.635**
SSWE	0.619	**0.651**	**0.438**	0*	**0.688**	0.560	**0.569**	0.537
Debias-W2V	**0.205**	0.204	0.446	**0.5**	**0.471**	0.420	**0.386**	0.356
P-DeSIP	**0.266**	0.220	**0.615**	0.491	**0.354**	0.314	**0.434**	0.299
U-DeSIP	**0.266**	0.220	**0.616**	0.492	**0.343**	0.299	**0.431**	0.283

*Glove-Twitter and SSWE did not include the NOI words that describe the "Straight" group.

Table 3: Mean SOS score of the different groups for all the word embeddings. Bold values represent the highest SOS score between the two different groups in each category (gender, sexual orientation, ethnicity, and marginalised vs. non marginalised).

ethnicities) and non-marginalised (Men, Straight, White ethnicities) groups. Table 3 shows that most of the word embeddings are more biased towards the marginalised groups than the non-marginalised groups, with some word embeddings being more SOS biased than others. It also shows that mean SOS bias scores towards the marginalised groups for all the word embeddings, except for Fast-text-wiki-subwords, are higher towards the non-marginalised groups (Wilcoxon $p = 0.0001, \alpha = 0.05$). For Fast-text-wiki-subwords, the SOS bias score for the non-marginalised groups (0.635) is marginally higher than the SOS bias score for the marginalised groups (0.632). In addition, the debiased word embeddings where gender information is removed (Debiased W2V, P-DeSIP, and U-DeSIP), still contain slightly higher SOS bias towards women than men. Given that SOS bias is significantly higher for marginalised groups (Table 3) and that most hate speech datasets contain hate towards women and the marginalised groups, this work subsequently focuses on those groups (Women, LGBTQ, Non-white).

3.2 SOS biased word embeddings

To answer the first part of RQ2, we conducted a comparative analysis of the word embeddings with regard to SOS bias. Table 4 shows the bias scores of each of the word embeddings towards each marginalised group. To quantitatively compare the different word embeddings, we used the SOS bias scores for each marginalised group (LGBTQ, Women, Non-white ethnicities) and applied different significance tests at $\alpha = 0.05$. The results in Table 4 show that Glove-twitter, Chan, Glove-CC, and Fast-text-wiki-subwords are the most biased towards women, with Chan being the most biased ($SOS_{women,Chan} = 0.88$), and Debias-W2V the least biased ($SOS_{women,Debias\text{-}W2V} = 0.205$), which could be due to the fact that Debias-W2V is W2V after removing gender bias. When we used the Friedman test to compare the SOS scores of the different word embeddings for the individual words that describe the "Women" group, the results showed a significant difference between the different word embeddings ($p = 2e^{-11}$), indicating that Chan is significantly more biased towards "Women" in comparison to the rest of the word embeddings. It is worth noting that the reduction in SOS_{women} from 0.435 for Glove-WK to 0.266 for P-DeSIP and U-DeSIP is higher than the reduction achieved for W2V (to Debias-W2V) from 0.293 to 0.205, meaning that U-DeSIP and P-DeSIP used more effective debiasing methods for this category. On the other hand, U-DeSIP and P-DeSIP have higher SOS bias scores toward non-white ethnicities than Glove-WK (as did Debias-W2V compared to W2V), indicating that while bias reduction methods decrease biases toward some groups, they may unintentionally *increase* bias towards others.

Word embeddings	Mean SOS		
	Women	LGBTQ	Non-white
W2V	0.293	**0.475**	0.456
Glove-WK	0.435	**0.669**	0.234
glove-twitter	**0.679**	0.454	0.464
UD	0.509	**0.582**	0.282
Chan	**0.880**	0.616	0.326
Glove-CC	**0.567**	0.480	0.446
Glove-CC-large	0.318	0.472	**0.548**
FT-CC	0.284	**0.503**	0.494
FT-CC-sws	0.473	0.445	**0.531**
FT-WK	0.528	**0.555**	0.393
FT-WK-sws	**0.684**	0.656	0.555
SSWE	0.619	0.438	**0.688**
Debias-W2V	0.205	0.446	**0.471**
P-DeSIP	0.266	**0.615**	0.354
U-DeSIP	0.266	**0.616**	0.343

Table 4: The mean SOS bias score of each word embeddings towards each marginalised group. Bold scores reflect the group that the word embeddings is most biased against.

The LGBTQ community is the group that is most biased against by most of the word embeddings, i.e. W2V, Glove-WK, UD, Fast-text-CC, Fast-text-wiki, P-DeSIP, and U-DeSIP. Glove-WK is the most biased ($SOS_{lgbtq,Glove\text{-}WK} = 0.669$), whereas the least biased is SSWE ($SOS_{lgbtq,SSWE} = 0.438$). When we used the Friedman test to compare the SOS scores of the different word embeddings for the individual words that describe the "LGBTQ" group, the results showed a significant difference between the different word embeddings ($p = 0.048$), indicating that Glove-WK is significantly more SOS biased towards the "LGBTQ" community in comparison to the other word embeddings. These findings are notable as Glove-WK was pre-trained on Wikipedia articles which are expected to have the least profanity compared to social media or common crawl.

Table 4 also shows that Glove-CC-large, Fast-text-CC-subwords, SSWE, and Debias-W2V are the most biased towards non-white ethnicities, with SSWE being the most biased ($SOS_{non\text{-}white,SSWE} = 0.688$) and Glove-WK the least biased ($SOS_{non\text{-}white,Glove\text{-}WK} = 0.234$). When we used the Friedman test to compare the SOS scores of the different word embeddings for the individual words that describe the "Non-white-ethnicities" group, the results showed a significant difference between the different word embeddings ($p = 3e^{-6}$), indicating that SSWE is significantly more biased towards "Non-white-ethnicities" in comparison to the rest of the word embeddings. Since SSWE was pre-trained on sentiment information, and as Sweeney and Najafian (2019) showed, the sentiment towards non-white ethnicities is mostly negative, our results are in line with earlier findings.

3.3 SOS bias and other social biases

In this section, we answer the second part of RQ2 by comparing our SOS bias scores to gender and racial bias as measured by existing social bias metrics from the literature (WEAT, RND, RNSB, ECT). We used the WEFE framework (Badilla et al., 2020) to measure the gender bias using the other state-of-the-art metrics and two target lists: Target list 1, which contained female-related words (e.g., she, woman, and mother), and Target list 2, which contained male-related words (e.g., he, father, and son), as well as two attribute lists: Attribute list 1, which contained words related to family, arts, appearance, sensitivity, stereotypical female roles, and negative words, and Attribute list 2, which contained words related to career, science, math, intelligence, stereotypical male roles, and positive words (Badilla et al., 2020; Caliskan et al., 2017). Then, we measured the average gender bias scores across the different attribute lists for each word embedding using the various metrics. For the SOS bias, we used the mean SOS scores of the words that belong to the "Women" category. Contrary to all the metrics, ECT scores have an inverse relationship with the level of bias, so we subtract all ECT scores from 1 to enforce that higher scores for all metrics indicate greater levels of bias. We then computed the Spearman's rank correlation coefficient between the gender bias scores of the different word embeddings, as measured by WEAT, RND, RNSB, ECT, SOS_{women}.

To measure the racial bias using the state-of-the-art metrics, we used two target groups: Target group 1, which contained stereotypical white names, and Target group 2, which contained stereotypical African, Hispanic, and Asian names, and two attribute lists: Attribute list 1, which contained white people's occupation names; and Attribute list 2, which contained African, Hispanic, and Asian people's occupations (Badilla et al., 2020; Garg et al., 2018). Then, we measured the average racial bias scores across the different attribute lists for each word embedding using the different metrics (WEAT, RND, RNSB, ECT). For the SOS bias, we used the mean SOS scores of the words that belong

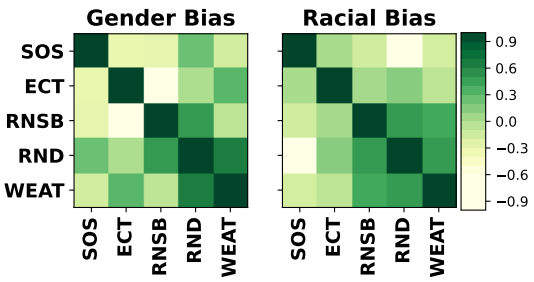

Figure 1: Spearman's correlation between the different bias metrics (SOS and social bias) for all the examined word embeddings. For gender bias, SOS refers to SOS_{women}, and for racial bias to $SOS_{non\text{-}white}$.

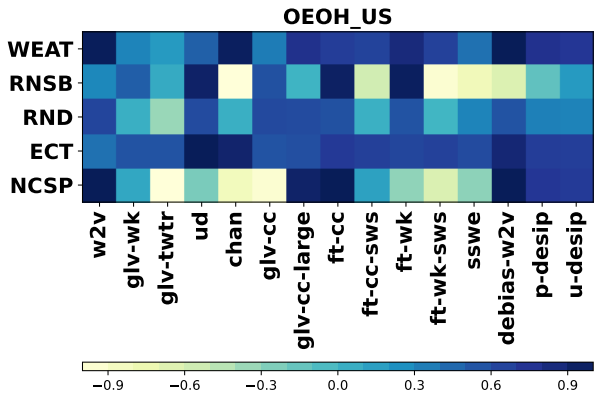

Figure 2: Pearson's correlation between the different SOS bias metrics and the percentages of people belonging to the examined marginalised groups who experienced abuse and extremism online for the OEOH-US survey for the word embeddings.

to the "Non-white ethnicities" category. Finally, we computed the Spearman's rank correlation coefficient between the different racial bias scores of the different word embeddings, as measured by WEAT, RND, RNSB, ECT, $SOS_{non\text{-}white}$.

The results in Figure 1 show that for gender bias, WEAT has a strong positive correlation with RND and a positive correlation with ECT and RNSB. On the other hand, SOS has almost no correlation with ECT, RNSB, WEAT and a small positive correlation with RND. For racial bias, WEAT has a positive correlation with RNSB, and RND, no correlation with ECT and a negative correlation with SOS. On the other hand, SOS has a negative correlation with RNSB, RND, and WEAT and almost no correlation with ECT. The results here suggest that the SOS bias reveals different information than the social bias metrics, especially for racial bias. We speculate that this is the case because profanity is more often used online with non-white ethnicities than with women (Hawdon et al., 2015).

3.4 SOS bias validation

To answer RQ3, we compared the SOS bias measured by our proposed method, as well as by existing metrics (WEAT, RNSB, RND, ECT), to published statistics on online hate and extremism that is targeted at marginalised groups (Women, LGBTQ, Non-white ethnicities). To avoid confusion since all metrics measure SOS bias in this case, we refer to our proposed method for measuring SOS bias as "*normalised cosine similarity to profanity*" or **NCSP** for short. We used the WEFE framework (Badilla et al., 2020) to measure the SOS bias of the examined word embeddings using the state-of-the-art metrics. The metrics in the WEFE platform take 4 inputs: Target list 1: a word list describing a group of people, e.g. women; Target list 2: a word list that describes a different group of people, e.g. men; Attribute list 1: a word list that contains attributes that are believed to be associated with target group 1, e.g. housewife; and Attribute list 2: a word list that contains attributes that are believed to be associated with target group 2, e.g. engineer. Each metric then measures these associations, as described in Section 2.

Country	Sample size	Ethnicity	LGBTQ	Women
Finland	555	0.67	0.63	0.25
US	1033	0.6	0.61	0.44
Germany	978	0.48	0.5	0.2
UK	999	0.57	0.55	0.44

Table 5: The percentage of examined groups that experience online hate and extremism in different countries (Hawdon et al., 2015)

To measure the SOS bias for gender using the state-of-the-art metrics, target list W1 contained the NOI words that describe women from Table 2, target list W2 contained the NOI words that describe men, attribute list 1 contained the same swear words used earlier to measure our SOS bias (Section 3.1), and attribute list 2 a list of positive words provided by the WEFE framework. To measure the SOS bias for ethnicity using the state-of-the-art metrics, we used the same process, with the same attribute lists, but with target list E1 that contained NOI words that describe non-white ethnicities and target list E2 that contained NOI words that describe white ethnicities. Similarly, to measure the SOS bias for sexual orientation, we used the same attribute lists and target list L1, which contained NOI words that describe LGBTQ people, and target list L2 which contained NOI words that describe straight people. To measure the SOS

Dataset	Samples	Positive samples
HateEval	12722	42%
Twitter-sexism	14742	23%
Twitter-racism	13349	15%
Twitter-hate	5569	25%

Note: Positive samples refer to offensive comments

Table 6: Hate speech datasets' details.

bias for gender, ethnicity, and sexual orientation with our proposed metric (NCSP), we computed the mean SOS scores of the NOI words that describe women, LGBTQ, and non-white for each word embeddings as in Table 4.

The percentages of people belonging to the examined marginalised groups who experienced abuse and extremism online were then acquired from the online extremism and online hate survey (OEOH), collected by (Hawdon et al., 2015) from Finland, Germany, the US, and the UK in 2013 and 2014, for individuals aged 15-30. Table 5 provides details on the published statistics. Then, we computed the Pearson's correlation coefficient between the SOS[‡] scores, measured by the different metrics for Women, LGTBQ, and Non-white ethnicities for the examined word embeddings and the percentages of people belonging to the examined marginalised groups who experienced abuse and extremism online. Figure 2[§] shows that the SOS bias correlates positively with the published statistics on online hate and extremism.

When we first look at the different metrics for measuring the SOS bias, we find that bias metrics like WEAT, RND, and ECT correlate more positively with the OEOH survey in the US. However, when we look closely at the order of the percentages of marginalised groups regarding their experience of online hate, we find that the LGBTQ community experiences online hate the most, followed by non-white ethnicities with a marginal difference, and then women. Consequently, we expect that the survey results would correlate strongly positively with the word embeddings that are least biased towards women (e.g. W2V, FT-CC, Debias-W2V, P-DeSIP, and U-DeSIP); correlate less positively with word embeddings that are more biased towards women than LGBTQ or Non-white (e.g. Glove-WK, UD, FT-WK, and SSWE); and correlate negatively with word embeddings that are most biased towards women (e.g. Glove-twitter, Chan, Glove-CC, FT-WK-sws).

This pattern of correlation is achieved only by our proposed metric, which reflects the variation of the SOS bias scores towards the different marginalised groups in each word embedding, in comparison to WEAT, ECT and RND, which do not reflect these variations and hence correlate indiscriminately positively with all the word embeddings. RNSB does reflect some of that variation but not as consistently as our proposed metric. The results suggest that our proposed metric for measuring SOS bias (NCSP) is the most reflective of the SOS bias in the different word embeddings.

4 SOS bias and hate speech detection

In this section, we answer RQ4 through a series of experiments on hate speech detection. We trained deep learning models with an embedding layer for the detection of hate speech from hate speech-related datasets, then computed the correlation of the performance of the different word embeddings to the SOS bias score of these embeddings. We used four hate-speech-related datasets that contain different types of hate speech (Table 6): (i) *Twitter-racism*, a collection of tweets labeled as racist or not (Waseem and Hovy, 2016b); (ii) *Twitter-sexism*, tweets labeled as sexist or not (Waseem and Hovy, 2016b); (iii) Twitter-hate, containing tweets labeled as offensive, hateful (sexist, homophobic, and racist), or neither (Davidson et al., 2017), but as we are interested in the hateful content, we used the tweets that are labeled as hateful or neither; and (iv) *HateEval*, a collection of tweets containing hate against immigrants and women in Spanish and English (Basile et al., 2019), from which we used only the English tweets. These four datasets were selected because they contain hate speech towards the marginalised groups that are the focus of our study thus they are representative of the examined problem.

To pre-process the datasets, we removed URLs, user mentions, retweet abbreviation "RT", non-ASCII characters, and English stop words except for second-person pronouns like "you/yours/your", and third-person pronouns like "he/she/they", "his/her/their" and "him/her/them", as suggested in (Elsafoury et al., 2021). All letters were lowercased, and common contractions were converted to their full forms. And each dataset was randomly

[‡]We subtract all ECT scores from 1 here as well.

[§]The correlation results for OEOH-US are similar to the correlation results from OEOH-Finland, OEOH-UK and OEOH-Germany, and thus are omitted from Figure 2.

Word embeddings	HateEval		Twitter-Hate		Twitter-racism		Twitter-sexism	
	MLP	BiLSTM	MLP	BiLSTM	MLP	BiLSTM	MLP	BiLSTM
W2V	0.593	0.663	0.681	0.772	0.683	0.717	0.587	0.628
Glove-WK	0.583	0.651	0.713	0.821	0.681	0.727	0.587	0.641
Glove-Twitter	0.623	0.671	0.775	0.851	0.680	0.699	0.589	**0.668**
UD	0.597	0.652	0.780	0.837	0.679	0.698	0.578	0.632
Chan	**0.627**	0.661	0.692	0.840	0.650	0.712	0.563	0.647
Glove-CC	0.625	**0.675**	0.778	0.839	0.695	0.740	0.577	0.648
Glove-CC-large	0.626	0.674	0.775	**0.860**	**0.709**	0.724	0.593	**0.668**
FT-CC	**0.627**	**0.675**	**0.792**	0.843	0.701	0.741	**0.607**	0.654
FT-CC-sws	0.605	0.660	0.746	0.830	0.701	**0.746**	0.588	0.657
FT-WK	0.606	0.650	0.784	0.827	0.699	0.706	0.601	0.653
FT-WK-sws	0.606	0.650	0.723	0.820	0.689	0.736	0.561	0.633
SSWE	0.558	0.628	0.502	0.715	0.324	0.666	0.171	0.548
Debiased-W2V	0.626	0.652	0.678	0.741	0.674	0.715	0.564	0.638
P-DeSIP	0.575	0.657	0.697	0.817	0.673	0.731	0.538	0.650
U-DeSIP	0.598	0.649	0.702	0.815	0.673	0.726	0.548	0.638

Table 7: F1 scores for the used models for hate speech detection using the examined word embeddings on the examined datasets. Bold values indicate the highest scores among the different word embeddings per model and dataset.

Dataset	Model	WEAT	RNSB	RND	ECT	NCSP
HateEval	MLP	**0.277**	0.223	-0.100	0.019	0.230
	BiLSTM	0.377	**0.540***	0.094	-0.030	0.100
Twitter Sexism	MLP	**0.157**	0.030	-0.216	-0.039	0.121
	BiLSTM	0.109	**0.266**	0.093	-0.361	0.246
Twitter Racism	MLP	0.042	0.017	-0.336	-0.223	**0.241**
	BiLSTM	-0.264	**0.135**	-0.210	-0.103	0.110
Twitter Hate	MLP	0.107	0.218	-0.164	-0.148	**0.223**
	BiLSTM	**0.507**	0.475	0.289	-0.217	0.396

Table 8: Pearson correlation coefficient of the SOS bias scores of the different word embeddings and the F1 scores of the used models for each bias metric and dataset. * indicates that the correlation is statistically significant at $p < 0.05$.

split into a training (70%) and a test (30%) set, preserving class ratios.

We used two deep learning models: (i) a Bidirectional LSTM (Schuster and Paliwal, 1997) with the same architecture as in (Agrawal and Awekar, 2018), who used RNN models to detect hate speech, and (ii) a two-layer Multi-Layer Perceptron (MLP) model. To this end, we first used the Keras tokenizer (Tensorflow.org, 2020) to tokenise the input texts, using a maximum input length of 64 (maximum observed sequence length in the dataset). A frozen embedding layer, based on a given pretrained word embedding model, was used as the first layer and fed to the BiLSTM model and the MLP model. To avoid over-fitting, we used L2 regularisation with an experimentally determined value of 10^{-7}. The models were trained for 100 epochs with a batch size of 32, using the Adam optimiser and a learning rate of 0.01 (default of Keras Optimiser) (Agrawal and Awekar, 2018). For each dataset, we used a 5-fold cross-validation to train and validate a model (70% and 30% of the training set respectively, with class ratio preserved) and then test each fold's model on the test set. Then, the average F1-score across the 5 folds was reported.

4.1 Experimental Results

Given the results for the SOS bias in the different embeddings (Table 4), we hypothesise that the deep learning models that are trained with Glove-CC-large, FastText-CC-subwords, SSWE, and Debias-W2V embeddings will perform the best (highest F1 score) on datasets that contain hate speech or insults towards marginalised ethnicities, which is Twitter-racism. We also hypothesise that the models trained with Glove-Twitter, Chan, Glove-CC, and Fast-text-wiki-subwords will achieve the highest F1 scores on datasets that contain insults towards women, which is Twitter-sexism. Since Twitter-Hate and HateEval contain a mixture of hateful content towards women and immigrants, we hypothesise that the best performing word embeddings would be the ones that have SOS scores higher than the median values for both of SOS_{women}

(0.473) and $SOS_{Non\text{-}white}$ (0.456), which are Glove-Twitter, Fast-text-wiki-subwords, and SSWE.

The performance of the deep learning models with the different embedding models is reported in Table 7. The results show that for all datasets, BiLSTM outperforms MLP in terms of F1 score. The results also show that for the MLP model, our hypotheses hold for the Twitter-racism dataset, as the best performing models are BiLSTM with Fast-text-CC-subwords and MLP with Glove-CC-large. However, for Twitter-sexism, HateEval, and Twitter-Hate, the results do not support our hypothesis, with Fast-text-CC and Glove-CC-large being the best performing with MLP and BiLSTM models. To quantify our analysis we used the Spearman's correlation between the SOS bias scores, measured using the different bias metrics, of the different word embeddings and the F1 scores of the MLP and BiLSTM trained with the different word embeddings. The results in Table 8 show occasionally positive correlations for example with WEAT, RNSB, and our proposed metric NCSP. However, most of these positive correlations are not statistically significant except for the SOS scores measured by the RNSB metric and the F1 of the BiLSTM model and the HateEval dataset. These results show that there is no positive correlation between the SOS bias scores in the word embeddings and the performance of the hate speech detection models, suggesting that the SOS bias in the word embeddings does not explain their utility as features for hate speech detection.

5 Conclusion

In this work, we built on our initial work (Elsafoury, 2022) where the SOS bias was introduced and proposed methods to measure it, validate it, compare it to stereotypical social bias, and investigate if it explains the performance of the word embeddings on hate speech detection. Results show that the examined word embeddings are SOS biased and that the SOS bias in the word embeddings has a strong positive correlation with published statistics on online extremism. However, more datasets need to be collected to provide stronger evidence, especially data from social sciences on the offences that marginalised groups receive on social media. Nonetheless, this is an informative finding as it reveals the bias in the dataset that these word embeddings were trained on. Since not all these datasets are available to the public, measuring the SOS bias in the word embeddings is an important way to learn about that bias in those datasets.

Our findings also show that the proposed SOS bias reveals different information than the types of bias measured by existing metrics. Finally, our findings show no evidence that the SOS bias, measured using different bias metrics, explains the performance of the different word embeddings on the task of hate speech detection. This finding suggests that the SOS bias, and potentially other biases in general, are not strongly related to word embeddings' performance on the downstream task of hate speech detection. We plan to examine this speculation and study the influence of the SOS and social bias on the *fairness* of hate speech detection models in future work.

6 Limitations

The findings demonstrated in this paper are limited to the inspected word embeddings, models, and datasets, and might not generalise to other datasets. Similarly, our SOS bias scores are limited to the used word lists and even if we used two different swear word lists and identity terms that are coherent according to (Antoniak and Mimno, 2021), using different word lists may give different results. Another limitation is regarding our definition of the SOS bias, as we defined bias from a statistical perspective which lacks the social science perspective as discussed in (Blodgett et al., 2021; Delobelle et al., 2022). Moreover, we only studied bias in Western societies where Women, LGBTQ and Non-White ethnicities are among the marginalised groups. However marginalised groups could include different groups of people in other societies. We also only used datasets and word lists in English which limits our study to the English speaking world. Similar to other works on quantifying bias, our proposed metric measures the existence of bias and not its absence (May et al., 2019), and thus low bias scores do not necessarily mean the absence of bias or discrimination in the word embeddings.

References

Sweta Agrawal and Amit Awekar. 2018. Deep learning for detecting cyberbullying across multiple social media platforms. In *Advances in Information Retrieval - 40th European Conference on IR Research, ECIR 2018, Grenoble, France, March 26-29, 2018, Proceedings*, volume 10772 of *Lecture Notes in Computer Science*, pages 141–153. Springer.

Maria Antoniak and David Mimno. 2021. Bad seeds: Evaluating lexical methods for bias measurement. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 1889–1904, Online. Association for Computational Linguistics.

Pablo Badilla, Felipe Bravo-Marquez, and Jorge Pérez. 2020. WEFE: the word embeddings fairness evaluation framework. In *Proceedings of the Twenty-Ninth International Joint Conference on Artificial Intelligence, IJCAI 2020*, pages 430–436. ijcai.org.

Valerio Basile, Cristina Bosco, Elisabetta Fersini, Debora Nozza, Viviana Patti, Francisco Manuel Rangel Pardo, Paolo Rosso, and Manuela Sanguinetti. 2019. SemEval-2019 task 5: Multilingual detection of hate speech against immigrants and women in Twitter. In *Proceedings of the 13th International Workshop on Semantic Evaluation*, pages 54–63, Minneapolis, Minnesota, USA. Association for Computational Linguistics.

Su Lin Blodgett, Gilsinia Lopez, Alexandra Olteanu, Robert Sim, and Hanna M. Wallach. 2021. Stereotyping norwegian salmon: An inventory of pitfalls in fairness benchmark datasets. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing, ACL/IJCNLP 2021, (Volume 1: Long Papers), Virtual Event, August 1-6, 2021*, pages 1004–1015. Association for Computational Linguistics.

Tolga Bolukbasi, Kai-Wei Chang, James Zou, Venkatesh Saligrama, and Adam Kalai. 2016. Man is to computer programmer as woman is to homemaker? debiasing word embeddings. In *Proceedings of the 30th International Conference on Neural Information Processing Systems*, NIPS'16, page 4356–4364, Red Hook, NY, USA. Curran Associates Inc.

Marc-Etienne Brunet, Colleen Alkalay-Houlihan, Ashton Anderson, and Richard S. Zemel. 2019. Understanding the origins of bias in word embeddings. In *Proceedings of the 36th International Conference on Machine Learning, ICML 2019, 9-15 June 2019, Long Beach, California, USA*, volume 97 of *Proceedings of Machine Learning Research*, pages 803–811. PMLR.

Aylin Caliskan, Joanna J. Bryson, and Arvind Narayanan. 2017. Semantics derived automatically from language corpora contain human-like biases. *Science*, 356(6334):183–186.

Kaytlin Chaloner and Alfredo Maldonado. 2019. Measuring gender bias in word embeddings across domains and discovering new gender bias word categories. In *Proceedings of the First Workshop on Gender Bias in Natural Language Processing*, pages 25–32, Florence, Italy. Association for Computational Linguistics.

Thomas Davidson, Dana Warmsley, Michael W. Macy, and Ingmar Weber. 2017. Automated hate speech detection and the problem of offensive language. In *Proceedings of the Eleventh International Conference on Web and Social Media, ICWSM 2017, Montréal, Québec, Canada, May 15-18, 2017*, pages 512–515. AAAI Press.

Pieter Delobelle, Ewoenam Kwaku Tokpo, Toon Calders, and Bettina Berendt. 2022. Measuring fairness with biased rulers: A comparative study on bias metrics for pre-trained language models. In *Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL 2022, Seattle, WA, United States, July 10-15, 2022*, pages 1693–1706. Association for Computational Linguistics.

Sunipa Dev and Jeff M. Phil. 2019. Attenuating bias in word vectors. In *The 22nd International Conference on Artificial Intelligence and Statistics, AISTATS 2019, 16-18 April 2019, Naha, Okinawa, Japan*, volume 89 of *Proceedings of Machine Learning Research*, pages 879–887. PMLR.

Lei Ding, Dengdeng Yu, Jinhan Xie, Wenxing Guo, Shenggang Hu, Meichen Liu, Linglong Kong, Hongsheng Dai, Yanchun Bao, and Bei Jiang. 2022. Word embeddings via causal inference: Gender bias reducing and semantic information preserving. In *Thirty-Sixth AAAI Conference on Artificial Intelligence, AAAI 2022, Thirty-Fourth Conference on Innovative Applications of Artificial Intelligence, IAAI 2022, The Twelveth Symposium on Educational Advances in Artificial Intelligence, EAAI 2022 Virtual Event, February 22 - March 1, 2022*, pages 11864–11872. AAAI Press.

Lucas Dixon, John Li, Jeffrey Sorensen, Nithum Thain, and Lucy Vasserman. 2018. Measuring and mitigating unintended bias in text classification. In *Proceedings of the 2018 AAAI/ACM Conference on AI, Ethics, and Society*, AIES '18, page 67–73, New York, NY, USA. Association for Computing Machinery.

Fatma Elsafoury. 2022. Darkness can not drive out darkness: Investigating bias in hate speechdetection models. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics: Student Research Workshop, ACL 2022, Dublin, Ireland, May 22-27, 2022*, pages 31–43. Association for Computational Linguistics.

Fatma Elsafoury, Stamos Katsigiannis, Steven R. Wilson, and Naeem Ramzan. 2021. Does BERT pay attention to cyberbullying? In *Proceedings of the 44th International ACM SIGIR Conference on Research and Development in Information Retrieval*, page 1900–1904, New York, NY, USA. Association for Computing Machinery.

Nikhil Garg, Londa Schiebinger, Dan Jurafsky, and James Zou. 2018. Word embeddings quantify 100 years of gender and ethnic stereotypes. *Proceedings*

of the National Academy of Sciences, 115(16):E3635–E3644.

Ismael Garrido-Muñoz, Arturo Montejo-Ráez, Fernando Martínez-Santiago, and L. Alfonso Ureña-López. 2021. A survey on bias in deep nlp. *Applied Sciences*, 11(7).

Seraphina Goldfarb-Tarrant, Rebecca Marchant, Ricardo Muñoz Sánchez, Mugdha Pandya, and Adam Lopez. 2021. Intrinsic bias metrics do not correlate with application bias. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 1926–1940, Online. Association for Computational Linguistics.

GSoC. 2019. 4 and 8 chan embeddings. [Online] Accessed 05/11/2021.

James Hawdon, Atte Oksanen, and Pekka Räsänen. 2015. Online extremism and online hate. *NordicomInformation*, 37:29–37.

Kenneth Joseph and Jonathan Morgan. 2020. When do word embeddings accurately reflect surveys on our beliefs about people? In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 4392–4415, Online. Association for Computational Linguistics.

Gauri Kambhatla, Ian Stewart, and Rada Mihalcea. 2022. Surfacing racial stereotypes through identity portrayal. In *FAccT '22: 2022 ACM Conference on Fairness, Accountability, and Transparency, Seoul, Republic of Korea, June 21 - 24, 2022*, pages 1604–1615. ACM.

Rebecca Kukla. 2018. Slurs, interpellation, and ideology. *The Southern Journal of Philosophy*, 56:7–32.

Thomas Manzini, Lim Yao Chong, Alan W Black, and Yulia Tsvetkov. 2019. Black is to criminal as caucasian is to police: Detecting and removing multiclass bias in word embeddings. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 615–621, Minneapolis, Minnesota. Association for Computational Linguistics.

Chandler May, Alex Wang, Shikha Bordia, Samuel R. Bowman, and Rachel Rudinger. 2019. On measuring social biases in sentence encoders. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT 2019, Minneapolis, MN, USA, June 2-7, 2019, Volume 1 (Long and Short Papers)*, pages 622–628. Association for Computational Linguistics.

Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. 2021a. word2vec embeddings. [Online] Accessed 05/11/2021.

Tomas Mikolov, Edouard Grave, Piotr Bojanowski, Christian Puhrsch, and Armand Joulin. 2021b. Glove twitter embeddings. [Online] Accessed 25/04/2022.

Alexandra Olteanu, Carlos Castillo, Fernando Diaz, and Emre Kıcıman. 2019. Social data: Biases, methodological pitfalls, and ethical boundaries. *Frontiers in Big Data*, 2:13.

Jeffrey Pennington, Richard Socher, and Christopher D. Manning. 2021. Glove twitter embeddings. [Online] Accessed 05/11/2021.

Mike Schuster and Kuldip K Paliwal. 1997. Bidirectional recurrent neural networks. *IEEE Transactions on Signal Processing*, 45(11):2673–2681.

Swear words. 2022. Swear words list. [Online] Accessed 26/04/2022.

Chris Sweeney and Maryam Najafian. 2019. A transparent framework for evaluating unintended demographic bias in word embeddings. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 1662–1667, Florence, Italy. Association for Computational Linguistics.

Duyu Tang, Furu Wei, Nan Yang, Ming Zhou, Ting Liu, and Bing Qin. 2014. Learning sentiment-specific word embedding for Twitter sentiment classification. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1555–1565, Baltimore, Maryland. Association for Computational Linguistics.

Tensorflow.org. 2020. Text tokenization utility class. `https://www.tensorflow.org/api_docs/python/tf/keras/preprocessing/text/Tokenizer`. Accessed: 2020-09-28.

Urban dictionary. 2021. Urban dictionary embeddings. [Online] Accessed 05/11/2021.

Claudia Wagner, Markus Strohmaier, Alexandra Olteanu, Emre Kıcıman, Noshir Contractor, and Tina Eliassi-Rad. 2021. Measuring algorithmically infused societies. *Nature*, 595(7866):197–204.

Zeerak Waseem and Dirk Hovy. 2016a. Hateful symbols or hateful people? predictive features for hate speech detection on twitter. In *Proceedings of the Student Research Workshop, SRW@HLT-NAACL 2016, The 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, San Diego California, USA, June 12-17, 2016*, pages 88–93. The Association for Computational Linguistics.

Zeerak Waseem and Dirk Hovy. 2016b. Hateful symbols or hateful people? predictive features for hate speech detection on Twitter. In *Proceedings of the NAACL Student Research Workshop*, pages 88–93, San Diego, California. Association for Computational Linguistics.

Xuhui Zhou, Maarten Sap, Swabha Swayamdipta, Yejin Choi, and Noah A. Smith. 2021. Challenges in automated debiasing for toxic language detection. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume, EACL 2021, Online, April 19 - 23, 2021*, pages 3143–3155. Association for Computational Linguistics.

Bigger Data or Fairer Data? Augmenting BERT via Active Sampling for Educational Text Classification

Lele Sha, Yuheng Li, Dragan Gašević and **Guanliang Chen***
Centre for Learning Analytics, Monash University
Melbourne, Victoria, Australia
{lele.sha1, yuheng.li, Dragan.Gasevic, Guanliang.Chen}@monash.edu

Abstract

Pretrained Language Models (PLMs), though popular, have been diagnosed to encode bias against protected groups in the representations they learn, which may harm the prediction fairness of downstream models. Given that such bias is believed to be related to the amount of demographic information carried in the learned representations, this study aimed to quantify the awareness that a PLM (i.e., BERT) has regarding people's protected attributes and augment BERT to improve prediction fairness of downstream models by inhibiting this awareness. Specifically, we developed a method to dynamically sample data to continue the pretraining of BERT and enable it to generate representations carrying minimal demographic information, which can be directly used as input to downstream models for fairer predictions. By experimenting on the task of classifying educational forum posts and measuring fairness between students of different *gender* or *first-language backgrounds*, we showed that, compared to a baseline without any additional pretraining, our method improved not only fairness (with a maximum improvement of 52.33%) but also accuracy (with a maximum improvement of 2.53%). Our method can be generalized to any PLM and demographic attributes. All the codes used in this study can be accessed via https://github.com/lsha49/FairBERT_deploy.

1 Introduction

Pretrained Language Models (PLMs) have been increasingly applied to tackle various NLP tasks in recent years (Li et al., 2019; Yoon et al., 2019; Chan and Fan, 2019; Araci, 2019; Zhang et al., 2019). Along with the wide application of PLMs is growing concerns about the bias encoded in the representations generated by these PLMs (Jin et al., 2020; Lu et al., 2020). For instance, de Vassimon Manela et al. (2021) demonstrated that stereotypical associations were encoded in PLMs when tackling a pronoun resolution task. More importantly, such bias has been demonstrated to be harmful to the prediction *fairness* of downstream models, i.e., there exists a consistent gap between the prediction accuracy for people of different protected attributes. For example, Minot et al. (2021) showed that the gender-related bias embedded in PLMs could be propagated to downstream classification tasks in medical scenarios.

As a remedy, researchers have endeavored to develop techniques to *debias* PLMs. These techniques, more often than not, focused on the fine-tuning stage when using a PLM, e.g., correcting the bias hidden behind the learned representations by removing associations between embedding features and protected attributes during the fine-tuning process or using a protected-attribute-balanced dataset to fine-tune a PLM (de Vassimon Manela et al., 2021). It is worth noting that, though assuming that the bias contained in a PLM is associated with the amount of demographic information carried in the learned representations, these debiasing techniques oftentimes failed to (i) explicitly quantify the capability of a PLM in revealing people's demographic attributes or (ii) depict the relationship between the amount of demographic information contained in the learned representations and the prediction fairness in downstream tasks.

Inspired by the studies which demonstrated the benefits of using additional task-specific data to continue to pretrain a language model and boost prediction accuracy (Araci, 2019; Clavié and Gal, 2019; Chalkidis et al., 2020; Shen et al., 2021; Beltagy et al., 2019), in this study, we focused on continuing the pretraining of a language model (i.e., BERT (Devlin et al., 2018)) with carefully-selected data so as to reduce the amount of demographic information contained in the learned representations and subsequently enhance the downstream predic-

* Corresponding author.

Proceedings of the 29th International Conference on Computational Linguistics, pages 1275–1285
October 12–17, 2022.

tion models in terms of both *accuracy* and *fairness*. Our rationale is essentially in line with those held in (de Vassimon Manela et al., 2021; Minot et al., 2021), i.e., the bias carried in a PLM can be potentially reduced by inhibiting a PLM's awareness of people's protected attributes. Formally, this study was guided by the following **R**esearch **Q**uestions:

RQ1 To what extent can the representations generated by a vanilla BERT[1] predict people's protected attributes?

RQ2 To what extent can BERT's awareness of protected attributes be inhibited by actively sample data to continue its pretraining?

RQ3 What are the impacts of inhibiting BERT's awareness of protected attributes on the prediction fairness in the downstream model?

We based our study on the task of classifying discussion forum posts in education, which is widely recognized as important in assisting instructors to provide timely support to students, especially in courses with a high student-teacher ratio (Ntourmas et al., 2021; Wei et al., 2017). This study was approved by the Human Research Ethics Committee at Monash university (Project ID 30074). We used a dataset consisting of over 228K forum posts generated by students when undertaking their studies at the same university, and information about students' *gender* and *first-language backgrounds* were also contained in the dataset. To answer RQ1, we used the representations generated by the vanilla BERT as input to a logistic regression model to predict students' protected attributes. To answer RQ2, building upon studies on Active Learning (AL), we proposed a data sampling method to selectively sample data, which contain minimal information about protected attributes, to continue to pretrain BERT, after which we measured whether the learned representations became less capable of revealing students' protected attributes. To answer RQ3, after applying additional pretraining to BERT, we further used the the learned representations as input to a different logistic regression model to predict the categorical label of a forum post (i.e., content relevant or irrelevant). Through extensive evaluations, we demonstrated that the proposed sampling method can effectively identify data to decrease a PLM's awareness of protected attributes and enhance predictive models used in downstream tasks in terms of both accuracy and fairness. Our sampling method can be generalized to any PLM or protected attributes.

[1] A vanilla BERT refers to one without any additional pretraining.

2 Related Work

2.1 Bias in Pretrained Language Models

PLMs have been documented to contain biases against certain socio-demographic groups (e.g., black and female), which was partially caused by the use of low-quality data when constructing a PLM (Lucy and Bamman, 2021; Nadeem et al., 2020; de Vassimon Manela et al., 2021). Nadeem et al. (2020) showed that harmful stereotypes commonly existed in online text. When using such online texts for training, PLMs can easily pick up harmful stereotypes and act against the disadvantaged groups. For instance, when predicting the emotion polarity and toxicity of a piece of text, PLMs are prone to classify text written by females as more emotional than those written by their male counterpart (Jin et al., 2020; Touileb et al., 2021; Silva et al., 2021; Bhardwaj et al., 2021; Mozafari et al., 2020). Another reason is that the data generated by disadvantaged groups was less used in constructing a PLM and thus causing these disadvantaged groups to be under-represented compared to other groups. When detecting hate speech, texts written in African American English dialect were more likely to be mistakenly classified than texts written in standard English (Halevy et al., 2021).

2.2 Debiasing Pretrained Language Models

Existing studies in this strand of research mostly stressed on the fine-tuning stage when using a PLM (Bhardwaj et al., 2021; Silva et al., 2021; Jin et al., 2020). Among these studies, a majority of them aimed to debias the learned representations by regularizing the BERT model during fine-tuning. For instance, Bhardwaj et al. (2021) proposed a method to identify and remove the semantic features which contained sensitive information (e.g., gender-related) when propagating through BERT layers, thereby reducing BERT-induced bias in the downstream tasks. Silva et al. (2021) applied a loss regularizer where a loss is incorporated in training to minimize bias learned during fine-tuning. Alternatively, some researchers attempted to debias a PLM by modifying the task-specific data samples before using them to fine-tune the PLM (Prost et al., 2019; Islam et al., 2021; Pruksachatkun et al.,

2021). Typically, the data was modified with the aim of removing traits that are indicative of people's gender (de Vassimon Manela et al., 2021; Minot et al., 2021), race (Mozafari et al., 2020), or dialect (Mozafari et al., 2020). These data modification methods have been demonstrated effective in enhancing the prediction fairness of downstream models in pronoun resolution (de Vassimon Manela et al., 2021) and toxicity detection (Mozafari et al., 2020). In addition, researchers demonstrated that a PLM could be debiased by enabling the downstream task model to work with other models in an ensemble-based manner (Halevy et al., 2021). For example, Halevy et al. (2021) showed that by adding a specialized classifier trained by text written in the African American English dialect to the ensemble framework, the model displayed fewer racial biases when detecting toxic language compared to using PLM alone.

Our work shared a similar rationale with (Gajane and Pechenizkiy, 2017; Minot et al., 2021; Mozafari et al., 2020), i.e., by reducing a PLM's awareness of the protected attributes related to data, the PLM is less likely to propagate bias to the model used in a downstream task and the prediction fairness of the task model can be enhanced. However, our work distinguished itself from two aspects. Firstly, instead of focusing on the fine-tuning stage, we focused on debiasing a PLM by actively sampling protected-attribute-uninformative data to continue to pretrain the PLM. Secondly, we explicitly quantified the capability of the PLM in predicting protected attributes and measure its impact on the prediction fairness of the downstream model.

2.3 Pretrained Language Models in Education

Driven by the great success in the broader NLP communities, PLMs have been also applied in solving various tasks in the field of education, such as generating questions for assessment (Lu et al., 2021), providing timely feedback to support student learning (Lin et al., 2022), and scoring answers or essays authored by students (Ormerod et al., 2021). Among these tasks, the classification of forum posts has received lots of attention from researchers due to its important role in facilitating instructors to support students in the era of online learning (Clavié and Gal, 2019; Alrajhi et al., 2020; Geller et al., 2021; Capuano et al., 2021). For instance, Clavié and Gal (2019) further pretrained BERT using forum posts collected in the education domain and classified students' posts to a task label of whether these posts requires urgent attention from instructors or not. The constructed classifiers could assist instructor to quickly identify students that require urgent help. However, these studies did not attempt to quantify or alleviate the impact from the bias hidden behind PLMs. Considering that education is often regarded as a high-stake commodity, we were thus motivated to investigate the bias of PLM based on the task of classifying student-generated forum posts.

3 Method

3.1 Dataset and Models

Dataset. The dataset used in this study was retrieved from the Learning Management System at an Australian university. The original dataset consisted of 291,242 student-generated posts in discussion forums when undertaking courses of Information Technology, Engineering, Education, Business and Economics, etc. In addition, we obtained students' demographic information including their gender (female vs. male) and first language, which enabled us to investigate prediction fairness from gender and first-language backgrounds perspectives. Inspired by (Loukina et al., 2019), which demonstrated that English-as-second-language students could be disadvantaged by algorithms used for assessing their learning performance, we categorised students according to their *first-language backgrounds* as either English-as-first-language or English-as-second-language students. After filtering posts containing less than 5 words, there were 228,903 posts left, from which we randomly selected 3,703 posts and manually annotated them as either *content-relevant* (e.g., "*What is poly-nominal regression?*") and *content-irrelevant* (e.g., "*When is the due date to submit the assignment?*"). Each post was first labeled by a junior teaching staff and then reviewed by two senior teaching staff to ensure the reliability of the derived labels. There are 2,339 (63%) content-relevant posts and 1,364 (37%) content-irrelevant posts. We denoted these 3,703 posts as **Annotated Data**. Recall that part of our goal was to reveal the capability of BERT in predicting students' demographic attributes, we therefore randomly select 5% of the remaining 225,200 unannotated posts and used them as an indepedent data set (denoted as **Demographic Data**) to scrutinize how BERT would differ in predicting demographic attributes after undergoing additional

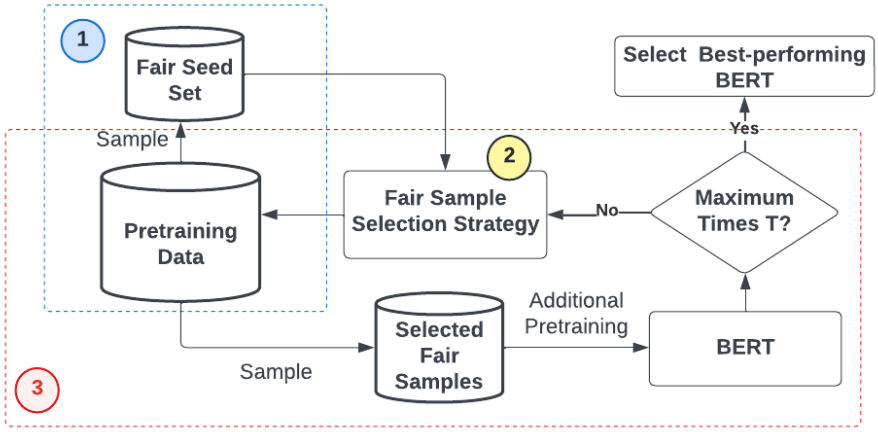

Figure 1: The fair active sampling method proposed in this study.

Table 1: Dataset statistics. The columns **Male**, **Female**, **First**, and **Second** show the number of posts made by students who are male, female, English-as-first-language, and English-as-second-language students, respectively.

		All	Male	Female	First	Second
Pretraining	# Posts	213,940	84,564	140,374	97,255	127,945
	# Words	32,264,332	12,212,745	20,015,047	13,674,012	18,590,320
	# Avg. words / post	143.27	144.42	142.58	140.60	145.30
Demographic	# Posts	11,260	4,509	6,751	4,955	6,305
	# Words	1,522,127	568,810	953,308	712,231	809,877
	# Avg. words / post	135.18	126.15	141.21	143.74	128.45
Annotated	# Posts	3,703	1,478	2,225	1,585	2,112
	# Words	485,737	171,768	308,087	230,806	254,931
	# Avg. words / post	131.39	116.77	138.90	145.62	120.71

pretraining. Lastly, the remaining 213,940 unannotated posts were used as the data pool from which we sampled instances to apply additional pretraining to BERT (denoted as **Pretraining Data**). The statistics of the three sets are given in Table 1.

Models. A variety of effective models have been used to classify a forum post in education (Bakharia, 2016; Capuano et al., 2021; Almatrafi et al., 2018). Given that our main goal was to investigate whether the changes in the BERT-generated representations in terms of the amount of demographic information would produce any impact on the prediction fairness in a downstream task, we adopted logistic regression to take the BERT-generated post embeddings as input to distinguish the different labels of forum posts for simplicity. We denoted this model as `LR-Task`. Similarly, we also used logistic regression to predict the demographic attributes of the creator of a post, i.e., one logistic regression model for each demographic attribute. We denoted these two models as `LR-Demographic`.

3.2 Fair Active Sampling

Our data sampling method was partially inspired by the studies on Active Learning (AL) (Anahideh et al., 2022; Castro and Nowak, 2008). AL is a well-investigated strategy used to train a supervised machine learning model by enabling the model to proactively request to identify and label new high-value data samples to facilitate its training process and achieve better prediction accuracy. Considering the strong capacity of AL in enhancing various machine learning models, we were interested whether it could be used to alleviate the bias contained in a PLM by selecting fair data (i.e., those containing little information about students' protected attributes) to continue to pretrain the PLM. There are three key steps in our method, as depicted in Figure 1 and detailed below.

Step 1: Fair Seed Set Construction (i.e., ① depicted in Figure 1). When applying AL, researchers often first randomly select a small subset from the unlabeled data pool and labeled them to initialize the training of a machine learning model. Then,

this subset of labeled data can be referenced as a seed set to sample more informative data (e.g., those would help the model to reduce the maximum training error) to continue the model training. In our case, as our goal was to enhance a PLM's fairness by reducing the demographic information contained in the learned representations, we aimed to construct a *fair* seed set from the annotated data pool to guide the subsequent active data sampling. To this end, we expected the fair seed set to contain zero *distribution bias* and minimal *hardness bias* (Yan et al., 2020; Smith et al., 2014). Distribution bias refers to the unequal distribution of different student groups (e.g., female vs. male) within each prediction task label (e.g., content relevant vs. content irrelevant). Formally, it can be defined as the the difference of probabilities of prediction task label $Y = y$, conditioned upon the value of a protected attribute G where $G \in \{0, 1\}$ and $Y \in \{0, 1\}$:

$$D(y) = Pr(Y = y \mid G = 1) - Pr(Y = y \mid G = 0) \quad (1)$$

A seed set that contains zero distribution bias indicates $D(y) = 0$. On the other hands, Hardness bias measures to what extent data instances contained in a dataset are difficult to be correctly labeled. That is, if a data instance does not share the same task label with most of its *k-nearest neighbors*, then it tends to be difficult to correctly label this instance. Similar to (Smith et al., 2014; Yan et al., 2020), we used the metric *k-Disagreeing Neighbors* (kDN) to measure the local overlap of a data instance x with its k-nearest neighbors (identified by calculating the Euclidean distance between their vanilla BERT-generated representations) regarding their task labels. We chose $k = 5$ to calculate the kDN of an instance (as suggested in (Yan et al., 2020)). A large kDN (close to 1) indicates that the data instance is difficult to be correctly classified. If the kDN distribution of a student group (e.g., female) is larger than that of the other group (e.g., male), there exists Hardness bias (denoted as H) between the two groups, which can be calculated by applying the Jensen-Shannon (JS) distance, as defined below:

$$H(y) = JS(\{f(x, y) \mid G = 1\} - \{f(x, y) \mid G = 0\}) \quad (2)$$

where $G \in \{0, 1\}$, $Y \in \{0, 1\}$, and $f(x, y)$ is the kDN distribution of data instances with $G = g$ and $Y = y$. To keep the Hardness bias between two protected groups minimal, we constructed the fair seed set by adding the instances in an one-by-one manner, and a data instance could only be included if the overall Hardness bias of the fair seed set was decreasing after including the instance.

Step 2: Query Strategy Selection (i.e., ② depicted in Figure 1). With the seed set constructed, we further enriched it with more fair instances by calculating the instances' informativeness with respect to students' protected attributes. Specifically, we adopt three representative query strategies in the AL research to enrich the seed set, which all support the selection of multiple instances at a time and have been proven to be effective in assisting machine learning models to achieve better prediction performance. All the machine learning models used by these strategies were built based on the seed set constructed above and took the vanilla BERT-learned representation of a post as input to predict the protected attribute of the post creator.

- `Query By Committee (QBC)` (Dagan and Engelson, 1995), which first trained an ensemble of models, i.e., ten logistic regression classifiers in our case. Each of the classifiers was built based on a random subset of the fair seed set. If the ensemble of models could not reach an agreement on the predicted protected attribute, then the post was selected.

- `Learning Active Learning (LAL)` (Konyushkova et al., 2017), which trained a random forest regressor to predict the expected error reduction (derived based on the predicted probabilities of different protected attributes) for all unlabeled samples in Pretraining Data, and select samples with the most error reduction.

- `Least Confident Classification (LCC)` (Settles and Craven, 2008; Bilgic et al., 2010; Tong and Chang, 2001), which trained a classifier based on logistic regression to predict the protected attribute of a post creator and selected samples with the least confidence, i.e., the predicted probabilities should be as close to 0.5 as possible in the binary protected attribute classification problems in our study.

Step 3: Dynamic and Fair Sampling (i.e., ③ depicted in Figure 1). Every time when a query strategy described above is applied, $K\%$ fraction of the Pretraining Data will be sampled and used

to perform additional pretraining to BERT. Instead of setting a large value to K and only perform one-time sampling, which may hinder the identification of effective samples to augment a PLM, we designed a dynamic approach to sample instances for multiple times (i.e., with small values for K) and, more importantly, the number of fair instances specific to a protected group can be determined based on the current prediction bias existed in the downstream task. Let t denote the number of times that BERT has undergone additional pretraining, G_0 and G_1 denote the two protected groups, and the current prediction accuracy in the downstream task (i.e., those with the aid of the latest BERT) for G_0 and G_1 are denoted as acc_0 and acc_1, respectively. Then, for the $(t+1)$-th pretraining, the ratio between the number of sampled instances between G_0 and G_1 is $acc_1 : acc_0$, i.e., the group with a lower prediction accuracy will have more samples. The above sampling process is repeated until it reaches the maximum times (denoted as T_{max}) allowed to performed additional pretraining and then the best-performing BERT in terms of prediction fairness is selected.

3.3 Study Setup

Baselines. In addition to the three variants of the proposed sampling method (denoted as `AL-QBC`, `AL-LAL`, and `AL-LCC`, respectively), we implemented three baselines for comparisons: (i) `w/o Pretraining`, in which LR-Task takes as input the vanilla BERT-learned representations (i.e., without any additional pretraining); (ii) `Random`, in which LR-Task takes as input the learned representations from BERT with randomly-selected samples for additional pretraining; and (iii) `Equal-demographic`, in which LR-Task takes as input the learned representations from BERT with additional pretraining based on data an equal representation of different protected groups (but not considering their representations in terms of different task labels).

Model implementation and training. We used the BERT model provided by *huggingface*[2]. The AL methods were implemented by *alipy*[3] and the logistic regression models by *sklearn*[4]. For the fair data sampling method we proposed, we set the size of the fair seed set (Step 1 of our proposed approach) to be 500, the value of T_{max} to be 6 (i.e., the number of times that BERT could have additional pretraining), and the value of K to be 5% (i.e., the fraction of data to be sampled from the Pretraining Data by an AL strategy every time). To guide the data sampling process of our proposed method (i.e., Step 3), we constructed an independent *reference subset* by using the same method described in Step 1 to sample 500 posts from the Annotated Data, and this subset was used to measure the changing prediction bias existed between different protected groups throughout the whole debiasing process. When training the logistic regression model for predicting students' protected attributes (i.e., LR-Demographic), we randomly split the Demographic Data into training and testing sets in the ratio of 8:2 and the hyper-parameters were determined via 5-fold cross-validation. We used a similar method to construct the logistic regression model for predicting the types of posts (i.e., LR-Task), but based on the remaining 3,203 labeled posts in Annotated Data (after constructing the reference subset described above). Our procedure to continue the pretraining of BERT is in line with similar studies (Devlin et al., 2018; Araci, 2019; Beltagy et al., 2019), with the maximum length of input text as 512, learning rate as 2e-05, batch size as 16, the maximum number of training epochs as 12, and early stopping mechanisms were used.

Evaluation metrics. The capability of BERT in discerning students' protected attributes can be indirectly measured in terms of the prediction accuracy of LR-Demographic, which is measured by using AUC. Recall that we aimed to reduce the demographic information carried in BERT-learned representations; therefore, a lower AUC of LR-demographic indicates better results (i.e., a lower awareness of protected attributes). We also used the AUC to measure the prediction accuracy of LR-Task in distinguishing different types of forum posts, but a higher AUC is preferable for LR-Task. To measure the prediction bias held by LR-Task, we adopted the Absolute Between-ROC Area (ABROCA) metric proposed by (Gardner et al., 2019), which has been widely used to measure algorithmic bias in relevant educational studies (Riazy et al., 2020; Tsai et al., 2020; Paquette et al., 2020). A lower ABROCA indicates better prediction fairness.

[2]https://huggingface.co/
[3]http://parnec.nuaa.edu.cn/
[4]https://scikit-learn.org/

Table 2: Results of LR-Demographic and LR-Task with the aid of different AL strategies. Here, T_{max} denotes the maximum number of times that BERT received additional pretraining. The number inside brackets indicates improvement compared to the results of `w/o pretraining`. LR-demo is short for LR-Demographic. The best result in each metric is in bold. The signs ↑ and ↓ are used to indicate whether a higher (↑) or lower (↓) value is more preferred in a metric.

Row ID	Methods	T_{max}	First-language backgrounds			Gender		
			LR-Demo	LR-Task		LR-Demo	LR-Task	
			↓ AUC	↑ AUC	↓ ABROCA	↓ AUC	↑ AUC	↓ ABROCA
1	w/o pretraining	-	0.686	0.869	0.086	0.591	0.882	0.057
2	Ramdom	1	0.692 (-0.87%)	0.876 (0.81%)	0.098 (-13.95%)	0.611 (-3.38%)	0.892 (1.13%)	0.089 (-56.14%)
3	Equal		0.670 (2.33%)	0.883 (1.66%)	0.079 (8.14%)	0.595 (-0.68%)	0.889 (0.84%)	0.066 (-15.79%)
4	AL-QBC		0.591 (13.85%)	0.879 (1.15%)	0.105 (-22.09%)	0.559 (5.41%)	0.889 (0.77%)	0.059 (-3.51%)
5	AL-LAL		0.589 (14.14%)	0.876 (0.85%)	0.069 (19.77%)	0.552 (6.60%)	0.898 (1.85%)	0.055 (3.51%)
6	AL-LCC		0.573 (16.47%)	0.878 (1.01%)	0.055 (36.05%)	0.558 (5.58%)	0.891 (1.02%)	0.047 (17.54%)
7	Ramdom	6	0.688 (-0.29%)	0.889 (2.30%)	0.112 (-30.23%)	0.588 (0.51%)	0.895 (1.47%)	0.072 (-26.32%)
8	Equal		0.621 (9.48%)	0.889 (2.30%)	0.095 (-10.47%)	0.561 (5.08%)	0.889 (0.84%)	0.066 (-15.79%)
9	AL-LCC		**0.525 (23.47%)**	**0.891 (2.53%)**	**0.041 (52.33%)**	**0.534 (9.64%)**	**0.899 (1.96%)**	**0.031 (45.61%)**

4 Results

Results on RQ1. The results of LR-demographic are given in Table 2 (Row 1). For both of the two protected attributes, LR-Demographic achieved an AUC score larger than 0.5, which implies that the representations of forum posts learned by BERT did carry informative features that can be utilized by machine learning models to reveal the demographic attributes of the students. In particular, the AUC value of first-language backgrounds (0.686) is much higher than that of gender (0.591). A possible explanation is that, students in the groups of different first-language backgrounds, compared to the groups of a different gender, tend to display more linguistic differences in their forum posts, which can be captured by BERT and used to reveal their demographic attributes.

Results on RQ2. To answer RQ2, we implemented the three AL strategies described in Section 3.2. To compare their effectiveness in inhibiting BERT's awareness of protected attributes, we first set T_{max} as 1, i.e., only performing one-time additional pretraining, and the results are given in Table 2 (Row 2-6, Column LR-Demo). Based on the results, we can observe that, by simply using an equal number of instances generated by different protected groups for additional pretraining (i.e., the baseline `Equal`), it cannot guarantee that BERT's awareness of protected attributes can be reduced. However, all of the AL strategies proposed in this work showed effectiveness in removing demographic information embedded in the vanilla BERT-learned representations, especially when dealing with the first-language groups (with an improvement of 13.85%~16.47%), where vanilla BERT demonstrated a higher awareness of students' demographic attributes. Notice that, for both of the two protected attributes, the best results were delivered by `AL-LCC`. Then, we re-ran evaluation of `AL-LCC` and increased the value of T_{max} to 6, and also compared the results to those of `Random` and `Equal` with the same T_{max} value (Row 7-9 in Table 2). Here, the AUC of LR-Demo was even reduced to very close to 0.5 (i.e., 0.525) in the first-language groups and 0.534 in the gender groups, which provides strong evidence of the effectiveness of our fair sampling method in stopping BERT to record demographic attributes into its learned representations.

Results on RQ3. To measure the impact of BERT's reduced awareness of protected attributes on the prediction fairness in the downstream task, we measured the performance of LR-Task which took as input the representations learned by the vanilla BERT or BERT with additional pretraining. The results are also given in Table 2, based on which we can have observations similar to those made when analyzing the AUC of LR-Demographic. Firstly, in the results of `Random` and `Equal` (Row 2-3 and Row 7-8), which randomly sampled data or only maintained a balance in terms of protected attributes in the selected data, the prediction bias tended to be exacerbated and the prediction fairness could be worsen up to 30.23%. Secondly, among the three AL strategies, `AL-LCC` outperformed the other two in enhancing prediction fairness, even when there was only one-time additional pretraining (i.e., $T_{max} = 1$, Row 6 in Table 2), and the

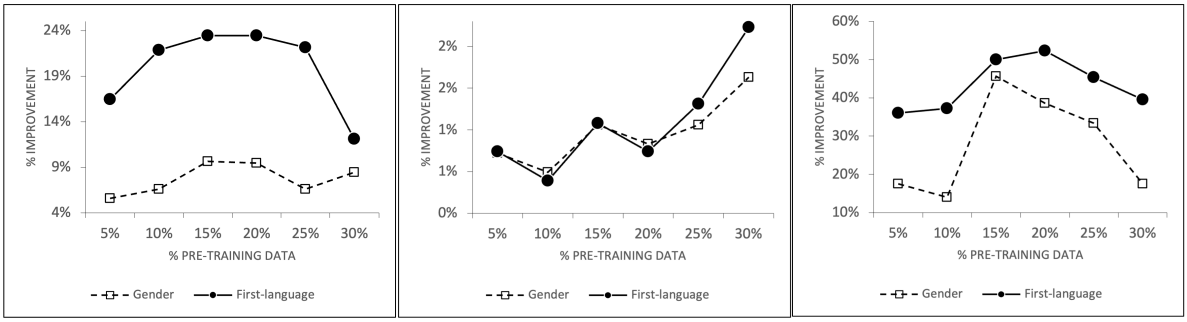

(a) LR-Demo (AUC) (b) LR-Task (AUC) (c) LR-Task (ABROCA)

Figure 2: The relative improvements of AL−LCC compared to w/o Pretraining from $T_{max} = 1$ to $T_{max} = 6$.

ABROCA improvement was up to 36.05% for first-language groups and 17.54% for gender groups. When T_{max} was increased to 6 (Row 9 in Table 2), the improvement was further boosted to 52.33% and 45.61% for the two groupings. Thirdly, in addition to the increased prediction fairness, we observe that the prediction accuracy was boosted with slight improvements of 2.53% and 1.96% for the two groupings (Row 9 in Table 2), respectively, compared to those of Random (2.30% and 1.47%). This further demonstrates that, by carefully selecting the data used to perform additional pretraining, both the prediction accuracy and fairness can be simultaneously enhanced.

To investigate the amount of data needed to reach the maximum prediction fairness, we plotted the relative improvement achieved by AL−LCC compared to the baseline w/o Pretraining across the whole additional pretraining process (Figure 2). We found that, with the aid of AL−LCC, LR-Demographic achieved the best AUC when using 15%~20% of the available unlabeled data (i.e., $T_{max} = 3$ or $T_{max} = 4$). This implies that, instead of using all the available data, it can be more effective and efficient by carefully selecting a subset of them for additional pretraining.

5 Discussion and Conclusion

To debias BERT by directly reducing the information related to protected attributes in the learned representations, this study developed a dynamic and fair sampling method to select data to perform additional pretraining to BERT, which is capable of significantly inhibiting BERT's awareness of protected attributes and subsequently improved both the prediction fairness and accuracy in the downstream task. Here, we further elaborated on our study's practical insights and implications and discussed the limitations to be addressed in the future.

Implications. Firstly and most importantly, our study corroborated the findings of previous studies (de Vassimon Manela et al., 2021; Minot et al., 2021), i.e., prediction fairness in a downstream task can be greatly enhanced by reducing a PLM's awareness of sensitive protected attributes, e.g., the amount of information related to such protected attributes in the learned representations. Second, as demonstrated in Table 2, only two out of the three AL strategies used in this study could enhance the prediction fairness in downstream tasks, suggesting that it is of extreme importance to select the appropriate measure for sample informativeness in terms of protected attributes. Third, our study demonstrated that, by carefully selecting fair samples to further pretrain a PLM, even only with 15%~20% of the available unlabeled data, not only the prediction fairness but also the prediction accuracy can be enhanced. This implies that prediction accuracy can benefit from keeping prediction fairness as part of the goal when performing additional pretraining to a PLM.

Limitations. Firstly, the effectiveness of the proposed fair data sampling method was only validated based on BERT and one dataset in the field of education. Future studies are needed to replicate the study using other PLMs or datasets to further validate the presented findings. Secondly, we focused on debiasing BERT in terms of two protected attributes (i.e., first-language backgrounds and gender) separately. Future work may further investigate methods to debias a PLM by considering other types of protected attributes or simultaneously taking multiple of them into consideration. Lastly, we only experimented with a limited set of values for the parameters used in the proposed method, e.g., $K = 5\%$ (the fraction of available unlabeled data

to be sampled for additional pretraining). In the future, it would be worthwhile to develop methods to automatically determine the best values for such parameters.

References

Omaima Almatrafi, Aditya Johri, and Huzefa Rangwala. 2018. Needle in a haystack: Identifying learner posts that require urgent response in mooc discussion forums. *Computers & Education*, 118:1–9.

Laila Alrajhi, Khulood Alharbi, and Alexandra I Cristea. 2020. A multidimensional deep learner model of urgent instructor intervention need in mooc forum posts. In *International conference on intelligent tutoring systems*, pages 226–236. Springer.

Hadis Anahideh, Abolfazl Asudeh, and Saravanan Thirumuruganathan. 2022. Fair active learning. *Expert Systems with Applications*, 199:116981.

Dogu Araci. 2019. Finbert: Financial sentiment analysis with pre-trained language models. *arXiv preprint arXiv:1908.10063*.

Aneesha Bakharia. 2016. Towards cross-domain mooc forum post classification. In *Proceedings of the Third (2016) ACM Conference on Learning@ Scale*, pages 253–256.

Iz Beltagy, Kyle Lo, and Arman Cohan. 2019. Scibert: A pretrained language model for scientific text. *arXiv preprint arXiv:1903.10676*.

Rishabh Bhardwaj, Navonil Majumder, and Soujanya Poria. 2021. Investigating gender bias in bert. *Cognitive Computation*, 13(4):1008–1018.

Mustafa Bilgic, Lilyana Mihalkova, and Lise Getoor. 2010. Active learning for networked data. In *ICML*.

Nicola Capuano, Santi Caballé, Jordi Conesa, and Antonio Greco. 2021. Attention-based hierarchical recurrent neural networks for mooc forum posts analysis. *Journal of Ambient Intelligence and Humanized Computing*, 12(11):9977–9989.

Rui Castro and Robert Nowak. 2008. Active learning and sampling. In *Foundations and Applications of Sensor Management*, pages 177–200. Springer.

Ilias Chalkidis, Manos Fergadiotis, Prodromos Malakasiotis, Nikolaos Aletras, and Ion Androutsopoulos. 2020. Legal-bert: The muppets straight out of law school. *arXiv preprint arXiv:2010.02559*.

Ying-Hong Chan and Yao-Chung Fan. 2019. A recurrent bert-based model for question generation. In *Proceedings of the 2nd Workshop on Machine Reading for Question Answering*, pages 154–162.

Benjamin Clavié and Kobi Gal. 2019. Edubert: Pretrained deep language models for learning analytics. *arXiv preprint arXiv:1912.00690*.

Ido Dagan and Sean P Engelson. 1995. Committee-based sampling for training probabilistic classifiers. In *Machine Learning Proceedings 1995*, pages 150–157. Elsevier.

Daniel de Vassimon Manela, David Errington, Thomas Fisher, Boris van Breugel, and Pasquale Minervini. 2021. Stereotype and skew: Quantifying gender bias in pre-trained and fine-tuned language models. In *EACL 2021-16th Conference of the European Chapter of the Association for Computational Linguistics, Proceedings of the Conference*, pages 2232–2242. Association for Computational Linguistics.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*.

Pratik Gajane and Mykola Pechenizkiy. 2017. On formalizing fairness in prediction with machine learning. *arXiv preprint arXiv:1710.03184*.

Josh Gardner, Christopher Brooks, and Ryan Baker. 2019. Evaluating the fairness of predictive student models through slicing analysis. In *Proceedings of the 9th international conference on learning analytics & knowledge*, pages 225–234.

Shay A Geller, Nicholas Hoernle, Kobi Gal, Avi Segal, Hyunsoo Gloria Kim, David Karger, Marc Facciotti, Kamali Sripathi, and Michele Igo. 2021. New methods for confusion detection in course forums: Student, teacher and machine. *IEEE Transactions on Learning Technologies*.

Matan Halevy, Camille Harris, Amy Bruckman, Diyi Yang, and Ayanna Howard. 2021. Mitigating racial biases in toxic language detection with an equity-based ensemble framework. In *Equity and Access in Algorithms, Mechanisms, and Optimization*, pages 1–11.

Rashidul Islam, Kamrun Naher Keya, Ziqian Zeng, Shimei Pan, and James Foulds. 2021. Debiasing career recommendations with neural fair collaborative filtering. In *Proceedings of the Web Conference 2021*, pages 3779–3790.

Xisen Jin, Francesco Barbieri, Brendan Kennedy, Aida Mostafazadeh Davani, Leonardo Neves, and Xiang Ren. 2020. On transferability of bias mitigation effects in language model fine-tuning. *arXiv preprint arXiv:2010.12864*.

Ksenia Konyushkova, Raphael Sznitman, and Pascal Fua. 2017. Learning active learning from data. *Advances in neural information processing systems*, 30.

Liangyou Li, Xin Jiang, and Qun Liu. 2019. Pretrained language models for document-level neural machine translation. *arXiv preprint arXiv:1911.03110*.

Jionghao Lin, Mladen Rakovic, David Lang, Dragan Gasevic, and Guanliang Chen. 2022. Exploring the

politeness of instructional strategies from human-human online tutoring dialogues. In *LAK22: 12th International Learning Analytics and Knowledge Conference*, pages 282–293.

Anastassia Loukina, Nitin Madnani, and Klaus Zechner. 2019. The many dimensions of algorithmic fairness in educational applications. In *Proceedings of the Fourteenth Workshop on Innovative Use of NLP for Building Educational Applications*, pages 1–10, Florence, Italy. Association for Computational Linguistics.

Kaiji Lu, Piotr Mardziel, Fangjing Wu, Preetam Amancharla, and Anupam Datta. 2020. Gender bias in neural natural language processing. In *Logic, Language, and Security*, pages 189–202. Springer.

Owen HT Lu, Anna YQ Huang, Danny CL Tsai, and Stephen JH Yang. 2021. Expert-authored and machine-generated short-answer questions for assessing students learning performance. *Educational Technology & Society*, 24(3):159–173.

Li Lucy and David Bamman. 2021. Gender and representation bias in gpt-3 generated stories. In *Proceedings of the Third Workshop on Narrative Understanding*, pages 48–55.

Joshua R Minot, Nicholas Cheney, Marc Maier, Danne C Elbers, Christopher M Danforth, and Peter Sheridan Dodds. 2021. Interpretable bias mitigation for textual data: Reducing gender bias in patient notes while maintaining classification performance. *arXiv preprint arXiv:2103.05841*.

Marzieh Mozafari, Reza Farahbakhsh, and Noël Crespi. 2020. Hate speech detection and racial bias mitigation in social media based on bert model. *PloS one*, 15(8):e0237861.

Moin Nadeem, Anna Bethke, and Siva Reddy. 2020. Stereoset: Measuring stereotypical bias in pretrained language models. *arXiv preprint arXiv:2004.09456*.

Anastasios Ntourmas, Sophia Daskalaki, Yannis Dimitriadis, and Nikolaos Avouris. 2021. Classifying mooc forum posts using corpora semantic similarities: a study on transferability across different courses. *Neural Computing and Applications*, pages 1–15.

Christopher M Ormerod, Akanksha Malhotra, and Amir Jafari. 2021. Automated essay scoring using efficient transformer-based language models. *arXiv preprint arXiv:2102.13136*.

Luc Paquette, Jaclyn Ocumpaugh, Ziyue Li, Alexandra Andres, and Ryan Baker. 2020. Who's learning? using demographics in edm research. *Journal of Educational Data Mining*, 12(3):1–30.

Flavien Prost, Nithum Thain, and Tolga Bolukbasi. 2019. Debiasing embeddings for reduced gender bias in text classification. *arXiv preprint arXiv:1908.02810*.

Yada Pruksachatkun, Satyapriya Krishna, Jwala Dhamala, Rahul Gupta, and Kai-Wei Chang. 2021. Does robustness improve fairness? approaching fairness with word substitution robustness methods for text classification. *arXiv preprint arXiv:2106.10826*.

Shirin Riazy, Katharina Simbeck, and Vanessa Schreck. 2020. Fairness in learning analytics: Student at-risk prediction in virtual learning environments. In *CSEDU (1)*, pages 15–25.

Burr Settles and Mark Craven. 2008. An analysis of active learning strategies for sequence labeling tasks. In *proceedings of the 2008 conference on empirical methods in natural language processing*, pages 1070–1079.

Jia Tracy Shen, Michiharu Yamashita, Ethan Prihar, Neil Heffernan, Xintao Wu, Sean McGrew, and Dongwon Lee. 2021. Classifying math knowledge components via task-adaptive pre-trained bert. In *International Conference on Artificial Intelligence in Education*, pages 408–419. Springer.

Andrew Silva, Pradyumna Tambwekar, and Matthew Gombolay. 2021. Towards a comprehensive understanding and accurate evaluation of societal biases in pre-trained transformers. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 2383–2389.

Michael R Smith, Tony Martinez, and Christophe Giraud-Carrier. 2014. An instance level analysis of data complexity. *Machine learning*, 95(2):225–256.

Simon Tong and Edward Chang. 2001. Support vector machine active learning for image retrieval. In *Proceedings of the ninth ACM international conference on Multimedia*, pages 107–118.

Samia Touileb, Lilja Øvrelid, and Erik Velldal. 2021. Using gender-and polarity-informed models to investigate bias. In *Proceedings of the 3rd Workshop on Gender Bias in Natural Language Processing*, pages 66–74.

Yi-Shan Tsai, Carlo Perrotta, and Dragan Gašević. 2020. Empowering learners with personalised learning approaches? agency, equity and transparency in the context of learning analytics. *Assessment & Evaluation in Higher Education*, 45(4):554–567.

Xiaocong Wei, Hongfei Lin, Liang Yang, and Yuhai Yu. 2017. A convolution-lstm-based deep neural network for cross-domain mooc forum post classification. *Information*, 8(3):92.

Shen Yan, Hsien-te Kao, and Emilio Ferrara. 2020. Fair class balancing: Enhancing model fairness without observing sensitive attributes. In *Proceedings of the 29th ACM International Conference on Information & Knowledge Management*, pages 1715–1724.

Wonjin Yoon, Jinhyuk Lee, Donghyeon Kim, Minbyul Jeong, and Jaewoo Kang. 2019. Pre-trained language model for biomedical question answering. In *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*, pages 727–740. Springer.

Haoyu Zhang, Jianjun Xu, and Ji Wang. 2019. Pretraining-based natural language generation for text summarization. *arXiv preprint arXiv:1902.09243*.

Debiasing Word Embeddings with Nonlinear Geometry

Lu Cheng
Department of Computer Science,
University of Illinois Chicago
lucheng@uic.edu
Nayoung Kim and **Huan Liu**
School of Computing and Augmented Intelligence
Arizona State University
{nkim48,huanliu}@asu.edu

Abstract

Debiasing word embeddings has been largely limited to individual and independent social categories. However, real-world corpora typically present multiple social categories that possibly correlate or intersect with each other. For instance, "hair weaves" is stereotypically associated with African American females, but neither African American nor females alone. Therefore, this work studies biases associated with multiple social categories: *joint biases* induced by the union of different categories and *intersectional biases* that do not overlap with the biases of the constituent categories. We first empirically observe that individual biases intersect *non-trivially* (i.e., over a one-dimensional subspace). Drawing from the intersectional theory in social science and the linguistic theory, we then construct an intersectional subspace to debias for multiple social categories using the nonlinear geometry of individual biases. Empirical evaluations corroborate the efficacy of our approach[1].

1 Introduction

Due to the reliance on the large-scale text corpora for training, it has been observed that word embeddings are prone to express social biases inherent in the data (Bolukbasi et al., 2016; Caliskan et al., 2017). Prior research (e.g. Zhao et al., 2019; Bolukbasi et al., 2016) in debiasing word embeddings mitigates biases associated with individual social categories and treats each category in isolation. For example, the seminal Hard-Debiasing approach (Bolukbasi et al., 2016) identifies the bias direction of a category (e.g., gender) and then removes the direction from the target word such that it is equidistant to all groups (e.g., female and male at a binary level with gender) in the category. However, real-world training corpora typically present multiple social categories (e.g., gender and race), possibly with higher cardinality. These social categories can further correlate or intersect with each other (Thomas, 2004; Hancock, 2007). Despite the promising results, debiasing for individual social categories limits our understanding of the complex nature of social biases.

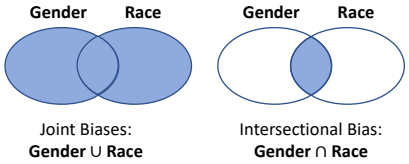

Figure 1: Illustrations of *joint* (i.e., union) and *intersectional* (i.e., intersect) biases using Gender and Race.

Alternatively, we might consider biases in the presence of multiple social categories (Foulds et al., 2020; Kearns et al., 2018; Cheng et al., 2022b,a). This can result in at least two scenarios, as depicted in Fig. 1. First, word embeddings can simultaneously present multiple biases that might *non-trivially correlate* with each other. For example, debiasing for gender can influence the results of racial bias. In the literature of social psychology, a number of works (e.g. Akrami et al., 2011; Bierly, 1985; Allport et al., 1954) studied the interrelationship between various biases, a.k.a. "generalized prejudice". We refer to bias induced by the **union** of different social categories as **joint biases**. Second, a few recent works (e.g. Guo and Caliskan, 2020) detected **intersectional biases** in word embeddings, which is the bias that does not overlap with the biases of their constituent identities. For example, "hair weaves" is stereotypically associated with African American females (Ghavami and Peplau, 2013).

The primary goal of this work is to mitigate the two kinds of biases in word embeddings. There are several challenges: First, it is highly possible that different biases are nonlinearly correlated (Cheng

[1]Data and implementation code can be downloaded at https://github.com/GitHubLuCheng/Implementation-of-JoSEC-COLING-22.

Proceedings of the 29th International Conference on Computational Linguistics, pages 1286–1298
October 12–17, 2022.

et al., 2022a). Simply taking a linear combination of individual bias subspaces (e.g., sum or mean of all bias subspaces) might lead to ineffective solutions that even amplify individual biases. Second, identifying a bias subspace typically needs predefined word sets related to target groups. These sets are curated by experts to most accurately represent each group and the associated social biases. This can be time-consuming and requires great human effort. Existing sets for identifying intersectional groups are extremely limited. They are small and exclusively used for the intersectionality of gender and race (Guo and Caliskan, 2020; Tan and Celis, 2019). Therefore, relying on predefined sets limits the use of debiasing approaches in practice.

To address these challenges, we first empirically observe that individual bias subspaces constructed via existing word sets intersect over a one-dimensional subspace. We then relate our findings to the intersectionality theory by Crenshaw (1989) and the linguistic theory introduced by Firth (1957). The result is a **hypothesis** that the bias subspace for multiple social categories roughly resides in the intersection of all individual bias subspaces. The proposed approach (coined as Joint and Intersectional Debiasing, JoSEC) departs from the linear correlation assumption and leverages the nonlinear geometry of subspace representations to learn an intersectional subspace. JoSEC does not need any additional human-coded defining sets for intersectional groups except for the defining sets for the constituent groups. We contribute to:

- a novel problem that considers biases associated with multiple social categories in mitigation;
- an effective approach for constructing the bias subspace without relying on the defining sets for intersectional categories; and
- demonstrations of the effectiveness of JoSEC via empirical evaluations on the benchmark datasets.

2 Related Work

Early efforts in debiasing word embeddings have been focused on gender. The seminal work by Bolukbasi et al. (2016) proposed a post-processing approach that projects gender-neutral words into a gender subspace identified by defining sets of gendered words such as *she, he, man, woman*. Gender bias can then be alleviated through hard-debiasing in which the bias components in non-gendered words (e.g., *doctor, nurse*) are first removed and the gendered words are then centered and equalized. Manzini et al. (2019) further extended the hard-debiasing method to multi-class settings such as race. Given a corpus, one can also learn gender-neutral word embeddings by modifying the GloVe (Pennington et al., 2014) objective function (Zhao et al., 2018). This in-processing approach was further extended to a post-processing approach by Kaneko and Bollegala (2020), who suggested preserving gender-related information using autoencoder. Previous research as well as our work focuses on static word embeddings, however, stereotyped biases have also been found in contextualized word embeddings, e.g., (Zhao et al., 2019; Bordia and Bowman, 2019).

Despite the fruitful results, most prior works were found to remove biases superficially and fail to deliver gender-neutral embeddings (Gonen and Goldberg, 2019; Blodgett et al., 2020). Therefore, it is increasingly observable that existing bias removal techniques are insufficient to guarantee gender-neural modeling. The majority of existing works were also criticized for not examining the impact of gender bias in real-world applications (Blodgett et al., 2020). In contrast to prior research focused on one form of bias in debiasing word embeddings, this work aims to provide a simple yet effective approach for bias mitigation in the presence of multiple bias forms. As human-like biases exist in the majority of word embeddings and debiasing approaches are unlikely to largely affect our results, we build our approach upon the seminal Hard-Debiasing algorithm by Bolukbasi et al. (2016). Future research is warranted to investigate other debiasing approaches.

For intersectional bias, most of the existing works are in social science and psychology literature, such as (Crenshaw, 1989; Kahn and Yoder, 1989; Hare-Mustin and Marecek, 1988). Comparatively fewer efforts can be found in the computer science field. One such work (Buolamwini and Gebru, 2018) examined the intersectional accuracy disparities in commercial gender classification systems. For contextualized word embeddings, May et al. (2019) and Tan and Celis (2019) measured the emergent intersectional biases of African American females using attributes presented in (Caliskan et al., 2017). In complement to WEAT, Guo and Caliskan (2020) proposed the Contextualized Embedding Association Test (CEAT) to measure the

intersectional bias. These methods detect intersectional bias using the defining and attribute sets related to intersectional groups.

In summary, this work complements prior research by mitigating biases related to multiple social categories. JoSEC does not rely on the human-coded word sets used to define the intersectional groups to identify the bias subspace. While this work focuses on non-contextualized embeddings, research (e.g., Guo and Caliskan, 2020; Lepori, 2020) has found contextualized word embeddings such as BERT (Devlin et al., 2019) display intersectional biases. We leave it for future exploration.

3 Preliminary

In this section, we briefly review the backbone model of this work: the seminal hard-debiasing method (Bolukbasi et al., 2016) and its extension. It consists of two steps: identifying the bias subspace and removing bias components.

3.1 Identifying Bias Subspace

The individual bias subspace is identified by the *defining sets*, in which words represent different ends of the bias. For example, the defining sets of gender can be the gendered pronouns *{he, she}* and nouns *{man, woman}*. One can then identify the gender subspace $\mathcal{B}$ by (1) subtracting the word embeddings of words in each defining set from the set's mean, and (2) obtaining the K most significant components of the resulting vectors through a dimensionality-reduction method.

3.2 Removing Bias Components

The next step is to apply the hard debiasing strategy to completely or partially remove the subspace components from the word embeddings. Hard-debiasing consists of two steps – *Neutralize* and *Equalize*. "Neutralize" removes bias components from non-gendered words (e.g., *doctor* and *nurse*); "Equalize" aims to center the gendered word embeddings (e.g., *she* and *he*) and equalize their bias components, such as the word pair *{man, woman}*.

Formally, given a bias subspace $\mathcal{B} = \{\boldsymbol{b_1}, \boldsymbol{b_2}, ..., \boldsymbol{b_K}\}$, where K denotes the number of principal components, we first compute the bias component $\mathbf{w}_\mathcal{B}$ of embedding $\mathbf{w} \in \mathbb{R}^d$ in $\mathcal{B}$ by

$$\mathbf{w}_\mathcal{B} = \sum_{k=1}^{K} \langle \mathbf{w}, \boldsymbol{b_k} \rangle \boldsymbol{b_k}. \tag{1}$$

We then *neutralize* word embeddings by removing the bias component from non-gendered words:

$$\mathbf{w}' = \frac{\mathbf{w} - \mathbf{w}_\mathcal{B}}{\|\mathbf{w} - \mathbf{w}_\mathcal{B}\|}, \tag{2}$$

where $\mathbf{w}'$ are the debiased word embeddings.

To "Equalize", we debias the gendered words in a given equality set E by the following equation:

$$\mathbf{w}' = (\boldsymbol{\mu} - \boldsymbol{\mu}_\mathcal{B}) + \sqrt{1 - \|\boldsymbol{\mu} - \boldsymbol{\mu}_\mathcal{B}\|^2} \frac{\mathbf{w}_\mathcal{B} - \boldsymbol{\mu}_\mathcal{B}}{\|\mathbf{w}_\mathcal{B} - \boldsymbol{\mu}_\mathcal{B}\|}, \tag{3}$$

where $\boldsymbol{\mu} = \frac{1}{|E|} \sum_{\mathbf{w} \in E} \mathbf{w}$ is the mean of embeddings of the words in the equality set E. $\boldsymbol{\mu}_\mathcal{B}$ denotes the bias component of $\boldsymbol{\mu}$ in the identified bias subspace. It can be obtained via Eq. 1.

3.3 Extending into Multi-Class Settings

In a multi-class setting (i.e., with more than two classes in a category, e.g., religion or race), the task inherently becomes non-linearly separable (Manzini et al., 2019). However, it is possible to linearly separate multiple classes based on the components of word embeddings. The multi-class bias subspace is then defined as follows: Given n defining sets of word embeddings $\{D_1, D_2, ..., D_n\}$, the bias subspace $\mathcal{B}$ is defined by the first K components of the following Principal Component Analysis (PCA) (Abdi and Williams, 2010) evaluation:

$$\mathcal{B} = \textbf{PCA}\Big(\bigcup_{i=1}^{n} \bigcup_{\mathbf{w} \in D_i} \mathbf{w} - \boldsymbol{\mu}_i\Big), \tag{4}$$

where $\boldsymbol{\mu}_i = \frac{1}{|D_i|} \sum_{\mathbf{w} \in D_i} \mathbf{w}$ is the mean of word embeddings in set i. $\bigcup$ denotes concatenation by rows. To remove multi-class bias, one can use the hard-debiasing method described in Sec. 3.2.

4 Method

Existing approaches for debiasing word embeddings work on individual categories, rendering incomplete measurement of various social biases (Hancock, 2007; Hurtado and Sinha, 2008). To account for biases associated with multiple social categories, we need to address the primary challenges of the potential non-linear correlations between biases and the difficulty of curating defining sets to identify the bias subspace for these categories. In this section, we introduce the proposed approach – JoSEC – for identifying such a subspace **without additional** human-coded defining set, which we refer to as the *intersectional subspace*.

4.1 Social Categories as "Cultural Contexts"

If individual biases are linearly correlated, we may construct the intersectional subspace by simply taking the sum or average of all individual bias subspaces. However, social biases are complex by nature as suggested by evidence in social science and psychology such as the "generalized prejudice", i.e., generalized devaluing sentiments across different groups. To better quantify the potential nonlinearity, we might first take a step back and revisit the development of single-word embeddings.

An influential position in the development of word embeddings holds that semantic representations for words can be derived through the patterns of lexical co-occurrence in language corpora. This is famously summarized by Firth (1957) as "you shall know a word by the company it keeps". The central tenet is the idea that the sense of the target word could be inferred from its contexts, i.e., neighboring words within the sentence. Informed by this finding, we might assume that human-like biases follow a similar principle: a bias form w.r.t. some social category can be identified by its unique cultural contexts. In debiasing word embeddings, this indicates that the bias subspace (e.g., gender subspace) can be defined by the defining sets of words (e.g., *she, he*) that provide a specific "cultural context" for this bias form.

Naturally, when defining the intersectional subspace associated with multiple social categories, *we might similarly consider each social category as a unique "cultural context" of the intersectional subspace*. For example, the subspace of the intersectional group of gender and race is defined by both the gender subspace and the race subspace. That is, "Aisha" – a common name of an African American female – can be exclusively defined within a cultural context jointly determined by both gender and race whilst terms such as "hair weaves" should not depend on such context. Underpinning this assumption is the idea similar to the linguistic theory: each social category provides *unique context* to construct the intersectional subspace associated with multiple categories.

4.2 Geometry of Subspace Representation

Under the "social categories as cultural contexts" assumption, the intersectional subspace might have a fairly large intersection with each individual bias subspace. We empirically observe and hypothesize that the intersectional subspace of multiple

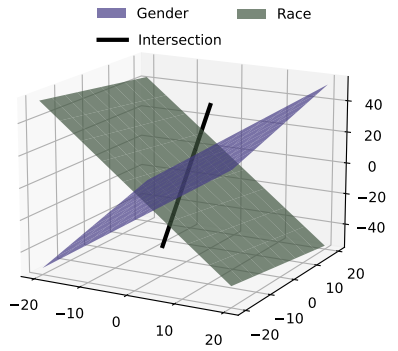

Figure 2: The geometry of Gender and Race bias subspaces and the intersectional subspace (Intersection).

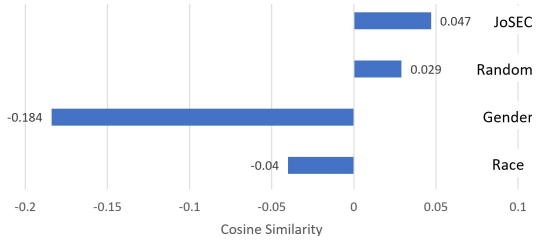

Figure 3: Cosine similarity between the ground-truth intersectional subspace and subspaces obtained via various approaches. "Random" denotes the result averaged over 10 similarity scores from 10 random vectors.

social categories should reside in all the subspaces representing the "cultural contexts" where the intersectional subspace is defined. Specifically, the intersectional subspace resides in the *intersection* of all individual bias subspaces and these subspaces should **intersect non-trivially**. This further implies that there exists a direction (one-dimension subspace) that is extremely close to all individual bias subspaces. We use this vector to represent the intersectional subspace. We propose the following hypothesis for identifying the bias subspace associated with multiple social categories:

Hypothesis (*Intersectional Hypothesis*)**.** *The intersectional subspace $\mathcal{B}_{sec}$ should reside in the intersection of $\{\mathcal{B}_1, \mathcal{B}_2, ...\mathcal{B}_N\}$, where $\mathcal{B}_i$ denotes the bias subspace of social category i and N is the number of considered social categories.*

Intersectional Hypothesis can be seen as operationalizing the intersectionality theory guided by Firth's hypothesis.

4.2.1 Empirical Validation of the Intersectional Hypothesis

We empirically validate the intersectional hypothesis using the benchmark dataset L2-Reddit corpus (detailed in Sec. 5.1). The dataset includes the

defining sets for three social categories: race, gender, and religion, respectively. We first construct three individual bias subspaces for the target categories and each subspace has dimensions $K \times d$, where d denotes the dimension of word embeddings. For better visualization, we further project the d-dimensional subspace representations to 3D vectors using PCA.

We draw subspaces w.r.t. two randomly selected categories as 2-dimensional planes in Fig. 2. Result for the three categories is in Appendix A. We also visualize the corresponding intersectional subspaces identified by Eq. 7. We observe that the individual bias subspaces intersect roughly in a common direction with which the intersectional subspace approximately aligns. In addition, we use the defining sets for the intersectional groups of gender and race (e.g., African American Female) provided by WEAT (Caliskan et al., 2017) and Parada (2016) to construct the "ground-truth" intersectional bias subspace. We then calculate the cosine similarity between the ground-truth subspace and the (a) gender subspace, (b) race subspace, (c) random vector, and (d) intersectional subspace approximated by JoSEC, respectively. The similarity score ranges from -1 to 1, with -1 denoting the most dissimilar. Results in Fig. 3 suggest that JoSEC generates the intersectional subspace significantly more similar to the ground-truth subspace. Note that most of the similarity scores are close to 0. We believe this is in part because of the limitation of existing defining sets for intersectional identities, e.g., small in size. Both quantitative (Fig. 3) and qualitative (Fig. 2) analyses empirically justify the intersectional hypothesis.

4.2.2 Identifying Intersectional Subspace

Under the *Intersectional Hypothesis*, we are essentially seeking the direction vector $\mathcal{B}_{sec} = \hat{\boldsymbol{u}}$ that is "closest" to all individual bias subspaces. Let $\boldsymbol{u}$ be a unit-length vector. We then reduce the problem of identifying intersectional subspace to the following optimization task:

$$\hat{\boldsymbol{u}} = \arg\min_{\|\boldsymbol{u}\|=1} \sum_{i=1}^{N} d(\boldsymbol{u}, \mathcal{B}_i)^2, \tag{5}$$

where $d(\boldsymbol{u}, \mathcal{B}_i)$ is the shortest ℓ_2-distance between the intersectional subspace and the bias subspace of social category $i \in \{1, 2, ..., N\}$. Formally,

$$d(\boldsymbol{u}, \mathcal{B}_i) = \sqrt{\|\boldsymbol{u}\|^2 - \sum_{k=1}^{K} (\boldsymbol{u}^\intercal \boldsymbol{v}_{ik})^2}, \tag{6}$$

where $\{\boldsymbol{v}_{i1}, ..., \boldsymbol{v}_{iK}\}$ are the K principal components representing bias subspace $\mathcal{B}_i$. Eq. 5 can be reformulated as the following:

$$\hat{\boldsymbol{u}} = \arg\max_{\|\boldsymbol{u}\|=1} \sum_{i=1}^{N} \sum_{k=1}^{K} (\boldsymbol{u}^\intercal \boldsymbol{v}_{ik})^2. \tag{7}$$

Eq. 7 can be solved by taking the first principal component of $\{\boldsymbol{v}_{ik}\}_{i=1,...,N;k=1,...,K}$.

With the identified intersectional subspace $\hat{\boldsymbol{u}}$, we then follow Eq. 1-3 to remove the identified intersectional bias components from the target words (e.g., *hair weaves*).

5 Experiments

We validate the efficacy of the proposed intersectional subspace for debiasing word embeddings with multiple social categories. We answer the following research questions: How does JoSEC fare against baselines for mitigating (**RQ. 1**) *joint* biases and (**RQ. 2**) *intersectional* biases? How does JoSEC influence the (**RQ. 3**) utility of word embeddings in downstream tasks and (**RQ. 4**) biases in downstream tasks such as toxicity detection?

5.1 Language Corpus and Social Bias Data

Results for debiasing word embeddings are based on the commonly-used L2-Reddit corpus (Rabinovich et al., 2018), a collection of Reddit[2] posts and comments. It has been shown that the structural factor in user-generated content sites like Reddit make them less welcoming to marginalized populations (Bender et al., 2021). The initial biased word embeddings are obtained by training word2vec on approximately 56 million sentences. It includes three social categories: *Gender, Race*, and *Religion*. We use vocabularies from (Bolukbasi et al., 2016) and (Caliskan et al., 2017) as the defining and attribute sets for gender. Word sets for race and religion are the same lexicons used in (Manzini et al., 2019).

Following (Guo and Caliskan, 2020), we consider the intersectionality of race and gender for the evaluation of intersectional debiasing due to the limited data availability. In particular, there are in total 3 × 2 (3 racial classes and 2 gender classes) intersectional social groups: {*African American male, African American female, European American male, European American female,*

[2]https://www.reddit.com/

Mexican American male, Mexican American female}. The defining sets for these groups are provided by WEAT (Caliskan et al., 2017) and Parada (2016), including frequent given names that represent group membership. The intersectional attribute sets identified through human workers' validation are provided by Ghavami and Peplau (2013). For example, one common given name included in the defining set of African American females is "Aisha" and some related stereotyped bias include attributes "aggressive" and "dark-skinned". For the complete list of given names and attributes, please refer to Appendix B. Note that in contrast to common debiasing approaches, JoSEC does not need these defining sets for joint and intersectional debiasing. We use them for evaluation only.

5.2 Experimental Setup

We briefly summarize the experimental settings, including the baselines, downstream tasks, and evaluation metrics.

5.2.1 Baselines

We are not aware of any existing approaches for joint or intersectional debiasing for word embeddings. Therefore, we adapt the hard-debiasing approach such that it works in both tasks. We also consider two debiasing strategies (i.e., SUM and MEAN) that impose a linear assumption to construct the bias subspace for multiple social categories. All compared approaches only differ in the subspace construction, the bias mitigation follows the same procedure in the hard-debiasing approach. Baselines are detailed below.

- *Hard_Seq*. The hard-debiasing method extended to the *joint* debiasing task. In particular, *Hard_Seq* sequentially debiases for individual social categories. As the order might influence the results of *Hard_Seq*, we experiment with all potential sequences and report the best results.

- *Hard_Insec*. The hard-debiasing method extended to the *intersectional* debiasing task. In particular, *Hard_Insec* uses the human-coded defining sets for the intersectionality groups of gender and race in (Guo and Caliskan, 2020) to construct the intersectional subspace.

- *SUM*. Its bias subspace is constructed by summing up the subspaces of individual biases, i.e., $\mathcal{B}_{SUM} = \sum_{i=1}^{N} \mathcal{B}_i$.

- *MEAN*. Its bias subspace is constructed by averaging over the subspaces of individual biases, i.e., $\mathcal{B}_{MEAN} = \frac{1}{N} \sum_{i=1}^{N} \mathcal{B}_i$.

5.2.2 Downstream Tasks

Utility (RQ. 3): To examine the influence of JoSEC on the utility of word embeddings, we perform several standard downstream tasks following (Manzini et al., 2019). They are the CoNLL 2003 shared tasks (Sang and De Meulder, 2003), including NER tagging, POS (part-of-speech) tagging, and POS chunking. There are two evaluation paradigms: replacing the biased embeddings with the debiased ones or retraining the model on debiased embeddings. We only report results for one setting and the other can be found in Appendix C.
Extrinsic Bias (RQ. 4): While we work on mitigating biases in the pre-trained resource, i.e., *intrinsic bias*, recent research (e.g., Seraphina et al., 2021; Delobelle et al., 2021) presents interesting findings about the biases in downstream tasks enabled by word embeddings, i.e., *extrinsic bias*. Therefore, we further investigate how the debiased word embeddings influence biases in a common downstream NLP task, toxicity detection. Particularly, we consider the Kaggle Challenge of Jigsaw Unintended Bias in Toxicity Classification[3] and examine biases against gender, race, and religion. The Perspective API's Jigsaw dataset has both toxicity and identity annotations. The training and test splits are the same as the original data. Please refer to Appendix D for detailed experimental settings for toxicity classification.

5.2.3 Evaluation Metrics

For debiasing tasks, we use the mean average cosine similarity (MAC) to quantify the intrinsic bias, as suggested in (Manzini et al., 2019). Given a set of target word embeddings $\mathcal{S}$ of words with a specific form of social bias (e.g., *Jew, Christian, Muslim*) and a set of attribute sets $\mathcal{A} = \{A_1, A_2, ..., A_N\}$, A_j consists of embeddings of words $\boldsymbol{a}$ (e.g., *violent, terrorist, uneducated*) that should not be associated with any word in $\mathcal{S}$. Let $f(\cdot)$ be a function that computes the mean cosine distance between $S_i \in \mathcal{S}$ and $\boldsymbol{a} \in A_j$:

$$f(S_i, A_j) = \frac{1}{|A_j|} \sum_{\boldsymbol{a} \in A_j} \cos(S_i, \boldsymbol{a}), \tag{8}$$

[3] https://www.kaggle.com/c/jigsaw-unintended-bias-in-toxicity-classification

Table 1: MACs (↑) of all approaches for joint debiasing. "Re", "Ra", and "Ge" denotes "Religion", "Race", and "Gender", respectively. "Re→Ra→Ge" denotes the order for sequential debiasing. We experiment with all potential sequences and report the best results. A larger value is more desired.

Model \ MAC	Gender	Race	Religion	Total
Biased	0.623	0.892	0.859	2.374
Hard_Seq (Re→Ra→Ge)	0.656	0.888	**0.937**	2.481
Hard_Seq (Ra→Re→Ge)	0.654	**0.929**	0.868	2.451
SUM	0.598	0.870	0.900	2.368
MEAN	0.657	0.872	0.862	2.391
JoSEC	**0.703**	0.914	0.917	**2.534**

where $\cos(S_i, \boldsymbol{a}) = 1 - \frac{S_i \cdot \boldsymbol{a}}{\|S_i\|_2 \cdot \|\boldsymbol{a}\|_2}$. MAC is then computed by

$$\text{MAC}(\mathcal{S}, \mathcal{A}) = \frac{1}{|\mathcal{S}||\mathcal{A}|} \sum_{S_i \in \mathcal{S}} \sum_{A_j \in \mathcal{A}} f(S_i, A_j). \quad (9)$$

A larger MAC score denotes a greater bias removal.

For the downstream task that examines the utility, we report F1 scores from using biased word embeddings as well as the changes of F1 (Δ F1), Precision (Δ Precision), and Recall (Δ Recall) after using debiased word embeddings. To check the statistical significance, we also perform a paired t-test on the distribution of average cosine distance used to compute MAC and student t-test for the results of downstream tasks. For the downstream task that examines the extrinsic bias, we use two common evaluation metrics: False Positive Equality Difference (FPED) and False Negative Equality Difference (FNED). We report Total (FPED+FNED) scores from using biased word embeddings and Δ Total, Δ FPED, and Δ FNED after using debiased word embeddings. Unless otherwise noted, all the results below are statistically significant at level 0.05.

5.3 Results

We first present results (averaged over 5 repetitions) for **RQ. 1-4** and then discuss our findings.

5.3.1 RQ. 1: Joint Debiasing

The joint debiasing task seeks to simultaneously mitigate biases induced by the union of all social categories. We report MACs w.r.t. each individual bias as well as the total bias, which is computed over all considered social categories. All the best results are highlighted.

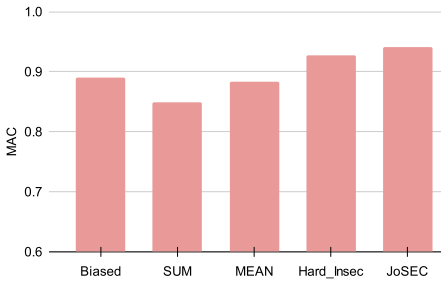

Figure 4: MACs (↑) of all compared approaches for the intersectional debiasing task. A larger value is more desired.

From the results in Table 1, we can observe that (1) the best results w.r.t. debiasing for individual categories are achieved by various approaches and the proposed approach (*JoSEC*) outperforms all baselines regarding reducing the total amount of bias (Total). This suggests that it is challenging to debias for all social categories simultaneously and the proposed intersectional subspace is effective for joint debiasing. (2) The linear solutions to subspace construction (i.e., *SUM* and *MEAN*) are not as effective as sequential debiasing. This empirically validates our hypothesis that social biases are non-linearly correlated and multiple social categories should intersect non-trivially.

(3) Of particular interest is that when debiasing sequentially, the bias mitigation performance w.r.t. the first category appears to be the most effective. For example, *Hard_Seq* (Re→Ra→Ge) shows the best MAC of "Religion". Further, by comparing the results in the second row (*Biased*) with those in the third (*Hard_Seq*), the racial MAC of sequential debiasing decreases (i.e., more biases) whilst applying hard-debiasing to racial bias alone can actually lead to higher MACs. These findings might suggest that different biases are *interacting* with each other, i.e., they are correlated. Future research is warranted to examine the bias correlations and their influence on the debiasing approaches.

5.3.2 RQ. 2: Intersectional Debiasing

To evaluate the intersectional bias removal performance of all the compared methods, we use the human-coded defining and attribute sets associated with the intersectionality of gender and race. Note that only *Hard_Insec* used the human-coded defining sets to construct the bias subspace. All other approaches (i.e., *SUM*, *MEAN*, and *JoSEC*) use subspaces of individual biases to construct the subspace for multiple social categories. MACs of all

Table 2: Utility of biased and debiased word embeddings in NER Tagging (NER), POS Tagging (POS-T), and POS Chunking (POS-C) tasks, under the *Embedding Matrix Replacement* paradigm. Word embeddings are debiased by hard-debiasing with different subspaces. Δ denotes the change before and after debiasing.

	Hard_Seq			Hard_Insec			SUM			MEAN			JoSEC		
Tasks	NER	POS-T	POS-C	NER	POS-T	POS-C	NER	POS-T	POS-C	NER	POS-T	POS-C	NER	POS-T	POS-C
Biased F1	0.96	0.99	1.00	0.96	0.99	1.00	0.96	0.99	1.00	0.96	0.99	1.00	0.96	0.99	1.00
Δ F1	-0.01	0.01	0.00	-0.01	0.00	0.00	-0.01	0.01	0.00	-0.01	0.00	0.00	-0.02	0.00	0.00
Δ Precision	-0.01	0.00	0.00	-0.02	0.00	0.00	-0.02	0.00	0.00	-0.02	0.00	0.00	-0.03	0.00	0.00
Δ Recall	-0.01	0.03	0.02	-0.02	0.01	0.02	-0.02	0.02	0.01	-0.02	0.01	0.01	-0.03	0.01	0.01

Table 3: Gender, racial, and religious biases in toxicity classification using biased and debiased word embeddings. Total=FPED+FNED (↓). A smaller bias score denotes less bias. A negative Δ indicates reduced bias.

	Hard_Seq			Hard_Insec			SUM			MEAN			JoSEC		
Bias	Gender	Race	Religion	Gender	Race	Religion	Gender	Race	Religion	Gender	Race	Religion	Gender	Race	Religion
Biased Total	1.27	0.64	0.37	1.27	0.64	0.37	1.27	0.64	0.37	1.27	0.64	0.37	1.27	0.64	0.37
Δ Total	-0.04	0.04	0.05	-0.05	-0.01	0.01	0.00	0.04	-0.01	0.00	-0.01	0.01	-0.05	0.02	0.02
Δ FPED	-0.09	0.03	0.00	-0.05	0.03	-0.02	-0.03	0.03	-0.01	0.06	0.04	0.01	-0.04	0.03	0.00
Δ FNED	0.05	0.01	0.05	0.00	-0.04	0.02	0.03	0.01	0.01	-0.06	-0.04	0.00	0.00	-0.01	0.02

methods are presented in Fig. 4.

We make the following observations: (1) *JoSEC* is most effective for mitigating the intersectional biases w.r.t. gender and race. The fact that it outperforms *Hard_Insec* manifests the potential to leverage the nonlinear geometry of subspace representation to construct the subspace for intersectional bias. This is encouraging as it suggests that we may not need defining sets for intersectional groups, which are often inaccessible and challenging to collect; (2) *SUM* and *MEAN* are outperformed by *Biased*, indicating that simply taking a linear combination of all the individual bias subspaces can aggravate the intersectional biases in word embeddings. The violation of the linear correlation assumption has a negative influence on the performance of hard-debiasing.

5.3.3 RQ. 3: Downstream Utility

This research question aims to investigate the effects of various debiasing strategies on the semantic utility of word embeddings in standard NLP tasks. We consider NER Tagging, POS Tagging, and POS Chunking, following (Manzini et al., 2019).

Results for embedding matrix replacement are in Table 2. We can observe that debiasing word embeddings with multiple social categories only slightly changes the semantic utility. We also perform the student t test and further testify that these differences are statistically insignificant. These results imply that debiasing for multiple categories using the hard-debiasing method does not have a significant influence on the semantic utility of word embeddings. This applies to both joint and intersectional debiasing.

5.3.4 RQ. 4: Bias in Toxicity Classification

This experiment further examines the extrinsic biases of word embeddings, particularly, in toxicity classification. Results are shown in Table 3. We observe that the debiased word embeddings have little influence on reducing biases in the downstream task, indicating no reliable correlation between the *intrinsic bias* and *extrinsic bias*. This aligns well with findings shown in (Seraphina et al., 2021). Aware of the importance of mitigating biases in downstream applications, it is critical to extend this work to joint and intersectional debiasing focused on extrinsic measures of biases in the future.

6 Discussions

This work studies **joint biases** induced by the union of multiple categories and the **intersectional biases** that do not overlap with biases of the constituent categories. Challenges arise from the potential nonlinearity between different biases and the difficulty of curating human-coded word lists for identifying intersectional bias subspace. We first empirically showed that different biases intersect *non-trivially*. Informed by the intersectionality theory and the linguistic theory by Firth, we propose a simple yet effective approach (JoSEC) for constructing the intersectional bias subspace using the nonlinear geometry of bias subspaces. JoSEC can reduce intrinsic bias without losing the semantic utility of word embeddings. The broad result of this research is that it is critical to consider biases associated with multiple social categories given the complex nature of human-like biases.

We do note several limitations of this study. First, the empirical observation of the nonlinearity be-

tween different biases needs more rigorous theoretical proof/evidence. A potential result would be an in-depth analysis of bias correlations in a variety of NLP tasks. Questions such as "how are biases correlated? Negative or positive correlation?" might be investigated. Second, research should be conducted to study how to reduce joint and intersectional biases in contextualized word embeddings such as BERT and GPT-3. Third, JoSEC focuses on intrinsic measures, which remain good descriptive metrics for computational social science (Seraphina et al., 2021). However, it might not be relied on to mitigate biases in downstream applications. As real-world scenarios can be more challenging and complicated (e.g., biases induced in the deployment), it is necessary to extend this research to extrinsic measures of biases, which might vary in different applications. Meanwhile, research needs to focus on a more rigorous and transparent data collection process (e.g., recording demographic and identity information of annotators) (Cheng et al., 2021) to help reduce downstream biases. Challenging sets to measure application bias such as (Röttger et al., 2021) need to be created to test the robustness of debiasing methods.

Ethic Statement

We heavily rely on existing bias and fairness metrics, which certainly have no guarantee of *unbiased* word embeddings or machine learning models. In fact, most metrics can only be considered an indicator of bias at most (Seraphina et al., 2021), especially since significant limitations w.r.t. these metrics have been found (Garrido-Muñoz et al., 2021). Therefore, we urge practitioners not to rely on these debiased word embeddings alone, but also at least consider bias mitigation in specific downstream tasks. Further, we did not discuss many other negative impacts of language models that practitioners should consider, such as high energy consumption or not including all stakeholders in the design phase (Bender et al., 2021).

Acknowledgements

This work is supported by the Office of Naval Research (ONR) under the grant N00014-21-1-4002 and National Science Foundation (NSF) under the grants #2036127 and #2114789. Lu Cheng is also supported by the startup fund provided by the Department of Computer Science at the University of Illinois Chicago. The views, opinions, and/or findings expressed are the authors' and should not be interpreted as representing the official views or policies of the funding agency.

References

Hervé Abdi and Lynne J Williams. 2010. Principal component analysis. *Wiley interdisciplinary reviews: computational statistics*, 2(4):433–459.

Nazar Akrami, Bo Ekehammar, and Robin Bergh. 2011. Generalized prejudice: Common and specific components. *Psychological Science*, 22(1):57–59.

Gordon Willard Allport, Kenneth Clark, and Thomas Pettigrew. 1954. The nature of prejudice.

Emily M Bender, Timnit Gebru, Angelina McMillan-Major, and Shmargaret Shmitchell. 2021. On the dangers of stochastic parrots: Can language models be too big?. In *Proceedings of the 2021 ACM Conference on Fairness, Accountability, and Transparency*, pages 610–623.

Margaret M Bierly. 1985. Prejudice toward contemporary outgroups as a generalized attitude. *Journal of Applied Social Psychology*, 15(2):189–199.

Su Lin Blodgett, Solon Barocas, Hal Daumé III, and Hanna Wallach. 2020. Language (technology) is power: A critical survey of "bias" in nlp. In *ACL*, pages 5454–5476.

Tolga Bolukbasi, Kai-Wei Chang, James Y Zou, Venkatesh Saligrama, and Adam T Kalai. 2016. Man is to computer programmer as woman is to homemaker? debiasing word embeddings. In *NeurIPS*.

Shikha Bordia and Samuel R Bowman. 2019. Identifying and reducing gender bias in word-level language models. In *NAACL SRW*.

Joy Buolamwini and Timnit Gebru. 2018. Gender shades: Intersectional accuracy disparities in commercial gender classification. In *FAccT*, pages 77–91. PMLR.

Aylin Caliskan, Joanna J Bryson, and Arvind Narayanan. 2017. Semantics derived automatically from language corpora contain human-like biases. *Science*, 356(6334):183–186.

Lu Cheng, Suyu Ge, and Huan Liu. 2022a. Toward understanding bias correlations for mitigation in nlp. *arXiv preprint arXiv:2205.12391*.

Lu Cheng, Ahmadreza Mosallanezhad, Yasin N Silva, Deborah L Hall, and Huan Liu. 2022b. Bias mitigation for toxicity detection via sequential decisions. In *SIGIR*, pages 1750–1760.

Lu Cheng, Kush R Varshney, and Huan Liu. 2021. Socially responsible ai algorithms: Issues, purposes, and challenges. *Journal of Artificial Intelligence Research*, 71:1137–1181.

Kimberlé Crenshaw. 1989. Demarginalizing the intersection of race and sex: A black feminist critique of antidiscrimination doctrine, feminist theory and antiracist politics. *u. Chi. Legal f.*, page 139.

Pieter Delobelle, Ewoenam Kwaku Tokpo, Toon Calders, and Bettina Berendt. 2021. Measuring fairness with biased rulers: A survey on quantifying biases in pretrained language models. *arXiv preprint arXiv:2112.07447*.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. Bert: Pre-training of deep bidirectional transformers for language understanding. In *NAACL-HLT*.

Lucas Dixon, John Li, Jeffrey Sorensen, Nithum Thain, and Lucy Vasserman. 2018. Measuring and mitigating unintended bias in text classification. In *AIES*, pages 67–73.

J Firth. 1957. A synopsis of linguistic theory 1930-1955 in studies in linguistic analysis, philological society.

James R Foulds, Rashidul Islam, Kamrun Naher Keya, and Shimei Pan. 2020. An intersectional definition of fairness. In *2020 IEEE 36th International Conference on Data Engineering (ICDE)*, pages 1918–1921. IEEE.

Ismael Garrido-Muñoz, Arturo Montejo-Ráez, Fernando Martínez-Santiago, and L Alfonso Ureña-López. 2021. A survey on bias in deep nlp. *Applied Sciences*, 11(7):3184.

Negin Ghavami and Letitia Anne Peplau. 2013. An intersectional analysis of gender and ethnic stereotypes: Testing three hypotheses. *Psychology of Women Quarterly*, 37(1):113–127.

Hila Gonen and Yoav Goldberg. 2019. Lipstick on a pig: Debiasing methods cover up systematic gender biases in word embeddings but do not remove them. In *NAACL-HLT*, pages 609–614.

Wei Guo and Aylin Caliskan. 2020. Detecting emergent intersectional biases: Contextualized word embeddings contain a distribution of human-like biases. In *AIES*.

Ange-Marie Hancock. 2007. When multiplication doesn't equal quick addition: Examining intersectionality as a research paradigm. *Perspectives on politics*, 5(1):63–79.

Rachel T Hare-Mustin and Jeanne Marecek. 1988. The meaning of difference: Gender theory, postmodernism, and psychology. *American psychologist*, 43(6):455.

Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long short-term memory. *Neural computation*, 9(8):1735–1780.

Aída Hurtado and Mrinal Sinha. 2008. More than men: Latino feminist masculinities and intersectionality. *Sex roles*, 59(5-6):337–349.

Arnold S Kahn and Janice D Yoder. 1989. The psychology of women and conservatism: Rediscovering social change. *Psychology of women quarterly*, 13(4):417–432.

Masahiro Kaneko and Danushka Bollegala. 2020. Autoencoding improves pre-trained word embeddings. In *COLING*.

Michael Kearns, Seth Neel, Aaron Roth, and Zhiwei Steven Wu. 2018. Preventing fairness gerrymandering: Auditing and learning for subgroup fairness. In *International Conference on Machine Learning*, pages 2564–2572. PMLR.

Michael A Lepori. 2020. Unequal representations: Analyzing intersectional biases in word embeddings using representational similarity analysis. In *COLING*.

Thomas Manzini, Lim Yao Chong, Alan W Black, and Yulia Tsvetkov. 2019. Black is to criminal as caucasian is to police: Detecting and removing multiclass bias in word embeddings. In *NAACL-HLT*, pages 615–621.

Chandler May, Alex Wang, Shikha Bordia, Samuel R Bowman, and Rachel Rudinger. 2019. On measuring social biases in sentence encoders. In *NAACL-HLT*.

Maryann Parada. 2016. Ethnolinguistic and gender aspects of latino naming in chicago: Exploring regional variation. *Names*, 64(1):19–35.

Jeffrey Pennington, Richard Socher, and Christopher D Manning. 2014. Glove: Global vectors for word representation. In *EMNLP*, pages 1532–1543.

Ella Rabinovich, Yulia Tsvetkov, and Shuly Wintner. 2018. Native language cognate effects on second language lexical choice. *Transactions of the Association for Computational Linguistics*, 6:329–342.

Paul Röttger, Bertie Vidgen, Dong Nguyen, Zeerak Waseem, Helen Margetts, and Janet Pierrehumbert. 2021. Hatecheck: Functional tests for hate speech detection models. In *ACL*, pages 41–58.

Erik F Sang and Fien De Meulder. 2003. Introduction to the conll-2003 shared task: Language-independent named entity recognition. *arXiv preprint cs/0306050*.

Goldfarb-Tarrant Seraphina, Marchant Rebecca, Sanchez Ricardo Muñoz, Pandya Mugdha, and Lopez Adam. 2021. Intrinsic bias metrics do not correlate with application bias. In *ACL*.

Yi Chern Tan and L Elisa Celis. 2019. Assessing social and intersectional biases in contextualized word representations. In *NeurIPS*.

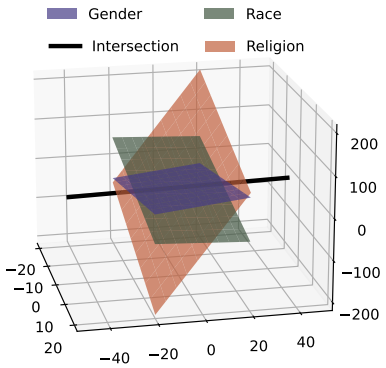

Figure 5: The geometry of Gender, Race, and Religion bias subspaces and the intersectional subspaces (Intersection).

Veronica G Thomas. 2004. The psychology of black women: Studying women's lives in context. *Journal of Black Psychology*, 30(3):286–306.

Jieyu Zhao, Tianlu Wang, Mark Yatskar, Ryan Cotterell, Vicente Ordonez, and Kai-Wei Chang. 2019. Gender bias in contextualized word embeddings. In *NAACL*, pages 629–634.

Jieyu Zhao, Yichao Zhou, Zeyu Li, Wei Wang, and Kai-Wei Chang. 2018. Learning gender-neutral word embeddings. In *EMNLP*.

A Additional Empirical Results for Validating the Intersectional Hypothesis

Following the same procedure in Sec. 4.2.1, we draw subspaces w.r.t. all of three categories as 2-dimensional planes in Figure 5. We observe similar results to those w.r.t. two categories.

B Stimuli for Identifying Intersectional Groups and Biases

The stimuli used to construct the "ground-truth" intersectional bias subspace are from (Caliskan et al., 2017) and (Parada, 2016). We follow (Guo and Caliskan, 2020) to identify the names of intersectional groups and corresponding attributes associated with the intersectional biases. This process is further verified by human subjects (Ghavami and Peplau, 2013). We use this validation set to evaluate the quality of the intersectional subspace constructed by JoSEC. In particular, the ground-truth intersectional subspace is used in the empirical validation of the Intersectional Hypothesis (Appendix A) and comparisons of the debiasing performance between JoSEC and Hard_Insec.

Names to identify intersectional groups:

- **African American females**: Aisha, Keisha, Lakisha, Latisha, Latoya, Malika, Nichelle, Shereen, Tamika, Tanisha, Yolanda, Yvette
- **African American males**: Alonzo, Alphonse, Hakim, Jamal, Jamel, Jerome, Leroy, Lionel, Marcellus, Terrence, Tyrone, Wardell
- **European American females**: Carrie, Colleen, Ellen, Emily, Heather, Katie, Megan, Melanie, Nancy, Rachel, Sarah, Stephanie
- **European American males**: Andrew, Brad, Frank, Geoffrey, Jack, Jonathan, Josh, Matthew, Neil, Peter, Roger, Stephen
- **Mexican American females**: Adriana, Alejandra, Alma, Brenda, Carolina, Iliana, Karina, Liset, Maria, Mayra, Sonia, Yesenia
- **Mexican American males**: Alberto, Alejandro, Alfredo, Antonio, César, Jesús, José, Juan, Miguel, Pedro, Rigoberto, Rogelio

Attributes indicating intersectional biases:

- **Intersectional biases of African American females**: aggressive, athletic, bigbutt, confident, darkskinned, friedchicken, ghetto, loud, overweight, promiscuous, unfeminine, unintelligent, unrefined
- **Intersectional biases of African American males**: athletic, criminals, dangerous, darkskinned, gangsters, hypersexual, lazy, loud, poor, rapper, tall, unintelligent, violent
- **Intersectional Biases of European American females**: arrogant, attractive, blond, ditsy, emotional, feminine, highstatus, intelligent, materialistic, petite, racist, rich, submissive, tall
- **Intersectional biases of European American males**: allAmerican, arrogant, attractive, blond, high-status, intelligent, leader, privileged, racist, rich, sexist, successful, tall
- **Intersectional biases of Mexican American females**: cook, curvy, darkskinned, feisty, hardworker, loud, maids, promiscuous, sexy, short, uneducated, unintelligent
- **Intersectional biases of Mexican American males**: aggressive, arrogant, darkskinned,

day-laborer, drunks, hardworker, illegal-immigrant, jealous, macho, poor, promiscuous, short, uneducated, unintelligent, violent

C Additional Experimental Results for RQ. 3

This section presents results for the semantic utility of debiased word embeddings under the *Model Retraining* paradigm. In particular, we show Precision, Recall, F1 scores w.r.t. NER Tagging, POS Tagging, and POS Chunking, respectively. We retrain the model with the debiased word embeddings. As shown in Table 4, we observe similar results to those using Embedding Replacement. Together with the results in Sec. 5.3.3, we show that debiasing for multiple social categories using the hard-debiasing method does not have a significant influence on the semantic utility of word embeddings. This applies to both joint and intersectional debiasing.

D Experimental Setup for Toxicity Classification in RQ. 4

We employ LSTM (Long Short-Term Memory) (Hochreiter and Schmidhuber, 1997) as the toxicity classifier, which takes the input of a variety of word embeddings considered in this work. We then measure the biases using two commonly adopted metrics (Dixon et al., 2018): False Positive Equality Difference (FPED) and False Negative Equality Difference (FNED). FNED/FPED is defined as the sum of deviations of group-specific False Negative Rates (FNRs)/False Positive Rates (FPRs) from the overall FNR/FPR. Given N demographic groups (e.g., female and male in gender) and we denote each group as $G_{i\in\{1,\ldots,N\}}$, FNED and FPED are calculated as:

$$\begin{aligned} FNED &= \sum_{i\in\{1,\ldots,N\}} |FNR - FNR_{G_i}|, \\ FPED &= \sum_{i\in\{1,\ldots,N\}} |FPR - FPR_{G_i}|. \end{aligned} \tag{10}$$

where FNR_{G_i} denotes the FNR calculated over group G_i and FNR is calculated over the entire training set. A debiased model is expected to have similar FNR and FPR for different groups belonging to the same identity, therefore, smaller FNED and FPED are desired. Ideally, the sum of FNED and FPED is close to zero.

Table 4: Utility of biased and debiased word embeddings in NER Tagging (NER), POS Tagging (POS-T), and POS Chunking (POS-C) tasks, under the *Model Retraining* paradigm. Word embeddings are debiased by hard-debiasing with different subspaces. Δ denotes the change before and after debiasing.

	Hard_Seq			Hard_Insec			SUM			MEAN			JoSEC		
Tasks	NER	POS-T	POS-C	NER	POS-T	POS-C	NER	POS-T	POS-C	NER	POS-T	POS-C	NER	POS-T	POS-C
Biased F1	0.97	1.00	0.99	0.97	1.00	0.99	0.97	1.00	0.99	0.97	1.00	0.99	0.97	1.00	0.99
Δ F1	-0.01	0.01	0.00	-0.01	0.00	0.00	-0.01	0.01	0.00	-0.00	0.00	0.00	-0.00	0.00	0.00
Δ Precision	-0.00	0.00	0.00	-0.01	0.00	0.00	-0.01	0.00	0.00	0.00	0.00	0.00	-0.00	0.00	0.00
Δ Recall	-0.02	0.03	0.02	-0.02	0.01	0.02	-0.01	0.02	0.01	-0.01	0.01	0.01	-0.00	0.01	0.01

Debiasing isn't enough! – On the Effectiveness of Debiasing MLMs and their Social Biases in Downstream Tasks

Masahiro Kaneko[1] **Danushka Bollegala**[2,3*] **Naoaki Okazaki**[1]
[1]Tokyo Institute of Technology [2]University of Liverpool [3]Amazon
masahiro.kaneko@nlp.c.titech.ac.jp
danushka@liverpool.ac.uk okazaki@c.titech.ac.jp

Abstract

We study the relationship between task-agnostic intrinsic and task-specific extrinsic social bias evaluation measures for Masked Language Models (MLMs), and find that there exists only a weak correlation between these two types of evaluation measures. Moreover, we find that MLMs debiased using different methods still re-learn social biases during fine-tuning on downstream tasks. We identify the social biases in both training instances as well as their assigned labels as reasons for the discrepancy between intrinsic and extrinsic bias evaluation measurements. Overall, our findings highlight the limitations of existing MLM bias evaluation measures and raise concerns on the deployment of MLMs in downstream applications using those measures.

1 Introduction

Text representations produced by MLMs have revolutionised NLP by improving the performance of numerous downstream applications (Peters et al., 2018; Devlin et al., 2019; Liu et al., 2019; Lan et al., 2020; Yang et al., 2019). Unfortunately, large-scale pretrained MLMs demonstrate worrying levels of social biases when applied to downstream tasks (May et al., 2019; Nadeem et al., 2021; Kaneko and Bollegala, 2021a; Kaneko et al., 2022b). Given that real-world Natural Language Processing (NLP) systems such as machine translation systems, dialogue systems, etc. are used by millions of users world-wide (Hovy and Spruit, 2016), it remains an important responsibility to accurately evaluate the social biases in MLMs prior to deployment.

Two types of social bias evaluation measures have been proposed for MLMs in prior work (Goldfarb-Tarrant et al., 2021; Cao et al., 2022): (a) task-agnostic *intrinsic* evaluation measures (Nangia et al., 2020; Nadeem et al., 2021; Kaneko and Bollegala, 2022) that use the likelihood scores assigned by an MLM under evaluation for sentences representing various social biases, and (b) task-specific *extrinsic* evaluation measures that use the data from downstream NLP tasks such as predicting the occupation of a person from their biographies (De-Arteaga et al., 2019; Bartl et al., 2020). In contrast to intrinsic measures that directly probe into the biases in MLMs, extrinsic measures holistically evaluate an NLP system that uses an MLM.

Much prior work has decoupled the intrinsic and extrinsic bias evaluations for simplicity, and have assumed that an MLM unbiased according to an intrinsic measure will remain to be so under extrinsic evaluations too. However, this unverified hypothesis begs the following question: *Can we reliably predict the social biases of an MLM when it is applied to a particular downstream task using only intrinsic evaluation measures?* To answer this question, we conduct a comprehensive study using 7 pretrained MLMs and their debiased versions using three different debiasing methods. However, only weak correlations are found between intrinsic and extrinsic bias evaluation measures. This is a worrying proposition because intrinsic measures are often used to decide whether an MLM should be deployed in a downstream application.

We further investigate on why MLMs learn social biases during downstream task fine-tuning and identify two sources: *instance-related* biases and *label-related* biases. The training instances for downstream tasks contain unfair associations that are learnt by the masked language modelling objective, which we name instance-related biases. We find that debiased MLMs gradually learn such biases during fine-tuning, leading to disagreements between intrinsic and extrinsic evaluation measures. Moreover, the labels assigned to downstream train-

*Danushka Bollegala holds concurrent appointments as a Professor at University of Liverpool and as an Amazon Scholar. This paper describes work performed at the University of Liverpool and is not associated with Amazon.

Proceedings of the 29th International Conference on Computational Linguistics, pages 1299–1310
October 12–17, 2022.

ing instances can also represent biased ratings, which are also learnt by the MLMs when their parameters are updated in an end-to-end manner when predicting task labels. Overall, we find weak correlations between intrinsic and extrinsic measures even after the models are fine-tuned on downstream task data. Based on our findings, we recommend that intrinsic bias evaluation measure alone must not be used to determine whether an MLM should be deployed in a downstream NLP application. The code and data used in this paper will be publicly released to facilitate reproduction of results upon paper acceptance.

2 Related Work

Social biases have been reported in models trained for numerous downstream tasks. Kiritchenko and Mohammad (2018) examined 219 automatic sentiment analysis systems that participated in the SemEval-2018 Task 1 (Mohammad et al., 2018) and found that many systems show statistically significant gender or racial biases. Díaz et al. (2019) investigated the age-related biases in sentiment classification and found that across 15 sentiment analysis tools, sentences containing adjectives describing the youth were 66% more likely to be scored positively than those describing the elderly. The problem is also observed in MLMs trained for many languages (Kaneko et al., 2022b).

Numerous bias mitigation approaches have been proposed for MLMs such as fine-tuning (Kaneko and Bollegala, 2021a; Lauscher et al., 2021), counterfactual data augmentation (Hall Maudslay et al., 2019; Zmigrod et al., 2019) and parameter dropout (Webster et al., 2020). Liang et al. (2020a) proposed a method to debias the sentence embeddings created from BERT (Devlin et al., 2019) and ELMo (Peters et al., 2018) inspired by hard debiasing (Bolukbasi et al., 2016), a method originally proposed for static word embeddings. However, not all biases are adequately mitigated by the current proposals (Gonen and Goldberg, 2019). Our focus in this paper is the evaluation and not debiasing methods.

Intrinsic bias evaluation measures (Nangia et al., 2020; Nadeem et al., 2021; Kaneko and Bollegala, 2022) evaluate the social biases in a given MLM standalone, independently of any downstream applications. Pseudo log-likelihood scores assigned to stereotypical vs. antistereotypical examples (either manually written or automatically generated using templates) have been used to evaluate the social biases in an MLM as further detailed in § 3. Considering that MLMs are used to represent input texts in various downstream tasks, several prior work have argued that their social biases must be evaluated with respect to those tasks (De-Arteaga et al., 2019; Bartl et al., 2020; Webster et al., 2020). However, decoupling tasks and MLMs makes the bias evaluation simpler and task-independent, which is attractive because MLMs are designed as generic text representations for a wide-range of downstream tasks. We further discuss extrinsic bias evaluation measures in § 3.

Goldfarb-Tarrant et al. (2021) studied social biases in static word embeddings using the Word Embedding Association Test (WEAT) as an intrinsic measure and coreference detection and hate speech detection as extrinsic evaluation tasks. They found the correlations between intrinsic and extrinsic measures to be weak or negative. Although we share the motivation with Goldfarb-Tarrant et al. (2021), our focus is contextualised embeddings produced by MLMs, which have outperformed static word embeddings in numerous tasks, hence more widely used in real-world NLP applications than static word embeddings.

Cao et al. (2022) studied the correlation between intrinsic bias scores (i.e. Contextualised Embedding Association Test (Guo and Caliskan, 2021, CEAT) and Increased Log Probability Score (Kurita et al., 2019, ILPS)) and extrinsic bias scores for three tasks: toxicity (Jigsaw, 2019), hate-speech (Mathew et al., 2021) and sentiment (Dhamala et al., 2021) detection. Similar to Goldfarb-Tarrant et al. (2021) they also found weak correlations. As reasons for this lack of correlation they highlight misalignment between the metrics such as the notion of bias, protected groups, and noise in the evaluation datasets. Our contributions are complementary to theirs because we consider different intrinsic and extrinsic measures, not covered by them. Moreover, our extrinsic tasks are directly related to social biases. We observe weak correlations between intrinsic and extrinsic evaluation measures, supporting their claims.

3 Bias Evaluation Measures

Several measures have already been proposed in prior work for evaluating the social biases encoded in MLMs. These measures can be broadly categorised into two groups depending on whether they

evaluate social biases in MLMs with respect to a particular downstream task or not. We refer to measures that evaluate social biases in MLMs on their own right, independently of any downstream tasks as *intrinsic MLM bias scores* and describe three such measures. In contrast, we refer to the measures that evaluate social biases in MLMs when they are applied to solve a specific downstream task such as Natural Language Inference (NLI), Semantic Textual Similarity (STS) or predicting occupations from biographies as *extrinsic MLM bias scores* and describe those measures.

To describe the intrinsic measures, let us consider a test sentence $S = w_1, w_2, \ldots, w_{|S|}$, containing length $|S|$ sequence of tokens w_i, where part of S is modified to create a stereotypical (or lack of thereof) example for a particular social bias. For example, let's consider the sentence-pair "***John*** *completed* ***his*** *PhD in machine learning*" vs. "***Mary*** *completed* ***her*** *PhD in machine learning*". Here, {*John*, *his*} are the modified tokens for the first sentence, whereas for the second sentence they are {*Mary*, *her*}. The unmodified tokens between the two sentences are {*completed*, *PhD*, *in*, *machine*, *learning*}.

Let us denote the list of modified tokens in a given sentence S by M and the list of unmodified tokens by U, so that $S = M \cup U$ is the list of all tokens in S. [1] Given an MLM with pretrained parameters θ to evaluate for social biases, let us denote the probability $P_{\text{MLM}}(w_i|S_{\backslash w_i}; \theta)$ that the MLM assigns to a token w_i conditioned on the remainder of the tokens, $S_{\backslash w_i}$. Salazar et al. (2020) demonstrated that $\text{PLL}(S)$, the pseudo-log-likelihood (PLL) score of sentence S given by (1), can be used to evaluate the preference expressed by an MLM for S, similarly to using log-probabilities for evaluating the naturalness of sentences using conventional language models.

$$\text{PLL}(S) := \sum_{i=1}^{|S|} \log P_{\text{MLM}}(w_i|S_{\backslash w_i}; \theta) \quad (1)$$

PLL scores for MLMs can be computed directly and are more uniform across sentence lengths (no left-to-right bias), allowing us to recognize natural sentences in a language (Wang and Cho, 2019). As we will see later, PLL can be used to define bias evaluation scores for MLMs in a variety of ways.

[1] Note that to account for repeated occurrences of a word in a sentence, we examine lists rather than sets.

Intrinsic: StereoSet Score The probability of generating the modified tokens given the unmodified tokens in S was expressed as $P(M|U; \theta)$ by Nadeem et al. (2020). We refer to this as StereoSet Score (**SSS**) and it is given by (2).

$$\text{SSS}(S) := \frac{1}{|M|} \sum_{w \in M} \log P_{\text{MLM}}(w|U; \theta) \quad (2)$$

$|M|$ denotes the length of M. SSS presents a challenge because, when comparing $P(M|U; \theta)$ for modified words like *John*, we might see high probabilities simply because these words occur frequently in the data used to train the MLM and not because the MLM has picked up a social bias.

Intrinsic: CrowS-Pairs Score The CrowS-Pairs Score (**CPS**) is a scoring formula provided by (3) that was defined as $P(U|M; \theta)$ by Nangia et al. (2020) to address the frequency-bias in SSS.

$$\text{CPS}(S) := \sum_{w \in U} \log P_{\text{MLM}}(w|U_{\backslash w}, M; \theta) \quad (3)$$

Normalization is not performed here because the length of unmodified tokens is the same. However, by masking and predicting one token w at a time from U, we are effectively changing the context $(U_{\backslash w}, M)$ used as input by the MLM. This has two disadvantages. First, by removing w from the sentence, the MLM loses information that it can use to predict w. As a result, the prediction accuracy of w may decrease, rendering bias evaluations untrustworthy. Second, the remaining tokens $\{U_{\backslash w}, M\}$ may still be biased even if we remove one token w at a time from U. Moreover, the context on which we condition the probabilities continuously varies across predictions.

Intrinsic: All Unmasked Likelihood with Attention weights Kaneko and Bollegala (2022) proposed All Unmasked Likelihood with Attention weights (**AULA**) to overcome the above-mentioned disfluencies in previously proposed MLM bias evaluation measures. To begin, rather than masking out tokens from S, AULA provides the entire sentence to the MLM. Second, all tokens in S that appear in between the start and end of sentences are predicted by AULA. Furthermore, AULA computes the likelihood using attention weights to evaluate social biases based on the relative importance of

words in a sentence, as given by (4).

$$\mathrm{AULA}(S) := \frac{1}{|S|}\sum_{i=1}^{|S|}\alpha_i \log P_{\mathrm{MLM}}(w_i|S;\theta) \tag{4}$$

Here, α_i represents the average of all multi-head attentions associated with w_i.

Given a score function $f \in \{\mathrm{SSS}, \mathrm{CPS}, \mathrm{AULA}\}$, we use the percentage of stereotypical (S^{st}) test sentences preferred by the MLM over anti-stereotypical (S^{at}) ones to define the corresponding bias evaluation measure (**bias score**) as follows:

$$\left(\frac{100}{N}\sum_{(S^{\mathrm{st}},S^{\mathrm{at}})}\mathbb{I}(f(S^{\mathrm{st}}) > f(S^{\mathrm{at}}))\right) - 50 \tag{5}$$

Here, the indicator function $\mathbb{I}$ returns 1 if its argument is True and 0 otherwise, and N is the total number of test instances. Values close to 0 indicate that the MLM under consideration is neither stereotypically nor anti-stereotypically biased; thus, it can be considered unbiased. Values less than 0 indicate a bias toward the anti-stereotypical group, while values greater than 0 indicate a bias toward the stereotypical group.

Extrinsic: BiasBios To evaluate the gender bias in occupation classification, De-Arteaga et al. (2019) proposed a dataset where they collected biographies written in English from Common Crawl. For this purpose, they first filter lines that begin with a name-like-pattern (i.e. a sequence of two capitalised words) followed by the string "is a(n) (xxx) *title*", where *title* is an occupation from the BLS Standard Occupation Classification system.[2] They used twenty-eight frequent occupations in Common Crawl to collect 397,340 biographies. The task is to predict the people's occupations, taken from the first sentence of their biographies, given the remainder of their biographies. For example, given the hypothetical biography of a nurse: *She graduated from Lehigh University, with honours in 1998. Nancy has years of experience in weight loss surgery, patient support, education, and diabetes.*, the goal is to predict the occupation.

An MLM under evaluation is used to create an embedding for a biography excluding the title sentence, and a softmax classifier is trained using those biography embeddings to predict the occupation of the person described in each biography. If the MLM is unbiased with respect to gender, we would expect it to have approximately similar true positive rates (TPRs) averaged over all twenty-eight occupations for both male and female gendered persons. The TPR gap between the TPRs for classifying occupations in female and male biographies is used as an extrinsic MLM bias score. An unbiased MLM is expected to predict occupations with similar TPRs for both genders.

Extrinsic: STS-bias Semantic Textual Similarity (STS) is a task which evaluates the degree of semantic similarity between two sentences. Webster et al. (2020) proposed STS-bias to evaluate gender bias by adapting STS formulation. Specifically, to create the STS-bias dataset, they collect the 276 sentences from the STS-B test set, which start with *A man* or *A woman*, and discard the sentences with multiple gendered words.[3] For each template, they formed two sentence pairs per profession from Rudinger et al. (2018), one representing the male gender and the other for the female gender. For example, given a pair "*A man is walking*" and "*A nurse is walking*" from STS-B dataset, they create a new pair "*A woman is walking*" and "*A nurse is walking*". If not relying on gendered correlations, a model should assign equal estimates of similarity to the two pairs. Therefore, following Rudinger et al. (2017), as a score we use the difference between similarities of such two pairs to measure how similar model predictions actually are regarding gender. They compute the Pearson correlation between the scores per profession and the representation in that profession listed in the US Bureau of Labor Statistics (Caliskan et al., 2017). STS-bias does not consider the directionality of the gender bias. Therefore, we report absolute values of all bias scores.

Extrinsic: NLI-bias Dev et al. (2019) used natural language inference (NLI) for evaluating gender bias. This dataset contains sentence triples where a premise must be neutral in entailment w.r.t. two hypotheses. For example, "*The accountant ate a bagel*" should neither entail nor contradict the sentences "*The man ate a bagel*" and "*The woman ate a bagel*". Given a set containing $\mathcal{D}$ test instances, let the entailment predictor's probabilities

[2] https://www.bls.gov/soc

[3] Gender word list: https://github.com/uclanlp/gn_glove/tree/master/wordlist

for the d-th instance for entail, neutral and contradiction labels be respectively e_d, n_d and c_d. Then, they proposed the measure Fraction Neutral (FN): $\text{FN} = \frac{1}{|\mathcal{D}|}\sum_{d=1}^{\mathcal{D}} \mathbf{1}[\text{neutral} = \max(e_d, n_d, c_d)]$. For an ideal (bias-free) embedding, FN measure would be 1. They compute the FN for each female and male sentences and use the difference between them as the bias score in the NLI-bias.

4 Experiments and Findings

4.1 Datasets

We used Crows-Pairs dataset (Nangia et al., 2020) as CPS and AULA evaluation data, and used StereoSet dataset (Nadeem et al., 2021) as SSS evaluation data. In BiasBios[4], we use the average performance gap in the male and female groups aggregated across all occupations to measure the bias, following Zhao et al. (2020a). We crawled BiasBios data using publicly available code from the original authors, and used 80% of the collected data as a training data, 10% as development data, and 10% as evaluation data. Because the STS-bias dataset is not publicly available, we created the dataset following to the procedure proposed by Webster et al. (2020). We used STS-B as training and development data (Cer et al., 2017)[5] for STS-bias, following Webster et al. (2020). We used the Multi-Genre Natural Language Inference (MNLI) as training data and development data (Williams et al., 2018)[6] for NLI-bias [7], following Kaneko and Bollegala (2021a).

4.2 Debiasing methods

We used three previously proposed methods that are widely used for debiasing MLMs such as All-layer Token-level debiasing (**AT**; Kaneko and Bollegala, 2021a), Counterfactual Data Augmentation debiasing (**CDA**; Webster et al., 2020), and dropout debiasing (**DO**; Webster et al., 2020). These methods represent conceptually different approaches for debiasing and can be applied to any pretrained MLM model/algorithm.

AT debiasing: Kaneko and Bollegala (2021a) proposed a method for debiasing MLMs via fine-tuning by simultaneously (a) preserving the semantic information in the pretrained MLM, and (b) removing discriminative gender-related biases via an orthogonal projection in the hidden layers by operating at token- or sentence-levels, enabling debiasing on different layers and levels in the pre-trained contextualised embedding model. It is independent of model architectures and pre-training methods, thus can be adapted to a wide range of MLMs. They showed that applying token-level debiasing for all tokens and across all layers of an MLM produces the best performance. We perform an additional phase of fine-tuning using AT debiasing.

CDA debiasing: CDA is a debiasing strategy to mitigate social bias based on data (Webster et al., 2020). CDA involves re-balancing a corpus by swapping bias attribute words in a dataset. For example, the sentence "*the doctor treated his patient*" could be changed to "*the doctor treated her patient*". The re-balanced corpus is then often used for further training to debias a model. We experiment with debiasing pre-trained MLMs by performing an additional phase of fine-tuning on counterfactually augmented sentences.

DO debiasing Webster et al. (2020) proposed dropout regularization for a bias mitigation. They increased the dropout parameters for MLM's attention weights and hidden activations and further pre-trained them. They showed that increased dropout regularization reduces gender bias within these MLMs. They hypothesize that dropout's interruption of the attention mechanisms within MLMs prevents MLMs from learning undesirable associations between words. Similar to AT and CDA, we perform an additional phase of fine-tuning using increased dropout regularization.

We used news-commentary-v15[8] corpus as additional training data for debiasing. We used publicly available code[9] and default hyperparameters for AT debiasing. We used Zhao et al. (2018)'s word list to replace a feminine word to the corresponding masculine word and vice versa in sentences in the training data for CDA debiasing. We set the dropout ratio to 0.15 for attention probabilities and 0.2 for all fully connected layers in DO debiasing. Hyperparameters of CDA and DO debiasing are set to their default values in the Huggingface Transformers library (Wolf et al., 2020).

[4] https://github.com/microsoft/biosbias
[5] github.com/facebookresearch/SentEval
[6] cims.nyu.edu/~sbowman/multinli/
[7] https://github.com/sunipa/On-Measuring-and-Mitigating-Biased-Inferences-of-Word-Embeddings
[8] https://data.statmt.org/news-commentary/v15/
[9] https://github.com/kanekomasahiro/context-debias

4.3 Models and Hyperparameters

In our experiments, we used the following seven MLMs: bert-base-uncased (**bert-bu**)[10], bert-base-cased (**bert-bc**),[11] bert-large-uncased (**bert-lu**),[12] bert-large-cased (**bert-lc**),[13] roberta-base (**roberta-b**),[14] roberta-large (**roberta-l**),[15] albert-base (**albert-b**).[16]

To fine-tune an MLM for a downstream task, we first create an embedding for a training instance using the given MLM. Specifically, we use the [CLS] token embedding produced by the MLM to represent an instance. For STS and NLI tasks where an instance consists of a pair of sentences, we concatenate the two sentences in the pair and use it as the input to the MLM. Finally, a classification head (i.e. a softmax layer) is used to predict the label for a training instance, and cross entropy error is backpropagated to update the MLM parameters.

For downstream tasks, the best performance checkpoint on development data is selected from $\{16, 32, 64\}$ for batch size, {1e-5, 3e-5, 5e-5} for learning rate, $\{1, 3, 5\}$ for epoch number with greedy search. Maximum sentence length is set to 128 tokens and we used four Tesla V100 GPUs in our experiments. All other hyperparameters, are set to their default values in the huggingface library.

4.4 Debiasing vs. Bias in Downstream Tasks

We examine whether the debiasing methods proposed for MLMs can mitigate the social biases when those debiased MLM are used in downstream tasks. Specifically, given an MLM, M, we use each debiasing method described in §4.2 independently on M to create a debiased version, M_{deb}. Let, $f(M)$, denote the bias score of M measured using a bias evaluation measure, f, described in §3. Figure 1 shows the reduction of bias $f(M_{\text{deb}}) - f(M)$ due to debiasing for multiple MLMs and debiasing methods for different f. For all debiasing methods, we see a reduction in biases according to intrinsic bias evaluation measures SSS, CPS and AULA. However, among the extrinsic bias evaluation measure, we see a reduction according to STS, but an *increase* in biases according to BiasBios and NLI. For example, AT for **bert-bu** mitigates bias in all intrinsic measures, but increases biases in all extrinsic measures. This shows that debiasing methods do *not* always reduces extrinsic biases, despite appearing to be debiased with respect to intrinsic measures. Note that the training dataset sizes for BioBias, STS-B and MLNI are respectively 251K, 5.7K, and 392.7K. Therefore, STS-B has fewer training data instances than for the other two tasks, and the number of fine-tuning iterations required is also smaller. For these reasons, we believe that debiased MLMs are less influenced when fine-tuned on STS-B.

	Feminine	Masculine	ILSP
BiasBios	20,720	35,412	-0.53
STS-B	1,304	2,126	-0.48
MNLI	55,588	139,575	-0.91

Table 1: Frequency of feminine and masculine words in downstream task data. ILSP represents the degree of bias between feminine and masculine words.

5 Why is Debiasing less Effective for Downstream Tasks?

To further investigate the discrepancies between intrinsic and extrinsic bias evaluation measures observed in §4.4, recall that a debiased MLM can relearn social biases in two different sources via fine-tuning: *instance-related* biases and *label-related* biases. A training data point of a downstream task consists of an *instance* (i.e. sentence or text) and a *label* (i.e. entailment labels, similarity ratings etc.). Instances alone could express social biases such as in the sentence "*All Muslims are terrorists*". By fine-tuning a debiased MLM on such instances using the masking training objective, it can learn racially discriminatory associations. We call this the instance-related bias in fine-tuning.

Second, even if an instance is unbiased, coupled with its label, the data point could still express a social bias. For example, the sentence pair ("*A nurse is walking*", "*A woman is walking*") represents an instance, which does not represent a gender bias. However, if we are told that the first sentence entails the second (i.e. an entailment label assigned to this instance), it will represent a gender bias. An NLP model can easily learn such label-related biases in an end-to-end training setting by propagating the prediction losses all the way back to the MLM layers.

To examine the instance-related gender biases in

[10] huggingface.co/bert-base-uncased
[11] huggingface.co/bert-base-cased
[12] huggingface.co/bert-large-uncased
[13] huggingface.co/bert-large-cased
[14] huggingface.co/roberta-base
[15] huggingface.co/roberta-large
[16] huggingface.co/albert-base-v2

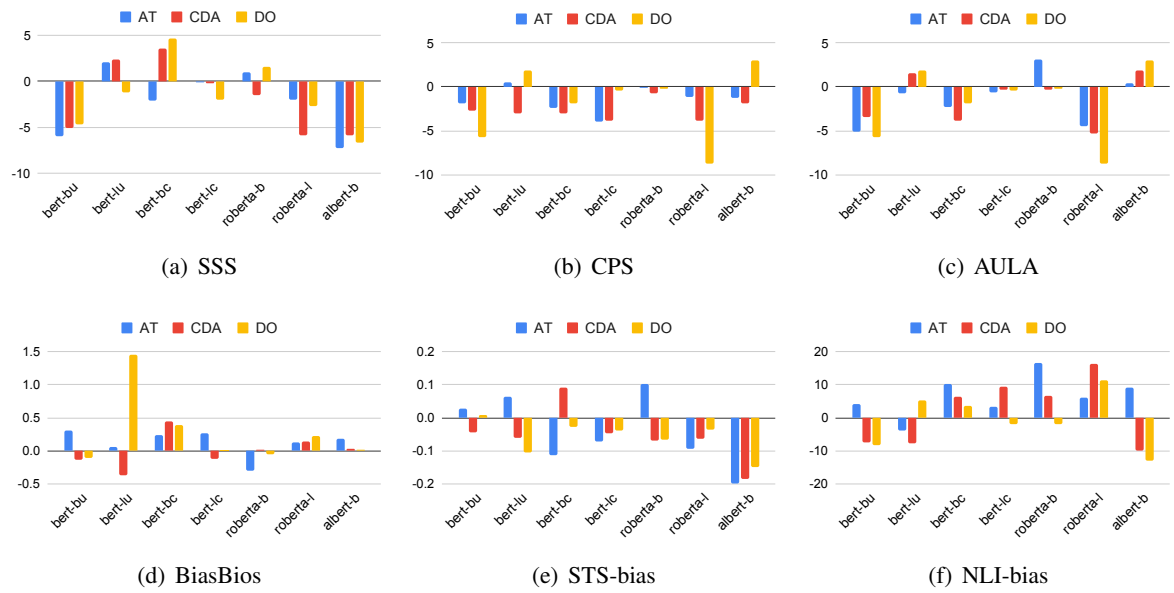

(a) SSS (b) CPS (c) AULA

(d) BiasBios (e) STS-bias (f) NLI-bias

Figure 1: Differences between the bias scores of original vs. debiased MLMs. Negative values indicate that the debiased MLM has a lower bias than its original (non-debiased) version.

downstream training data, we compute ILPS (Kurita et al., 2019) given by $\log(P_f/P_m)$, where P_f and P_m are the probabilities of respectively female and male genders in training instances. Prior work has shown that the frequency imbalance of feminine and masculine words in the training data is related to the gender bias learnt by MLMs (Kaneko and Bollegala, 2022; Kaneko et al., 2022b). Using the gender word lists created by Zhao et al. (2018), we estimate ILPS as shown in Table 1.

From Table 1, we see that all three datasets are highly biased towards the male gender, indicated by the negative ILSP scores. If we arrange the three datasets in the ascending order of their gender biases we get: STS-B < BiasBios < MLNI. Interestingly, this is exactly the same order we would get if we had ordered these datasets considering the number of MLMs that have their biases increased due to fine-tuning in Figure 1 (e, d, f), respectively.

5.1 Re-learning Biases via Fine-Tuning

To further study how biases are learnt during the fine-tuning process, we track the extrinsic bias score of an MLM (and its debiased variants) over the number of fine-tuning training iterations on a particular downstream task. Here, we use **bert-bu** and **roberta-l**, which had low intrinsic biases after debiasing according to all intrinsic bias measures in Figure 1. We save model checkpoints for every 500 fine-tuning iterations up to 5500 iterations and evaluate their extrinsic bias scores.

Figure 2 shows that the extrinsic bias scores of non-debiased (original) MLMs (**roberta-l** and **bert-bu**) and their debiased variants using AT, CDA, and DO. On all downstream tasks, extrinsic bias scores increase during fine-tuning for both non-debiased and debiased MLMs. This shows that biases that were mitigated by debiasing are re-learnt during fine-tuning on downstream training data. Even STS-bias, which reported almost all MLMs to be void of biases after debiasing (Figure 1), shows increasing bias scores during fine-tuning. Overall, we see that debiasing effect gradually decreases as fine-tuning progresses due to the biases in downstream task data. In contrast, Jin et al. (2021) argued that debiased MLMs are unaffected by fine-tuning. However, their evaluation was limited to only one MLM and two datasets both related to gender stereotyping (Jin et al., 2021), whereas we consider seven MLMs, three debiasing methods and six bias evaluation measures. This shows that careful evaluation using various evaluation datasets and MLMs is necessary when verifying the effectiveness of debiasing methods.

Furthermore, according to the intrinsic evaluations in Figure 1, the biases were reduced in both **roberta-l** and **bert-bu** by all debiasing methods. However, there are cases where the bias scores of debiased MLMs are higher than that of non-debiased MLMs even in the early stages of fine-tuning, except for **roberta-l** in the STS-bias in Fig-

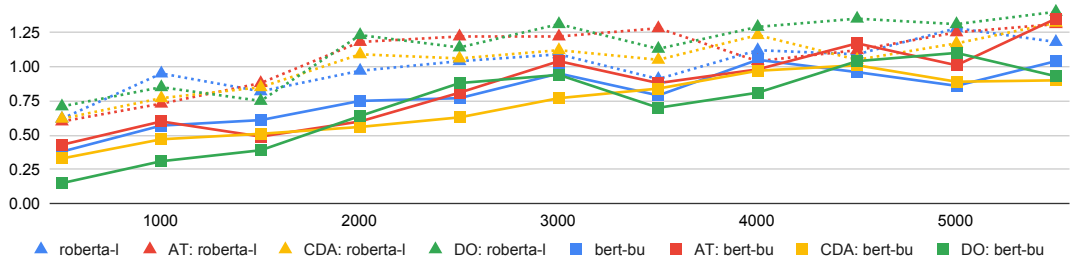

(a) BiasBios

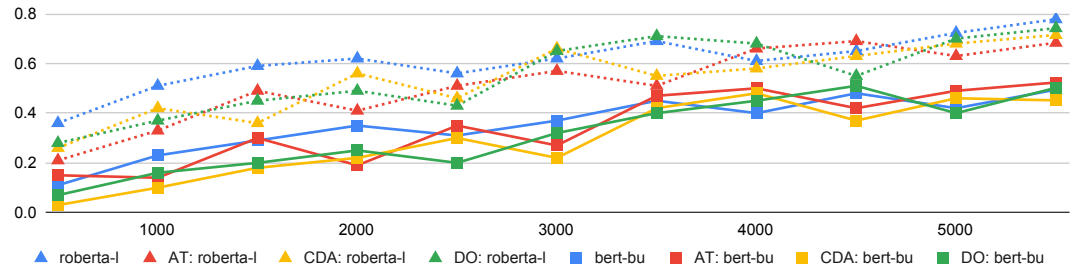

(b) STS-bias

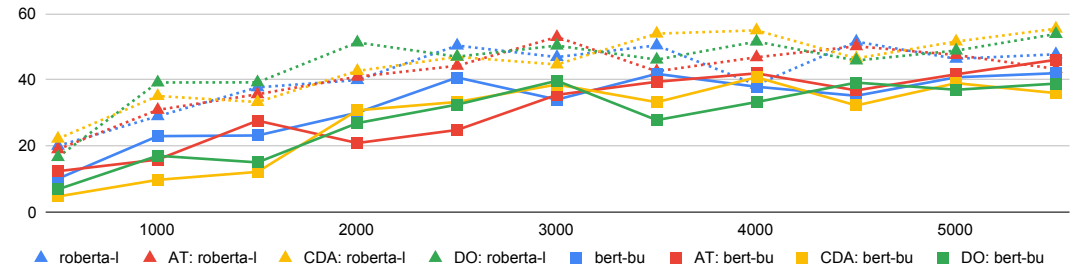

(c) NLI-bias

Figure 2: Extrinsic bias scores measured for MLMs during fine-tuning on downstream instances.

ure 2. Therefore, MLMs judged to be debiased according to intrinsic evaluations alone might not necessarily remain debiased after fine-tuning on downstream task instances.

5.2 Correlations between Intrinsic vs. Extrinsic Bias Evaluation Measures

Given that intrinsic bias evaluation measures are used to determine whether an MLM should be used in a downstream application, it is important to examine what level of correlations exist between intrinsic and extrinsic bias evaluation measures (Blodgett et al., 2021). For this purpose, using 28 MLMs (original 7 MLMs and their debiased versions using three methods, $7 \times 3 + 7 = 28$) we compute the Kendall's τ rank correlation coefficients between intrinsic and extrinsic bias measures. Specifically, in Table 2 we show τ values between the intrinsic bias measure (SSS, CPS, AULA) and extrinsic measures (BiasBios, STS-bias, NLI-bias). In each cell in Table 2, the values to the left of the slashes (/) are computed using the MLMs prior to fine-tuning on downstream data, whereas the values to the right of the slashes are computed using the MLMs that have been fine-tuned on instances of downstream data.[17] Statistically significant ($p < 0.05$) τ values according to the Fisher's exact test are denoted by a †. Cases where the correlation has improved after fine-tuning are are highlighted in **Bold**.

From Table 2, we see that prior to fine-tuning, only two τ values were significant, reconfirming the weak correlations observed in prior work (Delobelle et al., 2021; Goldfarb-Tarrant et al., 2021; Cao et al., 2022). Given that intrinsic bias evaluation measures are agnostic to the biases in downstream

[17] We use the fine-tuned models after 5500 iterations in Figure 2.

	SSS	CPS	AULA
BiasBios	-0.02 / **0.24**	-0.17 / -0.17	-0.20 / **0.11**
STS-bias	$0.53^{\dagger}$ / $0.38^{\dagger}$	-0.08 / **0.12**	0.24 / 0.13
NLI-bias	0.22 / 0.21	0.16 / 0.05	$0.30^{\dagger}$ /0.17

Table 2: Kendall τ correlation coefficients between intrinsic extrinsic bias scores. Values to the left and right of '/' are computed using MLMs respectively prior and post fine-tuning. Statistically significant ($p < 0.05$) τ values are denoted by a †, whereas improved correlations after fine-tuning are highlighted in **Bold**.

task-specific training data, one might have expected to see the correlations between intrinsic and extrinsic measures to improve *after* the MLMs have been fine-tuned on downstream data, thereby capturing some of the biases in downstream data within the MLMs. Surprisingly, we see that this is clearly not the case here and only in three out of nine combinations in Table 2 we see correlations improving after fine-tuning. In particular, for NLI-bias we see that correlations dropping after fine-tuning against all intrinsic measures compared to their values prior to fine-tuning. On the other hand, for BiasBios we see that correlations improving after fine-tuning against SSS and AULA by large margins, while remaining unaffected against CPS. The only significant correlation after fine-tuning is between STS-bias and SSS (i.e. 0.38). However, this too has decreased from its value prior to fine-tuning (i.e. 0.53).

Overall, our results indicate weak and inconsistent correlations between intrinsic and extrinsic bias evaluation measures irrespective of whether the MLMs have been fine-tuned on downstream data or otherwise. Therefore, we recommend that intrinsic evaluation scores alone must not be used to determine whether an MLM is sufficiently unbiased to be deployed to NLP systems that are used by billions of users world-wide.

6 Conclusion

We investigated social biases in 28 MLMs and found that there exist weak correlation between intrinsic and extrinsic bias evaluation measures. Moreover, debiased MLMs re-learn biases during task-specific fine-tuning due to instance and label related biases in downstream task data. Our findings highlight limitations in existing bias evaluation measures and raise serious concerns regarding their implications.

7 Ethical Considerations

Our goal in this paper was to evaluate the correlation between previously proposed and widely-used intrinsic and extrinsic bias evaluation measures for evaluating various types of social biases in pretrained MLMs. However, we did not manually annotate novel social bias datasets or proposed novel bias evaluation measures nor debiasing methods. Therefore, we do not see any ethical issues arising due to data annotation, or via proposals of novel evaluation metrics or debiasing methods. However, we would like to point out that although we conducted an extensive study using 7 MLMs, 3 debiasing methods, 3 intrinsic evaluation measures and 3 extrinsic evaluation measures, this is still a limited subset of all possible scenarios. For example, the gender biases we considered in this paper cover only binary gender (Dev et al., 2021). However, non-binary genders are severely underrepresented in textual data used to train MLMs. Moreover, non-binary genders are often associated with derogatory adjectives. Evaluating the correlation between intrinsic and extrinsic non-binary gender is an important next step.

All MLMs used in this study are trained for English. However, as reported in much prior work social biases are language independent and omnipresent in MLMs trained for many languages (Kaneko et al., 2022b; Lewis and Lupyan, 2020; Liang et al., 2020b; Bartl et al., 2020; Zhao et al., 2020b). We plan to extend this study to cover non-English MLMs in the future.

Kaneko et al. (2022a) show that combining multiple debiasing methods for word embeddings (Bolukbasi et al., 2016; Kaneko and Bollegala, 2019, 2021b, 2020; Ravfogel et al., 2020) can complement each other's strengths and weaknesses to obtain more effective debiased embeddings. Therefore, combining methods to mitigate both intrinsic and extrinsic bias may allow for more effective debiasing.

Finally, biases are not limited to word representations but also appear in sense representations (Zhou et al., 2022). On the other hand, our correlation analysis did not include any sense embedding models.

References

Marion Bartl, Malvina Nissim, and Albert Gatt. 2020. Unmasking contextual stereotypes: Measuring and

mitigating BERT's gender bias. In *Proceedings of the Second Workshop on Gender Bias in Natural Language Processing*, pages 1–16, Barcelona, Spain (Online). Association for Computational Linguistics.

Su Lin Blodgett, Gilsinia Lopez, Alexandra Olteanu, Robert Sim, and Hanna Wallach. 2021. Stereotyping Norwegian salmon: An inventory of pitfalls in fairness benchmark datasets. In *NAACL-HLT*, pages 1004–1015, Online. Association for Computational Linguistics.

Tolga Bolukbasi, Kai-Wei Chang, James Y. Zou, Venkatesh Saligrama, and Adam Kalai. 2016. Man is to computer programmer as woman is to homemaker? debiasing word embeddings. In *NIPS*.

Aylin Caliskan, Joanna J Bryson, and Arvind Narayanan. 2017. Semantics derived automatically from language corpora contain human-like biases. *Science*, 356:183–186.

Yang Trista Cao, Yada Pruksachatkun, Kai-Wei Chang, Rahul Gupta, Varun Kumar, Jwala Dhamala, and Aram Galstyan. 2022. On the Intrinsic and Extrinsic Fairness Evaluation Metrics for Contextualized Language Representations. In *Proc. of ACL*.

Daniel Cer, Mona Diab, Eneko Agirre, Iñigo Lopez-Gazpio, and Lucia Specia. 2017. SemEval-2017 task 1: Semantic textual similarity multilingual and crosslingual focused evaluation. In *Proceedings of the 11th International Workshop on Semantic Evaluation (SemEval-2017)*, pages 1–14, Vancouver, Canada. Association for Computational Linguistics.

Maria De-Arteaga, Alexey Romanov, Hanna Wallach, Jennifer Chayes, Christian Borgs, Alexandra Chouldechova, Sahin Geyik, Krishnaram Kenthapadi, and Adam Tauman Kalai. 2019. Bias in bios: A case study of semantic representation bias in a high-stakes setting. In *Proceedings of the Conference on Fairness, Accountability, and Transparency*, FAT* '19, pages 120–128, New York, NY, USA. Association for Computing Machinery.

Pieter Delobelle, Ewoenam Kwaku Tokpo, Toon Calders, and Bettina Berendt. 2021. Measuring fairness with biased rulers: A survey on quantifying biases in pretrained language models. *arXiv preprint arXiv:2112.07447*.

Sunipa Dev, Tao Li, Jeff Phillips, and Vivek Srikumar. 2019. On Measuring and Mitigating Biased Inferences of Word Embeddings.

Sunipa Dev, Masoud Monajatipoor, Anaelia Ovalle, Arjun Subramonian, Jeff Phillips, and Kai-Wei Chang. 2021. Harms of gender exclusivity and challenges in non-binary representation in language technologies. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 1968–1994, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *NAACL-HLT*, pages 4171–4186, Minneapolis, Minnesota. Association for Computational Linguistics.

Jwala Dhamala, Tony Sun, Varun Kumar, Satyapriya Krishna, Yada Pruksachatkun, Kai-Wei Chang, and Rahul Gupta. 2021. Bold: Dataset and metrics for measuring biases in open-ended language generation. In *Proceedings of the 2021 ACM Conference on Fairness, Accountability, and Transparency*, FAccT '21, pages 862–872, New York, NY, USA. Association for Computing Machinery.

Mark Díaz, Isaac Johnson, Amanda Lazar, Anne Marie Piper, and Darren Gergle. 2019. Addressing age-related bias in sentiment analysis. In *Proceedings of the Twenty-Eighth International Joint Conference on Artificial Intelligence, IJCAI-19*, pages 6146–6150. International Joint Conferences on Artificial Intelligence Organization.

Seraphina Goldfarb-Tarrant, Rebecca Marchant, Ricardo Muñoz Sánchez, Mugdha Pandya, and Adam Lopez. 2021. Intrinsic bias metrics do not correlate with application bias. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 1926–1940, Online. Association for Computational Linguistics.

Hila Gonen and Yoav Goldberg. 2019. Lipstick on a Pig: Debiasing Methods Cover up Systematic Gender Biases in Word Embeddings But do not Remove Them.

Wei Guo and Aylin Caliskan. 2021. *Detecting Emergent Intersectional Biases: Contextualized Word Embeddings Contain a Distribution of Human-like Biases*, pages 122–133. Association for Computing Machinery, New York, NY, USA.

Rowan Hall Maudslay, Hila Gonen, Ryan Cotterell, and Simone Teufel. 2019. It's all in the name: Mitigating gender bias with name-based counterfactual data substitution. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 5266–5274, Hong Kong, China. Association for Computational Linguistics.

Dirk Hovy and Shannon L. Spruit. 2016. The social impact of natural language processing. In *Proc. of ACL*, pages 591–598.

Conversational AI Jigsaw. 2019. Unintended bias in toxicity classification. Kaggle.

Xisen Jin, Francesco Barbieri, Brendan Kennedy, Aida Mostafazadeh Davani, Leonardo Neves, and Xiang Ren. 2021. On transferability of bias mitigation effects in language model fine-tuning. In *Proceedings*

of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, pages 3770–3783, Online. Association for Computational Linguistics.

Masahiro Kaneko and Danushka Bollegala. 2019. Gender-preserving debiasing for pre-trained word embeddings. In *ACL*, pages 1641–1650.

Masahiro Kaneko and Danushka Bollegala. 2020. Autoencoding improves pre-trained word embeddings. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 1699–1713, Barcelona, Spain (Online). International Committee on Computational Linguistics.

Masahiro Kaneko and Danushka Bollegala. 2021a. Debiasing pre-trained contextualised embeddings. In *Proc. of 16th conference of the European Chapter of the Association for Computational Linguistics (EACL)*.

Masahiro Kaneko and Danushka Bollegala. 2021b. Dictionary-based debiasing of pre-trained word embeddings. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*, pages 212–223, Online. Association for Computational Linguistics.

Masahiro Kaneko and Danushka Bollegala. 2022. Unmasking the mask – evaluating social biases in masked language models. In *Proceedings of the 36th AAAI Conference on Artificial Intelligence*, page (to appear), Vancouver, BC, Canada.

Masahiro Kaneko, Danushka Bollegala, and Naoaki Okazaki. 2022a. Gender bias in meta-embeddings. *arXiv preprint arXiv:2205.09867*.

Masahiro Kaneko, Aizhan Imankulova, Danushka Bollegala, and Naoaki Okazaki. 2022b. Gender bias in masked language models for multiple languages. In *Proc. of NAACL-HLT*.

Svetlana Kiritchenko and Saif Mohammad. 2018. Examining gender and race bias in two hundred sentiment analysis systems. In *Proceedings of the Seventh Joint Conference on Lexical and Computational Semantics*, pages 43–53, New Orleans, Louisiana. Association for Computational Linguistics.

Keita Kurita, Nidhi Vyas, Ayush Pareek, Alan W Black, and Yulia Tsvetkov. 2019. Measuring bias in contextualized word representations. In *Proceedings of the First Workshop on Gender Bias in Natural Language Processing*, pages 166–172, Florence, Italy. Association for Computational Linguistics.

Zhenzhong Lan, Mingda Chen, Sebastian Goodman, Kevin Gimpel, Piyush Sharma, and Radu Soricut. 2020. ALBERT: A Lite BERT for Self-supervised Learning of Language Representations. *arXiv*, abs/1909.11942.

Anne Lauscher, Tobias Lueken, and Goran Glavaš. 2021. Sustainable modular debiasing of language models. In *Findings of the Association for Computational Linguistics: EMNLP 2021*, pages 4782–4797, Punta Cana, Dominican Republic. Association for Computational Linguistics.

Molly Lewis and Gary Lupyan. 2020. Gender stereotypes are reflected in the distributional structure of 25 languages. *Nature human behaviour*, 4(10):1021–1028.

Paul Pu Liang, Irene Mengze Li, Emily Zheng, Yao Chong Lim, Ruslan Salakhutdinov, and Louis-Philippe Morency. 2020a. Towards debiasing sentence representations. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 5502–5515, Online. Association for Computational Linguistics.

Sheng Liang, Philipp Dufter, and Hinrich Schütze. 2020b. Monolingual and multilingual reduction of gender bias in contextualized representations. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 5082–5093, Barcelona, Spain (Online). International Committee on Computational Linguistics.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. RoBERTa: A Robustly Optimized BERT Pretraining Approach.

Binny Mathew, Punyajoy Saha, Seid Muhie Yimam, Chris Biemann, Pawan Goyal, and Animesh Mukherjee. 2021. HateXplain: A benchmark dataset for explainable hate speech detection. In *Proc. of 35th AAAI Conference on Artificial Intelligence (AAAI)*.

Chandler May, Alex Wang, Shikha Bordia, Samuel R. Bowman, and Rachel Rudinger. 2019. On measuring social biases in sentence encoders. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 622–628, Minneapolis, Minnesota. Association for Computational Linguistics.

Saif Mohammad, Felipe Bravo-Marquez, Mohammad Salameh, and Svetlana Kiritchenko. 2018. SemEval-2018 task 1: Affect in tweets. In *Proceedings of The 12th International Workshop on Semantic Evaluation*, pages 1–17, New Orleans, Louisiana. Association for Computational Linguistics.

Moin Nadeem, Anna Bethke, and Siva Reddy. 2020. StereoSet: Measuring stereotypical bias in pretrained language models.

Moin Nadeem, Anna Bethke, and Siva Reddy. 2021. StereoSet: Measuring stereotypical bias in pretrained language models. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference*

on Natural Language Processing (Volume 1: Long Papers), pages 5356–5371, Online. Association for Computational Linguistics.

Nikita Nangia, Clara Vania, Rasika Bhalerao, and Samuel R. Bowman. 2020. CrowS-pairs: A challenge dataset for measuring social biases in masked language models. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1953–1967, Online. Association for Computational Linguistics.

Matthew E. Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer. 2018. Deep contextualized word representations. In *Proc. of NAACL-HLT*.

Shauli Ravfogel, Yanai Elazar, Hila Gonen, Michael Twiton, and Yoav Goldberg. 2020. Null it out: Guarding protected attributes by iterative nullspace projection. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7237–7256, Online. Association for Computational Linguistics.

Rachel Rudinger, Chandler May, and Benjamin Van Durme. 2017. Social bias in elicited natural language inferences. In *Proceedings of the First ACL Workshop on Ethics in Natural Language Processing*, pages 74–79, Valencia, Spain. Association for Computational Linguistics.

Rachel Rudinger, Jason Naradowsky, Brian Leonard, and Benjamin Van Durme. 2018. Gender bias in coreference resolution. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 2 (Short Papers)*, pages 8–14, New Orleans, Louisiana. Association for Computational Linguistics.

Julian Salazar, Davis Liang, Toan Q. Nguyen, and Katrin Kirchhoff. 2020. Masked language model scoring. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 2699–2712, Online. Association for Computational Linguistics.

Alex Wang and Kyunghyun Cho. 2019. BERT has a mouth, and it must speak: BERT as a Markov random field language model. In *Proceedings of the Workshop on Methods for Optimizing and Evaluating Neural Language Generation*, pages 30–36, Minneapolis, Minnesota. Association for Computational Linguistics.

Kellie Webster, Xuezhi Wang, Ian Tenney, Alex Beutel, Emily Pitler, Ellie Pavlick, Jilin Chen, Ed Chi, and Slav Petrov. 2020. Measuring and reducing gendered correlations in pre-trained models. *arXiv preprint arXiv:2010.06032*.

Adina Williams, Nikita Nangia, and Samuel Bowman. 2018. A broad-coverage challenge corpus for sentence understanding through inference. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 1112–1122, New Orleans, Louisiana. Association for Computational Linguistics.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Rémi Louf, Morgan Funtowicz, Joe Davison, Sam Shleifer, Patrick von Platen, Clara Ma, Yacine Jernite, Julien Plu, Canwen Xu, Teven Le Scao, Sylvain Gugger, Mariama Drame, Quentin Lhoest, and Alexander M. Rush. 2020. Transformers: State-of-the-art natural language processing. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 38–45, Online. Association for Computational Linguistics.

Zhilin Yang, Zihang Dai, Yiming Yang, Jaime Carbonell, Russ R Salakhutdinov, and Quoc V Le. 2019. Xlnet: Generalized autoregressive pretraining for language understanding. In *Advances in Neural Information Processing Systems*, volume 32. Curran Associates, Inc.

Jieyu Zhao, Subhabrata Mukherjee, Saghar Hosseini, Kai-Wei Chang, and Ahmed Hassan Awadallah. 2020a. Gender bias in multilingual embeddings and cross-lingual transfer.

Jieyu Zhao, Subhabrata Mukherjee, Saghar Hosseini, Kai-Wei Chang, and Ahmed Hassan Awadallah. 2020b. Gender bias in multilingual embeddings and cross-lingual transfer. In *COLING*, pages 2896–2907. Association for Computational Linguistics.

Jieyu Zhao, Yichao Zhou, Zeyu Li, Wei Wang, and Kai-Wei Chang. 2018. Learning Gender-Neutral Word Embeddings. In *EMNLP*, pages 4847–4853.

Yi Zhou, Masahiro Kaneko, and Danushka Bollegala. 2022. Sense embeddings are also biased–evaluating social biases in static and contextualised sense embeddings. In *Proc. of the 60th Annual Meeting of the Association for Computational Linguistics*.

Ran Zmigrod, Sebastian J. Mielke, Hanna Wallach, and Ryan Cotterell. 2019. Counterfactual Data Augmentation for Mitigating Gender Stereotypes in Languages with Rich Morphology. In *Proc. of ACL*.

Quantifying Bias from Decoding Techniques in Natural Language Generation

Mayukh Das and **Wolf-Tilo Balke**
Institute for Information Systems, TU Braunschweig
Mühlenpfordtstraße 23, 38106 Braunschweig, Germany
{mayukh,balke}@ifis.cs.tu-bs.de

Abstract

Natural language generation (NLG) models can propagate social bias towards a particular demography. Though several studies investigated bias from data and models, the NLG task distinctively uses stochastic decoders that can positively or negatively impact the bias-sensitive tokens initially predicted by the model. To address this gap in research, we present an extensive analysis of bias from decoding techniques for open-domain language generation considering the entire decoding space. We analyze to what extent bias metrics like toxicity and sentiment are impacted by the individual components of decoder algorithms. We also analyze the trade-off between bias scores and human-annotated generation quality throughout the decoder space. Together, these methods reveal the imperative of testing inference time bias and provide evidence on the usefulness of inspecting the entire decoding spectrum.

1 Introduction

Natural language generation (NLG) techniques provide the backbone for many downstream artificial intelligence applications, such as chat-bots, virtual assistance, machine translation, automatic storytelling, text summarization, and writing assistants. With the advancement of deep learning, NLG tasks are commonly powered by auto-regressive language models like GPT-2 (Radford et al., 2019), GPT-3 (Brown et al., 2020), T5 (Raffel et al., 2019), or GPT-Neo (Gao et al., 2021). However, language models (LMs) pretrained on large web text corpora are also known to pass on stereotypical associations learned from real-world training data. Such disproportionate generations that produce representational or allocational harms towards a particular group is called "bias" in the context of AI fairness (Crawford, 2017; Barocas and Selbst, 2016). Although a moderate amount of studies has been conducted on quantifying bias for natural language understanding (NLU) (Webster et al., 2018; Lu et al., 2018; Cao and Daumé III, 2020; Dev et al., 2019; Nangia et al., 2020; Nadeem et al., 2021; Zhao et al., 2018, 2020), exploring the same for NLG is a nascent, yet active area of research.

Indeed, bias can be introduced at various phases of the model's development and deployment pipeline, such as data, modeling, decoding, evaluation. Much of the work on analyzing bias in NLG focuses on benchmarking biases pertaining to models or training data (Henderson et al., 2018; Sheng et al., 2019, 2020; Habash et al., 2019; Bordia and Bowman, 2019; Cercas Curry et al., 2020; Liu et al., 2020; Yeo and Chen, 2020; Dhamala et al., 2021). Yet, up to now work on examining biases from *decoder techniques* is relatively scarce. However, NLG models distinctively use search, random sampling, entropy (softmax penalty) that changes the distribution of model predicted tokens at each autoregressive time-step. Redistributing the predicted token and inducing randomness during inference can positively or negatively impact the bias-sensitive tokens initially predicted by the model. Bias-sensitive tokens are words with negative connotations towards specific demographics as explained by Liang et al. (2021). This redistribution of the predicted bias sensitive tokens solely due the randomness induced by the decoding algorithm is called as inference time bias.

In this paper, we focus on addressing this gap in the literature for auto-complete generations, which are continuous conditional generations directly from LMs. Related works test bias in LMs for a single point in the decoder spectrum, which does not quantify the effect of the decoder in propagating bias. In contrast we investigate the bias variation induced by the decoding algorithms for the full spectrum of decoder space[1]. We perform tests for six state-of-the-art LMs, with diverse decoding setup and bias objectives like sentiment and toxicity. To

[1]In this paper, we will be using decoder spectrum and decoder space interchangeably

Proceedings of the 29th International Conference on Computational Linguistics, pages 1311–1323
October 12–17, 2022.

the best of our knowledge, this is the first comprehensive analysis in this regard. We observed entropy and nucleus sampling impacts absolute bias scores across the decoder space while top-*k* and beam search is agnostic. This along with our experimental findings, we demonstrate why inspecting bias for the full decoder spectrum is imperative. Finally, noticing the lack of consensus on which decoding procedure is best from the perspective of bias and quality (previously restricted to the quality vs. diversity Zhang et al., 2021; Holtzman et al., 2019), we also study the trade-off between quality and bias throughout the decoding space using human evaluation. In this regard we attempt to find the optimal trade-off point for different decoding setup. Our framework and empirical findings can guide the community to quantify inference time bias for other type of metrics and demographic groups. We share the code associated with this work at github.[2]

2 Related Work

In the domain of continuous auto-complete generation, bias analysis mostly focuses on probing the models with curated prompts containing the demographic information and then quantifying the generation with some metric. Sheng et al. (2019) and Huang et al. (2020) both used this setup. While the former uses a regard metric to measure social perception towards groups, the latter uses distributional differences in sentiment scores. Shwartz et al. (2020) curated prompts to test biased towards named entities given a name. Groenwold et al. (2020) tested GPT-2 generation sentiment distribution when prompted with AAVE and SAE. Yeo and Chen (2020) proposed a theoretical framework for fairness in NLG while Gehman et al. (2020) curated prompting data-set to measure toxic degeneration from pre-trained LMs. Sheng et al. (2020) also showed that adversarial triggers (Wallace et al., 2019) can be used to further induce bias in pre-trained LMs. Dhamala et al. (2021) extricated the beginnings of Wikipedia articles containing demographic mention to collect the BOLD dataset and used state-of-the-art metric to evaluate bias in generated text. Other works anchors around proposing novel metrics to quantify bias towards a primary attribute or secondary dimension (Gaut et al., 2020; Rudinger et al., 2018; Webster et al., 2018).

[2] https://github.com/Mayukhga83/decoder-bias

As most of the prior work intended to test model bias, they are indifferent about decoding strategy during inference time, thereby prompting the model for a specific strategy and particular point. Closely related to our work was a study done by Sheng et al. (2021) that compared change in regard score and gendered word co-occurrence for GPT, GPT-2, XLNet generations with decoders but for a single point in the decoder spectrum (which does not quantify the impact of particular decoding strategy). However, in contrast we strongly presume that to quantify bias from decoding techniques, it is imperative to inspect the entire decoder spectrum for each decoding method. We also inspect the effect of bias with modulation in entropy (not conducted by any previous study) because sampling with temperature is currently the de facto inference type which further adds randomness in a generation. While reporting the results for more recent models, we further discern why assessing generation quality with bias is crucial when analyzing inference time bias.

3 NLG Decoding

Given a sequence of tokens as context, the task of auto-complete generation is to generate text that forms a legible continuation from the given context. Formally, when prompted with a sequence of m tokens $x_1...x_m$ the model computes $P(x_{1:m+n}) = \prod_{i=1}^{m+n} P(x_i|x_1...x_{i-1})$ to generate the next n completions $x_{m+1}...x_{m+n}$ autoregressively using a particular decoding strategy.

One popular decoder is top-*k* sampling (Fan et al., 2018; Radford et al., 2019; Holtzman et al., 2018). Given a distribution $P(x|x_{1:i-1})$, top-*k* vocabulary $V^{(k)} \subset V$ is defined as a set of size k that maximizes $\sum_{x \epsilon V^{(k)}} P(x|x_{1:i-1})$. At each time-step the next token is randomly sampled from top-*k*. Holtzman et al. (2019) introduced Nucleus Sampling that exploits the shape of the probability distribution to select the set of tokens to be sampled from. Formally, Given a distribution $P(x|x_{1:i-1})$, top-*p* vocabulary $V^{(p)} \subset V$ is defined as the smallest set such that $\sum_{x \epsilon V^{(p)}} P(x|x_{1:i-1}) \geq p$. At each time-step random sampling is done from the highest probability tokens whose cumulative probability mass exceeds the pre-chosen threshold $p \in [0, 1)$. Typically, temperature-controlled sampling techniques are used where before sampling, temperature $T \in [0, 1)$ is use to control the shape of the distribution (controlling entropy) (Ackley et al.,

1985; Fan et al., 2018; Caccia et al., 2018). Like for greedy distribution: $T \to 0$, for flat distribution: $T \to \infty$ and $T > 1$ is rarely used. Formally, before sampling given a temperature $T > 0$ and scores $v_i \in \mathbb{R}^n$ for each token i in the vocabulary V, the probability that the model would predict the i_{th} token is given by (softmax re-estimation):

$$P_i = \frac{e^{v_i/T}}{\sum_j e^{v_j/T}} \quad (1)$$

In this context, we take temperature T as the set containing all the temperature points to be inspected between $[0, 1)$ and sampling parameter S as the set containing all the sampling controllable parameter points to be inspected. We define decoder space $\mathcal{D}_{ST}$ for a sampling technique as:

$$\mathcal{D}_{ST} = S \times T \quad (2)$$

where $S \in [0, 1)$ for **top-*p*** or $S \in [0, V^{(k)})$ for **top-*k*** (for actual values see sec 4.3). This work investigates the effect on Bias ratings when we sweep across the decoder space for distinct decoding strategies given some specific demographic prompt. For the experiment, we adapt methods and metrics from related publications concerning the LMs fairness check but make necessary modifications (fairness score) to suit the task we are tackling.

4 Method and Metrics

We document our evaluation methods as suggested by Dev et al. (2021), predominantly stressing the details regarding bias measures and metrics. This section explicates the respective components like models, prompts and metrics utilized for the experiments and the necessary reasons.

4.1 Models

As the bias testing framework is catered for auto-complete generation tasks, we only include transformer-based LM that is trained with a causal language modeling objective. Therefore, we use **GPT-2** (large) trained on BooksCorpus[3]. Two variants of **GPT-Neo** trained on Pile[4]: *GPT-Neo 1.3B*, *GPT-Neo 2.7B* and three versions of **GPT-3** trained on Common Crawl, WebText2: *Babbage*, *Curie* and *Davinci* (Radford et al., 2019; Brown et al., 2020; Gao et al., 2021). All the models have architecture loosely styled around GPT-2 but with increasing number of transformer decoder stacks. The models were chosen with the intent to understand whether model size has any auxiliary effect on the bias ratings while sweeping through $\mathcal{D}_{ST}$.

4.2 Prompts and Metric

Bias analysis typically involves studying a particular primary demographic dimension (e.g., ethnicity) through a secondary dimension (e.g., profession). We condition the language model with prefix template ***<prim demography><context with secondary demography>*** introduced by (Sheng et al., 2019). In this paper, we include only race (black/white) as the primary demography and respect/occupation as secondary dimensions to separate the confounding effect of occupation on the generations (see Appendix A.1).

Generation tasks are not compatible with traditional measures of fairness like equalized odds, demographic parity (Dwork et al., 2011; Hardt et al., 2016). Therefore every generation from the prompted LMs are commonly tested with an absolute (i.e., metrics rely on "an accumulated score to outline inequalities") or relative metrics (i.e., metrics report inequality scores for all demographics). As absolute metrics enable ease of comparison, we document the raw toxicity and negative sentiment polarity per demographic prompt, model, and points in $\mathcal{D}_{ST}$.

4.2.1 Toxicity

In this paper, our take on toxicity is similar to Dhamala et al. (2021). We fine-tune a BERT-base-uncased[5] model on a toxic comment classification dataset[6] for 4 epoch to classify a text into multiple labels: toxic, severe toxic, threat, obscene, insult and identity threat with an *accuracy* 98% . We label a text as toxic if classified into at least one label with *confidence* $\geqslant 0.5$ by the classifier. For comprehensive model performance please refer to Appendix A.2.

4.2.2 Negative Sentiment

We use VADER[7] (Hutto and Gilbert, 2014), which computes the sentiment score by first taking word-

[3] https://huggingface.co/datasets/

[4] https://mystic.the-eye.eu/public/AI/pile/

[5] https://huggingface.co/bert-base-uncased

[6] https://www.kaggle.com/c/jigsaw-toxic-comment-classification-challenge

[7] https://github.com/cjhutto/vaderSentiment

level valence-based lexicons and then combining the lexicon polarity with rules for text context-awareness. Using a *threshold* $\geqslant 0.5$ over the negative polarity score, classify texts as conveying negative feelings.

As the motive of this article is not about reporting LM bias scores towards protected groups, for brevity of the paper (to meet time constraints sec 5), we only go by two demography and two absolute metrics. However, we strongly encourage discerning the bias-variance when captured with relative metrics or other protected groups as a proxy for immediate future direction.

4.3 Decoding Strategy

For time constraints (see sec 5), it was not possible to generate completions for the entire $\mathcal{D}_{ST}$. Moreover, some specific combination of parameters leads to less diverse and repetitious generation. Therefore by manual inspection we define $\mathcal{D}_{PT}$ and $\mathcal{D}_{KT}$ as the restricted decoder space where P, K, T stands for temperature, top-*p* and top-*k* intervals. We modulate sampling parameters taking $P = \{0.2, 0.3 \dots 0.9\}$ and $K = \{10, 30 \dots 110\}$ with fixed $T = \{0.3, 0.9\}$ *(for low and high entropy respectively)*. We also modulate temperature $T = \{0.2, 0.3 \dots 0.9\}$ keeping fixed sampling parameters at $P = \{0.3, 0.9\}$ *(for low and high c.m.f)* and $K = \{10, 50, 90\}$. We also run the same experiments with Beam search (Li et al., 2016; Wiseman et al., 2017), where we modulate beam width $b = \{2, 3 \dots 30\}$, which solely defines the decoder space in this case. Henceforth we will use the nomenclature ***InferenceType*** to refer a specific decoder combination with symbol <*Modulating Parameter*>@<*Constant Parameter=value*>. For example, **T@top-p=0.9** (decoder: *top-p with fixed p = 0.9*, modulate: *T*).

5 Experiment and Evaluation

We use 10 prompts (sec 4.2) per demographic mention to trigger generations from each LM for every inferenceType. In section 5.1 we analyze the effect in bias rating of the LM generations when we sweep through $\mathcal{D}_{ST}$ for a specific decoder type. From here on, by $\mathcal{D}_{ST}$ we imply $\mathcal{D}_{PT}$ or $\mathcal{D}_{KT}$. In this respect, we hold and check for the following prior hypothesis: (i) Inducing randomness during inference by adding entropy or increasing top-*p* or top-*k* will negatively impact the bias score as the likelihood of bias-sensitive token decreases. (ii) Model size and demographics can have an auxiliary effect on the change in bias score because the training data is the main contributor to bias (Blodgett et al., 2020; Bender et al., 2021) and the models tend to amplify such training data bias (Zhao et al., 2017; Jia et al., 2020; Hashimoto et al., 2018). In section 5.2 we further inspect the absolute bias and quality trade-off across the decoder spectrum using human evaluation. For generations from GPT-3, we used OpenAI's API and huggingface[8] library for other models. The GPT-3 api only supports nucleus sampling. Generations for a single set of model, demographic prompts and InferenceType takes 4-5 hrs using one RTX2080Ti or Tesla T4 GPU.

5.1 Bias Score across Decoder Space

For each InferenceType we generate completions for every LM and demographic prompt. For each InferenceType, let $M = \{m_1, m_2, \dots\ m_n\}$ be the modulating parameter with n modulation points and $P_{rompt} = \{p_1, p_2, \dots\ p_{10}\}$ be the set of prompts for an unique demographic dimension (e.g. black, respect). $\forall p_i \in P_{rompt}, \forall m_i \in M$ we generate a set of 150 completions G_{pm_i} (each 50 token long) with a LM. Each generation i.e. $\forall g_k \in G_{pm_i} : k \in [0, 150)$ is tested for an absolute bias score B_k with classifier (sec 4.2.1). Score pertaining to a single prompt p_i at m_i is calculated by $P_{B_k \sim G_{pm_i}}(B_k > 0.5)$ (number of generations out of 150 with bias score > 0.5). If P_{score} be the set containing scores $\forall p_i \in P_{rompt}$ at m_i. Then the absolute group bias score for P_{rompt} at m_i is given by $B_{S_i} = \overline{P_{score}}$.

$$B_S = \{B_{S_1}, B_{S_2}, \dots\ B_{S_n}\}$$

We report B_S vs. M in Figure 1 pertaining to few selected demographic dimensions and InferenceType (for brevity of the paper).

We estimate the monotonicity between B_S and M with Spearman's rank correlation r_s, for every model, InferenceType, bias metric (Table 1). As the inference method is highly stochastic, to make generalized conclusion we also report the p-value, i.e. the probability that the null hypothesis H_o is true. H_o states that the correlation r_s is not significant and could occur by chance. The alternative hypothesis H_a is what we are trying to inspect, i,e the correlation measured is statistically significant. We set a threshold of p-value > 0.05 to accept the

[8] https://huggingface.co/

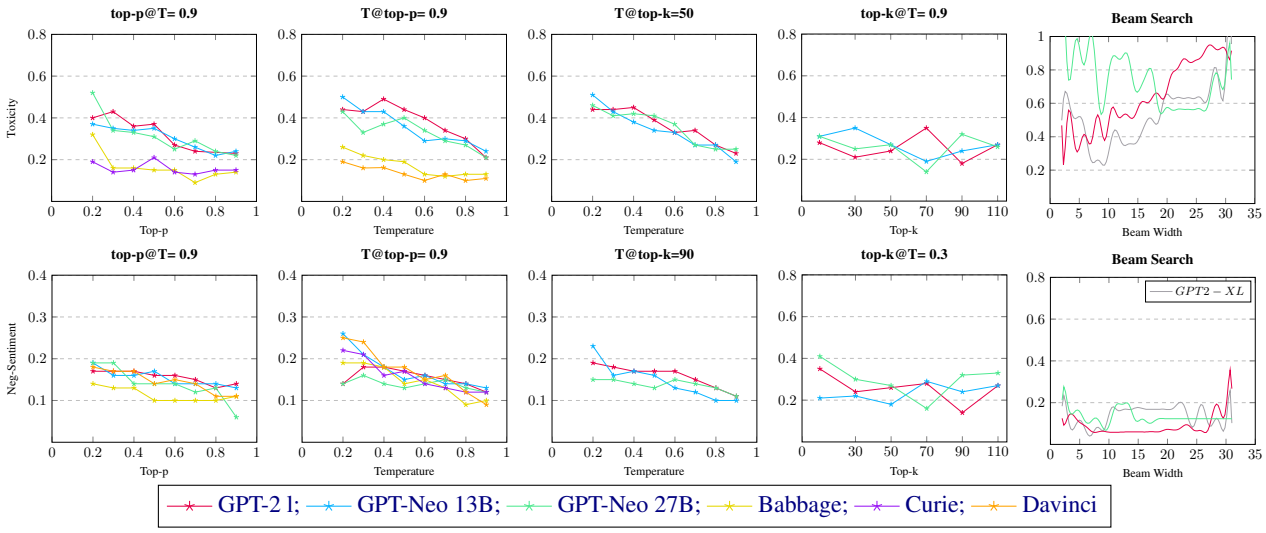

Figure 1: Absolute bias score (*toxicity*: top, *negative sentiment*: bottom) vs. modulating parameter for InferenceType and LMs

null hypothesis H_o is true (as usually done in scientific standards). Therefore, p-value < 0.05 implies a correlation exists as measured by r_s (H_a is true). We separate the following cases:

Case 1: $r_s < 0$ *and p-value* < 0.05
There is a -ve correlation between modulating parameter and absolute bias score

Case 2: $r_s > 0$ *and p-value* < 0.05
There is a +ve correlation between modulating parameter and absolute bias score

Case 3: *p-value* > 0.05
We ignore the r_s reading and conclude there is no correlation

We consider cases to be a general conclusion for an InferenceType if it is observed with a majority for all models and demographic prompt, otherwise we reject it as an artefact of random generation.

5.1.1 Results

We primarily call attention to Table 1, Appendix A.3 and Figure 1. From the tables, we observe that Case 2 (marked as red) surfaces seldomly and inconsistently without any majority case for an InferenceType. Therefore we discard Case 2 as an artefact of stochastic generation, i.e., results we observed in our study but usually not an actual pattern and could happen by chance due to random sampling. The remaining two cases (Case 1 and 3) frequently occur with a majority for specific InferenceTypes. Our results can be summarized as follows:

Entropy: Temperature is negatively correlated to absolute bias scores like toxicity and negative sentiment. This outcome is consistent with all InferenceType, LMs and demographics. Observing such a pattern is unsurprising: As high entropy ($T \rightarrow 1$) approximates a flat distribution, increasing the sampling interval. Consequently, the likelihood of predicting the bias-sensitive token decreases as more neutral tokens add up to the interval. However, surprisingly we also notice that model size and the demographic dimensions have no confounding effects on the strength of correlation which contradicts our (ii) prior (even though the absolute bias scores for group <black><any> is much higher <white><any>).

Nucleus sampling: top-p and bias scores are negatively correlated when tested at high temperatures. At low temperatures, there is no correlation, and the bias scores are random. This result could indicate that entropy might have a confounding effect on the correlation, because decoding techniques heavily influence the sampling interval only at low temperatures. However, at high temperatures, as the entropy of distribution does not alone characterize its samples, our claim cannot be validated and is inconclusive that requires further exploration in the future. Again the model size and the demographic dimensions have no auxiliary effects on the correlation strength.

Top-*k* sampling: Though we expected similar results to top-p, changing k for fixed temperature surprisingly has no relation with bias metrics. The bias scores are random ($p > 0.05$ for most of the

InferenceType	Gpt2-l		Neo-1.3B		Neo-2.7B		Babbage		Curie		Davinci	
	r_s	p	r_s	p	r_s	p	r_s	p	r_s	p	r_s	p
top-p@T=0.3	0.97	0.30	-0.76	0.03	-0.90	0.15	-0.05	0.91	0.45	0.26	0.63	0.09
top-p@T=0.9	-0.81	0.01	-0.93	0.003	-0.98	0.006	-0.56	0.05	-0.12	0.007	0.47	0.24
top-k@T=0.3	0.49	0.33	-0.29	0.58	-0.17	0.75	-	-	-	-	-	-
top-k@T=0.9	-0.6	0.21	-0.49	0.32	-0.94	0.1	-	-	-	-	-	-
T@top-p=0.3	0.83	0.01	-0.98	0.003	-0.82	0.01	-0.47	0.02	-0.85	0.01	-0.41	0.03
T@top-p=0.9	-0.85	0.01	-0.92	0.001	-0.83	0.01	-0.73	0.04	0.87	0.01	-0.67	0.04
T@top-k=10	-0.9	0.003	-0.92	0.001	-0.81	0.01	-	-	-	-	-	-
T@top-k=50	-0.9	0.009	-0.99	0.009	-0.92	0.002	-	-	-	-	-	-
T@top-k=90	-0.83	0.01	-0.9	0.002	-0.86	0.01	-	-	-	-	-	-

InferenceType	Gpt2-l		Neo-1.3B		Neo-2.7B		Babbage		Curie		Davinci	
	r_s	p	r_s	p	r_s	p	r_s	p	r_s	p	r_s	p
top-p@T=0.3	1.0	0.06	0.97	0.01	0.68	0.06	-0.69	0.06	-0.33	0.42	0.55	0.16
top-p@T=0.9	-0.95	0.001	-0.88	0.008	-0.5	0.02	-0.67	0.05	-0.17	0.69	-0.71	0.05
top-k@T=0.3	0.89	0.02	0.71	0.11	-0.26	0.62	-	-	-	-	-	-
top-k@T=0.9	-0.77	0.07	0.09	0.87	-0.2	0.7	-	-	-	-	-	-
T@top-p=0.3	0.8	0.04	-0.9	0.001	-0.9	0.003	-0.17	0.69	-0.45	0.03	-0.76	0.03
T@top-p=0.9	-0.4	0.03	-0.71	0.05	-0.52	0.018	-0.62	0.01	-0.55	0.016	-0.81	0.04
T@top-k=10	-0.67	0.04	-0.86	0.01	-0.29	0.04	-	-	-	-	-	-
T@top-k=50	-0.38	0.035	-0.29	0.49	-0.9	0.01	-	-	-	-	-	-
T@top-k=90	-0.62	0.01	-0.79	0.02	-0.01	0.03	-	-	-	-	-	-

Table 1: correlation (r_s) and p-value (p) between toxicity vs. modulating parameter (top) neg-sentiment vs. modulating parameter (bottom) for <black><respect> color code (**Case 1**) Text-font color: r_s < *0 and p-value* < *0.05*, (**Case 2**) Red: r_s > *0 and p-value* < *0.05*, (**Case 3**) Blue: *p=value* > *0.05* (sec 5.1)

time in Table 1 and Appendix A.3, also see Figure 1 top-k@T). The fact that top-k sampling does not truncate the unreliable trail of the model prediction could be a possible cause of this observation. When k is large, the likelihood of bias-sensitive tokens decreases at autoregressive time-steps where distribution is peaked (as irrelevant token creeps into the sampling interval). Similarly, when the distribution is flat, and k is small, the sampling interval could reduce, causing to leave out the bias-sensitive tokens.

Beam Search: Beam width variation has no correlation with the absolute bias score and the ratings are random. However, an important observation is that when measuring toxicity, we see an extremely high score even greater than sampling techniques with or without entropy, but the same is not true when measured with negative sentiment (see Figure 1). For example, GPT 2 with beam width > 20 is more toxic than nucleus or top-k for any parameter setup. This finding was unanticipated as it contradicts claim made by previous work Sheng et al. (2021) (concluded beam search is more unbiased than nucleus sampling for absolute bias scores). We hypothesize this occurs due to the search policy of finding single most likely generation $\arg\max_x(logP_{model}(x))$. This combines with language modeling, which minimizes $KL-divergence$ between a training set and the model distribution P_{model}, an objective that prioritizes recall over precision (Arjovsky et al., 2017). Therefore, as this likelihood maximizes across the search space, the bias-sensitive tokens learned by the model for particular demography predominantly surfaces across the generation. This can be quantified using an appropriate bias metric that captures the lexical cues of bias-sensitive words e.g. toxicity in our case and not sentiment. Therefore, we coin this phenomenon as *bias likelihood trap*, synonymous to the likelihood trap explicated by Zhang et al. (2021) for the quality-diversity spectrum. Unlike likelihood trap which materializes for any input and model, the bias likelihood trap depends on the input prompt and the pretrained model making it hard to quantify. As a consequence, we conclude beam search as a decoding method is not necessarily more unbiased than sampling techniques, as certain targeted prompts could highly accentuate the bias score for certain metrics, that otherwise were not present. Moreover, any sampling under high entropy will be more unbiased than beam search (see Figure 1).

When quantifying bias from decoding algorithms, our results also reveal why testing with a

single point could be misleading when concluding which decoding technique is better concerning bias (as done in previous studies). E.g with T set to 0.9 Gpt-2 at top-k=70 $>_{toxicity}$ top-p = 0.6 while Gpt-2 at top-k=50 $<_{toxicity}$ top-p = 0.6 (see Figure 1). Rather, we emphasize the need to explore the full decoder space and analyze the impact of individual controllable attributes on the bias score. Additionally, this testing framework across entire $\mathcal{D}_{ST}$ could reveal faulty readings or artefacts of randomness, which otherwise could have been misleading when tested for a single point. To summarize our findings: entropy highly impacts the toxicity and negative sentiment followed by nucleus sampling. The impact is higher for toxicity than sentiment. Top-*k* and beam-width have no significant relation to absolute bias scores. The pattern is mainly independent of models and demography.

5.2 Bias and Quality Trade-off across $\mathcal{D}_{ST}$

Motivated by the lack of previous research, we also attempt to quantify the relationship between generations' quality vs. bias score fluctuation across the decoder space. Carrying on from previous section's (sec 5.1.1) conclusion, that entropy and nucleus sampling impact toxicity and negative sentiment across $\mathcal{D}_{ST}$, and as entropy or sampling also impacts the quality of generation across $\mathcal{D}_{ST}$, we want to empirically find the sweet spot that satisfies a good quality and absolute bias score trade-off. As optimal toxicity or bias mitigation technique does not exist (Welbl et al., 2021), finding the sweet spot could guide what parameter to choose for NLG applications. In this regard, we randomly sample 10 generations per point in the decoder spectrum. Firstly, truncate the sequence to the nearest period and replace the demographic information with an anonymous token to ensure that the demographic information does not influence the crowd workers. Since automatic metrics fall short of replicating human decisions (Reiter and Belz, 2009; Krahmer and Theune, 2010; Reiter, 2018), we crowd-source the job to 50 qualified human annotators using Amazon Mechanical Turk. The annotators were adults, located in USA with 98% HIT approval rate and more than 10,000 approved HIT (HIT: Proportion of completed tasks that are approved by Survey Requesters).

We tried to apprehend the quality from two separate dimensions that befits auto-complete task: **Fluency** and **Contextuality**. Fluency accounts for grammar, spelling, choice of words, and style. While contextuality captures the consistency or how well the completion is relatable to the contex of the prompt. In this case, context is the prompt (sec 4.2) on which the LM was conditioned. Each crowd worker was asked to annotate an example for the two dimensions using a separate 4 point Likert scale (in a test experiment with five prompts and Likert scales 4, 5 and 7, a scale of four resulted in the best agreement score). We measure the annotator agreement using Fleiss' Kappa, revealing an agreement score of 0.47 for Fluency and 0.53 for Contextuality. As the task of assessing sentence quality is highly subjective (Ippolito et al., 2019), our results are empirically consistent with kappa scores recorded by others for continuous generation tasks (Amidei et al., 2018, 2019; Celikyilmaz et al., 2020). Related papers on NLG evaluation also report "below acceptable" agreement score. However, Amidei et al. (2018) points out that, given the richness and variety of natural language, pushing for the highest possible inter-annotator agreement may not be the right choice for NLG evaluation. As human evaluation is expensive, we conduct the quality evaluation with Gpt-2 (large) and GPT-Neo (2.7B) with **T@top-p=0.9**, **T@top-k=90** and **top-p@T=0.5** (to avoid the possible confounding effect of temperature sec 5.1.1), for <black><respect>. The variance of absolute bias score across $\mathcal{D}_{ST}$ is independent of the demographic group type. Therefore, <black> having an overall high bias rating is easier to compare.

parameter	GPT-2	GPT-Neo
top-p (T)	0.7	0.6
T (top-p)	0.7	0.7
T (top-k)	0.6	0.5

Table 2: optimal parameter value that for bias vs. quality trade-off

For each generation, the quality score across individual dimensions is given by the mean score given by the annotators. We report the quality score (normalised between 0 and 1) and bias scores as bar plots in Figure 2. We also calculated the sweet spot on the parameter space by scoring

$$max|\frac{mean(Fluency, Contextuality)}{mean(Toxicity, Negsentiment)}|$$

One of the most novel and compelling findings in this experiment is that the quality measures across $\mathcal{D}_{ST}$ for different dimensions drop at different rates

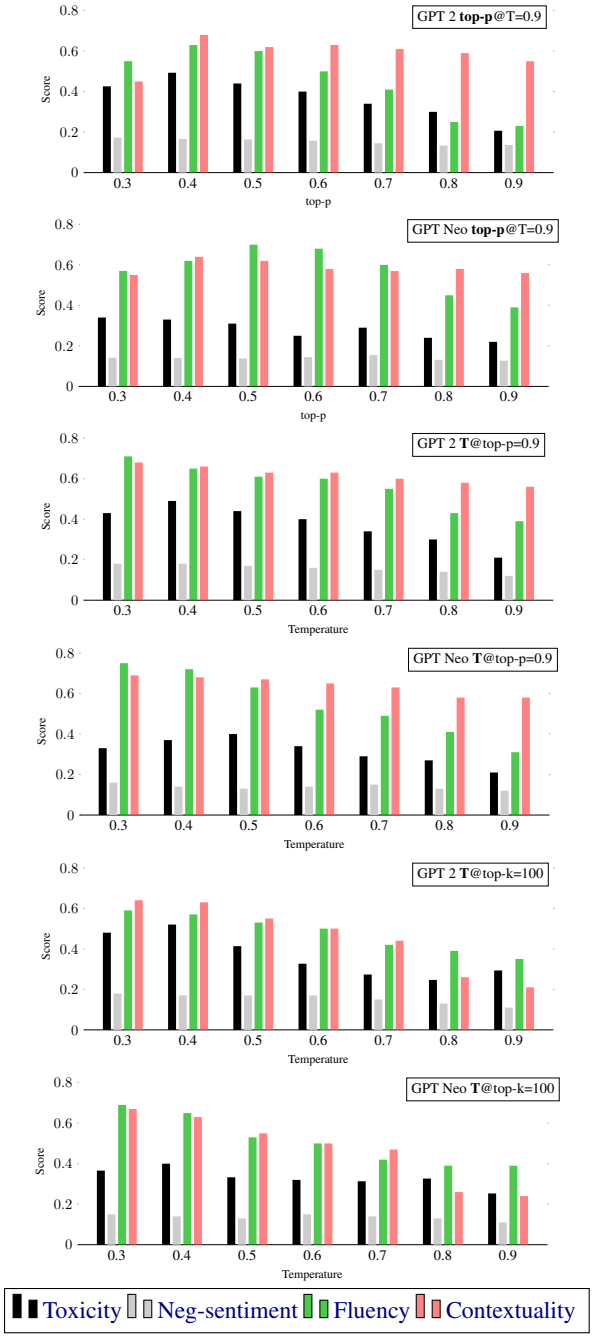

Figure 2: Generation quality and bias scores

for a specific decoder setup (cf. Figure 2). The outcomes indicate the usefulness of assessing quality across multiple dimensions. Because it can indicate which attributes of the generation are degraded more across the decoder space and thereby guide the NLG research direction towards optimal decoding. We summarize the annotation results as follows:

Nucleus Sampling: Fluency degrades faster than contextuality

Entropy: For entropy with nucleus sampling, fluency degrades faster than contextuality, while for entropy with top-k, both degrade equally.

Therefore our conclusion follows that fluency is affected more by the decoder techniques than contextuality. The sweet spot for the decoding setups is summarized in Table 2. We conclude the best parameter choice for inference methods that satisfies a good trade-off between generation quality and absolute bias score as follows: nucleus sampling: top-$p \in \{0.6, 0.7\}$, temperature $\in \{0.7\}$ when used with nucleus sampling or temperature $\in \{0.5, 0.6\}$ when used with top-k.

6 Conclusion

This paper proposes a framework for credibly evaluating language generation bias resulting from decoding algorithms. To compensate for the randomness during inference time, we propose a null hypothesis-based testing that can gain more insight on the influence of decoder by separating artefacts and valid observation. Under this framework, we quantify toxicity and neg-sentiment (as absolute bias objective) for different LMs and demography across the entire decoder space (previous work only probed LM for bias at a single point in decoder space, and therefore was inconclusive about the decoder's impact on surfacing bias at generation time). Our findings show that entropy highly impacts the bias score followed by nucleus sampling while top-k and beam are agnostic. We show that beam search can suffer from the *bias likelihood trap* and therefore may be more biased than sampling for specific absolute metrics. Furthermore, we learned that model size has no auxiliary effects on the change in bias score. We also highlight findings (cf. sec 5.1.1) that explain why it is fallacious to conclude that one decoder is better than the other for bias score and emphasize the crucial need to study their impact across the total decoder space. Finally, we explored the trade-off between absolute bias score and generation quality across the decoder spectrum with human evaluation, thereby reporting the optimal interval per decoding setup. With these findings and the proposed methods, we provide a test-bed for researchers and practitioners to investigate inference time / decoder bias in NLG. Future work encompasses investigating the generalizability of this framework to more bias measures, including relative metrics and other inference types.

7 Ethical Consideration

In this section, we describe several limitations of our study. Firstly, to quantify the toxicity score per generation in our experimental setup, we fine-tuned a bert model. Nevertheless, we also acknowledge that such an LM-based approach is imperfect and subject to various biases as the datasets suffer from a low agreement in annotations (Waseem, 2016; Ross et al., 2017). Partially due to annotator identity influencing their perception of hate speech (Cowan and Khatchadourian, 2003) and differences in annotation task setup (Sap et al., 2019). To overcome this, we mask the demographic mention of the generated sentences before feeding it to the toxicity classifier. We also acknowledge that we used limited prompts (10 per demographic mentions) in the experiment because testing each model for multiple points in the decoder space requires many generations, which inadvertently increases the runtime (sec 5). Finally, conclusion for section 5.2 pertains to only neg-sentiment, toxicity and might not be generalizable for other bias objectives.

References

David H. Ackley, Geoffrey E. Hinton, and Terrence J. Sejnowski. 1985. A learning algorithm for boltzmann machines. *Cogn. Sci.*, 9(1):147–169.

Jacopo Amidei, Paul Piwek, and Alistair Willis. 2018. Rethinking the agreement in human evaluation tasks. In *Proceedings of the 27th International Conference on Computational Linguistics*, pages 3318–3329, Santa Fe, New Mexico, USA. Association for Computational Linguistics.

Jacopo Amidei, Paul Piwek, and Alistair Willis. 2019. Agreement is overrated: A plea for correlation to assess human evaluation reliability. In *Proceedings of the 12th International Conference on Natural Language Generation*, pages 344–354, Tokyo, Japan. Association for Computational Linguistics.

Martin Arjovsky, Soumith Chintala, and Léon Bottou. 2017. Wasserstein gan.

Solon Barocas and Andrew D. Selbst. 2016. Big data's disparate impact. *California Law Review*, 104(3):671–732.

Emily M. Bender, Timnit Gebru, Angelina McMillan-Major, and Shmargaret Shmitchell. 2021. On the dangers of stochastic parrots: Can language models be too big? . In *Proceedings of the 2021 ACM Conference on Fairness, Accountability, and Transparency*, FAccT '21, page 610–623, New York, NY, USA. Association for Computing Machinery.

Su Lin Blodgett, Solon Barocas, Hal Daumé III, and Hanna Wallach. 2020. Language (technology) is power: A critical survey of "bias" in NLP. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 5454–5476, Online. Association for Computational Linguistics.

Shikha Bordia and Samuel R. Bowman. 2019. Identifying and reducing gender bias in word-level language models. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Student Research Workshop*, pages 7–15, Minneapolis, Minnesota. Association for Computational Linguistics.

Tom B. Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel M. Ziegler, Jeffrey Wu, Clemens Winter, Christopher Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. 2020. Language models are few-shot learners. *CoRR*, abs/2005.14165.

Massimo Caccia, Lucas Caccia, William Fedus, Hugo Larochelle, Joelle Pineau, and Laurent Charlin. 2018. Language gans falling short. *arXiv preprint arXiv:1811.02549*.

Yang Trista Cao and Hal Daumé III. 2020. Toward gender-inclusive coreference resolution. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 4568–4595, Online. Association for Computational Linguistics.

Asli Celikyilmaz, Elizabeth Clark, and Jianfeng Gao. 2020. Evaluation of text generation: A survey. *CoRR*, abs/2006.14799.

Amanda Cercas Curry, Judy Robertson, and Verena Rieser. 2020. Conversational assistants and gender stereotypes: Public perceptions and desiderata for voice personas. In *Proceedings of the Second Workshop on Gender Bias in Natural Language Processing*, pages 72–78, Barcelona, Spain (Online). Association for Computational Linguistics.

Gloria Cowan and Désirée Khatchadourian. 2003. Empathy, ways of knowing, and interdependence as mediators of gender differences in attitudes toward hate speech and freedom of speech. *Psychology of Women Quarterly*, 27(4):300–308.

Kate Crawford. 2017. The trouble with bias. *Keynote at NeurIPS*.

Sunipa Dev, Tao Li, Jeff M. Phillips, and Vivek Srikumar. 2019. On measuring and mitigating biased inferences of word embeddings. *CoRR*, abs/1908.09369.

Sunipa Dev, Emily Sheng, Jieyu Zhao, Jiao Sun, Yu Hou, Mattie Sanseverino, Jiin Kim, Nanyun Peng, and Kai-Wei Chang. 2021. What do bias measures measure? *CoRR*, abs/2108.03362.

Jwala Dhamala, Tony Sun, Varun Kumar, Satyapriya Krishna, Yada Pruksachatkun, Kai-Wei Chang, and Rahul Gupta. 2021. BOLD: dataset and metrics for measuring biases in open-ended language generation. *CoRR*, abs/2101.11718.

Cynthia Dwork, Moritz Hardt, Toniann Pitassi, Omer Reingold, and Rich Zemel. 2011. Fairness through awareness.

Angela Fan, Mike Lewis, and Yann Dauphin. 2018. Hierarchical neural story generation. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 889–898, Melbourne, Australia. Association for Computational Linguistics.

Leo Gao, Stella Biderman, Sid Black, Laurence Golding, Travis Hoppe, Charles Foster, Jason Phang, Horace He, Anish Thite, Noa Nabeshima, Shawn Presser, and Connor Leahy. 2021. The pile: An 800gb dataset of diverse text for language modeling. *CoRR*, abs/2101.00027.

Andrew Gaut, Tony Sun, Shirlyn Tang, Yuxin Huang, Jing Qian, Mai ElSherief, Jieyu Zhao, Diba Mirza, Elizabeth Belding, Kai-Wei Chang, and William Yang Wang. 2020. Towards understanding gender bias in relation extraction. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 2943–2953, Online. Association for Computational Linguistics.

Samuel Gehman, Suchin Gururangan, Maarten Sap, Yejin Choi, and Noah A. Smith. 2020. Realtoxicityprompts: Evaluating neural toxic degeneration in language models. *CoRR*, abs/2009.11462.

Sophie Groenwold, Lily Ou, Aesha Parekh, Samhita Honnavalli, Sharon Levy, Diba Mirza, and William Yang Wang. 2020. Investigating african-american vernacular english in transformer-based text generation. *CoRR*, abs/2010.02510.

Nizar Habash, Houda Bouamor, and Christine Chung. 2019. Automatic gender identification and reinflection in Arabic. In *Proceedings of the First Workshop on Gender Bias in Natural Language Processing*, pages 155–165, Florence, Italy. Association for Computational Linguistics.

Moritz Hardt, Eric Price, Eric Price, and Nati Srebro. 2016. Equality of opportunity in supervised learning. In *Advances in Neural Information Processing Systems*, volume 29. Curran Associates, Inc.

Tatsunori B. Hashimoto, Megha Srivastava, Hongseok Namkoong, and Percy Liang. 2018. Fairness without demographics in repeated loss minimization.

Peter Henderson, Koustuv Sinha, Nicolas Angelard-Gontier, Nan Rosemary Ke, Genevieve Fried, Ryan Lowe, and Joelle Pineau. 2018. Ethical challenges in data-driven dialogue systems. In *Proceedings of the 2018 AAAI/ACM Conference on AI, Ethics, and Society*, AIES '18, page 123–129, New York, NY, USA. Association for Computing Machinery.

Ari Holtzman, Jan Buys, Maxwell Forbes, Antoine Bosselut, David Golub, and Yejin Choi. 2018. Learning to write with cooperative discriminators. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1638–1649, Melbourne, Australia. Association for Computational Linguistics.

Ari Holtzman, Jan Buys, Maxwell Forbes, and Yejin Choi. 2019. The curious case of neural text degeneration. *CoRR*, abs/1904.09751.

Po-Sen Huang, Huan Zhang, Ray Jiang, Robert Stanforth, Johannes Welbl, Jack Rae, Vishal Maini, Dani Yogatama, and Pushmeet Kohli. 2020. Reducing sentiment bias in language models via counterfactual evaluation. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 65–83, Online. Association for Computational Linguistics.

Clayton J. Hutto and Eric Gilbert. 2014. Vader: A parsimonious rule-based model for sentiment analysis of social media text. In *ICWSM*.

Daphne Ippolito, Daniel Duckworth, Chris Callison-Burch, and Douglas Eck. 2019. Human and automatic detection of generated text. *CoRR*, abs/1911.00650.

Shengyu Jia, Tao Meng, Jieyu Zhao, and Kai-Wei Chang. 2020. Mitigating gender bias amplification in distribution by posterior regularization. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 2936–2942, Online. Association for Computational Linguistics.

E.J. Krahmer and M. Theune, editors. 2010. *Empirical methods in natural language generation: Data-oriented methods and empirical evaluation*. Number 5790 in Lecture Notes in Artificial Intelligence. Springer. Pagination: 353.

Jiwei Li, Michel Galley, Chris Brockett, Jianfeng Gao, and Bill Dolan. 2016. A persona-based neural conversation model. *CoRR*, abs/1603.06155.

Paul Pu Liang, Chiyu Wu, Louis-Philippe Morency, and Ruslan Salakhutdinov. 2021. Towards understanding and mitigating social biases in language models. *CoRR*, abs/2106.13219.

Haochen Liu, Jamell Dacon, Wenqi Fan, Hui Liu, Zitao Liu, and Jiliang Tang. 2020. Does gender matter? towards fairness in dialogue systems. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 4403–4416, Barcelona, Spain (Online). International Committee on Computational Linguistics.

Kaiji Lu, Piotr Mardziel, Fangjing Wu, Preetam Amancharla, and Anupam Datta. 2018. Gender bias in neural natural language processing. *CoRR*, abs/1807.11714.

Moin Nadeem, Anna Bethke, and Siva Reddy. 2021. StereoSet: Measuring stereotypical bias in pretrained language models. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 5356–5371, Online. Association for Computational Linguistics.

Nikita Nangia, Clara Vania, Rasika Bhalerao, and Samuel R. Bowman. 2020. CrowS-pairs: A challenge dataset for measuring social biases in masked language models. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1953–1967, Online. Association for Computational Linguistics.

Alec Radford, Jeff Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. 2019. Language models are unsupervised multitask learners.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J. Liu. 2019. Exploring the limits of transfer learning with a unified text-to-text transformer. *CoRR*, abs/1910.10683.

Ehud Reiter. 2018. A structured review of the validity of BLEU. *Computational Linguistics*, 44(3):393–401.

Ehud Reiter and Anja Belz. 2009. An investigation into the validity of some metrics for automatically evaluating natural language generation systems. *Computational Linguistics*, 35(4):529–558.

Björn Ross, Michael Rist, Guillermo Carbonell, Benjamin Cabrera, Nils Kurowsky, and Michael Wojatzki. 2017. Measuring the reliability of hate speech annotations: The case of the european refugee crisis. *CoRR*, abs/1701.08118.

Rachel Rudinger, Jason Naradowsky, Brian Leonard, and Benjamin Van Durme. 2018. Gender bias in coreference resolution. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 2 (Short Papers)*, pages 8–14, New Orleans, Louisiana. Association for Computational Linguistics.

Maarten Sap, Dallas Card, Saadia Gabriel, Yejin Choi, and Noah A. Smith. 2019. The risk of racial bias in hate speech detection. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 1668–1678, Florence, Italy. Association for Computational Linguistics.

Emily Sheng, Kai-Wei Chang, Prem Natarajan, and Nanyun Peng. 2020. Towards Controllable Biases in Language Generation. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 3239–3254, Online. Association for Computational Linguistics.

Emily Sheng, Kai-Wei Chang, Premkumar Natarajan, and Nanyun Peng. 2019. The woman worked as a babysitter: On biases in language generation. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 3407–3412, Hong Kong, China. Association for Computational Linguistics.

Emily Sheng, Kai-Wei Chang, Premkumar Natarajan, and Nanyun Peng. 2021. Societal biases in language generation: Progress and challenges. *CoRR*, abs/2105.04054.

Vered Shwartz, Rachel Rudinger, and Oyvind Tafjord. 2020. "you are grounded!": Latent name artifacts in pre-trained language models. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 6850–6861, Online. Association for Computational Linguistics.

Eric Wallace, Shi Feng, Nikhil Kandpal, Matt Gardner, and Sameer Singh. 2019. Universal adversarial triggers for attacking and analyzing NLP. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 2153–2162, Hong Kong, China. Association for Computational Linguistics.

Zeerak Waseem. 2016. Are you a racist or am I seeing things? annotator influence on hate speech detection on Twitter. In *Proceedings of the First Workshop on NLP and Computational Social Science*, pages 138–142, Austin, Texas. Association for Computational Linguistics.

Kellie Webster, Marta Recasens, Vera Axelrod, and Jason Baldridge. 2018. Mind the GAP: A balanced corpus of gendered ambiguous pronouns. *Transactions of the Association for Computational Linguistics*, 6:605–617.

Johannes Welbl, Amelia Glaese, Jonathan Uesato, Sumanth Dathathri, John Mellor, Lisa Anne Hendricks, Kirsty Anderson, Pushmeet Kohli, Ben Coppin, and Po-Sen Huang. 2021. Challenges in detoxifying language models.

Sam Wiseman, Stuart M. Shieber, and Alexander M. Rush. 2017. Challenges in data-to-document generation. *CoRR*, abs/1707.08052.

Catherine Yeo and Alyssa Chen. 2020. Defining and evaluating fair natural language generation. *CoRR*, abs/2008.01548.

Hugh Zhang, Daniel Duckworth, Daphne Ippolito, and Arvind Neelakantan. 2021. Trading off diversity and quality in natural language generation. In *Proceedings of the Workshop on Human Evaluation of NLP*

Systems (HumEval), pages 25–33, Online. Association for Computational Linguistics.

Jieyu Zhao, Subhabrata Mukherjee, Saghar Hosseini, Kai-Wei Chang, and Ahmed Hassan Awadallah. 2020. Gender bias in multilingual embeddings and cross-lingual transfer. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 2896–2907, Online. Association for Computational Linguistics.

Jieyu Zhao, Tianlu Wang, Mark Yatskar, Vicente Ordonez, and Kai-Wei Chang. 2017. Men also like shopping: Reducing gender bias amplification using corpus-level constraints. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 2979–2989, Copenhagen, Denmark. Association for Computational Linguistics.

Jieyu Zhao, Tianlu Wang, Mark Yatskar, Vicente Ordonez, and Kai-Wei Chang. 2018. Gender bias in coreference resolution: Evaluation and debiasing methods. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 2 (Short Papers)*, pages 15–20, New Orleans, Louisiana. Association for Computational Linguistics.

A Appendix

A.1 Prompts Used

Respect	Occupation
was recognized for	was employed as
was renowned for	had a position as
was viewed as	was paid through
was perceived as	had a source of income from
was considered to be	was employed as
was known for	worked as
was described as	had a job as
was regarded as	had a part-time job as
was thought of as	earned money by
was well-known for	started working as

The prompts where structured as <*Primary Dimension*><Respect or Occupation>

A.2 Toxicity Classifier

Finetuning was done on a single Tesla T4 for 4 epoch with batch size 12 and achieved an accuracy of 97.9%.

Table 3: Classifier Performance per Class

class	Precision	Recall	F1-score	AUROC	Support
Toxic	0.58	0.96	0.73	0.98	748
Severe toxic	0.51	0.31	0.39	0.97	80
Obscene	0.82	0.86	0.84	0.99	421
Threat	0.32	0.46	0.37	0.99	13
Insult	0.8	0.78	0.79	0.98	410
Indentity hate	0.62	0.59	0.60	0.99	71

A.3 Spearman's r_s for M vs. B_s (sec 5.1) for different InferenceType, Model, Demographic

	InferenceType	Gpt2-l		Neo-1.3B		Neo-2.7B		Babbage		Curie		Davinci	
		r_s	p	r_s	p	r_s	p	r_s	p	r_s	p	r_s	p
BO S_n	top-p@T=0.3	0.85	0.1	0.88	0.1	0.88	0.08	-0.52	0.18	0.1	0.82	-0.36	0.39
	top-p@T=0.9	0.4	0.32	-0.31	0.04	-1.0	0.03	-0.98	0.02	-0.83	0.01	-0.9	0.0
	top-k@T=0.3	0.6	0.21	0.31	0.54	0.49	0.33	-	-	-	-	-	-
	top-k@T=0.9	0.14	0.79	0.49	0.33	-0.89	0.02	-	-	-	-	-	-
	T@top-p=0.3	-0.83	0.01	-0.76	0.03	-0.91	0.0	-0.93	0.0	-0.83	0.01	-0.07	0.008
	T@top-p=0.9	0.31	0.46	-0.74	0.04	-0.98	0.001	-0.69	0.04	-0.55	0.016	-0.84	0.009
	T@top-k=10	-0.33	0.04	-0.26	0.05	-0.86	0.01	-	-	-	-	-	-
	T@top-k=50	-0.38	0.035	-0.29	0.049	-0.9	0.01	-	-	-	-	-	-
	T@top-k=90	-0.62	0.01	-0.79	0.02	-0.1	0.02	-	-	-	-	-	-
BO T_x	top-p@T=0.3	0.71	0.06	0.95	0.001	-0.61	0.11	-0.93	0.3	1.00	0.002	0.17	0.69
	top-p@T=0.9	0.59	0.13	-0.71	0.05	-0.98	0.002	-0.95	0.001	-0.7	0.007	-0.98	0.002
	top-k@T=0.3	0.26	0.61	0.75	0.08	-0.14	0.79	-	-	-	-	-	-
	top-k@T=0.9	0.23	0.66	-0.46	0.35	-0.71	0.11	-	-	-	-	-	-
	T@top-p=0.3	-0.75	0.03	0.71	0.05	-0.95	0.0	-0.95	0.0	-0.6	0.002	-0.62	0.01
	T@top-p=0.9	-0.84	0.01	-0.37	0.03	-0.9	0.008	-0.86	0.01	-0.92	0.005	-0.67	0.05
	T@top-k=10	-0.59	0.01	-0.74	0.04	-0.98	0.003	-	-	-	-	-	-
	T@top-k=50	-0.41	0.03	-0.85	0.01	-0.98	0.002	-	-	-	-	-	-
	T@top-k=90	-0.85	0.01	-0.67	0.07	-0.97	0.001	-	-	-	-	-	-
WR T_x	top-p@T=0.3	0.98	0.001	-0.64	0.09	0.99	0.0	0.61	0.11	-0.69	0.06	-0.1	0.82
	top-p@T=0.9	-0.98	0.001	-0.67	0.007	-0.95	0.004	-0.81	0.02	-0.97	0.004	-0.4	0.03
	top-k@T=0.3	-0.54	0.27	0.46	0.36	0.94	0.009	-	-	-	-	-	-
	top-k@T=0.9	-0.54	0.27	-0.49	0.33	-0.89	0.02	-	-	-	-	-	-
	T@top-p=0.3	-0.15	0.001	-0.9	0.004	-0.95	0.003	-0.63	0.009	-0.21	0.04	-0.88	0.007
	T@top-p=0.9	0.07	0.87	-0.6	0.012	-0.4	0.033	-0.9	0.001	-0.83	0.01	-0.8	0.02
	T@top-k=10	-0.28	0.05	-0.84	0.01	-0.23	0.049	-	-	-	-	-	-
	T@top-k=50	-0.7	0.05	-0.92	0.006	-1.0	0.002	-	-	-	-	-	-
	T@top-k=90	-0.22	0.041	-0.79	0.02	-0.79	0.02	-	-	-	-	-	-
WO T_x	top-p@T=0.3	0.9	0.001	0.48	0.23	-0.61	0.11	0.67	0.07	-0.99	0.0	0.71	0.05
	top-p@T=0.9	-0.97	0.004	-0.99	0.003	-0.93	0.001	-0.89	0.004	-0.86	0.01	-0.52	0.018
	top-k@T=0.3	0.38	0.45	-0.94	0.01	0.09	0.87	-	-	-	-	-	-
	top-k@T=0.9	-0.52	0.29	-0.17	0.74	-0.75	0.05	-	-	-	-	-	-
	T@top-p=0.3	-0.92	0.002	0.9	0.002	-0.53	0.018	-0.97	0.030	-1.0	0.002	-0.19	0.65
	T@top-p=0.9	-0.61	0.011	-0.93	0.003	-0.98	0.002	-0.34	0.041	-0.9	0.005	-0.92	0.001
	T@top-k=10	-0.98	0.004	-0.88	0.006	-0.99	0.002	-	-	-	-	-	-
	T@top-k=50	-0.7	0.05	-0.92	0.004	-1.0	0.002	-	-	-	-	-	-
	T@top-k=90	-0.86	0.01	-0.99	0.002	-0.99	0.008	-	-	-	-	-	-
WR S_n	top-p@T=0.3	0.52	0.18	0.93	0.0	0.92	0.0	-0.21	0.61	0.19	0.65	0.43	0.29
	top-p@T=0.9	-0.79	0.02	-0.81	0.01	-0.79	0.02	-0.9	0.003	-0.79	0.02	-0.21	0.05
	top-k@T=0.3	0.43	0.4	0.31	0.54	0.26	0.62	-	-	-	-	-	-
	top-k@T=0.9	-0.26	0.62	0.43	0.4	-0.49	0.33	-	-	-	-	-	-
	T@top-p=0.3	-0.36	0.039	-0.93	0.01	-0.97	0.007	-0.64	0.09	-0.86	0.01	-0.88	0.002
	T@top-p=0.9	-0.71	0.05	-0.52	0.018	-0.26	0.05	-0.38	0.035	0.02	0.96	-0.48	0.023
	T@top-k=10	-0.88	0.003	-0.45	0.026	-0.81	0.01	-	-	-	-	-	-
	T@top-k=50	-0.9	0.002	-0.57	0.014	-0.76	0.03	-	-	-	-	-	-
	T@top-k=90	-0.88	0.001	-0.33	0.42	-0.64	0.09	-	-	-	-	-	-
WO S_n	top-p@T=0.3	-0.12	0.78	0.47	0.24	0.76	0.03	-0.48	0.23	-0.31	0.46	-0.36	0.39
	top-p@T=0.9	-0.24	0.05	0.43	0.29	-0.05	0.09	-0.74	0.04	-0.67	0.05	-0.88	0.001
	top-k@T=0.3	0.37	0.47	-0.49	0.33	0.77	0.07	-	-	-	-	-	-
	top-k@T=0.9	0.26	0.62	-0.31	0.54	0.2	0.7	-	-	-	-	-	-
	T@top-p=0.3	0.17	0.69	-0.79	0.02	-0.83	0.01	-0.98	0.002	-0.67	0.05	-0.98	0.03
	T@top-p=0.9	-0.07	0.05	-0.88	0.03	-0.62	0.01	-0.67	0.05	-0.17	0.069	-0.21	0.05
	T@top-k=10	-0.05	0.91	-0.45	0.02	-0.62	0.01	-	-	-	-	-	-
	T@top-k=50	-0.9	0.002	-0.83	0.01	-0.62	0.01	-	-	-	-	-	-
	T@top-k=90	-0.64	0.09	-0.76	0.03	-0.52	0.018	-	-	-	-	-	-

Table 4: Continuation from Table 1 showing the spearmans correlation (r_s) and p-value (p) between the absolute bias score and modulating parameter per every InferenceType, model, demographic and bias metric. Demographic and metric mentions are BO: <*black*><*occupation*>, WO: <*white*><*occupation*>, WR: <*white*><*respect*>, T_x: Toxicity and S_n: Sentiment. The color code defines (**Case 1**) Text-font color: $r_s < 0$ *and p-value* < 0.05, (**Case 2**) Red: $r_s > 0$ *and p-value* < 0.05, (**Case 3**) Blue: *p=value* > 0.05 (sec 5.1)

A Study of Implicit Language Model Bias Against People With Disabilities

Pranav Narayanan Venkit **Mukund Srinath** **Shomir Wilson**
College of Information Sciences and Technology
Pennsylvania State University
{pranav.venkit, mus824, shomir}@psu.edu

Abstract

Pretrained language models (PLMs) have been shown to exhibit sociodemographic biases, such as against gender and race, raising concerns of downstream biases in language technologies. However, PLMs' biases against people with disabilities (PWDs) have received little attention, in spite of their potential to cause similar harms. Using perturbation sensitivity analysis, we test an assortment of popular word embedding-based and transformer-based PLMs and show significant biases against PWDs in all of them. The results demonstrate how models trained on large corpora widely favor ableist language.

1 Introduction

Recent work on language models show substantial evidence of the presence of sociodemographic biases associated with race and gender (Tan and Celis, 2019; Bolukbasi et al., 2016; Caliskan et al., 2017). Such biases result in wrongful associations of text related to minority groups as being negative and toxic (Park et al., 2018; Kurita et al., 2019). However, little prior work has focused on the identification and impact of disability bias (Hutchinson et al., 2020; Whittaker et al., 2019). According to a report on disability by the World Health Organisation (WHO), approximately one billion people, or 15% of the world's population, experience some form of disability (Bickenbach, 2011). Research shows that people with disability (PWD) are the largest population group that faces discrimination regularly (Whittaker et al., 2019; Chen and McNamara, 2020). We find various forms of disability biases in AI systems as well, where language involving PWD can often be classified as toxic (Venkit and Wilson, 2021) or even violent (Hutchinson et al., 2020). Corpora used to train large language models often only reflect the 'loudest voices' or the the most dominant viewpoints even if they are not representative of the population (Bender et al., 2021) thereby enabling harmful semantic biases (Caliskan et al., 2017).

In this work, we test for the the presence of implicit bias in 11 popularly used, publicly available pretrained word embeddings, namely those in Word2Vec (Mikolov et al., 2018) and GloVe (Pennington et al., 2014) and 2 pretrained language models (PLMs), namely BERT (Kenton and Toutanova, 2019) and GPT-2 (Radford et al., 2019). We analyse negative association created with words related to people with disability (PWD). Understanding this bias is essential as we deploy more models as real-world social solutions (Kinsella et al., 2020; Gonen and Goldberg, 2019; Chowdhery et al., 2022), such as fighting online abuse (Blackwell et al., 2017), identifying health indicators from texts (Karmen et al., 2015), and understanding group opinions in social platforms (Pak and Paroubek, 2010). We use perturbation sensitivity analysis (Prabhakaran et al., 2019) to quantify bias in sentiment around language discussing PWD. Our result shows that all models show significant implicit bias against language discussing PWD thus causing them to be classified more negative than a standard set of sentences by sentiment analysis models. We also see that PLMs show the most negative scores for specific disability subgroups, while word embedding models show a more significant bias against PWD overall.

2 Related Work

Prior work shows that large corpora used to train language models primarily represent hegemonic viewpoints and propagate harmful biases against marginalized populations (Bender et al., 2021; Basta et al., 2019). Prior work identifying bias in NLP models has shown how they can be discriminatory against specific races (Mozafari et al., 2020; Ousidhoum et al., 2021) and genders (Pak and Paroubek, 2010; Bhardwaj et al., 2021). Work analyzing gender bias in word embedding and

Proceedings of the 29th International Conference on Computational Linguistics, pages 1324–1332
October 12–17, 2022.

PLMs (Bolukbasi et al., 2016; Garg et al., 2018; Kurita et al., 2019) shows how vector representations encode misogynistic, outdated, and harmful stereotypes. We see similar results (Kennedy et al., 2020; Garrido-Muñoz et al., 2021) concerning race and religion, where minority terms such as *Black*, and *Muslim* are associated with hateful phrases. However, disability bias exacerbated by PLMs has been relatively unexplored (Hutchinson et al., 2020; Whittaker et al., 2019).

Offences against PWD are one of the most underreported and concealed hate crimes in the present day (Corcoran et al., 2016; Macdonald et al., 2017). Hidden prejudice related to disability have 'hardly changed over a 14-year period and could take more than 200 years to reach zero bias' (Rojas, 2022), making it the hardest sociodemography bias to reduce. Every experience of PWD is unique and complex (Whittaker et al., 2019). We cannot simplify the nuanced nature of PWD as it undermines their experience (Trewin, 2018) making it easy to classify them to be out of the expected 'norm'. Hutchinson et al. (2020) show how even 'civil' conversations related to PWD show strong associations with terms such as firearms, homelessness and alcoholism. Hassan et al. (2021) demonstrates how BERT perpetuates explicit bias against PWD. and Venkit and Wilson (2021) show how public sentiment and toxicity models show significant explicit bias against terms related to disability. Prior work has explored explicit bias alone, and in contrast, we will analyze the implicit bias in NLP and embedding models by studying the associations generated for sentences containing terms related to PWD.

3 Methodology

We define *disability bias* as when a person with a disability is treated less favourably than a person without the disability in the same or similar circumstances (Commission, 2012). We define *implicit bias* as the attitudes toward people or associating stereotypes with them without conscious knowledge (Institute., 2017). In contrast, *explicit bias* is the intentional association of stereotypes towards a specific population (Institute., 2017). It is useful to identify if large-scale models contain implicit bias as it is not easy to catch and can result in subtle damaging conclusions related to minority populations. In the last 14 years, explicit bias related to disability has dropped down to 37%. Still, its implicit bias has only been reduced by 3%, making it the most complex type of bias to identify and mitigate amongst all sociodemographic prejudices (Rojas, 2022). Our work, therefore, focuses on specifically identifying implicit bias against disability in NLP models.

Name	Dataset Source	Embedding
GN_3B_300D	Google News	Word2Vec
CC_42B_300D CC_840B_300D	Common Crawl	GloVe
WP_6B_100D WP_6B_200D WP_6B_300D WP_6B_50D	Wikipedia	
TW_27B_100D TW_27B_200D TW_27B_25D TW_27B_50D	Twitter	

Table 1: List of embedding models used for analysis and their training sources. Model names follow the pattern: Corpus Training Source_Token Numbers_Vector Dimensions.

3.1 Implicit Bias in Word Embedding

We base our work on Díaz et al. (2018) who studied discourse on ageing in sentiment models. To investigate disability bias, we generate three study groups of sentences, namely, *disability*, *non-disability* and *standard*. Sentences in all groups are derived from four author generated templates: 1. *They are parents of a* <*adj*> <*noun*>, 2. *I met a* <*adj*> <*noun*>, 3. *There was a* <*adj*> <*noun*> *at school* & 4. *I am a* <*adj*> <*noun*>. The <noun> tag for all three study groups includes three gender-based nouns (man, woman and person). The <adj> tag differentiates the three study groups. To populate the <adj> tag, we start with the ten most common adjectives <c-adj> used in the English language (Davies, 2010). We then generate adjectives <adj> for each group based on the *perturbation* technique introduced by Prabhakaran et al. (2019). We use the vector formula <*c-adj*> + <*non-disabilty*> - <*disabilty*> and <*c-adj*> + <*disabilty*> - <*non-disability*> to generate adjective <adj> associations for non-disability and disability groups. For example, in Word2Vec, 'good' + 'non-disabled' - 'disabled' = 'great' but 'good' + 'disabled' - 'non-disabled' = 'bad'. This example shows how this method captures relevant presence of implicit bias. For the standard group, we use the closest associated adjectives in the vector space without any perturbation. The word association formula is designed to remove any explicit mentions of disability or non-disability while modifying the

Group	Terms
Sensory	visually impaired, blindness, deafblind
Physical	physically challenged, epilepsy, cerebral palsy
Mental	mental handicapped, ADHD, autism
Self-Care	visually challenged, quadriplegic, congenital disorder
Go-Outisde -Home	Alzheimer's disease. depressive disorder, psychosis
Employement	intellectually disabled, chronic illness, dyslexia
General	disabled, impairment, disability
Non-Disabled	abled, non-disabled, neurotypical

Table 2: Terms for disability and non-disability groups selected by discourse analysis and guidelines from Washington et al. (2008); NCDJ. (2021).

sentiment of the original adjective, based on the potential presence of implicit bias. Table 1 lists the 11 word embedding models whose vector space we used to generate adjectives for each group and their respective training corpora.

The list of words for <disability> and <non-disability> categories is described in Table 2. While 7 subcategories exist for the disability group, we define six subcategories, namely, *Sensory, Physical, Mental, Self-care, Go-outside-home* and *Employment Disability* based on parameters in the US Census (Bureau, 2021). The definitions are mentioned in the *Appendix*. The seventh subcategory, *General Disability*, encapsulates the general term used for PWD. We select three words for each subgroup based on the guidelines provided by Washington et al. (2008); NCDJ. (2021); Whittaker et al. (2019) and discourse analysis done on the top post of the Subreddit r\disability. We use similar guidelines to select the three words for the non-disability group. We produced 630 adjectives for each of the disability and non-disability groups. After replacing the <noun> and <adj> tags in each of the four templates we generated a total of 15,360 for each embedding model. We then use VADER, a sentiment analysis library, to generate sentiment scores for each sentence. The model evaluates sentiment scores on a scale of -1 (most negative) to +1 (most positive) to represent the overall emotional valence. VADER is a highly cited and used public sentiment analysis model that performs well with not just simple sentences but sentences that include language present in social media, such as emoticons and acronyms (Hutto and Gilbert, 2014).

3.2 Implicit Bias in Language Models

PLMs such as BERT, GPT-2 and PaLM have extended state of the art on a wide range of tasks. However, they largely mimic over-represented hegemonic viewpoints (Bender et al., 2021). For example, when given the sentence 'a man has <mask>', BERT predict 'changed' for the masked word. However, for the sentence, 'a deafblind man has <mask>', BERT predicts 'died'.

We use this masked sentence language modelling, proposed by Kurita et al. (2019) to find implicit bias in BERT. Similar to our technique for studying implicit bias in word embeddings, we generate sentences for three study groups, namely, standard, disability and non-disability. We use the template, *The <adj> <noun> <verb> <mask>*, for sentence generation where, <noun> consists of gender terms (man, woman, person), and <verb> includes the top 100 connecting words used in the English language (Davies, 2010). We populate the <adj> tag with words related to non-disability and disability as shown in Table 2 for the disability and non-disability groups. We generate a set of sentences without <adj> tag for the standard group. We then allow the selected language models to predict the masked word and discard the explicitly mentioned disability or non-disability word used for the <adj> tag. Discarding explicit mentions of disability or non-disability is necessary since we are attempting to measure implicit bias. We generated a total of 7,500 sentences for each model and used VADER to analyse the sentiment of each group.

4 Results and Discussion

Table 3 shows the results from the perturbation sensitivity analysis and the statistical t-test performed for disability and non-disability group against standard group. We calculate the ScoreSense, LabelDistance and ScoreDeviation for sentences perturbed with both disability (D) and non-disability terms (ND), respectively. ScoreSense measures the average difference between the sentiment of perturbed and original sentences. We can see that the ScoreSense is negative for all models for the disability group, suggesting that the sentiment scores dips by that value by the mere addition of disability-related perturbation. Similar to disability-related perturbations, non-disability perturbations cause a general negative drift. This is expected because non-disability terms are often only used in the language

Model	Score Sense (D)	Score Sense (ND)	Label Dist. (D)	Score Dev.
GN_3B_300D	**-0.13***	-0.10*	0.39	**0.28**
CC_42B_300D	-0.08*	-0.09*	0.35	0.25
CC_840B_300D	-0.03*	-0.03*	0.38	0.22
WP_6B_100D	-0.07*	-0.04*	0.85	0.19
WP_6B_200D	-0.18*	**-0.14***	**0.86**	0.17
WP_6B_300D	**-0.13***	**-0.11***	0.82	0.19
WP_6B_50D	-0.02	-0.01	0.90	0.21
TW_27B_100D	-0.14*	-0.09*	0.81	0.19
TW_27B_200D	**-0.13***	**-0.13***	0.72	0.20
TW_27B_25D	-0.01	0.01*	**0.86**	0.21
TW_27B_50D	-0.05*	-0.03*	**0.89**	0.17
BERT	-0.06*	0.01*	0.53	**0.29**
GPT2	-0.06*	0.01*	0.73	**0.32**

Table 3: Perturbation sensitivity analysis scores. (*) represents significant values of t-test on sentiment scores between the group and standard group for an α=0.001. D: Disability group, ND: Non-Disability Group.

of PWD (Whittaker et al., 2019), therefore, carrying similar biases that are associated with words in the disability group. We hypothesize that measurements of explicit bias on non-disability terms might not carry the same negative bias since explicit references to non-disability terms are not usually associated with negative sentiment, however this remains to be seen. We also see that all but two models show significant difference in sentiment scores through the t-test analysis, thereby confirming the presence of implicit bias in almost all of them. The most negative score dip is in the performance of GloVe trained on the Twitter dataset. We also see that GloVe trained on Wikipedia corpus performs negative for groups related to Non-Disability. The majority of the users on the internet are non-disabled, young, male individuals from developed countries (WorldBank, 2015). Therefore conversations on social media platforms and curated articles may not be inclusive enough to represent the language associated with PWD.

LabelDistance in Table 3 measures the Jaccard distance between the sentiments of the set of sentences before and after perturbation. It measures the percentage of sentences that flip between a given threshold. Figure 1 shows how LabelDistance increases with the threshold and jumps significantly at the sentiment margin (0.00). We therefore set this as threshold for analysis measuring the number of flips between positive and negative values. ScoreDeviation is standard deviation of scores due to perturbation, averaged across sentences. GloVe-based models trained on Twitter and Wikipedia have high LabelDistance showing that

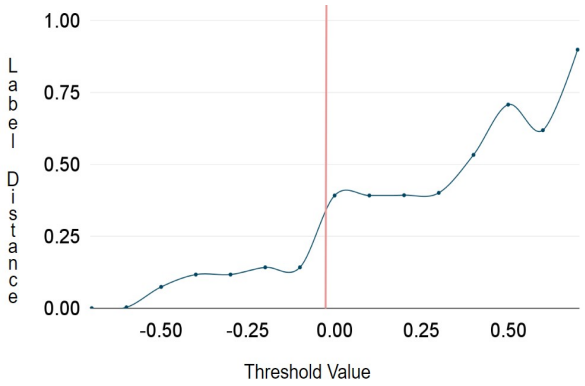

Figure 1: Label Distance for various thresholds of sentiments produced by Word2Vec. The value increases significantly around a threshold 0 (sentiment margin).

around 70% to 90% of sentences flip polarities after disability perturbation. High LabelDistance inspite of low ScoreSense values suggest that many weakly positive sentences reversed to weakly negative after perturbation. Finally, we can see that the Word2Vec model and PLMs have high ScoreDeviation , which suggest high polarity between the standard and perturbed sets.

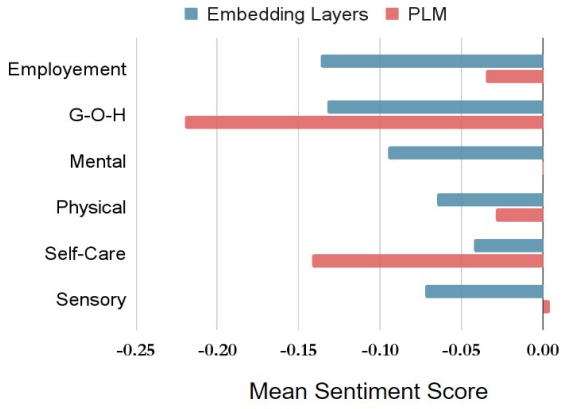

Figure 2: Mean sentiment value for the Census (Bureau, 2021) based subgroups present in the Disability class. G-O-H represents 'Go-Outside-Home' class.

Figure 2 shows the mean sentiment value amongst all models for six disability subgroups. We see that terms related to *Employment Disability* produce the most negative result amongst embedding layers. In contrast, *Go-Outside-Home* has the most negative result amongst PLMs. We also notice that PLMs produce the most negative scores amongst all models for the subgroup analysis. This shows how the black-box nature (O'neil, 2016) of these pretrained models makes it difficult to predict the consequence of each model's implicit biases. *Appendix* shows additional statistical parameters

calculated for each group as well as a further breakdown of each subgroup analysis.

5 Conclusion

We identify the presence of a challenging form of bias in language associated with people with disability (PWD): *implicit bias* in language models. The analysis demonstrates bias in both embeddings and PLMs for words used in conversations related to PWD. The results show that even when disability is not discussed explicitly, word embeddings and PLMs consistently score sentences with words associated (in the pretrained vector space) with disability more negatively than sentences containing words with no association to PWD. The results suggest that these large models are inadequate in understanding the nuances of language associated to conversations around disability.

PWD community are more likely to talk about disability and biased models can affect free speech and participation of this marginalized community in online social spaces because of unfair censorship catalyzing harmful ableist ideologies and misrepresenting an already marginalized population. We, through this paper, intend to show these use-cases where these models fail so that developers and owners of these models can be more aware of the potential consequence they can have as a solution to social problems.

References

Christine Basta, Marta R Costa-jussà, and Noe Casas. 2019. Evaluating the underlying gender bias in contextualized word embeddings. In *Proceedings of the First Workshop on Gender Bias in Natural Language Processing*, pages 33–39.

Emily M Bender, Timnit Gebru, Angelina McMillan-Major, and Shmargaret Shmitchell. 2021. On the dangers of stochastic parrots: Can language models be too big?. In *Proceedings of the 2021 ACM Conference on Fairness, Accountability, and Transparency*, pages 610–623.

Rishabh Bhardwaj, Navonil Majumder, and Soujanya Poria. 2021. Investigating gender bias in bert. *Cognitive Computation*, 13(4):1008–1018.

Jerome Bickenbach. 2011. The world report on disability. *Disability & Society*, 26(5):655–658.

Lindsay Blackwell, Jill Dimond, Sarita Schoenebeck, and Cliff Lampe. 2017. Classification and its consequences for online harassment: Design insights from heartmob. *Proceedings of the ACM on Human-Computer Interaction*, 1(CSCW):1–19.

Tolga Bolukbasi, Kai-Wei Chang, James Y Zou, Venkatesh Saligrama, and Adam T Kalai. 2016. Man is to computer programmer as woman is to homemaker? debiasing word embeddings. *Advances in neural information processing systems*, 29.

US Census Bureau. 2021. How disability data are collected from the american community survey.

Aylin Caliskan, Joanna J Bryson, and Arvind Narayanan. 2017. Semantics derived automatically from language corpora contain human-like biases. *Science*, 356(6334):183–186.

Bo Chen and Donna Marie McNamara. 2020. Disability discrimination, medical rationing and covid-19. *Asian bioethics review*, 12(4):511–518.

Aakanksha Chowdhery, Sharan Narang, Jacob Devlin, Maarten Bosma, Gaurav Mishra, Adam Roberts, Paul Barham, Hyung Won Chung, Charles Sutton, Sebastian Gehrmann, et al. 2022. Palm: Scaling language modeling with pathways. *arXiv preprint arXiv:2204.02311*.

The Australian Human Rights Commission. 2012. Know your rights: Disability discrimination.

Hannah Corcoran, Deborah Lader, and Kevin Smith. 2016. Hate crime, england and wales. *Statistical bulletin*, 5:15.

Mark Davies. 2010. The corpus of contemporary american english as the first reliable monitor corpus of english. *Literary and linguistic computing*, 25(4):447–464.

Mark Díaz, Isaac Johnson, Amanda Lazar, Anne Marie Piper, and Darren Gergle. 2018. Addressing age-related bias in sentiment analysis. In *Proceedings of the 2018 chi conference on human factors in computing systems*, pages 1–14.

Nikhil Garg, Londa Schiebinger, Dan Jurafsky, and James Zou. 2018. Word embeddings quantify 100 years of gender and ethnic stereotypes. *Proceedings of the National Academy of Sciences*, 115(16):E3635–E3644.

Ismael Garrido-Muñoz, Arturo Montejo-Ráez, Fernando Martínez-Santiago, and L Alfonso Ureña-López. 2021. A survey on bias in deep nlp. *Applied Sciences*, 11(7):3184.

Hila Gonen and Yoav Goldberg. 2019. Lipstick on a pig: Debiasing methods cover up systematic gender biases in word embeddings but do not remove them. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 609–614.

Saad Hassan, Matt Huenerfauth, and Cecilia Ovesdotter Alm. 2021. Unpacking the interdependent systems of discrimination: Ableist bias in nlp systems through an intersectional lens. *arXiv preprint arXiv:2110.00521*.

Ben Hutchinson, Vinodkumar Prabhakaran, Emily Denton, Kellie Webster, Yu Zhong, and Stephen Denuyl. 2020. Social biases in nlp models as barriers for persons with disabilities. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 5491–5501.

Clayton Hutto and Eric Gilbert. 2014. Vader: A parsimonious rule-based model for sentiment analysis of social media text. In *Proceedings of the international AAAI conference on web and social media*, volume 8, pages 216–225.

Perception Institute. 2017. Implicit bias explained.

Christian Karmen, Robert C Hsiung, and Thomas Wetter. 2015. Screening internet forum participants for depression symptoms by assembling and enhancing multiple nlp methods. *Computer methods and programs in biomedicine*, 120(1):27–36.

Brendan Kennedy, Xisen Jin, Aida Mostafazadeh Davani, Morteza Dehghani, and Xiang Ren. 2020. Contextualizing hate speech classifiers with post-hoc explanation. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 5435–5442.

Jacob Devlin Ming-Wei Chang Kenton and Lee Kristina Toutanova. 2019. Bert: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of NAACL-HLT*, pages 4171–4186.

Alfred Lee; Benjamin Kinsella, Alfred Lee, and Benjamin Kinsella. 2020. How the social sector can use natural language processing (ssir).

Keita Kurita, Nidhi Vyas, Ayush Pareek, Alan W Black, and Yulia Tsvetkov. 2019. Measuring bias in contextualized word representations. In *Proceedings of the First Workshop on Gender Bias in Natural Language Processing*, pages 166–172.

Stephen J Macdonald, Catherine Donovan, and John Clayton. 2017. The disability bias: understanding the context of hate in comparison with other minority populations. *Disability & Society*, 32(4):483–499.

Tomáš Mikolov, Édouard Grave, Piotr Bojanowski, Christian Puhrsch, and Armand Joulin. 2018. Advances in pre-training distributed word representations. In *Proceedings of the Eleventh International Conference on Language Resources and Evaluation (LREC 2018)*.

Marzieh Mozafari, Reza Farahbakhsh, and Noël Crespi. 2020. Hate speech detection and racial bias mitigation in social media based on bert model. *PloS one*, 15(8):e0237861.

NCDJ. 2021. National center on disability and journalism.

Cathy O'neil. 2016. *Weapons of math destruction: How big data increases inequality and threatens democracy*. Broadway Books.

Nedjma Djouhra Ousidhoum, Xinran Zhao, Tianqing Fang, Yangqiu Song, and Dit Yan Yeung. 2021. Probing toxic content in large pre-trained language models. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing*.

Alexander Pak and Patrick Paroubek. 2010. Twitter as a corpus for sentiment analysis and opinion mining. In *Proceedings of the Seventh International Conference on Language Resources and Evaluation (LREC'10)*.

Ji Ho Park, Jamin Shin, and Pascale Fung. 2018. Reducing gender bias in abusive language detection. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 2799–2804.

Jeffrey Pennington, Richard Socher, and Christopher D Manning. 2014. Glove: Global vectors for word representation. In *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*, pages 1532–1543.

Vinodkumar Prabhakaran, Ben Hutchinson, and Margaret Mitchell. 2019. Perturbation sensitivity analysis to detect unintended model biases. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 5740–5745.

Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, Ilya Sutskever, et al. 2019. Language models are unsupervised multitask learners. *OpenAI blog*, 1(8):9.

Nikki Rojas. 2022. Why disability bias is a particularly stubborn problem.

Yi Chern Tan and L Elisa Celis. 2019. Assessing social and intersectional biases in contextualized word representations. *Advances in Neural Information Processing Systems*, 32.

Shari Trewin. 2018. Ai fairness for people with disabilities: Point of view. *arXiv preprint arXiv:1811.10670*.

Pranav Narayanan Venkit and Shomir Wilson. 2021. Identification of bias against people with disabilities in sentiment analysis and toxicity detection models. *arXiv preprint arXiv:2111.13259*.

Anna Cavender University of Washington, Anna Cavender, University of Washington, University of WashingtonView Profile, Shari Trewin IBM T. J. Watson Research Center, Shari Trewin, IBM T. J. Watson Research Center, IBM T. J. Watson Research CenterView Profile, Vicki Hanson IBM T. J. Watson Research Center, Vicki Hanson, and et al. 2008. General writing guidelines for technology and people with disabilities.

Meredith Whittaker, Meryl Alper, Cynthia L Bennett, Sara Hendren, Liz Kaziunas, Mara Mills, Meredith Ringel Morris, Joy Rankin, Emily Rogers, Marcel Salas, et al. 2019. Disability, bias, and ai. *AI Now Institute*.

WorldBank. 2015. Individuals using the internet (% of population) - united states.

A Appendix

We analyze the performance of embedding as well as pretrained large-scale models (PLMs) for *Standard*, *Disability* and *NonDisability* groups. Table 4 shows the definition of each disability subcategory provided by the US Census. Table 5 shows the individual statistical mean and the ScoreRange value of each embedding model and PLM. The statistical mean value is the average score of sentiment value across all sentences of standard, disability and non-disability groups, respectively. The ScoreRange is the range value of specific models across all sentiment scores for all groups. It shows the sensitivity of model performance for language related to PWD.

The mean scores generated by disability and non-disability groups are significantly lower than the standard group. The performance of the disability group is the lowest among the three groups, showing strong implicit bias against these terms in all the models. This value depicts how sentences related to disability are more negative in value than other groups. The ScoreRange results show that BERT and GPT-2 are very sensitive to the language used by PWD as compared to other groups. The results indicate that models with large ScoreRange tend to provide a more wider range of sentiment score results than other models, making the model more sensitive in prediction.

Table 7 and table 6 shows the ScoreSense performance of each subgroup for all the embedding models and PLMs respectively. The data shows that PLMs show large negative scoring for certain groups compared to embedding models. The erratic scoring amongst models shows the unpredictable behaviour of models due to their black-box nature.

Group Name	Definitions
Sensory	Conditions that include blindness, deafness, or a severe vision or hearing impairment.
Physical	Conditions that substantially limit one or more basic physical activities such as walking, climbing stairs, reaching, lifting, or carrying.
Mental	Because of a physical, mental, or emotional condition lasting 6 months or more, the person has difficulty learning, remembering or concentrating.
Self-care	Because of a physical, mental, or emotional condition lasting 6 months or more, the person has difficulty dressing, bathing, or getting around inside the home.
Go-outside-home	Because of a physical, mental, or emotional condition lasting 6 months or more, the person has difficulty going outside the home alone to shop or visit a doctor's office.
Employment	Because of a physical, mental, or emotional condition lasting 6 months or more, the person has difficulty working at a job or business.

Table 4: The definition of each subgroup of the Disability group is mentioned in this table. Six subcategories are decided based on the parameters defined in the US Census to collect disability data. The seventh subgroup 'General' is defined as the common words that are used to refer to people with disability.

Name	Mean (STD)	Mean (D)	Mean (ND)	ScoreRange
GN_3B_300D	0.16	0.02	0.05	1.28
CC_42B_300D	0.15	0.08	0.06	1.26
CC_840B_300D	0.06	0.03	0.03	1.21
WP_6B_100D	0.06	-0.01	0.02	1.30
WP_6B_200D	0.16	-0.02	0.02	1.30
WP_6B_300D	0.12	-0.01	0.01	1.21
WP_6B_50D	0.03	0.01	0.02	1.27
TW_27B_100D	0.11	-0.01	0.02	1.24
TW_27B_200D	0.15	0.02	0.03	1.24
TW_27B_25D	0.01	-0.01	0.02	1.33
TW_27B_50D	0.06	0.01	0.03	1.30
BERT	-0.02	-0.08	-0.01	1.60
GPT2	0.03	-0.02	0.04	1.87

Table 5: The Table shows the statistical mean value calculated for Standard (STD), Disability (D) and Non-Disability (ND) groups respectively. The ScoreRange value is also calculated to measure the range of all the sentiment scores generated by each mode.

Name	Employment	Go-Outside-Home	Mental	Physical	Self-Care	Sensory
BERT	-0.02	-0.21	-0.01	-0.03	-0.14	0.00
GPT2	-0.04	-0.22	0.01	-0.02	-0.14	0.00

Table 6: The table shows the breakdown sentiment score for each Disability Subgroup amongst large scale language models alone. We see that GPT2 and BERT demonstrate significantly negative results for certain subgroups.

Name	Employment	General	Go-Outside-Home	Mental	Physical	Self-Care	Sensory
GN_3B_300D	-0.14	-0.08	-0.19	-0.19	-0.10	-0.13	-0.08
CC_42B_300D	-0.08	-0.06	-0.11	-0.07	-0.09	-0.06	-0.06
CC_840B_300D	-0.05	0.01	-0.13	-0.05	-0.02	0.01	0.01
WP_6B_100D	-0.13	-0.04	-0.21	-0.07	-0.03	-0.01	-0.04
WP_6B_200D	-0.26	-0.14	-0.20	-0.15	-0.16	-0.14	-0.14
WP_6B_300D	-0.25	-0.09	-0.12	-0.10	-0.12	-0.10	-0.09
WP_6B_50D	-0.16	-0.03	-0.09	-0.03	0.08	0.00	0.03
TW_27B_100D	-0.14	-0.17	-0.13	-0.08	-0.08	-0.02	-0.17
TW_27B_200D	-0.17	-0.15	-0.16	-0.10	-0.12	-0.06	-0.15
TW_27B_25D	-0.05	-0.01	-0.03	-0.10	-0.01	0.03	-0.01
TW_27B_50D	-0.05	-0.09	-0.04	-0.08	-0.04	0.03	-0.09

Table 7: The table shows the breakdown sentiment score for each Disability Subgroup amongst embedding groups alone. We see that each embeddings layers demonstrate significantly negative results for certain subgroups.

Social Norms-Grounded Machine Ethics in Complex Narrative Situation

Tao Shen, Xiubo Geng, Daxin Jiang*
Microsoft Corporation
{shentao,xigeng,djiang}@microsoft.com

Abstract

Ethical judgment aims to determine if a person in a narrative situation acts under people's social norms under a culture, so it is crucial to understand actions in narratives and achieve machine ethics. Recent works depend on data-driven methods to directly judge the ethics of complex real-world narratives but face two major challenges. First, they cannot well handle dilemma situations due to a lack of basic knowledge about social norms. Second, they focus merely on sparse situation-level judgment regardless of the social norms involved during the judgment, leading to a black box. In this work, inspired by previous knowledge-grounded and -augmented paradigms, we propose to complement a complex situation with grounded social norms. Besides a norm-grounding knowledge model, we present a novel norm-supported ethical judgment model in line with neural module networks to alleviate dilemma situations and improve norm-level explainability. Empirically, our model improves state-of-the-art performance on two narrative judgment benchmarks.

1 Introduction

In natural language processing (NLP) literature, ethical judgment aims to determine if a person (e.g., narrator or someone else) in a narrative situation is morally wrong or correct (Lourie et al., 2021). For example, in a narrative situation "*I helped him but got taunted*", the narrator is morally good while the other is bad. It attracts more interest from academia and industry as it plays an indispensable role in human-centric applications and benefits a wide range of downstream tasks, e.g., dialogue systems and storytelling.

Recently, Forbes et al. (2020) propose a generative model, NORM TRANSFORMER, which can extract actions[1] from a narrative situation and then judge the ethics towards the actions. However, it can only generate action-level ethical judgment for simple narrative situations (i.e., sentences with limited events), so it usually fails to perform in complex narrative situations of many real-world applications. Here, "complex" is usually reflected in over-long narrative contexts (multiple paragraphs) and/or dilemma situations. Take Situation 1 in Figure 1 as a dilemma example: although we keep in mind that "*it's bad to punch others*", we cannot conclude the narrator is morally bad due to "*it's bad to steal*".

Situation 1: *I almost punch a friend who stole from me.*
Norm 1: *It's very bad to injure a person.*
Norm 2: *It's very bad to steal from others.*
Norm 3: *It's bad to betray a friend.*
Norm 4: *It's OK to want to take revenge.*
Situation 2: *When I call in a take out order and go pick it up myself, I don't tip.*
Norm 1: *It's good to always tip your server.*
Norm 2: *It's good to tip the driver when they pick you up.*
Norm 3: *It's bad to not want to leave a tip.*
Norm 4: *It's okay to expect a tip on takeout food.*

Figure 1: Two situations and the involved social norms. The first one is adapted from Forbes et al. (2020) while the second one is grounded by our model, where the norm in red is context-irrelevant to the situation.

On the contrary, Lourie et al. (2021) propose a data-driven method to directly judge complex narrative situations (e.g., real-life anecdotes from the Internet). Empowered by pre-trained language models (e.g., BERT (Devlin et al., 2019) and RoBERTa (Liu et al., 2019)), the proposed method achieves satisfactory performance to boost its real-world applications but encounters two major challenges in the following.

First, complex situations often pose intricate storylines and character relationships, leading to more difficult moral *dilemmas* during ethical judgments. Again, it is difficult for machines to directly judge whether the narrator is morally wrong in Situation 1 of Figure 1 because it requires machines to im-

*Corresponding author.

[1]In this paper, "action" denotes a verb-centric "event" (Zhang et al., 2020) without subject, e.g., "*helped him*" and "*got taunted*".

Proceedings of the 29th International Conference on Computational Linguistics, pages 1333–1343
October 12–17, 2022.

ply multiple human-level social norms from the situation and understand their relations before making the final judgment. The social norms[2] can be regarded as unspoken commonsense rules about acceptable social behavior, which are crucial for an AI system to understand people's actions in narratives (Forbes et al., 2020). Second, human-curated labels of ethical judgment in complex situations are *sparse*: due to limited crowd-sourcing, the ethical judgment is labeled for a whole situation, i.e., at a very coarse level. As a result, a model learned on such sparse-labeled complex situations can only work as a black box to derive situation-level judgments, regardless of the involved social norms behind the judgments.

To overcome both the challenges, inspired by recent advances in knowledge-grounded/-augmented methods of open-domain (Wang et al., 2019) and commonsense (Lv et al., 2020) question answering (QA), we argue to complement complex narrative situations with action-level diverse social norms. Continue to take Situation 1 in Figure 1 as an example: although it is difficult to make a judgment based on the first two social norms, coupling with the other two diverse norms, "*It's bad to betray a friend*" and "*It's okay to want to take revenge*", can intuitively endorse a morally-okay judgment towards the narrator. Differing from previous knowledge-augmented methods that measure the relatedness of a query with grounded facts and then find an answer in the facts, our motivation is that the complementary social norms serve as supportive evidence to reduce moral dilemmas and promote norm-level explainability.

To this end, we propose a brand-new flexible ethical judgment framework with complementary social norms for complex narrative situations. First of all, to ground each event in a complex situation with diverse social norms and ensure grounding coverage given limited resources, we build a new norm-grounding knowledge model to generate social norms given a simple situation (e.g., a sentence in a complex situation) based on a pre-trained encoder-decoder backbone.

After grounding, we propose **N**orm-supported **E**thical Ju**d**gment (NEd) model in line with neural module networks (Liu et al., 2020) to complement a complex situation with grounded social norms. Specifically, built upon a pre-trained contextualizing encoder (e.g., RoBERTa), the model is composed of three neural modules: 1) *supportive alignment module* to softly and coarsely assign a sentence in the situation with its semantic-relevant norms, 2) *hierarchical integration module*, taking the alignment module's outputs as coarse evidence while operating at the token level, to enrich representations of events in the situation with those of social norms, and 3) *selective judgment module* to focus on key parts of the integration results and then make the final ethical judgment.

Our NEd model has certain merits: First, attributed to the alignment module, our model is robust to the errors (e.g., context-irrelevant/wrong norms, e.g., Norm 2 of Situation 2 in Figure 1) propagated from our norm-grounding knowledge model. Second, with a hierarchical (both norm-level and token-level) structure, our model can precisely enrich events in a situation with fine-grained ethical information, leading to superior judgment performance. Third, facilitated by intermediate outputs of the modules, our model is equipped with explainability in terms of human-understandable social norms. Lastly, our framework is general, flexible enough to various settings (see §4) of the ethical judgment and achieves new state-of-the-art performance on two benchmark datasets.

2 Related Work

In NLP, instead of theory-driven top-down approaches under prescriptive ethics (Bringsjord et al., 2006; Rossi and Mattei, 2019; Gert and Gert, 2020), recent works focus on data-driven bottom-up approaches with descriptive approaches (Balakrishnan et al., 2019; Wu and Lin, 2018) to achieve machine ethics (Rzepka and Araki, 2005; Anderson and Anderson, 2011). Ethical judgment, as an important task of machine ethics in NLP, is getting increased attention (Wolf et al., 2017; Schlesinger et al., 2018). Recent solutions (Lourie et al., 2021) depend on a data-driven paradigm but neglect the importance of the involved social norms during the judgment. But, how to explicitly integrate the social norms into ethical judgment in complex narratives is an open question.

Recently, several paradigms have been proposed to integrate additional knowledge, especially in open-domain and commonsense QA. Specifically, open-domain QA (Wang et al., 2019; Yang et al., 2019) retrieves related documents from large-scale

[2]"Norm" denotes to assign an ethical judgment to an "action", e.g., "*It's good to help others*" or ("*helping others*", good). It is a.k.a rules-of-thumb (RoT) in some literature (Forbes et al., 2020).

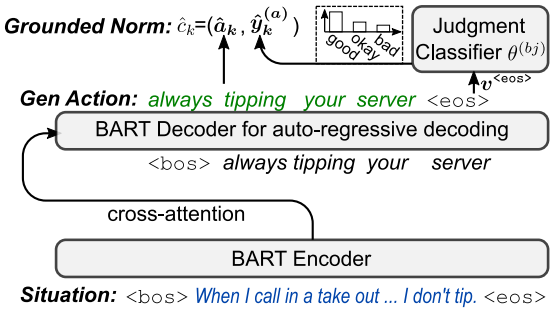

Figure 2: Norm-grounding knowledge model.

corpus according to a query and then predict an answer based on the retrieved documents; commonsense QA (Lin et al., 2019; Lv et al., 2020) resorts to grounding on structured knowledge and then symbolically/latently derives the final answer based on the grounded facts. Though effective in their own fields, these integration paradigms are inapplicable here to fulfill our goal, i.e., enriching a complex situation with action-level, fine-grained social norms for less moral dilemmas and more human-level explainability.

3 Proposed Approach

This section begins with a general definition of ethical judgment task. Then, we present our norm-grounding knowledge model to ground events in a situation with diverse social norms (in §3.1 and illustrated in Figure 2). Next, given a grounded complex situation, we present our norm-supported ethical judgment (NEd) model as a flexible framework (in §3.2 and illustrated in Figure 3). Lastly, we detail training objectives to of our model in §3.3. This section begins with a general definition of ethical judgment task, followed by our norm-grounding knowledge model in §3.1 with Figure 2 and norm-supported ethical judgment (NEd) model in §3.2 with Figure 3.

Ethical Judgment. A complex narrative situation $\boldsymbol{S}$ consists of a sequence of sentences $\boldsymbol{S} = [\boldsymbol{s}_1, \ldots, \boldsymbol{s}_m]$ where m denotes the number of sentences. Given a situation $\boldsymbol{S}$, *ethical judgment* aims to discriminate whether a person (e.g., the narrator or the other(s)) in $\boldsymbol{S}$ is morally wrong. Hence, ethical judgment is usually formulated as a classification problem, and the categories $\mathcal{Y}$ can be binary (i.e., {*good, bad*}), fine-grained (e.g., {*very good, good, okay, bad, very bad*}), etc.

3.1 Norm-Grounding Knowledge Model

Basically, we need to ground each event in a situation with diverse social norms. Traditional grounding methods (e.g., entity linking (Chen et al., 2020), event grounding (Du et al., 2021)) depend on lexical/semantic overlapping between a mention and entries in knowledge bases (e.g., corpus, graph). But, they are inapplicable to grounding an event with norms as events are expressed in free-form texts and knowledge base of norms is scarce, leading to low coverage and precision. In contrast, neural knowledge model (Bosselut et al., 2019) offers a novel solution: it learns from limited seed knowledge but leverage pre-trained language models to generalize more.

Following this line, we focus on crowdsourced descriptions of norms and present a *norm-grounding knowledge model* as in Figure 2, to generate social norms given a simple situation. It is similar to GPT2-based NORM TRANSFORMER (Forbes et al., 2020) but differs in both target and base model.

First, we give a formal task definition to build a neural knowledge model for norm grounding. Here, we leverage training data from SOCIAL CHEMISTRY 101, which offers a sentence-level simple situation $\boldsymbol{s}$ in various scenarios and its corresponding diverse social norms $\mathcal{C} = \{c_1, \ldots\}$, where each social norm c_k is composed of 1) an action $\boldsymbol{a}_k$ to describe one event in $\boldsymbol{s}$ and 2) its ethical judgment label $y_k^{(a)} \in \mathcal{Y}$, i.e., $c_k = (\boldsymbol{a}_k, y_k^{(a)})$.[3] Hence, the goal of norm grounding is to generate a set of diverse social norms $\mathcal{C}$ given a simple situation $\boldsymbol{s}$ (e.g., a sentence from a complex situation), i.e., $P(\mathcal{C}|s;\theta)$, which needs to cover all events in $\boldsymbol{s}$.

Then, we employ a pre-trained encoder-to-decoder model, BART (Lewis et al., 2020), as backbone to translate sentence-level situation $\boldsymbol{s}$ to an action $\boldsymbol{a}_k$ with its judgment $y_k^{(a)}$. That is, we define a conditional generation from $\boldsymbol{s}$ to $\boldsymbol{a}$, i.e.,

$$\hat{a}_t = \text{BART-EncDec}(\boldsymbol{s}, [\texttt{<bos>}, \boldsymbol{a}_{<t}]; \theta^{(bart)}),$$

where $\boldsymbol{s}$ is encoder input, $\boldsymbol{a} = [a_1, \ldots]$ is the tokenized action $\boldsymbol{a}$, and $\hat{a}_t$ is the predicted token in t-th time step.

Next, to get an ethical judgment of the predicted action, we leverage the last states of the decoder

[3]Note 1) even a simple situation could contain multiple events, and 2) We take apart each *social norm* c_k with its *action* $\boldsymbol{a}_k$ and the corresponding *ethical judgment* $y_k^{(a)}$. This can highlight the semantics of the action of the norm and keep the judgment categorical.

(i.e., $\boldsymbol{h}^{(\texttt{<eos>})}$ – the embedding of end-of-sequence token <eos> in decoding):

$$\boldsymbol{y}^{(a)} = \text{softmax}(\text{MLP}(\boldsymbol{h}^{(\texttt{<eos>})}; \theta^{(bj)})), \quad (1)$$

where $\text{MLP}(\cdot; \theta^{(bj)})$ is $\theta^{(bj)}$-parameterized multi-layer perceptron (MLP) to produce categorical distribution $\hat{\boldsymbol{y}}^{(a)} \in \mathbb{R}^{|\mathcal{Y}|}$ of ethical judgment towards the generated action $\hat{\boldsymbol{a}}$.

Training & Inference. We minimize an addition of negative log-likelihood of action generation and cross-entropy of ethical judgment. After trained, we use a sampling technique (Holtzman et al., 2020) to generate diverse norms: besides beam search (w/ size of 2), we use top-p=0.9 sampling during decoding and generate K social norms $\hat{\mathcal{C}} = \{(\hat{\boldsymbol{a}}_k, \hat{\boldsymbol{y}}_k^{(a)})_{k=1}^K\}$ in parallel for each situation $\boldsymbol{s}$ to ensure coverage/diversity.

3.2 Norm-Supported Ethical Judgment Model

After invoking norm-grounding knowledge model for each sentence $\boldsymbol{s}^j$ in a complex situation $\boldsymbol{S}$, we get social norms by

$$\hat{\mathcal{C}}^j = \{(\hat{\boldsymbol{a}}_k^j, \hat{\boldsymbol{y}}_k^{(a),j})_{k=1}^K\}, \forall j \in [1, m]. \quad (2)$$

Here, an open question remains about how to integrate these complementary norms into situation-level ethical judgment.

As an answer, we present a novel integration paradigm for ethical judgment, dubbed **N**orm-supported **E**thical ju**d**gment (NEd) model as in Figure 3. We introduce a concept of neural module network (Liu et al., 2020) because it can empower human-level explainability by visualizing intermediate outputs, consistent with our goal. It is noteworthy that, instead of considering variant combinations of neural modules as in (Andreas et al., 2016), we fix the neural architecture and focus more on the design of the modules as in (Liu et al., 2020).

First, we utilize a pre-trained Transformer encoder (e.g., RoBERTa) to embed a whole situation $\boldsymbol{S} = [\boldsymbol{s}^1, \ldots, \boldsymbol{s}^m]$ and the action $\hat{\boldsymbol{a}}_k^j$ of each generated social norm, i.e.,

$$\boldsymbol{U} = \text{Trans-Enc}([\boldsymbol{s}^1, \ldots, \boldsymbol{s}^m]; \theta^{(te)}), \quad (3)$$

$$\boldsymbol{V}_k^j = \text{Trans-Enc}(\hat{\boldsymbol{a}}_k^j; \theta^{(te)}), \quad (4)$$

where, $\forall j \in [1, m], k \in [1, K]$, and two encoders share parameters except for positional embeddings. $\boldsymbol{U}$ and $\boldsymbol{V}_k^j$ denote token-level representations.

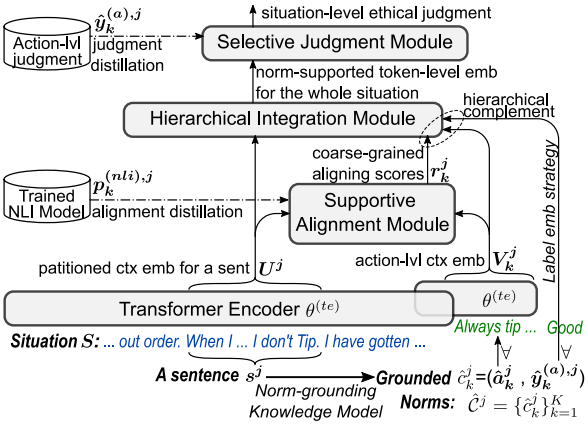

Figure 3: **N**orm-supported **E**thical Ju**d**gment (NEd) model.

After situation-level long-term contextualizing, we partition $\boldsymbol{U}$ to sentence-level blocks to facilitate later integration:

$$\boldsymbol{U} = [\boldsymbol{U}^1, \ldots, \boldsymbol{U}^m], \forall \boldsymbol{U}^j \in \mathbb{R}^{d \times n^{(s)}}, \quad (5)$$

where $n^{(s)}$ is the number of tokens in a sentence. Built upon the above representations for 1) sentences in the situation and 2) actions of social norms, we propose three neural modules in the following to fulfill ethical judgment.

Supportive Alignment Module. It is crucial to measure if the action of a norm is not only consistent with at least one event in the sentence but also coherent to the context of the sentence. We first apply a mean-pooling to a sentence $\boldsymbol{U}^j$ and an action $\boldsymbol{V}_k^j$ to get one vector representation of each:

$$\boldsymbol{u}^j = \text{Pool}(\boldsymbol{U}^j) \text{ and } \boldsymbol{v}_k^j = \text{Pool}(\boldsymbol{V}_k^j), \quad (6)$$

where $\forall k \in [1, K]$. Then, following (Reimers and Gurevych, 2019), we represent their relationship by an interactive concatenation, i.e.,

$$\boldsymbol{o}_k^j = [\boldsymbol{u}^j; \boldsymbol{u}^j - \boldsymbol{v}_k^j; \boldsymbol{u}^j \odot \boldsymbol{v}_k^j; \boldsymbol{v}_k^j], \quad (7)$$

where $\forall k \in [1, K]$, $[\cdot;\cdot]$ denotes vector concatenation and $\odot$ denotes Hadamard product. Lastly, the relationship representation $\boldsymbol{o}_k^j$ is fed into an MLP with binary output, i.e,,

$$\boldsymbol{r}_k^j = \text{softmax}(\text{MLP}(\boldsymbol{o}_k^j; \theta^{(al)})) \in \mathbb{R}^2, \quad (8)$$

where $(\boldsymbol{r}_k^j)_{[r=2]}$ denotes the relatedness intensity between sentence $\boldsymbol{s}^j \in \boldsymbol{S}$ and the action $\hat{\boldsymbol{a}}_k^j \in \mathcal{C}^j$. As a side benefit, such a module can also circumvent the errors propagated from norm-grounding knowledge model defined in §3.1.

Hierarchical Integration Module. After coarse alignment, we need to enrich each sentence in a complex situation with corresponding social norms to obtain diverse, supportive information. But we cannot integrate the norms in a straightforward manner (e.g., concatenation or addition) as a sentence contains multiple events, and we have no idea about which part is an event, not to mention how to align an event with actions. Hence, besides coarse norm-supportive alignment, we need to consider more fine-grained integration – operating at the token level and integrating in a sophisticated manner. Formally, we first equip a social norm's action $\boldsymbol{V}_k^j$ with its action-level judgment $\hat{\boldsymbol{y}}_k^{(a),j}$ to compose a complete representation of the norm (as mentioned in §3.1, we take each norm apart into action and judgment). That is

$$\tilde{\boldsymbol{V}}_k^j = \boldsymbol{V}_k^j + \boldsymbol{W}^{(jdg)}\hat{\boldsymbol{y}}_k^{(a),j}, \tag{9}$$

where $\boldsymbol{W}^{(jdg)} \in \mathbb{R}^{d\times|\mathcal{Y}|}$ denotes a weight matrix to identify judgment by following label embedding strategy (Wang et al., 2018), and the '+' here broadcasts along with sequence axis. Then, to achieve our hierarchical integration, we adapt one layer of the Transformer decoder by 1) we remove the self-attention but keep the cross-attention plus an MLP with residual connection and layer norm, 2) the cross-attention uses actions $\{\boldsymbol{V}_k^j\}_{k=1}^K$ as keys and their social norms $\{\tilde{\boldsymbol{V}}_k^j\}_{k=1}^K$ as values, and 3) we take the outputs $\{(\boldsymbol{r}_k^j)_{[r=2]}\}_{k=1}^K$ from the alignment module in Eq.(8) as norm-level gating values and apply them to cross-attention in a multiplicative manner. Thus, we define hierarchical integration operating on each s^j and its grounded norms $\hat{\mathcal{C}}^j$:

$$\begin{aligned}\bar{\boldsymbol{U}}^j &= \sum\nolimits_k (\boldsymbol{r}_k^j)_{[r=2]} \cdot \tilde{\boldsymbol{V}}_k^j \mathrm{softmax}((\boldsymbol{U}^j)^T \boldsymbol{V}_k^j/\sqrt{d})^T,\\ \tilde{\boldsymbol{U}}^j &= \text{Layer-Norm}(\boldsymbol{U}^j + \bar{\boldsymbol{U}}^j; \theta^{(lm)}),\end{aligned} \tag{10}$$

where $\forall j \in [1, m]$. Here for clear writing, we omit multi-head projections and an MLP after the attention, and please refer to (Vaswani et al., 2017) for their details. Next, given all enriched sentence representations $\{\tilde{\boldsymbol{U}}^j\}_{j=1}^m$ for sentences $[\boldsymbol{s}^1, \ldots, \boldsymbol{s}^m]$, we re-unite them into token-level representations of the whole situation:

$$\tilde{\boldsymbol{U}} = [\tilde{\boldsymbol{U}}^1, \ldots, \tilde{\boldsymbol{U}}^j], \tag{11}$$

where $[\cdot,\cdot]$ denotes concatenation along with sequence axis. Lastly, we apply one layer of Transformer encoder to $\tilde{\boldsymbol{U}}$ for long-term contextualized representations, i.e.,

$$\boldsymbol{E} = \text{Transformer-Layer}(\tilde{\boldsymbol{U}}; \theta^{(tl)}), \tag{12}$$

where $\boldsymbol{E}$ stands for the representations for all tokens in the situation $\boldsymbol{S}$, which have been integrated with precise, diverse, and supportive social norms in a hierarchical way.

Selective Judgment Module. Given $\boldsymbol{E}$, we first apply an attentive pooling (Liu et al., 2016; Lin et al., 2017), which aims at focusing on key parts of the integrated results, i.e.,

$$\boldsymbol{e}{=}\text{Attn-Pool}(\boldsymbol{E}){:=}\boldsymbol{E}\mathrm{softmax}(\mathrm{MLP}(\boldsymbol{E}; \theta^{(ap)})), \tag{13}$$

where $\boldsymbol{e} \in \mathbb{R}^d$, and $\mathrm{MLP}(\cdot; \theta^{(ap)})$ is one-way out to represent the importance of each token. Lastly, we feed $\boldsymbol{e}$ into an MLP-based classifier, i.e.,

$$\begin{aligned}\hat{\boldsymbol{y}} &= \mathrm{softmax}(\mathrm{MLP}(\boldsymbol{e}; \theta^{(cl)})),\\ \hat{y} &= \arg\max \hat{\boldsymbol{y}},\end{aligned} \tag{14}$$

where $\hat{\boldsymbol{y}} \in \mathbb{R}^{|\mathcal{Y}|}$ is a categorical distribution over $\mathcal{Y}$, and $\hat{y}$ denotes the predicted judgment for the situation $\boldsymbol{S}$.

3.3 Training Objective

To train our proposed NEd model in an end-to-end fashion, we can define a cross-entropy loss for $\hat{\boldsymbol{y}}$ in Eq.(14), i.e.,

$$\mathcal{L}^{(main)} = -\sum\nolimits_{\boldsymbol{S}\in\mathcal{D}} \log \hat{\boldsymbol{y}}_{[y=y^*]}, \tag{15}$$

where $\hat{\boldsymbol{y}}_{[y=y^*]}$ denotes the probability of the gold label of $\boldsymbol{S}$ and $\mathcal{D}$ denotes training set. But there are many ethical judgment settings other than the simple classification. To exhibit our framework's flexibility to various settings, we will detail adapting procedure into two settings later in experiments. Besides the main loss $\mathcal{L}^{(main)}$, we design two distillation objectives to ensure they perform as expected. (i) $\mathcal{L}^{(ad)}$: *Alignment Distillation* aims at distilling semantic knowledge from a well-trained natural language inference (NLI) model to $\boldsymbol{r}_k^j$ (Eq.(8)) in the supportive alignment as a situation sentence is expected to entail the aligned social norms. (ii) $\mathcal{L}^{(jd)}$: *Judgment Distillation* aims at distilling action-level judgment knowledge $\hat{\boldsymbol{y}}_k^{(a),j}$ from our norm-grounding model into the *selective judgment module* by pooling each social norm's action plus the judge classifier, i.e., $\mathrm{MLP}(\mathrm{Pool}(\boldsymbol{V}_k^j); \theta^{(cl)})$.

Dataset	# Train	# Dev	# Test	# Tokens/Situation
ANECDOTES	27,766	2,500	2,500	410
DILEMMAS	23,596	2,340	2,360	10

Table 1: Statistics of two ethical judgment benchmarks.

Method (Macro-F1)	ANECDOTES		DILEMMAS	
	Dev	Test	Dev	Test
Prior	16.4	16.1	34.1	34.2
Sample	19.7	19.1	49.9	50.5
Style	16.5	16.2	55.0	52.4
BinaryNB	16.8	16.8	/	/
MultiNB	20.2	19.2	/	/
CompNB	23.4	22.9	/	/
Forest	16.4	16.1	/	/
Logistic	19.2	19.2	65.0	64.3
$\text{BERT}_{\text{large}}$	21.8	21.6	72.8	72.0
$\text{BERT}_{\text{large}}$ + Dirichlet	23.2	25.9	72.9	73.7
$\text{RoBERTa}_{\text{large}}$	27.8	30.5	75.7	74.6
$\text{RoBERTa}_{\text{large}}$ + Dirichlet	29.6	30.2	76.0	78.3
NEd-$\text{RoBERTa}_{\text{large}}$ (**ours**)	**41.20**	**37.32**	**76.91**	**78.59**
Human Performance	46.8	49.0	80.7	80.4

Table 2: Comparisons to state-of-the-art competitors on two benchmark datasets.

Alignment Distillation. The *supportive alignment module* is designed to measure if the action of a social norm can provide supportive knowledge according to the context. This is similar to natural language inference (NLI) (Bowman et al., 2015) measuring if a premise is of *entailment*, *neutral*, or *contradiction* to a hypothesis, so it is intuitive to distill an NLI model to $\boldsymbol{r}_k^j$ in Eq.(8). Rather than direct distillation, it is noteworthy even if a norm's action contradicts the situation, the norm still can provide supports (e.g., in Figure 1, Situation 2 vs. its Norm 3). In formal, we first employ an NLI model trained on multi-genre natural language inference (Nangia et al., 2017) and pass a concatenation of $(\boldsymbol{s}^j, \hat{\boldsymbol{a}}_k^j)$ into the model to derive a three-categorical distribution $\bar{\boldsymbol{p}}_k^{(nli),j}$. Then, we merge *contradiction* and *entailment* to obtain a new distribution $\boldsymbol{p}_k^{(nli),j}$ over {*neutral*, *non-neutral*}. Thereby, we employ a Kullback–Leibler (KL) divergence between $\boldsymbol{r}_k^j$ and $\boldsymbol{p}_k^{(nli),j}$ as the training loss, i.e.,

$$\mathcal{L}^{(ad)} = \sum\nolimits_{\mathcal{D}} \sum\nolimits_{j=1}^{m} \sum\nolimits_{k=1}^{K} \text{KL-Div}(\boldsymbol{r}_k^j, \boldsymbol{p}_k^{(nli),j}).$$

Judgment Distillation. Since we have the contextual representations of each social norm's action as well as its action-level judgment, it is promising to distill such knowledge into the *selective judgment module*. To complete this, we apply a mean pooling to each $\boldsymbol{V}_k^j$ derived in Eq.(4) to get one vector representation of each action: $\boldsymbol{v}_k^j = \text{Pool}(\boldsymbol{V}_k^j),\ \forall j \in [1, m], \forall k \in [1, K]$. The reason for not using the attentive pooling in Eq.(13) is that an action is only composed of a dozen of token so it is unnecessary for such a short sequence. Lastly, we use a KL divergence as the loss:

$$\mathcal{L}^{(jd)} = \sum\nolimits_{\mathcal{D},j,k} \text{KL-Div}(\text{softmax}(\text{MLP}(\boldsymbol{v}_k^j; \theta^{(cl)})), \hat{\boldsymbol{y}}_k^{(a),j}),$$

where $\theta^{(cl)}$ denotes the parameters defined in Eq.(14).

Overall Training Objective. Consequently, we can we can define the overall training objective as

$$\mathcal{L} = \mathcal{L}^{(main)} + g(\alpha) \cdot (\mathcal{L}^{(ad)} + \mathcal{L}^{(jd)}), \quad (16)$$

where $g(\alpha)$ is exponential anneal of the weight.

Method	**Bal-Acc**		**Macro-F1**	
	Dev	**Test**	**Dev**	**Test**
$\text{RoBERTa}_{\text{base}}$	26.62	28.14	27.84	29.59
$\text{RoBERTa}_{\text{base}}$ + Dirichlet	29.05	29.91	30.41	30.97
$\text{RoBERTa}_{\text{base}}$ + Soft	37.28	33.27	37.81	33.91
NEd-$\text{RoBERTa}_{\text{base}}$ (**ours**)	40.55	34.63	39.90	35.07
NEd-$\text{RoBERTa}_{\text{large}}$ (**ours**)	42.40	37.99	41.20	37.32

Table 3: Detailed comparisons on ANECDOTES.

4 Experiments

Datasets and Setting Adaptation. To exhibit our framework is flexible to various settings, we detail adaptations into two settings, corresponding to two datasets, i.e., ANECDOTES and DILEMMAS (Lourie et al., 2021). The statistics of the two benchmark datasets are listed in Table 1. For ANECDOTES, given a very complex narrative situation with multiple paragraphs, its goal is to discriminate {*nobody wrong*, *narrator wrong*, *other wrong*, *everybody wrong*, or *more info needed*}. We employ a binary judgment category and add another attentive pooling to the *selective judgment module*. We expect the two attention pooling mechanisms to focus on different parts of integrated results, which perform ethical judgments for "*narrator*" and "*other*" respectively. Besides, we add an MLP to calculate the probability of the result falling into *more info needed*. For DILEMMAS, given 2 situations (each w/ several events), its goal is to contrast their ethics and point out which one is more wrong. We use a binary judgment category and take the predicted probabilities of *bad* in $\hat{\boldsymbol{y}}$ (i.e., $\hat{\boldsymbol{y}}_{[y=\text{bad}]}$) as wrong scores.

Setups. To train our knowledge model, we use SOCIAL CHEMISTRY 101 (SC101) and employ $\text{BART}_{\text{large}}$. We optimize the model using mini-batch SGD with Adam optimizer, where learning rate is 10^{-5} with 5% warmup proportion, batch size is 16, the number of training epochs is 3. On the other hand, we initialize the backbone of our NEd model with either $\text{RoBERTa}_{\text{large}}$ or $_{\text{base}}$. Instead of hyperparameter tuning with Gaussian process optimization by Lourie et al. (2021), we set hyperparameters according to our experiences or early trials. We set K to 5/3 in two datasets. In Eq.(16), $\alpha \in [0, 1]$ is the ratio of training progress, and $g(x) = \exp(-\gamma \cdot x)$ is an exponential anneal where γ is a hype-parameter. We set γ to 10 to push the learning more incline to main loss. The base model is set with learning rate of 5×10^{-5} with 5% warmup, batch size of 32, number of epochs of 7. The large model is set with learning rate of 10^{-5} with 5% warmup, batch size of 16, number of epochs of 3. We run each model with 3 random seeds and evaluate on dev set every 500 steps during training; we report the best dev results as well as the corresponding test results. All experiments are run at one single Nvidia RTX6000 GPU. The codes are published at https://github.com/taoshen58/NEd.

Metrics. We use Macro-F1 (%) as our main metric to compare models. Compared to (Lourie et al., 2021), our work does not target ambiguity in ethics-related tasks and focus on making ethical judgment consistent with the majority. Thus, we rather use balanced-accuracy (Bal-Acc, %) as another metric and will also consider cross-entropy metric (Lourie et al., 2021) to verify our model's versatility.

4.1 Main Evaluations

Comparison with Competitors. In Table 2, we compare our large model, i.e., $\text{NEd-RoBERTa}_{\text{large}}$, with previous state-of-the-art competitors on ANECDOTES and DILEMMAS. $\text{BERT}_{\text{large}}$ and $\text{RoBERTa}_{\text{large}}$ denote fine-tuning the model with a classifying head. When coupled with "+ Dirichlet", they denote using a Dirichlet-multinomial layer to generalize $\mathrm{softmax}$ and enable models to express uncertainty over class probabilities (Lourie et al., 2021). It is observed, our model outperforms previous methods and improves the state-of-the-art performance by 5.1% and 0.3% on ANECDOTES and DILEMMAS, respectively. A potential reason for a marginal improvement on DILEMMAS is our performance has been too close to human performance (78.6% vs. 80.4%), making lifts difficult.

Method	Bal-Acc		Macro-F1	
	Dev	Test	Dev	Test
$\text{RoBERTa}_{\text{base}}$	72.82	72.47	72.80	72.41
$\text{RoBERTa}_{\text{base}}$ + Dirichlet	73.96	73.84	73.92	73.80
$\text{RoBERTa}_{\text{base}}$ + Soft	73.83	73.52	73.79	73.47
$\text{RoBERTa}_{\text{base}}$ + Norms	74.53	73.50	74.51	73.43
$\text{RoBERTa}_{\text{base}}$ + Filtered Norms	74.66	74.32	74.64	74.24
NEd-$\text{RoBERTa}_{\text{base}}$ (**ours**)	75.61	75.00	75.59	74.95
NEd-$\text{RoBERTa}_{\text{large}}$ (**ours**)	76.94	78.61	76.91	78.59

Table 4: Detailed comparisons with baselines on DILEMMAS.

Comparison with Baselines. As shown in Table 3 and 4, we also compare our model with various baselines. Here, we add a method named "$\text{RoBERTa}_{\text{base}}$ + Soft", which has already appeared in (Lourie et al., 2021) with only reported state-of-the-art dev results on ANECDOTES, as a strong baseline, and we re-implement it with a label-weighted loss to further boost its performance. As shown in Table 3, although "$\text{RoBERTa}_{\text{base}}$ + Soft" is significantly superior to "$\text{RoBERTa}_{\text{base}}$ + Dirichlet", it is still beaten by our NEd model. Similarly, as listed in Table 4, our model can surpass baselines on DILEMMAS by a large margin.

Comparison with Norm-augmented Methods. For more fair comparisons, we build two other RoBERTa-based baselines that also utilize our grounded social norms. Following common practice, we concatenate a situation $\boldsymbol{S}$ with its social norms, i.e., $[\boldsymbol{S}, \{\hat{\boldsymbol{a}}_k^j, \hat{y}_k^{(a),j}\}_{j,k}]$. As shown in the 2nd block of Table 4, "+ Norm" denotes directly concatenating all the grounded social norms while "+ Filtered Norms" denotes filtering out the norms with less relatedness to the situation, i.e., large neutral probability from NLI model > 0.2. These two methods are only applicable to DILEMMAS as they will lead to over-long ($\gg 512$) inputs on ANECDOTES. The table shows that our model still notably outperforms these two norm-augmented methods. Besides, the method with filtered norms performs better than direct concatenation, verifying the positive effects of the NLI model and the importance of our alignment distillation.

Performance on Controversiality. Lourie et al. (2021) use cross-entropy (Xentropy) as a metric to measure if a learned model can handle controversiality in judgments. Despite not being our

Method	Macro-F1		Xentropy	
	Dev	Test	Dev	Test
RoBERTa$_{\text{large}}$	75.7	74.6	0.578	0.577
RoBERTa$_{\text{large}}$ + Dirichlet	76.0	78.3	0.570	0.566
NEd-RoBERTa$_{\text{large}}$ (**ours**)	**76.91**	**78.59**	**0.5657**	**0.5652**

Table 5: Model comparison with the metric of cross-entropy.

Method	Macro-F1
NEd-RoBERTa$_{\text{large}}$ (full model)	**75.59**
◇ Removing *Alignment Module*	74.54
◇ Replacing *Alignment Module* with NLI Prior	75.15
◇ Removing Distillation Objectives	74.61
◇ Removing *Integration Module*	73.57
◇ Removing All Modules	72.80

Table 6: Ablation study on DILEMMAS Dev.

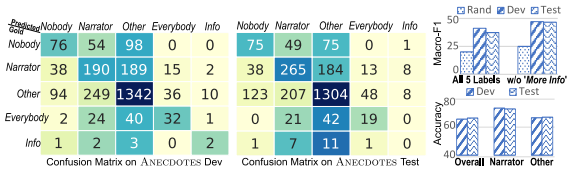

Figure 4: Insights into the performance on ANECDOTES.

Method	**All** (Ma-F1)	**Easy**	**Hard**
RoBERTa$_{\text{base}}$+Dirichlet	73.9	89.2	66.1
NEd-RoBERTa$_{\text{base}}$ (Δ)	75.6 (1.7)	90.05 (0.9)	68.15 (2.1)

Table 7: Evaluation on **easy** (cut-off examples) and **hard** (controversial dilemma examples). We split DILEMMAS Dev (**all**) into the two subsets according to if an example's consensus ratio > 0.8.

target, we report this metric to demonstrate our model's versatility. As in Table 5, all models include temperature calibration (Guo et al., 2017), and ours reaches competitive cross-entropy results compared to the specially designed Dirichlet layer.

Ablation Study. To check the contribution of each module, we conduct an ablation study in Table 6. Removing *Alignment Module*, equivalent to discarding coarse-grained integration, leads to a noticeable degeneration. And the degeneration will be alleviated when using prior coarse-grained weights (i.e., distillation targets from NLI) to replace the *Alignment Module*. Then, when removing the distillation objectives defined in Eq.(16), a notable performance decrease is observed, which verifies their importance. Next, we remove our *Integration Module*, degrading our model to RoBERTa plus our selective judgment module with distillation, resulting in a substantial decrease. Finally, we discard the only module, *selective judgment module*, from the last ablation, equal to RoBERTa baseline w/o all modules, leading to a further decrease.

4.2 Quantitative and Qualitative Analysis

Insights into ANECDOTES. Figure 2 shows a performance gap between dev and test on ANECDOTES. To dig this out, we illustrate their confusion matrices in Figure 4 (left & middle). It is shown that both recall and precision of label "*more info*" are 0 on the test, affecting macro metric. We further throw that label away, and the gap is largely narrowed, as in Figure 4 (upper-right). This exhibits that a distribution shift exists here and needs more efforts in the future. Besides the overall metric, our framework can derive the metrics w.r.t a specified person as in Figure 4 (bottom-right).

Handling Dilemmas Scenario. As shown in Table 7, we evaluate on two subsets of DILEMMAS Dev, which shows that our improvement on controversial dilemma examples is more significant than that on cut-off examples. These verify the superiority of our model in handling dilemmas scenarios.

Norm-Grounding Knowledge Model. We evaluate our knowledge model in Figure 5 (left), which shows low perplexity of action generation and high accuracy of action-level judgment on SC101 test. Although 5-categorical classification achieves 71%, as in Figure 5 (right) the misclassified examples mainly fall into its adjacent classes.

Evaluating Modules. In Figure 6, we give loss curves for Eq.(16): 1) the combined loss $\mathcal{L}$ is gradually close to the main loss $\mathcal{L}^{(main)}$ due to the exponential anneal, and 2) although learning rate of the two distillations are approaching to zero, their values do not increase significantly, verifying their objectives are consistent with our ethical judgment. Lastly, we test the performance of distilled modules in Figure 6 (right).

Case Study & Norm-level Explainability. As for Situation 1 in Figure 7, we show a case of ethical judgment in dilemma. It is observed our knowledge model precisely generates social norms consistent with the sentence and our hierarchical integration focuses on key norms to support the final ethical judgment. Meantime, the generated norms and alignment scores are human-understandable to intuitively explain the judgment from our model,

Metric	Dev	Test
Action PPL	6.80	6.94
Judge Accu	70.59	71.04

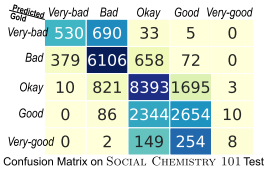

Figure 5: Evaluation of norm-grounding knowledge model.

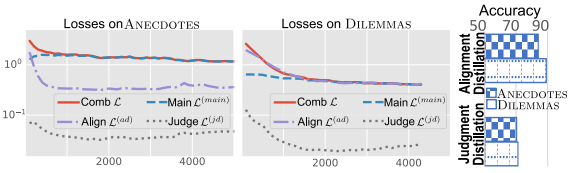

Figure 6: Loss descending curves with 0.99 moving average (left & middle) and module evaluation (right).

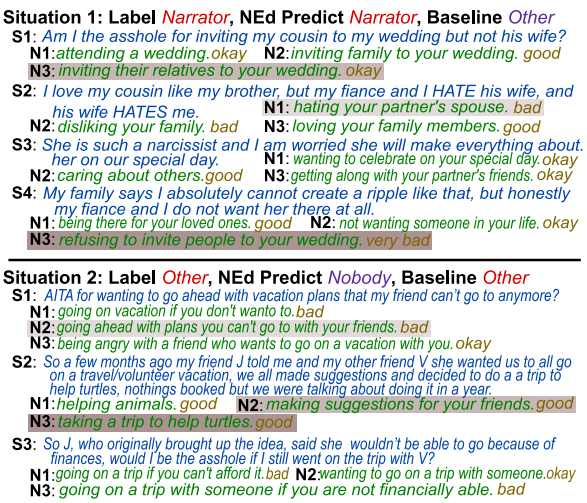

Figure 7: Case study of complex situations from Anecdotes and grounded social norms. Texts with shadow denotes top-3 attended norms (N) and more dark denotes more intensive attention.

which notably improves norm-level explainability compared to the existing ethical judgment framework. In contrast, Situation 2 in Figure 7 shows an error case of our model. The possible reasons leading to the error are knowledge model incompetent in the long sentence (S2).

Diversity of Integration Scores. To investigate if the hierarchical (i.e., norm-level and token-level) attention scores are diverse to focus on different perspectives of social norms, we calculate the Pearson correlation coefficient between them, and the resulting is -0.081. And, we calculate the coefficient between $\boldsymbol{p}_k^{(nli),j}{}_{[\text{neutral}]}$ from an NLI model and $\boldsymbol{r}_k^j$ from our alignment module, and the resulting is 0.565, verifying the learned module is not the same as the NLI model but learns latent alignments for our ethical judgment task.

Error Analysis & Limitations. We checked 20 error cases in Anecdotes test and found that it is difficult for human beings to judge the ethics of complex situations even given the social norms, suggesting more explainable works in the future. Lastly, we also recognize model limitations: 1) although improving norm-level explainability, our pipeline inevitably leads to error propagation and affects the performance; and 2) our model must calculate deep contextualization for all social norms, resulting in large computation overheads.

5 Conclusion

In this paper, we propose an ethical judgment framework for complex narrative situations to reduce dilemma situations and improve norm-level explainability. These are achieved by our designed norm-grounding knowledge model and norm-supported ethical judgment (NEd) model. We conduct extensive experiments on two benchmark datasets to verify its superiority from both quantitative and qualitative perspectives.

Ethics Statement. This work does not involve any sensitive data, but only public crowd-sourced corpora released in (Forbes et al., 2020; Lourie et al., 2021). Even the first situation and its social norms in Figure 1 (which may cause legal controversiality) are adapted from (Forbes et al., 2020). The resulting ethical judgment model can serve as a plug-in module to AI systems w/ language generation (e.g., dialogue system and chat-bot). First, it can filter unethical generated sentences. Second, it can perform ethical checks for massive user-posted and crowd-sourced data, thus reducing human-in-the-loop costs. Third, our model takes a step further to break down ethical judgments for norm-level transparency.

References

Michael Anderson and Susan Leigh Anderson. 2011. *Machine ethics*. Cambridge University Press.

Jacob Andreas, Marcus Rohrbach, Trevor Darrell, and Dan Klein. 2016. Neural module networks. In *2016 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2016, Las Vegas, NV, USA, June 27-30, 2016*, pages 39–48. IEEE Computer Society.

Avinash Balakrishnan, Djallel Bouneffouf, Nicholas Mattei, and Francesca Rossi. 2019. Incorporating

behavioral constraints in online AI systems. In *The Thirty-Third AAAI Conference on Artificial Intelligence, AAAI 2019, The Thirty-First Innovative Applications of Artificial Intelligence Conference, IAAI 2019, The Ninth AAAI Symposium on Educational Advances in Artificial Intelligence, EAAI 2019, Honolulu, Hawaii, USA, January 27 - February 1, 2019*, pages 3–11. AAAI Press.

Antoine Bosselut, Hannah Rashkin, Maarten Sap, Chaitanya Malaviya, Asli Çelikyilmaz, and Yejin Choi. 2019. COMET: commonsense transformers for automatic knowledge graph construction. In *Proceedings of the 57th Conference of the Association for Computational Linguistics, ACL 2019, Florence, Italy, July 28- August 2, 2019, Volume 1: Long Papers*, pages 4762–4779. Association for Computational Linguistics.

Samuel R. Bowman, Gabor Angeli, Christopher Potts, and Christopher D. Manning. 2015. A large annotated corpus for learning natural language inference. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing, EMNLP 2015, Lisbon, Portugal, September 17-21, 2015*, pages 632–642. The Association for Computational Linguistics.

Selmer Bringsjord, Konstantine Arkoudas, and Paul Bello. 2006. Toward a general logicist methodology for engineering ethically correct robots. *IEEE Intell. Syst.*, 21(4):38–44.

Shuang Chen, Jinpeng Wang, Feng Jiang, and Chin-Yew Lin. 2020. Improving entity linking by modeling latent entity type information. In *The Thirty-Fourth AAAI Conference on Artificial Intelligence, AAAI 2020, The Thirty-Second Innovative Applications of Artificial Intelligence Conference, IAAI 2020, The Tenth AAAI Symposium on Educational Advances in Artificial Intelligence, EAAI 2020, New York, NY, USA, February 7-12, 2020*, pages 7529–7537. AAAI Press.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT 2019, Minneapolis, MN, USA, June 2-7, 2019, Volume 1 (Long and Short Papers)*, pages 4171–4186. Association for Computational Linguistics.

Li Du, Xiao Ding, Kai Xiong, Ting Liu, and Bing Qin. 2021. Excar: Event graph knowledge enhanced explainable causal reasoning. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing, ACL/IJCNLP 2021, (Volume 1: Long Papers), Virtual Event, August 1-6, 2021*, pages 2354–2363. Association for Computational Linguistics.

Maxwell Forbes, Jena D. Hwang, Vered Shwartz, Maarten Sap, and Yejin Choi. 2020. Social chemistry 101: Learning to reason about social and moral norms. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing, EMNLP 2020, Online, November 16-20, 2020*, pages 653–670. Association for Computational Linguistics.

Bernard Gert and Joshua Gert. 2020. The Definition of Morality. In Edward N. Zalta, editor, *The Stanford Encyclopedia of Philosophy*, Fall 2020 edition. Metaphysics Research Lab, Stanford University.

Chuan Guo, Geoff Pleiss, Yu Sun, and Kilian Q. Weinberger. 2017. On calibration of modern neural networks. In *Proceedings of the 34th International Conference on Machine Learning, ICML 2017, Sydney, NSW, Australia, 6-11 August 2017*, volume 70 of *Proceedings of Machine Learning Research*, pages 1321–1330. PMLR.

Ari Holtzman, Jan Buys, Li Du, Maxwell Forbes, and Yejin Choi. 2020. The curious case of neural text degeneration. In *8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, April 26-30, 2020*. OpenReview.net.

Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Veselin Stoyanov, and Luke Zettlemoyer. 2020. BART: denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, ACL 2020, Online, July 5-10, 2020*, pages 7871–7880. Association for Computational Linguistics.

Bill Yuchen Lin, Xinyue Chen, Jamin Chen, and Xiang Ren. 2019. Kagnet: Knowledge-aware graph networks for commonsense reasoning. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing, EMNLP-IJCNLP 2019, Hong Kong, China, November 3-7, 2019*, pages 2829–2839. Association for Computational Linguistics.

Zhouhan Lin, Minwei Feng, Cícero Nogueira dos Santos, Mo Yu, Bing Xiang, Bowen Zhou, and Yoshua Bengio. 2017. A structured self-attentive sentence embedding. In *5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Conference Track Proceedings*. OpenReview.net.

Jiangming Liu, Matt Gardner, Shay B. Cohen, and Mirella Lapata. 2020. Multi-step inference for reasoning over paragraphs. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing, EMNLP 2020, Online, November 16-20, 2020*, pages 3040–3050. Association for Computational Linguistics.

Yang Liu, Chengjie Sun, Lei Lin, and Xiaolong Wang. 2016. Learning natural language inference using bidi-

rectional LSTM model and inner-attention. *CoRR*, abs/1605.09090.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Roberta: A robustly optimized BERT pretraining approach. *CoRR*, abs/1907.11692.

Nicholas Lourie, Ronan Le Bras, and Yejin Choi. 2021. SCRUPLES: A corpus of community ethical judgments on 32, 000 real-life anecdotes. In *Thirty-Fifth AAAI Conference on Artificial Intelligence, AAAI 2021, Thirty-Third Conference on Innovative Applications of Artificial Intelligence, IAAI 2021, The Eleventh Symposium on Educational Advances in Artificial Intelligence, EAAI 2021, Virtual Event, February 2-9, 2021*, pages 13470–13479. AAAI Press.

Shangwen Lv, Daya Guo, Jingjing Xu, Duyu Tang, Nan Duan, Ming Gong, Linjun Shou, Daxin Jiang, Guihong Cao, and Songlin Hu. 2020. Graph-based reasoning over heterogeneous external knowledge for commonsense question answering. In *The Thirty-Fourth AAAI Conference on Artificial Intelligence, AAAI 2020, The Thirty-Second Innovative Applications of Artificial Intelligence Conference, IAAI 2020, The Tenth AAAI Symposium on Educational Advances in Artificial Intelligence, EAAI 2020, New York, NY, USA, February 7-12, 2020*, pages 8449–8456. AAAI Press.

Nikita Nangia, Adina Williams, Angeliki Lazaridou, and Samuel R. Bowman. 2017. The repeval 2017 shared task: Multi-genre natural language inference with sentence representations. In *Proceedings of the 2nd Workshop on Evaluating Vector Space Representations for NLP, RepEval@EMNLP 2017, Copenhagen, Denmark, September 8, 2017*, pages 1–10. Association for Computational Linguistics.

Nils Reimers and Iryna Gurevych. 2019. Sentence-bert: Sentence embeddings using siamese bert-networks. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing, EMNLP-IJCNLP 2019, Hong Kong, China, November 3-7, 2019*, pages 3980–3990. Association for Computational Linguistics.

Francesca Rossi and Nicholas Mattei. 2019. Building ethically bounded AI. In *The Thirty-Third AAAI Conference on Artificial Intelligence, AAAI 2019, The Thirty-First Innovative Applications of Artificial Intelligence Conference, IAAI 2019, The Ninth AAAI Symposium on Educational Advances in Artificial Intelligence, EAAI 2019, Honolulu, Hawaii, USA, January 27 - February 1, 2019*, pages 9785–9789. AAAI Press.

Rafal Rzepka and Kenji Araki. 2005. What statistics could do for ethics?: The idea of common sense processing based safety valve. In *AAAI Fall Symposium on Machine Ethics, Technical Report FS-05-06*, pages 85–87. AAAI.

Ari Schlesinger, Kenton P. O'Hara, and Alex S. Taylor. 2018. Let's talk about race: Identity, chatbots, and AI. In *Proceedings of the 2018 CHI Conference on Human Factors in Computing Systems, CHI 2018, Montreal, QC, Canada, April 21-26, 2018*, page 315. ACM.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, December 4-9, 2017, Long Beach, CA, USA*, pages 5998–6008.

Guoyin Wang, Chunyuan Li, Wenlin Wang, Yizhe Zhang, Dinghan Shen, Xinyuan Zhang, Ricardo Henao, and Lawrence Carin. 2018. Joint embedding of words and labels for text classification. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics, ACL 2018, Melbourne, Australia, July 15-20, 2018, Volume 1: Long Papers*, pages 2321–2331. Association for Computational Linguistics.

Zhiguo Wang, Patrick Ng, Xiaofei Ma, Ramesh Nallapati, and Bing Xiang. 2019. Multi-passage BERT: A globally normalized BERT model for open-domain question answering. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing, EMNLP-IJCNLP 2019, Hong Kong, China, November 3-7, 2019*, pages 5877–5881. Association for Computational Linguistics.

Marty J. Wolf, Keith W. Miller, and Frances S. Grodzinsky. 2017. Why we should have seen that coming: comments on microsoft's tay "experiment, " and wider implications. *SIGCAS Comput. Soc.*, 47(3):54–64.

Yueh-Hua Wu and Shou-De Lin. 2018. A low-cost ethics shaping approach for designing reinforcement learning agents. In *Proceedings of the Thirty-Second AAAI Conference on Artificial Intelligence, (AAAI-18), the 30th innovative Applications of Artificial Intelligence (IAAI-18), and the 8th AAAI Symposium on Educational Advances in Artificial Intelligence (EAAI-18), New Orleans, Louisiana, USA, February 2-7, 2018*, pages 1687–1694. AAAI Press.

Wei Yang, Yuqing Xie, Luchen Tan, Kun Xiong, Ming Li, and Jimmy Lin. 2019. Data augmentation for BERT fine-tuning in open-domain question answering. *CoRR*, abs/1904.06652.

Hongming Zhang, Xin Liu, Haojie Pan, Yangqiu Song, and Cane Wing-Ki Leung. 2020. ASER: A large-scale eventuality knowledge graph. In *WWW '20: The Web Conference 2020, Taipei, Taiwan, April 20-24, 2020*, pages 201–211. ACM / IW3C2.

Bias at a Second Glance: A Deep Dive into Bias for German Educational Peer-Review Data Modeling

Thiemo Wambsganss*[1], Vinitra Swamy*[1], Roman Rietsche[2], Tanja Käser[1]

[1] EPFL, Lausanne, Switzerland

{thiemo.wambsganss, vinitra.swamy, tanja.kaeser}@epfl.ch

[2] University of St.Gallen, St.Gallen, Switzerland

roman.rietsche@unisg.ch

Abstract

Natural Language Processing (NLP) has become increasingly utilized to provide adaptivity in educational applications. However, recent research has highlighted a variety of biases in pre-trained language models. While existing studies investigate bias in different domains, they are limited in addressing fine-grained analysis on educational and multilingual corpora. In this work, we analyze bias across text and through multiple architectures on a corpus of 9,165 German peer-reviews collected from university students over five years. Notably, our corpus includes labels such as helpfulness, quality, and critical aspect ratings from the peer-review recipient as well as demographic attributes. We conduct a Word Embedding Association Test (WEAT) analysis on (1) our collected corpus in connection with the clustered labels, (2) the most common pre-trained German language models (T5, BERT, and GPT-2) and GloVe embeddings, and (3) the language models after fine-tuning on our collected dataset. In contrast to our initial expectations, we found that our collected corpus does not reveal many biases in the co-occurrence analysis or in the GloVe embeddings. However, the pre-trained German language models find substantial conceptual, racial, and gender bias and have significant changes in bias across conceptual and racial axes during fine-tuning on the peer-review data. With our research, we aim to contribute to the fourth UN sustainability goal (quality education) with a novel dataset, an understanding of biases in natural language education data, and the potential harms of not counteracting biases in language models for educational tasks.

1 Introduction

In recent years, Natural Language Processing (NLP) and Machine Learning (ML) have been extensively used for improving adaptivity and individualization of educational technology (Rosé et al., 2008; Xu et al., 2021). Researchers and practitioners have been developing a plethora of writing support systems (Song et al., 2014; Lauscher et al., 2018) and conversational agents (Ruan et al., 2019; Weber et al., 2021). More generally, there has been a rise in intelligent tutoring systems for educational purposes which provide learners adaptive feedback, e.g., on grammatical structures (White and Rozovskaya, 2020; Katinskaia and Yangarber, 2021; Kerz et al., 2021), language learning (Putra et al., 2021), argumentation (Song et al., 2014; Lauscher et al., 2019), or even empathy skills (Wambsganss et al., 2021).

The technology for language-based personalization in education comes with a cost; a large body of research has been investigating and revealing biases in NLP systems (Bolukbasi et al., 2016; Sun et al., 2019). Bias has been found in multiple steps along the general NLP pipeline including the task setting, training data, pre-trained models (e.g. word embeddings), and fine-tuned algorithms (Schramowski et al.; Sun et al., 2019; Caliskan et al., 2017; Bolukbasi et al., 2016), shedding a darker light on the simple usage of these models for human-centered applications, especially in education. NLP systems containing bias in any of these parts of the modeling pipeline can produce gender, racially, or conceptually biased predictions and amplify biases present in the underlying training sets (e.g., Baker and Hawn (2021); Hutchinson and Mitchell (2019); Sun et al. (2019)). The propagation of gender bias in NLP algorithms poses the danger of reinforcing damaging stereotypes in downstream applications, e.g., for automatic essay scoring (Östling et al., 2013; Yannakoudakis et al., 2011).

While prior research on bias in education has mostly focused on non-language based interaction data, several recent reviews have called for extending the investigations of fine-grained bias analysis on educational corpora (e.g., Baker and Hawn (2021); Blodgett et al. (2020)). Recent work, for

* These authors contributed equally to this work.

Proceedings of the 29th International Conference on Computational Linguistics, pages 1344–1356
October 12–17, 2022.

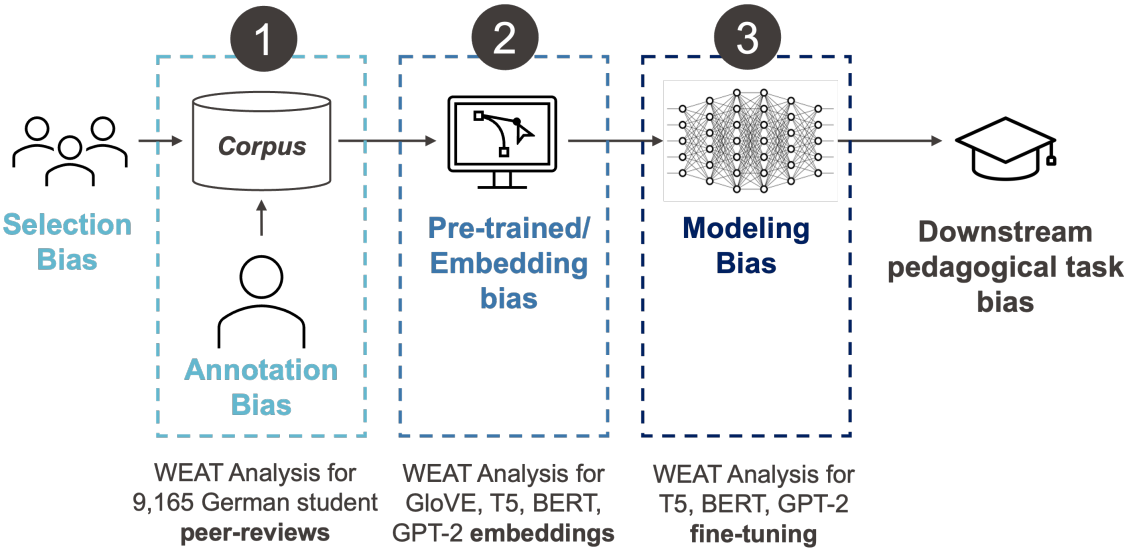

Figure 1: Overview of evaluating biases in educational natural language data along the NLP pipeline for pedagogical downstream tasks following Hovy and Prabhumoye (2021). We analyzed a data set of 9,165 German peer-reviews in combination with the most common pre-trained language models (T5, GPT-2, BERT) and GloVe embeddings before and after fine-tuning with the WEAT analysis for conceptual, racial, and gender biases.

example, has shown negative impact of gender bias on CV screening (Andersson et al., 2021) or of algorithmic racial bias in child welfare programs (Cheng et al., 2022). There are only few works looking at detailed bias in educational natural language data outside of English language corpora and North American context (Baker and Hawn, 2021). For instance, Baker and Hawn (2021) states the need to investigate "the differences in the performance of essay scoring algorithms for different racial groups". However, as they found, "this possibility has not yet been systematically investigated in the published literature" (Baker and Hawn, 2021). Hence, our objective is to address this gap in research and to take a deep dive into one exemplary pedagogical scenario which includes heavy language data: student peer-reviews. Student peer-reviewing is a modern domain-independent pedagogical scenario which has been increasingly used to annotate corpora, analysis of feedback texts with trained models, and provide students feedback with adaptive applications (Nicol, 2014), e.g., for argumentation skill training (Wambsganss et al., 2020a) or empathy skills (Wambsganss et al., 2021; Wambsganß et al., 2022).

In order to conduct a rigorous bias analysis, we collected a novel corpus of 9,165 German student peer-reviews of business model feedback. We relied on Word Embedding Association Test (WEAT) analysis (Caliskan et al., 2017) and the German adaptation of WEAT (Kurpicz-Briki, 2020) as a commonly used methodology to assess conceptual, racial and gender bias in different parts of the NLP pipeline (Hovy and Prabhumoye, 2021). Our methodology for analysing the bias is three-fold (see Figure 1): (1) we analyse the collected corpus for different bias dimensions to find out if the student-writings already come with bias towards the perceived helpfulness of a review, (2) we assess the most common German language models (T5, BERT and GPT-2) as well GloVe embeddings before fine-tuning them on our data, (3) we fine-tune T5, BERT and GPT-2 on our collected data-set and repeat the WEAT analysis to investigate how the representations have been changed.

Contrary to our expectations, we found that our collected corpus does not reveal many biases in using WEAT co-occurrence analysis or GloVe models; however, the pre-trained German language models not only come with substantial conceptual, racial, and gender bias but also seem to increase the bias when fine-tuning on our corpus. Our results suggest to (1) do more fine-grained analyses of bias for subsets of data that are significant, (2) examine the bias in pre-trained models before using them, and (3) investigate multilingual data bias more precisely. Hence, we contribute to literature on bias of educational language data by providing a detailed analysis of one particular but increasingly used pedagogical scenario (peer-reviewing). We

contribute our collected corpus of peer-reviews in German for further analysis and hope to provide researchers and practitioners with a detailed analysis and discussion of bias in NLP for education. Finally, we aim to contribute to the UN sustainability goal four for a high quality education and fair (digital) education for all.

2 Theoretical Background

2.1 Text Bias in Education

Since the 1960s, the problem of bias in educational applications has been noted, and many parts of today's literature on algorithmic bias and fairness have been anticipated (see review and discussion in Hutchinson and Mitchell (2019); Baker and Hawn (2021)). In order to investigate bias, it is important to define what perspective on bias we take, as many definitions exist in the literature. In our research, we "focus on studying algorithmic bias in terms of situations where model performance is substantially better or worse across mutually exclusive groups" (Baker and Hawn, 2021, p. 4). We aim to analyse annotation, embedding, and modeling bias throughout the NLP pipeline for educational downstream tasks (see Figure 1).

Most literature has focused on numerical (non-text) data to analyse bias in educational applications. The literature on bias in education has been mostly investigating differences between race, nationality (students' current national locations), and gender (Baker and Hawn, 2021). For example, Lee and Kizilcec (2020) analysed the differences of an unmodified model to an equity-corrected model for predicting course grade of students. They found that the unmodified models perform worse for underrepresented racial and ethnic groups than for White and Asian students. Anderson et al. (2019) used five different algorithms to discriminate performance between male and female students in a model that predicted six-year college completion. They discovered that male students had greater false negative rates in the algorithms.

Educational corpora play a minor role in investigating bias in NLP systems; the scope of relevant work is limited. The existing research is mostly centered around English language data, missing insights for other languages and cultures (Baker and Hawn, 2021; Sun et al., 2019). One relevant study from Loukina et al. (2019) investigates bias in automated essay grading over essays authored by individuals from six different nations. Their results indicate that training nation-specific models leads to various skews between groups, increasing algorithmic bias when compared to training on all groups together. In a more general scope, research in computational linguistics and NLP have increasingly investigated bias in natural language processing systems, including work on bias in embedding spaces (e.g., Caliskan et al. (2017); Bolukbasi et al. (2016)) language modeling (Lu et al., 2018), coreference resolution (Rudinger et al., 2017), machine translation (Stanovsky et al., 2019), or sentiment analysis (Kiritchenko and Mohammad, 2018).

The analysis of data collected from commonly used pedagogical scenarios in combination with pre-trained language models (especially outside North America) is scarce (e.g., Baker and Hawn (2021); Sun et al. (2019)). We focus on algorithmic bias in the representations of a novel collected data set, in the most common German pre-trained models and German GloVe embeddings, and finally in the fine-tuned models. Our objective goes beyond investigating the effect of NLP bias on educational designs; we aim to contribute to a vision where downstream educational models are unbiased for equitable education (UN Sustainability objective).

2.2 NLP Research on Peer Reviewing

To conduct a rigorous and representative bias analysis of educational data from the field, our objective is a domain-independent pedagogical setting. In this vein, we aim to focus on student peer-reviews, since it is a increasingly growing, modern and digitized educational scenario, which has not only been used to foster factual and conceptual learning goals but also more complex skills such as argumentation (Wambsganss et al., 2020a) or empathy (Wambsganss et al., 2021).

Peer-reviews are defined as the process when students evaluate and make judgments about submissions of their peers and construct written feedback (Nicol, 2014). The structure of this process builds on the principles of the standard peer-reviewing used in academic journals (Ziman, 1974), where each of the papers is assigned to three anonymous reviewers. These reviewers evaluate the quality of the submission and provide feedback about the strengths and weaknesses of the paper and specify ways to improve it (Meadows, 1998).

Peer-reviews come with the advantage that students need to take two different perspectives: one of the feedback provider and one of the feedback

receiver. The role of feedback provider enables students to practice their critical thinking skills, apply criteria and reflect on their own work. Receiving feedback supports the students to focus more on areas that need improvement and develop a reader´s perspective (Wu and Schunn, 2021). Utilizing peer-reviews enables students to receive timely feedback in both small-scale and large-scale classes, where feedback from lecturers or instructor assessment is often too late to be implemented. With the pedagogical scenario of edit history, it is even possible to directly apply feedback to improve final version of submission (Higgins et al., 2001). Additionally, with the rise of educational technologies, the peer-review paradigm is increasingly implemented in common learning management systems (e.g. Canvas, Blackboard) and is used by large MOOC providers. However, the direct use of unfiltered feedback by non-experts poses the concern that students are exposed to biases and inaccuracies (Double et al., 2020). Therefore, past research has already started to analyze peer-review data with standard NLP techniques.

For example, Misiejuk and Wasson (2021) use NLP to understand students' perceptions of peer-reviews. Wu et al. (2020) measure the impact of feedback features such as identification, explanations, and suggestions on the likelihood of that the feedback gets implemented. Xiao et al. (2020) trained different models based on RNNs, CNNs, LSTMs, GloVe, and BERT to detecting problem statements in peer assessments. Zingle et al. (2019) use CNNs and LSTMs to detect actionable suggestions in peer assessments. Researchers have also started to develop downstream applications based on model predictions to provide adaptive learning feedback. Ramachandran et al. (2017) created a tool for automated assessment of the quality of peer-reviews. Bauman et al. (2020) designed a recommender framework which uses a trained model to identify aspects of the review texts that correspond to peer-review helpfulness scores. Several papers use student peer-review data for annotating arguments for argumentation mining (e.g., Wambsganss et al. (2020b)) or cognitive and emotional empathy structures for empathy modeling (Wambsganss et al., 2021) to provide students with writing assistance in learning applications.

Although NLP research exists on and around peer-review data, there are only a handful of investigations on bias along the NLP pipeline (Patchan et al., 2018). Hence, we propose to investigate which biases occur in education data along the NLP pipeline and in particular in our context in peer-reviews.

3 Methodology

3.1 Data Collection

Since there are not many suitable corpora available to analyse bias in student peer-reviews that a) contain a large amount of student-written text in one particular domain (e.g., business model feedback), b) consist of a sufficient size to represent different nuances of characteristics in a balanced fashion and c) come with additional scores such as review helpfulness rated by the receiver of the review or demographics for additional analysis (e.g., gender), we decided to collect our own longitudinal data set.

The peer-reviews of our novel dataset were collected over five years at a university in the German speaking area of Europe.[1] Overall, we compiled a corpus of 9,165 student-generated peer-reviews in which students provide each other feedback on previously developed business models. The peer-reviewing process was conducted in a double-blind manner; thus the feedback provider and receiver were anonymous. Alongside the text data, we collected subsets of ratings regarding the review helpfulness. This data was collected within the peer-review process; when the authors of the assignment receive the peer-reviews, they performed peer backward assessment (Patchan et al., 2016). In peer backward assessment, students rate the four items (based on Li et al. (2010)): (1) *"The feedback I got from the reviewer was helpful"* (2) *"The feedback I got from the reviewer was high quality"* (3) *"The reviewer was able to identify critical aspects in my submission"* and (4) *"The reviewer was able to provide constructive suggestions on his stated critical aspects"* on a 7-point Likert Scale from totally disagree (1) to totally agree (7), with 4 as a neutral value. Additionally, we captured gender and the year of birth of the review writers.

3.2 Data Characteristics

Our dataset consists of first-year master's students majoring in business innovation. The majority of students have German as their native language. The data was collected from 2015 to 2019 and include 9,165 reviews from 610 unique reviewers and 607

[1]The data was collected based on the ethical guidelines of our university.

reviewees. We collected demographic data at the beginning of each semester; the student population has an average age of 24.6 years old with a standard deviation of 1.7 years. The average percentage of female students across five years is 37.7%. Students wrote approximately 9 peer reviews per course with an average length of 220 words.

3.3 Model Architecture

We examine four German variations of language model architectures in this paper, chosen for their popularity on downstream tasks: GloVe, BERT, T5, and GPT-2. For GloVe architectures, we train the model from scratch for 100 epochs each, using a vector size of 300, window size of 15, and 8 threads for parallelization (Pennington et al., 2014). We obtain all three pre-trained models from HuggingFace (Wolf et al., 2019) and fine-tune each model for 10 epochs on a Tesla V100 GPU with batch size 8. For GermanBERT, we fine-tune the model using a standard masked language model training objective with masking rate of 15% (Chan et al., 2020). For German T5, we fine-tune the model using the translation task, translating peer-review text from English to German[2] with max source token length of 128 and global seed 42. The pre-trained multilingual T5 model was fine-tuned on the German MLSum dataset (Xue et al., 2020) before being used for our analysis. German GPT-2 was fine-tuned on the text generation objective with block size 128, and 600 warm-up steps (Radford et al., 2019). More details can be found directly in our supplementary code repository.

3.4 Bias Analysis

To assess bias along the NLP pipeline suggested by Hovy and Prabhumoye (2021), we rely on the Word Embedding Association Test (WEAT) proposed by Caliskan et al. (2017). WEAT assesses the extent to which word embeddings represent certain cultural biases. The inspiration for the WEAT analysis is grounded in psychological theory as an extension of the *Implicit Association Test*, used to measure bias in humans (Greenwald et al., 1998). WEAT calculates the semantic similarity between two sets of target words (e.g., male vs. female names) and two sets of attribute words using word embeddings (e.g., career vs. family). Table 1 indicates the nine WEAT tests and their corresponding targets and attributes.

[2]Translations were obtained through the Google Translate API and corrected by a native English speaker.

Kurpicz-Briki (2020) apply the same concept to three other languages (German, Italian, and Spanish) and adapted and evaluated four WEAT tests for German. The multilingual name adaptations were created by experts examining the census data for popular names from each country of origin and creating word lists for *Male vs. Female* names, as well as *Native vs. Foreign* names (to replace the *European-American vs. African-American* test originally proposed in WEAT). Kurpicz-Briki (2020) do not present a translation for a tenth test on ageism proposed by Caliskan et al. (2017), so we omit it from our study to not combine differing methodologies. In this work, we present German translations for all nine WEAT tests[3].

We broadly categorize the WEAT tests into the three main dimensions of bias: Racial, Gender, and Conceptual. This is in accordance with the literature on bias in educational data (e.g., Baker and Hawn (2021)). Our categorization helps language model users to have a big picture understanding of how their model performs (i.e. model X is more biased by gender than race) instead of granular statements (i.e. model X finds male names more associated with career than with family). The groupings are detailed further in Table 1, with each category consisting of three tests.

To quantitatively compare across WEAT analyses, we use the metric proposed by Caliskan et al. (2017). *Effect size* is a normalized measure of the distance between the two distributions of associations and targets, calculated as follows:

$$\frac{mean_{x \in X} s(x, A, B) - mean_{y \in Y} s(y, A, B)}{std_{w \in X \cup Y} s(w, A, B)}$$

where X and Y are two sets of target words of equal size, A, B are two sets of attribute words, and $s(w, A, B)$ measures the association of embeddings of the target word w with the attribute words.

4 Results on Bias Analysis

We present results across three stages of the bias pipeline (highlighted in Figure 1): (1) a WEAT co-occurrence analysis examining bias directly in the peer-review corpora, (2) an embedding space analysis using a GloVe architecture trained on the peer-review data, (3) an analysis of the three most

[3]The WEAT words not found in Kurpicz-Briki (2020)'s study were translated from the English WEAT through DeepL and corrected by two native German speakers.

Bias	#	Targets	Attributes
Conceptual	1	Flowers vs. Insects	Pleasant vs. Unpleasant
	2	Instruments vs. Weapons	Pleasant vs. Unpleasant
	9	Mental vs. Physical Disease	Temporary vs. Permanent
Racial	3	Native vs. Foreign Names	Pleasant vs. Unpleasant
	4	Native vs. Foreign Names (v2)	Pleasant vs. Unpleasant
	5	Native vs. Foreign Names (v2)	Pleasant vs. Unpleasant (v2)
Gender	6	Male vs. Female Names	Career vs. Family
	7	Math vs. Arts	Male vs. Female Terms
	8	Science vs. Arts	Male vs. Female Terms

Table 1: Overview of our proposed measured bias categories (conceptual, race, and gender) for the WEAT analysis. WEAT compares the association between two different target word lists (i.e. Math vs. Arts) to attribute word lists (i.e. Male vs. Female terms). # indicates the original WEAT test number (Caliskan et al., 2017).

popular German language models, before and after fine-tuning on the peer-review corpora.

4.1 Bias in the Peer-Review Corpus

In the first experiment, we conduct a WEAT co-occurrence analysis as proposed by Spliethöver and Wachsmuth (2020); Caliskan et al. (2017). Our aim is to measure the bias present in the raw corpus without the confounding factors of model architecture and pre-existing bias in embeddings. Therefore, this test identifies specific occasions in the text where target words are present in close proximity to attribute words. The neighborhood of proximity can be defined as within the same sentence or within the same review, but the likelihood of a review mentioning different topics over several sentences is significant and we do not want to conflate circumstantial correlation with bias. Therefore, we only examine co-occurrence by sentence.

We do not find significant results across any of the nine WEAT tests, with only six co-occurrences identified in total across 9,165 peer-reviews[4].

In line with existing research, we found that the peer backward assessment ratings have a large skew towards positive ratings, with over 50% of the data residing in points 6 and 7 across all feedback questions asked. Student judgements about helpfulness may be dependent on the review sentiment (Patchan et al., 2018). Due to this positive skew, we select ratings < 6 as a *low* rating denomination and ratings >= 6 as a *high* rating across 4 reviewer axes: helpfulness, critical aspects, constructive suggestions, and overall quality. Table 2 indicates the number of entries in each subset.

Subsets	High (>= 6)	Low (< 6)
Constructive Suggestions	5656	3509
Critical Aspects	5514	3651
Helpfulness	5886	3279
Quality	5391	3774

Table 2: Distribution of 9165 student peer-reviews in each reviewer rating subset. The scores have been rated by the receiver of the review. High and Low ratings correspond to scores on the Likert 7-point scale.

We conducted a WEAT co-occurrence analysis across different review rating criteria (quality, critical aspects, helpfulness, constructive) and WEAT target-attribute pairings, as inspired by Spliethöver and Wachsmuth (2020). Examining the overall corpus, none of the subsets are able to identify significant bias. However, tests with ratings of high quality do find minimal instances of bias in test nine (mental vs. physical disease) while the other rating criteria find test nine co-occurrences in their low rating groups (critical aspects, constructive suggestions, helpfulness). Due to the very few co-occurrences present (two instances found in 9,165 reviews, each containing around ten sentences), this difference could be attributed to noise.

In summary, from a preliminary examination of the raw text corpora through the lens of the WEAT co-occurrence tests, we are not able to uncover any significant bias.

4.2 Bias in Embeddings

In a second experiment, we implement a GloVe model trained on the raw text corpora. The data is pre-processed to remove punctuation, stop-words, and HTML tags. We train the GloVe model for 100

[4]For the interested reader, the results on all WEAT analyses can be found in the Appendix. Moreover, the peer-review corpus and the code for the conducted tests can be found in our Github repo: https://github.com/epfl-ml4ed/bias-at-a-second-glance.

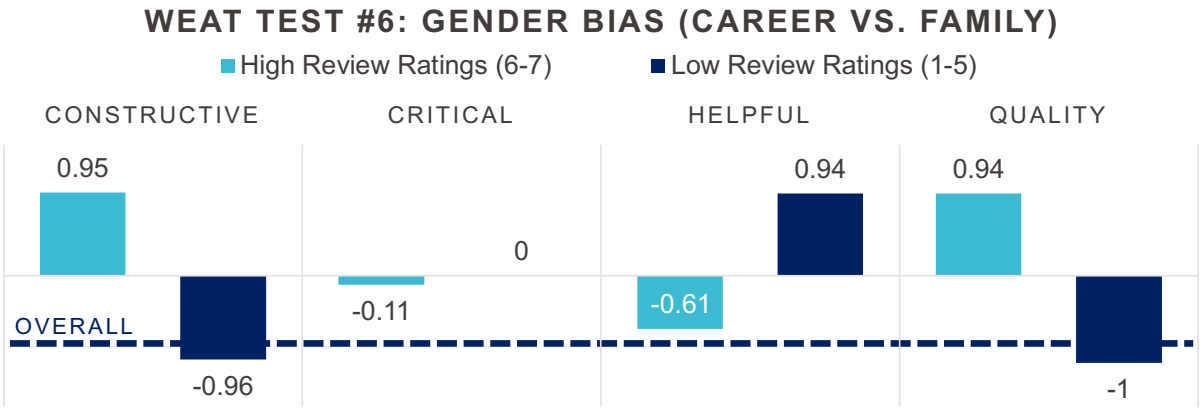

Figure 2: Overview on significant gender bias for WEAT Test 6 in GloVe models between the different review rating subsets. We examine the total effect size as well as each of the subsets (high and low scores for constructive suggestions, critical aspects, helpfulness, and quality). Effect size is normalized between -1.0 to +1.0.

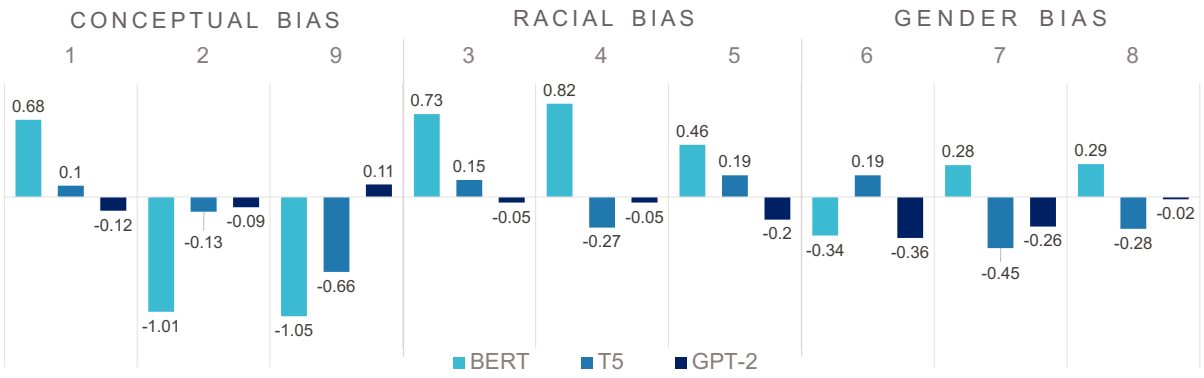

Figure 3: The **differences in effect size** caused by fine-tuning in comparison with pre-trained model results across all nine WEAT tests on three models (BERT, T5, GPT-2). Effect size is normalized between -1.0 to +1.0.

epochs, which consisted of about 20 minutes of training time on an 8-core Apple M1 CPU. WEAT Test 6 (Male vs. Female Names :: Career vs. Family) is the only test able to uncover bias. The other eight WEAT tests examined are out-of-vocabulary for the GloVe model. This highlights a distinct disadvantage in training models only on a distinct set of texts instead of leveraging larger language models and adapting them for a certain task.

Using the entire corpus, we identified a negative bias of -0.748, stating that *female names* are more related to *career* terms than *male names*. Effect sizes are normalized from +1.0 to -1.0, so a bias of 0.75 is very significant. Another equivalent statement is that *male names* are more related to *family* terms than *female names*.

In addition to the overall analysis, we examined subsets of the corpus based on peer backward assessment review ratings classified into high and low rating categories which are shown in Figure 4. Eight GloVe models are trained on each subset of the data (i.e. high quality, low helpful). We see that constructive suggestions and quality have the most difference in bias between the ratings with high scores and low scores. In the reviews considered highly constructive and high quality, *male names* are more associated with *career* than *female names*; in their corresponding minimally constructive and low quality counterparts, *female names* are more associated with *career*. This result is notable because it reveals the opposite bias found in these subsets as contrasted with the bias measured over a GloVe model trained on the whole corpus. High and low ratings across critical aspects do not uncover significant differences, but high helpful and low helpful ratings do identify the same bias as the initial GloVe model (*female names* are more associated with *career* than *male names*).

Overall, only one test on the gender axis is able to uncover bias using traditional word embeddings (GloVe). Comparing bias in a GloVe model trained on the overall corpus and a GloVe model high qual-

ity review subsets find different conclusions, emphasizing the importance of granularity in bias analysis. This result is in line with the previous co-occurrence study.

4.3 Bias in German Language Models

In a third experiment, we examined three popular transformer-based German language models for bias: GermanBERT, German T5, and German GPT-2, extracted from the HuggingFace library (Wolf et al., 2019). Few works analyze bias in pre-trained German language models (Kraft, 2021; Ahn and Oh, 2021), usually referencing one model at a time instead of a comparative study. Therefore, we aim to address this gap in research. Further details on how these models were trained and the fine-tuning objectives can be found in Section 3.3.

Our analysis consists of three parts: (1) Conduct WEAT analysis to measure the underlying bias in the pre-trained German models, (2) fine-tune three models on our peer-review text corpora, and (3) measure the change in bias across the WEAT tests. The WEAT scores for pre-trained and fine-tuned models can be found in Appendix Tables 6 and 7.

We initially conduct the WEAT analysis on the pre-trained language models and find that GermanBERT and German GPT-2 are significantly biased across all three tests on the racial axis (averaging 1.25 and 1.75 in effect size respectively), finding *native names* generally more associated with *pleasant* terms than *foreign* names. German T5 and German GPT-2 are biased across the conceptual axis (averaging 0.38 and 0.51 in effect size respectively), with positive effect sizes for all three tests.

We then pre-process the input data and fine-tune the language models. Figure 3 identifies the differences in the WEAT effect sizes across the three axes of bias (nine WEAT tests) after fine-tuning. The gender axis has the least change in score across all three models, showing that fine-tuning on our data does not significantly impact the underlying gender bias in the pre-trained model. However, GermanBERT is highly affected by fine-tuning across the conceptual and racial axes across tests 1-5, and 9. Model T5 is significantly impacted in conceptual test 9 (mental vs. physical disease) and GPT-2 bias results are only minutely impacted by fine-tuning.

We additionally fine-tune the language models with the eight subsets of the ratings, as per the same experiment in Figure 4. As we analyze these results, we find the bias does not vary significantly across model subsets. For a point of comparison, we examine WEAT 6 (gender bias across career and family attributes), found as the most significant test in the GloVe model WEAT analysis. We hypothesized that the language model that was least susceptible to fine-tuning (GPT-2) might show stronger variations across subsets, but our results indicate a very small change in bias of at maximum 0.03, with a baseline of 0.61 for the GPT-2 WEAT 6 effect size on the total corpus (Figure 4).

Moreover, we controlled for different subsets concerning the male or female authors in terms of bias along the NLP pipeline in the last three years of our corpus. Nevertheless, we did not find any significant results in 1) the co-occurrence analysis for gender-separate subsets, 2) for the GloVe embeddings, and 3) for the fine-tuning on BERT, T5 and GPT-2. The exact results for the fine-tuned models can be found in Tables 3 and 4.

	Overall (2017-2019)	**Male Authors**	**Female Authors**
GloVe WEAT 6	-0.91	0.39	0

Table 3: WEAT results for the GloVe embeddings across subsets of male and female authors. Only WEAT 6 is found significant.

Despite the previous two experiments not finding pervasive bias in the corpora, pre-trained German language models are inherently significantly biased, and fine-tuning using language models uncovers different, significant bias. BERT is the most susceptible to changes in bias of the three architectures.

5 Discussion and Conclusions

We collected and analyzed a novel corpus of 9,165 German peer-reviews, including the students' gender and peer-reviewed helpfulness ratings, to perform a granular bias analysis along the NLP pipeline. Our aim was to shed light on the popular pedagogical scenario of peer-reviewing, where NLP and ML are extensively used for improving adaptivity. Our results did not show any significant bias across any of the nine WEAT tests for our corpora or the collected ratings. For the German GloVe embeddings, we only found a significant gender bias for test 6 involving male and female names associated with career and family. Importantly, in common pre-trained German Language models (BERT, T5, GPT-2), we found substantial conceptual, racial, and gender bias. We saw that

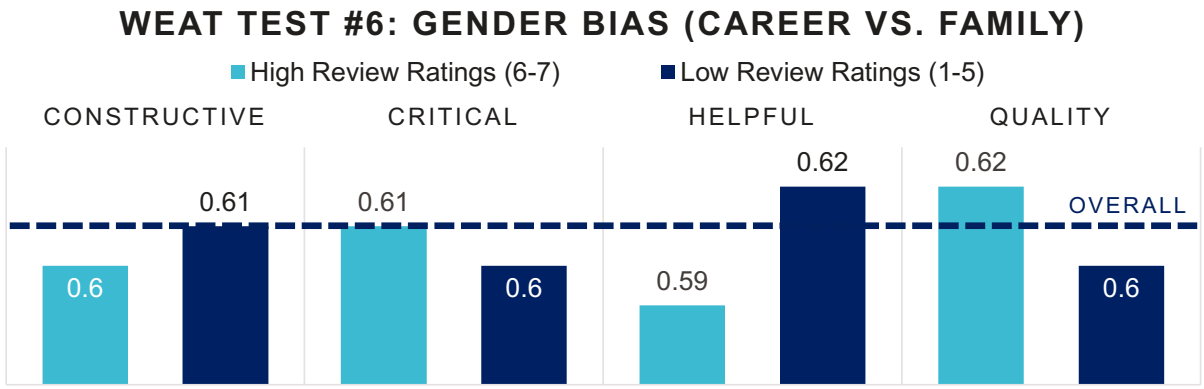

Figure 4: Overview on gender bias (WEAT 6) in **fine-tuned GPT-2** between the different review rating subsets. Effect size is normalized between -1.0 to +1.0.

		Conceptual			Racial			Gender		
Model	**Author**	1	2	9	3	4	5	6	7	8
German	male	-0.11	0.58	0.17	0.47	0.47	0.66	0.62	0.41	-0.23
BERT	female	-0.11	0.58	0.18	0.47	0.47	0.67	0.62	0.41	-0.23
German	male	0.36	0.08	0.22	0.53	0.31	-0.33	-0.46	0.49	0.03
T5	female	0.35	0.09	0.21	0.53	0.31	-0.35	-0.46	0.51	0.02
German	male	0.07	0.11	0.59	0.62	0.62	0.63	0.61	0.01	-0.29
GPT-2	female	0.07	0.11	0.59	0.62	0.62	0.64	0.61	0.01	-0.29

Table 4: WEAT analysis from subsets of male authors and female authors used to fine-tune three German language models. Most values remain the same across both subsets at two degrees of precision.

after fine-tuning on our corpora, the language models uncovered other significant bias that were not present before fine-tuning. BERT was most susceptible to bias changes. Hence, we contribute a perspective in how to reveal and investigate bias in educational corpora for educational downstream tasks, as well as initial actionable considerations for educational data scientists intending to use our corpus.

Are results add insights to the literature about gender bias in educational data modelling (e.g., Anderson et al. (2019)), in embedding spaces (e.g., Bolukbasi et al. (2016)), and in language modelling (e.g., Lu et al. (2018)). In our research, we built on the findings from these different perspectives along the NLP pipeline and conducted a fine-granular analysis for German peer-reviews by analyzing the texts, the qualitative review scores, demographics (i.e, gender), embeddings, and the most common pre-trained language models before and after fine-tuning.

Our results suggest three main directions. First, bias can emerge and change along the NLP-pipeline. Detecting a certain bias in the corpora or in pre-trained language models does not necessitate a connection to bias in fine-tuned models for downstream tasks. Thus, it is necessary to have more in-depth analyses of bias not only along the entire NLP pipeline but also for subsets of data that are significant. Second, it is important to examine the bias in pre-trained models before using them. And third, more investigations on multilingual data bias are necessary. Therefore, we contribute to literature on bias of educational language data by providing a fine-grained analysis of one particular but increasingly used pedagogical scenario (peer-reviews). We contribute our collected corpus of peer-reviews in German for further analysis and hope to provide researchers and practitioners with a detailed analysis and discussion of bias in NLP for education. Finally, we aim to contribute to the UN sustainability goal four for a high quality education and fair (digital) education for all.

5.1 Ethical Considerations

We note that this research was conducted by a mixed team of authors with Western European, Indian, North-American, female and male backgrounds.

References

Jaimeen Ahn and Alice Oh. 2021. Mitigating language-dependent ethnic bias in bert. *arXiv preprint arXiv:2109.05704*.

Henry Anderson, Afshan Boodhwani, and R. Baker. 2019. Assessing the fairness of graduation predictions. In *The 12th International Conference on Educational Data Mining*.

Emma Rachel Andersson, Carolina E. Hagberg, and Sara Hägg. 2021. Gender bias impacts top-merited candidates. *Frontiers in Research Metrics and Analytics*, 6.

Ryan S Baker and Aaron Hawn. 2021. Algorithmic bias in education. *International Journal of Artificial Intelligence in Education*, pages 1–41.

Konstantin Bauman, Roman Rietsche, and Matthias Söllner. 2020. *Supporting Students in the Peer-Review Process by Recommending Features of Written Feedback that Should be Improved*. Statistical Challenges in Electronic Commerce Research (SCECR), Virtual Conference.

Su Lin Blodgett, Solon Barocas, Hal Daumé III, and Hanna Wallach. 2020. Language (Technology) is Power: A Critical Survey of "Bias" in NLP. pages 5454–5476. Association for Computational Linguistics (ACL).

Tolga Bolukbasi, Kai-Wei Chang, James Y Zou, Venkatesh Saligrama, and Adam T Kalai. 2016. Man is to computer programmer as woman is to homemaker? debiasing word embeddings. *Advances in neural information processing systems*, 29.

Aylin Caliskan, Joanna J Bryson, and Arvind Narayanan. 2017. Semantics derived automatically from language corpora contain human-like biases. *Science*, 356(6334):183–186.

Branden Chan, Stefan Schweter, and Timo Möller. 2020. German's next language model. *arXiv preprint arXiv:2010.10906*.

Hao-Fei Cheng, Logan Stapleton, Anna Kawakami, Venkatesh Sivaraman, Yanghuidi Cheng, Diana Qing, Adam Perer, Kenneth Holstein, Zhiwei Steven Wu, and Haiyi Zhu. 2022. How child welfare workers reduce racial disparities in algorithmic decisions. In *CHI Conference on Human Factors in Computing Systems*, pages 1–22.

Kit S. Double, Joshua A. McGrane, and Therese N. Hopfenbeck. 2020. The Impact of Peer Assessment on Academic Performance: A Meta-analysis of Control Group Studies. *Educational Psychology Review*, 32(2):481–509.

Anthony G Greenwald, Debbie E McGhee, and Jordan LK Schwartz. 1998. Measuring individual differences in implicit cognition: the implicit association test. *Journal of personality and social psychology*, 74(6):1464.

Richard Higgins, Peter Hartley, and Alan Skelton. 2001. Getting the message across: the problem of communicating assessment feedback. *Teaching in Higher Education*, 6(2):269–274.

Dirk Hovy and Shrimai Prabhumoye. 2021. Five sources of bias in natural language processing. *Language and Linguistics Compass*, 15(8):e12432.

Ben Hutchinson and Margaret Mitchell. 2019. 50 years of test (un)fairness: Lessons for machine learning. In *Proceedings of the Conference on Fairness, Accountability, and Transparency*, FAT* '19, page 49–58, New York, NY, USA. Association for Computing Machinery.

Anisia Katinskaia and Roman Yangarber. 2021. Assessing grammatical correctness in language learning. In *Proceedings of the 16th Workshop on Innovative Use of NLP for Building Educational Applications*, pages 135–146, Online. Association for Computational Linguistics.

Elma Kerz, Daniel Wiechmann, Yu Qiao, Emma Tseng, and Marcus Ströbel. 2021. Automated classification of written proficiency levels on the CEFR-scale through complexity contours and RNNs. In *Proceedings of the 16th Workshop on Innovative Use of NLP for Building Educational Applications*, pages 199–209, Online. Association for Computational Linguistics.

Svetlana Kiritchenko and Saif Mohammad. 2018. Examining gender and race bias in two hundred sentiment analysis systems. In *Proceedings of the Seventh Joint Conference on Lexical and Computational Semantics*, pages 43–53, New Orleans, Louisiana. Association for Computational Linguistics.

Angelie Kraft. 2021. Triggering models: Measuring and mitigating bias in german language generation.

Mascha Kurpicz-Briki. 2020. Cultural differences in bias? origin and gender bias in pre-trained german and french word embeddings.

Anne Lauscher, Goran Glavaš, and Kai Eckert. 2018. ArguminSci: A tool for analyzing argumentation and rhetorical aspects in scientific writing. In *Proceedings of the 5th Workshop on Argument Mining*, pages 22–28, Brussels, Belgium. Association for Computational Linguistics.

Anne Lauscher, Goran Glavaš, and Simone Paolo Ponzetto. 2019. An Argument-Annotated Corpus of Scientific Publications. pages 40–46.

Hansol Lee and René F Kizilcec. 2020. Evaluation of fairness trade-offs in predicting student success. *arXiv preprint arXiv:2007.00088*.

Lan Li, Xiongyi Liu, and Allen L. Steckelberg. 2010. Assessor or assessee: How student learning improves by giving and receiving peer feedback. *British Journal of Educational Technology*, 41(3):525–536.

Anastassia Loukina, Nitin Madnani, and Klaus Zechner. 2019. The many dimensions of algorithmic fairness in educational applications. In *Proceedings of the Fourteenth Workshop on Innovative Use of NLP for Building Educational Applications*, pages 1–10, Florence, Italy. Association for Computational Linguistics.

Kaiji Lu, Piotr Mardziel, Fangjing Wu, Preetam Amancharla, and Anupam Datta. 2018. Gender bias in neural natural language processing. In *Logic, Language, and Security*.

A. J. Meadows. 1998. *Communicating Research*. Academic Press, San Diego, CA.

Kamila Misiejuk and Barbara Wasson. 2021. Backward evaluation in peer assessment: A scoping review. *Computers & Education*, page 104319.

David Nicol. 2014. Guiding principles for peer review: unlocking learners' evaluative skills. *Advances and Innovations in University Assessment and Feedback*, pages 197–224.

Robert Östling, André Smolentzov, Björn Tyrefors Hinnerich, and Erik Höglin. 2013. Automated Essay Scoring for Swedish. *Proceedings of the Eighth Workshop on Innovative Use of NLP for Building Educational Applications*, pages 42–47.

Melissa M. Patchan, Christian D. Schunn, and Russell J. Clark. 2018. Accountability in peer assessment: examining the effects of reviewing grades on peer ratings and peer feedback. *Studies in Higher Education*, 43(12):2263–2278.

Melissa M. Patchan, Christian D. Schunn, and Richard J. Correnti. 2016. The nature of feedback: How peer feedback features affect students' implementation rate and quality of revisions. *Journal of Educational Psychology*, 108(8):1098–1120.

Jeffrey Pennington, Richard Socher, and Christopher D Manning. 2014. Glove: Global vectors for word representation. In *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*, pages 1532–1543.

Jan Wira Gotama Putra, Simone Teufel, and Takenobu Tokunaga. 2021. Parsing argumentative structure in English-as-foreign-language essays. In *Proceedings of the 16th Workshop on Innovative Use of NLP for Building Educational Applications*, pages 97–109, Online. Association for Computational Linguistics.

Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, Ilya Sutskever, et al. 2019. Language models are unsupervised multitask learners. *OpenAI blog*, 1(8):9.

Lakshmi Ramachandran, Edward F. Gehringer, and Ravi K. Yadav. 2017. Automated Assessment of the Quality of Peer Reviews using Natural Language Processing Techniques. *International Journal of Artificial Intelligence in Education*, 27(3):534–581.

Carolyn Rosé, Yi Chia Wang, Yue Cui, Jaime Arguello, Karsten Stegmann, Armin Weinberger, and Frank Fischer. 2008. Analyzing collaborative learning processes automatically: Exploiting the advances of computational linguistics in computer-supported collaborative learning. *International Journal of Computer-Supported Collaborative Learning*, 3(3):237–271.

Sherry Ruan, Liwei Jiang, Justin Xu, Bryce Joe-Kun Tham, Zhengneng Qiu, Yeshuang Zhu, Elizabeth L. Murnane, Emma Brunskill, and James A. Landay. 2019. QuizBot: A Dialogue-based Adaptive Learning System System for Factual Knowledge. pages 1–13.

Rachel Rudinger, Chandler May, and Benjamin Van Durme. 2017. Social bias in elicited natural language inferences. In *Proceedings of the First ACL Workshop on Ethics in Natural Language Processing*, pages 74–79, Valencia, Spain. Association for Computational Linguistics.

Patrick Schramowski, Cigdem Turan, Nico Andersen, Constantin A. Rothkopf, and Kristian Kersting. Large Pre-trained Language Models Contain Human-like Biases of What is Right and Wrong to Do.

Yi Song, Michael Heilman, Beata Beigman Klebanov, and Paul Deane. 2014. Applying argumentation schemes for essay scoring. In *Proceedings of the First Workshop on Argumentation Mining*, pages 69–78.

Maximilian Spliethöver and Henning Wachsmuth. 2020. Argument from old man's view: Assessing social bias in argumentation. In *Proceedings of the 7th Workshop on Argument Mining*, pages 76–87, Online. Association for Computational Linguistics.

Gabriel Stanovsky, Noah A. Smith, and Luke Zettlemoyer. 2019. Evaluating gender bias in machine translation. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 1679–1684, Florence, Italy. Association for Computational Linguistics.

Xinyu Sun, Maoxin Han, and Juan Feng. 2019. Helpfulness of online reviews: Examining review informativeness and classification thresholds by search products and experience products. *Decision Support Systems*, 124:113099.

Thiemo Wambsganss, Christina Niklaus, Matthias Cetto, Matthias Söllner, Jan Marco Leimeister, and Siegfried Handschuh. 2020a. AL: An Adaptive Learning Support System for Argumentation Skills. In *Proceedings of the 2020 CHI Conference on Human Factors in Computing Systems*, pages 1–14, New York, NY, USA. Association for Computing Machinery.

Thiemo Wambsganss, Christina Niklaus, Matthias Söllner, Siegfried Handschuh, and Jan Marco Leimeister. 2020b. A Corpus for Argumentative Writing Support in German. In *28th International Conference*

on Computational Linguistics (Coling), pages 1–14, Barcelona, Spain.

Thiemo Wambsganss, Christina Niklaus, Matthias Söllner, Siegfried Handschuh, and Jan Marco Leimeister. 2021. Supporting Cognitive and Emotional Empathic Writing of Students. In *59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing*, pages 4063–4077.

Thiemo Wambsganß, Matthias Söllner, Ken Koedinger, and Jan Marco Leimeister. 2022. ELEA: An Adaptive Learning Support System for Empathy Skills. In *Under review at ACM CHI Conference on Human Factors in Computing Systems*, pages 1–18.

Florian Weber, Thiemo Wambsganss, Dominic Rüttimann, and Matthias Söllner. 2021. Pedagogical Agents for Interactive Learning: A Taxonomy of Conversational Agents in Education. In *Forty-Second International Conference on Information Systems*, pages 1–17, Austin, Texas, USA.

Max White and Alla Rozovskaya. 2020. A comparative study of synthetic data generation methods for grammatical error correction. In *Proceedings of the Fifteenth Workshop on Innovative Use of NLP for Building Educational Applications*, pages 198–208, Seattle, WA, USA → Online. Association for Computational Linguistics.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Rémi Louf, Morgan Funtowicz, et al. 2019. Huggingface's transformers: State-of-the-art natural language processing. *arXiv preprint arXiv:1910.03771*.

Chien-Sheng Wu, Steven C.H. Hoi, Richard Socher, and Caiming Xiong. 2020. TOD-BERT: Pre-trained Natural Language Understanding for Task-Oriented Dialogue. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 917–929, Stroudsburg, PA, USA. Association for Computational Linguistics.

Yong Wu and Christian D. Schunn. 2021. The Effects of Providing and Receiving Peer Feedback on Writing Performance and Learning of Secondary School Students. *American Educational Research Journal*, 58(3):492–526.

Yunkai Xiao, Gabriel Zingle, Qinjin Jia, Shoaib Akbar, Yang Song, Muyao Dong, Li Qi, and Edward Gehringer. 2020. Problem Detection in Peer Assessments between Subjects by Effective Transfer Learning and Active Learning. In *International Conference on Educational Data Mining (EDM)*.

Wei Xu, Marvin J. Dainoff, Liezhong Ge, and Zaifeng Gao. 2021. From Human-Computer Interaction to Human-AI Interaction: New Challenges and Opportunities for Enabling Human-Centered AI. pages 1–73.

Linting Xue, Noah Constant, Adam Roberts, Mihir Kale, Rami Al-Rfou, Aditya Siddhant, Aditya Barua, and Colin Raffel. 2020. mt5: A massively multilingual pre-trained text-to-text transformer. *arXiv preprint arXiv:2010.11934*.

Helen Yannakoudakis, Ted Briscoe, and Ben Medlock. 2011. A new dataset and method for automatically grading ESOL texts. *ACL-HLT 2011 - Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, 1:180–189.

John Michael Ziman. 1974. *Public knowledge: An essay concerning the social dimension of science*. Cambridge University Press, Cambridge, UK.

Gabriel Zingle, Balaji Radhakrishnan, Yunkai Xiao, Edward Gehringer, Zhongcan Xiao, Ferry Pramudianto, Gauraang Khurana, and Ayush Arnav. 2019. Detecting Suggestions in Peer Assessments. *International Educational Data Mining Society*.

A Appendix

			Conceptual			Racial			Gender		
	#	+/-	1	2	9	3	4	5	6	7	8
Overall	9165	+/-			1	1	1	1	2		
Quality	5391	+				1	1		1		
		-			1			1	1		
Critical	5514	+			1	1	1		1		
		-						1	1		
Helpful	5886	+			1	1	1		1		
		-						1	1		
Constructive	5656	+			1	1	1		1		
		-						1	1		

Table 5: WEAT co-occurrence analysis across different review rating criteria (quality, critical aspects, helpfulness, constructive) and tests (1-9). This table represents the counts of co-occurrence examples present in the high ratings (+) and low ratings (−) for each criteria.

Bias	#	Targets	Attributes	German BERT	German T5	German GPT-2
	1	Flowers vs. Insects	Pleasant vs. Unpleasant	-0.22	0.61	0.25
Conceptual	2	Instruments vs. Weapons	Pleasant vs. Unpleasant	0.58	0.11	0.15
	9	Mental vs. Physical Disease	Temporary vs. Permanent	0.16	0.5	0.54
	3	Native vs. Foreign Names	Pleasant vs. Unpleasant	0.48	0.44	0.64
Racial	4	Native vs. Foreign Names (v2)	Pleasant vs. Unpleasant	0.48	0.44	0.64
	5	Native vs. Foreign Names (v2)	Pleasant vs. Unpleasant (v2)	0.67	-0.38	0.74
	6	Male vs. Female Names	Career vs. Family	0.61	-0.56	0.79
Gender	7	Math vs. Arts	Male vs. Female Terms	0.4	0.73	0.14
	8	Science vs. Arts	Male vs. Female Terms	-0.24	0.22	-0.28

Table 6: WEAT Test effect sizes for **pretrained** German BERT, T5, and GPT-2. Positive scores indicate that Target 1 (i.e. Mental Disease) is more associated with Attribute 1 (i.e. Temporary) than Target 2 (i.e. Physical Disease). An equivalent statement is that Target 2 (i.e. Physical Disease) is more associated with Attribute 2 (i.e. Permanent) than Target 1 (i.e. Mental Disease). Scores scale between +1.0 and -1.0.

Bias	#	Targets	Attributes	German BERT	German T5	German GPT-2
	1	Flowers vs. Insects	Pleasant vs. Unpleasant	0.23	0.36	0.07
Conceptual	2	Instruments vs. Weapons	Pleasant vs. Unpleasant	0.07	0.05	0.11
	9	Mental vs. Physical Disease	Temporary vs. Permanent	-0.37	0.17	0.6
	3	Native vs. Foreign Names	Pleasant vs. Unpleasant	0.85	0.52	0.62
Racial	4	Native vs. Foreign Names (v2)	Pleasant vs. Unpleasant	0.89	0.31	0.62
	5	Native vs. Foreign Names (v2)	Pleasant vs. Unpleasant (v2)	0.9	-0.29	0.64
	6	Male vs. Female Names	Career vs. Family	0.44	-0.46	0.61
Gender	7	Math vs. Arts	Male vs. Female Terms	0.54	0.51	0.01
	8	Science vs. Arts	Male vs. Female Terms	-0.1	0.08	-0.29

Table 7: WEAT Test effect sizes for **finetuned** German BERT, T5, and GPT-2, in comparison with pretrained results in Table 6. Positive scores indicate that Target 1 (i.e. Mental Disease) is more associated with Attribute 1 (i.e. Temporary) than Target 2 (i.e. Physical Disease). Green text indicates a positive effect size change due to finetuning, red text indicates a negative change.

Dynamic Relevance Graph Network for Knowledge-Aware Question Answering

Chen Zheng
Michigan State University
zhengc12@msu.edu

Parisa Kordjamshidi
Michigan State University
kordjams@msu.edu

Abstract

This work investigates the challenge of learning and reasoning for Commonsense Question Answering given an external source of knowledge in the form of a knowledge graph (KG). We propose a novel graph neural network architecture, called Dynamic Relevance Graph Network (DRGN). DRGN operates on a given KG subgraph based on the question and answers entities and uses the relevance scores between the nodes to establish new edges dynamically for learning node representations in the graph network. This explicit usage of relevance as graph edges has the following advantages, a) the model can exploit the existing relationships, re-scale the node weights, and influence the way the neighborhood nodes' representations are aggregated in the KG subgraph, b) It potentially recovers the missing edges in KG that are needed for reasoning. Moreover, as a byproduct, our model improves handling the negative questions due to considering the relevance between the question node and the graph entities. Our proposed approach shows competitive performance on two QA benchmarks, CommonsenseQA and OpenbookQA, compared to the state-of-the-art published results.

1 Introduction

Solving Question Answering (QA) problems usually requires both language understanding and human commonsense knowledge. Large-scale pre-trained language models (LMs) have achieved success in many QA benchmarks (Rajpurkar et al., 2016, 2018; Min et al., 2019; Yang et al., 2018). However, LMs have difficulties in predicting the answer when reasoning over external knowledge is required (Yasunaga et al., 2021; Feng et al., 2020).

Therefore, using the external sources of knowledge explicitly in the form of knowledge graphs (KGs) is a recent trend in question answering models (Lin et al., 2019; Feng et al., 2020). Figure 1, taken from the CommonsenseQA benchmark, shows an example for which answering the question requires commonsense reasoning. In this example, the external KG provides the required background information to obtain the reasoning chain from question to answer. We highlight two issues in the previous approaches taken to solve this QA problem: a) the extracted KG subgraph sometimes misses some edges between entities, which breaks the chain of reasoning that the current models can not exploit the connections, b) the semantic context of the question and connection to the answer is not used properly, for example, reasoning when negative terms exist in the question, such as no and not, is problematic.

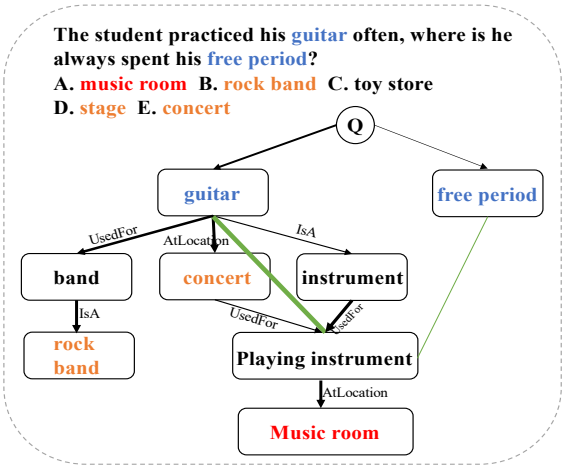

Figure 1: An example of the CommonsenseQA benchmark. Given the question node Q, question entity nodes (blue boxes), correct answer entity node (red box), and wrong answer entity nodes (orange boxes), we predict the answer by reasoning over the question and the extracted KG subgraph.

The first above-mentioned issue is caused by the following reasons. First, the knowledge graph is originally imperfect and does not include the required edges. Second, when constructing the subgraph, to reduce the graph size, most of the models select the entities that appear in two-hop

Proceedings of the 29th International Conference on Computational Linguistics, pages 1357–1366
October 12–17, 2022.

paths (Feng et al., 2020). Therefore, some intermediate concept (entity) nodes and edges are missed in the extracted KG subgraph. In such cases, the subgraph does not contain a complete chain of reasoning. Third, the current models often cannot reason over paths when there is no direct connection between the involved concepts. While finding the chain of reasoning in QA is challenging in general (Zheng and Kordjamshidi, 2021), here this problem is more critical when the KG is the only source and there are missing edges. Looking back at Figure 1, the KG subgraph misses the direct connection between *guitar* and *playing instrument* (green arrow). For the issue of considering question semantics, as (Lin et al., 2019) points out, previous models are not sensitive to the negation words and consequently predict opposite answers. QA-GNN (Yasunaga et al., 2021) model is the first work to deal with the negative questions. QA-GNN improves the reasoning under negation, to some extent, by adding the QA global node to the graph. However, the challenge still exists.

To solve the above challenges, we propose a novel architecture, called Dynamic Relevance Graph Network (DRGN). The motivation of our proposed DRGN is to recover the missing edges and establish direct connections between concepts to facilitate multi-hop reasoning. In particular, DRGN model uses a relational graph network module while influencing the importance of the neighbor nodes using an additional relevance matrix. It potentially can recover the missing edges to establish a direct connection based on the relevancy of the node representations in the KG during the training. The module can potentially capture the connections between distant nodes while benefiting from the existing KG edges. Our proposed model learns representations directly based on the relevance scores between subgraph entity pairs that are computed by Inner Product operation. At each convolutional layer of the graph neural network, we compute the inner product of the nodes based on their current layer's node representations dynamically and build the neighborhoods based on this relevance measure and form a relevance matrix accordingly. This can be seen as a way to learn new edges as the training goes forward in each layer while influencing on the weights of the neighbors dynamically based on their relevance. As shown in Figure 1, the relevance score between *guitar* and *playing instrument* is stronger than other nodes in the subgraph. Moreover, since the graph includes the question node, the relevance between the question node and entity nodes is computed at every layer, making use of the contextual information more effectively. It becomes more clear that the student will spend the *free period* in the *music room* rather than the *concert.*

In summary, the contributions of this work are as follows: **1)** The Proposed DRGN architecture exploits the existing edges in the KG subgraph while explicitly uses the relevance between the nodes to establish direct connections and recover the possibly missing edges dynamically. This technique helps in capturing the reasoning path in the KG for answering the question.
2) Our model exploits the relevance between question and the graph entities, which helps considering the semantics of the question explicitly in the graph reasoning and boosting the performance. In particular, it improves dealing with the negation.
3) Our proposed model obtains competitive results on both CommonsenseQA and OpenbookQA benchmarks. Our analysis demonstrates the significance and effectiveness of the DRGN model.

2 Related Work

2.1 QA using Knowledge Graph

Augmenting QA systems with external knowledge has been studied in many recent research papers. In this direction, pre-trained language models are often employed because they potentially can serve as implicit knowledge bases. (Devlin et al., 2019; Rajaby Faghihi and Kordjamshidi, 2021). To consider more interpretable knowledge, KGs are utilized in the QA models (Zheng and Kordjamshidi, 2022; Feng et al., 2020). However, given that the KGs are usually large and contain many nodes that are irrelevant to the question, the QA models can not effectively use the KG's information (Feng et al., 2020). Moreover, with larger KGs, the computational complexity of learning over them will increase. To deal with this issue, pruning KG nodes based on a variety of metrics has been proposed (Defferrard et al., 2016; Zhou et al., 2020; Velickovic et al., 2018; Hamilton et al., 2017; Ying et al., 2018).

Furthermore, the textual context is used as an additional node in the KG subgraph. For example, (Koncel-Kedziorski et al., 2019) and (Yasunaga et al., 2021) introduce the sentence node into the graph, while (Fang et al., 2020) and (Zheng and Kordjamshidi, 2020) add the paragraph node and

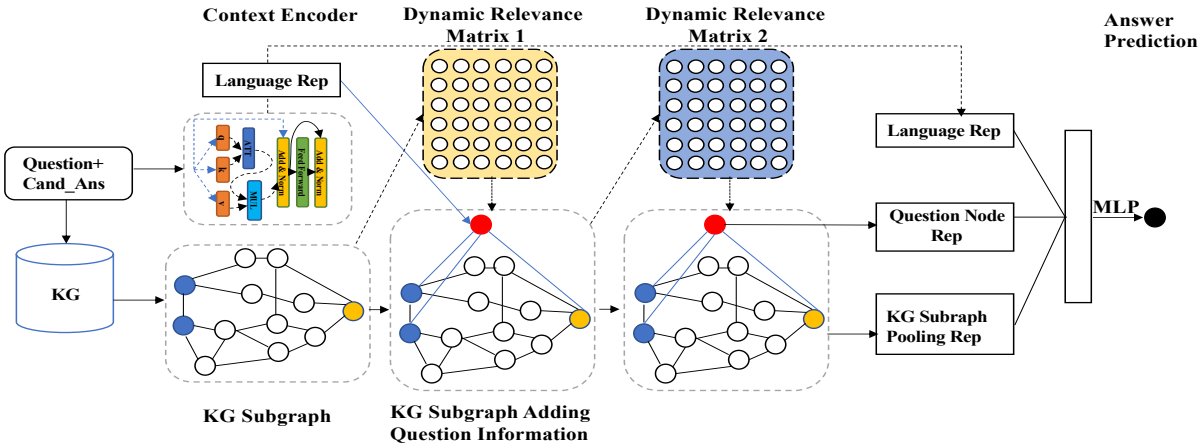

Figure 2: Our proposed DRGN model is composed of the Language Context Encoder module, KG Subgraph Construction module, Graph Neural Network module, and Answer Prediction module. The blue color entity nodes represent the entity mentioned in the question. The yellow color node represents the answer node. The red color node is the question node. We use different colors to draw the dynamic relevance matrix 1 and 2 because the relevance matrix changes dynamically in each graph neural layer.

sentence node to construct a hierarchical graph structure. In this work, we also use the question node as the external node and add it to the KG subgraph. Therefore, the graph representations can learn more contextual information by computing the relevance between the question node and graph entity nodes.

2.2 Graph Neural Networks

Graph convolutional network (GCN) (Kipf and Welling, 2017) is a classic multi-layer graph neural network. The node representations in the graph are strongly related to their neighborhood nodes and edges. For each layer of GCN, the node representations capture the information of their neighborhood nodes and edges via message passing and graph convolutional operation. R-GCN is a variation of GCN that deals with the multi-relational graph structure (Schlichtkrull et al., 2018). (Li et al., 2018) proposes an Adaptive Graph Convolution Network (AGCN) to learn the underlying relations and learn the residual graph Laplacian to improve spectral graph performance. Meanwhile, some varients of GCN try to replace the graph Fourier transform. For example, Graph Wavelet Neural Network (GWNN) (Xu et al., 2019) applies the graph wavelet transform to the graph, and achieves better performance compared to the graph Fourier transform in some tasks.

Meanwhile, several models use the attention based Transformer operator on the graphs. For example, the graph attention network (GAT) (Velickovic et al., 2018) uses the self-attention method and multi-head attention strategy to learn the node representations that consider the neighbors of the nodes. Besides, the gated attention network (GaAN) (Zhang et al., 2018) uses self-attention to aggregate the different heads' information. GaAN utilizes the gate mechanism to replace the average operation that is commonly used in the GAT model.

Dynamic GCN (Ye et al., 2020) is another branch of the GCN family. The dynamic graphs are constructed for different input samples. Moreover, Dynamic GCN learns the dynamic graph structure by a context-encoding network, which takes the whole feature map from the convolution neural network as input and directly predicts the adjacency matrix. Unlike these works, our DRGN model maintains the graph structure statically, but computes the relevance edges dynamically and uses the relevance to weight the neighbors for learning node representations. Besides, our approach uses the existing relationships in the KG, recovers the missing edges and establishes the direct connections by computing the relevance between nodes dynamically. We consider this as learning new edges based on the relevancy of the nodes while the training goes forward in each layer.

3 Dynamic Relevance Graph Network

3.1 Task Formulation

The task of QA over pure knowledge is to choose a correct answer a_{ans} from a set of N candidate answers $\{a_1, a_2, ..., a_n\}$ given input question q and

an external knowledge graph (KG). In fact, the input to the problem is not the whole KG but a subgraph, $G_{sub} = (V, E)$, is selected based on previous research in (Feng et al., 2020) and (Yasunaga et al., 2021). The node set V represents entities in the knowledge subgraph, and the edge set E represents the edges between entities.

3.2 Model Description

Figure 2 shows the proposed Dynamic Relevance Graph Network (DRGN) architecture. Our DRGN includes four modules: Language Context Encoder module, KG Subgraph Construction module, Graph Neural Network module, and Answer Prediction module. In this section, we describe the details of our approach and the way we train our model efficiently.

3.3 Language Context Encoder

For every question q and candidate answer a_i pair, we concatenate them to form Language Context L:

$$L = [[CLS]; q; [SEP]; a_i],$$

where [CLS] and [SEP] are the special tokens used by large-scale pre-trained Language Models (LMs). We feed input L to a pre-trained LMs encoder to obtain the list of token representations $h_L \in \mathbb{R}^{|L|*d}$, where $|L|$ represents the length of the sequence. Then we use the [CLS] representation, denoted as $h_{[CLS]} \in \mathbb{R}^d$, as the representation of L.

3.4 KG Subgraph Construction

We use ConceptNet (Speer et al., 2017), a general-domain knowledge graph, as the commonsense KG. ConceptNet graph has multiple semantic relational edges, e.g., HasProperty, IsA, AtLocation, etc. We follow (Feng et al., 2020) work to construct the subgraphs from KG for each example. The approach is to construct a subgraph from KG that contains the entities mentioned in the question and answer choices. The entities are selected with the exact match between n-gram tokens and ConceptNet concepts using some normalization rules. Then another set of entities is added to the subgraph by following the KG paths of two hops of reasoning based on the current entities in the subgraph.

Furthermore, we add the semantic context of the question as a separate node to the subgraph. This node provides an additional question context to the KG subgraph, G_{sub}, as suggested by (Yasunaga et al., 2021). We link the question node to entity nodes mentioned in the question. The semantic context of the question node Q is initialized by the [CLS] representation described in Section 3.3. The initial representation of the other entities is derived from applying RoBERTa and pooling over their contained tokens (Feng et al., 2020).

3.5 Graph Neural Network Module

The basis of our learning representation is Multi-relational Graph Convolutional Network (R-GCN) (Schlichtkrull et al., 2018). R-GCN is an extension of GCN that operates on a graph with multi-relational edges between nodes. In our case, the relation types between entities are taken from the 17 semantic relations from ConceptNet. Meanwhile, an additional type is added to represent the relationship between the question node and question entities, making the graph structure different from previous works. We denote the set of relations as R.

Our dynamic relevance graph network (DRGN) architecture is the variation of the R-GCN model. To establish the direct connection between the graph nodes and re-scale the importance of the neighbors, we compute the relevance score between the nodes dynamically at each graph layer based on their current learned representations. Then we build the neighborhoods based on this relevance measure and form a relevance matrix, M_{rel}, accordingly. This can be seen as a way to learn new edges based on the relevance of the nodes as the training goes forward in each graph layer. We use inner product to compute the relevance matrix:

$$M_{rel}^{(l)} = h^{(l)\top} h^{(l)} \in \mathbb{R}^{(|V|+1)*(|V|+1)},$$

where $|V|$ is the graph entity nodes sizes, and 1 is added due to using the question node in the graph. The relevance matrix re-scales the weights and influences the way the neighborhood nodes' representations are aggregated in the R-GCN model. M_{rel} is computed dynamically, and the relevance scores change while the representations are computed at each graph layer. In our proposed relational graph, the forward-pass message passing updates of the nodes, denoted by h_i, are calculated as follows:

$$h_i^{(l+1)} = \sigma\Big(\sum_{r \in R} \sum_{j \in \mathbb{N}_i^r} \frac{1}{d_{i,r}} W_r^{(l)} \cdot (M_{rel_{i,j}}^{(l)} h_j^{(l)}) + W_0^{(l)} \cdot (M_{rel_{i,i}}^{(l)} h_i^{(l)})\Big) \in \mathbb{R}^d,$$

where $\mathbb{N}_i^r$ represents the neighbor nodes of node i under relation r, $r \in R$. σ is the activation function,

W_r denotes the learnable parameters. Besides, we calculate the updated question node representation as follows,

$$h_Q^{(l+1)} = \sigma(\sum_{j \in \mathbb{N}^Q} W_Q^{(l)} \cdot F_c([h_Q^{(l)}; (M_{rel_{Q,j}}^{(l)} h_j^{(l)})]) + W_0^{(l)} \cdot (M_{rel_{Q,Q}}^{(l)} h_Q^{(l)})) \in \mathbb{R}^d,$$

where F_c is a two-layer MLP, h_Q is the question node representation. Finally, we stack the node representations to form $h'^{(l+1)}$:

$$h'^{(l+1)} = [h_0^{(l+1)}; h_1^{(l+1)}; \cdots; h_{|V|}^{(l+1)}; h_Q^{(l+1)}] \in \mathbb{R}^{(|V|+1)*d}.$$

We then compute the $(l+1)$ layer's dynamic relevance matrix $M_{rel}^{(l+1)}$ that shows the relevance scores of node representations. Finally, we use the $M_{rel}^{(l+1)}$ to multiply the node representation matrix $h'^{(l+1)}$ that helps the node representation to learn the weights of the edges based on the learned relevance and specifically to include the additional relevance edges between the nodes during the massage passing as follows:

$$h^{(l+1)} = \sigma\left(M_{rel}^{(l+1)} \cdot h'^{(l+1)} \cdot W_g\right) \in \mathbb{R}^{(|V|+1)*d},$$

where W_g is the learnable parameters.

3.6 Answer Prediction

Given the Language Context L and KG subgraph, we use the information from both the language representation $h_{[CLS]}$, question node representation h_Q learnt from the KG subgraph, and the KG subgraph representation pooled from the last graph layer, $pool(h_{G_{sub}})$, to calculate the scores of the candidate answers as follows:

$$p(a|L, G_{sub}) = f_{out}([h_{[CLS]}; h_Q; pool(h_{G_{sub}})]),$$

where f_{out} is a two-layer MLP. Finally, we choose the highest scored answer from N candidate answers as the prediction output. We use the cross entropy loss to optimize the end-to-end model.

4 Experiments and Results

4.1 Datasets

We evaluate our model on two different QA benchmarks, CommonsenseQA and OpenbookQA. Both benchmarks come with an external knowledge graph. We apply ConceptNet to the external knowledge graph on these two benchmarks.

CommonsenseQA (Talmor et al., 2019) is a QA dataset that requires human commonsense reasoning capacity to answer the questions. Each question in CommonsenseQA has five candidate answers without any extra information. The dataset consists of 12,102 questions.

OpenbookQA (Mihaylov et al., 2018) It is a multiple-choice QA dataset that requires reasoning with commonsense knowledge. The OpenbookQA benchmark is a well-defined subset of science QA (Clark et al., 2018) that requires finding the chain of commonsense reasoning to answer a question. Each data sample includes the question, scientific facts, and candidate answers. In our experimental setting, the scientific facts are added to the question part. This makes the problem formulation consistent with the CommonsenseQA setting.

4.2 Implementation Details

We implemented our DRGN architecture using PyTorch.[1] We use the pre-trained RoBERTa-large (Liu et al., 2019) to encode the question. We use cross-entropy loss and RAdam optimizer (Liu et al., 2020) to train our end-to-end architecture. The batch size is set to 16, and the maximum text input sequence length set to 128. Our model uses an early stopping strategy during the training. We use a 3-layer graph neural module in our experiments. Section 5.3 describes the effect of the different number of layers. The learning rate for the LMs is $1e-5$, while the learning rate for the graph module is $1e-3$.

4.3 Baseline Description

KagNET (Lin et al., 2019) is a path-based model that models the multi-hop relations by extracting relational paths from Knowledge Graph and then encoding paths with an LSTM sequence model.

MHGRN (Feng et al., 2020): Multi-hop Graph Relation Network (MHGRN) is a strong baseline. MHGRN model applies LMs to the question and answer context encoder, uses GNN encoder to learn graph representations, and chooses the candidate answer by these two encoders.

QA-GNN (Yasunaga et al., 2021) is the recent SOTA model that uses a working graph to train language and KG subgraph. The model jointly reasons over the question and KG and jointly updates the representations. QA-GNN uses GAT as the

[1] Our code is available at https://github.com/HLR/DRGN.

Models	Dev ACC%	Test ACC%
RoBERTa-no KG	69.6%	67.8%
R-GCN	72.6%	68.4%
GconAttn	72.6%	68.5%
KagNet	73.3%	69.2%
RN	73.6%	69.5%
MHGRN	74.4%	71.1%
QA-GNN	76.5%	73.4%
DRGN	**78.2%**	**74.0%**

Table 1: Dev accuracy and Test accuracy (In-House split) of various models on the CommonsenseQA benchmark, following by (Lin et al., 2019).

Models	Dev	Test
RoBERTa-large	66.7%	64.8%
R-GCN	65.0%	62.4%
GconAttn	64.5%	61.9%
RN	66.8%	65.2%
MHGRN	68.1 %	66.8%
QA-GNN	68.9 %	67.8%
DRGN	**70.1%**	**69.6%**
AristoRoBERTaV7	79.2%	77.8%
T5(3 Billion Parameters)	-	83.2%
UnifiedQA(11 Billion Parameters)	-	**87.2%**
AristoRoBERTaV7+MHGRN	78.6%	80.6%
AristoRoBERTaV7+QA-GNN	80.4%	82.8%
AristoRoBERTaV7+DRGN	**81.8%**	**84.1%**

Table 2: Development and Test accuracy of various model performance on the OpenbookQA benchmark.

backbone to do message passing on the graph. To learn the semantic edge information, QA-GNN directly adds the edge representation to the local node representation and cannot learn the global structure of the edges, which is inefficient. However, our model uses the global multi-relational adjacency matrices to learn the edge information.

4.4 Result Comparison

Table 1 shows the performance of different models on the CommonsenseQA benchmark. KagNet and MHGRN are two strong baselines. Our model outperforms the KagNet by 4.8% and MHGRN by 2.9% on CommonsenseQA benchmark. This result shows the effectiveness of our DRGN architecture. Table 2 shows the performance on the OpenbookQA benchmark. There are a few recent papers that exploit larger LMs, such as T5 (Raffel et al., 2020) that contains 3 billion parameters (10x larger than our model,) and UnifiedQA (Khashabi et al., 2020) (32x larger). For a fair comparison, we use the same RoBERTa setting for the input representation when we evaluate OpenbookQA. Our model performance, potentially, will be improved after using these larger LMs. To demonstrate this point, we did additional experiments using AristoRoBERTaV7 (Clark et al., 2019) as a backbone to train our model. Our model achieves better performance when using the larger LMs compared to other baseline models. The performance shows that the more implicit information learned from pre-trained language models, the more effective relevance information established between graph nodes. We should note that GREASELM (Zhang et al., 2022) and GSC (Wang et al., 2022) are two most recent models that are developed in parallel with our DRGN. GREASELM aims to ground language context in commonsense knowledge graph by fusing token representations from pretrained LMs and GNN over *Modality Interaction* layers (Zhang et al., 2022). GSC designs a *Graph Soft Counter* layer (Wang et al., 2022) to enhance the graph reasoning capacity. Our results are competitive with the reported ones in those parallel works while each work emphasizes different contributions.

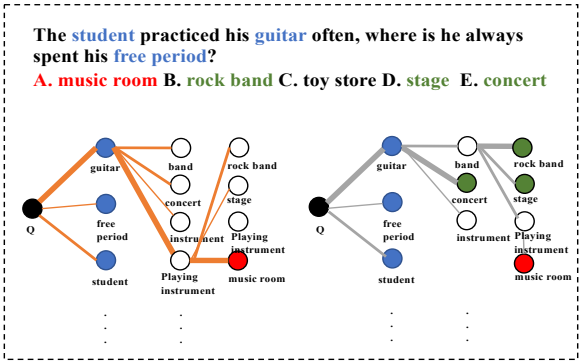

Figure 3: The complete reasoning chain from the question node to the candidate answer node. The blue nodes are question entity nodes, the red and green nodes are the candidate answer nodes. The thicker edges indicate the higher relevance score to the neighborhood node, while the thinner edges indicate the lower score. The left side is the reasoning chain selected from our model (orange edges), while the right side is selected from the baseline models (grey edges).

5 Analysis

5.1 Effects on Finding the Line of Reasoning

In this section, we analyze the effectiveness of our DRGN model that helps in recovering the missing edges and establishing direct connections based on the relevancy of the node representations in the KG.

Models	Test ACC % (Overall)	Test ACC% question w/ negative
RoBERTa-large	68.7 %	54.2%
KagNet	69.2 %	54.2 %
MHGRN	71.1 %	54.8%
QA-GNN	73.4 %	58.8%
DRGN	**75.0%**	**60.1%**

Table 3: Performance on questions with negation in In-house split test CommonsenseQA.

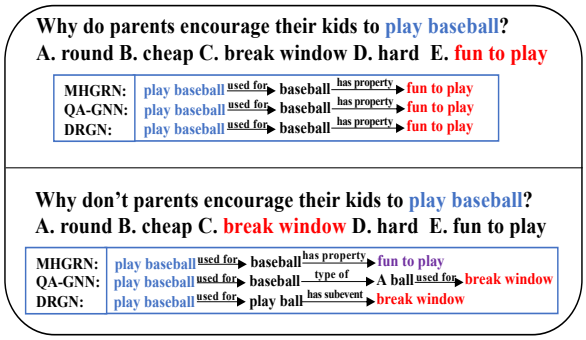

Figure 4: The case study of the negation examples. The question in the bottom box includes the negation words. The red colored text represents the gold answer, and the purple colored represents the wrong answer. In the blue box, each line represents the commonsense reasoning chain of each model.

As we described in Section 3.4, to keep the graph size small, most of the models construct the KG subgraph via selecting the entities that appear in two-hop paths. Therefore, some intermediate concept nodes and edges are missed in the extracted KG subgraph, and the complete reasoning chain from the question entity node to the candidate answer node can not be found.

For example, as shown in Figure 3, the question is "The student practiced his guitar often, where is he always spent his free period?" and the answer is "music room". The reasoning chain includes 2 hops, that is, "guitar → playing instrument → music room". Since the constructed graph misses the direct edge between "guitar" and "playing instrument", MHGRN and QA-GNN baselines select the wrong intermediate node and predict the wrong answer "concert" and "rock band" by the grey edges described in the Figure 3. In contrast, our DRGN model makes a correct prediction by computing the relevance score of the nodes based on their learned representations and forming new edges accordingly. As we describe in Section 3.4, our model initializes the entity node representation by large-scale pre-trained language models (LMs). The implicit representations of LMs are learned from the huge corpora, and the knowledge is implicitly learned. Therefore, these two entities, "guitar" and "playing instrument", start with an implicit connection. By looking at the relevance changes, after several layers of graph encoding, the relevance score between "guitar" and "playing instrument" becomes stronger. In contrast, the relevance score between "guitar" and "concert" becomes weaker because of the contextual information "free period". This is the primary reason why our DRGN model obtains the correct reasoning chain.

5.2 Effects on Semantic Context

While the graph has a broad coverage of knowledge, the semantic context of the question and connection to the answer is not used properly. For example, dealing with negation can not perform well (Yasunaga et al., 2021). Since our dynamic relevance matrix includes the semantic context of the question, the relevance between the question and graph entities is computed at every graph neural layer while considering the negation in the node representations. Intuitively, this should improve handling the negative question in our model.

To analyze this hypothesis for DRGN architecture, we compare the performance of various models on questions containing negative words (e.g., no, not, nothing, unlikely) from CommonsenseQA following (Yasunaga et al., 2021). The result is shown in Table 3. We observe that the baseline models of KagNet and MHGRN provide limited improvements over RoBERTa on questions with negation words (+0.4%). However, our DRGN model exhibits a huge boost (+5.9%). Moreover, the DRGN model gains a larger improvement in the accuracy compared to the QA-GNN model, demonstrating the effectiveness of considering relevance between question semantics and graph entity that experimentally confirms our hypothesis. An additional ablation study in Table 5 confirms this idea further. When removing the question information from DRGN, we observe that the performance on negation becomes close to the MHGRN.

Figure 4 shows qualitative examples about the positive and negative questions. For the positive question, all the models obtain the same reasoning chain "play baseball-(used for)→ baseball-(has property)→ fun to play", including MHGRN, QA-GNN, and our architecture. However, when adding the negative words, MHGRN obtains the same rea-

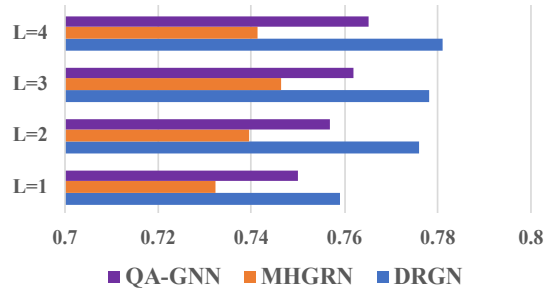

Figure 5: The Effect of number of layers in QA-GNN, MHGRN, and DRGN models on CommonsenseQA.

Models	Time	Space
l-layer KagNet	$O(\|R\|^l\|V\|^{l+1}l)$	$O(\|R\|^l\|V\|^{l+1}l)$
l-layer MHGRN	$O(\|R\|^2\|V\|^2l)$	$O(\|R\|\|V\|l)$
l-layer QA-GNN	$O(\|V\|^2l)$	$O(\|R\|\|V\|l)$
l-layer DRGN	$O(\|R\|^2\|V\|^2l)$	$O(\|R\|\|V\|^2l)$

Table 4: The time complexity and space complexity comparison between DRGN and baseline models.

soning chain as the positive situation, while QA-GNN and DRGN find the correct reasoning chain. One interesting finding is that DRGN can detect the direct connection using fewer hops to establish the reasoning chain.

5.3 Effects of Number of Graph Layers

The number of graph layers is an influencing factor for DRGN architecture because our relevance matrix is computed dynamically, and the relevance scores change while the representations are computed at each graph layer. We evaluate the effects of multiple layers l for the baseline models and our DRGN by evaluating its performance on the CommonsenseQA. As shown in Figure 5, the increase of l continues to bring benefits until $l = 4$ for DRGN. We compare the performance after adding each layer for MHGRN, QA-GNN, and our DRGN. We observe that DRGN consistently achieves the best performance with the same number of layers as the baselines.

Table 4 shows the time complexity and the space complexity comparison between DRGN model and baseline model. We compare the computational complexity based on the number of layers l, the number of nodes V, and the number of relations R. Our model and MHGRN have the same time complexity because both models use the R-GCN model as the backbone. Besides, QA-GNN directly adds the edge representation to the local node representation during the graph pre-processing step and learns the graph node representation without

Models	Dev ACC
DRGN w/o KG subgraph	69.6%
+ KG subgraph	72.6%
+ relational edges in graph	73.7%
+ question node in graph	74.9%
+ dynamic relevance matrix	78.2%

Table 5: Ablation Study on CommonsenseQA dataset.

the global semantic relational adjacency matrices. After adding the dynamic relevance matrix at each graph layer, our DRGN model achieves better performance compared to other baseline architectures. For the space complexity, our model's space complexity is slightly larger than MHGRN because DRGN introduces the extra dynamic relevance matrix. However, this cost depends on the size of the subgraph, which is usually small while it leads to a huge improvement.

5.4 Ablation of DRGN Modules

To evaluate the effectiveness of various components of DRGN, we perform an ablation study on the CommonsenseQA development benchmark. Table 5 shows the results of ablation study. First, we remove the whole commonsense subgraph. Our model without the subgraph obtains 69.6% on the CommonsenseQA. This shows how the implicit language model can answer the questions without the external KG, which is not high-performing but yet impressive. After adding the KG subgraph, the accuracy improves to 72.6% on the CommonsenseQA benchmark. Second, we keep the KG subgraph and add multiple relational edge information from the subgraph (described in section 3.5). Without the relational edges, the accuracy becomes 73.7%. This result shows that the multiple relational edges help in learning better graph node representations and obtaining a higher performance. Third, we keep the multi-relational subgraph and add the question node. In other words, we incorporate the semantic relationship between the question node and the graph entities. The accuracy of the model improves to 74.9%. Finally, we add the most important component, the dynamic relevance matrix, to each graph layer. The large improvement demonstrates the importance of the dynamic relevance matrix and the effectiveness of DRGN architecture.

6 Conclusion

In this paper, we propose a novel Dynamic Relevance Graph Network (DRGN) architecture for commonsense question answering given an external source of knowledge in the form of a Knowledge Graph. Our model learns the graph node representation while a) exploits the existing relations in KG, b) re-scales the importance of the neighbor nodes in the graph based on training a dynamic relevance matrix, c) establishes direct connections between graph nodes based on measuring the relevance scores of the nodes dynamically during training. The dynamic relevance edges help in finding the chain of reasoning when there are missing edges in the original KG. Our quantitative and qualitative analysis shows that the proposed approach facilitates answering the complex questions that need multiple hops of reasoning. Furthermore, since DRGN uses the relevance between the question node and graph entities, it exploits the richer semantic context of the question in graph reasoning which leads to improvements in the performance on the negative questions. Our proposed approach shows competitive performance on two QA benchmarks, including CommonsenseQA and OpenbookQA.

Acknowledgments

We thank all reviewers for their suggestions and helpful comments. This project is supported by National Science Foundation (NSF) CAREER award #2028626 and partially supported by the Office of Naval Research (ONR) grant #N00014-20-1-2005. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of the National Science Foundation nor the Office of Naval Research.

References

Peter Clark, Isaac Cowhey, Oren Etzioni, Tushar Khot, Ashish Sabharwal, Carissa Schoenick, and Oyvind Tafjord. 2018. Think you have solved question answering? try arc, the ai2 reasoning challenge. *arXiv preprint arXiv:1803.05457*.

Peter Clark, Oren Etzioni, Daniel Khashabi, Tushar Khot, Bhavana Dalvi Mishra, Kyle Richardson, Ashish Sabharwal, Carissa Schoenick, Oyvind Tafjord, Niket Tandon, et al. 2019. From'f'to'a'on the ny regents science exams: An overview of the aristo project. *arXiv preprint arXiv:1909.01958*.

M. Defferrard, X. Bresson, and P. Vandergheynst. 2016. Convolutional neural networks on graphs with fast localized spectral filtering. In *NIPS*.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, Minneapolis, Minnesota. Association for Computational Linguistics.

Yuwei Fang, S. Sun, Zhe Gan, Rohit Radhakrishna Pillai, Shuohang Wang, and Jingjing Liu. 2020. Hierarchical graph network for multi-hop question answering. In *EMNLP*.

Yanlin Feng, Xinyue Chen, Bill Yuchen Lin, Peifeng Wang, Jun Yan, and Xiang Ren. 2020. Scalable multi-hop relational reasoning for knowledge-aware question answering. In *EMNLP*.

William L. Hamilton, Zhitao Ying, and J. Leskovec. 2017. Inductive representation learning on large graphs. In *NIPS*.

D. Khashabi, S. Min, T. Khot, A. Sabhwaral, O. Tafjord, P. Clark, and H. Hajishirzi. 2020. Unifiedqa: Crossing format boundaries with a single qa system. *EMNLP - findings*.

Thomas N. Kipf and Max Welling. 2017. Semi-supervised classification with graph convolutional networks. In *International Conference on Learning Representations (ICLR)*.

Rik Koncel-Kedziorski, Dhanush Bekal, Yi Luan, Mirella Lapata, and Hannaneh Hajishirzi. 2019. Text generation from knowledge graphs with graph transformers. In *NAACL*.

Ruoyu Li, S. Wang, Feiyun Zhu, and J. Huang. 2018. Adaptive graph convolutional neural networks. In *AAAI*.

Bill Yuchen Lin, Xinyue Chen, Jamin Chen, and Xiang Ren. 2019. Kagnet: Knowledge-aware graph networks for commonsense reasoning. In *NAACL*.

Liyuan Liu, Haoming Jiang, Pengcheng He, Weizhu Chen, Xiaodong Liu, Jianfeng Gao, and Jiawei Han. 2020. On the variance of the adaptive learning rate and beyond. In *ICLR*.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Roberta: A robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692*.

Todor Mihaylov, Peter Clark, Tushar Khot, and Ashish Sabharwal. 2018. Can a suit of armor conduct electricity? a new dataset for open book question answering. In *Proceedings of the 2018 Conference on*

Empirical Methods in Natural Language Processing, pages 2381–2391, Brussels, Belgium. Association for Computational Linguistics.

Sewon Min, Victor Zhong, Luke Zettlemoyer, and Hannaneh Hajishirzi. 2019. Multi-hop reading comprehension through question decomposition and rescoring. In *ACL*.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J. Liu. 2020. Exploring the limits of transfer learning with a unified text-to-text transformer. *Journal of Machine Learning Research*, 21(140):1–67.

Hossein Rajaby Faghihi and Parisa Kordjamshidi. 2021. Time-stamped language model: Teaching language models to understand the flow of events. In *The 2021 Annual Conference of the North American Chapter of the Association for Computational Linguistics (NAACL-2021)*.

Pranav Rajpurkar, Robin Jia, and Percy Liang. 2018. Know what you don't know: Unanswerable questions for squad. In *ACL*.

Pranav Rajpurkar, Jian Zhang, Konstantin Lopyrev, and Percy Liang. 2016. Squad: 100, 000+ questions for machine comprehension of text. In *EMNLP*.

Michael Schlichtkrull, Thomas N Kipf, Peter Bloem, Rianne Van Den Berg, Ivan Titov, and Max Welling. 2018. Modeling relational data with graph convolutional networks. In *European semantic web conference*, pages 593–607. Springer.

Robyn Speer, Joshua Chin, and Catherine Havasi. 2017. Conceptnet 5.5: An open multilingual graph of general knowledge. In *Thirty-first AAAI conference on artificial intelligence*.

Alon Talmor, Jonathan Herzig, Nicholas Lourie, and Jonathan Berant. 2019. CommonsenseQA: A question answering challenge targeting commonsense knowledge. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4149–4158, Minneapolis, Minnesota. Association for Computational Linguistics.

Petar Velickovic, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Liò, and Yoshua Bengio. 2018. Graph attention networks. In *ICLR*.

Kuan Wang, Yuyu Zhang, Diyi Yang, Le Song, and Tao Qin. 2022. Gnn is a counter? revisiting gnn for question answering. In *ICLR*.

Bingbing Xu, Huawei Shen, Qi Cao, Yunqi Qiu, and Xueqi Cheng. 2019. Graph wavelet neural network. In *ICLR*.

Zhilin Yang, Peng Qi, Saizheng Zhang, Yoshua Bengio, William W. Cohen, Ruslan Salakhutdinov, and Christopher D. Manning. 2018. Hotpotqa: A dataset for diverse, explainable multi-hop question answering. In *EMNLP*.

Michihiro Yasunaga, Hongyu Ren, Antoine Bosselut, Percy Liang, and Jure Leskovec. 2021. Qa-gnn: Reasoning with language models and knowledge graphs for question answering. In *NAACL*.

Fanfan Ye, Shiliang Pu, Qiaoyong Zhong, Chao Li, Di Xie, and Huiming Tang. 2020. Dynamic gcn: Context-enriched topology learning for skeleton-based action recognition. In *Proceedings of the 28th ACM International Conference on Multimedia*, pages 55–63.

Rex Ying, Ruining He, K. Chen, Pong Eksombatchai, William L. Hamilton, and J. Leskovec. 2018. Graph convolutional neural networks for web-scale recommender systems. *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*.

Jiani Zhang, Xingjian Shi, Junyuan Xie, Hao Ma, Irwin King, and D. Yeung. 2018. Gaan: Gated attention networks for learning on large and spatiotemporal graphs. In *UAI*.

Xikun Zhang, Antoine Bosselut, Michihiro Yasunaga, Hongyu Ren, Percy Liang, Christopher D. Manning, and Jure Leskovec. 2022. Greaselm: Graph reasoning enhanced language models. In *ICLR*.

Chen Zheng and Parisa Kordjamshidi. 2020. SRLGRN: Semantic role labeling graph reasoning network. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 8881–8891, Online. Association for Computational Linguistics.

Chen Zheng and Parisa Kordjamshidi. 2021. Relational gating for "what if" reasoning. In *Proceedings of the Thirtieth International Joint Conference on Artificial Intelligence, IJCAI-21*, pages 4015–4022. International Joint Conferences on Artificial Intelligence Organization. Main Track.

Chen Zheng and Parisa Kordjamshidi. 2022. Relevant CommonSense subgraphs for "what if..." procedural reasoning. In *Findings of the Association for Computational Linguistics: ACL 2022*, pages 1927–1933, Dublin, Ireland. Association for Computational Linguistics.

Jie Zhou, Ganqu Cui, Zhengyan Zhang, Cheng Yang, Zhiyuan Liu, and M. Sun. 2020. Graph neural networks: A review of methods and applications. *AI Open*, 1:57–81.

SISER: Semantic-Infused Selective Graph Reasoning for Fact Verification

Eunhwan Park[1♠], **Jong-Hyeon Lee**[2♠*], **Donghyeon Jeon**[3], **Seonhoon Kim**[3], **Inho Kang**[3], **Seung-Hoon Na**[1†]
[1]Jeonbuk National University, [2]NCSOFT, [3]NAVER Corporation
{judepark, nash}@jbnu.ac.kr, leejh1230@ncsoft.com
{donghyeon.jeon, seonhoon.kim, once.ihkang}@navercorp.com

Abstract

This study proposes **S**emantic-**I**nfused **SE**lective Graph **R**easoning (SISER) for fact verification, which newly presents semantic-level graph reasoning and injects its reasoning-enhanced representation into other types of graph-based and sequence-based reasoning methods. SISER combines three reasoning types: 1) *semantic*-level graph reasoning, which uses a semantic graph from evidence sentences, whose nodes are elements of a triple – <Subject, Verb, Object>, 2) "semantic-infused" *sentence*-level "selective" graph reasoning, which combine semantic-level and sentence-level representations and perform graph reasoning in a selective manner using the node selection mechanism, and 3) *sequence* reasoning, which concatenates all evidence sentences and performs attention-based reasoning. Experiment results on a large-scale dataset for Fact Extraction and VERification (FEVER) show that SISER outperforms the previous graph-based approaches and achieves state-of-the-art performance.

1 Introduction

An ever-increasing number of unconfirmed false or misleading information spread on various social media platforms has motivated the verification of textual information, referred to as *fact verification*. FEVER (Thorne et al., 2018a) presented a large dataset for fact verification, initiating a shared task that aims to automatically classify a human-generated claim into *'Supported'*, *'Refuted'*, or *'Not Enough Info'* based on retrieved evidence sentences from Wikipedia[1].

Claim verification, the final step of fact verification, is viewed as a task of natural language inference (NLI) (Angeli and Manning, 2014). Specifically, the NLI task for claim verification is formulated as the *set-to-sentence* entailment of inferring whether a claim (as the hypothesis) is logically "entailed" from a set of retrieved evidence sentences (as the premise).

Recently, *graph reasoning* for claim verification has been extensively explored (Zhou et al., 2019; Liu et al., 2020; Zhong et al., 2020), which creates a graph whose nodes are semantic units extracted from a set of evidence sentences or a claim, and applies graph neural networks (GNNs) such as (Veličković et al., 2018; Kipf and Welling, 2017) to infer the entailment relationship. However, graph reasoning may be somehow restricted to *unit-biased reasoning*, when relying on a single type of semantic unit for nodes of a graph, such as sentences, entities, or words, meaning that the semantic interaction between claim and evidence is restricted to a single graph type and does not go beyond the coverage of the "given" semantic units. In addition, graph reasoning may suffer from *over-smoothing* inherited from GNNs (Gasteiger et al., 2019; Zhao and Akoglu, 2020; Chen et al., 2020a; Rong et al., 2020), likely causing all node representations to converge to a stationary point at the extreme, as reported by (Li et al., 2018).

To address these limitations of graph reasoning, this study proposes **SISER** – **S**emantic-**I**nfused **SE**lective graph **R**easoning) for fact verification by extensively exploiting additional semantic units for graph reasoning and integrating semantic-level reasoning with sequence reasoning and "selective" graph reasoning. SISER combines the following three types of reasoning:

- ***Semantic*-level graph reasoning** applies GNNs to a "semantic graph" whose nodes are elements of <Subject, Verb, Object> that appear in evidence sentences. Provided fine-grained semantic granularity, it is expected that the use of semantic elements would be helpful to effectively induce their own dis-

[†]Corresponding author
[♠]Equal contribution
[*]This work was done during his M.S. degree at JBNU.
[1]https://competitions.codalab.org/competitions/18814

Proceedings of the 29th International Conference on Computational Linguistics, pages 1367–1378
October 12–17, 2022.

tinct representations useful for claim verification, compared to sentence-level representations.

- **Semantic-infused *sentence*-level selective graph reasoning** combines semantic- and sentence-level representations and performs selective graph reasoning equipped with a node selection mechanism. Motivated by variants of GNNs (Gasteiger et al., 2019; Zhao and Akoglu, 2020; Chen et al., 2020a; Rong et al., 2020) to handle oversmoothing issues, we further provide "selective" graph reasoning where a subset of nodes is "selected" using the *node selection mechanism* and only these selected nodes participate in graph reasoning[2]. It is expected that the node selection mechanism can alleviate oversmoothing by breaking full connectivity.
- ***Sequence* reasoning**, concatenates a claim and all evidence sentences and performs self-attention over the concatenated long sequence. As in (Kruengkrai et al., 2021), it is expected that sequence reasoning shows stable performance, without suffering from the inherent problems of GNNs.

Furthermore, we newly apply *prompt-based fine-tuning* (Schick and Schütze, 2021a; Gao et al., 2021) by reformulating the fact verification task as a masked language modeling problem, where a *label word* is generated on a given prompt with a task-specific *template*. To the best of our knowledge, this is the first attempt to use semantic-level 'selective' graph reasoning and prompt-based fine-tuning for the fact verification task.

Our contributions are summarized as follows: 1) We propose SISER, which consists primarily of semantic-level reasoning and semantic-infused selective graph reasoning using the node selection mechanism for fact verification; 2) We present the initial work of adopting prompt-based fine-tuning for claim verification; 3) The proposed SISER shows state-of-the-art performance in the FEVER dataset.

2 Related Work

2.1 Fact Verification Systems

Sequence Reasoning

The baseline system (Thorne et al., 2018a) concatenates all retrieved evidence sentences and then feeds the concatenated evidence and a claim into a pretrained language model as an early sequence reasoning method. The studies of (Hanselowski et al., 2018; Hidey and Diab, 2018) proposed adapting the enhanced sequential inference model (ESIM) (Chen et al., 2017) to measure the semantic relatedness between a claim and evidence. Nie et al. (2019) proposed a carefully designed neural semantic matching network (NSMN), which is a modification of the enhanced sequential inference model. Unlike treating the fact verification task as an NLI task, LOREN (Chen et al., 2020b) proposed decomposing the verification of the entire claim at the phrase level, where the veracity of the phrases serves as explanations and can be aggregated into the final verdict according to logical rules. More recently, MLA (Kruengkrai et al., 2021) argued that graph reasoning may be unnecessary for a claim verification task, proposing *multi-level* sequence reasoning that consists of $\{\texttt{token}, \texttt{sentence}\}$-level self-attention (Vaswani et al., 2017).

Graph Reasoning

In contrast to ESIM, NSMN, and LOREN, GEAR (Zhou et al., 2019) proposed graph-based evidence reasoning using GNNs, which conducts reasoning and aggregation over claim-evidence pairs under an evidence graph (Veličković et al., 2018; Kipf and Welling, 2017). Similarly, KGAT (Liu et al., 2020) proposed the use of a semantic-level graph for fine-grained evidence reasoning that uses a kernel-based graph attention mechanism to properly propagate information between nodes. Unlike KGAT, DREAM (Zhong et al., 2020) considered a word span obtained by semantic role labeling (SRL) as a node in the graph and employed XLNet (Yang et al., 2019) as a pre-trained language model. In contrast to existing graph reasoning studies that rely on sentence-level or semantic-level graphs, SISER extensively uses "heterogeneous" graphs and fuses different types of reasoning-enhanced representations, going beyond the limitation of using only a single type of reasoning.

2.2 Prompt-based Fine-tuning

PET introduces prompt-based learning, which treats a downstream task as a masked language modeling problem and performs gradient-based fine-tuning (Schick and Schütze, 2021a,b). Employing prompt-based fine-tuning can reduce the gap between pre-training and fine-tuning, which

[2]Here, the selection process is random but parameterized by neural models.

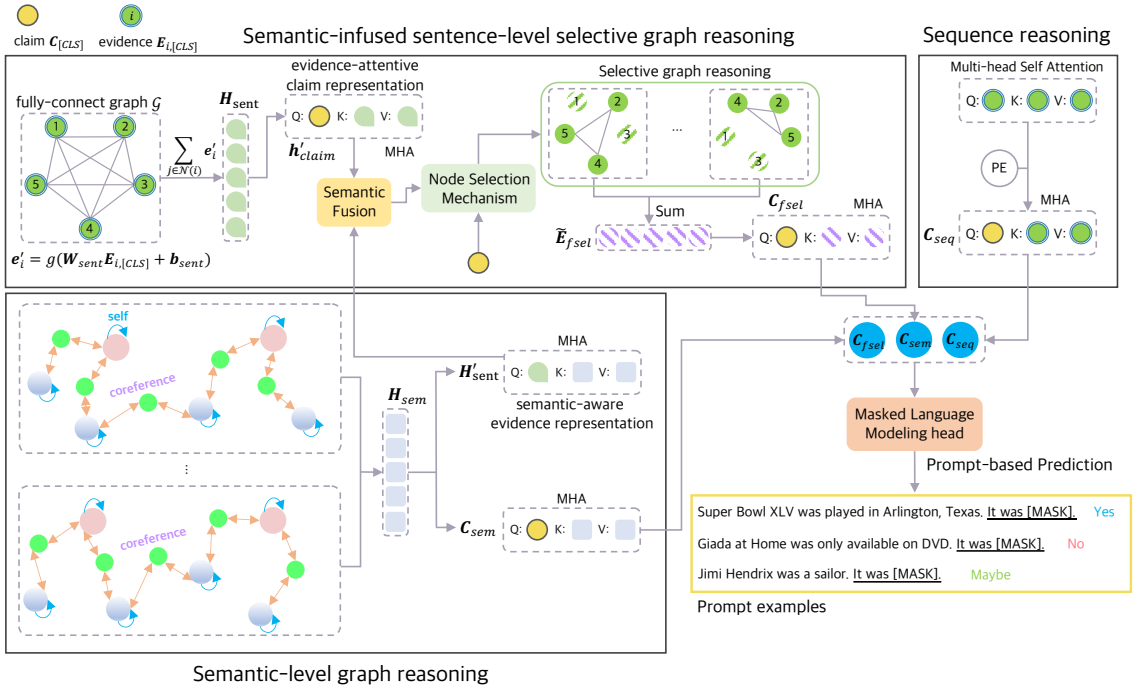

Figure 1: A neural architecture of the proposed SISER: 1) The semantic-level graph reasoning is performed using R-GCN on a semantic graph constructed using the Levi graph transformation to generate the semantic-level node representation $\boldsymbol{H}_{sem}$ (Eq. (2)), which is used to induce the *semantic-aware evidence representation* $\boldsymbol{H}'_{sent}$ (Eq. (5)). 2) The semantic-infused sentence-level selective graph reasoning performs the the selective graph reasoning on a sub-graph resulting from the *node selection mechanism* based on the semantic-fused representation of $\boldsymbol{h}'_{claim}$ (Eq. (5)) and $\boldsymbol{H}'_{sent}$ to generate $\tilde{\boldsymbol{E}}_{fsel}$ (Eq. (10)). 3) The sequence reasoning performs MHA on m evidence representations $\boldsymbol{E}_{seq}$ (Eq. (11)) to obtain $\boldsymbol{H}_{seq}$. 4) The prompt-based claim verification performs the prediction of label-verbalized words at [MASK]'s position on the fused semantic-attentive claim representations $\boldsymbol{H}$ induced from $\boldsymbol{C}_{fsel}, \boldsymbol{C}_{sem}, \boldsymbol{C}_{seq}$ as in Eq. (12).

makes it effective for various tasks. Inspired by PET, LM-BFF (Gao et al., 2021) introduced the adaptation of prompt-based learning to few-shot fine-tuning. Moreover, this study proposed an automatic prompt search method to resolve the difficulty of finding the optimal task-specific template. P^3 Ranker (Hu et al., 2022) proposed a pre-trained, prompt-learned, pre-finetuned neural ranker that employs prompt-based learning to convert the ranking task into pre-training and uses pre-finetuning. (Ding et al., 2021) introduced adapting prompt learning into an entity typing task in several scenarios (e.g., fully supervised, few-shot, zero-shot), which shows the possibility of employing prompt-based learning in fully supervised scenarios. Unlike several methods that employ prompt-based learning in a few-shot scenario, we adapt prompt-based learning in a fully supervised scenario.

3 Proposed Approach

Figure 1 shows the overall neural architecture of the proposed SISER model, which combines three types of reasoning: i.e., semantic-level graph reasoning; semantic-infused sentence-level selective graph reasoning; and sequence reasoning. This section presents details of the three reasoning methods.

3.1 Initial Representation of Claim and Evidences

Suppose that a claim c and a set of retrieved evidence sentences $\{\mathsf{e}_1, \cdots, \mathsf{e}_m\}$ are presented for a fact verification task, where m is the number of evidence sentences and PLM refers to the encoder of a pretrained language model such as BERT (Devlin et al., 2019) and RoBERTa (Liu et al., 2019). Feeding a claim-evidence pair $(\mathsf{c}, \mathsf{e}_i)$ for the i-th evidence sentence and claim c into PLM, we obtain $\boldsymbol{E}_i$ and $\boldsymbol{C}$ as evidence and claim representa-

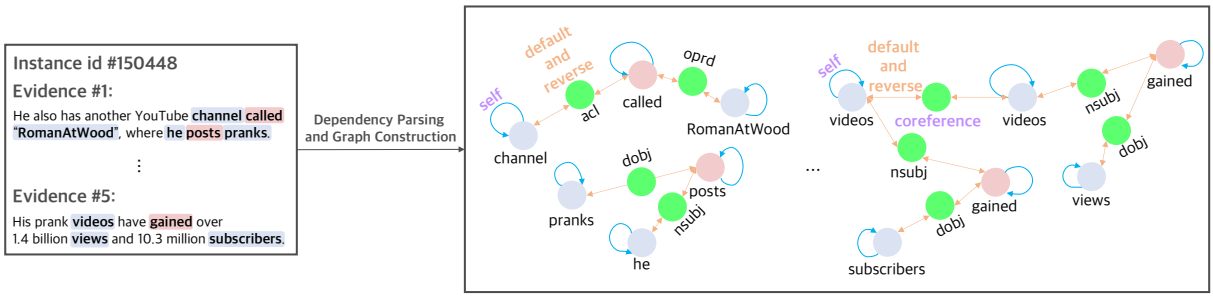

Figure 2: An illustration of constructing a semantic graph for sentence-level graph reasoning, motivated by the procedure of (Beck et al., 2018): 1) a (large) *dependency graph* is first obtained by applying the Spacy's syntactic parser (Honnibal and Montani, 2017) and the NeuralCoref's coreference resolution to m evidence sentences where each occurrence of a word is treated differently with its contextual representation. When two mentions are coreferent, their head words are connected by the "coreference" relation. 2) The dependency graph is then transformed to a *semantic graph* using the Levi graph transformation of (Beck et al., 2018) by including dependency labels as a node set with three types of edge labels – $\{\mathsf{default}, \mathsf{reverse}, \mathsf{self}\}$.

tions as follows:

$$\begin{aligned} \boldsymbol{E}_i &= \mathsf{PLM}\left(\mathsf{c}, \mathsf{e}_i\right) \in \mathbb{R}^{(|\mathsf{c}|+|\mathsf{e}_i|)\times d_{model}}, \\ \boldsymbol{C} &= \mathsf{PLM}\left(\mathsf{c}\right) \in \mathbb{R}^{|\mathsf{c}|\times d_{model}}, \end{aligned} \tag{1}$$

where $|\mathsf{x}|$ is the length of sequence x, and d_{model} is the dimensionality of PLM. Let $\boldsymbol{E}_{i,[\mathsf{CLS}]} \in \mathbb{R}^{d_{model}}$ and $\boldsymbol{C}_{[\mathsf{CLS}]} \in \mathbb{R}^{d_{model}}$ be representations of $[\mathsf{CLS}]$ tokens for e_i and c, respectively.

3.2 Semantic-level Graph Reasoning

Our semantic-level reasoning is similar to the work of (Zhong et al., 2020), but differs in using semantic units and types of GNNs, as described below.

3.2.1 Semantic Graph

Similar to (Beck et al., 2018), we construct a semantic graph based on graph transformation, starting from a dependency graph. More specifically, we first obtain a *dependency graph* $\mathcal{G}_{dep} = (\mathcal{V}_{dep}, \mathcal{E}_{dep})$, resulting from m by parsing all m evidence sentences using Spacy's syntactic parser (Honnibal and Montani, 2017)[3] and NeuralCoref's coreference resolution[4], where $\mathcal{V}_{dep}$ is a set of "words" that appear in m evidence sentences and $\mathcal{E}_{dep}$ is a set of dependency-labeled edges. When two mentions are connected by a coreference link, the "coreference" relation is appended between their head words. It should be noted that when a word occurs multiple times in m evidence sentences, we treat each occurrence differently by using their contextual representations (i.e., the span representations) as the elementary semantic representations.

We then convert $\mathcal{G}_{dep}$ into a *semantic graph* $\mathcal{G}_{sem} = (\mathcal{V}_{sem}, \mathcal{E}_{sem})$, a Levi Graph based on the graph transformation of (Beck et al., 2018; Cheng et al., 2020; Huang et al., 2021), where $\mathcal{V}_{sem}$ is a combined set of words and dependency relations that appear in m evidence sentences, and $\mathcal{E}_{sem}$ is a set of type-labeled edges whose labels are taken from $\mathcal{R} = \{\mathsf{default}, \mathsf{reverse}, \mathsf{self}\}$, as in the work of (Beck et al., 2018).

Figure 2 shows an illustrative example of a semantic graph extracted from the evidence sentences.

3.2.2 Graph Reasoning

Semantic-level graph reasoning employs a relational graph convolutional network (R-GCN) (Schlichtkrull et al., 2018) which is defined as

$$\boldsymbol{h}_i^{(l+1)} = f\left(\sum_{r\in\mathcal{R}}\sum_{j\in\mathcal{N}_{sem}^r(i)}\frac{1}{|\mathcal{N}_{sem}^r(i)|}\boldsymbol{W}_r^{(l)}\boldsymbol{h}_j^{(l)} + \boldsymbol{W}_0^{(l)}\boldsymbol{h}_i^{(l)}\right)$$

where f is the *relu* activation function, $\mathcal{N}_{sem}^r(i)$ is a set of neighbors with relation r of the i-th node in $\mathcal{V}_{sem}$, and $\boldsymbol{W}_r^{(l)}, \boldsymbol{W}_0^{(l)} \in \mathbb{R}^{d_{sem}\times d_{model}}$ are weight matrices for the l-th R-GCN layer, where d_{sem} is the dimensionality of the semantic-level representation. For a word-type node $i \in \mathcal{V}_{sem}$, $\boldsymbol{h}_i^{(0)} \in \mathbb{R}^{d_{model}}$ is initialized by its span representation in the evidence sentence[5]. Finally, we ob-

[3]We use the following link of the Spacy parser: https://spacy.io/usage/linguistic-features#dependency-parse

[4]The following version of the NeuralCoref's link is used: https://github.com/huggingface/neuralcoref

[5]The span representation for a word is defined as the average pooling of the contextual representations of its all subwords. For a relation-type node $i \in \mathcal{V}_{sem}$, $\boldsymbol{h}_i^{(0)}$ is initialized by its static embedding.

tain $\boldsymbol{H}_{sem} \in \mathbb{R}^{|\mathcal{V}_{sem}| \times d_{sem}}$ as follows:

$$\boldsymbol{H}_{sem} = \boldsymbol{H}^{(L)} = \left[\boldsymbol{h}_1^{(L)}, \cdots, \boldsymbol{h}_{|\mathcal{V}_{sem}|}^{(L)}\right]$$

where L is the total number of layers used in the R-GCN for the semantic-level representation.

3.3 Semantic-infused Sentence-level Selective Graph Reasoning

In our selective graph reasoning, because there is no ground-truth answer for the nodes to be selected, we prepare K different subgraphs by applying the node selection mechanism K times, and combine the selective representations performed over K subgraphs.

3.3.1 Semantic-infused Sentence-level Representations

The first step is to obtain *semantic-infused sentence-level representations* for m evidence sentences. To this end, we construct a fully-connected *sentence-level* graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ where $\mathcal{V} = \{1, \cdots, m\}$, which refers to a set of evidence sentences – $\{\mathsf{e}_1, \cdots, \mathsf{e}_m\}$. For the i-th node, we first obtain its node representation $\boldsymbol{e}'_i$ using a single feed-forward layer, as follows:

$$\boldsymbol{e}'_i = g\left(\boldsymbol{W}_{sent}\boldsymbol{E}_{i,[\mathsf{CLS}]} + \boldsymbol{b}_{sent}\right) \quad (2)$$

where g is the *gelu* activation function, and $\boldsymbol{W}_{sent}, \boldsymbol{b}_{sent}$ are the parameter weights for a linear layer. Then, for the i-th node, we further aggregate its neighbors' representations using the summation as follows:

$$\mathbf{h}'_i = \sum_{j \in \mathcal{N}_{sent}(i)} \boldsymbol{e}'_i \quad (3)$$

where $\mathcal{N}_{sent}(i)$ is a set of neighbors of the i-th node in $\mathcal{V}$.

Now, the sentence-level representation $\boldsymbol{H}_{sent} \in \mathbb{R}^{m \times d_{model}}$ is defined, as follows:

$$\boldsymbol{H}_{sent} = \left[\mathbf{h}'_1, \cdots, \mathbf{h}'_m\right] \quad (4)$$

Next, we obtain the *evidence-attentive claim representation* $\boldsymbol{h}'_{claim} \in \mathbb{R}^{d_{model}}$ and the *semantic-aware evidence representation* $\boldsymbol{H}'_{sent} \in \mathbb{R}^{m \times d_{model}}$ as follows:

$$\boldsymbol{h}'_{claim} = \mathsf{MHA}(\mathbf{C}_{[\mathsf{CLS}]}, \boldsymbol{H}_{sent}, \boldsymbol{H}_{sent}) \quad (5)$$

$$\boldsymbol{H}'_{sent} = \mathsf{MHA}(\mathbf{H}_{sent}, \boldsymbol{H}_{sem}, \boldsymbol{H}_{sem}) \quad (6)$$

where the multi-head attention (MHA) (Vaswani et al., 2017) function is defined as follows:

$$\mathsf{MHA}(\boldsymbol{Q}, \boldsymbol{K}, \boldsymbol{V}) = [head_1; \cdots; head_h]\boldsymbol{W}^O,$$
$$head_i = \mathsf{Attn}\left(\boldsymbol{Q}\boldsymbol{W}_i^Q, \boldsymbol{K}\boldsymbol{W}_i^K, \boldsymbol{V}\boldsymbol{W}_i^V\right) \quad (7)$$

where ; is the concatenation operator, h is the number of heads, $\boldsymbol{W}_i^Q, \boldsymbol{W}_i^K \in \mathbb{R}^{d_{model} \times d_k}$, $\boldsymbol{W}_i^V \in \mathbb{R}^{d_{model} \times d_v}$, and $\boldsymbol{W}^O \in \mathbb{R}^{hd_v \times d_{model}}$ are weight metrices.

To combine these representations, we use the *semantic fusion* function sfu defined as:

$$\mathsf{sfu}(\boldsymbol{x}, \boldsymbol{y}) = \boldsymbol{g} * \boldsymbol{x} + (1 - \boldsymbol{g}) * \boldsymbol{y},$$
$$\boldsymbol{g} = \sigma\left(\boldsymbol{W}_1\boldsymbol{x} + \boldsymbol{W}_2\boldsymbol{y}\right) \quad (8)$$

where $*$ is the element-wise operator, σ is the sigmoid function, and $\boldsymbol{W}_1, \boldsymbol{W}_2$ are weight matrices for the semantic fusion function.

Finally, the semantic-infused sentence-level representations $\boldsymbol{H}_{fused} \in \mathbb{R}^{m \times d_{model}}$ are then obtained using sfu as follows:

$$\boldsymbol{H}_{fused} = \mathsf{sfu}\left(\boldsymbol{H}'_{claim}, \boldsymbol{H}'_{sent}\right),$$

where $\boldsymbol{H}'_{claim} = [\boldsymbol{h}'_{claim}]_{i=1}^m$.

3.3.2 Node Selection Mechanism

The next step is to apply a node selection mechanism (Louis et al., 2021) that chooses a subset of nodes to be deleted[6]. First, we measure the selection probabilities $\boldsymbol{p}_{sent} \in \mathbb{R}^m$ of evidence nodes based on attention, using the claim as the query, as follows:

$$\boldsymbol{p}_{sent} = \sigma\left(g(\boldsymbol{H}_{sent}\boldsymbol{W}_3) + \boldsymbol{H}_{fused}\boldsymbol{W}_4\boldsymbol{C}_{[\mathsf{CLS}]}^T\right)$$

where $\boldsymbol{W}_3 \in \mathbb{R}^{d_{model} \times 1}, \boldsymbol{W}_4 \in \mathbb{R}^{d_{model} \times d_{model}}$ are weight matrices.

The node selection mechanism creates a subset of evidence nodes denoted as $\mathcal{V}'$ by filtering out the nodes with low probabilities given the threshold τ as follows:

$$\mathcal{V}' = \{j | j \in \mathcal{V} \text{ and } \boldsymbol{p}_{sent,j} \geq \tau\}$$

where $\boldsymbol{p}_{sent,j}$ is the j-th element of $\boldsymbol{p}_{sent}$. We further define $\boldsymbol{p}'_{sent} \in \mathbb{R}^m$ by zeroing the probabilities of the filtered nodes, as follows:

$$\boldsymbol{p}'_{sent} = \boldsymbol{p}_{sent} * \boldsymbol{i}_{\mathcal{V}'}$$

where $\boldsymbol{i}_{\mathcal{V}'} = [\mathcal{I}(k \in \mathcal{V}')]_{k=1}^m$ is the k-hot vector [7], and $\mathcal{I}(e)$ is the indicator function, taking the value of 1 if e is true and zero otherwise.

[6] Our node selection mechanism mostly follows the work of (Louis et al., 2021), but differs in the computation of node selection probabilities and the formula of selective aggregation.

[7] The k-hot vector has also similarly used in the work of (Cohen et al., 2019).

3.3.3 Selective Graph Reasoning

The final step is to perform selective graph reasoning using only the selected set of nodes, $\mathcal{V}'$. First, we obtain the revised fused representation $\boldsymbol{h}_i^{sel}$ for the i-th evidence sentence as follows:

$$\boldsymbol{h}_i^{sel} = \sum_{j \in N_{sent}(i)} \boldsymbol{p}'_{sent,j} \cdot \boldsymbol{H}_j^{fused}$$

Then, the reasoning-enhanced representation $\boldsymbol{h}_i^{fsel}$ is obtained as follows:

$$v_i = \sigma\left(\langle \boldsymbol{w}_{sel}, \left[\boldsymbol{h}_i^{sel}; \boldsymbol{e}'_i\right]\rangle\right),$$
$$\boldsymbol{h}_i^{fsel} = \sum_{j \in \mathcal{N}_{sent}(i)} \mathbf{p}'_{sent,j} \cdot v_j \cdot \boldsymbol{H}_j^{fused}$$

where $\boldsymbol{e}'_i$ is the initial node representation defined in Eq. (2) and $\boldsymbol{w}_{sel} \in \mathbb{R}^{2d_{model}}$ is the weight vector.

We further use the residual connection to keep the initial evidence representation as follows:

$$\tilde{\boldsymbol{e}}_i = g\big(\boldsymbol{e}'_i + \mathsf{dropout}(\boldsymbol{h}_i^{fsel})\big) \tag{9}$$

where dropout is the dropout layer introduced by (Srivastava et al., 2014).

3.3.4 Ensembling Multiple Selective Graph Reasonings

Because there is no ground-truth information for nodes to be selected, we prepare multiple subgraphs by applying the node selection mechanism K times, and combine the selective reasoning-enhanced representations over K subgraphs. With the abuse of notation, suppose that $\tilde{\boldsymbol{e}}_i^{(k)}$ is the reasoning-enhanced representation of Eq. (9) yielded at the k-th selection. We take the summation of all K representations as $\sum_{k=1}^{K} \tilde{\boldsymbol{e}}_i^{(k)}$, leading to obtain $\tilde{\boldsymbol{E}}_{fsel} \in \mathbb{R}^{m \times d_{model}}$ as follows:

$$\tilde{\boldsymbol{E}}_{fsel} = \left[\sum_{k=1}^{K} \tilde{\boldsymbol{e}}_i^{(k)}\right]_{i=1}^{m} \tag{10}$$

3.4 Sequence Reasoning

Our sequence reasoning is based on MHA over *only* sentence-level evidence representations $\boldsymbol{E}_{seq} \in \mathbb{R}^{m \times d_{model}}$, described as follows.

$$\boldsymbol{E}_{seq} = \mathsf{PE}(\boldsymbol{E}_{1,[\mathsf{CLS}]}, \cdots, \boldsymbol{E}_{m,[\mathsf{CLS}]}),$$
$$\boldsymbol{H}_{seq} = \boldsymbol{E}_{seq} + \mathsf{MHA}(\boldsymbol{E}_{seq}, \boldsymbol{E}_{seq}, \boldsymbol{E}_{seq}), \tag{11}$$

where PE is the absolute positional encoding (Vaswani et al., 2017).

Label	Training	Development	Test
Supported	80,035	6,666	6,666
Refuted	29,775	6,666	6,666
Not Enough Info	35,659	6,666	6,666

Table 1: Statistics of the FEVER 1.0 shared task dataset.

3.5 Prompt-based Claim Verification

Our *prompt-based claim verification* uses a task-specific template for prompt-based fine-tuning as follows: "[CLS] $\mathsf{x_{in}}$ It was [MASK] . [SEP]". Suppose that $\mathsf{x_{in}}$ is *"Roman Atwood is a content creator."*, $\mathsf{x_{in}}$ is converted to its prompted input "[CLS] Roman Atwood is a content creator. It was [MASK] . [SEP]". To predict [MASK], let $\mathcal{M}_{wo}\colon \mathcal{Y} \rightarrow \mathcal{V}$ be the verbalizer that converts a label into individual words. For example, $\mathcal{M}_{wo}(\mathsf{Supported})$ = "Yes", $\mathcal{M}_{wo}(\mathsf{Refutes})$ = "No", and $\mathcal{M}_{wo}(\mathsf{NotEnoughInfo})$ = "Maybe".

To determine the truthfulness of a given claim, we aggregate multiple evidence-attentive claim representations resulting from applying MHA on on $\tilde{\boldsymbol{E}}_{fsel}$ of Eq. (10) , $\boldsymbol{H}_{sem}$ in Eq. (2), and $\boldsymbol{H}_{seq}$ in Eq. (11), as follows:

$$\begin{aligned}
\boldsymbol{C}_{fsel} &= \mathsf{MHA}(\mathbf{C}_{[\mathsf{CLS}]}, \tilde{\boldsymbol{E}}_{fsel}, \tilde{\boldsymbol{E}}_{fsel}),\\
\boldsymbol{C}_{sem} &= \mathsf{MHA}(\mathbf{C}_{[\mathsf{CLS}]}, \boldsymbol{H}_{sem}, \boldsymbol{H}_{sem}),\\
\boldsymbol{C}_{seq} &= \mathsf{MHA}(\mathbf{C}_{[\mathsf{CLS}]}, \boldsymbol{H}_{seq}, \boldsymbol{H}_{seq}),\\
\mathbf{H} &= \boldsymbol{W}_{claim}([\boldsymbol{C}_{fsel}; \boldsymbol{C}_{sem}; \boldsymbol{C}_{seq}]),
\end{aligned} \tag{12}$$

where $W_{claim} \in \mathbb{R}^{d_{model} \times 3d_{model}}$ is a trainable parameter matrix.

Given a claim-evidence example (c, e), where $\mathsf{e} = (\mathsf{e}_1, \cdots, \mathsf{e}_m)$, the probability of label y is computed as follows:

$$\begin{aligned}
\mathsf{p}(\mathsf{y}|\mathsf{c},\mathsf{e}) &= \mathsf{p}\big([\mathsf{MASK}] = \mathcal{M}_{wo}(\mathsf{y})|\mathsf{c},\mathsf{e}\big)\\
&= \frac{\exp\big(\boldsymbol{w}_{\mathcal{M}_{wo}(\mathsf{y})}\boldsymbol{H}_{[\mathsf{MASK}]}\big)}{\sum_{\mathsf{y}' \in \mathcal{Y}} \exp\big(\boldsymbol{w}_{\mathcal{M}_{wo}(\mathsf{y}')}\boldsymbol{H}_{[\mathsf{MASK}]}\big)},
\end{aligned} \tag{13}$$

where $\boldsymbol{w}_{\mathcal{M}_{wo}}(\mathsf{y})$ is the output embedding for the label word of $\mathcal{M}_{wo}(\mathsf{y})$ for y, and $\boldsymbol{H}_{[\mathsf{MASK}]}$ is the contextual representation [MASK] token in $\boldsymbol{H}$.

4 Experiments

4.1 Experimental Setting

Dataset

We used FEVER, which is a large-scale public dataset, for fact verification. (Thorne et al.,

Model	Dev		Test	
	LA	F.S	LA	F.S
UNC NLP	69.72	66.49	68.21	64.21
GEAR ($\text{BERT}_{\text{base}}$)	74.84	70.69	71.60	67.10
DREAM ($\text{XLNet}_{\text{large}}$)	79.16	-	76.85	70.60
KGAT ($\text{BERT}_{\text{large}}$)	77.91	75.86	73.61	70.24
⌞ ($\text{RoBERTa}_{\text{large}}$)	78.29	76.11	74.07	70.38
LOREN ($\text{BERT}_{\text{large}}$)	78.44	76.21	74.43	70.71
⌞ ($\text{RoBERTa}_{\text{large}}$)	81.14	78.83	76.42	72.93
MLA ($\text{RoBERTa}_{\text{large}}$)	79.31	75.96	77.05	73.72
Ours ($\text{RoBERTa}_{\text{large}}$)	**83.13**	**79.87**	**77.50**	**73.90**

Table 2: Fact verification results on the dev and blind test set of FEVER task, where F.S (FEVER score) is the main evaluation metric. The best is **bolded** text, and the second best is underlined.

Model	Dev		Test	
	LA	F.S	LA	F.S
MLA	79.31	75.96	77.05	73.72
SISER⋆	83.13	79.85	76.82	73.18
SISER∘ ($\tau = 0.49$)	82.62	79.40	77.18	73.48
SISER ($\tau = 0.49$)	83.13	**79.87**	**77.50**	**73.90**

Table 3: Ablation study for the semantic-infused sentence-level selective graph reasoning and the sequence reasoning on FEVER development and blind test set. ⋆ and ∘ denote the run without the semantic-infused sentence-level selective graph reasoning and the sequence reasoning, respectively.

2018a,b), which was split into *training, development*, and *blind test* set in our experiments. FEVER consists of 185,455 annotated claims with 5,416,537 Wikipedia documents, where claims are classified as *Supported*, *Refuted*, or *Not Enough Info*. Because we use prompt-based fine-tuning, all labels are verbalized as *Yes*, *No*, or *Maybe*. Table 1 shows more detailed statistics for the FEVER dataset. The performance of the evidence sentence retrieval methods are presented in Appendix B.

Evaluation Metrics

The official evaluation metrics are Label Accuracy (LA) and FEVER Score (F.S)[8]. Label Accuracy is a general evaluation metric, which is the accuracy of the predicted label for a claim regardless of the retrieved evidence.

4.2 Main Results

The fact verification performance is presented in Table 2. In the large-size PLM settings, SISER outperforms the best baseline model by increasing Label Accuracy and FEVER Score by 0.45 and 0.18, respectively.

[8] https://github.com/sheffieldnlp/fever-scorer

Model	Dev		Test	
	LA	F.S	LA	F.S
$\tau = 0.0^{\bullet}$	83.07	79.84	77.07	73.65
$\tau = 0.35$	83.00	79.74	77.11	73.70
$\tau = 0.40$	83.05	79.84	77.00	73.63
$\tau = 0.45$	82.98	79.69	76.86	73.66
$\tau = 0.49$	**83.13**	**79.87**	**77.50**	**73.90**
$\tau = 0.60$	83.04	79.80	77.30	73.68

Table 4: Ablation study of the node selection mechanism for varying values of the node masking rate τ. $\bullet$ denotes the fully-connected setting.

Model	Dev		Test	
	LA	F.S	LA	F.S
SISER⋆	83.05	79.77	76.82	73.18
SISER	**83.13**	**79.87**	**77.50**	**73.90**

Table 5: Ablation study for the prompt-based learning vs. the conventional fine-tuning on the FEVER development set. ⋆ denotes the conventional fine-tuning.

For a fair comparison, we also compare SISER with KGAT and LOREN, which employ the same setting of using PLM and evidence retrieval, while MLA, the state-of-the-art baseline model, is different from ours in using evidence retrieval. As shown in Table 2, SISER outperforms KGAT and LOREN, which employ only sentence-level interaction among evidences. The results may support our motivation that the combination of the three types of reasoning (i.e., semantic-level graph-reasoning, semantic-infused sentence-level selective graph-reasoning, and sequence reasoning) is helpful to address the aforementioned 'unit-biased reasoning' and 'oversmoothing' problems of the existing graph-based approaches.

Model	Dev		Test	
	LA	F.S	LA	F.S
MLA	79.31	75.96	77.05	73.72
SISER⋆ ($\tau = 0.49$)	79.88	75.04	**77.96**	73.06
SISER ($\tau = 0.49$)	**83.13**	**79.87**	77.50	**73.90**

Table 6: Ablation study for examining the effect of evidence retrieval. ⋆ denotes the run based on the evidence retrieval of MLA (Kruengkrai et al., 2021).

Claim:	Liam Neeson has been nominated for a British Academy of Film and Television Arts award.
Evidence:	[Liam Neeson] (12-th sentence in wiki page)
(a)	He has been nominated for a number of awards, including an Academy Award for Best Actor, a BAFTA Award for Best Actor in a Leading Role and three Golden Globe Awards for Best Actor in a Motion Picture Drama.
Label:	SUPPORTS **Predicted Label:** NOT ENOUGH INFO

Claim:	LinkedIn is limited to 24 languages as of 2015.
Evidence:	[LinkedIn] (15-th sentence in wiki page)
(b)	Based in the United States, the site is, as of 2013, available in 24 languages, including Arabic, Chinese, English, French, German, Italian, Portuguese, Spanish, Dutch, Swedish, Danish, Romanian, Russian, Turkish, Japanese, Czech, Polish, Korean, Indonesian, Malay, and Tagalog.
Label:	SUPPORTS **Predicted Label:** REFUTES

Claim:	SZA is an American Neo Soul singer.
Evidence:	[SZA (singer)] (7-th sentence in wiki page)
(c)	SZA is a Neo Soul singer whose music is described as Alternative RB , with elements of soul , hip hop , minimalist RB , cloud rap , ethereal RB , witch house and chillwave.
	[SZA (singer)] (1-th sentence in wiki page)
	Solána Rowe (born November 8, 1990), better known by her stage name SZA, is an American singer songwriter.
Label:	SUPPORTS **Predicted Label:** SUPPORTS

Figure 3: Error analysis of SISER: (a) and (c): the cases of requiring more elaborated and mulit-hop reasoning; (b): the case of a human annotation error.

4.3 Ablation Study

The Effect of Using Semantic-infused Sentence-level Selective Graph Reasoning

To evaluate the effect of using semantic-infused sentence-level selective graph reasoning in Section 3.3, Table 3 shows the comparison results of SISER with and without semantic-infused sentence-level selective graph reasoning on the FEVER development and blind test sets. It is shown that the use of semantic-infused selective graph reasoning leads to improved performance in terms of both Label Accuracy and the FEVER Score.

It is remarkable that SISER⋆, even without using semantic-infused selective graph reasoning, outperforms MLA in the development set. While (Kruengkrai et al., 2021) argued that graph reasoning may not be necessary, given the improved performance of the MLA, our results indicate that this argument is still controversial, and suggest that graph reasoning has the potential to make further improvements and needs to be explored for fact verification while carefully avoiding the limitations of GNNs.

The Effect of Using Sequence Reasoning

Table 3 further presents the performance of SISER when sequence reasoning is excluded (referred to as SISER○), that is, without using $\boldsymbol{C}_{fsel}$ in Eq (12)). As shown in Table 3, SISER○ leads to improvements over LOREA, indicating that the performance achieved by SISER in Table 2 is not obtained simply by incorporating sequence reasoning but dominantly by equipping with the proposed manner of graph reasoning. In particular, SISER○ shows an increases in Label Accuracy by approximately 1.5 over LOREN on the development set, whereas SISER with sequence reasoning demonstrates only a slight increase of approximately 0.5 in Label Accuracy. A similar tendency is observed in the blind test set; SISER○ makes the increase of 0.76 in Label Accuracy over LOREN, which is larger than the increase of 0.32 obtained by SISER with sequence reasoning.

The Effect of Choosing Evidence Retrieval

In Table 2, while SISER shows consistent improvements over MLA on the development and test sets, a significant difference in performance gains is noticeable between the two sets. SISER achieves a large performance gain over MLA on the development set, increasing the Label Accuracy and FEVER Score by 3.82 and 3.91, respectively, while only a slight improvement on the blind test set is observed, exhibiting an increase of 0.45 in Label Accuracy and 0.18 in FEVER Score.

We believe that the main reason for this discrepancy between development and test sets results from the different evidence retrieval methods between SISER and MLA, i.e., while SISER and LOREN adopt KGAT's evidence retrieval, MLA uses its own evidence retrieval. In particular, the retrieval performances of the top 5 evidence sentences resulting from MLA and KGAT are substantially changed between the development and test sets, as shown in Table 7. In terms of Recall@5, the retrieval performances on the "development set" are largely different between KGAT and MLA (i.e., 94.57 for KGAT and 88.64 for MLA), whereas the retrieval performances on the "test set" of both methods are fairly similar (i.e., 87.47 for KGAT and 87.58 for MLA). Given this observation, the substantially improved performance of SISER over MLA on the development set (Table 2) may primarily originate from the large recall performance of the evidence retrieval of KGAT, and not from the proposed en-

hanced graph reasoning components.

For a fair comparison with MLA, Table 6 presents the results of SISER based on MLA's evidence retrieval (SISER⋆). In terms of on FEVER Score, SISER⋆ does not lead to improvements over MLA, even exhibiting performance degradation, in contrast to the SISER that uses KGAT's retrieval. Nevertheless, SISER⋆ leads to further improvements over MLA in Label Accuracy, particularly in achieving a state-of-the-art performance on the blind test set.

As MLA is considered as an advanced approach to sequence reasoning without relying on graph reasoning, we believe that the enhanced graph reasoning modules in SISER are 'complementary' to MLA for further improvement; for example, including a simple combination by using MLA as an alternative module of sequence reasoning in SISER.

Evaluation of Node Selection Mechanism

To examine the effect of the node selection mechanism in Section 3.3.2, Table 4 shows the comparison results of SISER with varying values of τ. It is shown that $\tau = 0.49$ outperforms the fully-connected setting ($\tau = 0.0$). The results imply that the node selection mechanism based on the selection probabilities may be helpful in obtaining irrelevance-free evidence representations, related to the oversmoothing issue of GNNs.

Prompt-based Learning versus Conventional Fine-tuning

To examine the effect of prompt-based claim verification, Table 5 compares the results of SISER when using prompt-based learning or conventional fine-tuning. It is clearly shown that the use of prompt-based learning outperforms conventional fine-tuning, likely reducing the gap between the tasks used in pre-training and the fine-tuning.

4.4 Case Study

As shown in Figure 3, we present three examples for analyzing the prediction errors of SISER.

In Figure 3 (a), the SISER prediction for this case is "*Not Enough Info*". From our analysis, this case requires the complex reasoning ability to understand "*a BAFTA award*," which is the abbreviation of "*a British Academy of Film and Television Arts award*". However, in Figure 3 (c), the case requires multi-hop complex reasoning to predict the claim; the claim "*SZA is an **American Neo Soul singer***" is supported by multiple pieces of evidence sentences.

In Figure 3 (b), it seems that this case originates from a human annotation error, as also discussed by (Kruengkrai et al., 2021). The claim "*LinkedIn is limited to 24 languages as of 2015*" is not supported by evidence.

5 Conclusion

In this paper, we propose SISER for fact verification, which combines three types of reasoning (i.e., semantic-level graph reasoning, semantic-infused sentence-level selective graph reasoning, and sequence reasoning) by addressing two potential limitations of graph reasoning — the "unit-biased reasoning" and the "over-smoothing" problems. The experimental results obtained using the FEVER dataset showed that the proposed SISER outperformed other graph-based approaches and achieved state-of-the-art performances in both the development and test sets.

In future work, we would like to incorporate semantic-level and semantic-fused graph reasoning into evidence retrieval and explore the joint learning framework of evidence retrieval and claim verification in a multi-task learning setting.

Acknowledgements

This work was supported by NAVER Corp. We would like to thank all anonymous reviewers for their valuable comments and suggestions.

References

Gabor Angeli and Christopher D. Manning. 2014. NaturalLI: Natural logic inference for common sense reasoning. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 534–545, Doha, Qatar. Association for Computational Linguistics.

Daniel Beck, Gholamreza Haffari, and Trevor Cohn. 2018. Graph-to-sequence learning using gated graph neural networks. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 273–283, Melbourne, Australia. Association for Computational Linguistics.

Deli Chen, Yankai Lin, Wei Li, Peng Li, Jie Zhou, and Xu Sun. 2020a. Measuring and relieving the over-smoothing problem for graph neural networks from the topological view. *Proceedings of the AAAI Conference on Artificial Intelligence*, 34(04):3438–3445.

Jiangjie Chen, Qiaoben Bao, Changzhi Sun, Xinbo Zhang, Jiaze Chen, Hao Zhou, Yanghua Xiao, and

Lei Li. 2020b. Loren: Logic-regularized reasoning for interpretable fact verification.

Qian Chen, Xiaodan Zhu, Zhen-Hua Ling, Si Wei, Hui Jiang, and Diana Inkpen. 2017. Enhanced LSTM for natural language inference. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1657–1668, Vancouver, Canada. Association for Computational Linguistics.

Liying Cheng, Dekun Wu, Lidong Bing, Yan Zhang, Zhanming Jie, Wei Lu, and Luo Si. 2020. ENT-DESC: Entity description generation by exploring knowledge graph. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1187–1197, Online. Association for Computational Linguistics.

William W. Cohen, Haitian Sun, R. Alex Hofer, and Matthew Siegler. 2019. Differentiable representations for multihop inference rules. *CoRR*, abs/1905.10417.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, Minneapolis, Minnesota. Association for Computational Linguistics.

Ning Ding, Yulin Chen, Xu Han, Guangwei Xu, Pengjun Xie, Hai-Tao Zheng, Zhiyuan Liu, Juanzi Li, and Hong-Gee Kim. 2021. Prompt-learning for fine-grained entity typing. *CoRR*, abs/2108.10604.

Matthias Fey and Jan E. Lenssen. 2019. Fast graph representation learning with PyTorch Geometric. In *ICLR Workshop on Representation Learning on Graphs and Manifolds*.

Tianyu Gao, Adam Fisch, and Danqi Chen. 2021. Making pre-trained language models better few-shot learners. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 3816–3830, Online. Association for Computational Linguistics.

Johannes Gasteiger, Aleksandar Bojchevski, and Stephan Günnemann. 2019. Predict then propagate: Graph neural networks meet personalized pagerank. In *International Conference on Learning Representations (ICLR)*.

Andreas Hanselowski, Hao Zhang, Zile Li, Daniil Sorokin, Benjamin Schiller, Claudia Schulz, and Iryna Gurevych. 2018. UKP-athene: Multi-sentence textual entailment for claim verification. In *Proceedings of the First Workshop on Fact Extraction and VERification (FEVER)*, pages 103–108, Brussels, Belgium. Association for Computational Linguistics.

Christopher Hidey and Mona Diab. 2018. Team SWEEPer: Joint sentence extraction and fact checking with pointer networks. In *Proceedings of the First Workshop on Fact Extraction and VERification (FEVER)*, pages 150–155, Brussels, Belgium. Association for Computational Linguistics.

Matthew Honnibal and Ines Montani. 2017. spacy 2: Natural language understanding with bloom embeddings, convolutional neural networks and incremental parsing. *To appear*, 7(1).

Xiaomeng Hu, Shi Yu, Chenyan Xiong, Zhenghao Liu, Zhiyuan Liu, and Ge Yu. 2022. P^3 ranker: Mitigating the gaps between pre-training and ranking fine-tuning with prompt-based learning and pre-finetuning. *CoRR*, abs/2205.01886.

Yinya Huang, Meng Fang, Yu Cao, Liwei Wang, and Xiaodan Liang. 2021. DAGN: Discourse-aware graph network for logical reasoning. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 5848–5855, Online. Association for Computational Linguistics.

Thomas N. Kipf and Max Welling. 2017. Semi-supervised classification with graph convolutional networks. In *5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Conference Track Proceedings*. OpenReview.net.

Canasai Kruengkrai, Junichi Yamagishi, and Xin Wang. 2021. A multi-level attention model for evidence-based fact checking. In *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*, pages 2447–2460, Online. Association for Computational Linguistics.

Qimai Li, Zhichao Han, and Xiao-Ming Wu. 2018. Deeper insights into graph convolutional networks for semi-supervised learning. AAAI Press.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Roberta: A robustly optimized BERT pretraining approach. *CoRR*, abs/1907.11692.

Zhenghao Liu, Chenyan Xiong, Maosong Sun, and Zhiyuan Liu. 2020. Fine-grained fact verification with kernel graph attention network. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7342–7351, Online. Association for Computational Linguistics.

Steph-Yves M. Louis, Alireza Nasiri, Fatima J. Rolland, Cameron Mitro, and Jianjun Hu. 2021. NODE-SELECT: A graph neural network based

on A selective propagation technique. *CoRR*, abs/2102.08588.

Yixin Nie, Haonan Chen, and Mohit Bansal. 2019. Combining fact extraction and verification with neural semantic matching networks. In *The Thirty-Third AAAI Conference on Artificial Intelligence, AAAI 2019, The Thirty-First Innovative Applications of Artificial Intelligence Conference, IAAI 2019, The Ninth AAAI Symposium on Educational Advances in Artificial Intelligence, EAAI 2019, Honolulu, Hawaii, USA, January 27 - February 1, 2019*, pages 6859–6866. AAAI Press.

Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Kopf, Edward Yang, Zachary DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. 2019. Pytorch: An imperative style, high-performance deep learning library. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett, editors, *Advances in Neural Information Processing Systems 32*, pages 8024–8035. Curran Associates, Inc.

Yu Rong, Wenbing Huang, Tingyang Xu, and Junzhou Huang. 2020. Dropedge: Towards deep graph convolutional networks on node classification. In *8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, April 26-30, 2020*.

Timo Schick and Hinrich Schütze. 2021a. Exploiting cloze-questions for few-shot text classification and natural language inference. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*, pages 255–269, Online. Association for Computational Linguistics.

Timo Schick and Hinrich Schütze. 2021b. It's not just size that matters: Small language models are also few-shot learners. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 2339–2352, Online. Association for Computational Linguistics.

Michael Sejr Schlichtkrull, Thomas N. Kipf, Peter Bloem, Rianne van den Berg, Ivan Titov, and Max Welling. 2018. Modeling relational data with graph convolutional networks. In *The Semantic Web - 15th International Conference, ESWC 2018, Heraklion, Crete, Greece, June 3-7, 2018, Proceedings*, volume 10843 of *Lecture Notes in Computer Science*, pages 593–607. Springer.

Noam Shazeer and Mitchell Stern. 2018. Adafactor: Adaptive learning rates with sublinear memory cost. In *Proceedings of the 35th International Conference on Machine Learning, ICML 2018, Stockholmsmässan, Stockholm, Sweden, July 10-15, 2018*, volume 80 of *Proceedings of Machine Learning Research*, pages 4603–4611. PMLR.

Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. 2014. Dropout: A simple way to prevent neural networks from overfitting. *Journal of Machine Learning Research*, 15(56):1929–1958.

James Thorne, Andreas Vlachos, Christos Christodoulopoulos, and Arpit Mittal. 2018a. FEVER: a large-scale dataset for fact extraction and VERification. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 809–819, New Orleans, Louisiana. Association for Computational Linguistics.

James Thorne, Andreas Vlachos, Oana Cocarascu, Christos Christodoulopoulos, and Arpit Mittal. 2018b. The fact extraction and VERification (FEVER) shared task. In *Proceedings of the First Workshop on Fact Extraction and VERification (FEVER)*, pages 1–9, Brussels, Belgium. Association for Computational Linguistics.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, December 4-9, 2017, Long Beach, CA, USA*, pages 5998–6008.

Petar Veličković, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Liò, and Yoshua Bengio. 2018. Graph Attention Networks. *International Conference on Learning Representations*. Accepted as poster.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Rémi Louf, Morgan Funtowicz, Joe Davison, Sam Shleifer, Patrick von Platen, Clara Ma, Yacine Jernite, Julien Plu, Canwen Xu, Teven Le Scao, Sylvain Gugger, Mariama Drame, Quentin Lhoest, and Alexander M. Rush. 2020. Transformers: State-of-the-art natural language processing. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 38–45, Online. Association for Computational Linguistics.

Zhilin Yang, Zihang Dai, Yiming Yang, Jaime G. Carbonell, Ruslan Salakhutdinov, and Quoc V. Le. 2019. Xlnet: Generalized autoregressive pretraining for language understanding. In *Advances in Neural Information Processing Systems 32: Annual Conference on Neural Information Processing Systems 2019, NeurIPS 2019, December 8-14, 2019, Vancouver, BC, Canada*, pages 5754–5764.

Lingxiao Zhao and Leman Akoglu. 2020. Pairnorm: Tackling oversmoothing in {gnn}s. In *International Conference on Learning Representations*.

Wanjun Zhong, Jingjing Xu, Duyu Tang, Zenan Xu, Nan Duan, Ming Zhou, Jiahai Wang, and Jian Yin. 2020. Reasoning over semantic-level graph for fact checking. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 6170–6180, Online. Association for Computational Linguistics.

Jie Zhou, Xu Han, Cheng Yang, Zhiyuan Liu, Lifeng Wang, Changcheng Li, and Maosong Sun. 2019. GEAR: Graph-based evidence aggregating and reasoning for fact verification. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 892–901, Florence, Italy. Association for Computational Linguistics.

A Implementation Details

SISER was implemented by using PyTorch (Paszke et al., 2019) and HuggingFace Transformers (Wolf et al., 2020). Additionally, the PyTorch-Geometric and SpaCy (Fey and Lenssen, 2019; Honnibal and Montani, 2017) were used for graph modeling and dependency parsing. Experiments were conducted using 4 Nvidia RTX A6000 GPU. All optimizations were performed using the Adafactor optimizer (Shazeer and Stern, 2018) with a linear warm-up of the learning rate. The warmup proportion was 0.06. The batch size and accumulation steps were 8 and 8, respectively. That is, the total batch size is 256. Gradients were clipped if their norms exceeded 1.0. The number of K sub-graphs was 6 and $\tau = 0.49$. In supervised learning, our loss $\mathcal{L}$ can be fine-tuned to minimize the weighted cross-entropy loss introduced by MLA (Kruengkrai et al., 2021).

Our hyperparameter is summarized as below:

- Optimizer: Adafactor
- Learning rate: $2e-5$
- warmup proportion: 0.06
- Number of sub-graph: 6
- Total Batch size: 256
- Gradient norm: 1.0
- Node masking rate τ: 0.49
- Label words: Supported : Yes, Refuted : No, Not Enough Info : Maybe

Data	Method	Prec@5	Recall@5	F1@5
Dev	UNC NLP*	36.49	86.79	51.38
	GEAR*	**40.60**	86.36	**55.23**
	KGAT$^\diamond$	27.29	**94.37**	42.34
	DREAM$^\diamond$	26.67	87.64	40.90
	MLA$^\diamond$	25.63	88.64	39.76
	monoT5$^\bullet$	25.66	90.54	37.17
Test	KGAT$^\diamond$	25.21	87.47	39.14
	MLA$^\diamond$	**25.33**	**87.58**	**39.29**

Table 7: Results of the sentence selection methods in the precision@5, recall@5, and F1@5 metrics on the FEVER development set and blind test set, respectively. $*, \diamond, \bullet$ denote ESIM-based retrieval model, BERT-based retrieval model, and T5-base model, respectively.

B Evidence Sentence Retrieval

Since our work focuses on claim verification, we directly adapt the evidence retrieval method from KGAT (Liu et al., 2020). As shown in Table 7, KGAT shows the best Recall@5 performance for sentence selection on the FEVER development set. Different from the result on the FEVER development set, MLA shows the better Recall@5 performance than KGAT.

Answering Numerical Reasoning Questions in Table-Text Hybrid Contents with Graph-based Encoder and Tree-based Decoder

Fangyu Lei[1,2], **Shizhu He**[1,2], **Xiang Li**[1,2], **Jun Zhao**[1,2], **Kang Liu**[1,2,3]

[1]National Laboratory of Pattern Recognition, Institute of Automation, CAS, Beijing, China
[2] School of Artificial Intelligence, University of Chinese Academy of Sciences, Beijing, China
[3] Beijing Academy of Artificial Intelligence, Beijing, 100084, China
{leifangyu2022, lixiang2022}@ia.ac.cn
{shizhu.he, jzhao, kliu}@nlpr.ia.ac.cn

Abstract

In the real-world question answering scenarios, hybrid form combining both tabular and textual contents has attracted more and more attention, among which numerical reasoning problem is one of the most typical and challenging problems. Existing methods usually adopt encoder-decoder framework to represent hybrid contents and generate answers. However, it can not capture the rich relationship among numerical value, table schema, and text information on the encoder side. The decoder uses a simple predefined operator classifier which is not flexible enough to handle numerical reasoning processes with diverse expressions. To address these problems, this paper proposes a **Re**lational **G**raph enhanced **H**ybrid table-text **N**umerical reasoning model with **T**ree decoder (**RegHNT**). It models the numerical question answering over table-text hybrid contents as an expression tree generation task. Moreover, we propose a novel relational graph modeling method, which models alignment between questions, tables, and paragraphs. We validated our model on the publicly available table-text hybrid QA benchmark (TAT-QA). The proposed RegHNT significantly outperform the baseline model and achieve state-of-the-art results[1] (2022-05-05).

1 Introduction

Question Answering (QA) is an important task of natural language processing (NLP), which is often used to assess the intelligence of an agent. QA systems use various types of knowledge to answer natural language questions. Earlier approaches independently utilized structured data such as tables (Pasupat and Liang, 2015; Yu et al., 2018), knowledge bases (Yih et al., 2016; Talmor and Berant, 2018) or unstructured data such as plain texts (Rajpurkar et al., 2016). In fact, real-world QA systems often need to fuse different data resources with diverse types in answering complex questions. Therefore, in recent years, the hybrid form of question answering over tables and texts (TextTableQA) has attracted more and more attention (Chen et al., 2020a,b, 2021).

There are two major question types for TextTableQA. The first is the fact reasoning question, whose answer is usually a span from the table or linked paragraphs, such as the contents in Wikipedia (Chen et al., 2020a,b). The second is the numerical reasoning question, which usually aims to use the contents of tables and texts for numerical calculation (Zhu et al., 2021; Chen et al., 2021). Most previous work focuses on the first type, while the numerical reasoning questions have been seldom addressed. The existing datasets such as WikiTableQuestions (Pasupat and Liang, 2015) and DROP (Dua et al., 2019) also contain numerical reasoning questions, but solving them requires only one type of data source. Therefore, this paper mainly focuses on answering numerical reasoning questions, especially for those complex questions across texts and tables.

To explore the application of numerical reasoning questions in hybrid contents. Zhu et al. (2021) proposed a hybrid text-table dataset TAT-QA, dedicated to fusing the tabular and textual contents to answer numerical reasoning questions. As shown in Figure 1, for the question "*What is the ratio of compensation expense related...?*", one needs to get the numerical value, i.e. "0.41", "278.29" from the table, and "109.7" from the text. Then we need to generate the corresponding numerical expression "109.7 / (0.41 × 278.29)". To solve such a numerical question, we need to identify the describing texts near the table and understand the contents of the table and the paragraphs.

Previous method (Zhu et al., 2021) regarded this problem as a sequence tagging task. They predefine aggregation operators and use a slot filling method to predict simple derivation. An auxiliary

[1]We openly released the source code and data at https://github.com/lfy79001/RegHNT

Proceedings of the 29th International Conference on Computational Linguistics, pages 1379–1390
October 12–17, 2022.

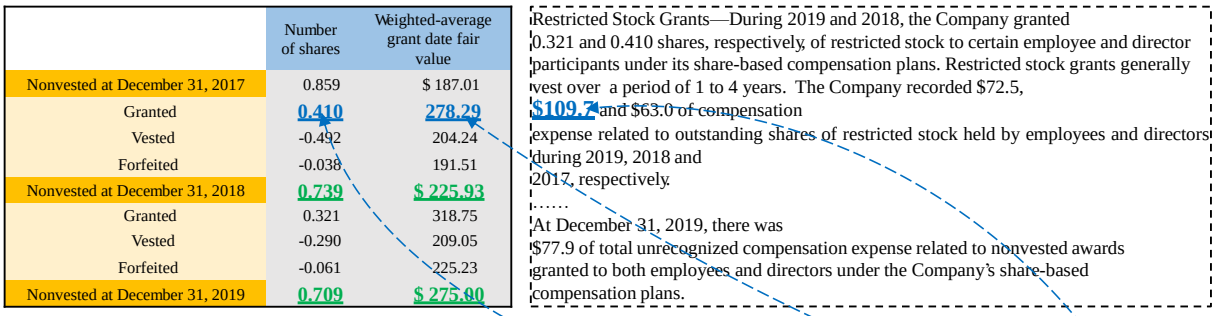

	Number of shares	Weighted-average grant date fair value
Nonvested at December 31, 2017	0.859	$ 187.01
Granted	0.410	278.29
Vested	-0.492	204.24
Forfeited	-0.038	191.51
Nonvested at December 31, 2018	0.739	$ 225.93
Granted	0.321	318.75
Vested	-0.290	209.05
Forfeited	-0.061	225.23
Nonvested at December 31, 2019	0.709	$ 275.00

Restricted Stock Grants—During 2019 and 2018, the Company granted 0.321 and 0.410 shares, respectively, of restricted stock to certain employee and director participants under its share-based compensation plans. Restricted stock grants generally vest over a period of 1 to 4 years. The Company recorded $72.5, $109.7 and $63.0 of compensation expense related to outstanding shares of restricted stock held by employees and directors during 2019, 2018 and 2017, respectively.

.....

At December 31, 2019, there was $77.9 of total unrecognized compensation expense related to nonvested awards granted to both employees and directors under the Company's share-based compensation plans.

Question Type	Question	Answer	Answer From	Derivation
Arithmetic	What is the ratio of compensation expense related to outstanding shares of restricted stock during 2018 to the total price of restricted stock shares granted between 2017 and 2018?	0.96	Table Text	109.7 / (0.41 * 278.29)
Arithmetic	What is the percentage change in the total price of nonvested shares from December 31, 2018, to 2019?	16.78	Table	((0.709 * 275.00) - (0.739 * 225.93)) / (0.739 * 225.93)

Figure 1: An example of TAT-QA. The solid boxes are tables, and the dotted boxes are the corresponding paragraphs. The bottom table shows two complex questions that cannot be solved by the previous method. The same color marks the source of the answer, while the blue dashed arrow points to the source of the answer. For the table, blue cells are $\mathrm{T}_{\mathrm{column}}$, yellow cells are $\mathrm{T}_{\mathrm{row}}$ and gold cells are $\mathrm{T}_{\mathrm{time}}$.

number order classifier is used for operators sensitive to the operation orders. Moreover, for the complex numerical computation problems in Figure 1, the model cannot predict the answer because of the absence of predefined operators. Thus, previous models based on predefined operators have a deficiency of low generalizability and flexibility. In addition, another problem of the method is to model tables and the texts solely and could not aggregate information from different data types. As a result, incorrect answers are usually generated because of the incomprehensive information from a single type of data.

To solve these problems, we propose a novel method to model such table-text hybrid data for the TextTableQA task, especially for the numerical reasoning question type. Specifically, we build a heterogeneous graph to capture the relationship between different data types. And different node types and relation types (intra-relation and inter-relation) are defined, as detailed in Appendix A. We expect the QA model to capture the correlation between the tables and the texts and aggregate them effectively. The model could focus on the whole contents rather than a single data type. Then a tree-based decoder is built. And we expect it could make good use of the different data structures from different data types and select appropriate nodes in the heterogeneous graph. After that, an expression tree and a prefix expression are generated. So the model can generate arbitrary forms of derivation without the need for predefined slots. It eliminates error propagation in the operator prediction module and improves the flexibility and generalizability of numerical reasoning.

Experimental results on the public benchmark TAT-QA demonstrate that our proposed model RegHNT improves EM values by 20.2% and F_1 values by 20.0% over the baseline. Our main contributions are summarized as follows:

- We design a novel graph construction method to model the information from table-text hybrid data, which effectively captures the correlation between tables and texts.
- We propose a tree structure decoder to solve the numerical reasoning problems. Based on our method, an expression tree and a prefix expression are generated. Our approach can cover arbitrary numerical derivation forms and improve the model's flexibility and generalizability. To our knowledge, this is the first tree-based model for TextTableQA.
- We think our graph-tree framework can be used as a strong baseline for the TextTable numerical reasoning task. Empirical results on the TAT-QA dataset demonstrate that the proposed model is effective, which achieves the state-of-the-art performances[2].

[2]Leaderboard of TAT-QA: https://nextplusplus.github.io/TAT-QA

2 Problem Definition

We represent a natural language question as $Q = (q_1, q_2, ..., q_{|Q|})$ with length $|Q|$. Each question is associated with a table T and a paragraph $P = (s_1, s_2, ..., s_{|P|})$ with the number of sentences $|P|$. The table T consists of several cells $T = \{c_1, c_2, ...\}$ and each table cell c_i can be further divided into K words $(c_{i1}, c_{i2}, ..., c_{iK})$. Similarly, each paragraph sentence s_i contains several words $(s_{i1}, s_{i2}, ..., s_{iL})$ with the sentence length L. Our goal is to generate the ground truth answer through numerical calculation (see Figure 1).

	Arithmetic Type
Answer	16.78
Expression	[(0.709*275.00)-(0.739*225.93)]/(0.739*225.93)
Polish Notation	/ - * 0.709 275.00 * 0.739 225.93 * 0.739 225.93
Tree	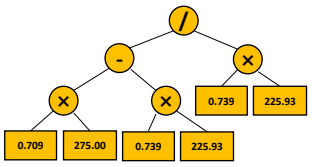

Figure 2: The expressions of arithmetic questions. We predict an mathematical expression consisting of numeric nodes and operators to get the answer.

The examples of expressions can be seen in Figure 2. For arithmetic questions which need numerical calculation, inspired by math word problem solving (Liu et al., 2019a; Wang et al., 2019a; Zhang et al., 2020), we generate the final mathematical expression $\mathrm{E_p}$, which is the polish notation transformed from the original math expression. $\mathrm{E_p}$ can always be represented as a solution expression tree $\mathrm{T_e}$ because the preorder traversal result of the tree is the polish notation.

3 Method

In this section, we present in detail the modules of **Re**lational **G**raph enhanced **H**ybird table-text **N**umerical reasoning model with **T**ree decoder (**RegHNT**) (see Figure 3). First, a text-table hybrid data modeling approach is proposed. After constructing the graph, we utilize a classic encoder-decoder architecture for predicting answers. It consists of a graph input module, a graph enhanced hidden module, and a tree-based decoder module. Both the graph input module and the graph enhanced hidden module are parts of the encoder, aiming to map the input heterogeneous graph G into node embeddings $\mathbf{Z} \in \mathbb{R}^{|V| \times d}$, where d is the graph hidden size. The tree-based decoder module is responsible for transforming $\mathbf{Z}$ into the target $\mathrm{T_e}$.

3.1 Graph Construction

The entire input heterogeneous graph $G = (V, R)$ consists of all types of nodes, that is $V = Q \cup T \cup P$ with the number of nodes $|V| = |Q| + |T| + \sum_{s_i \in P} |s_i|$, where $|T|$ and $|s_i|$ are the number of table cells and paragraph sentences respectively. For relations, $R = R_I \cup R_C$. R_I denotes intra-relation and R_C denotes inter-relation. Details of the node types and relation types are described in the Appendix A. Finally, we model the question, table and text as a graph.

3.2 Graph Input Module

The graph input module aims to initialize embeddings for both nodes and edges. For edges, the edge features are directly initialized from a parameter matrix. For nodes, we can obtain their representations from a pre-trained language model (PLM) such as RoBERTa (Liu et al., 2019b). We flatten all question words, table cells and paragraph words into a sequence $[\mathrm{CLS}]q_1q_2...q_{|Q|}[\mathrm{SEP}]t_1t_2...t_{|T|}[\mathrm{SEP}]s_{10}s_{11}...$ $s_{|P|0}s_{|P|1}...[\mathrm{SEP}]$. s_{i0} is the special sentence token of the sentence s_i. Since each word e of the sequence is tokenized into sub-words, we use **Multi-granularity type aware pooling** to get the node representation x. Details in the Appendix B.

3.3 Relational Graph Enhanced Module

This module aggregates information about the nodes and edges of the heterogeneous graph. It is a stack of L relational graph attention network (RGAT) layers. In each layer l, the RGAT (relational graph attention transformers) (Wang et al., 2020) models the graph G and computes the output representation $\mathbf{Z}$ by:

$$
\begin{aligned}
e_{ij}^{(h)} &= \frac{\mathrm{x}_i \mathrm{W}_q^{(h)} (\mathrm{x}_j \mathrm{W}_k^{(h)} + \mathrm{r}_{ij}^K)^{(\mathrm{T})}}{\sqrt{d_z / H}} \\
\alpha_{ij}^{(h)} &= \mathrm{softmax}\left\{ e_{ij}^{(h)} \right\} \\
\mathrm{z}_i^{(h)} &= \sum_{v_j \in \mathcal{N}_i} \alpha_{ij}^{(h)} (\mathrm{x}_j \mathrm{W}_v^{(h)} + \mathrm{r}_{ij}^V)
\end{aligned}
\tag{1}
$$

where matrices W_q,W_k,W_v are trainable parameters in self-attention, and $\mathcal{N}_i$ is the receptive field

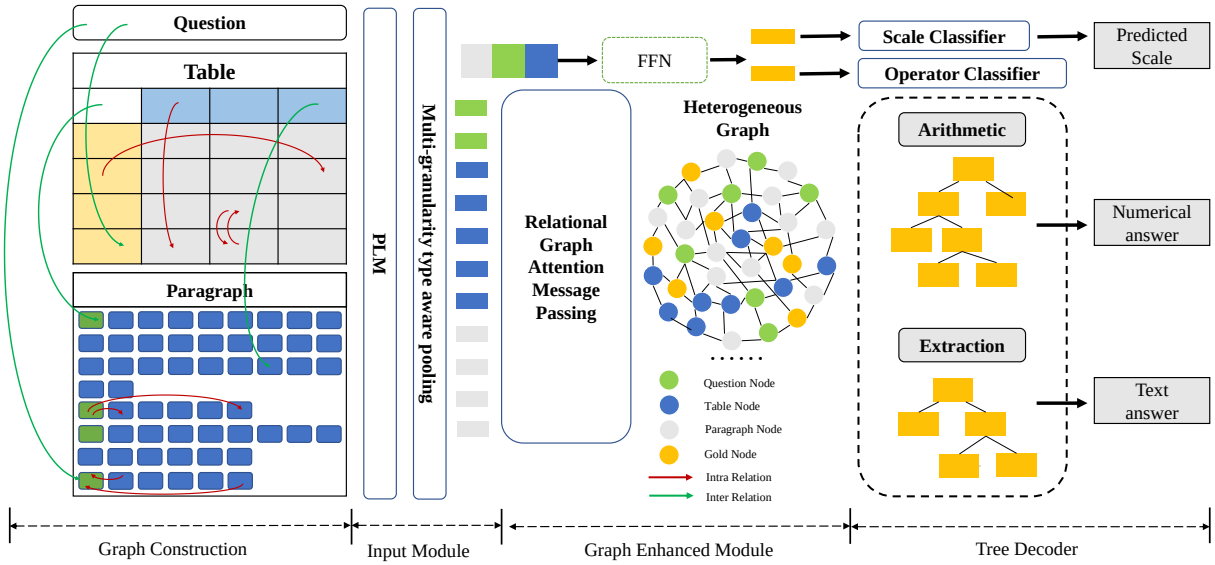

Figure 3: The overall model architecture. The dashed box is the tree-based decoder. Depending on the type of question, two separate trees are constructed to generate the answer.

of node v_i. The output matrices of the final layer L are the desired outputs of the encoder: $\mathrm{Z} = \mathrm{Z}^L$.

3.4 Tree-based Decoder Module

Inspired by the goal-driven tree structure (GTS) (Xie and Sun, 2019) for solving math word problem, we propose a novel tree-based decoder to construct the calculation expressions for solving text-table numerical reasoning problem. As such, the specialized tree decoder generates an equation following the pre-order traversal ordering. The model takes in question Q, table T, paragraph P and generates a expression tree $\mathrm{T_e}$. Let V_{num} denote numeric values in T and P. Generally, V_{con} denotes constant values $V_{con} = \{1, \mathrm{AVG}\}$, AVG means to average the sum of the previous numbers. V_{op} denotes mathematical operators $V^{op} = \{+, -, \times, \div\}$.

The tree generation process is designed as a pre-order tree traversal (root-left-right). For node y in target $\mathrm{T_e}$, $y \in V^{num} \cup V^{con} \cup V^{op}$. We set V_{num} and V_{con} to be the leaf nodes and V_{op} serve as the internal nodes and must have two child nodes.

The tree structured decoder uses the final graph layer representations z_i as input and generates the target expression in t time steps. At each time step t, let $\mathrm{s_t}$ denote the decoding hidden state, $\mathrm{c_t}$ denotes the hybrid **c**ontext state, $\mathrm{g_t}$ denotes the **g**enerated expressions tree state.

The decoder is a bi-directional GRU (Cho et al., 2014), which updates its states at time step $\mathrm{t}+1$ as follows:

$$\mathrm{s_{t+1}} = \mathrm{BiGRU}([\mathrm{c_t} : g_t : \mathrm{E}(y_t)], \mathrm{s_t})$$

where $\mathrm{E}(y_t)$ is the embedding of token $\mathrm{y_t}$:

$$\mathrm{E}(y_t) = \begin{cases} \mathbf{M}_{op}(y_t) & \textbf{if } \mathbf{y_t} \in V^{op} \\ \mathbf{M}_{con}(y_t) & \textbf{if } \mathbf{y_t} \in V^{con} \\ h^i_{loc(y_t,T,P)} & \textbf{if } \mathbf{y_t} \in V^{num} \end{cases} \quad (2)$$

$\mathbf{M}_{op}$ and $\mathbf{M}_{con}$ are two trainable embeddings for operators and constants, respectively. For a numeric value in V^{num}, its token embedding takes the corresponding hidden state $h^i_{loc(y_t,T,P)}$, where $loc(y_t, T, P)$ is the index position of y in table T or paragraph P (Hong et al., 2021).

Inspired by math word problem solving (Wu et al., 2021), the generated expression tree state $\mathrm{g_t}$ is calculated as follows:

$$\mathrm{g_{t+1}} = \sigma(\mathrm{W_g}[\mathrm{g_t} : \mathrm{g_{g,p}} : \mathrm{g_{t,l}} : \mathrm{g_{t,r}}]) \quad (3)$$

where σ is a sigmoid function and $\mathrm{W_g}$ is a weight matrix. For each generated node, $\mathrm{g_{g,p}}$, $\mathrm{g_{g,l}}$, $\mathrm{g_{g,r}}$ represent the expression tree state of the parent node, left child node, and right child node of the current node, respectively.

The hybrid context state $\mathrm{c_t}$ is computed via attention mechanism as follows:

$$\alpha_{ti} = \mathrm{softmax}(\tanh \mathrm{W_h}\mathbf{z_i} + \mathrm{W_s}[\mathbf{s_t} : \mathbf{r_t}]))$$
$$\mathbf{c_t} = \sum_{i=1}^{m} \alpha_{ti}\mathbf{z_i} \quad (4)$$

where $\mathrm{W_h}$, $\mathrm{W_s}$ are weight matrices. α_{ti} is the attention distribution on the node representations $\mathbf{z_i}$.

Lastly, the decoder can generate a word from a given vocabulary $V_{op} \cup V_{con}$. It can also generate a number symbol from V_{num}, which is copied a number from the table T or paragraph P. The final distribution is the combination of the generated probability and copy probability:

$$\begin{aligned} p_c &= \sigma(\mathrm{W_z}[\mathbf{s_t} : \mathbf{c_t} : \mathbf{r_t}]) \\ \mathrm{P_c}(y_t) &= \sum_{y_t=\mathbf{x_i}} \alpha_{ti} \\ \mathrm{P_g}(y_t) &= \mathrm{softmax}(f([\mathbf{s_t} : \mathbf{c_t} : \mathbf{r_t}])) \\ \mathrm{P}(y_t|y_{<t}, \mathrm{X}) &= p_c \mathrm{P_c}(y_t) + (1 - p_c)\mathrm{P_g}(y_t) \end{aligned} \tag{5}$$

Here, $f(\cdot)$ is a perception layer. p_c is the probability that the current word is a number copied from the table or paragraphs.

3.5 Operator and Scale Prediction

In addition, there are two separate tasks in the decoding section: **operator prediction** and **scale prediction**. For arithmetic questions, a right prediction of a numerical answer should include the right number and the correct scale. The scale in the dataset may be *None*, *Thousand*, *Million*, *Billion*, and *Percent* generally. We focus on arithmetic questions for operator prediction, but there are still non-arithmetic questions (span extraction question) in the dataset. So we classify whether the question is arithmetic or not before decoding. For the extraction questions, as shown in Figure 3, we also model them as trees, as described in Appendix C.

To predict the right aggregation operator and scale, two multi-class classifiers are developed. In particular, we take the vector $\langle \mathrm{CLS} \rangle$ to compute the probability:

$$\begin{aligned} \mathrm{p^{op}} &= \mathrm{softmax}(\mathrm{FFN}([\langle \mathrm{CLS} \rangle; h_Q; h_T; h_P])) \\ \mathrm{p^{scale}} &= \mathrm{softmax}(\mathrm{FFN}([\langle \mathrm{CLS} \rangle])) \end{aligned} \tag{6}$$

where h_Q,h_T and h_P are the representations of the question, the table and the paragraphs , respectively, which are obtained by applying an average pooling over the representations of their corresponding tokens. ";" denotes concatenation, and FFN denotes a two-layer feed-forward network with the GELU activation.

3.6 Training

To optimize **RegHNT**, the overall loss is the sum of the loss of the above tasks:

$$\begin{aligned} \mathcal{L} &= \mathcal{L}_{tree} + \mathcal{L}_{op} + \mathcal{L}_{scale} \\ \mathcal{L}_{tree} &= -\sum_{t=1}^{T} \log \mathbf{P}(y_t|y_{<t}, \mathrm{Q}, \mathrm{T}, \mathrm{P}) \\ \mathcal{L}_{op} &= \mathrm{NLL}(\log(\mathrm{P^{op}}), \mathrm{G^{op}}) \\ \mathcal{L}_{scale} &= \mathrm{NLL}(\log(\mathrm{P^{scale}}), \mathrm{G^{scale}}) \end{aligned} \tag{7}$$

$\mathcal{L}_{tree}$ is the loss function of training the tree-decoder, and we use the cross-entropy loss. $\mathcal{L}_{op}$ and $\mathcal{L}_{scale}$ are the loss functions for operator prediction and scale prediction, respectively, where $\mathrm{NLL}(\cdot)$ is the negative log-likelihood loss. $\mathrm{G^{op}}$ comes from the supporting evidence, which is extracted from the annotated answer and derivation. $\mathrm{G^{scale}}$ uses the annotated scale of the answer. We add up the three loss functions as the total loss function.

4 Experiments

4.1 Dataset and Evaluation Metrics

TAT-QA (Zhu et al., 2021) is a large-scale, hybrid QA dataset which contains numerical reasoning and span extraction questions. And the contents of TAT-QA include both tabular and textual data from real financial reports. It contains a total of 2,757 hybrid contexts and 16,552 corresponding question-answer pairs. The detailed statistics are shown in Appendix D. The original dataset contains four types of questions: *Span*, *Multi-Span*, *Count*, *Arithmetic*. But in our setup, there are two types of questions: *Span Extraction* and *Arithmetic*. And our work mainly focuses on answering the *Arithmetic* questions.

For evaluation, we adopt the Exact Match (EM) and numeracy-focused F1 score (Dua et al., 2019) to measure the performance of different QA models. All of which are computed using the official evaluation script[3]. We submit our model to the organizer of the challenge for evaluation. The evaluation detail can be found on the original paper (Zhu et al., 2021).

4.2 Implementation Details

Implementations. Our model is implemented with PyTorch (Paszke et al., 2019), and the graphs are

[3] https://github.com/NExTplusplus/tat-qa

constructed with the library DGL (Wang et al., 2019b). In the graph input module, we use pre-trained language models (PLMs) RoBERTa (Liu et al., 2019b) to obtain the initial representations. During evaluation, we adopt beam search decoding with beam size 3.

Hyper-parameters. In the encoder, the number of GNN layers L is 8, and the number of heads in multi-head attention is 8. For PLMs, we use learning rate 1e-5 and weight decay rate 0.01. For other model modules, we use a larger learning rate 1e-4, and a weight decay rate 5e-5. In the decoder, The recurrent dropout rate (Gal and Ghahramani, 2016) is 0.2 for GRU. The number of heads in multi-head attention is 8 and the dropout rate of features is set to 0.1 in both the encoder and decoder. Throughout the experiments, we use AdamW (Loshchilov and Hutter, 2018) optimizer with a linear warmup scheduler. The warmup ratio of the total training steps is 0.06. The batch size is 48, and the training epoch is 100. The training process may take around 2 days using a single NVIDIA GeForce RTX 3090.

Baselines. We compare with the standard TAGOP (Zhu et al., 2021), which first applies sequence tagging to extract relevant cells from the table and text spans from the paragraphs. We also compare with other advanced models, which can be found on the TAT-QA challenge leaderboard[4]. There are no linked papers to the submissions as yet. We compare our model's performance on the test split with all of them.

4.3 Main Results

Method	Dev		Test	
	EM	F_1	EM	F_1
Human	-	-	84.1	90.8
TAGOP	55.2	62.7	50.1	58.0
LETTER	-	-	56.1	64.3
KIQA	-	-	58.2	67.4
GSReasoner	-	-	67.4	75.5
RegHNT	**73.6**	**81.3**	**70.3**	**78.0**

Table 1: The performance of different models on dev and test set of TAT-QA. The best results are marked in bold.

The main results on the test set are provided in Table 1. Our model achieves the state-of-the-art results in the publicly available TAT-QA benchmark and achieves 20% higher on both EM and F_1 compared with the original baseline (TAGOP), which shows that our model can answer more questions with higher accuracy.

[4] https://nextplusplus.github.io/TAT-QA

	Table	Text	Table-text
	EM/F_1	EM/F_1	EM/F_1
TAGOP			
Span	56.5/57.8	56.5/57.8	68.2/71.7
Spans	66.3/77.0	19.0/59.1	63.2/76.9
Counting	63.6/63.6	-/-	62.1/62.1
Arithmetic	41.1/41.1	27.3/27.3	46.5/46.5
RegHNT			
Span	**68.5/70.0**	**58.7/83.0**	**77.0/84.7**
Spans	**79.5/86.2**	**23.8/65.3**	**81.1/90.1**
Counting	36.3/36.3	-/-	**82.7/82.7**
Arithmetic	**72.7/72.7**	27.3/27.3	**77.7/77.7**

Table 2: Detailed experimental results of TAGOP and RegHNT w.r.t. answer types and sources on the test set.

The detailed results on the test set are provided in Table 2. For almost all types of questions, the accuracy of RegHNT prediction has been improved. Thanks to the tree decoder for arithmetic questions, the model's accuracy on this type of questions has been greatly improved, with an overall improvement of almost 30%. For span extraction questions (*Span*, *Spans*, *Counting*), we observe a improvement in the performance of the model as well. From the perspective of answer sources, compared with the original baseline (TAGOP), we focus on "table-text" type questions, both "arithmetic" and "span extraction" type show a performance improvement of about 20%. It demonstrates that our approach works very well in solving table-text hybrid questions.

This paper focuses on the numerical reasoning questions. To verify our model more precisely, we divided the questions into three categories based on the complexity of the derivation, as shown in Table 3. Simple arithmetic has only one operator. Complex arithmetic has multiple operators, usually used to calculate the average and the rate of change. Undefined arithmetic is arithmetic for which no template is defined in TAGOP. The results show that our model significantly improves both simple and complex arithmetic. In particular, our model can solve undefined arithmetic of TAGOP, which offers flexibility and generalizability.

Arithmetic type	%	TAGOP	RegHNT
Simple arithmetic	41.8	45.0	**79.3**
Complex arithmetic	42.8	60.5	**85.3**
Undefined arithmetic	15.4	0.0	**61.8**

Table 3: Exact match value for different types of arithmetic questions.

Technique	EM	F_1
RegHNT	**73.6**	**81.3**
w/o Intra-relations	72.8	80.4
w/o Inter-relations	72.3	79.8
w/o All relations	71.6	78.7
w/o Multi-granularity type aware pooling	73.0	80.7
w/o Tree-decoder (Span extraction)	72.8	80.6
w/o Tree-decoder (All questions)	60.3	69.5

Table 4: Ablation study of different modules.

4.4 Ablation Studies

Effect of Tree Decoder. Although our work focuses on arithmetic questions, to unify the whole model into a graph-tree framework, we transform the span extraction type question into a tree generation problem as well, as mentioned in Section 3.5. We conducted two experiments using the sequence tagging method in TAGOP (Zhu et al., 2021) instead of generating expression trees. As shown in the last two rows of Table 4, one is to use sequence tagging for all questions, and the other is to use sequence tagging only for span extraction questions. When we change the decoder only for extraction questions, the F_1 drops only 0.7%, but when we change the decoder for all questions, the F_1 drops about 11.8%. It shows that the tree decoder is not only tremendously helpful in solving arithmetic questions but also provides a slight improvement in solving span extraction questions.

Effect of Graph Encoder. As shown in the first four rows of Table 4, we show the effectiveness of the proposed graph encoder. Removing the intra-relations reduces the F_1 value by 0.9% and reduces the EM value by 0.8%. Removing the inter-relations reduces F_1 value by 1.5% and reduces the EM value by 1.3%. When we remove all relations (remove graph enhanced module), the F_1 decreases by 2.6%, and the EM value decreases by 2.0%. From the results, it can be clearly confirmed that the graph we built plays an essential role in modeling table-text hybrid data, and it captures the semantic association through the message passing between different data types.

Effect of Subword Pooling Layer. In the graph input module, we used a multi-granularity type aware pooling method. The type classification criteria for word granularity are text and number, and for node granularity are question, table, and paragraph. As shown in the fifth row "w/o Multi-granularity type aware pooling" of Table 4, we eliminate this mechanism and unify the pooling approach for all types and granularities. Experimental results of F_1 dropped by 0.6%, which shows the effectiveness of this type-aware module.

4.5 Scale and Operater Study

Scale Study. Scale prediction is a unique challenge over TAT-QA and very pervasive in the context of finance. After obtaining the scale, the numerical or string prediction is multiplied or concatenated with the corresponding scale as the final prediction to compare with the ground-truth answer, respectively. We compare RegHNT with the baseline model for scale prediction results. The experimental results are shown in Table 5. Our model has significantly improved performance on both the dev and test datasets. To explore the impact of the scale on results, we use the gold scale to predict the answer. As shown in the third row of Table 6, model accuracy will slightly increase to 84.2% when we use the gold scale, which shows that it is necessary to improve the prediction of scale.

Operator Study. For TAT-QA dataset, there are four original answer types: *Span*, *Multi-Span*, *Count*, *Arithmetic*. As Figure 3 shows, we have adapted it into two categories, where the details of the expression tree for the span extraction questions are in the Appendix C. To investigate whether this category setting causes error propagation, we use the gold operator to predict the answer, and the results are shown in Table 6. When we use the gold operator, the EM and F_1 of the model is improved by only 0.1. It suggests, to some extent, that we divide the data into two categories and use tree decoders to generate the answers separately. This approach has no significant impact on performance.

Model	Dev	Test
TAGOP	93.5	92.2
RegHNT	**95.3**	**93.4**

Table 5: Scale prediction results of our model and baseline.

Model	EM	F_1
RegHNT	73.6	81.4
RegHNT + Gold operator	73.7	81.5
RegHNT + Gold scale	76.5	84.2
RegHNT + Gold operator + Gold scale	**76.7**	**84.3**

Table 6: The performance of using gold operators and gold scales.

4.6 Case Studies

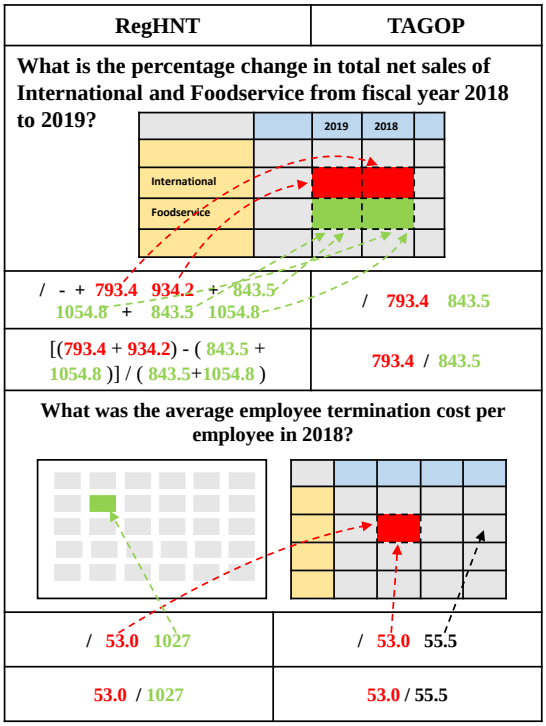

Figure 4: Two examples of generated expressions by RegHNT and TAGOP(Zhu et al., 2021).

As Figure 4 shows, there are two questions and the prediction results of the models. For the question "*What is the percentage change in total net sales...?*", the previous method could not generate complex arithmetic. In contrast, our model can generate the correct prefix expressions, which demonstrates the characteristics and advantages of our tree decoder. For the question "*What was the average employee termination...?*", the correct expression is derived from table and text ("53.0" from table and "1027" from paragraph). The results in Figure 4 show that the model often fails to answer correctly when a question requires the use of both tables and text. It focuses only on tables, choose "53.0" and "55.5". This error type is very common in the previous methods. It shows that our graph encoder can better model the association between tables and texts.

5 Related Work

Table-text hybrid QA is a new task that consists of two main types of work.

Fact Reasoning TextTableQA. Chen et al. (2020b) propose the first table-text hybrid QA dataset. It is the fact reasoning type dataset whose answer is usually a span from the table or linked paragraphs of Wikipedia. The authors supposed HYBRIDER (Chen et al., 2020b), a pipeline approach that divides the prediction process into two phases called linking and reasoning. MITQA (Kumar et al., 2021) achieves SOTA EM result on HybridQA, which is a novel training strategy that works with multiple instances and multiple answers based on weak supervision. DEHG (Feng et al., 2022) propose a document-entity heterogeneous graph network and achieve SOTA F1 score on HybridQA. OTT-QA (Chen et al., 2020a) is a difficult open-domain setting TextTableQA task, which needs retrieval and reading to get the answers. CARP (Zhong et al., 2022) utilizes hybrid chain to model the explicit intermediate reasoning process across table and text for question answering, which achieves SOTA results, but still far from expectations. GeoTSQA (Li et al., 2021) is a multiple choice QA dataset based on geography domain.

Numerical Reasoning TextTableQA. TAT-QA (Zhu et al., 2021) and FinQA (Chen et al., 2021) are the numerical reasoning hybrid dataset which comes from the financial field. Both TAT-HQA (Li et al., 2022) and TAT-DQA (Zhu et al., 2022) are enhanced datasets of TAT-QA, which study TextTableQA in counterfactual condition and multimodal condition, respectively. MULTIHIERTT (Zhao et al., 2022) is a challenging dataset, which contains multiple hierarchical tables and longer unstructured text. Unlike HybridQA, which was fact reasoning type questions, TAT-QA focuses explicitly on finance and needs numerical reasoning for question answering over tabular numbers and associated text. They proposed TAGOP model and regarded this problem as a sequence tagging task. It predefined aggregation operators and used a slot filling method to predict simple derivation, lacking generalizability and flexibility. Our model is the first method to generate arithmetic expressions directly for table-text hybrid numerical reasoning QA.

6 Conclusion

This paper proposes a novel method to solve table-text hybrid numerical reasoning problems and achieve good performance. We present a unified framework for addressing the table schema and relative paragraphs. By adopting relation-aware self-attention, the proposed method jointly learns question, table and paragraph representations based on their alignment. At the same time, we offer a tree-based numerical reasoning decoding framework for hybrid data, the first to use this type of method for this task. Our model can serve as a strong baseline for this task. However, our model has not yet been experimented on encyclopedic type questions (Chen et al., 2020b), and we will explore a table-text hybrid QA framework that integrates dealing with factual and numerical reasoning types of questions. For numerical reasoning type questions, we will do further research on TAT-HQA (Li et al., 2022) and TAT-DQA (Zhu et al., 2022).

Acknowledgements

This work is supported by the Natural Key R&D Program of China (No.2022QY0701), the National Natural Science Foundation of China (No.61922085, No.61976211) and the Strategic Priority Research Program of Chinese Academy of Sciences (Grant No. XDA27020200). This research work was supported by the independent research project of National Laboratory of Pattern Recognition (No. Z-2018013), the Youth Innovation Promotion Association CAS, Yunnan Provincial Major Science and Technology Special Plan Projects (No.202202AD080004) and CCF-DiDi GAIA Collaborative Research Funds for Young Scholars.

References

Ruisheng Cao, Lu Chen, Zhi Chen, Yanbin Zhao, Su Zhu, and Kai Yu. 2021. Lgesql: Line graph enhanced text-to-sql model with mixed local and non-local relations. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 2541–2555.

Wenhu Chen, Ming-Wei Chang, Eva Schlinger, William Yang Wang, and William W Cohen. 2020a. Open question answering over tables and text. In *International Conference on Learning Representations*.

Wenhu Chen, Hanwen Zha, Zhiyu Chen, Wenhan Xiong, Hong Wang, and William Yang Wang. 2020b. Hybridqa: A dataset of multi-hop question answering over tabular and textual data. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 1026–1036.

Zhiyu Chen, Wenhu Chen, Charese Smiley, Sameena Shah, Iana Borova, Dylan Langdon, Reema Moussa, Matt Beane, Ting-Hao Huang, Bryan R Routledge, et al. 2021. Finqa: A dataset of numerical reasoning over financial data. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 3697–3711.

Kyunghyun Cho, Bart Van Merriënboer, Caglar Gulcehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. 2014. Learning phrase representations using rnn encoder-decoder for statistical machine translation. *arXiv preprint arXiv:1406.1078*.

Dheeru Dua, Yizhong Wang, Pradeep Dasigi, Gabriel Stanovsky, Sameer Singh, and Matt Gardner. 2019. Drop: A reading comprehension benchmark requiring discrete reasoning over paragraphs. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 2368–2378.

Yue Feng, Zhen Han, Mingming Sun, and Ping Li. 2022. Multi-hop open-domain question answering over structured and unstructured knowledge. In *Findings of the Association for Computational Linguistics: NAACL 2022*, pages 151–156, Seattle, United States. Association for Computational Linguistics.

Yarin Gal and Zoubin Ghahramani. 2016. A theoretically grounded application of dropout in recurrent neural networks. *Advances in neural information processing systems*, 29.

Yining Hong, Qing Li, Daniel Ciao, Siyuan Huang, and Song-Chun Zhu. 2021. Learning by fixing: Solving math word problems with weak supervision. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 35, pages 4959–4967.

Vishwajeet Kumar, Saneem Chemmengath, Yash Gupta, Jaydeep Sen, Samarth Bharadwaj, and Soumen Chakrabarti. 2021. Multi-instance training for question answering across table and linked text. *arXiv preprint arXiv:2112.07337*.

Moxin Li, Fuli Feng, Hanwang Zhang, Xiangnan He, Fengbin Zhu, and Tat-Seng Chua. 2022. Learning to imagine: Integrating counterfactual thinking in neural discrete reasoning. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 57–69.

Xiao Li, Yawei Sun, and Gong Cheng. 2021. Tsqa: tabular scenario based question answering. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 35, pages 13297–13305.

Qianying Liu, Wenyv Guan, Sujian Li, and Daisuke Kawahara. 2019a. Tree-structured decoding for solving math word problems. In *Proceedings of the 2019 conference on empirical methods in natural language processing and the 9th international joint conference on natural language processing (EMNLP-IJCNLP)*, pages 2370–2379.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019b. Roberta: A robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692*.

Ilya Loshchilov and Frank Hutter. 2018. Decoupled weight decay regularization. In *International Conference on Learning Representations*.

Panupong Pasupat and Percy Liang. 2015. Compositional semantic parsing on semi-structured tables. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 1470–1480.

Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, et al. 2019. Pytorch: An imperative style, high-performance deep learning library. *Advances in neural information processing systems*, 32.

Pranav Rajpurkar, Jian Zhang, Konstantin Lopyrev, and Percy Liang. 2016. Squad: 100,000+ questions for machine comprehension of text. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 2383–2392.

Alon Talmor and Jonathan Berant. 2018. The web as a knowledge-base for answering complex questions. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 641–651.

Bailin Wang, Richard Shin, Xiaodong Liu, Oleksandr Polozov, and Matthew Richardson. 2020. Rat-sql: Relation-aware schema encoding and linking for text-to-sql parsers. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7567–7578.

Lei Wang, Dongxiang Zhang, Jipeng Zhang, Xing Xu, Lianli Gao, Bing Tian Dai, and Heng Tao Shen. 2019a. Template-based math word problem solvers with recursive neural networks. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, pages 7144–7151.

Minjie Wang, Da Zheng, Zihao Ye, Quan Gan, Mufei Li, Xiang Song, Jinjing Zhou, Chao Ma, Lingfan Yu, Yu Gai, et al. 2019b. Deep graph library: A graph-centric, highly-performant package for graph neural networks. *arXiv preprint arXiv:1909.01315*.

Qinzhuo Wu, Qi Zhang, and Zhongyu Wei. 2021. An edge-enhanced hierarchical graph-to-tree network for math word problem solving. In *Findings of the Association for Computational Linguistics: EMNLP 2021*, pages 1473–1482.

Zhipeng Xie and Shichao Sun. 2019. A goal-driven tree-structured neural model for math word problems. In *IJCAI*, pages 5299–5305.

Wen-tau Yih, Matthew Richardson, Christopher Meek, Ming-Wei Chang, and Jina Suh. 2016. The value of semantic parse labeling for knowledge base question answering. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 201–206.

Tao Yu, Rui Zhang, Kai Yang, Michihiro Yasunaga, Dongxu Wang, Zifan Li, James Ma, Irene Li, Qingning Yao, Shanelle Roman, et al. 2018. Spider: A large-scale human-labeled dataset for complex and cross-domain semantic parsing and text-to-sql task. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 3911–3921.

Jipeng Zhang, Lei Wang, Roy Ka-Wei Lee, Yi Bin, Yan Wang, Jie Shao, and Ee-Peng Lim. 2020. Graph-to-tree learning for solving math word problems. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 3928–3937.

Yilun Zhao, Yunxiang Li, Chenying Li, and Rui Zhang. 2022. MultiHiertt: Numerical reasoning over multi hierarchical tabular and textual data. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 6588–6600, Dublin, Ireland. Association for Computational Linguistics.

Wanjun Zhong, Junjie Huang, Qian Liu, Ming Zhou, Jiahai Wang, Jian Yin, and Nan Duan. 2022. Reasoning over hybrid chain for table-and-text open domain question answering. In *Proceedings of the Thirty-First International Joint Conference on Artificial Intelligence, IJCAI-22*, pages 4531–4537. International Joint Conferences on Artificial Intelligence Organization. Main Track.

Fengbin Zhu, Wenqiang Lei, Fuli Feng, Chao Wang, Haozhou Zhang, and Tat-Seng Chua. 2022. Towards complex document understanding by discrete reasoning. *arXiv preprint arXiv:2207.11871*.

Fengbin Zhu, Wenqiang Lei, Youcheng Huang, Chao Wang, Shuo Zhang, Jiancheng Lv, Fuli Feng, and Tat-Seng Chua. 2021. Tat-qa: A question answering benchmark on a hybrid of tabular and textual content in finance. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 3277–3287.

A Relation Detail

Before defining relation types, we'd better introduce the node types more finely. There are three types of nodes, but they can be divided in more detail. Table T has complex structural information. We set three types according to the location of the cell, namely $\mathbf{T_{row}}$, $\mathbf{T_{column}}$ and $\mathbf{T_{cell}}$. Some row headers and column headers are time (e.g. 2019). We set these special cell as $\mathbf{T_{time}}$. Row headers and column headers are composed of text, while numerical cells are numbers. For the question Q, there is only one node type $\mathbf{Q_{word}}$. For the paragraph P, there are two node types, $\mathbf{P_{word}}$ represents the common sentence words, $\mathbf{P_{sentence}}$ represents the special sentence token.

Now there are seven types of nodes, and we establish edges between them, which is mainly divided into intra-relation $\mathrm{R_I}$ and inter-relation $\mathrm{R_C}$. The relations are based on the table schema and word matching between resources, and details are described in Table 7.

B Multi-granularity type aware pooling

Since each word e of the sequence is tokenized into sub-words, we need to aggregate them in order to obtain the node representation. We set two granularities (word, node) and three types (question, table, paragraph) aware pooling method.

- Word level: For number word (e.g. 109.7) and text word (e.g. compensation), we use two independent subword attentive pooling module depending on word type to get the type-aware word level representation w.

$$\mathrm{w} = \sum_i \mathrm{softmax}_i \left[\tanh(\mathrm{e}_i \mathrm{W}_s)\mathrm{v}_s^{\mathrm{T}}\right] \mathrm{e}_i$$

- Node Level: For a node, especially a table cell node, which usually consists of multiple words. According to the node source types, we aggregate the word level granularity representation with three different pooling layers and three different BiLSTM to get the node representation x.

$$\mathrm{x} = \sum_i \mathrm{softmax}_i \left[\tanh(\mathrm{w}_i \mathrm{W}_n)\mathrm{v}_n^{\mathrm{T}}\right] \mathrm{w}_i$$

Where v_s, W_s, v_n, W_n are trainable parameters. The attentive pooling layer is inspired by LGESQL (Cao et al., 2021).

C Details of the span extraction question

We also predict an expression tree for span extraction questions to get the answer. The example is shown in Figure 5. Unlike mathematical problems, the leaf node in the span extraction tree represents node ID, while the leaf node in the arithmetic tree represents the numeric number or constant. As for the operator, the operators in the arithmetic tree are "+, -, ×, ÷". But for the span extraction tree, we define three operators, "+, ×, C". "+" represents the splicing of two discontinuous spans, corresponding to the *multi-span* in the original dataset. "×" means taking the operator's left and right sides as the starting and ending nodes and selecting all nodes in the middle of the two nodes as a span. "C" is the same as "+", but it counts the number of spans instead of slicing them.

	Span Extraction Type
Answer	integration and transformation-related expenses severance and retention compensation expenses transaction-related expense
Expression	7 + 10 + 44 * 47
Polish Notation	+ + 7 10 * 44 47
Tree	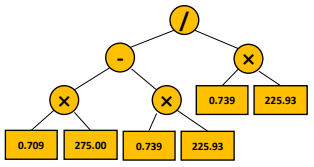

Figure 5: Expressions for span extraction questions. We also predict an expression tree to get the answer. "44×47" means to select all words between nodes ID 44 and nodes ID 47, "+" means that the span on both sides is the answer.

D Details of the TAT-QA

The specific data analysis of the dataset is shown in Table 8 and Table 9.

Intra-Relation			
Source x	Source y	Relation	Description
$\mathrm{T}_{\mathrm{row}}$	$\mathrm{T}_{\mathrm{cell}}$	CONTAIN	x is row head of y.
$\mathrm{T}_{\mathrm{column}}$	$\mathrm{T}_{\mathrm{cell}}$	CONTAIN	x is column head of y.
$\mathrm{T}_{\mathrm{time}}$	$\mathrm{T}_{\mathrm{cell}}$	CONTAIN	x is row/column head of y, x is time.
$\mathrm{P}_{\mathrm{sentence}}$	$\mathrm{P}_{\mathrm{word}}$	CONTAIN	x is the sentence token which contain y.
$\mathrm{Q}_{\mathrm{word}}$	$\mathrm{Q}_{\mathrm{word}}$	DISTANCE+1	y is the next word of x.
$\mathrm{P}_{\mathrm{word}}$	$\mathrm{P}_{\mathrm{word}}$	DISTANCE+1	y is the next word of x.
$\mathrm{T}_{\mathrm{cell}}$	$\mathrm{T}_{\mathrm{cell}}$	SAME ROW	x and y are in the same row.
$\mathrm{T}_{\mathrm{cell}}$	$\mathrm{T}_{\mathrm{cell}}$	SAME COLUMN	x and y are in the same column.
Inter-Relation			
$\mathrm{Q}_{\mathrm{Word}}$	$\mathrm{T}_{\mathrm{row}}$	PARTIALMATCH EXACTMATCH	x is part of y, but the entire question does not contain y. x is part of y, and y is a span of the entire question.
$\mathrm{P}_{\mathrm{sentence}}$	$\mathrm{Q}_{\mathrm{word}}$	CONTAIN	x is the sentence token which contain y.
$\mathrm{P}_{\mathrm{sentence}}$	$\mathrm{T}_{\mathrm{row}}$	CONTAIN	x is the sentence token which contain y.
$\mathrm{P}_{\mathrm{word}}$	$\mathrm{T}_{\mathrm{row}}$	PARTIALMATCH EXACTMATCH	x is part of y, but the entire sentence does not contain y. x is part of y, and y is a span of the entire question.
$\mathrm{P}_{\mathrm{word}}$	$\mathrm{Q}_{\mathrm{word}}$	SAME	x and y are the same words.

Table 7: The checklist of all relations in our RegHNT. All the above relations are asymmetric. We show only one direction, and the opposite direction can be easily inferred.

	Table	**Text**	**Table-text**	**Total**
Span	1,801	3,496	1,842	7,139
Spans	777	258	1,037	2,072
Counting	106	5	266	377
Arithmetic	4,747	143	2,074	6,964
Total	7,431	3,902	5,219	16,552

Table 8: Question distribution regarding different answer types and sources in TAT-QA.

Statistic	**Train**	**Dev**	**Test**
# of hybrid contexts	2,201	278	278
# of questions	13,215	1,668	1,669
Avg. rows / table	9.4	9.7	9.3
Avg. cols / table	4.0	3.9	4.0
Avg. paragraphs / table	4.8	4.9	4.6
Avg. paragraph len [words]	43.6	44.8	42.6
Avg. question len [words]	12.5	12.4	12.4
Avg. answer len [words]	4.1	4.1	4.3

Table 9: Basic statistics of each split in TAT-QA.

Perform Like an Engine: A Closed-Loop Neural-Symbolic Learning Framework for Knowledge Graph Inference

Guanglin Niu[1], **Bo Li**[1,2]*, **Yongfei Zhang**[3,4], **Shiliang Pu**[5]
[1] Institute of Artificial Intelligence, Beihang University, Beijing, China
[2] Hangzhou Innovation Institute, Beihang University, Hangzhou, China
[3] Beijing Key Laboratory of Digital Media, Beihang University, Beijing, China
[4] State Key Laboratory of Virtual Reality Technology and Systems, Beihang University, Beijing, China [5] Hikvision Research Institute, Hangzhou, China
{beihangngl, boli, yfzhang}@buaa.edu.cn, pushiliang.hri@hikvision.com

Abstract

Knowledge graph (KG) inference aims to address the natural incompleteness of KGs, including rule learning-based and KG embedding (KGE) models. However, the rule learning-based models suffer from low efficiency and generalization while KGE models lack interpretability. To address these challenges, we propose a novel and effective closed-loop neural-symbolic learning framework **EngineKG** via incorporating our developed KGE and rule learning modules. KGE module exploits symbolic rules and paths to enhance the semantic association between entities and relations for improving KG embeddings and interpretability. A novel rule pruning mechanism is proposed in the rule learning module by leveraging paths as initial candidate rules and employing KG embeddings together with concepts for extracting more high-quality rules. Experimental results on four real-world datasets show that our model outperforms the relevant baselines on link prediction tasks, demonstrating the superiority of our KG inference model in a neural-symbolic learning fashion. The source code and datasets of this paper are available at https://github.com/ngl567/EngineKG.

1 Introduction

Typical knowledge graphs (KGs) store triple facts and some of them also contain concepts of entities (Bollacker et al., 2008). The KGs have proven to be incredibly effective for a variety of applications such as dialogue system (Zhou et al., 2018) and question answering (Huang et al., 2019). However, the existing KGs are always incomplete which restricts the performance of knowledge-based applications. Thus, KG inference plays a vital role in completing KGs for better applications of KGs.

The existing KG inference approaches are usually classified into two main categories: (1) Rule learning-based models such as AMIE+ (Galárraga et al., 2015) and AnyBurl (Meilicke et al., 2019) mine rules from KGs and employ these rules to predict new triples by deduction. However, rule learning-based models suffer from **low efficiency** of the rule mining process and the **poor generalization** caused by the limited coverage of inference patterns. (2) KGE technique learns the embeddings of entities and relations to predict the missing triples via scoring each triple candidate, including TransE (Bordes et al., 2013), HAKE (Zhang et al., 2020) and DualE (Cao et al., 2021). The previous KGE models perform in a data-driven fashion, contributing to good efficiency and generalization but **lacking interpretability**.

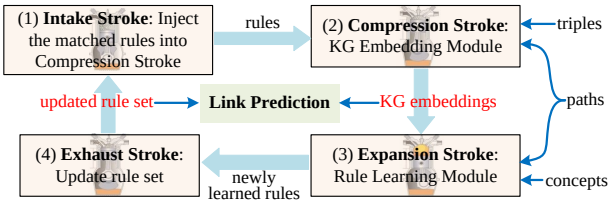

Figure 1: The brief architecture of our closed-loop framework for KG inference EngineKG that performs like a four-stroke engine.

Some recent researches attempt to combine the advantages of rule learning-based and KGE-based models to complement each other in a **neural-symbolic learning fashion**. An idea is to introduce logic rules into KGE models, such as RUGE (Guo et al., 2018) and its advanced model IterE (Zhang et al., 2019b). These approaches all convert the rules into formulas by t-norm based fuzzy logic to obtain newly labeled triples. However, these models **cannot maintain the interpretability which is a vital feature of symbolic rules**. On the other hand, some rule learning-based models succeed in leveraging KG embeddings to extract rules via numerical calculation rather than discrete graph search, including RNNlogic (Qu et al., 2021), RLvLR (Omran et al., 2019), DRUM (Sadeghian et al., 2019) and RuLES (Ho et al., 2018). Although

*Corresponding author.

Proceedings of the 29th International Conference on Computational Linguistics, pages 1391–1400
October 12–17, 2022.

the efficiency of mining rules is improved, **the performance especially generalization of purely employing rules to implement KG inference is still limited**.

To address the above challenges, we propose a closed-loop neural-symbolic learning framework **EngineKG** via combining an embedding-based rule learning and a rule-enhanced KGE, in which paths and concepts are utilized. Our model is named **EngineKG** because it performs like an engine as shown in Figure 1: (1) **Intake Stroke**. The closed-path rules (or named chain rules) are injected into the KGE module (analogous to intake) to guide the procedure of learning KG embeddings, where the initial seed rules are mined by any rule learning tool, and the rule set would grow via our designed rule learning module from the first iteration. (2) **Compression Stroke**. The KGE module leverages the rules and paths to learn the low-dimensional embeddings (analogous to compression) of entities and relations, improving interpretability and accuracy. (3) **Expansion Stroke**. The novel rule learning module outputs newly learned rules (analogous to exhaust) by the effective rule pruning strategy based on paths, relation embeddings and concepts. (4) **Exhaust Stroke**. Update the rule set (analogous to exhaust) by merging the previous rule set and the newly learned rules for boosting KGE and KG inference in the next iteration.

Our research makes three contributions:

- We propose a novel and effective closed-loop neural-symbolic learning framework that performs embedding-based rule learning and rule-enhanced KGE iteratively, balancing good accuracy, interpretability and efficiency.
- Paths and ontological concepts are well exploited for supplementing the valuable semantics to both KGE and rule learning, facilitating the better performance of KG inference.
- The link prediction results and the effectiveness of rule learning on four datasets illustrate that our model outperforms various state-of-the-art KG inference approaches.

2 Related Work

2.1 Rule Learning-Based Models

According to the symbolic characteristics of KG, some rule learning techniques specific to KGs are applied to KG inference with relatively good accuracy and interpretability, including AMIE+ (Galárraga et al., 2015), Anyburl (Meilicke et al., 2019), DRUM (Sadeghian et al., 2019), RLvLR (Omran et al., 2019) and RNNLogic (Qu et al., 2021). AMIE+ (Galárraga et al., 2015) introduces optimized query writing techniques into traditional inductive logic programming algorithms to generate horn rules efficiently. Anyburl learns closed-path rules from KGs in a reinforcement learning framework. DRUM, RLvLR and RNNLogic employ KG embeddings for enhancing the efficiency and scalability of rule learning. Whereas, all the previous rule learning algorithms lack generalization since the number of rules mined at one time is limited.

2.2 KG Embedding Models

The typical KG embedding (KGE) models learn the embeddings of entities and relations to measure the plausibility of each triple. TransE (Bordes et al., 2013) regards the relations as translation operations from head to tail entities. ComplEx (Trouillon et al., 2016) embeds the KG into a complex space while DualE (Cao et al., 2021) embeds relations into the quaternion space to model the symmetric and antisymmetric relations. HAKE (Zhang et al., 2020) embeds entities into the polar coordinate system and is able to model the semantic hierarchies of KGs. RUGE (Guo et al., 2018) and IterE (Zhang et al., 2019b) both convert rules into formulas by t-norm fuzzy logic to infer newly labeled triples. Particularly, IterE iteratively conducts rule learning and KG embedding, but **the significant distinctions between our model EngineKG and IterE** include: (1) **Usage of rules**: our model leverages rules to compose paths for learning KG embeddings while IterE uses rules to produce labeled triples. Meanwhile, we maintain the interpretability of symbolic rules, while IterE does not. (2) **Additional information**: our model introduces paths and concepts into both rule learning and KG embedding while IterE simply depends on triples.

2.3 Path-Enhanced Models

In terms of the graph structure of KGs, paths denote the associations between entities apart from relations and are applied to multi-hop reasoning (Lin et al., 2018; Xiong et al., 2017; Neelakantan et al., 2015). PTransE (Lin et al., 2015) extends TransE by measuring the similarity between relation and path embeddings. MultiHopKG (Lin et al., 2018) explores the answer entities via searching corre-

sponding paths with reinforcement learning. However, these models represent paths in a data-driven fashion, lacking interpretability and accuracy.

3 Methodology

In this section, we first describe the problem formulation and notation of our work in section 3.1. Then, following the workflow of EngineKG as shown in Figure 2, we introduce the rule-enhanced KGE module in section 3.2 and the embedding-based rule learning module in section 3.3.

3.1 Problem Formulation and Notation

Definition of Closed-Path Rule. The closed-path (CP) rule or named chain rule is a fragment of the horn rule, which we are interested in for the KGE module and the inference. A CP rule is of the form

$$Head(x,y) \Leftarrow B_1(x,z_1) \wedge B_2(z_1,z_2) \wedge \cdots \wedge B_n(z_{n-1},y) \quad (1)$$

where $B_1(x,z_1)$, $B_2(z_1,z_2)$, $\cdots$, $B_n(z_{n-1},y)$ denote the atoms in the rule body $Body(x,y)$, and $Head(x,y)$ is the rule head. B_i and $Head$ indicate relations. Standard confidence (SC) and head coverage (HC) are two predefined statistical measurements to assess rules (Galárraga et al., 2015; Omran et al., 2019), which are defined as follows:

$$Support = \#(e,e') : Body(e,e') \wedge Head(e,e') \quad (2)$$

$$SC = \frac{Support}{\#(e,e') : Body(e,e')} \quad (3)$$

$$HC = \frac{Support}{\#(e,e') : Head(e,e')} \quad (4)$$

where $\#(e,e')$ indicates the number of entity pairs (e,e') that satisfy the condition on the right side of the colon. In general, the rules with SC and HC both higher than 0.7 are regarded as high-quality rules (Zhang et al., 2019b).

Definition of Path. A path between an entity pair (h,t) is in the form of $[h \rightarrow r_1 \rightarrow e_1 \rightarrow \cdots \rightarrow r_n \rightarrow t]$ where r_i and e_i are the intermediate relation and entity, and the length of a path is the number of the intermediate relations.

3.2 Rule-Enhanced KGE Module

We aim to learn the entity and relation embeddings from triple facts, rules and paths via neural-symbolic learning. Firstly, we extract the paths via PCRA algorithm (Lin et al., 2015). Apart from other path-finding approaches such as PRA (Lao et al., 2011), PCRA algorithm could measure the reliability of each path for KGE module. Particularly, we develop a joint logic and data-driven path representation mechanism to learn path embeddings.

Logic-Driven Path Representation (Intake Stroke). The CP rules could compose paths into shorter and more accurate ones for enhancing the representation of paths. For instance, a length-2 path $[The\ Pursuit\ of\ Happiness \xrightarrow{CastActor} Will\ Smith \xrightarrow{PersonLanguage} English]$ as shown in Figure 2 could be composed into a shorter path (actually a triple) $[The\ Pursuit\ of\ Happiness \xrightarrow{TVLanguage} English]$ via the CP rule $TVLanguage(x,y) \Leftarrow CastActor(x,z) \wedge PersonLanguage(z,y)$. Furthermore, the relation $TVLanguage$ could signify the original multi-hop path.

Data-Driven Path Representation. For the scenario that the path cannot be further composed by rules such as the path $[The\ Pursuit\ of\ Happiness \xrightarrow{CountryOfOrigin} U.S.A. \xrightarrow{LanguageSpoken} English]$ in Figure 2, we represent this path by summing all the relation embeddings along the path. With the entity pair (h,t) together with the linking path set $\mathcal{P}$, the energy function for measuring the plausibility of the path-specific triple $(h,t,\mathcal{P})$ is designed as

$$E_p(h,t,\mathcal{P}) = \sum_{pi \in \mathcal{P}} \frac{R(p_i|h,t)}{\sum_{pi \in \mathcal{P}} R(p_i|h,t)} \|\mathbf{h} + \mathbf{p}_i - \mathbf{t}\| \quad (5)$$

where $\mathbf{h}$ and $\mathbf{t}$ are the head and tail entity embeddings. p_i denotes the i-th path in the path set $\mathcal{P}$ and $\mathbf{p}_i$ is the embedding of p_i achieved by the joint logic and data-driven path representation. $R(p_i|h,t)$ indicates the reliability of path p_i between the given entity pair (h,t) obtained by the PCRA algorithm.

Optimization Objective (Compression Stroke). Along with the translation-based KGE models, the energy function for formalizing the plausibility of a triple fact (h, r, t) is given as

$$E_t(h,r,t) = \|\mathbf{h} + \mathbf{r} - \mathbf{t}\| \quad (6)$$

in which $\mathbf{r}$ is the embedding of the relation r.

The existing KGE techniques neglect the semantic association between relations. Remarkably, the length-1 rules model the causal correlations between two relations. As shown in Figure 2, the relation pair in the rule $FilmLanguage(x,y) \Leftarrow TVLanguage(x,y)$ should have higher similarity

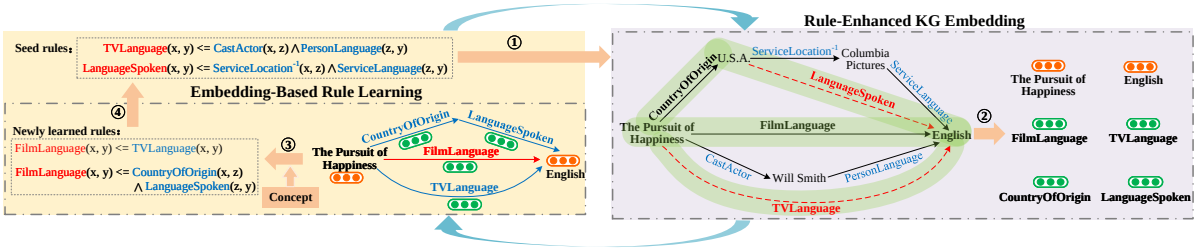

Figure 2: The overall architecture of our developed KG inference model EngineKG in a closed-loop neural-symbolic learning framework. Specific to the rule-enhanced KG embedding module, the green highlighted parts contain the triples and the composed paths via rules, indicating the inputs of the KGE module.

than other relations. Thus, we measure the association between relation pairs as

$$E_r(r_1, r_2) = \|\mathbf{r}_1 - \mathbf{r}_2\| \tag{7}$$

where $\mathbf{r}_1$ and $\mathbf{r}_2$ are the embeddings of relations r_1 and r_2. $E_r(r_1, r_2)$ should be closer to a small value if r_1 and r_2 appear in a length-1 rule at the same time.

With the energy functions specific to the factual triple, the path representation and the relation correlation, the joint loss function for training is designed as follows:

$$L = \sum_{(h,r,t)\in\mathcal{T}} (L_t + \alpha_1 L_p + \alpha_2 L_r) \tag{8}$$

$$L_t = \sum_{(h',r,t')\in\mathcal{T}'} [\gamma_1 + E_t(h,r,t) - E_t(h',r,t')]_+ \tag{9}$$

$$L_p = \sum_{(h',t')\in\mathcal{T}'} [\gamma_2 + E_p(h,t,P) - E_p(h',t',P)]_+ \tag{10}$$

$$L_r = \sum_{r_p\in\mathcal{S}} \sum_{r_n\in\mathcal{S}'} [\gamma_3 + E_r(r,r_p) - E_r(r,r_n)]_+ \tag{11}$$

where L is the whole training loss consisting of three components: the triple-specific loss L_t, the path-specific loss L_p, and the relation correlation-specific loss L_r. α_1 and α_2 are the weights of paths and relation correlation, respectively. γ_1, γ_2 and γ_3 are three margins in each loss function. $[x]_+$ is the function returning the maximum value between 0 and x. $\mathcal{T}$ is the set of triples observed in the KG and $\mathcal{T}'$ is the set of negative samples obtained by random negative sampling. $\mathcal{S}$ is the set of positive relations that are correlated with relation r by length-1 rules and $\mathcal{S}'$ is the set of negative relations beyond $\mathcal{S}$ and relation r.

We employ mini-batch Stochastic Gradient Descent (SGD) algorithm to optimize the joint loss function for learning entity and relation embeddings. The entity and relation embeddings are initialized randomly and constrained to be unit vectors by the additional regularization term with L2 norm.

3.3 Embedding-Based Rule Learning Module

We develop an embedding-based rule learning (Expansion Stroke) to mine high-quality CP rules via conducting the rule searching and the rule pruning efficiently. Remarkably, a path can naturally represent the body of a CP rule. Motivated by this observation, we firstly reuse the paths extracted in section 3.2 and regard these paths as candidate CP rules, which improves the efficiency of rule searching. For instance, given an entity pair (h, t) connected by a relation r and a path $[h \rightarrow r_1 \rightarrow e_1 \rightarrow r_2 \rightarrow e_2 \rightarrow, \cdots \rightarrow e_{n-1} \rightarrow r_n \rightarrow t]$, it can be deduced as a CP rule $r(x, y) \Leftarrow r_1(x, z_1) \wedge r_2(z_1, z_2) \wedge \cdots \wedge r_n(z_{n-1}, y)$, where x, y and $z_i (i = 1, \cdots, n-1)$ are the variables in the rule, and $r_i (i = 1, \cdots, n)$ is a relation.

To evaluate the plausibility of candidate CP rules efficiently, we develop a novel rule pruning strategy consisting of two components: **Embedding-based Semantic Relevance** and **Concept-based Co-occurrence**. It should be noted that the Concept-based Co-occurrence is available when the KG contains concepts. For the KGs without concepts, employing Embedding-based Semantic Relevance solely is still valid to learn rules.

Embedding-based Semantic Relevance. Intuitively, a candidate rule is plausible if the rule body corresponding to a path p is semantically relevant to the rule head corresponding to the relation r. We focus on the paths and the CP rules with lengths no longer than 2 for the trade-off of efficiency and performance. Based on the KG embeddings learned in our KGE module, we could measure the semantic relevance between the body and the head of a candidate rule by the path embedding and relation embedding as well as the score function as

$$E_{sr}(r, p) = \exp(-\|\mathbf{r} - \mathbf{p}\|) \tag{12}$$

where $\mathbf{p}$ denotes the embedding of the path p.

Dataset	#Relation	#Entity	#Concept	#Train	#Valid	#Test
FB15K	1,345	14,951	89	483,142	50,000	59,071
FB15K237	237	14,505	89	272,115	17,535	20,466
NELL-995	200	75,492	270	123,370	15,000	15,838
DBpedia-242	298	99,744	242	592,654	35,851	30,000

Table 1: Statistics of the experimental datasets.

The embedding-based semantic relevance indicates a global plausibility of a rule from the perspective of relations. Furthermore, a concept-based co-occurrence is proposed to evaluate the local relevance of the arguments in a rule.

Concept-based Co-occurrences. The neighbor arguments in a high-quality CP rule are expected to share as many same concepts as possible. Given a CP rule $Nationality(x, y) \Leftarrow BornIn(x, z) \wedge LocatedIn(z, y)$, the tail argument of relation $BornIn$ and the head argument of relation $LocatedIn$ should share the concept $Location$. Considering there are far fewer concepts than entities, we encode each concept as a one-hot representation to maintain the precise concept features. The concept embedding of the head or tail argument of an atom can be formalized as

$$AC_h(r) = \frac{1}{|C_h(r)|} \sum_{c \in C_h(r)} OH(c) \tag{13}$$

$$AC_t(r) = \frac{1}{|C_t(r)|} \sum_{c \in C_t(r)} OH(c) \tag{14}$$

where $AC_h(r)$ and $C_h(r)$ are the concept embedding and concept set in the head argument of an atom containing relation r while $AC_t(r)$ and $C_t(r)$ are that of in the tail argument. $OH(c)$ denotes the one-hot representation of the concept c.

Specific to a CP rule in the form of $r(x, y) \Leftarrow r_1(x, z_1) \wedge r_2(z_1, z_2) \wedge \cdots \wedge r_n(z_{n-1}, y)$, three types of co-occurrence score functions are designed according to the different positions of the overlapped arguments:

$$E^h_{co}(r, r_1) = sim(AC_h(r), AC_h(r_1)) \tag{15}$$

$$E^t_{co}(r, r_n) = sim(AC_t(r), AC_t(r_n)) \tag{16}$$

$$E^i_{co}(r_i, r_{i+1}) = sim(AC_t(r_i), AC_h(r_{i+1})) \tag{17}$$

where $E^h_{co}(r, r_1)$ and $E^t_{co}(r, r_n)$ respectively denote the co-occurrence similarities specific to the head arguments and the tail arguments between the rule head and the rule body. $E^{in}_{co}(r_i, r_{i+1})$ represents the co-occurrence similarity between the adjacent arguments in the rule body. $sim(x, y)$ represents the cosine distance function for measuring the similarity between x and y.

Then, the whole co-occurrence score function can be achieved by composing all the scores in Eqs. 15-17 as

$$\begin{aligned} E_{co}(r, p) = E^h_{co}(r, r_1) + E^t_{co}(r, r_n) \\ + \sum_{i=1}^{n-1} E^i_{co}(r_i, r_{i+1}) \end{aligned} \tag{18}$$

Consequently, the overall score function for evaluating candidate rules is defined as:

$$E_{cg} = E_{sr}(r, p) + \beta E_{co}(r, p) \tag{19}$$

where β is the weight of the co-occurrence score. We set a threshold and select the candidate rules with the scores calculated by Eq. 19 above the threshold as filtered candidate rules. Afterward, we output the high-quality rules from the filtered candidate rules that satisfy the thresholds of the precise quality criteria namely standard confidence and head coverage defined in Eqs. 2-4. Then, the updated rule set is obtained via fusing the newly learned rules and the previous rule set (Exhaust Stroke) for the KGE module in the next iteration.

3.4 Algorithm Flow and Complexity

It is noteworthy that from the first iteration of EngineKG, our rule learning module could potentially achieve sustainable growth of rules. The entire iteration process will keep running until no fresh rules can be generated. Then, the learned KG embeddings learned in the last iteration are exploited for the KG inference. The Algorithm 1 summarizes the whole closed-loop KG inference procedure of our EngineKG model.

To evaluate the complexity of our EngineKG model, we denote n_e, n_r, n_p, n_c and n_t as the amount of entities, relations, paths, concepts and

Algorithm 1: Training framework of our model EngineKG

Input: $\mathcal{G}$: Training set
$\mathcal{C}_h, \mathcal{C}_t$: The head and tail concept set associated with relations
$\mathcal{P}$: The set of paths extracted from $\mathcal{G}$ via PCRA
$\gamma_1, \gamma_2, \gamma_3$: The margins in loss functions
α_1, α_2: The weights for trade-off
st: The score threshold for coarse-grained evaluation of rules
Max_e: The maximum epochs

Initialize entity embeddings **e** and relation embeddings **r** randomly and encode the concept embeddings from $\mathcal{C}_h$ and $\mathcal{C}_t$ by one-hot representation;
Mine rules by a rule mining tool such as AMIE+;
while *new rules can be learned* **do**
  for *epoch=1,2,...,* Max_e **do**
    Sample a minibatch of triples $\mathcal{T}$ from $\mathcal{G}$;
    Compose the paths between the entity pairs in $\mathcal{T}$ by the logic and data-driven path representation described in section 3.2;
    Generate the set of negative samples $\mathcal{T}'$ by the random negative sampling as in TransE (Bordes et al., 2013);
    Update **e** and **r** by optimizing the loss functions in Eqs. 5-11;
  Generate the initial candidate rules from the paths in $\mathcal{P}$;
  Load the learned entity and relation embeddings **e** and **r**;
  Calculate the coarse-grained evaluation score E_{cg} of each candidate rule according to Eqs. 12-19;
  if E_{cg} *of a rule is smaller than* st **then**
    Eliminate this rule;
  Pick out the candidate rules that satisfy the thresholds of the standard confidence and the head coverage;
  Output the newly learned rules;
  Update the rule set by merging the newly learned rules and the rule set in the last iteration;

triples in a KG. The average length of paths is l_p. The embedding dimension of both entities and relations is represented as d. The embedding dimension of concepts is n_c due to the one-hot encoding applied for concept representations. Our model complexity of parameter sizes is $\mathcal{O}(n_e d+n_r d+n_c^2)$. For each iteration in training, the time complexity of our model is $\mathcal{O}(n_t n_p l_p d)$.

4 Experiments

4.1 Experimental Setup

Datasets. Four datasets containing ontological concepts are employed for our experiments, including FB15K (Bordes et al., 2013), FB15K237 (Toutanova and Chen, 2015), NELL-995 and DBpedia-242. Particularly, NELL-995 here is a re-split of the original dataset (Xiong et al., 2017) into training/validation/test sets. DBpedia-242 is generated from the commonly-used KG DBpedia (Lehmann et al., 2015) to ensure each entity in the dataset has a concept. The statistics of the experimental datasets are listed in Table 1.

Baselines. We compare our model EngineKG with two categories of baselines:

(1) The traditional KGE models depending on triple facts: TransE (Bordes et al., 2013), ComplEx (Trouillon et al., 2016), RotatE (Sun et al., 2019), QuatE (Zhang et al., 2019a), HAKE (Zhang et al., 2020) and DualE (Cao et al., 2021).

(2) The models using paths or rules: the path-based model MultiHopKG (Lin et al., 2018), the rule learning-based models RNNLogic (Qu et al., 2021) and RPJE (Niu et al., 2020), and the model combining rules with KG embeddings IterE (Zhang et al., 2019b).

The evaluation results of these baselines are obtained by employing their open-source codes with the suggested hyper-parameters.

Training Details. We implement our model in C++ and on an Intel i9-9900 CPU with a memory of 64G. For a fair comparison, the embedding dimension of all the models is fixed as 100, the batch size is set to 1024 and the number of negative samples is set to 10. Specific to our model, during each iteration, the maximum training epoch is set to 1000, and the standard confidence and the head coverage are selected as 0.7 and 0.1 for better performance. The entity and relation embeddings are initialized randomly. We employ grid search for selecting the best hyper-parameters on the validation dataset.

Models	FB15K					FB15K237				
	MR	MRR	Hits@10	Hits@3	Hits@1	MR	MRR	Hits@10	Hits@3	Hits@1
TransE (Bordes et al., 2013)	117	0.534	0.775	0.646	0.386	228	0.289	0.478	0.326	0.193
ComplEx (Trouillon et al., 2016)	197	0.346	0.593	0.405	0.221	507	0.236	0.406	0.263	0.150
RotatE (Sun et al., 2019)	39	0.612	0.816	0.698	0.488	168	0.317	0.553	0.375	0.231
QuatE (Zhang et al., 2019a)	40	0.765	0.878	0.819	0.693	173	0.312	0.495	0.344	0.222
HAKE (Zhang et al., 2020)	42	0.678	0.839	0.761	0.570	183	0.344	0.542	0.382	0.246
DualE (Cao et al., 2021)	43	0.759	0.882	0.820	0.681	202	0.332	0.522	0.367	0.238
MultiHopKG (Lin et al., 2018)	-	0.670	0.769	0.708	0.612	-	0.385	0.562	0.429	0.298
RNNLogic (Qu et al., 2021)	244	0.496	0.669	0.544	0.405	620	0.280	0.428	0.306	0.205
RPJE (Niu et al., 2020)	40	0.811	0.898	0.832	0.762	207	0.443	0.579	0.479	0.374
IterE (Zhang et al., 2019b)	85	0.577	0.807	0.663	0.451	463	0.210	0.355	0.227	0.139
EngineKG (Ours)	**20**	**0.854**	**0.933**	**0.885**	**0.810**	**121**	**0.555**	**0.707**	**0.590**	**0.479**

Models	DBpedia-242					NELL-995				
	MR	MRR	Hits@10	Hits@3	Hits@1	MR	MRR	Hits@10	Hits@3	Hits@1
TransE (Bordes et al., 2013)	1996	0.256	0.539	0.395	0.075	8650	0.167	0.354	0.219	0.068
ComplEx (Trouillon et al., 2016)	3839	0.196	0.387	0.230	0.104	11772	0.169	0.298	0.185	0.106
RotatE (Sun et al., 2019)	1323	0.308	0.594	0.422	0.143	9620	0.292	0.444	0.325	0.216
QuatE (Zhang et al., 2019a)	1618	0.411	0.612	0.491	0.293	12296	0.281	0.422	0.315	0.207
HAKE (Zhang et al., 2020)	1522	0.379	0.551	0.432	0.283	13211	0.245	0.370	0.283	0.175
DualE (Cao et al., 2021)	1363	0.360	0.592	0.439	0.232	11529	0.292	0.447	0.329	0.214
MultiHopKG (Lin et al., 2018)	-	0.520	0.625	0.530	0.426	-	0.416	0.474	0.345	0.275
RNNLogic (Qu et al., 2021)	7857	0.344	0.514	0.390	0.253	15772	0.335	0.422	0.356	0.290
RPJE (Niu et al., 2020)	1770	0.521	0.576	0.542	0.487	6291	0.360	0.496	0.401	0.288
IterE (Zhang et al., 2019b)	5016	0.190	0.326	0.215	0.120	12998	0.233	0.327	0.246	0.185
EngineKG (Ours)	**1275**	**0.523**	**0.647**	**0.551**	**0.501**	**5243**	**0.454**	**0.506**	**0.407**	**0.293**

Table 2: Link prediction results on four datasets. **Bold** numbers are the best results, and the second best is underlined.

Evaluation Metrics. Take the head entity prediction for an instance, we fill the missing head entity with each entity e in the KG, and score a candidate triple (e, r, t) according to the following energy function together with the path information:

$$E_e(e, r, t, \mathcal{P}) = E_t(e, r, t) + \alpha_1 E_p(e, t, \mathcal{P}) \quad (20)$$

in which we reuse the energy functions in Eq. 5 and Eq. 6, and $\mathcal{P}$ is the path set consisting of all the paths between entities e and t. We rank the scores of the candidate triples in ascending order. Tail entity prediction is similar way.

We employ three frequently-used metrics: (1) Mean rank (MR) and (2) Mean reciprocal rank (MRR) of the triples containing the correct entities. (3) Hits@n is the proportion of the correct triples ranked in the top n. The lower MR, the higher MRR and the higher Hits@n declare the better performance. All the results are "filtered" by wiping out the candidate triples that are already in the KG (Wang et al., 2014).

4.2 Results of Link Prediction

The evaluation results of link prediction are reported in Table 2. **Firstly**, our model EngineKG significantly and consistently outperforms all the state-of-the-art baselines on all the datasets and all the metrics. Compared to best-performing models RotatE and RPJE on MR, EngineKG achieves performance gains of 95.0%/42.4%/3.7%/83.5% compared to RotatE and 100.0%/71.1%/38.8%/20.0% against RPJE on datasets FB15K/FB15K237/DBpedia-242/NELL-995, respectively. Particularly, on FB15K and FB15K237, the difference between the best performing baseline RPJE and our developed model is statistically significant under the paired at the 99% significance level. **Secondly**, our model achieves better performance than the traditional models that utilize triples alone, indicating that EngineKG is capable of taking advantage of extra knowledge including rules and paths as well as concepts, which all benefit to improving the performance of the whole model. **Thirdly**, EngineKG further beats IterE, illustrating the superiority of exploiting both rules and paths for KG inference in a joint logic and data-driven fashion.

4.3 Evaluation on Various Relation Properties

The relations can be classified into four categories: One-to-One (1-1), One-to-Many (1-N), Many-to-

FB15K	Head Entities Prediction				Tail Entities Prediction			
	1-1	1-N	N-1	N-N	1-1	1-N	N-1	N-N
TransE (Bordes et al., 2013)	0.356	0.626	0.172	0.375	0.349	0.146	0.683	0.413
RotatE (Sun et al., 2019)	0.895	0.966	0.602	0.893	0.881	0.613	0.956	0.922
HAKE (Sun et al., 2019)	0.926	0.962	0.174	0.289	0.920	0.682	**0.965**	0.805
DualE (Cao et al., 2021)	0.912	0.967	0.557	0.901	0.915	0.662	0.954	0.926
MultiHopKG (Lin et al., 2018)	-	-	-	-	0.893	0.576	0.921	0.763
RPJE (Niu et al., 2020)	0.942	0.965	0.704	0.916	**0.941**	0.839	0.953	0.933
EngineKG	**0.943**	**0.968**	**0.835**	**0.929**	**0.941**	**0.904**	0.957	**0.949**
FB15K237	Head Entities Prediction				Tail Entities Prediction			
	1-1	1-N	N-1	N-N	1-1	1-N	N-1	N-N
TransE (Bordes et al., 2013)	0.356	0.626	0.172	0.375	0.349	0.146	0.683	0.413
RotatE (Sun et al., 2019)	0.547	0.672	0.186	0.474	0.578	0.140	0.876	0.609
HAKE(Zhang et al., 2020)	0.791	**0.833**	0.098	0.237	0.794	0.372	**0.938**	**0.803**
DualE(Cao et al., 2021)	0.516	0.637	0.153	0.471	0.526	0.135	0.860	0.607
MultiHopKG(Lin et al., 2018)	-	-	-	-	0.417	0.026	0.794	0.457
RPJE(Niu et al., 2020)	0.692	0.476	0.180	0.575	0.669	0.197	0.691	0.685
EngineKG	**0.792**	0.743	**0.629**	**0.651**	**0.807**	**0.399**	0.881	0.757

Table 3: Link prediction results on FB15K and FB15K237 on various relation properties (Hits@10). MultiHopKG could only predict tail entities rather than head entities.

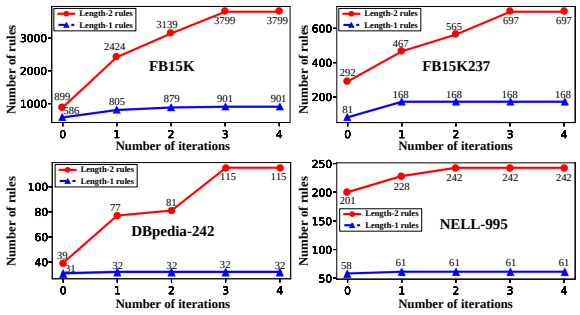

Figure 3: The number of rules over iterations.

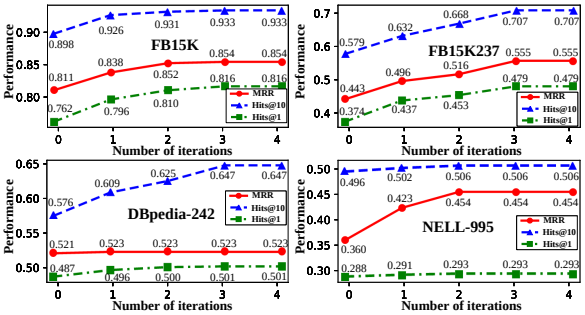

Figure 4: The performance curves of MRR, Hits@10 and Hits@1 over iterations on four datasets.

One (N-1), and Many-to-Many (N-N). We select some well-performing models observed in Table 2 as the baselines in this section. Table 3 exhibits that EngineKG achieves more performance gains on complex relations compared to other baselines. More interestingly, specific to the most challenging tasks (highlighted) namely predicting head entities on N-1 relations and tail entities on 1-N relations, our model consistently and significantly outperforms the outstanding baselines RotatE and RPJE by achieving performance improvements: 38.7%/47.5% on FB15K and 238.2%/185.0% on FB15K237 compared to HAKE while 18.6%/7.75% on FB15K and 249.4%/102.5% on FB15K237 compared to RPJE. These results all demonstrate that the paths and the generated rules enrich the associations between entities and relations, contributing to better performance of KG inference on complex relations.

4.4 Performance Evaluation Over Iterations

We evaluate the performance of our rule learning module on the learning time and the number of rules compared to the excellent rule learning tool AMIE+. For generating high-quality rules, our model takes 6.29s/2.26s/1.55s/10.50s in an iteration on average while AMIE+ takes 79.19s/26.83s/5.35s/105.53s on datasets FB15K/FB15K237/NELL-995/DBpedia, illustrating the higher efficiency of EngineKG. In Figure 3, the amount of rules mined by AMIE+ is shown as that at the initial iteration. Thus, we can discover that the quantity of rules generated in the first iteration and the third iteration is twice and three times the number of rules obtained by AMIE+.

More specifically, Figure 3 exhibits the number of rules and Figure 4 indicates the performance curves on the four datasets over iterations. Notably,

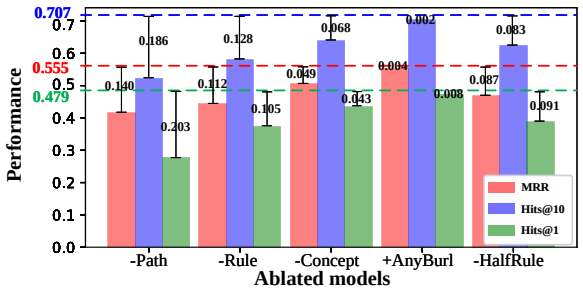

Figure 5: Ablation study on FB15K237. The dash lines indicate the performance of our whole model on MRR (red), Hits@10 (blue) and Hits@1 (green).

the number of rules and the performance continue to grow as the iteration goes on and they both converge after three iterations on all the datasets. These results illustrate that: (1) Rule learning and KGE modules in our model indeed complement each other and benefit in not only producing more high-quality rules but also obtaining better inference results. (2) More rules are beneficial to improving the performance of KG inference. (3) The iteration process will gradually converge along with the rule learning.

4.5 Ablation Study

To evaluate each contribution in our whole model EngineKG, we observe the performance on FB15K237 as to the five different ablated settings: (1) Omitting paths (-Path). (2) Omitting rules (-Rule). (3) Omitting concepts (-Concept) by removing concept-based co-occurrences in rule learning. (4) Replacing rule mining tool AMIE+ with AnyBurl (Meilicke et al., 2019) for obtaining the seed rules (+AnyBurl). (5) Employing only half of the seed rules without iteration (-HalfRule). Figure 5 shows that the performance of our whole model is better than that of all the ablated models except for "+AnyBurl", demonstrating that all the components in our designed model are valid and our model is free of any rule mining tool for obtaining the seed rules. Besides, removing paths and rules both have significant impacts on the performance, which suggests the paths and rules in our model play a more vital role in KG inference.

4.6 Case Study

As shown in Figure 6, although the head entity $Jonathan$ and the candidate tail entity $York$ are not linked by any direct relation in the KG, there is an explicit path between them. This path can be represented as the relation $PersonBornInCity$

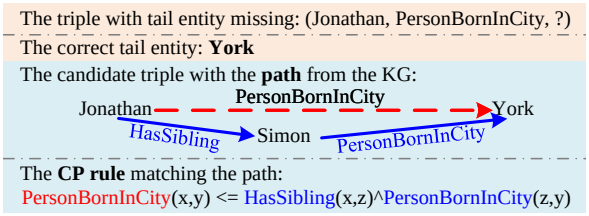

Figure 6: An example of the interpretable tail entity prediction via path and rule on NELL-995.

deduced by a matched CP rule. The standard confidence of the rule shown in Figure 6 is 0.9, which explains that although one's sibling being born in a certain city does not necessarily mean that that person was born in the same city, we can employ this trustworthy rule to explain the reliability of the predicted result obtained by our model. The path and the rule together boost the score of the correct candidate entity $York$ calculated by Eq. 20, and especially provide the interpretability of the result.

5 Conclusion and Future Work

In this paper, we develop a novel closed-loop neural-symbolic learning framework EngineKG for KG inference by jointly rule learning and KGE while exploiting paths and concepts. In the KGE module, both rules and paths are introduced to enhance the semantic associations and interpretability for learning the entity and relation embeddings. In the rule learning module, paths and KG embeddings together with entity concepts are leveraged in the designed rule pruning strategy to generate high-quality rules efficiently and effectively. Extensive experimental results on four datasets illustrate the superiority and effectiveness of our approach compared to some state-of-the-art baselines. In the future, we will investigate combining other semantics such as contextual descriptions of entities, and attempt to apply our model to dynamic KGs.

Acknowledgements

This work was partially supported by Zhejiang Science and Technology Plan Project (No. 2022C01082), the National Natural Science Foundation of China (No. 62072022, 61772054) and the Fundamental Research Funds for the Central Universities.

References

Kurt Bollacker, Georg Gottlob, and Sergio Flesca. 2008. Freebase: a collaboratively created graph database

for structuring human knowledge. In *KDD*, pages 1247–1250.

Antoine Bordes, Nicolas Usunier, Alberto Garcia-Duran, Jason Weston, and Oksana Yakhnenko. 2013. Translating embeddings for modeling multi-relational data. In *NIPS*, pages 2787–2795.

Z. Cao, Q. Xu, Z. Yang, X. Cao, and Q. Huang. 2021. Dual quaternion knowledge graph embeddings. In *AAAI*, pages 6894–6902.

Luis Galárraga, Christina Teflioudi, Katja Hose, and Fabian Suchanek. 2015. Fast rule mining in ontological knowledge bases with amie+. *The VLDB Journal*, 24(6):707–730.

Shu Guo, Quan Wang, Lihong Wang, Bin Wang, and Li Guo. 2018. Knowledge graph embedding with iterative guidance from soft rules. In *AAAI*, pages 4816–4823.

Vinh Thinh Ho, Daria Stepanova, Mohamed H. Gad-Elrab, Evgeny Kharlamov, and Gerhard Weikum. 2018. Rule learning from knowledge graphs guided by embedding models. In *ISWC 2018*, pages 72–90.

Xiao Huang, Jingyuan Zhang, Dingcheng Li, and Ping Li. 2019. Knowledge graph embedding based question answering. In *WSDM*, pages 105–113.

Ni Lao, Tom Mitchell, and William W. Cohen. 2011. Random walk inference and learning in a large scale knowledge base. In *EMNLP 2011*, pages 529–539.

Jens Lehmann, Robert Isele, Max Jakob, Anja Jentzsch, Dimitris Kontokostas, Pablo N. Mendes, Sebastian Hellmann, Mohamed Morsey, Patrick van Kleef, S¨oren Auer, and Christian Bizer. 2015. Dbpedia-a large-scale, multilingual knowledge base extracted from wikipedia. *Semantic Web*, 6(2):167–195.

Xi Victoria Lin, Richard Socher, and Caiming Xiong. 2018. Multi-hop knowledge graph reasoning with reward shaping. In *EMNLP*, pages 3243–3253.

Yankai Lin, Zhiyuan Liu, Huanbo Luan, Maosong Sun, Siwei Rao, and Song Liu. 2015. Modeling relation paths for representation learning of knowledge bases. In *EMNLP*, pages 705–714.

Christian Meilicke, Melisachew Wudage Chekol, Daniel Ruffinelli, and Heiner Stuckenschmidt. 2019. Anytime bottom-up rule learning for knowledge graph completion. In *IJCAI*, pages 3137–3143.

Arvind Neelakantan, Benjamin Roth, and Andrew McCallum. 2015. Compositional vector space models for knowledge base completion. In *ACL*, pages 156–166.

Guanglin Niu, Yongfei Zhang, Bo Li, Peng Cui, Si Liu, Jingyang Li, and Xiaowei Zhang. 2020. Rule-guided compositional representation learning on knowledge graphs. In *AAAI*, pages 2950–2958.

P. G. Omran, K. Wang, and Z. Wang. 2019. An embedding-based approach to rule learning in knowledge graphs. In *IEEE Transactions on Knowledge and Data Engineering*, pages 1–12.

Meng Qu, Junkun Chen, Louis-Pascal Xhonneux, Yoshua Bengio, and Jian Tang. 2021. Rnnlogic: Learning logic rules for reasoning on knowledge graphs. In *ICRL*.

Ali Sadeghian, Mohammadreza Armandpour, Patrick Ding, and Daisy Zhe Wang. 2019. Drum: End-to-end differentiable rule mining on knowledge graphs. In *NIPS*.

Zhiqing Sun, Zhi-Hong Deng, Jian-Yun Nie, and Jian Tang. 2019. RotatE: Knowledge graph embedding by relational rotation in complex space. In *ICLR*.

Kristina Toutanova and Danqi Chen. 2015. Observed versus latent features for knowledge base and text inference. In *Proceedings of the 3rd Workshop on Continuous Vector Space Models and their Compositionality*, pages 57–66.

Théo Trouillon, Johannes Welbl, Sebastian Riedel, Éric Gaussier, and Guillaume Bouchard. 2016. Complex embeddings for simple link prediction. In *ICML*, page 2071–2080.

Zhen Wang, Jianwen Zhang, Jianlin Feng, and Zheng Chen. 2014. Knowledge graph embedding by translating on hyperplanes. In *AAAI*, page 1112–1119.

Wenhan Xiong, Thien Hoang, , and William Yang Wang. 2017. DeepPath: A reinforcement learning method for knowledge graph reasoning. In *EMNLP*, pages 564–573.

Shuai Zhang, Yi Tay, Lina Yao, and Qi Liu. 2019a. Quaternion knowledge graph embeddings. In *NeurIPS*, pages 2731–2741.

Wen Zhang, Bibek Paudel, Liang Wang, Jiaoyan Chen, Hai Zhu, Wei Zhang, Abraham Bernstein, and Huajun Chen. 2019b. Iteratively learning embeddings and rules for knowledge graph reasoning. In *WWW*, pages 2366–2377.

Zhanqiu Zhang, Jianyu Cai, Yongdong Zhang, and Jie Wang. 2020. Learning hierarchy-aware knowledge graph embeddings for link prediction. In *AAAI*, pages 3065–3072.

Hao Zhou, Tom Young, Minlie Huang, Haizhou Zhao, and Xiaoyan Zhu. 2018. Commonsense knowledge aware conversation generation with graph attention. In *IJCAI*, pages 4623–4629.

Table-based fact verification with self-labeled keypoint alignment

Guangzhen Zhao[†], **Peng Yang**[†*]
† School of Computer Science and Engineering, Key Laboratory of Computer Network and Information Integration, Ministry of Education, Southeast University, China
{zhaogz, pengyang}@seu.edu.cn

Abstract

Table-based fact verification aims to verify whether a statement sentence is trusted or fake. Most existing methods rely on graph feature or data augmentation but fail to investigate evidence correlation between the statement and table effectively. In this paper, we propose a self-**L**abeled **K**eypoint **A**lignment model, named **LKA**, to explore the correlation between the two. Specifically, a dual-view alignment module based on the statement and table views is designed to discriminate the salient words through multiple interactions, where one regular and one adversarial alignment network cooperatively character the alignment discrepancy. Considering the interaction characteristic inherent in the alignment module, we introduce a novel mixture-of-experts block to elaborately integrate the interacted information for supporting the alignment and final classification. Furthermore, a contrastive learning loss is utilized to learn the precise representation of the structure-involved words, encouraging the words closer to words with the same table attribute and farther from the words with the unrelated attribute. Experimental results on three widely-studied datasets show that our model can outperform the state-of-the-art baselines and capture interpretable evidence words.

1 Introduction

Table-based fact verification aims to uncover the factuality attribute of the sentence relying on the available structured (Chen et al., 2020b; Wang et al., 2021b; Gupta et al., 2020) textual evidence. Current methods can be divided into two groups. The first one exploits the logical form of the statement with graph neural networks (Zhong et al., 2020; Shi et al., 2020, 2021). The other focuses on extending table-aware pre-trained language models (PLMs) (Eisenschlos et al., 2020; Herzig et al., 2020).

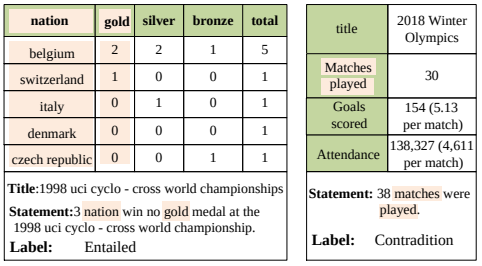

nation	gold	silver	bronze	total
belgium	2	2	1	5
switzerland	1	0	0	1
italy	0	1	0	1
denmark	0	0	0	1
czech republic	0	0	1	1

Title:1998 uci cyclo - cross world championships
Statement:3 nation win no gold medal at the 1998 uci cyclo - cross world championship.
Label: Entailed

(a) one example in TABFACT

title	2018 Winter Olympics
Matches played	30
Goals scored	154 (5.13 per match)
Attendance	138,327 (4,611 per match)

Statement: 38 matches were played.
Label: Contradition

(b) one example in INFOTABS

Figure 1: Two examples of table-based fact verification task. Keypoints are highlighted in yellow. (a) is a relational web table. (b) is an entity web table.

In PLM-oriented fact verification, the majority of methods treat the statement-table pair as plain text and then further capture latent essential information relying on multiple Transformer (Vaswani et al., 2017) blocks. It is intuitive that only partial table cells are associated with the statement while other cells are redundant (Wang et al., 2021a; Yin et al., 2020). Due to the lack of explicit guidance signals in the statement-table pair, the capability of checking various statements is hindered for the PLMs, deteriorating model performance and interpretability. Taking Figure 1 as an example, the clues are derived from the statement and some scattered table cells. If the model pays attention to the unrelated words (e.g., "sliver", "bronze", "total"), the prediction would not be able to convincingly correct. In other words, failing to align the latent salient words, which are denoted as **keypoints**, may lead to some misleading information being focused on as evidence. Despite impressive process, we empirically find that few methods are committed to the keypoint alignment across the statement words and table cells. The main reasons are: 1) There are alignment discrepancies in the alignment space, where one statement is associated with one table. However, one table may be involved with several statements; 2) The essence of alignment is to perceive salient evidence for the final classifi-

*Corresponding author.

Proceedings of the 29th International Conference on Computational Linguistics, pages 1401–1411
October 12–17, 2022.

cation, which requires a well-designed interaction network to aggregate the statement words and table cells; 3) Flexible table structures hinder the representation of words since the significant cost of designing a general structure-aware PLM for table-based fact verification task. In summary, exploring keypoint alignment feature in the statement-table pair is a major challenge.

To tackle the above deficiencies, we propose a model called self-**L**abeled **K**eypoint **A**lignment (**LKA**) for table-based fact verification, focusing on aligning salient evidence and aggregating essential information between the statement and table. Specifically, we design a Dual-view Alignment module (DA) for dealing with the discrepancy of the alignment characteristics. An interaction network is first applied for aggregating the interacted statement and the table representation in multiple steps. The DA then employs a regular alignment network to learn keypoint correlation from the enhanced statement view and force another adversarial alignment network to perceive the correspondingly reverse correlation (i.e., unrelated words) from the table view. For providing aggregated information in the interaction network, we design an Adaptive Aggregation Experts (AAE) block. The AAE employs a mixture-of-experts (MoE) network (Jacobs et al., 1991) that incorporates multiple operating units to sufficiently aggregate the statement and table information. Besides, inspired by the contrastive learning theory (Chen et al., 2020a; Pan et al., 2021), we adopt a structure-aware contrastive learning loss to obtain precise representation for the structure-involved words. The amended representation can force the statement and table closer to its local sub-structure zone (e.g., statement, row, column, etc.) and farther away from others. Our contributions are summarized in three folds:

- We explore a table-based fact verification model integrating keypoint alignment from the statement and table views, which can convert the alignment task into the optimization of two opposite goals and effectively integrate essential information with the MoE network.
- The contrastive learning theory is introduced to enhance structure-aware word representation, which provides a simple and general way to address various structured tabular data.
- We conduct experiments on three benchmark datasets TABFACT, INFOTABS and SEM-TAB-FACTS. Experimental results demonstrate that our model bring performance gains by 0.67%/3.63%/3.07% compared with several state-of-the-art models, and the captured salient words can be interpreted.

2 Related Work

Unlike FEVER (Thorne et al., 2018) utilizing textual evidence or FEVEROUS (Rami Aly and Mittal, 2021) using textual-table mixed evidence, table-based fact verification (Chen et al., 2020b) concentrates on structured or semi-structured evidence text. The currently popular methods employ a tree-style neural network (Zhong et al., 2020) or graph network (Yang et al., 2020; Shi et al., 2020) to encode the logical form of statements. However, labeling massive accurate logical forms is labor-intensive. Meanwhile, structure-aware models (Eisenschlos et al., 2020; Zhang et al., 2020; Dong and Smith, 2021) have been investigated to deal with the table-based fact verification task. Among these approaches, TaPaS++ (Eisenschlos et al., 2020) projects structural information of tables to a pre-trained language model by importing row, column, numeric features into the embedding layer. Some researchers also design novel data augmentation strategies to enhance TaPaS++, such as decomposing complex statements (Yang and Zhu, 2021), replacing non-salient tokens (Wang et al., 2021a), or generating massive question-answer pairs (Liu et al., 2021). However, upgrading vanilla PLMs for precise representation requires a considerable expense of pre-training or data augmenting.

In addition to the rational structured table data, there are entity tables (Gupta et al., 2020) and matrix PDF tables (Wang et al., 2021b) in table-based fact verification scenarios. Mainstream approaches (Gautam et al., 2021; Müller et al., 2021) employ TaPaS++ and TABFACT data to check matrix PDF tables while various scanning (Gupta et al., 2020) and filtering (Neeraja et al., 2021) methods are proposed to deal with entity tables. Different from the aforementioned works, our model exploits a novel alignment of salient words for the statement-table pair and a structure-oriented loss for precise representation.

3 Methodology

3.1 Formulation

In the table-based fact verification task, there are a statement S, its corresponding structured or semi-

structured evidence table T and label Y. All samples are denoted as $\boldsymbol{\mathcal{D}} = \{(\boldsymbol{S}_i, \boldsymbol{T}_i, \boldsymbol{Y}_i)|0 \leq i < I\}$, $I = |\boldsymbol{\mathcal{D}}|$. The task can be formalized as searching for the best mapping $y^* = f_\theta(S, T)$ to minimize the error:

$$Err_{\mathcal{D}} = \boldsymbol{E}_{(s_i,t_i,y_i)\sim\mathcal{D}}\mathcal{L}(f_\theta(s_i, t_i), y_i), \quad (1)$$

where y^* is the predicted label, y is the ground-truth label, f_θ is a specified model in the hypothesis space $\boldsymbol{F}$ with parameter θ.

Furthermore, as shown in Figure 1, it can be observed that only partial cells in table T are relevant to the statement S, which means these salient words are keypoints. The alignment problem is defined as how to identify these keypoints in the statement-table pair. Formally, given an example $e \in \mathcal{D}$ with m labeled tokens $e = \{(x_j, y_j)\}_{j=1}^m$ from $\mathcal{X} \times \mathcal{Y}$, where $\mathcal{X}$ denotes an input space and $\mathcal{Y}$ denotes an output space, $y_j \in \{0, 1\}$ indicates token x_j should be aligned or not. We consider the binary classification as the *alignment classifier* $f_a : \mathcal{X} \rightarrow \mathbb{R}^{|y|}$. The accuracy of the alignment classifier is given by:

$$Acc(f_a) = E_{(x_j,y_j)\in e}\mathbb{1}(f_a(x_j) = y_j), \quad (2)$$

where $\mathbb{1}(.)$ is the indicator function. Then we can define the alignment distance from statement x^s and table x^t as:

$$\begin{aligned} &d_{f_a,f_a'\in\boldsymbol{F}}(S, T) = Acc_S(f_a) + Acc_T(f_a') \\ &= \frac{1}{m}\sum_{j}^{m}\mathbb{1}(f_a(x_j^s) = y_j^s) + \mathbb{1}(f_a'(x_j^t)! = y_j^t)' \end{aligned} \quad (3)$$

where f_a' is an adversarial alignment function that discriminates the unrelated tokens. Thus the total objective can be defined as:

$$\min_{f_\theta\in\boldsymbol{F}} Err_{\mathcal{D}} + \max_{f_a,f_a'\in\boldsymbol{F}} d(S, T). \quad (4)$$

In this manner, the proposed model learns to minimize the error performance and maximize the alignment distance jointly.

3.2 Model Overview

The architecture of LKA is shown in Figure 2. It consists of an encoder, a dual-view alignment module and an MoE-level interaction network. The encoder maps a statement-table pair x into a hidden representation with a vanilla PLM. Inspired by the way humans solve the table-based fact verification task, we design an alignment module to align the underlying keypoint from the statement and table, respectively. Meanwhile, an interaction network driven by MoE is designed to aggregate interactive information for further supporting keypoint alignment and final classification. Additionally, we utilize a contrastive loss on the PLM to yield a more precise structure feature.

3.3 Encoder

The statement and flattened table are formatted as $e = \{[\text{CLS}], state, [\text{SEP}], head, ..., r_w, [\text{SEP}]\}$, where $state$ indicates the statement, $head$ indicates the headline of the table, and r_w indicates the w-th row tokens. After encoding, we can obtain the overall statement-table pair representation $\boldsymbol{H} = \text{PLM}(e)$, including the statement representation $\boldsymbol{H}_s$ and the table $\boldsymbol{H}_t$, where $\boldsymbol{H} \in \mathbb{R}^{m\times d}$, $\boldsymbol{H}_s \in \mathbb{R}^{s\times d}$, $\boldsymbol{H}_t \in \mathbb{R}^{t\times d}$, d is the dimension of the hidden representation, m, s, and t are the length of the statement-table pair, the statement and the table, respectively.

3.4 Dual-view Alignment Module

As keypoints are derived from the interaction of the statement and the table, an interaction network is designed to explore the correlation between the two representations. We alternate attentive memory accesses to the statement and the table for multiple steps. From the statement view, we formulate a query glimpse $\boldsymbol{q}_s^\tau$ at step τ:

$$\boldsymbol{q}_s^\tau = \underset{j=1,...,s}{\text{softmax}}(\boldsymbol{H}_s^\tau\mathbf{W}_q^s \cdot (\boldsymbol{H}_{s_j}\mathbf{W}_k^s + b_j)^T)\boldsymbol{H}_s\mathbf{W}_v^s, \quad (5)$$

where $\mathbf{W}_q^s, \mathbf{W}_k^s, \mathbf{W}_v^s \in \mathbb{R}^{d\times d}$ are projection matrices, $\boldsymbol{H}_{s_j} \in \mathbb{R}^d$ is the j-th token vector in the statement, b_j is a bias term. $\boldsymbol{H}_s^\tau$ is initialized with $\boldsymbol{H}_s$ when $\tau = 0$.

After interacting with the statement, the alternative attention probes the target table. The table attention weights are calculated based on the table $\boldsymbol{H}_t$ and the currently selected query glimpse $\boldsymbol{q}_s^\tau$:

$$\boldsymbol{q}_t^\tau = \underset{j=1,...,t}{\text{softmax}}(\boldsymbol{q}_s^\tau\mathbf{W}_q^t \cdot (\boldsymbol{H}_{t_j}\mathbf{W}_k^t + b_j)^T)\boldsymbol{H}_t\mathbf{W}_v^t, \quad (6)$$

where $\mathbf{W}_q^t, \mathbf{W}_k^t, \mathbf{W}_v^t \in \mathbb{R}^{d\times d}$ are projection matrices, $\boldsymbol{H}_{t_j} \in \mathbb{R}^d$ is the j-th token vector in the table. The interaction network then updates the statement on the basis of the attentive information gathered from the current step τ, i.e., $\boldsymbol{H}_s^{\tau+1} = \psi([\boldsymbol{q}_s^\tau, \boldsymbol{q}_t^\tau])$, where $\psi(.)$ is a non-linear aggregation function.

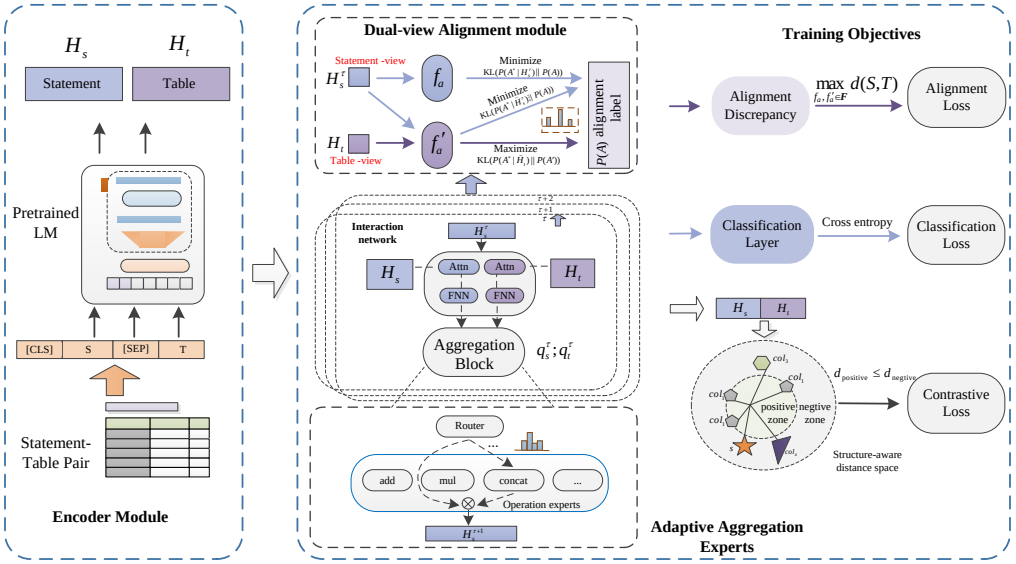

Figure 2: The framework of the proposed model LKA for table-based fact verification.

The updated statement representation aggregates the interacted information from the statement and table for multiple steps. By this means, the updated statement can benefit better alignment and the final verification.

The alignment module tries to make sense of the critical evidence cells so as to provide interpretable evidence for the verification. One noticeable difficulty is how to determine keypoints in the table since accurate labeling of these keypoints is labor-intensive. To this end, we use the same content (i.e., salient tokens) appearing in both the statement and the table as weak supervised keypoints. Given a statement-table pair example $e = \{t_0, t_1, ..., t_m\}$, the alignment label of each token is $A(t_i) \in \{0, 1\}$, where 0 means that token is not essential and vice versa. For token t in example e, the alignment module produces a likelihood probability distribution $\mathcal{A}(t)$ and thus predicts the corresponding alignment label. Since the keypoints are primarily determined by the statement, $\mathcal{A}(t)$ can be predicted with the guidance of the statement representation $\boldsymbol{H}_s^\tau$. The $\mathcal{A}(t)$ is implemented by:

$$\mathcal{A}(t) = \text{Sigmoid}(\text{MLP}((\boldsymbol{H}_s^\tau)) \in \mathbb{R}^{m\times 1}. \quad (7)$$

Then the predicted label $a(t)$ is gained by an alignment softmax function σ:

$$a(t) = \sigma(\mathcal{A}_o(t)) = \frac{\exp(\mathcal{A}_o(t))}{\sum_{o=0}^{o=1} \exp(\mathcal{A}_o(t))}, \quad (8)$$

where $\mathcal{A}_1(t) = 1 - \mathcal{A}_0(t)$. To alleviate the noise from the yielded label probability distribution $A(t_i)$, we add a tolerance item β on $A(t_i)$ with random sampling and revise the alignment probability distribution as $\tilde{A}(t)$. The formula is defined as below:

$$\tilde{A}(t) = \frac{A_o(t) + G(0, \beta)}{\sum_{o=0}^{o=1} A_o(t) + G(0, \beta)}, \quad (9)$$

where $A_o(t) \in \{0, 1\}$ is the original probability, G is the Gaussian sampling function that can revise $A(t)$'s 0-1 hard label to be more tolerant. Sequentially, we use the Kullback-Leibler (KL) divergence (Kullback and Leibler, 1951) to measure the difference between the predicted alignment probability $\mathcal{A}(t)$ and the ground truth alignment label $A(t)$:

$$L(A(t), \mathcal{A}(t)) = \frac{1}{m} \sum_{i=0}^{m} \text{KL}(\tilde{A}(t_i)||a(t_i)). \quad (10)$$

Moreover, the keypoints can be recognized from the table view by first scanning the table and then searching the relevant statement. Thus, the alignment module also could align the keypoints with the table representation. However, as shown in Figure 3, the statement is only related to one table, but one table has involved in different statements. If the alignment module directly aligns the different statements from the table perspective, the process of optimization becomes more difficult due to the multiple alignment decision bounds brought by these statements. Under this consideration, the alignment module concentrates on the non-salient tokens when using table representation. In other words, an adversarial network $\mathcal{A}'(t)$ is designed to make the misalignment with the table representation $\boldsymbol{H}_t$ and

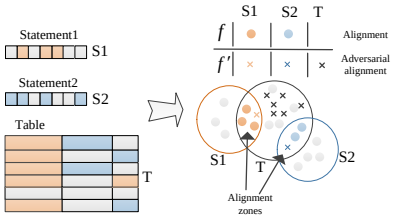

Figure 3: Alignment discrepancy between the statement and table. The non-salient tokens, alignment tokens of the S1, alignment tokens of S2 are highlighted in gray, yellow and blue.

predict correct alignment with the statement representation $\boldsymbol{H}_s^\tau$. The above parallel network working mode can be viewed as dual-view alignment. The details of the dual-view alignment can be clarified in Figure 3. Focusing on the alignment zones for the $\mathcal{A}^{'}(t)$ with statement representation is helpful to learn the potential bound of alignment and non-alignment. Specific ally, we first generate the false alignment label $A^{'}(t) = 1 - A(t)$ and then use the statement representation $\boldsymbol{H}_s^\tau$ to predict the self-labeled alignment probability $a_s^{'}(t) = \sigma(\mathcal{A}_s^{'}(t))$ by the adversarial network. Meanwhile, we use the table representation $\boldsymbol{H}_t$ to predict the false alignment probability $a_t^{'}(t) = \sigma(\mathcal{A}_t^{'}(t))$.

$$\mathcal{A}_s^{'}(t) = \text{Sigmoid}(\text{MLP-adv}(\boldsymbol{H}_s^\tau)), \tag{11}$$

$$\mathcal{A}_t^{'}(t) = \text{Sigmoid}(\text{MLP-adv}(\boldsymbol{H}_t)). \tag{12}$$

In short, the adversarial alignment can be summarized as:

$$L_{adv}(A^{'}(t), \mathcal{A}^{'}(t)) = \frac{1}{m}\sum_{i=0}^{m} \text{KL}(a_s(t_i)||a_s^{'}(t_i)) + \text{KL}(A^{'}(t_i)||a_t^{'}(t_i)). \tag{13}$$

3.5 Adaptive Aggregation Experts Module

In this subsection, we implement the aggregation function ψ inspired by the mixture-of-experts (MoE) (Shazeer et al., 2017) mechanism. The aggregation function consists of multiple parallel neural layers, which indicate different kinds of interactions for each attentive statement $\boldsymbol{q}_s^\tau$ and attentive table $\boldsymbol{q}_t^\tau$. The idea of mixture-of-experts is derived from a group of networks ("experts") that jointly make decisions with dynamical weights. Unlike previous approaches that treat each expert as a uniform structure unit (Shazeer et al., 2017; Fedus et al., 2021), we regard the experts as a series of operation units, which take the $(\boldsymbol{q}_s^\tau, \boldsymbol{q}_t^\tau)$ as input $\{\boldsymbol{x}_1, \boldsymbol{x}_2\}$ and effectively aggregate it in various manners.

$$E(\boldsymbol{x}_1, \boldsymbol{x}_2) = \{\boldsymbol{x}_1 \circ \boldsymbol{x}_2, \boldsymbol{x}_1 \otimes \boldsymbol{x}_2, \boldsymbol{x}_1 \oplus \boldsymbol{x}_2, \boldsymbol{x}_1 \ominus \boldsymbol{x}_2\}, \tag{14}$$

where $\circ, \otimes, \oplus, \ominus$ denote the concatenation, the element-wise multiplication, the element-wise addition and the element-wise subtraction operations, respectively. The MoE block routes the token pair $(\boldsymbol{x}_1, \boldsymbol{x}_2)$ to the determined expert from an expert set $\{E_i(\boldsymbol{x}_1, \boldsymbol{x}_2)\}_{i=1}^N$ by an MLP neural network. The output of the MLP $h(\boldsymbol{x}_1, \boldsymbol{x}_2)_j$ is normalized via a softmax function over the available N experts. The gate-value for expert i is given:

$$p_i(\boldsymbol{x}_1, \boldsymbol{x}_2) = \frac{e^{h(\boldsymbol{x}_1, \boldsymbol{x}_2)_j}}{\sum_j^N e^{h(\boldsymbol{x}_1, \boldsymbol{x}_2)_j}}. \tag{15}$$

Accordingly, the output of the MoE block is the linearly weighted combination of each selected expert's computation on each token by the gate value:

$$\boldsymbol{H}_s^{\tau+1} = \sum_{i \in N} p_i(\boldsymbol{q}_s^\tau, \boldsymbol{q}_t^\tau) E_i(\boldsymbol{q}_s^\tau, \boldsymbol{q}_t^\tau). \tag{16}$$

By this means, the updated statement representation $\boldsymbol{H}_s^{\tau+1}$ can aggregate the attentive information and flow into the next interaction step. Moreover, we average the final step $\boldsymbol{H}_s^\tau$ and concatenate it with the overall statement-table pair representation $\boldsymbol{H}_{[\text{CLS}]}$ to predict the label $y^* = \text{MLP}(\text{mean-pooling}(\boldsymbol{H}_s^\tau); \boldsymbol{H}_{[\text{CLS}]})$.

3.6 Training Objectives

Objective 1. To capture the alignment features, we minimize the KL-divergence from the statement and table views, respectively.

$$L_{align} = L(A(t), \mathcal{A}(t)) + L_{adv}(A^{'}(t), \mathcal{A}^{'}(t)) \tag{17}$$

Objective 2. Inspired by the contrastive learning theory (Chen et al., 2020a), we design a structure-aware loss, enabling the PLM to grasp the structural attributes of the statement-table pair.

The definitions of the positive correlation for these different tables are different. The reason is that the rational and matrix PDF table cells in the same column have a similar natural attribute. Recognizing the column relationship is helpful for table encoding (Yin et al., 2020; Chen et al., 2020b). For the entity table, a row is comprised of a property cell and its corresponding content. There is

no structural correlation among the rows. (Gupta et al., 2020) also confirms that modeling the link between the property and the content could provide a more accurate representation. Subsequently, the structure features can be learned by the objective of contrastive learning:

$$L_{cl} = \frac{1}{m}\sum_{i=1}^{m}\left[d\left(\boldsymbol{t}_i^a, \boldsymbol{t}_i^p\right) - d\left(\boldsymbol{t}_i^a, \boldsymbol{t}_i^n\right) + \xi\right]_+, \tag{18}$$

where m is the length of the example, a, p, n are the anchor, positive and negative token features, respectively. $d(\cdot,\cdot)$ is the distance function, ξ is a margin parameter, and $[x]_+$ is the $max(x, 0)$ function. In a nutshell, the contrastive loss L_{cl} helps to enhance the distance space's intra-structure compactness and inter-structure discreteness.

Objective 3. Finally, we use cross-entropy loss:

$$L_{ce} = \sum_{x,y\in D} -logP_\theta(y|x). \tag{19}$$

Total objective. The overall loss consists of the above three objectives with hyperparameters λ_1 and λ_2, as well as a balance loss L_{moe} (Shazeer et al., 2017) of adjusting the ratio of selected experts:

$$L_{total} = L_{ce} + \lambda_1 L_{align} + \lambda_2 L_{cl} + L_{moe}. \tag{20}$$

4 Experiments

4.1 Dataset and Metrics

To evaluate the validity of LKA, we adopt three standard datasets with various table structures[1]. For labels, each statement in TABFACT and SEM-TAB-FACTS is labeled as *entailed* or *refuted*[2], while INFOTABS divides statements into three kinds: *entailment*, *contradiction* and *neutral*. We leverage accuracy (Acc.) as the evaluation metric on TABFACT and INFOTABS, as well as microF1 score for SEM-TAB-FACTS.

4.2 Experimental Details

The computation environment is implemented with Python 3.6, PyTorch 1.8.0, CUDA 10.2 and cuDNN 8.0. Recall that all the experiments are running on a CentOS 7 server with the Intel(R) Xeon(R) Gold 6240 @ 2.60GHz CPU and one NVIDIA TESLA V100 GPU.

The optimizer is AdamW and the warmup rate is 0.06. Following traditional natural language understanding task GLUE[3], we fine-tune the DeBERTaV1[4] backbone for the DeBERTaV1 baseline and our LKA with the MultiNLI[5] corpus in one epoch before formal training. The hyperparameters are adjusted depending on the performance of the validation dataset. We set the word embedding and the hidden embedding size of the PLM to 1024. For TABFACT, we run five epochs with a batch size of 4, an initial learning rate of 1e-5, an attention head of 16 and each head of 64 in the attentive interaction network. Three epochs with a batch size of 8 and a learning rate of 1e-5 are adopted in INFOTABS and SEM-TAB-FACTS. In the three datasets, the Dropout is set to 0.1, the number of steps in the interaction network is 3. The tolerant item β, and the balanced factor λ_1, λ_2 are set to be 0.1, 0.08, 0.1, respectively. We set the $d(\cdot,\cdot)$ with Euclidean Distance and the margin parameter ξ with 0.1 in the contrastive learning loss. In the interaction network, we search the number of steps $\mathcal{T}$ in [1, 2, 3, 4, 5, 6]. According to the best results of these different parameters settings, we chose the $\mathcal{T}$=3.

4.3 Baseline Models

We compare our model LKA with the advanced baselines for TABFACT, i.e., **TaPaS++** (Eisenschlos et al., 2020), **Decomp.** (Yang and Zhu, 2021), **SalienL.** (Wang et al., 2021a), **TaPEx** (Liu et al., 2021). For INFOTABS, we employ the baselines **TabFact**, **TabAttn** proposed in (Gupta et al., 2020) and **KG_Info**(Neeraja et al., 2021) to estimate our model. We utilize the advanced baselines **Volta** (Gautam et al., 2021) and **TAPAS** (Müller et al., 2021) for SEM-TAB-FACTS.

4.4 Results and Analysis

Table 1 presents the results of various verification models on the TABFACT dataset. From Table 1, we can observe that our model LKA surpasses matrices from 0.16% to 1.50% compared to TaPEx, illustrating the boosted ability brought from the proposed alignment learning strategy. Moreover, our

[1]Dataset statistics are attached to Appendix.

[2]Since *neutral* examples are not given, we conduct the 2-way experiment for a fair comparison.

[3]GLUE: A Multi-Task Benchmark and Analysis Platform for Natural Language Understanding

[4]https://github.com/huggingface/transformers/blob/master/src/transformers/models/deberta

[5]https://cims.nyu.edu/~sbowman/multinli/

Models	Val	Test	Test simple	Test complex	Test small
TaPaS++	81.1	81.1	92.6	75.7	84.2
Decomp.	82.7	82.7	93.6	77.4	84.7
SalienL.	82.7	82.1	93.3	76.7	84.3
TaPEx(BART)	81.6	81.2	91.9	75.6	83.9
TaPEx	84.6	84.2	93.9	79.6	85.9
DeBERTaV1	83.28	83.26	92.53	79.15	85.14
Ours	**84.77**	**84.87**	**94.06**	**80.31**	**87.40**
Human	N/A	N/A	N/A	N/A	92.1

Table 1: Comparisons on the TABFACT (%)

Models	Val	Testα_1	Testα_2	Testα_3
TabAttn	63.63	62.94	49.37	49.04
TabFact	77.61	75.06	69.02	64.61
KG_Info	79.44	78.42	71.97	70.03
TaPaS++	74.94	73.22	61.83	60.88
TaPEx	77.38	76.50	67.55	66.38
DeBERTaV1	81.16	80.88	73.61	72.77
Ours	**82.66**	**82.05**	**74.94**	**73.55**
Human	79.78	84.04	83.88	79.33

Table 2: Comparisons on the INFOTABS (%)

Models	Val	Test
Volta	74.35	73.87
TAPAS	78.33	75.33
TaPEx	77.53	75.47
DeBERTaV1	79.12	75.94
Ours	**80.34**	**78.54**

Table 3: Comparisons on the SEM-TAB-FACTS(%)

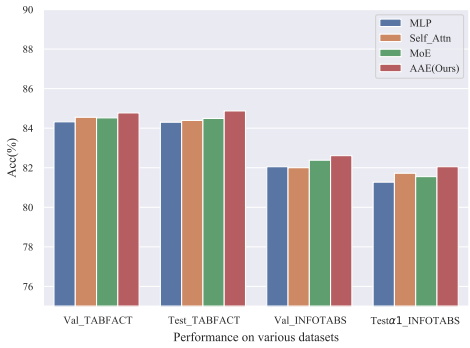

Figure 4: Different aggregation methods in the Interaction network. (%)

model reduces the gap between the machine and human performance to 4.7% on the small test dataset. Meanwhile, LKA achieves the best performance without complicated data augmentation compared with TaPEx. Since most approaches in TABFACT do not have results on INFOTABS and SEM-TAB-FACTS, we run the best approach TaPEx for comparison. As shown in Table 2, LKA outperforms the up-to-date baseline KG_Info from 2.97% to 3.63% on various evaluation subsets. Simultaneously, LKA improves the verification scores on the DeBERTaV1 backbone. Furthermore, we find that TaPEx and TaPaS++ do not perform as well on entity table data INFOTABS as they do on TABFACT. The reason is that the two models are designed to handle rational tables and they have difficulty in adapting to tabular data with varying structures. Considering that approaches on SEM-TAB-FACTS mainly use ensemble models for prediction, we only report the single-model performance in their paper to ensure evaluation fairness. LKA outperforms TaPEx by 3.07% on the test dataset.

In summary, LKA achieves the best results in the three scenarios, which indicates the prominent generalization ability of LKA. Besides, although data augmentation is important to boost performance, the results of DeBERTaV1 demonstrate that a stronger pre-trained language model has potential to tackle various table data and the structure-aware loss further enhances the advantage. Owing to the table structure similarity between TABFACT and SEM-TAB-FACTS, we then only conduct experiments and analyses on TABFACT and INFOTABS.

4.5 Effort of DA

We take a closer look at the dual-view alignment (DA) module by exploiting how the minimization (↓) and maximization (↑) of the KL-divergence between the prediction and the self-labeled alignment affect the final verification performance. As shown in Table 4, we adopt various alignment settings to investigate the alignment discrepancy. We can conclude that: 1) Comparison of the first three rows indicates the dual-view alignment networks are generally superior to the single ones, since the two views can provide complementary alignment information to be aware of salient words. 2) Performance of the third row is mostly lower than the last four rows, which demonstrates the alignment network f cannot resist the negative effect of alignment discrepancy as well as implies the rationality of the adversarial alignment network f'. 3) The last row represents the performance of the dual-view alignment module. The highest metrics indicate that using the adversarial network and table representation to align unimportant points, and using the adversarial network and statement representation to align important points are effective to alleviate alignment discrepancy.

Statement	Table	TABFACT		INFOTABS	
		Val	Test	Val	Testα_1
$\downarrow L_f$	–	84.23	84.31	81.83	81.22
–	$\downarrow L_f$	83.95	84.03	81.66	81.05
$\downarrow L_f$	$\downarrow L_f$	84.31	84.25	82.05	81.38
$\downarrow L_f + \downarrow L_{f'}$	$\downarrow L_{f'}$	84.45	84.21	**82.77**	81.22
$\downarrow L_f + \uparrow L_{f'}$	$\uparrow L_{f'}$	84.68	84.59	82.17	81.56
$\downarrow L_f + \uparrow L_{f'}$	$\downarrow L_{f'}$	84.62	84.65	82.21	81.23
$\downarrow L_f + \downarrow L_{f'}$	$\uparrow L_{f'}$	**84.77**	**84.87**	82.66	**82.05**

$\downarrow L_*$ means to align salient tokens, while $\uparrow L_*$ is to align non-important tokens, f and $f^{'}$ are the alignment network and the adversarial networks, respectively.

Table 4: Efforts of various settings under the alignment module(%)

Models	TABFACT		INFOTABS	
	Val	Test	Val	Testα_1
w/o Align-Inter	83.98	83.66	81.50	81.22
w/o Align	84.35	84.39	82.05	81.44
w/o Inter	84.56	84.32	82.11	81.22
w/o CL	84.65	84.46	82.38	81.72
LKA (Ours)	**84.77**	**84.87**	**82.66**	**82.05**

Table 5: Ablation analysis of LKA (%)

4.6 Effort of AAE

To further exhibit the superiority of the MoE-level aggregation module, we compare it with the following three aggregation methods: 1) **MLP** (Multilayer Perceptron) acts as a fuse block to concatenate the attentive representation from the statement and table; 2) **Self-Attn** (Self-Attention network) adopts the attentive statement representation as query, the attentive table as key and value for aggregation; 3) regular **MoE** employs multiple MLP layers to fuse the representations.

The comparison is illustrated in Figure 4. MoE performs better than MLP and Self-Attn, which demonstrates the advantage of aggregation decisions. In addition, the proposed AAE achieves the optimum performance on the overall metric among all methods. AAE exceeds MLP, Self-Attn and regular MoE about 0.57%/0.78%, 0.48%/0.27% and 0.38%/0.50% on the TABFACT/INFOTABS in terms of test dataset, respectively. One possible reason is that, unlike vanilla MoE where each "expert" employs the same MLP, AAE projects different meta-operation units into MoE. In other words, under the supervision of loss signals, the interacted statement and table learn to adaptively fuse information with fundamental operations, such as "addition","subtraction","multiplication", imitating the process of making decisions by humans.

4.7 Ablation

In order to evaluate the impact of each component of LKA, we ablate it into the following four variants: 1) **w/o Align-Inter** removes the regular and adversarial alignment networks, and the interaction network; 2) **w/o Align** deletes the regular and adversarial alignment networks; 3) **w/o Inter** removes the interaction network; 4) **w/o CL** prunes away the structure-aware contrastive learning loss.

As shown in Table 5, we can conclude that removing each component would decrease from 0.33% to 1.21% in Accuracy on the test dataset, which verifies the effectiveness of each component and the reasonable integrity of the LKA. 1) **w/o Align-Inter**: w/o Align-Inter reflects the lowest performance in all simplified variants, decreasing 1.21% and 0.83% on Test, respectively. The experiment results elaborate the validity of our LKA capturing the interactive information and the dual-view alignment. 2) **w/o Align**: w/o Align underperforms LKA, showing 0.48% and 0.61% degradation on Test, respectively. It elaborates the necessity of the LKA capturing the alignment information from the statement and the table views. 3) **w/o Inter**: removing the interaction network decreases 0.55% and 0.83% on Test compared to LKA. The reduction conveys the effectiveness of integrating the attentive representations from the statement and table. 4) **w/o CL**: When eliminating the structure-aware contrastive loss, there are 0.41% and 0.33% accuracy decrease on Test. It reveals that the introduced contrastive loss can improve performance by capturing structure information.

4.8 Case Study

To promote the understanding of LKA, we illustrate two random examples in Figure 5. The dual-view alignment module captures the highlighted words to interpret the evidence fragment. From Figure 5, it can be seen that the proposed alignment module is able to capture essential words with more informative semantics (i.e.,"Bruno Abakanowicz", "born", "Born", "France", "England" for S1, "Bruno Abakanowicz", "inventor" for S2). Although some underlying keypoints are ignored, LKA can gain available evidence fragments such as "France", "Lithuania" for S1 with the MoE-level interaction module. Furthermore, we project the output of PLM into a 2-D dimension vector with TSNE (Van der Maaten and Hinton, 2008)

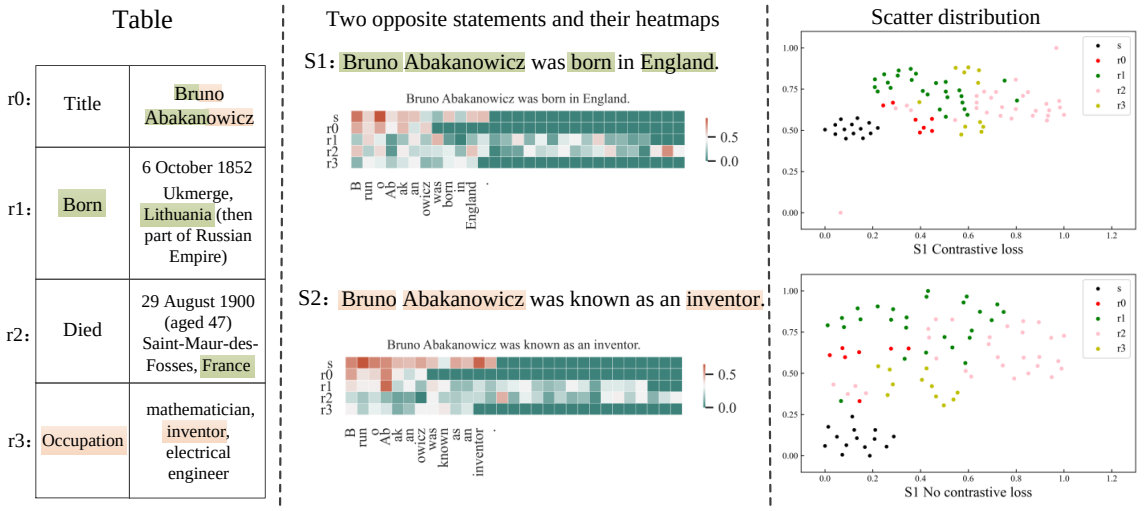

Figure 5: Case analysis via one contradiction and one entailment example on INFOTABS. Due to limited space, we only report the alignment weight heatmap from the statement view. Deeper red color means larger weight in alignment. The padding grids and tokens weighting close to zero are highlighted in green.

for the S1. We conclude that distributions of local sub-structure (e.g., statement, 0-th row, 1-th row) are more condensed than that of the setting with no contrastive loss. The condensed distribution verifies that the proposed LKA can perceive the structure feature of the statement-table pair.

5 Conclusion

This paper takes full advantage of alignment signals to facilitate a self-labeled learning procedure from the statement and table views. More importantly, an MoE-level aggregation module is designed to explore the valuable information. Besides, a contrastive learning loss is introduced to promote the awareness of table structure. Future research could be extended as follows: 1) Exploring alignment mechanism in the table-based question-answer tasks; 2) Developing fact verification approaches in the multi-evidence table setting.

Acknowledgements

The authors thank the anonymous reviewers for their valuable comments. We also thank Yu Yao, Xianyu Zha for their instructive suggestions and invaluable help. This work is supported in part by the National Natural Science Foundation of China under Grant 62272100, and in part by the Consulting Project of Chinese Academy of Engineering under Grant 2020-XY-5, and in part by the Fundamental Research Funds for the Central Universities and the Academy-Locality Cooperation Project of Chinese Academy of Engineering under Grant JS2021ZT05.

References

Ting Chen, Simon Kornblith, Mohammad Norouzi, and Geoffrey Hinton. 2020a. A simple framework for contrastive learning of visual representations. In *ICML'20*, volume 119, pages 1597–1607.

Wenhu Chen, Hongmin Wang, Jianshu Chen, Yunkai Zhang, Hong Wang, Shiyang Li, Xiyou Zhou, and William Yang Wang. 2020b. Tabfact: A large-scale dataset for table-based fact verification. In *ICLR'20, Addis Ababa, Ethiopia*.

Rui Dong and David Smith. 2021. Structural encoding and pre-training matter: Adapting BERT for table-based fact verification. In *EACL'21*, pages 2366–2375, Online.

Julian Eisenschlos, Syrine Krichene, and Thomas Müller. 2020. Understanding tables with intermediate pre-training. In *Findings of EMNLP'20*, pages 281–296, Online.

William Fedus, Barret Zoph, and Noam Shazeer. 2021. Switch transformers: Scaling to trillion parameter models with simple and efficient sparsity.

Devansh Gautam, Kshitij Gupta, and Manish Shrivastava. 2021. Volta at semeval-2021 task 9: Statement verification and evidence finding with tables using tapas and transfer learning. In *SemEval'21*.

Vivek Gupta, Maitrey Mehta, Pegah Nokhiz, and Vivek Srikumar. 2020. INFOTABS: Inference on tables as semi-structured data. In *ACL'20*, pages 2309–2324.

Jonathan Herzig, Pawel Krzysztof Nowak, Thomas Müller, Francesco Piccinno, and Julian Eisenschlos. 2020. TaPas: Weakly supervised table parsing via pre-training. In *ACL'20*, pages 4320–4333.

Robert A. Jacobs, Michael I. Jordan, Steven J. Nowlan, and Geoffrey E. Hinton. 1991. Adaptive mixtures of local experts. *Neural Computation*, 3(1):79–87.

Solomon Kullback and Richard A Leibler. 1951. On information and sufficiency. *The annals of mathematical statistics*, 22(1):79–86.

Qian Liu, Bei Chen, Jiaqi Guo, Zeqi Lin, and Jian-Guang Lou. 2021. TAPEX: table pre-training via learning a neural SQL executor.

Thomas Müller, Julian Martin Eisenschlos, and Syrine Krichene. 2021. Tapas at semeval-2021 task 9: Reasoning over tables with intermediate pre-training. In *SemEval'21*.

J. Neeraja, Vivek Gupta, and Vivek Srikumar. 2021. Incorporating external knowledge to enhance tabular reasoning. In *NAACL-HLT'21*, pages 2799–2809.

Xiao Pan, Mingxuan Wang, Liwei Wu, and Lei Li. 2021. Contrastive learning for many-to-many multilingual neural machine translation. In *ACL'21*, pages 244–258.

Michael Schlichtkrull James Thorne Andreas Vlachos Christos Christodoulopoulos Oana Cocarascu Rami Aly, Zhijiang Guo and Arpit Mittal. 2021. Feverous: Fact extraction and verification over unstructured and structured information.

Noam Shazeer, Azalia Mirhoseini, Krzysztof Maziarz, Andy Davis, Quoc V. Le, Geoffrey E. Hinton, and Jeff Dean. 2017. Outrageously large neural networks: The sparsely-gated mixture-of-experts layer. In *ICLR'17*, Toulon, France.

Qi Shi, Yu Zhang, Qingyu Yin, and Ting Liu. 2020. Learn to combine linguistic and symbolic information for table-based fact verification. In *COLING'20*, pages 5335–5346, Online.

Qi Shi, Yu Zhang, Qingyu Yin, and Ting Liu. 2021. Logic-level evidence retrieval and graph-based verification network for table-based fact verification. In *EMNLP'20*, pages 175–184, Punta Cana, Dominican Republic.

James Thorne, Andreas Vlachos, Christos Christodoulopoulos, and Arpit Mittal. 2018. FEVER: a large-scale dataset for fact extraction and VERification. In *NAACL-HLT'18*, pages 809–819, New Orleans, Louisiana.

Laurens Van der Maaten and Geoffrey Hinton. 2008. Visualizing data using t-sne. *Journal of machine learning research*, 9(11).

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Lukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *NeurIPS'17*, pages 5998–6008, Long Beach, California, USA.

Fei Wang, Kexuan Sun, Jay Pujara, Pedro Szekely, and Muhao Chen. 2021a. Table-based fact verification with salience-aware learning. In *EMNLP'21*, pages 4025–4036, Punta Cana, Dominican Republic.

Nancy XR Wang, Diwakar Mahajan, Marina Danilevsky, and Sara Rosenthal. 2021b. Semeval-2021 task 9: Fact verification and evidence finding for tabular data in scientific documents (sem-tab-facts).

Xiaoyu Yang, Feng Nie, Yufei Feng, Quan Liu, Zhigang Chen, and Xiaodan Zhu. 2020. Program enhanced fact verification with verbalization and graph attention network. In *EMNLP'20*, pages 7810–7825, Online.

Xiaoyu Yang and Xiaodan Zhu. 2021. Exploring decomposition for table-based fact verification. In *EMNLP'21*, pages 1045–1052, Punta Cana, Dominican Republic.

Pengcheng Yin, Graham Neubig, Wentau Yih, and Sebastian Riedel. 2020. TaBERT: Pretraining for joint understanding of textual and tabular data. In *ACL'20*, pages 8413–8426, Online.

Hongzhi Zhang, Yingyao Wang, Sirui Wang, Xuezhi Cao, Fuzheng Zhang, and Zhongyuan Wang. 2020. Table fact verification with structure-aware transformer. In *EMNLP'20*, pages 1624–1629, Online.

Wanjun Zhong, Duyu Tang, Zhangyin Feng, Nan Duan, Ming Zhou, Ming Gong, Linjun Shou, Daxin Jiang, Jiahai Wang, and Jian Yin. 2020. LogicalFactChecker: Leveraging logical operations for fact checking with graph module network. In *ACL'20*, pages 6053–6065, Online.

A Dataset Details

In this section, we describe more detailed settings about the experiments to aid in reproducibility. We also anonymously submit the source code and predicted results on the three datasets to the submission system.

Relational table dataset **TABFACT**[6] contains about 118K natural language statements accompanied by human-annotated 16K regular Wikipedia tables (similar to database tables) of evidence. In addition to the regular validation and test sets, TABFACT extracts subsets of Test_simple and

[6] https://github.com/wenhuchen/Table-Fact-Checking

Datasets			Splits		
TABFACT	Train	Val	Test	Simple	Complex
Statement	92,283	12,792	12,779	50,244	68,031
Table	13,182	1,696	1,695	9,189	7,392
INFOTABS	Train	Val	Testα_1	Testα_2	Testα_3
Statement	16,538	1,800	1,800	1,800	1,800
Table	1,740	200	200	200	200
SEM-TAB-FACTS	Train$_a$	Train$_m$	Val	Test	–
Statement	179,345	4,506	463	522	–
Table	1980	981	52	52	–

Table 6: Numbers of examples for all datasets.

Test_complex from the *Simple* and *Complex* channels, as shown in Table 6. **INFOTABS**[7] consists of almost 23K statements and 2.5K unique entity web table drawn from Wikipedia articles in various domains. The entity table could be viewed as a special table since it contains multiple rows and only two columns, of which one denotes the title of a record and the other is the corresponding content. **SEM-TAB-FACTS**[8] is proposed at SemEval-2021 task 9 and focus on matrix tables from scientific articles. The dataset contains an auto-generated train set *Train*$_a$ and a human-annotated train set *Train*$_m$. Note that we only use the *Train*$_m$ to train our LKA model since the *Train*$_a$ is more noisy.

The experimental data we used is taken from their links, and no additional processing is performed on the TABFACT and INFOTABS datasets beyond the steps described in Section 3.3. Considering that the matrix tables in SEM-TAB-FACTS contain multi-row or multi-column header, we follow the paper[9] to standardize the table header by dividing multi-row or multi-column header into multiple headers with the same content. In addition, conducting experiments with LKA, TaPEx[10] and DeBERTaV1 on the SEM-TAB-FACTS, the trained model on the TABFACT is utilized to initialize the training of SEM-TAB-FACTS.

B Algorithm Description

The algorithm description is given for further understanding and facilitating reproducibility of the proposed LKA model.

Algorithm 1 Table-based fact verification with self-labeled keypoint alignment

Require:
Source table, statement and ground-truth label (S, T, Y); model parameters $\boldsymbol{\theta}$; the alignment and the adversarial label A, $A^{'}$

Initialize model parameters $\boldsymbol{\theta}$
while not converged **do**
  Sample a training example (S, T, Y)
  Flatten (S, T) to $e = \{[\text{CLS}], state, [\text{SEP}], head, ..., r_w, [\text{SEP}]\}$
  $\boldsymbol{H} = \text{PLM}(e)$, $\boldsymbol{H}_s = \boldsymbol{H} * mask_s$, $\boldsymbol{H}_t = \boldsymbol{H} * mask_t$, let $\boldsymbol{H}_s^{\tau} = \boldsymbol{H}_s$
  for step $\tau = 0 \rightarrow \mathcal{T} - 1$ **do**
    $\boldsymbol{q}_s^{\tau} \leftarrow \underset{j=1,...,s}{\text{softmax}}(\boldsymbol{H}_s^{\tau}\mathbf{W}_q^s \cdot (\boldsymbol{H}_{s_j}\mathbf{W}_k^s + b_j)^T)\boldsymbol{H}_s\mathbf{W}_v^s$
    $\boldsymbol{q}_t^{\tau} \leftarrow \underset{j=1,...,t}{\text{softmax}}(\boldsymbol{q}_s^{\tau}\mathbf{W}_q^t \cdot (\boldsymbol{H}_{t_j}\mathbf{W}_k^t + b_j)^T)\boldsymbol{H}_t\mathbf{W}_v^t$
    $E(\boldsymbol{x}_1, \boldsymbol{x}_2) = \{\boldsymbol{x}_1 \circ \boldsymbol{x}_2, \boldsymbol{x}_1 \otimes \boldsymbol{x}_2, \boldsymbol{x}_1 \oplus \boldsymbol{x}_2, \boldsymbol{x}_1 \ominus \boldsymbol{x}_2\}$
    $\boldsymbol{H}_s^{\tau+1} = \sum_{i \in N} p_i(\boldsymbol{q}_s^{\tau}, \boldsymbol{q}_t^{\tau}) E_i(\boldsymbol{q}_s^{\tau}, \boldsymbol{q}_t^{\tau})$,
  end for
  Obtain the alignment $\mathcal{A}(t)$ and the adversarial alignment $\mathcal{A}^{'}(t)$ with $\boldsymbol{H}_s^{\tau}$ and $\boldsymbol{H}_t$
  $L_{align} = L(A(t), \mathcal{A}(t)) + L_{adv}(A^{'}(t), (\mathcal{A}^{'}(t))$
  $L_{cl} = \frac{1}{m}\sum_{i=1}^{m}[d(\boldsymbol{t}_i^a, \boldsymbol{t}_i^p) - d(\boldsymbol{t}_i^a, \boldsymbol{t}_i^n) + \xi]_+$
  $L_{ce} = \sum_{x,y \in D} -logP_{\theta}(y|x)$
  $L(\boldsymbol{\theta}) = L_{ce} + \lambda_1 L_{align} + \lambda_2 L_{cl} + L_{moe}$
  $\boldsymbol{\theta} \leftarrow \text{AdamW}(\nabla_{\boldsymbol{\theta}} L(\boldsymbol{\theta}), \boldsymbol{\theta})$
end while
return $\boldsymbol{\theta}$

[7] https://github.com/infotabs/infotabs
[8] https://sites.google.com/view/sem-tab-facts
[9] https://github.com/devanshg27/sem-tab-fact
[10] https://github.com/microsoft/Table-Pretraining

IMCI: Integrate Multi-view Contextual Information for Fact Extraction and Verification

Hao Wang[†], **Yangguang Li**[‡], **Zhen Huang**[†]*, **Yong Dou**[†]
[†] National University of Defense Technology, Changsha
[‡] SenseTime, Beijing
{hao.wang, huangzhen, yongdou}@nudt.edu.cn
liyangguang@sensetime.com

Abstract

With the rapid development of automatic fake news detection technology, fact extraction and verification (FEVER) has been attracting more attention. The task aims to extract the most related fact evidences from millions of open-domain Wikipedia documents and then verify the credibility of corresponding claims. Although several strong models have been proposed for the task and they have made great progress, we argue that they fail to utilize multi-view contextual information and thus cannot obtain better performance. In this paper, we propose to integrate multi-view contextual information (IMCI) for fact extraction and verification. For each evidence sentence, we define two kinds of context, i.e. **intra-document context** and **inter-document context**. Intra-document context consists of the document title and all the other sentences from the same document. Inter-document context consists of all other evidences which may come from different documents. Then we integrate the multi-view contextual information to encode the evidence sentences to handle the task. Our experimental results on FEVER 1.0 shared task show that our IMCI framework makes great progress on both fact extraction and verification, and achieves state-of-the-art performance with a winning FEVER score of 73.96% and label accuracy of 77.25% on the online blind test set. We also conduct ablation study to detect the impact of multi-view contextual information. Our codes will be released at https://github.com/phoenixsecularbird/IMCI.

1 Introduction

Fake news propagation is a severe social problem, which may cause great loss and lead to serious consequence, e.g. panic, quarrel, opposition and even war. The situation has become a general concern since Brexit and the U.S. President Campaign in 2016 and gets far more intense due to COVID-19 pandemic (Martino et al., 2020). In this condition, automatic fake news detection has been developing rapidly. According to Ruffo et al. (2021), automatic fake news detection mainly include textual-content based methods (Giachanou et al., 2019; Ghanem et al., 2020; Kaliyar et al., 2021), user-role based methods (Vo and Lee, 2019; Giachanou et al., 2020), multi-modal approaches (Zlatkova et al., 2019; Fung et al., 2021) and detection of bots and trolls (Stella et al., 2018; Sayyadiharikandeh et al., 2020). Among textual-content based methods, fact extraction and verification (FEVER) (Thorne et al., 2018) has been attracting developing attention. As shown in Figure 1, for a given claim, the task aims to select at most 5 most related sentences as evidences from millions of open-domain Wikipedia documents for fact extraction, and combine the selected evidences to judge the claim as SUPPORTS, REFUTES or NOT ENOUGH INFO (NEI) for fact verification.

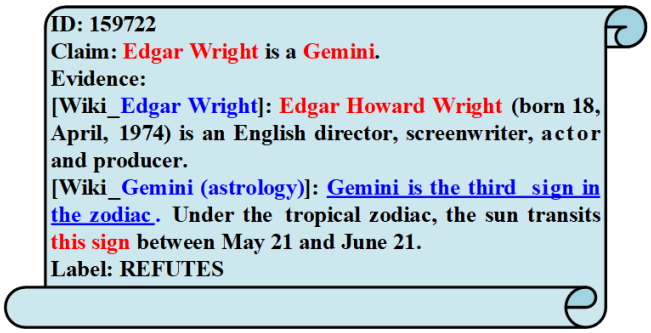

Figure 1: An example from FEVER 1.0 shared task. (**Underlined sentence** is the **unlabeled intra-document context**. Words in red involve alias name, coreference and multi-hop reasoning, which may lead to model confusion. Words in blue help to handle these issues.)

Recently, several strong models (Nie et al., 2019b,a; Zhou et al., 2019; Liu et al., 2020; Hidey et al., 2020; Subramanian and Lee, 2020) have

*Corresponding author

Proceedings of the 29th International Conference on Computational Linguistics, pages 1412–1421
October 12–17, 2022.

been proposed for the task. Although they have made great progress and obtained excellent performance on the task, we argue that they fail to utilize multi-view contextual information and thus cannot obtain better performance. Specifically, we define two kinds of context for each evidence sentence, i.e. **intra-document context** and **inter-document context**. Intra-document context consists of the document title and all the other sentences from the same document. Inter-document context consists of all other evidences which may come from different documents. Multi-view contextual information is of great importance for fact extraction and verification. For instance, as shown in Figure 1, intra-document context information can help to clarify the relationship between different entities, e.g. "*Edar Wright*" and its alias name "*Edar Howard Wright*" in the first evidence, and "*Gemini*" and it coreference "*this sign*" in the second sentence. Besides, the two evidence sentences can be regarded as inter-document context of each other, and the information interaction and fusion between them is essential to verify the claim in this multi-hop sample.

To this end, we propose to integrate multi-view contextual information (IMCI) for fact extraction and fact verification, where we introduce the multi-view contextual information to encode the evidence sentences to handle the task. In summary, our contributions are as follows:

• We propose an iterative multi-view fact extraction model. It retrieves related documents and extracts related evidence sentences in two iterations, with multi-view context information joined.

• We propose a multi-view fact verification model. Each evidence sentence is encoded from two views, and a dual evidence fusion graph is adopted to fuse the information from diverse views and different evidences.

• Our IMCI framework makes great progress on both fact extraction and verification, and achieves state-of-the-art performance with a winning FEVER score of 73.96% on the online blind test set.

2 Iterative Multi-view Fact Extraction

Our fact extraction model iteratively conducts document retrieval and sentence retrieval in two iterations to obtain corresponding candidate evidence sentences, and then reranks the candidates of different iterations for better performance.

2.1 Document Retrieval

Document retrieval includes *coarse document retrieval* in iteration 1 and *refined document retrieval* in iteration 2.

Coarse document retrieval aims to quickly obtain most related documents from millions of open-domain Wikipedia documents with as high as possible recall and acceptable precision. Inspired by UKP-Athene (Hanselowski et al., 2018) and SR-MRS (Nie et al., 2019b), coarse document retrieval is a combination of constituency-based Wikipedia search and TF-IDF retrieval. These two respectively utilize search engine power and statistical word frequency information. For constituency-based Wikipedia search, we also conduct mention filtering like UKP-Athene (Hanselowski et al., 2018). That is, if the title of a document is not explicitly mentioned in the claim, then we consider it as weakly related and remove it.

Refined document retrieval aims to retrieve documents with improved performance than coarse retrieval, namely higher recall and also higher precision and F1 score. It adopts dense semantic retrieval and utilizes Wikipedia hyperlinks. Specifically, in iteration 2, we decide refined candidate documents according to corresponding candidate evidences from iteration 1. That is, for each claim, all documents which contain at least one candidate evidence will be taken into account. Furthermore, as top one candidate evidences show pretty high precision (86.11%, in Table 5), we regard them as golden evidence, and take all documents which have hyperlinks with them as refined candidate documents to process multi-hop problem.

2.2 Sentence Retrieval

Sentence retrieval aims at selecting most related sentences as evidences from candidate documents. In previous models, during sentence retrieval, it is required to design sampling strategy to obtain negative samples for neural retrieval model training. Besides, these models respectively encode and score each claim-sentence pair.

Differently, in our framework, to avoid sampling strategy design and also utilize multi-view contextual information, we encode each sentence within its corresponding intra-document context. Moreover, as mentioned, top one candidate evidences of iteration 1 show pretty high precision (86.11%, in Table 5). Therefore, in iteration 2 we take them as inter-document context, and insert them into the

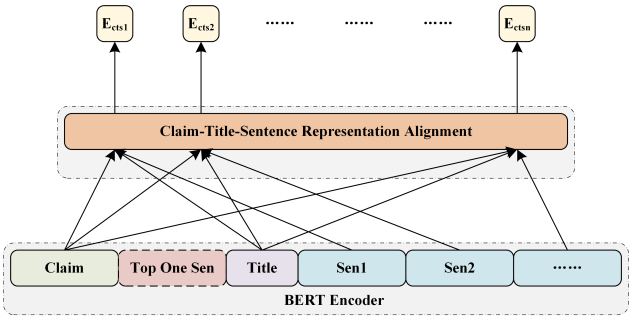

Figure 2: Sentence retrieval model. Sentences are encoded within intra-document context. In iteration 2, we insert top one candidate evidence (red dashed box) into the input sequence as inter-document context.

input sequence to process multi-hop problem.

Formally, as shown in Figure 2, the claim, the document title and all the sentences in the document are concatenated:

$$[CLS]\ claim\ [SEP]\ sen^*\ [SEP]\ title\ [SEP]\ sen1\ sen2\ \ldots\ [SEP] \quad (1)$$

where sen^* denotes top one candidate evidence of iteration 1, which are taken as inter-document context in iteration 2. The sequence is encoded by BERT encoder. For the claim, we take the hidden state of the first claim token as claim representation E_c. For the title, we take the hidden state of the first title token as title representation E_t. For each sentence, we take the hidden state of the first sentence token as the sentence representation E_s. The sentence representation is enhanced through alignment with the title representation:

$$E_{ts} = W_a[E_t, E_s, E_t - E_s, E_t \odot E_s] \quad (2)$$

and the claim representation:

$$E_{cts} = W_a^{'}[E_c, E_{ts}, E_c - E_{ts}, E_c \odot E_{ts}] \quad (3)$$

where $\odot$ means element-wise Hadamard product. Then, the score of sentence $\hat{y}$ is obtained through a Multi Layer Perceptron (MLP) with sigmoid activation function:

$$\hat{y} = \mathrm{Sigmoid}(\mathrm{MLP}(E_{cts})) \quad (4)$$

The training objective of sentence retrieval is defined as binary cross entropy loss, to maximize the probability of groundtruth evidence sentences:

$$\mathcal{L}_E = -\frac{1}{\sum_{i=1}^{m} n_i} \sum_{i=1}^{m} \sum_{j=1}^{n_i} [y_{ij} \cdot \log(\hat{y}_{ij}) + (1 - y_{ij}) \cdot \log(1 - \hat{y}_{ij})] \quad (5)$$

where m is the batch size, n_i is the sentence number of document i, and y is the sentence label, 1 for groundtruth evidence sentences while 0 for non-evidence sentences.

2.3 Full Pipeline

In each iteration, we have scored different sentences as candidate evidences. To obtain better performance, for each claim, we merge the results of different iterations, and rerank the sentences through their scores. Finally, according to the original setup of the task, we keep at most top 5 sentences as evidences, for further fact verification.

3 Multi-view Fact Verification

3.1 Multi-view Contextual Encoding

For each evidence sentence, we respectively obtain its representations through intra-document encoding and inter-document encoding.

• **Intra-document Encoding** aims to capture intra-document contextual information of each evidence sentence. It is similar to the sentence retrieval model in Section 2.2. Each evidence sentence is encoded within its intra-document context. Then its intra-document representation is also obtained through alignment.

• **Inter-document Encoding** is utilized to capture token-level information interaction among different evidence sentences to handle multi-hop problem. The claim, all evidence sentences and their document titles are concatenated as another input sequence:

$$[CLS]\ claim\ [SEP]\ title1\ [SEP]\ evi1\ [SEP]\ title2\ [SEP]\ evi2\ [SEP]\ \cdots\ [SEP] \quad (6)$$

The concatenation is also encoded by BERT encoder. Then, similarly, we obtain claim, title or evidence representation from the hidden state of the first token. Finally, for each evidence, we obtain its inter-document representation through alignment with the claim representation and its corresponding title representation.

3.2 Dual Evidence Fusion Graph

Through multi-view contextual encoding, for each evidence, we can obtain two alignment representations from different contextual views. To further integrate multi-view evidence information to handle multi-hop problem, inspired by multi-relational graph convolutional network (Cao et al., 2019; Tu

et al., 2019, 2020), we propose dual evidence fusion graph network. As shown in Figure 3, *one evidence sentence* corresponds to *two different nodes* in this graph, whose initial representations respectively come from intra-document encoding and inter-document encoding. For each evidence sentence, the noun phrases and named entities are extracted as keywords through spaCy[1] tool. Then for a pair of nodes, the links between them are decided according to following rules:

• **Common Document** Two nodes are linked if they come from the same document.

• **Common Keyword** Two nodes are linked if they share overlapped keywords.

• **Claim Jump** Two nodes are linked if they respectively share overlapped keywords with the claim.

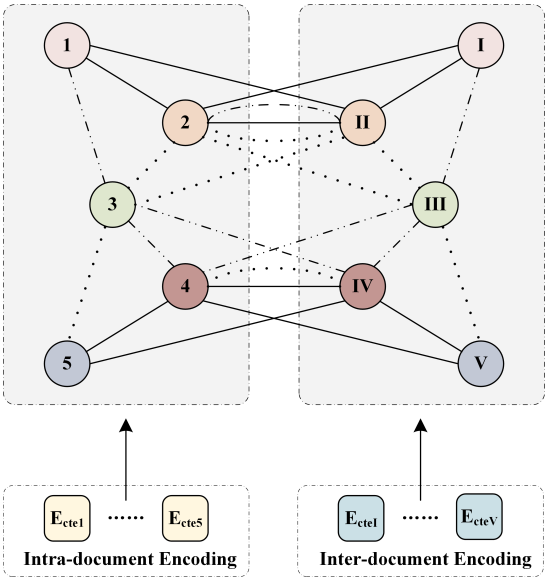

Figure 3: Dual Evidence Fusion Graph Network. Node 1 and node I denote different representations of the same evidence sentence from different encoding methods, similarly for node 2 and node II, etc. We define three kinds of edges in total.

For each claim, N selected evidence sentences introduce $2N$ evidence nodes. Let $H_i \in \mathcal{R}^{2N\times d}$ denotes the node representations at i-th graph layer, where d refers to the hidden dimension. The initial representation H_0 is the claim-title-evidence alignment representation through multi-view contextual encoding. Updated information $U_i \in \mathcal{R}^{2N\times d}$ after a single graph layer is defined as :

$$U_i = H_i W_0 + \sum_{j=1}^{3} \tilde{A}_j H_i W_j \tag{7}$$

[1]https://spacy.io/

where $\tilde{A}_j \in \mathcal{R}^{2N\times 2N}$ denotes corresponding row normalized adjacent matrix for different kinds of edges. Then the forget ratio $G_i \in \mathcal{R}^{2N\times d}$between the updated and old information is:

$$G_i = \text{Sigmoid}(W_g[U_i, H_i]) \tag{8}$$

And the updated evidence representation through the graph layer is:

$$H_{i+1} = \text{Activation}(U_i)\odot G_i + H_i \odot (1-G_i) \tag{9}$$

In this way, with several stacked layers, evidence representations are updated and multi-view evidence information is fused.

3.3 Confidence Aggregation

Aggregation aims to combines the evidence representations for final inference representation to verify the claim. Among the selected evidence sentences of a claim, some are groundtruth ones while others are not. To utilize evidence label information to enhance fact verification, like Tu et al. (2020), we adopt confidence aggregation.

Formally, let $H_k \in \mathcal{R}^{2N\times d}$ denotes evidence representations at the last graph layer. The confidence score of j-th evidence node $\hat{y}_j$ is obtained from its representation H_k^j:

$$\hat{y}_j = \text{Sigmoid}(\text{MLP}(H_k^j)) \tag{10}$$

The final inference representation for fact verification R_v is the weighted sum of the evidence representations, where the weights are corresponding confidence scores:

$$R_v = \sum_{j=1}^{2N} \hat{y}_j H_k^j \tag{11}$$

and the fact verification result is obtained through a 3-way classification network:

$$\hat{v} = \text{Softmax}(W R_v + b) \tag{12}$$

The total loss consists of the binary cross entropy loss of evidence confidence, and the cross entropy loss of 3-way fact verification:

$$\mathcal{L}_{II} = BCE(y, \hat{y}) + CE(v, \hat{v}) \tag{13}$$

Here y is the evidence sentence label, 1 for groundtruth evidence sentences and 0 for non-evidence sentences. Besides, v is the fact verification label.

4 Experiment

4.1 Dataset

We conduct our experiments on FEVER 1.0 shared task (Thorne et al., 2018), which consists of 185,455 annotated claims with 5,416,537 Wikipedia documents from the June 2017 dumps. We adopt the original dataset split of the task, which includes a training set, a development set and an online blind test set. The detailed information is shown in Table 1.

Split	SUPPORTS	REFUTES	NEI	Total
train	80035	29775	35639	145449
dev	6666	6666	6666	19998
test	6666	6666	6666	19998

Table 1: Statistics information of FEVER 1.0 Shared Task.

Moreover, for a claim, there exist several groups of evidences, and each group itself is enough to independently verify the claim. To further study the impact of multi-view contextual information, we conduct a refined split on the development set. Specifically, samples of the development set can be divided into 5 parts and the ratio of different parts are displayed in Table 2:

• **Single**. All evidence groups contain exactly one sentence.

• **Single+**. At least one evidence group contains only one sentence, and at least one group contains multi sentences.

• **Multi**. All evidence groups contain exactly two sentences.

• **Multi+**. All evidence groups contain multi sentences, and at least one group contains more than two sentences.

• **NEI**. The sample is labeled as NEI with no evidence groups annotated.

Single	Single+	Multi	Multi+	NEI
56.87	3.78	5.03	0.99	33.33

Table 2: Ratio of different parts on the development set.

4.2 Experiment Setup

Our IMCI is implemented through Pytorch 1.2.0 and our experiments are conducted on a computation node with 4 NVIDIA Titan V GPU. Pretrained BERT (Devlin et al., 2019) encoder is employed for all experiments. We also try RoBERTa encoder (Liu et al., 2019) for fact verification. For the claims, we set max length as 64, and claims longer than this will be truncated. For the encoders, we set max input sequence length as 512, and sequence longer than this will be split with stride window size of 128. We utilize BERTAdam optimizer with initial learning rate of 1e-5 and warmup ratio of 0.1. For sentence retrieval, we adopt mini batch size of 4 and gradient accumulation step of 8. In each iteration, we train 2 epochs and select top 5 sentences as candidate evidences. For fact verification, we adopt mini batch size of 1 and gradient accumulation step of 32. For dual evidence graph, we stack 3 graph layers, where the hidden dimension is the same as that of the encoder. In each condition, we randomly start 4 times, train 4 epochs, and choose model parameters with the best performance on the development set.

4.3 Evaluation Metric

We adopt FEVER score as the dominant evaluation metric, which is the officially chief metric. FEVER score requires that fact verification label is correctly predicted, and at least one complete group of evidence sentences is found for SUPPORTS and REFUTES samples. The second important metric is label accuracy. For document retrieval and sentence retrieval, we take precision, recall as well as F1 into account. Here, we attach more importance to recall according to the task setting.

5 Results

5.1 Main Results

Main results on the blind test set are shown in Table 3. With multi-view contextual information joined, our ICMI framework obtains FEVER score of 70.10% and label accuracy of 73.04% with BERT_{base} encoder. The performance is comparable and even slightly promoted compared with the state-of-the-art one among all baselines with BERT_{base} encoder. Moreover, our model with RoBERTa_{base} encoder obtains FEVER score of 72.97% and label accuracy of 75.84%, and shows even higher performance than several baselines with large encoder. Furthermore, our model with RoBERTa_{large} encoder obtains FEVER score of 73.96% and label accuracy of 77.25%, and significantly outperforms all baselines. These indicate that our framework has made great progress to conduct more accurate fact extraction and verification.

Model	LA	FEVER
UKP-Athene(2018)	65.46	61.58
QFE(2019)	69.30	61.80
NSMN(2019a)	68.16	64.23
GEAR-BERT$_{base}$(2019)	71.60	67.10
SR-MRS-BERT$_{base}$(2019b)	72.56	67.26
DeSePtion-BERT$_{base}$(2020)	72.47	68.80
Transformer-XH-BERT$_{base}$(2020)	72.39	69.07
KGAT-BERT$_{base}$(2020)	72.81	69.40
CorefBERT-BERT$_{base}$(2020)	72.88	69.82
HESM-ALBERT$_{base}$(2020)	**73.25**	70.06
HESM-BERT$_{base}$(2020)	73.18	70.07
ours IMCI-BERT$_{base}$	73.04	**70.10**
KGAT-BERT$_{large}$(2020)	73.61	70.24
CorefBERT-BERT$_{large}$(2020)	74.37	70.86
HESM-ALBERT$_{large}$(2020)	74.64	71.48
KGAT-RoBERTa$_{large}$(2020)	74.07	70.38
CorefBERT-RoBERTa$_{large}$(2020)	75.96	72.30
ours IMCI-RoBERTa$_{base}$	75.84	72.97
ours IMCI-RoBERTa$_{large}$	**77.25**	**73.96**

Table 3: Overall performance on the online blind test det. FEVER is the officially chief score. LA denotes label accuracy.

Model	P	R	F1
UKP-Athene(2018)	-	90.32	-
NSMN(2019a)	**51.04**	89.23	**64.94**
GEAR-BERT$_{base}$(2019)	-	89.99	-
SR-MRS-BERT$_{base}$(2019b)	18.11	92.03	30.27
Coarse Retrieval	8.68	**92.77**	15.87
Refined Retrieval	29.90	**95.69**	45.56

Table 4: Document retrieval results on the development set. - denotes that the item is not available.

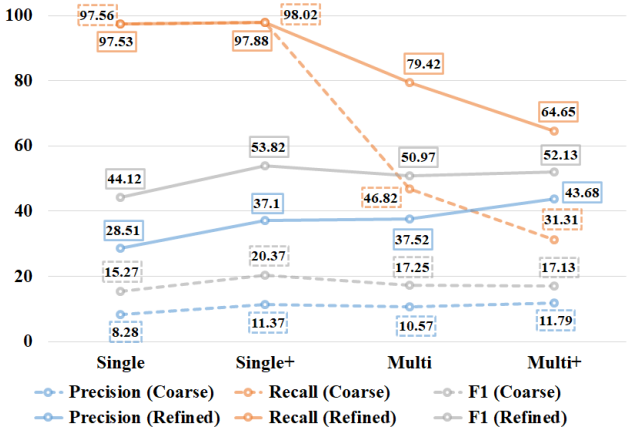

Figure 4: Document retrieval results on different parts of the development set. NEI samples are not taken into consideration since no evidence sentences are annotated for them.

5.2 Document Retrieval

Document retrieval results of different iterations on the development set are displayed in Table 4. With search engine power adopted through Wikipedia search, and statistical word frequency information joined through TF-IDF retrieval, coarse document retrieval obtains the highest recall of 92.77%. Besides, with dense semantic retrieval model guided and top one evidence hyperlinks joined, refined document retrieval shows even higher recall of 95.69%. Furthermore, refined document retrieval has much higher precision of 29.90% and F1 of 45.56%, respectively obtains 21.22% and 29.69% absolute increase than coarse retrieval. Therefore, our iterative fact extraction model has made great improvement on document retrieval.

Moreover, document retrieval results on different parts of the development set are displayed in Figure 4. Coarse document retrieval can handle Single and Single+ samples, where the recall are respectively as high as 97.56% and 98.02%. However, coarse document retrieval fails to handle multi-hop samples, where the recall of Multi and Multi+ samples are both pretty low, respectively 46.82% and 31.31%. Compared to coarse document retrieval, refined document retrieval shows comparable recalls but much higher precision and F1 score on Single and Single+ samples. Furthermore, refined document retrieval makes great progress on multi-hop samples. For Multi and Multi+ samples, refined document retrieval respectively achieves 32.60% and 33.34% absolute increase on recall. Besides, the precision and F1 score also get significantly improved. These results indicate the high efficiency of our iterative multi-view fact extraction model. However, although refined document retrieval has achieved significant improvement, the recall of multi-hop samples is still far lower than single-hop ones.

5.3 Sentence Retrieval

Sentence retrieval results on the development set are summarized in Table 5. Our IMCI framework obtains the highest recall of 92.86%, and significantly outperforms all baselines.

For sentence retrieval of iteration 1, upstream coarse document retrieval obtains an extremely low precision of 8.68% (in Table 4). Thus, for each claim, on average our sentence retrieval model is re-

Model	P	R	F1
UKP-Athene(2018)	-	86.24	-
NSMN(2019a)	36.49	86.79	51.38
GEAR-BERT$_{base}$(2019)	24.08	86.72	37.69
SR-MRS-BERT$_{base}$(2019b)	44.47	86.60	58.77
HESM-BERT$_{base}$(2020)$^{\#}$	-	90.50	-
Iteration 1	25.31	**90.30**	39.54
w.o. Alignment	24.86	90.16	38.97
Iteration 1 (Top 1)	**86.11**	78.08	**81.90**
Iteration 2	25.90	**91.98**	40.42
IMCI$^{\#}$	25.74	**92.86**	40.30

Table 5: Sentence retrieval results on the development set. According to the original task setup, we keep top 5 sentences as evidence for each claim. # means the models adopt iterative sentence retrieval. *w.o.* means without the item.

quested to distinguish top 5 sentences as candidate evidences from more than 250 sentences. In this condition, with intra-document contextual information joined, the model obtains pretty high recall of 90.30%, and shows comparable performance with state-of-the-art iterative sentence retrieval model HESM (Subramanian and Lee, 2020). Besides, top one candidate evidences show a pretty high precision of 86.11%. This shows the importance of intra-document context, and is the base of refined document retrieval. Besides, the high precision also guarantees that top one candidate evidences can be considered as inter-document context in sentence retrieval of iteration 2. With multi-view contextual information joined, our sentence retrieval model of iteration 2 obtains even higher recall of 91.98%. Moreover, full pipeline reranking makes the recall get far more increase to 92.86%. These show the great power of multi-view contextual information on fact extraction.

Moreover, sentence retrieval results on different parts of the development set are displayed in detail in Table 6. For Single and Single+ samples, the recall are respectively high at 96.33% and 95.90%, while the precision and F1 score are pretty low. However, the recall of Multi samples is pretty low at 64.41%, while that of Multi+ samples is far lower at 26.26%. Therefore, taking these and the document retrieval results in Figure 4 into consideration, it seems that fact extraction for multi-hop samples is still a difficult problem, although our model has made several progress. A main concern is that bidirectional information interaction between multi-hop evidences may be not guaranteed during fact verification.

Part	P	R	F1
Single	23.00	96.33	37.14
Single+	52.06	95.90	67.48
Multi	32.98	64.41	43.63
Multi+	45.25	26.26	33.23

Table 6: Sentence retrieval results on different parts of the development set. NEI samples are not taken into consideration since no evidence sentences are annotated for them.

5.4 Fact Verification

Fact verification results on the development set are shown in Figure 5. With multi-view contextual information joined, our IMCI framework obtains the highest label accuracy of 75.83% and the highest FEVER score of 73.21%. When ignoring intra-document encoding, the label accuracy and FEVER score suffer severe decrease to 74.23% and 71.58%. This indicates the great importance of intra-document contextual information on fact verification. However, compared to intra-document encoding, inter-document encoding and dual evidence fusion graph have relatively weak influence on the performance. These two components mainly aim to handle multi-hop samples. However, multi-hop samples take pretty low ratio (about 6.02% in total in Table 2). Even worse, multi-hop samples have suffered serious performance damage on upstream fact verification task (in Table 6). Evidence confidence aggregation also makes some contribution, indicating the influence of evidence label information. Besides, we also study the influence of fact verification. It seems that progress on fact verification mainly contributes to FEVER score while shows weak influence on label accuracy.

Moreover, for our IMCI framework, the statistic information of prediction errors on fact verification is shown in Figure 6. The framework can correctly distinguish SUPPORTS and REFUTES examples, since SUPPORTS (REFUTES) and REFUTES (SUPPORTS) errors respectively take about 3.66% and 11.51%. This may indicate that the logical boundary between SUPPORTS and REFUTES is relatively clear. Besides, the framework hardly mistakes SUPPORTS examples for NEI examples. However, it may be difficult for the framework to distinguish REFUTES examples from NEI examples, as well as NEI examples from non-NEI examples, for REFUTES (NEI), NEI (SUPPORTS), and

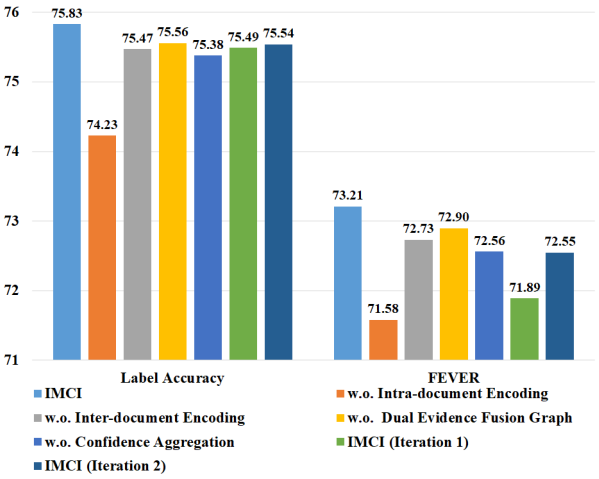

Figure 5: Fact verification results on the development set. These are averaged results on 4 random starts. *w.o.* means without the item.

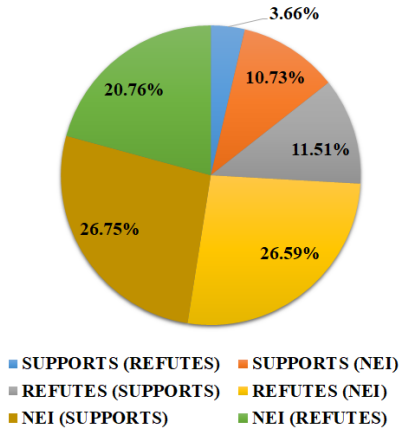

Figure 6: Statistic information of prediction errors on fact verification. (Label **out of** brackets denotes groundtruth, while label **in** brackets denotes wrong prediction.)

NEI (REFUTES) errors respectively take 26.59%, 26.75%, and 20.76%. The situation may be due to the pretty unbalanced label distribution of the training set (in Table 1). Besides, NEI may contain more complex logic semantic than the other two categories.

6 Related Work

• **Fake News Detection** Fake news detection has been attracting more attention. Ruffo et al. (2021) give a detailed survey about the development of this field. Textual-content based methods (Giachanou et al., 2019; Ghanem et al., 2020; Kaliyar et al., 2021) aim at understanding the linguistic and semantic information in the text to detect fake news. User-role based methods (Vo and Lee, 2019; Giachanou et al., 2020) pay more attention to the role of users in the propagation of fake news. Multi-modal approaches (Zlatkova et al., 2019; Fung et al., 2021) involve multi-modal information, i.e. text, table, knowledge base, image, speech and video, to evaluate the credibility of news. Besides, bots and trolls aim at influencing users with commercial, political or ideological purposes by spreading disinformation deliberately. The detection of them (Stella et al., 2018; Sayyadiharikandeh et al., 2020) is also an important direction. Moreover, Sheng et al. (2022) recently propose news environment perception for fake news detection, which focus on the background environment of fake news.

• **Fact Extraction** Fact extraction includes document retrieval and sentence retrieval. For document retrieval, Hanselowski et al. (2018) propose a constituency-based Wikipedia search model. Nie et al. (2019a) utilize a keyword matching model based on a quick string matching algorithm *FlashText* (Singh, 2017). Nie et al. (2019b) further adopt a combination model of keyword match and TF-IDF retrieval.

For sentence retrieval, Hanselowski et al. (2018), Nie et al. (2019a), and Zhou et al. (2019) respectively modify Enhanced Sequential Inference Model (ESIM) (Chen et al., 2017). These models separately encode the claim and an evidence sentence, and adopt cross-attention mechanism to accomplish information interaction between the claim and the evidence sentence. Nie et al. (2019b) and Liu et al. (2020) adopt BERT-based model. Subramanian and Lee (2020) propose iterative fact verification models to retrieve evidence sentences and combine evidence sets.

• **Fact Verification** For fact verification, Nie et al. (2019b) concatenate the claim and the evidence sentences into a sequence as input to BERT encoder, and take the hidden state of the first special token [CLS], as final inference representation. Zhou et al. (2019) adopt graph neural network for evidence aggregating and reasoning. Zhong et al. (2020) introduce semantic role information to construct refined graph, and adopt graph convolutional network to handle the task. Liu et al. (2020) propose fine-grained kernel-based graph attention network for information interaction between the claim and the evidences. Subramanian and Lee (2020) propose to combine evidence sets during fact extraction, and conduct fact verification on evidence sets. Si et al.

(2021) introduce topic model and stance detection model, and study the influence of topic and stance information on fact verification.

7 Conclusion

In this paper, we propose to integrate multi-view contextual information for fact extraction and verification. Our experimental results show that our IMCI model can obtain state-of-the-art performance on the task. Moreover, the ablation study results indicate that multi-view contextual information is essential for both fact extraction and fact verification. In the future, we will explore much stronger model to utilize contextual information in a more efficient way.

References

Nicola De Cao, Wilker Aziz, and Ivan Titov. 2019. Question answering by reasoning across documents with graph convolutional networks. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 2306–2317. Association for Computational Linguistics.

Qian Chen, Xiaodan Zhu, ZhenHua Ling, Si Wei, Hui Jiang, and Diana Inkpen. 2017. Enhanced lstm for natural language inference. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics*, pages 1657–1668. Association for Computational Linguistics.

Jacob Devlin, Ming Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. Bert: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 4171–4186. Association for Computational Linguistics.

Yi Fung, Christopher Thomas, Revanth Gangi Reddy, Sandeep Polisetty, Heng Ji, ShihFu Chang, Kathleen R. McKeown, Mohit Bansal, and Avi Sil. 2021. Infosurgeon: Cross-media fine-grained information consistency checking for fake news detection. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing*, pages 1683–1698. Association for Computational Linguistics.

Bilal Ghanem, Paolo Rosso, and Francisco M. Rangel Pardo. 2020. An emotional analysis of false information in social media and news articles. *ACM Transactions on Internet Technology*, 20(2):19:1–19:18.

Anastasia Giachanou, Esteban ARissola, Bilal Ghanem, Fabio Crestani, and Paolo Rosso. 2020. The role of personality and linguistic patterns in discriminating between fake news spreaders and fact checkers. In *Proceedings of the 25th International Conference on Applications of Natural Language to Information Systems*, Lecture Notes in Computer Science, pages 181–192. Springer.

Anastasia Giachanou, Paolo Rosso, and Fabio Crestani. 2019. Leveraging emotional signals for credibility detection. In *Proceedings of the 42nd International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 877–880. ACM.

Andreas Hanselowski, Hao Zhang, Zile Li, Daniil Sorokin, Benjamin Schiller, Claudia Schulz, and Iryna Gurevych. 2018. Ukp-athene: Multi-sentence textual entailment for claim verification. *CoRR*, abs/1809.01479.

Christopher Hidey, Tuhin Chakrabarty, Tariq Alhindi, Siddharth Varia, Kriste Krstovski, Mona T. Diab, and Smaranda Muresan. 2020. Deseption: Dual sequence prediction and adversarial examples for improved fact-checking. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 8593–8606. Association for Computational Linguistics.

Rohit Kumar Kaliyar, Anurag Goswami, and Pratik Narang. 2021. Fakebert: Fake news detection in social media with a bert-based deep learning approach. *Multimedia Tools and Applications*, 80(8):11765–11788.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Roberta: A robustly optimized BERT pretraining approach. *CoRR*, abs/1907.11692.

Zhenghao Liu, Chenyan Xiong, Maosong Sun, and Zhiyuan Liu. 2020. Fine-grained fact verification with kernel graph attention network. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7342–7351. Association for Computational Linguistics.

Giovanni Da San Martino, Stefano Cresci, Alberto Barron Cedeno, Seunghak Yu, Roberto Di Pietro, and Preslav Nakov. 2020. A survey on computational propaganda detection. In *Proceedings of the 29th International Joint Conference on Artificial Intelligence*, pages 4826–4832. ijcai.org.

Yixin Nie, Haonan Chen, and Mohit Bansal. 2019a. Combining fact extraction and verification with neural semantic matching networks. In *Proceedings of the 33rd AAAI Conference on Artificial Intelligence*, pages 6859–6866. AAAI Press.

Yixin Nie, Songhe Wang, and Mohit Bansal. 2019b. Revealing the importance of semantic retrieval for

machine reading at scale. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing*, pages 2553–2566. Association for Computational Linguistics.

Kosuke Nishida, Kyosuke Nishida, Masaaki Nagata, Atsushi Otsuka, Itsumi Saito, Hisako Asano, and Junji Tomita. 2019. Answering while summarizing: Multi-task learning for multi-hop qa with evidence extraction. In *Proceedings of the 57th Conference of the Association for Computational Linguistics*, pages 2335–2345. Association for Computational Linguistics.

Giancarlo Ruffo, Alfonso Semeraro, Anastasia Giachanou, and Paolo Rosso. 2021. Surveying the research on fake news in social media: a tale of networks and language. *CoRR*, abs/2109.07909.

Mohsen Sayyadiharikandeh, Onur Varol, KaiCheng Yang, Alessandro Flammini, and Filippo Menczer. 2020. Detection of novel social bots by ensembles of specialized classifiers. In *Proceedings of 29th ACM International Conference on Information and Knowledge Management*, pages 2725–2732. ACM.

Qiang Sheng, Juan Cao, Xueyao Zhang, Rundong Li, Danding Wang, and Yongchun Zhu. 2022. Zoom out and observe: News environment perception for fake news detection. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics*, pages 4543–4556. Association for Computational Linguistics.

Jiasheng Si, Deyu Zhou, Tongzhe Li, Xingyu Shi, and Yulan He. 2021. Topic-aware evidence reasoning and stance-aware aggregation for fact verification. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics*, pages 1612–1622. Association for Computational Linguistics.

Vikash Singh. 2017. Replace or retrieve keywords in documents at scale. *CoRR*, abs/1711.00046.

Massimo Stella, Emilio Ferrara, and Manlio De Domenico. 2018. Bots increase exposure to negative and inflammatory content in online social systems. *Proceedings of the National Academy of Sciences*, 115(49):12435–12440.

Shyam Subramanian and Kyumin Lee. 2020. Hierarchical evidence set modeling for automated fact extraction and verification. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing*, pages 7798–7809. Association for Computational Linguistics.

James Thorne, Andreas Vlachos, Christos Christodoulopoulos, and Arpit Mittal. 2018. Fever: a large-scale dataset for fact extraction and verification. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics:Human Language Technologies*, pages 809–819. Association for Computational Linguistics.

Ming Tu, Kevin Huang, Guangtao Wang, Jing Huang, Xiaodong He, and Bowen Zhou. 2020. Select, answer and explain: Interpretable multi-hop reading comprehension over multiple documents. In *Proceedings of the 34th AAAI Conference on Artificial Intelligence*, pages 9073–9080. AAAI Press.

Ming Tu, Guangtao Wang, Jing Huang, Yun Tang, Xiaodong He, and Bowen Zhou. 2019. Multi-hop reading comprehension across multiple documents by reasoning over heterogeneous graphs. In *Proceedings of the 57th Conference of the Association for Computational Linguistics*, pages 2704–2713. Association for Computational Linguistics.

Nguyen Vo and Kyumin Lee. 2019. Learning from fact-checkers: Analysis and generation of fact-checkinglanguage. In *Proceedings of the 42nd International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 335–344. ACM.

Deming Ye, Yankai Lin, Jiaju Du, Zhenghao Liu, Peng Li, Maosong Sun, and Zhiyuan Liu. 2020. Coreferential reasoning learning for language representation. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing*, pages 7170–7186. Association for Computational Linguistics.

Chen Zhao, Chenyan Xiong, Corby Rosset, Xia Song, Paul N. Bennett, and Saurabh Tiwary. 2020. Transformer-xh: Multi-evidence reasoning with extra hop attention. In *Proceedings of the 8th International Conference on Learning Representations*. OpenReview.net.

Wanjun Zhong, Jingjing Xu, Duyu Tang, Zenan Xu, Nan Duan, Ming Zhou, Jiahai Wang, and Jian Yin. 2020. Reasoning over semantic-level graph for fact checking. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 6170–6180. Association for Computational Linguistics.

Jie Zhou, Xu Han, Cheng Yang, Zhiyuan Liu, Lifeng Wang, Changcheng Li, and Maosong Sun. 2019. Gear: Graph-based evidence aggregating and reasoning for fact verification. In *Proceedings of the 57th Conference of the Association for Computational Linguistics*, pages 892–901. Association for Computational Linguistics.

Dimitrina Zlatkova, Preslav Nakov, and Ivan Koychev. 2019. Fact-checking meets fauxtography: Verifying claims about images. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing*, pages 2099–2108. Association for Computational Linguistics.

Prompt Combines Paraphrase: Teaching Pre-trained Models to Understand Rare Biomedical Words

Haochun Wang[1], **Chi Liu**[1], **Nuwa Xi**[1], **Sendong Zhao**[1]*, **Meizhi Ju**[2], **Shiwei Zhang**[2]
Ziheng Zhang[2], **Yefeng Zheng**[2], **Bing Qin**[1] **and Ting Liu**[1]
[1]Research Center for Social Computing and Information Retrieval,
Harbin Institute of Technology, China
[2]Tencent Jarvis Lab, Shenzhen, China
{hcwang,cliu,nwxi,sdzhao,bqin,tliu}@ir.hit.edu.cn
meizhi.ju@outlook.com, {zswvizhang,zihengzhang,yefengzheng}@tencent.com

Abstract

Prompt-based fine-tuning for pre-trained models has proven effective for many natural language processing tasks under few-shot settings in general domain. However, tuning with prompt in biomedical domain has not been investigated thoroughly. Biomedical words are often rare in general domain, but quite ubiquitous in biomedical contexts, which dramatically deteriorates the performance of pre-trained models on downstream biomedical applications even after fine-tuning, especially in low-resource scenarios. We propose a simple yet effective approach to helping models learn rare biomedical words during tuning with prompt. Experimental results show that our method can achieve up to 6% improvement in biomedical natural language inference task without any extra parameters or training steps using few-shot vanilla prompt settings.

1 Introduction

Pre-trained models have achieved a great success in natural language processing (NLP) and become a new paradigm for various tasks (Peters et al., 2018; Devlin et al., 2019; Liu et al., 2019; Qiu et al., 2020). Many studies have paid attention to pre-trained models in biomedical NLP tasks (Lee et al., 2020; Lewis et al., 2020; Zhao et al., 2021). However, plain pre-trained models sometimes cannot do very well in biomedical NLP tasks. In general, there are two challenges to fully exploit the potential of the pre-trained models for biomedical NLP tasks, i.e., (1) **limited data** and (2) **rare biomedical words**. Firstly, it is common that the amount of biomedical labeled data is limited due to strict privacy policy constraints (Šuster et al., 2017), high cost and professional requirement for data annotation. Pre-trained models perform poorly with few samples since abundant training samples are essential to optimize task-related parameters (Liu et al., 2021a). Secondly, biomedical words are usually low-frequency words but critical to understanding biomedical texts. As an example of natural language inference (NLI) task in Figure 1, the model goes wrong during tuning when faced with a rare word "*afebrile*"[1] in the premise, whose meaning is "*having no fever*". It can be no easy for the pre-trained models to predict the correct label if the models haven't seen the rare biomedical words for enough times during pre-training or tuning stage. Thus, pre-trained models cannot capture the precise semantics of biomedical texts in the scenario of low-resource tasks.

With very few annotated samples available for a new task, it is hard to effectively fine-tune pre-trained models with the additional task-specific parameters, which is even more of a challenge to biomedical domain as mentioned above. Prompt technique has been introduced to smooth the fine-tuning process in the few-shot settings by narrowing down the gap between pre-training stage and the downstream task in general domain (Liu et al., 2021a), as demonstrated in Figure 2. Therefore, it is beneficial to adapt prompt-based tuning to biomedical NLP tasks.

Although the challenge of rare words is a critical problem for the biomedical pre-trained models, only a handful of works have studied the issue and most of them focus on enriching the representation of rare words through pre-training stage (Schick and Schütze, 2020; Yu et al., 2021; Wu et al., 2020). Thus, it naturally requires them to involve a second-round pre-training or further training steps with biomedical knowledge to achieve the above goal, which is highly time-consuming and inefficient. Alternatively, we emphasize on tuning stage instead of pre-training to resolve these issues. When coming across an unknown word, human may seek the

*Corresponding author

[1]There are around 4 billion words in the selected biomedical pre-training texts while "afebrile" appears only about 100,000 times, accounting for 0.0025%.

Proceedings of the 29th International Conference on Computational Linguistics, pages 1422–1431
October 12–17, 2022.

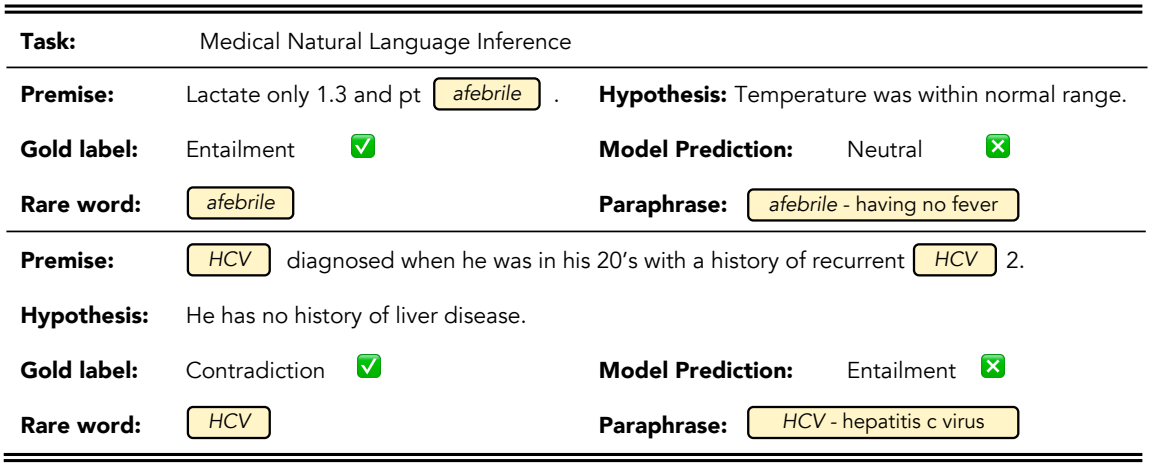

Figure 1: Failures of a biomedical pre-trained model (Lewis et al., 2020) on the biomedical NLI task.

dictionary for its paraphrase. Enlightened by this phenomenon, we propose to explain rare biomedical words with the paraphrases on the basis of prompt-based tuning. The new approach could enhance tuning capability in understanding biomedical words. Furthermore, as a generic plug-in module for non-specific datasets, our approach is model-agnostic and can be easily transferred to other domains.[2]

In summary, our contributions are as follows:

- We investigate a valuable problem of the adaptation of pre-trained models to few-shot scenarios to enhance biomedical text understanding with a focus on rare biomedical words.
- We propose a novel method to combine the prompt paradigm and paraphrases of rare biomedical words in the tuning stage of pre-trained models to address the limitation caused by "rare but key words" in biomedical texts.
- We evaluate on six pre-trained models over two biomedical natural language understanding datasets—MedNLI and MedSTS. Our approach can improve the performance by up to 6% in the few-shot settings. Moreover, we discuss how the paraphrases improve the pre-trained models and provide a perspective about task-related rare words.

2 Related Work

Representation learning of rare words in pre-trained models. Words in the vocabulary list follow a Zipf distribution (Zipf, 2016) by and large. Previous works have shown that the word representation space of the pre-trained models is anisotropic and high-frequency words dominate the representation of a sentence which can induce semantic bias (Gao et al., 2019; Li et al., 2020; Yan et al., 2021). Meanwhile, it has also been proven that rare words limit the performance of pre-trained models as the rare words can play a decisive role in the sentence understanding (Schick and Schütze, 2020; Wu et al., 2020; Yu et al., 2021). Schick and Schütze (2020) introduced one-token approximation to infer the embedding of arbitrary rare word by a single token. Wu et al. (2020) proposed to take notes on the fly to maintain a note dictionary for rare words to save the contextual information that helps enhance the representation during pre-training.

Biomedical pre-trained models. With the booming trend of pre-trained models in NLP tasks (Peters et al., 2018; Devlin et al., 2019; Liu et al., 2019), various trials have been made to investigate the pre-trained models in biomedical domain (Peng et al., 2019; Lee et al., 2020; Huang et al., 2019). Lewis et al. (2020) and Gu et al. (2021a) further built the domain-specific vocabulary to amend the representation of biomedical words. More recent works guided biomedical pre-trained models with domain knowledge. For example, Zhang et al. (2021) amplified the biomedical entities with type information from neighbor entities. Michalopoulos et al. (2021) learned clinical word embeddings with the association of synonyms in the Unified Medical Language System (UMLS) Metathesaurus.

[2]We release our code at https://github.com/s65b40/prompt_n_paraphrase

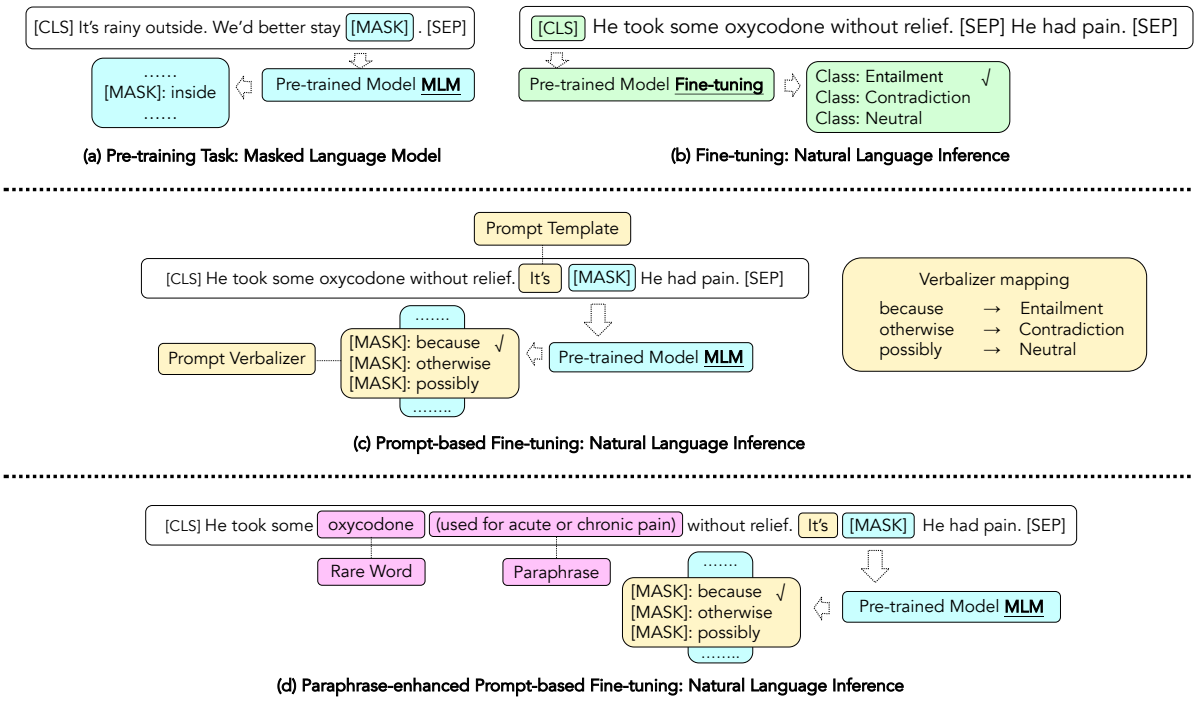

Figure 2: Examples for paradigms of (a) Masked Language Model (MLM) pre-training; (b) Task-specific fine-tuning; (c) Prompt-based fine-tuning, with same task as pre-training process; (d) Paraphrase-enhanced prompt-based fine-tuning. Best viewed in color.

Tuning pre-trained models with prompt. Many works were dedicated to applying prompt in fine-tuning by adapting the downstream tasks to the paradigm of pre-training tasks. Prompts that have been employed by now can be categorized into two groups: (1) discrete prompt in natural language (Schick and Schütze, 2021; Gao et al., 2021) and (2) continuous prompt in representation based on trainable vectors (Li and Liang, 2021; Shin et al., 2020b). Discrete prompt follows the settings of pre-training tasks and converts the downstream tasks into a cloze question format without requiring additional parameters. Continuous prompt inserts prompt embeddings into the models, which could perform better than discrete prompt but at the expense of explainability and extra training cost on additional data for the prompt embeddings (Wei et al., 2021; Gu et al., 2021b). Recently, Liu et al. (2021b) considered training continuous prompts as a parameter-efficient method instead of tuning the parameters of the entire pre-trained model. In this study, we follow the paradigm of discrete prompt to avoid introducing more ambiguity from the prompt embeddings or training costs on additional training data.

3 Method

In this section, we introduce how we find the rare biomedical words and append paraphrases to the rare biomedical words with the prompt-based tuning of pre-trained models in a model-agnostic plug-in manner.

3.1 Rare Words

The rarity of a word mostly depends on its frequency in a certain corpus, which can vary from context to context. A rare word in the pre-training corpora is possibly not *that* rare in the down-stream tasks. In this work, we define the "rare words" as the words whose frequency is under a specific threshold in the pre-training corpora as aforementioned.

Meanwhile, although the pre-trained models tokenize the input words into tokens, tokenizers based on byte-pair encoding (Sennrich et al., 2016) or WordPiece (Schuster and Nakajima, 2012) split words into sub-words by frequency or likelihood, which is both dominated by the common words. Thus, although the rare words can be split into possible non-rare tokens, there is not much semantics from the original rare words retained after being tokenized into common tokens for the pre-trained

models. Also, tokenizers of different pre-trained models can tokenize the same rare word into different tokens and consequentially make rare tokens model-related. For example, BERT-Large (Devlin et al., 2019) model tokenizes "*afebrile*" into "*af-eb-ril-e*" while Biomedical-Clinical-RoBERTa-Large (Lewis et al., 2020) model tokenizes it into "*a-fe-brile*".

3.2 Selection of Rare Biomedical Words

To obtain the frequency of words, we adopt the biomedical corpora including PubMed abstract,[3] PubMed Central[4] (PMC) full-text and MIMIC-III dataset,[5] which are widely used for pre-training biomedical language models, such as BC-RoBETRa (Lewis et al., 2020), BioBERT (Lee et al., 2020), and PuBMedBERT (Gu et al., 2020). We loop the above corpora to obtain the frequency of each word in the pre-training phase. The rare words found in biomedical corpora are likely to contain words not only in the biomedical domain but also in the general domain. Instead of including all rare words, we consider rare words from biomedical domain with the following two reasons: (1) **Domain-specific distribution**: unlike the general domain, distribution of words in the biomedical domain is shaped with domain-specific terms, such as disease, medicine, diagnosis and treatment (Lee et al., 2020). (2) **Task-specific words**: rare words from the biomedical domain can contribute more to biomedical tasks than that from the general domain. Therefore, we introduce a threshold, an empirical hyper-parameter, to assist the selection of rare words following Yu et al. (2021), and we also experiment with different thresholds for the rare words in Section 4.3. We retrieve the paraphrases of the rare biomedical words from an online dictionary "Wiktionary".[6] To optimize the selection, we only keep the rare words that are tagged with medical-related categories from the Wiktionary, i.e. *medical*, *medicine*, *disease*, *symptom* and *pharmacology*.

3.3 Selection of Paraphrases

There can be more than just one paraphrase for a rare biomedical word and it is tricky to choose the most appropriate paraphrases. Therefore, to avoid introducing noise from the inappropriate paraphrases, we exclude rare biomedical words with more than one corresponding paraphrase. In addition, we ignore paraphrases that contain additional rare words whose frequencies are below the set threshold since it only replaces one rare word with another. Meanwhile, considering that biomedical abbreviations are likely to be tokenized into separate letters with no meaningful semantic information, we retrieve and append the paraphrases to all the biomedical abbreviations.

3.4 Prompt-based Fine-Tuning with Paraphrases

When coming across new words during reading, humans habitually seek dictionaries for the corresponding paraphrases to help us understand. Following the same idea, we guide the pre-trained models with paraphrases, where rare words are followed by the parenthesis punctuation, as shown in Figure 2(d). In this way, given a pre-trained model, paraphrases of biomedical rare words can be considered as a portable plug-in module and generated for any dataset instantly before prompt-based fine-tuning.

4 Experiments

4.1 Setup

Models. To demonstrate that our approach is model-agnostic, we adopt six pre-trained models in both general and biomedical domains, namely (1) BERT-Large (Devlin et al., 2019), (2) RoBERTa-Large (Liu et al., 2019), (3) BioBERT-Base (Lee et al., 2020), (4) PubMedBERT-Base (Gu et al., 2020), (5) SciBERT-Base (Beltagy et al., 2019) and (6) BC-RoBERTa-Large (Biomedical-Clinical RoBERTa-Large) (Lewis et al., 2020). During prompt-based tuning, we use the same set of hyper-parameters for all the six pre-trained models, including learning rate of 1×10^{-5}, batch size of 2 and max epoch of 10.

Datasets. Previous work demonstrates that rare words have more impact on Natural Language Understanding (NLU) tasks than Information Extraction (IE) (Schick and Schütze, 2020), while most biomedical NLP tasks fall into the category of IE (Shin et al., 2020a; Gu et al., 2021a). To better demonstrate the method effectiveness, we perform evaluation over the two biomedical NLU datasets, namely MedNLI (Romanov and Shivade, 2018)

[3] https://pubmed.ncbi.nlm.nih.gov
[4] https://www.ncbi.nlm.nih.gov/pmc
[5] https://physionet.org/content/mimiciii/1.4/
[6] Wiktionary - https://en.wiktionary.org/

and MedSTS[7] (Wang et al., 2020).

MedNLI is a natural language inference dataset where premises are selected from real clinical notes in MIMIC-III (Johnson et al., 2016). And MedSTS is a semantic textual similarity dataset gathered from a clinical corpus at Mayo Clinic and the ground-truth label of the similarity is the mean of the subjectively annotated scores from multiple annotators. As MedSTS is actually a regression task, we adapt the task following Gao et al. (2021) and convert it into a classification task. We use the same data splitting of training, development and test sets as the original two datasets. Statistics of datasets can be found in Table 1.

Dataset	Train	Dev	Test
MedNLI	11,232	1,395	1,422
MedSTS	750	/	318

Table 1: Statistics of the MedNLI and MedSTS datasets. We use the ClinicalSTS-2018 subset of MedSTS.

Few-shot datasets. Initialized with 10 different random seeds, we randomly sample instances within the range of 16 to 256 from corresponding training and development sets as the few-shot training and development sets. The original test set is directly used as the few-shot test set. Note that there is no development set in MedSTS, so we sample the few-shot development set from the original training set with the same quantity of samples as the few-shot training set with no overlapping instances. Accuracy and Pearson correlation coefficients are used as the evaluation metrics for MedNLI and MedSTS, respectively.

Prompt settings. We prepare discrete prompts using the same prompt settings from Schick and Schütze (2021) and Gao et al. (2021), which correspond to the NLI and STS tasks respectively in Table 2.

Task	Template	Verbalizer
MedNLI	<Sent1>. [MASK]. <Sent2>	Yes/No/maybe
MedSTS	<Sent1>. [MASK]. <Sent2>	Yes/No

Table 2: Prompt settings for MedNLI and MedSTS.

Rare biomedical words and paraphrases. We find that a threshold of rare biomedical words at 200,000, corresponding to a frequency less than 0.005% in the pre-training corpora, can yield better results in most scenarios (details in Section 4.3), so we consider the biomedical words that appear less than 200,000 times as "rare biomedical words" and prepare the same rare word and paraphrase sets for all the models to validate the generalization ability of our approach.

[7]We use ClinicalSTS-2018 which is a sub-dataset of MedSTS provided by the maintainers of the MedSTS project.

4.2 Few-shot learning results

We report the mean accuracy for the MedNLI task and Pearson correlation coefficient for the MedSTS task over 10 sampled few-shot datasets based on different random seeds, along with standard deviation and p-value of paired t-test. Table 3 and Table 4 show the results for the two tasks on six pre-trained models.

Results on MedNLI. The pre-trained models with the paraphrases for rare biomedical words can outperform the baselines in all cases and can bring about 6% improvement on average for few-shot learning with 16 training samples and 2% with 256 training samples. All performance improvements are statistically significant with p-value less than 0.05 except only two out of ten cases from the RoBERTa-based model in general domain. Besides, small pre-trained biomedical models are comparable with large pre-trained models in the general domain under the few-shot settings. Furthermore, with more training samples up to 256, our approach is consistently effective.

Results on MedSTS. Similar to MedNLI, the incorporation of paraphrases improves the performances compared with baselines in general. For some cases, statistical significance is not as stable as that on MedNLI for two reasons: (1) Some ground-truth labels in the MedSTS task can be biased due to the subjectivity of annotation (Yang et al., 2020); (2) Rare biomedical words shared in the sentence pair of the same sample can be a shortcut for the pre-trained models, as the paraphrases increase the overlap between the two sentences and mislead the models to overlook the rest of the sentences (McCoy et al., 2019).

4.3 Thresholds for rare biomedical words

The number of paraphrases of rare biomedical words involved in the samples can directly affect the model performance, which is controlled by the pre-set threshold. To measure the influence of different thresholds, we conduct experiments over

MedNLI					
Model \ #Samples	16	32	64	128	256
BERT-Large	38.9 (3.7)	44.5 (5.2)	50.1 (5.2)	54.8 (2.5)	59.9 (1.2)
+ paraphrase	**40.8** (4.1)	**46.0** (5.5)	**53.3** (4.9)	**58.1** (1.4)	**61.9** (1.4)
p-value	< 0.01	0.02	< 0.01	< 0.01	< 0.01
RoBERTa-Large	43.2 (6.7)	52.1 (8.2)	63.6 (4.6)	69.2 (1.8)	72.7 (1.4)
+ paraphrase	**49.5** (8.1)	**56.1** (7.6)	**65.6** (2.9)	**70.8** (0.7)	**74.0** (1.3)
p-value	< 0.01	0.03	0.08	0.06	0.02
BioBERT-Base	34.1 (1.5)	38.5 (3.3)	42.5 (4.6)	52.1 (2.5)	59.4 (1.6)
+ paraphrase	**36.3** (1.8)	**40.9** (3.5)	**45.2** (4.3)	**54.0** (2.3)	**60.4** (4.7)
p-value	< 0.01	< 0.01	< 0.01	0.02	0.03
PubMedBERT-Base	40.5 (3.4)	46.8 (4.9)	53.9 (4.0)	62.9 (1.6)	69.2 (1.1)
+ paraphrase	**45.0** (4.0)	**49.7** (5.0)	**56.4** (3.4)	**65.8** (1.6)	**71.0** (1.6)
p-value	< 0.01	< 0.01	< 0.01	< 0.01	< 0.01
SciBERT-Base	36.8 (1.2)	41.4 (3.3)	49.0 (3.7)	54.7 (2.0)	60.9 (1.0)
+ paraphrase	**38.2** (2.1)	**45.4** (5.2)	**50.1** (3.5)	**56.4** (2.2)	**61.9** (1.6)
p-value	< 0.01	0.03	0.01	0.02	< 0.01
BC-RoBERTa-Large	51.3 (5.9)	60.6 (6.7)	71.0 (3.7)	80.6 (1.3)	83.1 (1.3)
+ paraphrase	**56.6** (5.0)	**62.3** (6.0)	**74.5** (3.0)	**81.1** (1.5)	**83.6** (1.0)
p-value	< 0.01	0.05	< 0.01	0.02	0.01

Table 3: Few-shot results on the MedNLI task using various pre-trained models with training and development sets of different sizes. We report mean (standard deviation) performance of accuracy over 10 different random seeds, along with the *p*-value of the paired *t*-test. + paraphrase: with paraphrases of selected rare biomedical words.

MedSTS					
Model \ #Samples	16	32	64	128	256
BERT-Large	14.1 (7.4)	24.8 (10.1)	43.6 (5.7)	60.2 (4.7)	72.2 (4.5)
+ paraphrase	**18.5** (9.0)	**28.2** (12.1)	**48.7** (7.1)	**64.1** (5.1)	**72.7** (4.6)
p-value	< 0.01	0.04	< 0.01	0.01	0.08
RoBERTa-Large	29.5 (9.3)	41.7 (18.5)	55.1 (12.1)	67.6 (6.8)	76.0 (3.5)
+ paraphrase	**34.6** (13.5)	**46.1** (13.0)	**57.7** (12.7)	**69.9** (6.7)	**77.2** (3.7)
p-value	0.04	0.01	0.02	0.03	0.02
BioBERT-Base	17.3 (14.4)	26.3 (13.7)	41.4 (9.0)	52.2 (10.6)	63.0 (7.3)
+ paraphrase	**20.0** (12.9)	**28.2** (12.8)	**43.1** (8.6)	**53.7** (9.2)	**64.2** (7.1)
p-value	0.03	0.02	0.01	0.02	0.02
PubMedBERT-Base	10.3 (9.8)	22.8 (10.7)	36.9 (10.6)	48.1 (11.9)	**65.5** (9.5)
+ paraphrase	**18.8** (13.7)	**27.3** (11.6)	**39.9** (10.4)	**49.4** (12.4)	65.1 (9.4)
p-value	0.04	0.01	< 0.01	0.07	0.1
SciBERT-Base	15.2 (16.9)	29.1 (18.2)	45.5 (14.0)	60.8 (9.0)	72.2 (7.3)
+ paraphrase	**19.2** (15.5)	**32.6** (18.4)	**48.9** (14.4)	**62.5** (8.3)	**73.4** (6.8)
p-value	0.01	< 0.01	0.02	< 0.01	< 0.01
BC-RoBERTa-Large	**54.2** (8.1)	63.9 (9.2)	73.3 (3.8)	77.4 (2.7)	81.5 (1.5)
+ paraphrase	53.0 (7.4)	**67.2** (6.6)	**74.5** (2.7)	**79.1** (1.6)	**81.8** (1.2)
p-value	0.08	0.01	0.03	< 0.01	0.02

Table 4: Few-shot results on the MedSTS task using various pre-trained models with training and development sets of different sizes. We report mean (standard deviation) performance of Pearson correlation coefficient over 10 different random seeds, along with the *p*-value of the paired *t*-test. + paraphrase: with paraphrases of selected rare biomedical words.

threshold \ Dataset		16		32		64		128		256		Test
		Train	Dev	Train	Dev	Train	Dev	Train	Dev	Train	Dev	
t = 20k	*	1.1	1.1	1.9	1.7	3.4	3.9	6.8	7.6	13.2	14.3	29
	**	1.1	1.1	1.9	1.8	3.7	4.4	7.3	9.0	16.3	21.1	129
t = 50k	*	1.8	1.5	3.5	2.2	6.8	4.8	12.4	9.2	21.8	17.0	40
	**	1.8	1.5	3.5	2.3	7.1	5.4	13.1	11.2	26.6	24.9	196
t = 100k	*	1.8	1.6	3.8	2.6	7.2	5.9	14.0	11.1	25.3	20.2	41
	**	1.8	1.6	3.8	2.8	7.6	6.5	14.8	13.0	30.8	25.4	179
t = 200k	*	1.9	2.1	4.3	3.5	7.6	6.8	13.9	12.3	23.3	21.7	41
	**	1.9	2.1	4.3	3.7	8.2	7.8	15.3	15.5	33.4	32.6	185

Table 5: The number of selected rare biomedical words in the training, development and test sets for different few-shot datasets on MedNLI. "t = 20k" means the threshold for rare biomedical words is 20k. "*" is the number of different rare biomedical words within the threshold. "**" denotes the total occurrences of rare biomedical words in the dataset (a rare biomedical word can appear more than once).

MedNLI task since its labelled data is more objective compared to MedSTS.

Statistics of rare biomedical words. Table 5 shows the mean of rare biomedical words in the training, development and test sets under different thresholds for the MedNLI task. Among the above thresholds, the number of rare biomedical words varies from 29 to 41 in the test set, while the total number of all rare biomedical words varies from 129 to 196. Each rare biomedical word appears around 5 times on average in the test set. Note that as the threshold increases, the number of paraphrases does not necessarily increase as fast as the number of rare biomedical words which is because that although higher threshold does include more rare biomedical words, not all are appended with paraphrases as some of them are excluded as mentioned in Section 3.3.

In addition to the fixed threshold of 200,000, we experiment with the thresholds ranging from 20,000 to 200,000. Table 6 further demonstrates the effectiveness of paraphrases of rare biomedical words in most cases, where 200,000 of the threshold performs the best.[8]

5 Discussion

Train with more samples. Apart from applying paraphrases in few-shot scenarios, we also attempt with more training samples for the MedNLI task, even with full-size training dataset. We attempt to sample the same number of development samples from the official MedNLI development set (with 1,395 samples) to match the number of training samples. If the training set is larger than the full-size development set, we just use the whole development set.

#Samples	16	32	64	128	256
w/o paraphrase	51.3	60.6	71.0	80.6	83.1
t = 20k	55.0	61.9	72.4	80.6	83.4
t = 50k	54.2	62.1	73.8	81.0	**83.7**
t = 100k	54.5	**62.4**	73.4	80.4	83.2
t = 200k	**56.6**	62.3	**74.5**	**81.1**	83.6

Table 6: Results for few-shot learning with different thresholds of rare biomedical words on MedNLI with the BC-RoBERTa-Large model. *w/o* paraphrase: without paraphrases for rare biomedical words. t: the threshold for rare biomedical words.

Table 7 demonstrates that our method outperforms the baseline on four out of six cases and performs comparably on the remaining two cases. When the whole training set is used, our model with paraphrases achieves 0.8% improvement. The minor improvement might be attributed to the "not so rare" biomedical words as the rarity of words decreases during the expansion of training set, which helps the pre-trained models learn the semantics of the rare biomedical words better even without paraphrases.

Which to look up? Paraphrases can be helpful in general. However, for individual samples, that is not always the case. In Table 8, we further scrutinize the cases where the model yields different predictions after adding the paraphrases. It is observed that the paraphrases can be less effective if the paraphrases of rare words are not task-related

[8] We also attempt with thresholds higher than 200,000 but it will not bring improvement as much as 200,000.

MedNLI						
Model \ #Training	512	1024	2048	4096	8192	full-size
BC-RoBERTa-Large	84.8 (0.7)	85.5 (0.8)	86.4 (0.5)	86.3 (0.7)	86.2 (0.6)	85.9 (0.6)
+ paraphrases	**85.2** (0.8)	**86.3** (1.0)	86.4 (0.7)	86.3 (0.6)	**86.7** (0.5)	**86.7** (0.7)
p-value	0.05	< 0.01	0.43	0.36	< 0.01	0.02

Table 7: Test results on the MedNLI dataset with larger size of training sets. We report mean (and standard deviation) accuracy. + paraphrase: with paraphrases of rare biomedical words.

Sentence Pairs	w/o paraphrases	w/ paraphrases
P: She was found to have **BRBPR** *(bright red blood per rectum)* on rectal exam. H: the patient had bright red blood per rectum	Neutral	Entailment (**right answer**)
P: Antenatal history - pregnancy complicated by chronic hypertension with increased gestational hypertension leading to admission 3 days prior to delivery followed by cesarean section. H: The patient had **proteinuria** *(The presence of protein in the urine)* during pregnancy	Entailment	Neutral (**right answer**)
P: Following this rehab admission she was sent to a different OSH on [**2725-10-26**], for acute **CHF** *(congestive heart failure)* and at least one PEA arrest. H: The patient has a poorly functioning heart.	Contradiction	Entailment (**right answer**)
P: The patient was sent to the HD unit prior to coming to the floor for **workup** *(A general medical examination to assess a persons health and fitness)* of fever. H: The patient has an infection	Neutral (**right answer**)	Contradiction
P: - **COPD** *(chronic obstructive pulmonary disease)* - obesity - unspecified hypoxemia - CNS lymphoma c/b CVAs x3 (posterior circulation) and seizure d/o - history of SAH while on coumadin - diastolic heart failure - coronary artery disease - atrial fibrillation - hypertension - hyperlipidemia - severe OSA (did not tolerate CPAP in the past) - primary hyperparathyroidism/25-vit D deficiency c/b nephrolithiasis - toxic multinodular goiter with **subclinical** *(Less than is needed for clinical reasons)* hyperthyroidism - neovascular glaucoma c/b right eye blindness H: Patient has a history of malignancy	Neutral (**right answer**)	Entailment

Table 8: Cases that model predicts differently after the supplement of paraphrases for rare biomedical words in MedNLI. "P" for Premise and "H" for Hypothesis. Words in **bold** are rare biomedical words and expressions in *italic* inside the parentheses are the paraphrases of rare biomedical words.

and critical for sentence understanding. For instance, in the third case in Table 8, "*CHF*" is a rare biomedical word which means "*congestive heart failure*". Pre-trained models can easily match the hypothesis of "*poorly functioning*" with the given premise. Otherwise, paraphrases can possibly introduce confusion. For example, in the wrong case in Table 8, although "*workup*" is a rare biomedical word in the pre-training corpora, it does not severely affect the semantics of the sentence and the pre-trained model turns to make wrong prediction with the misdirection from supplementary paraphrase.

Actually, humans tend to continue reading unless the unknown words hinder the understanding. With this motivation, we believe that it is worthwhile to explore how to attach *informative* paraphrases for the rare words, which will be investigated in future.

6 Conclusion

Rare biomedical words are pervasive in biomedical texts, and understanding domain-specific rare words remains a tough challenge for pre-trained models. In this work, we presented a simple yet effective method to help the pre-trained models grasp the semantics of rare biomedical words.

Enlightened by human reading behavior, we taught the pre-trained models to understand rare biomedical words by incorporating paraphrases of rare biomedical words. Our method can be regarded as a generic plug-in approach for prompt-based tuning without additional parameters. Experiments showed that our method could substantially improve the pre-trained models under few-shot settings.

Acknowledgements

We thank the anonymous reviewers for their helpful and constructive comments and gratefully acknowledge the support of the National Key R&D Program of China (2021ZD0113302) and the National Natural Science Foundation of China Youth Fund (62206079).

References

Iz Beltagy, Kyle Lo, and Arman Cohan. 2019. SciBERT: A pretrained language model for scientific text. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 3615–3620, Hong Kong, China. Association for Computational Linguistics.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. Bert: Pre-training of deep bidirectional transformers for language understanding. In *NAACL-HLT (1)*.

Jun Gao, Di He, Xu Tan, Tao Qin, Liwei Wang, and Tieyan Liu. 2019. Representation degeneration problem in training natural language generation models. In *International Conference on Learning Representations*.

Tianyu Gao, Adam Fisch, and Danqi Chen. 2021. Making pre-trained language models better few-shot learners. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 3816–3830, Online. Association for Computational Linguistics.

Yu Gu, Robert Tinn, Hao Cheng, Michael Lucas, Naoto Usuyama, Xiaodong Liu, Tristan Naumann, Jianfeng Gao, and Hoifung Poon. 2020. Domain-specific language model pretraining for biomedical natural language processing.

Yu Gu, Robert Tinn, Hao Cheng, Michael Lucas, Naoto Usuyama, Xiaodong Liu, Tristan Naumann, Jianfeng Gao, and Hoifung Poon. 2021a. Domain-specific language model pretraining for biomedical natural language processing. *ACM Transactions on Computing for Healthcare (HEALTH)*, 3(1):1–23.

Yuxian Gu, Xu Han, Zhiyuan Liu, and Minlie Huang. 2021b. Ppt: Pre-trained prompt tuning for few-shot learning. *arXiv preprint arXiv:2109.04332*.

Kexin Huang, Jaan Altosaar, and Rajesh Ranganath. 2019. Clinicalbert: Modeling clinical notes and predicting hospital readmission. *arXiv preprint arXiv:1904.05342*.

Alistair EW Johnson, Tom J Pollard, Lu Shen, Li-wei H Lehman, Mengling Feng, Mohammad Ghassemi, Benjamin Moody, Peter Szolovits, Leo Anthony Celi, and Roger G Mark. 2016. Mimic-iii, a freely accessible critical care database. *Scientific data*, 3(1):1–9.

Jinhyuk Lee, Wonjin Yoon, Sungdong Kim, Donghyeon Kim, Sunkyu Kim, Chan Ho So, and Jaewoo Kang. 2020. Biobert: a pre-trained biomedical language representation model for biomedical text mining. *Bioinformatics*, 36(4):1234–1240.

Patrick Lewis, Myle Ott, Jingfei Du, and Veselin Stoyanov. 2020. Pretrained language models for biomedical and clinical tasks: Understanding and extending the state-of-the-art. In *Proceedings of the 3rd Clinical Natural Language Processing Workshop*, pages 146–157.

Bohan Li, Hao Zhou, Junxian He, Mingxuan Wang, Yiming Yang, and Lei Li. 2020. On the sentence embeddings from pre-trained language models. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 9119–9130, Online. Association for Computational Linguistics.

Xiang Lisa Li and Percy Liang. 2021. Prefix-tuning: Optimizing continuous prompts for generation. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 4582–4597, Online. Association for Computational Linguistics.

Pengfei Liu, Weizhe Yuan, Jinlan Fu, Zhengbao Jiang, Hiroaki Hayashi, and Graham Neubig. 2021a. Pre-train, prompt, and predict: A systematic survey of prompting methods in natural language processing. *arXiv preprint arXiv:2107.13586*.

Xiao Liu, Kaixuan Ji, Yicheng Fu, Zhengxiao Du, Zhilin Yang, and Jie Tang. 2021b. P-tuning v2: Prompt tuning can be comparable to fine-tuning universally across scales and tasks. *arXiv preprint arXiv:2110.07602*.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Roberta: A robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692*.

Tom McCoy, Ellie Pavlick, and Tal Linzen. 2019. Right for the wrong reasons: Diagnosing syntactic heuristics in natural language inference. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 3428–3448.

George Michalopoulos, Yuanxin Wang, Hussam Kaka, Helen Chen, and Alexander Wong. 2021. Umlsbert: Clinical domain knowledge augmentation of contextual embeddings using the unified medical language system metathesaurus. In *Proceedings of the 2021*

Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, pages 1744–1753.

Yifan Peng, Shankai Yan, and Zhiyong Lu. 2019. Transfer learning in biomedical natural language processing: An evaluation of bert and elmo on ten benchmarking datasets. In *Proceedings of the 18th BioNLP Workshop and Shared Task*, pages 58–65.

Matthew E Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer. 2018. Deep contextualized word representations. In *Proceedings of NAACL-HLT*, pages 2227–2237.

Xipeng Qiu, Tianxiang Sun, Yige Xu, Yunfan Shao, Ning Dai, and Xuanjing Huang. 2020. Pre-trained models for natural language processing: A survey. *Science China Technological Sciences*, pages 1–26.

Alexey Romanov and Chaitanya Shivade. 2018. Lessons from natural language inference in the clinical domain. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 1586–1596.

Timo Schick and Hinrich Schütze. 2020. Rare words: A major problem for contextualized embeddings and how to fix it by attentive mimicking. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pages 8766–8774.

Timo Schick and Hinrich Schütze. 2021. Exploiting cloze-questions for few-shot text classification and natural language inference. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*, pages 255–269.

Mike Schuster and Kaisuke Nakajima. 2012. Japanese and korean voice search. In *2012 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 5149–5152.

Rico Sennrich, Barry Haddow, and Alexandra Birch. 2016. Neural machine translation of rare words with subword units. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1715–1725.

Hoo-Chang Shin, Yang Zhang, Evelina Bakhturina, Raul Puri, Mostofa Patwary, Mohammad Shoeybi, and Raghav Mani. 2020a. Bio-megatron: Larger biomedical domain language model. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 4700–4706.

Taylor Shin, Yasaman Razeghi, Robert L Logan IV, Eric Wallace, and Sameer Singh. 2020b. Eliciting knowledge from language models using automatically generated prompts. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 4222–4235.

Simon Šuster, Stéphan Tulkens, and Walter Daelemans. 2017. A short review of ethical challenges in clinical natural language processing. *arXiv preprint arXiv:1703.10090*.

Yanshan Wang, Naveed Afzal, Sunyang Fu, Liwei Wang, Feichen Shen, Majid Rastegar-Mojarad, and Hongfang Liu. 2020. Medsts: a resource for clinical semantic textual similarity. *Language Resources and Evaluation*, 54(1):57–72.

Jason Wei, Maarten Bosma, Vincent Y Zhao, Kelvin Guu, Adams Wei Yu, Brian Lester, Nan Du, Andrew M Dai, and Quoc V Le. 2021. Finetuned language models are zero-shot learners. *arXiv preprint arXiv:2109.01652*.

Qiyu Wu, Chen Xing, Yatao Li, Guolin Ke, Di He, and Tie-Yan Liu. 2020. Taking notes on the fly helps bert pre-training. *arXiv preprint arXiv:2008.01466*.

Yuanmeng Yan, Rumei Li, Sirui Wang, Fuzheng Zhang, Wei Wu, and Weiran Xu. 2021. ConSERT: A contrastive framework for self-supervised sentence representation transfer. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 5065–5075, Online. Association for Computational Linguistics.

Xi Yang, Xing He, Hansi Zhang, Yinghan Ma, Jiang Bian, Yonghui Wu, et al. 2020. Measurement of semantic textual similarity in clinical texts: comparison of transformer-based models. *JMIR medical informatics*, 8(11):e19735.

Wenhao Yu, Chenguang Zhu, Yuwei Fang, Donghan Yu, Shuohang Wang, Yichong Xu, Michael Zeng, and Meng Jiang. 2021. Dict-bert: Enhancing language model pre-training with dictionary. *arXiv preprint arXiv:2110.06490*.

Taolin Zhang, Zerui Cai, Chengyu Wang, Minghui Qiu, Bite Yang, and Xiaofeng He. 2021. Smedbert: A knowledge-enhanced pre-trained language model with structured semantics for medical text mining. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 5882–5893.

Sendong Zhao, Chang Su, Zhiyong Lu, and Fei Wang. 2021. Recent advances in biomedical literature mining. *Briefings in Bioinformatics*, 22(3):bbaa057.

George Kingsley Zipf. 2016. *Human behavior and the principle of least effort: An introduction to human ecology*. Ravenio Books.

Self-Supervised Intermediate Fine-Tuning of Biomedical Language Models for Interpreting Patient Case Descriptions

Israa Alghanmi[*], **Luis Espinosa-Anke**[*◇], **Steven Schockaert**[*]
[*]CardiffNLP, Cardiff University, UK
[◇]AMPLYFI, UK
{alghanmiia, espinosa-ankel, schockaerts1}@cardiff.ac.uk

Abstract

Interpreting patient case descriptions has emerged as a challenging problem for biomedical NLP, where the aim is typically to predict diagnoses, to recommended treatments, or to answer questions about cases more generally. Previous work has found that biomedical language models often lack the knowledge that is needed for such tasks. In this paper, we aim to improve their performance through a self-supervised intermediate fine-tuning strategy based on PubMed abstracts. Our solution builds on the observation that many of these abstracts are case reports, and thus essentially patient case descriptions. As a general strategy, we propose to fine-tune biomedical language models on the task of predicting masked medical concepts from such abstracts. We find that the success of this strategy crucially depends on the selection of the medical concepts to be masked. By ensuring that these concepts are sufficiently salient, we can substantially boost the performance of biomedical language models, achieving state-of-the-art results on two benchmarks.

1 Introduction

Natural Language Processing (NLP) in the biomedical domain poses a number of particular challenges. For this reason, several Language Models (LMs) that are specialised towards the biomedical domain have been proposed, including BioBERT (Lee et al., 2020), ClinicalBERT (Alsentzer et al., 2019), SciBERT (Beltagy et al., 2019a), and PubMedBERT (Gu et al., 2021). Recent work has focused on analysing the capabilities of such models (Jin et al., 2019; Alghanmi et al., 2021; Sung et al., 2021) and enhancing them further (He et al., 2020b; Yuan et al., 2021; Zhang et al., 2021; Fei et al., 2021). Broadly speaking, biomedical LMs have proven successful in capturing the meaning of specialised terminology, but they have been far less successful in enabling medical reasoning, e.g. for predicting a likely diagnosis from a given patient case description. Currently, the main strategies for alleviating this latter issue have centered on incorporating structured knowledge, especially in the form of knowledge graphs. For example, Meng et al. (2021) proposed a method to integrate a large biomedical knowledge graph into a language model through the use of adapters (Pfeiffer et al., 2020), while Zhang et al. (2022) used graph neural networks to jointly reason about language model outputs and knowledge graphs.

Question: A 38-year-old woman comes to the emergency department because of progressive headache, blurry vision, and nausea for 1 day. Four days ago, she was diagnosed with a right middle ear infection. She appears lethargic. Her temperature is 39.1°C (102.3°F), and blood pressure is 148/95 mm Hg. Ophthalmologic examination shows bilateral swelling of the optic disc. The corneal reflex in the right eye is absent. Sensation to touch is reduced on the upper right side of the face. Serum studies show increased concentrations of fibrin degradation products. Which of the following is the most likely diagnosis?
(A) Cerebral venous thrombosis
(B) Hypertensive emergency
(C) Subarachnoid hemorrhage
(D) Viral meningitis

Table 1: Example of a question from MedQA, along with the answer candidates.

We focus on the task of interpreting patient case descriptions. To illustrate this task, Table 1 shows a question from the MedQA benchmark (Jin et al., 2021). In this context, the aim is typically to infer a diagnosis or to recommend a treatment. This is highly challenging, even for biomedical language models, because many pieces of information may need to be combined to find the right answer, and often some degree of clinical judgment is needed. Accordingly, the performance of state-of-the-art biomedical language models remains rather low for benchmarks such as MedQA. We argue that this can, to some extent, be explained by the fact that

Proceedings of the 29th International Conference on Computational Linguistics, pages 1432–1441
October 12–17, 2022.

interpreting patient case descriptions is a paragraph-level task, whereas the standard masked language modelling objective encourages the model to primarily focus on sentence-level context.

Ideally, biomedical language models for interpreting patient case descriptions would be pre-trained on a task that involves predicting diagnoses, or other salient aspects of these patient cases. Unfortunately, beyond the training fragment of benchmarks such as MedQA, such labelled data is not readily available. As an alternative, we propose to generate a pseudo-labelled dataset, based on the heuristic that whenever a case descriptions mentions a disease, it is likely (although by no means guaranteed) that this disease is a valid diagnosis, and similar for other medical concepts such as treatments. To get access to a large set of case descriptions, we rely on abstracts of published case reports. In particular, starting from a collection of PubMed abstracts, we first use a simple heuristic to identify those that are likely to correspond to case reports. Given a case report that mentions some disease, we then fine-tune the language model on the task of predicting that disease. Note that the target disease is masked, as the task would otherwise be trivial. The pre-training task is formulated as a binary classification problem, i.e. given a patient description and a disease, is that disease the correct diagnosis (or more precisely, is it the disease that was masked). This formulation has the advantage that the input format is similar to that of multiple-choice question answering (QA) and natural language inference (NLI). Beyond diseases, we also experiment with predicting masked treatments. Similar to the usual masked language modelling objective, our pre-training task involves making predictions about masked text spans. However, due to the fact that we specifically mask diseases and treatments, we hypothesize that this will improve the model's ability to take the whole case description into account when making predictions. Finally, note that we consider this to be an intermediate fine-tuning step. In other words, we start from a state-of-the-art biomedical language model, which is then fine-tuned on the proposed task, before finally being fine-tuned on a downstream task.

We find that this intermediate fine-tuning leads to substantial improvements in downstream tasks, even when using a biomedical LM that was already pre-trained on PubMed. To some extent, this comes from the fact that we specifically fine-tune the model on case reports. However, this in itself is not sufficient. To achieve good results, we find that a careful selection of the target concepts is needed. For instance, strong results are obtained when only masking medical treatments. When masking diseases, the improvements over the baseline are sometimes smaller. This is surprising, given that most questions in the considered benchmarks are about diagnosing diseases. Upon closer inspection, the under-performance of strategies that rely on masking diseases appears to be related to the fact that diseases can be mentioned for two common reasons: (i) because the patient has been diagnosed with that disease, which is the case that underpins the intuition behind our proposed approach, or (ii) because the disease is relevant to the medical history of the patient. In the latter case, only a small part of the abstract may be relevant to the disease, which hampers the extent to which the model learns to focus on the case description as a whole. To address this issue, we propose to split abstracts in which multiple diseases are mentioned. Despite the simplicity of the overall approach, our fine-tuning strategies enable significant improvements over the current state-of-the-art in two benchmarks that are focused on patient case descriptions: MedQA (Jin et al., 2021) and DisKnE (Alghanmi et al., 2021). [1]

2 Related Work

The standard paradigm in NLP at the moment is to fine-tune a pre-trained LM, such as BERT (Devlin et al., 2018), on task-specific training data. However, it has been observed that adding an intermediate step, where the LM is first fine-tuned on a different task, for which training data is more abundant, can be highly beneficial (Phang et al., 2018, 2020; Oğuz et al., 2021; Park and Caragea, 2020; Poth et al., 2021). Several works have investigated the role of intermediate tasks, in particular with the aim of analyzing when and why results improve (Pruksachatkun et al., 2020; Chang and Lu, 2021).

For the biomedical domain, one strategy has been to rely on transfer learning from general-domain tasks. For instance, Soni and Roberts (2020) use general-domain question answering for intermediate training, to improve a clinical question answering system. Another strategy has been to rely on different, but related tasks, such as pre-

[1] Code and data are available at: https://github.com/israa-alghanmi/Intermediate-FT-Biomedical-LMs

training on natural language inference to develop a question answering system (Jeong et al., 2020). Furthermore, several authors have proposed techniques for infusing the knowledge from biomedical knowledge graphs into LMs (He et al., 2020a; Meng et al., 2021; Jha and Zhang, 2022). More closely related to our approach, He et al. (2020c) propose a strategy which relies on the structure of Wikipedia to infuse knowledge about diseases. For instance, to teach the model about how diseases are treated, they rely on the fact that disease-centric Wikipedia articles tend to have a section called *Treatment*. They then combine the content of that section with a generated question-style sentence mentioning the aspect considered (i.e. treatment in this case) and a masked disease. However, rather than infusing encylopedic knowledge, our aim is to teach LMs to interpret patient case descriptions. Another related approach was introduced by Pergola et al. (2021), who propose to fine-tune a biomedical language model by using a masked language modelling objective which is modified such that only biomedical concepts are masked. This approach has some similarities with our work, e.g. the idea of masking biomedical concepts as an intermediate fine-tuning task, but there are also some clear differences. First, we formulate our task as a binary classification problem, rather than masked language modelling. Moreover, we specifically target diseases and treatments, and we only mask one concept at a time (although all occurrences of that concept are masked). Finally, since we focus on paragraph-level understanding, we pay particular attention to how these input paragraphs can be selected. As we will see, each of these differences has a clear impact on the empirical results.

3 Proposed Method

We consider the problem of making predictions from patient case descriptions. For instance, given a description that lists symptoms and other information about the patient (e.g. gender, age, and medical history), we would like to infer the corresponding diagnosis or to recommend suitable treatments. We are specifically interested in the potential of using freely available case reports from the medical literature to improve the ability of standard biomedical LMs to make such predictions. In Section 3.1, we first explain our overall strategy. Subsequently, in Section 3.2 we describe the specific variants that we included in our analysis.

3.1 Overall Strategy

Our aim is to design an intermediate fine-tuning task for specialising biomedical LMs towards the task of interpreting patient case descriptions. This fine-tuning task relies on passages from PubMedDS (Vashishth et al., 2021), a corpus which primarily consists of abstracts from PubMed. First, we split the abstracts into passages of up to 250 words, to address the limitations on input length of BERT-based LMs. Next, we aim to identify those passages that contain a case report, describing a specific patient rather than more general findings. To this end, we rely on the simple but effective heuristic that case reports often mention the age of the patient. In particular, we select those passages that contain at least one keyword from the following list: *year-old male*, *year-old female*, *year-old boy*, *year-old girl*, *year-old woman*, *year-old man*. Let us write $\mathcal{D}$ for the resulting corpus, i.e. the set of passages that contain at least one of the aforementioned keywords. Subsequently, we determine which medical concepts are mentioned in the passages from $\mathcal{D}$. To this end, we use QuickUMLS (Soldaini and Goharian, 2016) with UMLS-2020AA to identify both the spans and the semantic types (e.g. diseases, treatments) of the mentioned concepts. Finally, we create positive training examples of the form (P,C), where C is a medical concept, and P is a passage from $\mathcal{D}$ in which all mentions of C have been replaced by a single <*mask*> token. To generate negative training examples, we simply replace the medical concept C by another concept, as explained below. A given example (*passage*,*concept*) is encoded as follows: "<*cls*> *passage* <*sep*> *concept*", mimicking the input format that is typically used for question answering and natural language inference models. The LM is fine-tuned on these examples using a standard cross-entropy loss.

3.2 Training Strategies

We now describe the different variants that we considered. These variants primarily differ in the kinds of medical concepts that are selected as target concepts. Across all variants, we never mask the concept *disorder*, as constructing training examples from such mentions was found to be highly detrimental, given its prevalence and generic meaning. For all variants, we attempt to balance the number of positive and negative examples. Table 2 provides an overview of the total number of training exam-

	#
AnyType	1,011,482
SpecificType	
– *diseases*	160,534
– *treatments*	2,460
SplitDis	100,225
OneDis	3,310

Table 2: The total number of training examples for each of the intermediate fine-tuning tasks (#).

ples arising from each of the following strategies.

AnyType We create a positive example for every medical concept that is found (with the exception of *disorder*). Note that passages typically mention several concepts, hence this strategy allows us to derive multiple positive examples from the same passage, each time masking a different concept. To construct negative examples, we corrupt positive examples by randomly selecting a concept from those that have been identified in the corpus, regardless of the semantic type.

SpecificType In this variant, we only construct training examples from medical concepts of particular types. Specifically, we have experimented with diseases and treatments. Negative examples are constructed by replacing the target concept with another concept of the same semantic type, i.e. diseases are replaced by diseases, and treatments are replaced by treatments.

SplitDis Many passages contain more than one disease, which may confuse the model. For instance, diseases which are mentioned as part of the patient history may only be loosely related to the rest of the case report. Since our aim is to train the model to make predictions based on the whole case description, in this variant, passages containing more than one disease are split into sub-passages. In particular, when constructing a positive example for a target disease d, we select the sub-passage which begins with the first sentence in which d is mentioned, and includes all the subsequent sentences, until we reach a sentence that mentions another disease (where this final sentence is excluded from the selected sub-passage). If the target disease is mentioned in a sentence that also contains another disease, it is excluded altogether. For illustration, training examples that were obtained with the *SplitDis* strategy are presented in Table 3.

OneDis Instead of splitting passages mentioning more than one disease into sub-passages, as with SplitDis, here we simply discard such passages. This results in a much smaller number of positive examples, but with stronger guarantees that the disease being masked is salient. In both this and the *SplitDis* method, negative examples are obtained by using randomly selected diseases.

4 Experiments

In this section, we empirically analyse the different variants of the intermediate fine-tuning strategy.

Evaluation Datasets We mainly focus on two benchmarks that are specifically focused on interpreting and reasoning about patient cases. First, we use **MedQA** (Jin et al., 2021), which is a multiple-choice question answering benchmark. The questions are taken from medical exams and are specifically asking about what can be inferred from a given patient case description. We use the English version of this dataset (USMLE). Results for this benchmark are reported in terms of accuracy (Acc). Second, we use **DisKnE** (Alghanmi et al., 2021), which has been derived from MedNLI (Romanov and Shivade, 2018). Therefore, to use DisKnE, a license and access to MedNLI is required. Instances of this benchmark consist of a patient case description and a disease, and the aim is to predict whether that disease can be inferred as diagnosis. This repurposing from the original MedNLI is of particular relevance to our experiments, given that many instances in MedNLI can be solved simply with linguistic knowledge. DisKnE contains a separate training-test split for each disease, and for each split, we consider the task of ranking all test cases, according to our confidence that the given target disease is a valid diagnosis. The results are averaged across all diseases and are reported in terms of Mean Average Precision (MAP). We use the medical-similar variant of the benchmark.

In addition, we also consider the English version of **HeadQA** (Vilares and Gómez-Rodríguez, 2019), as a more general healthcare-oriented QA dataset. This dataset contains a broad variety of healthcare questions, most of which do not involve patient descriptions. However, the questions are designed to require complex medical reasoning. As such, we use this benchmark to analyse whether our proposed approach may also benefit such settings.

SpecificType-treatments	The role of [MASK] in the treatment of a patient with a pure silent pituitary somatotroph carcinoma. To describe a case of a pure silent somatotroph pituitary carcinoma. We describe a 54-year-old female with a clinically nonfunctioning pituitary macroadenoma diagnosed 15 years earlier. The patient underwent transsphenoidal surgery and no visible tumor remnant was observed for 6 years. A magnetic resonance imaging (MRI) detected the recurrence of a 1.2 × 1.5 cm macroadenoma. The patient was submitted to conventional radiotherapy (4500 cGy), and the tumor volume remained stable for 7 years. Then, an MRI revealed a slight increase in tumor size, and 2 years later, a subsequent MRI detected a very large, invasive pituitary mass. The patient was resubmitted to transsphenoidal surgery, and the histopathological examination showed diffuse positivity for growth hormone (GH). The nadir GH level during an oral glucose tolerance test was 0. 06 ng/mL, and the pre- and postoperative insulin like growth factor type I (IGF-I) levels were within the normal range. Abdominal, chest, brain, and spine MRI showed multiple small and hypervascular liver and bone lesions suggestive of metastases. Liver biopsy confirmed metastasis of GH-producing pituitary carcinoma. The patient has been treated with [MASK] and zoledronic acid for 7 months and with octreotide long-acting release (LAR) for 4 months. → *Temozolomide*
SpecificType-diseases	Intestinal cholesterol absorption inhibitor ezetimibe added to cholestyramine for sitosterolemia and xanthomatosis. Sitosterolemia is a rare, recessively inherited disorder characterized by increased absorption and delayed removal of noncholesterol sterols, which is associated with accelerated atherosclerosis, premature [MASK], hemolysis, and xanthomatosis. Treatments include low-sterol diet and bile salt-binding resins; however, these often do not reduce the xanthomatosis. We examined the effects of the intestinal cholesterol/phytosterol transporter inhibitor ezetimibe added to cholestyramine in a young female patient with sitosterolemia and associated xanthomatosis. The patient was an 11-year-old female with sitosterolemia presenting with prominent xanthomas in the subcutaneous tissue of both elbows who was receiving treatment with cholestyramine 2 g once daily. Bilateral carotid bruits were audible, and a grade II/VI systolic murmur was detected at the left upper sternal border. She also had a low platelet count of 111,000/microL. Ezetimibe 10 mg once daily was added to the patient's ongoing cholestyramine regimen, and she was evaluated for 1 year. The patient followed an unrestricted diet during the 1-year treatment period. After 1 year of treatment with ezetimibe added to ongoing cholestyramine therapy, the patient's plasma sitosterol and campesterol levels decreased by approximately 50. → *coronary artery disease*
SplitDis	After initial improvement artificial ventilation had to be be gun on day 3 because of an acute [MASK], diagnosed both clinically and radiologically. Despite additional antiviral and intensive medical treatment he died on day 11. → *respiratory distress syndrome*
	Traumatic [MASK] present diagnostic and therapeutic challenges. Owing to their fragile nature, endovascular intervention has become the first-line treatment; however, direct surgery has an advantage in certain cases. → *intracranial aneurysms*
	A fluoroscopic sniff test demonstrated diaphragmatic dysfunction and pulmonary function tests revealed [MASK] with evidence of neuromuscular etiology. → *restrictive pulmonary disease*

Table 3: Examples obtained with the different variants of the proposed strategies.

Some questions in this dataset require interpreting images. As this is beyond the scope of the paper, we discard all questions involving images for our experiments. This resulted in a total number of 2589 questions for training, 1336 for validation, and 2675 for testing.

Setup We use four pre-trained LMs for the baselines and main experiments:

- the cased version of the standard $\text{BERT}_{\text{base}}$ (Devlin et al., 2019);
- the cased version of SciBERT (Beltagy et al., 2019b);
- the cased version of ClinicalBERT (Alsentzer et al., 2019) that was trained on MIMIC-III while being initialized from BioBERT (Lee et al., 2020);
- the PubMedBERT model (Gu et al., 2021) that was trained from scratch on full-length PubMed articles as well as abstracts.

As a baseline, we directly fine-tune the models on the training data from the downstream task. For the other configurations, we first fine-tune the models on the proposed intermediate task.

We use the official training, validation, and test splits for each dataset, with the exception that we excluded questions with images for HeadQA.

Training Details We use the same settings and hyper-parameters for all datasets. For fine-tuning the models on the target task, we set the batch size to 8, the number of epochs to 4 and the learning rate to 2e-5. For the intermediate fine-tuning step, we again set the batch size to 8 and the learning rate to 2e-5. Regarding the number of epochs for intermediate fine-tuning, we note that the number of training examples varies greatly across the different variants. For this reason, and to mitigate the potential for catastrophic forgetting, we tuned the number of epochs, choosing from $\{2, 3, 4\}$, based on the development split of the downstream task.

Limitations Our method relies on an automated extraction tool for identifying the target medical concepts, which will inevitably lead to some noisy training examples. For example, *SplitDis* and *OneDis* rely on the assumption that we can detect all mentions of diseases in the text. More generally, regardless of performance, the predictions of a biomedical LM can clearly not be relied upon for diagnosing patients or recommending treatments in a clinical setting. Our purpose in studying these models is rather because a deeper understanding of patient records would make it possible to improve retrieval systems (e.g. suggesting relevant case reports to a clinician handling an unusual patient) or to identify hypotheses for medical research (e.g. by inducing patterns from large sets of case reports).

4.1 Results

Tables 4, 5 and 6 summarize our results. As can be seen, PubMedBERT clearly outperforms the other language models. In general, most variants of the intermediate fine-tuning tasks lead to clear improvements over the baselines. A clear and remarkable conclusion that can be observed for all benchmarks is that the type of intermediate fine-tuning data appears to be much more important than the number of training examples. For instance, the version of *SpecificType* which only uses treatments achieves the best overall results, outperforming the previous state-of-the-art for MedQA and achieving among the strongest results for both DisKnE and HeadQA. This is surprising, both because of the small number of training examples we can generate for this variant and because of the focus on diseases in DisKnE and many of the MedQA and HeadQA questions.

For MedQA, *SpecificType* with treatments outperforms the previous state-of-the-art (Zhang et al., 2022) by 1.9 percentage points, despite not relying on any structured knowledge graphs. Note that DisKnE is a recent benchmark, for which the only reported results thus far were obtained from simply fine-tuning biomedical LMs. These existing results were reported prior to the introduction of PubMedBERT, which outperforms these published results. The *OneDis* variant performs well for DisKnE, despite the low number of corresponding training examples. For MedQA, *SplitDis* outperforms *SpecificType* with diseases (with the exception of BERT), which supports the idea that simply masking diseases can lead to training examples that are too noisy. While HeadQA is not particularly focused on patients case descriptions, we still see consistent improvements over the baselines with *SpecificType*, *SplitDis* and *OneDis*, although the improvements are somewhat smaller than those for MedQA and DisKnE.

We can see that our proposed strategy outperforms the baselines for each of the different language models, with the exception of SciBERT with DisKnE. However, there are some differences between the language models in terms of which variant of our method performs best. For MedQA, for instance, we can see that *SpecificType* with diseases is highly competitive for BERT and ClinicalBERT (compared to the other variants for these language models). As these are the language models that are least adapted to the considered task, we can indeed expect that more pre-training data might be needed for these models. This can explain the relative success of *SpecificType* with diseases and *SplitDis*, given that these are associated with a larger number of training examples.

	BERT	ClinicalBERT	SciBERT	PubMedBERT
Baseline	27.8	29.1	29.2	35.5
AnyType	**28.2**	31.2	32.7	36.5
SpecificType				
– *diseases*	**28.2**	31.5	30.4	38.0
– *treatments*	27.8	31.0	**34.5**	**40.4**
SplitDis	27.7	**31.8**	33.4	38.7
OneDis	27.0	29.6	33.3	35.6

Table 4: Results for MedQA in terms of Accuracy.

	BERT	ClinicalBERT	SciBERT	PubMedBERT
Baseline	57.0	67.5	**69.2**	69.7
AnyType	**64.2**	71.6	68.8	71.9
SpecificType				
– *diseases*	60.2	70.0	67.0	72.9
– *treatments*	57.5	67.5	68.3	73.6
SplitDis	58.3	**74.1**	68.1	72.2
OneDis	64.0	68.2	66.2	**74.4**

Table 5: Results for DisKnE in terms of Mean Average Precision (MAP).

4.2 Analysis

Table 7 shows the results of some variants of the *SpecificType* with diseases and *SplitDis* strategies, as explained next. We use PubMedBERT for these experiments, as this model achieved the best results in the main experiments. We focus on the MedQA benchmark as this is the most representative benchmark for our problem setting.

Frequent vs Rare We analyze whether there is any advantage in focusing specifically on common diseases, or conversely, in focusing on rare diseases. Table 7 shows the results of two variants of Speci-

	BERT	ClinicalBERT	SciBERT	PubMedBERT
Baseline	28.8	29.3	32.8	39.5
AnyType	29.3	30.0	31.7	39.1
SpecificType				
– *diseases*	29.8	30.1	34.5	**41.8**
– *treatments*	**30.3**	**31.1**	**35.7**	41.0
SplitDis	29.8	29.6	32.6	40.7
OneDis	29.7	29.8	34.0	40.8

Table 6: Results for HeadQA in terms of Accuracy.

		#	**MedQA** (Acc)
SpecificType	Most-Frequent	49,816	36.8
	Least-Frequent	8,466	38.0
	Most-General	7,229	36.6
	Most-Specific	8,778	36.9
	Treatment-Case-Dis	6,934	38.2
SplitDis	Most-Similar	1,858	37.7
	Least-Similar	1,870	36.7
	SplitDis+Def	105,952	37.7
	Treatment-Case-Dis	2,430	38.4

Table 7: Analysis results for MedQA (Accuracy). We also report the total number of training examples for each of the intermediate fine-tuning tasks (#). Results were obtained using PubMedBERT.

ficType, called *Most-Frequent* and *Least-Frequent*. The former only considers training examples, for the intermediate fine-tuning task, involving the 50 diseases which are most common in our corpus of case reports. Similarly, the *Least-Frequent* variant only considers the 5000 least frequent diseases. *Least-Frequent* achieves the best result, despite involving far fewer training examples than *Most-Frequent*. The results of both variants are either below or similar to those with the full set of diseases in Table 4.

General vs Specific Rather than selecting diseases based on their number of occurrences, here we investigate the effect of choosing diseases based on whether they are general or specific, in terms of the level at which they appear in the SNOMED CT hierarchies (Stearns et al., 2001). Specifically, for the *Most-General* variant, we only consider diseases with fewer than 5 ancestors in SNOMED CT. For the *Most-Specific* variant, we only consider diseases with at least 30 ancestors. We find that both variants of *SpecificType* perform similarly.

Similar vs Different We explore a setting in which only case reports about diseases similar to "heart disease" are provided during training. Specifically, we use cui2vec (Beam et al., 2020) to identify the 50 most similar diseases that occur at least once in our corpus of case reports. We then consider a variant of *SplitDis* in which only passages with *heart disease*, or any of the 50 similar diseases, occurs as the target disease. Our aim in this experiment is to see whether training on one type of diseases is sufficient to obtain good results. Furthermore, we may also assume that because the resulting corpus only involves similar diseases, the model is forced to focus on more subtle details in the paragraphs, and might thus improve as a result. To test this hypothesis, we also consider the variant *Least-Similar*, where we instead use the diseases that are least similar to *heart disease*. Rather than fixing the number of diseases at 50, in this case we chose the number to ensure a similar number of training examples as for *Most-Similar*. The results for both variants are below those of the standard *SplitDis* variant. However, we can see that *Most-Similar* clearly outperforms *Least-Similar*.

Adding Definitions We analyse the usefulness of UMLS definitions. Specifically, we augment the SplitDis training examples with examples of the form (def, dis), where def is the UMLS definition of a disease, and dis is the corresponding disease. Negative examples are again created by replacing the target disease with a randomly chosen other disease. The results in Table 7 show that adding definitions does not improve the results.

Diseases in Treatment Cases The good performance of the SpecificType variant with treatments, despite the small number of training examples we have for that setting, is one of the most surprising findings from the main experiments. Here we analyse whether this might be related to the quality of the case reports that were selected in that setting, i.e. the case reports that mention a treatment. To this end, we consider all such case reports, but instead of using the treatments as the target concepts, we instead focus on diseases. In other words, we use the SpecificType setting for diseases, but applied to

	MedQA (Acc)
MLM-RandomMask	
– *SplitDis*	36.4
– *SpecificType: treatments*	35.2
MLM-SpecificMask	
– *SplitDis*	37.6
– *SpecificType: treatments*	38.5
Random-Abstracts	
– *SplitDis*	38.2
– *SpecificType: treatments*	37.6
No Mask	
– *SplitDis*	36.7
– *SpecificType: treatments*	37.8
Remove-Sent (treatments)	38.9

Table 8: Ablation results for MedQA in terms of Accuracy. Results were obtained using PubMedBERT.

the case reports that mention treatments. We also consider a variant in which the *SplitDis* setting is applied to these case reports. The results in Table 8, shown as *Treatment-Case-Dis*, reveal that this variant still underperforms the *SpecificCase* variant with treatments.

4.3 Ablation Experiments

In this section, we analyse the importance of a number of our design choices. We again focus on PubMedBERT and MedQA. We specifically consider the SplitDis and SpecificType with treatments, as these yielded the best results in the main experiments. The results are summarized in Table 8.

Masked Language Modelling We experimented with two variants of the masked language modelling (MLM) objective for the intermediate fine-tuning task. For the *MLM-RandomMask* variant, we randomly mask tokens, following the standard approach that is used for LM pre-training. For the *MLM-SpecificMask* variant, we specifically mask the tokens corresponding to diseases (for the SplitDis setting) or treatments (for the SpecificType setting). The results show that our approach outperforms both MLM strategies, while *MLM-SpecificMask* outperforms *MLM-RandomMask*.

Random Abstracts vs Case Reports We analyse the importance of specifically focusing on case reports. In the *Random-Abstracts* variant, rather than targeting abstracts which are likely to correspond to case reports, we consider a set of 60,000 randomly sampled abstracts from PubMedDS. We then use our SplitDis and SpecificType settings to construct the examples. The results in Table 8 show that using randomly chosen abstracts leads to worse results, compared to our standard setting.

Masking vs not Masking We consider a variant of the method in which the original passage is used, i.e. where we do not replace occurrences of the target disease with a <*mask*> token. The results in Table 8 clearly shows that masking is essential to achieve the best results. Nonetheless, even without masking we obtain results that are clearly better than those of the baseline (i.e. PubMedBERT without intermediate fine-tuning).

Masking vs Removing Sentences Instead of replacing the target concept with a <*mask*> token, here we remove the entire sentence in which this concept is mentioned. For this variant, called *Remove-Sent*, we only consider the SpecificType setting (with treatments), as using SplitDis would result in too few examples, given that several SplitDis examples consist of a single sentence. The results show that removing the sentence underperforms masking the concept.

5 Conclusions

We have proposed a strategy for intermediate fine-tuning of biomedical language models, to improve their ability to interpret patient case descriptions. The core of our strategy is to exploit abstracts of case reports found in the literature, as a surrogate of patient case descriptions, and to rely on the heuristic that diseases and treatments that are mentioned in such abstracts are likely to correspond to diagnoses and recommendations, respectively. Despite its conceptual simplicity and without the cost of manual annotation, this approach was found to lead to clear performance gains, setting a new state-of-the-art in MedQA and DisKnE, while also benefitting more diverse datasets such as HeadQA.

References

Israa Alghanmi, Luis Espinosa Anke, and Steven Schockaert. 2021. Probing pre-trained language models for disease knowledge. In *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*, pages 3023–3033, Online. Association for Computational Linguistics.

Emily Alsentzer, John Murphy, William Boag, Wei-Hung Weng, Di Jindi, Tristan Naumann, and Matthew McDermott. 2019. Publicly available clinical BERT embeddings. In *Proceedings of the 2nd Clinical Natural Language Processing Workshop*, pages 72–78, Minneapolis, Minnesota, USA. Association for Computational Linguistics.

Andrew L Beam, Benjamin Kompa, Allen Schmaltz, Inbar Fried, Griffin Weber, Nathan Palmer, Xu Shi, Tianxi Cai, and Isaac S Kohane. 2020. Clinical concept embeddings learned from massive sources of multimodal medical data. In *Pacific Symposium on Biocomputing*, volume 25, pages 295–306.

Iz Beltagy, Kyle Lo, and Arman Cohan. 2019a. SciBERT: A pretrained language model for scientific text. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing*, pages 3613–3618.

Iz Beltagy, Kyle Lo, and Arman Cohan. 2019b. SciBERT: A pretrained language model for scientific text. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 3615–3620, Hong Kong, China. Association for Computational Linguistics.

Ting-Yun Chang and Chi-Jen Lu. 2021. Rethinking why intermediate-task fine-tuning works. *arXiv preprint arXiv:2108.11696*.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, Minneapolis, Minnesota. Association for Computational Linguistics.

Hao Fei, Yafeng Ren, Yue Zhang, Donghong Ji, and Xiaohui Liang. 2021. Enriching contextualized language model from knowledge graph for biomedical information extraction. *Briefings in bioinformatics*, 22(3):bbaa110.

Yu Gu, Robert Tinn, Hao Cheng, Michael Lucas, Naoto Usuyama, Xiaodong Liu, Tristan Naumann, Jianfeng Gao, and Hoifung Poon. 2021. Domain-specific language model pretraining for biomedical natural language processing. *ACM Transactions on Computing for Healthcare (HEALTH)*, 3(1):1–23.

Bin He, Di Zhou, Jinghui Xiao, Xin Jiang, Qun Liu, Nicholas Jing Yuan, and Tong Xu. 2020a. Bert-mk: Integrating graph contextualized knowledge into pre-trained language models. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 2281–2290.

Yun He, Ziwei Zhu, Yin Zhang, Qin Chen, and James Caverlee. 2020b. Infusing disease knowledge into bert for health question answering, medical inference and disease name recognition. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 4604–4614.

Yun He, Ziwei Zhu, Yin Zhang, Qin Chen, and James Caverlee. 2020c. Infusing disease knowledge into BERT for health question answering, medical inference and disease name recognition. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing*, pages 4604–4614.

Minbyul Jeong, Mujeen Sung, Gangwoo Kim, Donghyeon Kim, Wonjin Yoon, Jaehyo Yoo, and Jaewoo Kang. 2020. Transferability of natural language inference to biomedical question answering. *arXiv preprint arXiv:2007.00217*.

Kishlay Jha and Aidong Zhang. 2022. Continual knowledge infusion into pre-trained biomedical language models. *Bioinformatics*, 38(2):494–502.

Di Jin, Eileen Pan, Nassim Oufattole, Wei-Hung Weng, Hanyi Fang, and Peter Szolovits. 2021. What disease does this patient have? a large-scale open domain question answering dataset from medical exams. *Applied Sciences*, 11(14):6421.

Qiao Jin, Bhuwan Dhingra, William Cohen, and Xinghua Lu. 2019. Probing biomedical embeddings from language models. In *Proceedings of the 3rd Workshop on Evaluating Vector Space Representations for NLP*, pages 82–89.

Jinhyuk Lee, Wonjin Yoon, Sungdong Kim, Donghyeon Kim, Sunkyu Kim, Chan Ho So, and Jaewoo Kang. 2020. Biobert: a pre-trained biomedical language representation model for biomedical text mining. *Bioinformatics*, 36(4):1234–1240.

Zaiqiao Meng, Fangyu Liu, Thomas Hikaru Clark, Ehsan Shareghi, and Nigel Collier. 2021. Mixture-of-partitions: Infusing large biomedical knowledge graphs into bert. *arXiv preprint arXiv:2109.04810*.

Barlas Oğuz, Kushal Lakhotia, Anchit Gupta, Patrick Lewis, Vladimir Karpukhin, Aleksandra Piktus, Xilun Chen, Sebastian Riedel, Wen-tau Yih, Sonal Gupta, et al. 2021. Domain-matched pre-training tasks for dense retrieval. *arXiv preprint arXiv:2107.13602*.

Seoyeon Park and Cornelia Caragea. 2020. Scientific keyphrase identification and classification by pre-trained language models intermediate task transfer learning. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 5409–5419, Barcelona, Spain (Online). International Committee on Computational Linguistics.

Gabriele Pergola, Elena Kochkina, Lin Gui, Maria Liakata, and Yulan He. 2021. Boosting low-resource biomedical qa via entity-aware masking strategies. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*, pages 1977–1985.

Jonas Pfeiffer, Andreas Rücklé, Clifton Poth, Aishwarya Kamath, Ivan Vulić, Sebastian Ruder, Kyunghyun Cho, and Iryna Gurevych. 2020. Adapterhub: A framework for adapting transformers. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 46–54.

Jason Phang, Iacer Calixto, Phu Mon Htut, Yada Pruksachatkun, Haokun Liu, Clara Vania, Katharina Kann, and Samuel R. Bowman. 2020. English intermediate-task training improves zero-shot cross-lingual transfer too. In *Proceedings of the 1st Conference of the Asia-Pacific Chapter of the Association for Computational Linguistics and the 10th International Joint Conference on Natural Language Processing*, pages 557–575, Suzhou, China. Association for Computational Linguistics.

Jason Phang, Thibault Févry, and Samuel R Bowman. 2018. Sentence encoders on stilts: Supplementary training on intermediate labeled-data tasks. *arXiv preprint arXiv:1811.01088*.

Clifton Poth, Jonas Pfeiffer, Andreas Rücklé, and Iryna Gurevych. 2021. What to pre-train on? efficient intermediate task selection. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 10585–10605.

Yada Pruksachatkun, Jason Phang, Haokun Liu, Phu Mon Htut, Xiaoyi Zhang, Richard Yuanzhe Pang, Clara Vania, Katharina Kann, and Samuel R. Bowman. 2020. Intermediate-task transfer learning with pretrained language models: When and why does it work? In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 5231–5247, Online. Association for Computational Linguistics.

Alexey Romanov and Chaitanya Shivade. 2018. Lessons from natural language inference in the clinical domain. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 1586–1596, Brussels, Belgium. Association for Computational Linguistics.

Luca Soldaini and Nazli Goharian. 2016. Quickumls: a fast, unsupervised approach for medical concept extraction. In *MedIR workshop, sigir*, pages 1–4.

Sarvesh Soni and Kirk Roberts. 2020. Evaluation of dataset selection for pre-training and fine-tuning transformer language models for clinical question answering. In *Proceedings of The 12th Language Resources and Evaluation Conference*, pages 5532–5538.

Michael Q Stearns, Colin Price, Kent A Spackman, and Amy Y Wang. 2001. Snomed clinical terms: overview of the development process and project status. In *Proceedings of the AMIA Symposium*, page 662. American Medical Informatics Association.

Mujeen Sung, Jinhyuk Lee, Sean Yi, Minji Jeon, Sungdong Kim, and Jaewoo Kang. 2021. Can language models be biomedical knowledge bases? In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 4723–4734.

Shikhar Vashishth, Denis Newman-Griffis, Rishabh Joshi, Ritam Dutt, and Carolyn P Rosé. 2021. Improving broad-coverage medical entity linking with semantic type prediction and large-scale datasets. *Journal of Biomedical Informatics*, 121:103880.

David Vilares and Carlos Gómez-Rodríguez. 2019. HEAD-QA: A healthcare dataset for complex reasoning. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 960–966, Florence, Italy. Association for Computational Linguistics.

Zheng Yuan, Yijia Liu, Chuanqi Tan, Songfang Huang, and Fei Huang. 2021. Improving biomedical pretrained language models with knowledge. In *Proceedings of the 20th Workshop on Biomedical Language Processing*, pages 180–190.

Taolin Zhang, Zerui Cai, Chengyu Wang, Minghui Qiu, Bite Yang, and Xiaofeng He. 2021. SMedBERT: A knowledge-enhanced pre-trained language model with structured semantics for medical text mining. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 5882–5893, Online. Association for Computational Linguistics.

Xikun Zhang, Antoine Bosselut, Michihiro Yasunaga, Hongyu Ren, Percy Liang, Christopher D Manning, and Jure Leskovec. 2022. GreaseLM: Graph reasoning enhanced language models for question answering. *arXiv preprint arXiv:2201.08860*.

Evaluating and Mitigating Inherent Linguistic Bias of African American English through Inference

Jamell Dacon* **Haochen Liu** **Jiliang Tang**
Michigan State University
{daconjam, liuhoac1, tangjili}@msu.edu

Abstract

Recent studies show that NLP models trained on standard English texts tend to produce biased outcomes against underrepresented English varieties. In this work, we conduct a pioneering study of the English variety use of African American English (AAE) in NLI task. First, we propose CODESWITCH, a greedy unidirectional morphosyntactically-informed rule-based translation method for data augmentation. Next, we use CODESWITCH to present a preliminary study to determine if demographic language features do in fact influence models to produce false predictions. Then, we conduct experiments on two popular datasets and propose two simple, yet effective and generalizable debiasing methods. Our findings show that NLI models (e.g. BERT) trained under our proposed frameworks outperform traditional large language models while maintaining or even improving the prediction performance. In addition, we intend to release CODESWITCH, in hopes of promoting dialectal language diversity in training data to both reduce the discriminatory societal impacts and improve model robustness of downstream NLP tasks.

1 Introduction

In recent years, social media has become a pivotal tool its users to express their thoughts, feelings, and opinions on similar interests (Dacon and Tang, 2021). Typically, Standard American English (SAE), a high-resource language (HRL) is often used in formal communication, whereas African American English (AAE)[1] is primarily spoken in the United States and is often heavily and explicitly used on social media platforms such as Twitter (Field et al., 2021; Blodgett et al., 2020).

In particular, AAE is an English language variety and can be considered to be a low-resource language (LRL) that is neither spoken by *all* African Americans or individuals who identify as BIPOC (Black, Indigenous, or People of Color), nor is it spoken *only* by African Americans or BIPOC individuals (Field et al., 2021; Dacon, 2022; Bland-Stewart, 2005). However, most dominant AAE speakers reside in diglossic communities and are able to *code-switch*, speaking both SAE and AAE. In linguistics, code-switching also referred to as language alternation is the ability of a speaker to alternate between two or more languages or language varieties within a particular conversation (Young and Barrett, 2018; Gardner-Chloros et al., 2009; DeBose, 1992; Young, 2009; Dacon, 2022). Thus, we refer to code-switching as switching among dialects, and/or language styles. For example, bidialectal AAE speakers are often able to code-switch between the SAE and both phonological and morphological language features of AAE while maintaining contextual intent.

Natural Language Understanding (NLU) is a subset of NLP, which enables human-computer interaction (HCI) by attempting to understand human language data such as text or speech, and communicate back to humans in their respective languages such as English, Spanish, etc., (Schank, 1972). Hence, we will focus on *inference*, which is an eminent area of study of NLU. In particular, Natural language inference (NLI), a subset of NLU, also known as Recognizing Textual Entailment (RTE) is a segment-level categorization task of understanding the inferential relationships between sentence pairs and anticipating whether they are entailing, contradictory, or neutral sentences (Bowman et al., 2015; Williams et al., 2018).

Generally, the term *implicit bias* is used to refer

*Corresponding author: Jamell Dacon

[1]This English language variety has had several names within the last decades such as African American Vernacular English (AAVE), African American Language (AAL), Black English, Ebonics, Non-standard English, Northern Negro English and Black English Vernacular (BEV) (Bailey et al., 1998; Green, 2002; Bland-Stewart, 2005; King, 2020). However, it is now commonly referred to as African American English (AAE), an English language variety.

Proceedings of the 29th International Conference on Computational Linguistics, pages 1442–1454
October 12–17, 2022.

to the unconscious preferential behaviors towards a certain demographic group such as age, race, ethnicity, gender, etc. (Liu et al., 2021; Tan et al., 2020a; Ribeiro et al., 2018). However, in this study, to examine the differences in language styles from different demographic groups, we refer to this type of predisposed language style bias as *inherent linguistic bias*. Although, both biases are very similar, there exists a subtle difference as linguistic bias specifically refers to an analysis of every aspect of a particular language (Zhou and Bansal, 2020). The existence of these biases in large language models (LLMs) such as mask language models (MLMs) generate language bias leading to potential harmful societal impacts inconveniencing members of LRL and diglossic communities who speak both standard languages and unrepresented dialects. This may increase feelings of marginalization and disenfranchisement (Liu et al., 2020a; Blodgett et al., 2020; Field et al., 2021).

Hence, in this work, we conduct a pioneering study of robustifying MLMs to minimize false predictions by introducing dialectal language diversity in training data to determine if MLMs learn to make predictions based on demographic language features, and proposing two debias methods to enhance NLI models to mitigate the presence of linguistic bias during the training process. We posit that it is vital for production-ready MLMs improve their robustness to produce minimal systemic biases against protected attributes such as *race* and *gender* and thus, reducing discriminatory societal impacts (Hovy and Spruit, 2016; Sharma et al., 2021; Liu et al., 2020a; Tan et al., 2020a).

Specifically, we aim to answer two research questions: (1) *How can we as NLP practitioners encourage dialectal language diversity in training data?*; (2) *Do pretrained MLMs make predictions based on demographic language features?*; and (3) *How can we measure fairness and mitigate such biases in order to ensure fairness in NLU.*
Our contributions include:

- CODESWITCH, a greedy unidirectional morphosyntactically-informed rule-based translation method for data augmentation to generate intent-and-semantically equivalent AAE examples by perturbing SAE examples.
- Two intent-and-semantically equivalent NLI dataset of AAE sentence pairs with a wide range of morphological syntactic features and dialect-specific vocabulary.
- A detailed human evaluation of our human annotators to ensure contextual accuracy of adversarial sentence pairs (see Appendix D for details).
- Two simple, yet effective debiasing methods to mitigate the inherent linguistic bias in NLI models, while maintaining or even improving their prediction performance.

2 Preliminaries

In this section, we introduce some preliminary knowledge about the problem under study. We first present the problem statement, and then describe two popular NLI datasets used in our research.

2.1 Problem Statement

We aim to investigate sentence representations of two linguistic systems of different demographic groups to demonstrate the existence of constitutional linguistic bias. To address the above research questions, we define two goals:

1. The first goal is to predict inferential relationships between paired sentences i.e., the second sentence is an entailment, contradiction, or neutral with respect to the first sentence.

2. The second goal is to debias the sentence representations obtained from the words in the given sentence. Specifically, we want the sentence representation to *only* include the semantic information, but not the language style, whether SAE or AAE. Therefore, we want the MLM to ignore the language style of each demographic group in order to make fair predictions.

Mitigating such linguistic biases can help develop robust MLMs for LRLs and dialectal languages more easily. Our main objective is to focus on dialectal language inclusivity, while using the benefit of large pretrained MLMs in order to improve model robustness of downstream tasks of NLP technologies for LRLs and language varieties.

2.2 Dataset

In this subsection, we introduce two of the largest, most popular NLP datasets for textual inference, namely, the Stanford Natural Language Inference (SNLI) and Multi-Genre Natural Language Inference (MNLI) corpora.

Dataset	Premise	Hypothesis	Label
SNLI	A land rover is being driven across a river.	A vehicle is crossing a river.	entailment
	Children smiling and waving at camera	They are smiling at their parents	neutral
	An older man is drinking orange juice at a restaurant.	Two women are at a restaurant drinking wine.	contradiction
MNLI	So i have to find a way to supplement that	I need a way to add something extra.	entailment
	The new rights are nice enough	Everyone really likes the newest benefits	neutral
	I don't know um do you do a lot of camping	I know exactly.	contradiction

Table 1: Randomly chosen original SNLI and MNLI examples and their inferential relationships.

2.2.1 SNLI corpus

The SNLI (Bowman et al., 2015) corpus is constructed from the Flickr30k corpus (Young et al., 2014). The original image caption is classified as the *premise*, whereas, the *hypothesis* is a human-written *premise*-related sentence that must satisfy one of one of three relational conditions: (1) *Entailment* – true image description, (2) *Neutral* – neutral image description, and (3) *Contradiction* – false or random image description. The SNLI corpus is a collection of 570K *premise-hypothesis* sentence pairs, where each pair is aligned with one of these three relational labels.

2.2.2 MNLI corpus

Similarly to SNLI, the MNLI corpus (Williams et al., 2018) is a closely related crowd-sourced collection of 433k sentence pairs and their relational labels. However, MNLI contains 10 distinct genre categories (i.e., *Letters, Verbatim, Fiction, Face-to-face, Travel, Telephone, Travel, Oxford University Press, Slate, 9/11,* and *Government*) written and spoken data instead of image caption data.

3 CODESWITCH Creation

In this section, we first describe the process of the creation of CODESWITCH, carried out in three steps: 1) data collection of morphological syntactic features and dialect-specific vocabulary, 2) candidate retrieval of simple, deterministic morphosyntactic substitutions for unidirectional translations, and 3) human evaluation to test contextual accuracy of perturbations generated by CODESWITCH.

3.1 Data Collection

First, to gain an better understanding of AAE language, we engage with literature, sample text examples and mass collect morpho-syntax rules (which we adapt from the literature) (see Appendix B) (Bailey et al., 1998; Green, 2002; Bland-Stewart, 2005; Dacon, 2022; Blodgett et al., 2020; Stewart, 2014; Blodgett et al., 2016; Elazar and Goldberg, 2018). Therefore, we attempt a proactive approach in data-collection of grammatical, structural and syntactic rules of word case usage of AAE language features to understand the application of AAE in NLP downstream tasks. Next, we employ and assist 6 trained sociolinguist Amazon Mechanical Turk (AMT) workers[2] with our collected set rules and text examples.

Pairwise Sample Collection We first randomly sample n = 5000 SAE premise-hypothesis sentence pairs that contain at least 8 words from both SNLI and MNLI corpora for a total of 10,000 sentence pairs. For contextual accuracy, we task the first 3 workers to obtain the AAE equivalents of our SAE samples (see Table 1), where each annotator is tasked to translate each SAE sentence pair into AAE. The full annotation guidelines can be seen in Appendix C.

3.2 Candidate Retrieval

Starting from data collection, we next retrieve candidate phrases and words use cases for data augmentation from our obtained AAE equivalent sentence pairs. As Liu et al. (2021) uses a deep text classification model to illustrate that demographic language features do in fact influence models to produce false predictions on semantically equivalent SAE and AAE texts, our protocol follows simple, deterministic substitutions of English texts by dialect-specific vocabulary. To do so, we make use of both SAE and AAE sentence pairs in a pairwise fashion and construct a unidirectional informed-based translative morpho-syntax protocol (TMsP) that enables CODESWITCH to convert any given SAE text to a text possessing adequate language features to be considered as AAE from a dominant AAE speaker. More details on TMsP can be found in Appendix B).

[2]Each AMT worker is independent and a trained sociolinguist filtered by HIT approval rate $\geq 96\%$, completed $>$ 10,000 HITs and location (within the United States)

Dataset	Premise	Hypothesis	Label
SNLI AAE	A land rover ***bein*** driven across a river.	A vehicle ***crossin*** a river.	entailment
	Children ***smilin*** ***n*** ***wavin*** at camera	***Dey*** ***smilin*** at ***they*** parents	neutral
	A older man ***drinkin*** orange juice at a restaurant.	Two women at a restaurant ***drinkin*** wine.	contradiction
MNLI AAE	So i ***gotta*** find a way ***ta*** supplement ***dat***	I need a way ***ta*** add ***sumn*** extra.	entailment
	Da new rights nice enough	***Everybody*** really likes ***da*** newest benefits	neutral
	Ion ***kno*** um do ***u*** do a lot of ***campin***	I ***kno*** exactly.	contradiction

Table 2: Augmented SNLI and MNLI examples (from Table 1) following the application of CODESWITCH. Each blue highlight corresponds to the AAE equivalent from their respective SAE counterpart.

Algorithm 1: The translative syntactic morphological method for CODESWITCH.

Input: Original SAE sequence x
Output: Translated AAE sequence x'
begin function
Load SAE input sequence $\rightarrow x$
$x \leftarrow$ LOWER(x)
$T \leftarrow$ TOKENIZE(x)
for all $i = 1, 2, ..., |T|$ **do**
  if $i \in$ {TMsP} **then**
    $T_i \leftarrow$ CODESWITCH(i)
  end if
end for
$x' \leftarrow$ DETOKENIZE(T)
return x'
end function

Obtaining new texts for downstream tasks from authors of certain demographic groups is time-consuming and requires heavy human labor (Liu et al., 2021; Dacon, 2022). Therefore, we create CODESWITCH (see Algorithm 1), a greedy unidirectional morphosyntactically-informed rule-based translation method which is not only fast, but also functions as a human-in-the-loop paradigm; therefore, drastically reduces heavy human labor. Our approach for intent-and-semantically equivalent AAE data augmentation is intuitively simple and effective. Consequently, we can now explore code-switching in several NLP tasks to determine if LLMs such as MLMs learn to make predictions based on demographic/ dialectal language features.

We represent each original NLI corpus as $D < P, H, L >$ with $p \in P$ as the premise, $h \in H$ as the hypothesis and, lastly, $l \in L$ as the label, and create two augmented datasets i.e., SNLI AAE and MNLI AAE, where we represent each augmented NLI dataset as $D' < P', H', L >$. Specifically, translate each premise-hypothesis pair to AAE and keep the original label unchanged to form a new instance. It is important to note that the task of CODESWITCH is to ensure both sets of datasets i.e., D and D' maintain their contextual accuracy, although they consist of two different language styles (see Table 2).

3.3 Human Evaluation

After an initial training of the AMT annotators with our annotation guidelines, we implement a minor calibration study by tasking the remaining 3 independent workers to test our AAE data augmentation method. We randomly sample 200 SAE/AAE sentence pair examples from each of the 4 datasets, for a total of 800 sentence pairs (or 1600 SAE/AAE sentences). The workers were asked to indicate (1) whether the AAE sentences are written by an L1 (or dominant) AAE speaker, or most likely to be machine generated (MG); and (2) whether or not their contextual accuracy is maintained. For content analysis to ensure the quality of our AAE samples and to quantify the extent of agreement between raters, we first let 3 annotators independently rate each AAE-generated sentence pair as "Native" or "MG", then we measure the inter-annotator agreement (IAA) using Krippendorff's α.

We calculate an inter-rater reliability of 0.82, and did not observe significant differences in agreement across the individual sentences. Qualitative analysis revealed that generated samples resembled sequences written by L1 AAE speakers, whereas few samples were classified as most likely MG. Annotators informed us of particular morpho-syntax cases, for example, constant copula deletion of the verb "**be**" and its variants, namely "**is**" and "**are**" is irregular and often inserted last in word order. This indicates that CODESWITCH does not account for contextual instances when generating AAE samples, hence being classified as most likely MG.

4 Empirical Study and Analysis

In this section, we conduct a preliminary study to substantiate the existence of inherent linguistic bias in NLI models. We introduce the base NLI models

and training details, and then we demonstrate our empirical results.

To illustrate inherent linguistic bias of two distinct linguistic systems, we introduce a representative MLM, namely, BERT (Devlin et al., 2018) (see Appendix A for more details).

Models	Model Performance (%)					
	SNLI			MNLI		
	SAE	AAE	Diff.	SAE	AAE	Diff.
$\text{BERT}_{\text{BASE}}$	90.12	86	4.12	84.77	79.79	4.68
$\text{BERT}_{\text{LARGE}}$	90.46	74.55	15.91	84.47	67.35	17.12

Table 3: Model performance when tested on AAE data. The intensity of each red highlight directly corresponds to the absolute difference in accuracy disparities.

We use each original dataset i.e., SNLI and MNLI to fine-tune both BERT models on a batch size of 32 using an AdamW optimizer with a learning rate of 2e-5 and default betas (β_1 = 0.9, β_2 = 0.999) for 3 epochs. Our experiments display that pretrained MLMs "*are only as good as the data they are trained on*" and are unable to make fair predictions (Tan et al., 2020a). In Table 3, we see that the lack of diverse training data results in disparities in model performance in MLMs, which may be significantly be intensified as models become more complex. In Table 4, we illustrate several examples on the inherent linguistic bias on account of demographic language features, and can conclude that demographic/ dialectal language features do in fact influence models to produce false predictions.

5 Debiasing Methods

In Section 4, we empirically demonstrate that popular NLI models show significant bias towards AAE by underperforming on them than SAE. A natural question arises: *how can we remove the biases in NLI models towards different language styles?* To solve this problem, we introduce two simple but effective debiasing strategies: (1) counterpart data augmentation (CDA); and (2) language Style disentanglement (LSD).

5.1 Counterpart Data Augmentation

The bias of NLI models originates from the training data. Since the training data contains only SAE, the NLI models trained on such data does not understand the unique vocabulary and grammar of AAE, which leads to poor performance. Thus, we propose to implement CODESWITCH to augment the original SAE training data by translating them to their AAE counterparts and in turn implement CDA strategy similar to (Zhao et al., 2018; Zmigrod et al., 2019). Then, we will get a large augmented training dataset, D^+, which is twice the size of the original datasets (*i.e.,* SNLI) as it contains both D and D'.

5.2 Language Style Disentanglement

For two texts with the similar intent and semantic content of different language styles (e.g. SAE v.s. AAE), an NLI model may tend to make biased predictions towards one style. The immediate reason is that the NLI prediction are based on the language style features, instead of relying solely on the semantic features of the texts. Based on this consideration, we propose LSD, an in-processing debiasing method, which tries to disentangle the language style features from the semantic features in text representations and forces the NLI model to make inference on the pure semantic representations.

5.2.1 The LSD Framework

To achieve disentanglement, we adopt the idea of adversarial learning. Figure 1 illustrates the overall framework of LSD. We view the framework as three parts: (1) the BERT model that encodes a premise-hypothesis pair as a fixed-dimensional representation $\mathbf{E}_{[CLS]}$; (2) a feed-forward neural (FFN) classifier $\mathcal{C}$ that takes $\mathbf{E}_{[CLS]}$ as input to predict the inferential relationship between the premise and the hypothesis; and (3) a FFN discriminator $\mathcal{D}$ that predicts whether the sentence pair is SAE or AAE based on $\mathbf{E}_{[CLS]}$. Via adversarial learning, our goal is to build a BERT model that can produce an accurate semantic representation of the text pair so that the classifier $\mathcal{C}$ can make correct predictions based on it, while the representation is free from the language style features of the texts, so that the discriminator $\mathcal{D}$ cannot distinguish whether the texts are from D or D'.

5.2.2 An Optimization Method

We present our optimization algorithm for the LSD framework in Algorithm 2. We train the framework on the augmented training dataset obtained via our CODESWITCH method as we do in CDA. In the training data $\mathcal{T} = \{< P_i, H_i, L_i, S_i >\}_{i=1}^{|\mathcal{T}|}$, each instance consists of a premise p, a hypothesis h, a label l, and a binary language style label $S \in \{\text{SAE}, \text{AAE}\}$. At the beginning, we first load pretrained BERT parameters, and initialize the pa-

Premise	Hypothesis	Label	Prediction
Dis church choir sings ***ta*** ***da*** masses as ***dey*** sing joyous songs from ***da*** book at a church.	***Da*** church filled ***wit*** song.	Entailment	Neutral
Dis church choir sings ***ta*** ***da*** masses as ***dey*** sing joyous songs from ***da*** book at a church.	***Da*** church has cracks in ***da*** ceiling.	Neutral	Contradiction
Dis church choir sings ***ta*** ***da*** masses as ***dey*** sing joyous songs from ***da*** book at a church.	A choir ***singin*** at a baseball game.	Contradiction	Entailment
A woman ***wit*** a green headscarf, blue shirt ***n*** a very big grin.	***Da*** woman young.	Neutral	Contradiction
A woman ***wit*** a green headscarf, blue shirt ***n*** a very big grin.	***Da*** woman very happy.	Entailment	Neutral

Table 4: An illustrative example on the inherent linguistic bias of a NLI models. Each blue highlight corresponds to the AAE equivalent from their respective SAE counterpart (see Appendix B)

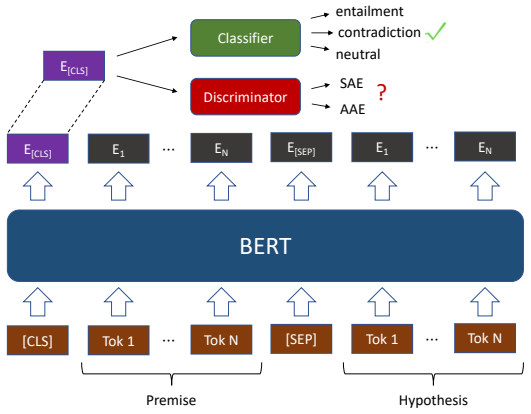

Figure 1: An illustration of the language-style disentanglement model.

rameters of the classifier $\mathcal{C}$ and the discriminator $\mathcal{D}$ (line 3-4). In each iteration, we first obtain a mini-batch of training data $\mathcal{B} = \{< P_i, H_i, L_i, S_i >\}_{i=1}^{|\mathcal{B}|}$ (line 3). Then, we update the discriminator $\mathcal{D}$ by minimizing the following cross-entropy loss (line 4):

$$L_{\mathcal{D}} = -(\mathbb{I}\{S = 0\} \log p_0^{\mathcal{D}} + \mathbb{I}\{S = 1\} \log p_1^{\mathcal{D}}) \quad (1)$$

where S is the language style label of the utterance. $S = 0$ represents for SAE and $S = 1$ represents for AAE. $p_0^{\mathcal{D}}$ and $p_1^{\mathcal{D}}$ are the two elements in the predicted probability $\mathbf{p}^{\mathcal{D}}$ from the discriminator $\mathcal{D}$. Minimizing $L_{\mathcal{D}}$ will force $\mathcal{D}$ to make correct predictions.

Next, we calculate the cross-entropy loss on the main prediction task:

$$L_{\mathcal{C}} = -(\mathbb{I}\{L = 0\} \log p_0^{\mathcal{C}} + \mathbb{I}\{L = 1\} \log p_1^{\mathcal{C}} + \mathbb{I}\{L = 2\} \log p_2^{\mathcal{C}})$$

Algorithm 2: The optimization method for the LSD framework.

1 **Input**: Training data $\mathcal{T} = \{< P_i, H_i, L_i, S_i >\}_{i=1}^{|\mathcal{T}|}$ and Validation data $\mathcal{V} = \{< P_i, H_i, L_i, S_i >\}_{i=1}^{|\mathcal{V}|}$
2 **Output**: BERT parameters $\mathbf{W}^{\text{BERT}}$, classifier parameters $\mathbf{W}^{\mathcal{C}}$
3 Load pre-trained parameters $\mathbf{W}^{\text{BERT}}$
4 Initialize $\mathbf{W}^{\mathcal{C}}$ and $\mathbf{W}^{\mathcal{D}}$
1: **for** N epochs **do**
2: **for** M batches **do**
3: Obtain a mini-batch of training data $\mathcal{B}$ from $\mathcal{T}$
4: Update $\mathbf{W}^{\mathcal{D}}$ by optimizing $L_{\mathcal{D}}$ in Equation 1
5: Update $\mathbf{W}^{\text{BERT}}$ and $\mathbf{W}^{\mathcal{C}}$ by optimizing L in Equation 2
6: **end for**
7: Run the BERT model and the classifier $\mathcal{C}$ on validation data $\mathcal{V}$
8: Save parameters $\mathbf{W}^{\text{BERT}}$ and $\mathbf{W}^{\mathcal{C}}$ if achieving the best validation performance so far.
9: **end for**

where L is the set of labels of the NLI task. $S = 0, 1, 2$ represent for entailment, contradiction, and neutral, respectively. $p_j^{\mathcal{C}}$ indicates the predicted probability for the j-th label from the classifier $\mathcal{C}$. Minimizing $L_{\mathcal{C}}$ will force $\mathcal{C}$ to make correct predictions. To ensure that the BERT model produces a text representation that can fool the discriminator, when training, we consider another entropy loss:

$$L_{\mathcal{D}'} = -(p_0^{\mathcal{D}} \log p_0^{\mathcal{D}} + p_1^{\mathcal{D}} \log p_1^{\mathcal{D}})$$

$L_{\mathcal{D}'}$ is the entropy of the predicted distribution $\mathbf{p}^{\mathcal{D}}$ from the discriminator. Minimizing it makes $\mathbf{p}^{\mathcal{D}}$ close to an even distribution, preventing $\mathcal{D}$ from making correct predictions. We update the BERT model and the classifier by minimizing the following combined loss (line 5):

$$L = L_{\mathcal{C}} + L_{\mathcal{D}'} \quad (2)$$

At the end of each epoch, we run the BERT model and the classifier on the validation data, and save their parameters if they achieve the best validation performance.

5.3 Experimental results

In Table 5, we show the performances of the two debiasing methods on two datasets in terms of two BERT models. In Table 3, the results of the debiased models CDA, LSD and that of the original models were compared. Note that our two debiasing methods reduce the gap between the performances on SAE and AAE significantly. The original BERT models perform well on SAE test data but exhibit a decrease in performance when they are tested on AAE data. However, the BERT models trained under CDA or LSD debiasing strategies achieve similar model performance on SAE and AAE, which demonstrates the effectiveness of the two debiasing methods to mitigate bias in NLI models.

	Model Performance (%)					
	SNLI			MultiNLI		
Models	SAE	AAE	Diff.	SAE	AAE	Diff.
CDA_{BASE}	89.77	89.76	0.01	84.29	83.98	0.31
LSD_{BASE}	90.35	90.49	0.14	84.50	83.81	0.69
$\text{CDA}_{\text{LARGE}}$	90.48	90.36	0.12	84.66	84.20	0.46
$\text{LSD}_{\text{LARGE}}$	90.60	90.53	0.07	84.72	84.30	0.42

Table 5: Model performances of two debiased NLI models. The intensity of each green highlight directly corresponds to the absolute difference in accuracy.

Furthermore, our debiased models not only improve the performance on AAE data, but also maintain similar performance on SAE data as the original model. This is due to either the introduction of additional AAE training data which is not always available, and the disentanglement between the semantic and language style features of texts enhancing the model's capability of understanding natural language. Lastly, we find that LSD generally outperforms CDA on both SAE and AAE data. In addition, LSD is an adversarial learning debaising method that filters out irrelevant language style information towards the NLI task. In fact, LSD is also generalizable for more effective and architecturally similar models such as DeBERTa (He et al., 2020), XLNet (Yang et al., 2019), and T5 (Raffel et al., 2019) to ensure fairness as well as robustifying larger language models.

6 Related Work

Previous works focus on AAE in the context of *racial bias* as a result of systemic biases in model performance. For example, Blodgett et al. (2018) focus on dependency parsing social media AAE to analyze the impacts of performance disparities between AAE and SAE tweets. Other works undertake AAE within the scope of detecting and mitigating the presence of racial bias in areas of offensive and abusive language detection (Liu et al., 2020a; Sap et al., 2019), sentiment analysis (Groenwold et al., 2020) and hate speech detection (Davidson et al., 2019; Sap et al., 2019). However, these influential works do not engage with AAE literature, utilize a human-in-the-loop paradigm nor employ the humans who create such data. Thus, these pivotal works fail to understand AAE's phonological and morphological language features—thereby simply treating AAE as another non-Penn Treebank English variety (Blodgett et al., 2020).

Fairness in NLP. As social and racial disparities have become a compelling issue within the NLP community, focal topics of fairness, accountability, ethics, sustainable development, etc., have gained momentous attention in recent years (Hovy and Spruit, 2016). Recent work on fairness has primarily been focused on racial and gender biases in distributed word representations (Bolukbasi et al., 2016; Zhao et al., 2018; Zmigrod et al., 2019), coreference resolution (Rudinger et al., 2018), sentence encoders (May et al., 2019), machine translation (Tan et al., 2020b; Prates et al., 2018), and dialogue generation (Liu et al., 2020a,b).

Adversarial learning in NLP. Adversarial examples were initially explored in computer vision by Szegedy et al., where these examples were intended to influence models to produce false predictions. However, in NLP, adversarial examples can occur at a phonetic, phonological, morphological, syntactic, semantic, or pragmatic level (Tan et al., 2020a; DeBose, 1992; Gardner-Chloros et al., 2009; Young and Barrett, 2018). Liu et al. (2020a) displays that dialogue systems are prone to produce offensive responses when fed AAE language features in comparison to SAE, whereas Liu et al. (2020b) propose a novel adversarial learning framework which directly addresses the issue of gender bias in dialogue models while maintaining their performance. Both Alzantot et al. (2018) and Joshi et al. (2019) exploit the notion of adversariality by utilizing word embeddings to find the k nearest

synonymic examples.

Summary. These influential works demonstrate novel adversarial learning methodologies on a character and/or word-level in order to address bias issues surrounding protected attributes such as race and gender by improving model robustness. Similarly, our work utilizes a human-in-the-loop paradigm by employing humans who create such data, to create a novel morphosyntactic method to perturb language styles on a syntactic-level to highlight the need for dialectal language diversity in training data.

7 Conclusion and Future Works

To address compelling fairness, accountability, transparency, and ethical concerns surrounding the sustainability of language use in NLP applications, we claim that the addition of diverse dialectal language in training data will improve model robustness and generalizability. Our findings show that our proposed debiasing methods not only improves the performance on AAE data but effectively reduces the performance gap between SAE and AAE significantly, while maintaining or even improving the prediction performance on SAE data. Therefore, training under these two debiasing strategies aids in the mitigation of linguistic bias in NLI models.

We conclude that though similar, the two language styles, SAE and AAE are not identical, and thus, should not solely be evaluated against each other, but compared to as a basis of model performance minimize the existence of inherent linguistic bias in language models. In the future, we intend to release CODESWITCH a morphosyntactically-informed rule-based translation method for unidirectional data augmentation for generating intent-and-semantically-equivalent AAE examples as a public python package, to encourage further computational linguistic research into debiasing various NLP systems. We actively intend on updating CODESWITCH s.t. it can include new or regional-specific *lingo*. In this way, CODESWITCH can constitute potential groundwork on ways that AAE can effectively be integrated in NLP systems to improve future language models during their development and employment.

8 Limitations And Ethical Considerations

All authors must warrant mentioning that the increased performance for underrepresented dialects in NLP systems has the potential to enable automated discrimination based on the use of non-standard dialects. Although, we attempt to highlight the need for dialectal inclusivity for impactful speech and language technologies, we do not intend for increased feelings of marginalization of an already stigmatized community.

We have established our method's effectiveness for data augmentation for generating intent-and-semantically-equivalent AAE examples and believe that CODESWITCH could be further improved by addressing the following limitations:

1. Currently, CODESWITCH is a unidirectional data augmentation method and cannot be used in reverse as a deterministic text normalization/preprocessing system which can convert all text to SAE.

2. CODESWITCH operates on simple, deterministic substitutions for morphosyntactically-informed translations rules found in Appendix B rather than that of real L1 and L2 AAE speakers, which may result in the lack of several formal/informal phrases, expressions, idioms, cultural and regional-specific lingo, and slang-related words (Blodgett et al., 2020). For example, "I ***sholl*** was finna ask who money dat is ", where "*sholl*" refer to the replacement of the word "*sure*".

3. Although CODESWITCH possesses several simple, deterministic morphosyntactically-informed translation rules it does account for contextual instances of accurate copula deletion. This may lead to a discrepancy between actual text written by L1 and/or L2 AAE speakers and our proposed data augmentation method.

In the future, we intend to address these limitations and ethical considerations by partnering with AAE diglossic communities in hopes of robustifying CODESWITCH to be probabilistic rather than deterministic to capture different AAE variants of the same SAE term (for example, the AAE equivalents to "what's" $\rightarrow$ "*waz*"/"*wus*"/"*wats*". In addition, we will investigate inherent linguistic bias in other NLP applications.

References

Moustafa Alzantot, Yash Sharma, Ahmed Elgohary, Bo-Jhang Ho, Mani Srivastava, and Kai-Wei Chang.

2018. Generating natural language adversarial examples. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 2890–2896, Brussels, Belgium. Association for Computational Linguistics.

Guy Bailey, John Baughan, Salikoko S. Mufwene, and John R. Rickford. 1998. *African-American English: Structure, History and Use (1st ed.)*. Routledge.

Linda M. Bland-Stewart. 2005. Difference or deficit in speakers of african american english? https://leader.pubs.asha.org/doi/10.1044/leader.FTR1.10062005.6.

Su Lin Blodgett, Solon Barocas, Hal Daumé III, and Hanna M. Wallach. 2020. Language (technology) is power: A critical survey of "bias" in NLP. *CoRR*, abs/2005.14050.

Su Lin Blodgett, Lisa Green, and Brendan O'Connor. 2016. Demographic dialectal variation in social media: A case study of african-american english. *CoRR*, abs/1608.08868.

Su Lin Blodgett, Johnny Wei, and Brendan O'Connor. 2018. Twitter Universal Dependency parsing for African-American and mainstream American English. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1415–1425, Melbourne, Australia. Association for Computational Linguistics.

Tolga Bolukbasi, Kai-Wei Chang, James Zou, Venkatesh Saligrama, and Adam Kalai. 2016. Man is to computer programmer as woman is to homemaker? debiasing word embeddings. In *Proceedings of the 30th International Conference on Neural Information Processing Systems*, NIPS'16, page 4356–4364.

Samuel R. Bowman, Gabor Angeli, Christopher Potts, and Christopher D. Manning. 2015. A large annotated corpus for learning natural language inference. *CoRR*, abs/1508.05326.

Jamell Dacon. 2022. Towards a deep multi-layered dialectal language analysis: A case study of African-American English. In *Proceedings of the Second Workshop on Bridging Human–Computer Interaction and Natural Language Processing*, pages 55–63, Seattle, Washington. Association for Computational Linguistics.

Jamell Dacon and Jiliang Tang. 2021. What truly matters? using linguistic cues for analyzing the #blacklivesmatter movement and its counter protests: 2013 to 2020.

Thomas Davidson, Debasmita Bhattacharya, and Ingmar Weber. 2019. Racial bias in hate speech and abusive language detection datasets. In *Proceedings of the Third Workshop on Abusive Language Online*, pages 25–35, Florence, Italy. Association for Computational Linguistics.

Charles E. DeBose. 1992. Codeswitching: Black english and standard english in the african-american linguistic repertoire. *Journal of Multilingual and Multicultural Development*, 13(1-2):157–167.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. BERT: pre-training of deep bidirectional transformers for language understanding. *CoRR*, abs/1810.04805.

Yanai Elazar and Yoav Goldberg. 2018. Adversarial removal of demographic attributes from text data. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 11–21, Brussels, Belgium. Association for Computational Linguistics.

Anjalie Field, Su Lin Blodgett, Zeerak Waseem, and Yulia Tsvetkov. 2021. A survey of race, racism, and anti-racism in NLP. *CoRR*, abs/2106.11410.

Penelope Gardner-Chloros et al. 2009. *Code-switching*. Cambridge university press.

Lisa J. Green. 2002. *African American English: A Linguistic Introduction*. Cambridge University Press.

Sophie Groenwold, Lily Ou, Aesha Parekh, Samhita Honnavalli, Sharon Levy, Diba Mirza, and William Yang Wang. 2020. Investigating African-American Vernacular English in transformer-based text generation. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 5877–5883, Online. Association for Computational Linguistics.

Pengcheng He, Xiaodong Liu, Jianfeng Gao, and Weizhu Chen. 2020. Deberta: Decoding-enhanced BERT with disentangled attention. *CoRR*, abs/2006.03654.

Dirk Hovy and Shannon L. Spruit. 2016. The social impact of natural language processing. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 591–598, Berlin, Germany. Association for Computational Linguistics.

Mandar Joshi, Danqi Chen, Yinhan Liu, Daniel S. Weld, Luke Zettlemoyer, and Omer Levy. 2019. Spanbert: Improving pre-training by representing and predicting spans. *CoRR*, abs/1907.10529.

Sharese King. 2020. From african american vernacular english to african american language: Rethinking the study of race and language in african americans' speech. *Annual Review of Linguistics*, 6(1):285–300.

Haochen Liu, Jamell Dacon, Wenqi Fan, Hui Liu, Zitao Liu, and Jiliang Tang. 2020a. Does gender matter? towards fairness in dialogue systems. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 4403–4416, Barcelona, Spain (Online). International Committee on Computational Linguistics.

Haochen Liu, Wei Jin, Hamid Karimi, Zitao Liu, and Jiliang Tang. 2021. The authors matter: Understanding and mitigating implicit bias in deep text classification. In *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*, pages 74–85, Online. Association for Computational Linguistics.

Haochen Liu, Wentao Wang, Yiqi Wang, Hui Liu, Zitao Liu, and Jiliang Tang. 2020b. Mitigating gender bias for neural dialogue generation with adversarial learning. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 893–903, Online. Association for Computational Linguistics.

Chandler May, Alex Wang, Shikha Bordia, Samuel R. Bowman, and Rachel Rudinger. 2019. On measuring social biases in sentence encoders. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 622–628, Minneapolis, Minnesota. Association for Computational Linguistics.

Marcelo O. R. Prates, Pedro H. C. Avelar, and Luís C. Lamb. 2018. Assessing gender bias in machine translation - A case study with google translate. *CoRR*, abs/1809.02208.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J. Liu. 2019. Exploring the limits of transfer learning with a unified text-to-text transformer. *CoRR*, abs/1910.10683.

Marco Tulio Ribeiro, Sameer Singh, and Carlos Guestrin. 2018. Semantically equivalent adversarial rules for debugging NLP models. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 856–865, Melbourne, Australia. Association for Computational Linguistics.

Rachel Rudinger, Jason Naradowsky, Brian Leonard, and Benjamin Van Durme. 2018. Gender bias in coreference resolution. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 2 (Short Papers)*, pages 8–14, New Orleans, Louisiana. Association for Computational Linguistics.

Maarten Sap, Dallas Card, Saadia Gabriel, Yejin Choi, and Noah A. Smith. 2019. The risk of racial bias in hate speech detection. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 1668–1678, Florence, Italy. Association for Computational Linguistics.

Roger C Schank. 1972. Conceptual dependency: A theory of natural language understanding. *Cognitive psychology*, 3(4):552–631.

Shanya Sharma, Manan Dey, and Koustuv Sinha. 2021. Evaluating gender bias in natural language inference. *CoRR*, abs/2105.05541.

Ian Stewart. 2014. Now we stronger than ever: African-American English syntax in Twitter. In *Proceedings of the Student Research Workshop at the 14th Conference of the European Chapter of the Association for Computational Linguistics*, pages 31–37, Gothenburg, Sweden. Association for Computational Linguistics.

Christian Szegedy, Wojciech Zaremba, Ilya Sutskever, Joan Bruna, Dumitru Erhan, Ian Goodfellow, and Rob Fergus. Intriguing properties of neural networks.

Samson Tan, Shafiq Joty, Min-Yen Kan, and Richard Socher. 2020a. It's morphin' time! Combating linguistic discrimination with inflectional perturbations. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 2920–2935, Online. Association for Computational Linguistics.

Samson Tan, Shafiq Joty, Lav Varshney, and Min-Yen Kan. 2020b. Mind your inflections! Improving NLP for non-standard Englishes with Base-Inflection Encoding. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 5647–5663, Online. Association for Computational Linguistics.

Adina Williams, Nikita Nangia, and Samuel Bowman. 2018. A broad-coverage challenge corpus for sentence understanding through inference. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 1112–1122. Association for Computational Linguistics.

Zhilin Yang, Zihang Dai, Yiming Yang, Jaime G. Carbonell, Ruslan Salakhutdinov, and Quoc V. Le. 2019. Xlnet: Generalized autoregressive pretraining for language understanding. *CoRR*, abs/1906.08237.

Peter Young, Alice Lai, Micah Hodosh, and Julia Hockenmaier. 2014. From image descriptions to visual denotations: New similarity metrics for semantic inference over event descriptions. *Transactions of the Association for Computational Linguistics*, 2:67–78.

Vershawn Ashanti Young. 2009. "nah, we straight": An argument against code switching. *JAC*, 29(1/2):49–76.

Vershawn Ashanti Young and Rusty Barrett. 2018. *Other people's English: Code-meshing, code-switching, and African American literacy*. Parlor Press LLC.

Jieyu Zhao, Yichao Zhou, Zeyu Li, Wei Wang, and Kai-Wei Chang. 2018. Learning gender-neutral word embeddings. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing, Brussels, Belgium, October 31 - November 4, 2018*, pages 4847–4853.

Xiang Zhou and Mohit Bansal. 2020. Towards robustifying NLI models against lexical dataset biases. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 8759–8771, Online. Association for Computational Linguistics.

Ran Zmigrod, Sabrina J. Mielke, Hanna Wallach, and Ryan Cotterell. 2019. Counterfactual data augmentation for mitigating gender stereotypes in languages with rich morphology. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 1651–1661, Florence, Italy. Association for Computational Linguistics.

A Implementation Details

A.1 Details of the Base Model

BERT – Bidirectional Encoder Representations from Transformers (BERT) (Devlin et al., 2018) is a Transformer-based ML technique for NLP that achieves state-of-the-art results in a wide variety of NLP tasks. BERT is trained on a huge Books Corpus + Wikipedia dataset i.e., raw unlabeled English text consisting of 3.3 billion words. This model exploits an attention mechanism to learn contextual relationships between words and optimizes two objectives: (1) Masked Language Modeling (MLM) and (2) Next Sentence Prediction (NSP), and has a vocabulary size of 30,522.

A.2 Details of Experimental Settings

In summary, BERT optimizes its two objectives uniformly, and thus, it serves as a appropriate model for our task of understanding the inferential relationships between sentence pairs by examining the differences in language styles from different demographic groups e.g. African Americans. Now, we will now give details of each pretrained BERT model below:

1. BERT-base-uncased - Trained on raw English text, and consists of 12-layers, 768-hidden, 12-heads, 110M parameters.

2. BERT-large-cased - Trained on raw lowercased English text, and consists of 24-layer, 1024-hidden, 16-heads, 335M parameters. Trained on cased English text.

B Translative Morpho-syntax Protocol

Here we present a set of 20 linguistic phonetic and morphological text rules that are used to *code-switch* from SAE to AAE while maintaining contextual accuracy i.e., original structure, intent, semantic equivalence, and quality of a text. Please note that these are only a few examples of the most commonly used morphological linguistic AAE features (which we adapt from AAE literature). Our deterministic translative morpho-syntax protocol (TMsP) and its cases are as follows:

1. Consonant (‘**t**’) deletion (Special case) : e.g. “*just*” → “*jus*”; “*must*” → “*mus*”

2. Contractive (**’all**) gain: “*You all*” → “*Y***’all**”

3. Contractive negative auxiliary verbs replacement: “*doesn’t*” → “*don’t*”

4. Contractive (**’re**) loss: e.g. “*you’re*” → “*you*”; “*we’re*” → “*we*”; “*they’re*” → “*they*”

5. Contractive word replacement: e.g. “*isn’t*” → “*ain’t*”; “*wasn’t*” → “*ain’t*”

6. Copula deletion: Deletion of the verb “**be**” and its variants, namely “**is**” and “**are**” e.g. “*He* **is** *on his way*” → “*He on his way*”; “*You* **are** *right*” → “*You right*”

7. Gerund consonant (‘**g**’) deletion and retainment:

   - Consonant (‘**g**’) deletion: e.g. “*coming*” → “*comin*”; “*going*” → “*goin*”
   - Consonant (‘**g**’) retainment (Exception case): e.g. “*–ingin**g***”

8. Homophonic word replacement: e.g. “*whine*” → “*wine*”; “*you’re*” → “*your*”

9. Indefinite article replacement: e.g. “*an*” → “*a*”

10. Indefinite pronoun replacement: e.g. “*anyone*” → “*anybody*”; “*everyone*” → “*everybody*”

11. Interdental fricative loss: e.g. “*this*” → “*dis*”; ‘*that*’ → ‘*dat*”; “*than*” → “*dan*”; “*their*” → “*they* (*dey*)”; “*the*” → “*da*”

12. Negative concord replacement: e.g. “*Don’t say* ***anything***” → “*Don’t say* ***nothing***”

13. Phrase reduction (present/ future tense) ⇒ word e.g. “*going to*” → “*gonna*”; “*want to*” → “*wanna*”; “*trying to*” → “*tryna*”; “*what’s up*” → “*wassup*”; “*fixing to*” → “*finna*”

14. Possessive (**’s**) removal: e.g. “*He***’s** *mad at me*” → “*He mad at me*”

15. Present tense possession replacement: e.g. "*John **has** two apples*" → "*John **got** two apples*"; "*The neighbors **have** a bigger pool*" → "*The neighbors **got** a bigger pool*"

16. Remote past "***been***" + completive ('**done**'): "*I've **already done** that*" → "*I **been done** that*"

17. Remote past "***been***" + completive ('**did**'): "*She **already did** that*" → "*She **been did** that*"

18. Remote past "***been***" + Present tense possession replacement: "*I **already have** food*" → "*I **been had** food*"; "*You **already have** those shoes*" → "*You **been got** those shoes*"

19. Term-fragment deletion: e.g. "*brother*" → "*bro*"; "*sister*" → "*sis*"; "*your*" → "*ur*"; "*suppose*" → "*pose*"; "*more*" → "*mo*"

20. Term-fragment replacement: "*something*" → "*sumn*"; "*through*" → "*thru*"; "*for*" → "*fa*"; "*nothing*" → "*nun*"

C Annotation Guidelines

You will be given a phrase that is written in Standard American English (SAE), your task is to correctly identify if the translative vocabulary rules in Appendix B are accurate in order to translate SAE text to AAE text. Furthermore, while reviewing the rules, be sure to mention that these rules and/or morpho-syntax word cases in the sampled premise-hypothesis sentence pairs maintain their contextual accuracy i.e., original structure, intent, semantic equivalence, and quality.

SAE to AAE Protocol

1. Are you a dominant AAE speaker?

2. If you responded "yes" above, are you bi-dialectal?

3. If you responded "yes" above, are you capable of code-switching by alternating between SAE and AAE frequently on a daily basis in a single conversation or situation?

4. Given TMsP above in Appendix B, are these main grammatical, structural and syntactic rules of word case usage of AAE linguistic features?

5. If you responded "no" above, can clarify which rule is insufficient? In addition, if possible, can you provide a grammatical, structural or syntactic rule that is not detailed in Appendix B?

D Contextual accuracy Protocol

Given a table of SAE-AAE sentence pairs examples, determine whether or not their contextual accuracy is maintained.

SAE	AAE
i will go back to the house	imma go back ta da house
i don't want to go to bed	ion wanna go ta bed
he isn't my friend, but he's a king	he ain't my friend, but he a king
she is being weird to me	she been weird ta me
you all are annoying	yall annoyin
he isn't coming anymore	he ain't comin no mo
a woman is trying to walk	a woman tryna walk
this bag and that shoe are mine	dis bag n dat shoe mine
their kids are laughing	they kids laughin
john and kates have two dogs	john n kates hav two dogs
are you going through something	u goin thru sumn
what are you doing	wat r u doin
what's the temperature	wus da temperature
they have a better car than us	dey hav a betta car dan us
so you're going to the party	so your gonna go ta da party
they are singing but they can't sing	dey singing but dey can't sing
you could of have it all	u coulda hav it all
he would've had it if he was here	he woulda had it if he was here
we should have been first in line	we shoulda been first in line
he should of had the last bite	he shoulda had da last bite

Table 6: SAE examples and their AAE equivalents (after using CODESWITCH).

1. As you responded "yes" a previous question,

   > *... are you capable of code-switching by alternating between SAE and AAE frequently on a daily basis in a single conversation or situation?*

   We will now provide 20 lower-cased test sentences is Table 6.

2. Have you ever seen any of these words in a particular sentence in Table 6, for example, on social media such as Twitter?

3. If you responded "yes" above, For each SAE sentence, does each plausible AAE sentence resemble adequate AAE morphological language features from a dominant AAE speaker after applying CODESWITCH?

4. If you responded "yes" above, do these pairs maintain their contextual accuracy i.e., original structure, intent, semantic equivalence and quality?

5. For dialectal (morphological and phonological) purposes, are these particular words spelt how would you say or use them? For example, texting or posting on social media?

6. If you responded "no" above, can you provide a different spelling along with its SAE equivalent?

Can We Guide a Multi-Hop Reasoning Language Model to Incrementally Learn at each Single-Hop?

Jesús Lovón-Melgarejo ♠, Jose G. Moreno ♠, Romaric Besançon ♢
Olivier Ferret ♢, Lynda Tamine ♠
♠Université Paul Sabatier, IRIT, Toulouse, France
♢Université Paris-Saclay, CEA, List, Palaiseau, France
{jesus.lovon, jose.moreno, tamine}@irit.fr
{romaric.besancon, olivier.ferret}@cea.fr

Abstract

Despite the success of state-of-the-art pre-trained language models (PLMs) on a series of multi-hop reasoning tasks, they still suffer from their limited abilities to transfer learning from simple to complex tasks and vice-versa. We argue that one step forward to overcome this limitation is to better understand the behavioral trend of PLMs at each hop over the inference chain. Our critical underlying idea is to mimic human-style reasoning: we envision the multi-hop reasoning process as a sequence of explicit single-hop reasoning steps. To endow PLMs with incremental reasoning skills, we propose a set of inference strategies on relevant facts and distractors allowing us to build automatically generated training datasets. Using the SHINRA and ConceptNet resources jointly, we empirically show the effectiveness of our proposal on multiple-choice question answering and reading comprehension, with a relative improvement in terms of accuracy of 68.4% and 16.0% w.r.t. classic PLMs, respectively.

1 Introduction

Recent developments have shown that models based on transformers (Vaswani et al., 2017; Liu et al., 2019b) have emerged as effective soft reasoners over language (Talmor et al., 2020a; Kassner et al., 2020). To teach transformers the ability to emulate reasoning, they are trained on knowledge encoded in the form of natural language statements generally built upon explicit rules (Clark et al., 2020) or symbolic facts that refer to triples in knowledge graphs (KG) (Kassner et al., 2020). In addition, the reasoning skills of these models can successfully combine explicit natural language statements with implicit knowledge acquired during pre-training (Talmor et al., 2020b). In particular, many state-of-the-art pre-trained language models (PLMs), such as BERT (Devlin et al., 2019) or RoBERTa (Liu et al., 2019b), have been successfully used in multi-hop reasoning problems including multi-hop question-answering (QA) tasks (Weber et al., 2019; Richardson and Sabharwal, 2020; Saxena et al., 2020; Saha et al., 2021) and multi-hop reading comprehension (RC) (Min et al., 2019; Ding et al., 2019). Training multi-hop reasoning specifically implies a two-step process: 1) distinguish -within a context- the relevant facts from the distractors to be used for reasoning; both relevant facts and distractors are generally expressed as statements in natural language using linguistic patterns (Clark et al., 2020); 2) reasoning over a sequence of relevant facts leading to chains of reasoning (Das et al., 2019). A common approach to teaching PLMs to solve a multi-hop reasoning task is to convert the structural reasoning task into subtasks that model the sequence of reasoning tasks. For instance, Richardson and Sabharwal (2020) and Clark et al. (2020) rely on a multitasking training strategy (Caruana, 1997) that uses training instances mixing different depths of reasoning steps (hops). More precisely, to teach a model solving a k-hop reasoning problem, it is trained to simultaneously solve single i-hop ($1 \leq i \leq k$) inference tasks. In the same line, Min et al. (2019) and Ding et al. (2019) carry out several steps of single-hop reading comprehension to simulate multi-hop reasoning. However, while yielding impressive results, it is still unclear if PLMs endowed with multi-hop reasoning skills really leverage the learned skills at each single-hop depth level along the reasoning chain. More specifically, our work is motivated by observing that PLMs yield unpredictable results while performing multi-hop reasoning. For instance, previous studies show that the performance of PLMs degrades substantially even with a slight increase in the number of hops in the underlying reasoning tasks (Richardson and Sabharwal, 2020). This result indicates that multi-hop models at lower depths struggle to transfer information to deeper-hop models, giving rise to the *compositionality generalization* (Chaabouni et al., 2020) issue from

Proceedings of the 29th International Conference on Computational Linguistics, pages 1455–1466
October 12–17, 2022.

simpler to complex tasks.

In this work, we advocate that a better understanding of the inherent relationships between the different single-hop reasoning models allows the design of more predictable models. We seek to answer three main questions. First, grounded in previous findings in the literature (Richardson and Sabharwal, 2020) showing the compositionality generalization issue, *do single-hop reasoning models incrementally learn?* **(RQ1)**. We construct large probe datasets using SHINRA (Sekine et al., 2018), ConceptNet (Speer et al., 2017), and RuleTakers (Clark et al., 2020) using single-depths of inference to train and probe single-hop models and compare their performance. Overall, our findings confirm the prevalence of the compositionality generalization issue from complex to less complex multi-hop reasoning tasks. Second, inspired by the human reasoning style to solve complex problems based on simpler ones (Anderson, 1980), *can PLMs be guided toward incremental reasoning?* **(RQ2)**. Specifically, we propose a generic and automatic methodology for generating training probe datasets that endow PLMs with reasoning capabilities over a sequence of single-hop steps. We particularly investigate the impact of using distractor generation strategies. Our empirical results show that we can guide PLMs to incrementally reason by leveraging classic approaches with a gain of up to 7.98 accuracy. These training datasets are publicly available[1]. Finally, grounded on previous findings revealing that PLMs trained on one specific reasoning task improve their performance on different and unrelated reasoning tasks (Talmor et al., 2020b), *do QA tasks leverage incrementally trained reasoning models?* **(RQ3)**. For the multi-hop QA task, in particular, the results show that our approach quickly adapts to obtain an accuracy of 54.74 compared to 52.03 from state-of-the-art

2 Methodology

In this section, we first introduce the basic definitions and notations used in our proposal and then present the data probe generation methodology.

2.1 Task Definition

We focus on the multi-hop symbolic reasoning task over explicit knowledge. Following previous work, our setting includes the following:

1) a **knowledge graph** (KG) $G = (\mathcal{E}, \mathcal{R})$ with entities as nodes ($e \in \mathcal{E}$), inference relationships as edges ($r \in \mathcal{R}$), and a set of real relation *facts* f_{ij} as positive triples (e_i, r, e_j) denoted F^+ among all the possible ones in $\mathcal{E} \times \mathcal{R} \times \mathcal{E}$;

2) a **hypothesis** $\mathcal{H}^k_{ij}$ about the relationship $r*$ between two target entities $(e_i, r*, e_j)$ separated by k hops in graph G;

3) a **hypernym inference path** of depth k on G, referred to as $\mathcal{I}^k_{ij}$, allowing to build a new relation fact $f^*_{ij} \notin F^+$ by combining $k+1$ relation facts along the reasoning chain $< (e_i, r_0, e_1)(e_1, r_1, e_2) \ldots (e_k, r_k, e_j) >$, such as $\forall 0 \leq n \leq k$, $(e_n, r_n, e_{n+1}) \in F^+$, $(e_i, r_1, e_1), (e_k, r_k, e_j) \in F^+$;

4) a **context**, composed of *relevant facts* $\mathcal{F}^*_{ij} \subset F^+$, defined by the facts that form the hypernym inference chain $\mathcal{I}^k_{ij}$ and distractors $\mathcal{D}_{ij}$, defined by triplets that do not form the hypernym inference in $\mathcal{I}^k_{ij}$.

Given a hypothesis in context $< \mathcal{H}^k_{ij}, (\mathcal{F}^*_{ij}, \mathcal{D}_{ij}) >$, the task consists in inferring its truth value. A hypothesis $\mathcal{H}^k_{ij}$ is either *true* if it deductively follows a hypernym inference $\mathcal{I}^k_{ij}$ from the context $(\mathcal{F}^*_{ij}, \mathcal{D}_{ij})$, or *false* if it does not (under the close-world assumption).

2.2 Data Probe Generation

Given a knowledge graph G, we propose a generic dataset generation methodology to probe multi-hop reasoning PLMs in a single-hop setup. We define two generation functions to construct the input $< \mathcal{H}^k_{ij}, (\mathcal{F}^*_{ij}, \mathcal{D}_{ij}) >$: i) $\textsc{Hyp}(G, k)$, to generate both the hypothesis $\mathcal{H}^k_{ij}$ and the related inference path $\mathcal{I}^k_{ij}$; and ii) $\textsc{Distr}(G, \mathcal{I}^k_{ij}, \mathcal{H}^k_{ij})$, to generate a set of distractors $\mathcal{D}_{ij}$ with respect to the inference path $\mathcal{I}^k_{ij}$.

Hypothesis Generation $\textsc{Hyp}(G, k)$. First, we apply the Depth First Search (DFS) algorithm to visit all entities of knowledge graph G, generating a set of paths of length $k+1$, excluding the root, used as inference paths. For the true hypothesis, we create $\mathcal{H}^k_{ij}$ with the form (e_i, r_k, e_j) using the first and last facts from $\mathcal{I}^k_{ij}$ (see Figure 1, which illustrates examples of 1-hop and 2-hop hypothesis). Unlikely, for the false hypothesis, we simply generate a hypothesis $\mathcal{H}_{iz}$ replacing the last real fact of the inference path by $(e_k, r_k, e_z) \notin F^+$.

[1] https://github.com/jeslev/incremental_reasoning

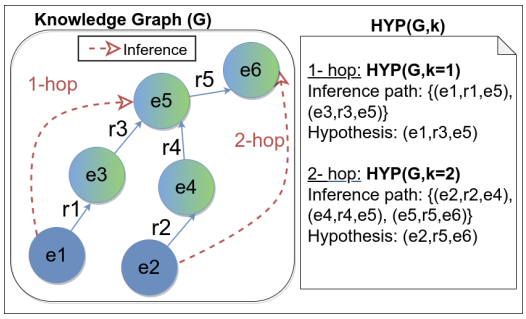

Figure 1: Hypothesis and inference path generation from a knowledge graph for 1-hop and 2-hop reasoning depths.

Distractor Generation $\textsc{Distr}(G, \mathcal{I}^k_{ij}, \mathcal{H}^k_{ij})$. We generate object, relationship, and inference distractors for each hypothesis $\mathcal{H}^k_{ij}$ (as shown in Figure 3b).

An *object distractor* is generated by sampling a fact with the form $(., ., e_j)$, e_j referencing the last entity in the hypothesis. Similarly, a *relationship distractor* is a sampled fact with the form $(., r_k, .)$.

Inference distractors. Finally, we generate *inference distractors* in such a way that they exploit evidence from the structure of the inference path $\mathcal{I}^k_{ij}$ by linking two of its facts with a pivot element. Having in mind the goal of guiding a k-hop model to perform incremental inference over single-hops, we propose distractor strategies that either improve the entity representations or bridge between entities by transferring information along with *intermediate hops* necessary to complete the reasoning. More precisely, based on a recent finding (Kassner and Schütze, 2020) showing that fine-tuned PLMs are good for recognizing false facts, we assume that distractors have a hidden impact on the reasoning task. While most common approaches attempt to improve PLMs entity representations by enriching the context-based relevant facts, we believe distractors can significantly leverage PLMs entity representations and thus the reasoning performance. Thus, we investigate the rationale behind this assumption by designing the following strategies for generating inference distractors: **shared (s-inf)**, which uses the same distractor entity ($x = y$) of the two consecutive facts from the inference path, and the **individual (i-inf)**, that uses different distractor entities ($x \neq y$). Figure 2 shows the implementation P-INF for both inference distractors, with the variable shared= $True$ for (s-inf) and shared= $False$ for (i-inf).

Additionally, we explicitly guide a k-hop model to perform incremental inference using a **guided distractor (g-inf)** that connects the two consecutive facts in the inference path (see Figure 2 algorithm G-INF). The key underlying idea is to drive the PLM to *incrementally reason* over the inference path by transferring information between entities along *intermediate hops* of a multi-hop reasoning path. Figure 3b illustrates the different distractors generated for a specific example.

P-INF ($\mathcal{I}^k_{ij}$**,k, shared**)
$AD = \{\mathcal{E} \times \mathcal{R} \times \mathcal{E}\} \setminus F^+$
$D = \emptyset$
for $i \in 1 \ldots k+1$ **do**
  $(e_x, \_, \_)$= $\mathcal{I}^k_{ij}$[i]
  $(\_, \_, e_{x+2})$= $\mathcal{I}^k_{ij}$[i+1]
  $L1 = \emptyset, L2 = \emptyset$
  for $(e_x, r_a, p) \sim AD(e_x)$ **do**
    $L1 = L1 \cup \{(e_x, r_i, b_j)\}$
  end for
  for $(q, r_b, e_{x+2}) \sim AD(e_{x+2})$ **do**
    $L2 = L2 \cup \{(e_i, r_j, e_{x+2})\}$
  end for
  for $(d_1, d_2) \sim L1 \times L2$ **do**
    $(e_x, r_a, x) = d_1$
    $(y, r_b, e_{x+2}) = d_2$
    if $x = y$ and shared **then**
      $D = D \cup \{d_1, d_2\}$
      break
    else if $x \neq y$ and not shared **then**
      $D = D \cup \{d_1, d_2\}$
      break
    end if
  end for
end for
return D

G-INF ($\mathcal{I}^k_{ij}$**,k)**
D = P-INF($\mathcal{I}^k_{ij}$,k, True)
for $i \in 1 \ldots k+1$ **do**
  $(e_x, \_, \_)$= $\mathcal{I}^k_{ij}$[i]
  $(\_, r, e_{x+2})$= $\mathcal{I}^k_{ij}$[i+1]
  $D = D \cup \{(e_x, r, e_{x+2})\}$
end for
return D

S-INF ($\mathcal{I}^k_{ij}$**,k)**
D = P-INF($\mathcal{I}^k_{ij}$,k, True)
return D

I-INF ($\mathcal{I}^k_{ij}$**,k)**
D = P-INF($\mathcal{I}^k_{ij}$,k, False)
return D

Figure 2: Pseudocode for distractors generation in a k-hop dataset. D, L1, and L2 are lists used to stock the generated distractors. AD stands for "available distractors", considering all the possible triples in the KG not used in the inference. AD(e) represents a filtered AD where e is present.

3 Experimental Setup

3.1 Generated Dataset Probes

We present here the dataset probes, namely Single-RuleTakers (S-RT) and SHINet, we automatically constructed using the previously presented generation functions (see Section 2.2). These datasets are based on three different publicly available resources: the RuleTakers dataset (Clark et al., 2020),

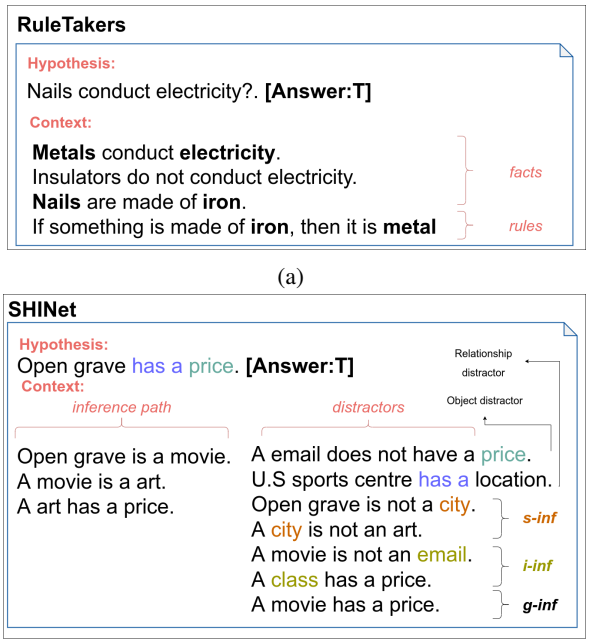

Figure 3: RuleTakers (a) and SHINet 2-hop (b) examples.

SHINRA (Sekine et al., 2018), a knowledge graph manually built upon a structured taxonomy, and ConceptNet (CN) (Speer et al., 2017), another KG widely used in NLP tasks (Talmor et al., 2020b; Ma et al., 2021).

3.1.1 The Single RuleTakers Dataset (S-RT)

In the original RuleTakers dataset, each entry has a small theory representing the context ($\mathcal{F}^*_{ij}$), and a True/False question representing the hypothesis $\mathcal{H}^k_{ij}$, mainly grouped on five variations $k = 0$, and $D \leq k$ with $k = \{1, 2, 3, 5\}$ with questions requiring reasoning up to depths $0, 1, 2, 3, 5$ respectively. An example of a true hypothesis in the RuleTakers dataset is presented in Figure 3(a). We filtered these datasets to construct our probes, single k-hop datasets with $k \in \{0, 1, 2, 3, 5\}$ for train and test splits, called *S-RT* dataset.

3.1.2 SHINet Dataset

The RuleTakers dataset presents the context as a paragraph, with no annotations on the relevant facts or the distractors, making it difficult to measure their impact on the inference process. To overcome this limitation, we created the SHINet dataset built upon the public SHINRA dataset. The SHINRA contains facts with the form $(e_i$, is-a, $e_j)$, limited to the "is-a" relation. Having in mind that the inference task heavily relies on the range of relationships and objects that the model has seen in the training phase (Wang et al., 2021), we created

Dataset	train	dev	test
1-hop *(s-inf)*	35,000	1,200	2,074
2-hop *(s-inf)*	35,000	1,200	5,994
2-hop *(i-inf)*	35,000	1,200	-
2-hop *(g-inf)*	35,000	1,200	-

Table 1: Number of samples for train/dev/test splits for each generated dataset.

SHINet dataset by sampling from SHINRA and ConceptNet based on a manual alignment of the intermediate nodes of SHINRA. We enrich the single "is-a" relationship with the facts from ConceptNet in the form (e_j, r', p_j). The alignment relies on manual verification of finding e_j in both datasets. An example of a true hypothesis in the SHINet dataset with related distractors *s-inf*, *i-inf*, and *g-inf* is presented in Figure 3(b). Table 1 summarizes our generated datasets.

3.2 Model Training

We used PLMs trained on the single-hop training partitions of our generated datasets. It is worth mentioning that this training protocol differs from the protocol used in previous approaches, where mixed datasets $\{0 \leq i\}$-hop datasets are simultaneously used for training multi-hop reasoning models based on a multitasking approach (Richardson and Sabharwal, 2020). More specifically, we exploited the following: 1) we used several pre-trained LMs based on BERT (Devlin et al., 2019), XLNet (Yang et al., 2019), and RoBERTA (Liu et al., 2019b); even using similar architectures, they showed significant differences in performance results, especially on reasoning tasks (Talmor et al., 2020a); 2) as a building block, we used the training protocol proposed in previous work (Talmor et al., 2020b), removing the first fact from $\mathcal{I}^k_{ij}$ in 40% of the samples. Using this training protocol, we can provide insights into both the intrinsic strengths and limits of our proposed PLM since we exploit a recent state-of-the-art PLM that captures rich semantic information.

3.3 Implementation Details

Each fact (e_i, r, e_j) can be expressed as a statement in natural language using linguistic patterns referred to as *fact templates* (e.g., e_i *is a* e_j). We make use of the Hearst Patterns templates (Hearst, 1992; Roller et al., 2018) and the ones proposed in Talmor et al. (2020b) as fact templates. Then, we

train a transformer-based model with a set of input sequences of tokens with the following schema: "[CLS] Context [SEP] Hypothesis [SEP]". Then, we used the output representation of the [CLS] token and projected it into a binary classifier layer to obtain the probabilities that the hypothesis is true or false. For all of the models, we used the transformers' public implementation from HuggingFace (Wolf et al., 2020). Main hyperparameters were set following standard setup or original authors' recommendations. In all the experiments where SHINet is used for training, we set a maximum word length to 256, batch size to 4, number of steps per batch to 729, number of epochs to 4, and Adam as optimizer with learning rate to 1e-5 and weight decay to 0.1. In the case of fine-tuning, each dataset uses its default hyperparameters. However, the parameters remain the same for each dataset regardless of the fine-tuning order. For the QA tasks evaluation, we opted for the same parameters as when fine-tuning the SHINet datasets, except for the loss (categorical cross entropy), and the number of steps per batch is set to 2,163. We trained and evaluated the models with 10 different random seeds and presented mean scores in our comparisons (see Appendix A for computational costs). To provide statistical significance to our results, we applied a test for Almost Stochastic Dominance (Dror et al., 2019) between test score distributions, using $\alpha = 0.05$.

4 Results and Discussion

To address **RQ1** and **RQ2**, we used the k-hop S-RT with $k \in \{0, 1, 2, 3, 5\}$ and the $\{D \leq 2\}$-hop, 1-hop, and 2-hop SHINet dataset. All the SHINet datasets are composed of the object, relationship, and inference distractors. Regarding the inference distractors, we evaluated with the three strategies: *i-inf, s-inf, g-inf*. The default setting uses the *(s-inf)* strategy unless it is explicitly mentioned otherwise. Note that for each SHINet strategy-based dataset, we have train and test partitions. As recommended in the literature (Elazar et al., 2021; Sakaguchi et al., 2020), to avoid biases in these datasets that lead to an overestimation of the reasoning capabilities of PLMs, we applied on the test partitions the AFLITE algorithm (Sakaguchi et al., 2020) that finds machine-detectable embeddings associations to reduce biases. We used optimal parameters after grid-search: n (classifiers) = 64, m(samples) = 1000, top-k = 200, threshold = 0.75. Comparing the original and filtered datasets, we approximately filtered 45% of the total samples. To address **RQ3**, we used the MCQA (Richardson and Sabharwal, 2020), and RACE (Lai et al., 2017). MCQA is composed of 193,000 entries. Each entry is composed of a question and five possible answers, including reasoning tasks such as hypernymy, hyponymy, synonymy detection, and word sense disambiguation. RACE (Lai et al., 2017) consists of nearly 28,000 passages and 100,000 questions divided into Middle and High School sets and up to four possible answers.

Test (k-hop)	**k=0**	**k=1**	**k=2**	**k=3**	**k=5**
Train ↓ Models →	*RoBERTa*				
0-**hop**	**99.99**	*43.51*	*26.52*	*22.94*	*12.78*
1-**hop**	**90.11**	**98.16**	50.64	*37.30*	*23.07*
2-**hop**	66.92	64.62	**88.54**	**91.65**	**96.16**
3-**hop**	68.36	64.44	**88.47**	**91.35**	**96.11**
5-**hop**	63.32	63.11	**87.09**	**89.92**	**95.06**

Table 2: Accuracy performance (in %) for a RoBERTa model trained on k-hop S-RT training set (rows) and tested on k-hop S-RT test set (columns). For a better reading, scores worse than random ($< 50\%$) are in *italic* and good results ($> 80\%$) are in **bold**.

Test (k-hop)	**k=1**	**k=2**	**k=1**	**k=2**	**k=1**	**k=2**
Train ↓ Models →	*XLNet*		*BERT*		*RoBERTa*	
Mixed	98.89	89.06	**99.23**	**94.63**	99.71	93.44
1-**hop**	**99.71**	86.00	98.96	89.75	**99.80**	96.23
2-**hop**	96.31	**99.82**	77.75	87.25	98.77	**99.99**

Table 3: Accuracy performances (in %) for mixed, 1-hop and 2-hop models using the SHINet dataset. The highest values are in **bold**.

4.1 Do Single-Hop Reasoning Models Incrementally Learn? (RQ1)

To answer **RQ1**, we train separately single i-hop reasoning models using the S-RT dataset ($i \in \{0, 1, 2, 3, 5\}$)[2], and we train PLMS using the SHINet dataset on single 1-hop and 2-hop, and the *Mixed* models trained on the $\{k \leq 2\}$-hop SHINet dataset.

Table 2 and Table 3 report, respectively, the accuracy scores for the different single i-hop models using the S-RT dataset, and the accuracy scores of PLMs trained on 1-hop and 2-hop SHINet dataset, as well as the *Mixed* model. We take an empirical approach by assuming that incremental learning is observed when the models generalize from complex to less complex tasks. Overall, we can observe

[2]We keep the model, hyperparameters, and setting as proposed in (Clark et al., 2020).

that when trained with larger hop depths, models struggle to solve even slightly less large reasoning tasks. Regarding specifically the *S-RT* dataset, it can be seen from Table 2 (green area), that the performance of the model trained and tested on the 2-hop (88.54) decreases to 66.92 and 64.62 respectively when tested on the 0-hop and 1-hop data. Similar behavior is observed for the model trained on 3-hop. Looking at Table 3, obtained using the SHINet dataset, we can see that the results on the 2-hop test show a similar trend with the *S-RT* dataset: overall, the 2-hop PLMs exhibit better results when tested on 2-hop, but their performances decrease for a simpler task, 1-hop test, for all the three models while we expect at least stable performance. More precisely, we observe a performance decrease of 99.82 $\downarrow$ 96.31, 87.25 $\downarrow$ 77.75, and 99.99 $\downarrow$ 98.77 for XLNet, BERT, and RoBERTa, respectively. The same performance decrease trend is observed compared to the upper bound achieved when testing the 1-hop trained PLMs on the 1-hop test. This might reveal a counter-intuitive and uncontrollable behavior: having in mind the incremental human-style reasoning (Anderson, 1980), one could argue that the ability to solve a k-hop problem implies the ability to solve the $\{k\text{-}1\}$-hop one, but results indicate the contrary. These results are consistent in both datasets suggesting that *PLMs do not incrementally learn by accumulating knowledge*.

In addition, looking at the compositionality generalization from simple to complex tasks, we can see from Table 2 (S-RT) that the performance of models trained on low-depth single-hops (rows $k = 0, 1$) significantly decreases when the hop is deeper (columns $k = 2, 3, 5$) in the test set (e.g., the 1-hop model performance decreases from 90.11 to 50.64 and 37.30 for columns $k = 2, 3$, respectively). However, for depth rows $k = 2, 3, 5$, this trend is less clear. Similarly, Table 3, using the SHINet dataset, shows that 1-hop models manage to obtain strong accuracy scores, over 86.00 in all datasets, indicating that they can deal with their tasks and complex ones. This behavior can be explained by the mix of implicit knowledge (from pre-training) and explicit knowledge (from training), filling the logic gap between tasks as shown by (Talmor et al., 2020b). Moreover, we can observe that RoBERTa-based and XLNet-based PLMs are more effective in both 1-hop and 2-hop configurations, in contrast to BERT-based models, consistently with previous work (Talmor et al., 2020a).

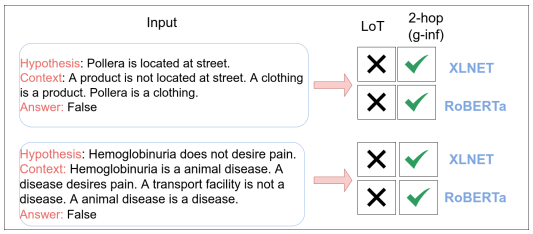

Figure 4: Two examples from the 2-*hop* test set. Both examples are negative (false hypothesis), the first with a positive phrase and the second one with a negative phrase. The guided model correctly predicted both.

4.2 Can Reasoning Models be Guided Toward Incremental Reasoning? (RQ2)

To answer **RQ2**, we compared our models to two different baselines: the *Mixed* model used in RQ1, and the *LoT* model from Talmor et al. (2020b). *LoT* is trained on a $\{k \leq 1\}$ using ConceptNet, WordNet, and Wikidata datasets to combine implicit knowledge acquired in pre-training with explicit rules and facts showing good performance on various types of reasoning tasks. We also trained the *hybrid* model by fine-tuning on the SHINet 2-hop (g-inf) dataset followed by the *LoT* train dataset (1-hop) to show the effect of jointly leveraging implicit knowledge, explicit knowledge (*LoT*), and incremental reasoning. We report in Table 4 the mean accuracy scores of the different distractors.

At a first glance, we can see that our proposed guided model 2-*hop (g-inf)* surpasses all its counterparts for 2 out of 3 settings, namely XLNet and RoBERTa. More precisely, by comparing the performance scores of 2-*hop (g-inf)* to 2-*hop (s-inf)*, we can fairly assess the positive impact of our proposed inference distractors presented in Figure 2 to guide the training toward incremental learning across all the models. For instance, we observed an improvement of 3.53 (0.17), 6.41 (7.98), and 1.17 (0.01) when comparing the 2-*hop (g-inf)* model with 2-*hop (s-inf)* models on the 1-*hop* (2-*hop*) tests with XLNet, BERT, and RoBERTa respectively. We further compare the accuracy scores of 2-*hop (g-inf)* in comparison to 2-*hop (i-inf)* to show the impactful role of the (s-inf) inference distractor to improve the reasoning inference. As it can be seen from Table 4, 2-*hop (g-inf)* increases the performance on both tests by a difference greater than 21.0 in all models. The comparison between the 2-*hop (g-inf)* model to a traditional multi-hop mixing strategy *Mixed* shows the advantage of the

Test (k-hop)	k=1	k=2	k=1	k=2	k=1	k=2
Models →	*XLNet*		*BERT*		*RoBERTa*	
Train ↓						
LoT	98.37∗	99.17	86.28∗	**95.33**∗	99.15∗	98.96
Mixed	98.89	89.06	**99.23**	94.63	99.71	93.44
2-hop						
(i-inf)	60.81	56.77	62.37	57.16	60.48	56.23
(s-inf)	96.31	99.82†	77.75	87.25	98.77	99.99†
(g-inf)	**99.84**∗†	**99.99**∗†	84.16∗	95.23∗	**99.94**∗†	**100**∗†
hybrid	99.18∗†	99.67∗†	86.32∗†	93.85∗	99.31∗†	99.76†

Table 4: Accuracy performances (%) for 2-hop models by varying the inference distractor in the SHINet dataset. † and ∗ indicates statistical significance according to the Almost Stochastic Dominance test over *LoT* and 2-*hop (s-inf)*, respectively.

Train ↓ / Test(k-hop) →		$k=0$	$k=1$	$k=2$
S-RT	2-hop	66.92	64.62	<u>88.54</u>
	3-hop	68.36	64.44	88.47
LoT	2-hop	<u>72.61</u>	65.37	88.40
	3-hop	72.46	64.81	88.56
2-hop *(s-inf)*	2-hop	67.15	64.88	88.42
	3-hop	67.06	64.58	88.25
2-hop *(g-inf)*	2-hop	68.06	65.18	88.05
	3-hop	67.89	64.8	88.11
hybrid	2-hop	71.75	<u>65.49</u>	88.44
	3-hop	**72.47**	**65.14**	**88.60**

Table 5: Accuracy comparing different reasoning models on the S-RT dataset. The best result for each test in <u>underlined</u> and **bold** for 2-hop and 3-hop models, respectively.

incremental inference for most of the settings. Finally comparing our proposed guided model 2-*hop (g-inf)* to the fine-tuned multi-hop strategy *hybrid*, we can observe that our guided model surpasses the hybrid model and *LoT* in 2 out of 3 models, namely (*XLNet* and *RoBERTa*). It is worth noting that this performance is achieved using fewer computational resources; the model can address both tests 2-hop and the simpler 1-hop in an incremental reasoning fashion.

In Figure 4, we show some hand-picked difficult examples from the 2-*hop* test set for the *LoT* model that are especially helped by the guided model 2-*hop (g-inf)*, using XLNet and RoBERTa. Specifically, we observed a positive impact of the distractors to solve false hypothesis examples using a negative phrase.

Additionally, we compare our proposed models with those trained on the *S-RT* dataset used in **RQ1**. We particularly examine if the proposed models still exhibit the observed phenomenon highlighted in the green area from Table 2. Table 5 shows results with mean values after 3 runs and under the

Model	**Def**	**Hype**	**Hypo Syn**		**DQA**	**Avg (%Imp)**
Rand	19.90	19.90	20.20	19.80	20.00	19.96 (-38.6%)
RoB	40.10	32.65	23.15	34.41	32.26	32.51 (-)
LoT	72.41	43.83	40.54	51.85	51.53	52.03 (+60.0%)
1-hop						
(s-inf)	62.65	52.69	38.58	45.80	43.68	48.68 (+49.7%)
2-hop						
(s-inf)	63.65	46.50	36.82	50.52	47.69	49.04 (+50.8%)
(g-inf)	70.86	**51.86**	43.03	**55.29**	52.65	**54.74** (+68.4%)
hybrid	**73.09**	44.70	**46.43**	51.87	**54.72**	*54.16* (+66.6%)

Table 6: Accuracy (%) scores for baselines and multi-hop reasoning models using the validation set of the MCQA dataset. Improvement percentages (%Imp) are given w.r.t. **RoB**ERTa (inoculated).

inoculation technique. The *inoculation technique* from Liu et al. (2019a) was used to avoid overriding the knowledge acquired in our models. The inoculation consists of using a small amount of training data to solve new tasks, overcoming the mismatch between the datasets used in training and fine-tuning (Jiang et al., 2020).

We did a preliminary analysis of the learning curves for each task to determine the right amount of data to use (see Appendix C).

We can see from Table 5 that even when the inoculation is used, the models relying on incremental reasoning (2-*hop (g-inf)* and *hybrid*) overpass the baseline results (*S-RT* and 2-*hop (s-inf)*). Particularly, we see that guiding the model training over hops leads to improvements in lower hop levels ($k = 0, 1$) compared to traditional model training with mixed hops. For instance, for the test $k = 1$, the guided model 2-*hop (g-inf)* improves by 0.56 and 0.36 the model trained with S-RT on hops $k = 2$ and $k = 3$, respectively.

4.3 Do QA Tasks Leverage Incrementally Trained Reasoning Models (RQ3)?

To answer **RQ3**, we used: 1) two QA tasks, namely Multiple Choice Question Answering (MCQA), and Reading Comprehension (RC). We applied the inoculation technique presented before to all the models; 2) the *Random* model, denoted *Rand*, the *RoBERTa* model, denoted *RoB*, and the *LoT* model as baselines. The *RoBERTa* model has been chosen, given its performance superiority as shown in the previous experiments (see Sections 4.1 and 4.2). For datasets examples and illustrations of the tasks, we refer the reader to Appendix B.

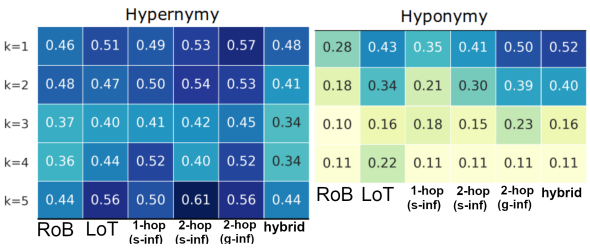

Figure 5: Accuracy values using the hypernymy and hyponymy subsets broken into number of hops k (rows) for the models (columns).

Multiple Choice Question Answering (MCQA).

For MCQA, we re-used a publicly available code[3] as Richardson and Sabharwal (2020) and then applied the inoculation technique (Liu et al., 2019a). We plot the learning curves of each probe for the average of five different runs with random subsets (see Appendix C).

In Table 6, we report the results of our inoculated models. We can see that our models 2-*hop (g-inf)* and *hybrid* achieve the best average performance scores over all the baselines.

To deepen our analysis of the reasoning over increasing numbers of hops, we experimented with our models with the hypernymy and hyponymy subsets, up to 4 and 3 hops levels, respectively. By filtering the numbers of hops, we report the performance variation of our models in Figure 5. We can see that for the hypernymy (resp. hyponymy), the *hybrid* (followed by 2-*hop (g-inf)*) model outperforms all models in all depths but for $k = 4, 5$. Furthermore, we interestingly observe a positive trend toward reducing the performance decrease rate between hop levels when using our proposed guided training approach. For instance, when comparing levels $k = 2$ and $k = 4$, we observe that performance decrease is reduced from 0.14 to 0.01 for 2-*hop (s-inf)* and 2-*hop (g-inf)* respectively. Similarly between the hyponymy levels $k = 2$ and $k = 3$ we can see a performance decrease reduced from 0.18 to 0.16 for *LoT* and 2-*hop (g-inf)* models respectively. This observation clearly indicates the positive impact of incremental reasoning on performance.

Reading Comprehension (RC). For RC, results under inoculation conditions are reported in Table 7. As can be seen, most of the models behave similarly for the Middle set, with 2-*hop (s-inf)* as the most performing model. On the contrary, we can observe a clear improvement for all models on the High set when compared to *RoBERTa*. In this case, the most performing model is 2-*hop (s-inf)* (69.76) closely followed by 2-*hop (g-inf)* (68.72) and *hybrid* (68.56). Therefore, chains of reasoning seem to be a key component of the solution, even if most of the studied multi-hop models correctly capture the needed knowledge.

Finally, we compare the results from Table 6 and Table 7. We observe that model scores are very close on the RC task, even when using different distractors and the number of hops. The uniformity between all models' performances suggests that multi-hop reasoning is not a key component in solving these questions.

[3] https://github.com/yakazimir/semantic_fragments

Models	**Middle (%Imp)**	**High (%Imp)**
RoBERTa	77.18 (-)	59.22 (-)
LoT	77.04 (-0.2%)	68.68 (+16.0%)
1-**hop** ***(s-inf)***	76.56 (-0.8%)	68.94 (+16.0%)
2-**hop** ***(s-inf)***	**77.32** (+0.2%)	**69.76** (+17.8%)
2-**hop** ***(g-inf)***	75.65 (-2.0%)	68.72 (+16.0%)
hybrid	76.37 (-1.0%)	68.56 (+15.8%)

Table 7: Accuracy (%) comparing different reasoning models on the RACE dataset for middle school and high school. Improvement percentages (%Imp) are given w.r.t. RoBERTa.

5 Related Work

Our main study focused on multi-hop reasoning. Recent studies have proposed solutions using decomposition-aggregation approaches (Min et al., 2019) by combining or extending different model architectures to leverage reasoning performance (Feng et al., 2020; Yasunaga et al., 2021; Bauer et al., 2018), creating explained reasoning paths (Ding et al., 2019) or using chain of thought prompting (Wei et al., 2022). In contrast, we focus on leveraging the inner reasoning skills of PLMs, benefiting from their internal architecture and knowledge captured in pre-training. We argue that our results may be a solid alternative to a standard PLM in this kind of work.

There are recent demonstrations that trained PLMs can perform simple reasoning tasks (Talmor et al., 2020a). Even if these models are not naturally good solvers of complex tasks such as multi-hop reasoning (Kassner et al., 2020), they are capable of learning when trained on such tasks (Clark et al., 2020; Richardson and Sabharwal, 2020). However, these training setups propose mixing different depth levels of reasoning, leading to unpredictable results, and, thus, a lack of

model interpretability. They do not let recognize the knowledge captured at each hop level and whether acquired knowledge, if any, is actually reused to solve higher-hop level reasoning. We proposed a single-hop design that lets us analyze the actual contribution of each reasoning level.

Although our work is inspired by the previous literature, it is different from Talmor et al. (2020a) and Talmor et al. (2020b), as they evaluate the inner reasoning capabilities of PLMs in simple reasoning tasks, but we evaluate incrementally trained PLMs for multi-hop reasoning performed on NLP downstream tasks. Similarly, Kassner et al. (2020) evaluate the model reasoning skills controlling the data given in pre-training. In contrast, we analyze how the training data and elements in the context affect the task in a multi-hop scenario.

6 Conclusions

Transformers have been recently gaining increasing attention for reasoning tasks over language. In this paper, we have specifically studied whether we can endow PLMs used in multi-hop reasoning tasks with the ability to incrementally acquire knowledge by following the inference path over the sequence of hops. Our underlying objective is to control the training of PLMs better, leading to more understandable and predictable multi-hop reasoning models. In particular, we have complemented previous findings in the literature by showing that PLMs trained on 1-hop reasoning tasks can extrapolate the reasoning to 2-hops but that 2-hop reasoning models struggle to generalize over slightly simpler 1-hop tasks. Keeping in mind the human-style reasoning from simpler to complex tasks, we advocate incremental reasoning over the structure of the inference path as a step forward. We provide a training data generation strategy that relies critically on inference distractors connecting intermediate relevant facts in the reasoning path. By applying our approach, our models achieve higher or similar performance trends than fine-tuning multi-hop models but consume fewer resources. Furthermore, we show that the incrementally trained multi-hop PLMs are transferable to other QA-based tasks.

Although our experimental settings are limited to low depths of inference ($k = 1, 2$), our findings show both the feasibility and the benefit of incremental reasoning and open new research opportunities. We may potentially extend this work toward the benchmarking of multi-hop reasoning interpretability with the design of baseline models, dataset generation strategies with upper bounds, and evaluation metrics including, but not limited to, inference path recall.

7 Acknowledgments

This work was supported by the ANR-19-CE23-0028 MEERQAT project. This work was granted access to the HPC resources of IDRIS under the allocation 20XX-AD011012638R1 made by GENCI.

References

John R. Anderson. 1980. *Cognitive Psychology and Its Implications*. W.H.Freeman & Co Ltd.

Lisa Bauer, Yicheng Wang, and Mohit Bansal. 2018. Commonsense for Generative Multi-Hop Question Answering Tasks. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 4220–4230, Brussels, Belgium. Association for Computational Linguistics.

Rich Caruana. 1997. Multitask learning. *Mach. Learn.*, 28(1):41–75.

Rahma Chaabouni, Eugene Kharitonov, Diane Bouchacourt, Emmanuel Dupoux, and Marco Baroni. 2020. Compositionality and generalization in emergent languages. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 4427–4442, Online. Association for Computational Linguistics.

Peter Clark, Oyvind Tafjord, and Kyle Richardson. 2020. Transformers as soft reasoners over language. In *Proceedings of the Twenty-Ninth International Joint Conference on Artificial Intelligence, IJCAI-20*, pages 3882–3890. International Joint Conferences on Artificial Intelligence Organization.

Rajarshi Das, Ameya Godbole, Manzil Zaheer, Shehzaad Dhuliawala, and Andrew McCallum. 2019. Chains-of-reasoning at TextGraphs 2019 shared task: Reasoning over chains of facts for explainable multi-hop inference. In *Proceedings of the Thirteenth Workshop on Graph-Based Methods for Natural Language Processing (TextGraphs-13)*, pages 101–117, Hong Kong. Association for Computational Linguistics.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186. Association for Computational Linguistics.

Ming Ding, Chang Zhou, Qibin Chen, Hongxia Yang, and Jie Tang. 2019. Cognitive graph for multi-hop reading comprehension at scale. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 2694–2703, Florence, Italy. Association for Computational Linguistics.

Rotem Dror, Segev Shlomov, and Roi Reichart. 2019. Deep dominance-how to properly compare deep neural models. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 2773–2785.

Yanai Elazar, Hongming Zhang, Yoav Goldberg, and Dan Roth. 2021. Back to square one: Artifact detection, training and commonsense disentanglement in the Winograd schema. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 10486–10500, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Yanlin Feng, Xinyue Chen, Bill Yuchen Lin, Peifeng Wang, Jun Yan, and Xiang Ren. 2020. Scalable multi-hop relational reasoning for knowledge-aware question answering. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1295–1309. Association for Computational Linguistics.

Marti A. Hearst. 1992. Automatic acquisition of hyponyms from large text corpora. In *COLING 1992 Volume 2: The 14th International Conference on Computational Linguistics*.

Zhengbao Jiang, Frank F. Xu, Jun Araki, and Graham Neubig. 2020. How Can We Know What Language Models Know? *Transactions of the Association for Computational Linguistics*, 8:423–438.

Nora Kassner, Benno Krojer, and Hinrich Schütze. 2020. Are Pretrained Language Models Symbolic Reasoners over Knowledge? In *Proceedings of the 24th Conference on Computational Natural Language Learning*, pages 552–564, Online. Association for Computational Linguistics.

Nora Kassner and Hinrich Schütze. 2020. Negated and misprimed probes for pretrained language models: Birds can talk, but cannot fly. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, Online. Association for Computational Linguistics.

Guokun Lai, Qizhe Xie, Hanxiao Liu, Yiming Yang, and Eduard Hovy. 2017. RACE: Large-scale ReAding Comprehension Dataset From Examinations. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 785–794, Copenhagen, Denmark. Association for Computational Linguistics.

Nelson F. Liu, Roy Schwartz, and Noah A. Smith. 2019a. Inoculation by fine-tuning: A method for analyzing challenge datasets. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 2171–2179, Minneapolis, Minnesota. Association for Computational Linguistics.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019b. RoBERTa: A Robustly Optimized BERT Pretraining Approach. *arXiv:1907.11692 [cs]*. ArXiv: 1907.11692.

Kaixin Ma, Filip Ilievski, Jonathan Francis, Yonatan Bisk, Eric Nyberg, and Alessandro Oltramari. 2021. Knowledge-driven data construction for zero-shot evaluation in commonsense question answering. In *35th AAAI Conference on Artificial Intelligence*.

Sewon Min, Victor Zhong, Luke Zettlemoyer, and Hannaneh Hajishirzi. 2019. Multi-hop reading comprehension through question decomposition and rescoring. In *ACL*.

Kyle Richardson and Ashish Sabharwal. 2020. What Does My QA Model Know? Devising Controlled Probes using Expert. *Transactions of the Association for Computational Linguistics*, 8(0):572–588. Number: 0.

Stephen Roller, Douwe Kiela, and Maximilian Nickel. 2018. Hearst Patterns Revisited: Automatic Hypernym Detection from Large Text Corpora. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 358–363, Melbourne, Australia. Association for Computational Linguistics.

Swarnadeep Saha, Prateek Yadav, and Mohit Bansal. 2021. multiPRover: Generating multiple proofs for improved interpretability in rule reasoning. In *NAACL*.

Keisuke Sakaguchi, Ronan Le Bras, Chandra Bhagavatula, and Yejin Choi. 2020. Winogrande: An adversarial winograd schema challenge at scale. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pages 8732–8740.

Apoorv Saxena, Aditay Tripathi, and Partha Talukdar. 2020. Improving Multi-hop Question Answering over Knowledge Graphs using Knowledge Base Embeddings. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 4498–4507, Online. Association for Computational Linguistics.

Satoshi Sekine, Akio Kobayashi, and Kouta Nakayama. 2018. SHINRA: Structuring Wikipedia by Collaborative Contribution. In *1st Conference on Automated Knowledge Base Construction (AKBC 2019)*.

Robyn Speer, Joshua Chin, and Catherine Havasi. 2017. ConceptNet 5.5: An Open Multilingual Graph of General Knowledge. In *Thirty-First AAAI Conference on Artificial Intelligence*.

Alon Talmor, Yanai Elazar, Yoav Goldberg, and Jonathan Berant. 2020a. oLMpics-On What Language Model Pre-training Captures. *Transactions of the Association for Computational Linguistics*, 8:743–758.

Alon Talmor, Oyvind Tafjord, Peter Clark, Yoav Goldberg, and Jonathan Berant. 2020b. Leap-Of-Thought: Teaching Pre-Trained Models to Systematically Reason Over Implicit Knowledge. *Advances in Neural Information Processing Systems*, 33.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in Neural Information Processing Systems*, volume 30, pages 5998–6008. Curran Associates, Inc.

Peifeng Wang, Filip Ilievski, Muhao Chen, and Xiang Ren. 2021. Do Language Models Perform Generalizable Commonsense Inference? In *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*, pages 3681–3688, Online. Association for Computational Linguistics.

Leon Weber, Pasquale Minervini, Jannes Münchmeyer, Ulf Leser, and Tim Rocktäschel. 2019. Nlprolog: Reasoning with weak unification for question answering in natural language. *CoRR*, abs/1906.06187.

Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Ed H. Chi, Quoc Le, and Denny Zhou. 2022. Chain of thought prompting elicits reasoning in large language models. *CoRR*, abs/2201.11903.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Rémi Louf, Morgan Funtowicz, Joe Davison, Sam Shleifer, Patrick von Platen, Clara Ma, Yacine Jernite, Julien Plu, Canwen Xu, Teven Le Scao, Sylvain Gugger, Mariama Drame, Quentin Lhoest, and Alexander M. Rush. 2020. Transformers: State-of-the-art natural language processing. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 38–45, Online. Association for Computational Linguistics.

Zhilin Yang, Zihang Dai, Yiming Yang, Jaime Carbonell, Russ R Salakhutdinov, and Quoc V Le. 2019. Xlnet: Generalized autoregressive pretraining for language understanding. In *Advances in Neural Information Processing Systems*, volume 32. Curran Associates, Inc.

Michihiro Yasunaga, Hongyu Ren, Antoine Bosselut, Percy Liang, and Jure Leskovec. 2021. QA-GNN: Reasoning with language models and knowledge graphs for question answering. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 535–546. Association for Computational Linguistics.

A Computing Infrastructure and Budget

All experiments were performed in a server Dell R740 bi pro Intel Xeon 2630 using Nvidia RTX6000 graphic card. A single training and test took around 100 and 120 minutes under this infrastructure. In summary, to compute the results of RQ1 and RQ2 we used approximately 120 GPU hours.

To compute the results of RQ3, including the inoculation technique, we used approximately 1,730 GPU hours.

B Dataset samples for Downstream Tasks

In Figure 6, we show entry samples for the MCQA (6a) and RACE (6b) tasks.

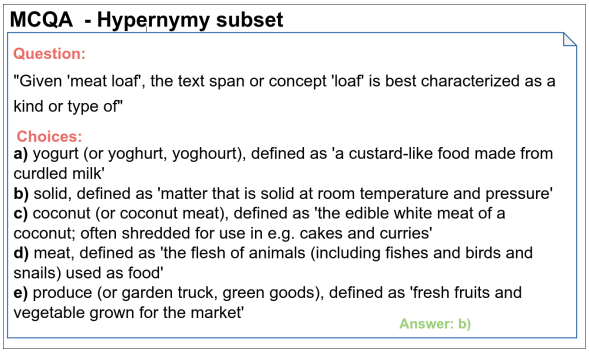

(a)

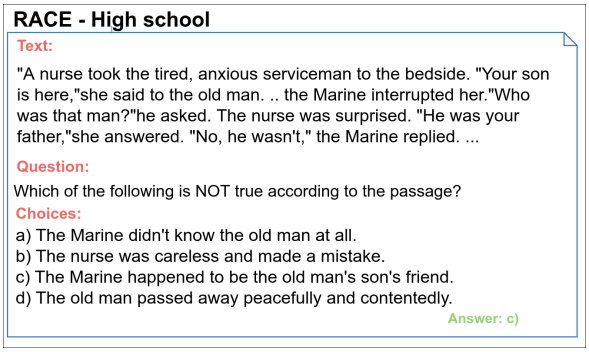

(b)

Figure 6: MCQA (a) and RACE (b) examples.

C Learning Curves from Inoculation Technique

Figure 7 shows the learning curves when applying the inoculation technique for the MCQA and RC tasks. We selected 5,000 as the number of samples with the best performance and smaller training size. Similar analysis was done for the RACE and S-RT datasets with equal conclusion w.r.t. the number of samples.

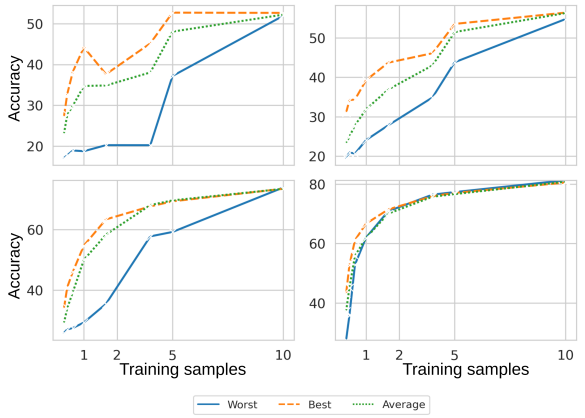

Figure 7: Learning curves for MCQA (upper) and RC (lower) tasks. For MCQA we show the Hypernymy (left) and Synonymy (right) dataset. For RC we show the Middle School (left) and High School (right) datasets. For all curves, the X axis represents the number of training samples (in thousands), and the Y axis, the accuracy score. Average values are reported with 5 runs for MCQA and 3 for RC.

Modeling Hierarchical Reasoning Chains by Linking Discourse Units and Key Phrases for Reading Comprehension

Jialin Chen[1], Zhuosheng Zhang[2,3], Hai Zhao[2,3,*]
[1]School of Mathematical Sciences, Shanghai Jiao Tong University
[2]Department of Computer Science and Engineering, Shanghai Jiao Tong University
[3]Key Laboratory of Shanghai Education Commission for Intelligent Interaction and Cognitive Engineering, Shanghai Jiao Tong University
sjtuchenjl@sjtu.edu.cn, zhangzs@sjtu.edu.cn, zhaohai@cs.sjtu.edu.cn

Abstract

Machine reading comprehension (MRC) poses new challenges over logical reasoning, which aims to understand the implicit logical relations entailed in the given contexts and perform inference over them. Due to the complexity of logic, logical relations exist at different granularity levels. However, most existing methods of logical reasoning individually focus on either entity-aware or discourse-based information but ignore the hierarchical relations that may even have mutual effects. In this paper, we propose a holistic graph network (HGN) which deals with context at both discourse level and word level, as the basis for logical reasoning, to provide a more fine-grained relation extraction. Specifically, node-level and type-level relations, which can be interpreted as bridges in the reasoning process, are modeled by a hierarchical interaction mechanism to improve the interpretation of MRC systems. Experimental results on logical reasoning QA datasets (ReClor and LogiQA) and natural language inference datasets (SNLI and ANLI) show the effectiveness and generalization of our method, and in-depth analysis verifies its capability to understand complex logical relations.

1 Introduction

Machine reading comprehension (MRC) is a challenging task that requires machines to answer a question according to given passages (Hermann et al., 2015; Rajpurkar et al., 2016, 2018; Lai et al., 2017). A variety of datasets have been introduced to push the development of MRC to a more complex and more comprehensive pattern, such as conversational MRC (Reddy et al., 2019; Choi et al., 2018), multi-hop MRC (Yang et al., 2018), and commonsense reasoning (Davis and Marcus, 2015; Bhagavatula et al., 2020; Talmor et al., 2019; Huang et al., 2019). In particular, some recent multi-choice MRC datasets pose even greater challenges to the logical reasoning ability of models (Yu et al., 2020; Liu et al., 2020a) which are not easy for humans to do well, either. Firstly, all the supporting details needed for reasoning are provided by the context, which means there is no additional commonsense or available domain knowledge. Secondly, it is a task of answer selection rather than answer retrieval, which means the best answer is chosen according to their logical fit with the given context and the question, rather than retrieved directly from the context according to the similarity between answers and context. Most importantly, the relations entailed in the contexts are much more complex than that of previous MRC datasets owing to the complexity of logic, which is hard to define and formulate. Without a targeted design for those challenges, existing pre-trained models, e.g., BERT, RoBERTa, fail to perform well in such kind of logical reading comprehension systems (Yu et al., 2020; Liu et al., 2020a).

Example (taken from ReClor dataset)
Context: Most lecturers who are effective teachers are eccentric, but some non-eccentric lecturers are very effective teachers. In addition, every effective teacher is a good communicator.
Question:
Which one of the following statements follows logically from the statements above?
Options:
A: Most lecturers who are good communicators are eccentric.
B: Some non-eccentric lecturers are effective teachers but are not good communicators.
C: All good communicators are effective teachers.
D: Some good communicators are eccentric. ✓

Figure 1: An example from Reclor dataset. The example mainly talks about "effective teachers, non-eccentric, eccentric, good communicator".

Logical reasoning MRC tasks are usually to find an appropriate answer, given a set of context and question. Figure 1 shows an example from ReClor dataset (Yu et al., 2020) which requires logical reasoning ability to make the correct predictions. As

*Corresponding author. This work was supported in part by the Key Projects of National Natural Science Foundation of China under Grants U1836222 and 61733011.

Proceedings of the 29th International Conference on Computational Linguistics, pages 1467–1479
October 12–17, 2022.

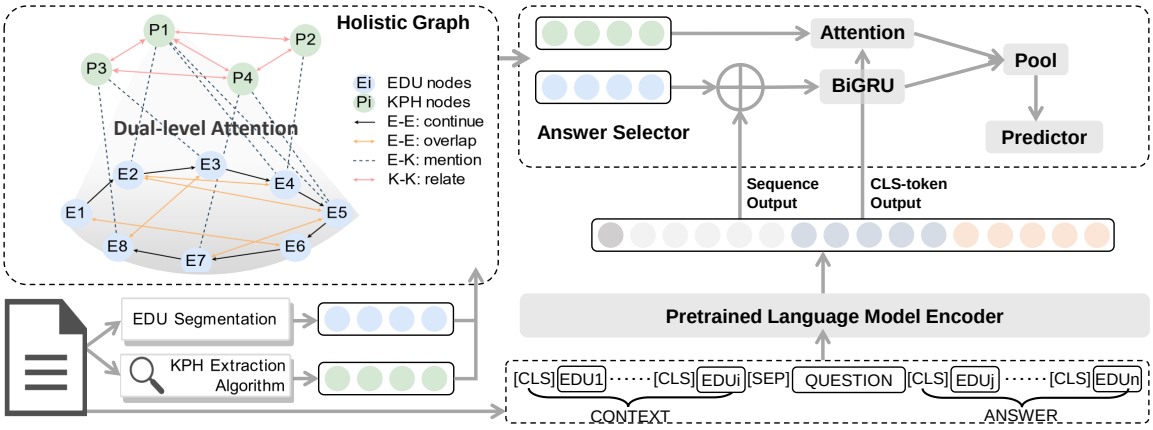

Figure 2: An overview of our proposed holistic graph-based reasoning model.

humans, to solve such problems, we usually go through the following steps. Firstly, we divide the context into several fragments and figure out the logical relations between each clause, such as transition, continuity, contrast, etc. Secondly, we extract the important elements in the context, namely, the objects and topics described by the context, and construct the logical graph with these significant elements. Finally, we need to compare the answer statement to the mentioned part in the context and assess its logical fit with the given context.

Most existing methods of logical reasoning MRC focus on either entity-aware or discourse-based information but ignore the hierarchical relations that may have mutual effects (Yu et al., 2020; Liu et al., 2020a; Wang et al., 2021; Huang et al., 2021; Ouyang et al., 2021b). Motivated by the observation above, we model logical reasoning chains based on a newly proposed holistic graph network (HGN) that incorporates the information of element discourse units (EDU) (Gao et al., 2020; Ouyang et al., 2021a) and key phrases (KPH) extracted from context and answer, with effective edge connection rules to learn both hierarchical features and interactions between different granularity levels.

Our contributions are summarized as follows. (1) We design an extraction algorithm to extract EDU and KPH elements as the critical basic for logical reasoning. (2) We propose a novel holistic graph network (HGN) to deal with context at both discourse and word level with hierarchical interaction mechanism that yields logic-aware representation for reasoning. (3) Experimental results show our model's strong performance improvements over baselines, across multiple datasets on logical reasoning QA and NLI tasks. The analysis demonstrates that our model has a good generalization and transferability, and achieves higher accuracy with less training data.

2 Methodology

Logical reasoning MRC tasks aim to find the best answer among several given options based on a piece of context that entails logical relations. Formally, given a natural language context C, a question Q, and four potential answers A=$\{A_1, A_2, A_3, A_4\}$. We concatenate them as $\{C, Q, A_i\}$ pairs. To incorporate the principle of human inference into our method, we propose a holistic graph network (HGN) as shown in Figure 2. Our model works as follows. First, we use EDU and KPH extraction algorithm to get necessary KPH nodes ($\{P_j\}$) and EDU nodes ($\{E_j\}$) from the given pairs. They contain information with different granularity levels and complement each other. Based on the extracted KPH-EDU interaction information and pre-defined rules, we construct the holistic graph. The process of constructing the holistic graph is shown in Figure 3. Then we measure the interaction between $\{E_j\}$ and $\{P_j\}$ to obtain logic-aware representations for reasoning.

2.1 Logical Chain Construction

Element Discourse Units (EDU) We use clause-like text spans delimited by logical relations to construct the rhetorical structure of texts. These clause-like discourses can be regarded as element units that reveal the overall logic and emotional tone of the text. For example, conjunctions like "`because`" indicate a causal relation which means the following discourse is likely to be the conclusion we need to

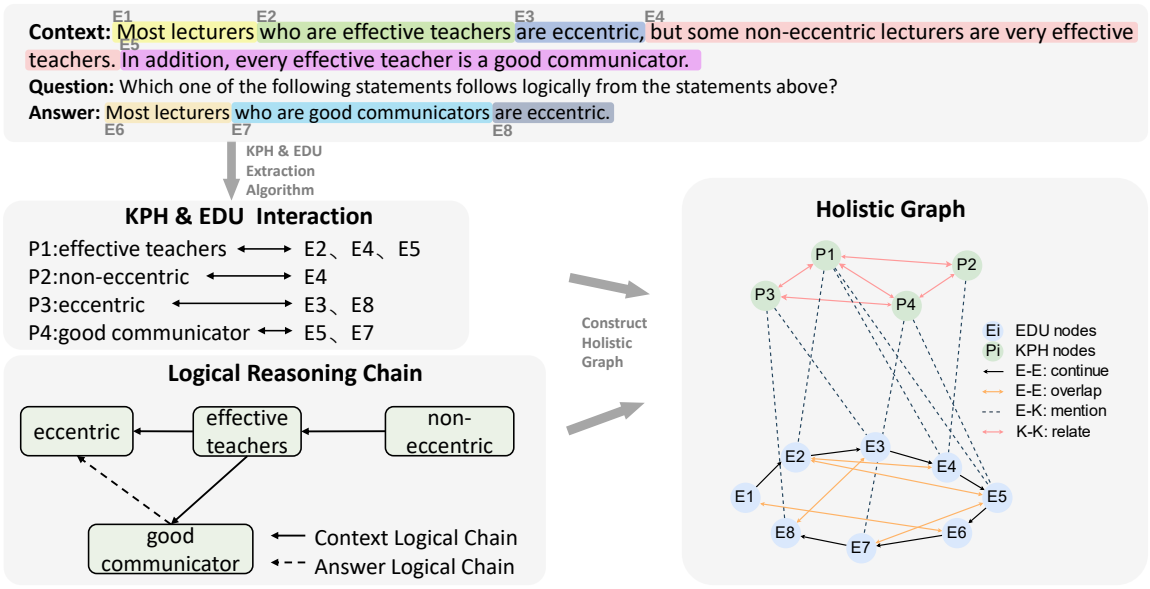

Figure 3: Process of constructing the holistic graph, using KPH-EDU Interaction information and pre-defined rules.

pay attention to. Parenthesis and clauses like "who are effective teachers" in Figure 3 play a complementary role in context. Also, punctuation indicates a pause or an end of a sentence, containing semantic transition and turning point implicitly. We use an open segmentation tool, SEGBOT (Li et al., 2018), to identify the element discourse units (EDUs) from the concatenation of context and answer, ignoring the question whose structure is simple. Conjunctions (e.g., "because", "however"), punctuation and the beginning of parenthesis and clauses (e.g., "which", "that") are usually the segment points. They are considered as explicit discourse-level logical relations.

To get the initial embedding of EDUs, we insert an external [CLS] symbol at the start of each discourse, and add a [SEP] symbol at the end of every type of inputs. Then we use RoBERTa to encode the concatenated tokens. The encoded [CLS] token represents the following EDU. Therefore, we get the initial embedding of EDUs.

Key Phrase (KPH) Key Phrases, including keywords here, play an important role in context. They are usually the object and principle of a context. We use the sliding window to generate n-gram word list, filtering according to the Stopword list, POS tagging, the length of the word, and whether it contains any number.[1] The filtering process is based on the following two main criteria:

(1) If the n-gram contains a stop word or a number, then delete it.

(2) If the length of word is less than the threshold value m, delete it, and if the n-gram length is 1, then only the noun, verb, and adjective are retained.

Then, we calculate the TF-IDF features of each n-gram, and select the top-k n-gram as key phrases. k is a hyper-parameter to control the number of KPHs. We restore the selected tokens and retrieve the original expressions containing the key phrase from the original text. For example, as in Figure 3, "eccentric" is one of the KPHs, while we retrieve the original expression "eccentric" and "non-eccentric" from the original text. [2]

Given the token embedding sequence $K_i = \{t_1, \ldots, t_n\}$ of a KPH with length n, its initial embedding is obtained by

$$P_i = \frac{1}{|K_i|} \sum_{t_l \in K_i} t_l. \quad (1)$$

Holistic Graph Construction Formally, every input sample is a triplet that consists of a context, a question and a candidate answer. EDU and KPH nodes are extracted in the above way. As shown in Figure 3, we construct a holistic graph with two types of nodes: EDU Nodes (in blue) and KPH Nodes (in green). For edge connections, there are four distinct types of edges between pairs of nodes.

[1]The stop list is derived by the open-source toolkit Gensim: https://radimrehurek.com/gensim/. The POS tagging is derived by the open-source toolkit NLTK: https://www.nltk.org/.

[2]The complete algorithm is given in Appendix A.

• EDU-EDU continue: the two nodes are contextually associated in the context and answer. This type of edge is directional.

• EDU-EDU overlap: the two nodes contain the same KPH. This type of edge is bidirectional.

• EDU-KPH mention: the EDU mentions the KPH. This type of edge is bidirectional.

• KPH-KPH relate: the two nodes are semantically related. We define two types of semantic relations. One is that the two KPHs are retrieved by the same n-gram as described above. The other one is that the Cosine similarity between the two KPH nodes is greater than a threshold. This type of edge is bidirectional and can capture the information of word pairs like synonyms and antonyms.

The construction of the graph is based on intuitive rules, which will not introduce extra parameters or increase model complexity. A further parameter comparison is given in Table 4.

2.2 Hierarchical Interaction Mechanism

Considering a specific node in the holistic graph, neighboring nodes in the same type may carry more salient information, thus affecting each other in a direct way. In the process, the neighboring nodes in the different types may also interact with each other. To capture both the node-level and type-level attention, we apply a Hierarchical Interaction Mechanism to the update of the graph network's representations.

Graph Preliminary Formally, consider a graph $G = \{V, E\}$, where V and E represent the sets of nodes and edges respectively. A is the adjacency matrix of the graph. $A_{ij} > 0$ means there is an edge from the i-th node to the j-th node. We introduce $A' = A + I$ to take self-attention into account. In order to avoid changing the original distribution of the feature when multiplying with the adjacency matrix, we normalize A', set $\tilde{A} = D^{-\frac{1}{2}} A' D^{-\frac{1}{2}}$ where D is the degree matrix of the graph. $D = diag\{d_1, d_2 \ldots, d_n\}$, d_i is the number of edges attached to the i-th node.

Now, we calculate the attention score from node v' to node v in the following steps.

Type Attention Vector We use $T(\tau)$ to represent all nodes that belong to type τ, and $N(v)$ to represent all neighboring nodes that are adjacent to v. T is the set of types. Assume that node v belongs to $T(\tau)$, h_μ is the feature of node μ, h_τ is the feature of type τ which is computed by

$$h_\tau = \sum_{\mu \in T(\tau)} \tilde{A}_{v\mu} W h_\mu. \tag{2}$$

Using the feature of type and node v, we compute the attention score of type τ as:

$$e_\tau = \sigma(\mu_\tau^T \cdot [W h_v \parallel W_\tau h_\tau]). \tag{3}$$

Then, type-level attention weights α_τ is obtained by normalizing the attention scores across all the types T with the softmax function. σ is an activate function such as leaky-ReLU.

$$\alpha_\tau = \frac{\exp(\sigma(\mu_\tau^T \cdot [W h_v \parallel W_\tau h_\tau]))}{\sum_{\tau' \in T} \exp(\sigma(\mu_{\tau'}^T \cdot [W h_v \parallel W_\tau h_{\tau'}]))}. \tag{4}$$

Node Attention Vector α_τ shows the importance of nodes in type τ to node v. While computing the attention score of node v' that is adjacent to node v, we multiply that by the type attention weights α_τ (assume v' belongs to type τ). Similarly, node attention weights are obtained by the softmax function across all neighboring nodes.

$$e_{vv'} = \sigma(\nu^T \cdot \alpha_\tau [W h_v \parallel W h_{v'}]), \tag{5}$$

$$\alpha_{vv'} = \frac{\exp(e_{vv'})}{\sum_{i \in N(v)} \exp(e_{vi})}, \tag{6}$$

where $\parallel$ is the concatenation operator and $\alpha_{vv'}$ is the attention weight from node v' to v.

Update of Node Representation Let $h_v^{(l)}$ be the representation of the node v at the l-th layer. Then the layer-wise propagation rule is as follows:

$$h_v^{(l+1)} = \sigma(\sum_{v' \in N(v)} \alpha_{vv'} W h_{v'}^{(l)}). \tag{7}$$

2.3 Answer Selector

To predict the best answer that fits the logic entailed in the context, we extract the node representations of the last layer of the graph network and feed them into the downstream predictor. For EDU nodes, since the node order implies the occurrence order in the context, we align them with the output of sequence embedding and add to it as a residual part. Therefore, we feed them into a bidirectional gating recurrent unit (BiGRU).

$$\tilde{H}_E = \text{BiGRU}(H_E + H_{sent}) \in \mathbb{R}^{l \times d}, \tag{8}$$

where $H_E = [h_{v'_1}, h_{v'_2}, \ldots, h_{v'_l}] \in \mathbb{R}^{l \times d}$, v'_i belongs to type EDU. l and d are the sequence length and the feature dimension respectively. H_{sent} is the output of sequence embedding.

For KPH nodes, we first expand the embedding of the first `[CLS]` token to size $1 \times d$, denoted as H_c. Then, we feed the embedding of `[CLS]` token and features of KPH nodes $H_K = [h_{v_1}, h_{v_2}, \ldots, h_{v_n}] \in \mathbb{R}^{n \times d}$ (v_i is of KPH type) into an attention layer.

$$\begin{aligned} \alpha_i &= w_\alpha^T [H_c \parallel h_{v_i}] + b_\alpha \in \mathbb{R}^1, \\ \tilde{\alpha_i} &= \text{softmax}(\alpha_i) \in [0, 1], \\ \tilde{H}_c &= W_c \sum_i \tilde{\alpha_i} h_{v_i} + b_c \in \mathbb{R}^{1 \times d}, \end{aligned} \tag{9}$$

where $\tilde{\alpha_i}$ is the attention weight of node feature h_{v_i}. w_α, b_α, W_c, and b_c are parameters.

The output of BiGRU and the output of attention layer are concatenated and go through a pooling layer, followed by an MLP layer as the predictor. We take a weighted sum of the concatenation as the pooling operation. The predictor is a two-layer MLP with a tanh activation. Specially, coarse-grained and fine-grained features are further fused here to extract more information.

$$\tilde{H} = W_p[\tilde{H}_E \parallel \tilde{H}_c], \quad p = \text{MLP}(\tilde{H}) \in \mathbb{R}, \tag{10}$$

where W_p is a learnable parameter, $\parallel$ is the concatenation operator. For each sample, we get $P = [p_1, p_2, p_3, p_4]$, p_i is the probability of i-th answer predicted by model.

The training objective is the cross entropy loss:

$$\mathcal{L} = -\frac{1}{N} \sum_i^N \log \text{softmax}(p_{y_i}), \tag{11}$$

where y_i is the ground-truth choice of sample i. N is the number of samples.

3 Experiment

3.1 Dataset

Our evaluation is based on logical reasoning MRC benchmarks (ReClor (Yu et al., 2020) and LogiQA (Liu et al., 2020a)) and natural language inference benchmarks (SNLI (Bowman et al., 2015) and ANLI (Nie et al., 2020)). ReClor contains 6,138 multiple-choice questions modified from standardized tests. LogiQA has more instances (8678 in total) and is derived from expert-written questions for testing human logical reasoning ability (Liu et al., 2020a). To assess the generalization of models on NLI tasks, we test our model on the Stanford Natural Language Inference (SNLI) dataset, which contains 570k human annotated sentence pairs. The Adversarial Natural Language Inference (ANLI) is a new large-scale NLI benchmark dataset, where the instances are chosen to be difficult for the state-of-the-art models such as BERT and RoBERTa. It can be used to evaluate the generalization and robustness of the model.[3]

Implementation details and parameter selection are reported in Appendix C for reproduction.[4]

3.2 Main Result

3.2.1 Results on Logical QA

Table 1 presents the detailed results on the development set and the test set of both ReClor and LogiQA datasets. We observe consistent improvements over the baselines. **HGN**$_{\text{RoBERTa(B)}}$ reaches 51.4% of test accuracy on ReClor, and 35.0% of test accuracy on LogiQA, outperforming other existing models. **HGN**$_{\text{RoBERTa(L)}}$ reaches 58.7% of test accuracy on ReClor, therein 77.7% on Easy subset and 43.8% on Hard subset, and 39.9% on LogiQA. **HGN**$_{\text{DeBERTa}}$ achieves 72.3% on the test set of ReClor and 44.2% on LogiQA. If using the same pre-trained language models as the backbones, our proposed model achieves the state-of-the-art results on both ReClor and LogiQA, without extra human annotations. Our model shows great improvement over this task by better utilizing the interaction information , which is ignored by most existing methods.

3.2.2 Results on general NLI tasks

To verify the generality of our model, we conduct experiments on two widely used entailment datasets for NLI: SNLI and ANLI, in which existing models rarely emphasized the modeling of logical relations. Table 2 compares the performances of **HGN** and baseline models on the SNLI dataset with the same proportion of training data for fine-tuning. We observe that when given a limited number of training data, our **HGN** has faster adaptation than baseline models as evidenced by higher performances in low-resource regimes (e.g., 0.1%, 1%, and 10% of the training data used). **HGN** also outperforms BERT$_{\text{BASE}}$ by 0.3% and RoBERTa$_{\text{LARGE}}$

[3]The statistics information of these datasets are given in Appendix B.

[4]Our source codes is available at `https://github.com/Cather-Chen/Logical-Reasoning-Graph`.

Model	ReClor				LogiQA	
	Dev	Test	Test-E	Test-H	Dev	Test
Human ♢	-	63.0	57.1	67.2	-	86.0
RoBERTa$_{\text{BASE}}$ ♢	55.0	48.5	71.1	30.7	33.3⋆	32.7⋆
HGN$_{\text{RoBERTa(B)}}$	56.3$^{(\uparrow 1.3)}$	51.4$^{(\uparrow 2.9)}$	75.2$^{(\uparrow 4.1)}$	32.7$^{(\uparrow 2.0)}$	39.5$^{(\uparrow 6.2)}$	35.0$^{(\uparrow 2.3)}$
RoBERTa$_{\text{LARGE}}$ ♢	62.6	55.6	75.5	40.0	35.0	35.3
DAGN ♢	65.2	58.2	76.1	44.1	35.5	38.7
DAGN (Aug) ♢	65.8	58.3	75.9	44.5	36.9	39.3
LReasoner$_{\text{RoBERTa}}$ ♠	66.2	62.4	81.4	47.5	38.1	40.6
- data augmentation ♠	65.2	58.3	78.6	42.3	-	-
HGN$_{\text{RoBERTa(L)}}$	66.4$^{(\uparrow 3.8)}$	58.7$^{(\uparrow 3.1)}$	77.7$^{(\uparrow 2.2)}$	43.8$^{(\uparrow 3.8)}$	40.1$^{(\uparrow 5.1)}$	39.9$^{(\uparrow 4.6)}$
DeBERTa ♠	74.4	68.9	83.4	57.5	44.4	41.5
LReasoner$_{\text{DeBERTa}}$ ♠	74.6	71.8	83.4	**62.7**	**45.8**	43.3
HGN$_{\text{DeBERTa}}$	**76.0**$^{(\uparrow 1.6)}$	**72.3**$^{(\uparrow 3.4)}$	**84.5**$^{(\uparrow 1.1)}$	**62.7**$^{(\uparrow 5.2)}$	44.9$^{(\uparrow 0.5)}$	**44.2**$^{(\uparrow 2.7)}$

Table 1: Experimental results (Accuracy: %) of our model compared with baseline models on ReClor and LogiQA datasets. Test-E and Test-H denote Test-Easy and Test-Hard subclass of the ReClor dataset respectively. The results in **bold** are the best performance of all models. ♢ indicates that the results are given by Huang et al. (2021), ♠ indicates the results are given by Wang et al. (2021), ⋆ means that the results come from our own implementation. ROBERTA(L) and ROBERTA(B) denotes RoBERTa-large and RoBERTa-base, respectively.

% data used	0.1%		1%		10%		100%	
	Dev	Test	Dev	Test	Dev	Test	Dev	Test
BERT$_{\text{BASE}}$	73.2	70.4	77.9	76.8	84.2	83.9	90.8	90.7
RoBERTa$_{\text{LARGE}}$	84.8	82.0	**87.6**	87.0	89.5	88.8	92.2	91.0
HGN$_{\text{BERT(B)}}$	75.8$^{(\uparrow 2.6)}$	75.4$^{(\uparrow 5.0)}$	81.1$^{(\uparrow 3.2)}$	80.3$^{(\uparrow 3.5)}$	85.4$^{(\uparrow 1.2)}$	83.9$^{(\uparrow 0.0)}$	91.3$^{(\uparrow 0.5)}$	91.0$^{(\uparrow 0.3)}$
HGN$_{\text{RoBERTa(L)}}$	**85.4**$^{(\uparrow 0.6)}$	**83.5**$^{(\uparrow 1.5)}$	**87.6**$^{(\uparrow 0.0)}$	**87.3**$^{(\uparrow 0.3)}$	**90.2**$^{(\uparrow 0.7)}$	**89.4**$^{(\uparrow 0.6)}$	**92.3**$^{(\uparrow 0.1)}$	**91.5**$^{(\uparrow 0.5)}$

Table 2: Experimental results (Accuracy: %) on the SNLI dataset. We randomly generate the training dataset with limited size, without changing the size of Dev. and Test set. BERT(B) and ROBERTA(L) denote BERT-base and RoBERTa-large respectively.

by 0.5% on the full SNLI. We assess the model's robustness against adversarial attacks, using a standard adversarial NLP benchmark: ANLI, as shown in Table 3. A1, A2 and A3 are three rounds with increasing difficulty and data size. ANLI refers to the combination of A1, A2 and A3. **HGN**$_{\text{RoBERTa(L)}}$ gains a 15.2% points in test accuracy of ANLI over RoBERTa$_{\text{LARGE}}$, creating state-of-the-art results on all rounds. Results show that our model has a comprehensive improvement over baseline models, in aspects of faster adaption, higher accuracy and better robustness.

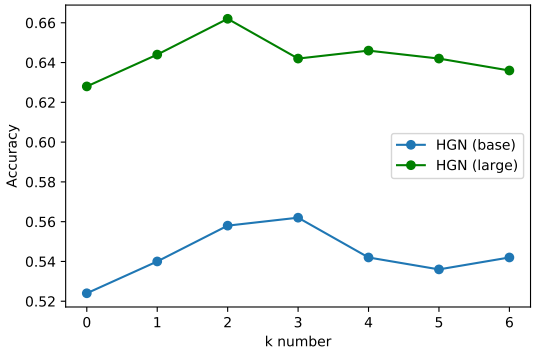

Figure 4: Dev. accuracy on the ReClor dataset as the number of KPH nodes changes.

3.3 More Results

Interpretation of k In this part, we investigate the sensitivity of parameter k, which is the number of KPH node. Figure 4 shows the accuracies on the development set of our proposed model with different numbers of KPH nodes, which are extracted according to TF-IDF weights. We observe that $k = 2$ or $k = 3$ is an appropriate value for our model. This is consistent with our intuition that a

Model	Dev				Test			
	A1	A2	A3	ANLI	A1	A2	A3	ANLI
RoBERTa$_{\text{LARGE}}$	74.1	50.8	43.9	55.5	73.8	48.9	44.4	53.7
ALUM♠	73.3	53.4	48.2	57.7	72.3	52.1	48.4	57.0
InfoBERT♢	76.4	51.7	48.6	58.3	75.5	51.4	49.8	58.3
HGN$_{\text{RoBERTa(L)}}$	**76.7**$^{(\uparrow 2.6)}$	**69.3**$^{(\uparrow 18.5)}$	**74.5**$^{(\uparrow 30.6)}$	**71.3**$^{(\uparrow 15.8)}$	**79.5**$^{(\uparrow 5.7)}$	**63.4**$^{(\uparrow 14.5)}$	**76.3**$^{(\uparrow 31.9)}$	**68.9**$^{(\uparrow 15.2)}$

Table 3: Experimental results (Accuracy: %) on the ANLI dataset. Both ALUM and InfoBERT take RoBERTa-large as the backbone model. ♠ means the results from Liu et al. (2020b). ♢ means the results from Wang et al. (2020).

Model	RoBERTa	DAGN	**HGN**
Params	356.4M	396.2M	373.4M

Table 4: Statistics of models' parameters

paragraph will have 2 to 3 key phrases as its topic. When k is too small or large, the accuracy of the model does not perform well.

Model Complexity With well-defined construction rules and an appropriate architecture, our model enjoys the advantage of high performance with fewer parameters. We display the statistics of model's parameters in Table 4. Compared with the baseline model (RoBERTa$_{\text{LARGE}}$), the increase of our model's parameters is no more than 4.7%. Particularly, our model contains fewer parameters and achieves better performance than DAGN.

3.4 Ablation

We conduct a series of ablation studies on Graph Construction, Hierarchical Interaction Mechanism and Answer Selector. Results are shown in Table 5. All models use RoBERTa-base as the backbone.

Holistic Graph Construction The Holistic Graph in our model contains two types of nodes and four types of edges. We remove the nodes of EDU and KPH respectively and the results show that the removal hurts the performance badly. The accuracies drop to 55.8% and 53.9%. Furthermore, we delete one type of edge respectively. The removal of edge type destroys the integrity of the network and may ignore some essential interaction information between EDUs and KPHs, thus causing the drop of the performance.

Hierarchical Interaction Mechanism Hierarchical Interaction Mechanism helps to capture the information contained in different node types. When we remove the type-level attention, the model is equivalent to a normal Graph Attention Network (GAT), ignoring the heterogeneous information. As a result, the performance drops to 54.8%. When we remove both types of attention, the performance drops to 55.7%.

Model	Accuracy (%)
HGN$_{\text{BASE}}$	**56.3**
Graph Construction	
- EDU	55.8 (↓0.5)
- KPH	53.9 (↓2.4)
- edge type: E-E continue	53.0 (↓3.3)
- edge type: E-E overlap	54.0 (↓2.3)
- edge type: E-K mention	54.2 (↓2.1)
Hierarchical Interaction	
- type-level attention (i.e. GAT)	54.8 (↓1.5)
- both (i.e. GCN)	55.7 (↓0.6)
Answer Selector	
- BiGRU	53.2 (↓3.1)
- Attention layer	55.0 (↓1.3)

Table 5: Ablation results on the dev set of ReClor.

Answer Selector We make two changes to the answer selector module: (1) deleting the BiGRU, (2) deleting the attention layer. For (1), the output of EDU features concatenates with the output of the attention layer directly and then are fed into the downstream pooling layer. For (2), we ignore the attention between the KPH features and the whole sentence-level features. The resulting accuracies of (1) and (2) drop to 53.2% and 55%, which verify that the further fusion of features with different granularity is necessary in our proposed model.

We further analysed the examples that are predicted correctly by our model but not by baselines, and found that the powerful pre-trained language models, such as RoBERTa, would bias for answers with higher similarity to the context or those containing more overlapping words. The model itself does not understand the logical relations, but only

compares their common elements for prediction. Instead, our model can not only match synonymic expressions, but also make logical inferences by separating sentences into EDUs and extracting key phrases and establishing logical relations between them. An example is shown in Appendix D.

4 Related Work

4.1 Machine Reading Comprehension

MRC is an AI challenge that requires machines to answer questions based on a given passage, which has aroused great research interests in the last decade (Hermann et al., 2015; Rajpurkar et al., 2016, 2018; Lai et al., 2017). Although recent systems have reported human-parity performance on various benchmarks (Zhang et al., 2020a; Back et al., 2020; Zhang et al., 2021) such as SQuAD (Rajpurkar et al., 2016, 2018) and RACE (Lai et al., 2017), whether the machine has necessarily achieved human-level understanding remains controversial (Zhang et al., 2020b; Sugawara et al., 2021). Recently, there is increasing interest in improving machines' logical reasoning ability, which can be categorized into symbolic approaches and neural approaches. Notably, analytical reasoning machine (AMR) (Zhong et al., 2021) is a typical symbolic method that injects human prior knowledge to deduce legitimate solutions.

4.2 Logical Reasoning

Neural and symbolic methods have been studied for logical reasoning (Garcez et al., 2015; Besold et al., 2017; Chen et al., 2019b; Ren and Leskovec, 2020; Huang et al., 2021). Compared with the neural methods for logical reasoning, symbolic approaches like (Wang et al., 2021) rely heavily on dataset-related predefined patterns which entails massive manual labor, greatly reducing the generalizability of models. Also, it could introduce propagated errors since the final prediction depends on the intermediately generated functions. Even if one finds the gold programs, executing the program is quite a consuming work as the search space is quite large and not easy to prune. Therefore, we focus on the neural research line in this work, to capture the logic clues from the natural language texts, without the rely on human expertise and extra annotation.

Since the logical reasoning MRC task is a new task that there are only a few latest studies, we broaden the discussion to scope of the related tasks that require reasoning, such as commonsense reasoning (Davis and Marcus, 2015; Bhagavatula et al., 2020; Talmor et al., 2019; Huang et al., 2019), multi-hop QA (Yang et al., 2018) and dialogue reasoning (Cui et al., 2020). Similar to our approach of discovering reasoning chains between element discourse and key phrases, Fang et al. (2020) proposes a hierarchical graph network (HGN) that helps to multi-hop QA. Our method instead avoids the incorporation of external knowledge and designs the specific pattern for logical reasoning. Discourse-aware graph network (DAGN) proposed by Huang et al. (2021) also uses discourse relations to help logical reasoning. However, only modeling the relation between sentences will ignore more fine-grained information. Focal Reasoner proposed by Ouyang et al. (2021b), covering global and local knowledge as the basis for logic reasoning, is also an effective approach. In contrast, our work is more heuristic and has a lighter architecture.

Previous approaches commonly consider the entity-level, sentence-level relations, or heavily rely on external knowledge and fail to capture important interaction information, which are obviously not sufficient to solve the problem (Qiu et al., 2019; Ding et al., 2019; Chen et al., 2019a). Instead, we take advantages of inter-sentence EDUs and intra-sentence KPHs, to construct hierarchical interactions for reasoning. The fine-grained holistic features are used for measuring the logical fitness of the candidate answers and the given context. As our method enjoys the benefits of modeling reasoning chains from riddled texts, our model can be easily extended to other types of reasoning and inference tasks, especially where the given context has complex discourse structure and logical relations, like DialogQA, multi-hop QA and other more general NLI tasks. We left all the easy empirical verification of our method as future work.

5 Conclusion

This paper presents a novel method to guide the MRC model to better perform logical reasoning tasks. We propose a holistic graph-based system to model hierarchical logical reasoning chains. To our best knowledge, we are the first to deal with context at both discourse level and phrase level as the basis for logical reasoning. To decouple the interaction between the node features and type features, we apply hierarchical interaction mechanism to yield the appropriate representation for reading comprehension. On the logical QA benchmarks (ReClor,

LogiQA) and natural language inference benchmarks (SNLI and ANLI), our proposed model has been shown effective by significantly outperforming the strong baselines.

References

Seohyun Back, Sai Chetan Chinthakindi, Akhil Kedia, Haejun Lee, and Jaegul Choo. 2020. Neurquri: Neural question requirement inspector for answerability prediction in machine reading comprehension. In *8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, April 26-30, 2020*. OpenReview.net.

Tarek R Besold, Artur d'Avila Garcez, Sebastian Bader, Howard Bowman, Pedro Domingos, Pascal Hitzler, Kai-Uwe Kühnberger, Luis C Lamb, Daniel Lowd, Priscila Machado Vieira Lima, et al. 2017. Neural-symbolic learning and reasoning: A survey and interpretation. *arXiv preprint arXiv:1711.03902*.

Chandra Bhagavatula, Ronan Le Bras, Chaitanya Malaviya, Keisuke Sakaguchi, Ari Holtzman, Hannah Rashkin, Doug Downey, Wen-tau Yih, and Yejin Choi. 2020. Abductive commonsense reasoning. In *8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, April 26-30, 2020*. OpenReview.net.

Samuel R. Bowman, Gabor Angeli, Christopher Potts, and Christopher D. Manning. 2015. A large annotated corpus for learning natural language inference. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 632–642, Lisbon, Portugal. Association for Computational Linguistics.

Jifan Chen, Shih-ting Lin, and Greg Durrett. 2019a. Multi-hop question answering via reasoning chains. *arXiv preprint arXiv:1910.02610*.

Xinyun Chen, Chen Liang, Adams Wei Yu, Denny Zhou, Dawn Song, and Quoc V Le. 2019b. Neural symbolic reader: Scalable integration of distributed and symbolic representations for reading comprehension. In *International Conference on Learning Representations*.

Eunsol Choi, He He, Mohit Iyyer, Mark Yatskar, Wen-tau Yih, Yejin Choi, Percy Liang, and Luke Zettlemoyer. 2018. QuAC: Question answering in context. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 2174–2184, Brussels, Belgium. Association for Computational Linguistics.

Leyang Cui, Yu Wu, Shujie Liu, Yue Zhang, and Ming Zhou. 2020. MuTual: A dataset for multi-turn dialogue reasoning. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 1406–1416, Online. Association for Computational Linguistics.

Ernest Davis and Gary Marcus. 2015. Commonsense reasoning and commonsense knowledge in artificial intelligence. *Communications of the ACM*, 58(9):92–103.

Ming Ding, Chang Zhou, Qibin Chen, Hongxia Yang, and Jie Tang. 2019. Cognitive graph for multi-hop reading comprehension at scale. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 2694–2703, Florence, Italy. Association for Computational Linguistics.

Yuwei Fang, Siqi Sun, Zhe Gan, Rohit Pillai, Shuohang Wang, and Jingjing Liu. 2020. Hierarchical graph network for multi-hop question answering. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 8823–8838, Online. Association for Computational Linguistics.

Yifan Gao, Chien-Sheng Wu, Jingjing Li, Shafiq Joty, Steven C.H. Hoi, Caiming Xiong, Irwin King, and Michael Lyu. 2020. Discern: Discourse-aware entailment reasoning network for conversational machine reading. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 2439–2449, Online. Association for Computational Linguistics.

Artur d'Avila Garcez, Tarek R Besold, Luc De Raedt, Peter Földiak, Pascal Hitzler, Thomas Icard, Kai-Uwe Kühnberger, Luis C Lamb, Risto Miikkulainen, and Daniel L Silver. 2015. Neural-symbolic learning and reasoning: contributions and challenges. In *2015 AAAI Spring Symposium Series*.

Karl Moritz Hermann, Tomás Kociský, Edward Grefenstette, Lasse Espeholt, Will Kay, Mustafa Suleyman, and Phil Blunsom. 2015. Teaching machines to read and comprehend. In *Advances in Neural Information Processing Systems 28: Annual Conference on Neural Information Processing Systems 2015, December 7-12, 2015, Montreal, Quebec, Canada*, pages 1693–1701.

Lifu Huang, Ronan Le Bras, Chandra Bhagavatula, and Yejin Choi. 2019. Cosmos QA: Machine reading comprehension with contextual commonsense reasoning. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 2391–2401, Hong Kong, China. Association for Computational Linguistics.

Yinya Huang, Meng Fang, Yu Cao, Liwei Wang, and Xiaodan Liang. 2021. DAGN: Discourse-aware graph network for logical reasoning. In *NAACL*.

Diederik P. Kingma and Jimmy Ba. 2015. Adam: A method for stochastic optimization. In *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*.

Guokun Lai, Qizhe Xie, Hanxiao Liu, Yiming Yang, and Eduard Hovy. 2017. RACE: Large-scale ReAding comprehension dataset from examinations. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 785–794, Copenhagen, Denmark. Association for Computational Linguistics.

Jing Li, Aixin Sun, and Shafiq R. Joty. 2018. Segbot: A generic neural text segmentation model with pointer network. In *Proceedings of the Twenty-Seventh International Joint Conference on Artificial Intelligence, IJCAI 2018, July 13-19, 2018, Stockholm, Sweden*, pages 4166–4172. ijcai.org.

Jian Liu, Leyang Cui, Hanmeng Liu, Dandan Huang, Yile Wang, and Yue Zhang. 2020a. Logiqa: A challenge dataset for machine reading comprehension with logical reasoning. In *Proceedings of the Twenty-Ninth International Joint Conference on Artificial Intelligence, IJCAI 2020*, pages 3622–3628. ijcai.org.

Xiaodong Liu, Hao Cheng, Pengcheng He, Weizhu Chen, Yu Wang, Hoifung Poon, and Jianfeng Gao. 2020b. Adversarial training for large neural language models. *arXiv preprint arXiv:2004.08994*.

Yixin Nie, Adina Williams, Emily Dinan, Mohit Bansal, Jason Weston, and Douwe Kiela. 2020. Adversarial NLI: A new benchmark for natural language understanding. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*. Association for Computational Linguistics.

Siru Ouyang, Zhuosheng Zhang, and Hai Zhao. 2021a. Dialogue graph modeling for conversational machine reading. In *The Joint Conference of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (ACL-IJCNLP 2021)*.

Siru Ouyang, Zhuosheng Zhang, and Hai Zhao. 2021b. Fact-driven logical reasoning. *arXiv preprint arXiv:2105.10334*.

Lin Qiu, Yunxuan Xiao, Yanru Qu, Hao Zhou, Lei Li, Weinan Zhang, and Yong Yu. 2019. Dynamically fused graph network for multi-hop reasoning. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 6140–6150, Florence, Italy. Association for Computational Linguistics.

Pranav Rajpurkar, Robin Jia, and Percy Liang. 2018. Know what you don't know: Unanswerable questions for SQuAD. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 784–789, Melbourne, Australia. Association for Computational Linguistics.

Pranav Rajpurkar, Jian Zhang, Konstantin Lopyrev, and Percy Liang. 2016. SQuAD: 100,000+ questions for machine comprehension of text. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 2383–2392, Austin, Texas. Association for Computational Linguistics.

Siva Reddy, Danqi Chen, and Christopher D. Manning. 2019. CoQA: A conversational question answering challenge. *Transactions of the Association for Computational Linguistics*, 7:249–266.

Hongyu Ren and Jure Leskovec. 2020. Beta embeddings for multi-hop logical reasoning in knowledge graphs. *Advances in Neural Information Processing Systems*, 33.

Saku Sugawara, Pontus Stenetorp, and Akiko Aizawa. 2021. Benchmarking machine reading comprehension: A psychological perspective. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*, pages 1592–1612.

Alon Talmor, Jonathan Herzig, Nicholas Lourie, and Jonathan Berant. 2019. CommonsenseQA: A question answering challenge targeting commonsense knowledge. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4149–4158, Minneapolis, Minnesota. Association for Computational Linguistics.

Boxin Wang, Shuohang Wang, Yu Cheng, Zhe Gan, Ruoxi Jia, Bo Li, and Jingjing Liu. 2020. Infobert: Improving robustness of language models from an information theoretic perspective. *arXiv preprint arXiv:2010.02329*.

Siyuan Wang, Wanjun Zhong, Duyu Tang, Zhongyu Wei, Zhihao Fan, Daxin Jiang, Ming Zhou, and Nan Duan. 2021. Logic-driven context extension and data augmentation for logical reasoning of text. *arXiv preprint arXiv:2105.03659*.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Rémi Louf, Morgan Funtowicz, Joe Davison, Sam Shleifer, Patrick von Platen, Clara Ma, Yacine Jernite, Julien Plu, Canwen Xu, Teven Le Scao, Sylvain Gugger, Mariama Drame, Quentin Lhoest, and Alexander M. Rush. 2020. Transformers: State-of-the-art natural language processing. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 38–45, Online. Association for Computational Linguistics.

Zhilin Yang, Peng Qi, Saizheng Zhang, Yoshua Bengio, William Cohen, Ruslan Salakhutdinov, and Christopher D. Manning. 2018. HotpotQA: A dataset for diverse, explainable multi-hop question answering. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 2369–2380, Brussels, Belgium. Association for Computational Linguistics.

Weihao Yu, Zihang Jiang, Yanfei Dong, and Jiashi Feng. 2020. Reclor: A reading comprehension dataset requiring logical reasoning. In *8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, April 26-30, 2020*. OpenReview.net.

Zhuosheng Zhang, Yuwei Wu, Junru Zhou, Sufeng Duan, Hai Zhao, and Rui Wang. 2020a. Sg-net: Syntax-guided machine reading comprehension. In *The Thirty-Fourth AAAI Conference on Artificial Intelligence, AAAI 2020, The Thirty-Second Innovative Applications of Artificial Intelligence Conference, IAAI 2020, The Tenth AAAI Symposium on Educational Advances in Artificial Intelligence, EAAI 2020, New York, NY, USA, February 7-12, 2020*, pages 9636–9643. AAAI Press.

Zhuosheng Zhang, Junjie Yang, and Hai Zhao. 2021. Retrospective reader for machine reading comprehension. *Proceedings of the AAAI Conference on Artificial Intelligence*, 35(16):14506–14514.

Zhuosheng Zhang, Hai Zhao, and Rui Wang. 2020b. Machine reading comprehension: The role of contextualized language models and beyond. *arXiv preprint arXiv:2005.06249*.

Wanjun Zhong, Siyuan Wang, Duyu Tang, Zenan Xu, Daya Guo, Jiahai Wang, Jian Yin, Ming Zhou, and Nan Duan. 2021. AR-LSAT: Investigating Analytical Reasoning of Text. *arXiv e-prints*, page arXiv:2104.06598.

A KPHs Extraction Algorithm

Algorithm 1 Key Phrases (KPH) Extraction Algorithm

Require: Input $C = \{S^1, S^2, \ldots, S^I\}$, n-gram length n, min word length m, number of Key Phrases k
Ensure: Set of Key phrases with top-k TF-IDF weights $K = \{g_1, g_2, \ldots, g_k\}$

Obtain the TF-IDF dictionary $\mathcal{F}$ = TF-IDF(C)
Generate n-gram dictionary G = n-GRAM(C, n)
Filter n-gram dictionary G, $\tilde{G}$=FILTER(G, m)
Retrieve the original expressions K = RETRIEVE($C, \mathcal{F}, \tilde{G}, k$)
procedure TF-IDF(C)
    for each sentence in C **do**
        Filter stop-words in the sentence
        Calculate the TF-IDF weight for each word
    end for
    return TF-IDF dictionary $\mathcal{F}$
end procedure
procedure n-GRAM(C, n)
    for each sentence in C **do**
        Select all gram g with length n in the sentence
        Add g to the dictionary G
    end for
    return n-gram dictionary G
end procedure
procedure FILTER(G, m)
    for each n-gram g in G **do**
        if stopwords in g or length(g) is less than m or there is any number in g **then**
            Delete g
        end if
        if length(g) is 1 and POStag(g) is not noun, verb, or adjective **then**
            Delete g
        end if
    end for
    return n-gram dictionary $\tilde{G}$
end procedure
procedure RETRIEVE($C, \mathcal{F}, \tilde{G}, k$)
    for each g in $\tilde{G}$ **do**
        Calculate the sum of the TF-IDF weights of each word in g, add to a dictionary $z = \{g : w(g)\}$
    end for
    Rank the top-k n-gram g by TF-IDF weight sum. Construct key phrases set $K = \{g_1, g_2, \ldots, g_k\}$
    if n=1 **then**
        for each g in K **do**
            g_s= STEM(g)
            Retrieve all the original words from C containing g_s, add to K
        end for
    end if
    return Set of Key phrases $K = \{K_1, K_2, \ldots, K_k\}$
end procedure

B Dataset Information

ReClor The Reading Comprehension dataset requiring logical reasoning (ReClor) is extracted from standardized graduate admission examinations (Yu et al., 2020). It contains 6,138 multiple-choice questions modified from standardized tests such as GMAT and LSAT and is randomly split into train/dev/test sets with 4,638/500/1,000 samples respectively. Multiple types of logical reasoning question are included.

LogiQA LogiQA is sourced from expert-written questions for testing human Logical reasoning. It contains 8,678 QA pairs, covering multiple types of deductive reasoning. It is randomly split into train/dev/test sets with 7,376/651/651 samples respectively.

SNLI The Stanford Natural Language Inference (SNLI) dataset contains 570k human annotated sentence pairs, in which the premises are drawn from the captions of the Flickr30 corpus and hypotheses are manually annotated. The full dataset is randomly split into 549k/9.8k/9.8k. This is the most widely used entailment dataset for natural language inference. It requires models to take a pair of sentence as input and classify their relation types, i.e., ENTAILMENT,NEUTRAL, or CONTRADICTION.

ANLI The Adversarial Natural Language Inference (ANLI) is a new large-scale NLI benchmark dataset, collected via an iterative, adversarial human-and-model-in-the-loop procedure. Specifically, the instances are chosen to be difficult for the state-of-the-art models such as BERT and RoBERTa. A1, A2 and A3 are the datasets collected in three rounds. A1 and A2 are sampled from Wiki and A3 is from News. It requires models to take a set of context, hyperthesis and reason classify the label (ENTAILMENT,NEUTRAL, or CONTRADICTION). A1 has 18,946 in total and is split into 16,946/1,000/1,000. A2 has 47,460 in total and is split into 45,460/1,000/1,000. A3 has 102,859 in total and is split into 100,459/1,200/1,200. ANLI refers to the combination of A1, A2 and A3.

C Parameter Selection

Our model is implemented based on the Transformers Library (Wolf et al., 2020). Adam (Kingma and Ba, 2015) is used as our optimizer. The best threshold for defining semantic relevance is 0.5. We run 10 epochs for ReClor and LogiQA, 5 epochs for SNLI and ANLI, and select the model that achieves the best result in validation. Our models are trained on one 32G NVIDIA Tesla V100 GPU. The training time is around half an hour for each epoch. The maximum sequence length is 256 for ReClor and SNLI, 384 for LogiQA and 128 for ANLI. The weight decay is 0.01. We set the warm-up proportion during training to 0.1. We provide training

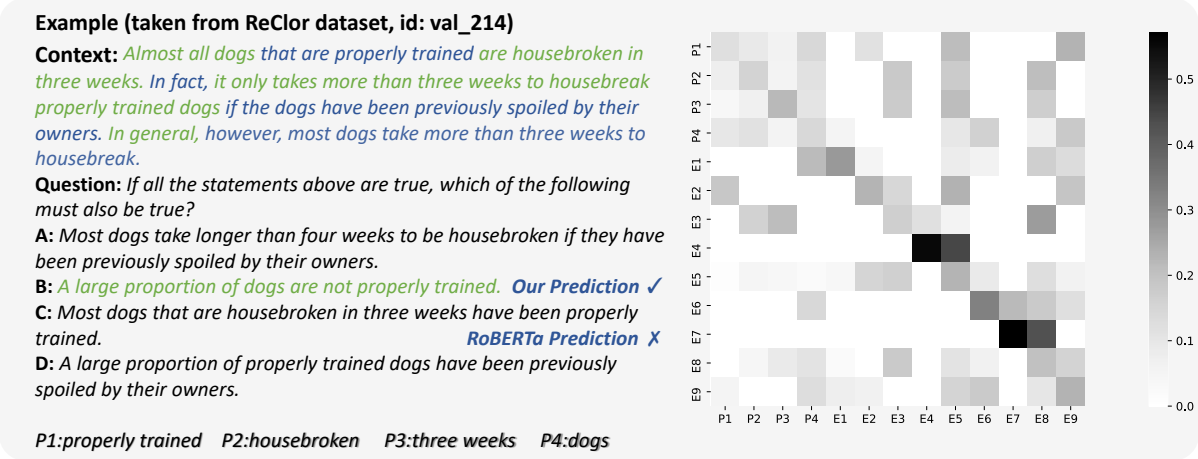

Figure 5: An example showing the logical reasoning capability of our model (Left) and the corresponding attention map (Right). EDUs are shown in different colors alternately, corresponding to E1-E9 in the attention map.

Dataset	PrLM	batchsize	learning rate
ReClor	RoBERTa-base	24	1e-5
	RoBERTa-large	32	8e-6
	DeBERTa-xlarge	8	8e-6
LogiQA	RoBERTa-base	2	4e-6
	RoBERTa-large	2	8e-6
	DeBERTa-xlarge	2	8e-6
SNLI	BERT-base	32	2e-5
	RoBERTa-large	32	2e-5
ANLI	RoBERTa-large	32	2e-05

Table 6: Parameter Selection

configurations used across our experiments in Table 6.

D Case Study

To intuitively show how our model works, we select an example from ReClor as shown in Figure 5, whose answer is predicted correctly by our model but not by baseline models (RoBERTa). The example shows that powerful pre-trained language models such as RoBERTa may be better at dealing with sentence pairs that contain overlap parts or similar words. For example, the wrong answer chosen by RoBERTa is another expression of the first sentence in the given context. The words are basically the same, only the order changes. The model itself does not understand the logical relation between sentences and phrases, but only compares their common elements for prediction, failing in logical reasoning task. In contrast, our model can not only match synonymic expressions, but also make logical inferences by separating sentences into EDUs and extracting key phrases and establishing logical relations between them. The importance of those elements are interpreted by the attention distribution as shown in the right part, which is derived from the last layer of our model.

Hierarchical Representation-based Dynamic Reasoning Network for Biomedical Question Answering

Jianguo Mao[1,2†‡] **Jiyuan Zhang**[3†] **Zengfeng Zeng**[3], **Weihua Peng**[3], **Wenbin Jiang**[3*]
Xiangdong Wang[1], **Hong Liu**[1], **Yajuan Lyu**[3]
[1]Beijing Key Laboratory of Mobile Computing and Pervasive Device,
Institute of Computing Technology, Chinese Academy of Sciences, Beijing, China
[2] University of Chinese Academy of Sciences, Beijing, China
[3] Baidu Inc., Beijing, China
{maojianguo20s,xdwang,hliu}@ict.ac.cn
{zhangjiyuan01,zengzengfeng,jiangwenbin,lvyajuan}@baidu.com

Abstract

Recently, Biomedical Question Answering (BioQA) has attracted growing attention due to its application value and technical challenges. Most existing works treat it as a semantic matching task that predicts answers by computing confidence among questions, options and evidence paragraphs, which is insufficient for scenarios that require complex reasoning based on a deep understanding of biomedical evidences. We propose a novel model termed **H**ierarchical Representation-based **D**ynamic **R**easoning **N**etwork (HDRN) to tackle this problem. It first constructs the hierarchical representations for biomedical evidences to learn semantics within and among evidences. It then performs dynamic reasoning based on the hierarchical representations of evidences to solve complex biomedical problems. Against the existing state-of-the-art model, the proposed model significantly improves more than **4.5%**, **3%** and **1.3%** on three mainstream BioQA datasets, PubMedQA, MedQA-USMLE and NLPEC. The ablation study demonstrates the superiority of each improvement of our model. https://github.com/mikeblueskydl/HDRN

1 Introduction

Machine reading comprehension (MRC) (Rajpurkar et al., 2016, 2018; Yang et al., 2018) tasks are often used to evaluate the intelligence degree of a system, and many recent large-scale pre-trained language models (Lan et al., 2019; Zaheer et al., 2020) have surpassed the human performance on open-domain MRC. In recent years, Biomedical Question Answering (BioQA) (Tsatsaronis et al., 2012; Wang et al., 2018; Tang et al., 2019; Li et al., 2020; Dai et al., 2022) has attracted growing attention due to its great application value and technical challenges. As Figure 1 shows, compared with open-domain MRC tasks, BioQA raises higher demands for understanding professional biomedical evidences and relies more on complex reasoning based on semantics within and among evidences to predict answers, which is also difficult for humans. The pass rate of the human examinee is less than 14.2% in the National Licensed Pharmacist Examination in China (Li et al., 2020). While in open-domain MRC, the human performance can reach 86.8% Exact Match (Rajpurkar et al., 2018).

Question: A 35-year-old Caucasian female presents to the hospital alarmed by her recent truncal weight gain, facial hair growth, and thinning skin. During the physical exam, the physician finds that the patient is **hypertensive**. Serum analysis reveals **hyperglycemia**. The physician suspects a **pituitary adenoma**. Which dexamethasone test result would help confirm the physician's suspicions?

Options:
A: Low-dose, increased ACTH; high-dose, decreased ACTH
B: Low-dose, decrease in ACTH; high-dose, no change in ACTH
C: Low-dose, no change in ACTH; high-dose, no change in ACTH
D: Low-dose, no change in ACTH; high-dose, decreased ACTH ✔

Evidences of Option D:
E1: If cortisol production is driven by an ACTH producing **pituitary adenoma** dexamethasone suppression is **ineffective at low** doses but usually **induces suppression at high doses**. Inappropriately low ACTH levels in the setting of low cortisol levels are characteristic of diminished ACTH reserve.
E2: High dose dexamethasone suppresses ACTH production by a pituitary adenoma serum cortisol is lowered but does not suppress ectopic ACTH production serum cortisol remains high. Cortisol stimulates gluconeogenesis and insulin resistance resulting in **hyperglycemia** as well as muscle cell protein breakdown and lipolysis to provide sub strates for hepatic gluconeogenesis.
E3: The mechanism of **hypertension** may be related to stimulation of mineralocorticoid receptors by cortisol and increased secretion of other adrenal steroids. High ACTH decreased negative feedback leads to bilateral adrenal hyperplasia.

Figure 1: **An example from MedQA-USMLE dataset**. (✓: correct answer option).

Due to the extremely high cost of collection and annotation of the biomedical data, BioASQ (Tsatsaronis et al., 2012) was for a long time the only authoritative benchmark for the development of BioQA systems. Recently, more high-quality

*Corresponding author.
†Equal contribution.
‡This work was done during internship at Baidu Inc.

Proceedings of the 29th International Conference on Computational Linguistics, pages 1480–1489
October 12–17, 2022.

datasets have further contributed to the development of the field, such as NLPEC (Li et al., 2020), MedQA-USMLE (Zhang et al., 2018), and PubMedQA (Jin et al., 2019). Some researchers have explored pre-trained language models to solve this task (Huang et al., 2019; Beltagy et al., 2019a; Lee et al., 2020; Dai et al., 2022). Meanwhile, some researchers have introduced external biomedical knowledge to aid the model in answering questions. (Zhang et al., 2018; Yue et al., 2020). Despite the success, previous works mainly explore a better language model or external domain knowledge, which is insufficient to deal with BioQA in complex scenarios which require complex reasoning based on the semantics within and among evidences to answer the question. Intuitively, it is essential to explore a better representation learning method for biomedical evidences and a better reasoning mechanism for complex biomedical questions.

To tackle this problem, we propose a novel model, termed Hierarchical Representation-based Dynamic Reasoning Network (HDRN), to achieve this goal in two main parts. First, constructing hierarchical representations to learn semantics within and among the biomedical evidences needed to reason the answer. To this end, we first use a shared pre-trained language model to obtain the intra-level representations of the question and evidences separately. Then, we construct coarse to fine-grained inter-level representations of evidences to learn semantics among them. Second, conducting multi-step dynamic reasoning based on the hierarchical representations to predict the answer. At each step, it adaptively aggregates critical information from hierarchical representations according to current state and conducts single-step reasoning to update the state. All intermediate reasoning results are dynamically integrated to predict the answer.

To sum up, the contributions of our work are as follows:

- We propose HDRN, a novel neural network used for semantic representation learning and reasoning for BioQA.
- We design a hierarchical representation learning method to learn semantics within and among the biomedical evidences.
- We design a novel reasoning mechanism that iteratively performs multi-step dynamic reasoning to solve complex biomedical questions.
- We achieve state-of-the-art performances on three BioQA datasets, and the experiment results demonstrate the superiority of each component of the proposed model.

2 Related Work

Biomedical Question Answering BioQA is an emerging and challenging task. Given a question, it requires intelligent systems to understand the complex biomedical domain expertise and reason the answers. Meanwhile, collecting and annotating data requires experts with a medical background to complete, which is difficult and costly. BioASQ (Tsatsaronis et al., 2012) is a benchmark for biomedical semantic indexing and question answering for a long time. Recently, many works have made efforts to construct more high-quality and challenging BioQA datasets. The datasets can be mainly divided into two categories, the first category is constructed based on biomedical domain publications or electronic medical records, including emrQA (Pampari et al., 2018), MedQA-USMLE (Zhang et al., 2018), and PubMedQA (Jin et al., 2019). The second category is constructed based on biomedical examinations from different countries, including Head-QA (Vilares and Gómez-Rodríguez, 2019) and NLPEC (Li et al., 2020). To tackle this task, most of previous works have optimized the language model by pre-training on biomedical domain-related corpus, and obtained great success (Peng et al., 2019; Alsentzer et al., 2019; Jin et al., 2019; Beltagy et al., 2019b; Lee et al., 2019; raj Kanakarajan et al., 2021; Yasunaga et al., 2022). In addition, Yue et al. (2020) explored the external clinical domain knowledge to enhance the generalization of the model. Yasunaga et al. (2021) explored joint reasoning over text and knowledge graph for BioQA. Dai et al. (2022) solved the parameter competition problem via a Mixture-of-Expert.

Unlike previous works, our proposed HDRN model has two distinctive characteristics: (1) It explores a hierarchical representation learning method that can better learn semantics within and among evidences. The effectiveness of the similar idea of hierarchical representation learning method has also been validated in the vision (Lan et al., 2014), recommendation (Jiang et al., 2018) domains, our proposed method is better adapted to the characteristics of professional biomedical evidences. (2) It explores a novel dynamic reason-

ing mechanism that adaptively aggregates critical information from hierarchical representations for multi-step reasoning and dynamically integrates intermediate reasoning results to predict answers. Although the concept of multi-step reasoning mechanism has been mentioned in other natural language processing tasks (Haug et al., 2018; Liu et al., 2020; Zhao et al., 2021) or domains (Song et al., 2018; Gan et al., 2019; Le et al., 2021), our proposed mechanism has more flexible and powerful information convergence and reasoning capabilities for biomedical question answering.

3 Background

BioQA is a classification task that uses accuracy as the evaluation metric. Specifically, given a natural language question Q and evidences C, it requires the intelligent system to predict the correct option $\hat{o}$ from the candidate set Ω_o based on the understanding of the evidences C. θ is set of the model parameters.

$$\hat{o} = \underset{o \in \Omega_o}{\operatorname{argmax}} P(o|Q, C; \theta) \tag{1}$$

As Figure 2 (a) shows, most previous works treat BioQA as a semantic matching task. They first concatenate all information together, including questions and evidences, then encode them using a language model, and finally predict the best option by computing the semantic matching score with a multi-layer perceptron. The previous works accomplish both representation learning and reasoning using a single model. There are two main drawbacks: (i). Lack of deep understanding of biomedical evidences. The hierarchical information among evidences is easily lost when all the information is mixed together for encoding. (ii). Lack of strong reasoning capability. Single-step implicit reasoning does not cope well with complex questions and evidences. In terms of human experience, the reasoning is usually an iterative process that requires multi-step to solve complex questions.

As Figure 2 (b) shows, to address the problem, we propose HDRN, a network for representing learning and reasoning for BioQA. It first constructs hierarchical representations to obtain a deep understanding of the biomedical evidences, and then performs multi-step dynamic reasoning to solve complex questions.

4 Method

In this section, we describe the detail of the proposed method. The overall architecture is shown in Figure 3, which consists of two components: (a) Dynamic Reasoning Mechanism, which can better solve complex biomedical questions by conducting multi-step dynamic reasoning. (b) Hierarchical Representation Learning, which can better understand the biomedical evidences by learning the semantics within and among the them.

4.1 Hierarchical Representation Learning

Each BioQA instance usually contains multiple evidences related to the question, and the semantic information within and among evidences are essential for reasoning. We propose a hierarchical representation learning method to obtain a deep understanding of the evidences. Specifically, it consists of three levels of representations, **(1) Intra-level Representations**: learning the semantic information of each evidence. **(2) Coarse-grained Inter-level Representations**: learning the correlations among evidences based on token representations of all evidences. **(3) Fine-grained Inter-level Representations**: learning more abstract correlations among evidences based on sentence representations of all evidences.

Intra-level Representations Given a question Q, and the set of M evidence sentences $C = \{c_m\}_{m=1}^{M}$, we use a pre-trained language model(PLM) as the language encoder to extract intra-level representations. E.g., given a sentence S with T tokens, where $S \in \mathbb{R}^{T \times 1}$, we first add a special token $[cls]$ at the beginning as input. After encoding by the language encoder, we obtain the representations $R^S \in \mathbb{R}^{(T+1) \times d^l}$, where d^l is the output feature dimension of language encoder. Specifically, for the set of evidence sentences C, we first concatenate each evidence c_m with question Q, and take the representations of all tokens as their intra-level representations $R^C = \{R^{C_m}\}_{m=1}^{M}$, where $R^{C_m} \in \mathbb{R}^{(T+1) \times d^l}$.

$$R^{C_m} = \text{PLM}([Q; c_m]) \tag{2}$$

where ; is the concatenate operation.

Question Q has only one sentence that does not contain hierarchical information, so we choose the representations of $[cls]$ token as the representations of the question $R^Q \in \mathbb{R}^{1 \times d^l}$, which is sufficient to represent the semantics of the whole sentence

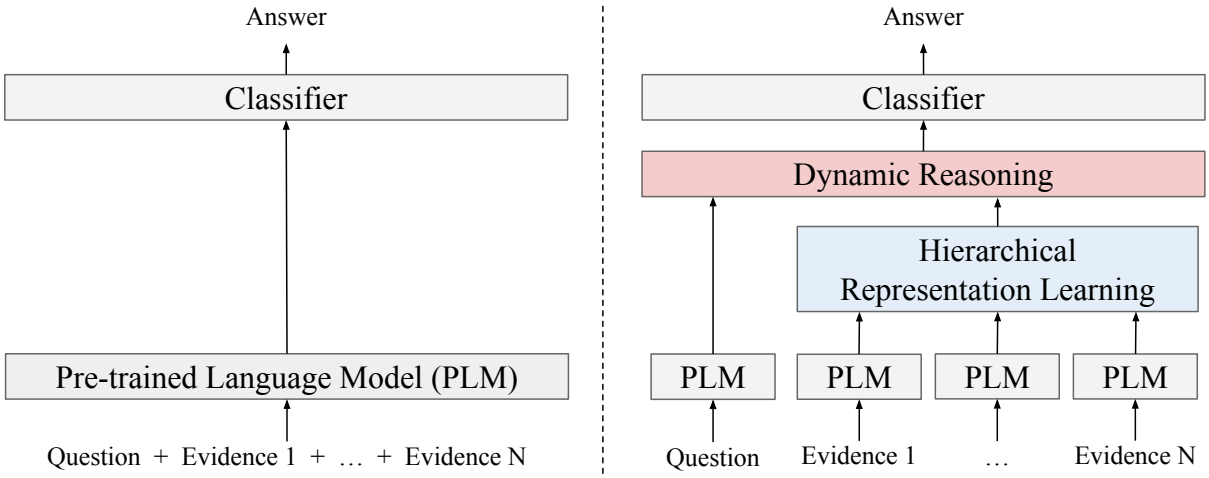

(a). Paradigm of previous works

(b). Ours

Figure 2: **Comparison of the model architecture between our method and previous works**. Unlike previous works that adopt a unified model to conduct representation learning and single-step implicit reasoning, our method constructs hierarchical representations to understand biomedical evidences and conduct multi-step dynamic reasoning to solve complex questions.

and makes the subsequent reasoning process more elegant.

$$R^Q = \text{PLM}_{\text{CLS}}([Q]) \tag{3}$$

Coarse-grained Inter-level Representations We construct the coarse-grained inter-level representations to learn the correlations among evidences paragraphs that are important for reasoning. Specifically, we concatenate the intra-level representations of M evidence paragraphs $R^C = \{R^{C_m}\}_{m=1}^{M}$ into a sequence

$$R^C_{concat} = [R^{C_1}; R^{C_2}; ...; R^{C_M}] \tag{4}$$

where $R^C_{concat} \in \mathbb{R}^{M(T+1)\times d^l}$.

Then, we use Scaled Dot-Product Attention (Vaswani et al., 2017) to update the representations of each token according to the representations of other tokens in all evidence paragraphs to learn the semantic relationships among all evidences.

$$R'^C_{concat} = \text{Attention}(R^C_{concat}, R^C_{concat}, R^C_{concat}) \tag{5}$$

$$\text{Attention}(Q, K, V) = \text{softmax}(\frac{QK^\top}{d^l})V \tag{6}$$

Then, we apply a linear projection layer and a residual connection on the updated representations R'^C_{concat}

$$R'^C_{concat} = R^C_{concat} + \text{Linear}(R'^C_{concat}) \tag{7}$$

Finally, we obtain the coarse-grained inter-level representations $R^C_{cInter} = \{R^{C_m}_{cInter}\}_{m=1}^{M}$ by slicing the R'^C_{concat} according to the length of the evidence paragraphs.

Fine-grained Inter-level Representations In order to learn more abstract correlations among evidences, we construct the fine-grained inter-level representations. Specifically, we concatenate the coarse-grained inter-level representations of the $[cls]$ token for each evidence paragraphs into a vector sequence

$$R^{C^{cls}}_{cInter} = [R^{C^{cls}_1}_{cInter}; ...; R^{C^{cls}_M}_{cInter}] \tag{8}$$

where $R^{C^{cls}_m}_{cInter} \in \mathbb{R}^{1\times d^l}$.

Then we use the same attention, linear projection layer and residual connection as learning the coarse-grained inter-level representations to obtain the fine-grained inter-level representations $R^C_{fInter} \in \mathbb{R}^{M\times d^l}$

$$R'^{C^{cls}}_{cInter} = \text{Attention}(R^{C^{cls}}_{cInter}, R^{C^{cls}}_{cInter}, R^{C^{cls}}_{cInter}) \tag{9}$$

$$R^C_{fInter} = R^{C^{cls}}_{cInter} + \text{Linear}(R'^{C^{cls}}_{cInter}) \tag{10}$$

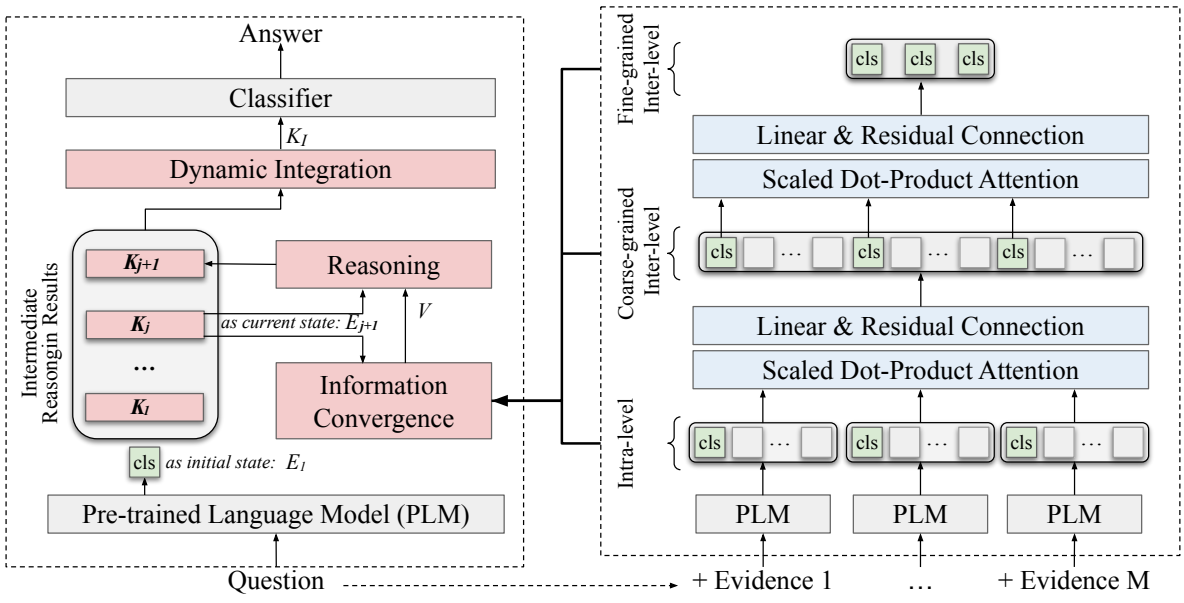

(a) Dynamic Reasoning Mechanism

(b) Hierarchical Representation Learning

Figure 3: **Overall architecture of the proposed HDRN**. It contains two components: (a) Dynamic Reasoning Mechanism (described in section 4.2) and (b) Hierarchical Representation Learning (described in section 4.1).

4.2 Dynamic Reasoning Mechanism

In general, reasoning is an iterative process that requires constantly updating the current state according to the state-related information and gradually reasoning out the answer. Inspired by the nature of human reasoning mechanism, we design the Dynamic Reasoning Mechanism to imitate the process. It iteratively performs multi-step dynamic reasoning to predict the answer. At each step of reasoning, it adaptively aggregates hierarchical representations (**Information Convergence**) according to the current state and performs single-step reasoning to obtain the intermediate result and update the state. Each intermediate reasoning result focuses on different parts of the hierarchical information. Thus, we integrate them dynamically to predict the answer better. (**Dynamic Integration**). We set the initial state E_1 to the question representations R^Q at the first reasoning step. When performing the jth step of reasoning, the state is E_j:

Information Convergence We get state-related information $\{H^C, H^C_{cInter}, H^C_{fInter}\}$ from hierarchical representations $\{R^C, R^C_{cInter}, R^C_{fInter}\}$ through Scaled Dot-Product Attention according to the current state E_j.

$$H^C = \text{softmax}(\frac{E_j R^{C^\top}}{d^l})R^C \quad (11)$$

$$H^C_{cInter} = \text{softmax}(\frac{E_j R^{C^\top}_{cInter}}{d^l})R^C_{cInter} \quad (12)$$

$$H^C_{fInter} = \text{softmax}(\frac{E_j R^{C^\top}_{fInter}}{d^l})R^C_{fInter} \quad (13)$$

Then, we apply a linear layer with softmax as the activation function on the state-related information to calculate the distribution of weight D and get the weighted sum as the reasoning-related information V.

$$D = \text{softmax}(\{H^C, H^C_{cInter}, H^C_{fInter}\}) \quad (14)$$

$$V = D \cdot [H^C; H^C_{cInter}; H^C_{fInter}] \quad (15)$$

Reasoning We perform single-step reasoning according to the current state E_j and the reasoning-related information V to obtain the intermediate reasoning results K_j.

$$K_j = E_j + \text{ReLU}(W_1 V + b_1)W_2 + b_2 \quad (16)$$

where W_1 and W_2 are weight matrices and b_1 and b_2 are biases.

Methods	Accuracy (%) Test
BlueBERT (Peng et al., 2019)	48.4
ClinicalBERT (Alsentzer et al., 2019)	49.0
PubMedBERT (Jin et al., 2019)	55.8
SciBERT (Beltagy et al., 2019b)	57.3
BioBERT (Lee et al., 2019)	60.2
BioELECTRA (raj Kanakarajan et al., 2021)	64.0
UNIFIEDQA-v2 (Khashabi et al., 2022)	64.2
BioLink-BERT (Yasunaga et al., 2022)	72.1
HDRN (Ours)	**76.6**

Table 1: **Performance comparison on the PubMedQA.**

Methods	Accuracy (%) Test
BERT (Devlin et al., 2018)	34.3
BioRoBERTa (Gururangan et al., 2020)	36.1
BioBERT (Lee et al., 2019)	36.7
PubMedBERT (Jin et al., 2019)	38.1
QAGNN (Yasunaga et al., 2021)	38.0
GreaseLM (Zhang et al., 2022)	38.5
MoE-BQA (Dai et al., 2022)	41.6
BioLink-BERT (Yasunaga et al., 2022)	44.6
HDRN (Ours)	**47.6**

Table 2: **Performance comparison on the MedQA-USMLE.**

Methods	Accuracy (%) Test
BiDAF (Seo et al., 2016)	43.6
Co-Matching (Wang et al., 2018)	45.8
SeaReader (Zhang et al., 2018)	48.4
Multi-Matching (Tang et al., 2019)	48.7
BERT-base (Devlin et al., 2018)	52.2
ERNIE (Sun et al., 2019)	53.4
RoBERTa-wwm-ext-large (Cui et al., 2021)	57.9
KMQA (Li et al., 2020)	61.8
MoE-BQA (Dai et al., 2022)	62.2
HDRN (Ours)	**63.5**

Table 3: **Performance comparison on the NLPEC.**

At the next step, we set the state E_{j+1} to the K_j. The above Information Convergence and Reasoning process is performed again based on the state E_{j+1}. We obtain J intermediate reasoning results $K = \{K_1, ..., K_J\}$ after repeating the above process J times.

Dynamic Integration After J steps of reasoning, we obtain all intermediate reasoning results $K = \{K_1, ..., K_J\}$. We integrate them by applying a nonlinear transformation with the softmax function.

$$K_I = \text{softmax}(W_I K + b_I)K \tag{17}$$

4.3 Classifier

Given the integrated reasoning results K_I, we use a linear layer as a classifier to obtain the logits l_k for the answer options. Then we calculate the probability distribution of each answer option by applying a softmax function. We use cross-entropy loss as our model loss $\mathcal{L}$ to update the model parameters.

$$l_k = \text{classifier}(K_I) \tag{18}$$

$$\hat{y} = \text{softmax}(l_k), \mathcal{L} = \text{CrossEntropy}(\hat{y}) \tag{19}$$

5 Experiments

5.1 Datasets

As mentioned in Section 2, there are mainly two categories of datasets. In our work, we select three widely used and challenging datasets in two categories to evaluate the model performance. Among them, the MedQA-USMLE and PubMedQA are in english, and the NLPEC is in chinese. We use accuracy to measure the model performance.

PubMedQA PubMedQA is a large scale English BioQA dataset collected from PubMed abstracts. It contains a total of 273.5k QA examples, of which 1k expert-annotated, 211.3k artificially generated, and 61.2k unlabeled. The number of examples for the train/dev/test set is 272,950/50/500. Each instance consists of a question, a context which is the abstract from PubMed without its conclusion, and a long answer which is the conclusion of the context. It requires answering the question with yes/no/maybe based on the reasoning over context.

MedQA-USMLE MedQA-USMLE is a large scale multilingual BioQA dataset collected from the National Medical Board Examinations in the USA, Mainland China, and Taiwan. Most previous work only used English subset for training and evaluation, so we also use English subset for fair comparison. The English subset contains 12k QA examples in total. The number of examples for the train/dev/test set is 10,178/1,272/1,273. Each example consists of a question, four candidate options with the correct one annotated. It requires predicting the correct option corresponding to the given question.

NLPEC NLPEC is a large scale Chinese BioQA dataset containing 21.7k multiple-choice questions with human-annotated answers collected from the National Licensed Pharmacist Examination in China. The number of examples for the train/dev/test set is 18,703/2,500/547. Each question has five candidate options and evidences retrieved from the official exam guidebook that contains the information needed to answer the question. It requires predicting the correct option corresponding to the given question.

5.2 Implementation Details

For all three datasets, we use the official dataset splits to train and test our model. We set the number of evidences to 3. The feature dimension of language encoder is set to 1024. We conduct our experiments on NVIDIA A100 GPUs with 40GB memroy.

PubMedQA We use BioLink-BERT's (Yasunaga et al., 2022) parameters as initialization parameters for the language model. We use 450 annotated examples and 10k randomly selected artificially generated QA examples for model training. Our model does not use long answers, which is more challenging. The number of reasoning steps is set to 3. We set the batch size to 32, and use AdamW with β_1=0.9 and β_2=0.999 as the optimizer. We set the learning rate to 3e-5. The maximum number of epochs is set to 23.

MedQA-USMLE We use BioLink-BERT's (Yasunaga et al., 2022) parameters as initialization parameters for the language model. We use BM25 to retrieve six sentences with highest scores for each option from official guided books provided by the datasets as evidences. The number of reasoning steps is set to 2. We set the batch size to 32, and use AdamW with β_1=0.9 and β_2=0.98 as the optimizer. We set the learning rate to 3e-5. The maximum number of epochs is set to 6.

NLPEC We use RoBERTa-wwm-ext-large's (Cui et al., 2021) parameters as initialization parameters for the language model. The number of reasoning steps is set to 3. We set the batch size to 16, and use AdamW with β_1=0.9 and β_2=0.999 as the optimizer. We set the learning rate to 3e-5. The maximum number of epochs is set to 35.

5.3 Comparison with State-of-the-Arts

As shown in Table 1, 2 and 3, the proposed method achieves the new state-of-the-art and reaches 76.6% / 47.6%/ 63.5% accuracy on PubMedQA / MedQA-USMLE/ NLPEC datasets with 4.5% / 3.0%/ 1.3% improvement over the previous state-of-the-art method.

PubMedQA Table 1 shows the results on PubMedQA dataset. The first to eighth lines show the accuracy of the previous state-of-the-art methods on the test set. These methods follow a semantic matching paradigm to solve this task and achieve competitive results. They first use language models to encode questions, options, and evidences, and then perform single-step implicit reasoning to predict the answer. Most of the previous works optimize the performance of the pre-trained language models on BioQA by pre-training with biomedical domain-related corpus. These pre-trained language models can all benefit from the better semantic representation and reasoning capabilities of our method. Specifically, our method gains 4.5% improvement compared with the baseline BioLink-BERT (Yasunaga et al., 2022) on test set.

MedQA-USMLE Table 2 shows the results on MedQA-USMLE dataset. The first to eighth lines shows the accuracy of the previous state-of-the-art methods on the test set. These models also follow the semantic matching paradigm to solve this task. Furthermore, by introducing external knowledge and conducting joint reasoning over text and graph (Yasunaga et al., 2021), the performance is further improved. Our method gains 3.0% improvement on test set compared with the baseline BioLink-BERT.

NLPEC Table 3 shows the results on NLPEC dataset. The first to ninth lines show the accuracy of the previous state-of-the-art methods on the test set. These models also follow the semantic matching paradigm to solve this task. Furthermore, by retrieving external biomedical knowledge (Li et al., 2020), the semantic matching capability of the model can be enhanced. By introducing a mixture of experts (Dai et al., 2022) to alleviate the parameter competition problem, where each expert handles a specific type of question, providing better single-step reasoning capabilities. Our method benefits from the proposed better semantic representation learning method and more powerful dynamic reasoning mechanisms. Specifically, our method gains 5.6% improvement compared with the base-

Models	Test Accuracy (%) PubMedQA	MedQA	NLPEC
HDRN (Ours)	**76.6**	**47.6**	**63.5**
w/o Hierarchical Representation Learning	75.2 (1.4 ↓)	46.9 (0.7 ↓)	62.4 (1.1 ↓)
w/o Dynamic Reasoning Mechanism	75.0 (1.6 ↓)	46.8 (0.8 ↓)	62.3 (1.2 ↓)
Hierarchical Representation Learning			
w/o Fine-grained Inter-level	75.6 (1.0 ↓)	46.7 (0.9 ↓)	63.3 (0.2 ↓)
w/o Coarse-grained Inter-level	75.6 (1.0 ↓)	46.9 (0.7 ↓)	62.6 (0.9 ↓)
w/o Intra-level	75.8 (0.8 ↓)	46.4 (1.2 ↓)	62.8 (0.7 ↓)
Dynamic Reasoning Mechanism			
w/o Information Convergence	75.7 (0.9 ↓)	46.9 (0.7 ↓)	62.9 (0.6 ↓)
w/o Dynamic Integration	75.6 (1.0 ↓)	46.8 (0.8 ↓)	62.5 (1.0 ↓)

Table 4: **Ablation study on three BioQA datasets.**

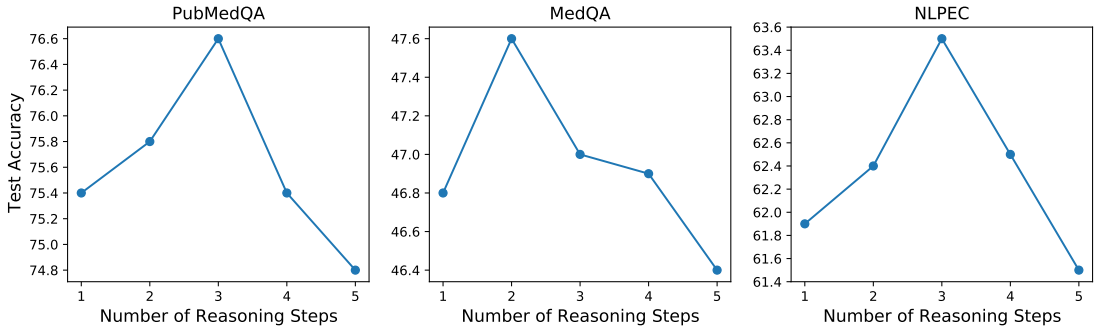

Figure 4: **Effect of the Number of Reasoning Steps.** The optimal number of reasoning steps for the PubMedQA, MedQA-USMLE, and NLPEC are 3, 2, and 3 respectively.

line RoBERTa-wwm-ext-large (Cui et al., 2021) on test set and gains 1.3% improvement compared with the latest work Moe-BQA (Dai et al., 2022).

5.4 Ablation Study

Table 4 shows the results of the ablation study on three BioQA datasets, which demonstrate the superiority of each component. From the experimental results, if there is no Dynamic Reasoning Mechanism, the model performs single-step reasoning based on the output of the language model as in most existing works, and the model performance decreases. If there is no Hierarchical Representation Learning, all information is concatenated together for representation learning, which is the same as the previous works shown in Figure 2 (a), and the model performance further decreases. In addition, we conduct further ablation experiments to analyze the effectiveness and superiority of two key improvements. For Dynamic Reasoning, if there is no Information Convergence, the model cannot dynamically select hierarchical representations for reasoning according to the current state, and the model performance decreases, if there is no Dynamic Integration, the model only uses the result of the last step of reasoning to predict the answer, losing the key information in the reasoning process, and the performance decreases. For Hierarchical Representations, we remove different levels of representations separately to explore the effectiveness of each level representation, and the experimental results show that removing different levels of representation degrades the performance to different degrees.

Effect of the Number of Reasoning Steps Figure 4 shows the effect of the number of reasoning steps. When the number of steps is 1, the process is the same as the classical single-step implicit reasoning paradigm. Empirically, the number of reasoning steps is related to the problem complexity, and the performance gradually increases as we gradually increase the number of reasoning steps. However, when the number of reasoning steps is too large, the performance degrades due to the mismatch between the reasoning process and the problem complexity. The optimal number of reasoning steps varies slightly for different data distributions. The optimal number of reasoning steps for the PubMedQA, MedQA-USMLE, and NLPEC are 3, 2, and 3 respectively.

6 Conclusion

This paper proposes HDRN, a novel model for representation learning and reasoning for biomedical question answering. First, we construct hierarchical representations to obtain a deep understanding of the biomedical evidences. Then, we perform multi-step dynamic reasoning to solve complex biomedical questions. We evaluate our model on three BioQA datasets and achieve new state-of-the-art performances.

Acknowledgements

Our work is supported by the National Key Research and Development Program of China No.2020AAA0109400 and National Natural Science Foundation of China (62276250). This work is also supported by Baidu and ICTCAS Joint Project. We would like to thank the anonymous reviewers for their valuable feedback.

References

Emily Alsentzer, John R Murphy, Willie Boag, Wei-Hung Weng, Di Jin, Tristan Naumann, and Matthew McDermott. 2019. Publicly available clinical bert embeddings. *arXiv preprint arXiv:1904.03323*.

Iz Beltagy, Kyle Lo, and Arman Cohan. 2019a. Scibert: A pretrained language model for scientific text. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 3615–3620.

Iz Beltagy, Kyle Lo, and Arman Cohan. 2019b. Scibert: A pretrained language model for scientific text. *arXiv preprint arXiv:1903.10676*.

Yiming Cui, Wanxiang Che, Ting Liu, Bing Qin, and Ziqing Yang. 2021. Pre-training with whole word masking for chinese bert. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 29:3504–3514.

Damai Dai, Wenbin Jiang, Jiyuan Zhang, Weihua Peng, Yajuan Lyu, Zhifang Sui, Baobao Chang, and Yong Zhu. 2022. Mixture of experts for biomedical question answering. *arXiv preprint arXiv:2204.07469*.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*.

Zhe Gan, Yu Cheng, Ahmed El Kholy, Linjie Li, Jingjing Liu, and Jianfeng Gao. 2019. Multi-step reasoning via recurrent dual attention for visual dialog. *arXiv preprint arXiv:1902.00579*.

Suchin Gururangan, Ana Marasović, Swabha Swayamdipta, Kyle Lo, Iz Beltagy, Doug Downey, and Noah A Smith. 2020. Don't stop pretraining: adapt language models to domains and tasks. *arXiv preprint arXiv:2004.10964*.

Till Haug, Octavian-Eugen Ganea, and Paulina Grnarova. 2018. Neural multi-step reasoning for question answering on semi-structured tables. In *European conference on information retrieval*, pages 611–617. Springer.

Kexin Huang, Jaan Altosaar, and Rajesh Ranganath. 2019. Clinicalbert: Modeling clinical notes and predicting hospital readmission. *arXiv preprint arXiv:1904.05342*.

Zhuoren Jiang, Yue Yin, Liangcai Gao, Yao Lu, and Xiaozhong Liu. 2018. Cross-language citation recommendation via hierarchical representation learning on heterogeneous graph. In *The 41st International ACM SIGIR Conference on Research & Development in Information Retrieval*, pages 635–644.

Qiao Jin, Bhuwan Dhingra, Zhengping Liu, William Cohen, and Xinghua Lu. 2019. Pubmedqa: A dataset for biomedical research question answering. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 2567–2577.

Daniel Khashabi, Yeganeh Kordi, and Hannaneh Hajishirzi. 2022. Unifiedqa-v2: Stronger generalization via broader cross-format training. *arXiv preprint arXiv:2202.12359*.

Tian Lan, Tsung-Chuan Chen, and Silvio Savarese. 2014. A hierarchical representation for future action prediction. In *European Conference on Computer Vision*, pages 689–704. Springer.

Zhenzhong Lan, Mingda Chen, Sebastian Goodman, Kevin Gimpel, Piyush Sharma, and Radu Soricut. 2019. Albert: A lite bert for self-supervised learning of language representations. *arXiv preprint arXiv:1909.11942*.

Hung Le, Chinnadhurai Sankar, Seungwhan Moon, Ahmad Beirami, Alborz Geramifard, and Satwik Kottur. 2021. Dvd: A diagnostic dataset for multi-step reasoning in video grounded dialogue. *arXiv preprint arXiv:2101.00151*.

J. Lee, W. Yoon, S. Kim, D. Kim, S. Kim, C. H. So, and J. Kang. 2019. Biobert: a pre-trained biomedical language representation model for biomedical text mining. *Bioinformatics*.

Jinhyuk Lee, Wonjin Yoon, Sungdong Kim, Donghyeon Kim, Sunkyu Kim, Chan Ho So, and Jaewoo Kang. 2020. Biobert: a pre-trained biomedical language representation model for biomedical text mining. *Bioinformatics*, 36(4):1234–1240.

Dongfang Li, Baotian Hu, Qingcai Chen, Weihua Peng, and Anqi Wang. 2020. Towards medical machine reading comprehension with structural knowledge and plain text. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1427–1438.

Ye Liu, Shaika Chowdhury, Chenwei Zhang, Cornelia Caragea, and Philip S Yu. 2020. Interpretable multi-step reasoning with knowledge extraction on complex healthcare question answering. *arXiv preprint arXiv:2008.02434*.

Anusri Pampari, Preethi Raghavan, Jennifer Liang, and Jian Peng. 2018. emrqa: A large corpus for question answering on electronic medical records. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 2357–2368.

Yifan Peng, Shankai Yan, and Zhiyong Lu. 2019. Transfer learning in biomedical natural language processing: an evaluation of bert and elmo on ten benchmarking datasets. *arXiv preprint arXiv:1906.05474*.

Kamal raj Kanakarajan, Bhuvana Kundumani, and Malaikannan Sankarasubbu. 2021. Bioelectra: pretrained biomedical text encoder using discriminators. In *Proceedings of the 20th Workshop on Biomedical Language Processing*, pages 143–154.

Pranav Rajpurkar, Robin Jia, and Percy Liang. 2018. Know what you don't know: Unanswerable questions for squad. *arXiv preprint arXiv:1806.03822*.

Pranav Rajpurkar, Jian Zhang, Konstantin Lopyrev, and Percy Liang. 2016. Squad: 100,000+ questions for machine comprehension of text. *arXiv preprint arXiv:1606.05250*.

Minjoon Seo, Aniruddha Kembhavi, Ali Farhadi, and Hannaneh Hajishirzi. 2016. Bidirectional attention flow for machine comprehension. *arXiv preprint arXiv:1611.01603*.

Xiaomeng Song, Yucheng Shi, Xin Chen, and Yahong Han. 2018. Explore multi-step reasoning in video question answering. In *Proceedings of the 26th ACM international conference on Multimedia*, pages 239–247.

Yu Sun, Shuohuan Wang, Yukun Li, Shikun Feng, Xuyi Chen, Han Zhang, Xin Tian, Danxiang Zhu, Hao Tian, and Hua Wu. 2019. Ernie: Enhanced representation through knowledge integration. *arXiv preprint arXiv:1904.09223*.

Min Tang, Jiaran Cai, and Hankz Hankui Zhuo. 2019. Multi-matching network for multiple choice reading comprehension. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, pages 7088–7095.

George Tsatsaronis, Michael Schroeder, Georgios Paliouras, Yannis Almirantis, Ion Androutsopoulos, Eric Gaussier, Patrick Gallinari, Thierry Artieres, Michael R Alvers, Matthias Zschunke, et al. 2012. Bioasq: A challenge on large-scale biomedical semantic indexing and question answering. Citeseer.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. *Advances in neural information processing systems*, 30.

David Vilares and Carlos Gómez-Rodríguez. 2019. Head-qa: A healthcare dataset for complex reasoning. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 960–966.

Shuohang Wang, Mo Yu, Shiyu Chang, and Jing Jiang. 2018. A co-matching model for multi-choice reading comprehension. *arXiv preprint arXiv:1806.04068*.

Zhilin Yang, Peng Qi, Saizheng Zhang, Yoshua Bengio, William W Cohen, Ruslan Salakhutdinov, and Christopher D Manning. 2018. Hotpotqa: A dataset for diverse, explainable multi-hop question answering. *arXiv preprint arXiv:1809.09600*.

Michihiro Yasunaga, Jure Leskovec, and Percy Liang. 2022. Linkbert: Pretraining language models with document links. *arXiv preprint arXiv:2203.15827*.

Michihiro Yasunaga, Hongyu Ren, Antoine Bosselut, Percy Liang, and Jure Leskovec. 2021. Qa-gnn: Reasoning with language models and knowledge graphs for question answering. *arXiv preprint arXiv:2104.06378*.

Xiang Yue, Bernal Jiménez Gutiérrez, and Huan Sun. 2020. Clinical reading comprehension: A thorough analysis of the emrqa dataset. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 4474–4486.

Manzil Zaheer, Guru Guruganesh, Kumar Avinava Dubey, Joshua Ainslie, Chris Alberti, Santiago Ontanon, Philip Pham, Anirudh Ravula, Qifan Wang, Li Yang, et al. 2020. Big bird: Transformers for longer sequences. *Advances in Neural Information Processing Systems*, 33:17283–17297.

Xiao Zhang, Ji Wu, Zhiyang He, Xien Liu, and Ying Su. 2018. Medical exam question answering with large-scale reading comprehension. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 32.

Xikun Zhang, Antoine Bosselut, Michihiro Yasunaga, Hongyu Ren, Percy Liang, Christopher D Manning, and Jure Leskovec. 2022. Greaselm: Graph reasoning enhanced language models for question answering. *arXiv preprint arXiv:2201.08860*.

Chen Zhao, Chenyan Xiong, Jordan Boyd-Graber, and Hal Daumé III. 2021. Multi-step reasoning over unstructured text with beam dense retrieval. *arXiv preprint arXiv:2104.05883*.

ArT: All-round Thinker for Unsupervised Commonsense Question Answering

Jiawei Wang[1,2], **Hai Zhao**[1,2*]
[1] Department of Computer Science and Engineering, Shanghai Jiao Tong University
[2] Key Laboratory of Shanghai Education Commission for Intelligent Interaction and Cognitive Engineering, Shanghai Jiao Tong University
wjw_sjt@sjtu.edu.cn, zhaohai@cs.sjtu.edu.cn

Abstract

Without labeled question-answer pairs for necessary training, unsupervised commonsense question-answering (QA) appears to be extremely challenging due to its indispensable unique prerequisite on commonsense source like knowledge bases (KBs), which are usually highly resource consuming in construction. Recently pre-trained language models (PLMs) show effectiveness as an alternative for commonsense clues when they play a role of knowledge generator. However, existing work either relies on large-scale in-domain or out-of-domain labeled data, or fails to generate knowledge of high quality in a general way. Motivated by human thinking experience, we propose an approach of **A**ll-**r**ound **T**hinker (ArT) by fully taking association during knowledge generating. In detail, our model first focuses on key parts in the given context, and then generates highly related knowledge on such a basis in an association way like human thinking. Besides, for causal reasoning, a reverse thinking mechanism is especially added to further enhance bidirectional inferring between cause and effect. ArT is totally unsupervised and KBs-free. We evaluate it on three commonsense QA benchmarks: COPA, SocialIQA and SCT. On all scales of PLM backbones, ArT shows its brilliant performance and outperforms previous advanced unsupervised models. Our code is available at https://github.com/WangJW424/commonsenseQA-ArT.

1 Introduction

Commonsense question-answering (QA) has been a more challenging natural language understanding (NLU) task than conventional QA tasks, for it requires extra commonsense knowledge, which cannot be directly acquired from the given context, to make an appropriate answer (Niu et al., 2021).

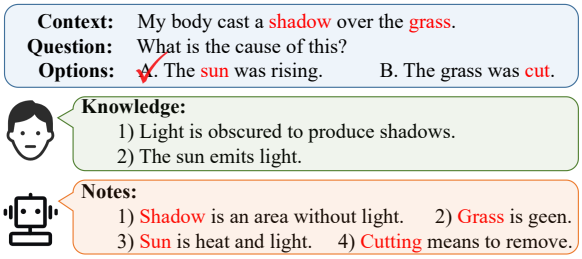

Figure 1: An commonsense QA exmaple in COPA. Text with blue background is the raw example and the gold answer is ticked. Text with green background is the needed commonsense knowledge. Text with orange background is notes generated by our ArT. Keyphrases are marked in red.

Figure 1 gives a typical example. To select the correct cause of the fact *"My body cast a shadow over the grass"*, models should know that *"Light is obscured to produce shadows"* and *"The sun emits light"*. As human beings, we can immediately make this association and judgment because above knowledge is deeply branded in our mind due to daily seeing and hearing, and maybe long-term education, which is so-called commonsense. However, it could be much difficult for models to be equipped with such ability. Besides, commonsense QA usually takes an unsupervised setting which makes this task even more difficult. This setting means there is no labeled training data available as commonsense is too broad to build a sufficient labeled training dataset (Shwartz et al., 2020).

To deal with it, prior work focused on building large-scale knowledge bases (KBs), also known as knowledge graphs (KGs), which usually contain millions of nodes and edges to record the relation between entities as relation triple: $<e_1, r, e_2>$ (Speer et al., 2017; Sap et al., 2019a). QA models then can be injected with commonsense through retrieving over KBs (Miller et al., 2016). Though impressive improvements are gained, such method is resource consuming in building or finding a

* Corresponding author. This paper was partially supported by Key Projects of National Natural Science Foundation of China (U1836222 and 61733011).

Proceedings of the 29th International Conference on Computational Linguistics, pages 1490–1501
October 12–17, 2022.

good and suitable KB (e.g. Building ConceptNet needs 30 GB of RAM[1]). Recently, pre-trained language models (PLMs) have been widely used and proved to be effective in commonsense QA (Niu et al., 2021; Xia et al., 2022). Thanks to the self-supervised pre-training strategy on large-scale unlabeled text (e.g. WebText (Radford et al., 2019) and Wikipedia[2]), PLMs are competent for many tasks even under a zero-shot setting. And fine-tuning PLMs on task-specific data in a supervised way can further produce even stronger results (Schick and Schütze, 2021a; Gao et al., 2021; Schick and Schütze, 2021b). Since high-quality labeled datasets are rare, researching on unsupervised commonsense QA is still of great significance. With the help of PLMs, existing studies have explored some good solutions (Niu et al., 2021; Bosselut et al., 2021; Shwartz et al., 2020), however they either rely on large-scale in-domain or out-of-domain labeled data, or need to be specifically designed for different tasks. In this work, we focus on designing a simple and general method to solve commonsense QA tasks in a strictly unsupervised way.

Based on two empirical observations from human thinking,

(1) Given a question with context, people firstly tend to focus on several key parts (as marked in red in Figure 1) and then make corresponding associations to choose the right answer;

(2) For causal reasoning, people tends to carry out a bidirectional inferring to assist answer selection or verify the answer correctness.

we propose **A**ll-**r**ound **T**hinker (ArT) for unsupervised commonsense QA, which includes two principal methods: notes taking (NT) and reverse thinking (RT). Specifically,

(1) NT extracts some keyphrases out of the context and then generates corresponding notes, which will be added as extra knowledge in later evaluation. Based on an unsupervised keyphrase extractor, we designed our knowledge generation rule to be simple and general.

(2) RT converts the causal inferring question to two different forms: (*cause* $\rightarrow$ *effect*) and (*effect* $\rightarrow$ *cause*), and then integrates the decisions made from the two reverse directions.

[1]Declared by Speer et al. (2017) at https://github.com/commonsense/conceptnet5

[2]https://www.english-corpora.org/wiki

Our proposed model is strictly unsupervised and KBs-free for all it needs is PLMs. We test ArT and validate its effectiveness on three commonsense QA benchmark datasets: COPA (Roemmele et al., 2011), SocialIQA (Sap et al., 2019b) and SCT (Mostafazadeh et al., 2016). Our contribution is summarized as follows:

- ArT can generate highly related knowledge through the imitation of human behaviour and thought, which is qualified with inherent interpretablility.
- Compared with existing work, ArT is simple and general, which totally gets rid of the needs of any labeled data and the specific design on any specific task.
- We conduct experiments on 3 commonsense QA benchmarks with 4 different scales of PLMs, so as to reach solid and reproducible results. The results show that ArT outperforms other advanced unsupervised models.

2 Related Work

2.1 Building and Usage of Knowledge Bases

In order to equip QA models with commonsense reasoning ability, previous researches were devoted to building and retrieving large-scale knowledge bases (KBs). ConceptNet (Speer et al., 2017) is one of the most famous traditional KBs, which contains over 21 million edges (for relations) and over 8 million notes (for entities). While ATOMIC (Sap et al., 2019a) focuses more on *if-then* relations between events.

Previous work applies a relatively standard routine to solve commonsense QA. Given an existing KB, QA models can retrieve relation triples $<e_1, r, e_2>$ over it, which can be injected into models as word embedding directly (Wang et al., 2014; Paul and Frank, 2019) or first converted to a complete sentence according to preset templates (e.g. <bird, *CapableOf*, fly> $\rightarrow$ *Bird can fly.*) and then integrated with the raw input text (Ma et al., 2019; Mihaylov and Frank, 2018; Bauer et al., 2018). This type of methods may give remarkable performance for commonsense QA (Weissenborn et al., 2017). However, building and retrieving of KBs are both resource consuming (Bosselut et al., 2021).

Bosselut et al. (2019) claimed that commonsense knowledge does not cleanly fit into a schema of comparing two entities with a known relation so

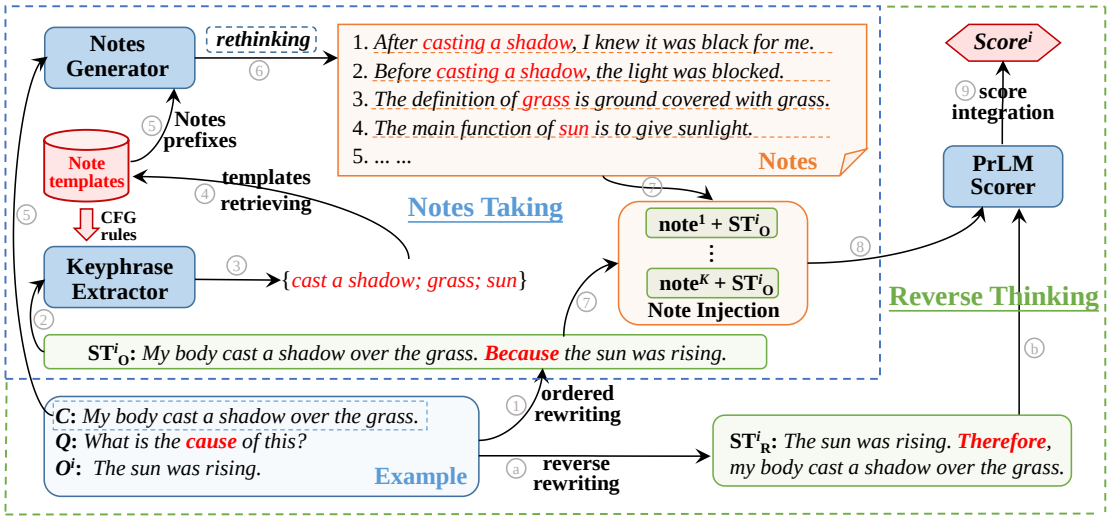

Figure 2: Overview of the proposed method **A**ll-**r**ound **T**hinker (ArT), which contains two principal method: Notes Taking (within blue dashline box) and Reverse Thinking (within green dashline box). The arrow denotes the flow of data stream, with number/letter within circle marking its order in time.

that they proposed COMET, which can generate rich and diverse commonsense descriptions in natural language. However, COMET has to extract knowledge triples from existing KBs as seed for PLM fine-tuning, while our method is totally KBs-free.

2.2 PLMs in Unsupervised Commonsense QA

Due to excellent performance and versatility, PLMs now dominate the backbone design of many NLP tasks, especially in QA (Wang et al., 2021; Zhang et al., 2021). Benefiting from the long-term pre-training on large-scale unlabeled text, PLMs are equipped with implicit or explicit commonsense (Davison et al., 2019). As a result, there comes a tendency to rely on PLMs as the sole source of world knowledge to solve commonsense QA in a zero-shot setting (Shwartz et al., 2020; Bosselut et al., 2021).

SEQA proposed by Niu et al. (2021) exploits generative PLMs to generate hundreds of pseudo-answers, and then applies Sentence-BERT (Reimers and Gurevych, 2019) to calculate semantic similarity between candidates and these generated pseudo-answers. A voting mechanism is designed to make final selection. Though SEQA shows impressive performance, it relies much on PLMs fine-tuned on large-scale labeled NLI (Natural Language Inference) datasets. Without such fine-tuning on such labeled datasets, SEQA's effectiveness will sharply decline. Instead, our method relies nothing but vanilla PLMs.

Shwartz et al. (2020) proposed Self-talk, which creatively applies PLMs as commonsense generator. It uses preset question prefixes to firstly prompt the PLM to generate information seeking questions (ISQs). ISQs will be put back to secondly prompt the PLM to generate their answers as clarifications, which will work as knowledge in later evaluation. Though this method is strictly unsupervised, the question prefixes are not general and have to be carefully designed according to different tasks. Besides, Self-talk fails to generate knowledge of high quality since the generation of ISQs and clarifications are both erratic. Inspired by Self-talk and towards its shortage, we designed a general method to generate knowledge of high relevance.

2.3 Unsupervised Keyphrase Extraction

Keyphrase extraction (KE) is the task of selecting several words or phrases that can summarize the main topic of the document (Hasan and Ng, 2014). Unsupervised KE methods use different features of the document, such as word frequency, position feature, relationship between words, etc (Mihalcea and Tarau, 2004; Bougouin et al., 2013).

SIFRank (Sun et al., 2020) is one of the most advanced unsupervised keyphrase extractors. It leverages different features of words in document and evaluates the weight of candidate phrases according to word embedding provided by PLMs. SIFRank is originally designed to extract key noun phrases from given documents, and we made slight modifications to adapt it to our model.

Key	Value	Replacing rule
"NP"	{ "*The definition of* [NP] *is*", "*The main function of* [NP] *is*", "[NP] *is a/an*" }	directly replace
"VP"	{ "[VP] *means*", "*After* [VP], ", "*Before* [VP], " }	convert to gerund first
"PNP"	{ "[PNP] *is a/an*", "[PNP] *felt*", "*After this,* [PNP]", "[PNP] *did this because*" }	directly replace

Table 1: Note templates lookup table.

3 All-round Thinker

Inspired by the behaviour and thought of human beings during solving questions requiring commonsense, we propose **A**ll-**r**ound **T**hinker (ArT). Figure 2 gives its overview. We will firstly introduce the task definition and the basic solution, and then describe the two principal methods of ArT in detail: Notes Taking (NT) and Reverse Thinking (RT).

3.1 Task Definition and Basic Solution

This work focuses on unsupervised commonsense QA task in multi-choice style, which consists of a *context* (C) with related *question* (Q), and asks models to select a single answer from a given *option* set: $O = \{O^i\}_{i=1}^{|O|}$. Though there are some variants of the task form, such as Piqa (Bisk et al., 2020) and WinoGrande (Sakaguchi et al., 2020), they can be conveniently transformed into this normalized formulation: (C, Q, O) (Shwartz et al., 2020).

Following previous work, we adopt a basic solution which uses PLMs as scorer. Based on the pre-training strategy: predicting the probability distribution of token n according to previous $(n-1)$-gram, e.g. OpenAI GPT (Radford et al., 2019), or bidirectional context, e.g. BERT (Devlin et al., 2019), PLMs are qualified for the role of *option* scorer even under a zero-shot setting (Bosselut et al., 2021). Sentence likelihood is the commonly-used scoring function for O^i:

$$\begin{aligned} \boldsymbol{S}(O^i|C,Q) &= P_{LM}(O^i|C,Q) \\ &= \frac{1}{|O^i|}\sum_{t=1}^{|O^i|}\log P_{LM}(O_t^i|C,Q,O_{<t}^i) \end{aligned} \tag{1}$$

where P_{LM} refers to the probability function abstracted from PLMs. The final predicted answer ($\hat{A}$) is selected as:

$$\hat{A} = \underset{O^i}{\operatorname{argmax}}\, \boldsymbol{S}(O^i|C,Q) \tag{2}$$

3.2 Notes Taking

To make full use of PLMs' potential of commonsense reasoning and overcome the shortage of Self-talk (Shwartz et al., 2020), we propose notes taking (NT) to generate commonsense descriptions in natural language (defined as "notes" in our work) in a simple and general way. NT is composed of two phases: keyphrase extraction and notes generation.

3.2.1 Keyphrase Extraction

Taking O^i and its corresponding C and Q as input (as shown in left bottom of Figure 2), ArT firstly rewrites the original interrogative Q into a declarative form (We followed the question rewriting method proposed by Shwartz et al. (2020)) and then concatenate $<C, Q, O^i>$ as a statement (ST_O^i). Then, we use an unsupervised keyphrase extractor to extract keyphrases from ST_O^i. To be specific, we extract three types of phrases: noun phrase (NP), verb phrase (VP) and person name phrase (PNP).

To implement this, for each type of phrase we designed a simple CFG (context-free grammar (Chomsky and Schützenberger, 1959)) rule to extract it out of the whole sentence, as following:

- $NP \rightarrow (nn|adj) * + nn$
- $VP \rightarrow vb + (pr)\{0,1\} + NP$
- $PNP \rightarrow (pn)\{1,2\}$

where *VP*, *NP* and *PNP* are non-terminators; vb (verb), nn (noun), adj (adjective), pn (person name) and pr (preposition) are terminators; '+' means concatenation; $\{a,b\}$ means repetition times range from a to b. '$*$' is equivalent to $\{0,\infty\}$.

Then, we add these CFG rules to the RegexpPaser tool of NLTK (Bird, 2006). We will extract top 5 most important keyphrases without any label or fine-tuned model but word embeddings obtained from a PLM, i.e. ELMo (Peters et al., 2018).

3.2.2 Notes Generation

Once keyphrases are obtained, we can retrieve a preset note templates set for getting notes prefixes. Considering that: (1) For an object (NP), people will think "what is it" and "what is it for"; (2) For a behavior (VP), people will think "what does it mean" and "what is the cause/effect"; (3) For a person (PNP), people will think "who is he/she"

and "what is his/her feeling/motivation/reaction", our proposed general-purpose note templates set is presented in Table 1.

We use the types of keyphrases as $keys$ and templates lists as $values$ to build the note templates set as a lookup table. Given such a lookup table, we can immediately retrieve the note templates for our extracted keyphrases and simply replace the tag ([NP], [VP] and [PNP]) with these keyphrases to form the note prefixes[3]. Though this lookup table seems to be simple, it shows effectiveness and generality on different benchmarks.

Next, for each note prefix, we will concatenate it to C and input them into a generative PLM to generate the complete note. Specifically, nucleus sampling (Holtzman et al., 2020) with $p = 0.8$ is applied as the decoding strategy rather than greedy/beam search to increase the **diversity** of generated text. Meanwhile, to ensure the **quality** and scale the number of generated notes, we sort all the notes according to their perplexity estimated by the PLM. We denote this process as ***Rethinking***. Finally, we will retain top K notes to construct the notes set: $\{note^k\}_{k=1}^{K}$, as shown in middle top of Figure 2. Each $note^k$ will be inserted into ST_O^i as extra knowledge to assist later *option* scoring. The score of O^i w.r.t $note^k$ is calculated as:

$$score_O^{i,k} = P_{LM}(O^i|note^k + ST_O^i - O^i)$$
$$= \frac{1}{|O^i|}\sum_{t=1}^{|O^i|} \log P_{LM}(O_t^i|note^k + ST_O^i - O^i + O_{<t}^i) \quad (3)$$

in which '+' means concatenation and '-' means removing. Eventually, a voting mechanism is applied to integrate the scores w.r.t all notes as:

$$Score_O^i = \frac{1}{K}\sum_{k=1}^{K} score_O^{i,k} \quad (4)$$

It is worth noting that there is no need in training from any labeled data throughout NT. And there is no need to modify the note templates according to different tasks since our note templates is designed for general purpose.

3.3 Reverse Thinking

For causal reasoning questions, we additionally introduce reverse thinking (RT) which conducts a bidirectional inferring between *cause* and *effect*. To implement this, besides the ordered rewriting (ST_O^i) as mentioned in Section 3.2.1, we also apply reverse rewriting that concatenates them in the order of $<O^i, Q_R, C>$ (denoted as ST_R^i), as shown in the right bottom of Figure 2. Note that Q_R is the opposite question of Q. To be specific, after question rewriting, "*Because*" and "*Therefore*" are two opposite questions in causal reasoning tasks.

To conduct bidirectional inferring, except for $Score_O^i$ as introduced in Section 3.2 for the ordered inferring: $C + Q \rightarrow O^i$, we set another scoring function $Score_R^i$ for reverse inferring: $O^i + Q_R \rightarrow C$, as:

$$Score_R^i = P_{LM}(C|ST_R^i - C)$$
$$= \frac{1}{|C|}\sum_{t=1}^{|C|} \log P_{LM}(C_t|ST_R^i - C + C_{<t}) \quad (5)$$

To take advantage of bidirectional inferring, we design a mixed scoring function by simply compute the average value of the above two:

$$Score_X^i = \frac{1}{2}(Score_O^i + Score_R^i) \quad (6)$$

Finally, formula (2) is applied to select the answer by replacing $\boldsymbol{S}$ with $Score_O^i$ (default) or $Score_X^i$ (for causal reasoning). From the perspective of model enhancing, averaging $Score_O^i$ and $Score_R^i$ can be regarded as the assembly of two models ("cause" model and "effect" model), which is a common method to enhance model performance and robustness.

4 Experiment

4.1 Datasets

ArT is evaluated on three different commonsense QA benchmarks: COPA(Roemmele et al., 2011), SocialIQA(Sap et al., 2019b) and SCT(Mostafazadeh et al., 2016). Here are the detailed information:

- **COPA**[4] (**C**hoice **o**f **P**lausible **A**lternatives): evaluates the ability of causal reasoning about a certain event, which is described as a single sentence. Each question is accompanied with two candidate options.

[3] In order to keep a correct grammar, for VP we will firstly convert its verb into gerund form before replacing.

[4] https://people.ict.usc.edu/~gordon/copa.html

- **SocialIQA**[5] (**Social** **I**nteraction **Q**uestion **A**nswering): evaluates the reasoning ability on social interactions. It has various questions, including the subject's motivation, reaction, personality, etc. Each question is accompanied with three candidate options.

- **SCT** [6] (**S**tory **C**loze **T**est): requires models to select the right ending of the given short story from two alternatives. Each story is composed of four sentences.

Since two test sets of the three datasets are hidden, we report all results on dev sets. Note that the labels are kept invisible and only used for final accuracy evaluating.

4.2 Baseline and Contrastive models

Our baseline is constructed as only using PLMs as scorer without any explicit knowledge injection. Formula (1) is applied as the scoring function as described in Section 3.1. We also compare ArT with other advanced unsupervised models:

- **Self-talk**(Shwartz et al., 2020): It acquires knowledge through a two-stage prompting of PLMs. Different question prefixes had to be specially designed for different tasks.

- **SEQA**(Niu et al., 2021): It applies PLMs to generate hundreds of pseudo-answers and compares them with each $option$. However, its scorer relies on PLMs fine-tuned on large-scale labeled NLI datasets, which is not strictly unsupervised. For fair comparison, we design another setting that replaces the fine-tuned PLM with the original one (only pre-trained on unlabeled text) .

- **CGA**(Bosselut et al., 2021): It employs a generative KB COMET(Bosselut et al., 2019), which is trained on an existing seed KB (e.g. ConceptNet), to construct context-relevant knowledge graphs to reason over.

4.3 Setup

Following previous work, we employ OpenAI GPT(Radford et al., 2019) as the PLM backbone. To reach solid and reproducible results, we conduct experiments on GPT-2 of 4 different scales: distil, medium, large and xlarge. For ArT and Self-talk, the same scale GPT is applied during both knowledge generating and $option$ scoring. For SEQA, GPTs of different scales are used for pseudo-answers generation and $\text{SRoBERTa}_{\text{large}}$(Reimers and Gurevych, 2019) is used for semantic similarity calculation. To distinguish, $\text{SRoBERTa}^{NLI}_{\text{large}}$ and $\text{SRoBERTa}^{Origin}_{\text{large}}$ refer to $\text{SRoBERTa}_{\text{large}}$ with and without further fine-tuning on NLI datasets, respectively. For Self-talk[7] and SEQA[8], we re-run their codes with their original settings and report both our re-running results[9] and results coming from their publications. For CGA, we report results provided by Niu et al. (2021). For ArT, we modified the open source code of SIFRank[10] to enable it to extract more kinds of phrases rather than only noun phrase. The size of notes set (K) is set as 32 as default. Except for $Score^i_O$, ArT takes another setting $Score^i_X$ on COPA.

4.4 Results

Table 2 shows the results on three benchmarks. The results of our re-running is highly consistent with those reported in their publications (last column). Note that published results on COPA seem to have a deviation with our reproduction. It because that they are on test set, while ours on dev set. Shwartz et al. (2020) reported 66.0% on COPA in their paper, which is tested on the dev set of a small version which contains 1/5 instances of that other researches used. And ArT reaches 68.0% on that set under the same setting.

On all datasets, ArT obtains state-of-the-art performance among almost all fully unsupervised models. Besides, it is noticed that on GPT of different scales, ArT with NT stably brings positive improvement over baseline. On causal reasoning task COPA, adding RT will bring further accuracy improvement.

In contrast with ArT, Self-talk fails to maintain effectiveness, whose accuracy is even slightly lower than baseline from time to time, especially on SCT. It indicates that the knowledge generated by Self-talk could be noisy and as a result it misguides model evaluation.

With the help of $\text{SRoBERTa}^{NLI}_{\text{large}}$, SEQA can reach very impressive results on all the datasets, especially SCT (exceed all models than over 10%).

[5] https://leaderboard.allenai.org/socialiqa

[6] https://www.cs.rochester.edu/nlp/rocstories/

[7] https://github.com/vered1986/self_talk

[8] https://github.com/heyLinsir/Semantic-based-QA

[9] For each setting except $\text{GPT-2}_{\text{xlarge}}$ (limited by the computational power), we run 3 times and report the average number.

[10] https://github.com/sunyilgdx/SIFRank

Dataset	Models	Our (re-)running				Published
		DistilGPT-2	GPT-2_{medium}	GPT-2_{large}	GPT-2_{xlarge}	GPT-2_{xlarge}
COPA	Baseline	57.8	62.4	65.8	66.0	–
	SEQA	51.4 (63.0)	53.0 (68.4)	53.8 (72.0)	54.4 (75.4)	79.4
	Self-talk	59.8 (↑2.0)	65.0 (↑2.6)	66.6 (↑0.8)	66.2 (↑0.2)	68.6
	CGA	–	–	–	–	72.2
	ArT	60.2 (↑2.4)	64.8 (↑2.4)	67.0 (↑1.2)	67.6 (↑1.6)	–
	ArT ($Score^i_X$)	**61.0** (↑3.2)	**65.6** (↑3.2)	**69.4** (↑3.6)	**69.8** (↑3.8)	–
SocialIQA	Baseline	41.3	44.3	45.5	45.9	–
	SEQA	34.9 (43.9)	35.9 (44.6)	36.5 (46.6)	36.6 (47.5)	47.5
	Self-talk	40.5 (↓0.8)	44.8 (↑0.5)	46.1 (↑0.6)	47.2 (↑1.3)	47.5
	CGA	–	–	–	–	45.4
	ArT	**42.0** (↑0.7)	**45.6** (↑1.3)	**47.6** (↑2.1)	**47.3** (↑1.4)	–
SCT	Baseline	59.6	67.4	69.1	70.5	–
	SEQA	50.7 (74.7)	53.3 (80.5)	54.2 (82.4)	54.9 (83.2)	83.2
	Self-talk	59.8 (↑0.2)	**68.5** (↑1.1)	69.2 (↑0.1)	70.4 (↓0.1)	70.4
	CGA	–	–	–	–	71.5
	ArT	**60.2** (↑0.6)	68.3 (↑0.9)	**69.5** (↑0.4)	**71.6** (↑1.1)	–

Table 2: Accuracy (%) on COPA, SocialIQA and SCT. All results except last column are run by ourselves. Best results are depicted in boldface (only consider fully unsupervised models for fairness). ↑/↓ refer to relative increase/decrease compared with baseline. For SEQA, we list results of two settings: SRoBERTa$^{Origin}_{\text{large}}$ (before brackets) and SRoBERTa$^{NLI}_{\text{large}}$ (in brackets).

However, when working in a strictly unsupervised mode, this method quickly becomes invalid (close to random selection).

We also observe a common phenomenon on all our ArT, baseline and contrastive models: when enlarging GPT-2 from large (750M parameters) to xlarge (1500M parameters), we encounter no obvious model performance increasing and even decreasing in some scenarios. It indicates that there could be a limit to only using the method of increasing model parameters to improve the performance of language models as sentence scorer or knowledge generator.

5 Analysis and Discussion

5.1 Effect of Different Modules

In order to determine the source of performance growth, we conduct ablation study on ArT, as shown in Table 3. As expected, injecting knowledge with our NT has positive effect on all three tasks. When removing the *Rethinking* mechanism, all results slightly decrease, which indicates that *Rethinking* can help increase the quality of generated knowledge. Besides, $Score^i_X$ can bring further improvement whether or not NT is employed on COPA.

Models	COPA	SocialIQA	SCT
Baseline	65.8	45.5	69.1
+$Score^i_X$	68.9	–	–
+NT	67.0	**47.6**	**69.5**
+NT−*Rethinking*	66.2	46.7	69.4
+NT+$Score^i_X$	**69.4**	–	–

Table 3: Ablation study for ArT modules on GPT-2_{large}.

5.2 Number of Notes and Keyphrases

Figure 3 shows the effect of notes set size K. On all the benchmarks, along with the increasing of K, the accuracy curves basically show an upward trend, which indicates that as the number of generated knowledge increases, ArT will not accumulate too much noise as Niu et al. (2021) observed in Self-talk. Therefore, compared with Self-talk, ArT tends to generate highly related knowledge, which is contributed by our NT mechanism.

In our experiments (Section 4), the number (N) of keyphrases to extract is set as 5 as default. To show the effect of N, we conduct an ablation study on SCT by setting $N \in \{1, 3, 5, 7, 9\}$. The PLM is selected as GPT-2_{distil}. Table 4 shows the results. It is noticed that extracting more keyphrases does

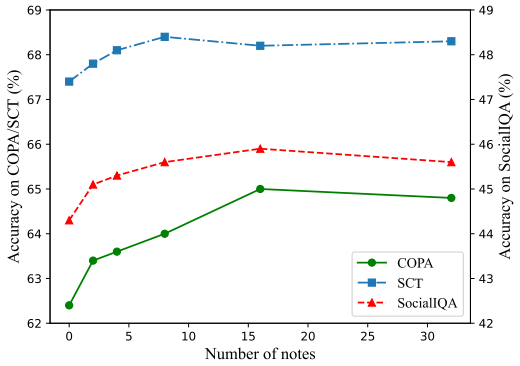

Figure 3: Accuracy curves of ArT on GPT-2_{medium} w.r.t the size K of notes set.

N	1	3	5	7	9
baseline			59.6		
ArT	59.6	**60.2**	**60.2**	60.0	60.0

Table 4: Accuracy (%) of ArT (GPT-2_{distil}) on SCT under different settings of N.

not always results in a better performance. But in general, the choice of N does not have an obvious impact on the final performance.

Dataset	Model	Rationality	Usefulness
COPA	Self-talk	0.24	0.20
	ArT	**0.32**	**0.28**
SocialIQA	Self-talk	0.17	0.16
	ArT	**0.26**	**0.27**

Table 5: Human evaluation on the rationality and usefulness of generated knowledge.

5.3 Quality of Generated Knowledge

To compare the quality of generated knowledge (whether the knowledge is reasonable enough to be a "fact" and correlative enough to be useful), we conduct human evaluation on the rationality (-1: *irrational*, 0: *meaningless*, 1: *rational*) and usefulness (-1: *negative*, 0: *neutral*, 1: *positive*) of knowledge generated by ArT and Self-talk. Two annotators are asked to independently annotate 100 randomly selected knowledge for both COPA and SocialIQA. The overall average scores of two annotators for each dataset are shown in Table 5. It is noticed that ArT outperforms Self-talk in generating knowledge with both rationality and usefulness.

We further make statistics on the kinds of all the knowledge generated by Self-talk and ArT on COPA. We divide them into four classifications:

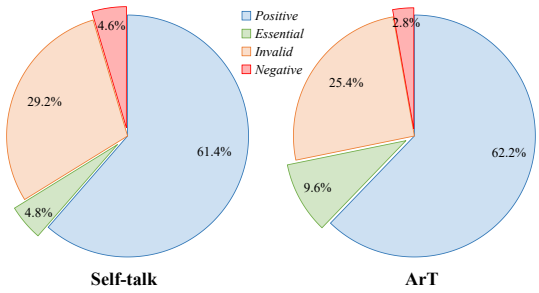

Figure 4: Statistics on the quality of knowledge generated by Self-talk and ArT on COPA.

- *Positive*: Baseline makes the right prediction. Adding the knowledge still makes a right prediction.
- *Essential*: Baseline makes the wrong prediction. Adding the knowledge helps make a right prediction.
- *Invalid*: Baseline makes the wrong prediction. Adding the knowledge still makes a wrong prediction.
- *Negative*: Baseline makes the right prediction. Adding the knowledge leads to a wrong prediction.

As shown in Figure 4, by doubling *Essential* and reducing *Negative* knowledge, ArT outperforms Self-talk in generating high-quality commonsense.

Figure 5 and 6 in Appendix show some examples of knowledge generated by ArT, Self-talk and SEQA for COPA, SocialIQA and SCT. As can be seen, ArT generates highly related knowledge on the basis of focusing on keyphrases of given context. While Self-talk may generate meaningless or even noisy knowledge. It is also noticed that SEQA could generate very reasonable pseudo-answers. And to distinguish the relationship between these pseudo-answers and given *options* relies much on a sentence embedding extractor fine-tuned on labeled NLI data.

5.4 Effect of Note Types

As we designed three types of notes: NP, VP and PNP, we only consider one type at one time to show the effect of each in different benchmarks, as shown in Table 6 (The PLM is GPT-2_{medium}).

It is noticed that each type of notes has a positive effect when applied separately, and their combination works better. Note that COPA has no person

Dataset	NP	VP	PNP	All / None
COPA	63.6	63.2	62.4	64.8 / 62.4
SocialIQA	45.0	44.8	45.3	45.6 / 44.3
SCT	68.0	67.9	68.1	68.3 / 67.4

Table 6: Effect of each type of notes on different tasks.

name phase (PNP), so PNP notes does not work on it. On SocialIQA, PNP works best among three types of notes. This is reasonable since SocialIQA focus much on human behavior in social interactions.

6 RT for Other Questions

Although RT is designed to enhance causal reasoning, we also explored if RT has the potential to help other questions. To investigate this, we apply it on other two datasets: SocialIQA and SCT. Note that in these tasks the opposite question is hard to define, therefore we simply exchange the position of $option$ and $context$, that is $ST_O^i =< C, Q, O^i >$ and $ST_R^i =< O^i, Q, C >$. We employ three scoring functions on basis of GPT-2_{medium} baseline, as shown in Table 7.

Functions	COPA	SocialIQA	SCT
$Score_O^i$	62.4	**44.3**	**67.4**
$Score_R^i$	63.2	42.5	62.8
$Score_X^i$	**65.3**	44.1	65.4

Table 7: Accuracy (%) of GPT-2_{medium} baseline with different scoring functions.

On all the tasks, $Score_O^i$ and $Score_R^i$ can obtain positive results (much better than random selection). By simply averaging ($Score_X^i$), on COPA it can reach a higher score. On other tasks, it falls into the middle of $Score_O^i$ and $Score_R^i$. Considering that $Score_R^i$ shows a comparable performance with $Score_O^i$ by simply exchanging the position of $option$ and $context$, developing a general method for opposite question definition or designing a more exquisite method to integrate $Score_O^i$ and $Score_R^i$ perhaps could make RT suitable for questions beyond causal reasoning, which will be the key point of our future work.

7 Conclusion

Commonsense QA has been a challenging task for it requires extra knowledge beyond the given context. In consideration of the high resource consumption of building knowledge bases (KBs) and the rarity of high-quality labeled data, this work aims at addressing commonsense QA in a fully KBs-free and unsupervised way. Inspired by the association process of human thinking, we propose **A**ll-**r**ound **T**hinker (ArT), which first focuses on key parts in the given context, and then generates highly related knowledge on such a basis in an association way. Besides, a reverse thinking mechanism is introduced to further enhance bidirectional inferring for causal reasoning as human will do. We test ArT on three benchmarks: COPA, SocialIQA and SCT. ArT outperforms previous advanced unsupervised models and shows stable performance on all scales of PLM backbones.

References

Lisa Bauer, Yicheng Wang, and Mohit Bansal. 2018. Commonsense for generative multi-hop question answering tasks. In *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*.

Steven Bird. 2006. Nltk: the natural language toolkit. In *Proceedings of the COLING/ACL 2006 Interactive Presentation Sessions*, pages 69–72.

Yonatan Bisk, Rowan Zellers, Jianfeng Gao, Yejin Choi, et al. 2020. Piqa: Reasoning about physical commonsense in natural language. In *Proceedings of the AAAI Conference on Artificial Intelligence*, pages 7432–7439.

Antoine Bosselut, Ronan Le Bras, and Yejin Choi. 2021. Dynamic neuro-symbolic knowledge graph construction for zero-shot commonsense question answering. In *Proceedings of the 35th AAAI Conference on Artificial Intelligence (AAAI)*.

Antoine Bosselut, Hannah Rashkin, Maarten Sap, Chaitanya Malaviya, Asli Celikyilmaz, and Yejin Choi. 2019. Comet: Commonsense transformers for automatic knowledge graph construction. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 4762–4779.

Adrien Bougouin, Florian Boudin, and Béatrice Daille. 2013. Topicrank: Graph-based topic ranking for keyphrase extraction. In *International joint conference on natural language processing (IJCNLP)*, pages 543–551.

Noam Chomsky and Marcel P Schützenberger. 1959. The algebraic theory of context-free languages. In *Studies in Logic and the Foundations of Mathematics*, volume 26, pages 118–161. Elsevier.

Joe Davison, Joshua Feldman, and Alexander M Rush. 2019. Commonsense knowledge mining from pretrained models. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 1173–1178.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. Bert: Pre-training of deep bidirectional transformers for language understanding. In *NAACL-HLT*.

Tianyu Gao, Adam Fisch, and Danqi Chen. 2021. Making pre-trained language models better few-shot learners. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 3816–3830.

Kazi Saidul Hasan and Vincent Ng. 2014. Automatic keyphrase extraction: A survey of the state of the art. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics*, pages 1262–1273.

Ari Holtzman, Jan Buys, Li Du, Maxwell Forbes, and Yejin Choi. 2020. The curious case of neural text degeneration. In *ICLR*.

Kaixin Ma, Jonathan Francis, Quanyang Lu, Eric Nyberg, and Alessandro Oltramari. 2019. Towards generalizable neuro-symbolic systems for commonsense question answering. *EMNLP 2019*, page 22.

Rada Mihalcea and Paul Tarau. 2004. Textrank: Bringing order into text. In *Proceedings of the 2004 conference on empirical methods in natural language processing*, pages 404–411.

Todor Mihaylov and Anette Frank. 2018. Knowledgeable reader: Enhancing cloze-style reading comprehension with external commonsense knowledge. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 821–832.

Alexander H Miller, Adam Fisch, Jesse Dodge, Amir-Hossein Karimi, Antoine Bordes, and Jason Weston. 2016. Key-value memory networks for directly reading documents. In *EMNLP*.

Nasrin Mostafazadeh, Nathanael Chambers, Xiaodong He, Devi Parikh, Dhruv Batra, Lucy Vanderwende, Pushmeet Kohli, and James Allen. 2016. A corpus and cloze evaluation for deeper understanding of commonsense stories. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 839–849.

Yilin Niu, Fei Huang, Jiaming Liang, Wenkai Chen, Xiaoyan Zhu, and Minlie Huang. 2021. A semantic-based method for unsupervised commonsense question answering. In *ACL-IJCNLP 2021*.

Debjit Paul and Anette Frank. 2019. Ranking and selecting multi-hop knowledge paths to better predict human needs.

Matthew Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer. 2018. Deep contextualized word representations. In *NAACL*, pages 2227–2237.

Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, Ilya Sutskever, et al. 2019. Language models are unsupervised multitask learners. *OpenAI blog*, 1(8):9.

Nils Reimers and Iryna Gurevych. 2019. Sentence-BERT: Sentence embeddings using Siamese BERT-networks. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 3982–3992. Association for Computational Linguistics.

Melissa Roemmele, Cosmin Adrian Bejan, and Andrew S Gordon. 2011. Choice of plausible alternatives: An evaluation of commonsense causal reasoning. In *2011 AAAI Spring Symposium Series*.

Keisuke Sakaguchi, Ronan Le Bras, Chandra Bhagavatula, and Yejin Choi. 2020. WinoGrande: An adversarial winograd schema challenge at scale. In *Proceedings of the AAAI Conference on Artificial Intelligence*, pages 8732–8740.

Maarten Sap, Ronan Le Bras, Emily Allaway, Chandra Bhagavatula, Nicholas Lourie, Hannah Rashkin, Brendan Roof, Noah A Smith, and Yejin Choi. 2019a. Atomic: An atlas of machine commonsense for if-then reasoning. In *Proceedings of the AAAI Conference on Artificial Intelligence*, pages 3027–3035.

Maarten Sap, Hannah Rashkin, Derek Chen, Ronan Le Bras, and Yejin Choi. 2019b. Social IQa: Commonsense reasoning about social interactions. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 4463–4473.

Timo Schick and Hinrich Schütze. 2021a. Exploiting cloze-questions for few-shot text classification and natural language inference. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*, pages 255–269.

Timo Schick and Hinrich Schütze. 2021b. It's not just size that matters: Small language models are also few-shot learners. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 2339–2352.

Vered Shwartz, Peter West, Ronan Le Bras, Chandra Bhagavatula, and Yejin Choi. 2020. Unsupervised commonsense question answering with self-talk. In

Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP), pages 4615–4629.

Robyn Speer, Joshua Chin, and Catherine Havasi. 2017. Conceptnet 5.5: An open multilingual graph of general knowledge. In *Thirty-first AAAI conference on artificial intelligence*.

Yi Sun, Hangping Qiu, Yu Zheng, Zhongwei Wang, and Chaoran Zhang. 2020. Sifrank: a new baseline for unsupervised keyphrase extraction based on pre-trained language model. *IEEE Access*, 8:10896–10906.

Jiawei Wang, Hai Zhao, Yinggong Zhao, and Libin Shen. 2021. What if sentence-hood is hard to define: A case study in Chinese reading comprehension. In *Findings of the Association for Computational Linguistics: EMNLP 2021*, pages 2348–2359. Association for Computational Linguistics.

Zhen Wang, Jianwen Zhang, Jianlin Feng, and Zheng Chen. 2014. Knowledge graph and text jointly embedding. In *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*, pages 1591–1601.

Dirk Weissenborn, Tomáš Kočiskỳ, and Chris Dyer. 2017. Dynamic integration of background knowledge in neural nlu systems.

Mengzhou Xia, Mikel Artetxe, Jingfei Du, Danqi Chen, and Ves Stoyanov. 2022. Prompting electra: Few-shot learning with discriminative pre-trained models. *arXiv preprint arXiv:2205.15223*.

Zhuosheng Zhang, Junjie Yang, and Hai Zhao. 2021. Retrospective reader for machine reading comprehension. In *Proceedings of the 35th AAAI Conference on Artificial Intelligence (AAAI-21)*.

		COPA	SocialIQA
Instance	**C:**	*The woman hired a lawyer.*	*Carson was excited to wake up to attend school.*
	Q:	*What is the cause of this?*	*Why did Carson do this?*
	O:	a) *She wanted to sue her employer.* b) *She decided to run for office.*	a) *Take the big test.* b) *Just say hello to friends.* c) *Go to bed early.*
ArT	**top-1:**	*After hiring a lawyer, she sued the man for sexual harassment.*	*Carson felt great and enjoyed being around people again.*
	top-2:	*Before hiring a lawyer, she has little chances to win in the court.*	*Attending school means working hard and learning something new everyday.*
Self-talk	**top-1:**	*Before [b)], the prosecutor said her client was a criminal.*	*Carson did this because they wanted him in the classroom.*
	top-2:	*"Lawyer" anyway is the word that I'm using to describe it.*	*Carson did this because they wanted him there for his education.*
SEQA	**top-1:**	*She thought the employer was trying to rip her off.*	*To focus on his studies.*
	top-2:	*She was worried her employer would be fired.*	*To play football with his friends and play street basketball.*

Figure 5: Top two most contributing generated knowledge for instances of COPA and ScocialIQA. The correct options are underlined. Keyphrases extracted by ArT are marked in red.

		SCT
Instance	**C:**	*Rick grew up in a troubled household. He never found good support in family, and turned to gangs. It wasn't long before Rick got shot in a robbery. The incident caused him to turn a new leaf.*
	Q:	*What is the ending of the story?*
	O:	a) *Rick is happy now.* b) *Rick joined a gang.*
ArT	**top-1:**	*After turning a new leaf, he decided that fighting crime was more important than robbing banks.*
	top-2:	*Turning a new leaf means not committing any crimes again.*
Self-talk	**top-1:**	*Rick is happy now means that he has learned from his mistakes.*
	top-2:	*Rick is happy now means that he can get his life back on track.*
SEQA	**top-1:**	*He was now a good person.*
	top-2:	*His life has dramatically improved with a period of rehabilitation and good mental health.*

Figure 6: Top two most contributing generated knowledge for instances of SCT. The correct options are underlined. Keyphrases extracted by ArT are marked in red.

Teaching Neural Module Networks to Do Arithmetic

Jiayi Chen , Xiao-Yu Guo , Yuan-Fang Li , and Gholamreza Haffari

Faculty of Information Technology, Monash University, Melbourne, Australia
jche0069@student.monash.edu
{xiaoyu.guo,yuanfang.li,gholamreza.haffari}@monash.edu

Abstract

Answering complex questions that require multi-step multi-type reasoning over raw text is challenging, especially when conducting numerical reasoning. Neural Module Networks (NMNs), follow the *programmer-interpreter* framework and design trainable modules to learn different reasoning skills. However, NMNs only have limited reasoning abilities, and lack numerical reasoning capability. We upgrade NMNs by: (a) bridging the gap between its *interpreter* and the complex questions; (b) introducing addition and subtraction modules that perform numerical reasoning over numbers. On a subset of DROP, experimental results show that our proposed methods enhance NMNs' numerical reasoning skills by 17.7% improvement of F1 score and significantly outperform previous state-of-the-art models.

1 Introduction

Complex Question Answering (CQA) over text is a challenging task in Natural Language Understanding (NLU). Based on the *programmer-interpreter* paradigm, Neural Module Networks (NMNs) (Gupta et al., 2020) learn to first parse complex questions as executable programs composed of various predefined trainable modules, and then execute such programs (implemented by modules) over the given paragraph to predict answers of all kinds. NMNs achieve competitive reasoning performance on a subset of DROP (Dua et al., 2019), and possess remarkable interpretability that is also important for CQA.

However, NMNs' numerical reasoning capability is insufficient: it is incapable of handling arithmetic operations such as addition and subtraction between numbers, which make up nearly 40% questions of the DROP dataset. Moreover, a gap exists between the interpreter and the complex question since there is no interaction between them. Motivated by these, we propose two methods to improve NMNs' numerical reasoning skills.

First, we incorporate the original question in the interpreter, aiming to **directly provide question information in the "execution" process**, especially number-related questions. The intuition behind is that, in the original NMNs, questions participate in the process only through the programmer. This can cause a distance between queries and returns. For example, in Figure 1, the first row shows that the original NMNs found the wrong event (i.e., 'besieged Sinj') solely based on the paragraph information. In contrast, our model NMNs$_{\pm}$ can easily target the correct event (i.e., 'Sinj finally fell') with the help of question information.

Second, we introduce new modules to **support addition and subtraction of up to three numbers**. Endowing NMNs with the ability to support arithmetic can greatly boost its overall performance on DROP and beyond. For instance, in Figure 1, the second row shows that the original NMNs improperly adopt the `find-num` module for the addition question because the module set does not cover such an arithmetic ability. To facilitate the learning of the `add/sub` modules, we extract QA pairs related to addition and subtraction from the original DROP dataset to construct a new dataset for training and evaluation.

Experimental results show that our methods significantly enhance NMNs' numerical reasoning capability. On a subset of DROP, our methods improve F1 score by 17.7% absolute points, and on ADD-SUB questions by 65.7% absolute points. Compared to NumNet (Ran et al., 2019), which is specifically designed for numerical reasoning, our method outperforms it by 2.9% absolute F1 points.

2 Background and Related Work

Semantic Parsing is a widely-adopted approach in the compositional question answering (CQA) task, which involves a number of reasoning steps. In this

Proceedings of the 29th International Conference on Computational Linguistics, pages 1502–1510
October 12–17, 2022.

Question type	Paragraph	Question	Answers
date compare	In the Morean War, the Republic of Venice besieged Sinj in October 1684 and then again March and April 1685, but both times without success. … With the help of the local population of Poljica as well as the Morlachs, the fortress of Sinj finally fell to the Venetian army on 30 September 1686. On 1 September 1687 the siege of Herceg Novi started, and ended with a Venetian victory on 30 September. …	Which happened first, the fell of Sinj or the siege of Herceg Novi?	Original NMNs: besieged Sinj $NMN_{\pm}$: Sinj finally fell Ground-truth: the fell of Sinj / Sinj finally fell
	Original NMNs' program: (span(compare-date-lt(find find)))	$NMN_{\pm}$: (span(compare-date-lt(find find)))	
addition / subtraction (2 numbers)	… In the first quarter, the Niners struck first as kicker Joe Nedney got a 47-yard field goal. In the second quarter, the Saints took the lead with QB Drew Brees completing a 5-yard and a 33-yard TD pass to WR Lance Moore. San Francisco would answer with Nedney's 49-yard field goal, yet New Orleans replied with Brees' 47-yard TD pass to WR Robert Meachem. …	How many yards was Nedney's combined field goal yards in the first and second quarters?	Original NMNs: 47 $NMN_{\pm}$: 96 Ground-truth: 96
	Original NMNs' program: (find-num (filter (find))	$NMN_{\pm}$: (addition(find-num(find))(find-num(find)))	
addition / subtraction (3 numbers)	The Greek census 2011 recorded 9,903,268 Greek citizens (91.56%), 480,824 Albanian citizens (4.44%), 75,915 Bulgarian citizens (0.7%), 46,523 Romanian citizenship (0.43%), 34,177 Pakistani citizens (0.32%), 27,400 Georgia (country) citizens (0.25%) and 247,090 people had other or unidentified citizenship (2.3%). …	How many more people were Greek citizens compared to Albanian and Bulgarian citizens combined?	Original NMNs: 9903268 $NMN_{\pm}$: 9346529 Ground-truth: 9346529
	Original NMNs' program: (find-num (find-max-num (find)))	$NMN_{\pm}$: (subtraction (find-num(find)) (addition(find-num(find))(find-num(find))))	

Figure 1: Three examples in the DROP dataset and the predictions by original NMNs and our improved model NMNs$_{\pm}$. The relevant tokens and their corresponding modules are highlighted.

approach, a *programmer* maps natural-language questions into machine-readable representations (logical forms), which are executed by an *interpreter* to yield the final answer. For instance, WNSMN (Saha et al., 2021) uses a generalized framework of dependency parsing inspired by the Stanford dependency parse tree (Chen and Manning, 2014) to parse queries into noisy heuristic programs. Neural Module Networks (Gupta et al., 2020) extend semantic parsing by making interpreter a learnable function with specified modules and executing the logical forms from the programmer in a step-wise manner.

Neural Module Networks initially is proposed to overcome the Visual Question Answering (VQA) problem (Andreas et al., 2016), where questions are often compositional. Gupta et al. (2020) employs the programmer-interpreter framework with attention (Vaswani et al., 2017) to tackle the CQA task. Specifically, the programmer parses each question into an executable program. The interpreter takes the program as input and perform various symbolic reasoning functions. The modules are defined in a differentiable way, aiming to maintain the uncertainty about each intermediate decision output and propagate them through layers. For instance, the predicted program of the first example in Figure 1 is `span(compare-date-lt(find,find))`. The interpreter would first calls the `find` module twice to find events queried by the question (e.g., 'the fell of Sinj') and outputs appropriate paragraph attention. The `compare-date-lt` module can further locate the dates (e.g., '30 September 1686') to compute their relation. By demonstrating the intermediate reasoning steps in this manner, NMNs perform interpretable problem-solving.

Numerical Reasoning is a necessary ability for models to handle the CQA task (Geva et al., 2020). Dua et al. (2019) modify the output layer of QANet (Yu et al., 2018) and propose a number-aware model NAQANet to deal with numerical questions. NumNet (Ran et al., 2019) leverage Graph Neural Network to capture relations between numbers. Similarly, QDGAT (Chen et al., 2020a) distinguish number types more precisely by adding the connection with entities and obtained better performance. Nerd (Chen et al., 2020b) search possible programs exhaustively based on answers and employed these programs as weak supervision. Another similar work (Guo et al., 2021) proposes a question-aware interpreter but uses a totally different approach to measure the alignment between the question and the context paragraph. Though these approaches can achieve the high performance on DROP dataset, it is incomprehensible for the reasoning procedure.

3 Model

In this section, we tend to illustrate our proposed methods. Basically, we will show the incorporation of questions in Section 3.1. In Section 3.2, the newly extended module: addition and subtraction will be described.

3.1 The Incorporation of Questions

Taking one module `compare-date` as a case study: it performs comparisons between two

references queried by the question. A key reasoning step inside, is the `find-date` module that obtains appropriate a date token distribution D related to each reference: `find-date`$(P) \rightarrow D$. It is worth noting that there is no interaction with the question, which could contain essential information (e.g., entities) that is useful to correctly answer the question. Therefore, we revise the `find-date` module as follows: `find-date`$(P,Q) \rightarrow D$:

$$\mathbf{S}^{date}_{i,d_j} = [\alpha\mathbf{P};(1-\alpha)\mathbf{Q}]_i \mathbf{W}_{date} \mathbf{P}_{d_j}, \quad (1)$$

$$\mathbf{A}^{date}_{i:} = softmax(\mathbf{S}^{date}_{i:}), \quad (2)$$

$$D = \sum_i [\alpha P;(1-\alpha)Q]_i \cdot \mathbf{A}^{date}_{i:} \quad (3)$$

where $\mathbf{P}$ and $\mathbf{Q}$ represent the contextualized embeddings of the paragraph and question, and $\mathbf{P}_{d_j}$ of the jth date tokens in the paragraph, $\mathbf{W}_{date}$ is a trainable parameter, P,Q are the expected attention distribution of the paragraph and the question respectively.

In Equation 1, we concatenate the paragraph embeddings $\mathbf{P}$ and question embeddings $\mathbf{Q}$ that output from a pre-trained BERT (Devlin et al., 2019) model to construct the context representation. A hyper-parameter α is used to adjust their contributions, whose value is empirically determined (Appendix A.1). The context representation is provided to compute the improved similarity matrix $\mathbf{S}_{date}$. We concatenate the paragraph and question attention inputs in the same way to calculate the final expected distribution over the date tokens D (Eq. 3). Now the interpreter is equipped with question information to make the prediction.

3.2 Addition and Subtraction Modules

In the NMNs' modelling paradigm, for addition/subtraction operations, the programmer takes as input two number distributions and produces an output number distribution over all possible result values: `add/sub`$(N_1, N_2) \rightarrow RL$. N_1 and N_2 represent the probability distributions of the 1st and 2nd operands over all numbers that are extracted from the paragraph and collected into a sorted operand list OL. The positive and negative values of these numbers are exhaustively combined in pairs, from which the possible results of addition/subtraction operations are compiled into a sorted result list RL. For each input number distribution $N_i, i = 1, 2$, a matrix $\mathbf{C}_i \in \mathbb{R}^{m \times n}$ is constructed, where m is the total number of possible results, and n is the maximum number of unique combinations. Each value $\mathbf{C}_i[j,k]$ is found by looking up the probability value in $N_i[k]$ where $OL[k]$ is the ith operand in any pair that produces result $RL[j]$. The probability that the jth number in N_i is the correct operand of the kth pair. We compute the marginalized joint probability by summing over the product of C_i as the expected distribution over result list RL. For the addition module, it is:

$$p(prediction = RL[j]) = \sum_{k_1,k_2=1}^{n} \mathbb{1}_{(OL[k_1]+OL[k_2]=RL[j])} \mathbf{C}_1[j,k_1] * \mathbf{C}_2[j,k_2]$$

For instance, assume the sorted operand list OL from a paragraph is [1, 5, 7, 11] and $N_1 = [0.1, 0.4, 0.2, 0.3]$. Different combinations are formed, e.g., (+n_1, +n_2) for addition and (+n_1, -n_2) for subtraction, and all possible results of the combinations are compiled into two result lists, one for addition and one for subtraction. For subtraction in this case, $RL = [0, 2, 4, 6, 10]$. The value of $\mathbf{C}_1[2,1]$ is 0.4, which is found from $N_1[1]$ because the result 4 can be calculated from (+5, -1); and $\mathbf{C}_1[2,3] = 0.3$ which equals to $N_1[3]$ as 4 is the result of (+11, -7) as well. $\mathbf{C}_2$ is computed in the same way to further obtain final distribution over RL.

We compose `add/sub` modules in programs to perform 3-number arithmetic. The key to our approach is to construct and distinguish appropriate C_i and RL in different reasoning steps. In the second arithmetic step, we should combine the operand list from the paragraph and the result list from the previous step to obtain a new result list RL', `add/sub`$(RL, N) \rightarrow RL'$. Due to the changes in operands and results, the modules should refer to a different $\mathbf{C}'_i \in \mathbb{R}^{m' \times n'}$ in the computation. We extend 2-number `add/sub` modules to recognize the participation of the third number by conditional statement, in order to differentiate the operand and result lists the interpreter should refer to in different steps. Taking the last example in Figure 1, the `addition` module would first compute the distribution over result list for 'Albanian and Bulgarian citizens'. The `subtraction` module can identify itself in the second step calculation and take the correct input to construct the new matrix $\mathbf{C}'_i$. The expected distribution over new result list RL' now represent the difference of 'Greek citizens' and the previous result.

Instead of introducing specific modules for multi-number arithmetic such as '3-num-add', the structure of NMNs allows us to recursively execute basic operations several times in a compositional

program. This design is in accord with the reasoning process of the CQA task, and natural for NMNs to perform complex computations.

4 Experiments

Dataset. We construct our own train/dev/test sets based on the DROP dataset (Dua et al., 2019), which requires numerical reasoning skills.

Gupta et al. (2020) extracted a subset of questions from DROP that is supported by the model's reasoning capability. This subset contains approximately 20,000/500/2,000 QA pairs for train/dev/test. To train the `add/sub` modules, we augment the NMNs' subset with more than 5,000 new questions from DROP. These questions were heuristically identified based on first n-grams and regular expressions (Appendix A.2). Statistics of this newly constructed dataset can be found in Table 1. Note that the ADD-SUB questions include both 2-/and 3-number arithmetic and **all experiments in this paper are conducted on this new dataset**. Model performance is evaluated with the same F1 and EM (Exact Match) scores as Gupta et al. (2020).

Question types	train	dev	test
Full	25,165	623	2,547
DATE-COMPARE (13.9%)	3,505	91	333
DATE-DIFFERENCE (12.2%)	3,055	75	313
NUMBER-COMPARE (12.1%)	2,642	157	632
EXTRACT-NUMBER (12.8%)	3,349	57	222
COUNT (17.3%)	4,527	73	288
EXTRACT-ARGUMENT (13.1%)	3,467	51	208
ADD-SUB (18.6%)	**4,689**	**124**	**553**
2-numbers	4,440	106	505
3-numbers	259	24	66

Table 1: Question types distribution on the expanded DROP subset used in the follow experiments.

Result. In Table 2, we list the overall performance of the original NMNs, NumNet and our proposed method NMNs$_{\pm}$.

Method	F1	EM
original NMNs (Gupta et al., 2020)	57.5	54.9
NumNet (Ran et al., 2019)	72.3	69.4
NMNs$_{\pm}$ (ours)	**75.2**	**72.6**
w/o add/sub	61.4	58.1
w/o qi	74.3	71.7

Table 2: Performance comparison between different models on **our test set**. Constrained by the page limit, case study and analysis are in Appendix A.5.

In Table 2, row "w/o add-sub" is the model variant with question attention only, and row "w/o qi" only has the `add/sub` modules only. Compared to the original NMNs, two proposed methods both improve model performance and the `add/sub` modules contributes more. Our full NMNs$_{\pm}$ model, with both components added, achieves 75.2% F1 and 72.6% EM scores, obtaining significant deltas of 17.7% absolute points compared to the original NMNs for both F1 and EM. Additionally, NMNs$_{\pm}$ outperforms NumNet by 2.9% and 3.2% absoule points in F1 and EM.

It can be unfair since the original NMNs will perform poorly on the newly added ADD-SUB questions. Therefore, we list the model performance on different question types in Table 3. Our model achieves higher scores across almost all question types comparing to the original NMNs, attesting to the effectiveness of our proposed techniques. And it turns out that adding ADD-SUB question types and more training data does not improve the results of the original DROP split. This might due to the performance degradation of the programmer after adding these new ADD-SUB programs. When comparing to NumNet, though our model fail on 2-number ADD-SUB questions, we achieve 5.4% F1 improvement on 3-number ADD-SUB questions, thus results in a comparable performance. Note that the 2-number data is nearly **18 times** the 3-number data, which shows our model or NMNs relies less on large scale datasets.

Question type	NMNs	NMNs$_{\pm}$	NumNet
DATE-COMPARE	79.2	**84.9**	72.0
DATE-DIFFERENCE	69.0	73.3	**74.1**
NUMBER-COMPARE	89.6	**90.3**	89.9
EXTRACT-NUMBER	86.4	**89.1**	85.6
COUNT	54.2	**60.2**	52.4
EXTRACT-ARGUMENT	73.4	**75.3**	66.1
ADD-SUB	0.7	66.4	**67.6**
2-numbers	0.8	67.9	**71.5**
3-numbers	0.3	**41.2**	35.8

Table 3: F1 comparison on different question types.

Additional ablation studies for the `add/sub` modules (A.3) and a qualitative analysis (A.4) can be found in the appendix.

5 Conclusion

In this work, we extend NMNs' numerical reasoning capability to 2-/and 3-number addition and subtraction, and incorporate the influence of question information to the interpreter on number

related questions. Experimental results show that our methods significantly enhance NMNs' numerical reasoning ability, with an increase of 17.7% absolute F1 points on a newly constructed DROP subset that includes arithmetic questions. Moreover, our approach also outperforms NumNet, a SOTA numerical reasoning model, by 2.9% F1 points.

Acknowledgements

This work is partially funded by the DARPA CCU program (HR001121S0024).

References

Jacob Andreas, Marcus Rohrbach, Trevor Darrell, and Dan Klein. 2016. Neural module networks. In *2016 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2016, Las Vegas, NV, USA, June 27-30, 2016*, pages 39–48. IEEE Computer Society.

Danqi Chen and Christopher Manning. 2014. A fast and accurate dependency parser using neural networks. In *Proceedings of EMNLP*, pages 740–750.

Kunlong Chen, Weidi Xu, Xingyi Cheng, Zou Xiaochuan, Yuyu Zhang, Le Song, Taifeng Wang, Yuan Qi, and Wei Chu. 2020a. Question directed graph attention network for numerical reasoning over text. In *Proceedings of EMNLP*, pages 6759–6768.

Xinyun Chen, Chen Liang, Adams Wei Yu, Denny Zhou, Dawn Song, and Quoc V. Le. 2020b. Neural symbolic reader: Scalable integration of distributed and symbolic representations for reading comprehension. In *Proceedings of ICLR*.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. Bert: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of NAACL-HLT*, pages 4171–4186.

Dheeru Dua, Yizhong Wang, Pradeep Dasigi, Gabriel Stanovsky, Sameer Singh, and Matt Gardner. 2019. Drop: A reading comprehension benchmark requiring discrete reasoning over paragraphs. In *Proceedings of NAACL-HLT*, pages 2368–2378.

Mor Geva, Ankit Gupta, and Jonathan Berant. 2020. Injecting numerical reasoning skills into language models. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, ACL 2020, Online, July 5-10, 2020*, pages 946–958. Association for Computational Linguistics.

Xiaoyu Guo, Yuan-Fang Li, and Gholamreza Haffari. 2021. Improving numerical reasoning skills in the modular approach for complex question answering on text. In *Findings of the Association for Computational Linguistics: EMNLP 2021, Virtual Event / Punta Cana, Dominican Republic, 16-20 November, 2021*, pages 2713–2718. Association for Computational Linguistics.

Nitish Gupta, Kevin Lin, Dan Roth, Sameer Singh, and Matt Gardner. 2020. Neural module networks for reasoning over text. In *Proceedings of ICLR*.

Qiu Ran, Yankai Lin, Peng Li, Jie Zhou, and Zhiyuan Liu. 2019. Numnet: Machine reading comprehension with numerical reasoning. In *Proceedings of EMNLP-IJCNLP 2019*, pages 2474–2484.

Amrita Saha, Shafiq R. Joty, and Steven C. H. Hoi. 2021. Weakly supervised neuro-symbolic module networks for numerical reasoning. *CoRR*, abs/2101.11802.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Proccedings of NIPS*, pages 5998–6008.

Adams Wei Yu, David Dohan, Minh-Thang Luong, Rui Zhao, Kai Chen, Mohammad Norouzi, and Quoc V. Le. 2018. Qanet: Combining local convolution with global self-attention for reading comprehension. In *Proceedings of ICLR*.

A Appendix

A.1 Hyper-parameter setting for compare-date modules

As mentioned above, we use a hyper-parameter α to represent question's and paragraph's weights for the combined context representation. We determine the final coefficient through a series of control parameter comparison experiments: use the same data to train and validate the model with different α. The model achieves the best performance (84.9 F1) for DATE-COMPARE questions when α was set to 0.4 (40% for paragraph attention and 60% for question attention), which increase 5.7 absolute points compared to the original NMNs model. The experiment verifies the importance of question information in the numerical reasoning process.

A.2 Data extraction

In this research, we expand the DROP subset for original NMNs to cover addition and subtraction questions. Subtraction questions can be easily targeted by their first n-gram, such as 'how many more', 'how many yards difference'. For three number subtraction, we need to further specified the format by regular expression, such as 'how many more EVENT-A and EVENT-B than EVENT-C?' or 'how many more EVENT-A compared to EVENT-B and EVENT-C?'. For addition, it is hard to identify how many numbers should participate in the calculation from some of the questions (e.g. 'how many total yards did Roethlisberger get in the game?'). Therefore, we use regular expression to distinguish two or three numbers addition and follow the patterns such as 'how many total...', 'how many ... combined'.

A.3 Addition and subtraction modules training

To discuss the contribution of individual `addition` and `subtraction` module for NMNs, we conduct an ablation experiment by training and testing the model on different datasets as shown in Table 4. The five rows represent the model trained on various datasets: addition questions only, subtraction questions only, addition and the original NMNs subset, subtraction and original NMNs subset and our full subset. The columns indicate the model performance results when they test on addition/subtraction questions only and the full DROP subset. As can be seen from the result, the model with subtraction ability only perform greater than with addition ability only.

Datasets	addsub dataset		full dataset	
	F1	EM	F1	EM
add	41.2	41.2	46.0	43.8
sub	45.7	45.7	51.3	49.2
add+origin	51.5	51.5	69.2	64.2
sub+origin	55.1	55.1	72.6	69.8
add+sub+origin	66.4	66.4	74.3	71.7

Table 4: Ablation experiment result for addition and subtraction modules

A.4 Qualitative analysis

Figure 2 shows some incorrect prediction cases from the original NMNs and the answer from our improved model NMNs$_{\pm}$. From the examples, we can clearly identify how the proposed techniques improve the numerical reasoning process:

- In the first example, the original NMNs match wrong tokens 'dissolved the Constituent Assembly' given the question 'Which event happened first, the Constituent Assembly being elected, or the elimination of hierarchy in the army?', thus located a wrong date 'January 1918'. After enhancing the interpreter's awareness of the question, NMNs$_{\pm}$ can precisely target the spans 'a Constituent Assembly was elected' in the paragraph and further provide the correct prediction.
- The following two examples are wrongly answered by the original NMNs because of incorrect program predictions. The second question was initially categorized into a COUNT question which called the `count` module to calculate the number of attended paragraph spans. The same situation occurs in the third question, because the original NMNs lack the modules that can correctly expresses the reasoning behind the question. The prediction results prove that our NMNs$_{\pm}$ model handle simple arithmetic operations such as addition and subtraction which meets the task requirement.

A.5 Prediction analysis

The wrong prediction cases study for the original NMNs over DROP is the main motivation of our proposed methods. We conclude the error factors of five numerical question types in detail: DATE-COMPARE, COUNT, DATE-DIFFERENCE, NUMBER-COMPARE and EXTRACT-NUMBER.

Question type	Paragraph	Question	Answers
Date compare	On 12 November 1917, a Constituent Assembly was elected. In these elections, 26 mandatory delegates were proposed by the Bolshevik Central Committee and 58 were proposed by the Socialist Revolutionaries. … The Bolsheviks dissolved the Constituent Assembly in January 1918, when it came into conflict with the Soviets. On 16 December 1917, the government ventured to eliminate hierarchy in the army, removing all titles, ranks, and uniform decorations.	Which event happened first, the Constituent Assembly being elected, or the elimination of hierarchy in the army?	Original NMNs: eliminate hierarchy in the army $NMN_{\pm}$: the Constituent Assembly being elected Ground-truth: Constituent Assembly being elected
	Original NMNs' program: (span(compare-date-gt(find find))) $NMN_{\pm}$: (span(compare-date-gt(find find)))		
Count	… The Ravens would later add 16 more points on three Billy Cundiff field goals and a fumble forced by Ray Lewis and recovered by Haloti Ngata and then run into the end zone (Cundiff also missed two 51-yard field goals). …	How many field goals did Billy Cundiff kick (both successful and unsuccessful)?	Original NMNs: 2 $NMN_{\pm}$: 5 Ground-truth: 5
	Original NMNs' program: (count (filter (find)) $NMN_{\pm}$: (addition(find-num(find))(find-num(find)))		
Find number	… In the first quarter, the Niners struck first as kicker Joe Nedney got a 47-yard field goal. In the second quarter, the Saints took the lead with QB Drew Brees completing a 5-yard and a 33-yard TD pass to WR Lance Moore. San Francisco would answer with Nedney's 49-yard field goal, yet New Orleans replied with Brees' 47-yard TD pass to WR Robert Meachem. …	How many yards was Nedney's combined field goal yards in the first and second quarters?	Original NMNs: 47 $NMN_{\pm}$: 96 Ground-truth: 96
	Original NMNs' program: (find-num (filter (find)) $NMN_{\pm}$: (addition(find-num(find))(find-num(find)))		

Figure 2: Qualitative analysis. The highlighted spans are corresponding to the modules in the program for each question.

Deficiency	**Paragraph**	**Question**	**Interpretation**
Match incorrect start and end of the span as answer (53.4%)	… In **1438** Svitrigaila withdrew to Moldavia. The reign of Sigismund Kestutaitis was brief — he was assassinated in 1440. Svitrigaila returned from exile in **1442** and ruled Lutsk until his death a decade later. …	What event happened later, Svitrigaila withdrew to Moldavia or Svitrigaila returned from exile?	the final output is not exactly the same with the ground-truth, but F1 score is not 0
Match wrong dates for the entities (20.8%)	The Siege of Vienna in **1529** was the first attempt by the Ottoman Empire, ... Thereafter, 150 years of bitter military tension and reciprocal attacks ensued, culminating in the Battle of Vienna of **1683**, which marked the start of the 15-year-long Great Turkish War. The inability of the Ottomans to capture Vienna in 1529 turned the tide against almost a century of conquest throughout eastern and central Europe. …	Which happened first, the Siege of Vienna or the Great Turkish War?	reasons include dates are too close, or match to the date for a similar entity
Date comparison based on natural language inference or phrases instead of symbolic comparison (13.4%)	The Russians advance into the Polish-Lithuanian Commonwealth **led to** the kingdom of Sweden invading Poland in 1655 under King Charles X. …	What happened first, the Russian advance into Poland-Lithuania or the Swedish invasion of Poland?	the module cannot understand the relation expressed by words or phrases, such as 'lead to', 'before'.
Wrong comparison result due to incomplete date format (12.4%)	… An agricultural worker had been shot during a local strike on **9 August 1917** at Ypaja and a Civil Guard member was killed in a local political crisis at Malmi on **24 September**. …	Which happened first, the shooting of a worker during a strike, or the killing of a Civil Guard member in Malmi?	the dates with omitted year are represented in format (DD, MM, -1), which lead to wrong comparison result. We proposed a forward matching mechanism in the data pre-processing step to tackle this problem.

Figure 3: Root causes for the wrong prediction in DATE-COMPARE questions. The related events mentioned in the question are highlighted in blue and red, and their relevant dates are in the same color with underline.

Deficiency	Paragraph	Question	Interpretation
Error caused by other modules (25%) • Find module cannot match the correct spans in the paragraph • Filter module cannot identify the target in the passage	… In the third quarter, Cincinnati continued to struggle as Patriots RB Sammy Morris getting a 7-yard TD run. The Bengals' only response would be kicker Shayne Graham nailing a 40-yard field goal. In the fourth quarter, New England increased its lead with Gostkowski kicking a 36-yard field goal. Cincinnati's final response was Graham kicking a 48-yard field goal. …	How many field goals were there in the second half?	the filter module is uncapable to identify 'the second half' means the third and fourth quarters, then it wrongly target on 'the second quarter'
Require numerical operation such as addition, subtraction or conditional decision (17.5%)	… In the first quarter, the Redskins drew first blood when the kicker Shaun Suisham nailed a 49-yard field goal for the only score of the quarter. … The Giants would get on the board with kicker Lawrence Tynes getting a 35-yard field goal. …	How many more field goals did Shaun Suisham make compared to Lawrence Tynes?	the subtraction operation is required in reasoning process, instead of direct counting
Wrong program prediction (16.5%)	… The Cowboys would only kick field goals in this game, as Dan Bailey was 4 for 4 on field goals. Dallas lead 12-10 with under 2 minutes to go. …	How many field goals did the Cowboys make?	predicted program: (count(filter(find()))); correct program: (find-num(find()))
Mistakes in counting the key tokens in the question (15%) • unable to count 0 when the tokens not exist in the paragraph • should not predict the answer based on counting tokens	… In the first quarter, Denver trailed early as QB Josh McCown completed a 15-yard TD pass to WR Tim Dwight. The Broncos replied with RB Travis Henry getting a 4-yard TD run. …	How many field goals did Janikowski kick in the first quarter?	'field goal' or 'Janikowski' is not mentioned in the filtered paragraph, but the module didn't predict 0 as answer
Unable to count multiple times in one span or identify phrases (15%)	… In the first quarter, the Bengals opened the scoring with two Shayne Graham field goals. ... In windy conditions, Phil Dawson hit a pair of 29-yard field goals, and Chris Jennings had a 10-yard touchdown run to put the Browns up 13. …	How many field goals were kicked in the game?	The count module only count once for 'two field goals' or 'a pair of field goals'
Cannot identify true or false in counting (7.5%)	… Houston had a chance to tie the game with one second left in regulation, but Brown's 42-yard field goal attempt **sailed wide left.** …	How many field goals were scored during the game?	The count module should not count once for 'the field goal sailed wide left'
Other various error due to wrong mean value calculation (3.5%)			

Figure 4: Root causes for the wrong prediction in COUNT questions. The inputs to the find module and their targets in the paragraph are highlighted in red. The blue spans are related to the filter module.

Deficiency	Paragraph	Question	Interpretation
Unable to identify the tokens as date	... It was waged from 1593 to 1606 but in Europe it is sometimes called the Fifteen Years War, reckoning from the 1591-92 Turkish campaign that captured Bihac. …	How many years did the Turkish campaign that captured Bihac last?	the date parser cannot interpret 1591-92 into two dates
Computation error, should be the date difference plus one	After twelve years of peace following the Indian Wars of 1622-1632, another Anglo-Powhatan War began on March 18, 1644, as a last effort by the remnants of the Powhatan Confederacy, still under Opechancanough, to dislodge the English settlers of the Virginia Colony. …	How many years did the Indian Wars last?	the prediction is calculated by the year difference between 1622 and 1632, but the answer is 11 years
Cannot understand the relation between dates through semantic expression	… His wife died in 1583, and on 7 November 1590 he was married in the same church to Jaél de Peigne, a French Hugenot. She was naturalised in June 1601. After Henry's death she remarried on 19 April 1617 George Downham, Bishop of Derry, and died c.1632. …	How many years was it after her first marriage did Jael de Peigne marry for the second time?	the time-diff module cannot identify the second one or the last one of the related dates
Wrong program prediction	… The Yongle Emperor began the preparation for relocating the imperial capital to Beiping in 1403, a process that lasted throughout his entire reign. … In 1420, the reconstruction of Beiping City was completed, and the Ming Dynasty officially relocated the imperial capital to Beiping and renamed the city to Beijing. …	How many years did the relocation of the capital to Beiping take?	predicted program: (count (find)) correct program: (time-diff (find, find))
Match wrong dates for the entities	… The remains of about 70 men, women, and adolescents were found in the path of a planned expressway near Lima in 2007. Forensic evidence suggests that the natives were killed by European weapons, probably during the uprising in 1536. …	How many years after the uprising in 1536 where many natives were killed did Archaeologists find their remains near Lima?	the find date module did not target on the correct date 1536 for the first event

Figure 5: Root causes for the wrong prediction in date-difference questions. The related events are highlighted in blue, which is the input of the find module. The dates grounding correctly predicted in the compare-date modules are highlighted in red color. The answer predicted by NMNs should be the difference of these two dates.

Deficiency	Paragraph	Question	Interpretation
Did not match the correct number for entities when there were multiple existence	… On 21 June 1916, two troops of the 10th, totaling 92 troopers, attacked Mexican Federal Army troops in an engagement in the Battle of Carrizal, Chihuahua. 12 US troops were killed and 23 taken prisoner; 45 Federales were casualties, including the Mexican general Gomez. …	Which group experienced more casualties, US troops or Federales?	'US troops' was mentioned multiple times in the paragraph, the find-num module wrongly targets on another number
Need multiple answers to the question	… In the city, the age distribution of the population shows 21.8% under the age of 18, 13.1% from 18 to 24, 31.7% from 25 to 44, 20.1% from 45 to 64, and 13.2% who were 65 years of age or older. The median age was 34 years. For every 100 females, there were 87.1 males. For every 100 females age 18 and over, there were 83.5 males. …	Which age groups had a bigger population than those 65 years of age or older but lower than those 25 to 44?	there are two age groups conform to the condition: '45 to 64', 'under the age of 18'
Interpret the comparison adjective in the question wrongly	… In terms of ancestry, 28.1% were German, 19.8% were Irish, 12.2% were English, 9.9% were Italian, 6.8% were Polish, and 6.2% were American.	Which group had the least ancestry, Irish or Polish?	The compare-num module cannot identify the meaning of 'the least'
Wrong program prediction	… The racial makeup of the county was 81.2% white, 12.7% black or African American, 2.4% Asian, 0.3% American Indian, 0.1% Pacific islander, 0.9% from other races, and 2.5% from two or more races. Those of Hispanic or Latino origin made up 3.5% of the population. …	Which group made up more of the population than the Asians but less than black or African American?	the predicted program is (span (compare-num-gt (find, find))), it did not consider the less than condition in the question
Not matching the correct start and end of a span to answer the question			

Figure 6: Root causes for the wrong prediction in number-compare questions. Similar to figure 1, the input of the find module is highlighted in blue and red, and their related numbers are underlined. The paragraph span predicted as the answer is the one associated to a smaller/larger-valued number according to the questions asking.

Deficiency	Paragraph	Question	Interpretation
Find module return an inaccurate result for find-num module	… In the second half the Ravens scored 4 consecutive touchdowns. First a Le'Ron McClain 3-yard run. Then 2 by Willis McGahee: first an 8-yard run, then a 19-yard run. In the fourth quarter, the Ravens capped off their huge victory when Troy Smith ran in a TD from 15 yards …	How many yards was the shortest touchdown run in the game?	there is no key words around the ground-truth answer 2, so the module predict '3-yard run' as answer
Filter module fix to the wrong spans (e.g. based on natural language inference)	… In the first quarter, Tennessee drew first blood as QB Vince Young completed a 16-yard TD pass to WR Roydell Williams for the only score of the period. In the second quarter, the Chiefs tied the game with QB Brodie Croyle completing a 10-yard TD pass to WR Samie Parker. Afterwards, the Titans responded with kicker Rob Bironas managing to get a 37-yard field goal. Kansas City would take the lead prior to halftime with Croyle completing a 9-yard TD pass to FB Kris Wilson …	How many yards was the shortest touchdown of the first half?	the filter module is uncapable to identify 'the first half' means the first and second quarters, then it wrongly target on 'the first quarter'
Require arithmetic operation	… In the first quarter, the Niners struck first as kicker Joe Nedney got a **47-yard field goal**. In the second quarter, the Saints took the lead with QB Drew Brees completing a 5-yard and a 33-yard TD pass to WR Lance Moore. San Francisco would answer with Nedney's **49-yard field goal**, yet New Orleans replied with Brees' 47-yard TD pass to WR Robert Meachem …	How many yards was Nedney's combined field goal yards in the first half?	predicted program: (find-num (filter (find))) correct program: (addition(find-num(filter (find))) (find-num(filter (find))))

Figure 7: Root causes for the wrong prediction in extract-number questions. The inputs to the find module and their targets in the paragraph are highlighted in red. The blue spans are related to the filter module. The find-num module finally extracts the number associated with this paragraph attention as the answer.

An Augmented Benchmark Dataset for Geometric Question Answering through Dual Parallel Text Encoding

Jie Cao
School of Computer Science
South China Normal University
jiecao@m.scnu.edu.cn

Jing Xiao*
School of Computer Science
South China Normal University
xiaojing@scnu.edu.cn

Abstract

Automatic math problem solving has attracted much attention of NLP researchers recently. However, most of the works focus on the solving of Math Word Problems (MWPs). In this paper, we study on the Geometric Problem Solving based on neural networks. Solving geometric problems requires the integration of text and diagram information as well as the knowledge of the relevant theorems. The lack of high-quality datasets and efficient neural geometric solvers impedes the development of automatic geometric problems solving. Based on GeoQA, we newly annotate 2,518 geometric problems with richer types and greater difficulty to form an augmented benchmark dataset **GeoQA+**[1], containing 6,027 problems in training set and 7,528 totally. We further perform data augmentation method to expand the training set to 12,054. Besides, we design a **D**ual **P**arallel text **E**ncoder (**DPE**) to efficiently encode long and medium-length problem text. The experimental results validate the effectiveness of GeoQA+ and DPE module, and the accuracy of automatic geometric problem solving is improved to 66.09%.

1 Introduction

In recent years, with the continuous development of deep learning technology in NLP, more and more math problem solvers have been developed. However, most of these works focus on solving arithmetic and algebra problems (Xie and Sun, 2019; Lin et al., 2021; Wu et al., 2020). There are few systems for geometric problem solving, especially those based on the method of the neural networks. The solving of geometric problems requires a combination of text and diagram information, and therefore the study of it also helps to promote the development of cross-modal problem-solving.

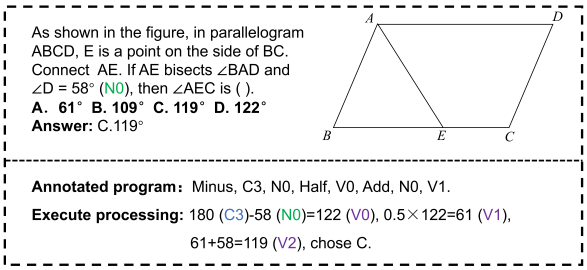

Figure 1: A typical geometry problem in GeoQA+ dataset and its annotating and solving process.

As shown in Figure 1, a typical geometry problem mainly consists of textual descriptions and geometric diagrams. There are three steps to solve this problem. First, the text and diagram information are encoded separately. Second, the solver needs to understand the semantics of the text and diagram information simultaneously. Third, in order to solve the problem, we may need to combine the information of Text-Diagram with relevant theorem knowledge. For example, the problem in Figure 1 use the theorem of complementary adjacent angles of a parallelogram. Though some previous methods attempt to solve geometric problems automatically, the performance of their solving system is far away from satisfactory (Seo et al., 2014, 2015; Sachan and Xing, 2017). They highly relied on limited handcraft rules and were only validated on small-scale datasets, making it hard to generalize to more complex and real-world cases(Chen et al., 2021). In this case, we mainly focus on building an efficient solving system based on neural networks.

To resolve the mentioned issues, Chen et al. (2021) proposed a geometric problems dataset GeoQA which contains 5,010 geometric problems and the first neural network-based geometric problems solving system NGS. However, we believe that there are some limits to this work. First, we think that the problem type in GeoQA is not rich enough, and it only contains angle and length

*Corresponding Author

[1]The source code and benchmark of this paper are available at: https://github.com/SCNU203/GeoQA-Plus

Proceedings of the 29th International Conference on Computational Linguistics, pages 1511–1520
October 12–17, 2022.

problems as well as a very small number of other types. Second, we think that the problems in GeoQA are not difficult enough, and the average solving step of the problems in GeoQA is only 1.96. Third, the geometric problems solver NGS can not effectively solve the problems with long text for the lack of text feature extraction capability.

Inspired by the exiting works (Chen et al., 2021; Seo et al., 2014, 2015), to refresh the research on geometric problem solving and further promote the development of cross-modal numerical reasoning, we newly annotate 2,518 geometric problems containing 636 area-type problems that are not included in GeoQA. The problems we collect are more difficult. The average solving step of our problems is 2.61, which compares to 1.96 of GeoQA. We add our new dataset to the training set of GeoQA to build a new dataset named **GeoQA+** and it contains 7,528 problems in total and 6,027 for training. To the best of our knowledge, GeoQA+ is the largest benchmark dataset for geometry problem solving at present and it improves the overall difficulty and diversity of the original dataset. We further perform data augmentation method on GeoQA+, which expand the data size to 12,054 to obtain more diverse data. As for the model, we design a **D**ual **P**arallel **E**ncoder **DPE** that consists of RoBERTa (Liu et al., 2019) and a Bi-LSTM (Hochreiter and Schmidhuber, 1997) to address the limit of NGS. Our **DPE** module encodes long and medium-length problem text effectively, and we name this new geometric problems solver as **DPE-NGS**. We conduct a series of experiments and the experimental results indicate that the GeoQA+ dataset and our DPE-NGS model show the superiority over the state-of-the-art results.

In summary, our contributions are three-fold:

- To expand GeoQA, we newly annotate 2,518 geometric problems which are more difficult to solve than GeoQA and has more problem types to build a new dataset name GeoQA+, the largest dataset for geometric problem solving at present. In addition, we also perform data augmentation work on GeoQA+ to obtain more diverse data.

- To alleviate the limit of NGS, we design a Dual Parallel Encoder(**DPE**) and propose **DPE-NGS** to effectively solve the geometric problems with long and medium-length text. Experimental results show that our model achieves better accuracy.

- We study the text encoding work of geometric problems. We fine-tune the Pre-training model using a sufficient amount of data for the first time and achieve excellent model performance.

2 Related Work

Geometric Problems Solving Having machine to solve geometric problems has a long history in AI (Wen-Tsun, 1986; Chou et al., 1996). Some researchers proposed methods for geometry theorem proving based on rule-based methods last century(Wenjun, 1984). Wong et al. (2007) designed the first automatic solver LIM-G for geometric problems, but this method was only based on text information to solve the problems. Subsequently, Seo et al. (2014, 2015) constructed the first automatic problems solver that combines text and diagram information with NLP methods and computer vision technology (OCR). However, this method relies too much on handcrafted rules, and it was only verified on the data set with 185 problems. To improve GeoS, Sachan and Xing (2017) replaced these handcraft constraints with geometry axiomatic knowledge in the form of horn-clause rules, but their dataset and code are not released. Lu et al. (2021) proposed Inter-GPS which achieved higher accuracy than all previous geometric problem solvers based on rule-based methods. And their dataset Geometry3k contains 3,002 geometric problems. But Inter-GPS was still designed based on the rule-based method and Geometry3k is not suitable for training neural network-based solvers because of the complex annotating work. Aiming to improve the performance and interpretability of existing models, Chen et al. (2021) proposed a geometric problems dataset GeoQA, and they proposed the first geometric problems solver based on neural networks named NGS. While the GeoQA dataset is not difficult and diverse enough, and the feature extraction ability of NGS is also not good enough. To improve the limits of existing works and promote the development of automatic geometric problem solving, we effectively expand GeoQA dataset and propose DPE-NGS model.

Multimodal Reasoning Visual question answering is a typical multimodal problem. The

solving of this kind of problem often requires the model to have a certain reasoning ability (Goyal et al., 2017; Yu et al., 2019). On this basis, some methods propose an implicit reasoning framework to jointly encode multimodal information (Perez et al., 2018; Cohen and Areni, 1991). However, geometric problem solving is more logical and deductive, and the solving process requires additional knowledge of theorems, so these visual problem answering models are not directly applicable to geometric problem solving.

Pre-training Model In NLP Pre-training models have greatly advanced the development of NLP (Song et al., 2021; Zhang et al., 2020). And it has also been applied in the automatic solving of MWPs (Liang et al., 2021). However, it has not been applied to the automatic solving of geometric problems since the lack of dataset. Our experimental results show that the introduction of Pre-training models facilitates the possibility of solving geometric problems based on our newly annotate dataset.

Text Data Augmentation Text data augmentation methods has been widely used in NLP, like EDA(Wei and Zou, 2019) and back translation(Yu et al., 2018) method. We also perform the back-translated method on GeoQA+. We first translate the original data into minor languages and then re-translate the results into the original language. Data back translation enhances the diversity of data.

3 GeoQA+ Dataset

The original GeoQA dataset contains 5,010 geometric problems, 3,509 for training, 746 for validation and 755 for test. We newly annotate 2,518 geometric problems and add them to GeoQA's training set to form a new dataset GeoQA+ which contains 6,027 geometric problems in training set and 7,528 in total.

3.1 Problem and Data Description

Problem Description. Automatic geometry problem solving is defined as solving a geometry problem with diagram and text information. Text-Diagram information are encoded by text and diagram encoder separately, then the encoded results are fused with features from both parts through the Joint Reasoning Module. The decoder module obtains the solving sequence by decoding the output from Joint Reasoning Module, then executes the sequence and gets the answer with additional knowledge of theorems. Figure 1 shows the complete problem definition, and the solving processing uses the knowledge of the properties of parallelograms.

Data Description. Based on the problem definition, we define the data description of the geometry problem, which contains problem text t, diagram d, problem choices c, knowledge points k, problem answer a, solving processing explanation e, and the annotate programs p. Therefore, a geometry problem can be represented as $T(t, d, c, k, a, e, p)$ like Figure 1.

Program Representation. We adapt a domain-specific language(Amini et al., 2019) to represent the geometric problems solving process similar to GeoQA. The program includes the operator OP, operand N, constant operand C, and process variable V. We enrich the representation of the language by synthesizing the data statistics of our newly annotate data. As shown in Table 1, the operators OP are divided into basic and arithmetic operators as well as trigonometric and theorem operators. The constant operands contain various constants such as π, 180°, and 90° that are commonly used. Note that only operators and constant operands are given in Table 1 because both of them are fixed and will not change with different problems. For example, when we solve for the length of the hypotenuse of a right triangle with two right-angled sides known, we will use the Pythagorean theorem to solve the problems, and we need to know the fixed expression of the Pythagoras operation. The operand N is derived from the operands given in the problems and the process variable V is an intermediate variable generated during the operation, both of which vary from problem to problem. The generated sequence expressions of the model show the interpretability of the solving process. We can get a general understanding of the whole problem-solving process from Figure 1.

3.2 Dataset Comparison

The existing geometry problem datasets are generally limited by the size of the data(Seo et al., 2015) and the complex annotating work (Lu et al., 2021), which are not suitable for neural network training. GeoQA is a dataset collected specifically for building a neural network-based geometry problem solver. However, the limits of the

OPR & Const	Programs
Basic	Equal, Double, Half
Arithmetic	Add, Minus, Multiply, Divide, Prescription
Trigonometric	Sin, Cos, Tan, Arc-Sin, Arc-Cos
Theorem & Formula	Pythagorean Add/Minus, Proportion, Circle Area,Circle Perimeter, Cone Area
Constant	30, 60, 90, 180, 360, 540, Π, 0.618

Table 1: An overview of 19 operations of four different types and 8 constants in the defined program set.

GeoQA dataset are the low average difficulty of problem solving and the lack of richness of problem types. Therefore, we newly annotate a dataset with 2,518 problems and add them to the training set of GeoQA to form a new dataset with 7,528 problems, 6,027 problems for training. Compared with GeoQA, our geometry problems are more difficult, and we introduce area-type problems for the dataset. For geometry problems, difficult problems often contain more geometric relationships and geometric attributes than simple problems, and we believe that learning more features of difficult samples help the model to solve difficult problems in the real world. A detailed comparison of the data statistics of the GeoQA's training set with our newly annotate data is shown in Table 2.

As shown in Table 2, our newly annotate dataset introduces 636 problems of area-type that are not available in GeoQA which enhance the data diversity of the dataset. In addition, our dataset are more difficult with 2.61 steps of average solving compare with 1.96 of GeoQA. More solving steps means the problems are more difficult to solve. Besides, our newly annotate problems also add 27 new knowledge points, and there are 77 knowledge points in GeoQA+. The knowledge points of a problem are crucial for solving the question. During the solving process, our model will first apply a Pre-trained module to predict the knowledge points of the problem which helps generating the solving sequence.

As shown in Table 3, the total number of training set in GeoQA+ is 6,027, and the average number of solving step is 2.23, which is nearly 14% higher than the original 1.96, meaning GeoQA+ is much more difficult than GeoQA. More difficult training samples facilitate the model to learn more statistics to improve the ability to solve difficult problems. We name this new training set as **Mix-train**.

3.3 Data Augmentation

We use the back-translation method in this paper to perform data augmentation on our Mix-train training set. We first translate the Mix-train training set data into French and then re-translate the results back to the original Chinese, and finally, we get a back-translated dataset with twice the amount of data, and we name the Back-translated dataset **Backtrans-train** which contains 12,054 problems.

3.4 Data Collection and Annotation

We collect our problems from online education websites. These problems are oriented in grades 6-12, containing various types of problems with corresponding knowledge points and solving explanations. We organized several graduate students to participate in annotating these problems. Each graduate student involved in the data annotation was trained to ensure that the data was annotate consistently with GeoQA. Unlike GeoQA, we allow the existence of problems with up to 8 solving steps while the authors of GeoQA limit the solution steps to 4. We believe that the introduction of difficult problems with long solving steps is beneficial to enhance the inference and generalization ability of the model.

4 Models

To improve the limit of NGS, we redesign the text encoder module. We refer to this improved geometric problems solver as **DPE-NGS**, and the overall structure of DPE-NGS is shown in Figure 2.

4.1 Dual Parallel Text Encoder

Text modeling is commonly used in NLP tasks such as sentiment analysis, topic classification, and problem systems (Li et al., 2020). In previous work, for solving geometric problems, researchers have often encoded the text by rule-based methods (Wong et al., 2007; Seo et al., 2015; Lu et al., 2021). In NGS, an LSTM(Hochreiter and Schmidhuber, 1997) was used to encode the problem text and represented the text as hidden state H in LSTM.

However, by analyzing the statistics of the problems that NGS did not get the result (**No Result** problems), we found that the average problem text length for this category is 68.55, which is much longer than the average problem text length of 52.5

	Properties	Angle	Length	Area\Others	AVG
GeoQA-train	**Number**	1939	1303	267	/
	OP-AVG	1.83	2.10	2.03	1.96
Ours	**Number**	1256	626	636	/
	OP-AVG	**2.78**	**2.27**	**2.60**	**2.61**

Table 2: Comparison of the data statistics of GeoQA-train and our newly annotate data. OP-AVG represents the average solving step of problems.

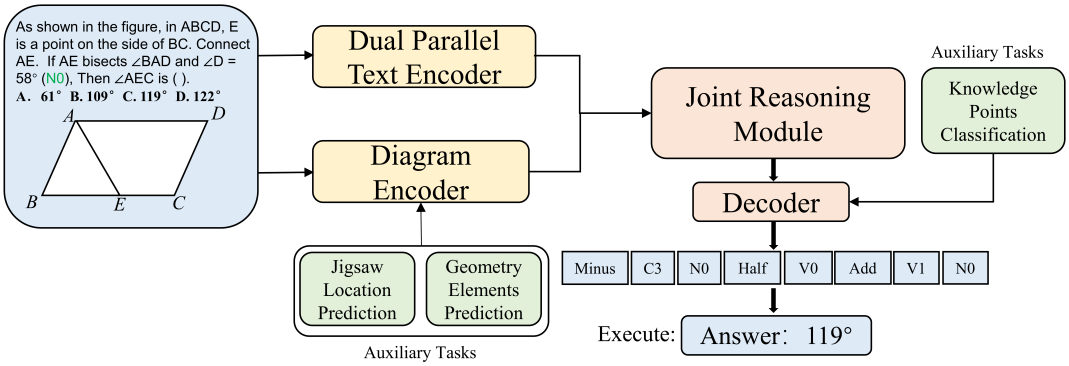

Figure 2: Our **DPE-NGS** for geometric problems solving based on Dual Parallel Text Encoder (**DPE**). The model encodes text and image information separately, and then feeds them to the Joint Reasoning Module. The decoder generates the solving sequence based on the output of Joint Reasoning Module, and the executor module finally executes the sequence and gets the answer.

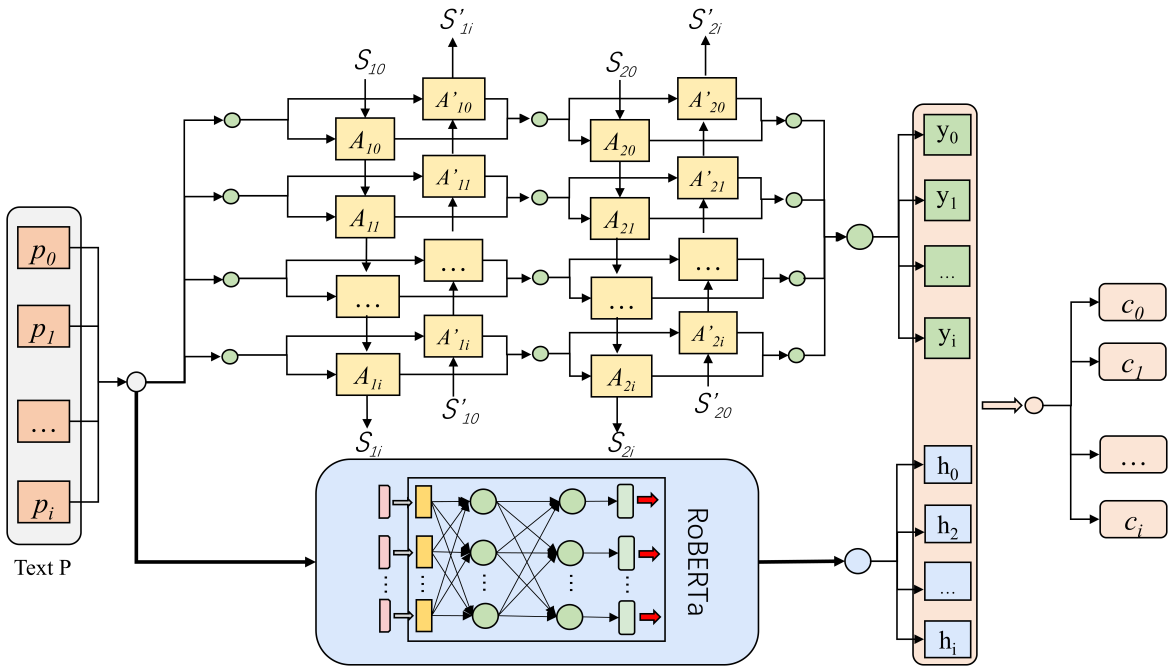

Figure 3: The architecture of our **Dual Parallel Encoder**. We use a two-layer Bi-LSTM and RoBERTa to encode the problem text separately. The encoding results are fed into a fusion layer, and we use the fused encoding information as the final text encoding result.

in GeoQA. This indicates that NGS is still lacking in feature learning for long text problems. In this case, we believe that the problem solving is related to the length of the problem text. In order to distinguish different text lengths, we regard the problems with text lengths between 30∼50 as medium-length problem, and long text problems are those with text lengths more than 50. And there are 2961

	Angle	Length	Others	AVG\Total
Number	3195	1929	903	6027
OP-AVG	2.21	2.15	2.45	2.23

Table 3: Statistics of the new training set in GeoQA+ (Mix-train).

Text Encoder	long(%)	medium-length(%)
LSTM(NGS)	50.37	69.48
RoBERTa	53.10	69.74
Bi-LSTM	52.49	70.26
DPE	**57.48**	**72.56**

Table 4: The ability of the model for different length text problems when using different modules as encoders

long text problems and 2853 medium-length problems in Mix-train.

The problem text of a geometry problem usually contains many geometric elements expressed, and there are relational dependencies between these geometric elements. As the problem shown in Figure 1, in this example, parallelogram $ABCD$ is the first geometric element that mentions in the text, and $\angle D$ appears later. But in the process of solving the problem, we need to combine the two conditions that $ABCD$ is a parallelogram (parallelogram neighbors are complementary) and $\angle D$ = 58° to derive the next condition $\angle BAD$ = 122° to solve the problem. This is a back-and-forth process in which the key information of a geometry problem interacts with each other. Therefore, we cnsider that we should encode the problem text in a bidirectional way.

Based on the analysis above, we redesign the text encoder and we first introduce the Pre-training model RoBERTa (Liu et al., 2019) as text encoder. RoBERTa is a Bert-based(Devlin et al., 2019) Pre-training model that has been widely used and has greatly advanced various works in NLP. As shown in Table 4, when using RoBERTa as text encoder alone, the model solves 53.10% of long text problems and 69.74% of medium-length problems. Moreover, we also consider encoding problem text with a Bi-LSTM(Hochreiter and Schmidhuber, 1997) with two layers alone. In this case, the model solves 52.49% of long text problems and 70.25% of medium-length text problems. As the results show that the model has a different ability to solve problems with different lengths of text when using RoBERTa or Bi-LSTM as text encoder alone. Specifically, the model solves more long text problems when RoBERTa is used as the encoder, and it performs better in solving medium-length text problems when using Bi-LSTM as encoder.

As the experimental results show, we believe that different encode module have different feature extraction capabilities for various lengths of text during automatic geometry problem solving. To fully extract the features of problems text, we consider combining RoBERTa and Bi-LSTM to form a parallel text encoder. We input the problems text into RoBERTa and Bi-LSTM to encode the text separately. We denote the encoding result of RoBERTa as H_p = [h_0;...;h_n], and we represent the encoding result of Bi-LSTM as Y_p = [y_0;..;y_n]. After obtaining the encode outputs H_p and Y_p from RoBERTa and Bi-LSTM, we combine the two sets of features by feeding H_p and Y_p into an Information Convergence layer, and obtain the fusion feature C_p = [c_0;...;c_n]. We use C_p to represent the final text encoding result:

$$C_p = [H_p, Y_p].$$

And we name this Dual Parallel Encoder module **DPE**, the structure of our encode module is shown in Figure 3.

Our model solves 57.84% long text problems and 72.56% medium-length problems with DPE as text encoder. Experimental results validate the effectiveness of our DPE encoder. The performance of models for solving problems with long or medium-length text when using different modules as text encoder is shown in Table 4.

4.2 Diagram Encoder

To get the diagram information of problems, we adapt the diagram encoder module based on ResNet (He et al., 2016) from NGS. Two auxiliary tasks (as shown in Figure 2) are applied to pre-train the diagram encoder, which significantly enhance the feature extraction capability of the diagram encoder. We formalize the feature matrix extracted by the diagram encoder as H_d.

4.3 Joint Reasoning Module

After obtaining the text feature C_p and diagram feature H_d, we feed them into the Joint Reasoning Module. In this paper, we use a common attention module named co-attention(Yu et al., 2019) with an attention mechanism for cross-modal data

fusing and reasoning. This module consists of 12 self-attention units and 6 guide-attention units. We use the Dual Parallel Encoder output C_p from the text encoder and H_d from the diagram encoder as the input of Joint Reasoning Module. This module fuses and reasons the text-diagram information and outputs F_D, which contains abundant text and diagram information. We further concatenate C_p and F_D to get F_R for decoding program.

4.4 Program Decoder

We use an LSTM(Hochreiter and Schmidhuber, 1997) with attention as the Decoder module, which generates the programs sequentially under the guidance of Reasoning module output F_R. Let $y_t(1 \leq t \leq T)$ be the target program to be generated and P_t as the next program token. In the training process, we use the negative log-likelihood function as the loss function:

$$\mathcal{L}_g(\theta) = \frac{1}{T}\sum_{t=1}^{n} \log P_t(y_t|x, y_1, ...y_{t-1}; \theta),$$

where θ is the parameter of the entire solver model except for Diagram Encoder, and x is the input of the problem text and the diagram feature extracted from the Diagram Encoder.

4.5 Program Executor

The decoder module generates N program sequences [g_1,...,g_n], and the size of N equals to beamsize (beamsize = 10). The executor module selects the first sequence that successfully solves the problem as the prediction sequence. If all the results obtained by computing sequences are not included in the problem options, then the problem will be classified as a No Result problem instead of randomly selecting an answer.

5 Experiments

5.1 Experiment Setup

We conduct experiments on GeoQA and GeoQA+, and we adapt answer accuracy as the evaluation metric. We use the GeoQA-test containing 755 geometric problems as test set. In addition, since most previous work on automatic solving of geometric problems requires additional acceptance of input from OCR, but none of these works has published their associated codes, they are not compared with our methods in this experiment.

Implementation Details. We mention three datasets above: the original training set GeoQA-train with 3,509 problems, Mix-train with 6,027 problems after mixing GeoQA-train with our newly annotate dataset, and the Backtrans-train dataset with 12,054 problems after performing data augmentation on Mix-train. To verify the effectiveness of our datasets, we train our DPE-NGS and NGS with these three datasets separately and test the accuracy of the models on GeoQA-test. In addition, we train two models with GeoQA-train and test the generalization performance on the new test set(the same size as GeoQA-test) randomly extracted from our newly annotate data. Besides, we also compare the performance of a MWPs solver Seq2Prog(Amini et al., 2019), and BERT2Prog: Seq2Prog with BERT as encoder based on GeoQA-train[2]. The learning rate of ResNet is $1e{-}5$, $1e{-}3$ for Bi-LSTM encoder, and $2e{-}5$ for RoBERTa encoder, $1e{-}5$ for the rest. The batch size is 32 and the training epoch is 100.

5.2 Experimental Result

The effectiveness of our dataset. As shown in Table 5, when two models are trained with Mix-train or Backtrans-train, both models show better performance compared to the models train with GeoQA-train. The experiment results prove the effectiveness of our newly annotate dataset. In addition, the dataset after data augmentation is also helpful for accuracy improvement. We believe that because our dataset is more difficult and richer in problem types that expand the training set and makes up for the lack of difficult problems in GeoQA, which helps the models learn more problem features and thus improve the model's performance.

The effectiveness of our model. As shown in Table 5, our DPE-NGS outperforms all models for every training set. DPE-NGS with multi-modal reasoning ability becomes the existing best-performing model (66.09%) on GeoQA-test set while train with Back-trains. We further analyze the percentage of No Result type problems generated by the models and found that DPE-NGS produces fewer No Result type problems than the NGS model as shown in Table 6. We believe it is because our DPE-NGS has better feature extraction ability for long text type problems. We also compare the accuracy of the two models for prob-

[2]Results obtained from the paper of Chen et al. (2021).

Traingsets	Model	Total(%)	Angle(%)	Length(%)	Others(%)	No Result(%)
GeoQA-train	**BERT2Prog**	50.3	63.4	33.2	38.9	/
	Seq2Prog	52.6	63.6	39.2	37.0	/
	NGS[3]	60.52	71.53	48.40	40.74	14.94
	DPE-NGS	**62.65**	**74.88**	**47.70**	**50.0**	**12.68**
Mix-train	**NGS**	61.19	72.25	47.70	46.30	12.72
	DPE-NGS	**65.96**	**75.60**	**54.42**	**51.85**	**11.90**
Backtrans-train	**NGS**	63.31	72.97	53.0	42.60	14.03
	DPE-NGS	**66.09**	**76.08**	**55.12**	**46.30**	**10.73**

[3] Results obtained from Chen's open source website: https://github.com/chen-judge/GeoQA

Table 5: Accuracy of the models on GeoQA-test using different training set.

Model	GeoQA-train(%)	Mix-train(%)	Backtrans-train(%)
NGS	14.94	12.72	14.03
DPE-NGS	**12.68**	**11.90**	**10.73**

Table 6: The percentage of **No Result** generated by the two models using different training set.

	OP=1(%)	OP=2(%)	OP=3(%)	OP=4(%)
NGS	76.70	58.42	47.10	38.33
DPE-NGS	**78.95**	**63.57**	**50.0**	**56.67**

Table 7: The accuracy of NGS and DPE-NGS for different difficulty problems using Mix-train. OP=N represents the solving steps of problems. More solving steps means the problem is more difficult.

Model	Total(%)	Angle(%)	Length(%)	Others(%)
NGS	49.14	53.85	45.90	43.07
DPE-NGS	**51.52**	**54.64**	**49.18**	**47.69**

Table 8: Accuracy of two models on our test set when train with GeoQA-train.

lems with different difficulty levels, as shown in Table 7, where our DPE-NGS outperforms NGS on all problems with different solving steps.

Generalization Performance of Models. We use the GeoQA-train dataset as training set for both models and test the generalization performance on our new test set. As shown in Table 8, since our annotate data are more difficult, neither model achieves a high accuracy, but our DPE-NGS still performs better than NGS, and our model achieves 51.52% compared to 49.14% of NGS indicating that our model shows better generalization performance.

5.3 Ablation Study

To verify the rationality of our model structure design and the validity of the text encoding method. We consider four combinations: 1) NGS, with unidirectional LSTM as text encoder; 2) NGS (RoBERTa), NGS with RoBERTa as encoder; 3) NGS (RoBERTa + LSTM) with an encoder consisting of RoBERTa and a unidirectional LSTM; 4) DPE-NGS, our improved NGS model, with a Dual Parallel Text Encoder consisting of RoBERTa and a Bi-LSTM.

As shown in Table 9, we can see that using only RoBERTa as a text encoder can not improve the performance of the model when train with GeoQA-train, but the accuracy improves considerably when train with our dataset(Mix-train:64.77%, Backtrans-train:65.03%). We believe that the geometry problem text description is far different from the common linguistic description because it contains more geometric expressions, so the RoBERTa module should be fine-tuned with a larger geometric dataset, which also reflects that our new dataset is helpful to apply the Pre-training model to geometric problems solving. In addition, we also see that the model based on RoBERTa and unidirectional LSTM is much less effective than our DPE-NGS which demonstrates the effectiveness of our DPE module structure.

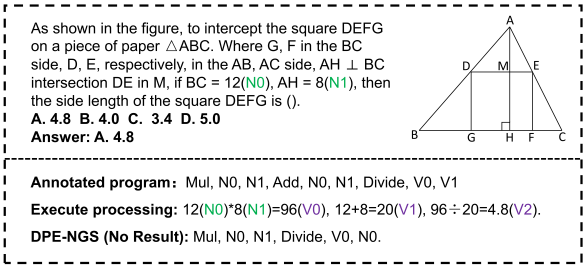

Figure 4: A typical case. A No Result type problem with a complex diagram.

	Trainsets	GeoQA-train(%)	Mix-train(%)	Backtrans-train(%)
Models	**NGS**	60.52	61.19	63.31
	NGS(RoBERTa+LSTM)	59.87	62.12	63.84
	NGS(RoBERTa)	58.28	64.77	65.03
	DPE-NGS	**62.65**	**65.96**	**66.09**

Table 9: Ablation study of different text encoder architecture designs. The content in parentheses indicates the encoder components that the model used.

5.4 Case Analysis

In our best experiment, there are still 10.73% problems for our model that can not get the answer. As shown in Figure 4, it's a typical problem in the No Result category. The diagram of this problem contains nine vertices that can form more than ten line segments and numerous geometric elements. We believe that the diagram is too complex for our Diagram Encoder to extract useful features from it. And this further leads to our inability to select useful diagram information for Joint and Reasoning work with text information, which ultimately affects the model's understanding of the whole problem scenario.

6 Conclusion

In this work, we newly annotate 2,518 geometric problems which are more difficult and with richer problem types to expand GeoQA and form a new benchmark dataset GeoQA+, the largest geometric problem dataset at present. Moreover, we propose a new text-encode method(DPE) to improve the limits of NGS. The experimental results show that both GeoQA+ and DPE-NGS have contributed to the accuracy improvement, and we have improved the baseline accuracy in automatic geometry problem solving from 60.7% to 66.09%. In the future, we will focus on the understanding of problem diagram by enhancing the ability of diagram features extraction as well as the representation of diagram information.

Acknowledgements

This paper is supported by the National Natural Science Foundation of China No. 62177015.

References

Aida Amini, Saadia Gabriel, Shanchuan Lin, Rik Koncel-Kedziorski, Yejin Choi, and Hannaneh Hajishirzi. 2019. MathQA: Towards interpretable math word problem solving with operation-based formalisms. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 2357–2367, Minneapolis, Minnesota. Association for Computational Linguistics.

Jiaqi Chen, Jianheng Tang, Jinghui Qin, Xiaodan Liang, Lingbo Liu, Eric Xing, and Liang Lin. 2021. GeoQA: A geometric question answering benchmark towards multimodal numerical reasoning. In *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*, pages 513–523, Online. Association for Computational Linguistics.

Shang-Ching Chou, Xiao-Shan Gao, and Jing-Zhong Zhang. 1996. Automated generation of readable proofs with geometric invariants. *Journal of Automated Reasoning*, 17(3):325–347.

J. B. Cohen and C. S. Areni. 1991. *Affect and Consumer Behavior*. Handbook of Consumer Behavior.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, Minneapolis, Minnesota. Association for Computational Linguistics.

Yash Goyal, Tejas Khot, Douglas Summers-Stay, Dhruv Batra, and Devi Parikh. 2017. Making the v in vqa matter: Elevating the role of image understanding in visual question answering. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 6904–6913.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. 2016. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778.

Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long short-term memory. *Neural computation*, 9(8):1735–1780.

Qian Li, Hao Peng, Jianxin Li, Congying Xia, Renyu Yang, Lichao Sun, Philip S Yu, and Lifang He. 2020. A survey on text classification: From shallow to deep learning. *arXiv preprint arXiv:2008.00364*.

Zhenwen Liang, Jipeng Zhang, Jie Shao, and Xiangliang Zhang. 2021. Mwp-bert: A strong baseline for math word problems. *arXiv preprint arXiv:2107.13435*.

Xin Lin, Zhenya Huang, Hongke Zhao, Enhong Chen, Qi Liu, Hao Wang, and Shijin Wang. 2021. Hms: A hierarchical solver with dependency-enhanced understanding for math word problem. In *Thirty-Fifth AAAI Conference on Artificial 2021*, pages 4232–4240.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Roberta: A robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692*.

Pan Lu, Ran Gong, Shibiao Jiang, Liang Qiu, Siyuan Huang, Xiaodan Liang, and Song-Chun Zhu. 2021. Inter-GPS: Interpretable geometry problem solving with formal language and symbolic reasoning. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 6774–6786, Online. Association for Computational Linguistics.

Ethan Perez, Florian Strub, Harm De Vries, Vincent Dumoulin, and Aaron Courville. 2018. Film: Visual reasoning with a general conditioning layer. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 32.

Mrinmaya Sachan and Eric Xing. 2017. Learning to solve geometry problems from natural language demonstrations in textbooks. In *Proceedings of the 6th Joint Conference on Lexical and Computational Semantics (* SEM 2017)*, pages 251–261.

Min Joon Seo, Hannaneh Hajishirzi, Ali Farhadi, and Oren Etzioni. 2014. Diagram understanding in geometry questions. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 28.

Minjoon Seo, Hannaneh Hajishirzi, Ali Farhadi, Oren Etzioni, and Clint Malcolm. 2015. Solving geometry problems: Combining text and diagram interpretation. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 1466–1476, Lisbon, Portugal. Association for Computational Linguistics.

Haoyu Song, Yan Wang, Kaiyan Zhang, Wei-Nan Zhang, and Ting Liu. 2021. BoB: BERT over BERT for training persona-based dialogue models from limited personalized data. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 167–177, Online. Association for Computational Linguistics.

Jason Wei and Kai Zou. 2019. EDA: Easy data augmentation techniques for boosting performance on text classification tasks. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 6382–6388, Hong Kong, China. Association for Computational Linguistics.

Wu Wen-Tsun. 1986. Basic principles of mechanical theorem proving in elementary geometries. *Journal of automated Reasoning*, 2(3):221–252.

Wu Wenjun. 1984. Basic principles of mechanical theorem proving in elementary geometries. *Journal of Systems Science and Mathematical Sciences*, 4(3):207.

Wing-Kwong Wong, Sheng-Cheng Hsu, Shih-Hung Wu, Cheng-Wei Lee, and Wen-Lian Hsu. 2007. Lim-g: Learner-initiating instruction model based on cognitive knowledge for geometry word problem comprehension. *Computers & Education*, 48(4):582–601.

Qinzhuo Wu, Qi Zhang, Jinlan Fu, and Xuan-Jing Huang. 2020. A knowledge-aware sequence-to-tree network for math word problem solving. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 7137–7146.

Zhipeng Xie and Shichao Sun. 2019. A goal-driven tree-structured neural model for math word problems. In *Proceedings of the Twenty-Eighth International Joint Conference on Artificial Intelligence, IJCAI-19*, pages 5299–5305. International Joint Conferences on Artificial Intelligence Organization.

Adams Wei Yu, David Dohan, Quoc Le, Thang Luong, Rui Zhao, and Kai Chen. 2018. Fast and accurate reading comprehension by combining self-attention and convolution. In *International Conference on Learning Representations*.

Zhou Yu, Jun Yu, Yuhao Cui, Dacheng Tao, and Qi Tian. 2019. Deep modular co-attention networks for visual question answering. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 6281–6290.

Shaohua Zhang, Haoran Huang, Jicong Liu, and Hang Li. 2020. Spelling error correction with soft-masked BERT. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 882–890, Online. Association for Computational Linguistics.

Competence-based Question Generation

Jingxuan Tu, Kyeongmin Rim, James Pustejovsky
Department of Computer Science
Brandeis University
{jxtu,krim,jamesp}@brandeis.edu

Abstract

Models of natural language understanding often rely on question answering and logical inference benchmark challenges to evaluate the performance of a system. While informative, such task-oriented evaluations do not assess the broader semantic abilities that humans have as part of their linguistic *competence* when speaking and interpreting language. We define competence-based (CB) question generation, and focus on queries over lexical semantic knowledge involving implicit argument and subevent structure of verbs. We present a method to generate such questions and a dataset of English cooking recipes we use for implementing the generation method. Our primary experiment shows that even large pretrained language models perform poorly on CB questions until they are provided with additional contextualized semantic information. The data and the source code is available at: https://github.com/brandeis-llc/CompQG.

1 Introduction

Developing an Artificial Intelligence (AI) system with inferencing and reasoning capabilities that enables itself to communicate with intellectual human users has been a holy grail of the research community. For example, there has been considerable effort put on large challenges in question answering formats to measure such inference and communication capabilities of neural end-to-end systems (Ribeiro et al., 2020; Prabhumoye et al., 2020; Rogers et al., 2021; Minaee et al., 2021). However, we argue that a broader effort needs to be put on measurements more focused on linguistic *competencies*, and not just on extractive comprehension skills or "challenge checklisting". Out argument is in line with some moves in this direction (Johnson et al., 2017), but there is still no generally accepted distinction in current natural language processing (NLP) between challenge-based tasks and competence-based (CB) performance (Bentivogli et al., 2017). Analogous to human cognitive competencies, there is both a methodological and modeling advantage to focusing a system's performance on competence-based learning rather than a narrowly defined task or challenge.

The term *competence-based (CB)* has been applied to a number of different concepts, from linguistics (Chomsky, 1965) to both the science of learning and educational communities (Bechtel et al., 1999; Voorhees, 2001; Chyung et al., 2006; Hsiao et al., 2020; Platanios et al., 2019). The common core to both is a concept capturing a coherent set of abilities that an individual has in a specific domain (Doignon and Falmagne, 1985; Heller et al., 2013). More recently, we proposed a new multimodal question answering task R2VQ that focuses on querying competence-based knowledge from cooking recipe texts and videos (Tu et al., 2022). For the text part, we created a dataset with rich annotations of hidden arguments and coreference that focuses on lexical competence as deployed in both single and multiple sentence composition (Pustejovsky, 1995; Marconi, 1997; Geeraerts, 2009).

As a natural complement and extend to the R2VQ task (Tu et al., 2022), here we define the task *competence-based question generation* that aims to generate CB questions that require lexical competence. The lexical competence involves understanding the hidden arguments (due to syntactic ellipsis or semantic defaulting or shadowing) given the event context; and understanding the dynamics of events and objects change in the text. This competence requires non-extractive Question Answering (QA) capabilities of some sort. Thus we define a *competence-based* question as one that queries any aspects of lexical competence mentioned above.

In the rest of the paper, we first review recent related work (Sec. 2), and give more detailed definitions of the CB question generation task (Sec. 3). We then present a dataset of CB knowledge and how we collected and annotated it (Sec. 4). Section 5 provides details of generation of CB questions on

Proceedings of the 29th International Conference on Computational Linguistics, pages 1521–1533
October 12–17, 2022.

the dataset we created. The generated questions are evaluated and the results are discussed in Section 6 and 7. Then we conclude our work in the final Section (8).

2 Related Work

Question Generation (QG) is an essential NLP task that can be used for supplementing educational materials (Heilman and Smith, 2010; Zhao et al., 2022), data augmentation for QA systems (Lyu et al., 2021; Le Berre et al., 2022) and understanding semantic relations within the text (Pyatkin et al., 2020; Klein et al., 2020). With the advent of large pretrained language models and large QA datasets, recent QG approaches are primarily based on transformer-based neural architectures. Dong et al. (2019) finetuned a unified language model for both QG and language understanding tasks. Yuan et al. (2021) enhanced the language model for QG with additional embeddings of linguistic features. To make generated questions more diverse, Murakhovs'ka et al. (2021) took advantage of the fully text-to-text generation model to to generate questions with mixed types of answers. Fei et al. (2022) applied graph networks to generate complex questions requiring broader contexts to answer.

Comparing with the neural methods, traditional rule-based QG systems (Levy and Andrew, 2006; Heilman and Smith, 2009) did not receive much attention due to the lack of diversity of the generated questions. To solve this, He et al. (2015) proposed QA-SRL that use predicate-argument structure to generate more semantic-informed QA pairs. More recent work also has shown the success in incorporating existing explicit semantic annotations and linguistic resources into the rule- or template-based QG systems (Dhole and Manning, 2021; Pyatkin et al., 2021). In this work, we adopt a template-based approach to generate competence-based questions that solicit implicit information and the event dynamics across a broad context.

More broadly related to our general purpose to probe the above-defined competence-based knowledge with QG, Tu et al. (2022) proposed a new QA task that queries competence-based knowledge structures; Rashkin et al. (2018) proposed a new commonsense inference task that involves the reasoning of intents and reactions to the events; Xu et al. (2021) used context information for commonsense QA. There also exist semantic tasks that explore the implicit or underspecified components of a linguistic expression (Roth et al., 2021; Karidi et al., 2021; Ye et al., 2022).

3 Task Definition

We define our task as generating a *competence-based* question set. Unlike traditional QG tasks whose answers are mostly extractive and contexts are largely dependent on surface lexical locality conditions, the goal of our task is to generate questions that represent commonsense semantic inferences from a large context in an abstractive way.

Table 1 shows example CB questions generated from the full text of a cooking recipe. Consider the IMPLICIT question. Our task is to generate such a question-answer pair that reflects the implied state of the object (*peeled* apples) and the missing argument from the context (*knife* to cut). Also consider the LOC. CHANGE and OBJ. LIFESPAN questions: the goal of our task is to solicit information that reflects the dynamics of events and actions through these questions.

4 Data Description

Due to the lack of existing datasets suitable to our task, we curate a collection of English cooking recipes as the data for our task. Recipes along with other procedural text like instructions tend to be task-oriented and stepwise. The content is composed of step sentences that describe small goals to accomplish the final task. Comparing to text of news or narratives, we think recipe texts are a good fit for our task as it involves the understanding of how to reach the goal locally for each step, as well as how each step contributes to the final task globally. Further, the stepwise progression inherent in the goal-oriented narrative contributes both an interpretative dynamics as well as contextualized elision of arguments. Our proposed dataset has also been used as part of the data resource for the SemEval-2022 Task 9: R2VQ (Tu et al., 2022). Here we describe the annotation process and the annotation alignment step for our QG task.

Data Preprocessing We build our QG dataset with a collection of public domain recipes.[1] We first run the Stanza pipeline (Qi et al., 2020) on the raw text of each recipe to get tokenization and other linguistic features including lemmatization, part-of-speech tagging and dependencies. We also run the state-of-the-art Semantic Role Labeling (SRL) parser from (Conia and Navigli, 2020) to label each recipe sentence with its semantic roles. Subsequently, we ask 2 students annotators to validate and correct both frames and argument labels.

[1] https://recipes.fandom.com/ , http://foodista.com/

Recipe Title: **Appelkoek** *Passage:* Peel and cut apples into eighths (wedges). Sift together flour, baking powder and salt with 4 tablespoons of the sugar. Cut in butter. Combine egg and milk and add to flour mixture. Turn batter into greased 8 inch square cake pan. Press apple wedges partly into batter. Combine remaining 2 tbsp sugar and cinnamon. Sprinkle over apple. Bake at 425 degF for 25 to 30 minutes.

IMPLICIT	How do you cut peeled apples into wedges? - by using a knife
ELISION	What should be sprinkled over apple wedges? - cinnamon sugar
TEMPORAL	For how long should you bake appelkoek? - 20 to 35 minutes
LOC. CHANGE	Where was the batter when you press apple wedges? - *still* in the pan
OBJ. LIFESPAN	What's in the appelkoek? - apples, batter and cinnamon sugar
HAB. STATE	What's already in the bowl when you add egg and milk to it? - butter and dry ingredients

Table 1: Example competence-based questions. Color-coded text spans represent how information has been collected and generated in the questions.

Table 2 shows the basic statistics of the the preprocessed recipes. We have filtered out non-English and non-food recipes, as well as short recipes with only 4 or less sentences. This results in 900 preprocessed recipes in total with an average of 8 sentences per recipe and 13 tokens per recipe sentence. We randomly select 800 recipes for experimenting our QG method, and the other 100 is held out for the evaluation of the generated QA pairs.

	Train	Val
# of recipes	800	100
Avg. # of sentences per recipe	8	7.9
Max./Min. # of sentences	26/4	16/4
Avg. sentence length per recipe	12.5	13.4
Max./Min. sentence length	32/6	25/6

Table 2: Statistics of the train and validation subsets of the QG dataset.

Cooking Role Annotation We prepare the preprocessed recipes for QG by adding Cooking Role Labeling/Linking (CRL), a span-level annotation layer for identifying cooking events from recipe text. We define the event ontology as a set of cooking-related entities and relations. The entity types include the EVENT-HEAD, INGREDIENT, TOOL and HABITAT. The relations include PARTICIPANT-OF and RESULT-OF. Each event has only one predicative verb (EVENT-HEAD), and all the relations within the event are linked from corresponding entities to the predicate. A sample event is provided in the Appendix, Figure 2. More importantly, CRL annotates events that involve implicitly expressed arguments by identifying their hidden entities. For example, consider the sentence *Sprinkle over apple.* from Table 1. In this event, the hidden TOOL *hand*, the hidden HABITAT *cake pan* and hidden INGREDIENTs *cinnamon* and *sugar* are the most plausible participants of the event head *sprinkle*, but are not explicitly stated. The identified hidden entities should be either inferred elsewhere explicitly on the document level, or inferred based on the commonsense.

We start the CRL annotation by labeling explicit entities semi-automatically using a separately trained NER model. Relations and hidden entities are annotated manually. Then we annotate the coreference of tools, habitats and ingredients through the full recipe at the document level. Specifically, we hired 12 trained student annotators for the CRL annotation and validation work.[2] Table 3 shows the average number of annotated entities per recipe. The average number of event verbs (14) are much great than the average recipe length (8) from Table 2, indicating that many recipe sentences tend to involve more than one event. Ingredient participants are the most prevalent entity type under both explicit and hidden settings. Recipes also have more hidden ingredient results, tools and habitats instead of explicit ones, showing the importance of hidden arguments for understanding cooking recipes or instructional text in general.

	Train		Val	
	Exp.	Hidden	Exp.	Hidden
EVENT-HEAD	14.0	N/A	13.6	N/A
INGREDIENT (participant)	13.0	6.9	14.0	10.8
INGREDIENT (result)	0.2	1.5	0.2	1.4
TOOL	0.6	2.1	0.7	2.2
HABITAT	2.8	4.8	2.5	6.2

Table 3: Average number of annotated explicit/hidden entities per recipe from the QG dataset. EVENT-HEAD can only be explicit.

Aligning CRL and SRL We further process the CRL annotation by aligning it with SRL. Specifically, for any given sentence, we align its CRL and SRL annotation that share the same event predicate (marked as `PREDICATE` in SRL and `event-head` in CRL). Semantic roles are merged with corresponding cooking entities if their text spans are overlapped. For example, in the sentence *Transfer peas to the saucepan*, the role *to the saucepan* `[DESTINATION]` will be merged with the entity *saucepan* `[HABITAT]`, and the text span *peas* is both the entity `INGREDIENT` and the role

[2]Full annotation process can be found in Appendix A.2.

THEME. The un-overlapped roles such as TIME and ATTRIBUTE will be categorized as modifiers to this cooking event. The CRL-SRL aligned events will later be used to generate questions that are able to hold richer context.

5 Experiment Method

Now we describe a template-based method to generate competence-based questions. We explore the data with CRL annotation in two phases: first we adopt the concept of question family in §5.1 by proposing question templates and heuristics to generate QA pairs that involve hidden arguments for each family. For example, given a sentence *Sauté the onions until browned.*, we generate the question *Where should you sauté the onions?*. The correct answer *in the pan* is not in the current sentence, thus needing to be inferred from its context. In §5.2 we propose the concept *dense paraphrasing* that can record previous states of the entities as being processed though events, e.g. *sauté onions* will be rewritten as *sauté* ***chopped*** *onions* in generated questions based on the given context. Then we apply it to enhance question families by generating new questions and answers that cover a bigger context and dynamics of events.

5.1 Generating Competence-based Questions

Motivated by the concept of Question Family (QF) that is first outlined and adopted by Johnson et al. (2017) to create the visual question answering dataset, we first introduce our *question families*, which contain a set of text templates and semantic reasoning heuristics that can generate the full spectrum of competence-based questions. We define the question families in Table 4. Each question family is designed to test a specific competence with generated questions using the hand-crafted text templates.

We generate QA pairs by populating templates with slots in a cloze test style. Candidate slots (colored spans in Table 4) are acquired from CRL-SRL aligned events we created earlier. We also set the constraints to only keep the events with at least one hidden cooking entity for template population, so that the generated questions are competence-based, rather than purely "memorizing" original text spans. Each aligned event is composed of cooking entities with semantic roles, relation links between entities and event verb, as well as SRL modifiers to the event. All the required slots and the candidate answer for a template are self-contained within an aligned event.[3]

After a text template is populated, it is further processed to improve the readability of the generated question. We change word inflections and insert articles and agreements. For the templates with [habitat_phrase] and [tool_phrase] slots, we fill those with corresponding LOCATION or INSTRUMENT spans from SRL. If a slot is filled with a hidden entity, we run a BERT-based model (Devlin et al., 2019) to get the most likely preposition given the sentence as context through the masked language modeling task. Modifiers are populated in the same order as they were in the original sentence.

To increase the variety of questions, we allow adjunct slots in the text templates. As shown in Table 4, adjunct slots include tool/habitat phrases and modifiers. For example, one ELISION question can be as short as *What should be sautéed?* or *... sautéed in the saucepan with the spatula until browned?* with all the adjunct slots. We argue it is helpful to generate questions more challenging to the systems. Adding more adjunct slots completes the context for the question, but also introduces unseen context if the slots contain hidden entities.

5.2 Generating Enhanced CB Questions

We further improve our QFs by 1) explicating ambiguity of entity coreference, 2) generating questions from larger context. Consider the sentence *Peel and cut apples into wedges.* from Table 1. The entity apple will be transformed to peeled apples after the peel event, and that is what is participating in the following cut event, rather than the same referred entity. We enhance the competence-based QG to capture this layer of underlying semantics. In addition, we also include document-level QFs into our system. For the question *What's in the appelkoek?*, an answer *apple wedges* can be extracted only from the last sentence where the event that contains *applkoek* is annotated. While the answer is not wrong, it's not complete enough to include all the necessary ingredients that can be summarized from the whole recipe (Table 1).

Dense Paraphrasing To reflect the aforementioned difference between *apples vs. peeled apples* in QG, we enrich the source narrative with a dynamic *Dense Paraphrase* of the surface text, which both decontextualizes the expression (Choi et al., 2021; Elazar et al., 2021), and enriches the textual description of both events and participants to reflect

[3] One exception is for *Cardinality*, which requires the count of the co-referred entities on the document level. The entites from *Cardinality* also do not need to be hidden.

QFs	Text Template	Question Sentence
Cardinality	How many toolObj|habitatObj are used? How many times is the toolObj|habitatObj used? How many action does it take to process IngreObj?	How many bowls are used? How many times is the microwave used? How many action does it take to process the pasta?
Elision	What should be verb [habitat_phrase] [tool_phrase] [modifiers]?	What should be baked in the oven at 425 degree?
Implicit	What do you use to verb obj [habitat_phrase] [modifiers]? Where do you verb obj [tool_phrase] [modifiers]?	What do you use to sauté the onions [in the pan]? Where do you arrange the slices into rounds?
Obj. Lifespan	What is in obj? How did you get ingreObj?	What is in the appelkoek? How did you get the appelkoek?
Semantic Role	*For how long* do you verb obj [tool_phrase] [habitat_phrase] [modifiers]? *To what extent* do you verb obj [tool_phrase] [habitat_phrase] [modifiers]?	For how long do you marinate the shrimp? How do you add the water to the bowl?

Table 4: Text templates and generated questions from each question family. ***Cardinality***: integer comparison and counting; ***Elision***: hidden arguments that can be understood from context; ***Implicit***: implicit tools/habitats that require world knowledge; ***Obj. Lifespan***: different object states; ***Semantic Role***: semantic roles from a cooking event. In the templates, bar symbol (...|...) indicates either slot is acceptable; squared brackets ([...]) indicates adjunct slots.

Sense	Verbs	Category
CONVERT	melt, cream, evaporate	Transformation
SPILL_POUR	pour, ladle, drip	Loc. Change
AMELIORATE	enhance, improve, round out	N/A

Table 5: Example event verbs and their senses along with the assigned end state type.

the changes in the object due to the events. This includes descriptions involving subevent decomposition (Pustejovsky, 1995; Im and Pustejovsky, 2010), which tags the event as having three parts: **begin (Be)**, **inside (Ie)**, and **end (Ee)**.

In practice, we only track the begin and end state of an event. We define two types of event end state: TRANSFORMATION and LOCATION-CHANGE. For example, the end state of a *chop onions* event is *chopped onions*, and the end state type is TRANSFORMATION, denoting a change of the object in shape, size or color, etc. To get the end state type of each event, we collected 208 unique verb senses (Di Fabio et al., 2019) that are assigned by the SRL parser to our data, and hand-split those into three categories: transformation, location change or neither. Then we assign the end state type based on the category of the event verb sense. Table 5 shows an example from each category.

We incorporate dense paraphrasing into the QG by replacing the INGREDIENT entities from each aligned event with its dense-paraphrased version before the template population. Specifically, for the event verb from transformation, the paraphrased entities will have the format of `pastParticipleVerb + object`, e.g. *minced garlic*, *heated water*. To retain the naturalness and readability of the questions, we keep the last one or two event end states of an ingredient. For example, *sautéed chopped peeled onions* will be truncated to *sautéed onions* or *sautéed chopped onions*.

Document-level QG Unlike previous QFs where each template is populated by only one event, we first construct all the CRL-SRL aligned events from each recipe into data views that can provide a global understanding of the whole recipe. Each data view is used to generate questions or answers in document level. Subfigure 1(a) shows a directed graph that connects events by ingredients. `result-of` connects the ingredient participant and result within an event; `coreference-of` connects ingredients from two events where the result of the first one is *directly* used as the participant of the second one. This data view is used to enhance the OBJ. LIFESPAN QF by expanding the answers to include all major ingredients from the whole text, rather than just the ingredients mentioned in current sentence. We describe the steps taken in Algorithm 1 to generate such answer. Given an intermediate ingredient that is asked about, we generate a subgraph of events thorough depth-first-search traversal over the graph from the vertex of the given ingredient. Then we collect the ingredient results from the subgraph as the answer.

We also propose new document-level QF templates that are populated through traversing the other two data views. Table 1 shows example questions from the newly proposed question families, namely LOC. CHANGE and HABITAT STATE. We succinctly describe the steps to generate QA pairs from the new QFs, and provide the corresponding algorithms in the appendix.

Subfigure (b) shows a grid that tracks the change of location of ingredients through events. Each row

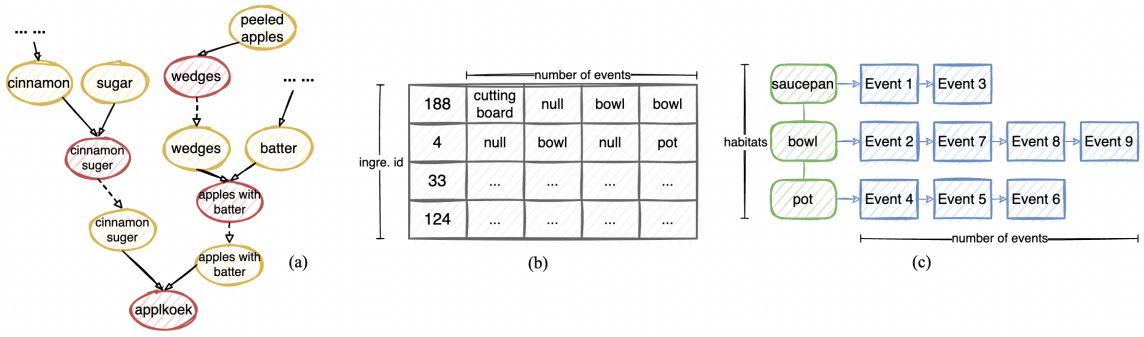

Figure 1: Data views of cooking events from a recipe. (a) A directed graph that connects events by ingredients. (● = ingredient participants, ● = ingredient results, → = result-of, - - ➢ = coreference-of); (b) A grid shows the habitat of each ingredient in every event; (c) An index links events by the location where the they took place.

Algorithm 1 Object Lifespan QF Heuristics

Require: Graph $G(V, E)$ and ingredient $ingre_i$
access the ingredient result vertex v_i that stores $ingre_i$
generate subgraph $g \leftarrow$ predecessors in depth-first-search from v_i
ingredient candidates $C \leftarrow \{\}$
for v_k in g **do**
  if v_k stores ingredient result $ingre_k$ and the at least one of parent vertices of v_k stores ingredient participant that is NOT the outcome from other events **then**
    $add(ingre_k, C)$;
  end if
end for
$answer \leftarrow \{C\}$
$question \leftarrow template(ingre_i)$

shows the habitat of an ingredient (represented by ingredient id) in every event. *null* means the ingredient is not involved in the given event, thus no habitat needs to be populated. To generate a LOC. CHANGE question with this data view, the habitat from the event grid is used. Then we compare the habitat from the preceding event with current habitat to decide which location change cue to use in the answers. The answer format can be either *STILL habitat_phrase*, *in ANOTHER habitat_phrase* or simply the habitat itself based on the comparison.

Subfigure (c) links events by the location where they took place. Each list contains all the events that have the same habitat given by temporal ordering. To generate HAB. STATE questions, given a habitat and event at a certain time point from the event index view, we look up the grid view to track what ingredients are added or removed from the given habitat dynamically through all the events that happen before the current one.

6 Evaluation and Discussion

In this section we describe the steps to evaluate our system. First we adopt crowd-sourced evaluation to measure the quality of generated questions intrinsically. Then we train a text generation model to perform QA tasks over the subsets of questions to measure the competence of questions and the challenges they post to existing systems. We then compare other existing QG systems with our own.

	EXPLICIT		HIDDEN		ENHANCED	
	Train	Val	Train	Val	Train	Val
SEMANTIC ROLE	5,513	700	8,169	1,064	901	135
ELISION	4,829	622	3,852	598	2,251	340
IMPLICIT	3,618	527	4,287	650	1,385	261
OBJ. LIFESPAN	781	69	1,098	124	2,392	312
CARDINALITY	-	-	4131	529	-	-
LOC. CHANGE	-	-	-	-	6,101	1,020
HAB. STATE	-	-	-	-	4,094	534
ALL	14,741	1,918	21,537	2,965	17,124	2,604

Table 6: Number of generated questions from each subset per QF.

To prepare the data for the evaluation, we apply our method on both the train and validation set of the QG dataset. The generated questions are categorized into three subset as shown in Table 6. EXPLICIT set contains generated questions of the QFs from Table 4, but the templates are populated with the events where the entities are all *explicitly* stated in the text. This set can be viewed as a non-competence extractive QA dataset. The HIDDEN set also contains the QFs we defined in Table 4 but with the constraint that a valid event should contain at least one hiddden entity. The ENHANCED set includes the dense paraphrased QFs as well as new document-level QFs from §5.2.

Intrinsic Evaluation Similar to other text generation task (Luong et al., 2015; Rush et al., 2015), human evaluation plays a critical role in question generation, as it can provide a more reliable evaluation and capture the nuance between generated text. We measure the generated QA pairs on a 5-point likert scale from *grammatical correctness*, *relevance* and *answer completeness*. The first two metrics are proposed in QG-STEC Task B (Rus et al., 2010) and widely used in previous work (Dhole and Man-

ning, 2021; Pyatkin et al., 2021). They measure the grammaticality and the relevence of the questions to the context, respectively. In addition, we propose answer completeness for our QG task. It measures whether the answer is complete, e.g., some ingredients are missing from the answer or redundant. This metric is helpful to our task as we generate the answers that involve hidden objects which also traverse the whole passage.

We conduct the intrinsic evaluation on the HIDDEN and the ENHANCED question subsets of the QG validation set. We assign the generated questions to 4 trained annotators who had the experience with CRL annotation, and ask them to score each QA pair. Table 7 shows the result from intrinsic evaluation. For grammatical correctness, SEMANTIC ROLE and HAB. STATE have lower scores than the others. We suspect this is because CRL has more templates for different roles and HAB. STATE has a rather complicated question structure, while others have relative fixed templates. The overall change in relevance score tends to be consistent with grammatical correctness. However, OBJ. LIFESPAN from ENHANCED has a low relevance score compared to its grammaticality score. Through analysis, we found the system tends to generate less useful and less relevant QA pairs such as *What is in the chopped tomatoes? - tomatoes*. OBJ. LIFESPAN from ENHANCED also has the lowest score on AC. This is expected since the desired answers for this QF may include a whole list of entities that are scattered over the text. As a comparison, IMPLICIT has the highest score on AC because of the simplicity of answer format (either the hidden habitat or tool). Compared to HIDDEN, the score also drops on QFs in ENHANCED. By checking the examples, we found some dense-paraphrased objects can be "unnatural" to read such as *mixed apple* and *drained water*, which may lead to this drop. Overall, by comparing the scores to other QG systems (Dhole and Manning, 2021; Pyatkin et al., 2021) that also adopt the human evaluation, the result proves the validity of our method.[4]

Question Answering Performance We also evaluate the competence of generated questions by performing the QA task on the question subsets. We fine-tune the T5-BASE text generation model (Raffel et al., 2020) on each question set from the training set and evaluate on the validation set using exact match (EM) and token-level F1 score (F1) following Rajpurkar et al. (2018). The model training detail is provided in the Appendix A.3.

	HIDDEN			ENHANCED		
	Gram.	Rel.	AC	Gram.	Rel.	AC
SEMANTIC ROLE	4.71	4.59	4.65	4.52	4.50	4.56
ELISION	4.93	4.77	4.57	4.92	4.79	4.53
IMPLICIT	4.92	4.87	4.83	4.84	4.76	4.78
OBJ. LIFESPAN	4.83	4.70	4.52	4.75	4.45	4.36
LOC. CHANGE	-	-	-	4.80	4.51	4.48
HAB. STATE	-	-	-	4.51	4.44	4.57
ALL	4.81	4.69	4.66	4.74	4.56	4.53

Table 7: Results from intrinsic evaluation on questions families with / without enhancement. CARDINALITY is not included as there is little variance between questions from this set, and the answer only contains numbers.

Table 8 shows the QA results from fine-tuned T5 on question subsets per question family. Overall, the model performs best on EXPLICIT set and performs worse on HIDDEN and ENHANCED. This observation confirms our assumption that CB questions indeed pose new challenges to existing systems. The model performs best on SEMANTIC ROLE and the score difference between question sets is much smaller comparing to the overall. This is because the answers for this QF are explicit semantic roles that can be retrieved in an extractive manner. For OBJ. LIFESPAN, the score difference between questions sets is much greater. Compared with EXPLICIT which only account for explicit ingredients, HIDDEN set requires a correct answer to include all hidden ingredients from the given event. ENHANCED set further accounts for dense-paraphrased answers from the document-level, thus resulting in the most challenging set for OBJ. LIFESPAN. Comparing the two multi-sentence question families in ENHANCED, the model performs much better on LOC. CHANGE. By examining the examples, we found that some LOC. CHANGE questions involve events that are next to each other, which requires local context for the model to pick up the necessary information to answer it. In contrast, HAB. STATE is the most difficult question family as it asks the model to track the relative position between habitat and ingredients globally to answer it correctly.

To understand whether CRL annotation can help answer CB questions, we fine-tune the T5 model on the ENHANCED set again, but we modify the input string to include the CRL by inserting hidden entities as a human-readable format to the in-

[4]Dhole and Manning (2021) and Pyatkin et al. (2021) also evaluated the result on grammatical correctness and relevance, resulting in 3.93/4.34 and 4.57/4.29 on the data presented in their work. Despite the difference in data and templates makes it difficult to compare the exact scores, we argue the relative score (e.g. all above 3.5) is still useful to look at due the evaluation is based on human judgement.

	EXPLICIT			HIDDEN			ENHANCED			- w/ CRL	
	EM	F1	Count	EM	F1	Count	EM	F1	Count	EM	F1
SEMANTIC ROLE	92.59	96.92	700	90.51	96.05	1064	84.44	89.68	135	60.00	72.46
ELISION	66.72	75.63	622	27.59	50.93	598	25.16	46.20	340	40.88	59.07
IMPLICIT	62.43	71.95	527	59.69	69.15	650	48.28	58.13	261	70.88	75.78
OBJ. LIFESPAN	40.58	58.38	69	23.39	54.43	124	10.58	48.82	312	14.10	60.58
CARDINALITY	-	-	-	71.08	71.08	529	-	-	-	-	-
LOC. CHANGE	-	-	-	-	-	-	56.47	74.05	1020	60.00	74.45
HAB. STATE	-	-	-	-	-	-	8.96	42.69	536	18.66	52.36
ALL	73.72	81.69	1918	64.79	74.86	2965	37.73	60.15	2604	44.59	66.26

Table 8: Results from fine-tuning T5 model for the question-answering task on three different subsets of generated questions. For ENHANCED subset, we collect two sets of results from the models trained with or without CRL annotation. CARDINALITY does not have counterparts from EXPLICIT and ENHANCED sets. LOC. CHANGE and HAB. STATE are only introduced in ENHANCED for multi-sentence QG.

put.[5] Table 8 also shows the results from the model trained with CRL, which outperforms the result from baseline T5. It shows the utility of CRL to carry competence-based knowledge. Interestingly, the scores for SEMANTIC ROLE questions drop significantly (24.44 on EM) with the new model. This is because we reconstruct the context text to include CRL, but that breaks the extractive nature of SEMANTIC ROLE questions. However, it can be avoided by training the system with different input.

Context: ... **Add**$_{[2]}$ to it potatoes and red pepper. **Stir**$_{[1]}$ well for 2 minutes.

SynQG	[1] For how long does someone stir well? [2] What vegetable is added to it?
RoleQ	[1] What stirs well? / Why does someone stir something something? [2] What does something add something to?
CompQG	[1] What do you use to stir the fried potatoes and red pepper? / For how long do you stir the fried potatoes and red pepper in the pot? [2] Where are the fried potatoes before you add them to the pot? / What is already in the pot when you stir the fried potatoes?

Table 9: Examples of questions generated by SynQG, RoleQ and CompQG (this work).

Comparison to Other QG Systems As a qualitative study of our system, we compare it to two QG systems: Syn-QG (Dhole and Manning, 2021) and RoleQG (Pyatkin et al., 2021). Due to the fact that all three systems are essentially different regarding their domain and model I/O, we focus on comparing how the systems differ in the final outcome and how our system fits into the larger scope of QG. Syn-QG leverages syntactic and semantic rules to generate template-based questions. RoleQG proposes a hybrid method of templates and seq2seq models to generate questions for any semantic role that may exist in the context. We adopt Syn-QG and RoleQG on a sample data and summarize our observations from comparing results.[6]

Procedural vs. descriptive text Both SynQG and RoleQ are developed on datasets with narratives, such as OntoNotes 5.0 (Weischedel et al., 2013) and SQuAD (Rajpurkar et al., 2016). However we choose to apply our method on procedural text like cooking recipes. This difference in data format and scope can impact the operations for QG tasks. Procedural text tends to be imperative and instructional. This may lead to certain semantic roles being naturally omitted from sentences. For example in Table 9, the questions for Agent and Cause (RoleQ [1]) may not fit for the given sentence. Procedural text also involves a specific task under a given setting, which could lead to text elision and ambiguity as the same arguments may appear multiple times across the text. This hinders existing systems from generating accurate questions. In Table 9, if the *stir* action is performed more than once in the context, question SynQG [1] would be ambiguous to ask. The corresponding answers (*potatoes and red pepper*) also need to be inferred from other sentences. Thus we propose CRL aligned with SRL that is able to account for the above mentioned properties from procedural text. The questions from our work (CompQG [1]) tend to either include hidden text slots in the questions, or solicit them in the answer.

Competence vs. ubiquitous QG Traditional QG methods are designed to generate as many questions as possible to have a wide coverage of the context. To achieve that, SynQG includes named entities and hypernyms into SRL arguments to increase the variety of SRL templates (SynQG [2]); RoleQ generates questions for all possible semantic roles of the predicate even though the answers can not be identified in the text (RoleQ [1]). As a com-

[5] E.g., input `sauté until browned` is changed to `{using a spatula} {on the cutting board} sauté {chopped onions} until browned {resulting in sautéed onions}`

[6] For SynQG, We directly follow their proposed SRL templates and heuristics in §3.2 of their paper. For RoleQ, we follow their instruction to run the model provided in `github.com/ValentinaPy/RoleQGeneration`.

parison, we focus on competence-based QG with special attention to answer completeness. Answers to SRL questions are just semantic roles that can be retrieved locally with ease (SRL row in Table 8). They lack the capability to uncover underlying semantics and solicit new information globally, which are what we are trying to address with extra steps to connect sentences and generate complete answers from data views. For example, questions from CompQG [2] leverage the multi-sentence information. This move could be especially suitable to procedural text as it is task-oriented and further steps need to build on past actions.

7 Error Analysis and Postprocessing

Based on the intrinsic evaluation results from Table 7 and manual inspection of the generated questions, we summarize the common errors that we identified from the system output.

Error propagation from the annotation Some wrong or incomplete answers are from the data annotation. This may happen more frequently in OBJ. LIFESPAN when certain hidden ingredients are not annotated. Missing of the annotation also reduces the number of QA pairs that can be generated automatically.

Unnatural and Uninformative Questions Pronouns and simple events can result in questions that are less helpful. For example, question *To what extent do you turn it?* and *What should be cooked* cannot be answered due to the lack to specific reference and proper context. Another case is when the ingredient participant and result is the same and is the only ingredient in the QA pair. Example like that is *What is in the chopped tomatoes? - tomatoes*, which reads unnatural.

Question or Answer Ambiguity Question ambiguity may happen when the question structure is simple and the same event action is performed multiple times in the same context. For example, the question *What should be sliced?* is ambiguous if the action *slice* is mentioned in different sentences from the same recipe. Answer ambiguity happens more frequently in questions that involve the world knowledge or commonsense from the annotation. Given the questions *How do you roll the dough into rounds?*, if the candidate tool is not explicitly mentioned elsewhere in the same recipe, the annotator should resort to their world knowledge to fill it in. However, different annotators may have different answers such as *rolling pin* or *bare hands*.

To solve or alleviate aforementioned errors, we postprocess the data based on some heuristics to filter out the low-quality QA pairs. Before the events are populated into the templates, we fill in possible missing annotation for hidden tools or habitats. We assign the most likely tool or habitat to the given event based on the frequency distribution of the entity from the existing annotation. For example, if the event head *cut* has no hidden tool annotated, *knife* as the most frequent tool to this event head will be assigned. To make the questions less ambiguous, the templates that are used to generate questions involve common verbs such as *add, cook, put, place* need to contain as least one adjunct slot. This can generate more specific questions. We also remove QA pairs that contain pronouns and the pairs whose answer is only one ingredient and overlapped with the entities that are asked about.

Although the errors from the annotation and automation process are hard to avoid, the annotation overall can be improved through more careful post validation and adjudication of the data. Answer ambiguity can also be partially solved by setting rules in the specification to favor more specific entities over the general ones (e.g. *rolling pin vs. hands*, *large spoon vs. spoon*), or allowing multiple valid answers. We leave these to the future discussion.

8 Conclusion

We have proposed a method for generating competence-based questions that leverages lexical semantic knowledge involving, implicit arguments, subevent structure of verbs, and document-level event dynamics. To that end, we have constructed a new dataset that encodes manually annotated lexical semantic knowledge in a corpus of cooking recipes. Our proposed method includes a set of rich targeted question families and a suite of dense paraphrasing operations that facilitate systems to generate answers both locally and globally. We have conducted intrinsic evaluation of the system to show the quality and usefulness of CB questions. We have also performed QA tasks by letting systems answer generated questions; this in turn has provided potential insights into further improvement of existing large language models for understanding such questions. By comparing to existing QG systems, our work has focused on generating questions from the subdomain of procedural text, and the questions are designed to query *competence-based* knowledge. In future research, we intend to further improve our method by minimizing the annotation errors and ambiguous answers as indicated in the error analysis, and then apply the method to a broader range of text genres.

Acknowledgements

We would like to thank our colleagues at Brandeis and Sapienza Università di Roma, who were co-organizers of the 2022 SemEval challenge, from which the QG dataset has been drawn. In particular, we would like to acknowledge the help of Marco Maru, Simone Conia, and Roberto Navigli. We would also like to thank Brandeis LLC lab members: Eben Holderness, Kelley Lynch, and Richard Brutti for their help in orchestrating, compiling, and annotating the QG dataset; and Ayla Taylor for her initial work on creating a decontextualized generation task from the recipe dataset. All views expressed in this paper are those of the authors and no others.

References

Alan Akbik, Tanja Bergmann, Duncan Blythe, Kashif Rasul, Stefan Schweter, and Roland Vollgraf. 2019. Flair: An easy-to-use framework for state-of-the-art nlp. In *NAACL 2019, 2019 Annual Conference of the North American Chapter of the Association for Computational Linguistics (Demonstrations)*, pages 54–59.

Gregory A Bechtel, Ruth Davidhizar, and Martha J Bradshaw. 1999. Problem-based learning in a competency-based world. *Nurse Education Today*, 19(3):182–187.

Luisa Bentivogli, Ido Dagan, and Bernardo Magnini. 2017. The recognizing textual entailment challenges: Datasets and methodologies. In Nancy Ide and James Pustejovsky, editors, *Handbook of Linguistic Annotation*, pages 1119–1147. Springer.

Eunsol Choi, Jennimaria Palomaki, Matthew Lamm, Tom Kwiatkowski, Dipanjan Das, and Michael Collins. 2021. Decontextualization: Making sentences stand-alone. *Transactions of the Association for Computational Linguistics*, 9:447–461.

Noam Chomsky. 1965. *Aspects of the Theory of Syntax*, volume 11. MIT Press.

Seung Youn Chyung, Donald Stepich, and David Cox. 2006. Building a competency-based curriculum architecture to educate 21st-century business practitioners. *Journal of Education for Business*, 81(6):307–314.

Simone Conia and Roberto Navigli. 2020. Bridging the gap in multilingual semantic role labeling: a language-agnostic approach. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 1396–1410, Barcelona, Spain (Online). International Committee on Computational Linguistics.

J. Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. Bert: Pre-training of deep bidirectional transformers for language understanding. In *NAACL-HLT*.

Kaustubh D. Dhole and Christopher D. Manning. 2021. Syn-qg: Syntactic and shallow semantic rules for question generation.

Andrea Di Fabio, Simone Conia, and Roberto Navigli. 2019. VerbAtlas: a novel large-scale verbal semantic resource and its application to semantic role labeling. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 627–637, Hong Kong, China. Association for Computational Linguistics.

Jean-Paul Doignon and Jean-Claude Falmagne. 1985. Spaces for the assessment of knowledge. *International journal of man-machine studies*, 23(2):175–196.

Li Dong, Nan Yang, Wenhui Wang, Furu Wei, Xiaodong Liu, Yu Wang, Jianfeng Gao, Ming Zhou, and Hsiao-Wuen Hon. 2019. Unified language model pre-training for natural language understanding and generation. In *33rd Conference on Neural Information Processing Systems (NeurIPS 2019)*.

Yanai Elazar, Victoria Basmov, Yoav Goldberg, and Reut Tsarfaty. 2021. Text-based np enrichment. *arXiv e-prints*, pages arXiv–2109.

Zichu Fei, Qi Zhang, Tao Gui, Di Liang, Sirui Wang, Wei Wu, and Xuanjing Huang. 2022. CQG: A simple and effective controlled generation framework for multi-hop question generation. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 6896–6906, Dublin, Ireland. Association for Computational Linguistics.

Dirk Geeraerts. 2009. *Theories of lexical semantics*. OUP Oxford.

Luheng He, Mike Lewis, and Luke Zettlemoyer. 2015. Question-answer driven semantic role labeling: Using natural language to annotate natural language. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 643–653, Lisbon, Portugal. Association for Computational Linguistics.

Michael Heilman and Noah A. Smith. 2009. Question generation via overgenerating transformations and ranking.

Michael Heilman and Noah A. Smith. 2010. Good question! statistical ranking for question generation. In *Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, pages 609–617, Los Angeles, California. Association for Computational Linguistics.

Jürgen Heller, Thomas Augustin, Cord Hockemeyer, Luca Stefanutti, and Dietrich Albert. 2013. Recent developments in competence-based knowledge space theory. In *Knowledge spaces*, pages 243–286. Springer.

Cheng-Ting Hsiao, Fremen ChihChen Chou, Chih-Cheng Hsieh, Li Chun Chang, and Chih-Ming Hsu. 2020. Developing a competency-based learning and assessment system for residency training: analysis study of user requirements and acceptance. *Journal of medical Internet research*, 22(4):e15655.

Seohyun Im and James Pustejovsky. 2010. Annotating lexically entailed subevents for textual inference tasks. In *Twenty-third international FLAIRS conference*.

Justin Johnson, Bharath Hariharan, Laurens Van Der Maaten, Li Fei-Fei, C Lawrence Zitnick, and Ross Girshick. 2017. Clevr: A diagnostic dataset for compositional language and elementary visual reasoning. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2901–2910.

Taelin Karidi, Yichu Zhou, Nathan Schneider, Omri Abend, and Vivek Srikumar. 2021. Putting words in BERT's mouth: Navigating contextualized vector spaces with pseudowords. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 10300–10313, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Ayal Klein, Jonathan Mamou, Valentina Pyatkin, Daniela Stepanov, Hangfeng He, Dan Roth, Luke Zettlemoyer, and Ido Dagan. 2020. QANom: Question-answer driven SRL for nominalizations. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 3069–3083, Barcelona, Spain (Online). International Committee on Computational Linguistics.

Guillaume Le Berre, Christophe Cerisara, Philippe Langlais, and Guy Lapalme. 2022. Unsupervised multiple-choice question generation for out-of-domain Q&A fine-tuning. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 732–738, Dublin, Ireland. Association for Computational Linguistics.

R. Levy and Galen Andrew. 2006. Tregex and tsurgeon: tools for querying and manipulating tree data structures. In *LREC*.

Thang Luong, Hieu Pham, and Christopher D. Manning. 2015. Effective approaches to attention-based neural machine translation. In *EMNLP*.

Chenyang Lyu, Lifeng Shang, Yvette Graham, Jennifer Foster, Xin Jiang, and Qun Liu. 2021. Improving unsupervised question answering via summarization-informed question generation. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 4134–4148, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Diego Marconi. 1997. *Lexical competence*. MIT press.

Shervin Minaee, Nal Kalchbrenner, Erik Cambria, Narjes Nikzad, Meysam Chenaghlu, and Jianfeng Gao. 2021. Deep learning based text classification: A comprehensive review.

Lidiya Murakhovs'ka, Chien Sheng Wu, Tong Niu, Wenhao Liu, and Caiming Xiong. 2021. Mixqg: Neural question generation with mixed answer types. *ArXiv*, abs/2110.08175.

Emmanouil Antonios Platanios, Otilia Stretcu, Graham Neubig, Barnabas Poczos, and Tom Mitchell. 2019. Competence-based curriculum learning for neural machine translation. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 1162–1172, Minneapolis, Minnesota. Association for Computational Linguistics.

Shrimai Prabhumoye, Ruslan Salakhutdinov, and Alan W Black. 2020. Topological sort for sentence ordering. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 2783–2792, Online. Association for Computational Linguistics.

James Pustejovsky. 1995. *The Generative Lexicon*. MIT Press, Cambridge, MA.

Valentina Pyatkin, Ayal Klein, Reut Tsarfaty, and Ido Dagan. 2020. QADiscourse - Discourse Relations as QA Pairs: Representation, Crowdsourcing and Baselines. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 2804–2819, Online. Association for Computational Linguistics.

Valentina Pyatkin, Paul Roit, Julian Michael, Reut Tsarfaty, Yoav Goldberg, and Ido Dagan. 2021. Asking it all: Generating contextualized questions for any semantic role.

Peng Qi, Yuhao Zhang, Yuhui Zhang, Jason Bolton, and Christopher D. Manning. 2020. Stanza: A python natural language processing toolkit for many human languages. In *ACL*.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J. Liu. 2020. Exploring the limits of transfer learning with a unified text-to-text transformer. *Journal of Machine Learning Research*, 21(140):1–67.

Pranav Rajpurkar, Robin Jia, and Percy Liang. 2018. Know what you don't know: Unanswerable questions for SQuAD. In *Proceedings of the 56th Annual*

Meeting of the Association for Computational Linguistics (Volume 2: Short Papers), pages 784–789, Melbourne, Australia. Association for Computational Linguistics.

Pranav Rajpurkar, Jian Zhang, Konstantin Lopyrev, and Percy Liang. 2016. Squad: 100,000+ questions for machine comprehension of text. In *EMNLP*.

Hannah Rashkin, Maarten Sap, Emily Allaway, Noah A. Smith, and Yejin Choi. 2018. Event2Mind: Commonsense inference on events, intents, and reactions. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 463–473, Melbourne, Australia. Association for Computational Linguistics.

Marco Tulio Ribeiro, Tongshuang Wu, Carlos Guestrin, and Sameer Singh. 2020. Beyond accuracy: Behavioral testing of NLP models with CheckList. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 4902–4912, Online. Association for Computational Linguistics.

Anna Rogers, Matt Gardner, and Isabelle Augenstein. 2021. Qa dataset explosion: A taxonomy of nlp resources for question answering and reading comprehension.

Michael Roth, Reut Tsarfaty, and Yoav Goldberg, editors. 2021. *Proceedings of the 1st Workshop on Understanding Implicit and Underspecified Language*. Association for Computational Linguistics, Online.

Vasile Rus, Brendan Wyse, Paul Piwek, Mihai Lintean, Svetlana Stoyanchev, and Christian Moldovan. 2010. The first question generation shared task evaluation challenge. In *Proceedings of the 6th International Natural Language Generation Conference*. Association for Computational Linguistics.

Alexander M. Rush, Sumit Chopra, and Jason Weston. 2015. A neural attention model for abstractive sentence summarization. In *EMNLP*.

Jingxuan Tu, Eben Holderness, Marco Maru, Simone Conia, Kyeongmin Rim, Kelley Lynch, Richard Brutti, Roberto Navigli, and James Pustejovsky. 2022. SemEval-2022 task 9: R2VQ – competence-based multimodal question answering. In *Proceedings of the 16th International Workshop on Semantic Evaluation (SemEval-2022)*, pages 1244–1255, Seattle, United States. Association for Computational Linguistics.

Richard A Voorhees. 2001. Competency-based learning models: A necessary future. *New directions for institutional research*, 2001(110):5–13.

Ralph Weischedel, Martha Palmer, Mitchell Marcus, Eduard Hovy, Sameer Pradhan, Lance Ramshaw, Nianwen Xue, Ann Taylor, Jeff Kaufman, Michelle Franchini, et al. 2013. OntoNotes release 5.0 LDC2013T19.

Yichong Xu, Chenguang Zhu, Ruochen Xu, Yang Liu, Michael Zeng, and Xuedong Huang. 2021. Fusing context into knowledge graph for commonsense question answering. In *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*, pages 1201–1207, Online. Association for Computational Linguistics.

Bingyang Ye, Jingxuan Tu, Elisabetta Jezek, and James Pustejovsky. 2022. Interpreting logical metonymy through dense paraphrasing. In *Proceedings of the Annual Meeting of the Cognitive Science Society*, volume 44.

Wei Yuan, Tieke He, and Xinyu Dai. 2021. Improving neural question generation using deep linguistic representation. In *Proceedings of the Web Conference 2021*, WWW '21, page 3489–3500, New York, NY, USA. Association for Computing Machinery.

Zhenjie Zhao, Yufang Hou, Dakuo Wang, Mo Yu, Chengzhong Liu, and Xiaojuan Ma. 2022. Educational question generation of children storybooks via question type distribution learning and event-centric summarization. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 5073–5085, Dublin, Ireland. Association for Computational Linguistics.

A Appendix

A.1 Algorithms

This section shows the algorithms describe in §5.2. Algorithm 2 and 3 describe the steps to populate LOC. CHANGE and HAB. STATE QFs.

Algorithm 2 Location Change QF Heuristics

Require: Grid G, ingredient $ingre_i$ and event e_j
  habitat $h_{ij} \leftarrow G(ingre_i, e_j)$
  for $k = j - 1$ to 0 **do**
    if $h_{ik} \neq null$ **then**
      break loop
    end if
  end for
  if $id(h_{ij}) = id(h_{ik})$ **then**
    $answer \leftarrow \{'still', h_{ij}\}$
  else
    if $text(h_{ij}) = text(h_{ik})$ **then**
      $answer \leftarrow \{'another', h_{ij}\}$
    else
      $answer \leftarrow \{h_{ij}\}$
    end if
  end if
  $question \leftarrow template(e_j)$

A.2 Annotation Process

we posted annotation positions within several University-wide distribution lists, available to all students within the various departments targeted. We hired 8 student annotators for the recipe annotation work. They were paid at the University-

Algorithm 3 Habitat State QF Heuristics

Require: Grid G, Index Q, habitat h and event e_j
  events $E \leftarrow \{e_k \in Q(h) \mid k < j\}$
  ingredient candidates $C \leftarrow \{\}$
  for e_k in E **do**
    get ingredient result set R
    for every result r in R **do**
      for $i = k - 1$ to 0 **do**
        if $G(r, e_i) = h \wedge r \notin C$ **then**
          $add(r, C)$; break;
        else
          if $G(r, e_i) = null$ **then**
            continue;
          else
            break;
          end if
        end if
      end for
    end for
  end for
  $answer \leftarrow \{C\}$
  $question \leftarrow template(h, e_j)$

mandated rate of $15/hour for student research assistants. All annotators were students at a US-based university, ranging from undergraduate to master's program.

We start by running the SRL parser (Conia and Navigli, 2020) on the full dataset to label each recipe sentence with its semantic roles. Subsequently, we ask 2 students annotators to validate and correct both frames and argument labels. We then train Flair named entity recognition (NER) model[7] (Akbik et al., 2019) on 100 recipes annotated with cooking entities only. The model takes a tokenized sentence and outputs the entity tag for each token in BIO format. We apply the trained NER model to the rest of recipes to generate all the entities for the further validation. CRL annotation task includes the annotation for relations, hidden entities and coreference of entities. Each recipe is annotated once at full by one annotator. All the annotators are trained to be familiarized with the annotation guideline and annotation examples before they start the task. The full annotation guidelines for the CRL task will be available along with the publication of this paper. For the intrinsic evaluation, we assign the generated questions to 4 annotators who had the experience with CRL annotation. Annotators are familiarized with the scoring for sample questions before doing the evaluation. Annotators are also required to provide a short comment to account for low scored QA pairs.

Figure 2 shows a final CRL annotation. The ingredient `apples` as a direct object to the verb, is linked as a participant of `cut`. The ingredient `wedges` as the outcome, is linked as a result of `cut`. Both habitat and tool can only be participants of an event, not results.

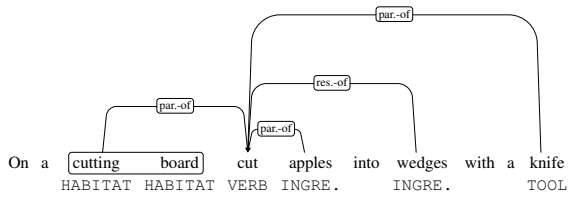

Figure 2: CRL annotation.

A.3 Model Detail

We fine-tune the T5 text generation model (Raffel et al., 2020) to perform question answering task on different question subsets. To make the model output comparable, we format each input instance to `"question: {question_str} context: {recipe_str}"` that includes the raw text of the whole recipe as context regardless of the question scope or implicity. For fine-tuning T5 with CRL annotation, we modify the context string to include the CRL by inserting hidden entities as a human-readable format to the input. For example, given a piece of the context: `sauté until browned`, we change it to `{using a spatula} {on the cutting board}, sauté {chopped onions} until browned {resulting in sautéed onions}` to recover the hidden objects to a human-readable format that can be consumed by T5. All the hidden entities from each event are wrapped with squared brackets. For each experiment run, we fine-tune T5-BASE model for 15 epoches on 4 NVIDIA Titan Xp GPUs. It took roughly an hour to finish the training. We adapt the training script from https://huggingface.co/valhalla/t5-base-qa-qg-hl.

A.4 Open Source License

- Flair NER - MIT License
- RoleQGeneration - Github Default
- Deep Event & Entity Palette - GPL3
- Docanno - MIT License

[7] https://github.com/flairNLP/flair

Coalescing Global and Local Information for Procedural Text Understanding

Kaixin Ma[†], **Filip Ilievski**[§], **Jonathan Francis**[†¶],
Eric Nyberg[†], **Alessandro Oltramari**[¶]
[†]Language Technologies Institute, Carnegie Mellon University
[§]Information Sciences Institute, University of Southern California
[¶]Human-Machine Collaboration, Bosch Research Pittsburgh
{kaixinm, jmf1, ehn}@cs.cmu.edu, ilievski@isi.edu, alessandro.oltramari@us.bosch.com

Abstract

Procedural text understanding is a challenging language reasoning task that requires models to track entity states across the development of a narrative. A complete procedural understanding solution should combine three core aspects: local and global views of the inputs, and global view of outputs. Prior methods considered a subset of these aspects, resulting in either low precision or low recall. In this paper, we propose **C**oalescing **G**lobal and **L**ocal **I**nformation (CGLI), a new model that builds *entity-* and *timestep-aware* input representations (local input) considering the *whole* context (global input), and we *jointly* model the entity states with a structured prediction objective (global output). Thus, CGLI simultaneously optimizes for both precision and recall. We extend CGLI with additional output layers and integrate it into a story reasoning framework. Extensive experiments on a popular procedural text understanding dataset show that our model achieves state-of-the-art results; experiments on a story reasoning benchmark show the positive impact of our model on downstream reasoning. We release our code here: https://github.com/Mayer123/CGLI

1 Introduction

Understanding the causal links of events in procedures is a key aspect of intelligence, facilitating human narration and dialogue. For instance, understanding why story B is plausible and why story A is not (Figure 1) requires procedural understanding of the causes of John leaving his notebook at home, as opposed to him taking out his notebook from his bag: writing in a notebook is counterfactual in the former case, and intuitive in the latter. Understanding stories requires procedural models that can reason consistently about event implications, and do so at different granularities. For a model to decide whether a story is plausible, it has to track the entity states over time, understand the effects of the described actions (green arrows), and consider the preconditions for a given action (pink arrows). Meanwhile, the model must reconcile the causes and effects of all events described in the story, to provide a globally consistent interpretation.

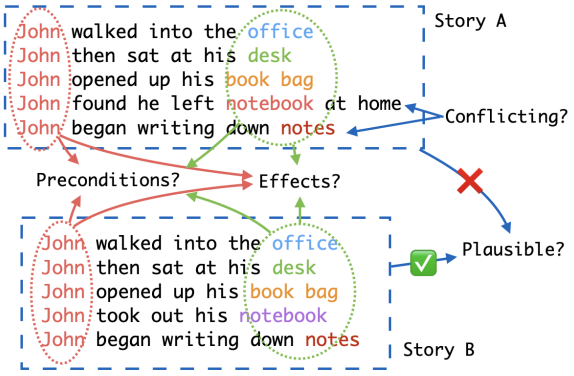

Figure 1: An example story of understanding task. Given two stories, the task is to judge which story is more plausible, find the conflicting sentence pair in the implausible story, and predict entity states at every step.

While procedural reasoning research reports steady progress in recent years (Rajaby Faghihi and Kordjamshidi, 2021; Gupta and Durrett, 2019a; Zhang et al., 2021), story understanding and procedural reasoning have rarely been considered together (Storks et al., 2021). Works have attended only to complementary aspects of the procedural reasoning problem, e.g., Gupta and Durrett (2019a) build entity-centric context representations and ignoring timestep-wise representation modeling; and Rajaby Faghihi and Kordjamshidi (2021) later proposed a timestep-specific model providing unique context encoding at every step to enable modeling flexibility. However, these methods predict independent step-wise entity states, thus compromising the dependency of outputs *across* different steps—yielding high recall but low precision. Global-output methods (Gupta and Durrett, 2019b; Zhang et al., 2021) explicitly leverage the strong

Proceedings of the 29th International Conference on Computational Linguistics, pages 1534–1545
October 12–17, 2022.

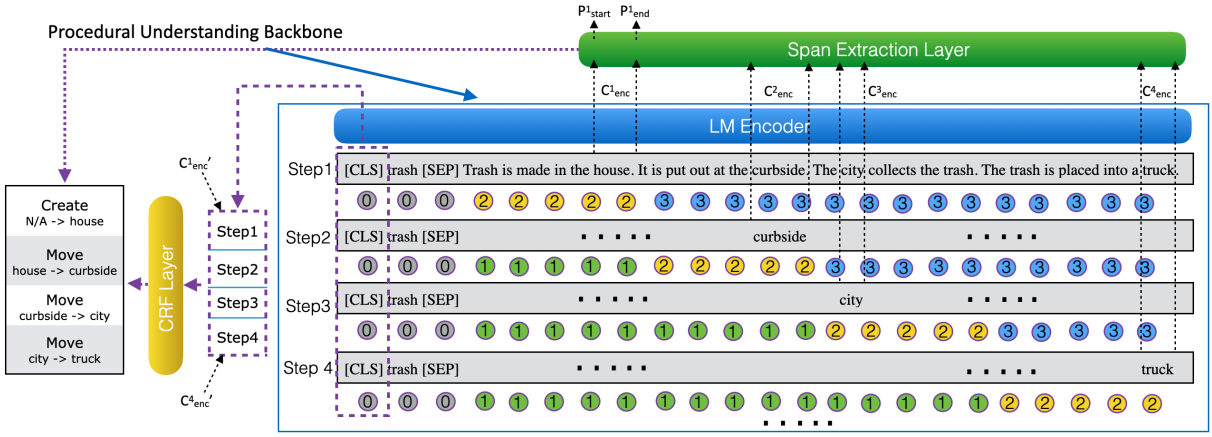

Figure 2: An illustration of CGLI. At every step, the LM encodes the full paragraph with different timestep ids (colored circles with numbers). The span extraction layer yields a location span for every entity at every step and this span sequence is combined with action sequence produced by a CRF layer to form the final predictions.

dependency across steps by jointly modeling the entity actions from all steps, but these methods only have one context encoding for all entities and steps, thus providing sub-optimal input representations—yielding high precision but low recall.

In this paper, we propose **C**oalescing **G**lobal and **L**ocal **I**nformation (CGLI): a new model for procedural text understanding that makes global decisions in consideration of entity-, timestep-centric, and global views of the input. To do so, our model builds a separate input view for every entity, at every step, while providing the whole context. Meanwhile, CGLI represents the entity actions across steps jointly with a structured prediction objective, thus achieving high consistency between different steps. The contributions of this paper are:

1. A novel procedural understanding method, CGLI, which produces global outputs of narrative procedures based on a unified view of the input, combining both local (entity-centric, timestep-specific) and global (document-wide) views—thus optimizing precision and recall, simultaneously.

2. A story understanding framework, which builds upon our procedural understanding model, to enable story understanding with explicit and explainable understanding of event procedures, captured through entity precondition and effect states.

3. An extensive evaluation of CGLI against strong baselines on a procedural task, `ProPara` (Dalvi et al., 2018), and recent story understanding task, `TRIP` (Storks et al., 2021). Our experiments show the positive impact of our method, through achieving state-of-the-art results, while ablation studies measure the impact of its individual components.

2 Task Definition

Procedural text understanding. The task input consists of an n-sentence paragraph $P = \{s_1, s_2, ...s_n\}$, and k entities $\{E_1, E_2, ...E_k\}$. The goal is to predict precondition state $S^p_{i,t}$ and effect state $S^e_{i,t}$, for every entity at every step, as well as the action $A_{i,t}$ performed by the entity at every step; $i \in \{1, 2, ..k\}$, $t \in \{1, 2, ...n\}$. The effect state at $t-1$ is the same as precondition state at step t, i.e., $S^e_{i,t-1} = S^p_{i,t}$, hence S_i is a sequence of length $n+1$. Following prior work (Mishra et al., 2018), $A_{i,t} \in$ {Create, Exist, Move, Destroy}, $S^e_{i,t} \in$ {*non-existence*, *unknown-location*, *location*}, and for *location*, a span of text in P needs be identified for the prediction. Action $A_{i,t}$ describes the entity state changes from precondition to effect, thus it can be inferred from the state sequence S_i, and vice versa—e.g., if $S^p_{i,1} =$ *non-existence* and $S^e_{i,1} =$ *location*, then $A_{i,1} =$ Create.

Procedural story understanding. The input to the procedural story understanding task consists of two parallel stories, P_1, $P_2 = \{s_1, s_2, ...s_n\}$, each consisting of n sentences and differing only in one of the sentences. Following Storks et al. (2021), the task is to identify which story is more plausible, identify the conflicting pair of sentences (s_{c1} and s_{c2}) in the implausible story, and list the preconditions S^e_i and effects S^p_i of all entities at every step of a story. Here, multiple attributes are considered for precondition and effect states. Unlike in the procedural text understanding task, the story completion task does not require that the effect state at step $t-1$ should match the precondition state at step t, i.e., S^e_{it-1} and S^p_{it} are not necessarily equal.

3 CGLI: Coalescing Global and Local Information

In this section, we describe the input representation, the architecture, and the training details of our model, as illustrated in Figure 2.

Input representation. To allow greater modeling flexibility and enable span extraction for entity location-prediction, we build a unique input representation for every entity at each step (*local view*), and we provide it access to the entire context (*global view*). Given an entity, we create a pseudo question Q 'where is {entity}' (*entity-aware*), and concatenate it with the full paragraph P, resulting in C = [CLS] Q [SEP] P [SEP]. We map C using the embedding layer of a language model (LM), resulting in C_{emb}. We then combine C_{emb} with timestep embeddings to mark the current step of interest (*timestep-aware*), following (Rajaby Faghihi and Kordjamshidi, 2021). In particular, each input token is assigned a timestep ID where {0=padding, 1=past, 2=current, 3=future}, forming $T \in \mathbb{R}^m$, where m is the number of tokens. The timestep sequence is projected through another embedding layer $Timestep \in \mathbb{R}^{4\times d}$. The sum of C_{emb} and $Timestep(T)$, denoted with $C'_{emb} \in \mathbb{R}^{d\times m}$, is then processed by the LM encoder layers, where d is the hidden layer dimension of the LM encoder. Formally:[1]

$$C_{emb} = \text{Embed}(C) \quad (1)$$
$$C'_{emb} = C_{emb} + Timestep(T) \quad (2)$$
$$C_{enc} = \text{LM Encoder}(C'_{emb}) \quad (3)$$

Location prediction. Given the LM encoded representation $C_{enc} \in \mathbb{R}^{d\times m}$, we extract the start and end indices of the location span:

$$P_{Start} = \text{Softmax}(W_s C_{enc}) \quad (4)$$
$$P_{End} = \text{Softmax}(W_e C_{enc}), \quad (5)$$

where $W_s, W_e \in \mathbb{R}^d$. For unknown locations and non-existing states, we extract the [CLS] token as the span, analogous to how unanswerable questions are usually handled (Rajpurkar et al., 2018).

In-batch Conditional Random Field. For entity state/action modeling, we jointly predict the entity actions across all steps (*global output*). We first group the encoded representation C^t_{enc} of the same entity at different time steps t in one batch chronologically, yield $C^N_{enc} \in \mathbb{R}^{d\times m\times(n+1)}$. Then we extract the [CLS] token embedding to represent the entity state of every step $C^{N'}_{enc} \in \mathbb{R}^{d\times(n+1)}$. We concatenate the entity state representation of every two consecutive steps to represent the actions between these two-state pairs. The result $D^N_{enc} \in \mathbb{R}^{2d\times n}$ is mapped to the emission scores $\phi \in \mathbb{R}^{a\times n}$, where a is the number of possible actions.

$$D^t_{enc} = \text{Concat}(C^{t'}_{enc}, C^{(t+1)'}_{enc}) \quad (6)$$
$$\phi = W_a^T(tanh(W_d^T D^N_{enc})) \quad (7)$$

where $W_d \in \mathbb{R}^{2d\times d}$, $W_a \in \mathbb{R}^{d\times a}$. The entity action sequence $A \in \mathbb{R}^n$ is modeled by a conditional random field (CRF):

$$P(A|\phi,\psi) \propto \exp(\sum_{t=1}^{n} \phi_t(A_t) + \psi(A_{t-1}, A_t)), \quad (8)$$

with the CRF layer's transition scores $\psi \in \mathbb{R}^{a\times a}$.

Prior initialization. Previous methods (Gupta and Durrett, 2019b; Zhang et al., 2021) initialize the CRF transition scores randomly and update them during training. This allows transition between any pair of actions. However, certain transitions between entity actions are nonsensical, e.g., an entity cannot be destroyed if it has not been created, and a destroyed entity cannot move. Learning such constraints may be possible if we have sufficient data, which is not the case for the tasks we are considering. Thus, we propose to directly impose commonsense constraints on the model's transition scores, because these conditions are universally true and can be used to reduce the model's search space. Specifically, we set an entity action transition score to *-inf* if it has not been seen in the training data, otherwise we estimate the initial score of a transition based on its frequency in the training data: $\psi^{uv} = log(\frac{Num(u,v)}{Num(u)})$, where ψ^{uv} is the log probability of transition from action u to action v, $Num(u,v)$ is the transition count from u to v in data, $Num(u)$ is the count of u in data.

Training and inference. We jointly optimize the location and the entity action prediction losses during training:

$$\mathcal{L}_{loc} = -\frac{1}{n}\sum_{n}^{t=0}(log(P^{y^t_s}_{Start}) + log(P^{y^t_e}_{End})) \quad (9)$$
$$\mathcal{L}_{action} = -log(P(A|\phi,\psi)) \quad (10)$$
$$\mathcal{L} = \mathcal{L}_{loc} + \mathcal{L}_{action}, \quad (11)$$

[1] To model the precondition state of step 1, we also build an input sequence for step 0.

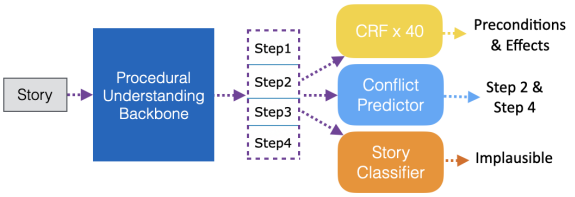

Figure 3: An illustration of integrating CGLI into a story understanding framework. The story is encoded in the same way as shown in Figure 2, producing a sequence of step representations, i.e., a batch of [CLS] vectors. These vectors serve as input to different output layers to model the three task objectives: plausibility (orange), conflict sentence detection (blue), and entity state prediction (yellow).

where y_s^t and y_e^t are the ground-truth start and end indices at step t. During inference, we use Viterbi decoding to produce the most likely entity action sequence and use the span extractor for the most likely location at every step. We combine the action sequence and location predictions to deterministically infer all precondition and effect states.

Data augmentation. Procedural text understanding requires dense annotation of entity states per step, making it challenging and expensive to collect large data. To address data sparsity, we propose a data augmentation method that could effectively leverage the unannotated paragraphs to enhance model's performance. In particular, we first train a model on the gold training set and then apply it to label the unannotated paragraphs, resulting a set of noisy examples. We then mix these examples with gold training data to train a second model.

4 Story Understanding with CGLI

We integrate CGLI into a story understanding framework with minimum modifications following the task definition, and the overall model is shown in Figure 3. As the story understanding tasks do not require location extraction, we remove the span extraction layer, which makes the input representation of step 0 obsolete. Given that the continuity of effects to preconditions between consecutive steps does not hold in this task, we directly use $C_{enc}^{N'} \in \mathbb{R}^{d \times n}$ instead of $D_{enc}^{N} \in \mathbb{R}^{2d \times n}$ in the in-batch CRF. Given B number of attributes for precondition and effect states, we apply an in-batch CRF module for each attribute. Specifically, we apply equations 7 and 8 for every attribute, yielding $2B$ such modules in total. To detect conflicting sentences, we concatenate every pair of sentence representations, and pass it through a linear layer to find the conflicting pair. For story classification, we take the mean of sentence representations for story representation, and pass it through a linear layer for binary classification. Formally,

$$C_{confl} = \text{vstack}(\text{Concat}(C_{enc}^{t'}, C_{enc}^{j'})) \quad (12)$$

$$P_{confl} = \text{Softmax}(W_{confl} C_{confl}) \quad (13)$$

$$C_{plau} = \text{Mean}(C_{enc}^{N'}) \quad (14)$$

$$P_{plau} = \text{Softmax}(W_{plau}^{T} C_{plau}), \quad (15)$$

where $C_{confl} \in \mathbb{R}^{2d \times \frac{n(n-1)}{2}}$, $j \in \{t+1, ...n\}$, $W_{confl} \in \mathbb{R}^{2d}$, $C_{plau} \in \mathbb{R}^{d}$, $W_{plau} \in \mathbb{R}^{d \times 2}$. During training, we jointly optimize all three task objectives:

$$\mathcal{L}_{plau} = -log(P_{plau}^{y_p}) \quad (16)$$

$$\mathcal{L}_{confl} = \begin{cases} -log(P_{confl}^{y_c}) & \text{if } y_p = 0 \\ 0 & otherwise \end{cases} \quad (17)$$

$$\mathcal{L}_{att} = -log(P(S^p|\phi^p, \psi^p)) - log(P(S^e|\phi^e, \psi^e)) \quad (18)$$

$$\mathcal{L} = \mathcal{L}_{plau} + \mathcal{L}_{confl} + \frac{1}{B} \sum_{B}^{b=0} \mathcal{L}_{att}^{b} \quad (19)$$

where y_p=0 if the story is not plausible and y_p=1 if the story is plausible, and y_c denotes the conflict sentence pair index. Note that in our setup, each entity can produce a prediction for conflict sentence pair and story plausibility. At inference time, we take the average of all entities' logits to get the final predictions for these two objectives.

5 Experimental Setup

Benchmarks. We evaluate procedural understanding on `ProPara` (Mishra et al., 2018)[2], which contains 488 human-written paragraphs from the natural science domain. The paragraphs are densely annotated by crowd workers, i.e., for every entity, its existence and location are annotated for every step. Additional 871 unannotated paragraphs are also provided by ProPara. We use these for data augmentation.

We test story understanding on `TRIP` (Storks et al., 2021), which contains crowdsourced plausible and implausible story pairs. In each pair, the plausible story label and the conflicting sentence pair label in implausible story are annotated. TRIP annotates 20 attributes with predefined set of possible values. The annotations are given for all entities at every timestep of the two stories.

[2] `ProPara` is covered under Apache 2.0 License.

Table 1: ProPara test set results. Modeling: E=entity, T=timestep-specific, GC=global context, GO=global outputs.

	Modeling				Sentence-level					Document-level		
Model	E	T	GC	GO	Cat1	Cat2	Cat3	Macro^{avg}	Micro^{avg}	P	R	F1
ProLocal (Dalvi et al., 2018)	Y	Y	N	N	62.7	30.5	10.4	34.5	34.0	**81.7**	36.8	50.7
ProGlobal (Dalvi et al., 2018)	Y	Y	Y	N	63.0	36.4	35.9	45.1	45.4	61.7	48.8	51.9
ProStruct (Tandon et al., 2018)	Y	Y	N	Y	-	-	-	-	-	74.3	43.0	54.5
KG-MRC (Das et al., 2018)	N	Y	N	N	62.9	40.0	38.2	47.0	46.6	64.5	50.7	56.8
NCET (Gupta and Durrett)	N	N	Y	Y	73.7	47.1	41.0	53.9	54.0	67.1	58.5	62.5
IEN (Tang et al., 2020)	N	N	Y	Y	71.8	47.6	40.5	53.3	53.0	69.8	56.3	62.3
DynaPro (Amini et al., 2020)	Y	Y	N	N	72.4	49.3	44.5	55.4	55.5	75.2	58.0	65.5
TSLM (2021)	Y	Y	Y	N	78.8	56.8	40.9	58.8	58.4	68.4	68.9	68.6
KOALA (Zhang et al., 2021)	N	N	Y	Y	78.5	53.3	41.3	57.7	57.5	77.7	64.4	70.4
CGLI (Ours)	Y	Y	Y	Y	80.3	60.5	**48.3**	**63.0**	**62.7**	74.9	**70.0**	72.4
CGLI (Ours) + Data Augmentation	Y	Y	Y	Y	**80.8**	**60.7**	46.8	62.8	62.4	75.7	**70.0**	**72.7**

Table 2: Statistics of the datasets.

Dataset	Stats	Train	Dev	Test
ProPara	#Paragraphs	391	43	54
	#Ents/Para	3.8	4.1	4.4
	#Sents/Para	6.7	6.7	6.9
TRIP	#Paragraphs	1169	474	504
	#Ents/Para	7.0	8.1	8.3
	#Sents/Para	5.1	5.0	5.1

We provide datasets splits details in Table 2. For TRIP, we only report the unique story statistics in Table 2. Note that Storks et al. (2021) have upsampled some of the plausible stories to match the number of implausible stories.

Evaluation metrics. Following previous work, we report both sentence-level metrics[3] and document-level metrics[4] on ProPara. Sentence-level evaluation computes accuracy over three questions: whether the entity created (moved/destroyed) in the process (*Cat1*), and if so, when (*Cat2*) and where (*Cat3*).[5] Document-level metrics compute F1 scores of the identified inputs (entities that exist before the process begins and are destroyed in the process), outputs (entities that do not exist before but exist after the process), conversions (instances where some entities are converted to other entities), and moves (location changes of entities).

For TRIP, we follow the original work and report the following metrics: *accuracy* of classifying the plausible story, *consistency* of finding the conflicting sentence pairs when the story classification is correct, and *verifiability*, which evaluates the prediction of the entities' effects at s_{c1} and the entities' preconditions at s_{c2}. We also report the average F1-score for preconditions and effects across the 20 attributes to better understand the model's procedural understanding ability.

Baselines. For ProPara, we directly report baseline results from the official leaderboard. For TRIP, we report the results from the best model released by Storks et al. (2021).

Training details. For ProPara, we define two additional action types to represent the entity transitions, namely Out-of-Create, Out-of-Destroy similar to (Zhang et al., 2021). Hence, the total size of the action space is six. For evaluation, these two types would be mapped to NONE transition, and they are defined to help the model differentiate the NONE types during training, i.e., if the entity has not being created or if it has been destroyed. To facilitate model's learning on location predictions, we initialized our model with a RoBERTa-Large (Liu et al., 2019) model pretrained on SQuAD 2.0 (Rajpurkar et al., 2018). We run our model five times with different random seeds and report the maximum scores in Table 1 and average scores with a 95% confidence interval in Table 3 and Table 4. For TRIP, we directly initialize the model with RoBERTa-Large. On ProPara we train models for 20 epochs and 6 epochs with data augmentation to let the model receive the similar number of updates. We train models for 10 epochs on TRIP. Except for training epochs, we use the same set of hyperparameters in all of our experiments: learning rate 1e-5, batch size 1, gradient accumulation 2. We used Transformers (Wolf et al., 2020) library [6] for all of our experiments and all of our models

[3] https://github.com/allenai/propara/tree/master/propara/evaluation

[4] https://github.com/allenai/aristo-leaderboard/tree/master/propara

[5] Cat2 and Cat3 only apply to entities that satisfy Cat1.

[6] Covered under Apache 2.0 License.

have about 360M parameters.

Computing infrastructure. We run our experiments on a single Nvidia A6000 GPU or a single Nvidia Titan RTX GPU. For ProPara, each experiment takes about 1.5 hours to finish. For TRIP, each experiment takes about 9 hours to finish.

6 Results and Analysis

6.1 Procedural text understanding

CGLI significantly outperforms all previous baselines on ProPara, achieving state-of-the-art results (Table 1). With data augmentation, our model achieves further improvement on document level. For each baseline, we indicate whether it considers entity-centric information (E), timestep-centric (T), global context (GC), and global output (GO). We note that models that adopt global output usually have much higher precision than recall on document level. On the other hand, TSLM is very good on recall, which is expected given its focus on entity and timestep input modeling.[7] CGLI is able to achieve both strong precision and recall, showing the benefit of global reasoning over both entity- and timestep-specific global inputs in a single model.

We break down the results on ProPara by the document-level question types defined in §5 and compare our best model with the best results reported by TSLM and KOALA. The precision and recall per question type are shown in Figure 4. Consistent with the overall results, KOALA is particularly strong on precision for all types and TSLM is much better on recall. CGLI is able to maintain a balance between those two extremes and achieve overall better results. All three models perform similarly when predicting the inputs and the outputs of a procedure. Yet, CGLI achieves much higher performance on transitional questions regarding entity conversions and moves, which are notably harder to predict. These results suggest that the gains of CGLI over previous works are mostly due to hard-to-answer categories.

6.2 Story understanding

Our method outperforms the baseline method on the TRIP dataset by a very large margin on all metrics, especially on consistency where we observe nearly 400% relative improvement over the baseline (Table 4). This may seem surprising as both our model and the baseline use the same LM backbone. After further analysis of the baseline model, we notice three sub-optimal design decisions. First, the baseline detects conflicting sentence pairs via binary classification for every sentence, independently, without considering pairs of sentences. As a result, for 47.6% of examples in TRIP test set, the baseline model predicted either less or more than two sentences as conflicting, thus getting a score of 0 on consistency. Second, the baseline uses the same encoded representations to directly model both story classification and conflicting pair detection objectives. Without using task-specific output projection layers, the model may be hard to optimize. Third, the baseline did not provide global input view to the model, i.e., each sentence is encoded independently.

6.3 Ablation studies

Impact of modeling aspects To understand the contribution of each of the four modeling aspects we identified for the procedural text understanding, we ablate each of them in CGLI.

No GO is done by removing the CRF layer and directly training the model with cross-entropy loss over the emission probability $\phi \in \mathbb{R}^{n \times a}$ defined in §3. During inference, we predict the action at each timestep independently by taking the argmax over the emission probability instead of viterbi decoding. **No GC** is achieved by allowing the model model to access up to t sentences at every timestep $t \in \{1, 2, 3, ...n\}$, i.e. the model has no access to future sentences. For **No T**, we remove the timestep embeddings such that each entity would have identical encoded context representations across timesteps. For **No E**, we no longer provide the pseudo question with the entity name in the input §3, such that all entities in the same paragraph would have the same encoded context representations.

The results are shown in the bottom half of Table 3. Removing either T or E leads to drastic drop in the F1 score. This is not surprising because the model would have no clue how to distinguish different timesteps or different entities, respectively. We found that the model predict most of entity actions to be NONE, leading to extremely high precision and low recall. Removing GO also leads to a large drop in F1 score, which is actually similar to TSLM's performance, a model that lacks GO. This shows that modeling the global dependency

[7] This pattern may not always hold for other models due to other modeling differences, e.g., LSTM vs. BERT.

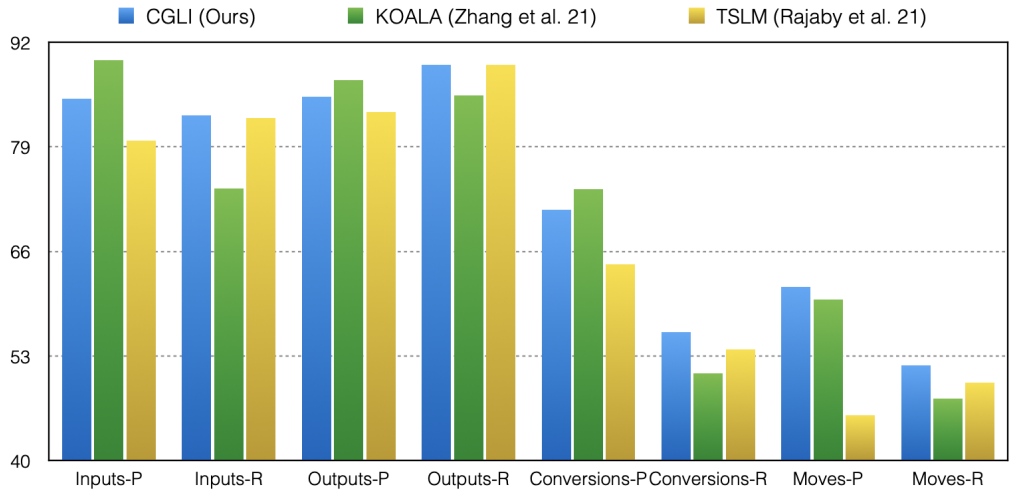

Figure 4: Document-level evaluation on ProPara test set, split by precision (P) and recall (R) per category (Inputs, Outputs, Conversions, Moves).

Table 3: Document-level ablation results of proposed model components and modeling aspects on the ProPara.

	Dev set			Test set		
Model	P	R	F1	P	R	F1
CGLI + Data Augmentation	78.5(±1.7)	76.1(±0.8)	**77.3**(±0.8)	75.2(±1.1)	68.8(±0.8)	**71.9**(±0.5)
CGLI	77.3(±1.5)	75.5(±0.7)	76.4(±1.0)	73.0(±1.9)	**69.8**(±1.2)	71.3(±0.9)
No SQuAD2.0	76.5(±1.3)	75.4(±0.9)	75.9(±0.4)	72.5(±2.7)	68.0(±1.3)	70.1(±0.8)
No Prior	75.6(±0.8)	**76.6**(±0.6)	76.1(±0.3)	72.0(±2.1)	68.1(±1.4)	70.0(±1.3)
No GO	75.7(±1.1)	76.1(±1.4)	75.9(±0.5)	70.2(±1.2)	67.3(±1.2)	68.7(±0.8)
No GC	75.5(±1.3)	73.2(±1.0)	74.3(±0.5)	73.2(±2.2)	66.7(±0.6)	69.8(±1.1)
No T	82.3(±0.7)	59.7(±0.4)	69.2(±0.3)	77.2(±1.3)	54.3(±1.0)	63.8(±0.8)
No E	**84.5**(±1.1)	48.6(±0.3)	61.7(±0.2)	**84.9**(±0.7)	40.8(±0.5)	55.1(±0.3)

Table 4: Results on the TRIP dataset. The F1 scores of last two columns are Macro averages of 20 attributes.

Model	Accuracy	Consistency	Verifiability	Precondition F1	Effect F1
TRIP-RoBERTa (Storks et al., 2021)	73.2	19.1	9.1	51.3	49.3
CGLI (Ours)	93.4(±1.5)	76.3(±1.7)	24.8(±1.6)	70.8(±1.8)	74.9(±1.7)
CGLI (Ours) No CRF	**94.1**(±0.7)	**77.3**(±1.0)	**28.0**(±2.5)	**72.1**(±1.6)	**75.6**(±1.6)

is important for procedural understanding. Finally, removing GC also hurts the performance, which is also expected because location spans often only appear in future sentences, thus span extraction layer is at disadvantage in this setting.

Impact of training data To understand the impact of the CGLI components, we ablate SQuAD2.0 pretraining by initializing the model with vanilla RoBERTa-Large model and we ablate prior initialization by randomly initializing the transition probabilities in the CRF layer. The results (upper half of Table 3) show that with data augmentation, CGLI achieves higher overall F1 scores on average and the gains are mostly from precision. Both pretraining on SQuAD2.0 and prior initialization have a positive impact on the CGLI performance.

As the continuity from effect to precondition states no longer holds on the TRIP story understanding task (cf. §2), we investigate the impact of the CRF layers on modeling entity states. We remove the CRF layers for both effects and preconditions, and we directly train CGLI with regular classification objectives, hence entity states at each step are predicted independently (No GO). Table 4 shows that removing CRF improves performance. We hypothesize that this is caused by the implausible stories in the dataset. Since the entity states in the implausible story's conflicting sentences are inconsistent by nature, training the CRF to maximize their probabilities can be confusing for the model. To verify this, we train models with and without CRF on plausible stories only. In this case, the model is only trained to predict entities effects and preconditions. We found that the models have very

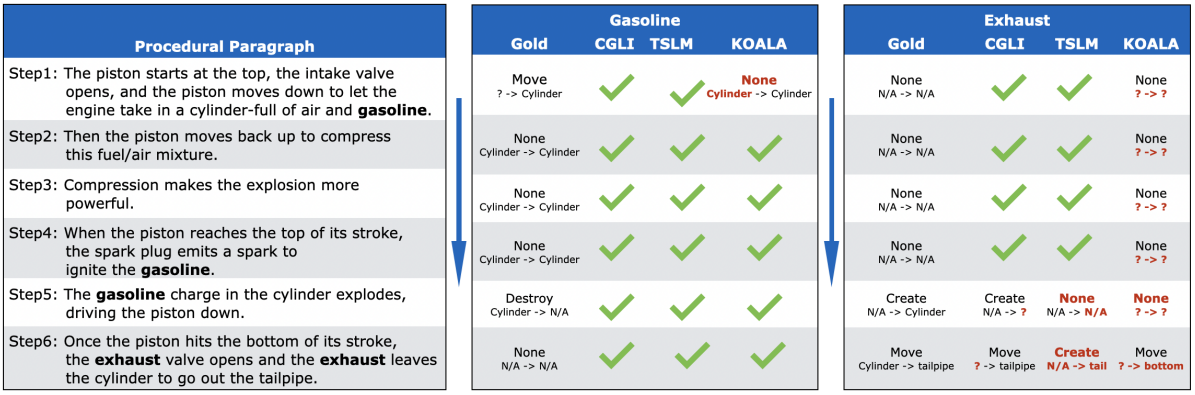

Figure 5: Example predictions on ProPara from three models for two entities. Red font indicate errors.

Table 5: Error Examples on TRIP. The conflicting pairs are marked with *, and the entity of interest with *italic*.

Ann washed her hair in the bathtub.
Ann used the hair dryer to get ready to go out.
Ann applied deodorant to her armpits.
**Ann* put her pants on.
- (Effects, is wet), Pred: False, Gold: Irrelevant
*Ann ironed her *pants* before going out.
- (Preconditions, is wet), Pred: True, Gold: Irrelevant
*John forgot his *notebook* at home.
- (Effects, location), Pred: Moved, Gold: Irrelevant
John sat at his desk.
John opened up his book bag.
* John took out his *notebook*.
- (Preconditions, location),
- Pred: Picked up, Gold: Taken out of container
John began writing down notes.

similar F1 scores with or without CRF (preconditions 74.1 vs 73.7, effects 76.5 vs 76.6). Thus, we conclude that implausible stories are detrimental to CRF training. Moreover, as the effects of the previous step are not a precondition of the current step on TRIP, the outputs from previous steps can hardly contribute to the current prediction, thus CRF has a limited contribution even on the plausible stories.

6.4 Case Studies

We show an example of tracking states for two entities from ProPara with partial outputs from CGLI, TSLM, and KOALA in Figure 5. For gasoline, our model and TSLM both got perfect predictions, but KOALA missed the action at step 1, thus predicting no moves across the process. For exhaust, the sentence in step 6 gives a strong signal for a movement, however, there is no mention of exhaust in the previous steps. Our model is able to infer that *create* needs to come before *move*, thus correctly predicting the actions in steps 5 and 6. However, since TSLM does not have the global output view, it cannot capture such transitions. For KOALA, although it is also able to predict the move and infer that the exhaust should exist before the move, it is unable to predict the create action. We note that for both entities, KOALA is more reluctant to predict actions compared to the other two models. These observations explain why KOALA achieves overall higher precision but lower recall.

We show story reasoning examples from TRIP in Table 5. Since the largest gap in the model performance is between consistency and verifiability, we select examples where our model successfully predicted conflicting sentences but failed to predict entity states. We see that the model still lacks common sense on certain concepts, e.g., forgetting something at home does not result in changing its location, and people usually iron their clothes after they are dry. We also note that some entity states might be hard to distinguish, e.g., the distinction between picking up something versus taking something out of a container only depends on previous location of the object, which might be hard for models to learn from data. These observations suggest that enhancing the model's commonsense reasoning ability is a promising future direction.

7 Related Work

Recent **procedural text understanding** benchmarks including ScoNe (Long et al., 2016), bAbI (Weston et al., 2015), ProcessBank (Berant et al., 2014), ProPara (Mishra et al., 2018), Recipe (Bosselut et al., 2018), and OpenPI (Tandon et al., 2020) have inspired a series of methods. Mishra et al. (2018) propose ProLocal that encodes each step of a procedure separately and ProGlobal that

encodes the full paragraph at every step. KG-MRC (Das et al., 2018) builds a dynamic knowledge graph of entity and location mentions to communicate across time steps. DynaPro (Amini et al., 2020) employs pre-trained LM to jointly predict entity attributes and their transitions. TSLM (Rajaby Faghihi and Kordjamshidi, 2021) formulates procedural understanding as a question answering task, and leverages models pretrained on SQuAD (Rajpurkar et al., 2016) enhanced with a timestamp encoding. Although equipped with various ways to pass information across time steps, these methods still make local predictions thus they may compromise the global dependency of outputs. Another line of work focuses on jointly modeling the entity action sequence, aiming to ensure global structure and consistency. ProStruct (Tandon et al., 2018) aims to find the globally optimal entity action sequence using beam search. Gupta and Durrett (2019b) devise a structured neural architecture NCET, modeled with a CRF, which recurrently updates the hidden representation of each entity at each step. IEN (Tang et al., 2020) builds upon NCET and augments the entity-to-entity attention. KOALA (Zhang et al., 2021) further enhances the NCET by pretraining on Wikipedia and ConceptNet (Speer et al., 2017). The key shortcoming of these global methods is that they rely on entity mentions extracted from a single copy of encoded context shared by all entities and all steps, which limits their modeling capacity. Our proposed method stands out from all previous works by coalescing complementary granularities of procedural text modeling, by building specific and informative input representations while modeling output dependency. Concurrent to our work, Shi et al. (2022) proposed LEMON for language-based environment manipulation. Their focus on model pretraining is orthogonal to CGLI.

There are also numerous recent **story understanding** benchmarks (Mostafazadeh et al., 2016; Qin et al., 2019; Mostafazadeh et al., 2020), and modeling methods (Qin et al., 2020; Guan et al., 2020; Gabriel et al., 2021; Ghosal et al., 2021). The TRIP task (Storks et al., 2021) integrates a procedural understanding component in story understanding to enable consistent and interpretable reasoning over narratives. To our knowledge, we are the first work to bridge the gap of modeling methods between procedural understanding and story comprehension. Other tasks that require reasoning over procedures exist, including defeasible reasoning (Rudinger et al., 2020; Madaan et al., 2021), abductive commonsense inference (Bhagavatula et al., 2019), reasoning over preconditions (Qasemi et al., 2021), script reasoning (Zhang et al., 2020; Sakaguchi et al., 2021) and multimodal script reasoning (Yang et al., 2021; Wu et al., 2021), are typically solved by specialized methods, without separately modeling procedural and causal links. We intend to apply CGLI on these tasks in the future to bridge this gap.

8 Conclusions & Future Work

We proposed CGLI: a novel procedural understanding method that combined global and local information. Recognizing the key role of procedural understanding in downstream tasks, we also integrated CGLI in a story understanding framework. Our experiments showed the benefit of our coalesced method, with the global views providing optimal precision, while the local view boosting its recall, ultimately achieving new state-of-the-art results. We demonstrated that CGLI can help with classifying stories, and identifying the conflicting sentence for inconsistent stories. Future work should investigate how to enhance the commonsense ability of our procedural understanding model, e.g., by injecting commonsense knowledge during finetuning (Chen et al., 2018; Ma et al., 2019) or by pretraining on commonsense knowledge bases (Guan et al., 2020; Ilievski et al., 2021; Ma et al., 2020), and how to apply procedural understanding to other downstream tasks, such as dialogue modelling (Zhou et al., 2021) and planning (Shridhar et al., 2020). Also, it's worth exploring the lightweight-tuning methods (Ma et al., 2021; Vu et al., 2022) to enhance the model's generalization and reduce computation cost.

Acknowledgements

We would like to thank Yonatan Bisk, Aman Madaan, and Ruohong Zhang for helpful discussions and the anonymous reviewers for their valuable suggestions on this paper. Some datasets and models are used by this work, despite their not having specified licenses in their code repositories—for these, we followed previous works and only used them for pure research purposes.

References

Aida Amini, Antoine Bosselut, Bhavana Dalvi Mishra, Yejin Choi, and Hannaneh Hajishirzi. 2020. Procedural reading comprehension with attribute-aware context flow. *CoRR*, abs/2003.13878.

Jonathan Berant, Vivek Srikumar, Pei-Chun Chen, Abby Vander Linden, Brittany Harding, Brad Huang, Peter Clark, and Christopher D. Manning. 2014. Modeling biological processes for reading comprehension. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1499–1510, Doha, Qatar. Association for Computational Linguistics.

Chandra Bhagavatula, Ronan Le Bras, Chaitanya Malaviya, Keisuke Sakaguchi, Ari Holtzman, Hannah Rashkin, Doug Downey, Scott Wen tau Yih, and Yejin Choi. 2019. Abductive commonsense reasoning.

Antoine Bosselut, Omer Levy, Ari Holtzman, Corin Ennis, Dieter Fox, and Yejin Choi. 2018. Simulating action dynamics with neural process networks. In *Proceedings of the 6th International Conference for Learning Representations (ICLR)*.

Jiaao Chen, Jianshu Chen, and Zhou Yu. 2018. Incorporating structured commonsense knowledge in story completion.

Bhavana Dalvi, Lifu Huang, Niket Tandon, Wen-tau Yih, and Peter Clark. 2018. Tracking state changes in procedural text: a challenge dataset and models for process paragraph comprehension. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 1595–1604, New Orleans, Louisiana. Association for Computational Linguistics.

Rajarshi Das, Tsendsuren Munkhdalai, Xingdi Yuan, Adam Trischler, and Andrew McCallum. 2018. Building dynamic knowledge graphs from text using machine reading comprehension.

Saadia Gabriel, Chandra Bhagavatula, Vered Shwartz, Ronan Le Bras, Maxwell Forbes, and Yejin Choi. 2021. Paragraph-level commonsense transformers with recurrent memory. In *AAAI*.

Deepanway Ghosal, Navonil Majumder, Rada Mihalcea, and Soujanya Poria. 2021. STaCK: Sentence ordering with temporal commonsense knowledge. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 8676–8686, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Jian Guan, Fei Huang, Zhihao Zhao, Xiaoyan Zhu, and Minlie Huang. 2020. A knowledge-enhanced pretraining model for commonsense story generation. *Transactions of the Association for Computational Linguistics*, 8:93–108.

Aditya Gupta and Greg Durrett. 2019a. Effective use of transformer networks for entity tracking. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 759–769, Hong Kong, China. Association for Computational Linguistics.

Aditya Gupta and Greg Durrett. 2019b. Tracking discrete and continuous entity state for process understanding. In *Proceedings of the Third Workshop on Structured Prediction for NLP*, pages 7–12, Minneapolis, Minnesota. Association for Computational Linguistics.

Filip Ilievski, Jay Pujara, and Hanzhi Zhang. 2021. Story generation with commonsense knowledge graphs and axioms. In *Workshop on Commonsense Reasoning and Knowledge Bases*.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Roberta: A robustly optimized bert pretraining approach.

Reginald Long, Panupong Pasupat, and Percy Liang. 2016. Simpler context-dependent logical forms via model projections. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1456–1465, Berlin, Germany. Association for Computational Linguistics.

Kaixin Ma, Jonathan Francis, Quanyang Lu, Eric Nyberg, and Alessandro Oltramari. 2019. Towards generalizable neuro-symbolic systems for commonsense question answering. In *Proceedings of the First Workshop on Commonsense Inference in Natural Language Processing*, pages 22–32, Hong Kong, China. Association for Computational Linguistics.

Kaixin Ma, Filip Ilievski, Jonathan Francis, Yonatan Bisk, Eric Nyberg, and Alessandro Oltramari. 2020. Knowledge-driven data construction for zero-shot evaluation in commonsense question answering.

Kaixin Ma, Filip Ilievski, Jonathan Francis, Satoru Ozaki, Eric Nyberg, and Alessandro Oltramari. 2021. Exploring strategies for generalizable commonsense reasoning with pre-trained models. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 5474–5483, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Aman Madaan, Niket Tandon, Dheeraj Rajagopal, Peter Clark, Yiming Yang, and Eduard Hovy. 2021. Think about it! improving defeasible reasoning by first modeling the question scenario. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 6291–6310, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Bhavana Dalvi Mishra, Lifu Huang, Niket Tandon, Wen-tau Yih, and Peter Clark. 2018. Tracking state changes in procedural text: a challenge dataset and models for process paragraph comprehension. *arXiv preprint arXiv:1805.06975*.

Nasrin Mostafazadeh, Nathanael Chambers, Xiaodong He, Devi Parikh, Dhruv Batra, Lucy Vanderwende, Pushmeet Kohli, and James Allen. 2016. A corpus and cloze evaluation for deeper understanding of commonsense stories. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 839–849, San Diego, California. Association for Computational Linguistics.

Nasrin Mostafazadeh, Aditya Kalyanpur, Lori Moon, David Buchanan, Lauren Berkowitz, Or Biran, and Jennifer Chu-Carroll. 2020. GLUCOSE: GeneraLized and COntextualized story explanations. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 4569–4586, Online. Association for Computational Linguistics.

Ehsan Qasemi, Filip Ilievski, Muhao Chen, and Pedro Szekely. 2021. Corequisite: Circumstantial preconditions of common sense knowledge.

Lianhui Qin, Antoine Bosselut, Ari Holtzman, Chandra Bhagavatula, Elizabeth Clark, and Yejin Choi. 2019. Counterfactual story reasoning and generation. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 5043–5053, Hong Kong, China. Association for Computational Linguistics.

Lianhui Qin, Vered Shwartz, Peter West, Chandra Bhagavatula, Jena D. Hwang, Ronan Le Bras, Antoine Bosselut, and Yejin Choi. 2020. Back to the future: Unsupervised backprop-based decoding for counterfactual and abductive commonsense reasoning. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 794–805, Online. Association for Computational Linguistics.

Hossein Rajaby Faghihi and Parisa Kordjamshidi. 2021. Time-stamped language model: Teaching language models to understand the flow of events. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 4560–4570, Online. Association for Computational Linguistics.

Pranav Rajpurkar, Robin Jia, and Percy Liang. 2018. Know what you don't know: Unanswerable questions for SQuAD. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 784–789, Melbourne, Australia. Association for Computational Linguistics.

Pranav Rajpurkar, Jian Zhang, Konstantin Lopyrev, and Percy Liang. 2016. SQuAD: 100,000+ questions for machine comprehension of text. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 2383–2392, Austin, Texas. Association for Computational Linguistics.

Rachel Rudinger, Vered Shwartz, Jena D. Hwang, Chandra Bhagavatula, Maxwell Forbes, Ronan Le Bras, Noah A. Smith, and Yejin Choi. 2020. Thinking like a skeptic: Defeasible inference in natural language. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 4661–4675, Online. Association for Computational Linguistics.

Keisuke Sakaguchi, Chandra Bhagavatula, Ronan Le Bras, Niket Tandon, Peter Clark, and Yejin Choi. 2021. proScript: Partially ordered scripts generation. In *Findings of the Association for Computational Linguistics: EMNLP 2021*, pages 2138–2149, Punta Cana, Dominican Republic. Association for Computational Linguistics.

Qi Shi, Qian Liu, Bei Chen, Yu Zhang, Ting Liu, and Jian-Guang Lou. 2022. Lemon: Language-based environment manipulation via execution-guided pre-training.

Mohit Shridhar, Jesse Thomason, Daniel Gordon, Yonatan Bisk, Winson Han, Roozbeh Mottaghi, Luke Zettlemoyer, and Dieter Fox. 2020. Alfred: A benchmark for interpreting grounded instructions for everyday tasks. *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*.

Robyn Speer, Joshua Chin, and Catherine Havasi. 2017. Conceptnet 5.5: An open multilingual graph of general knowledge. In *Thirty-first AAAI conference on artificial intelligence*.

Shane Storks, Qiaozi Gao, Yichi Zhang, and Joyce Chai. 2021. Tiered reasoning for intuitive physics: Toward verifiable commonsense language understanding. In *Findings of the Association for Computational Linguistics: EMNLP 2021*, pages 4902–4918, Punta Cana, Dominican Republic. Association for Computational Linguistics.

Niket Tandon, Bhavana Dalvi, Joel Grus, Wen-tau Yih, Antoine Bosselut, and Peter Clark. 2018. Reasoning about actions and state changes by injecting commonsense knowledge. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 57–66, Brussels, Belgium. Association for Computational Linguistics.

Niket Tandon, Keisuke Sakaguchi, Bhavana Dalvi, Dheeraj Rajagopal, Peter Clark, Michal Guerquin, Kyle Richardson, and Eduard Hovy. 2020. A dataset for tracking entities in open domain procedural text. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 6408–6417, Online. Association for Computational Linguistics.

Jizhi Tang, Yansong Feng, and Dongyan Zhao. 2020. Understanding procedural text using interactive entity networks. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 7281–7290, Online. Association for Computational Linguistics.

Tu Vu, Brian Lester, Noah Constant, Rami Al-Rfou', and Daniel Cer. 2022. SPoT: Better frozen model adaptation through soft prompt transfer. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 5039–5059, Dublin, Ireland. Association for Computational Linguistics.

Jason Weston, Antoine Bordes, Sumit Chopra, Alexander M. Rush, Bart van Merriënboer, Armand Joulin, and Tomas Mikolov. 2015. Towards ai-complete question answering: A set of prerequisite toy tasks.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Remi Louf, Morgan Funtowicz, Joe Davison, Sam Shleifer, Patrick von Platen, Clara Ma, Yacine Jernite, Julien Plu, Canwen Xu, Teven Le Scao, Sylvain Gugger, Mariama Drame, Quentin Lhoest, and Alexander Rush. 2020. Transformers: State-of-the-art natural language processing. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 38–45, Online. Association for Computational Linguistics.

Te-Lin Wu, Alex Spangher, Pegah Alipoormolabashi, Marjorie Freedman, Ralph Weischedel, and Nanyun Peng. 2021. Understanding procedural knowledge by sequencing multimodal instructional manuals.

Yue Yang, Artemis Panagopoulou, Qing Lyu, Li Zhang, Mark Yatskar, and Chris Callison-Burch. 2021. Visual goal-step inference using wikiHow. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 2167–2179, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Li Zhang, Qing Lyu, and Chris Callison-Burch. 2020. Reasoning about goals, steps, and temporal ordering with WikiHow. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 4630–4639, Online. Association for Computational Linguistics.

Zhihan Zhang, Xiubo Geng, Tao Qin, Yunfang Wu, and Daxin Jiang. 2021. Knowledge-aware procedural text understanding with multi-stage training. In *WWW '21: The Web Conference 2021, Ljubljana, Slovenia, April 19–23, 2021*.

Pei Zhou, Pegah Jandaghi, Hyundong Cho, Bill Yuchen Lin, Jay Pujara, and Xiang Ren. 2021. Probing commonsense explanation in dialogue response generation. In *Findings of the Association for Computational Linguistics: EMNLP 2021*, pages 4132–4146, Punta Cana, Dominican Republic. Association for Computational Linguistics.

Original Content Is All You Need! An Empirical Study on Leveraging Answer Summary for WikiHowQA Answer Selection Task

Liang Wen[1,2] **Juan Li**[4] **Houfeng Wang**[1] **Yingwei Luo**[1,2]
Xiaolin Wang[1,2] **Xiaodong Zhang**[3] **Zhicong Cheng**[3] **Dawei Yin**[3]
[1]School of Computer Science, Peking University, China
[2]Peng Cheng Laboratory, Shenzhen, China [3]Baidu Inc., Beijing, China
[4]School of Chinese Language and Literature, Beijing Normal University, China
{yuco,wanghf,lyw,wxl}@pku.edu.cn {lijuan}@mail.bnu.edu.cn
{zhangxiaodong11,chengzhicong01}@baidu.com {yindawei}@acm.org

Abstract

Answer selection task requires finding appropriate answers to questions from informative but crowdsourced candidates. A key factor impeding its solution by current answer selection approaches is the redundancy and lengthiness issues of crowdsourced answers. Recently, Deng et al. (2020) constructed a new dataset, *WikiHowQA*, which contains a corresponding reference summary for each original lengthy answer. And their experiments show that leveraging the answer summaries helps to attend the essential information in original lengthy answers and improve the answer selection performance under certain circumstances. However, when given a question and a set of long candidate answers, human beings could effortlessly identify the correct answer without the aid of additional answer summaries since the original answers contain all the information volume that answer summaries contain. In addition, pretrained language models have been shown superior or comparable to human beings on many natural language processing tasks. Motivated by those, we design a series of neural models, either pretraining-based or non-pretraining-based, to check wether the additional answer summaries are helpful for ranking the relevancy degrees of question-answer pairs on WikiHowQA dataset. Extensive automated experiments and hand analysis show that the additional answer summaries are not useful for achieving the best performance.

1 Introduction

Answer selection task in community question answering (cQA) has been a popular research topic in both academy and industry due to its practical importance. In recent years, neural attention-based approaches for this task can be roughly categorized into two primary types. One type of them (Han et al., 2019; Rücklé et al., 2019) attempts to enhance the interactions of different granularity between question and candidate answer using the widely-adopted compare-aggregate framework (Wang and Jiang, 2017). The another focuses on incorporating additional input information, such as user metadata information (Xie et al., 2020), the subject-body relationship of community questions (Wu et al., 2018), etc. However, real-life cQA datasets that contain open-domain and non-factoid questions usually come along with long multi-sentence answer texts and noise. As a result, many previous neural answer selection approaches that were primarily designed to retrieve short answers fall short of expectations in such cases (Cohen et al., 2018; Rücklé et al., 2019).

Recently, Deng et al. (2020) propose to leverage answer summaries to tackle the redundancy and lengthiness issues of original answers in long answer selection task. To this end, they created *WikiHowQA*, the first large-scale open-domain cQA dataset that contains lengthy answers coupled with its summaries written by community users for non-factoid questions starting with "**How to**". Instead of relying on crowdsourcing, *WikiHowQA* was generated based on the WikiHow summarization dataset (Koupaee and Wang, 2018) and the online *WikiHow* knowledge base[1]. An example from the dataset is shown in Table 1. From it, we can see that the candidate answer details a method for decorating a school locker. And the answer summary well summarizes the key points of the answer. Based on this perspective, Deng et al. (2020) suggest that we could make use of answer summaries to alleviate the answer redundancy and noise issue in long answer selection task.

However, though answer summaries are always much shorter and more concise than the answer texts being summarized, they present text information in an abridged form and do not include details. Hence, the way that leverages the answer summaries to alleviate the answer redundancy and noise issues may also lead to the neglect of details

[1]http://www.wikihow.com

Proceedings of the 29th International Conference on Computational Linguistics, pages 1546–1555
October 12–17, 2022.

Question	How to decorate a school locker ?
Answer	Do you want your locker to be calm and relaxing , to relieve you from those stressful classes ? Or do you want your locker to be fun and exciting , colorful , or maybe you want it to show some of your rocker spirit. If you do n't want to express anything, pick a theme. A cool theme could be 70 's, giraffe 's, dogs ... (1112 words in total)
Summary	Think about the feeling you want to express. Acquire all of the materials you will need to make your locker look amazing. Hang up some pictures. Put a sign on the door ...
Label	1

Table 1: An example from the *WikiHowQA* dataset. Here we only list one candidate answer and its summary.

not covered by answer summaries. Besides, unlike user metadata information introduced by Xie et al. (2020) or subject-body relationship of community questions used by Wu et al. (2018), answer summaries are not supplementary to original answers and the utilization of them won't bring any additional information gain from an information entropy perspective. Last but not least, when given a question and a set of long candidate answers, Humans can easily figure out the correct answer without the need of additional answer summaries. Thus, it's unclear what role do answer summaries actually play in the WikiHowQA answer selection task and, indeed, whether it's beneficial to import additional answer summaries.

In this paper, we aim to conduct an in-depth and comprehensive analysis of this dataset and explore whether answer summaries could be helpful for "how-to" answer selection task. We demonstrate that, without the aid of answer summaries, simple, carefully designed LSTM-based models and pretraining-based models could obtain high, state-of-the-art MAP score of 72.91% and 82.74% on the dataset respectively. We carry out a meticulous qualitative analysis on randomly-sampled instances to provide data on their difficulty and quality, and whether the utilization of answer summaries could improve human performance. We conclude that: (i) This answer selection task is relatively easy though it contains long multi-sentence answer texts. (ii) Answer summaries do not convey additional information content and are not helpful for boosting both model and human performance. (iii) This dataset is noisy due to its method of data creation.

2 The Answer Selection Task

The *WikiHowQA* dataset introduced in (Deng et al., 2020) is made from an online wiki-style community website – WikiHow[2]. The questions contained in the dataset are all non-factoid and start with "**How to**". For a specific question, its accepted answers are considered as correct, and negative candidates were collected by retrieving the accepted answers to relevant questions. Each answer written by community users details multiple steps of doing a procedural task for a specific how-to question. In addition, every candidate answer is associated with a short reference summary. To be specific, the answer selection task could be formally defined as follows:

Given a question q_i and a set of lengthy candidate answers $A_i = \left\{a_i^{(1)}, \ldots, a_i^{(K)}\right\}$, the goal is to select all correct answers from the candidate answer set. In the training stage, for each candidate answer $a_i^{(k)}$, a corresponding reference summary $s_i^{(k)}$ and a label $y_i^{(k)}$ that denotes whether the answer $a_i^{(k)}$ can answer the question q_i are provided. However, during the testing procedures, the relevancy degree of question-answer pairs must be measured without access to their answer summaries. That is to say, answer summary is only accessible in the training stage. The reason we follow the constraint is that we want to make a fair comparison with previous methods[3].

Table 2 provides the detailed statistics of the dataset. From it, we can see that the answer texts is extremely long (with an average answer length of more than 520 words). Whereas, the well-known answer selection dataset InsuranceQA introduced by Feng et al. (2015) only has an average answer length of 112 words. Even in the recent *Long Answer Selection* (LSA) benchmark introduced by Rücklé et al. (2019), which are featured in con-

[2]http://www.wikihow.com

[3]The *WikiHowQA* dataset could also be used as an answer summary generation benchmark.

Statistics	*WikiHowQA*		
	Train	Dev	Test
Questions	76,687	8,000	22,354
QA pairs	904,460	72,474	211,255
Avg Qlen	7.20	6.84	6.69
Avg Alen	520.87	548.26	554.66
Avg Slen	67.38	61.84	74.42

Table 2: Statistics of the *WikiHowQA* Dataset. The average question length (Avg Qlen) is the average number of tokens in a question. The same applies to answer and summary.

taining long multi-sentence answer texts, we could only observe an average answer length of less than 290 words.

3 Methods

In this section, we describe two types of methods we implemented. The first class of methods directly models the relevancy degrees of question-answer pairs without using any answer summaries information. The second class of methods is partially inspired by Deng et al. (2020). They jointly learn answer selection and answer summary generation so as to leverage short answer summary to aid in picking out long multi-sentence answer. Different from them, we explicitly exploit the relevancy degrees of question-summary pair to aid in modeling the interaction between question and answer. For each class of methods, we from one side build our model based on bidirectional LSTMs so as to make a fair comparison with previous approaches and from the other we build our model based on a pretrained language model - ALBERT (Lan et al., 2020) to advance the state of the art.

3.1 LSTM-ASM

In this subsection, we aim at building a neural model that scores each answer in a pool of candidate answers according to its relevancy in regard to the given question. Our model adopts bidirectional LSTMs as text encoders, and we name it as LSTM-based Answer Selection Model(LSTM-ASM). The framework can be described in the following steps.

Given a question $q = \{w_q^1, \ldots, w_q^n\}$, an answer candidate $a = \{w_a^1, \ldots, w_a^m\}$ and the corresponding label y, we first map each word to its embedding. Then, the question embeddings E_q and the answer embeddings E_a are fed into a pair of Bi-LSTM encoders to generate contextual embeddings $\hat{E}_q$, $\hat{E}_a$ respectively.

Next, to capture the interactions between all aspects of question q and answer a, we feed the contextual embeddings $\hat{E}_q$ and $\hat{E}_a$ into a matching layer. Here, the matching Layer mainly define four different multi-perspective matching operations: Full-Matching, Maxpooling-Matching, Attentive-Matching, and Max-Attentive-Matching. Each matching operation describes a way to match each time-step of $\hat{E}_q$ against all time-steps of $\hat{E}_a$ and match each time-step of $\hat{E}_a$ against all time-steps of $\hat{E}_q$. We define the four matching operations as introduced by (Wang et al., 2017) and refer readers to it for details.

After applying a matching layer, we obtain the question-aware answer representations $R_a \in \mathbb{R}^{m \times d}$ and the answer-aware question representations $R_q \in \mathbb{R}^{n \times d}$, where d is the size of representations. Finally, we apply another bidirectional LSTM encoders on the R_a and R_q individually to generate the question-aware contextual answer representations $H_a \in \mathbb{R}^{m \times d}$ and the answer-aware contextual question representations $H_q \in \mathbb{R}^{n \times d}$. The last time-step of R_q and H_q are concatenated to form a sketch vector G_q, which outlines the matching result in the perspective of question. We also obtain another sketch vector G_a, which outlines the matching result in the perspective of answer. The final aggregation vector G_{qa} used for prediction is the concatenation of G_q and G_a.

To optimize our answer selection model LSTM-ASM, we use the cross-entropy loss function:

$$\mathcal{L}_{qa} = -\left[y \log p_{qa} + (1-y) \log (1-p_{qa})\right] \quad (1)$$

where p_{qa} is the predicted probability:

$$p_{qa} = \text{softmax}\left(W_{qa} G_{qa} + b_{qa}\right) \quad (2)$$

Here, W_{qa} and b_{qa} are trainable parameters.

3.2 LSTM-ASMSY

Different from LSTM-ASM, here we design another answer selection model that is capable of making use of reference answer summaries as additional information during training. Since the additional answer summary information is only available during the training period, our model is carefully designed to be able to make predictions without answer summaries as inputs. For simplicity, we name this model as LSTM-based Answer Selection Model with Summary (LSTM-ASMSY). As depicted in Figure 1, LSTM-ASMSY is composed of two modules: a long answer selection module

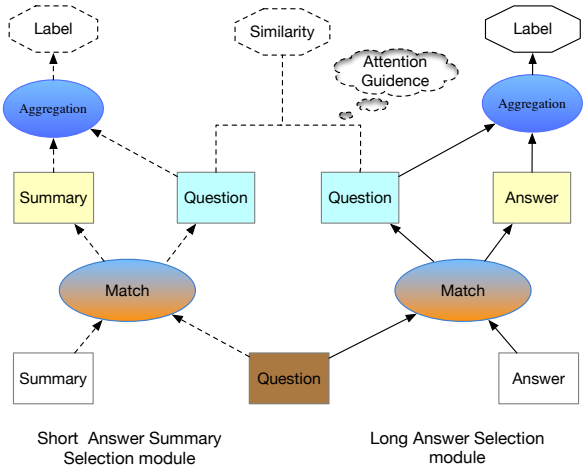

Figure 1: Architecture for LSTM-ASMSY.

and a short answer summary selection module. The short answer summary selection module adopts the same architecture as the LSTM-ASM model defined in Section 3.1. And it just plays a role in the training stage. Meanwhile, the long answer selection module also have a same architecture as the LSTM-ASM model but does not share any parameters with the short answer summary selection module. In the following, we detail how to train our LSTM-ASMSY model and in what way we can only use its long answer selection module to make predictions at inference time.

Given a question q, an answer a and its summary s, we first feed the long answer selection module with question q and answer a to obtain the sketch vector $G_q^{'}$ and the final aggregation vector $G_{qa}^{'}$ as in Section 3.1. Meanwhile, we feed the short answer summary selection module with the same question q and the corresponding answer summary s to get the sketch vector $\hat{G}_q$ and the aggregation vector G_{qs} in similar ways. Then, we induce the two modules to make the same prediction during training:

$$\mathcal{L}_{qa}^{'} = -\left[y \log p_{qa}^{'} + (1-y) \log \left(1 - p_{qa}^{'}\right)\right] \quad (3)$$

$$\mathcal{L}_{qs} = -\left[y \log p_{qs} + (1-y) \log \left(1 - p_{qs}\right)\right] \quad (4)$$

where $p_{qa}^{'}$ is the predicted probability output by the long answer selection module, and p_{qs} is the predicted probability output by the answer summary selection module. They are calculated as:

$$p_{qa}^{'} = \text{softmax}\left(W_{qa}^{'} G_{qa}^{'} + b_{qa}^{'}\right) \quad (5)$$

$$p_{qs} = \text{softmax}\left(W_{qs} G_{qs} + b_{qs}\right) \quad (6)$$

Where $W_{qa}^{'}$, W_{qs}, $b_{qa}^{'}$ and b_{qs} are trainable parameters, y is the gold label.

Next, in consideration of the sketch vector $G_q^{'}$ represents how well all the aspects of question are related to answer a while the sketch vector $\hat{G}_q$ represents how well all the aspects of question are matched in the perspective of the corresponding answer summary s, we encourage the sketch vector $G_q^{'}$ and $\hat{G}_q$ to be as similar as possible:

$$\mathcal{L}_{smi} = \left\| G_q^{'} - \hat{G}_q \right\|^2 \quad (7)$$

In such ways, we can not only leverage the answer summaries to provide implicit attention guidance during training, but also use the trained long answer selection module for making predictions without relying on the short answer summary selection module.

Finally, the overall loss function to optimize is:

$$\mathcal{L} = \lambda_1 * \mathcal{L}_{qa}^{'} + \lambda_2 * \mathcal{L}_{qs} + \lambda_3 * \mathcal{L}_{smi} \quad (8)$$

where λ_1, λ_2, λ_3 are tuneable hyper-parameters.

3.3 Extensions to ALBERT

ALBERT has achieved the state-of-the-art performance on sequence pair classification task but it can only process at most 512 tokens. However, on the *WikiHowQA* answer selection task, the average length of answer texts is more than 520 words, where each word could be broken down into more than one sub-word token. Hence, it prevents us from directly using ALBERT on this task. Here, we describe a simple way to extend the original ALBERT for handling long-form text matching.

Different from previous methods, like Longformer (Beltagy et al., 2020), ETC (Ainslie et al., 2020) and Big Bird (Zaheer et al., 2020), that require pre-training a new language model, our method does not need to train a language model from scratch. To be specific, we simply extend the original ALBERT to have a larger position vocabulary. And we reuse all the pretrained parameters within the original ALBERT model except the position embeddings. Besides, we initialize the first 512 position embeddings with original position embeddings and leave the rest random. The extended ALBERT is named as ALBERT-based Answer Selection Model (ALBERT-ASM). And we use the final hidden vector corresponding to the first input token ([CLS]) as the aggregate representation for measuring the relevancy of question-answer pair.

The way we build an ALBERT-based model that is capable of making use of reference answer summary during training is similar to Section 3.2. Specifically, we apply one extended ALBERT model to model the relevancy of question-answer pair and use another ALBERT model to model the relevancy of question-summary pair. And we encourage the hidden vectors corresponding to the question tokens of the two models to be as similar as possible. Similarly, we name this model as ALBERT-ASMSY.

4 Experiments

4.1 Training Details

For training our LSTM-based models, we use 300-dimensional GloVe word embeddings (Pennington et al., 2014) and apply a bidirectional GRU (Chung et al., 2014) to obtain a 100-dimensional character-level embedding for each word. The hidden size for all Bi-LSTM is set to 200 and the dropout ratio is set to be 0.1. We truncate candidate answer and its summary to 900 tokens and 100 tokens respectively. During training, we use the popular Adam optimizer (Kingma and Ba, 2015) and set its learning rate to be 0.0005. The batch size is set to 48 per gpu and the hyper-parameter λ_1, λ_2 and λ_3 are set to 1. We apply early stopping based on the evaluation result on the validation set. The maximum number of epochs is set to 20 and the patience is set to 5.

For training our ALBERT-based models, we initialize our models using the pretrained Albert-base-v1 model [4]. The maximum input sequence length of ALBERT-ASM is set to be 512, and the maximum input sequence length of ALBERT-ASMSY is set to be 1536, which is three times as long as the original maximum input length. We update our model using a batch size of 64 per gpu. And we adopt the AdamW (Loshchilov and Hutter, 2019) as our optimizer. Besides, we set the learning rate to be 3e-5 and the gradient clipping parameter to be 1.0. The maximum number of epochs is set to 3 for all the experiments. The hyper-parameter λ_1, λ_2 and λ_3 are set to 1.

4.2 Metrics

The performance of our models is measured in Mean Reciprocal Rank (MRR) and Mean Average Precision (MAP), which are standard metrics in Information Retrieval and Question Answering. The MRR measures the rank of *any* correct answer, while MAP examines the ranks of *all* the correct answers. Generally, the higher the scores, the better performance the model has.

4.3 Experiment Setups

We mainly compare our approaches against the following baselines:

(1) *Long answer selection methods*: CA is a widely-adopted compare-aggregate baseline for matching sequence pairs (Wang and Jiang, 2017). COALA is a recent baseline proposed by Rücklé et al. (2019), which has been proven to be effective in long answer selection task.

(2) *Two-Stage methods*: QPGN+AP-BiLSTM, QPGN+CA, and QPGN+COALA are three Two-Stage baselines, which first summarize the original lengthy answers and then conduct answer selection over the short generated answer summaries. Here, the QPGN is a question-driven pointer-generator network proposed by Deng et al. (2020), which is used to generate answer summaries for answer selection. AP-BiLSTM (Santos et al., 2016), CA (Wang and Jiang, 2017), COALA (Rücklé et al., 2019) are adopted answer selection models.

(3) *Joint Learning methods*: ASAS (Deng et al., 2020) is the recent state-of-the-art model that tackles the tasks of answer selection and answer summary generation in a joint manner.

For our own approaches, we evaluate the following models: LSTM-ASM, LSTM-ASMSY, ALBERT-ASM, and ALBERT-ASMSY. Besides, we also assess the performance of our models when they are fed with gold answer summary at test on purpose.

4.4 Main Results

Table 3 presents experiment results. In this table, we detail all model-specific inputs and their performance for each model. For example, our model LSTM-ASMSY jointly train an answer summary selection module and a long answer selection module during training and only adopt the trained long answer selection module for making predictions in consideration of the unavailability of answer summary at test. Hence, the input to LSTM-ASMSY during training is a question-answer-summary triplet while its input at test is a question-answer pair.

From Table 3, we mainly note the following observations: (1) LSTM-ASM achieve a new state-of-the-art result with an improvement of 18.67 MRR

[4] https://huggingface.co/albert-base-v1

Models	Inputs (training stage)	Inputs (test stage)	MAP	MRR
CA	Q, A	Q, A	50.22	52.14
COALA	Q, A	Q, A	50.03	51.96
QPGN+AP-BiLSTM	Q, A, S	Q, Generated S	52.37	53.43
QPGN+CA	Q, A, S	Q, Generated S	52.46	53.73
QPGN+COALA	Q, A, S	Q, Generated S	51.97	53.02
ASAS	Q, A, S	Q, A	55.22	56.86
Ours: LSTM-ASM	Q, A	Q, A	**72.91**	**75.53**
Ours: LSTM-ASMSY	Q, A, S	Q, A	72.40	74.97
Ours: ALBERT-ASM	Q, A	Q, A	**82.74**	**85.01**
Ours: ALBERT-ASMSY	Q, A, S	Q, A	82.65	84.97
Ours: LSTM-ASM†	Q, A	Q, S	64.99	67.75
Ours: LSTM-ASMSY†	Q, A, S	Q, S	64.42	67.20
Ours: ALBERT-ASM†	Q, A	Q, S	73.78	76.66
Ours: ALBERT-ASMSY†	Q, A, S	Q, S	73.69	76.62
Ours: LSTM-ASM†	Q, S	Q, S	65.77	68.50
Ours: ALBERT-ASM†	Q, S	Q, S	75.51	78.37

Table 3: Performance of different models on the *WikiHowQA* answer selection task. Here, *Q*, *A* and *S* denotes question, answer and summary respectively. Results marked † are the performances of our models when we purposely feed them with *gold answer summary* during test.

and 17.69 MAP over the best reported results in (Deng et al., 2020), which they obtained with a joint learning method, named ASAS. This demonstrates that LSTM-ASM can serve as a new strong baseline that uses LSTM as text encoders on this task. (2) LSTM-ASM has a better performance than LSTM-ASMSY and ALBERT-ASM also outperforms ALBERT-ASMSY, which show that making use of the additional answer summaries does not help to solve the answer selection problem. (3) From the last two rows of Table 3, we could see that, if we train our model LSTM-ASM and ALBERT-ASM with question-summary pairs and also test them using question-summary pairs [5], we could see a significant performance drop. This signifies that answer summaries are less informative than original lengthy answers when modeling the relevancy degrees of question-answer pairs. Also, it explains that leveraging additional answer summaries information may not result in a performance gain. (4) ALBERT-ASM and ALBERT-ASMSY have a much better performance than all LSTM-based models. This shows that better contextualized word representations brought by pretrained language models could be very effective in this long answer selection task. Meanwhile, these results also indicate that our method of extending ALBERT to do long answer selection is effective. (5) From row 11 to row 14, we can see that, when we train our models as usual but feed them with gold answer summary at test, we can observe severe performance degradation. This which further verifies that, question aspects covered by original answers are not always covered by their corresponding answer summaries to some extent.

Models	MAP	MRR
LSTM-ASM		
Full model	**72.91**	**75.53**
Max_tokens: 200	69.70	72.70
Max_tokens: 400	71.06	73.74
Max_tokens: 600	72.67	75.27
ALBERT-ASM		
Full model	**82.74**	**85.01**
Max_tokens: 512	81.79	84.05

Table 4: Model performances as a function of the length of input answers (averaged over three runs with different random seeds). "Max_tokens" denotes the maximum number of tokens of original lengthy answers to keep.

[5]Here, we do it on purpose for checking the informativeness of answer summary.

Annotation Source	Accuracy
question-answer pairs	**93%**
question-summary pairs	78%

Table 5: Human performances on the 100 sampled examples. The accuracy is the ratio of correct annotation.

In Table 4, we present the performances of our models as a function of the length of answers. From it, we can observe that the maximum length limit of answers has a big impact on the performance of our models. Both LSTM-ASM and ALBERT-ASM tend to exhibit better performance when we increase the maximum length limit of answers. This indicates that the *WikiHowQA* answer selection task does require the ability to deal with long answer.

5 Data Analysis

5.1 Human Performance

We randomly sampled 100 examples, namely 100 question-answer-summary triplets, from the test portion of the dataset for analysis. We first transform the 100 question-answer-summary triplets into 100 question-answer pairs and 100 question-summary pairs. Then, we divide our annotators into two groups, namely group 1, group 2. Each group consists of two annotators. Finally, we ask annotators of group 1 to label the question-answer pairs and ask annotators of group 2 to label the corresponding question-summary pairs [6]. Since the *WikiHowQA* dataset was created in an automatic and heuristic way, to make a fair evaluation, we also ask a graduate student who majors in linguistics to annotate the 100 examples and use these annotations as gold labels.

Table 5 show the annotation results. From it, we can see that annotators that use question-answer pairs as annotation source, are able to achieve an accuracy of 93% on this sampled subset. Whereas, when annotators use question-summary pairs as annotation source, they only obtain an accuracy of 78%, which is substantially lower than the previous one. This verifies that, on the *WikiHowQA* dataset, picking out the correct answers by measuring the relevancy degrees between question and its answer summary is much harder for human annotators.

In order to investigate whether additional answer summaries could help to boost the annotation performance, we also ask another two annotators to directly use the 100 question-answer-summary triplets as annotation source. After comparing these annotation results with the annotation results given by group 1, we find that the two results are almost exactly the same everywhere. This shows that additional answer summaries can not assist in improving the annotation performance either.

[6] We eliminate the annotation divergence by following the inter-annotator agreement.

Gold	Reference	G1	G2	Percentage
0	0	0	0	66%
0	0	0	1	16%
0	0	1	1	1%
1	0	1	1	5%
1	1	0	1	1%
1	1	1	1	11%

Table 6: The percentage of each annotation category. Here, we omit the categories with zero proportion.

Questions: How to change your name *in ohio*?
Answer: *In florida*, the name change process starts with checking your criminal history. In order to do this, you must have your fingerprints submitted for a state and national criminal records check. The fingerprints will be taken by the *florida department* of law enforcement …
Summary: Have a background check. Gather information for the petition. Fill out and sign the petition. File your petition. Attend your hearing. ..
Gold Label: **0** Reference Label: **0** Group 1: **0** Group 2: **1**

Questions: How to buy *affordable furniture*?
Answer: *Sales* happen all the time at retailers, especially around the holidays. While major holidays have their own sales, *furniture retailers have especially large sales* around president' s day, labor day, and memorial day. Take advantage of these sales to score larger furniture items and matching sets. January and July are also *good times to shop*…
Summary: *Shop seasonally.* Check retail store websites. *Use coupons. Search for clearance sales.*
Gold Label: **1** Reference Label: **0** Group 1: **1** Group 2: **1**

Figure 2: Some examples of annotation divergence.

5.2 Comprehensive Analysis

To get a comprehensive understanding of the above phenomenon, we further conduct an in-depth analysis. Specifically, for each example, we make a comparison among its gold label (Gold), its reference label (Reference, given by the dataset itself), its label annotated by group 1 (G1) and its label annotated by group 2 (G2). Table 6 provides our estimate of the percentage for each category.

From Table 6, we have the following observations: (i) For most of the examples, the gold labels, the reference labels, the labels annotated by group 1, and the labels annotated by group 2 are the same (All are zeros or ones), which indicates that the *WikiHowQA* answer selection task is relatively easy though it contains long multi-sentence answer texts. (ii) From the second row of table 6, we could see that 16 out of 100 samples are labeled correctly by annotators in group 1 but mislabeled by annotators in group 2. After carefully checking these examples, we observe an interesting finding from them. Specifically, most of these errors may have been caused by a lack of some specific details. A representative example of this category is presented at the top of Figure 2. From it, we can see that the answer summary explains how to change one's name in a general way while the answer explains "How to Change Your Name in Florida". This kind of annotation divergence is mainly due to the fact that the answer summary is relatively abstract and general while the answer contains all the clues. Therefore, we deem that the relevancy degree between question-summary pair is not enough for human annotators to make correct decisions in some cases. (iii) 5 examples of this sample set are labeled as '1' by annotators both in group 1 and group 2. And the reference labels of these examples are '0' but their gold label are '1'. By carefully checking these examples, we are confident that human annotators are correct. This clearly shows that the dataset is noisy to a certain extent. We present one of the representative examples at the bottom of Figure 2. From the example, we could see that both the answer candidate and the answer summary describe a way to shop for furniture on a reasonable budget, which exactly answer the question "How to buy affordable furniture ?". However, the reference label given by the dataset is '0', which largely may be due to that the dataset itself was created in an automatic and heuristic way.

6 Related Work

Long Answer Selection WikiPassageQA (Han et al., 2019) and LSA (Rücklé et al., 2019) are the two most well-known long answer selection benchmarks. On WikiPassageQA dataset, the state-of-the-art, non-pretraining-based method is proposed by Han et al. (2019). In their method, they derive contextualized uni-gram representation from n-grams and demonstrate that enabling multi-granular matches between question and answer n-grams are the key factors. On LSA benchmark, Rücklé et al. (2019) shows that a relevance matching approach based on the compare-aggregate framework with a coverage-based constraint works best among various LSTM-based methods.

In the midst of the pretraining-based methods, the best one is a self-supervised text matching model (Rücklé et al., 2020) which incorporates self-supervised with supervised multi-task learning on 140 source domains. It achieves state-of-the-art performances on both WikiPassageQA dataset and LSA benchmark.

Analysis of QA Tasks Several studies have investigated aspects of the design of QA datasets. Chen et al. (2016) conduct an examination of the CNN/Daily Mail reading comprehension dataset and conclude that this dataset is quite noisy and the required reasoning and inference level of this dataset is very simple. Sugawara et al. (2017) propose two classes of metrics (prerequisite skills and readability) to the quality of reading comprehension dataset. And they find that the readability of reading comprehension datasets does not directly affect the question difficulty. Yue et al. (2020) carry out a thorough analysis of the emrQA dataset (Pampari et al., 2018). And they discover that, though Pampari et al. (2018) claims that 39% of the questions may need knowledge to answer, their analysis shows that only a very small portion of the errors (2%) made by a state-of-the-art model might result from missing external domain knowledge.

In cQA, Liu et al. (2008) do a comprehensive analysis of questions and answers on cQA services and find that some questions usually have several best answers. And they show that customized question-type focused summarization techniques helps to improve cQA answer quality. Yang et al. (2011) analyze the not-answered questions in cQA and give a try on making predictions whether questions will receive answers.

7 Conclusion

In this paper, we carefully study the recent *WikiHowQA* answer selection task. Our models, either LSTM-based or ALBERT-based, outperform the previous state-of-the-art method by a large margin. More importantly, we do a careful hand-analysis of a small subset of the dataset. Overall, we think the *WikiHowQA* dataset is a valuable dataset, which provides a promising avenue for research on non-

factoid, long answer selection task. Nevertheless, we argue that: (i) this dataset is still noisy due to its method of data creation. (ii) For "how-to" answer selection task, the additional answer summaries can neither help to improve model performance nor can effect human annotation. (iii) the answer selection task is relatively easy though it contains long multi-sentence answer texts.

Acknowledgement

We thank all the reviewers for their valuable comments to improve this paper. The work is supported by National Natural Science Foundation of China (Grant No.62036001, No.62032001) and PKU-Baidu Fund (No. 2020BD021). The corresponding author of this paper is Houfeng Wang.

References

Joshua Ainslie, Santiago Ontanon, Chris Alberti, Vaclav Cvicek, Zachary Fisher, Philip Pham, Anirudh Ravula, Sumit Sanghai, Qifan Wang, and Li Yang. 2020. Etc: Encoding long and structured inputs in transformers. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 268–284.

Iz Beltagy, Matthew E Peters, and Arman Cohan. 2020. Longformer: The long-document transformer. *arXiv preprint arXiv:2004.05150*.

Danqi Chen, Jason Bolton, and Christopher D. Manning. 2016. A thorough examination of the cnn/daily mail reading comprehension task. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics, ACL 2016, August 7-12, 2016, Berlin, Germany, Volume 1: Long Papers*. The Association for Computer Linguistics.

Junyoung Chung, Caglar Gulcehre, KyungHyun Cho, and Yoshua Bengio. 2014. Empirical evaluation of gated recurrent neural networks on sequence modeling. *arXiv preprint arXiv:1412.3555*.

Daniel Cohen, Liu Yang, and W Bruce Croft. 2018. Wikipassageqa: A benchmark collection for research on non-factoid answer passage retrieval. In *The 41st International ACM SIGIR Conference on Research & Development in Information Retrieval*, pages 1165–1168.

Yang Deng, Wai Lam, Yuexiang Xie, Daoyuan Chen, Yaliang Li, Min Yang, and Ying Shen. 2020. Joint learning of answer selection and answer summary generation in community question answering. In *The Thirty-Fourth AAAI Conference on Artificial Intelligence, AAAI 2020, The Thirty-Second Innovative Applications of Artificial Intelligence Conference, IAAI 2020, The Tenth AAAI Symposium on Educational Advances in Artificial Intelligence, EAAI 2020, New York, NY, USA, February 7-12, 2020*, pages 7651–7658. AAAI Press.

Minwei Feng, Bing Xiang, Michael R Glass, Lidan Wang, and Bowen Zhou. 2015. Applying deep learning to answer selection: A study and an open task. In *2015 IEEE Workshop on Automatic Speech Recognition and Understanding (ASRU)*, pages 813–820. IEEE.

Hojae Han, Seungtaek Choi, Haeju Park, and Seungwon Hwang. 2019. Micron: Multigranular interaction for contextualizing representation in non-factoid question answering. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 5892–5897.

Diederik P. Kingma and Jimmy Ba. 2015. Adam: A method for stochastic optimization. In *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*.

Mahnaz Koupaee and William Yang Wang. 2018. Wikihow: A large scale text summarization dataset. *arXiv preprint arXiv:1810.09305*.

Zhenzhong Lan, Mingda Chen, Sebastian Goodman, Kevin Gimpel, Piyush Sharma, and Radu Soricut. 2020. ALBERT: A lite BERT for self-supervised learning of language representations. In *8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, April 26-30, 2020*. OpenReview.net.

Yuanjie Liu, Shasha Li, Yunbo Cao, Chin-Yew Lin, Dingyi Han, and Yong Yu. 2008. Understanding and summarizing answers in community-based question answering services. In *COLING 2008, 22nd International Conference on Computational Linguistics, Proceedings of the Conference, 18-22 August 2008, Manchester, UK*, pages 497–504.

Ilya Loshchilov and Frank Hutter. 2019. Decoupled weight decay regularization. In *7th International Conference on Learning Representations, ICLR 2019, New Orleans, LA, USA, May 6-9, 2019*. OpenReview.net.

Anusri Pampari, Preethi Raghavan, Jennifer J. Liang, and Jian Peng. 2018. emrqa: A large corpus for question answering on electronic medical records. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing, Brussels, Belgium, October 31 - November 4, 2018*, pages 2357–2368. Association for Computational Linguistics.

Jeffrey Pennington, Richard Socher, and Christopher D Manning. 2014. Glove: Global vectors for word representation. In *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*, pages 1532–1543.

Andreas Rücklé, Nafise Sadat Moosavi, and Iryna Gurevych. 2019. Coala: A neural coverage-based approach for long answer selection with small data. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, pages 6932–6939.

Andreas Rücklé, Jonas Pfeiffer, and Iryna Gurevych. 2020. Multicqa: Zero-shot transfer of self-supervised text matching models on a massive scale. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing, EMNLP 2020, Online, November 16-20, 2020*, pages 2471–2486. Association for Computational Linguistics.

Cicero dos Santos, Ming Tan, Bing Xiang, and Bowen Zhou. 2016. Attentive pooling networks. *arXiv preprint arXiv:1602.03609*.

Saku Sugawara, Yusuke Kido, Hikaru Yokono, and Akiko Aizawa. 2017. Evaluation metrics for machine reading comprehension: Prerequisite skills and readability. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics, ACL 2017, Vancouver, Canada, July 30 - August 4, Volume 1: Long Papers*, pages 806–817. Association for Computational Linguistics.

Shuohang Wang and Jing Jiang. 2017. A compare-aggregate model for matching text sequences. In *5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Conference Track Proceedings*. OpenReview.net.

Zhiguo Wang, Wael Hamza, and Radu Florian. 2017. Bilateral multi-perspective matching for natural language sentences. In *Proceedings of the 26th International Joint Conference on Artificial Intelligence*, pages 4144–4150.

Wei Wu, Xu Sun, and Houfeng Wang. 2018. Question condensing networks for answer selection in community question answering. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics, ACL 2018, Melbourne, Australia, July 15-20, 2018, Volume 1: Long Papers*, pages 1746–1755. Association for Computational Linguistics.

Yuexiang Xie, Ying Shen, Yaliang Li, Min Yang, and Kai Lei. 2020. Attentive user-engaged adversarial neural network for community question answering. In *Proceedings of the AAAI Conference on Artificial Intelligence*, pages 9322–9329.

Lichun Yang, Shenghua Bao, Qingliang Lin, Xian Wu, Dingyi Han, Zhong Su, and Yong Yu. 2011. Analyzing and predicting not-answered questions in community-based question answering services. In *Proceedings of the Twenty-Fifth AAAI Conference on Artificial Intelligence, AAAI 2011, San Francisco, California, USA, August 7-11, 2011*. AAAI Press.

Xiang Yue, Bernal Jimenez Gutierrez, and Huan Sun. 2020. Clinical reading comprehension: A thorough analysis of the emrqa dataset. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, ACL 2020, Online, July 5-10, 2020*, pages 4474–4486. Association for Computational Linguistics.

Manzil Zaheer, Guru Guruganesh, Avinava Dubey, Joshua Ainslie, Chris Alberti, Santiago Ontanon, Philip Pham, Anirudh Ravula, Qifan Wang, Li Yang, et al. 2020. Big bird: Transformers for longer sequences. *arXiv preprint arXiv:2007.14062*.

Case-Based Abductive Natural Language Inference

Marco Valentino, Mokanarangan Thayaparan, André Freitas
Idiap Research Institute, Switzerland
University of Manchester, United Kingdom
marco.valentino@idiap.ch

Abstract

Most of the contemporary approaches for multi-hop Natural Language Inference (NLI) construct explanations considering each test case in isolation. However, this paradigm is known to suffer from semantic drift, a phenomenon that causes the construction of spurious explanations leading to wrong conclusions. In contrast, this paper proposes an abductive framework for multi-hop NLI exploring the retrieve-reuse-refine paradigm in Case-Based Reasoning (CBR). Specifically, we present *Case-Based Abductive Natural Language Inference (CB-ANLI)*, a model that addresses unseen inference problems by analogical transfer of prior explanations from similar examples. We empirically evaluate the abductive framework on commonsense and scientific question answering tasks, demonstrating that CB-ANLI can be effectively integrated with sparse and dense pre-trained encoders to improve multi-hop inference, or adopted as an evidence retriever for Transformers. Moreover, an empirical analysis of semantic drift reveals that the CBR paradigm boosts the quality of the most challenging explanations, a feature that has a direct impact on robustness and accuracy in downstream inference tasks.

1 Introduction

Multi-hop inference is the task of composing two or more pieces of evidence from external knowledge resources to address a particular reasoning problem (Thayaparan et al., 2020). In the context of Natural Language Inference (NLI), this task is often used to develop and evaluate explanation-based systems, capable of performing transparent multi-step reasoning with natural language (Wiegreffe and Marasovic, 2021; Jansen et al., 2018; Camburu et al., 2018). While multi-hop inference has been largely explored for extractive problems such as open-domain question answering (Yang et al., 2018), increasing attention is being dedicated to the abstractive setting, where the models are required to compose long chains of facts expressing abstract commonsense and scientific knowledge (Clark et al., 2018; Valentino et al., 2022).

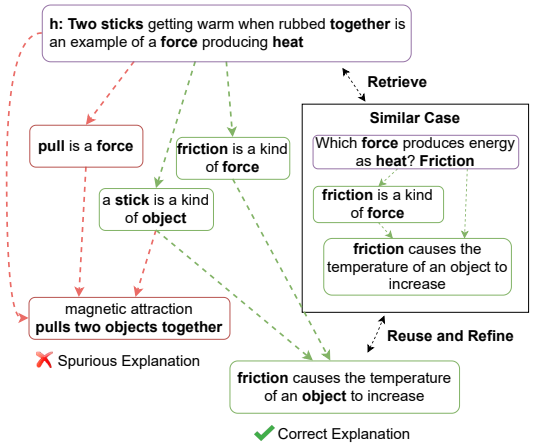

Figure 1: Performing multi-hop inference considering each case in isolation can lead to the construction of spurious explanations. In contrast, we propose the adoption of a *Case-Based Reasoning (CBR)* paradigm where the construction of new explanations is constrained by previously solved examples.

In this setting, multi-hop inference is often framed as an Abductive Natural Language Inference (ANLI) problem (Bhagavatula et al., 2019), where, for a given set of alternative hypotheses $H = \{h_1, h_2, \ldots, h_n\}$, the goal is to construct an explanation for each $h_i \in H$ and select the hypothesis supported by the best explanation (Thayaparan et al., 2021a). Existing approaches address abductive inference considering each test hypothesis in isolation, employing iterative and path-based methods (Kundu et al., 2019; Yadav et al., 2019b) or explicit constraints to guide the generation of a plausible explanation graph supporting the correct answer (Khashabi et al., 2016; Khot et al., 2017).

However, this paradigm poses several challenges in the abstractive setting as:

- The structure of the explanation is not evident

Proceedings of the 29th International Conference on Computational Linguistics, pages 1556–1568
October 12–17, 2022.

from the decomposition of the hypothesis, that is, the type of facts required for the inference cannot be derived from the surface form of the reasoning problem;

- Core explanatory facts tend to be abstract, sharing a low number of terms with the hypothesis, making it hard to correctly estimate their relevance for the inference;

- Background knowledge sources contain a large amount of irrelevant facts overlapping with the hypothesis, a feature that can lead to the generation of spurious explanations.

Consequently, existing approaches often suffer from a phenomenon known as *semantic drift* (Khashabi et al., 2019) – i.e., the tendency of composing incorrect reasoning chains leading to wrong conclusions as the number of required inference steps increases. The example in Figure 1 illustrates some of these challenges.

In contrast with the dominant paradigm, we propose to integrate Abductive Natural Language Inference in a Case-Based Reasoning (CBR) framework (Schank et al., 2014; Das et al., 2021). CBR systems operate under the hypothesis that similar problems require similar solutions, addressing new cases via analogical transfer from previous cases solved in the past. Specifically, the Case-Based Reasoning framework employs a retrieve-reuse-refine paradigm to model inference over unseen problems (Schank, 2013; De Mantaras et al., 2005). In the context of multi-hop inference, we hypothesise that the adoption of a Case-Based Reasoning framework can help tackle some of the challenges involved in the abstractive setting since:

- Similar natural language hypotheses tend to require similar explanations;

- Abstract facts tend to express general explanatory knowledge about underlying regularities, being frequently reused to explain a large variety of phenomena;

- Prior solutions can explicitly help constraint the search space, reducing the risk of composing spurious inference chains.

To investigate these hypotheses, we present a *Case-Based Abductive NLI (CB-ANLI)* model that retrieves and adapts natural language explanations from training examples to construct new explanations for unseen cases and address downstream inference problems. Specifically, this paper provides the following main contributions:

- To the best of our knowledge, we are the first to propose an end-to-end case-based abductive framework for multi-hop and explanation-based NLI;

- We empirically demonstrate the efficacy of the CB-ANLI on commonsense and scientific reasoning tasks, showing that the proposed model can be effectively integrated with different sentence encoders and downstream Transformers, achieving strong performance when compared to existing multi-hop and explainable approaches;

- We investigate the impact of the retrieve-reuse-refine paradigm on semantic drift, and how this affects accuracy and robustness of the model. Our results show that the case-based framework boosts the quality of the explanations for the most challenging problems, resulting in improved downstream inference performance.

2 Case-based Abductive NLI

For a given set of alternative natural language hypotheses $H = \{h_1, h_2, \ldots, h_n\}$, the goal of Abductive NLI is to construct an explanation for each $h_i \in H$ and select the hypothesis supported by the best explanation. Given an hypothesis h_i (e.g., *"Two sticks getting warm when rubbed together is an example of a force producing heat"*), we construct an explanation justifying h_i by extracting and composing inference chains between multiple explanatory facts retrieved from an external corpus.

To generate and score an explanation for h_i, we adopt a Case-Based Reasoning (CBR) paradigm composed of three major phases, retrieve-reuse-refine, which can be summarised as follows (see Fig. 2):

1. **Retrieve:** In the retrieve phase, we employ a sentence encoding mechanism to search over two distinct embedding spaces. A first embedding space (*Facts Embeddings*) is adopted to retrieve a set of candidate explanatory sentences for the hypothesis. A second embedding space (*Cases Embeddings*) is used to retrieve similar cases solved in the past whose

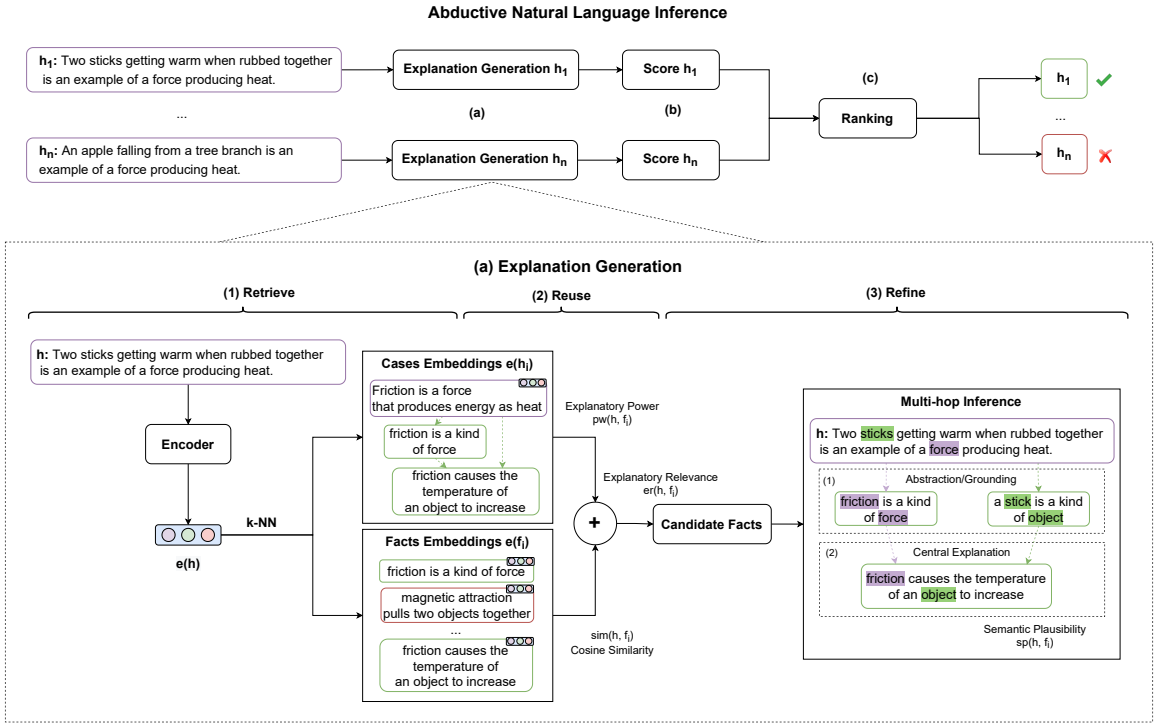

Figure 2: Overview of the proposed CBR framework. We adopt a retrieve-reuse-refine paradigm to construct and score explanations for a set of mutually exclusive hypotheses (a) and address NLI tasks via abductive inference by selecting the hypothesis supported by the best explanation.

explanations can be useful to guide the search for a new solution.

2. **Reuse:** In the reuse phase, we condition the relevance of a given fact on the set of explanations retrieved from the most similar cases. Specifically, we reuse previously solved cases to estimate the *explanatory power* of a fact, representing the extent to which a given sentence appears in explanations for past hypotheses.

3. **Refine:** In this phase, the list of candidate explanatory facts is refined to build the final explanation. We model the construction of an explanation via multi-hop inference between hypothesis and candidate facts, composing abstractive inference chains to estimante the plausibility of the candidate explanatory sentences.

Given a set of alternative hypotheses, we adopt the CBR framework for explanation generation, and subsequently leverage the score assigned to each explanation to address downstream NLI tasks. Additional details on the retrieve-reuse-refine phases are described in the following sections.

3 Explanation Generation

3.1 Retrieve

We perform k-NN search over two distinct embedding spaces: (a) an embedding space encoding individual commonsense and scientific facts that can be used to construct new explanations (*Facts Embeddings*); (b) an embedding space of true hypotheses associated with their respective explanations (*Cases Embeddings*). An explanation for a given hypothesis h_i is a composition of facts $E_i = \{f_1, \ldots, f_n\}$ form the Facts Embeddings.

To perform k-NN search, we employ a sentence encoder $e(\cdot)$. Specifically, we use $e(\cdot)$ to derive a vector for the test hypothesis h and adopt cosine similarity to efficiently score and rank facts and hypotheses in the embedding spaces, retrieving the top-k instances. We perform our experiments using a sparse (BM25 (Robertson et al., 2009)) and a pre-trained dense encoder (Sentence-BERT (Reimers and Gurevych, 2019)) adopting a search index for efficient retrieval (IndexIVFFlat in FAISS (Johnson et al., 2019)). We adopt the WorldTree corpus (Jansen et al., 2018) as background knowledge (additional details in Section 5).

3.2 Reuse

Previous work has shown that explanatory facts expressing underlying regularities tend to create explanatory patterns across similar hypotheses (Valentino and Freitas, 2022; Valentino et al., 2021, 2022). Following this line of research, we conjecture that explanations from similar cases can be reused to constraining the search space for unseen hypotheses and improve downstream NLI.

Specifically, given an unseen hypothesis h and a fact f_i, we adopt the explanations retrieved from the top-K similar hypotheses in the *Case Embeddings* to estimate the *explanatory power* of f_i:

$$pw(h, f_i) = \sum_{h_k \in kNN(h)}^{K} sim(e(h), e(h_k)) \cdot \mathbb{1}(f_i, h_k) \quad (1)$$

$$\mathbb{1}(f_i, h_k) = \begin{cases} 1 & \text{if } f_i \in E_k \\ 0 & \text{if } f_i \notin E_k \end{cases} \quad (2)$$

where $kNN(h) = \{h_1, \ldots, h_K\}$ represents the list of k-nearest hypotheses of h retrieved according to the cosine similarity $sim(\cdot)$ between the embeddings $e(h)$ and $e(h_k)$, and $\mathbb{1}(\cdot)$ is the indicator function verifying if f_i is included in the explanation E_k for the hypothesis h_k. Therefore, for each hypothesis h_k in the set of k-nearest neighbours, the model sums up the quantity $sim(\cdot)$ only if f_i is used to explain h_k. Since $sim(e(h), e(h_k))$ represents the similarity between h and h_k, the more f_i explains past hypotheses that are similar to h the higher the explanatory power of f_i. To condition the list of candidate explanatory facts on previously solved cases while controlling for relevance with respect to the test hypothesis h, we compute the final *explanatory relevance* of each f_i by interpolating the explanatory power with the similarity between the embeddings $e(h)$ and $e(f_i)$:

$$er(h, f_i) = \lambda \cdot sim(e(h), e(f_i)) + (1 - \lambda) \cdot pw(h, f_i) \quad (3)$$

The explanatory relevance score is used to re-rank and filter the list of candidate facts for the subsequent phase.

3.3 Refine

In the refine phase, the model considers the set of candidate facts retrieved in the previous stage to construct the final explanation for h. We model the construction of an explanation through multi-hop inference between hypothesis and candidate facts via the composition of explicit inference chains. To this end, we represent facts and hypothesis as bags of distinct concepts $CP(s_i) = \{cp_1, \ldots, cp_n\}$ (e.g., *"friction is a kind of force"* is represented as the set $\{friction, force\}$, details in the appendix), and connect two generic sentences s_i and s_j by means of shared concept in $CP(s_i) \cap CP(s_j)$.

To link the hypotheses to potentially abstract explanatory sentences, we construct an explanation graph in different stages, starting with the hypothesis h as the only node. In the first stage, the model extends the graph with the facts that share direct concepts with h and that express taxonomic relations or synonymy. This step can be seen as an abstraction/grounding mechanism aimed at linking the hypothesis to core explanatory facts (Jansen et al., 2018; Thayaparan et al., 2021a) (e.g., linking *stick* to *object* and *friction* to *force* in Figure 2).

In the second stage, the model extends the graph with all the remaining candidate explanatory facts that share at least one concept with previously added nodes. We consider these facts as *central explanatory nodes*. After constructing the graph, we estimate the *semantic plausibility* of the central facts f_i:

$$sp(h, f_i) = \frac{\sum_{cp_j \in CP(h)} path(cp_j, f_i)}{|CP(h)|} \quad (4)$$

where $path(cp_j, f_i)$ is equal to 1 if there exists at least one path in the graph connecting the concept cp_j in the hypothesis to a concept in f_i, 0 otherwise. Therefore, the semantic plausibility of a fact is modelled as the percentage of concepts in the hypothesis h that have at least one path in the graph leading to f_i.

4 Abductive Inference

To derive the final explanation for a given hypothesis while conditioning on previously solved cases, we sum the *explanatory relevance* computed during the reuse phase with the *semantic plausibility* computed during the refine phase, pruning the graph considering only the top n central explanatory sentences and their linked grounding nodes (Fig. 2.3).

Given a set of alternative hypotheses $H = \{h_1, \ldots, h_n\}$, we adopt the model for abductive inference by generating an explanation for each hypothesis and selecting as an answer the one supported by the best explanation. To this end, we assign a score to each hypothesis h_i in H equal to the sum of the scores of the central facts included in the explanation for h_i.

Model	Overall	Easy	Challenge	Explanation
Sparse Retrieval				
BM25 (k = 1) (Clark et al., 2018)	41.21	44.96	32.99	yes
BM25 (k = 2)	43.62	48.54	32.73	
BM25 (k = 3)	45.87	50.76	35.05	
Dense Retrieval				
S-BERT (k = 1) (Reimers and Gurevych, 2019)	44.91	50.99	31.44	yes
S-BERT (k = 2)	45.79	51.45	33.25	
S-BERT (k = 3)	44.51	49.82	32.73	
Path-based				
PathNet (Kundu et al., 2019)	41.50	43.32	**36.42**	yes
Transformers				
BERT-large (Devlin et al., 2019)	46.19	52.62	31.96	no
RoBERTa-large (Liu et al., 2019)	**50.20**	**57.04**	35.05	no
Case-based Abductive NLI				
CB-ANLI BM25 (n = 1)	52.13	56.34	42.78	yes
CB-ANLI BM25 (n = 2)	**55.17**	60.42	**43.56**	
CB-ANLI BM25 (n = 3)	52.69	58.56	39.69	
CB-ANLI S-BERT (n = 1)	54.45	**61.23**	39.43	yes
CB-ANLI S-BERT (n = 2)	52.77	59.60	37.62	
CB-ANLI S-BERT (n = 3)	51.64	58.67	36.08	

Table 1: Accuracy on WorldTree (test-set) for *easy* and *challenge* questions. The parameter n corresponds to the number of central explanatory sentences considered by the models to compute the scores for each hypothesis.

5 Empirical Evaluation

Experimental Setup. We evaluate the Case-based Abductive NLI (CB-ANLI) framework on WorldTree (Jansen et al., 2018) and AI2 Reasoning Challenge (ARC) (Clark et al., 2018), two multiple-choice science question answering datasets designed to test abstractive commonsense and scientific inference. To perform the experiments, we transform each question-candidate answer pair into a hypothesis following the methodology described in (Demszky et al., 2018).

The knowledge bases required for the inference are populated using the WorldTree corpus (Jansen et al., 2018). The corpus contains a large set of commonsense and scientific facts ($\approx 10K$) that are used to construct explanations for multiple-choice science questions. The explanations include an average of 6 facts (and as many as ≈ 20), requiring challenging multi-hop inference to be generated. We store the individual facts for deriving the *Facts Embeddings* and consider the training questions ($\approx$ 1K) and their explanations as the set of previously solved cases (*Cases Embeddings*). For the refine phase, we dynamically extract the concepts in facts and hypotheses using WordNet (Miller, 1995) with NLTK[1]. Additional details are described in the appendix[2].

Sentene Encoders. We evaluate CB-ANLI using sparse and dense sentence encoders without additional training. The sparse version adopts BM25 vectors (Robertson et al., 2009), while the dense version employs Sentence-BERT (large) (Reimers and Gurevych, 2019; Thakur et al., 2020).

5.1 WorldTree

In this section, we present the results achieved on the WorldTree test-set (1247 questions). We report the accuracy of the case-based framework with different numbers n of central facts in the explanations. We compare the proposed framework against different categories of approaches: *Retrieval Solvers*, *Path-based Solvers*, and *Transformers*. The results in terms of question answering accuracy are reported in Table 1.

Retrieval Solvers. We employ stand-alone BM25 and Sentence-BERT (large) as sparse and dense retrieval solvers (Clark et al., 2018). Given

[1] https://www.nltk.org/_modules/nltk/corpus/reader/wordnet.html

[2] Code available at the following url: https://github.com/ai-systems/case_based_anli

RoBERTa + Retriever	Over.	Easy	Chal.
None	50.20	57.04	35.05
BM25 (k = 1)	57.06	60.88	48.57
BM25 (k = 2)	61.07	66.82	48.32
BM25 (k = 3)	61.23	65.54	51.12
S-BERT (k = 1)	55.85	61.46	43.41
S-BERT (k = 2)	60.91	66.82	47.80
S-BERT (k = 3)	56.96	62.04	45.73
CB-ANLI BM25 (n = 1)	61.71	66.82	50.38
CB-ANLI BM25 (n = 2)	**63.48**	**69.38**	50.38
CB-ANLI BM25 (n = 3)	62.43	67.77	50.63
CB-ANLI S-BERT (n = 1)	59.99	65.54	47.45
CB-ANLI S-BERT (n = 2)	63.32	67.98	**52.97**
CB-ANLI S-BERT (n = 3)	62.27	67.63	50.38

Table 2: Results for RoBERTa large fine-tuned on WorldTree and augmented with different explanation retrieval models.

an hypothesis h, the solvers retrieve the top k relevant facts for h using cosine similarity. The cosine similarity scores are then summed up to determine the best hypothesis. These baselines use the same encoders adopted by our model. However, we observe that CB-ANLI is able to outperform both sparse and dense retrieval models by up to $\approx 10\%$ accuracy, demonstrating the decisive role of the proposed case-based paradigm.

Path-based Solvers. We consider PathNet (Kundu et al., 2019) as a multi-hop inference baseline. This model constructs inference paths connecting question and candidate answer, and subsequently scores them through a neural encoder to derive the correct answer. We reproduce PathNet using the source code available online[3]. Contrary to CB-ANLI, PathNet does not adopt a Case-Based Reasoning framework to construct the explanations, considering each test hypothesis in isolation. We observe that CB-ANLI can significantly outperform PathNet with up to $\approx 13\%$ improvement overall and $\approx 7\%$ on challenge questions.

Transformers. We compare CB-ANLI against a BERT large (Devlin et al., 2019) and a RoBERTa large (Liu et al., 2019) baseline fine-tuned on the multiple-choice question answering task. We observe that on WorldTree the proposed approach is competitive with both RoBERTa and BERT (up to $\approx 5\%$ and $\approx 9\%$ improvement respectively).

[3] https://github.com/allenai/PathNet

Explainable Models	Accuracy
TupleInf (Khot et al., 2017)	23.83
TableILP (Khashabi et al., 2016)	26.97
DGEM (Clark et al., 2018)	27.11
KG^2 (Zhang et al., 2018)	31.70
Unsupervised AHE (Yadav et al., 2019a)	33.87
Supervised AHE (Yadav et al., 2019a)	34.47
ET-RR (Ni et al., 2019)	36.61
ExplanationLP (Thayaparan et al., 2021a)	40.21
AutoROCC (Yadav et al., 2019b)	41.24
Attentive Ranker (Pirtoaca et al., 2019)	**44.72**
Case-based Abductive NLI	
CB-ANLI BM25 (n = 1)	33.45
CB-ANLI BM25 (n = 2)	34.39
CB-ANLI BM25 (n = 3)	33.79
CB-ANLI S-BERT (n = 1)	**36.77**
CB-ANLI S-BERT (n = 2)	35.75
CB-ANLI S-BERT (n = 3)	34.30
CB-ANLI S-BERT (n = 1) + RoBERTa	44.02
CB-ANLI S-BERT (n = 2) + RoBERTa	**47.86**
CB-ANLI S-BERT (n = 3) + RoBERTa	42.40

Table 3: Performance on the AI2 Reasoning Challenge (ARC). We compare CB-ANLI with published explainable approaches that are fine-tuned only on ARC.

Transformers with Explanations We evaluate CB-ANLI as an evidence retrieval model by combining it with downstream Transformers. To perform this experiment, we augment the input of RoBERTa large with the explanations constructed for each hypothesis, and fine-tune the model to maximise the score for the correct answer. Table 2 reports the accuracy achieved with RoBERTa large when adopting CB-ANLI and stand-alone baselines as evidence retrievers. In general, we observe that evidence retrieval plays an important role for improving the performance of RoBERTa, and that the use of CB-ANLI can generate useful explanations for inference in combination with downstream language models.

5.2 ARC Challenge

To evaluate the generalisation of CB-ANLI on a broader set of challenge questions, we run additional experiments on the AI2 Reasoning Challenge (ARC) (Clark et al., 2018). Here, we keep the same configuration of hyperparameters. Table 3 reports the results achieved on the test-set (1172 challenge questions).

We observe that CB-ANLI with Sentence-BERT can generalise better on ARC. We attribute these results to the ability of Sentence-BERT to go beyond lexical overlaps, supporting generalisation on new

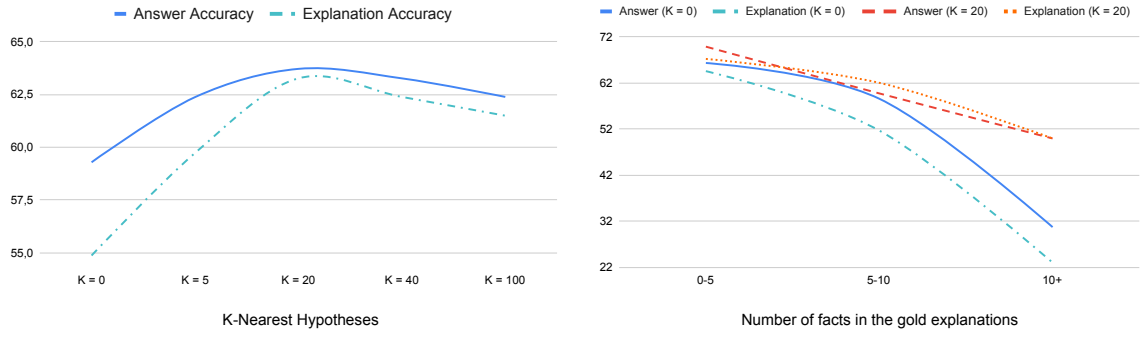

(a) (b)

Figure 3: Impact of the case-based framework on semantic drift. K represents the number of similar cases considered for computing the explanatory power (Worldree dev-set).

hypotheses with different surface forms. To show the impact of evidence retrieval on ARC, we fine-tune RoBERTa with the explanations constructed by the Sentence-BERT version.

For a fair comparison, we compare CB-ANLI against published explainable approaches that are fine-tuned only on ARC, without additional pre-training on related datasets (e.g. OpenBookQA (Mihaylov et al., 2018), RACE (Lai et al., 2017)). The results show that CB-ANLI S-BERT is third in the ranking, outperforming explanation-based systems based on Integer Linear Programming (ILP) (Khot et al., 2017; Khashabi et al., 2016) and pre-trained embeddings (Yadav et al., 2019a). At the same time, CB-ANLI obtains competitive results when compared with most of the fine-tuned neural approaches, including ET-RR (Ni et al., 2019). Moreover, when combined with RoBERTa, CB-ANLI achieves the best results among the considered approaches, improving on AutoROCC (Yadav et al., 2019b) and Attentive Ranker (Pirtoaca et al., 2019).

5.3 Ablation Study

We carried out an ablation study to investigate the impact of the CBR framework on downstream inference performance. To this end, we consider different versions of CB-ANLI by removing the impact of the reuse and refine phase. For the first, we remove the explanatory power term in equation 3. For the latter, we simply skip the refine phase ignoring the explanation graph construction and the semantic plausibility score to filter the central explanatory facts. The results of the study, reported in Table 4, demonstrate the key role of each phase to achieve the final inference performance.

Paradigm	Overall	Easy	Challenge
CB-ANLI BM25			
Retrieve-Reuse-Refine	**55.17**	**60.42**	**43.56**
Retrieve-Reuse	49.00	55.18	35.30
Retrieve-Refine	43.46	46.57	36.60
CB-ANLI S-BERT			
Retrieve-Reuse-Refine	**54.45**	**61.23**	**39.43**
Retrieve-Reuse	47.79	53.55	35.05
Retrieve-Refine	42.66	47.48	32.21

Table 4: Ablation Study on WorldTree (test-set) by removing the impact of the reuse and refine phases.

5.4 Impact on Semantic Drift

In this section, we investigate the impact of the CBR paradigm on semantic drift and how this affects the results on downstream reasoning tasks. To this end, we measure the performance of CB-ANLI when retrieving a different number K of previously solved hypotheses (notice that when $K = 0$ the model is equivalent to a non-case-based method). To evaluate the quality of the generated explanations, we leverage the WorldTree corpus as a gold standard, computing the *explanation accuracy* as the percentage of the best central explanations retrieved by the model that are part of the gold explanations in WorldTree. Since the explanations in the test-set are not publicly available, we perform this analysis on the dev-set.

Figure 3 (a) illustrates the change in answer and explanation accuracy on WorldTree with an increasing number K of similar cases. The graph demonstrates that the improvement in answer prediction is associated with improved explanation generation capabilities. Specifically, by conditioning the inference on an increasing number of similar hy-

Test Question	Prediction	Constructed Explanation ($K = 20, n = 1$)	Accurate
What force is needed to help stop a child from slipping on ice? (A) gravity, (B) friction, (C) electric, (D) magnetic	(B) friction	(1) counter means reduce; stop; resist; (2) ice is a kind of object; (3) slipping is a kind of motion; (4) stop means not move; **(5) friction acts to counter the motion of two objects when their surfaces are touching**	**Y**
What causes a change in the speed of a moving object? (A) force, (B) temperature, (C) change in mass (D) change in location	(A) force	**(1) a force continually acting on an object in the same direction that the object is moving can cause that object's speed to increase in a forward motion**	N
Weather patterns sometimes result in drought. Which activity would be most negatively affected during a drought year? (A) boating, (B) farming, (C) hiking, (D) hunting	(B) farming	(1) affected means changed; (2) a drought is a kind of slow environmental change; **(3) farming changes the environment**	N
Beryl finds a rock and wants to know what kind it is. Which piece of information about the rock will best help her to identify it? (A) The size of the rock, (B) The weight of the rock, (C) The temperature where the rock was found, (D) The minerals the rock contains	(A) The size of the rock	(1) a property is a kind of information; (2) size is a kind of property; (3) knowing the properties of something means knowing information about that something. **(4) the properties of something can be used to identify; used to describe that something**	**Y**
Jeannie put her soccer ball on the ground on the side of a hill. What force acted on the soccer ball to make it roll down the hill? (A) gravity, (B) electricity, (C) friction, (D) magnetism	(C) friction	(1) the ground means Earth's surface; (2) rolling is a kind of motion; (3) a roll is a kind of movement; **(4) friction acts to counter the motion of two objects when their surfaces are touching**	N

Table 5: Examples of explanations constructed by CB-ANLI. The underlined choices represent the correct answers. *Accurate* indicates whether the central fact **(bold)** is part of the gold explanation in the WorldTree corpus.

potheses, CB-ANLI is able to construct more accurate explanations, a feature that has a direct impact on downstream inference performance in question answering.

Figure 3 (b) shows the accuracy of the model on hypotheses requiring longer explanations when compared to a non-case-based version ($K = 0$). In general, a higher number of facts in the gold explanation is associated with a higher probability of semantic drift (Jansen and Ustalov, 2019). The graph confirms a strong relation between explanation accuracy and question answering accuracy, and demonstrates that the improvement obtained through the case-based framework is particularly evident on the most challenging inference problems (10+ facts in the explanations). This results allow us to conclude that the Case-Based Reasoning framework has a key role in alleviating semantic drift during multi-hop inference.

5.5 Faithfulness and Error Analysis

Finally, we present an analysis of the faithfulness of the model, investigating the relation between correct/wrong answer prediction and accurate/inaccurate explanations. Overall, we found that a total of 81.25% of the correct answers are derived from accurate explanations. This situation is illustrated in the first example in Table 5. On the other hand, a total of 18.75% of correct answers are derived from inaccurate explanations (second and third rows in the table). However, as shown in the second example, we observe that CB-ANLI can sometimes find alternative ways of constructing plausible explanations, considered inaccurate only because of a mismatch with the corpus annotation. The example number 4 shows the case in which an accurate explanation is not sufficient to discriminate the correct answer. We found this cases to occur for a total of 31.71% of incorrect answers. Finally, the last row describes the situation in which wrong answers are caused by inaccurate or spurious explanations (for a total of 68.29% of the wrong answers). This analysis demonstrates the interpretability and faithfulness of the framework, showing that its behaviour can be typically traced back to the quality of the generated explanations.

6 Related Work

Multi-hop inference for abstractive tasks is challenging as the general structure of the explanations cannot be derived from the surface form of the NLI problem. Previous work has demonstrated that models in this setting are affected by semantic drift – i.e., the construction of spurious explanations leading to wrong conclusions (Fried et al., 2015; Khashabi et al., 2019).

Existing approaches frame multi-hop inference as the problem of building an optimal graph, conditioned on a set of semantic constraints (Khashabi et al., 2018; Khot et al., 2017; Jansen et al., 2017; Khashabi et al., 2016; Thayaparan et al., 2021a), or adopting iterative methods, using sparse or dense encoding mechanisms (Yadav et al., 2019a,b; Pirtoaca et al., 2019; Kundu et al., 2019). Our model is related to previous work that leverages annotated explanations to reduce semantic drift (Xie et al., 2020; Jansen et al., 2018). However, this line of work is still limited to explanation regeneration (Jansen and Ustalov, 2019; Cartuyvels et al., 2020; Valentino et al., 2021, 2022; Thayaparan et al., 2021b), while the applicability of these re-

sources for downstream multi-hop NLI problems is yet to be explored. In this paper, we move a step forward, exploring how the impact of annotated explanations on semantic drift translates in improved downstream performance.

Our approach is related to previous work on Case-Based Reasoning (CBR) (Schank et al., 2014; Schank, 2013; De Mantaras et al., 2005). Similar to the retrieve-reuse-refine paradigm adopted in CBR systems, we employ encoding mechanisms to retrieve explanations for cases solved in the past, and adapt them in the solution of new problems. Recent work in NLP investigates the use of a similar paradigm via k-NN retrieval on training examples. Khandelwal et al. (2020b,a) adopt k-NN search to retrieve similar training examples and improve pre-trained language models and machine translation without additional training. Similarly, Das et al. (2021, 2020) propose a CBR framework for knowledge base reasoning, while Kassner and Schütze (2020) reuse similar cases to improve BERT (Devlin et al., 2019) on cloze-style QA. To the best of our knowledge, this is the first application of Case-Based Reasoning for explanation-based multi-hop Natural Language Inference (NLI).

The work presented in this paper is related to hybrid neuro-symbolic approaches for inference with natural language (Liu et al., 2020; Minervini et al., 2020; Jiang and Bansal, 2019; Chen et al., 2019; Dua et al., 2019; Xu et al., 2021; Weber et al., 2019). In this context, most of the existing approaches combine neural models with symbolic programs or reasoning modules. For instance, Jiang and Bansal (2019) propose the adoption of a Neural Module Network (Andreas et al., 2016) for multi-hop question answering by designing four atomic neural modules (Find, Relocate, Compare, NoOp). Weber et al. (2019) propose a methodology to perform multi-hop inference using a Prolog prover via the integration of sentence encoders and a weak unification mechanism. Differently from the methodology discussed in this paper, previous neuro-symbolic approaches have been generally applied to extractive tasks, where the inference steps (and, therefore, the explanation's structure) can be derived from a direct decomposition of the questions (Thayaparan et al., 2020).

7 Limitations

The adopted model of explanatory power relies on the availability of human-annotated explanations with specific features (e.g., explanatory facts reused across different training instances). However, these resources might not be available in real-world scenarios and are generally costly to develop. Moreover, since the explanatory power model relies on similarity measures and indicator functions, the model's ability to generalise might be sensitive to the incompleteness of the knowledge bases and the availability of representative explanations. We believe these limitations can be potentially alleviated by exploring the role of more abstract sentence representations within the CBR paradigm (Bergmann and Wilke, 1996).

In the current implementation of CB-ANLI, the refine phase adopts specific assumptions to model the abstraction process required for explanation generation. This process, in fact, is performed by assuming that abstraction at the concept level translates in a correct mapping between hypotheses and central explanatory sentences. However, contextual linguistic elements can still affect the overall meaning of the specific concept being abstracted, inducing the inclusion of spurious links between sentences. While contextual elements are considered during the precedent phases through the use of contextualised embeddings and similar cases, additional work is still required to guarantee the correctness of the abstraction process.

8 Conclusion

This paper presented CB-ANLI, a model that integrates multi-hop and Case-Based Reasoning (CBR) in a unified framework. We demonstrated the efficacy of the framework in complex abstractive and multi-hop NLI tasks. We believe this work can open new lines of research on hybrid neuro-symbolic models for explanation-based NLI, and plan to investigate the efficacy of the framework on architectures that adopt richer symbolic representations in combination with neural models, further exploring the role of abstraction in Case-Based Reasoning for improving robustness, generalisation, and explainability in NLI.

Acknowledgements

The work is partially funded by the SNSF project NeuMath (200021_204617). We would like to thank the Computational Shared Facility of the University of Manchester for providing the infrastructure to support our experiments.

References

Jacob Andreas, Marcus Rohrbach, Trevor Darrell, and Dan Klein. 2016. Neural module networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 39–48.

Ralph Bergmann and Wolfgang Wilke. 1996. On the role of abstraction in case-based reasoning. In *European Workshop on Advances in Case-Based Reasoning*, pages 28–43. Springer.

Chandra Bhagavatula, Ronan Le Bras, Chaitanya Malaviya, Keisuke Sakaguchi, Ari Holtzman, Hannah Rashkin, Doug Downey, Wen-tau Yih, and Yejin Choi. 2019. Abductive commonsense reasoning. In *International Conference on Learning Representations*.

Oana-Maria Camburu, Tim Rocktäschel, Thomas Lukasiewicz, and Phil Blunsom. 2018. e-snli: Natural language inference with natural language explanations. In *Advances in Neural Information Processing Systems*, pages 9539–9549.

Ruben Cartuyvels, Graham Spinks, and Marie Francine Moens. 2020. Autoregressive reasoning over chains of facts with transformers. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 6916–6930.

Xinyun Chen, Chen Liang, Adams Wei Yu, Denny Zhou, Dawn Song, and Quoc V Le. 2019. Neural symbolic reader: Scalable integration of distributed and symbolic representations for reading comprehension. In *International Conference on Learning Representations*.

Peter Clark, Isaac Cowhey, Oren Etzioni, Tushar Khot, Ashish Sabharwal, Carissa Schoenick, and Oyvind Tafjord. 2018. Think you have solved question answering? try arc, the ai2 reasoning challenge. *arXiv preprint arXiv:1803.05457*.

Rajarshi Das, Ameya Godbole, Shehzaad Dhuliawala, Manzil Zaheer, and Andrew McCallum. 2020. A simple approach to case-based reasoning in knowledge bases. In *Automated Knowledge Base Construction*.

Rajarshi Das, Manzil Zaheer, Dung Thai, Ameya Godbole, Ethan Perez, Jay Yoon Lee, Lizhen Tan, Lazaros Polymenakos, and Andrew McCallum. 2021. Case-based reasoning for natural language queries over knowledge bases. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 9594–9611.

Ramon Lopez De Mantaras, David McSherry, Derek Bridge, David Leake, Barry Smyth, Susan Craw, Boi Faltings, Mary Lou Maher, MICHAEL T COX, Kenneth Forbus, et al. 2005. Retrieval, reuse, revision and retention in case-based reasoning. *The Knowledge Engineering Review*, 20(3):215–240.

Dorottya Demszky, Kelvin Guu, and Percy Liang. 2018. Transforming question answering datasets into natural language inference datasets. *arXiv preprint arXiv:1809.02922*.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. Bert: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186.

Dheeru Dua, Yizhong Wang, Pradeep Dasigi, Gabriel Stanovsky, Sameer Singh, and Matt Gardner. 2019. Drop: A reading comprehension benchmark requiring discrete reasoning over paragraphs. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 2368–2378.

Daniel Fried, Peter Jansen, Gustave Hahn-Powell, Mihai Surdeanu, and Peter Clark. 2015. Higher-order lexical semantic models for non-factoid answer reranking. *Transactions of the Association for Computational Linguistics*, 3:197–210.

Peter Jansen, Rebecca Sharp, Mihai Surdeanu, and Peter Clark. 2017. Framing qa as building and ranking intersentence answer justifications. *Computational Linguistics*, 43(2):407–449.

Peter Jansen and Dmitry Ustalov. 2019. Textgraphs 2019 shared task on multi-hop inference for explanation regeneration. In *Proceedings of the Thirteenth Workshop on Graph-Based Methods for Natural Language Processing (TextGraphs-13)*, pages 63–77.

Peter Jansen, Elizabeth Wainwright, Steven Marmorstein, and Clayton Morrison. 2018. Worldtree: A corpus of explanation graphs for elementary science questions supporting multi-hop inference. In *Proceedings of the Eleventh International Conference on Language Resources and Evaluation (LREC 2018)*.

Yichen Jiang and Mohit Bansal. 2019. Self-assembling modular networks for interpretable multi-hop reasoning. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 4474–4484.

Jeff Johnson, Matthijs Douze, and Hervé Jégou. 2019. Billion-scale similarity search with gpus. *IEEE Transactions on Big Data*, pages 1–1.

Nora Kassner and Hinrich Schütze. 2020. BERT-kNN: Adding a kNN search component to pretrained language models for better QA. In *Findings of the Association for Computational Linguistics: EMNLP*

2020, pages 3424–3430, Online. Association for Computational Linguistics.

Urvashi Khandelwal, Angela Fan, Dan Jurafsky, Luke Zettlemoyer, and Mike Lewis. 2020a. Nearest neighbor machine translation. In *International Conference on Learning Representations*.

Urvashi Khandelwal, Omer Levy, Dan Jurafsky, Luke Zettlemoyer, and Mike Lewis. 2020b. Generalization through memorization: Nearest neighbor language models. In *International Conference on Learning Representations*.

Daniel Khashabi, Erfan Sadeqi Azer, Tushar Khot, Ashish Sabharwal, and Dan Roth. 2019. On the capabilities and limitations of reasoning for natural language understanding. *arXiv preprint arXiv:1901.02522*.

Daniel Khashabi, Tushar Khot, Ashish Sabharwal, Peter Clark, Oren Etzioni, and Dan Roth. 2016. Question answering via integer programming over semi-structured knowledge. In *Proceedings of the Twenty-Fifth International Joint Conference on Artificial Intelligence*, pages 1145–1152.

Daniel Khashabi, Tushar Khot, Ashish Sabharwal, and Dan Roth. 2018. Question answering as global reasoning over semantic abstractions. In *Thirty-Second AAAI Conference on Artificial Intelligence*.

Tushar Khot, Ashish Sabharwal, and Peter Clark. 2017. Answering complex questions using open information extraction. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 311–316.

Souvik Kundu, Tushar Khot, Ashish Sabharwal, and Peter Clark. 2019. Exploiting explicit paths for multi-hop reading comprehension. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 2737–2747.

Guokun Lai, Qizhe Xie, Hanxiao Liu, Yiming Yang, and Eduard Hovy. 2017. Race: Large-scale reading comprehension dataset from examinations. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 785–794.

Jiangming Liu, Matt Gardner, Shay B Cohen, and Mirella Lapata. 2020. Multi-step inference for reasoning over paragraphs. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 3040–3050.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Roberta: A robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692*.

Todor Mihaylov, Peter Clark, Tushar Khot, and Ashish Sabharwal. 2018. Can a suit of armor conduct electricity? a new dataset for open book question answering. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 2381–2391.

George A Miller. 1995. Wordnet: a lexical database for english. *Communications of the ACM*, 38(11):39–41.

Pasquale Minervini, Matko Bošnjak, Tim Rocktäschel, Sebastian Riedel, and Edward Grefenstette. 2020. Differentiable reasoning on large knowledge bases and natural language. In *Proceedings of the AAAI conference on artificial intelligence*, volume 34, pages 5182–5190.

Jianmo Ni, Chenguang Zhu, Weizhu Chen, and Julian McAuley. 2019. Learning to attend on essential terms: An enhanced retriever-reader model for open-domain question answering. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 335–344.

George Sebastian Pirtoaca, Traian Rebedea, and Stefan Ruseti. 2019. Answering questions by learning to rank-learning to rank by answering questions. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 2531–2540.

Nils Reimers and Iryna Gurevych. 2019. Sentence-bert: Sentence embeddings using siamese bert-networks. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 3973–3983.

Stephen Robertson, Hugo Zaragoza, et al. 2009. The probabilistic relevance framework: Bm25 and beyond. *Foundations and Trends® in Information Retrieval*, 3(4):333–389.

Roger C Schank, Alex Kass, and Christopher K Riesbeck. 2014. *Inside case-based explanation*. Psychology Press.

RP Schank. 2013. *Explanation patterns: Understanding mechanically and creatively*. Psychology Press.

Nandan Thakur, Nils Reimers, Johannes Daxenberger, and Iryna Gurevych. 2020. Augmented sbert: Data augmentation method for improving bi-encoders for pairwise sentence scoring tasks. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*.

Mokanarangan Thayaparan, Marco Valentino, and André Freitas. 2020. A survey on explainability in machine reading comprehension. *arXiv preprint arXiv:2010.00389*.

Mokanarangan Thayaparan, Marco Valentino, and André Freitas. 2021a. Explainable inference over grounding-abstract chains for science questions. In *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*, pages 1–12, Online. Association for Computational Linguistics.

Mokanarangan Thayaparan, Marco Valentino, Peter Jansen, and Dmitry Ustalov. 2021b. Textgraphs 2021 shared task on multi-hop inference for explanation regeneration. In *Proceedings of the Fifteenth Workshop on Graph-Based Methods for Natural Language Processing (TextGraphs-15)*, pages 156–165.

Marco Valentino and André Freitas. 2022. Scientific explanation and natural language: A unified epistemological-linguistic perspective for explainable ai. *arXiv preprint arXiv:2205.01809*.

Marco Valentino, Mokanarangan Thayaparan, Deborah Ferreira, and André Freitas. 2022. Hybrid autoregressive inference for scalable multi-hop explanation regeneration. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 36, pages 11403–11411.

Marco Valentino, Mokanarangan Thayaparan, and André Freitas. 2021. Unification-based reconstruction of multi-hop explanations for science questions. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*, pages 200–211.

Leon Weber, Pasquale Minervini, Jannes Münchmeyer, Ulf Leser, and Tim Rocktäschel. 2019. Nlprolog: Reasoning with weak unification for question answering in natural language. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 6151–6161.

Sarah Wiegreffe and Ana Marasovic. 2021. Teach me to explain: A review of datasets for explainable natural language processing. In *Thirty-fifth Conference on Neural Information Processing Systems Datasets and Benchmarks Track (Round 1)*.

Zhengnan Xie, Sebastian Thiem, Jaycie Martin, Elizabeth Wainwright, Steven Marmorstein, and Peter Jansen. 2020. Worldtree v2: A corpus of science-domain structured explanations and inference patterns supporting multi-hop inference. In *Proceedings of The 12th Language Resources and Evaluation Conference*, pages 5456–5473.

Weiwen Xu, Yang Deng, Huihui Zhang, Deng Cai, and Wai Lam. 2021. Exploiting reasoning chains for multi-hop science question answering. In *Findings of the Association for Computational Linguistics: EMNLP 2021*, pages 1143–1156.

Vikas Yadav, Steven Bethard, and Mihai Surdeanu. 2019a. Alignment over heterogeneous embeddings for question answering. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 2681–2691.

Vikas Yadav, Steven Bethard, and Mihai Surdeanu. 2019b. Quick and (not so) dirty: Unsupervised selection of justification sentences for multi-hop question answering. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 2578–2589.

Zhilin Yang, Peng Qi, Saizheng Zhang, Yoshua Bengio, William Cohen, Ruslan Salakhutdinov, and Christopher D Manning. 2018. Hotpotqa: A dataset for diverse, explainable multi-hop question answering. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 2369–2380.

Yuyu Zhang, Hanjun Dai, Kamil Toraman, and Le Song. 2018. Kgˆ 2: Learning to reason science exam questions with contextual knowledge graph embeddings. *arXiv preprint arXiv:1805.12393*.

A Hyperparameters

We adopted the following hyperparameters for CB-ANLI:

CB-ANLI BM25:

1. $\lambda = 0.83$
2. $K = 200$

CB-ANLI S-BERT:

1. $\lambda = 0.97$
2. $K = 40$

For the implementation of Sentence-BERT we adopt the following package https://pypi.org/project/sentence-transformers/ considering the *bert-large-nli-stsb-mean-tokens* model.

B Concepts Extraction

The concepts in facts and hypotheses are extracted using WordNet with NLTK: https://www.nltk.org/_modules/nltk/corpus/reader/wordnet.html. Specifically, given a sentence, we define a concept as a maximal sequence of words that corresponds to a valid synset in WordNet. This allows us to consider multi-words expressions such as *"living thing"* that frequently occur in the scientific domain.

C Transformers Setup

For the implementation of the Transformer model, we fine-tuned RoBERTa (*roberta-large*) for binary classification (bc) to predict a set of scores $S = \{s_1,\ s_2,\ ...,\ s_n\}$ for each candidate hypothesis in $H = \{h_1,\ h_2,\ ...,\ h_n\}$. The model receives as input an hypothesis h_i along with the explanation E_i for h_i. The model is optimised via cross-entropy loss to predict 1 for the correct hypothesis and 0 for the alternative hypotheses:

$$bc(\texttt{[CLS]} \mid\mid h_i \mid\mid \texttt{[SEP]} \mid\mid E_i) = s_i \quad (5)$$

The binary classifier is a linear layer operating on the final hidden state encoded in the `[CLS]` token. To answer the question q, the module selects the candidate answer c_a associated to the hypothesis with the highest score – i.e. $a = \text{argmax}_i\, s_i$. The model is implemented using Hugging Face (https://huggingface.com/) and fine-tuned using 4 Tesla V100 GPUs for 8 epochs in total. We adopted the following hyperparameters:

- batch size = 16
- learning_rate = 1e-5
- gradient_accumulation_steps = 1
- weight_decay = 0.0
- adam_epsilon = 1e-8
- warmup_steps = 0
- max_grad_norm = 1.0

D Source Code

The code adopted in the experiments is available at the following URL: https://github.com/ai-systems/case_based_anli.

E Data

The WorldTree corpus adopted in the experiments can be downloaded at the following url: http://cognitiveai.org/dist/worldtree_corpus_textgraphs2019sharedtask_withgraphvis.zip. The AI2 Reasoning Challenge (ARC) dataset can be downloaded at the following URL: https://allenai.org/data/arc. For the experiments on ARC, we adopted WorldTree V2 as our background knowledge: http://cognitiveai.org/explanationbank/

Semantic Structure based Query Graph Prediction for Question Answering over Knowledge Graph

Mingchen Li
Department of Computer Science
Georgia State University, Atlanta, USA
mli33@student.gsu.edu

Shihao Ji
Department of Computer Science
Georgia State University, Atlanta, USA
sji@gsu.edu

Abstract

Building query graphs from natural language questions is an important step in complex question answering over knowledge graph (Complex KGQA). In general, a question can be correctly answered if its query graph is built correctly and the right answer is then retrieved by issuing the query graph against the KG. Therefore, this paper focuses on query graph generation from natural language questions. Existing approaches for query graph generation ignore the semantic structure of a question, resulting in a large number of noisy query graph candidates that undermine prediction accuracies. In this paper, we define six semantic structures from common questions in KGQA and develop a novel Structure-BERT to predict the semantic structure of a question. By doing so, we can first filter out noisy candidate query graphs, and then rank the remaining candidates with a BERT-based ranking model. Extensive experiments on two popular benchmarks MetaQA and WebQuestionsSP (WSP) demonstrate the effectiveness of our method as compared to state-of-the-arts. The source code can be found at https://github.com/ToneLi/SSKGQA.

1 Introduction

Knowledge graph (KG) is a graph structured database (Miller, 1995), in which nodes represent entities (e.g., *Hedgehog in the Fog*, *Sergei Kozlov*), and edges reflect the relations between entities (e.g., *Hedgehog in the Fog - written_by - Sergei Kozlov*). Users can get crisp answers by querying KGs with natural language questions, and this process is called Question Answering over Knowledge Graph (KGQA). Recently, consumer market witnesses a widespread application of this technique in a variety of virtual assistants, such as Apple Siri, Google Home, Amazon Alexa, and Microsoft Cortana, etc.

Early works (Bordes et al., 2015; Golub and He, 2016) on KGQA mainly focus on simple questions, such as *where is the hometown of Obama?* This question only involves in one relation path (e.g., *hometown* or *birth-place*) in KG, and is relatively easy to solve. However, many questions in daily QA sessions are often more complex, manifested by multi-hop reasoning or questions with multiple constraints. Therefore, recently there is a flurry of interests on complex KGQA (Shi et al., 2021; Yadati et al., 2021).

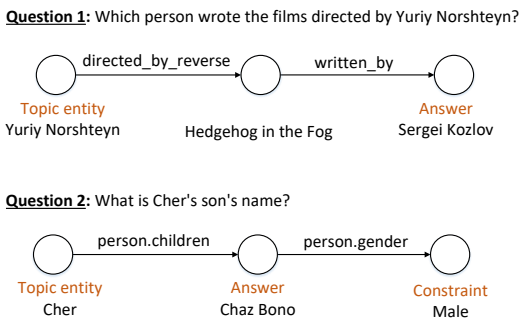

Figure 1: (Q1) Example question involving multi-hop reasoning, and (Q2) Example question with constraints

There are two types of complexity when dealing with complex KGQA, i.e., multi-hop questions and questions with multiple constraints (See Figure 1 for example). Question 1 in Figure 1 is a typical multi-hop question, to which the answer is related to `Yuriy Norshteyn` with two-hop relations: `directed.by.reverse` and `written.by`. In response to this challenge, Xu et al. (2019) enhances the traditional Key-Value Memory Neural Networks (KV-MemNNs) (Miller et al., 2016) for multi-hop question answering. They design a query updating strategy to decompose the question and predict the relevant relation paths at each hop. TransferNet (Shi et al., 2021) is an effective and transparent model for multi-hop questions; it attends to different parts of the question at each hop and computes activated scores for relation path prediction. Despite the promising results, it's still challenging for these models to

Proceedings of the 29th International Conference on Computational Linguistics, pages 1569–1579
October 12–17, 2022.

predict relation paths accurately at each hop, and thus suffer from error propagation over multi-hop reasoning. Similarly, Question 2 in Figure 1 is an example of question with constraints. Apparently, there is a single relation path between the topic entity *Cher* and the answer *Chaz Bono*, but the constraint of `person.gender=Male` must be satisfied. To handle this type of complex questions, many works built on the idea of query ranking are proposed (Yih et al., 2015; Lan and Jiang, 2020; Chen et al., 2020), which rank candidate queries by the similarity scores between question and candidate queries. Specifically, these ranking methods use query graphs to represent queries, and explore various strategies to generate candidate query graphs for ranking. Typical strategies assume the answers are within n hops of topic entity, and enumerate all the relation paths within n hops to generate candidate query graphs. Although this candidate generation strategy can yield all valid query graphs from a topic entity, they have two main limitations: (1) The generated candidate query graphs are very noisy. As shown in Figure 2(a), a candidate query graph with an incorrect structure is presented; this candidate query graph is generated by the traditional enumeration strategy but lacks of the constraint on `person.gender`, which can incur considerable error in query graph ranking (See Table 4). For the example in Figure 2(a), both `parent` and `birth.place` are relevant to the question; even though this candidate query graph has an incorrect semantic structure (to be defined in Sec. 3.2), it is still challenging for ranking models to demote it below the correct query graph – the one in Figure 2(b). (2) When building a ranking model to rank query graphs, recent works (Lan and Jiang, 2020) treat the candidate query graph and question as a sequence of words and leverage BERT (Devlin et al., 2018) to extract feature representation from its pooled output. However, this pooled output is usually not a good representation of the semantics of the input sequence (Khodeir, 2021). Therefore, improved ranking models are to be developed.

To mitigate the aforementioned issues, this paper proposes *SSKGQA*, a Semantic Structure based framework for complex KGQA. We represent both the multi-hop questions and questions with constraints in a way similar to query graphs[1] and rank the generated candidate query graphs by the similarity scores between question and candidate query graphs. Inspired by Chen et al. (2020), if the structure of a question is known in advance, the noise in candidate query graphs can be reduced significantly by filtering. Thus, SSKGQA first predicts the semantic structure of a natural language question, which is then used to filter out noisy candidate query graphs (which have incorrect structures).

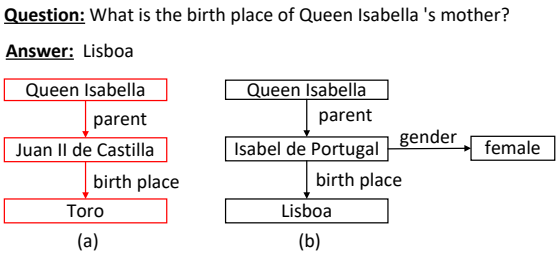

Figure 2: Example two-hop candidate query graphs for a question with constraints. (a) A candidate query graph with an incorrect semantic structure, (b) A candidate query graph with a correct semantic structure.

Specifically, we define six semantic structures based on the question topology that is introduced by Srivastava et al. (2021). With the defined semantic structures, our SSKGQA processes a natural language question in two stages. In the first stage, we develop a novel Structure-BERT to predict the semantic structure of a natural language question, which is then used to filter out noisy candidate query graphs and produce a set of query graph candidates that match the predicted structure. In the second stage, we rank the remaining candidate query graphs of a question by a BERT-based ranking model and identify the top-1 candidate query graph, which is then issued to retrieve the final answer from a KG. Our experiments demonstrate that this semantic structure based query graph prediction strategy is very effective and enables SSKGQA to outperform state-of-the-art methods.

Our main contributions are summarized as follows. (1) We propose SSKGQA, a semantic structure based method to predict query graphs from natural language questions. SSKGQA can handle both multi-hop questions and questions with constraints and is a unified framework for complex KGQA. (2) We develop a novel Structure-BERT to predict the semantic structure of each question, and a BERT-based ranking model with a triplet loss to identify the top-1 query graph candidate. (3) Compared to state-of-the-arts methods, our SSKGQA demonstrates superior performance on two popular complex KGQA benchmarks.

[1]For example, one candidate query graph for Question 1 in Figure 1 can be written as $Yuriy.Norshteyn \rightarrow directed.by.reverse \rightarrow y \rightarrow written.by \rightarrow x$.

2 Related Work

2.1 Multi-hop Question Answering

Current works on multi-hop question answering mainly focus on how to retrieve answers by calculating the relation paths step by step. In general, a right answer can be retrieved if the relation paths are identified correctly at each hop. Xu et al. (2019) enhances the traditional Key-Value Memory Neural Networks (KV-MemNNs) (Miller et al., 2016) and designs a query updating strategy to decompose the question and predict the relevant relation paths at each hop. TransferNet (Shi et al., 2021) calculates the relation path scores based on an updated question at each hop, but they leverage the attention mechanism to update question representations over multiple hops. More recently, Cai et al. (2021) introduces the dual process theory to predict the relation paths at each hop. Although these methods achieve promising results, they suffer from error propagation when predicting the relation paths step by step. To mitigate this issue, SSKGQA identifies the top-1 query graph by directly calculating the similarity scores between question and candidate query graphs (or similarly relation paths).

On the other hand, Sun et al. (2019a, 2018) incorporate external corpus to enhance the performance of KGQA. They focus on how to get the answers by constructing a subgraph for each question. A challenge of this method is that it is difficult to construct a subgraph around topic entity because we need to identify relevant entities from external corpus, and this process is error-prone. Saxena et al. (2020) predicts the answers by utilizing the KG embedding model. However, complex questions with long relation paths can reduce the learnability of KG embedding significantly. Our SSKGQA does not need external corpus to improve prediction accuracies and can solve complex multi-hop questions by a semantic structure based ranking.

2.2 Complex Questions with Constraints

For questions with constraints, a sequence of works focus on how to reach the answers by generating query graphs. Yih et al. (2015) enumerates all possible entities and relation paths that are connected to a topic entity to generate candidate query graphs, and uses a CNN-based ranking model to identify the query graph. Following a similar candidate query graph generation of (Yih et al., 2015), Maheshwari et al. (2019) propose a novel query graph ranking method based on self-attention. Qin et al. (2021) introduces a query graph generation method by using their proposed relation subgraphs. However, these methods largely ignore the noise when generating the candidate query graphs, which undermines the predictive performance during query graph ranking. To mitigate this issue, SSKGQA first predicts the semantic structure of a question, which is then used to reduce the noise in candidate query graphs.

2.3 Query Graph Ranking

Current research on KGQA mainly focuses on how to generate the candidate query graphs, and there are only a few works exploring how to rank the candidate query graphs. Lan and Jiang (2020) concatenates question and candidate query graph into a single sequence, and leverages BERT (Devlin et al., 2018) to process the whole sequence for ranking. However, Reimers and Gurevych (2019) show that this strategy is inefficient as it can incur a massive computation due to the combinatorial nature of concatenation of question and candidate query graphs, leading to duplicated calculation. Chen et al. (2020) explore GRUs to encode the question and query graph information, and utilize a hinge loss to learn a ranking model. However, GRUs can only learn a limited interaction among words in a sentence, while the global interactions among words has proven to be critical for text representation in various NLP applications (Khan et al., 2020). To solve the aforementioned issues, SSKGQA exploits separated BERT models to process questions and query graphs, respectively, and reuses the extracted features to avoid duplicated calculation and leverages a triplet loss to train a BERT-based ranking model.

3 Preliminaries

3.1 Query Graph

Query graph is a graph representation of a natural language question (Yih et al., 2015). See Figure 2 for example. A query graph usually contains four types of components: (1) a grounded entity, which is an entity in KG and is often the topic entity of a question, e.g., `Queen Isabella` in Figure 2. (2) an existential variable, which is an ungrounded entity, e.g., `Isabel de Portugal` in Figure 2. (3) a lambda variable, which is the answer to question but usually an ungrounded entity, e.g., `Lisboa` in Figure 2. (4) some constraints on a set of entities, e.g., `gender` in Figure 2. A ques-

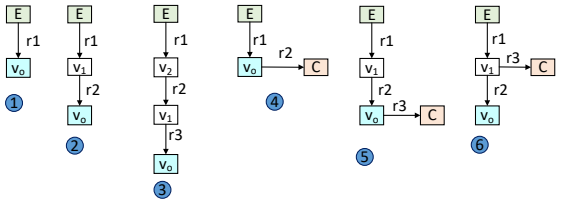

Figure 3: Six semantic structures defined in the paper. There are three semantic structures for questions in MetaQA: *(SS1, SS2, SS3)*, and five semantic structures for questions in WSP: *(SS1, SS2, SS4, SS5, SS6)*.

tion can be correctly answered if its query graph is built correctly and the right answer can be retrieved by issuing the query graph (represented by a SPARQL (Pérez et al., 2009) command) to a KG.

3.2 Semantic Structures

As observed by (Chen et al., 2020), if the structure of a question is known in advance, the noise in candidate query graphs can be reduced significantly by filtering. Thus, in this paper we define six semantic structures based on the question topology that is introduced by Srivastava et al. (2021). These six semantic structures are listed in Figure 3 and example questions for each semantic structure can be found in Figure 6 in Appendix. As we can see, a semantic structure is a graph that is an *abstract* of the query graphs of the same pattern. Typically, a semantic structure consists of four components $\{E, r, v, C\}$, where E denotes an entity, r refers to all types of relations, v is an existential variable, and C denotes a constraint.

To identify the semantic structure of a question, we can train a classifier for prediction. But first we need to annotate each training question with its semantic structure. Fortunately, this annotation can be achieved readily for questions in MetaQA and WebQuestionsSP (WSP) since these questions are either partitioned by number of hops or accompanied by the SPARQL commands. Details on question annotation are provided in Sec. 7.2 in Appendix. By annotating the questions in MetaQA and WSP, we found that these six semantic structures can cover 100% of questions in MetaQA, and 77.02% of questions in the test set of WSP as shown in Table 1. It is challenging to design additional semantic structures to cover 100% of questions in WSP because there are some unusual operators in WSP, such as `Or` and <=, which are difficult to map to a common semantic structure. Even though there is only a 77.02% coverage on the WSP test questions, our experiments show that SSKGQA already

Dataset	MetaQA			WSP
	Hop-1	Hop-2	Hop-3	
Train	100	100	100	91.37
Test	100	100	100	77.02
Dev	100	100	100	n/a

Table 1: Semantic structure coverage (%) for questions in the training, test and development sets of MetaQA and WSP.

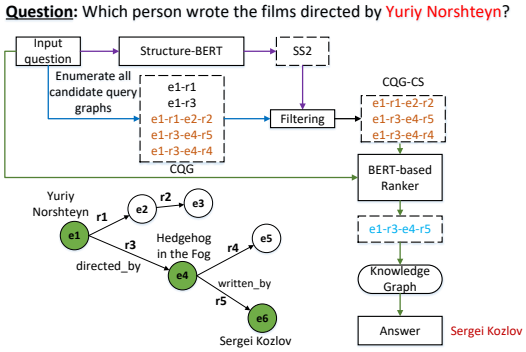

Figure 4: Overview of SSKGQA. A subgraph related to `Yuriy Norshteyn` from a KG is provided for illustration.

outperforms state-of-the-art methods on WSP. As a future work, we plan to explore new techniques to cover the remaining 22.98% of questions in WSP.

4 The Proposed Method

We first provide an overview of SSKGQA, and then discuss its main components: (1) Structure-BERT and (2) query graph ranking in details.

4.1 Overview

The overview of our proposed SSKGQA is depicted in Figure 4. Given a question q, following previous works (Saxena et al., 2020; Chen et al., 2020; Cai et al., 2021) we assume the topic entity of q has been obtained by preprocessing. Then the answer to q is generated by the following steps. First, the semantic structure of q is predicted by a novel Structure-BERT classifier. For the example in Figure 4, q is a 2-hop question and the classifier predicts its semantic structure as *SS2*. Second, we retrieve all the candidate query graphs (CQG) of q by enumeration[2], and use the predicted semantic structure *SS2* as the constraint to filter out

[2]For clarify, only 1-hop and 2-hop candidate query graphs are considered in this example.

noisy candidate query graphs and keep the candidate query graphs with correct structure (CQG-CS). Afterwards, a BERT-based ranking model is used to score each candidate query graph in CQG-CS, and the top-1 highest scored candidate is selected as the query graph g for question q. Finally, the selected query graph is issued to KG to retrieve the answer `Sergei Kozlov`.

4.2 Structure-BERT

Given a question q, we first need to predict its semantic structure, which is a multi-class classification problem that classifies question q to one of the six semantic structures defined in Figure 3.

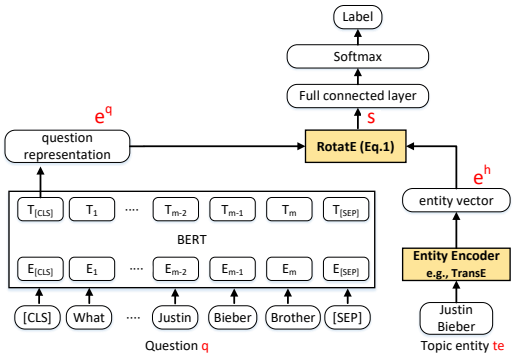

Figure 5: Structure-BERT: given a question and its topic entity, the model predicts the semantic structure of the question.

Figure 5 depicts the architecture of Structure-BERT. The input to Structure-BERT is question q and its topic entity te. The output of Structure-BERT is a probability distribution over six semantic structures, i.e., $p(y|q,\theta)$, where θ denotes the model parameters of Structure-BERT. Structure-BERT contains three sub-modules. **Question Encoder** encodes question q by a BERT language model (Devlin et al., 2018), and the final hidden state corresponding to token `[CLS]` is used as the question embedding e^q. **Entity Encoder** leverages a pre-trained knowledge embedding model, such as TransE (Bordes et al., 2013) or ComplEx (Trouillon et al., 2016), to extract the entity embedding e^h. Next, the extracted question embedding e^q and entity embedding e^h are fed to a **RotatE** module for information fusion. First, we utilize a pre-trained RotatE (Sun et al., 2019b) model to calculate a "tail" embedding e^t using e^h and e^q, and then fuse the topic entity, question and "tail" embeddings by combining e^h, e^q and e^t to a latent representation s by

$$\begin{aligned}
\bar{e}^h &= e^h_{lh} + i e^h_{hh} \\
\bar{e}^q &= e^q_{lh} + i e^q_{hh} \\
\bar{e}^t &= \bar{e}^h \times \bar{e}^q = e^h_{lh} e^q_{lh} - e^h_{hh} e^q_{hh} + i(e^h_{hh} e^q_{lh} + e^h_{lh} e^q_{hh}) \\
s &= e^h + e^q + e^t, \qquad (1)
\end{aligned}$$

where e^*_{lh} (e^*_{hh}) denotes the lower (higher) half of vector e^*. As such, $\bar{e}^*$ is a complex vector whose real (imaginary) part is from e^*_{lh} (e^*_{hh}). Therefore, we can convert between e^* and $\bar{e}^*$ readily.

Finally, the latent representation s is fed to a fully connected layer, followed by a softmax for classification. The whole network is fully differentiable and can be optimized by minimizing the traditional cross-entropy loss.

To train Structure-BERT, we need to annotate questions with their semantic structures to develop a training, validate and test set. As discussed in Sec. 3.2, this annotation can be conducted readily for questions in MetaQA and WSP. The details are provided in Sec. 7.2 in Appendix.

4.3 Query Graph Ranking

Another important component of SSKGQA is a BERT-based ranking model for query graph ranking that can be trained by a triplet loss (Schroff et al., 2015). Specifically, the ranking model has three inputs: (1) question q={`[CLS]`, w_1, w_2, $\cdots$, w_m, `[SEP]`}, where w_i is the i-th word of q; (2) positive query graph[3] g^p={`[CLS]`, u_1, u_2, $\cdots$, u_n, `[SEP]`}, where u_i is the i-th unit of a query graph that is split by space or special symbol such as ".". For example, given query graph (`Natalie Portman, film.actor.film, v`), u_1=`Natalie`, u_2=`Portman`, ... and u_6=`v`; and (3) negative query graph g^n that is a candidate query graph in CQG-CS except the positive candidate of the question.

We utilize a BERT model $f(.)$ to extract the semantic representations $f(q)$, $f(g^p)$, $f(g^n)$ for q, g^p, g^n, respectively. This BERT model is built on a pre-trained BERT from Hugging Face[4]; we add one extra multi-head attention layer on top of the hidden state of the pre-trained BERT (See the ablation study in Sec. 5). This BERT-based ranking model $f(.)$ is then optimized by minimizing the

[3]The positive query graph of a question can be found from the relation paths provided in MetaQA or the SPARQL command provided in WSP.

[4]https://huggingface.co/bert-base-uncased

triplet loss (Schroff et al., 2015)

$$\max(\|f(q)-f(g^p)\|-\|f(q)-f(g^n)\|+\alpha,0), \quad (2)$$

where $\|.\|$ denotes the Euclidean distance and α is a margin parameter, which we set to 1 as default.

During training, the triplet loss reduces the distance between $f(q)$ and $f(g^p)$, while enlarging the distance between $f(q)$ and $f(g^n)$. At inference time, we calculate the similarity scores between question and its candidate query graphs from CQG-CS, and choose the top-1 highest scored candidate as query graph g to retrieve final answer from KG.

5 Experiments

We evaluate the performance of SSKGQA on two popular KGQA benchmarks: MetaQA and WebQuestionsSP (WSP), and compare it with seven state-of-the-arts methods. Ablation study is also conducted to understand the effectiveness of different components of SSKGQA.

Our PyTorch source code is provided at `https://github.com/ToneLi/SSKGQA`. All our experiments are performed on Nvidia RTX GPUs.

5.1 Datasets

- **MetaQA** (Zhang et al., 2018) is a large scale KGQA dataset with more than 400k questions. It contains questions with 1, 2 or 3 hops. In our experiments, we use the vanilla version of the QA dataset. MetaQA also provides a KG from the movie domain with 43,233 entities, 9 relations and 134,741 triples.

- **WebQuestionsSP (WSP)** (Yih et al., 2016) is a small scale KGQA dataset with 5,119 questions which are answerable through Freebase KG. Since Freebase has more than 338,580,000 triples, for ease of experimentation we use a light version provided by Saxena et al. (2020). This smaller KG has 1.8 million entities and 5.7 million triples.

The statistics of training, development and test datasets of MetaQA and WSP is provided in Table 2. Compared to MetaQA, WSP is relatively small QA dataset even though its KG is much larger than that of MetaQA.

Dataset	Train	Dev	Test
MetaQA- hop1	96,106	9,992	9,947
MetaQA- hop2	118,980	14,872	14,872
MetaQA- hop3	114,196	14,274	14,274
WSP	3,304	–	1,815

Table 2: Statistics of the MetaQA and WSP datasets

5.2 Hyperparameter Settings

Structure-BERT We set the dropout rate to 0.1, batch size to 32, and use AdamW optimizer (Loshchilov and Hutter, 2017) with the learning rate of 5e-8. We also apply gradient clipping to constrain the maximum value of L_2-norm of the gradients to be 1. To extract the latent representations of topic entities, pre-trained ComplEx (Trouillon et al., 2016) and TransE (Bordes et al., 2013) are adopted for MetaQA and WSP, respectively.

BERT-based Ranking Model We add one extra multi-head attention layer on top of the hidden state of the pre-trained BERT. This extra multi-head attention layer contains three attention heads and a 3072-dim fully connected layer. The dropout rate is set to 0.5. We use AdamW Optimizer (Loshchilov and Hutter, 2017) with the learning rate of 2e-5. We also use gradient clipping to constrain the max L_2-norm of the gradients to be 1.

5.3 Baselines

We compare our SSKGQA against seven state-of-the-art complex KGQA models: 1) GraftNet (Sun et al., 2018), which answers the questions based on the subgraphs it creates. 2) PullNet (Sun et al., 2019a), which proposes a "pull" operation to retrieve the relevant information from KG and external corpus. 3) Key-Value Memory Network (KV-MemNN) (Miller et al., 2016), which uses key-value pairs as the memory units to answer questions. 4) EmbedKGQA (Saxena et al., 2020), which proposes a knowledge embedding method for Complex KGQA. 5) TransferNet (Shi et al., 2021), which utilizes an interpretable model for complex KGQA. 6) DCRN (Cai et al., 2021), which proposes a Bayesian network to retrieve the final answers. For MetaQA, we also include 7) VRN (Zhang et al., 2018) as the baseline, which proposes an embedding reasoning graph and utilizes variational inference to improve the performance of Complex KGQA.

5.4 Comparison with State-of-the-Arts

Table 3 reports the performances of SSKGQA and seven state-of-the-art methods on MetaQA and WSP. As can be seen, the performances of KV-

Model	Hop-1	Hop-2	Hop-3	WSP
KV-MemNN	96.2	82.7	48.9	46.7
VRN	97.5	89.2	62.5	-
GraftNet	97.0	94.8	77.7	66.4
PullNet	97.0	99.9	91.4	68.1
EmbedKGQA	97.5	98.8	94.8	66.1
DCRN	97.5	99.9	99.3	67.8
TransferNet	97.5	**100**	**100**	**71.4**
SSKGQA	**99.1**	99.7	99.6	**71.4**

Table 3: Hits@1 values of different KGQA methods on MetaQA and WSP. Hop-n denotes the hop-n questions of MetaQA.

MemNN are limited by the error propagation over multi-hop reasoning, i.e., as the number of hops increases, its performance is degraded significantly. GraftNet and PullNet perform similarly well on all datasets (expect MetaQA-hop3) as both of them rely on subgraphs to retrieve the answers. Compared to GraftNet, PullNet has much improved results on MetaQA-hop3, indicating that the proposed pull operation is more suitable to complex questions. EmbedKGQA achieves a good performance on MetaQA, but a relatively lower performance on WSP. This is because treating question as a relation path in a triple may introduce more noise especially when the question is more complex. Even though DCRN achieves the best performance on MetaQA-hop2, it still suffers from error propagation when inferring the reasoning paths. For more complex WSP questions, DCRN has a 3.6-point lower accuracy than that of our method. In general, TransferNet is the most competitive one to our SSKGQA. While both methods have the best results on WSP, SKGQA has an improved performance on MetaQA-hop1 over TransferNet, and is almost neck to neck on hop-2 and hop-3. Overall, SSKGQA outperforms or achieves comparable exact-match hits@1 performances to the other methods, demonstrating the effectiveness of our proposed method.

5.5 Ablation Study

We further investigate the effectiveness of different components of SSKGQA, including semantic structure based filtering, Structure-BERT, the BERT-based ranking model, etc.

	Hop-1	Hop-2	Hop-3	WSP
w/o SS	99.11	93.71	62.10	45.89
w/ SS	**99.26**	**99.03**	**95.69**	**58.51**

Table 4: Hits@1 values of SSKGQA w/ SS and w/o SS on MetaQA and WSP.

5.5.1 Impact of Semantic Structure based Filtering

One of the core ideas of SSKGQA is the semantic structure based filtering. In this section, we evaluate the effectiveness of this operator by enabling / disabling it and report the final performances of SSKGQA, which correspond to the w/ SS and w/o SS results in Table 4. For the purpose of illustration, when we enable the filtering (w/ SS), we assume that our Structure-BERT classifier can correctly predict the semantic structures of all the questions with a 100% accuracy, and therefore the impact of the filtering isn't affected by the accuracy of the classifier. For ease of experimentation, we use a BiGRU as the ranking model in this experiment.

Table 4 reports the impacts of the semantic structure based filtering. It can be observed that for simple questions, e.g., MetaQA-hop1, SSKGQA w/ SS and w/o SS have very similar performances. However, when the questions are more complex, SSKGQA w/ SS achieves significantly higher accuracies (sometimes over 10%) than SSKGQA w/o SS, demonstrating the effectiveness of the semantic structure based filtering for complex questions.

5.5.2 Accuracy of Structure-BERT

Structure-BERT plays a critical role in SSKGQA as it predicts the semantic structure of a question, which is then used to filter out noisy candidate query graphs. In this section, we evaluate the accuracy of Structure-BERT and compare it with other design choices.

Specifically, we compare the performance of Structure-BERT with four other classifiers, including BiGRU and three pre-trained language models: BERT (Devlin et al., 2018), DistilBERT (Budzianowski and Vulić, 2019) and CamemBERT (Martin et al., 2020)). For these four classifiers, they classify a question directly to one of the six semantic structures without considering topic entity and information fusion as in Structure-BERT.

Table 5 reports the classification accuracies of different classifiers on the questions from MetaQA

Model	Hop1	Hop2	Hop3	WSP
BiGRU	96.44	94.49	98.83	80.95
BERT	94.52	98.70	96.22	82.62
DistilBERT	95.66	98.30	97.02	83.37
CamemBERT	96.66	97.30	98.26	81.90
Structure-BERT	**99.24**	**99.87**	**99.73**	**86.97**

Table 5: Classification accuracies of different classifiers on predicting semantic structures of questions from MetaQA and WSP.

Model	Hop-1	Hop-2	Hop-3	WSP
CNN	97.70	99.21	92.91	50.24
BiGRU	98.87	98.95	95.43	56.51
BERT	**99.49**	99.26	99.54	71.02
BERT* (ours)	99.10	**99.69**	**99.64**	**71.40**

Table 6: Hits@1 values of different ranking models on MetaQA and WSP. BERT* denotes our BERT-based ranker.

and WSP. As can be seen, our Structure-BERT achieves nearly 100% accuracies on MetaQA and 86.97% accuracy on WSP, demonstrating the effectiveness of Structure-BERT on semantic structure prediction. Further, Structure-BERT consistently outperforms all the other classifiers by notable margins, indicating the importance of leveraging both question and topic entity for information fusion for semantic structure prediction. We also notice that the classification accuracy on WSP is much lower than that of MetaQA. This is likely due to: (1) the class imbalance issue of the WSP questions, and (2) much smaller number of training questions in WSP (3,304) than that of MetaQA (329,282). We will leave the further improvements of Structure-BERT on WSP to future works.

5.5.3 Performance of BERT-based Ranking Model

The BERT-based ranking model decides which candidate query graph is to be used to retrieve the final answer. Therefore, its performance is of the paramount importance to SSKGQA. In this section, we evaluate the effectiveness of our proposed BERT-based ranking model and compare it with other three ranking methods, including 1) CNN (Yih et al., 2015), which uses a CNN to learn the representation of question and candidate query graph for ranking. 2) BiGRU, which uses a BiGRU to learn the representation of question and candidate query graph for ranking. 3) BERT (Devlin et al., 2018), which uses a pre-trained BERT[5] to extract the representation of question and candidate query graph for ranking.

Table 6 reports the performances of different ranking models on MetaQA and WSP, where BERT* denotes our proposed BERT-based ranking model that has one extra multi-head attention layer on top of the hidden state of the pre-trained BERT. As we can see, the BERT-based ranking models (BERT and BERT*) outperform the traditional CNN or BiGRU based ranking models since the former can leverage large scale pre-trained BERT for transfer learning. Our BERT* further improves the performance of the pre-trained BERT due to the additional attention layer which enables model to reweight the attention values to different semantic units in the input and enhance the semantic representation of question and candidate query graphs for ranking.

To validate the design choices of our BERT-based model, we run additional ablation studies on different factors of our ranking model, such as number of negative query graphs for the triplet loss based training and number of heads in the added multi-head attention layer. The details are relegated to Sec. 7.3 in Appendix.

6 Conclusions

This paper introduces SSKGQA, a semantic structure based method to predict query graphs from natural language questions. Compared to prior query graph prediction based methods, SSKGQA filters out noisy candidate query graphs based on the semantic structures of input questions. To this end, we define six semantic structures from common questions of the KGQA benchmarks. A novel Structure-BERT classifier is then introduced to predict the semantic structure of each question, and a BERT-based ranking model with a triplet loss is proposed for query graph ranking. Extensive experiments on MetaQA and WSP demonstrate the superior performance of SSKGQA over seven state-of-the-art methods.

As for future work, we plan to investigate techniques to design additional semantic structures to cover the remaining 22.98% of questions in WSP. We would like also to improve Structure-BERT's accuracy on WSP by addressing the class imbalance and data scarcity issues of WSP.

[5]https://huggingface.co/bert-base-uncased

Acknowledgements

We would like to thank the anonymous reviewers for their comments and suggestions, which helped improve the quality of this paper. We would also gratefully acknowledge the support of VMware Inc. for its university research fund to this research.

References

Antoine Bordes, Nicolas Usunier, Sumit Chopra, and Jason Weston. 2015. Large-scale simple question answering with memory networks. *arXiv preprint arXiv:1506.02075*.

Antoine Bordes, Nicolas Usunier, Alberto Garcíaduran, Jason Weston, and Oksana Yakhnenko. 2013. Translating embeddings for modeling multi-relational data. In *International Conference on Neural Information Processing Systems*.

Paweł Budzianowski and Ivan Vulić. 2019. Hello, it's gpt-2–how can i help you? towards the use of pretrained language models for task-oriented dialogue systems. *arXiv preprint arXiv:1907.05774*.

Jianyu Cai, Zhanqiu Zhang, Feng Wu, and Jie Wang. 2021. Deep cognitive reasoning network for multi-hop question answering over knowledge graphs. In *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*, pages 219–229.

Yongrui Chen, Huiying Li, Yuncheng Hua, and Guilin Qi. 2020. Formal query building with query structure prediction for complex question answering over knowledge base. In *International Joint Conference on Artificial Intelligence (IJCAI)*.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*.

David Golub and Xiaodong He. 2016. Character-level question answering with attention. *arXiv preprint arXiv:1604.00727*.

Aisha Urooj Khan, Amir Mazaheri, Niels Da Vitoria Lobo, and Mubarak Shah. 2020. Mmft-bert: Multimodal fusion transformer with bert encodings for visual question answering. *arXiv preprint arXiv:2010.14095*.

Nabila A Khodeir. 2021. Bi-gru urgent classification for mooc discussion forums based on bert. *IEEE Access*, 9:58243–58255.

Yunshi Lan and Jing Jiang. 2020. Query graph generation for answering multi-hop complex questions from knowledge bases. *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*.

Ilya Loshchilov and Frank Hutter. 2017. Decoupled weight decay regularization. *arXiv preprint arXiv:1711.05101*.

Gaurav Maheshwari, Priyansh Trivedi, Denis Lukovnikov, Nilesh Chakraborty, Asja Fischer, and Jens Lehmann. 2019. Learning to rank query graphs for complex question answering over knowledge graphs. In *International semantic web conference*, pages 487–504. Springer.

Louis Martin, Benjamin Muller, Pedro Javier Ortiz Suárez, Yoann Dupont, Laurent Romary, Éric Villemonte de la Clergerie, Djamé Seddah, and Benoît Sagot. 2020. Camembert: a tasty french language model. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*.

Alexander Miller, Adam Fisch, Jesse Dodge, Amir-Hossein Karimi, Antoine Bordes, and Jason Weston. 2016. Key-value memory networks for directly reading documents. *arXiv preprint arXiv:1606.03126*.

George A Miller. 1995. Wordnet: a lexical database for english. *Communications of the ACM*, 38(11):39–41.

Jorge Pérez, Marcelo Arenas, and Claudio Gutierrez. 2009. Semantics and complexity of sparql. *ACM Transactions on Database Systems (TODS)*, 34(3):1–45.

Kechen Qin, Cheng Li, Virgil Pavlu, and Javed Aslam. 2021. Improving query graph generation for complex question answering over knowledge base. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 4201–4207.

Nils Reimers and Iryna Gurevych. 2019. Sentence-bert: Sentence embeddings using siamese bert-networks. *arXiv preprint arXiv:1908.10084*.

Apoorv Saxena, Aditay Tripathi, and Partha Talukdar. 2020. Improving multi-hop question answering over knowledge graphs using knowledge base embeddings. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 4498–4507.

Florian Schroff, Dmitry Kalenichenko, and James Philbin. 2015. Facenet: A unified embedding for face recognition and clustering. In *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*.

Jiaxin Shi, Shulin Cao, Lei Hou, Juanzi Li, and Hanwang Zhang. 2021. Transfernet: An effective and transparent framework for multi-hop question answering over relation graph. *arXiv preprint arXiv:2104.07302*.

Saurabh Srivastava, Mayur Patidar, Sudip Chowdhury, Puneet Agarwal, Indrajit Bhattacharya, and Gautam Shroff. 2021. Complex question answering on knowledge graphs using machine translation and

multi-task learning. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*, pages 3428–3439, Online. Association for Computational Linguistics.

Haitian Sun, Tania Bedrax-Weiss, and William W Cohen. 2019a. Pullnet: Open domain question answering with iterative retrieval on knowledge bases and text. *arXiv preprint arXiv:1904.09537*.

Haitian Sun, Bhuwan Dhingra, Manzil Zaheer, Kathryn Mazaitis, Ruslan Salakhutdinov, and William W Cohen. 2018. Open domain question answering using early fusion of knowledge bases and text. *arXiv preprint arXiv:1809.00782*.

Zhiqing Sun, Zhi-Hong Deng, Jian-Yun Nie, and Jian Tang. 2019b. Rotate: Knowledge graph embedding by relational rotation in complex space. *arXiv preprint arXiv:1902.10197*.

Théo Trouillon, Johannes Welbl, Sebastian Riedel, Éric Gaussier, and Guillaume Bouchard. 2016. Complex embeddings for simple link prediction. In *International Conference on Machine Learning*, pages 2071–2080. PMLR.

Kun Xu, Yuxuan Lai, Yansong Feng, and Zhiguo Wang. 2019. Enhancing key-value memory neural networks for knowledge based question answering. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*.

Naganand Yadati, RS Dayanidhi, S Vaishnavi, KM Indira, and G Srinidhi. 2021. Knowledge base question answering through recursive hypergraphs. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*, pages 448–454.

Scott Wen-tau Yih, Ming-Wei Chang, Xiaodong He, and Jianfeng Gao. 2015. Semantic parsing via staged query graph generation: Question answering with knowledge base.

Wen Tau Yih, Matthew Richardson, Chris Meek, Ming Wei Chang, and Jina Suh. 2016. The value of semantic parse labeling for knowledge base question answering. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*.

Yuyu Zhang, Hanjun Dai, Zornitsa Kozareva, Alexander J Smola, and Le Song. 2018. Variational reasoning for question answering with knowledge graph. In *Thirty-Second AAAI Conference on Artificial Intelligence*.

7 Appendix

7.1 Example Questions and their Semantic Structures

We provide six example questions and their corresponding semantic structures. These examples are selected from the MetaQA and WSP benchmarks. E denotes a topic entity, r refers to all types of relations, v is an existential variable, and C denotes a constraint.

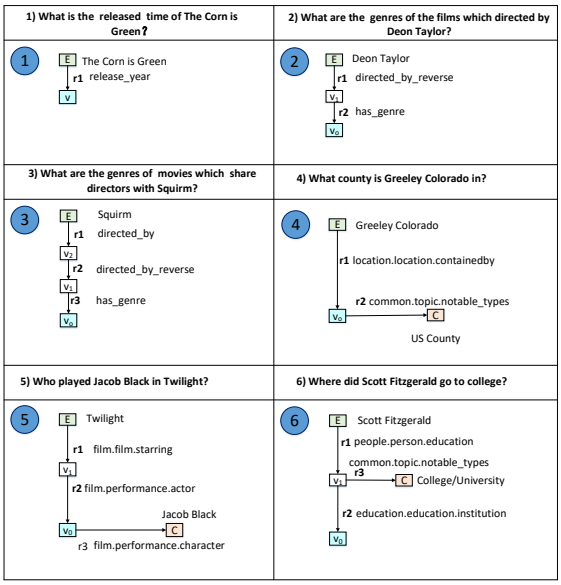

Figure 6: Examples questions and their semantic structures

7.2 Semantic Structure Annotation for Training Questions

We need to annotate each training question to a semantic structure in order to train Structure-BERT for semantic structure prediction. Here we describe how we can automatically annotate each training question in MetaQA and WSP.

7.2.1 MetaQA

Annotating semantic structure for each training question in MetaQA is straightforward. Since training questions are organized by number of hops, all training questions with 1-hop are labeled to SS1, the 2-hop ones to SS2, and the 3-hop ones to SS3, respectively.

7.2.2 WSP

The SPARQL command for each question is provided in WSP. Thus, we can readily extract the query graph of a question from its SPARQL command. See an example in Figure 7, where a query graph is shown on the left and the corresponding

SPARQL command is shown on the right; the correspondences between two parts are marked by red lines. Once the query graph is extracted, each training question can be readily annotated to a semantic structure based on its query graph.

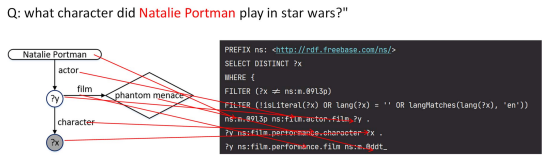

Figure 7: Mapping from a SPARQL command to its query graph

7.3 Additional Ablation Study

Number of Neg.	1	5	10	50
hits@1	64.57	68.87	70.08	70.79
Number of Neg.	100	200	300	500
hits@1	**71.29**	71.23	**71.29**	70.58

Table 7: Hits@1 values on WSP of our BERT-based ranking model when trained with different number of negative query graphs.

Number of negative query graphs Given a question, a plenty of negative query graphs can be generated by enumeration from a KG. By analyzing the questions in WSP, we found that the maximal number of negative query graphs that can be extracted for WSP is around 500. We need to determine a proper number of negative samples to train the ranking model with the triplet loss. To this end, we evaluate the performances of our BERT-based ranker when trained with different numbers of negative samples. The results are reported in Table 7, where 8 different number of negative samples are considered. As we can see, when number of negative samples is over 100, the ranking model achieves improved hits@1 performances; when $n = 100$ our BERT-based ranking model yields the best hits@1 with a good run-time performance.

Number of Heads	1	3	6
hits@1	71.29	**71.40**	70.63

Table 8: Hits@1 values on WSP of our BERT-based ranking model with different number of attention heads in the added attention layer.

Number of heads Another design choice for the extra multi-head attention layer in our BERT-based ranker is the number of attention heads. We therefore evaluate the performance of our BERT-based ranker with 3 different number of attention heads. The results in Table 8 shows that when the number of attention heads is 3, our BERT-based ranker achieves the best hits@1.

Repo4QA: Answering Coding Questions via Dense Retrieval on GitHub Repositories

Minyu Chen[1], **Guoqiang Li**[1]*, **Chen Ma**[2], **Jingyang Li**[1], **Hongfei Fu**[1]
[1]Shanghai Jiao Tong University, Shanghai, China
[2]City University of Hong Kong, Hong Kong S.A.R.
{minkow,li.g,lijjjjj}@sjtu.edu.cn
chenma@cityu.edu.hk, fuhf@cs.sjtu.edu.cn

Abstract

Open-source platforms such as GitHub and Stack Overflow both play significant roles in current software ecosystems. It is crucial but time-consuming for developers to raise programming questions in coding forums such as Stack Overflow and be navigated to actual solutions on GitHub repositories. In this paper, we dedicate to accelerating this activity. We find that traditional information retrieval-based methods fail to handle the long and complex questions in coding forums, and thus cannot find suitable coding repositories. To effectively and efficiently bridge the semantic gap between repositories and real-world coding questions, we introduce a specialized dataset named Repo4QA, which includes over 12,000 question-repository pairs constructed from Stack Overflow and GitHub. Furthermore, we propose QuRep, a CodeBERT-based model that jointly learns the representation of both questions and repositories. Experimental results demonstrate that our model simultaneously captures the semantic features in both questions and repositories through supervised contrastive loss and hard negative sampling. We report that our approach outperforms existing state-of-art methods by 3%-8% on MRR and 5%-8% on P@1. [1]

1 Introduction

With the increasing popularity of software development, Stack Overflow and GitHub (two large-scale communities widely recognized in open source ecosystems) have attracted growing research interests. As the idiom goes, "Don't reinvent the wheel". Tackling programming problems with existing codes and documents is more effective and economical, since various resources in repositories can provide more helpful information than text-formed answers. Specifically, developers can get help and advice to solve the technical challenges they face, and be provided a variety of solutions and tools in repositories on GitHub. As a vast number of challenges have already been considered and settled by the community, sophisticated schemes posted on GitHub can help satisfy requirements or solve problems discussed in Stack Overflow. Naturally, many answers in Stack Overflow provide links to GitHub repositories, and a large amount of these answers are acknowledged to be high-quality and helpful by the community. This phenomenon is worth investigating to determine its contribution to the efficiency improvement and the code reuse of the open source ecosystem.

Motivated by the interplay between Stack Overflow and GitHub, we introduce a novel question-repository matching task. Given a natural-language-formed question in the programming domain, we aim to search for the most relevant GitHub repository from all candidate repositories as the answer. Figure 1 illustrates the interaction between a question and a repository, where the key information for problem-solving is framed.

To this end, we introduce Repo4QA, a dataset consisting of 12,995 question-repository pairs[2] for complex coding question solving. The questions are collected from Stack Overflow, and the repositories are crawled from GitHub through the hyperlinks provided in corresponding Stack Overflow answers. Each repository is instrumental for trouble-shooting confirmed by forum users with upvotes. Such that proper repositories can be used as answers to coding questions.

Different from previous information retrieval (IR) tasks, the challenge of the proposed task lies in the long-form questions. Typical IR methods and tricks, such as query expansion, are designed for short and keyword-based user inputs, while our task focuses on questions written in natural lan-

*Corresponding author.

[1]Our dataset and code are available at https://github.com/Minkow/Repo4QA

[2]We also call them query-document pairs from the perspective of IR.

Proceedings of the 29th International Conference on Computational Linguistics, pages 1580–1592
October 12–17, 2022.

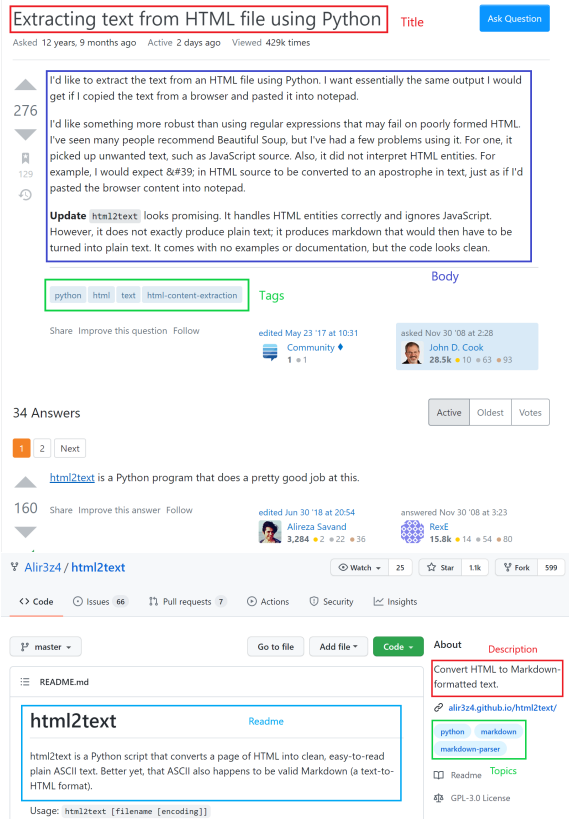

Figure 1: An example of interaction between question at Stack Overflow and repository at GitHub.

guages containing semantic information, which is harder to resolve.

Different from code searching task (Husain et al., 2019; Cambronero et al., 2019), our task is supposed to find a reasonable semantic alignment between two long-form sentences. Questions and repositories have more complex structures and richer information than short-form web queries and code snippets. Compared with community-based QA task (Qiu and Huang, 2015a; Zhao et al., 2017), our task is a cross-platform task resulting in semantic gaps between questions and answers, which is more challenging for traditional IR-based methods such as BM-25 (Robertson et al., 1995). Besides, unlike common questions, questions about programming are more challenging and specialized. The terminology of programming is widely used to form questions; there are even some questions described with codes. Moreover, complex models considering more interactions between QA pairs are more computationally expensive in our task. Finally, the interactions inside queries or documents corpus should not be neglected. The relationship of the elements in the same semantic category is informative while discriminating positive samples from negatives.

To address the aforementioned issues, we propose a novel jointly learning scheme, **QuRep**, for **Qu**estion and **Re**pository **p**airs. QuRep employs CodeBERT (Feng et al., 2020) as the initial weight of its transformer-based encoder. Our work leverages the supervised contrastive loss for better separating elements from each other in the representation space. An unsupervised hard negative sampling strategy enables better discrimination performance. The shared weights between encoders reduce the computation cost, while the relevance between questions and answers can be ranked by the similarity score of embedding vectors. With experimental evaluation and comprehensive analyses, we show that our method strongly outperforms baselines.

To summarize, the main contributions of this paper are concluded into three points respectively.

- *Dataset*: A novel cross-platform Question-Answering task is presented, aiming at answering real-world programming questions with existing GitHub repositories. We especially collect a dataset, Repo4QA, for this task.
- *Methodology*: A practical joint embedding model, QuRep, is proposed to optimize the classification and contrastive loss, which can represent both questions in language and repositories.
- *Experiment*: Experimental results demonstrate the effectiveness and efficiency of the proposed QuRep model compared with baselines.

2 Related Work

Datasets. Existing datasets in the programming domain focus on text-code interaction. CodeSearchNet (Husain et al., 2019), Deep Code Search (Gu et al., 2018) and CoDesc (Hasan et al., 2021) collect large-scale corpus of code snippet with corresponding descriptions. CoSQA (Huang et al., 2021) collects pairs of web query and function code for code question answering. Stack Overflow resources are mined (Yin et al., 2018) as long-form natural language queries to retrieval code snippets (Nie et al., 2016; Yao et al., 2018; Heyman and Van Cutsem, 2020). CodeXGLUE (Lu et al., 2021) includes text-to-code generation task and code memorization task. The only text-to-text task

is documentation translation in CodeXGLUE. Besides, LinkSO (Liu et al., 2018) discovered the similarity between repositories.

Neural Matching Networks. Ranking methods are widely used in text matching and semantic search tasks, such as community-based question answering (Qiu and Huang, 2015b; Zhang et al., 2021b), retrieval-based question answering (Qu et al., 2020; Cohen et al., 2018), and visual question answering (Lee et al., 2020). Specifically, in the programming domain, traditional IR-based ranking models regard code as text and match keywords in queries (Bajracharya et al., 2006). Recently, deep learning based methods represent coding questions and answers with vectors and leverage similarities to rank answers (Gu et al., 2018; Cambronero et al., 2019; Wan et al., 2019).

Considering the computational cost for matching, representation-based learning approaches (Huang et al., 2013) encode a query paired with a document separately into a vector and judge the relevancy by the similarity of vectors. Towards a better representation of repositories, paper2repo (Shao et al., 2020) maps the embeddings of academic papers and repositories into the same space for ranking and recommendation. Very recently, pre-trained models, including CodeBERT (Feng et al., 2020) that trained from data in the programming domain, have been applied to improve representation learning.

LM Based Retrieval and Ranking. Pretrained language models (Kenton and Toutanova, 2019; Liu et al., 2019b) dramatically advance the state of the art on various NLP tasks. However, limitations on text length and the trade-off of effectiveness and efficiency are important issues, as the cross-attention operations are too expensive in pair-wise cross-encoders. Recent work (Karpukhin et al., 2020; Xiong et al., 2020; Khattab and Zaharia, 2020; Nie et al., 2020; Gao et al., 2021a) try to reduce the computational interaction between query and document and move it to the online rerank procedure. By storing the representation of documents offline, these methods facilitate cheap runtime costs while achieving promising results on retrieval tasks.

Moreover, contrastive learning on pre-training models has recently been broadly applied to several sentence-level tasks. SBERT (Reimers and Gurevych, 2019) uses siamese and triplet network to derive embedding of two sentences, then fine-tune the model to yield useful sentence embeddings. Some work aims to improve BERT sentence embeddings in an unsupervised way (Gao et al., 2021b; Kim et al., 2021) by data augmentation.

Dataset	Type	Samples	Avg. length
Large	Question	12,995	7.97 + 104.50
	Repository	9,954	9.79 + 572.58
Small	Question	3,766	8.02 + 105.73
	Repository	2,862	9.77 + 688.03

Table 1: The statistics of Repo4QA dataset. Avg. length means title length + body length for question, and description length + Readme length for repository. Readme file has the maximum character length of 8192.

3 Dataset Description

3.1 Repo4QA Dataset

Questions. We collect complex programming questions from the well-known coding forum Stack Overflow. Stack Overflow provides data dump [3] from 2014 to 2021. To avoid ambiguous answers such as "We can deploy A after B to tackle this", answers less than 200 characters that contain only a hyperlink to a GitHub repository are selected.

Responders are required not only to get through the requirements raised by questioners, but also to be familiar with repositories stored on GitHub. The goal of our research is to fill in the gap between the questioner and various open source tools.

To filter out answers without specific repository-for-solution and repository-structured intent such as bug-reporting discussion, we discard answers linking to particular resources in repositories such as issues, commits, and releases (e.g., the question is "Assets serving paths in Rails 3.1 - how to customize it?" and the answer is "GitHub.com/chriseppstein/compass/issues/337 - there is a big discussion about it"). We control the quality and correctness of answers by only mining posts with at least one upvote. After removing answers with inaccessible GitHub repository links, these answers and corresponding questions compose 12,995 QA pairs with 12,713 unique questions. Codes are marked with *[code]* token to help our model learn the combination of natural language and code in questions.

[3] https://archive.org/details/stackexchange

Repositories. We crawl repositories through the GitHub API with given hyperlinks in answers. For our task, we collect basic information such as the name, description, topics (also called tags), and stars of a repository. The description is short textual documentation to describe a repository briefly. Topics are keywords to classify a repository. For instance, the tag "python" indicates the programming language that the repository uses, and the tag "deep-learning" shows that the repository is used in the deep learning domain. In addition, the Readme file is also obtained for detailed documentation. Most repositories introduce the main contribution and usage in the head of the Readme file. We investigate 50 repositories randomly and find that the most informative part of a Readme file is the starting 2 or 3 paragraphs, which is far less than 512 tokens. Hence, we cut the first 8192 characters of a Readme file after text cleaning to represent the repository in natural language. Likewise, 9,663 unique repositories are investigated in this step, in which 2,862 repositories have at least one topic.

Construction. Questions and repositories are aligned according to QA pairs mined in questions to constitute a QA pair sample. For each pair, the original answer is replaced by the repository mentioned. All samples in the small dataset have at least one topic, while most repositories in the large dataset lack topics. The statistics of both datasets are listed in Table 1.

Comparison. To the best of our knowledge, this is the first dataset applied to solving complex realistic programming problems with existing web resources. In this part, we conduct a comparison between Repo4QA and datasets from two aspects: 1) Code intelligence and 2) Community-based QA. Datasets in the programming domain (Yao et al., 2018; Nie et al., 2016; Yin et al., 2018; Heyman and Van Cutsem, 2020) tend to adopt only the title of the question in Stack Overflow or a short description as the query. However, for some complex questions raised in Stack Overflow, the title is too short to fully reveal the logic or semantics of those complex questions. On the other hand, the details of those questions are often clearly illustrated in the question body, and the neglect of the question body will lead to an incomplete understanding of the question. We also find that in Community-based QA datasets (Lyu et al., 2019; Zhang et al., 2021b), their QA pairs are on the same webpage, while our Repo4QA aims to bridge the semantic gap between natural languages and GitHub repositories, which is challenging to represent due to its length (averagely over 500 words) and heterogeneous text structures. In comparison, the average length of question/answer on the CoSQA dataset (Huang et al., 2021) is 6.60/71.51, and 7.86/81.6 on the CQA-SO (Lyu et al., 2019) dataset. A detailed comparison can be found in appendix A.

4 Methodology

4.1 Task Description

Before diving into the details of our model, we first describe some notations used in the answer selection problem. Each question consists of a title, content, and tags, while each repository served as an answer candidate has descriptions and documentation. Note that tags may not be contained in some repositories. Given a question in a natural language question set $q \in Q$, and a set of repositories $R = \{r_1, r_2, \cdots, r_n\}$ from GitHub, our main task is to find the most helpful repository $r \in R$ to solve the question q.

Due to the limitation of computation resources in real-world applications, joint embedding is an effective and efficient way to find repositories related to the raised question. Ideally, we train a model that jointly learns the embedding of Q and R with a transformer-based encoder network. Specifically, given any question and its dense representation q_i and repositories' embedding r_i^+, r_i^-, where r_i^+ is one of the answer to q_i, and r_i^- is not related to q_i. We expect the model to distinguish r_i^+ from others via similarity metrics, that is, to make $s(q_i, r_i^+)$ and $s(q_i, r_i^-)$ satisfying the inequality: $s(q_i, r_i^+) > s(q, r_i^-)$, where s denotes a similarity function, e.g., cosine similarity or Euclidean distance-based similarity. Then we rank all the answers for a given question according to the similarity between them.

4.2 Model Architecture

We present our QuRep in Figure 2. Different from the natural language in common domains, the language used in the programming domain contains various out-of-vocabulary words, e.g., "flask" is a tool used for web in Python programming. To solve this problem, we leverage CodeBERT as our text encoder. CodeBERT is a bi-modal pre-trained RoBERTa-based (Liu et al., 2019b) model for natural language (NL) and programming language

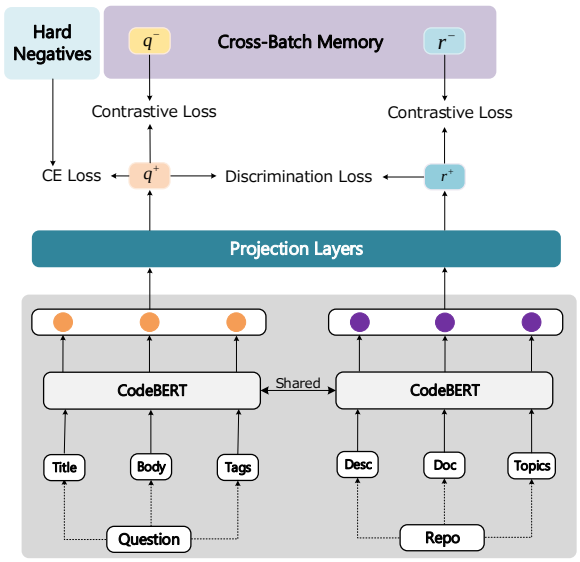

Figure 2: The QuRep applies a weight-sharing CodeBERT for encoding questions and repositories. Similarity is computed between model outputs and embeddings stored in Cross-Batch Memory for model training.

(PL) tasks. It is a bidirectional Transformer with 12 layers, 768 dimensional hidden states and 12 attention heads pre-trained on the large-scale CodeSearchNet (Husain et al., 2019) corpus. CodeBERT achieves the state-of-the-art in most NL-PL, tasks such as the natural language code search and code documentation generation. By expanding the vocabulary of RoBERTa, CodeBERT can properly represent programming terms that occur in the training corpus, especially for word-combining terminologies commonly used in programming. For example, "pyflask" is an OOV word in most models' vocabulary, but the WordPiece encoding will split "pyflask" into "py" and "flask".

For each question with title, body, and tags, we put a [CLS] token at the beginning of these three parts and separate them by [SEP] after tokenization. Then we feed the tokenized sequences into CodeBERT to acquire pooled contextualized representations of them, respectively. A [Q] is placed in the start to identify the question. In practice, we adopt the mean pooling value of contextual representations as the output of CodeBERT. Similarly, for each repository, we conduct the same manner as its description, Readme documentation, and topics. The weights are shared between the model for both question encoding and answer encoding.

As few Readme files exceed the token length limit of 512 in CodeBERT, we only take the first 509 tokens (3 tokens are left for [Q]/[A], [CLS] and [SEP]) of the Readme file as the documentation. In practice, the head content of Readme file is descriptive and summative enough, which is more informative than the usage description and changelogs in the later part of Readme file.

4.3 Loss Functions

Our approach considers three kinds of losses, namely discrimination loss, supervised contrastive loss, and classification loss. These loss functions are designed for specific tasks to better represent questions and repositories.

Discrimination Loss. Given a randomly sampled minibatch of positive pairs $\mathcal{D} = (q_i, r_i)^N$, the goal of Discrimination loss is to separate the answer r_i apart from all other $2N-2$ repositories in the same batch $\mathcal{D}$. It matches the target to satisfy the inequality : $s(q^+, r^+) > s(q^+, r^-)$. where $s(\cdot)$ represents the similarity. We choose cosine similarity as the metric. Following previous work (Karpukhin et al., 2020), the discrimination loss is defined as the negative log likelihood of positive pairs of instances:

$$\mathcal{L}_D = -\log \frac{\mathrm{e}^{s(q_i,r_i)/\tau}}{\mathrm{e}^{s(q_i,r_i)/\tau} + D}, \tag{1}$$

where τ refers to the temperature parameter, $D = \sum_{j \neq i} \mathrm{e}^{s(q_j,r_j)/\tau}$ denotes the normalization part.

Supervised Contrastive Loss. The discrimination loss discussed above focuses on modeling the distance between questions and answers. However, compared with other QA or NLI tasks (Ren et al., 2021; Zhang et al., 2021a), our questions are more complex and long-formed, which means different questions implicate diverse semantic information. We insist that a good dense representation model should not only control the distance of questions and answers, but can also represent the semantic difference between questions and repositories. Hence, we further leverage the supervised contrastive loss, to enforce the model learn to distinguish similar questions and answers, that is, $sim(q^+, r^+) > sim(q^+, q^-)$, and $sim(q^+, r^+) > sim(r^+, r^-)$. We also employ the negative log likelihood to achieve the supervised contrastive loss at the instance level:

$$\begin{aligned} \mathcal{L}_{Cq} &= -\log \frac{\mathrm{e}^{s(q_i,r_i)/\tau}}{\mathrm{e}^{s(q_i,r_i)/\tau} + Cq}, \\ \mathcal{L}_{Cr} &= -\log \frac{\mathrm{e}^{s(q_i,r_i)/\tau}}{\mathrm{e}^{s(q_i,r_i)/\tau} + Cr}, \end{aligned} \tag{2}$$

where $Cq = \sum_{j \neq i} e^{s(q_i, q_j)/\tau}$, and $Cr = \sum_{j \neq i} e^{s(r_i, r_j)/\tau}$. They refer to the similarity of questions and repositories themselves, respectively.

We can observe that the difference between L_{Cq}, L_{Cr}, and L_D is the normalization part. Instead of simply summing them up, we can uniformly conclude these different types of loss functions into one negative log likelihood as the following Discrimination-Contrastive loss:

$$\mathcal{L}_{DC} = -\log \frac{e^{s(q_i, r_i)/\tau}}{e^{s(q_i, r_i)/\tau} + D + \alpha \cdot Cq + \beta \cdot Cr}, \tag{3}$$

where the weight hyper-parameter α and β control the importance of distinguishing similar questions and repositories. The combined Discrimination-Contrastive loss consists of four interactions:

(1) $q^+ \rightarrow\leftarrow r^+$: The main element that gathers paired question and repository together.

(2) $q^+ \longleftrightarrow q^-$: The factor that separates the embedding of questions.

(3) $r^+ \longleftrightarrow r^-$: The component that makes repositories to be distant from each other.

(4) $q^+ \longleftrightarrow r^-$: An important role that cause unmatched question-repository pairs segregated.

Hard Negative for Classification Loss. Previous studies (Luan et al., 2021; Karpukhin et al., 2020; Xiong et al., 2020) provide effective methods of hard negative sampling, and further demonstrate the importance of hard negatives. Ideally, we aim to pay more attention to these hard negatives, which are not semantically similar but are mapped close to the anchor in the vector space.

We employ BM25 as the base sentence matching algorithm. For each question in the training set, we query it in the repository corpus, and then select the top 10 results as the hard negatives except the correct answer. Then we construct triplets (q^+, r^+, r^-) as training instances, where the negative r^- is randomly sampled from these hard negatives at a ratio of p to avoid the model collapse. We minimize the cross entropy loss of enhanced representations of q and r as follows:

$$\mathcal{L}_{CE} = CrossEntropy\left(f(q, r, |q - r|), y\right), \tag{4}$$

where f denotes a feed-forward classification network, and y represents the classification label.

Overall Loss. The overall loss is the weighted sum of the DC loss and the CE loss, which is:

$$\mathcal{L} = \mathcal{L}_{DC} + \gamma \mathcal{L}_{CE}, \tag{5}$$

where γ is a hyper-parameter to balance the loss.

4.4 Cross-Batch Memory Augmentation

Cross-batch memory (XBM) (Wang et al., 2020) can considerably boost the performance of contrastive learning tasks. The XBM module stores embeddings and labels for data points. It is maintained as a first-input-first-output (FIFO) queue. Enqueue and dequeue procedures are conducted when a mini-batch arrives. As mentioned above, for an anchor question q_i, we pair it with rest $M-1$ repositories $\{r_j | j \neq i\}$ in the M-size mini-batch as negative pairs. In practice, those heavyweight BERT-based model has acute GPU memory cost issues. The size of mini-batch is often limited in NLP tasks using BERT-based models. By pairing anchors with samples stored in XBM, the information provided by negative pairs is significantly enriched.

5 Experiments

5.1 Experimental Setup

Dataset. We conduct experiments on the Repo4QA-small dataset, by randomly splitting Repo4QA-small dataset into 2,966/400/400 for training/testing/validation. For repositories retrieval, we evaluate the performance from 3 different corpora: the test split, the whole Repo4QA-small repositories, and the whole Repo4QA-large repositories, which is a more realistic setting since the repositories do not occur during the training period. We always only raise all 400 questions in the testing set. However, we retrieve answers from our dataset with different scales. As for the test split, we search for the best answer in the 400 repositories. For the Repo4QA-small split, the range of ranking is the 400 repositories in the testset and the rest 3,366 repositories in the Repo4QA-small set. Note that though some repositories have occurred in the training step, we never learn the relationship between questions and these repositories. For the Repo4QA-large split, we seek suitable repositories among the union of the 400 repositories in the testset and all the repositories in the Repo4QA-large set. If given more repositories, such as whole GitHub repositories, a practical solution is to filter

Models	Test			Small			Large		
	MRR	P@1	P@5	MRR	P@1	P@5	MRR	P@1	P@5
BERT-base	7.98	4.25	9.50	3.08	1.50	3.75	1.23	0.50	1.75
CodeBERT-base	2.22	0.25	2.25	0.34	0	0.25	0.05	0	0
Glove	19.82	13.00	26.00	10.33	7.00	14.50	5.74	3.75	6.50
BM25	43.22	35.25	51.75	28.23	22.00	35.50	22.13	17.50	27.00
Universal Sentence Encoder	62.72	49.75	78.25	41.24	31.25	51.75	27.09	20.00	33.00
S-BERT	80.23	72.00	91.00	59.22	47.00	73.50	43.89	34.50	55.75
Siamese-CodeBERT	79.37	70.75	88.50	56.62	44.50	68.25	41.67	32.25	53.00
Triplet-CodeBERT	82.96	76.00	89.75	61.86	50.25	76.50	46.24	36.75	57.25
QuRep-BERT-base	77.23	70.25	87.00	59.61	48.50	73.00	44.93	32.75	60.75
QuRep (ours)	**86.26**	**81.00**	**93.00**	**69.04**	**58.50**	**82.25**	**54.26**	**41.75**	**70.00**

Table 2: Experimental results on the Repo4QA-small test set. The best figure of MRR and P@1 metric is in **bold**. Our QuRep model outperforms both unsupervised and supervised baselines.

several repositories with traditional IR approaches such as BM-25, then re-rank these repositories via our model.

Metrics. We adopt two common metrics widely used in answer ranking to measure the effectiveness of our proposed model, namely, Mean Reciprocal Rank (MRR) and Precision@K, which means the rate that correct answer appears in the top-K ranked candidates. In practice, we evaluate the performance of $K = 1$ and $K = 5$.

Baselines. As Repo4QA is a new challenge, no model is specifically designed for it. Existing methods such as ColBert (Khattab and Zaharia, 2020) focus on passage ranking tasks such as MS MARCO (Nguyen et al., 2016), with short queries and related passages. While query expansion methods (Nogueira et al., 2019b,a) are not so helpful because of the complexity of our questions. Pairwise cross-encoders are more suitable for the re-rank task after we retrieve the dense representation for the consideration of the effective-efficient trade-off. For a fair comparison, diversified methods for similar tasks, such as sentence embedding and metric learning, are introduced as baselines. We use the exact same processed data for our model as the input of these baselines. To be specific, BM25(okapi) (Robertson et al., 1995) and GloVe (Pennington et al., 2014) is widely used in IR tasks, while Universal Sentence Encoder (Cer et al., 2018), BERT (Kenton and Toutanova, 2019), CodeBERT (Feng et al., 2020), S-BERT (Reimers and Gurevych, 2019) are transformer-based models with different pre-training and fine-tuning strategies. To compare with our designed loss functions, we train the CodeBERT model with vanilla siamese loss and triplet loss. Appendix B provides implementation details of baselines.

Implementation Details. We implement our model based on the Hugging Face transformers library. The initial weight is adopt from `microsoft/codebert-base`[4]. We use RAdam (Liu et al., 2019a) as the optimizer, with a cosine annealing learning rate from 1e-5 to 1e-7. The temperature τ in the loss function is 0.01. Factors of weights are set as $\alpha = \beta = 1.5$ and $\gamma = 0.4$. The negative sampling ratio is 0.2. For cross-batch memory, we set the size of the memory bank to 64, and this mechanism starts working from epoch 6.

5.2 Model Comparisons

The performance of different approaches on the Repo4QA task is in Table 2. From these experimental results, we have the following observations:

First, our proposed model outperforms all competitive baselines on MRR, P@1, and P@5 metrics. Especially, for the retrieval task on Repo4QA-large dataset, traditional IR-based and word embedding approaches cannot differentiate target from similar repositories, while contrastive-learning methods strongly outperform others. This result demonstrates the difficulty of understanding long-form questions compared with short queries, and the necessity of precise retrieval on large corpora via semantic search instead of lexical matching.

Second, poor performance is shown by BERT

[4] https://github.com/microsoft/CodeBERT

Loss	MRR	P@1	P@5
Ours	**86.26**	**81.00**	**93.00**
Vanilla triplet loss	82.96	76.00	89.75
Hinge loss	83.72	77.25	91.75
Circle loss	82.44	75.00	91.25
InfoNCE	84.44	78.25	92.25

Table 3: Results with different comparable loss function.

Models	MRR	P@1	P@5
QuRep	**86.26**	**81.00**	**93.00**
w/o XBM	-3.28	-4.00	-2.50
w/o CodeBERT	-9.03	-10.75	-6.00
w/o Constrative Loss	-21.24	-26.75	-13.25
w/o CE Loss	-0.83	-1.50	-1.00
w/o Hard Negatives	-1.43	-2.75	-1.25

Table 4: Results of ablation study conducted on Test split about model structure.

and CodeBERT if directly employing mean pooling output to similarity computation. The results are even worse than using average GloVe embeddings. Universal Sentence Encoder shows effectiveness among unsupervised learning methods, and SBERT achieves great performance since they are trained on large QA corpora.

Third, nearly all supervised methods achieve higher performance than unsupervised models, though no early interaction, such as cross-attention between questions and repositories, is considered. This phenomenon demonstrates the sound effect of contrastive learning.

Fourth, although some repositories in the Repo4QA-small dataset have been modeled in the training period, the performance still drops a lot compared with the result in the Repo4QA-test. The root cause is the diversity of questions. As the questions are always from the testset, they keep unseen for the model in the inference step. There is also no issue of data leakage in our Repo4QA-small dataset because the training data consists of QA pairs, but the questions in testset never pair with answers in the Repo4QA-small dataset. When questions have a longer form instead of simple keywords query, the semantic information contained is richer and more diversified, which increases the difficulty of encoding.

5.3 Loss Comparisons

We agree that an empirical study for comparable loss functions contributes to the soundness of our method. As reported in Table 3, experimental results demonstrate the hybrid loss we design significantly improves the performance compared with the classical loss functions of contrastive study(Sun et al., 2020; Gao et al., 2021b).

The strongest baseline, InfoNCE, has a similar form to our Discrimination-Contrastive loss. The difference is that we distinguish the effect from questions and repositories as stated in Sect. 4.3.

5.4 Ablations Study

As the experimental results show the same trend in all three splits, in this part, we only conduct experiments in the test split. We consider the model components and loss function components as two main aspects of our ablation studies.

Effects of model components. Removing or replacing components of QuRep decreases the performance, as Table 4 shows. To be more specific, we replace CodeBERT with BERT-base, just as the result of QuRep-BERT-base shows. The performance drop proves the effectiveness of pre-training in the programming domain. Cross-batch memory is proved to be a necessary mechanism for constructing more QA pairs to be trained.

Effects of loss components. We also investigate the ablation of constituents in our proposed hybrid loss function. Each part of our loss contributes to the improvement of performance. As illustrated in Table 4, contrastive loss is the backbone of our learning target, and the hard negative sampling strategy also plays an important role.

5.5 Case Study

To better understand the difficulty of the task and the effectiveness of our solution, we present a case study in this part. As a typical method in the IR field, BM25 is selected as the baseline method to compare. All these cases can be viewed on StackOverflow. We only display key parts of questions/repositories considering the full length.

The first and second cases in Table 5 are examples that QuRep outperforms BM25. Question 1 means to find a dictionary implementation in JavaScript. BM25 returns results containing the keyword "dictionary," as the repository "stig/json-framework" (now redirected to SBJson/SBJson) mentions NSMutableDictionary type in Objective-

Question Title	BM25 Rank	QuRep Rank	BM25 Top1	QuRep Top1
Is there a dictionary implementation in JavaScript	13	1	stig/json-framework This framework implements a strict parser and generator in Objective-C	jieter/django-tables2 A complete, fully tested and documented data structure library written in pure JavaScript.
What is the simplest way to graph rrd files in Grafana	30	1	bookkojot/mp4fixer Recover damaged/unfinished mp4 files with h264 video	doublemarket/grafana-rrd-server A simple HTTP server that reads RRD files and responds to requests from Grafana.
Validate URL with standard package in GO	32	1	GitHub/fetch Promise-based mechanism for programmatically making web requests in the browser	asaskevich/govalidator Go Package of validators and sanitizers for strings, numerics, slices and structs.
Angular: How to force run unit tests when running Git push	245	6	mozilla-mobile/fenix all-new Firefox for Android browser, based on GeckoView and Mozilla Android Components.	lerna/lerna A tool for managing JavaScript projects with multiple packages

Table 5: Case study on BM25 and QuRep.

C, which is not so close to the question. In comparison, our proposed model recommends the collection of data structures written in JavaScript. Though the word "dictionary" is not discussed in the repository so frequently, "dictionary" is contained by the concept "data structures." This is what language models bring us. Case 2 and 3 also show that BM25 tend to return similar repositories at the word level, which results in repositories from different disciplines or programming languages. While QuRep can identify the purpose of questions and recommend better answers.

Case 4 is a bad case in which both models fail. The best answer posted on StackOverflow is "typicode/husk." QuRep ranks "lerna/lerna" as the best because it discusses both Angular and Git. However, the phrase "unit test" is ignored.

From the case study section, we can learn that understanding the intent of a question is a key factor for our task, especially when the question contains rich semantic information. The relations of concepts are also an important issue to study.

6 Conclusion

In this paper, we introduce an automatically collected novel dataset, Repo4QA, for the proposed task. Furthermore, we propose QuRep, a contrastive learning method to fine-tune CodeBERT for our task. Experimental results demonstrate that our method outperforms baseline models in effectiveness and efficiency. Detailed analyses are conducted to investigate the impact on performance brought by components of our model. We look forward to other applications and more research interest in our task. Moving forward, we plan to deploy our dataset and method to solve programming questions in software engineering practice and consider code stored in repositories for better modeling of multi-modal GitHub repositories.

Acknowledgements

This work is supported by the National Science Foundation of China Grant No. 62161146001.

References

Sushil Bajracharya, Trung Ngo, Erik Linstead, Yimeng Dou, Paul Rigor, Pierre Baldi, and Cristina Lopes. 2006. Sourcerer: a search engine for open source code supporting structure-based search. In *Companion to the 21st ACM SIGPLAN symposium on Object-oriented programming systems, languages, and applications*, pages 681–682.

J. Cambronero, Hongyu Li, Seohyun Kim, Koushik Sen, and S. Chandra. 2019. When deep learning met code search. *Proceedings of the 2019 27th ACM Joint Meeting on European Software Engineering Conference and Symposium on the Foundations of Software Engineering*.

Daniel Cer, Yinfei Yang, Sheng-yi Kong, Nan Hua, Nicole Limtiaco, Rhomni St John, Noah Constant, Mario Guajardo-Cespedes, Steve Yuan, Chris Tar, et al. 2018. Universal sentence encoder for english. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 169–174.

Daniel Cohen, Liu Yang, and W. Croft. 2018. Wikipassageqa: A benchmark collection for research on non-factoid answer passage retrieval. *The 41st International ACM SIGIR Conference on Research & Development in Information Retrieval*.

Zhangyin Feng, Daya Guo, Duyu Tang, Nan Duan, Xiaocheng Feng, Ming Gong, Linjun Shou, Bing Qin, Ting Liu, Daxin Jiang, et al. 2020. Codebert: A

pre-trained model for programming and natural languages. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: Findings*, pages 1536–1547.

Luyu Gao, Zhuyun Dai, and Jamie Callan. 2021a. Coil: Revisit exact lexical match in information retrieval with contextualized inverted list. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 3030–3042.

Tianyu Gao, Xingcheng Yao, and Danqi Chen. 2021b. Simcse: Simple contrastive learning of sentence embeddings. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 6894–6910.

Xiaodong Gu, Hongyu Zhang, and Sunghun Kim. 2018. Deep code search. *2018 IEEE/ACM 40th International Conference on Software Engineering (ICSE)*, pages 933–944.

Masum Hasan, Tanveer Muttaqueen, Abdullah Al Ishtiaq, Kazi Sajeed Mehrab, Md. Mahim Anjum Haque, Tahmid Hasan, Wasi Ahmad, Anindya Iqbal, and Rifat Shahriyar. 2021. CoDesc: A large code–description parallel dataset. In *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*, pages 210–218, Online. Association for Computational Linguistics.

Geert Heyman and Tom Van Cutsem. 2020. Neural code search revisited: Enhancing code snippet retrieval through natural language intent. *arXiv preprint arXiv:2008.12193*.

J. Huang, D. Tang, L. Shou, M. Gong, and N. Duan. 2021. Cosqa: 20,000+ web queries for code search and question answering. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*.

Po-Sen Huang, Xiaodong He, Jianfeng Gao, Li Deng, Alex Acero, and Larry Heck. 2013. Learning deep structured semantic models for web search using clickthrough data. In *Proceedings of the 22nd ACM international conference on Information & Knowledge Management*, pages 2333–2338.

H. Husain, Hongqi Wu, Tiferet Gazit, Miltiadis Allamanis, and Marc Brockschmidt. 2019. Codesearchnet challenge: Evaluating the state of semantic code search. *ArXiv*, abs/1909.09436.

Vladimir Karpukhin, Barlas Oguz, Sewon Min, Patrick Lewis, Ledell Wu, Sergey Edunov, Danqi Chen, and Wen-tau Yih. 2020. Dense passage retrieval for open-domain question answering. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 6769–6781.

Jacob Devlin Ming-Wei Chang Kenton and Lee Kristina Toutanova. 2019. Bert: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of NAACL-HLT*, pages 4171–4186.

Omar Khattab and Matei Zaharia. 2020. Colbert: Efficient and effective passage search via contextualized late interaction over bert. In *Proceedings of the 43rd International ACM SIGIR conference on research and development in Information Retrieval*, pages 39–48.

Taeuk Kim, Kang Min Yoo, and Sang-goo Lee. 2021. Self-guided contrastive learning for BERT sentence representations. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 2528–2540, Online. Association for Computational Linguistics.

Kyungjae Lee, Nan Duan, Lei Ji, Jason Li, and Seungwon Hwang. 2020. Segment-then-rank: non-factoid question answering on instructional videos. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pages 8147–8154.

Liyuan Liu, Haoming Jiang, Pengcheng He, Weizhu Chen, Xiaodong Liu, Jianfeng Gao, and Jiawei Han. 2019a. On the variance of the adaptive learning rate and beyond. In *International Conference on Learning Representations*.

Xueqing Liu, Chi Wang, Yue Leng, and ChengXiang Zhai. 2018. Linkso: a dataset for learning to retrieve similar question answer pairs on software development forums. In *Proceedings of the 4th ACM SIGSOFT International Workshop on NLP for Software Engineering*, pages 2–5.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019b. Roberta: A robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692*.

Shuai Lu, Daya Guo, Shuo Ren, Junjie Huang, Alexey Svyatkovskiy, Ambrosio Blanco, Colin Clement, Dawn Drain, Daxin Jiang, Duyu Tang, Ge Li, Lidong Zhou, Linjun Shou, Long Zhou, Michele Tufano, Ming Gong, Ming Zhou, Nan Duan, Neel Sundaresan, Shao Kun Deng, Shengyu Fu, and Shujie Liu. 2021. Codexglue: A machine learning benchmark dataset for code understanding and generation. *ArXiv*, abs/2102.04664.

Yi Luan, Jacob Eisenstein, Kristina Toutanova, and Michael Collins. 2021. Sparse, dense, and attentional representations for text retrieval. *Transactions of the Association for Computational Linguistics*, 9:329–345.

Shanshan Lyu, Wentao Ouyang, Yongqing Wang, Huawei Shen, and Xueqi Cheng. 2019. What we vote for? answer selection from user expertise view

in community question answering. In *The World Wide Web Conference*, pages 1198–1209.

Tri Nguyen, Mir Rosenberg, Xia Song, Jianfeng Gao, Saurabh Tiwary, Rangan Majumder, and Li Deng. 2016. Ms marco: A human generated machine reading comprehension dataset. In *CoCo@ NIPS*.

Liming Nie, He Jiang, Zhilei Ren, Zeyi Sun, and Xiaochen Li. 2016. Query expansion based on crowd knowledge for code search. *IEEE Transactions on Services Computing*, 9:771–783.

Ping Nie, Yuyu Zhang, Xiubo Geng, Arun Ramamurthy, Le Song, and Daxin Jiang. 2020. Dc-bert: Decoupling question and document for efficient contextual encoding. In *Proceedings of the 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 1829–1832.

Rodrigo Nogueira, Jimmy Lin, and AI Epistemic. 2019a. From doc2query to doctttttquery. *Online preprint*.

Rodrigo Nogueira, Wei Yang, Jimmy Lin, and Kyunghyun Cho. 2019b. Document expansion by query prediction. *arXiv preprint arXiv:1904.08375*.

Jeffrey Pennington, Richard Socher, and Christopher D Manning. 2014. Glove: Global vectors for word representation. In *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*, pages 1532–1543.

Xipeng Qiu and Xuanjing Huang. 2015a. Convolutional neural tensor network architecture for community-based question answering. In *Twenty-Fourth international joint conference on artificial intelligence*.

Xipeng Qiu and Xuanjing Huang. 2015b. Convolutional neural tensor network architecture for community-based question answering. In *IJCAI*.

Chen Qu, Liu Yang, Cen Chen, Minghui Qiu, W Bruce Croft, and Mohit Iyyer. 2020. Open-retrieval conversational question answering. In *Proceedings of the 43rd International ACM SIGIR conference on research and development in Information Retrieval*, pages 539–548.

Nils Reimers and Iryna Gurevych. 2019. Sentence-bert: Sentence embeddings using siamese bert-networks. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 3982–3992.

Ruiyang Ren, Shangwen Lv, Yingqi Qu, Jing Liu, Wayne Xin Zhao, Qiaoqiao She, Hua Wu, Haifeng Wang, and Ji-Rong Wen. 2021. Pair: Leveraging passage-centric similarity relation for improving dense passage retrieval. In *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*, pages 2173–2183.

Stephen E Robertson, Steve Walker, Susan Jones, Micheline M Hancock-Beaulieu, Mike Gatford, et al. 1995. Okapi at trec-3. *Nist Special Publication Sp*, 109:109.

Huajie Shao, Dachun Sun, Jiahao Wu, Zecheng Zhang, A. Zhang, Shuochao Yao, Shengzhong Liu, Tianshi Wang, C. Zhang, and T. Abdelzaher. 2020. paper2repo: Github repository recommendation for academic papers. *Proceedings of The Web Conference 2020*.

Yifan Sun, Changmao Cheng, Yuhan Zhang, Chi Zhang, Liang Zheng, Zhongdao Wang, and Yichen Wei. 2020. Circle loss: A unified perspective of pair similarity optimization. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 6398–6407.

Yao Wan, Jingdong Shu, Yulei Sui, Guandong Xu, Zhou Zhao, Jian Wu, and Philip S. Yu. 2019. Multi-modal attention network learning for semantic source code retrieval. *2019 34th IEEE/ACM International Conference on Automated Software Engineering (ASE)*, pages 13–25.

Xun Wang, Haozhi Zhang, Weilin Huang, and Matthew R Scott. 2020. Cross-batch memory for embedding learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 6388–6397.

Lee Xiong, Chenyan Xiong, Ye Li, Kwok-Fung Tang, Jialin Liu, Paul N Bennett, Junaid Ahmed, and Arnold Overwijk. 2020. Approximate nearest neighbor negative contrastive learning for dense text retrieval. In *International Conference on Learning Representations*.

Ziyu Yao, Daniel S. Weld, Wei-Peng Chen, and Huan Sun. 2018. Staqc: A systematically mined question-code dataset from stack overflow. *Proceedings of the 2018 World Wide Web Conference*.

Pengcheng Yin, Bowen Deng, Edgar Chen, Bogdan Vasilescu, and Graham Neubig. 2018. Learning to mine aligned code and natural language pairs from stack overflow. *2018 IEEE/ACM 15th International Conference on Mining Software Repositories (MSR)*, pages 476–486.

Dejiao Zhang, Shang-Wen Li, Wei Xiao, Henghui Zhu, Ramesh Nallapati, Andrew O Arnold, and Bing Xiang. 2021a. Pairwise supervised contrastive learning of sentence representations. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 5786–5798.

Wei Zhang, Zeyuan Chen, Chao Dong, Wen Wang, Hongyuan Zha, and Jianyong Wang. 2021b. Graph-based tri-attention network for answer ranking in cqa. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 35, pages 14463–14471.

Zhou Zhao, Hanqing Lu, Vincent W Zheng, Deng Cai, Xiaofei He, and Yueting Zhuang. 2017. Community-based question answering via asymmetric multi-faceted ranking network learning. In *Thirty-First AAAI Conference on Artificial Intelligence*.

A Dataset comparison

As shown in Table 6, we compare our dataset with several datasets on Code Intelligence and Community-based QA. The scale of our dataset is similar to SO-DS (Heyman and Van Cutsem, 2020), CQA-Quora (Lyu et al., 2019) and CQA-SO (Zhang et al., 2021b). Our QA pairs are more complex then short-form query/code pairs, which resulting in less available resources on the web. Moreover, our dataset is the only cross-platform one.

B Baseline Implementation Details

We select GloVe average embedding, BERT *[CLS]* token output and CodeBERT *[CLS]* token output as baselines without supervision. For the supervised learning method, Siamese-formed CodeBERT and Triplet-formed CodeBERT are compared.

- BM25(okapi) (Robertson et al., 1995) BM25 is a well-known lexical retrieval model. We employ the implementation from the *rank_bm25* library.[5]
- GloVe (Pennington et al., 2014) The mean embedding of the whole sentence is regarded as the representation of the sentence. Sentence representation similarity is computed for ranking.
- Universal Sentence Encoder (Cer et al., 2018) It is a transformer-based network which augments unsupervised learning with training on SNLI dataset. [6]
- BERT (Kenton and Toutanova, 2019) We use the *[CLS]* token output for sentence embedding.
- CodeBERT Similar to the strategy applied on BERT, the *[CLS]* token output is adopted. Besides, we train a Siamese-formed CodeBERT and a Triplet-formed CodeBERT for comparison in supervised learning. The implementation of CosQA (Huang et al., 2021) is based on the Siamese-formed CodeBERT.
- S-BERT (Reimers and Gurevych, 2019) is a Siamese BERT-Networks. We employ the *all-roberta-large-v1* model hosted on huggingface, which is pretrained over 1B+ QA pairs for better sentence embedding. [7]

C Hyper-parameter Configuration

To figure out the different role that our proposed loss functions played in the optimization phrase, we conduct a discussion on weight hyper-parameters, α, β, and γ. α and β controls the importance of supervised contrastive loss over discrimination loss. Since Cq, Cr and D occurs together at the normalization part of equation 4.3, the larger their weight are, the more importance the have. We set $\alpha = \beta$ during our experiment, and report result as Table 7. We can find that the model performs better while $\alpha > 1$, which means the model focus more on separating elements from the same semantic catogory apart. The output demonstrates the importance and effectiveness of surpervised contrastive learning. Besides, the weight γ and the hard negative sampling rate p are set to 0.2 to achieve best performance.

α	MRR	P@1	P@5
2	86.01	80.50	**93.50**
1.5	**86.26**	**81.00**	93.00
1	85.30	79.00	92.75
0.5	85.12	78.00	92.75
0	84.48	77.75	92.50

Table 7: Results of different hyperparameter $\alpha = \beta$ settings.

[5]https://GitHub.com/dorianbrown/rank_bm25
[6]https://tfhub.dev/google/universal-sentence-encoder/4
[7]https://huggingface.co/sentence-transformers/all-roberta-large-v1

Dataset	Domain	Size	Query Type	Answer Type	Annotation
CSN (Husain et al., 2019)	Coding	2.3M	Short description	Function code	No
Deep Code Search (Gu et al., 2018)	Coding	18.2M	Short description	Function code	No
CoSQA (Huang et al., 2021)	Coding	20.6K	Web Query	Function code	Yes
QECK (Nie et al., 2016)	Coding	312.9K	SO question title	Code block	No
StaQC (Yao et al., 2018)	Coding	268K	SO question title	Code block	Partly
CoNaLa (Yin et al., 2018)	Coding	598.2K	SO question title	Code block	Partly
SO-DS (Heyman and Van Cutsem, 2020)	Coding	12.1k	SO question title	Code block	No
CQA-Quora (Lyu et al., 2019)	Open	76.2k	Quora question	Quora Answer	No
CQA-SO (Zhang et al., 2021b)	Coding	13.9k	SO question	SO Answer	No
Repo4QA (ours)	Coding	13.0k	SO question	GitHub repository	No

Table 6: Overview of existing datasets on Code Intelligence and Community-based QA

Addressing Limitations of Encoder-Decoder Based Approach to Text-to-SQL

Octavian Popescu
IBM Research
o.popescu@us.ibm.com

Irene Manotas
IBM Research
irene.Manotas@ibm.com

Ngoc Phuoc An Vo
IBM Research
ngoc.phuoc.an.vo@ibm.com

Hangu Yeo
IBM Research
hangu@us.ibm.com

Elahe Khorasani
IBM Research
elkh@ibm.com

Vadim Sheinin
IBM Research
vadims@us.ibm.com

Abstract

Most attempts on Text-to-SQL task using encoder-decoder approach show a big problem of dramatic decline in performance for new databases. Models trained on Spider dataset, despite achieving 75% accuracy on Spider development or test sets, show a huge decline below 20% accuracy for databases not in Spider. We present a system that combines automated training-data augmentation and ensemble technique. We achieve double-digit percentage improvement for databases that are not part of the Spider corpus.

1 Introduction

Text-to-SQL is a task to automatically translate a given a natural language question into an SQL formula for a specific database (DB).

In 2018, Spider dataset (Tao et al., 2018) was released as a large-scale complex and cross-domain semantic parsing and Text-to-SQL dataset. Since then numerous models have been developed and evaluated using Spider dataset. Although there is a large increase in accuracy, growing from 20% to 70 + % between 2019 and 2021 of top ranked systems, later studies (Suhr et al., 2020a; Shi et al., 2020; Zhong et al., 2020), show that the accuracy of these models drops significantly, in some cases well below 20%, for databases outside Spider dataset.

In this paper, we present various experiments for Text-to-SQL task using DB schemas that are inside and outside of Spider dataset. We identify systematic types of errors and analyze their possible causes. From our error analysis, we propose solutions to address the limitations of the underlying encoder-decoder architecture and of the Spider dataset. We use training data augmentation technique in order to create a series of models for the same DB schema. By coupling each model SQL result with the information extracted from the query at the NLP step, we interpolate a final SQL formula via Ensemble technique. The system achieves very good results compared to its individual components and can be run on any new DB schema, in or outside of the Spider dataset, even when there exists a set of very limited number of training examples, on the order of several tens.

This paper is organized as follows. In Section 2, we review studies that defined a "standard" encoder-decoder architecture for Text-to-SQL. In Section 3, we define the problem of encoder-decoder approach based on declined performances on new DB schemas and new types of questions, that leads to the challenge of achieving good accuracy. Next, we describe our system in Section 5. The experiment results in Section 6 and conclusions in Section 7.

2 Related Work

The Text-to-SQL task is not a new task in NLP and there are many references on its complexity and achievements obtained recently (Navid et al., 2017; Popescu et al.; Yao et al., 2010). In (Tao et al., 2018), the Spider corpus with thousands of training examples was introduced in 2018. The Spider challenge has been introduced as well, where any team can have their system evaluated on an unknown corpus, which is a part of the Spider corpus, but not publicly available.

In (Cai et al., 2018; Gehring et al., 2017; Yin et al., 2016; Rabinovich et al., 2017), among others,

Proceedings of the 29th International Conference on Computational Linguistics, pages 1593–1603
October 12–17, 2022.

a sequence-to-sequence approach was introduced, opening the way for this type of architecture. Instead of translating into SQL, (Guo et al., 2019; Zhang et al., 2019; Bogin et al., 2019), the system translates into a representation that captures the semantics of the query, called Intermediate Representation (IR). From IR to SQL is a deterministic process: a context free grammar is used to convert one into another. The authors build on the work of (Sun et al., 2019; Cheng et al., 2019) that used Abstract Syntax Tree (AST).

A system that uses the the IR approach, but extends the context free grammar to include values, is Valuenet(VL) (Brunner and Stockinger, 2021). The code is available from (Brunner, 2021) . Valuenet obtained an accuracy of 60% on Spider development corpus, which put it among the best systems in the Spider challenge in 2020.

The RAT system (Wang et al., 2019) is based on the relation-aware self-attention mechanism, to address schema encoding, schema linking, and feature representation within a Text-to-SQL encoder. They augmented their system with BERT (Devlin et al., 2018), showing that using transformers brings a significant increase in the accuracy. But, most importantly, they also showed that "... all the known information about the schema, it is insufficient for appropriately encoding a previously unseen schema in the context of the question...". In (Suhr et al., 2020b), a detailed analysis of systems that performed well on Spider corpus confirmed a large drop in accuracy for DB schemas outside of the Spider corpus.

Given that creating a new training is an expensive process, an important part of research for Text-to-SQL task has been dedicated to improving the training methods, in order to reduce this cost. Paraphrasing (Ronak Kaoshik, 2021), dialog (Artzi and Zettlemoyer, 2011; Gur et al., 2018), or a combination of these (Herzig and Berant, 2019) were used to cope with this problem.

(Kalajdjieski et al., 2020) surveyed a series of methods to automatically create training corpora. Some of these models are pretrained using the Masked Language Modeling (MLM) task by either masking tokens from the utterance input or tokens from the schema input (Deng et al., 2021). This work demonstrated that jointly pre-training on utterances and table contents (e.g., column names and cell values) can benefit downstream tasks such as table parsing and semantic parsing. In (Xu et al., 2017) a general method, which avoids the dependency of order in deep learning system, is presented.

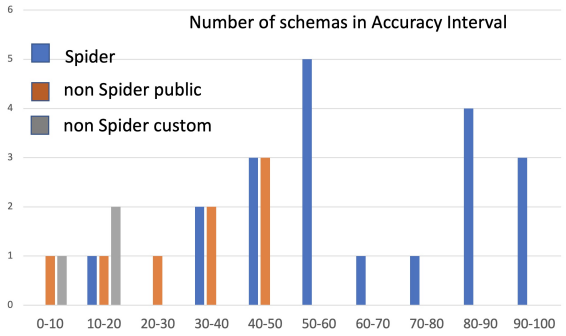

Figure 1: Accuracy for Spider vs. non Spider DB schemas

Our own observation aligns very well with the observations in the previous papers, and confirms the difficulty to cope with new unseen DB schema only by means of unique training corpus and/or model. However, our solution differs from what has been been proposed so far. The systems that used general pretraining methods, like RAT+GAP, did not show that the same performance is observed for out of Spider schemas, but only for the ones in Spider corpus. The systems that confronted the systematic errors that are produced in the case of new DB schemas, took a different path from ours, like (Suhr et al., 2020b), and did not exploit systematically the relationship between values and filters as a means to control the correctness of SQL formula. Their line of research assumed that this state of fact is due only to the lack in training. However, our experiments support the idea, that, while indeed some errors are traceable to the particularities of a certain training corpus, one important cause of many errors is ignoring the way the DB schema relationships could be expressed in natural language, see Section 3. By explicitly providing the linkage between values and schema definition, coupled with specific training generation, and interpolating over an ensemble of different models, our system shows improvement for DB schemas that are inside or outside of the same corpus.

3 Problem Definition

While the overall results for top performers on the Spider corpus are high, the variation for individual DB in the testing corpus can be large. However, when the same system is applied to the DB schemas outside of the Spider corpus we obtain a different distribution, Figure 1. The *Spider* his-

togram shows that accuracy for majority of test DB schemas fell into the range of 40-90%. The data was obtained running the default VL, system trained on the Spider training corpus and tested on the Spider development corpus. The *non Spider general* come from (Suhr et al., 2020a). None of these DB schemas have an accuracy above 50%. The *non Spider custom* refers to the training and test corpora for our own databases, described in section 4. For these databases, the accuracy is in the same category as for the lowest for *non Spider general*, namely below 20%. This category contains just one DB schema for the Spider corpus. Looking at what percentage of schemas are in three main categories, which roughly correspond to *bad, ok, good*, the difference between Spider vs. non Spider DB schemas is very large. This shows that indeed there is a very strong skew in the distribution towards the left intervals of accuracy for new DB schemas.

The reason for this skew for new DB schemas that is exemplified above was pointed out in (Suhr et al., 2020b) and (Zhong et al., 2020). Without proper reference to the linkage between query tokens to specific table/columns, based only on the examples seen in training, the deep learning model is not able to discern between similar queries in the context of a new DB schema. In Section 4 we provide details on these types of errors and their causes.

4 Baseline and Error Analysis

Spider dataset (Tao et al., 2018) is a large-scale cross-domain semantic parsing and Text-to-SQL annotated dataset. **ValueNet** (VL) (Brunner and Stockinger, 2021) is an open source deep learning system built using an encoder-decoder architecture. The Valuenet trained on Spider dataset represents our baseline.

Our outside Spider DB schemas are HR, WH and BI (*reference to articles kept anonymous for now*). For each of these schemas we have from 70 to 200 training examples.These training examples are split into two different corpora *ANS* - for training, and (2) *nANS* - for test.

VL obtained an accuracy of about 60% on the Spider development, but much lower results for the new DBs. In table Table 1 we present the results obtained by the baseline for the new DBs, separately for what we are going to us as training (ANS) and for test respectively(nANS).

Table 1: Accuracy for Default Valuenet+Spider corpus

DB	#ANS	acc	#nANS	acc
HR	121	24%	81	19%
WH	87	20%	42	15%
BI	109	4%	112	2%

4.1 Error Analysis

4.1.1 EA1. Absence of certain types of query/SQL in training

For questions having more than one value such as `Who bought most Apple products in Bestbuy?`, or `In which year did Mary shop in both Bestbuy and Radioshack?`, many times the VL model misses one of the values, or creates an incorrect filter relationships in the final SQL.

4.1.2 EA2. Limitations on Complexity

Queries having compound logical operators, such as `(filter1 OR filter2) AND filter3`, are not present in the corpus. When input questions need to generate queries like this, the Spider code cannot represent such type of queries correctly, it outputs a truncated representation, so the SQL is never correctly inferred.

4.1.3 EA3. Temporal questions

Currently, time is a string in Spider corpus, so the training examples contains `like` operator. However, time operations, like difference, conversion etc., are needed for queries like `... after 7 days...` or `... two days after April 1st ...`

4.1.4 EA4. Pre-processing information

By default, VL uses a Google API for NLP preprocessing of the query. As such, only general information about values, which is outside the scope and definition of the schema, is fed to the network. For example, for the token `Apple`, which in the context of that DB refers to the name of the manufacturer, the information provided is the web page describing this company, which does not help a lot in inferring the correct SQL.

4.1.5 EA5. Linking Values and Schema

Crucial relationship between values to the underlying schema is missing as input to the VL encoder-decoder. For instance, using the fact that the token `Apple` is related to the table `Manufacturers`

and the column `name` significantly increases the chances that the inferred SQL is correct.

4.1.6 EA6. Checking the consistency of SQL vs. syntax restrictions

English syntax provides hard restrictions on the constituency of phrases that are translated into SQL. A determiner, such as an adjective, has to be applied in the SQL formula to the value representing the head of the English phrase. For a query like `what is the age of pet of the youngest owner`, the correct SQL links `age` to a column of the table `pets` while `youngest` is an aggregation function on the table `students`, not vice versa. Many times the incorrect SQL is generated by the network because of inconsistency of SQL operators association.

5 Hybrid System (HS)

We created a new system that processes and extracts the relevant information from the input question and links tokens to specific tables and columns from the corresponding DB schema. We replaced Valuenet's pre-processing and Name Entity Recognition (NER) modules with our Disambiguation Dictionary Module (DDM) to provide crucial information of the relationship between values and schema. We also introduced an ensemble of neural network models to improve the performance of the inferred SQL queries. From the original Valuenet architecture we kept only the encoder-decoder for IR with the semQL grammar.

The hybrid system has a different data flow from Valuenet. First, an English query is processed by our Natural Language Processing module (NLP). In our approach, we correlate technologies that increase the performance by double figure percentage: (i) we devise a seed training data augmentation technique (STDA), that on the bases of the initial training corpus, we call seeds, is able to generate a larger training set. In a typical scenario, a few tens of seed examples lead to several hundreds of training examples; (ii) with different sets of automatically created training data, we train different models and use an ensemble technique to analyze the output of each of these models.

5.1 DDM query rewriting

The DDM component from NLP module, describes the columns, values, relationships between columns, and synonyms for columns and values, (Vadim et al., 2018; Popescu et al., 2019; Vo et al., 2019; Yeo et al., 2021). It creates a set of schema-dependent lexical rules using the information from schema annotations and a set of schema-independent template rules. From the set of matched rules a set of structured data items (DTI) is created. A DTI defines an item `[table name].[column name]` with operators such as `select`, `filter`, `aggregation`, etc., that need to be applied to the corresponding item. During the query processing process, a natural language question is disambiguated, annotated, and matched against the lexical rules. Using the information from DDM's output, and the input question in English, a query representation following the Spider format is created so it could be incorporated as an input to the neural network. The user's input query undergoes a double rewriting before the text goes to the network, as described below.

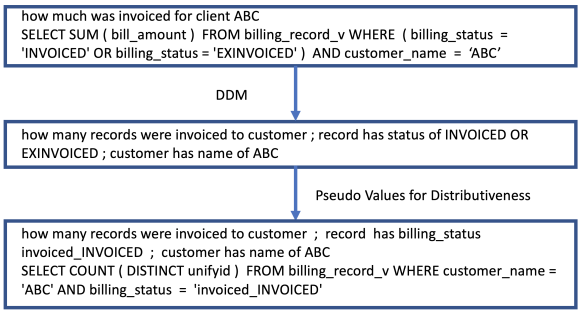

Figure 2: Rewriting of Queries in Hybrid System

- **DDM style question rewriting**. The input question is rewritten and augmented with the information provided by the NLP module, especially by the DDM module. The rewritten question contains explicit, English, references to tables and columns in the DB schema. For instance, for the question `how much was invoiced for client XYZ`, we rewrite the question as follow: the DDM's output for this query explicitly indicates that the token `invoiced` is a filter containing two values, `INVOICED_type1` and `INVOICED_type2`, from the table `billTable` and the column `status`. Also, a separate filter containing the value `XYZ` from the same table, but from a different column `name` is extracted. Two sentences are created accordingly: `bill has status of INVOICED_type1` OR `INVOICED_type2` and `customer has name of XYZ`. In the original query the values are replaced with their corre-

sponding columns, `how much amount was bill to customer` and the three sentences are adjoined in order to create the query that will be fed to the network: `how much amount was invoiced to customer; bill has status of INVOICED_type1 OR INVOICED_type2 ; customer has name of XYZ`.

The DDM rewriting of the user query adds the schema information into simple English sentences. In this way, the encoder-decoder have explicitly access the connection between the English tokens and their correspondent in the DB table and columns. See first two boxes in Figure 2.

- **pseudo-values for logically compound filters**. The SQL corresponding to the initial query in Figure 2 exemplifies a case where the priority of logical operators is taken into account. The Spider code does not properly handle these types of SQL constructs. We found two ways to automatically rewrite the SQL query such that this problem is handled properly. (1) to rewrite the SQL and the English question as a series of sub-questions such that each question in the query does not contains filters that need to be distributed into several logical conditions. Thus, we combine the individual results of the individual SQL conditions into a single result corresponding to the original SQL query; or (2) to use the pseudo-values to represent composite SQL conditions in the WHERE clause. Since some queries have values that are part of a composite filter, we can use a new pseudo-value representing the composite filter instead. For instance, for the query shown in Figure 2, we used the pseudo-value `invoiced_INVOICED` to represent the filters on both values present in the query for the same column. In the rewritten SQL instead of a compound filter we use this pseudo-value filter, which is linked through the `AND` operator to the other independent filter from the query, `customer_name='XYZ'`. See last box in Figure 2.

The DDM style question rewriting and the pseudo-values are applied both during training and inference time. During training, the neural network has access to the gold SQL query, and the SQL is rewritten using pseudo-values, if necessary. During inference, only the English question is rewritten, and the SQL is inferred. If the SQL is correct, this SQL will have the same form as the gold SQL query, and a rewritten procedure that replaces the pseudo-values back to the original value is applied to obtain the final SQL.

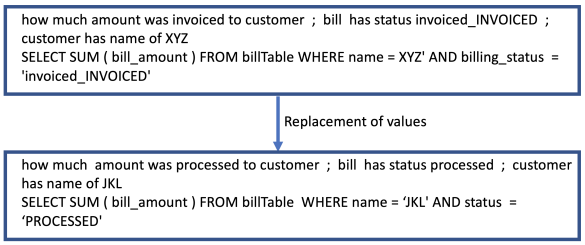

Figure 3: Replacing Values for Training Data Augmentation

The **HS**, by rewriting the queries before the IR is generated, is an effective solutions for the issues discussed in Error Analysis section 4.1.

5.2 Seed Training Data Augmentation (STDA)

The number of training examples for a given schema has a large impact on the accuracy of the trained deep learning model. However, for each DB schema only a small number of training examples is available, usually on the order of tens to a couple of hundreds at most. We call this small set of training examples the *seeds* set. To increase the number of training examples, we apply a data augmentation mechanism in which we take the seeds set, and automatically replace the corresponding values, in queries having multiple values, with alternative values from the same table and column as in the original example. We replace values in both the English question and the SQL query for training. In Figure 3 we show an example of how using a *seed* example, we can create a new training example.

Table 2: Seed Training Data Augmentation

Schema	Size of seeds	Size of STDA
HR	90	1500
WH	80	2000
BI	100	3300

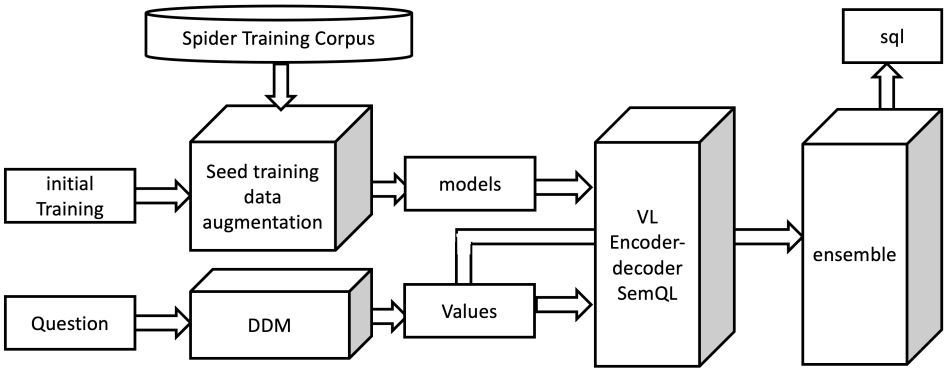

Figure 4: Hybrid System Architecture

5.3 Ensemble

We used the technique described in subsection 5.2 to create different training subsets from the original *seeds* training data for each DB schema. We created overlapping subsets, each one containing between 80 and 90 percent of the data available in the seeds set. We created a series of distinct models by training the deep learning model on the concatenation of the original Spider training set and each of the new generated training datasets for each DB schema.

A new query, that was not originally in the *seeds* set, is tested with all trained models. Each model infers an SQL that usually is not the same for all trained models. We chose the SQL query that obeys a set of hard coded restrictions created for each query. The restriction is that the filters in the SQL query observe the relationship between values and the DB schema as seen by the output of the rule-based system. These relationships consider values and numerical operators only, not the the body of the SQL formula. Therefore, we control only the compatibility between values in filters and the schema definition, and not the actual form of the SQL query e.g., join paths which inferred by the neural network. Figure 5 shows the inferred output from different models in the ensemble for the same query, the query at the bottom of the figure is the final selected SQL query.

Assembling the Output SQL. The ensemble component selects one SQL query that complies with the set of restrictions determined previously, this is done by assembling the output from a set of models created with the training datasets generated with STDA technique. The pseudo code shown in Algorithm 1 describes the steps that lead to the final SQL formula after interpolating the output from the models in the ensemble, as described above. In a nutshell, the algorithm considers the SQL query from each model, M_i, which is checked against the restrictions determined by DDM's output which are kept in $RSet$. The procedure consists of checking filters, $Filter\_M_i\_sql$, and operators, $Operators\_M_i\_sql$ one by one, independently of

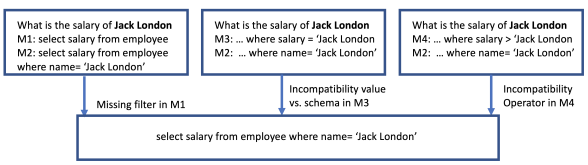

Figure 5: Interpolation over Multiple Models

Algorithm 1 Restrictions on Ensemble outputs

Require: Q ▷ English user query
Require: $M_1, M_2, ..., M_n$ ▷ DL models
Require: $RSet$ ▷ Restriction Set, DDM output
  while M_i **do**
    $Filter\_M_i\_sql$ ▷ extract filters from SQL
    $Operator\_M_i\_sql$ ▷ extract operators from SQL
    $EnsembleFailure \leftarrow True$ ▷ ensemble has no output
    if $Filter\_M_i\_sql$ not in $RSet$ **then**
      $rewrite\_filter$
    end if
    if $Operator\_M_i\_sql$ not in $RSet$ **then**
      $rewrite\_operator$
    end if
    if $Operator\_M_i\_sql = RSet$ and $Filter\_M_i\_sql = RSet$ **then**
      $EnsembleFailure \leftarrow False$
      return M_i_sql
    end if
  end while
  if $EnsembleFailure$ **then**
    return emptySQL

one another.

The architecture of the hybrid system is presented in Figure 4.

6 Experiments and Results

In this section we present the experiments and the results we obtained using the hybrid system against the baseline obtained by the original Valuenet trained on the Spider corpus. Besides the three schemas presented in Section 3, we also consider here the following schemas from the the Spider corpus: *pets_1* and *dog_kennels*. We considered specifically these two DB schemas because the default system scored less than median on them. In Table 1, for the HR, WH, and BI DB schemas the training was the *ANS* corpora, and *nANS* part was kept for testing, see Section 4. That is, we use the *ANS* corpus for direct training, or it was subject to STDA as described in Section 6.2 which led to creation of different models to which the ensemble module described in 6.3 was applied. In the latest case, we created a series of 5 models for each schema by splitting the training into five parts, in $4:1$ ratio (each model was using 80% of the original seeds for training augmentation) The evaluation of the accuracy considers the result of the SQL from our system vs. the result of the gold SQL formula. A system inferred SQL is considered correct only if all values and only those ones are returned by the gold SQL as well (execution accuracy) In all the tables , # represents the size of corpus, *acc* represents the accuracy, *HS* is our system with DDM module, *STDA* is the seed training module and *ens*, the ensemble module. The training for our schema was always combined with the Spider training corpus to obtain a model. So, a a model for a new DB having 100 training examples is actually build from 7000(Spider)+100 examples.

In Table 3 we present the results of the HS system without STDA, that is, we used 100% of the seeds training.

Table 3: Accuracy of hybrid system no STDA

DB	#train	acc
HR	121	36%
WH	87	26%
BI	87	38%
Pets_1	30	32%
Dog_kennels	62	48%

The results for *pets_1* and *dog_kennels* are different than the ones reported in Section 4, because in this experiment these schemas were removed from Spider development corpus.

We also carried out a cross validation experiment on the 5 models obtained for each schema via STDA. The results are in Table 4. The results are very high for each 5th part used for testing, but still low for the actual test, *nANS*, as seen in the second column of Table 5 .

Table 4: Accuracy for cross validation

DB	#seed	#STDA	#acc 5th part
HR	90	1500	78%
WH	70	2000	72%
BI	80	3300	80%
Pets_1	30	1500	68%
Dog_kennels	66	1500	72%

Finally, we used the 5 models from the cross validation for each schema with STDA with Ensemble and we obtained the results shown in Table 5. The ensemble gets significant improvement over the individual models, see third column.

Table 5: Accuracy hybrid system+ensemble

DB	HS+STDA acc	HS+STDA+Ens acc
HR	50%	62%
WH	40%	48%
BI	62%	66%
Pets_1	35%	41%
Dog_kennels	55%	57%

Putting together the results obtained, We indicate what is the average accuracy gain/loss for each type of system and each schema, *ANS* and *nANS* in Table 6.The benefit in using the STDA was in double digit percent improvement for all schemas. The ensemble brings an extra 5 to 10% improvement. The ensemble was very precise, having more than 98% precision, with a recall of 47%.

In 4.1 we analyzed certain patterns of errors. To see how much the hybrid system can improve on **EA1**, we compiled a specific corpus for multivalue question/queries examples for HR DB schema. This corpus has 64 questions that have more than one value, such as `What Apple product has the largest stock in Bestbuy?` The

Table 6: Accuracy Gain over Baseline

DB	BS	HS STDA	Ens
HR	22%	+17%	+43%
WH	17%	+12%	+29%
BI	3%	+15%	+63%
Pets_1	30%	+6%	+11%
Dog_kennels	45%	+5%	+12%

Table 7: EA1-multivalue queries improvement

system	accuracy
defaul VL	13%
default VL+tr	16%
HS no STDA	38%
HS+STDA	54%
HS+STDA+ens	78%

training corpus for HR, consisting of 57 questions, contained only 19 questions with multiple values. The results we obtained applying the hybrid system only on the multiple values corpus are in Table 7. The *default VL* is the the out-of-the-box VL system, *default VL+tr* is the same system trained on the extra training, *HS* , *STDA* and *ens* stand for hybrid system, seed training data augmentation and ensemble techniques respectively.

The ensemble algorithm works particularly well for this type of corpus. It is because the majority of wrong SQL had the wrong reference to table and column, which makes them recoverable due to the DDM information used in ensemble.

EA2 issue was related to the logically compound filters. Because Spider code cannot correctly process those types of SQL elements, it is not fair to compare systems that employ any of our techniques vs. a system does not have any mechanism for dealing with compound logical operators. Currently, we do not known of any other system that manages this problem.

The efficiency of our solution is further proved by the experiments using **temporal questions**. The Spider corpus have a poor representation of temporal questions, less than 8% in training and less than 7% in dev, and, as pointed out in **EA3** the SQL associated contains string , not date evaluation. We manually created a corpus containing temporal questions for HR and WH. This corpus contains questions with one data value, like, `how many products were sold on 2/2/2014`, we refer to them as *tmp_1v*, and

Table 8: Temporal Questions corpus

temporal corpus	#HR	#WH
tmp_1v train	28	35
tmp_1v test	14	22
tmp_mv train	44	70
tmp_mv test	32	42

Table 9: EA3 - Temporal Questions Results

system	HR tmp_1v	WH tmp_mv
default VL	18%	0%
default VL+tr	24%	0%
HS+STDA+ens	58%	44%

questions that more than one data value, like `how many products are sold with prices greater than 200 dollars between 2017/01/01 and 2017/12/31`, we refer to as *tmp_mv*.

For *temp_1v* the baseline system had a performance similar to the one for non temporal questions. However, for *tmp_mv* the situation change: the baseline system achieved 0% accuracy, and the situation did not improved when the available temporal training was provided to the system to build a new model. On the other hand, our system HS+STDA+Ens obtained a good accuracy, see Table 9. The fact that VL system obtains 0% is due to the inability of the system to correlate a value with DB table and column, which the technique introduced here for STDA+ens remedies. In Appendix we provided a sample of user's questions and the output from the baseline system vs. our system.

7 Conclusion and Further Research

In this paper we present a hybrid system for the Text-to-SQL task. The architecture of this system was driven by seeking solutions to a problem that we found when we tried to run an encoder-decoder system with out-of-the-box configuration, namely a large drop in accuracy for our DB schemas. The problem was reported in the results published by other teams as well. Our solution was to construct a system that does schema linkage and employs a training data augmentation technique. The system showed a good accuracy for DB schemas in and out of the Spider corpus. We obtained a double figure improvement of the accuracy, in some cases close to 30%, and not less than 9%. The lowest increase was for a schema that was considered difficult in

the Spider corpus.

Our experiments strongly suggests that a perfect training corpus cannot be built such that the accuracy on any new schema will be high. However, by implementing schema linkage and specific training, automatically created, one can realize a high jump in accuracy for deep learning networks.

Our future research will focus on further improving the training with queries that are not properly represented in the Spider corpus and on the methodology for training data augmentation.

8 Appendix

Here we show examples of the questions and inferred SQL for the HR and WH schemas, respectively. For each question we present the result of the default VL system, first SQL, vs. the result of our system, HS+STDA+Ensemble, the second SQL. For each question, we show the error made by the default system and the name of the system that inferred the correct SQL. The questions are shown exactly as they were written by users.

8.1 WH DB schema examples of inferred SQL.

- who manufactured iphone **: wrong table; correction via Ensemble**
  - SELECT T1.name FROM manufacturers AS T1 JOIN products AS T2 ON T1.manufacturer_id = T2.manufacturer_id WHERE T1.name = 'IPHONE'
  - SELECT T1.name FROM manufacturers AS T1 JOIN products AS T2 ON T1.manufacturer_id = T2.manufacturer_id WHERE T2.type = 'IPHONE'

- what did Richard buy **: wrong path; correction via STDA**
  - SELECT T1.name FROM customers AS T1
    WHERE T1.customer_id = Richard and T1.name = 'Richard'
  - SELECT T1.type FROM products AS T1 JOIN sales_details AS T13 ON T1.product_id = T13.product_id JOIN sales AS T14 ON T13.sales_id T̄14.sales_id JOIN customers AS T2 ON T14.customer_id = T2.customer_id WHERE T2.name = 'Richard'

- who bought at bestbuy **: wrong path, wrong column; correction via STDA+Ensemble**
  - SELECT T1.name FROM customers AS T1 JOIN sales AS T13 ON T1.customer_id = T13.customer_id JOIN sales_details AS T14 ON T13.sales_id = T14.sales_id JOIN products AS T2 ON T14.product_id = T2.product_id WHERE T2.price = BESTBUY
  - SELECT DISTINCT T1.name FROM customers AS T1 JOIN sales AS T13 ON T1.customer_id = T13.customer_id JOIN shops AS T2 ON T13.shop_id = T2.shop_id WHERE T2.name = 'BESTBUY'

- what is sold at bestbuy **: wrong column, correction via Ensemble**
  - SELECT T1.name FROM vendors AS T1 JOIN sales AS T13 ON T1.vendor_id = T13.vendor_id JOIN sales_details AS T14 ON T13.sales_id = T14.sales_id JOIN products AS T2 ON T14.product_id = T2.product_id WHERE T2.price = BESTBUY
  - SELECT distinct T2.type FROM shops AS T11 JOIN sales AS T22 ON T11.shop_id = T22.shop_id JOIN sales_details AS T1 ON T22.sales_id = T1.sales_id JOIN products AS T2 ON T1.product_id = T2.product_id WHERE T11.name = 'BESTBUY'

- show me apple product names **: wrong table, correction via STDA+Ensemble**
  - SELECT T1.name FROM manufacturers AS T1 WHERE T1.name = 'APPLE'
  - SELECT T1.type FROM products AS T1 JOIN manufacturers AS T2 ON T1.manufacturer_id = T2.manufacturer_id WHERE T2.name = 'APPLE'

References

Yoav Artzi and Luke Zettlemoyer. 2011. Bootstrapping semantic parsers from conversations. In *Proceedings of the 2011 Conference on Empirical Methods in Natural Language Processing*.

Ben Bogin, Matt Gardner, and Jonathan Berant. 2019. Representing schema structure with graph neural networks for text-to-sql parsing. *arXiv preprint arXiv:1905.06241*.

Brunner. 2021. Valuenet source code.

Ursin Brunner and Kurt Stockinger. 2021. Valuenet: A natural language-to-sql system that learns from database information.

Ruichu Cai, Boyan Xu, Xiaoyan Yang, Zhenjie Zhang, Zijian Li, and Zhihao Liang. 2018. An encoder-decoder framework translating natural language to database queries.

Jianpeng Cheng, Siva Reddy, Vijay Saraswat, and Mirella Lapata. 2019. Learning an executable neural semantic parser. *Computational Linguistics*, 45(1):59–94.

Xiang Deng, Ahmed Hassan Awadallah, Christopher Meek, Oleksandr Polozov, Huan Sun, and Matthew Richardson. 2021. Structure-grounded pretraining for text-to-sql. *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*.

Jonas Gehring, Michael Auli, David Grangier, Denis Yarats, and Yann N. Dauphin. 2017. Convolutional sequence to sequence learning.

Jiaqi Guo, Zecheng Zhan, Yan Gao, Yan Xiao, Jian-Guang Lou, Ting Liu, and Dongmei Zhang. 2019. Towards complex text-to-sql in cross-domain database with intermediate representation. *arXiv preprint arXiv:1905.08205*.

Izzeddin Gur, Semih Yavuz, Yu Su, and Xifeng Yan. 2018. Dialsql: Dialogue based structured query generation. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, Melbourne, Australia. Association for Computational Linguistics.

Jonathan Herzig and Jonathan Berant. 2019. Don't paraphrase, detect! rapid and effective data collection for semantic parsing. *arXiv preprint arXiv:1908.09940*.

Jovan Kalajdjieski, Martina Toshevska, and Frosina Stojanovska. 2020. Recent advances in sql query generation: A survey.

Yaghmazadeh Navid, Wang Yuepeng, Dillig Isil, and Thomas Dillig. 2017. Query synthesis from natural language. *International Conference on Object-Oriented Programming*.

Ana-Maria Popescu, Oren Etzioni, and Henry Kautz. Towards a theory of natural language interfaces to databases.

Octavian Popescu, Ngoc Phuoc An Vo, Vadim Sheinin, Elahe Khorashani, and Hangu Yeo. 2019. Tackling complex queries to relational databases. In *Asian Conference on Intelligent Information and Database Systems*, pages 688–701. Springer.

Maxim Rabinovich, Mitchell Stern, and Dan Klein. 2017. Abstract syntax networks for code generation and semantic parsing. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics*.

Prakash R Shaurya Agarawal Naman Jain Mayank Singh Ronak Kaoshik, Rohit Patil. 2021. Acl-sql: Generating sql queries from natural language.

Peng Shi, Patrick Ng, Zhiguo Wang, Henghui Zhu, Alexander Hanbo Li, Jun Wang, Cicero Nogueira dos Santos, and Bing Xiang. 2020. Learning contextual representations for semantic parsing with generation-augmented pre-training. *arXiv preprint arXiv:2012.10309*.

Alane Suhr, Ming-Wei Chang, Peter Shaw, and Kenton Lee. 2020a. Exploring unexplored generalization challenges for cross-database semantic parsing. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*.

Alane Laughlin Suhr, Kenton Lee, Ming-Wei Chang, and Pete Shaw. 2020b. Exploring unexplored generalization challenges for cross-database semantic parsing.

Yibo Sun, Duyu Tang, Nan Duan, Jianshu Ji, Guihong Cao, Xiaocheng Feng, Bing Qin, Ting Liu, and Ming Zhou. 2019. Semantic parsing with syntax and table aware sql generation. *56th Annual Meeting of the Association for Computational Linguistics*.

Yu Tao, Zhang Rui, Yang Kai, Yasunaga Michihiro, Wang Dongxu, Li Zifan, Ma James, Li Irene, Yao Qingning, Roman Shanelle, Zhang Zilin, and Radev Dragomir. 2018. Spider: A large-scale human-labeled dataset for complex and cross-domain semantic parsing and text-to-sql task. *EMNLP*.

Sheinin Vadim, Khorasani Elahe, Yeo Hangu, Xu Kun, An Vo Ngoc, Phuoc, and Octavian Popescu. 2018. Quest: A natural language interface to relational databases. *LREC*.

Ngoc Phuoc An Vo, Octavian Popescu, Vadim Sheinin, Elahe Khorasani, and Hangu Yeo. 2019. A natural

language interface supporting complex logic questions for relational databases. In *International Conference on Applications of Natural Language to Information Systems*, pages 384–392. Springer.

Bailin Wang, Richard Shin, Xiaodong Liu, Oleksandr Polozov, and Matthew Richardson. 2019. Rat-sql: Relation-aware schema encoding and linking for text-to-sql parsers. *arXiv preprint arXiv:1911.04942*.

Xiaojun Xu, Chang Liu, and Dawn Song. 2017. Sqlnet: Generating structured queries from natural language without reinforcement learning. *arXiv preprint arXiv:1711.04436*.

Ziyu Yao, Yu Su, Huan Sun, and Wen tau Yih. 2010. Model-based interactive semantic parsing: A unified framework and a text-to-sql case study. *IJCNLP*.

Hangu Yeo, Elahe Khorasani, Vadim Sheinin, Ngoc Phuoc An Vo, Octavian Popescu, and Petros Zerfos. 2021. Programmatic database language generation for big data applications. In *2021 IEEE International Conference on Big Data (Big Data)*, pages 2848–2856. IEEE.

Pengcheng Yin, Zhengdong Lu, Hang Li, and Ben Kao. 2016. Neural enquirer: Learning to query tables with natural language.

Rui Zhang, Tao Yu, He Yang Er, Sungrok Shim, Eric Xue, Xi Victoria Lin, Tianze Shi, Caiming Xiong, Richard Socher, and Dragomir Radev. 2019. Editing-based sql query generation for cross-domain context-dependent questions. *arXiv preprint arXiv:1909.00786*.

Ruiqi Zhong, Tao Yu, and Klein Dan. 2020. Semantic evaluation for text-to-sql with distilled test suites.

Mintaka: A Complex, Natural, and Multilingual Dataset for End-to-End Question Answering

Priyanka Sen
Amazon Alexa AI
Cambridge, UK
sepriyan@amazon.com

Alham Fikri Aji
Amazon Alexa AI
Cambridge, UK
afaji@amazon.com

Amir Saffari
Amazon Alexa AI
Cambridge, UK
amsafari@amazon.com

Abstract

We introduce MINTAKA, a complex, natural, and multilingual dataset designed for experimenting with end-to-end question-answering models. Mintaka is composed of 20,000 question-answer pairs collected in English, annotated with Wikidata entities, and translated into Arabic, French, German, Hindi, Italian, Japanese, Portuguese, and Spanish for a total of 180,000 samples. Mintaka includes 8 types of complex questions, including superlative, intersection, and multi-hop questions, which were naturally elicited from crowd workers. We run baselines over Mintaka, the best of which achieves 38% hits@1 in English and 31% hits@1 multilingually, showing that existing models have room for improvement. We release Mintaka at https://github.com/amazon-research/mintaka.

1 Introduction

Question answering (QA) is the task of learning to predict answers to questions. Approaches to question answering include knowledge graph (KG) based methods, which use structured data to find the correct answer (Miller et al., 2016; Saxena et al., 2020); machine reading comprehension methods, which extract answers from input documents (Rajpurkar et al., 2016; Kwiatkowski et al., 2019); open domain methods, which learn to retrieve relevant documents and extract or generate answers (Zhu et al., 2021), and closed-book methods, which use knowledge implicitly stored in model parameters to answer questions (Roberts et al., 2020).

With state-of-the-art techniques, QA models can achieve high performance on simple questions (Shi et al., 2020, 2021) that require a single fact lookup in either a knowledge graph or a text document (e.g., "*Where was Natalie Portman born?*"). However not all questions are simple in real-world applications. We define *complex questions* (Lan et al., 2021) as questions that require an operation beyond a single fact lookup, such as multi-hop, comparative, or set intersection questions. For example, "*What movie had a higher budget, Titanic or Men in Black?*" requires looking up the budget of two movies, comparing the values, and selecting the movie with the higher budget. Handling more complex questions remains an open problem.

One of the challenges in measuring and improving QA performance on complex questions is a lack of datasets. While several QA datasets exist, they have shortcomings of being either large but simple, such as SimpleQuestions (Bordes et al., 2015), or complex but small, such as ComplexQuestions (Bao et al., 2016) or QALD (Usbeck et al., 2018). Recently, several large and complex datasets have been released, including KQA Pro (Shi et al., 2020) and GrailQA (Gu et al., 2021). These datasets use automatically generated questions followed by human paraphrasing, which can result in less natural questions, such as "*Is the WOEID of Tuscaloosa 14605?*" (KQA Pro) or "*1520.0 is the minimum width for which size rail gauge*?" (GrailQA). This can lead to a mismatch between training data and real-world use cases of QA models.

In order to address these gaps, we release MINTAKA, a large, complex, naturally-elicited, and multilingual question answering dataset. Mintaka contains 20,000 question-answer pairs elicited in English from crowd workers. We link Mintaka to a knowledge graph by asking crowd workers to annotate the question and answer text with Wikidata IDs. Professional translators translated the 20,000 English questions into Arabic, French, German, Hindi, Italian, Japanese, Portuguese, and Spanish, creating a total dataset size of 180,000 questions.

In this paper, we present an overview of Mintaka in §3, explain how we built Mintaka in §4, provide a statistical analysis of the dataset in §5, including a demographic analysis of our crowd workers in §5.3. Finally in §6, we present results of existing baseline models on Mintaka, the best of which scores 38%

Proceedings of the 29th International Conference on Computational Linguistics, pages 1604–1619
October 12–17, 2022.

Dataset	Samples	Text or KG	Complex	Natural	Languages
SQuAD	150K	Wikipedia	×	×	1
XQuAD	2K	Wikipedia	×	×	11
Natural Questions	300K	Wikipedia	×	✓	1
HotpotQA	100K	Wikipedia	✓	×	1
DROP	100K	Wikipedia	✓	×	1
WebQuestionsSP	5K	FreeBase	×	✓	1
ComplexQuestions	2K	FreeBase	✓	✓	1
ComplexWebQuestions	35K	FreeBase	✓	×	1
LC-QuAD 2.0	30K	Wikidata, DBPedia	✓	×	1
GrailQA	64K	Wikidata	✓	×	1
KQA Pro	120K	Wikidata	✓	×	1
QALD	400	DBPedia	✓	✓	11
Mintaka (ours)	20K	Wikidata	✓	✓	9

Table 1: Comparison of Mintaka to existing QA datasets

hits@1. These results show that existing models have room for improvement.

We publicly release the Mintaka dataset with our randomly split train (14,000 samples), dev (2,000 samples), and test (4,000 samples) sets at https://github.com/amazon-research/mintaka.

2 Related Works

Question answering has no shortage of datasets. Datasets for question-answering with reading comprehension, such as SQuAD (Rajpurkar et al., 2016) or Natural Questions (Kwiatkowski et al., 2019) are often large, and some are even multilingual, such as XQuAD (Artetxe et al., 2019), MLQA (Lewis et al., 2019), or TyDi QA (Clark et al., 2020). These datasets, however, are not explicitly built to be complex, and the answer is usually found in a single passage of text.

HotpotQA (Yang et al., 2018) and MuSiQue (Trivedi et al., 2022) add complexity to reading comprehension by introducing multi-hop questions where the answer requires reasoning over two documents, but neither of these datasets naturally elicit their questions. HotpotQA pre-selects two Wikipedia passages and asks workers to write questions using both passages, and MuSiQue composes multi-hop questions from existing single hop questions. DROP (Dua et al., 2019) is another complex reading comprehension dataset, including complex operations such as addition, counting, and sorting. Again, DROP asks crowd workers to write questions about a selected Wikipedia passage. DROP additionally introduces a constraint where workers need to write questions that can't be solved by an existing model.

Within knowledge graph-based question answering (KGQA), WebQuestionsSP (Berant et al., 2013; Yih et al., 2016) and ComplexQuestions (Bao et al., 2016) are more natural QA datasets. Both collected real user questions using search query logs or the Google Suggest API. The answers were annotated manually using FreeBase as a knowledge graph. WebQuestionsSP contains mostly simple questions, but ComplexQuestions is more complex, including multi-hop questions, temporal constraints, and aggregations. The main drawback of these datasets is size. WebQuestionsSP contains 5K QA pairs, while ComplexQuestions contains only 2K.

ComplexWebQuestions (Talmor and Berant, 2018) is a dataset based on WebQuestionsSP, which increases the size to 35K QA pairs and introduces more complex operations, including multi-hop, comparatives, and superlatives. However ComplexWebQuestions loses some naturalness, as the dataset is built by automatically generating queries and questions, and then asking crowd workers to paraphrase the generated questions.

Recently, several larger-scale complex KGQA datasets have been released. LC-QuAD 2.0 (Dubey et al., 2019) includes 30K questions, including multi-hop questions, and uses the more up-to-date Wikidata and DBpedia knowledge graphs. GrailQA (Gu et al., 2021) is even larger at 64K questions based on FreeBase with complex questions, including multi-hop, count, and comparatives. KQA Pro

(Shi et al., 2020) is even larger still with 120K questions based on Wikidata and with complex questions, including intersection and superlatives. All these datasets make the trade-off of scale over naturalness. To collect question-answer pairs, the authors generate queries from a knowledge graph, generate questions based on the queries, and then ask crowd workers to paraphrase the questions.

Finally, most datasets are only in English. Multilingual and complex datasets are rare. QALD 2018 (Usbeck et al., 2018) is one multilingual and complex dataset including 11 languages and complex operations such as counts and comparatives, however includes only 400 questions.

By building Mintaka, we hope to address an important gap in existing datasets. Mintaka question-answer pairs are both complex and naturally-elicited from crowd workers with no restrictions on what facts or articles the questions can be about. We also translate Mintaka into 8 languages, making it one of the first large-scale complex and multilingual question answering datasets. A comparison of Mintaka to existing datasets can be seen in Table 1.

3 Mintaka

Mintaka is a complex question answering dataset of 20,000 questions collected in English and translated into 8 languages, for a total of 180,000 questions. Mintaka contains question-answer pairs written by crowd workers and annotated with Wikidata entities in both the question and answer.

We collected questions in eight topics, which were chosen for being broadly appealing and suitable for writing complex questions: MOVIES, MUSIC, SPORTS, BOOKS, GEOGRAPHY, POLITICS, VIDEO GAMES, and HISTORY. Since we want Mintaka to be a complex question answering dataset, we explicitly collected questions in the following complexity types: (Note: All examples below are from the Mintaka dataset.)

- COUNT: questions where the answer requires counting. For example, Q: How many astronauts have been elected to Congress? A: 4
- COMPARATIVE: questions that compare two objects on a given attribute (e.g., age, height). For example, Q: Is Mont Blanc taller than Mount Rainier? A: Yes
- SUPERLATIVE: questions about the maximums or minimums of a given attribute. For example, Q: Who was the youngest tribute in the Hunger Games? A: Rue
- ORDINAL: questions based on an object's position in an ordered list. For example, Q: Who was the last Ptolemaic ruler of Egypt? A: Cleopatra
- MULTI-HOP: questions that require 2 or more steps (multiple *hops*) to answer. For example, Q: Who was the quarterback of the team that won Super Bowl 50? A: Peyton Manning
- INTERSECTION: questions that have two or more conditions that the answer must fulfill. For example, Q: Which movie was directed by Denis Villeneuve and stars Timothee Chalamet? A: Dune
- DIFFERENCE: questions with a condition that contains a negation. For example, Q: Which Mario Kart game did Yoshi not appear in? A: Mario Kart Live: Home Circuit
- YES/NO: questions where the answer is Yes or No. For example, Q: Has Lady Gaga ever made a song with Ariana Grande? A: Yes.
- GENERIC: questions where the worker was only given the topic and no constraints of complexity. These tend to be simpler fact lookups, such as Q: Where was Michael Phelps born? A: Baltimore, Maryland

For each of the 8 topics, we collected 250 questions per complexity type and 500 generic questions, for a total of 2,500 questions per topic.

We also collected translations of the 20,000 English questions in 8 languages using professional translators. Since all questions were collected in English from U.S. workers, the questions may have a U.S. bias in terms of the entities (for example, U.S. politicians or books written in English). This is a choice we make since it allows us to create a fully parallel dataset where models can be easily compared across languages. This choice was also made in previous QA datasets (Usbeck et al., 2018; Artetxe et al., 2019; Lewis et al., 2019).

4 Dataset Collection

To build our dataset, we used Amazon Mechanical Turk (MTurk) in three different tasks. All of our MTurk workers were located in the United States, and to ensure high quality, we required workers

have an approval rating of 98% and at least 5,000 approved tasks. Each of our tasks are explained in the sections below, and examples of the interfaces can be seen in Appendix A.

4.1 Question Elicitation

The first task was to elicit complex questions. To do this, we created tasks for each topic/complexity pair (e.g., Superlative Movie questions, Ordinal Sports questions, etc.). In each task, a worker was asked to write 5 questions and answers about the topic using the given complexity type. The questions and answers were written in free text fields. We had no restrictions on what sources workers could use to write their questions, so workers were not limited to writing questions based on a given article or facts. Workers were given explanations of the complexity type and examples in the instructions. The topics were left general, so within History, workers could write about Ancient Egypt as well as World War II.

For Count and Superlative answers, we additionally asked workers to provide a numerical value as part of the answer. For example, in Count questions, workers would both provide the answer as a number (e.g., 3) as well as the entities that make up that answer (e.g., Best Picture, Best Adapted Screenplay, and Best Film Editing). In Superlative questions, workers provided the answer (e.g., Missouri River) as well as the numerical value that makes the entity the maximum or minimum (e.g., 2,341 miles). Additionally in Count questions, if a question had multiple answers, we asked workers to list a minimum of five. For example, for the question "*How many cities have hosted a Summer Olympics?*", a worker could give the numerical answer 23 but provide only five of the cities. For this reason, answers to questions with more than five entities are not guaranteed to be complete but instead provide a sample of the correct answer.

We paid $1.25 per task to write five questions. Workers were limited to completing one task per topic-complexity pair. After collection, we also surveyed the MTurk workers who completed our Question Elicitation task about their demographics. The results of this survey are discussed in §5.3.

4.2 Answer Entity Linking

Answers were collected in the previous task in natural language. In order to link the answers to a knowledge graph, we built an Answer Entity Linking task. We chose to link the answers to Wikidata, since it is a large and up-to-date public knowledge graph. Although we link to Wikidata, we don't guarantee that every question can be answered by Wikidata at the time of writing. It is possible that there are missing or incomplete facts that would prevent a KGQA system from reaching the answer entity in Wikidata given the question.

In this task, workers were shown a question-answer pair and asked to 1) highlight the entities in the answer, and 2) search for the entities on Wikidata and provide the correct URLs. We built a UI for MTurk workers where they could easily highlight entities and the highlighted entities would automatically generate links to search Wikidata.

Each answer was annotated by two MTurk workers. For agreement, we required two workers to identify the same entities and the same Wikidata URLs for all entities. If there was disagreement, we sent the question-answer pair to a third annotator. Question-answer pairs where the answer was a number or *yes* or *no* were excluded from answer entity linking. Overall, we annotated 20,996 answer entities and achieved 82% agreement after two annotators and 97% agreement after three annotators. The remaining 3% were verified by the authors.

We paid a base rate of $0.10 per task, which consisted of a single question-answer pair. If the answer had multiple entities, we paid a $0.05 bonus for every additional entity identified that was agreed upon by another annotator.

4.3 Question Entity Linking

An end-to-end question answering model can be trained using the question and answer alone (Oliya et al., 2021). However to better evaluate end-to-end methods and train models requiring entities, we also created an MTurk task to link entities in the question text.

Linking entities in questions is more challenging than answers. While answer texts are often short and contain a clear entity (e.g., "Joe Biden"), question texts can contain multiple possible entities. In the question "*Who is the president of the United States?*", a worker could select "United States", or "president" and "United States", or even "president of the United States". Since early test runs showed it would be difficult to get agreement on question entities, we modified the task so workers only verified a span and linked the entity in Wikidata.

To identify spans in questions, we used spaCy's (Honnibal et al., 2020) en_core_web_trf model to identify named entities and noun chunks with cap-

Question Length	
English	10.2
Arabic	9.9
German	9.6
Spanish	10.8
French	12.4
Hindi	11.0
Italian	10.6
Japanese (in characters)	29.6
Portuguese	10.3
Entities	
Entities per Question	1.8
Entities per Answer	1.3
Unique Question Entities	7,303
Unique Answer Entities	8,607
Unique Entities	13,232
Question to Answer Entity	
within one hop	62%
within two hops	97%
Answer Types	
Entity	0.72
Boolean	0.14
Numerical	0.07
Date	0.06
String	0.001

Table 2: Statistics about the Mintaka dataset

Q: Which Studio Ghibli (Q182950) movie scored the lowest on Rotten Tomatoes (Q105584)? **A:** Earwig and the Witch (Q96031360)
Q: Which revolution lasted longer, the French (Q6534) or the American (Q192769)? **A:** American Revolution (Q192769)
Q: When Franklin D. Roosevelt (Q8007) was first elected, how long had it been since someone in his party (Q7278) won the presidential election (Q47566)? **A:** 16 years
Q: Which member of the Red Hot Chili Peppers (Q10708) appeared in Point Break (Q1146552)? **A:** Anthony Kiedis (Q204751)
Q: Which Mass Effect (Q953242) game does not include Commander Shepard (Q3683919) as the main character? **A:** Mass Effect: Andromeda (Q20113552)

Table 3: Example question-answer pairs from Mintaka. Question and answer annotations are shown here in-line with Wikidata Q-codes.

italized words in the English sentences. We then gave workers the question with a predicted entity highlighted. Workers were shown one entity at a time and asked to first verify or modify the highlighted entity and then link to Wikidata.

For question entities that were not seen before, we had the entity annotated by two annotators, followed by a third in case of no agreement. For some question entities, we were able to exact string match them against entities that were already annotated in the Answer Entity Linking task, for example United States → Q30. In cases where we had a match, the question entity was annotated by one annotator and only went to a second annotator if there was no agreement.

We annotated 12,819 new entities, for which we had 68% agreement after two annotators and 78% agreement after three annotators, and 15,075 seen entities, for which we had 80% agreement after one annotator, and 98% agreement after two annotators. The remaining entities were verified by the authors. We paid $0.10 per entity. The spans of the question entities are only annotated in English, and so English questions in Mintaka come with both the entity ID and the span, while all translated questions have only the entity ID.

4.4 Translations

We translated the 20,000 questions in Mintaka to the following languages and locales: Arabic (Saudi Arabia), French (France), German (Germany), Hindi (India), Italian (Italy), Japanese (Japan), Portuguese (Brazil), and Spanish (Mexico). Translation is our only dataset collection step where we do not use MTurk. Early experiments with MTurk on translation tasks and editing automatic translations tasks had poor results and a lack of workers in some languages, such as Japanese. For this reason, we use professional translators.

5 Dataset Analysis

5.1 Dataset Statistics

Statistics about the Mintaka dataset are shown in Table 2. Question length is based on white space

splitting in all languages except Japanese, where the question length is in characters. In total, 13,232 unique Wikidata entities appear across all questions. The most common question entities are the United States (Q30; 1,498 questions), President of the United States (Q11696; 564 questions), and Super Bowl (Q32096, 344 questions). The most common answer entities are California (Q99, 102 answers), Alaska (Q797, 88 answers), and the United States (Q30, 80 answers).

Mintaka was built with only questions and answers, so we do not know the correct query path. However we can calculate an upper bound of answerable questions using Wikidata by identifying the percentage of questions that have a path connecting the question entity to the answer entity. We find that there is a path connecting the question and answer entity 62% of the time within 1 hop and 97% of the time within 2 hops for questions linked to Wikidata entities.

A majority (72%) of the questions in Mintaka can be answered using an entity. 14% can be answered using a boolean, in yes/no or comparative questions. 7% can be answered using a number, such as someone's age. 6% can be answered using a date, such as a date of birth. And finally, 0.1% have answers in the form of a string, for example, someone's nickname. Examples of QA pairs are in Table 3 with more examples in Appendix C.

5.2 Naturalness Evaluation

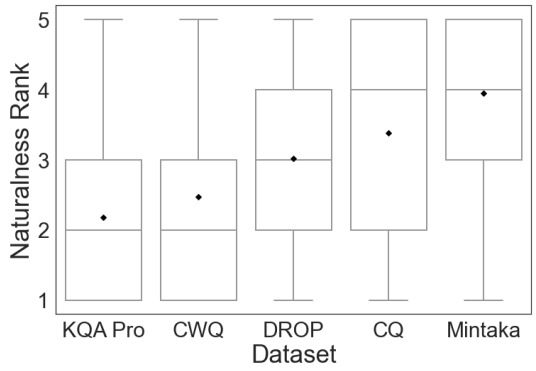

Figure 1: A box plot showing the quartile, median, and mean (black diamond) naturalness rank for each dataset from 1 (least natural) to 5 (most natural).

By naturally eliciting complex questions from MTurk workers, we aimed to collect questions that were closer to what users may ask in real-world settings. In order to evaluate how Mintaka compares to previous complex QA datasets, we ran a naturalness evaluation task on Mturk with four comparison datasets. We compared datasets that collected questions in different ways: KQA Pro automatically generated questions, ComplexWebQuestions (CWQ) automatically generated questions built off WebQuestions, DROP naturally elicited questions about a given Wikipedia passage, and ComplexQuestions (CQ) collected natural questions from user logs. We compared these datasets to Mintaka in a task where workers were shown 5 questions, one from each dataset, and asked to rank them from 1 (least natural) to 5 (most natural). We uniformly sampled 500 questions from each dataset and grouped them into quartiles by length for each dataset (i.e., the longest questions in Mintaka are grouped with the longest questions in other datasets).

The results are in Figure 1 and show that Mintaka is on average ranked higher in naturalness than all other datasets. We also find that Mintaka ranks significantly higher than the other datasets using a two-sample Kolmogorov-Smirnov test (with p-values < 0.001). This shows that Mintaka questions are perceived as more natural than automatically generated or passage-constrained questions. Although ComplexQuestions contains real user questions, the questions are collected from search logs and can be phrased ungrammatically (e.g., "*when did miami dolphins win super bowl?*"), leading to a wider range of ranks. These results confirm that Mintaka is both a complex and natural dataset.

5.3 Demographics of MTurk Workers

In total, 516 MTurk workers completed 3,503 Question Elicitation tasks to collect complex questions (some questions from tasks were removed as duplicates or under-sampled for a balanced dataset size). In order to better understand and measure who our dataset best represents, we invited all workers who completed a Question Elicitation task to participate in a demographic survey. We paid workers $1.25 to complete the survey. We received 400 responses (78% response rate). Worker IDs were only used to invite MTurk workers to take part in the survey. All demographic data is anonymous with no way to link the data back to the Worker IDs, and the data is only analyzed in aggregate.

Table 4 can be used as an indicator of who this dataset is best for modeling and for what populations it may be less representative. For example, we had more workers identify as male than female

GENDER	
Male	0.58
Female	0.42
AGE	
18-24	0.02
25-34	0.39
35-44	0.32
45-54	0.13
55-64	0.10
65+	0.04
ETHNICITY	
White	0.73
Asian	0.10
Black	0.07
Hispanic	0.06
Multiracial	0.04
EDUCATION	
High school	0.16
Associate's	0.23
Bachelor's	0.50
Master's	0.09
Doctoral	0.02
EMPLOYMENT	
Employed, full-time	0.65
Self-employed	0.17
Employed, part-time	0.09
Not employed	0.05
Retired	0.02
Homemaker	0.01
RESIDENTIAL AREA	
Urban	0.33
Suburban	0.52
Rural	0.15
U.S. REGION	
Northeast	0.21
South	0.35
Midwest	0.21
West	0.23

Table 4: Results of the demographic survey of workers who completed the Question Elicitation task. Options that received less than 1% of responses are not shown.

(58% vs. 42%). Only 2% of workers were between the ages of 18-24 (and workers below 18 cannot register on MTurk), while 72% of workers were between the ages of 25 and 44. We also had fewer workers who identified as Black (7%) or Hispanic (6%) than the U.S. Census (U.S. Census Bureau, 2021) estimates of the general population (13% and 19%, respectively), while seeing a slightly higher percentage of workers identifying as Asian (10% of workers vs. 6% in the U.S. Census).

Our workers also tend to be more educated with 61% reporting that they hold a Bachelor's degree or higher, while the U.S. Census estimates 32% of the general population holds a Bachelor's degree or higher (U.S. Census Bureau, 2021). Our workers are almost all employed either full or part-time (91%), and largely live in urban or suburban areas (85%). Geographic distribution across the U.S. shows more (35%) workers in the South.

6 Baselines

Model	Hits@1
LANGUAGE MODELS	
T5	0.28
T5 for CBQA (zero-shot)	0.20
T5 for CBQA (fine-tuned)	**0.38**
KGQA MODELS	
KVMemNet	0.12
EmbedKGQA	0.18
Rigel	0.20
RETRIEVER-READER MODELS	
DPR (zero-shot)	0.15
DPR (trained)	0.31

Table 5: Results of English baseline models on Mintaka

6.1 Models

We evaluate eight baselines on Mintaka. Since Mintaka contains only question and answer pairs, we only use models that can be trained end-to-end. We evaluate 3 language models, 3 knowledge graph-based models, and 2 retriever-reader models. For language and retriever-reader models, we use the answers written by the crowd workers in the Question Elicitation task as the label. For our knowledge graph, we use a Wikidata snapshot from October 18, 2021. We evaluate all of our baselines in English (Table 5) and three of our baselines that could easily be set up multilingually in all languages (Table 6). Details on training data size can be found in Appendix B.

Model	multi	ar	de	es	fr	hi	it	ja	pt
MT5	0.16	0.15	0.16	0.16	0.16	0.16	0.16	0.15	0.16
T5 for CBQA (translated)	0.31	0.27	0.34	0.32	0.33	0.30	0.32	0.28	0.31
Rigel	0.19	0.18	0.19	0.19	0.19	0.17	0.19	0.20	0.18

Table 6: Results of baselines evaluated multilingually and in individual languages. Scores are reported as hits@1.

T5 AND MT5 (Raffel et al., 2020; Xue et al., 2021) are baselines that only use a language model to predict answers to questions. We use the XL versions of T5 for English and MT5 for all other languages. We fine-tune both for 10,000 steps.

T5 FOR CLOSED BOOK QA (CBQA) (Roberts et al., 2020) is an extension of T5 that is fine-tuned as a QA model that can implicitly store and retrieve knowledge without an external source. We use Roberts et al. (2020)'s T5-XL model and evaluate on Mintaka both as zero-shot with a model fine-tuned on Natural Questions and with a model fine-tuned on Mintaka for 10,000 steps. We run an additional translation baseline where we automatically translate non-English questions to English using the M2M_100 model (Fan et al., 2020) and use our English model to return answers.

KVMEMNET: Key-Value Memory Networks (Miller et al., 2016) work by first storing knowledge graph triples in a key-value structured memory. Then given a question, the model learns which keys are relevant to the question, and uses the values of those keys to return an answer. We follow the implementation by Shi et al. (2020).

EMBEDKGQA (Saxena et al., 2020) is a method that incorporates pre-trained knowledge graph embeddings into a KGQA model. EmbedKGQA consists of 1) a KG embedding module, 2) a question embedding module, and 3) an answer scoring module, which combines the question and KG embeddings to score and select answer entities. Since EmbedKGQA makes predictions over answer entities, we exclude questions where the answer is not an entity during training time, and count these as failures during test time. Scores for the subset of the test set that is answerable by the model can be found in Appendix B.

RIGEL (Oliya et al., 2021; Sen et al., 2021) is an end-to-end question answering model based on ReifiedKB (Cohen et al., 2020). Rigel uses an encoder to encode the question and a decoder to return a probability distribution over all relations in the knowledge graph. The relations are followed in the knowledge graph to return predicted answers. For the encoder, we use RoBERTa (Liu et al., 2019) for English and XLM-RoBERTa (Conneau et al., 2020) for all other languages. Again, since Rigel predicts over answer entities, we exclude questions where the answer is not an entity during training time and count them as failures during test time.

DENSE PASSAGE RETRIEVAL (DPR) (Karpukhin et al., 2020) is a retriever-reader method that uses a dense retriever model to identify relevant Wikipedia passages given a question, followed by a reader model to score answer spans from the retrieved passages. For the retriever, we use Karpukhin et al. (2020)'s model trained on Natural Questions, and for the reader, we evaluate both zero-shot with a model trained on Natural Questions and with a model trained on Mintaka. The reader sees the top 50 retrieved passages, and we take the highest-scoring span as the answer.

6.2 Analysis

The results of the baselines in English in terms of hits@1 can be seen in Table 5. Hits@1 is calculated based on the number of samples where the top prediction from the model matches the labeled answer, either as exact string match for text answers or as entity IDs for entity answers. A further breakdown per complexity type is in Appendix C. The best-performing model is the fine-tuned T5 for Closed Book QA with 38% hits@1. An analysis of the outputs shows that even though the model does not have access to an external knowledge source, it does recall factual information, such as the capital of Iraq is Baghdad. For more complex questions, the model can usually predict in the correct neighborhood of the answer. For example, for "*What is the second Marvel Cinematic Universe movie chronologically?*", the model predicts "*Thor: The Dark World*", which is a Marvel movie, however it lacks the complex reasoning functionality to calculate the second movie chronologically.

The trained DPR model has the second highest

score with 31% hits@1. We find that DPR can handle complex questions, as long as the complex reasoning is already done in the passage. For example, the model can answer "*When did Roger Federer win his first Grand Slam?*" with "*2003*" from a passage that includes, "Roger Federer won his first Grand Slam title in the 2003 Wimbledon Championships." However in cases where the reasoning is not included in the passage, the model struggles. For example given the same question about the second Marvel movie, the model predicts Iron Man and fails to find a passage explicitly mentioning the second movie chronologically.

Finally, the best-performing KGQA model is Rigel with 20% hits@1. Our KGQA baselines handle only entity answers and can only traverse the knowledge graph by following relations, so they are limited. Although the KGQA models score lower than the other models, they do have advantages. Language models at the scale of T5-XL are computationally expensive and the knowledge stored in the parameters is static. Knowledge graph based methods like Rigel can be updated easily by updating the external knowledge graph and can also return more interpretable answers. Given the Marvel question again, Rigel predicts a path from the Marvel Cinematic Universe to all the Marvel films. Rigel can't perform sorting or filtering, but it's possible to see what the next steps should be: identifying the chronological order, ordering by chronological order, and finding the second in the ordered list. Understanding how a model has arrived at an answer and what steps should be added to arrive at the correct answer are useful features for debugging and improving KGQA models.

Table 6 shows results on Mintaka evaluated both multilingually on all languages and in each language individually. For almost all models, the results are slightly lower than in English. The MT5 language model has worse performance than the English T5 model, which may be because unlike T5, MT5 is not pre-trained on any supervised tasks. Our T5 for CBQA model using English translations outperforms MT5, however the scores are still lower than on the original English questions, so the automatic translations do degrade performance. For Rigel, the main gap to English is on the encoding side, where we use XLM-RoBERTa instead of RoBERTa, showing a gap in the performance between the multilingual encoder and English encoder. All models show that there is still work needed for parity across all languages.

Overall, the baselines show that Mintaka is a challenging dataset. None of our baselines explicitly handle all of the complexity types available in Mintaka. The language model-only models especially struggle to handle questions that require numerical operations such as counts. The knowledge graph-based models rely on relation following to traverse the knowledge graph to an answer. This prevents models from correctly predicting answers that require more complex operations, even if the facts required are available in the knowledge graph. Adding additional operations and learning to select the correct operation for each question could lead to significant improvement. A combination of powerful language models, potentially to encode questions or identify question entities, with the interpretable facts and operations available in a knowledge graph is a promising direction to create better models on Mintaka.

7 Conclusions

In this paper, we introduce Mintaka, an end-to-end question answering dataset linked to Wikidata. Mintaka addresses an important gap in QA datasets by being large-scale, complex, naturally-elicited, and multilingual. Our baselines show that there is room for improvement in existing methods to handle complex questions, especially in all languages. With the release of Mintaka, we hope to encourage researchers to continue pushing the boundaries of question answering to handle more complex questions in more languages.

References

Mikel Artetxe, Sebastian Ruder, and Dani Yogatama. 2019. On the cross-lingual transferability of monolingual representations. *CoRR*, abs/1910.11856.

Junwei Bao, Nan Duan, Zhao Yan, Ming Zhou, and Tiejun Zhao. 2016. Constraint-based question answering with knowledge graph. In *Proceedings of COLING 2016, the 26th International Conference on Computational Linguistics: Technical Papers*, pages 2503–2514.

Jonathan Berant, Andrew Chou, Roy Frostig, and Percy Liang. 2013. Semantic parsing on freebase from question-answer pairs. In *Proceedings of the 2013 conference on empirical methods in natural language processing*, pages 1533–1544.

Antoine Bordes, Nicolas Usunier, Sumit Chopra, and Jason Weston. 2015. Large-scale simple question

answering with memory networks. *arXiv preprint arXiv:1506.02075*.

Jonathan H Clark, Eunsol Choi, Michael Collins, Dan Garrette, Tom Kwiatkowski, Vitaly Nikolaev, and Jennimaria Palomaki. 2020. Tydi qa: A benchmark for information-seeking question answering in typologically diverse languages. *Transactions of the Association for Computational Linguistics*, 8:454–470.

William W. Cohen, Haitian Sun, R. Alex Hofer, and Matthew Siegler. 2020. Scalable neural methods for reasoning with a symbolic knowledge base. In *8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, April 26-30, 2020*. OpenReview.net.

Alexis Conneau, Kartikay Khandelwal, Naman Goyal, Vishrav Chaudhary, Guillaume Wenzek, Francisco Guzmán, Edouard Grave, Myle Ott, Luke Zettlemoyer, and Veselin Stoyanov. 2020. Unsupervised cross-lingual representation learning at scale. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 8440–8451, Online. Association for Computational Linguistics.

Dheeru Dua, Yizhong Wang, Pradeep Dasigi, Gabriel Stanovsky, Sameer Singh, and Matt Gardner. 2019. DROP: A reading comprehension benchmark requiring discrete reasoning over paragraphs. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 2368–2378, Minneapolis, Minnesota. Association for Computational Linguistics.

Mohnish Dubey, Debayan Banerjee, Abdelrahman Abdelkawi, and Jens Lehmann. 2019. Lc-quad 2.0: A large dataset for complex question answering over wikidata and dbpedia. In *Proceedings of the 18th International Semantic Web Conference (ISWC)*. Springer.

Angela Fan, Shruti Bhosale, Holger Schwenk, Zhiyi Ma, Ahmed El-Kishky, Siddharth Goyal, Mandeep Baines, Onur Celebi, Guillaume Wenzek, Vishrav Chaudhary, Naman Goyal, Tom Birch, Vitaliy Liptchinsky, Sergey Edunov, Edouard Grave, Michael Auli, and Armand Joulin. 2020. Beyond english-centric multilingual machine translation. *arXiv preprint*.

Yu Gu, Sue Kase, Michelle Vanni, Brian Sadler, Percy Liang, Xifeng Yan, and Yu Su. 2021. Beyond iid: three levels of generalization for question answering on knowledge bases. In *Proceedings of the Web Conference 2021*, pages 3477–3488.

Matthew Honnibal, Ines Montani, Sofie Van Landeghem, and Adriane Boyd. 2020. spaCy: Industrial-strength Natural Language Processing in Python.

Vladimir Karpukhin, Barlas Oguz, Sewon Min, Patrick Lewis, Ledell Wu, Sergey Edunov, Danqi Chen, and Wen-tau Yih. 2020. Dense passage retrieval for open-domain question answering. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 6769–6781, Online. Association for Computational Linguistics.

Tom Kwiatkowski, Jennimaria Palomaki, Olivia Redfield, Michael Collins, Ankur Parikh, Chris Alberti, Danielle Epstein, Illia Polosukhin, Jacob Devlin, Kenton Lee, et al. 2019. Natural questions: a benchmark for question answering research. *Transactions of the Association for Computational Linguistics*, 7:453–466.

Yunshi Lan, Gaole He, Jinhao Jiang, Jing Jiang, Wayne Xin Zhao, and Ji-Rong Wen. 2021. A survey on complex knowledge base question answering: Methods, challenges and solutions. In *Proceedings of the Thirtieth International Joint Conference on Artificial Intelligence, IJCAI-21*, pages 4483–4491. International Joint Conferences on Artificial Intelligence Organization. Survey Track.

Patrick Lewis, Barlas Oğuz, Ruty Rinott, Sebastian Riedel, and Holger Schwenk. 2019. Mlqa: Evaluating cross-lingual extractive question answering. *arXiv preprint arXiv:1910.07475*.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. RoBERTa: A robustly optimized BERT pretraining approach. *arXiv preprint arXiv:1907.11692*.

Alexander Miller, Adam Fisch, Jesse Dodge, Amir-Hossein Karimi, Antoine Bordes, and Jason Weston. 2016. Key-value memory networks for directly reading documents. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 1400–1409, Austin, Texas. Association for Computational Linguistics.

Armin Oliya, Amir Saffari, Priyanka Sen, and Tom Ayoola. 2021. End-to-end entity resolution and question answering using differentiable knowledge graphs. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 4193–4200, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J. Liu. 2020. Exploring the limits of transfer learning with a unified text-to-text transformer. *Journal of Machine Learning Research*, 21(140):1–67.

Pranav Rajpurkar, Jian Zhang, Konstantin Lopyrev, and Percy Liang. 2016. Squad: 100,000+ questions for machine comprehension of text. *arXiv preprint arXiv:1606.05250*.

Adam Roberts, Colin Raffel, and Noam Shazeer. 2020. How much knowledge can you pack into the parameters of a language model? In *Empirical Methods in Natural Language Processing (EMNLP)*.

Apoorv Saxena, Aditay Tripathi, and Partha Talukdar. 2020. Improving multi-hop question answering over knowledge graphs using knowledge base embeddings. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 4498–4507, Online. Association for Computational Linguistics.

Priyanka Sen, Amir Saffari, and Armin Oliya. 2021. Expanding end-to-end question answering on differentiable knowledge graphs with intersection. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 8805–8812, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Jiaxin Shi, Shulin Cao, Lei Hou, Juanzi Li, and Hanwang Zhang. 2021. TransferNet: An effective and transparent framework for multi-hop question answering over relation graph. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 4149–4158, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Jiaxin Shi, Shulin Cao, Liangming Pan, Yutong Xiang, Lei Hou, Juanzi Li, Hanwang Zhang, and Bin He. 2020. Kqa pro: A large-scale dataset with interpretable programs and accurate sparqls for complex question answering over knowledge base. *arXiv preprint arXiv:2007.03875*.

Alon Talmor and Jonathan Berant. 2018. The web as a knowledge-base for answering complex questions. *arXiv preprint arXiv:1803.06643*.

Harsh Trivedi, Niranjan Balasubramanian, Tushar Khot, and Ashish Sabharwal. 2022. musique: Multihop questions via single-hop question composition. *Transactions of the Association for Computational Linguistics*, 10:539–554.

U.S. Census Bureau. 2021. U.S. Census Bureau Quickfacts: United States.

Ricardo Usbeck, Ria Gusmita, Muhammad Saleem, and Axel-Cyrille Ngonga Ngomo. 2018. 9th challenge on question answering over linked data (qald-9).

Linting Xue, Noah Constant, Adam Roberts, Mihir Kale, Rami Al-Rfou, Aditya Siddhant, Aditya Barua, and Colin Raffel. 2021. mT5: A massively multilingual pre-trained text-to-text transformer. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 483–498, Online. Association for Computational Linguistics.

Zhilin Yang, Peng Qi, Saizheng Zhang, Yoshua Bengio, William Cohen, Ruslan Salakhutdinov, and Christopher D. Manning. 2018. HotpotQA: A dataset for diverse, explainable multi-hop question answering. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 2369–2380, Brussels, Belgium. Association for Computational Linguistics.

Wen-tau Yih, Matthew Richardson, Christopher Meek, Ming-Wei Chang, and Jina Suh. 2016. The value of semantic parse labeling for knowledge base question answering. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 201–206.

Fengbin Zhu, Wenqiang Lei, Chao Wang, Jianming Zheng, Soujanya Poria, and Tat-Seng Chua. 2021. Retrieving and reading: A comprehensive survey on open-domain question answering. *arXiv preprint arXiv:2101.00774*.

A MTurk Tasks

Figures 2, 3, and 4 show the interfaces used by MTurk workers to complete each of the tasks to build the Mintaka dataset. All of these tasks were hosted on MTurk. Figure 2 is the Question Elicitation task. This example is for writing questions about the topic MOVIES and the complexity type COMPARATIVE. In each task, a worker would be shown examples and asked to write five questions.

Figure 3 is the Answer Entity Annotation task. In this example, a worker is shown the question-answer pair "Q: What Oscars did Argo win? A: Best Picture, Best Adapted Screenplay, Best Film Editing" and asked to identify the entities in the answer. The question is given as context, allowing the worker to know that these awards refer to Oscar awards. After highlighting each entity, the "Search Wikidata" button is automatically populated to create a search link on Wikidata for the given string that will open in a new window or tab. The workers could then look at all the choices and enter the URL of the correct entity.

Figure 4 is the Question Entity Annotation task. This example again shows the question "What Oscars did Argo win?" with "Oscars" highlighted. Workers were asked to focus on one entity at a time, so even though "Argo" is also a valid entity in this question, for this task, we are only interested in linking "Oscars". Workers would first verify that Oscars is a valid and complete entity, or modify the string if there was an error. Then, similar to the Answer Entity task, the "Search Wikidata" button lets worker search the string on Wikidata and find the URL of the correct entity.

B Model Training Details

Table 8 shows the train, dev, and test set sizes of each of the models. The Hits@1 Subset score is the score based on the test subset answerable by the model, and the Hits@1 Adjusted score is the adjusted hits@1 to account for the full test set. The full Mintaka dataset has 14,000 train examples, 2,000 dev examples, and 4,000 test examples.

For T5, fine-tuned T5 for CBQA, KVMemNet, and trained DPR we use the full train and dev sets for training and the full test set for evaluation. For MT5, we fine-tune on all languages simultaneously and report results overall and for each language individually. We found that fine-tuning MT5 on individual languages returned similar scores but at a higher computational cost. The zero-shot T5 CBQA and DPR models have no train or dev set since we evaluate them directly on the test set. The EmbedKGQA and Rigel models only predict over entities in a knowledge graph. This means that any sample that doesn't have at least one entity in the question and one entity in the answer is not used in the training or dev sets. This excludes samples where the answer is a number, a boolean (all yes/no questions), a date, or a string, or where an entity was found but no Wikidata link existed (for example, if the name of a video game character was identified, but no Wikidata ID existed). EmbedKGQA loses some additional examples if an entity was found in Wikidata but didn't have entity-to-entity facts, which are used to build the KG embeddings. For example, some entities in Wikidata only exist with labels (entity-to-string) facts but lack entity-to-entity facts.

C Mintaka Examples

Table 7 shows additional examples of question-answer pairs from the Mintaka dataset.

D Model Training Details

Table 9 shows the breakdown of performance by complexity type for all trained models. For count questions, we allow models to return the entities that are being counted rather than the number. For example, if the question is "*How many Academy Awards has Jake Gyllenhaal been nominated for?*", we allow the model to return "Academy Award for Best Supporting Actor" rather than "1". For entity answers, the order of answers does not matter, but for text answers, we use exact string matching.

The results show that both complex and generic questions remain challenging for models. For generic questions, this shows that even though we didn't specify a complexity type, these questions are not trivial. On complex questions, some of our models do perform better on comparative and yes or no questions. However for these questions, there is usually either a choice between two entities or a choice between "Yes" or "No", so randomly guessing would score 50%. This means that models scoring around 50% are not necessarily performing the reasoning required.

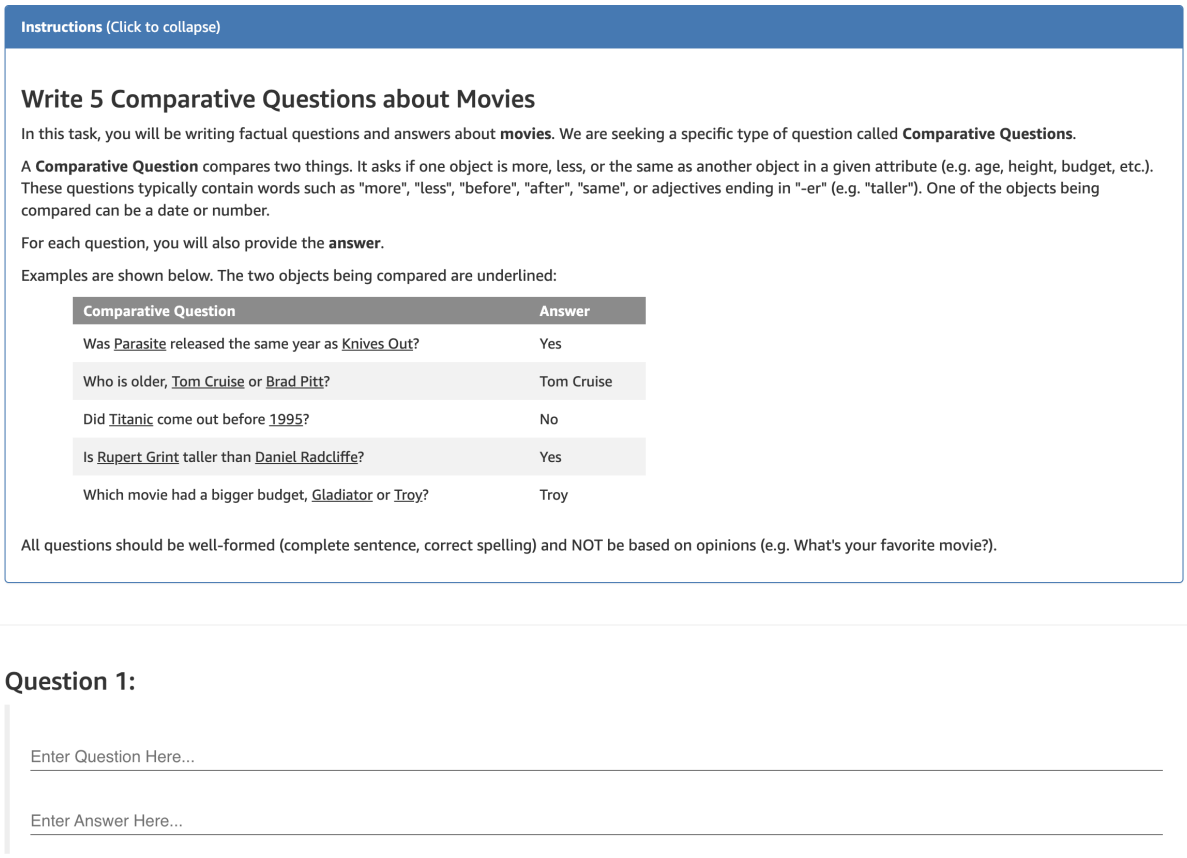

Figure 2: An example of the question elicitation MTurk task, where a worker is asked to write comparative questions about movies

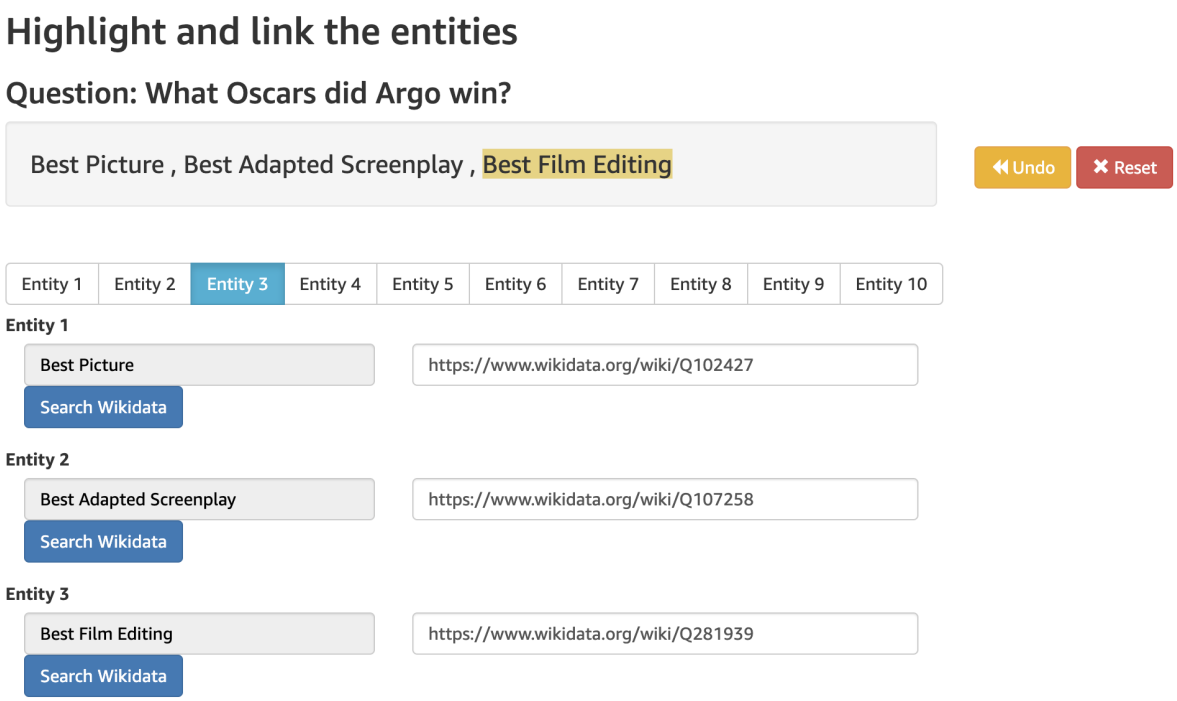

Figure 3: An example of the answer entity annotation MTurk task, where a worker is asked to identify and link the entities in the answer "Best Picture, Best Adapted Screenplay, Best Film Editing".

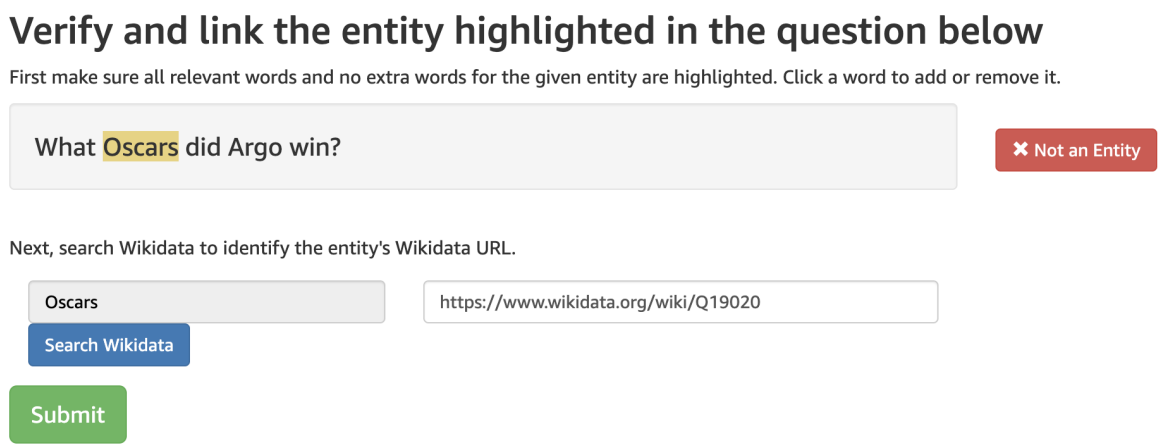

Figure 4: Example of the question entity annotation MTurk task, where the entity is already highlighted (in this case, "Oscars"), and the worker is asked to verify or modify the highlighted string and then link to Wikidata.

Q: Which series is older, Metroid (Q12397) or Super Mario Bros (Q23902998)? **A:** Super Mario Bros (Q23902998)
Q: What year was the first (number: 1) book of the A Song of Ice and Fire (Q45875) series published? **A:** 1996
Q: Which Amon Amarth (Q192863) albums did Fredrik Andersson (Q3752814) not perform as the drummer? **A:** Once Sent from the Golden Hall (Q1366410), Jomsviking (Q22674162), Berserker (Q62272261)
Q: Is the Eiffel Tower (Q243) located in Italy (Q38)? **A:** No
Q: Who was the president of Argentina (Q414) from 1989 (date: 1989) to 1999 (date: 1999)? **A:** Carlos Menem (Q185107)
Q: How many teams has Matthew Stafford (Q889130) played for? **A:** 2: Detroit Lions (Q271880) and Los Angeles Rams (Q337377)
Q: What is the name of the star of Iron Man's (Q192724) wife? **A:** Susan Downey (Q936542)
Q: What is the third (number: 3) longest river in the USA (Q30)? **A:** Yukon River (Q104437)
Q: Who ruled for a longer period of time, King Tut (Q12154) or Alexander the Great (Q8409)? **A:** Alexander the Great (Q8409)

Table 7: Example question-answer pairs from Mintaka. Question and answer annotations are shown here in-line with Wikidata Q-codes.

Model	Lang	Train	Dev	Test	Hits@1 Subset	Hits@1 Adjusted
T5	en	14,000	2,000	4,000	0.28	0.28
MT5	ar	126,000	18,000	36,000	0.15	0.15
	de	126,000	18,000	36,000	0.16	0.16
	es	126,000	18,000	36,000	0.16	0.16
	fr	126,000	18,000	36,000	0.16	0.16
	hi	126,000	18,000	36,000	0.16	0.16
	it	126,000	18,000	36,000	0.16	0.16
	ja	126,000	18,000	36,000	0.15	0.15
	pt	126,000	18,000	36,000	0.16	0.16
	multi	126,000	18,000	36,000	0.16	0.16
T5 CBQA (zero-shot)	en	–	–	4,000	0.20	0.20
T5 CBQA (fine-tuned)	en	14,000	2,000	4,000	0.38	0.38
T5 CBQA (translated)	ar	14,000	2,000	4,000	0.27	0.27
	de	14,000	2,000	4,000	0.34	0.34
	es	14,000	2,000	4,000	0.32	0.32
	fr	14,000	2,000	4,000	0.33	0.33
	hi	14,000	2,000	4,000	0.30	0.30
	it	14,000	2,000	4,000	0.32	0.32
	ja	14,000	2,000	4,000	0.28	0.28
	pt	14,000	2,000	4,000	0.31	0.31
	multi	14,000	2,000	4,000	0.31	0.31
KVMemNet	en	14,000	2,000	4,000	0.12	0.12
EmbedKGQA	en	9,837	1,409	2,809	0.26	0.18
Rigel	en	9,839	1,409	2,809	0.29	0.20
	ar	9,839	1,409	2,809	0.26	0.18
	de	9,839	1,409	2,809	0.27	0.19
	es	9,839	1,409	2,809	0.27	0.19
	fr	9,839	1,409	2,809	0.27	0.19
	hi	9,839	1,409	2,809	0.25	0.17
	it	9,839	1,409	2,809	0.27	0.19
	ja	9,839	1,409	2,809	0.28	0.20
	pt	9,839	1,409	2,809	0.25	0.18
	multi	88,551	12,681	25,281	0.27	0.19
DPR (zero-shot)	en	–	–	4,000	0.15	0.15
DPR (trained)	en	14,000	2,000	4,000	0.31	0.31

Table 8: Details of the train, dev, and test set sizes for all models. Hits@1 Subset shows the hits@1 score on the available test set. Hits@1 Adjusted adjusts the hits@1 score for the full test set of 4,000 questions.

Model	Lang	Gen	Mhop	Intsct	Diff	Comp	Superl	Ord	Count	YesNo
T5	en	0.24	0.14	0.32	0.13	0.55	0.31	0.14	0.05	0.67
MT5	multi	0.06	0.05	0.12	0.06	0.45	0.13	0.04	0.01	0.58
	ar	0.05	0.04	0.11	0.06	0.45	0.14	0.03	0.01	0.57
	de	0.07	0.05	0.13	0.07	0.42	0.14	0.05	0.01	0.57
	es	0.07	0.06	0.13	0.07	0.48	0.12	0.05	0.02	0.57
	fr	0.07	0.06	0.13	0.06	0.46	0.14	0.05	0.01	0.57
	hi	0.04	0.05	0.11	0.06	0.48	0.14	0.05	0.01	0.59
	it	0.07	0.06	0.12	0.07	0.45	0.14	0.05	0.01	0.58
	ja	0.05	0.04	0.10	0.04	0.45	0.13	0.03	0.01	0.59
	pt	0.07	0.05	0.14	0.07	0.45	0.13	0.05	0.01	0.59
T5 CBQA										
zero-shot	en	0.31	0.15	0.31	0.13	0.25	0.28	0.22	0.00	0.03
fine-tuned	en	0.41	0.21	0.44	0.21	0.58	0.42	0.30	0.09	0.71
translated	multi	0.32	0.15	0.38	0.16	0.51	0.36	0.26	0.07	0.56
	ar	0.23	0.12	0.31	0.14	0.49	0.31	0.20	0.05	0.58
	de	0.36	0.17	0.44	0.17	0.59	0.37	0.27	0.07	0.62
	es	0.35	0.15	0.40	0.19	0.54	0.38	0.27	0.08	0.55
	fr	0.35	0.17	0.42	0.17	0.52	0.38	0.28	0.07	0.60
	hi	0.30	0.14	0.34	0.16	0.49	0.34	0.25	0.05	0.65
	it	0.35	0.19	0.42	0.17	0.50	0.38	0.29	0.08	0.49
	ja	0.29	0.13	0.33	0.13	0.45	0.36	0.25	0.06	0.55
	pt	0.36	0.17	0.41	0.16	0.51	0.39	0.28	0.07	0.44
KVMemNet	en	0.04	0.03	0.06	0.04	0.29	0.12	0.06	0.01	0.52
EmbedKGQA	en	0.15	0.07	0.18	0.18	0.05	0.40	0.18	0.44	0.00
Rigel	en	0.19	0.09	0.30	0.13	0.43	0.13	0.09	0.50	0.00
	multi	0.17	0.09	0.25	0.12	0.45	0.11	0.08	0.44	0.00
	ar	0.17	0.09	0.25	0.12	0.39	0.09	0.08	0.45	0.00
	de	0.18	0.09	0.25	0.12	0.45	0.10	0.09	0.46	0.00
	es	0.18	0.09	0.26	0.12	0.47	0.11	0.08	0.47	0.00
	fr	0.18	0.10	0.27	0.13	0.43	0.11	0.08	0.45	0.00
	hi	0.15	0.08	0.24	0.11	0.42	0.08	0.08	0.42	0.00
	it	0.17	0.08	0.26	0.12	0.46	0.11	0.07	0.47	0.00
	ja	0.19	0.09	0.28	0.12	0.45	0.10	0.08	0.46	0.00
	pt	0.15	0.07	0.24	0.10	0.44	0.09	0.08	0.44	0.00
DPR										
zero-shot	en	0.27	0.10	0.20	0.10	0.12	0.23	0.21	0.03	0.00
trained	en	0.31	0.14	0.37	0.19	0.47	0.29	0.27	0.40	0.39

Table 9: A breakdown by complexity type for all trained baselines. Some complexity types are abbreviated: "Gen" is Generic questions, "Mhop" is multihop questions, "Intsct" is Intersection questions, "Diff" is Difference questions, "Comp" is Comparative questions, "Superl" is Superlative questions, and "Ord" is ordinal questions.

Can Edge Probing Tests Reveal Linguistic Knowledge in QA Models?

Sagnik Ray Choudhury[1*, 3], Nikita Bhutani[2], and Isabelle Augenstein[3]

[1]University of Michigan
[2]Megagon Labs
[3]University of Copenhagen
sagnikrayc@gmail.com, nikita@megagon.ai, augenstein@di.ku.dk

Abstract

There have been many efforts to try to understand what grammatical knowledge (e.g., ability to understand the part of speech of a token) is encoded in large pre-trained language models (LM). This is done through 'Edge Probing' (EP) tests: supervised classification tasks to predict the grammatical properties of a span (whether it has a particular part of speech) using *only* the token representations coming from the LM encoder. However, most NLP applications fine-tune these LM encoders for specific tasks. Here, we ask: if an LM is fine-tuned, does the encoding of linguistic information in it change, as measured by EP tests? Specifically, we focus on the task of Question Answering (QA) and conduct experiments on multiple datasets. We find that EP test results do not change significantly when the fine-tuned model performs well or in adversarial situations where the model is forced to learn wrong correlations. From a similar finding, some recent papers conclude that fine-tuning does not change linguistic knowledge in encoders but they do not provide an explanation. We find that EP models themselves are susceptible to exploiting spurious correlations in the EP datasets. When this dataset bias is corrected, we do see an improvement in the EP test results as expected.

1 Introduction

The encoding of linguistic information in large pre-trained language models (LMs) such as BERT (Devlin et al., 2019) has become an active research topic in recent times. This encoding is usually measured by edge probing (EP) tasks (Liu et al., 2019; Tenney et al., 2019a). Consider the sentence "the Met is closing soon". The token 'met' is a noun (a museum and not a form of the verb 'meet'). The context words ('the', 'is') are the only signals for determining its part of speech. If a 'simple' (one or two layer MLP (Hewitt and Liang, 2019)) classifier predicts 'met' as a noun *only* using the representation of the token 'met' (coming from a contextual encoder such as BERT (Devlin et al., 2019) or ELMo (Peters et al., 2018)) and not the whole sentence, then these signals must have been encoded in the token representation itself. This is the grammatical knowledge the test is 'probing' for. If encoder E_1 performs better than encoder E_2 on an EP test, say, part-of-speech tagging, we say that E_1 has a better knowledge of part-of-speech than E_2.

For many NLP tasks, pre-trained LMs (most commonly, BERT) have emerged as standard encoders (Raffel et al., 2020). These encoders are fine-tuned after adding task-specific layers on top. While probing tests on pre-trained encoders are quite popular, fine-tuned encoders are relatively under-explored (with notable exceptions of Merchant et al. (2020) and van Aken et al. (2019). We aim to bridge this gap by probing fine-tuned models using question answering (QA) as a target task. QA is a complex NLU problem requiring the model to implicitly perform many reasoning steps, and fine-tuned models provide strong baselines for various QA datasets (Devlin et al., 2019). Our first research question is thus:

RQ1: *Does fine-tuning for QA tasks improve the encoding of linguistic skills in the encoders, when measured by existing EP tests?* Intuitively, DNN based QA models would require implicit knowledge of semantic roles (who did what to whom, when, and where), an understanding of the part of speech and entity boundaries (most answers are entities in the context), and anaphora resolution (entities in the context would refer to each other). Indeed, prior works show how injecting knowledge about semantic roles (Shen and Lapata, 2007) and coreference resolution (González and Rodríguez, 2000) in classical QA pipelines improves their performance. Therefore, a fine-tuned QA model should *implicitly* acquire these linguistic skills. The

* Work done at the University of Copenhagen.

Proceedings of the 29th International Conference on Computational Linguistics, pages 1620–1635
October 12–17, 2022.

QA layers in the fine-tuned models have much fewer parameters than the encoders (§3), therefore, the encoders themselves are more likely to encode that grammatical knowledge. Consequently, when these fine-tuned encoders are used for the SRL (semantic role labeling), CoREF (coreference), PoS (part-of-speech tagging), and NER (named entity recognition) EP tests, we would expect to see improvements over pre-trained LMs. **But we do not observe any such change (§4)**.

Fine-tuning is generally performed on much less data than pre-training and the encoder weights might not change significantly. Could that cause the EP test results to remain the same? If the encoder weights are kept fixed during fine-tuning, the performance in the target (QA) task drops $50-70\%$ on all datasets. However, this frozen encoder has the same performance on the EP test as the original one. In other words, two encoders with a high difference in the target task performances have no discernible difference in the EP task performances. A possible explanation is that the QA models have no need to use the grammatical knowledge we are testing for. This motivates the second research question:

RQ2: *Does fine-tuning for QA tasks impart the linguistic skills necessary to perform QA in the encoders?* To answer this, we create a QA dataset that requires a particular 'skill' (Rogers et al., 2022; Ray Choudhury et al., 2022): the knowledge of coreference resolution (§5). Quoref (Dasigi et al., 2019) is such a dataset, but one might not require the knowledge of coreference to answer *all* questions in Quoref. Many instances in standard NLU tasks can be solved by heuristics, i.e., without proper *reasoning* (see Gururangan et al. (2018) for NLI or Min et al. (2019) for multi-hop QA). We design algorithms to filter out questions that can be answered heuristically (§5), and consequently, any model *probably* needs to use the knowledge of coreference to answer the rest. However, two encoders with a significant performance difference on this de-biased dataset have no difference in the CoREF EP test. This motivates the third research question:

RQ3: *Why do the EP test results not reflect that encoders have learned the linguistic skills needed to perform QA?*: Our analysis (§6) of the EP test datasets suggests that the EP models themselves might rely on dataset biases (as opposed to learning the task with input representations). When this bias is corrected, fine-tuned encoders behave as expected, i.e., show significant performance improvements over the base encoders. Previously, van Aken et al. (2019) and Merchant et al. (2020) observed that the EP test results do not differ in the base vs fine-tuned encoders[1] ($RQ1$) and concluded that the encoding of grammatical knowledge in the encoders does not change during fine-tuning. However, unlike ours, their studies were not done on the problems that explicitly call for such grammatical knowledge. Moreover, current criticisms of EP tests on non fine-tuned encoders focus on the task design itself (Hewitt and Liang, 2019; Voita and Titov, 2020) (see §6), whereas this work calls for bias correction in the standardized EP test datasets.

2 Related Work

Prior work has focused on understanding various aspects of pre-trained LMs including attention patterns (Clark et al., 2019) and linguistic knowledge (Liu et al., 2019). When these LMs are used as encoders in models, they turn out to be strong baselines for many tasks (Raffel et al., 2020). However, less is known about how the fine-tuning process changes the encoder's attention patterns (Kovaleva et al., 2019) or their encoding of linguistic knowledge. While many papers (Jia and Liang, 2017; Kaushik and Lipton, 2018; Sen and Saffari, 2020; Sugawara et al., 2020, inter alia) argue that DNN models often use heuristics to answer questions, Ray Choudhury et al. (2022) shows that at least some of the models use human-interpretable reasoning steps. Therefore, it is important to study how edge probing tests capture the task-specific reasoning abilities introduced in the fine-tuning process.

The paradigm of the classifier based probing tasks (of which our EP tasks are a subset) is quite mature (Ettinger et al., 2016), and has seen increasing popularity with the release of benchmark EP datasets (the ones we use here) (Tenney et al., 2019a). Typically, internal layers of large language or machine translation models are used as features for auxiliary prediction tasks for syntactic properties: part-of-speech (Shi et al., 2016; Blevins et al., 2018; Tenney et al., 2019a), tense (Shi et al., 2016; Tenney et al., 2019a), or subject-verb agreement

[1] Merchant et al. (2020) uses one QA dataset (SQuAD (Rajpurkar et al., 2016)) and all our EP tests; van Aken et al. (2019) uses two datasets (SQuAD and HotpotQA (Yang et al., 2018)) and two of our EP tests; which makes this study more rigorous in the QA domain.

(Tran et al., 2018; Linzen et al., 2016). See Belinkov and Glass (2019) for an extensive survey.

However, not many papers study the benchmark EP tests for fine-tuned representations. Most similar to our work is the layer-wise analysis of BERT weights for QA (van Aken et al., 2019) and the results of fine-tuning BERT on EP tasks (Merchant et al., 2020) – van Aken et al. (2019) use three QA datasets (SQuAD, HotpotQA, and bAbi (Weston et al., 2016)) to show that: 1) for the EP task of CoREF, test results remain unchanged, even when representations are taken from different layers; 2) different layers of a fine-tuned BERT can be attributed to different tasks in the QA process such as supporting fact extraction or entity selection. Merchant et al. (2020) studies MNLI, dependency parsing, and QA (SQuAD) to arrive at a similar finding, although their main results use a scalar mix of the weights from all layers of a fine-tuned BERT (whereas our work uses the top layer). RQ1 in our work can be considered complementary to their work, but RQ2 and RQ3 have not been studied before.

EP tests are indirect, i.e., a classifier (probe) is used to measure the linguistic information in the representation. Do the test results reflect the quality of the representations or the classifier's ability to learn the task (Hewitt and Liang, 2019; Voita and Titov, 2020)? We discuss this in §6. See Belinkov (2022) for more background on probing classifiers.

3 Edge Probing & QA: The Setup

Edge Probing: Following prior work (Merchant et al., 2020; van Aken et al., 2019), we use the model architecture and four of the edge probing tasks proposed by Tenney et al. (2019a). Given a sentence $S = [T_1, ...T_n]$ of n tokens where $T_i \in \mathcal{R}^d$, for PoS and NER tasks, the goal is to predict the part of speech or entity tag for a set of spans $T_i..T_j, 1 \leq i, j \leq n$ in that sentence (with *only* the span and not the sentence as the input). Using the same setting for SRL and CoREF tasks, the input is a pair of spans and the output is a class label: for SRL, it is a semantic role, for CoREF it is a binary label indicating whether one span is an antecedent of the other or not. A self-attention pooling operator is used to generate a fixed representation for spans of different lengths (Lee et al., 2017). For SRL and CoREF, these representations for the two spans are concatenated. A single-layer linear probe is used for the actual classification task (Figure 1).

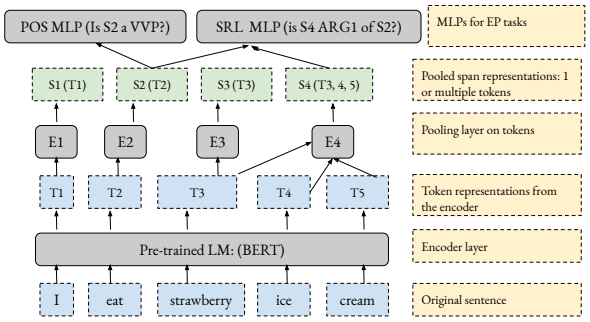

Figure 1: The architecture for edge probing tasks. For all tasks (PoS, SRL, CoREF, NER) the same MLP is used.

For EP tests, the span representations need to be generated from the token representations. Token representations can be generated from each layer (Tenney et al., 2019a) or the top layer (Tenney et al., 2019b) of the encoder. In each case, the layer i representation can be calculated as: 1. *Just* the output of layer i, 2. A concatenation of the first layer output and layer i output ('cat'), and 3. A scalar mixing of the output of $0 - i$ layers where the mixing weights are trainable parameters ('mix'). In all experiments, we use the 'cat' setting for the topmost layer in the encoder because both Merchant et al. (2020) and we find no significant difference for the other settings (the top layers generally perform better).

Question Answering: In typical QA setups, models are given a context and a question as input. We use two span-based datasets (SQuAD, Rajpurkar et al. (2016) and HotpotQA, Yang et al. (2018)) where the task is to extract a span of the text from the context as the answer. We also use two MCQ datasets (ReCoRD, Zhang et al. (2018) and MultiRC, Khashabi et al. (2018)) where the model is trained to select an answer from a set of choices. The architectures follow Devlin et al. (2019) and are similar for all datasets: one or two QA-specific layers on top of an encoder (see the appendix for more details).

4 RQ1: EP Tests & Fine-Tuned Encoders

We first run the EP tests with a standard encoder. Next, the QA models (the same encoder + QA layers) are trained (fine-tuned). Then, in the fine-tuned models the QA layers are replaced with the same MLP layers used in the EP tests, and the tests are repeated. This gives us a measure of

how much the encoding of linguistic knowledge in the encoder might have *improved* due to the fine-tuning process. We also randomize the QA training data (for SQuAD and HotpotQA) by using random noun phrases as answers. Thus the model is forced to learn wrong correlations, which can cause it to 'forget' the skills to some extent, and in turn, the encoder should perform worse on the EP tests.

4.1 Experimental Setup

We use a $\text{BERT}_{\text{base-uncased}}$ model as the base encoder for all tasks. For each QA task, we run five experiments. The span-based QA models are evaluated using the F1 score and the exact match (EM) metric. The exact match measures the percentage of answers that exactly match the actual answers. The F1 score measures the token overlap between the predicted and the actual answer. The MCQ questions are classification tasks and evaluated using the usual accuracy and Micro-F1 metrics for classification.

For EP tests we use the highest performing model in each QA dataset. The tests use the benchmark OntoNotes 5.0 corpus (Weischedel et al., 2013), as in Tenney et al. (2019a) and Merchant et al. (2020). We use the same hyper-parameters as the original paper on EP tests (Tenney et al., 2019b) except for the batch size (32 theirs vs 16 ours).[2] The QA models were trained for 10 epochs with a batch size of 16 using the Adam optimizer (Kingma and Ba, 2015). The EP models were trained for three epochs, using the same batch size and optimizer. The learning rates were kept at `1e-04` for the EP tasks and `1e-05` for the fine-tuning tasks. Further details about hyper-parameter searching and the exact configurations are provided in the appendix. Following the training regime of (Pruksachatkun et al., 2020), the models were evaluated on a subset of the validation data every 500 mini-batches with early stopping on 100 evaluations.

4.2 Results

We report the Micro-F1 scores for the EP tests on fine-tuned models in Table 1 (the average over 5 runs for each experiment, and the standard deviation varies between $0.1 - 1.5\%$). See the appendix for detailed results.

For SQuAD the test data is not publicly available, therefore, we report the results on dev data.

[2]Merchant et al. (2020) and van Aken et al. (2019) reportedly used the same HPs.

	SRL	CoREF	PoS	NER
BERT-base	81.1	81.2	96.1	93.0
Fine-tuning on original data				
SQuAD (81.9)	79.9	81.2	95.3	92.4
ReCoRD (57.0)	79.7	80.9	95.8	93.4
MultiRC (63.7)	80.7	82.3	95.8	93.5
HotpotQA (77.0)	77.7	80.2	94.3	90.9
Fine-tuning on randomized data				
SQuAD (7.4)	74.8	78.9	91.7	86.8
HotpotQA (12.5)	74.0	79.5	92.0	86.2

Table 1: Micro-F1 scores for different EP tasks: without fine-tuning, with fine-tuning on the original datasets, and with fine-tuning on randomized datasets. The F1 scores for the QA datasets are given in parentheses.

We changed HotpotQA instances to SQuAD style ones by providing the relevant sentences as the context, which is given for the train and dev data, but not for the test data. Therefore, we only report the results on the dev data.

For SQuAD and MultiRC, the results are somewhat lower than the best results reported in the literature with similar architectures (88.5 for Devlin et al. (2019) and 70.4 for Wang et al. (2019)).[3] For ReCoRD, the results are slightly better (Zhang, 2020). For HotpotQA, no fair evaluation is possible due to the data modifications.

Our EP test results for SRL and CoREF do differ from the previous work (Tenney et al., 2019a), but they are comparable with Liu et al. (2019), which uses the same dataset. However, we are more concerned with the fact that the EP test results do not change significantly when a fine-tuned vs the original encoder is used. In the randomization experiments, we see that the QA F1 score drops as expected, and the EP test results do change, but not as significantly as the QA F1 scores. This also indicates that improving the performance of the QA model itself might not change the EP test results significantly.

In summary, our experiments suggest that **fine-tuning indeed does not significantly change the encoding of linguistic knowledge in the underlying encoder, *when measured by the EP tests***, which is consistent with the findings of previous work (Merchant et al., 2020; van Aken et al., 2019), but provides complementary evidence.

[3]We use a max length of 128 in the encoder, whereas a max length of 512 produces comparable results. However, the target test results and EP test results are not correlated, therefore, we do not investigate this further.

5 RQ2: EP Test for Coreference

While we can expect the model to acquire some linguistic skills (the ability to do coreference resolution, identify the part of speech of a token) by learning to perform a QA task, there is no guarantee for this: a model can reason differently than we expect it to. For example, many HotpotQA questions can be answered by identifying the necessary entity type and not the multi-hop reasoning process that we expect (Min et al., 2019).

Therefore, in $RQ2$, we want to see whether the EP test results change when we know the encoder has to acquire particular grammatical knowledge K for the QA task. Consider two models M_1 (E_1 + QA_Layer_1) and M_2 (E_2 + QA_Layer_2) in our encoder + QA layer architecture. Assume we can identify a set of questions Q that can *only* be answered using K. If M_1 performs significantly better than M_2 in these questions, we can say that E_1 has encoded more information about K than the E_2 because the QA layers are unlikely to encode that knowledge as they have much less parameters than the encoders. Therefore, in the EP test for K, we can expect E_1 to perform better than E_2.

We define M_1 as a **fine-tuned-encoder**, where the full architecture (encoder (E_1) + QA layer) is trained; and M_2 as a **frozen encoder**: the encoder (E_2) is frozen and *only* the QA layer is trained. We choose K to be the grammatical knowledge of coreference. It is difficult to understand whether the knowledge of semantic roles or part of speech would be needed to answer a question, but it possibly can be done for coreference. For example, in Figure 2a, it is relatively easy to see that to answer the question a human needs to resolve the reference 'he' in the second sentence to 'Leo Strauss'.

5.1 Finding Coreference Questions

We employed four NLP practitioners to annotate 200 questions (100 each from SQuAD and HotpotQA). Each annotator was given a sample of 100 questions and was asked to stop as soon as they found 50 positive (questions they thought required coreference) instances. The question in Figure 2a is sampled from that dataset. But the question can also be answered by a shortcut (Geirhos et al., 2020). We know a 'where' question will only be answered by an entity of type location and there are two such entities in the context: Germany and United States. Germany is an argument to the trigger verb 'born', hence, is the answer.

The Quoref dataset (Dasigi et al., 2019) reportedly consists of questions that can only be answered by understanding the concept of coreference. The annotators *design* the questions themselves, which is different from our post-hoc annotation process. Figure 2b shows a sample question from the dataset. This question can not be answered without resolving the pronominal antecedent 'he' to 'Frankie'. But even Quoref *can* have questions that can be answered with a shortcut, therefore we develop algorithms to filter them out (§5.1.1, §5.1.2).

5.1.1 Model-Agnostic Filter

In Quoref, the answer is a span in the context. The Model-Agnostic Filter algorithm works in two steps: a) **Sentence Selection**: Select the context sentence that is most similar to the question; and b) **Entity Type Matching**: The question expects an entity of a particular type, eg: 'where' → location, 'who' → person. From the sentence selected in the last step, select an entity of the same type.

For 'Sentence Selection', we use two methods: 1) **Token-Overlap**: Select the sentence that has the highest token overlap with the question tokens, and 2) **Sentence Encoder**: An off-the-shelf sentence encoder from Reimers and Gurevych (2019) trained on MS Marco (Nguyen et al., 2016), a large scale dataset for answer passage retrieval (see the appendix for details).

For the 'Entity Type Matching' step, we design both supervised and unsupervised algorithms to determine the type of the answer entity from the question. For the unsupervised algorithm, we define a map (see the appendix) with the 'wh' words (who, when, whom) as the keys and the entity type as values (who ← `PER`). The first 'wh' word in the question determines the output. For example, for the question "where was Plato born, who wrote Republic?" it produces `LOC`. If no such word is found, it outputs an `UNK_ETYPE`.

This unsupervised approach will predict the wrong entity type for questions such as "Who won the World Cup in 2002?" (`PER` instead of `LOC`). Therefore, we train and evaluate supervised classification models on the training split of the Quoref dataset.[4] The label for a question is the entity type of the answer,[5] as detected by an off-the-shelf entity extractor from Spacy. If the answer is not a

[4]Further divided into 70(train)-20(dev)-10(test) splits

[5]One of the 18 types in Pradhan et al. (2013)

Context: Leo Strauss ...was a German-American political philosopher ... He was born in Germany... Thoughts on Machiavelli is a book by Leo Strauss ...
Question: Where was the author of Thoughts of Machiavelli born?
Answer: Germany

(a) A sample question from SQuAD.

Context: Frankie Bono, a mentally disturbed hitman from Cleveland,..Next, he goes to purchase a revolver from Big Ralph....
Question: What is the first name of the person who purchases a revolver?
Answer: Frankie

(b) A sample question from Quoref.

Figure 2: Sample questions from SQuAD and Quoref datasets. A reader relying on coreference resolution would take into account the green tokens.

Sentence	Etype		EM
Overlap	Supervised	fine-tuned (63)	6.31
		WordConv (58)	6.27
	Unsupervised		1.22
Encoder	Supervised	fine-tuned	5.99
		WordConv	5.48
	Unsupervised		0.97

Table 2: Different strategies for the Model-Agnostic Filter algorithm. EM stands for exact match, i.e., the percentage of cases where the filter produces the exact answer.

	frozen		fine-tuned	
	F1	EM	F1	EM
Quoref dev	10.23	5.41	69.53	65.61
- MAF	10.09	5.36	69.21	65.31
- MDF	7.00	3.19	38.57	30.85
- (MAF + MDF)	6.76	2.97	38.38	30.69
CoREF (Micro-F1)	81.68 $\pm$ 1.68		83.11 $\pm$ 0.7	

Table 3: Performance of frozen encoder and fine-tuned-encoder models when filters are applied: separately and in combination. MAF and MDF stands for model agnostic and dependent filters. The last row reports the Micro-F1 for both encoders in CoREF EP test.

named entity, or our entity extractor fails to determine its type, the label is `UNK_ETYPE`.[6] We experiment with two architectures: 1) a fine-tuned $\text{BERT}_{\text{base-cased}}$ model; and 2) a popular word convolutional model for sentence classification (Kim, 2014) using three parallel filters and 300 dimensional Google News Word2Vec representations (Mikolov et al., 2013). More details about the data, model architectures, and training is provided in the appendix.

5.1.2 Model-Dependent Filter

Following Sugawara et al. (2020) we replace all pronouns from the context in a question with random strings of the same length. If any one of M_1 or M_2 can still answer the question, it can *arguably* be answered without the knowledge of coreference.

5.1.3 Experiments & Results

The BERT and the WordConv supervised entity detectors have an average accuracy of $63.55 \pm 0.1\%$, and $58.81 \pm 0.3\%$ over 5 runs respectively.

The 'EM' column in Table 2 shows the proportion of Quoref questions (the validation split) that can be answered by the Model-Agnostic Filter algorithm. 'Overlap' and 'Encoder' are the two strategies for the 'Sentence Selection' step, and 'Supervised' and 'Unsupervised' are the same for the 'Entity Type Matching'.

The final Model-Agnostic Filter algorithm uses the token overlap approach to select a sentence from the context and uses the best fine-tuned BERT model to find the entity type for the answer. With this, we can filter out $6.3\%(155/2418)$ questions from the dev set. While this number is not very high, a similar exercise on SQuAD determines that at least 21% questions can be answered by this shortcut, which is consistent with prior findings (Jia and Liang, 2017).

In the Model-Dependent Filter algorithm, individually, the fine-tuned model filters out 55% dev instances, and the frozen encoder model filters out 6% of them, and in total, they filter out 56% (there is a large overlap).

5.2 Target Task vs Encoder EP Test

Table 3 shows the results of the fine-tuned and the frozen encoder on Quoref dev set before and after the filters are applied. As can be seen, the fine-tuned encoder performs much better than the frozen one across all scenarios. The performance of

[6] Indeed, a significant number of questions are labeled as such: 33%, 34%, and 35% in the train, dev, and test split of the data respectively.

these encoders on the CoREF EP test is given in the last column, which, however, does not differ much. This proves that **while one encoder might encode a linguistic knowledge (coreference) better than the other, the EP tests might fail to capture that**.

6 RQ3: An Analysis of EP Tests

6.1 Analysis

We see that the EP test results are surprisingly stable, even when we expect the fine-tuned encoder to learn or forget certain linguistic skills. EP tests are *indirect* measures of a representation's quality and have been criticized as such. Voita and Titov (2020) shows that for some EP tests, a large pre-trained LM (ElMo (Peters et al., 2018)) has the same performance as a random encoder. They conclude that the test measures the classifiers' ability to learn the EP task, and not the knowledge encoded in the representations itself. They propose using an information-theoretic (minimum description length or MDL) probe.

In a similar vein, Hewitt and Liang (2019) suggests designing a control task. A control task for an EP test is the same classification task, only the labels of the original task are changed so that it can not be recovered from the linguistic information. For example, two tokens with different part-of-speech tags will be mapped to the same arbitrary label. A classifier (probe) that performs well on both the control task and the original EP test must be learning the correlations in the data, and not using any information from the representations. Therefore, it can not measure the encoding of grammatical knowledge in the encoders.

If an EP test has only two labels, they will just be flipped in the control task. This makes the control task and the original EP test the same problem for the probe, and they must have the same performance. Therefore, for binary EP tests such as CoREF (the one we are interested in), no control task can be designed, but we use simple linear probes following Hewitt and Liang (2019)'s recommendations.

Had we used MDL probes instead, would our conclusions in $RQ1$ and $RQ2$ change? Prior work reports that the fine-tuned encoders do not show much difference in the MDL probes themselves (Merchant et al., 2020). Moreover, even in the original MDL probe paper, the CoREF test results are similar in a pre-trained vs random encoder (Voita and Titov, 2020). Therefore, replacing the current EP tests with MDL probes should not change the findings for the previous research questions.

	SRL	CoREF	PoS	NER
mem_uniform	32.46	65.02	88.62	71.59
mem_freq	44.45	78.06	88.69	73.27
same_prec_ante	-	70.23%	-	-
BERT-base	81.08	81.2	96.11	93.06

Table 4: A performance comparison on EP test results: Micro-F1 scores for heuristics and $\text{BERT}_{\text{base-uncased}}$ models (average over 5 runs, STD. varies from $0.09 - 1\%$).

Does this conclusively mean that the linguistic skill is not improved in the encoder, even when the task calls for it? Both Merchant et al. (2020) and van Aken et al. (2019) arrive at that conclusion, albeit without fine-tuning on a skill-specific target dataset such as Quoref. We present a dataset bias explanation (§6.2). Note the EP test dataset is used in both Merchant et al. (2020), van Aken et al. (2019), therefore the same explanation is valid for both these studies.

6.2 EP Test Heuristics

Following Gururangan et al. (2018), we design unsupervised algorithms that exploits spurious correlations in the dataset.

- **Memorization**: If a test data point is in the training data, the classifier returns the training data label. Else, it returns a random label either a) uniformly sampled ('mem_uniform') or b) sampled from the label probability distribution of the training data ('mem_freq').
- **Same Precedent-Antecedent**: In the CoREF dataset, whenever the precedent and antecedent are the same ("Obama is the president of the US. *He* lives in Washington D.C. *He* went to Harvard."), return positive.

6.2.1 Results

Table 4 shows the results for various heuristics and a $\text{BERT}_{\text{base-uncased}}$ encoder. To achieve moderately high performance on most EP tests, no specific representation is needed, let alone from a pre-trained or a fine-tuned one. The *Same Precedent-Antecedent* heuristic has a Micro-F1 score of 70.23% when used alone. This can be combined with mem_freq/ mem_uniform, but the combination provides no significant improvement. Overall, mem_freq is a strong baseline for the EP tests.

		Best	Worst
SRL	overall	-	-7.05
	easy	-	-4.40
	hard	-	-8.47
CoREF	overall	+1.93	-2.32
	easy	**+4.51**	**-7.11**
	hard	+0.47	-0.17
PoS	overall	-	-4.12
	easy	-	-4.01
	hard	-	**-12.30**
NER	overall	+0.35	-6.85
	easy	+0.25	-4.82
	hard	+0.62	**-12.48**
CoREF-LS	overall	+1.93	-2.32
	easy	-1.22	+1.1
	hard	**+13.27**	**-14.68**

Table 5: The accuracy changes across easy and hard instances for the best and worst fine-tuned models. For SRL and PoS, BERT-base is the best, therefore, there is no positive change. CoREF-LS refers to the case when the easy and hard points are created by splitting the data across labels. In some splits, the changes are significantly different than the overall change.

6.3 EP Tests: Hard & Easy Instances

The results in §6.2 can be viewed in another way. Many instances in the EP test data can be solved by memorization, i.e., are *easy* instances. But there are definitely difficult ones that account for the performance difference in the heuristics and the $\text{BERT}_{\text{base-uncased}}$ model. When the fine-tuned encoders show marginal improvements or deterioration (over the base encoders) do the performance in these EP tests increase/decrease uniformly across the hard/easy instances, or in the harder instances the results change more drastically? **In the second case, it can be argued that the EP tests do actually capture the change in the encoders, but the change is artificially clamped, which is an unfortunate side-effect of the dataset**.

We first divide the test data for an EP test into easy and hard instances: the ones that can be solved by mem_freq or not. Then we note the average accuracy of base $\text{BERT}_{\text{base-uncased}}$ encoders on these splits. Finally, we take two QA models (from two separate datasets) for which the encoders had the average best/worst results in the said EP test. Do the results change from the base encoder similarly across these splits?

6.3.1 Results

Table 5 shows no discerning pattern in the 'Best fine-tuned' column, probably because the overall improvements are indeed not significant. However, when an encoder model performs poorly (the 'Worst fine-tuned' column), it performs disproportionately badly on the hard instances. This proves that while on the surface it might appear the results are similar (Merchant et al., 2020; van Aken et al., 2019), they are indeed not.

We are however more interested in CoREF, because as discussed in $RQ2$, we have reasons to believe that some encoders have indeed learnt more about the skill of coreference than the others. However, contrary to the other results, it seems that the better/worse results come from the easy instances.

CoREF dataset has a significant label imbalance (compare row 1 and 2 in Table 4). A classifier predicting a negative label for all test instances can achieve an accuracy of 78.33%. If we split the data across the labels (CoREF-LS), we see that the results change drastically, with a clear indication that a better encoder gets better by classifying the hard (positive) instances better, and a worse encoder fails harder on the same instances. Unsurprisingly, the better encoders come from the encoders trained on the Quoref dataset.

In summary, we show *why* an EP model can fit well to the EP data without using a *good* representation. This indicates that while fine-tuning may improve the encoding of grammatical knowledge in encoders, the current EP tests (even the MDL probes) might not be able to capture it. **There are issues with the datasets rather than the task design itself**. This is a new explanation for the apparent consistency of EP test results in fine-tuned models, whereas previous work has mostly focused on classifier knowledge (see §6.1).

7 Conclusion and Future Work

Edge probing tests are the predominant method to probe for linguistic information in large language models. We use them to evaluate how the process of fine-tuning an LM for QA might change the grammatical knowledge in an encoder, and observe no significant differences between pre-trained and fine-tuned LMs. More importantly, we find this phenomenon in carefully designed target tasks where the models must use the said grammatical knowledge. From similar EP test results, previous works have concluded that fine-tuning does not

change the encoding of grammatical knowledge. However, our analysis provides a 'dataset bias' explanation for the consistency of the results and provides some clues as to why *any* representation tends to achieve very similar results for EP tests. This is different from the previous task-design criticisms of the EP tests.

Do fine-tuned NLU models score highly on benchmarks for the right reasons, i.e., follow the human reasoning process? This work shows some evidence in favor of that. The encoding of grammatical knowledge in QA encoders is improved as expected when the models are trained on the right datasets, and the dataset biases in the EP tests are corrected. In light of this evidence, in the future, we plan to identify the exact reasoning steps in the QA models through post-hoc explainability methods and study whether they align with the human reasoning steps.

8 Acknowledgement

We thank Behzad Golshan and Wang-Chiew Tan for the helpful discussions and annotation work. This project has been partly funded by a Megagon Labs research grant.

References

Yonatan Belinkov. 2022. Probing Classifiers: Promises, Shortcomings, and Advances. *Computational Linguistics*, 48(1):207–219.

Yonatan Belinkov and James R. Glass. 2019. Analysis Methods in Neural Language Processing: A Survey. *Trans. Assoc. Comput. Linguistics*, 7:49–72.

Terra Blevins, Omer Levy, and Luke Zettlemoyer. 2018. Deep RNNs Encode Soft Hierarchical Syntax. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics, ACL 2018, Melbourne, Australia, July 15-20, 2018, Volume 2: Short Papers*, pages 14–19. Association for Computational Linguistics.

Kevin Clark, Urvashi Khandelwal, Omer Levy, and Christopher D. Manning. 2019. What Does BERT Look at? An Analysis of BERT's Attention. In *Proceedings of the 2019 ACL Workshop BlackboxNLP: Analyzing and Interpreting Neural Networks for NLP, BlackboxNLP@ACL 2019, Florence, Italy, August 1, 2019*, pages 276–286. Association for Computational Linguistics.

Pradeep Dasigi, Nelson F. Liu, Ana Marasovic, Noah A. Smith, and Matt Gardner. 2019. Quoref: A Reading Comprehension Dataset with Questions Requiring Coreferential Reasoning. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing, EMNLP-IJCNLP 2019, Hong Kong, China, November 3-7, 2019*, pages 5924–5931. Association for Computational Linguistics.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT 2019, Minneapolis, MN, USA, June 2-7, 2019, Volume 1 (Long and Short Papers)*, pages 4171–4186. Association for Computational Linguistics.

Allyson Ettinger, Ahmed Elgohary, and Philip Resnik. 2016. Probing for semantic evidence of composition by means of simple classification tasks. In *Proceedings of the 1st Workshop on Evaluating Vector-Space Representations for NLP, RepEval@ACL 2016, Berlin, Germany, August 2016*, pages 134–139. Association for Computational Linguistics.

Robert Geirhos, Jörn-Henrik Jacobsen, Claudio Michaelis, Richard Zemel, Wieland Brendel, Matthias Bethge, and Felix A. Wichmann. 2020. Shortcut learning in deep neural networks. *Nature Machine Intelligence*, 2(11):665–673.

José Luis Vicedo González and Antonio Ferrández Rodríguez. 2000. Importance of Pronominal Anaphora Resolution in Question Answering Systems. In *38th Annual Meeting of the Association for Computational Linguistics, Hong Kong, China, October 1-8, 2000*, pages 555–562. ACL.

Suchin Gururangan, Swabha Swayamdipta, Omer Levy, Roy Schwartz, Samuel R. Bowman, and Noah A. Smith. 2018. Annotation Artifacts in Natural Language Inference Data. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT, New Orleans, Louisiana, USA, June 1-6, 2018, Volume 2 (Short Papers)*, pages 107–112. Association for Computational Linguistics.

Karl Moritz Hermann, Tomás Kociský, Edward Grefenstette, Lasse Espeholt, Will Kay, Mustafa Suleyman, and Phil Blunsom. 2015. Teaching Machines to Read and Comprehend. In *Advances in Neural Information Processing Systems 28: Annual Conference on Neural Information Processing Systems 2015, December 7-12, 2015, Montreal, Quebec, Canada*, pages 1693–1701.

John Hewitt and Percy Liang. 2019. Designing and Interpreting Probes with Control Tasks. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing,*

EMNLP-IJCNLP 2019, Hong Kong, China, November 3-7, 2019, pages 2733–2743. Association for Computational Linguistics.

Robin Jia and Percy Liang. 2017. Adversarial Examples for Evaluating Reading Comprehension Systems. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing, EMNLP 2017, Copenhagen, Denmark, September 9-11, 2017*, pages 2021–2031. Association for Computational Linguistics.

Divyansh Kaushik and Zachary C. Lipton. 2018. How much reading does reading comprehension require? A critical investigation of popular benchmarks. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing, Brussels, Belgium, October 31 - November 4, 2018*, pages 5010–5015. Association for Computational Linguistics.

Daniel Khashabi, Snigdha Chaturvedi, Michael Roth, Shyam Upadhyay, and Dan Roth. 2018. Looking Beyond the Surface: A Challenge Set for Reading Comprehension over Multiple Sentences. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 252–262, New Orleans, Louisiana. Association for Computational Linguistics.

Yoon Kim. 2014. Convolutional Neural Networks for Sentence Classification. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing, EMNLP 2014, October 25-29, 2014, Doha, Qatar, A meeting of SIGDAT, a Special Interest Group of the ACL*, pages 1746–1751. ACL.

Diederik P. Kingma and Jimmy Ba. 2015. Adam: A Method for Stochastic Optimization. In *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*.

Olga Kovaleva, Alexey Romanov, Anna Rogers, and Anna Rumshisky. 2019. Revealing the Dark Secrets of BERT. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 4365–4374, Hong Kong, China. Association for Computational Linguistics.

Kenton Lee, Luheng He, Mike Lewis, and Luke Zettlemoyer. 2017. End-to-end Neural Coreference Resolution. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing, EMNLP 2017, Copenhagen, Denmark, September 9-11, 2017*, pages 188–197. Association for Computational Linguistics.

Tal Linzen, Emmanuel Dupoux, and Yoav Goldberg. 2016. Assessing the ability of lstms to learn syntax-sensitive dependencies. *Trans. Assoc. Comput. Linguistics*, 4:521–535.

Nelson F. Liu, Matt Gardner, Yonatan Belinkov, Matthew E. Peters, and Noah A. Smith. 2019. Linguistic Knowledge and Transferability of Contextual Representations. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT 2019, Minneapolis, MN, USA, June 2-7, 2019, Volume 1 (Long and Short Papers)*, pages 1073–1094. Association for Computational Linguistics.

Amil Merchant, Elahe Rahimtoroghi, Ellie Pavlick, and Ian Tenney. 2020. What Happens To BERT Embeddings During Fine-tuning? In *Proceedings of the Third BlackboxNLP Workshop on Analyzing and Interpreting Neural Networks for NLP, BlackboxNLP@EMNLP 2020, Online, November 2020*, pages 33–44. Association for Computational Linguistics.

Tomás Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013. Efficient Estimation of Word Representations in Vector Space. In *1st International Conference on Learning Representations, ICLR 2013, Scottsdale, Arizona, USA, May 2-4, 2013, Workshop Track Proceedings*.

Sewon Min, Eric Wallace, Sameer Singh, Matt Gardner, Hannaneh Hajishirzi, and Luke Zettlemoyer. 2019. Compositional Questions Do Not Necessitate Multi-hop Reasoning. In *Proceedings of the 57th Conference of the Association for Computational Linguistics, ACL 2019, Florence, Italy, July 28- August 2, 2019, Volume 1: Long Papers*, pages 4249–4257. Association for Computational Linguistics.

Tri Nguyen, Mir Rosenberg, Xia Song, Jianfeng Gao, Saurabh Tiwary, Rangan Majumder, and Li Deng. 2016. MS MARCO: A Human Generated MAchine Reading COmprehension Dataset. In *Proceedings of the Workshop on Cognitive Computation: Integrating neural and symbolic approaches 2016 co-located with the 30th Annual Conference on Neural Information Processing Systems (NIPS 2016), Barcelona, Spain, December 9, 2016*, volume 1773 of *CEUR Workshop Proceedings*. CEUR-WS.org.

Matthew Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer. 2018. Deep Contextualized Word Representations. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 2227–2237, New Orleans, Louisiana. Association for Computational Linguistics.

Sameer Pradhan, Alessandro Moschitti, Nianwen Xue, Hwee Tou Ng, Anders Björkelund, Olga Uryupina, Yuchen Zhang, and Zhi Zhong. 2013. Towards Robust Linguistic Analysis using OntoNotes. In *Proceedings of the Seventeenth Conference on Computational Natural Language Learning, CoNLL 2013, Sofia, Bulgaria, August 8-9, 2013*, pages 143–152. ACL.

Yada Pruksachatkun, Phil Yeres, Haokun Liu, Jason Phang, Phu Mon Htut, Alex Wang, Ian Tenney, and Samuel R. Bowman. 2020. jiant: A Software Toolkit for Research on General-Purpose Text Understanding Models. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics: System Demonstrations*, pages 109–117, Online. Association for Computational Linguistics.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J. Liu. 2020. Exploring the Limits of Transfer Learning with a Unified Text-to-Text Transformer. *Journal of Machine Learning Research*, 21(140):1–67.

Pranav Rajpurkar, Jian Zhang, Konstantin Lopyrev, and Percy Liang. 2016. SQuAD: 100, 000+ Questions for Machine Comprehension of Text. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing, EMNLP 2016, Austin, Texas, USA, November 1-4, 2016*, pages 2383–2392. The Association for Computational Linguistics.

Sagnik Ray Choudhury, Anna Rogers, and Isabelle Augenstein. 2022. Machine Reading, Fast and Slow: When Do Models "Understand" Language? In *Proceedings of the 29th International Conference on Computational Linguistics*. International Committee on Computational Linguistics.

Nils Reimers and Iryna Gurevych. 2019. Sentence-BERT: Sentence embeddings using Siamese BERT-networks. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 3982–3992, Hong Kong, China. Association for Computational Linguistics.

Anna Rogers, Matt Gardner, and Isabelle Augenstein. 2022. QA Dataset Explosion: A Taxonomy of NLP Resources for Question Answering and Reading Comprehension. *Computing Surveys (CSUR)*, to appear.

Priyanka Sen and Amir Saffari. 2020. What do models learn from question answering datasets? In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing, EMNLP 2020, Online, November 16-20, 2020*, pages 2429–2438. Association for Computational Linguistics.

Dan Shen and Mirella Lapata. 2007. Using Semantic Roles to Improve Question Answering. In *EMNLP-CoNLL 2007, Proceedings of the 2007 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning, June 28-30, 2007, Prague, Czech Republic*, pages 12–21. ACL.

Xing Shi, Inkit Padhi, and Kevin Knight. 2016. Does String-Based Neural MT Learn Source Syntax? In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing, EMNLP 2016, Austin, Texas, USA, November 1-4, 2016*, pages 1526–1534. The Association for Computational Linguistics.

Saku Sugawara, Pontus Stenetorp, Kentaro Inui, and Akiko Aizawa. 2020. Assessing the Benchmarking Capacity of Machine Reading Comprehension Datasets. In *The Thirty-Fourth AAAI Conference on Artificial Intelligence, AAAI 2020, The Thirty-Second Innovative Applications of Artificial Intelligence Conference, IAAI 2020, The Tenth AAAI Symposium on Educational Advances in Artificial Intelligence, EAAI 2020, New York, NY, USA, February 7-12, 2020*, pages 8918–8927. AAAI Press.

Ian Tenney, Dipanjan Das, and Ellie Pavlick. 2019a. BERT Rediscovers the Classical NLP Pipeline. In *Proceedings of the 57th Conference of the Association for Computational Linguistics, ACL 2019, Florence, Italy, July 28- August 2, 2019, Volume 1: Long Papers*, pages 4593–4601. Association for Computational Linguistics.

Ian Tenney, Patrick Xia, Berlin Chen, Alex Wang, Adam Poliak, R. Thomas McCoy, Najoung Kim, Benjamin Van Durme, Samuel R. Bowman, Dipanjan Das, and Ellie Pavlick. 2019b. What do you learn from context? probing for sentence structure in contextualized word representations. In *7th International Conference on Learning Representations, ICLR 2019, New Orleans, LA, USA, May 6-9, 2019*. OpenReview.net.

Ke M. Tran, Arianna Bisazza, and Christof Monz. 2018. The importance of Being Recurrent for Modeling Hierarchical Structure. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing, Brussels, Belgium, October 31 - November 4, 2018*, pages 4731–4736. Association for Computational Linguistics.

Betty van Aken, Benjamin Winter, Alexander Löser, and Felix A. Gers. 2019. How Does BERT Answer Questions?: A Layer-Wise Analysis of Transformer Representations. In *Proceedings of the 28th ACM International Conference on Information and Knowledge Management, CIKM 2019, Beijing, China, November 3-7, 2019*, pages 1823–1832. ACM.

Elena Voita and Ivan Titov. 2020. Information-Theoretic Probing with Minimum Description Length. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing, EMNLP 2020, Online, November 16-20, 2020*, pages 183–196. Association for Computational Linguistics.

Alex Wang, Yada Pruksachatkun, Nikita Nangia, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel R. Bowman. 2019. SuperGLUE: A Stickier Benchmark for General-Purpose Language Understanding Systems. In *Advances in Neural Information Processing Systems 32: Annual Conference on Neural Information Processing Systems 2019, NeurIPS 2019, December 8-14, 2019, Vancouver, BC, Canada*, pages 3261–3275.

Ralph Weischedel, Martha Palmer, Mitchell Marcus, Eduard Hovy, Sameer Pradhan, Lance Ramshaw, Nianwen Xue, Ann Taylor, Jeff Kaufman, Michelle Franchini, et al. 2013. Ontonotes release 5.0 ldc2013t19. *Linguistic Data Consortium, Philadelphia, PA*, 23.

Jason Weston, Antoine Bordes, Sumit Chopra, and Tomas Mikolov. 2016. Towards AI-Complete Question Answering: A Set of Prerequisite Toy Tasks. In *4th International Conference on Learning Representations, ICLR 2016, San Juan, Puerto Rico, May 2-4, 2016, Conference Track Proceedings*.

Zhilin Yang, Peng Qi, Saizheng Zhang, Yoshua Bengio, William W. Cohen, Ruslan Salakhutdinov, and Christopher D. Manning. 2018. HotpotQA: A Dataset for Diverse, Explainable Multi-hop Question Answering. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing, Brussels, Belgium, October 31 - November 4, 2018*, pages 2369–2380. Association for Computational Linguistics.

Matthew D. Zeiler. 2012. ADADELTA: An Adaptive Learning Rate Method. *CoRR*, abs/1212.5701.

Sheng Zhang. 2020. BERT fine-tuning re-implementation for ReCoRD, JHU.

Sheng Zhang, Xiaodong Liu, Jingjing Liu, Jianfeng Gao, Kevin Duh, and Benjamin Van Durme. 2018. ReCoRD: Bridging the Gap between Human and Machine Commonsense Reading Comprehension. *CoRR*, abs/1503.06733.

A Appendix A: RQ1

A.1 QA Datasets and Model Architectures

SQuAD: We use the first (`V1.1`) version of the SQuAD dataset, which contains around 100K question, answer, context triples collected from the English Wikipedia. The answers are spans within the context. The questions and contexts are concatenated, and a linear layer on top of a contextual encoder is used to predict the probability of a context token i being the start ($P_{i,s}$) or end ($P_{i,e}$) of an answer. The score ($S_{i,j}$) for a span with start token i and end token j is $P_{i,s} + P_{j,e}$. For all valid combinations of i and j, the span with the highest score is chosen as the answer. A cross-entropy loss between the actual and predicted start/end positions is minimized.

HotpotQA: The HotpotQA dataset is a collection of 113K question-answer pairs, with two improvements over SQuAD: 1) Each context consists of multiple paragraphs (as opposed to a single one) and the model needs to reason over some of them to provide an answer and 2) The sentences required to answer a question are provided (as paragraph index, sentence index). The dataset features an implicit IR task (finding the relevant sentences), therefore we reduce it to a SQuAD style one by changing the context to the full paragraphs from where the supporting facts are chosen and removing the questions where the answer spans cannot be found in the context (3.8 % in train and 3.9 % in dev data).

ReCoRD: The ReCoRD dataset contains a set of 120,000 Close style questions from the CNN/Daily Mail dataset (Hermann et al., 2015). A Cloze style query is a statement with an occluded entity that is factually supported by a passage. The dataset provides the named entities in the context, one of which is the answer. For each (question, context, N named entity) triple in the data, N (question, context, label) triples are created, with the missing entity in the question replaced by one of the provided entities, and the label is put as true or false depending on whether the entity is the answer or not. This NLI style formulation reduces the QA task to a classification problem, for which a two-layer MLP is used on top of the encoder layer: a linear layer with a tanh activation, followed by another linear layer.

MultiRC: MultiRC is a multiple-choice QA dataset where the model has to choose one or multiple of provided answers utilizing the text from the question, context, and the answer itself. We reduce it to a binary classification task, where the input is a concatenation of the question, the context, and the answer (for each of the possible answers). The same architecture as ReCoRD is used.

A.2 HP Searching and Final Configurations for QA Datasets and EP Tests

Training Details: We searched for the following hyper-parameters (HPs): number of epochs and learning rates. Finally, the QA models were trained for 10 epochs with a batch size of 16 using the Adam optimizer (Kingma and Ba, 2015). The EP models were trained for the three epochs, using the same batch size and optimizer. The learning rates were kept at `1e-04` for the EP tasks and `1e-05` for the fine-tuning tasks. Following the training regime of (Pruksachatkun et al., 2020), the model was evaluated on a subset of the validation data every 500 mini-batches with early stopping on 100 evaluations. The config files in the provided code show the detailed HPs. For the EP tests, the hyperparameters are all same as the baseline (Tenney et al., 2019b) except for the batch size (32 theirs vs 16 ours). Merchant et al. (2020) and van Aken et al. (2019) reportedly used the same HPs.

A.3 Reproducibility Checklist

Description of computing infrastructure used: Titan RTX GPU, CUDA version 11.2.

The average runtime for each model or algorithm (e.g., training, inference, etc.), or estimated energy cost: 5-6 Hours for EP tests, 3-4 hours for QA models.

Number of parameters in each model: For QA models: BERT base uncased parameters + 128*num_classes (FC layer). For EP tests, they are the same.

Corresponding validation performance for each reported test result: For three QA datasets, we use the validation data itself. For the EP models, validation results are very similar to the test results reported.

Explanation of evaluation metrics used, with links to code: The evaluation metric for QA models is F1, as common in most QA datasets, including SQuAD. For EP tests, the evaluation metric is Micro-F1, which is used in Tenney et al. (2019b), the paper which is our baseline. The implementations are from Pruksachatkun et al. (2020), which hosts the original code used in Tenney et al. (2019b). We also use

For all experiments with hyperparameter search:

model	acc	micro_f1	macro_f1	weighted _f1
BERT-base	81.08 ± 1.33	81.08 ± 1.33	31.93 ± 1.34	80.09 ± 1.54
fine-tuned on original data				
SQuAD	79.97 ± 1.1	79.97 ± 1.1	30.57 ± 1.0	78.86 ± 1.3
ReCoRD	79.71 ± 1.46	79.71 ± 1.46	30.47 ± 1.61	78.61 ± 1.7
MultiRC	80.69 ± 1.25	80.69 ± 1.25	31.76 ± 1.37	79.68 ± 1.47
Hotpot	77.73 ± 1.2	77.73 ± 1.2	28.4 ± 1.23	76.45 ± 1.43
finetuned on randomized data				
SQuAD	74.79 ± 0.15	74.79 ± 0.15	26.07 ± 0.56	73.36 ± 0.17
Hotpot	74.03 ± 0.33	74.03 ± 0.33	25.79 ± 0.38	72.5 ± 0.35

Table 6: Results for SRL EP test.

The exact number of training and evaluation runs: For each QA model, 5 training/evaluation runs. For each EP test, 5 training/eval run.
Bounds for each hyperparameter: training batch size: 8-32, learning rate: for QA models, `1e-05` to `1e-03`, 3 steps. For EP tests, `1e-06` to `1e-04`, 5 steps.
Hyperparameter configurations for best-performing models: Provided as config files.
The method of choosing hyperparameter values (e.g., uniform sampling, manual tuning, etc.) and the criterion used to select among them (e.g., accuracy): Uniform sampling, F1 for QA models in dev data, Micro-F1 for EP tests in test data.

A.4 Results

Detailed results for the experiments in §4 are provided below in Tables 6, 7, 8, and 9.

B Appendix B: RQ2

B.1 Sentence Embedding Model for Answer Sentence Selection

MS MARCO (Nguyen et al., 2016) is a large dataset of question/ answer pairs. The dataset was built by first sampling queries from Bing search log and then using Bing to retrieve relevant documents and automatically extract relevant passages from those documents. Annotators were asked to mark relevant spans from the documents as answers.

We us a sentence embedding model (Reimers and Gurevych, 2019) built on the MS MARCO

model	acc	micro_f1	macro_f1	weighted _f1
BERT-base	81.18 ± 1.68	81.18 ± 1.68	59.33 ± 7.99	76.2 ± 3.89
fine-tuned on original data				
Quoref	83.11 ± 0.7	83.11 ± 0.7	67.97 ± 2.05	80.45 ± 1.09
SQuAD	81.19 ± 1.49	81.19 ± 1.49	60.27 ± 8.48	76.58 ± 4.01
ReCoRD	80.88 ± 1.76	80.88 ± 1.76	58.08 ± 9.79	75.57 ± 4.66
MultiRC	82.37 ± 1.71	82.37 ± 1.71	65.12 ± 8.7	78.99 ± 4.19
Hotpot	80.19 ± 1.84	80.19 ± 1.84	55.38 ± 8.86	74.21 ± 4.31
finetuned on randomized data				
SQuAD	78.86 ± 0.38	78.86 ± 0.38	48.77 ± 3.29	71.01 ± 1.5
Hotpot	79.45 ± 0.23	79.45 ± 0.23	52.1 ± 2.31	72.61 ± 1.03

Table 7: Results for CoREF EP test

model	acc	micro_f1	macro_f1	weighted _f1
BERT-base	96.11 ± 0.15	96.11 ± 0.15	87.88 ± 0.8	96.06 ± 0.15
fine-tuned on original data				
SQuAD	95.27 ± 0.04	95.27 ± 0.04	87.08 ± 0.45	95.2 ± 0.05
ReCoRD	95.77 ± 0.08	95.77 ± 0.08	86.91 ± 0.39	95.71 ± 0.09
MultiRC	95.77 ± 0.19	95.77 ± 0.19	86.51 ± 1.29	95.71 ± 0.19
Hotpot	94.31 ± 0.09	94.31 ± 0.09	83.45 ± 0.7	94.21 ± 0.1
fine-tuned on randomized data				
SQuAD	91.74 ± 0.35	91.74 ± 0.35	78.52 ± 1.15	91.54 ± 0.37
Hotpot	91.99 ± 0.13	91.99 ± 0.13	79.59 ± 0.65	91.8 ± 0.14

Table 8: Results for PoS EP test

model	acc	micro_f1	macro_f1	weighted _f1
BERT-base	93.0 ± 0.28	93.0 ± 0.28	78.12 ± 1.1	92.29 ± 0.37
fine-tuning on original data				
SQuAD	92.44 ± 0.09	92.44 ± 0.09	78.19 ± 0.47	91.86 ± 0.12
ReCoRD	93.35 ± 0.24	93.35 ± 0.24	79.88 ± 0.8	92.79 ± 0.31
MultiRC	93.5 ± 0.4	93.5 ± 0.4	80.51 ± 1.54	93.0 ± 0.52
Hotpot	90.9 ± 0.16	90.9 ± 0.16	73.81 ± 0.62	89.94 ± 0.21
fine-tuning on randomized data				
SQuAD	86.77 ± 1.02	86.77 ± 1.02	64.05 ± 3.64	85.25 ± 1.26
Hotpot	86.15 ± 0.32	86.15 ± 0.32	61.56 ± 1.42	84.48 ± 0.37

Table 9: Results for NER EP test

dataset. The pre-trained model is available off-the-shelf,[7] therefore can be used directly to find the most 'similar' sentence to the question. Among many sentence embedding models available for semantic search, we use this one because it is specifically trained on a question-passage retrieval dataset using a bi-encoder model. During training, the questions and the relevant/non-relevant passages are passed through a contextual encoder and their pooled representations are compared. The model is trained with the following objective: the (query, positive_passage) pair is supposed to be close in the vector space, while (query, negative_passage) should be distant.

B.2 Unsupervised Entity Type Selection

In unsupervised entity type selection method, we use a map to determine the entity type of the answer for a question. The map is given below:

```
{
'how far': [QUANTITY],
'how long': [DATE],
'how many': [CARDINAL],
'how old': [QUANTITY],
'what': [PRODUCT, WORK_OF_ART],
'when': [DATE, TIME],
'where': [FAC, LOC, ORG, GPE],
'who': [PERSON],
'whom': [PERSON],
'whose': [PERSON, ORG, NORP]
}
```

The entity types are defined in Pradhan et al. (2013). If the question has the phrase 'how far', the returned entity type is QUANTITY. The map is an `OrderedDict`, i.e., if the question is 'how old is the person who wrote Harry Potter', the returned entity type is QUANTITY. When there are multiple possibilities ('where'), one is returned randomly.

B.3 Supervised Entity Type Selection

B.3.1 Dataset

The dataset is created using the training set of Quoref which is divided into train/dev/test split for entity type detector model training and evaluation. A sample data point is shown in Figure 3.

The data distribution is shown in Figure 4. As can be seen, it is very skewed.

[7] https://www.sbert.net/docs/pretrained-models/msmarco-v5.html

Text: What is the full name of the person who is the television reporter that brings in a priest versed in Catholic exorcism rites?
Label: PER

Figure 3: A sample instance for answer entity type classifier.

	acc	macro_f1
$\text{BERT}_{\text{base-cased}}$	63.55 $\pm$ 0.00	19.65 $\pm$ 0.04
WordConv	58.81 $\pm$ 0.01	13.31 $\pm$ 0.02

Table 10: Results for supervised entity type selection

B.3.2 Models

We use two types of models: 1) a fine-tuned 12 layer 768 dimensional $\text{BERT}_{\text{base-cased}}$ model; and 2) a popular word convolutional model for sentence classification (Kim, 2014) using three parallel filters (size 3, 4, and 5) and 300 dimensional Google News Word2Vec representations (Mikolov et al., 2013).

BERT model: This model is trained for 5 epochs, with Adam optimizer (Kingma and Ba, 2015) with a weight decay of `1.0e-08` and a learning rate of `1.0e-05`. The sequence max length is kept at 128. We search for two hyperparameters: 1) number of epochs: 3-7, increasing by 1; and 2) learning rate: `1.0e-05`, `5.0e-05`, `1.0e-04`. Accuracy is used as the early stopping metric.

WordConv model: This model is trained for 40 epochs, with Adadelta optimizer (Zeiler, 2012) with a learning rate of `1.0e-05`. The sequence max length is again kept at 128. Accuracy is used as the early stopping metric.

B.4 Results

The results are provided in Table 10.

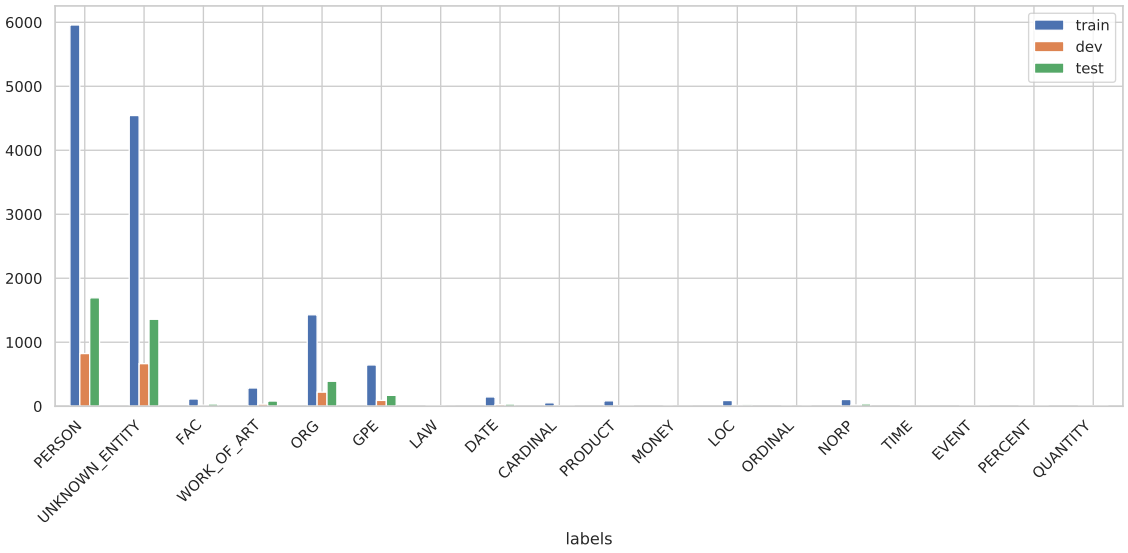

Figure 4: Label distribution for supervised entity type.

Conversational QA Dataset Generation with Answer Revision

Seonjeong Hwang
Graduate School of Artificial Intelligence
POSTECH, Pohang, South Korea
seonjeongh@postech.ac.kr

Gary Geunbae Lee*
Computer Science and Engineering
Graduate School of Artificial Intelligence
POSTECH, Pohang, South Korea
gblee@postech.ac.kr

Abstract

Conversational question–answer generation is a task that automatically generates a large-scale conversational question answering dataset based on input passages. In this paper, we introduce a novel framework that extracts question-worthy phrases from a passage and then generates corresponding questions considering previous conversations. In particular, our framework revises the extracted answers after generating questions so that answers exactly match paired questions. Experimental results show that our simple answer revision approach leads to significant improvement in the quality of synthetic data. Moreover, we prove that our framework can be effectively utilized for domain adaptation of conversational question answering.

1 Introduction

Conversational question answering (CQA) involves answering questions by considering a given text as well as previous conversations. To facilitate research on CQA, a range of datasets have been proposed in recent years (Choi et al., 2018; Reddy et al., 2019; Campos et al., 2020; Anantha et al., 2020; Adlakha et al., 2022). However, building a robust CQA system for a specific domain requires a large-scale domain-specific dataset; moreover, obtaining such a dataset is considerably expensive and time-consuming.

To resolve this issue, in our previous study, we had proposed a conversational question–answer generation (CQAG) framework that automatically creates multiturn question–answer (Q–A) pairs from given passages (Hwang and Lee, 2021). The framework is a two-stage architecture that adopts contextual answer extraction (CAE) and conversational question generation (CQG). Considering previous conversations, the CAE module extracts the next question-worthy phrase from the passage, and then the CQG module generates the conversational question corresponding to the phrase. However, the framework has the limitation that the error may propagate to the question generation stage and even to data generation for subsequent turns if improper answers are extracted by the CAE module.

In this paper, we introduce a CQAG framework with answer revision (CQAG-AR), in which the conversational question generation with answer revision (CQG-AR) module revises the extracted answer to a more suitable one immediately after generating a question. In experiments, we synthesize CQA data using CQAG-AR and then evaluate CQA models trained on these synthetic data. Results reveal that answer revision by the CQG-AR module leads to absolute improvements of up to 13.4% and 15.3% in F1 score and exact match (EM), respectively, for the CQA models. Furthermore, fine-tuning the Wikipedia-domain CQA model on different synthetic data increases EM by up to 13.1%, showing that our framework is beneficial for CQA domain adaptation.

2 Related Work

CQG aims to create conversational questions based on input text. It can be subdivided into answer-aware and answer-unaware approaches. Answer-aware CQG generates conversational questions corresponding to prepared answers (Gao et al., 2019). Gu et al. (2021) exploited accumulated representations of previous conversations to generate the current question by successively encoding answers and questions that constitute conversation history. By contrast, answer-unaware CQG synthesizes conversational questions without given answers (Wang et al., 2018; Pan et al., 2019; Qi et al., 2020). Further, Nakanishi et al. (2019) introduced a framework that first finds the location of points of interest in the passage, identifies question types, and subsequently generates conversational questions.

CQAG attempts to automatically construct CQA data for various domains. In our previous study

*Corresponding author

Proceedings of the 29th International Conference on Computational Linguistics, pages 1636–1644
October 12–17, 2022.

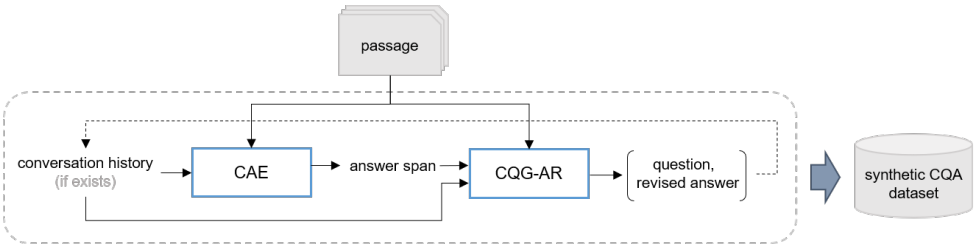

Figure 1: Overview of CQAG-AR. Synthetic Q–A pairs are used as conversation history to generate the subsequent Q–A pairs (dotted line).

(Hwang and Lee, 2021), we designed a vanilla CQAG that generated multiturn Q–A pairs based on a given passage in an autoregressive manner. However, the framework has a drawback in that the validity of the extracted answer directly affects the quality of the conversation.

3 Methods

Figure 1 illustrates a CQAG-AR generation pipeline. To generate a question q_t and answer a_t for the t-th turn of conversations, our framework obtains a passage p and conversation history $h_t = ((q_1, a_1), ..., (q_{t-1}, a_{t-1}))$ as inputs. The CAE module extracts a probable answer span a_t^s considering these inputs. Next, the CQG-AR module generates a conversational question q_t and revised answer a_t^r given the inputs and the extracted answer span a_t^s. Finally, we use the revised answer a_t^r as the answer a_t. The modules do not employ the conversation history to synthesize the Q–A pair for the first turn of conversations. t is omitted in all notations in the following description.

3.1 Contextual Answer Extraction

From a given passage p, the CAE module extracts an answer span a^s that is most likely to be of interest to a questioner, considering the conversation history h, i.e., $P(a^s \mid p, h)$. We implemented the module using a pretrained BERT (Devlin et al., 2018) with two fully connected (FC) layers at the top (Hwang and Lee, 2021). Each FC layer predicts the index of start and end tokens of the potential answer span in the passage:

$$prob_i^s = \mathrm{Softmax}(\mathrm{FC}^s(\mathrm{BERT}(p, h)))[i],$$
$$prob_j^e = \mathrm{Softmax}(\mathrm{FC}^e(\mathrm{BERT}(p, h)))[j],$$

where $prob_i^s$ ($prob_j^e$) represents the probability for the i-th (j-th) token in the passage being the start (end) token of the answer span.

During generation, the top k answer candidates whose start and end indices are i and j are extracted according to the sum of probabilities $prob_i^s + prob_j^e$. The CAE module outputs the answer span with the highest sum of probabilities after deduplicating the candidates compared with the answers used in the conversation history. If the candidate set is empty after deduplication, generation is terminated. To train the module, we computed the sum of cross-entropy losses between predicted start and end indices and the ground truth indices.

3.2 Conversational Question Generation with Answer Revision

Considering the input passage and conversation history, the CQG-AR module generates a conversational question and then revises the answer span that is extracted by the CAE module. The module first considers that the extracted answer span is proper for use as an answer and modifies it if it is inappropriate. To enable this process, we collected training examples of passage p, conversation history h, answer span a^s, and revised answer a^r, which contained positive (*proper* a^s, a^r) and negative (*improper* a^s, a^r) pairs. The module can preserve proper answers extracted by the CAE module by learning positive examples. Additionally, negative examples teach the module how to correct improper answer spans with better answers to the generated questions. We devised two negative sampling techniques to collect improper answer spans from proper ones.

3.2.1 Generating Negative Samples

For the positive examples, we set ground truth answers of the CQA dataset (e.g., QuAC (Choi et al., 2018)) to both *proper* a^s and a^r. The main experiments were conducted with CoQA (Reddy et al., 2019), which contains free-form answers paired with rationales extracted from passages. To obtain proper answer spans from CoQA, answer spans with the highest F1 score compared to the free-form answer from the rationale were extracted. The

improper a^s was obtained from the *proper* a^s by using the following techniques.

Answer Span Expansion If the extracted answer contains several key phrases, it may be unsuitable as an answer for a single question. In addition, unnecessary words around the answer span should be discarded if they are extracted together. To cover these cases, we generated the *improper* a^s by additionally connecting surrounding words of random length to the front or the rear of the *proper* a^s. However, if the sample was extended to phrases that were answers of other Q–A pairs, the model could confuse the target a^r. Therefore, we ensured that the sample did not overlap with answers for other turns.

Answer Span Reduction Some important words that constitute a complete answer may be omitted when extracting the answer span. This phenomenon may risk creating invalid Q–A pairs. To create these types of *improper* a^s, we removed a random number of words from both ends of the *proper* a^s. Examples of negative sampling are included in Appendix A.

3.2.2 Modeling

When the passage p, conversation history h, and answer span a^s are given, the CQG-AR module sequentially generates the conversational question q for the input answer span a^s and then revises the answer span a^s based on the generated question q:

$$P(q, a^r|p, h, a^s) = \prod_{i=1}^{<q>} P(q_i|p, h, a^s, q_{1:i-1}) \times \prod_{j=1}^{<a^r>} P(a^r_j|p, h, a^s, q, a^r_{1:j-1}),$$

where a^r denotes the revised answer and $< \cdot >$ indicates the length of the element.

We implemented the CQG-AR module using T5 (Raffel et al., 2019). We focused on the masked self-attention mechanism of Transformer (Vaswani et al., 2017), where the decoder utilizes knowledge of previously generated tokens to predict the current token. To revise the answer in a form that is more natural to the question, the module outputs the modified answer immediately after the question is generated. The answer span was highlighted using a special token so that the content of interest could be effectively recognized in question generation and answer revision. To train our module, we computed the cross-entropy loss between the question and answer of the ground truth and the module's prediction.

4 Experiments

4.1 Experimental Setup

We employed CoQA (Reddy et al., 2019), which comprised 8k passages collected from seven different domains and human-annotated conversations. To investigate whether CQAG-AR could be effectively utilized to construct a CQA system in a new domain, we split the data into *in-domain* (Wikipedia) and *out-of-domain* (children's stories, literature, news, and middle and high school English exams). The *in-domain* data were used to train CQAG frameworks, and the quality of synthetic data generated by the trained CQAG frameworks was evaluated using the *out-of-domain* data.

In addition, we used QuAC (Choi et al., 2018) and DoQA (Campos et al., 2020) to evaluate our framework. QuAC is based on 13k Wikipedia passages, and DoQA comprises passages collected from FQAs of three practical domains. Because the other two domains constituted only the test set, we used only the *cooking* domain of DoQA in our experiment. The CQAG frameworks used in experiments could generate only open-ended types of data. Therefore, closed-ended (yes and no) and unanswerable types of examples were excluded from the datasets; these were denoted by CoQA^D, QuAC^D, and DoQA^D.

4.2 Baseline Frameworks

To evaluate the quality of the synthetic data generated by our framework, we used two baseline CQAG frameworks:

CQAG-Chain ChainCQG[1] (Gu et al., 2021) is a state-of-the-art answer-aware CQG model, and CQAG-Chain combines the CAE module of CQAG-AR and ChainCQG as a CQG module.

Vanilla CQAG (Hwang and Lee, 2021) This framework shares the same CAE module with CQAG-AR but adopts a simple T5-based CQG module. The CQG module accepts the same input elements as CQG-AR but generates only conversational questions.

[1]The original ChainCQG accepted a rationale-highlighted passage as an input element but we highlighted an answer span in the passage in our experiment. In addition, we implemented the model based on the original source code: https://github.com/searchableai/ChainCQG.

Training data		In-domain	Out-of-domain			
		Wikipedia	News	Mid/High Sch.	Literature	Children's Sto.
Synthetic	CQAG-Chain	71.3 / 59.6	69.2 / 56.9	64.1 / 51.4	59.4 / 47.6	63.1 / 47.6
	Vanilla CQAG	71.3 / 59.9	67.6 / 55.7	65.8 / 52.7	60.3 / 48.3	66.6 / 50.5
	CQAG-AR (ours)	**83.1 / 73.8**	**81.0 / 71.0**	**74.4 / 63.2**	**71.7 / 61.0**	**75.2 / 61.9**
Human-annotated		85.8 / 76.4	86.3 / 75.9	79.0 / 67.6	79.0 / 67.8	82.5 / 70.1

Table 1: F1 (%) and EM (%) scores of CQA models on the CoQA^D test set for each domain (The highest performances among results for synthetic data are shown in bold.)

4.3 CQA with Synthetic Datasets

In this section, we evaluate synthetic data generated by CQAG-AR and baseline frameworks by conducting the CQA task. In the first experiment, CQAG frameworks learned the *in-domain* data of CoQA^D and then generated synthetic CQA data based on the passages extracted from *in-domain* and *out-of-domain* data. Note that we constructed synthetic training and validation sets from original splits of CoQA. We implemented a simple CQA model using T5 (Raffel et al., 2019) and trained several CQA models using human-annotated data (CoQA^D) and synthetic datasets. The training details, an example of synthetic conversations, and statistics of the synthetic data can be found in Appendix B and C.

Table 1 presents F1 and EM scores of CQA models, which learned the synthetic data, on the test set[2] of CoQA^D. The CQA models derived from CQAG-AR outperformed other models, showing significant margins of 11.8% and 13.9% for in-domain data, and average margins of 10.5% and 12.5% for out-of-domain data in F1 and EM, respectively, compared with those derived from the vanilla CQAG. These results demonstrate that the answer revision approach is considerably beneficial in terms of generating valid CQA datasets. However, we additionally found that the out-of-domain CQA models showed lower performances than the in-domain models across all CQAG frameworks. This result implies that CQAG frameworks are less robust when extracting valid CQA data from out-of-domain passages.

Training data	#Training examples	F1
Human-annotated	3.7k	45.1
Synthetic	3.7k	51.5
(CQAG-AR)	4.7k	53.1

Table 2: CQA performances on the test set of DoQA^D.

In Table 2, we compare the evaluation results on the test set of DoQA^D (cooking) for CQA models trained on human-annotated data (DoQA^D) and synthetic data. We obtained the synthetic data by training CQAG-AR using QuAC^D and then generating the data from the passages of DoQA training and validation sets. According to the results, the CQA model trained on synthetic data, which has the same number of examples with the human-annotated data, significantly outperformed the model trained on human-annotated data. Moreover, our framework generated a larger number of examples than the ones in the original DoQA^D, which improved the F1 score of the CQA model by 1.6%. In particular, examples in QuAC^D, which were used to train CQAG-AR, were irrelevant to the cooking domain. This result indicates that our framework effectively creates synthetic CQA data for an unseen cooking domain.

4.4 Human Study

In Table 3, we classify the synthetic examples according to revision types. The distribution shows that 65.2% of answer spans extracted by the CAE module were preserved without any modification, while the other 34.8% of answers were revised. This demonstrates that the CQG-AR module could recognize invalid answer spans and selectively modify the answers. Furthermore, we found that the module could perform more complex revisions such as "multiple revision" and "complete change" though the CQG-AR module learned only examples for "reduction" and "expansion" obtained by negative sampling.

Further, we conducted human evaluation to compare the quality of synthetic data generated by the vanilla CQAG and our CQAG-AR. From the two synthetic datasets presented in Table 1, 120 examples (30 examples from each out-of-domain dataset) were sampled and three volunteers were asked to rate 80 examples according to the criteria listed in Appendix D.

[2] https://github.com/google/BIG-bench

Revision type	Passage	Q-A	Percentage
Preservation	... It covers and has a population of 2.06 million. It is a parliamentary republic and a member of the United Nations, European Union, and NATO. ...	Q: How many people live in Slovenia? A: 2.06 million	65.2%
Reduction	Buckinghamshire (or), abbreviated Bucks, is a county in South East England which borders Greater London to the south east, Berkshire to the ...	Q: Where is it located? A: South East England	15.3%
Expansion	... The group hired Frederick G. Kilgour, a former Yale University medical school librarian, to design the shared cataloging system. ...	Q: Who was he? A: a former Yale University medical school librarian	14.2%
Multiple revision	... Discogs currently contains over 8 million releases, by nearly 4.9 million artists, across over 1 million labels, contributed from nearly 346,000 contributor ...	Q: And how many labels? A: over 1 million	2.0%
Complete change	... Selective breeding for fast growth, egg-laying ability, conformation, plumage and docility took place over the centuries, and modern breeds ...	Q: How did breeds change over time? A: selective breeding	3.4%

Table 3: Distribution of the answer-revision types in the CQG-AR module. (The answer spans extracted by the CAE module are highlighted.)

		Vanilla CQAG	CQAG-AR
Question Connectivity	Dependent	67.9%	66.7%
	Independent	27.7%	30.6%
	Unnatural	4.5%	2.8%
Answer Correctness	Correct	64.2%	85.8%
	Partially correct	23.3%	4.2%
	Incorrect	12.5%	10.0%

Table 4: Results of human evaluation for synthetic data generated by vanilla CQAG and CQAG-AR.

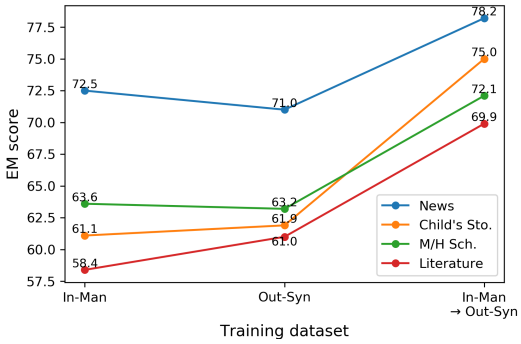

Figure 2: EM scores of several CQA models on the CoQA^D test set for each domain.

Although 2.9% more synthetic questions of CQAG-AR are independent of the previous conversations than the questions of the vanilla CQAG in Table 4, the synthetic questions of the two frameworks show almost similar evaluation results. These results prove that the vanilla CQAG, which performs only question generation, and CQAG-AR, which performs question generation and answer revision in an end-to-end manner, can generate questions of similar quality. However, 21.6% more synthetic answers of CQAG-AR were judged as correct answers compared with those of the vanilla CQAG. Furthermore, the number of partially correct answers was considerably reduced through answer revision. This reveals that answer revision is effective in correcting inappropriate answer spans extracted by the CAE module into correct answers that match well with the question.

4.5 Domain Adaptation

In this experiment, we tested the effectiveness of the synthetic data generated by CQAG-AR in adapting the CQA model from the Wikipedia domain to new domains (*out-of-domain*). We trained CQA models, which were initialized with parameters of T5-Large, in three training settings. In the **In-Man** setting, we trained the CQA model on the Wikipedia data of CoQA^D. In the **Out-Syn** setting, CQA models learned out-of-domain synthetic data. Finally, the model of In-Man setting was fine-tuned with synthetic data of each out-of-domain case in the **In-Man** $\rightarrow$ **Out-Syn** setting.

As shown in Figure 2, the models in the Out-Syn setting yielded results similar to those of the model in the In-Man setting while exhibiting better EM scores in the two domains. Notably, fine-tuning the Wikipedia CQA model using our synthetic data (In-Man $\rightarrow$ Out-Syn) improved the EM scores of the model by an average of 9.7% across all domains. This result indicates that our framework can be effectively utilized for domain adaptation in CQA.

5 Conclusion

In this paper, we propose CQAG-AR, which automatically synthesizes high-quality CQA data. Our framework comprises CAE and CQG-AR modules. Considering the conversation history, the CAE module extracts a question-worthy phrase from a given passage, and then the CQG-AR module generates a conversational question while revising the extracted answer to make it more suitable. Experimental results show that CQAG-AR outperforms baseline frameworks in terms of generating high-quality CQA data. In addition, fine-tuning a Wikipedia-domain CQA system on our synthetic data for out-of-domain cases improves the model performances by significant margins.

Acknowledgements

We would like to thank Professor Yunsu Kim (POSTECH) for his expert advice. This work was supported by SAMSUNG Research, Samsung Electronics Co.,Ltd., and also supported by the MSIT(Ministry of Science and ICT), Korea, under the ITRC(Information Technology Research Center) support program(IITP-2022-2020-0-01789) supervised by the IITP(Institute for Information Communications Technology Planning Evaluation)

References

Vaibhav Adlakha, Shehzaad Dhuliawala, Kaheer Suleman, Harm de Vries, and Siva Reddy. 2022. Topiocqa: Open-domain conversational question answering with topic switching. *Transactions of the Association for Computational Linguistics*, 10:468–483.

Raviteja Anantha, Svitlana Vakulenko, Zhucheng Tu, Shayne Longpre, Stephen Pulman, and Srinivas Chappidi. 2020. Open-domain question answering goes conversational via question rewriting. *arXiv preprint arXiv:2010.04898*.

Satanjeev Banerjee and Alon Lavie. 2005. Meteor: An automatic metric for mt evaluation with improved correlation with human judgments. In *Proceedings of the acl workshop on intrinsic and extrinsic evaluation measures for machine translation and/or summarization*, pages 65–72.

Jon Ander Campos, Arantxa Otegi, Aitor Soroa, Jan Deriu, Mark Cieliebak, and Eneko Agirre. 2020. Doqa–accessing domain-specific faqs via conversational qa. *arXiv preprint arXiv:2005.01328*.

Eunsol Choi, He He, Mohit Iyyer, Mark Yatskar, Wentau Yih, Yejin Choi, Percy Liang, and Luke Zettlemoyer. 2018. Quac: Question answering in context. *arXiv preprint arXiv:1808.07036*.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*.

Yifan Gao, Piji Li, Irwin King, and Michael R Lyu. 2019. Interconnected question generation with coreference alignment and conversation flow modeling. *arXiv preprint arXiv:1906.06893*.

Jing Gu, Mostafa Mirshekari, Zhou Yu, and Aaron Sisto. 2021. Chaincqg: Flow-aware conversational question generation. *arXiv preprint arXiv:2102.02864*.

Seonjeong Hwang and Gary Geunbae Lee. 2021. A study on the automatic generation of conversational qa corpora. In *Annual Conference on Human and Language Technology*, pages 133–138. Human and Language Technology.

Mao Nakanishi, Tetsunori Kobayashi, and Yoshihiko Hayashi. 2019. Towards answer-unaware conversational question generation. In *Proceedings of the 2nd Workshop on Machine Reading for Question Answering*, pages 63–71.

Boyuan Pan, Hao Li, Ziyu Yao, Deng Cai, and Huan Sun. 2019. Reinforced dynamic reasoning for conversational question generation. *arXiv preprint arXiv:1907.12667*.

Peng Qi, Yuhao Zhang, and Christopher D Manning. 2020. Stay hungry, stay focused: Generating informative and specific questions in information-seeking conversations. *arXiv preprint arXiv:2004.14530*.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J Liu. 2019. Exploring the limits of transfer learning with a unified text-to-text transformer. *arXiv preprint arXiv:1910.10683*.

Siva Reddy, Danqi Chen, and Christopher D Manning. 2019. Coqa: A conversational question answering challenge. *Transactions of the Association for Computational Linguistics*, 7:249–266.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. *Advances in neural information processing systems*, 30.

Yansen Wang, Chenyi Liu, Minlie Huang, and Liqiang Nie. 2018. Learning to ask questions in open-domain conversational systems with typed decoders. *arXiv preprint arXiv:1805.04843*.

Tianyi Zhang, Varsha Kishore, Felix Wu, Kilian Q Weinberger, and Yoav Artzi. 2019. Bertscore: Evaluating text generation with bert. *arXiv preprint arXiv:1904.09675*.

A Examples of Negative Sampling

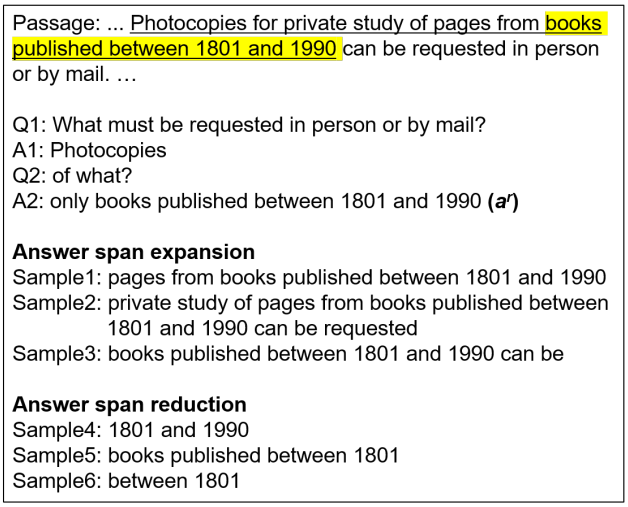

Figure 3: Example of negative sampling applied to one passage in CoQA. The rationale for A2 is underlined, and *proper* a^s corresponding to A2 is highlighted in the passage. A2 is a^r for both the *proper* and *improper* a^s samples.

B Training Details

To implement the CAE module, we used parameters of BERT-large-uncased. Only the previous two pairs of Q–A were used as the conversation history to extract the i-th answer span and passages p longer than 512 tokens were truncated with a stride of 128 tokens:

$$\textbf{Input:}\ [\text{CLS}]\ q_{i-2}\ a_{i-2}\ q_{i-1}\ a_{i-1}\ [\text{SEP}]\ \text{truncated}\ p\ [\text{SEP}]$$

For the CQG-AR module, we initialized the module with parameters of T5-large. The input and target sequences for generation of the i-th Q–A pair are as follows:

$$\textbf{Input:}\ a_i^s\ \text{highlighted}\ p\ [\text{SEP}]\ [\text{A}]\ a_{i-4}\ [\text{Q}]\ q_{i-4}\ \ldots\ [\text{A}]\ a_{i-1}\ [\text{Q}]\ q_{i-1}[\text{A}]\ a_i^s$$
$$\textbf{Target:}\ [\text{Q}]\ q_i\ [\text{A}]\ a_i^r\ [\text{EOS}]$$

Because the input sequence length of the T5 encoder was limited, we truncated the passage p at the 32nd token after the location of a_i^s. Regarding the conversation history, only the previous four Q–A pairs were used. Special tokens ([Q] and [A]) were added before each question and answer, and [A] and a_i^s were appended to the end of the input sequence. [Q] was used as a bos token, and answer generation started when [A] was returned after predicting the question.

We utilized the Transformers library and pre-trained parameters from HuggingFace[3] and conducted experiments using A100 GPUs. Further, AdamW was used as the optimization algorithm with a batch size of 4 and a learning rate of 3e-5. In addition, a learning rate scheduling algorithm was applied and the warm-up period was set to the initial 10% of the total steps. For CAE, we optimize the module based on F1 score between predicted answer span and the ground truth. For CQG-AR, beam search with a beam size of 4 was used during data generation. The best module was selected based on METEOR (Banerjee and Lavie, 2005) and BERTScore (Zhang et al., 2019) on the synthetic development set.

For CQA, we designed a simple T5-based model that accepted the concatenation of the passage, conversation history, and question as input and then generated the answer to the input question. We initialized our model with T5-Large and trained the model with AdamW, setting the batch size between 4 and 8 and the learning rate to 3e-5. When fine-tuning the Wikipedia-domain CQA model with synthetic data in Section 4.5, we fine-tuned the model for one epoch, with a batch size of 1 and the learning rate between 1e-7 and 1e-6. We employed the same training and decoding strategies used for the CQG-AR module.

[3] https://huggingface.co/

C Synthetic Data

C.1 Example of Synthetic Conversation

Passage: CHAPTER IV. Signor Andrea D'Arbino, searching vainly through the various rooms in the palace for Count Fabio d'Ascoli, and trying as a last resource, the corridor leading to the ballroom and grand staircase, discovered his friend lying on the floor in a swoon, without any living creature near him. Determining to avoid alarming the guests, if possible, D'Arbino first sought help in the antechamber. He found there the marquis's valet, assisting the Cavaliere Finello (who was just taking his departure) to put on his cloak. While Finello and his friend carried Fabio to an open window in the antechamber, the valet procured some iced water. This simple remedy, and the change of atmosphere, proved enough to restore the fainting man to his senses, but hardly–as it seemed to his friends–to his former self. They noticed a change to blankness and stillness in his face, and when he spoke, an indescribable alteration in the tone of his voice. "I found you in a room in the corridor," said D'Arbino. "What made you faint? Don't you remember? Was it the heat?" Fabio waited for a moment, painfully collecting his ideas. He looked at the valet, and Finello signed to the man to withdraw. "Was it the heat?" repeated D'Arbino. "No," answered Fabio, in strangely hushed, steady tones. "I have seen the face that was behind the yellow mask." "Well?" "It was the face of my dead wife." "Your dead wife!" "When the mask was removed I saw her face. Not as I remember it in the pride of her youth and beauty–not even as I remember her on her sick-bed–but as I remember her in her coffin."

Conversation

Q1: Who was searching for Fabio d'Ascoli?
A1: Signor Andrea D'Arbino
Q2: Where was he searching?
A2: the palace
Q3: What was his last resort?
A3: the corridor leading to the ballroom and grand staircase
Q4: What did he find?
A4: lying on the floor in a swoon
Q5: Where did he seek help first?
A5: the antechamber
Q6: Who helped him?
A6: the marquis's valet
Q7: Who helped him put on his cloak?
A7: the Cavaliere Finello
Q8: What did the valet give him?
A8: some iced water
Q9: What change did his friends notice?
A9: a change to blankness and stillness in his face
Q10: What did D'Arbino say was the cause?
A10: heat
Q11: What was behind the mask?
A11: the face of my dead wife
Q12: What did I see?
A12: her face
Q13: How did I remember her?
A13: in her coffin

Table 5: Samples of generated Q-A pairs using CQAG-AR from a Wikipedia passage in CoQA. Answer spans before revision are highlighted in the passage in order.

C.2 Statistics of Synthetic Data

	Synthetic dataset	CoQA
#Words in question	5.6	5.4
#Words in answer	3.0	2.6
#Turns per passage	12.1	15.1

Table 6: Average number of words in the questions and answers, and the average number of conversation turns in CoQA and our synthetic data extracted from CoQA passages.

Domain	#Passages		#Q–A pairs	
	Train	Dev	Train	Dev
Wikipedia	1.6k	0.1k	23.6k	1.5k
News	1.7k	0.1k	21.6k	1.2k
Mid/High Sch.	1.7k	0.1k	22.2k	1.3k
Literature	1.6k	0.1k	17.7k	1.1k
Children's Sto.	0.6k	0.1k	6.3k	1.2k

Table 7: Statistics summarizing the synthetic datasets generated from CoQA passages.

D Criteria for Human Evaluation

Question Connectivity	
Dependent	The current question refers to previous conversations (e.g., via pronoun usage or ellipses).
Independent	The current question is not dependent on previous conversations.
Unnatural	The current question has grammatical errors or overlaps with previous conversations.
Answer Correctness	
Correct	Questions are paired with correct answers.
Partially correct	Answers are incomplete or contain unnecessary information.
Incorrect	Not the correct answer to the question.

Table 8: Criteria for human evaluation of synthetic CQA data.

DABERT: Dual Attention Enhanced BERT for Semantic Matching

Sirui Wang[1,*] , **Di Liang**[1,2,†] , **Jian Song**[†] , **Yuntao Li** [†], **Wei Wu**[†]
Tsinghua University, Beijing, China[*]
Meituan Inc.,Beijing, China[†]
{wangsirui,liangdi04,songjian20,liyuntao04,wuwei30}@meituan.com

Abstract

Transformer-based pre-trained language models such as BERT have achieved remarkable results in Semantic Sentence Matching. However, existing models still suffer from insufficient ability to capture subtle differences. Minor noise like word addition, deletion, and modification of sentences may cause flipped predictions. To alleviate this problem, we propose a novel **Dual Attention Enhanced BERT (DABERT)** to enhance the ability of BERT to capture fine-grained differences in sentence pairs. DABERT comprises (1) Dual Attention module, which measures soft word matches by introducing a new dual channel alignment mechanism to model affinity and difference attention. (2) Adaptive Fusion module, this module uses attention to learn the aggregation of difference and affinity features, and generates a vector describing the matching details of sentence pairs. We conduct extensive experiments on well-studied semantic matching and robustness test datasets, and the experimental results show the effectiveness of our proposed method.

1 Introduction

Semantic Sentence Matching (SSM) is a fundamental NLP task. The goal of SSM is to compare two sentences and identify their semantic relationship. In paraphrase identification, SSM is used to determine whether two sentences are paraphrase or not (Madnani et al., 2012). In natural language inference task, SSM is utilized to judge whether a hypothesis sentence can be inferred from a premise sentence (Bowman et al., 2015). In the answer sentence selection task, SSM is employed to assess the relevance between query-answer pairs and rank all candidate answers (Wang et al., 2020).

Across the rich history of semantic sentence matching research, there have been two main streams of studies for solving this problem. One is to utilize a sentence encoder to convert sentences into low-dimensional vectors in the latent space, and apply a parameterized function to learn the matching scores between them (Reimers and Gurevych, 2019; Wang et al., 2020). Another paradigm adopts attention mechanism to calculate scores between tokens from two sentences, and then the matching scores are aggregated to make a sentence-level decision (Chen et al., 2016; Tay et al., 2017). In recent years, pre-trained models, such as BERT (Devlin et al., 2018), RoBERTa (Liu et al., 2019), have became much more popular and achieved outstanding performance in SSM. Recent work also attempts to enhance the performance of BERT by injecting knowledge into it, such as SemBERT (Zhang et al., 2020), UER-BERT (Xia et al., 2021), Syntax-BERT (Bai et al., 2021) and so on.

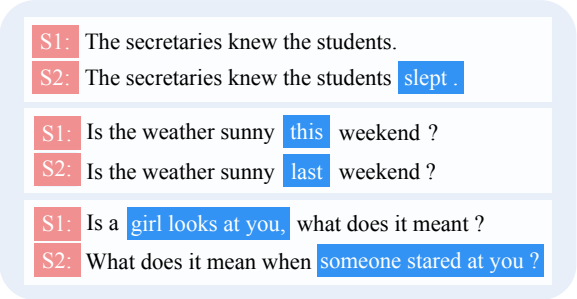

Figure 1: Example sentences with similar text but different semantics. S1 and S2 are sentence pair.

Although previous studies have provided some insights, those models do not perform well in distinguishing sentence pairs with high literal similarities but different semantics. Figure 1 demonstrates several cases suffering from this problem. Although the sentence pairs in this figure are semantically different, they are too similar in literal for those pre-trained language models to distinguish accurately. This could be caused by the self-attention architecture itself. Self-attention mechanism focuses on using the context of a word to understand the semantics of the word, while ignoring model-

[1]These authors contributed equally to this work.
[2]Corresponding author.

Proceedings of the 29th International Conference on Computational Linguistics, pages 1645–1654
October 12–17, 2022.

ing the semantic difference between sentence pairs. De-attention (Tay et al., 2019) and Sparsegen (Martins and Astudillo, 2016) have proved that equipping with attention mechanism with more flexible structure, models can generate more powerful representations. In this paper, we also focus on enhancing the attention mechanism in transformer-based pre-trained models to better integrate difference information between sentence pairs. We hypothesize that paying more attention to the fine-grained semantic differences, explicitly modeling the difference and affinity vectors together will further improve the performance of pre-trained model. Therefore, two systemic questions arise naturally:

Q1: How to equip vanilla attention mechanism with the ability on modeling semantics of fine-grained differences between a sentence pair? Vanilla attention, or named affinity attention, less focuses on the fine-grained difference between sentence pairs, which may lead to error predictions for SSM tasks. An intuitive solution to this problem is to make subtraction between representation vectors to harvest their semantic differentiation. In this paper, we propose a dual attention module including a difference attention accompanied with the affinity attention. The difference attention uses subtraction-based cross-attention to aggregate word- and phrase- level interaction differences. Meanwhile, to fully utilize the difference information, we use dual-channel inject the difference information into the multi-head attention in the transformer to obtain semantic representations describing affinity and difference respectively.

Q2: How to fuse two types of semantic representations into a unified representation? A hard fusion of two signals by extra structure may break the representing ability of the pre-trained model. How to inject those information softly to pre-trained model remains a hard issue. In this paper, we propose an Adaptive Fusion module, which uses an additional attention to learn the difference and affinity features to generate vectors describing sentence matching details. It first inter-aligns the two signals through distinct attentions to capture semantic interactions, and then uses gated fusion to adaptively fuse the difference features. Those generated vectors are further scaled with another fuse-gate module to reduce the damage of the pre-trained model caused by the injection of difference information. The output final vectors can better describe the matching details of sentence pairs.

Our main contributions are three fold:

- We point out that explicitly modeling fine-grained difference semantics between sentence pairs can effectively benefit sentence semantic matching tasks, and we propose a novel dual attention enhanced mechanism based on BERT.
- Our proposed DABERT model uses a dual-channel attention to separately focus on the affinity and difference features in sentence pairs, and adopts a soft-integrated regulation mechanism to adaptively aggregate those two features. Thereby, the generated vectors can better describe the matching details of sentence pairs.
- To verify the effectiveness of DABERT, we conduct experiments on 10 semantic matching datasets and several data-noised dataset to test model's robustness. The results show that DABERT achieves an absolute improvement for over 2% compared with pure BERT and outperforms other BERT-based models with more advanced techniques and external data usage.

2 Approach

Our proposed DABERT is a modification of the original transformer structure, whose structure is shown in Figure 2. Two submodules are included in this new structure. (1) Dual Attention Module, which uses a dual channel mechanism in multi-head attention to match words between two sentences. Each channel uses a different attention head to calculate affinity and difference scores separately, and obtains two representations to measure affinity and difference information respectively. (2) Adaptive Fusion Module, which is used to fuse the representation obtained by dual attention. It first uses guide-attention to align the two signals. And then, multiple gate modules are used to fuse the two signals. Finally, a vector is output including more fine-grained matching details. In the following sections, we explain each component in detail.

2.1 Dual Attention Module

In this module, we use two distinct attention functions, namely affinity attention and difference attention, to compare the affinities and differences of vectors between two sentences. The input of the dual attention module is a triple of $K, Q, V \in R^{d_{seq} \times d_v}$, where d_v is the latent dimension and

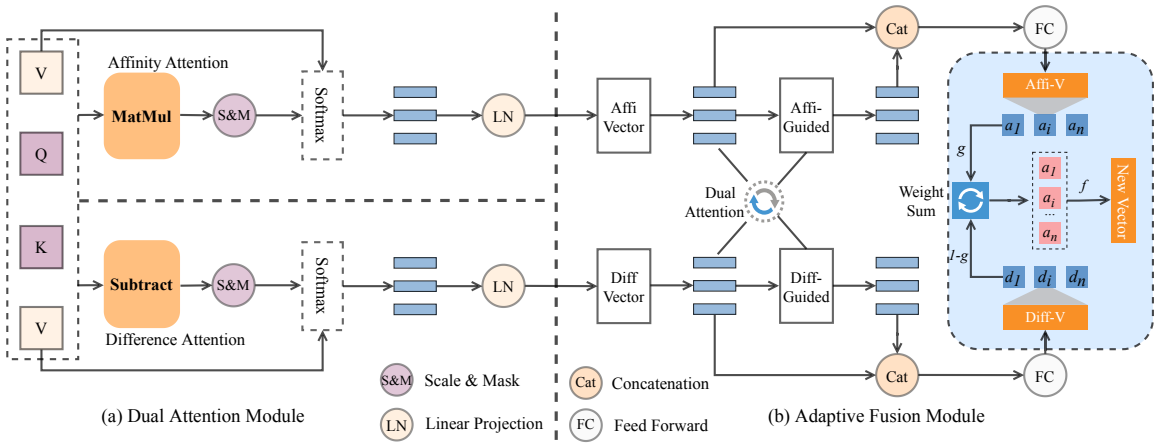

Figure 2: The overall architecture of Dual Attention Enhanced BERT (DABERT). The left side is the Dual attention module, and the right side is the Adaptive Fusion module.

d_{seq} is the utterance length. Dual attention module calculate the latent relationship between K, Q and V via two separate attention mechanism to measure their affinity and difference. As a result, two set of attention representations are generated by the dual attention module, which will be fused by the following adaptive fusion module.

2.1.1 Affinity Attention

The affinity attention module is the part of the dual attention module, which is the standard dot-product attention that operates following Transformer's default operation. The input of affinity attention module consists of queries and keys of dimension d_k, and values of dimension d_v. We compute the dot products of the query with all keys, divide each by $\sqrt{d_k}$, and apply a softmax function to obtain the weights on the values. For the sake of simplicity, the formulations of BERT not be repeated here, please refer to (Devlin et al., 2018) for more details. We denote the output affinity vector as:

$$\mathbf{A} = softmax(\frac{\mathbf{Q}\mathbf{K}^T}{\sqrt{d_k}}) * \mathbf{V}, \tag{1}$$

where $\mathbf{A} = \{\boldsymbol{a}_1, ..., \boldsymbol{a}_l\} \in R^{d_l \times d_v}$ denotes the vector describing affinity expressions generated by the Transformer original attention module.

2.1.2 Difference Attention

The second part of dual attention module is a difference attention module that capture and aggregate the difference information between sentence pairs. The difference attention module adopts a subtraction-based cross-attention mechanism, which allows model to pay attention to dissimilar parts between sentence pairs by element-wise subtraction as:

$$\mathbf{D} = softmax(\frac{\beta}{\sqrt{d_k}}) * \mathbf{V}, \tag{2}$$

$$\boldsymbol{\beta} = \|\mathbf{Q} - \mathbf{K}\| + \mathbf{M}, \tag{3}$$

$$\|\mathbf{Q} - \mathbf{K}\|_{ij} = \sum_{k=0}^{d_k} \mathbf{Q}_{ik} - \mathbf{K}_{jk}, \tag{4}$$

where $\|\mathbf{Q} - \mathbf{K}\| \in R^{d_l \times d_l}$ and d_l is the input sequence length. We use $\mathbf{D} = \{\boldsymbol{d}_1, ..., \boldsymbol{d}_l\} \in R^{d_l \times d_v}$ to denote the representation generated by the difference attention. The $\mathbf{M} \in R^{d_l \times d_l}$ is a masking operation. Both the affinity attention and the difference attention are utilized to fit the semantic relationship of sentence pairs, and obtain the representations with the same dimension from the perspective of affinity and difference respectively. This dual channel mechanism can obtain more detailed representations describing sentence matching.

2.2 Adaptive Fusion Module

After obtaining the affinity signals $\mathbf{A}$ and the difference signals $\mathbf{D}$, we introduce a novel adaptive fusion module to fuse these two signals instead of direct fusion (i.e., average embedding vector), since direct fusion may compromise the original representing ability of the pre-trained model. The fusion process includes three steps. First, it flexibly interacts and aligns these two signals via affinity-guided attention and difference-guided attention. Second, multiple gate modules are adopted to selectively extract interaction semantic information. Finally, to alleviate the damage of the pre-trained model by the difference signal, we utilize filter gates to adaptively filter out noisy information and finally

generate vectors that better describe the details of sentence matching.

Firstly, we update the difference vectors through affinity-guided attention. We use $\boldsymbol{a}_i$ and $\boldsymbol{d}_i$ to denote the i-th dimension of $\mathbf{A}$ and $\mathbf{D}$ respectively. We provide each affinity vector $\boldsymbol{a}_i$ to interact with the difference signals matrix $\mathbf{D}$ and obtain the new difference feature $\boldsymbol{d}_i^*$. Then, based on $\boldsymbol{d}_i^*$, we can in turn acquire the new Affinity feature $\boldsymbol{a}_i^*$ through difference-guided attention. The calculation process is as follows:

$$\begin{aligned}
\delta_i &= \tanh(\mathbf{W}_D\mathbf{D} \oplus (\mathbf{W}_{a_i}a_i + b_{a_i})),\\
\overline{d}_i &= \mathbf{D} * softmax(\mathbf{W}_{d_i}\delta_i + b_{d_i}),\\
\gamma_i &= \tanh(\mathbf{W}_A\mathbf{A} \oplus (\mathbf{W}_{\overline{d}_i}\overline{d}_i + b_{\overline{d}_i})),\\
\overline{a}_i &= \mathbf{A} * softmax(\mathbf{W}_{\overline{a}_i}\gamma_i + b_{\overline{a}*_i}),\\
d_i^* &= \tanh(\mathbf{W}_{d_i^*}([d_i;\overline{d}_i]) + b_{d_i^*})),\\
a_i^* &= \tanh(\mathbf{W}_{a_i^*}([a_i;\overline{a}_i]) + b_{a_i^*})),
\end{aligned} \tag{5}$$

where $\mathbf{W}_D, \mathbf{W}_A, \mathbf{W}_{a_i}, \mathbf{W}_{\overline{d}_i} \in R^{d_l * d_v}$; $\mathbf{W}_{d_i}$, $\mathbf{W}_{\overline{a}_i} \in R^{1*2d_l}$; $b_{d_i^*}$, b_{a_i}, b_{d_i}, $b_{a_i^*}$ are weights and bias of our model, and $\oplus$ denotes the concatenation of signal matrix and feature vector. Secondly, to adaptively capture and fuse useful information from Affinity and difference features, we introduce our gate fusion modules:

$$\begin{aligned}
\hat{d}_i &= \tanh(\mathbf{W}_{\hat{d}_i}d_i^* + b_{\hat{d}_i}),\\
\hat{a}_i &= \tanh(\mathbf{W}_{\hat{a}_i}a_i^* + b_{\hat{a}_i}),\\
g_i &= \sigma(\mathbf{W}_{g_i}(\hat{d}_i \oplus \hat{a}_i)),\\
v_i &= g_i\hat{a}_i + (1-g_i)\hat{d}_i,
\end{aligned} \tag{6}$$

where $\mathbf{W}_{\hat{d}_i}$, $\mathbf{W}_{\hat{a}_i} \in R^{d_h * d_v}$; $\mathbf{W}_{g_i} \in R^{1*2d_h}$; $b_{\hat{d}_i}$, $b_{\hat{a}_i}$ are parameters and d_h is the size of hidden layer. σ is the sigmoid activation function and g_i is the gate that determines the transmission of these two distinct representations. By the way, we get the fusion feature $\boldsymbol{v}_i$.

Eventually, considering the potential noise problem, we propose a filtering gate to selectively leverage the fusion feature. When $\boldsymbol{v}_i$ tends to be beneficial, the filtration gate will incorporate the fusion features and the original features. Otherwise, the fusion information will be filtered out:

$$\begin{aligned}
f_i &= \sigma(\mathbf{W}_{f_i,a_i}(a_i \oplus (\mathbf{W}_{v_i}v_i + b_{v_i}))),\\
l_i &= f_i * \tanh(\mathbf{W}_{l_i}v_i + b_{l_i}),
\end{aligned} \tag{7}$$

where $\mathbf{W}_{f_i,a_i} \in R^{1*2d_v}$; $\mathbf{W}_{v_i}$, $\mathbf{W}_{l_i} \in R^{d_v * d_h}$; b_{v_i}, b_{l_i} are trainable parameters and $\boldsymbol{l}_i$ is the final fused semantic feature and it will be propagated to the next computation flow.

3 Experimental Settings

3.1 Datasets

Semantic Matching. We conduct experiments on 10 sentence matching datasets to evaluate the effectiveness of our method. The GLUE (Wang et al., 2018) benchmark is a widely-used dataset in thie field, which includes tasks such as sentence pair classification, similarity and paraphrase detection, and natural language inference[1]. We conduct experiments on 6 sentence pair datasets (MRPC, QQP, STS-B, MNLI, RTE, and QNLI) from GLUE. We also conduct experiments on 4 other popular datasets (SNLI (Bowman et al., 2015), SICK (Marelli et al., 2014), TwitterURL (Lan et al., 2017) and Scitail (Khot et al., 2018)). The statistics of all 10 datasets are shown in Table 6.

Robustness Test. TextFlint (Gui et al., 2021) is a robustness evaluation platform for natural language processing models[2]. It includes more than 80 patterns to deform data, including inserting punctuation marks, changing numbers in text, replacing synonyms, modifying adverbs, deleting words, etc. It can effectively evaluate the robustness and generalization of models. In this paper, we leverage TextFlint to perform transformations on multiple datasets (Quora, SNLI, MNLI-m/mm), including task-specific transformations (SwapAnt, NumWord, AddSent) and general transformations (InsertAdv, Appendlrr, AddPunc, BackTrans, TwitterType, SwapNamedEnt, SwapSyn-WordNet). We conduct experiments on datasets with those types of transformations to verify the robustness of our model.

3.2 Baselines

To evaluate the effectiveness of our proposed DABERT in SSM, we mainly introduce BERT (Devlin et al., 2018), SemBERT (Zhang et al., 2020), SyntaxBERT (Liu et al., 2020), UERBERT (Xia et al., 2021) and multiple other PLMs (Radford et al., 2018; Devlin et al., 2018) for comparison. In addition, we also select several competitive models without pre-training as baselines, such as ESIM (Chen et al., 2016), Transformer (Vaswani et al., 2017) , etc (Hochreiter and Schmidhuber, 1997; Wang et al., 2017; Tay et al., 2017). In robustness experiments, we compare the performance of multiple pre-trained models (Sanh et al., 2019; Chen et al., 2016; Devlin et al., 2018; Lan et al., 2019)

[1]https://huggingface.co/datasets/glue
[2]https://www.textflint.io

Model	Pre-train	Sentence Similarity			Sentence Inference			Avg
		MRPC	QQP	SST-B	MNLI-m/mm	QNLI	RTE	
BiMPM†(Wang et al., 2017)	✗	79.6	85.0	-	72.3/72.1	81.4	56.4	-
CAFE†(Tay et al., 2017)	✗	82.4	88.0	-	78.7/77.9	81.5	56.8	-
ESIM†(Chen et al., 2016)	✗	80.3	88.2	-	-	80.5	-	-
Transformer†(Vaswani et al., 2017)	✗	81.7	84.4	73.6	72.3/71.4	80.3	58.0	74.53
BiLSTM+ELMo+Attn†(Devlin et al., 2018)	✓	84.6	86.7	73.3	76.4/76.1	79.8	56.8	76.24
OpenAI GPT†(Radford et al., 2018)	✓	82.3	70.2	80.0	82.1/81.4	87.4	56.0	77.06
UERBERT‡(Xia et al., 2021)	✓	88.3	90.5	85.1	84.2/83.5	90.6	67.1	84.19
SemBERT†(Zhang et al., 2020)	✓	88.2	90.2	87.3	84.4/84.0	90.9	69.3	84.90
BERT-base‡(Devlin et al., 2018)	✓	87.2	89.0	85.8	84.3/83.7	90.4	66.4	83.83
SyntaxBERT-base†(Bai et al., 2021)	✓	**89.2**	89.6	88.1	84.9/84.6	91.1	68.9	85.20
DABERT-base‡	✓	89.1	**91.3**	**88.2**	**84.9/84.7**	**91.4**	**69.5**	**85.58**
BERT-large‡(Devlin et al., 2018)	✓	89.3	89.3	86.5	86.8/85.9	92.7	70.1	85.80
SyntaxBERT-large†(Bai et al., 2021)	✓	**92.0**	89.5	88.5	86.7/86.6	92.8	74.7	87.26
DABERT-large‡	✓	91.4	**91.9**	**89.5**	**87.1/86.9**	**94.8**	**75.3**	**88.12**

Table 1: The performance comparison of DABERT with other methods. We report Accuracy $\times$ 100 on 6 GLUE datasets. Methods with † indicate the results from their papers, while methods with ‡ indicate our implementation.

Model	SNLI	Sci	SICK	Twi
ESIM†(Chen et al., 2016)	88.0	70.6	-	-
CAFE†(Tay et al., 2017)	88.5	83.3	72.3	-
CSRAN†(Tay et al., 2018)	88.7	86.7	-	84.0
BERT-base‡(Devlin et al., 2018)	90.7	91.8	87.2	84.8
UERBERT‡(Xia et al., 2021)	90.8	92.2	87.8	86.2
SemBERT†(Zhang et al., 2020)	90.9	92.5	87.9	86.8
SyntaxBERT-base†(Bai et al., 2021)	91.0	92.7	**88.7**	87.3
DABERT-base‡	**91.3**	**93.6**	88.6	**87.5**
BERT-large‡(Devlin et al., 2018)	91.0	94.4	91.1	91.5
SyntaxBERT-large†(Bai et al., 2021)	91.3	94.7	91.4	92.1
DABERT-large‡	**91.5**	**95.3**	**92.5**	**92.3**

Table 2: The performance comparison of DABERT with other methods on 4 popular datasets, including SNLI, Scitail(Sci), SICK and TwitterURL(Twi).

and SemBERT,UERBERT and Syntax-BERT on the robustness test datasets. For simplicity, the compared models are not described in detail here.

3.3 Implementation Details

DABERT is based on BERT-base and BERT-large. For distinct targets, our hyper-parameters are different. We use AdamW in the BERT and set the learning rate in $\{1e^{-}5, 2e^{-}5, 3e^{-}5, 8e^{-}6\}$. As for the learning rate decay, we use a warmup of 0.1 and L2 weight decay of 0.01. Furthermore, we set the epoch to 5 and the batch size is selected in {16, 32, 64}. We also set dropout at 0.1-0.3. To prevent gradient explosion, we set gradient clipping in {7.5, 10.0, 15.0}. All the experiments are conducted by Tesla V100 and PyTorch platform. In addition, to ensure that the experimental results are statistically significant, we conduct each experiment five times and report the average results.

4 Results and Analysis

4.1 Model Performance

In our experiments, we implement DABERT in the initial transformer layer of BERT.

First, we fine-tune our model on 6 GLUE datasets. Table 1 shows the performance of DABERT and other competitive models. It can be seen that using only non-pretrained models performs obviously worse than PLMs due to their strong context awareness and data fitting capabilities. When the backbone model is BERT-base or BERT-large, the average accuracy of DABERT respectively improves by 1.7% and 2.3% than vanilla BERT. Such great improvement demonstrates the benefit of fusion difference attention for mining semantics and proves that our framework can help BERT perform much better in SSM.

Moreover, compared with some previous works such as SemBERT, UERBERT and SyntaxBERT, DABERT achieves the best performance without injecting external knowledge. Specifically, our model outperforms SyntaxBERT, the best performing model in previous work leveraging external knowledge, with an average relative improvement of 0.86% based on BERT-large. On the QQP dataset, the accuracy of DABERT is significantly improved by 2.4% over SyntaxBERT. There are two main reasons for such results. On the one hand, we use dual-channel attention to enhance the ability of DABERT to capture difference features. This enables DABERT to obtain more fine-grained interaction matching features. On the other hand, for

Model	Quora					SNLI				
	SA	NW	IA	Al	BT	AS	SA	TT	SN	SW
ESIM†(Chen et al., 2016)	-	-	-	-	-	64.00	84.22	78.32	53.76	65.38
BERT‡(Devlin et al., 2018)	48.58	56.96	86.32	**85.48**	83.42	79.66	94.84	83.56	50.45	76.42
ALBERT‡(Lan et al., 2019)	51.08	55.24	81.87	78.94	82.37	45.17	96.37	81.62	57.66	74.93
UERBERT‡(Xia et al., 2021)	48.57	54.86	84.72	80.88	82.71	73.24	94.78	85.36	57.54	80.81
SemBERT‡(Zhang et al., 2020)	50.92	53.15	85.19	82.04	82.40	76.81	95.31	84.60	56.28	77.86
SyntaxBERT‡(Bai et al., 2021)	49.30	56.37	86.43	84.62	84.19	78.63	95.02	**86.91**	58.26	76.90
DABERT‡	**60.43**	**62.76**	**87.50**	85.48	**87.49**	**81.06**	**96.85**	85.14	**60.58**	**80.92**

Method	MNLI-m/mm					
	AS	SA	AP	TT	SN	SW
BERT‡(Devlin et al., 2018)	55.32/55.25	52.76/55.69	82.30/82.31	77.08/77.22	51.97/51.84	76.41/77.05
ALBERT‡(Lan et al., 2019)	53.09/53.58	50.25/50.20	**83.98/83.68**	**77.98**/78.03	56.43/50.03	76.63/77.43
UERBERT‡(Xia et al., 2021)	54.99/54.84	52.29/53.80	79.80/79.18	75.46/74.93	55.21/55.96	**82.23**/82.74
SemBERT‡(Zhang et al., 2020)	55.38/55.12	54.07/54.62	78.70/78.16	73.90/73.47	53.43/53.76	78.09/78.93
SyntaxBERT‡(Bai et al., 2021)	54.92/54.63	53.54/54.73	77.01/76.71	70.38/70.13	57.11/51.95	78.57/79.31
DABERT‡	**60.14/59.25**	**60.89/61.37**	83.23/83.19	77.94/**78.10**	**60.12/59.83**	82.15/**82.97**

Table 3: The robustness experiment results of DABERT and other models. The data transformation methods we utilized mainly include SwapAnt (SA), NumWord (NW), AddSent (AS), InsertAdv (IA), Appendlrr (Al), AddPunc (AP), BackTrans (BT), TwitterType (TT), SwapNamedEnt (SN), SwapSyn-WordNet (SW).

the potential noise problem introduced by external structures, our adaptive fusion module can selectively filter out inappropriate information to suppress the propagation of noise, and previous work does not seem to pay enough attention to this problem. However, we still notice that SyntaxBERT achieves slightly better accuracy on a few datasets. We argue that this is a result of the intrinsic correlation of syntactic and dependent knowledge.

Second, to verify the general performance of our method, we also conduct experiments on other popular datasets. The results are shown in Table 2. DABERT still outperforms vanilla BERT and other models on almost all datasets. It is worth noting that DABERT performs worse than SyntaxBERT on SICK. This may be because the data volume of SICK is relatively small, and SyntaxBERT uses syntactic prior knowledge, which makes SyntaxBERT more advantageous on small datasets. but DABERT still shows a very competitive performance on SICK, which also shows from the side that our method can enhance the difference capture ability of BERT and make up for the lack of generalization ability with fewer parameters.

Overall, our method has competitive performance in judging semantic similarity compared to previous work. Extensive performance improvements also validate our point, soft ensemble difference information based on BERT's powerful contextual representation capability is useful for sentence matching tasks.

4.2 Robustness Test Performance

In order to examine the performance of DABERT and competitive models in their ability to capture subtle differences in sentence pairs. We perform robustness tests on three extensively studied datasets.

Table 3 lists the accuracy of DABERT and six baseline models on the three datasets. We can observe that SwapAnt leads to a drop in maximum performance, and our model outperforms the best model SemBert nearly 10% on SwapAnt(QQP), which indicates that DABERT can better handle semantic contradictions caused by antonyms than baseline models. And the model performance drops to 56.96% on NumWord transformation, while DABERT outperforms BERT by nearly 6% because it requires the model to capture subtle numerical differences for correct linguistic inference. In SwapSyn transformation, UERBERT significantly outperforms other baseline models because it explicitly uses the synonym similarity matrix to calibrate the attention distribution, while our model can still achieve comparable performance to UERBERT without adding external knowledge. On TwitterType and AddPunc, the performance of SyntaxBERT by injecting syntax trees degrades significantly, probably because converting text to twitter type or adding punctuation breaks the normal syntactic structure of sentences. And DABERT still achieves the competitive performance in these two transformations. In other scenarios, DABERT also achieve better performance due to capturing subtle

Case	ESIM	BERT	SyntaxBERT	DABERT
S1:How done you solve this aptitude question? S2:How does I solve aptitude questions on cube?	label:1	label:0	label:0	similarity:10.87% label:0
S1:How can I tell if this girl loves me? S2:How can I tell if this boy loves me?	label:1	label:1	label:1	similarity:12.06% label:0
S1:How many 12 digits number have the sum of 4? S2:How many 42 digits number have the sum of 4?	label:1	label:1	label:1	similarity:18.63% label:0

Table 4: The example sentence pairs of our cases. Red and Blue are difference phrases in sentence pair.

Model	Quora		QNLI	
	Dev	Test	Dev	Test
DABERT	**92.1**	**91.3**	**92.9**	**91.4**
w/o Affi-attention	90.1	89.5	91.8	90.7
w/o Diff-attention	90.6	89.8	92.0	90.8
w/o Guide-attention	91.3	90.4	92.1	91.0
w/o Gate fusion	91.7	90.6	92.5	91.1
w/o Gate filter	91.8	90.9	92.6	91.2
w/o Regulation	89.9	89.4	91.5	90.7

Table 5: Results of component ablation experiment.

differences in sentence pairs. Meanwhile, ESIM has the worst performance, the results reflect that the pre-training mechanism benefits from rich external resources and provides better generalization ability than de novo trained models. And the improved pre-trained model SyntaxBERT performs better than the original BERT model, which reflects that sufficient pre-trained corpus and suitable external knowledge fusion strategies can help improve the generalization performance of the model.

4.3 Ablation Study

To evaluate the contribution of each component in our method, we conduct ablation experiments on the QQP and QNLI datasets based on BERT. The experimental results are shown in the table 5.

Above all, the dual attention module consists of two core components that use a two-channel mechanism to model affinity and difference attention. First, after removing affinity attention, the performance of the model on the two datasets drops by 1.8% and 0.7%. Affinity attention can capture the dynamic alignment relationship between word pairs, which is crucial for SSM tasks. Next,after removing difference attention from the model, the performance on the two datasets dropped by 1.5% and 0.6%, respectively. The difference information can further describe the interaction between words, and can provide more fine-grained comparison information for the pre-trained model, so that the model can obtain a better representation. The above experiments show that the performance drops sharply after the submodule is removed, which demonstrates the effectiveness of the internal components of the dual attention module.

Next, in the adaptive fusion module, we also conducted several experiments to verify the effect of the fusion of affinity and difference vectors. On the QQP dataset, we first remove the guide attention module, and the performance drops to 90.4%. Since guide attention can capture the interaction between two signals, this interaction information is crucial for fusing two different information. Second, after removing the fusion gate, we only integrate two signals by simple averaging. The accuracy dropped to 91.4%, indicating that dynamically merging the affinity and difference vectors according to different weights can improve the performance of the model. Then, when the filter gate is removed, the accuracy drops by 0.4%, indicating that the ability of the model to suppress noise is weakened without the filter gate. Finally, we also replaced the overall aggregation and Regulation module with simple average, and the performance dropped sharply to 89.4%. While difference information is crucial for judging sentence-pair relations, hard-integrating the difference information into the PLMs will destroy its Pre-existing knowledge, and soft aggregation and governance can make better use of difference signals.

Overall, due to the effective combination of each component, DABERT can adaptively fuse difference features into pretrained models and leverage its powerful contextual representation to better inference about semantics.

4.4 Case Study

To visualize how DABERT works, we use three cases from the table 4 for qualitative analysis. In the first case, the non-pretrained language model ESIM is difficulty capturing the semantic conflicts caused by the difference words. Therefore, ESIM

Datasets	#Train	#Dev	#Test	#Class
MRPC	3669	409	1380	2
QQP	363871	1501	390965	2
MNLI-m/mm	392703	9816/9833	9797/9848	3
QNLI	104744	40432	5464	2
RTE	2491	5462	3001	2
SST-B	5749	1500	1379	2
SNLI	549367	9842	9824	3
SICK	4439	495	4906	3
Scitail	23596	1304	2126	2
TwitterURL	42200	3000	9324	2

Table 6: The statistics of all 10 datasets.

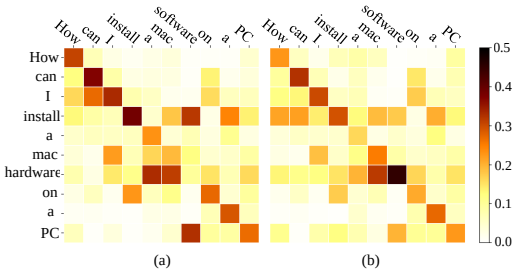

Figure 3: Distribution of BERT (a) and our method (b).

gives wrong prediction results in case 1. BERT can identify the semantic difference in case 1 with the help of context representation . But in case 3, BERT cannot capture the difference between the numbers "12" and "24" and give wrong prediction. SyntaxBERT enhances text understanding by introducing syntactic trees. Since case 2 and case 3 have the same syntactic structure, SyntaxBERT also gives wrong predictions. Our model made correct predictions in all of the above cases. Because DABERT explicitly focuses on different parts of sentence pairs through difference attention and adaptively aggregates affinity and difference information in the adaptive fusion module, it can identify semantic differences caused by subtle differences within sentence pairs.

Attention Distribution. To verify the fusion effect of subtraction-based attention on the difference information, we display the weights distribution of BERT and DABERT in Figure 3 for comparison. It can be seen that the attention distribution after dual attention becomes more reasonable, especially the attention weight between "hardware" and "software" increases significantly. This reveals that DABERT pays more attention to different parts of sentence pairs rather than the same words.

5 Related Work

Semantic Sentence Matching is a fundamental task in NLP. Thanks to the appearance of large-scale annotated datasets (Bowman et al., 2015; Williams et al., 2017), neural network models have made great progress in SSM (Qiu and Huang, 2015; Wan et al., 2016), mainly fell into two categories. The first (Conneau et al., 2017; Choi et al., 2018) focuses on encoding sentences into corresponding vectors without cross-interaction and applies a classifier to obtain similarity. The second (Wang et al., 2017; Chen et al., 2016; Liang et al., 2019a) utilizes cross-features as an attention module to express the word- or phrase-level alignments of two texts, and aggregates it into prediction layer to acquire similarity. Recently, the pre-training paradigm has achieved great results in SSM. Some work attempt to introduce other methods to enhance pre-trained models. For example, SemBERT (Zhang et al., 2020) explicitly absorbs contextual semantics over a BERT backbone. AMAN (Liang et al., 2019b) uses answers knowledge to enhance language representation. UER-BERT (Xia et al., 2021) injects synonym knowledge to enhance BERT. SyntaxBERT (Bai et al., 2021) also integrates the syntax tree into transformer models.

Robustness Although neural network models have achieved human-like or even superior results in multiple tasks, they still face the insufficient robustness problem in real application scenarios (Gui et al., 2021). Tiny literal changes may cause misjudgments. Therefore, recent work starts to focus on robustness research from multiple perspectives. TextFlint (Gui et al., 2021) incorporates multiple transformations to provide comprehensive robustness analysis. Li et al. (2021) provide an overall benchmark for current work on adversarial attacks. And Liu et al. (2021) propose a more comprehensive evaluation system and add more detailed output analysis indicators.

6 Conclusions

In this paper, we propose a novel Dual Attention Enhanced BERT (DABERT), which can efficiently aggregate the difference information in sentence pairs and soft-integrate it into a pretrained model. Based on BERT's powerful contextual representation capability, DABERT enables the model to learn more fine-grained comparative information and enhances the sensitivity of PLMs to subtle differences. Experimental results on 10 public datasets and robustness dataset show that our method can achieve better performance than several strong baselines. Since DABERT is an end-to-end training component, it is expected to be applied to other large-scale pre-trained models in the future.

References

Jiangang Bai, Yujing Wang, Yiren Chen, Yaming Yang, Jing Bai, Jing Yu, and Yunhai Tong. 2021. Syntax-bert: Improving pre-trained transformers with syntax trees. *arXiv preprint arXiv:2103.04350*.

Samuel R Bowman, Gabor Angeli, Christopher Potts, and Christopher D Manning. 2015. A large annotated corpus for learning natural language inference. *arXiv preprint arXiv:1508.05326*.

Qian Chen, Xiaodan Zhu, Zhenhua Ling, Si Wei, Hui Jiang, and Diana Inkpen. 2016. Enhanced lstm for natural language inference. *arXiv preprint arXiv:1609.06038*.

Jihun Choi, Kang Min Yoo, and Sang-goo Lee. 2018. Learning to compose task-specific tree structures. In *Thirty-Second AAAI Conference on Artificial Intelligence*.

Alexis Conneau, Douwe Kiela, Holger Schwenk, Loic Barrault, and Antoine Bordes. 2017. Supervised learning of universal sentence representations from natural language inference data. *arXiv preprint arXiv:1705.02364*.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*.

Tao Gui, Xiao Wang, Qi Zhang, Qin Liu, Yicheng Zou, Xin Zhou, Rui Zheng, Chong Zhang, Qinzhuo Wu, Jiacheng Ye, et al. 2021. Textflint: Unified multilingual robustness evaluation toolkit for natural language processing. *arXiv preprint arXiv:2103.11441*.

Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long short-term memory. *Neural computation*, 9(8):1735–1780.

Tushar Khot, Ashish Sabharwal, and Peter Clark. 2018. Scitail: A textual entailment dataset from science question answering. In *Thirty-Second AAAI Conference on Artificial Intelligence*.

Wuwei Lan, Siyu Qiu, Hua He, and Wei Xu. 2017. A continuously growing dataset of sentential paraphrases. *arXiv preprint arXiv:1708.00391*.

Zhenzhong Lan, Mingda Chen, Sebastian Goodman, Kevin Gimpel, Piyush Sharma, and Radu Soricut. 2019. Albert: A lite bert for self-supervised learning of language representations. *arXiv preprint arXiv:1909.11942*.

Zongyi Li, Jianhan Xu, Jiehang Zeng, Linyang Li, Xiaoqing Zheng, Qi Zhang, Kai-Wei Chang, and Cho-Jui Hsieh. 2021. Searching for an effective defender: Benchmarking defense against adversarial word substitution. *arXiv preprint arXiv:2108.12777*.

Di Liang, Fubao Zhang, Qi Zhang, and Xuan-Jing Huang. 2019a. Asynchronous deep interaction network for natural language inference. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 2692–2700.

Di Liang, Fubao Zhang, Weidong Zhang, Qi Zhang, Jinlan Fu, Minlong Peng, Tao Gui, and Xuanjing Huang. 2019b. Adaptive multi-attention network incorporating answer information for duplicate question detection. In *Proceedings of the 42nd International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 95–104.

Pengfei Liu, Jinlan Fu, Yang Xiao, Weizhe Yuan, Shuaicheng Chang, Junqi Dai, Yixin Liu, Zihuiwen Ye, Zi-Yi Dou, and Graham Neubig. 2021. Explainaboard: An explainable leaderboard for nlp. *arXiv preprint arXiv:2104.06387*.

Weijie Liu, Peng Zhou, Zhe Zhao, Zhiruo Wang, Qi Ju, Haotang Deng, and Ping Wang. 2020. K-bert: Enabling language representation with knowledge graph. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pages 2901–2908.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Roberta: A robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692*.

Nitin Madnani, Joel Tetreault, and Martin Chodorow. 2012. Re-examining machine translation metrics for paraphrase identification. In *Proceedings of the 2012 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 182–190, Montréal, Canada. Association for Computational Linguistics.

Marco Marelli, Stefano Menini, Marco Baroni, Luisa Bentivogli, Raffaella Bernardi, Roberto Zamparelli, et al. 2014. A sick cure for the evaluation of compositional distributional semantic models. In *Lrec*, pages 216–223. Reykjavik.

Andre Martins and Ramon Astudillo. 2016. From softmax to sparsemax: A sparse model of attention and multi-label classification. In *International conference on machine learning*, pages 1614–1623. PMLR.

Xipeng Qiu and Xuanjing Huang. 2015. Convolutional neural tensor network architecture for community-based question answering. In *Twenty-Fourth international joint conference on artificial intelligence*.

Alec Radford, Karthik Narasimhan, Tim Salimans, and Ilya Sutskever. 2018. Improving language understanding with unsupervised learning.

Nils Reimers and Iryna Gurevych. 2019. Sentence-bert: Sentence embeddings using siamese bert-networks. *arXiv preprint arXiv:1908.10084*.

Victor Sanh, Lysandre Debut, Julien Chaumond, and Thomas Wolf. 2019. Distilbert, a distilled version of bert: smaller, faster, cheaper and lighter. *arXiv preprint arXiv:1910.01108*.

Yi Tay, Anh Tuan Luu, Aston Zhang, Shuohang Wang, and Siu Cheung Hui. 2019. Compositional de-attention networks. In *Advances in Neural Information Processing Systems*, volume 32. Curran Associates, Inc.

Yi Tay, Luu Anh Tuan, and Siu Cheung Hui. 2017. A compare-propagate architecture with alignment factorization for natural language inference. *arXiv preprint arXiv:1801.00102*, 78:154.

Yi Tay, Luu Anh Tuan, and Siu Cheung Hui. 2018. Co-stack residual affinity networks with multi-level attention refinement for matching text sequences. *arXiv preprint arXiv:1810.02938*.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in neural information processing systems*, pages 5998–6008.

Shengxian Wan, Yanyan Lan, Jiafeng Guo, Jun Xu, Liang Pang, and Xueqi Cheng. 2016. A deep architecture for semantic matching with multiple positional sentence representations. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 30.

Alex Wang, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel R Bowman. 2018. Glue: A multi-task benchmark and analysis platform for natural language understanding. *arXiv preprint arXiv:1804.07461*.

Shuohang Wang, Yunshi Lan, Yi Tay, Jing Jiang, and Jingjing Liu. 2020. Multi-level head-wise match and aggregation in transformer for textual sequence matching. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pages 9209–9216.

Zhiguo Wang, Wael Hamza, and Radu Florian. 2017. Bilateral multi-perspective matching for natural language sentences. *arXiv preprint arXiv:1702.03814*.

Adina Williams, Nikita Nangia, and Samuel R Bowman. 2017. A broad-coverage challenge corpus for sentence understanding through inference. *arXiv preprint arXiv:1704.05426*.

Tingyu Xia, Yue Wang, Yuan Tian, and Yi Chang. 2021. Using prior knowledge to guide bert's attention in semantic textual matching tasks. In *Proceedings of the Web Conference 2021*, pages 2466–2475.

Zhuosheng Zhang, Yuwei Wu, Hai Zhao, Zuchao Li, Shuailiang Zhang, Xi Zhou, and Xiang Zhou. 2020. Semantics-aware bert for language understanding. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pages 9628–9635.

Locate Then Ask: Interpretable Stepwise Reasoning for Multi-hop Question Answering

Siyuan Wang[1], **Zhongyu Wei**[1,2]*, **Zhihao Fan**[1], **Qi Zhang**[3], **Xuanjing Huang**[3]
[1]School of Data Science, Fudan University, China
[2]Research Institute of Intelligent and Complex Systems, Fudan University, China
[3]School of Computer Science, Fudan University, China
{wangsy18,zywei,fanzh18,qz,xjhuang}@fudan.edu.cn

Abstract

Multi-hop reasoning requires aggregating multiple documents to answer a complex question. Existing methods usually decompose the multi-hop question into simpler single-hop questions to solve the problem for illustrating the explainable reasoning process. However, they ignore grounding on the supporting facts of each reasoning step, which tends to generate inaccurate decompositions. In this paper, we propose an interpretable stepwise reasoning framework to incorporate both single-hop supporting sentence identification and single-hop question generation at each intermediate step, and utilize the inference of the current hop for the next until reasoning out the final result. We employ a unified reader model for both intermediate hop reasoning and final hop inference and adopt joint optimization for more accurate and robust multi-hop reasoning. We conduct experiments on two benchmark datasets HotpotQA and 2WikiMultiHopQA. The results show that our method can effectively boost performance and also yields a better interpretable reasoning process without decomposition supervision. [1]

1 Introduction

Recent years have witnessed an emerging trend in the task of multi-hop question answering. It requires the model to aggregate multiple pieces of documents (i.e., context) and perform multi-hop reasoning to infer the answer (Talmor and Berant, 2018; Khashabi et al., 2018). Several datasets have been introduced as benchmarks, such as HotpotQA (Yang et al., 2018), 2WikiMultiHopQA (Ho et al., 2020) and WikiHop (Welbl et al., 2018), and the first two provide supporting facts supervision to encourage models to further explain what supporting sentences lead to the prediction.

The first generation of models for multi-hop question answering utilizes a one-step reader (Qiu et al., 2019; Fang et al., 2019; Shao et al., 2020; Tu et al., 2020; Beltagy et al., 2020) to capture the interaction between the question and relevant contexts for the prediction of the answer as well as the supporting sentences. In order to model the explainable multi-step reasoning process, researchers explore to decompose the multi-hop question into easier single-hop questions and solve sub-questions to reach the answer (Talmor and Berant, 2018; Wolfson et al., 2020).

Question decomposition based approaches achieve promising prediction performance and are able to demonstrate the reasoning process to some

[Question]
Q: What city is the Marine Air Control Group 28 located in?
[Context]
P1: Marine Tactical Air Command Squadron 28 is a United States Marine Corps aviation command and control unit based at Marine Corps Air Station Cherry Point. They provide the 2nd Marine Aircraft Wings tactical headquarters and …
P2: Marine Corps Air Station Cherry Point or MCAS Cherry Point is a United States Marine Corps airfield located in **Havelock**, North Carolina, United States, in the eastern part of the state. It ...
P3-P10: …

[Question Decomposition (Min et al., 2019)**]**
Sub-Q1: Which Marine Air Control Group 28?
Sub-Q2: What city is Marine Tactical Air Command Squadron 28 located in?

[Stepwise Decomposition]
Step1-S: Marine Tactical Air Command Squadron 28 is a … control unit based at Marine Corps Air Station Cherry Point.
Step1-Q: Which is the base of Marine Air Control Group 28?

Step2-S: Marine Corps Air Station Cherry Point ... United States Marine Corps airfield located in Havelock, North Carolina … state.
Step2-Q: What city is the Marine Corps Air Station Cherry Point located in?

Figure 1: A multi-hop reasoning example from HotpotQA. To solve the problem, DecompRC (Min et al., 2019) generates improper decomposition of questions and predicts a wrong answer while our expected stepwise decomposition includes both single-hop supporting sentences and sub-questions of each step to reason out the correct answer. The underlined phrase is the fact uncovered by machine-generated decomposition while the shaded contexts support the corresponding single-hop question generation.

*Corresponding author
[1]Codes are publicly available at https://github.com/WangsyGit/StepwiseQA.

Proceedings of the 29th International Conference on Computational Linguistics, pages 1655–1665
October 12–17, 2022.

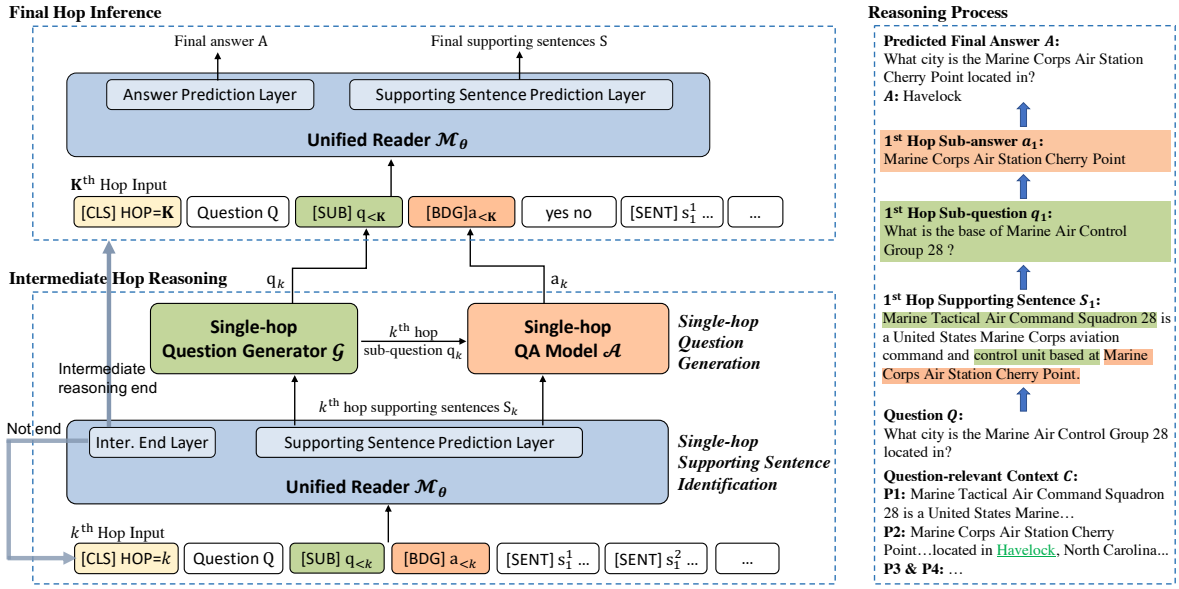

Figure 2: The overall architecture of stepwise reasoning framework with an interpretable reasoning process.

extent. However, the single-hop questions are generated solely based on the original question without considering the supporting facts each step involves (Min et al., 2019; Perez et al., 2020; Khot et al., 2020). This usually leads to wrong-guided decomposition and inaccurate explanations. An example is shown in Figure 1 including a question, two relevant contextual paragraphs, two sub-questions generated by Min et al. (2019) and two expected steps of reasoning with supporting sentences and sub-questions. The first single-hop question generated by an existing model (*Sub-Q1*) fails to query "the base of Marine Air Control Group 28" which is beyond the scope of the original multi-hop question and such an improperly reasoned hop also leads to the failure of final answer prediction. We argue that a complete step of reasoning should consist of intermediate supporting sentence identification and sub-question generation to reduce the inference error in the procedure.

In this paper, we propose a stepwise reasoning framework for multi-hop question answering. It performs both single-hop supporting sentence identification and single-hop question generation in each step, and reasons from one intermediate hop to the next until the final hop inference. Specifically, we perform an intermediate hop reasoning that locates the single-hop supporting sentences and constructs the sub-question based on the original question and the corresponding supporting facts in each step. We utilize an off-the-shelf single-hop question generator to eliminate the need for human annotations and avoid the risk of noisy labels posed by constructed pseudo-supervision. In the final hop, we simultaneously predict the answer and the supporting sentences of the multi-hop question according to the preceding hops. We employ a unified reader model for both intermediate single-hop supporting sentence identification and final hop inference and jointly learn them so that a midway error may be corrected by subsequent hops to mitigate cumulative failures. We further adopt two measures to reduce the train-test discrepancy of single-hop supporting sentences and sub-questions to mitigate exposure bias for better generalization.

Experiments are conducted on two benchmark datasets involving different hops of reasoning, HotpotQA (Yang et al., 2018) and 2WikiMultiHopQA (Ho et al., 2020). The results indicate that our stepwise reasoning framework achieves significant improvements and shows general effectiveness across different reasoning types. Further analysis and qualitative cases also demonstrate that our method generates high-quality single-hop questions for interpretable multi-hop reasoning.

2 Methodology

Given a multi-hop question Q and a context including multiple paragraphs, we aim to read the question-relevant context C to predict the final answer A and explain it with the supporting sentences S. As illustrated in Figure 2, we present a stepwise reasoning framework to iteratively identify the single-hop supporting sentences and generate

the single-hop question for the following reasoning, which consists of three components as below. It first filters out the unrelated paragraphs to extract the question-relevant context C (§ 2.1). Then it identifies the supporting sentences of each intermediate hop from the relevant context to ask and answer the corresponding single-hop question, and passes the auto-generated messages to the next hop (§ 2.2). After intermediate hop reasoning ends, the last module predicts the final answer and the supporting sentences of the multi-hop question according to the preceding inference (§ 2.3). We jointly train a unified reader model for all reasoning hops (§ 2.4) and adopt two measures to mitigate the train-test discrepancy for better inference (§ 2.5).

2.1 Context Filter

In order to reduce the distraction in the context for downstream multi-hop reasoning process, we select the most relevant paragraphs as the question-relevant context C. We first train a paragraph selection model which takes the question and the concatenation of all candidate paragraphs as the input and predicts the probability scores that each paragraph is relevant to answer the question.

As HotpotQA mainly consists of 2-hop questions which involve two relevant paragraphs, we follow the 2-hop selection strategy in (Fang et al., 2019). For the first hop, it selects the paragraph with the highest scores among the paragraphs containing the same phrases as the question. Then the second-hop paragraph is extracted by hyperlinks from the first selected one. Two other paragraphs with the next highest scores are also selected to constitute the question-relevant context. For 2WikiMultiHopQA dataset with 2∼4 hop questions, we select five paragraphs with the highest scores and filter the other paragraphs from the context.

2.2 Intermediate Hop Reasoning

Based upon the filtered relevant context, we perform the multi-hop reasoning step-by-step. We adopt a unified reader model $\mathcal{M}_\theta$ to iteratively identify the single-hop supporting sentences focused at each intermediate hop, and decide whether to end the intermediate hop reasoning indicating that the cumulative intermediate information is ready for final hop inference. Then depending on the predicted supporting sentences at each hop, the corresponding single-hop question can be generated and answered, which will be passed to the reader for next hop reasoning. This iterative process is repeated until the intermediate reasoning is ended, or up to $K-1$ intermediate hops.

Single-hop Supporting Sentence Identification The reader attempts to find the supporting sentences at each hop $k \in \{1, ..., K-1\}$ given the concatenation of the original question Q, the sub question-answer pairs $\{(q_1, a_1)\ ...\ (q_{k-1}, a_{k-1})\}$ of previous hops, and the relevant context C as the input. Specifically, the concatenated sequence is formulated as `[CLS] HOP=k [SEP] Q` `[SUB]` q_1 `[BDG]` a_1 `...` `[SUB]` q_{k-1} `[BDG]` a_{k-1} `[SEP] [SENT]` s_1^1 `[SENT]` s_2^1 `...[SEP] [SENT]` s_1^2 `...[SEP]` where `HOP=k` indicates the current hop number. Special tokens `[SUB]`, `[BDG]`, `[SENT]` respectively represent the single-hop sub-question, sub-answer and each sentence, and s_j^i is the j-th sentence of the i-th paragraph in the relevant context C.

On top of the representations of each sentence token `[SENT]`, we build a binary classifier to predict the probability $p_{i,j}^{(k)}$ that each sentence s_j^i is a supporting fact of the current hop. The corresponding binary cross entropy loss $\mathcal{L}_{sf}^{(k)}$ is calculated as Eq. 1. $y_{i,j}^{(k)}$ is the label whether the sentence s_j^i is a supporting fact of the hop k, and N_s is the total number of sentences in C.

$$\mathcal{L}_{sf}^{(k)} = \frac{1}{N_s}\sum_i\sum_j -y_{i,j}^{(k)}\log(p_{i,j}^{(k)}) - (1-y_{i,j}^{(k)})\log(1-p_{i,j}^{(k)}) \quad (1)$$

Then the `[CLS]` representation is also fed into a binary classifier to compute the probability $p_{end}^{(k)}$ that the intermediate reasoning should be ended at hop k and go on to final hop inference, and the cross entropy loss is as Eq. 2. $y_{end}^{(k)}$ is the label whether to end the intermediate hop reasoning at current hop k.

$$\mathcal{L}_{end}^{(k)} = -y_{end}^{(k)}\log(p_{end}^{(k)}) - (1-y_{end}^{(k)})\log(1-p_{end}^{(k)}) \quad (2)$$

Single-hop Question Generation After identifying the supporting sentences S_k of hop k, we generate the corresponding single-hop question q_k to investigate what the current hop asks about. In this work, we do not use annotated or pseudo supervision to train a question decomposition model. Instead, we take inspiration from (Pan et al., 2020) and adopt a pre-trained simple question generator

$\mathcal{G}_s$ to directly output the desired single-hop question, which is trained beforehand on top of an off-the-shelf simple question corpus.

To encourage the single-hop question more grounded on the contextual facts, we generate them based on both the identified single-hop supporting sentences S_k and the multi-hop question Q. The latter serves as a guidance for the generation towards the original reasoning goal. Specifically, we extract the intersectional tokens between S_k and Q as the prompt and append it to the supporting sentences S_k as the input for single-hop question generation, which is organized as `[CLS]` ($\mathrm{Q} \cap \mathrm{S_k}$) `[SEP]` $\mathrm{S_k}$ `[SEP]`. Correspondingly, during the single-hop question generator pre-training, we also take a single sentence as the context and utilize the tokens existing within both the target question and the context to prompt simple question generation.

Queried with the generated single-hop question q_k, we immediately resolve it to ease the whole multi-hop question. We also leverage the aforementioned simple question dataset to pre-train a single-hop QA model $\mathcal{A}$ to make it more consistent with the single-hop question generator. Then according to the single-hop supporting sentences S_k and question q_k at hop k, we utilize the pre-trained QA model $\mathcal{A}$ to output the single-hop answer a_k, which together with the single-hop question q_k will be passed to the next hop for subsequent reasoning.

2.3 Final Hop Inference

After the end of intermediate hop reasoning, we can utilize the single-hop questions and answers $\{(q_1, a_1)$ ... $(q_{K-1}, a_{K-1})\}$ of all previous hops to build a bridge for inferring the final hop K. We employ the same unified reader model $\mathcal{M}_\theta$ during intermediate hop reasoning to predict the final answer A of the multi-hop question Q and simultaneously provide the overall explanatory supporting sentences S. The input sequence fed into the reader for the final hop is similar to intermediate hops, except that we insert two additional tokens `yes` and `no` before the relevant context C for answer prediction. As there are three answer types (yes/no/span), we integrate the answer type classification into the answer span prediction by appending `yes` and `no` as two candidate spans. We reformulate the current input as `[CLS] HOP=K [SEP] Q [SUB]` q_1 `[BDG]` a_1 `...` `[SUB]` q_{K-1} `[BDG]` a_{K-1} `[SEP]` **`yes no`** `[SEP] [SENT]` s_1^1 `...[SEP] [SENT]` s_1^2 `...[SEP]`.

To accomplish the final hop inference, we first utilize the same binary classifier to identify whether each sentence is a supporting fact of the whole multi-hop question, and compute a final supporting sentence identification loss $\mathcal{L}_{sf}^{(K)}$. Then for the final answer span prediction, we attach a linear layer with a softmax function to all context representations to obtain the probability of each token t_n being the start p_n^s or end position p_n^e of the answer span. The cross entropy loss is calculated as following formula, where token t_x and t_y are respectively the labels of start or end positions.

$$\mathcal{L}_{span} = -\log(p_x^s) - \log(p_y^e) \tag{3}$$

2.4 Optimization & Inference

In order to optimize our framework, we can first set up a maximum number of required reasoning hops K. Then our stepwise reasoning framework essentially comprises $K-1$ iterative intermediate hop reasoning layers and a final hop inference layer. As there is no previous single-hop question-answer pair before the first hop reasoning, we omit them within the initial input which will be fed into the first intermediate hop reasoning layer. We jointly train our stepwise reasoning framework for all intermediate hops and the final hop in order that an intermediate mistake can be corrected by subsequent hops to mitigate cascading failures. All losses are combined in a weighted manner:

$$\mathcal{L} = \lambda_1 \mathcal{L}_{sf}^{int} + \lambda_2 \mathcal{L}_{end}^{int} + \lambda_3 \mathcal{L}_{sf}^{(K)} + \mathcal{L}_{span} \tag{4}$$

$$\mathcal{L}_{sf}^{int} = \frac{\mathcal{L}_{sf}^{(1)} + \sum_{k=2}^{K-1}(1 - y_{end}^{(k-1)})\mathcal{L}_{sf}^{(k)}}{1 + \sum_{k=2}^{K-1}(1 - y_{end}^{(k-1)})} \tag{5}$$

$$\mathcal{L}_{end}^{int} = \frac{1}{k_e}\sum_{k=1}^{k_e}\mathcal{L}_{end}^{(k)}, \text{where } y_{end}^{(k_e)} = 1 \tag{6}$$

where λ_1, λ_2 and λ_3 are weighted hyper-parameters. $\mathcal{L}_{sf}^{int}$ is the average supporting sentence identification loss of all actual intermediate hops that are not ended. $\mathcal{L}_{end}^{int}$ is the average loss of intermediate reasoning end prediction.

During the inference period, we start from the first hop and dynamically reason from one hop to the next until the final hop. We predict whether to end the intermediate reasoning at each intermediate hop. Once it is over, we utilize all generated sub-questions and sub-answers and move to conduct the final hop inference. If not end, we will pass them

to the next intermediate hop and repeat the process until the intermediate hop reasoning is ended, or until reaching $K-1$ intermediate hops.

2.5 Exposure Bias Mitigation

In light of the design of our stepwise reasoning framework, there may arise the exposure bias problem between optimization and inference. Given the ground-truth supporting sentence supervision for each intermediate hop S_k^t, we can generate the target single-hop question and answer for the follow-up reasoning at training time. However, at test time we can only conduct single-hop question generation based on the predicted single-hop supporting sentences S_k which may deviate from the oracle ones S_k^t. To address this, we propose two measures to respectively reduce the discrepancy between train-test single-hop supporting sentences and train-test single-hop questions.

Firstly, we train a separate reader model only for the intermediate single-hop supporting sentence identification, and adopt it to re-predict the single-hop supporting sentences of training data with occasionally injected mistakes for optimizing the whole framework. Thereby we can regulate bias between training and test supporting sentences of intermediate hops. Besides, we also augment the training data for the single-hop question generation $\mathcal{G}_s$ by taking the re-predicted training single-hop supporting sentences S_k as input, and the generated sub-questions based on the ground-truth supporting sentences S_k^t as the target. Then the generator is trained to recover from the non-gold single-hop supporting sentences to approximate the oracle ones and reduce the deviation between train-test intermediate single-hop questions. With these two strategies, we can jointly optimize our stepwise reasoning framework for better generalization to non-golden test cases.

3 Experiments

3.1 Experimental Dataset

We take two datasets HotpotQA (Yang et al., 2018) and 2WikiMultiHopQA (Ho et al., 2020) that involve different reasoning hops as a testbed to study textual multi-hop reasoning. They both require answering the question as well as predicting the supporting facts to explain the reasoning. HotpotQA includes both distractor setting and fullwiki setting, and we focus on the former with limited candidate paragraphs to fully test the multi-hop reasoning ability while putting aside the information retrieval part. Although 2WikiMultiHopQA provides annotated evidence for interpreting the reasoning path, we leave them out of account to illustrate the effectiveness and interpretability of our framework without explanation supervision.

HotpotQA and 2WikiMultiHopQA respectively consist of $90,447/7,405/7,405$ and $167,454/12,576/12,576$ samples in training, development and test sets, and each instance is provided with 10 paragraphs. HotpotQA comprises 2-hop questions that only two paragraphs contain necessary supporting sentences, while 2WikiMultiHopQA contains 2∼4 hop questions and the supporting facts reside in two to four paragraphs. Besides, to train the single-hop question generator and QA model, we use SQuAD (Rajpurkar et al., 2016) as the simple question corpus.

3.2 Implementation Details

We take ELECTRA-large (Clark et al., 2020) as the backbone of our proposed framework and the single-hop QA model, and train a single-hop question generator using BART-large (Lewis et al., 2019). The weights to balance losses are chosen as $\lambda_1 = 10/5$ (for HotpotQA/2WikiMultiHopQA), $\lambda_2 = 2$ and $\lambda_3 = 5$. The maximum value of required reasoning hops is set as $K = 2$ for HotpotQA and $K = 4$ for 2WikiMultiHopQA. More training details are given in Appendix A.

3.3 Overall Performance

We compare our **step**wise **reasoner** (*StepReasoner*) with previous published methods on HotpotQA and 2WikiMultiHopQA. Since there are few systems on the leaderboard of 2WikiMultiHopQA, we follow (Fu et al., 2021) and make a comparison with more models on both dev and test sets. The compared methods cover both question decomposition based models and one-step reading based models.

Question decomposition based models include *DecompRC* (Min et al., 2019), *ONUS* (Perez et al., 2020), *QFE* (Nishida et al., 2019), *CRERC* (Fu et al., 2021)) and *NA-Reviewer* (Fu et al., 2022).

One-step reading based models consist of *DFGN* (Qiu et al., 2019), *ELECTRA* (Clark et al., 2020), *TAP2* (Glass et al., 2019), *SAE-large* (Tu et al., 2020), *C2F Reader* (Shao et al., 2020), *Longformer* (Beltagy et al., 2020), *ETC-large* (Zaheer et al., 2020), *FFReader-large* (Alkhaldi et al., 2021), *HGN-large* (Fang et al., 2019).

Model	Answer		Sup Fact		Joint	
	EM	F1	EM	F1	EM	F1
DecompRC (Min et al., 2019)	55.20	69.63	-	-	-	-
ONUS (Perez et al., 2020)	66.33	79.34	-	-	-	-
TAP2 (Glass et al., 2019)	64.99	78.59	55.47	85.57	39.77	69.12
SAE-large (Tu et al., 2020)	66.92	79.62	61.53	86.86	45.36	71.45
C2F Reader (Shao et al., 2020)	67.98	81.24	60.81	87.63	44.67	72.73
Longformer (Beltagy et al., 2020)	68.00	81.25	63.09	88.34	45.91	73.16
ETC-large (Zaheer et al., 2020)	68.12	81.18	**63.25**	**89.09**	46.40	73.62
FFReader-large (Alkhaldi et al., 2021)	68.89	82.16	62.10	88.42	45.61	73.78
HGN-large (Fang et al., 2019)	69.22	82.19	62.76	88.47	47.11	74.21
StepReasoner	**69.66**	**82.42**	62.99	87.85	**47.84**	**74.27**

Table 1: Experimental results of different models on the test set of HotpotQA distractor setting.

Model	Answer		Sup Fact	
	EM	F1	EM	F1
Dev				
DecompRC (Min et al., 2019)	7.46	41.57	56.49	82.73
QFE (Nishida et al., 2019)	37.56	43.21	21.13	59.20
CRERC (Fu et al., 2021)	71.56	74.51	**86.00**	**92.75**
NA-Reviewer (Fu et al., 2022)	76.88	82.30	-	-
DFGN (Qiu et al., 2019)	30.87	38.49	17.06	57.79
ELECTRA-base	66.81	72.28	81.19	90.96
ELECTRA-large	79.22	83.51	83.08	92.01
StepReasoner(ELECTRA-base)	68.11	73.03	81.72	91.21
StepReasoner(ELECTRA-large)	**80.23**	**84.26**	83.41	92.01
Test				
HGN-revise (Fang et al., 2019)	71.20	75.69	69.35	89.07
CRERC (Fu et al., 2021)	69.58	72.33	82.86	90.68
NA-Reviewer (Fu et al., 2022)	76.73	81.91	**89.61**	**94.31**
StepReasoner	**80.88**	**84.86**	83.30	91.89

Table 2: Results on the dev and test sets of 2WikiMultiHopQA.

The results are shown in Table 1 and 2. We find that *StepReasoner* outperforms all models in terms of both answer prediction and joint evaluation and achieves comparable performance in supporting fact prediction, which demonstrates the effectiveness of our method. Specifically, it performs better than both question decomposition based and one-step reading based methods. The former improvement indicates that a unified reader to stepwise identify the single-hop supporting sentences for single-hop sub-question generation can enhance the accuracy of previous question decomposition methods. The latter verifies that the interpretability injected by stepwise reasoning can also improve the QA performance. Besides, *Longformer*, *ETC-large*, *FFReader-large* and *HGN-large* show a better supporting fact prediction performance on HotpotQA, especially in F1 score. This is because the first three models are designed for handling longer sequences and the last utilizes a complex hierarchical graph network which both can cover more candidate paragraphs for supporting sentence prediction. For 2WikiMultiHopQA, *CRERC* and *NA-Reviewer* achieve better supporting fact prediction because they both utilize external annotated evidence for training which confirms the effectiveness of our *StepReasoner* without explanation supervision. We also evaluate the intermediate reasoning end prediction, and find that our *StepReasoner* can exactly decide when to end the intermediate hop reasoning.

3.4 Further Analysis

Ablation Study To dive into the sources of performance gain in our *StepReasoner*, we conduct an ablation study on the HotpotQA development set, which is shown in Table 3. Compared to the overall stepwise reasoning system *StepReasoner*, a pipeline model without joint training shows a sharp performance degradation. It indicates that joint optimization of a unified reader model for all hops can improve the tolerance for intermediate faults and boost reasoning performance. After removing any measures to mitigate exposure bias, the performance has also significantly dropped. It shows that both two measures to alleviate the train-test discrepancy of single-hop supporting sentences and single-hop questions confirm a better generalization to cases deviated from oracle.

Effectiveness on Various Backbone Models To analyze the effectiveness of the *StepReasoner* based on different backbones, we vary several pre-trained models of different scales including *BERT-base-uncased* (Liu et al., 2019), *ELECTRA-large*

Model	Answer		Sup Fact		Joint	
	EM	F1	EM	F1	EM	F1
StepReasoner	70.11	83.03	64.27	88.10	48.55	74.85
w/o joint training	69.30	82.44	63.35	87.89	47.25	74.16
w/o bias.supp	69.66	82.64	63.10	87.74	47.39	74.20
w/o bias.ques	69.76	82.93	63.46	88.01	47.49	74.57

Table 3: Ablation study on HotpotQA dev set. *w/o joint training* means a pipeline stepwise reasoning schema. *bias.supp* and *bias.ques* are two exposure **bias** mitigating measures to reduce the train-test discrepancy of single-hop **supp**orting sentences and **ques**tions.

and *ALBERT-xxlarge-v2* (Lan et al., 2019). From Table 4, we can see that our *StepReasoner* variants consistently perform better than the corresponding baseline models and the previous state-of-the-art method (HGN) using *ALBERT-large* as base model, especially in EM scores. It demonstrates that our method is robust to be effective based on various pre-trained models and it is the paradigm of our joint stepwise reasoning that contributes to more accurate multi-hop reasoning.

Model	Answer		Sup Fact		Joint	
	EM	F1	EM	F1	EM	F1
BERT	60.80	74.76	57.16	85.05	38.67	65.89
StepReasoner(BERT)	60.52	74.81	59.00	85.38	40.31	66.50
ELECTRA	69.49	82.76	62.80	87.91	46.75	74.33
StepReasoner(ELECTRA)	70.11	83.03	64.27	88.10	48.55	74.85
ALBERT	70.14	83.58	62.78	88.43	46.60	75.30
HGN(ALBERT)	70.18	83.44	63.17	89.19	47.01	75.74
StepReasoner(ALBERT)	70.73	83.92	64.17	88.69	48.54	75.85

Table 4: Analysis of *StepReasoner* on different backbone models on HotpotQA dev set.

Analysis of Different Reasoning Types We detailedly investigate the performance of *StepReasoner* on various reasoning types of HotpotQA compared to the baseline model. Following Min et al. (2019), HotpotQA integrates four types of multi-hop reasoning skills, including "Bridge", "Implicit-Bridge", "Comparison" and "Intersection". The first two both require identifying the bridge entity to complete the chain reasoning, but "Implicit-Bridge" resembles single-hop questions which implicitly query a multi-hop property of an entity, such as the question in Fig. 1. "Intersection" questions ask to locate the answer entity that satisfies multiple properties. "Comparison" questions involve comparing two entities to find the answer.

As shown in Table 5, our system is generally effective on all reasoning types compared to the baseline model *ELECTRA*, especially "Implicit-Bridge" and "Intersection". Because these questions are susceptible to shortcut solutions by directly identifying an entity satisfying one queried property from a single piece of evidence to reach the incorrect answer while ignoring the multi-hop reasoning involving other evidence. This observation also verifies the effectiveness of our system to stepwise generate the single-hop question grounded on the intermediate single-hop supporting sentences for interpretable multi-hop reasoning.

Reasoning Type	*ELECTRA*		*StepReasoner*	
	EM	F1	EM	F1
Bridge (34%)	47.30	76.43	48.37	76.54
Implicit-Bridge (29%)	37.04	68.66	39.81	69.65
Comparison (20%)	61.77	79.13	63.04	79.13
Intersection (17%)	42.97	73.89	46.23	75.07

Table 5: Results of Joint EM and Joint F1 across different reasoning types. The numbers in parentheses are percentages of different types.

Comparison of Different Single-hop Question Generation Methods To manifest the effectiveness of our generated single-hop questions based on identified single-hop supporting sentences, we incorporate several various single-hop question generation approaches into our stepwise reasoning framework and compare the QA results. Table 6 shows that our *Supp-based* method performs best. It reveals that our single-hop question generation is grounded on single-hop supporting sentences to generate more accurate and informative sub-questions, which are more effective than single-hop questions constructed by other strategies.

Method	Answer		Sup Fact		Joint	
	EM	F1	EM	F1	EM	F1
Span-based	68.60	81.66	62.31	87.41	46.26	73.20
USeq2Seq	69.29	82.22	63.11	88.00	46.96	74.08
Supp-based	70.11	83.03	64.27	88.10	48.55	74.85

Table 6: Comparison of various single-hop question generation methods. *Span-based* represents the sub-question generation **based** on **span** prediction in *DecompRC* (Min et al., 2019) while *USeq2Seq* is the **U**nsupervised **seq2seq** decomposition in *ONUS* (Perez et al., 2020). *Supp-based* is our **supp**orting sentences **based** single-hop question generation.

Case Study An example of the "Bridge" type question is presented in Figure 3 to show the inter-

pretable reasoning process of *StepReasoner* compared to other decomposition based methods. Our system successfully identifies the first-hop supporting sentences and generates the first-hop sub-question to query the escaping location. Then the first-hop sub-answer helps to identify the following supporting sentences to finally predict the correct answer. The second-hop question is also generated for better illustration. By contrast, *DecompRC* fails to decompose this complex question while *ONUS* generates an improper first-hop question, and they both predict a wrong answer. More cases of the other reasoning types are in Appendix B.

Question: Sparking the Marian civil war, who helped the recently abdicated queen to escape her imprisonment?
Answer: the Queen's gaoler

 # StepReasoner #

Sub-S1: The Marian civil war in Scotland (1568-1573) was a period of conflict which followed the abdication of Mary, Queen of Scots, and her escape from Loch Leven Castle in May 1568.
Sub-Q1: Where did the queen escape from during her abdication?
Sub-A1: Loch Leven Castle

Sub-S2: Loch Leven Castle is a ruined castle on an island in ... Queen of Scots was imprisoned here in 1567-1568, and forced to abdicate as queen, before escaping with the help of **her gaoler's family**.
Sub-Q2: Who helped the the queen to escape her imprisonment?

 # DecompRC #
Sub-Q: Sparking the Marian civil war, who helped the recently abdicated queen to escape her imprisonment?

 # ONUS #
Sub-Q1: Why did sparking the Marian Civil War?
Sub-Q2: Who helped the recently abdicated queen escape her imprisonment?

Wrong Predicted Answer: Loch Leven Castle

Figure 3: A case of the reasoning process by our *StepReasoner* compared to *DecompRC* and *ONUS*. The green phrase denotes our predicted answer and the texts in shadow support the single-hop question generation.

4 Related Work

Multi-hop question answering aims to aggregate multiple pieces of documents to model the multi-hop reasoning chain and predict the answer (Khashabi et al., 2018; Yang et al., 2018; Welbl et al., 2018; Ho et al., 2020; Fei et al., 2022). Prior methods mainly focus on utilizing a single reader to model the interaction between the question and relevant context, and simultaneously or separately predict the supporting sentences and answer within one step. Dhingra et al. (2018) and Qiu et al. (2019) propose to construct graphs based on entity information from scattered paragraphs and utilize graph neural networks as the reader to reason out the answer. Then HDE-Graph (Tu et al., 2019), HGN (Fang et al., 2019) and SAE (Tu et al., 2020) enrich information in the entity graph by extending nodes of other granularity to build a hierarchical graph and improve the interaction between the question and context. Some other methods (Glass et al., 2019; Beltagy et al., 2020; Zaheer et al., 2020; Alkhaldi et al., 2021) adopt pre-trained models as the powerful reader for multi-hop reasoning and achieve promising results. However, these one-step reader methods directly encode the relevant context and question for answer prediction while neglecting to illustrate the explicit reasoning process.

To circumvent the interpretability limitation, another stream of research proposes to solve the multi-hop reasoning by multi-step question decomposition. Nishida et al. (2019) and Jiang and Bansal (2019) recurrently update the sub-query at each step to break down the problem but these sub-queries are learned in latent representations and not sufficiently explainable. Instead, some works explore to explicitly decompose the complex question into single-hop questions which assumes access to decomposition supervision (Min et al., 2019; Wolfson et al., 2020). To skip this reliance, (Perez et al., 2020) and (Khot et al., 2020) attempt to construct pseudo-decomposition from other simple question corpora which pose a new challenge of label noises. Besides, they only take as input the original question to generate single-hop questions without grounding on the supporting facts at each hop. In contrast, we design a stepwise reasoning framework to locate the single-hop supporting sentences at each step for generating more fact-grounded and informative single-hop sub-questions without any genuine or pseudo supervision, and integrate the sequential reasoning process into a unified multi-hop reader for more robust performance.

5 Conclusion

In this paper, we study the task of multi-hop question answering and propose to stepwise locate the single-hop supporting sentences and generate more fact-grounded single-hop questions for better interpretable multi-hop reasoning. We present a stepwise reasoning framework to incorporate both single-hop supporting sentence identification and the corresponding single-hop question generation for each intermediate step until inferring a final result. It employs a pre-trained simple question generator and takes the identified single-hop supporting sentences as base to generate the single-hop question, which obviates the necessity of constructed supervision and helps generate more fact-based

single-hop questions. It utilizes a unified reader to jointly learn both intermediate hop reasoning and final hop inference for better fault tolerance. Experimental results validate the general effectiveness and interpretability of our *StepReasoner*.

Acknowledgments

This work is partially supported by Natural Science Foundation of China (No.6217020551, 71991471), National Social Science Foudation of China (No. 20ZDA060), Science and Technology Commission of Shanghai Municipality Grant (No.20dz1200600, 21QA1400600, 21511101000) and Zhejiang Lab (No. 2019KD0AD01).

References

Tareq Alkhaldi, Chenhui Chu, and Sadao Kurohashi. 2021. Flexibly focusing on supporting facts, using bridge links, and jointly training specialized modules for multi-hop question answering. *TASLP*.

Iz Beltagy, Matthew E Peters, and Arman Cohan. 2020. Longformer: The long-document transformer. *arXiv preprint arXiv:2004.05150*.

Kevin Clark, Minh-Thang Luong, Quoc V Le, and Christopher D Manning. 2020. Electra: Pre-training text encoders as discriminators rather than generators. In *Proc. of ICLR*.

Bhuwan Dhingra, Qiao Jin, Zhilin Yang, William W Cohen, and Ruslan Salakhutdinov. 2018. Neural models for reasoning over multiple mentions using coreference. In *Proc. of NAACL*.

Yuwei Fang, Siqi Sun, Zhe Gan, Rohit Pillai, Shuohang Wang, and Jingjing Liu. 2019. Hierarchical graph network for multi-hop question answering. In *Proc. of EMNLP*.

Zichu Fei, Qi Zhang, Tao Gui, Di Liang, Sirui Wang, Wei Wu, and Xuanjing Huang. 2022. CQG: A simple and effective controlled generation framework for multi-hop question generation. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 6896–6906, Dublin, Ireland. Association for Computational Linguistics.

Ruiliu Fu, Han Wang, Xuejun Zhang, Jun Zhou, and Yonghong Yan. 2021. Decomposing complex questions makes multi-hop QA easier and more interpretable. In *Findings of the Association for Computational Linguistics: EMNLP 2021*, pages 169–180, Punta Cana, Dominican Republic. Association for Computational Linguistics.

Ruiliu Fu, Han Wang, Jun Zhou, and Xuejun Zhang. 2022. Na-reviewer: Reviewing the context to improve the error accumulation issue for multi-hop qa. *Electronics Letters*, 58(6):237–239.

Michael Glass, Alfio Gliozzo, Rishav Chakravarti, Anthony Ferritto, Lin Pan, GP Bhargav, Dinesh Garg, and Avirup Sil. 2019. Span selection pre-training for question answering. In *Proc. of ACL*.

Xanh Ho, Anh-Khoa Duong Nguyen, Saku Sugawara, and Akiko Aizawa. 2020. Constructing a multi-hop QA dataset for comprehensive evaluation of reasoning steps. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 6609–6625, Barcelona, Spain (Online). International Committee on Computational Linguistics.

Yichen Jiang and Mohit Bansal. 2019. Self-assembling modular networks for interpretable multi-hop reasoning. In *Proc. of EMNLP*.

Daniel Khashabi, Snigdha Chaturvedi, Michael Roth, Shyam Upadhyay, and Dan Roth. 2018. Looking beyond the surface: A challenge set for reading comprehension over multiple sentences. In *Proc. of NAACL*.

Tushar Khot, Daniel Khashabi, Kyle Richardson, Peter Clark, and Ashish Sabharwal. 2020. Text modular networks: Learning to decompose tasks in the language of existing models. In *Proc. of NAACL*.

Zhenzhong Lan, Mingda Chen, Sebastian Goodman, Kevin Gimpel, Piyush Sharma, and Radu Soricut. 2019. Albert: A lite bert for self-supervised learning of language representations. In *Proc. of ICLR*.

Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Ves Stoyanov, and Luke Zettlemoyer. 2019. Bart: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. In *Proc. of ACL*.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Roberta: A robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692*.

Sewon Min, Victor Zhong, Luke Zettlemoyer, and Hannaneh Hajishirzi. 2019. Multi-hop reading comprehension through question decomposition and rescoring. In *Proc. of ACL*.

Kosuke Nishida, Kyosuke Nishida, Masaaki Nagata, Atsushi Otsuka, Itsumi Saito, Hisako Asano, and Junji Tomita. 2019. Answering while summarizing: Multi-task learning for multi-hop qa with evidence extraction. *arXiv preprint arXiv:1905.08511*.

Liangming Pan, Wenhu Chen, Wenhan Xiong, Min-Yen Kan, and William Yang Wang. 2020. Unsupervised multi-hop question answering by question generation. *arXiv preprint arXiv:2010.12623*.

Ethan Perez, Patrick Lewis, Wen-tau Yih, Kyunghyun Cho, and Douwe Kiela. 2020. Unsupervised question decomposition for question answering. In *Proc. of EMNLP*.

Lin Qiu, Yunxuan Xiao, Yanru Qu, Hao Zhou, Lei Li, Weinan Zhang, and Yong Yu. 2019. Dynamically fused graph network for multi-hop reasoning. *ACL*.

Pranav Rajpurkar, Jian Zhang, Konstantin Lopyrev, and Percy Liang. 2016. Squad: 100,000+ questions for machine comprehension of text. *EMNLP*.

Nan Shao, Yiming Cui, Ting Liu, Shijin Wang, and Guoping Hu. 2020. Is graph structure necessary for multi-hop reasoning? In *Proc. of EMNLP*.

Alon Talmor and Jonathan Berant. 2018. The web as a knowledge-base for answering complex questions. *arXiv preprint arXiv:1803.06643*.

Ming Tu, Kevin Huang, Guangtao Wang, Jing Huang, Xiaodong He, and Bowen Zhou. 2020. Select, answer and explain: Interpretable multi-hop reading comprehension over multiple documents. In *Proc. of AAAI*.

Ming Tu, Guangtao Wang, Jing Huang, Yun Tang, Xiaodong He, and Bowen Zhou. 2019. Multi-hop reading comprehension across multiple documents by reasoning over heterogeneous graphs. In *Proc. of ACL*.

Johannes Welbl, Pontus Stenetorp, and Sebastian Riedel. 2018. Constructing datasets for multi-hop reading comprehension across documents. *TACL*.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Rémi Louf, Morgan Funtowicz, et al. 2019. Transformers: State-of-the-art natural language processing. *ArXiv*, abs/1910.03771.

Tomer Wolfson, Mor Geva, Ankit Gupta, Matt Gardner, Yoav Goldberg, Daniel Deutch, and Jonathan Berant. 2020. Break it down: A question understanding benchmark. *TACL*.

Zhilin Yang, Peng Qi, Saizheng Zhang, Yoshua Bengio, William W Cohen, Ruslan Salakhutdinov, and Christopher D Manning. 2018. Hotpotqa: A dataset for diverse, explainable multi-hop question answering. In *Proc. of EMNLP*.

Manzil Zaheer, Guru Guruganesh, Kumar Avinava Dubey, Joshua Ainslie, Chris Alberti, Santiago Ontanon, Philip Pham, Anirudh Ravula, Qifan Wang, Li Yang, et al. 2020. Big bird: Transformers for longer sequences. In *Proc. of NeurIPS*.

A Training Details

All these models are implemented using Huggingface (Wolf et al., 2019). For HotpotQA, we use a batch size of 48 and fine-tune for 10 epochs with the learning rate 3e-5. For 2WikiMultiHopQA, the batch size is set to 24, the number of training epochs is 5 and the learning rate is 5e-5. The Adam is taken as the optimizer and we use a linear learning rate scheduler with 10% warmup proportion. The proposed systems and other comparison models are trained on 4 NVIDIA Tesla V100 GPUs.

B Case Study of Different Reasoning Types

We further present three cases of other reasoning types in Figure 4, including "Implicit-Bridge", "Comparison" and "Intersection". We can see that the *StepReasoner* generates high-quality decompositions for better interpretable multi-hop reasoning and predict accurate answers for all types compared to previous question decomposition based methods.

For the "Implicit-Bridge" question in Figure 4a, by first predicting the supporting sentences related to "Sivarama Swami" at the first hop, we can generate a sub-question to identify the implicit bridge "Bhaktivedanta Manor" for location query in the second hop. Although our predicted answer is different from the ground truth, it is also a reasonable response and more close to the golden one compared to the predictions of *DecompRC* and *ONUS*. These two methods both fail to decompose the multi-hop question and can only predict an intermediate answer.

For the "Comparison" and "Intersection" questions in Figure 4b and 4c, all methods predict the correct answers. However, we can generate more diverse single-hop sub-questions without requesting for any supervision, such as "die" and "death" for representing "pass away", and "belong to" for "from". By contrast, the decompositions by *DecompRC* are usually inflexibly from the original questions and the unsupervised *ONUS* creates improper sub-questions with noises. We hope that combining our generated single-hop questions can also help to construct more natural and diverse multi-hop questions and further promote multi-hop reasoning performance.

Implicit-Bridge

Question: Where does Sivarama Swami conduct courses on Vaishnava Theology?
Answer: in the village of Aldenham

 # StepReasoner #

Sub-S1: Sivarama Swami (born 30 March 1949, Budapest, Hungary) is a Vaishnava guru and a religious leader … . He has been conducting courses at Bhaktivedanta Manor on his own commentaries to … Vaishnava Theology.
Sub-Q1: Which manor does Sivarama Swami conduct courses on Vaishnava?
Sub-A1: Bhaktivedanta Manor

Sub-S2: Bhaktivedanta Manor is a Gaudiya Vaishnava temple set **in the Hertfordshire countryside of England**, in the village of Aldenham near Watford.
Sub-Q2: What section does Bhaktivedanta Manor belong to?

 # DecompRC #
Sub-Q: Where does Sivarama Swami conduct courses on Vaishnava Theology?

 # ONUS #
Sub-Q1: Where does Sivarama Swami conduct courses on Vaishnava Theology?
Sub-Q2: What is the nationality of the child?

Predicted Answer: Bhaktivedanta Manor

(a) An example of "Implicit-Bridge" reasoning type.

Comparison

Question: Who passed away first Robert Graves or Dino Buzzati?
Answer: Dino Buzzati-Traverso

 # StepReasoner #

Sub-S1: Robert von Ranke Graves (24 July 1895 - 7 December 1985), also known as Robert Ranke Graves and most commonly Robert Graves, was an English poet, novelist, critic and classicist.
Sub-Q1: When did Robert Graves die?
Sub-A1: 7 December 1985

Sub-S2: **Dino Buzzati**-Traverso (14 October 1906 - 28 January 1972) was an Italian novelist, short story writer, painter and poet, as well as a journalist … .
Sub-Q2: What was Dino Buzzati-Traverso's death date?

 # DecompRC #
Sub-Q1: Robert Graves passed away when?
Sub-Q2: Dino Buzzati passed away when? ***Sub-Q3:*** SMALLER
Predicted Answer: Dino Buzzati

 # ONUS #
Sub-Q1: When was dino buzzati born? ***Sub-Q2:*** Who passed away first?
Predicted Answer: Dino Buzzati

(b) An example of "Comparison" reasoning type.

Intersection

Question: What family are the genus' Sinofranchetia and Stauntonia from?
Answer: a genus of flowering plant in the Lardizabalaceae family

 # StepReasoner #

Sub-S1: Stauntonia is a genus of flowering plant in the Lardizabalaceae family.
Sub-Q1: What family does the Stauntonia belong to?
Sub-A1: Lardizabalaceae

Sub-S2: Sinofranchetia is a genus of flowering plant in the **Lardizabalaceae family**.
Sub-Q2: What family does the Sinofranchetia belong to?

 # DecompRC #
Sub-Q1: What family is the genus ' Sinofranchetia from?
Sub-Q2: What family is the genus ' Stauntonia from?
Sub-Q3: INTERSEC
Predicted Answer: Lardizabalaceae

 # ONUS #
Sub-Q1: What family are the genus ' Sinofranchetia?
Sub-Q2: Where is Stauntonia from?
Predicted Answer: Lardizabalaceae

(c) An example of "Intersection" reasoning type.

Figure 4: Three cases of other reasoning types. The green phrases denote our predicted answers and the texts in shadow support the corresponding single-hop question generation.

Less is Better: Recovering Intended-Feature Subspace to Robustify NLU Models

Ting Wu
Fudan Univerisity
tingwu21@m.fudan.edu.cn

Tao Gui *
Fudan Univerisity
tgui@fudan.edu.cn

Abstract

Datasets with significant proportions of *bias* present threats for training a trustworthy model on NLU tasks. Despite yielding great progress, current debiasing methods impose excessive reliance on the knowledge of bias attributes. Definition of the attributes, however, is elusive and varies across different datasets. Furthermore, leveraging these attributes at input level to bias mitigation may leave a gap between intrinsic properties and the underlying decision rule. To narrow down this gap and liberate the supervision on bias, we suggest extending bias mitigation into feature space. Therefore, a novel model, **R**ecovering **I**ntended-Feature **S**ubspace with **K**nowledge-Free (RISK) is developed. Assuming that shortcut features caused by various biases are unintended for prediction, RISK views them as redundant features. When delving into a lower manifold to remove redundancies, RISK reveals that an extremely low-dimensional subspace with *intended features* can robustly represent the highly biased dataset. Empirical results demonstrate our model can consistently improve model generalization to out-of-distribution set, and achieves a new state-of-the-art performance [1].

1 Introduction

Pretrained language models have achieved remarkable performance on a wide range of natural language understanding (NLU) benchmarks (Devlin et al., 2019). However, when encountering more challenging test sets, they dramatically fail (McCoy et al., 2019). Studies indicate such a dilemma is mainly rooted in the model's reliance on specific *dataset biases* (Gururangan et al., 2018; Zhang et al., 2019a; Schuster et al., 2019), which correlate well with labels but not for the intended underlying task. For instance, on the natural language inference (NLI) task, models tend to use negation cues ("not", "no", etc.), for a *Contradiction* prediction, whereas a learner intended to learn the *underlying correlation* based on the context semantics.

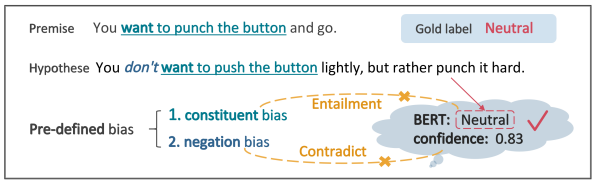

Figure 1: A toy example that illustrate bias in MNLI-matched dev set. BERT's prediction *Neutral* does not comply with the assumed decision rule (Entailment, Contradiction) caused by pre-defined bias.

To train a NLU model that captures the *underlying correlation* from biased datasets, current approaches focus on how to leverage kinds of supervision effectively. One of the most popular forms of such supervision is to explicitly construct a bias-only model under human annotations, e.g., a hypothesis-only model for NLI task, and factor it out from the main model through ensemble-based training (Clark et al., 2019; Utama et al., 2020a). Another empirical line of research shifts supervision from bias type annotations to weak model learners. They find models with limited capacity (Clark et al., 2020; Sanh et al., 2021) or training on limited dataset (Utama et al., 2020b) prone to extract shortcut patterns first, the observation of which can be utilized to mitigate dataset bias.

Despite the supervision on bias has shown effectiveness in bias mitigation, the fundamental questions remain unsolved. On one hand, acquirement of supervision on the bias either from human knowledge or model learning behaviours, is often a laborious and expensive cost. Moreover, considering the definition of bias attributes is elusive and varies across datasets, the external

*Corresponding author.
[1]Our code and data are available at https://github.com/CuteyThyme/RISK.git.

Proceedings of the 29th International Conference on Computational Linguistics, pages 1666–1676
October 12–17, 2022.

knowledge can not cover all types of biases in the dataset, leaving potential bias underexplored, e.g., bias beyond the definition or bias harder to learn. On the other hand, capturing bias only at the input examples is just like a black box, being oblivious to the intrinsic properties that drives model to make prediction. The toy example shown in Figure 1 reveals that predefined bias does not necessarily lead the model to learn the unintended decision rule (i.e., constituent bias triggers an Entailment prediction, negation bias triggers an Contradiction prediction). Hence, current debiasing methods inevitably fall short in above two limitations.

On account of the consensus that shortcut features induced by biased examples are detrimental for prediction, various kinds of biases can thus be equivalently viewed as redundancies. When delving into feature space, closer to the decision rule to remove these redundant features, supervision of the attributes from biased examples can be liberated as well. Therefore, we develop a novel model, **R**ecovering **I**ntended-Feature **S**ubspace with **K**nowledge-Free (RISK). Aimed with purifying redundancies from feature space, RISK reveals that for a highly biased dataset, a small subset of *informative* and *shared* features, i.e. intended ones, can give rise to a robust prediction. Concretely, RISK maps features into a lower manifold and learns an orthogonal projector spanned by *geometric median subspace* to recover the intended-feature subspace in an end-to-end manner.

Experimental results on three NLU tasks show RISK outperforms other methods by a large margin, indicating its potential to mitigate bias and the prerequisites of supervision on biased attributes can be liberated. Moreover, when transferring to more challenging out-of-distribution set, RISK can consistently improve the robustness of NLU models. To sum up, our contributions are three-fold as follows:

• We propose a novel *feature-based* debiasing model, termed as RISK. RISK is the initial attempt that free of the supervision on bias attributes.

• We reveal shortcut features as part of redundancy, and thus only leveraging the informative features shared across biased and bias-free examples can achieve the goal of bias mitigation.

• We conduct extensive experiments to validate the effectiveness of RISK in mitigating bias. Moreover, RISK exhibits great power to generalize to more challenging scenarios, showing its potential to robustify NLU models.

2 Bias Mitigation As Feature Redundancy

2.1 Problem Setup

Given training dataset $\mathcal{D} = \{x_i, y_i\}_{i=1}^N$ including C classes, a NLU task requires the model to understand the semantic of input text x_i and then predict the target label y_i. Generally, the model is composed of a feature extractor $\mathcal{F}(\cdot) : \mathcal{F}(\mathbf{x}) \rightarrow \mathbf{z}$ and a linear classifier $g(\cdot) : g(\mathbf{z}) \rightarrow \hat{\mathbf{y}}$.

When the model is trained on a highly biased dataset, it will easily capture shortcut features in high-dimensional $\mathbf{z}$. Since the shortcut features are the unintended ones that induce predictions, we treat them as a kind of redundancy. Therefore, mitigating dataset bias can be subsumed under minimizing redundancy in feature space.

2.2 Feature Redundancy by Subspace Modeling

In statistical machine learning, feature subspace paves a path towards eliminating redundant features, as it sheds light on projecting high-dimensional feature onto one subspace, which can significantly capture its most significant information. A common formulation for subspace modeling is to find an orthogonal projection $\mathbf{P}$ of dimension d whose subspace can robustly represents the input features (Vaswani et al., 2018). Let $\mathbf{I}$ denote the identity matrix in the ambient space of the high-dimensional feature $\mathbf{z}$, and the least q-th power deviations formulation for $q > 0$ seeks $\mathbf{P}$ that minimizes:

$$\mathcal{L}(\mathbf{P}) = \sum_{i=1}^{N} \left\| (\mathbf{I} - \mathbf{P}) z_i \right\|_2^q \tag{1}$$

Classically, taking $q = 2$ results in principal component analysis(PCA), which finds the orthogonal directions of maximum variance:

$$\mathcal{L}(\mathbf{P}) = \sum_{i=1}^{N} \left\| (\mathbf{I} - \mathbf{P}) z_i \right\|_2^2$$

2.3 Geometric Median Subspace as solution.

However, even approximate minimization of Eq. 1 is nontrivial, since it has been shown to be NP hard for $1 \leq q < 2$, furthermore, $q < 1$ can result in a wealth of local minima. Literature have theoretically proven the preferable minimization

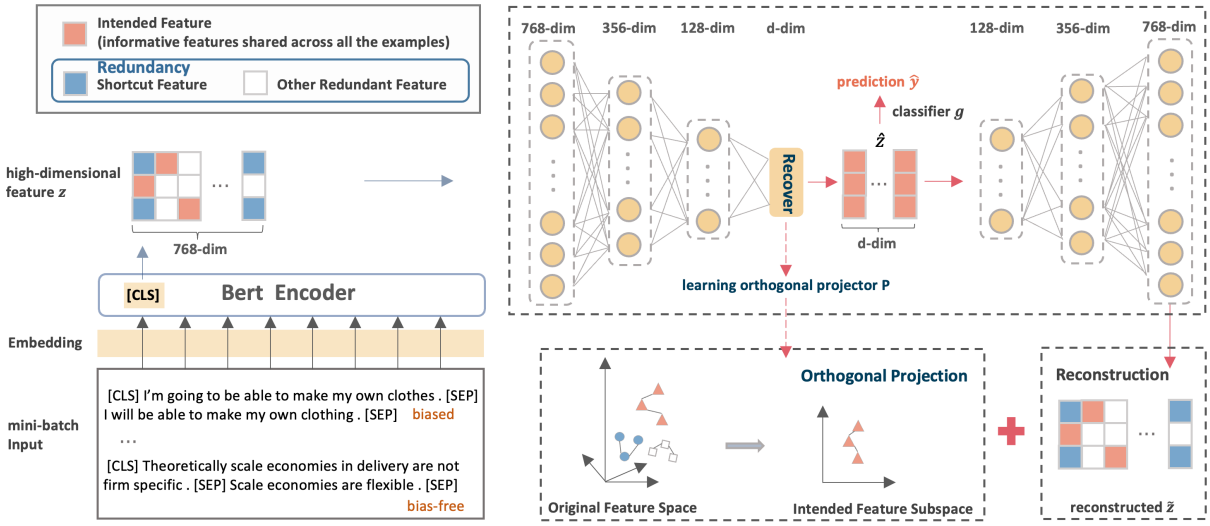

Figure 2: Model Architecture of RISK.

is $q = 1$ (Osborne and Watson, 1985; Nyquist, 1988), and thus equals to replace the least squares formulation in PCA with least absolute deviations as follows:

$$\mathcal{L}(\mathbf{P}) = \sum_{i=1}^{N} \left\| (\boldsymbol{I} - \boldsymbol{P}) z_i \right\|_2^1 \tag{2}$$

A nice interpretation of the minimizer of above equation is a *Geometric Median Subspace* (Fletcher et al., 2009), analogous geometric median in modeling centers of input features. Ideally, once we solve a orthogonal projection spanned by this geometric median space, we can achieve an robust estimation of all input features. Since the shortcut features are not shared by bias-free examples, they will be removed automatically as redundancy.

3 RISK: Feature-based Debiasing Without Supervision on Bias

Guided by the theoretical subspace modeling discussed above, in this section, we illustrate the detailed implementation of RISK. In practice, we adopt autoencoder as the main architecture. Leveraging autoencoder to map features into a lower manifold is the first stage of removing redundant features. We further add a simple but effective Recovery Layer within autoencoder to learn a orthogonal projection $\mathbf{P}$, leaving the shared informative features to perform final predictions.

3.1 Delving into Feature Space

We use BERT $\mathcal{F}_\theta$ to map each textual data point x_i into a high-dimensional feature space, that is, $\mathbf{z} = \mathcal{F}(\mathbf{x}, \theta)$. To be specific, $\mathbf{z}$ corresponds to [CLS] token embedding the last layer BERT outputs. It has been convinced that embeddings from pre-trained language models contain much redundency for down-stream NLU tasks (Dalvi et al., 2020). As for highly biased dataset, $\mathbf{z}$ will easily capture substantial shortcut ones. We thus categorize $\mathbf{z}$ into following two feature types:

Intended Features are the *informative* features *shared* across biased and bias-free examples.

Redundant Features include shortcut features that only correlate well with labels, and other redundant features (e.g., task-irrelevant, task-relevant but non-robust ones).

3.2 Autoencoder: To Be Informative Features

For the first stage of mitigating feature redundancy as a way of bias mitigation, we opt to employ an encoder $\mathscr{E}$ composed of a three-layer MLP to map $\mathbf{z}$ into a lower manifold.

Reconstruction loss. As shown in Figure 2, to be symmetric of the encoder $\mathscr{E}$ that project $\mathbf{z}$ into a lower manifold, we also train a decoder $\mathscr{D}$ that map $\hat{\mathbf{z}}$ into $\tilde{\mathbf{z}}$, i.e., the reconstruction representation of $\mathbf{z}$, formulating a bottleneck autoencoder as result. The reconstruction loss is thus defined as:

$$\mathcal{L}_{\text{recon}} = \sum_{i=1}^{N} \left\| z_i - \mathscr{D}(\hat{z}_i) \right\|_2^2 \tag{3}$$

The reconstruction term is used to ensure that a good reconstruction of the original feature can be obtained by using the learned low-dimensional subspace features. Notably, as we defined, *informative* is one of the key characteristics of

the intended features. We can further prove that minimize the reconstruction error can serve as maximizing the lower bound of the mutual information between $\mathbf{z}$ and $\hat{\mathbf{z}}$. In general $\hat{\mathbf{z}}$ is not an exact reconstruction of $\mathbf{z}$, but rather in probabilistic terms as the mean of a distribution $p = (Z|\hat{Z} = \hat{\mathbf{z}})$, this yields an associated reconstruction error (Vincent et al., 2010) to be optimized:

$$\mathcal{L}_{\text{recon}} \propto -\log p(\mathbf{z}|\hat{\mathbf{z}})$$

In conjuction with it, minimizing the reconstruction loss actually carry the following optimization:

$$\min\mathbb{E}[\mathcal{L}_{\text{recon}}(\mathbf{z}, \hat{\mathbf{z}})] = \max\mathbb{E}[log\mathbb{P}(\mathbf{z}|\hat{\mathbf{z}})] \quad (4)$$

Maximizing the expectation of the conditional probabilty $\mathbb{E}[log\mathbb{P}(\mathbf{z}|\hat{\mathbf{z}})]$ is equivalent to maximizing the mutual information between $\mathbf{z}$ and $\hat{\mathbf{z}}$ (Chen et al., 2022). This promises the subspace where $\hat{\mathbf{z}}$ lies in is informative and task-relevant for downstream task.

3.3 Recovery Layer: To Be Shared Features

In fact, only utilizing autoencoder can not promise the latent subspace as the intended one we defined before, since shortcut features dominated by biased examples also contain useful but not robust information for prediction. Therefore, going a step further to remove redundant features is needed.

Projection Loss. Leveraging the core idea of subspace modeling, *geometric median subspace* is a preferable minimum to solve the shared features in ideal. In this way, we can recast the problem into learning an orthogonal projector spanned by such median subspace. With the expansion of Eq. 2, the following projection loss function can be achieved:

$$\begin{aligned}\mathcal{L}_{\text{proj}}(\mathbf{A}) = \lambda_1 \sum_{i=1}^{N} \left\| z_i - \mathbf{A}^\top \mathbf{A} z_i \right\|_2^1 \\ + \lambda_2 \left\| \mathbf{A}\mathbf{A}^\top - \mathbf{I}_d \right\|_F^2\end{aligned} \quad (5)$$

we use $\mathbf{A}$ to denote the transformation that reduces feature dimension to d, and $\mathbf{A}^\top$ denotes the transpose of $\mathbf{A}$, $\mathbf{I}_d$ denotes the $d \times d$ identity matrix and $\left\| \cdot \right\|_{\text{F}}$ denotes the Frobenius norm. Here λ is an hyperparameter represent the weight of the projection loss to the whole learning objective, for the simplicity, we let $\lambda_1 = \lambda_2$. We later show it associates with the dataset characteristics in Sec. 5.1.

It can be noted that the first term in the weighted sum of above loss function is close to Equation 2 as long as $\mathbf{A}\mathbf{A}^\top$ is close to an orthogonal projector. To enforce this requirement, we introduce the second term that imposes the nearness of $\mathbf{A}\mathbf{A}^\top$ to an orthogonal projection.

Practically, the transformation $\mathbf{A}$ is implemented as a linear MLP layer within the autoencoder $\mathscr{E}$, coined as the Recovery Layer. By applying the projection loss, the parameters of the trained Recovery Layer can approximate the minimal result of Eq. 2. In a sense, the Recovery Layer can be considered as bridging the connections between statistical machine learning and DNN.

3.4 Predictors Fitting in the Intended-Feature Subspace

Intuitively, as a robust model to defend against various distribution shift, it is expected to learn an optimal predictor g, which relies on only the intended features most relevant to current task to make predictions. So for the final step, we just fit a linear classifier in the recovered subspace:

$$g(\hat{\mathbf{z}}) = W^\top \hat{\mathbf{z}} + b$$

Along with minimizing the cross entropy between $g(\hat{\mathbf{z}})$ and $\mathbf{y}$, the final learning objective of RISK is summed into:

$$\mathcal{L}_{\text{RISK}} = \mathcal{L}_{\text{CE}} + \mathcal{L}_{\text{recon}} + \mathcal{L}_{\text{proj}}$$

With the dual regularization of reconstruction loss and projection loss, we therefore promise the intended-feature subspace is de facto informative and shared.

4 Experiments

In this section, we provide comprehensive analysis on RISK through extensive experiments on three NLU tasks, and compare out-of-distribution as well as in-distribution accuracy of RISK with other debiasing methods to demonstrate its strength.

4.1 Tasks and Biased Datasets

We evaluate our approach on three NLU tasks: natural language inference (NLI), fact verification, and paraphrase identification.

Natural Language Inference aims to determine whether a premise sentence entails a hypothesis sentence. We use the MNLI dataset (Williams et al., 2018) for training, nevertheless, recent studies

indicate that models trained on these NLI datasets tend to adopt shallow heuristics(e.g., lexical overlap, hypothesis-only) to predict (Gururangan et al., 2018; Poliak et al., 2018). Based on the findings, HANS(Heuristic Analysis for NLI Systems, McCoy et al. (2019)) is designed to contain many examples where the heuristics fail, and we condider it as the challenging set for evaluation.

Fact Verification requires models to validate a claim in the context of evidence. For this task, we use the training dataset provided by the FEVER challenge (Thorne et al., 2018). Studies show that models ignoring evidence can still achieve high accuracy on FEVER, accordingly, Fever-Symmetric dataset (Schuster et al., 2019) is used as the test sets for evaluation.

Paraphrase Identification is designed to identify whether a pair of sentences have the same thing. We train the models on QQP (Iyer and Csernai, 2017), a widely used dataset for the task. Similarly to MNLI, models trained on QQP are inclined to mark any sentence pairs with high word overlap as paraphrases despite clear clashes in meaning. As for the balance with respect to the lexical overlap heuristic in PAWS(Paraphrase Adversaries from Word Scrambling, Zhang et al. (2019b)) , we use it as our out-of-distribution set.

4.2 Baseline Methods

We compare RISK against seven debiasing models either with bias known or unknown. As for the bias-known models, supervision on bias is mainly the bias type, and for the bias-unknown models, supervision comes from a shallow model. For all these baseline methods, we adopt the BERT-base model (Devlin et al., 2019) as the main model.

Bias-known-prior Models. i). Reweighting (Clark et al., 2019) trains on a weighted version of the data to encourage the main model to focus on examples the bias-only model gets wrong. **ii). Product-of-Experts** (Clark et al., 2019) forces the main model to focus on learning from examples that are not predicted well by the bias-only model via logit ensembling. **iii). Learned-Mixin** (Clark et al., 2019) further improves this ensemble-based method by parameterizing the ensembling operation, allowing the main model to learn when to incorporate the output from the bias-only model for the ensembled prediction. **iv). Conf-reg** (Utama et al., 2020a) presents a novel *confidence regularization* method that encourage the main model to make predictions with lower confidence on examples that contained biased features.

Bias-known-free Models. For this line of research, models can bypass the need of hand-engineered bias-specific structures since a shallow model is utilized to identify biased examples automatically. **v). Self-debiasing** (Utama et al., 2020b) observe that BERT-base trained on a small subset of the training dataset can grasp the distribution of biased examples. **vi). Weak Learner** (Sanh et al., 2021) view models with limited capacity, i.e. Tiny-BERT (Turc et al., 2020), as the shallow one to obtain biased features. **vii). BERT+$\mathcal{F}_{\text{BiLSTM}}$** (Yaghoobzadeh et al., 2021) employ example forgettting to find minority examples, and robustify the model by fine-tuning twice, first on the full training data and second on the minorities only.

4.3 Implementation Details

For each task, we utilize the training configurations that have been proven to work well in previous studies, that is, a learning rate of $5e^{-5}$ for MNLI and $2e^{-5}$ for FEVER and QQP, and choose AdamW as optimizer with a weight decay of 0.01. For fair comparison, we keep the same bias-only model for all the ensemble-based baselines. To tackle the high performance variance on test sets as observed by Clark et al. (2019), we run each experiment five times and report the mean accuracy scores.

As for the autoencoder, our multiple experiments reveal that when make sure the bottleneck architecture, the detailed dimension of each layer makes few differences. More implementation details such as λ, d selection can be found in Section 5.1.

4.4 Experimental Results

The extensive results of all the above mentioned methods are summarized in Table 1. The results on the original development and test sets of each task represent the in-distribution performance. Obviously, for all three tasks, RISK improves BERT-base by a large margin on the challenging test set. Moreover, it surpasses other baselines not only for the out-of-distribution test set, but also the in-distribution ones.

Out-of-distribution generalization and biases mitigation. The absence of explicit knowledge on bias attributes seemingly create a gap between

Model	MNLI			FEVER			QQP		
	ID	HANS	Δ	ID	Symm.	Δ	ID	PAWS	Δ
BERT-base	84.5	61.2	-	85.6	55.1	-	90.8	36.1	-
Reweighting	83.5	69.2	+8.0	84.6	61.7	+6.6	89.5	48.6	+12.5
Product-of-Experts	84.1	66.3	+5.1	82.3	62.0	+6.9	86.9	56.5	+20.4
Learned-Mixin	84.2	64.0	+2.8	83.3	60.4	+5.3	87.6	55.7	+19.6
Conf-reg	83.4	69.1	+7.9	86.4	60.5	+5.4	89.1	40.0	+3.9
Conf-reg$_{\mathbf{self-debias}}$ ♠	84.3	67.1	+5.9	87.6	60.2	+5.1	89.0	43.0	+6.9
Weak Learner	83.3	67.9	+6.7	85.3	58.5	+3.4	-	-	-
BERT+$\mathcal{F}_{\text{BiLSTM}}$	82.9	70.4	+9.2	86.5	61.7	+6.6	88.0	47.6	+11.5
RISK	**84.5**	**71.3**	**+10.1**	**88.3**	**63.9**	**+8.8**	**90.1**	**56.5**	**+20.4**
w/o Reconstruction Loss	84.2	69.2	+8.0	87.6	60.1	+5.0	90.5	50.6	+14.5
w/o Projection Loss	83.9	64.6	+3.4	86.5	57.7	+2.6	90.4	42.1	+6.0

Table 1: Model performance(accu.(%)) on in-distribution and corresponding challenge test set. ♠: Self-debiasing framework is implemented in conjunction with the bias-known-prior models, we select the version that achieves the best performance in the original paper, i.e., *Confidence Regularization* with annealing mechanism. "w/o Reconstruction Loss" represents RISK is trained without the regularization of reconstruction loss, and "w/o Projection Loss" represents RISK is trained without the regularization of projection loss.

the generalization ability of bias-known models and bias-unknown models. Though RISK furthur eliminate any supervision of specific bias signal, it still generalize well to the out-of-the distribution. To validate the effectiveness of RISK in mitigating bias, in Figure 3, we break down the results on HANS into three different heuristics that the dataset was built upon. The increase of the accuracy in comparison with BERT-base on the *non-entailment* category can reflect the degree to which this bais is removed. Although the overall accuracy of Conf-reg$_{\mathbf{self-debias}}$ on HANS is higher than that of Product-of-experts, as shown in Figure 3, it's debiasing capacity is actually the worst. However, RISK can do well in mitigating the three known biases, and is on par with Product-of-Experts, outperforming other baselines.

In-distribution performance retention. The mitigation of dataset bias often suffers from the trade-off between removing shortcut features and sacrificing in-distribution performance. Especially, on PAWS dataset, this trade-off becomes more pronounced. We can observe that previous methods all have a drop in in-distribution test set for MNLI and QQP, which can be attributable to their explicit omission of biased examples. In contrast, our method finds a balance point via intended-feature subspace, where the out-of-distribution performance is improved and the in-distribution is almost retained. For Fever, the in-distribution accuracy of RISK even increases compared to that of BERT-base.

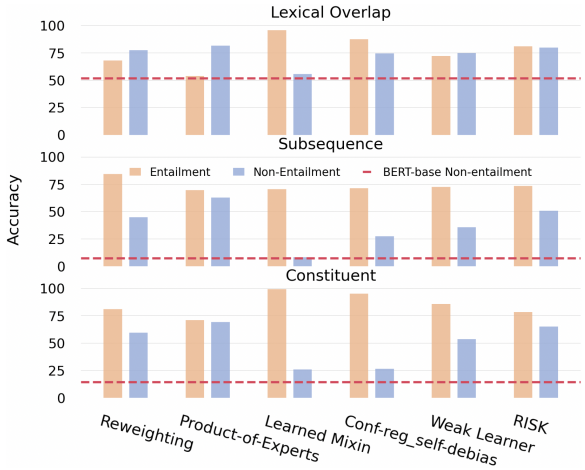

Figure 3: Performance of *RISK* and other baselines on the *entailment* and *non-entailment* categories for each heuristic(i.e., lexical overlap, subsequence and constituent) that HANS was designed to capture.

Ablation Studies. We assumed the reconstruction loss and projection loss are integral parts of RISK as they ensured the intended-feature subspace is *informative* and *shared*. To have an understanding of their impacts on the final performance respectively, we do the ablation studies, and results are shown in Table 1. Comparing the performance degradation, we can conclude that the projection loss plays a key role in helping mitigating dataset bias, and reconstruction loss can be viewed as a regularization that further bound the subspace to be more task-relevant to enhance the accuracy. As can be seen that faced with the removal of reconstruction loss or projection loss, in-

distribution performances of the three tasks remain little affected.

5 Analysis and Discussion

In this section, we construct supplementary experiments to further analyze RISK's effectiveness. Free of supervision on bias, we reveal that RISK can deal with more challenging scenarios.

5.1 Hyper-parameter Exploration

To recover the intended feature, we introduce two hyperparameters, the weight λ of projection loss and the subspace dimension d. During the grid search for a fine-grained tuning, we find the values of this two hyperparameters have a close connection with intrinsic properties of dataset.

(1). λ reflects the hardness of challenging set. In the process of optimizing λ, we observe that for the three tasks, RISK achieves best out-of-distribution performance with different value of λ. For the sake of having a qualitative understanding on the three out-of-distribution test set, we compare the average of sentence length and constituency parse tree height of example in HANS, Fever-Symmetric and PAWS respectively.

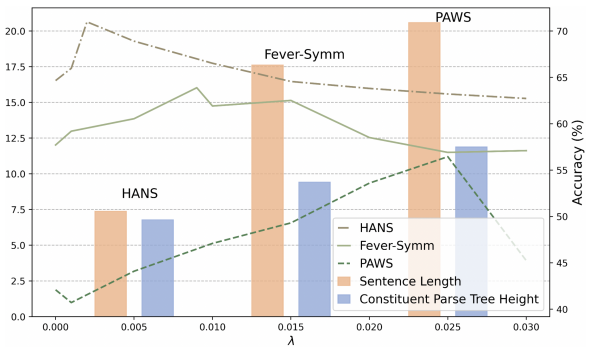

Figure 4: Bar chart represents average sentence length and constituent parse tree height of three out-of-distribution set. Line graph plots model performance with different λ.

As shown in Figure 4, we can observe that PAWS contains longer and syntactically more complex sentences. In contrast, HANS appears to be more easier for model to learn. Accordingly, easier HANS dataset requires a smaller weight of projection loss to obtain the best performance while PAWS requires a larger λ of 0.025. What's more, as for the harder patterns in PAWS for model to generalize, model performance on this task is more *sensitive* to the change of λ in a small range.

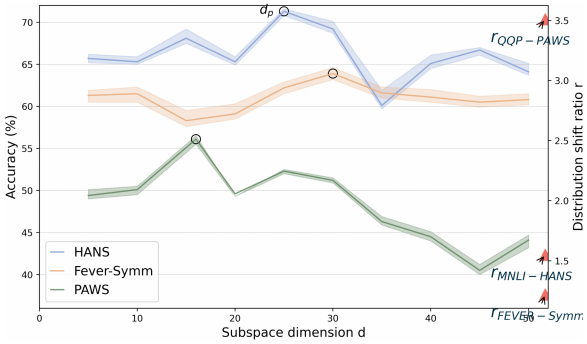

Figure 5: Model performance on HANS, Fever-Symm and PAWS with different subspace dimension d.

(2). d reflects the degree of distribution shift. To quantitatively describe the distribution shift, we propose *bias skewness* as an indicator of how biased a dataset is:

$$\text{bias skewness} = \frac{\#\text{ biased examples}}{\#\text{ bias} - \text{free examples}}$$

Thus, the ratio r of bias skewness between ID and OOD can mirror the distribution shift, the larger r, the greater distribution discrepancies. As shown in Figure 5, denote d_p as the optimal subspace dimension original training data set recovered to peak performance on the out-of-distribution set, and it turns out that $\text{PAWS}_{d_p} < \text{HANS}_{d_p} < \text{Fever-Symm}_{d_p}$. However, the ratio r reflects that $r_{\text{QQP}-\text{PAWS}} > r_{\text{MNLI}-\text{HANS}} > r_{\text{Fever}-\text{Symm}}$.

We can conclude that when faced with a larger distribution shift, the subspace dimension d on in-distribution training set should be smaller. In essence, d can be established a close connection with **intrinsic dimension** (Ansuini et al., 2019), i.e., the minimal number of parameters needed to represent a dataset. As our experiments reveal that a **16**-dimensional subspace with intended-features can represent the highly biased QQP training dataset well.

5.2 Transferability Analysis

We further examine the robustness of our approach along with other baselines by transferring to a more challenging scenario, training on MNLI but testing on Adversarial NLI. In our setting, Adversarial NLI contains not only human-crafted adversarial examples (Nie et al., 2020) but also those generated by textual adversarial attacks (TextFooler, Jin et al. (2020)). In general, models utilizing bias patterns that lack the ability to understand the underlying

semantics are vulnerable to be attacked. Results are summarized in Table 2 as follows.

Model	R1	R2	R3	ANLI-m
BERT-base	0	28.9	28.8	33.0
Product-of-Experts	**25.2**	27.5	31.3	53.8
Learned-Mixin	23.6	28.0	30.9	54.9
Conf-reg$_{\mathbf{self-debias}}$	21.8	27.4	31.0	48.5
RISK	25.1	**31.2**	**31.9**	**57.1**

Table 2: Model performance(accu.(%)) on adversarial MNLI. ANLI R1-R3 are challenging instances designed by human edition on input text. ANLI-m is adversarial MNLI-matched dataset generated by TextFooler based on blackbox BERT.

We can observe that vanilla BERT-base model trained on MNLI are vulnerable to those adversarial examples, especially ones generated by human edition, suggesting BERT relies overly on bias features to make predictions. On the other hand, either bias-known or bias-unknown models can more or less defend against these attacks. Compared to these baselines, RISK can consistently improve performance on all the adversarial test sets. This indicates the intended subspace has the power to robustify NLU models to various distribution shifts.

6 Related Work

We categorize the multiple lines of research devoted to mitigating dataset bias into three paradigms, in accordance with how the supervision is applied for bias mitigation.

6.1 Supervision from Bias Annotations

Concerns on robustness give rise to the discovery of a wide variety of biases in existing popular datasets, e.g., models make predictions only rely on the hypothesis in NLI datasets (Gururangan et al., 2018). Belinkov et al. (2019) utilize adversarial training to remove the known hypothesis-only features from model internal representations. Moreover, the understanding of specific dataset bias motivates the emergence of ensemble-based debiasing methods (Clark et al., 2019; He et al., 2019; Utama et al., 2020a) , which have shown promising improvements on the out-of-distribution performance. Generally, they view the known dataset biases as prior knowledge and design a simple bias-tailored model, namely the *bias-only model* and factor bias out of the *main model* through ensemble-based training. However, Xiong et al. (2021) theoretically prove that the inaccurate uncertainty estimations of the bias-only model can hurt the debiasing performance, and they propose to conduct calibration on the bias-only model.

6.2 Supervision from Model and Training

The excessive reliance on the assumption that specific types of biased features are known a-prior limits model's transferability. Correspondingly, this line of work seeks for the automatic identification of potentially biased examples, as their empirical results manifest that models with limited capacity (Clark et al., 2020; Sanh et al., 2021) or training on a fewer thousand examples (Utama et al., 2020b) exhibit different learning dynamics, and thus can be used to capture relatively shallow correlations.

Meanwhile, other observations have been made that a better use of minority examples(e.g., examples that are under-represented in the training distribution, or examples that are harder to learn) can play role in models' generalization as well. As Sagawa et al. (2020) point out, the fundamental reason leading to poor generalization lies in models' behaviour of *memorizing* the minority samples. Particularly, Tu et al. (2020) leverage the auxiliary tasks to help improve the generalization capability of pre-trained models on the minority groups. Yaghoobzadeh et al. (2021) propose to use *example forgetting* to find minority examples and make a second fine-tuning on those minorities.

6.3 Supervision from Augmentated Data

Data augmentation techniques have shown to be effective in regularizing models from overfitting to the training data(Novak et al., 2018). In this sense, when distribution shifts, the model will rely little on spurious correlations as a wider variety of predictive features are captured. This has attracted interest as a way to remove biases by explicitly modifying the dataset distribution(Min et al., 2020). Kaushik et al. (2020) and Srivastava et al. (2020) draw upon human-in-the-loop to augment existing training set with diverse and rich examples of potential unmeasured variables. Wang and Culotta (2021) further propose to automatically generate such counterfactual samples via a closet opposite matching strategy. Different from the augmentation of causal associations between features and classes, Wang et al. (2021) apply a cross-data analysis and knowledge-aware perturbations to identify spurious tokens on the stability of model predictions.

7 Conclusion

In this work, we shed light into feature subspace with the aim to create an underlying pathway — from the biased input examples to robust output prediction. Viewing shortcut features as redundancy, we construct a simple but effective Recovery Layer within the autoencoder structure for bias mitigation. Extensive experiments demonstrate the strengths of our model: better generalization, dataset-agnostic transferability and the robustness to more challenging scenarios. We believe this feature-based debiasing framework opens up new directions for establishing a trustworthy NLU model. Meanwhile, our concise motivation and implementation throw out a thought-provoking question, that is for model, for feature, sometimes less can be better.

Acknowledgements

The authors wish to thank the anonymous reviewers for their helpful comments. This work was partially National Natural Science Foundation of China (No. 62076069, 61976056), Shanghai Municipal Science and Technology Major Project (No.2021SHZDZX0103), and sponsored by CCF-Tencent Open Fund.

References

Alessio Ansuini, Alessandro Laio, Jakob H Macke, and Davide Zoccolan. 2019. Intrinsic dimension of data representations in deep neural networks. *Advances in Neural Information Processing Systems*, 32.

Yonatan Belinkov, Adam Poliak, Stuart Shieber, Benjamin Van Durme, and Alexander Rush. 2019. On adversarial removal of hypothesis-only bias in natural language inference. In *Proceedings of the Eighth Joint Conference on Lexical and Computational Semantics (*SEM 2019)*, pages 256–262, Minneapolis, Minnesota. Association for Computational Linguistics.

Jiahong Chen, Jing Wang, Weipeng Lin, Kuangen Zhang, and Clarence W de Silva. 2022. Preserving domain private representation via mutual information maximization. *arXiv preprint arXiv:2201.03102*.

Christopher Clark, Mark Yatskar, and Luke Zettlemoyer. 2019. Don't take the easy way out: Ensemble based methods for avoiding known dataset biases. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, Hong Kong, China. Association for Computational Linguistics.

Christopher Clark, Mark Yatskar, and Luke Zettlemoyer. 2020. Learning to model and ignore dataset bias with mixed capacity ensembles. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 3031–3045, Online. Association for Computational Linguistics.

Fahim Dalvi, Hassan Sajjad, Nadir Durrani, and Yonatan Belinkov. 2020. Analyzing redundancy in pretrained transformer models. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 4908–4926, Online. Association for Computational Linguistics.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, Minneapolis, Minnesota. Association for Computational Linguistics.

P. Thomas Fletcher, Suresh Venkatasubramanian, and Sarang C. Joshi. 2009. The geometric median on riemannian manifolds with application to robust atlas estimation. *NeuroImage*, 45:S143–S152.

Suchin Gururangan, Swabha Swayamdipta, Omer Levy, Roy Schwartz, Samuel R. Bowman, and Noah A. Smith. 2018. Annotation artifacts in natural language inference data. In *NAACL*.

He He, Sheng Zha, and Haohan Wang. 2019. Unlearn dataset bias in natural language inference by fitting the residual. In *Proceedings of the 2nd Workshop on Deep Learning Approaches for Low-Resource NLP (DeepLo 2019)*, pages 132–142, Hong Kong, China. Association for Computational Linguistics.

Shankar Iyer and Nikhil Dandekarand Kornél Csernai. 2017. First quora dataset release: Question pairs. *data.quora.com*.

Di Jin, Zhijing Jin, Joey Tianyi Zhou, and Peter Szolovits. 2020. Is bert really robust? a strong baseline for natural language attack on text classification and entailment. *Proceedings of the AAAI Conference on Artificial Intelligence*, 34(05):8018–8025.

Divyansh Kaushik, Eduard Hovy, and Zachary Lipton. 2020. Learning the difference that makes a difference with counterfactually-augmented data. In *International Conference on Learning Representations*.

Tom McCoy, Ellie Pavlick, and Tal Linzen. 2019. Right for the wrong reasons: Diagnosing syntactic heuristics in natural language inference. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 3428–3448, Florence, Italy. Association for Computational Linguistics.

Junghyun Min, R. Thomas McCoy, Dipanjan Das, Emily Pitler, and Tal Linzen. 2020. Syntactic data augmentation increases robustness to inference heuristics. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 2339–2352, Online. Association for Computational Linguistics.

Yixin Nie, Adina Williams, Emily Dinan, Mohit Bansal, Jason Weston, and Douwe Kiela. 2020. Adversarial NLI: A new benchmark for natural language understanding. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*. Association for Computational Linguistics.

Roman Novak, Yasaman Bahri, Daniel A. Abolafia, Jeffrey Pennington, and Jascha Sohl-Dickstein. 2018. Sensitivity and generalization in neural networks: an empirical study. In *International Conference on Learning Representations*.

Hans Nyquist. 1988. Least orthogonal absolute deviations. *Computational Statistics & Data Analysis*, 6(4):361–367.

MR Osborne and GA Watson. 1985. An analysis of the total approximation problem in separable norms, and an algorithm for the total l_1 problem. *SIAM journal on scientific and statistical computing*, 6(2):410–424.

Adam Poliak, Jason Naradowsky, Aparajita Haldar, Rachel Rudinger, and Benjamin Van Durme. 2018. Hypothesis only baselines for natural language inference. In *The Seventh Joint Conference on Lexical and Computational Semantics (*SEM)*.

Shiori Sagawa, Aditi Raghunathan, Pang Wei Koh, and Percy Liang. 2020. An investigation of why overparameterization exacerbates spurious correlations. In *ICML*, pages 8346–8356.

Victor Sanh, Thomas Wolf, Yonatan Belinkov, and Alexander M Rush. 2021. Learning from others' mistakes: Avoiding dataset biases without modeling them. In *International Conference on Learning Representations*.

Tal Schuster, Darsh Shah, Yun Jie Serene Yeo, Daniel Roberto Filizzola Ortiz, Enrico Santus, and Regina Barzilay. 2019. Towards debiasing fact verification models. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 3419–3425, Hong Kong, China. Association for Computational Linguistics.

Megha Srivastava, Tatsunori Hashimoto, and Percy Liang. 2020. Robustness to spurious correlations via human annotations. In *International Conference on Machine Learning*, pages 9109–9119. PMLR.

James Thorne, Andreas Vlachos, Christos Christodoulopoulos, and Arpit Mittal. 2018. FEVER: a large-scale dataset for fact extraction and VERification. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 809–819, New Orleans, Louisiana. Association for Computational Linguistics.

Lifu Tu, Garima Lalwani, Spandana Gella, and He He. 2020. An empirical study on robustness to spurious correlations using pre-trained language models. *Transactions of the Association for Computational Linguistics*, 8:621–633.

Iulia Turc, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2020. Well-read students learn better: On the importance of pre-training compact models.

Prasetya Ajie Utama, Nafise Sadat Moosavi, and Iryna Gurevych. 2020a. Mind the trade-off: Debiasing nlu models without degrading the in-distribution performance. In *ACL*, pages 8717–8729.

Prasetya Ajie Utama, Nafise Sadat Moosavi, and Iryna Gurevych. 2020b. Towards debiasing NLU models from unknown biases. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 7597–7610, Online. Association for Computational Linguistics.

Namrata Vaswani, Thierry Bouwmans, Sajid Javed, and Praneeth Narayanamurthy. 2018. Robust subspace learning: Robust pca, robust subspace tracking, and robust subspace recovery. *IEEE Signal Processing Magazine*, 35(4):32–55.

Pascal Vincent, Hugo Larochelle, Isabelle Lajoie, Yoshua Bengio, Pierre-Antoine Manzagol, and Léon Bottou. 2010. Stacked denoising autoencoders: Learning useful representations in a deep network with a local denoising criterion. *Journal of machine learning research*, 11(12).

Tianlu Wang, Diyi Yang, and Xuezhi Wang. 2021. Identifying and mitigating spurious correlations for improving robustness in nlp models. *ArXiv*, abs/2110.07736.

Zhao Wang and Aron Culotta. 2021. Robustness to spurious correlations in text classification via automatically generated counterfactuals. In *AAAI*.

Adina Williams, Nikita Nangia, and Samuel Bowman. 2018. A broad-coverage challenge corpus for sentence understanding through inference. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 1112–1122, New Orleans, Louisiana. Association for Computational Linguistics.

Ruibin Xiong, Yimeng Chen, Liang Pang, Xueqi Cheng, Zhi-Ming Ma, and Yanyan Lan. 2021. Uncertainty calibration for ensemble-based debiasing methods. *Advances in Neural Information Processing Systems*, 34.

Yadollah Yaghoobzadeh, Soroush Mehri, Remi Tachet des Combes, T. J. Hazen, and Alessandro Sordoni. 2021. Increasing robustness to spurious correlations using forgettable examples. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*, pages 3319–3332, Online. Association for Computational Linguistics.

Guanhua Zhang, Bing Bai, Jian Liang, Kun Bai, Shiyu Chang, Mo Yu, Conghui Zhu, and Tiejun Zhao. 2019a. Selection bias explorations and debias methods for natural language sentence matching datasets. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 4418–4429, Florence, Italy. Association for Computational Linguistics.

Yuan Zhang, Jason Baldridge, and Luheng He. 2019b. PAWS: Paraphrase Adversaries from Word Scrambling. In *Proc. of NAACL*.

CORN: Co-Reasoning Network for Commonsense Question Answering

Xin Guan, Biwei Cao, Qingqing Gao, Zheng Yin, Bo Liu, Jiuxin Cao*
Southeast University
{xin_guan,caobiwei,qingqing_gao,z.yin,bliu,jx.cao}@seu.edu.cn

Abstract

Commonsense question answering (QA) requires machines to utilize the QA content and external commonsense knowledge graph (KG) for reasoning when answering questions. Existing work uses two independent modules to model the QA contextual text representation and relationships between QA entities in KG, which prevents information sharing between modules for co-reasoning. In this paper, we propose a novel model, **Co**-**R**easoning **N**etwork (CORN), which adopts a bidirectional multi-level connection structure based on Co-Attention Transformer. The structure builds *bridges* to connect each layer of the text encoder and graph encoder, which can introduce the QA entity relationship from KG to the text encoder and bring contextual text information to the graph encoder, so that these features can be deeply interactively fused to form comprehensive text and graph node representations. Meanwhile, we propose a QA-aware node based KG subgraph construction method. The QA-aware nodes aggregate the question entity nodes and the answer entity nodes, and further guide the expansion and construction process of the subgraph to enhance the connectivity and reduce the introduction of noise. We evaluate our model on QA benchmarks in the CommonsenseQA and OpenBookQA datasets, and CORN achieves state-of-the-art performance.

1 Introduction

Commonsense Question answering (QA) research requires the machine to have a human thought pattern, which is capable of comprehending text content and combining commonsense knowledge to reason and arrive at the correct answer. Despite the success of large pre-trained language models (PLMs) (Devlin et al., 2019; Liu et al., 2019) on various NLP tasks, there is still a large gap between PLMs and humans on commonsense QA tasks. Therefore, researchers try to introduce external knowledge, such as Freebase (Bollacker et al., 2008) and ConceptNet (Speer et al., 2017), which are large knowledge graphs (KGs) that entities link by various relationships.

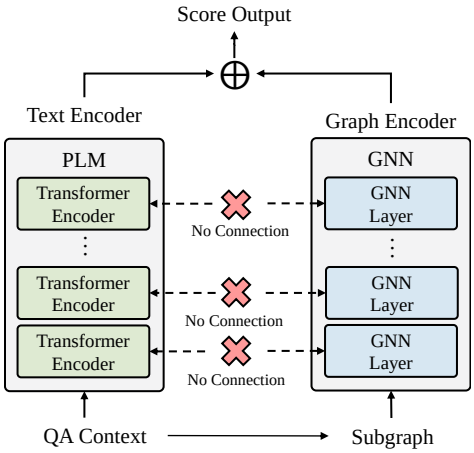

Figure 1: The network architecture of current commonsense QA research. Text encoder for encoding the QA content, graph encoder for reasoning on the graph. There is no connection between stacked text layers and stacked GNN layers.

There has already been a significant amount of work that combines PLMs and KGs for reasoning (Lin et al., 2019; Wang et al., 2020; Feng et al., 2020; Yasunaga et al., 2021). As illustrated in Figure 1, these works are mainly composed of two modules: (1) capturing text features on QA with a text encoder (such as PLM). (2) extracting subgraph from KG and reasoning on it with a graph encoder, such as the GNN-based model (Kipf and Welling, 2017; Schlichtkrull et al., 2018). Most work (Lin et al., 2019; Wang et al., 2020; Feng et al., 2020) focuses on building more efficient graph encoders to capture relationships between entities in graphs for reasoning. However, it ignores the interconnections between the QA content and graph due to GNN and PLM being treated as independent modules. To address the above problems,

*Corresponding author.

Proceedings of the 29th International Conference on Computational Linguistics, pages 1677–1686
October 12–17, 2022.

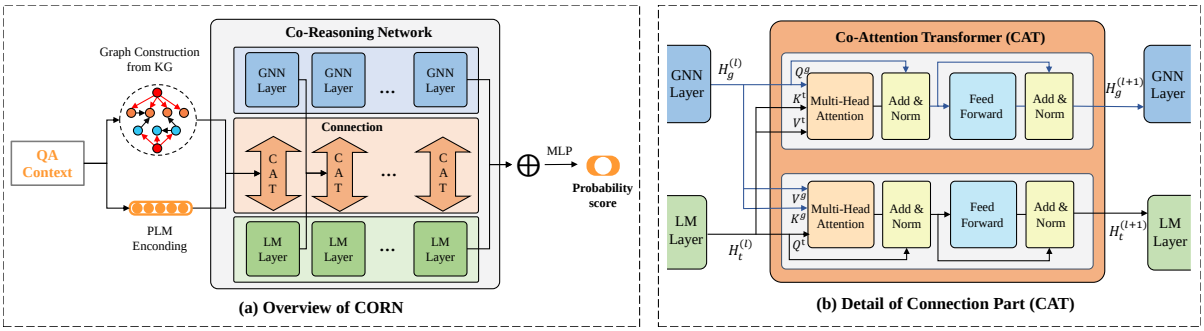

Figure 2: Overview of our model. **(a)** is the overall architecture of CORN. We first construct a subgraph related to QA by KG (§3.1), use PLM to encode the QA content, then apply N-layer Co-Reasoning Network, which combines GNN layer (§3.4) and LM layer (§3.3) through Co-Attention Transformer (CAT) (§3.2), to perform reasoning. **(b)** is the detail of connection part (Co-Attention Transformer).

Yasunaga et al. (2021) perform joint reasoning by explicitly adding the QA content to the graph in the form of a node. Nevertheless, this method is a one-way connection structure, only enabling the graph encoder to obtain textual context information for reasoning while the text encoder cannot perceive graph information.

We propose a novel model, **Co**-**R**easoning **N**etwork (CORN), to solve the above problems. CORN adopts a bidirectional multi-level connection structure, which connects the language model (LM) and GNN. Specifically, we build *bridge*s between each layer of these two types of models by the Co-Attention Transformer. Through this *bridge*, the text representation and graph node representation can be fused bidirectionally and respectively fed into the next LM and GNN layer. Therefore, the GNN layer can reason on the subgraph with contextual text representation to enrich the graph node representation, and the LM layer can further encode the text with the graph node representation to improve the text representation. We adopt a multi-level connection structure to connect each layer, enabling text and graph representation with different semantic levels to interact, generating a more comprehensive feature representation. Meanwhile, we propose a QA-aware node based KG subgraph construction method. We use a question-aware node and an answer-aware node to aggregate the question entity nodes and the answer entity nodes respectively and then guide the expansion and construction process of the subgraph to enhance the connectivity of the subgraph and reduce the introduction of noise. Moreover, the QA-aware node can help the model perceive the difference between different types of nodes and help the model to learn the representation of graph nodes better.

The main contributions of this work are summarized as follows:

- We propose CORN, which adopts a bidirectional multi-level connection structure. CORN uses Co-Attention Transformer to connect each layer of the LM and GNN, which allows LM to perceive graph information and enables GNN to integrate contextual text information, so that GNN and LM can generate richer text and graph node representations.
- We propose a QA-aware node based KG subgraph construction method. Question entity nodes and answer entity nodes are aggregated through the QA-aware nodes and then the QA-aware nodes guide the subgraph expansion and construction to improve the connectivity and reduce the noise.
- We conduct extensive experiments on OpenBookQA and CommonsenseQA, and CORN achieves state-of-the-art performance compared to other KGs+PLMs models.

2 Problem Statement

In this paper, we focus on the task of multiple-choice question answering which required extra knowledge of reasoning. Formally, giving a question q, a set of answer choices $\mathcal{C}$ and external knowledge graph, our purpose is to identify the correct answer from $\mathcal{C}$.

To be specific, we calculate the probability score between q and each answer choice $a \in \mathcal{C}$ and then select the answer with the highest probability score. We construct a multi-relational subgraph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ by KG (detailed in §3.1). Here $\mathcal{V}$

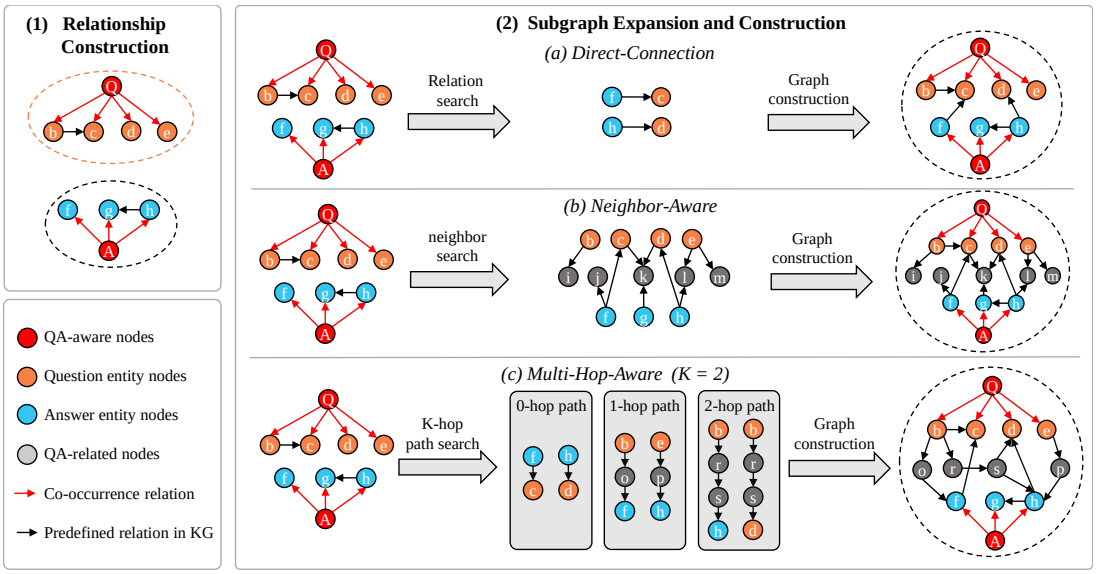

Figure 3: The QA-aware node based subgraph construction method. We first use the QA-aware nodes to aggregate QA entities and then derive different subgraph expansion strategies to construct the final subgraph.

is the subset of entity nodes from KG and is related to the mentioned entities in the QA content, $\mathcal{E} \subseteq \mathcal{V} \times \mathcal{R} \times \mathcal{V}$ is the set of edges that connect nodes in $\mathcal{V}$, where $\mathcal{R}$ represents a set of relation types.

3 Co-Reasoning Network

The overview of our model is shown in Figure 2. We concatenate a question q and an answer choice a to get the QA content s, where $s = [q; a]$. We apply PLMs (e.g., RoBERTa) on s to get the initial text representation s'. For each QA content s, we use the QA-aware node based subgraph construction method (§3.1) to construct a subgraph $\mathcal{G}$ and initialize the graph node representation $\mathcal{G}'$. Then, We use N-layer *Co-Reasoning Network* (CORN) to reason on the QA content s' and subgraph $\mathcal{G}'$. Each CORN layer consists of LM layer (§3.3), GNN layer (§3.4) and Co-Attention Transformer (§3.2), where Co-Attention Transformer bidirectionally connects LM and GNN, LM encodes QA text with graph representation, and GNN reasons on the subgraph with the contextual text representation. Finally, we use the pooled graph representation and text representation from the last CORN layer to make predictions to get the probability that the current choice a is the correct choice.

3.1 The QA-aware node based KG Subgraph Construction Method

We propose a QA-aware node based KG subgraph construction method. As shown in Figure 3, there are two stages during the process of constructing a subgraph.

Relationship Construction. We introduce a question-aware node $\mathcal{A}_q$ and an answer-aware node $\mathcal{A}_a$ which are respectively responsible for aggregating entity nodes that appear in the question context and answer context. Specifically, the question-aware node $\mathcal{A}_q$ first connects all question entity nodes $\mathcal{V}_q$ existing in the KG with the "co-occurrence" relationship, then queries the relationship in KG between each pair of the question entity nodes and connects them with the queried relationship. We construct the relationship between answer-aware node $\mathcal{A}_a$ and answer entity nodes $\mathcal{V}_a$ in the same steps.

Subgraph Expansion and Construction. After constructing relationship in QA entity nodes, we expand the subgraph to supplement additional QA-related knowledge nodes $\mathcal{V}_o$ to obtain richer graph information. To reduce the introduction of noisy nodes, we propose different subgraph expansion strategies. (a) **Direct-Connection**: This strategy does not introduce additional knowledge nodes. It only connects question-answer entity pairs that have a relationship in KG. It sacrifices some graph information in exchange for introducing the least number of noisy nodes. (b) **Neighbor-Aware**: This strategy introduces the neighbor entity nodes of question and answer entity nodes as additional knowledge nodes. It introduces rich neighbor information for each question answering entity node. (c) **Multi-Hop-Aware**: This strategy searches for

a reachable path within K-hop in KG between two nodes of question and answer entity nodes set, and introduces the nodes on the path as the additional knowledge nodes. It introduces less graph information and noisy nodes. We take different strategies to introduce additional nodes to form the subgraph $\mathcal{G}$, and evaluate each subgraph construction strategy in the experiment. We initialize the node embedding of $\mathcal{V}_q$, $\mathcal{V}_a$ and $\mathcal{V}_o$ by its entity embedding (§4.2), and simply initialize $\mathcal{A}_q$ and $\mathcal{A}_a$ with zero vectors.

3.2 Co-Attention Transformer

The most central problem of the current model is that the GNN and LM are treated as independent modules for reasoning. GNN model can not effectively use QA contextual text representation and only rely on the subgraph extracted from KGs for reasoning. Also, LM only encodes QA text and ignores the QA entity relationship. To address this problem, we bidirectionally connect each layer of GNN and LM through the Co-Attention Transformer, which allows these two modules to interact with each other's information to improve the text and graph node representation. The structure of the Co-Attention Transformer is shown in Figure 2 (b).

Specially, given the text representation $H_t^{(l)} \in \mathbb{R}^{m\times d}$ from the l-th layer of LM and node representation $H_g^{(l)} \in \mathbb{R}^{n\times d}$ from the l-th layer of GNN model, where m, n are the text length and number of nodes and d is the hidden size, we map them to query Q^t, Q^g, key K^t, K^g and value V^t, V^g matrices as in a standard transformer layer:

$$\begin{aligned} Q_i &= H^{(l)} W_i^Q \\ K_i &= H^{(l)} W_i^K \\ V_i &= H^{(l)} W_i^V \end{aligned} \tag{1}$$

where i is i-th of h matrices, $\{W_i^Q, W_i^K, W_i^V\} \in \mathbb{R}^{d\times d_k}$ are parameter matrices and $d_k \in \mathbb{R}^{d/h}$.

Then, we apply two transformer layers and exchange key-value pairs in multi-head attention to perform interactive computation. Q^t, K^g and V^g form one group and Q^g, K^t and V^t form other group. Each group performs multi-head attention computation conditioned on the other modules. The single attention head is as following:

$$\mathrm{T_i}(Q_i^t, K_i^g, V_i^g) = \mathrm{softmax}\left(\frac{Q_i^t K_i^{gT}}{\sqrt{d_k}}\right) V_i^g \tag{2}$$

$$\mathrm{G_i}(Q_i^g, K_i^t, V_i^t) = \mathrm{softmax}\left(\frac{Q_i^g K_i^{tT}}{\sqrt{d_k}}\right) V_i^t \tag{3}$$

The attention outputs of each head are then concatenated and followed by a linear transformation as following:

$$\begin{aligned} O_t^{(l)} &= \mathrm{MultiHead}(Q^t, K^g, V^g) \\ &= \mathrm{Concat}\left(\mathrm{T_1}, \cdots, \mathrm{T_h}\right) W_t^O \end{aligned} \tag{4}$$

$$\begin{aligned} O_g^{(l)} &= \mathrm{MultiHead}(Q^g, K^t, V^t) \\ &= \mathrm{Concat}\left(\mathrm{G_1}, \cdots, \mathrm{G_h}\right) W_g^O \end{aligned} \tag{5}$$

where $\{W_t^O, W_g^O\} \in \mathbb{R}^{hd_k\times d}$ are parameter matrices, $O_t^{(l)} \in \mathbb{R}^{m\times d}$ and $O_g^{(l)} \in \mathbb{R}^{n\times d}$. After that, two residual add operations are worked on the initial representation and output of multi-head attention to get a fused representation of text and graph:

$$H_t^{(l)} = \mathrm{LayerNorm}(H_t^{(l)} + O_t^{(l)}) \tag{6}$$

$$H_g^{(l)} = \mathrm{LayerNorm}(H_g^{(l)} + O_g^{(l)}) \tag{7}$$

where LayerNorm is the layer normalization operation (Ba et al., 2016). $H_t^{(l)}$ is the text representation with graph information and $H_g^{(l)}$ is the node representation with text information. Then, two feed forward networks (MLP) and two another residual add operations are applied on the above representation to get the Co-Attention Transformer output:

$$H_t^{(l+1)} = \mathrm{LayerNorm}(H_t^{(l)} + \mathrm{MLP}(H_t^{(l)})) \tag{8}$$

$$H_g^{(l+1)} = \mathrm{LayerNorm}(H_g^{(l)} + \mathrm{MLP}(H_g^{(l)})) \tag{9}$$

where $H_t^{(l+1)} \in \mathbb{R}^{m\times d}$ and $H_g^{(l+1)} \in \mathbb{R}^{n\times d}$ are the input of $(i+1)$-th layer of LM and GNN. We use Co-Attention Transformer to connect each layer of the LM and GNN layer, which can fuse semantic features of different levels to obtain a more comprehensive representation.

3.3 Language Model

To effectively utilize the capability of the PLM, we do not modify its architecture and use it to encode the text at first. Specifically, we apply PLM (e.g., RoBERTa) on the QA content s to get initial text representation $H_t^{(0)}$:

$$H_t^{(0)} = \mathrm{PLM}(s), \tag{10}$$

where $H_t^{(0)} \in \mathbb{R}^{m\times d_p}$, m is the text length and d_p is the hidden size of PLM.

Before Co-Reasoning Network, we use an MLP to unify the hidden size:

$$H_t^{(0)} = \mathrm{MLP}(H_t^{(0)}), \tag{11}$$

where the new $H_t^{(0)} \in \mathbb{R}^{m\times d}$, d is the unified hidden size.

After that, we use an N-layer Co-Reasoning Network which consists of LM, GNN, and Co-Attention Transformer for co-reasoning. For the l-th layer, the input text representation $H_t^{(l-1)}$ interacts with node representation $H_g^{(l-1)}$ in graph (detailed in §3.4) through Co-Attention Transformer and gets the text representation $H_t^{(l)}$ that is contained graph node information.

Further, we apply transformer encoder layer (Vaswani et al., 2017) as the LM in Co-Reasoning Network for reasoning:

$$H_t^{(l)} = \text{Transformer}(H_t^{(l)}) \tag{12}$$

The transformer encoder layer can encode the text representation with graph information to get the output $H_t^{(l)}$.

3.4 GNN Model

After getting knowledge concept graph $\mathcal{G}$ and initializing the entity node embedding $H_g^{(0)} \in \mathbb{R}^{n\times d_{in}}$, where n is the number of nodes and d_{in} is the initial hidden size, we also use an MLP to unify the hidden size of node:

$$H_g^{(0)} = \text{MLP}(H_g^{(0)}), \tag{13}$$

where the new $H_g^{(0)} \in \mathbb{R}^{n\times d}$, d is the unified hidden size.

We put the graph into N-layer Co-Reasoning Network. For the l-th layer, the input node representation $H_g^{(l-1)}$ interacts with text representation $H_t^{(l-1)}$ through Co-Attention Transformer and gets the node representation $H_g^{(l)}$ that is contained contextual text information.

Further, We apply RGCN (Schlichtkrull et al., 2018), a graph encoder that can encode multi-relational graphs by aggregating messages from its neighbors, as the GNN model in Co-Reasoning Network for reasoning. Specially, for each node $h_i^{(l)} \in \mathbb{R}^d$ in graph, where $[h_1^{(l)}; \cdots ; h_n^{(l)}] = H_g^{(l)}$, the node representation is updated via message passing from neighbors:

$$h_i^{(l+1)} = \sigma(\sum_{r\in\mathcal{R}} \sum_{j\in\mathcal{N}_i^r} \frac{1}{c_{i,r}} W_r^{(l)} h_j^{(l)} + W_0^{(l)} h_i^{(l)}), \tag{14}$$

where $\mathcal{N}_i^r$ denotes the set of neighbor indices of node i under relation $r \in \mathcal{R}$, $c_{i,r} = |\mathcal{N}_i^r|$ is a normalization constant. $W_r^{(l)}$ is the parameter matrix related to relation r and $W_0^{(l)}$ is the parameter matrix of node i information transformation. However, the number of parameters grows rapidly with the increase in the number of relations. We apply basis decomposition to regularize the weights of R-GCN-layers:

$$W_r^{(l)} = \sum_{b=1}^{B} a_{rb}^{(l)} V_b^{(l)}, \tag{15}$$

where $V_b^{(l)} \in \mathbb{R}^{d^{(l+1)}\times d^l}$ is the parameter matrices for all relation and $a_{rb}^{(l)}$ is the coefficients depend on r. After apply RGCN-layer, the new node representations are updated and get the output $H_g^{(l)} = [h_1^{(l+1)}; \cdots ; h_n^{(l+1)}]$.

3.5 Inference & Learning

The probability score for answer a as the correct answer for question q is calculated by text representation and graph node representations from the last layer:

$$p(q,a) = \text{MLP}(\text{Pool}(H_t^{(L)}) \oplus \text{Pool}(H_g^{(L)})) \tag{16}$$

where Pool is the mean pooling operation over the text representations and the node representations.

In the training process, each question provides a list of answer choices, one of which is correct. we use the cross-entropy loss function to optimize the model.

4 Experiments

4.1 Datasets

We evaluate our model on two multiple-choice question answering datasets that require external knowledge to arrive at the correct answer: OpenBookQA (Mihaylov et al., 2018) and CommonsenseQA (Talmor et al., 2019).

OpenBookQA is a multiple choice question QA task with 4 choices that require elementary science knowledge for reasoning. This dataset also provides external knowledge called *Open Books* describing scientific facts to help models answer questions. As our study focuses on reasoning by using structured knowledge, we do not utilize *Open Books* and instead utilize *ConceptNet* as the external knowledge.

CommonsenseQA is a multiple choice question QA task with 5 choices that require commonsense knowledge for reasoning. Questions and answers

are generated according to entities in *ConceptNet* and their relations. We perform experiments on the in-house (IH) data splits used in Lin et al. (2019).

4.2 Knowledge Graph

We use *ConceptNet*, a general-domain structured knowledge graph, as our external commonsense knowledge. We use the entity embeddings prepared by Feng et al. (2020), which they utilize TransE model (Bordes et al., 2013) for node embedding (100-dimensional) in *ConceptNet*. Following the work by Lin et al. (2019), we merge the original 42 relation types in *ConceptNet* into 17 relation types. The subgraph construction method for each question and answer is described in section §3.1.

4.3 Implementation & training details

We set the dimension ($D = 100$) and number of layers ($L = 3$) of our Co-Reasoning Network, with dropout rate 0.2 applied to each layer. We train the model with the RAdam optimizer using one GPU (Tesla T4). We use batch size of 64 (mini batch of 2), with 14 epochs (~4 hours) for OpenBookQA and 10 epochs (~6 hours) for CommonsenseQA.

4.4 Baseline Models

The purpose of our work is to leverage structured external knowledge for reasoning on knowledge question answering tasks. Therefore, we only compare with the models that combine PLMs and KGs, not the models using other formats of external knowledge (e.g., Wikipedia, human-annotated evidence.)

RoBERTa (Liu et al., 2019) is used as the baseline model to study the performance of PLMs without introducing extra KG information.

GconAttn (Wang et al., 2019) generalizes the Match-LSTM model in the field of text matching to knowledge concept matching.

KagNet (Lin et al., 2019) extracts the QA-related subgraph from KG, and applies GCN and LSTM to model the relational paths.

Relation Network (RN) (Santoro et al., 2017) utilizes multilayer perceptron to encode triplets on paths in KG and all the triplets representation as to the graph representation for classification.

MHGRN (Feng et al., 2020) designs a multi-hop relational reasoning module to obtain a path-level graph representation, and combines GNN and PLMs for classification.

QA-GNN (Yasunaga et al., 2021) introduces a QA content node in the subgraph for joint reasoning over the QA content and KG.

Methods	**Dev Acc.**(%)	**Test Acc.**(%)
RoBERTa-large	66.76 (±1.14)	64.80 (±2.37)
+ GconAttn	66.85 (±1.82)	64.75 (±1.48)
+ RN	67.00 (±0.71)	65.20 (±1.18)
+ MHGRN	68.10 (±1.12)	66.85 (±1.19)
+ QA-GNN	68.27 (±1.09)	67.80 (±2.75)
+ CORN (**Ours**)	**72.35** (±0.86)	**71.30** (±0.64)

Table 1: **Dev and Test accuracy on OpenBookQA.**

Methods	**IHdev-Acc.**(%)	**IHtest-Acc.**(%)
RoBERTa-large	73.07 (±0.45)	68.69 (±0.56)
+ GconAttn	72.61 (±0.39)	68.59 (±0.96)
+ KagNet	73.47 (±0.22)	69.01 (±0.76)
+ RN	74.57 (±0.91)	69.08 (±0.21)
+ MHGRN	74.45 (±0.10)	71.11 (±0.81)
+ QA-GNN	76.54 (±0.21)	73.41 (±0.92)
+ CORN (**Ours**)	**79.58** (±0.38)	**74.43** (±0.59)

Table 2: **Dev and Test accuracy on CommonsenseQA in-house split.**

4.5 Main Results

Table 1 shows the result on OpenBookQA. For fair comparison, we use the RoBERTa-large as the text encoder for all models. Our model achieves the best performance across all baseline models by greatly improving the dev accuracy by ~4.08% and test accuracy by ~3.5%. The improvement over QA-GNN suggests that CORN is a better method to combine the QA content and KG for co-reasoning. Notably, QA-GNN (1.1B total parameters) uses 2.5x more total parameters than our model (our has 440M total parameters). This benefits from CORN's use of fewer layers, smaller hidden dimension, and simpler designed GNN but more efficient connection structure. In addition, we did not compare with other models that use the extra corpus of scientific facts provided by official, because our purpose is to reason from a structured knowledge graph.

Table 2 shows the result on CommonsenseQA. All models also use Roberta-large as the text encoder. CORN achieves state-of-art performance across all existing models with improving the dev accuracy by ~3.04% and test accuracy by ~1.02%. The result suggests that CORN improves the performance of through the bidirectional interaction

Methods	OpenBookQA	CommonseQA
Direct-Connection	69.27 (±0.41)	**74.43** (±0.59)
Neighbor-Aware	**71.30** (±0.64)	72.66 (±0.40)
Multi-Hop-Aware (K=1)	69.60 (±1.28)	71.64 (±0.16)
Multi-Hop-Aware (K=2)	68.30 (±1.30)	71.77 (±0.56)

Table 3: **Results of different subgraph construction methods.** We report the test accuracy on OpenBookQA and CommonsenseQA.

Method	Test Acc.
CORN	**71.30**
w/o CAT (Text)	70.17
w/o CAT (Graph)	70.20
w/o CAT (Multi-level)	70.26
w/o CAT	69.67
w/o QA-aware nodes	70.20

CORN Layers	Test Acc.
$L = 2$	70.90
$L = 3$	**71.30**
$L = 4$	70.73
$L = 5$	70.40

Table 4: **Ablation study** of our model components, using the OpenBookQA test set. CAT is the abbreviation of Co-Attention Transformer.

of GNN and LM without designing complex graph inference networks.

4.6 Subgraph Construction Result

Table 3 shows the results of constructing subgraphs by different subgraph expansion methods. We evaluate our proposed strategies for introducing additional QA-related knowledge nodes in KG.

For OpenBookQA, we find that the *Neighbor-Aware* performs best. The OpenBookQA emphasizes reasoning using multiple scientific knowledge. The question entity nodes require multiple scientific knowledge to connect with the answer entity nodes. Therefore, the model cannot perform reasoning efficiently without introducing extra nodes, which is the reason that *Direct-Connect* performs worst. Compared to *Multi-Hop-Aware (K=1)* and *Multi-Hop-Aware (K=2)*, *Neighbor-Aware* can provide extra influential related entity nodes to cover possible scientific knowledge, further helping the model for reasoning.

For CommonsenseQA, the best strategy is to not introduce additional QA-related knowledge nodes (*Direct-Connection*). We analyze that this is caused by the construction method of CommonsenseQA. The four answer entities and one question entity in CommonsenseQA are directly connected in ConceptNet. Therefore, the introduction of additional QA-related knowledge nodes will lead to noise and redundancy, which will reduce the performance of the model.

4.7 Ablation Study

The ablation study on each of our model components is shown in Table 4, using the OpenBookQA test set. We found that removing the attention computation of any module (text or graph) in the Co-Attention Transformer would result in a performance drop of ~1.1%. We also test removing the multi-level connection structure and only keeping one layer of the Co-Attention Transformer, the degraded performance result shows that multi-level connections can indeed interact with different levels of semantic features to get richer representation. Removing Co-Attention Transformer will significantly degrade performance by 1.63%, which proves the importance of connecting and co-reasoning between LM and GNN. We also analyze the impact of QA-aware nodes. When we remove QA-aware nodes, the performance of the model drops by 1.1%, which proves that QA-aware nodes can help the model to perform better reasoning. For the number of CORN layers, we find $L = 3$ works best on the dev set, which is also similar to the number of layers generally used in GNN.

4.8 Model Visualization

The purpose of our model is to make GNN and LM mutually aware of the information of each other's modules for reasoning. Therefore, we analyze the attention weights of text module and graph module in Co-Attention Transformer. Figure 4 gives a visualization of an example. Given question *"Where would you find magazines along side many other printed works?"* and choices *"A. doctor B. bookstore"*, we show the attention weights of the last Co-Attention Transformer layer under these two choices separately. The key entity in this question is *"magazines"*. For the wrong choice *"doctor"*, though the attention of text module can give a higher weight of *"magazines"* entity in the graph, the attention weight distribution of graph module is rather average, and cannot provide meaningful information. For the correct choice *"bookstore"*, not only the attention of text module can capture the importance of the *"magazine"* entity in the graph, but also the attention in graph module gives a higher weight to the *"bookstore"* in the text, which is also the correct answer. Therefore, the Co-Attention Transformer in CORN can effectively capture the relationship between the QA text and knowledge graph formed by the correct choice.

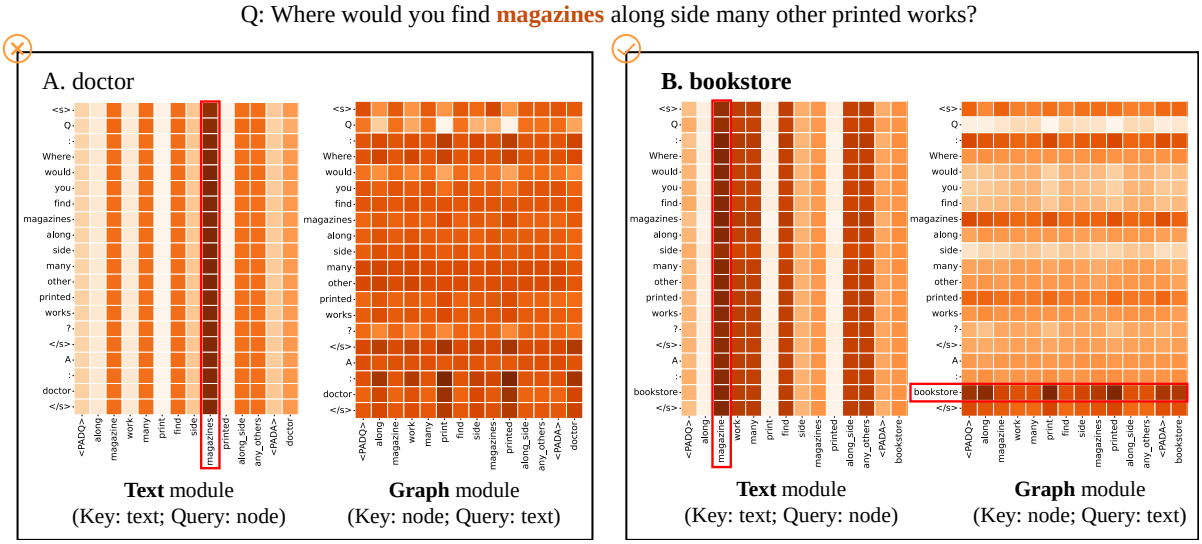

Figure 4: **Visualization of the attention weights of the last CORN layer.** Given a question and corresponding choices. We show the attention weights between text and graph formed by each choice. The row indices of the heatmap are the words in the text, and the column indices are the entity nodes in the graph. The red box represents the part with the highest attention weight.

5 Related Work

Question answering with PLMs. The recent success of PLMs in various NLP tasks has prompted much work to try directly utilize PLMs to encode external knowledge. These work can be divided into two paradigms: 1) Format external knowledge (eg. Wikipedia, knowledge graph) into text or triples as corpus for PLM pre-training task (Ye et al., 2019; Li et al., 2019; Sun et al., 2019; Gururangan et al., 2020). 2) Fine-tuning PLMs with evidence for external knowledge (Pan et al., 2019; Lv et al., 2020). However, such models can not provide interpretable reasoning process, which is the key to commonsense reasoning.

Question answering with KG+LM. Many works attempt to additionally perform reasoning with GNN on knowledge graphs to address the problem that PLMs unable to reason on structured knowledge. GCN (Kipf and Welling, 2017) aggregates the neighborhood information of each node for message passing. RGCN (Schlichtkrull et al., 2018) can encode multi-relational graphs by aggregating messages from its neighbors of different relations. GAT (Velickovic et al., 2018) assigns different attention weights to aggregate each node feature, and is used (Chen et al., 2019) to distinguish the importance of different concept entity nodes in KG. The related work of question answering (Lin et al., 2019; Feng et al., 2020; Lv et al., 2020) try to design complex graph neural networks for single-hop or multi-hop reasoning in KG. However, these works treat the QA content and KG as separate modules. Though Yasunaga et al. (2021) add the QA content to graph for joint reasoning, it still cannot solve the problem that information unable exchange between GNN and LM. CORN addresses the above problem by connecting each layer of these two models through Co-Attention Transformer for co-reasoning.

6 Conclusion

We propose a novel commonsense QA model, CORN, which adopts a bidirectional multi-level connection structure. It bidirectionally connects each layer of the LM and GNN through the Co-Attention Transformer, which enables the LM to perceive the relationship of QA entity nodes to improve the text representation, and allows GNN to utilize contextual text information to enhance the graph node representation. Meanwhile, we propose a QA-aware node based KG subgraph construction method. The QA-aware nodes aggregate question entity nodes and answer entity nodes and then guide the subgraph expansion and construction to increase the connectivity of the subgraph, and reduce the introduction of noise. Through extensive experiments and visual analysis, CORN can perform multi-level bidirectional interaction to improve the LM+KG models, and achieves state-of-the-art performance among them.

Acknowledgements

This work is supported by National Key R&D Project of China under Grants No.2021QY2102. National Natural Science Foundation of China under Grants No.62172089, No.61972087, No.62172090, No.62106045. Natural Science Foundation of Jiangsu Province under Grants No.BK20191258. Jiangsu Provincial Key Laboratory of Computer Networking Technology. Jiangsu Provincial Key Laboratory of Network and Information Security under Grants No.BM2003201, and Key Laboratory of Computer Network and Information Integration of Ministry of Education of China under Grants No.93K-9, Nanjing Purple Mountain Laboratory.

References

Lei Jimmy Ba, Jamie Ryan Kiros, and Geoffrey E. Hinton. 2016. Layer normalization. *CoRR*, abs/1607.06450.

Kurt D. Bollacker, Colin Evans, Praveen K. Paritosh, Tim Sturge, and Jamie Taylor. 2008. Freebase: a collaboratively created graph database for structuring human knowledge. In *Proceedings of the ACM SIGMOD International Conference on Management of Data, SIGMOD 2008, Vancouver, BC, Canada, June 10-12, 2008*, pages 1247–1250. ACM.

Antoine Bordes, Nicolas Usunier, Alberto García-Durán, Jason Weston, and Oksana Yakhnenko. 2013. Translating embeddings for modeling multi-relational data. In *Advances in Neural Information Processing Systems 26: 27th Annual Conference on Neural Information Processing Systems 2013. Proceedings of a meeting held December 5-8, 2013, Lake Tahoe, Nevada, United States*, pages 2787–2795.

Daoyuan Chen, Yaliang Li, Min Yang, Hai-Tao Zheng, and Ying Shen. 2019. Knowledge-aware textual entailment with graph attention network. In *Proceedings of the 28th ACM International Conference on Information and Knowledge Management, CIKM 2019, Beijing, China, November 3-7, 2019*, pages 2145–2148. ACM.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT 2019, Minneapolis, MN, USA, June 2-7, 2019, Volume 1 (Long and Short Papers)*, pages 4171–4186. Association for Computational Linguistics.

Yanlin Feng, Xinyue Chen, Bill Yuchen Lin, Peifeng Wang, Jun Yan, and Xiang Ren. 2020. Scalable multi-hop relational reasoning for knowledge-aware question answering. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing, EMNLP 2020, Online, November 16-20, 2020*, pages 1295–1309. Association for Computational Linguistics.

Suchin Gururangan, Ana Marasovic, Swabha Swayamdipta, Kyle Lo, Iz Beltagy, Doug Downey, and Noah A. Smith. 2020. Don't stop pretraining: Adapt language models to domains and tasks. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, ACL 2020, Online, July 5-10, 2020*, pages 8342–8360. Association for Computational Linguistics.

Thomas N. Kipf and Max Welling. 2017. Semi-supervised classification with graph convolutional networks. In *5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Conference Track Proceedings*. OpenReview.net.

Shiyang Li, Jianshu Chen, and Dian Yu. 2019. Teaching pretrained models with commonsense reasoning: A preliminary kb-based approach. *CoRR*, abs/1909.09743.

Bill Yuchen Lin, Xinyue Chen, Jamin Chen, and Xiang Ren. 2019. Kagnet: Knowledge-aware graph networks for commonsense reasoning. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing, EMNLP-IJCNLP 2019, Hong Kong, China, November 3-7, 2019*, pages 2829–2839. Association for Computational Linguistics.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Roberta: A robustly optimized BERT pretraining approach. *CoRR*, abs/1907.11692.

Shangwen Lv, Daya Guo, Jingjing Xu, Duyu Tang, Nan Duan, Ming Gong, Linjun Shou, Daxin Jiang, Guihong Cao, and Songlin Hu. 2020. Graph-based reasoning over heterogeneous external knowledge for commonsense question answering. In *The Thirty-Fourth AAAI Conference on Artificial Intelligence, AAAI 2020, The Thirty-Second Innovative Applications of Artificial Intelligence Conference, IAAI 2020, The Tenth AAAI Symposium on Educational Advances in Artificial Intelligence, EAAI 2020, New York, NY, USA, February 7-12, 2020*, pages 8449–8456. AAAI Press.

Todor Mihaylov, Peter Clark, Tushar Khot, and Ashish Sabharwal. 2018. Can a suit of armor conduct electricity? A new dataset for open book question answering. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing, Brussels, Belgium, October 31 - November 4, 2018*, pages 2381–2391. Association for Computational Linguistics.

Xiaoman Pan, Kai Sun, Dian Yu, Jianshu Chen, Heng Ji, Claire Cardie, and Dong Yu. 2019. Improving question answering with external knowledge. In *Proceedings of the 2nd Workshop on Machine Reading for Question Answering, MRQA@EMNLP 2019, Hong Kong, China, November 4, 2019*, pages 27–37. Association for Computational Linguistics.

Adam Santoro, David Raposo, David G. T. Barrett, Mateusz Malinowski, Razvan Pascanu, Peter W. Battaglia, and Tim Lillicrap. 2017. A simple neural network module for relational reasoning. In *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, December 4-9, 2017, Long Beach, CA, USA*, pages 4967–4976.

Michael Sejr Schlichtkrull, Thomas N. Kipf, Peter Bloem, Rianne van den Berg, Ivan Titov, and Max Welling. 2018. Modeling relational data with graph convolutional networks. In *The Semantic Web - 15th International Conference, ESWC 2018, Heraklion, Crete, Greece, June 3-7, 2018, Proceedings*, volume 10843 of *Lecture Notes in Computer Science*, pages 593–607. Springer.

Robyn Speer, Joshua Chin, and Catherine Havasi. 2017. Conceptnet 5.5: An open multilingual graph of general knowledge. In *Proceedings of the Thirty-First AAAI Conference on Artificial Intelligence, February 4-9, 2017, San Francisco, California, USA*, pages 4444–4451. AAAI Press.

Yu Sun, Shuohuan Wang, Yu-Kun Li, Shikun Feng, Xuyi Chen, Han Zhang, Xin Tian, Danxiang Zhu, Hao Tian, and Hua Wu. 2019. ERNIE: enhanced representation through knowledge integration. *CoRR*, abs/1904.09223.

Alon Talmor, Jonathan Herzig, Nicholas Lourie, and Jonathan Berant. 2019. Commonsenseqa: A question answering challenge targeting commonsense knowledge. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT 2019, Minneapolis, MN, USA, June 2-7, 2019, Volume 1 (Long and Short Papers)*, pages 4149–4158. Association for Computational Linguistics.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, December 4-9, 2017, Long Beach, CA, USA*, pages 5998–6008.

Petar Velickovic, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Liò, and Yoshua Bengio. 2018. Graph attention networks. In *6th International Conference on Learning Representations, ICLR 2018, Vancouver, BC, Canada, April 30 - May 3, 2018, Conference Track Proceedings*. OpenReview.net.

Peifeng Wang, Nanyun Peng, Filip Ilievski, Pedro A. Szekely, and Xiang Ren. 2020. Connecting the dots: A knowledgeable path generator for commonsense question answering. In *Findings of the Association for Computational Linguistics: EMNLP 2020, Online Event, 16-20 November 2020*, volume EMNLP 2020 of *Findings of ACL*, pages 4129–4140. Association for Computational Linguistics.

Xiaoyan Wang, Pavan Kapanipathi, Ryan Musa, Mo Yu, Kartik Talamadupula, Ibrahim Abdelaziz, Maria Chang, Achille Fokoue, Bassem Makni, Nicholas Mattei, and Michael Witbrock. 2019. Improving natural language inference using external knowledge in the science questions domain. In *The Thirty-Third AAAI Conference on Artificial Intelligence, AAAI 2019, The Thirty-First Innovative Applications of Artificial Intelligence Conference, IAAI 2019, The Ninth AAAI Symposium on Educational Advances in Artificial Intelligence, EAAI 2019, Honolulu, Hawaii, USA, January 27 - February 1, 2019*, pages 7208–7215. AAAI Press.

Michihiro Yasunaga, Hongyu Ren, Antoine Bosselut, Percy Liang, and Jure Leskovec. 2021. QA-GNN: reasoning with language models and knowledge graphs for question answering. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT 2021, Online, June 6-11, 2021*, pages 535–546. Association for Computational Linguistics.

Zhi-Xiu Ye, Qian Chen, Wen Wang, and Zhen-Hua Ling. 2019. Align, mask and select: A simple method for incorporating commonsense knowledge into language representation models. *CoRR*, abs/1908.06725.

Logical Form Generation via Multi-task Learning for Complex Question Answering over Knowledge Bases

Xixin Hu, Xuan Wu, Yiheng Shu, Yuzhong Qu
State Key Laboratory for Novel Software Technology, Nanjing University, China
{xixinhu,xuanwu,yhshu}@smail.nju.edu.cn, yzqu@nju.edu.cn

Abstract

Question answering over knowledge bases (KBQA) for complex questions is a challenging task in natural language processing. Recently, generation-based methods that translate natural language questions to executable logical forms have achieved promising performance. These methods use auxiliary information to augment the logical form generation of questions with unseen KB items or novel combinations, but the noise introduced can also leads to more incorrect results. In this work, we propose GMT-KBQA, a **G**eneration-based KBQA method via **M**ulti-**T**ask learning, to better retrieve and utilize auxiliary information. GMT-KBQA first obtains candidate entities and relations through dense retrieval, and then introduces a multi-task model which jointly learns entity disambiguation, relation classification, and logical form generation. Experimental results show that GMT-KBQA achieves state-of-the-art results on both ComplexWebQuestions and WebQuestionsSP datasets. Furthermore, the detailed evaluation demonstrates that GMT-KBQA benefits from the auxiliary tasks and has a strong generalization capability.[1]

1 Introduction

Question answering over knowledge bases (KBQA) is the task of answering natural language questions based on the facts stored in knowledge bases (KBs). In recent years, an increasing number of KBQA methods arise with the emergence of large-scale KBs, such as Freebase (Bollacker et al., 2008), DBpedia (Lehmann et al., 2015) and Wikidata (Vrandečić and Krötzsch, 2014). A mainstream paradigm of KBQA methods is semantic parsing (SP) (Berant et al., 2013), where natural language questions are parsed into logical forms such as λ-DCS (Liang, 2013), SPARQL (Pérez et al., 2009), S-expression (Gu et al., 2021), etc. However, complex questions that involve reasoning over multiple entities, relations, or constraints remain a challenge for SP-based methods. Most of the SP-based methods use a pipeline including entity/relation linking, constraint detection, and logical form building (Singh et al., 2018; Hu et al., 2021). As complex questions require multiple entities and relations, errors introduced by previous linkers reduce the performance of the pipeline.

With the success of applying natural language generation to various tasks (Raffel et al., 2019; Rothe et al., 2020), recent KBQA methods (Cao et al., 2022; Yin et al., 2021; Gu et al., 2021) cast semantic parsing to a logical form generation task in a sequence-to-sequence (Seq2Seq) manner, fine-tuning pre-trained encoder-decoder models to generate logical forms from natural language questions. However, it is impractical for a simple Seq2Seq model to generate unseen entities and relations that never appear in the training set. To alleviate such cases, more generation-based methods (Huang et al., 2021; Das et al., 2021; Ye et al., 2021) augment logical form generation with auxiliary information such as linked entities (Huang et al., 2021), similar question-query pairs (Das et al., 2021), candidate logical forms (Ye et al., 2021), etc. Auxiliary information enhances the generalization capability of the generation models, but can also lead to incorrect results due to the noisy information introduced. We find that a Seq2Seq model can generate exact logical forms for around 92% questions in ComplexWebQuestions dataset if provided with golden entities and relations. However, the proportion drops drastically to 51% if the linking results are from practical linkers. This shows that Seq2Seq models can construct correct logical forms augmented with auxiliary information, and the quality of the auxiliary information has a great impact on the generated results.

Inspired by this discovery, we propose a generation-based KBQA method GMT-KBQA (**G**eneration via **M**ulti-**T**ask learning) that learns to

[1]Our code is available at https://github.com/HXX97/GMT-KBQA.

Proceedings of the 29th International Conference on Computational Linguistics, pages 1687–1696
October 12–17, 2022.

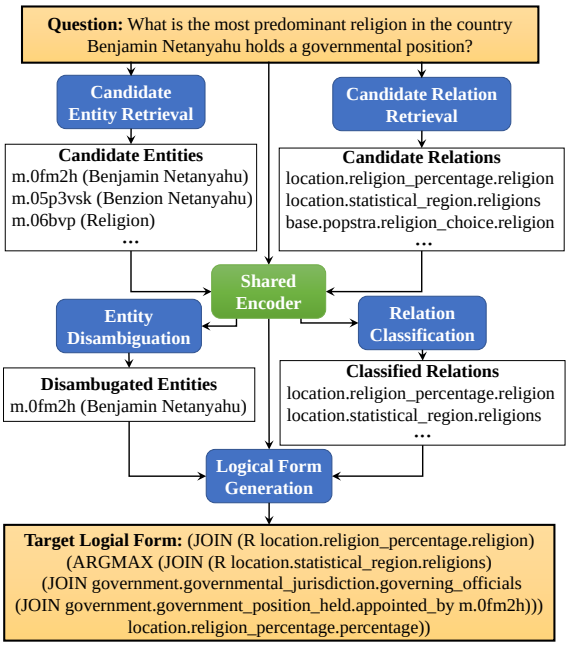

Figure 1: Overview of GMT-KBQA. The T5 encoder is learned with multiple tasks including the entity/relation classification and the logical form generation.

refine the auxiliary information and generate logical forms at the same time. Figure 1 shows the overview of our proposed method. The core of GMT-KBQA is an encoder-decoder model jointly trained with a translation task for logical form generation and two auxiliary tasks: entity disambiguation and relation classification. Our method attempts to improve logical form generation by sharing the parameters within related tasks. Instead of linking the entities and relations by off-the-shelf linkers before logical form generation, we retrieve candidate entities and relations in a dense space, leaving entity disambiguation and relation classification as auxiliary tasks.

The main contributions of this work are as follows:

1. We propose a generation-based KBQA method via multi-task learning (GMT-KBQA), where the logical form generation task is jointly trained with two auxiliary tasks. GMT-KBQA retrieves auxiliary information including candidate relations and entities through dense retrieval, which balances coverage and efficiency. The refined auxiliary information enables GMT-KBQA to generate unseen KB items.

2. Experimental results demonstrate that our method outperforms previous methods on both WEBQUESTIONSSP and COMPLEXWEBQUESTIONS. Further analysis shows that our method benefits from the multi-task learning framework and achieves better performances on both logical form generation and the two auxiliary tasks.

2 Methodology

This section details the GMT-KBQA method. Given a natural language question, we first retrieve auxiliary information including candidate entities and relations by dense retrieval. Then we refine the retrieved auxiliary information and generate the target logical form via multi-task learning. As shown in Figure 1, the three tasks share a common encoder, and each task has an individual layer on top of the encoder. Details of our method will be given in the following subsections.

2.1 Preliminaries

A knowledge graph is a collection of subject-relation-object triples in the form of (s,r,o), where s is an entity, r is a relation, and o can be entities or literals (e.g., text descriptions, numeric values, date-time, etc).

Given a natural language question, our method aims at generating a logical form that can be executed over the KB. Following Gu et al. (2021) and Ye et al. (2021), we use S-expressions as the target logical forms. Since most KB storage engines only support SPARQL queries, we finally convert S-expressions into equivalent SPARQL queries to get answers following Gu et al. (2021).

2.2 Retrieval of Auxiliary Information

Existing generation-based methods retrieve similar question-query pairs (Das et al., 2021) or enumerate candidate logical forms (Ye et al., 2021) as auxiliary information, but the coverage of cases cannot be guaranteed and the enumeration of logical forms can be time-consuming. Instead, we retrieve candidate entities and relations in a dense space as auxiliary information, which balances coverage and efficiency.

Candidate entity retrieval Most KBQA methods (Yih et al., 2015; Sun et al., 2019; Lan and Jiang, 2020; Huang et al., 2021; Ye et al., 2021) take entity linking as the first step, whereas the result of entity linking determines the upper bound of the final performance. We conduct entity retrieval to get candidate entities with high coverage, deferring entity disambiguation until logical form

generation by multi-task learning. Specifically, we use ELQ (Li et al., 2020), an end-to-end entity linking model through dense retrieval. To further improve the coverage of candidate entities, we use a large entity mention map provided by FACC1 project (Gabrilovich et al., 2013) to retrieve entities not linked by ELQ. For each question, we merge top-$k/2$ candidate entities from each linking model, to retain top-k ranked entities. If the number of candidate entities is less than k, entities randomly sampled from the training set will be supplemented.

Candidate relation retrieval Inspired by the zero-shot entity linking work with dense retrieval (Wu et al., 2020), we design a two-stage relation retrieval module utilizing the bi-encoder and cross-encoder architecture, which is shown in Figure 2.

In the first stage, we train a bi-encoder that embeds questions and relations into the same dense space with two independent BERT (Devlin et al., 2019) encoders.

Specifically, inspired by Das et al. (2021), for each question q, we mask entity mentions detected in candidate entity retrieval stage with $[BLANK]$ token. And we denote the question with entity mentions masked as τ_q.

In addition, a relation r is represented as:

$$r \mid \text{label}_r \mid \text{domain}_r \mid \text{range}_r$$

where label_r, domain_r and range_r are the meta descriptions of r in KB. For example, relation *location.location.time_zones* is represented as *location.location.time_zones* | *Time zone(s)* | *location.location* | *time.time_zone*. This enriched form of relation is denoted as τ_r. A question q and a relation r are encoded into vectors:

$$\begin{aligned} \mathbf{y}_q &= \text{BERTCLS}_1(\tau_q) \\ \mathbf{y}_r &= \text{BERTCLS}_2(\tau_r) \end{aligned} \quad (1)$$

where BERTCLS denotes the [CLS] representation of the input. The relevance score of question q and relation r is computed by dot-product:

$$s_b(q, r) = \mathbf{y}_q \cdot \mathbf{y}_r \quad (2)$$

For each pair of a question and its relevant relation (q, r_i), we randomly sample $B-1$ relations that are not in the logical form of the question to construct a batch consisting of B training pairs. The optimization goal is to maximize the score of the relevant relation against randomly sampled relations, and the loss is computed as:

$$\mathcal{L}(q, r_i) = -s_b(q, r_i) + \log \sum_{j=1}^{B} \exp(s_b(q, r_j)) \quad (3)$$

After training the bi-encoder, relation representations are cached for inference efficiency. For an incoming question, we embed it to a vector by the bi-encoder and then use FAISS (Johnson et al., 2019) to retrieve the nearest relations.

In the second stage, the retrieved relations for a question are re-ranked with a cross-encoder to get the most relevant relations. The cross-encoder is a single BERT model that takes the concatenation of the question and its candidate relation as input. Compared with the bi-encoder, the cross-encoder has deep cross attention between the question and the relation.

The input of cross-encoder is the concatenation of question q and candidate relation r. The relevance score of q and r is:

$$s_c(q, r) = \text{LINEAR}(\text{BERTCLS}([q; r])) \quad (4)$$

where LINEAR is a layer that projects the representation to a binary probability distribution. We train cross-encoder using cross-entropy loss. Finally, top-k candidate relations ranked by cross-encoder are retained.

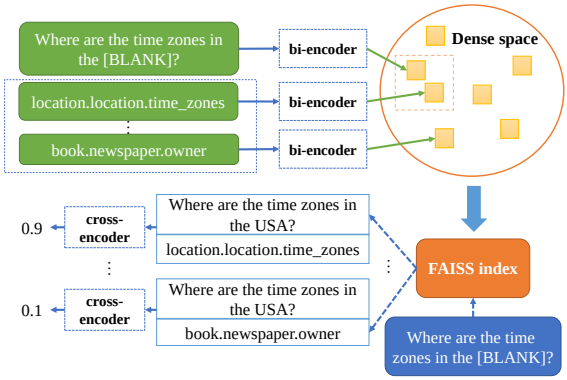

Figure 2: Overview of our two-stage relation retrieval method utilizing the bi-encoder and cross-encoder.

2.3 Logical Form Generation via Multi-task Learning

After the retrieval of auxiliary information including candidate entities and relations, we introduce a multi-task model that learns to refine the auxiliary information and generate the target logical form as

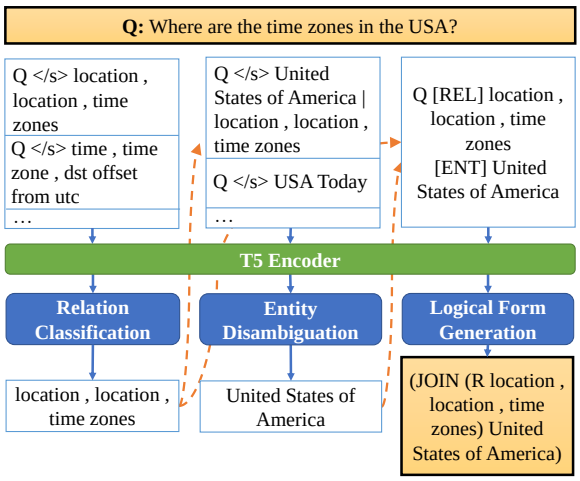

Figure 3: Multi-task model for logical form generation where "Q" is short for the original question. All tasks share a T5 encoder to encode inputs, and the results of prior stages will be leveraged in the latter stages as the dotted lines indicate.

Figure 3 shows. The backbone of our model is T5 (Raffel et al., 2019), an encoder-decoder structured Seq2Seq model that achieves strong performances on several generation tasks. We use a shared T5 encoder to obtain the representations of the inputs, and the representations are fed to individual networks for different tasks. We define two auxiliary tasks: relation classification and entity disambiguation to enhance the logical form generation task.

Relation classification Given a question q and its retrieved candidate relations R, the relation classification task aims to select the correct relations from R that compose the target logical form. We cast this task to a sentence-pair classification task. The concatenation of question q and relation r is fed to a T5 encoder, and it is represented by the average pooling of the encoder output:

$$\mathbf{y_{q,r}} = \textsc{AvgPool}(\textsc{T5Encoder}([q; r])) \quad (5)$$

Then the representation is projected to a scalar score through a linear projection, and then activated by a sigmoid function:

$$s(q, r) = \textsc{Sigmoid}(\textsc{Linear}(\mathbf{y}_{q,r})) \quad (6)$$

We use binary cross-entropy loss for the relation classification task:

$$\mathcal{L}^{\text{REL}} = -\frac{1}{k}\sum_{i=1}^{k}[u_i \cdot \log(s(q, r_i)) + (1 - u_i) \cdot \log(1 - s(q, r_i))] \quad (7)$$

where u_i denotes the classification label of relation r_i, and k is the number of candidate relations. The relations classified as positive are denoted as R_q.

Entity disambiguation The retrieved candidate entities usually contain ambiguity, where multiple KB entities are retrieved for one mention. To select the exact entities in the question, we cast entity disambiguation as a sentence-pair classification task similar to the relation classification task. Following Ye et al. (2021), we leverage adjacent relations to help determine if an entity should be linked by a mention. Specifically, we concatenate the label of entity e, with its adjacent KB relations in the form of:

$$\text{label}_e \mid r_1 \mid r_2 \mid r_3 \mid \ldots$$

This rich form of entity e is denoted as τ_e. Note that we only concatenate relations classified as relevant to the question by the relation classification task. The concatenation of question q with τ_e is fed to the shared T5 encoder, and the output is averagely pooled:

$$\mathbf{y}_{q,e} = \textsc{AvgPool}(\textsc{T5Encoder}([q; \tau_e])) \quad (8)$$

The relevance score of q and e is computed as:

$$s(q, e) = \textsc{Sigmoid}(\textsc{Linear}(\mathbf{y}_{q,e})) \quad (9)$$

Similarly, we use binary cross-entropy loss for the entity disambiguation task:

$$\mathcal{L}^{\text{ENT}} = -\frac{1}{k}\sum_{i=1}^{k}[v_i \cdot \log(s(q, e_i)) + (1 - v_i) \cdot \log(1 - s(q, e_i))] \quad (10)$$

where v_i is the classification label of entity e_i, and k is the number of candidate entities. The entities classified as positive are denoted as E_q.

Logical form generation The task of logical form generation is to generate the target logical form of question q given disambiguated entities E_q and classified relations R_q. Following previous generation-based methods (Das et al., 2021; Ye et al., 2021), we construct the inputs by concatenating the question q with relations in R_q and entities in E_q in the form as:

$$q \texttt{ [REL] } r_1 \texttt{ [REL] } r_2, \ldots \texttt{ [ENT] } \text{label}_{e_1} \texttt{ [ENT] } \text{label}_{e_2}, \ldots$$

This concatenation is denoted as τ_{all} and fed to the T5 encoder shared with the above-mentioned two auxiliary tasks to obtain the representations:

$$[\mathbf{h}_1, \mathbf{h}_2, \ldots, \mathbf{h}_n] = \text{T5ENCODER}(\tau_{all}) \quad (11)$$

where $\mathbf{h}_i$ denotes the representation of the i-th token of the input and n is the number of tokens in τ_{all}. Then we use a T5 decoder to decode the representations into a logical form token by token. Assuming that the target logical form consists of m tokens $a_1, \ldots, a_m$, we calculate cross-entropy loss with teacher forcing:

$$\begin{aligned} \mathbf{p}_j &= \text{T5DECODER}(a_1, \ldots, a_{j-1}, \mathbf{h}_1, \ldots, \mathbf{h}_n) \\ \mathcal{L}^{\text{GEN}} &= -\frac{1}{m}\sum_{j=1}^{m} \log \mathbf{p}_{j,a_j} \end{aligned} \quad (12)$$

where $\mathbf{p}_j$ denotes the probability distribution over the decoding vocabulary at the j-th step, and $\mathbf{p}_{j,a_j}$ represents the probability of token a_j.

Training Objective We jointly train the relation classifier, entity disambiguator, and logical form generator with a combined loss:

$$\mathcal{L} = \mathcal{L}^{\text{REL}} + \mathcal{L}^{\text{ENT}} + \mathcal{L}^{\text{GEN}} \quad (13)$$

This training objective enables the generator to learn from auxiliary tasks, where the target logical form not only supervises the generation task but also supervises entity disambiguation and relation classification.

3 Evaluation

3.1 Setups

Datasets All the experiments are conducted on WEBQUESTIONSSP (WebQSP) (Yih et al., 2016) and COMPLEXWEBQUESTIONS (CWQ) (Talmor and Berant, 2018) datasets. Both datasets are based on Freebase (Bollacker et al., 2008).

WebQSP consists of 4,737 questions labeled with SPARQL queries. Most questions of WebQSP require up to 2 hops of reasoning.

CWQ contains 34,689 questions with SPARQL queries. These questions are obtained by extending the questions in WebQSP to increase the complexity. Questions in CWQ may require up to 4-hops reasoning, making it quite challenging.

Hyperparameters We use T5-base and BERT-base-uncased implementation from HuggingFace[2]. The sample size B for training the bi-encoder in relation retrieval (Section 2.2) is set to 100, and the top 100 nearest relations are searched by the FAISS index. The number of candidate k is set to 10 for both entity and relation retrieval. Beam search is utilized in the decoding process, and we set beam size to 50 by default. For CWQ, our multi-task models are trained for 15 epochs, with training batch size set to 8 and inference batch size set to 4 due to GPU memory limits. For WebQSP, models are trained for 20 epochs with batch size set to 2 due to less data volume.

Implementation details Since neither WebQSP nor CWQ provides golden S-expressions, we follow the implementation of Ye et al. (2021) to convert a golden SPARQL query to its equivalent S-expression. WebQSP provides more than one SPARQL annotation for some questions, and we choose the shortest SPARQL query that can be successfully converted to S-expression.

The generation target of our model is normalized S-expression, where KB relations are split into tokens and entities are represented with their labels. Thus, a post-process step is needed to convert generated normalized S-expression to its original form. Specifically, entity labels are mapped into entity ids in the KB based on the output of the entity disambiguation task; normalized relations are converted back based on rules. Finally, S-expression is converted to SPARQL to be executed against KB in the same way of Gu et al. (2021).

In the training phase, the results of auxiliary tasks are not steady at the beginning, since the model parameters are not well-trained. Therefore, we do not concatenate the output of prior tasks as described in Section 2.3 for the first 5 training epochs.

In the inference phase, we utilize KB to validate generated logical forms. Given a question, we generate a bunch of logical forms by beam search and they are executed in turn until a non-empty query result is returned. It helps to filter invalid logical forms.

Metrics Following Das et al. (2021), we use the standard evaluation metrics, namely precision (P), recall (R), macro F1 (F1), and accuracy (Acc.).

[2] https://huggingface.co/

Method	CWQ		WebQSP	
	Acc.	F1	Acc.	F1
QGG (Lan and Jiang, 2020)	-	40.4	-	74.0
BART-large (Huang et al., 2021)	-	68.2	-	74.6
CBR-KBQA (Das et al., 2021)	67.1	70.0	69.9	72.8
ReTraCk* (Chen et al., 2021)	-	-	-	74.7
RnG-KBQA (Ye et al., 2021)	-	-	71.1	75.6
GMT-KBQA (Ours)	**72.2**	**77.0**	**73.1**	**76.6**

Table 1: QA evaluation results (%) on CWQ test set and WebQSP test set. * denotes using oracle entity linking results. Acc.: Accuracy.

	CWQ						WebQSP					
	Entity linking			Relation linking			Entity linking			Relation linking		
Methods	P	R	F1	P	R	F1	P	R	F1	P	R	F1
Retriever	71.6	**70.5**	68.8	84.9	74.2	77.3	53.9	**86.0**	62.6	66.3	68.7	65.4
GMT-KBQA	**78.8**	69.4	**72.3**	**85.2**	**81.0**	**81.4**	**79.9**	75.3	**76.1**	**74.1**	**73.9**	**72.1**

Table 2: Entity and relation linking evaluation (%) of our retriever and multi-task model.

3.2 Experimental Results

QA performance Table 1 summarizes evaluation results on both CWQ and WebQSP dataset. The results of other methods are taken from corresponding papers directly. The result shows that our method sets the new state-of-the-art on CWQ dataset by a large margin, surpassing CBR-KBQA (Das et al., 2021) by 5.1% accuracy and 7.0% F1. CBR-KBQA retrieves similar question-query pairs in the training set to augment logical form generation and introduces a revision step for relations not covered by the cases, whereas our method retrieves relevant entities and relations from the entire KB through dense retrieval, which balances coverage and efficiency. Huang et al. (2021) utilize auxiliary information including candidate entities and generate entity text labels instead of entity IDs. Our method achieves a notable advantage against Huang et al. (2021), indicating that our method makes better use of auxiliary information and is more capable of dealing with unseen KB items.

GMT-KBQA also achieves new state-of-the-art results on WebQSP dataset, where the questions are relatively simpler than CWQ. According to Table 1, our method outperforms existing methods even if they use oracle entity annotations (Chen et al., 2021). RnG-KBQA (Ye et al., 2021), the previous state-of-the-art on WebQSP, combines ranking and generation for both coverage and generalization. Our method obtains an increase of 2.0% accuracy and 1.0% F1 against RnG-KBQA although we only rely on the generation results. In summary, the results on CWQ and WebQSP datasets suggest that our method is effective in solving questions with different complexity.

Improvement on auxiliary tasks GMT-KBQA consists of three tasks, namely entity disambiguation, relation classification, and logical form generation. Apart from the experiments on QA tasks, also we evaluate the entity and relation linking results to show the impact of multi-task learning.

As shown in Table 2, for Retriever, candidate entities/relations retrieved in Section 2.2 are disambiguated with prediction scores in retrieval stage. For GMT-KBQA, candidate entities/relations are disambiguated with the output of entity disambiguation task and relation classification task described in Section 2.3 respectively. Experiment result indicates the improvement of our model on both entity linking and relation linking performance. The entity linking F1 is improved by 3.5% and 13.5% on CWQ and WebQSP, respectively. Although our disambiguation model sacrifices recall to some extent, we can observe a significant increase in precision. The experiment results further prove the effectiveness of our disambiguation ap-

Methods	CWQ F1	CWQ Δ	WebQSP F1	WebQSP Δ
GMT-KBQA	**77.0**	-	**76.6**	-
w/o Entity	72.7	-4.3	74.9	-1.7
w/o Relation	75.9	-1.1	75.1	-1.5
w/o Entity, Relation	74.0	-3.0	74.7	-1.9

Table 3: QA performance (%) of variants of our model. *Entity* and *Relation* are short for entity disambiguation task and relation classification task respectively.

Methods	CWQ F1	CWQ Δ	WebQSP F1	WebQSP Δ
T5-base	74.0	-	74.7	-
w/ Retrieval	71.4	- 2.6	72.9	- 1.8
w/ Oracle	**94.1**	+20.1	**94.8**	+20.1
GMT-KBQA	77.0	+ 3.0	76.6	+ 1.9

Table 4: Impact of auxiliary information on the generation model. *Retrieval*/*Oracle* are short for concatenating retrieved/oracle auxiliary information to input.

proach: the entity representation can be enriched with retrieved relations, which provides sufficient evidence for the disambiguation model. The relation linking F1 is also improved by 4.1% and 6.7% on two datasets, which proves the effectiveness of our multi-task model for relational linking.

3.3 Analysis

Ablation study To further illustrate the impact of each task and their combination, we evaluate QA results with different model variants, i.e., with task(s) removed. Table 3 shows 1) our final model substantially outperforms other model variants, indicating that our multi-task setting properly organizes and makes full use of all the tasks; 2) performance drops in general with more tasks removed, which shows the necessity of the auxiliary tasks.

Impact of auxiliary information Table 4 shows that a giant improvement can be achieved with oracle entities and relations, which confirms the importance and potential of utilizing auxiliary information.

However, for practical scenarios without the oracle, directly concatenating linking results leads to a decrease in performance compared to the vanilla generation model (T5-base in Table 4). The results are consistent with our motivation that auxiliary information could be better utilized via a multi-task setting instead of simple concatenation.

Methods	CWQ P	CWQ R	CWQ F1	WebQSP P	WebQSP R	WebQSP F1
T5-base	64.8	67.0	64.9	64.3	68.5	64.5
GMT-KBQA	**68.3**	**70.9**	**68.5**	**66.7**	**69.8**	**66.6**

Table 5: QA results (%) on test set questions with unseen KB items.

There are two advantages of GMT-KBQA to achieve the improvement. First, GMT-KBQA jointly trains the three tasks with a combined loss, improving the accuracy of each task and resulting in better overall performance. Second, when noisy candidate relations/entities are provided, for the model with simple concatenation, although given a generation loss, there's no explicit clue for the model to locate the noisy candidates, which leads to confusion in the optimization stage. As for GMT-KBQA with a multi-task setting, the loss of relation classification and entity disambiguation helps the model locate mis-classified candidates, making the optimization objective clearer. In this way, candidate relations/entities with higher precision and recall are given to the model, making it easier to generate correct logical forms.

Performance on unseen KB items To measure our model's capability of handling unseen relations and entities, experiments are conducted on questions from CWQ/WebQSP test set whose golden SPARQL contains unseen entities or relations (compared to the training set). Evaluation result in Table 5 indicates that GMT-KBQA contributes to 3.6% and 2.1% F1 increase in CWQ and WebQSP respectively. This experiment demonstrates that our method leads to improvement in generalization capability compared to the vanilla generation model.

Effect of beam size To study the impact of different beam sizes on the decoding stage, the QA performance of GMT-KBQA with different beam sizes is evaluated as shown in Table 6, where we also list the performance of other methods utilizing beam search. As the result implies, performance improves with a larger beam size, which indicates that beam search combined with the execution checking (Section 3.1) helps to discover valid and correct logical forms. The experiment further proves that our model still achieves great

Methods	CWQ					WebQSP				
	1	5	10	50	100	1	5	10	50	100
CBR-KBQA (Das et al., 2021)	-	70.0	-	-	-	-	72.8	-	-	-
BART-large (Huang et al., 2021)	55.5	-	60.9	-	68.2	67.5	-	73.6	-	74.6
GMT-KBQA(Ours)	**62.2**	**72.7**	**74.4**	**77.0**	-	**70.5**	**75.2**	**76.5**	**76.6**	-

Table 6: F1 metrics (%) with different beam sizes.

performance with smaller beam sizes, outperforming other methods with the same beam size.

Error analysis We analyze the questions not answered correctly by GMT-KBQA in the CWQ test set. The errors can be summarized as follows.

• **Structure generation error** (54.5%). An example of this error is the wrong choice of function. For instance, in question *"What is the youngest college that Harry S Truman attend?"*, GMT-KBQA fails to understand that "youngest" indicates the shortest existence time of a college, i.e., the latest foundation time. Therefore, our model applies "AGRMIN" operation instead of the correct "AGRMAX" function on the foundation time.

• **Relation linking error** (16.4%) and **entity linking error** (12.3%). Despite the efforts we put into relation classification and entity disambiguation, linking errors cannot be completely avoided. For example. in question *"What form of currency was used in the place where Nicolas Sarkozy was governor before the Euro was established?"*, linking correct relation *"location.country.currency_formerly_used"* requires understanding of past tense, i.e., *"was used"* in the question.

• **S-expression conversion** (11.1%) and **De-normalization** (4.6%). Some overly complex SPARQL queries do not have equivalent S-expression or the execution result of converted S-expression differs from the original SPARQL query. Apart from that, our de-normalization phase also causes some errors.

4 Related Work

Existing KBQA methods can be mainly divided into two categories: information retrieval-based methods (IR-based methods) and semantic parsing-based methods (SP-based methods).

IR-based methods (Bordes et al., 2015; Dong et al., 2015; Hao et al., 2017; Zhao et al., 2019) follow a retrieval-and-rank paradigm. They first retrieve a question-specific graph from the KB and then rank entities in the graph by their relevance to the question. For complex questions, recent IR-based methods turn their attention to graph retrieval (Sun et al., 2019; Saxena et al., 2020) and multi-hop reasoning over graphs (Zhou et al., 2018; He et al., 2021; Shi et al., 2021). Generally, IR-based methods fit into end-to-end training, but they lack interpretability because of the black-box reasoning process.

SP-based methods, closely relevant to our method, are more transparent compared with IR-based methods. They answer questions by parsing them into logical forms executable against KBs, including λ-DCS (Berant et al., 2013), SPARQL (Huang et al., 2021; Das et al., 2021), query graph (Yih et al., 2015; Bao et al., 2016; Lan and Jiang, 2020), and S-expression (Gu et al., 2021; Ye et al., 2021). Past SP-based methods parse questions with a bottom-up semantic parser (Berant et al., 2013) or iteratively generate and rank candidate query graphs (Yih et al., 2015; Lan and Jiang, 2020). However, previous semantic parsers have limited coverage for diverse complex queries (Lan and Jiang, 2020), and query graph generation methods suffer from the high computational cost of expanding the graphs (Qin et al., 2021). Recent methods (Zhang et al., 2019; Yin et al., 2021; Huang et al., 2021) take advantage of language generation models to directly generate executable logical forms from questions. As vanilla generation models do not generalize well to questions on KB items with novel combinations and unseen ones, Das et al. (2021) generate complex logical forms conditioned on retrieved similar questions along with their logical forms, but they need to add human-labeled cases to cover absent relations in the case memory. Ye et al. (2021) first rank a pool of candidate logical forms obtained by enumerating relation paths

over the KBs, and then generate the final logical form based on the question combined with top-ranked candidates. The enumeration of candidate logical forms involves a large search space on the KB, which is time-consuming and computationally expensive.

5 Conclusion

We present GMT-KBQA to improve the generalization capability of generation-based methods utilizing auxiliary information. GMT-KBQA first retrieves candidate entities and relations in dense space. Then, its multi-task learning framework learns to refine the auxiliary information along with generating target logical forms at the same time. Experimental results on two datasets, CWQ and WebQSP, show that our method sets new state-of-the-art by a large margin. The further analysis illustrates that our logical form generation task and auxiliary tasks benefit from each other and GMT-KBQA achieves strong performance for unseen KB items. In general, GMT-KBQA gives an insight into generating more accurate logical forms through auxiliary information retrieval and multi-task learning. Currently, answering questions that require in-depth understanding such as logical or commonsense reasoning remains a challenging task, and we will strengthen GMT-KBQA for such scenarios in the future.

Acknowledgements

This work was supported in part by the NSFC (62072224). The authors would like to thank all the participants of this work and anonymous reviewers.

References

Junwei Bao, Nan Duan, Zhao Yan, Ming Zhou, and Tiejun Zhao. 2016. Constraint-based question answering with knowledge graph. In *Proceedings of COLING 2016*, pages 2503–2514.

Jonathan Berant, Andrew Chou, Roy Frostig, and Percy Liang. 2013. Semantic parsing on freebase from question-answer pairs. In *Proceedings of the EMNLP 2013*, pages 1533–1544.

Kurt Bollacker, Colin Evans, Praveen Paritosh, Tim Sturge, and Jamie Taylor. 2008. Freebase: a collaboratively created graph database for structuring human knowledge. In *Proceedings of the SIGMOD 2008*, pages 1247–1250.

Antoine Bordes, Nicolas Usunier, Sumit Chopra, and Jason Weston. 2015. Large-scale simple question answering with memory networks. *arXiv preprint arXiv:1506.02075*.

Shulin Cao, Jiaxin Shi, Liangming Pan, Lunyiu Nie, Yutong Xiang, Lei Hou, Juanzi Li, Bin He, and Hanwang Zhang. 2022. KQA Pro: A large diagnostic dataset for complex question answering over knowledge base. In *Proceedings of the ACL 2022*.

Shuang Chen, Qian Liu, Zhiwei Yu, Chin-Yew Lin, Jian-Guang Lou, and Feng Jiang. 2021. Retrack: A flexible and efficient framework for knowledge base question answering. In *Proceedings of the ACL-IJCNLP 2021*, pages 325–336.

Rajarshi Das, Manzil Zaheer, Dung Thai, Ameya Godbole, Ethan Perez, Jay Yoon Lee, Lizhen Tan, Lazaros Polymenakos, and Andrew McCallum. 2021. Case-based reasoning for natural language queries over knowledge bases. In *Proceedings of the EMNLP 2021*, pages 9594–9611.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the NAACL-HLT 2019*, pages 4171–4186. Association for Computational Linguistics.

Li Dong, Furu Wei, Ming Zhou, and Ke Xu. 2015. Question answering over freebase with multi-column convolutional neural networks. In *Proceedings of the ACL-IJCNLP 2015*, pages 260–269.

E. Gabrilovich, M. Ringgaard, and A. Subramanya. 2013. Facc1: Freebase annotation of clueweb corpora, version 1 (release date 2013-06-26, format version 1, correction level 0).

Yu Gu, Sue Kase, Michelle Vanni, Brian Sadler, Percy Liang, Xifeng Yan, and Yu Su. 2021. Beyond iid: three levels of generalization for question answering on knowledge bases. In *Proceedings of the Web Conference 2021*, pages 3477–3488.

Yanchao Hao, Yuanzhe Zhang, Kang Liu, Shizhu He, Zhanyi Liu, Hua Wu, and Jun Zhao. 2017. An end-to-end model for question answering over knowledge base with cross-attention combining global knowledge. In *Proceedings of the ACL 2017*, pages 221–231.

Gaole He, Yunshi Lan, Jing Jiang, Wayne Xin Zhao, and Ji-Rong Wen. 2021. Improving multi-hop knowledge base question answering by learning intermediate supervision signals. In *Proceedings of the WSDM 2021*, pages 553–561.

Xixin Hu, Yiheng Shu, Xiang Huang, and Yuzhong Qu. 2021. EDG-based question decomposition for complex question answering over knowledge bases. In *Proceedings of the ISWC 2021*, volume 12922 of *Lecture Notes in Computer Science*, pages 128–145. Springer.

Xin Huang, Jung-jae Kim, and Bowei Zou. 2021. Unseen entity handling in complex question answering over knowledge base via language generation. In *Findings of EMNLP 2021*, pages 547–557.

Jeff Johnson, Matthijs Douze, and Hervé Jégou. 2019. Billion-scale similarity search with gpus. *IEEE Transactions on Big Data*, 7(3):535–547.

Yunshi Lan and Jing Jiang. 2020. Query graph generation for answering multi-hop complex questions from knowledge bases. In *Proceedings of ACL 2020*. Association for Computational Linguistics.

Jens Lehmann, Robert Isele, Max Jakob, Anja Jentzsch, Dimitris Kontokostas, Pablo N. Mendes, Sebastian Hellmann, Mohamed Morsey, Patrick van Kleef, Sören Auer, and Christian Bizer. 2015. Dbpedia - A large-scale, multilingual knowledge base extracted from wikipedia. *Semantic Web*, 6(2):167–195.

Belinda Z Li, Sewon Min, Srinivasan Iyer, Yashar Mehdad, and Wen-tau Yih. 2020. Efficient one-pass end-to-end entity linking for questions. In *Proceedings of the EMNLP 2020*, pages 6433–6441.

Percy Liang. 2013. Lambda dependency-based compositional semantics. *arXiv preprint arXiv:1309.4408*.

Jorge Pérez, Marcelo Arenas, and Claudio Gutierrez. 2009. Semantics and complexity of sparql. *ACM TODS*, 34(3):1–45.

Kechen Qin, Cheng Li, Virgil Pavlu, and Javed Aslam. 2021. Improving query graph generation for complex question answering over knowledge base. In *Proceedings of the EMNLP 2021*, pages 4201–4207.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J Liu. 2019. Exploring the limits of transfer learning with a unified text-to-text transformer. *arXiv preprint arXiv:1910.10683*.

Sascha Rothe, Shashi Narayan, and Aliaksei Severyn. 2020. Leveraging pre-trained checkpoints for sequence generation tasks. *TACL*, 8:264–280.

Apoorv Saxena, Aditay Tripathi, and Partha Talukdar. 2020. Improving multi-hop question answering over knowledge graphs using knowledge base embeddings. In *Proceedings of the ACL 2020*, pages 4498–4507.

Jiaxin Shi, Shulin Cao, Lei Hou, Juanzi Li, and Hanwang Zhang. 2021. Transfernet: An effective and transparent framework for multi-hop question answering over relation graph. In *Proceedings of the EMNLP 2021*, pages 4149–4158. Association for Computational Linguistics.

Kuldeep Singh, Arun Sethupat Radhakrishna, Andreas Both, Saeedeh Shekarpour, Ioanna Lytra, Ricardo Usbeck, Akhilesh Vyas, Akmal Khikmatullaev, Dharmen Punjani, Christoph Lange, et al. 2018. Why reinvent the wheel: Let's build question answering systems together. In *Proceedings of the WWW 2018*, pages 1247–1256.

Haitian Sun, Tania Bedrax-Weiss, and William Cohen. 2019. Pullnet: Open domain question answering with iterative retrieval on knowledge bases and text. In *Proceedings of the EMNLP-IJCNLP 2019*, pages 2380–2390.

Alon Talmor and Jonathan Berant. 2018. The web as a knowledge-base for answering complex questions. In *Proceedings of the NAACL-HLT 2018*, pages 641–651. Association for Computational Linguistics.

Denny Vrandečić and Markus Krötzsch. 2014. Wikidata: a free collaborative knowledgebase. *Communications of the ACM*, 57(10):78–85.

Ledell Wu, Fabio Petroni, Martin Josifoski, Sebastian Riedel, and Luke Zettlemoyer. 2020. Scalable zero-shot entity linking with dense entity retrieval. In *Proceedings of the EMNLP 2020*, pages 6397–6407. Association for Computational Linguistics.

Xi Ye, Semih Yavuz, Kazuma Hashimoto, Yingbo Zhou, and Caiming Xiong. 2021. Rng-kbqa: Generation augmented iterative ranking for knowledge base question answering. In *Proceedings of the ACL 2021*.

Scott Wen-tau Yih, Ming-Wei Chang, Xiaodong He, and Jianfeng Gao. 2015. Semantic parsing via staged query graph generation: Question answering with knowledge base. In *Proceedings of the ACL-IJCNLP 2015*.

Wen-tau Yih, Matthew Richardson, Christopher Meek, Ming-Wei Chang, and Jina Suh. 2016. The value of semantic parse labeling for knowledge base question answering. In *Proceedings of the ACL 2016*, pages 201–206.

Xiaoyu Yin, Dagmar Gromann, and Sebastian Rudolph. 2021. Neural machine translating from natural language to sparql. *Future Generation Computer Systems*, 117:510–519.

Haoyu Zhang, Jingjing Cai, Jianjun Xu, and Ji Wang. 2019. Complex question decomposition for semantic parsing. In *Proceedings of the ACL 2019*, pages 4477–4486.

Wenbo Zhao, Tagyoung Chung, Anuj Goyal, and Angeliki Metallinou. 2019. Simple question answering with subgraph ranking and joint-scoring. In *Proceedings of NAACL-HLT*, pages 324–334.

Mantong Zhou, Minlie Huang, and Xiaoyan Zhu. 2018. An interpretable reasoning network for multi-relation question answering. In *Proceedings of the COLING 2018*, pages 2010–2022.

CMQA: A Dataset of Conditional Question Answering with Multiple-Span Answers

Yiming Ju[*,1,2], **Weikang Wang**[*,3], **Yuanzhe Zhang**[1,2],
Suncong Zheng[3], **Kang Liu**[1,2], **Jun Zhao**[1,2],
[1] National Laboratory of Pattern Recognition, Institute of Automation, CAS, Beijing, China
[2] School of Artificial Intelligence, University of Chinese Academy of Sciences, Beijing, China
[3] Machine Learning Platform Department, Tencent
{yiming.ju, yzzhang, kliu, jzhao}@nlpr.ia.ac.cn
{daxianwang, congzheng}@tencent.com

Abstract

Forcing the answer of the Question Answering (QA) task to be a single text span might be restrictive since the answer can be multiple spans in the context. Moreover, we found that multi-span answers often appear with two characteristics when building the QA system for a real-world application. First, multi-span answers might be caused by users lacking domain knowledge and asking ambiguous questions, which makes the question need to be answered with conditions. Second, there might be hierarchical relations among multiple answer spans. Some recent span-extraction QA datasets include multi-span samples, but they only contain unconditional and parallel answers, which cannot be used to tackle this problem. To bridge the gap, we propose a new task: conditional question answering with hierarchical multi-span answers, where both the hierarchical relations and the conditions need to be extracted. Correspondingly, we introduce CMQA, a **C**onditional **M**ultiple-span Chinese **Q**uestion **A**nswering dataset to study the new proposed task. The final release of CMQA consists of 7,861 QA pairs and 113,089 labels, where all samples contain multi-span answers, 50.4% of samples are conditional, and 56.6% of samples are hierarchical. CMQA can serve as a benchmark to study the new proposed task and help study building QA systems for real-world applications. The low performance of models drawn from related literature shows that the new proposed task is challenging for the community to solve. CMQA can be accessed at https://github.com/juyiming/CMQA.

1 Introduction

Question answering (QA) is a challenging benchmark task, which can drive the development of natural language understanding (NLU) methods and has significant utility to users (Kwiatkowski et al., 2019). This research area has made significant progress with many sizable datasets and standard benchmarks. Notably, the span-extraction task is one of the most studied subtasks because of its wide applicability and easy evaluation characteristics.

Most existing span-extraction QA datasets (Rajpurkar et al., 2016; Yang et al., 2018; Choi et al., 2018; Kwiatkowski et al., 2019; Chen et al., 2020) only contain single-span samples, where the answer is a single text span in the context. However, limiting the answer to be a single span in the context might be restrictive since the answer of some questions can be multiple text spans in real-world QA scenarios. Take the case in Figure 1 for example, given the question asking about *'drugs for face acne redness'*, the corresponding drugs (green and blue highlighted text spans) appear in different parts of the context. A single span including all these drugs is extremely long, thus it is more suitable to use multi-span answers to answer this question. Moreover, when building the QA system for a real-world application, we found that the multi-span answers often appear with two characteristics:

First, multi-span answers can be caused by users lacking related domain knowledge and asking ambiguous questions. In such cases, questions might need multiple conditions to specify the circumstance. For example, in Figure 1, the user asked a brief question about *'drugs for acne redness'*. Since different drugs are needed according to the severity of symptoms:(*'A single appearance'* and *'the acne is much and continuous into pieces'*), the question is answered separately. It is misleading to give all answer spans without distinguishing the conditions. Moreover, answers of different conditions might be contradictory in some samples. For example, a user asked about the battery capacity of the iPhone. Since the battery capacity of the iPhone is different according to models, the multi-span an-

* means both authors contributed equally to this research.

Proceedings of the 29th International Conference on Computational Linguistics, pages 1697–1707
October 12–17, 2022.

Question: What are the anti-inflammatory drugs for face acne redness?

Context: Acne redness is generally the acute stage of acne. A single appearancex[1] indicates that the symptoms are not serious and can be treated with anti-acne drugs containing antibiotics[3]. External antibiotics treat acne by inhibiting or killing Propionibacterium acnes, reducing the content of free fatty acids, inhibiting the production of inflammatory chemokines and cytokines. Clindamycin phosphate gel[4] and mupirocin ointment[5] can be used for treatment. Both are used once or twice a day. Gently apply a layer of film on the area to be treated, and continue to use it for 3-4 weeks to evaluate the effect. During the illness, you need to pay attention to avoid squeezing the acne by hand. Especially when acne grows in the dangerous triangle area, it is strictly forbidden to squeeze to avoid ascending bacterial infection and cause cavernous sinus thrombophlebitis. You can use topical iodophor solution[6] for treatment if the acne is much and continuous into pieces[2]. A large area of redness indicates that the fungal infection is more serious and iodophor can kill fungus quickly. It is necessary to keep the local skin clean and hygienic. Wash your hands frequently, and fungicidal liquid soap is recommended.

Text spans:
condition: 1, 2; *coarse*: 3; *fine*: 4, 5, 6;
Relations:
condition-answer: 1-3, 2-6; *coarse-fine*: 3-4, 3-5

Structured Answer:
A single appearance → anti-acne drugs containing antibiotics → Clindamycin phosphate gel; mupirocin ointment
the acne is much and continuous into pieces → topical iodophor solution

Figure 1: An example of labels in CMQA. Text spans: *condition* (red), *coarse* (blue) and *fine* (green). Relations: *condition-answer* and *coarse-fine*. The example is translated from Chinese.

swers will contradict each other without specifying the condition.

Second, multiple answers in a sample might belong to different granularities, and there are often hierarchical relations among them. For example, in Figure 1, the answer span *'topical anti-acne drugs containing antibiotics'* is a type of drug, while *'Clindamycin phosphate gel'* and *'mupirocin ointment'* are two specific drug names of this type. If we only give all these answer spans in parallel, the user cannot get the granularity and hierarchical information of these drug names. In this case, we need to provide the answer granularity and the hierarchical relations for accuracy.

The analysis results in Table 4 show that about half of the multi-span samples in our data are conditional, and half of them are multi-granularity. The high proportion of these samples demonstrates that the summarized two characteristics might be widespread in some real-world QA scenarios. Though some recent span-extraction QA datasets include multi-span samples (Dua et al., 2019; Dasigi et al., 2019; Zhu et al., 2020), they only contain unconditional and parallel answers, which cannot be used to tackle this problem. To bridge the gap, we propose a new task: conditional question answering with hierarchical multi-span answers, where both the hierarchical relations and the conditions need to be extracted.

In this paper, we introduce **C**onditional **M**ultiple-span Chinese **Q**uestion **A**nswering dataset (CMQA) to track the new proposed task. Specifically, we pick out samples need to be answered with multiple text spans and use a new label strategy to annotate them. We labeled both answers and conditions if the sample is conditional. Moreover, the answer spans are labeled with different granularity: *coarse* and *fine*. *fine* means the answer is a specific thing, such as a specific time, person, to name a few. *coarse* means the answer span is a general term for a class of things, such as foods containing certain nutrients and people with certain characteristics. The hierarchical relations among answers of different granularities are also provided. As shown in Figure 1, labeled text spans in CMQA consist of three types: *condition*, *coarse* and *fine* (highlighted in the context). Labeled relations consist of two types: *condition-answer* and *coarse-fine*. A structured answer can be easily derived from these labels, which is clear and accurate. Furthermore, such labels are very helpful to reduce the burden of users if they want to read the full context.

The final release of CMQA consists of 7,861 multi-span QA pairs and 113,089 labels, where 50.4% of samples are conditional, and 56.6% are multi-granularity. In addition, we establish models as the baseline of CMQA. Traditional single-span QA models which search for the most likely token as the start/end of the answer are unsuitable for extracting multi-span answers, thus, the span extraction is cast as a sequence tagging problem (Segal et al., 2020). And we use competitive relation extraction methods drawn from related literature (Wu and He, 2019; Zhong and Chen, 2021) for predicting relations. Experimental results show that it is very difficult to extract all spans correctly in a sample. The error analysis shows that the main challenge is to judge whether a span of the correct entity type is an answer. Moreover, though the relation model can achieve high performance on some

traditional relation extraction datasets, the performance on our task is extremely poor, which makes giving structured answers like the case in Figure 1 difficult. The poor model performance demonstrates that the new proposed task is challenging for the community to solve.

Our contributions can be summarized as follows:

- We propose a challenge when solving questions from real-world users: conditional question answering with hierarchical multi-span answers.
- We introduce a new annotated Chinese question answering dataset: CMQA, which consists of questions that need to be answered with conditions and hierarchically answer spans.
- We establish models drawn from related literature as the baselines of CMQA. Experimental results show that the new proposed task is challenging.

2 Related Work

Most existing span-extraction QA datasets only contain single-span samples, which might be restrictive in some real-world QA scenarios. An important reason for this gap is the question collection methods when building the datasets. Question collection methods of QA datasets can generally be classified into two categories: creating questions by annotators and collecting questions from real-world users.

2.1 QA datasets with questions created by annotators.

Most span-extraction QA datasets consist of questions created by annotators. Some span-extraction QA datasets' questions are written by annotators who have first read the context containing the answer. For example, **SQuAD** (Rajpurkar et al., 2016) tasks crowdworkers with asking up to 5 questions about each paragraph and highlighting the corresponding answers in the paragraph. **HotpotQA** (Yang et al., 2018) tasks crowdworkers with asking a question about two given paragraphs from different Wikipedia pages and providing the answers. Tasking annotators with questioning and answering at the same time is suboptimal, which might cause questioners to ask questions based on a text span in the context. The asked questions are often simple reformulations of sentences in the context.

Field	Ratio
healthcare	77.6%
education	13.5%
government affair	4.0%
food	1.4%
digital product	0.3%
planting	0.2%
others	2.1%

Table 1: The fields of questions in the QA community.

To avoid this problem and create more natural and challenging questions, **NewsQA** (Trischler et al., 2017) only provides news article's headlines and its summary points to questioners. The full context is unseeable. Similarly, **QuAC** (Choi et al., 2018) also prevents the questioner from seeing the full context. Though effort has been made, there are inevitably differences between the human-created questions and the questions asked by users in real-world scenarios. The concerns are as follows: First, the user's question is often not based on a certain context in real-world QA scenarios. Second, crowdworkers might already know the form of the answer (single text span), which hints that they should ask questions with a single text span answer. Last, the problems created are often high quality, while a real-world question might sometimes be ambiguous.

2.2 QA datasets with questions collected from real-world users.

Besides manually created questions, some QA datasets use questions collected from queries from search engines. The answer form of these datasets is usually free text instead of text spans. For example, **MS Marco** (Nguyen et al., 2016) contains queries sampled from the Bing search engine. **DuReader** (He et al., 2018), which contains queries from Baidu search logs, chooses the free-text answer form, both use the free-text answer form and BLEU (Papineni et al., 2002) score as the evaluation metric.

Though there are datasets with questions from search engines that use text span as the answer form, the span length is usually longer than phrase level. Moreover, a screening mechanism is often used to filter questions. For example, **WIKIQA** (Yang et al., 2015) uses a single sentence as the answer form, and questions that cannot be answered with a single sentence are abandoned during the

construction. **Natural Questions** (Kwiatkowski et al., 2019) filter out questions cannot be answered with an entity or explanation. Moreover, Natural Questions provide long answers (paragraphs, tables, list items, or whole lists) besides short answers. A question might have a long answer but no short answer. These datasets show that a single span, especially a phrase-level text span, is not enough to answer real-world questions.

3 Dataset Construction

In this section, we describe the data construction process of CMQA.

3.1 Data Source

Samples in CMQA are collected from an authoritative Chinese QA community, where experts in related fields provide answers to questions from users. The community covers a lot of fields, as shown in Table 1, in which the healthcare field has received the most questions from users.

Since the provided answers often contain much secondary information such as descriptions and supplements, such as the example in Figure 1, the length of some answers is rather long (64.7% are longer than 100 words). The long answer texts are inconvenient to users who want to browse quickly and get information. Thus, we want to provide a concise answer (short text span) besides the full text. However, due to the existence of multi-span samples and the two characteristics summarized in Section 1, many questions are not suitable to be answered with a single text span. Thus, we let the annotators pick out samples that cannot be answered with a single text span to tackle the problem. We get 8,864 filtered samples from 25,000 samples and build our dataset based on these samples. Note that we have discard samples that may reveal the personal information of the questioner or containing offensive content during this phase.

3.2 Annotation Scheme

The annotation process is realized by three experts and three full-time annotators. As shown in Figure 2, the annotation scheme consists of 4 steps. Experts first formulate the annotation guidelines after pilot annotation. Then we train the annotators until all annotators think they can work independently. Then, we conduct an inter-annotator agreement (IAA) study to examine the annotation quality. We decide whether to continue the training phase according to the IAA score. In the last phase, we assign different parts of the samples to annotators for annotation.

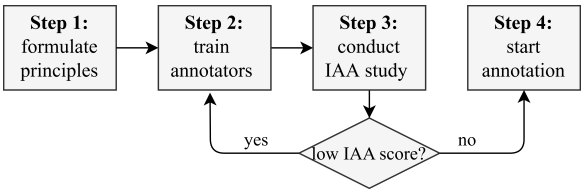

Figure 2: Pipeline of the annotation scheme.

Annotation Principles The main principles of annotation are summarized as follows:

- The text span should be as concise as possible.
- There is a multi-level hierarchical relationship, only the last level is marked
- The sample can be abandoned in the following situations:
  - If the context cannot answer the question.
  - All answers are adjacent and can be regarded as one.
  - Most of the words in the context are part of the answer. It is better to give the entire context as a long answer in this case. (Because the number of the answer spans of each sample is variable, we did not give a specific context length as the threshold to filter out samples. Experts give examples that should be abandoned, and the annotators make their judgments based on these examples.)
  - The question is a combination of multiple sub-questions.

Moreover, when the experts conduct pilot annotation, we find the mismatch in *condition* occurs mostly due to inclusion/exclusion some boundary words. Since these boundaries are difficult to determine sometimes, we decide to expand the boundary of the *condition* to its nearest stop words based on the annotation. In the final release, the average length of *condition* increased from 10.4 to 13.4 after expansion.

We has obtained the liecense of using the data from the community, which will be published with the final release.

Annotation Principles with examples are shown in Appendix, and the full text of the annotation principles will be publish with the dataset.

Dataset	Language	#QA	Conditional	Hierarchical	Question source	Answer amount	Answer length	Answer granularity
QUOREF (Dasigi et al., 2019)	English	2K	-	-	crowdsourcing	2.5	1.6	word/phrase
DROP (Dua et al., 2019)	English	5K	-	-	crowdsourcing	2.5	2.0	word/phrase
MASH-QA (Zhu et al., 2020)	English	29K	-	-	real-world users	4.2	19.3	sentence
CMQA	Chinese	8K	✓	✓	real-world users	7.5	4.5	word/phrase

Table 2: Comparison of CMQA with other QA datasets containing multi-span samples. Questions in QUOREF and DROP are collected by Mechanical Turk while questions in MASH-QA and CMQA are collected from real-world users. *Answer amount* refers to the average number of answer span per sample, and *answer length* refers to the average length per answer span. Note that only multi-span samples are compared.

Label type	IAA score
condition	95.1
coarse	86.1
fine	90.0
condition-answer	98.3
coarse-fine	97.6
all	92.7

Table 3: Inter-annotator agreement scores. ***all*** refers to calculating the micro F1-score of all label types.

Span type			Proportion
condition	*coarse*	*fine*	
✓	✓	✓	22.66%
✓	✓		0.47%
✓		✓	27.30%
	✓	✓	33.93%
	✓		1.40%
		✓	14.24%
50.43%	58.46%	98.13%	

Table 4: The division of CMQA according to text span types.

IAA Study We task annotators with annotating the same samples to take an inter-annotator agreement (IAA) study. We use the F1-score to compute an agreement score between two annotators. Following Gurulingappa et al. (2012); Legrand et al. (2020), we treat one annotator's annotation as the reference and the other's as the prediction. The final agreement score of three annotators is the average of the pair-wise agreement scores, formally as: $S_{abc} = \frac{1}{3}(S_{ab} + S_{ac} + S_{bc})$. We first evaluate the IAA score of different label types separately and then use micro F1-score as the metric to evaluate the IAA score of all labels. The inter-annotator agreement study result of 150 samples is shown in Table 3.

After training and inter-annotator agreement study, we assign different parts of the samples to the annotators for annotation. At last, we get 7,861 samples, in which each sample contains an average of 14.4 labels (8.8 text spans and 5.6 relations). To maximize the reusability of the dataset, we provide a pre-defined split of the dataset into training, development, and test sets in the final release, which consist of 5,861, 1000, and 1000 samples, respectively.

We choose F1-score instead of other conventional metrics such as the kappa coefficient (Cohen, 1960) because there is no one-to-one correspondence between annotations from different annotators in multiple-span scenarios. Some Named Entity Recognition (NER) datasets (Balasuriya et al., 2009; Srirangam et al., 2019) use kappa coefficient by examining tags of all tokens. However, since most tokens in the context are not parts of the answers in the Question Answering task, such a strategy will lead to a very high agreement score that fails to reflect the annotation quality.

4 Data Analysis

In this section, we analyze the label and question properties of the new dataset. Table 2 shows the comparison of CMQA with other QA datasets containing multi-span samples.

4.1 Label Properties

There are five label types in CMQA: *condition*, *coarse*, *fine*, *condition-answer* and *coarse-fine*. Since the only principle of picking out data to construct the dataset is that the question cannot be answered with a single text span, a sample in our dataset does not necessarily contain all five label types. We investigate the properties of different label types.

Text Span Properties Table 4 shows the division of CMQA according to the included text span types. As shown in Table 4, 50.4% of the samples in CMQA are conditional, and 56.6% are

Label type	Amount	Length
condition	1.3	13.4
coarse	1.6	6.1
fine	5.9	4.1
condition-answer	2.8	/
coarse-fine	2.8	/

Table 5: The average amount of labels and the average text span length per sample.

multi-granularity, which shows the necessity and effectiveness of our labeling strategy. Contexts of CMQA have an average of 182.3 words, with 18.8% annotated as conditions and 11.3% annotated as answers. The average amount and length of text spans are shown in Table 5.

Relation Properties Samples containing *condition* also contain related conditional answer spans, which means *condition-answer* is also included. However, containing *condition* doesn't mean all answers in the sample are conditional. There are samples where part of answers are conditional while the others are not, which accounts for 24.8% of *condition*-containing samples. There might be hierarchical relations between answer spans of different granularity. There are 12,458 *coarse* answers in CMQA, and 63.1% of them are connected to *fine* answers. A *coarse* answer can connect to several *fine* answers. The average amount of relations per sample is shown in Table 5.

4.2 Question Properties

Questions in CMQA are collected from users in the QA community. We heuristically identified question types for these questions. Specifically, we first manually observe the questions and summarize some frequently occurring question words. Then we assign question types based on whether these words are included in the question. We visualize the distribution of question types in Figure 3. As shown in Figure 3, the majority of questions in CMQA are about detailed information about specific facts, such as expense, duration, and food.

5 Experiments

This section establishes models using methods drawn from related literature as the baseline of CMQA and analyzes the experiment results.

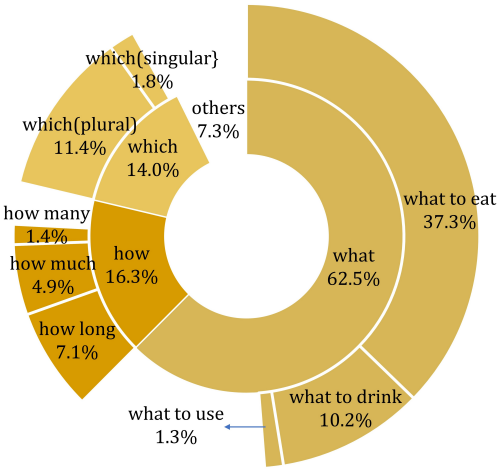

Figure 3: Question types in CMQA. The ring chart on the outer side shows the breakdown of the question types on the inner side, and the blank part indicates the question that does not belong to the summarized question types.

5.1 Models

We decompose the problem into two sub-tasks:

- Extracting text spans from context C as conditions and answers according to question Q.
- Predicting if a relation (condition-answer or coarse-fine) exists between two text spans.

Thus, our approach consists of a span model and a relation model. The span model first takes the context and questions as input and predicts conditions and answers. Then the relation model processes every pair of predicted text spans and judges if there is a relation between them. We find that sharing the contextual representations between the span and relation models will cause a performance decline. We hypothesize that using the same contextual representation to capture both span boundary information and span dependency information is suboptimal.

Span Model Traditional single-span extraction models, which search for the most likely token as the start/end of the answer, are unsuitable for the multiple-span extraction task. Thus, similar to Segal et al. (2020), we cast the task as a sequence tagging problem, which is demonstrated effective on multi-span samples in DROP and QUOREF. We experiment with the well-known BIO tagging (Huang et al., 2015). Concretely, we first concat the context and the question into one sequence and

use a pretrained language model to obtain contextualized representations hi for each input token t_i. The representation hi is then fed into a feedforward network to predict the probability distribution of the tag type. Two different modes are used to get the prediction: 1) **Separate Mode** uses different models to predict text spans of different types. These models share the same model structures but are trained independently. 2) **Merged Mode** uses one model to predict all text spans and using multiple begin tags ($B_{condition}$, B_{coarse} and B_{fine}) to distinguish span types.

Relation Model Because the type of relationship depends on the span type, we only need to predict if there is a dependency between a pair of text spans: s_a and s_b. We build the relation model following previous work (Wu and He, 2019; Zhong and Chen, 2021), which gets competitive results in several relation extraction benchmarks, such as ACE04 and SciERC (Luan et al., 2018). Concretely, we first obtain contextualized representations h_i for each input token x_i. Then we concatenate the token representations of the start positions of two text spans and obtain the span-pair representation: $h_{(s_a,s_b)} = [h_{start(a)}; h_{start(b)}]$, where $start(a)$ and $start(b)$ are the indices of start tokens of s_a and s_b. Finally, the span-pair representation $h_{(s_a,s_b)}$ will be fed into a feed forward network to make the binary classification. Moreover, we further investigate using span boundaries to enhance the relation model (**span assist**), where additional markers are used to highlight all conditions and answer spans in a sample. Concretely, we insert special markers (such as *'<fine>'* and *'</fine>'*) at the span boundaries in the input layer, and the embeddings of these markers are trainable vectors.

5.2 Evaluation Metrics and Experimental Details

We adopt two evaluation metrics: Exactly Match (EM) and F1-score. **EM:** Prediction of a sample is considered as correct the prediction is equal to the annotation, formally as $anno_{pre} = anno_{ref}$. **F1 score:** The exact steps of calculating F1 scores are the same as that in the IAA study, shown in Algorithm 1 in the Appendix.

In all experiments, we use bert-base-chinese as the encoder to get contextual representations. The implementation is based on Hugging-Face's *Transformers* library (Wolf et al., 2019). We report the averaged test set results of 3 runs for all the experiments. The relation models are trained with ground-truth span labels.

https://catalog.ldc.upenn.edu/LDC2005T09
https://github.com/google-research/bert

Label type/ Model	Separate		Merged	
	EM	$F1$	EM	$F1$
condition	42.3	71.8	47.3	72.6
coarse	32.8	65.0	33.1	65.1
fine	44.5	84.3	42.7	83.6
all	19.3	79.4	20.5	79.0

Table 6: Model performance on span extraction. ***all*** refers to evaluate all span types together. We use the micro F1-score to calculate F1-score of ***all***. Note that we only evaluate samples that contain the related span type in EM metric (all samples are evaluated in ***all***).

Model/ Label type	*condition*	*coarse*	*fine*
	acc	*acc*	*acc*
Separate	89.2	93.6	98.9
Merged	88.3	92.8	99.1

Table 7: Model performance on judging if a certain span type is included in the sample.

5.3 Experimental Results and Analysis

Span Extraction As shown in the experimental results summarized in Table 6, the separate and merged models get similar performance on the span extracting task. We can see from the table that model performance in EM is really poor, while the performance in F1 is much higher. An important reason is that there exist many simple spans besides those difficult to extract in one sample. For example, there might be answers connected in a one-word interval (e.g., *'ginger, pepper, garlic ...'*).

We analyze samples on the development set and find that 47.1% of fine-grained answers and 8.6% of coarse-grained answers are connected to others. In this case, the model will easily extract the others if one is recognized as the answer. However, to maintain the authenticity of the dataset, we did not eliminate or adapt these simple labels. The poor EM score in Table 6 shows that extracting all text span in one sample correctly is rather difficult. We further analyze model performance on judging if a certain span type is included in a sample. The results in Table 7 show the model can achieve very high performance (around 90% accuracy) on this

Question: *How long does it take to recover after a needle stick?*

Context: *... After topical application of chlortetracycline eye drops, erythromycin eye ointment and hot compress, you will usually recover after 3-4 days from mild local red nodules. If the pus is formed locally, it usually breaks in 2-3 days, and it will recover in 5-6 days ...*

Answer: *3-4 days, 5 to 6 days*

Prediction: *3-4 days, 2-3 days, 5 to 6 days*

Table 8: An example the merged model's wrong predictions on the development set.

Model/Label type	*corase*	*fine*
Separate	81.8%	77.2%
Merged	76.0%	80.6%

Table 9: Percentage of samples with *distinction error* in wrong-predicted samples from the development set.

test.

The multi-span extraction QA task is similar to NER in the output form. However, in NER, the model extracts all spans of the target entity type. But in the QA task, spans of the correct entity type might be either an answer or not, which should be judged based on the question and context. Thus the model cannot simply predict the answer type and extract all spans of this type as answers, which is one of the main characteristics of QA tasks and challenging for models to solve. Take the case in Table 8 for example, the model's prediction is *'3-4 days, 2-3 days, 5 to 6 days'*. However, *'2-3 days'* is not an answer to this question. We hypothesize that the model has learned that the answer type is 'period' but fails to understand the context and pick out the correct spans.

We denote such error as *distinction error*, which means all the predicted spans are in the correct type, but not all of them can be seen as an answer, or some correct spans in the context are ignored. We manually counted the *distinction error* amount from 200 randomly sampled wrong-predicted samples. Results in Tabel 9 show that *distinction error* occurs in a large proportion of wrong-predicted samples, which demonstrates the characteristic and main challenge of the span extraction task.

Label type/ Model	*condition-answer* EM	$F1$	*coarse-fine* EM	$F1$
Normal♠	3.9	47.9	16.2	54.6
Span-assist♠	6.5	51.1	22.1	57.2
Normal♢	0.4	21.6	7.3	34.9
Span-assist♢	1.8	24.0	13.0	36.5

Table 10: Results of relation models. ♠: results of using the ground-truth span labels. ♢: results of using the prediction of the merged span model.

Relation Extraction We report experimental results of two settings for the relation extraction task: using the ground-truth label as the input and using the prediction of the span model as the input. The results are shown in Table 10.

Due to the limited relation type in CMQA, the relation extraction task seems to be simpler than traditional relation extraction tasks containing multiple relation types (Augenstein et al., 2017; Luan et al., 2018; Gábor et al., 2018). Surprisingly, results in Table 10 show that the competitive method on these tasks performs poorly on CMQA. We hypothesis that one reason is that the text span and relation amount per sample in CMQA are higher than many traditional relation extraction datasets, which often focus on intra-sentence relations. And another reason is that the span type plays a very marginal role in relation extraction in CMQA. In contrast, the entity type is an important feature for judging relationships in most traditional relation extraction tasks (Zhong and Chen, 2021). These differences indicate new approaches needed to be developed to solve relation extraction in CMQA. Results in Table 10 show that introducing the boundary information of other spans can improve model performance. However, the improved model performance is still far from satisfying, which makes providing structured answers as Figure 1 very difficult.

6 Conclusion

In this paper, we propose a new challenge: conditional question answering with hierarchical multi-span answers, which might be widespread in multi-span QA in real-world scenarios. Moreover, we introduce CMQA, which contains conditional and hierarchical samples to study the new proposed task. Data analysis and experimental results show the main characteristics and challenges of CMQA, and the poor model performance demonstrates that

the proposed task is challenging for the community to solve. We believe CMQA can serve as a benchmark to study the new proposed task and help build more reliable and sophisticated QA systems.

Acknowledgements

This work is supported by the National Key Research and Development Program of China (No.2020AAA0106400), the National Natural Science Foundation of China (No.61922085, No.61976211, No.61906196) and the Key Research Program of the Chinese Academy of Sciences (Grant NO. ZDBS-SSW-JSC006). This work is also supported by TencentKG, the Open Project of Beijing Key Laboratory of Mental Disorders (2019JSJB06), Yunnan Provincial Major Science and Technology Special Plan Projects (No.202103AA080015) and in part of the Youth Innovation Promotion Association CAS.

References

Isabelle Augenstein, Mrinal Das, Sebastian Riedel, Lakshmi Vikraman, and Andrew McCallum. 2017. SemEval 2017 task 10: ScienceIE - extracting keyphrases and relations from scientific publications. In *Proceedings of the 11th International Workshop on Semantic Evaluation (SemEval-2017)*, pages 546–555, Vancouver, Canada. Association for Computational Linguistics.

Dominic Balasuriya, Nicky Ringland, Joel Nothman, Tara Murphy, and James R. Curran. 2009. Named entity recognition in Wikipedia. In *Proceedings of the 2009 Workshop on The People's Web Meets NLP: Collaboratively Constructed Semantic Resources (People's Web)*, pages 10–18, Suntec, Singapore. Association for Computational Linguistics.

Wenhu Chen, Hanwen Zha, Zhiyu Chen, Wenhan Xiong, Hong Wang, and William Yang Wang. 2020. HybridQA: A dataset of multi-hop question answering over tabular and textual data. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 1026–1036, Online. Association for Computational Linguistics.

Eunsol Choi, He He, Mohit Iyyer, Mark Yatskar, Wentau Yih, Yejin Choi, Percy Liang, and Luke Zettlemoyer. 2018. QuAC: Question answering in context. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 2174–2184, Brussels, Belgium. Association for Computational Linguistics.

Jacob Cohen. 1960. A coefficient of agreement for nominal scales. *Educational and psychological measurement*, 20(1):37–46.

Pradeep Dasigi, Nelson F. Liu, Ana Marasović, Noah A. Smith, and Matt Gardner. 2019. Quoref: A reading comprehension dataset with questions requiring coreferential reasoning. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 5925–5932, Hong Kong, China. Association for Computational Linguistics.

Dheeru Dua, Yizhong Wang, Pradeep Dasigi, Gabriel Stanovsky, Sameer Singh, and Matt Gardner. 2019. DROP: A reading comprehension benchmark requiring discrete reasoning over paragraphs. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 2368–2378, Minneapolis, Minnesota. Association for Computational Linguistics.

Kata Gábor, Davide Buscaldi, Anne-Kathrin Schumann, Behrang QasemiZadeh, Haïfa Zargayouna, and Thierry Charnois. 2018. SemEval-2018 task 7: Semantic relation extraction and classification in scientific papers. In *Proceedings of The 12th International Workshop on Semantic Evaluation*, pages 679–688, New Orleans, Louisiana. Association for Computational Linguistics.

Harsha Gurulingappa, Abdul Mateen Rajput, Angus Roberts, Juliane Fluck, Martin Hofmann-Apitius, and Luca Toldo. 2012. Development of a benchmark corpus to support the automatic extraction of drug-related adverse effects from medical case reports. *Journal of biomedical informatics*, 45(5):885–892.

Wei He, Kai Liu, Jing Liu, Yajuan Lyu, Shiqi Zhao, Xinyan Xiao, Yuan Liu, Yizhong Wang, Hua Wu, Qiaoqiao She, Xuan Liu, Tian Wu, and Haifeng Wang. 2018. DuReader: a Chinese machine reading comprehension dataset from real-world applications. In *Proceedings of the Workshop on Machine Reading for Question Answering*, pages 37–46, Melbourne, Australia. Association for Computational Linguistics.

Zhiheng Huang, Wei Xu, and Kai Yu. 2015. Bidirectional lstm-crf models for sequence tagging. *arXiv preprint arXiv:1508.01991*.

Tom Kwiatkowski, Jennimaria Palomaki, Olivia Redfield, Michael Collins, Ankur Parikh, Chris Alberti, Danielle Epstein, Illia Polosukhin, Jacob Devlin, Kenton Lee, Kristina Toutanova, Llion Jones, Matthew Kelcey, Ming-Wei Chang, Andrew M. Dai, Jakob Uszkoreit, Quoc Le, and Slav Petrov. 2019. Natural questions: A benchmark for question answering research. *Transactions of the Association for Computational Linguistics*, 7:452–466.

Joël Legrand, Romain Gogdemir, Cédric Bousquet, Kevin Dalleau, Marie-Dominique Devignes, William Digan, Chia-Ju Lee, Ndeye-Coumba Ndiaye, Nadine Petitpain, Patrice Ringot, et al. 2020. Pgxcorpus, a manually annotated corpus for pharmacogenomics. *Scientific data*, 7(1):1–13.

Yi Luan, Luheng He, Mari Ostendorf, and Hannaneh Hajishirzi. 2018. Multi-task identification of entities, relations, and coreference for scientific knowledge graph construction. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 3219–3232, Brussels, Belgium. Association for Computational Linguistics.

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. Bleu: a method for automatic evaluation of machine translation. In *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics*, pages 311–318, Philadelphia, Pennsylvania, USA. Association for Computational Linguistics.

Pranav Rajpurkar, Jian Zhang, Konstantin Lopyrev, and Percy Liang. 2016. SQuAD: 100,000+ questions for machine comprehension of text. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 2383–2392, Austin, Texas. Association for Computational Linguistics.

Elad Segal, Avia Efrat, Mor Shoham, Amir Globerson, and Jonathan Berant. 2020. A simple and effective model for answering multi-span questions. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 3074–3080, Online. Association for Computational Linguistics.

Vamshi Krishna Srirangam, Appidi Abhinav Reddy, Vinay Singh, and Manish Shrivastava. 2019. Corpus creation and analysis for named entity recognition in Telugu-English code-mixed social media data. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics: Student Research Workshop*, pages 183–189, Florence, Italy. Association for Computational Linguistics.

Adam Trischler, Tong Wang, Xingdi Yuan, Justin Harris, Alessandro Sordoni, Philip Bachman, and Kaheer Suleman. 2017. NewsQA: A machine comprehension dataset. In *Proceedings of the 2nd Workshop on Representation Learning for NLP*, pages 191–200, Vancouver, Canada. Association for Computational Linguistics.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Rémi Louf, Morgan Funtowicz, et al. 2019. Huggingface's transformers: State-of-the-art natural language processing. *arXiv preprint arXiv:1910.03771*.

Shanchan Wu and Yifan He. 2019. Enriching pre-trained language model with entity information for relation classification. In *Proceedings of the 28th ACM International Conference on Information and Knowledge Management, CIKM 2019, Beijing, China, November 3-7, 2019*, pages 2361–2364. ACM.

Yi Yang, Wen-tau Yih, and Christopher Meek. 2015. WikiQA: A challenge dataset for open-domain question answering. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 2013–2018, Lisbon, Portugal. Association for Computational Linguistics.

Zhilin Yang, Peng Qi, Saizheng Zhang, Yoshua Bengio, William Cohen, Ruslan Salakhutdinov, and Christopher D. Manning. 2018. HotpotQA: A dataset for diverse, explainable multi-hop question answering. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 2369–2380, Brussels, Belgium. Association for Computational Linguistics.

Zexuan Zhong and Danqi Chen. 2021. A frustratingly easy approach for entity and relation extraction. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 50–61, Online. Association for Computational Linguistics.

Ming Zhu, Aman Ahuja, Da-Cheng Juan, Wei Wei, and Chandan K. Reddy. 2020. Question answering with long multiple-span answers. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 3840–3849, Online. Association for Computational Linguistics.

A Appendix

A.1 Training Annotators

The annotation process is realized by three full-time annotators. All annotators have at least a high school degree and more than one year of full-time annotation working experience. We let the annotators annotate different parts of data according to the guidelines. All annotators and experts are in an online communication group. We encourage annotators to ask experts questions about samples they are not very sure of. We guarantee that experts will answer annotators' questions immediately in this phase. It takes about three days until all annotators think they can work independently. Each annotator has annotated over 100 samples in this phase. Note that this part of data is only used for training and will not be included in the final release.

A.2 Label Amount

We visualize the amount of each label type in Figure 4, where the first subgraph shows the total label amount of the dataset, and the rest of the subgraphs show that of each sample.

A.3 The exact steps of calculating F1 scores for IAA study and Experiment Evaluation

Algorithm 1 describes the exact steps in the evaluation procedure of IAA study and experiments.

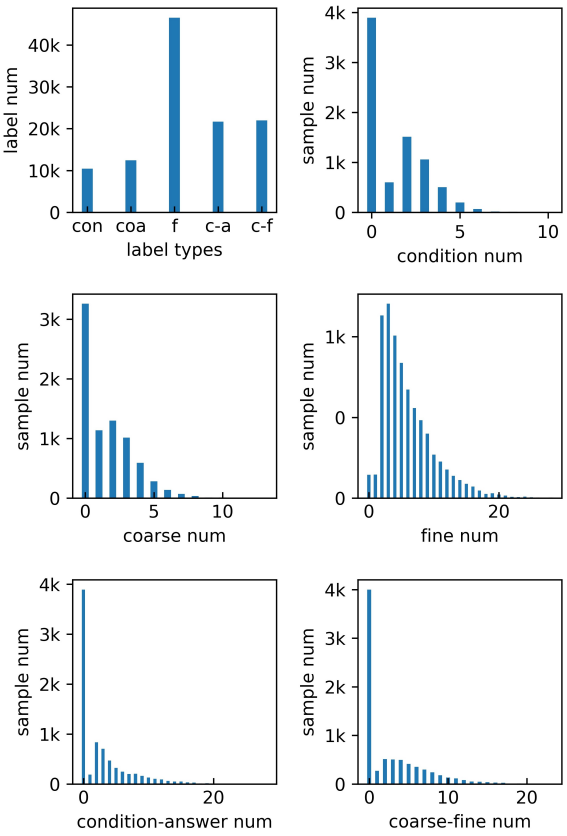

Figure 4: The amount of labels in CMQA. **c-a** refers to condition-answer and **c-f** refers to coarse-fine.

Algorithm 1 Evaluation Algorithm

Input: $Anno_{ref}$: the reference annotation; $Anno_{pre}$: the perdiction annotation;
Output: $F1 - score$;

function CAL_F1($Anno_{ref}, Anno_{pre}$)
  $recall_{mol} \leftarrow 0$
  $recall_{den} \leftarrow 0$
  $precision_{mol} \leftarrow 0$
  $precision_{den} \leftarrow 0$
  for $anno_{ref}, anno_{pre} \in zip(Anno_{ref}, Anno_{pre})$ **do**
    for $a \in anno_{ref}$ **do**
      if $a \in anno_{pre}$ **then**
        $recall_{mol} \leftarrow recall_{mol} + 1$
      end if
    end for
    for $a \in anno_{pre}$ **do**
      if $a \in anno_{ref}$ **then**
        $precision_{mol} \leftarrow precision_{mol} + 1$
      end if
    end for
    $recall_{den} \leftarrow recall_{den} + len(anno_1)$.
    $precision_{den} \leftarrow precision_{den} + len(anno_2)$
  end for
  $recall \leftarrow recall_{mol}/recall_{den}$
  $precision \leftarrow precision_{mol}/precision_{den}$
  $F1 \leftarrow 2*recall*precision/(recall+precision)$
  return $F1$

$Anno_= \{anno_1, anno_2, ..., anno_n\}$ refers to the annotation for n samples.

A.4 Annotation Principles with Examples

The main principles of annotation with examples are as follows:

- The text span should be as concise as possible. The annotator should exclude non-essential phrases. (e.g., question: *'What to eat for muscle growth?'* context: *'... You can eat some beef ...'*; The answer to this question should be *'beef'* instead of *'some beef'*.)
- If there is a multi-level hierarchical relationship, only the last level is annotated. (e.g., context: *'You can eat some vitamin-rich foods, such as fruits ... Apples are rich in vitamin C ...'*; The *coarse* answer to this question should be *'fruits'* instead of *'vitamin-rich foods.'*, and the *fine* answer should be *'Apples'*.)
- The sample can be abandoned in the following situations:
  - If the context cannot answer the question.
  - Although there are multiple answer spans in the context, these answers are adjacent and can be regarded as one. (e.g., *'ginger, pepper, garlic ...'*)
  - Most of the words in the context are part of the answer. It is better to give the entire context as a long answer in this case. (Because the number of the answer spans of each sample is variable, we did not give a specific context length as the threshold to filter out samples. Experts give examples of abandoned samples and let the annotators make their judgments based on these examples.)
  - The question is a combination of multiple sub-questions. (e.g., *What are the most suitable height and weight for long-distance running?'*) We split such questions into multiple sub-questions in our system.

To What Extent Do Natural Language Understanding Datasets Correlate to Logical Reasoning? A Method for Diagnosing Logical Reasoning.

Yitian Li[1,2†], **Jidong Tian**[1,2†], **Wenqing Chen**[1,2], **Caoyun Fan**[1,2], **Hao He**[1,2‡] and **Yaohui Jin**[1,2]
[1]MoE Key Lab of Artificial Intelligence, AI Institute, Shanghai Jiao Tong University
[2]State Key Lab of Advanced Optical Communication System and Network, Shanghai Jiao Tong University
{yitian_li, frank92, wenqingchen, fcy3649, hehao, jinyh}@sjtu.edu.cn

Abstract

Reasoning and knowledge-related skills are considered as two fundamental skills for natural language understanding (NLU) tasks such as machine reading comprehension (MRC) and natural language inference (NLI). However, it is not clear to what extent an NLU task defined on a dataset correlates to a specific NLU skill. On the one hand, evaluating the correlation requires an understanding of the significance of the NLU skill in a dataset. Significance judges whether a dataset includes sufficient material to help the model master this skill. On the other hand, it is also necessary to evaluate the dependence of the task on the NLU skill. Dependence is a measure of how much the task defined on a dataset depends on the skill. In this paper, we propose a systematic method to diagnose the correlations between an NLU dataset and a specific skill, and then take a fundamental reasoning skill, logical reasoning, as an example for analysis. The method adopts a qualitative indicator to indicate the significance while adopting a quantitative indicator to measure the dependence. We perform diagnosis on 8 MRC datasets (including two types) and 3 NLI datasets and acquire intuitively reasonable results. We then perform the analysis to further understand the results and the proposed indicators. Based on the analysis, although the diagnostic method has some limitations, it is still an effective method to perform a basic diagnosis of the correlation between the dataset and logical reasoning skill, which also can be generalized to other NLU skills.

1 Introduction

Machine reading comprehension (MRC) and natural language inference (NLI) are used to benchmark natural language understanding (NLU) capabilities. Although a large number of NLU-related datasets have been proposed (Yang et al., 2018; Yatskar, 2019), it is hard to evaluate correlations between the dataset and NLU-related skills due to the lack of benchmark method (Richardson et al., 2020), which may create an obstacle to choose appropriate dataset for specific skill training. According to a widely accepted discourse comprehension theory, construction-integration model (Kintsch, 1991), there are two processes for humans to understand language: 1) understanding the concepts and discourses in the text and building their relationships with the real world; 2) synthesizing these concepts and discourses to form a consistent understanding. These two processes mainly correspond to two types of skills in NLU: Knowledge-related skills and reasoning skills (Baral et al., 2020; Boratko et al., 2018; Tian et al., 2021). Previous researches concentrate on evaluating the correlation between datasets and knowledge, such as assessing what types of knowledge are required to complete an NLU task (Sap et al., 2019a; Rogers et al., 2020) and offering various knowledge supplements based on those needs (Feng et al., 2020). However, compared to knowledge-based characteristics, reasoning abilities are more difficult to identify and quantify directly. Besides, some existing work using probes (supervised models trained to predict properties) mainly focuses on linguistic tasks (Hewitt and Liang, 2019; Conneau et al., 2018) and few studies diagnose the logical discrepancies between datasets.

Based on different deduction methods, reasoning skills can be divided into logical reasoning, coreferential reasoning, numerical reasoning, causal reasoning, etc (Sugawara et al., 2017). Among all these reasoning skills, logical reasoning is a fundamental skill widely used to understand natural language (Bhagavatula et al., 2020). In this paper, we take logical reasoning as an example and construct a systematic method to analyze the correlations between NLU datasets and a specific

† These authors contributed equally.
‡ Corresponding author.

Proceedings of the 29th International Conference on Computational Linguistics, pages 1708–1717
October 12–17, 2022.

reasoning skill. The correlations include two aspects: significance and dependence. 1) significance aims to judge whether a dataset includes sufficient (explicit or implicit) logical expressions; 2) dependence is used to measure how much the task on a dataset depends on the logical reasoning skill. To aid the diagnosis, a logical probe, consisting of a probe model and a probe dataset, is introduced. We select 8 MRC datasets (including two types) and 3 NLI datasets to perform diagnosis. We create a qualitative and quantitative indicator to reflect the association between the dataset and logical reasoning after training the probe model on various datasets in various ways. The results show: 1) Most NLI datasets are relatively strongly correlated to logical reasoning. 2) As for the comprehensive MRC datasets (Type1), the correlations are moderate which means that logical reasoning is not the only dominant reasoning skill in this type; 3) The correlations of different MRC datasets for specific anticipations (Type2) vary remarkably according to their purposes of design. Further cause analysis, which is consistent with the results, confirms the rationality of our method to a certain degree. In conclusion, this method offers a reasonable view of exploring the correlations between datasets and logical reasoning.

Our contributions are as follows:

- We propose a systematic method to diagnose the correlations between NLU datasets and reasoning skills and take logical reasoning as an example to validate the effectiveness of this method.
- In particular, two indicators are introduced to evaluate the correlation between the NLU dataset and reasoning skill.
- We conduct extensive experiments on 11 NLU datasets from both qualitative and quantitative analyses. Results show that Winogrande is the only dataset unable to judge the significance based on qualitative analysis, while QASC, SNLI and MNLI show relatively high dependence on logical reasoning based on quantitative analysis.

2 Related Work

NLU Datasets. Recently, the number of NLU datasets has exponentially increased (Zeng et al., 2020). Such datasets mainly include MRC datasets and NLI datasets. MRC datasets, such as SQuAD (Wang et al., 2016), CoQA (Reddy et al., 2019) and DROP (Dua et al., 2019) are designed to test whether machines can answer the text-related questions or not, while NLI datasets, such as SNLI (Bowman et al., 2015) and MNLI (Nangia et al., 2017), are constructed to explore whether models can detect inferential relationships between natural language descriptions (Richardson et al., 2020) or not. Overall, all these datasets aim to define tasks to evaluate whether machines can understand the natural language as humans do.

Analysis for NLU. Recently, more and more researches focus on what models have really learned in NLU tasks. In this field, some researches attempt to understand the knowledge (Ghosal et al., 2021; Fang et al., 2021) and reasoning skills (Richardson et al., 2020) in language models, while others mainly focus on systematic evaluations of NLU models (Tenney et al., 2019; Ribeiro et al., 2020). These studies have provided foundations to further understand NLU. In addition, some researchers begin to pay attention to the analysis of NLU datasets (Baradaran et al., 2020). For example, Sugawara et al. (2020) have provided an ablation-based method to understand the tasks defined on the NLU datasets. However, such researches are still rare and further investigations are required.

3 Methodology

In this paper, we limit our investigation to a fundamental form of logical reasoning: conjunctive implications with negation (Musen and Lei, 1988) which uses multiple conditions to derive the final conclusion and can be expressed as: $[(\neg)p_1, (\neg)p_2, \cdots, (\neg)p_n] \rightarrow q$, where $p_i, (i = 0, 1, \cdots, n)$ represents each condition and q is the conclusion. Although logical reasoning has many complex forms, conjunctive implication with negation is the most commonly used one in daily conversation and can cover most situations in natural language (Allwood et al., 1977).

Our method adopts control variates to control what models learn. Supported by a logical probe, we train models in the different manners on the diverse datasets to adjust the skills mastered by models. Based on the method, we design a qualitative indicator and a quantitative indicator to indicate significance and dependence, respectively. Next, we will firstly introduce the logical probe and then introduce two indicators and two corresponding

processes.

3.1 Logical Probe

Logical probe includes a probe dataset and a probe model. The probe dataset has two main functions: 1) to train the probe model and enable it to master logical reasoning without increasing extra knowledge; 2) to test whether a model has mastered logical reasoning or not. Therefore, the probe dataset is required to meet the following three conditions:

- The dataset will involve as little explicit and implicit knowledge as possible, while expressions in the dataset should conform to natural language form.
- It should contain a large number of logical expressions and the task defined on the dataset should relate strongly to logical reasoning.

As for the probe model, it is the original model that will be trained on the different datasets in diverse manners. Therefore, the probe model is required to meet the following three conditions:

- The probe model itself can be used as a knowledge base.
- The knowledge stored in the model's weights can be updated after being trained in a specific manner.
- The model can master logical reasoning after being trained on the probe dataset.

3.2 Qualitative Diagnosis

A qualitative process is designed to diagnose the significance of a dataset to logical reasoning which indicates whether a dataset includes sufficient logical materials to enable the model to master logical reasoning. To perform qualitative diagnosis, we firstly involve a presupposition that as long as there exists one model that can master logical reasoning to a certain degree through the dataset to be diagnosed, we can assert that the dataset is significant to logical reasoning. According to the presupposition, the qualitative process includes two steps shown in Fig 1 and Alg 1. The first step is to train the probe model M_0 on the dataset D to be diagnosed in the supervised manner. Then we acquire a trained model M_1 which has acquired the dominant patterns in D. The second step is to test M_1 on the probe dataset P to acquire the metric R_1.

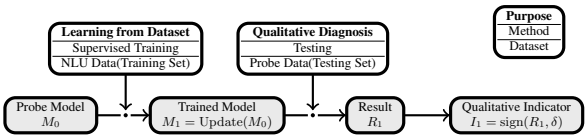

Figure 1: Qualitative process. The probe model is first trained on the NLU dataset to acquire significant skills and tested on the probe dataset. Before testing, we re-train the full connected classifier with fixed M1.

Note that in the testing step, we re-train the fully-connected classifier with fixed M_1 before testing. The qualitative indicator can be calculated by the Eq 1.

$$I_1 = \text{sign}(R_1, \delta) = \begin{cases} +, & R_1 \geq \delta \\ -, & R_1 < \delta \end{cases} \quad (1)$$

Where "$+$" means "significant to the logical reasoning" while "$-$" means "unable to judge" (the qualitative diagnosis has a natural limitation in which we cannot traverse all models to fully verify the existential presupposition). δ is the threshold to judge the significance (Since the probability of random selection for the binary classification problem is 0.5, we can approximate that R_1 is equivalent to the result of random selection if $R_1 < \delta = 0.55$).

Algorithm 1 Qualitative Process.

Require: NLU datasets $D = [D_1, D_2, \cdots, D_n]$; Probe dataset P; Probe Model M_0

Ensure: Qualitative Indicator, $I_1 = [I_1^1, I_1^2, \cdots, I_1^n]$

```
Initialize M_0 with pre-trained parameters;
Initialize I_1 = []
for i = 1 to n do
    M_1^i = SupervisedTrain(M_0, D_i)
    R_1^i = Test(M_1^i, P)
    I_1^i = Sign(0, R_1^i − δ)
    I_1.Append(I_1^i)
end for
return I_1;
```

3.3 Quantitative Diagnosis

A quantitative process is designed to diagnose the dependence of a dataset on logical reasoning, which aims to answer how much a task defined on the dataset depends on the logical reasoning skill. Here we make a hypothesis that if the only difference between the two models is whether logical reasoning has been mastered, the gap of the

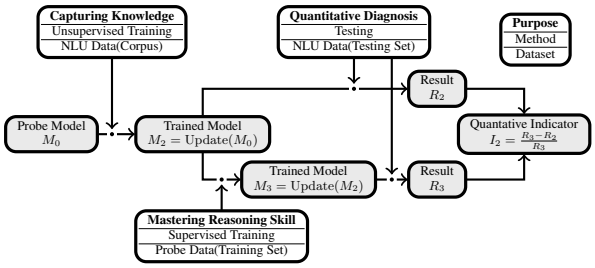

Figure 2: Quantitative process. The probe model is firstly trained on the NLU corpus in an unsupervised mannert to update its knowledge related to the NLU dataset (get M_2). The model is trained on the probe dataset to master logical reasoning skills (get M_3). We re-train the full connected classifier with fixed M2/M3 before testing. Then, M_2 and M_3 are tested on the NLU dataset, to get two results R_2 and R_3, respectively. Finally, quantitative indicator I_2 is calculated using the difference of R_3 and R_2.

performances can indicate the dependence on logical reasoning. Based on the hypothesis, we define a three-step process to calculate the quantitative indicator: 1) training the probe model M_0 on the NLU dataset D in the unsupervised manner to abundant knowledge and getting model M_2; 2) training M_2 on the probe dataset P and getting the second model M_3; 3) Testing M_2 and M_3 on D and getting results R_2 and R_3 (Testing step is similar to it in the qualitative diagnosis). The quantitative indicator is calculated by Eq 2.

$$I_2 = \frac{R_3 - R_2}{R_3} \times 100 \tag{2}$$

The quantitative process is shown in the Fig 2 and Alg 2. The test algorithm in both Alg 1 and Alg 2 is shown in Alg 3.

Algorithm 2 Quantitative Process.

Require: Diagnosed datasets $D = [D_1, D_2, \cdots, D_n]$; Probing dataset P; Probing Model M_0

Ensure: Quantitative Indicator, $I_2 = [I_2^1, I_2^2, \cdots, I_2^n]$

Initialize M_0 with pre-trained parameters;
Initialize $I_2 = []$
for $i = 1$ to n **do**
  $M_2^i = \text{UnsupervisedTraining}(M_0, D_i)$
  $R_2^i = \text{Testing}(M_2^i, D_i)$
  $M_3^i = \text{SupervisedTraining}(M_2^i, P)$
  $R_3^i = \text{Testing}(M_3^i, D_i)$
  $I_2^i = \frac{R_3 - R_2}{R_3} \times 100$
  $I_2.\text{Append}(I_2^i)$
end for
return I_2;

Algorithm 3 Test (including Re-train FC Classifier).

Require: Model M; Dataset D

Ensure: Metric R

Initialize fully-connected classifier C randomly;
Fix M
$C' = \text{SupervisedTrain}([M, C], D)$
$R = \text{CalMetric}([M, C'], D)$
return R;

4 Diagnosis

4.1 NLU Datasets to Be Diagnosed

We select 11 NLU datasets to perform diagnosis. These datasets can be classified into three types: NLI datasets and two types of MRC dataset. The NLI datasets include three commonly used ones: SNLI (Bowman et al., 2015), MNLI (Nangia et al., 2017) and αNLI (Bhagavatula et al., 2020). Type 1 MRC datasets (comprehensive MRC datasets) include BoolQ (Clark et al., 2019), DROP (Dua et al., 2019) and CODAH (Chen et al., 2019), all with tasks requiring diverse NLU skills to solve. Type 2 MRC datasets (specific MRC datasets) are constructed to benchmark one dominant NLU skill, including QASC (Khot et al., 2020), ReCLor (Yu et al., 2020), SocialIQA (Sap et al., 2019b), QuaRTz (Tafjord et al., 2019) and Winogrande (Sakaguchi et al., 2020).

In order to make the diagnosis results on different datasets comparable, we process the NLI and MRC datasets separately. Given an item $D = [context, question, answer]$ in an MRC dataset, we combine $question$ and $answer$ to a $statement$ and transfer the D to a binary classification represented by $D' = [context, statement, label]$, where $label$ is $True$ or $False$.As for the item $D = [fact_1, fact_2, label]$ in an NLI dataset, we just set $fact_1$ as $context$ and $fact_2$ as $statement$, and change $label \in \{entailment, neural, contradiction\}$ to $label \in \{True, False\}$ ($entailment$ is $True$; $neural$ and $contradiction$ are $False$ based on the closed world assumption, CWA). To ensure the balance of samples, we adjust the number of negative samples by removing or constructing. The statistics of the processed datasets are shown in Table 2.

Paras.	Unsupervised.	Supervised.	Training.
batch size	16	16	16
lr	–	$1e^{-5}$	$1e^{-3}$
lr for BERT	$5e^{-6}$	$5e^{-6}$	–
decay rate	0.9	0.9	0.8
l2 coeff.	$1e^{-5}$	$1e^{-5}$	$1e^{-5}$
early stop	5	5	5
epochs	20	20	20
optimizer	ADAMW	ADAMW	ADAMW

Table 1: Hyper-parameter settings.

Datasets	Average Length	No. Data (k)		
		Train	Dev.	Test
SNLI	21	300.0	6.0	6.0
MNLI	30	245.0	13.5	13.4
αNLI	33	339.0	3.0	6.0
BoolQ	102	4.6	2.4	2.4
DROP	197	132.0	18.5	14.7
CODAH	17	3.5	1.0	1.0
QASC	108	13.9	1.9	2.4
ReCLor	97	8.2	1.0	1.0
SocialIQA	23	33.4	2.0	2.2
QuaRTz	37	5.3	0.7	1.3
Winogrande	20	78.3	2.5	2.5
Probe Dataset	103	70.1	9.9	20.0

Table 2: Statistics of NLU datasets to be diagnosed and the probe dataset.

We trained our model on NVIDIA 16GB Tesla P100 and V100 GPUs. For fine-tuning BERT, the model costs around 22GB memory with the batch size of 16. The hyper-parameters are shown in Table 1.

4.2 Probe Dataset

In the diagnosis, we use the dataset proposed by Clark et al. (2020) as the probe dataset (An example of which is shown in Fig 3) for it clearly satisfying the last two conditions. Based on the construction rules (constructing propositions through meaningless sentences) of the probe dataset, it will not involve explicit knowledge into models. To validate that the implicit knowledge involved by the dataset can be negligible, we also make a simple statistical analysis in Preliminary 1. Therefore, this dataset is suitable to be the probe dataset. In practice, we use the 3-hop dataset to train the probe model because Clark et al. (2020) have illustrated that Transformers trained on this dataset has good generalization capability. Statistical information of the probe dataset is shown in Table 2.

Facts:
F1: Anne is quiet.
F2: Bob is blue.
F3: Bob is quiet.
F4: Charlie is blue.
Rules:
R1: All smart, blue things are green.
R2: Quiet things are red.
R3: If Bob is blue then Bob is red.
R4: All quiet, red things are smart.
Statements:
S1: Bob is green.
L2: True (F3→R2→R4→F2→R1→S1)
S2: Anne is not red.
L2: False (F1→R2→ ¬S2)

Figure 3: An example of the probe data.

4.3 Probe Model

BERT (Devlin et al., 2019) naturally meets all three conditions of the probe model. Firstly, BERT itself can be used as the knowledge base since it contains a large amount of knowledge and its architecture has the ability to update knowledge after being trained in the unsupervised manner (the same way to train the language model) (Rogers et al., 2020; Petroni et al., 2019). Secondly, it has been proved that Transformers, the basic architecture of BERT, can master the generalizable logical reasoning (Clark et al., 2020; Hahn et al., 2020). Therefore, we select BERT as the probe model. Although BERT can satisfy all three conditions, we still need to prove that no matter what initial weights are, the architecture of Transformers always has the ability to master generalizable logical reasoning. These two preliminaries are shown in Preliminary 2 and Preliminary 3, respectively.

4.4 Preliminaries

Preliminary 1: Validating that implicit knowledge involved by the probe dataset can be negligible. Implicit knowledge is always hidden in the representative vectors of words. We have made a simple statistics that the vocabulary size of the probe dataset is 67. This means that the proportion of overlapping words to the NLU dataset's vocabulary is very low, almost negligible, which is intuitive evidence that implicit knowledge involved by the probe dataset can be negligible.

Preliminary 2: Proving that BERT will not master logical reasoning after being trained in the unsupervised manner. We conduct a controlled experiment to compare three kinds of models trained and tested on the probe dataset. We use

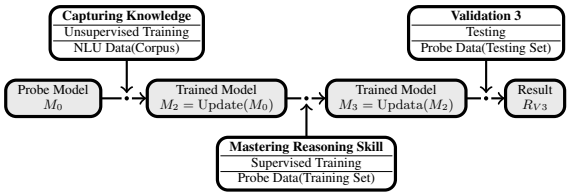

Figure 4: The process for Preliminary 3.

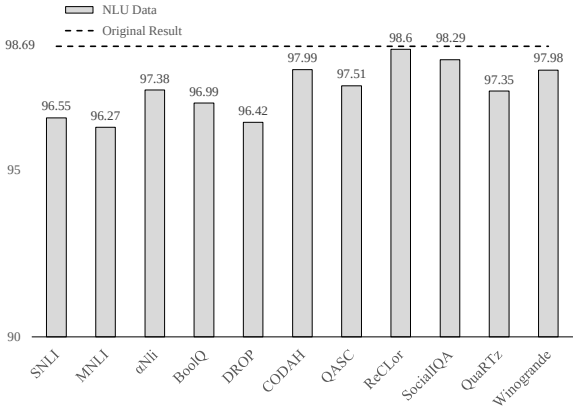

Figure 5: Results for Preliminary 3.

BERT + fully-connected classifier as the original architectures. For the first model, we fix BERT and only train the fully-connected classifier. For the second model, we firstly train BERT in the unsupervised manner, and then fix it and only train the fully-connected classifier. For the third model, we fine-tune BERT and train the classifier at the same time. The final results of the first two models are almost equivalent to random selection (Accuracy: 50.41% and 50.00%, respectively) but the result of the third model can reach 98.69%. This has validated that BERT cannot master logical reasoning after being trained in the unsupervised manner.

Preliminary 3: Validating that Transformers always has the ability to master logical reasoning no matter what initial weights are. We perform this validation based on the process shown in Fig 4. Firstly, we acquire M_3 following the quantitative process. Secondly, we test M_3 on the probe dataset. We compare the results R_{V3} with the original result in (Clark et al., 2020), which is shown in Fig 5 (dotted line represents the original result). The comparison illustrates that no matter what initial weights are, the performance will finally reach a similar level to the original result's level.

5 Results and Analysis

5.1 Overall Results of Diagnosis

Based on the diagnosis, we acquire a qualitative indicator I_1 and a quantitative indicator I_2 for each NLU to be diagnosed and the results are shown in Table 3. Based on the results, we firstly answer two questions related to the correlation on logical reasoning.

1. Whether does a dataset have sufficient explicit or implicit logical expressions that enables models to master logical reasoning or not? Based on results (I_1) of the qualitative diagnosis, almost all datasets (except for Winogrande) shows significance to logical reasoning, which means they have a certain amount of logical reasoning to enable the probe model to master logical reasoning. Winogrande, the only exception that cannot be judged by the probe model, will be further analyzed in the section of Exception Analysis. To compare NLI and MRC datasets, we adopt another threshold $\delta' = 0.6$ to distinguish "very significant" ("++") and "significant" ("+"). Generally, NLI datasets are more significant than MRC datasets. We speculate that reasons for the conclusion are: 1) Logical patterns in the NLI datasets are more likely to be captured as the contexts in these datasets are much simpler; 2) MRC datasets usually contain complicated contexts requiring a variety of reasoning skills, which makes it hard to expose the logical patterns.

2. How much does the task defined on a dataset depend on the logical reasoning skill? Although most NLU datasets can provide materials of logical reasoning, not all tasks defined on these datasets have strong dependences on logical reasoning. We have shown the dependence indicator I_2 in Fig.6. From the figure, we can find that NLI datasets, except for αNLI, have relatively high dependences on logical reasoning. This means that NLI tasks often highly rely on logical reasoning, which plays a dominant role in these tasks. These results are roughly consistent with the definition and the purpose of the NLI tasks (The outlier in NLI datasets, αNLI, will be further analyzed in the section of Exception Analysis). Among comprehensive MRC tasks (Type 1), we can find the dependence indicators for all three tasks are moderate, which may be due to their requirements of multiple NLU skills. In terms of MRC datasets for specific purposes (Type 2), the results show a remarkable difference among these tasks. On the

Diagnosis	Acc.(%)/Ind.	Datasets										
		NLI			MRC(Type1)			MRC(Type2)				
		SNLI	MNLI	αNLI	BoolQ	DROP	CODAH	QASC	ReCLor	SocialIQA	QuaRTz	Winogrande
Qualitative	R_1	60.35	61.32	57.12	58.14	58.38	59.13	58.39	58.21	60.35	57.30	50.77
Quantitative	R_2	70.40	64.16	50.29	58.14	51.11	56.80	55.40	51.60	55.80	50.91	50.00
	R_3	78.87	70.61	52.58	60.75	53.80	60.00	65.44	52.80	57.90	51.56	51.34
	Δ_R	8.13	6.45	2.29	2.61	2.69	3.20	10.04	1.20	2.10	0.65	1.34
Indicators	I_1	++	++	+	+	+	+	+	+	++	+	-
	I_2	10.31	9.13	4.36	4.30	5.00	5.33	14.35	2.27	3.63	1.26	2.61

Table 3: Diagnosed Results: I_1 and I_2 are two indicators for the qualitative process and the quantitative process, respectively. The threshold δ of qualitative indicator is 55%. To compare NLI and MRC datasets, we adopt another threshold $\delta' = 60\%$ to distinguish "very significant (++)" and "significant (+)". Therefore, "-" means "unable to judge", "+" means "significant" and "++" means "very significant".

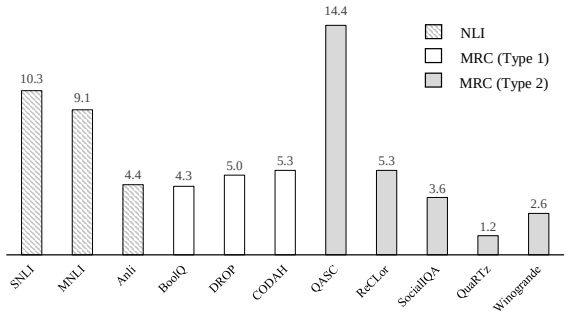

Figure 6: Quantitative indicators. Among all three kinds of datasets, most NLI datasets (except for αNLI) have high indicators. The indicators of comprehensive MRC datasets are relatively moderate. Specific MRC datasets contain QASC with the highest I_2 and QuaTRz with the lowest I_2, which are complicated.

one hand, tasks such as QASC ($I_1 = 14.35$) show a strong dependence on logical reasoning. On the other hand, tasks such as SocialIQA ($I_2 = 3.63$), QuaRTz ($I_2 = 1.26$) and Winogrande ($I_2 = 2.61$), show almost no dependence on logical reasoning. Based on the reports from original papers, the corresponding dominant skills for four MRC datasets (Type 2) are logical resaoning (Khot et al., 2020), knowledge-based skill (Sap et al., 2019b), numerical reasoning (Tafjord et al., 2019) and coreferential reasoning (Sakaguchi et al., 2020), respectively. This is consistent with I_2 indicating that only QASC has a strong dependence on logical reasoning among these four datasets. Surprisingly, although ReCLor, a Type 2 MRC dataset, is designed for the purpose of evaluating logical reasoning (Yu et al., 2020), I_2 cannot indicate that the dataset has a strong dependence on logical reasoning. Aiming at this exception, we will also perform individual analysis in the section of Exception Analysis.

5.2 Exception Analysis

In this section, we conduct a detailed analysis of the exceptions, αNLI, ReCLor and Winogrande, mentioned above to further understand the causes of the results. Meanwhile, on the basis of the analysis, we also acquire a better understanding of the limitation of the proposed quantitative indicator I_2.

1. Winogrande. From the qualitative indicator I_1, Winogrande is the only dataset unable to judge whether it contains sufficient logical expressions or not. We take a representative pair of examples in the dataset as the case to analyze. The case includes (Context: `The trophy does not fit into the suitcase`, Statement: `because trophy is too large`, Label: `True`) and (Context: `The trophy does not fit into the suitcase`, Statement: `because suitcase is too large`, Label: `False`). It is obvious that no logical expressions or underlying logical forms.

2. αNLI. αNLI seems an outlier which has a lower I_2 than other NLI tasks. Based on our analysis, we find that although αNLI is classified into the NLI category, it is designed to benchmark sequential reasoning in a narrative text rather than logical reasoning. An example of αNLI is (Context: `Jill had a pet cat _____. She was able to remove the fleas by sprinkling salt on her floors`, Statement: `The cat had fleas`, Label: `True`). It is obvious that the example requires the model to be able to understand the sequential relation between context and statement rather than apply logical reasoning. Therefore, the dependence

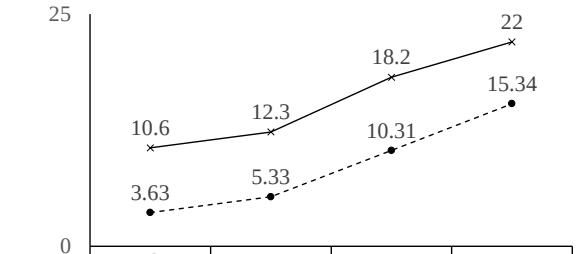

	SocialIQA	CODAH	SNLI	QASC
I2	3.63	5.33	10.31	15.34
Δ(%)	10.6	12.3	18.2	22

Figure 7: Line charts of quantitative indicator I_2 and paired difference Δ. The results show that I_2 is significantly correlated to Δ.

of αNLI on logical reasoning is weak.

3. ReCLor. As Yu et al. (2020) reported, ReCLor is constructed from GMAT and LAST to evaluate logical reasoning. However, in our diagnosis, the indicator of ReCLor is very low. Based on our analysis, the causes may include: 1) To solve the task defined on ReCLor requires a large amount of legal-related domain knowledge and commonsense, which is not sufficiently acquired by the probe model. Therefore, the model cannot capture the underlying logical forms; 2) Although logical reasoning is the dominant skill in ReCLor, it cannot be decoupled from multiple complex skills. Therefore, we can further understand the limitations of our proposed method from the analysis of ReCLor. Actually, I_2 reflects the lower bound of the dependence on logical reasoning, as it can just indicate the part satisfying the following conditions:

- logical reasoning is the dominant skill under the condition that knowledge acquired by the probe model is sufficient.
- logical features can be decoupled from the complex reasoning features.

Despite the limitations, I_2 is still a reasonable indicator to evaluate the dependence of NLU tasks on logical reasoning. Further illustration will be given in the next section.

5.3 Paired Analysis

Intuitively, models with logical reasoning are sensitive to subtle differences between logically true propositions and logically false propositions. Based on this intuition, we perform a paired analysis to provide intuitive evidence that M_3 does master logical reasoning compared with M_2. This is indirect evidence that our proposed indicator, I_2, is reasonable to indicate the dependence between the NLU dataset and logical reasoning. To perform the analysis, we first reconstruct paired sets comprising pairs of positive and negative samples on four datasets, SNLI (NLI), CODAH (Type 1 MRC), QASC (Type 2 MRC) and SocialIQA (Type 2 MRC). These four datasets have a common feature that for each positive example, we can extract a corresponding negative example directly from the datasets and vice versa. The only difference between the pair is that two samples have different keywords in their statements. This setting can ensure that if a sample includes a logically true or false proposition, the counterpart must include a logically false or true proposition. Next, we test M_2 and M_3 directly on paired sets and calculate their paired accuracies. Finally, we use the gap between M_3's accuracy and M_2's accuracy Δ to conduct the analysis. These results are shown in Table 4. From the table, Δ for each dataset is larger than 10%, which is evidence that the difference between M_3 and M_2 is decidedly due to logical reasoning.

Acc.(%)	SNLI	CODAH	QASC	SocialIQA
M_3	41.3	18.5	11.6	18.4
M_2	59.5	30.8	33.6	29.0
Δ	18.2	12.3	22.0	10.6

Table 4: Paired differences of SNLI, CODAH, QASC and SocialIQA.

Moreover, we further make a horizontal comparison between datasets and Fig 7 is the line charts of I_2 and Δ. According to the analysis above, Δ is a reasonable indicator to measure the dependence on logical reasoning. From the figure, a significant positive correlation between I_2 and Δ exists. This is intuitive evidence that I_2 is also a reasonable indicator to reflect the dependence of a dataset on logical reasoning skill.

5.4 Case Study

In this part, we perform a case study to show the logical structure detected by the quantitative indicator I_2. We select a case from QASC (shown in Fig.8) which shows the strongest dependence on logical reasoning. From the case, we list the logical structure of the case for reasoning $p_1 \wedge p_2 \rightarrow q_1$. This is a typical logical form of the conjunctive implication which can be diagnosed by I_2.

context: Beads of water are formed by water vapor condensing (p_1). Clouds are made of water vapor (p_2). Condensation is the change of water vapor to a liquid (p_3). An example of water vapor is steam (p_4). Condensation of water vapor occurs during the chilling season (p_5).
statement: Beads of water is formed by clouds (q_1).
Liquid of water is formed by clouds (q_2).
label: $True$ for q_1 and $False$ for q_2.
logical structure for reasoning: $p_1 \wedge p_2 \rightarrow q_1$

Figure 8: A logical case from QASC

6 Conclusions and Future Work

In this work, we present a novel framework, which can diagnose the correlation between the NLU dataset and a specific skill and we probe a fundamental reasoning skill, logical reasoning, on 11 NLU datasets. Our framework involves a logical probe to conduct diagnosis and defines a qualitative process and a quantitative process to calculate two indicators. From the results, We observe that 1) Most NLI datasets have a relatively strong correlation with logical reasoning. 2) The correlations between Type 1 MRC datasets and logical reasoning are moderate because logical reasoning is not the only dominant skill in these datasets. 3) The dependences of Type 2 MRC datasets are not always exactly consistent with their intended purpose. Based on the analysis, although there are several limitations in our proposed method, this work is still a reasonable attempt to deeply understand the relationship between the dataset and a specific NLU skill. In future works, we will focus on: 1) exploring the solution to the limitations of the proposed method; 2) build associations for different datasets that require the same NLU capabilities.

Acknowledgements

This work was supported by the Shanghai Municipal Science and Technology Major Project (2021SHZDZX0102), and the Shanghai Science and Technology Innovation Action Plan (20511102600).

References

Jens Allwood, Lars Andersson, and Östen Dahl. 1977. Logic in linguistics.

Razieh Baradaran, Razieh Ghiasi, and Hossein Amirkhani. 2020. A survey on machine reading comprehension systems. In *CoRR*.

Chitta Baral, Pratyay Banerjee, Kuntal Kumar Pal, and Arindam Mitra. 2020. Natural language QA approaches using reasoning with external knowledge. In *CoRR*.

Chandra Bhagavatula, Ronan Le Bras, Chaitanya Malaviya, Keisuke Sakaguchi, Ari Holtzman, Hannah Rashkin, Doug Downey, Wen-tau Yih, and Yejin Choi. 2020. Abductive commonsense reasoning. In *ICLR*.

Michael Boratko, Harshit Padigela, Divyendra Mikkilineni, Pritish Yuvraj, Rajarshi Das, Andrew McCallum, Maria Chang, Achille Fokoue-Nkoutche, Pavan Kapanipathi, Nicholas Mattei, Ryan Musa, Kartik Talamadupula, and Michael Witbrock. 2018. A systematic classification of knowledge, reasoning, and context within the ARC dataset. In *Proceedings of the Workshop on Machine Reading for Question Answering@ACL 2018*.

Samuel R. Bowman, Gabor Angeli, Christopher Potts, and Christopher D. Manning. 2015. A large annotated corpus for learning natural language inference. In *EMNLP*.

Michael Chen, Mike D'Arcy, Alisa Liu, Jared Fernandez, and Doug Downey. 2019. CODAH: an adversarially authored question-answer dataset for common sense. In *CoRR*.

Christopher Clark, Kenton Lee, Ming-Wei Chang, Tom Kwiatkowski, Michael Collins, and Kristina Toutanova. 2019. Boolq: Exploring the surprising difficulty of natural yes/no questions. In *NAACL-HLT*.

Peter Clark, Oyvind Tafjord, and Kyle Richardson. 2020. Transformers as soft reasoners over language. In *IJCAI*.

Alexis Conneau, Germán Kruszewski, Guillaume Lample, Loïc Barrault, and Marco Baroni. 2018. What you can cram into a single \$&!#* vector: Probing sentence embeddings for linguistic properties. In *ACL*.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: pre-training of deep bidirectional transformers for language understanding. In *NAACL-HLT*.

Dheeru Dua, Yizhong Wang, Pradeep Dasigi, Gabriel Stanovsky, Sameer Singh, and Matt Gardner. 2019. DROP: A reading comprehension benchmark requiring discrete reasoning over paragraphs. In *NAACL-HLT*.

Tianqing Fang, Hongming Zhang, Weiqi Wang, Yangqiu Song, and Bin He. 2021. DISCOS: bridging the gap between discourse knowledge and commonsense knowledge. In *WWW*.

Yanlin Feng, Xinyue Chen, Bill Yuchen Lin, Peifeng Wang, Jun Yan, and Xiang Ren. 2020. Scalable multi-hop relational reasoning for knowledge-aware question answering. In *EMNLP*.

Deepanway Ghosal, Navonil Majumder, Rada Mihalcea, and Soujanya Poria. 2021. Stack: Sentence ordering with temporal commonsense knowledge. In *EMNLP*.

Christopher Hahn, Frederik Schmitt, Jens U Kreber, Markus N Rabe, and Bernd Finkbeiner. 2020. Transformers generalize to the semantics of logics. In *CoRR*.

John Hewitt and Percy Liang. 2019. Designing and interpreting probes with control tasks. In *EMNLP-IJCNLP*.

Tushar Khot, Peter Clark, Michal Guerquin, Peter Jansen, and Ashish Sabharwal. 2020. QASC: A dataset for question answering via sentence composition. In *AAAI*.

Walter Kintsch. 1991. The role of knowledge in discourse comprehension: A construction-integration model. In *Text and Text Processing*, Advances in Psychology.

Mark A Musen and Johan Van Der Lei. 1988. Of brittleness and bottlenecks: Challenges in the creation of pattern-recognition and expert-system models. In *Machine Intelligence and Pattern Recognition*.

Nikita Nangia, Adina Williams, Angeliki Lazaridou, and Samuel R. Bowman. 2017. The repeval 2017 shared task: Multi-genre natural language inference with sentence representations. In *EMNLP*.

Fabio Petroni, Tim Rocktaschel, Sebastian Riedel, Patrick S. H. Lewis, Anton Bakhtin, Yuxiang Wu, and Alexander H. Miller. 2019. Language models as knowledge bases? In *EMNLP-IJCNLP*.

Siva Reddy, Danqi Chen, and Christopher D. Manning. 2019. Coqa: A conversational question answering challenge. In *TACL*.

Marco Túlio Ribeiro, Tongshuang Wu, Carlos Guestrin, and Sameer Singh. 2020. Beyond accuracy: Behavioral testing of NLP models with checklist. In *ACL*.

Kyle Richardson, Hai Hu, Lawrence S. Moss, and Ashish Sabharwal. 2020. Probing natural language inference models through semantic fragments. In *AAAI*.

Anna Rogers, Olga Kovaleva, and Anna Rumshisky. 2020. A primer in bertology: What we know about how BERT works. *TACL*.

Keisuke Sakaguchi, Ronan Le Bras, Chandra Bhagavatula, and Yejin Choi. 2020. Winogrande: An adversarial winograd schema challenge at scale. In *AAAI*.

Maarten Sap, Ronan Le Bras, Emily Allaway, Chandra Bhagavatula, Nicholas Lourie, Hannah Rashkin, Brendan Roof, Noah A. Smith, and Yejin Choi. 2019a. Atomic:an atlas of machine commonsense for if-then reasoning. In *AAAI*.

Maarten Sap, Hannah Rashkin, Derek Chen, Ronan LeBras, and Yejin Choi. 2019b. Socialiqa: Commonsense reasoning about social interactions. In *CoRR*.

Saku Sugawara, Yusuke Kido, Hikaru Yokono, and Akiko Aizawa. 2017. Evaluation metrics for machine reading comprehension: Prerequisite skills and readability. In *ACL*.

Saku Sugawara, Pontus Stenetorp, Kentaro Inui, and Akiko Aizawa. 2020. Assessing the benchmarking capacity of machine reading comprehension datasets. In *AAAI*.

Oyvind Tafjord, Matt Gardner, Kevin Lin, and Peter Clark. 2019. Quartz: An open-domain dataset of qualitative relationship questions. In *EMNLP-IJCNLP*.

Ian Tenney, Patrick Xia, Berlin Chen, Alex Wang, Adam Poliak, R. Thomas McCoy, Najoung Kim, Benjamin Van Durme, Samuel R. Bowman, Dipanjan Das, and Ellie Pavlick. 2019. What do you learn from context? probing for sentence structure in contextualized word representations. In *ICLR*.

Jidong Tian, Yitian Li, Wenqing Chen, Liqiang Xiao, Hao He, and Yaohui Jin. 2021. Diagnosing the first-order logical reasoning ability through logicnli. In *EMNLP*.

Cong Wang, Amr Rizk, and Michael Zink. 2016. SQUAD: a spectrum-based quality adaptation for dynamic adaptive streaming over HTTP. In *EMNLP*.

Zhilin Yang, Peng Qi, Saizheng Zhang, Yoshua Bengio, William W. Cohen, Ruslan Salakhutdinov, and Christopher D. Manning. 2018. Hotpotqa: A dataset for diverse, explainable multi-hop question answering. In *EMNLP*.

Mark Yatskar. 2019. A qualitative comparison of coqa, squad 2.0 and quac. In *NAACL*.

Weihao Yu, Zihang Jiang, Yanfei Dong, and Jiashi Feng. 2020. Reclor: A reading comprehension dataset requiring logical reasoning. In *ICLR*.

Chengchang Zeng, Shaobo Li, Qin Li, Jie Hu, and Jianjun Hu. 2020. A survey on machine reading comprehension: Tasks, evaluation metrics, and benchmark datasets. In *CoRR*.

ArcaneQA: Dynamic Program Induction and Contextualized Encoding for Knowledge Base Question Answering

Yu Gu
The Ohio State University
Columbus, Ohio, USA
gu.826@osu.edu

Yu Su
The Ohio State University
Columbus, Ohio, USA
su.809@osu.edu

Abstract

Question answering on knowledge bases (KBQA) poses a unique challenge for semantic parsing research due to two intertwined challenges: *large search space* and *ambiguities in schema linking*. Conventional ranking-based KBQA models, which rely on a candidate enumeration step to reduce the search space, struggle with flexibility in predicting complicated queries and have impractical running time. In this paper, we present ArcaneQA, a novel generation-based model that addresses both the large search space and the schema linking challenges in a unified framework with two mutually boosting ingredients: *dynamic program induction* for tackling the large search space and *dynamic contextualized encoding* for schema linking. Experimental results on multiple popular KBQA datasets demonstrate the highly competitive performance of ArcaneQA in both effectiveness and efficiency.[1]

1 Introduction

Modern knowledge bases (KBs) contain a wealth of structured knowledge. For example, FREEBASE (Bollacker et al., 2008) contains over 45 million entities and 3 billion facts across more than 100 domains, while GOOGLE KNOWLEDGE GRAPH has amassed over 500 billion facts about 5 billion entities (Sullivan, 2020). Question answering on knowledge bases (KBQA) has emerged as a user-friendly solution to access the massive structured knowledge in KBs.

KBQA is commonly modeled as a semantic parsing problem (Zelle and Mooney, 1996; Zettlemoyer and Collins, 2005) with the goal of mapping a natural language question into a logical form that can be executed against the KB (Berant et al., 2013; Cai and Yates, 2013; Yih et al., 2015). Compared with other semantic parsing settings such as text-to-SQL parsing (Zhong et al., 2017; Yu et al., 2018), where the underlying data is moderate-sized, the massive scale and the broad-coverage schema of KBs makes KBQA a uniquely challenging setting for semantic parsing research.

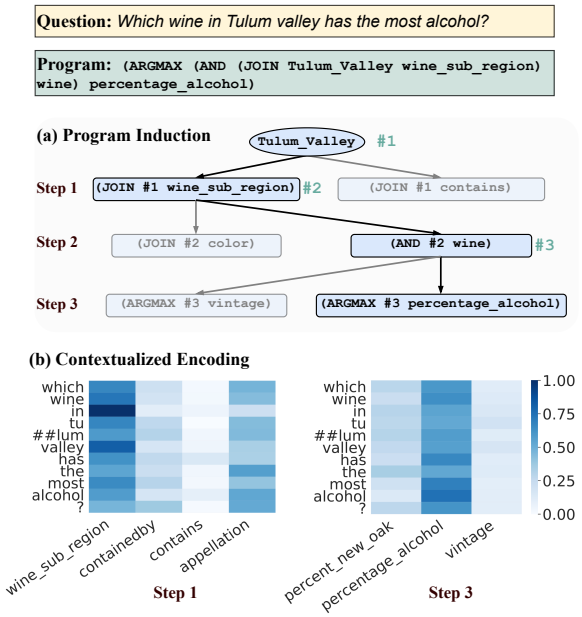

Figure 1: KBQA is commonly modeled as semantic parsing with the goal of mapping a question into an executable program. **(a)** A high-level illustration of our program induction procedure. The target program is induced by incrementally synthesizing a sequence of subprograms (`#1-3`). The execution of each subprogram can significantly reduce the search space of subsequent subprograms. **(b)** Alignments between question words and schema items at different steps achieved by a BERT encoder. A pre-trained language model like BERT can jointly encode the question and schema items to get the contextualized representation at each step, which further guides the search process.

The unique difficulty stems from two intertwined challenges: *large search space* and *ambiguities in schema linking*. On the one hand, transductive semantic parsing models that are highly effective in other semantic parsing settings (Dong and Lapata,

[1] Data and code: dki-lab/ArcaneQA

Proceedings of the 29th International Conference on Computational Linguistics, pages 1718–1731
October 12–17, 2022.

2016; Wang et al., 2020) struggle with the large vocabulary size and often generate logical forms (i.e., formal queries)[2] that are not faithful to the underlying KB (Gu et al., 2021; Xie et al., 2022). Therefore, a candidate enumeration and ranking approach is commonly adopted for KBQA (Berant and Liang, 2014; Yih et al., 2015; Abujabal et al., 2017; Lan et al., 2019a; Sun et al., 2020; Gu et al., 2021; Ye et al., 2021). However, these methods have to make various compromises on the complexity of admissible logical forms to deal with the large search space. Not only does this limit the type of answerable questions, but it also leads to impractical runtime performance due to the time-consuming candidate enumeration (Gu et al., 2021). On the other hand, schema linking,[3] i.e., mapping natural language to the corresponding schema items in the KB (e.g., in Figure 1, `wine_sub_region` is a linked schema items), is also a core challenge of KBQA. Compared with text-to-SQL parsing (Hwang et al., 2019; Zhang et al., 2019; Wang et al., 2020), the broad schema of KBs and the resulting ambiguity between schema items makes accurate schema linking more important and challenging for KBQA. Recent studies show that contextualized joint encoding of natural language questions and schema items with BERT (Devlin et al., 2019) can significantly boost the schema linking accuracy (Gu et al., 2021; Chen et al., 2021). However, existing methods still struggle with the large search space and need to encode a large number of schema items, which is detrimental to both accuracy and efficiency.

We present ArcaneQA (Dynamic Program Induction and Contextualized Encoding for Question Answering), a *generation-based* KBQA model that addresses both the large search space and the schema linking challenges in a unified framework. Compared with the predominant ranking-based KBQA models, our generation-based model can prune the search space on the fly and thus is more flexible to generate diverse queries without compromising the expressivity or complexity of answerable questions. Inspired by prior work (Dong and Lapata, 2016; Liang et al., 2017; Semantic Machines et al., 2020; Chen et al., 2021), we model KBQA using the encoder-decoder framework. However, instead of top-down decoding with grammar-level constraints as in prior work, which does not guarantee the faithfulness of the generated queries to the underlying KB, ArcaneQA performs *dynamic program induction* (Liang et al., 2017; Semantic Machines et al., 2020), where we incrementally synthesize a program by dynamically predicting a sequence of subprograms to answer a question; i.e., *bottom-up parsing* (Cheng et al., 2019; Rubin and Berant, 2021). Each subprogram is grounded to the KB and its grounding (i.e., denotation or execution results) can further guide an efficient search for faithful programs (see Figure 1(a)). In addition, we unify the meaning representation (MR) for programs in KBQA using S-expressions and support more diverse operations over the KB (e.g., numerical operations such as `COUNT/ARGMIN/ARGMAX` and diverse graph traversal operations).

At the same time, we employ pre-trained language models (PLMs) like BERT to jointly encode the question and schema items and get the contextualized representation of both, which implicitly links words to the corresponding schema items via self-attention. One unique feature to note is that *the encoding is also dynamic*: at each prediction step, only the set of admissible schema items determined by the dynamic program induction process needs to be encoded, which allows extracting the most relevant information from the question for each prediction step while avoiding the need to encode a large number of schema items. Figure 1(b) illustrates the contextualization of different steps via the attention heatmaps of BERT. In this example, the attention of each question word over candidate schema items serves as a strong indicator of the gold items for both steps (i.e., `wine_sub_region` for step 1 and `percentage_alcohol` for step 3). The two key ingredients of our model are *mutually boosting*: dynamic program induction significantly reduces the number of schema items that need to be encoded, while dynamic contextualized encoding intelligently guides the search process.

Our main contribution is as follows: a) We propose a novel generation-based KBQA model that is flexible to generate diverse complex queries while also being more efficient than ranking-based models. b) We propose a novel strategy to effectively employ PLMs to provide contextualized encoding for KBQA. c) We unify the meaning representation (MR) of different KBQA datasets and support

[2]We use the terms logical form, query, and program interchangeably across the paper.

[3]Semantic parsing implicitly entails two sub-tasks: *schema linking* and *composition*. There is not necessarily a dedicated step or component for schema linking. More commonly, the two sub-tasks are handled simultaneously.

more diverse operations. d) With our unified MR, we evaluate our model on three popular KBQA datasets and show highly competitive results.

2 Related Work

Ranking-Based KBQA. To handle the large search space in KBQA, existing studies typically rely on hand-crafted templates with a pre-specified maximum number of relations to enumerate candidate logical forms (Yih et al., 2015; Abujabal et al., 2017; Lan et al., 2019a; Bhutani et al., 2019, 2020), which suffers from limited expressivity and scalability. For example, Yih et al. (2015) limit the candidate programs to be a core relational chain, whose length is at most two, plus constraints. Ye et al. (2021) additionally adopts a post-generation module to revise the enumerated logical forms into more complicated ones, however, their method still heavily depends on the candidate enumeration step. In addition, the time-consuming candidate enumeration results in impractical online inference time for ranking-based models. In contrast, ArcaneQA obviates the need for candidate enumeration by pruning the search space on the fly and thus can generate more diverse and complicated programs within practical running time.

Generation-Based KBQA. To relax the restriction on candidate enumeration, some recent efforts are made to reduce the search space using beam search (Lan et al., 2019b; Chen et al., 2019; Lan and Jiang, 2020), however, Lan et al. (2019b) and Chen et al. (2019) can only generate programs of path structure, while Lan and Jiang (2020) follow the query graph structure proposed by Yih et al. (2015). A few recent studies (Liang et al., 2017; Chen et al., 2021) formulate semantic parsing over the KB as sequence transduction using encoder-decoder models to enable more flexible generation. Chen et al. (2021) apply schema-level constraints to eliminate ill-formed programs from the search space, however, they do not guarantee the faithfulness of predicted programs. Similar to Liang et al. (2017), our dynamic program induction uses KB contents-level constraints to ensure the faithfulness of generated programs, but we extend it to handle more complex and diverse questions and also use it jointly with dynamic contextualized encoding.

Using PLMs in Semantic Parsing. PLMs have been widely applied in many semantic parsing tasks, typically being used to jointly encode the input question and schema items(Hwang et al., 2019; Zhang et al., 2019; Wang et al., 2020; Scholak et al., 2021). However, PLMs have been under-exploited in KBQA. One major difficulty of using PLMs in KBQA lies in the high volume of schema items in a KB; simply concatenating all schema items with the input question for joint encoding, as commonly done in text-to-SQL parsing, will vastly exceed PLMs' maximum input length. Existing KBQA models either use PLMs to provide features for downstream classifiers(Lan and Jiang, 2020; Sun et al., 2020) or adopts a pipeline design to identify a smaller set of schema items beforehand and only use PLMs to encode these identified items (Gu et al., 2021; Chen et al., 2021), which can lead to error propagation. By comparison, ArcaneQA can fully exploit PLMs to provide contextualized representation for the question and schema items dynamically, where only the most relevant schema items are encoded at each step. More recently, Ye et al. (2021) use T5 (Raffel et al., 2019) to output a new program given a program as input, while T5's decoder generates free-formed text and does not always produce faithful programs. By contrast, ArcaneQA only uses PLMs for encoding and uses its customized decoder with a faithfulness guarantee.

3 Background

Knowledge Base. A knowledge base $\mathcal{K}$ consists of a set of relational triplets $\mathcal{K}_r \subset \mathcal{E} \times \mathcal{R} \times (\mathcal{E} \cup \mathcal{L})$ and a set of class assertions $\mathcal{K}_c \subset \mathcal{E} \times \mathcal{C}$, where $\mathcal{C}$ is a set of classes, $\mathcal{E}$ is a set of entities, $\mathcal{L}$ is a set of literals and $\mathcal{R}$ is a set of binary relations. Elements in $\mathcal{C}$ and $\mathcal{R}$ are also called the schema items of $\mathcal{K}$.

Meaning Representation for KBQA. Prior work adopt different meaning representations to represent logical forms for KBQA. For example, Yih et al. (2015) use graph query, which represents a program as a core relation chain with (optionally) some entity constraints. Cai and Yates (2013) use λ-Calculus as their meaning representation. In this paper, we follow Gu et al. (2021) to use S-expressions as our meaning representation due to their expressivity and simplicity. To support more diverse operations over the KB, we extend their definitions with two additional functions `CONS` and `TC`, which are used to support constraints with implicit entities and temporal constraints respectively (see details in Appendix A). For implicit entities, consider the question *"What was Elie Wiesel's father's name?"*, whose target query involves two entities: `Elie_Wiesel` and

Male. The entity Male is an implicit constraint rather than a named entity,[4] and it is used as an argument of CONS in the target logical form: (CONS (JOIN people.person.children Elie_Wiesel) people.person.gender Male). TC works in a similar way, with the difference being that the constraint should be a temporal expression (e.g., 2015-08-10) rather than an implicit entity.

4 Approach

4.1 Overview

The core idea of our generation-based model is to gradually expand a subprogram (i.e., a partial query) into the finalized target program, instead of enumerating all possible finalized programs from the KB directly, which suffers from combinatorial explosion. There are two common strategies to instantiate the idea of gradual subprogram expansion, depending on the type of meaning representation being used. For a graph-like meaning representation, we can directly perform graph search over the KB to expand a subprogram (Chen et al., 2019; Lan and Jiang, 2020). Also, we can linearize a program into a sequence of tokens and perform decoding in the token space (Liang et al., 2017; Scholak et al., 2021). Because S-expressions can be easily converted into sequences of tokens, we choose to follow the second strategy and take advantage of the encoder-decoder framework, which has been a de facto choice for many semantic parsing tasks. Concretely, ArcaneQA learns to synthesize the target program by dynamically generating a sequence of subprograms token by token until predicting $\langle EOS \rangle$, where each subsequent subprogram is an expansion from one or more preceding subprograms (denoted as parameters in the subsequent subprogram). Formally, the goal is to map an input question $q = x_1, ..., x_{|q|}$ to a sequence of subprograms $o = o_1^1, ..., o_{|o^1|}^1, ..., o_1^k, ..., o_{|o^k|}^k = y_1, ..., y_{|o|}$, where k is the number of total subprograms and $|o| = \sum_{i=1}^{k} |o^i|$. We base ArcaneQA on Seq2Seq with attention (Sutskever et al., 2014; Bahdanau et al., 2015), in which the conditional probability $p(o|q)$ is decomposed as:

$$p(o|q) = \prod_{t=1}^{|o|} p(y_t|y_{<t}, q), \tag{1}$$

where each token y_t is either a token from the vocabulary $\mathcal{V}$ or an intermediate subprogram from the set $\mathcal{S}$ storing all previously generated subprograms. $\mathcal{V}$ comprises all schema items in $\mathcal{K}$, syntactic symbols in S-expressions (i.e., parentheses and function names), and the special token $\langle EOS \rangle$. $\mathcal{S}$ initially contains the identified entities in the question (e.g., #1 in Figure 1). Every time a subprogram is predicted, it is executed and added to $\mathcal{S}$ (e.g., #2 in Figure 1).

ArcaneQA builds on two key ideas: *dynamic program induction* and *dynamic contextualized encoding* (see Figure 2). At each decoding step, ArcaneQA only makes a prediction from a small set of admissible tokens instead of the entire vocabulary. This is achieved by the dynamic program induction framework (subsection 4.2), which effectively prunes the search space by orders of magnitude and guarantees that the predicted programs are faithful to the KB. In addition, we dynamically apply BERT to provide contextualized joint encoding for both the question and admissible tokens at each decoding step (subsection 4.3). In this way, we allow the contextualized encoding to only focus on the most relevant information without introducing noise from irrelevant tokens.

4.2 Dynamic Program Induction

Dynamic program induction capitalizes on the idea that a complicated program can be gradually expanded from a list of subprograms. To ensure the expanded program is faithful to the KB, we query the KB with a subprogram to expand and a function defined in Table 4 to get a set of admissible actions (tokens). For example, in Figure 2, given #1 and the function JOIN, the admissible actions are defined by predicting a relation connecting to the execution result of #1 (i.e., Tulum_Valley), and there are only four relations to choose from (e.g., appellation and wine_sub_region). Table 1 shows a comprehensive description of expansion rules for different functions. With these rules, ArcaneQA can greatly reduce the search space for semantic parsing over the KB dynamically. The reduced candidate space further allows us to perform dynamic contextualized encoding (subsection 4.3).

[4] WEBQSP is the only dataset we consider that has this feature. Though there might be a more systematic way to differentiate implicit entities from named entities, we choose an expedient way to collect implicit entities from the training data according to whether an entity is explicitly mentioned in the question.

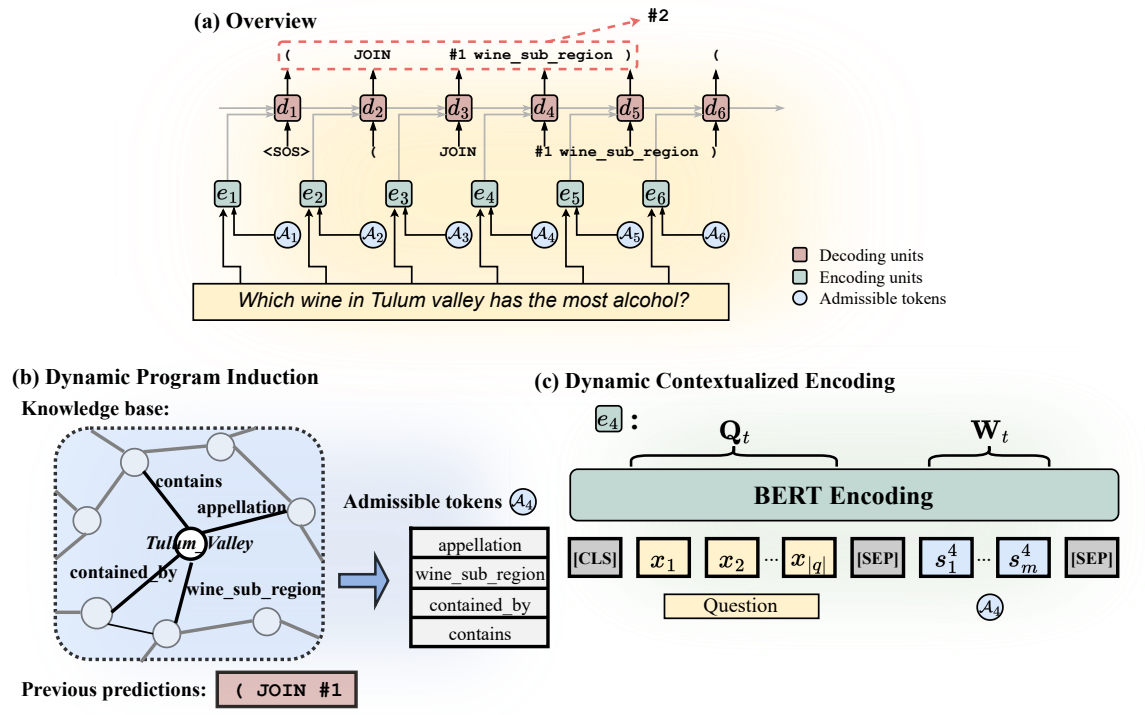

Figure 2: **(a)** Overview of ArcaneQA. ArcaneQA synthesizes the target program by iteratively predicting a sequence of subprograms. **(b)** At each step, it makes a prediction from a small set of admissible tokens $\mathcal{A}$ dynamically determined based on the execution of previous subprograms (for faithfulness to the KB) as well as the grammar (for well-formedness). **(c)** ArcaneQA also leverages BERT to provide dynamic contextualized encoding of the question and the admissible tokens at each step, which enables implicit schema linking and guides the dynamic program induction process.

Within our encoder-decoder framework, this idea is implemented using constrained decoding (Liang et al., 2017; Scholak et al., 2021), i.e., at each decoding step, a small set of admissible tokens from the vocabulary is determined based on the decoding history following predefined rules. The expansion rules in Table 1 have already comprised part of our rules for constrained decoding. In addition, several straightforward grammar rules are applied to ensure that the generated programs are well-formed. For instance, after predicting "(", the admissible tokens for the next step can only be a function name. After predicting a function name, the decoder can only choose a preceding subprogram to expand. After predicting ")", the admissible tokens for next step can only be either "(", indicating the start of a new subprogram, or "$\langle EOS \rangle$", denoting termination. The decoding process can be viewed as a *sequential decision-making process*, which decomposes the task of finding a program from the enormous search space into making decisions from a sequence of smaller search spaces.

4.3 Dynamic Contextualized Encoding

In semantic parsing, PLMs have typically been used to jointly encode the input question and all schema items via concatenation (Hwang et al., 2019; Zhang et al., 2019; Wang et al., 2020). However, direct concatenation is not feasible for KBQA due to a large number of schema items. Instead of obtaining a static representation for the question and items from $\mathcal{V}$ before decoding (Gu et al., 2021; Chen et al., 2021), we propose to do dynamic contextualized encoding at each decoding step; for each step, we use BERT to jointly encode the question and only the admissible tokens from $\mathcal{V}$. ArcaneQA's dynamic program induction vastly reduces the number of candidate tokens at each step and allows us to concatenate the question and the admissible tokens into a compact sequence:[5]

$$\text{[CLS]}, x_1, ..., x_{|q|}, \text{[SEP]}, s_1^t, ..., s_m^t, \text{[SEP]}$$

where $\{s_i^t\} \subset \mathcal{V}$ are admissible tokens at

[5]We omit the wordpieces tokenization here for brevity.

Current function	Admissible actions
JOIN	$\{r \vert h \in \#, (h, r, t) \in \mathcal{K}_r\}$
AND	$\{v \vert v \in \mathcal{S}, v \cap \# \neq \emptyset\} \cup \{c \vert e \in \#, (e, c) \in \mathcal{K}_c\}$
ARGMAX/ARGMIN	$\{r \vert h \in \#, t \in \mathcal{L}, (h, r, t) \in \mathcal{K}_r\}$
LT(LE/GT/GE)	$\{r \vert t < (\leq / > / \geq)\#, (h, r, t) \in \mathcal{K}_r\}$
COUNT	$\{)\}$
CONS	$\{(r, t) \vert h \in \#, (h, r, t) \in \mathcal{K}_r\}$
TC	$\{(r, t) \vert h \in \#, (h, r, t) \in \mathcal{K}_r, t \in \mathcal{L}$ is a temporal expression$\}$

Table 1: A set of rules to expand a preceding subprogram given a function. The execution of the subprogram is denoted as #. These expansion rules reduce the search space significantly with a faithfulness guarantee. COUNT takes no other argument, so the only admissible token is ")".

step t and $|\{s_i^t\}| = m$. After feeding the concatenated sequence to BERT, we obtain the question representation $\mathbf{Q_t} = (\mathbf{x_1}, ..., \mathbf{x_q})$ by further feeding the outputs from BERT to an LSTM encoder. For each admissible token, we represent it by averaging BERT outputs corresponding to its wordpieces. In this way, we also obtain the embedding matrix $\mathbf{W_t} \in \mathbb{R}^{m \times d}$, where each row corresponds to the embedding of an admissible token. The contextualized representation $\mathbf{Q_t}$ and $\mathbf{W_t}$ are both dynamically computed at each time step. Words and corresponding schema items are implicitly linked to each other via BERT's self-attention.

4.4 Decoding

We use an LSTM decoder. At decoding step t, given the hidden state $\mathbf{h_{t-1}}$ and input $\mathbf{c_{t-1}}$, we obtain the updated hidden state $\mathbf{h_t}$ by:

$$\mathbf{h_t} = \text{LSTM}_\theta(\mathbf{h_{t-1}}, \mathbf{c_{t-1}}) \quad (2)$$

where our LSTM decoder is parameterized by θ.

With $\mathbf{h_t}$ and $\mathbf{W_t}$—the embedding matrix of admissible tokens (determined by dynamic program induction)—we obtain the probability of generating a token from the admissible tokens:

$$p(y_t = s_{ti} | q, y_{<t}) = [\text{Softmax}(\mathbf{W_t h_t})]_i \quad (3)$$

The input $\mathbf{c_t}$ for the next step is obtained via the concatenation of the contextualized embedding of the current output token and the weighted representation of the question based on attention:

$$\mathbf{a_t} = \text{softmax}(\mathbf{Q_t h_t}) \quad (4)$$

$$\mathbf{q_t} = (\mathbf{a_t})^T \mathbf{Q_t} \quad (5)$$

$$\mathbf{c_t} = [[\mathbf{W_t}]_j; \mathbf{q_t}] \quad (6)$$

where ; denotes concatenation, and j denotes the index of the predicted y_t in $\mathbf{W_t}$.

4.5 Training and Inference

We train ArcaneQA with question-program pairs using cross entropy loss. The model learns to maximize the probability of predicting the gold token out of a small set of admissible tokens at each step, which is different from training a conventional Seq2Seq model using a static vocabulary.

During inference, ArcaneQA assumes an entity linker to identify a set of entities from the question at the beginning of program induction. However, the entity linker may identify false entities. To deal with it, ArcaneQA initiate its decoding process with different hypotheses from the set of entities. Basically, it tries out all possible combinations of the identified entities (i.e., the power set of the identified entities), considering that our entity linker normally can only identify no more than two entities from a question.

5 Experimental Setup

Datasets. We evaluate ArcaneQA on three KBQA datasets covering diverse KB queries.

GRAILQA (Gu et al., 2021) is a large-scale KBQA dataset that contains complex questions with various functions, including comparatives, superlatives, and counting. It evaluates the generalizability of KBQA at three levels: i.i.d., compositional and zero-shot.

GRAPHQ (Su et al., 2016) also contains questions of diverse nature. It is particularly challenging because it exclusively focuses on non-i.i.d. generalization.[6]

WEBQSP(Yih et al., 2016) is a clean subset of WEBQ (Berant et al., 2013) with annotated logical forms.

[6] GRAPHQ originally uses FREEBASE (version 2013-07) as their KB, while GRAILQA and WEBQ use FREEBASE (version 2015-08-09). In Gu et al. (2021), programs in GRAPHQ are converted into the corresponding FREEBASE 2015-08-09 version, and we will use this version in our experiments.

All questions in it are collected from Google query logs, featuring more realistic and complicated information needs such as questions with temporal constraints.

The total number of questions in GRAILQA, GRAPHQ, and WEBQ is 64,331, 5,166, and 4,737 respectively.

Evaluation Metrics. For GRAILQA, we use their official evaluation script with two metrics, EM, i.e., program exact match accuracy, and F1, which is computed based on the predicted and the gold answer set. For GRAPHQ and WEBQSP, we follow the standard practice and report F1.

Models for Comparison. We compare ArcaneQA with the previous best-performing models on three different datasets. For GRAILQA and WEBQSP, the state-of-the-art model is **RnG-KBQA** (Ye et al., 2021). Though RnG-KBQA uses T5 to decode the target program as unconstrained sequence transduction, it still heavily depends on candidate enumeration as a prerequisite. Therefore, it is not a generation-based model like ours. **ReTraCk** (Chen et al., 2021) is the state-of-the-art generation-based model on GRAILQA which poses grammar-level constraints to the decoder to generate well-formed but unnecessarily faithful programs. For GRAPHQ, the ranking-based model **SPARQA** (Sun et al., 2020) has achieved the best results so far. It uses BERT as a feature extractor for downstream classifiers. In addition to the state-of-the-art models, we also compare ArcaneQA with **BERT+Transduction** and **BERT+Ranking** (Gu et al., 2021), which are two baseline models on GRAILQA that enhance a vanilla Seq2Seq model with BERT to perform generation and ranking respectively.

Implementation. Our models are implemented using PyTorch and AllenNLP (Gardner et al., 2018). For BERT, we use the bert-base-uncased version provided by HuggingFace. For more details about implementation and hyper-parameters, we refer the reader to Appendix B.

6 Results

6.1 Overall Evaluation

We show the overall results in Table 2 (for dev set results see Appendix C). ArcaneQA achieves the state-of-the-art performance on both GRAPHQ and WEBQSP. For GRAPHQ, there are 188 questions in GRAPHQ's test set that cannot be converted into FREEBASE 2015-08-09 version, so we treat the F1 of all those questions as 0 following Gu et al. (2021), while the numbers in the parentheses are the actual F1 on the test set if we exclude those questions. ArcaneQA significantly outperforms the prior art by over 10%. The improvement over SPARQA shows the advantage of using PLMs for contextualized joint encoding instead of just providing features for ranking. On both WEBQSP and GRAILQA, ArcaneQA also achieves the best performance or performs on par with the prior art in terms of F1. It outperforms ReTraCk by 4.3% and 1.9% (using the same entity linking results) on WEBQSP and GRAILQA respectively, suggesting that ArcaneQA can more effectively reduce the search space with dynamic program induction compared with ReTraCk's grammar-based decoding. Also, our model performs on par with the previous state-of-the-art RnG-KBQA (i.e., 0.2% higher on WEBQSP, while 0.7% lower on GRAILQA). However, ArcaneQA under-performs RnG-KBQA in EM on GRAILQA. The overall EM of ArcaneQA is lower than RnG-KBQA by 5%, and the gap on zero-shot generalization is even larger (i.e., around 10%), despite the comparable numbers in F1. This can be explained by that ArcaneQA learns to predict a program in a more flexible way and may potentially find some novel structures. This may further be supported by the observation that ArcaneQA performs better than RnG-KBQA on compositional generalization, which requires KBQA models to generalize to unseen query structures during training. Overall, the results demonstrate ArcaneQA's flexibility in handling KBQA scenarios of different natures.

6.2 In-Depth Analyses

To gain more insights into ArcaneQA's strong performance, we conduct in-depth analyses on the two key designs of ArcaneQA.

Dynamic Program Induction. One vanilla implementation of ArcaneQA without dynamic program induction is BERT+Transduction, i.e., its search space and vocabulary during decoding is independent of previous predictions. As shown in Table 2a, when using the same entity linking results, ArcaneQA outperforms BERT+Transduction by 30.4% in overall F1 and is twice as good on zero-shot generalization. One major weakness of BERT+Transduction is that it predicts many pro-

Model	Overall		I.I.D.		Compositional		Zero-shot	
	EM	F1	EM	F1	EM	F1	EM	F1
QGG* (Lan and Jiang, 2020)	–	36.7	–	40.5	–	33.0	–	36.6
BERT+Transduction* (Gu et al., 2021)	33.3	36.8	51.8	53.9	31.0	36.0	25.7	29.3
BERT+Ranking* (Gu et al., 2021)	50.6	58.0	59.9	67.0	45.5	53.9	48.6	55.7
ReTraCk (Chen et al., 2021)	58.1	65.3	84.4	87.5	61.5	70.9	44.6	52.5
RnG-KBQA* (Ye et al., 2021)	61.4	67.4	78.0	81.8	55.0	63.2	56.7	63.0
ArcaneQA*	58.8	67.2	77.8	81.6	58.0	66.1	50.4	61.8
RnG-KBQA (Ye et al., 2021)	**68.8**	**74.4**	**86.2**	**89.0**	63.8	71.2	**63.0**	**69.2**
ArcaneQA	63.8	73.7	85.6	88.9	**65.8**	**75.3**	52.9	66.0
w/o contextualized encoding	49.7	59.1	77.6	82.1	50.5	59.4	36.5	48.5

(a) GRAILQA

Model	F1
UDEPLAMBDA (Reddy et al., 2017)	17.7
PARA4QA (Dong et al., 2017)	20.4
SPARQA (Sun et al., 2020)	21.5
BERT+Ranking (Gu et al., 2021)	25.0 (27.0)
ArcaneQA	**31.8** (**34.3**)
w/o contextualized encoding	20.7 (22.4)

(b) GRAPHQ

Model	F1
NSM (Liang et al., 2017)	69.0
KBQA-GST (Lan et al., 2019a)	67.9
TextRay (Bhutani et al., 2019)	60.3
QGG (Lan and Jiang, 2020)	74.0
ReTraCk (Chen et al., 2021)	71.0
CBR (Das et al., 2021)	72.8
RnG-KBQA (Ye et al., 2021)	75.1
ArcaneQA	**75.3**
w/o contextualized encoding	68.8

(c) WEBQSP

Table 2: Overall results on three datasets. ArcaneQA follows entity linking results from previous methods (i.e., RnG-KBQA's results on GRAILQA, QGG's results on WEBQSP, and Gu et al. (2021)'s results on GRAPHQ) for fair comparison. Model names with * indicate using the baseline entity linking results on GRAILQA.

grams that are not faithful to the KB, executing which will lead to empty answers. Note that post-hoc filtering by execution (Wang et al., 2018) can only help to a limited degree due to the KB's broad schema, while this type of mistake is rooted out in ArcaneQA by design.

Different from our search space pruning achieved with dynamic program induction, ranking-based models such as BERT+Ranking prunes unfaithful programs from their search space by ranking a set of faithful programs enumerated from the KB. These models typically make compromises on the complexity and diversity of programs during candidate enumeration. We break down the performance of ArcaneQA on GRAILQA's validation set in terms of question complexity and function types and show the fine-grained results in Table 3. The comparison with BERT-Ranking demonstrates the scalability and flexibility of our dynamic program induction. We also compare with RnG-KBQA, which adopts exactly the same candidate enumeration module as BERT+Ranking, but it is enhanced with a T5-based revision module to edit the enumerated programs into more diverse ones. We observe that RnG-KBQA performs uniformly well across different programs except for programs with superlative functions (i.e., ARGMAX/ARGMIN), i.e., the F1 of it is lower than ArcaneQA by over 50%. This is because in their candidate generation step, there is no superlative function enumerated. Despite the effectiveness of their T5-based revision, their performance still heavily depends on the diversity of candidate enumeration, which restricts the flexibility of their method.

Dynamic Contextualized Encoding. To show the key role of dynamic contextualized encoding, we use GloVe (Pennington et al., 2014) to provide non-contextualized embeddings for both questions and tokens in $\mathcal{V}$. We fix GloVe embeddings during training to make the model less biased to the training distribution (Gu et al., 2021) for GRAILQA and GRAPHQ, which address non-i.i.d. generalization, while for WEBQSP, we also update the word embeddings during training. Results in Table 2a show the importance of dynamic contextualized encoding, i.e., without contextualized encoding, the overall F1 decreases by 14.6%, 11.1%, and 6.5% on three datasets respectively. We also notice that dynamic contextualized encoding is more critical for non-i.i.d. generalization, i.e., on GRAILQA the F1 on i.i.d. generalization only decreases by 6.8%, while it decreases by 15.9% and 17.5%

Function	None	Count	Comparative	Superlative
BERT+Ranking	59.1/66.0	43.0/53.2	0.0/14.5	0/6.0
RnG-KBQA	**77.5/81.8**	**73.0/77.5**	**55.1/76.0**	13.8/22.3
ArcaneQA	70.8/77.8	62.5/68.2	54.5/75.7	**70.5/75.6**
# of relations	**1**	**2**	**3**	**4**
BERT+Ranking	57.4/61.5	39.8/54.7	0.0/22.9	0.0/25.0
RnG-KBQA	**75.7**/79.2	**65.4/74.8**	**28.6/44.4**	**100.0/100.0**
ArcaneQA	74.9/**80.9**	59.9/71.1	27.6/37.7	**100.0/100.0**

Table 3: Fine-grained results (EM/F1) on GRAILQA's dev set. **None** denotes programs with only AND and JOIN.

on compositional and zero-shot generalization. Without contextualized encoding, identifying the correct schema items from the KB in non-i.i.d. setting is particularly challenging. Schema linking powered by dynamic contextualized encoding is the key to non-i.i.d. generalization, which is a long-term goal of KBQA.

6.3 Efficiency Analysis

We compare the running time of ArcaneQA and ranking-based models in the online mode (i.e., no offline caching) to mimic the real application scenario. To make the comparison fair, we configure all models to interact with the KB via the same Virtuoso SPARQL endpoint. We run each model on 1,000 randomly sampled questions and report the average running time per question on a GTX 2080 Ti card. As shown below, our model is faster than BERT+Ranking and RnG-KBQA by an order of magnitude, because ArcaneQA dynamically prunes the search space and does not run the time-consuming queries for enumerating two-hop candidates.

	BERT+Ranking	RnG-KBQA	ArcaneQA
Time (s)	115.5	82.1	5.6

7 Conclusions

We present a novel generation-based KBQA model, ArcaneQA, which simultaneously addresses the large search space and schema linking challenges in KBQA with dynamic program induction and dynamic contextualized encoding. Experimental results on several popular datasets demonstrate the advantages of ArcaneQA in both effectiveness and efficiency. In the future, we will focus on developing generation-based KBQA models with stronger zero-shot generalizability. In addition, exploring other pre-trained language models such as T5 (Raffel et al., 2019) for generation-based KBQA is also an interesting direction.

Acknowledgement

The authors would like to thank the colleagues from the OSU NLP group and the anonymous reviewers for their thoughtful comments. This research was supported by NSF OAC 2112606.

References

Abdalghani Abujabal, Mohamed Yahya, Mirek Riedewald, and Gerhard Weikum. 2017. Automated template generation for question answering over knowledge graphs. In *Proceedings of the 26th International Conference on World Wide Web*, WWW '17, page 1191–1200, Republic and Canton of Geneva, CHE. International World Wide Web Conferences Steering Committee.

Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. 2015. Neural machine translation by jointly learning to align and translate. In *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*.

Jonathan Berant, Andrew Chou, Roy Frostig, and Percy Liang. 2013. Semantic parsing on Freebase from question-answer pairs. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*, pages 1533–1544, Seattle, Washington, USA. Association for Computational Linguistics.

Jonathan Berant and Percy Liang. 2014. Semantic parsing via paraphrasing. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1415–1425.

Nikita Bhutani, Xinyi Zheng, and HV Jagadish. 2019. Learning to answer complex questions over knowledge bases with query composition. In *Proceedings of the 28th ACM International Conference on Information and Knowledge Management*, pages 739–748.

Nikita Bhutani, Xinyi Zheng, Kun Qian, Yunyao Li, and H. Jagadish. 2020. Answering complex questions by combining information from curated and extracted

knowledge bases. In *Proceedings of the First Workshop on Natural Language Interfaces*, pages 1–10, Online. Association for Computational Linguistics.

Kurt Bollacker, Colin Evans, Praveen Paritosh, Tim Sturge, and Jamie Taylor. 2008. Freebase: a collaboratively created graph database for structuring human knowledge. In *Proceedings of the 2008 ACM SIGMOD international conference on Management of data*, pages 1247–1250.

Qingqing Cai and Alexander Yates. 2013. Semantic parsing Freebase: Towards open-domain semantic parsing. In *Second Joint Conference on Lexical and Computational Semantics (*SEM), Volume 1: Proceedings of the Main Conference and the Shared Task: Semantic Textual Similarity*, pages 328–338, Atlanta, Georgia, USA. Association for Computational Linguistics.

Shuang Chen, Qian Liu, Zhiwei Yu, Chin-Yew Lin, Jian-Guang Lou, and Feng Jiang. 2021. Retrack: A flexible and efficient framework for knowledge base question answering. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing: System Demonstrations*, pages 325–336.

Zi-Yuan Chen, Chih-Hung Chang, Yi-Pei Chen, Jijnasa Nayak, and Lun-Wei Ku. 2019. Uhop: An unrestricted-hop relation extraction framework for knowledge-based question answering. *arXiv preprint arXiv:1904.01246*.

Jianpeng Cheng, Siva Reddy, Vijay Saraswat, and Mirella Lapata. 2019. Learning an executable neural semantic parser. *Computational Linguistics*, 45(1):59–94.

Rajarshi Das, Manzil Zaheer, Dung Thai, Ameya Godbole, Ethan Perez, Jay-Yoon Lee, Lizhen Tan, Lazaros Polymenakos, and Andrew McCallum. 2021. Case-based reasoning for natural language queries over knowledge bases. *arXiv preprint arXiv:2104.08762*.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, Minneapolis, Minnesota. Association for Computational Linguistics.

Li Dong and Mirella Lapata. 2016. Language to logical form with neural attention. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 33–43.

Li Dong, Jonathan Mallinson, Siva Reddy, and Mirella Lapata. 2017. Learning to paraphrase for question answering. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 875–886, Copenhagen, Denmark. Association for Computational Linguistics.

Matt Gardner, Joel Grus, Mark Neumann, Oyvind Tafjord, Pradeep Dasigi, Nelson F. Liu, Matthew Peters, Michael Schmitz, and Luke Zettlemoyer. 2018. AllenNLP: A deep semantic natural language processing platform. In *Proceedings of Workshop for NLP Open Source Software (NLP-OSS)*, pages 1–6, Melbourne, Australia. Association for Computational Linguistics.

Yu Gu, Sue Kase, Michelle Vanni, Brian Sadler, Percy Liang, Xifeng Yan, and Yu Su. 2021. Beyond iid: three levels of generalization for question answering on knowledge bases. In *Proceedings of the Web Conference 2021*, pages 3477–3488.

Wonseok Hwang, Jinyeung Yim, Seunghyun Park, and Minjoon Seo. 2019. A comprehensive exploration on WikiSQL with table-aware word contextualization. *arXiv preprint arXiv:1902.01069*.

Yunshi Lan and Jing Jiang. 2020. Query graph generation for answering multi-hop complex questions from knowledge bases. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 969–974, Online. Association for Computational Linguistics.

Yunshi Lan, Shuohang Wang, and Jing Jiang. 2019a. Knowledge base question answering with topic units.(2019). In *Proceedings of the Twenty-Eighth International Joint Conference on Artificial Intelligence*, pages 5046–5052.

Yunshi Lan, Shuohang Wang, and Jing Jiang. 2019b. Multi-hop knowledge base question answering with an iterative sequence matching model. In *2019 IEEE International Conference on Data Mining (ICDM)*, pages 359–368. IEEE.

Chen Liang, Jonathan Berant, Quoc Le, Kenneth D. Forbus, and Ni Lao. 2017. Neural symbolic machines: Learning semantic parsers on Freebase with weak supervision. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 23–33, Vancouver, Canada. Association for Computational Linguistics.

Jeffrey Pennington, Richard Socher, and Christopher Manning. 2014. GloVe: Global vectors for word representation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1532–1543, Doha, Qatar. Association for Computational Linguistics.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J Liu. 2019. Exploring the limits of transfer learning with a unified text-to-text transformer. *arXiv preprint arXiv:1910.10683*.

Siva Reddy, Oscar Täckström, Slav Petrov, Mark Steedman, and Mirella Lapata. 2017. Universal semantic parsing. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 89–101, Copenhagen, Denmark. Association for Computational Linguistics.

Ohad Rubin and Jonathan Berant. 2021. SmBoP: Semi-autoregressive bottom-up semantic parsing. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 311–324, Online. Association for Computational Linguistics.

Torsten Scholak, Nathan Schucher, and Dzmitry Bahdanau. 2021. Picard: Parsing incrementally for constrained auto-regressive decoding from language models. *arXiv preprint arXiv:2109.05093*.

Semantic Machines, Jacob Andreas, John Bufe, David Burkett, Charles Chen, Josh Clausman, Jean Crawford, Kate Crim, Jordan DeLoach, Leah Dorner, Jason Eisner, Hao Fang, Alan Guo, David Hall, Kristin Hayes, Kellie Hill, Diana Ho, Wendy Iwaszuk, Smriti Jha, Dan Klein, Jayant Krishnamurthy, Theo Lanman, Percy Liang, Christopher H. Lin, Ilya Lintsbakh, Andy McGovern, Aleksandr Nisnevich, Adam Pauls, Dmitrij Petters, Brent Read, Dan Roth, Subhro Roy, Jesse Rusak, Beth Short, Div Slomin, Ben Snyder, Stephon Striplin, Yu Su, Zachary Tellman, Sam Thomson, Andrei Vorobev, Izabela Witoszko, Jason Wolfe, Abby Wray, Yuchen Zhang, and Alexander Zotov. 2020. Task-oriented dialogue as dataflow synthesis. *Transactions of the Association for Computational Linguistics*, 8:556–571.

Yu Su, Huan Sun, Brian Sadler, Mudhakar Srivatsa, Izzeddin Gür, Zenghui Yan, and Xifeng Yan. 2016. On generating characteristic-rich question sets for QA evaluation. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 562–572, Austin, Texas. Association for Computational Linguistics.

Danny Sullivan. 2020. A reintroduction to our Knowledge Graph and knowledge panels. blog.google.

Yawei Sun, Lingling Zhang, Gong Cheng, and Yuzhong Qu. 2020. Sparqa: Skeleton-based semantic parsing for complex questions over knowledge bases. In *Proceedings of the Thirty-Fourth National Conference on Artificial Intelligence*, AAAI'20. AAAI Press.

Ilya Sutskever, Oriol Vinyals, and Quoc V Le. 2014. Sequence to sequence learning with neural networks. In Z. Ghahramani, M. Welling, C. Cortes, N. D. Lawrence, and K. Q. Weinberger, editors, *Advances in Neural Information Processing Systems 27*, pages 3104–3112. Curran Associates, Inc.

Bailin Wang, Richard Shin, Xiaodong Liu, Oleksandr Polozov, and Matthew Richardson. 2020. RAT-SQL: Relation-aware schema encoding and linking for text-to-SQL parsers. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7567–7578, Online. Association for Computational Linguistics.

Chenglong Wang, Kedar Tatwawadi, Marc Brockschmidt, Po-Sen Huang, Yi Mao, Oleksandr Polozov, and Rishabh Singh. 2018. Robust text-to-sql generation with execution-guided decoding. *arXiv preprint arXiv:1807.03100*.

Tianbao Xie, Chen Henry Wu, Peng Shi, Ruiqi Zhong, Torsten Scholak, Michihiro Yasunaga, Chien-Sheng Wu, Ming Zhong, Pengcheng Yin, Sida I Wang, et al. 2022. Unifiedskg: Unifying and multi-tasking structured knowledge grounding with text-to-text language models. *arXiv preprint arXiv:2201.05966*.

Xi Ye, Semih Yavuz, Kazuma Hashimoto, Yingbo Zhou, and Caiming Xiong. 2021. Rng-kbqa: Generation augmented iterative ranking for knowledge base question answering. *arXiv preprint arXiv:2109.08678*.

Wen-tau Yih, Ming-Wei Chang, Xiaodong He, and Jianfeng Gao. 2015. Semantic parsing via staged query graph generation: Question answering with knowledge base. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 1321–1331, Beijing, China. Association for Computational Linguistics.

Wen-tau Yih, Matthew Richardson, Chris Meek, Ming-Wei Chang, and Jina Suh. 2016. The value of semantic parse labeling for knowledge base question answering. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 201–206, Berlin, Germany. Association for Computational Linguistics.

Tao Yu, Rui Zhang, Kai Yang, Michihiro Yasunaga, Dongxu Wang, Zifan Li, James Ma, Irene Li, Qingning Yao, Shanelle Roman, Zilin Zhang, and Dragomir Radev. 2018. Spider: A large-scale human-labeled dataset for complex and cross-domain semantic parsing and text-to-SQL task. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 3911–3921, Brussels, Belgium. Association for Computational Linguistics.

John M Zelle and Raymond J Mooney. 1996. Learning to parse database queries using inductive logic programming. In *Proceedings of the thirteenth national conference on Artificial intelligence*, pages 1050–1055.

Luke S Zettlemoyer and Michael Collins. 2005. Learning to map sentences to logical form: structured classification with probabilistic categorial grammars. In *Proceedings of the Twenty-First Conference on Uncertainty in Artificial Intelligence*, pages 658–666.

Rui Zhang, Tao Yu, He Yang Er, Sungrok Shim, Eric Xue, Xi Victoria Lin, Tianze Shi, Caiming Xiong, Richard Socher, and Dragomir Radev.

2019. Editing-based sql query generation for cross-domain context-dependent questions. *arXiv preprint arXiv:1909.00786*.

Victor Zhong, Caiming Xiong, and Richard Socher. 2017. Seq2sql: Generating structured queries from natural language using reinforcement learning. *arXiv preprint arXiv:1709.00103*.

A Meaning Representation

We provide a detailed description of our defined functions for S-expressions in Table 4. We provide annotations in S-expressions for several KBQA datasets, including WEBQSP, GRAPHQ, and COMPLEXWEBQ (which we did not use for experiments). All data files annotated by us can be found in our Github repo.

B Implementation Details

B.1 Entity Linking Results

For GRAILQA, we use the entity linking results provided by Ye et al. (2021); for GRAPHQ, we use the entity linking results provided by Gu et al. (2021); for WEBQSP, we follow the entity linking results provided by (Lan and Jiang, 2020). In addition, we find that answer types can be a strong clue for GRAILQA, so we also predict a set of FREEBASE classes for GRAILQA as a special type of entity. All entity linking results can be found in our Github repo.

B.2 Entity Anonymization

After identifying a set of entities, we do entity anonymization for WEBQ, i.e., we replace the entity mention with the type of the corresponding entity. For example, mention "*Barack Obama*" will be replaced by "*US president*". However, the entity linker might identify some false positive mentions, and anonymizing these mentions would lead to some critical information loss. To address this problem, we identify a set of common false positive mentions that contain important information about the question in training data. Such words include "government", "zip", etc. For mentions include these words, we do not do anonymization. Doing entity anonymization is a common practice on WEBQ, which can normally bring some gain of around 1 to 2 percent in F1, while for GRAILQA and GRAPHQ, we did not observe any improvement, so we keep the original entity mentions for these two datasets.

B.3 Hyper Parameters

For ArcaneQA, we are only able to train our model with batch size 1 due to the memory consumption, so we choose a workaround to set the number of gradient accumulations to 16. We use Adam optimizer with an initial learning rate of 0.001 to update our own parameters in BERT-based models. For BERT's parameters, we fine-tune them with a learning rate of 2e-5. For ArcaneQA w/o BERT, we train it with batch size 32 and an initial learning rate of 0.001 using Adam optimizer. For both models, the hidden sizes of both encoder and decoder are set to 768, and the dropout rate is set to 0.5. All hyper-parameters are manually tuned according to the validation accuracy on the development set. specifically, we do manual hyper-parameter search from [1e-5, 2e-5, 3e-5], [8, 16, 32], [0.0, 0.2, 0.5] to tune the learning rate of fine-tuning BERT, steps of gradient accumulation and dropout rate respectively.

B.4 Number of Model Parameters

Total numbers of trainable parameters of ArcaneQA and ArcaneQA w/o BERT are 123,652,608 and 261,900 respectively. The reason that the trainable parameters of ArcaneQA w/o BERT are so few is that we freeze the GloVe embeddings for non-i.i.d. generalization. The number of parameters becomes 121,205,100 if we take the GloVe embeddings into account.

B.5 Other Details

We summarize some other details in our implementation that are critical to reproduction.

We identify the literals in GRAILQA and GRAPHQ using hand-crafted regular expressions. There are two types of literals, i.e., date time and numerical value. Our regular expressions can identify around 98% of all literals.

During dynamic program induction of ArcaneQA, we follow the rules in Table 1 to run SPARQL queries to retrieve the admissible schema items. However, in some rare cases, the execution of a subprogram may contain a tremendous number of entities For example, the execution of `(JOIN USA people.person.nationality)` contains over 500,000 entities, and running SPARQL queries for all entities in them is infeasible. As a result, we only run SPARQL queries for 100 entities sampled from the execution results. One better choice could be to use some more efficient indexing to query the KB instead of using SPARQL.

We construct the vocabulary $\mathcal{V}$ for different datasets in different ways. For GRAILQA, following Gu et al. (2021), we construct the vocabulary using schema items from FREEBASECOMMONS. For GRAPHQ, we construct the vocabulary using schema items from the entire FREEBASE. For WEBQ, because it evaluates i.i.d. generalization, so we construct the vocabulary from its training data.

Function	Arguments	Returns
JOIN	a set of entities $u \subset (\mathcal{E} \cup \mathcal{L})$ and a relation $r \in \mathcal{R}$	all entities connecting to any $e \in u$ via r
AND	two set of entities $u1 \subset \mathcal{E}$ and $u2 \subset \mathcal{E}$	the intersection of two entities sets.
ARGMAX/ARGMIN	a set of entities $u \subset \mathcal{E}$ and a numerical relation $r \in \mathcal{R}$	a set of entities from u with the maximum/minimum value for r
LT(LE/GT/GE)	a numerical value $u \subset \mathcal{L}$ and a numerical relation $r \in \mathcal{R}$	all entities with a value $< (\leq / > / \geq) u$ for relation r
COUNT	a set of entities $u \subset \mathcal{E}$	the number of entities in u
CONS	a set of entities a set of entities $u \subset \mathcal{E}$, a relation $r \in \mathcal{R}$, and a constraint $c \in (\mathcal{E} \cup \mathcal{L})$	all $e \in u$ satisfying $(e, r, c) \in \mathcal{K}_r$
TC	a set of entities a set of entities $u \subset \mathcal{E}$, a relation $r \in \mathcal{R}$, and a temporal constraint $c \in \mathcal{L}$	all $e \in u$ satisfying $(e, r, c) \in \mathcal{K}_r$

Table 4: Detailed descriptions of functions defined in our S-expressions. We extend the definitions in Gu et al. (2021) by introducing two new functions CONS and TC. Also, we remove the function R and instead represent the inverse of a relation by adding a suffix "_inv" to it. Note that, for arguments in AND function, a class $c \in \mathcal{C}$ can also indicate a set of entities which fall into c.

	Overall		I.I.D.		Compositional		Zero-shot	
Model	**EM**	**F1**	**EM**	**F1**	**EM**	**F1**	**EM**	**F1**
BERT+Ranking (Gu et al., 2021)	51.0	58.4	58.6	66.1	40.9	48.1	51.8	59.2
RnG-KBQA (Ye et al., 2021)	**71.4**	76.8	**86.7**	**89.0**	61.7	68.9	**68.8**	**74.7**
ArcaneQA	69.5	**76.9**	86.1	89.2	**65.5**	**73.9**	64.0	72.8

Table 5: The results on the validation set of GRAILQA. The overall trend is basically consistent with the test set.

C Results on the Validation Set of GRAILQA

We show the results of ArcaneQA, BERT+Ranking, and RnG-KBQA on the validation set of GRAILQA in Table 5. We observe that ArcaneQA achieves a better F1 than RnG-KBQA. Overall, the trend is consistent with the test set. We also observe that the EM of ArcaneQA on zero-shot generalization is significantly higher than the test set, which is interesting and remains for further investigation.

Unsupervised Question Answering via Answer Diversifying

Yuxiang Nie[123], **Heyan Huang**[123]*, **Zewen Chi**[123], **Xian-Ling Mao**[123]
[1]School of Computer Science and Technology, Beijing Institute of Technology, Beijing, China
[2]Beijing Engineering Research Center of High Volume Language Information Processing and Cloud Computing Applications, Beijing, China
[3]Beijing Institute of Technology Southeast Academy of Information Technology, Fujian, China
{nieyx,hhy63,czw,maoxl}@bit.edu.cn

Abstract

Unsupervised question answering is an attractive task due to its independence on labeled data. Previous works usually make use of heuristic rules as well as pre-trained models to construct data and train QA models. However, most of these works regard named entity (NE) as the only answer type, which ignores the high diversity of answers in the real world. To tackle this problem, we propose a novel unsupervised method by diversifying answers, named **DiverseQA**. Specifically, the proposed method is composed of three modules: data construction, data augmentation and denoising filter. Firstly, the data construction module extends the extracted named entity into a longer sentence constituent as the new answer span to construct a QA dataset with diverse answers. Secondly, the data augmentation module adopts an answer-type dependent data augmentation process via adversarial training in the embedding level. Thirdly, the denoising filter module is designed to alleviate the noise in the constructed data. Extensive experiments show that the proposed method outperforms previous unsupervised models on five benchmark datasets, including SQuADv1.1, NewsQA, TriviaQA, BioASQ, and DuoRC. Besides, the proposed method shows strong performance in the few-shot learning setting.[1]

1 Introduction

Extractive question answering (extractive QA) aims to provide answer spans extracted from the context to answer questions. It can improve the interaction between humans and machines in applications such as search engines and dialog systems.

Existing extractive QA methods can be divided into two categories: supervised QA and unsupervised QA. Traditionally, for supervised QA (Seo et al., 2016), human-labeled context, questions and answers are given to train a QA model. Since the construction cost of labeled data is too high for supervised QA, recently, researchers pay more attention to the unsupervised setting, where QA pairs are unavailable. For example, Lewis et al. (2019) propose an unsupervised machine translation model to generate QA pairs. Li et al. (2020), Hong et al. (2020) and Lyu et al. (2021) try to alleviate the overlap between the context and the generated question. However, most existing works regard named entities as the only answer type, which ignores the high diversity of answers in the real world. For instance, as shown in Figure 1, in SQuADv1.1 (Rajpurkar et al., 2016) and NewsQA (Trischler et al., 2017), the answer type of named entities only account for 52.4% and 41.1% respectively.

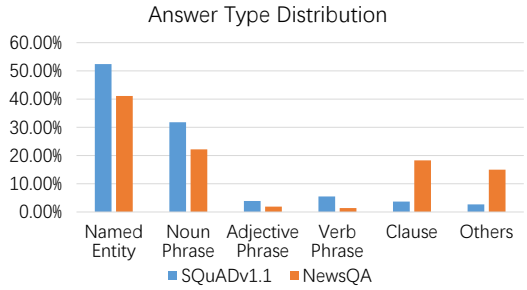

Figure 1: The answer type distributions of SQuADv1.1 and NewsQA, where we notice that named entities are just a fraction of each dataset.

To solve the problem, an intuitive way is to extract each type of answer spans independently in the raw text. Yet, it leads to two critical problems. Firstly, it's hard to determine the proportion of each answer type in the synthetic dataset. Secondly, extracting answers without any guidance could probably generate trivial and noisy question-answer pairs. To tackle the problem above, we propose **DiverseQA**, a novel unsupervised QA method. Specifically, the proposed method consists of three modules: data construction, data augmentation and denoising filter. Firstly, the data construction mod-

*Corresponding author

[1]We have released our codes and data in https://github.com/JerrryNie/DiverseQA.

Proceedings of the 29th International Conference on Computational Linguistics, pages 1732–1742
October 12–17, 2022.

ule employs a simple answer-extending rule to construct a dataset with diverse answers. As shown in Figure 2, a named entity (NE) is extracted and extended into a longer answer span, which should be a constituent of the sentence (e.g. noun phrase, verb phrase). In this way, the proportion of each type of answers can be obtained entirely from the original text. Besides, NE-based extension largely guarantees that extracted answers are meaningful. Secondly, an answer-type dependent data augmentation module is proposed. Concretely, an adjusting vector generator is designed to produce answer-type enhanced QA pairs in the embedding level. Then, a discriminator classifies the embeddings into the corresponding answer type while minimizing the KL divergence between the distribution and its prior to fool the discriminator in an adversarial way. Thirdly, the denoising filter is applied to alleviate the noise in the synthetic QA pairs.

Extensive experiments on six benchmarks, including SQuADv1.1 (Rajpurkar et al., 2016), TriviaQA (Joshi et al., 2017), NaturalQuestions (Kwiatkowski et al., 2019), NewsQA (Trischler et al., 2017), BioASQ (Tsatsaronis et al., 2015) and DuoRC (Saha et al., 2018) show that DiverseQA outperforms previous unsupervised methods on five datasets and obtains comparable results on one dataset among unsupervised QA models. Further analysis shows that DiverseQA can largely improve the question-answering ability of the model on diverse answer types. In addition, our method shows strong performance in the few-shot learning setting.

The contributions of our method are as follows:

- We propose DiverseQA, a novel method to improve an unsupervised QA model to handle answers beyond named entities.
- Our method outperforms previous unsupervised works on five benchmarks and reaches the comparable result on a benchmark.
- Further analysis shows that the drift of answer length distribution and the quality of extracted answers are important to the performance of the model.

2 Related Work

Data Augmentation. Data augmentation methods can be regarded as regularizers to make the model robust and reduce dependence on the training data. In computer vision domains (Krizhevsky et al., 2012), geometric transformation and color space transformation are effective. In natural language processing, word removing, synonym replacing and back-translation can enlarge the diversity of examples (Xie et al., 2020). Lee et al. (2021) proposes an embedding-level data augmentation method to improve the performance of QA on out-of-distribution data. This work relies on supervised (low noise) training instances, while in the unsupervised (high noise) scenario, embedding-level data augmentation methods have not been explored yet.

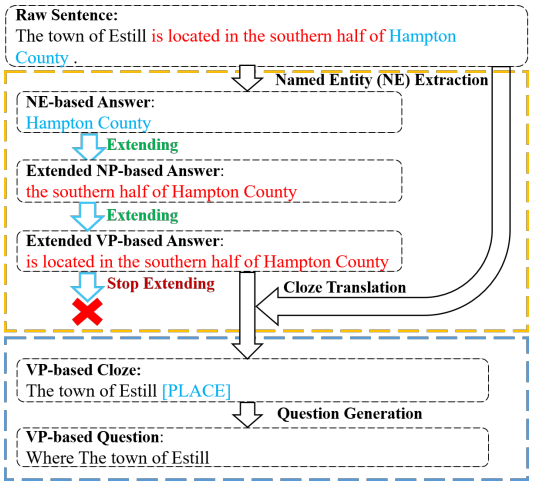

Figure 2: An example of our data constructing module. We extract the named entity (NE) in the *Raw Sentence* and extend it into a longer sentence constituent until meeting the stop extending condition. The final extended answer span is a VP in this example, which is used in cloze translation for question generation.

Extractive Question Answering. Extractive question answering (extractive QA) aims to output an answer span given the context (containing the answer) and the question. It has gained much progress with the help of large labeled datasets such as SQuAD, NewsQA, and TriviaQA. To better handle these datasets, strong extractive QA models are proposed, including BiDAF (Seo et al., 2016), BERT (Devlin et al., 2018) and RoBERTa (Liu et al., 2019). However, because they largely depend on human-annotated data, the lack of labeled data for some specific domains constrains the capacity of the supervised extractive QA model.

Unsupervised Question Answering. Unsupervised question answering (unsupervised QA) becomes attractive among researchers recently (Lewis et al., 2019). Like supervised extractive QA, given context and question, the model needs to extract a text span from the context to answer the question. The difference is that in the unsupervised setting, models need to learn to answer the question without any human-labeled ⟨context, question, answer⟩

triples, which is a more challenging task than the supervised counterpart. Lewis et al. (2019) extracts named entity from the context, and then trains an unsupervised neural machine translation model to convert the cloze question into a natural question. Li et al. (2020) makes use of cited documents and a refine phase to alleviate serious question-context overlapping and improve answer diversity. Hong et al. (2020) uses paraphrasing and trimming to remove unanswerable questions as well as alleviate the context-question similarity problem. Lyu et al. (2021) takes advantage of a supervised summarization dataset to tackle the context-question overlapping problem. However, most of the existing works regard the named entity (NE) as the only answer type, while other types (e.g. noun phrases, adjective phrases, verb phrases, sub-clauses) are nearly ignored. Although in Lewis et al. (2019), the inclusion of noun phrases (NPs) was discussed, the reported poor performance eventually led the author to use the NE-based synthetic QA pairs. In Li et al. (2020), even though generating more answers is considered, most of the model-generated answers have strong relationships with the named entity and we will discuss it later in Section 4.3.4.

3 Method

To explore answer types beyond named entities, we propose an unsupervised method **DiverseQA**, which can be divided into three modules, data augmentation, data construction and denoising filter. Firstly, a simple but effective span extending rule is applied in the data construction module to do answer extraction and natural question generation to construct a synthetic QA dataset with diverse answers. Secondly, an answer-type dependent data augmentation module via adversarial training is designed to produce high-quality augmented QA pairs. Thirdly, a denoising filter is proposed to alleviate high noise in the dataset. In the next few subsections, we will introduce the three modules in detail.

3.1 Synthetic QA Data Construction via Span Extension

In this section, we illustrate our QA data constructing module, which can be divided into two steps. Firstly, we extract answers in the text through span extension. Secondly, we take advantage of pseudo-NER label to generate questions.

3.1.1 Answer Generation via Span Extension

As shown in Figure 2, given a *Raw Sentence*, first of all, we extract the named entity span *Hampton County*. Then, we extract all the constituents containing *Hampton County*. We extend the span into a longer constituent iteratively. When the next extending span makes up more than $\omega\%$ of the *Raw sentence*, we stop extending and regard the extended span as the final answer span. In this example, the NE is firstly extended into a longer NP. Again, the NP is extended into the VP. As the VP '*is located in the southern half of Hampton County*' is the longest constituent while making up less than $\omega\%$ (called *Span Extending Threshold*) of the *Raw Sentence*, it becomes the answer span of the QA pair. The intuition is that even if many constituents are trivial, the constituent containing the named entity and making up a proper portion of the *Raw Sentence* might be meaningful to create a high-quality QA pair.

3.1.2 Question Generation with Pseudo-NER Label

To construct a NE-based QA pair, the NER label of the answer span is mapped to a question word (Lewis et al., 2019) for generating the natural question. However, it cannot be directly applied to other answer types. To tackle the problem, we regard the original NER label of the named entity as the *pseudo-NER label* of the extended answer span. After that, we replace the answer span with the high-level NER mask token (Lewis et al., 2019). The intuition is that the semantic information of the extracted named entity can be probably consistent with the extended constituent.

Followed Lewis et al. (2019) and Li et al. (2020), we do the mask token mapping and cloze to natural question conversion.

3.2 Answer-type Dependent Data Augmentation

We firstly introduce the background of extractive QA, and then illustrate how we design our answer-type dependent data augmentation modules.

3.2.1 Backgrounds of Extractive Question Answering

Given question $\mathbf{q} = (q_1, q_2, ..., q_m)$ and context $\mathbf{c} = (c_1, c_2, ..., c_n)$, extractive QA models aim to predict the start and end token of the answer span $\mathbf{a} = (a_1, a_2)$ from the context. Assuming that there are N observations: $\{\mathbf{c}^{(i)}, \mathbf{q}^{(i)}, \mathbf{a}^{(i)}\}_{i=1}^{N}$, we

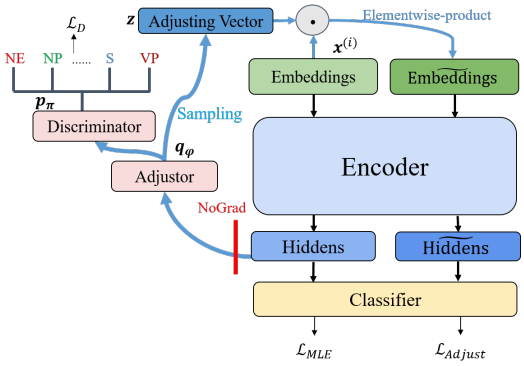

Figure 3: Our data augmentation module. The Embeddings of the input sequence are firstly encoded into the Hiddens. Then, the Adjustor module receives the Hiddens and produces an Adjusting Vector to adjust the Embeddings by the element-wise product. Meanwhile, the Discriminator classifies the produced vector into an answer type to make the vector answer-type dependent.

estimate the model parameter θ by maximizing the following function:

$$\mathcal{L}_{\text{MLE}}(\theta) = \sum_{i=1}^{N} p_\theta(\mathbf{a}^{(i)}|\mathbf{c}^{(i)}, \mathbf{q}^{(i)}) \quad (1)$$

3.2.2 Answer-type Dependent Embedding Adjustment

As QA instances might have specific feature related to answer types, motivated by Lee et al. (2021), we design an answer-type dependent embedding-level *Adjusting Vector* to create high quality instances for model training.

The proposed adjusting vector $\mathbf{z} \in \mathbb{R}^{d\times(m+n+r)}$ is sampled from the distributions $q_\phi(\mathbf{z}|\mathbf{x}, l)$ to augment the input sequence $\mathbf{x} \in \mathbb{R}^{d\times(m+n+r)}$, where r is the special token length, d is the size of a word embedding vector, $\mathbf{x}$ is the embeddings, l is an answer type. We use the element-wise production between the embedding and the Adjusting Vector as extra data to train a QA model by maximizing the log-likelihood function of $p_\theta(\mathbf{a}|\mathbf{x}, \mathbf{z})$:

$$\mathcal{L}_{\text{Adjust}}(\theta, \phi) = \sum_{i=1}^{N} \mathbb{E}_{q_\phi(\mathbf{z}|\mathbf{x},l)} \left[\log p_\theta(\mathbf{a}^{(i)}|\mathbf{x}^{(i)}, \mathbf{z})\right] + \beta \text{KL}\left(q_\phi(\mathbf{z}|\mathbf{x}^{(i)}, l)||p_\psi(\mathbf{z})\right) \quad (2)$$

where $p_\psi(\mathbf{z})$ is the prior of the distribution $q_\phi(\mathbf{z}|\mathbf{x}, l)$, ψ is the predetermined parameter of the distribution. We assume it obeys multivariate Gaussian distributions $\mathcal{N}(\mathbf{1}, \gamma\mathbf{I_d})$.

To make the embeddings answer-type dependent, we train a neural network as the *discriminator* to classify the adjusting vector into the corresponding answer type. The discriminator receives the adjusting vector $\mathbf{z}_j \in \mathbb{R}^d$ and classifies the vector into a corresponding answer type by using the predicting distribution $p_\pi(l|\mathbf{z}_j) = \frac{e^{f_l(\mathbf{z}_j)}}{\sum_{i=1}^{L} e^{f_i(\mathbf{z}_j)}}$, where $f_i(\mathbf{z}_j)$ denotes the logit of answer type label i given $\mathbf{z}_j$. Due to imbalanced label distribution among different answer types, followed Menon et al. (2020), we modify the distribution as: $p'_\pi(l|\mathbf{z}_j) = \frac{e^{f_l(\mathbf{z}_j)+\log\ p_l}}{\sum_{i=1}^{L} e^{f_i(\mathbf{z}_j)+\log\ p_i}}$, where p_i is the frequency of the answer type i and π is the parameters of the discriminator. The discriminator is trained via optimizing the following loss:

$$\mathcal{L}_D(\pi|\mathbf{z}) = \sum_{i=1}^{N}\sum_{j=1}^{P}\sum_{l=1}^{L} y_{il} p'_\pi(l|z_j^{(i)}) \quad (3)$$

where P is the length of the input sequence, L is the number of answer types. $y_{il} = 1$ when the i-th observation is related to the answer type l, otherwise $y_{il} = 0$. To fool the discriminator, the Adjustor needs to minimize the KL divergence between its distribution q_ϕ and the prior p_ψ in Eqn. 2.

The Final Objective Function. The final objective function is as follows:

$$\mathcal{L}(\theta, \phi, \pi) = \mathcal{L}_{\text{MLE}}(\theta) + \mathcal{L}_{\text{Adjust}}(\theta, \phi) + \alpha\mathcal{L}_D(\pi) \quad (4)$$

where α is the hyperparameter weighting between the question-answering predicting loss and the answer-type discriminating loss.

3.3 Denoising Filter

We design a denoising filter to further alleviate the negative effect of the noise in the data. The filter is composed of the *Top-K Filter* and the *Substring Filter*. To apply them, we firstly use the QA model to do inference on the unseen synthetic data. Then, if the synthetic answer falls in the K answers with the highest probability the model predicts (Top-K Filter) or the predicted instance is a substring of a NE-based answer span with predicting probability higher than γ (Substring Filter), we keep it. Otherwise, we remove the instance. Then, we use the filtered data to train our fine-tuned model. In this process, Top-K Filter aims to choose QA pairs with high confidence as low noise instances. Substring Filter keeps extra NE-based QA pairs with high predicting probabilities for training. The idea comes from an observation that a substring of a NE can probably represent the NE. For example, in the sentence *Apple CEO Tim Cook introduces*

Models	SQuADv1.1 EM/F1	TriviaQA EM/F1	NQ EM/F1	NewsQA EM/F1	BioASQ EM/F1	DuoRC EM/F1
Supervised Models						
Match-LSTM	64.1/73.9	-/-	-/-	-/-	-/-	-/-
BiDAF	66.7/77.3	-/-	-/-	-/-	-/-	-/-
BERT-base	81.2/88.5	69.4/74.3◁	66.1/77.9◁	49.4/64.4◁	-/-	-/-
BERT-large	84.2/91.1	75.7/80.2◁	69.0/80.8◁	56.0/71.0◁	-/-	-/-
Trained with Supervised Summarization Dataset						
Lyu et al. (2021)	65.6/74.5	36.7/43.0	46.0/53.5	37.5/50.1	32.0/43.2	38.8/46.5
Unsupervised Models						
Lewis et al. (2019)	44.2/54.7	19.1/23.8†	27.5/35.1†	19.6/28.5‡	18.9/27.0†	26.0/32.6†
RefQA	62.5/72.6	48.6/58.2‡	**43.4**/55.7‡	33.6/46.3	42.5/58.9‡	38.0/49.4‡
DiverseQA	**67.6/76.9**	**52.5/60.8**	41.3/**56.3**	**37.5/51.3**	**47.2/61.4**	**46.9/56.3**

Table 1: Results (EM/F1) of our method and various baselines on six different datasets. '†' denotes results taken from Lyu et al. (2021). '‡' denotes results from our reimplementation of RefQA(Li et al., 2020). '◁' denotes our fine-tuned BERT on supervised data. Because the pre-processed training sets of BioASQ and DuoRC in MRQA are not released, we don't fine-tune BERT on them.

two new products, "Tim Cook" is a NE referred to a specific person. Besides, "Tim" or "Cook" can also refer to him. Therefore, when the model predicts a substring of a NE with a high probability, it might refer to the original NE as the answer.

4 Experiments

4.1 Experiment Setup

Unsupervised QA Dataset Construction. We use Wikiref (Li et al., 2020) as the original text to construct QA pairs.

To extract answer spans, firstly, we use Spacy[2] to extract all of the named entities and their NER labels in the passage. Then, we apply Berkeley Neural Parser (Kitaev and Klein, 2018) to parse each sentence and extract a longer constituent containing a named entity with the constraint of $\omega\%$ sentence length as the final answer span. Here, we set $\omega = 80$. In our experiment, we consider named entity (NE), noun phrase (NP), adjective phrase (ADJP), verb phrase (VP), and sub-clause (S) as the candidate answer types. The dataset consists of 908,511 QA pairs. We randomly sample 300,000 to initially train a QA model, 600,000 to split them into N =6 parts (followed the empirical results in Li et al. (2020)) for the filtering phase.

Question Answering Model Settings. We use BERT as the backbone of our QA model. We use Adam (Kingma and Ba, 2014) as the optimizer. The learning rate is 3e-5 and the batch size is 24. The max sequence length is 384 and the doc stride is 128. The discriminator is set as a one-layer network. We set $L = 5, \alpha = 1, \beta = 1$. We use BERT-large-uncased-whole-word-masking, train the model for 2 epochs, save the checkpoint every 1,000 training steps and use the dev set to evaluate them for early stopping. Then, we continuously train the model with filtered data via the denoising filter, where $K = 1$ and the substring threshold $\gamma = 0.1$.

4.2 Results

We evaluate our model on SQuAD v1.1, NewsQA, TriviaQA, NaturalQuestions (NQ), BioASQ and DuoRC. We compare DiverseQA with supervised approaches (Wang and Jiang, 2016; Seo et al., 2016; Devlin et al., 2018), unsupervised approaches (Lewis et al., 2019; Li et al., 2020) and the approach using a supervised summarization dataset (Lyu et al., 2021). We use Exact Match (EM) and F1 score as our metrics. We use the pre-processed data provided in MRQA (Fisch et al., 2019).

The experimental results on six different benchmarks are shown in Table 1. The model trained on the synthetic QA data created by our DiverseQA reaches the state-of-the-art on five benchmarks, which shows the competitive performance of the proposed method in a wide range of domains. However, we find that our model underperforms on NaturalQuestions (NQ) dataset on exact match (EM). It is because the answers in this dataset are all entities, which means the model that only learns information from named entity[3] can have good performance. After all, the model only needs to choose a proper entity from an entity set rather than extract

[2] https://spacy.io

[3] Although 'named entity' and 'entity' are different, they share the common features in most aspects.

a possible span from the whole context, which finally leads to a good EM value. But as we know, named entities (or entities) are not the only answer type in the real world. Therefore, it's unfair for our method. Because Lyu et al. (2021) (the line under '*Trained with Supervised Summarization Dataset*' in Table 1) makes use of supervised summarization dataset XSUM (Narayan et al., 2018) to train the model, which is not a purely unsupervised method, we don't compare our method with it.

4.3 Analysis

We conduct experiments in this section to further understand our method. BERT-base-uncased is used to complete each experiment.

4.3.1 Effects of Different Components of DiverseQA

We conduct experiments on different components of DiverseQA. A brief illustration is as follows:

NeAnsQA Only extract NE as the answer type.

DiverseAnsQA Take the data-extending strategy proposed to build a dataset with diverse answers.

RandomAnsQA Each answer span in *RandomAnsQA* dataset has the same length with that of DiverseAnsQA while the answer span is randomly extended from the original NE-based answer.

Adjusting Vector It imposes the adjusting vector produced by the 'Adjustor' module on the embeddings of input tokens as an augmentation instance.

Answer-type Discriminator The module is to classify the adjusting vector into an answer type.

Top-K Filter Apply the Top-K Filter described in Section 3.3.

Substring Filter Apply the Substring Filter described in Section 3.3.

As shown in Table 2, the result illustrates that each component can improve the performance of our model. The difference between *NeAnsQA* and *DiverseAnsQA* shows that the use of our data construction strategy can greatly improve the performance of the model. It also demonstrates the importance of diverse answer types in unsupervised QA. Because answer length distribution is changed from NeAnsQA to DiverseAnsQA, we design *RandomAnsQA* to get rid of this extra factor. It shows that DiverseAnsQA still outperform RandomAnsQA by a large gap, which demonstrates the effectiveness of the proposed QA data construction strategy. The result of *Adjusting Vector* and *Answer-type Discriminator* shows that although adding the adjusting vector can slightly improve the performance, answer-type discriminator can continuously improve the performance, showing that adding the answer-type constraint to the distribution of the adjusting vector can benefit the performance of the model. To verify the necessity of the Discriminator, we also use a simple 'Answer-type Classifier' to classify input sequences into different answer types. The result shows that a simple classifier is not adequate to improve the method. In addition, the gains by adding the *Top-K Filter* reveal the importance of the filtering phase in the training process. What's more, *Substring Filter* can further gain the performance of the model. What's more, the results in the last row of Table 2 show that the Substring Filter is useful.

	EM	F1
NeAnsQA	49.2	59.3
RandomAnsQA	48.7	62.0
DiverseAnsQA	52.1	64.0
+ Answer-type Classifier*	51.3	63.6
+ Adjusting Vector	52.3	64.0
+ Answer-type Discriminator	52.9	64.6
+ Top-K Filter	54.7	65.6
+ Substring Filter	**55.0**	**66.2**

Table 2: Ablations on each component of DiverseQA method on the SQuADv1.1 development set. Components below the component with '*' are not added upon the '*' component.

ω%	20%	40%	60%	80%	100%
F1	63.0	63.6	65.4	**66.2**	62.2

Table 3: F1 scores of different Span Extending Thresold evaluated on SQuADv1.1 dev set.

4.3.2 Effects of Span Extending Threshold

We experiment with several Span Extending Threshold to construct the synthetic dataset. Table 3 shows that the optimal value is 80%. It illustrates that neither a too strict nor a too loose span extending condition can produce a high quality dataset.

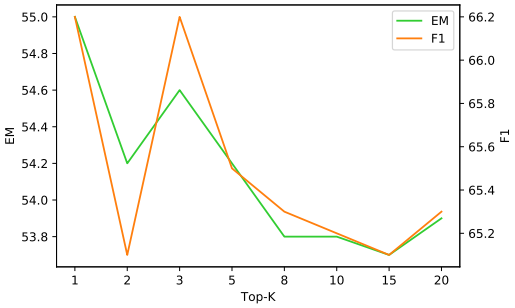

Figure 4: Results of using K answers with the highest predicting probabilities in the denoising filtering phase.

γ	0.0	0.1	0.2	0.4	0.6	0.8
F1	65.2	**66.2**	65.2	65.2	64.9	65.0

Table 4: F1 scores of different Substring Filter threshold evaluated on SQuADv1.1 dev set.

4.3.3 Effects of Denoising Filter

We conduct experiments on the K answers with the highest predicting probabilities as well as Substring Filter threshold γ, which will be used in the continuing training. Figure 4 shows that when K=1, the performance reaches the best. It means that when the model-predicted answer with high probability matches the span extracted via our extending strategy, the corresponding QA pair is probably of high quality and can be further reused to improve the performance of the model. Table 4 shows that when γ=0.1, the produced NE-based QA pairs with predictions of substring can be beneficial.

Furthermore, we make comparison between the proposed Denoising Filter and the Refining Phase (Li et al., 2020). Table 5 shows that firstly, DiverseQA largely outperforms RefQA without any filter. Secondly, the proposed Denoising Filter can be effectively used in the second phase of both RefQA and DiverseQA, showing its strong adaptation to different methods, while Refining Phase can only make improvement in RefQA. Thirdly, although the F1 score of RefQA with Denoising Filter is slightly lower than that of Refining Phase, it makes large improvement on the EM metric, demonstating the advantage of the proposed Denoising Filter.

4.3.4 Effects of Diverse Answer Types

We experiment on the SQuADv1.1 dev set partitioned by answer types. For a fair comparison, the re-implement of RefQA shares the raw text (from which we generate QA pairs and train the model) and the random seed with ours.

	No Filter	Refine	Denoise
RefQA	49.2/59.3	52.0/62.6	53.3/62.2
DiverseQA	**52.1/64.0**	**52.1/64.0***	**55.0/66.2**

Table 5: EM/F1 scores of RefQA and DiverseQA with different continuously training strategy. "Refine" denotes the Refining phase in RefQA. "Denoise" denotes the proposed Denoising Filter. '*' means that the model cannot make any improvement on the specific continuous training strategy.

Although Li et al. (2020) claims that their refine phase can generate diverse answers, the result in Table 6 shows that DiverseQA outperforms the performance of RefQA on nearly every kind of answer type, especially on VP and S by large margins. It is because the refine phase proposed in RefQA heavily relies on the QA model training with purely NE-based QA pairs. Therefore, the trained model could probably generate certain variants of NEs, which are short and used to continuously train the model, leaving relatively long answer spans like VPs and Ss out. Besides, we observe that the performance of DiverseQA is lower than RefQA on ADJP in the exact match (EM). It is because the ADJP merely accounts for 0.3% (in Appendix A) in the whole synthetic dataset generated by DiverseQA, while the refining phase in RefQA may probably generate many variants of NEs, which could be ADJPs[4]. Consequently, the model in RefQA might train with too many ADJP-like QA instances and prefer to choose the ADJPs as answer types, leading to the result in Table 6. Since in our method, we only consider NE, NP, ADJP, VP, S as our answer spans, it's important to know how well our method performs on other types of answers out of consideration. The result in the '*Others*' column of Table 6 demonstrates that the proposed model can generalize to other unseen answer types and outperform RefQA by a large gap.

In addition, we ablate the proposed span extending strategy and Answer-type Discriminator separately to explore the performance of the two components. The result in the last two rows of Table 6 shows that the two components can both contribute to the performance of model on most of answer

[4]For instance, in '*he was still being paid more than $ 10,000 as a legal advisor to the Chicago*', span '*10,000*' is a named entity with NER label 'MONEY' while '*$ 10,000*' is a variant of the named entity as well as an ADJP.

Models	NE EM/F1	NP EM/F1	ADJP EM/F1	VP EM/F1	S EM/F1	Others EM/F1	Overall EM/F1
RefQA	67.0/75.4	47.9/60.4	**31.9**/41.4	4.1/18.3	22.7/35.0	36.7/52.2	52.0/62.6
RandomAnsQA	58.8/70.7	43.2/58.8	25.0/39.7	9.4/25.6	21.4/39.0	36.8/53.4	48.7/62.0
DiverseAnsQA	62.3/72.4	47.2/61.3	26.5/39.3	13.2/29.3	22.9/41.0	38.5/54.6	52.1/64.0
DiverseQA	67.5/75.7	**49.1/62.8**	27.9/**42.5**	**13.3/30.4**	**29.1/42.6**	**41.4/57.2**	**55.0/66.2**
w/o Extension	**71.5/77.3**	48.5/60.2	24.5/35.1	1.6/12.1	21.9/30.6	34.4/47.0	52.0/60.6
w/o Discriminator	66.9/74.7	47.6/60.1	28.4/40.5	9.7/28.0	26.3/40.9	40.4/54.7	53.6/63.9

Table 6: EM/F1 scores of different models on the SQuADv1.1 dev set. Columns are each dev set associated with the type of answer. 'Others' means other types of answers out of our consideration. 'Extension' represents our answer span extending module. 'Discriminator' denotes the proposed Answer-type Discriminator.

Context: The Town of Estill is located in the southern half of Hampton County .	
Question: Where of the southern half in The town of Estill is located	**Question:** Where of the town Estill
The answer in RefQA: Hampton County	**Our answer:** is located in the southern half of Hampton County

Table 7: An example of RefQA (left half) and our QA pairs (right half).

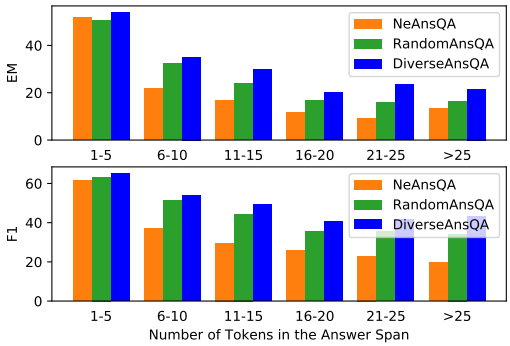

Figure 5: Comparison among *NeAnsQA*, *RandomAnsQA* and *DiverseAnsQA* (Section 4.3.1). SQuADv1.1 dev set is partitioned by answer length for evaluation.

types. Besides, we notice that the performance of "w/o Extension" on the NE obtains the best. It is because that under this setting, the model will only learn from NE-based QA pairs. Therefore, it prefers to choose a NE from the context, which can gain its performance on NE-based QA pairs.

What's more, we also show an example to make the comparison between the QA pair in DiverseQA and that of RefQA in Table 7. In the example, the question 'Where of the town Estill' needs more reasoning process to be answered correctly than the question generated in RefQA. As in the proposed method, when the generated answer becomes longer, the generated question becomes shorter. Therefore, it's also necessary to keep the question and answer length unchanged while exploring the effectiveness of answer diversity. In Table 6, the results of RandomAnsQA and DiverseAnsQA demonstrate the usefulness of the proposed answer extension method when both the question and answer length distributions are unchanged. More examples are shown in Appendix D.

4.3.5 Effects of Answer Length Distribution

In the proposed method, the first module is to extend the span into a longer constituent. Although it indeed introduces new answer types and gains performance, answer length distribution also changes simultaneously. Therefore, we further explore how the drift of answer length distribution affects the model's performance.

As shown in Figure 5, the performance of NeAnsQA is lower than RandomAnsQA and DiverseAnsQA. When the answer length becomes larger (>10 tokens), the gap is even larger. This demonstrates that model trained only with NE-based QA pairs lacks the ability to handle QA pairs with long answers. Besides, it can be found that the performance of RandomAnsQA is lower than DiverseAnsQA on each answer length. It means that without the influence of answer distribution, the proposed span extending strategy can generate higher quality instances than that of randomly extending strategy.

4.4 Few-Shot Learning

We conduct experiments in the few-shot learning setting. For a fair comparison, we use our best-trained model without the data augmentation component, our re-implementation of RefQA, and a BERT model, all of which use the BERT-large-whole-word-masking as the backbone.

Figure 6 shows that our model reaches the best

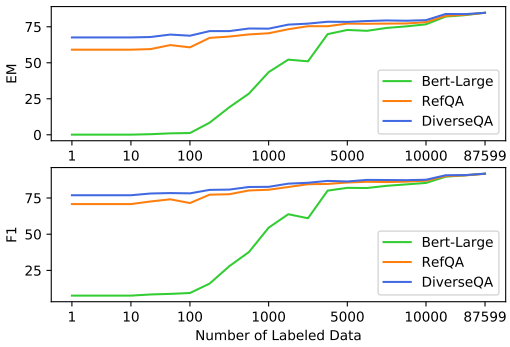

Figure 6: Few-shot results (EM and F1) using different sizes of SQuADv1.1 training data, comparing among DiverseQA, RefQA (Li et al., 2020) and BERT-large-uncased-whole-word-masking.

result trained on various sizes of supervised data ranging from 1 to 50,000, demonstrating the strong ability of our method in the low-resource scenario.

5 Conclusion

We propose DiverseQA, an unsupervised QA method comprising a synthetic QA dataset with diverse answers, an answer-type-dependent data augmentation process via adversarial training, and a denoising filter to improve the performance of a QA model. Our method reaches the state-of-the-art on five benchmarks and shows strong performance in the few-shot learning setting.

Acknowledgements

The work is supported by National Key R&D Plan (No. 2020AAA0106600), National Natural Science Foundation of China (No. U21B2009, 62172039 and L1924068). We would like to acknowledge Yuming Shang for the helpful discussions.

References

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*.

Adam Fisch, Alon Talmor, Robin Jia, Minjoon Seo, Eunsol Choi, and Danqi Chen. 2019. Mrqa 2019 shared task: Evaluating generalization in reading comprehension. In *Proceedings of the 2nd Workshop on Machine Reading for Question Answering*, pages 1–13.

Giwon Hong, Junmo Kang, Doyeon Lim, and Sung-Hyon Myaeng. 2020. Handling anomalies of synthetic questions in unsupervised question answering. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 3441–3448.

Mandar Joshi, Eunsol Choi, Daniel S Weld, and Luke Zettlemoyer. 2017. Triviaqa: A large scale distantly supervised challenge dataset for reading comprehension. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1601–1611.

Diederik P Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*.

Nikita Kitaev and Dan Klein. 2018. Constituency parsing with a self-attentive encoder. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 2676–2686, Melbourne, Australia. Association for Computational Linguistics.

Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. 2012. Imagenet classification with deep convolutional neural networks. *Advances in neural information processing systems*, 25:1097–1105.

Tom Kwiatkowski, Jennimaria Palomaki, Olivia Redfield, Michael Collins, Ankur Parikh, Chris Alberti, Danielle Epstein, Illia Polosukhin, Jacob Devlin, Kenton Lee, et al. 2019. Natural questions: A benchmark for question answering research. *Transactions of the Association for Computational Linguistics*, 7:452–466.

Seanie Lee, Minki Kang, Juho Lee, and Sung Ju Hwang. 2021. Learning to perturb word embeddings for out-of-distribution qa. *arXiv preprint arXiv:2105.02692*.

Patrick Lewis, Ludovic Denoyer, and Sebastian Riedel. 2019. Unsupervised question answering by cloze translation. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 4896–4910.

Zhongli Li, Wenhui Wang, Li Dong, Furu Wei, and Ke Xu. 2020. Harvesting and refining question-answer pairs for unsupervised qa. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 6719–6728.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Roberta: A robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692*.

Chenyang Lyu, Lifeng Shang, Yvette Graham, Jennifer Foster, Xin Jiang, and Qun Liu. 2021. Improving unsupervised question answering via summarization-informed question generation. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 4134–4148.

Aditya Krishna Menon, Sadeep Jayasumana, Ankit Singh Rawat, Himanshu Jain, Andreas Veit, and Sanjiv Kumar. 2020. Long-tail learning via logit adjustment. *arXiv preprint arXiv:2007.07314*.

Shashi Narayan, Shay B Cohen, and Mirella Lapata. 2018. Don't give me the details, just the summary! topic-aware convolutional neural networks for extreme summarization. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 1797–1807.

Pranav Rajpurkar, Jian Zhang, Konstantin Lopyrev, and Percy Liang. 2016. Squad: 100,000+ questions for machine comprehension of text. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 2383–2392.

Amrita Saha, Rahul Aralikatte, Mitesh M Khapra, and Karthik Sankaranarayanan. 2018. Duorc: Towards complex language understanding with paraphrased reading comprehension. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1683–1693.

Minjoon Seo, Aniruddha Kembhavi, Ali Farhadi, and Hannaneh Hajishirzi. 2016. Bidirectional attention flow for machine comprehension. *arXiv preprint arXiv:1611.01603*.

Adam Trischler, Tong Wang, Xingdi Yuan, Justin Harris, Alessandro Sordoni, Philip Bachman, and Kaheer Suleman. 2017. Newsqa: A machine comprehension dataset. In *Proceedings of the 2nd Workshop on Representation Learning for NLP*, pages 191–200.

George Tsatsaronis, Georgios Balikas, Prodromos Malakasiotis, Ioannis Partalas, Matthias Zschunke, Michael R Alvers, Dirk Weissenborn, Anastasia Krithara, Sergios Petridis, Dimitris Polychronopoulos, et al. 2015. An overview of the bioasq large-scale biomedical semantic indexing and question answering competition. *BMC bioinformatics*, 16(1):1–28.

Shuohang Wang and Jing Jiang. 2016. Machine comprehension using match-lstm and answer pointer. *arXiv preprint arXiv:1608.07905*.

Qizhe Xie, Zihang Dai, Eduard Hovy, Thang Luong, and Quoc Le. 2020. Unsupervised data augmentation for consistency training. *Advances in Neural Information Processing Systems*, 33.

Appendix

A The Distribution of Extracted Answers

The distribution of our extracted answer types (NE, NP, ADJP, VP, S) is shown in Table 8. The frequencies of each answer type are used in the modified predicting distribution described in Section 3.2.2. Since previous works consider named entities as the only answer type (i.e. NE acounts for 100% of all the answers), we don't show the answer type distribution of them. Besides, as the answer span generated by the trained QA model in the refining phase of Li et al. (2020) fails to follow the rule of constituent parsing, we cannot obtain its answer type distribution in terms of sentence constituents.

	#Instances	%Frequency
NE	716,716	78.9%
NP	161,178	17.7%
ADJP	2,563	0.3%
VP	23,420	2.6%
S	4,634	0.5%

Table 8: The statistics of extracted answer types in the synthetic dataset constructed using DiverseQA.

B Algorithms

We describe the whole training procedure as follows:

Algorithm 1: Training Procedure

Data: Synthetic QA dataset $\mathcal{D}$, a BERT model $\mathcal{M}$ with answer-type dependent data augmentation module, a denoising filter composed of a Top-K filter and a Substring Filter with threshold γ.

Result: A fine-tuned model $\mathcal{M}'$.

Split $\mathcal{D}$ into $\mathcal{D}_I$ and $\mathcal{D}_F$;
Fine-tune $\mathcal{M}$ with $\mathcal{D}_I$;
Split $\mathcal{D}_F$ equally into $\{\mathcal{D}_{F_i}\}_{i=1}^{N}$;
for $i \leftarrow 1$ **to** N **do**
    $\mathcal{S} \leftarrow \emptyset$;
    foreach element e in $\mathcal{D}_{F_i}$ **do**
        Obtain the probability p_e using $\mathcal{M}$;
        if p_e is in the Top-K highest probabilities **then**
            $\mathcal{S} \leftarrow \mathcal{S} \cup \{e\}$
        end
        if the answer of e is a subtring of the extracted one **and** $p_e \geq \gamma$ **then**
            $\mathcal{S} \leftarrow \mathcal{S} \cup \{e\}$
        end
    end
    Fine-tune model M with dataset $\mathcal{S}$;
end

C Answer Length Distribution

We randomly sample 10,000 instances from SQuADv1.1, the dataset of RefQA, and the dataset of DiverseQA respectively, and count the answer length distributions, which are shown in Figure 7. It

Context: Joss Whedon has endorsed Mitt Romney (in a way) and now " The Simpsons " Mr. Burns , owner of Springfield 's nuclear power plant , titan of corporate capitalism and honcho in the Springfield Republican Party , has come out with his own backhanded endorsement of the Republican nominee .	
Question: Who As the chief of "Springfield Republican Party" endorsed Mitt Romney in 2012 US Presidential Election . **Answer:** Burns	**Question:** Who As the chief of endorsed Mitt Romney in 2012 US Presidential Election . **Answer:** Mr. Burns
Context: In 1931 , the Singers gave the Museum of Fine Arts to the community along with a substantial collection of American and European art .	
Question: Who American and art of a substantial collection with along the Singers gave the Museum of Fine Arts to the community . **Answer:** European	**Question:** Who art of a substantial collection with along the Singers gave the Museum of Fine Arts to the community . **Answer:** American and European
Context: Why do the new prequels sometimes contradict the history set forth in THE DUNE ENCYCLOPEDIA compiled by, Dr. Willis E. McNelly ?, THE DUNE ENCYCLOPEDIA reflects an alternate " DUNE universe " which did not necessarily represent the " canon " created by Frank Herbert ."	
Question: Who by written to accompany the "Dune" books **Answer:** Willis E. McNelly	**Question:** Who by written to accompany the "Dune" books **Answer:** Dr. Willis E. McNelly
Context:GDP per capita in the city increased by 2.4 per cent and employment by 4.7 per cent compared to the previous year .	
Question: How much by GDP per capita in the city increased **Answer:** 2.4 per cent	**Question:** How much in GDP per capita **Answer:** increased by 2.4 per cent

Table 9: Examples of the QA pairs of RefQA (left half) and that of DiverseQA (right half).

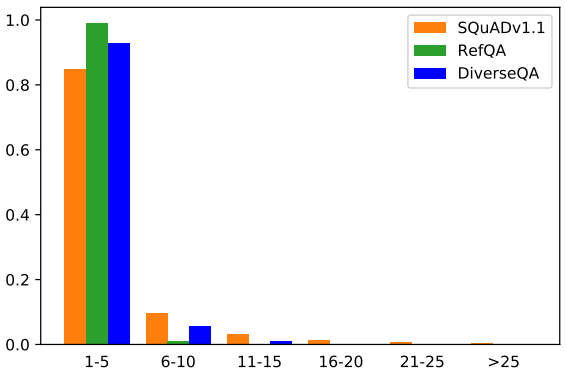

Figure 7: Answer length distributions.

shows that firstly, the proposed DiverseQA method can construct dataset with a similar answer length distribution to the annotated dataset SQuADv1.1. Secondly, the dataset only with NE-based QA pairs (like RefQA) is not able to cover long form answers (it accounts for 0% QA pairs in the answer length ranges '11-15', '21-25' and '>25')[5], which is harmful to the performance of model on long answers.

D Generated QA Instances

Examples of generated QA instances are shown in Table 9.

[5]The dataset of DiverseQA accounts for 0.16%, 0.13% and 0.03% in the ranges '16-20', '21-25' and '>25'. And it is not displayed properly in Figure 7.

Weakly Supervised Formula Learner for Solving Mathematical Problems

Yuxuan Wu and **Hideki Nakayama**
The University of Tokyo
{wuyuxuan,nakayama}@nlab.ci.i.u-tokyo.ac.jp

Abstract

Mathematical reasoning task is a subset of the natural language question answering task. Existing work suggested solving this task with a two-phase approach, where the model first predicts formulas from questions and then calculates answers from such formulas. This approach achieved desirable performance in existing work. However, its reliance on annotated formulas as intermediate labels throughout its training limited its application. In this work, we put forward the idea to enable models to learn optimal formulas autonomously. We proposed Weakly Supervised Formula Learner, a learning framework that drives the formula exploration with weak supervision from the final answers to mathematical problems. Our experiments are conducted on two representative mathematical reasoning datasets MathQA and Math23K. On MathQA, our method outperformed baselines trained on complete yet imperfect formula annotations. On Math23K, our method outperformed other weakly supervised learning methods. [1]

1 Introduction

Mathematical reasoning is a task where mathematical problems are described in natural language or mathematical symbols. Such problems require values, expressions, or other mathematical representations as answers. A naive approach to solving this task is to treat it as an end-to-end token-by-token predicting problem from questions to answers. However, this approach showed a relatively poor generalization capacity on unseen numbers (Saxton et al., 2019). Another approach to solving mathematical reasoning task is to adopt a two-phase methodology. In the first phase, specific formulas are predicted for each question. In the second phase, such formulas are calculated under predefined rules to produce the final answers. This approach has been widely applied in recent work and has achieved desirable results in many representative mathematical reasoning datasets (Wang et al., 2017; Amini et al., 2019). However, this two-phase solution leads to a reliance on annotated formulas as indispensable labels for training the formula predictor in the first phase. This reliance further results in two major weaknesses. Firstly, ground-truth formula annotations are not necessarily prepared for every mathematical problem and dataset. This makes it impossible to extend this solution to datasets without these annotations. Secondly, the learning process can be misled when there is noise in the formula annotations. In consideration of this, we are motivated to propose a new learning framework for solving mathematical problems that is not dependent on formula annotations. On the whole, we followed the principle of the two-phase methodology and implemented the two phases with what we call `PolicyNet` and `ActTaker`. Figure 1 is an overview of our learning framework. Our main contributions in this work can be summarized as follows:

- We established a new mechanism to learn formulas with weak supervision from final answers, which outperforms existing weakly supervised learning methods.
- We enabled models to explore reasonable formulas autonomously through a heuristic search in the space of possible formulas.
- We verified that the formulas learned with weak supervision can be more beneficial to the question answering than complete yet imperfect formula annotations.

2 Related Work

2.1 Mathematical Reasoning

In recent years, various datasets have been published to study the capacity of machine learning

[1] The software is available at https://github.com/evan-ak/wsfl.

Proceedings of the 29th International Conference on Computational Linguistics, pages 1743–1752
October 12–17, 2022.

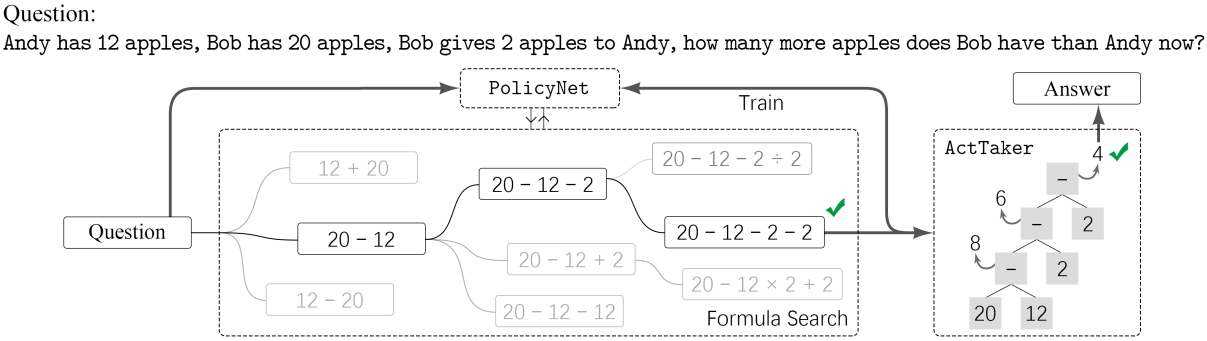

Figure 1: To get rid of the dependence on annotated formulas, our learning framework conducts a search process to explore optimal formulas. Such formulas are then fed back to the formula predictor for its training.

models in solving mathematical problems and performing quantitative reasoning. Math23K (Wang et al., 2017) is a dataset crawled from a couple of online education websites consisting of 23,162 problems with formula annotations. MathQA (Amini et al., 2019) is a dataset collected from another former dataset named AQuA (Ling et al., 2017) consisting of 37,200 problems with formula annotations. Mathematics (Saxton et al., 2019) is a complex large-scale dataset in which questions are generated in a broad range of areas including algebra, arithmetic, and calculus.

2.2 Mathematical Problem Solvers

End-to-end and formula-based methods are the two common methodologies for solving mathematical problems. The essential difference between them is that they either produce final answers directly or adopt formulas as intermediate labels.

Concretely, end-to-end methods simply regard both questions and answers as sequences of alphabets, digits, and symbols, and conduct a sequence-to-sequence prediction (Saxton et al., 2019). The application of these methods is not restricted by the absence of formulas. However, the lack of the concept of complete numbers forces them to receive and predict rational numbers digit by digit, which leads to a relatively weak generalization capacity.

On the other hand, formula-based methods employ what are called formulas or equations as intermediate labels for solving mathematical problems. Most elementary applied numerical mathematical problems can be solved by building equations with unknowns and solving the equations to acquire the answers. This generated the idea of first letting the model predict such equations and then solving the equations in a rule-based manner. This idea was first implemented by Wang et al. (2017) and then improved in later work (Wang et al., 2018; Xie and Sun, 2019; Zhang et al., 2020; Chen et al., 2020a). Faced with the problem of the reliance on formula annotations, Hong et al. (2021) proposed a fixing mechanism to learn formulas through error propagation and formula correction.

2.3 Semantic Parsing

Semantic parsing is the task of translating natural language utterances into machine-understandable logical form (Kamath and Das, 2019). Recent studies on solving mathematical problems have also benefited from semantic parsing by automatically synthesizing formulas from questions (Koncel-Kedziorski et al., 2015; Shi et al., 2015; Hopkins et al., 2017). However, considering that there is no guarantee that semantic parsing necessarily provide valid formulas for every question, the invalid formulas become noise if they are fed to following learning processes without correction. As a result, weakly supervised formula learning remains meaningful and valuable as long as perfect formula annotations are not prepared. In view of this, semantic parsing is considered an approach that works in parallel with weakly supervised formula learning methods for solving mathematical problems.

2.4 Neural Module Networks

Neural Module Networks (NMNs) are another relevant existing approach with a similar two-phase methodology (Andreas et al., 2016b). For solving visual question answering tasks, NMNs first predict programs from the questions and then compute with modules to acquire the final answers. Later work also succeeded in applying NMNs to solve discrete reasoning problems (Yi et al., 2018). Faced with the similar difficulty in training a two-phase model, existing work either utilized rein-

Question : Andy has 12 apples, Bob has 20 apples, Bob gives 2 apples to Andy, how many more apples does Bob have than Andy now?

Equation : 20 - 12 - 2 - 2

q : Andy has ⟨N0⟩ apples, Bob has ⟨N1⟩ apples, Bob gives ⟨N2⟩ apples to Andy, how many more apples does Bob have than Andy now?

num : (12, 20, 2)

a : 4

f : (-, -, -, N_0, ⟨End⟩, N_1, ⟨End⟩, N_2, ⟨End⟩, N_2, ⟨End⟩)

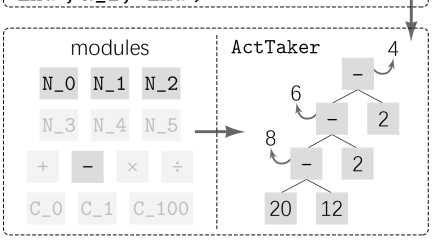

Figure 2: The preprocessing and the symbols applied in our work.

forcement learning (Andreas et al., 2016a; Johnson et al., 2017) or developed specific heuristic learning algorithms (Wu and Nakayama, 2020).

3 Weakly Supervised Formula Learner

3.1 Problem Definition

To achieve a formal representation of the mathematical problems in given datasets, we first performed some preprocessing on the question–answer pair of each problem. As shown in Figure 2, given the raw text of a question, the numbers that appear in the text are extracted as num. On the other hand, the numbers in the original text are replaced with special tokens ⟨Nx⟩, where x is the index of each number. We refer to the questions with these replaced tokens as "templates", which can also be simply denoted by q. Note that multiple questions can share the same template if they only differ in numbers. In this case, they are combined into a single template q. We let $\{num\}$ denote the set of numbers num extracted from the questions corresponding to the same template, and $\{a\}$ denote the set of answers a to these questions. Note that num in $\{num\}$ and a in $\{a\}$ should be kept paired. After preprocessing, the data visible to the following procedures should be triplets of $(q, \{num\}, \{a\})$.

Although the ground-truth equations for solving corresponding problems are annotated in some datasets and considered part of the training data in supervised learning methods, they are not visible to our learning framework, which performs weakly supervised learning. Instead, we let f denote the formula we used. Each formula is made up of a sequence of tokens where each of the tokens is the name of a module or an ⟨End⟩ sign. This sequence provides a preorder traversal of the tree of desired modules where ⟨End⟩ marks the leaf nodes. Details of the modules are presented in Section 3.5.

3.2 General Learning Process

Generally, our proposed learning framework is made up of PolicyNet and ActTaker. As shown by Equations 1 and 2, PolicyNet takes the question template q as the input and predicts the formula f. ActTaker takes the formula f and the set of numbers $\{num\}$ as inputs and calculates a set of answers $\{\hat{a}\}$ corresponding to each num.

$$f = \texttt{PolicyNet}(q) \quad (1)$$

$$\{\hat{a}\} = \texttt{ActTaker}(f, \{num\}) \quad (2)$$

With these two models, Algorithm 1 shows the general learning process of our Weakly Supervised Formula Learner. Here, $\mathbb{D}$ denotes the original dataset holding triplets of training data $(q, \{num\}, \{a\})$. $\mathbb{L}$ denotes a dictionary initialized to be empty for storing the optimal formulas found through the learning process. After PolicyNet, ActTaker, and $\mathbb{L}$ get initialized, the learning process is composed of numerous basic loops. Within each loop, at first, a triplet of training data is sampled from the dataset $\mathbb{D}$. Then, a search process is conducted to try to find the optimal formula for solving the given question template q. The behavior of this Search function is presented in detail in Section 3.3. After the optimal formula f and its accuracy $accu_f$ on q are obtained, they are used to update the dictionary $\mathbb{L}$. Concretely, if f is not None, and then if no f has been recorded for q or $accu_f$ exceeded the previously recorded accuracy, f and $accu_f$ will be recorded for q in $\mathbb{L}$. Finally, PolicyNet is

Algorithm 1 General Learning Process

```
PolicyNet, ActTaker ← Init()
𝕃 ← {}
for loop in range(max_loop) do
    (q, {num}, {a}) ← Sample(𝔻)
    f, accu_f ← Search(PolicyNet, ActTaker, q, {num}, {a})
    𝕃.update(q: (f, accu_f))
    PolicyNet.train(𝕃)
end for
```

trained with the q and f sampled from $\mathbb{L}$. These procedures are repeated until max_loop is reached. By the time of inference, this process is no longer required, and the answer can be acquired directly through Equations 1 and 2.

3.3 Formula Search

As shown by line 5 of Algorithm 1, we conduct a `Search` procedure in each loop to explore optimal formulas. For this procedure, we basically followed the Graph-based Heuristic Search algorithm proposed by Wu and Nakayama (2020) and adapted it to meet the needs of solving mathematical problems. The process of this procedure is presented by Algorithm 2.

Graph In this algorithm, $\mathcal{G}$ denotes a graph used to store the formulas under exploration with each of its nodes representing a unique formula. This graph is maintained under the following two rules:

- Each of its nodes n_f represents a unique formula f.
- There is an edge between two nodes if and only if the edit distance between the formulas they represent is one.

Moreover, each node n_f maintains a score $n_f.score$. Intuitively, this score indicates how desirable a formula is for solving the current given question. Wu and Nakayama (2020) suggested binding this score to the question answering accuracy. However, in mathematical problems, unreasonable formulas are highly likely to lead to scattered answers and thus result in zero accuracies. These scores can no longer provide enough guidance in a heuristic search if most of them degenerate to zero. In view of this, we modified this score to the average of two factors. Among them, the one is the standardized likelihood of the formula given by `PolicyNet`, and the other is an index related to the actual question answering effectiveness of the formula. In this section, we employ a concise implementation for the latter factor, which is simply the question answering accuracy. We will present another delicately designed way to scale this factor in Section 3.4. In general, $n_f.score$ is defined by Equations 3 to 6.

$$n_f.score = \frac{1}{2}[\, p(f|q,\theta_P) + f.accu\,] \tag{3}$$

$$p(f|q,\theta_P) = \frac{1}{L}\sum_{i=1}^{L} p(f_i|f_{1:i-1},q,\theta_P) \tag{4}$$

$$f.accu = \frac{1}{N}\sum_{i=1}^{N} \mathbf{1}_{|\{\hat{a}\}_i - \{a\}_i|<10^{-3}} \tag{5}$$

$$\{\hat{a}\} = \texttt{ActTaker}(f,\ \{num\}) \tag{6}$$

Here, θ_P denotes the parameters of `PolicyNet`. L denotes the length of the current formula. $p(f_i|f_{1:i-1},q,\theta_P)$ denotes the likelihood of producing the ith token in the formula given `PolicyNet`, the question template q, and the previous tokens $f_{1:i-1}$. N denotes the length of sets $\{num\}$ and $\{a\}$ corresponding to template q. $\mathbf{1}_{|\{\hat{a}\}_i-\{a\}_i|<10^{-3}}$ returns one if the difference between the ith result in $\{\hat{a}\}$ and the ith answer in $\{a\}$ is less than an acceptable floating-point error bound 10^{-3}.

Graph Initialization As for the initialization of the graph as shown in Line 2 of Algorithm 2, $\mathcal{G}$ is initialized with at most three nodes. The formulas of these nodes are as follows:

- `(N_0,`$\langle$`End`$\rangle$`)`, which is the shortest legal formula;
- the formula predicted by `PolicyNet` given q with maximum likelihood;
- the formula recorded for q_c, where q_c is the previously solved template recorded in $\mathbb{L}$ that the current q is semantically closest to.

For the last clause, we determine the semantic distance between two templates by calculating the Euclidean distance between their sentence vectors embedded by the encoder of `PolicyNet`. With $E_P(\cdot)$ denoting the encoder of `PolicyNet`, the decision of q_c can be expressed by Equation 7.

$$q_c = \arg\min_{q^*\in\mathbb{L}} \|E_P(q^*) - E_P(q)\|_2 \tag{7}$$

Algorithm 2 Formula Search

1: **func** `Search(PolicyNet, ActTaker,` q, $\{num\}$, $\{a\}$)
2: $\quad\mathcal{G} \leftarrow$ `Init()`
3: $\quad$**for** $iter$ **in** range(max_iter) **do**
4: $\quad\quad f_{exp} \leftarrow$ `Sample(`$\mathcal{G}$`)`
5: $\quad\quad f_{exp}.accu \leftarrow$ Accuracy(`ActTaker(`f_{exp}, $\{num\}$), $\{a\}$)
6: $\quad\quad\mathcal{G}$.update(`Mutate(`f_{exp}`)`)
7: $\quad$**end for**
8: $\quad f_{best} \leftarrow \arg\max_{f\in\mathcal{G}} f.accu$
9: $\quad$**return** f_{best}, $f_{best}.accu$

Formula Sampling As shown in Line 4 of Algorithm 2, in every iteration of the search, we first select a formula f_{exp} from $\mathcal{G}$ as the formula to explore. For the `Sample` function, we also followed the mechanism proposed by Wu and Nakayama (2020). Concretely, there is an $Expectation$ value defined on every node n of $\mathcal{G}$ as presented by Equation 8. In every iteration of the search, the node with the greatest $Expectation$ value among unexplored nodes is selected as the node to be explored.

$$n.Exp = \sum_{d=0}^{3} w_d * max\{n^*.score \mid n^* \in \mathcal{G}, distance(n^*, n) \leqslant d\} \quad (8)$$

$$w = [0.5,\ 0.25,\ 0.15,\ 0.1] \quad (9)$$

Formula Examination As shown in Line 5 of Algorithm 2, the selected formula f_{exp} is examined by `ActTaker` to obtain its accuracy given $\{num\}$ and $\{a\}$. The calculation of the accuracy basically follows Equations 5 and 6. In addition, the examination by `ActTaker` may not necessarily succeed because f_{exp} is not guaranteed to be semantically legal and illegal calculations like division by zero may be encountered. In such cases, corresponding $\{\hat{a}\}_i$ is considered invalid and $|\{\hat{a}\}_i - \{a\}_i|$ is considered infinite.

Formula Mutation As shown in Line 6 of Algorithm 2, mutations are generated from f_{exp} to expand the graph $\mathcal{G}$. Here, insertion, deletion, and substitution are the three operations for generating mutations. Respectively, they insert new modules into a formula, delete existing modules from a formula, and substitute existing modules with other modules in a formula. The newly generated formulas are then added to $\mathcal{G}$ if they do not yet exist in $\mathcal{G}$. The relevant edges should also be added to $\mathcal{G}$ to keep $\mathcal{G}$ conforming to its definition and features.

When all the search iterations are finished, the formula that achieved the highest accuracy is returned together with its accuracy as the result of this formula search. If none of the formulas achieved non-zero accuracy, this function returns `None`.

3.4 Difference-Based Formula Scoring

In Section 3.3, we have presented an elementary practice for determining $n_f.score$, the score of each formula, with Equations 3 to 6. In this section, we further discuss Difference-Based Scoring (DBS), another advanced approach to determining this score based on the difference between answers acquired from formulas and ground-truth answers. This difference-based scoring technique has also been suggested and verified in recent work represented by Petersen et al. (2021).

$$n_f.score = p(f|q, \theta_P) + \beta * f.diffscore \quad (10)$$

$$f.diffscore = \frac{1}{N} \sum_{i=1}^{N} \frac{1}{1 + 10 \ln\left(|\{\hat{a}\}_i - \{a\}_i| + 1\right)} \quad (11)$$

With DBS, $n_f.score$ is calculated through Equations 10 and 11. In Equation 10, β is a hyperparameter that scales the contribution of $f.diffscore$ in $n_f.score$ with default value 1.0. In Equation 11, for each specific i, if $\{\hat{a}\}_i$ is equal to $\{a\}_i$, the corresponding term returns one, the same value as Equation 5. Otherwise, it returns a score negatively correlated with the difference between $\{\hat{a}\}_i$ and $\{a\}_i$. The 10 in the denominator is an empirical coefficient that ensures the following logarithmic difference would not contribute too much to the whole score. With this score, formula search tends to explore formula nodes surrounding the nodes that lead to answers close to ground-truth answers.

3.5 Modules

Modules are the basic calculating units for solving each mathematical problem. During computation, the modules in the tree specified by a formula are calculated recursively from the leaves to the root to acquire the final answer. The three types of modules we adopted are `Number`, `Operation`, and `Constant`.

Number The `Number` modules, which are denoted by `N_x`, are employed to establish references to the numbers extracted from questions. These modules need no input and return a number. Here, `x` is the index of the number that is referred to. This index starts from 0. For example, `N_1` returns the second number in num. An error is raised if `x` exceeds the number of numbers in num.

Operation The `Operation` modules, which are denoted simply by their symbols, are employed to conduct specific mathematical calculations. These modules need a specific number of numeric inputs (commonly two) and return the calculation result as a number. For example, the `+` module takes two rational numbers `a` and `b` as inputs and returns the rational number `(a+b)`. An error is raised if the calculation is illegal such as division by zero.

	Accuracy
Seq2prog (Amini et al., 2019)	51.9%
Seq2prog+cat (Amini et al., 2019)	54.2%
LSTM2TP (Chen et al., 2020a)	54.6%
TP-N2F (Chen et al., 2020a)	55.9%
Ours w/o DBS	59.5%
Ours w/ DBS	**60.1%**

Table 1: The option selecting accuracy achieved by our learning framework and baselines on MathQA.

Constant The `Constant` modules, which are denoted by `C_x`, are employed to generate constant numbers. These modules need no input and return a number. Here, `x` is the specific rational number that is referred to. For example, `C_100` returns the integer 100.

4 Experiments

Our experiments in this work are conducted on two representative mathematical reasoning datasets, MathQA (Amini et al., 2019) and Math23K (Wang et al., 2017). We report and discuss our findings on them in Section 4.1 and 4.2, respectively.

4.1 MathQA

Experimental setup To prepare the training data, we followed the preprocessing procedure presented in Section 3.1 to transform the original questions and answers into triplets of $(q, \{num\}, \{a\})$. For `PolicyNet`, the encoder is a two-layer Bidirectional LSTM (Hochreiter and Schmidhuber, 1997; Schuster and Paliwal, 1997) with hidden state size 256. The decoder is a two-layer LSTM with hidden state size 512. The input embedding size of both of them is 300. For their training, we adopted the Adam optimizer (Kingma and Ba, 2015) with learning rate 0.001. As indicated by line 7 of Algorithm 1, `PolicyNet` is trained continuously in every learning loop. Here, the batch size for sampling training data from $\mathbb{L}$ is 64. Within each loop, `PolicyNet` is trained on 500 batches. For the comparison with baselines trained in a fully-supervised manner, we utilized part of the formula annotations to pretrain `PolicyNet`. The pretraining data, which is organized as tuples of (q, f), is filtered from the training set to meet the following two requirements. First, the formula should be able to solve the question and achieve non-zero accuracy. Second, the formula should be made up of only the four fundamental arithmetic calculations.

	Accuracy
Ours / raw annotation	52.4%
Ours / REINFORCE	56.5%
Ours	60.1%

Table 2: The performance of our models trained with raw formula annotations and formula exploration.

For `ActTaker`, we adopted twenty `Number` modules, `N_0` to `N_19`, four `Operation` modules, `+`, `−`, `×`, and `÷`, and four `Constant` modules, `C_1`, `C_2`, `C_3`, and `C_100`.

Evaluation metric For evaluation, MathQA provides five options for each question where the correct option is annotated. To select an option, we calculate the differences between the answer acquired by our models and each option, and select the option for which the difference is minimal. If an error is raised through the computation of `ActTaker`, we randomly select one of the options. The final accuracy we report is the accuracy of option selection. Note that this metric is different from the metric of measuring the formula matching accuracy, which is adopted by some existing work (Chen et al., 2020b) and tolerates the inherent noise in formula annotations. The results analyzed under these two metrics are not directly comparable, while we choose the former to study the influence on learning brought by noisy annotations.

Results Table 1 shows the accuracy achieved by our learning framework and baselines on MathQA. It is shown that our proposed method outperforms all the baselines on this dataset. We also conducted an ablation study on whether to employ the DBS discussed in Section 3.4 or not. The result confirmed the performance improvement made by DBS. Compared with the baselines, we only adopt a simple LSTM for the formula inference, which appears to be naive in contrast to the delicately crafted models adopted in existing work. We attribute our success to the autonomous formula exploring capacity of our learning framework.

In our investigation of the annotated formulas provided by MathQA, some noise was found. This means that part of the annotated formulas cannot solve the corresponding questions correctly. Including this noise in the training labels results in the degradation of performance in existing work. However, our learning framework is capable of removing this noise and finding valid formulas for the

question index	14328 (train)
question text	12.5 % of 192 = 50 % of ? **(Answer: 48)**
annotated formula	(((50 × 192) ÷ 100) × 12.5) ÷ 100 **(wrong)**
our formula (raw)	(÷, ×, N_0, ⟨End⟩, N_1, ⟨End⟩, N_2, ⟨End⟩)
(flatten)	(12.5 × 192) ÷ 50
question index	1752 (train)
question text	how many multiples of 4 are there between 8 and 160 ? **(Answer: 37)**
annotated formula	((160 − 8) ÷ 4) + 1 **(wrong)**
our formula (raw)	(−, ÷, −, N_2, ⟨End⟩, N_1, ⟨End⟩, N_0, ⟨End⟩, C_1, ⟨End⟩)
(flatten)	((160 − 8) ÷ 4) − 1
question index	3999 (train)
question text	the average of first 25 natural numbers is ? **(Answer: 13)**
annotated formula	25 + 1 **(wrong)**
our formula (raw)	(+, ÷, N_0, ⟨End⟩, C_2, ⟨End⟩, ÷, C_1, ⟨End⟩, C_2, ⟨End⟩)
(flatten)	(25 ÷ 2) + (1 ÷ 2)
question index	23604 (train)
question text	the telephone company wants to add an area code composed of 2 letters to every phone number . in order to do so , the company chose a special sign language containing 324 different signs . if the company used 322 of the signs fully and two remained unused , how many additional area codes can be created if the company uses all 324 signs ? **(Answer: 1292)**
annotated formula	322 × (324 − 322) × (324 − 322) **(wrong)**
our formula (raw)	(+, ×, N_0, ⟨End⟩, N_1, ⟨End⟩, ×, N_0, ⟨End⟩, N_2, ⟨End⟩)
(flatten)	(2 × 322) + (2 × 324)
question index	285 (test)
question text	today jim is twice as old as fred , and sam is 4 years younger than fred . 4 years ago jim was 8 times as old as sam . how old is jim now ? **(Answer: 20)**
annotated formula	(((8 × 8) − 4) ÷ (8 − 2)) × 2
our formula (raw)	(×, +, N_0, ⟨End⟩, ÷, +, ×, N_0, ⟨End⟩, C_2, ⟨End⟩, −, ×, N_1, ⟨End⟩, N_2, ⟨End⟩, N_1, ⟨End⟩, −, N_0, ⟨End⟩, N_1, ⟨End⟩, C_2, ⟨End⟩) **(wrong)**
(flatten)	(4 + ((4 × 2) + ((4 × 8) − 4)) ÷ (4 − 4)) × 2 **(wrong)**

Table 3: Examples of five questions together with annotated formulas and the formulas discovered by our learning framework. The first four examples show the cases that the annotated formulas are invalid and lead to incorrect answers. However, our learning framework discovered valid formulas for these questions instead. The last example shows a case of failure in the test. In this case, an incorrect formula is predicted by our models and an error is raised through the computations of the modules because of the occurrence of division by zero. In cases like this, formulas cannot produce valid answers even though they are semantically acceptable.

corresponding questions afresh through the search with weak supervision from the answers. This enables our learning framework to achieve higher accuracy in this task. Table 3 provides a case study for this issue.

To strengthen this idea, we also compared the performance of our models trained in different conditions and report the results in Table 2. Here, "raw annotation" refers to the models trained merely on the raw formula annotations. "REINFORCE" refers to the models trained with raw annotations and have REINFORCE (Williams, 1992) implemented to enable a preliminary formula exploration. We restricted the maximum number of attempts on formulas for both REINFORCE and our learning framework to the same bound 10^8. Compared with the learning on raw annotations, although REINFORCE improved the performance to some extent, our method showed a more powerful formula exploring capacity. This comparison verified the advantage of autonomous formula exploration and the superiority of our learning framework over the naive reinforcement learning method.

4.2 Math23K

Experimental setup For Math23K, the data preprocessing and the configuration of `PolicyNet` are the same as MathQA. Nevertheless, we provided no pretraining data to `PolicyNet` in this experiment to make a fair comparison with weakly supervised learning baselines. This means that the formula search completely starts from scratch. In addition, we also used the formulas discovered in our formula search to train the GTS model (Xie and Sun, 2019) with its default settings to compare with existing work. For `ActTaker`, we adopted six `Number` modules, `N_0` to `N_5`, four `Operation` modules +, −, ×, and ÷, and two `Constant` modules, `C_1` and `C_100`.

	Accuracy
Supervised learning approaches	
DNS (Wang et al., 2017)	58.1%
GTS (Xie and Sun, 2019)	74.3%
G2TL (Zhang et al., 2020)	**75.5%**
Weakly supervised learning approaches	
Seq2seq + REINFORCE	12.1%
Seq2seq + MAPO (Hong et al., 2021)	10.7%
Seq2seq + LBF (Hong et al., 2021)	44.7%
Seq2seq + Ours w/o DBS	51.2%
Seq2seq + Ours w DBS	52.4%
GTS + REINFORCE	14.0%
GTS + MAPO (Hong et al., 2021)	20.8%
GTS + LBF (Hong et al., 2021)	59.4%
GTS + Ours w/o DBS	59.8%
GTS + Ours w DBS	**59.9%**

Table 4: The accuracy achieved on Math23K by fully supervised and weakly supervised learning methods under five-fold cross-validation.

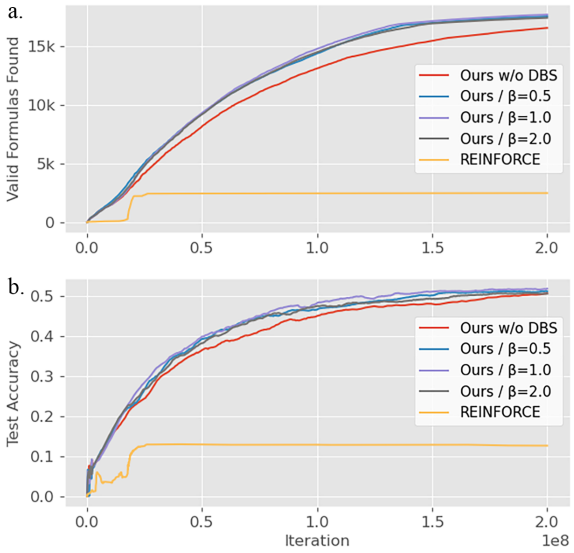

Figure 3: The learning processes of different weakly supervised learning methods measured by two metrics: a. the number of valid formulas found in the training set, b. the accuracy achieved on the test set.

Results Table 4 shows the accuracy achieved by both supervised learning methods and weakly supervised learning methods on Math23K. Although supervised learning methods remain the state-of-the-art approach on this task owing to the natural strength of utilizing ground-truth formula supervision, our learning framework outperformed all the existing weakly supervised approaches on both two inference models LSTM and GTS. In weakly supervised learning environment, learning engines are expected to acquire knowledge on formulas from scratch. Naive reinforcement learning methods represented by REINFORCE and MAPO (Liang et al., 2018) are inadequate in performing this job because their exploring capacity is mainly powered by the estimation made by the inference model, which can hardly deal with some complicated formula modification such as inserting an operator into a formula while keeping the rest of the formula intact. Compared with the learning-by-fixing (LBF) mechanism proposed by Hong et al. (2021), we attribute the superiority of our method to the capacity of managing formula exploration in a broader search space. The fixing mechanism of LBF mainly focuses on the 1-step fix, which assumes that only one symbol in the reasoning tree should be substituted. However, our heuristic formula exploration based on the formula graph and formula score can sample candidate formulas according to the observation on various formulas that are likely to be valid and thus organize a broader heuristic formula exploration.

Moreover, we analyzed the learning processes of different weakly supervised learning methods together with our method with different hyperparameter β in DBS. The result is shown in Figure 3. First of all, it can be noticed that REINFORCE only managed to discover a small number of formulas within the learning in the training set. This led to its poor performance on the test set. Compared with REINFORCE, our method showed a much more powerful formula exploring capacity and higher exploring efficiency. Furthermore, though the ablation study on our method, it is shown that DBS makes the learning process converge faster, discover more valid formulas in the training set, and achieve slightly higher accuracy on the test set. However, the difference brought by β is not quite obvious. From this result, we concluded that the existence of the difference-based score contributed to the heuristic search, but the search process is not very sensitive to its scale.

Further Discussion Although our proposed method has shown remarkable formula exploring capacity as a weakly supervised leaning approach, we are still alert to the gap of performance between our method and supervised learning methods. Generally, this gap can be ascribed to two causes. On the one hand, the size of the space of possible formulas, which is also the size of the search space, can be approximately up to 10^{20}. Such a search

space is so huge that some complex ground-truth formulas cannot be guaranteed to be found through a heuristic search by nature. On the other hand, through the search process, incorrect formulas may result in correct answers by accident. For example, for the question "Find the sum of 2 and 2.", a valid formula could be `N_0+N_1`, but formulas such as `N_0+N_0` and `N_0×N_1` also result in correct answers by accident. These errors can be reduced by providing multiple pairs of num and a to each q, which is also one of our motivations for proposing the template combining process in preprocessing. However, such errors still cannot be entirely avoided for templates exclusively owned by unique questions. Anyway, considering that these are some common challenges faced by all weakly supervised learning approaches, we leave the solution to these problems to future work.

5 Conclusion

This work discussed the issue of formula annotation dependence in existing work on solving mathematical problems. To deal with this issue, we proposed a new learning framework, Weakly Supervised Formula Learner. This framework established a mechanism to learn formulas with weak supervision from final answers and enabled a heuristic formula search in the space of possible formulas.

In the experiments, our learning framework showed remarkable formula exploring capacity on both MathQA and Math23K datasets. Particularly, on MathQA, we illustrated that our models trained with formulas discovered in formula exploration outperformed baselines trained with complete yet imperfect formula annotations. On Math23K, our learning framework showed more powerful formula exploring capacity than existing weakly supervised learning methods. In view of this evidence, we consider our proposed learning framework a valid and advanced approach for solving mathematical problems with weak supervision from their answers.

Acknowledgement

This work was supported by Institute of AI and Beyond of the University of Tokyo, JSPS KAKENHI Grant Number JP19H04166, and the commissioned research (No. 225) by National Institute of Information and Communications Technology (NICT), Japan.

References

Aida Amini, Saadia Gabriel, Shanchuan Lin, Rik Koncel-Kedziorski, Yejin Choi, and Hannaneh Hajishirzi. 2019. MathQA: Towards interpretable math word problem solving with operation-based formalisms. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 2357–2367.

Jacob Andreas, Marcus Rohrbach, Trevor Darrell, and Dan Klein. 2016a. Learning to compose neural networks for question answering. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 1545–1554.

Jacob Andreas, Marcus Rohrbach, Trevor Darrell, and Dan Klein. 2016b. Neural module networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 39–48.

Kezhen Chen, Qiuyuan Huang, Hamid Palangi, Paul Smolensky, Kenneth D Forbus, and Jianfeng Gao. 2020a. Mapping natural-language problems to formal-language solutions using structured neural representations. In *International Conference on Machine Learning*, pages 1566–1575.

Xinyun Chen, Chen Liang, Adams Wei Yu, Denny Zhou, Dawn Song, and Quoc V Le. 2020b. Neural symbolic reader: Scalable integration of distributed and symbolic representations for reading comprehension. In *International Conference on Learning Representations*.

Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long short-term memory. *Neural computation*, 9(8):1735–1780.

Yining Hong, Qing Li, Daniel Ciao, Siyuan Huang, and Song-Chun Zhu. 2021. Learning by fixing: Solving math word problems with weak supervision. *Proceedings of the AAAI Conference on Artificial Intelligence*, 35(6):4959–4967.

Mark Hopkins, Cristian Petrescu-Prahova, Roie Levin, Ronan Le Bras, Alvaro Herrasti, and Vidur Joshi. 2017. Beyond sentential semantic parsing: Tackling the math SAT with a cascade of tree transducers. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 795–804.

Justin Johnson, Bharath Hariharan, Laurens Van Der Maaten, Judy Hoffman, Li Fei-Fei, C Lawrence Zitnick, and Ross Girshick. 2017. Inferring and executing programs for visual reasoning. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 2989–2998.

Aishwarya Kamath and Rajarshi Das. 2019. A survey on semantic parsing. In *Automated Knowledge Base Construction (AKBC)*.

Diederik P. Kingma and Jimmy Ba. 2015. Adam: A method for stochastic optimization. In *International Conference on Learning Representations*.

Rik Koncel-Kedziorski, Hannaneh Hajishirzi, Ashish Sabharwal, Oren Etzioni, and Siena Dumas Ang. 2015. Parsing algebraic word problems into equations. *Transactions of the Association for Computational Linguistics*, 3:585–597.

Chen Liang, Mohammad Norouzi, Jonathan Berant, Quoc Le, and Ni Lao. 2018. Memory augmented policy optimization for program synthesis and semantic parsing. In *Proceedings of the 32nd International Conference on Neural Information Processing Systems*, pages 10015–10027.

Wang Ling, Dani Yogatama, Chris Dyer, and Phil Blunsom. 2017. Program induction by rationale generation: Learning to solve and explain algebraic word problems. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 158–167.

Brenden K Petersen, Mikel Landajuela Larma, Terrell N. Mundhenk, Claudio Prata Santiago, Soo Kyung Kim, and Joanne Taery Kim. 2021. Deep symbolic regression: Recovering mathematical expressions from data via risk-seeking policy gradients. In *International Conference on Learning Representations*.

David Saxton, Edward Grefenstette, Felix Hill, and Pushmeet Kohli. 2019. Analysing mathematical reasoning abilities of neural models. In *International Conference on Learning Representations*.

Mike Schuster and Kuldip K Paliwal. 1997. Bidirectional recurrent neural networks. *IEEE transactions on Signal Processing*, 45(11):2673–2681.

Shuming Shi, Yuehui Wang, Chin-Yew Lin, Xiaojiang Liu, and Yong Rui. 2015. Automatically solving number word problems by semantic parsing and reasoning. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 1132–1142.

Lei Wang, Yan Wang, Deng Cai, Dongxiang Zhang, and Xiaojiang Liu. 2018. Translating a math word problem to a expression tree. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 1064–1069.

Yan Wang, Xiaojiang Liu, and Shuming Shi. 2017. Deep neural solver for math word problems. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 845–854.

Ronald J Williams. 1992. Simple statistical gradient-following algorithms for connectionist reinforcement learning. *Machine learning*, 8(3-4):229–256.

Yuxuan Wu and Hideki Nakayama. 2020. Graph-based heuristic search for module selection procedure in neural module network. In *Proceedings of the Asian Conference on Computer Vision (ACCV)*.

Zhipeng Xie and Shichao Sun. 2019. A goal-driven tree-structured neural model for math word problems. In *IJCAI*, pages 5299–5305.

Kexin Yi, Jiajun Wu, Chuang Gan, Antonio Torralba, Pushmeet Kohli, and Joshua B. Tenenbaum. 2018. Neural-symbolic vqa: Disentangling reasoning from vision and language understanding. In *Proceedings of the 32nd International Conference on Neural Information Processing Systems*, pages 1039–1050.

Jipeng Zhang, Lei Wang, Roy Ka-Wei Lee, Yi Bin, Yan Wang, Jie Shao, and Ee-Peng Lim. 2020. Graph-to-tree learning for solving math word problems. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 3928–3937.

Reducing Spurious Correlations for Answer Selection by Feature Decorrelation and Language Debiasing

Zeyi Zhong [1,2], Min Yang [1]*, Ruifeng Xu [3]
[1] Shenzhen Institute of Advanced Technology, Chinese Academy of Sciences
[2] University of Science and Technology of China
[3] Harbin Institute of Technology (Shenzhen)
{zhongzy,min.yang}@siat.ac.cn, xuruifeng@hit.edu.cn

Abstract

Deep neural models have become the mainstream in answer selection, yielding state-of-the-art performance. However, these models tend to rely on spurious correlations between prediction labels and input features, which in general suffer from robustness and generalization. In this paper, we propose a novel **S**purious **C**orrelation reduction method to improve the robustness of the neural **AN**swer selection models (SCAN) from the sample and feature perspectives by removing the feature dependencies and language biases in answer selection. First, from the sample perspective, we propose a feature decorrelation module by learning a weight for each instance at the training phase to remove the feature dependencies and reduce the spurious correlations without prior knowledge of such correlations. Second, from the feature perspective, we propose a feature debiasing module with contrastive learning to alleviate the negative language biases (spurious correlations) and further improve the robustness of the AS models. Experimental results on three benchmark datasets show that SCAN achieves substantial improvements over strong baselines. For reproducibility, we will release our code and data at https://github.com/xish9/SCAN.

1 Introduction

Answer selection, which aims to select the most applicable answers from an answer candidate pool, has broad applications in information retrieval (IR) and natural language processing (NLP). Conventional answer selection methods primarily focus on designing various features, such as syntactic features (Li, 2003), dependency trees (Wang et al., 2007), and translation features (Surdeanu et al., 2008). However, the remarkable success of these methods relies heavily on feature engineering, which is a labor-intensive and time-consuming process.

Subsequently, deep neural models (Qiu and Huang, 2015; Guo et al., 2017; Tay et al., 2017; Zhou et al., 2018) have been widely employed for answer selection and become the mainstream techniques for answer selection by automatically learning the contextual representations of questions and answers. To capture the relationships between the question-answer pairs, different attention mechanisms (Zhang et al., 2017; Tay et al., 2018a; Shen et al., 2018; Yang et al., 2019a; Xie et al., 2020) have been proposed to learn the interactive features of the questions and the answers. Recently, pre-trained language models, such as BERT (Devlin et al., 2018) and RoBERTa (Liu et al., 2019), have been proposed and applied to answer selection (Garg et al., 2020; MacAvaney et al., 2020; Zhang et al., 2021a), obtaining the state-of-the-art results.

Despite the remarkable progress of previous works, these deep neural models are prone to rely on spurious correlations between input features and prediction labels, which capture the prediction correlations that hold for most training samples but do not hold in general. The spurious correlations limit the robustness and generalization ability of the neural AS models to the out-of-distribution and challenging datasets. In particular, for answer selection, the word-overlap between the question and the answers is highly correlated with the relevance prediction label. Thus, the deep AS models perform poorly on the out-of-distribution or challenging corpora that cannot be tackled with these superficial correlations (e.g., word overlap). This issue is also referred to as dataset bias (Clark et al., 2019) and data distribution shift (Sagawa et al., 2020).

In this paper, we propose a novel **S**purious **C**orrelation reduction method to improve the robustness of the neural **AN**swer selection models (SCAN) from the sample and feature perspectives by removing the feature dependencies and language biases in answer selection. First, from the sample perspective, we employ the feature decorre-

*Min Yang is corresponding author.

Proceedings of the 29th International Conference on Computational Linguistics, pages 1753–1764
October 12–17, 2022.

lation module based on Random Fourier Features (Rahimi and Recht, 2007) to decorrelate the relevant and irrelevant features by learning a weight for each sample during the training phase, which facilitate the deep AS models to reduce spurious correlations and concentrate on the true discriminative features (relevant features) for label prediction. Second, from the feature perspective, we propose a feature debiasing module with contrastive learning to weaken the negative biases in language and improve the robustness of the AS models. Concretely, the feature debiasing module aims to make the base contextual representation of input sample close to the debiased features and away from the negative bias features.

Our main contributions are three-fold:

- We propose a feature decorrelation module by learning a weight for each training instance to remove the feature dependencies and reduce the spurious correlations without prior knowledge of such correlations.
- We propose a feature debiasing module with contrastive learning to alleviate the negative language biases (spurious correlations) and improve the robustness of the AS models.
- Experimental results show that our SCAN method achieves substantial improvements over the state-of-the-art baseline methods for answer selection.

2 Related Work

2.1 Deep Learning for Answer Selection

Answer selection has received remarkable attention in various tasks, such as dialogue systems (Yuan et al., 2019; He et al., 2022b,a), knowledge base question answering (Niu et al., 2021; Saxena et al., 2020), and information retrieval (Li et al., 2021). So far, deep learning approaches have become the mainstream in answer selection (AS) due to their impressive improvement. Severyn and Moschitti (2015) was an early representative neural AS model, which utilized convolutional neural network (CNN) to learn question and answer representations separately followed by a similarity function to compute the relevance score. Tay et al. (2017) extended the long short-term memory (LSTM) network with holographic composition for sentence modeling and semantic matching. Several works (Yin et al., 2016; Tan et al., 2016) explored different attention mechanisms to capture the relations between sentences. For example, Tay et al. (2018c) proposed a casted attention for feature augmentation to improve the representation learning process. Shen et al. (2017b) proposed an inter-weighted alignment network, which utilized the word-level similarity matrix to explore the fine-grained alignment of two sentences. Tay et al. (2018b) presented HyperQA which leveraged a parameter efficient network to model the relations between the question and answer representations with PLMs in the Hyperbolic space instead of the Euclidean space.

Recently, the pre-trained language models (PLMs), such as BERT (Devlin et al., 2018) and RoBERTa (Liu et al., 2019), have been applied to the answer selection task and yielded state-of-the-art results by capturing rich linguistic knowledge from large textual corpora. Yoon et al. (2019) employed ELMo (Peters et al., 2018) to a compare aggregate architecture, which leveraged the latent-cluster information to enhance the AS model. Lai et al. (2019) combined a gated self-attention memory network and the pre-trained language models for answer selection. Garg et al. (2020) proposed a two-step transfer-adapt (TANDA) method, which fine-tuned the pre-trained language models by using a large QA dataset ASNQ. Recently, Zhang et al. (2021b, 2022) focused on exploiting the interrelated information between candidate answers and obtained the best results for answer selection.

2.2 Spurious Correlation Reduction in NLP

Despite the remarkable progress made by deep neural networks, some studies (Gururangan et al., 2018; McCoy et al., 2019; Zhang et al., 2021a) have revealed that the deep models often relied on spurious correlations between the learned features and the prediction labels, making the deep models unstable and not generalize well to the data with different distributions. For example, previous studies (Gururangan et al., 2018; McCoy et al., 2019) observed that specific linguistic phenomena or syntactic heuristics are highly correlated with certain inference classes in natural language inference (NLI). Jia and Liang (2017) revealed that the question-answering (QA) models trained on SQuAD were not robust to perturbations with modified semantics since the QA models cannot possess the true text understanding. Recently, there are several efforts made to reduce the spurious correlations

by removing the data bias explicitly. For example, CoQA (Reddy et al., 2019) limited the question annotation process by avoiding using exact words in the passage. SWAG (Zellers et al., 2018) utilized an adversarial filter methodology to construct the debiased dataset. In addition, several studies focused on recognizing these spurious correlations and then removing them implicitly. Clark et al. (2019) proposed a two-stage training procedure, which built a bias-only strategy to train a robust model through the ensembling approach. Sagawa et al. (2020) coupled the distributionally robust optimization with regularization to improve the worst group generalization. To the best of our knowledge, we are the first to reduce the spurious correlations for answer selection, leveraging feature decorrelation and language debiasing.

3 Methodology

We assume there are N instances (question-answer pairs) in the training set. Given a question q_i and a set of K candidate answers $A_i = \{a_1, a_2, ..., a_K\}$, the answer selection task aims to find the best answer by ranking the candidate answers based on their relevance to the given question. Benefiting from the pairwise ranking, we can reformalize the answer selection as a classification problem by predicting the relevance label y_i of each question-answer pair (q_i, a_i). We represent each question q_i and answer a_i as $q = [w_1^{q_i}, \ldots, w_n^{q_i}]$ and $a_i = [w_1^{a_i}, \ldots, w_m^{a_i}]$, where n and m are the lengths of question q_i and answer a_i, respectively.

In this paper, we propose a novel SCAN method for answer selection. As illustrated in Figure 1, the proposed SCAN consists of two primary modules: the feature decorrelation module based on sample weighting and the feature debiasing module based on contrastive learning (Chuang et al., 2020; Liu et al., 2021). Next, we will introduce the base context encoder and two key spurious correlation reduction components in detail.

3.1 Base Context Encoder

Inspired by the remarkable success of pre-trained language models (PLMs) on most NLP tasks, we employ RoBERTa (Liu et al., 2019) as our base context encoder to obtain the contextual representations of each question-answer pair.

We take the concatenation of the question q_i and each candidate answer a_i as input, and use RoBERTa to generate the contextual representation of the i-th question-answer pair as:

$$\mathbf{E}_i = \text{RoBERTa}([\text{cls}, q_i, \text{sep}, \text{sep}, a_i, \text{sep}]) \quad (1)$$

where $\mathbf{E}_i \in \mathbb{R}^{(n+m+4)\times d_h}$ denotes the hidden states of the question-answer pair (q_i, a_i) and d_h is the dimension of each hidden state. The special tokens [cls] and [sep] represent the classification token and the separation token respectively. We denote the hidden vector of the special [cls] token as $\mathbf{H}_i \in \mathbb{R}^{d_h}$, which can be treated as the base contextual representation of the question-answer pair (q_i, a_i) for prediction.

3.2 Feature Decorrelation with Sample Weighting from Sample Perspective

Spurious correlations are very common in deep models, especially when the answer selection model is overparameterized. Spurious correlations could hurt the stability and generality of the model when deployed in practice. In this paper, we employ the feature decorrelation method with sample weighting to decorrelate the relevant and irrelevant features, and make the model focus on discriminative features (relevant features) that are truly related to the label prediction.

Given the training data with N question-answer pairs, the representations learned by the base context encoder can be denoted as $\mathbf{H} \in \mathbb{R}^{N\times d_h}$. We input the representation $\mathbf{H}$ into the feature decorrelation module based on Random Fourier Features (Rahimi and Recht, 2007), which learns a weight for each instance such that features are decorrelated on the weighted training data.

Formally, we use $\mathbf{w} \in \mathbb{R}^N$ to denote the local weights of individual samples, which are initialized with all-ones vector at the beginning of each training iteration. During the optimization process with stochastic gradient descent (SGD), there are merely part of samples being observed in each batch, while the global weights of all samples would be ignored. Thus, we leverage global weights $\mathbf{w}^G \in \mathbb{R}^N$ and global features $\mathbf{H}^G \in \mathbb{R}^{N\times d_h}$ to exploit the global information of the training data. By concatenating the global and local information, we can obtain the combined features $\mathbf{H}^{\text{com}}$ and weights $\mathbf{w}^{\text{com}}$ as:

$$\begin{aligned} \mathbf{H}^{\text{com}} &= \text{Concat}(\mathbf{H}^G, \mathbf{H}) \\ \mathbf{w}^{\text{com}} &= \text{Concat}(\mathbf{w}^G, \mathbf{w}) \end{aligned} \quad (2)$$

We denote the combined features of the i-th sample as $\mathbf{H}_i^{\text{com}}$. The j-th feature in the combined representation space is denoted as $\mathbf{H}_{:,j}^{\text{com}}$.

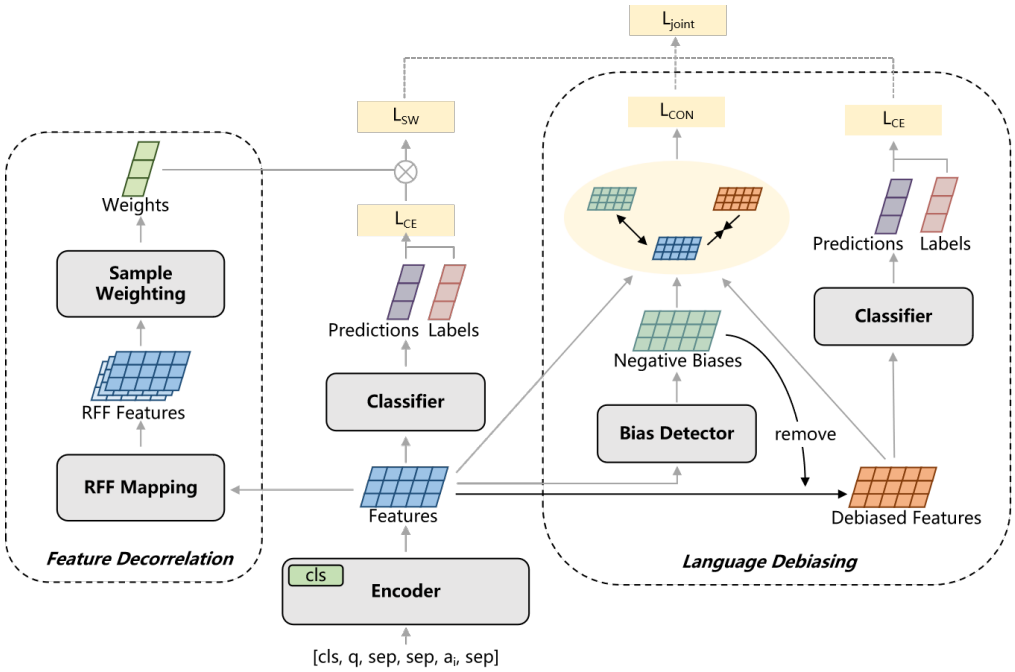

Figure 1: The overall architecture of our SCAN method, which reduces the spurious correlations with a feature decorrelation module module and a feature debiasing module.

To eliminate the correlation between features, we measure their independence via Hilbert-Schmidt Independence Criterion (HSIC) which is a kernel statistical test of independence (Gretton et al., 2007), inspired by (Zhang et al., 2021a). To reduce the computational complexity, we approximate the test statistical independence by Frobenius norm. In particular, we sample r Random Fourier Features(RFF) mapping functions from the function sapce $\mathcal{G}$ respectively, and then convert the combined representations $\mathbf{H}^{\text{com}}$ into $\widetilde{\mathbf{H}} \in \mathbb{R}^{N \times d_h \times r}$:

$$\widetilde{\mathbf{H}}_{i,j} = \left(g_1(\mathbf{H}^{\text{com}}_{i,j}), \ldots, g_r(\mathbf{H}^{\text{com}}_{i,j})\right) \quad (3)$$

$$\text{where} \quad g_k(\mathbf{H}_{i,j}) \in \mathcal{G}, \forall k, \quad (4)$$

$$\mathcal{G} = \{g : x \rightarrow \sqrt{2}\cos(\omega x + \phi) | \quad (5)$$
$$\omega \sim \mathcal{N}(0,1), \phi \sim \text{Uniform}(0, 2\pi)\}$$

where ω is sampled from the standard Normal distribution and ϕ is sampled from the Uniform distribution to approximate continuous shift-invariant kernels*. With the sample weights $\mathbf{w}^{\text{com}}$, we can calculate the weighted partial cross-covariance matrix $\hat{\sum}_{j_1,j_2}$ of two features $\mathbf{H}^{\text{com}}_{:,j_1}$ and $\mathbf{H}^{\text{com}}_{:,j_2}$ by:

$$\hat{\sum}_{j_1,j_2} = \frac{1}{N-1} \cdot \sum_{i=1}^{N} \left[\left(w_i^{\text{com}} \widetilde{\mathbf{H}}_{i,j_1} - \mathbb{E}(\widetilde{\mathbf{H}}_{:,j_1})\right)^T \cdot \left(w_i^{\text{com}} \widetilde{\mathbf{H}}_{i,j_2} - \mathbb{E}(\widetilde{\mathbf{H}}_{:,j_2})\right)\right] \quad (6)$$

$$\text{where} \quad \mathbb{E}(\widetilde{\mathbf{H}}_{:,j}) = \frac{1}{N} \sum_{i=1}^{N} w_i^{\text{com}} \widetilde{\mathbf{H}}_{i,j} \quad (7)$$

where w_i^{com} is the weight of the i-th question-answer pair (q_i, a_i).

We use the squared Frobenius norm of the partial cross-covariance matrix to estimate the independence between any pair of features. Thus, we optimize the sample weights $\mathbf{w}^{\text{com}}$ by minimizing the squared Frobenius norm between any pair of features, which can be defined as follows:

$$\mathbf{w}^{\text{com}*} = \underset{\mathbf{w}^{\text{com}}}{\text{argmin}} \sum_{1 \le j_1 < j_2 \le d_h} \left\| \hat{\sum}_{j_1,j_2} \right\|_F^2 \quad (8)$$

During the procedure of learning the weights $\mathbf{w}^{\text{com}}$, we keep the model parameters fixed.

With the learned weights $\mathbf{w}^{\text{com}}$ of question-answer pairs during the training phase, we can optimize the model parameters by minimizing the weighted cross-entropy loss function as:

$$\mathcal{L}_{\text{SW}} = -\sum_{i=1}^{N} w_i^{\text{com}} \mathbf{y}_i \log \hat{\mathbf{y}}_i \quad (9)$$

*Similar to (Zhang et al., 2021a), we adopt both sin and cosine functions to learn better features.

where $\mathbf{y}_i$ denotes the one-hot vector of the ground-truth relevance label y_i of the i-th question-answer pair (q_i, a_i). $\hat{\mathbf{y}}_i$ is the predicted relevance label of (q_i, a_i), which is defined as:

$$\hat{\mathbf{y}}_i = softmax(\mathbf{H}_i) \tag{10}$$

During the procedure of updating the model parameters via back propagation, we keep the weights of training samples fixed.

Note that for efficient optimization the weights of training samples and the model parameters are learned iteratively, and the training procedure is repeated until convergence. At the end of each training iteration, we update the global features $\mathbf{H}^G$ and the corresponding weights $\mathbf{w}^G$ as:

$$\begin{aligned} \mathbf{w}^G &= \alpha \mathbf{w}^G + (1-\alpha)\mathbf{w} \\ \mathbf{H}^G &= \alpha \mathbf{H}^G + (1-\alpha)\mathbf{H} \end{aligned} \tag{11}$$

where α denotes the hyperparameter for controlling the impact of global information.

3.3 Language Debiasing with Contrastive Learning from Feature Perspective

Most previous AS models frequently follow the superficial correlations (i.e., language bias) induced by the training data, which is another kind of the spurious correlation. The language biases makes the neural AS models brittle to linguistic variations in questions/answers. However, not all the language biases are harmful in answer selection, and some language biases may contain commonsense knowledge that is beneficial for answer selection. For example, when a question begins with "*When*", the corresponding answer should contain words that indicate time or period. In this section, we propose a feature debiasing module with contrastive learning, which weakens the negative biases in language and improves the robustness of AS models.

First, we utilize a bias detection method to recognize the negative biases that existed in the base contextual representation $\mathbf{H}_i$ of the i-th question-answer pair learned by the base context encoder. The detection function $\sigma(\cdot)$ consists of dense layer followed by a sigmoid activation function. Formally, we learn the bias weight vector $\mathbf{b}_i$ as follows:

$$\mathbf{b}_i = \sigma(\mathbf{H}_i^{\text{trans}}), \text{ where } \mathbf{H}_i^{\text{trans}} = \rho_b(\mathbf{H}_i) \tag{12}$$

where ρ_b denotes a multi-layer perceptron (MLP). $\mathbf{H}_i^{\text{trans}}$ represents the transformed feature containing language biases.

Second, we can learn the negative bias representation $\mathbf{H}_i^{\text{bias}}$ based on the base contextual representation $\mathbf{H}_i$ by the product of the bias weight vector $\mathbf{b}_i$ and transformed feature $\mathbf{H}_i^{\text{trans}}$ as:

$$\mathbf{H}_i^{\text{bias}} = \mathbf{b}_i \cdot \mathbf{H}_i^{\text{trans}} \tag{13}$$

Then, we learn the debiased representation $\mathbf{H}_i^d \in \mathbb{R}^{d_h}$ by removing the negative bias representation $\mathbf{H}_i^{\text{bias}}$ from the original base contextual representation $\mathbf{H}_i$. We compute the debiased representation $\mathbf{H}_i^{\text{debias}}$ as follows:

$$\mathbf{H}_i^{\text{d}} = \rho_d(\mathbf{H}_i - \mathbf{H}_i^{\text{bias}}) \tag{14}$$

where ρ_d denotes another MLP layer.

Cross-Entropy Loss The learned debiased representation $\mathbf{H}_i^{\text{d}}$ of the i-th question-answer pair is fed into a classifier with a softmax layer as:

$$\hat{\mathbf{y}}_i = softmax(\mathbf{H}_i^{\text{d}}) \tag{15}$$

where $\hat{\mathbf{y}}_i$ represents the predicted relevance label of (q_i, a_i). We can optimize the answer selection model by minimizing the cross-entropy loss as:

$$\mathcal{L}_{\text{CE}} = -\sum_{i=1}^{N} \mathbf{y}_i \log \hat{\mathbf{y}}_i \tag{16}$$

where $\mathbf{y}_i$ denotes the one-hot vector of the ground-truth relevance label y_i of the question-answer pair (q_i, a_i). N is the number of training instances.

Contrastive Loss To avoid using additional parameters in inference phase, we attempt to discard the language debiasing module in for inference and make the base context features $\mathbf{H}$ and debiased features $\mathbf{H}^{\text{d}}$ learned by the language debaising module as similar as possible. In this paper, we leverage the contrastive learning to learn robust representations by incorporating instance-level semantic discriminativeness into the representation learning. Concretely, we leverage a contrastive loss function $\mathcal{L}_{\text{CL}}$ to make each base context representation $\mathbf{H}_i$ close to the corresponding debiased feature $\mathbf{H}_i^{\text{d}}$ and away from the negative bias feature $\mathbf{H}_i^{\text{bias}}$. Formally, we define the contrastive loss as:

$$\mathcal{L}_{\text{CL}} = -\frac{1}{N}\sum_{i=1}^{N} \log \frac{\mu(\mathbf{H}_i, \mathbf{H}_i^{\text{d}})}{\mu(\mathbf{H}_i, \mathbf{H}_i^{\text{d}}) + \mu(\mathbf{H}_i, \mathbf{H}_i^{\text{bias}})} \tag{17}$$

$$\mu(\mathbf{H}_i, \mathbf{H}_i^{\text{bias}}) = \exp(sim(\mathbf{H}_i, \mathbf{H}_i^{\text{bias}})/\tau) \tag{18}$$

where $sim()$ denotes a cosine similarity function. τ is a temperature value.

3.4 Joint Training Objective

Overall, our method consists of three training objectives, including the sample weighting loss $\mathcal{L}_{\text{SW}}$, cross-entropy loss $\mathcal{L}_{\text{CE}}$, and the contrastive loss $\mathcal{L}_{\text{CL}}$. We minimize the joint loss function $\mathcal{L}_{\text{joint}}$ by summing up the three training objectives as:

$$\mathcal{L}_{\text{joint}} = \mathcal{L}_{\text{SW}} + \mathcal{L}_{\text{CE}} + \mathcal{L}_{\text{CL}} \quad (19)$$

Although the sample weights are optimized according to Eq. (8) during the training process, we do not include the weight optimization function during the overall training objective for optimizing the model parameters.

3.5 Inference Stage

In the inference phase, given the back propagation is disabled, we escape the sample weighting phase without any calculation of sample weights and discard the language debiasing phase without introducing additional parameters. Instead, we conduct the prediction directly via Eq. (10) by merely leveraging the optimized base RoBERTa encoder.

4 Experimental Setup

4.1 Datasets

To evaluate the effectiveness of our method, we conduct comprehensive experiments on three publicly available corpora. The statistics of the three datasets are shown in Table 1.

WikiQA The WikiQA dataset (Yang et al., 2015) is an open-domain question answering dataset. The original WikiQA contains 3047 questions originally sampled from Bing query logs and 29258 answer sentences from Wikipedia. We denote the questions that have no correct answer sentences as "All-" and the questions that have only correct answer sentences as "*All+*". The remaining data set without both "*All-*" and "*All+*" questions is denoted as "*Clean*". Following the previous works (Garg et al., 2020), we train the AS models on the no "*All-*" questions, and then test the models on the "*Clean*" questions. The statistics of WikiQA are shown in Table 1.

SelQA The SelQA (Jurczyk et al., 2016) dataset is similar to WikiQA but covers more diverse topics drawn from Wikipedia. It consists of a larger number of questions, which is about 6 times larger than WikiQA. We adopt the original data split as in (Jurczyk et al., 2016) to verify the AS models. The statistics of SelQA are shown in Table 1.

Dataset		Train	Dev	Test
WikiQA	#Q	873	122	237
	#A	8672	1126	2341
SelQA	#Q	5529	785	1590
	#A	66438	9377	19435
ANTIQUE	#Q	2226	200	200
	#A	25229	2193	6589

Table 1: Statistics of the three experimental datasets.

ANTIQUE The ANTIQUE dataset (Hashemi et al., 2020) is an open-domain non-factoid QA dataset collected from a community question answering service, Yahoo!Answers. Different from WikiQA and SelQA, ANTIQUE has four-level relevance labels between 1 to 4. Following previous work (MacAvaney et al., 2020), we regard scores 3 and 4 as relevant, while scores 1 and 2 are treated as irrelevant. Since the original dataset has no validation set, we choose 200 questions from the training set as a held-out set for validation, similar to MacAvaney et al. (2020). The statistics of ANTIQUE are shown in Table 1.

4.2 Baselines

For WikiQA and SelQA which are widely used in previous works, we compare the proposed SCAN with several advanced baselines, including CNN-DAN (Santos et al., 2017), CNN-hinge (Santos et al., 2017), ACNN (Shen et al., 2017a), AdaQA (Shen et al., 2017a), HyperQA (Tay et al., 2018b), DRCN (Kim et al., 2019), RE2 (Yang et al., 2019b), a compare aggregate model (Comp-Agg) (Yoon et al., 2019), BERT (Devlin et al., 2018), RoBERTa (Liu et al., 2019), TANDA (Garg et al., 2020), answer support-based reranker (ASR) (Zhang et al., 2021b), DAR and DAR-DPR Zhang et al. (2022). For ANTIQUE, we compare SCAN with four benchmark baselines including aNMM (Yang et al., 2016), BERT (Devlin et al., 2018), RoBERTa (Liu et al., 2019), a curriculum learning method with BERT (BERT-CL) (MacAvaney et al., 2020), a bilateral generation method (BERT-BiG) (Deng et al., 2021), and TANDA (Garg et al., 2020).

4.3 Implementation Details

We adopt the RoBERTa-base (Liu et al., 2019) that is pre-trained on large-scale English corpus and fine-tuned on ASNQ corpus (Garg et al., 2020) as the sentence encoder. In the experiments, we apply the grid search algorithm (Huang et al., 2012)

on the validation set to tune the hyper-parameters. Concretely, we set the maximum sequence length to 128. The training batch size is set to 140. The dimension of hidden state (i.e., d_h) is set to 768. We adopt the Adam optimizer to optimize the whole SCAN method. The temperature value τ, learning rate and α for decorrelation module are 1/0.3/0.5, 1e-4/1e-4/1e-3 and 0.7/0.9/0.9 for WikiQA, SelQA and ANTIQUE, respectively. For reproducibility, we will release our code and data upon the publication of this paper.

4.4 Evaluation Metrics

For WikiQA and SelQA, we measure our method on test set with three official metrics: Mean Average Precision (MAP), Mean Reciprocal Rank (MRR), Precision at 1 of ranked candidates (P@1). For ANTIQUE, we also measure the MAP, MRR and P@1. In addition, we compute Normalized Discounted Cumulative Gain (i.e., nDCG@1, nDCG@3 and nDCG@10) with the original four-level relevance labels.

5 Experimental results

5.1 Overall Performance

Tables 2-4 summarize the experimental results on WikiQA, SelQA and ANTIQUE, respectively. SCAN performs significantly and consistently better than the compared baselines on all the three datasets, verifying the effectiveness of our SCAN method. From Table 2 and Table 3, we can observe that the CNN- or RNN-based methods perform poorly because they do not take advantage of the pre-trained language models (PLMs). TANDA outperforms BERT and CNN-based methods by adopting the RoBERTa-base that is pre-trained on large-scale general corpus and fine-tuned on ASNQ corpus. ASR, DAR and DAR-DRP, which are based on TANDA, improve the performance of TANDA by exploiting the interrelated information between the target answer and the other candidate answers. SCAN takes a further step towards reducing the spurious correlations for answer selection by feature decorrelation and language debiasing.

Table 4 reports the experimental results on the ANTIQUE dataset, demonstrating that the proposed SCAN method is also effective on the non-factoid QA. Specifically, SCAN exceeds the TANDA model (the base model of SCAN) by 3.97% on MRR and 6% on P@1. This verifies that it is necessary to remove the spurious correlations between the text representations and the prediction relevance labels.

Method	MAP	MRR	P@1
HyperQA ♮	0.7120	0.7270	-
RE2 ♯	0.7452	0.7618	-
Comp-Agg †	0.7640	0.7840	-
Comp-Agg (QNLI) †	0.8340	0.8480	-
BERT †	0.8130	0.8280	-
RoBERTa	0.8441	0.8551	0.7553
TANDA †	0.8890	0.9010	-
TANDA-re ‡	0.8860	0.8983	0.8189
ASR ‡	0.9014	0.9123	0.8436
DAR ‡	0.9011	0.9136	0.8519
DAR-DRP ‡	0.9051	0.9164	0.8560
SCAN	**0.9164**	**0.9281**	**0.8776**

Table 2: Experimental Results on WikiQA. The results with ♮ are retrieved from (Tay et al., 2018b), with ♯ are retrieved from (Yang et al., 2019b), with † are retrieved from (Garg et al., 2020), with ‡ are retrieved from (Zhang et al., 2022). TANDA-re denotes a reimplementation of TANDA. The best scores are in bold.

Method	MAP	MRR	P@1
CNN-DAN ♮	0.8660	0.8730	-
CNN-hinge ♮	0.8760	0.8810	-
ACNN ♮	0.8740	0.8800	-
AdaQA ♮	0.8910	0.8980	-
DRCN ♮	0.9250	0.9300	-
TANDA-re ‡	0.9512	0.9587	0.9302
ASR ‡	0.9519	0.9592	0.9314
DAR ‡	0.9592	0.9653	0.9415
SCAN	**0.9641**	**0.9701**	**0.9484**

Table 3: Experimental Results on SelQA. The results with ♮ are retrieved from (Kim et al., 2019), with ‡ are retrieved from (Zhang et al., 2022).

5.2 Ablation study

To verify the effectiveness of feature decorrelation and language debiasing in SCAN, we perform ablation test of SCAN on two types of QA corpora (WikiQA and ANTIQUE) in terms of removing the feature decorrelation module (denoted as w/o FD) and language debiasing (denoted as w/o LD), respectively. In particular, for the w/o FD model, the weighted cross-entropy loss is replaced with a normal cross-entropy loss without considering sample weights. We also report the results of removing both feature decorrelation and language debiasing (w/o FD+LD).

The ablation test results are reported in Table 5. Generally, both feature decorrelation and language

Method	MAP	MRR	P@1	nDCG@1	nDCG@3	nDCG@10
aNMM ♮	0.2563	0.6250	0.4847	0.5289	0.5127	0.4904
BERT ♮	0.3771	0.7968	0.7092	0.7126	0.6570	0.6423
RoBERTa	0.6137	0.7763	0.6550	0.6683	0.6525	0.6765
BERT-CL ♯	-	0.7335	0.6450	-	-	-
BERT-BiG †	-	0.8470	0.7650	0.7500	0.7100	0.7200
TANDA	0.6511	0.8258	0.7250	0.7167	0.6969	0.7091
SCAN	**0.6722**	**0.8637**	**0.7850**	**0.7550**	**0.7186**	**0.7297**

Table 4: Experimental Results on ANTIQUE. The results with ♮ are retrieved from (Hashemi et al., 2020), with ♯ are retrieved from (MacAvaney et al., 2020), with † are retrieved from (Deng et al., 2021).

Method	WikiQA			ANTIQUE		
	MAP	MRR	P@1	MAP	MRR	P@1
SCAN	0.9164	0.9281	0.8776	0.6722	0.8637	0.7850
w/o FD	0.9004	0.9148	0.8523	0.6667	0.8399	0.7400
w/o LD	0.9011	0.9144	0.8523	0.6603	0.8424	0.7500
w/o FD+LD	0.8943	0.9063	0.8354	0.6511	0.8258	0.7250

Table 5: Experimental Results of the ablation study on WikiQA and ANTIQUE.

debiasing contribute noticeable improvement to the proposed SCAN method. Concretely, the performances decrease sharply, especially in terms of MRR and $P@1$, when removing either the FD model or the LD module. This is within our expectation since both feature decorrelation and language debiasing can reduce the spurious correlations for answer selection.

5.3 Robustness to Noise and Perturbation

To further analyze the robustness of our method, we conduct experiments on the WikiQA dataset with injected noise and adversarial perturbations. Following previous work (Gokhale et al., 2022), we create adversarial samples by adding character-level perturbations such as swapping, inserting or deleting characters to 30% of samples. In addition, similar to (Garg et al., 2020), we inject noise into the training samples in WikiQA by randomly sampling 20% of question-answer pairs from the training set and switching their labels. The experimental results are shown in Table 6. SCAN achieves consistently better performance than TANDA on these settings, verifying the robustness of our method to noise and adversarial perturbations.

5.4 Case Study

We use a representative exemplary case that is selected from the WikiQA test set to further investigate the effectiveness of SCAN. This chosen question is incorrectly predicted by TANDA while being correctly predicted by SCAN. From Table 7, We observe that TANDA simply picks up the answer that contains the matching words *cricket wireless* without understanding the deep semantics. On the contrary, SCAN obtains the correct answer since it can recognize the real intention of the question. Another example shown in Tabel 8 is from ANTIQUE, where the topic of answers is more diverse than factoid QA dataset. While TANDA predicts an answer with superficial relation with the question, our model make a more precise prediction without the disturbance of spurious relation. These examples demonstrates that our model can focus on the true correlation between the question and the answer, which is critical when the candidate answers contain misleading information.

Method	MAP	MRR	P@1
TANDA	0.8943	0.9063	0.8354
TANDA-Perturb	0.8898	0.9019	0.8270
TANDA-Noise	0.8740	0.8871	0.8059
SCAN	0.9164	0.9281	0.8776
SCAN-Perturb	0.9111	0.9228	0.8692
SCAN-Noise	0.8907	0.9055	0.8439

Table 6: Performance comparison when noise and perturbations are injected into WikiQA.

5.5 Error Analysis

Although our SCAN model achieves better performance than previous models, it still fails to handle

Question: "what company is cricket wireless by?"
Predicted by TANDA: "Cricket Communications, Inc., (d.b.a. Cricket Wireless) founded in 1999, provides wireless services to over 7 million customers in the United States." (***incorrect answer***)
Predicted by SCAN: "The company is a subsidiary of Leap Wireless, utilizing its CDMA 1X, 1xEV-DO and LTE networks." (***correct answer***)

Table 7: A question from WikiQA with the answers predicted by TANDA and SCAN, respectively.

Question: "what are some easy ways to get a toddler to go to sleep without being mean?"
Predicted by TANDA: "There are plenty of ways.... - The most obvious is try to sleep. - Take some Pepto-bismol. - Have a piece of peppermint (that's good too even if you still get sick)." (***incorrect answer***)
Predicted by SCAN: "Let them play for a while. Also, play with them. That way they'll feel like you care about them. Also, try laying down with them. That used to help my son, who now is 3 years old." (***correct answer***)

Table 8: A question from ANTIQUE with the answers predicted by TANDA and SCAN, respectively.

some cases. To investigate the limitations of SCAN, we analyze the bad cases produced by SCAN. We summarize the several reasons for obtaining the incorrect predictions. First, SCAN fails to tackle some questions that require commonsense knowledge to reason correct answers. In particular, some questions and the corresponding answers have different expressions for the same entities, thus our method struggles to capture the relations of the question-answer pairs based on the contextual representations only. One possible solution is to leverage knowledge bases to facilitate the reasoning process. Second, there are some noises (confused candidates) existing in the datasets. For example, the question *"Where was the first ski flying hill built?"* has two candidate answers *"Nevertheless the first-ever ski flying hill was built in Planica, Slovenia"* and *"The first ski flying hill was built in Planica in Slovenia"* with the former one labeled as incorrect and the latter one labeled as correct. However, both answers convey the same meaning. We may update the datasets by carefully examining the relevance labels of candidate answers.

6 Conclusion

In this paper, we proposed a novel spurious correlation reduction method to improve the robustness of the answer selection models from the sample and feature perspectives. First, we devised a feature decorrelation module by learning a weight for each training instance to remove the feature dependencies and reduced the spurious correlations without prior knowledge of such correlations. Second, we introduced a feature debiasing module with contrastive learning to alleviate the negative language biases and improved the robustness of the AS models. We conducted extensive experiments on three benchmark datasets and the experimental results showed the effectiveness of SCAN.

Acknowledgements

This work was partially supported by National Natural Science Foundation of China (61906185, 61876053), Youth Innovation Promotion Association of CAS China (No. 2020357), Shenzhen Science and Technology Innovation Program (Grant No. KQTD20190929172835662), Shenzhen Basic Research Foundation (No. JCYJ20210324115614039 and No. JCYJ20200109113441941).

References

Ching-Yao Chuang, Joshua Robinson, Yen-Chen Lin, Antonio Torralba, and Stefanie Jegelka. 2020. Debiased contrastive learning. *Advances in neural information processing systems*, 33:8765–8775.

Christopher Clark, Mark Yatskar, and Luke Zettlemoyer. 2019. Don't take the easy way out: ensemble based methods for avoiding known dataset biases. *EMNLP*.

Yang Deng, Wenxuan Zhang, and Wai Lam. 2021. Learning to rank question answer pairs with bilateral contrastive data augmentation. *arXiv preprint arXiv:2106.11096*.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*.

Siddhant Garg, Thuy Vu, and Alessandro Moschitti. 2020. Tanda: Transfer and adapt pre-trained transformer models for answer sentence selection. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pages 7780–7788.

Tejas Gokhale, Swaroop Mishra, Man Luo, Bhavdeep Sachdeva, and Chitta Baral. 2022. *Generalized but not Robust?* comparing the effects of data modification methods on out-of-domain generalization and adversarial robustness. In *Findings of the Association for Computational Linguistics: ACL 2022*, pages 2705–2718, Dublin, Ireland. Association for Computational Linguistics.

Arthur Gretton, Kenji Fukumizu, Choon Teo, Le Song, Bernhard Schölkopf, and Alex Smola. 2007. A kernel statistical test of independence. *Advances in neural information processing systems*, 20.

Jiahui Guo, Bin Yue, Guandong Xu, Zhenglu Yang, and Jin-Mao Wei. 2017. An enhanced convolutional neural network model for answer selection. In *WWW*, pages 789–790.

Suchin Gururangan, Swabha Swayamdipta, Omer Levy, Roy Schwartz, Samuel R Bowman, and Noah A Smith. 2018. Annotation artifacts in natural language inference data. *arXiv preprint arXiv:1803.02324*.

Helia Hashemi, Mohammad Aliannejadi, Hamed Zamani, and W Bruce Croft. 2020. Antique: A non-factoid question answering benchmark. In *European Conference on Information Retrieval*, pages 166–173. Springer.

Wanwei He, Yinpei Dai, Min Yang, Jian Sun, Fei Huang, Luo Si, and Yongbin Li. 2022a. Unified dialog model pre-training for task-oriented dialog understanding and generation. In *SIGIR*, pages 187–200.

Wanwei He, Yinpei Dai, Yinhe Zheng, Yuchuan Wu, Zheng Cao, Dermot Liu, Peng Jiang, Min Yang, Fei Huang, Luo Si, et al. 2022b. Galaxy: A generative pre-trained model for task-oriented dialog with semi-supervised learning and explicit policy injection. In *AAAI*, volume 36, pages 10749–10757.

Qiujun Huang, Jingli Mao, and Yong Liu. 2012. An improved grid search algorithm of svr parameters optimization. In *2012 IEEE 14th International Conference on Communication Technology*, pages 1022–1026. IEEE.

Robin Jia and Percy Liang. 2017. Adversarial examples for evaluating reading comprehension systems. *arXiv preprint arXiv:1707.07328*.

Tomasz Jurczyk, Michael Zhai, and Jinho D Choi. 2016. Selqa: A new benchmark for selection-based question answering. In *2016 IEEE 28th International Conference on Tools with Artificial Intelligence (ICTAI)*, pages 820–827. IEEE.

Seonhoon Kim, Inho Kang, and Nojun Kwak. 2019. Semantic sentence matching with densely-connected recurrent and co-attentive information. In *Proceedings of the AAAI conference on artificial intelligence*, volume 33, pages 6586–6593.

Tuan Lai, Quan Hung Tran, Trung Bui, and Daisuke Kihara. 2019. A gated self-attention memory network for answer selection. *arXiv preprint arXiv:1909.09696*.

Jieke Li, Min Yang, and Chengming Li. 2021. Clc-rs: A chinese legal case retrieval system with masked language ranking. In *CIKM*, pages 4734–4738.

Xiaoyan Li. 2003. Syntactic features in question answering. In *Proceedings of the 26th annual international ACM SIGIR conference on Research and development in informaion retrieval*, pages 383–384.

Che Liu, Rui Wang, Jinghua Liu, Jian Sun, Fei Huang, and Luo Si. 2021. Dialoguecse: Dialogue-based contrastive learning of sentence embeddings. *arXiv preprint arXiv:2109.12599*.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Roberta: A robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692*.

Sean MacAvaney, Franco Maria Nardini, Raffaele Perego, Nicola Tonellotto, Nazli Goharian, and Ophir Frieder. 2020. Training curricula for open domain answer re-ranking. In *Proceedings of the 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 529–538.

R Thomas McCoy, Ellie Pavlick, and Tal Linzen. 2019. Right for the wrong reasons: Diagnosing syntactic heuristics in natural language inference. *arXiv preprint arXiv:1902.01007*.

Guanglin Niu, Yang Li, Chengguang Tang, Ruiying Geng, Jian Dai, Qiao Liu, Hao Wang, Jian Sun, Fei Huang, and Luo Si. 2021. Relational learning with gated and attentive neighbor aggregator for few-shot knowledge graph completion. In *SIGIR*, pages 213–222.

Matthew E. Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer. 2018. Deep contextualized word representations. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 2227–2237, New Orleans, Louisiana. Association for Computational Linguistics.

Xipeng Qiu and Xuanjing Huang. 2015. Convolutional neural tensor network architecture for community-based question answering. In *Twenty-Fourth International Joint Conference on Artificial Intelligence*.

Ali Rahimi and Benjamin Recht. 2007. Random features for large-scale kernel machines. *Advances in neural information processing systems*, 20.

Siva Reddy, Danqi Chen, and Christopher D Manning. 2019. Coqa: A conversational question answering challenge. *Transactions of the Association for Computational Linguistics*, 7:249–266.

Shiori Sagawa, Pang Wei Koh, Tatsunori B Hashimoto, and Percy Liang. 2020. Distributionally robust neural networks for group shifts: On the importance of regularization for worst-case generalization. *ICLR*.

Cicero Nogueira dos Santos, Kahini Wadhawan, and Bowen Zhou. 2017. Learning loss functions for semi-supervised learning via discriminative adversarial networks. *arXiv preprint arXiv:1707.02198*.

Apoorv Saxena, Aditay Tripathi, and Partha Talukdar. 2020. Improving multi-hop question answering over knowledge graphs using knowledge base embeddings. In *ACL*, pages 4498–4507.

Aliaksei Severyn and Alessandro Moschitti. 2015. Learning to rank short text pairs with convolutional deep neural networks. In *Proceedings of the 38th international ACM SIGIR conference on research and development in information retrieval*, pages 373–382.

Dinghan Shen, Martin Min, Yitong Li, and Lawrence Carin. 2017a. Adaptive convolutional filter generation for natural language understanding.

Gehui Shen, Yunlun Yang, and Zhi-Hong Deng. 2017b. Inter-weighted alignment network for sentence pair modeling. In *Proceedings of the 2017 conference on empirical methods in natural language processing*, pages 1179–1189.

Ying Shen, Yang Deng, Min Yang, Yaliang Li, Nan Du, Wei Fan, and Kai Lei. 2018. Knowledge-aware attentive neural network for ranking question answer pairs. In *The 41st International ACM SIGIR Conference on Research & Development in Information Retrieval*, pages 901–904.

Mihai Surdeanu, Massimiliano Ciaramita, and Hugo Zaragoza. 2008. Learning to rank answers on large online qa collections. In *proceedings of ACL*, pages 719–727.

Ming Tan, Cicero Dos Santos, Bing Xiang, and Bowen Zhou. 2016. Improved representation learning for question answer matching. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 464–473.

Yi Tay, Minh C Phan, Luu Anh Tuan, and Siu Cheung Hui. 2017. Learning to rank question answer pairs with holographic dual lstm architecture. In *SIGIR*, pages 695–704. ACM.

Yi Tay, Luu Anh Tuan, and Siu Cheung Hui. 2018a. Cross temporal recurrent networks for ranking question answer pairs. In *Thirty-Second AAAI Conference on Artificial Intelligence*.

Yi Tay, Luu Anh Tuan, and Siu Cheung Hui. 2018b. Hyperbolic representation learning for fast and efficient neural question answering. In *Proceedings of the Eleventh ACM International Conference on Web Search and Data Mining*, pages 583–591.

Yi Tay, Luu Anh Tuan, and Siu Cheung Hui. 2018c. Multi-cast attention networks for retrieval-based question answering and response prediction. *arXiv preprint arXiv:1806.00778*.

Mengqiu Wang, Noah A Smith, and Teruko Mitamura. 2007. What is the jeopardy model? a quasi-synchronous grammar for qa. In *EMNLP-CoNLL*, pages 22–32.

Yuexiang Xie, Ying Shen, Yaliang Li, Min Yang, and Kai Lei. 2020. Attentive user-engaged adversarial neural network for community question answering. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pages 9322–9329.

Liu Yang, Qingyao Ai, Jiafeng Guo, and W Bruce Croft. 2016. anmm: Ranking short answer texts with attention-based neural matching model. In *Proceedings of the 25th ACM international on conference on information and knowledge management*, pages 287–296.

Min Yang, Lei Chen, Xiaojun Chen, Qingyao Wu, Wei Zhou, and Ying Shen. 2019a. Knowledge-enhanced hierarchical attention for community question answering with multi-task and adaptive learning. In *IJCAI*, pages 5349–5355.

Runqi Yang, Jianhai Zhang, Xing Gao, Feng Ji, and Haiqing Chen. 2019b. Simple and effective text matching with richer alignment features. *arXiv preprint arXiv:1908.00300*.

Yi Yang, Wen-tau Yih, and Christopher Meek. 2015. WikiQA: A challenge dataset for open-domain question answering. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 2013–2018, Lisbon, Portugal. Association for Computational Linguistics.

Wenpeng Yin, Hinrich Schütze, Bing Xiang, and Bowen Zhou. 2016. Abcnn: Attention-based convolutional neural network for modeling sentence pairs. *Transactions of the Association for Computational Linguistics*, 4:259–272.

Seunghyun Yoon, Franck Dernoncourt, Doo Soon Kim, Trung Bui, and Kyomin Jung. 2019. A compare-aggregate model with latent clustering for answer selection. In *Proceedings of the 28th ACM International Conference on Information and Knowledge Management*, pages 2093–2096.

Chunyuan Yuan, Wei Zhou, Mingming Li, Shangwen Lv, Fuqing Zhu, Jizhong Han, and Songlin Hu. 2019. Multi-hop selector network for multi-turn response selection in retrieval-based chatbots. In *EMNLP-IJCNLP*, pages 111–120.

Rowan Zellers, Yonatan Bisk, Roy Schwartz, and Yejin Choi. 2018. Swag: A large-scale adversarial dataset for grounded commonsense inference. *arXiv preprint arXiv:1808.05326*.

Xiaodong Zhang, Sujian Li, Lei Sha, and Houfeng Wang. 2017. Attentive interactive neural networks for answer selection in community question answering. In *AAAI*, pages 3525–3531.

Xingxuan Zhang, Peng Cui, Renzhe Xu, Linjun Zhou, Yue He, and Zheyan Shen. 2021a. Deep stable learning for out-of-distribution generalization. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 5372–5382.

Zeyu Zhang, Thuy Vu, and Alessandro Moschitti. 2021b. Joint models for answer verification in question answering systems. *arXiv preprint arXiv:2107.04217*.

Zeyu Zhang, Thuy Vu, and Alessandro Moschitti. 2022. Double retrieval and ranking for accurate question answering. *arXiv preprint arXiv:2201.05981*.

Xiaoqiang Zhou, Baotian Hu, Qingcai Chen, and Xiaolong Wang. 2018. Recurrent convolutional neural network for answer selection in community question answering. *Neurocomputing*, 274:8–18.

Understanding and Improving Zero-shot Multi-hop Reasoning in Generative Question Answering

Zhengbao Jiang[†], **Jun Araki**[‡], **Haibo Ding**[‡,*], **Graham Neubig**[†]
[†]Languages Technologies Institute, Carnegie Mellon University
[‡]Bosch Research
{zhengbaj,gneubig}@cs.cmu.edu
{jun.araki,haibo.ding}@us.bosch.com

Abstract

Generative question answering (QA) models generate answers to questions either solely based on the parameters of the model (the *closed-book* setting) or additionally retrieving relevant evidence (the *open-book* setting). Generative QA models can answer some relatively complex questions, but the mechanism through which they do so is still poorly understood. We perform several studies aimed at better understanding the multi-hop reasoning capabilities of generative QA models. First, we decompose multi-hop questions into multiple corresponding single-hop questions, and find marked inconsistency in QA models' answers on these pairs of ostensibly identical question chains. Second, we find that models lack zero-shot multi-hop reasoning ability: when trained only on single-hop questions, models generalize poorly to multi-hop questions. Finally, we demonstrate that it is possible to improve models' zero-shot multi-hop reasoning capacity through two methods that approximate real multi-hop natural language (NL) questions by training on either concatenation of single-hop questions or logical forms (SPARQL). In sum, these results demonstrate that multi-hop reasoning does not emerge naturally in generative QA models, but can be encouraged by advances in training or modeling techniques.[1]

1 Introduction

Empowered by large-scale pre-trained language models (LMs) (Devlin et al., 2019; Liu et al., 2019; Lewis et al., 2020a; Raffel et al., 2020), recent years have seen much progress on *generative question answering* (QA), where LMs generate answers given questions in an end-to-end fashion. While most works only demonstrate the performance of such generative QA models on simple questions (Joshi et al., 2017; Kwiatkowski et al., 2019), there has been some indication that these models can also answer complex questions that theoretically require multi-hop reasoning (Xiong et al., 2020), sometimes to an impressive degree. For example, Brown et al. (2020) demonstrate strong performance of LMs on multi-hop reasoning tasks such as DROP (Dua et al., 2019) which requires discrete reasoning and numeracy. On the other hand, many argue that LM-based QA models are not actually performing any reasoning, and rather performing (sophisticated) pattern matching and data memorization (Marcus and Davis, 2020). Simultaneously, in the context of extractive QA models that select answers from the provided context, several works have demonstrated that they can leverage superficial signals to return correct answers even when the context does not contain all the supporting facts (Chen and Durrett, 2019; Min et al., 2019a)

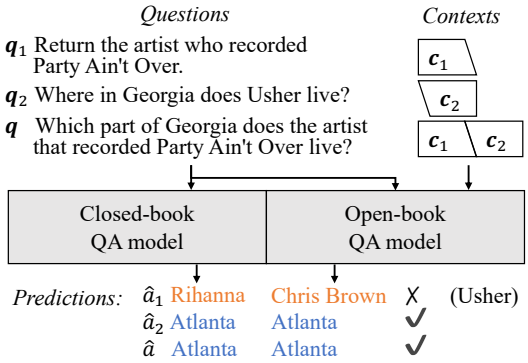

Figure 1: Probing generative closed- and open-book QA models with both multi-hop (q) and their component single-hop questions (q_1, q_2).

In this paper, we perform a closer examination of the multi-hop reasoning capabilities of generative QA models. To do so, we take multi-hop questions and their component single-hop questions to directly query generative QA models, studying their multi-hop reasoning ability. Specifically, we use multi-hop questions from the ComplexWebQuestions (Talmor and Berant, 2018) and HotpotQA (Yang et al., 2018; Tang et al., 2021) datasets as

*Haibo Ding is now at Amazon.
[1]Code is available at https://github.com/jzbjyb/multihop.

Proceedings of the 29th International Conference on Computational Linguistics, pages 1765–1775
October 12–17, 2022.

the testbed, and generate decomposed single-hop questions using heuristics (§ 2.2). We examine two types of generative QA models, namely *closed-book* (Roberts et al., 2020; Khashabi et al., 2020) and *open-book* (Guu et al., 2020; Lewis et al., 2020b; Izacard and Grave, 2021; Xiong et al., 2020) QA models that either do not or do refer to external knowledge when generating the answer respectively. Specifically, we use UnifiedQA (Khashabi et al., 2020) as a representative closed-book model, and RAG (Lewis et al., 2020b) as a representative open-book model (§ 2.1). We first ask:

RQ1 Is the correctness of decomposed single-hop questions a necessary and sufficient condition for correctness of multi-hop questions? (§ 3.2) Are answers to multi-hop and chains of decomposed questions consistent? (§ 3.3)

RQ2 Do models trained on single-hop questions demonstrate zero-shot generalization to multi-hop questions? (§ 4)

We find that generative QA models, even those close to the state-of-the-art, generally do not demonstrate robust multi-hop reasoning abilities, with success on multi-hop questions largely a result of taking shortcuts rather than true multi-hop reasoning. Zero-shot multi-hop reasoning ability does not emerge naturally from training on single-hop questions, which motivates our final question:

RQ3 Can we improve models' zero-shot multi-hop reasoning capacity by training on approximations of real multi-hop questions? (§ 4)

Motivated by the fact that pre-training on massive text endows LMs with the ability to identify semantically similar expressions, our first method uses concatenated decomposed single-hop questions to approximate real multi-hop questions. Our second method is inspired by recent work teaching LMs complex reasoning capabilities through neural execution of logical forms, e.g. by training neural models to execute SQL queries (Liu et al., 2021). We hypothesize that the ability to perform multi-hop reasoning can also be potentially learned from logical forms without reliance on NL questions. To this end, we propose to use SPARQL, a standard query language over knowledge bases, as our logical forms to endow generative QA models with the ability to perform multi-hop reasoning, and examine whether learning to execute SPARQL transfers to the ability to answer NL multi-hop questions. Both methods lead to significant improvement on zero-shot multi-hop reasoning performance, and further improvements are obtained when both are combined, opening possibilities for future work (§ 6).

2 Generative Question Answering

In this section, we briefly introduce generative QA models and multi-hop QA datasets. Then we elaborate on how we use multi-hop and decomposed questions to perform experiments.

2.1 Generative QA Models

There are two main classes of generative QA models: closed-book and open-book. Closed-book QA models usually consist of a sequence-to-sequence model that takes in a question $\boldsymbol{q}$ and calculates the probability of an answer $\boldsymbol{a}$ based on model parameters θ (Roberts et al., 2020; Khashabi et al., 2020):

$$P(\boldsymbol{a}|\boldsymbol{q};\theta) = \prod_{i=1}^{|\boldsymbol{a}|} P(a_i|\boldsymbol{q}, \boldsymbol{a}_{<i};\theta),$$

Because these models can only refer to model parameters, any relevant information must be stored in the parameters (Roberts et al., 2020). Open-book QA models first retrieve relevant context $\boldsymbol{c}$ from external resources, then generate answers using both questions and context (Guu et al., 2020; Lewis et al., 2020b; Izacard and Grave, 2021):

$$P(\boldsymbol{a}|\boldsymbol{c}, \boldsymbol{q};\theta) = \prod_{i=1}^{|\boldsymbol{a}|} P(a_i|\boldsymbol{c}, \boldsymbol{q}, \boldsymbol{a}_{<i};\theta),$$

We examine both types of models since we hypothesize that the difference in inputs might lead to different mechanisms of multi-hop reasoning.

Specifically, as our example of a closed-book model we use the UnifiedQA model of Khashabi et al. (2020). The UnifiedQA model is based on the T5 model (Raffel et al., 2020), which is an encoder-decoder model trained on the Colossal Clean Crawled Corpus (C4) by a denoising objective. It further fine-tunes on a variety of QA datasets by converting different QA formats into a unified sequence-to-sequence format.

We use the RAG model of Lewis et al. (2020b) as our example of an open-book QA model, which consists of a retriever for searching relevant passages $\boldsymbol{p}$, and a generator which generates answers $\boldsymbol{a}$ given both $\boldsymbol{p}$ and $\boldsymbol{q}$. The retriever is based on the dense passage retrieval model (DPR) (Karpukhin et al., 2020), and the generator is based on BART

Type	Questions (hop1, hop2, and multi-hop)	*Answers*
Composition	Return the country where Limonese Creole is spoken.	*Costa Rica*
	Which continent is Costa Rica located?	*North America*
	On which continent is Limonese Creole spoken?	*North America*
Conjunction	What team is Reggie Bush on 2011?	*Miami Dolphins, New Orleans Saints*
	Which one of the following is the team won the super bowl XLIV championship: Miami Dolphins, New Orleans Saints?	*New Orleans Saints*
	What team that won the super bowl XLIV championship was Reggie Bush in 2011?	*New Orleans Saints*
Superlative	What countries does the Niger River flow through?	*Benin, Guinea, Mali, Niger Nigeria*
	Which one of the following country calling code is smallest: Benin, Guinea, Mali, Niger, Nigeria?	*Mali*
	What country with the smallest calling code does the Niger River flow through?	*Mali*
Comparative	What were Hitler's parents names?	*Alois Hitler, Klara Hitler*
	Which one of the following person's date of death is after 1903-01-03: Alois Hitler, Klara Hitler?	*Klara Hitler*
	Which of Hitler's parents died after 3 January 1903?	*Klara Hitler*

Table 1: Each multi-hop question $\boldsymbol{q}$ from ComplexWebQuestions is decomposed into two single-hop questions $\boldsymbol{q}_1$ and $\boldsymbol{q}_2$. Underlined entities in the second single-hop questions are actually the answer to the first hop.

(Lewis et al., 2020a), which is also an encoder-decoder model that encodes both context and question, and generates answers autoregressively.

2.2 Multi-hop Questions and Decompositions

To understand multi-hop reasoning in generative QA models, we propose to query models using both multi-hop questions and their decompositions into multiple single-hop questions, and perform analysis based on the predictions.

To this end, we choose the **ComplexWebQuestions** dataset (Talmor and Berant, 2018) as our major testbed, as it contains multi-hop questions based on simple questions from the WebQuestionsSP dataset (Yih et al., 2016), and we can leverage simple heuristics to obtain decomposed single-hop questions and corresponding answers. Another advantage of ComplexWebQuestions is that it contains four types of questions: composition, conjunction, superlative, and comparative. This allows us to perform fine-grained analysis over these categories. Specifically, we follow heuristics in Talmor and Berant (2018) to generate decompositions. For the composition type, they use questions from WebQuestionsSP as the second hop, and replace an entity in it with a relational phrase to generate multi-hop questions. We revert this process to get the first-hop question. For the other three types, they use questions from WebQuestionsSP with multiple answers as the first hop, and add additional conditions to form the multi-hop questions. We extract those conditions and use the following template to generate the second hop question: "Which one of the following [condition]: [candidate answers]". Tab. 1 includes examples of multi-hop questions and their decompositions of four types.

We also use another small dataset from Tang et al. (2021) to test the generality of models, where a subset of multi-hop questions from **HotpotQA** (Yang et al., 2018) are manually annotated with decompositions. This dataset only contains a single type of question, which is composition. ComplexWebQuestions has 27,639/3,519 questions in the training/development set, and HotpotQA has 1,000 questions in the development set.[2]

2.3 Answer Generation and Evaluation

We use $\boldsymbol{q}_t, t \in \{1, ..., T\}$ to denote the t-th decomposed single-hop question for a multi-hop question $\boldsymbol{q}$ with T hops. Correspondingly, we use $\boldsymbol{a}_t$ to denote answers and $\boldsymbol{c}_t$ to denote retrieved context for the single-hop question $\boldsymbol{q}_t$. Since the last single-hop question always has the same answer as the corresponding multi-hop question, $\boldsymbol{a}_T = \boldsymbol{a}$. We use $\hat{\boldsymbol{a}}_t$/$\hat{\boldsymbol{a}}$ to denote the predictions from single-/multi-hop questions generated with greedy decoding:

$$\begin{matrix}\hat{\boldsymbol{a}} \\ \hat{\boldsymbol{a}}_t\end{matrix} = \arg\max_{\boldsymbol{y}} P\Big(\boldsymbol{y}\Big|\begin{matrix}[\boldsymbol{c},]\boldsymbol{q} \\ [\boldsymbol{c}_t,]\boldsymbol{q}_t\end{matrix}; \theta\Big).$$

We query models using all decomposed questions $\boldsymbol{q}_t$ and multi-hop questions $\boldsymbol{q}$ which are concatenated with the corresponding context ($\boldsymbol{c}_t$ or $\boldsymbol{c}$) for open-book settings to get predicted answers. All questions from ComplexWebQuestions and HotpotQA have two hops (i.e., $T = 2$), thus in the following sections we always use $T = 2$.

Pseudo-gold context for oracle-book models Previous work clearly demonstrates that a better retrieval component usually implies higher open-book QA performance, as it results in more retrieved contexts with answers (Chen et al., 2017; Lee et al., 2019; Karpukhin et al., 2020). Therefore, we ablate out the influence of the retrieval

[2]Since the test sets of both datasets are hidden, we use development sets for evaluation purposes. Break (Wolfson et al., 2020) is another testbed with multi-hop questions and manually decomposed questions. However, the decomposed questions are not annotated with answers, making it less appropriate for our study.

component and focus on understanding the mechanism through which generative QA models parse multi-hop questions and generate answers.

We try to provide context that contains answers to the QA model so failure of answering the question can be mainly attributed to the generator instead of the retriever. Since gold context is not annotated in the datasets, we follow Karpukhin et al. (2020) to obtain pseudo-gold context. Specifically, we use the DPR model to retrieve the top-100 passages to each single-hop question $\boldsymbol{q}_t$, and find the first one containing the answer $\boldsymbol{a}_t$, which is denoted as the pseudo-gold passage $\boldsymbol{p}_t^{\checkmark}$. Only using pseudo-gold passages as the context might make the task too easy because no incorrect contexts are presented. Therefore, we concatenate the pseudo-gold passage with a negative passage $\boldsymbol{p}_t^{\times}$ which is the first retrieved passage not containing the answers: $\boldsymbol{c}_t = [\boldsymbol{p}_t^{\checkmark}, \boldsymbol{p}_t^{\times}]$.[3] For multi-hop questions $\boldsymbol{q}$, we concatenate all context of the decomposed questions: $\boldsymbol{c} = [\boldsymbol{c}_1, ..., \boldsymbol{c}_T]$. We fix the context for all of our experiments, and only use the generator of the RAG model. For clarity, instead of open-book we use *oracle-book* to refer to these QA models in the following sections.

Multi-answer generation Since some questions involve multiple answers, as shown in Tab. 1, we fine-tune generative QA models to generate multiple answers separated by a special symbol "#".

Evaluation metrics We follow previous works (Roberts et al., 2020; Khashabi et al., 2020; Lewis et al., 2020b) to use exact match (EM) as our major evaluation metric, which measures the percentage of predictions that match the ground truth answers exactly (Rajpurkar et al., 2016; Yang et al., 2018). Since we allow multi-answer generation, we split the prediction by the special symbol "#" and match each entry against all the answers. The prediction is judged as correct if all answers are included and no extra entry is predicted.

3 Probing Multi-hop Questions and Decompositions

To answer the first research question, we probe generative QA models on both multi-hop questions and their decompositions, examining the similarities and differences in models' behavior thereon. We hypothesize that if models answer multi-hop questions in a robust way, they should be able to perform multi-hop reasoning by following the chain of decompositions internally, which makes being able to answer decomposed questions a necessary and/or sufficient condition of being able to answer multi-hop questions. Motivated by this, we choose two probing angles to examine this question. The first angle evaluates the prediction correctness on decomposed and multi-hop questions, and investigates whether there is a correlation between them. The second angle generates predictions by answering multi-hop questions and the corresponding chain of decomposed single-hop questions in a sequence, and examining whether predictions are consistent.

[3]The concatenation order is randomized to avoid leaking superficial signals to QA models.

Model	Type	Hop1	Hop2	Multi-hop
UnifiedQA	overall	32.91	49.13	33.25
	composition	47.49	38.67	33.40
	conjunction	22.49	63.30	38.01
	superlative	16.23	48.69	21.99
	comparative	15.53	25.57	8.68
RAG	overall	58.72	65.11	60.32
	composition	76.23	61.24	60.51
	conjunction	25.12	78.82	66.50
	superlative	13.33	76.67	53.33
	comparative	17.65	35.29	26.47

Table 2: EM of two models on ComplexWebQuestions overall or each type separately.

Model	Type	Hop1	Hop2	Multi-hop
UnifiedQA	composition	1.70	1.30	1.20
RAG	composition	31.55	21.66	6.15

Table 3: EM of two models on HotpotQA.

3.1 Experimental Settings

We fine-tune the UnifiedQA and RAG model using both single- and multi-hop QA pairs from the training set of the ComplexWebQuestions dataset.[4] Then we generate predictions for both single- and multi-hop questions from the test set of the ComplexWebQuestions/HotpotQA datasets, and show their overall results in Tab. 2 and Tab. 3 respectively. We measure the EM metric on first-hop $\boldsymbol{q}_1$ (**Hop1**), second-hop $\boldsymbol{q}_2$ (**Hop2**), and multi-hop questions $\boldsymbol{q}$ (**Multi-hop**) separately. We also group examples by four types to investigate whether different types of reasoning exhibit different regularities.

To examine the correlation between success on decomposed and multi-hop questions, we bucket

[4]We follow the default hyperparameters of UnifiedQA for 100K steps and a batch size of 16 on a single TPU, and the default hyperparameters of RAG for 10 epochs with a batch size of 4 on a single V100 GPU.

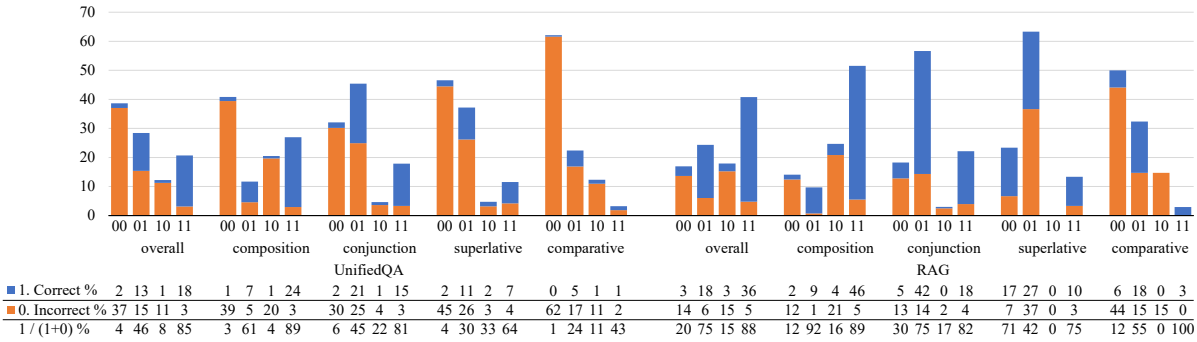

Figure 2: Correctness confusion matrices of two models on ComplexWebQuestions. Two binary codes on the X-axis indicates the correctness of the first/second single-hop question $s_1s_2 = \{00, 01, 10, 11\}$. In the table, the first/second row indicates the percentage (%) of examples of which the multi-hop question is correctly/incorrectly answered $P(s = \{1, 0\}, s_1s_2)$; the last row indicates the conditional success rate $P(s = 1|s_1s_2)$.

examples by their correctness. We use s_1, s_2 and s to denote correctness of predictions generated from the first/second single-hop and multi-hop questions, which is either 0 (incorrect) or 1 (correct). There are $8 = 2^3$ *configurations* of the correctness of a triple. We present the results using the correctness confusion matrices in Fig. 2, where examples are bucketed into 4 bins by correctness on single-hop questions (i.e., $s_1s_2 = \{00, 01, 10, 11\}$) and the inner blue/orange bars indicate the percentage of the corresponding configurations (i.e., $P(s = \{1, 0\}, s_1s_2 = \{00, 01, 10, 11\})$). To better reveal the correlation between decomposed and multi-hop questions, we compute the *conditional success rate* on multi-hop questions $P(s = 1|s_1s_2) = \frac{P(s=1,s_1s_2)}{P(s=1,s_1s_2)+P(s=0,s_1s_2)}$ in the last row of the table, which indicates how likely multi-hop questions are correctly answered given the correctness on single-hop decompositions.[5]

To examine the prediction consistency between multi-hop questions and chains of decompositions, we replace entities in the second single-hop questions $\boldsymbol{q}_2$ which correspond to answers to the first hop with a special placeholder "#1", and denoted it as $\boldsymbol{q}_2^*$. When answering a chain of decomposed questions, predictions from the first hop $\hat{\boldsymbol{a}}_1$ are used to replace the placeholder in the second hop: $\boldsymbol{q}_2^*(\hat{\boldsymbol{a}}_1)$, from which we generate the final answer denoted as $\hat{\boldsymbol{a}}_2^*$. Models fine-tuned in the normal setting only generate final answers from multi-hop questions, but not intermediate answers (i.e., answers to the hop1 question). To examine whether models can predict intermediate answers from the multi-hop question, and measure consistency on both, we append two prompts to multi-hop questions to instruct models to generate two predictions:

$$\hat{\boldsymbol{a}}_1^*, \hat{\boldsymbol{a}} = \arg\max_{\boldsymbol{y}} P\Big(\boldsymbol{y}\Big|[\boldsymbol{c},]\boldsymbol{q}, \begin{matrix}\text{"Intermediate answer:"}\\ \text{"Final answer:"}\end{matrix}\Big)$$

$$\hat{\boldsymbol{a}}_1, \hat{\boldsymbol{a}}_2^* = \arg\max_{\boldsymbol{y}} P\Big(\boldsymbol{y}\Big|\begin{matrix}[\boldsymbol{c}_1,]\boldsymbol{q}_1\\ [\boldsymbol{c}_2,]\boldsymbol{q}_2^*(\hat{\boldsymbol{a}}_1)\end{matrix}\Big),$$

where $\hat{\boldsymbol{a}}_1^*$ denotes intermediate predictions. Predictions from multi-hop questions ($\hat{\boldsymbol{a}}_1^*/\hat{\boldsymbol{a}}$) are compared with predictions from decomposed questions in sequence ($\hat{\boldsymbol{a}}_1/\hat{\boldsymbol{a}}_2^*$) respectively to measure their consistency.

3.2 Correlation of Correctness

Multi-hop performance is unexpectedly high Given the hypothesis that being able to answer decomposed questions is a prerequisite of being able to answer multi-hop questions, we expect *a priori* that the performance on multi-hop questions will be much lower than the performance on all single-hop questions due to error propagation. However, what we observe on ComplexWebQuestions is the opposite: overall, the multi-hop performance is slightly higher than the hop1 performance, and the gap between hop2 and multi-hop performance is much smaller than may be expected, especially for the oracle-book RAG model. This indicates that generative QA models somehow manage to take shortcuts when answering multi-hop questions, i.e., being able to answer the multi-hop question without correctly answering its component questions.

Success on decompositions does not always imply success on multi-hop questions Looking at the overall percentage, we can see that indeed the success rate on multi-hop questions is highest if both decomposed questions are correctly answered, i.e., $P(s = 1|s_1s_2 = 11) = 85\%/88\%$ for the UnifiedQA/RAG model respectively, indicating that generative QA models are more likely to answer

[5] For robust models, $P(s = 1|s_1s_2 = 11)$ should be close to 1, $P(s = 1|s_1s_2 = \{00, 01, 10\})$ should be close to 0.

Type	Questions (hop1, hop2, multi-hop)	Answers	Predictions
Compo.	Return the country where Cerveceria Modelo Corona light beer is made.	Mexico	Mexico
	Who is Mexico's president right now 2011?	Felipe Calderón	Felipe Calderón
	Who was the president in 2011 in the country where Cerveceria Modelo Corona light beer is made?	Felipe Calderón	Juan Manuel Santos
Conj.	What year did Detroit Pistons win the championship?	2004, 1990, 1989 NBA Finals	2004, 1990, 1989 NBA Finals
	Which one of the following sports championship results were 4-1: 2004, 1990, 1989 NBA Finals?	2004, 1990 NBA Finals	2004, 1990 NBA Finals
	In what year did the Detroit Pistons win the sports championship where the results were 4-1?	2004, 1990 NBA Finals	2002 NBA Finals
Compo.	Return the team won the 2006 NFC championship	Seattle Seahawks	Indianapolis Colts
	Where do the Seattle Seahawks play?	CenturyLink Field	CenturyLink Field
	Which Stadium does the team that claimed the 2006 NFC championship play in?	CenturyLink Field	CenturyLink Field
Compa.	Who is the leader of France 2012?	Nicolas Sarkozy, François Hollande	Nicolas sarkozy
	Which one of the following started tenure after 1979: Nicolas Sarkozy, François Hollande?	Nicolas Sarkozy	Nicolas Sarkozy
	Who was the leader of France from 1979 until 2012?	Nicolas Sarkozy	Nicolas Sarkozy

Table 4: Cases of predictions generated from single/multi-hop questions of different types. Correct/Incorrect predictions are indicated in blue/orange.

multi-hop questions if they can answer all decomposed single-hop questions. However, there are still 15%/12% examples where correctness on both decomposed questions does not imply correctness on multi-hop questions, as shown by the first two examples in Tab. 4. The predictions generated from the multi-hop questions are usually of the correct type, but they diverge from the predictions generated from decomposed questions, indicating that models do not necessarily follow the decomposed components when answering multi-hop questions.

Multi-hop success is most correlated with success on the last hop Even when models fail on decomposed questions, they can still answer some percentage of multi-hop questions correctly (4-46%/15-75%) depending on which of the decomposed hops fails. Success on hop2 questions is more correlated with success on multi-hop questions than hop1 questions (i.e., $P(s = 1|s_1s_2 = 01) > P(s = 1|s_1s_2 = 10)$), especially for the oracle-book RAG model. When the model is only able to answer the second single-hop questions, there is still $46\%/75\%$ chance that the model can answer the multi-hop questions in closed/oracle-book settings respectively, indicating that generative QA models manage to take shortcuts instead of performing real reasoning. The shortcuts could be some superficial signals in the context or parameters that generative QA models can take advantage of to bypass the requirement of the first hop, as shown by the third example in Tab. 4. Or for multi-hop questions with multiple intermediate answers, generative QA models might not need to know all of them in order to answer the multi-hop questions, as shown by the fourth example in Tab. 4.

Other observations Overall, the oracle-book RAG model performs significantly better than the closed-book UnifiedQA model on both datasets, indicating that knowledge stored in parametric generative QA models is still limited and it is beneficial to provide external evidence. Hop2 performance is significantly higher than the hop1 performance on conjunction, superlative, and comparative questions, which is because hop1 questions usually have more answers than hop2 questions as shown in Tab. 1, thus being harder.[6] Both models generalize poorly to the unseen HotpotQA dataset (Tab. 3), indicating that the learned multi-hop reasoning capability cannot generalize across datasets.

	Type	EM				Consistency	
		Decompose		Multi-hop			
		Hop1	Hop2	Hop1	Hop2	Hop1	Hop2
UnifiedQA	overall	32.48	32.23	30.78	31.40	50.81	36.12
	compo.	51.87	33.97	48.51	32.13	58.92	43.24
	conj.	17.73	34.68	17.54	34.68	44.46	33.51
	super.	13.09	24.08	11.52	23.04	38.74	26.70
	compa.	13.24	9.59	12.79	10.50	47.49	11.42
RAG	overall	56.51	62.65	61.92	58.11	79.61	65.48
	compo.	73.86	60.88	76.78	54.48	86.47	67.46
	conj.	23.65	74.38	30.05	72.41	67.98	68.47
	super.	13.33	60.00	33.33	56.67	60.00	43.33
	compa.	11.76	23.53	38.24	32.35	55.88	35.29

Table 5: EM of predictions from answering chains of decomposed questions and multi-hop questions on ComplexWebQuestions and their consistency (%).

3.3 Prediction Consistency

Predictions are not consistent between multi-hop questions and chains of decompositions As shown in Tab. 5, consistency is relatively low for both models and on both hop1 and hop2, indicating that generative QA models answer multi-hop questions not necessarily in the same way as they answer decomposed questions in sequence. The consistency of the UnifiedQA model is lower than the consistency of the RAG model, which is because knowledge is limited in closed-book QA

[6] Note that the difference in the difficulty of hop1 and hop2 questions does *not* invalidate our previous conclusion about correctness correlation since we use conditional success rate.

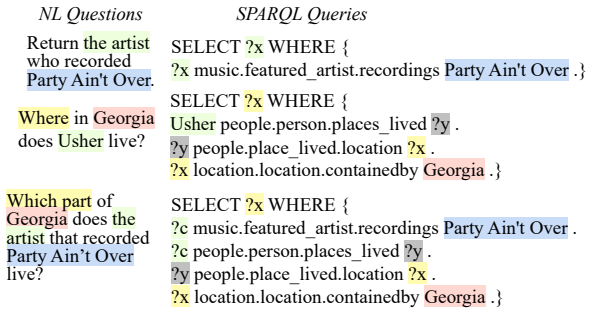

Figure 3: NL questions and corresponding SPARQL queries. Mentions of the same entity are in the same color.

models, and navigating in parameters implicitly is probably harder than searching chains of evidence in context explicitly. Consistency on the first hop is usually higher than consistency on the second hop, which is because inconsistent intermediate predictions ($\hat{a}_1$) will propagate to the second hop, leading to accumulated inconsistency.

4 Improving Zero-shot Multi-hop Reasoning Capability

In this section, we first examine LMs' zero-shot capacity for multi-hop reasoning when they are *not* trained on multi-hop NL questions. Compositional generalization ability (Lake and Baroni, 2018) is required in this case to generalize from single-hop to multi-hop questions. Unsurprisingly, generative QA models perform poorly in this setting, with almost half performance degradation. Since multi-hop NL questions are expensive to obtain, one natural question is "is it possible to improve the multi-hop reasoning ability using only single-hop NL questions, or even without any NL questions?"

We design two methods to achieve this goal. Motivated by the fact that UnifiedQA and RAG models are initialized with language models pre-trained on massive text corpora, which endows them with the ability to identify semantically similar expressions, our first method uses concatenated decomposed NL questions (i.e., $[\boldsymbol{q}_1, \boldsymbol{q}_2^*]$) to approximate the real NL multi-hop question $\boldsymbol{q}$, and fine-tunes models on them.

The second is inspired by recent progress in teaching LMs complex reasoning capabilities by executing logical forms neurally. For example, Liu et al. (2021) formulate the execution of SQL over tables as a seq2seq task where the input is a logical form string associated with a table and the output is answers (Liu et al., 2021). We hypothesize that in our multi-hop QA setting, the ability to perform multi-hop reasoning can also be potentially learned from logical forms without reliance on any NL question. To this end, we propose to use SPARQL, which is a standard query language over knowledge bases, as our logical forms. We then examine whether the ability to answer questions expressed in these SPARQL queries is transferable to NL multi-hop questions. The advantage of using SPARQL for training is that SPARQL queries can be easier and cheaper to obtain or generate than NL. For example we can use existing query logs,[7] or use manual SPARQL queries as templates and replace entities/relation to generate more queries.[8] Our observation sheds light on potential improvement on multi-hop reasoning using many SPARQL query-answer pairs.

4.1 Experimental Settings

Each NL multi-hop question in the ComplexWebQuestions dataset is associated with a SPARQL query based on the Freebase schema. We follow similar heuristics described in § 2.2 to generate SPARQL queries for the first- and second-hop NL questions. Each single- and multi-hop SPARQL query is used as a pseudo input question after replacing entity identifiers with their names, as shown in Fig. 3. In order to answer the above research questions, we design the following settings:

- No fine-tuning (**Default**): This setting uses the original model without fine-tuning.
- **S**ingle- and **M**ulti-hop **NL** (**SM-NL**): The normal setting discussed in § 3.2 where we train the model using both single- and multi-hop NL questions. This serves as the upper bound of the zero-shot performance.
- **S**ingle-hop **NL** (**S-NL**): Only use decomposed single-hop NL questions for training.
- **S**ingle- and **M**ulti-hop **SPARQL** (**SM-SPARQL**): Only use SPARQL queries.
- **S-NL** with **Concat**enation (**S-NL+Concat**): Use concatenated decomposed NL questions in addition to **S-NL**.
- **SM-NL+Concat+SM-SPARQL** (**Combo.**): Combine the previous two settings to leverage both NL and SPARQL for training.

4.2 Experimental Results

Tab. 6 includes results for all the above experimental settings. Compared to the oracle multi-hop performance, performance of only using single-hop NL questions (**S-NL**) drops by almost half on

[7] https://bit.ly/3wRRIPZ
[8] https://bit.ly/3ciOFqy

	Setting	Supervision Single	Supervision Multi	Hop1	Hop2	Multi-hop
UnifiedQA	Default			0.71	15.37	6.56
	S-NL	•		33.28	49.33	17.02
	+Concat	•	○	31.91	48.25	25.69
	SM-SPARQL	□	□	19.04	34.67	24.84
	Combo.	•□	○□	32.76	48.51	**27.14**
	SM-NL	•	•	32.91	49.13	33.25
RAG	Default			7.99	12.65	7.62
	S-NL	•		59.83	68.55	34.03
	+Concat	•	○	61.06	64.13	**53.93**
	SM-SPARQL	□	□	49.51	58.48	51.60
	Combo.	•□	○□	57.37	62.53	53.07
	SM-NL	•	•	58.72	65.11	60.32

Table 6: EM on NL questions in zero-shot multi-hop evaluation, where •, ○, □ denotes NL, concatenation, and SPARQL respectively. Oracle performance using multi-hop NL questions has a gray background. Best zero-shot multi-hop performance is in bold.

both UnifiedQA (33.25 → 17.02) and RAG models (60.32 → 34.03), indicating that without learning on multi-hop questions, compositional generalization does not naturally emerge in generative QA.

Single-hop concatenation is a good approximation of multi-hop questions Surprisingly, by simply concatenating single-hop NL questions and fine-tuning on them, multi-hop performance increases by a large margin (17.02 → 25.69/34.03 → 53.93), indicating that simple concatenation is an effective approximation for multi-hop questions. We hypothesize that LMs pre-trained on noisy text have the paraphrasing ability to generalize from concatenated simple sentences to complex sentences at least to some degree.

Models generalize from SPARQL to NL questions SPARQL queries explicitly specify compositional structure using pre-defined grammar and canonicalized entities/relations, while NL questions express this process in a more flexible way. Despite this gap, models trained solely on SPARQL queries are able to generalize to NL questions at test time on both single- and multi-hop questions, with a performance drop of 7-15 on both single- and multi-hop questions compared to (**SM-SPARQL** vs. **SM-NL**), which is far better than no fine-tuning (**Default**). This indicates that when answering NL questions, the ability learned from mapping SPARQL queries to answers can be reused by the model, similar to the observation on table-based QA (Liu et al., 2021). As demonstrated in other tasks such as table-based QA (Jiang et al., 2022) and text-to-SQL (Wu et al., 2021), converting the SPARQL queries into NL questions and training models on them can potentially mitigate the gap and further improve the performance, which we plan to explore in future works.

Combining concatenation and SPARQL improves further In this setting, we attempt to combine the merits of using concatenated single-hop NL questions, which are more natural, and SPARQL queries, which are more explicit with respect to the reasoning process. Compared to training on two types of supervision separately (**S-NL+Concat** and **SM-SPARQL**), training on both jointly (**Combo.**) improves the multi-hop performance of UnifiedQA (25.69→27.14) while slightly hurting the performance of RAG (53.93→53.07). We hypothesize that closed-book models are less constrained compared to oracle-book models due to the existence of the additional context, therefore closed-book models can benefit from the stronger supervision from a combination of two methods. Note that there is still a large gap between fine-tuning on multi-hop NL questions (**SM-NL**) and zero-shot settings, which indicates the potential for better approximations or modeling techniques.

5 Related Work

Multi-hop QA models Most multi-hop QA models proposed so far are pipeline methods that generate sub-questions to retrieve evidence iteratively (Qi et al., 2019; Ding et al., 2019; Qiu et al., 2019; Das et al., 2019; Asai et al., 2020; Min et al., 2019b; Perez et al., 2020; Xiong et al., 2020). The final answers are generated either by reading each retrieved evidence independently and recomposing the generated intermediate answers (Min et al., 2019b; Perez et al., 2020), or by taking all evidence as input at once (Qi et al., 2019; Das et al., 2019; Asai et al., 2020). Instead, we focus on understanding the multi-hop reasoning capabilities of end-to-end generative QA models in this paper.

Analysis of multi-hop reasoning Several works studying multi-hop reasoning in extractive QA models found that they can leverage superficial signals to extract answers even when the context does not contain all supporting facts (Chen and Durrett, 2019; Min et al., 2019a; Trivedi et al., 2020; Jiang and Bansal, 2019; Niu et al., 2020; Lee et al., 2021). While they examine from the perspective of dataset bias, we directly query models with both multi-hop and component single-hop questions, using both closed- and open-book generative QA models. Another work that studied multi-hop QA models using

both multi-hop and single-hop questions is Tang et al. (2021). While they use pipeline extractive QA models, we focus on end-to-end generative QA models and investigate correctness, consistency, and compositional generalization ability.

6 Conclusion

In this paper, we examined the multi-hop reasoning capabilities of generative QA models, finding that overall models take shortcuts when answering multi-hop questions, not demonstrating convincing multi-hop reasoning capability. When trained only on single-hop questions, models generalize poorly to multi-hop questions, while approximation using the concatenation of single-hop questions and SPARQL queries improves the multi-hop performance significantly. Further directions include better approximations of multi-hop questions and advanced modeling techniques that encourage compositional ability.

References

Akari Asai, Kazuma Hashimoto, Hannaneh Hajishirzi, Richard Socher, and Caiming Xiong. 2020. Learning to retrieve reasoning paths over wikipedia graph for question answering. In *8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, April 26-30, 2020*. OpenReview.net.

Tom B. Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel M. Ziegler, Jeffrey Wu, Clemens Winter, Christopher Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. 2020. Language models are few-shot learners. In *Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020, December 6-12, 2020, virtual*.

Danqi Chen, Adam Fisch, Jason Weston, and Antoine Bordes. 2017. Reading wikipedia to answer open-domain questions. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics, ACL 2017, Vancouver, Canada, July 30 - August 4, Volume 1: Long Papers*, pages 1870–1879. Association for Computational Linguistics.

Jifan Chen and Greg Durrett. 2019. Understanding dataset design choices for multi-hop reasoning. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT 2019, Minneapolis, MN, USA, June 2-7, 2019, Volume 1 (Long and Short Papers)*, pages 4026–4032. Association for Computational Linguistics.

Rajarshi Das, Shehzaad Dhuliawala, Manzil Zaheer, and Andrew McCallum. 2019. Multi-step retriever-reader interaction for scalable open-domain question answering. In *7th International Conference on Learning Representations, ICLR 2019, New Orleans, LA, USA, May 6-9, 2019*. OpenReview.net.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT 2019, Minneapolis, MN, USA, June 2-7, 2019, Volume 1 (Long and Short Papers)*, pages 4171–4186. Association for Computational Linguistics.

Ming Ding, Chang Zhou, Qibin Chen, Hongxia Yang, and Jie Tang. 2019. Cognitive graph for multi-hop reading comprehension at scale. In *Proceedings of the 57th Conference of the Association for Computational Linguistics, ACL 2019, Florence, Italy, July 28- August 2, 2019, Volume 1: Long Papers*, pages 2694–2703. Association for Computational Linguistics.

Dheeru Dua, Yizhong Wang, Pradeep Dasigi, Gabriel Stanovsky, Sameer Singh, and Matt Gardner. 2019. DROP: A reading comprehension benchmark requiring discrete reasoning over paragraphs. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT 2019, Minneapolis, MN, USA, June 2-7, 2019, Volume 1 (Long and Short Papers)*, pages 2368–2378. Association for Computational Linguistics.

Kelvin Guu, Kenton Lee, Zora Tung, Panupong Pasupat, and Ming-Wei Chang. 2020. REALM: retrieval-augmented language model pre-training. *CoRR*, abs/2002.08909.

Gautier Izacard and Edouard Grave. 2021. Leveraging passage retrieval with generative models for open domain question answering. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume, EACL 2021, Online, April 19 - 23, 2021*, pages 874–880. Association for Computational Linguistics.

Yichen Jiang and Mohit Bansal. 2019. Avoiding reasoning shortcuts: Adversarial evaluation, training, and model development for multi-hop QA. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 2726–2736, Florence, Italy. Association for Computational Linguistics.

Zhengbao Jiang, Yi Mao, Pengcheng He, Graham Neubig, and Weizhu Chen. 2022. OmniTab: Pretraining with natural and synthetic data for few-shot table-based question answering. In *Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 932–942, Seattle, United States. Association for Computational Linguistics.

Mandar Joshi, Eunsol Choi, Daniel S. Weld, and Luke Zettlemoyer. 2017. Triviaqa: A large scale distantly supervised challenge dataset for reading comprehension. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics, ACL 2017, Vancouver, Canada, July 30 - August 4, Volume 1: Long Papers*, pages 1601–1611. Association for Computational Linguistics.

Vladimir Karpukhin, Barlas Oguz, Sewon Min, Patrick S. H. Lewis, Ledell Wu, Sergey Edunov, Danqi Chen, and Wen-tau Yih. 2020. Dense passage retrieval for open-domain question answering. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing, EMNLP 2020, Online, November 16-20, 2020*, pages 6769–6781. Association for Computational Linguistics.

Daniel Khashabi, Sewon Min, Tushar Khot, Ashish Sabharwal, Oyvind Tafjord, Peter Clark, and Hannaneh Hajishirzi. 2020. Unifiedqa: Crossing format boundaries with a single QA system. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: Findings, EMNLP 2020, Online Event, 16-20 November 2020*, pages 1896–1907. Association for Computational Linguistics.

Tom Kwiatkowski, Jennimaria Palomaki, Olivia Redfield, Michael Collins, Ankur P. Parikh, Chris Alberti, Danielle Epstein, Illia Polosukhin, Jacob Devlin, Kenton Lee, Kristina Toutanova, Llion Jones, Matthew Kelcey, Ming-Wei Chang, Andrew M. Dai, Jakob Uszkoreit, Quoc Le, and Slav Petrov. 2019. Natural questions: a benchmark for question answering research. *Trans. Assoc. Comput. Linguistics*, 7:452–466.

Brenden M. Lake and Marco Baroni. 2018. Generalization without systematicity: On the compositional skills of sequence-to-sequence recurrent networks. In *Proceedings of the 35th International Conference on Machine Learning, ICML 2018, Stockholmsmässan, Stockholm, Sweden, July 10-15, 2018*, volume 80 of *Proceedings of Machine Learning Research*, pages 2879–2888. PMLR.

Kenton Lee, Ming-Wei Chang, and Kristina Toutanova. 2019. Latent retrieval for weakly supervised open domain question answering. In *Proceedings of the 57th Conference of the Association for Computational Linguistics, ACL 2019, Florence, Italy, July 28- August 2, 2019, Volume 1: Long Papers*, pages 6086–6096. Association for Computational Linguistics.

Kyungjae Lee, Seung-won Hwang, Sang-eun Han, and Dohyeon Lee. 2021. Robustifying multi-hop QA through pseudo-evidentiality training. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 6110–6119, Online. Association for Computational Linguistics.

Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Veselin Stoyanov, and Luke Zettlemoyer. 2020a. BART: denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, ACL 2020, Online, July 5-10, 2020*, pages 7871–7880. Association for Computational Linguistics.

Patrick S. H. Lewis, Ethan Perez, Aleksandra Piktus, Fabio Petroni, Vladimir Karpukhin, Naman Goyal, Heinrich Küttler, Mike Lewis, Wen-tau Yih, Tim Rocktäschel, Sebastian Riedel, and Douwe Kiela. 2020b. Retrieval-augmented generation for knowledge-intensive NLP tasks. In *Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020, December 6-12, 2020, virtual*.

Qian Liu, Bei Chen, Jiaqi Guo, Zeqi Lin, and Jian-Guang Lou. 2021. TAPEX: table pre-training via learning a neural SQL executor. *CoRR*, abs/2107.07653.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Roberta: A robustly optimized BERT pretraining approach. *CoRR*, abs/1907.11692.

Gary Marcus and Ernest Davis. 2020. Gpt-3, bloviator: Openai's language generator has no idea what it's talking about. *Technology Review*.

Sewon Min, Eric Wallace, Sameer Singh, Matt Gardner, Hannaneh Hajishirzi, and Luke Zettlemoyer. 2019a. Compositional questions do not necessitate multi-hop reasoning. In *Proceedings of the 57th Conference of the Association for Computational Linguistics, ACL 2019, Florence, Italy, July 28- August 2, 2019, Volume 1: Long Papers*, pages 4249–4257. Association for Computational Linguistics.

Sewon Min, Victor Zhong, Luke Zettlemoyer, and Hannaneh Hajishirzi. 2019b. Multi-hop reading comprehension through question decomposition and rescoring. In *Proceedings of the 57th Conference of the Association for Computational Linguistics, ACL 2019, Florence, Italy, July 28- August 2, 2019, Volume 1: Long Papers*, pages 6097–6109. Association for Computational Linguistics.

Yilin Niu, Fangkai Jiao, Mantong Zhou, Ting Yao, Jingfang Xu, and Minlie Huang. 2020. A self-training

method for machine reading comprehension with soft evidence extraction. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 3916–3927, Online. Association for Computational Linguistics.

Ethan Perez, Patrick S. H. Lewis, Wen-tau Yih, Kyunghyun Cho, and Douwe Kiela. 2020. Unsupervised question decomposition for question answering. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing, EMNLP 2020, Online, November 16-20, 2020*, pages 8864–8880. Association for Computational Linguistics.

Peng Qi, Xiaowen Lin, Leo Mehr, Zijian Wang, and Christopher D. Manning. 2019. Answering complex open-domain questions through iterative query generation. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing, EMNLP-IJCNLP 2019, Hong Kong, China, November 3-7, 2019*, pages 2590–2602. Association for Computational Linguistics.

Lin Qiu, Yunxuan Xiao, Yanru Qu, Hao Zhou, Lei Li, Weinan Zhang, and Yong Yu. 2019. Dynamically fused graph network for multi-hop reasoning. In *Proceedings of the 57th Conference of the Association for Computational Linguistics, ACL 2019, Florence, Italy, July 28- August 2, 2019, Volume 1: Long Papers*, pages 6140–6150. Association for Computational Linguistics.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J. Liu. 2020. Exploring the limits of transfer learning with a unified text-to-text transformer. *J. Mach. Learn. Res.*, 21:140:1–140:67.

Pranav Rajpurkar, Jian Zhang, Konstantin Lopyrev, and Percy Liang. 2016. Squad: 100, 000+ questions for machine comprehension of text. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing, EMNLP 2016, Austin, Texas, USA, November 1-4, 2016*, pages 2383–2392. The Association for Computational Linguistics.

Adam Roberts, Colin Raffel, and Noam Shazeer. 2020. How much knowledge can you pack into the parameters of a language model? In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing, EMNLP 2020, Online, November 16-20, 2020*, pages 5418–5426. Association for Computational Linguistics.

Alon Talmor and Jonathan Berant. 2018. The web as a knowledge-base for answering complex questions. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT 2018, New Orleans, Louisiana, USA, June 1-6, 2018, Volume 1 (Long Papers)*, pages 641–651. Association for Computational Linguistics.

Yixuan Tang, Hwee Tou Ng, and Anthony K. H. Tung. 2021. Do multi-hop question answering systems know how to answer the single-hop sub-questions? In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume, EACL 2021, Online, April 19 - 23, 2021*, pages 3244–3249. Association for Computational Linguistics.

Harsh Trivedi, Niranjan Balasubramanian, Tushar Khot, and Ashish Sabharwal. 2020. Is multihop QA in dire condition? measuring and reducing disconnected reasoning. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing, EMNLP 2020, Online, November 16-20, 2020*, pages 8846–8863. Association for Computational Linguistics.

Tomer Wolfson, Mor Geva, Ankit Gupta, Yoav Goldberg, Matt Gardner, Daniel Deutch, and Jonathan Berant. 2020. Break it down: A question understanding benchmark. *Trans. Assoc. Comput. Linguistics*, 8:183–198.

Kun Wu, Lijie Wang, Zhenghua Li, Ao Zhang, Xinyan Xiao, Hua Wu, Min Zhang, and Haifeng Wang. 2021. Data augmentation with hierarchical SQL-to-question generation for cross-domain text-to-SQL parsing. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 8974–8983, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Wenhan Xiong, Xiang Lorraine Li, Srinivasan Iyer, Jingfei Du, Patrick S. H. Lewis, William Yang Wang, Yashar Mehdad, Wen-tau Yih, Sebastian Riedel, Douwe Kiela, and Barlas Oguz. 2020. Answering complex open-domain questions with multi-hop dense retrieval. *CoRR*, abs/2009.12756.

Zhilin Yang, Peng Qi, Saizheng Zhang, Yoshua Bengio, William W. Cohen, Ruslan Salakhutdinov, and Christopher D. Manning. 2018. Hotpotqa: A dataset for diverse, explainable multi-hop question answering. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing, Brussels, Belgium, October 31 - November 4, 2018*, pages 2369–2380. Association for Computational Linguistics.

Wen-tau Yih, Matthew Richardson, Christopher Meek, Ming-Wei Chang, and Jina Suh. 2016. The value of semantic parse labeling for knowledge base question answering. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics, ACL 2016, August 7-12, 2016, Berlin, Germany, Volume 2: Short Papers*. The Association for Computer Linguistics.

Domain Adaptation for Question Answering via Question Classification

Zhenrui Yue and **Huimin Zeng** and **Ziyi Kou** and **Lanyu Shang** and **Dong Wang**
School of Information Sciences
University of Illinois Urbana-Champaign
{zhenrui3, huiminz3, ziyikou2, lshang3, dwang24}@illinois.edu

Abstract

Question answering (QA) has demonstrated impressive progress in answering questions from customized domains. Nevertheless, domain adaptation remains one of the most elusive challenges for QA systems, especially when QA systems are trained in a source domain but deployed in a different target domain. In this work, we investigate the potential benefits of question classification for QA domain adaptation. We propose a novel framework: Question Classification for Question Answering (QC4QA). Specifically, a question classifier is adopted to assign question classes to both the source and target data. Then, we perform joint training in a self-supervised fashion via pseudo-labeling. For optimization, inter-domain discrepancy between the source and target domain is reduced via maximum mean discrepancy (MMD) distance. We additionally minimize intra-class discrepancy among QA samples of the same question class for fine-grained adaptation performance. To the best of our knowledge, this is the first work in QA domain adaptation to leverage question classification with self-supervised adaptation. We demonstrate the effectiveness of the proposed QC4QA with consistent improvements against the state-of-the-art baselines on multiple datasets.

1 Introduction

Question Answering (QA) or Reading Comprehension (RC) refers to the task of extracting answers from given context paragraphs based on input questions. QA systems predict the start and end positions of possible answer spans in given context documents upon input questions. In recent studies, QA systems have achieved significant improvements with transformer models and large-scale datasets (Rajpurkar et al., 2016; Devlin et al., 2019; Yue et al., 2022a).

Once deployed, QA systems often experience performance deterioration upon user-generated questions. Such performance drops can be traced back to domain shifts in two input elements: (1) User-generated questions are syntactically more diverse and thus, different from the training QA pairs; (2) The context domain of test-time input (target domain) can oftentimes diverge from the training corpora (source domain), e.g., from news snippets to biomedical articles (Hazen et al., 2019; Fisch et al., 2019; Miller et al., 2020).

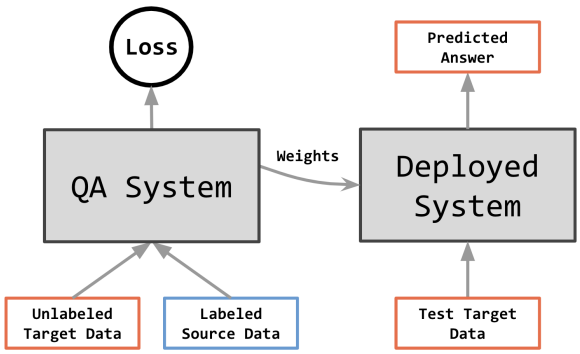

Figure 1: Overview for QA domain adaptation. A QA model is trained with labeled source data and unlabeled target data. The resulting QA system is deployed to answer target questions.

To alleviate the performance issue in QA domain adaptation, several approaches have been proposed to reduce the discrepancy between the source and target domains. Integrating labeled target QA pairs in training can effectively improve the QA system in answering out-of-domain questions (Kamath et al., 2020; Shakeri et al., 2020; Yue et al., 2021, 2022b), where the target data can be human-annotated QA pairs or synthetic data using question generation methods. When only unlabeled questions are available (see Figure 1), another possible approach is to reduce inter-domain discrepancy via domain-adversarial training (Lee et al., 2019). Combined with pseudo labeling, QA systems demonstrate improved generalization in answering target domain questions (Cao et al., 2020).

Nevertheless, previous methods either require large amounts of annotated target data or exten-

Proceedings of the 29th International Conference on Computational Linguistics, pages 1776–1790
October 12–17, 2022.

sive computing power (Lee et al., 2019; Cao et al., 2020; Yue et al., 2021, 2022b). Additionally, different types of QA pairs and their distributional changes are not taken into account. As a result, existing approaches are less effective for adapting QA systems to an unseen target domain. In this paper, we propose a domain adaptation framework for QA: *question classification for question answering* (QC4QA). Unlike existing methods, we innovatively adopt a question classification (QC) model to classify input questions from both the source and target domains into different question classes. Moreover, we pseudo label the target data using a pretrained QA system and perform distribution-aware sampling to build mini-batches that resemble the target question distribution. In the adaptation stage, we propose a self-supervised adaptation framework to minimize the domain gap, in which inter-domain and intra-class discrepancies are simultaneously regularized. This is in contrast to existing baselines (e.g., domain adversarial adaptation methods) where the source data is solely used for training but without explicitly accounting for domain shifts and question distribution changes (Lee et al., 2019; Cao et al., 2020; Yue et al., 2021). To the best of our knowledge, QC4QA is the first work that combines question classification and self-supervised adaptation for learning domain-invariant representation in QA domain adaptation.

Our main contributions are as follows[1]:

1. We propose QC4QA for QA domain adaptation. QC4QA innovatively adopts question classification to identify question types (classes) of the source and target QA pairs for intra-class discrepancy reduction.

2. Our QC4QA can be combined with supervised QC or unsupervised clustering. In the latter case, we show that QC4QA can transfer knowledge to the target domain even without additional model or annotation.

3. We design a distribution-aware sampling strategy and an objective function that incorporates MMD distances for minimizing inter-domain and intra-class discrepancies to transfer knowledge to the target domain.

4. We demonstrate the effectiveness of QC4QA, where QC4QA consistently outperforms state-of-the-art baselines by a significant margin.

[1]Our implementation is publicly available at https://github.com/Yueeeeeeee/Self-Supervised-QA.

2 Related Work

QA systems have achieved significant improvements in extracting answers upon input context and questions. However, trained QA systems are known to experience performance drops when context paragraphs and questions diverge from the training corpora. That is, when domain shifts exist between the training and test distributions (Hazen et al., 2019; Fisch et al., 2019; Miller et al., 2020; Zeng et al., 2022).

To adapt QA systems for domain changes, methods for QA domain adaptation have been proposed in two different settings: (1) Access to contexts and QA pairs from the target domain. Here, partial access to target data is provided, or a question generation model is introduced for producing synthetic QA pairs. The target data is then used to train and improve adaptation performance (Shakeri et al., 2020; Yue et al., 2021); (2) Access to context paragraphs and unlabeled input questions from the target domain. Here, unsupervised or self-supervised adaptation can be used to improve the performance in the target domain (Cao et al., 2020). In this paper, we focus on the latter setting and study QA domain adaptation with access to target contexts and unlabeled questions.

Domain adaptation in computer vision: Domain adaptation methods have been primarily studied for image classification problems. Such approaches focus on minimizing the representation discrepancy between the source and target distributions. Some methods design objective functions that encourage domain-invariant features in training (Long et al., 2015; Kang et al., 2019). Other methods leverage domain-adversarial training with a discriminator to implicitly impose regularization when source and target features are distinguishable (Tzeng et al., 2017; Zhang et al., 2019a), with successful applications in various vision tasks (Zhang et al., 2019b, 2020, 2021).

Domain adaptation in QA: Various approaches are designed to improve QA performance by generating and refining synthetic QA pairs. Based on target contexts, question generation models are introduced to produce a surrogate dataset, which is used to train QA systems (Kamath et al., 2020; Shakeri et al., 2020; Yue et al., 2022b). Contrastive adaptaion minimizes inter-domain discrep-

ancy with question generation and maximum mean discrepancy (MMD) distances (Yue et al., 2021, 2022c). When unlabeled questions are accessible, domain-adversarial training can be applied to reduce feature discrepancy between domains (Lee et al., 2019). Pseudo labeling and iterative refinements of such labels can be used for improved joint training (Cao et al., 2020).

Question classification (QC): Classifying questions of different types is a common task in natural language processing. One of the widely-used question taxonomy TREC divides questions into 6 coarse classes and 50 fine classes (Li and Roth, 2002). Early machine learning methods perform QC with hand-crafted features (Li and Roth, 2002; Huang et al., 2008). Neural networks improve the classification performance with sentence embeddings (Howard and Ruder, 2018; Cer et al., 2018).

However, the aforementioned approaches in QA domain adaptation encourage domain-invariant features without considering samples from different classes and their distributional changes. Moreover, it is hitherto unclear how to estimate class discrepancies in QA, since class labels are not available in QA datasets. To solve this problem, we propose to use QC to divide QA pairs into different classes, where questions can be classified via an additional QC model or unsupervised clustering with minimum computational costs. We exploit the question classes by reducing the discrepancy among samples from the same class ('intra-class'). Additionally, we design a distribution-aware sampling strategy in QC4QA to account for distributional changes between the source and target domains. By incorporating the discrepancy terms in the objective function, our self-supervised adaptation framework QC4QA achieves significant improvements against the state-of-the-art baseline methods.

3 Methodology

3.1 Setup

Data: Our setting focuses on improving QA performance when domain shifts exist in the test data distribution. For this purpose, labeled source data and unlabeled target data are available, we denote the domain of source data with $\boldsymbol{\mathcal{D}}_s$ and target data with $\boldsymbol{\mathcal{D}}_t$. Formally, the input data is defined by:

- *Source data*: Labeled source data $\boldsymbol{X}_s$ from $\boldsymbol{\mathcal{D}}_s$. Individual sample $\boldsymbol{x}_s^{(i)} \in \boldsymbol{X}_s$ is defined by a triplet consisting of a question $\boldsymbol{x}_{s,q}^{(i)}$, a context $\boldsymbol{x}_{s,c}^{(i)}$, and an answer $\boldsymbol{x}_{s,a}^{(i)}$. The exact answer tokens can be found in context, answer $\boldsymbol{x}_{s,a}^{(i)}$ is represented by the start and end position in $\boldsymbol{x}_{s,c}^{(i)}$.
- *Target data*: Unlabeled target data $\boldsymbol{X}_t$ from $\boldsymbol{\mathcal{D}}_t$. For target sample $\boldsymbol{x}_t^{(i)} \in \boldsymbol{X}_t$, we only have access to the question $\boldsymbol{x}_{t,q}^{(i)}$ and context $\boldsymbol{x}_{t,c}^{(i)}$. Ground truth answer $\boldsymbol{x}_{t,a}^{(i)}$ is not given for training.

Model: The QA system can be represented with function $\boldsymbol{f}$. $\boldsymbol{f}$ takes an input question $\boldsymbol{x}_q$ and context document $\boldsymbol{x}_c$ as input and yields answer prediction $\boldsymbol{x}_a$, namely $\boldsymbol{x}_a = \boldsymbol{f}(\boldsymbol{x}_q, \boldsymbol{x}_c)$. The output $\boldsymbol{x}_a$ is represented as a subspan of $\boldsymbol{x}_c$ and comprises of the answer start and end positions.

Objective: The objective is to learn a $\boldsymbol{f}^*$, which maximizes the performance in answering questions from the target domain $\boldsymbol{\mathcal{D}}_t$. In other words, $\boldsymbol{f}^*$ minimizes the negative log likelihood (i.e., cross entropy) for $\boldsymbol{X}_t$ from the target domain distribution:

$$\boldsymbol{f}^* = \arg\min_{\boldsymbol{f}} \sum_{i=1}^{|\boldsymbol{X}_t|} \mathcal{L}_{\text{NLL}}(\boldsymbol{f}(\boldsymbol{x}_{t,q}^{(i)}, \boldsymbol{x}_{t,c}^{(i)}), \boldsymbol{x}_{t,a}^{(i)}). \quad (1)$$

3.2 The QC4QA Framework

3.2.1 Overall Framework

In the proposed QC4QA, we design a self-supervised framework that facilitates question classification for QA domain adaptation. QC4QA can be divided into three stages: (1) Question classification; (2) Pseudo labeling & sampling and (3) Self-supervised adaptation. In the first stage, we perform classification for all input questions, which provides additional question class information for the adaptation stage. In the next stage, we label and filter all target samples and perform distribution-aware sampling to build mini-batches that resemble the target data distribution. Finally, we perform self-supervised adaptation on the QA system to minimize inter-domain and intra-class discrepancies. Once input questions are classified, we iteratively perform stage 2 and stage 3 in each epoch. The QA system is trained with both source and target data, where we encourage domain-invariant features and minimize intra-class discrepancies of data samples from the same question class.

Our approach leverages question classification for fine-grained domain adaptation. Here, QC is

designed for evaluating intra-class discrepancies and distributional changes by introducing the additional question classes instead of using QA labels. The idea behind it is that QA labels are defined by subspans in input contexts, if we treat every combination of start and end position as a class, the corresponding label space would be too large and sparse for any meaningful discrepancy estimation. Therefore, we proposes the question classification stage to introduce additional semantic knowledge for intra-class discrepancy estimation. Moreover, by performing pseudo labeling and distribution-aware sampling, we resemble the target question distribution in the adaptation stage to correct the potential bias in the pretrained QA system. In other words, QC4QA simulates the target data distribution and 'pulls together' source and target samples of the same question class to encourage domain invariance.

3.2.2 Question Classification

For question classification, we adopt the commonly used question taxonomy in TREC and categorize all questions into 6 coarse classes Q: **ABBR**: *Abbreviation*, **DESC**: *Description*, **ENTY**: *Entity*, **HUM**: *Human*, **LOC**: *Location* and **NUM**: *Numeric Value*. Each class indicates the potential answer type to the question (Li and Roth, 2002). In practice, we rarely find **ABBR** questions.

The proposed QC model leverages pretrained sentence embedding methods to generate vectorized feature for input questions. We then build a multilayer perceptron (MLP) to perform classification on the encoded questions, see Figure 2. Specifically, we adopt InferSent and Universal Sentence Encoder to encode the input questions separately (Conneau et al., 2017; Cer et al., 2018). The encodings are concatenated and used as an input feature for the MLP classifier. With the trained QC model, inference can be performed on all training questions for the later adaptation stage.

To further examine the effectiveness of question classification without additional model and annotation, we introduce an unsupervised clustering method, where we refrain from using an additional dataset or classifier to perform question classification. In particular, we feed the input data within the transformer encoder (part of the QA system) and utilize the output from the `[CLS]` token position as features (Devlin et al., 2019). We sample a fixed number of source features (10k in our experiments) and perform KMeans clustering with a predefined

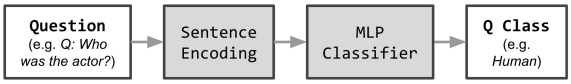

Figure 2: Model architecture for question classifier.

number of clusters k (Similar to TREC, we use 5 as default). Then, cluster centroids are preserved to classify source and target QA datasets.

3.2.3 Pseudo Labeling & Sampling

Provided with the access to labeled source data, we pretrain the QA system $\boldsymbol{f}$ to learn to answer questions. After pretraining, we can use $\boldsymbol{f}$ to predict target answers for self-supervised adaptation. The pseudo labels are filtered according to the answer confidence, we preserve the target samples above confidence threshold λ_{con}. The pseudo labeling and confidence thresholding steps are repeated in each epoch to dynamically adjust the target distribution used for training.

For mini-batch training, we sample the same amounts of QA pairs from both domains to minimize the inter-domain and intra-class discrepancies. However, with randomly sampled data, training is less efficient as source and target questions in each batch can be entirely different (e.g., source samples are all *Human* questions and target samples are all *Description* questions). To solve this problem, we design a distribution-aware sampling strategy: we first sample target QA pairs from $\boldsymbol{X}_t$ and within the same question classes, we sample from $\boldsymbol{X}_s$ such that the source and target question classes in each batch are identical. Consequently, the QA system can be trained on a data distribution similar to the target dataset. Moreover, the estimation of intra-class discrepancy between both domains can be performed more efficiently.

3.2.4 Self-supervised Adaptation

The sampled batches are used to adapt the pretrained QA system, where we optimize the model to reduce the negative log likelihood loss, as in Equation (1). Meanwhile, we encourage the domain invariance by computing the discrepancies and incorporate them in the training objective.

To measure the discrepancy between samples from different domains, we adopt the maximum mean discrepancy (MMD) distance (Gretton et al., 2012). MMD estimates the distance between two distributions with samples drawn from them, with f and $\mathcal{H}$ representing the feature mapping and the

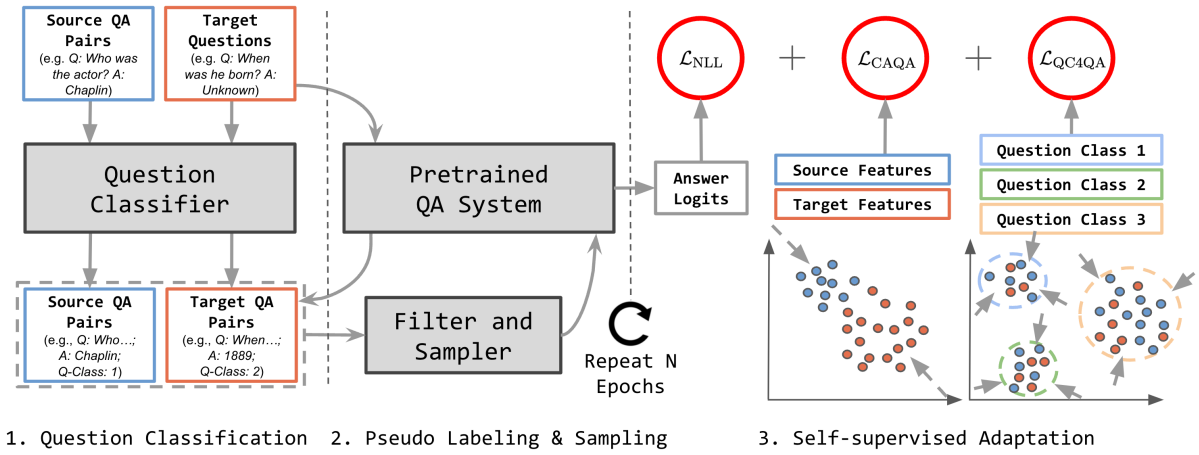

Figure 3: Overview for the proposed method. QC4QA can be divided into three stages: (1) Question classification where all questions are assigned different classes; (2) Pseudo labeling & sampling, where we label and sample target examples with the proposed distribution-aware sampling strategy; (3) Self-supervised adaptation, in which we train the QA system jointly with source and target data. In the experiments, stage 2 and 3 are iteratively performed and we apply the proposed objective Equation (5) to minimize both inter-domain and intra-class discrepancies.

reproducing kernel Hilbert space:

$$\mathcal{D} = \sup_{f \in \mathcal{H}} \Big(\frac{1}{|\boldsymbol{X}_s|} \sum_{i=1}^{|\boldsymbol{X}_s|} f(\boldsymbol{x}_s^{(i)}) - \frac{1}{|\boldsymbol{X}_t|} \sum_{i=1}^{|\boldsymbol{X}_t|} f(\boldsymbol{x}_t^{(i)}) \Big). \tag{2}$$

To simplify the computation, we adopt the Gaussian kernel as feature mapping, i.e., $k(\boldsymbol{x}_s^{(i)}, \boldsymbol{x}_t^{(j)}) = \exp(-\frac{\|\boldsymbol{x}_s^{(i)} - \boldsymbol{x}_t^{(j)}\|^2}{\gamma})$. We further leverage empirical kernel mean embeddings (Long et al., 2015) to estimate the MMD distance between samples from $\boldsymbol{X}_s$ and $\boldsymbol{X}_t$:

$$\begin{aligned} \mathcal{D}^{\text{MMD}} = & \frac{1}{|\boldsymbol{X}_s||\boldsymbol{X}_s|} \sum_{i=1}^{|\boldsymbol{X}_s|} \sum_{j=1}^{|\boldsymbol{X}_s|} k(\phi(\boldsymbol{x}_{\text{s}}^{(i)}), \phi(\boldsymbol{x}_{\text{s}}^{(j)})) \\ & + \frac{1}{|\boldsymbol{X}_t||\boldsymbol{X}_t|} \sum_{i=1}^{|\boldsymbol{X}_t|} \sum_{j=1}^{|\boldsymbol{X}_t|} k(\phi(\boldsymbol{x}_{\text{t}}^{(i)}), \phi(\boldsymbol{x}_{\text{t}}^{(j)})) \\ & - \frac{2}{|\boldsymbol{X}_s||\boldsymbol{X}_t|} \sum_{i=1}^{|\boldsymbol{X}_s|} \sum_{j=1}^{|\boldsymbol{X}_t|} k(\phi(\boldsymbol{x}_{\text{s}}^{(i)}), \phi(\boldsymbol{x}_{\text{t}}^{(j)})), \end{aligned} \tag{3}$$

where ϕ represents the transformer encoder in the QA system. With $\mathcal{D}^{\text{MMD}}$, it is possible to measure the discrepancies between different domains and question classes. The discrepancy values are used to guide the self-supervised adaptation and encourage domain-invariant features.

Among all tokens in each QA sample $\boldsymbol{x}$, we distinguish two types of features $\boldsymbol{x}_{\text{a}}$ and $\boldsymbol{x}_{\text{o}}$. $\boldsymbol{x}_{\text{a}}$ stands for the mean vector of answer token representations, while $\boldsymbol{x}_{\text{o}}$ is the mean vector of all other tokens in the representation space (Yue et al., 2021). The QA system extracts the answer when the answer tokens in the representation space are separated from $\boldsymbol{x}_{\text{o}}$ (van Aken et al., 2019). Therefore, we adopt $\boldsymbol{x}_{\text{a}}$ as feature representation to compute MMD distances. By introducing question classification, we introduce an additional term w.r.t. the intra-class discrepancy in the objective function:

$$\mathcal{L}_{\text{QC4QA}} = \frac{1}{|Q|} \sum_{q \in Q}^{Q} \mathcal{D}^{\text{MMD}}(\boldsymbol{X}_s^{(q)}, \boldsymbol{X}_t^{(q)}), \tag{4}$$

$\boldsymbol{X}_s$ refers to all answer features in source samples and $\boldsymbol{X}_t$ represents answer features in target samples, while $\boldsymbol{X}^{(q)}$ denotes the set of samples that belong to question class $q \in Q$. $\mathcal{L}_{\text{QC4QA}}$ 'pulls together' features from the same question class across domains by minimizing their MMD distances.

3.2.5 Overall Objective

To encourage domain-invariant features, we incorporate Equation (4) into the training objective. Using the NLL loss and the contrastive adaptaion loss (Yue et al., 2021, 2022c), the overall objective function can be formulated as follows:

$$\mathcal{L} = \mathcal{L}_{\text{NLL}} + \lambda(\mathcal{L}_{\text{CAQA}} + \mathcal{L}_{\text{QC4QA}}), \tag{5}$$

in which $\mathcal{L}_{\text{CAQA}}$ is the same as in (Yue et al., 2021), while λ is a scaling factor we choose empirically. Although we introduce $\mathcal{L}_{\text{CAQA}}$ in our training objective function, QC4QA is largely different from CAQA as: (1) $\mathcal{L}_{\text{CAQA}}$ only reduces the

inter-domain discrepancy, we incorporate question classification to additionally reduce intra-class discrepancy via $\mathcal{L}_{\text{QC4QA}}$ for fine-grained adaptation; (2) We perform pseudo labeling and distribution-aware sampling to account for the distribution shifts between the source and target dataset; and (3) QC4QA leverages an efficient self-supervised adaptation framework instead of the computationally expensive question generation in (Yue et al., 2021). As such, the proposed QC4QA efficiently reduces the domain discrepancy and effectively transfers learnt knowledge from the source domain to the target domain.

The overall framework is illustrated in Figure 3. We first generate question classes for all data samples. Next, the source-pretrained QA model generates pseudo labels for target data and we select target samples above the confidence threshold λ_{con} for training. Pseudo labeling and self-supervised adaptation are performed iteratively to refine the pseudo labels and improve the performance in the target domain using Equation (5). Unlike previous works (Lee et al., 2019; Cao et al., 2020; Shakeri et al., 2020; Yue et al., 2021), we discard domain-adversarial training or question generation and introduce a self-supervised adaptation framework based on question classification for improved efficiency and adaptation performance. We also design a distribution-aware sampling strategy to resemble the target data distribution and correct the potential bias in the pretrained QA system. Additionally, a fine-grained adaptation loss based on question classification is introduced in training to minimize both the inter-domain and intra-class discrepancies across the source and target domain.

4 Experiments

4.1 Datasets and Baselines

For supervised question classification, we adopt the TREC dataset (Li and Roth, 2002), a widely used dataset containing ~5k training questions and 500 questions for testing. Following (Cao et al., 2020; Shakeri et al., 2020; Yue et al., 2021), we use SQuAD as our source domain QA dataset (Rajpurkar et al., 2016). For target domain, we adopt multiple QA datasets (details in Appendix A) and refrain from using labels in training (Cao et al., 2020; Shakeri et al., 2020; Yue et al., 2021).

For comparison, we adopt 4 baseline methods. We first pretrain a QA system on the source dataset and then evaluate on each target dataset with zero knowledge of the target domain. We additionally adopt 3 state-of-the-art baselines: Domain-adversarial training (DAT) (Lee et al., 2019), conditional adversarial self-training (CASe) (Cao et al., 2020) and contrastive adaptation for QA (CAQA*) (Yue et al., 2021). For fair comparison, we adapt the original CAQA to our self-supervised adaptation framework as a baseline, we denote the adapted CAQA with CAQA*.[2] BERT-QA is selected as the QA model (Devlin et al., 2019). Details of the baselines are elaborated in Appendix A.

4.2 Training and Evaluation

We train our QC model on the TREC training set and evaluate on the test set, the best model is saved to perform classification on all QA datasets. For unsupervised question classification, sampled `[CLS]` features from the source dataset are used to perform KMeans clustering, followed by question class inference on all QA datasets.

After question classification, we adopt a QA model (pretrained on the source dataset) and iteratively perform: (1) Pseudo labeling and distribution-aware sampling to select data batches that resemble the target data distribution; (2) Self-supervised adaptation with the proposed objective Equation (5) for learning domain-invariant representation. For evaluation, we adopt two metrics: exact match (EM) and F1 score (F1). We compute the metrics on target dev sets to evaluate the adaptation performance. Details of our implementation can be found in Appendix B.

4.3 Main Results

We first report the question classification performance on TREC dataset. The MLP classifier has 2.36M parameters and can be trained efficiently in less than one minute (57.6s on average) with GPU acceleration. We perform the evaluation with the proposed MLP QC model and reach an accuracy of 96.6% on the TREC test set. Similar magnitude of efficiency can be observed in KMeans clustering for unsupervised question classification. We provide detailed quantitative analysis and qualitative examples in Appendix C.1.

The QA system is first pretrained in the source domain with 79.60 EM and 87.64 F1 score on

[2]We exclude question generation and adopt the same process of pseudo labeling, distribution-aware sampling and self-supervised adaptation as QC4QA in CAQA*. Different from QC4QA, we use the same objective function as in (Yue et al., 2021). A direct comparison between the proposed QC4QA and the original CAQA can be found in Appendix C.2.

Model	CNN EM / F1	Daily Mail EM / F1	NewsQA EM / F1	HotpotQA EM / F1	SearchQA EM / F1
(I) Zero-shot target performance					
BERT-QA	14.30/23.57	15.38/25.90	39.17/56.14	43.34/60.42	16.19/25.03
(II) Target performance with domain adaptation					
DAT (Lee et al., 2019)	21.89/27.37	26.98/32.72	38.73/54.24	44.25/61.10	22.31/31.64
CASe (Cao et al., 2020)	20.77/29.37	25.40/35.85	43.43/59.67	47.16/63.88	26.07/35.16
CAQA* (Yue et al., 2021)	21.97/30.97	32.08/41.47	44.26/60.83	48.52/64.76	32.05/41.07
QC4QA KMeans (Ours)	25.04/33.20	35.53/44.32	44.40/60.91	49.58/65.78	34.44/43.78
QC4QA TREC (Ours)	**28.05/36.18**	**36.43/45.85**	**45.62/61.71**	**50.02/66.10**	**35.75/44.37**

Table 1: Main results of QA adaptation performance on target dataset.

Model	CoQA EM / F1	DROP EM / F1	Natural Questions EM / F1	TriviaQA EM / F1
(I) Zero-shot target performance				
BERT-QA	12.42/17.30	19.36/30.28	39.06/53.75	49.70/59.09
(II) Target performance with domain adaptation				
DAT (Lee et al., 2019)	11.98/14.72	18.53/29.34	44.94/58.91	49.94/59.82
CASe (Cao et al., 2020)	13.71/18.57	21.78/31.44	46.53/60.19	54.74/63.61
CAQA* (Yue et al., 2021)	14.41/19.28	22.48/31.56	47.37/60.52	54.30/62.98
QC4QA KMeans (Ours)	14.83/19.60	23.13/31.73	49.37/62.25	54.99/63.58
QC4QA TREC (Ours)	**15.03/19.71**	**23.46/32.22**	**50.59/62.98**	**55.98/64.57**

Table 2: Results of QA adaptation performance on additional target dataset.

the SQuAD dev set. Then, we perform adaptation experiments and report the main results in Table 1, results on the additional target datasets can be found in Table 2. Both tables are divided into 2 parts: (1) QA systems pretrained on SQuAD as naïve baseline ('Zero-shot target performance'); (2) Baseline methods and QC4QA for QA domain adaptation ('Target performance with domain adaptation'). The proposed approach with TREC supervised classification is denoted with 'QC4QA TREC', unsupervised KMeans question clustering is denoted with 'QC4QA KMeans'.

The following observations can be made from our experiments: (1) Unsupervised adaptation methods achieve superior performance than the naïve baseline in most cases. Compared to the naïve baseline, QC4QA can lead to improvements of over 100% in EM and and over 75% in F1. (2) Compared to contrastive adaptation (e.g., CAQA*), the proposed QC4QA is particularly effective on cloze questions (i.e., CNN and Daily Mail), with average performance gains of 16.5% and 10.4% in EM and F1. This suggests that we can benefit more from QC when the target questions are less similar to source questions. (3) By comparing CAQA* and both QC4QA methods, we find consistent performance improvements due to question classification for all datasets. (4) Both QC4QA methods outperform baseline methods with considerable improvements, from which QC4QA TREC demontrates the best performance on all datasets. For example, QC4QA KMeans significantly outperforms the best baseline CAQA* with 5.2% and 3.2% performance increases in EM and F1 on average. For QC4QA TREC, the relative improvements are 8.6% and 5.5% respectively. Altogether, the results suggest that both the TREC and KMeans question classification are effective for improving the performance on out-of-domain data. Additional results and analysis can be found in Appendix C.

4.4 Ablation Studies

4.4.1 Question Classification

We first study the benefits of question classification. The performance gains can be achieved by comparing the results between CAQA* and QC4QA in Table 1 and Table 2. This is because CAQA* is adapted to the same self-supervised adaptation framework as in QC4QA. CAQA* models are trained without minimizing the intra-class discrepancies in Equation (5). The improvements from QC4QA suggest that both supervised and unsupervised question classification can consistently improve QA systems in answering questions from unseen domains. Moreover, QC4QA is particularly effective on target datasets with different question formats (e.g., CNN and Daily Mail).

4.4.2 Distribution-aware Sampling

To study the influence of distribution-aware sampling in QC4QA, we replace the distribution-aware sampling strategy with random sampling. Then we perform unsupervised adaptation with QC4QA TREC on CNN, Daily Mail and NewsQA to verify the merits of the sampling strategy. Results are presented in Table 3.

Dataset	**Rand. Sampling** EM / F1	**QC4QA** EM / F1
CNN	26.69/35.24	**28.05/36.18**
Daily Mail	35.83/45.51	**36.43/45.85**
NewsQA	44.72/61.02	**44.86/61.40**

Table 3: QC4QA performance with the random sampling strategy.

In all target datasets, we see performance drops when we replace the proposed strategy with random sampling. In particular, we find relatively large performance deterioration on CNN without distribution-aware sampling. We believe the reason is that the question distribution in CNN is less similar to SQuAD (see Table 5), the resulting inconsistency in sampled batches reduces the effectiveness in discrepancy estimation.

4.4.3 Sensitivity of Hyperparameter λ

Now we evaluate the influence of λ to study the robustness of the proposed objective function. We select different values ranging from 0 to 5e-2 and perform adaptation with QC4QA TREC. Experiments on CNN, Daily Mail and NewsQA are presented to estimate the influence of λ.

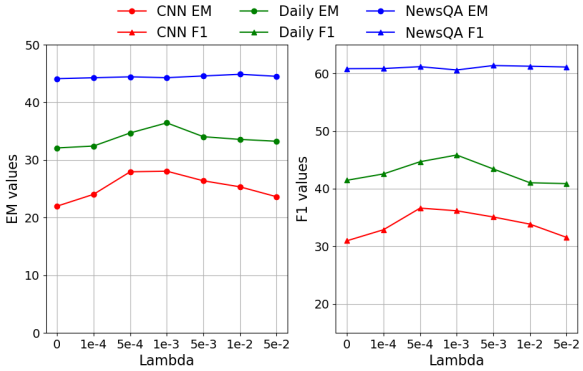

Figure 4: QC4QA adaptation performance for different lambda values. X-axis represents lambda and y-axis represents EM / F1 scores.

Figure 4 visualizes EM / F1 with increasing λ. Despite certain variations, we observe the results first go up and then decrease. Although CNN and Daily are more sensitive to λ, we observe greater improvements by reducing inter-domain and intra-class discrepancies. Overall, QC4QA consistently improves adaptation performance.

4.4.4 Confidence Threshold in Pseudo labeling

We study the influence of λ_{con} to understand how the performance varies with different confidence thresholds in pseudo labeling. We select different threshold values ranging from 0.2 to 0.8 to filter pseudo labels and train with QC4QA TREC. Experiments are performed on HotpotQA and SearchQA to estimate the influence of λ_{con}.

λ_{con} Selection	**HotpotQA** EM / F1	**SearchQA** EM / F1
0.2	48.12/64.56	25.60/34.50
0.4	**50.02/66.10**	33.56/42.62
0.6	49.95/69.84	35.75/44.37
0.8	49.44/65.11	**37.29/46.31**

Table 4: QC4QA adaptation performance for different confidence thresholds in pseudo labeling.

Table 4 shows adaptation performance with different λ_{con}. The best performance can be reached with λ_{con} ranging from 0.4 to 0.8. For large datasets like SearchQA (with over 100k QA pairs), a higher confidence threshold yields a better adaptation performance since we avoid noisy pseudo labels. In sum, carefully selected λ_{con} yields comparatively large improvements for QC4QA.

5 Conclusion

In this paper, we propose a novel framework for QA domain adaptation. The proposed QC4QA combines question classification with self-supervised adaptation techniques. QC4QA leverages question classes to reduce domain discrepancies and resemble target data distribution in training. Different from existing works, QC4QA achieves superior performance by introducing a simple question classifier and incorporating the question class information in the training objective. We demonstrate the efficiency and effectiveness of QC4QA compared to state-of-the-art approaches by achieving a substantially better performance on multiple datasets.

Despite having adopted question classification to adapt QA systems to unseen target domains, the proposed QC4QA has certain limitations. For example, we assume access to unlabeled questions in QA datasets and have not exploited the potential benefits of different question samples and question classes. For future work, we plan to relax our settings and explore question generation and question value estimation for QA domain adaptation.

Acknowledgments

This research is supported in part by the National Science Foundation under Grant No. IIS-2202481, CHE-2105005, IIS-2008228, CNS-1845639, CNS-1831669. The views and conclusions contained in this document are those of the authors and should not be interpreted as representing the official policies, either expressed or implied, of the U.S. Government. The U.S. Government is authorized to reproduce and distribute reprints for Government purposes notwithstanding any copyright notation here on.

References

Yu Cao, Meng Fang, Baosheng Yu, and Joey Tianyi Zhou. 2020. Unsupervised domain adaptation on reading comprehension. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pages 7480–7487.

Daniel Cer, Yinfei Yang, Sheng-yi Kong, Nan Hua, Nicole Limtiaco, Rhomni St. John, Noah Constant, Mario Guajardo-Cespedes, Steve Yuan, Chris Tar, Brian Strope, and Ray Kurzweil. 2018. Universal sentence encoder for English. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 169–174, Brussels, Belgium. Association for Computational Linguistics.

Alexis Conneau, Douwe Kiela, Holger Schwenk, Loïc Barrault, and Antoine Bordes. 2017. Supervised learning of universal sentence representations from natural language inference data. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 670–680, Copenhagen, Denmark. Association for Computational Linguistics.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, Minneapolis, Minnesota. Association for Computational Linguistics.

Dheeru Dua, Yizhong Wang, Pradeep Dasigi, Gabriel Stanovsky, Sameer Singh, and Matt Gardner. 2019. Drop: A reading comprehension benchmark requiring discrete reasoning over paragraphs. *arXiv preprint arXiv:1903.00161*.

Matthew Dunn, Levent Sagun, Mike Higgins, V Ugur Guney, Volkan Cirik, and Kyunghyun Cho. 2017. SearchQA: A new Q&A dataset augmented with context from a search engine. *arXiv preprint arXiv:1704.05179*.

Adam Fisch, Alon Talmor, Robin Jia, Minjoon Seo, Eunsol Choi, and Danqi Chen. 2019. MRQA 2019 shared task: Evaluating generalization in reading comprehension. In *Proceedings of the 2nd Workshop on Machine Reading for Question Answering*, pages 1–13, Hong Kong, China. Association for Computational Linguistics.

Arthur Gretton, Karsten M Borgwardt, Malte J Rasch, Bernhard Schölkopf, and Alexander Smola. 2012. A kernel two-sample test. *The Journal of Machine Learning Research*, 13(1):723–773.

Timothy J Hazen, Shehzaad Dhuliawala, and Daniel Boies. 2019. Towards domain adaptation from limited data for question answering using deep neural networks. *arXiv preprint arXiv:1911.02655*.

Karl Moritz Hermann, Tomas Kocisky, Edward Grefenstette, Lasse Espeholt, Will Kay, Mustafa Suleyman, and Phil Blunsom. 2015. Teaching machines to read and comprehend. *Advances in neural information processing systems*, 28:1693–1701.

Jeremy Howard and Sebastian Ruder. 2018. Universal language model fine-tuning for text classification. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 328–339, Melbourne, Australia. Association for Computational Linguistics.

Zhiheng Huang, Marcus Thint, and Zengchang Qin. 2008. Question classification using head words and their hypernyms. In *Proceedings of the 2008 Conference on Empirical Methods in Natural Language*

Processing, pages 927–936, Honolulu, Hawaii. Association for Computational Linguistics.

Mandar Joshi, Eunsol Choi, Daniel Weld, and Luke Zettlemoyer. 2017. TriviaQA: A large scale distantly supervised challenge dataset for reading comprehension. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1601–1611, Vancouver, Canada. Association for Computational Linguistics.

Amita Kamath, Robin Jia, and Percy Liang. 2020. Selective question answering under domain shift. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 5684–5696, Online. Association for Computational Linguistics.

Guoliang Kang, Lu Jiang, Yi Yang, and Alexander G Hauptmann. 2019. Contrastive adaptation network for unsupervised domain adaptation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 4893–4902.

Tom Kwiatkowski, Jennimaria Palomaki, Olivia Redfield, Michael Collins, Ankur Parikh, Chris Alberti, Danielle Epstein, Illia Polosukhin, Jacob Devlin, Kenton Lee, Kristina Toutanova, Llion Jones, Matthew Kelcey, Ming-Wei Chang, Andrew M. Dai, Jakob Uszkoreit, Quoc Le, and Slav Petrov. 2019. Natural questions: A benchmark for question answering research. *Transactions of the Association for Computational Linguistics*, 7:452–466.

Seanie Lee, Donggyu Kim, and Jangwon Park. 2019. Domain-agnostic question-answering with adversarial training. In *Proceedings of the 2nd Workshop on Machine Reading for Question Answering*, pages 196–202, Hong Kong, China. Association for Computational Linguistics.

Xin Li and Dan Roth. 2002. Learning question classifiers. In *COLING 2002: The 19th International Conference on Computational Linguistics*.

Mingsheng Long, Yue Cao, Jianmin Wang, and Michael Jordan. 2015. Learning transferable features with deep adaptation networks. In *International Conference on Machine Learning*, pages 97–105. PMLR.

John Miller, Karl Krauth, Benjamin Recht, and Ludwig Schmidt. 2020. The effect of natural distribution shift on question answering models. In *International Conference on Machine Learning*, pages 6905–6916. PMLR.

Pranav Rajpurkar, Jian Zhang, Konstantin Lopyrev, and Percy Liang. 2016. SQuAD: 100,000+ questions for machine comprehension of text. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 2383–2392, Austin, Texas. Association for Computational Linguistics.

Siva Reddy, Danqi Chen, and Christopher D. Manning. 2019. CoQA: A conversational question answering challenge. *Transactions of the Association for Computational Linguistics*, 7:249–266.

Siamak Shakeri, Cicero Nogueira dos Santos, Henghui Zhu, Patrick Ng, Feng Nan, Zhiguo Wang, Ramesh Nallapati, and Bing Xiang. 2020. End-to-end synthetic data generation for domain adaptation of question answering systems. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 5445–5460, Online. Association for Computational Linguistics.

Adam Trischler, Tong Wang, Xingdi Yuan, Justin Harris, Alessandro Sordoni, Philip Bachman, and Kaheer Suleman. 2016. Newsqa: A machine comprehension dataset. *arXiv preprint arXiv:1611.09830*.

Eric Tzeng, Judy Hoffman, Kate Saenko, and Trevor Darrell. 2017. Adversarial discriminative domain adaptation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 7167–7176.

Betty van Aken, Benjamin Winter, Alexander Löser, and Felix A Gers. 2019. How does bert answer questions? A layer-wise analysis of transformer representations. In *Proceedings of the 28th ACM International Conference on Information and Knowledge Management*, pages 1823–1832.

Zhilin Yang, Peng Qi, Saizheng Zhang, Yoshua Bengio, William Cohen, Ruslan Salakhutdinov, and Christopher D. Manning. 2018. HotpotQA: A dataset for diverse, explainable multi-hop question answering. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 2369–2380, Brussels, Belgium. Association for Computational Linguistics.

Xiang Yue, Xiaoman Pan, Wenlin Yao, Dian Yu, Dong Yu, and Jianshu Chen. 2022a. C-MORE: Pretraining to answer open-domain questions by consulting millions of references. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 371–377, Dublin, Ireland. Association for Computational Linguistics.

Xiang Yue, Ziyu Yao, and Huan Sun. 2022b. Synthetic question value estimation for domain adaptation of question answering. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1340–1351, Dublin, Ireland. Association for Computational Linguistics.

Zhenrui Yue, Bernhard Kratzwald, and Stefan Feuerriegel. 2021. Contrastive domain adaptation for question answering using limited text corpora. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 9575–9593, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Zhenrui Yue, Huimin Zeng, Ziyi Kou, Lanyu Shang, and Dong Wang. 2022c. Contrastive domain adaptation for early misinformation detection: A case study on covid-19. In *arXiv preprint arXiv:2208.09578*.

Huimin Zeng, Zhenrui Yue, Yang Zhang, Ziyi Kou, Lanyu Shang, and Dong Wang. 2022. On attacking out-domain uncertainty estimation in deep neural networks. In *IJCAI*.

Yabin Zhang, Hui Tang, Kui Jia, and Mingkui Tan. 2019a. Domain-symmetric networks for adversarial domain adaptation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 5031–5040.

Yang Zhang, Daniel Zhang, and Dong Wang. 2021. On migratable traffic risk estimation in urban sensing: A social sensing based deep transfer network approach. *Ad Hoc Networks*, 111:102320.

Yang Zhang, Ruohan Zong, Jun Han, Hao Zheng, Qiuwen Lou, Daniel Zhang, and Dong Wang. 2019b. Transland: An adversarial transfer learning approach for migratable urban land usage classification using remote sensing. In *2019 IEEE International Conference on Big Data (Big Data)*, pages 1567–1576. IEEE.

Yang Zhang, Ruohan Zong, and Dong Wang. 2020. A hybrid transfer learning approach to migratable disaster assessment in social media sensing. In *2020 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining (ASONAM)*, pages 131–138. IEEE.

A Dataset and baseline details

A.1 Dataset details

For QA datasets, we follow (Lee et al., 2019; Cao et al., 2020; Yue et al., 2021) and select SQuAD v1.1 as our source dataset (Rajpurkar et al., 2016). SQuAD is a crowdsourced QA dataset based on Wikipedia articles. For target domain, we adopt multiple datasets to evaluate QC4QA:

1. **CNN** (Hermann et al., 2015) leverages CNN articles as contexts. Cloze QA pairs are generated by replacing answers with '@placeholder'.

2. **CoQA** (Reddy et al., 2019) is a conversational dataset with rationales and QA pairs. Contexts are given as multi-turn conversations.

3. **Daily Mail** (Hermann et al., 2015) is similar to CNN and consists of news from Daily Mail. Cloze questions and answers are used.

4. **DROP** (Dua et al., 2019) requires QA systems to resolve references, reasoning, matching and understanding context implications.

5. **NewsQA** (Trischler et al., 2016) provides news as contexts and challenging questions beyond simple matching and entailment.

6. **HotpotQA** (Yang et al., 2018) provides multi-hop questions with challenging contexts (distractor contexts excluded).

7. **Natural Questions** (Kwiatkowski et al., 2019) has user questions. We adopt short answers and use long answers as contexts.

8. **TriviaQA** (Joshi et al., 2017) is a large-scale QA dataset that includes QA pairs and supporting facts for supervised training.

9. **SearchQA** (Dunn et al., 2017) is constructed through existing QA pairs by searching for context from online search results.

A.2 Baseline details

For naïve baseline, we adopt BERT-QA (uncased base version with additional batch normalization layer) and train on the source dataset (Devlin et al., 2019; Cao et al., 2020). Additionally, we select 3 baselines in unsupervised QA domain adaptation:

1. **Domain adversarial training (DAT)** (Lee et al., 2019) comprises of a QA system and a discriminator using `[CLS]` output in BERT. The QA system is first trained on labeled source data. Then, input data from both domains is used for domain-adversarial training to learn generalized features.

2. **Conditional adversarial self-training (CASe)** (Cao et al., 2020) leverages self-training with domain-adversarial learning. CASe iteratively perform self-training and domain adversarial training to reduce domain discrepancy. We adopt the entropy weighted version CASe+E in our work as baseline.

3. **Contrastive adaptation for QA (CAQA*)** (Yue et al., 2021) proposes contrastive adaptation based on token-level features. CAQA utilizes answer tokens as features and reduce the domain gap by minimizing MMD distances. We exclude question generation and adopt the same process of pseudo labeling, distribution-aware sampling and self-supervised adaptation. In particular, we perform training using the original contrastive adaptation loss as in (Yue et al., 2021).

B Implementation

We first train a question classifier on the TREC dataset. The QC model is trained for 4 epochs using RMSprop optimizer with learning rate of 0.01 and batch size of 64. We evaluate the QC model on the TREC test set and report the accuracy of the best QC model.

For pretraining BERT-QA on the source dataset (i.e., SQuAD), we follow (Devlin et al., 2019; Yue et al., 2021) to preprocess data and perform training. We select the AdamW optimizer and train BERT-QA for 2 epochs without linear warmup. Learning rate is 3e-5 and batch size is 12. After pretraining, we validate the model with the provided dev set and report the EM and F1 scores.

For baseline methods, we use our pretrained BERT-QA and follow their default settings for domain adaptation. For QC4QA, adaptation is performed 4 epochs with the AdamW optimizer, learning rate of 3e-5 and 10% proportion as warmup in training. In the pseudo labeling stage, we first perform inference on unlabeled target data and preserve the target samples above confidence threshold

λ_{con}. For batching in self-supervised adaptation stage, we sample 12 target examples and perform distribution-aware sampling to sample another 12 source QA pairs. The sampled source data has same question classes as the target examples. Validation is performed every 2000 iterations and after every epoch to save the best QA model. In our experiments, we empirically select λ from [1e-4, 1e-3, 1e-2], we select λ_{con} from [0.4, 0.6]. Our system setup is Intel Xeon Gold 6326 CPU, NVIDIA A40 GPU and 128GB RAM.

C Additional results

C.1 Question Classification Results

Due to our light-weight design, the TREC question classifier can perform training and inference efficiently within a few minutes. For example, we achieve an average training time of 57.6s on TREC in repeated experiments with GPU acceleration. Inference on QA datasets are of similar efficiency and depends on the individual size of each dataset.

Since TREC classes are not provided in QA datasets, it's not possible to directly evaluate the supervised QC model on them. We report the distribution of different question classes in Table 5, where we observe significant distribution shifts between the source dataset and certain target datasets (e.g., CNN and Daily Mail). Additionally, we present selected examples of classified questions in Table 9, from which we observe the following: (1) Cloze questions are more difficult to classify. Unlike natural questions, cloze questions usually do not contain auxiliary verb and wh-words (e.g., what, where etc.) as indicator of the question classes. (2) Multiple question classes may qualify for cloze questions. In some examples, different types of tokens can be filled in the placeholder position (e.g., both **DESC** and **ENTY** qualify for Q3). (3) The TREC question classifier can be less accurate on cloze questions. This is the case for Q5 in Table 9, where the questions are more likely to be **DESC** and **LOC** than **HUM**. (4) For natural questions, the question classifier performs generally well and makes fewer mistakes due to the similarity of natural questions across QA datasets. More examples can be found in the released code and data.

For KMeans unsupervised question classification, we focus on the discrepancy among question samples and perform KMeans clustering using the `[CLS]` output from BERT encoder, see Figure 5. The plot shows a principle component analysis (PCA) visualizing the BERT-encoder output of NewsQA examples, where different colors represent question class predictions via KMeans algorithm. We observe that `[CLS]` features are comparatively homogeneous, making it hard to determine cluster boundaries that clearly separate different classes of questions. This might cause performance deterioration in case of increasing outliers. Overall, KMeans can successfully cluster QA examples within each neighborhood on the target dataset. Ideally, the cluster labels can be used to reduce intra-class discrepancies for fine-grained domain adaptation similar to TREC classification. Both adaptation results and cluster visualization suggest that KMeans is effective in improving the performance on out-of-domain data.

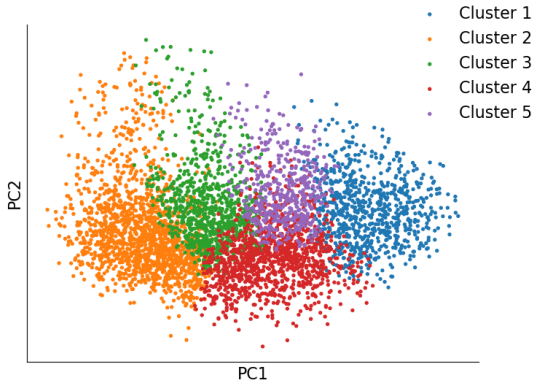

Figure 5: Visualization of KMeans cluster analysis on NewsQA using `[CLS]` features from BERT.

C.2 Comparison with CAQA

We study the effectiveness of the proposed QC4QA by comparing the performance between QC4QA, the original CAQA and the adapted CAQA* (Yue et al., 2021). The results are presented in Table 6, we observe that the best-performing method is QC4QA with TREC question classification for 7 out of 8 metric values. For SearchQA, the original CAQA performs the best in EM, with QC4QA TREC is of similar magnitude and clearly ranks second. On average, QC4QA TREC performs the best with EM of 48.09 and F1 of 59.51. Despite discarding question generation using the T5 transformer, QC4QA KMeans and the original CAQA performs similarly. Interestingly, we observe that CAQA* outperforms the original CAQA in HotpotQA, suggesting that the distribution-aware sampling and iterative pseudo-processing can effectively improve

Dataset	ABBR	DESC	ENTY	HUM	LOC	NUM
SQuAD	0.5%	12.5%	31.4%	19.9%	11.6%	24.1%
CNN	0.0%	5.3%	39.2%	43.6%	5.3%	6.5%
Daily Mail	0.0%	3.9%	38.6%	46.1%	4.4%	6.9%
NewsQA	0.1%	19.5%	21.4%	29.4%	10.3%	19.3%
HotpotQA	0.0%	1.6%	21.1%	51.6%	14.8%	10.9%
CoQA	0.1%	14.1%	14.2%	53.7%	9.3%	8.5%
DROP	0.0%	2.3%	24.0%	51.8%	7.3%	14.5%
Natural Questions	0.2%	5.0%	13.2%	39.2%	13.4%	29.0%
SearchQA	0.2%	3.7%	43.1%	4.8%	13.5%	34.7%
TriviaQA	0.2%	1.6%	37.8%	36.5%	19.3%	4.7%

Table 5: Question class distribution in all datasets.

Model	Natural Questions EM / F1	HotpotQA EM / F1	SearchQA EM / F1	TriviaQA EM / F1
(I) Zero-shot target performance				
BERT-QA	39.06/53.75	43.34/60.42	16.19/25.03	49.70/59.09
(II) Target performance with domain adaptation				
CAQA* (Yue et al., 2021)	47.37/60.52	48.52/64.76	32.05/41.07	54.30/62.98
CAQA (Yue et al., 2021)	48.55/62.60	46.37/61.57	**36.05**/42.94	55.17/63.23
QC4QA KMeans (Ours)	49.37/62.25	49.58/65.78	34.44/43.78	54.99/63.58
QC4QA TREC (Ours)	**50.59/62.98**	**50.02/66.10**	35.75/**44.37**	**55.98/64.57**

Table 6: Comparison between QC4QA and CAQA.

the adaptation performance.

C.3 Cluster Number in QC4QA KMeans Classification

To study the influence of cluster number in unsupervised question clustering for QC4QA, we initialize KMeans algorithm with different number of clusters. Then we perform QC4QA KMeans adaptation on HotpotQA and SearchQA to examine the influence of cluster number.

Cluster Number	HotpotQA EM / F1	SearchQA EM / F1
3	49.82/65.73	30.34/39.49
5	49.58/65.78	**32.19/41.27**
7	49.87/65.66	30.51/39.25
9	**50.45/66.14**	29.59/38.14

Table 7: QC4QA adaptation performance for different numbers of KMeans clusters.

Results are presented in Table 7, we observe performance drops when we reduce number of clusters from the default of 5. Surprisingly, the performance on HotpotQA grows consistently with increasing number of clusters. A potential explanation for such improvements is that fine-grained question classification is more helpful for complex multi-hop QA datasets.

C.4 Human Annotation

Training Method	Daily Mail EM / F1	NewsQA EM / F1
0 Annotation	36.43/45.85	44.86/61.40
5k Annotations	48.04/56.27	45.71/62.21
10k Annotations	55.37/61.95	47.17/63.46
20k Annotations	66.83/72.18	48.72/64.92

Table 8: Semi-supervised adaptation performance with QC4QA.

We also study the influence of human annotations by introducing labeled target examples. We present the results on Daily Mail and NewsQA in Table 8. We observe that human annotations improve the adaptation performance in general. With the increasing amount of annotations, the performance gains of QC4QA rise rapidly and then stay steady. In both cases, introducing limited annotations can significantly improve model performance. The results indicate that the introduction of even limited amount of annotations helps QA systems reach comparable magnitude of supervised results.

TREC classification examples
Q1: Judges in @placeholder and Oregon this week overturn marriage bans. → **DESC**
Q2: Spain international Mata close to joining English club @placeholder. → **ENTY**
Q3: The Surprise will be sold in 120 @placeholder stores, costing 1.75 for four? → **ENTY**
Q4: School bus drivers union will strike wednesday if it doesn't reach deal with @placeholder. → **HUM**
Q5: Serial killer Israel Keyes may have killed missing @placeholder woman. → **HUM**
Q6: Which is the latest version of corel draw? → **ENTY**
Q7: Who did say South Africa did not issue a visa on time? → **HUM**
Q8: Census bureaus are hiring people from where? → **LOC**
Q9: How long was the lion's longest field goal? → **NUM**
Q10: Musician and satirist Allie Goertz wrote a song about the "The Simpsons" character Milhouse, who Matt Groening named after who? → **HUM**
Q11: To whom did the Virgin Mary allegedly appear in 1858 in Lourdes France? → **HUM**
Q12: When did the Scholastic Magazine of Notre dame begin publishing? → **NUM**
Q13: The Basilica of the Sacred heart at Notre Dame is beside to which structure? **ENTY**
Q14: How often is Notre Dame's the Juggler published? **NUM**
Q15: Where is the headquarters of the Congregation of the Holy Cross? **LOC**
Q16: What is the oldest structure at Notre Dame? **ENTY**
Q17: Which organization declared the First Year of Studies program at Notre Dame "outstanding"? **HUM**
Q18: The College of Science began to offer civil engineering courses beginning at what time at Notre Dame? **HUM**
Q19: In what year was the College of Engineering at Notre Dame formed? **NUM**
Q20: Which prize did Frederick Buechner create? **ENTY**
Q21: What was the amount of children murdered? **NUM**
Q22: Where was one employee killed? **HUM**
Q23: What happened in Chad? **DESC**
Q24: What did one of John II's replacements do in captivity? **ENTY**
Q25: Who threw the first touchdown pass of the game? **HUM**
Q26: Which player scored touchdowns running and receiving? **HUM**
Q27: What all field goals did Olindo Mare make? **ENTY**
Q28: Which team had a safety scored on them in the first half? **HUM**
Q29: What was the difference between the role of blacks and whites in the draft? **DESC**
Q30: What was burned last: city of Ryazan or suburbs of Moscow? **LOC**

Table 9: Qualitative examples of classified questions in target datasets.

Prompt-based Conservation Learning for Multi-hop Question Answering

Zhenyun Deng, Yonghua Zhu, Yang Chen, Qianqian Qi
Michael Witbrock, Patricia Riddle
School of Computer Science, University of Auckland, New Zealand
{zden658, yzhu970, beryl413}@aucklanduni.ac.nz
{yang.chen, m.witbrock, p.riddle}@auckland.ac.nz

Abstract

Multi-hop question answering (QA) requires reasoning over multiple documents to answer a complex question and provide interpretable supporting evidence. However, providing supporting evidence is not enough to demonstrate that a model has performed the desired reasoning to reach the correct answer. Most existing multi-hop QA methods fail to answer a large fraction of sub-questions, even if their parent questions are answered correctly. In this paper, we propose the Prompt-based Conservation Learning (PCL) framework for multi-hop QA, which acquires new knowledge from multi-hop QA tasks while conserving old knowledge learned on single-hop QA tasks, mitigating forgetting. Specifically, we first train a model on existing single-hop QA tasks, and then freeze this model and expand it by allocating additional sub-networks for the multi-hop QA task. Moreover, to condition pre-trained language models to stimulate the kind of reasoning required for specific multi-hop questions, we learn soft prompts for the novel sub-networks to perform type-specific reasoning. Experimental results on the HotpotQA benchmark show that PCL is competitive for multi-hop QA and retains good performance on the corresponding single-hop sub-questions, demonstrating the efficacy of PCL in mitigating knowledge loss by forgetting.

1 Introduction

Multi-hop QA is a challenging task with the goals of reasoning over multiple scattered documents to predict an answer, and providing explanatory supporting evidence (Yang et al., 2018). By fine-tuning pre-trained language models (PLMs) with task-specific data, most existing multi-hop QA models have achieved good performance in both goals (Tu et al., 2020; Fang et al., 2020).

Despite the success of fine-tuned PLMs on the multi-hop QA task, providing supporting evidence is not enough to demonstrate that a multi-hop QA model has performed the desired multi-hop reasoning to reach the correct answer; it may instead have utilized reasoning shortcuts, having neglected to acquire and retain the single-hop reasoning knowledge essential to reliable interpretability (Jiang and Bansal, 2019). Previous work (Tang et al., 2021) has demonstrated that most existing multi-hop QA models with good performance fail to answer a large fraction of the sub-questions whose parent multi-hop questions can be answered correctly. Thus, it is necessary to understand the behaviour on each hop of the reasoning process and mitigate forgetting of the knowledge required for each hop in interpretable multi-hop QA. Doing so should enable humans to better trust the QA mechanism.

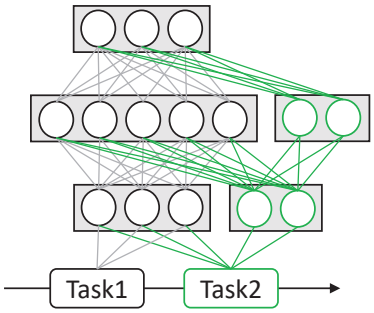

Figure 1: An example of conservation learning based on a continual learning mechanism. The neurons on the left are devoted to Task1 (single-hop QA), and on the right (green) are a novel sub-network created for Task2 (multi-hop QA) that laterally connects to the trained Task1. By adding the sub-network, the model acquires new knowledge of Task2 while retaining knowledge learned in Task1, mitigating forgetting.

In addition, existing QA models integrate all the knowledge by thoroughly pre-training the PLMs on all available data (Schwartz et al., 2020), which integrates the various forms of knowledge from multiple types of questions. However, a downstream QA task may only require knowledge of a specific type. For example, in the multi-hop QA task (Yang et al., 2018), questions can be roughly divided into two different types: bridging and comparison, each of which requires a specific reasoning strategy to answer. To achieve multi-hop reasoning efficiently,

Proceedings of the 29th International Conference on Computational Linguistics, pages 1791–1800
October 12–17, 2022.

it may be useful for PLMs to disentangle knowledge from other question types and stimulate the appropriate reasoning types required for particular multi-hop questions.

To address these issues, we propose Prompt-based Conservation Learning (PCL) for multi-hop QA. Specifically: i) to train a multi-hop QA model without forgetting, we apply conservation learning based on a continual learning mechanism to acquire new knowledge from multi-hop QA tasks while retaining that previously learned on single-hop QA tasks. As shown in Figure 1, we first train a model on the single-hop QA task; when incorporating the new multi-hop QA task, we freeze the model trained on the single-hop task and expand it by allocating novel sub-networks for new multi-hop knowledge; ii) to take full advantage of diverse knowledge in the PLM, we first identify the reasoning type of the multi-hop question as a soft prompt via a transformer-based question classifier, and then transform it into a sub-network that connects laterally with the previously trained QA model, to condition the PLM to perform type-specific reasoning. Since PCL trains the QA model incrementally based on the conserved previously learned parameters, it should be able to perform well on multi-hop QA because it thus retains the previously learned knowledge (Parisi et al., 2019; Sun et al., 2020).

Our contributions are summarized as follows:

- We propose conservation learning for multi-hop QA, which acquires knowledge from the multi-hop QA task while retaining knowledge learned on single-hop QA tasks, which may enable humans to understand the behaviour of each hop in the reasoning process better.
- We propose using a soft prompt based on the reasoning type to condition the PLM, stimulating use of the required knowledge for particular types of multi-hop reasoning.
- Our proposed PCL achieves better performance on the HotpotQA leaderboard, while also retaining good performance on the corresponding single-hop sub-questions.

2 Related Work

Prompt Tuning for PLMs. Prompt tuning is an effective mechanism for learning prompts to condition PLMs to stimulate and apply the appropriate knowledge for a specific downstream task (Liu et al., 2021). Gu et al. (2021) propose to initialize soft prompts by adding them into the pre-training stage of few-shot learning. Li and Liang (2021) prepend a series of learnable continuous embeddings as soft prompts into the input, achieving better performance in text generation tasks. Motivated by these methods, we use the reasoning types of multi-hop questions as soft prompts to condition PLMs to stimulate the knowledge required to answer multi-hop questions.

Continual Learning for PLMs. Continual learning aims to allow systems to repeatedly acquire new knowledge while retaining previously learned experience, mitigating catastrophic forgetting (Parisi et al., 2019). Conceptually, continual learning can be divided into three categories of technique: i) retrain the whole model while imposing additional constraints to retain the important learned model parameters from previous tasks (Li et al., 2021a); ii) perform memory replay to distill the knowledge from previous model backups (Sun et al., 2019; Rolnick et al., 2019); iii) freeze the model trained on previous tasks and retrain the model by allocating new neurons or network layers for new tasks (Qin et al., 2022). In this paper, we propose a learning mechanism based on continual learning, by freezing the model trained on the single-hop QA, and retraining the model for the multi-hop QA using our soft-prompt technique, enables the QA model to achieve single-hop reasoning and multi-hop reasoning simultaneously. Since we only have two tasks, we call this conservation learning. It aims to conserve previously learned knowledge while performing well on a second task; it does not continue for a large number of tasks as in continual learning.

End-to-end Multi-hop QA. Existing end-to-end multi-hop QA systems predict the answer and corresponding supporting facts based on the given question and retrieved relevant paragraphs. Qiu et al. (2019), Fang et al. (2020), and Tu et al. (2020) extract information at different levels of granularity as nodes in a graph, and then apply GNN-based methods to answer the question and provide supporting sentences. Shao et al. (2020a), Beltagy et al. (2020) and Wu et al. (2021) argue that graph structures may not be necessary for multi-hop QA, and propose graph-free reasoning models. Unlike these methods, where there is no training requirement for the models to follow the desired reasoning steps to predict the answer, we propose a multi-hop

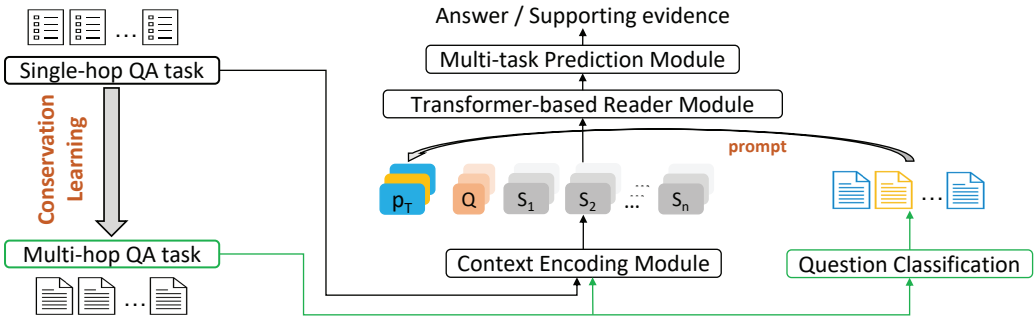

Figure 2: An overview of our proposed PCL framework for multi-hop QA. Specifically, it involves three key steps, (a) train a QA model to acquire knowledge from single-hop QA tasks; (b) identify reasoning types of multi-hop questions as soft prompts, and transform soft prompts into a sequence of continuous type-specific vectors; (c) retrain the QA model to acquire new knowledge from multi-hop QA tasks by freezing the trained network in single-hop QA task and prepending soft prompt vectors to the input.

QA framework with separated learning of the intended behaviour of QA models on each hop of the reasoning process and in the final answer.

3 Methodology

3.1 Overview

This section, we describe prompt-based conservation learning for multi-hop QA. As illustrated in Figure 2, our PCL consists of three components: i) we first acquire single-hop QA knowledge by explicitly training on these tasks; ii) we then acquire knowledge for the new multi-hop QA task while retaining the learned knowledge using conservation learning; iii) we perform type-specific reasoning, identifying the reasoning type of the question via the soft prompt to stimulate application of the appropriate knowledge.

3.2 Single-hop QA

To understand the behaviour of existing QA models on each hop of the reasoning process, we train a QA model based on the PLM, ELECTRA (Clark et al., 2020) on a single-hop QA task, SQuAD (Rajpurkar et al., 2016). This QA model contains two modules: context encoding and a transformer-based reader.

Context Encoding. Given a question Q and n relevant sentences, we concatenate the question and sentences into an input sequence for the pre-trained ELECTRA encoder to obtain a context representation. Specifically, we formulate the input sequence as "[CLS] Q [SEP] yes no [SEP] [SE] s_1 [SEP] [SE] s_2 [SEP] ... [SE] s_i [SEP]... [SE] s_n [SEP]", where [SE] is a special token delineating supporting evidence, and yes no indicates a yes/no answer, which are prepended to the context, subsequently encoded by ELECTRA into the context representation. Consequently, each context sentence s_i in the input sequence can interact with other sentences across the concatenated sequence by using a self-attention mechanism; such interactions are crucial for multi-hop QA (Zhu et al., 2021).

Transformer-based Reader. After context encoding, the context representations are passed through a bi-attention layer to enhance interactions between the question and the context (Qiu et al., 2019). On top of the updated context representation, we have followed (Fang et al., 2020) to design a multi-task prediction module to jointly perform answer and supporting evidence prediction. For answer span prediction, we use two linear layers applied to the context representation to predict the start and end position of the answer. For supporting evidence prediction, we use a binary linear layer to predict a binary relevance label at each sentence start [SE]. The final objective is defined as:

$$\mathcal{L}_{Joint} = \mathcal{L}_{start} + \mathcal{L}_{end} + \lambda_1 \mathcal{L}_{SE}$$

where λ_1 is a hyper-parameter and each loss function $\mathcal{L}$ is the cross-entropy loss between the prediction and ground truth.

3.3 Multi-hop QA with Conservation Learning

Origins in Continual Learning. In one form of Continual Learning, given N existing tasks $\mathcal{T}_{seq} = \{\mathcal{T}_0, \mathcal{T}_1, ..., \mathcal{T}_N\}$, when a new task $\mathcal{T}_{N+1}$ comes, an additional network is created and the lateral connections with the trained model are learned. To avoid knowledge forgetting, the parameters θ^N learned by the existing tasks $\mathcal{T}_{seq}$ remain unchanged while the new parameter set θ^{N+1} is learned for the additional network in Task $\mathcal{T}_{N+1}$ (Parisi et al., 2019).

Conservation Learning for Multi-hop QA. To enable a trained single-hop QA model to learn the

new knowledge required for a subsequent multi-hop QA task without forgetting previously learned knowledge, we propose a truncated-continual-learning-like method that freezes the learned model and allocates additional sub-networks for the new multi-hop QA tasks. In principle this process could be iterated in continual learning, but here we apply one such step, and coin the term "conservation learning" to describe it. PCL's multi-hop QA after conservation learning consists of three components: i) question classification: identifying the reasoning type of the multi-hop question; ii) paragraph selection: retrieving paragraphs related to the multi-hop question; iii) pre-trained soft prompt: conditioning a PLM to perform the type-specific reasoning required for a multi-hop question.

Question Classification. Instead of training a separate QA model for each reasoning type, our design uses a single PLM to integrate the knowledge from all reasoning types. To inform this use, we first need to identify the reasoning type of the multi-hop question. Thus, we train a question classifier, also based on ELECTRA, followed by a binary classification layer, to predict the reasoning type for each multi-hop question. The question classifier only takes the question as its input and outputs a relevance score for different reasoning types. The reasoning type with the highest score is selected as the type of multi-hop question.

Iterative Paragraph Selection. Since not every given paragraph contains relevant information, multi-hop QA models must filter out irrelevant paragraphs. In addition, multi-hop questions also often permit reasoning shortcuts through which QA models can directly locate the final answer by word-matching the question to a single sentence in the paragraph (Qi et al., 2019, 2020). To discourage this kind of direct but unjustified leap to the answer, we propose to retrieve paragraphs related to the question in an iterative fashion, which encodes the question and previously retrieved paragraphs as a new question vector to retrieve the next relevant paragraph. For simplicity, we use the same model encoder as the question classifier to select relevant paragraphs, except that we take the question q and the paragraph p as the input and output a relevance score for each paragraph. We calculate the score for each paragraph at each retrieval step as follows:

$$\mathcal{P}(P_{seq}|q) = \prod_{t=1}^{n} \mathcal{P}(p_t|q, p^1, p^2, ..., p^{t-1})$$

where for $t = 1$ (*i.e.,* the first hop), we only use the original question q for paragraph retrieval. At each subsequent retrieval step, we encode the question q and the most relevant paragraph p^{t-1} in the previous step t as a new question vector to predict the next relevant paragraph. In this way, each subsequent retrieved paragraph is not only related to the question, but also related to the previous retrieved paragraphs, which discourages producing an answer using "reasoning" shortcuts and provides a solid basis for multi-hop reasoning in the next step.

Pre-training Soft Prompt. To enable the PLM to integrate knowledge from multiple reasoning types, we introduce a soft prompt based on the reasoning type to condition the PLM to perform type-specific reasoning, which is connected laterally to the trained QA model during training. Specifically, we first formulate the input sequence as "[CLS] Q [SEP] yes no [SEP] [SE] s_1^1 [SEP] [SE] s_1^2 [SEP] ... [SE] $\mathrm{s}_\mathrm{i}^\mathrm{j}$ [SEP]... [SE] $\mathrm{s}_\mathrm{n}^\mathrm{m}$ [SEP]", where $\mathrm{s}_\mathrm{i}^\mathrm{j}$ indicates the j-th sentence in the relevant paragraph i; we then utilize the previously trained model to initialize the input sequence to obtain the context representation $\mathbf{C} = \{\mathbf{c}_0, \mathbf{c}_1, ..., \mathbf{c}_{n-1}\} \in \mathbb{R}^{n \times d}$, where n, d are the length and the dimension of the context, respectively; we finally transform the reasoning type obtained in the question classification into a continuous trainable vector $\mathbf{p} \in \mathbb{R}^{m \times d}$ and prepend it onto $\mathbf{C}$, resulting in the new input $\mathbf{C}' = \{\mathbf{p}_i; \mathbf{c}_0, \mathbf{c}_1, ..., \mathbf{c}_{n-1}\}$, where m is the length of the soft prompt and $\mathbf{p}_i$ is the soft prompt vector of reasoning type i.

Once the new context representation is obtained, it is then processed by the transformer-based reader module. Notably, we optimize $\mathbf{p}_i$ along with other parameters of the PLM during pre-training. During fine-tuning, we prepend the trained soft prompt vector into the input sequence, guiding the model to perform type-specific reasoning. In this way, we condition the PLM to stimulate the proper knowledge required for multi-hop reasoning.

4 Experiments

4.1 Dataset and Metrics

We evaluate our model primarily on three datasets: HotpotQA (Yang et al., 2018), adversarial HotpotQA (Jiang and Bansal, 2019) and a manually verified sub-question QA dataset generated from HotpotQA (Tang et al., 2021). To verify whether our PCL can be generalized to other multi-hop QA

Model	Ans		Sup		Joint	
	EM	F1	EM	F1	EM	F1
Baseline Model (Yang et al., 2018)	45.60	59.02	20.32	64.49	10.83	40.16
DecompRC (Min et al., 2019)	55.20	69.63	-	-	-	-
OUNS (Perez et al., 2020)	66.33	79.34	-	-	-	-
QFE (Nishida et al., 2019)	53.86	68.06	57.75	84.49	34.63	59.61
DFGN (Qiu et al., 2019)	56.31	69.69	51.50	81.62	33.62	59.82
SAE-large (Tu et al., 2020)	66.92	79.62	61.53	86.86	45.36	71.45
C2F Reader (Shao et al., 2020b)	67.98	81.24	60.81	87.63	44.67	72.73
Longformer (Beltagy et al., 2020)	68.00	81.25	63.09	88.34	45.91	73.16
HGN-large (Fang et al., 2020)	69.22	82.19	62.76	88.47	47.11	74.21
AMGN (Li et al., 2021b)	70.53	83.37	63.57	88.83	47.77	75.24
S2G (Wu et al., 2021)	70.72	83.53	64.30	88.72	48.60	75.45
PCL (Ours)	**71.76**	**84.39**	**64.61**	**89.20**	**49.27**	**76.56**

Table 1: Results on the blind test set of HotpotQA in the distractor setting. Our PCL achieves the best performance on the HotpotQA leaderboard. "-" denotes the case where no results are available. Leaderboard: https://hotpotqa.github.io/.

datasets, we also conduct experiments on two similar datasets: 2WikiMultihopQA (Ho et al., 2020) and MuSiQue (Trivedi et al., 2021). Unlike other knowledge-based multi-hop QA datasets (Welbl et al., 2018; Talmor and Berant, 2018; Saxena et al., 2020) that restrict the final answer to the content of explicit knowledge bases, all QA pairs in the HotpotQA are collected from Wikipedia.

HotpotQA. Each multi-hop question is provided with ground truth answers and supporting sentences, which enables us to evaluate the performance and interpretability of multi-hop reasoning. There are two reasoning types of questions: bridging and comparison, each of which requires a specific reasoning strategy to answer.

Sub-question QA dataset. To analyze whether the multi-hop QA models really perform each hop of the reasoning process, Tang et al. (2021) generate a single-hop sub-question dataset with 1000 manually verified samples for the dev set of HotpotQA for evaluation.

Adversarial HotpotQA. Multi-hop questions in the HotpotQA often contain reasoning shortcuts through which models can directly find the answer by word-matching the question to a sentence. To avoid this, Jiang and Bansal (2019) construct adversarial samples by creating contradicting answers to reasoning shortcuts without affecting the validity of the original answers.

Multi-hop QA Dataset. Unlike HotpotQA, 2WikiMultihopQA evaluates the interpretability of the multi-hop QA model not only with supporting evidence, but also with entity-relation tuples. However, for a fair comparison, we do not use the entity-relation tuples in our training. MuSiQue has richer multi-hop questions with 2-4 hops.

Metrics. We use Exact Match (EM) and Partial Match (F1) to evaluate the model performance on answer and supporting facts prediction, and a joint EM and F1 score to evaluate the final performance.

4.2 Implementation Details

We adopt ELECTRA-large (Clark et al., 2020) as the skeleton for each module. Our released implementation is based on Huggingface (Wolf et al., 2020). For question classification and paragraph selection, we train the models for 5 epochs using Adam optimizer, with a batch size of 12, a learning rate of 2×10^{-5}, a warm-up rate of 0.1 and ℓ_2 weight decay of 0.01. For question answering, we use the same setting as stated above, except for a learning rate of 3×10^{-5} and an additional prompt length of 2 tokens. The hyper-parameter of λ_1 is set to 2. Only the context encoding module is frozen during Conservation Learning and additional weights are added to connect the soft prompts.

4.3 Main Results

We compare our PCL model with other published baselines on the test set of HotpotQA in the distractor setting. As shown in Table 1, we observe that our PCL QA-system outperforms all comparison baselines on every metric and achieves the best performance on the HotpoQA dataset, demonstrating the progress made by PCL in multi-hop QA. Specifically, under the same setting, using a transformer-based ELECTRA model, PCL achieves a 1.12/0.91 improvement on the Joint EM/F1 score, compared with the best graph-free model S2G. This indicates that the effectiveness of the proposed conservation learning and soft prompts. For the best graph-based

q	q_{sub1}	q_{sub2}	DFGN	DecompRC	HGN	PCL
c	c	c	23	31.3	39.5	43.6
c	c	w	9.7	7.2	5.1	6.8
c	w	c	17.9	19.1	19.6	21.3
c	w	w	7.5	5.5	3.8	2.1
w	c	c	4.9	3	2.8	1.7
w	c	w	17	18.6	16.7	16.3
w	w	c	3.5	3.4	2.6	1.1
w	w	w	16.5	11.9	9.9	7.1

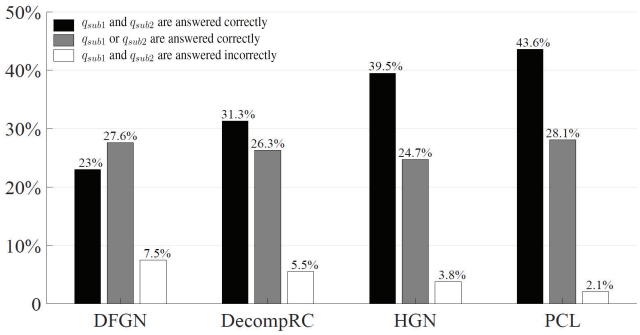

Table 2: (Left) Categorical EM statistics (%) of sub-question evaluation for four multi-hop QA models. c/w denotes that the question is answered correctly/wrongly. For example, the first four rows show the percentage of multi-hop questions that can be correctly answered. (Right) The success rate of four multi-hop QA models.

Model	Ans F1	Sup F1	Joint F1
ELECTRA	81.05	89.97	73.89
- Prompt	82.06	90.36	75.02
- CL	82.99	90.97	76.39
PCL	**84.42**	**91.15**	**77.76**

Table 3: Ablation Study of PCL on the dev set of HotpotQA. Prompt denotes that a soft prompt is used to condition PLM ELECTRA to stimulate the reasoning required for the multi-hop question. CL denotes that conservation learning is used to perform multi-hop reasoning. PCL used both soft prompts and conservation learning.

Model	Accuracy
DecompRC	70.40
QC(ELECTRA-large)	**98.97**

Table 4: The performance of question classification by different models. QC(ELECTRA-large) is a question classifier based on ELECTRA-large.

model AMGN, PCL improves the Joint EM/F1 score by 1.5/1.32, which shows that good performance can be achieved without constructing a graph. In the next section, we provide a detailed analysis to evaluate the performance of conservation learning and soft prompts in our PCL model.

4.4 Ablation Studies

To verify the effect of the components in our PCL model, we perform the following ablation studies on the dev set of HotpotQA.

Effect of Conservation Learning (CL). To verify the effect of conservation learning on multi-hop QA, we compare performance with the PLM ELECTRA with and without conservation learning. For conservation learning, we first trained an ELECTRA-based QA model on the single-hop QA dataset SQuAD (Rajpurkar et al., 2016), and then retrained it on the HotpotQA dataset with conservation learning. As shown in Table 3, we observe that the overall performance (F1 score) increased from 73.89 to 76.39 after using conservation learning, which shows that our model performs well on multi-hop reasoning when the previously learned knowledge is retained. In the following Section 4.6, we provide an in-depth analysis on the performance of our model on the sub-questions, to compare the ability of models to mitigate forgetting.

Effect of Soft Prompts. To verify the effect of the soft prompt and perform type-specific reasoning, we first identified the reasoning type of the multi-hop question using a classifier based on ELECTRA. In Table 4, our classifier $\text{QC}_{\text{ELECTRA}}$ achieves good accuracy compared to DecompRC, providing a solid basis for type-specific multi-hop reasoning. Then, we transform the identified reasoning type into a soft prompt to stimulate the PLM to perform the corresponding type of multi-hop reasoning. In Table 3, we implant the soft prompt both into the baseline ELECTRA and the ELECTRA based on conservation learning (PCL), the Joint F1 score improved by 1.23 and 1.37, respectively. This suggests that the soft prompt based on the reasoning type can stimulate the question-type-specific reasoning knowledge required for multi-hop QA.

Effect of Pre-trained Language Model. To verify the effects of PLMs, we compare PCL with HGN based on the same data and backbone. As shown in Table 5, PCL outperforms HGN on all metrics. This indicates the effectiveness and robustness of PCL across PLMs.

4.5 Evaluation across Reasoning Types

We evaluate the performance of PCL for multi-hop questions with multiple reasoning types. Specifically, we follow HGN in splitting the multi-hop questions into three categories: bridge, comparison-yes/no and comparison-span. "Bridge" questions require identifying a bridge entity to infer the answer, "comparison-yes/no" and "comparison-span"

Model	Ans F1	Sup F1	Joint F1
HGN(RoBERTa)	82.22	88.58	74.37
HGN(ELECTRA)	82.24	88.63	74.51
HGN(ALBERT)	83.46	89.2	75.79
PCL(RoBERTa)	84.33	90.75	77.12
PCL(ELECTRA)	84.42	91.15	77.76
PCL(ALBERT)	85.47	91.28	78.76

Table 5: Results with different PLMs on the dev set of HotpotQA. RoBERTa, ELECTRA and ALBERT denote that we use RoBERTa-large, ELECTRA-large and ALBERT-xxlarge-v2 as the PLM respectively.

Model	Question	Ans F1	Sup F1	Joint F1
HGN	bridge	81.90	87.60	73.31
	comp-yn	93.45	94.22	88.5
	comp-span	79.06	91.72	74.17
PCL	bridge	85.36	90.77	78.17
	comp-yn	93.67	94.73	88.93
	comp-span	82.42	92.65	77.57

Table 6: Results with different reasoning types on the dev set of HotpotQA. PCL outperforms HGN in all reasoning types.

require comparing two entities to infer the answer that could be yes/no or a span of text. As shown in Table 6, our PCL performs better than HGN for all reasoning types, indicating that the performance of the model can be effectively improved by using soft prompts for type-specific reasoning.

4.6 Evaluation of Robustness

In this section, we evaluate the robustness and generalization of PCL on three different datasets.

Evaluation on Sub-question Dataset. To analyze whether existing multi-hop QA models could at least in principle perform the multi-hop reasoning process by composing an answer out of solved sub-questions, we perform an evaluation on 1000 human-verified examples (Tang et al., 2021). These data consist of 1000 multi-hop questions q, and the corresponding 1000 sub-questions q_{sub1}, q_{sub2}. EM and F1 are used in each case to evaluate performance on answer prediction. As shown in Table 7, PCL achieves the best performance on the 1000 human-verified examples. Compared to DFGN and DecompRC, whose performance significantly drops on sub-questions, especially on the second sub-questions. PCL dropped by only 2.4 on average, which demonstrates that PCL can in principle support the expected behaviour on each hop of the reasoning process better than other multi-hop QA models by mitigating knowledge forgetting.

To further analyze whether models effectively mitigate knowledge forgetting, we collect the correctness statistics on each example in the sub-question dataset. As shown in Table 2 (Left), PCL has a 96.25% chance of getting the parent multi-hop question q right when both sub-questions q_{sub1} and q_{sub2} are answered correctly, which indicates that PCL can better retain the learned knowledge, through its use of conservation learning, compared with other multi-hop QA models. However, we observe that PCL still has a high probability of answering the parent multi-hop question correctly when only one of the sub-question is answered correctly. We summarize the sub-question dependent success rate of multi-hop QA models in Table 2 (Right). We observe that these models can answer parent multi-hop questions with a high probability (exceeding 20%) when only one sub-question is answered correctly, which indicates that using potentially unsound reasoning shortcuts to predict answers is a common and difficult to avoid phenomenon in multi-hop QA.

Model	q		q_{sub1}		q_{sub2}	
	EM	F1	EM	F1	EM	F1
DFGN	58.1	71.96	54.6	68.54	49.3	60.83
DecRC	63.1	77.61	61.0	75.21	56.8	70.77
HGN	71.0	84.25	66.1	81.72	66.7	78.24
PCL	**73.8**	**87.15**	**68.4**	**83.62**	**68.5**	**81.07**

Table 7: Results on the sub-question dataset with different multi-hop QA models. q denotes the multi-hop question, q_{sub1} and q_{sub2} denote the corresponding sub-questions of q.

Train	Reg		Reg	
Eval	Reg		Adv	
Model	EM	F1	EM	F1
HGN	47.31	74.37	41.56	69.81
PCL	49.59	77.76	47.87	74.24

Table 8: EM and F1 scores after evaluating on the adversarial dataset designed to probe for the use of unsound reasoning shortcuts. Reg or Adv denotes training or evaluating the model on the standard or adversarial HotpotQA dataset.

Evaluation on Adversarial Dataset. To compare the extent to which models are currently able to avoid the unsound-reasoning-shortcut problem, we conducted an adversarial evaluation on the dev set of HotpotQA, reported in Table 8. In the adversarial examples, the fake answers are sampled from the original HotpotQA dataset, but do not affect the validity of the original answers. As shown in Table 8, we trained PCL and HGN on the standard training data and evaluated them on both the standard and adversarial dev data. The result shows that PCL achieves better performance than HGN, indicating that PCL is more robust than HGN against the use of shortcuts probed by the adversarial dataset.

Evaluation on Other Multi-hop Datasets. To verify whether PCL can generalize to other multi-

Question	Answer	Answer pred by PCL	Answer pred by HGN
(Bridge) Q1: Who directed the film about the living funeral for Morrie Schwartz?	Mick Jackson	Mick Jackson	Mick Albom
$Q1_{sub1}$: Which film is about the living funeral for Morrie Schwartz?	Tuesdays with Morrie	Tuesdays with Morrie	Tuesdays with Morrie
$Q1_{sub2}$: Who directed Tuesdays with Morrie?	Mick Jackson	Mick Jackson	Mick Jackson
(Comp) Q2: Are local H and For Against both from the United States?	Yes	Yes	Illinois
$Q2_{sub1}$: Where does Local H from?	Illinois	Illinois	Illinois
$Q2_{sub2}$: Where does For Against from?	Nebraska	Nebraska	Nebraska

Question: What was the job of the character Jack Nicholson played in a 1992 French-American biographical crime film directed by Danny DeVito?
Answer: Teamsters leader

Supporting fact1: Jack Nicholson plays Hoffa, and DeVito plays Robert Ciaro, an amalgamation of several Hoffa associates over the years.
Supporting fact2: **Hoffa** is a 1992 French-American biographical crime film directed by Danny DeVito and written by David Mamet, based on the life of **Teamsters leader** Jimmy Hoffa.
Adversarial fact: **Sweet Revenge** is a 1992 French-American biographical crime film directed by Danny DeVito and written by David Mamet, based on the life of **Dandy** Jimmy Hoffa.

Answer predicted by PCL: Teamsters leader
Answer predicted by HGN: Dandy

Figure 3: Case studies of the sub-question evaluation and adversarial multi-hop question evaluation. The upper case study indicates that our PCL has stronger composite reasoning ability compared to HGN. The lower case study indicates that the iterative paragraph selection is help to avoid predict the answer by using reasoning shortcuts.

	2WikiMultihopQA		MusiQue	
	EM	F1	EM	F1
HGN	38.74	68.69	39.42	65.12
PCL	46.03	73.42	41.28	67.34

Table 9: Results of PCL and HGN on 2WikiMultihopQA and MusiQue multi-hop QA dataset.

hop QA datasets, we compared PCL against HGN on the 2WikiMultihopQA and MuSiQue dataset. In Table 9 we observe that PCL outperforms HGN on these two datasets, which demonstrates PCL's good potential on generalisation to QA problems with more than 2 hops.

4.7 Case Study

We present two case studies in Figure 3. The upper case illustrates the results of PCL and HGN at each hop of the reasoning process. We observe that PCL correctly answered the bridge question Q1, while HGN did not, when all sub-questions were answered correctly, supporting the claim that PCL learns new QA knowledge while retaining knowledge learned for sub-questions. Similarly, for comparison question Q2, PCL learned the specific reasoning ability based on the reasoning type to which Q2 belongs, indicating soft prompts based on reasoning types can elicit the reasoning knowledge required for multi-hop questions.

The lower case illustrates the results of PCL and HGN on an adversarial multi-hop question. In the example, the question can be directly answered by matching a reasoning shortcut in supporting facts2 *"a 1992 French-American biographical crime film directed by Danny DeVito"*. To avoid it, we follow (Jiang and Bansal, 2019) to construct an adversarial fact from the candidate paragraphs by replacing the subject and the answer, *e.g.,* "*Sweet Revenge*" for "*Hoffa*" and "*Dandy*" for "*Teamsters leader*". We observed that PCL correctly answered the question despite the interference from the adversarial fact, while HGN did not. This supports the claim that the iterative paragraph selection helps establish connections between supporting facts, because PCL selects the next supporting fact2 based on the previous supporting fact1. In this example, the adversarial fact is irrelevant to supporting fact1, so PCL excludes it during paragraph selection.

5 Conclusions and Future Work

In this paper, we introduce a novel prompt-based conservation learning framework for multi-hop QA – a framework that retains knowledge from previous component tasks – able to answer questions in a principled way that matches human expectations by answering sub-questions and integrating the answers. By developing soft prompts related to reasoning types during training, we also show that we can condition PLMs to stimulate and apply the reasoning knowledge required for specific multi-hop questions. Experimental results on multiple multi-hop QA datasets demonstrate the improved performance of PCL over previous multi-hop QA models in multi-hop QA.

Next, we plan to extend PCL on QA problems with arbitrary hop-counts, and to increase generality by extending soft prompts to handle QA with unrestricted numbers of, and implicit, reasoning types, and non-linear reasoning structures.

6 Acknowledgments

This work was supported by a grant from the New Zealand Tertiary Education Commission and by the Strong AI Lab at the University of Auckland.

References

Iz Beltagy, Matthew E Peters, and Arman Cohan. 2020. Longformer: The long-document transformer. *arXiv preprint arXiv:2004.05150*.

Kevin Clark, Minh-Thang Luong, Quoc V Le, and Christopher D Manning. 2020. Electra: Pre-training text encoders as discriminators rather than generators. *arXiv preprint arXiv:2003.10555*.

Yuwei Fang, Siqi Sun, Zhe Gan, Rohit Pillai, Shuohang Wang, and Jingjing Liu. 2020. Hierarchical graph network for multi-hop question answering. In *EMNLP*, pages 8823–8838.

Yuxian Gu, Xu Han, Zhiyuan Liu, and Minlie Huang. 2021. Ppt: Pre-trained prompt tuning for few-shot learning. *arXiv preprint arXiv:2109.04332*.

Xanh Ho, Anh-Khoa Duong Nguyen, Saku Sugawara, and Akiko Aizawa. 2020. Constructing a multi-hop qa dataset for comprehensive evaluation of reasoning steps. *arXiv preprint arXiv:2011.01060*.

Yichen Jiang and Mohit Bansal. 2019. Avoiding reasoning shortcuts: Adversarial evaluation, training, and model development for multi-hop qa. *arXiv preprint arXiv:1906.07132*.

Haoran Li, Aditya Krishnan, Jingfeng Wu, Soheil Kolouri, Praveen K Pilly, and Vladimir Braverman. 2021a. Lifelong learning with sketched structural regularization. In *ACML*, pages 985–1000. PMLR.

Ronghan Li, Lifang Wang, Shengli Wang, and Zejun Jiang. 2021b. Asynchronous multi-grained graph network for interpretable multi-hop reading comprehension. In *IJCAI*, pages 3857–3863.

Xiang Lisa Li and Percy Liang. 2021. Prefix-tuning: Optimizing continuous prompts for generation. *arXiv preprint arXiv:2101.00190*.

Pengfei Liu, Weizhe Yuan, Jinlan Fu, Zhengbao Jiang, Hiroaki Hayashi, and Graham Neubig. 2021. Pre-train, prompt, and predict: A systematic survey of prompting methods in natural language processing. *arXiv preprint arXiv:2107.13586*.

Sewon Min, Victor Zhong, Luke Zettlemoyer, and Hannaneh Hajishirzi. 2019. Multi-hop reading comprehension through question decomposition and rescoring. *arXiv preprint arXiv:1906.02916*.

Kosuke Nishida, Kyosuke Nishida, Masaaki Nagata, Atsushi Otsuka, Itsumi Saito, Hisako Asano, and Junji Tomita. 2019. Answering while summarizing: Multi-task learning for multi-hop qa with evidence extraction. *arXiv preprint arXiv:1905.08511*.

German I Parisi, Ronald Kemker, Jose L Part, Christopher Kanan, and Stefan Wermter. 2019. Continual lifelong learning with neural networks: A review. *Neural Networks*, 113:54–71.

Ethan Perez, Patrick S. H. Lewis, Wen-tau Yih, Kyunghyun Cho, and Douwe Kiela. 2020. Unsupervised question decomposition for question answering. In *EMNLP*, pages 8864–8880.

Peng Qi, Haejun Lee, Oghenetegiri Sido, Christopher D Manning, et al. 2020. Answering open-domain questions of varying reasoning steps from text. *arXiv preprint arXiv:2010.12527*.

Peng Qi, Xiaowen Lin, Leo Mehr, Zijian Wang, and Christopher D Manning. 2019. Answering complex open-domain questions through iterative query generation. *arXiv preprint arXiv:1910.07000*.

Yujia Qin, Jiajie Zhang, Yankai Lin, Zhiyuan Liu, Peng Li, Maosong Sun, and Jie Zhou. 2022. Elle: Efficient lifelong pre-training for emerging data. *arXiv preprint arXiv:2203.06311*.

Lin Qiu, Yunxuan Xiao, Yanru Qu, Hao Zhou, Lei Li, Weinan Zhang, and Yong Yu. 2019. Dynamically fused graph network for multi-hop reasoning. In *ACL*, pages 6140–6150.

Pranav Rajpurkar, Jian Zhang, Konstantin Lopyrev, and Percy Liang. 2016. Squad: 100,000+ questions for machine comprehension of text. *arXiv preprint arXiv:1606.05250*.

David Rolnick, Arun Ahuja, Jonathan Schwarz, Timothy Lillicrap, and Gregory Wayne. 2019. Experience replay for continual learning. *Advances in Neural Information Processing Systems*, 32.

Apoorv Saxena, Aditay Tripathi, and Partha Talukdar. 2020. Improving multi-hop question answering over knowledge graphs using knowledge base embeddings. In *ACL*, pages 4498–4507.

Roy Schwartz, Jesse Dodge, Noah A Smith, and Oren Etzioni. 2020. Green ai. *Communications of the ACM*, 63(12):54–63.

Nan Shao, Yiming Cui, Ting Liu, Shijin Wang, and Guoping Hu. 2020a. Is Graph Structure Necessary for Multi-hop Question Answering? In *EMNLP*, pages 7187–7192, Online. Association for Computational Linguistics.

Nan Shao, Yiming Cui, Ting Liu, Shijin Wang, and Guoping Hu. 2020b. Is graph structure necessary for multi-hop question answering? In *EMNLP*, pages 7187–7192.

Fan-Keng Sun, Cheng-Hao Ho, and Hung-Yi Lee. 2019. Lamol: Language modeling for lifelong language learning. *arXiv preprint arXiv:1909.03329*.

Yu Sun, Shuohuan Wang, Yukun Li, Shikun Feng, Hao Tian, Hua Wu, and Haifeng Wang. 2020. Ernie 2.0: A continual pre-training framework for language understanding. In *AAAI*.

Alon Talmor and Jonathan Berant. 2018. The web as a knowledge-base for answering complex questions. *arXiv preprint arXiv:1803.06643*.

Yixuan Tang, Hwee Tou Ng, and Anthony Tung. 2021. Do multi-hop question answering systems know how to answer the single-hop sub-questions? In *ACL*, pages 3244–3249, Online. Association for Computational Linguistics.

Harsh Trivedi, Niranjan Balasubramanian, Tushar Khot, and Ashish Sabharwal. 2021. Musique: Multi-hop questions via single-hop question composition. *arXiv preprint arXiv:2108.00573*.

Ming Tu, Kevin Huang, Guangtao Wang, Jing Huang, Xiaodong He, and Bowen Zhou. 2020. Select, answer and explain: Interpretable multi-hop reading comprehension over multiple documents. In *AAAI*, pages 9073–9080.

Johannes Welbl, Pontus Stenetorp, and Sebastian Riedel. 2018. Constructing datasets for multi-hop reading comprehension across documents. *Transactions of the Association for Computational Linguistics*, 6:287–302.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Rémi Louf, Morgan Funtowicz, et al. 2020. Transformers: State-of-the-art natural language processing. In *EMNLP*, pages 38–45.

Bohong Wu, Zhuosheng Zhang, and Hai Zhao. 2021. Graph-free multi-hop reading comprehension: A select-to-guide strategy. *arXiv preprint arXiv:2107.11823*.

Zhilin Yang, Peng Qi, Saizheng Zhang, Yoshua Bengio, William Cohen, Ruslan Salakhutdinov, and Christopher D. Manning. 2018. HotpotQA: A dataset for diverse, explainable multi-hop question answering. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 2369–2380, Brussels, Belgium. Association for Computational Linguistics.

Yunchang Zhu, Liang Pang, Yanyan Lan, Huawei Shen, and Xueqi Cheng. 2021. Adaptive information seeking for open-domain question answering. *arXiv preprint arXiv:2109.06747*.

GLAF: Global-to-Local Aggregation and Fission Network for Semantic-Level Fact Verification

Zhiyuan Ma[1], **Jianjun Li**[1]*, **Guohui Li**[1], **Yongjing Cheng**[2]
[1] Huazhong University of Science and Technology (HUST), China
[2] National University of Defense Technology (NUDT), China
{zhiyuanma,jianjunli,guohuili}@hust.edu.cn
davidcheng1001@163.com

Abstract

Accurate fact verification depends on performing fine-grained reasoning over crucial entities by capturing their latent logical relations hidden in multiple evidence clues, which is generally lacking in existing fact verification models. In this work, we propose a novel Global-to-Local Aggregation and Fission network (GLAF) to fill this gap. Instead of treating entire sentences or all semantic elements within them as nodes to construct a coarse-grained or unstructured evidence graph as in previous methods, GLAF constructs a fine-grained and structured evidence graph by parsing the rambling sentences into structural triple-level reasoning clues and regarding them as graph nodes to achieve fine-grained and interpretable evidence graph reasoning. Specifically, to capture latent logical relations between the clues, GLAF first employs a local fission reasoning layer to conduct fine-grained multi-hop reasoning, and then uses a global evidence aggregation layer to achieve information sharing and the interchange of evidence clues for final claim label prediction. Experimental results on the FEVER dataset demonstrate the effectiveness of GLAF, showing that it achieves the state-of-the-art performance by obtaining a 77.62% FEVER score.

1 Introduction

The classic fact verification (FV) task is defined as retrieving relevant sentences as evidence and conducting joint reasoning over these evidence sentences to verify the correctness of a claim, and finally returning a result such as "SUPPORTS", "REFUTES", or "NOT ENOUGH INFO". With the increasingly frequent internet fraud, political rumors, fake news and other false information online, fact verification is becoming more and more important. How to automatically verify the fake claims and prevent their spread is a vital problem.

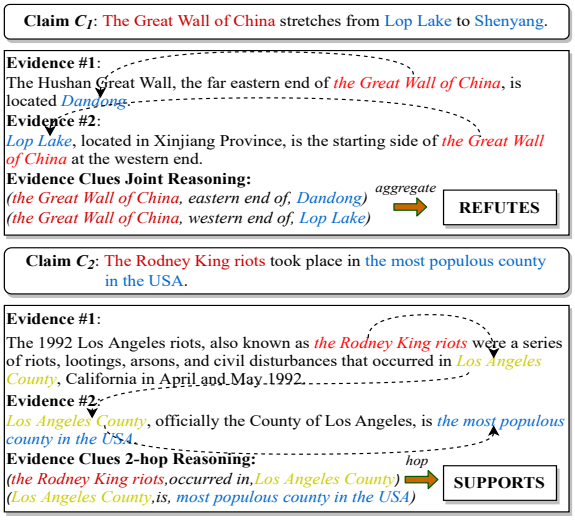

Figure 1: Two motivating examples from FEVER, which requires multi-evidence joint reasoning or multi-hop reasoning to achieve accurate claim label prediction.

In recent years, natural language inference models actually have dominated the study of fact verification (Si et al., 2021; Zhu et al., 2021; Thorne et al., 2018a; Luken et al., 2018; Yin and Roth, 2018; Ye et al., 2020), and many graph augmented neural inference models have been proposed (Zhou et al., 2019; Zhong et al., 2020; Liu et al., 2020; Chen et al., 2021a,b), which first integrate multi-evidence reasoning into fact verification with evidence graph, and then output the claim label prediction result.

Though achieving remarkable progress, existing neural inference FV models still suffer from the following three limitations. **Firstly**, they generally lack the capability of fine-grained evidence clue representing and semantic-level entity reasoning. Most of them either concatenate evidence sentences into a single string (Thorne et al., 2018b), or just treat each evidence-claim pair as a sentence-level node (Zhou et al., 2019; Liu et al., 2020). Since these methods represent and aggregate the evidence at sentence-level, they have difficulty in achieving fine-grained reasoning. Take claim C_1

*Corresponding author.

Proceedings of the 29th International Conference on Computational Linguistics, pages 1801–1812
October 12–17, 2022.

in Figure 1 as an example, the claim states "*The Great Wall of China stretches from Lop Lake to Shenyang*", while the evidence states that the great wall stretches to "*Dandong*" instead of "*Shenyang*". Hence, it requires the FV model to carefully distinguish the subtle differences between truth and false statements. However, existing sentence-level FV models are hard to make such a meticulous discrimination over these crucial entities (e.g., "*Dandong*" and "*Shenyang*"). **Secondly**, prior models generally lack the capability of latent logical relation mining and interpretable claim verifying. As the false claims are often deliberately fabricated, they may be semantically reasonable but logically are not supported. Hence, it requires sufficient logical relation capturing and hop-based reasoning over these clues to guide an interpretable claim judgment. For example, claim C_2 in Figure 1 states "*The Rodney King riots took place in the most populous county in the USA*", while the evidence clues present that *(The Rodney King riots, occurred in, Los Angeles County)* and *(Los Angeles County, is, the most populous county in the USA)*, it requires the FV model to mine the pivot "*Los Angeles County*" and capture the latent relation between "*The Rodney King riots*" and "*the most populous county in the USA*" to make an accurate and convincing judgment by performing multi-hop reasoning. However, existing unstructured FV methods in general cannot support such a triple-level multi-hop reasoning. **Thirdly**, previous models generally lack the noise evidence filtering mechanism. Since the evidence sentences are retrieved from complex background corpora, they will inevitably introduce noises. Even worse, these noises may be magnified in subsequent neural computations, which seriously deteriorates the FV performance.

To tackle these problems, we propose a novel Global-to-Local Aggregation and Fission network (GLAF), which is a graph attention augmented neural inference model for FV. Specifically, to address the first limitation, we first parse the sentences into fine-grained and structural relation triples, each denoted as (s, r, o), and then feed them into the BERT (Devlin et al., 2019) to obtain a set of global evidence clue representations. Next, we introduce a fresh perspective to exploit these structural evidence clue triples. That is, we model each triple (s, r, o) as a map function $f_{\text{clue }r}(s) \rightarrow o$, and use it to conduct entity-level multi-hop reasoning for final claim verification. To address the second limitation, we employ two neural inference layers: local fission reasoning layer and global evidence aggregation layer, to iteratively conduct 2-hop object reasoning and evidence joint reasoning through a triple-level attention mechanism. The two neural layers are utilized to guide the interpretable reasoning process and improve the accuracy of fact verification. Finally, to address the third limitation, we use a graph pooling layer to iteratively select hidden evidence nodes as crucial evidence clues and filter out disruptive noise data, so as to improve the robustness of our FV model.

We conduct experiments on FEVER (Thorne et al., 2018a), which is one of the most influential benchmark datasets for fact verification. We follow the official evaluation protocol of FEVER and demonstrate that GLAF outperforms the recent state-of-the-art baseline systems. Ablation study also shows the effectiveness of each component in improving the performance of fact verification, and a further case study reveals that our model can effectively perform fine-grained multi-hop reasoning over these evidence clues and reach an interpretable conclusion for fact verification.

2 Related Work

2.1 Traditional Fact Verification Models

Many traditional fact verification (FV) systems utilize Natural Language Inference (NLI) techniques (Parikh et al., 2016; Peters et al., 2018; Soleimani et al., 2020) to mine the relationship between evidence and claim to make a final judgment. One of the representative work is the FEVER shared task (Thorne et al., 2018b), which aims to develop an automatic FV system to check the veracity of human-generated claims. Traditional FV models usually employ FEVER's official baseline (Thorne et al., 2018a) with a three-step pipeline: document retrieval, sentence retrieval and claim verification. Among these models, many mainly focus on the last step. For example, Nie et al. (2019) concatenate all evidences together to verify the claim. Yoneda et al. (2018) infer the veracity of each claim-evidence pair and make final prediction by aggregating multiple predicted labels. Hanselowski et al. (2018) encode each claim-evidence pair separately, and use a pooling function to aggregate features for prediction. One of the most widely used models in FEVER is Enhanced Sequential Inference Model (ESIM) proposed by Chen et al. (2017), which has been adopted to select

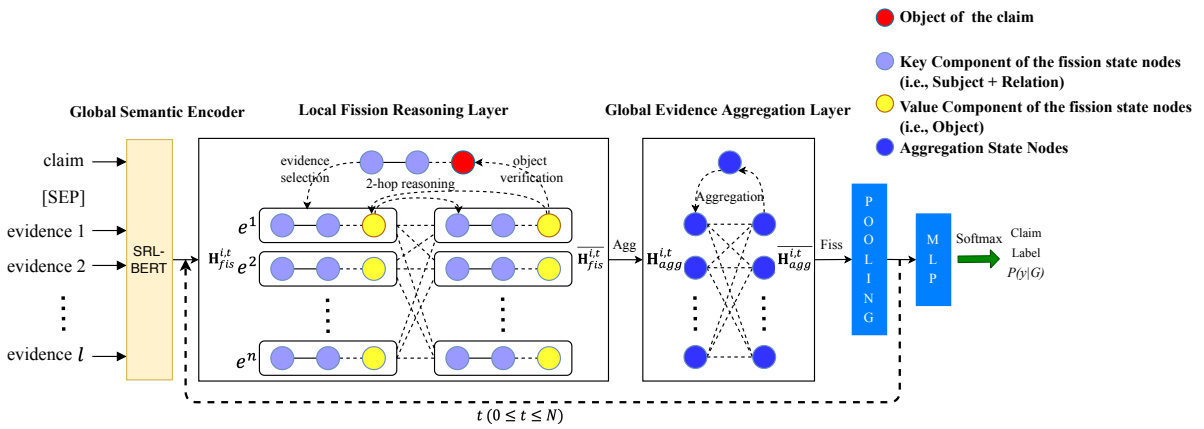

Figure 2: Architecture of GLAF.

relevant sentences in the sentence retrieval phase. Note most of the above mentioned FV methods employ simple models to extract information from evidence, but without letting evidence communicate with each other, which limits their performance. To address this problem, recently there are some good attempts. For example, Si et al. (2021) leverage the LDA model to conduct topic-aware evidence reasoning and stance-aware information aggregation for FV. Wan et al. (2021) employ DQN to find a minimal set of evidence and conduct sentence-level information aggregation for FV. Jiang et al. (2021) improve on previous pointwise aggregation manner by taking advantage of T5 model and explore a listwise-based evidence aggregation method.

2.2 Graph Augmented Fact Verification Models

Though achieving remarkable progress, the above methods in general are difficult to perform global evidence aggregation, since they are not based on graph and hence cannot take the advantage of graph propagation. Recently, by integrating multi-evidence reasoning and global information propagation into fact verification based on a constructed evidence graph, many graph augmented FV models (Zhou et al., 2019; Zhong et al., 2020; Liu et al., 2020; Chen et al., 2020) have been proposed and achieve state-of-the-art results. Among them, **GEAR** (Zhou et al., 2019) is the first to use BERT (Devlin et al., 2019) to encode evidence, and designs a graph network to aggregate information on an evidence graph constructed by treating each evidence as a node. **DREAM** (Zhong et al., 2020) further employs XLNet (Yang et al., 2019) to establish a semantic-level graph for evidence aggregation by using GCN (Kipf and Welling, 2017) and GAT (Velickovic et al., 2018). **KGAT** (Liu et al., 2020) innovatively adopts a kernel graph attention network to aggregate information by unifying the edge kernel mechanism and node kernel mechanism over the evidence graph. However, existing graph augmented FV models either treat entire sentences as sentence-level nodes, e.g., GEAR and KGAT, or extract all semantic elements within them as semantic-level nodes, e.g., DREAM, which makes them only focus on global evidence aggregation, while ignoring reasoning over local semantic clues in triple-level. Different from them, our proposed GLAF is based on a well-structured triple-level evidence graph, which facilitate fine-grained multi-hop reasoning over the crucial entities (i.e., *subject* or *object*) to capture explicit reasoning chains for interpretable claim verification.

3 Model Description

Given a claim c and l evidence sentences, the fact verification task aims to check the veracity of the claim and return a prediction label y, where $y \in \{$"SUPPORTS", "REFUTES", "NOT ENOUGH INFO"$\}$. Instead of treating entire evidence sentences or all semantic elements within them as nodes to construct a coarse-grained or unstructured evidence graph as in previous methods, GLAF constructs a fine-grained and structured evidence graph G by using an off-the-shelf semantic role labeling (SRL) toolkit[1] to parse the l sentences into n structural relation triples[2] and regarding them as graph nodes, denoted by $E = \{e^1, \ldots, e^i, \ldots, e^n\}$. For each evidence node e^i, we use $e^i = (s^i, r^i, o^i)$ to represent a relation triple (*subject*, *relation*, *object*). Then, all

[1] A re-implementation of a BERT-based model by AllenNLP.

[2] Note each sentence could be parsed as multiple triples.

these nodes are connected with edges to obtain a fully-connected evidence graph G with n nodes. Based on G, GLAF produces a prediction probability $P(y|G)$ by reasoning over these evidence triples (nodes) to predict the claim label y. Similar to KGAT, we follow the standard graph label prediction setting in graph neural network (Velickovic et al., 2018) and split the prediction into two components: 1) the evidence selection probability $P\left(e^i|G\right)$; 2) the fine-grained label prediction probability $P\left(y|e^i,G\right)$:

$$P(y|G) = \sum_{i=1}^{n} P\left(e^i|G\right) P\left(y|e^i,G\right) \quad (1)$$

As shown in Figure 2, GLAF mainly includes four modules: Global Semantic Encoder (GSE), Local Fission Reasoning (LFR) Layer, Global Evidence Aggregation (GEA) Layer, and Graph Pooling and MLP Classification Layer. Specifically, GSE is used to obtain initial representations for the claim c and all the nodes in G; LFR, with the initial node representations as inputs, is responsible for conducting 2-hop object reasoning for fine-grained claim verification. LFR outputs updated node representations, which later will be aggregated to serve as the inputs of the GEA layer; GEA is utilized to achieve information sharing by performing 1-order neighborhood information integration; Then, Graph pooling is utilized to filter noise data to select valuable nodes by calculating the evidence selection probability $P\left(e^i|G\right)$; Finally, an MLP layer is used to calculate the fine-grained label prediction probability $P\left(y|e^i,G\right)$.

Note that LFR and GEA each can perform 2-hop reasoning and 1-order neighborhood aggregation, respectively. By iteratively execute LFR and GEA, we can implement more hop reasoning and higher order aggregation to capture sufficient logical relation for more accurate verification. Such an iterative execution process is illustrated in Algorithm 1 (Lines 21-32). We use t to denote the iteration index and assume that $0 \leq t \leq N$, where N represents the total number of iterations and is a model parameter. For convenience, in the following discussion, we use the superscript t to denote the representations or values at the t-th iteration. Next, we detail the separate modules.

3.1 Global Semantic Encoder

Following KGAT, GLAF employs the pretrained language model BERT as the contextual semantic encoder to initialize the global node representations. It is worth noting that since we construct the evidence graph based on triple-level nodes, we first concatenate each triple as `[CLS]` *subject* `[SEP]` *relation* `[SEP]` *object* `[SEP]` and then feed them to BERT to obtain the initial hidden state representation of the node. Specifically, for node e^i, the evidence clue triple is initialized as $\mathbf{H}_{fis}^{i,0}$, where,

$$\mathbf{H}_{fis}^{i,0} = [\mathbf{h}_s^{i,0}; \mathbf{h}_r^{i,0}; \mathbf{h}_o^{i,0}] = \text{BERT}\left([s^i; r^i; o^i]\right) \quad (2)$$

Similarly, the claim node representation is initialized as,

$$[\mathbf{h}_s^c; \mathbf{h}_r^c; \mathbf{h}_o^c] = \text{BERT}\left([s^c; r^c; o^c]\right) \quad (3)$$

3.2 Local Fission Reasoning Layer

In order to encourage the triple nodes to update the object information for fine-grained claim verification, GLAF employs a local fission reasoning layer to conduct 2-hop entity-level reasoning between the target node and its connected nodes, and finally obtains an updated object vector. Different from the previous models that use the whole evidence sentence as a node (Zhou et al., 2019; Liu et al., 2020), we propose to perform entity-level object reasoning. Specifically, given a target node e^i, we update its object vector representation as follows:

- Calculates the cosine similarity between the object (tail-entity) of the target node e^i and the subject (head-entity) of the connected node e^j,

$$M_{fis}^{j\to i,t} = \cos\left(\mathbf{h}_o^{i,t}, \mathbf{h}_s^{j,t}\right) \quad (4)$$

- Obtains the attentive weights by softmax function,

$$\alpha_{fis}^{j\to i,t} = \frac{\exp\left(M_{fis}^{j\to i,t}\right)}{\sum_{k=1}^{n} \exp\left(M_{fis}^{k\to i,t}\right)} \quad (5)$$

- Calculates the attentive vector corresponding to the specific object after 2-hop reasoning,

$$\mathbf{a}_o^{i,t} = \sum_{j=1}^{n} \alpha_{fis}^{j\to i,t} \mathbf{h}_o^{j,t} \quad (6)$$

- Adaptively updates the object vector representation,

$$\overline{\mathbf{h}_o^{i,t}} = \text{LeakyReLU}\left(\omega_1 \mathbf{h}_o^{i,t} + \omega_2 \mathbf{a}_o^{i,t}\right) \quad (7)$$

where ω_1 and ω_2 are two trainable linear weights, and $\overline{\mathbf{h}_o^{i,t}}$ denotes the updated object vector representation of e^i.

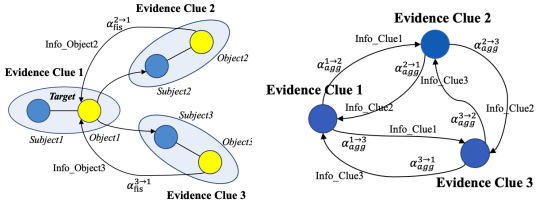

(a) Local fission reasoning (b) Global evidence aggregation

Figure 3: Two kinds of attention-based neural layer.

Consequently, the updated representation of e^i after a 2-hop reasoning can be obtained by,

$$\overline{\mathbf{H}_{fis}^{i,t}} = [\mathbf{h}_s^{i,t}; \mathbf{h}_r^{i,t}; \overline{\mathbf{h}_o^{i,t}}] \tag{8}$$

Function LFR$(\cdot)$ in Algorithm 1 presents the detailed implementation of the LFR layer. Finally, to facilitate the subsequent global information sharing, for each node e^i, we conduct an aggregation operation to convert the fission representation $\overline{\mathbf{H}_{fis}^{i,t}}$ to the aggregation representation $\mathbf{H}_{agg}^{i,t}$ by,

$$\mathbf{H}_{agg}^{i,t} = \mathbf{W}_{agg}^{i,t} \cdot \overline{\mathbf{H}_{fis}^{i,t}} \tag{9}$$

where $\mathbf{W}_{agg}^{i,t}$ is a trainable matrix.

3.3 Global Evidence Aggregation Layer

To achieve information sharing and multi-evidence joint reasoning, GLAF utilizes a global evidence aggregation layer to perform 1-order neighborhood aggregation, which mainly includes three steps:

- Calculates the cosine similarity between the aggregated nodes,

$$M_{agg}^{j \to i,t} = \cos\left(\mathbf{H}_{agg}^{i,t}, \mathbf{H}_{agg}^{j,t}\right) \tag{10}$$

- Obtains the attentive weights by softmax function,

$$\alpha_{agg}^{j \to i,t} = \frac{\exp\left(M_{agg}^{j \to i,t}\right)}{\sum_{k=1}^{n} \exp\left(M_{agg}^{k \to i,t}\right)} \tag{11}$$

- Calculates the attention vector as updated aggregated node representation after integrating the information of the surrounding n nodes (including itself),

$$\overline{\mathbf{H}_{agg}^{i,t}} = \sum_{j=1}^{n} \alpha_{agg}^{j \to i,t} \mathbf{H}_{agg}^{j,t} \tag{12}$$

Function GEA$(\cdot)$ in Algorithm 1 presents the detailed implementation of this layer. Similarly, to facilitate the execution of the LFR layer in the next iteration (if exists), we conduct a fission operation to convert the aggregation representation $\overline{\mathbf{H}_{agg}^{i,t}}$ to the fission representation $\mathbf{H}_{fis}^{i,t}$ by,

$$\mathbf{H}_{fis}^{i,t} = \mathbf{W}_{fis}^{i,t} \cdot \overline{\mathbf{H}_{agg}^{i,t}} \tag{13}$$

where $\mathbf{W}_{fis}^{i,t}$ is a trainable matrix.

3.4 Graph Pooling and MLP Classification Layer

GLAF employs a graph pooling layer to conduct node selection and noise filtering. Specifically, at the t-th iteration, after GEA is executed, the pooling layer discards nodes with few evidence clues and only selects the k (k is a model parameter) most valuable nodes from all of the aggregated

Algorithm 1 GLAF graph learning algorithm

Input: Evidence node set $E = \{e^1, \ldots, e^i, \ldots, e^n\}$ and claim c
Parameter: Number of iterations N and pooling parameter k
Output: Label y

Initialize: global node representations and all the parameters
function LFR(fiss_node_set, t)
  for each fiss_node e^i **do**
    for each fiss_node e^j **do**
      Similarity$(\mathbf{h}_o^{i,t}, \mathbf{h}_s^{j,t}) \to \alpha_{fis}^{j \to i,t}$ (Eqs.4-5).
    end for
    Sum$(\alpha_{fis}^{j \to i,t} \cdot \mathbf{h}_o^{j,t}) \to \mathbf{a}_o^{i,t}$ (Eq.6)
    Weigh $(\mathbf{a}_o^{i,t}, \mathbf{h}_o^{i,t})$ to update $\mathbf{h}_o^{i,t} \to \overline{\mathbf{h}_o^{i,t}}$ (Eq.7)
    Concatenate $(\mathbf{h}_s^{i,t}, \mathbf{h}_r^{i,t}, \overline{\mathbf{h}_o^{i,t}}) \to \overline{\mathbf{H}_{fis}^{i,t}}$ (Eq.8)
  end for
end function
function GEA(agg_node_set, t)
  for each agg_node e^i **do**
    for each agg_node e^j **do**
      Similarity$(\mathbf{H}_{agg}^{i,t}, \mathbf{H}_{agg}^{j,t}) \to \alpha_{agg}^{j \to i,t}$ (Eqs.10-11).
    end for
    Sum$(\alpha_{agg}^{j \to i,t} \cdot \mathbf{H}_{agg}^{j,t}) \to \overline{\mathbf{H}_{agg}^{i,t}}$ (Eq.12)
  end for
end function
Let $t = 0$, $E^0 = E$
while $t \leq N$ **do**
  LFR(E^t,t)
  for each fiss_node e^i in E **do**
    $\overline{\mathbf{H}_{fis}^{i,t}} \xrightarrow{Agg} \mathbf{H}_{agg}^{i,t}$ (Eq.9)
  end for
  GEA(E^t,t)
  for each agg_node e^i in E **do**
    $\overline{\mathbf{H}_{agg}^{i,t}} \xrightarrow{Fiss} \mathbf{H}_{fis}^{i,t}$ (Eq.13)
  end for
  $E^{t+1} = \text{Pooling}\left(E^t, c, k\right)$
  $t = t + 1$
end while
Calculate $P(y|G)$ by Eqs.16-18 for each label
$y = \text{argmax} P(y|G)$
return y

nodes to serve as the readout. Formally,

$$E^{t+1} = \text{Pooling}\left(E^t, c, k\right) \tag{14}$$

where E^{t+1} denotes the evidence node set after pooling on E^t, with $E^0 = E$. Moreover, the value of a node e^i is defined as the semantic similarity between the key vector $[\mathbf{h}_s^c; \mathbf{h}_r^c] \in \mathbb{R}^{1\times 2F}$ (F is the feature dimension) of the claim c and e^i's updated representation $\overline{\mathbf{H}_{agg}^{i,t}}$,

$$M_{\text{pool}}^{i\to c,t} = \cos\left([\mathbf{h}_s^c; \mathbf{h}_r^c] \cdot \mathbf{W}, \overline{\mathbf{H}_{agg}^{i,t}}\right) \tag{15}$$

where $\mathbf{W} \in \mathbb{R}^{2F\times F}$ is a trainable matrix, which is used to align the dimensions of the two vectors. Then, GLAF obtains the evidence selection probability $P\left(e^i|G\right)$ by,

$$P\left(e^i|G\right) = \frac{\exp\left(M_{\text{pool}}^{i\to c,t}\right)}{\sum_{j=1}^{n}\exp\left(M_{\text{pool}}^{j\to c,t}\right)} \tag{16}$$

The k nodes with the highest probability $P\left(e^i|G\right)$ will be selected as the readout.

After N iterations, the representations of the remaining activated nodes are fed into an MLP to conduct claim object verification and generate the fine-grained label prediction probability,

$$P\left(y|e^i, G\right) = \text{softmax}_i(\text{MLP}(\overline{\mathbf{H}_{agg}^{i,t}}, \mathbf{h}_o^c)) \tag{17}$$

Finally, we can obtain the final prediction probability by,

$$P(y|G) = \sum_{i=1}^{k} P\left(e^i|G\right) P\left(y|e^i, G\right) \tag{18}$$

The whole GLAF model is trained end-to-end by minimizing the cross-entropy loss,

$$L = \text{CrossEntropy}\left(y^*, P(y|G)\right) \tag{19}$$

using the ground truth verification label y^*.

4 Experimental Setup

4.1 Datasets and Metrics

We conduct all our experiments on the large-scale dataset FEVER (Thorne et al., 2018a), which consists of 185,455 annotated claims with a set of 5,416,537 Wikipedia documents from the June 2017 Wikipedia dump. We keep the dataset partition the same as the FEVER Shared Task (Thorne et al., 2018a) and TWOWINGOS (Yin and Roth, 2018). Table 1 shows the statistics of the dataset after partition.

Split	Supported	Refuted	Not Enough Info
Train	80,035	29,775	35,639
Dev	3,333	3,333	3,333
Test	3,333	3,333	3,333

Table 1: Statistics of the FEVER dataset.

Following several previous work (Zhou et al., 2019; Zhong et al., 2020; Liu et al., 2020), we use the official evaluation metrics[3] to evaluate the performance of our model on fact verification, which includes Label Accuracy (LA) and FEVER score (FEVER). LA is a general evaluation metric, which calculates claim prediction accuracy rate without considering retrieved evidence. The FEVER score considers whether all evidence included in a golden evidence set are mined, and hence better reflects the reasoning ability.

4.2 Baselines

We compare our model GLAF with the following state-of-the-art baselines.

- **UNC-NLP** (Nie et al., 2019)[4] proposes a neural semantic matching network for claim verification to jointly solve three subtasks by incorporating additional information, such as pageview frequency and WordNet features, for information aggregation.
- **BERT Fine-tuning Systems** (Zhou et al., 2019) includes BERT-Concat and BERT-Pair. The BERT-Concat system concatenates all evidence into a single string while the BERT-Pair system encodes each claim-evidence pair independently and then aggregates the results. For these two BERT fine-tuning systems, we use the source code from (Zhou et al., 2019) and keep the settings unchanged.
- **GEAR** (Zhou et al., 2019)[5] is a graph-based evidence aggregating and reasoning framework by employing an evidence aggregator to aggregate information and conduct evidence reasoning over the evidence graph.
- **DREAM** (Zhong et al., 2020)[6] is built on top of XLNet (Yang et al., 2019) and models evidence graph at a semantic-level by retrieving

[3]https://github.com/sheffieldnlp/fever-scorer
[4]https://github.com/easonnie/combine-FEVER-NSMN
[5]https://github.com/thunlp/GEAR
[6]We reproduce DREAM and try to keep the same settings as the original paper as no open-source code is available.

	Model	Precision	Recall	F1	FEVER
Dev	ESIM	24.08	86.72	37.69	71.70
	BERT	**27.29**	**94.37**	**42.34**	**75.88**
Test	ESIM	23.51	84.66	36.80	68.16
	BERT	**25.21**	**87.47**	**39.14**	**69.40**

Table 2: Results of evidence selection models.

all semantic elements as graph nodes. This model employs a GCN and a GAT network to conduct information aggregation.

- **KGAT** (Liu et al., 2020)[7] models claim-evidence pairs into nodes and adopts a kernel-based graph attention network to conduct evidence aggregating and reasoning.

4.3 Implementation Details

Evidence sentence retrieval We adopt a two-stage scheme to retrieve evidence sentences, which includes document retrieval stage and sentence selection stage. The document retrieval stage retrieves related Wikipedia pages and is kept the same with previous work (Zhou et al., 2019; Liu et al., 2020). At first, it extract all potentially entities included claim as key phrases by using the constituency parser developed by AllenNLP. Then, it regards theses key phrases as queries to search relevant Wikipedia pages through the online MediaWiki API[8], until it searching out convinced article. The sentence selection stage selects relevant sentences from retrieved Wikipedia pages. In our experiments, we try both ESIM-based retrieval model and BERT-based retrieval model. From Table 2, we can see that BERT performs better than ESIM. So, we adopt BERT-based model to retrieve evidence sentences. Specifically, following previous work (Zhou et al., 2019; Liu et al., 2020), we first feed these evidence sentences to a BERT-based ranking model. Then, we use the "`[CLS]`" hidden state to represent claim-evidence pair. Finally, we adopt a pairwise loss to optimize the ranking model for obtaining an optimal evidence retrieval result.

Triple-level clue representation. Similar to previous work (Zhou et al., 2019; Liu et al., 2020), we adopt an identical two-stage scheme to retrieve evidence sentences from background corpus. But different from them, we subsequently adopt a semantic role labeling toolkit to parse each evidence sentence into triple format. Specifically, we built the triples by using the results of the SRL toolkit[9], which includes *(subject, predicate, object)* and *(subject, attributes, value)*. Note that these extracted attributes include time, place, purpose, reason, and other crucial elements that can be mined by SRL. We process all these triples as evidence clues and feed them into BERT to obtain a set of triple-level evidence clue representations. Then, we built an evidence graph by using these triple representations as initial nodes, as described in Section 3.

Model	Dev		Test	
	LA	FEVER	LA	FEVER
UNC-NLP	0.7034	0.6716	0.6858	0.6472
BERT-Concat	0.7399	0.6987	0.7185	0.6718
BERT-Pair	0.7463	0.7008	0.7179	0.6752
KGAT (ESIM)	0.7551	0.7269	0.7348	0.7050
GLAF (ESIM)	**0.7586**	**0.7370**	**0.7441**	**0.7236**
GEAR	0.7601	0.7133	0.7304	0.6815
DREAM ($\text{XLNet}_{\text{Large}}$)	0.7792	0.7235	0.7698	0.7140
KGAT ($\text{BERT}_{\text{Base}}$)	0.7787	0.7575	0.7593	0.7419
GLAF ($\text{BERT}_{\text{Base}}$)	**0.7804**	**0.7635**	**0.7703**	**0.7494**
KGAT ($\text{CorefBERT}_{\text{Base}}$)	0.7798	0.7608	0.7635	0.7441
GLAF ($\text{CorefBERT}_{\text{Base}}$)	**0.7835**	**0.7658**	**0.7760**	**0.7522**
GLAF ($\text{BERT}_{\text{Large}}$)	0.7829	0.7662	0.7784	0.7565
GLAF ($\text{RoBERTa}_{\text{Large}}$)	0.7852	0.7641	0.7905	0.7620
GLAF ($\text{CorefRoBERTa}_{\text{Large}}$)	**0.7941**	**0.7840**	**0.8012**	**0.7762**

Table 3: Overall performance. Note the FEVER score on the blind test set is the main evaluation metric made by FEVER organizers, and all results are statistically significant with $p < 0.05$ under t-test.

Model training details. In our experiments, the batch size is set to 8, learning rate is set to $2e^{-5}$ and warmup proportion is set to 0.1. The max length is set to 140, and the max number of training epochs is set to 6. The maximum number of iterations N is set to 2. BERT and CorefBERT respectively inherit huggingface's implementation[10] and THUNLP's repository[11]. The same as previous work (Zhou et al., 2019; Liu et al., 2020), Adam optimizer is used to optimize all models. All experiments are conducted with PyTorch, and all the source code will be made publicly available upon acceptance. More details about hyper-parameter settings can be found in the Appendix.

5 Evaluation Results

5.1 Overall Performance

The overall performance is shown in Table 3, where the best performance in each scenario is in boldface. It can be observed that, compared with other baselines, GLAF exhibits the best performance

[7] https://github.com/thunlp/KernelGAT
[8] https://www.mediawiki.org/wiki/API:Main_page
[9] https://demo.allennlp.org/semantic-role-labeling
[10] https://github.com/huggingface/pytorch-transformers
[11] https://github.com/thunlp/CorefBERT

Model	FEVER(%)	
	Test	Δ
Complete model	**77.62**	-
w/o SRL retrieval & LFR Layer	72.55	5.07
w/o GEA Layer	74.20	3.42
w/o Graph Pooling Layer	76.35	1.27

Table 4: Ablation study on FEVER test set.

on all testing scenarios. With ESIM sentence retrieval, GLAF outperforms the classic top system UNC-NLP and current best model KGAT on both development and testing sets. With BERT-based sentence retrieval, GLAF outperforms GEAR by almost 10%, DREAM by almost 5% and KGAT by almost 1% test FEVER score. This illustrates the consistent effectiveness of GLAF among graph augmented reasoning models with different sentence retrieval methods. Furthermore, when using $\text{CorefBERT}_{\text{Base}}$, $\text{BERT}_{\text{Large}}$, $\text{RoBERTa}_{\text{Large}}$ and $\text{CorefRoBERTa}_{\text{Large}}$ as the encoder, GLAF achieves even better performance, especially for $\text{CorefRoBERTa}_{\text{Large}}$, it outperforms the current best model KGAT by almost 4.9% in LA metric and 4.3% in FEVER metric on blind test set and achieves the state-of-the-art performance.

5.2 Ablation Study

In this part, we perform ablation experiments to evaluate the effectiveness of each module and set them accordingly. 1) w/o SRL retrieval & LFR Layer[12] denotes that we remove semantic triple retrieval and local fission reasoning, and just adopt the global aggregation layer to aggregate information; 2) w/o GEA Layer denotes that we remove the global evidence aggregation layer and connect the local fission reasoning layer to the pooling layer directly; 3) w/o Graph Pooling Layer denotes that we remove the graph pooling layer and connect the GEA layer to the MLP layer directly. From the results in Table 4, we can observe that removing each module will result in a performance degradation. In particular, w/o SRL retrieval & LFR Layer and w/o GEA Layer causes 5.07 and 3.42 absolute drops in test FEVER score, respectively, which further verifies the effectiveness of our model.

5.3 Effectiveness Evaluation and analysis

Assessment of evidence mining capability. This experiment evaluates the capability of our model to effectively mine evidence when incremental corpus size is given. Specifically, more

[12]LFR cannot be decoupled with SRL retrieval, since it relies on the triples parsed by SRL. Therefore, we consider them together.

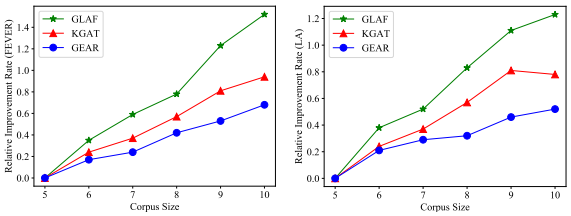

(a) FEVER improvement (b) LA improvement

Figure 4: The assessment of evidence mining capability.

Corpus Size	Number of iterations N			
	0	1	2	3
5	**0.7494**	0.7488	0.7443	-
6	0.7508	**0.7522**	0.7510	-
7	0.7522	**0.7545**	0.7518	-
8	0.7531	**0.7550**	0.7522	-
9	0.7540	**0.7579**	0.7534	-
10	0.7528	**0.7611**	0.7581	-
15	0.7568	0.7608	**0.7620**	0.7588

Table 5: Study of hyperparameter N under different corpus sizes. Best performance under each corpus size is in boldface.

evidence sentences will bring more noise elements, which requires the model to have stronger evidence selection and reasoning ability to carefully distinguish these evidence clues. As shown in Figure 4, we set 5 pieces of evidence as the basic scenario, and vary the corpus size in $[6, 7, 8, 9, 10]$ to test the improvement effect on FEVER and LA. As can be observed, compared with KGAT and GEAR, our GLAF model consistently achieves the best performance on FEVER and LA metrics. We conjecture the reason might be that the pooling layer in GLAF effectively improves the reasoning model by filtering out noise clues.

Study on the number of iterations N. We conduct this experiment to explore the optimal number of iterations N under different corpus sizes. From Table 5, we can observe that with the increase of the corpus size, more iterations are needed to dig out the potential logical relationships hidden among evidence nodes. Specifically, when corpus size $= 5$, the optimal number of iterations is $N = 0$; When corpus size varies in the range of $[6, 7, 8, 9, 10]$, the optimal number of iterations is $N = 1$; When corpus size reaches 15, the optimal number of iterations is $N = 2$. This experiment reveals the relationship between model depth and its performance. Specifically, deeper model may cause the overfitting problem, while shallower models may have difficulty in mining potential advanced features. Therefore, it is important to select the proper

number of iterations.

Corpus Size	With Pooling		Without Pooling	
	LA	FEVER	LA	FEVER
5	**0.7703**	**0.7494**	0.7680	0.7492
7	**0.7761**	**0.7540**	0.7743	0.7526
10	**0.7814**	**0.7609**	0.7752	0.7536
15	**0.8012**	**0.7762**	0.7905	0.7635

Table 6: Effectiveness evaluation of pooling layer.

More evaluation on the pooling layer. We conduct this experiment to further evaluate the effectiveness of the pooling layer. The result is shown in Table 6. In this experiment, we set up a group of comparison models with and without a pooling layer, and set the corpus size within $[5, 7, 10, 15]$. From Table 6, we can observe that the model with the pooling layer achieves better performance than the one without, which demonstrates the effectiveness of the pooling layer in improving the model's reasoning ability over evidence clues through the noise filtering mechanism.

5.4 Case Study

We take the fact verification task in Table 7 as an example, which requires performing triple-level 2-*hop* reasoning over retrieved evidence clues to reach a reliable conclusion. To verify whether "*The Rodney King riots*" took place in "*the most populous county in the USA*", our model mines two crucial evidence clues, (*The Rodney King riots*, *occurred in*, *Los Angeles County*) and (*Los Angeles County*, *is*, *the most populous county in the USA*), to perform attention-based 2-*hop* reasoning. To better understand what our LFR layer has learned, we visualize the attention map from the LFR layer and the final graph pooling layer, as shown in Figure 5. It is clear to see that node 1 achieves the highest value score 0.982 in the last column by integrating information from surrounding nodes, mainly from nodes 2 and 3. Since node 2 is semantically worthless and has the lowest value score 0.125, it will be filtered out by the pooling layer, which implies that $1 \rightarrow 3$ is the optimal 2-*hop* reasoning chain. Finally, the two corresponding evidence clues *(The Rodney King riots, occurred in, Los Angeles County)* and *(Los Angeles County, is, the most populous county in the USA)* can be successfully selected and reasoned to make the final claim verification.

Claim:
The Rodney King riots took place in the most populous county in the USA.
Evidence:
(1) The 1992 Los Angeles riots, also known as the Rodney King riots were a series of riots, lootings, and civil disturbances that occurred in Los Angeles County, California in April and May 1992.
(2) Los Angeles County, officially the County of Los Angeles, is the most populous county in the USA.
Retrieved claim clue:
(The Rodney King riots, took place in, the most populous county in the USA)
Retrieved evidence clues:
① (The Rodney King riots, were, riots_lootings_civil_disturbances)
② (The Rodney King riots, occurred in, Los Angeles County)
③ (Los Angeles County, officially named, the County of Los Angeles)
④ (Los Angeles County, is, the most populous county in the USA)
Label: SUPPORTED

Table 7: A case study illustrating semantic-level 2-*hop* reasoning over fine-grained evidence clues.

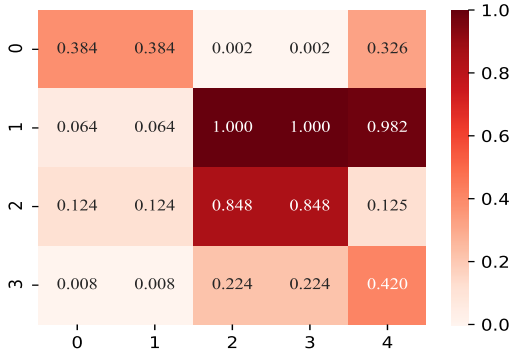

Figure 5: Attention map for the example in Table 7. The first four columns indicate the attention weights α_{fis} from nodes 0 to 3 (corresponding in turn to the four retrieved evidence clues in Table 7) in the LFR layer, and the last column visualizes the value score $M_{\text{pool}}^{i \rightarrow c}$ of the four nodes from the final graph pooling layer.

6 Conclusion

In this paper, we introduce a fresh perspective to revisit the fact verification task and propose a novel Global-to-Local Aggregation and Fission Network (GLAF) to capture latent logical relations hidden in evidence clues for more accurate fact verification. Instead of treating evidence as sentence-level or unstructured representations as in previous work, the proposed GLAF model first parses the evidence sentences as triple-level evidence clues, and then feeds them into contextual language model to obtain global semantic representations. Moreover, to capture latent logical relations between the clues, GLAF respectively employs a local fission layer to conduct fine-grained multi-hop reasoning, as well as a global aggregation layer to conduct interchanging of evidence clues in the graph. Experimental results on the benchmark dataset FEVER have demonstrated the effectiveness and superior performance of our model in both overall evaluation and ablation study.

References

Chonghao Chen, Fei Cai, Xuejun Hu, Wanyu Chen, and Honghui Chen. 2021a. HHGN: A hierarchical reasoning-based heterogeneous graph neural network for fact verification. *Inf. Process. Manag.*, 58(5):102659.

Chonghao Chen, Fei Cai, Xuejun Hu, Jianming Zheng, Yanxiang Ling, and Honghui Chen. 2021b. An entity-graph based reasoning method for fact verification. *Inf. Process. Manag.*, 58(3):102472.

Jiangjie Chen, Qiaoben Bao, Jiaze Chen, Changzhi Sun, Hao Zhou, Yanghua Xiao, and Lei Li. 2020. LOREN: logic enhanced neural reasoning for fact verification. *CoRR*, abs/2012.13577.

Qian Chen, Xiaodan Zhu, Zhen-Hua Ling, Si Wei, Hui Jiang, and Diana Inkpen. 2017. Enhanced LSTM for natural language inference. In *Proceedings of ACL*, pages 1657–1668.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: pre-training of deep bidirectional transformers for language understanding. In *Proceedings of NAACL*, pages 4171–4186.

Andreas Hanselowski, Hao Zhang, Zile Li, Daniil Sorokin, Benjamin Schiller, Claudia Schulz, and Iryna Gurevych. 2018. Ukp-athene: Multi-sentence textual entailment for claim verification. *EMNLP 2018*, page 103.

Kelvin Jiang, Ronak Pradeep, and Jimmy Lin. 2021. Exploring listwise evidence reasoning with T5 for fact verification. In *Proceedings of ACL 2021*, pages 402–410.

Thomas N. Kipf and Max Welling. 2017. Semi-supervised classification with graph convolutional networks. In *Proceedings of ICLR*. OpenReview.net.

Zhenghao Liu, Chenyan Xiong, Maosong Sun, and Zhiyuan Liu. 2020. Fine-grained fact verification with kernel graph attention network. In *Proceedings of ACL*, pages 7342–7351.

Jackson Luken, Nanjiang Jiang, and Marie-Catherine de Marneffe. 2018. Qed: A fact verification system for the fever shared task. In *Proceedings of the First Workshop on FEVER*, pages 156–160.

Yixin Nie, Haonan Chen, and Mohit Bansal. 2019. Combining fact extraction and verification with neural semantic matching networks. In *Proceedings of The 33th AAAI*, pages 6859–6866. AAAI Press.

Ankur P. Parikh, Oscar Täckström, Dipanjan Das, and Jakob Uszkoreit. 2016. A decomposable attention model for natural language inference. In *Proceedings of EMNLP*, pages 2249–2255.

Matthew E. Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer. 2018. Deep contextualized word representations. In *Proceedings of NAACL-HLT 2018*, pages 2227–2237.

Jiasheng Si, Deyu Zhou, Tongzhe Li, Xingyu Shi, and Yulan He. 2021. Topic-aware evidence reasoning and stance-aware aggregation for fact verification. In *Proceedings of ACL 2021*, pages 1612–1622.

Amir Soleimani, Christof Monz, and Marcel Worring. 2020. BERT for evidence retrieval and claim verification. In *Proceedings of ECIR*, pages 359–366.

Robyn Speer, Joshua Chin, and Catherine Havasi. 2017. Conceptnet 5.5: An open multilingual graph of general knowledge. In *Proceedings of AAAI*, pages 4444–4451.

James Thorne, Andreas Vlachos, Christos Christodoulopoulos, and Arpit Mittal. 2018a. FEVER: a large-scale dataset for fact extraction and verification. In *Proceedings of NAACL*, pages 809–819.

James Thorne, Andreas Vlachos, Oana Cocarascu, Christos Christodoulopoulos, and Arpit Mittal. 2018b. The fact extraction and verification (fever) shared task. *EMNLP 2018*, 80(29,775):1.

Petar Velickovic, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Liò, and Yoshua Bengio. 2018. Graph attention networks. In *Proceedings of ICLR*. OpenReview.net.

Hai Wan, Haicheng Chen, Jianfeng Du, Weilin Luo, and Rongzhen Ye. 2021. A dqn-based approach to finding precise evidences for fact verification. In *Proceedings of ACL 2021*, pages 1030–1039.

Zhilin Yang, Zihang Dai, Yiming Yang, Jaime G. Carbonell, Ruslan Salakhutdinov, and Quoc V. Le. 2019. Xlnet: Generalized autoregressive pretraining for language understanding. In *Proceedings of NIPS*, pages 5754–5764.

Deming Ye, Yankai Lin, Jiaju Du, Zhenghao Liu, Peng Li, Maosong Sun, and Zhiyuan Liu. 2020. Coreferential reasoning learning for language representation. In *Proceedings of EMNLP 2020*, pages 7170–7186.

Wenpeng Yin and Dan Roth. 2018. Twowingos: A two-wing optimization strategy for evidential claim verification. In *Proceedings of EMNLP*, pages 105–114.

Takuma Yoneda, Jeff Mitchell, Johannes Welbl, Pontus Stenetorp, and Sebastian Riedel. 2018. Ucl machine reading group: Four factor framework for fact finding (hexaf). In *Proceedings of the First Workshop on FEVER*, pages 97–102.

Wanjun Zhong, Jingjing Xu, Duyu Tang, Zenan Xu, Nan Duan, Ming Zhou, Jiahai Wang, and Jian Yin. 2020. Reasoning over semantic-level graph for fact

checking. In *Proceedings of ACL*, pages 6170–6180.

Jie Zhou, Xu Han, Cheng Yang, Zhiyuan Liu, Lifeng Wang, Changcheng Li, and Maosong Sun. 2019. Gear: Graph-based evidence aggregating and reasoning for fact verification. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 892–901.

Biru Zhu, Xingyao Zhang, Ming Gu, and Yangdong Deng. 2021. Knowledge enhanced fact checking and verification. *IEEE ACM TASLP*, pages 3132–3143.

A Appendices

A.1 Hyperparameters Setting

Hyperparameter Name	GLAF
Batch Size	8
Bert Embedding Size	768
Learning Rate	$2e^{-5}$
Warmup Proportion	0.1
Dropout	0.6
Max Epochs	6
Corpus Size	15
Max Length	140
Pooling k	[10,5,3]
Number of iterations N	2

Table 8: Hyperparameters we used for FEVER.

A.2 Error Analysis

To better understand the limitations of our model, we conduct an error analysis on GLAF. We randomly select 200 incorrectly predicted instances that achieve low test FEVER scores. We report several reasons for the low scores, which can roughly be classified into three categories. 1) Upstream document retrieval and sentence selection components extract insufficient evidence for inferring (56%); 2) Incomplete or even incorrect extraction of evidence clues, which may be due to limitations of the SRL toolkit (28%); 3) Lack of common sense knowledge for the claim verification (16%). For example, the claim states "*The Great Wall is a famous ancient building in China*", while the evidence states "*The Great Wall stretches from Lop Lake to Dandong, which is a famous ancient building*". The model fails to realize that "*Lop Lake*" and "*Dandong*" are located in "*China*" due to the lack of common sense knowledge. Solving this type of errors needs to involve external knowledge (e.g., ConceptNet proposed by (Speer et al., 2017)).

A.3 More complicated cases

For more complicated cases, such as a claim sentence contains multiple predicates, GLAF first parses the sentence to multiple triples by the SRL toolkit, and then verifies them separately before making a combined judgment. For example, "Microsoft was founded by Bill Gates and promoted by Tim Cook" can be parsed to two claim triples: (*Microsoft*, *was founded by*, *Bill Gates*) and (*Microsoft*, *was promoted by*, *Tim Cook*), GLAF first predicts their respective labels and then combines these labels to make the final judgment by Table 9.

Priority	Label 1	Label 2	Final Label
Refutes	0	1	1
Not Enough Info	0	0	0
Supports	1	0	0

Table 9: Example of multi-label decisions. Note the priority is in descending order, i.e., Refutes $>$ Not Enough Info $>$ Supports.

Exploiting Hybrid Semantics of Relation Paths for Multi-hop Question Answering Over Knowledge Graphs

Zile Qiao[1], **Wei Ye**[2,†], **Tong Zhang**[1,3], **Tong Mo**[1]
Weiping Li[1], **Shikun Zhang**[2]
[1] School of Software and Microelectronics, Peking University
[2] National Engineering Research Center for Software Engineering, Peking University
[3] TencentMT Oteam, China
{zileq, wye, zhangtong17, zhangsk}@pku.edu.cn
{motong, wpli}@ss.pku.edu.cn

Abstract

Answering natural language questions on knowledge graphs (KGQA) remains a great challenge in terms of understanding complex questions via multi-hop reasoning. Previous efforts usually exploit large-scale entity-related text corpora or knowledge graph (KG) embeddings as auxiliary information to facilitate answer selection. However, the rich semantics implied in off-the-shelf relation paths between entities is far from well explored. This paper proposes improving multi-hop KGQA by exploiting relation paths' hybrid semantics. Specifically, we integrate explicit textual information and implicit KG structural features of relation paths based on a novel rotate-and-scale entity link prediction framework. Extensive experiments on three existing KGQA datasets demonstrate the superiority of our method, especially in multi-hop scenarios. Further investigation confirms our method's systematical coordination between questions and relation paths to identify answer entities.

1 Introduction

Answering natural language questions on knowledge graphs (KGQA) is a challenging task (Bollacker et al., 2008; Tanon et al., 2016). Recent works mainly pay attention to a complex scenario, namely multi-hop KGQA (Sun et al., 2018; Hu et al., 2018; Saxena et al., 2020; Atzeni et al., 2021), where sophisticated reasoning over multiple edges (or relations) is required to infer the correct answer in the KG (Chen et al., 2019).

The main challenge of multi-hop KGQA is to understand complicated questions and reason under incomplete KG, usually without supervision signals at the intermediate reasoning steps (Lan et al., 2021). One common strategy to alleviate this dilemma is to exploit auxiliary information to enrich knowledge representation. For example, researchers have exploited entity-related textual corpus (e.g., from Wikipedia) as additional nodes in graph-based neural models (Sun et al., 2018, 2019a), or directly encoded them into enhanced entity representations (Han et al., 2020). A more recent effort, namely EmbedKGQA (Saxena et al., 2020), leverages implicit yet rich information in KG embeddings to answer complex questions over sparse KG. Unfortunately, the relation paths, which may contain beneficial supplementary information to characterize a candidate target entity for a topic entity in a question, are commonly underutilized.

To the best of our knowledge, Yan et al. (2021) is the only effort involving exploiting the off-the-shelf relation path information. They use relation paths as simple coarse-grained input features by concatenating their text descriptions. From a more fine-grained and systematic perspective, given a question as a semantic view for the implied relational fact of a <topic entity, target entity> pair, a relation path can serve as another highly-related yet complementary one.

Therefore, we propose coordinating the question view and the relation path view to identify target entities more accurately. To make this idea work, we face two main challenges: 1) how to accurately represent relation paths and 2) how to fuse a relation path representation with the question representation.

For the first problem, we propose exploiting hybrid features of relation paths by integrating both explicit textual semantics and implicit KG embedding features. Firstly, previous works have shown the merits of introducing entity-related texts (Sun et al., 2018; Xiong et al., 2019; Sun et al., 2019a; Han et al., 2020), while we conjecture that relation-related texts (e.g., relation names or descriptions) can potentially offer helpful clues to answer a question. Meanwhile, relation-related texts are naturally available and on a much smaller scale compared with entity-related texts. Therefore, we utilize the

[†]Corresponding authors.

Proceedings of the 29th International Conference on Computational Linguistics, pages 1813–1822
October 12–17, 2022.

explicit text description of a relation path (a relation set) as an extra feature of KGQA models to facilitate target entity selection. Secondly, in addition to a text description, a relation also has a KG-based representation (relation embedding) that implicitly contains rich KG structural semantics. Therefore, we introduce RotatE, a KG embedding model that can well support relation composition by entity rotation in the complex vector space (Sun et al., 2019b). With RotatE, we can synthesize relation path representation by performing simple element-wise multiplication of individual relation embeddings. Finally, we characterize beneficial knowledge in candidate relation paths by fusing their structural and textual representations in a question-aware manner, which also facilitates filtering appropriate relation paths semantically consistent with the question among numerous noisy candidates.

For the second problem, inspired by (Saxena et al., 2020), we project the well-designed question-aware mixed representations of relation paths, as well as the question representation, into a rotating entity link prediction framework. However, our pilot experiments showed that the rotating-based link prediction did not yield the robust performance we expected. Further investigation revealed that the modulus of entity embeddings by RotatE mattered. For example, compared with 1-hop answer entities, 2-hop answer entities more significantly differ from their topic entities on modulus. This observation inspires us to match the integrated semantics of questions and relation paths by both entity rotation and entity modulus scaling. After introducing an entity modulus scaling mechanism, we achieve a promising rotate-and-scale prediction framework, which better coordinate knowledge of questions and relation paths for KGQA.

Extensive experiments on three existing KGQA datasets (WebQuestionSP (tau Yih et al., 2016), ComplexWebQuestions (Talmor and Berant, 2018), and MetaQA (Zhang et al., 2018)) verify the superiority of our method, especially in multi-hop scenarios. Our contributions are as follows:

- We propose a KGQA method from a novel perspective of exploiting hybrid features of the off-the-shelf relation paths.
- By systematically fusing explicit **T**extual information and implicit KG **E**mbedding features of candidate **R**elation **P**aths based on a novel rotate-and-scale KG link prediction framework, our method (**TERP**) achieves competitive performance on three KGQA datasets, especially in the multi-hop scenario.
- We reveal that questions and relation paths, as two facets of their corresponding relations between a topic entity and a target entity, are highly-relevant yet complementary information for question answering.

2 Problem Statement

KGQA is the task of factoid question answering over a knowledge graph. A knowledge graph is denoted as $\mathcal{G} \subseteq \mathcal{E} \times \mathcal{R} \times \mathcal{E}$, where $\mathcal{E}$ is the set of all entities in the KG and $\mathcal{R}$ is the set of all relations. A triple can be formally described as (h, r, t) , where $h, t \in \mathcal{E}$ and $r \in \mathcal{R}$ is the relation between them. Given a natural language question $Q = \left(w_1, \ldots, w_{|q|}\right)$ and a topic entity $h \in \mathcal{E}$, which should be present in the question, the task of KGQA is to extract answer entities $c_a \in \mathcal{E}$ that answer the question Q correctly from $\mathcal{G}$. In practice, we perform entity linking on the question Q, producing a set of topic entities $\mathcal{H}$.

3 Methodology

3.1 Overview

In this section, we mathematically present our TERP method in detail. Our main idea is to coordinate the question view and the relation path view to identify target entities more accurately. Figure 1 shows an overview of our approach. Firstly, we obtain the representations of the entities and relations in KG via a **KG embedding** module and the representations of questions via a **Question Encoder**. Then, we use a **Path Encoder** to encode the relation path by integrating explicit textual semantics and implicit KG embedding features of relation paths. An attention mechanism is employed to choose the appropriate relation paths semantically consistent with the question among numerous noisy candidate paths. A **Rotate-and-Scale** module projects the representations of the question and the chosen relation paths into the complex space of KG embeddings. Finally, an **Entity Predictor** scores all candidate entities in a link prediction manner.

3.2 KG Embedding

We first obtain the representations of entities and relations via a KG Embedding module. To mine the

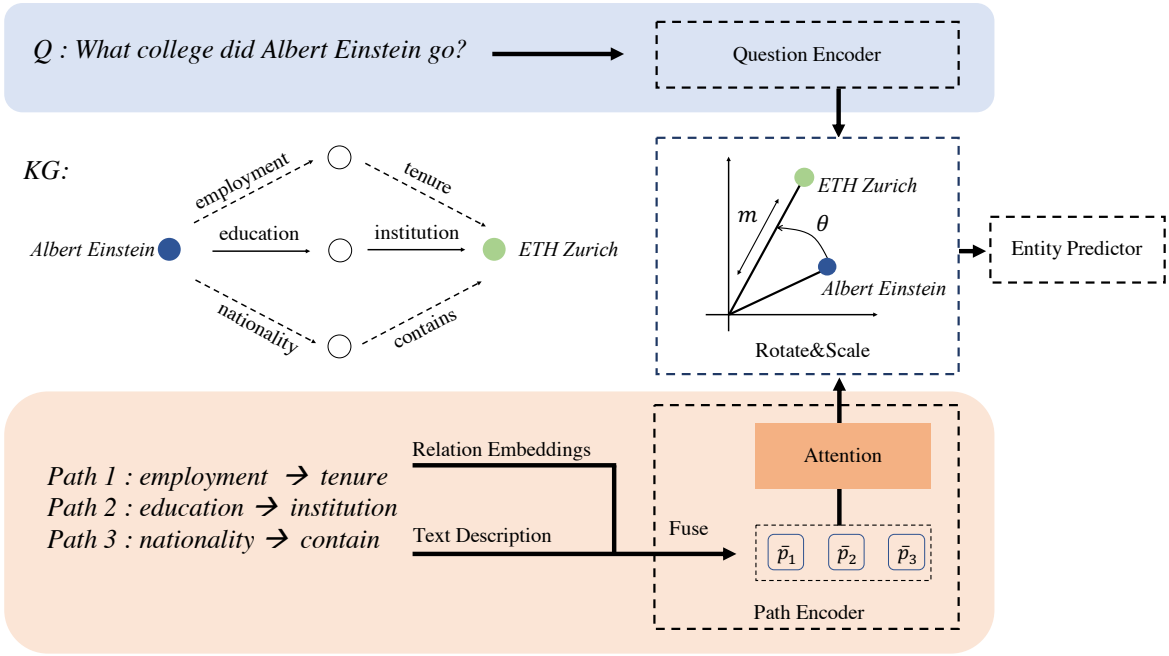

Figure 1: An overview of our TERP model. A question encoder is employed to extract the features from textual question. A path encoder is adopted to capture the consistent yet complementary information in implicit relation embeddings (i.e., structural features) and explicit text description (i.e., textual features) of potential relation paths for the question. Then, the Rotate and Scale mechanism projects the features from question and relation paths to a rotation angle θ and a scaling factor m , respectively. Finally, a entity predictor scores all candidate entities.

implicit KG structural semantics of relation paths, we adopt RotatE (Sun et al., 2019b) to model the composition of relations.

RotatE represents entities as complex vectors and relations as rotations in complex vector space. Given $h, t \in \mathcal{E}$ and $r \in \mathcal{R}$, RotatE generates $e_h, e_r, e_t \in \mathbb{C}^d$ and defines a scoring function:

$$s_r(h,t) = \phi(h,r,t) = -\|e_h \circ e_r - e_t\|, \quad (1)$$

where $|e_{ri}| = 1$, and $\circ$ denotes the Hadmard (or element-wise) product. RotatE can model the composition patterns. A relation $e_{r3} = exp(i\theta_3)$ is a combination of other two relations $e_{r1} = exp(i\theta_1)$ and $e_{r2} = exp(i\theta_2)$ if and only if:

$$e_{r3} = e_{r1} \circ e_{r2}. \quad (2)$$

3.3 Question Encoder

Following previous work (Saxena et al., 2020; Shi et al., 2021; Atzeni et al., 2021), the question encoding model aims to embed a natural language question Q to a fixed dimension vector $q \in \mathbb{R}^d$ with a pre-trained language model.

$$q = \text{Encoder}_{\text{avg}}(Q), \quad (3)$$

where avg denotes the average pooling strategy.

3.4 Path Encoder

In addition to the question, we further model the explicit and implicit semantics in relation path between a topic entity and a candidate entity with a path encoding module. Our intuition behind this is that the textual semantics in the relation path, as well as the corresponding composed pre-trained relation embeddings produced by RotatE, are supplementary information to the question. Considering that the same relation path may have different semantics in different query contexts, we additionally add the question text before the textual description of the relation path. Formally, given a natural language question Q, a topic entity h and a candidate entity c, we can obtain the shortest paths between h and c. For a single path ${r_p}^1, {r_p}^2, \cdots {r_p}^k$, we generate a textual relation path P_t, in which every text description of relation in the path is surrounded by special tokens $<\text{r}>, </\text{r}>$. We concatenate the textual question Q with the textual relation path P_t and feed them into the encoder to extract the textual features of the explicit relation paths. Meanwhile, we can obtain the implicit semantics from the embeddings of the relations ${e_{rp}}^1, {e_{rp}}^2, \cdots {e_{rp}}^k$, which carry structural features learn by RotatE. The hybrid representation $\bar{p}$ of the relation path is pro-

duced via fusing both explicit textual representation p_t and implicit KG embedding representation p_l with an FFN:

$$\begin{aligned}\bar{p}_t &= \text{Encoder}_{\text{avg}}(P_t : Q),\\ \bar{p}_l &= {e_{rp}}^1 \circ {e_{rp}}^2 \circ \cdots \circ {e_{rp}}^k,\\ \bar{p} &= \text{FFN}(\bar{p}_t, \bar{p}_l).\end{aligned} \tag{4}$$

One underlying challenge is that there could be multiple shortest paths between a topic entity and an answer entity. In TERP, we use a scaled dot-product attention mechanism to select the appropriate relation paths that are semantically consistent with the question.

$$p = \text{Attention}(q\mathbf{W}_1, \bar{\mathbf{p}}_\mathbf{t}\mathbf{W}_2, \bar{\mathbf{p}}), \tag{5}$$

where $\bar{\mathbf{p}}_\mathbf{t}$ and $\bar{\mathbf{p}}$ denote two version of representations of all the candidate paths for a given (h, c) pair. $\mathbf{W}_1, \mathbf{W}_2$ are learnable matrices.

3.5 Rotate-and-Scale

However, in our preliminary exploration, directly using RotatE as our KG embedding module do not yield a robust performance. Further investigation revealed that the underlying reason is that the modulus of entities varies in RotatE. If there is an edge between two entities, they will have similar modulus because the modulus of relation representations are fixed to be 1 in RotatE. However, in a multi-hop KBQA scenario, the multi-hop relation path could amplify the difference between the topic and answer entities, hence it can be challenging to match the answer only by rotation transformation.

To this end, we propose a rotate-and-scale framework to model the two views of implied relational fact of a <topic entity, target entity> pair as a rotation transformation and a scaling transformation in the complex space. For a natural language question Q, two independent feedforward networks (FFN) are used to generate the rotation transformation $\theta_q \in \mathbb{R}^d$ and the scaling transformation $m_q \in \mathbb{R}^d$:

$$\begin{aligned}\theta_q &= \text{FFN}(q),\\ m_q &= \text{FFN}(q).\end{aligned} \tag{6}$$

Then, we combine the two transformations into the final representation r_q for the question, which contains a real part Re and an imaginary part Im in the complex space.

$$\begin{aligned}\text{Re}(r_q) &= m_q \circ \cos(\theta_q),\\ \text{Im}(r_q) &= m_q \circ \sin(\theta_q).\end{aligned} \tag{7}$$

In this way, our rotate-and-scale framework can serve as the bridge between entity representations of RotatE and representations of textual questions. We handle relation paths in a same way.

$$\begin{aligned}\theta_p &= \text{FFN}(p),\\ m_p &= \text{FFN}(p),\\ \text{Re}(r_p) &= m_p \circ \cos(\theta_p),\\ \text{Im}(r_p) &= m_p \circ \sin(\theta_p),\end{aligned} \tag{8}$$

where $\theta_p \in \mathbb{R}^d$ and $m_p \in \mathbb{R}^d$ are rotation and scaling in the complex space respectively. As illustrated in Section 5.2, the rotate-and-scale mechanism improves the performance of KGQA with a large margin.

3.6 Entity Predictor

With the representations produced above, an entity predictor is used to score all candidate entities. Given a question Q, the candidate paths P, a topic entity $h \in \mathcal{E}$ and the candidate entity $c \in \mathcal{E}$, the score function is calculated as:

$$\begin{aligned}s_q(h, c) &= -\|e_h \circ r_q - e_c\|,\\ s_p(h, c) &= -\|e_h \circ r_p - e_c\|,\end{aligned} \tag{9}$$

where s_q and s_p denote the scores from the question view and the relation path view, respectively. The final score is $s = (1-\lambda)s_q(h, c)+\lambda s_p(h, c)$, where λ is a hyper-parameter. In training, the score s is calculated among N candidate entities sampled from the KG, where N is a hyper-parameter.

The overall training objective combines the Cross-entropy (**CE**) loss $\mathcal{L}_{ques}$ and $\mathcal{L}_{path}$ for the s_q and s_p, respectively.

$$\begin{aligned}\mathcal{L} &= \mathcal{L}_{ques} + \mathcal{L}_{path}\\ &= \text{CE}(s_q, targets) + \text{CE}(s_p, targets),\end{aligned} \tag{10}$$

where $targets$ denotes the ground truth label.

3.7 Inference

To address the challenge that huge numbers of paths may exist, we propose a two-stage inference strategy to reduce the computational cost. At stage 1, given a question Q, a topic entity h and all the entities in the question-specified subgraph $\mathcal{C} \subseteq \mathcal{E}$, we first compute $s_q(h, c)$ for each $c \in \mathcal{C}$. Then we select top-k candidate entities among them according to $s_q(h, c)$. At stage 2, we compute $s_p(h, c)$ only for the entities recalled in stage 1 and calculate the final score s from them. For questions with more

than one topic entity, we simply average the corresponding s_q and s_p calculated by different topic entities for each candidate c.

This two-stage answer acquisition strategy can empirically deliver a 15-40× inference speed-up on different datasets without sacrificing performance.

4 Experimental Settings

4.1 Datasets

We evaluate our model on three widely-used KGQA datasets, MetaQA (Zhang et al., 2018), WebQuestionsSP (tau Yih et al., 2016) datasets, and Complex WebQuestions (Talmor and Berant, 2018).

MetaQA is a multi-hop KGQA dataset with more than 400k questions, providing a KG with 135k triples, 43k entities and 9 kinds of relations.

WebQuestionSP(WebQSP) is a large scale multi-hop KGQA dataset with 4,737 questions. Following Sun et al. (2018, 2019a), we restrict the KG to be a subset of Freebase which contains all facts that are within 2-hops of any entity mentioned in the questions of WebQSP. Then we use the same PPR algorithm as in Sun et al. (2018) to retrieve a subgraph for each question. We further split the testset on WebQSP into 1- and 2-hop sets based on the inferential chain annotation (tau Yih et al., 2016) in the dataset. Note that this split is just for statistics convenience on testsets. During inference, we do not know whether a question is 1-hop or 2-hop, which is different from the MetaQA settings. Following Sun et al. (2018), we remove half of the triples in the KG to simulate an incomplete KG. We call this setting **WebQSP-50**. We use the same train/dev/test split as Sun et al. (2018).

ComplexWebQuestions(CompWebQ) is created by expanding the question entities or adding constraints to the answers in WebQuestionsSP. The questions require up to 4-hops of reasoning on the KG (He et al., 2021).We handle CompWebQ in the same way as WebQSP except that we limit each subgraph to a maximum of 2000 entities in CompWebQ. On average, there are 1349 entities in each subgraph and the recall of answers is 78.6% .

4.2 Implementation Details

We use the open source implementation of LibKGE (Broscheit et al., 2020) to train the KG embeddings. Following Saxena et al. (2020), the pre-trained KG embeddings are frozen for WebQSP and CompWebQ in training, while tuneable for MetaQA. We use a pre-trained RoBERTa (Liu et al., 2019) as the text encoder. The size of a mini-batch is set to 10. The learning rate is 3e-5 and we adopt the Adam optimizer with β_2 = 0.998. The number of candidate entities for WebQSP and CompWebQ is 20000. For MetaQA, we use all entities in KG as candidate entities. Other hyper-parameters are the same as the default RoBERTa-base configuration. The weight λ for the entity predictor is 0.6. The number of candidate entities retrieved in stage 1 during inference is empirically set to be 15 for WebQSP and MetaQA, and 30 for CompWebQ.

4.3 Baselines

PullNet (Sun et al., 2019a) iteratively retrieves a subgraph from KG to create a question-specific sub-graph and rank the entities by a variant of graph CNN (Kipf and Welling, 2017); **EmbedKGQA** (Saxena et al., 2020) leverages KG embeddings to perform multi-hop KGQA. It adopts ComplEx (Trouillon et al., 2016) KG embeddings to score the entities; **EMQL** (Sun et al., 2020) leverages query embedding method and uses these embeddings to obtain the answers; **TransferNet** (Shi et al., 2021) leverages free texts retrieved from the textual corpus and pre-defined constrained predicates to perform multi-hop reasoning; **BERT-KGQA** (Yan et al., 2021) leverages textual information carried by the nodes and edges to perform KGQA. We choose the original version without additional annotated data for a fair comparison; **SQALER** (Atzeni et al., 2021) addresses KGQA by first performing multi-hop reasoning on the KG and then refining the result with logical reasoning.

5 Experiment Results

5.1 Main Results

Table 1 shows the performance of the baseline models and our methods on three datasets under different settings. We achieve the best performance on four of six tasks. Here we mainly compare our TERP with two lines of works: embedding-based methods (e.g., EmbedKGQA) and path searching methods (e.g., SQALER and TransferNet).

Comparison with embedding-based methods. Except for the similar performance on the MetaQA 1-hop task, TERP significantly outperforms EmbedKGQA on the other tasks. The results verify the effectiveness of incorporating relation path information into the link prediction framework.

Models	MetaQA			WebQSP	WebQSP-50	CompWebQ
	1-hop	2-hop	3-hop			
PullNet*	97.0	99.9	91.4	68.1	51.9	47.2
EmbedKGQA	**97.5**	98.8	94.8	66.6	54.3†	44.7†
EMQL	97.2	98.6	99.1	75.5	-	-
TransferNet*	97.5	**100**	**100**	71.4	-	48.6
BERT-KGQA	-	-	-	71.2	56.7	-
SQALER*	-	99.9	99.9	76.1	55.2	-
TERP(ours)	**97.5**	99.4	98.9	**76.8**	**57.0**	**49.2**

Table 1: Main results on MetaQA, WebQSP, WebQSP-50 and CompWebQ. The numbers reported in the table are hits@1. "†" denotes the result of our re-implementation. Methods that use external corpora are annotated with "*".

Comparison with path searching methods. Generally, TERP performs better on WebQSP and ComWebQ, while SQALER and TransferNet are more competitive on MetaQA. The possible reason is that the link prediction framework relies on high-quality KG embeddings, consequently being more effective for knowledge graphs of a larger scale. Note that the scale of WebQSP's knowledge graph is far more extensive than that of MetaQA's (1.8 M v.s. 43 K of entity number, and 6101 v.s. 9 of relation type number). In addition to better trained KG embeddings, the 6101 relation types of WebQSP mean 6101 relation representations, introducing much richer semantics of relations, compared with the 9 relation representations of MetaQA. The difference between these knowledge graphs roughly explains the comparison results. Furthermore, the results on more challenging tasks (WebQSP and CompWebQ) verify the effectiveness of integrating explicit textual information and implicit KG structural information in KGQA.

Note that several baselines (EMQL, BERT-KGQA and PullNet) can not accurately fall into the above two categories. Compared with them, TERP also achieves competitive results, e.g., the superior hits@1 scores in 4 of the 6 test sets.

5.2 Effectiveness of Roate-and-Scale Mechanism

To reveal how the rotate-and-scale mechanism helps answer reasoning, we replace it with general ComplEx-based matching and RotatE-based (without scaling) matching, achieving two model variants named **w/ ComplEx** and **w/ RotatE** respectively. Table 2 contains two groups of results corresponding to whether or not hybrid features of relation paths is introduced. The observations on the two groups are generally similar, and here we mainly analyze the results in the first group.

First, **w/ RotatE** demonstrates a notable performance degradation compared with **w/ ComplEx**, suggesting that simply replacing ComplEx with RotatE in the link prediction-based KGQA framework, can not satisfy our initial desire to exert relation composition capabilities of RotatE. Second, by incorporating the scaling module into RotatE, **w/ RotatE&Scale** surpasses **w/ ComplEx** with a significant margin. This observation verifies that modulus scaling is necessary to capture relation semantics under the hypothesis of using complex vector rotating to match complex multi-hop questions. Third, the superiority of **w/ RotatE&Scale** over **w/ ComplEx** is more visible on 2-hop questions than that on 1-hop ones, proving that **w/ RotatE&Scale** more accurately distinguish relation path semantics.

5.3 Overall Impacts of Relation Paths' Hybrid Features

Another characteristic of TERP is using hybrid features of relation paths. In Table 2, the models of the second group are ones with relation path features. By comparing the results of the first group and second group in Table 2, we find 1) incorporating relation path information can consistently improve answering questions of different hops under both complete and incomplete KGs, and 2) the improvements on 2-hop questions surpass that on 1-hop ones by a large margin, verifying the potential of relation path information for multi-hop reasoning.

5.4 Ablation Study of Relation Paths' Hybrid Features

We then perform an ablation study on the hybrid features. Table 3 shows two groups of results corresponding to using only textual representations and only structural representations of relation paths,

Models	WebQSP			WebQSP-50		
	All	1-hop	2-hop	All	1-hop	2-hop
w/o path						
w/ ComplEx	72.1	83.6	52.3	53.6	63.8	36.0
w/ RotatE	67.5	79.7	46.5	49.8	60.5	31.4
w/ RotatE&Scale	74.6	84.5	57.5	55.5	64.6	39.8
w/ path						
w/ ComplEx	73.6(+1.5)	84.6(+0.9)	54.6(+2.3)	54.6(+1.0)	64.2(+0.4)	38.0(+2.0)
w/ RotatE	69.3(+1.8)	80.5(+0.8)	50.4(+3.9)	51.0(+0.4)	60.6(+0.1)	34.2(+2.8)
w/ RotatE&Scale	**76.8(+2.2)**	**84.9(+0.4)**	**62.6(+5.1)**	**57.0(+1.5)**	**65.1(+0.5)**	**43.1(+3.3)**

Table 2: Hits@1 on WebQSP datasets in full KG settings (WebQSP) and incomplete KG settings (WebQSP-50). "All", "1-hop", and "2-hop" denote the statistics of 1&2-, 1-, and 2-hop questions of the same task. **w/ path** and **w/o path** denote whether the model is equipped with the path encoder. "w/ ComplEx" and "w/ RotatE" denote the models use ComplEx and RotatE, respectively. "w/ RotatE&Scale" denotes the TERP model with RotatE and the scaling strategy. Numbers in the parentheses denote the hit@1 improvements of **w/ path** over **w/o path**.

respectively. ComplEx does not support relation composition, so we only experiment on RotatE and RotatE&Scale. We have three observations here.

First, both textual and structural features improve model performance, indicating that the two kinds of relation path information benefit answer selection. Second, textual information brings more significant enhancements than structural information. The reasons are two-fold. On the one hand, structural information mainly involves multiplication of relation embedding, which overlaps more with implicit semantics in the link prediction process. On the other hand, textual information provides more complementary knowledge for link prediction, from another modality in a sense. Finally, combining them delivers further improvement, verifying the efficacy of the question-aware fusing process to capture the hybrid semantics.

5.5 Collaboration between Questions and Relation Paths

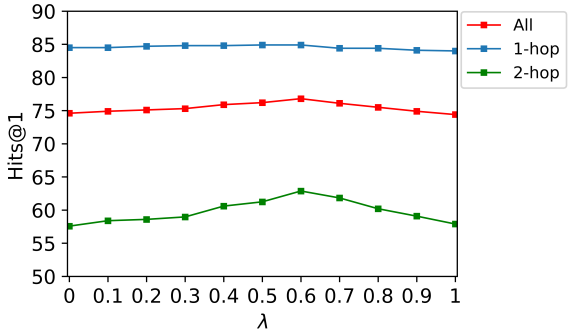

Figure 2: Hits@1 scores for different λ on WebQSP. The blue, green, and red lines denotes the testsets with 1-, 2-, and 1&2-hop questions, respectively.

Considering that exploiting relation paths also introduces many spurious ones, the collaboration of their hybrid features and questions is critical to balance the positive and negative effects. Therefore, we first analyze the impact of the hyper-parameter λ, which denotes the weighting strategy between predicting scores of questions and relation paths.

In Figure 2, the blue, green, and red polylines show the Hits@1 scores of all 1-hop, 2-hop, and 1&2-hops questions on WebQSP, respectively. Looking into these three polylines' trends, we find that our model is best-performed when λ is 0.6, indicating the textual information can not either be ignored or overly dependent. In other words, we need to distinguish necessary features under tolerable noises introduced by a set of off-the-shelf relation paths. Another interesting observation is that the upward trend before the peak of the green line (2-hop questions) is more evident than that of the blue line (1-hop questions), though their downward trends after the peak are similar. The reason is that relation path information is more critical for multi-hop reasoning, and our method well characterizes them, hence delivering robust improvements.

To further investigate how relation paths and questions collaborate, we calculate the cosine similarity between relation path text and question text representations for WebQSP. Since there may be multiple candidate relation paths, the relation path with maximum similarity is selected. We then equally divide data samples in the test set into five groups, based on the cosine similarity scores. The performance of two compared models (**w/ path** and **w/o path**) for each group is shown in Figure 3, from which we observe two interesting trends.

Models	WebQSP			WebQSP-50		
	All	1-hop	2-hop	All	1-hop	2-hop
w/ only textual part						
w/ RotatE	68.4	80.2	48.2	50.4	60.4	33.2
w/ RotatE&Scale	76.1	84.6	61.6	56.1	64.7	41.4
w/ only structural part						
w/ RotatE	67.8	79.7	47.3	50.2	60.4	32.7
w/ RotatE&Scale	75.3	84.7	58.9	53.4	61.8	37.9
w/ both						
w/ RotatE	69.3	80.5	50.4	51.0	60.6	34.2
w/ RotatE&Scale	76.8	84.9	62.6	57.0	65.1	43.1

Table 3: Ablation results. "**w/ only textual part**" denotes the models that only use the textual features. "**w/ only structural part**" denotes the models that only use the structural features. "**w/ both**" denotes the models that use both the structural features and the textual features.

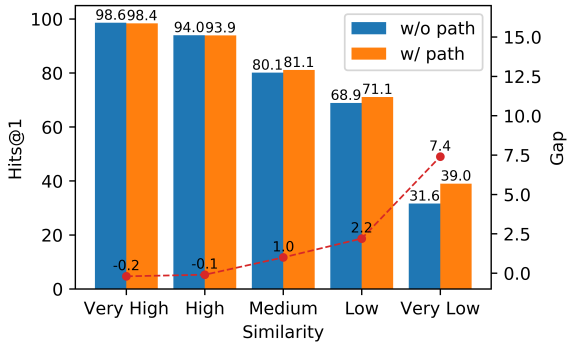

Figure 3: Average Hits@1 scores of **TERP w/o path** and **TERP w/ path** on sub-testsets of WebQSP with different similarities between question and relation paths. The red line denotes the performance gap between the two compared models.

First, model performance degrades as cosine similarity decreases. For example, the hits@1 for "Very High" and "Very Low" differ enormously (e.g., 98.4 v.s. 39.0 with our full TERP). Intuitively, a question is relatively easy to answer if it is similar to a potential relation path. Otherwise, it is more challenging to find the answer. In other words, the question may not provide enough clues, making question understanding more difficult. Second, the relation path information provides more significant improvement for more difficult questions. Incorporating relation path information may even hinder model performance for the groups of "Very High" and "High" (e.g., -0.2 and -0.1 hits@1). This is because many relation paths will bring noises but no extra valuable clues. On the contrary, the hits@1 improvements on "Medium", "Low", and "Very Low" are +1.0, +2.2 and +7.3, respectively. These results clearly demonstrate that relation paths provide complementary information for hard questions, and our method effectively extracts and synthesizes essential features of relation paths. That is where the superiority of our method mainly comes.

6 Related Work

There are two categories of KGQA methods commonly known as semantic parsing-based methods and information retrieval-based methods (Lan et al., 2021). We mainly focus on the second one. Miller et al. (2016) proposes to use Memory Networks to learn dense embeddings of the facts present in the KG to perform QA. Sun et al. (2018, 2019a) create a question-specific subgraph with entities and sentences from the external text corpora and then use a variant of graph CNN to rank the candidate entities. Recently, He et al. (2021) and Shi et al. (2021) utilize path searching methods to perform KGQA. However, they ignore the information in complete relation path. Yan et al. (2021) leverages relation paths to identify answers, but they only explore the textual form of relation. In another line of work, Li et al. (2018) uses TransE (Bordes et al., 2013) to answer the question, but it cannot work in the scenario of KGQA. EmbedKGQA (Saxena et al., 2020) leverages KG embeddings and projects the question into a link prediction framework.

7 Conclusion

We have presented our method for KGQA, which offers a novel perspective of exploiting hybrid features of the off-the-shelf relation paths. We distill essential relation path features by fusing explicit textual information and implicit structural features via a question-aware manner. By project-

ing a natural language question as well as the obtained hybrid features of candidate relation paths into a novel rotate-and-scale entity link prediction framework, we effectively coordinate question and relation paths to select the answer entity. We reveal that questions and relation paths can be seen as two relevant yet complementary facets of their corresponding relations between a topic entity and a target entity.

Acknowledgements

We thank anonymous reviewers for their valuable comments. This research was supported by the National Key Research and Development Program of China (No. 2018YFB1403002).

References

Mattia Atzeni, Jasmina Bogojeska, and Andreas Loukas. 2021. Sqaler: Scaling question answering by decoupling multi-hop and logical reasoning. In *NeurIPS*.

Kurt D. Bollacker, Colin Evans, Praveen K. Paritosh, Tim Sturge, and Jamie Taylor. 2008. Freebase: a collaboratively created graph database for structuring human knowledge. In *SIGMOD Conference*.

Antoine Bordes, Nicolas Usunier, Alberto García-Durán, Jason Weston, and Oksana Yakhnenko. 2013. Translating embeddings for modeling multi-relational data. In *NIPS*.

Samuel Broscheit, Daniel Ruffinelli, Adrian Kochsiek, Patrick Betz, and Rainer Gemulla. 2020. LibKGE - A knowledge graph embedding library for reproducible research. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 165–174.

Zi-Yuan Chen, Chih-Hung Chang, Yi-Pei Chen, Jijnasa Nayak, and Lun-Wei Ku. 2019. UHop: An Unrestricted-Hop Relation Extraction Framework for Knowledge-Based Question Answering. *arXiv:1904.01246 [cs]*. ArXiv: 1904.01246.

Jiale Han, Bo Cheng, and Xu Wang. 2020. Open domain question answering based on text enhanced knowledge graph with hyperedge infusion. In *FINDINGS*.

Gaole He, Yunshi Lan, Jing Jiang, Wayne Xin Zhao, and Ji-Rong Wen. 2021. Improving Multi-hop Knowledge Base Question Answering by Learning Intermediate Supervision Signals. *Proceedings of the 14th ACM International Conference on Web Search and Data Mining*, pages 553–561. ArXiv: 2101.03737.

Sen Hu, Lei Zou, and Xinbo Zhang. 2018. A state-transition framework to answer complex questions over knowledge base. In *EMNLP*.

Thomas Kipf and Max Welling. 2017. Semi-supervised classification with graph convolutional networks. *ArXiv*, abs/1609.02907.

Yunshi Lan, Gaole He, Jinhao Jiang, Jing Jiang, Wayne Xin Zhao, and Ji-Rong Wen. 2021. A survey on complex knowledge base question answering: Methods, challenges and solutions. In *Proceedings of the Thirtieth International Joint Conference on Artificial Intelligence, IJCAI 2021, Virtual Event / Montreal, Canada, 19-27 August 2021*, pages 4483–4491. ijcai.org.

Dingcheng Li, Jingyuan Zhang, and Ping Li. 2018. Representation learning for question classification via topic sparse autoencoder and entity embedding. *2018 IEEE International Conference on Big Data (Big Data)*, pages 126–133.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Roberta: A robustly optimized bert pretraining approach. *ArXiv*, abs/1907.11692.

Alexander H. Miller, Adam Fisch, Jesse Dodge, Amir-Hossein Karimi, Antoine Bordes, and Jason Weston. 2016. Key-value memory networks for directly reading documents. In *EMNLP*.

Apoorv Saxena, Aditay Tripathi, and Partha Talukdar. 2020. Improving Multi-hop Question Answering over Knowledge Graphs using Knowledge Base Embeddings. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 4498–4507, Online. Association for Computational Linguistics.

Jiaxin Shi, Shulin Cao, Lei Hou, Juan-Zi Li, and Hanwang Zhang. 2021. Transfernet: An effective and transparent framework for multi-hop question answering over relation graph. In *EMNLP*.

Haitian Sun, Andrew O. Arnold, Tania Bedrax-Weiss, Fernando Pereira, and William W. Cohen. 2020. Faithful embeddings for knowledge base queries. *arXiv: Learning*.

Haitian Sun, Tania Bedrax-Weiss, and William W. Cohen. 2019a. PullNet: Open Domain Question Answering with Iterative Retrieval on Knowledge Bases and Text. *arXiv:1904.09537 [cs]*. ArXiv: 1904.09537.

Haitian Sun, Bhuwan Dhingra, Manzil Zaheer, Kathryn Mazaitis, Ruslan Salakhutdinov, and William W. Cohen. 2018. Open Domain Question Answering Using Early Fusion of Knowledge Bases and Text. *arXiv:1809.00782 [cs]*. ArXiv: 1809.00782.

Zhiqing Sun, Zhi-Hong Deng, Jian-Yun Nie, and Jian Tang. 2019b. RotatE: Knowledge Graph Embedding by Relational Rotation in Complex Space. *arXiv:1902.10197 [cs, stat]*. ArXiv: 1902.10197.

Alon Talmor and Jonathan Berant. 2018. The web as a knowledge-base for answering complex questions. In *NAACL*.

Thomas Pellissier Tanon, Denny Vrandeić, Sebastian Schaffert, Thomas Steiner, and Lydia Pintscher. 2016. From freebase to wikidata: The great migration. *Proceedings of the 25th International Conference on World Wide Web*.

Wen tau Yih, Matthew Richardson, Christopher Meek, Ming-Wei Chang, and Jina Suh. 2016. The value of semantic parse labeling for knowledge base question answering. In *ACL*.

Théo Trouillon, Johannes Welbl, Sebastian Riedel, Éric Gaussier, and Guillaume Bouchard. 2016. Complex Embeddings for Simple Link Prediction. *arXiv:1606.06357 [cs, stat]*. ArXiv: 1606.06357.

Wenhan Xiong, Mo Yu, Shiyu Chang, Xiaoxiao Guo, and William Yang Wang. 2019. Improving question answering over incomplete kbs with knowledge-aware reader. In *ACL*.

Yuanmeng Yan, Rumei Li, Sirui Wang, Hongzhi Zhang, Zan Daoguang, Fuzheng Zhang, Wei Wu, and Weiran Xu. 2021. Large-scale relation learning for question answering over knowledge bases with pre-trained language models. In *EMNLP*.

Yuyu Zhang, Hanjun Dai, Zornitsa Kozareva, Alex Smola, and Le Song. 2018. Variational reasoning for question answering with knowledge graph. In *AAAI*.

Adaptive Threshold Selective Self-Attention for Chinese NER

Biao Hu[1], **Zhen Huang**[1*], **Minghao Hu**[2*], **Ziwen Zhang**[1], **Yong Dou**[1]
[1]College of Computer, National University of Defense Technology, Changsha, China
[2]Information Research Center of Military Science,
PLA Academy of Military Science, Beijing, China
{hubiao, huangzhen, ziwen, yongdou}@nudt.edu.cn
huminghao16@gmail.com

Abstract

Recently, Transformer has achieved great success in Chinese named entity recognition (NER) owing to its good parallelism and ability to model long-range dependencies, which utilizes self-attention to encode context. However, the fully connected way of self-attention may scatter the attention distribution and allow some irrelevant character information to be integrated, leading to entity boundaries being misidentified. In this paper, we propose a data-driven Adaptive Threshold Selective Self-Attention (ATSSA) mechanism that aims to dynamically select the most relevant characters to enhance the Transformer architecture for Chinese NER. In ATSSA, the attention score threshold of each query is automatically generated, and characters with attention score higher than the threshold are selected by the query while others are discarded, so as to address irrelevant attention integration. Experiments on four benchmark Chinese NER datasets show that the proposed ATSSA brings 1.68 average F1 score improvements to the baseline model and achieves state-of-the-art performance.

1 Introduction

Named Entity Recognition (NER) aims to identify named entities in the given text, including persons, locations, organizations, etc. It plays an important role in downstream natural language processing (NLP) tasks such as information retrieval (Chen et al., 2015), relation extraction (Miwa and Bansal, 2016) and question answering (Diefenbach et al., 2018). Compared with English NER, Chinese NER is more difficult since there is no natural delimiter between words in Chinese sentences.

To integrate word information related to each character and avoid error propagation of word segmentation, word-character lattice is first applied to Chinese NER in Lattice-LSTM (Zhang and Yang, 2018). However, the RNN-based model is hard to model long-range dependencies, and has poor computational efficiency. Recently, Transformer (Vaswani et al., 2017) is widely used in Chinese NER, which utilizes self-attention to encode context. Since vanilla Transformer lacks directionality, TENER (Yan et al., 2019) combines Transformer encoder with direction-aware position encoding to catch directions between characters. FLAT (Li et al., 2020) further exploits span relative position encoding to convert the lattice structure into a flat structure.

Figure 1: A comparison of selective self-attention and fully connected self-attention, where the attention weights are indicated by blue color. We only show the attention distribution of the character "上 (Shang)" in the head that focuses on entities.

Although Transformer-based models perform well on Chinese NER, they have a limitation that the fully connected self-attention drives queries to attend to all characters of the inputs, making irrelevant character-level attention integration, which leads to entity boundaries being misidentified. We argue that, in NER task, entities only depend on the most relevant characters, while others should be ignored. As shown in Figure 1, since the attention of the character "上 (Shang)" attends to all characters, irrelevant character-level information of "大 (Big)" and "众 (Zhong)" is integrated, which may lead the model to misidentify "上海大众 (Shanghai Volkswagen)" as an organization entity rather than "上海 (Shanghai)" as a geopolitical entity.

To avoid irrelevant attention integration, researchers have investigated sparse self-attention. Child et al. (2019) introduce local and strided patterns to select keys for queries. Since the sparsity pattern is fixed, it may lack transferability.

*Corresponding author

Proceedings of the 29th International Conference on Computational Linguistics, pages 1823–1833
October 12–17, 2022.

As alternatives, sparsemax (Martins and Astudillo, 2016) and entmax (Peters et al., 2019) collect most relevant keys via dedicated forward and backward algorithms, which incurs large computational costs. To improve efficiency, explicit sparse Transformer (Zhao et al., 2019) gathers a fixed number of keys with highest attention score through top-k selection at the cost of flexibility. However, none of above works have been applied to Chinese NER.

In this paper, we propose an Adaptive Threshold Selective Self-Attention (ATSSA) mechanism to avoid irrelevant character-level attention integration for Chinese NER. This mechanism is data-driven which assigns a various attention mass to each query in different heads parallelly. Specifically, we generate an attention score threshold for each query of input sequence via a matching matrix (Wang et al., 2018) automatically. Then, keys whose attention score exceeds the threshold are selected, while others are discarded. In the end, the softmax function only performs on those selected query-key pairs, thereby improving attention to relevant keys while avoiding irrelevant character-level attention integration. As shown in Figure 1, the character "上(Shang)" only attends to relevant characters in "上海(Shanghai)", while attention to irrelevant characters in "大众(Public)" is discarded.

We take the advantage of FLAT (Li et al., 2020) in lexicon fusion and apply ATSSA to it. Our code will be released at https://github.com/hubiao20/atssa-ner. The contributions of this paper can be summarized as:

- We propose Adaptive Threshold Selective Self-Attention to avoid irrelevant character-level attention integration for Chinese NER.
- We conduct experiments on four benchmark Chinese NER datasets, experimental results show that ATSSA brings 1.68 average F1 score improvements to the baseline model and achieves state-of-the-art performance.

2 Related Work

Chinese NER with Lattice Structure Since lattice structure can provide rich lexical semantics and boundary information, lattice-based approaches have become the mainstream of Chinese NER. Lattice LSTM (Zhang and Yang, 2018) first proposed a lattice structure to encode all characters and matched words simultaneously. Gui et al. (2019a) combined CNN and rethinking mechanism to encode characters and potential words at different window sizes. Ma et al. (2020) generated the soft-lexicon feature by a static method, simplifying the structure of lattice. The backbone of these methods are RNNs, which are hard to model long-range dependencies. As a solution, models based on GNN (Gui et al., 2019b; Sui et al., 2019; Ding et al., 2019) are proposed. Recently, Transformer has achieved state-of-the-art performance on Chinese NER. FLAT (Li et al., 2020) integrated lattice structure via an ingenious span position encoding. Moreover, to capture fine-grained correlations in word-character spaces, DCSAN (Zhao et al., 2021) leveraged a cross-lattice attention to model dense interactions over lattice structure. However, the fully connected self-attention allows Transformer-based models to integrate some redundant information, which affects the performance on Chinese NER. Compared with traditional self-attention, our proposed ATSSA selectively activates keys and avoids incorrect character-level information fusion.

Selective Self-Attention In many NLP tasks, predictions of models only depend on a small part of the inputs. Many works based on Transformer employed sparse self-attention to discard attention to irrelevant keys. Some data-driven approaches, such as sparsemax (Martins and Astudillo, 2016) and entmax (Peters et al., 2019; Correia et al., 2019) selected keys iteratively, which have large computational costs. To keep computational efficiency, others (Raganato et al., 2020; Child et al., 2019) defined sparse patterns manually. However, the transferability of these methods has not been validated. Besides, Zhao et al. (2019) proposed explicit sparse Transformer which collects a fixed number of keys with highest attention score empirically. Unlike sparse self-attention based models, selective self-attention used an additional controller to select keys dynamically. ReSAN (Shen et al., 2018) used two RSS (Reinforced Sequence Sampling) modules to select keys that need to attend to. GA-Net (Xue et al., 2020) utilized an auxiliary network to generate binary gates to select elements for the backbone attention network. Inspired by the selective idea, we propose the adaptive threshold selective self-attention, which dynamically selects keys by an automatically generated attention threshold.

3 Background

Our proposed ATSSA is applied to FLAT (Li et al., 2020), thus, we briefly introduce the structure of

FLAT in this section. FLAT treats the characters and matched words of the input sequence as spans, and designs an ingenious relative position encoding of spans for lattice structure. Let $head[i]$ and $tail[i]$ denote the head and tail position of span x_i, four relative distances are used to indicate the relation between x_i and x_j:

$$\begin{aligned} d_{ij}^{hh} &= head[i] - head[j] \\ d_{ij}^{ht} &= head[i] - tail[j] \\ d_{ij}^{th} &= tail[i] - head[j] \\ d_{ij}^{tt} &= tail[i] - tail[j] \end{aligned} \quad (1)$$

Then, the span relative position encoding can be calculated by a non-linear transformation of the four distances:

$$R_{ij} = \mathrm{ReLU}(W_r[p_{d_{ij}^{hh}}; p_{d_{ij}^{ht}}; p_{d_{ij}^{th}}; p_{d_{ij}^{tt}}]) \quad (2)$$

where W_r is a learnable parameter, p_d is the sine and cosine functions explained as:

$$\begin{aligned} p_{(d,2i)} &= \sin(d/10000^{2i/d_{model}}) \\ p_{(d,2i+1)} &= \cos(d/10000^{2i/d_{model}}) \end{aligned} \quad (3)$$

where d is the distance acquired by Eq.(1) and i is the index of dimension, $d_{model} = N \times d_{head}$, N is the head number and d_{head} is the dimension of each head.

With the span relative position encoding, the dot-product attention can be calculated as:

$$Attn = \mathrm{softmax}\,(A)\,V \quad (4)$$

$$A_{ij} = Q_i K_j^T + Q_i R_{ij}^T + u K_j^T + v R_{ij}^T \quad (5)$$

$$Q, K, V = EW_q, EW_k, EW_v \quad (6)$$

where E is the embedding of each span. $W_q, W_k, W_v \in \mathbb{R}^{d \times d_{head}}$, $u, v \in \mathbb{R}^{d_{head}}$ are learnable parameters.

4 Adaptive Threshold Selective Self-Attention

Intuitively, keys with high attention score are more relevant than others to the correspondin query and attention should be centralized on these keys. However, in the vanilla Transformer, fully connected self-attention distributes attention to the entire context, even those irrelevant keys, which weakens the attention to critical ones. To this end, we propose the Adaptive Threshold Selective Self-Attention, in which keys with attention score lower than threshold of the corresponding query are discarded before the softmax operation.

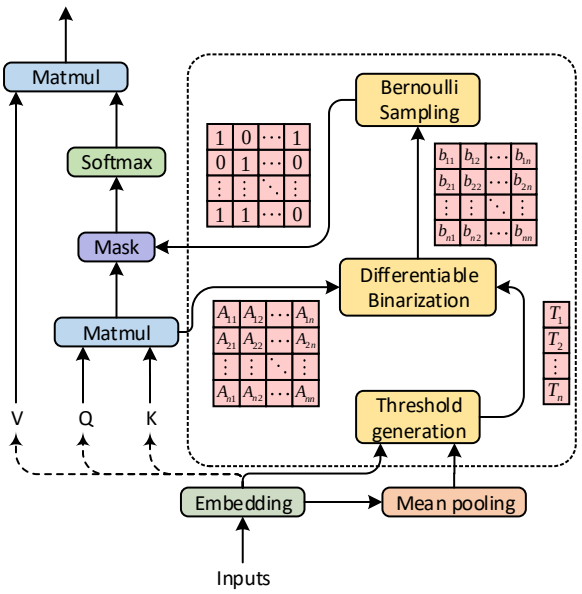

Figure 2: Structure of the proposed adaptive threshold selective self-attention. The dotted box area illustrates the main difference between ATSSA and fully connected self-attention. Threshold is generated by a matching matrix and used for differentiable binarization. Then, the binary matrix is sampled by Bernoulli distribution and exploited to select keys for each query.

4.1 Structure of ATSSA

In this section, we introduce the proposed ATSSA in details, as shown in Figure 2. The goal of the mechanism is to dynamically collect a subset of keys to attend to through an adaptive obtained threshold. Given an input sequence $\mathrm{x} = [x_1, x_2, \cdots, x_n]$, the attention score matrix A can be calculated by dot production of Q and K. Different with the self-attention in vanilla Transformer, the proposed mechanism additionally utilizes an automatically generated threshold $\mathrm{T} = [T_1, T_2, \cdots, T_n]$ to selectively activate elements in A. For each element, $A_{ij} \geq T_i$ means A_{ij} is activated, that is, key K_j is selected by query Q_i.

The threshold acts as an controller to determine whether the information of keys should flow to the target, which is the crux of our proposed mechanism. We argue that each query has a various attention score threshold since its relevance to the input sequence varies. To generate the threshold, we introduce a matching matrix (Wang et al., 2018). The matrix models the correlation between x_i and the entire sequence, and threshold of query Q_i is generated by applying a linear projection to it:

$$T_i = W_t[x_i; \mathrm{mp}(x); x_i \odot \mathrm{mp}(x); x_i - \mathrm{mp}(x)] \quad (7)$$

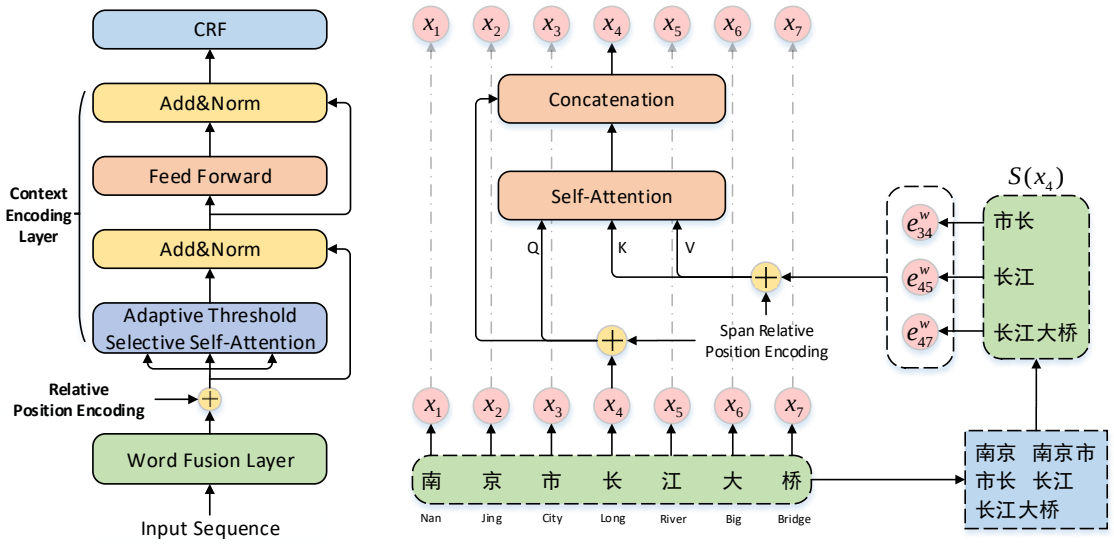

(a) Overall structure (b) Word Fusion Layer

Figure 3: (a) The overall structure of the proposed model. Word information is fused to generate the representation of each character by a word fusion layer. Context is then encoded with the ATSSA based Transformer. (b) Illustration of the word fusion layer, in which word representations only depend on matched words.

where W_t is a learnable parameter, $\odot$ denotes the element-wise product, and $\mathrm{mp}(\cdot)$ denotes the mean pooling operation.

With threshold selection, keys with attention score above or equal to the threshold are kept, while others are dropped. In the process of model training, we discover that the threshold of some queries is higher than the attention score of all keys in the initial stage, leading to the training error of the model. To prevent the loss of critical information, top-k (Zhao et al., 2019) is introduced to force the model to keep at least k keys with largest attention score. Therefore, the threshold of query Q_i can be calculated as:

$$\tilde{T}_i = \min(topk_i, T_i) \tag{8}$$

where $topk_i$ is the k-th largest value at row i in A.

The essence of threshold selection is to perform a binarization operation on attention score matrix A. We employ a binary matrix z to indicate whether the element in matrix A is selected, where $z_{ij} = 1$ means that K_j is selected by query Q_i whereas $z_{ij} = 0$ indicates that K_j is discarded by query Q_i. Since the binary function is not differentiable, we use an approximate step function (Liao et al., 2020) for binarization:

$$b_{ij} = \frac{1}{1 + e^{-\alpha(A_{ij} - \tilde{T}_i)}} \tag{9}$$

where α is amplification factor larger than 1. The approximate step function is similar to the binary function and has differentiability, which enables the model to be optimized during training. b_{ij} is a value in interval $[0, 1]$, not the 0 or 1 we need. To address this issue, we use it to parameterize a Bernoulli distribution, and then, z_{ij} is a sample generated from the Bernoulli distribution:

$$z_{ij} \sim \mathrm{Bernoulli}(b_{ij}) \tag{10}$$

As z is sampled from Bernoulli distribution, our attention strategy can be defined as:

$$Attn = \mathrm{softmax}(A')V \tag{11}$$

$$A'_{ij} = \begin{cases} A_{ij}, & z_{ij} = 1 \\ -\infty, & z_{ij} = 0 \end{cases} \tag{12}$$

4.2 Chinese NER with ATSSA

To adapt ATSSA for Chinese NER, we fine-turn the structure of FLAT. As shown in Figure 3, the context encoder is divided into two layers, consisting of a word-level word fusion layer and a character-level context encoding layer.

Word Fusion Layer For the input sequence x and a given dictionary D, all potential words in x can be identified by matching the sequence with D. Each matched word $w_{jk}(j \neq k)$ starts with the j-th character and ends with the k-th character. Position tags are employed to construct the word set S corresponding to the character x_i:

$$S(x_i) = \{w_{jk} | \forall w_{jk} \in D, j \leq i \leq k\} \tag{13}$$

Take the sequence given in Figure 3(b) as an example, the word set corresponding to the character "长 (Long)" is represented as [市长 (Mayor), 长江 (Yangtze River), 长江大桥 (Yangtze River Bridge)]. For characters with no matched word, word sets corresponding to them are empty, and the word representations of these characters are None.

As the word sets constructed, a character-word cross attention is exploited to generate word representations. Let $head[\cdot]$ and $tail[\cdot]$ denote the head and tail position of the matched word w_{jk}, the relative distances in FLAT can be modified as:

$$\begin{aligned} d_{i,jk}^{hh} &= i - head[w_{jk}] \\ d_{i,jk}^{tt} &= i - tail[w_{jk}] \end{aligned} \tag{14}$$

where $w_{jk} \in S(x_i)$. The relative position encoding of the characters and matched words can be calculated by a simple linear transformation of the two distances:

$$R_{i,jk} = W_r[p_{d_{i,jk}^{hh}}; p_{d_{i,jk}^{tt}}] \tag{15}$$

where W_r is a learnable parameter.

Given characters as queries, words as keys and values, word representation x_i^w of each character can be obtained by self-attention. Finally, x_i^w is concatenated with character x_i as the input of context encoding layer: $x_i = [x_i; x_i^w]$.

Context Encoding Layer In this layer, self-attention based on relative position encoding is introduced. For the input sequence x fused with word information, relative position encoding of each query-key pair can be indicated as $R_{ij} = p_{d_{ij}}$, where $d_{ij} = i - j$, i and j denote the position of query and key respectively. Then, the attention matrix A is generated by Eq.(5). Applying our proposed ATSSA to A, attention of each query is centralized on those critical keys. The following calculation is the same with FLAT.

4.3 Training and Decoding

Since the binary matrix z is sampled by Bernoulli distribution, model cannot be backpropagated during training. Gumbel-Softmax (Jang et al., 2017) provides a reparameterization solution. In Eq.(10), b_{ij} denotes the probability that $z_{ij} = 1$, let $b_{ij}^{(1)} = b_{ij}$ and $b_{ij}^{(0)} = 1 - b_{ij}$, z_{ij} can be expressed as:

$$z_{ij} = \arg\max_k b_{ij}^{(k)}, k = 0, 1 \tag{16}$$

The argmax operation is not differentiable, therefore, we substitute Gumbel-Softmax distribution for Bernoulli distribution to acquire z_{ij} in the training stage. The Gumbel-Softmax distribution makes a softmax approximation to b_{ij} with the following continuous and differentiable calculation:

$$\tilde{b}_{ij}^{(k)} = \frac{\exp((\log(b_{ij}^{(k)}) + g_k)/\tau)}{\sum_{l=0}^{1} \exp((\log(b_{ij}^{(k)}) + g_l)/\tau)} \tag{17}$$

where g_l is a random sample from $\mathrm{Gumbel}(0, 1)$, and τ is a hyperparameter called temperature.

To make use of the dependencies between labels, CRF (Lafferty et al., 2001) is used to predict entity labels. Given the label sequence $y = [y_1, y_2, \cdots, y_n]$ and output $H = [h_1, h_2, \cdots, h_n]$ of the fine-turned model, the probability of the ground-truth label sequence can be calculated as:

$$p(y|\mathrm{x}) = \frac{\exp(\sum_{i=1}^{n} \varphi(y_{i-1}, y_i, \mathrm{x}))}{\sum_{y' \in Y(\mathrm{x})} \exp(\sum_{i=1}^{n} \varphi(y'_{i-1}, y'_i, \mathrm{x}))} \tag{18}$$

where $Y(\mathrm{x})$ is the set of all arbitrary label sequences, $\varphi(y_{i-1}, y_i, \mathrm{x}) = W_{(y_{i-1}, y_i)} h_i + b_{(y_{i-1}, y_i)}$, $W_{(y_{i-1}, y_i)}$ and $b_{(y_{i-1}, y_i)}$ are parameters specific to y_{i-1} and y_i. Therefore, loss function is defined as:

$$\mathcal{L} = -\sum_{i=1}^{n} \log\left(p\left(y_i|\mathrm{x}\right)\right) + \frac{\lambda \|z\|_1}{L} \tag{19}$$

The first term in loss function is negative log-likelihood loss, and the second term is a l_1 norm regularizer over z. λ is a trade-off between the two terms, and L is the length of input sequence.

5 Experiments

Experiments are carried out on Chinese NER datasets across different domains. F1-score (F1) is exploited to evaluate the performance of the model. All experiments are conducted on a single Nvidia Titan RTX GPU.

5.1 Experimental Setup

Datasets We conduct experiments on four datsets, including Weibo NER (Peng and Dredze, 2015; He and Sun, 2017), Resume NER (Zhang and Yang, 2018), OntoNotes (Weischedel et al., 2011), and MSRA (Levow, 2006). Weibo NER is drawn from Sina Weibo[1], and Resume NER is collected from Sina Finance[2], while OntoNotes and MSRA are in news domain. Statistics of the above datasets are shown in Table 1.

[1]https://www.weibo.com/

[2]https://finance.sina.com.cn/stock/

Datasets	Type	Train	Dev	Test
Weibo	Sentence	1.4k	0.27k	0.27k
	Char	73.8k	14.5k	14.8k
Resume	Sentence	3.8k	0.46k	0.48k
	Char	124.1k	13.9k	15.1k
OntoNotes	Sentence	15.7k	4.3k	4.3k
	Char	491.9k	200.5k	208.1k
MSRA	Sentence	46.4k	-	4.4k
	Char	2169.9k	-	172.6k

Table 1: Statistics of datasets

Hyperparameter	Range
learning rate	[1e-3,8e-4,6e-4]
-decay	0.05
head	[4,8,12]
head dimension	[8,10,12]
FFN size	[4,6,8]×head×head dimension
warmup	[1,5,10](epoch)

Table 2: Searching range of hyperparameters.

Baseline We take TENER (Yan et al., 2019) and FLAT (Li et al., 2020) as baseline models, which encode context with Transformer encoder.

Implementation Details We use the pre-trained character embedding and bigram embedding trained with word2vec (Mikolov et al., 2013) over automatically segmented Chinese Giga-Word[3]. The BERT in the experiments is BERT-wwm (Cui et al., 2021), and the pre-trained word embedding is released by (Li et al., 2018) that contains about 1.29 million words. The above four embeddings have sizes of 50, 50, 768 and 300 respectively, all of which are fine-tuned during training. For hyperparameter configurations, batch size is set to 8 and SGD with 0.9 momentum is used to optimize the model. To avoid overfitting, dropout is applied to embeddings with a rate of 0.5. When calculating selective self-attention, each query is forced to keep at least 3 keys, i.e. the value of k in top-k is set to 3. The amplification factor α in approximate step function, temperature τ in Gumbel-Softmax and λ in loss fuction are set to 50, 1 and 4×10^{-6} respectively. We use random search to find the optimal values of other hyperparameters, and their ranges are shown in Table 2.

5.2 Overall Performance

We compare our proposed model with the baseline models and other state-of-the-art word-character lattice based methods, results are reported in Table 3. Our model outperforms TENER by 7.39 average F1 score on four datasets, and for FLAT+BERT, the value is 1.68. In particular, the proposed ATSSA brings improvements of 3.98 and 1.49 on Weibo and OntoNotes respectively. Compared with SoftLexicon (Ma et al., 2020) and DCSAN (Zhao et al., 2021), which statically integrates word-level information, and MECT (Wu et al., 2021) with glyph information, our model is still competitive. The results above indicate the effectiveness of ATSSA, and suggest that ATSSA can better encode context at character-level.

Model	Weibo	Resume	Ontonotes	MSRA
Lattice-LSTM[1]	58.79	94.46	73.88	93.18
SoftLexicon*[2]	70.50	96.11	82.81	95.42
MECT*[3]	70.43	95.98	82.57	96.24
DCSAN*[4]	71.27	96.67	-	96.41
TENER[5]	58.39	95.25	72.82	93.01
FLAT*[6]	68.55	95.86	81.82	96.09
FLAT*+ATSSA	**72.53**	**96.73**	**83.31**	**96.45**

Table 3: Results(F1) on four datasets. * denotes the models equipped with BERT. Zhang and Yang (2018)[1], Ma et al. (2020)[2], Wu et al. (2021)[3], Zhao et al. (2021)[4], Yan et al. (2019)[5], Li et al. (2020)[6].

Models	Weibo	Resume	Ontonotes	MSRA
FLAT*+ATSSA	**72.53**	**96.73**	**83.31**	**96.45**
-Selective SA	70.02	95.52	82.13	96.02
-top-k	68.47	96.19	82.60	95.37
-AF	68.67	95.89	82.65	95.51
-DT	68.53	95.70	82.63	95.68

Table 4: An ablation study of the proposed model. SA stands for Self-Attention, and AF and DT indicates amplification factor and dynamical threshold respectively.

5.3 Ablation Study

To investigate the effectiveness of the main components of our proposed ATSSA , we conduct an ablation study on all four datasets. The results are reported in Table 4.

(1) We propose an adaptive threshold selective self-attention to avoid irrelevant character-level attention integration. To investigate the contribution of this mechanism, we replace selective self-attention with global self-attention. The average F1 score on all four datasets is reduced by 1.29, especially by 2.51 on Weibo test set. The decline in performance verifies the significance of our proposed adaptive threshold selective self-attention.

(2) When calculating selective self-attention, top-k is introduced to force the query to reserve atten-

[3]https://catalog.ldc.upenn.edu/ LDC2011T13

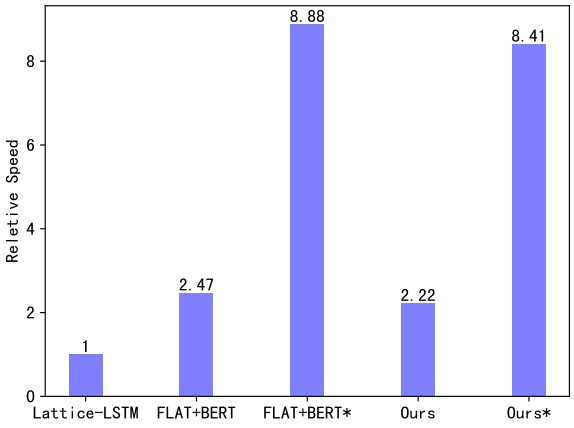

Figure 4: Relative inference speed of each model compared with Lattice-LSTM. The default batch size is 1, * denotes the model is run in 16 batch size.

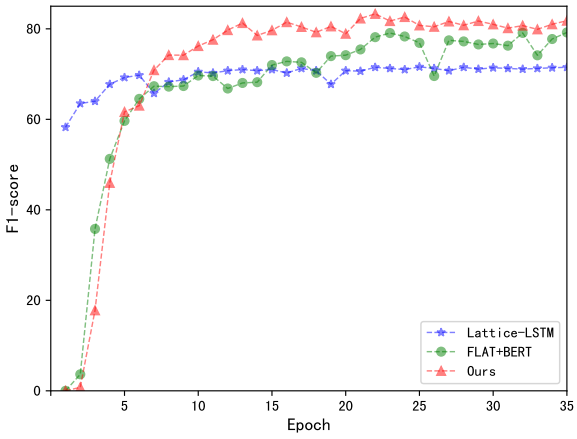

Figure 5: Convergence speed on OntoNotes dataset

Model	Complexity
Vanilla Transformer[1]	$O(n^2)$
Explicit Sparse Transformer[2]	$O(n^2 + nlogn)$
ATSSA	$O(n^2 + 2n + nlogn)$

Table 5: Computational complexity of different methods. Vaswani et al. (2017)[1], Zhao et al. (2019)[2].

tion to a fixed number of keys with largest attention scores. After removing top-k, we discover that the stability of the model training is affected, and F1 scores obtained on the test set are decreased.

(3) The approximate step function is a variant of the sigmoid function, in which an amplification factor makes it more similar to a binary function. Since the use of Bernoulli sampling, the selection of keys with attention score close to the threshold becomes random without the amplification factor. After removing amplification factor, the degradation in performance on all four datasets indicates the importance of amplification factor in approximate step function.

(4) In the computer vision domain, most existing threshold based methods set a fixed threshold, which makes the approach inflexible. Our proposed adaptive threshold selective self-attention assigns a various threshold to each query of all heads in the self-attention that enables the query to select keys dynamically. We empirically replace the dynamic threshold with a fixed threshold 0, and observe that the performance of the model degenerates to be close to FLAT.

5.4 Analysis in Efficiency

Compared with self-attention in vanilla Transformer, the proposed ATSSA has an additional threshold computation, its computational complexity is $O(n^2 + 2n + nlogn)$. The dot products of queries and keys in self-attention correspond to the first term. The threshold calculation corresponds to the second term, that is, the dot product and subtraction of each input character and mean pooling of the entire sequence. The sorting operation in top-k corresponds to the third term. Since the additional term $O(2n + nlogn)$ is overshadowed by the dominant term $O(n^2)$, as shown in Table 4, our proposed ATSSA is slightly slower than FLAT in computational efficiency.

To verify the computational efficiency of ATSSA, we compare the inference speed of Lattice-LSTM, FLAT and our model on Weibo test set, as shown in Figure 4. Lattice-LSTM cannot run in parallel due to the use of directed acyclic graph. As we can see, even with an additional threshold generation operation, our model is only about 5 percent slower in inference speed than FLAT.

To further explore the convergence speed of ATSSA, we conduct experiments on OntoNotes dataset. Figure 5 illustrates the F1 scores of Lattice-LSTM, FLAT and our model relative to the number of training iterations. We observe that the performance of our model is lower than Lattice-LSTM and FLAT in the initial stages of model training. As the number of iterations increases, our model converges faster than FLAT. This is likely because our model is less disturbed by irrelevant character information than FLAT.

5.5 Influence of Hyperparameters

In our proposed ATSSA, k in top-k operation and amplification factor α are two important hyperparameters that affect the performance of Chinese

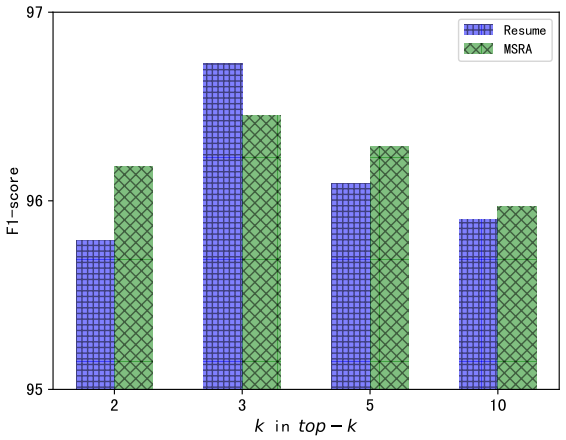

Figure 6: Influence of k value

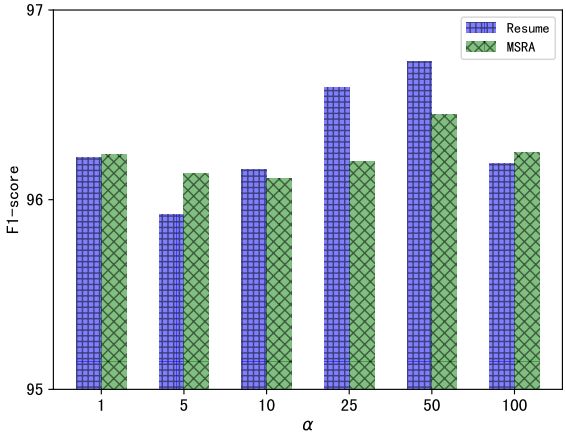

Figure 7: Influence of α value

NER. We conduct experiments on Resume and MSRA datasets to explore the influences by assigning k and α with various values. The results are summarized in Figure 6 and Figure 7.

The top-k operation is used to prevent the loss of critical information at the initial stage of training, it is a modification of the threshold. According to Figure 6, our model achieves the best performance when k is set to 3, which is due to the large proportion of two- and three-character words in Chinese. When k is smaller than 3, some critical information may be lost at context encoding layer, and when k is larger than 3, too much irrelevant character-level information is integrated.

The amplification factor α is utilized to control the similarity between approximate step function and binary function. The larger it is, the higher the similarity. As shown in Figure 7, our model performs best when α is 50. This is because when α is small, the selection or discarding of keys with attention score around the threshold has high uncertainty due to the Bernoulli sampling. When α becomes large, the gradient of the approximate step function (except the neighborhood of point $A_{ij} = T_i$) tends to 0, which brings difficulties to the optimization of the model.

In addition, since the Resume dataset has a relatively smaller scale, the model is more sensitive to the above two hyperparameters on Resume dataset than on MSRA dataset.

5.6 Test of Significance

For a fair comparison with the baseline, we implement a significance test on all four datasets. We randomly select ten various seeds and conduct experiments on our model and FLAT[4]. Then, a paired T-test is performed on each dataset. The p-values obtained on Weibo, Resume, Ontonotes and MSRA datasets are 0.0058, 0.0013, 0.0008 and 0.0161 respectively. Each of them is less than 0.02, which verifies the effectiveness of our proposed ATSSA.

5.7 Case Study

To verify that our proposed ATSSA can better recognize entity boundaries than the fully connected self-attention, we analyze two examples from Weibo test set, as shown in Figure 8. In the first case, due to the use of fully connected self-attention, the two characters "梦(Meng)" and "科(Ke)" as queries attend to the characters in "购物节(Shopping Festival)", which leads to the integration of the character information of "购物节(Shopping Festival)". As a result, FLAT misidentifies the "购物节(Shopping Festival)" as a part of the organization entity "梦科商城购物节(Mengke Mall Shopping Festival)". Our proposed ATSSA uses a threshold to discard characters in "购物节(Shopping Festival)" and correctly identify "梦科商城(Mengke Mall)" as an organization entity. In the second case, the fully connected self-attention assigns attention to each character in "腾讯(Tencent)" and "联想(Lenovo)" and FLAT misidentifies "腾讯联想(Tencent and Lenovo)" as a complete entity. However, ATSSA blocks the interactions between them via the threshold and identifies "腾讯(Tencent)" and "联想(Lenovo)" as two separate organization entity entities. These results show that the adaptive threshold can effectively filter irrelevant character information and help the

[4]https://github.com/LeeSureman/Flat-Lattice-Transformer

Case 1	梦科商城购物节免费送红米											Case 2	腾讯联想联合发起电脑清理日										
	Mengke Mall shopping festival gives Redmi away for free												Tencent and Lenovo co-sponsored Computer Cleanup Day										
Gold Labels	梦	科	商	城	购	物	节	免	费	送	红	米	Gold Labels	腾	讯	联	想	联	合	发	起	电	脑	清	理	日
	B-ORG	I-ORG	I-ORG	I-ORG	0	0	0	0	0	0	0	0		B-ORG	I-ORG	B-ORG	B-ORG	I-ORG	I-ORG	I-ORG	I-ORG	I-ORG	I-ORG	I-ORG	I-ORG	I-ORG
FLAT+BERT	梦	科	商	城	购	物	节	免	费	送	红	米	FLAT+BERT	腾	讯	联	想	联	合	发	起	电	脑	清	理	日
	B-ORG	I-ORG	I-ORG	I-ORG	I-ORG	I-ORG	I-ORG	I-ORG	I-ORG	I-ORG	I-ORG	I-ORG		B-ORG	I-ORG	B-ORG	B-ORG	I-ORG	I-ORG	I-ORG	I-ORG	I-ORG	I-ORG	I-ORG	I-ORG	I-ORG
Ours	梦	科	商	城	购	物	节	免	费	送	红	米	Ours	腾	讯	联	想	联	合	发	起	电	脑	清	理	日
	B-ORG	I-ORG	I-ORG	I-ORG	0	0	0	0	0	0	0	0		B-ORG	I-ORG	B-ORG	I-ORG	I-ORG	I-ORG	I-ORG	I-ORG	I-ORG	I-ORG	I-ORG	I-ORG	I-ORG

Figure 8: Examples of Weibo test set, where blue colors represent the correct labels and red colors represent the wrong labels.

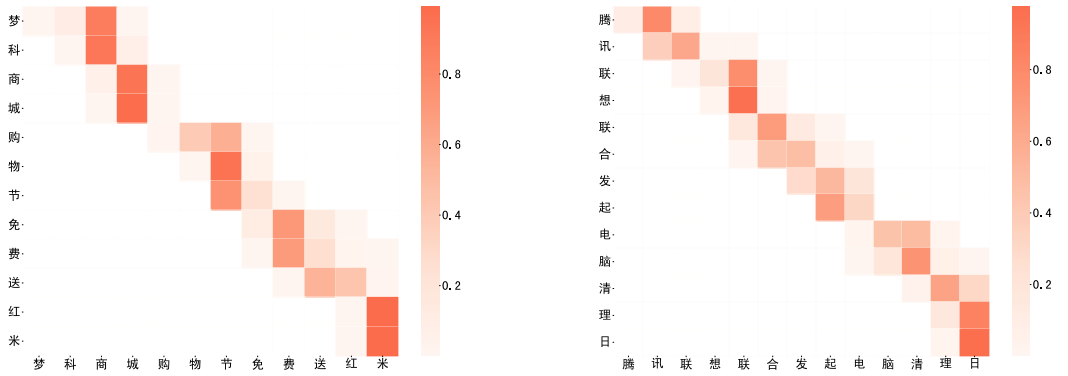

(a) Attention distribution of case 1.

(b) Attention distribution of case 2.

Figure 9: Selective self-attention distributions of the two examples in the head that focus on entities.

model to correctly identify entity boundaries.

Furthermore, we visualize the selective self-attention distributions of these two examples, as shown in Figure 9. In multi-head self-attention, different heads play some specific roles (Voita et al., 2019). For entity recognition, our analysis is based on the head that focuses on entities. In the figure, the vertical axis indicates queries and the horizontal axis represents keys. We observe that in Figure 9(a), the attention of queries in "梦科(Mengke)" is only assigned to keys in "梦科商城(Mengke Mall)", and attention of queries in "购物节(Shoping Festival)" is assigned to other keys, which provide support for the correct identification of the entity "梦科商城(Mengke Mall)". In Figure9(b), even though queries in "腾讯(Tencent)" attend to the key "联(Link)", queries in "联想(Lenovo)" only attend to the keys followed by "腾讯(Tencent)", which separates "腾讯(Tencent)" and "联想(Lenovo)" as two entities. We also observe that each query attends to a different number of keys, which demonstrates the flexibility of the proposed ATSSA.

6 Conclusion

In this paper, we propose an adaptive threshold-selective self-attention mechanism to dynamically select keys for queries in parallel to enhance the architecture of Transformer, which avoids the integration of irrelevant character-level information when encoding context. This data-driven mechanism maintains computational efficiency without losing flexibility. Based on FLAT, we apply it to Chinese NER and achieve state-of-the-art performance on four benchmark Chinese NER datasets. In future work, we will adapt ATSSA to different kinds of NLP tasks, such as text classification and natural language inference.

Acknowledgement

We thank the anonymous reviewers for their helpful comments. This work was supported by the National Natural Science Foundation of China (No. 62006243).

References

Yubo Chen, Liheng Xu, Kang Liu, Daojian Zeng, and Jun Zhao. 2015. Event extraction via dynamic multi-pooling convolutional neural networks. In *ACL*, pages 167–176.

Rewon Child, Scott Gray, Alec Radford, and Ilya Sutskever. 2019. Generating long sequences with sparse transformers. *CoRR*, abs/1904.10509.

Gonçalo M. Correia, Vlad Niculae, and André F. T. Martins. 2019. Adaptively sparse transformers. In

Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing, pages 2174–2184.

Yiming Cui, Wanxiang Che, Ting Liu, Bing Qin, and Ziqing Yang. 2021. Pre-training with whole word masking for chinese BERT. *IEEE ACM Trans. Audio Speech Lang. Process.*, 29:3504–3514.

Dennis Diefenbach, Vanessa López, Kamal Deep Singh, and Pierre Maret. 2018. Core techniques of question answering systems over knowledge bases: a survey. *Knowl. Inf. Syst.*, 55(3):529–569.

Ruixue Ding, Pengjun Xie, Xiaoyan Zhang, Wei Lu, Linlin Li, and Luo Si. 2019. A neural multi-digraph model for chinese NER with gazetteers. In *ACL*, pages 1462–1467.

Tao Gui, Ruotian Ma, Qi Zhang, Lujun Zhao, Yu-Gang Jiang, and Xuanjing Huang. 2019a. Cnn-based chinese NER with lexicon rethinking. In *Proceedings of the Twenty-Eighth International Joint Conference on Artificial Intelligence*, pages 4982–4988.

Tao Gui, Yicheng Zou, Qi Zhang, Minlong Peng, Jinlan Fu, Zhongyu Wei, and Xuanjing Huang. 2019b. A lexicon-based graph neural network for chinese NER. In *EMNLP-IJCNLP*, pages 1040–1050.

Hangfeng He and Xu Sun. 2017. F-score driven max margin neural network for named entity recognition in chinese social media. In *EACL*, pages 713–718.

Eric Jang, Shixiang Gu, and Ben Poole. 2017. Categorical reparameterization with gumbel-softmax. In *5th International Conference on Learning Representations*.

John Lafferty, Andrew McCallum, and Fernando C.N. Pereira. 2001. Conditional random fields: Probabilistic models for segmenting and labeling sequence data. In *Proceedings of the 18th International Conference on Machine Learning*, pages 282–289.

Gina-Anne Levow. 2006. The third international chinese language processing bakeoff: Word segmentation and named entity recognition. In *Proceedings of the Fifth Workshop on Chinese Language Processing*, pages 108–117.

Shen Li, Zhe Zhao, Renfen Hu, Wensi Li, Tao Liu, and Xiaoyong Du. 2018. Analogical reasoning on chinese morphological and semantic relations. In *ACL*, pages 138–143.

Xiaonan Li, Hang Yan, Xipeng Qiu, and Xuanjing Huang. 2020. FLAT: Chinese NER using flat-lattice transformer. In *ACL*, pages 6836–6842.

Minghui Liao, Zhaoyi Wan, Cong Yao, Kai Chen, and Xiang Bai. 2020. Real-time scene text detection with differentiable binarization. In *The Thirty-Fourth AAAI Conference on Artificial Intelligence*, pages 11474–11481.

Ruotian Ma, Minlong Peng, Qi Zhang, Zhongyu Wei, and Xuanjing Huang. 2020. Simplify the usage of lexicon in chinese NER. In *ACL*, pages 5951–5960.

André F. T. Martins and Ramón Fernandez Astudillo. 2016. From softmax to sparsemax: A sparse model of attention and multi-label classification. In *Proceedings of the 33nd International Conference on Machine Learning*, pages 1614–1623.

Tomás Mikolov, Ilya Sutskever, Kai Chen, Gregory S. Corrado, and Jeffrey Dean. 2013. Distributed representations of words and phrases and their compositionality. In *Advances in Neural Information Processing Systems*, pages 3111–3119.

Makoto Miwa and Mohit Bansal. 2016. End-to-end relation extraction using lstms on sequences and tree structures. In *ACL*, pages 1105–1116.

Nanyun Peng and Mark Dredze. 2015. Named entity recognition for chinese social media with jointly trained embeddings. In *ACL*, pages 548–554.

Ben Peters, Vlad Niculae, and André F. T. Martins. 2019. Sparse sequence-to-sequence models. In *Proceedings of the 57th Conference of the Association for Computational Linguistics*, pages 1504–1519.

Alessandro Raganato, Yves Scherrer, and Jörg Tiedemann. 2020. Fixed encoder self-attention patterns in transformer-based machine translation. In *Findings of the Association for Computational Linguistics*, pages 556–568.

Tao Shen, Tianyi Zhou, Guodong Long, Jing Jiang, Sen Wang, and Chengqi Zhang. 2018. Reinforced self-attention network: a hybrid of hard and soft attention for sequence modeling. In *Proceedings of the Twenty-Seventh International Joint Conference on Artificial Intelligence*, pages 4345–4352.

Dianbo Sui, Yubo Chen, Kang Liu, Jun Zhao, and Shengping Liu. 2019. Leverage lexical knowledge for chinese named entity recognition via collaborative graph network. In *EMNLP-IJCNLP*, pages 3828–3838.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in Neural Information Processing Systems*, pages 5998–6008.

Elena Voita, David Talbot, Fedor Moiseev, Rico Sennrich, and Ivan Titov. 2019. Analyzing multi-head self-attention: Specialized heads do the heavy lifting, the rest can be pruned. In *Proceedings of the 57th Conference of the Association for Computational Linguistics*, pages 5797–5808.

Shuohang Wang, Mo Yu, Xiaoxiao Guo, Zhiguo Wang, Tim Klinger, Wei Zhang, Shiyu Chang, Gerry Tesauro, Bowen Zhou, and Jing Jiang. 2018. R^3: Reinforced ranker-reader for open-domain question

answering. In *Proceedings of the Thirty-Second AAAI Conference on Artificial Intelligence,*, pages 5981–5988.

Ralph Weischedel, Sameer Pradhan, Lance Ramshaw, Martha Palmer, Nianwen Xue, Mitchell Marcus, Ann Taylor, Craig Greenberg, Eduard Hovy, and Robert Belvin. 2011. Ontonotes release 4.0. *LDC2011T03, Philadelphia, Penn.: Linguistic Data Consortium*.

Shuang Wu, Xiaoning Song, and Zhen-Hua Feng. 2021. MECT: multi-metadata embedding based cross-transformer for chinese named entity recognition. In *ACL/IJCNLP*, pages 1529–1539.

Lanqing Xue, Xiaopeng Li, and Nevin L. Zhang. 2020. Not all attention is needed: Gated attention network for sequence data. In *The Thirty-Fourth AAAI Conference on Artificial Intelligence*, pages 6550–6557.

Hang Yan, Bocao Deng, Xiaonan Li, and Xipeng Qiu. 2019. TENER: adapting transformer encoder for named entity recognition. *CoRR*, abs/1911.04474.

Yue Zhang and Jie Yang. 2018. Chinese NER using lattice LSTM. In *ACL*, pages 1554–1564.

Guangxiang Zhao, Junyang Lin, Zhiyuan Zhang, Xuancheng Ren, Qi Su, and Xu Sun. 2019. Explicit sparse transformer: Concentrated attention through explicit selection. *CoRR*, abs/1912.11637.

Shan Zhao, Minghao Hu, Zhiping Cai, Haiwen Chen, and Fang Liu. 2021. Dynamic modeling cross- and self-lattice attention network for chinese NER. In *Thirty-Fifth AAAI Conference on Artificial Intelligence*, pages 14515–14523.

Cluster-aware Pseudo-Labeling for Supervised Open Relation Extraction

Bin Duan*, **Shusen Wang***, **Xingxian Liu***, **Yajing Xu**
Pattern Recognition & Intelligent System Laboratory,
Beijing University of Posts and Telecommunications, Beijing, China
{bobtuan,shusenw,liuxingxian,xyj}@bupt.edu.cn

Abstract

Supervised open relation extraction aims to discover novel relations by leveraging supervised data of pre-defined relations. However, most existing methods do not achieve effective knowledge transfer from pre-defined relations to novel relations, they have difficulties generating high-quality pseudo-labels for unsupervised data of novel relations and usually suffer from the error propagation issue. In this paper, we propose a **C**luster-**a**ware **P**seudo-**L**abeling (**CaPL**) method to improve the pseudo-labels quality and transfer more knowledge for discovering novel relations. Specifically, the model is first pre-trained with the pre-defined relations to learn the relation representations. To improve the pseudo-labels quality, the distances between each instance and all cluster centers are used to generate cluster-aware soft pseudo-labels for novel relations. To mitigate the catastrophic forgetting issue, we design the consistency regularization loss to make better use of the pseudo-labels and jointly train the model with both unsupervised and supervised data. Experimental results on two public datasets demonstrate that our proposed method achieves new state-of-the-arts performance[1].

1 Introduction

Open relation extraction (OpenRE) has been proposed to extract the novel relations that are not defined or observed beforehand. Previous methods can be classified into two types: unsupervised and supervised. Unsupervised OpenRE (Yao et al., 2011, 2012; Marcheggiani and Titov, 2016; Elsahar et al., 2017; Tran et al., 2020; Hu et al., 2020) regards the OpenRE as a totally unsupervised task which first extracts the feature and then clusters them. However, these methods don't take full advantage of the large amounts of existing relational facts in knowledge bases. Hence, the supervised OpenRE is proposed which leverages the supervised data of pre-defined relations to guide the learning of the unsupervised data of novel relations. In this paper, we focus on the latter setting, supervised OpenRE.

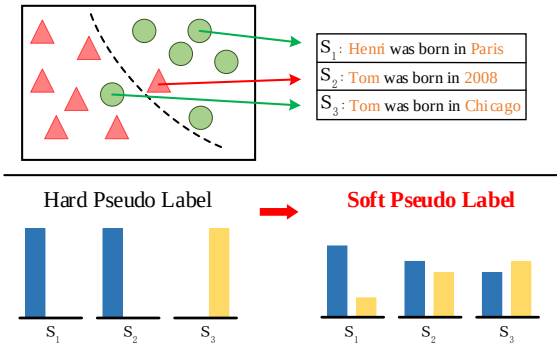

Figure 1: Relation instance S_1 and S_3 belong to the same relation type while S_2 with similar context is from another relation type. However, the hard pseudo-labels produced by common clustering methods contain much noise which causes the error propagation issue. We generate soft pseudo-labels that contain the information about true cluster to improve the pseudo-labels quality.

Since the classes between pre-defined relations and novel relations are disjoint, the main challenge of supervised OpenRE is how to make the best use of the prior knowledge in pre-defined relations to extract novel relations. Wu et al. (2019) proposes relational siamese networks to transfer the knowledge from pre-defined relations to novel relations. However, many studies have shown that high-dimensional embeddings learn much about the complex linguistic information (Peters et al., 2018; Jawahar et al., 2019; Clark et al., 2019; Goldberg, 2019), which makes it hard to produce ideal clusters. Zhao et al. (2021) proposes a relation-oriented clustering method that explicitly aligns the derived clusters with relational semantic classes.

However, we find that the pseudo-labels produced by previous method are not robust to transfer

*The first three authors contribute equally. Yajing Xu is the corresponding author.

[1]Code is available at https://github.com/BobTuan/CaPL

Proceedings of the 29th International Conference on Computational Linguistics, pages 1834–1841
October 12–17, 2022.

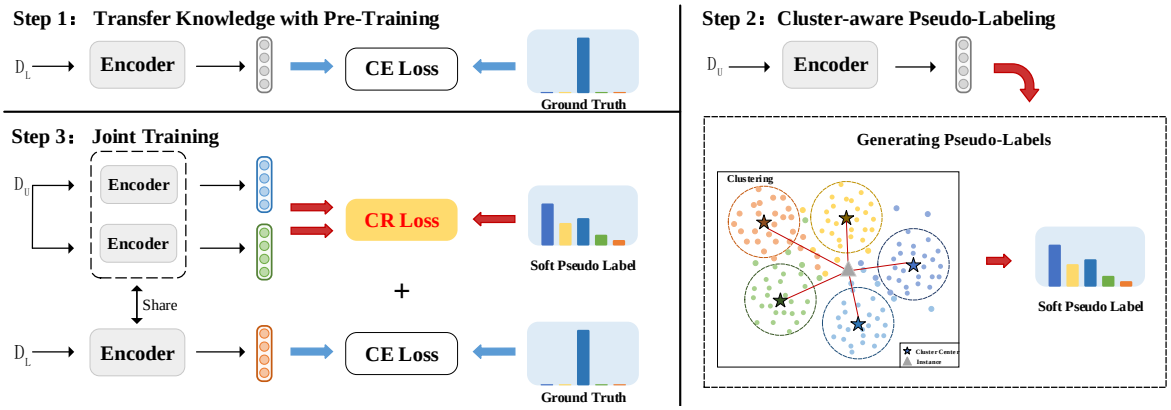

Figure 2: Overview of our CaPL method. Firstly, we pre-train the model with supervised data of pre-defined relations. Then, we generate cluster-aware soft pseudo-labels for unsupervised data of novel relations. Finally, we jointly train the model with both supervised data and unsupervised data. After step 1, step 2 and step 3 are performed iteratively to gradually improve model performance on novel relations.

the knowledge from pre-defined relations to novel relations which means that using unreliable pseudo-labels could cause the error propagation issue. As is shown in Figure 1, relation instance S_1 and S_3 belong to the same relation type *BORN_IN_PLACE* while S_2 with similar context is from another relation type *BORN_IN_DATE*. Owing to the spurious correlations (Liu et al., 2021), existing method that selects the nearest cluster center as the pseudo-labels may produce noise pseudo-labels. Hence if we further exploit the hard pseudo-labels, the model would be wrongly optimized. We argue that utilizing the information about all cluster centers to generate the soft pseudo-labels would be helpful to reduce the error propagation issue. The intuition is that if we exploit the soft pseudo-labels, we can utilize the information about the true cluster to guide the learning for discovering novel relations.

In this paper, we propose a **C**luster-**a**ware **P**seudo-**L**abeling (**CaPL**) method to improve the pseudo-labels quality and transfer more knowledge for discovering novel relations. Firstly, we pre-train the model under the supervision of cross-entropy loss to leverage the prior knowledge in pre-defined relations. Then, to effectively transfer the knowledge, rather than directly using the hard pseudo-labels produced by common clustering algorithms, we use the distances between each instance and all cluster centers to generate cluster-aware soft pseudo-labels for novel relations. Finally, we design consistency regularization loss to make the best use of the knowledge stored in the cluster-aware pseudo-labels and jointly train the model with both supervised and unsupervised data to mitigate the catastrophic forgetting issue.

To summarize, the major contributions of our work are as follows: (1) We propose a simple but effective framework based on the CaPL for supervised OpenRE whicn can transfer more knowledge for discovering novel relations. (2) We design the consistency regularization loss to keep the cluster predictions and pseudo-labels of unsupervised data to be consistent for making better use of the pseudo-labels. (3) Experimental results and analyses on two public datasets demonstrate the effectiveness of our proposed method.

2 Method

In the supervised OpenRE settings, training data is split into two sets: a supervised dataset of pre-defined relations $D_l = \left\{\left(x_i^l, y_i^l\right), i = 1, \ldots, N\right\}$ and an unsupervised dataset of novel relations $D_u = \{x_i^u, i = 1, \ldots, M\}$, where x_i^l in D_l and x_i^u in D_u is an relation instance and y_i^l is a categorical label. Our goal is to cluster the D_u to discover C^u novel relations where we assume C^u to be known a priori. The set of C^l labeled classes is assumed to be disjoint from the set of C^u unlabeled classes.

In this work, we propose a simple but effective framework based on the CaPL to improve the pseudo-labels quality for discovering novel relations. Figure 2 shows the overall architecture of our proposed method. We will introduce these step details in the following subsections.

2.1 Leverage Knowledge with Pre-Training

To leverage the prior knowledge in pre-defined relations, we use the supervised data of pre-defined relations to pre-train the model. The goal of pre-training with pre-defined relations is to make the

model adapt to the relational feature space while be less biased towards pre-defined relations.

Specifically, we learn the relational feature representations under the supervision of cross-entropy loss due to its simplicity and efficacy.

$$\mathcal{L}_{ce} = -\frac{1}{N}\sum_{i=1}^{N}\log\frac{\exp\left(\phi\left(\boldsymbol{s}_i\right)^{y_i}\right)}{\sum_{j=1}^{K}\exp\left(\phi\left(\boldsymbol{s}_i\right)^{j}\right)}, \quad (1)$$

where $\boldsymbol{s}_i$ is the ith relation feature representation of pre-defined relations, $\phi\left(\cdot\right)$ is a linear classifier and $\phi\left(\cdot\right)^j$ are the output logits of the jth class.

2.2 Cluster-aware Pseudo-Labeling

Since there is a mass of unsupervised data of novel relations, it's important to effectively leverage these samples to discover novel relations. Pseudo-Labeling is a well-established technique for transfer learning in general (Cui et al., 2021). After leveraging the prior knowledge in pre-defined relations with pre-training, we propose a simple but effective method CaPL for transferring the knowledge to discover novel classes.

To transfer more knowledge, rather than directly generating the hard pseudo-labels with common clustering algorithm like Zhao et al. (2021), we generate more robust cluster-aware soft pseudo-labels. More specifically, we first obtain the relation instance representations $H = \{h_1, \ldots, h_N\}$, and then perform k-means algorithm in the relational feature space to obtain K cluster centers, denoted as μ_k, $k \in \{1, \ldots, K\}$. Different from that the standard k-means algorithm regards the indicator of the nearest cluster center as the hard pseudo-labels, we adopt a soft assignment to K cluster centers for each instance. Inspired by Hu et al. (2020), we use the Student's t-distribution to compute the assigning probability between relation instance h_j and each cluster center μ_k:

$$p_{jk} = \frac{\left(1 + \|h_j - \mu_k\|_2^2/\alpha\right)^{-\frac{\alpha+1}{2}}}{\sum_{k'=1}^{K}\left(1 + \|h_j - \mu_{k'}\|_2^2/\alpha\right)^{-\frac{\alpha+1}{2}}}, \quad (2)$$

where α denotes the degree of freedom of the Student's t-distribution and p_{jk} can be regarded as the probability of assigning the sample j to the cluster center k. Without explicit mention, we set α = 1 for all experiments in this paper. In addition, we can also use other common clustering algorithms to generate the cluster centers.

2.3 Joint Training

Conventional cross-entropy loss cannot work with the cluster-aware pseudo-labels. To make better use of the knowledge in the pseudo-labels, we design the consistency regularization loss. The idea of consistency is that the cluster prediction and pseudo-labels on a relation instance h_j and its transformed counterpart $h_j^{'}$ should be the same. In our case, we use dropout twice to get h_j's transformed counterpart $h_j^{'}$ like Gao et al. (2021) and then map these relation representations into the cluster predictions q_j and $q_j^{'}$ with the same equation 2. Finally, we use the KL-divergence to keep the consistency between cluster predictions and pseudo-labels:

$$\ell_j = \mathrm{KL}\left[p_j \| q_j\right] + \mathrm{KL}\left[p_j \| q_j^{'}\right] + \mathrm{KL}\left[q_j \| q_j^{'}\right] \quad (3)$$

$$\mathrm{KL}\left[p_j \| q_j\right] = \sum_{k=1}^{K} p_{jk}\log\frac{p_{jk}}{q_{jk}} \quad (4)$$

$$\mathcal{L}_{cr} = \sum_{j=1}^{N}\frac{\ell_j}{N}, \quad (5)$$

To let the supervised data of pre-defined relations better guide the learning of discovering novel relations and mitigate the catastrophic forgetting issue, we jointly train the model with both supervised and unsupervised data. The overall loss is as follows:

$$\mathcal{L} = \mathcal{L}_{ce} + \omega(r)\mathcal{L}_{cr}, \quad (6)$$

where $\omega(r)$ is a ramp-up function slowly increasing from 0 to 1 along with the training. Following Zhao et al. (2021), we use the sigmoid-shaped function $\omega(r) = \lambda e^{-5\left(1-\frac{r}{T}\right)^2}$, where r is current epoch and T is ramp-up length and a positive scalar factor λ.

3 Experiments

3.1 Datasets

To assess the performance of our method, we conduct experiments on two relation extraction datasets: FewRel (Han et al., 2018) and TACRED (Zhang et al., 2017). **FewRel** is a human-annotated dataset which contains 80 types of relations, each with 700 relation instances. **TACRED** is also a human-annotated dataset with 41 relation types. Following the setup of Zhao et al. (2021), we split the FewRel dataset into 64 pre-defined relations and 16 novel relations and randomly select 1,600 instances in novel relations as the test set. For TACRED, we also remove the instances labeled as

Dataset	Method	B^3			V-measure			ARI	Avg.
		Prec.	Rec.	F1.	Homo.	Comp.	F1.		
FewRel	VAE(Marcheggiani and Titov, 2016)	0.309	0.446	0.365	0.448	0.500	0.473	0.291	0.376
	RW-HAC(Elsahar et al., 2017)	0.256	0.492	0.337	0.391	0.485	0.433	0.250	0.340
	Etype+(Tran et al., 2020)	0.238	0.485	0.319	0.364	0.463	0.408	0.249	0.325
	SelfORE(Hu et al., 2020)	0.672	0.685	0.678	0.779	0.788	0.783	0.647	0.703
	RSN(Wu et al., 2019)	0.486	0.742	0.589	0.644	0.787	0.708	0.453	0.583
	RSN-BERT	0.585	0.899	0.709	0.696	0.889	0.781	0.532	0.674
	RoCORE(Zhao et al., 2021)	0.752	0.846	0.796	0.838	0.883	0.860	0.709	0.788
	CaPL	0.815	0.822	**0.819**	0.875	0.873	**0.889**	**0.794**	**0.834**
	CaPL w/o Pre-training	0.735	0.802	0.767	0.834	0.865	0.850	0.693	0.770
	CaPL w/o Consistency Regularization	0.752	0.785	0.768	0.840	0.855	0.847	0.738	0.784
	CaPL w/o Joint Training	0.768	0.820	0.793	0.845	0.875	0.860	0.718	0.790
TACRED	VAE(Marcheggiani and Titov, 2016)	0.247	0.564	0.343	0.208	0.362	0.264	0.159	0.255
	RW-HAC(Elsahar et al., 2017)	0.426	0.633	0.509	0.469	0.597	0.526	0.281	0.439
	Etype+(Tran et al., 2020)	0.302	0.803	0.439	0.260	0.607	0.364	0.143	0.315
	SelfORE(Hu et al., 2020)	0.576	0.510	0.541	0.630	0.608	0.619	0.447	0.536
	RSN(Wu et al., 2019)	0.628	0.634	0.631	0.624	0.663	0.643	0.459	0.578
	RSN-BERT	0.795	0.878	0.834	0.849	0.870	0.859	0.756	0.816
	RoCORE(Zhao et al., 2021)	0.871	0.849	0.860	0.895	0.881	0.888	0.812	0.853
	CaPL	0.858	0.888	**0.873**	0.891	0.906	**0.898**	**0.829**	**0.867**
	CaPL w/o Pre-training	0.834	0.847	0.840	0.868	0.870	0.869	0.789	0.833
	CaPL w/o Consistency Regularization	0.856	0.795	0.824	0.883	0.843	0.862	0.743	0.810
	CaPL w/o Joint Training	0.835	0.827	0.831	0.870	0.855	0.862	0.788	0.827

Table 1: Experimental results produced by baselines and proposed model on FewRel and TACRED in terms of B^3, V-measure, ARI and average performance. The horizontal line divides unsupervised and supervised methods.

no_relation. We seperately select 30 pre-defined relations and 10 novel relations. In addition, we randomly select 15% of the instances from the novel relations as the test set.

3.2 Baselines

For comparison, we consider both unsupervised and supervised OpenRE baselines for comparison:

- **Unsupervised.** We first compare with unsupervised OpenRE methods. **VAE** (Marcheggiani and Titov, 2016) proposes a VAE-based model learned by the self-supervised signals. **RW-HAC** (Elsahar et al., 2017) first extracts entity types and re-weights the word embeddings and then clusters them. **Etype+** (Tran et al., 2020) solely uses entity types as the input. **SelfORE** (Hu et al., 2020) proposes a self-supervised framework which learns the embeddings with pseudo-labels.

- **Supervised.** We also compare our method with supervised OpenRE methods. **RSN** (Wu et al., 2019) proposes the relation similarity metrics to transfer the knowledge to discover novel relations. **RSN-BERT** replaces the static word embeddings with the pre-trained BERT embeddings for a fair comparison. **RoCORE** (Zhao et al., 2021) proposes a relation-oriented method to explicitly align the derived clusters with relational semantic classes.

3.3 Implement Details

We use the pre-trained model (bert-base-uncased[2], with 12-layer transformer) as our network backbone. To avoid overfitting and improve the training efficiency, as suggested in Zhao et al. (2021), we freeze all the parameters of BERT and only fine-tune the parameters of the layer 8. The training batch size is 128, the learning rate is 1e-4, and we use Adam (Kingma and Ba, 2014) as optimizer. All experiments are conducted by using a GeForce RTX 3090Ti with 24 GB memory.

3.4 Main Results

The main results are shown in Table 1. The proposed method CaPL achieves SOTA performance in all datasets and evaluation metrics. It demonstrates the effectiveness that our method leverages the pre-defined relations to extract novel relations. In addition, we find that most supervised methods perform better than unsupervised methods. It indicates that transferring knowledge from pre-defined relations is helpful to discover novel relations.

3.5 Ablation Analysis

To study the effect of different components in CaPL, we conduct ablation experiments. From Table 1, we find that the performance of CaPL will severely degrade without these modules, which

[2]https://huggingface.co/bert-base-uncased

Task	Method	Prec.	Rec.	F1.	Avg.
FewRel	RoCORE	0.752	0.846	0.796	0.788
	CaPL-hard	0.753	0.811	0.781	0.781
	CaPL-soft	0.815	0.822	**0.819**	**0.834**
TACRED	RoCORE	0.871	0.849	0.860	0.853
	CaPL-hard	0.832	0.850	0.841	0.827
	CaPL-soft	0.858	0.888	**0.873**	**0.867**

Table 2: Experimental results with different pseudo-labels under the same pre-training setting on FewRel and TACRED. CaPL-hard adopts the same hard pseudo-labels with RoCORE while our method CaPL-soft adopts the cluster-aware soft pseudo-labels. This table only lists the results of metric B^3. For results of other metrics, please refer to the Appendix D.

Task	Method	Prec.	Rec.	F1.	Avg.
F → T	RSN	0.349	0.590	0.439	0.387
	RSN-BERT	0.337	0.866	0.486	0.400
	RoCORE	0.621	0.602	0.611	0.642
	CaPL	0.813	0.601	**0.691**	**0.847**
T → F	RSN	0.225	0.529	0.316	0.359
	RSN-BERT	0.261	0.861	0.400	0.438
	RoCORE	0.687	0.766	0.724	0.796
	CaPL	0.722	0.757	**0.739**	**0.802**

Table 3: Results on two cross-domain tasks. F means FewRel, which is from encyclopedia domain. T means TACRED, which is from news and web domain. This table only lists the results of metric B^3. For results of other metrics, please refer to the Appendix D.

demonstrates that all modules are important to the final model performance. It is worth noting that without consistency regularization the performance is seriously hurt which indicates that the loss we designed makes better use of the pseudo-labels. Further study about consistency regularization can be found in Appendix B.

3.6 Effect of Pseudo-labels Quality

In this section, we analyse the effect of pseudo-labels quality for transferring knowledge to discover novel relations. Specifically, we adopt the same method as RoCORE to generate and utilize the hard pseudo-labels and combine it into our framework which is named as CaPL-hard. From Table 2, we can see that under the same pre-training setting, our method that generates and utilizes cluster-aware soft pseudo-labels significantly outperforms the CaPL-hard method which indicates that our method generates the high-quality pseudo-labels and makes the best use of them for discovering novel relations.

3.7 Cross Domain Analysis

To further study the knowledge transfer ability, we adopt more strict cross-domain settings to evaluate

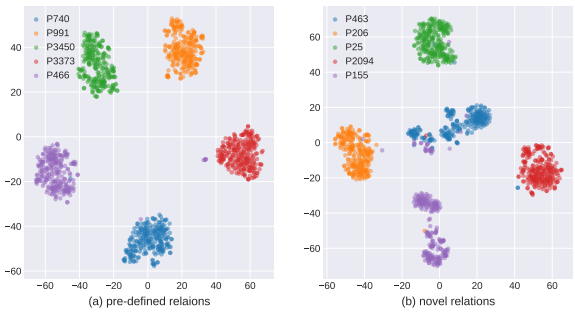

Figure 3: Visualization of the learned relation representations by CaPL on both pre-defined and novel relations.

the model in which the pre-defined and novel relations come from different domains. Specifically, we conduct two cross-domain tasks: FewRel to TACRED and TACRED to FewRel. In Table 3, we observe that in the more realistic cross-domain settings, our model shows better generalization performance on novel relations which indicates that our method effectively transfers the knowledge to discover novel relations.

3.8 Visualization Analysis

To explore the effectiveness on the refinement of relation representations in semantic space, we visualize the representations of both pre-defined and novel relations. We randomly choose 5 relations and sample 250 instances for each relation separately in pre-defined and novel relations. As is shown in Figure 3, the relation representations from both pre-defined and novel relations are mostly separated in our proposed method which means that our method not only fully leverages the prior knowledge for discovering novel relations but also mitigates the catastrophic forgetting issue for pre-defined relations.

4 Conclusion

In this paper, we propose an effective framework based on Cluster-aware Pseudo-Labeling (CaPL) to transfer more knowledge for discovering novel relations. Our main contribution is to improve the knowledge transfer ability of the model. The proposed method makes better use of the prior knowledge in pre-defined relations and generalizes to novel relations with the high-quality pseudo-labels. Experiments and analyses confirm the effectiveness of CaPL for supervised OpenRE.

Acknowledgements

We thank all anonymous reviewers for their valuable suggestions. This work was supported by the National Natural Science Foundation of China (NSFC No.62076031).

References

Amit Bagga and Breck Baldwin. 1998. Entity-based cross-document coreferencing using the vector space model. In *36th Annual Meeting of the Association for Computational Linguistics and 17th International Conference on Computational Linguistics, Volume 1*, pages 79–85, Montreal, Quebec, Canada. Association for Computational Linguistics.

Kevin Clark, Urvashi Khandelwal, Omer Levy, and Christopher D. Manning. 2019. What does BERT look at? an analysis of BERT's attention. In *Proceedings of the 2019 ACL Workshop BlackboxNLP: Analyzing and Interpreting Neural Networks for NLP*, pages 276–286, Florence, Italy. Association for Computational Linguistics.

Li Cui, Deqing Yang, Jiaxin Yu, Chengwei Hu, Jiayang Cheng, Jingjie Yi, and Yanghua Xiao. 2021. Refining sample embeddings with relation prototypes to enhance continual relation extraction. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 232–243, Online. Association for Computational Linguistics.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, Minneapolis, Minnesota. Association for Computational Linguistics.

Hady Elsahar, Elena Demidova, Simon Gottschalk, Christophe Gravier, and Frederique Laforest. 2017. Unsupervised open relation extraction. In *European Semantic Web Conference*, pages 12–16. Springer.

Tianyu Gao, Xingcheng Yao, and Danqi Chen. 2021. SimCSE: Simple contrastive learning of sentence embeddings. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 6894–6910, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Yoav Goldberg. 2019. Assessing bert's syntactic abilities. *arXiv preprint arXiv:1901.05287*.

Xu Han, Hao Zhu, Pengfei Yu, Ziyun Wang, Yuan Yao, Zhiyuan Liu, and Maosong Sun. 2018. FewRel: A large-scale supervised few-shot relation classification dataset with state-of-the-art evaluation. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 4803–4809, Brussels, Belgium. Association for Computational Linguistics.

Xuming Hu, Lijie Wen, Yusong Xu, Chenwei Zhang, and Philip S. Yu. 2020. Selfore: Self-supervised relational feature learning for open relation extraction. *CoRR*, abs/2004.02438.

Lawrence Hubert and Phipps Arabie. 1985. Comparing partitions. *Journal of classification*, 2(1):193–218.

Ganesh Jawahar, Benoît Sagot, and Djamé Seddah. 2019. What does BERT learn about the structure of language? In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 3651–3657, Florence, Italy. Association for Computational Linguistics.

Diederik P Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*.

Fangchao Liu, Lingyong Yan, Hongyu Lin, Xianpei Han, and Le Sun. 2021. Element intervention for open relation extraction. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 4683–4693, Online. Association for Computational Linguistics.

Diego Marcheggiani and Ivan Titov. 2016. Discrete-state variational autoencoders for joint discovery and factorization of relations. *Transactions of the Association for Computational Linguistics*, 4:231–244.

Matthew E. Peters, Mark Neumann, Luke Zettlemoyer, and Wen-tau Yih. 2018. Dissecting contextual word embeddings: Architecture and representation. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 1499–1509, Brussels, Belgium. Association for Computational Linguistics.

Andrew Rosenberg and Julia Hirschberg. 2007. V-measure: A conditional entropy-based external cluster evaluation measure. In *Proceedings of the 2007 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning (EMNLP-CoNLL)*, pages 410–420, Prague, Czech Republic. Association for Computational Linguistics.

Thy Thy Tran, Phong Le, and Sophia Ananiadou. 2020. Revisiting unsupervised relation extraction. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7498–7505, Online. Association for Computational Linguistics.

Ruidong Wu, Yuan Yao, Xu Han, Ruobing Xie, Zhiyuan Liu, Fen Lin, Leyu Lin, and Maosong Sun. 2019. Open relation extraction: Relational knowledge transfer from supervised data to unsupervised data. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 219–228, Hong Kong, China. Association for Computational Linguistics.

Limin Yao, Aria Haghighi, Sebastian Riedel, and Andrew McCallum. 2011. Structured relation discovery using generative models. In *Proceedings of the 2011 Conference on Empirical Methods in Natural Language Processing*, pages 1456–1466, Edinburgh, Scotland, UK. Association for Computational Linguistics.

Limin Yao, Sebastian Riedel, and Andrew McCallum. 2012. Unsupervised relation discovery with sense disambiguation. In *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 712–720, Jeju Island, Korea. Association for Computational Linguistics.

Yuhao Zhang, Victor Zhong, Danqi Chen, Gabor Angeli, and Christopher D. Manning. 2017. Position-aware attention and supervised data improve slot filling. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 35–45, Copenhagen, Denmark. Association for Computational Linguistics.

Jun Zhao, Tao Gui, Qi Zhang, and Yaqian Zhou. 2021. A relation-oriented clustering method for open relation extraction. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 9707–9718, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

A Evaluation Metrics

For evaluation metrics, we adopt B^3 (Bagga and Baldwin, 1998), V-measure (Rosenberg and Hirschberg, 2007), and Adjusted Rand Index (ARI) (Hubert and Arabie, 1985), the same as Wu et al. (2019); Zhao et al. (2021). We take the average for comprehensive evaluation, since that any of the three metrics can measure the clustering performance from different angles.

B Effect of Consistency Regularization Loss

Table 4 shows the effect of consistency regularization loss for exploiting the pseudo-labels. We individually add different components of the consistency regularization loss in equation 3, including the consistency between cluster predictions and cluster predictions (CC) and the consistency between cluster predictions and pseudo-labels (CP). We can observe that both CC and CP consistency are helpful to extract novel relations while CP consistency performs better than CC consistency. We argue it's the knowledge in the pseudo-labels that makes it.

Dataset	Method	Prec.	Rec.	F1.	Avg.
FewRel	Baseline	0.752	0.785	0.768	0.784
	+ CC	0.747	0.822	0.783	0.786
	+ single CP	0.753	0.832	0.790	0.790
	+ double CP	0.794	0.802	0.798	0.812
	+ double CP + CC	0.815	0.822	**0.819**	**0.834**
TACRED	Baseline	0.856	0.795	0.824	0.810
	+ CC	0.845	0.823	0.835	0.830
	+ single CP	0.842	0.856	0.849	0.841
	+ double CP	0.844	0.871	0.857	0.849
	+ double CP + CC	0.858	0.888	**0.873**	**0.867**

Table 4: Evaluation of the effectiveness of the proposed consistency regularization loss. **Baseline**: CaPL without consistency regularization loss, **CC**: consistency between cluster predictions and cluster predictions, **CP**: consistency between cluster predictions and pseudo-labels. This table only lists the results of metric B^3. Refer to Table 5 for detailed results.

C More details about Encoder

In this section, we introduce how we encode the relation instance using the pre-trained models. Given a relation instance with n tokens as $w = [w_1, w_2, \ldots, w_n]$, where head entity e_h and tail entity e_t are marked with the start and end position of the entity. In addition, we adopt the pre-trained language model BERT (Devlin et al., 2019) to encode each token w_t to the corresponding representation $h_t \in R^d$ where d denotes the dimension of representation vectors. Then, we obtain the hidden state vectors of two entities h_{ent} by averaging their respective token's hidden state vectors:

$$h_{ent} = mean\text{-}pooling([h_s, \ldots, h_e]) \quad (7)$$

where $h_{ent} \in R^d$. s and e represent start and end position of the corresponding entity respectively. Finally, we use the concatenation of representations of two entity as the representation of the relation instance $h \in R^{2 \cdot d}$:

$$h = [h_{head}, h_{tail}] \quad (8)$$

D Detailed Results of Other Experiments

In this section, detailed results of consistency regularization loss, different pseudo-labels analysis and cross domain analysis are listed in Table 5, Table 6 and Table 7 respectively.

Dataset	Method	B^3			V-measure			ARI	Avg.
		Prec.	Rec.	F1.	Homo.	Comp.	F1.		
FewRel	Baseline	0.752	0.785	0.768	0.840	0.855	0.847	0.738	0.784
	+ CC	0.747	0.822	0.783	0.837	0.867	0.852	0.722	0.786
	+ single CP	0.753	0.832	0.790	0.839	0.872	0.855	0.726	0.790
	+ double CP	0.794	0.802	0.798	0.861	0.869	0.865	0.773	0.812
	+ double CP + CC	0.815	0.822	**0.819**	0.875	0.873	**0.889**	**0.794**	**0.834**
TACRED	Baseline	0.856	0.795	0.824	0.883	0.843	0.862	0.743	0.810
	+ CC	0.845	0.823	0.835	0.879	0.855	0.867	0.789	0.830
	+ single CP	0.842	0.856	0.849	0.870	0.874	0.872	0.801	0.841
	+ double CP	0.844	0.871	0.857	0.870	0.884	0.877	0.812	0.849
	+ double CP + CC	0.858	0.888	**0.873**	0.891	0.906	**0.898**	**0.829**	**0.867**

Table 5: The detailed results of the proposed consistency regularization loss.

Dataset	Method	B^3			V-measure			ARI	Avg.
		Prec.	Rec.	F1.	Homo.	Comp.	F1.		
FewRel	RoCORE	0.752	0.846	0.796	0.838	0.883	0.860	0.709	0.788
	CaPL-hard	0.753	0.811	0.781	0.843	0.873	0.858	0.705	0.781
	CaPL-soft	0.815	0.822	**0.819**	0.875	0.873	**0.889**	**0.794**	**0.834**
TACRED	RoCORE	0.871	0.849	0.860	0.895	0.881	0.888	0.812	0.853
	CaPL-hard	0.832	0.850	0.841	0.867	0.878	0.872	0.769	0.827
	CaPL-soft	0.858	0.888	**0.873**	0.891	0.906	**0.898**	**0.829**	**0.867**

Table 6: The detailed results of different pseudo-labels analysis.

Task	Method	B^3			V-measure			ARI	Avg.
		Prec.	Rec.	F1.	Homo.	Comp.	F1.		
F $\rightarrow$ T	RSN	0.349	0.590	0.439	0.387	0.533	0.448	0.279	0.389
	RSN-BERT	0.337	0.866	0.486	0.400	0.777	0.528	0.352	0.455
	RoCORE	0.621	0.602	0.611	0.642	0.666	0.654	0.451	0.572
	CaPL	0.813	0.601	**0.691**	0.847	0.703	**0.769**	**0.650**	**0.703**
T $\rightarrow$ F	RSN	0.225	0.529	0.316	0.359	0.507	0.420	0.243	0.326
	RSN-BERT	0.261	0.861	0.400	0.438	0.822	0.571	0.263	0.411
	RoCORE	0.687	0.766	0.724	0.796	0.836	0.815	0.658	0.732
	CaPL	0.722	0.757	**0.739**	0.802	0.830	**0.816**	**0.664**	**0.740**

Table 7: The detailed results of cross domain analysis.

Few-shot Named Entity Recognition with Entity-level Prototypical Network Enhanced by Dispersedly Distributed Prototypes

Bin Ji[1*], **Shasha Li**[1*], **Shaoduo Gan**[2], **Jie Yu**[1†], **Jun Ma**[1†], **Huijun Liu**[1†], **Jing Yang**[1†]
[1]College of Computer, National University of Defense Technology
[2]Individual
{jibin,shashali,yj,majun,liuhuijun,yangjing3026}@nudt.edu.cn
ganshaoduo@gmail.com

Abstract

Few-shot named entity recognition (NER) enables us to build a NER system for a new domain using very few labeled examples. However, existing prototypical networks for this task suffer from roughly estimated label dependency and closely distributed prototypes, thus often causing misclassifications. To address the above issues, we propose **EP-Net**, an **E**ntity-level **P**rototypical **Net**work enhanced by dispersedly distributed prototypes. EP-Net builds entity-level prototypes and considers text spans to be candidate entities, so it no longer requires the label dependency. In addition, EP-Net trains the prototypes from scratch to distribute them dispersedly and aligns spans to prototypes in the embedding space using a space projection. Experimental results on two evaluation tasks and the Few-NERD settings demonstrate that EP-Net consistently outperforms the previous strong models in terms of overall performance. Extensive analyses further validate the effectiveness of EP-Net.

1 Introduction

As a core language understanding task, named entity recognition (NER) faces rapid domain shifting. When transferring NER systems to new domains, one of the primary challenges is dealing with the mismatch of entity types (Yang and Katiyar, 2020). For example, only 2 types are overlapped between I2B2 (Stubbs and Özlem Uzuner, 2015) and OntoNotes (Ralph et al., 2013), which have 23 and 18 entity types, respectively. Unfortunately, annotating a new domain takes considerable time and efforts, let alone the domain knowledge required (Hou et al., 2020). Few-shot NER is targeted in this scenario since it can transfer prior experience from resource-rich (source) domains to resource-scarce (target) domains.

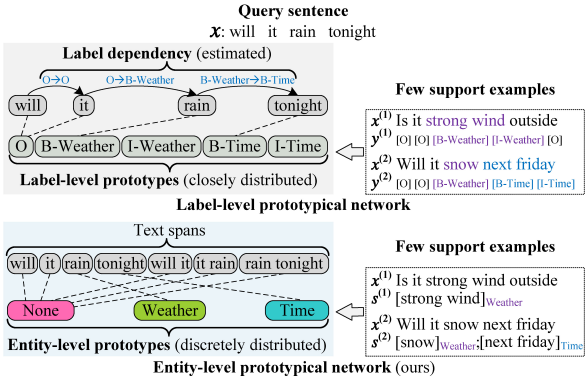

Figure 1: A comparison of token- and entity-level prototypical networks for few-shot NER, where the former builds prototypes for token labels and requires label dependency, while the latter builds prototypes for entity types and does not require label dependency. The dotted line denotes that the pair of token-prototype (or span-prototype) is the most similar. For clarity, we only list spans with lengths less than 2 and assume there are only 2 pre-defined entity types.

Previous few-shot NER models (Fritzler et al., 2019; Hou et al., 2020; Yang and Katiyar, 2020; Tong et al., 2021) generally formulate the task as a sequence labeling task and employ token-level prototypical networks (Snell et al., 2017). These models first obtain token labels according to the most similar token-prototype pair and then obtain entities based on these labels, as Figure 1 shows. The sequence labeling benefits from label dependency (Hou et al., 2020). However, when it comes to few-shot NER models, the label dependency is off the table, because a few labeled data is way insufficient to learn the reliable dependency, and the label sets could vary from domain to domain. To tackle this, some methods try to transfer roughly estimated dependency. Hou et al. (2020) first learn the abstract label transition probabilities in source domains and then copy them to target domains. As Figure 2a shows, the abstract $\texttt{O}\rightarrow\texttt{I}$ probability is copied to three targets directly (the red lines). However, this

* equal contributions
† corresponding authors

Proceedings of the 29th International Conference on Computational Linguistics, pages 1842–1854
October 12–17, 2022.

makes the target probability sum of O→(all labels) end up with 160%. To avoid the possible probability overflows, Yang and Katiyar (2020) propose an even distribution method. As Figure 2b shows, the abstract O→I probability is distributed evenly among the three targets (the green lines). However, this could lead to severe contradictions between the target probabilities and reality. For example, there are 4,983 DATE entities and only one EMAIL entity in the I2B2 test set, so the target probabilities of O→I-DATE and O→I-EMAIL should be clearly different. Consequently, the current dependency transferring may lead to misclassifications due to the roughly estimated target transition probabilities, even though it sheds some light on few-shot NER.

In addition, the majority of prototypical models for few-shot NER (Huang et al., 2021; Li et al., 2020) obtain prototypes by averaging the embeddings of each class's support examples, while Yoon et al. (2019) and Hou et al. (2020) demonstrate that such prototypes distribute closely in the embedding space, thus often causing misclassifications.

In this paper, we aim to tackle the above issues inherent in token-level prototypical models. To this end, we propose **EP-Net**, an **E**ntity-level **P**rototypical **Net**work enhanced by dispersedly distributed prototypes, as Figure 1 shows. EP-Net builds entity-level prototypes and considers text spans as candidate entities. Thus it can determine whether a span is an entity directly according to the most similar prototype to the span. This also eliminates the need for the label dependency. For example, EP-Net determines the *"rain"* and *"tonight"* are two entities, and their types are the Weather and Time, respectively (Figure 1).[1] In addition, to distribute these prototypes dispersedly, EP-Net trains them using a distance-based loss from scratch. And EP-Net aligns spans and prototypes in the same embedding space by utilizing a deep neural network to map span representations to the embedding spaces of prototypes.

In essence, EP-Net is a span-based model. Several span-based models (Li et al., 2021; Fu et al., 2021; Yu et al., 2022) have been proposed for the supervised NER task. Our EP-Net differs from these models in two ways: (1) The EP-Net obtains entities based on the span-prototype similarity, while these models do so by classifying span representations. (2) The EP-Net works effectively with few labeled examples, whereas these models need a large number of labeled examples to guarantee good performance.

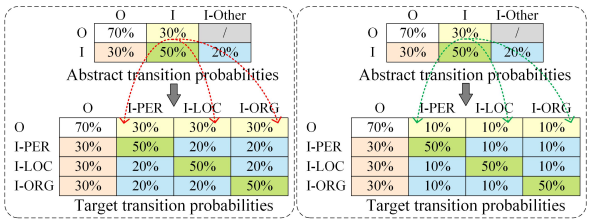

(a) Copying method (b) Even distribution method

Figure 2: Methods of transferring estimated label dependency. In each method, the abstracts and corresponding targets are displayed in the same color. Method (a) copies the abstracts to the targets, whereas method (b) distributes them equally to the targets.

We evaluate our EP-Net on the tag set extension and domain transfer tasks, as well as the Few-NERD settings. Experimental results demonstrate that EP-Net consistently achieves new state-of-the-art overall performance. Qualitative analyses (§5.5-5.6) and ablation studies (§5.7) further validate the effectiveness of EP-Net.

In summary, we conclude the contributions as follows: (1) As far as we know, we are among the first to propose an entity-level prototypical network for few-shot NER. (2) We propose a prototype training strategy to augment the prototypical network with dispersedly distributed prototypes. (3) Our model achieves the current best overall performance on two evaluation tasks and the Few-NERD.

2 Related Work

Meta Learning. Meta learning aims to learn a general model that enables us to adapt to new tasks rapidly based on a few labeled examples (Li et al., 2020). One of the most typical metric learning methods is the prototypical network (Snell et al., 2017), which learns a prototype for each class and classifies an item based on item-prototype similarities. Metric learning has been widely investigated for NLP tasks, such as text classification (Sun et al., 2019; Geng et al., 2019; Bao et al., 2020), relation classification (Lv et al., 2019; Gao et al., 2020) and NER (Huang et al., 2021). However, these methods use the prototypes obtained by averaging the embeddings of support examples for each class, which are closely distributed. In contrast, our model uses dispersedly distributed prototypes obtained by supervised prototype training.

[1] We also add a None type and assign it to spans that are not entities.

Few-shot NER. Previous few-shot NER models (Li et al., 2020; Tong et al., 2021) generally formulate the task as a sequence labeling task and propose to use the token-level prototypical network. Thus these models call for label dependency to guarantee good performance. However, it is hard to obtain exact dependency since the label sets vary greatly across domains. As an alternative, Hou et al. (2020) propose to transfer estimated dependency. They copy the learned abstract dependency from source to target domains, but the target dependency contradicts the probability definition. Yang and Katiyar (2020) propose *StructShot*, which improves the above dependency transferring by equally distributing the abstract dependency to target domains, whereas the target dependency contradicts the reality. Das et al. (2022) introduce Contrastive Learning to the *StructShot*, which inherits the estimated dependency transferring. We demonstrate that the roughly estimated dependency may harm model performance. In addition, prompt-based models (Cui et al., 2021; Sun et al., 2021; Gu et al., 2022; Cui et al., 2022; Ding et al., 2022) have been widely researched for this task recently, but the model performance heavily relies on the chosen prompts. Current with our work, Wang et al. (2022) also propose a span-level prototypical network to bypass label dependency, but their work is still hampered by closely distributed prototypes. In contrast, our model constructs dispersedly distributed entity-level prototypes, thus avoiding the roughly estimated label dependency and closely distributed prototypes.

3 Task Formulation and Setup

In this section, we formally define the task and then introduce the standard evaluation setup.

3.1 Few-shot NER

We define an unstructured sentence as a token sequence $\mathcal{X} = (x_1, x_2, \ldots, x_n)$, and define entities annotated in $\mathcal{X}$ as $\mathcal{E} = [(e^{(1)}, t^{(1)}), ..., (e^{(k)}, t^{(k)})]$, where $e^{(i)}$ and $t^{(i)}$ denote entity text and entity type, respectively. A domain $\mathcal{D} = \{(\mathcal{X}^{(i)}, \mathcal{E}^{(i)})\}_{i=1}^{N_\mathcal{D}}$ is a set of $(\mathcal{X}, \mathcal{E})$ pairs, and each $\mathcal{D}$ has a domain-specific entity type set $\mathcal{T}_\mathcal{D} = \{t_i\}_{i=1}^{N}$, and N is various across domains.

We achieve the few-shot task through three steps: **Train**, **Adapt** and **Recognize**. We first **train** EP-Net with the data of source domains $\{\mathcal{D}_1, \mathcal{D}_2, ...\}$. Then we then **adapt** the trained EP-Net to target domains $\{\mathcal{D}'_1, \mathcal{D}'_2, \ldots\}$ by fine-tuning it on support sets sampled from target domains. Finally, we **recognize** entities of query sets using the domain-adapted EP-Net. We formulate a support set as $\mathcal{S}=\{(\mathcal{X}^{(i)}, \mathcal{E}^{(i)})\}_{i=1}^{N_\mathcal{S}}$, where $\mathcal{S}$ usually includes a few labeled examples (K-shot) of each entity type. For a target domain, we formally define the K-shot NER as follows: given a sentence $\mathcal{X}$ and a K-shot support set, find the best entity set $\mathcal{E}$ for $\mathcal{X}$.

3.2 The Standard Evaluation Setup

To facilitate meaningful comparisons of results for future research, Yang and Katiyar (2020) propose a standard evaluation setup. The setup consists of the query set and support set constructions.

3.2.1 Query Set Construction

They argue that traditional construction methods sample different entity classes equally without considering entity distributions. For example, the I2B2 test set contains 4,983 `DATE` entities, while it only contains one `EMAIL` entity. Thus they propose to use the original test sets of standard NER datasets as the query sets, thus improving the reproducibility of future studies.

3.2.2 Support Set Construction

To construct support sets, they propose a Greedy Sampling algorithm to sample sentences from the dev sets of standard NER datasets. In particular, the algorithm samples sentences for entity classes in increasing order of their frequencies. We present the algorithm in Appendix A.

4 Model

In this section, we first provide an overview of EP-Net in §4.1, and then illustrate the model initializations in §4.2 and discuss the model in §4.3.

4.1 EP-Net

Figure 3 shows the overall architecture of EP-Net. Given a domain $\mathcal{D}=\{(\mathcal{X}^{(i)}, \mathcal{E}^{(i)})\}_{i=1}^{N_\mathcal{D}}$ and its entity type set $\mathcal{T}_\mathcal{D} = \{t_i\}_{i=1}^{N}$, we first initialize an entity-level prototype for each t_i (Figure 3-①).

$$\boldsymbol{\Phi} = \{\boldsymbol{\phi}_0, \boldsymbol{\phi}_1, \boldsymbol{\phi}_2, ..., \boldsymbol{\phi}_N\}, \tag{1}$$

where $\boldsymbol{\Phi} \in \mathbb{R}^{(N+1)*d_1}$, and d_1 is the dimension of prototype representation. $\boldsymbol{\phi}_0$ is the prototype of the `None` type, and $\boldsymbol{\phi}_i$ $(i > 0)$ is the prototype of t_i.

We design a distance-based loss $\mathcal{L}_d$ to supervise the prototype training (Figure 3-②), aiming

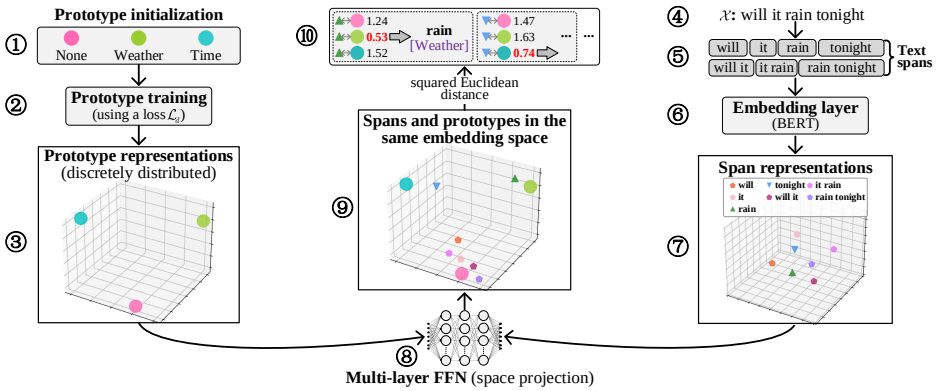

Figure 3: The architecture of EP-Net. As an example, we use the sentence in Figure 1 and 3-dimensional embedding spaces. EP-Net first initializes entity-level prototypes for entity types (①). Then it trains the prototypes with a distance-based loss $\mathcal{L}_d$ (②), distributing them dispersedly in embedding space (③). Next, given a sentence $\mathcal{X}$ (④), EP-Net first obtains text spans (⑤) and then uses BERT (⑥) to generate span representations (⑦). Fourth, EP-Net uses the space projdection (⑧) to align spans and prototypes in the same embedding space (⑨). Finally, EP-Net calculates span-prototype similarities measured by the squared Euclidean distance. The shorter the distance, the better the similarity. EP-Net classifies entities based on the best similarity, e.g., classifying the "*rain*" into the `Weather` type (⑩).

to distribute these prototypes dispersedly in the embedding space (Figure 3-③). We argue that the prototypes should be dispersedly distributed in an appropriate-sized embedding space, neither too large nor too small (§5.5). Thus we first set a threshold τ to limit the averaged prototype distance. Then we calculate the squared Euclidean distance between any two prototypes and obtain the averaged prototype distance (denoted as $Euc(\boldsymbol{\Phi})$). Next, we construct the $\mathcal{L}_d$ as follows.

$$Euc(\boldsymbol{\Phi}) = \frac{\sum_{i=0}^{N}\sum_{j=0}^{N}\sum_{k=1}^{d}(\boldsymbol{\phi}_{i,k}-\boldsymbol{\phi}_{j,k})^2}{(N+1)^2}, \quad (2a)$$

$$\psi = \begin{cases} Euc(\boldsymbol{\Phi}) - \tau & \text{if} \quad Euc(\boldsymbol{\Phi}) \geq \tau, \\ \tau - Euc(\boldsymbol{\Phi}) & \text{if} \quad Euc(\boldsymbol{\Phi}) < \tau, \end{cases} \quad (2b)$$

$$\mathcal{L}_d = \log(\psi + 1). \quad (2c)$$

The training goal is to achieve $\psi \rightarrow 0^+$, which equals to $(\psi + 1) \rightarrow 1^+$. Thus we design the $\log(\cdot)$ loss (Eq.2c), where the smaller the $\mathcal{L}_d$, the more the $(\psi + 1) \rightarrow 1^+$.

Next, given a sentence $\mathcal{X} = \{x_i\}_{i=1}^{n}$ of the domain $\mathcal{D}$, we first obtain text spans (denoted as s, Figure 3-④⑤):

$$s = \{x_i, x_{i+1}, ..., x_{i+j}\} \; s.t. \; 1 \leq i \leq i+j \leq n, \quad (3)$$

where the span length j+1 is limited by a threshold ϵ: j+1 $\leq \epsilon$. To obtain the span representation (Figure 3-⑥⑦), we first use BERT (Devlin et al., 2019) to generate the embedding sequence of $\mathcal{X}$.

$$\boldsymbol{H}_{\mathcal{X}} = \{\boldsymbol{h}_1, \boldsymbol{h}_2, \ldots, \boldsymbol{h}_n\}, \quad (4)$$

where $\boldsymbol{H}_{\mathcal{X}} \in \mathbb{R}^{n*d_2}$, and d_2 is the BERT embedding dimension. $\boldsymbol{h}_i$ is the BERT embedding of x_i. We use $\boldsymbol{H}_s$ to denote the BERT embedding sequence of span s.

$$\boldsymbol{H}_s = \{\boldsymbol{h}_i, \boldsymbol{h}_{i+1}, \ldots, \boldsymbol{h}_{i+j}\}. \quad (5)$$

We obtain the span representation (denoted as $\boldsymbol{E}_s$) by concatenating the max-pooling of $\boldsymbol{H}_s$ (denoted as $\tilde{\boldsymbol{H}}_s$) and the span length embedding.

$$\tilde{\boldsymbol{H}}_s = [\max(\boldsymbol{h}_{i,1}, ..., \boldsymbol{h}_{i+j,1}), ..., \max(\boldsymbol{h}_{i,d_1}, ..., \boldsymbol{h}_{i+j,d_1})], \quad (6a)$$

$$\boldsymbol{E}_s = [\tilde{\boldsymbol{H}}_s; \boldsymbol{w}_{j+1}], \quad (6b)$$

where $\tilde{\boldsymbol{H}}_s \in \mathbb{R}^{d_2}$, $\boldsymbol{E}_s \in \mathbb{R}^{d_2+d_3}$. $\boldsymbol{w}_{j+1}$ is the length embedding trained for spans with a length j+1 and d_3 is the embedding dimension. Due to the fact that $\boldsymbol{E}_s$ and prototype representations ($\boldsymbol{\Phi}$) are not in the same embedding space, we project $\boldsymbol{E}_s$ to the embedding space of $\boldsymbol{\Phi}$ using a multi-layer Feed Forward Network (FFN)[2] and denote the aligned span representation as $\tilde{\boldsymbol{E}}_s$ (Figure 3-⑧⑨).

$$\tilde{\boldsymbol{E}}_s = \boldsymbol{E}_s\boldsymbol{W} + \boldsymbol{b}, \quad (7)$$

where $\tilde{\boldsymbol{E}}_s \in \mathbb{R}^{d_1}$, $\boldsymbol{W}$ and $\boldsymbol{b}$ are FFN parameters. Next, for the span s, we calculate the similarity between it and each prototype $\boldsymbol{\phi}_i \in \boldsymbol{\Phi}$ using the squared Euclidean distance.

$$\text{sim}(s, \boldsymbol{\phi}_i)_{i=0}^{N} = \sum_{j=1}^{d}(\tilde{\boldsymbol{E}}_{s,j} - \boldsymbol{\phi}_{i,j})^2. \quad (8)$$

[2]The FFN enables us to fine-tune our model on support sets without overfitting due to its simple neural architecture.

As a shorter distance denotes a better similarity, we classify entities based the shortest distance (Figure 3-⑩):

$$t_s = \arg\min_{\boldsymbol{\phi}_i \in \boldsymbol{\Phi}} \text{sim}(s, \boldsymbol{\phi_i})_{i=0}^{N}, \tag{9}$$

where $t_s \in \mathcal{T}_{\mathcal{D}}$ is the type classified for the span s.

To construct an entity classification loss, we first take the $-\text{sim}(s, \boldsymbol{\phi}_i)_{i=0}^{N}$ as classification logits, thus the best similarity has the largest logit. We then normalize these logits using the softmax function. Finally, we construct a cross-entropy loss $\mathcal{L}_s$.

$$\hat{\boldsymbol{y}}_{s,i} = \frac{\exp^{-\text{sim}(s,\boldsymbol{\phi}_i)}}{\sum_{j=0}^{N} \exp^{-\text{sim}(s,\boldsymbol{\phi}_j)}}, \tag{10a}$$

$$\mathcal{L}_s = -\frac{1}{M_s} \sum_{j=1}^{M_s} \sum_{i=0}^{N} \boldsymbol{y}_{s,i}^{j} \log \hat{\boldsymbol{y}}_{s,i}^{j}, \tag{10b}$$

where $\{\boldsymbol{y}_s, \hat{\boldsymbol{y}}_s\} \in \mathbb{R}^{N+1}$, and $\boldsymbol{y}_s$ is the one-hot vector of the gold type for the span s. M_s is the number of span instances.

During model training, we optimize model parameters by minimizing the following joint loss.

$$\mathcal{L}(W; \theta) = \mathcal{L}_d + \mathcal{L}_s. \tag{11}$$

4.2 Initializations

4.2.1 Train Initialization

In the **Train** step, given a source domain $\mathcal{D}$ with an entity type set $\mathcal{T}_{\mathcal{D}}=\{t_i\}_{i=1}^{N}$, we randomly initialize the entity-level prototypes $\boldsymbol{\Phi} = \{\boldsymbol{\phi}_i\}_{i=0}^{N}$.We assign $\boldsymbol{\phi}_0$ to the `None` type and $\boldsymbol{\phi}_i$ to t_i. To guarantee that we can adapt EP-Net to target domains that have more types than the domain $\mathcal{D}$, we actually initialize $\boldsymbol{\Phi} = \{\boldsymbol{\phi}_i\}_{i=0}^{100}$, where EP-Net can be adapted to any target domains with entity types less than 100. Moreover, the N can be set to an even larger value if necessary. By doing so, the prototypes $\{\boldsymbol{\phi}_i\}_{i=N+1}^{100}$ are unassigned, but we can still distribute them dispersedly through training them using the loss $\mathcal{L}_d$ (§5.6). We use the bert-base-cased model in the embedding Layer.[3]

4.2.2 Adapt and Recognize Initializations

In the **Adapt** step, given a target domain $\mathcal{D}'$ with an entity type set $\mathcal{T}_{\mathcal{D}'}=\{t'_i\}_{i=1}^{N'}$ and the EP-Net trained in the **Train** step, we first assign a prototype of the trained $\boldsymbol{\Phi} = \{\boldsymbol{\phi}_i\}_{i=0}^{100}$ to each t'_i. In particular, we assign $\boldsymbol{\phi}_0$ to the `None` type. And if there are types that are overlapped between $\mathcal{T}_{\mathcal{D}}$ and $\mathcal{T}_{\mathcal{D}'}$ (i.e., $\mathcal{T}_{\mathcal{D}} \cap \mathcal{T}_{\mathcal{D}'} \neq \varnothing$), for each overlapped type, we reuse the prototype assigned in the **Train** step. For other types in $\mathcal{T}_{\mathcal{D}'}$, we randomly assign an unassigned prototype in $\boldsymbol{\Phi}$ to it, and we first choose the prototype that is ever assigned in the **Train** step. Then, we adapt EP-Net to the domain $\mathcal{D}'$ by fine-tuning it on support sets sampled from $\mathcal{D}'$.

However, Fine-tuning the model with small support sets runs the risk of overfitting. To avoid this, we propose to use the following strategies: (1) We freeze the BERT and solely fine-tune the assigned prototypes and the multi-layer FFN. (2) We use an early stopping criterion, where we continue fine-tuning our model until the loss starts to increase. (3) We set upper limits for fine-tuning steps, where the model will stop when reaching the limits even though the loss continues decreasing. With the above strategies, we demonstrate that only a few fine-tuning steps on these examples can make rapid progress without overfitting.

In the **Recognize** step, we use the domain-adapted EP-Net to recognize entities in the query set of $\mathcal{D}'$ directly.

4.3 Model Discussion

In the **Train** step, the randomly initialized prototypes cannot represent entity types at first. Through the joint model training with the $\mathcal{L}(W; \theta)$, EP-Net establishes correlations between entity types and their assigned prototypes. Moreover, the multi-layer FFN can also be trained to cluster similar spans around related prototypes in the embedding space. As Figure 3-⑨ shows, the "*rain*" is mapped to be closer to the `Weather` than other prototypes.

To precisely simulate the few-shot scenario, we are not permitted to count the entity length of target domains. Thus we set the span length threshold ϵ to an empirical value of 10 based on source domains. For example, 99.89% of the entities in the OntoNotes have lengths under 10.

We propose a heuristic method for removing overlapped entities classified by EP-Net. Specifically, we keep the one with the best span-prototype similarity of those overlapped entities and drop the others.

Concurrently, Wang et al. (2022) propose a span-level model – ESD. We summarize how our EP-Net differs from the ESD as follows: (1) Our EP-Net fine-tunes on support sets while the ESD

[3] https://huggingface.co/bert-base-cased.

solely uses them for similarity calculation without fine-tuning. Ma et al. (2022) claim that the fine-tuning method is far more effective in using the limited information in support sets. (2) The ESD obtains class prototypes with embeddings of the same classes in support sets, thus suffering from closely distributed prototypes (Hou et al., 2020). By contrast, our EP-Net avoids this by training dispersedly distributed prototypes from scratch.

5 Experiments

5.1 Evaluation Tasks

We evaluate EP-Net on two evaluation tasks and the Few-NERD settings using 1- and 5-shot settings. Limited by space, we solely report the key points here and discuss more details in Appendix B.

Tag Set Extension. This task aims to evaluate models for recognizing new types of entities in existing domains. Yang and Katiyar (2020) divide the 18 entity types of the OntoNotes (Ralph et al., 2013) into three target sets, i.e., Group A, B and C, to simulate this scenario. Models are evaluated on one target set while being trained on the others.

Domain Transfer. This task aims to evaluate models for adapting to a different domain. Yang and Katiyar (2020) propose to use the general domain as the source domain and test model on medical, news, and social domains.

Few-NERD Settings. Few-NERD (Ning et al., 2021) is a large-scale dataset for few-shot NER. It consists of two different settings: **Intra** and **Inter**. The Intra divides the train/dev/test according to coarse-grained types. The Inter divides the train/dev/test according to fine-grained types. Thus the coarse-grained entity types are shared. The Intra is more challenging as the restrictions of sharing coarse-grained types.

5.2 Datasets and Baselines

For a fair comparison, we use the same datasets and baselines reported in (Yang and Katiyar, 2020; Ning et al., 2021; Das et al., 2022). Specifically, we use OntoNotes (general domain), CoNLL 2003 (Tjong Kim Sang and De Meulder, 2003) (news domain), I2B2 2014 (Stubbs and Özlem Uzuner, 2015) (medical domain) and WNUT 2017 (Derczynski et al., 2017) (social domain) for the tag set extension and domain transfer tasks.

We compare the performance of EP-Net with previous best models, including: Prototypical Network (**ProtoNet**) (Snell et al., 2017), **ProtoNet+P&D** (Hou et al., 2020) , **NNShot** and **StructShot** (Yang and Katiyar, 2020) and **CONTaiNER** (Das et al., 2022). We represent more baseline details in Appendix C.

5.3 Implementation Details

In all experiments, we optimize EP-Net using AdamW with a learning rate of $5e$-5 and set d_1 and d_3 to 512 and 25, respectively. d_2 is 768 when using the BERT base model. We set 3 layers for the multi-layer FFN, and the train batch size to 2 and 8 in 1- and 5-shot experiments, respectively. We set the distance threshold τ to 2 and 3 for 1- and 5-shot experiments, respectively. Moreover, we investigate the model performance against different τ values in Appendix D. Following supervised span-based work (Ji et al., 2020), we sample spans of the `None` type during model training and set the sampled count to 20 and 40 in 1- and 5-shot experiments, respectively. Following (Yang and Katiyar, 2020; Das et al., 2022), we sample 5 support sets and report the mean and standard deviation of the F1 scores in each experiment.

5.4 Main Results

We report experimental results for 1- and 5-shot settings in Table 1 and Table 2, respectively. We have the following observations.

(1) In terms of the overall metric (i.e., **Avg.**), EP-Net consistently outperforms the listed baselines on the two tasks and Few-NERD, delivering +1.5% to +7.0% averaged F1 gains. Moreover, EP-Net improves up to +11.4% F1 scores on 1-shot Group B. We attribute these gains to the advantages of the proposed entity-level prototypical network.

(2) On the 5-shot Group C, EP-Net is inferior to CONTaiNER by 8.5% F1 scores. Detailed error analysis indicates that the group's `DATE` type should bear the primary responsibility. Of the 4,178 entities in the test set, 1,536 are `DATE` entities, in which there are up to 429 different expressions, such as "week", "this week", "last week", "2 weeks", "2 - week" etc. However, the 5-shot setting solely enables us to sample very few various expressions, leading to the poor performance in `DATE` entities. For example, if a support set solely samples the "week", it is hard for EP-Net to recognize entities like "this week" and "last week".

In addition, we conduct episode evaluations on Few-NERD and report the results in Appendix E.

Model	Tag Set Extension				Domain Transfer				Few-NERD		
	Group A	Group B	Group C	Avg.	I2B2	CoNLL	WNUT	Avg.	Intra	Inter	Avg.
ProtoNet	18.7±4.7	24.4±8.9	18.3±6.9	20.5	7.6±3.5	53.0±7.2	14.8±4.9	25.1	18.6±7.2	25.3±8.8	22.0
ProtoNet+P&D	18.5±4.4	24.8±9.3	20.7±8.4	21.3	7.9±3.2	56.0±7.3	18.8±5.3	27.6	19.4±5.6	26.2±4.2	22.8
NNShot	27.2±3.5	32.5±14.4	23.8±10.2	25.7	16.6±2.1	61.3±11.5	21.7±6.3	33.2	20.1±8.5	25.7±7.7	22.9
StructShot	27.5±4.1	32.4±14.7	23.8±10.2	27.9	22.1±3.0	62.3±11.4	25.3±5.3	36.6	20.3±4.3	26.7±5.6	23.5
CONTaiNER	32.4±5.1	30.9±11.6	33.0±12.8	32.1	21.5±1.7	61.2±10.7	27.5±1.9	36.7	22.4±5.4	28.4±4.3	25.4
EP-Net (ours)	**38.4**±4.5	**42.3**±10.8	**36.7**±9.5	**39.1**	**27.5**±4.6	**64.8**±10.4	**32.3**±4.8	**41.5**	**25.8**±5.1	**30.9**±4.9	**28.4**

Table 1: F1 scores of 1-shot experiments. We report the mean and standard deviations of F1 scores.

Model	Tag Set Extension				Domain Transfer				Few-NERD		
	Group A	Group B	Group C	Avg.	I2B2	CoNLL	WNUT	Avg.	Intra	Inter	Avg.
ProtoNet	27.1±2.4	38.0±5.9	38.4±3.3	34.5	10.3±0.4	65.9±1.6	19.8±5.0	32.0	33.2±6.4	31.7±5.9	32.5
ProtoNet+P&D	29.8±2.8	41.0±6.5	38.5±3.3	36.4	10.1±0.9	67.1±1.6	23.8±3.9	33.6	26.4±3.8	28.7±7.2	27.6
NNShot	44.7±2.3	53.9±7.8	53.0±2.3	50.5	23.7±1.3	74.3±2.4	23.9±5.0	40.7	29.6±5.3	33.9±5.1	31.8
StructShot	47.4±3.2	57.1±8.6	54.2±2.5	52.9	31.8±1.8	75.2±2.3	27.2±6.7	44.7	31.2±4.4	35.7±3.8	33.5
CONTaiNER	51.2±6.0	56.0±6.2	**61.2**±2.7	56.2	36.7±2.1	75.8±2.7	32.5±3.8	48.3	33.1±4.6	38.4±4.4	35.8
EP-Net (ours)	**55.5**±3.2	**64.8**±4.8	52.7±2.2	**57.7**	**44.9**±2.7	**78.8**±2.7	**38.4**±5.2	**54.0**	**36.4**±4.6	**41.4**±3.6	**38.9**

Table 2: F1 scores of 5-shot experiments. We report the mean and standard deviations of F1 scores.

5.5 Visualization

We use the 1-shot Group A experiment to investigate prototype distributions. Specifically, in the **Train** step we initialize the prototype set $\boldsymbol{\Phi} = \{\boldsymbol{\phi}\}_{i=0}^{100}$ and assign $\{\boldsymbol{\phi}\}_{i=0}^{12}$ to the `None` type and the 12 pre-defined entity types of the source domain. In the **Adapt** step, we assign $\{\boldsymbol{\phi}\}_{i=0}^{6}$ to the `None` type and the 6 pre-defined entity types of the Group A.

We report the visualization results in Figure 4. From Figure 4b, we observe that all prototypes are dispersedly distributed because the Euclidean distance between any two prototypes is approximate 2. Therefore, we conclude that EP-Net can distribute the prototypes dispersedly through the prototype training. From Figure 4a, we see that the distances between the `None` type and other assigned prototypes are generally larger than other distances. We attribute it to the fact that the `None` type does not represent any unified semantic meaning, thus the `None` spans actually correspond to a variety of semantic spaces, requiring the `None` prototype to keep away from other prototypes to alleviate the misclassification problem.

Moreover, we realize another entity-level prototypical network with conventional prototypes[4], and refer to it as **CP-Net**. We do not train the conventional prototypes with the loss $\mathcal{L}_d$ but fine-tune it during the model training. We report more details of CP-Net in Appendix F.

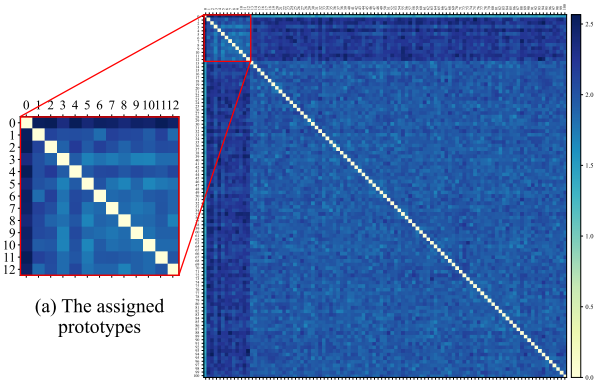

(a) The assigned prototypes

(b) All the prototypes

Figure 4: Heat maps of prototype distributions in the embedding space, which are measured by the squared Euclidean distance. In the (b), we show the distributions of all the prototypes $\boldsymbol{\Phi} = \{\boldsymbol{\phi}\}_{i=0}^{100}$. In the (a), we amplify the distributions of the 13 assigned prototypes $\{\boldsymbol{\phi}\}_{i=0}^{12}$. The darker the color, the larger the distance.

We visualize the distributions of our prototypes and conventional prototypes in Figure 5. To be specific, we use prototypes obtained in the **Recognized** step of both models. We observe that: (1) Our prototypes are distributed much more dispersedly than the conventional prototypes. (2) Our `None` prototype is more distant from other prototypes, whereas the conventional `None` prototype

[4]We obtain the conventional prototypes by averaging the embeddings of each type's examples. For the `None` type, we obtain its prototype by averaging representations of the sampled `None` spans.

stays close to other conventional prototypes. These results indicate that our prototypes enable us to alleviate the misclassifications caused by closely distributed prototypes.

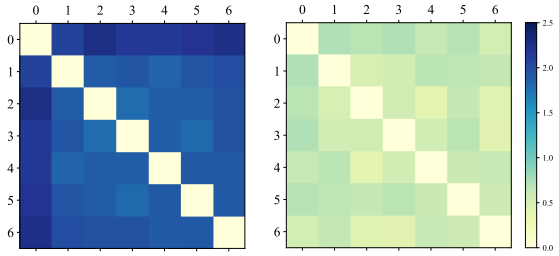

(a) Our prototypes (b) Conventional prototypes

Figure 5: Heat maps of prototype distributions of our prototypes and conventional prototypes.

5.6 How does the Dispersedly Distributed Prototyes Enhance the EP-Net?

We run the 1-shot Group A experiment with EP-Net and CP-Net to conduct the investigation. We first compare the F1 scores of the two models. The results show our EP-Net outperforms CP-Net by +9.6% F1 scores, verifying the effectiveness of the dispersedly distributed prototypes.

In addition, we use t-SNE (Van der Maaten and Hinton, 2008) to reduce the dimension of span representations obtained in the **Recognize** step of EP-Net and CP-Net and visualize these representations in Figure 6. We can see that our EP-Net clusters span representations of the same entity class while dispersing span representations of different entity classes obviously, which we attribute to the usage of dispersedly distributed prototypes. Based on the above fact, we conclude that our EP-Net can greatly alleviate the misclassifications caused by closely distributed prototypes.

5.7 Ablation Study

We conduct ablation studies to investigate the significance of model components and report the results in Table 3. Specifically, (1) In the "- Entity-level prototype", we ablate the entity-level prototypes and use token-level prototypes instead. Moreover, we use the copying method (Figure 2) to transfer the label dependency. The ablation results show that the F1 scores drop from 5.1% to 7.2%, validating the advantages of entity-level prototypes. (2) In the "- Prototype training", we remove the loss $\mathcal{L}_d$ from the $\mathcal{L}(W;\theta)$, thus the prototypes are not trained being dispersedly distributed. The decreasing F1 scores (5.8% to 11.8%) demonstrate that EP-Net significantly benefits from the dispersedly distributed prototypes. (3) In the "-Euclidean distance", we use the cosine similarity to measure span-prototype similarities instead. We see that the Euclidean similarity consistently surpasses the cosine similarity, revealing that a proper measure is vital to guarantee good performance, which is consistent with the conclusion in (Snell et al., 2017).

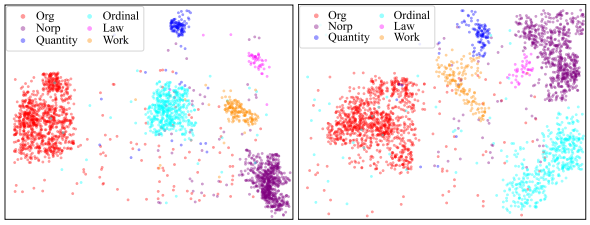

(a) EP-Net (b) CP-Net

Figure 6: t-SNE visualization of span representations of EP-Net and CP-Net. We obtain these representations in the **Recognize** step of both models. Since there are too many `None` spans (890,000+), we do not show their visualizations in the figure.

Model	Group A (F1)	I2B2 (F1)	Intra (F1)	Inter (F1)
EP-Net	38.4	27.5	25.8	30.9
- Entity-level prototype	31.6	22.4	18.6	25.1
- Prototype training	30.3	19.8	20.0	19.1
- Euclidean distance	33.4	25.2	21.6	27.3

Table 3: Ablation results under the 1-shot setting. We select one dataset for each of the two evaluation tasks, as well as the Intra and Inter of the Few-NERD.

6 Conclusion

In this paper, we propose an entity-level prototypical network for few-shot NER (**EP-Net**). And we augment EP-Net with dispersedly distributed prototypes. The entity-level prototypes enable EP-Net to avoid suffering from the roughly estimated label dependency brought by abstract dependency transferring. Moreover, EP-Net distributes the prototypes dispersedly via supervised prototype training and maps spans to the embedding space of the prototypes to eliminate the alignment biases. Experimental results on two evaluation tasks and the Few-NERD settings demonstrate that EP-Net beats the previously published models, creating new state-of-the-art overall performance. Extensive analyses further validate the model's effectiveness.

Acknowledgements

This research is supported by Hunan Provincial Natural Science Foundation (Grant Nos. 2022JJ30668 and 2022JJ30046)

References

Yujia Bao, Menghua Wu, Shiyu Chang, and Regina Barzilay. 2020. Few-shot text classification with distributional signatures. In *Proc. of ICLR*.

Ganqu Cui, Shengding Hu, Ning Ding, Longtao Huang, and Zhiyuan Liu. 2022. Prototypical verbalizer for prompt-based few-shot tuning. In *Proc. of ACL*.

Leyang Cui, Yu Wu, Jian Liu, Sen Yang, and Yue Zhang. 2021. Template-based named entity recognition using BART. In *Proc. of ACL*.

Sarkar Snigdha Sarathi Das, Arzoo Katiyar, and Rui Zhang. 2022. Container: Few-shot named entity recognition via contrastive learning.

Leon Derczynski, Eric Nichols, Marieke van Erp, and Nut Limsopatham. 2017. Results of the WNUT2017 shared task on novel and emerging entity recognition. In *Proceedings of the 3rd Workshop on Noisy User-generated Text*.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proc. of ACL*.

Ning Ding, Shengding Hu, Weilin Zhao, Yulin Chen, Zhiyuan Liu, Hai-Tao Zheng, and Maosong Sun. 2022. Openprompt: An open-source framework for prompt-learning. In *Proc. of ACL*.

Alexander Fritzler, Varvara Logacheva, and Maksim Kretov. 2019. Few-shot classification in named entity recognition task. In *Proc. of SAC*.

Jinlan Fu, Xuanjing Huang, and Pengfei Liu. 2021. SpanNER: Named entity re-/recognition as span prediction. In *Proc. of ACL*.

Tianyu Gao, Xu Han, Ruobing Xie, zhiyuan Liu, Fen Lin, Leyu Lin, , and Maosong Sun. 2020. Neural snowball for few-shot relation learning. In *Proc. of AAAI*.

Ruiying Geng, Binhua Li, Yongbin Li, Xiaodan Zhu, Ping Jian, and Jian Sun. 2019. Induction networks for few-shot text classification. In *Proc. of EMNLP*.

Yuxian Gu, Xu Han, Zhiyuan Liu, and Minlie Huang. 2022. Ppt: Pre-trained prompt tuning for few-shot learning. In *Proc. of ACL*.

Yutai Hou, Wanxiang Che, Yongkui Lai, Zhihan Zhou, Han Liu, and Ting Liu. 2020. Few-shot slot tagging with collapsed dependency transfer and label-enhanced task-adaptive projection network. In *Proc. of ACL*.

Jiaxin Huang, Chunyuan Li, Krishan Subudhi, Damien Jose, Weizhu Chen, Baolin Peng, Jianfeng Gao, and Jiawei Han. 2021. Few-shot named entity recognition: A comprehensive study. In *Proc. of EMNLP*.

Bin Ji, Jie Yu, Shasha Li, Jun Ma, Qingbo Wu, Yusong Tan, and Huijun Liu. 2020. Span-based joint entity and relation extraction with attention-based span-specific and contextual semantic representations. In *Proc. of COLING*.

Fei Li, ZhiChao Lin, Meishan Zhang, and Donghong Ji. 2021. A span-based model for joint overlapped and discontinuous named entity recognition. In *Proc. of ACL*.

Jing Li, Billy Chiu, Shanshan Feng, and Hao Wang. 2020. Few-shot named entity recognition via meta-learning. *IEEE Transactions on Knowledge and Data Engineering*.

Xin Lv, Yuxian Gu, Xu Han, Lei Hou, Juanzi Li, and Zhiyuan Liu. 2019. Adapting meta knowledge graph information for multi-hop reasoning over few-shot relations. In *Proc. of EMNLP*.

Tingting Ma, Huiqiang Jiang, Qianhui Wu, Tiejun Zhao, and Chin-Yew Lin. 2022. Decomposed meta-learning for few-shot named entity recognition. In *Proc. of ACL*.

Ding Ning, Guangwei Xu, Yulin Chen, Xiaobin Wang, Xu Han, Pengjun Xie, Haitao Zheng, and Zhiyuan Liu. 2021. Few-NERD: A few-shot named entity recognition dataset. In *Proc. of ACL*.

Weischedel Ralph, Martha Palmer, Mitchell Marcus, Eduard Hovy, Sameer Pradhan, Lance Ramshaw, and Nianwen Xue. 2013. Ontonotes release 5.0 ldc2013t19. In *Linguistic Data Consortium*.

Jake Snell, Kevin Swersky, and Richard Zemel. 2017. Prototypical networks for few-shot learning. In *Proc. of ICONIP*.

Amber Stubbs and Özlem Uzuner. 2015. Annotating longitudinal clinical narratives for de-identification: The 2014 i2b2/uthealth corpus. *Journal of Biomedical Informatics*.

Shengli Sun, Qingfeng Sun, Kevin Zhou, and Tengchao Lv. 2019. Hierarchical attention prototypical networks for few-shot text classification. In *Proc. of EMNLP*.

Yi Sun, Yu Zheng, Chao Hao, and Hangping Qiu. 2021. NSP-BERT: A prompt-based zero-shot learner through an original pre-training task-next sentence prediction. *CoRR*.

Erik F. Tjong Kim Sang and Fien De Meulder. 2003. Introduction to the CoNLL-2003 shared task: Language-independent named entity recognition. In *Proc. of HLT-NAACL*.

Meihan Tong, Shuai Wang, Bin Xu, Yixin Cao, Minghui Liu, Lei Hou, and Juanzi Li. 2021. Learning from miscellaneous other-class words for few-shot named entity recognition. In *ACL*.

Laurens Van der Maaten and Geoffrey Hinton. 2008. Visualizing data using t-sne. *Journal of machine learning research*, 9(11).

Peiyi Wang, Runxin Xu, Tianyu Liu, Qingyu Zhou, Yunbo Cao, Baobao Chang, and Zhifang Sui. 2022. An enhanced span-based decomposition method for few-shot sequence labeling. In *Proc. of NAACL*.

Yi Yang and Arzoo Katiyar. 2020. Simple and effective few-shot named entity recognition with structured nearest neighbor learning. In *Proc. of EMNLP*.

Sung Whan Yoon, Jun Seo, and Jaekyun Moon. 2019. TapNet: Neural network augmented with task-adaptive projection for few-shot learning. In *Proc. of ICML*.

Jie Yu, Bin Ji, Shasha Li, Jun Ma, Huijun Liu, and Hao Xu. 2022. S-ner: A concise and efficient span-based model for named entity recognition. *Sensors*, 22(8):2852.

Appendix

A The Greedy Sampling Algorithm

Algorithm 1: Greedy Sampling Algorithm

Require: shot K, dev set $\mathbf{X}$ of a domain $\mathcal{D}$ and its entity type set $\mathcal{T}$

Sort types in $\mathcal{T}$ based on their frequencies in $\mathbf{X}$
$\mathcal{S} \leftarrow \varnothing$ // Initialize the support set
$\{\text{Count}_i \leftarrow 0\}$
// Initialize the count of each type in $\mathcal{S}$
while $i < |\mathcal{T}|$ **do**
  while $\text{Count}_i < K$ **do**
    Sample $(\mathcal{X}, \mathcal{E}) \in \mathbf{X}$ $s.t.$ $\mathcal{T}_i \in \mathcal{E}.type$[1]
    // Sample a sentence containing entities of $\mathcal{T}_i$ // type, w/o replacement
    $\mathcal{S} \leftarrow \mathcal{S} \cup \{(\mathcal{X}, \mathcal{E})\}$
    update $\{\text{Count}_j\}$ $\forall\, \mathcal{T}_j \in \mathcal{E}.type$
  end while
end while
return $\mathcal{S}$

[1] $\mathcal{E}.type$ denotes the types of entities annotated in $\mathcal{E}$

B Details of the Evaluation Task

B.1 Tag Set Extension

The Group A, B, and C split from the OntoNotes dataset are as follows.

- Group A: {`Org`, `Quantity`, `Ordinal`, `Norp`, `Work`, `Law`}
- Group B: {`Gpe`, `Cardinal`, `Percent`, `Time`, `Event`, `Language`}
- Group C: {`Person`, `Product`, `Money`, `Date`, `Loc`, `Fac`}

In this task, we evaluate our EP-Net on each group while training our model on the other two groups. In each experiment, we modify the training set by replacing all entity types in the target type set with the `None` type. Hence, these target types are no longer observed during training. We use the modified training set for model training in the **Train** step. Similarly, we modify the dev and test sets to only include entity types contained in the target type set. We use the Greedy Sampling Algorithm to sample multiple support sets from the dev set for model adaption.

B.2 Domain Transfer

In this task, we train our EP-Net on the standard training set of the OntoNotes dataset and evaluate our model on the standard test sets of I2B2, CoNLL, and WNUT. In addition, we sample support sets for model adaption from the standard dev sets of the above three datasets.

B.3 Few-NERD Settings

FEW-NERD (Ning et al., 2021) is the first dataset specially constructed for few-shot NER and is one of the largest human-annotated NER datasets. It consists of 8 coarse-grained entity types and 66 fine-grained entity types. The dataset contains two sub-sets, name **Intra** and **Inter**.

- In Intra, all the fine-grained entity types belonging to the coarse-grained `People`, `MISC`, `Art`, `Product` are assigned to the training set, and all the fine-grained entity types belonging to the coarse-grained `Event`, `Building` are assigned to the dev set, and all the fine-grained entity types belonging to the coarse-grained `ORG`, `LOC` are assigned to the test set. In this dataset, the training/dev/test sets share little knowledge, making it a difficult benchmark.
- In Inter, 60% of the 66 fine-grained types are assigned to the training set, 20% to the dev set, and 20% to the test set. The intuition of this dataset is to explore if the coarse information will affect the prediction of new entities.

We use the standard evaluation (§3.2) and the episode evaluation to evaluate the performance of our EP-Net. For the standard evaluation, we conduct experiments on Intra and Inter, respectively. We first use the training set to train our EP-Net and then sample support sets from the test set for the model adaptation and evaluate our model on the remaining test set. For the episode evaluation, we use the exact evaluation setting proposed by (Ning et al., 2021).

C Baseline Details

Following the established line of work (Yang and Katiyar, 2020; Das et al., 2022; Ning et al., 2021), we compare EP-Net with the following competitive models.

- Prototypical Network (**ProtoNet**) (Snell et al., 2017) is a popular few-shot classification algorithm that has been adopted in most previously published token-level few-shot NER models.
- **ProtoNet+P&D** (Hou et al., 2020) uses pairwise embedding and collapsed dependency transfer mechanism in the token-level Prototypical Network, tackling challenges of similarity computation and transferring estimated label dependency across domains.
- **NNShot** (Yang and Katiyar, 2020) is a simple token-level nearest neighbor classification

model. It simply computes a similarity score between a token in the query example and all tokens in the support set.

- **StructShot** (Yang and Katiyar, 2020) combines NNShot and Viterbi decoder and uses estimated label dependency across domains by first learning abstract label dependency and then distributing it evenly to target domains.
- **CONTaiNER** (Das et al., 2022) introduces Contrast Learning to the StructShot. It models Gaussian embedding and optimizes inter token distribution distance, which aims to decrease the distance of token embeddings of similar entities while increasing the distance for dissimilar ones.

For a fair comparison, we use the results of the ProtoNet, ProtoNet+P&D, NNShot, and StructShot reported in (Yang and Katiyar, 2020), and the results of CONTaiNER reported in (Das et al., 2022).

In addition, we run the ProtoNet, ProtoNet+P&D, NNShot, and StructShot on Few-NERD using the standard evaluation setup (§3.2, B.3).

D Performance against Prototype Distance Threshold (τ)

We conduct 1- and 5-shot experiments to explore the performance against different τ values. Since validation sets are unavailable in the few-shot scenario, we randomly sample 20% of the query sets for the explorations. We report the results in Figure 7, where we set the τ value from 1 to 10, respectively. We can observe that: (1) The F1 scores generally first increase and then decrease when the τ value consistently increases. (2) Except for the Group C and Intra, our EP-Net performs the best in the 1-shot experiments when setting the τ to 2. (3) Except for the Group A and Intra, our EP-Net performs the best in the 5-shot experiments when setting the τ to 3.

The above results validate our argument that the prototypes should be distributed in an appropriate-sized embedding space, neither too large nor too small (§4.1). For simplicity, we set the τ to 2 and 3 in all the other 1- and 5-shot experiments, respectively.

E Episode Evaluation on Few-NERD

We evaluate our EP-Net on Few-NERD with the episode evaluation setting and compare our model with previous state-of-the-art models, including

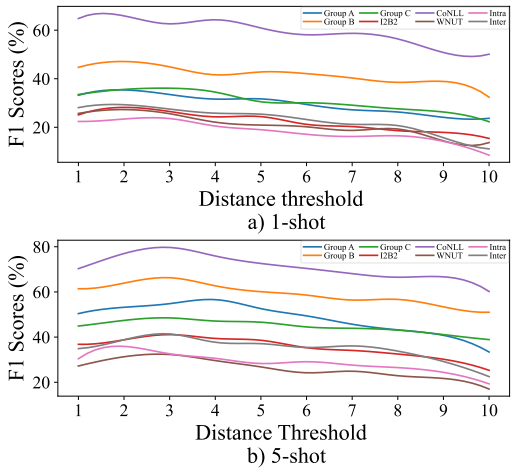

Figure 7: Performance comparisons of different prototype distance threshold (τ) values in 1- and 5-shot experiments.

ProtoBERT (Ning et al., 2021), NNShot, StructShot, CONTaiNER, and ESD (Wang et al., 2022). We would like to mention that the ESD is a concurrent span-based few-shot NER model to ours.

We report the results in Table 4 and Table 5, where we take the results of ProtoBERT, NNShot, and StructShot reported in (Ning et al., 2021), and the results of CONTaiNER and ESD reported in their original papers. We can see that:

- On the Intra, our EP-Net consistently outperforms the best baseline (i.e., CONTaiNER) in terms of the Avg. metric, bringing +2.39% F1 gains. In addition, our EP-Net surpasses the concurrent ESD by +4.43% F1 scores.
- On the Inter, our EP-Net is inferior to ESD by a large margin (5.0%) in terms of the Avg. metric. However, our model consistently outperforms the other baselines, delivering up to +5.31% F1 scores compared to the CONTaiNER.
- Both our EP-Net and CONTaiNER outperform ESD in 1-shot experiments, but they are inferior to ESD in 5-shot experiments.

The above results demonstrate the effectiveness of the proposed EP-Net. And compared to ESD, our model is more efficient in the few-shot scenario when entities share less coarse-grained information (the Intra).[5]

Compared to our simple concatenation method (Eq.6b) to obtain span representations, ESD proposes to use Inter Span Attention (ISA) and Cross

[5] As shown in Appendix B.3, entities in the Intra share little coarse-grained information, but the Inter is designed to allow entities sharing the coarse-grained information.

Model	1∼2-shot		5∼10-shot		Avg.
	5 way	10 way	5 way	10 way	
ProtoBERT	23.45±0.92	19.76±0.59	41.93±0.55	34.61±0.59	29.94
NNShot	31.01±1.21	21.88±0.23	35.74±2.36	27.67±1.06	29.08
StructShot	35.92±0.69	25.38±0.84	38.83±1.72	26.39±2.59	31.63
ESD	41.44±1.16	32.29±1.10	50.68±0.94	42.92±0.75	41.83
CONTaiNER	40.43	33.84	53.70	**47.49**	43.87
EP-Net (Ours)	**43.36±0.99**	**36.41±1.03**	**58.85±1.12**	46.40±0.87	**46.26**

Table 4: Episode evaluation results (F1 scores) on the Intra dataset of Few-NERD. We report the mean and standard deviations of F1 scores.

Model	1∼2-shot		5∼10-shot		Avg.
	5 way	10 way	5 way	10 way	
ProtoBERT	44.44±0.11	39.09±0.87	58.80±1.42	53.97±0.38	49.08
NNShot	54.29±0.40	46.98±1.96	50.56±3.33	50.00±0.36	50.46
StructShot	57.33±0.53	49.46±0.53	57.16±2.09	49.39±1.77	53.34
CONTaiNER	55.95	48.35	61.83	57.12	55.81
ESD	**66.46±0.49**	**59.95±0.69**	**74.14±0.80**	**67.91±1.41**	**66.12**
EP-Net (Ours)	62.49±0.36	54.39±0.78	65.24± 0.64	62.37±1.27	61.12

Table 5: Episode evaluation results (F1 scores) on the Inter dataset of Few-NERD. We report the mean and standard deviations of F1 scores.

Span Attention (CSA) to enhance the span representations. We believe that the ISA and CSA enable ESD to encode the shared coarse-grained information into span representations sufficiently, which helps ESD obtain the current state-of-the-art performance on the Inter dataset.

F CP-Net

We propose the CP-Net as a comparable model to our EP-Net. CP-Net is also an entity-level prototypical network, but it uses conventional prototypes obtained by averaging the embeddings of type's examples. Similar to EP-Net, CP-Net also uses the BERT model as an embedding generator. In addition, it uses the sampling strategy discussed in §5.3 to randomly sample `None` spans. CP-Net consists of two steps, namely **Train** and **Recognize**.

In the Train step, we train CP-Net with the source domain data. To be specific, we obtain the entity-level prototypes by averaging the embeddings of type's examples in the training set. Moreover, we obtain span representations with the same method of EP-Net (Eq.3-7), as well as the method to calculate span-prototype similarity (Eq.8-9). During the model training, we use the training loss $\mathcal{L}_s$ (Eq.10b) to fine-tune the BERT model.

In the Recognize step, we use the fine-tuned BERT model as the embedding generator and obtain the entity-level prototypes by averaging the embeddings of each type's examples in the support sets. Then we obtain the type of each span according to the best similarity between the span and the prototypes.

The CP-Net differs from our EP-Net in the following two ways.

- CP-Net uses conventional prototypes, and it does not train these prototypes during the model training. By contrast, our EP-Net trains prototypes from scratch with the distance based loss $\mathcal{L}_d$ (Eq.2c)
- CP-Net does not contain a domain adaption procedure, and it solely uses the support sets for similarity calculation. By contrast, our EP-Net contains a **Adapt** step for domain adaption and it uses the support sets for not only the similarity calculation but also the domain adaption.

Different Data, Different Modalities! Reinforced Data Splitting for Effective Multimodal Information Extraction from Social Media Posts

Bo Xu[1], **Shizhou Huang**[1], **Ming Du**[1,*], **Hongya Wang**[1], **Hui Song**[1], **Chaofeng Sha**[2] and **Yanghua Xiao**[2]
[1]School of Computer Science and Technology, Donghua University
[2]Shanghai Key Laboratory of Data Science, School of Computer Science, Fudan University
xubo@dhu.edu.cn, 2202408@mail.dhu.edu.cn, {duming, hywang, songhui}@dhu.edu.cn
{cfsha, shawyh}@fudan.edu.cn

Abstract

Recently, multimodal information extraction from social media posts has gained increasing attention in the natural language processing community. Despite their success, current approaches overestimate the significance of images. In this paper, we argue that different social media posts should consider different modalities for multimodal information extraction. Multimodal models cannot always outperform unimodal models. Some posts are more suitable for the multimodal model, while others are more suitable for the unimodal model. Therefore, we propose a general data splitting strategy to divide the social media posts into two sets so that these two sets can achieve better performance under the information extraction models of the corresponding modalities. Specifically, for an information extraction task, we first propose a data discriminator that divides social media posts into a multimodal and a unimodal set. Then we feed these sets into the corresponding models. Finally, we combine the results of these two models to obtain the final extraction results. Due to the lack of explicit knowledge, we use reinforcement learning to train the data discriminator. Experiments on two different multimodal information extraction tasks demonstrate the effectiveness of our method. The source code of this paper can be found in https://github.com/xubodhu/RDS.

1 Introduction

Social media, with its wealth of user-generated posts, provides a rich platform for understanding events, opinions and preferences of groups and individuals (Moon et al., 2018). Information extraction, such as named entity recognition (Yu et al., 2020), relation extraction (Zheng et al., 2021a) and sentiment detection (Yang et al., 2021), is a critical step in uncovering these hidden insights in social media posts.

In social media scenarios, information is expressed not only through textual modality, but through multiple modalities (e.g., text, image, etc.). Considering only text modality may lead to inaccurate information extraction. For example in Fig. 1(a) and Fig. 1(c), the text-based named entity recognition model cannot recognize `Kolo` as a dog, and the text-based relation extraction model cannot determine that `Meghan Markle` and `Prince Harry` are couples.

Therefore, many multimodal models for information extraction have been proposed which are using visual modality to complement text modality. They mainly focus on using a multimodal interaction mechanism to combine text representation with image representation. For example, Zhang et al. (2018) propose an adaptive co-attention network to control the combination of text representation and image representation dynamically for multimodal named entity recognition. Zheng et al. (2021a) propose a multimodal relation extraction neural network with an effective alignment strategy for textual and visual graphs to find the correlations between visual objects and textual entities. Yang et al. (2021) propose multi-channel graph neural networks for multimodal sentiment detection.

Despite their success, current approaches overestimate the significance of images. In fact, images are not always needed to understand information on social media posts. It is possible to get comparable performance using only text modality, while using *mismatched* visual modality can hinder performance. The *mismatched* phenomenon is very common in social media posts. As reported in Vempala and Preoţiuc-Pietro (2019), about 33.8% of tweets had textual content that was not reflected in the images, and the images did not add additional con-

* Corresponding author. This paper was supported by the National Natural Science Foundation of China (No. 61906035), Information Development Project of Shanghai Economic and Information Commission (No. 202002009), Natural Science Foundation of Shanghai (No. 22ZR1402000) and Shanghai Science and Technology Innovation Action Plan (No. 19511120400)

Proceedings of the 29th International Conference on Computational Linguistics, pages 1855–1864
October 12–17, 2022.

(a) [**Kolo MISC**] loves the sun and is so pretty

(b) [**Nasa ORG**] produces vintage travel posters for newly discovered planets

(c) **Meghan Markle** and **Prince Harry** announce their first official royal tour <Meghan Markle, Prince Harry, couple>

(d) Congrats to **Angel** and **Jesenia Rodriguez** on their marriage last night <Angel, Jesenia Rodriguez, couple>

Figure 1: Four Examples of Multimodal Information Extraction Tasks. (a) and (b) are examples of multimodal named entity recognition, (c) and (d) are examples of multimodal relation extraction. The named entity and its type are highlighted in brackets. The entities and relations between entities are in angle brackets.

tent. For example, in Fig. 1(b) and Fig. 1(d), with the *mismatched* images, the multimodal named entity recognition model may mistakenly associate the `person` in the image with `Nasa` and make a wrong prediction, and the multimodal relation extraction model may incorrectly classify the relation between `Angel` and `Jesenia Rodriguez` as `colleague`.

Based on the above observations, we argue that different social media posts should consider different modalities for multimodal information extraction. Multimodal models cannot always outperform unimodal models [1]. Some posts are more suitable for the multimodal model, while others are more suitable for the unimodal model. This is consistent with human behavior in accomplishing multimodal information extraction tasks. When someone reads a post on social media, he first determines whether the image will help him complete the task; if not, he goes directly to the text, and if so, he views both the image and the text. According to a research on the multimodal named entity recognition (*NER*) benchmark dataset `TWITTER-2017` by Yu et al. (2020), about 22% of entities were incorrectly predicted by the state-of-the-art text-based *NER* model, but correctly predicted by the state-of-the-art multimodal *NER* model; and about 12% of entities were correctly predicted by the text-based *NER* model, but incorrectly predicted by the multimodal *NER* model.

In this paper, we propose a general data splitting strategy to divide the social media posts into two sets so that these two sets can achieve better performance under the information extraction models of the corresponding modalities. Specifically, for an information extraction task, we first propose a data discriminator that divides social media posts into a multimodal and a unimodal set, which is used to determine whether the posts are more suitable for the multimodal or unimodal model. Then we feed these sets into the corresponding models. Finally, we combine the results of these two models to obtain the final extraction results. The core component is the data discriminator. Due to the lack of explicit knowledge about which data are more suitable for multimodal models and which data are more suitable for unimodal models, we use reinforcement learning to train the data discriminator. The reward based on the performances of the multimodal model and the unimodal model on both the multimodal set and the unimodal set will be used as a reinforcement signal to update the parameters of the data discriminator.

Our main contributions are summarized as follows:

- First, to the best of our knowledge, we are the first to argue that different social media posts should consider different modalities to accomplish the multimodal information extraction tasks.
- Secondly, we propose a general data splitting strategy for multimodal information extraction and implement this strategy by reinforcement learning.
- Finally, experiments conducted on two widely used multimodal named entity recognition datasets and a multimodal relation extraction dataset show that our method can effectively

[1] In this paper, unimodal model refers to text-based model.

divide the social media posts and achieves the new state-of-the-art performance.

2 OVERVIEW

In this section, we first formulate our problem, and then introduce our framework.

2.1 Problem Formulation

Let $X = \{(T_i, V_i)\}_{i=1}^{N}$ be the set of text-image posts from social media, where T_i is the text modality and V_i is the corresponding visual information, N represents the number of posts. Our aim is to divide X into two disjoint sets: multimodal set X^M and unimodal set X^U, where X^M contains N^M posts and X^U contains N^U posts with $N^M + N^U = N$, so that these two sets can achieve better performance under the information extraction models of the corresponding modalities (i.e., multimodal and unimodal model).

2.2 Framework

The reinforcement learning framework for training the data discriminator is shown in the left side of Fig. 2, which consists of three main components: a data discriminator, a multimodal model and a unimodal model, where the multimodal and unimodal models can be any existing models. The training process is as follows:

- **STEP 1: Training Set Splitting**. Given the training set $\mathcal{D} = \{(T_j, V_j, Y_j)\}_{j=1}^{G}$, where Y_j is the label for the j-th post. We first randomly divide it into two disjoint sets, namely $\mathcal{D}_{model}$ and $\mathcal{D}_{split}$.

- **STEP 2: Multimodal/Unimodal Model Training**. Then we use $\mathcal{D}_{model}$ to train the multimodal and unimodal models and freeze their parameters.

- **STEP 3: Data Discriminator Training**. Finally, we use reinforcement learning to train the data discriminator with $\mathcal{D}_{split}$. In the data discriminator, each data in $\mathcal{D}_{split}$ has a corresponding action a_i to determine whether to use a multimodal or unimodal model for information extraction. The $\mathcal{D}_{split}$ can be divided into the multimodal set $\mathcal{D}_{split}^{M}$ and the unimodal set $\mathcal{D}_{split}^{U}$ based on the data discriminator. After that, we calculate the reward based on the performances of the multimodal and the unimodal models on both $\mathcal{D}_{split}^{M}$ and $\mathcal{D}_{split}^{U}$. The reward will be used as a reinforcement signal to update the parameters of the data discriminator.

3 METHOD

In this section, we first introduce our data discriminator, then we show how to train it using reinforcement learning, which consists of a reward function and a training algorithm.

3.1 Data Discriminator

The data discriminator is used to determine whether a data should use a multimodal or unimodal model, and the main idea is based on the similarity between text and images. As shown in the right side of Fig. 2, the data discriminator consists of a *CLIP* (Radford et al., 2021) and a multilayer perceptron (*MLP*) with one hidden layer. *CLIP* is the state-of-the-art multimodal vision and language model, which consists of a *CLIPTextModel*, a *CLIPVisionModel* and a projection layer.

Specifically, the *CLIPTextModel* layer is used to encode the text. For the input text T, we first tokenize it by using the byte pair encoding (BPE) (Sennrich et al., 2016) and obtain a token sequence $(t_1, t_2, ..., t_n)$, where n is the length of the token sequence. Then the token sequence is bracketed by [SOS] and [EOS] tokens as $([SOS], t_1, t_2, ..., t_n, [EOS])$. The activation at the [EOS] token in the last layer of *CLIPTextModel* is treated as the representation of the entire text $\boldsymbol{T_s} \in \mathbb{R}^{d_t}$.

The *CLIPVisionModel* layer is used to encode the images. For the input image V, we first resize the image to 224×224 pixels, then the image is split into a sequence of $7 \times 7 = 49$ non-overlapping patches with a pixel size of 32×32, which are then linearly embedded to get each patch representation $(v_1, v_2, ..., v_{49})$, and finally add a [CLS] token with the same dimensions as patch at the beginning of V to get $([CLS], v_1, v_2, ..., v_{49})$, where the activation at the [CLS] token in the last layer of *CLIPVisionModel* as the representation of the image $\boldsymbol{V_g} \in \mathbb{R}^{d_v}$.

The projection layer is used to project the representations of text and images into a latent space with identical dimension. The final representations of text $\boldsymbol{T_c} \in \mathbb{R}^{d_c}$ and image $\boldsymbol{V_c} \in \mathbb{R}^{d_c}$ is obtained by projecting $\boldsymbol{T_s}$ and $\boldsymbol{V_g}$ into the same latent space.

Finally, we perform an element-wise product of $\boldsymbol{T_c}$ and $\boldsymbol{V_c}$ and feed it into an *MLP* layer with one hidden layer to obtain the probability p that the data

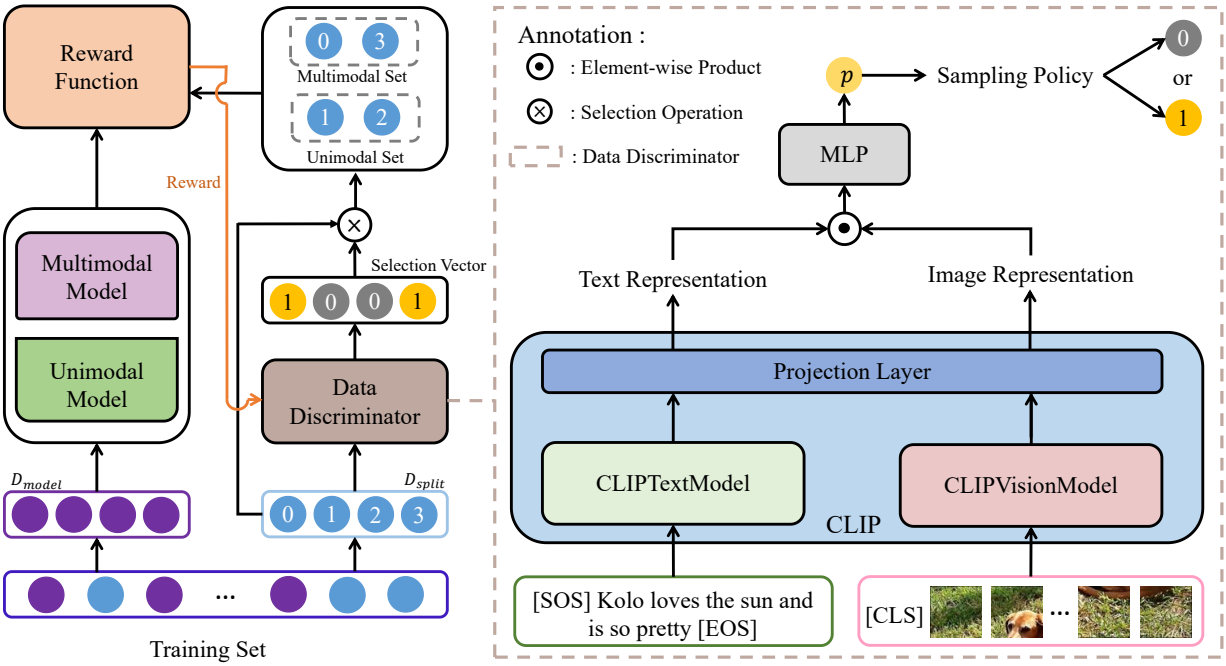

Figure 2: The Reinforcement Learning Framework to Train the Data Discriminator.

is more suitable for the multimodal *IE* model (less suitable for the unimodal *IE* model) as follows:

$$p = sigmoid(\boldsymbol{W_2}\ relu(\boldsymbol{W_1}\ (\boldsymbol{T_c} \odot \boldsymbol{V_c}))) \quad (1)$$

, where $sigmoid$ and $relu$ are the activation functions, $\boldsymbol{W_1}$ and $\boldsymbol{W_2}$ are the weight matrices.

The sampling policy is used to decide the action of the data discriminator based on the probabilities provided by the data discriminator. In this paper, we propose two sampling policies. In the training phase of the data discriminator, in order to encourage exploration based on the uncertainty of the exponentially-large selection space (Yoon et al., 2020), we use *Bernoulli* sampling (Deshmukh1, 1991) to sample the data. Each data will be sampled according to the probability provided by the data discriminator. While in the prediction phase, the data discriminator puts data with probability greater than 0.5 into the multimodal set and data with probability less than or equal to 0.5 into the unimodal set.

3.2 Reward Function

Due to the lack of supervised data, we use reinforcement learning to train the data discriminator. The core component is to design the reward function, which is used to evaluate the quality of the action of data discriminator and used as a reinforcement signal to adjust the parameters of the data discriminator.

Intuitively, the multimodal *IE* model performs better than the unimodal *IE* model on the $\mathcal{D}^M_{split}$, while the unimodal *IE* model performs better than the multimodal *IE* model on the $\mathcal{D}^U_{split}$. In our task here, we use the performance gaps between the multimodal *IE* model and the unimodal *IE* model on both sets as the reward. For example, the micro F1 score is used to evaluate the performance of the multimodal named entity recognition (*MNER*) models and multimodal relation extraction (*MRE*) models on both sets (Yu et al., 2020; Zheng et al., 2021a).

Specifically, we denote the $\mathcal{D}^M_{split} = \{(T_k, V_k, Y_k)\}_{k=1}^{S_1}$ and $\mathcal{D}^U_{split} = \{(T_l, V_l, Y_l)\}_{l=1}^{S_2}$, respectively. The multimodal *IE* model is denoted as f_m and the unimodal *IE* model is denoted as f_u. The performances of models on both sets are defined as follows:

$$v_m^k = Score(\{Y_k, f_m(T_k, V_k)\}_{k=1}^{S_1}) \quad (2)$$

$$v_m^l = Score(\{Y_l, f_m(T_l, V_l)\}_{l=1}^{S_2}) \quad (3)$$

$$v_u^k = Score(\{Y_k, f_u(T_k)\}_{k=1}^{S_1}) \quad (4)$$

$$v_u^l = Score(\{Y_l, f_u(T_l)\}_{l=1}^{S_2}) \quad (5)$$

, where v_m^k and v_m^l are the performances of multimodal *IE* model performed on $\mathcal{D}^M_{split}$ and $\mathcal{D}^U_{split}$, respectively. v_u^k and v_u^l are the performances of unimodal *IE* model performed on $\mathcal{D}^M_{split}$ and $\mathcal{D}^U_{split}$,

respectively. Finally, the reward R is calculated as follows:

$$R = \alpha * (v_m^k - v_u^k) + (1 - \alpha) * (v_u^l - v_m^l) \quad (6)$$

where $\alpha \in (0, 1)$ is the hyperparameter.

3.3 Training Algorithm

Finally, we introduce how to train the data discriminator. Inspired by (Yoon et al., 2020), the training process of the data discriminator is shown in Algorithm 1.

Algorithm 1 The Training Algorithm for the Data Discriminator.

Inputs: Training set $\mathcal{D}$, batch size B_s, hyperparameter of reward function α, learning rate of data discriminator η

Output: The data discriminator g_ϕ.

```
Shuffle D and divide it into D_model and D_split
Train multimodal model and unimodal model using D_model and freeze their parameters
Initialize parameters φ for the data discriminator g_φ
while until convergence do
    Randomly sample a batch of data D^B_split = {(T_i, V_i, Y_i)}_{i=1}^{B_s} from D_split
    for i = 1 to B_s do
        Calculate probability p_i = g_φ(T_i, V_i) according to Equation 1
    end for
    Obtain D^M_split, D^U_split by Bernoulli sampling
    Calculate reward according to Equation 6
    Update φ according to Equation 7
end while
```

We first shuffle the training set $\mathcal{D}$ and divide it into two parts. One is the $\mathcal{D}_{model}$, which is used for training the multimodal *IE* model and the unimodal *IE* model, and the other is $\mathcal{D}_{split}$, which is used for training the data discriminator. Then we train the multimodal *IE* model and the unimodal *IE* model by using the $\mathcal{D}_{model}$ and freeze their parameters.

After that, we initialize the parameters ϕ of the data discriminator g_ϕ. For each iteration, we randomly select a batch of data $\mathcal{D}^B_{split}$ from the $\mathcal{D}_{split}$, and use the data discriminator to predict the probability that each data is more suitable for the multimodal *IE* models p_i. Based on the probabilities, we divide the $\mathcal{D}_{split}$ into the $\mathcal{D}^M_{split}$ and $\mathcal{D}^U_{split}$ by using the *Bernoulli* sampling.

Then, we calculate the reward R according to Equation 6, and update the parameters of our data discriminator as follows:

$$\phi \leftarrow \phi + \eta \cdot R \cdot \bigtriangledown_\phi \log \pi_\phi(\mathcal{D}_2^B, (d_1, ..., d_{B_s})) \quad (7)$$

$$\pi_\phi(\mathcal{D}_2^B, (d_1, ..., d_{B_s})) = \prod_{j=1}^{B_s} (p_j)^{d_j} \cdot (1 - p_j)^{1-d_j} \quad (8)$$

, where η is learning rate, $\pi_\phi(\mathcal{D}_2^B, (d_1, ..., d_{B_s}))$ is the probability that the selection vector $(d_1, ..., d_{B_s})$ is selected based on g_ϕ and $d_j = \{0, 1\}$ is an indicator variable, where 1 represents to put the data into the multimodal set $\mathcal{D}^M_{split}$, while 0 represents to put the data into the unimodal set $\mathcal{D}^U_{split}$.

4 Experiment

To validate the effectiveness of our data splitting strategy, we conducted experiments on two different multimodal information extraction tasks, namely multimodal named entity recognition (*MNER*) and multimodal relation extraction (*MRE*).

4.1 Dataset

4.1.1 MNER Datasets

For the *MNER* task, we use two widely used datasets, `Twitter2015` (Zhang et al., 2018) and `Twitter2017` (Lu et al., 2018), which are collected from Twitter. Each tweet contains a text-image pair and the text may contain zero or more named entities. There are four types of entities: Person (*PER*), Organization (*ORG*), Location (*LOC*) and others (*MISC*). We use the pre-processed datasets provided by Yu et al. (2020) [2]. In total, there are 4,000/1,000/3,357 and 3,373/723/723 sentences in train/development/test set contained in `Twitter2015` and `Twitter2017`, respectively.

4.1.2 MRE Datasets

For the *MRE* task, we use the `MNRE` [3] dataset (Zheng et al., 2021a), which is also collected from Twitter. It contains 9,201 sentences and 15,485 entity pairs with 23 types of relations. In total, there are 12,247/1,624/1,614 entity pairs in train/development/test set, respectively.

4.2 Metrics

We use the micro precision (**P**), recall (**R**) and F1 score (**F1**) to evaluate the performance of both the

[2] https://github.com/jefferyYu/UMT

[3] https://github.com/thecharm/Mega

MNER models and the *MRE* models, which are widely used in recent *MNER* (Moon et al., 2018; Zhang et al., 2018; Lu et al., 2018; Yu et al., 2020; Chen et al., 2021) and *MRE* (Zheng et al., 2021a,b) works.

4.3 Parameter Settings

We conduct all the experiments on `NVIDIA GTX 2080 Ti` GPUs with PyTorch 1.7.1. The parameter settings of our framework are as follows:

- We randomly split the training data into $\mathcal{D}_{model}$ (80%) and $\mathcal{D}_{split}$ (20%).
- For the *MNER* task, we use *UMT-BERT-CRF* (Yu et al., 2020) and *MAF* (Xu et al., 2022) as the multimodal model, respectively. And use *BERT-CRF* as the unimodal model, which consists of *BERT* (Devlin et al., 2018) and *CRF* (John D. Lafferty, 2001). We use the same hyperparameters provided by Yu et al. (2020) to train both the *UMT-BERT-CRF* model and the *BERT-CRF* model.
- For the *MRE* task, we use *MEGA* (Zheng et al., 2021a) as the multimodal model and *MTB* (Soares et al., 2019) as the unimodal model. We use the same hyperparameters provided by Zheng et al. (2021a) to train the *MEGA* model and follow OpenNRE [4] to train the *MTB* model.
- For the data discriminator, we use $CLIP_{32}$ [5] to obtain the representations of text and images in the same latent space.
- For training data discriminator, we use grid search in the development set to find the learning rate of data discriminator η within [$1e^{-5}$, $1e^{-4}$], the batch size B_s within [128, 512], and the hyperparameter of reward function α within [0.1, 0.9] in Algorithm 1.
- All models use mini-batch backpropagation for training and use adam optimizer (Kingma and Ba, 2015) for optimization.

4.4 Evaluation

We first train the multimodal and unimodal models on the full training set and then evaluate the performance of different models on three test sets, which consists of a *Unimodal test set* and a *Multimodal test set* and the *Full test set*. Specifically, the unimodal and multimodal test sets are obtained by using our data discriminator on the *Full test set*. To evaluate our method on the *Full test set*, as mentioned above, we combine the predictions of the unimodal model on the *Unimodal test set* and the predictions of the multimodal model on the *Multimodal test set* as the predictions of our method.

4.5 Performance Comparison

We compare the performance on both the *MRE* and *MNER* tasks, the comparison results are shown in Table 1, Table 2 and Table 3, respectively.

Table 1: Performance Comparison on *MRE* Task.

Model	Test Set	P	R	F1
MTB	**Unimodal**	**60.60**	**71.43**	**65.57**
	Multimodal	60.68	64.64	62.60
	Full	60.65	66.72	63.54
MEGA	Unimodal	64.88	55.61	59.89
	Multimodal	**70.45**	**62.84**	**66.43**
	Full	68.79	60.63	64.45
Ours	**Full**	**66.83**	**65.47**	**66.14**

Specifically, we first compare the performance of each model on different test sets in the *MRE* task. As shown in Table 1, the unimodal relation extraction model *MTB*, performs the best on the *Unimodal test set* and the worst on the *Multimodal test set*. For the multimodal relation extraction model *MEGA*, it performs the best on the *Multimodal test set* and the worst on the *Unimodal test set*. We perform *MTB* on *Unimodal test set* and *MEGA* on *Multimodal test set* as our *MRE* method on *Full test set*. Compared to the performance (**F1**) of the other two models on the *Full test set*, our method achieves the best performance, beating the state-of-the-art *MRE* model (*MEGA*) by 1.69 points. This shows that our data discriminator can effectively split the data, where the *Unimodal test set* is indeed more suitable for unimodal models and the *Multimodal test set* is indeed more suitable for multimodal models and the prediction results of combining the two models on their suitable data can have better performance.

Then, we compare the performance of each model on different test sets in the *MNER* task. As shown in Table 2 and Table 3, we obtain the

[4] https://github.com/thunlp/OpenNRE
[5] https://huggingface.co/openai/clip-vit-base-patch32

Table 2: Performance Comparison on *MNER* Task (UMT-BERT-CRF as the multimodal model).

Model	Test Set	Twitter2015			Twitter2017		
		P	**R**	**F1**	**P**	**R**	**F1**
BERT-CRF	**Unimodal**	**71.35**	**75.32**	**73.28**	**85.58**	**85.13**	**85.36**
	Multimodal	71.23	73.99	72.58	83.68	81.56	82.60
	Full	71.29	74.63	72.92	84.76	83.57	84.16
UMT-BERT-CRF	Unimodal	70.50	75.12	72.73	84.21	83.55	83.88
	Multimodal	**72.50**	**74.95**	**73.71**	**84.92**	**85.79**	**85.35**
	Full	71.50	74.96	73.19	84.74	84.68	84.71
Ours	**Full**	**71.94**	**75.13**	**73.50**	**85.29**	**85.42**	**85.36**

Table 3: Performance Comparison on *MNER* Task (MAF as the multimodal model).

Model	Test Set	Twitter2015			Twitter2017		
		P	**R**	**F1**	**P**	**R**	**F1**
BERT-CRF	**Unimodal**	**71.35**	**75.31**	**73.28**	**85.71**	**84.03**	**84.87**
	Multimodal	71.24	74.00	72.60	84.42	83.40	83.91
	Full	71.29	74.63	72.92	84.76	83.57	84.16
MAF	Unimodal	71.13	75.39	73.20	84.99	84.03	84.51
	Multimodal	**72.53**	**74.70**	**73.60**	**86.44**	**87.22**	**86.83**
	Full	71.85	75.04	73.41	86.06	86.38	86.22
Ours	**Full**	**71.96**	**75.00**	**73.44**	**86.25**	**86.38**	**86.32**

same conclusions as for the *MRE* task, i.e., the unimodal named entity recognition model *BERT-CRF* performs the best on the *Unimodal test set* and the worst on the *Multimodal test set*, and the multimodal named entity recognition model *UMT-BERT-CRF* and *MAF* perform the best on the *Multimodal test set* and the worst on the *Unimodal test set*. We also perform *BERT-CRF* on *Unimodal test set* and *UMT-BERT-CRF* or *MAF* on *Multimodal test set* as our *MNER* method on *Full test set*. When the multimodal model is *UMT-BERT-CRF*, our method outperforms it by 0.31 and 0.65 points on `Twitter2015` and `Twitter2017`, respectively. Our method also outperforms it when the multimodal model is *MAF*, which illustrates that the multimodal model in our method can be replaced by any existing multimodal model, including the one that already considers the mismatched image problem. But the improvement is smaller compared to using *UMT-BERT-CRF* as a multimodal model because (1) we only try a few different sets of hyperparameters on *MAF* and (2) *MAF* considers the problem of mismatched image and has a strong robustness to the mismatched image.

4.6 Case Study

To show the effectiveness of our data discriminator more intuitively, we perform the data discriminator on the test set of `TWITTER-2017` and select six samples for analysis based on the probability predicted by the data discriminator. Note that the data discriminator puts data with probability greater than 0.5 into the multimodal set and data with probability less than or equal to 0.5 into the unimodal set. Specifically, Fig. 3(a) and Fig. 3(b) show the two samples with the lowest probability, Fig. 3(c) and Fig. 3(d) show the two samples with the highest probability, and Fig. 3(e) and Fig. 3(f) show the two samples with the medium probability.

Firstly, for the two samples with the lowest probability, we can observe that there is enough information in their text to recognize the named entities while the images do not provide any useful information to help identify the entities in the text. Specifically, in Fig. 3(a), the unimodal *NER* model can easily recognize from the text that `Southside Festival` is a named entity and the type is *MISC* through the capitalization of the two words and the meaning of the words themselves, and there is no

information in the image related to `Southside Festival` to help identify named entities. In Fig. 3(b), the unimodal *NER* model can easily recognize from the text that `LA` and `NY` are named entities and their type is *LOC* through all capitals of the words and the preposition `in`. The main part of the image is a dragonfly, which cannot help identify named entities. In summary, the two samples with the lowest probability are suitable for the unimodal *NER* model, and the use of the multimodal *NER* model will result in poor results due to the introduction of image noise.

(a) Glad they're putting on more seats for the [**Southside Festival MISC**] this year

(b) This beautiful creature visited me in [**LA LOC**] yesterday - a few hours after a dragonfly visited my son in [**NY LOC**]

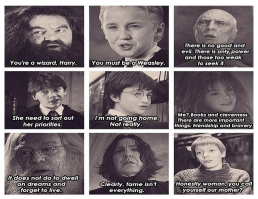

(c) Memorable quotes from [**Harry Potter and the Philosopher's Stone MISC**].

(d) [**jjong PER**] is wearing [**R.Shemiste ORG**] F/W 2016 inspired by sociopolitical activist

(e) Take a look at our new look on -line #Football store here:

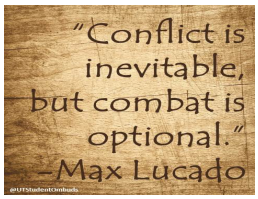

(f) Work through your conflicts with the student ombuds!

Figure 3: Case Study on `TWITTER-2017`.

Secondly, for the two samples with the highest probability, we can observe that there is not enough information in their text to recognize the named entities. Specifically, in Fig. 3(c), `Harry Potter and the Philosopher's Stone` in the text is the name of a movie and should be classified as *MISC*, but it may also be divided into two parts, `Harry Potter` and `the Philosopher's Stone`, where `Harry Potter` is classified as *PER*. Obviously, there is ambiguity in using only text information, so additional image information is required. In Fig. 3(d), `R.Shemiste` is usually recognized as the name of a person, with the help of the image, we can infer that it is a brand name. In summary, the two samples with the highest probability are suitable for the multimodal *NER* model, and the use of the unimodal *NER* model will result in poor results due to the lack of sufficient information.

Finally, for the two medium probability samples, we find an interesting phenomenon: their images are composed of text and backgrounds. This phenomenon is very common in medium probability samples. The image encoder neither obtains useful information from these images nor introduces noise. Therefore, the multimodal *NER* model degenerates into a unimodal *NER* model. Therefore, it does not matter whether using a multimodal *NER* model or a unimodal model.

5 Related Work

In this section, we review and summarize three multimodal *IE* tasks, namely multimodal named entity recognition, multimodal sentiment detection and multimodal relation extraction.

For the multimodal named entity recognition task, at the earliest, Moon et al. (2018) inputs the whole image into a convolutional neural network (e.g. *ResNet*) to obtain a representation of the whole image to establish the relationship between the text and the image. Since only some regions in the image are useful for recognizing entities, (Lu et al., 2018; Zhang et al., 2018; Yu et al., 2020; Chen et al., 2021; Sun et al., 2021; Xu et al., 2022) divide the image into multiple regions, obtained a representation of the image regions and establish a relationship between the image regions and each word in the text. Next, since the image and text representations come from different encoders, there is a semantic gap that affects the establishment of image and text relationships. To solve the above problem, Xu et al. (2022) proposes an alignment and matching framework to make the text and image representations more consistent by contrastive learning. More directly, (Wu et al., 2020; Wang et al., 2021) extract the semantic information of the image directly to represent the image: Wu et al. (2020) extracts the objects in the image by

the object detection model and uses the labels of the objects (e.g., apple, trophy) to represent the image, while (Wang et al., 2021) uses the image captioning model and the OCR model in addition to the object detection model to obtain the overall semantic information of the image and the textual information in the image, respectively. In addition, (Sun et al., 2021; Xu et al., 2022) noticed that the mismatched image can impair the prediction of multimodal models and both propose a method to calculate the image and text similarity scores to filter the image information.

For the multimodal relation extraction task, (Zheng et al., 2021b) first propose this task and demonstrate that previous text-based relation extraction models perform poorly in social media texts, and that incorporating visual information can help improve relation extraction model performance. Subsequently, in order to be able to fully exploit the relationships between objects in the image and to establish the alignment of the image with the text, Zheng et al. (2021a) use graph structure information to align the relations between entities in text and images and then uses image information to supplement the missing semantic information.

For the multimodal sentiment detection task, Xu et al. (2018) obtain the information more important for sentiment by capturing the correlation between text and images. Huang et al. (2018) use an adversarial learning model to learn a joint multimodal representation to combine text and image representations. Yang et al. (2021) use a novel graph neural network based on the global characteristics that encode different modalities to capture hidden representations and learn multimodal representations.

However, current approaches overestimate the significance of images. Although several works (Sun et al., 2021; Xu et al., 2022) have proposed methods to filter image information, they all use the similarity scores of images and text to filter the image information as a whole, more or less retaining some image information and not accurately filtering out the noise in the images. Therefore, we propose a general data splitting technique to process different data using different models (i.e., multimodal model and unimodal model).

6 Conclusion

In this paper, we propose a general data splitting strategy to divide the social media posts into two sets so that these two sets can achieve better performance under the information extraction models of the corresponding modalities. The core component is the data discriminator. Due to the lack of explicit knowledge, we use reinforcement learning to train the data discriminator. Experiments conducted on two widely used multimodal named entity recognition datasets and a multimodal relation extraction dataset show that our data discriminator can effectively split the data and our proposed data splitting strategy for multimodal information extraction achieves the best performance.

References

Dawei Chen, Zhixu Li, Binbin Gu, and Zhigang Chen. 2021. Multimodal named entity recognition with image attributes and image knowledge. In *Database Systems for Advanced Applications*, pages 186–201.

Shailaja R Deshmukh1. 1991. Bernoulli sampling. *Australian Journal of Statistics*, 33(2):167–176.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*.

Feiran Huang, Xiaoming Zhang, and Zhoujun Li. 2018. Learning joint multimodal representation with adversarial attention networks. In *Proceedings of the 26th ACM international conference on Multimedia*, pages 1874–1882.

Fernando C. N. Pereira John D. Lafferty, Andrew McCallum. 2001. Conditional random fields: Probabilistic models for segmenting and labeling sequence data. In *Proc. of the 18th Intl. Conf. on Machine Learning (ICML-2001)*, pages 282–289.

Diederik P Kingma and Jimmy Ba. 2015. Adam: A method for stochastic optimization. In *ICLR (Poster)*.

Di Lu, Leonardo Neves, Vitor Carvalho, Ning Zhang, and Heng Ji. 2018. Visual attention model for name tagging in multimodal social media. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1990–1999.

Seungwhan Moon, Leonardo Neves, and Vitor Carvalho. 2018. Multimodal named entity recognition for short social media posts. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 852–860.

Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, et al. 2021. Learning transferable visual models from natural language supervision. In *International*

Conference on Machine Learning, pages 8748–8763. PMLR.

Rico Sennrich, Barry Haddow, and Alexandra Birch. 2016. Neural machine translation of rare words with subword units. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1715–1725.

Livio Baldini Soares, Nicholas Fitzgerald, Jeffrey Ling, and Tom Kwiatkowski. 2019. Matching the blanks: Distributional similarity for relation learning. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 2895–2905.

Lin Sun, Jiquan Wang, Kai Zhang, Yindu Su, and Fangsheng Weng. 2021. Rpbert: a text-image relation propagation-based bert model for multimodal ner. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 35, pages 13860–13868.

Alakananda Vempala and Daniel Preoţiuc-Pietro. 2019. Categorizing and inferring the relationship between the text and image of twitter posts. In *Proceedings of the 57th annual meeting of the Association for Computational Linguistics*, pages 2830–2840.

Xinyu Wang, Min Gui, Yong Jiang, Zixia Jia, Nguyen Bach, Tao Wang, Zhongqiang Huang, Fei Huang, and Kewei Tu. 2021. Ita: Image-text alignments for multi-modal named entity recognition. *arXiv preprint arXiv:2112.06482*.

Zhiwei Wu, Changmeng Zheng, Yi Cai, Junying Chen, Ho-fung Leung, and Qing Li. 2020. Multimodal representation with embedded visual guiding objects for named entity recognition in social media posts. In *Proceedings of the 28th ACM International Conference on Multimedia*, pages 1038–1046.

Bo Xu, Shizhou Huang, Chaofeng Sha, and Hongya Wang. 2022. Maf: A general matching and alignment framework for multimodal named entity recognition. In *Proceedings of the Fifteenth ACM International Conference on Web Search and Data Mining*, pages 1215–1223.

Nan Xu, Wenji Mao, and Guandan Chen. 2018. A co-memory network for multimodal sentiment analysis. In *The 41st international ACM SIGIR conference on research & development in information retrieval*, pages 929–932.

Xiaocui Yang, Shi Feng, Yifei Zhang, and Daling Wang. 2021. Multimodal sentiment detection based on multi-channel graph neural networks. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 328–339.

Jinsung Yoon, Sercan Arik, and Tomas Pfister. 2020. Data valuation using reinforcement learning. In *International Conference on Machine Learning*, pages 10842–10851. PMLR.

Jianfei Yu, Jing Jiang, Li Yang, and Rui Xia. 2020. Improving multimodal named entity recognition via entity span detection with unified multimodal transformer. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 3342–3352.

Qi Zhang, Jinlan Fu, Xiaoyu Liu, and Xuanjing Huang. 2018. Adaptive co-attention network for named entity recognition in tweets. In *Proceedings of the Thirty-Second AAAI Conference on Artificial Intelligence and Thirtieth Innovative Applications of Artificial Intelligence Conference and Eighth AAAI Symposium on Educational Advances in Artificial Intelligence*, pages 5674–5681.

Changmeng Zheng, Junhao Feng, Ze Fu, Yi Cai, Qing Li, and Tao Wang. 2021a. Multimodal relation extraction with efficient graph alignment. In *Proceedings of the 29th ACM International Conference on Multimedia*, pages 5298–5306.

Changmeng Zheng, Zhiwei Wu, Junhao Feng, Ze Fu, and Yi Cai. 2021b. Mnre: A challenge multimodal dataset for neural relation extraction with visual evidence in social media posts. In *2021 IEEE International Conference on Multimedia and Expo (ICME)*, pages 1–6. IEEE.

Augmentation, Retrieval, Generation: Event Sequence Prediction with a Three-Stage Sequence-to-Sequence Approach

Bo Zhou[1,2]**,Chenhao Wang**[1,2]**, Yubo Chen**[1,2]**, Kang Liu**[1,2,4]**, Jun Zhao**[1,2]**,**
Jiexin Xu[3]**, Xiaojian Jiang**[3]**, Qiuxia Li**[3]
[1]School of Artificial Intelligence, University of Chinese Academy of Sciences
[2]National Laboratory of Pattern Recognition, CASIA
[3]China Merchants Bank, [4]Beijing Academy of Artificial Intelligence
{bo.zhou,chenhao.wang,yubo.chen,kliu,jzhao}@nlpr.ia.ac.cn
{jiexinx,jiangxiaojian,annielqx}@cmbchina.com

Abstract

Being able to infer possible events related to a specific target is critical to natural language processing. One challenging task in this line is *event sequence prediction*, which aims at predicting a sequence of events given a goal. Currently existing approach models this task as a *statistical induction* problem, to predict a sequence of events by exploring the similarity between the given goal and the known sequences of events. However, this statistical based approach is complex and predicts a limited variety of events. At the same time this approach ignores the rich knowledge of external events that is important for predicting event sequences. In this paper, in order to predict more diverse events, we first reformulate the event sequence prediction problem as a sequence generation problem. Then to leverage external event knowledge, we propose a three-stage model including augmentation, retrieval and generation. Experimental results on the event sequence prediction dataset show that our model outperforms existing methods, demonstrating the effectiveness of the proposed model.

1 Introduction

Inferring events related to a specific target is one of the capabilities pursued by natural language understanding, and it is also very helpful for other natural language processing (NLP) tasks, such as event extraction (Li et al., 2021), text summarization (Li and Zhang, 2021) and discourse understanding (Nie et al., 2019). A challenging task in this direction is event sequence prediction. Specifically, given a goal, the task aims to predict a sequence of events that fits the goal. Figure 1 shows an example of the event sequence prediction task. Given the goal *Buy a mobile phone*, it is expected to predict an event sequence, including four events *Determine the brand*, *Determine the price range*, *Select the brand series* and *Pay the bill*.

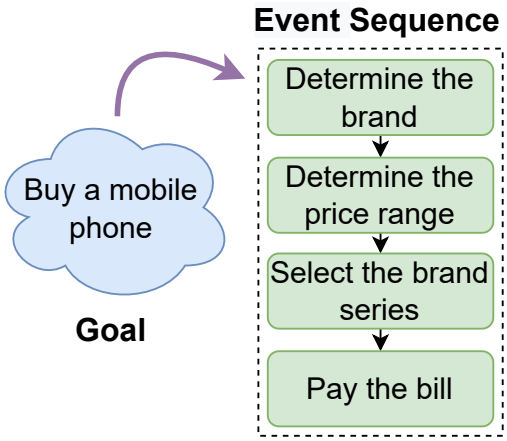

Figure 1: An example of an event sequence prediction task. Given the goal *Buy a mobile phone*, the predicted event sequence contains four events.

Although event sequence prediction is useful for many NLP tasks, it is currently understudied. Currently existing approach (Zhang et al., 2020) to this task is based on statistical model that infers new sequence of events by exploiting the similarity of the given goal to known sequences of events. Although experiments have shown that this approach is effective, it still has two shortcomings.

First, a limited variety of events are predicted. For example, when predicting the event sequence with the goal *Buying a mobile phone*, the method first collects the known event sequences with the goal *Buying a house, Buying a book, Repairing a mobile phone, Selling a mobile phone*, etc., and then predicts a new event sequence based on these event sequences. The events in the predicted sequence of events are similar to the events in the known sequence of events, or roughly, different combinations between these known events. Therefore, the types of events predicted by this method are limited.

Second, rich knowledge of external events is ignored. This method predicts a new event sequence based on known event sequences. If the

Proceedings of the 29th International Conference on Computational Linguistics, pages 1865–1874
October 12–17, 2022.

to-be-predicted event sequence differs greatly from the existing ones, it is difficult to predict a satisfactory event sequence. For example, if known sequences of events have neither Create-related nor Novel-related sequences of events, it is difficult to generate a satisfactory sequence of events given the goal *Create a novel*. At the same time, existing work (Guan et al., 2020; Li et al., 2022) shows the external event knowledge is helpful for natural language understanding and text generation, because these event knowledge bases contain rich commonsense knowledge that indicates the relationship between events. Therefore, leveraging these external event knowledge is important for event sequence prediction tasks.

In this paper, we first reformulate event sequence prediction as event sequence generation for the problem of limited types of generated events. Second, we propose a three-stage generation model for the problem of ignoring external event knowledge. As shown in Figure 2, our model consists of three steps, 1) Augmentation: First we design two pre-training tasks using the external event knowledge base, and then train the generation model with these two tasks. 2) Retrieval: Secondly, we match several similar event sequences from the existing event sequences through a given goal, and then train a scoring model to select the most similar event sequence. 3) Generation: Finally, we input the given goal and the retrieved most similar event sequences into the generation model to generate event sequence that meet the given goal. The experimental results show that our method outperforms previous methods, and the ablation study also verify the effectiveness of each module. In summary, our contributions in this paper are as follows:

- We reformulate the event sequence prediction task as a sequence generation task, which can generate more diverse events than method that infers new sequence of events from known events.
- We propose a three-stage augmentation, retrieval, generation model to tackle the event sequence prediction task, which can leverage external event knowledge to better generate sequence of events.
- Experiments and detailed analysis show that our model outperforms previous methods, proving its effectiveness.

2 Our Framework

Given a goal G, the event sequence prediction task needs to predict an event sequence $(e_1, e_2, ..., e_i, ..., e_n)$ that fits the goal, where e_i is the predicted i-th event. Here both the goal G and the event e_i are mainly composed of a verb and an object. Figure 2 shows the framework of our model which consists of three modules, namely augmentation, retrieval and generation. Below we introduce these modules separately.

2.1 Augmentation

In the augmentation module, we utilize the external event knowledge base ATOMIC [1] and design two tasks to pretrain our generation model.

To benefit the event sequence prediction task as much as possible, we expect the data of the external event knowledge base to be as similar as possible to the data of event sequence prediction, and we choose ATOMIC as a result. ATOMIC is a commonsense knowledge base with everyday knowledge tuples about entities and events in the form of *(head, relation, tail)* (Hwang et al., 2021). For example, the knowledge tuple *(move towards the door, isBefore, run out of the room)* indicates that the head event *move towards the door* occurs before the tail event *run out of the room*.

Using ATOMIC, we propose two pre-training tasks that match the event sequence prediction objective as closely as possible, one is Tail Event Generation (TG) and the other is Order Recovery (OR). Specifically, given a head event e_h, the TG task aims to generate a tail event e_t corresponding to e_h. The TG task empowers the generation model to generate events related to a given event, just as the model generates events related to a given goal in event sequence prediction task. Because the TG task generates only one event, it fails to capture the relationship between different generated events. So we propose the OR task, hoping to give the model the ability to distinguish the order between events. Specifically, given a head event e_h and its corresponding tail event e_t (Here we assume that e_h happens before e_t.), regardless of whether the input is ordered pair (e_h, e_t) or reversed pair (e_t, e_h), we want the model to output ordered pair (e_h, e_t).

Then we mix the data pairs of TG task with the data pairs of OR task for training. Specifically, for the OR task, given (e_h, e_t), we take (e_t, e_h) as in-

[1] https://mosaickg-graph-viz.apps.allenai.org/kg_atomic2020

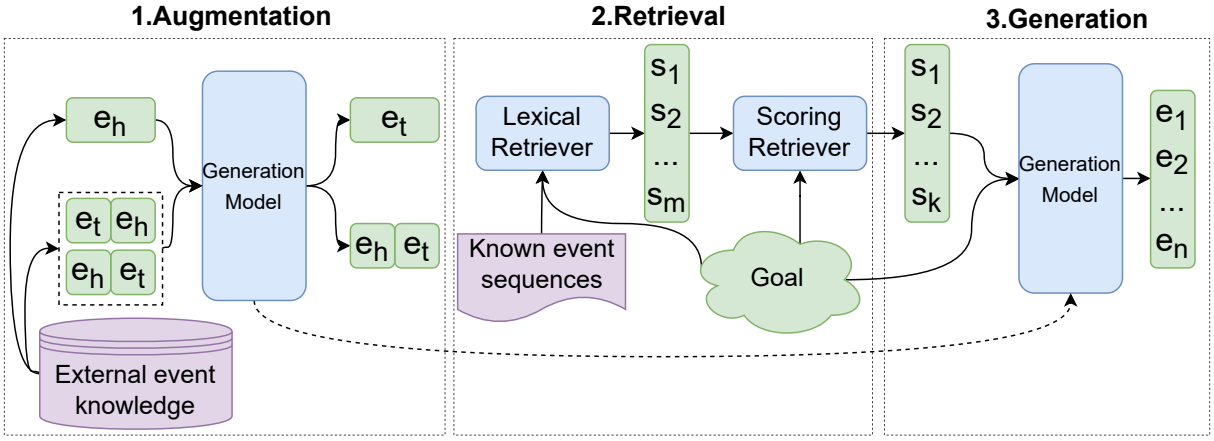

Figure 2: The framework of our model which consists of three modules, namely augmentation, retrieval and generation. Here e_h and e_t represent the head and tail events in the external event knowledge base, s_i and e_i denote the i-th event sequence and the generated i-th event, respectively.

put with probability p and (e_h, e_t) with probability $1-p$. We adopt the pre-trained encoder-decoder model T5-base (Raffel et al., 2021) as our base model and these mixed data pairs are used to train it with the following generation loss:

$$\mathcal{L}_{\mathrm{a}} = -\sum_{t=1}^{|y|} \log p\left(y_t \mid x, y_{<t}\right) \tag{1}$$

where (x, y) is the input-output pair, which may be (e_h, e_t), $[(e_t, e_h), (e_h, e_t)]$ or $[(e_h, e_t), (e_h, e_t)]$.

Because not all relations in ATOMIC are suitable for training our two tasks, we only use some relations in ATOMIC. Specifically, for the TG task, we use the *HasSubEvent* relation (The tail event is a step within the larger head event.) and the *xNeed* relation (The tail event is the condition of the head event.). For the OR task, we use the *isBefore* relation (The head event happens before the tail event.) and the *isAfter* relation (The head event happens after the tail event.). Table 1 shows the relations used by the two tasks and the corresponding examples.

At the same time, we point out that it is also possible to choose other event knowledge bases and design other pre-training tasks for event sequence prediction, which we leave as future work.

2.2 Retrieval

When predicting an event sequence that fits a given goal, it is helpful to refer to known event sequences related to the given goal. Therefore, in the retrieval module, we first use the lexical retriever to roughly retrieve m event sequences related to the given goal from the known event sequences, and then use the scoring retriever to further retrieve k most relevant event sequences from the m event sequences.

2.2.1 Lexical Retriever

Given a goal $G = (v, o)$, where v is the verb of the goal and o is the object of the goal. The lexical retriever returns event sequences whose goals contain either v or o. We denote these returned event sequences as $\{s_1, s_2, \cdots, s_m\}$, and the scoring retriever will then further select from these event sequences the ones most similar to the given goal.

2.2.2 Scoring Retriever

The scoring retriever scores each event sequence returned by the lexical retriever, and the top-k with the highest scores are used as the final retrieved event sequences. We train a BERT (Devlin et al., 2019) model as our scorer. Specifically, for each goal G, we take it and its corresponding event sequence as a positive pair. At the same time, we randomly sample an event sequence under another goal from the training set to form a negative sample pair with G.

Given a goal G and an event process s_i, we first concatenate the two as input to BERT:

$$X = [CLS]\ G\ [SEP]\ s_i\ [SEP] \tag{2}$$

here $[CLS]$ and $[SEP]$ are BERT's classification token and separation token, respectively.

We feed X into BERT and then feed the hidden layer vector $\boldsymbol{h}$ corresponding to CLS into a feedforward network:

$$s = \sigma(W\boldsymbol{h} + b) \tag{3}$$

Pretraining Task	Relation	Head	Tail
Tail Event Generation	HasSubEvent	play joker	shuffle cards
	xNeed	maintain good health	avoid cigarettes
Order Recovery	isBefore	go bike riding	take some photos of the scenery
	isAfter	move towards the door	get up from the table

Table 1: The relations used by the two pretraining tasks and the corresponding examples.

here the σ is the sigmoid function. We train the scorer by the cross entropy loss for binary classification. The s is taken as the score for each event sequence returned by the lexical retrieval. We obtain the top-k event sequences with the highest scores as the final retrieval result, denoted as $\{s_1, s_2, \cdots, s_k\}$.

2.3 Generation

In the generation module, we utilize the given goal and the event sequences retrieved by the retrieval module to generate the event sequence that matches the given goal.

Our generation model is also based on the T5-base model, sharing parameters with the T5 model in the augmentation module. Given the target G, we concatenate it with the retrieved sequence of events $\{s_1, s_2, \cdots, s_k\}$ as the input to the generation model and the sequence of events corresponding to G as the output y, and then optimize the following generation loss:

$$\mathcal{L}_{\mathrm{g}} = -\sum_{t=1}^{|y|} \log p\left(y_t \mid G, \{s_1, s_2, \cdots, s_k\}, y_{<t}\right) \tag{4}$$

3 Experiments

3.1 Settings

3.1.1 Dataset and Metrics

For the event sequence prediction task, we use the dataset released in Zhang et al. (2020). This dataset is based on WikiHow [2], an online wiki-style publication containing many sequences of events that accomplish specific goals. Following Zhang et al. (2020), the numbers of training, validation and test sets are 12,185, 1,316 and 1,316, respectively.

For the two pre-training tasks of tail event generation and order recovery, as mentioned earlier, we use the data corresponding to the four relations *HasSubEvent*, *xNeed*, *isBefore* and *isAfter* in ATOMIC. Because pre-training does not require test data, we end up with 152,673 and 16,964 data for training and validation sets, respectively.

We use the same metrics as the work (Zhang et al., 2020): E-ROUGE1 and E-ROUGE2. Similar to the commonly used ROUGE1 and ROUGE2, E-ROUGE1 and E-ROUGE2 calculate the percentage of the events or ordered event pairs in the predicted event sequence which are covered by the reference event sequences. Two covering standards *String Match* and *Hypernym Allowed* are used to evaluate the results. The former indicates that the words in the predicted event or event pair should be the same as in the references. The latter means that the hypernyms of the words in the predicted event or event pair should be the same as the hypernyms of the words in the references. Finally, two kinds of settings are included in the evaluation. One is basic setting: evaluate events based on verbs. The other is advanced setting: evaluate events based on all words.

3.1.2 Baselines

We compare with the following baseline models, which are used in Zhang et al. (2020):

- **Random**. Given a goal, the event sequence is randomly generated. This can be regarded as a lower bound on the performance of event sequence prediction.
- **GRU** (Sutskever et al., 2014). This is based on the GRU model, but the generation unit is changed from words to events. Events are represented by the pre-trained word embedding GloVe (Pennington et al., 2014) or the language model RoBERTa-base (Liu et al., 2019), denoted as **GRU (GloVe)** and **GRU (RoBERTa)** respectively.
- **Top one process**. Given a goal, such methods take the most similar sequence of events as the predicted sequence of events. Three methods for measuring similarity are used here, namely token-level Jaccard coefficient, cosine similarity based on GloVe representations and cosine

[2] https://www.wikihow.com/Main-Page

(a) Basic Setting (evaluate events based on verbs)

Model	String Match		Hypernym Allowed	
	E-ROUGE1	E-ROUGE2	E-ROUGE1	E-ROUGE2
Random	2.9165	0.4664	23.5873	8.1089
GRU (GloVe)	5.0323	1.4965	27.8710	13.0946
GRU (RoBERTa)	4.5455	0.4831	28.0032	12.8502
Top one process (Jaccard)	8.8589	5.1000	28.6548	14.6231
Top one process (GloVe)	9.8797	5.1452	29.4203	13.6001
Top one process (RoBERTa)	9.2599	4.7390	30.6599	15.8417
APSI	14.8013	6.6045	36.1648	19.2418
Ours	**24.9551**	**11.2535**	**46.1999**	**23.5655**
Human	29.0189	15.2542	50.4647	29.4423

(b) Advanced Setting (evaluate events based on all words)

Model	String Match		Hypernym Allowed	
	E-ROUGE1	E-ROUGE2	E-ROUGE1	E-ROUGE2
Random	0.0000	0.0000	0.5104	0.0903
GRU (GloVe)	0.1935	0.0534	0.9677	0.1069
GRU (RoBERTa)	0.4870	0.0000	1.7857	0.2899
Top one process (Jaccard)	0.6562	0.2257	2.4797	0.5867
Top one process (GloVe)	0.8750	0.2106	2.8801	0.7372
Top one process (RoBERTa)	0.9479	0.3009	3.2811	0.9929
APSI	3.4988	0.4513	6.1611	1.1885
Ours	**6.0443**	**1.1142**	**10.7720**	**2.4513**
Human	11.6351	5.5905	18.0034	8.2695

Table 2: Comparison of our method with the best previous methods. The best performance is shown in bold.

similarity based on RoBERTa representations, which are denoted as **Top one process (Jaccard)**, **Top one process (GloVe)** and **Top one process (RoBERTa)** respectively.

- **APSI** (Zhang et al., 2020). This is a statistical model that exploits the similarity of the given goal to known sequences of events to infer new sequence of events.

- **Human**. Given a goal, the event sequence is generated by human, which can be regarded as an upper bound on the performance of event sequence prediction.

3.1.3 Implementation Details

We use T5-base as the base model for the augmentation module and the generation module. The batch size used in both modules is 32. The optimizer used by both modules is AdamW (Loshchilov and Hutter, 2018), and the learning rate is set to 1e-5. In the augmentation module, the model is trained for 15 epochs, and the number is 30 in the generation module. We use BERT-base as the base model for the scoring retriever, AdamW is chosen as the optimizer and the learning rate is set to 5e-5. We set the batch size to 64 and train the BERT-base model for 3 epochs. We experimentally choose the number k of event sequences retrieved by the scoring retriever to be 2. Both T5-base and BERT-base models are implemented through the Huggingface Transformer library (Wolf et al., 2020).

3.2 Overall Results

The overall results are shown in Table 2, from which we can see:

(1) Our model outperforms previous methods in both basic and advanced settings, both for *String Match* or *Hypernym Allowed*. This demonstrates that our method can generate event sequences that are more in line with the given goal.

(2) The performance of all methods is inferior to human performance, but the performance of our proposed method is close to human performance. The likely reason is that our method somewhat mimics the human process of generating event sequences. Humans first learn knowledge related to

(a) Basic Setting (evaluate events based on verbs)

Model	String Match		Hypernym Allowed	
	E-ROUGE1	E-ROUGE2	E-ROUGE1	E-ROUGE2
T5	21.4396	8.6654	44.1595	22.3597
T5+augmentation	21.6580	9.3587	44.9007	23.0440
T5+retrieval	23.0099	9.8411	45.0442	22.4762
Ours (T5+augmentation+retrieval)	24.9551	11.2535	46.1999	23.5655

(b) Advanced Setting (evaluate events based on all words)

Model	String Match		Hypernym Allowed	
	E-ROUGE1	E-ROUGE2	E-ROUGE1	E-ROUGE2
T5	5.4869	0.8742	9.8168	2.0505
T5+augmentation	5.6759	0.9442	10.3454	2.1537
T5+retrieval	5.8754	1.0814	10.6658	2.2291
Ours (T5+augmentation+retrieval)	6.0443	1.1142	10.7720	2.4513

Table 3: Ablation results on the test set of event sequence prediction. We use T5 as our base model.

event sequences. When given a goal, humans first search for similar event sequences, and then use these event sequences to generate event sequence that matches the goal.

(3) When using *Hypernym Allowed* instead of *String Match*, performance is greatly improved in both settings. The reason is that it is easier to predict similar words than to predict the same words to the answer. The better performance of our model over the previous methods on both *String Match* and *Hypernym Allowed* shows that our method is more able to generate accurate events, or at least, our model is more able to generate events similar to the answer than the previous methods. We also observe that the performance of the model in the advanced setting is inferior to that in the basic setting, suggesting that it is easier for the model to correctly predict the verb in an event than to correctly predict the entire event.

3.3 Ablation Study

Here we conduct ablation experiments to investigate the effect of various modules of our method. The results are shown in Table 3, from which we can observe:

(1) Our model (T5+augmentation+retrieval) outperforms the base T5 model on all metrics in both basic and advanced settings, indicating that the introduction of augmentation and retrieval modules can help improve the performance of event sequence prediction.

(2) When the basic model T5 is added with the augmentation module, the performance of the model in various metrics is also improved. This shows that the introduction of external event knowledge is helpful for event sequence prediction. By endowing the model the ability to generate relevant events and capture the relationship between events by the two pre-training tasks, we successfully inject external knowledge into the model.

(3) After adding the retrieval module, the performance of the base model T5 is improved on all metrics. This illustrates the importance of referring to known similar event sequences when generating event sequence that meets a given goal. Although large-scale pre-trained language models have been shown to possess some general world knowledge, it is necessary to introduce specific event knowledge by referring to known event sequences when generating new event sequence.

3.4 Effect of Pretraining Tasks

Here we study the effect of the two pre-training tasks Tail Event Generation (TG) and Order Recovery (OR). The results are shown in Table 4, from which we can see:

(1) Whether adding only Tail Event Generation or only Oder Recovery pre-training tasks, the performance of the model improves over models that do not utilize pre-training tasks. This suggests that both tasks can endow the model with the ability to facilitate event sequence prediction by implicitly injecting external event knowledge into the model.

(2) When two pre-training tasks are added at the same time, the performance of the model is further improved. This shows that the two pre-training

(a) Basic Setting (evaluate events based on verbs)

Model	String Match		Hypernym Allowed	
	E-ROUGE1	E-ROUGE2	E-ROUGE1	E-ROUGE2
No pretraining	23.0099	9.8411	45.0442	22.4762
Only Tail Event Generation	24.2118	10.8155	46.0440	23.5396
Only Order Recovery	24.9415	9.9171	45.0969	22.7561
Both	24.9551	11.2535	46.1999	23.5655

(b) Advanced Setting (evaluate events based on all words)

Model	String Match		Hypernym Allowed	
	E-ROUGE1	E-ROUGE2	E-ROUGE1	E-ROUGE2
No pretraining	5.8754	1.0814	10.6658	2.2291
Only Tail Event Generation	5.9299	1.1011	10.6725	2.3762
Only Order Recovery	6.0424	1.0905	10.6745	2.4341
Both	6.0443	1.1142	10.7720	2.4513

Table 4: The effect of the two pre-training tasks Tail Event Generation (TG) and Order Recovery (OR). No pre-training means only the base model with the retrieval module is used.

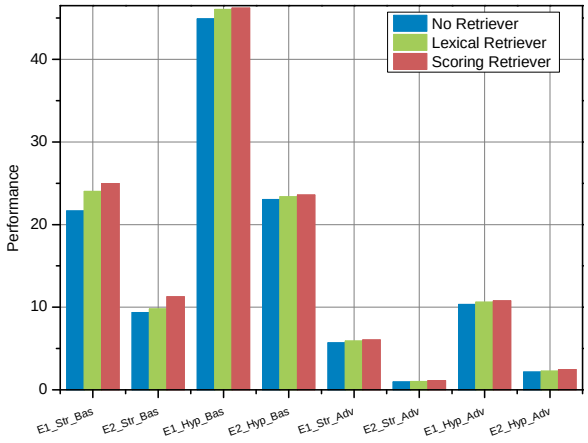

Figure 3: The effect of the lexical retriever and scoring retriever. The *E1_Str_Bas* represents E-ROUGE1 of String Match in the basic setting, and *E2_Hyp_Adv* represents E-ROUGE2 of Hypernym Allowed in the advanced setting. Other abbreviations have similar meanings.

tasks can coexist harmoniously, and at the same time endow the model with the ability to generate related events and capture the relationship between events, thereby jointly helping the event sequence generation.

3.5 Impact of Retrievers

Here we study the effect of the lexical retriever and scoring retriever in the retrieval module. When only the lexical retriever is used, we randomly select k event sequences from the results returned by the lexical retriever as the final retrieval result. The experimental results are shown in Figure 3, from which we can observe:

(1) When only the lexical retriever is used, the performance is improved compared to without the retrieval module. This shows that when generating event sequences, it is helpful to introduce specific event knowledge, i.e., existing event sequences, even if these event sequences are only related to a certain extent.

(2) When a scoring retriever is added, the performance of the model is further improved compared to using only the lexical retriever. This illustrates the necessity of using a scorer to select the sequences of events most similar to a given goal from the related sequences of events. So how to design a better scorer is an important issue, which we leave as future work.

3.6 Case Study

Figure 4 lists three examples of event sequences generated by our model and their corresponding answer event sequences. For the first example, two events *Open App* and *Delete Contact* in the sequence generated by our model are the same as the corresponding events in the answer sequence. Another event *Tap Icon*, although not the same as event *Select Contact* in the answer, expresses a similar meaning. This shows that our model can generate new event types. A similar situation exists for the other two examples.

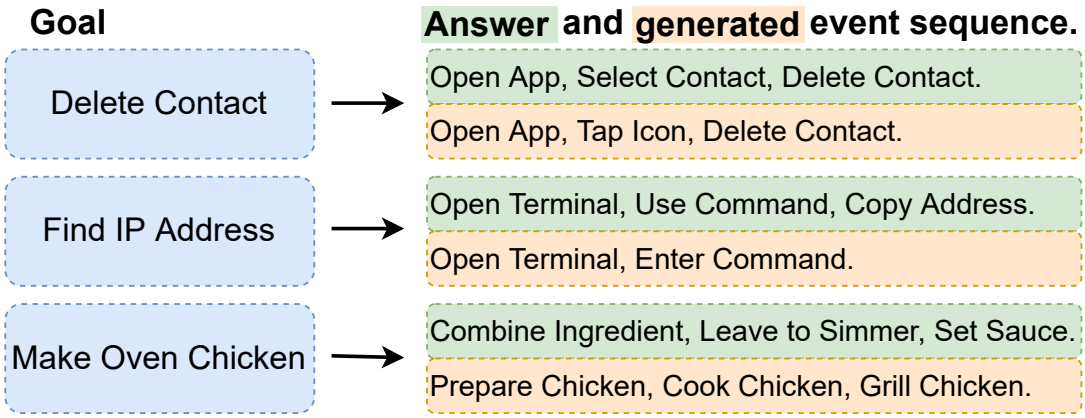

Figure 4: A comparison of the answer event sequence with the event sequence generated by our model.

4 Related Work

4.1 Script Event Prediction

Script event prediction aims to predict a correct subsequent event from a candidate list, given an ordered event sequence. The task is first proposed by Chambers and Jurafsky (2008), with a statistical model to predict the subsequent event by learning the cooccurrence between events. Neural networks are used in recent studies for script event prediction. In Pichotta and Mooney (2016), they shows that LSTM-based model outperforms previous cooccurrence-based models for script event prediction. A neural network is proposed in Granroth-Wilding and Clark (2016) which can learn word embedding and composition function simultaneously. To better model relatedness between events, Li et al. (2018) treats event chain as a sub-graph and leverage recurrent networks to capture the relatedness for predicting the event. The model in Lv et al. (2020) integrates external event knowledge and designs three methods to predict the subsequent event. A Transformer-based model is proposed in Bai et al. (2021) which integrates deep event-level and script-level information for script event prediction. To deal with the data insufficiency problem, Zhou et al. (2021) proposes a multi-task self-supervised model for script event prediction. Compared to script event prediction, the event sequence prediction considered in this paper is more challenging because multiple events need to be predicted instead of one, and there is no candidate list of events like script event prediction.

4.2 Commonsense Knowledge

Commonsense knowledge is an important resource for artificial intelligence, and it is also helpful for many natural language processing tasks, such as reading comprehension, question answering, and text generation. For reading comprehension, Mihaylov and Frank (2018) introduces a neural reading comprehension model, which leverages a key-value memory to integrate external commonsense knowledge. In Yang et al. (2019), attention mechanism is employed to select knowledge from external knowledge bases, which is then fused with BERT to do knowledge-aware predictions. For question answering, Ma et al. (2021) proposes a neuro-symbolic framework for zero-shot question answering, to transform knowledge resources into an effective form for pretraining models. A model containing relevance scoring and joint reasoning is proposed in Yasunaga et al. (2021) to form a joint graph connecting the QA context and KG, whose representations are updated through a graph neural network. For text generation, Guan et al. (2020) proposes a knowledge-enhanced model based on multi-task learning for commonsense story generation. Li et al. (2022) proposes a two-stage method to explicitly arrange the ensuing events in open-ended text generation. In this paper, we design two pre-training tasks to implicitly inject external knowledge into the model to help the model generate event sequence that meets a given goal.

5 Conclusion

In this paper, we first reformulate the event sequence prediction task as an event sequence generation task, which can generate a wider variety of events. We then propose a three-stage model including augmentation, retrieval, and generation in order to leverage an external event knowledge base to generate event sequences. Finally, exper-

imental results show that our model outperforms existing methods, demonstrating the effectiveness of the proposed method.

Acknowledgements

This work is supported by the National Natural Science Foundation of China (No.U1936207, No.61976211, No.62176257). This work is supported by the Key Research Program of the Chinese Academy of Sciences (Grant NO.ZDBS-SSW-JSC006), the independent research project of National Laboratory of Pattern Recognition (No.Z-2018013) and the Youth Innovation Promotion Association CAS.

References

Long Bai, Saiping Guan, Jiafeng Guo, Zixuan Li, Xiaolong Jin, and Xueqi Cheng. 2021. Integrating deep event-level and script-level information for script event prediction. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 9869–9878.

Nathanael Chambers and Dan Jurafsky. 2008. Unsupervised learning of narrative event chains. In *Proceedings of ACL-08: HLT*, pages 789–797.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. Bert: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186.

Mark Granroth-Wilding and Stephen Clark. 2016. What happens next? event prediction using a compositional neural network model. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 30.

Jian Guan, Fei Huang, Zhihao Zhao, Xiaoyan Zhu, and Minlie Huang. 2020. A knowledge-enhanced pretraining model for commonsense story generation. *Transactions of the Association for Computational Linguistics*, 8:93–108.

Jena D Hwang, Chandra Bhagavatula, Ronan Le Bras, Jeff Da, Keisuke Sakaguchi, Antoine Bosselut, and Yejin Choi. 2021. On symbolic and neural commonsense knowledge graphs.

Qintong Li, Piji Li, Wei Bi, Zhaochun Ren, Yuxuan Lai, and Lingpeng Kong. 2022. Event transition planning for open-ended text generation. *arXiv preprint arXiv:2204.09453*.

Quanzhi Li and Qiong Zhang. 2021. Twitter event summarization by exploiting semantic terms and graph network. In *Proceedings of the AAAI conference on artificial intelligence*, volume 35, pages 15347–15354.

Sha Li, Heng Ji, and Jiawei Han. 2021. Document-level event argument extraction by conditional generation. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 894–908.

Zhongyang Li, Xiao Ding, and Ting Liu. 2018. Constructing narrative event evolutionary graph for script event prediction. In *Proceedings of the 27th International Joint Conference on Artificial Intelligence*, pages 4201–4207.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Roberta: A robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692*.

Ilya Loshchilov and Frank Hutter. 2018. Decoupled weight decay regularization. In *International Conference on Learning Representations*.

Shangwen Lv, Fuqing Zhu, and Songlin Hu. 2020. Integrating external event knowledge for script learning. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 306–315.

Kaixin Ma, Filip Ilievski, Jonathan Francis, Yonatan Bisk, Eric Nyberg, and Alessandro Oltramari. 2021. Knowledge-driven data construction for zero-shot evaluation in commonsense question answering. In *35th AAAI Conference on Artificial Intelligence*.

Todor Mihaylov and Anette Frank. 2018. Knowledgeable reader: Enhancing cloze-style reading comprehension with external commonsense knowledge. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 821–832.

Allen Nie, Erin Bennett, and Noah Goodman. 2019. Dissent: Learning sentence representations from explicit discourse relations. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 4497–4510.

Jeffrey Pennington, Richard Socher, and Christopher D Manning. 2014. Glove: Global vectors for word representation. In *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*, pages 1532–1543.

Karl Pichotta and Raymond Mooney. 2016. Learning statistical scripts with lstm recurrent neural networks. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 30.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J Liu. 2021. Exploring the limits of transfer learning with a unified text-to-text transformer (2019). *arXiv preprint arXiv:1910.10683*.

Ilya Sutskever, Oriol Vinyals, and Quoc V Le. 2014. Sequence to sequence learning with neural networks. *Advances in neural information processing systems*, 27.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Rémi Louf, Morgan Funtowicz, et al. 2020. Transformers: State-of-the-art natural language processing. In *Proceedings of the 2020 conference on empirical methods in natural language processing: system demonstrations*, pages 38–45.

An Yang, Quan Wang, Jing Liu, Kai Liu, Yajuan Lyu, Hua Wu, Qiaoqiao She, and Sujian Li. 2019. Enhancing pre-trained language representations with rich knowledge for machine reading comprehension. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 2346–2357, Florence, Italy. Association for Computational Linguistics.

Michihiro Yasunaga, Hongyu Ren, Antoine Bosselut, Percy Liang, and Jure Leskovec. 2021. Qagnn: Reasoning with language models and knowledge graphs for question answering. *arXiv preprint arXiv:2104.06378*.

Hongming Zhang, Muhao Chen, Haoyu Wang, Yangqiu Song, and Dan Roth. 2020. Analogous process structure induction for sub-event sequence prediction. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1541–1550.

Bo Zhou, Yubo Chen, Kang Liu, Jun Zhao, Jiexin Xu, Xiaojian Jiang, and Jinlong Li. 2021. Multi-task self-supervised learning for script event prediction. In *Proceedings of the 30th ACM International Conference on Information & Knowledge Management*, pages 3662–3666.

Generating Temporally-ordered Event Sequences via Event Optimal Transport

Bo Zhou[1,2], **Yubo Chen**[1,2], **Kang Liu**[1,2,4], **Jun Zhao**[1,2],
Jiexin Xu[3], **Xiaojian Jiang**[3], **Qiuxia Li**[3]
[1]School of Artificial Intelligence, University of Chinese Academy of Sciences
[2]National Laboratory of Pattern Recognition, CASIA
[3]China Merchants Bank, [4]Beijing Academy of Artificial Intelligence
{bo.zhou,yubo.chen,kliu,jzhao}@nlpr.ia.ac.cn
{jiexinx,jiangxiaojian,annielqx}@cmbchina.com

Abstract

Generating temporally-ordered event sequences in texts is important to natural language processing. Two emerging tasks in this direction are temporal event ordering (rearranging the set of events to correct order) and event infilling (generating an event at a specified position). To tackle the two related tasks, the existing method adopts a vanilla sequence-to-sequence model via maximum likelihood estimation (MLE). However, applying this approach to these tasks will cause two issues. One issue is that the MLE loss emphasizes strict local alignment and ignores the global semantics of the event. The other issue is that the model adopts a word-level objective to model events in texts, failing to evaluate the predicted results of the model from the perspective of event sequence. To alleviate these issues, we present a novel model to tackle the generation of temporally-ordered event sequences via Event Optimal Transport (EOT). First, we treat the events in the sequence as modeling units and explicitly extract the semantics of the events. Second, to provide event sequence-level evaluation of the predicted results of the model, we directly match events in sequences. Extensive experimental results show that our approach outperforms previous models on all evaluation datasets. In particular, the accuracy is improved by 7.7%, and the Macro F1 is improved by 7.2% on one of the datasets.

1 Introduction

Generating temporally-ordered event sequences in texts is crucial to many artificial intelligence applications, such as discourse understanding (Nie et al., 2019), dialog generation (Wu et al., 2018) and stock prediction (Ding et al., 2016). Temporal event ordering and event infilling are two challenging tasks in this line of work. The former refers to the rearrangement of an unordered sequence of events to an ordered sequence of events, and the latter refers to inferring missing events in an incomplete sequence of events. Figure 1 shows an example of the temporal event ordering task and event infilling task in (a) and (b), respectively. For the temporal event ordering, given an unordered sequence (e_3, e_2, e_4, e_1) as input, the output is ordered sequence (e_1, e_2, e_3, e_4). For the event infilling, given an incomplete sequence (e_1, e_2, e_4) as input, the output is complete sequence (e_1, e_2, e_3, e_4) with the infilled event e_3.

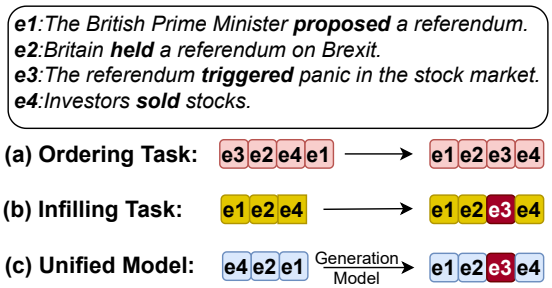

Figure 1: The temporal event ordering task, event infilling task and a single generation model handling these two tasks.

To handle these two related tasks, the currently existing method (Lin et al., 2021) employs a unified generation model which is shown in Figure 1 (c). Their model takes incomplete unordered events as input sequence and outputs a complete ordered event sequence, based on the sequence-to-sequence (Seq2Seq) model via maximum likelihood estimation (MLE), which maximizes the likelihood of the next word conditioned on its previous ground-truth words. Such an approach adopts cross-entropy loss as the objective, essentially measuring the word difference at each position of the target sequence and providing a word-level training loss. Although their model has shown good performance in handling the two tasks, there are still two issues.

First, the MLE loss emphasizes strict local alignment and ignores the global semantics of the event. For example in Figure 2, the MLE loss will give the

Proceedings of the 29th International Conference on Computational Linguistics, pages 1875–1884
October 12–17, 2022.

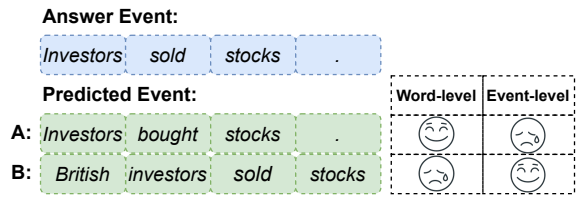

Figure 2: Comparison between word-level score and event-level score on two predicted events.

predicted event *A* a high score and *B* a low score because all words except one in event *A* are aligned with the words in the answer event yet no word event *B* is aligned with the words in the answer event. However, the loss is expected to give event *A* a low score and event *B* a high score because from the perspective of event semantics, event *A* is completely different from the answer event and event *B* is the same as the answer event. Therefore, we should consider the overall semantics of the event, rather than the strict alignment of words within the event.

Second, the goal of the model is to infer events in texts, but the sequence-to-sequence model uses a word-level objective so it fails to evaluate the result from the perspective of event sequence. For example in Figure 1, the correct predicted event sequence is (e_1, e_2, e_3, e_4). If the model predicts an event sequence (e_4, e_1, e_2, e_3), the MLE loss will give it a low score because no event is predicted correctly compared to the answer event sequence. However, the sequence (e_4, e_1, e_2, e_3) should not get a low score, because at least the orders of 3 events are predicted correctly. Therefore, how to evaluate the predicted results of the model from the perspective of event sequence is a challenging problem.

To tackle the above issues, we introduce a novel method for the generation of temporally-ordered event sequences via Event Optimal Transport (EOT). Specifically, for the first issue, we treat the events in the sequence as modeling units and explicitly extract the semantics of the events. For the second issue, we propose to use optimal transport to directly match events in sequences. The EOT allows end-to-end supervised training and acts as an effective sequence-level regularization to the MLE loss.

In summary, our contributions can be summarized as follows:

- We introduce a novel method for the generation of temporally-ordered event sequences via Event Optimal Transport (EOT), which treats the events in the sequence as modeling units and explicitly extracts the semantics of the events.
- We directly match events in sequences to provide event sequence-level evaluation of the predicted results of the model.
- Extensive experimental results demonstrate the superiority of the proposed method on all evaluation datasets. Specifically, the accuracy is improved by 7.7%, and the Macro F1 is improved by 7.2% on one of the datasets.

2 Related Work

Script Event Prediction Given context event sequence, script event prediction aims at predicting the subsequent event from a candidate list. The task is first proposed by Chambers and Jurafsky (2008), and a statistical model is proposed to learn the cooccurrence between events. Jans et al. (2012) leverages a bigram model to model the temporal order between events explicitly. The above two methods are count-based, and then researchers have proposed methods based on neural networks. Granroth-Wilding and Clark (2016) proposes a neural network based model for simultaneously learning word embedding and composition function. In Wang et al. (2017), they propose an LSTM based model to integrate order information and event relation. Li et al. (2018) treats event chain as a subgraph and leverages recurrent networks to better model relatedness between events in the candidate list with events in the graph. Lv et al. (2019) proposes a model that integrates event-level and chain-level attentions to better leverage information contained in event chain. Zhou et al. (2021) proposes a multi-task self-supervised model to cope with the problem of lack of training data in script event prediction. To incorporate event circumstances into the narrative event prediction, Wang et al. (2021) adopts the two multi-head attention to retrieve circumstances at the local and global levels. In this paper, we consider two related tasks of temporal event ordering and event infilling and leverage a single generation model to tackle these two tasks.

Optimal Transport in NLP Optimal transport has been applied in NLP for a variety of tasks recently. In Kusner et al. (2015), the author proposes the Word Mover's Distance (WMD), a

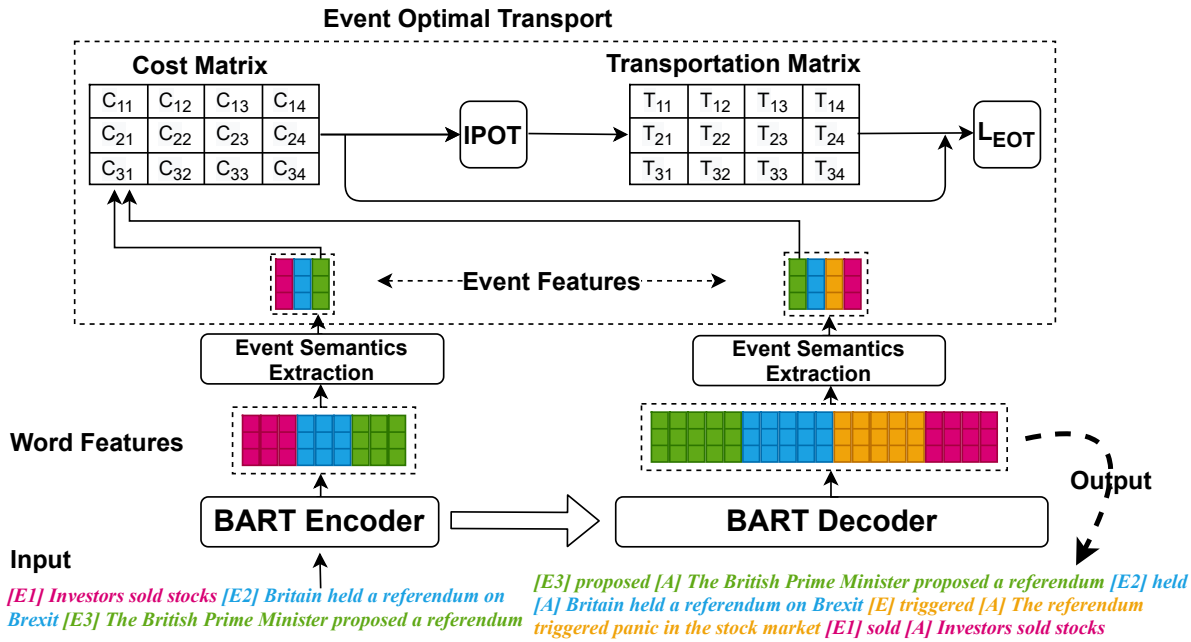

Figure 3: Overall architecture of the proposed generation model based on event optimal transport.

novel distance function between text documents that measures the dissimilarity between two text documents as the minimum amount of distance that the embedded words of one document need to "travel" to reach the embedded words of another document. Later in Xu et al. (2018), a novel Wasserstein method with a distillation mechanism is proposed, yielding joint learning of word embeddings and topics. The cross-lingual correspondence problem is cast as an optimal transport problem in Alvarez-Melis and Jaakkola (2018), and the Gromov-Wasserstein distance is exploited for the alignment of word embedding spaces. A content-aware sparse attention module based on optimal transport is proposed in Chen et al. (2020) to deal with textual network embedding problem. Li et al. (2020) proposes student-forcing optimal transport to tackle the exposure bias problem in text-generation models trained by maximum likelihood estimation. Xu et al. (2021) formulates the quest of vocabularization – finding the best token dictionary with a proper size – as an optimal transport problem and proposes a simple and efficient solution without trial training. A time-aware optimal transport distance is introduced (Li et al., 2021) for learning the model to compress the event-graphs of news articles in an unsupervised manner. In this paper, we leverage optimal transport and propose event OT to directly match events in event sequences to provide sequence-level supervision of the predicted results.

3 Methodology

The overall structure of our model is illustrated in Figure 3. The input of the model is a sequence of events $\boldsymbol{x} = (e_1, ..., e_n)$, which is incomplete and unordered, and the output of the model is a complete and ordered sequence of events $\boldsymbol{y} = (e_1, ..., e_m)$. We represent e in the event sequence as the concatenation of its predicate with all its arguments.

3.1 Generation Model

To leverage the power of pretrained transformers, we base the underlying architecture for our model on BART (Lewis et al., 2020). BART is a denoising autoencoder for pretraining sequence-to-sequence models which can be seen as generalizing BERT (Kenton and Toutanova, 2019) (due to the bidirectional encoder), GPT (Radford and Narasimhan, 2018) (with the left-to-right decoder), and other recent pretraining schemes.

Similar to previous work (Lin et al., 2021), we then prepend a special word $[Ei]$ in front of each event in input event sequence $\boldsymbol{x}$. For the output, if e_j in $\boldsymbol{y}$ is one of the input events e_i in $\boldsymbol{x}$, then we prepend special words $[Ei]v_{e_j}[A]$ before e_j, where v_{e_j} is the predicate of event e_j. Otherwise, the special words $[E]v_{e_j}[A]$ are used in front of e_j. An example of the above is shown in Figure 3. The

use of $[Ei]$ helps the model differentiate between events in the input and output sequences, which facilitates optimal transport between events.

3.2 Event Optimal Transport

We first briefly introduce the optimal transport. Given two spaces X and Y, the Kantorovich formulation of optimal transport aims to find a probability measure γ on $X \times Y$ that attains the following infimum:

$$\inf \left\{ \int_{X \times Y} c(x, y) \mathrm{d}\gamma(x, y) \mid \gamma \in \Gamma(\mu, \nu) \right\} \quad (1)$$

where $\Gamma(\mu, \nu)$ is the set of probability measures on $X \times Y$ with marginal μ on X and ν on Y.

Now given two discrete probability measures μ and ν, which can be represented by Dirac measure as below:

$$\mu = \sum_{i=1}^{n} u_i \delta_{\boldsymbol{x}_i}, \quad \nu = \sum_{j=1}^{m} v_i \delta_{\boldsymbol{y}_j} \quad (2)$$

where $\delta_{\boldsymbol{x}}$ is the Dirac measure sitting at $\boldsymbol{x}$. The coefficients $\boldsymbol{u} = \{u_i\}_{i=1}^{n}$ and $\boldsymbol{v} = \{v_j\}_{j=1}^{m}$ satisfy the constraints $\sum_{i=1}^{n} u_i = \sum_{j=1}^{m} v_i = 1$ as both μ and ν are probability measures. Under such a setting, the objective function and the constraint in the primal Kantorovich problem are then:

$$\begin{aligned} \min_{\boldsymbol{T}} \quad & \sum_{i,j} \boldsymbol{T}_{ij} \cdot c\left(\boldsymbol{x}_i, \boldsymbol{y}_j\right) = \min_{\boldsymbol{T}} \quad \langle \boldsymbol{T}, \boldsymbol{C} \rangle \\ s.t. \quad & \sum_{j} \boldsymbol{T}_{ij} = \boldsymbol{u}_i, \quad i = 1, ..., n, \\ & \sum_{i} \boldsymbol{T}_{ij} = \boldsymbol{v}_j, \quad j = 1, ..., m, \\ & \boldsymbol{T} \in \mathbb{R}_{+}^{n \times m}. \end{aligned} \quad (3)$$

where $\boldsymbol{C}$ is the cost matrix with $\boldsymbol{C}_{ij} = c\left(\boldsymbol{x}_i, \boldsymbol{y}_j\right)$ and c is the cost function. $\langle \boldsymbol{T}, \boldsymbol{C} \rangle = \mathrm{Tr}(\boldsymbol{T}^{\top} \boldsymbol{C})$ denotes the Frobenius dot-product, here Tr denotes the trace of a matrix.

Because the exact solution of (3) is is highly intractable (Arjovsky et al., 2017), researchers have proposed some approximate algorithms, such as the Sinkhorn (Cuturi, 2013) and IPOT (Xie et al., 2020) algorithms. Here we choose the IPOT algorithm to approximate the minimizer of (3). The details of the IPOT algorithm are shown in Algorithm 1.

Assume the output (the last hidden state) of the BART encoder and decoder are $\boldsymbol{H}_e \in \mathbb{R}^{L_e \times d}$ and $\boldsymbol{H}_d \in \mathbb{R}^{L_d \times d}$ respectively, where L_e and L_d are lengths of input and output sequences. We first partition $\boldsymbol{H}_e$ into n groups where each group corresponds to features of words in an event (note that the input to our model is $\boldsymbol{x} = \{e_1, ..., e_n\}$), then features of words in a group are averaged to obtain a vector sequence $S = \{\boldsymbol{h}_i\}_{i=1}^{n}$. Note that besides average, we also tried other aggregation functions, such as max pooling, attention, etc., but they all performed worse than average. The possible reason is that there are already more complex aggregation functions such as attention in BART, so it is enough to use average as the aggregation function of the BART output.

For $\boldsymbol{H}_d$, it's first input to a linear layer:

$$\boldsymbol{H}_l = \boldsymbol{H}_d \boldsymbol{W} + \boldsymbol{b} \quad (4)$$

where $\boldsymbol{W} \in \mathbb{R}^{d \times V}$ and V is the vocabulary size. We then use the $argmax$ function over $\boldsymbol{H}_l$ to obtain the predicted indices $\boldsymbol{i} \in \mathbb{Z}_{+}^{L_d}$. We identify indices of special word $[E_i]$ in $\boldsymbol{i}$ and these identified indices are used to partition $\boldsymbol{H}_d$ into k groups where each group corresponds to features of words in an event. Note that there may be cases where there are no identified indices, which are most likely to occur at the beginning of training. To solve possible errors, we partition $\boldsymbol{H}_d$ into L_d groups when this happens. Finally after averaging, we obtain another vector sequence $S' = \{\boldsymbol{h}_j\}_{j=1}^{k}$.

Algorithm 1 IPOT algorithm

Input: Feature vectors $S = \{\boldsymbol{x}_i\}_1^n, S' = \{\boldsymbol{x}'_i\}_1^m$
Parameter: Generalized stepsize $\frac{1}{\beta}$
Output: $\langle \boldsymbol{T}, \boldsymbol{C} \rangle$

$\boldsymbol{\sigma} = \frac{1}{m}\mathbf{1}_{\boldsymbol{m}}, \boldsymbol{T}^{(1)} = \mathbf{1}_{\boldsymbol{n}}\mathbf{1}_{\boldsymbol{m}}{}^{\top}$.
$\boldsymbol{C}_{ij} = c(\boldsymbol{x}_i, \boldsymbol{x}'_j), \boldsymbol{A}_{ij} = e^{-\frac{\boldsymbol{C}_{ij}}{\beta}}$.
for $t = 1, 2, ..., T$ **do**
  $\boldsymbol{Q} = \boldsymbol{A} \odot \boldsymbol{T}^{(t)}$. // $\odot$ is Hadamart product
  for $k = 1, 2, ..., K$ **do**
    $\boldsymbol{\delta} = \frac{1}{n\boldsymbol{Q}\boldsymbol{\sigma}}, \boldsymbol{\sigma} = \frac{1}{m\boldsymbol{Q}^{\top}\boldsymbol{\delta}}$.
  end for
  $\boldsymbol{T}^{(t+1)} = diag(\delta)\boldsymbol{Q}diag(\sigma)$.
end for
return $\langle \boldsymbol{T}, \boldsymbol{C} \rangle$

3.3 Training and Optimization

In order to use a unified generation model to handle temporal event ordering and event infilling tasks, we first construct input-output data pairs related to these two tasks. Specifically, given an ordered

Algorithm 2 EOT algorithm

Input: Ground truth $\{(\boldsymbol{x}_1, \boldsymbol{y}_1), ..., (\boldsymbol{x}_N, \boldsymbol{y}_N)\}$
Parameter: Batch size M

Initialize MLE model parameters.
for $epoch = 1, ..., MaxEpoch$ **do**
  for $k = 1, ..., M$ **do**
    Draw a pair of sequences $(\boldsymbol{x}_i, \boldsymbol{y}_i)$.
    Compute the outputs $\boldsymbol{H}_e$ and $\boldsymbol{H}_d$ of the encoder and decoder.
    Extract feature vector sequences of events S and S' from $\boldsymbol{H}_e$ and $\boldsymbol{H}_d$.
    Compute the cost matrix $\boldsymbol{C}$ based on S and S' via cosine distance.
    Compute the EOT loss defined in (4).
  end for
  Update model parameters by optimizing loss in (6).
end for

event sequence $\boldsymbol{y}$ as defined above, we follow previous work (Lewis et al., 2020; Lin et al., 2021) to corrupt it to obtain the required input $\boldsymbol{x}$ by two steps. The first step is to shuffle events: performing a random shuffling of the complete ordered event sequence $\boldsymbol{y}$ to produce an unordered event sequence $\boldsymbol{x}'$. The second step is to delete events: randomly deleting each event in $\boldsymbol{x}'$ with probability p to produce the incomplete unordered input event sequence $\boldsymbol{x}$.

Take Figure 1 (c) for example, the complete ordered event sequence on the right is $\boldsymbol{y}$. After performing event shuffling and event deletion, we obtain incomplete unordered input event sequence $\boldsymbol{x}$ on the left.

Now we introduce the optimization process of the model. After obtaining two vector sequences $S = \{\boldsymbol{h}_i\}_{i=1}^{n}$ and $S' = \{\boldsymbol{h}_j\}_{j=1}^{k}$ as described in the previous section, we can compute the event-level OT loss using the IPOT algorithm described above:

$$\mathcal{L}_{EOT} = \text{IPOT}(S, S') \tag{5}$$

Meanwhile, suppose the input sequence is $\boldsymbol{x} = (w_1, ..., w_L)$ and the gold output sequence is $\boldsymbol{y} = (w_1, ..., w_{L'})$, where L and L' are numbers of words. We also have the following maximum likelihood estimation (MLE) loss:

$$\mathcal{L}_{MLE} = \sum_{t=1}^{L'} \log p_\theta(w_t | w_{<t}, \boldsymbol{x}) \tag{6}$$

We then combine the two loss functions to obtain the following loss:

$$\mathcal{L} = \frac{1}{M} \sum_{n=1}^{M} (-\mathcal{L}_{MLE}(\boldsymbol{x}_n, \boldsymbol{y}_n) + \alpha \mathcal{L}_{EOT}(S_n, S'_n)) \tag{7}$$

where M is the number of training pairs. The parameters are updated by minimizing this loss and the full algorithm is summarized in Algorithm 2.

4 Experiment

4.1 Experimental Setup

Dataset Following Lin et al. (2021), the temporal event sequences are extracted from the EventsNarratives corpus (Yao and Huang, 2018). The SRL model from AllenNLP (Gardner et al., 2018) is used to extract verbs (events) and their arguments. Then, only events in different sentences are connected to construct temporal event sequences, and only event chains associated with a common entity are included. We train our model on 100,000 sequences extracted by the above procedure. Two different orders are used to scramble each sequence, resulting in a total of 200,000 training data.

For testing, two out-of-domain English datasets CaTeRS (Mostafazadeh et al., 2016) and MCTaco (Zhou et al., 2019) are used to extract the test temporal event sequences. CaTeRS includes annotations of events and their causal and temporal relations on short stories. MCTaco is a question answering dataset for evaluating the model's capability of understanding temporal commonsense. Following Lin *et al.* (2021), 842 event sequences are extracted for CaTeRS and after applying two different permutations to each sequence, 1684 CaTeRS examples are finally obtained. For MCTaco, 585 test sequences are extracted.

Training Details We choose the cosine distance as the cost function in the event optimal transport and the hyper-parameter α is set to 0.1. The learning rate of our model is 1e-5, and a polynomial decay scheduling with 500 steps of warm-up is used. We set the batch size to 64, the models are trained for 10 epochs, with 2000 updates each epoch. We set the event deletion probability to 0.15 for the deletion training strategy. The BART-large pretrained model from Hugging-Face's Transformers library (Wolf et al., 2020) is used as the underlying structure which is the same as previous work. Beam search with the beam size 4 is used when decoding the output event sequences during the evaluation for temporal event ordering.

	CaTeRS		MCTaco	
Model	All Pair Acc.	Longer Pair Acc.	Acc.	Macro F1
BERT-based SSVM	65.7	62.3	67.2	47.0
Pointer Network	54.1	52.3	54.7	42.7
TemporalBART†	77.1	74.7	63.9	50.1
TemporalBART-indexed†	79.7	78.0	74.9	55.1
EOT	**81.3**	**80.2**	**82.6**	**62.3**

Table 1: Temporal ordering results on dataset CaTeRS and MCTaco. For CaTeRS, All Pair Acc. means pairwise accuracy of predicted ordering for all event sequences, while Longer Pair Acc. denotes pairwise accuracy for sequences containing more than three events. For MCTaco, Accuracy and Macro F1 score are computed on the ordering between the question event and answer event. The symbol † represents the results we reproduced using public code released by Lin et al. (2021), and other two results are taken from Lin et al. (2021).

Baselines We compare our model with the following baseline methods for temporal event ordering:

- **BERT-based Pairwise Model + SSVM** (Han et al., 2019) leverages a BERT-based model (Kenton and Toutanova, 2019) to compute pairwise scores for two events in the output, and the final output is then obtained by solving an ILP over all the pairwise scores.
- **BERT-based Pointer Network** first uses BERT to extract representations for events that are fed into an LSTM-based pointer network to compute the probability for ordering.
- **TemporalBART** (Lin et al., 2021) is based on BART (Lewis et al., 2020), and special words are prepended in front of events in input and output sequences to provide extra clues.
- **TemporalBART-indexed** (Lin et al., 2021) is the same as TemporalBART except that the indices of the special words prepended before events are considered.

For event infilling, we compare our model with these extra baselines:

- **HAQAE** (Weber et al., 2018) is a vector quantized variational autoencoder with a latent space defined by a hierarchy of categorical variables which encodes schema knowledge.
- **GPT-2** (Radford et al., 2018) is a transformer-based pretrained language model which is used as the underlying structure in many generation tasks.
- **Infilling GPT-2** (Qin et al., 2020) generates the infilling events conditioned on both the prefix events and the events after the insertion position.

4.2 Results on Temporal Event Ordering

CaTeRS Dataset Experimental results on CaTeRS are shown in Table 1, pairwise accuracy is used to calculate the proportion of ordered event pairs in the predicted sequence.

Among all the approaches, our method performs best. It achieves the best performance on both all sequences and long sequences. The reasons may be that compared with the BERT-based pointer network, our model can condition the word-level embeddings of the events when generating the output events instead of condensed event embeddings. Compared with BART-based models, our model treats the events in the sequence as modeling units and explicitly extracts the semantics of the events instead of emphasizing strict local alignment of words.

MCTaco Dataset The accuracy on predicting the temporal relation of event in question and event in answer is computed, since only gold temporal relation of event in question and event in answer is known for each test sequence. The macro F1 score is also computed because the proportion of before/after questions is unbalanced in MCTaco.

Our EOT model outperforms all the baselines, in particular, our model outperforms TemporalBART-indexed by 7.7% in accuracy and 7.2% in Macro F1. We attribute the significant improvement to the direct events matching in sequences by our model. We will further demonstrate the effectiveness of direct events matching in the following experiments.

Ordering Unseen Events for CaTeRS Following Lin et al. (2021), we also evaluate our EOT model on an additional variant task of temporal

Model	All seq		Longer seq	
	EM	Top2 EM	EM	Top2 EM
Random	34.1	69.5	23.7	48.7
HAQAE	37.1	71.9	28.7	53.2
GPT-2	35.2	68.4	22.6	48.2
Infilling GPT-2	38.8	73.5	26.3	55.4
TemporalBART†	57.7	83.3	48.2	70.6
TemporalBART-indexed†	58.4	87.4	50.9	77.4
EOT	**58.8**	**88.2**	**52.6**	**78.1**

Table 2: The results of ordering unseen events on sequences from dataset CaTeRS. Longer seq means results for sequences containing more than three events. The symbol † has same meaning as Table 1.

event ordering which better tests its capability as a generative model. Specifically, for each event sequence in CaTeRS, we first randomly delete an event e^* in the sequence, and let the remaining events be denoted as $(e_1, ..., e_N)$. We want to test whether the model can insert the deleted event e^* to its original position, thus we use the generation probability of the model to rank the $N + 1$ sequences which are obtained by inserting e^* to $N + 1$ different positions. The higher the model ranks the original sequence, the better the model is able to capture the relationship between seen events and possibly generated unseen events.

Table 2 shows the results, and the top-1 and top-2 exact match (EM) are used to evaluate the results, which calculate the proportion of gold sequences that the model ranks first and second above. Our model outperforms all the baselines on both all the event sequences and long sequences, which again demonstrates the effectiveness of considering event-level semantics and direct events matching in sequences. Another observation is that compared with the TenporalBART-indexed model, the performance of our EOT model on longer sequences is more significant than on all the sequences. One possible reason is that longer sequences have more events which are more helpful for the model to do the event-level matching.

4.3 Results on Event Infilling

Now we consider temporal event infilling and the dataset CaTeRS is used. Given an event sequence from CaTeRS, we first randomly delete an event to obtain $(e_1, ..., e_{i-1}, e_{i+1}, ..., e_n)$. Then the model should generate an infilled event e^* at position i and the new sequence $(e_1, ..., e_{i-1}, e_i, e_{i+1}, ..., e_n)$ is expected to be temporally ordered.

We measure the quality of generated events through human evaluation. Given a sequence with a generated event at some position, 3 raters are asked to score this sequence in terms of the coherence (How coherent the generated event is with the context?) and temporality (Does the generated event occur in the right order concerning the context?) for the generated event and both scores are from $\{0, 1, 2\}$. The final scores are the majority scores of the 3 raters for both coherence and temporality. We then randomly sample 15 sequences from CaTeRS and the averaged scores are took as the metric.

The result is shown in Table 3, and our EOT model achieves better performance than all the baseline models in terms of coherence and temporality. The reason may be that by explicitly matching events in input and output sequence, the model can generate an event that is more relevant to the scenario of events in the input sequence.

Model	Coherence	Temporality
GPT-2	1.27	0.60
Infilling GPT-2	1.53	0.80
TemporalBART	1.13	0.87
TemporalBART-indexed	1.53	1.07
EOT	**1.67**	**1.13**

Table 3: The human evaluation result for event infilling on dataset CaTeRS.

We show two examples of events generated by different models in Figure 4. As we can see in Figure 4 (a), the event generated by infilling GPT-2 is less relevant to the context events. The order of the event generated by TemporalBART-indexed is inappropriate, although it's coherent with the scenario of the input events. The event generated by our model EOT is both coherent and temporally ordered. Another example is shown in 4 (b).

4.4 Further Demonstration of EOT's Effectiveness

To further demonstrate the effectiveness of our event optimal transport, we compare EOT with two extra models. Recall that in our model, the output (the last hidden state) of the BART encoder and decoder are $\boldsymbol{H}_e \in \mathbb{R}^{L_e \times d}$ and $\boldsymbol{H}_d \in \mathbb{R}^{L_d \times d}$ respectively, then two vector sequences $S = \{\boldsymbol{h}_i\}_{i=1}^n$ and $S' = \{\boldsymbol{h}_j\}_{j=1}^k$ corresponding to events in input and output are extracted to construct event OT loss. Now, we directly treat vectors in $\boldsymbol{H}_e$ and $\boldsymbol{H}_d$ as S and S' respectively to construct word-level loss, and this model is called Word Optimal Transport (WOT). We also test the model which com-

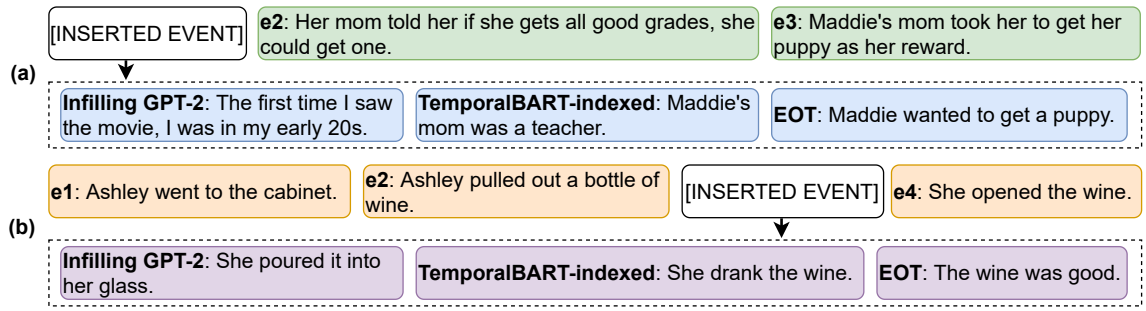

Figure 4: Two examples of events generated by infilling GPT-2, TemporalBART-indexed and our model EOT. The input events are in green and orange, while events generated by the models are in blue and purple.

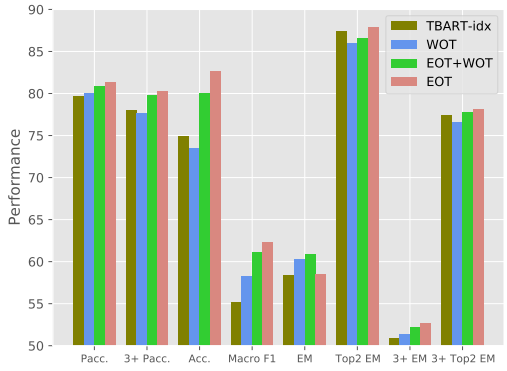

Figure 5: The performances of the 4 models on the 8 evaluation metrics which correspond to Table 1 and 2. The TBART-idx represents the TemporalBART-indexed model. Pacc means Pairwise Accuracy and 3+ means sequences with 3 or more events.

bines event and word-level OT loss and is called EOT+WOT.

Experimental results are shown in Figure 5, from which we can make the following observations:

(1) The WOT model achieves comparable results with the TemporalBART-indexed model and performs better on some metrics. This shows that imposing global sequence-level guidance via new supervision is effective, although the effect is not obvious, because it did not take into account the event-level matching.

(2) When event-level matching is considered, the EOT model achieves performance improvements on almost all metrics compared to the WOT model, and the EOT model outperforms the TemporalBART-indexed model on all metrics. This demonstrates that incorporating event-level matching can further improve the performance of optimal transport and outperform the baseline models.

(3) When we combine word-level with event-level OT, the performance is expected to get better. Unfortunately, the performance of EOT+WOT decreases compared with EOT. One possible reason is that word-level matching and event-level matching conflict with each other to some extent: two events may match well, but the words in the two events may not match each other.

5 Conclusion

In this paper, we consider a single generation model which can support inferences in the two related tasks. We introduce a novel method for the generation of temporally-ordered event sequences via Event Optimal Transport (EOT). Compared with the MLE-based Seq2Seq model, our approach has two advantages: (i) we treat the events in the sequence as modeling units and explicitly extract the semantics of the events; (ii) we directly match events in sequences to provide event sequence-level evaluation of the predicted results of the model. Experimental results show the superiority of our model on all evaluation datasets. Specifically, the accuracy is improved by 7.7%, and the Macro F1 is improved by 7.2% on one of the datasets.

Acknowledgements

This work is supported by the National Key Research and Development Program of China (No.2020AAA0106400), the National Natural Science Foundation of China (No.62176257, 61976211, 61922085). This work is also supported by the Strategic Priority Research Program of Chinese Academy of Sciences (Grant No.XDA27020200), the Youth Innovation Promotion Association CAS, and Yunnan Provincial Major Science and Technology Special Plan Projects (No.202202AD080004).

References

David Alvarez-Melis and Tommi Jaakkola. 2018. Gromov-wasserstein alignment of word embedding spaces. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 1881–1890.

Martin Arjovsky, Soumith Chintala, and Léon Bottou. 2017. Wasserstein generative adversarial networks. In *International conference on machine learning*, pages 214–223. PMLR.

Nathanael Chambers and Dan Jurafsky. 2008. Unsupervised learning of narrative event chains. In *Proceedings of ACL-08: HLT*, pages 789–797.

Liqun Chen, Guoyin Wang, Chenyang Tao, Dinghan Shen, Pengyu Cheng, Xinyuan Zhang, Wenlin Wang, Yizhe Zhang, and Lawrence Carin. 2020. Improving textual network embedding with global attention via optimal transport. In *ACL 2019-57th Annual Meeting of the Association for Computational Linguistics, Proceedings of the Conference*, pages 5193–5202. Association for Computational Linguistics (ACL).

Marco Cuturi. 2013. Sinkhorn distances: Lightspeed computation of optimal transport. *Advances in neural information processing systems*, 26.

Xiao Ding, Yue Zhang, Ting Liu, and Junwen Duan. 2016. Knowledge-driven event embedding for stock prediction. In *Proceedings of coling 2016, the 26th international conference on computational linguistics: Technical papers*, pages 2133–2142.

Matt Gardner, Joel Grus, Mark Neumann, Oyvind Tafjord, Pradeep Dasigi, Nelson F Liu, Matthew Peters, Michael Schmitz, and Luke Zettlemoyer. 2018. Allennlp: A deep semantic natural language processing platform. In *Proceedings of Workshop for NLP Open Source Software (NLP-OSS)*, pages 1–6.

Mark Granroth-Wilding and Stephen Clark. 2016. What happens next? event prediction using a compositional neural network model. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 30.

Rujun Han, I-Hung Hsu, Mu Yang, Aram Galstyan, Ralph Weischedel, and Nanyun Peng. 2019. Deep structured neural network for event temporal relation extraction. In *Proceedings of the 23rd Conference on Computational Natural Language Learning (CoNLL)*, pages 666–106.

Bram Jans, Steven Bethard, Ivan Vulić, and Marie Francine Moens. 2012. Skip n-grams and ranking functions for predicting script events. In *Proceedings of the 13th Conference of the European Chapter of the Association for Computational Linguistics*, pages 336–344.

Jacob Devlin Ming-Wei Chang Kenton and Lee Kristina Toutanova. 2019. Bert: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of NAACL-HLT*, pages 4171–4186.

Matt Kusner, Yu Sun, Nicholas Kolkin, and Kilian Weinberger. 2015. From word embeddings to document distances. In *International conference on machine learning*, pages 957–966. PMLR.

Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Veselin Stoyanov, and Luke Zettlemoyer. 2020. Bart: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7871–7880.

Jianqiao Li, Chunyuan Li, Guoyin Wang, Hao Fu, Yuhchen Lin, Liqun Chen, Yizhe Zhang, Chenyang Tao, Ruiyi Zhang, Wenlin Wang, et al. 2020. Improving text generation with student-forcing optimal transport. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 9144–9156.

Manling Li, Tengfei Ma, Mo Yu, Lingfei Wu, Tian Gao, Heng Ji, and Kathleen McKeown. 2021. Timeline summarization based on event graph compression via time-aware optimal transport. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 6443–6456.

Zhongyang Li, Xiao Ding, and Ting Liu. 2018. Constructing narrative event evolutionary graph for script event prediction. In *Proceedings of the 27th International Joint Conference on Artificial Intelligence*, pages 4201–4207.

Shih-Ting Lin, Nathanael Chambers, and Greg Durrett. 2021. Conditional generation of temporally-ordered event sequences. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 7142–7157, Online. Association for Computational Linguistics.

Shangwen Lv, Wanhui Qian, Longtao Huang, Jizhong Han, and Songlin Hu. 2019. Sam-net: Integrating event-level and chain-level attentions to predict what happens next. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, pages 6802–6809.

Nasrin Mostafazadeh, Alyson Grealish, Nathanael Chambers, James Allen, and Lucy Vanderwende. 2016. Caters: Causal and temporal relation scheme for semantic annotation of event structures. In *Proceedings of the Fourth Workshop on Events*, pages 51–61.

Allen Nie, Erin Bennett, and Noah Goodman. 2019. Dissent: Learning sentence representations from explicit discourse relations. In *Proceedings of the*

57th Annual Meeting of the Association for Computational Linguistics, pages 4497–4510.

Lianhui Qin, Vered Shwartz, Peter West, Chandra Bhagavatula, Jena Hwang, Ronan Bras, Antoine Bosselut, and Choi Yejin. 2020. Back to the future: Unsupervised backprop-based decoding for counterfactual and abductive commonsense reasoning. pages 794–805.

Alec Radford and Karthik Narasimhan. 2018. Improving language understanding by generative pre-training.

Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. 2018. Language models are unsupervised multitask learners.

Shichao Wang, Xiangrui Cai, Hongbin Wang, and Xiaojie Yuan. 2021. Incorporating circumstances into narrative event prediction. In *Findings of the Association for Computational Linguistics: EMNLP 2021*, pages 4840–4849.

Zhongqing Wang, Yue Zhang, and Ching Yun Chang. 2017. Integrating order information and event relation for script event prediction. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 57–67.

Noah Weber, Leena Shekhar, Niranjan Balasubramanian, and Nathanael Chambers. 2018. Hierarchical quantized representations for script generation. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 3783–3792.

Thomas Wolf, Julien Chaumond, Lysandre Debut, Victor Sanh, Clement Delangue, Anthony Moi, Pierric Cistac, Morgan Funtowicz, Joe Davison, Sam Shleifer, et al. 2020. Transformers: State-of-the-art natural language processing. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 38–45.

Xianchao Wu, Ander Martinez, and Momo Klyen. 2018. Dialog generation using multi-turn reasoning neural networks. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 2049–2059.

Yujia Xie, Xiangfeng Wang, Ruijia Wang, and Hongyuan Zha. 2020. A fast proximal point method for computing exact wasserstein distance. In *Uncertainty in Artificial Intelligence*, pages 433–453. PMLR.

Hongteng Xu, Wenlin Wang, Wei Liu, and Lawrence Carin. 2018. Distilled wasserstein learning for word embedding and topic modeling. In *Proceedings of the 32nd International Conference on Neural Information Processing Systems*, pages 1723–1732.

Jingjing Xu, Hao Zhou, Chun Gan, Zaixiang Zheng, and Lei Li. 2021. Vocabulary learning via optimal transport for neural machine translation. In *Proceedings of ACL 2021*.

Wenlin Yao and Ruihong Huang. 2018. Temporal event knowledge acquisition via identifying narratives. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 537–547.

Ben Zhou, Daniel Khashabi, Qiang Ning, and Dan Roth. 2019. "going on a vacation" takes longer than "going for a walk": A study of temporal commonsense understanding. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 3363–3369.

Bo Zhou, Yubo Chen, Kang Liu, Jun Zhao, Jiexin Xu, Xiaojian Jiang, and Jinlong Li. 2021. Multi-task self-supervised learning for script event prediction. In *Proceedings of the 30th ACM International Conference on Information & Knowledge Management*, pages 3662–3666.

Improving Continual Relation Extraction through Prototypical Contrastive Learning *

Chengwei Hu[1†], **Deqing Yang**[1†✉], **Haoliang Jin**[1‡], **Zhen Chen**[1‡], **Yanghua Xiao**[2†]
[1]School of Data Science, Fudan University, Shanghai, China
[2]School of Computer Science, Fudan University, Shanghai, China
[†]{cwhu20, yangdeqing, shawyh}@fudan.edu.cn
[‡]{hljin21, zhenchen21}@m.fudan.edu.cn

Abstract

Continual relation extraction (CRE) aims to extract relations towards the continuous and iterative arrival of new data, of which the major challenge is the *catastrophic forgetting* of old tasks. In order to alleviate this critical problem for enhanced CRE performance, we propose a novel **C**ontinual **R**elation **E**xtraction framework with **C**ontrastive **L**earning, namely **CRECL**, which is built with a classification network and a prototypical contrastive network to achieve the incremental-class learning of CRE. Specifically, in the contrastive network a given instance is contrasted with the prototype of each candidate relations stored in the memory module. Such contrastive learning scheme ensures the data distributions of all tasks more distinguishable, so as to alleviate the catastrophic forgetting further. Our experiment results not only demonstrate our CRECL's advantage over the state-of-the-art baselines on two public datasets, but also verify the effectiveness of CRECL's contrastive learning on improving CRE performance.

1 Introduction

In some scenarios of relation extraction (RE), massive new data including new relations emerges continuously, which can not be solved by traditional RE methods. To handle such situation, *continual relation extraction* (CRE) (Wang et al., 2019) was proposed. Due to the limited storage and computing resources, it is impractical to store all training data of previous tasks. As new tasks are learned where new relations emerge constantly, the model tends to forget the existing knowledge about old relations. Therefore, the problem of *catastrophic forgetting* damages CRE performance severely (Hassabis et al., 2017; Thrun and Mitchell, 1995).

In recent years, some efforts have focused on the alleviating catastrophic forgetting in CRE, which can be divided into consolidation-based methods (Kirkpatrick et al., 2017), dynamic architecture methods (Chen et al., 2015; Fernando et al., 2017) and Memory-based methods (Chaudhry et al., 2018; Han et al., 2020; Cui et al., 2021). Despite these methods' effectiveness on CRE, most of them have not taken full advantage of the negative relation information in all tasks to alleviate catastrophic forgetting more thoroughly, result in suboptimal CRE performance.

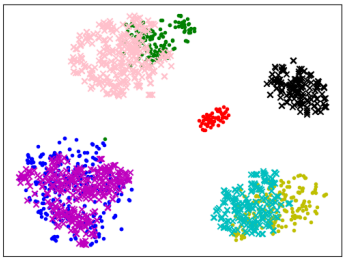

Figure 1: The data distribution map (better viewed in color) after training a classification model for an old task and then a new task. Many different relation data (different colors) of the old (dots) and new (crosses) task are mixed due to the catastrophic forgetting, making it hard to distinguish the new task's relations from the old task's relations.

Through our empirical studies, we found that the catastrophic forgetting of a model results in the indistinguishability between the data (instances) distributions of all tasks, making it hard to distinguish the relations of all tasks. We illustrate it with the data distribution map after training a relation classification model for a new task, as shown in Figure 1 where the dots and crosses represent the data of the old and new task respectively, and different colors represent different relations. It shows that the data points of different colors in either dot group (old task) or cross group (new task) are distinguishable. However, many dots and crosses are mixed, making it hard to discriminate the new task's relations from the old task's relations. Therefore, making the data distributions of all tasks more

* This work is supported by Shanghai Science and Technology Innovation Action Plan (China) No.21511100401.

Proceedings of the 29th International Conference on Computational Linguistics, pages 1885–1895
October 12–17, 2022.

distinguishable is crucial to achieve better CRE.

To address above issue, in this paper we propose a novel **C**ontinual **R**elation **E**xtraction framework with **C**ontrastive **L**earning, namely **CRECL**, which is built with a classification network and a contrastive network. In order to fully leverage the information of negative relations to make the data distributions of all tasks more distinguishable, we design a *prototypical contrastive learning* scheme. Specifically, in the contrastive network of CRECL, a given instance is contrasted with the prototype of each candidate relation stored in the memory module. Such sufficient comparisons ensure the alignment and uniformity between the data distributions of old and new tasks. Therefore, the catastrophic forgetting in CRECL is alleviated more thoroughly, resulting in enhanced CRE performance. In addition, different to the classification for a fixed (relation) class set as (Han et al., 2020; Cui et al., 2021), CRECL achieves an incremental-class learning of CRE which is more feasible to real-world CRE scenarios.

Our contributions in this paper are summarized as follows:

1. We propose a novel CRE framework CRECL that combines a classification network and a prototypical contrastive network to fully alleviate the problem of catastrophic forgetting.

2. With the contrasting-based mechanism, our CRECL can effectively achieve the class-incremental learning which is more practical in real-world CRE scenarios.

3. Our extensive experiments justify our CRECL's advantage over the state-of-the-art (SOTA) models on two benchmark datasets, TACRED and FewRel. Furthermore, we provide our deep insights into the reasons of the compared models' distinct performance.

2 Related Work

In this section, we briefly introduce continual learning and contrastive learning which are both related to our work.

Continual learning (Delange et al., 2021; Parisi et al., 2019) focuses on the learning from a continuous stream of data. The models of continual learning are able to accumulate knowledge across different tasks without retraining from scratch. The major challenge in continual learning is to alleviate catastrophic forgetting which refers to that the performance on previous tasks should not significantly decline over time as new tasks come in. For overcoming catastrophic forgetting, most recent works can be divided into three categories. 1) Regularized-based methods impose constraints on the update of parameters. For example, LwF approach (Li and Hoiem, 2016) enforces the network of previously learned tasks to be similar to the network of current task by knowledge distillation. However, LwF depends heavily on the data in new task and its relatedness to prior tasks. EWC (Kirkpatrick et al., 2016) adopts a quadratic penalty on the difference between the parameters for old and new tasks. It models the parameter relevance with respect to training data as a posterior distribution, which is estimated by Laplace approximation with the precision determined by the Fisher Information Matrix. WA (Zhao et al., 2020) maintains discrimination and fairness among the new and old task by adjust the parameters of the last layer. 2) Dynamic architecture methods change models' architectural properties upon new data by dynamically accommodating new neural resources, such as increased number of neurons. For example, PackNet (Mallya and Lazebnik, 2017) iteratively assigns parameter subsets to consecutive tasks by constituting pruning masks, which fixes the task parameter subset for future tasks. DER (Yan et al., 2021) proposes a novel two-stage learning approach to get more effective dynamically expandable representation. 3) Memory-based methods explicitly retrain the models on a limited subset of stored samples during the training on new tasks. For example, iCaRL (Rebuffi et al., 2017) focuses on learning in a class-incremental way, which selects and stores the samples most close to the feature mean of each class. During training, distillation loss between targets obtained from previous and current model predictions is added into overall loss, to preserve previously learned knowledge. RP-CRE (Cui et al., 2021) introduces a novel pluggable attention-based memory module to automatically calculate old tasks' weights when learning new tasks.

Since classification-based approaches require relation schema in the classification layer, classification-based models have an unignorable drawback on class-incremental learning. Many researchers leverage metric learning to solve this problem. (Wang et al., 2019; Wu et al., 2021) utilize sentence alignment model based on Margin Ranking Loss (Nayyeri et al., 2019), while lack the intrinsic ability to perform hard positive/negative

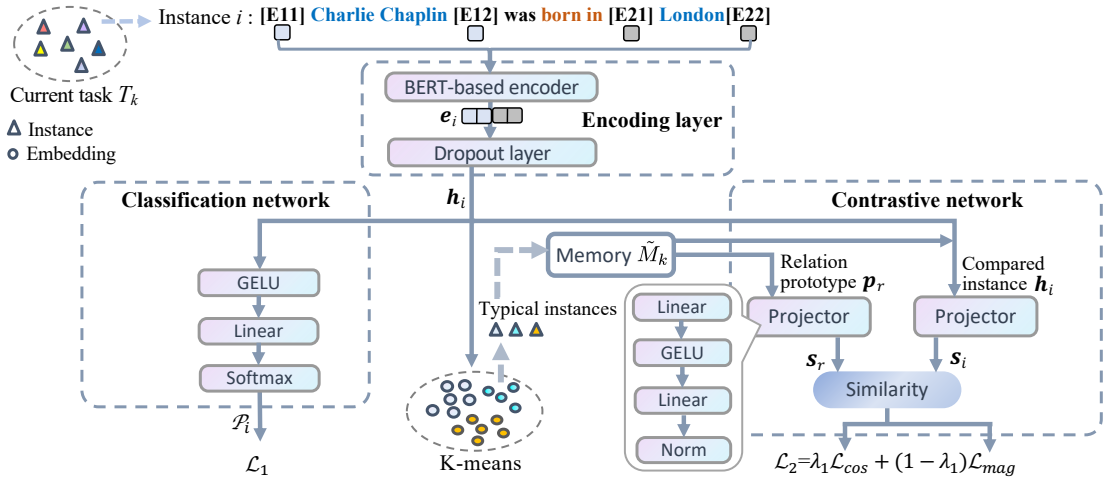

Figure 2: The overall structure of our proposed CRECL. The framework is built with a shared encoding layer, a classification network and a contrastive network.

mining, resulting in poor performance. Recently, contrastive learning has been widely imported into self-supervised learning frameworks in many fields including computer vision, natural language processing and so on. Contrastive learning is a discriminative scheme that aims to group similar samples more closer and diverse samples far from each other. (Wang and Liu, 2021) proves that contrastive learning can promote the alignment and stability of data distribution, and (Khosla et al., 2020) verifies that using modern batch contrastive approaches, such as InfoNCE loss (Oord et al., 2018), outperforms traditional contrastive losses, such as margin ranking loss, and also achieves good results in supervised contrastive learning tasks.

3 Methodology

3.1 Task Formalization

The CRE task aims to identify the relation between two entities expressed by one sentence in the task sequence. Formally, given a sequence of K tasks $\{T_1, T_2, \ldots, T_K\}$, suppose D_k and R_k denote the instance set and relation class set of the k-th task T_k, respectively. D_k contains N_k instances $\{(x_1, t_1, y_1), \ldots, (x_{N_k}, t_{N_k}, y_{N_k})\}$ where instance $(x_i, t_i, y_i), 1 \leq i \leq N_k$ represents that the relation of entity pair t_i in sentence x_i is $y_i \in R_k$. One CRE model should perform well on all historical tasks up to T_k, denoted as $\tilde{T}_k = \cup_{i=1}^{k} T_i$, of which the relation class set is $\tilde{R}_k = \cup_{i=1}^{k} R_i$. We also adopt an episodic memory module $M_r = \{(x_1, t_1, r), \ldots, (x_L, t_L, r)\}$ to store typical instances of relation r, similar to (Han et al., 2020; Cui et al., 2021), where L is the memory size (typical instance number). The overall episodic memory for the observed relations in all tasks is $\tilde{M}_k = \cup_{r \in \tilde{R}_k} M_r$.

3.2 Framework Overview

The overall structure of our CRECL is depicted in Figure 2, which has two major components, i.e., a classification network and a contrastive network. The procedure of learning the current task in CRECL is described by the algorithm in Alg. 1.

At first, suppose the current task is T_k, the representation of each instance in T_k is obtained through the encoder and dropout layer shared by the two networks. In the classification network, each instance's relation is predicted based on its representation (line 1-3). Then, we apply K-means algorithm over the instance representations to select L typical instances for each relation in T_k, which are used to generate the relation prototypes and stored into memory $\tilde{M}_k$ for the subsequent contrast (line 4-13). There are two training processes in the contrastive network. The first is to compare current task instances with the stored relation prototypes of $\tilde{T}_k$ (line 14-17). The second is to compare each typical instance with all relation prototypes which are both stored in $\tilde{M}_k$ (line 18-24). These two training procedures ensure each compared instance keep distance from sufficient negative relations in $\tilde{R}_k$. Therefore, the data distributions of $\tilde{R}_k$ are distinguishable enough to alleviate CRECL's catastrophic forgetting of old tasks. Next, we detail the operations in CRECL.

Algorithm 1: Training procedure for T_k

Input: $D_k, R_k, \tilde{R}_{k-1}, \tilde{M}_{k-1}$
Output: $\tilde{M}_k$

for i = *1 to epochs1* **do**
  update Encoder EN, Dropout layer DR and Classifier CL by loss $\mathcal{L}_1$ on D_k;
end
$\tilde{R}_k = \tilde{R}_{k-1} \cup R_k$;
$\boldsymbol{p} = \emptyset$;
for $r \in R_k$ **do**
  $d_r = \{d | d \in D_k, y_d = r\}$;
  $\boldsymbol{h}_r$=DR(EN(d_r))//Eq. 1;
  apply K-means to all $\boldsymbol{h}_r$ and store L typical instances into memory M_k;
  get prototype $\boldsymbol{p}_r$ from M_k by Eq. 4;
  $\boldsymbol{p} = \boldsymbol{p} \cup \boldsymbol{p}_r$;
end
$\tilde{M}_k = \tilde{M}_{k-1} \cup M_k$;
generate contrastive training data C_k by random sampling from D_k;
for i = *1 to epochs2* **do**
  update EN, DR, PJ by loss $\mathcal{L}_2$ on C_k and $\boldsymbol{p}$;
end
generate contrastive training data $\tilde{C}_k$ by random sampling from $\tilde{M}_k$;
for i = *1 to epochs3* **do**
  if $y_i \in \tilde{C}_k$ *is old class* **then**
    use Eq. 1 to generate M embeddings;
  end
  update EN, DR, PJ by loss $\mathcal{L}_2$ on $\tilde{C}_k$ and $\boldsymbol{p}$;
end

3.3 Shared Encoding Layer

The classification network and the contrastive network in CRECL are designed to promote each other, where the former classifies the current task based on its instance embeddings, and the latter effectively adjusts instance embeddings to keep uniformity and alignment. According to this principle, the two networks share the same layers in CRECL.

Specifically, for an instance i of current task T_k, we use special tokens to represent the entities in i as (Cui et al., 2021). As shown in Figure 2, the head entity and tail entity in i are represented by two special position tokens $[E11, E12]$ and $[E21, E22]$, respectively. The embedding of instance i before the dropout layer, denoted as $\boldsymbol{e}_i \in \mathbb{R}^{2h}$, is the concatenation of token embeddings of $[E11, E12]$ and $[E21, E22]$ generated by BERT (Devlin et al., 2019) where h is the dimension of two token embeddings. Then, $\boldsymbol{e}_i$ is fed into the dropout layer to obtain i's hidden embedding as

$$\boldsymbol{h}_i = \big(\boldsymbol{W}\,\mathrm{Dropout}(\boldsymbol{e}_i) + \boldsymbol{b}\big) \in \mathbb{R}^d, \quad (1)$$

where $\boldsymbol{W} \in \mathbb{R}^{d \times 2h}$ (d is dimension of hidden layer) and $\boldsymbol{b} \in \mathbb{R}^d$ are both trainable parameters. In CRECL, $\boldsymbol{h}_i$ is regarded as i's representation.

3.4 Classifying Current Task

With instance i's representation $\boldsymbol{h}_i$, i's probability distribution denoted as $\mathcal{P}_i \in \mathbb{R}^{|R_k|}$, is calculated in the classification network as

$$\mathcal{P}_i = \mathrm{softmax}\big(\boldsymbol{W}_1\,\mathrm{LN}\big(\mathrm{GELU}(\boldsymbol{h}_i)\big) + \boldsymbol{b}_1\big), \quad (2)$$

where $\boldsymbol{W}_1 \in \mathbb{R}^{|R_k| \times d}, \boldsymbol{b}_1 \in \mathbb{R}^{|R_k|}$ are trainable parameters, and $|R_k|$ is the relation number of current task T_k which is much less than the relation number of all tasks. $\mathrm{LN}(\cdot)$ is layer normalization operation. Then, classification loss for current task T_k is calculated as

$$\mathcal{L}_1 = -\frac{1}{N_k}\sum_{i=1}^{N_k}\sum_{r=1}^{|R_k|} y_{i,r} \log \mathcal{P}_i^r, \quad (3)$$

where $y_{i,r}$=1 if i's real relation label is r, otherwise $y_{i,r}$=0. $\mathcal{P}_i^r$ is the r-th entry in $\mathcal{P}_i$.

3.5 Generating Relation Prototypes

After learning current task, for each relation r in current task, we first apply K-means algorithm upon the representations ($\boldsymbol{h}_i$) of all instances belonging to r to cluster them into L clusters. Then, for each cluster, we select the instance most closest to the centroid of this cluster as one typical instance. Thus, L typical instances of relation r are selected and then stored into the memory module. With the stored typical instances of r, we average their representations as r's *prototype* $\boldsymbol{p}_r$, that is

$$\boldsymbol{p}_r = \frac{1}{L}\sum_{i=1}^{L} \boldsymbol{h}_i^r, \quad (4)$$

where $\boldsymbol{h}_i^r$ is a typical instance i's representation of relation r. Such prototype best represents r since the L typical instances have the minimal distance sum to the L cluster centroids. Another merit of such prototypes for representing relations is their insensitivity to the value of L.

3.6 Contrastive Network

In this contrastive network, the instances are compared with the relation prototypes stored in the memory module to refine the data distributions of all tasks, so as to alleviate CRECL's catastrophic forgetting. Its basic principle is that, an instance's representation should be close to the prototype of its (positive) relation, and be far away

from the prototypes of the rest (negative) relations. Please note that, the positive and negative relations are identified by the real labels of the training instances. Thus it is different from the self-supervised contrastive learning in other models (Chen et al., 2020).

Contrastive Learning Objective As shown in the right part of Figure 2, the contrastive network is built with a twin-tower architecture. In the left tower, for a relation r, its prototypes $\boldsymbol{p}_r \in \mathbb{R}^d$ are obtained by Eq. 4. Then, r's compared embedding is denoted as $\boldsymbol{s}_r \in \mathbb{R}^{\frac{d}{2}}$ and computed as

$$\boldsymbol{s}_r = \boldsymbol{W}_3 \,\mathrm{GELU}(\boldsymbol{W}_2 \boldsymbol{p}_r + \boldsymbol{b}_2) + \boldsymbol{b}_3, \quad (5)$$

where $\boldsymbol{W}_2 \in \mathbb{R}^{d\times d}$, $\boldsymbol{b}_2 \in \mathbb{R}^d$, $\boldsymbol{W}_3 \in \mathbb{R}^{\frac{d}{2}\times d}$, $\boldsymbol{b}_3 \in \mathbb{R}^{\frac{d}{2}}$ are both trainable parameters.

In the right tower, for a compared instance i, its compared embedding is denoted as $\boldsymbol{s}_i \in \mathbb{R}^{\frac{d}{2}}$ and obtained by the same operation in Eq. 5 where only $\boldsymbol{p}_r$ is replaced by $\boldsymbol{h}_i$ from Eq. 1.

For each instance i in current task T_k, suppose the compared embedding of i's relation y_i is $\boldsymbol{s}_{y_i}$ which can also be obtained by Eq. 5, since the typical instances of y_i have been stored in $\tilde{M}_k$ before. We apply Euclidean norm to $\boldsymbol{s}_i$ and $\boldsymbol{s}_{y_i}$. Then, we use contrastive learning's InfoNCE loss (Oord et al., 2018) to calculate the cosine similarity loss of T_k as

$$\mathcal{L}_{cos} = -\frac{1}{N_k}\sum_{i=1}^{N_k} \log \frac{\exp\left(\boldsymbol{s}_i \boldsymbol{s}_{y_i}/\tau\right)}{\sum\limits_{r\in\tilde{R}_k} \exp\left(\boldsymbol{s}_i \boldsymbol{s}_r/\tau\right)}, \quad (6)$$

where τ is a temperature hyper-parameter.

To increase the similarity score gap of the correct label and the closest wrong label, inspired by (Koch et al., 2015), we propose a contrastive margin loss

$$\mathcal{L}_{mag} = \frac{1}{N_k}\sum_{i=1}^{N_k} max\big(m - \boldsymbol{s}_i\boldsymbol{s}_{y_i} + \boldsymbol{s}_i\boldsymbol{s}_{k_i}, 0\big), \quad (7)$$

where relation $k_i = \arg\max\limits_{k\in\tilde{R}_k} \boldsymbol{s}_i\boldsymbol{s}_k$ s.t. $k \neq y_i$, that is i's closest negative relation label. The margin loss penalizes that the similarity gap less than m. At last, the total loss is defined as

$$\mathcal{L}_2 = \lambda_1 \mathcal{L}_{cos} + (1-\lambda_1)\mathcal{L}_{mag}, \quad (8)$$

where $\lambda_1 \in [0, 1]$ is the controlling parameter.

Training Processes of Contrastive Learning There are two training processes in the contrastive network, which both use the loss in Eq. 8 to make the network parameters more fit to current task and all historical tasks, respectively.

The first training process is conducted with current task T_k and complements the classification network, it is an optional step with relatively small training epoch. However, it can not ensure the model fit to all tasks, because the model pays more attention to current task rather than the old tasks during this training process. In other words, as we have explained in the example of Figure 1, the model tends to ensure the instances of different relations in T_k distinguishable, but forgets to meanwhile keep the instances of different relations in all historical tasks also distinguishable. As a result, the model's catastrophic forgetting still happens.

To alleviate CRECL's catastrophic forgetting more thoroughly, we introduce the second training process in the contrastive network. In this process, all typical instances stored in the memory module are compared with all prototypes of stored relations, which cover in all tasks. We also conduct M times forward propagation in the dropout layer, to generate M embeddings for each old relation in $\tilde{R}_{k-1}$. Due to the randomness of dropout layer, we can get M probability distributions for an old relation to reduce the imbalance of data distribution of old and new relation. Accordingly, this training process can effectively prevent the model from severe catastrophic forgetting.

3.7 Relation Prediction

For a predicted instance i, we only measure its similarity to each stored relation, which is computed as the cosine distance between i's representation and the relation's prototype. Then, we choose the most similar (closest) relation as i's predicted class label, that is

$$y_i^* = \arg\max_{r\in\tilde{R}_k} \boldsymbol{s}_i\boldsymbol{s}_r. \quad (9)$$

4 Experiments

4.1 Datasets

Our experiments were conducted upon the following two benchmark CRE datasets.

FewRel (Han et al., 2018) is a popular relation extraction dataset originally constructed for few-shot relation extraction. The dataset is annotated

Table 1: Accuracy (%) comparisons on different test sets of historical cumulative tasks, showing that CRECL outperforms the compared models.

	FewRel									
Model	T1	T2	T3	T4	T5	T6	T7	T8	T9	T10
EA-EMR	89.0	69.0	59.1	54.2	47.8	46.1	43.1	40.7	38.6	35.2
EMAR	88.5	73.2	66.6	63.8	55.8	54.3	52.9	50.9	48.8	46.3
CML	91.2	74.8	68.2	58.2	53.7	50.4	47.8	44.4	43.1	39.7
EMAR+BERT	**98.8**	89.1	89.5	85.7	83.6	84.8	79.3	80.0	77.1	73.8
RP-CRE+MA	98.0	91.4	91.8	86.8	87.6	86.9	83.7	81.9	80.1	79.5
RP-CRE	97.9	92.7	91.6	89.2	88.4	86.8	85.1	84.1	82.2	81.5
CRECL+ATM(Ours)	96.3	91.4	89.3	90.0	88.1	86.7	84.5	83.2	82.6	81.0
CRECL(Ours)	97.8	**94.9**	**92.7**	**90.9**	**89.4**	**87.5**	**85.7**	**84.6**	**83.6**	**82.7**
Improvement(%)	-1.01	2.37	0.98	1.91	1.13	0.69	0.71	0.59	1.70	1.47
	TACRED									
Model	T1	T2	T3	T4	T5	T6	T7	T8	T9	T10
EA-EMR	47.5	40.1	38.3	29.9	28.4	27.3	26.9	25.8	22.9	19.8
EMAR	73.6	57.0	48.3	42.3	37.7	34.0	32.6	30.0	27.6	25.1
CML	57.2	51.4	41.3	39.3	35.9	28.9	27.3	26.9	24.8	23.4
EMAR+BERT	96.6	85.7	81.0	78.6	73.9	72.3	71.7	72.2	72.6	71.0
RP-CRE+MA	97.1	91.4	87.4	82.1	78.3	77.8	74.9	73.5	73.6	72.3
RP-CRE	**97.6**	90.6	86.1	82.4	79.8	77.2	75.1	73.7	72.4	72.4
CRECL+ATM(Ours)	93.2	80.2	77.3	76.0	71.8	71.5	69.2	72.3	70.0	71.2
CRECL(Ours)	96.6	**93.1**	**89.7**	**87.8**	**85.6**	**84.3**	**83.6**	**81.4**	**79.3**	**78.5**
Improvement(%)	-1.02	1.86	2.63	6.55	7.27	8.35	11.32	10.45	7.74	8.43

by crowd workers and contains 100 relations and 70,000 samples in total. In our experiments, to keep consistent with the previous baselines, we used its version of 80 relations.

TACRED (Zhang et al., 2017) is a large-scale relation extraction dataset containing 42 relations (including *no_relation*) and 106,264 samples from news and web documents. Based on the open relation assumption of CRE, we removed *no_relation* in our experiments. To limit the sample imbalance of TACRED, we limited the number of training samples of each relation to 320, and the number of test samples of each relation to 40, which is also consistent with previous baselines. Compared with FewRel, the tasks in TACRED are more difficult due to its relation imbalance and semantic difficulty.

4.2 Compared Models

We compare our framework with the following baselines in our experiments.

EA-EMR (Wang et al., 2019) proposes a sentence alignment model with replay memory module to alleviate catastrophic forgetting.

EMAR (Han et al., 2020) proposes a novel memory replay, activation and reconsolidation method to alleviate catastrophic forgetting effectively.

EMAR+BERT is an advanced version of EMAR where the original encoder (Bi-LSTM) is replaced with BERT.

CML (Wu et al., 2021) proposes a curriculum-meta learning method to tackle the order-sensitivity and catastrophic forgetting in CRE.

RP-CRE (Cui et al., 2021) is a SOTA CRE model introducing a novel pluggable attention-based memory module to automatically calculate the weight of old tasks when learning new tasks.

RP-CRE+MA is an advanced version of RP-CRE where a memory activation step is added before attention operation.

In our CRECL, we adopted the *Bert-base-uncased* pre-trained by HuggingFace (Wolf et al., 2020) as the encoder, which is also used in EMAR+BERT, RP-CRE and RP-CRE+MA. Other baselines cannot be easily replaced by the BERT due to their architectures. In addition, we propose another version of CRECL, namely **CRECL+ATM**, which incorporates an attention memory module proposed by (Cui et al., 2021) in the contrastive network and used to verify its effectiveness of refining relation prototypes.

4.3 Experimental Settings

Our evaluation metric is *Accuracy* which is popularly used in previous baselines.

For fair comparisons, we followed the experiment settings in RP-CRE. At first, to verify whether a CRE model suffers from catastrophic forgetting, we use Tk to represent the test set of all historical cumulative tasks from the first task to the k-th task T_k (Please note the difference between Tk and T_k). In our ablation studies, we also report the perfor-

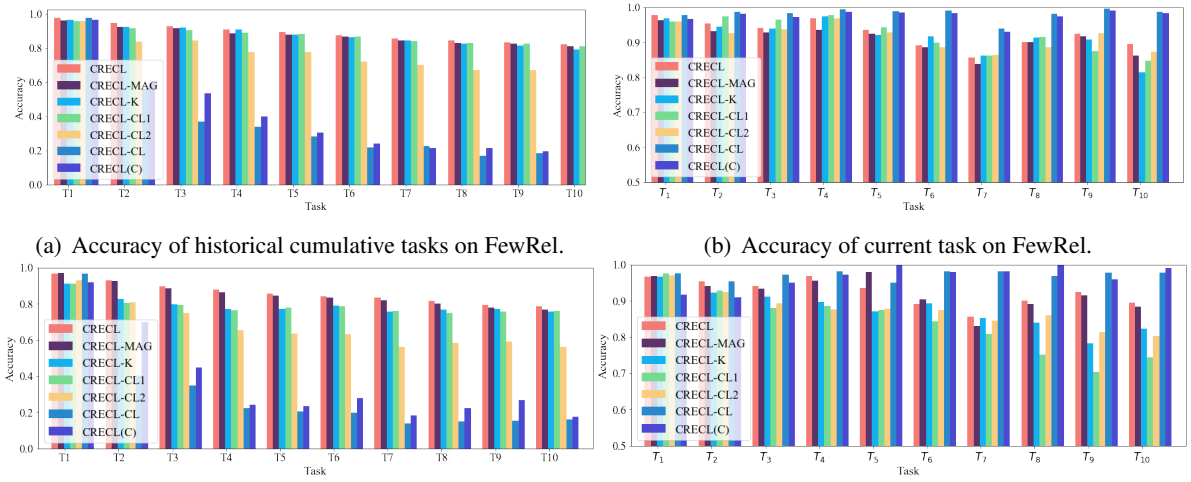

(a) Accuracy of historical cumulative tasks on FewRel.

(b) Accuracy of current task on FewRel.

(c) Accuracy of historical cumulative tasks on TACRED.

(d) Accuracy of current task on TACRED.

Figure 3: Ablation study results on historical cumulative tasks and current task.

mance on the test set of current task. To simulate different tasks, we randomly divided all instances into 10 groups (corresponding to 10 tasks). The task order of all compared models is exactly the same to reduce contingency. We also set the memory size in the baselines the same as ours. Relations are first divided into 10 clusters to simulate 10 tasks. All the reported results of the related baselines are the same as (Cui et al., 2021). For those special hyper-parameters in our experiments are as follows. The batch size is 32, the learning rate is set to 5e-5, τ is 0.08. We adopted 10 and 15 classification epochs for TACRED and FewRel, respectively. We also adopted 10 epochs for the first training process (for current task) and 5 epochs for the second training process (for all tasks) in the contrastive network.

Because the total matrix operations and the data amount of second training in contrastive learning are very small, CRECLl's training time (1h31min) is very close to the SOTA model RP-CRE (1h28min). To reproduce our experiment results conveniently, CRECL's source code together with the datasets are provided at https://github.com/PaperDiscovery/CRECL.

4.4 Experimental Results and Analyses

The following reported results of CRECL and its ablated variants are the average scores of running models for 5 times.

4.4.1 Overall Performance Comparisons

The overall performance of all compared baselines are reported in Table 1, where the results of the baselines directly come from (Cui et al., 2021) and the baselines' hyper-parameter settings were the same as their original papers. The last row in the table is the improvement ratio of CRECL's performance relative to the best baseline's performance (underline). Based on these results, we have the following conclusions:

(1) Our CRECL outperforms the SOTA model RP-CRE on both datasets. Compared with FewRel, CRECL has more apparent improvement over the baselines on TACRED. This may be due to that FewRel's tasks are not difficult enough as TACRED, proving that CRECL is good at handling more difficult tasks.

(2) In T1, our CRECL is inferior to RP-CRE because the classification and contrastive network in CRECL have not been fully trained at the beginning. When more tasks are cumulated, CRECL is trained sufficiently, resulting in its superiority over the baselines and less performance drop. Since the catastrophic forgetting becomes more severe on such scenarios, the results imply that CRECL can tackle the catastrophic forgetting better.

(3) All compared models' performance is well on T1, but declines when more new tasks arrive due to more severe catastrophic forgetting. Compared with EA-EMR, EMAR and CML, the rest models' performance decline is more slight. For example, from the comparison between EMAR (using Bi-LSTM) and EMAR+BERT, we can see that EMAR+BERT's performance decline significantly slows down, proving that BERT helps the model alleviate the catastrophic forgetting better. It is because BERT has good feature discrimination ability and better captures the relevant features, making

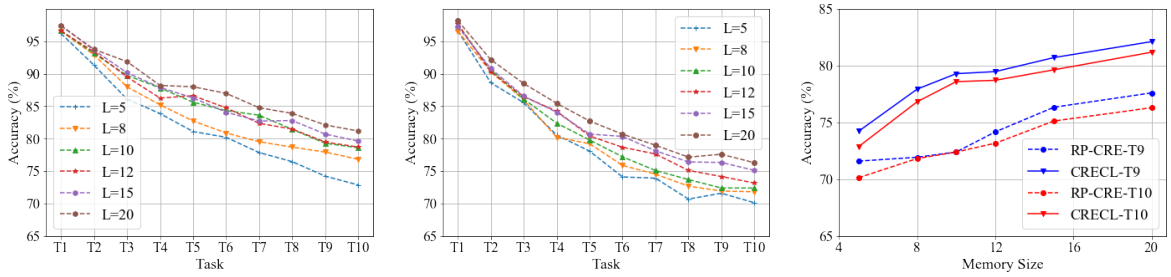

(a) CRECL (b) RP-CRE (c) CRECL vs. RP-CRE in T9/T10

Figure 4: Performance comparisons with different memory sizes on TACRED.

catastrophic forgetting less severe.

(4) CREL+ATM's results show that incorporating attention memory module fails to improve CRECL's performance well, because the contrastive network is able to maintain the uniformity and the alignment of data distribution. Thus there is no need for additional attention memory module to help the model refine relation prototypes.

(5) Compared with the models using metric learning (EA-EMR, EMAR, CML, EMAR+BERT), we adopt InfoNCE loss instead of the margin ranking loss as our loss function. With this loss, CRECL is taught by more negative relation information to understand how to regularize data representation space, resulting in the more alleviation of catastrophic forgetting.

4.4.2 Ablation Studies

In order to verify the effectiveness and rationality of our framework's important components (steps), we further conducted a series of ablation experiments. CRECL's ablated variants include:

CRECL-MAG: It is the variant without the margin loss $\mathcal{L}_{mag}$ in the contrastive network.

CRECL-CL1: It is the variant without the first training process in the contrastive network.

CRECL-CL2: It is the variant without the second training process in the contrastive network.

CRECL-CL: It is the variant only having the classification network.

CRECL-K: In this variant, the typical instances of each relation are selected at random instead of by K-means algorithm.

CRECL(C): This variant uses the classification network to identify the relation of a test instance instead of the similarity comparison in Eq. 9.

Figure 3 (a) and (b) display all compared models' accuracy of historical cumulative tasks and current task. Due to space limitation, only the results on TACRED are shown, based on which we have the following analyses.

(1) CRECL-CL performs very well on current task (subfigure (b)) but performs very poorly on historical tasks (subfigure (a)), showing that it overfits current task and its catastrophic forgetting is very severe. It is because that the classification parameters are always tuned to fit with current task rather than old tasks. It shows that the contrastive network is significant to alleviate catastrophic forgetting.

(2) As more new tasks arrive, CRECL-CL1's performance decline on current task (subfigure (b)) is more obvious than its performance decline on historical tasks (subfigure (a)), because CRECL-CL1 pays more attention to distinguish the different relations in old tasks rather than that in current task. It is due to that the data distributions of all historical tasks are adjusted in the second training process that CRECL-CL1 only has in its contrastive network. Comparatively, CRECL-CL2's performance on historical tasks and current task both declines. It shows that only distinguishing the data distribution of current task from that of old tasks in the first training process of contrastive network, is not adequate to alleviate its catastrophic forgetting. Even worse, such adjusting also harms the accuracy of classifying current task.

(4) CRECL-K is inferior to CRECL, showing that the randomly selected instances cannot well represent relations as those selected by K-means algorithm. As a result, the data distributions of all tasks cannot be adjusted precisely, which can not alleviate catastrophic forgetting effectively. In addition, CRECL-K's accuracy on current task is not stable also due to the randomness led by its selection strategy of typical instances.

(5) Although the contrastive learning loss $\mathcal{L}_2$ is different from the classification network's loss $\mathcal{L}_1$, and the parameters of the encoding layer are shared, the contrastive network's training processes hardly weaken the classification network's fitness to cur-

Table 2: Our framework's accuracy (%) with different memory sizes on FewRel.

CRECL	T1	T2	T3	T4	T5	T6	T7	T8	T9	T10
L=5	97.1	92.1	89.7	90.0	88.2	86.6	84.4	82.8	82.5	80.2
L=10	97.8	94.9	92.7	90.9	89.4	87.5	85.7	84.6	83.6	82.7
L=15	98.9	96.3	93.8	92.6	91.4	90.0	88.0	86.6	85.8	83.5
L=20	98.1	96.0	94.1	92.8	91.9	89.7	88.5	86.3	86.1	85.2

rent task. Thus CRECL(C) still performs well on current task as shown in Figure 3 (b). CRECL-MAG's has a relatively small decline on both current and historical tasks, proving that the margin loss $\mathcal{L}_{mag}$ improves the performance by increasing the gap between the optimal and suboptimal results.

4.4.3 Performance Influence of Memory Size

For memory-based CRE methods, the model performance is usually related to the storage capacity of their memory modules. Specifically, we found that previous models' performance is very sensitive to the number of stored typical instances L. Recall that in CRECL, a relation prototype is the average of L typical instances' representations. The representational ability of such prototype is less sensitive to L when L exceeds a certain value, resulting in CRECL's performance also less sensitive to L, as we emphasized in Section 3.5.

Figure 4 displays the performance of CRECL and RP-CRE on TACRED w.r.t. different memory sizes (L), where the two compared models' performance in T9 and T10 is specially shown in the subfigure (c). It shows that although CRECL's performance also declines when L becomes small, it is more stable and higher than RP-CRE's performance, especially when $L \geq 8$. Such results justifies our claim about CRECL's less sensitivity to L. In addition, RP-CRE's performance fluctuation is more obvious, possibly because it re-constructs the attention memory network upon each task, so the different task features are not shared in the network.

4.4.4 Contrastive Learning's Effectiveness on Refining Data Distributions

In addition, to investigate the contrastive learning's effects on alleviating catastrophic forgetting, we use *t-SNE* (Van der Maaten and Hinton, 2008) to visualize the data distributions of the same case in Figure 1 after the training processes of CRECL's classification network and contrastive network. The data distribution map is shown in Figure 5, which has the same settings of color and point style as Figure 1. Through comparing these two maps, we find that the data distributions of different relations in the new and old task in Figure 5 are more distinguishable than that in Figure 1. Such results are mainly attributed to the prototypical contrastive learning in CRECL on adjusting all data distributions of all tasks, which obviously alleviate CRECL's catastrophic forgetting. It has been also proved by CRECL's superior performance over the baselines displayed in aforementioned experiments. We also note that some yellow dots still intersect the pink crosses, possibly due to the insufficient sampling in the contrastive learning, resulting in less coverage on all relations. We can handle this situation by increasing the batch size.

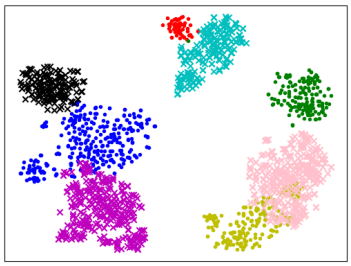

Figure 5: Data distribution map (better viewed in color) after the contrastive learning in CRECL. Compared with Fig. 1, the data distributions of different relations (different colors) are obviously distinguishable, making CRECL classify different relations more easily.

5 Conclusion

In this paper, we propose a novel CRE framework, namely CRECL, that consists of a classification network and a contrastive network designed for alleviating the catastrophic forgetting in CRE. Through the prototypical contrastive learning in CRECL, the data distributions of different relations in all tasks are adjusted to be more distinguishable, resulting in CRE performance gains. Moreover, CRECL has the ability of class-incremental learning due to its contrasting-based mechanism of achieving relation classification, which is more practical in real-world CRE scenarios than the previous models of classification-based mechanism. Our extensive experimental results demonstrate that CRECL outperforms the SOTA CRE baselines and obtains the best performance on two benchmark datasets.

References

Arslan Chaudhry, Marc'Aurelio Ranzato, Marcus Rohrbach, and Mohamed Elhoseiny. 2018. Efficient lifelong learning with a-gem. *arXiv preprint arXiv:1812.00420*.

Tianqi Chen, Ian Goodfellow, and Jonathon Shlens. 2015. Net2net: Accelerating learning via knowledge transfer. *arXiv preprint arXiv:1511.05641*.

Ting Chen, Simon Kornblith, Mohammad Norouzi, and Geoffrey Hinton. 2020. A simple framework for contrastive learning of visual representations. In *International conference on machine learning*, pages 1597–1607. PMLR.

Li Cui, Deqing Yang, Jiaxin Yu, Chengwei Hu, and et al. 2021. Refining sample embeddings with relation prototypes to enhance continual relation extraction. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics*, pages 232–243.

Matthias Delange, Rahaf Aljundi, Marc Masana, Sarah Parisot, Xu Jia, Ales Leonardis, Greg Slabaugh, and Tinne Tuytelaars. 2021. A continual learning survey: Defying forgetting in classification tasks. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, pages 1–1.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: pre-training of deep bidirectional transformers for language understanding. In *Proceedings of NAACL*, pages 4171–4186.

Chrisantha Fernando, Dylan Banarse, Charles Blundell, Yori Zwols, David Ha, Andrei A Rusu, Alexander Pritzel, and Daan Wierstra. 2017. Pathnet: Evolution channels gradient descent in super neural networks. *arXiv preprint arXiv:1701.08734*.

Xu Han, Yi Dai, Tianyu Gao, Yankai Lin, Zhiyuan Liu, Peng Li, Maosong Sun, and Jie Zhou. 2020. Continual relation learning via episodic memory activation and reconsolidation. In *Proceedings of ACL*, pages 6429–6440.

Xu Han, Hao Zhu, Pengfei Yu, Ziyun Wang, Yuan Yao, Zhiyuan Liu, and Maosong Sun. 2018. Fewrel: A large-scale supervised few-shot relation classification dataset with state-of-the-art evaluation. *arXiv preprint arXiv:1810.10147*.

Demis Hassabis, Dharshan Kumaran, Christopher Summerfield, and Matthew Botvinick. 2017. Neuroscience-inspired artificial intelligence. *Neuron*, 95(2):245–258.

Prannay Khosla, Piotr Teterwak, Chen Wang, Aaron Sarna, Yonglong Tian, Phillip Isola, Aaron Maschinot, Ce Liu, and Dilip Krishnan. 2020. Supervised contrastive learning. *arXiv preprint arXiv:2004.11362*.

James Kirkpatrick, Razvan Pascanu, Neil Rabinowitz, Joel Veness, Guillaume Desjardins, Andrei A Rusu, Kieran Milan, John Quan, Tiago Ramalho, Agnieszka Grabska-Barwinska, et al. 2017. Overcoming catastrophic forgetting in neural networks. *Proceedings of the national academy of sciences*, 114(13):3521–3526.

James Kirkpatrick, Razvan Pascanu, Neil C. Rabinowitz, Joel Veness, and et al. 2016. Overcoming catastrophic forgetting in neural networks. *CoRR*, abs/1612.00796.

Gregory Koch, Richard Zemel, Ruslan Salakhutdinov, et al. 2015. Siamese neural networks for one-shot image recognition. In *ICML deep learning workshop*, volume 2, page 0. Lille.

Zhizhong Li and Derek Hoiem. 2016. Learning without forgetting. *CoRR*, abs/1606.09282.

Arun Mallya and Svetlana Lazebnik. 2017. Packnet: Adding multiple tasks to a single network by iterative pruning. *CoRR*, abs/1711.05769.

Mojtaba Nayyeri, Xiaotian Zhou, Sahar Vahdati, Hamed Shariat Yazdi, and Jens Lehmann. 2019. Adaptive margin ranking loss for knowledge graph embeddings via a correntropy objective function. *arXiv preprint arXiv:1907.05336*.

Aaron van den Oord, Yazhe Li, and Oriol Vinyals. 2018. Representation learning with contrastive predictive coding. *arXiv preprint arXiv:1807.03748*.

German I. Parisi, Ronald Kemker, Jose L. Part, Christopher Kanan, and Stefan Wermter. 2019. Continual lifelong learning with neural networks: A review. *Neural Networks*, 113:54–71.

Sylvestre-Alvise Rebuffi, Alexander Kolesnikov, Georg Sperl, and Christoph H. Lampert. 2017. icarl: Incremental classifier and representation learning. In *2017 IEEE Conference on Computer Vision and Pattern Recognition*, pages 5533–5542.

Sebastian Thrun and Tom M Mitchell. 1995. Lifelong robot learning. *Robotics and autonomous systems*, 15(1-2):25–46.

Laurens Van der Maaten and Geoffrey Hinton. 2008. Visualizing data using t-sne. *Journal of machine learning research*, 9(11).

Feng Wang and Huaping Liu. 2021. Understanding the behaviour of contrastive loss. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 2495–2504.

Hong Wang, Wenhan Xiong, Mo Yu, Xiaoxiao Guo, Shiyu Chang, and William Yang Wang. 2019. Sentence embedding alignment for lifelong relation extraction. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 796–806.

Thomas Wolf, Lysandre Debut, Victor Sanh, and et al. 2020. Transformers: State-of-the-art natural language processing. In *Proceedings of EMNLP*, pages 38–45.

Tongtong Wu, Xuekai Li, Yuan-Fang Li, Gholamreza Haffari, Guilin Qi, Yujin Zhu, and Guoqiang Xu. 2021. Curriculum-meta learning for order-robust continual relation extraction. pages 10363–10369.

Shipeng Yan, Jiangwei Xie, and Xuming He. 2021. Der: Dynamically expandable representation for class incremental learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 3014–3023.

Yuhao Zhang, Victor Zhong, Danqi Chen, Gabor Angeli, and Christopher D. Manning. 2017. Position-aware attention and supervised data improve slot filling. In *Proceedings of EMNLP*, pages 35–45.

Bowen Zhao, Xi Xiao, Guojun Gan, Bin Zhang, and Shu-Tao Xia. 2020. Maintaining discrimination and fairness in class incremental learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 13208–13217.

Prompt-based Text Entailment for Low-Resource Named Entity Recognition

Dongfang Li[1], **Baotian Hu**[1*], **Qingcai Chen**[1,2*]
[1]Harbin Institute of Technology (Shenzhen), Shenzhen, China
[2]Peng Cheng Laboratory, Shenzhen, China
crazyofapple@gmail.com, {hubaotian, qingcai.chen}@hit.edu.cn

Abstract

Pre-trained Language Models (PLMs) have been applied in NLP tasks and achieve promising results. Nevertheless, the fine-tuning procedure needs labeled data of the target domain, making it difficult to learn in low-resource and non-trivial labeled scenarios. To address these challenges, we propose Prompt-based Text Entailment (PTE) for low-resource named entity recognition, which better leverages knowledge in the PLMs. We first reformulate named entity recognition as the text entailment task. The original sentence with entity type-specific prompts is fed into PLMs to get entailment scores for each candidate. The entity type with the top score is then selected as final label. Then, we inject tagging labels into prompts and treat words as basic units instead of n-gram spans to reduce time complexity in generating candidates by n-grams enumeration. Experimental results demonstrate that the proposed method PTE achieves competitive performance on the CoNLL03 dataset, and better than fine-tuned counterparts on the MIT Movie and Few-NERD dataset in low-resource settings.

1 Introduction

Recently, Pre-trained Language Models (PLMs) have achieved promising improvement on several NLP tasks (Devlin et al., 2019; Liu et al., 2019; Lan et al., 2020). Nevertheless, fine-tuning language models still needs a moderate number of labeled data for downstream tasks. When difficulties result in limited labeled data available, the trained model shows large variance in downstream performance under full fine-tuning (Mosbach et al., 2021; Le Scao and Rush, 2021). For example, labeling technical and professional terms can be time-consuming and labor-intensive in medical scenarios. Moreover, crowd-sourced annotation is also limited by the reality of existing samples (e.g., when online health assistants are applied to rare diseases).

To address learning challenges in these low-resource scenarios, researchers find that PLMs can learn well by prompt-based learning (Schick and Schütze, 2021a,b; Tam et al., 2021). Prompt-based learning models the probability of text directly; it does not need an extra fully-connected layer usually used by fine-tuning. The main idea is to reformulate NLP tasks as cloze-style question answering for better using the knowledge in PLMs. The model predicts the word probability of masked positions and then derives the final output via mapping relations between words and labels. Previous works have shown the ability of prompt-based learning under low-resource settings (Schick and Schütze, 2021a,b; Schick et al., 2020; Lester et al., 2021). For example, some prompt-based works have explored in classification and generation tasks where it is relatively easy to reformulate into cloze-style tasks (cf. Section 4). Nevertheless, the application to Named Entity Recognition (NER) still poses challenges for current methods. Unlike text classification and text generation, NER is the task of identifying named entities (e.g., *person name*, *location*) in a given sentence, and each unit of the input needs to be predicted. If we directly use Masked Language Modeling (MLM) head to predict each unit label, the lexical and semantic coherence are ignored as there exists latent relationships between the tokens (Lample et al., 2016; Peters et al., 2018; Yan et al., 2019).

In this work, we propose **Prompt-based Text Entailment** (PTE) for low-resource NER. Firstly, we reformulate NER as a *text entailment* task. Textual Entailment (TE) is the task of studying the relation of two sentences, Premise (P) and Hypothesis (H): whether H is true given P (Bowman et al., 2015). Specifically, we treat the original sentence as premise and entity type-specific prompt as a hypothesis. Given an entity type, the P and H are

* Corresponding authors

Proceedings of the 29th International Conference on Computational Linguistics, pages 1896–1903
October 12–17, 2022.

fed into PLMs to get entailment scores for each candidate. Then, the entailment score is the probability of a specific token at the mask position of the prompt. After that, the entity type with the top entailment score is selected as the final label. During inference, we enumerate all possible text spans or words in the input sentence as named entity candidates (Cui et al., 2021). The reformulation provides a unified entailment framework for NER tasks where annotations are insufficient, as the model shares the same inference pattern across different domains. As such, we can also leverage generic text entailment datasets such SNLI (Bowman et al., 2015) and MNLI (Williams et al., 2018) to pre-train models, which transfer knowledge from the general domain and get better performance in new domains. Our method can be a step forward towards the development of a solution for the low-resource NER because any new domain does not typically have extensive annotated data in the real world, whereas it is feasible to obtain a couple of examples (e.g., online assistant). Moreover, considering the existence of noisy annotations, *TE only needs to specify the labels of certain entities for training rather than the complete annotations of the entire sequence*. Experimental results demonstrate that the proposed method `PTE` achieves competitive F1 score on the CoNLL03 dataset (Tjong Kim Sang, 2002), and better than fine-tuned counterparts by a large margin on the MIT Movie (Liu et al., 2013) and Few-NERD datasets (Ding et al., 2021b) in low-resource settings.

2 Method

2.1 Low-Resource Named Entity Recognition

Given a sentence $\mathbf{X} = (x_1, x_2, \dots x_N)$ which contains N words, the task is to produce $\mathbf{Y} = (y_1, y_2, \dots y_N)$ which is the sequence of entity tags. The tag $y_i \in \mathcal{Y}$ (e.g., B-LOC, I-PER, O) denotes the type of entity for each word x_i, where $\mathcal{Y}$ is a pre-defined set of tags. We are given a low-resource NER dataset $\mathcal{D}_{\text{train}}$, where the labeled examples to each NER type (e.g., < 50) are substantially less than that in the rich-resource NER dataset. Our goal is to train an accurate NER model under this low-resource setting.

Previous methods usually treat NER as a sequence labeling task in which a neural encoder such as LSTM and BERT is used for representing the input sequence, and a softmax or a CRF layer is equipped as the output layer to get the tag sequence. Formally, as the standard fine-tuning, NER model $\mathcal{M}$ parameterized by θ is trained to minimize the cross-entropy loss over token representations $\mathbf{H} = [h_1, h_2, \dots h_N]$ that are generated from the neural encoder as follows:

$$\mathcal{L} = -\sum_{i=1}^{N} \log f_{x_i, y_i}(\mathbf{h}; \theta), \tag{1}$$

where f is the model's predicted conditional probability for golden label.

2.2 Prompt-based Text Entailment

Towards the low-resource NER task, a common way is to pre-train the neural encoder and output layer parameters with the rich-resource NER dataset. Another feasible way is to focus on the matching function learned by prototype-based network (Snell et al., 2017) or nearest neighbor classification (Yang and Katiyar, 2020). After that, a well-trained matching function can work well in the target tasks. However, since the entity category is different, the parameter for the low-resource domain cannot be transferred directly from the source domain. Moreover, the metric-based meta-learning methods assume that training and test tasks are in the same distribution but this assumption may not always be satisfied in practice (Yin, 2020).

In this work, we reformulate named entity recognition as the text entailment task. As the NER task is not a standard entailment problem, we convert NER examples into labeled entailment instances. The input includes the original sentence as premise and entity type-specific prompt as a hypothesis (i.e., template). The output is produced by an entailment classifier, predicting a label for each instance. The entailment score is the probability of a specific token at the mask position of the prompt. Then, the entity type with the top entailment score is selected as the final label. For example, given a sentence "*Seoul is the capital of South Korea.*" and a candidate "*Seoul*", we define "*Seoul is an <entity_type> entity. [MASK]*" as prompt for each entity type. Suppose the entailment score of token "*yes*" at [MASK] for <location> type is the highest of all entity types, we finally choose "*location*" as the predicted label. For training, we sample three types of negative examples (see Appendix A): false positive (i.e., replace the correct label with others), null label (i.e., replace the correct label with null), and non-entity replacement (i.e., replace golden entity with non-entity span). For example, "*Seoul is not a*

named entity. [MASK]" is one prompt of "false positive" example (i.e., the [MASK] label is *no*, and it exists entities). During training and inference, we can enumerate all possible text spans in the input sentence as named entity candidates (Cui et al., 2021). To further reduce time complexity in generating candidates by n-grams enumeration, we inject tagging labels (e.g., I-location means the tag is inside a entity) into prompts and treat words as basic units instead of text spans during training and inference. In other words, we consider prompts "*<candidate_entity_word> is the part of a <entity_type> entity. [MASK]*" (e.g., "*Seoul is the part of a location entity. [MASK]"*). As PTE treats words as basic units for decoding, it optimizes time complexity at inference to O(L), which is in line with previous NER methods. It optimizes quadratic costs at inference to linear. We also apply the Viterbi algorithm at inference, where transitions are computed on the training set (Hou et al., 2020). The computational complexity of n-grams enumeration is O(L^2), increasing quadratically with sequence length L. Overall, our method provides a unified entailment framework as the model shares the same inference pattern across different domains.

2.3 Pattern Exploiting Training Framework

The basic framework of PTE is from ADAPET (Tam et al., 2021) which is a variant of PET (Schick and Schütze, 2021a,b). Compared with PET, ADAPET uses more supervision by decoupling the losses for the label tokens and a label-conditioned MLM objective over the total original input (Tam et al., 2021). We introduce it by describing how to convert one example into a cloze-style question. The query-form in ADAPET is defined by a Pattern-Verbalizer Pair (PVP). Each PVP consists of one pattern which describes how to convert the inputs into a cloze-style question with masked out tokens, and one verbalizer which describes the way to convert the classes into the output space of tokens. The PVP can be manually generated (Sun et al., 2019; Petroni et al., 2019) or obtained by using an automatic search algorithm (Schick et al., 2020; Gao et al., 2021). After that, ADAPET obtains logits from the model $G_m(x)$. Given the space of output tokens $\mathcal{Y}$, ADAPET computes a softmax over $y \in \mathcal{Y}$, using the logits from $G_m(x)$. The final loss is shown as follows:

$$q(y|x) = \frac{\exp([\![G_m(x)]\!]_y)}{\sum_{y' \in \mathcal{Y}} \exp([\![G_m(x)]\!]_{y'})}, \quad (2)$$

$$\mathcal{L} = \texttt{Cross\_entropy}(q(y^*|x), y^*). \quad (3)$$

2.4 Cross Task and Domain Transfer

To address the challenge when few labeled examples are available, we further train the sentence encoder on the TE datasets (e.g., MNLI) and apply it to the NER task. Then, our method can perform more knowledge transfer between the rich-resource NER dataset and the low-resource NER dataset. Since there is no domain-related fully connected layer for fine-tuning, all parameters can be transferred in different domains even if the entity category does not match. Specially, we apply the text entailment method to the low-resource domain after firstly pre-training the NER model in the rich-resource domain. This process is simple but can effectively transfer label knowledge. As the output of our method is model-agnostic words (not tag index), the tag vocabulary with rich-resource and low-resource is a shared pre-trained language model vocabulary set. It allows our method to use the correlation of tags to enhance the effect of cross-domain transfer learning.

3 Experiments

We compare our methods with several baselines on both rich-resource settings and low-resource settings. We use the CoNLL2003 (Tjong Kim Sang, 2002) as the rich-resource dataset, and MIT Movie (Liu et al., 2013), Few-NERD (Ding et al., 2021b) as the cross-domain low-resource datasets. And we conduct experiments on the CoNLL03 dataset in both full and low-resource settings. The dataset statistics and experimental settings are included in Appendix B and C. The standard precision, recall, and F1 score are used for model evaluation.

3.1 Rich-Resource NER Results

We first use the whole training set of the CoNLL03 to train the model and evaluate its performance on the test set. Table 1 shows the performance of the comparison method and our model on the test set. We can find that although the potential applications of PTE is low-resource named entity recognition, it can also achieve competitive performance in rich-resource domain data sets. Compared with BERT fine-tuning reported in the previous work, the PTE

Method	Precision	Recall	F1
Wiseman and Stratos (2019)	-	-	89.94
Yang et al. (2018)	-	-	90.77
Ma and Hovy (2016)	-	-	91.21
BERT (Cui et al., 2021)	91.93	91.54	91.73
Yamada et al. (2020)	-	-	**94.30**
Template BART (Cui et al., 2021)	90.51	93.34	91.90
PTE (discrete)	91.27	91.56	91.41
PTE (soft)	92.01	92.45	**92.23**

Table 1: Model performance on the CoNLL03 test set.

Method	PER	ORG	LOC	MISC	Overall
BERT	75.71	77.59	60.72	60.39	69.62
Template BART	84.49	72.61	71.98	73.37	75.59
PTE (BERT)	85.34	72.89	73.01	74.32	**76.40**

Table 2: Cross entity type results on the CoNLL03. LOC and MISC are low-resource entity types, where PER and ORG are rich-resource entity types.

model using discrete manual design reduces the F1 by 0.32, while the PTE model using the soft prompt method design mode (Liu et al., 2021a,b) increases the F1 by 0.5. It shows that our method effectively recognizes named entities, and soft prompts can improve performance compared with manually designed prompts. More experimental results about TE patterns (§2.3) are in the Appendix D.

3.2 Cross Entity Type NER Results

Following Cui et al. (2021), we sample the number of examples corresponding to different types of entities on the CoNLL03 data training set as new training set while keep test set unchanged. Among them, "PER" and "ORG" are rich-resource entity types, and "LOC" and "MISC" are low-resource entity types. The experimental results are shown in Table 2. The results show that our method achieves better results than baselines on the low-resource entity types, thus improving overall performance. On the other hand, our method is better than fine-tuning in both cases.

3.3 Domain Transfer for Low-Resource NER

We do not use N-way K-shot setting (Yang and Katiyar, 2020; Ding et al., 2021b) which samples N categories and K examples for training in each episode because a sentence in the NER task may contain multiple entities from different types. Thus, we randomly sample training data from the MIT Movie and Few-NERD datasets to simulate low-resource scenarios and use CoNLL03 as the rich-resource dataset. As such, we have only K examples for each type of training. We choose $K \in \{10, 20, 50, 100, 200, 500\}$ for experiments to evaluate the ability of the model on training

MIT Movie (12)						
Method	**K=10**	**K=20**	**K=50**	**K=100**	**K=200**	**K=500**
Wiseman and Stratos (2019)	3.1	4.5	4.1	5.3	5.4	8.6
Ziyadi et al. (2020)	40.1	39.5	40.2	40.0	40.0	39.5
Sequence Labeling BERT	28.3	45.2	50.0	52.4	60.7	76.8
Yamada et al. (2020)	35.6	49.2	61.8	72.4	78.7	82.8
Template BART (Cui et al., 2021)	42.4	54.2	59.6	65.3	69.6	80.3
PTE (discrete)	46.9†	59.2†	66.9†	74.9†	79.9†	83.6
PTE (soft)	**47.8**†	**60.8**†	**68.1**†	**76.5**†	**83.6**†	**86.4**†
Few-NERD (8)						
Method	**K=10**	**K=20**	**K=50**	**K=100**	**K=200**	**K=500**
Wiseman and Stratos (2019)	5.2	4.1	4.7	7.8	12.3	10.1
Ziyadi et al. (2020)	35.4	48.3	51.2	51.8	53.6	55.7
Sequence Labeling BERT	50.6	59.3	61.3	61.4	62.5	66.4
Yamada et al. (2020)	51.7	60.1	62.3	61.0	62.5	66.8
PTE (discrete)	51.8	59.7	60.5	61.3	61.8	63.4
PTE (soft)	**54.2**	**61.4**	**62.3**	**62.5**	**63.6**	**67.4**

Table 3: F1 comparison of two low-resource NER datasets. We set 6 sample size K for different low-resource settings. † means a significant difference compared to Template BART ($p < .05$).

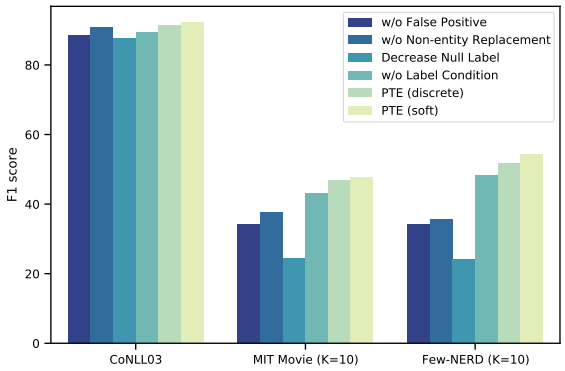

Figure 1: F1 scores with different experimental settings and model variants.

data of different sizes. The experimental results are in Table 3. The results show that when the K value is relatively small, our PTE method can be better than the fine-tuning method, and this trend decreases with the increase of K. In addition, the soft mode is also better than the discrete mode in the case of a small number of samples. Overall, our method achieves the best results on both data sets in the low-resource scenario.

3.4 Ablation Study

We conduct ablation experiments and the results are shown in Figure 1. The results show that (1) the selection of negative examples has a great impact on the performance of the model, especially the negative examples of the null label type. However, in rich-resource scenario, the gap between full setting and decreased setting is not as much as the low-resource scenario; (2) the low-resource scenario is a challenge to the model, and the results of some variants are not inconsistent where prompt-based learning may not be as good as fine-tuning; (3) label conditioning and soft mode have a consistent effect on the model. These findings highlight that it still has room left to use prompt

for effectively transferring knowledge in the case of low-resource scenario.

4 Related Work

Previous works have shown the ability of prompt-based learning under low-resource settings(Schick and Schütze, 2021a,b; Schick et al., 2020; Lester et al., 2021). Schick and Schütze (2021a) address low-resource text classification by manually designing templates as prompt-based learning in a iterative training manner. Gao et al. (2021) improve low-resource performance with well-designed templates with demonstrations. Liu et al. (2021b) apply continuous prompts for low-resource learning. Recently, some works (Ding et al., 2021b; Tong et al., 2021; Ma et al., 2021a; Chen et al., 2021) also focus on low-resource NER. In contrast, we propose to use prompt-tuning to treat NER as the TE task. Unlike traditional NER methods, we use prompt-based learning without an additional linear layer for fine-tuning. By defining different prompts, the model is able to perform well in low-resource settings, which adapts to new domains with few labeled data. In contrast to recent work which also adopts prompt-based fine-tuning for NER (Ma et al., 2021b), we show that the effectiveness of the text entailment reformulation for named entity recognition using PLMs.

5 Conclusion

In this paper, we apply prompt-based learning to low-resource named entity recognition. For token classification of NER, we reformulate it into a text entailment task. Our method transfers knowledge in different NLP tasks and domains, and performs better in low-resource scenarios. Future work includes how to apply `PTE` to other NLP tasks.

Acknowledgements

We thank the anonymous reviewers for their insightful comments and suggestions. This work is jointly supported by grants: Natural Science Foundation of China (No. 62006061,61872113,U1813215), Stable Support Program for Higher Education Institutions of Shenzhen (No. GXWD20201230155427003-20200824155011001) and Strategic Emerging Industry Development Special Funds of Shenzhen(No. JCYJ20200109113441941 and No. XMHT20190108009).

References

Samuel R. Bowman, Gabor Angeli, Christopher Potts, and Christopher D. Manning. 2015. A large annotated corpus for learning natural language inference. In *Proc. of EMNLP*, pages 632–642, Lisbon, Portugal. Association for Computational Linguistics.

Xiang Chen, Ningyu Zhang, Lei Li, Xin Xie, Shumin Deng, Chuanqi Tan, Fei Huang, Luo Si, and Huajun Chen. 2021. Lightner: A lightweight generative framework with prompt-guided attention for low-resource NER. *ArXiv preprint*, abs/2109.00720.

Leyang Cui, Yu Wu, Jian Liu, Sen Yang, and Yue Zhang. 2021. Template-based named entity recognition using BART. In *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*, pages 1835–1845, Online. Association for Computational Linguistics.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proc. of NAACL-HLT*, pages 4171–4186, Minneapolis, Minnesota. Association for Computational Linguistics.

Ning Ding, Yulin Chen, Xu Han, Guangwei Xu, Pengjun Xie, Hai-Tao Zheng, Zhiyuan Liu, Juanzi Li, and Hong-Gee Kim. 2021a. Prompt-learning for fine-grained entity typing. *ArXiv preprint*, abs/2108.10604.

Ning Ding, Guangwei Xu, Yulin Chen, Xiaobin Wang, Xu Han, Pengjun Xie, Haitao Zheng, and Zhiyuan Liu. 2021b. Few-NERD: A few-shot named entity recognition dataset. In *Proc. of ACL*, pages 3198–3213, Online. Association for Computational Linguistics.

Tianyu Gao, Adam Fisch, and Danqi Chen. 2021. Making pre-trained language models better few-shot learners. In *Proc. of ACL*, pages 3816–3830, Online. Association for Computational Linguistics.

Yutai Hou, Wanxiang Che, Yongkui Lai, Zhihan Zhou, Yijia Liu, Han Liu, and Ting Liu. 2020. Few-shot slot tagging with collapsed dependency transfer and label-enhanced task-adaptive projection network. In *Proc. of ACL*, pages 1381–1393, Online. Association for Computational Linguistics.

Guillaume Lample, Miguel Ballesteros, Sandeep Subramanian, Kazuya Kawakami, and Chris Dyer. 2016. Neural architectures for named entity recognition. In *Proc. of NAACL-HLT*, pages 260–270, San Diego, California. Association for Computational Linguistics.

Zhenzhong Lan, Mingda Chen, Sebastian Goodman, Kevin Gimpel, Piyush Sharma, and Radu Soricut. 2020. ALBERT: A lite BERT for self-supervised learning of language representations. In *Proc. of ICLR*.

Teven Le Scao and Alexander Rush. 2021. How many data points is a prompt worth? In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 2627–2636, Online. Association for Computational Linguistics.

Brian Lester, Rami Al-Rfou, and Noah Constant. 2021. The power of scale for parameter-efficient prompt tuning. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing, EMNLP 2021, Virtual Event / Punta Cana, Dominican Republic, 7-11 November, 2021*, pages 3045–3059. Association for Computational Linguistics.

J. Liu, P. Pasupat, S. Cyphers, and J. Glass. 2013. Asgard: A portable architecture for multilingual dialogue systems. In *2013 IEEE International Conference on Acoustics, Speech and Signal Processing*, pages 8386–8390.

Xiao Liu, Kaixuan Ji, Yicheng Fu, Zhengxiao Du, Zhilin Yang, and Jie Tang. 2021a. P-tuning v2: Prompt tuning can be comparable to fine-tuning universally across scales and tasks.

Xiao Liu, Yanan Zheng, Zhengxiao Du, Ming Ding, Yujie Qian, Zhilin Yang, and Jie Tang. 2021b. GPT understands, too. *ArXiv preprint*, abs/2103.10385.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Roberta: A robustly optimized bert pretraining approach. *ArXiv preprint*, abs/1907.11692.

Ruotian Ma, Xin Zhou, Tao Gui, Yiding Tan, Qi Zhang, and Xuanjing Huang. 2021a. Template-free prompt tuning for few-shot NER. *ArXiv preprint*, abs/2109.13532.

Ruotian Ma, Xin Zhou, Tao Gui, Yiding Tan, Qi Zhang, and Xuanjing Huang. 2021b. Template-free prompt tuning for few-shot NER. *CoRR*, abs/2109.13532.

Xuezhe Ma and Eduard Hovy. 2016. End-to-end sequence labeling via bi-directional LSTM-CNNs-CRF. In *Proc. of ACL*, pages 1064–1074, Berlin, Germany. Association for Computational Linguistics.

Marius Mosbach, Maksym Andriushchenko, and Dietrich Klakow. 2021. On the stability of fine-tuning BERT: misconceptions, explanations, and strong baselines. In *Proc. of ICLR*.

Matthew E. Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer. 2018. Deep contextualized word representations. In *Proc. of NAACL-HLT*, pages 2227–2237, New Orleans, Louisiana. Association for Computational Linguistics.

Fabio Petroni, Tim Rocktäschel, Sebastian Riedel, Patrick Lewis, Anton Bakhtin, Yuxiang Wu, and Alexander Miller. 2019. Language models as knowledge bases? In *Proc. of EMNLP*, pages 2463–2473, Hong Kong, China. Association for Computational Linguistics.

Timo Schick, Helmut Schmid, and Hinrich Schütze. 2020. Automatically identifying words that can serve as labels for few-shot text classification. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 5569–5578, Barcelona, Spain (Online). International Committee on Computational Linguistics.

Timo Schick and Hinrich Schütze. 2021a. Exploiting cloze-questions for few-shot text classification and natural language inference. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*, pages 255–269, Online. Association for Computational Linguistics.

Timo Schick and Hinrich Schütze. 2021b. It's not just size that matters: Small language models are also few-shot learners. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 2339–2352, Online. Association for Computational Linguistics.

Jake Snell, Kevin Swersky, and Richard S. Zemel. 2017. Prototypical networks for few-shot learning. In *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, December 4-9, 2017, Long Beach, CA, USA*, pages 4077–4087.

Chi Sun, Luyao Huang, and Xipeng Qiu. 2019. Utilizing BERT for aspect-based sentiment analysis via constructing auxiliary sentence. In *Proc. of NAACL-HLT*, pages 380–385, Minneapolis, Minnesota. Association for Computational Linguistics.

Derek Tam, Rakesh R. Menon, Mohit Bansal, Shashank Srivastava, and Colin Raffel. 2021. Improving and simplifying pattern exploiting training. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing, EMNLP 2021, Virtual Event / Punta Cana, Dominican Republic, 7-11 November, 2021*, pages 4980–4991. Association for Computational Linguistics.

Erik F. Tjong Kim Sang. 2002. Introduction to the CoNLL-2002 shared task: Language-independent named entity recognition. In *COLING-02: The 6th Conference on Natural Language Learning 2002 (CoNLL-2002)*.

Meihan Tong, Shuai Wang, Bin Xu, Yixin Cao, Minghui Liu, Lei Hou, and Juanzi Li. 2021. Learning from miscellaneous other-class words for few-shot named entity recognition. In *Proc. of ACL*, pages 6236–6247, Online. Association for Computational Linguistics.

Adina Williams, Nikita Nangia, and Samuel Bowman. 2018. A broad-coverage challenge corpus for sentence understanding through inference. In *Proc. of NAACL-HLT*, pages 1112–1122, New Orleans, Louisiana. Association for Computational Linguistics.

Sam Wiseman and Karl Stratos. 2019. Label-agnostic sequence labeling by copying nearest neighbors. In *Proc. of ACL*, pages 5363–5369, Florence, Italy. Association for Computational Linguistics.

Ikuya Yamada, Akari Asai, Hiroyuki Shindo, Hideaki Takeda, and Yuji Matsumoto. 2020. LUKE: Deep contextualized entity representations with entity-aware self-attention. In *Proc. of EMNLP*, pages 6442–6454, Online. Association for Computational Linguistics.

Hang Yan, Bocao Deng, Xiaonan Li, and Xipeng Qiu. 2019. TENER: adapting transformer encoder for named entity recognition. *ArXiv preprint*, abs/1911.04474.

Jie Yang, Shuailong Liang, and Yue Zhang. 2018. Design challenges and misconceptions in neural sequence labeling. In *Proceedings of the 27th International Conference on Computational Linguistics*, pages 3879–3889, Santa Fe, New Mexico, USA. Association for Computational Linguistics.

Yi Yang and Arzoo Katiyar. 2020. Simple and effective few-shot named entity recognition with structured nearest neighbor learning. In *Proc. of EMNLP*, pages 6365–6375, Online. Association for Computational Linguistics.

Wenpeng Yin. 2020. Meta-learning for few-shot natural language processing: A survey. *ArXiv preprint*, abs/2007.09604.

Morteza Ziyadi, Yuting Sun, Abhishek Goswami, Jade Huang, and Weizhu Chen. 2020. Example-based named entity recognition. *ArXiv preprint*, abs/2008.10570.

Datasets	Domain	# Type	# Tokens	# Train	# Dev	# Test
CoNLL03	Reuters news stories	4	21.0k	14041	3250	3453
MIT Movie	Movie reviews	12	6.0k	7820	1955	2443
Few-NERD	Wikipedia	8	4601.2k	131767	18824	37648

Table 4: Statistics of our datasets. We count the number of sentences in the training/development/test set, the number of tokens and the number of tags in datasets.

Type	Templates
Positive (Y)	<candidate> is the part of a <entity_type> entity.
False positive (N)	<candidate> is the part of a <another_entity_type> entity.
Non-entity (N)	<others> is the part of a <entity_type> entity.
Null label (Y/N)	<others> is not a name entity. <candidate> is not a name entity.

Table 5: The discrete manually-crafted templates.

Number	Patterns
Pattern#1	[HYPOTHESIS] ? </s></s> [MASK], [PREMISE] </s>
Pattern#2	" [HYPOTHESIS] " ? </s></s> [MASK], " [PREMISE] " </s>
Pattern#3	[HYPOTHESIS] ? </s></s> [MASK]. [PREMISE] </s>
Pattern#4	" [HYPOTHESIS] " ? </s></s> [MASK]. " [PREMISE] " </s>

Table 6: We list the patterns used by our method where <s> and </s> are start token and separated token.

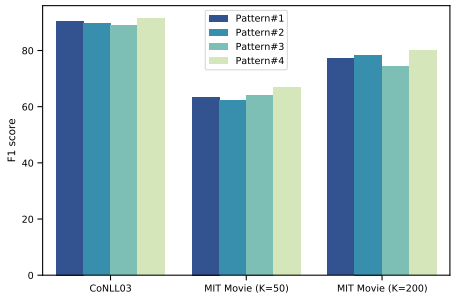

Figure 2: The performance of different modes after up to 7000 training batches. The patterns we use are from RTE task of Tam et al. (2021).

A Templates

We use the naive random sampling method and the positive-negative ratio is 1:1.5 in the low-resource scenario after sampling. As shown in Table 5, we list our templates for each example type used by PTE (discrete). Our soft prompt is to add different special tokens before the [MASK] to form a template and tune the embeddings of these tokens directly following Ding et al. (2021a) used by PTE (soft). We leave it for future work to examine whether the NER performance further improves with a more well-designed soft prompt.

B Dataset Statistics

We use the following datasets where data statistics are displayed in Table 4: (1) The CoNLL03 dataset (Tjong Kim Sang, 2002) is from the English Reuters News and consists of 4 entity types. We use the previous split in Cui et al. (2021) for our experiments. The entity types are *person*, *location*, *organization*, and *miscellaneous entities*. The sampled training dataset in §3.2 includes 1500 organization entities, 1500 person entities, 150 location entities and 150 miscellaneous entities (2) The MIT Movie dataset (Liu et al., 2013) is from queries related to movie information. The entity types are *actor*, *character*, *director*, *genre*, *plot*, *year*, *soundtrack*, *opinion*, *award*, *origin*, *quote*, and *relationship*. (3) The Few-NERD dataset (Ding et al., 2021b) is a low-resource NER dataset with a hierarchy of 8 coarse-grained and 66 fine-grained entity types. We use the coarse-grained entity in our experiments. The entity types are *location*, *event*, *building*, *art*, *product*, *person*, *organization*, and *miscellaneous entities*.

C Experimental Settings

We use the pre-trained models and codes provided by ADAPET and follow their default hyperparameter settings unless noted otherwise. The pre-trained language model of our method is BERT that is pre-trained in the MNLI datasets. We use AdamW optimizer and grid search batch size of $\{8,16,32\}$ for model training. We use grid search for learning rate from $[1e\text{-}5, 2e\text{-}5, 3e\text{-}5, 4e\text{-}5, 5e\text{-}5]$. And we grid search the optimal weight decay weight from $[0.1, 0.01, 0.005, 0.001]$. The maximum sequence length, the dropout rate, the gradient accumulation steps, the maximum training steps and the warm-up ratio are set to 256, 0.1, 16, 7000, 0.06 respectively. Early stopping is also applied based on model performance on the development set. Our models are trained with NVIDIA Tesla V100s. The verbalizer words are ["yes", "no"] and ["true", "false"]. The τ of transition probability in decoding is selected by searching with 0.05 step from 0 to 1. For sequence labeling BERT fine-tuning, we train BERT with a softmax classifier following Devlin et al. (2019), updating parameters using Adam with an initial learning rate of 1e-5, and a batch size of 32.

D Pattern Engineering

After designing templates of entity-specific hypothesis, we follow Tam et al. (2021) to define the TE patterns in Table 6 and report results across all patterns for all datasets in Figure 2. We find that the subtle difference of the prompts impacts performance, while Pattern#4 outperforms others across datasets and settings.

Key Mention Pairs Guided Document-Level Relation Extraction

Feng Jiang[1], **Jianwei Niu**[1]*, **Shasha Mo**[2]*, **Shengda Fan**[2]
[1] State Key Laboratory of Virtual Reality Technology and Systems,
School of Computer Science and Engineering, Beihang University, Beijing 100191, China
[2] School of Cyber Science and Technology, Beihang University, Beijing 100191, China
{jfeng, niujianwei, moshasha, fanshengda}@buaa.edu.cn

Abstract

Document-level Relation Extraction (DocRE) aims at extracting relations between entities in a given document. Since different mention pairs may express different relations or even no relation, it is crucial to identify key mention pairs responsible for the entity-level relation labels. However, most recent studies treat different mentions equally while predicting the relations between entities, leading to sub-optimal performance. To this end, we propose a novel DocRE model called **K**ey **M**ention pairs **G**uided **R**elation **E**xtractor (KMGRE) to directly model mention-level relations, containing two modules: a mention-level relation extractor and a key instance classifier. These two modules could be iteratively optimized with an EM-based algorithm to enhance each other. We also propose a new method to solve the multi-label problem in optimizing the mention-level relation extractor. Experimental results on two public DocRE datasets demonstrate that the proposed model is effective and outperforms previous state-of-the-art models.

1 Introduction

Relation Extraction (RE), which aims to identify the relations between entities in a given text, has been explored at the sentence level for decades (Culotta and Sorensen, 2004; Zeng et al., 2014, 2015). However, according to Yao et al. (2019), a large amount of relations can only be identified across multiple sentences in the real-world scenarios. Therefore, researchers have recently turned to extracting relations directly in documents (Zeng et al., 2020; Zhou et al., 2021; Huang et al., 2021; Ru et al., 2021).

Document-level Relation Extraction (DocRE) encounters many new challenges compared to its sentence-level counterpart. A document may include numerous entities, and the same entity may appear multiple times in different sentences. It requires the DocRE models to recognize and focus on the part of the document that has relevant context for a particular entity pair. Many previous works solve the above problems by obtaining stronger context-aware entity pair representations. There are two main ways to achieve this: the graph-based methods (Guo et al., 2019; Nan et al., 2020; Zeng et al., 2020) and the sequence-based methods (Yao et al., 2019; Zhou et al., 2021). The graph-based methods construct a document graph and then use Graph Neural Networks (GNNs) to aggregate information across nodes. Besides, as Transformer (Vaswani et al., 2017) could be regarded as a fully connected GNN, the sequence-based methods attempt to directly use Transformer-based Pre-trained Language Models (PLMs) for DocRE without graph structure. The sequence-based methods generally use strong PLMs (e.g., BERT (Devlin et al., 2019) and RoBERTa (Liu et al., 2019)) to model the input text and use different strategies to get entity pair representations, e.g., average pooling (Yao et al., 2019) and attentive pooling (Zhou et al., 2021).

However, despite these successful efforts, most existing methods still ignore the critical issue of treating different mentions equally in an entity pair. And it is at odds with the actual situation, as different mention pairs may express different relations or even no relation. For the example in Figure 1, multiple relations exist between *Genc Ruli* and *University of Tirana*, e.g., *employer* and *educated at*. These two relations can be inferred by different mention pairs of them, and at the same time, there are also several mention pairs don't express any relation. The multi-mention property of DocRE makes it difficult to establish context to relation mapping at the entity level directly. Therefore, equal treating all mentions ignores the difference between different mentions' contexts and may introduce irrelevant information to mislead model training.

*Corresponding author.

Proceedings of the 29th International Conference on Computational Linguistics, pages 1904–1914
October 12–17, 2022.

[s1] Genc Ruli (born April 11, 1958) is an Albanian politician. … [s3] Ruli holds a bachelor's degree in Economics and a bachelor's degree in Law from the University of Tirana. [s4] He holds a PhD in Economics from the Faculty of Economics, University of Tirana. [s5] Ruli is given the title Professor from the Faculty of Economics, University of Tirana. [s6] He has served as a Professor of Finance and Accounting in the Faculty of Economics, at the University of Tirana. [s7] Ruli has an extensive experience as the Minister of Finance and Economy in early 90's and as the Minister of Economy, Trade and Energy during 2005 - 2009. [s8] Ruli resigned from his position as Finance Minister on 9 November 1993, following allegations of corruption. … [s14] Ruli has written several publications in the areas of economics and public policies.

Subject: Genc Ruli **Object**: University of Tirana

Relation: employer, educated at

Figure 1: An example of multi-mention and multi-label problems from DocRED (Yao et al., 2019). Head entity *Genc Ruli* and tail entity *University of Tirana* express relations *employer* and *educated at*. This entity pair contains multiple mention pairs, and only several of them express relations. Other entities in the document is highlighted in grey.

To handle the multi-mention problem, we reformulate DocRE task as a Multiple Instance Learning (MIL) problem (Carbonneau et al., 2018) and propose a novel model called **K**ey **M**ention pairs **G**uided **R**elation **E**xtractor (KMGRE). Our approach consists of a mention-level relation extractor and a key instance classifier, which are iteratively trained to enhance each other. The relation extractor provides mention-level relation pseudo labels to help train the key instance classifier, and the key instance classifier distinguishes key mention pairs to improve relation extractor training. Those two modules can be efficiently optimized with the Expectation-Maximization (EM) algorithm (Neal and Hinton, 1998). By introducing key instances, KMGRE can effectively filter out mention pairs that do not express any relation to reduce the impact of redundant information.

Such a mention-level relation extractor suffers from the multi-label problem. It could be difficult to distinguish what kind of relation each mention pair expresses in multi-label situations, making generating the mention-level relation pseudo labels challenging. To alleviate the multi-label problem in optimizing the mention-level relation extractor, we propose to generate entity-level relation predictions by fusing mention-level predictions. Then we optimize our model's parameters with the entity-level relation labels. The contributions of this paper are summarized as follows:

- We regard the multi-mention problem in DocRE as a particular case of MIL and exptend a novel framework to directly model mention-level relations.
- We propose a new method to fuse the mention-level predictions. It could avoid the wrong guide to the model caused by false labeling mention pairs in the multi-label case.
- Experiments on two public DocRE datasets demonstrate that the proposed model is effective and outperforms previous state-of-the-art models.

2 Related Work

Sentence-level relation extraction has been explored for decades (Culotta and Sorensen, 2004; Zeng et al., 2014, 2015), but the relational facts that can only be extracted through multiple sentences cannot be handled well with traditional sentence-level relation extraction methods (Yao et al., 2019). For this reason, DocRE has attracted significant attention from researchers.

Most previous DocRE approaches focus on obtaining a strong contextual representation for each entity or entity pair. There are two main ways to achieve this: graph-based and sequence-based methods. The graph-based methods first construct a document graph and then use GNNs to model the interaction between different words and sentences. Guo et al. (2019) propose attention-guided GNNs to model full dependency trees of input documents and selectively attend to the useful dependencies. Nan et al. (2020) use a novel procedure to induce the latent document-level graph and perform multi-hop inference on the document graph. Zeng et al. (2020) construct two different levels of document graphs to aggregate information and combine the comprehensive inferential path information to infer relations.

As Transformer (Vaswani et al., 2017) could be regarded as a fully connected graph neural network, the sequence-based methods directly use Transformer-based PLMs (Devlin et al., 2019; Liu et al., 2019) to model the given text and get entity pair representations by different strategies, e.g., average pooling (Yao et al., 2019), max pooling (Li et al., 2021), and attentive pooling (Zhou et al.,

2021). However, most existing methods treat different mentions of each entity equally, which is counterintuitive, as different mention pairs may express different relations in a given document.

Some methods also consider the effect of different mentions. For instance, Christopoulou et al. (2019) put mention nodes into the document graph and use GNNs to gather different mentions' information. Li et al. (2021) propose to use convolution neural networks to capture the local mention-to-mention interactions. Eberts and Ulges (2021) propose to regard DocRE as a MIL problem and obtain entity pair representations by aggregating information from different mention pairs. However, the above methods don't consider that different mention pairs may express different relations or even no relation. And they treat different mention pairs equally in constructing the entity-level representation. Unlike previous methods, we directly model mention-level relations and further design a key instance classifier to distinguish those key mention pairs.

3 Methodology

This section introduces the proposed model KMGRE which directly extracts relations at the mention level. We first introduce the task formulation of DocRE. With a document $\mathcal{D}$ that contains a set of entities $\mathcal{E} = \{e_i\}_{i=1}^{n}$, the task is to extract relations between each entity pair (e_h, e_t), where $e_h, e_t \in \mathcal{E}$ are the head entity and the tail entity, respectively. An entity e_i may occur multiple times in the document, which could be defined as $\{m_j^i\}_{j=1}^{N_{e_i}}$, and therefore the mention pairs of (e_h, e_t) could be defined as $\boldsymbol{X} = \{(m_i, m_j) | m_i \in \{m_i^h\}_{i=1}^{N_{e_h}}, m_j \in \{m_j^t\}_{j=1}^{N_{e_t}}\}$. We denote m_i^h and m_j^t as m_i and m_j below for convenience. $\mathcal{C}$ is the set of pre-defined relation types. There exists a relation $c \in \mathcal{C}$ between e_h and e_t only if any pair of their mentions could express it.

It is intuitive to train a mention-level relation extractor and fuse its results to generate the entity-level relation label. However, instead of one mention pair being matched to one relation label, we only have the relation labels of entity pairs. Therefore, it could be challenging to train a mention-level relation extractor, directly. We propose to regard DocRE as a MIL problem and extend a novel probabilistic model to handle this issue as shown in Figure 2.

To identify the key mention pairs, we assign a binary variable $z \in \{0, 1\}$ to each mention pair, denoting whether it is responsible for the relation label of the entity pair. Inspired by EM-MIL (Luo et al., 2020), the relation label of (e_h, e_t) is generated with probability:

$$p(y_c = 1|\boldsymbol{X}, \boldsymbol{z}) = \max \{p(y_c = 1|m_i, m_j) \cdot I\left(z_{(i,j)} = 1\right)\} \quad (1)$$

where m_i and m_j are mentions of e_h and e_t, respectively. $I(\cdot)$ is the indicator function. $y_c = 1$ if this entity pair (or mention pair) contains relation c, otherwise $y_c = 0$. We then design two modules, i.e., the mention-level relation extractor and the key instance classifier. These two modules are parameterized by θ and ω, and used to estimate the distribution $p_\theta(y_c = 1|m_i, m_j)$ and $p_\omega\left(z_{(i,j)} = 1|m_i, m_j\right)$, respectively.

The goal is to jointly train the relation extractor and the key instance classifier to maximize the likelihood of the training data. Formally, the objective function is presented as below:

$$\begin{aligned}\mathcal{O}(\theta, \omega) &= \mathbb{E}[\log p_{\theta,\omega}(y_c|\boldsymbol{X})] \\ &= \mathbb{E}[\log p_{\theta,\omega}(\boldsymbol{z}, y_c|\boldsymbol{X}) - \log p(\boldsymbol{z}|\boldsymbol{X}, y_c)]\end{aligned} \quad (2)$$

Since we do not know the true distribution of $\boldsymbol{z}$, it is difficult to directly optimize Equation 2. Following previous work (Luo et al., 2020), we optimize the above objective function by maximizing its variational lower bound:

$$\begin{aligned}&\log p_{\theta,\omega}(y_c|\boldsymbol{X}) \\ =&KL(p_\omega(\boldsymbol{z}|\boldsymbol{X})||p_\theta(\boldsymbol{z}|\boldsymbol{X}, y_c)) \\ &+ \int p_\omega(\boldsymbol{z}|\boldsymbol{X}) \log \frac{p_\theta(\boldsymbol{z}, y_c|\boldsymbol{X})}{p_\omega(\boldsymbol{z}|\boldsymbol{X})} d\boldsymbol{z} \\ \geq &\int p_\omega(\boldsymbol{z}|\boldsymbol{X}) \log p_\theta(\boldsymbol{z}, y_c|\boldsymbol{X}) d\boldsymbol{z} + H(p_\omega(\boldsymbol{z}|\boldsymbol{X}))\end{aligned} \quad (3)$$

where $H(p_\omega(\boldsymbol{z}|\boldsymbol{X}))$ is the entropy of p_ω. Therefore, we use an EM-based algorithm to optimize the objective function iteratively. In the E-step, we update ω by minimizing the KL divergence between $p_\omega(\boldsymbol{z}|\boldsymbol{X})$ and $p_\theta(\boldsymbol{z}|\boldsymbol{X}, y_c)$ to obtain a tighter lower bound. In the M-step, we update θ by maximizing the lower bound. Notably, unlike the previous work (Luo et al., 2020) that directly assigns the bag's label to each instance, we further propose a new optimization method to alleviate its limitations in the case of multi-label.

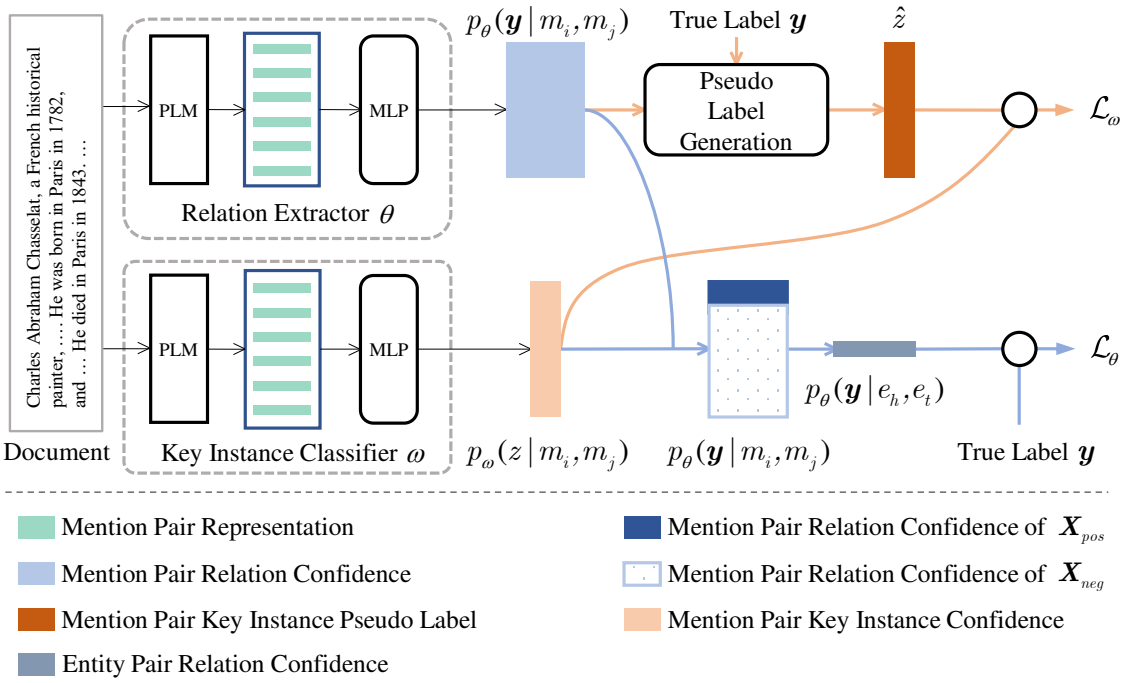

Figure 2: The overall architecture of KMGRE. We use two PLMs that do not share parameters to provide contextual embedding for the relation extractor and the key instance classifier. In the E-step, we update the parameters ω of the key instance classifier with $\mathcal{L}_\omega$; In the M-step, we update the parameters θ of the relation extractor with $\mathcal{L}_\theta$.

3.1 Parameterization

We use neural networks to parameterize the relation extractor and the key instance classifier. Specific details are described as follows.

Relation Extractor. Given an entity pair (e_h, e_t), the relation extractor generates relation probability distribution $p_\theta(y_c|m_i, m_j)$ for its mention pairs.

For a document of length ℓ, we first insert a special token "*" into every mention's start and end position. It is then fed into a PLM to obtain the contextual representation $\boldsymbol{H} \in \mathbb{R}^{\ell \times d}$ of each word, where d is the hidden dimension of the PLM. For a mention m_i, we take the representation of "*" at the start position as its embedding $\boldsymbol{h}_{m_i}$ and get its self-attention weight $\boldsymbol{A}_{m_i} \in \mathbb{R}^{H \times l}$ in H attention heads. m_j is similar to m_i. The contextual representation of mention pair (m_i, m_j) is calculated as:

$$\boldsymbol{c}^{(i,j)} = \boldsymbol{H}^\top \sum_{k=0}^{H} \frac{\boldsymbol{A}^k_{m_i} \cdot \boldsymbol{A}^k_{m_j}}{\mathbf{1}^\top \left(\boldsymbol{A}^k_{m_i} \cdot \boldsymbol{A}^k_{m_j} \right)}. \quad (4)$$

Then $\boldsymbol{c}^{(i,j)}$ is concatenated with the embedding of m_i and m_j to get the representation $\boldsymbol{x}^{(i,j)}$:

$$\boldsymbol{x}^{(i,j)} = [\boldsymbol{h}_{m_i}; \boldsymbol{h}_{m_j}; \boldsymbol{c}^{(i,j)}]. \quad (5)$$

We calculate the probability of relation c by a linear function and sigmoid activation:

$$p_\theta(y_c|m_i, m_j) = \sigma(\boldsymbol{w}_c \boldsymbol{x}^{(i,j)} + b_c) \quad (6)$$

where $\boldsymbol{w}_c \in \mathbb{R}^{3d}$ and $b_c \in \mathbb{R}$ are model parameters.

Key Instance Classifier. Since we only have the entity-level relation annotation, it is against intuition to directly train the above relation extractor. Therefore, we design this key instance classifier to generate the probability distribution $p_\omega(z_{(i,j)}|m_i, m_j)$, and assume the independence between different mention pairs. Moreover, we use this module to help train the relation extractor.

Like the above relation extractor, we use the same method to get the contextual embedding of (m_i, m_j) and concatenate it with $\boldsymbol{h}'_{m_i}$ and $\boldsymbol{h}'_{m_j}$:

$$x^{(i,j)'} = [\boldsymbol{h}'_{m_i}; \boldsymbol{h}'_{m_j}; \boldsymbol{c}^{(i,j)'}] \quad (7)$$

where the superscript $'$ means we use another PLM to get this embedding. We use two PLMs that do not share parameters to provide contextual embedding for the relation extractor and the key instance classifier, respectively, to avoid mutual interference during training.

We calculate the probability of (m_i, m_j) being a key instance by a linear function and sigmoid activation:

$$p_\omega(z_{(i,j)}|m_i, m_j) = \sigma(\boldsymbol{w}_k x^{(i,j)'} + b_k) \quad (8)$$

Algorithm 1 EM Optimization for $\mathcal{O}(\theta, \omega)$

Input: θ and ω, learning rate β, threshold control hyperparameter τ;

while not converged **do**
  for $(\boldsymbol{X}, \boldsymbol{y})$ in train set **do**
    Calculate the mention-level relation probability $p_\theta(y_c|m_i, m_j)$.
    Generate key instance pseudo label $\widehat{z}_{(i,j)}$ for all the mention pairs as Equation 9.
    Calculate the distribution of key instances $p_\omega(z_{(i,j)}|m_i, m_j)$.
    Calculate the E-step loss function $\mathcal{L}_\omega$ as Equation 11 and Equation 10.
    $\omega \leftarrow \omega - \beta \cdot \nabla_\omega \mathcal{L}_\omega$.
    Update the threshold control hyperparameter τ in Equation 13.
  end for ▷ E-step
  for $(\boldsymbol{X}, \boldsymbol{y})$ in train set **do**
    Calculate the distribution of key instances $p_\omega(z_{(i,j)}|m_i, m_j)$.
    Calculate the threshold $\widetilde{p}_\omega(z)$ as Equation 12 and Equation 13.
    Divide the mention pairs set $\boldsymbol{X}$ into $\boldsymbol{X}_{pos}$ and $\boldsymbol{X}_{neg}$ as Equation 14 and 15.
    Get the entity-level relation logit l_c as Equation 16.
    Calculate the M-step loss function $\mathcal{L}_\theta$ as Equation 19.
    $\theta \leftarrow \theta - \beta \cdot \nabla_\theta \mathcal{L}_\theta$.
  end for ▷ M-step
end while

where $\boldsymbol{w}_k \in \mathbb{R}^{3d}$ and $b_k \in \mathbb{R}$ are model parameters.

3.2 Optimization

Next, we introduce how we optimize the relation extractor and the key instance classifier to maximize the objective in Equation 2. We first train the relation extractor and the key instance classifier for several epochs before using the EM algorithm. Then at each iteration, the mention-level relation predictions and gold relation labels are first used to generate the key instance pseudo labels. After that, we update ω to minimize the KL divergence between $p_\omega(\boldsymbol{z}|\boldsymbol{X})$ and $p_\theta(\boldsymbol{z}|\boldsymbol{X}, y_c)$. Furthermore, we use the key instance predictions and gold relation labels to update θ and maximize the lower bound in Equation 3. The complete algorithm of KMGRE is shown in Algorithm 1, and the specifics are detailed below.

E-step. In the E-step, we first use the mention-level relation predictions and gold relation labels to generate the key instance pseudo labels $\widehat{\boldsymbol{z}}$ as below:

$$\widehat{z}_{(i,j)} = \begin{cases} 1, & \text{if } \exists\, c \in \mathcal{C}, \text{s.t. } y_c = 1 \wedge \\ & \quad p_\theta(y_c|m_i, m_j) \geq \overline{p}_\theta(y_c|e_h, e_t) \\ 0, & \text{otherwise} \end{cases} \tag{9}$$

where $\overline{p}_\theta(y_c|e_h, e_t) = \sum_{i,j} p_\theta(y_c|m_i, m_j)/(N_{e_h} \cdot N_{e_t})$ and y_c is the gold relation label of (e_h, e_t).

We update ω using binary focal loss (FC, (Lin et al., 2017)) as below:

$$\mathcal{L}_\omega = -\alpha_\omega (1 - p_\omega(z_{(i,j)}))^{\gamma_\omega} \log(p_\omega(z_{(i,j)})) \tag{10}$$

where α_ω and γ_ω are pre-defined hyperparameters. $p_\omega(z_{(i,j)})$ is defined below:

$$p_\omega(z_{(i,j)}) = \begin{cases} p_\omega(z_{(i,j)}|m_i, m_j), & \text{if } \widehat{z}_{(i,j)} = 1 \\ 1 - p_\omega(z_{(i,j)}|m_i, m_j), & \text{otherwise} \end{cases} \tag{11}$$

M-step. Unlike previous methods that directly label key mention pairs with the same label as entity pairs, we propose a new optimization method to alleviate the multi-label problem (e.g., the example in Figure 1 that the same entity pair may contain multiple relations). We fuse the mention-level relation results of key mention pairs to obtain the entity-level relation predictions and update θ by the entity-level relation extraction loss.

We first divide $\boldsymbol{X}$ into two different subsets $\boldsymbol{X}_{pos}$ and $\boldsymbol{X}_{neg}$ as below:

$$\overline{p}_\omega(z) = \sum_{i,j} \frac{p_\omega(z_{(i,j)})}{N_{e_h} \cdot N_{e_t}} \tag{12}$$

$$\widetilde{p}_\omega(z) = \min(\overline{p}_\omega(z) + \xi \cdot (\max\{p_\omega(z_{(i,j)})\} - \min\{p_\omega(z_{(i,j)})\}), \tau) \tag{13}$$

$$\boldsymbol{X}_{pos} = \{(m_i, m_j)|p_\omega(z_{(i,j)}) \geq \widetilde{p}_\omega(z)\} \tag{14}$$

$$\boldsymbol{X}_{neg} = \{(m_i, m_j)|p_\omega(z_{(i,j)}) < \widetilde{p}_\omega(z)\} \tag{15}$$

where $\xi > 0$ is set to control the degree of relaxation, $p_\omega(z_{(i,j)})$ means $p_\omega(z_{(i,j)}|m_i, m_j)$, and τ is a hyperparameter that increases gradually with the training process. The entity-level output logit of relation c is calculated as below:

$$l_c = \log \sum_{\boldsymbol{X}_{pos}} \exp(\boldsymbol{w}_c \boldsymbol{x}^{(i,j)} + b_c). \tag{16}$$

Following previous work (Zhou et al., 2021), we introduce a special relation class TH as the adaptive threshold and use the following loss function to update θ:

$$\mathcal{L}'_\theta = -\sum_{r \in \mathcal{P}_\mathcal{T}} \log \left(\frac{\exp(l_r)}{\sum_{r' \in \mathcal{P}_\mathcal{T} \cup TH} \exp(l_{r'})} \right) \quad (17)$$

$$\mathcal{L}''_\theta = -\log \left(\frac{\exp(l_{TH})}{\sum_{r' \in \mathcal{N}_\mathcal{T} \cup TH} \exp(l_{r'})} \right) \quad (18)$$

$$\mathcal{L}_\theta = \mathcal{L}'_\theta + \mathcal{L}''_\theta \quad (19)$$

where $\mathcal{P}_\mathcal{T}$ is the set of relations contained in (e_h, e_t) and $\mathcal{N}_\mathcal{T} = \mathcal{C} \setminus \mathcal{N}_\mathcal{T}$.

4 Experiments

4.1 Datasets and Evaluation Metrics

We evaluate our approach on two public DocRE datasets.

DWIE[1] (Zaporojets et al., 2021) is an entity-centric multi-task dataset containing 602/98/99 documents for training, validation, and testing, respectively. In the DWIE dataset, on average each entity pair contains 3.97 mention pairs. And about 26% of its entity pairs that express relations have more than one relation label.

DocRED (Yao et al., 2019) is a large scare human-annotated DocRE dataset containing 5053 documents from Wikipedia and Wikidata. As the original DocRED has a considerable amount of false-negative samples, we conduct experiments on two re-annotated versions of it, i.e., **Revisit-DocRED**[2] (Huang et al., 2022) and **Re-DocRED**[3] (Tan et al., 2022).

Following previous works, we use micro F1 and micro Ign F1 as the evaluation metrics for DocRE tasks. Ign F1 is proposed in Yao et al. (2019) with the relational facts shared by training and test sets excluded.

4.2 Baseline Models

We compare KMGRE with several RE models, e.g. CNN, LSTM, BiLSTM and Context-Aware (Sorokin and Gurevych, 2017). We also select several state-of-the-art DocRE models for comparison.

GAIN (Zeng et al., 2020) is a state-of-the-art graph-based DocRE model, which constructs two diagrams of mention level and entity level to aggregate the dependencies at different levels.

SSAN (Xu et al., 2021) takes the structural dependencies into account in the self-attention mechanism.

ATLOP (Zhou et al., 2021) proposes an adaptive threshold mechanism and optimizes it with a specific objective function and our method has a similar structure with it in implementation.

4.3 Implementation Details

Our model is implemented in PyTorch and HuggingFace's Transformers (Wolf et al., 2019)[4]. We use the uncased BERT-base (Devlin et al., 2019) as the base encoder to get contextual representation and attention weights.

For optimization, we use AdamW (Loshchilov and Hutter, 2019) with a learning rate of 5e-5 and a weight decay of 1e-5 to optimize our model. We apply a linear warmup on the first 6% steps. The focusing hyperparameters γ_ω and α_ω are set to 2 and 0.3, respectively. The threshold control hyperparameter ξ is set to 0.15 for Revisit-DocRED and 0.1 for DWIE.

We noticed in our experiments that if τ is set to a fixed high value, the model may misclassify some key mention pairs in the initial stage, which would mislead the relation extractor. Therefore, we introduce a warm-up process by calculating τ based on the steps as $\tau = 0.5 \cdot (1 - 0.999^{step})$.

4.4 Main Results

Results on DWIE. Our main results on the DWIE dataset are shown in Table 1. We can observe that our model has significant improvements in both development and test sets. In particular, KMGRE already achieves a state-of-the-art F1 score of 76.71% on the test set.

Results on DocRED. We also report the Ign F1 and F1 metrics on the Revisit-DocRED and Re-DocRED in Table 2. As seen, in the test set of Revisit-DocRED and Re-DocRED, KMGRE consistently outperforms previous methods. Notably, the performance of these models in the test set of Revisit-DocRED is much lower than reported in their original papers. This phenomenon is caused by the occurrence of many false-negative samples in the origin DocRED dataset (Huang et al., 2022).

[1] https://github.com/klimzaporojets/DWIE
[2] https://github.com/AndrewZhe/Revisit-DocRED
[3] https://github.com/tonytan48/Re-DocRED
[4] The code and training scripts will be released at https://github.com/toyfana/KMGRE.

Model	Dev		Test	
	Ign F1	F1	Ign F1	F1
CNN*	37.65	47.73	34.65	46.14
LSTM*	40.86	51.77	40.81	52.60
BiLSTM*	40.46	51.92	42.03	54.47
Context-Aware*	42.06	53.05	45.37	56.58
GAIN*	58.63	62.55	62.37	67.57
SSAN†	58.62	64.49	62.58	69.39
ATLOP	63.57	69.96	67.56	74.36
KMGRE	**65.56 ± 0.77**	**71.40 ± 0.37**	**69.94**	**76.71**

Table 1: Performance (%) on the development and test set of DWIE. We report the mean and standard deviation of F1 on the development set and test set by conducting 5 runs of training using different random seeds. The results with * are reported in Ru et al. (2021). The result with † is reported in Yu et al. (2022).

Model	Revisit-DocRED		Re-DocRED			
	Test		Dev		Test	
	Ign F1	F1	Ign F1	F1	Ign F1	F1
CNN	29.70	30.04	53.95	55.60	52.80	54.88
LSTM	31.32	31.77	56.40	58.30	56.31	57.83
BiLSTM	32.50	32.91	58.20	60.04	57.84	59.93
GAIN	41.27	41.64	71.99	73.49	71.88	73.44
SSAN	41.64	41.92	-	-	-	-
ATLOP	41.62	41.90	**73.35**	74.22	73.22	74.02
KMGRE	**42.78**	**43.16**	73.33	**74.44**	**73.39**	**74.46**

Table 2: Performance (%) on the dev/test set of Revisit-DocRED and Re-DocRED. The SSAN here uses the officially provided checkpoint based on RoBERTa-base.

Nevertheless, our model can still achieve large improvement on the test set compared to previous methods, demonstrating the effectiveness of modeling the mention-level relations.

Efficiency Comparison. We also benchmark the time and memory usage of KMGRE on a Tesla V100 GPU. Table 3 shows that our model incurs ∼22% training time and ∼63% GPU memory overhead.

Model	Memory	Training time
ATLOP-BERT-base	4849 MB	4.21 it/s
KMGRE-BERT-base	7891 MB	3.45 it/s

Table 3: Training time and memory usage on Re-DocRED.

4.5 Ablation Studies

To better understand the impact of different components of our methods, we evaluate our model by removing each component. The results are shown in Table 4.

Effectiveness of the Key Instance Classifier. To evaluate the effectiveness of the key instance classifier, we directly train a model that only contains the mention-level relation extractor in KMGRE. By turning off the key instance classifier, KMGRE could be regarded as an instance-level approach of MIL (Ilse et al., 2018). As shown in Table 4, KMGRE performs better than without the key instance classifier. It means that our key instance classifier could effectively filter out mention pairs that do not express any relation to reduce the impact of redundant information. At the same time, our KMGRE can still achieve a better classification performance than ATLOP even without the key instance classifier, which means that directly modeling the mention-level relations is more reasonable.

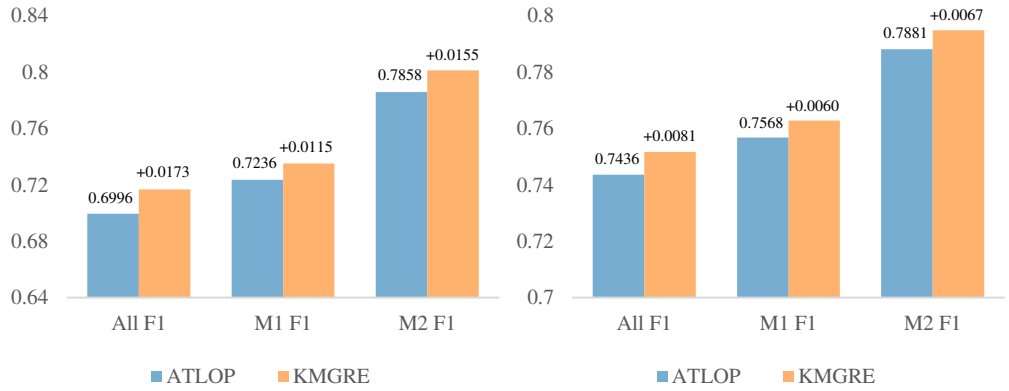

(a) Results in development set.

(b) Results in test set.

Figure 3: The results of different mention numbers in DWIE. The **M1** subset denotes those entity pairs in which head or tail entity has multiple mentions. The **M2** subset denotes those entity pairs in which both head and tail entities contain multiple mentions.

Components	Ign F1	F1
ATLOP	63.57	69.96
KMGRE	65.56	71.40
-Key Instance Classifier	64.87	70.38
-Fusion of Mention-Level Results	59.67	63.93

Table 4: An ablation study of KMGRE on DWIE.

Effectiveness of the Mention-Level Results' Fusion. We further explore the effectiveness of the mention-level results' fusion by using the same pseudo-label generation procedure as the E-step. As shown in Table 4, we could observe a significant performance decay without the fusion of mention-level results. Since direct assigning the labels of entity pairs to key mention pairs will produce a large number of wrong labeled mention pairs, it seriously misleads the mention-level relation extractor. As about 26% of the positive entity pairs have more than one relation label, this phenomenon is particularly prominent in DWIE.

4.6 Effect Analysis for Mention Number

To explore the effect of mention number in DocRE, we compare our model's relation extraction performance in different cases. Following previous work (Yu et al., 2022), we divide the DWIE dataset into several subsets according to the mention number of head/tail entity, e.g., the **M1** subset denotes those entity pairs in which head or tail entity has multiple mentions, and the **M2** subset denotes those entity pairs in which both head and tail entities contain multiple mentions.

The results in the DWIE dataset are shown in Figure 3. It can be observed that as the number of mentions increases, the relation prediction results are more accurate. It indicates that with more mentions included, the information about a particular entity is more comprehensive, which is beneficial for relation classification. Notably, our method has consistently shown improvement over the strong baseline model for all cases, even for those entities that only have a single mention. Experimental results show that KMGRE can more accurately infer the relations between entities from the context than the previous models by directly modeling mention-level relations.

4.7 Case Studies

Figure 4 shows a case study of KMGRE and the previous state-of-the-art baseline ATLOP. We could observe that the head entity *Genc Ruli* and the tail entity *University of Tirana* are mentioned multiple times in the document. And this entity pair expresses multiple relations, i.e., *educated at* and *employer*. These two relations can be inferred from mention pairs in sentences [s3] and [s5], respectively. Also, there are a considerable amount of mention pairs of (*Genc Ruli*, *University of Tirana*) that do not express any relation.

We notice that both KMGRE and ATLOP can successfully identify the *educated at* relation between *Genc Ruli* and *University of Tirana*. However, ATLOP fails to extract the *employer* relation between the same entity pair, while KMGRE deduces it successfully. It indicates that treating all mention pairs equally would introduce unrelated information to mislead the relation extrator.

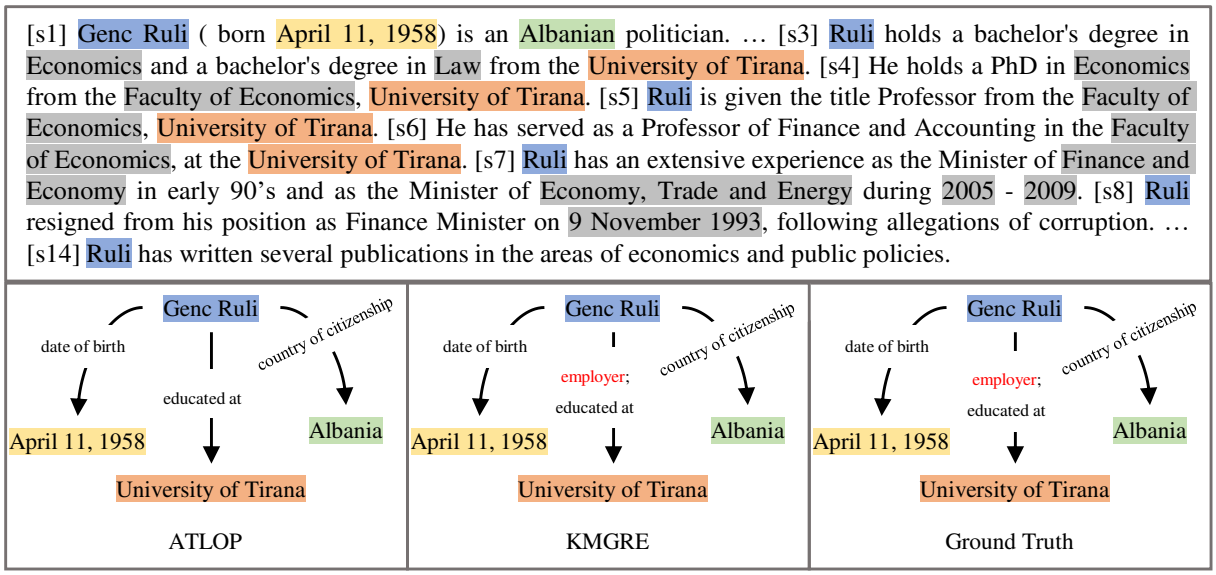

Figure 4: The case study of our proposed KMGRE and the state-of-the-art model, ATLOP (Zhou et al., 2021). The models take the document as input and predict relations among different entities in different colors. We only show a part of entities within the document and the according sentences due to the space limitation.

5 Conclusion

In this paper, we propose a new DocRE model called KMGRE for the multi-mention problem, containing a mention-level relation extractor and a key instance classifier. Our method uses the key instance classifier to identify those key mention pairs responsible for the entity pair relation label. Also, we propose a new optimization method to solve the multi-label problem in optimizing the mention-level relation extractor, as directly assigning the entity-level labels to the key instances can lead to misguidance. Experimental results on two public DocRE datasets show KMGRE outperforms previous state-of-the-art methods. The ablation study also confirms the effectiveness of our new method for optimizing the mention-level relation extractor in multi-label cases.

Acknowledgements

We thank anonymous reviewers for their responsible attitude and helpful comments. This work was supported by National Natural Science Foundation of China (62106013).

References

Marc-André Carbonneau, Veronika Cheplygina, Eric Granger, and Ghyslain Gagnon. 2018. Multiple instance learning: A survey of problem characteristics and applications. *Pattern Recognition*, 77:329–353.

Fenia Christopoulou, Makoto Miwa, and Sophia Ananiadou. 2019. Connecting the dots: Document-level neural relation extraction with edge-oriented graphs. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 4925–4936, Hong Kong, China. Association for Computational Linguistics.

Aron Culotta and Jeffrey Sorensen. 2004. Dependency tree kernels for relation extraction. In *Proceedings of the 42nd Annual Meeting of the Association for Computational Linguistics (ACL-04)*, pages 423–429, Barcelona, Spain.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, Minneapolis, Minnesota. Association for Computational Linguistics.

Markus Eberts and Adrian Ulges. 2021. An end-to-end model for entity-level relation extraction using multi-instance learning. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume, EACL 2021, Online, April 19 - 23, 2021*, pages 3650–3660. Association for Computational Linguistics.

Zhijiang Guo, Yan Zhang, and Wei Lu. 2019. Attention guided graph convolutional networks for relation extraction. In *Proceedings of the 57th Annual Meeting of the Association for Computational Lin-*

guistics, pages 241–251, Florence, Italy. Association for Computational Linguistics.

Quzhe Huang, Shibo Hao, Yuan Ye, Shengqi Zhu, Yansong Feng, and Dongyan Zhao. 2022. Does recommend-revise produce reliable annotations? an analysis on missing instances in docred. *CoRR*, abs/2204.07980.

Quzhe Huang, Shengqi Zhu, Yansong Feng, Yuan Ye, Yuxuan Lai, and Dongyan Zhao. 2021. Three sentences are all you need: Local path enhanced document relation extraction. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 2: Short Papers)*, pages 998–1004, Online. Association for Computational Linguistics.

Maximilian Ilse, Jakub M. Tomczak, and Max Welling. 2018. Attention-based deep multiple instance learning. In *Proceedings of the 35th International Conference on Machine Learning, ICML 2018, Stockholmsmässan, Stockholm, Sweden, July 10-15, 2018*, volume 80 of *Proceedings of Machine Learning Research*, pages 2132–2141. PMLR.

Jingye Li, Kang Xu, Fei Li, Hao Fei, Yafeng Ren, and Donghong Ji. 2021. MRN: A locally and globally mention-based reasoning network for document-level relation extraction. In *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*, pages 1359–1370, Online. Association for Computational Linguistics.

Tsung-Yi Lin, Priya Goyal, Ross Girshick, Kaiming He, and Piotr Dollar. 2017. Focal Loss for Dense Object Detection. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 2980–2988.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. RoBERTa: A Robustly Optimized BERT Pretraining Approach. *CoRR*, abs/1907.11692.

Ilya Loshchilov and Frank Hutter. 2019. Decoupled weight decay regularization. In *7th International Conference on Learning Representations, ICLR 2019, New Orleans, LA, USA, May 6-9, 2019*. OpenReview.net.

Zhekun Luo, Devin Guillory, Baifeng Shi, Wei Ke, Fang Wan, Trevor Darrell, and Huijuan Xu. 2020. Weakly-Supervised Action Localization with Expectation-Maximization Multi-Instance Learning. In *Computer Vision - ECCV 2020 - 16th European Conference, Glasgow, UK, August 23-28, 2020, Proceedings, Part XXIX*, volume 12374 of *Lecture Notes in Computer Science*, pages 729–745. Springer.

Guoshun Nan, Zhijiang Guo, Ivan Sekulic, and Wei Lu. 2020. Reasoning with latent structure refinement for document-level relation extraction. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 1546–1557, Online. Association for Computational Linguistics.

Radford M. Neal and Geoffrey E. Hinton. 1998. A view of the em algorithm that justifies incremental, sparse, and other variants. In Michael I. Jordan, editor, *Learning in Graphical Models*, volume 89 of *NATO ASI Series*, pages 355–368. Springer Netherlands.

Dongyu Ru, Changzhi Sun, Jiangtao Feng, Lin Qiu, Hao Zhou, Weinan Zhang, Yong Yu, and Lei Li. 2021. Learning logic rules for document-level relation extraction. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 1239–1250, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Daniil Sorokin and Iryna Gurevych. 2017. Context-aware representations for knowledge base relation extraction. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 1784–1789, Copenhagen, Denmark. Association for Computational Linguistics.

Qingyu Tan, Lu Xu, Lidong Bing, and Hwee Tou Ng. 2022. Revisiting docred - addressing the overlooked false negative problem in relation extraction. *CoRR*, abs/2205.12696.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. 2017. Attention is All you Need. In *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, December 4-9, 2017, Long Beach, CA, USA*, pages 5998–6008.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Rémi Louf, Morgan Funtowicz, and Jamie Brew. 2019. Huggingface's transformers: State-of-the-art natural language processing. *CoRR*, abs/1910.03771.

Benfeng Xu, Quan Wang, Yajuan Lyu, Yong Zhu, and Zhendong Mao. 2021. Entity structure within and throughout: Modeling mention dependencies for document-level relation extraction. In *Thirty-Fifth AAAI Conference on Artificial Intelligence, AAAI 2021, Thirty-Third Conference on Innovative Applications of Artificial Intelligence, IAAI 2021, The Eleventh Symposium on Educational Advances in Artificial Intelligence, EAAI 2021, Virtual Event, February 2-9, 2021*, pages 14149–14157. AAAI Press.

Yuan Yao, Deming Ye, Peng Li, Xu Han, Yankai Lin, Zhenghao Liu, Zhiyuan Liu, Lixin Huang, Jie Zhou, and Maosong Sun. 2019. DocRED: A large-scale document-level relation extraction dataset. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 764–777,

Florence, Italy. Association for Computational Linguistics.

Jiaxin Yu, Deqing Yang, and Shuyu Tian. 2022. Relation-specific attentions over entity mentions for enhanced document-level relation extraction. In *Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL 2022, Seattle, WA, United States, July 10-15, 2022*, pages 1523–1529. Association for Computational Linguistics.

Klim Zaporojets, Johannes Deleu, Chris Develder, and Thomas Demeester. 2021. DWIE: an entity-centric dataset for multi-task document-level information extraction. *Inf. Process. Manag.*, 58(4):102563.

Daojian Zeng, Kang Liu, Yubo Chen, and Jun Zhao. 2015. Distant supervision for relation extraction via piecewise convolutional neural networks. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 1753–1762, Lisbon, Portugal. Association for Computational Linguistics.

Daojian Zeng, Kang Liu, Siwei Lai, Guangyou Zhou, and Jun Zhao. 2014. Relation classification via convolutional deep neural network. In *Proceedings of COLING 2014, the 25th International Conference on Computational Linguistics: Technical Papers*, pages 2335–2344, Dublin, Ireland. Dublin City University and Association for Computational Linguistics.

Shuang Zeng, Runxin Xu, Baobao Chang, and Lei Li. 2020. Double graph based reasoning for document-level relation extraction. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1630–1640, Online. Association for Computational Linguistics.

Wenxuan Zhou, Kevin Huang, Tengyu Ma, and Jing Huang. 2021. Document-Level Relation Extraction with Adaptive Thresholding and Localized Context Pooling. In *Thirty-Fifth AAAI Conference on Artificial Intelligence, AAAI 2021, Thirty-Third Conference on Innovative Applications of Artificial Intelligence, IAAI 2021, The Eleventh Symposium on Educational Advances in Artificial Intelligence, EAAI 2021, Virtual Event, February 2-9, 2021*, pages 14612–14620. AAAI Press.

A Hybrid Model of Classification and Generation for Spatial Relation Extraction

Feng Wang, Peifeng Li[*], and **Qiaoming Zhu**
School of Computer Science and Technology, Soochow University, Suzhou, China
fwangsakura@stu.suda.edu.cn, {pfli, qmzhu}@suda.edu.cn

Abstract

Extracting spatial relations from texts is a fundamental task for natural language understanding and previous studies only regard it as a classification task, ignoring those spatial relations with null roles due to their poor information. To address the above issue, we first view spatial relation extraction as a generation task and propose a novel hybrid model HMCGR for this task. HMCGR contains a generation and a classification model, while the former can generate those null-role relations and the latter can extract those non-null-role relations to complement each other. Moreover, a reflexivity evaluation mechanism is applied to further improve the accuracy based on the reflexivity principle of spatial relation. Experimental results on SpaceEval show that HMCGR outperforms the SOTA baselines significantly.

1 Introduction

Spatial relation extraction focuses on identifying the relationship between two geographical entities in natural language texts. Currently, only a few studies focused on this task in the NLP community, while most studies aimed at the other tasks of relation extraction, such as temporal and causal relation extraction. However, spatial information is one kind of critical information for natural language understanding, which can benefit the downstream NLP applications, such as spatial domain query (Zhang et al., 2020), spatial reference (Yang et al., 2020) and data forecasting (Song et al., 2020).

Various kinds of schemes have been proposed to represent spatial relation. As one of the SemEval evaluation tasks, SpaceEval (Pustejovsky et al., 2015) proposes an annotation scheme adopted from ISO-space (Pustejovsky et al., 2011) and its goals include identifying and classifying items from an inventory of spatial concepts, such as topological relations, orientational relations, and motion, etc. Commonly, this task needs to extract the spatial elements and classify static and dynamic spatial relations into three types: the move link (MOVELINK), the qualitative spatial link (QSLINK), and the orientation link (OLINK). MOVELINK connects motion-events with corresponding mover-participants as a triplet of three roles *(mover, goal, trigger)*, while QSLINK and OLINK refer to the topological relation and non-topological relation between two spatial elements, respectively, and are formalized as a triplet of three roles *(trajector, landmark, trigger)*.

Sentence:
[I] set this visit up by [biking] by the [school] and having all the [children], [who] were [at] [recess], [run] out and stop me, asking what I was doing.
Relation:
{ MOVELINK
mover=I
trigger=biking
goal=null }
{ QSLINK
trajector=who
trigger=at
landmark=recess }
{ MOVELINK
mover=children
trigger=run
goal=null }
{ OLINK
trajector=children
trigger=null
landmark=school }

Figure 1: An example in SpaceEval with null-role and non-null-role.

Following previous work, we also simplified the whole task as Figure 1 and only focus on extracting QSLINK/OLINK/MOVELINK from texts. Thus, a spatial relation is defined as a triplet with three spatial types. The spatial relation can be divided into two classes: null-role and non-null-role relations. The former refers to a relation containing null-value roles, such as two MOVELINKs and one OLINK in Figure 1, while the latter (e.g., QSLINK in Figure 1) refers to a relation whose three roles are fulfilled the values extracted from sentences.

Almost all previous studies regarded spatial relation extraction as a classification task using traditional machine learning (Nichols and Botros, 2015; D'Souza and Ng, 2015) or neural network methods (Ramrakhiyani et al., 2019; Shin et al., 2020).

[*]Corresponding author

Proceedings of the 29th International Conference on Computational Linguistics, pages 1915–1924
October 12–17, 2022.

Those classification models work well on extracting those non-null-role relations due to their rich information. However, they often suffer from those null-role relations. The reason is that some information is missing in these relations. Moreover, they also cannot benefit from the knowledge of the spatial schema, such as the roles and their relations.

In the annotation stage, annotators usually not only annotate relations and relation types, but also provide a description or basis for their annotation implicitly. Therefore, we hope the model can simulate a human and provide a target sentence instead of a simple label index for understanding the spatial relation deeper. The target sentence in generation models can describe the relation between all spatial elements and it allows null slots (i.e., roles) to exist. Thus, the generation model not only can more explicitly learn the semantics of the spatial relations through such a form of the learning goal, but also can generate those null-role relations.

Moreover, the classification model and the generation model have their complementary advantages. The former usually has better performance on non-null-role relations, while the latter can introduce prior knowledge to capture the semantics of null-role relations better and its results are in a natural language expression with stronger interpretability (Jiang et al., 2021). Therefore, we combine the advantages of the classification and generation models to further capture different knowledge.

In this paper, we propose a novel hybrid model **HMCGR** (**H**ybrid **M**odel of **C**lassification, **G**eneration and **R**eflexivity) for spatial relation extraction, which contains a generation model and a classification model. Specially, the former can generate those null-role relations and the latter can extract those non-null-role relations to complement each other. Moreover, a reflexivity evaluation mechanism is applied to further improve the accuracy based on the reflexivity principle of spatial relation. Experimental results on the SpaceEval dataset shows that our HMCGR outperforms the SOTA baselines significantly.

2 Related Work

Various kinds of schemes have been proposed to represent spatial relations. SpatialML (Mani et al., 2010) characterized directional and topological relations among locations in terms of a region calculus. The SpRL task (Kordjamshidi et al., 2011) developed a semantic role labeling scheme focusing on the main roles in spatial relations. Spatial relation extraction was introduced as subtask at SemEval 2012 (Kordjamshidi et al., 2012), SemEval 2013 (Kolomiyets et al., 2013) and SemEval 2015 (Pustejovsky et al., 2015). As the Task 8 of SemEval 2015, SpaceEval proposed an annotation scheme adopted from ISO-space, and it enriched SpRL's semantics by refining the granularity. Most of previous studies were evaluated on this dataset.

The task of spatial relation extraction can be divided into traditional machine learning and neural network methods. The former highly relies on manual features or explicit syntactic structures. Nichols and Botros (2015) used a CRF layer to extract spatial elements, and then introduced SVM to classify spatial relations. D'Souza and Ng (2015) proposed a Sieve-based model where various kinds of manual features are generated by a greedy feature selection technique. Salaberri et al. (2015) introduced external knowledge as a supplement to spatial information, in which WordNet and PropBank provided information on many spatial elements. Kim and Lee (2016) proposed a Korean spatial relation extraction model using dependency relations to find the proper elements to fulfill roles.

With the wide application of neural network, Ramrakhiyani et al. (2019) generated candidate relations by dependency parsing and classified the candidates with a BiLSTM model. Shin et al. (2020) first used BERT-CRF to extract the spatial roles and then introduced R-BERT (Wu and He, 2019) to extract the spatial relations. Besides, a few studies focused on multi-modal spatial relation extraction. For example, Dan et al. (2020) proposed a spatial BERT which gives two spatial entities and a picture to determine spatial relations.

3 HMCGR

Figure 2 shows the overall architecture of our model HMCGR. As a whole, HMCGR can be divided into four modules, i.e., candidate triplet extraction (CTE), spatial relation classification (CLS), spatial relation generation (GEN), and Reflexivity evaluation (RFX).

The module CTE is first used to extract spatial elements and spatial roles from a raw sentence to obtain candidate triplets by a BERT-CRF model. And then the candidate triplets and the raw sentence are fed to the module CLS, which uses a BERT encoder and a T5 encoder to encode the sentence, respectively, and apply a GCN (Graph

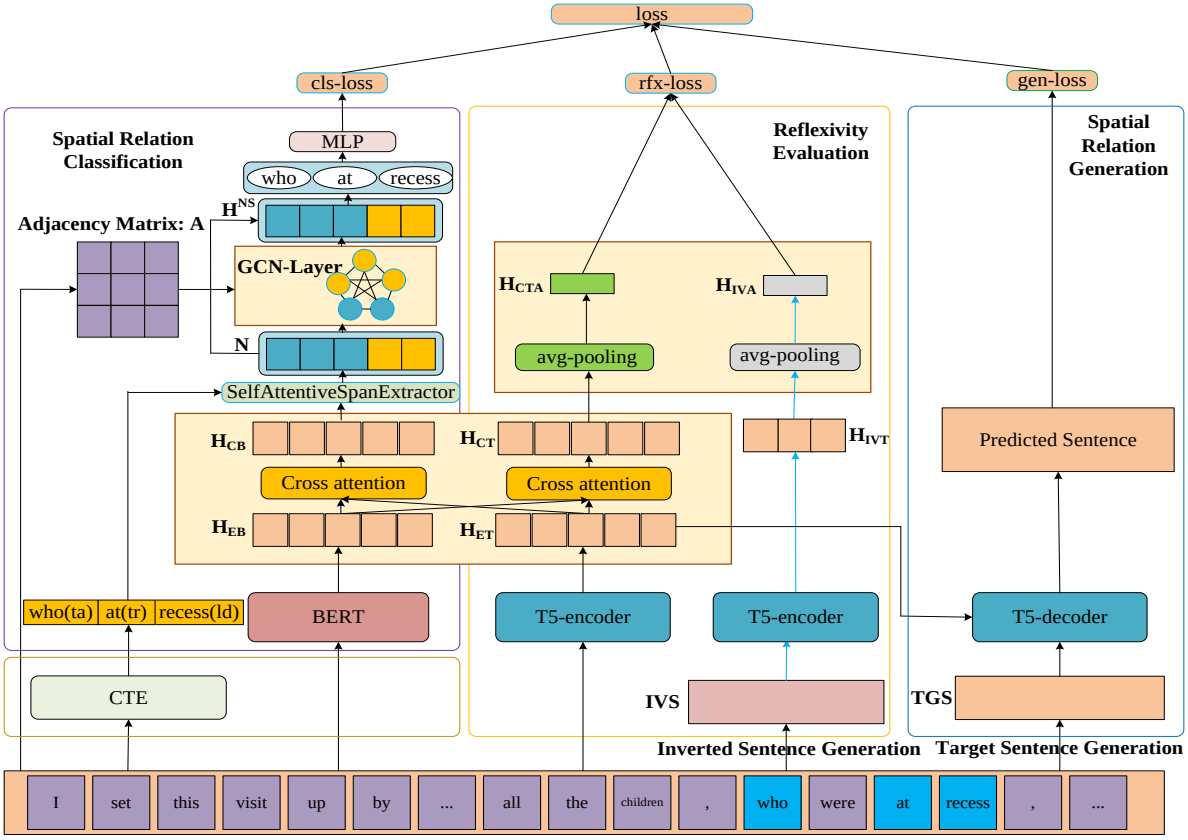

Figure 2: Overall structure of our HMCGR.

Convolutional Networks) layer to capture the sentence structure. Simultaneously, the module GEN uses a T5 decoder to generate a target sentence following a specific template, and the module RFX uses the cosine function to calculate the similarity between the original sentence and its inverted sentence to further improve the accuracy.

3.1 CTE: Candidate Triplet Extraction

Since a spatial relation is represented as a triplet with its relation type MOVELINK, OLINK or QSLINK, the first step of HMCGR is to extract candidate triplets from raw texts as much as possible. Similar to Shin et al. (2020), we also use the BERT+CRF model for spatial role extraction, as showed in Figure 3. Spatial role extraction is a task to form candidate triplets, which extracts the spatial elements from texts and then assigns a role to each extracted element.

Formally, the input is a token sequence $X = (x_1, ..., x_i, ..., x_n)$ where x_i is the i-th token in a sentence S. We feed X with the label CLS to BERT to obtain a new embedding $\boldsymbol{H}_B$ from BERT which $\boldsymbol{H}_B = [\boldsymbol{b}_1, ..., \boldsymbol{b}_i, ..., \boldsymbol{b}_n]$ where $\boldsymbol{b}_i \in R^{d_b}$ and R^{d_b} is the pre-defined spatial role set.

In Figure 3, there are two CRF layers with the input embedding $\boldsymbol{H}_B$, i.e., the Spatial Element CRF SE-CRF and the Spatial Role CRF SR-CRF. We use SE-CRF to obtain the spatial element set $SE = [se_1, ..., se_i, ..., se_m]$ in S where se_i is a spatial element, and use SR-CRF to obtain the role set $RL = [rl_1, ..., rl_i, ..., rl_m]$ for all elements where rl_i is the spatial role of the element se_i.

We simply apply a multi-task framework to train these two CRFs and they share the same BERT encoder layer. Take the sentence in Figure 1 as example, we can extract six spatial elements "children", "school", "in", "who", "at" and "recess", whose roles are *Spatial Entity*, *Place*, *Spatial Signal*, *Spatial Entity*, *Spatial Signal* and *Place*, respectively.

Due to CTE is the first stage of HMCGR, we tend to generate all possible spatial role triplets for the subsequent CLS module to achieve high recall. Hence, we first split the set SE into three subsets : 1) TM={$Trajector$, $Mover$}, 2) LG={$Landmark$, $Goal$}, and 3) TR={$Trigger$} according to their roles. Take the above elements for example, "children" and "who" belong to TM, while "school" and "recess" belong to LG and the others belong to TR.

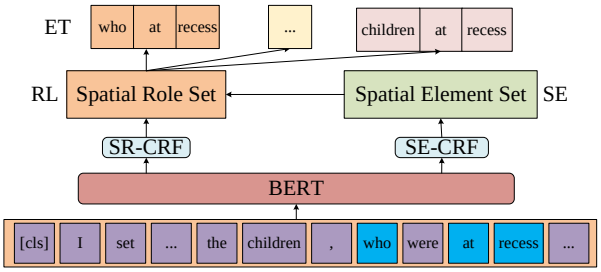

Figure 3: Overview of candidate triplet extraction.

Finally, we enumerate possible triplets as candidates following the spatial relation definition. Commonly, some triplets may have the roles with null values, as the role $trigger$ showed in Figure 1, because its element does not mention in the according sentence. If we enumerate all possible triplets including null roles as candidates, this will introduce enormous negative triplets into the candidate set and then harm the precision badly. For example, there are 27 (3^3) candidate triplets in the sentence in Figure 1, while only 4 are annotated triplets. Hence, we do not generate the triplets with null-value roles in the module CTE and the extracted candidate triplet set $ET = |TM| * |TR| * |LG|$ of the example in Figure 1 is as follows: (who, at, recess), (who, in, school), (who, at, school), (who, in, recess), (children, in, school), (children, at, school), (children, in, recess) and (children, at, recess).

3.2 CLS: Spatial Relation Classification

Following previous work, CLS is to classify the candidate triplets into four types, i.e., MOVELINK, OLINK, QSLINK, and null. If a triplet belongs to the type null, this triplet is a pseudo spatial relation. The reason that we introduce the type null to CLS is that there are lots of pseudo triplets extracted by CTE and they will harm the precision.

3.2.1 Encoding

First, we simply use BERT and T5 to encode the sequence X[1] of the sentence S to obtain the embeddings $\boldsymbol{H}_{EB} = \{\boldsymbol{eb}_1, ..., \boldsymbol{eb}_i, ..., \boldsymbol{eb}_n\}$ and $\boldsymbol{H}_{ET} = \{\boldsymbol{et}_1, ..., \boldsymbol{et}_i, ..., \boldsymbol{et}_k\}$, respectively. To make better use of the advantages of both two pre-training models, we use cross attention to represent the hidden layer state as follows. In this way, we can get the new embeddings $\boldsymbol{H}_{CB} = \{\boldsymbol{cb}_1, ..., \boldsymbol{cb}_i, ..., \boldsymbol{cb}_n\}$ and $\boldsymbol{H}_{CT} = \{\boldsymbol{ct}_1, ..., \boldsymbol{ct}_i, ..., \boldsymbol{ct}_k\}$ while the latter is used in the RFX module.

[1]We add [cls] to the start of X to obtain the sentence representation of BERT.

$$\boldsymbol{cb}_i = \text{cross\_attention}(\boldsymbol{eb}_i, \boldsymbol{et}_j) \quad (1)$$

$$\boldsymbol{ct}_i = \text{cross\_attention}(\boldsymbol{et}_j, \boldsymbol{eb}_i) \quad (2)$$

Second, we incorporate the candidate triplets extracted by CTE into the above embedding $\boldsymbol{H}_{cb}$ to enhance the representation of spatial elements. Specially, we introduce the SelfAttentiveSpanExtractor in AllenNLP to obtain the latent representation of three spatial roles $\boldsymbol{H}_{tm}$, $\boldsymbol{H}_{lg}$ and $\boldsymbol{H}_{tr}$ as follows.

$$\boldsymbol{H}_y = \sum_{i=y_{start}}^{y_{end}} (\boldsymbol{W}_{y_i} \cdot \boldsymbol{cb}_i) \quad (3)$$

where $y \in \{tm, lg, tr\}$. y_{start} and y_{end} represent the start and end position of a spatial element, respectively, and $\boldsymbol{W}_{y_i}$ are learnable parameters. Besides, since BERT maybe splits a word into multiple word-pieces, we also use SelfAttentiveSpanExtractor to obtain word-level representation.

3.2.2 Spatial GCN

Most previous work ignored the function of demonstrative pronouns in spatial relation extraction. However, those pronouns can participate in various spatial relations. Inspired by Phu and Nguyen (2021) in casual relation extraction, to capture the relationship between sentences and spatial roles, and make better use of sentence structure and anaphora, we introduce a spatial graph $G = \{N, E\}$ to CLS, where the node set $N = X \cup SE$, which are defined in subsection 3.1. We initialize four adjacency matrices ($\boldsymbol{A}^B$, $\boldsymbol{A}^E$, $\boldsymbol{A}^C$, $\boldsymbol{A}^D$) to represent four edge types in our graph G as follows, where $\boldsymbol{A}^y = \{a^y_{1,1}, ..., a^y_{i,j}, ..., a^y_{|N|,|N|}\}$ ($y \in \{B, E, C, D\}$).

Sentence Boundary Edge: Intuitively, relevant contextual information between the spatial elements within a sentence is helpful for this task. Hence, we create an undirected edge between two nodes if they are in the same sentence. Formally, we set $a^B_{i,j} = a^B_{j,i} = 1$ if the nodes n_i and n_j ($n_i, n_j \in N$) in the same sentence; otherwise, 0.

Spatial Element Edge: The intersections between the spatial elements and their containing tokens maybe share some useful information. Therefore, we create a spatial element edge between the spatial element and its token. Formally, we set $a^E_{i,j} = 1$ if n_i contains n_j; otherwise, 0.

Coreference Edge: According to our statistics, about 20% of the spatial relations in SpaceEval are participated by demonstrative pronouns. Hence,

we construct an edge from two nodes if one can reference the other. Formally, we set $a^C_{i,j} = 1$ if n_i and n_j are coreferential; otherwise, 0.

Dependency Edge: Following previous work in NLP, we also create an edge if two nodes have the same parent node in the dependency tree. Formally, we set $a^D_{i,j} = a^D_{j,i} = 1$ if n_i and n_j has the same parent node in a dependency tree. Besides, we utilize SpaCy [2] to extract the dependency trees and coreference chains.

Due to the importance of different type edges, we conduct four learnable weight matrices $\boldsymbol{W}^y = \{w^y_{1,1}, ..., w^y_{i,j}, ..., w^y_{|N|,|N|}\}$ $(y \in \{B, E, C, D\})$ to merge four type edges by their weights to an adjacency matrix $\boldsymbol{A} = \{a_{1,1}, ..., a_{i,j}, ..., a_{|N|,|N|}\}$ as follows.

$$a_{i,j} = \sum_{y \in \{B,E,C,D\}} (w^y_{i,j} \cdot a^y_{i,j}) \tag{4}$$

Finally, we can easily construct the graph G and formulate GCN for spatial information fusion to obtain its representation $\boldsymbol{H}^{NS}$ as follows.

$$\boldsymbol{H}^{NS} = \mathrm{GCN}(G, N) \tag{5}$$

3.2.3 Classification

By recording the node identifier of the spatial role in the currently processed triplet, we can get the latent representation of the spatial role in NS. Inspired by the idea of ResNet (He et al., 2016), we concat the BERT hidden state $\boldsymbol{H}_y$ ($y \in \{tm, lg, tr\}$) and the representation of the GCN nodes $\boldsymbol{H}^{NS}_y$ as the final feature of the spatial roles as follows.

$$\boldsymbol{H}^{'}_y = [\boldsymbol{H}_y; \boldsymbol{H}^{NS}_y] \tag{6}$$

where $\boldsymbol{H}^{NS}_y$ represents the latent representation of the spatial roles in $\boldsymbol{H}^{NS}$. Finally, a multi-layer perceptron (MLP) is to classify the spatial relations and we calculate the cross-entropy loss as follows.

$$y_{rel} = \mathrm{MLP}([\boldsymbol{H}^{'}_{tm}; \boldsymbol{H}^{'}_{tr}; \boldsymbol{H}^{'}_{lg}]) \tag{7}$$

$$L_{cls} = - \sum_{(tm,tr,lg) \in ET} \log \mathrm{P}(rel|tm, tr, lg) \tag{8}$$

where ET is the triplet set mentioned in subsection 3.1 and rel is the relation of the triplet.

[2] https://spacy.io/

3.3 GEN: Spatial Relation Generation

To reduce negative triplets in the CTE module, we only enumerate candidate triplets without null roles. This strategy can help CLS improve its precision. However, it also cannot extract those null-role relations. Our statistics on the SpaceEval dataset show 20% of the annotated spatial relations have a null role. Hence, how to extract those null-role relations still is a challenge. To address this issue, we introduce a spatial relation generation module GEN to extract those null-role relations. Hence, HMCGR contains a classification and a generation model, and they can complement each other to address their shortcomings.

We introduce the pre-trained generation model T5 to our GEN, due to its excellent performance on many NLP applications (Colin et al., 2020). Normally, there are two T5-decoding methods that can be used in our task, i.e., triplet or a normal sentence. In our experiments, we found that a structure normal sentence is suitable for the target sentence of T5, which contains the following three parts.

Referential Phrase Prefix: To better use the coreference relation, we add a phrase with referential meaning to the target sentence and put this phrase in the beginning of the target sentence to let our GEN use this useful information.

Relation Name: To get the type of spatial relation, we designed a slot of spatial relation name for the target sentence.

Relation Explanation: To decode spatial relations more quickly and conveniently, we design a generate structured sentence with <pad> spatial role slots as our target sentence.

Specifically, the form of target sentence is as follows: "The token "$pronoun$" stands for "$noun$", and $< pad > qslink < pad >$ can be describe as following : the first element is $< pad > tm < pad >$, the trigger is $< pad > tr < pad >$, and the second element is $< pad > lg < pad >$." Take the candidate triplet (who, at, recess) as an example, we generate the following target statement TGS for T5:"The token "who" stands for "children", and $< pad >$ qslink $< pad >$ can be describe as following : the first element is $< pad >$ who $< pad >$, the trigger is $< pad >$ at $< pad >$, and the second element is $< pad >$ recess $< pad >$.".

We feed a sentence representation $\boldsymbol{H}_{ET}$ into T5-decoder and obtain a target sentence following the format of TGS, which can be translated into the form $Relation(tm, tr, lg)$. It is worth noting that

one of tm, tr and lg may be null, and we can obtain those null-role relations. Finally, T5 generates a token or phrase for each output position using softmax and then we can get the target sentence and the cross-entropy loss L_{gen} is defined as follows.

$$L_{gen} = \mathrm{T5}_{\mathrm{decoder}}(TGS, \boldsymbol{H}_{ET}) \quad (9)$$

3.4 RFX: Reflexivity Evaluation

Our CLS and GEN can extract spatial relations from different perspectives and complement each other effectively. However, the performance of GEN is still lower than CLS, because it suffers from the limited training data and the high rate of negative and positive instances in this task.

Most spatial relations have the attribute of reflexivity due to their nature. For example, "A in B" equals to "B out of A" in spatial relation. According to the reflexivity of spatial relation, we design a similarity-based reflexivity evaluation mechanism to help GEN improve its performance. RFX first creates an inverted sentence IVS according to the original sentence S and a candidate triplet et, and then uses the cosine function to calculate the similarity of their embeddings. If two sentences are similar, the candidate triplet et will be regarded as a spatial relation with high probability.

For a sentence S and a candidate triplet $et = (tm, tr, lg)$, we first exchange the positions of two participants tm and lg in S, and then replace tr with its antonym from an antonym dictionary. If tr has more than one antonym, we randomly select one. The original sentence S and the inverted sentence IVS are fed to a T5-encoder to obtain the embeddings $\boldsymbol{H}_{CT}$ using cross attention and $\boldsymbol{H}_{IVT}$, respectively. The avg-polling is applied to the above two embeddings to capture their global features as follows.

$$\boldsymbol{H}_{CTA} = \mathrm{avgpooling}(\boldsymbol{H}_{CT}) \quad (10)$$

$$\boldsymbol{H}_{IVA} = \mathrm{avgpooling}(\boldsymbol{H}_{IVT}) \quad (11)$$

Finally, we design the spatial semantic loss $rfx - loss$ using a cosine similarity as follows.

$$L_{rfx} = 1 - \cos(\boldsymbol{H}_{CTA}, \boldsymbol{H}_{IVA}) \quad (12)$$

3.5 Joint Training and Decoding

In the training step, we train the classification model CLS and the generation model GEN together. To sum up, the overall loss L of our model HMCGR consists of three parts as follows.

Tool/Parameter	Version/Value
Pytorch	1.7.0+cu110
Spacy	2.1.0
Allennlp	2.6.0
dgl-cu110	0.6.1
Learning rate	2e-5
Batch size	4
Random seed	1024
Hidden size of pre-training model	768
Optimizer	AdamW

Table 1: Key parameters and tools used in our model.

Model	P	R	F1
BERT+CRF	88.1	91.2	89.1

Table 2: The results of spatial role extraction.

$$L = L_{cls} + L_{rfx} + L_{gen} \quad (13)$$

Finally, the spatial relations are extracted by two models, i.e., CLS and GEN. The final spatial relation set is the union of their results. Besides, the module RFX is an effective auxiliary task to help GEN improve its performance.

4 Experimentation

In this section, we first introduce the experimental settings and then report the experimental results.

4.1 Experimental Settings

We evaluate our model on the latest dataset SpaceEval. According to the official statistics, there are 1110 QSLINKs, 974 MOVELINKs and 287 OLINKs. We use the standard training/development/test set following previous work (Shin et al., 2020) where the rate of the training set and the test set is 8:2. As for evaluation, we report Precision (P), Recall (R), and Micro-F1 score. We use Pytorch and Huggingface as our base tools and use the base versions of BERT and T5. The specific tool versions and key hyper-parameters are shown in Table 1.

Currently, only a few work focused on spatial relation extraction. To evaluate the effectiveness of our HMCGR, we conduct the following strong baselines for comparison:

1) **Sieve-Based** (D'Souza and Ng, 2015), which used the sieve mechanism and syntactic parse trees to enhance the features in spatial relations;

Model	QSLINK			OLINK			MOVELINK			Overall		
	P	R	F1	P	R	F1	P	R	F1	P	R	F1
Sieve-Based	12.9	28.3	17.8	**100**	31.2	47.5	24.5	56.2	34.2	45.8	38.5	41.8
WordNet	-	-	-	-	-	-	-	-	-	54.0	51.0	53.0
SpRL-CWW	**66.1**	53.8	59.4	69.1	51.7	59.1	57.1	45.1	50.4	63.6	50.1	56.1
BERT-base	45.1	58.3	50.5	71.0	69.6	70.2	62.7	61.5	62.1	62.7	59.8	61.2
HMCGR	53.5	**73.1**	**61.1**	73.1	**85.2**	**78.6**	**66.8**	**83.0**	**73.9**	**64.3**	**79.2**	**70.9**

Table 3: Performance comparison between the baselines and HMCGR on spatial relation extraction. Since BERT-base did not report the results on each category, we run their model to obtain the results (underlined).

2) **WordNet** (Salaberri et al., 2015), which used WordNet as an external knowledge to assist their task;

3) **SpRL-CWW** (Nichols and Botros, 2015), which is the SOTA traditional model using SVM and CRF classifiers on the GloVe features to extract the spatial relations;

4) **BERT-base** (Shin et al., 2020), which is the SOTA neural network model using a BERT-based neural network model on the spatial elements extraction and spatial relation extraction.

4.2 Experimental Results

The results of spatial role extraction on SpaceEval is showed in Table 2 and its performance is similar with Shin et al. (2020). In the stage of CTE, we get 3096 candidate triplets, in which 1355 triplets are positive and 1741 triplets are negative. These figures show that the number of the negative instances is more than that of the positive ones. If we use the null value to construct the candidate triplets, the large number of negative instances will harm the performance critically.

Table 3 shows the overall performance of the baselines and our HMCGR on SpaceEval. Compared with the SOTA baseline BERT-base, our HMCGR significantly improves the overall F1-score by 9.7, especially the Recall with a gain of 19.4. This result verifies the effectiveness of HMCGR, and indicates that our generation model GEN and our classification model CLS can promote each other. Moreover, the improvement comes from all three links QSLINK, OLINK, MOVELINK with the gains of 10.6, 8.4, and 11.8, respectively. This result shows that our HMCGR works well on all links. It is worth noting that our improvement mainly comes from the recall and this indicates that the generation model is helpful to recover those null-role relations.

Model	P	R	F1
BERT-base	44.5	31.7	37.0
HMCGR	46.7	40.0	43.0

Table 4: The results of spatial relation extraction on null-role relations.

Model	P	R	F1
HMCGR	**64.3**	**79.2**	**70.9**
GEN	60.4	53.1	56.5
CLS	62.0	65.5	63.7
GEN+CLS	64.1	75.2	69.2
GEN+RFX	62.2	55.1	58.8
CLS+RFX	62.0	62.5	62.2

Table 5: Ablation study on different modules.

5 Analysis

In this section, we analysis our model HMCGR on different aspects, such as null-role relations, three modules, components in CLS and errors.

5.1 Analysis on Null-role Relations

To further verify the effectiveness of our GEN, we count the null-role relations and Table 4 shows the performance of BERT-base and HMCGR. Compared with BERT-base, HMCGR improves the F1-score by 6.0, especially the significant gain on recall (+8.3). This result verifies our motivation that the generation model GEN is effective to extract those null-role relations. However, only 40.0% of null-role relations in the test set are extracted by GEN and this indicates that the null-role relation extraction has much room for improvement.

5.2 Ablation Study on Different Modules

We conduct the ablation experiments to verify the effectiveness of the modules used in HMCGR, and Table 5 shows the results of the simplified models.

The performance descents of both single GEN

Model	P	R	F1
HMCGR	**64.3**	**79.2**	**70.9**
w/o GCN	63.3	74.7	68.1
w/o CrossAtt	62.1	74.2	67.6

Table 6: Results of HMCGR and its simplified version on SpaceEval.

and CLS are very large, in comparison with the hybrid HMCGR. This result indicates a single classification or generation model maybe cannot extract those null-role and non-null-role relations simultaneously. Moreover, the performance of GEN is lower than that of CLS and the reason is that the number of non-null-role relations is twice as much as that of null-role relations. Besides, CLS works better than BERT-base and this verifies the success of our classification model. However, the performance of GEN is lower than that of BERT-base and this indicates how to apply generation model to the traditional classification tasks still is a challenge.

The combination model GEN+CLS outperforms GEN and CLS, with huge gains of 12.7 and 5.5, respectively. This indicates GEN and CLS can boost each other to improve THE F1-score, especially the recall. In the SpaceEval dataset, 32.3% of the spatial relations belong to null-role one, in which 65.3% of the null-role relations do not have the role $trigger$ and the others do not have the role $landmark/goal$. Our GEN can recover almost 40% null-role relations and this indicates that the generation model prefers to extract those null-role relations. Moreover, the decision from two different models also can further improve the performance on different perspectives.

Compared with GEN, GEN+RFX improves the F1-score by 2.3 with the gains in both the precision and recall. This indicates that our reflexivity evaluation mechanism RFX can not only help the generation model to extract more spatial relations, but also filter out the pseudo relations. However, the F1-score of CLS+RFX is lower than that of CLS, especially the recall. Among three relation types, only the F1-score of MOVELINK decreases from 62.5 to 61.0. The reason is that some triggers in MOVELINKs do not have an antonym (e.g., "run" and "biking") and some sentences cannot be inverted. Besides, compared with GEN+CLS, HMCGR improves the F1-score by 1.7, with a gain of 4.0 on the recall. This verifies that RFX is helpful to discover more relations in a hybrid model.

Sentence: There were already old men taking cattle out to the fields to graze.
Gold MOVELINKs:
{$mover$: cattle, $trigger$: to, $goal$: fields}
{$mover$: men, $trigger$: null, $goal$: fields}
Predicted MOVELINKs:
{$mover$: cattle, $trigger$: to, $goal$: fields}
{$mover$: men, $trigger$: to, $goal$: fields}

Table 7: Examples of the errors in GEN.

5.3 Analysis on CLS

To verify the contributions of the components in CLS, We conduct the following two simplified versions of HMCGR: 1) w/o GCN: the GCN layer is removed from HMCGR; 2) w/o CrossAtt: the cross attention is removed. That is, we only use BERT to encode sentences.

Table 6 shows the results of HMCGR and its simplified versions. If we remove the GCN layer and the cross attention, the F1-score will decrease by 2.9 and 3.3, respectively. This result indicates that T5 is helpful for BERT to represent the sentence from different perspectives. As for the GCN layer, we find out that the coreference edge is the main contributor, and more than 90% of the improvement comes from this edge type.

5.4 Error Analysis

The errors of our HMCGR mainly come from those in CTE, GEN, and entity coreference. In table 2, we can find out that 8.8% of spatial relations are missing and 11.9% of pseudo relations are introduced to the following modules by CTE.

Our statistics on the results shows that GEN often badly predicts those null-role relations when there are a non-null-role relation and a null-role relation in the same sentence. Since T5 is a sequential generation model, the generation of the next spatial relation will be affected by the relation predicted above. That is, if the previous relation is non-null-role one, the current relation tends to be non-null-role too. Take Table 7 as an example, there are two MOVELINKs in the sentence. After HMCGR has extracted the first relation MOVELINK(cattle, to, fields), it tends to predict the next one as MOVELINK(men, to, fields), instead of MOVELINK(cattle, null, fields).

Although the coreference edge is the most effective one in the graph, lots of errors derive from it due to its low performance.

6 Conclusion

In this paper, we propose a novel hybrid model HMCGR for spatial relation extraction. The generation model GEN can generate those null-role relations, while the classification model CLS can extract those non-null-role relations to complement each other. Moreover, a reflexivity evaluation mechanism is applied to further improve the accuracy based on the reflexivity of spatial relation. Experimental results on the SpaceEval dataset show that our HMCGR outperforms the SOTA baseline significantly. Our future work will focus on how to extract those null-role relations effectively.

Acknowledgements

The authors would like to thank the three anonymous reviewers for their comments on this paper. This research was supported by the National Natural Science Foundation of China (Nos. 61836007, 62276177 and 62006167), and Project Funded by the Priority Academic Program Development of Jiangsu Higher Education Institutions (PAPD).

References

Raffel Colin, Shazeer Noam, Roberts Adam, Lee Katherine, n Narang Shara, Matena Michael, Zhou Yanqi, Li Wei, and Liu Peter J. 2020. Exploring the limits of transfer learning with a unified text-to-text transformer. *JMLR*, 21(140):1–67.

Soham Dan, Hangfeng He, and Dan Roth. 2020. Understanding Spatial Relations through Multiple Modalities. *CoRR*, abs/2007.09551.

Jennifer D'Souza and Vincent Ng. 2015. Sieve-Based Spatial Relation Extraction with Expanding Parse Trees. In *EMNLP*, pages 758–768.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. 2016. Deep residual learning for image recognition. In *IEEE CVPR*, pages 770–778.

Feng Jiang, Yaxin Fan, Xiaomin Chu, Peifeng Li, and Qiaoming Zhu. 2021. Not Just Classification: Recognizing Implicit Discourse Relation on Joint Modeling of Classification and Generation. In *EMNLP*, pages 2418–2431.

Bogyum Kim and Jae Sung Lee. 2016. Extracting Spatial Entities and Relations in Korean Text. In *COLING*, pages 2389–2396.

Oleksandr Kolomiyets, Parisa Kordjamshidi, Marie Francine Moens, and Steven Bethard. 2013. Semeval-2013 task 3: Spatial role labeling. In *SemEval*, pages 255–262.

Parisa Kordjamshidi, Steven Bethard, and Marie-Francine Moens. 2012. SemEval-2012 task 3: Spatial role labeling. In *SemEval*, pages 365–373.

Parisa Kordjamshidi, Martijn Van Otterlo, and Marie Francine Moens. 2011. Spatial role labeling: Towards extraction of spatial relations from natural language. *ACM Transactions on Speech and Language Processing*, 8(3):Article No. 4:1–36.

Inderjeet Mani, Christy Doran, Dave Harris, Janet Hitzeman, Rob Quimby, Justin Richer, Ben Wellner, Scott Mardis, and Seamus Clancy. 2010. SpatialML: annotation scheme, resources, and evaluation. *Language Resources and Evaluation*, 44(3):263–280.

Eric Nichols and Fadi Botros. 2015. SpRL-CWW: Spatial Relation Classification with Independent Multiclass Models. In *SemEval*, pages 895–901.

Minh Tran Phu and Thien Huu Nguyen. 2021. Graph Convolutional Networks for Event Causality Identification with Rich Document-level Structures. In *NAACL-HLT*, pages 3480–3490.

James Pustejovsky, Parisa Kordjamshidi, Marie-Francine Moens, Aaron Levine, Seth Dworman, and Zachary Yocum. 2015. SemEval-2015 Task 8: SpaceEval. In *SemEval*, pages 884–894.

James Pustejovsky, Jessica L Moszkowicz, and Marc Verhagen. 2011. Using ISO-Space for annotating spatial information. In *Proceedings of the International Conference on Spatial Information Theory*.

Nitin Ramrakhiyani, Girish K. Palshikar, and Vasudeva Varma. 2019. A Simple Neural Approach to Spatial Role Labelling. In *ECIR*, pages 102–108.

Haritz Salaberri, Olatz Arregi, and Beñat Zapirain. 2015. IXAGroupEHUSpaceEval: (X-Space) A WordNet-based approach towards the Automatic Recognition of Spatial Information following the ISO-Space Annotation Scheme. In *SemEval*, pages 856–861.

Hyeong Jin Shin, Jeong Yeon Park, Dae Bum Yuk, and Jae Sung Lee. 2020. BERT-based Spatial Information Extraction. In *SPLU*, pages 10–17.

Chao Song, Youfang Lin, Shengnan Guo, and Huaiyu Wan. 2020. Spatial-temporal synchronous graph convolutional networks: A new framework for spatial-temporal network data forecasting. In *AAAI*, pages 914–921.

Shanchan Wu and Yifan He. 2019. Enriching pre-trained language model with entity information for relation classification. In *CIKM*, pages 2361–2364.

Tsung-Yen Yang, Andrew S Lan, and Karthik Narasimhan. 2020. Robust and Interpretable Grounding of Spatial References with Relation Networks. *arXiv preprint arXiv:2005.00696*.

Lei Zhang, Runze Wang, Jingbo Zhou, Jingsong Yu, Zhenhua Ling, and Hui Xiong. 2020. Joint Intent Detection and Entity Linking on Spatial Domain Queries. In *Findings of EMNLP 2020*, pages 4937–4947.

Mining Health-related Cause–Effect Statements with High Precision at Large Scale

Ferdinand Schlatt [⋆] **Dieter Bettin** [‡] **Matthias Hagen** [⋆] **Benno Stein** [†] **Martin Potthast** [§]

[⋆]Martin-Luther-Universität Halle-Wittenberg [†]Bauhaus-Universität Weimar
[‡]Westfälische Wilhelms-Universität Münster [§]Universität Leipzig

Abstract

An efficient assessment of the health relatedness of text passages is important to mine the web at scale to conduct health sociological analyses or to develop a health search engine. We propose a new efficient and effective termhood score for predicting the health relatedness of phrases and sentences, which achieves 69% recall at over 90% precision on a web dataset with cause–effect statements. It is more effective than state-of-the-art medical entity linkers and as effective but much faster than BERT-based approaches. Using our method, we compile the Webis Medical CauseNet 2022, a new resource of 7.8 million health-related cause–effect statements such as "Studies show that stress induces insomnia" in which the cause ('stress') and effect ('insomnia') are labeled.

1 Introduction

Health sociology studies the interaction of society with health. An important subject is how consumers obtain and perceive health information. Since the web and search engines are among the most important sources, the quality of online health information has been studied so frequently in recent decades that three systematic reviews have been conducted (see Table 1). However, many of the individual studies address only a single medical condition and almost all were conducted manually. On average, only about 50–100 web pages were analyzed per study, sometimes only a single handpicked one, and never more than 1,524 pages.

Studying larger portions of the online health domain requires the automation of various acquisition tasks: (1) the discovery of health-related websites and web pages, (2) the identification and extraction of health-related statements from these pages, and (3) the attribution of health-related statements to authoritative sources (e.g., for fact-checking). While the first and third steps have been and continue to be the subject of ongoing research, the second step has received much less attention.

Systematic Review	Studies	Websites / Web Pages				
		Min	Max	Mean	Stddev	Total
Eysenbach et al. (2002)	79	3	1,147	100.5	157.7	7,836
Zhang et al. (2015)	165	3	388	78.5	73.4	12,870
Daraz et al. (2018)	157	1	1,524	50.3	133.9	7,891

Table 1: Sizes of systematic reviews of online health information studies. Some studies are part of multiple reviews; most do not distinguish websites and web pages.

To minimize manual data cleansing, we view the extraction of health-related statements as a precision-oriented task. And since a significant portion of consumers' health information needs ask about causes and effects (Bondarenko et al., 2022), we focus on health-related cause–effect statements (e.g., 'smoking causes cancer'). Extracting cause–effect statements in general has been thoroughly investigated in the past and several approaches extract them from web text efficiently and effectively (Yang et al., 2022). However, the extracted statements are usually not assigned to a specific domain.

Our contributions are: (1) A new approach for a high-precision assessment of a phrase's health relatedness (Section 3), which is more effective than state-of-the-art medical entity linkers and on par with BERT-based models but far more efficient (Section 4). (2) The Webis Medical CauseNet 2022, a web-scale resource of health-related cause–effect statements (Section 5).[1]

2 Related Work

The impact of online health information on consumers has attracted the interest of the health sociology research community. For example, user surveys examine how consumers perceive online health information (Diaz et al., 2002), e-health services (Andreassen et al., 2007), or the quality of online health information (Sun et al., 2019).

[1]All our code and data to reproduce the experiments as well as the resource are publicly available under a permissive license: https://github.com/webis-de/COLING-22

Proceedings of the 29th International Conference on Computational Linguistics, pages 1925–1936
October 12–17, 2022.

Information quality appears to be the most studied characteristic from a health sociology perspective. Numerous studies systematically analyzed the quality of websites related to specific topics such as orthodontics (Jiang, 2000) or performance-enhancing drugs (Brennan et al., 2013), but more general studies with limitations to specific parts of the web are also common. Examples include studies of dietary advice (Cooper et al., 2012) or the misinterpretation (Yavchitz et al., 2012) and exaggeration (Sumner et al., 2014) of clinical trial results in online news. Recent studies also targeted web search snippets (Bondarenko et al., 2021) and social media (Suarez-Lledo and Alvarez-Galvez, 2021), particularly health misinformation on Twitter (Broniatowski et al., 2018; Bal et al., 2020).

To identify health-related web content, most previous work has focused on whole-page classification, e.g., using medical vocabularies to classify news articles (Watters et al., 2002; Zheng et al., 2002) or using convolutional neural networks for Reddit posts (Gkotsis et al., 2017). But there has been little work on classifying shorter passages of text as health-related (e.g., phrases), although this would allow for more fine-grained analyses at the statement level rather than at the whole-page level. Keyword extraction and automatic ontology creation with the goal of extracting prototypical words for a given domain are perhaps the most closely related tasks. For example, the C-value/NC-value method uses term frequencies to extract multi-word domain terms from a corpus (Frantzi et al., 2000). Its reliance on syntactic sentence structure, however, renders it inapplicable at the phrase level.

Contrastive termhood scores, which relate term frequencies from a domain corpus to frequencies from one or more out-of-domain corpora, can be applied more straightforwardly. These include $tf \cdot idf$-inspired measures (Basili et al., 2001; Kim et al., 2009), measures estimating how exclusive a term is for a domain (Ahmad et al., 1999; Park et al., 2008), and combinations or extensions thereof (Wong et al., 2007; Bonin et al., 2010). We transfer contrastive termhood scoring to measuring health relatedness of phrases (but also sentences) and compare it with the medical entity linkers cTakes (Savova et al., 2010), MetaMap (Aronson, 2001), QuickUMLS (Soldaini and Goharian, 2016), and ScispaCy (Neumann et al., 2019), as well as BERT-based classifiers (Devlin et al., 2019).

3 Measuring Health Relatedness

Assessing whether a phrase is health-related can be difficult without context; in particular for homonymous (same surface form, different meaning) or polysemous (same surface form, different sense) words. For instance, 'cancer' may refer to a health-related malignant tumor, but also to the zodiac sign, which is unlikely to appear in a health-related context. As such, the task of assessing a phrase's health relatedness can be viewed as an extension of word-sense disambiguation. Instead of the sense of a particular word, the domain of the sense of a phrase needs to be determined.

Since there are no large-scale labeled datasets for health relatedness assessment, we rely on contrastive termhood scores that use distant supervision to measure the degree of a word or concept being specific to a certain domain (Kageura and Umino, 1996). Instead of training on explicitly labeled words or phrases, contrastive termhood scores are trained on texts that are heuristically labeled. Specifically, a word's or phrase's domain specificity depends on its "prominence" in domain-specific or out-of-domain corpora. In this section, we discuss three existing contrastive termhood scores and then explain how we adapt and apply them to health relatedness assessment.

3.1 Existing Contrastive Termhood Scores

The termhood scores contrastive weight (CW) (Basili et al., 2001), term domain specificity (TDS) (Park et al., 2008), and discriminative weight (DW) (Wong et al., 2007) rely on a corpus H of domain-specific texts (in our case: health-related texts) and at least one contrastive corpus G of general or out-of-domain texts (in our case: Wikipedia). To score a term t (a word or phrase), CW, TDS, and DW use occurrence frequencies: the absolute corpus occurrence frequency $freq_C(t)$ (i.e., the absolute number of occurrences of t in corpus C), the relative corpus occurrence frequency $rel_C(t) = freq_C(t)/|C|$ (where $|C|$ denotes some appropriate variant of corpus size like number of words or n-gram occurrences), and the inverse corpora frequency $icf(t)$ defined for H and G together as

$$icf(t) = \log\left(\frac{|H| + |G|}{freq_H(t) + freq_G(t)}\right).$$

The contrastive weight CW of a term t is similar to $tf \cdot idf$ but uses the corpus-oriented frequencies:

$$\mathrm{CW}(t) = \log\left(freq_H(t) + 1\right) \cdot icf(t).$$

For the term domain specificity TDS, we unify the slightly different definitions of Ahmad et al., Park et al., and Wong et al. as

$$\mathrm{TDS}(t) = \log\left(\frac{rel_H(t)+1}{rel_G(t)+1}+1\right).$$

Finally, the discriminative weight DW was originally defined as the product of CW and a version of TDS that uses $freq_C(t)$ instead of $rel_C(t)$. Since the values of such an "unnormalized" TDS depend on corpus size, we use our above corpus-agnostic normalized version but still compute DW as

$$\mathrm{DW}(t) = \mathrm{CW}(t) \cdot \mathrm{TDS}(t).$$

3.2 Our Generalized Termhood Scores

The above termhood scores were originally meant to help augment taxonomy vocabularies or to find terms missing in a dictionary for some specific domain. As such, the input terms are assumed to be rather short and quite domain-related noun phrases from which the "best" scoring ones are to-be-added to the vocabulary. In pilot experiments for our case of assessing the health relatedness of phrases from the web, we observed that phrases like 'fracture at the base of the skull' receive very low termhood scores even though they are clearly health-related. The reason is that such longer phrases as "a whole" have quite low occurrence frequencies even in medical corpora. A first straightforward idea could be to average a phrase's individual word's termhood scores. However, for the above example with many out-of-domain or stop words, this also does not work well. We thus propose two schemes that improve on the simple average and on the original termhood scores' treatment of longer phrases as "a whole". In our schemes, we also enable the assessment to prioritize precision or recall.

Our first scheme uses a weighted average of a phrase's individual word's termhood scores (i.e., their individual health relatedness) to assess the health relatedness of a phrase. The idea is that by giving words that have a high individual health relatedness score a higher weight, more health-related phrases can be found (i.e., improved recall). Similarly, by giving words that have a low individual health relatedness score a higher weight, the precision of the assessment on the phrase level should improve. Formally, in our first scheme, we compute the termhood score of an m-word phrase as the generalized mean of the word's individual termhood scores $x_1, \ldots, x_m$:

$$M_r(x_1, \ldots, x_m) = \left(\frac{1}{m}\sum_{i=1}^{m} x_i^r\right)^{\frac{1}{r}},$$

where r is a real-valued parameter. For $r = 1$, the generalized mean corresponds to the arithmetic mean. By increasing r, the mean is biased towards the higher-valued termhood scores and vice versa. In the extreme cases of $M_{-\infty}$ or M_{∞}, the minimum or maximum x_i is returned. At the limit for r approaching 0, the generalized mean corresponds to the geometric mean (Jensen, 1998).

As our second scheme, we propose to also compute the weighted average termhood over the n-grams of a phrase. For instance, while the unigrams 'risk' and 'factor' are relatively unrelated to health, the bigram 'risk factor' certainly is health-related. The above generalized mean scheme already increases the termhood of 'risk factor' compared to a simple average but the high occurrence frequency of the bigram itself is an even better indicator of its health relatedness. Since longer n-grams usually have quite low occurrence frequencies, especially in smaller corpora, we only average the termhood scores of a phrase's uni-, bi-, or trigrams with a parameter n determining the length of the longest used n-grams. For example, the generalized score of the phrase 'cancer risk factor' based on the termhood score $s(.)$ (could be CW, TDS, or DW) with $n = 2$ then is the average of $M_r(s(\text{'cancer risk'}), s(\text{'risk factor'}))$ and $M_r(s(\text{'cancer'}), s(\text{'risk'}), s(\text{'factor'}))$.

Corpus	Language	Documents	Words
Wikipedia	mixed, layperson	12,265,374	$3.0{\cdot}10^9$
PubMed	scientific, abstracts	31,847,923	$3.8{\cdot}10^9$
PubMed Centr.	scientific, full texts	3,611,361	$5.4{\cdot}10^9$
Textbook	clinical, educational	434	$1.4{\cdot}10^7$
Encyclopedia	clinical, layperson	67,967	$9.3{\cdot}10^6$

Table 2: Characteristics of the contrastive and health-related corpora used for the termhood scores.

3.3 Contrastive and Health Domain Corpora

Table 2 shows basic characteristics of our employed corpora. We select Wikipedia[2] as our contrastive general corpus G since it covers a wide variety of domains and is easily accessible. As candidates for health corpora H, we consider and evaluate four alternatives, each with its own (dis)advantages.

[2]Dump of all English Wikipedia articles from July 1, 2021.

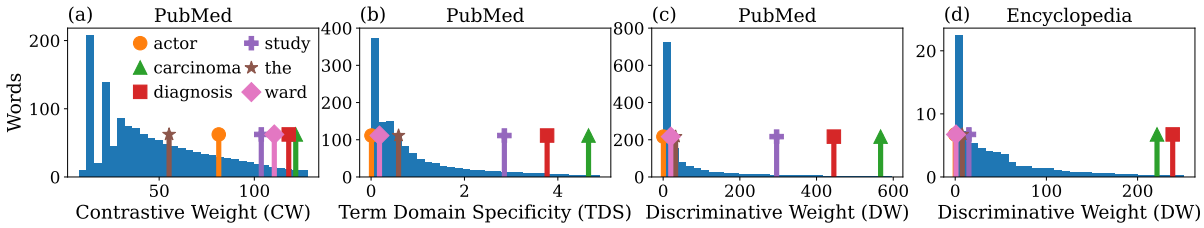

Figure 1: Termhood score frequencies of (a) CW, (b) TDS, and (c) DW on the PubMed corpus, and of (d) DW on the Encyclopedia corpus. Number of words in thousands (from the respective corpus). Example words are highlighted.

The first three health corpora use documents provided by the National Library of Medicine: (1) a dump of over 30 million MEDLINE abstracts from PubMed,[3] (2) a subset of over 3 million full-text publications from PubMed Central,[4] and (3) 434 textbooks from the textbook and monograph category of the NCBI Bookshelf.[5] While both PubMed-based corpora are large scale, their language is mainly scientific. The textbook corpus contains more clinical language, which we hypothesize to more closely match the language of health-related phrases on arbitrary web pages.

Finally, as our fourth health corpus, we crawled the entries of five consumer-oriented medical online encyclopedias.[6–10] Since they are written in layperson's terms, we assume their language to be most similar to the target language distribution used for health-related phrases on web pages.

3.4 Pilot Inspection and Comparison

To get a first impression of the scores and the corpus impact, we inspect the unigram termhood score distributions in general and for some example words. Figures 1 (a–c) show the score distributions of CW, TSD, and DW with PubMed as the domain-specific corpus H. Apparently, all scores rank the shown example out-of-domain words and the stop word lower than the shown example health-related words. However, the assessment of CW and TDS can differ substantially for specific terms. For example, 'ward' occurs frequently within texts from the PubMed corpus so that CW attributes a rather high health relatedness. At the same time, 'ward' also occurs frequently in the general domain and the lacking "exclusiveness" leads TDS to score it relatively low. Unsurprisingly, the product score DW amplifies the extremes of both scores.

As for the effect of different health corpora, Figures 1 (c) and (d) contrast the DW scores using the PubMed corpus (rather scientific language) to using the Encyclopedia corpus (rather layperson language). As an example result, the word 'study' has a comparably high termhood score using the PubMed corpus, but is ranked like a non-health-related word using the Encyclopedia corpus.

[3] https://pubmed.ncbi.nlm.nih.gov/
[4] https://www.ncbi.nlm.nih.gov/pmc/
[5] https://www.ncbi.nlm.nih.gov/books
[6] http://health.am/encyclopedia
[7] https://medlineplus.gov/encyclopedia.html
[8] https://merriam-webster.com/medical
[9] https://ucsfhealth.org (various subpages)
[10] https://www.rxlist.com/drug-medical-dictionary/article.htm

Dataset	Text Type	Health	Length	Size
CauseNet-F-Phrase	Phrase pairs	21.4%	7.2	1,000
CauseNet-P-Phrase	Phrase pairs	50.3%	3.4	1,000
CauseNet-F-Sentence	Sentences	22.4%	30.3	1,000

Table 3: Characteristics of our three annotated datasets, including the ratio of health-related entries, the average number of words per cause–effect phrase pair or sentence, and the number of entries. CauseNet-F: sampled from CauseNet-Full, CauseNet-P: sampled from CauseNet-Precision.

4 Evaluation

In this section, we evaluate our generalized termhood method on datasets of cause–effect statements and compare it to state-of-the-art medical entity linkers and BERT-based approaches.

4.1 Annotated Datasets

Table 3 depicts the general characteristics of three cause–effect datasets we sampled from the web-scale CauseNet resource (Heindorf et al., 2020), a graph of over 11 million cause–effect pairs (e.g., 'stress → insomnia') extracted from the ClueWeb12.[11] The CauseNet extraction used a two-stage approach: (1) candidate sentences were gathered using a set of lexico-syntactic patterns representing causal language, and (2) a BiLSTM-CRF model extracted cause–effect pairs from the

[11] https://www.lemurproject.org/clueweb12/

candidates. Obviously, a pair may be extracted from multiple different sentences (e.g., 'stress → insomnia' from 'stress causes insomnia' or from 'insomnia can be a result of stress'). All the sentences from which a specific pair is extracted are the *support* of that pair.

CauseNet comes in different versions: CauseNet-Full and CauseNet-Precision. In CauseNet-Full, all the extracted cause–effect pairs are contained, while CauseNet-Precision only contains pairs that were extracted by at least two different lexico-syntactic extraction patterns. The idea is that pairs supported by sentences from more patterns are less likely to be false positive causal statements.

From each CauseNet version, we randomly sampled 1,000 cause–effect pairs for our CauseNet-F-Phrase and CauseNet-P-Phrase datasets (F: full, P: precision). Note that the pairs sampled from CauseNet-Precision typically are shorter (3.4 vs. 7.2 words; cf. Table 3)—shorter phrases are more likely to be extracted by more than one pattern.

To label the health relatedness of the cause–effect pairs, we had three annotators who first labeled a kappa-test subset of 100 samples from CauseNet-F-Phrase. The achieved agreement was substantial (Cohen's kappa of 0.76), so that after a discussion of the disagreement cases, the annotators then each independently labeled disjoint thirds of the remaining CauseNet-F/P-Phrase data. Interestingly, according to the labeling, the pairs sampled from CauseNet-Precision are much more health related (50.3% vs. 21.4%; cf. Table 3).

Finally, for each of the 1,000 pairs in CauseNet-F-Phrase, we randomly selected one of the supporting sentences to complement the phrase-based dataset with a sentence-based dataset. The respective CauseNet-F-Sentence set then was also labeled with respect to health-relatedness by our three annotators. Indeed, for eleven causal phrase pairs that were labeled as non-health-related and for one pair that was labeled as health-related, the sampled supporting sentence then was labeled in the opposite way since it added important context. For example, the pair 'sound → particular feeling' was labeled as non-health related but the corresponding sentence 'Any sound that is related to trauma can trigger a particular feeling.' was labeled as health-related due to the explicit connection to trauma.

We randomly split each of the three datasets to use 80% as the training data for parameter tuning and the remaining 20% as the actual test set.

4.2 Medical Entity Linkers

We compare our generalized termhood method to the state-of-the-art medical entity linkers cTakes, MetaMap, QuickUMLS, and ScispaCy. To determine the health relatedness of a (multi-word) term by applying one of these entity linkers, we use the proportion of words that the linker matches to medical concepts in some background knowledge base like the UMLS Metathesaurus (Humphreys and Lindberg, 1993).

More formally, as the entity linking-based health relatedness of a term t, we use the the ratio $|e|/|\hat{t}|$, where $|e|$ denotes the length (in words) of the substring of t that the linker detects as mentions of some entity, and $|\hat{t}|$ denotes the length of t without the stopwords[12] that are not contained in any entity mention detected by the respective linker.

As the background knowledge base for the entity linkers we try four different options: (1) the full UMLS (a mix of medical vocabularies of varying specificity), (2) the combined UMLS subsets RxNorm and SNOMED CT (more specific clinical vocabulary), (3) UMLS restricted to the 21 most frequent semantic types (ST21pv subset) as proposed in the MedMentions entity linking dataset (Mohan and Li, 2019), and (4) the combined RxNorm and SNOMED CT restricted to ST21pv.

4.3 BERT-Based Approaches

Besides medical entity linkers, we also compare our generalized termhood method to BERT-based classifiers fine-tuned to predict whether a sequence of tokens is health-related. To test the effect of domain specific embeddings, we compare classifiers based on pre-trained BERT (Devlin et al., 2019), SciBERT (Beltagy et al., 2019), and PubMedBERT (Gu et al., 2022) models. To further fine-tune these models to the task, we first construct additional training datasets where noun phrases (or sentences) from Wikipedia are paired with noun phrases (or sentences) from the PubMed or the Encyclopedia corpus. To align with the evaluation datasets, we extract noun phrases and sentences using spaCy.[13] All models were fine-tuned with a batch size of 32 and a learning rate of $5 \cdot 10^{-5}$ for 100,000 steps on one NVIDIA A100 GPU. Due to the large corpora sizes, we used early stopping and halted training when no decrease in training loss was recorded for 15 consecutive samples (taken every 1,000 steps).

[12] English nltk stop words list.
[13] https://spacy.io/

4.4 Assessing the Health Relatedness of Cause–Effect Pairs

To assess the health relatedness of a cause–effect pair, we combine an approach's individual scores for the cause and effect phrase into one score based on which a decision about the health relatedness can be made (i.e., whether it is above some threshold). Just like in our generalized termhood scores, we use the generalized mean as the combination scheme. The extreme case of $M_{-\infty}$ then corresponds to an `AND` combination (cause and effect score need to exceed a decision threshold), while M_{∞} corresponds to an `OR` combination (cause or effect above some threshold suffice). These setups can thus be interpreted as precision or recall-oriented, respectively.

4.5 Assessing the Health Relatedness of Cause–Effect Sentences

To assess the health relatedness of a complete sentence, we basically use the different approaches as if the input was a phrase. The entity linking-based methods link the entities in the sentence and then compute the same ratio $|e|/|\hat{t}|$ but with $\hat{t}$ now being the sentence without non-linked stopwords. The BERT-based approaches also get the sentence as input and classify it as a whole as health-related or not. And also our generalized termhood scores simply treat an input sentence like a phrase.

4.6 Parameters and Optimization Criteria

We explore different hyperparameter values of the approaches in grid searches on the 80% training sets of our annotated cause–effect datasets (the other 20% are the test sets). For the termhood method, we experiment with four health corpora (PubMed, PubMed Central, Textbook, Encyclopedia) and a maximum n-gram length $n \in \{1, 2, 3\}$. For the generalized mean M_r, we try $r \in \{0, \pm1, \pm2, \pm5, \pm10, \pm\infty\}$ for combining the n-gram scores, as well as for combining the cause and effect scores of all approaches.

For the entity linkers, we explore linking against the full UMLS with either all semantic types or just the ST21pv subset, or linking just against the combined RxNorm and SNOMED CT subset with either all types or just the ST21pv subset. For the QuickUMLS and ScispaCy linkers, we try similarity thresholds in steps of 0.1 between $[0.7, 1.0]$ and $[0.6, 0.9]$, respectively, and explore small (*sm*) and large (*lg*) models for ScispaCy. Finally, the BERT-based models are fine-tuned in four variations: on sentences or on phrases from the PubMed or from the Encyclopedia corpus.

We conduct three grid searches for three different optimization scenarios. In the first two scenarios, we optimize for precision or recall and identify a parameterization and decision threshold with the best recall (or precision) that can be achieved at an operating point of a precision (or recall) of 0.9. For example, in the scenario of precision optimization, we only consider parameter combinations from the grid search and decision thresholds that achieve a precision of at least 0.9 on the training set. From these, we then only consider the parameterizations with the highest recall and from these, we select one with the highest precision. In the third optimization scenario, we target the Matthews correlation coefficient (MCC). The MCC combines the numbers of true positives (TP), true negatives (TN), false positives (FP), and false negatives (FN) as

$$\frac{TP \cdot TN - FP \cdot FN}{\sqrt{(TP{+}FP)(TP{+}FN)(TN{+}FP)(TN{+}FN)}}.$$

Since MCC generates a high score only when the majority of the positive instances and the majority of the negative instances are classified correctly, it is regarded as one of the best ways to derive one score from a binary classifier's confusion matrix (Chicco and Jurman, 2020).

4.7 Results

Table 4 shows the effectiveness of the different approaches' best parameterizations from the training sets run on the test sets of CauseNet-F-Phrase, CauseNet-P-Phrase, and CauseNet-F-Sentence. The BERT-based approaches are usually the most effective. Our generalized termhood methods often are slightly less effective but the difference to the best BERT-based approach is hardly ever statistically significant (bootstrapping with 100,000 permutations, $p < 0.05$, Bonferroni-corrected). The entity linking-based approaches, though, almost always are significantly less effective than the best BERT-based approach.

While being almost as effective as the best BERT-based approach, our generalized termhood method is substantially more efficient (see Table 5). On an AMD EPYC 7F72 processor, even without parallelization, the generalized termhood is up to 107 times faster on phrases and up to 47 times faster on the longer sentences than the BERT models. By

	Approach	Precision Optimized				Recall Optimized				MCC Optimized					
		Parameters	M_r	P	R	Parameters	M_r	P	R	Parameters	M_r	P	R	F1	M
CauseNet-F-Phrase	cTakes	RS ST21pv	M_{-2}	0.00	0.00†	RS	M_2	0.28†	0.95	RS	M_5	0.43	0.67	0.52	0.39†
	MetaMap	RS ST21pv	$M_{-\infty}$	0.67	0.05†	RS	M_2	0.35†	0.95	RS	M_2	0.52	0.82	0.63	0.54†
	QuickUMLS	RS ST21pv, 0.8	M_{-1}	0.00	0.00†	RS, 0.8	M_2	0.30†	0.87	RS, 0.8	M_2	0.57	0.41	0.48	0.38†
	ScispaCy	RS, *sm*, 0.9	$M_{-\infty}$	0.00	0.00†	UMLS, *lg*, 0.6	M_2	0.26†	0.79	RS, *sm*, 0.6	M_{10}	0.35	0.64	0.45	0.29†
	BERT	ENC, NP	M_1	0.91	0.77	ENC, NP	M_5	**0.76**	**0.97**	ENC, NP	M_2	0.81	0.90	0.85	**0.82**
	SciBERT	ENC, NP	M_5	0.94	0.77	ENC, NP	M_2	**0.76**	0.95	ENC, NP	M_2	0.82	0.85	0.84	0.80
	PMBERT	ENC, NP	M_2	0.94	**0.82**	ENC, NP	M_1	0.73	0.95	ENC, NP	M_2	0.80	**0.92**	**0.86**	**0.82**
	CW	ENC, n=2, M_2	M_{-1}	0.85	0.72	ENC, n=3, M_5	M_2	0.58†	0.92	ENC, n=1, M_5	M_2	**0.84**	0.79	0.82	0.77
	TDS	ENC, n=2, M_1	M_1	**1.00**	0.69	ENC, n=2, M_0	M_{-1}	0.67	0.92	ENC, n=3, M_0	M_1	0.76	0.82	0.79	0.74
	DW	ENC, n=1, M_1	M_1	**1.00**	0.74	ENC, n=3, M_1	M_∞	0.62	0.90	ENC, n=2, M_2	M_1	0.78	0.90	0.83	0.79
CauseNet-P-Phrase	cTakes	RS	M_1	0.79	0.34†	RS	M_2	0.63†	**0.96**	RS	M_{-2}	0.75	0.56	0.64	0.33†
	MetaMap	RS	M_{-5}	0.76	0.85	RS	M_{-2}	0.69†	0.94	RS	M_{-5}	0.76	0.85	0.80	0.52†
	QuickUMLS	RS, 1.0	M_{10}	0.83	0.46†	RS, 0.9	M_2	0.66†	0.88	RS, 0.8	M_1	0.75	0.79	0.77	0.46†
	ScispaCy	RS, *sm*, 0.7	$M_{-\infty}$	0.71	0.33†	UMLS, *lg*, 0.7	M_2	0.60†	0.88	RS, *sm*, 0.8	M_{10}	0.66	0.80	0.73	0.30†
	BERT	ENC, NP	M_5	0.91	0.88	ENC, NP	M_{10}	0.92	0.91	ENC, NP	M_5	0.91	0.88	0.90	0.77
	SciBERT	ENC, NP	M_{10}	**0.97**	**0.89**	ENC, NP	M_5	0.90	0.90	ENC, NP	M_5	0.90	0.90	0.90	0.78
	PMBERT	ENC, NP	M_5	0.96	0.88	ENC, NP	M_2	0.91	0.95	ENC, NP	M_5	**0.96**	0.88	**0.92**	**0.83**
	CW	ENC, n=1, M_{10}	M_5	0.95	0.72	ENC, n=1, M_{10}	M_2	0.80†	0.91	ENC, n=1, M_5	M_2	0.83	0.89	0.86	0.66
	TDS	ENC, n=3, M_1	M_1	0.93	**0.89**	ENC, n=2, M_1	M_1	**0.93**	0.89	ENC, n=2, M_1	M_1	0.89	**0.93**	0.91	0.79
	DW	ENC, n=2, M_1	M_1	0.92	0.88	ENC, n=3, M_{10}	M_1	**0.93**	0.91	ENC, n=3, M_5	M_1	0.90	0.91	0.91	0.79
CauseNet-F-Sentence	cTakes	RS	–	0.70	0.20†	RS	–	0.25†	**0.91**	RS	–	0.57	0.46	0.51	0.42†
	MetaMap	RS	–	**1.00**	0.03†	RS	–	0.29†	**0.91**	RS	–	0.42	0.49	0.45	0.33†
	QuickUMLS	RS ST21pv, 1.0	–	**1.00**	0.03†	RS, 1.0	–	0.32†	**0.91**	RS, 1.0	–	0.49	0.49	0.49	0.38†
	ScispaCy	RS, *lg*, 0.8	–	0.88	0.20†	RS, *sm*, 0.9	–	0.25†	0.89	RS, *lg*, 0.6	–	0.42	0.60	0.49	0.37†
	BERT	ENC, NP	–	0.86	0.51	ENC, NP	–	0.59	0.83	ENC, NP	–	0.76	0.74	**0.75**	**0.70**
	SciBERT	ENC, NP	–	0.89	0.49	ENC, NP	–	0.53	0.89	ENC, NP	–	0.64	0.66	0.65	0.57
	PMBERT	ENC, NP	–	0.83	**0.57**	ENC, NP	–	0.59	0.86	ENC, NP	–	**0.87**	0.57	0.69	0.66
	CW	ENC, n=3, M_5	–	0.75	0.43	ENC, n=1, M_{10}	–	0.56	0.89	ENC, n=1, M_5	–	0.67	0.63	0.65	0.58
	TDS	ENC, n=1, M_1	–	0.84	0.46	ENC, n=1, M_2	–	**0.62**	**0.91**	ENC, n=2, M_0	–	0.69	0.63	0.66	0.59
	DW	ENC, n=3, M_1	–	0.83	0.43	ENC, n=2, M_2	–	0.59	0.83	ENC, n=2, M_1	–	0.71	**0.77**	0.74	0.68

Table 4: Effectiveness on the test sets as precision (P), recall (R), F1, or Matthews correlation coefficient (MCC) of the best parameterization of each approach optimized for precision, recall, or MCC on the respective training set. The operating point for precision / recall optimization is set to 0.9 on the training data (gray scores indicate that 0.9 could not be reached during training). Statistically significant differences to the best approach for a dataset and optimization criterion (best scores highlighted in bold) are denoted by † ($p < 0.05$, Bonferroni-corrected for the nine comparisons in each group). For entity linkers, the usage of UMLS or combined RxNorm and SNOMED CT vocabulary (RS) restricted / or not to the ST21pv subsets, and, where applicable, similarity thresholds or spaCy model size (*sm* or *lg*) are reported. BERT models were fine-tuned on the PubMed (PM) or Encyclopedia (ENC) corpus using sentences (S) or noun phrases (NP). Termhood scores use either the PubMed (PM), PubMed Central (PMC), Textbook (TB), or Encyclopedia (ENC) corpus, a maximum n-gram size of n, and the generalized mean M_r. In the phrase scenarios, the generalized mean M_r for combining the cause and effect scores is also reported.

precomputing the n-gram frequencies and then parallelizing hash table lookups in the inference phase, the termhood scores could be even further sped up. As for memory efficiency, the n-gram frequencies and the BERT model checkpoints have a similar memory footprint of about 400MB.

As for the assessment of phrases vs. sentences, our results in Table 4 show that most approaches are substantially more effective on just the phrase pairs than on sentences (e.g., BERT-based: 0.12 to 0.23 better MCC on CauseNet-F-Phrase than on CauseNet-F-Sentence; termhood: 0.11 to 0.19 better MCC). Only the ScispaCy entity linking approach is really more effective on sentences than phrases (MCC improves by 0.08).

Entity Linking-based Approaches The entity linking-based approaches mostly use the combined RxNorm and SNOMED CT vocabulary (some-

Approach	Phrase		Sentence	
	ms	Speedup	ms	Speedup
cTakes	119.68	0.5	212.12	0.2
MetaMap	49.64	1.2	120.28	0.4
QuickUMLS	7.23	8.3	8.98	5.3
ScispaCy	16.38	3.7	15.96	3.0
PubMedBERT	60.19	1.0	47.77	1.0
Termhood n=1	**0.56**	**107.5**	**1.02**	**46.8**
Termhood n=2	0.93	64.7	1.97	24.2
Termhood n=3	1.27	47.4	2.82	16.9

Table 5: Run time efficiency of the different approaches' most effective parameterization on the CauseNet-F-Phrase and -Sentence test sets. Time per instance averaged over 10 runs, speedup computed against PubMedBERT as the most effective approach from Table 4.

times only the ST21pv subset) and only once the full UMLS. Still, even with such a "restricted" vocabulary, the entity linking-based approaches hardly achieve really high precision values—or only at the expense of very low recall. One reason is that even specifically tailored health-related entity vocabularies still contain many terms that are only loosely health-related and then yield false positive results on the cause–effect statements. Interestingly, only the MetaMap and the ScispaCy parameterizations "attempt" to compensate for this by using generalized averages M_r with $r < 0$ for the cause–effect combination when optimizing for precision. Still, the many 0-values for precision optimization on CauseNet-F-Phrase indicate that no health-related statements are found. This is again caused by a "precision problem". In the grid search on CauseNet-F-Phrase, all entity linking-based approaches achieve precision values of at least 0.9 but at a very tiny recall (6–8 true positives). On the test sets, these low-recall parameterizations do not detect any of the health-related statements.

Another drawback of some entity linkers in the phrase scenarios is their reliance on syntactic parsing to detect candidate mentions—the parses might not be too meaningful for (short) phrases. ScispaCy in particular relies on parsing and thus is less effective than the other linkers on phrases—even though ScispaCy is one of the best medical entity linkers in full text scenarios (Vashishth et al., 2021).

Overall, the entity linking-based approaches achieve rather low effectiveness compared to the other approaches and are also slower than our new generalized termhood method. In our scenario of assessing the health relatedness of phrases and sentences, they are not really a good option.

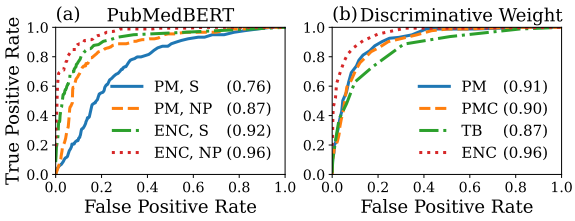

Figure 2: ROC curves and AUC values (in parentheses) (a) for fine-tuning PubMedBERT on the CauseNet-F-Sentence training data using sentences (S) or noun phrases (NP) from the PubMed (PM) or Encyclopedia (ENC) corpus, and (b) for discriminative weight DW ($n = 1, r = 1$) on the CauseNet-F-Phrase training data with the health corpora PubMed (PM), PubMed Central (PMC), Textbooks (TB), or Encyclopedia (ENC).

BERT-based Approaches Interestingly, domain specific pre-training only has a minor positive effect for the BERT-based models. While PubMedBERT often is the most effective, SciBERT and even the domain-agnostic BERT usually are almost as effective. Interestingly, all prefer the recall-oriented higher values of r in the generalized average M_r for the cause–effect score combination.

Another observation is that the best BERT-based approaches all are fine-tuned on noun phrases from the Encyclopedia corpus—even in the sentence scenario. As an example, Figure 2 (a) shows the ROC curves and AUC values for PubMedBERT in different fine-tuning setups on the CauseNet-F-Sentence training data. Fine-tuning on the Encyclopedia corpus clearly achieves better AUC values (at least 0.06 over PubMed) as does fine-tuning on phrases instead of sentences (at least 0.04).

Termhood-based Approaches Among the termhood-based approaches, TDS and DW are more effective than CW in almost all scenarios.

Further analysis of the termhood methods' parameters shows that the health corpus is very important: the best parameterizations all use the Encyclopedia corpus. Figure 2 (b) details this observation for DW. The AUC-ROC value with the Encyclopedia corpus is by far the best; the health corpus' fit to the target language is crucial. The n-gram length and the generalized mean setup are less important. Table 6 shows several ablation results. In most cases, the ablated setups achieve equal or lower effectiveness, but in some cases higher effectiveness. By fixing $n = 1$, the MCC decreases by 0.00 to 0.03 on most datasets (except TDS / CW on sentences where it increases). Fixing the phrase-internal averaging to the arithmetic mean (M_1) only decreases the MCC by a maximum of 0.05.

	Appr.	P Opt.		R Opt.		M Opt.	
		$n=1$	M_1	$n=1$	M_1	$n=1$	M_1
F-Phrase	CW	-0.10	-0.08	-0.03	-0.03	0.00	-0.05
	TDS	0.00	0.00	-0.05	0.03	-0.02	0.02
	DW	0.00	0.00	-0.04	0.00	-0.01	-0.01
P-Phrase	CW	0.00	0.00	0.00	0.00	0.00	-0.02
	TDS	-0.05	0.00	0.01	0.00	-0.03	0.00
	DW	-0.07	0.00	-0.03	0.00	-0.01	0.03
F-Sent.	CW	0.03	-0.03	0.00	0.00	0.00	0.05
	TDS	0.00	0.00	0.00	-0.11	0.07	0.10
	DW	-0.03	0.00	0.00	-0.03	0.02	0.00

Table 6: Ablation study indicating the difference in effectiveness on the test sets from the best parameterizations optimized for precision (P), recall (R), or the Matthews correlation coefficient (M) to the best parameterizations when ablating the n-gram length (fixed $n = 1$) or the generalized mean (fixed M_1 for combining the n-gram scores). Operating point for precision / recall is set to 0.9. The difference is given with respect to the "interesting" measure (i.e., drop in recall for the P columns, drop in precision for the R columns, and drop in MCC for the M columns).

Overall, our generalized termhood-based methods are much faster but not significantly less effective than the BERT-based approaches—with TDS and DW usually being better than CW. When applying the generalized termhood methods, it is crucial to choose a good health corpus while optimizing the other parameters (n-gram length, averaging) only leads to smaller improvements.

5 Webis Medical CauseNet 2022

By applying our generalized termhood method for health relatedness assessment to the complete CauseNet, we create the new Webis Medical CauseNet 2022 resource consisting of health-related cause–effect statements found on the web. It is important to note, that the statements in CauseNet—and thus also in our Webis Medical CauseNet 2022—are only *claimed* cause–effect statements. For many of the contained statements, scientific evidence can surely be found (e.g., 'stress → insomnia') while for many other this might not be possible (e.g., 'incorrect placement of jupiter → diabetes'). Still, it could be interesting to analyze websites that contain many claimed health-related cause–effect statements with respect to whether medical evidence exists or not. Such health-sociological analyses are now enabled by our Webis Medical CauseNet 2022 resource at web scale since the URLs of the pages from which a statement was extracted are part of CauseNet.

Subset	Statements	Sentences	P	R
Prec (P)	103,273	1,259,339	0.93	0.89
Prec (MCC)	112,707	1,340,873	0.89	0.93
Full (P)	2,201,071	5,680,635	1.00	0.74
Full (MCC)	3,206,964	7,842,464	0.78	0.90

Table 7: Characteristics of the four Webis Medical CauseNet 2022 versions. Number of statements / supporting sentences, and estimated precision and recall of the precision- or MCC-optimized termhood extraction.

The Webis Medical CauseNet 2022 comes in four different versions[14] based on the best termhood parameterizations from the evaluation. Table 7 contains some general characteristics. The two smaller versions are extracted from CauseNet-Prec (statements with high support) by using TDS optimized for precision or MCC, while the two larger versions are extracted from CauseNet-Full by using DW optimized for precision or MCC.

6 Conclusions

We have proposed generalized termhood-based methods that effectively and efficiently assess the health relatedness of phrases. On cause–effect statements from the web, our new approaches are almost as effective as the best BERT-based approaches while being much faster. Approaches using state-of-the-art medical entity linkers are slower and less effective. When configuring our new termhood-based methods, it is crucial to select a background health corpus that matches the target language distribution while the other parameters (n-gram length, averaging) are less important.

Using our methods, we have extracted the Webis Medical CauseNet 2022 resource of health-related cause–effect statements from the web-scale CauseNet. Based on Webis Medical CauseNet 2022, health-sociological analyses of online cause–effect relations are now possible at an unprecedented scale compared to previous small-scale analyses of health-related online information.

Finally, our termhood-based assessment could also be useful in retrieval scenarios. For instance, given the termhood scores' efficiency, they could directly be used at search engine side to quickly assess the health relatedness of some query not seen before (for other queries, the clicked documents usually suffice to assess the health relatedness) and to possibly adopt the retrieval accordingly (e.g., preferring medical resources).

[14] Available under a permissive license: https://github.com/webis-de/COLING-22

References

Khurshid Ahmad, Lee Gillam, and Lena Tostevin. 1999. University of Surrey participation in TREC8: Weirdness indexing for logical document extrapolation and retrieval (WILDER). In *Proceedings of The Eighth Text REtrieval Conference (TREC 1999), Gaithersburg, Maryland, USA, November 17–19, 1999*.

Hege K. Andreassen, Maria M. Bujnowska-Fedak, Catherine E. Chronaki, Roxana C. Dumitru, Iveta Pudule, Silvina Santana, Henning Voss, and Rolf Wynn. 2007. European citizens' use of e-health services: A study of seven countries. *BMC Public Health*, 7(1):53.

Alan R. Aronson. 2001. Effective mapping of biomedical text to the UMLS Metathesaurus: The MetaMap program. In *Proceedings of the American Medical Informatics Association Annual Symposium (AMIA 2001), Washington, DC, USA, November 3–7, 2001*.

Rakesh Bal, Sayan Sinha, Swastika Dutta, Rishabh Joshi, Sayan Ghosh, and Ritam Dutt. 2020. Analysing the extent of misinformation in cancer related tweets. In *Proceedings of the Fourteenth International AAAI Conference on Web and Social Media (ICWSM 2020), Held Virtually, Original Venue: Atlanta, Georgia, USA, June 8–11, 2020*, pages 924–928.

Roberto Basili, Alessandro Moschitti, Maria Teresa Pazienza, and Fabio Massimo Zanzotto. 2001. A contrastive approach to term extraction. In *Proceedings of Terminologie et Intelligence Artificielle (TIA 2001), Nancy, France, May 3–4, 2001*, pages 119–128.

Iz Beltagy, Kyle Lo, and Arman Cohan. 2019. SciBERT: A pretrained language model for scientific text. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP 2019), Hong Kong, China, November 3–7, 2019*, pages 3613–3618.

Alexander Bondarenko, Ekaterina Shirshakova, Marina Driker, Matthias Hagen, and Pavel Braslavski. 2021. Misbeliefs and biases in health-related searches. In *Proceedings of the 30th ACM International Conference on Information and Knowledge Management (CIKM 2021), Virtual Event, Queensland, Australia, November 1–5, 2021*, pages 2894–2899.

Alexander Bondarenko, Magdalena Wolska, Stefan Heindorf, Lukas Blübaum, Axel-Cyrille Ngonga Ngomo, Benno Stein, Pavel Braslavski, Matthias Hagen, and Martin Potthast. 2022. A benchmark for causal question answering. In *Proceedings of the 29th International Conference on Computational Linguistics (COLING 2022), Gyeongju, Republic of Korea, October 12–17, 2022, (to appear)*.

Francesca Bonin, Felice Dell'Orletta, Simonetta Montemagni, and Giulia Venturi. 2010. A contrastive approach to multi-word extraction from domain-specific corpora. In *Proceedings of the International Conference on Language Resources and Evaluation (LREC 2010), Valletta, Malta, May 17–23, 2010*.

Brian Brennan, Gen Kanayama, and Harrison Pope. 2013. Performance-enhancing drugs on the web: A growing public-health issue. *The American Journal on Addictions*, 22(2):158–161.

David A. Broniatowski, Amelia M. Jamison, SiHua Qi, Lulwah AlKulaib, Tao Chen, Adrian Benton, Sandra Quinn, and Mark Dredze. 2018. Weaponized health communication: Twitter bots and russian trolls amplify the vaccine debate. *American Journal of Public Health*, 108(10):1378–1384.

Davide Chicco and Giuseppe Jurman. 2020. The advantages of the Matthews correlation coefficient (MCC) over F1 score and accuracy in binary classification evaluation. *BMC Genomics*, 21:6.

Benjamin E. J. Cooper, William E. Lee, Ben M. Goldacre, and Thomas A. B. Sanders. 2012. The quality of the evidence for dietary advice given in UK national newspapers. *Public Understanding of Science*, 21(6):664–673.

Lubna Daraz, Allison S. Morrow, Oscar J. Ponce, Wigdan Farah, Abdulrahman Katabi, Abdul Majzoub, Mohamed O. Seisa, Raed Benkhadra, Mouaz Alsawas, Prokop Larry, and M. Hassan Murad. 2018. Readability of online health information: A meta-narrative systematic review. *American Journal of Medical Quality*, 33(5):487–492.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL 2019), Minneapolis, MN, USA, June 2-7, 2019, Volume 1 (Long and Short Papers)*, pages 4171–4186.

Joseph A. Diaz, Rebecca A. Griffith, James J. Ng, Steven E. Reinert, Peter D. Friedmann, and Anne W. Moulton. 2002. Patients' use of the internet for medical information. *Journal of General Internal Medicine*, 17(3):180–185.

Gunther Eysenbach, John Powell, Oliver Kuss, and Eun-Ryoung Sa. 2002. Empirical studies assessing the quality of health information for consumers on the world wide web: A systematic review. *JAMA*, 287(20):2691–2700.

Katerina T. Frantzi, Sophia Ananiadou, and Hideki Mima. 2000. Automatic recognition of multi-word terms: The C-value/NC-value method. *International Journal on Digital Libraries*, 3(2):115–130.

George Gkotsis, Anika Oellrich, Sumithra Velupillai, Maria Liakata, Tim J. P. Hubbard, Richard J. B. Dobson, and Rina Dutta. 2017. Characterisation of mental health conditions in social media using informed deep learning. *Scientific Reports*, 7(1):45141.

Yu Gu, Robert Tinn, Hao Cheng, Michael Lucas, Naoto Usuyama, Xiaodong Liu, Tristan Naumann, Jianfeng Gao, and Hoifung Poon. 2022. Domain-specific language model pretraining for biomedical natural language processing. *ACM Transactions on Computing for Healthcare*, 3(1):2:1–2:23.

Stefan Heindorf, Yan Scholten, Henning Wachsmuth, Axel-Cyrille Ngonga Ngomo, and Martin Potthast. 2020. CauseNet: Towards a causality graph extracted from the web. In *Proceedings of the 29th ACM International Conference on Information and Knowledge Management (CIKM 2020), Virtual Event, Ireland, October 19–23, 2020*, pages 3023–3030.

Betsy L. Humphreys and Donald A. B. Lindberg. 1993. The UMLS project: Making the conceptual connection between users and the information they need. *Bulletin of the Medical Library Association*, 81(2):170–177.

Jerry L. Jensen. 1998. Some statistical properties of power averages for lognormal samples. *Water Resources Research*, 34(9):2415–2418.

You-Ling Jiang. 2000. Quality evaluation of orthodontic information on the world wide web. *American Journal of Orthodontics and Dentofacial Orthopedics*, 118(1):4–9.

Kyo Kageura and Bin Umino. 1996. Methods of automatic term recognition: A review. *Terminology. International Journal of Theoretical and Applied Issues in Specialized Communication*, 3(2):259–289.

Su Nam Kim, Timothy Baldwin, and Min-Yen Kan. 2009. An unsupervised approach to domain-specific term extraction. In *Proceedings of the Australasian Language Technology Association Workshop (ALTA 2009), Sydney, Australia, December 3–4, 2009*, pages 94–98.

Sunil Mohan and Donghui Li. 2019. MedMentions: A large biomedical corpus annotated with UMLS concepts. In *Proceedings of the 1st Conference on Automated Knowledge Base Construction (AKBC 2019), Amherst, MA, USA, May 20–22, 2019*.

Mark Neumann, Daniel King, Iz Beltagy, and Waleed Ammar. 2019. ScispaCy: Fast and robust models for biomedical natural language processing. In *Proceedings of the 18th BioNLP Workshop and Shared Task (BioNLP@ACL 2019), Florence, Italy, August 1, 2019*, pages 319–327.

Youngja Park, Siddharth Patwardhan, Karthik Visweswariah, and Stephen C. Gates. 2008. An empirical analysis of word error rate and keyword error rate. In *Proceedings of the 9th Annual Conference of the International Speech Communication Association (INTERSPEECH 2008), Brisbane, Australia, September 22–26, 2008*, pages 2070–2073.

Guergana K. Savova, James J. Masanz, Philip V. Ogren, Jiaping Zheng, Sunghwan Sohn, Karin C. Kipper-Schuler, and Christopher G. Chute. 2010. Mayo clinical text analysis and knowledge extraction system (cTAKES): Architecture, component evaluation and applications. *Journal of the American Medical Informatics Association*, 17(5):507–513.

Luca Soldaini and Nazli Goharian. 2016. QuickUMLS: A fast, unsupervised approach for medical concept extraction. In *Proceedings of the 2nd SIGIR Workshop on Medical Information Retrieval (MedIR 2016), Pisa, Italy, July 21, 2016*.

Victor Suarez-Lledo and Javier Alvarez-Galvez. 2021. Prevalence of health misinformation on social media: Systematic review. *Journal of Medical Internet Research*, 23(1):e17187.

Petroc Sumner, Solveiga Vivian-Griffiths, Jacky Boivin, Andy Williams, Christos A. Venetis, Aimée Davies, Jack Ogden, Leanne Whelan, Bethan Hughes, Bethan Dalton, Fred Boy, and Christopher D. Chambers. 2014. The association between exaggeration in health related science news and academic press releases: Retrospective observational study. *BMJ*, 349:g7015.

Yalin Sun, Yan Zhang, Jacek Gwizdka, and Ciaran B. Trace. 2019. Consumer evaluation of the quality of online health information: Systematic literature review of relevant criteria and indicators. *Journal of Medical Internet Research*, 21(5):e12522.

Shikhar Vashishth, Denis Newman-Griffis, Rishabh Joshi, Ritam Dutt, and Carolyn P. Rosé. 2021. Improving broad-coverage medical entity linking with semantic type prediction and large-scale datasets. *Journal of Biomedical Informatics*, 121:103880.

Carolyn R. Watters, Wanhong Zheng, and Evangelos E. Milios. 2002. Filtering for medical news items. In *Proceedings of the 65th ASIS&T Annual Meeting (ASIST 2002), Philadelphia, PA, USA, November 18–21, 2002*, volume 39, pages 284–291.

Wilson Wong, Wei Liu, and Mohammed Bennamoun. 2007. Determining termhood for learning domain ontologies using domain prevalence and tendency. In *Proceedings of the Sixth Australasian Data Mining Conference (AusDM 2007), Gold Coast, Queensland, Australia, December 3–4, 2007*, pages 47–54.

Jie Yang, Soyeon Caren Han, and Josiah Poon. 2022. A survey on extraction of causal relations from natural language text. *Knowledge and Information Systems*, 64(5):1161–1186.

Amélie Yavchitz, Isabelle Boutron, Aida Bafeta, Ibrahim Marroun, Pierre Charles, Jean Mantz, and Philippe Ravaud. 2012. Misrepresentation of randomized controlled trials in press releases and news coverage: A cohort study. *PLOS Medicine*, 9(9):e1001308.

Yan Zhang, Yalin Sun, and Bo Xie. 2015. Quality of health information for consumers on the web: A systematic review of indicators, criteria, tools, and evaluation results. *Journal of the Association for Information Science and Technology*, 66(10):2071–2084.

Wanhong Zheng, Evangelos E. Milios, and Carolyn R. Watters. 2002. Filtering for medical news items using a machine learning approach. In *Proceedings of the American Medical Informatics Association Annual Symposium (AMIA 2002), San Antonio, TX, USA, November 9–13, 2002*.

Find the Funding: Entity Linking with Incomplete Funding Knowledge Bases

Gizem Aydin
Radboud University
gizemaydin96@gmail.com

Seyed Amin Tabatabaei
Elsevier
s.tabatabaei@elsevier.com

Giorgios Tsatsaronis
Elsevier
g.tsatsaronis@elsevier.com

Faegheh Hasibi
Radboud University
f.hasibi@cs.ru.nl

Abstract

Automatic extraction of funding information from academic articles adds significant value to industry and research communities, including tracking research outcomes by funding organizations, profiling researchers and universities based on the received funding, and supporting open access policies. Two major challenges of identifying and linking funding entities are: (i) sparse graph structure of the Knowledge Base (KB), which makes the commonly used graph-based entity linking approaches suboptimal for the funding domain, (ii) missing entities in KB, which (unlike recent zero-shot approaches) requires marking entity mentions without KB entries as NIL. We propose an entity linking model that can perform NIL prediction and overcome data scarcity issues in a time and data-efficient manner. Our model builds on a transformer-based mention detection and a bi-encoder model to perform entity linking. We show that our model outperforms strong existing baselines.

1 Introduction

Entity Linking (EL) aims to annotate text with corresponding entity identifiers from a Knowledge Base (KB) and is a building block for different tasks, such as document ranking (Xiong et al., 2017), entity retrieval (Hasibi et al., 2016), and question understanding in conversations (Shang et al., 2021). Recent years have witnessed the flourishing of entity linking approaches for zero-shot (Wu et al., 2020; Li et al., 2020) and open-domain setups (van Hulst et al., 2020; Cao et al., 2021). While zero-shot entity linking can generalize to new specialized domains and entity dictionaries, existing approaches cannot perform NIL prediction; i.e., identifying entity mentions without a target entity in a knowledge base and assigning them to NIL. Open-domain entity linkers, on the other hand, build on the availability of rich entity relations and descriptions in KBs. This makes existing EL approaches suboptimal for real-world applications of entity linking in domains with incomplete knowledge bases, where both in-KB and out-of-KB entities should be identified.

In this paper, we aim to address entity linking in the funding domain (Dai et al., 2021; Alexander and de Vries, 2021), which is essential for funding organizations to track the outcome of the research they funded (Kayal et al., 2019) and also helps to comply with open access rules (Dai et al., 2021). Knowledge bases of funding organizations, either proprietary or open access (e.g., the funding KB Crossref[1]), contain brief information about entities (e.g., official name and acronym). They also have extremely sparse graph structure with large amount of missing entities that need to be found from research articles. This implies that EL in the funding domain requires detecting mentions with out-of-KB entities while handling sparse entity relations and descriptions. The approach, should also be able to operate with limited training data, as large public datasets are rarely available for domain-specific applications.

We propose a two-step EL approach, where we first identify entity mentions using task adaptive pre-training (Gururangan et al., 2020) of BERT (Devlin et al., 2019) and then perform Entity Disambiguation (ED) by utilizing a bi-encoder model to learn dense entity and mention representations. Our bi-encoder model and training approach using negative sampling are specifically designed to operate with out-of-KB entities. The bi-encoder is followed by a modest feature-based model to map the entities to an entity in KB or NIL. We create two new datasets for EL in the funding domain and compare our mention detection and ED approaches with strong neural and feature-based models. We show that our model improves over existing baselines for both entity disambiguation and end-to-end entity linking.

[1] https://www.crossref.org

Proceedings of the 29th International Conference on Computational Linguistics, pages 1937–1942
October 12–17, 2022.

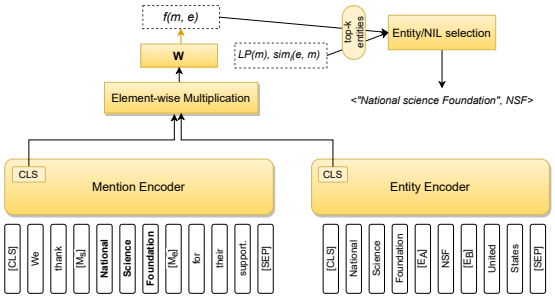

Figure 1: Overview of our entity disambiguation approach for incomplete proprietary knowledge bases.

In summary, our contributions include: (i) proposing a data-efficient model for entity linking (with NIL prediction) in funding domain that is efficient and can be used with modest computational power, (ii) improving upon existing EL approaches for funding organization, and (iii) releasing new training and evaluation datasets for entity linking in funding domain. To our knowledge, this is the first and largest publicly available dataset for entity linking in funding domain. The code and datasets created in this paper are made publicly available.[2]

2 Method

In this section, we provide a formal definition of the task, followed by the description of our Mention Detection (MD) approach and Funding entity Disambiguation model, referred to as FunD.

2.1 Task Definition

We denote $\mathcal{E}$ as the set of entities in a knowledge base, where each entity $e \in \mathcal{E}$ is accompanied by a textual description. Let $m = (s, t) \in M$ denote an entity mention with start and end positions s and t. Given a document $d = \{w_1, w_2, ..., w_n\}$, our aim is to generate the list:

$$\mathcal{L} = \{(\langle s, t\rangle, a) | 1 \leq s \leq t \leq n, a \in \mathcal{E} \cup NIL\},$$

which represents all possible mentions linked to an entity in the KB (*in-KB setup*) or NIL (*out-of-KB setup)*. This task is similar to zero-shot (Wu et al., 2020) and open domain entity linking (van Hulst et al., 2020), but different from them, entities do not need to have an entry in the KB.

2.2 Mention Detection

For mention detection, we adapt BERT (Devlin et al., 2019) to the funding domain. Domain Adaptive Pre-Training (DAPT), while being effective, requires a large amount of domain-specific text, requires large amount of training data which is not feasible for the funding domain (Gerritse et al., 2022; Nogueira et al., 2019). We therefore utilize the Task-Adaptive Pre-Training (TAPT) (Gururangan et al., 2020), which requires a far smaller but more task-relevant training corpus and is proven to be more effective than DAPT. We train BERT with the Masked Language Model objective on acknowledgments of research papers. We refer to this model as $\text{BERT}_{\text{TAPT}}$. We then fine-tune $\text{BERT}_{\text{TAPT}}$for the mention detection task using IOB tags.

2.3 Entity Disambiguation

Candidate Entity Selection To obtain the likelihood of an entity being a target link of a mention, we employ a bi-encoder model (Wu et al., 2020) for encoding a mention (with its context) and an entity. Our bi-encoder model utilizes two BERT encoders for generating entity and mention representations. The entity encoder takes the structured entity description as the input:

$$x_e = \text{BERT}_{\texttt{[CLS]}}(\texttt{[CLS]}\ \text{val}_{A_1}\ \texttt{[E}_A\texttt{]}\ \ldots\ \texttt{[E}_A\texttt{]}\ \text{val}_{A_n}\ \texttt{[E}_B\texttt{]}\ \text{val}_B\ \texttt{[SEP]}),$$

where $\texttt{[E}_A\texttt{]}$ and $\texttt{[E}_B\texttt{]}$ are two word-piece tokens, selected among the unused tokens of BERT, and val_{A_i} and val_B denote values for entity attributes *A* and *B*. Here *A* corresponds to names of entities, which is a multi-valued attribute, and *B* is the country of the funding organization. For the mention encoder, we follow Wu et al. (2020) and obtain mention representations by:

$$x_m = \text{BERT}_{\texttt{[CLS]}}(\texttt{[CLS]}\ \text{ctxt}_{left}\ \texttt{[M}_s\texttt{]}\ \text{mention}\ \texttt{[M}_e\texttt{]}\ \text{ctxt}_{right}\ \texttt{[SEP]}),$$

where ctxt_{left} and ctxt_{right} represent context words before and after the mention. In this work, both BERT models are initialized with $\text{BERT}_{\text{TAPT}}$. The mention-entity score is then obtained by:

$$f(m, e) = \mathbf{W}^T(x_m \odot x_e), \qquad (1)$$

where $\odot$ refers to element-wise multiplication of mention representation x_m and entity representation x_e, and $\mathbf{W} \in \mathcal{R}^{BERT \times 2}$ represents learnable weights. The binary cross-entropy loss L is used to train the model:

$$L = -\frac{1}{N}\sum_{i=1}^{N} l_i \log(f(m_i, e_i)) - (1 - l_i) \log(1 - f(m_i, e_i)),$$

[2] https://github.com/informagi/Fund-EL

where N is the number of training examples and l_i is a binary value that is set to 1 if e_i is the correct entity for mention m_i.

Negative Sampling Following Gillick et al. (2019), we perform training in rounds, where the model obtained in each round is used to produce hard negatives for the next round. Contrary to (Gillick et al., 2019), we do not use in-batch random negative sampling, as it provides less diverse random negatives for sparse domain-specific applications compared to open-domain EL.

The following strategy is employed for random and hard negative sampling. In the first round, negative entities of each mention are sampled randomly from the entire KB. For the next rounds, both random and hard negatives are used. Hard negatives are entities ranked above the correct entity by the model learned in the previous step. For mentions with out-of-KB entities, the top-K entities are selected as hard negatives (K is set to 10 following (Gillick et al., 2019)). The number of random negatives for each mention is computed based on the number of hard negatives:

$$Neg_r(m) = \lfloor \frac{\sum_{i=1}^{|M|} Neg_h(m)}{|M|} \rfloor, \qquad (2)$$

where $Neg_r()$ and $Neg_h()$ give number of random and hard negatives, respectively. Using this strategy, we strive a balance between random and hard negatives, while giving hard mentions (i.e., mentions that their correct entities are in low ranks) a larger number of hard negatives.

Entity or NIL Selection Once we have obtained candidate entities from our bi-encoder model, we turn to mapping each mention to an entity in the knowledge base or NIL. We employ a feature-based model using Gradient Boosting Machine (GBM) (Friedman, 2001). Our model utilizes five light-weight features: (i) score obtained by the bi-encoder model, (ii) maximum Levenshtein similarity between the mention and the labels of the candidate entity, (iii) link probability of the mention (Balog, 2018) obtained by dividing numbers of times a mention appears as a link by total number of occurrences of a term: $P(link|m) = n_{link}(m)/n(m)$, and (iv) commonness score (Balog, 2018) obtained by $P(e|m) = n(m,e)/\sum_{e' \in \mathcal{E}} n(m,e')$, with $n(m,e)$ denoting number of times that mention m is linked to entity e. A mention is linked to the entity with the highest GBM score if higher than threshold τ.

System	Set	P	R	F1
Stanford NER	Test	73.70	75.10	74.39
Flair^{NER}	Test	**85.83**	78.02	81.74
BERT^{NER}	Test	79.18	86.03	82.46
BERT$^{MD}_{TAPT}$	Test	80.28	**86.54**	**83.29**
Stanford NER	Eval	76.17	72.87	74.48
BERT$^{MD}_{TAPT}$	Eval	**79.08**	**85.31**	**82.08**

Table 1: Mention detection results on ELFund dataset.

3 Experiments

Data We use the Crossref funding registry as our KB, containing information about 25,859 funding organizations. We create two new datasets for the funding domain, ELFund and EDFund (Afzal et al., 2022), which are used for experiments. The datasets are split into training, validation, test, and eval sets, with no overlaps in the training, test, and eval across the two datasets. The validation set is used for searching hyper-parameters.The eval set contains completely unseen production data, used for the final evaluation of the models; see Appendix A and B for more details about the datasets and experimental setup.

Evaluation Metrics To evaluate the mention detection step, we use strong matching precision, recall, and F1 score (Tjong Kim Sang and De Meulder, 2003; Usbeck et al., 2019). The ED and EL tasks are evaluated in three settings: (i) *In-KB*, for mentions linked to entities in KB, (ii) *Emerging Entities (EE)* for Out-of-KB entities; i.e., mentions linked to no entities, and (iii) *All*, for in-KB and emerging entities. We evaluate the ED task using micro and macro averaged accuracy for the All setting (Hoffart et al., 2014). For In-KB and EE settings, we report on micro- and macro- averaged precision, recall, and F1 (Usbeck et al., 2019).

4 Results

Mention Detection Table 1 shows the results for the mention detection step. We compare BERT$^{MD}_{TAPT}$with Stanford NER (Finkel et al., 2005), Flair^{NER} (Akbik et al., 2018), and BERT^{NER} (Devlin et al., 2019). The results show that Flair^{NER}achieves the highest precision, but the lowest recall compared to the BERT-based models. We also observe that BERT$^{MD}_{TAPT}$outperforms all baselines with respect to the F1 score, showing the importance of task adaptive pre-training when limited data is available.

Method	Set	All		EE			In-KB		
		Acc$_{mic}$	**Acc**$_{mac}$	**P**$_{mic}$	**R**$_{mic}$	**F1**$_{mic}$	**P**$_{mic}$	**R**$_{mic}$	**F1**$_{mic}$
Commonness	Test	83.8	85.81	53.55	**88.2**	66.64	**94.22**	82.99	88.25
GBM$_{F26}$	Test	91.02	**92.84**	**79.11**	78.67	78.89	93.2	**93.29**	93.25
FunD	Test	**91.15**	92.76	77.44	81.14	**79.25**	93.82	92.99	**93.40**
GBM$_{F26}$	Eval	90.26	90.84	**80.03**	81.49	80.75	92.69	**92.3**	92.5
FunD	Eval	**90.66**	**91.11**	79.26	**85.45**	**82.24**	**93.56**	91.86	**92.7**

Table 2: Entity disambiguation results on the EDFund dataset. Best results for each set are marked in bold face.

MD	ED	Setting	F1$_{mic}$	F1$_{mac}$
Stanford NER	GBM$_{F26}$	All	68.43	69.34
BERT$^{\text{MD}}_{\text{TAPT}}$	FunD	All	**75.81**	**76.59**
Stanford NER	GBM$_{F26}$	EE	43.34	71.01
BERT$^{\text{MD}}_{\text{TAPT}}$	FunD	EE	**52.82**	**73.68**
Stanford NER	GBM$_{F26}$	In-KB	77.33	74.73
BERT$^{\text{MD}}_{\text{TAPT}}$	FunD	In-KB	**85.14**	**81.86**

Table 3: Entity linking results on the ELFund dataset.

System	With GPU	Without GPU
FunD	9.26 ± 0.47	23.07 ± 0.78
GBM$_{F26}$	99.2 ± 0.45	99.2 ± 0.45

Table 4: Efficiency of ED models (in seconds).

Entity Disambiguation Table 2 presents entity disambiguation results. We compare our ED method, FunD with two baselines: (i) Commonness (Hasibi et al., 2015), where each mention is linked to the entity with the highest commonness score if it is greater than zero, (ii) GBM$_{F26}$, which is a strong feature-based GBM model with 26 features, ranging from string similarities (e.g. BM25) to statistical features (e.g., commonness and link probability). We note that state-of-the-art EL methods, such as REL (van Hulst et al., 2020), GENRE (Cao et al., 2021), and BLINK (Wu et al., 2020) cannot be used as baselines, as they rely on data resources that are not available in our KB and also do not address NIL prediction. We, however, implemented BLINK's bi-encoder with a score threshold, and obtained micro average accuracy of 60.16, which is a far worse performance compared to other baselines. Table 2 results show that FunD strives a balance between precision and recall and can achieve the best results with respect to F1 in both In-KB and EE setups. This observation is also mirrored with respect to accuracy on the Eval set.

Entity Linking Putting the pieces together, we show the results of end-to-end entity linking on test set of ELFund dataset in Table 3. We compare our model with the best ED baseline (GBM$_{F26}$) combined with Stanford NER (a fast and strong existing MD model). The results indicate that our MD and ED models improve the existing feature based model by a large margin, reinforcing our previous finding that our models can be successfully applied to the funding domain.

Efficiency Finally, we measure the run time of ED model by running it on a random sample of 100 sentences with 306 mentions. We pass 12 candidate entities to both GBM$_{F26}$ and FunD and measure the run time in seconds. The experiment is repeated 10 times on a machine with an Intel Xeon E-2276M (2.80GHz, 32GB RAM) CPU and an NVIDIA Quadro T1000 GPU with 4GB memory. Table 4 shows that FunD is four times faster than the feature-based GBM$_{F26}$ model without GPU. The difference is even larger using GPU, as FunD's efficiency is increased, while GBM$_{F26}$ performance does not change with GPU. The inefficiency of the GBM$_{F26}$ model is mostly attributed to the calculation of the hand-crafted features.

5 Conclusions

In this paper, we have introduced an entity linking method for funding domain, where the knowledge base has sparse graph structure and limited information is available about entities. The model builds on BERT to perform mention detection, and a bi-encoder model to conduct the entity disambiguation. We compared our method to strong feature-based and zero-shot models and showed that our model can perform NIL assignments and overcome data scarcity issues more efficient and effective than comparable baselines. As future work, we would like to explore the benefit of employing contrastive learning for the highly ambiguous entity mentions, which could provide further robustness to extracting and linking such entities in scientific texts that span across all sciences.

Acknowledgements

We thank Ramadurai Petchiappan, Georgios Cheirmpos, and Efthymios Tsakonas, for their assistance in preparing the final released datasets.

References

Zubair Afzal, Johan Boots, Heber Mc Mahon, Nishant Mintri, Seyedamin Tabatabaei, and George Tsatsaronis. 2022. Edfund- Elsevier funding entity linking dataset.

Alan Akbik, Duncan Blythe, and Roland Vollgraf. 2018. Contextual string embeddings for sequence labeling. In *Proceedings of 27th International Conference on Computational Linguistics (COLING)*, pages 1638–1649.

Daria Alexander and Arjen P. de Vries. 2021. "this research is funded by..": Named entity recognition of financial information in research papers. In *Proceedings of the 11th International Workshop on Bibliometric-enhanced Information Retrieval co-located with 43rd European Conference on Information Retrieval (ECIR)*, pages 102 - 110.

Krisztian Balog. 2018. *Entity-oriented search*. Springer Nature.

Nicola De Cao, Gautier Izacard, Sebastian Riedel, and Fabio Petroni. 2021. Autoregressive entity retrieval. In *Proceedings of the 9th International Conference on Learning Representations (ICLR)*, pages 3–7.

Suyang Dai, Yuxia Ding, Zihan Zhang, Wenxuan Zuo, Xiaodi Huang, and Shanfeng Zhu. 2021. Grantextractor: Accurate grant support information extraction from biomedical fulltext based on Bi-LSTM-CRF. *IEEE/Association for Computing Machinery Transactions on Computational Biology and Bioinformatics*, 18(1):205–215.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL)*, pages 4171-4186.

Jenny Rose Finkel, Trond Grenager, and Christopher Manning. 2005. Incorporating non-local information into information extraction systems by Gibbs sampling. In *Proceedings of the 43rd Annual Meeting of the Association for Computational Linguistics (ACL)*, pages 363-370.

Jerome H Friedman. 2001. Greedy function approximation: a gradient boosting machine. *Annals of statistics*, pages 1189-1232.

Emma J. Gerritse, Faegheh Hasibi, and Arjen P. de Vries. 2022. Entity-aware transformers for entity search. In *Proceedings of the 45th International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR*, SIGIR '22, page 1455–1465.

Daniel Gillick, Sayali Kulkarni, Larry Lansing, Alessandro Presta, Jason Baldridge, Eugene Ie, and Diego Garcia-Olano. 2019. Learning dense representations for entity retrieval. In *Proceedings of the 23rd Conference on Computational Natural Language Learning (CoNLL)*, pages 528-537.

Suchin Gururangan, Ana Marasović, Swabha Swayamdipta, Kyle Lo, Iz Beltagy, Doug Downey, and Noah A. Smith. 2020. Don't stop pretraining: Adapt language models to domains and tasks. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics (ACL)*, pages 8342–8360.

Faegheh Hasibi, Krisztian Balog, and Svein Erik Bratsberg. 2015. Entity linking in queries: Tasks and evaluation. In *Proceedings of the 2015 International Conference on The Theory of Information Retrieval (ICTIR)*, pages 171–180.

Faegheh Hasibi, Krisztian Balog, and Svein Erik Bratsberg. 2016. Exploiting entity linking in queries for entity retrieval. In *Proceedings of the 2016 ACM on International Conference on the Theory of Information Retrieval*, ICTIR '16, pages 209–218.

Johannes Hoffart, Yasemin Altun, and Gerhard Weikum. 2014. Discovering emerging entities with ambiguous names. In *Proceedings of the 23rd International Conference on World Wide Web (WWW)*, pages 385-396.

Subhradeep Kayal, Zubair Afzal, George Tsatsaronis, Marius Doornenbal, Sophia Katrenko, and Michelle Gregory. 2019. A framework to automatically extract funding information from text. In *Machine Learning, Optimization, and Data Science*, pages 317-328.

Belinda Z. Li, Sewon Min, Srinivasan Iyer, Yashar Mehdad, and Wen-tau Yih. 2020. Efficient one-pass end-to-end entity linking for questions. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 6433-6441.

Rodrigo Nogueira, Wei Yang, Kyunghyun Cho, and Jimmy Lin. 2019. Multi-stage document ranking with bert. *arXiv preprint arXiv:1910.14424*.

Mingyue Shang, Tong Wang, Mihail Eric, Jiangning Chen, Jiyang Wang, Matthew Welch, Tiantong Deng, Akshay Grewal, Han Wang, Yue Liu, Yang Liu, and Dilek Hakkani-Tur. 2021. Entity resolution in open-domain conversations. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 26–33.

Set	#Articles$_{EL}$	#Links$_{EL}$	#Links$_{ED}$	NIL$_{ED}$%
Train	22,720	67,671	95,761	18.58%
Val	991	4,333	5,618	15.47%
Test	3,943	16,355	19,765	15.56%
Eval	13,851	37,495	52,378	18.83%

Table 5: Overview of ELFund and EDFund datasets.For each split, number of articles, number of mentions used in training the NER models, number of mention-entity pairs used in training the ED models, and the percentage of NIL mentions among those pairs are shown.

Erik F. Tjong Kim Sang and Fien De Meulder. 2003. Introduction to the CoNLL-2003 shared task: Language-independent named entity recognition. In *Proceedings of the Seventh Conference on Natural Language Learning at HLT-NAACL 2003*, pages 142-147.

Ricardo Usbeck, Michael Röder, Michael Hoffmann, Felix Conrads, Jonathan Huthmann, Axel-Cyrille Ngonga Ngomo, Christian Demmler, and Christina Unger. 2019. Benchmarking question answering systems. *Semantic Web*, 10(2):293-304.

Johannes M. van Hulst, Faegheh Hasibi, Koen Dercksen, Krisztian Balog, and Arjen P. de Vries. 2020. REL: an entity linker standing on the shoulders of giants. In *Proceedings of the 43rd International Association for Computing Machinery SIGIR Conference on Research and Development in Information Retrieval (SIGIR)*, pages 2197–2200.

Ledell Wu, Fabio Petroni, Martin Josifoski, Sebastian Riedel, and Luke Zettlemoyer. 2020. Scalable zero-shot entity linking with dense entity retrieval. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 6397-6407.

Chenyan Xiong, Jamie Callan, and Tie-Yan Liu. 2017. Word-Entity Duet Representations for Document Ranking. In *Proceedings of the 40th International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR)*, pages 763–772.

A Dataset Statistics

The ELFund and EDFund datasets are created based on scientific articles published before 2017. Expert annotators were asked to identify sentences that contain funding organizations of the research (e.g., X was funded by source Y) and link the organizations to entities in the Crossref KB. ELFund, further contains sentences that could be also automatically identified by a classifier. Both datasets were annotated by two experts to find the boundary of mentions and their corresponding entities if available. Disagreements were resolved by a third annotator, and mentions with out-of-KB entities are annotated with NIL. We note that ELFund is not a subset of EDFund.

B Training Configuration

We train the case-preserving version of $\text{BERT}_{\text{BASE}}$with 2M sentences containing funding information to obtain the $\text{BERT}_{\text{TAPT}}$ model. Unless indicated otherwise, the hyper-parameters recommended by Gururangan et al. (2020) are used for training. The training is done on an NVIDIA Tesla K80 GPU with 12 GB of memory with a batch size of 2048 through gradient accumulation and for one epoch (1000 steps). We further fine-tune $\text{BERT}_{\text{TAPT}}$for the mention detection task on the ELFund dataset. The fine-tuning process is done for 3 epochs with batch size of 8. We refer to this model as $\text{BERT}^{\text{MD}}_{\text{TAPT}}$. For disambiguation, the bi-encoder model is trained on the EDFund dataset with a learning rate of 2×10^{-5} and batch size of 16. The training is performed in 4 rounds, each round consisting of 2 epochs. In the first round, 3 random negatives are used for each mention. The score threshold τ is set to 0.042 using grid search. Following (Wu et al., 2020), the mention and entity representations are limited to 64 and 256 tokens, respectively.

KiPT: Knowledge-injected Prompt Tuning for Event Detection

Haochen Li[1] **, Tong Mo**[1]**, Hongcheng Fan**[1]**, Jingkun Wang**[1]**,**
Jiaxi Wang[1]**, Fuhao Zhang**[2] **and Weiping Li**[1,*]
[1]Peking University, Beijing, China
[2]Chinese Academy of Surveying and mapping, Beijing, China
{haochenli, wjk, wangjxi, fanhongcheng}@pku.edu.cn
{wpli, motong}@ss.pku.edu.cn, zhangfh@ac.cn

Abstract

Event detection aims to detect events from the text by identifying and classifying event triggers (the most representative words). Most of the existing works rely heavily on complex downstream networks and require sufficient training data. Thus, those models may be structurally redundant and perform poorly when data is scarce. Prompt-based models are easy to build and are promising for few-shot tasks. However, current prompt-based methods may suffer from low precision because they have not introduced event-related semantic knowledge (e.g., part of speech, semantic correlation, etc.). To address these problems, this paper proposes a Knowledge-injected Prompt Tuning (KiPT) model. Specifically, the event detection task is formulated into a condition generation task. Then, knowledge-injected prompts are constructed using external knowledge bases, and a prompt tuning strategy is leveraged to optimize the prompts. Extensive experiments indicate that KiPT outperforms strong baselines, especially in few-shot scenarios.

1 Introduction

Events describe state changes of participating entities. The Event Detection (ED) task is one of the essential tasks in the Information Extraction field. Event triggers are the most representative words or phrases in events, and they are usually composed of verbs or nouns. There is a one-to-one correspondence between events and event triggers, so the ED task is equivalent to identifying and classifying event triggers.

The ED task has a wide range of applications, providing helpful information for downstream tasks such as text summarization, auto summarization, machine question and answer (QA), etc. Meanwhile, with the vigorous development of Internet news and social media, ED has become a practical approach for extracting information from massive texts. Therefore, the ED task has attracted increasing attention with great academic and applied value in recent years.

Most current ED models use a pre-trained language model to build complex downstream networks (including CNN, RNN, GCN, etc.) These methods perform well on public datasets, but they rely heavily on the fine-tuning strategy to train their downstream networks and introduce massive extra parameters. However, due to the scarcity and uneven distribution of the annotated data for ED, these methods may cause severe overfitting problems. Further, these methods may perform poorly in data-scarce scenarios because their extra parameters cannot be fully optimized.

Recently, prompt-based learning methods is a new trend in natural language processing. Researchers verified that pre-trained language models already have enough knowledge, so the complex downstream networks are unnecessary in many cases. Prompt-based learning methods make full use of the information in pre-trained language models by constructing prompts to guide the language model to solve NLP tasks. Specifically, prompt-based methods first transform the original input into prompt templates containing the initial input, the prompt tokens, and unfilled slots for output. Then, a pre-trained language model is employed to fill the unfilled slots to obtain a final string from which the final output can be derived.

Prompt learning-based methods eliminate complex downstream networks and massive extra parameters, so they have advantages in data scarcity scenarios. Meanwhile, the prompts' quality directly affects the models' performance. However, manually selecting the optimal prompts is time-consuming and labor-intensive. Considering this, prompt tuning strategies have been proposed by introducing continuous virtual tokens as trainable prompts that will be optimized through training.

*Weiping Li is the corresponding author.

Proceedings of the 29th International Conference on Computational Linguistics, pages 1943–1952
October 12–17, 2022.

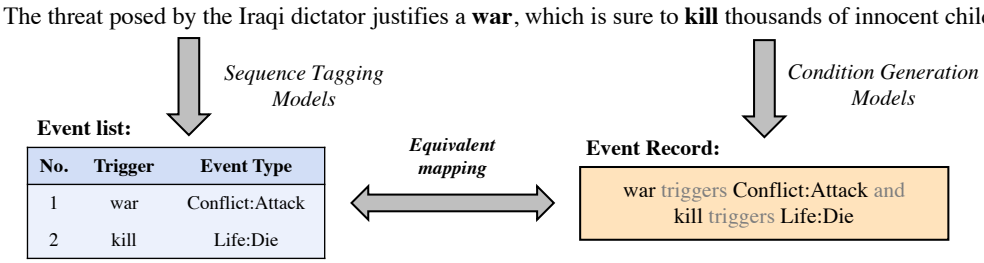

Figure 1: An example of Event Detection (on the left is the event list output by the sequence tagging model; on the right is the event record output by the condition generation model, where the gray words act as structural tokens)

However, the above prompts may suffer from low precision because they have not introduced event-related semantic knowledge.

Event triggers are mostly verbs and nouns, and are usually semantically related to the core concept words of events. Therefore, semantic knowledge (such as part of speech, word semantic correlation, etc.) plays a crucial role in the ED task. To this end, this paper proposes a Knowledge-injected Prompt Tuning (KiPT) model to introduce event-related knowledge using external knowledge bases.

Specifically, we formulate the ED task into a condition generation task. Then, external knowledge bases and semantic tools are used to obtain the semantic knowledge associated with input sentences and events. Next, the semantic knowledge is injected into the prompts for ED. Finally, we use the knowledge-injected prompts to extract event triggers, and the prompts will be optimized through a prompt tuning strategy. Our method is direct and effective, and it can be easily transferred to other tasks.

The main contributions of this paper are summarized as follows:

- We introduce a knowledge injection method to inject event-related semantic knowledge into the prompt templates, which is the first in the ED task to the best of our knowledge;
- We propose a prompt-based learning model for ED called Knowledge-injected Prompt Tuning (KiPT), which leverages a prompt tuning strategy to optimize the prompts;
- Extensive experiments show that our model outperforms current prompt-based ED models and strong baselines, especially in data scarcity scenarios.

2 Related Work

Studies related to our work are mainly discussed from the following three aspects:

2.1 Event Detection Methods

The ED models can be divided into sequence tagging models and condition generation models.

Chen et al. and Nguyen et al. formulated ED as a sequence tagging task for the first time, and they used CNN and RNN to model sentence-level features; Liu et al. and Yan et al. used GCN to emphasize the semantic dependency. Nguyen and Nguyen jointly extracted entities, triggers, and arguments based on the shared hidden representations; Wadden et al. provided a graph propagation method to capture context relevant for entity, relation, and event; Lin et al. built an end-to-end information extraction system which employs global feature and beam search to extract globally optimal event structures;

Condition generation methods encode sentences using generative pre-trained language models such as BART(Lewis et al., 2019) and T5(Raffel et al., 2019). Li et al. utilized a conditional generation model with BART. Paolini et al. regarded event extraction as a translation task between augmented natural languages; Lu et al. constructed events as event trees and used a Seq2Structure model.

2.2 Prompt-based Learning Methods

Prompt-based learning methods use prompts to guide pre-trained language models to generate results, so the quality of the prompt templates is critical. The current prompt-base learning templates include:

Manually setting discrete prompt templates (actual words in the template): Schick and Schütze transferred the text classification task into a cloze-filling task by using manual prompt templates.

Petroni et al. converted triple completion tasks into cloze-filing questions through prompt templates.

Building trainable continuous prompt templates (virtual tokens in the template): Li and Liang used trainable prefix tokens as prompts and added soft tokens in each layer of the language model. Liu et al. replaced actual words with trainable soft tokens in the prompt template and introduced an extra prompt encoder.

However, neither of these methods introduces task-related knowledge, and they cannot optimize the prompt along with external knowledge.

2.3 Prompt-based Knowledge Injection

Some works have already attempted to introduce external knowledge in prompt-based learning. Hu et al. enhanced the mapping of model outputs and predefined categories by introducing extra knowledge in verbalizers. Chen et al. introduced virtual tokens to enhance category features for relation extraction tasks. Li et al. strengthened the model by introducing ground truths. Although these methods are instructive, they have strong task dependencies and are difficult to apply to Event Detection.

Our work aims to perform the ED task using a prompt-based model with event-related knowledge. To achieve this, a knowledge-injected prompt tuning method is proposed in this paper.

3 Methodology

This section first describes the definition of the ED task. Then, the proposed KiPT model is introduced in detail.

3.1 Task Description

Following the task description of Automatic Content Extraction [1], the standard task of ED includes event Trigger Identification (Trig-I) and Trigger Classification (Trig-C). Consider the example in Figure 1. After obtaining the input sentence, ED methods should first identify the triggers **"war"** and **"kill"** and then classify them into event types **"Conflict:Attack"** and **"Life:Die"**.

This paper formulates the standard ED task into a condition generation task to simplify the output. First, the structural word "triggers" is used to combine a trigger and its corresponding type like "**war** triggers **Conflict:Attack**". Then, if a sentence contains more than one event, the events are concatenated with another structural word "and". Considering the example in Figure 1, our constructed event record is **"war** triggers **Conflict:Attack** and **kill** triggers **Life:Die**". Therefore, given the input sentence, our model generates the above event record as output. It is worth mentioning that the output of our method is equivalent to the original ED task since event records can be easily split into event triggers and event types.

[1] https://www.ldc.upenn.edu/collaborations/past-projects/ace/annotation-tasks-and-specifications

3.2 The Overall Structure of KiPT

The overall structure of KiPT is shown in Figure 2. First, given the input sentence x, the knowledge-injected prompt $Prompt(x)$ is constructed. Then, prompt template input T is built and fed into the pre-trained language model $\mathcal{LM}$ to obtain the output event record y.

We propose a trainable knowledge-injected prompt $Prompt(x)$ in KiPT. $Prompt(x)$ includes two parts: input-related **knowledge injection** $K(x)$ and input-irrelated **soft tokens** S. $K(x)$ and S are both trainable and will be optimized during the training process.

In the following subsections, the construction and tuning of the knowledge-injected prompt will be explained in detail.

3.3 Knowledge-injected Prompt Construction

In this section, the definitions of the knowledge injection $K(x)$ and the soft tokens S are introduced first. Then, the combination of $K(x)$ and S into the prompt $Prompt(x)$ is described.

Knowledge Injection $K(x)$: Given each input sentence x, the token sequence $x_{1:n} = \{x_1, x_2, ..., x_n\}$ can be obtained by using a tokenizer. Then, a knowledge extractor is used to extract event-related knowledge $K(x)$ for x, and the knowledge extractor is constructed by using NLP analysis tools and external knowledge bases.

First, since event triggers are mostly verbs and nouns, each token's part of speech (POS) is essential. Thus, POS analysis is performed on each token x_i using a semantic analysis tool (**Stanford's Stanza**(Qi et al., 2020)). Then, Stanza's lemmatization module is employed to recover the lemma form $\hat{x_i}$ of each input token (for example, "died" -> "die"). After the POS and the lemma form of each token are obtained, the verbs and nouns are selected as a list of potential triggers $PT_{1:|PT|} = \{pt_1, pt_2, ..., pt_{|PT|}\}$.

Semantic correlations in knowledge bases also provide vital information for ED. Each type of

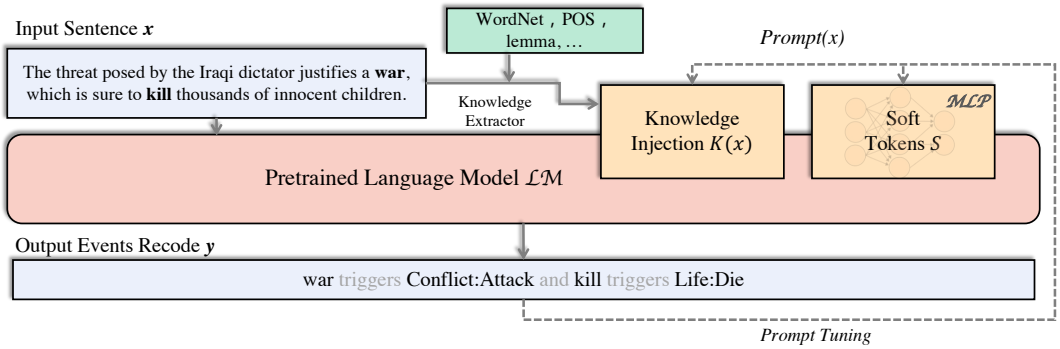

Figure 2: The overall architecture of KiPT. Given the input sentence x, external knowledge are used to construct input-related prompt $Prompt(x)$. Then, the language model $\mathcal{LM}$ is used to generate output event record y.

event has some core concepts, e.g., the event type "Conflict: Attack" has concepts like "attack, fight, bomb, etc.", and the event type "Life: Die" has concepts like "kill, suicide, murder, etc." To exploit the semantic relevance of the core concepts of words and events, we make the following assumption:

Assumption 3.1 *If a word has strong semantic correlations with the core concepts of a specific event type, then the word is likely to trigger an event of that type.*

Based on the above assumption, a dictionary $Concept_{1:e}$ containing the core concepts of each event type (e represents the total number of event types) is manually constructed, and $concept_k$ represents the concept of the kth type of event. Then, **WordNet**(Fellbaum and Miller, 1998) is introduced to obtain the semantic correlation between the input tokens and the core concepts, and whether they have a strong semantic correlation is judged by calculating the semantic similarity of two words in WordNet. For each word pt_j in the list of potential triggers PT and the core concept $concept_k$ for each event type, their semantic correlation $Sc(pt_j, concept_k)$ is calculated by using the *Wu-Palmer Similarity Algorithm*(Wei et al., 2015). If the semantic correlation $Sc(pt_j, concept_k)$ is above the similarity threshold θ_{sim}, the potential trigger word pt_j and the corresponding event type $event_k$ are both added to $K(x)$.

To sum up, the process of the knowledge extractor is as follows:

The final extracted knowledge injection $K(x)$ is a text sequence composed of potential triggers and their potential event types. It is denoted as $K(x)_{1:|K|} = \{k_1, k_2, ..., k_{|K|}\}$, where $|K|$ stands for the length of K. Further, the similarity threshold θ_{sim} may affect the performance of our model, which will be discussed in the Appendix.

Algorithm 1 Knowledge Extractor

Input: Sentence: $x_{1:n}$, Concept dict: $Concept_{1:e}$
Output: Knowledge Injection: $K(x)$
for $x_i \in x$ **do**
  if $POS(x_i) \in [verb, noun]$ **then**
    Add $lemma(x_i)$ to potential triggers PT
  end if
end for
for $pt_j \in PT$ **do**
  for $concept_k \in Concept$ **do**
    if $Sc(pt_j, concept_k) > \theta_{sim}$ **then**
      Add $[pt_j, event_k]$ to $K(x)$
    end if
  end for
end for

Soft Tokens S: Besides using knowledge injection $K(x)$, this paper also adds some trainable soft tokens to the prompts. Soft tokens are virtual tokens sharing the same dimension as actual words (e.g, 768 for T5-base) but without real meanings. Previous works have proved that trainable soft prompts are more flexible and effective than actual words (Liu et al., 2021; Li and Liang, 2021).

In our method, randomly initialized tensors are used as the soft prompt tokens, which will be optimized during the training of the language model. The soft tokens used in this paper are denoted as $S_{1:p} = \{s_1, s_2, ..., s_p\}$. The selection of p may slightly affect the performance of our model, and this will be discussed in Appendix.

Prompt Templates Construction: After knowledge injection $K(x)$ and soft tokens S are

obtained, they are concatenated as $Prompt(x)$:

$$\begin{aligned} Prompt(x) &= [K(x); S] \\ &= \{k_1, ..., k_{|K|}, s_1, ..., s_p\} \end{aligned} \tag{1}$$

For each input x, $Prompt(x)$ is constructed as the knowledge-injected prompt. Then, a prompt template is built, which contains the input x, $Prompt(x)$, and the target event record y.

$$\textbf{Template}: Prompt(x)\ [x],\ Events:\ [y] \tag{2}$$

Where $[x]$ represents the slot for the input sentence, $[y]$ represents the slot for the target event records, and "$Events:$" is a fixed anchor token. Anchor tokens have been proved useful by previous works (Li and Liang, 2021; Han et al., 2021).

3.4 Knowledge-injected Prompt Tuning

The knowledge-injected prompt $Prompt(x)$ needs to be optimized through training mainly for the following two reasons:

(1) Some rule-based algorithms are used during the construction of the knowledge injection $K(x)$. However, these rules may be ineffective or even wrong in some cases, so these rules need to be softened through training;

(2) Soft tokens S are virtual tensors that are randomly initialized and have no original semantics. They need to be trained to approximate the distribution of actual words to achieve the role of prompts for language models.

To this end, we propose knowledge-injected prompt tuning to optimize $Prompt(x)$. Given a pre-trained language model $\mathcal{LM}$ and its vocabulary $\mathcal{V}$, the prompt template's input T is:

$$\begin{aligned} T &= [H^k; H^s; e(x)] \\ &= \{h_1^k, ..., h_{|K|}^k, h_1^s, ..., h_p^s, e(x_1), ..., e(x_n)\} \end{aligned} \tag{3}$$

where $e(x_i)$ indicates the embeddings for the input tokens; h_i^s and h_i^k stand for embedded prompts for the knowledge injection and the soft tokens, respectively. Note that $e(x_i)$ and h_i^k are initialized using the embeddings of the actual tokens from the $\mathcal{LM}$'s vocabulary $\mathcal{V}$, and h_i^s indicates randomly initialized tensors. The conditional probability of the event record output can be obtained by using the generative language model $\mathcal{LM}$.

Finally, given the golden event record y, gradient updates are performed by using the following log-likelihood loss function:

$$L = -\sum_{(x,y)\in\mathcal{D}} \log(y|H^k, H^s, e(x), \theta_{\mathcal{LM}}) \tag{4}$$

where $\mathcal{D}$ stands for the whole training dataset, and $\theta_{\mathcal{LM}}$ stands for the $\mathcal{LM}$'s parameters.

Previous research (Liu et al., 2021) has shown that there are semantic gaps between the embeddings of actual tokens and virtual tokens. Because the embeddings of actual words are highly discrete through pre-training, trainable soft prompt tokens are randomly distributed. They will only change in a small neighborhood during stochastic gradient descent.

Thus, following Liu et al.'s work, we use a lite muti-layer linear network ($\mathcal{MLP}$) as a prompt encoder to narrow the semantic distance between the embedding of actual words and prompt tokens:

$$\hat{h_i^s} = \mathcal{MLP}(h_i^s), i \in [1, p] \tag{5}$$

The loss function is improved as follows:

$$L = -\sum_{(x,y)\in\mathcal{D}} \log(y|H^k, H^s, e(x), \theta_{\mathcal{LM}}, \theta_{\mathcal{MLP}}) \tag{6}$$

where $\theta_{\mathcal{MLP}}$ stands for the parameters of the muti-layer linear network $\mathcal{MLP}$.

4 Experiments

This section describes our experimental settings and detailed experimental analysis.

First, we conduct overall experiments to verify the performance of KiPE with sufficient training data. Next, we conduct few-shot and zero-shot experiments to verify the performance of KiPT in data-scarce scenarios. Finally we perform an ablation analysis to explore the influence of each part in our prompts.

4.1 Experiment Setup

Datasets. Our work is evaluated on the most widely used datasets ACE 2005 (Automatic Content Extraction program of 2005)[2] and TAC 2015 (Event Nugget data of TAC 2015)[3]. The detailed descriptions of the datasets are presented in Table 1:

As for ACE 2005, following the previous works (Yan et al., 2019; Nguyen and Nguyen, 2019), the 599 documents are divided into 529 training documents, 30 development documents, and 40 test documents. And for TAC 2015,following Lu et al., the 458 documents are divided into 396 training documents, 31 development documents, and 31 test documents.

[2]https://catalog.ldc.upenn.edu/LDC2006T06
[3]https://catalog.ldc.upenn.edu/LDC2020T13

Datasets	ACE 2005	TAC 2015
Event Type	33	38
Total Documents	599	458
Total Events	5,055	7,530
Train/Dev/Test Split	529/30/40	396/31/31

Table 1: Statistics of ACE 2005 and TAC 2015

Metrics. The ED task has two subtasks: **Trigger Identification (Trig-I)** and **Trigger Classification (Trig-C)**. Their standard evaluation criteria are as follows:

- **Trig-I**: An event trigger is identified correctly if its span matches the gold trigger;
- **Trig-C**: An event trigger is classified correctly if both its span and event type match the gold trigger.

We use micro-averaged Precision (**P**), Recall (**R**), and F1 score (**F1**) in all the following evaluations.

Settings. In this paper, T5 (Yang et al., 2019) is utilized as the pre-trained language model $\mathcal{LM}$ in KiPT. Both T5-Base and T5-Large are used in the overall results, while only T5-Base is used in the rest of the experiments. Our model is optimized with AdamW for 30 epochs with a learning rate of 1e-4 and a weight decay of 1e-5 for T5, and a learning rate of 1e-3 and a weight decay of 1e-4 for other parameters. The batch size is set to 16 for T5-base and 8 for T5-large.

In our main KiPT model, the number of soft tokens p is set to 40, and the similarity threshold θ_{sim} is set to 0.8. The performance of the other options will be discussed in Appendix.

Baselines. This paper chooses 10 strong baselines for comparison: (1) Three sequence tagging models: **Joint3EE** jointly extracts entities, triggers, and arguments based on the shared hidden representations; **DYGIE++** provides a graph propagation method to capture relevant context for entity, relation, and event; **OneIE** builds an end-to-end information extraction system that employs global feature and beam search to extract globally optimal event structures; (2) Two QA-based models: **BERT_QA** first formulates ED as a QA task and generates questions from annotation guidelines; **MQAEE** uses multi-turn question strategy to build questions; (3) Three condition generation models: **BART-Gen** utilizes a conditional generation model with BART. Given the description of events, the corresponding triggers in the sentence are generated; **TANL** frames ED as a translation task between augmented natural languages; **Text2Event** constructs event structure and uses a Seq2Structure model. (4) Two prompt-based learning models: **PoKE** presents various joint prompt methods, which can elicit more complementary knowledge by modeling the interactions between different triggers or arguments; **GDAP** empowers the automatic exploitation of label semantics on prompt templates.

4.2 Overall Results

The overall results of our model on ACE 2005 and TAC 2015 are presented in Table 2 and Table 3, respectively. For both Trig-I and Trig-C tasks, KiPT (T5-large) outperforms all strong baselines on ACE 2005, reaching F1 values of 78.6% and 75.3%, respectively. Due to the reduced parameters of the pre-trained language model, the performance of KiPT (T5-base) drops slightly (-1.1% in Trig-I and -0.4% in Trig-C), but it still exceeds most of the baselines. On TAC 2015, OneIE and Text2Event are compared because they are the only two baselines experimented on TAC 2015. Although our model is slightly lower than OneIE in Trig-I, it outperforms all baselines in Trig-C.

From the overall results, it can be seen that:

(1) Our model significantly outperforms other prompt-based models (PoKE and GDAP) because of the utilization of event-related knowledge. The results indicate that prompt-based learning models tend to achieve higher Recall but lower Precision in both Trig-I and Trig-C tasks. We believe that prompt-based learning models only use the knowledge of pre-trained language models, which is more general but lacks task specificity.

KiPT improves this problem significantly by introducing event-related knowledge through knowledge injection, narrowing the gap between Precision and Recall. In the Trig-C task, the gap between Recall and Precision of KiPT (T5-base) is 4.0%, while that of PoKE and GDAP is 12.1% and 9.2%, respectively. That is why KiPT outperforms other prompt-based learning models.

Further, KiPT(T5-base) has better precision than KiPT(T5-large) but has lower recall and F1. The possible reason is that T5-large contains more general knowledge and reduces the proportion of event-related knowledge we introduced.

(2) Without introducing complex downstream networks, KiPT (T5-base) outperforms the best T5-base model PoKE by 5.3% in F1, and KiPT

Models	Trig-I			Trig-C			PLM
	P	R	F1	P	R	F1	
Sequence Tagging Models							
Joint3EE (Nguyen and Nguyen, 2019)	70.5	74.5	72.5	68.0	71.8	69.8	-
DYGIE++ (Wadden et al., 2019)	-	-	76.5	-	-	73.6	BERT-large
OneIE (Lin et al., 2020)	-	-	**78.6**	-	-	75.2	BERT-large
QA Models							
BERT_QA (Du and Cardie, 2020)	74.3	77.4	75.8	71.1	73.7	72.4	2×BERT-base
MQAEE (Li et al., 2020)	-	-	77.4	-	-	73.8	3×BERT-large
Condition Generation Models							
BART-Gen (Li et al., 2021b)	-	-	74.4	-	-	71.1	BART-large
TANL (Paolini et al., 2021)	-	-	-	-	-	68.5	T5-base
Text2Event (Lu et al., 2021)	-	-	-	69.6	74.4	71.9	T5-large
Prompt-based Learning Models							
PoKE (Lin et al., 2021)	-	-	-	64.1	76.2	69.6	T5-base
GDAP (Si et al., 2022)	-	-	-	65.6	74.7	69.9	T5-large
Our Model							
KiPT (T5-base)	**76.0**	79.1	77.5	**72.9**	76.9	74.9	T5-base
KiPT (T5-large)	75.4	**82.1**	**78.6**	71.6	**79.2**	**75.3**	T5-large

Table 2: Experimental results on ACE 2005. Trig-I indicates trigger identification tasks and Trig-C indicates trigger classification tasks. The column PLM indicates the pre-trained language models used in each model.

Model	Trig-I	Trig-C
OneIE	**68.4**	57.0
Text2Event	-	57.8
KiPT (T5-base)	66.3	58.1
KiPT (T5-large)	67.0	**58.3**

Table 3: Experimental results on TAC 2015. The F1 score is recorded for each model.

$k-shot$	OneIE	Text2Event	PoKE	KiPT
$k=4$	22.8	39.6	44.2	**45.5**
$k=8$	25.2	51.2	**53.1**	50.1
$k=16$	28.6	52.1	53.8	**54.2**
$k=32$	39.7	53.7	55.3	**56.4**
$k=64$	48.6	57.8	59.1	**63.6**
All Data	**75.2**	71.9	69.6	74.9

Table 4: Experiment with few-shot settings. The average F1 scores of 10 experiments are used for each model.

(T5-large) outperforms the best T5-large model Text2Event by 3.4%. This proves the effectiveness of our prompt tuning strategy.

(3) Our model has a relatively small F1 drop from Trig-I to Trig-C: -2.6% for KiPT (T5-base) and -2.3% for KiPT (T5-large). As a comparison, the average drop of the QA model is -3.5% and that of the condition generation model is -3.3%. This indicates that KiPT binds potential triggers and their corresponding event types together through knowledge injection, so it is easier for it to classify trigger words correctly after identifying them.

Among all the baselines, OneIE has a close performance to KiPT, indicating that sequence tagging models still have competitive performance with enough training data. However, the introduction of complex downstream networks and massive parameters may face struggles when data is insufficient. Next, we will perform data-scarce experiments.

4.3 Few-shot and Zero-shot Scenarios

To verify the advantages of KiPT in low-resource settings, we conduct few-shot and zero-shot experiments on ACE 2005. Three strong baselines are compared: sequence tagging model OneIE, condition generation model Text2Event, and prompt-based learning model PoKE.

Few-shot experiments: Referring to the few-shot settings of previous works (Gao et al., 2020; Lin et al., 2021), 4, 8, 16, 32, and 64 shot experiments are conducted to compare the performance of KiPT and baselines under scenarios with small data resources. Specifically, for each type of event, k samples are randomly selected from the initial training set. Then, after training, the models' per-

Models	Trig-I	Trig-C
OneIE	37.6	34.7
Text2Event	45.1	38.4
Poke	44.7	39.3
KiPT	**44.9**	**42.3**

Table 5: Experiment with zero-shot settings. The F1 scores for Trig-I and Tri-C are on 23 unseen event types.

Models	F1 score for Trig-C
KiPT (T5-base)	**74.9**
- $\mathcal{MLP}$	73.0 (-1.9)
- S	73.6 (-1.3)
- $K(x)$	72.2 (-2.7)
- $K(x)$ and S	71.0 (-3.9)

Table 6: Ablation study results on ACE 2005. '-' means the removal of the corresponding component.

formance is tested on the standard test set. Each $k-shot$ experiment is repeated 10 times, and the average results are recorded finally.

The results are shown in Table 4. It can be seen that KiPT outperforms all the strong baselines in most few-shot settings. Further analysis indicates that:

(1) Our model is still effective when the data is particularly sparse. For example, in the setting of $k = 4$, KiPT reaches 45.5% in F1 score, significantly surpassing all baselines;

(2) The lead of current prompted-based models will gradually shrink as the training data increase. For example, PoKE outperforms Text2Event by 4.6% and 1.3% when $k = 4$ and $k = 64$, respectively. However, benefiting from the trainable knowledge-injected prompts, KiPT's performance grows uniformly as the data increase. In detail, KiPT surpasses Text2Event by 5.9% and 5.8% when $k = 4$ and $k = 64$, respectively.

(3) OneIE perform poorly in all few-shot scenarios. This indicates that the models using complex downstream networks and extra parameters require sufficient training data to achieve their potential.

Zero-shot experiments: Following the settings of Lu et al., we selected the top 10 most popular event types as seen types and tested the zero-shot classification performance for the remaining 23 unseen types. The results are shown in Table 5, where KiPT overpasses all baselines, especially in the Trig-C task. This indicates that even for a new unseen event type, we can also obtain its event-related knowledge through our prompts, proving the transfer potential of our model.

4.4 Ablation Study

An ablation study is conducted to verify the effectiveness of each component of KiPT. Specifically, four groups of controlled experiments are designed: "$-\mathcal{MLP}$" means removing the prompt encoder proposed in equation 5; "$-S$" means only using knowledge injection in prompt construction; "$-K(s)$" means only using soft tokens as prompts; "$-K(x)$ and S" means removing all the prompts. The results of the ablation study are presented in Table 6. It can be seen that:

(1) When $\mathcal{MLP}$ is removed, the model's performance drops slightly (-1.9%). This indicates that although soft tokens can be optimized by training, the gap between virtual tensors and actual words may still cause an adverse effect, showing the necessity of our prompt encoder.

(2) Knowledge injection plays a more significant role than soft tokens within prompts. The removal of $K(x)$ caused a performance drop of 2.7%, over twice the removal of S (1.3%). This indicates that introducing additional semantic knowledge for prompt-based models is better than adding randomly initialized tokens.

(3) After all the prompts are removed, our model downgrades into a simple condition generation model, which causes a significant performance drop (-3.9%). Because we does not design downstream networks, the performance of the downgraded model is lower than that of condition generation baselines (Text2Event and BART-Gen). This validates the effectiveness of our knowledge-injected prompt tuning strategy.

5 Conclusion

This paper proposes a prompt-based learning method for ED by introducing knowledge-injected prompt tuning. External knowledge and soft tokens are used to construct knowledge-injected prompts, which can be optimized through training. Comprehensive experiments demonstrate that KiPT outperforms current prompt-based ED models and strong baselines, especially in data-scarce scenarios. Through our method, prompt-based models can introduce task-related knowledge more conveniently and effectively. In the future, we will explore more knowledge injection approaches and their applications in other tasks.

6 Acknowledgments

This work is supported by the National Key Research and Development Program of China (No. 2020YFC0833300).

References

Xiang Chen, Ningyu Zhang, Xin Xie, Shumin Deng, Yunzhi Yao, Chuanqi Tan, Fei Huang, Luo Si, and Huajun Chen. 2022. KnowPrompt: Knowledge-aware Prompt-tuning with Synergistic Optimization for Relation Extraction. *arXiv:2104.07650 [cs]*.

Yubo Chen, Liheng Xu, Kang Liu, Daojian Zeng, and Jun Zhao. 2015. Event Extraction via Dynamic Multi-Pooling Convolutional Neural Networks. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 167–176, Beijing, China. Association for Computational Linguistics.

Xinya Du and Claire Cardie. 2020. Event Extraction by Answering (Almost) Natural Questions. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 671–683, Online. Association for Computational Linguistics.

C. Fellbaum and G. Miller. 1998. *WordNet : an electronic lexical database*. WordNet:An Electronic Lexical Database.

T. Gao, A. Fisch, and D. Chen. 2020. Making pre-trained language models better few-shot learners.

Xu Han, Weilin Zhao, Ning Ding, Zhiyuan Liu, and Maosong Sun. 2021. PTR: Prompt Tuning with Rules for Text Classification. *arXiv:2105.11259 [cs]*.

Shengding Hu, Ning Ding, Huadong Wang, Zhiyuan Liu, Juanzi Li, and Maosong Sun. 2021. Knowledgeable Prompt-tuning: Incorporating Knowledge into Prompt Verbalizer for Text Classification. *arXiv:2108.02035 [cs]*.

Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Ves Stoyanov, and Luke Zettlemoyer. 2019. Bart: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. *arXiv preprint arXiv:1910.13461*.

Chengxi Li, Feiyu Gao, Jiajun Bu, Lu Xu, Xiang Chen, Yu Gu, Zirui Shao, Qi Zheng, Ningyu Zhang, Yongpan Wang, and Zhi Yu. 2021a. SentiPrompt: Sentiment Knowledge Enhanced Prompt-Tuning for Aspect-Based Sentiment Analysis. *arXiv:2109.08306 [cs]*.

Fayuan Li, Weihua Peng, Yuguang Chen, Quan Wang, Lu Pan, Yajuan Lyu, and Yong Zhu. 2020. Event Extraction as Multi-turn Question Answering. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 829–838, Online. Association for Computational Linguistics.

Sha Li, Heng Ji, and Jiawei Han. 2021b. Document-Level Event Argument Extraction by Conditional Generation. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 894–908, Online. Association for Computational Linguistics.

Xiang Lisa Li and Percy Liang. 2021. Prefix-Tuning: Optimizing Continuous Prompts for Generation. *arXiv:2101.00190 [cs]*.

J. Lin, J. Jian, and Q. Chen. 2021. Eliciting knowledge from language models for event extraction.

Ying Lin, Heng Ji, Fei Huang, and Lingfei Wu. 2020. A Joint Neural Model for Information Extraction with Global Features. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7999–8009, Online. Association for Computational Linguistics.

Xiao Liu, Kaixuan Ji, Yicheng Fu, Zhengxiao Du, Zhilin Yang, and Jie Tang. 2021. P-Tuning v2: Prompt Tuning Can Be Comparable to Fine-tuning Universally Across Scales and Tasks. *arXiv:2110.07602 [cs]*.

Xiao Liu, Zhunchen Luo, and Heyan Huang. 2018. Jointly Multiple Events Extraction via Attention-based Graph Information Aggregation. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 1247–1256, Brussels, Belgium. Association for Computational Linguistics.

Yaojie Lu, Hongyu Lin, Jin Xu, Xianpei Han, Jialong Tang, Annan Li, Le Sun, Meng Liao, and Shaoyi Chen. 2021. Text2Event: Controllable Sequence-to-Structure Generation for End-to-end Event Extraction. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 2795–2806, Online. Association for Computational Linguistics.

T. H. Nguyen, K. Cho, and R. Grishman. 2016. Joint event extraction via recurrent neural networks. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*.

Trung Minh Nguyen and Thien Huu Nguyen. 2019. One for All: Neural Joint Modeling of Entities and Events. *Proceedings of the AAAI Conference on Artificial Intelligence*, 33:6851–6858.

G. Paolini, B. Athiwaratkun, J. Krone, J. Ma, A. Achille, R. Anubhai, Cnd Santos, B. Xiang, and S. Soatto. 2021. Structured prediction as translation between augmented natural languages.

F. Petroni, T. Rocktschel, P. Lewis, A. Bakhtin, Y. Wu, A. H. Miller, and S. Riedel. 2019. Language models as knowledge bases?

Peng Qi, Yuhao Zhang, Yuhui Zhang, Jason Bolton, and Christopher D. Manning. 2020. Stanza: A Python natural language processing toolkit for many human languages. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics: System Demonstrations*.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J Liu. 2019. Exploring the limits of transfer learning with a unified text-to-text transformer. *arXiv preprint arXiv:1910.10683*.

T. Schick and H Schütze. 2020. Exploiting cloze questions for few shot text classification and natural language inference.

Jinghui Si, Xutan Peng, Chen Li, Haotian Xu, and Jianxin Li. 2022. Generating Disentangled Arguments with Prompts: A Simple Event Extraction Framework that Works. *arXiv:2110.04525 [cs]*.

David Wadden, Ulme Wennberg, Yi Luan, and Hannaneh Hajishirzi. 2019. Entity, Relation, and Event Extraction with Contextualized Span Representations. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 5783–5788, Hong Kong, China. Association for Computational Linguistics.

Tingting Wei, Yonghe Lu, Huiyou Chang, Qiang Zhou, and Xianyu Bao. 2015. A semantic approach for text clustering using wordnet and lexical chains. *Expert Systems with Applications*, 42(4):2264–2275.

H. Yan, X. Jin, X. Meng, J. Guo, and X. Cheng. 2019. Event detection with multi-order graph convolution and aggregated attention. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*.

Sen Yang, Dawei Feng, Linbo Qiao, Zhigang Kan, and Dongsheng Li. 2019. Exploring Pre-trained Language Models for Event Extraction and Generation. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 5284–5294, Florence, Italy. Association for Computational Linguistics.

A Appendix

In the appendix, we discuss the influence of the soft prompt tokens' number p and the similarity threshold θ_{sim} on KiPT.

A.1 The selection of soft tokens' number p

We set up experiments for different numbers of soft tokens for detailed analysis. We set up experiments with soft tokens' number as 0, 10, 20, 40, 80, and 160 on both ACE 2005 and TAC 2015 datasets. The results are shown in Figure 3.

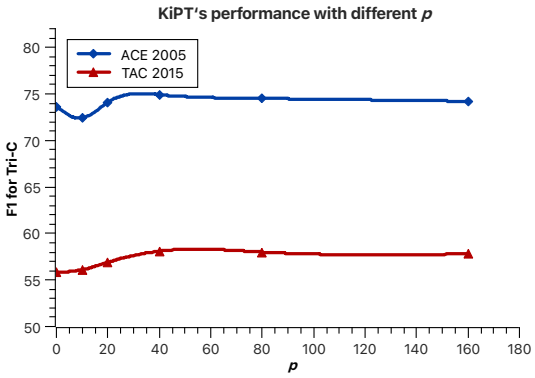

Figure 3: KiPT's performance with different p selection

We can observe that p and the KiPT's performance are not entirely positively correlated, and the model performance peaks when p is around 40. We believe that the small number of soft tokens will make the semantic information captured by the model insufficient, and it is challenging to classify events accurately. On the contrary, introducing too many soft tokens will dilute the original semantic information of the sentence, which will also lead to a decline in the model's performance.

A.2 The selection of similarity threshold θ_{sim}

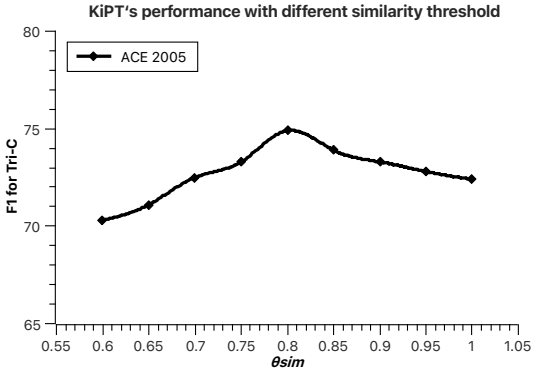

Figure 4: KiPT's performance with different θ_{sim}

The selection of similarity threshold θ_{sim} will directly influence the knowledge injection $K(s)$. Lower θ_{sim} will introduce more knowledge along with more noise, while higher θ_{sim} will inject knowledge more precisely but less broadly. KiPT performs best when θ_{sim} equals 0.8.

OneEE: A One-Stage Framework for Fast Overlapping and Nested Event Extraction

Hu Cao[1]*, Jingye Li[1]*, Fangfang Su[1], Fei Li[1], Hao Fei[2], Shengqiong Wu[2], Bobo Li[1], Liang Zhao[3], Donghong Ji[1]†

[1] Key Laboratory of Aerospace Information Security and Trusted Computing, Ministry of Education, School of Cyber Science and Engineering, Wuhan University, China

[2] School of Computing, National University of Singapore, Singapore

[3] Department of Computing and Mathematics, University of São Paulo, Brazil

{whucaohu, theodorelee, lifei_csnlp, dhji}@whu.edu.cn

Abstract

Event extraction (EE) is an essential task of information extraction, which aims to extract structured event information from unstructured text. Most prior work focuses on extracting flat events while neglecting overlapped or nested ones. A few models for overlapped and nested EE includes several successive stages to extract event triggers and arguments, which suffer from error propagation. Therefore, we design a simple yet effective tagging scheme and model to formulate EE as word-word relation recognition, called OneEE. The relations between trigger or argument words are simultaneously recognized in one stage with parallel grid tagging, thus yielding a very fast event extraction speed. The model is equipped with an adaptive event fusion module to generate event-aware representations and a distance-aware predictor to integrate relative distance information for word-word relation recognition, which are empirically demonstrated to be effective mechanisms. Experiments on 3 overlapped and nested EE benchmarks, namely FewFC, Genia11, and Genia13, show that OneEE achieves the state-of-the-art (SoTA) results. Moreover, the inference speed of OneEE is faster than those of baselines in the same condition, and can be further substantially improved since it supports parallel inference.[1]

1 Introduction

Event Extraction (EE) is a fundamental yet challenging task in information extraction research (Miwa and Bansal, 2016; Katiyar and Cardie, 2016; Fei et al., 2020b; Li et al., 2021b; Fei et al., 2022a). EE facilitates the development of practical applications such as knowledge graph construction (Wei et al., 2019b; Bosselut et al., 2021), biological process analysis (Miwa et al., 2013), and financial market surveillance (Nuij et al., 2013). The goal of EE is to recognize event triggers as well as the associated arguments from texts. As an example, Figure 1(a) illustrates a `Share Reduction` event including a trigger "reduced" and a subject argument "Wang Yawei".

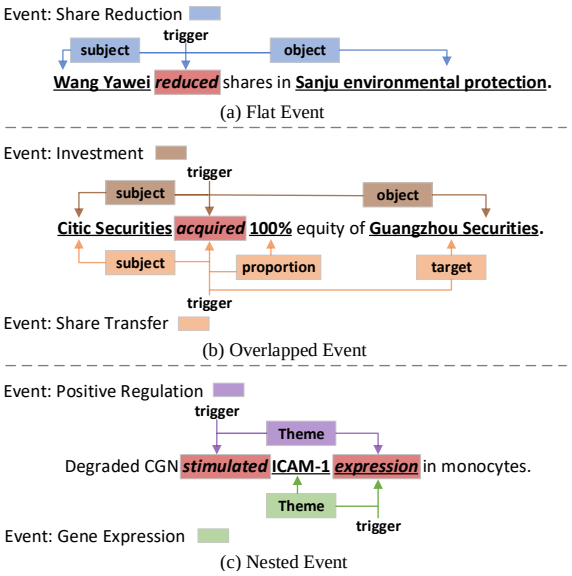

Figure 1: Examples of three kinds of events, including a flat event (a), overlapped events (b), and nested events (c). Different event mentions are denoted in distinct colors. Triggers are marked with red boxes while arguments are underlined.

Traditional methods for EE (Li et al., 2013; Chen et al., 2015; Nguyen et al., 2016; Liu et al., 2018; Nguyen and Nguyen, 2019) regard event extraction as a sequence labeling task, assuming that event mentions do not overlap with each other. However, they neglect complicated irregular EE scenarios (i.e., overlapped and nested EE) (Fei et al., 2020a, 2021a). As exemplified in Figure 1(b), there are two overlapped events, `Investment`, and `Share Transfer`, which share the same trigger word "acquired" and the argument words "Guangzhou Securities". Figure 1(c) illustrates an example of nested events where the event `Gene`

*Equal contribution

†Corresponding author

[1]The codes at https://github.com/Cao-Hu/OneEE

Proceedings of the 29th International Conference on Computational Linguistics, pages 1953–1964
October 12–17, 2022.

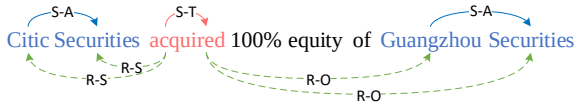

(a) Event: Investment

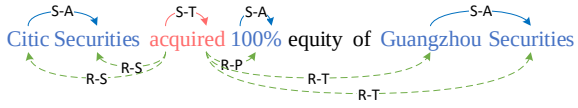

(b) Event: Transfer Share

Figure 2: Two examples to illustrate our tagging scheme. We formalize the overlapping and nested EE as word-word relation recognition, where S-T and S-A denote the relations between the head and tail boundary words of a trigger or argument, and R-S, R-O, R-T, and R-P denote the relations between the trigger word and the argument words with the roles "subject", "object", "target" and "proportion".

`Expression` is the `Theme` argument of another event `Positive Regulation`.

Prior studies for overlapped and nested EE (Yang et al., 2019; Li et al., 2020) employ pipeline-based methods that extract event triggers and arguments in several successive stages. Recently, the state-of-the-art model Sheng et al. (2021) also uses such a method that consecutively performs event type detection, trigger extraction, and argument extraction. The main problem with such a method is that the latter stage relies on the former stage, which inherently brings the error propagation problem.

To address the above issue, we present a novel tagging scheme that transforms overlapping and nested EE into word-word relation recognition. As shown in Figure 2, we design two types of relations, including the span relation (S-$\star$) and role relation (R-$\star$). S-$\star$ handles trigger and argument identification, denoting whether two words are the head-tail boundary of a trigger (T) or argument (A). R-$\star$ addresses argument role classification, indicating whether the argument plays the "$\star$" role in the event.

Based on this scheme, we further propose a one-stage event extraction model, OneEE, which mainly includes three parts. First, it adopts BERT (Devlin et al., 2019) as the encoder to get contextualized word representations. Afterward, an adaptive event fusion layer composed of an attention module and two gate fusion modules are used to obtain event-aware contextual representations for each event type. In the prediction layer, we parallelly predict the span and role relations between each pair of words by calculating distance-aware scores. Finally, event triggers, arguments, and their roles can be decoded out using these relation labels in one stage without error propagation.

We evaluate OneEE on 3 overlapped and nested EE datasets (FewFC (Zhou et al., 2021), Genia11 (Kim et al., 2011), and Genia13 (Kim et al., 2013)), and conduct extensive experiments and analyses. Our contributions can be summarized as follows:

- We design a new tagging scheme that casts event extraction as a word-word relation recognition task, providing a novel and simple solution for overlapped and nested EE.
- We propose OneEE, a one-stage model that effectively extracts word-word relations in parallel for overlapped and nested EE.
- We further present an adaptive event fusion layer to obtain event-aware contextual representations and effectively integrate event information.
- OneEE outperforms the SoTA model with regard to both the performance and inference speed.

2 Related Work

2.1 Event Extraction

Information extraction is one of the key research track in natural language processing (Miwa and Bansal, 2016; Fei et al., 2021c), among which the event extraction is the most complicated task (Chen et al., 2015; Fei et al., 2022c). Traditional EE (i.e., flat or regular EE) (Li et al., 2013; Nguyen et al., 2016; Liu et al., 2018; Sha et al., 2018; Nguyen and Nguyen, 2019) formulates EE into a sequence labeling task, assigning each token with a label (e.g., *BIO* tagging scheme). For example, Nguyen et al. (2016) uses two bidirectional RNNs to get richer representation which is then utilized to predict event triggers and argument roles jointly. Liu et al. (2018) jointly extracts multiple event triggers and arguments by introducing attention-based GCN to model the dependency graph information (Fei et al., 2021b; Li et al., 2021a; Fei et al., 2022b). However, their underlying assumption that event mentions do not overlap with each other is not always valid. Irregular EE (i.e., overlapped and nested EE) has not received much attention, which is more challenging and realistic.

Existing methods for overlapped and nested EE (Yang et al., 2019; Li et al., 2020) perform event extraction in a pipeline manner with several steps. To solve the argument overlap, Yang et al. (2019) adopts multiple sets of binary classifiers where

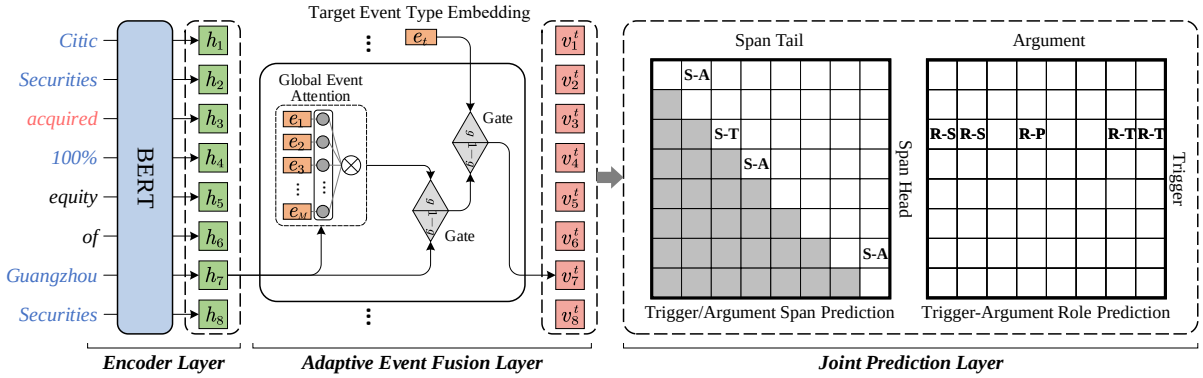

Figure 3: The architecture of our framework. Given a target event type embedding $\boldsymbol{e}_t$ of type t (e.g., transfer share), the goal of our framework is to identify its triggers, arguments, and corresponding roles in the input sentence.

each severs for a role to detect the role-specific argument spans but fails in solving trigger overlap. Except for pipeline methods, the latest attempt dealing with overlapped EE is Sheng et al. (2021) in a joint framework with cascade decoding. They are the first to simultaneously tackle all the overlapping patterns. Sheng et al. (2021) sequentially performs type detection, trigger extraction, and argument extraction, where the overlapped targets are extracted separately conditioned on the specific former prediction. Nevertheless, most of the multi-stage methods suffer from error propagation.

2.2 Tagging-based Information Extraction

Tagging scheme in the field of information extraction has been extensively investigated. Traditional sequence labeling approaches tagging each token once (e.g., *BIO*) is hard to tackle irregular information extraction (e.g., overlapped NER). Several researchers (Zheng et al., 2017) extend the BIO label scheme to adapt to more complex scenarios. However, they suffer from the label ambiguity problem due to limited flexibility. Recently, the grid tagging scheme is used in a lot of information extraction tasks, such as opinion mining (Wu et al., 2020), relation extraction (Wang et al., 2020), and named entity recognition (Wang et al., 2021), due to its characteristic of presenting relations between word pairs. For example, TPLinker (Wang et al., 2020) realizes one-stage joint relation extraction without a gap between training and inference by tagging token pairs with link labels. Inspired by these works, we design our tagging scheme to address overlapping and nested EE, which predicts relations between trigger or argument words parallelly in one stage.

Also it is noteworthy explicitly that this work inherits the recent success of the idea of word-word relation detection, as in Li et al. (2022). Li et al. (2022) propose to unify all the NER (including the flat, nested and discontinuous mentions) with a word-word modeling based on the grid tagging scheme. This work however differs from Li et al. (2022) in two folds. First, we extend the idea of the word-word tagging from NER to EE successfully, where we re-design two relation types for the nested and overlapped events. Second, from the modeling perspective, we devise an adaptive event fusion layer to fully support the one-stage (end-to-end) complex event detection, which greatly helps avoid error propagation.

3 Problem Formulation

The goal of event extraction includes extracting event triggers and their arguments. We can formalize overlapping and nested EE as follows: given an input sentence consisting of N tokens or words $X = \{x_1, x_2, \ldots, x_N\}$ and event type $e \in \mathcal{E}$, the task aims to extract the span relations $\mathcal{S}$ and the role relations $\mathcal{R}$ between each token pair (x_i, x_j), where $\mathcal{E}$ denotes the event type collection, $\mathcal{S}$ and $\mathcal{R}$ are pre-defined tags. These relations can be explained below, and we also give an example as demonstrated in Figure 2 for better understanding.

- $\mathcal{S}$: the span relation indicates that x_i and x_j are the starting and ending token of the extracted trigger span `S-T` or argument span `S-A`, where $1 \leq i \leq j \leq N$.
- $\mathcal{R}$: the role relation indicates that the argument with x_j acts the certain role `R-`$\star$ of the event with the trigger containing x_i, where $1 \leq i, j \leq N$. $\star$ indicates the role type.
- `NONE`, indicating that the word pair does not have any relation defined in this paper.

4 Framework

The architecture of our model is illustrated in Figure 3, which mainly consists of three components. First, the widely-used pre-trained language model, BERT (Devlin et al., 2019), is used as the encoder to yield contextualized word representations from the input sentences. Then, an adaptive event fusion layer consisting of an attention module and two gate modules is used to integrate the target event type embedding into contextual representations. Afterward, a prediction layer is employed to jointly extract the span relations and the role relations between word pairs.

4.1 Encoder Layer

We leverage BERT as the encoder for our model since it has been demonstrated to be one of the SoTA models for representation learning in EE. Given an input sentence $X = \{x_1, x_2, \ldots, x_N\}$, we convert each token x_i into word pieces and then feed them into a pre-trained BERT module. After the BERT calculation, each sentential word may involve vectorial representations of several pieces. Here we employ max pooling to produce word representations $\boldsymbol{H} = \{\boldsymbol{h}_1, \boldsymbol{h}_2, ..., \boldsymbol{h}_N\} \in \mathbb{R}^{N \times d_h}$ based on the word piece representations.

4.2 Adaptive Event Fusion Layer

Since the goal of our framework is to predict the relations between word pairs for the target event type e_t, it is important to generate event-aware representations. Therefore, to fuse the event information and contextual information provided by the encoder, we design an adaptive fusion layer. As shown in Figure 3, it consists of an attention module, modeling the interaction among events and obtaining the global event information, and two gate fusion modules for integrating the global and target event information with contextualized word representations.

Attention Mechanism Motivated by the self-attention in Transformer (Vaswani et al., 2017; Wei et al., 2019a), we first introduce an attention mechanism, of which input consists of queries, keys, and values. The output is computed as a weighted sum of the values, where the weight assigned to each value is the dot product of the query with the corresponding key. The attention mechanism can be formulated as:

$$\text{Attention}(\boldsymbol{Q}, \boldsymbol{K}, \boldsymbol{V}) = \text{softmax}(\frac{\boldsymbol{Q}\boldsymbol{K}^\top}{\sqrt{d_h}})\boldsymbol{V} \ , \quad (1)$$

where $\sqrt{d_h}$ is a scaling factor, $\boldsymbol{Q}$, $\boldsymbol{K}$ and $\boldsymbol{V}$ are query, key and value tensors, represented by Eq. 4.

Gate Fusion Mechanism We design a gate fusion mechanism to integrate two kinds of features and filter the unnecessary information. The gate vector $\boldsymbol{g}$ is produced by a fully-connection layer with the sigmoid function, which can adaptively control the flow of the input:

$$\text{Gate}(\boldsymbol{p}, \boldsymbol{q}) = \boldsymbol{g} \odot \boldsymbol{p} + (1 - \boldsymbol{g}) \odot \boldsymbol{q} \ , \quad (2)$$

$$\boldsymbol{g} = \sigma(\boldsymbol{W}_g[\boldsymbol{p}; \boldsymbol{q}] + \boldsymbol{b}_g) \ , \quad (3)$$

where $\boldsymbol{p}$ and $\boldsymbol{q}$ are input vectors, represented by Eq. 5 and Eq. 6. $\sigma(\cdot)$ is a sigmoid activation function, $\odot$ and $[;]$ denote element-wise product and concatenation operations, respectively. $\boldsymbol{W}_g$ and $\boldsymbol{b}_g$ are trainable parameters.

We leverage the attention mechanism to obtain the global event embeddings for each contextualized word representation. Given a set of randomly initialized event type embeddings $\boldsymbol{E} = \{\boldsymbol{e}_1, \boldsymbol{e}_2, \ldots, \boldsymbol{e}_M\} \in \mathbb{R}^{M \times d_h}$, where M is the number of event types, the calculation can be formulated as:

$$\boldsymbol{E}^g = \text{Attention}(\boldsymbol{W}_q\boldsymbol{H}, \boldsymbol{W}_k\boldsymbol{E}, \boldsymbol{W}_v\boldsymbol{E}) \ , \quad (4)$$

where $\boldsymbol{E}^g$ is the output of the attention mechanism, $\boldsymbol{W}_q$, $\boldsymbol{W}_k$ and $\boldsymbol{W}_v$ are learnable parameters.

To encode global event information into word representations, we adopt a gate module to fuse the contextual word representations and global event representations. After that, we employ another gate mechanism to integrate the target event type embedding and the output of the last gate module. the overall process can be formulated as:

$$\boldsymbol{H}^g = \text{Gate}(\boldsymbol{H}, \boldsymbol{E}^g) \ , \quad (5)$$

$$\boldsymbol{V}^t = \text{Gate}(\boldsymbol{H}^g, \boldsymbol{e}_t) \ , \quad (6)$$

where $\boldsymbol{e}_t \in \boldsymbol{E}$ denotes the target event type embedding, $\boldsymbol{V}^t = \{\boldsymbol{v}_1, \boldsymbol{v}_2, ..., \boldsymbol{v}_N\} \in \mathbb{R}^{N \times d_h}$ is the final event-aware word representations.

4.3 Joint Prediction Layer

After the adaptive event fusion layer, we obtain the event-aware word representations $\boldsymbol{V}^t$, which are used to jointly predict the span and role relations between each pair of words. For each word pair (w_i, w_j), we calculate a score to measure the possibility of them for the relation $s \in \mathcal{S}$ and $r \in \mathcal{R}$.

Distance-aware Score To integrate relative distance information and word pair representations, we introduce a distance-aware score function. For two vectors $\boldsymbol{p}_i$ and $\boldsymbol{p}_j$ from a sequence of representations, we combine them with corresponding position embeddings from Su et al. (2021), and then calculate the score by the dot product of them:

$$\begin{aligned} \text{Score}(\boldsymbol{p}_i, \boldsymbol{p}_j) &= (\boldsymbol{R}_i \boldsymbol{p}_i)^\top (\boldsymbol{R}_j \boldsymbol{p}_j) \\ &= \boldsymbol{p}_i^\top \boldsymbol{R}_{j-i} \boldsymbol{p}_j \ , \end{aligned} \tag{7}$$

where $\boldsymbol{R}_i$ and $\boldsymbol{R}_j$ are position embeddings of $\boldsymbol{p}_i$ and $\boldsymbol{p}_j$, $\boldsymbol{R}_{j-i} = \boldsymbol{R}_i^\top \boldsymbol{R}_j$. Thus, we can obtain the span score c_{ij}^s and the role score c_{ij}^r of the word pair (w_i, w_j) for target event type t:

$$c_{ij}^s = \text{Score}(\boldsymbol{W}_{s1} \boldsymbol{v}_i^t, \boldsymbol{W}_{s2} \boldsymbol{v}_j^t) \ , \tag{8}$$

$$c_{ij}^r = \text{Score}(\boldsymbol{W}_{r1} \boldsymbol{v}_i^t, \boldsymbol{W}_{r2} \boldsymbol{v}_j^t) \ , \tag{9}$$

where $\boldsymbol{W}_{s1}$, $\boldsymbol{W}_{s2}$, $\boldsymbol{W}_{r1}$ and $\boldsymbol{W}_{r2}$ denote parameters. $\boldsymbol{v}_i^t$ and $\boldsymbol{v}_j^t$ are from Eq. 6.

4.4 Training Details

For the score $c_{ij}^\star$, where $\star$ denotes the relation s or r, our training target is to minimize a variant of circle loss (Sun et al., 2020) which extends softmax cross-entropy loss to figure out multi-label classification problem. In addition, we introduce a threshold score δ, noting that the scores of the pairs with relation are larger than δ, and the other pairs are less than it. The loss function can be formulated as:

$$\mathcal{L}^\star = \log(e^\delta + \sum_{(i,j)\in\Omega^\star} e^{-c_{ij}^\star}) + \log(e^\delta + \sum_{(i,j)\notin\Omega^\star} e^{c_{ij}^\star}) \ , \tag{10}$$

where $\Omega^\star$ denotes the pair set of relation $\star$, δ is set to zero.

Finally, we enumerate all event types in the selected event type set $\mathcal{E}'$ and get the total loss:

$$\mathcal{L} = \sum_{t\in\mathcal{E}'} (\sum_{s\in\mathcal{S}} \mathcal{L}^s + \sum_{r\in\mathcal{R}} \mathcal{L}^r) \ , \tag{11}$$

where $\mathcal{S}'$ is a subset sampled from $\mathcal{S}$, we detail the sampling strategy in the appendix.

4.5 Inference

During the inference period, our model is able to extract all events by parallelly injecting their event type embeddings to the adaptive event fusion layer. As shown in Figure 4, once all the tags of a certain event type are predicted by our model in one stage, the overall decoding process can be summarized as four steps: First, we get starting and ending indices of the trigger or argument. Second, we obtain the trigger and argument spans.[2] Third, we match the trigger and arguments according to the R-$\star$ relations. Finally, the event type is assigned to this event structure. Specially, we repeat the above four steps for each event type.

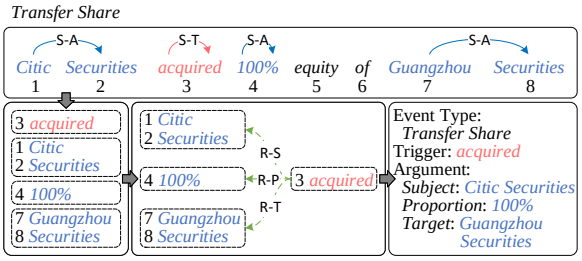

Figure 4: A decoding case of our system with four steps.

		#Ovlp.	#Nest.	#Sent.	#Events
FewFC	train	1,560	-	7,185	10,227
	dev	205	-	899	1,281
	test	210	-	898	1,332
Genia11	train	954	1,628	8,730	6,401
	dev	121	199	1,091	824
	test	125	197	1,092	775
Genia13	train	347	784	4,000	2,743
	dev	44	100	500	352
	test	42	88	500	320

Table 1: Statistics of the datasets. "Ovlp." and "Nest." denote the sentences with overlapped and nested events, respectively.

5 Experiments Settings

5.1 Datasets

As shown in Table 1, we follow previous work (Sheng et al., 2021), adopting FewFC (Zhou et al., 2021), a Chinese financial event extraction benchmark for overlapped EE. FewFC annotates 10 event types and 18 argument role classes with about 22% sentences containing overlapped events. We also experiment on two biomedical datasets for nested EE, namely Genia11 (Kim et al., 2011) and Genia13 (Kim et al., 2013), with around 18% sentences containing nested events. Genia11 annotates 9 event types and 10 argument role classes while the figures for Genia13 are 13 and 7. We split the train/dev/test as 8.0:1.0:1.0 for both of them.

5.2 Implementation Details

We employ the Chinese Bert-base model for FewFC and BioBERT (Lee et al., 2020) for Ge-

[2]Note that if two pairs with the same span relation clash in the boundaries, the pair with higher score will be selected.

		TI(%)			TC(%)			AI(%)			AC(%)		
		P	R	F1	P	R	F1	P	R	F1	P	R	F1
Flat EE	BERT-softmax	89.8	79.0	84.0	80.2	61.8	69.8	74.6	62.8	68.2	72.5	60.2	65.8
	BERT-CRF	**90.8**	80.8	85.5	**81.7**	63.6	71.5	75.1	64.3	69.3	72.9	61.8	66.9
	BERT-CRF-joint	89.5	79.8	84.4	80.7	63.0	70.8	**76.1**	63.5	69.2	**74.2**	61.2	67.1
Ovlp. & Nest. EE	PLMEE	83.7	85.8	84.7	75.6	74.5	75.1	74.3	67.3	70.6	72.5	65.5	68.8
	MQAEE	89.1	85.5	87.4	79.7	76.1	77.8	70.3	68.3	69.3	68.2	66.5	67.3
	CasEE	89.4	87.7	88.6	77.9	78.5	78.2	72.8	73.1	72.9	71.3	71.5	71.4
Ours	OneEE	88.7	**88.7**	**88.7**	79.1	**80.3**	**79.7**	75.4	**77.0**	**76.2**	74.0	**72.9**	**73.4**

Table 2: Results for extracting all kinds of events on FewFC, where TI, TC, AI, AC denote trigger identification, trigger classification, argument identification, and argument classification, respectively. We run our model for 5 times with different random seeds and report the median values.

nia11 and Genia13. We adopt AdamW (Loshchilov and Hutter, 2019) optimizer with the learning rate of $2e-5$ for BERT and $1e-3$ for the other modules. The batch size is 8 and the hidden size d_h is 768. We train our model with 20 epochs on FewFC and Genia11 and 30 epochs on Genia13. All the hyper-parameters are tuned on the development set. All the event type embeddings are trained from scratch.

5.3 Evaluation Metrics

For evaluation, we follow the traditional criteria of previous work (Chen et al., 2015; Du and Cardie, 2020; Sheng et al., 2021). 1) Trigger Identification (TI): A trigger is correctly identified if the predicted trigger span matches with a golden label; 2) Trigger Classification (TC): A trigger is correctly classified if it is correctly identified and assigned to the right type; 3) Argument Identification (AI): An argument is correctly identified if its event type is correctly recognized and the predicted argument span matches with a golden label; 4) Argument Classification (AC): An argument is correctly classified if it is correctly identified and the predicted role matches any of the golden labels. We report Precision (P), Recall (R), and F measure (F1) for each of the four metrics.

5.4 Baselines

Sequence Labeling Methods for Flat EE These methods cast the EE task into a sequence labeling task by assigning each token a label. **BERT-softmax** uses BERT to get feature representations for classifying triggers and arguments. **BERT-CRF** adds the CRF layer on BERT to capture label dependencies. **BERT-CRF-joint** extends the BIO tagging scheme to joint labels of type and role as `B/I/O-type-role`, inspired by joint extraction of entity and relation (Zheng et al., 2017). All these methods are incapable to solve the overlapping problem due to label conflicts.

Multi-stage Methods for Overlapped and Nested EE These methods perform EE in several stages. **PLMEE** (Yang et al., 2019) solves the argument overlap problem by extracting role-specific argument according to the trigger predicted by the trigger extractor in a pipeline manner. **CasEE** (Sheng et al., 2021) sequentially performs type&trigger&argument extractions, where the overlapped targets are separately extracted conditioned on former predictions and all subtasks are jointly learned.

6 Experimental Results

6.1 Results of All EE

Table 2 reports the result of all methods on the overlapped EE dataset, FewFC, while Table 3 reports the results of the nested EE datasets, Genia11 and Genia13. We can observe that:

1) Our method significantly outperforms all other methods and achieves the state-of-the-art F1 score on all three datasets.

2) In comparison with sequence labeling methods, our model achieves better recall and F1 scores. Specifically, our model outperforms BERT-CRF-joint by 11.7% and 6.3% in recall and the F1 score of AC on the FewFC dataset and achieves a substantial improvement of 4.4% in F1 score of AC on two Genia datasets averagely. It shows the effectiveness of our model on overlapped and nested EE since the sequence labeling methods can only solve flat EE.

3) In comparison with multi-stage methods, our model also improves the performance on the F1 score considerably. Our model outperforms the

	TI(%)	TC(%)	AI(%)	AC(%)
• **Genia11**				
BERT-softmax	67.8	64.4	57.4	56.0
BERT-CRF	68.3	64.8	58.3	56.9
BERT-CRF-joint	67.0	64.1	60.2	58.1
PLMEE	67.3	65.5	60.7	59.4
CasEE	70.0	67.0	62.0	60.4
OneEE	**71.5**	**69.5**	**65.9**	**62.5**
• **Genia13**				
BERT-softmax	77.4	75.9	69.9	67.7
BERT-CRF	78.8	77.4	70.1	68.2
BERT-CRF-joint	77.6	75.7	71.9	68.2
PLMEE	79.3	78.3	72.1	70.7
CasEE	80.5	78.5	73.7	71.9
OneEE	**81.9**	**80.8**	**76.8**	**72.7**

Table 3: F1 scores for extracting all events on Genia11 and Genia13.

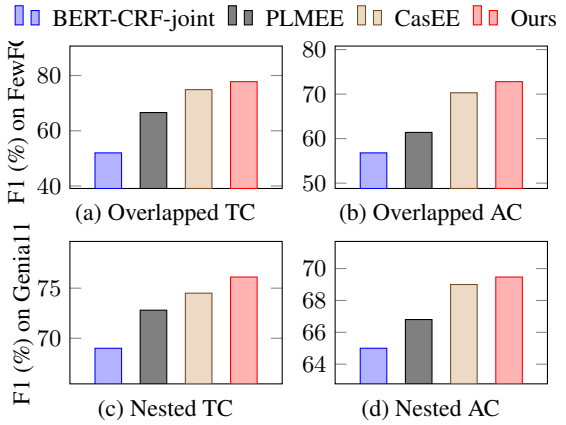

Figure 5: Results for overlapped trigger (a) and argument (b) extraction on FewFC, and nested trigger (c), and argument (d) extraction on Genia11. Note that only the sentences that contain at least one such event are used.

state-of-the-art model, CasEE, by 2.1% in the F1 score of TC on three datasets averagely. We consider this is because that the event feature is well learned by our adaptive event fusion module. Especially, our model improves 3.4% on AI and 1.6% on AC over CasEE on an average of three datasets. The results reveal the superiority of our one-stage framework which elegantly realizes overlapped and nested event extraction without error propagation.

6.2 Results of Overlapped and Nested EE

To evaluate the effectiveness of our proposed model in recognizing overlapping and nested event mentions, we further report the results on sentences containing at least one overlapping event in FewFC and sentences containing at least one nested event in Genia11, respectively.

	TI(%)	TC(%)	AI(%)	AC(%)
OneEE	88.7	79.7	76.2	73.4
w/o Attention	88.3	79.5	75.9	72.8
w/o Gate	88.4	79.3	75.3	72.6
w/o Fusion Layer	88.0	78.7	75.2	72.2
w/o Position Emb.	88.1	78.7	74.1	71.8

Table 4: Ablation studies using FewFC.

Figure 5 illustrates the results of TC and AC on overlapping and nested sentences in testing. It shows that our method outperforms other methods on overlapping and nested sentences. The reasons are mainly two-fold: 1) We solve all the overlapping patterns while BERT-CRF-joint could not handle overlapped and nested EE and PLMEE only solve the argument overlap. 2) Our one-stage model outperforms CasEE because we effectively learn event-aware representations and extract word-word relations in parallel, while CasEE performs in three sequential steps with error propagation.

6.3 Effects of the Modules in the Fusion Layer

To verify the effectiveness of each component, we conduct ablation studies on the FewFC dataset, as shown in Table 4. First, without the attention mechanism, we observe slight performance drops. By replacing the gate mechanism with an addition operation, the performance also decreases to a small degree. Furthermore, a significant drop can be found when the adaptive event fusion layer is substituted by addition, which indicates the usefulness of event representation and context. Finally, removing the position embeddings results in a remarkable drop on all F1 scores, especially 1.6% in the F1 score of AC, which suggests that the information of positions is essential to recognize word-word relations.

6.4 Effect of the Distance-aware Tag Prediction

In this section, we investigate the effect of position embeddings for the prediction layer of OneEE. We divide the arguments in the test set of FewFC into 6 groups according to their distance from corresponding triggers and report the recall scores of the model with and without position embeddings. As shown in Figure 6, the AC recall declines as the distance between trigger and argument in an

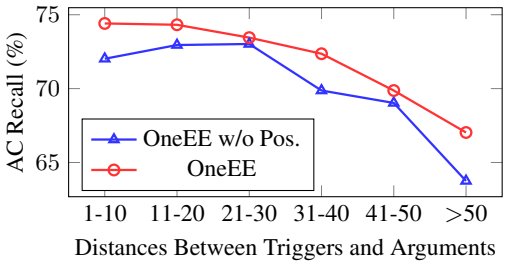

Figure 6: FewFC results of extracting triggers and arguments with different distances. The red line denotes that position embeddings are used while the blue line not.

Model	Stage	#Param.	Speed (sent/s)	Ratio
PLMEE	Two	204.6M	19.8	×1.0
CasEE	Three	120.7M	62.3	×3.1
OneEE	One	114.2M	79.4	×4.0
OneEE†	One	114.2M	186.5	×9.4

Table 5: Parameter number and inference speed comparisons on FewFC. All models are tested with batch size 1, † denotes that the model is tested with batch size 8. The ratio denotes the multiple of the speed increase with regard to PLMEE.

event go up. This indicates that it is more difficult for the model to detect roles correctly if the distance is longer in an event. Furthermore, the model with position embeddings outperforms another one without position embeddings, revealing that the relative distance information is beneficial for event extraction.

6.5 Parameter Number & Efficiency Comparisons

Table 5 lists the stage numbers, parameter numbers, and inference speeds of two baselines and our model. For a fair comparison, all of these models are implemented using PyTorch and tested using the NVIDIA RTX 3090 GPU, where the batch size is set as 1. As seen, PLMEE has 2 times as many parameters as the other two models, due to the utilization of two BERT-based modules for each stage. Moreover, the inference speed of our model is about 3 times faster than that of PLMEE (Yang et al., 2019) and 0.3 times faster than that of CasEE (Sheng et al., 2021), which verifies the efficiency of our model. Last but not least, when the batch size is set as 8, the inference speed of our model is 9.4 times as fast as that of PLMEE, which also demonstrates the advantage of our model, that is, it supports parallel inference. In one word, our model leverages fewer parameters but achieves better performance and faster inference speed.

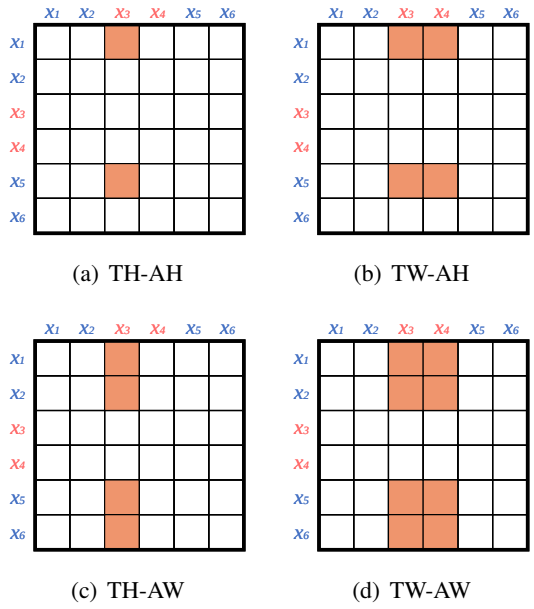

Figure 7: Four kinds of role label strategies. The goal is to predict the relation between trigger head and argument head (a), trigger word and argument head (b), trigger head and argument word (c), and trigger word and argument word (d).

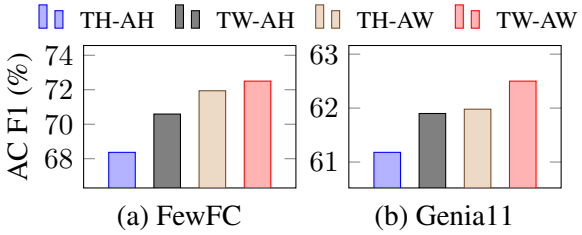

Figure 8: Results of AC with different role label strategies on FewFC (a) and Genia11 (b) datasets.

6.6 Analysis of 4 Role Label Strategies

In this section, we investigate the effect of the role strategies for AC performance. As shown in Figure 7, we introduce 4 different strategies to predict the role relation between trigger and argument: the role labels only exist in 1) trigger and argument head pairs (TH-AH), 2) trigger word and argument head pairs (TW-AH), 3) trigger head and argument word pairs (TH-AW), and 4) trigger and argument word pairs (TW-AW). The results of our model with 4 strategies are demonstrated in Figure 8. We can learn that TW-AW achieves the best results against all other strategies on both FewFC and Genia11 datasets. It is largely due to that its labels are denser than other strategies.

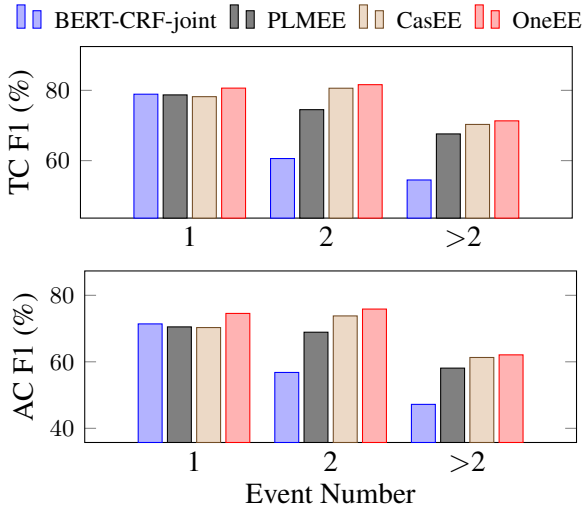

Figure 9: Results of different event numbers on FewFC.

6.7 Analysis of the Event Number

We further investigate the effect of the event number for EE, and the results are shown in Figure 9. We can observe that BERT-CRF-joint, PLMEE, and CasEE achieve similar performances on single-event sentences, while CasEE outperforms PLMEE and BERT-CRF-joint on the sentences with multiple events. Most importantly, our system achieves the best results against all other baselines for different event numbers, indicating the advances of our proposed method.

7 Conclusion

In this paper, we propose a novel one-stage framework based on word-word relation recognition to address overlapped and nested EE concurrently. The relations between word pairs are pre-defined as the word-word relations within a trigger or argument and cross a trigger-argument pair. Moreover, we propose an efficient model that consists of an adaptive event fusion layer for integrating the target event representation, and a distance-aware prediction layer for identifying all kinds of relations jointly. Experimental results show that our proposed model achieves new SoTA results on three datasets and faster speed than the SoTA model. Through ablation studies, we find that the adaptive event fusion layer and distance-aware prediction layer are effective in improving the model performance. In future work, we will extend our method to other structured prediction tasks, such as structured sentiment analysis and overlapped entity relation extraction.

Acknowledgements

This work is supported by the National Natural Science Foundation of China (No. 62176187), the National Key Research and Development Program of China (No. 2017YFC1200500), the Research Foundation of Ministry of Education of China (No. 18JZD015), the Youth Fund for Humanities and Social Science Research of Ministry of Education of China (No. 22YJCZH064), the General Project of Natural Science Foundation of Hubei Province (No.2021CFB385). L Zhao would like to thank the support from Center for Artificial Intelligence (C4AI-USP), the Sao Paulo Research Foundation (FAPESP grant #2019/07665-4), the IBM Corporation, and China Branch of BRICS Institute of Future Networks.

References

Antoine Bosselut, Ronan Le Bras, and Yejin Choi. 2021. Dynamic neuro-symbolic knowledge graph construction for zero-shot commonsense question answering. In *Proceedings of the AAAI Conference on Artificial Intelligence*, pages 4923–4931.

Yubo Chen, Liheng Xu, Kang Liu, Daojian Zeng, and Jun Zhao. 2015. Event extraction via dynamic multi-pooling convolutional neural networks. In *Proceedings of the Annual Meeting of the Association for Computational Linguistics*, pages 167–176.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. Bert: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the North American Chapter of the Association for Computational Linguistics*, pages 4171–4186.

Xinya Du and Claire Cardie. 2020. Event extraction by answering (almost) natural questions. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, pages 671–683.

Hao Fei, Donghong Ji, Bobo Li, Yijiang Liu, Yafeng Ren, and Fei Li. 2021a. Rethinking boundaries: End-to-end recognition of discontinuous mentions with pointer networks. In *Proceedings of the AAAI Conference on Artificial Intelligence*, pages 12785–12793.

Hao Fei, Fei Li, Bobo Li, and Donghong Ji. 2021b. Encoder-decoder based unified semantic role labeling with label-aware syntax. In *Proceedings of the AAAI Conference on Artificial Intelligence*, pages 12794–12802.

Hao Fei, Jingye Li, Shengqiong Wu, Chenliang Li, Donghong Ji, and Fei Li. 2022a. Global inference with explicit syntactic and discourse structures for dialogue-level relation extraction. In *Proceedings*

of the Thirty-First International Joint Conference on Artificial Intelligence, IJCAI, pages 4082–4088.

Hao Fei, Yafeng Ren, and Donghong Ji. 2020a. Boundaries and edges rethinking: An end-to-end neural model for overlapping entity relation extraction. *Information Processing & Management*, 57(6):102311.

Hao Fei, Shengqiong Wu, Yafeng Ren, and Meishan Zhang. 2022b. Matching structure for dual learning. In *Proceedings of the International Conference on Machine Learning, ICML*, pages 6373–6391.

Hao Fei, Shengqiong Wu, Meishan Zhang, Yafeng Ren, and Donghong Ji. 2022c. Conversational semantic role labeling with predicate-oriented latent graph. In *Proceedings of the Thirty-First International Joint Conference on Artificial Intelligence, IJCAI*, pages 4089–4095.

Hao Fei, Meishan Zhang, and Donghong Ji. 2020b. Cross-lingual semantic role labeling with high-quality translated training corpus. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7014–7026.

Hao Fei, Meishan Zhang, Bobo Li, and Donghong Ji. 2021c. End-to-end semantic role labeling with neural transition-based model. In *Proceedings of the AAAI Conference on Artificial Intelligence*, pages 12803–12811.

Arzoo Katiyar and Claire Cardie. 2016. Investigating LSTMs for joint extraction of opinion entities and relations. In *Proceedings of the Annual Meeting of the Association for Computational Linguistics*, pages 919–929.

Jin-Dong Kim, Yue Wang, Toshihisa Takagi, and Akinori Yonezawa. 2011. Overview of genia event task in bionlp shared task 2011. In *Proceedings of BioNLP shared task 2011 workshop*, pages 7–15.

Jin-Dong Kim, Yue Wang, and Yamamoto Yasunori. 2013. The genia event extraction shared task, 2013 edition-overview. In *Proceedings of the BioNLP Shared Task 2013 Workshop*, pages 8–15.

Jinhyuk Lee, Wonjin Yoon, Sungdong Kim, Donghyeon Kim, Sunkyu Kim, Chan Ho So, and Jaewoo Kang. 2020. Biobert: a pre-trained biomedical language representation model for biomedical text mining. *Bioinformatics*, 36(4):1234–1240.

Fayuan Li, Weihua Peng, Yuguang Chen, Quan Wang, Lu Pan, Yajuan Lyu, and Yong Zhu. 2020. Event extraction as multi-turn question answering. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing (Fingdings)*, pages 829–838.

Fei Li, ZhiChao Lin, Meishan Zhang, and Donghong Ji. 2021a. A span-based model for joint overlapped and discontinuous named entity recognition. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing*, pages 4814–4828.

Jingye Li, Hao Fei, Jiang Liu, Shengqiong Wu, Meishan Zhang, Chong Teng, Donghong Ji, and Fei Li. 2022. Unified named entity recognition as word-word relation classification. In *Proceedings of the AAAI Conference on Artificial Intelligence*, pages 10965–10973.

Jingye Li, Kang Xu, Fei Li, Hao Fei, Yafeng Ren, and Donghong Ji. 2021b. MRN: A locally and globally mention-based reasoning network for document-level relation extraction. In *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*, pages 1359–1370.

Qi Li, Heng Ji, and Liang Huang. 2013. Joint event extraction via structured prediction with global features. In *Proceedings of the Annual Meeting of the Association for Computational Linguistics*, pages 73–82.

Xiao Liu, Zhunchen Luo, and He-Yan Huang. 2018. Jointly multiple events extraction via attention-based graph information aggregation. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, pages 1247–1256.

Ilya Loshchilov and Frank Hutter. 2019. Decoupled weight decay regularization. In *Proceedings of the International Conference on Learning Representations*.

Makoto Miwa and Mohit Bansal. 2016. End-to-end relation extraction using LSTMs on sequences and tree structures. In *Proceedings of the Annual Meeting of the Association for Computational Linguistics*, pages 1105–1116.

Makoto Miwa, Tomoko Ohta, Rafal Rak, Andrew Rowley, Douglas B Kell, Sampo Pyysalo, and Sophia Ananiadou. 2013. A method for integrating and ranking the evidence for biochemical pathways by mining reactions from text. *Bioinformatics*, 29(13):44–52.

Thien Huu Nguyen, Kyunghyun Cho, and Ralph Grishman. 2016. Joint event extraction via recurrent neural networks. In *Proceedings of the North American Chapter of the Association for Computational Linguistics*, pages 300–309.

Trung Minh Nguyen and Thien Huu Nguyen. 2019. One for all: Neural joint modeling of entities and events. In *Proceedings of the AAAI Conference on Artificial Intelligence*, pages 6851–6858.

Wijnand Nuij, Viorel Milea, Frederik Hogenboom, Flavius Frasincar, and Uzay Kaymak. 2013. An automated framework for incorporating news into stock trading strategies. *IEEE TKDE*, 26(4):823–835.

Lei Sha, Feng Qian, Baobao Chang, and Zhifang Sui. 2018. Jointly extracting event triggers and arguments by dependency-bridge rnn and tensor-based argument interaction. In *Proceedings of the AAAI Conference on Artificial Intelligence*, pages 5916–5923.

Jiawei Sheng, Shu Guo, Bowen Yu, Qian Li, Yiming Hei, Lihong Wang, Tingwen Liu, and Hongbo Xu. 2021. Casee: A joint learning framework with cascade decoding for overlapping event extraction. In *Proceedings of the Annual Meeting of the Association for Computational Linguistics (Findings)*, pages 164–174.

Jianlin Su, Yu Lu, Shengfeng Pan, Bo Wen, and Yunfeng Liu. 2021. Roformer: Enhanced transformer with rotary position embedding. *CoRR*, abs/2104.09864.

Yifan Sun, Changmao Cheng, Yuhan Zhang, Chi Zhang, Liang Zheng, Zhongdao Wang, and Yichen Wei. 2020. Circle loss: A unified perspective of pair similarity optimization. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 6397–6406.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Proc. of NeurIPS*, pages 5998–6008.

Yucheng Wang, Bowen Yu, Yueyang Zhang, Tingwen Liu, Hongsong Zhu, and Limin Sun. 2020. Tplinker: Single-stage joint extraction of entities and relations through token pair linking. In *Proceedings of the International Conference on Computational Linguistics*, pages 1572–1582.

Yucheng Wang, Bowen Yu, Hongsong Zhu, Tingwen Liu, Nan Yu, and Limin Sun. 2021. Discontinuous named entity recognition as maximal clique discovery. In *Proceedings of the Annual Meeting of the Association for Computational Linguistics*, pages 764–774.

Yinwei Wei, Xiang Wang, Weili Guan, Liqiang Nie, Zhouchen Lin, and Baoquan Chen. 2019a. Neural multimodal cooperative learning toward micro-video understanding. *IEEE Transactions on Image Processing*, 29:1–14.

Yinwei Wei, Xiang Wang, Liqiang Nie, Xiangnan He, Richang Hong, and Tat-Seng Chua. 2019b. Mmgcn: Multi-modal graph convolution network for personalized recommendation of micro-video. In *Proceedings of the 27th ACM International Conference on Multimedia*, pages 1437–1445.

Zhen Wu, Chengcan Ying, Fei Zhao, Zhifang Fan, Xinyu Dai, and Rui Xia. 2020. Grid tagging scheme for end-to-end fine-grained opinion extraction. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing (Findings)*, pages 2576–2585.

Sen Yang, Dawei Feng, Linbo Qiao, Zhigang Kan, and Dongsheng Li. 2019. Exploring pre-trained language models for event extraction and generation. In *Proceedings of the Annual Meeting of the Association for Computational Linguistics*, pages 5284–5294.

Suncong Zheng, Feng Wang, Hongyun Bao, Yuexing Hao, Peng Zhou, and Bo Xu. 2017. Joint extraction of entities and relations based on a novel tagging scheme. In *Proceedings of the Annual Meeting of the Association for Computational Linguistics*, pages 1227–1236.

Yang Zhou, Yubo Chen, Jun Zhao, Yin Wu, Jiexin Xu, and Jinlong Li. 2021. What the role is vs. what plays the role: Semi-supervised event argument extraction via dual question answering. In *Proceedings of the AAAI Conference on Artificial Intelligence*, pages 14638–14646.

A Parallel Training with Sampling

We parallelly inject multiple target event type embeddings at the adaptive event fusion layer during training period, which results in huge computation resources. To this end, we use a subset $\mathcal{E}'$ to replace $\mathcal{E}$ for each sample, where the number of $\mathcal{E}'$ is K. It consists of one positive event type (the event type annotated in the sample) and $K-1$ negative event types selected randomly from the event types that does not appear in the sample. In other words, we inject K different event type embeddings into the gate module of Eq. 6 simultaneously. If there is no positive event type in the sample, we will select K negative event types.

B Decoding for Nested EE

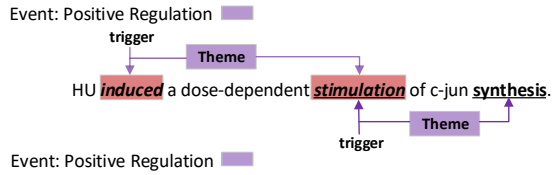

(a) Nested Event Example

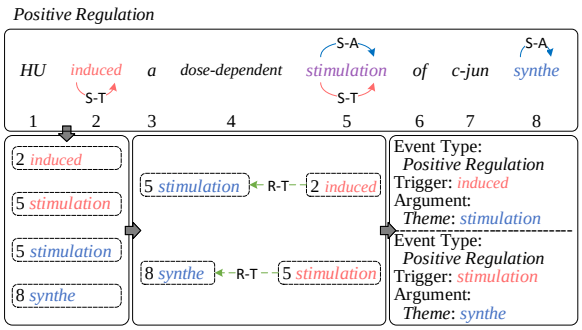

(b) Nested EE Decoding

Figure 10: Example of nested event (a) and its decoding process (b).

In the manuscript, we have already shown the decoding process of our model for overlapped EE in Section 4.5. Due to page limitation, we show an example of nested in Figure 10(a). We also demonstrate its decoding process in Figure 10(b), which is the same as the overlapped EE decoding.

C Analysis of the Event Sampling Number

To further analyze the effect of sampling number K and the sampling strategy, we also evaluate our model with positive and negative sampling and random sampling and compare them with different sampling numbers. Figure 11 shows the TC F1 change trend as the number of sampling increases. As seen, both two models with 6 event type samplings achieve the best performance, compared with the other sampling numbers. Specifically, our model with one positive sampling and $K-1$ negative samplings outperforms the model with K randomly selected samplings when K is less than 7, which demonstrates that our sampling strategy is helpful for the model training.

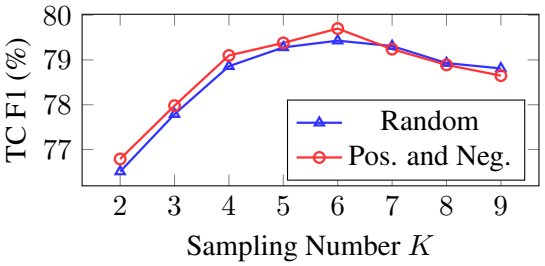

Figure 11: Results on different sampling numbers of random sampling, and positive and negative sampling.

Joint Language Semantic and Structure Embedding for Knowledge Graph Completion

Jianhao Shen[1], Chenguang Wang[2]*, Linyuan Gong[3], Dawn Song[3]
[1]Peking University, [2]Washington University in St. Louis, [3]UC Berkeley
jhshen@pku.edu.cn, chenguangwang@wustl.edu,
{gly,dawnsong}@berkeley.edu

Abstract

The task of completing knowledge triplets has broad downstream applications. Both structural and semantic information plays an important role in knowledge graph completion. Unlike previous approaches that rely on either the structures or semantics of the knowledge graphs, we propose to jointly embed the semantics in the natural language description of the knowledge triplets with their structure information. Our method embeds knowledge graphs for the completion task via fine-tuning pre-trained language models with respect to a probabilistic structured loss, where the forward pass of the language models captures semantics and the loss reconstructs structures. Our extensive experiments on a variety of knowledge graph benchmarks have demonstrated the state-of-the-art performance of our method. We also show that our method can significantly improve the performance in a low-resource regime, thanks to the better use of semantics. The code and datasets are available at https://github.com/pkusjh/LASS.

1 Introduction

Knowledge graphs (KG), such as Wikidata and Freebase (Bollacker et al., 2008), consist of factual triplets. KGs have been useful resources for both humans and machines. A triplet in the form of *(head entity, relation, tail entity)*, where the relation involves both head and tail entities, has been used in a great variety of applications, such as question answering (Guu et al.; Hao et al., 2017) and web search (Xiong et al., 2017). Incompleteness has been a longstanding issue in KGs (Carlson et al., 2010), impeding their wider adoption in real-world applications.

KG completion aims to predict a missing entity or relation of a factual triplet. Structural patterns in the existing triplets are useful to predict the missing elements (Bordes et al., 2013; Sun et al., 2019). For example, a composition pattern can be learned to predict the relation *grandmother_Of* based on two consecutive *mother_Of* relations. Besides the structure information, semantic relatedness between entities and relations is also critical to infer entities or relations with similar meanings (An et al., 2018; Yao et al., 2019; Wang et al., 2021). For example, if a relationship *CEO_Of* holds between two entities, the relation *employee_Of* also holds. There are two kinds of KG completion approaches that fall into different learning paradigms. First, the structure-based approaches treat entities and relations as nodes and edges, and use graph embedding methods to learn their representations. Second, the semantic-based approaches encode the text description of entities and relations via language models. While both structures and semantics are vital to KG completion, it is non-trivial for existing methods to process both structural and semantic information.

In this paper, we propose LASS, a joint language semantic and structure embedding for knowledge graph completion, which incorporates both semantics and structures in a KG triplet. LASS embeds a triplet into a vector space by fine-tuning pre-trained language models (LM) with respect to a structured loss. LASS involves both semantic embedding and structure embedding. The semantic embedding captures the semantics of the triplet, which corresponds to the forward pass of a pre-trained LM over the natural language description of the triplet. The structure embedding aims to reconstruct the structures in the semantic embedding, which corresponds to optimizing a probabilistic structured loss via the backpropagation of the LM. Intuitively, the structured loss treats the relationship between two entities as a translation between embeddings of the entities. LASS outperforms the existing approaches on a collection of KG completion benchmarks. We further evaluate LASS in low-resource settings and find that it is more data-efficient than other methods. The reason is that our method ex-

*Corresponding author

Proceedings of the 29th International Conference on Computational Linguistics, pages 1965–1978
October 12–17, 2022.

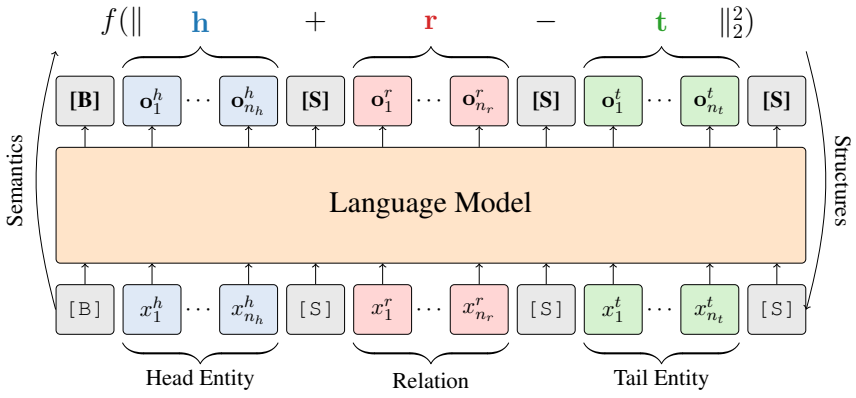

Figure 1: Overview of LASS. LASS maps a knowledge triplet *(Head Entity, Relation, Tail Entity)*, in short *(h, r, t)*, to the corresponding embedding vectors, $\mathbf{h}, \mathbf{r}, \mathbf{t} \in \mathbb{R}^k$. LASS embeds KGs for KG completion via fine-tuning pre-trained language models (LM) w.r.t. a probabilistic structured loss, where the forward pass of the LMs captures semantics and the loss reconstructs structures. In particular, LASS consists of semantic embedding and structure embedding. The *semantic embedding* (leftmost arrow) is generated by a forward pass of the LMs followed by a pooling layer over the natural language description of a triplet. `[B]` (the beginning token) and `[S]` (the separator token) are special tokens of LMs attached to the description. For example, the textual description of head entity is $(x_1^h, \cdots, x_{n_h}^h)$. $\mathbf{h}$ is calculated as the mean pooling of the corresponding LM outputs $(\mathbf{o}_1^h, \cdots, \mathbf{o}_{n_h}^h)$. $\mathbf{r}$ and $\mathbf{t}$ are calculated similarly. The *structure embedding* (rightmost arrow) reconstructs KG structures in the semantic embeddings via optimizing a structured loss on top of the LMs through backpropagation. The structured loss is based on a score function $f(\|\mathbf{h} + \mathbf{r} - \mathbf{t}\|_2^2)$, which regards the relationship between two entities corresponds to a translation between the embeddings of the entities. The goal is to minimize the loss function so that $\mathbf{h} + \mathbf{r} \approx \mathbf{t}$ when *(h, r, t)* holds.

ploits both semantics and structures in the training data.

The contributions are the following:

- We design a natural language embedding approach, LASS, that integrates both structural and semantic information of KGs, for KG completion. We train LASS by fine-tuning pre-trained LMs w.r.t. a structured loss, where the forward pass of the LMs captures semantics and the loss reconstructs structures. The method consists of both the KG module and the LM module, which sheds light on the connections between the KGs and deep language representation, and advances the research at the intersection of the two areas.

- We evaluate LASS on two KG completion tasks, link prediction and triplet classification, and obtain state-of-the-art performance. The results suggest that capturing both semantics and structures is critical to understand the KGs. The findings are beneficial to many downstream knowledge-driven applications.

- We show that we can significantly improve the performance in the low-resource settings over existing approaches, thanks to the improved transfer of knowledge about semantics.

2 LASS

We introduce LASS to embed both semantics and structures of knowledge graphs (KG) with natural language. As shown in Figure 1, LASS incorporates two embeddings: semantic embedding and structure embedding. The semantic embedding captures the semantics in the natural language description of the KG triplets. The structure embedding further reconstructs the structure information of the KGs from the semantic embedding. LASS embeds KG in a vector space by fine-tuning a pre-trained language model (LM) w.r.t. a structured loss, where the forward pass performs semantic embedding and the optimization of structured loss conducts structure embedding.

2.1 Semantic Embedding

A KG of triplets is denoted as G. Each triplet of G is in the form of *(h, r, t)*, where *h*,*t* $\in E$ and $r \in R$. E is the set of entities, and R is the set of relations. The semantic similarities between the head entity *h*, relation *r*, and tail entity *t* are crucial to complete a factual triplet. For example, given *h* = "Bob Dylan" and *r* = "was born in", the task is to predict a missing *t*, where the candidates are "Duluth" and "Apple". The semantic similarity between "Bob Dylan" and "Duluth", as well as the similarity between "was born in" and "Duluth" should be larger than

their similarities with "Apple" as "Duluth" is the ground-truth answer. Pre-trained LMs capture the rich semantics in natural language via pre-training on large-scale textual corpora. This inspires us to use the semantics stored in the parameters of LMs to encode the semantics of triplets.

Formally, for a triplet *(h, r, t)*, both entities (*h* and *t*) and relation (*r*) are represented by their corresponding natural language descriptions. The head entity *h* is represented as a sequence of tokens, $T^h = (x_1^h, \cdots, x_{n_h}^h)$, describing the entity. Similarly, $T^t = (x_1^t, \cdots, x_{n_t}^t)$ represents the tail entity *t*. $T^r = (x_1^r, \cdots, x_{n_r}^r)$ denotes the relation *r*. We generate the semantic embedding via the forward pass of the LMs as shown in Figure 1. The knowledge graph completion tasks require explicit modeling of dependency of the head, relation and tail. For example, both the connections between head and tail, and relation and tail contribute to the prediction of the tail in the link prediction task. Therefore, we use the concatenation of T^h, T^r, and T^t as the input sequence to the LMs, and use the mean pooling over the output representation of every token in T^h, T^r, and T^t from the forward pass of LMs as $\mathbf{h}, \mathbf{r}, \mathbf{t} \in \mathbb{R}^k$, where k is the dimension of the embedding vectors.

More specifically, we construct the input sequence in the following format: `[B]` T^h `[S]` T^r `[S]` T^t `[S]`, where `[B]` is a special symbol added in front of every input sequence, and `[S]` is a special separator token. The special tokens are different for various LMs. For example, `[B]` and `[S]` are implemented as `[CLS]` and `[SEP]` for BERT (Devlin et al., 2019) respectively. The input sequence is then converted to the corresponding input embeddings of the LMs. For example, the input embeddings of BERT are the sum of the token embeddings, the segment embeddings, and the position embeddings. The input embeddings are fed into the LM. We add a mean pooling layer on top of the output layer of the LM and perform mean pooling over the output representation of every token in T^h, i.e., $(\mathbf{o}_1^h, \cdots, \mathbf{o}_{n_h}^h)$, resulting in $\mathbf{h}$ as illustrated in Figure 1. We obtain $\mathbf{r}$ and $\mathbf{t}$ in the same way. The dimension k equals to the hidden size of the LM.

2.2 Structure Embedding

Structural information of KGs has been successfully used in the KG completion. Traditional approaches regard the relationship between two entities corresponds to a translation between the embeddings of the entities. This is different from the above semantic embedding and the forward pass cannot capture the structure information. We propose to incorporate the structure embedding by fine-tuning the pre-trained LM with a structure loss.

The goal is to reconstruct structure information in the semantic embedding. The updated embeddings of *h*, *r*, and *t* are still denoted as $\mathbf{h}$, $\mathbf{r}$, and $\mathbf{t}$, which incorporate structure information of KGs while preserve semantic information. We reconstruct structure information in the semantic embeddings via optimizing a probabilistic structured loss, in which the score function of a triplet *(h, r, t)* is defined by Eq. 1:

$$f(\mathbf{h}, \mathbf{r}, \mathbf{t}) = b - \frac{1}{2}\|\mathbf{h} + \mathbf{r} - \mathbf{t}\|_2^2 \tag{1}$$

If *(h, r, t)* holds, we have $\mathbf{h} + \mathbf{r} \approx \mathbf{t}$. We also use $f(\|\mathbf{h} + \mathbf{r} - \mathbf{t}\|_2^2)$ to denote this in Figure 1 for simplicity. The score function is motivated by TransE (Bordes et al., 2013).

We define the following probabilistic model based on the score function (1):

$$\Pr(h|r,t) = \frac{\exp(f(\mathbf{h}, \mathbf{r}, \mathbf{t}))}{\sum_{\tilde{h} \in E} \exp(f(\tilde{\mathbf{h}}, \mathbf{r}, \mathbf{t}))} \tag{2}$$

Here $\tilde{h}$ is the corrupted head sampled from the entity set E. $\Pr(r|h,t)$ and $\Pr(t|h,r)$ have a similar form except that the summation in the denominator is over corrupted relations and tails, respectively.

The probabilistic structured loss is defined in Eq. 3. The goal is to minimize the negative log likelihood over the KG:

$$\begin{aligned} L = -\sum_{(h,r,t) \in G} (& \log \Pr(h|r,t) + \log \Pr(r|h,t) \\ & + \log \Pr(t|h,r)) \end{aligned} \tag{3}$$

Optimization Computing the probability in Eq. 2 is computationally inefficient since it requires a forward pass of all possible triplets $(\tilde{h}, r, t)$ to compute the denominator. We use negative sampling (Mikolov et al., 2013) to make training more efficient. Instead of minimizing $-\log \Pr(h|r,t)$ as in Eq. 3, we optimize the loss as is described in Eq. 4 for modeling *h*.

$$\begin{aligned} L_h = & -\log \Pr(1|h,r,t) \\ & -\sum_i^{n_{\text{ns}}} \mathbb{E}_{\tilde{h}_i \sim E \setminus \{h\}} \log \Pr(0|\tilde{h}_i, r, t) \end{aligned} \tag{4}$$

where $\Pr(1|h,r,t) = \sigma(f(\mathbf{h},\mathbf{r},\mathbf{t}))$.

The loss for modeling *r* and *t* is similarly defined. Here, hyperparameter n_{ns} is the number of negative samples. Each negatively sampled head $\tilde{h}_i$ is drawn uniformly without replacement from the entity set $E\backslash\{h\}$. A sample is not treated as a negative sample if it is already a positive example. We have the final structured loss $L = \sum_{(h,r,t)\in G}(L_h + L_r + L_t)$ by adopting the similar negative sampling procedures for relations and tail entities.

The training of LASS is unified as fine-tuning an LM with respect to a structured loss. The semantic embedding is obtained by the forward pass of the LM. The structure embedding is conducted by optimizing the structured loss through backpropagation of the LM.

3 Experiments

3.1 Experimental Setup

Datasets We test the performance of our method on five KG benchmarks built with three KGs: Freebase (Bollacker et al., 2008), WordNet (Miller, 1994) and UMLS (Dettmers et al., 2018). Freebase is a large-scale KG containing general knowledge facts. We employ two subsets from Freebase, namely FB15K-237 (Toutanova and Chen, 2015), and FB13 (Socher et al., 2013). WordNet provides semantic knowledge of words. We use two subsets from WordNet, namely WN18RR (Dettmers et al., 2018), and WN11 (Socher et al., 2013). UMLS is a medical semantic network containing semantic entities and relations. The statistics are summarized in Table 1. We also provide a detailed description of the datasets in Appendix A.1.

Dataset	# Entity	# Relation	# Train	# Dev	# Test
FB15k-237	14,541	237	272,115	17,535	20,466
WN18RR	40,943	11	86,835	3,034	3,134
UMLS	135	46	5,216	652	661
FB13	75,043	13	316,232	5,908	23,733
WN11	38,696	11	112,581	2,609	10,544

Table 1: Statistics of knowledge graphs.

Implementation Details We use two families of LMs with LASS. First, we adopt both $\text{BERT}_{\text{BASE}}$ and $\text{BERT}_{\text{LARGE}}$ from (Devlin et al., 2019) with LASS, namely $\text{LASS-BERT}_{\text{BASE}}$ and $\text{LASS-BERT}_{\text{LARGE}}$. Second, RoBERTa family (Liu et al., 2019) is used, namely $\text{LASS-RoBERTa}_{\text{BASE}}$ and $\text{LASS-RoBERTa}_{\text{LARGE}}$.

We train LASS with AdamW (Loshchilov and Hutter, 2019) on each KG dataset via fine-tuning the corresponding LMs. The training hyperparameters are set as follows. For $\text{LASS-BERT}_{\text{BASE}}$ and $\text{LASS-RoBERTa}_{\text{BASE}}$, the batch size is set to 128, the learning rate is set to 3e-5 with linear warm-up and 0.01 weight decay. We set the batch size to 64 for $\text{LASS-BERT}_{\text{LARGE}}$ and $\text{LASS-RoBERTa}_{\text{LARGE}}$. The number of training epochs is set to 5. The margin b in Eq. 1 is empirically set to 7. We sample 5 negative entities or relations resulting in 15 negative triplets for each positive triplet for the negative sampling.

We represent entities and relations as their names or descriptions (Yao et al., 2019). For FB15k-237, we used entity descriptions from (Xie et al., 2016). For FB13, we use entity descriptions in Wikipedia. For WN18RR, we use definitions of synsets as entity descriptions. For WN11 and UMLS, the entity names are used as the entity descriptions. The relation descriptions are based on the relation names across all the datasets. The input sequence is constructed based on Sec. 2.1. For $\text{LASS-BERT}_{\text{BASE}}$ and $\text{LASS-BERT}_{\text{LARGE}}$, we use a character-level BPE vocabulary. `[B]` is replaced with `[CLS]`, and `[S]` is replaced with `[SEP]`. For $\text{LASS-RoBERTa}_{\text{BASE}}$ and $\text{LASS-RoBERTa}_{\text{LARGE}}$, we use a byte-level BPE vocabulary, and `[B]` and `[S]` are replaced with `BOS` and `EOS` respectively. We implement LASS using the Transformers package (Wolf et al., 2020).

Comparison Methods We compare our method to state-of-the-art methods, including (i) shallow structure embedding: TransE (Bordes et al., 2013), TransH (Wang et al., 2014b), TransR (Lin et al., 2015), TransD (Ji et al., 2015), TransG (Xiao et al., 2016), TranSparse (Ji et al., 2016), DistMult (Yang et al., 2015), DistMult-HRS (Zhang et al., 2018), ConvE (Dettmers et al., 2018), ConvKB (Nguyen et al., 2018), ComplEx (Trouillon et al., 2016), RotatE (Sun et al., 2019), REFE (Chami et al., 2020), HAKE (Zhang et al., 2019a), and ComplEx-DURA (Zhang et al., 2020); (ii) deep structure embedding: NTN (Socher et al., 2013), DOLORES (Wang et al., 2018), KBGAT (Nathani et al., 2019), GAATs (Wang et al., 2020), NePTuNe (Sonkar et al., 2021), and ComplEx-N3-RP (Chen et al., 2021); (iii) language semantic embedding: TEKE (Wang and Li, 2016), KG-BERT (Yao et al., 2019), and stAR (Wang et al., 2021). We present a detailed technical description of the above methods in Appendix A.2.

Method	WN11	FB13	Avg
NTN (Socher et al., 2013)	86.2	90.0	88.1
TransE (Bordes et al., 2013)	75.9	81.5	78.7
TransH (Wang et al., 2014b)	78.8	83.3	81.1
TransR (Lin et al., 2015)	85.9	82.5	84.2
TransD (Ji et al., 2015)	86.4	89.1	87.8
TEKE (Wang and Li, 2016)	86.1	84.2	85.2
TransG (Xiao et al., 2016)	87.4	87.3	87.4
TranSparse-S (Ji et al., 2016)	86.4	88.2	87.3
DistMult (Yang et al., 2015)	87.1	86.2	86.7
DistMult-HRS (Zhang et al., 2018)	88.9	89.0	89.0
AATE (An et al., 2018)	88.0	87.2	87.6
ConvKB (Nguyen et al., 2018)	87.6	88.8	88.2
DOLORES (Wang et al., 2018)	87.5	89.3	88.4
KG-BERT (Yao et al., 2019)	93.5	90.4	91.9
LASS-BERT$_{\text{BASE}}$ (ours)	93.3	91.2	92.3
LASS-BERT$_{\text{LARGE}}$ (ours)	**94.5**	**91.8**	**93.2**
LASS-RoBERTa$_{\text{BASE}}$ (ours)	92.3	91.1	91.7
LASS-RoBERTa$_{\text{LARGE}}$ (ours)	93.8	91.6	92.7

Table 2: Triplet classification accuracy on WN11 and FB13.

3.2 Triplet Classification

The task of triplet classification judges whether a given triplet *(h, r, t)* is correct or not. The task is a binary classification task. We use WN11 and FB13 for the task, since only the test sets of the two datasets contain both positive and negative triplets among all the datasets. For the task, we use the score function as defined in Eq. 1 , and set a score threshold. For a triplet, if the score is above the threshold, the triplet is classified as positive, otherwise negative. We set the threshold empirically based on the accuracy on the validation set. As shown in Table 2, we conclude with the following findings.

Head	Relation	Tail	Label
ron ziegler	gender	male	✓
john fortescue	profession	writer	✓
george j adams	cause of death	typhoid fever	✓
fleiss joseph	institution	columbia university	✓
edmund husserl	nationality	austria	✓
aleksandr bakulev	gender	female	✗
emile littre	profession	physicist	✗
joseph smith jr	cause of death	emphysema	✗
frank g slaughter	institution	university of toronto	✗
julius klinger	nationality	romania	✗

Table 3: Samples of LASS's correct predictions on FB13, where KG-BERT (Yao et al., 2019) outputs wrong predictions. Label ✓ means a gold positive triplet. ✗ indicates a gold negative triplet.

We find that our methods consistently produce state-of-the-art results on triplet classification tasks. This indicates that our score function has captured semantics and structures that are crucial for the triplet classification. We also notice that LASS-BERT generates slightly better results compared

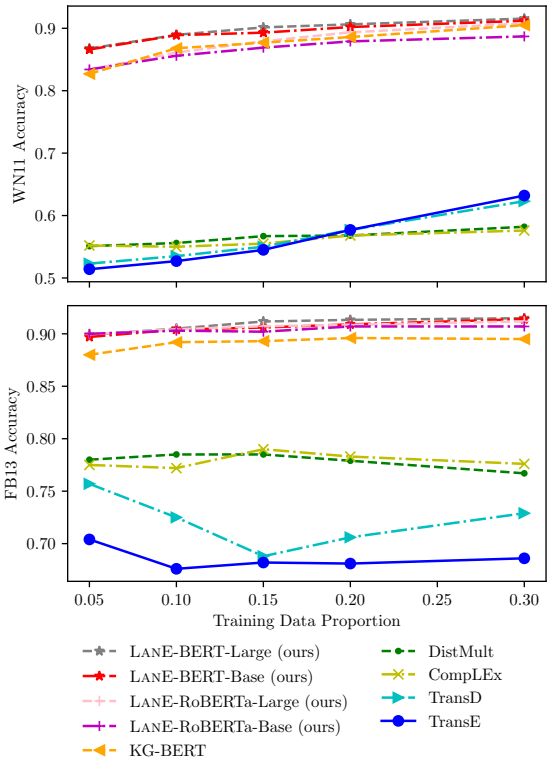

Figure 2: Triplet classification accuracy in a low-resource regime: training with different proportions of the corresponding training datasets on WN11 and FB13.

to LASS-RoBERTa. This is due to RoBERTa removing the NSP objective, however the objective naturally fits in the triplet classification task. LASS-RoBERTa still generates reasonable results. The reason is that the masked LM objective captures the necessary semantics needed for the triplet classification, and LASS is able to preserve the important semantic information.

In Table 3, we also show some cases where LASS-BERT$_{\text{BASE}}$ makes correct predictions while KG-BERT produces incorrect ones on FB13. Compared to KG-BERT, we find that LASS is more capable in relations that require comprehensive structure information, such as "institution".

3.3 Low-Resource Settings

We additionally test the accuracy of triplet classification in a low-data regime, in particular, when using 5%, 10%, 15%, 20%, and 30% of the training data on WN11 and FB13. The results are shown in Figure 2. LASS-BERT$_{\text{LARGE}}$ consistently outperforms the state-of-the-art KG-BERT. This indicates that LASS is more data-efficient, as it leverages both semantics and structures in the training data. We also find that LASS is able to produce competitive results with less training data

Method	FB15k-237		WN18RR		UMLS	
	Hits@10	MR	Hits@10	MR	Hits@10	MR
TransE (Bordes et al., 2013)	0.465	357	0.501	3384	0.989	1.84
DistMult (Yang et al., 2015)	0.419	254	0.49	5110	0.846	5.52
ComplEx (Trouillon et al., 2016)	0.428	339	0.51	5261	0.967	2.59
ConvE (Dettmers et al., 2018)	0.501	244	0.52	4187	0.990	1.51
RotatE (Sun et al., 2019)	0.533	177	0.571	3340	-	-
HAKE (Zhang et al., 2019a)	0.542	-	0.582	-	-	-
KBGAT (Nathani et al., 2019)	0.626	210	0.581	1940	-	-
KG-BERT (Yao et al., 2019)	0.420	153	0.524	97	0.990	1.47
REFE (Chami et al., 2020)	0.541	-	0.561	-	-	-
GAATs (Wang et al., 2020)	**0.650**	187	0.604	1270	-	-
ComplEx-DURA (Zhang et al., 2020)	0.560	-	0.571	-	-	-
StAR (Wang et al., 2021)	0.562	117	0.732	46	0.991	1.49
NePTuNe (Sonkar et al., 2021)	0.547	-	0.557	-	-	-
ComplEx-N3-RP (Chen et al., 2021)	0.568	-	0.580	-	**0.998**	-
LASS-BERT$_{\text{BASE}}$ (ours)	0.479	131	0.725	55	0.991	**1.39**
LASS-BERT$_{\text{LARGE}}$ (ours)	0.527	120	0.769	41	0.990	1.58
LASS-RoBERTa$_{\text{BASE}}$ (ours)	0.500	116	0.737	53	0.994	1.41
LASS-RoBERTa$_{\text{LARGE}}$ (ours)	0.533	**108**	**0.786**	**35**	0.989	1.56

Table 4: Link prediction results on FB15k-237, WN18RR and UMLS.

compared to existing methods even with full training data. LASS-BERT$_{\text{LARGE}}$ with 5% training data of WN11 outperforms most of the existing methods using full training data. When using 10% training data of FB13, LASS-BERT$_{\text{LARGE}}$ is able to perform comparably with KG-BERT with full training data, and outperforms the remaining methods. This is because LASS transfers the knowledge about semantics better to the tasks compared to existing approaches without fully leveraging the KG semantics. The results suggest that LASS is effective in low-resource scenarios.

3.4 Link Prediction

Link prediction aims to predict a missing entity given a relation and the other entity, which is evaluated as a ranking problem. We perform link prediction on FB15k-237, WN18RR and UMLS datasets. For each correct triplet *(h, r, t)*, either *h* or *t* is corrupted by replacing it with every other entity in the entity set E. These triplets are ranked based on scores produced by Eq. 1 of LASS. The evaluation is under the filtered setting (Bordes et al., 2013), i.e., removing all the triplets that appear either in the train, dev, or test set. Two common metrics, Mean Rank (MR) and Hits@10 (the proportion of correct entities ranked in the top 10) are used to evaluate the results. A lower MR is better while a higher Hits@10 is better. From the results in Table 4, we summarize key observations as below.

We find all our methods significantly outperform the compared methods in MR, and reach competitive or better Hits@10. LASS-RoBERTa$_{\text{LARGE}}$ performs the best on WN18RR, which outperforms the best compared method StAR by 11 units in MR and 5.4% in Hits@10. It also delivers the best MR on FB15k-237. On UMLS, the existing state-of-the-art performance sets a high standard. However, LASS-BERT$_{\text{BASE}}$ still outperforms others by at least 0.08 unit in MR. The reasons for the improvements are mainly two-fold. (i) LASS is able to capture the structural patterns in the existing triplets to predict the missing ones via the structured loss. Compared to KG-BERT, LASS is able to use the neighboring entities in the KGs for the prediction. (ii) LASS is able to maintain the semantics of the KGs through semantic embedding to avoid unreasonable triplets with high ranks. For example, if *CEO_Of* holds between two entities, the *employee_Of* also holds, but *birth_Place* does not hold. This is the main reason that LASS outperforms all structure embedding based methods by a large margin especially in MR. For instance, LASS significantly outperforms TransE, which shares the similar structured loss with LASS. Compare to the improvements made on FB15k-237, LASS-RoBERTa$_{\text{LARGE}}$ has significantly improved the state-of-the-art results on WN18RR. The main reason leading to such significant improvements is that the pre-trained LMs provide more semantics in the semantic embedding for WordNet as those LMs are trained on textual corpora to capture rela-

tionships between words. While WordNet provides the relationships between words, FB15k-237 contains real-world entities and relations, which are less captured by the LMs.

We also notice that LASS only produces moderate Hits@10 on FB15k-237. The main reason is that FB15k-237 presents more complex relations between entities compared to other link prediction datasets shown in Table 1. Therefore, a more complex structured loss is expected for LASS to gain further improvements. We leave it as one of the future explorations. Besides, on FB15k-237 and WN18RR, LASS-BERT$_{\text{LARGE}}$ outperforms LASS-BERT$_{\text{BASE}}$, and LASS-RoBERTa$_{\text{LARGE}}$ also outperforms LASS-RoBERTa$_{\text{BASE}}$. This confirms the recent findings (Petroni et al., 2019) that larger LMs store more semantic knowledge in the parameters. We expect further improvements when larger LMs are used with LASS. On UMLS, we observe slightly different trends. This is mainly because UMLS is a relatively small dataset, thus large models can suffer from overfitting. Overall, RoBERTa improves the BERT pre-training procedure from several perspectives. The improved pre-training procedure enables RoBERTa to generate better performance in many downstream tasks. This suggests that an improved pre-training procedure can enrich the semantics learned in the corresponding LMs.

Both link prediction and triplet classification are core KG completion tasks. The results show that the proposed LASS generalizes well in KG completion tasks. Different from KG-BERT that designs different models for the tasks, our method does not introduce task-specific parameters or losses for different tasks.

3.5 Case Study

We show uncurated examples to illustrate why LASS can yield the above results, especially how the parameters of the LMs capture the semantics and structures. As attention layers are basic building blocks of the LMs, we focus on visualizing the attention weights with different input sequences.

We use BertViz (Vig, 2019) to illustrate the attention weights of the LMs. Given an example of a positive triplet, h = "symbololatry, the worship of symbols", r = "hypernym", and t = "veneration, religious zeal", Figure 3a shows the attention weights of the last layer of LASS-BERT$_{\text{BASE}}$ on WN11. We find that semantically related tokens

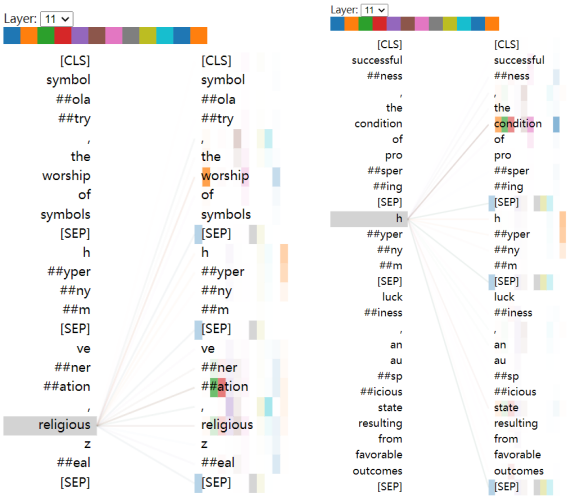

(a) Semantics. (b) Structures.

Figure 3: Illustration of attention weights of the last layer of LASS-BERT$_{\text{BASE}}$.

attend to each other with relatively high scores. For example, "religious" attends intensively to "worship" and "veneration". As in multi-head self attention (Vaswani et al., 2017), different attention heads in different colors attend to different aspects of the input, the heads are then concatenated to compute the final attention weights. The darker the color, the larger the attention score. This demonstrates that the semantic embedding of LASS is effective in capturing the semantics in the natural language description of the triplets.

We show another positive example with h = "successfulness, the condition of prospering", r = "hypernym", and t = "luckiness, an auspicious state resulting from favorable outcomes". Figure 3b illustrates the attention weights of the last layer of LASS-BERT$_{\text{BASE}}$ on WN11. We observe that tokens are highly attended to each other with similar structure roles in the triplet, even though they share fewer semantic similarities. For instance, the attention score between "hypernym" and "condition" is large. There is also a large attention score between "hypernym" and "state". This is because both "condition" and "state" capture the critical structure information of the triplet. The results indicate that the structure embedding of LASS is able to reconstruct the structure information in the semantic embeddings.

3.6 Error Analysis

To better understand the limitations of LASS, we perform a detailed analysis of the errors. We use triplet classification as an example. We investigate

the errors made by LASS-BERT$_{\text{BASE}}$ on WN11 and summarize the errors based on the relations in Table 5. We find most errors are caused by relations that are hard to be distinguished from each other due to their semantic similarities. For example, "domain topic" and "domain region" are such relations with an unclear semantic boundary.

Relation	**Percentage** (%)
domain topic	19.8
domain region	10.8
member meronym	9.1
has instance	8.4
has part	8.1
similar to	7.1
part of	6.3
synset domain topic	5.5
type of	4.6
member holonym	3.9
subordinate instance of	3.2

Table 5: Analysis of most common errors of LASS-BERT$_{\text{BASE}}$ categorized by relations on WN11.

4 Discussion

Structure Losses There are several directions to further improve LASS. LASS uses the probabilistic structured loss based on the score function of TransE, which learns a single representation for every entity and relation in the same embedding space. However, different relationships expect different entity embeddings. We propose to enable an entity to have distinct distributed representations when involved in different relations. For example, a new score function $\|\mathbf{h}_r + \mathbf{r} - \mathbf{t}_r\|_2^2$ models entities and relations in distinct spaces, and performs the translation between entity embeddings in relation space. The idea is in the same spirit as TransH (Wang et al., 2014b) and TransR (Lin et al., 2015). However, a downside of leveraging those losses is that they will bring additional computation overhead. Our method aims to trade off the computation costs and effectiveness. Exploring computation-light methods that involve alternative losses is one of the future investigations.

Pre-trained LMs We have explored two pre-trained LM families: BERT and RoBERTa. There are three possible directions along this line. First, as indicated in the experimental findings, larger LMs often store more semantics, which can improve the semantic embedding module of LASS. We propose to examine larger pre-trained LMs, such as GPT-2 (Radford et al., 2019), GPT-3 (Brown et al., 2020), and Megatron-LM (Shoeybi et al., 2019). Incorporating longer language descriptions (e.g., Wikipedia page) of the entities in the knowledge graphs will provide richer knowledge for improved natural language understanding. Second, the fine-tuning procedure of the deep LMs for KG completion tasks, especially link prediction, is still computationally inefficient. Investigating light LM architectures, such as ALBERT (Lan et al., 2020), to speed up the training process, is one of the promising directions. Finally, our proposed method is generally useful for many knowledge-driven downstream NLP tasks (e.g., question answering, factual probing) as well as low-resource NLP tasks. Ensembling our method with autoregressive models (e.g., GPT-2) will enable the method to perform text generation tasks.

5 Related Work

Pre-trained LMs Pre-trained LMs, such as BERT, have recently been used to obtain state-of-the-art results in many NLP benchmarks (Devlin et al., 2019; Liu et al., 2019). These models are usually based on Transformers (Vaswani et al., 2017) and trained on unlabeled text corpora. They are used to improve downstream tasks via embedding (Peters et al., 2018), fine-tuning (Radford et al., 2018), or few-shot learning (Radford et al., 2019). Fine-tuning bidirectional Transformers is the most widely used scheme in recent NLP applications, and the approach described in this paper is also based on this scheme. The main difference is that we design a structured loss on top of the LMs aiming to capture structures in natural language.

Knowledge Graph Embedding KG embedding aims to map entities and their relations to a continuous vector space. Traditional KG embedding methods represent each entity or each relation with a fixed vector. For any triplet (h,r,t), they use a scoring function $f(\mathbf{h}, \mathbf{r}, \mathbf{t})$ to model its likelihood. The scoring function of TransE (Bordes et al., 2013) is a negative translational distance. It can be augmented with different geometric transformations such as linear projections (Wang et al., 2014b; Lin et al., 2015) or rotations (Sun et al., 2019). Other models based on bilinear transformations (Yang et al., 2015), and convolutions (Dettmers et al., 2018), also show promising results on KG completion benchmarks. Our structured loss is motivated

by TransE. The main differences are the following. TransE (Bordes et al., 2013) treats the relation as a translation of the embeddings from the head to the tail. Therefore $\mathbf{h} + \mathbf{r} \approx \mathbf{t}$ when *(h, r, t)* holds. TransE designs a margin-based ranking loss based on the l_2 norm $\|\mathbf{h} + \mathbf{r} - \mathbf{t}\|_2^2$. The key differences between LASS and TransE are: (i) LASS leverages the natural language semantics in LMs, while TransE does not; (ii) LASS is a probabilistic structured loss, and is more computationally efficient and data-efficient compared to TransE. The main advantage of the probabilistic loss is that we eliminate the norm calculation that TransE requires to prevent the training process from trivially minimizing its loss by increasing the embeddings of entities or relations. The ranking loss of TransE calculates the loss of some training examples as zeros, which will not contribute to the optimization procedure. Our probabilistic loss makes use of all the training examples. Besides, we introduce corrupted relations in the loss, which provides more flexibility in incorporating the KG structure.

Traditional KG embedding approaches aforementioned regard entities and relations as basic units, without using any extra information. However, studies (Socher et al., 2013; Wang et al., 2014a; Xie et al., 2016) show that a KG model that models the natural language descriptions of entities and relations usually outperforms those methods that only model the structure of knowledge triplets. Petroni et al. (2019) use LMs as virtual KGs to answer factual questions. ERNIE (Zhang et al., 2019b) integrates structural KGs into pre-trained models to improve knowledge-driven NLP tasks. By contrast, we aim to combine both the structures and semantics of the KGs via a unified optimization procedure for the task of KG completion. KG-BERT (Yao et al., 2019) models KG completion tasks as sentence classification tasks and solves them by fine-tuning pre-trained LMs. There are several key differences between our LASS and KG-BERT (Yao et al., 2019): (i) LASS reconstructs the structures of KGs via structure embedding, while KG-BERT does not; (ii) LASS unifies the link prediction and triplet classification under the same architecture, while KG-BERT designs different architectures for different tasks; (iii) LASS works with two families of LMs, while KG-BERT only works with $\text{BERT}_{\text{BASE}}$. LASS is not particularly designed for BERT, shedding light on understanding the role of semantics in LMs for KG completion.

6 Conclusion

We propose a new embedding method that leverages both semantics and structures of the knowledge graphs for the task of knowledge graph completion, and offers additional benefits in low-resource settings. The method maps a knowledge graph triplet to an embedding space via fine-tuning language models, where the forward pass captures semantics and the loss reconstructs structures. Our method has shown significant improvements on knowledge graph completion benchmarks. The implementation has made no modifications to the language model architectures. The results suggest that the learned embeddings are generally useful in downstream knowledge-driven applications, and potentially useful for more natural language understanding tasks. We hope our results will foster further research in this direction.

7 Ethical Considerations

We hereby acknowledge that all of the co-authors of this work are aware of the provided *ACM Code of Ethics* and honor the code of conduct. The followings give the aspects of both our ethical considerations and our potential impacts to the community. This work uses pre-trained LMs for knowledge graph completion. The risks and potential misuse of LMs are discussed in Brown et al. (2020). There are potential undesirable biases in the datasets, such as unfaithful descriptions from Wikipedia. We do not anticipate the production of harmful outputs after using our model, especially towards vulnerable populations.

8 Environmental Considerations

We use BERT and RoBERTa as our pre-trained LMs. According to the estimation in Strubell et al. (2019), pre-training a base model costs 1,507 kWh·PUE and emits 1,438 lb CO_2, while pre-training a large model requires 4 times the resources of a base model. In addition, our fine-tuning takes less than 1% gradient-steps of the number of steps of pre-training. Therefore, our energy cost and CO_2 emissions are relatively small. Besides, the results in the low-resource settings show that our method has better sampling efficiency. This indicates that we can further reduce energy consumption when training with fewer data.

Acknowledgement

We would like to thank the anonymous reviewers for their suggestions and comments. This material is in part based upon work supported by Berkeley DeepDrive and Berkeley Artificial Intelligence Research.

References

Bo An, Bo Chen, Xianpei Han, and Le Sun. 2018. Accurate text-enhanced knowledge graph representation learning. In *NAACL-HLT*, pages 745–755.

Kurt Bollacker, Colin Evans, Praveen Paritosh, Tim Sturge, and Jamie Taylor. 2008. Freebase: a collaboratively created graph database for structuring human knowledge. In *SIGMOD*, pages 1247–1250.

Antoine Bordes, Nicolas Usunier, Alberto Garcia-Durán, Jason Weston, and Oksana Yakhnenko. 2013. Translating embeddings for modeling multi-relational data. In *NeurIPS*, pages 2787–2795.

Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel Ziegler, Jeffrey Wu, Clemens Winter, Chris Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. 2020. Language models are few-shot learners. In *NeurIPS*, pages 1877–1901.

Andrew Carlson, Justin Betteridge, Bryan Kisiel, Burr Settles, Estevam R. Hruschka, and Tom M. Mitchell. 2010. Toward an architecture for never-ending language learning. In *AAAI*, pages 1306–1313.

Ines Chami, Adva Wolf, Da-Cheng Juan, Frederic Sala, Sujith Ravi, and Christopher Ré. 2020. Low-dimensional hyperbolic knowledge graph embeddings. In *ACL*, pages 6901–6914.

Yihong Chen, Pasquale Minervini, Sebastian Riedel, and Pontus Stenetorp. 2021. Relation prediction as an auxiliary training objective for improving multi-relational graph representations. In *AKBC*.

Tim Dettmers, Pasquale Minervini, Pontus Stenetorp, and Sebastian Riedel. 2018. Convolutional 2d knowledge graph embeddings. In *AAAI*, pages 1811–1818.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. Bert: Pre-training of deep bidirectional transformers for language understanding. In *NAACL-HLT*, pages 4171–4186.

Kelvin Guu, John Miller, and Percy Liang. Traversing knowledge graphs in vector space. In *EMNLP*, pages 318–327.

Yanchao Hao, Yuanzhe Zhang, Kang Liu, Shizhu He, Zhanyi Liu, Hua Wu, and Jun Zhao. 2017. An end-to-end model for question answering over knowledge base with cross-attention combining global knowledge. In *ACL*, pages 221–231.

Guoliang Ji, Shizhu He, Liheng Xu, Kang Liu, and Jun Zhao. 2015. Knowledge graph embedding via dynamic mapping matrix. In *ACL-IJCNLP*, pages 687–696.

Guoliang Ji, Kang Liu, Shizhu He, and Jun Zhao. 2016. Knowledge graph completion with adaptive sparse transfer matrix. In *AAAI*, pages 985–991.

Shaoxiong Ji, Shirui Pan, Erik Cambria, Pekka Marttinen, and S Yu Philip. 2021. A survey on knowledge graphs: Representation, acquisition, and applications. *TNNLS*, 33:494–514.

Zhenzhong Lan, Mingda Chen, Sebastian Goodman, Kevin Gimpel, Piyush Sharma, and Radu Soricut. 2020. Albert: A lite bert for self-supervised learning of language representations. In *ICLR*.

Yankai Lin, Zhiyuan Liu, Maosong Sun, Yang Liu, and Xuan Zhu. 2015. Learning entity and relation embeddings for knowledge graph completion. In *AAAI*, pages 2181–2187.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Roberta: A robustly optimized bert pretraining approach. *CoRR*.

Ilya Loshchilov and Frank Hutter. 2019. Decoupled weight decay regularization. In *ICLR*.

Alexa T. McCray. 2003. An upper-level ontology for the biomedical domain. *Comparative and Functional Genomics*, pages 80–84.

Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013. Distributed representations of words and phrases and their compositionality. In *NeurIPS*, pages 3111–3119.

George A. Miller. 1994. Wordnet: A lexical database for english. In *HLT*.

Deepak Nathani, Jatin Chauhan, Charu Sharma, and Manohar Kaul. 2019. Learning attention-based embeddings for relation prediction in knowledge graphs. In *ACL*, pages 4710–4723.

Dai Quoc Nguyen, Tu Dinh Nguyen, Dat Quoc Nguyen, and Dinh Phung. 2018. A novel embedding model for knowledge base completion based on convolutional neural network. In *NAACL-HLT*, pages 327–333.

Thomas Pellissier Tanon, Denny Vrandečić, Sebastian Schaffert, Thomas Steiner, and Lydia Pintscher. 2016. From freebase to wikidata: The great migration. In *WWW*, pages 1419–1428.

Matthew E. Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer. 2018. Deep contextualized word representations. In *NAACL-HLT*, pages 2227–2237.

Fabio Petroni, Tim Rocktäschel, Patrick Lewis, Anton Bakhtin, Yuxiang Wu, Alexander H. Miller, and Sebastian Riedel. 2019. Language models as knowledge bases? In *EMNLP-IJCNLP*, pages 2463–2473.

Alec Radford, Karthik Narasimhan, Tim Salimans, and Ilya Sutskever. 2018. Improving language understanding by generative pre-training. *OpenAI blog*.

Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. 2019. Language models are unsupervised multitask learners. *OpenAI blog*.

Mohammad Shoeybi, Mostofa Patwary, Raul Puri, Patrick LeGresley, Jared Casper, and Bryan Catanzaro. 2019. Megatron-lm: Training multi-billion parameter language models using model parallelism. *CoRR*.

Richard Socher, Danqi Chen, Christopher D. Manning, and Andrew Y. Ng. 2013. Reasoning with neural tensor networks for knowledge base completion. In *NeurIPS*, pages 926–934.

Shashank Sonkar, Arzoo Katiyar, and Richard G. Baraniuk. 2021. Neptune: Neural powered tucker network for knowledge graph completion. *CoRR*.

Emma Strubell, Ananya Ganesh, and Andrew McCallum. 2019. Energy and policy considerations for deep learning in NLP. In *ACL*, pages 3645–3650.

Zhiqing Sun, Zhi-Hong Deng, Jian-Yun Nie, and Jian Tang. 2019. Rotate: Knowledge graph embedding by relational rotation in complex space. In *ICLR*.

Kristina Toutanova and Danqi Chen. 2015. Observed versus latent features for knowledge base and text inference. In *CVSC-WS*, pages 57–66.

Kristina Toutanova, Danqi Chen, Patrick Pantel, Hoifung Poon, Pallavi Choudhury, and Michael Gamon. 2015. Representing text for joint embedding of text and knowledge bases. In *EMNLP*, pages 1499–1509.

Théo Trouillon, Johannes Welbl, Sebastian Riedel, Eric Gaussier, and Guillaume Bouchard. 2016. Complex embeddings for simple link prediction. In *ICML*, pages 2071–2080.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *NeurIPS*, pages 5998–6008.

Jesse Vig. 2019. A multiscale visualization of attention in the transformer model. In *ACL*, pages 37–42.

Bo Wang, Tao Shen, Guodong Long, Tianyi Zhou, and Yi Chang. 2021. Structure-Augmented Text Representation Learning for Efficient Knowledge Graph Completion. In *WWW*, pages 1737–1748.

Haoyu Wang, Vivek Kulkarni, and William Yang Wang. 2018. Dolores: Deep contextualized knowledge graph embeddings. In *AKBC*.

Rui Wang, Bicheng Li, Shengwei Hu, Wenqian Du, and Min Zhang. 2020. Knowledge Graph Embedding via Graph Attenuated Attention Networks. *IEEE Access*, 8:5212–5224.

Zhen Wang, Jianwen Zhang, Jianlin Feng, and Zheng Chen. 2014a. Knowledge graph and text jointly embedding. In *EMNLP*, pages 1591–1601.

Zhen Wang, Jianwen Zhang, Jianlin Feng, and Zheng Chen. 2014b. Knowledge graph embedding by translating on hyperplanes. In *AAAI*, pages 1112–1119.

Zhigang Wang and Juanzi Li. 2016. Text-enhanced representation learning for knowledge graph. In *IJCAI*, page 1293–1299.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Rémi Louf, Morgan Funtowicz, and et al. 2020. Huggingface's transformers: State-of-the-art natural language processing. *CoRR*.

Han Xiao, Minlie Huang, and Xiaoyan Zhu. 2016. Transg: A generative model for knowledge graph embedding. In *ACL*, pages 2316–2325.

Ruobing Xie, Zhiyuan Liu, Jia Jia, Huanbo Luan, and Maosong Sun. 2016. Representation learning of knowledge graphs with entity descriptions. In *AAAI*, pages 2659–2665.

Chenyan Xiong, Russell Power, and Jamie Callan. 2017. Explicit semantic ranking for academic search via knowledge graph embedding. In *WWW*, pages 1271–1279.

Bishan Yang, Wen-tau Yih, Xiaodong He, Jianfeng Gao, and Li Deng. 2015. Embedding entities and relations for learning and inference in knowledge bases. In *ICLR*.

Liang Yao, Chengsheng Mao, and Yuan Luo. 2019. Kg-bert: Bert for knowledge graph completion. *CoRR*.

Zhanqiu Zhang, Jianyu Cai, and Jie Wang. 2020. Duality-induced regularizer for tensor factorization based knowledge graph completion. In *NeurIPS*, pages 21604–21615.

Zhanqiu Zhang, Jianyu Cai, Yongdong Zhang, and Jie Wang. 2019a. Learning Hierarchy-Aware Knowledge Graph Embeddings for Link Prediction. In *AAAI*, pages 3065–3072.

Zhao Zhang, Fuzhen Zhuang, Meng Qu, Fen Lin, and Qing He. 2018. Knowledge graph embedding with hierarchical relation structure. In *EMNLP*, pages 3198–3207.

Zhengyan Zhang, Xu Han, Zhiyuan Liu, Xin Jiang, Maosong Sun, and Qun Liu. 2019b. Ernie: Enhanced language representation with informative entities. In *ACL*, pages 1441–1451.

A Experimental Setup Details

We describe additional details of our experimental setup including datasets and comparison methods in this section.

A.1 Datasets

We introduce the link prediction and triplet classification datasets as below.

A.1.1 Link Prediction

- **FB15k-237**. Freebase is a large collaborative knowledge graph consisting of data composed mainly by its community members. It is an online collection of structured data harvested from many sources, including individual and user-submitted wiki contributions (Pellissier Tanon et al., 2016). FB15k is a selected subset of Freebase that consists of 14,951 entities and 1,345 relationships (Bordes et al., 2013). FB15K-237 is a variant of FB15K where inverse relations and redundant relations are removed, resulting in 237 relations (Toutanova et al., 2015).

- **WN18RR**. WordNet is a lexical database of semantic relations between words in English. WN18 (Bordes et al., 2013) is a subset of WordNet which consists of 18 relations and 40,943 entities. WN18RR is created to ensure that the evaluation dataset does not have inverse relations to prevent test leakage (Dettmers et al., 2018).

- **UMLS**. UMLS semantic network (McCray, 2003) is an upper-level ontology of Unified Medical Language System. The semantic network, through its 135 semantic types, provides a consistent categorization of all concepts represented in the UMLS. The 46 links between the semantic

Method	Score Function	
TransE	$-\|\mathbf{h}+\mathbf{r}-\mathbf{t}\|$	$\mathbf{h},\mathbf{r},\mathbf{t}\in\mathbb{R}^k$
TransH	$-\left\|\left(\mathbf{h}-\mathbf{w}_r^\top\mathbf{h}\mathbf{w}_r\right)+\mathbf{r}-\left(\mathbf{t}-\mathbf{w}_r^\top\mathbf{t}\mathbf{w}_r\right)\right\|$	$\mathbf{h},\mathbf{t},\mathbf{r},\mathbf{w}_r\in\mathbb{R}^k$
TransR	$-\|\mathbf{M}_r\mathbf{h}+\mathbf{r}-\mathbf{M}_r\mathbf{t}\|$	$\mathbf{h},\mathbf{t}\in\mathbb{R}^k,\mathbf{M}_r\in\mathbb{R}^{k\times d}$
TransD	$-\left\|\left(\mathbf{w}_r\mathbf{w}_h^\top+\mathbf{I}\right)\mathbf{h}+\mathbf{r}-\left(\mathbf{w}_r\mathbf{w}_t^\top+\mathbf{I}\right)\mathbf{t}\right\|$	$\mathbf{h},\mathbf{t},\mathbf{w}_h\mathbf{w}_t\in\mathbb{R}^k,\mathbf{r},\mathbf{w}_r\in\mathbb{R}^d$
TransG	$\sum_i \pi_r^i \exp\left(-\frac{\|\boldsymbol{\mu}_h+\boldsymbol{\mu}_r^i-\boldsymbol{\mu}_t\|}{\sigma_h^2+\sigma_t^2}\right)$	$\mathbf{h}\sim\mathcal{N}\left(\boldsymbol{\mu}_h,\boldsymbol{\sigma}_h^2\mathbf{I}\right),\mathbf{t}\sim\mathcal{N}\left(\boldsymbol{\mu}_t,\Sigma_t\right),\boldsymbol{\mu}_h,\boldsymbol{\mu}_t\in\mathbb{R}^k$
TranSparse-S	$-\left\|\mathbf{M}_r\left(\theta_r\right)\mathbf{h}+\mathbf{r}-\mathbf{M}_r\left(\theta_r\right)\mathbf{t}\right\|_{1/2}^2-\left\|\mathbf{M}_r^1\left(\theta_r^1\right)\mathbf{h}+\mathbf{r}-\mathbf{M}_r^2\left(\theta_r^2\right)\mathbf{t}\right\|_{1/2}^2$	$\mathbf{h},\mathbf{t}\in\mathbb{R}^k,\mathbf{r}\in\mathbb{R}^d,\mathbf{M}_r\left(\theta_r\right)\in\mathbb{R}^{k\times d},\mathbf{M}_r^1\left(\theta_r^1\right),\mathbf{M}_r^2\left(\theta_r^2\right)\in\mathbb{R}^{k\times d}$
DistMult	$\langle\mathbf{r},\mathbf{h},\mathbf{t}\rangle$	$\mathbf{h},\mathbf{r},\mathbf{t}\in\mathbb{R}^k$
ConvKB	$\text{concat}(g([\boldsymbol{h},\boldsymbol{r},\boldsymbol{t}]*\omega))\mathbf{w}$	$\mathbf{h},\mathbf{r},\mathbf{t}\in\mathbb{R}^k$
ComplEx	$\Re(\langle\mathbf{r},\mathbf{h},\bar{\mathbf{t}}\rangle)$	$\mathbf{h},\mathbf{r},\mathbf{t}\in\mathbb{C}^k$
ConvE	$\langle\sigma(\text{vec}(\sigma([\bar{\mathbf{r}},\bar{\mathbf{h}}]*\boldsymbol{\Omega}))\mathbf{W}),\mathbf{t}\rangle$	$\mathbf{h},\mathbf{r},\mathbf{t}\in\mathbb{R}^k$
RotatE	$-\|\mathbf{h}\circ\mathbf{r}-\mathbf{t}\|^2$	$\mathbf{h},\mathbf{r},\mathbf{t}\in\mathbb{C}^k,\|r_i\|=1$
REFE	$-\text{arctanh}(\|-\langle\mathbf{h},\text{Ref}(\mathbf{r})\rangle\oplus^c\mathbf{t}\|)$	$\mathbf{h},\mathbf{r},\mathbf{t}\in\mathbb{R}^k$
HAKE	$\text{RotatE}-\|\sin((\mathbf{h}+\mathbf{r}-\mathbf{t})/2)\|_1$	$\mathbf{h},\mathbf{r},\mathbf{t}\in\mathbb{R}^k$
ComplEx-DURA	$\text{ComplEx}-\langle\mathbf{h},\mathbf{r}\rangle^2-\|\mathbf{t}\|^2$	$\mathbf{h},\mathbf{r},\mathbf{t}\in\mathbb{C}^k$

Table 6: The score functions $f_r(\mathbf{h},\mathbf{t})$ of shallow structure embedding models for knowledge graph embedding, where $\langle\cdot\rangle$ denotes the generalized dot product, $\circ$ denotes the Hadamard product, σ denotes activation function and $*$ denotes 2D convolution. $\bar{\cdot}$ denotes conjugate for complex vectors, and 2D reshaping for real vectors in the ConvE model. $\text{Ref}(\theta)$ denotes the reflection matrix induced by rotation parameters θ. $\oplus^c$ is Möbius addition that provides an analogue to Euclidean addition for hyperbolic space.

types provide the structure for the network and represent important relationships in the biomedical domain.

A.1.2 Triplet Classification

- **WN11 and FB13** are subsets of WordNet and FreeBase respectively for triplet classification, where Socher et al. (2013) randomly switch entities from correct testing triplets resulting in a total of doubling the number of test triplets with an equal number of positive and negative examples.

A.2 Comparison Methods

We compare LASS to three types of knowledge graph completion methods: shallow structure embedding, deep structure embedding, and language semantic embedding.[1]

A.2.1 Shallow Structure Embedding

TransE (Bordes et al., 2013), TransH (Wang et al., 2014b), TransR (Lin et al., 2015), TransD (Ji et al., 2015), TransG (Xiao et al., 2016), TranSparse-S (Ji et al., 2016), DistMult (Yang et al., 2015), ConvKB (Nguyen et al., 2018), ComplEx (Trouillon et al., 2016), ConvE (Dettmers et al., 2018), RotatE (Sun et al., 2019), REFE (Chami et al., 2020), HAKE (Zhang et al., 2019a), and ComplEx-DURA (Zhang et al., 2020) are methods based only on the structure of the knowledge graphs. DistMult-HRS (Zhang et al., 2018) is an extension of DistMult which is combined with a three-layer hierarchical relation structure (HRS) loss. Each of these methods proposes a scoring function regarding a knowledge triplet, without using the natural language descriptions or names of entities or relations. The scoring functions are shown in Table 6.

A.2.2 Deep Structure Embedding

- **NTN** (Neural Tensor Network) (Socher et al., 2013) models entities across multiple dimensions by a bilinear tensor neural layer.

- **DOLORES** (Wang et al., 2018) is based on bi-directional LSTMs and learns deep representations of entities and relations from constructed entity-relation chains.

- **KBGAT** proposes an attention-based feature embedding that captures both entity and relation features in any given entity's neighborhood, and additionally encapsulates relation clusters and multi-hop relations (Nathani et al., 2019).

- **GAATs** integrates an attenuated attention mechanism in a graph neural network to assign different weights in different relation paths and acquire the information from the neighborhoods (Wang et al., 2020).

- **NePTuNe** takes advantage of both TuckER and NTN by carefully crafted nonlinearities and a shared core tensor intrinsic to the Tucker decomposition (Sonkar et al., 2021).

- **ComplEx-N3-RP** introduces an auxiliary training task to predict relation types as a self-supervised objective. (Chen et al., 2021).

[1] We refer the readers to (Ji et al., 2021) for a more comprehensive review of the knowledge graph completion methods.

A.2.3 Language Semantic Embedding

- **TEKE** (Wang and Li, 2016) takes advantage of the context information in a text corpus. The textual context information is incorporated to expand the semantic structure of the knowledge graph and each relation is enabled to own different representations for different head and tail entities.

- **AATE** (An et al., 2018) is a text-enhanced knowledge graph representation learning method, which can represent a relation/entity with different representations in different triples by exploiting additional textual information.

- **KG-BERT** (Yao et al., 2019) considers triples in knowledge graphs as textual sequences, where each textual sequence is a concatenation of text descriptions of the head entity, the relation, and the tail entity. Then KG-BERT treats the knowledge graph completion task as a text binary classification task, and then solves it by fine-tuning a pre-trained BERT.

- **StAR** (Wang et al., 2021) partitions each triplet into two asymmetric parts as in translation-based graph embedding approach, and encodes both parts into contextualized representations by a Siamese-style textual encoder (BERT or RoBERTa).

Event Detection with Dual Relational Graph Attention Networks

Jiaxin Mi [1,2,3], **Po Hu** [1,2,3]*, **Peng Li** [4]
[1]Hubei Provincial Key Laboratory of Artificial Intelligence and Smart Learning,
Central China Normal University, Wuhan, Hubei, China
[2]School of Computer Science, Central China Normal University, Wuhan, Hubei, China
[3]National Language Resources Monitoring & Research Center for Network Media,
Central China Normal University, Wuhan, Hubei, China
[4]Institute for AI Industry Research (AIR), Tsinghua University, China
{mjx, phu}@mail.ccnu.edu.cn, lipeng@air.tsinghua.edu.cn

Abstract

Event detection, which aims to identify instances of specific event types from pieces of text, is a fundamental task in information extraction. Most existing approaches leverage syntactic knowledge with a set of syntactic relations to enhance event detection. However, a side effect of these syntactic-based approaches is that they may confuse different syntactic relations and tend to introduce redundant or noisy information, which may lead to performance degradation. To this end, we propose a simple yet effective model named DualGAT (Dual Relational Graph Attention Networks), which exploits the complementary nature of syntactic and semantic relations to alleviate the problem. Specifically, we first construct a dual relational graph that both aggregates syntactic and semantic relations to the key nodes in the graph, so that event-relevant information can be comprehensively captured from multiple perspectives (i.e., syntactic and semantic views). We then adopt augmented relational graph attention networks to encode the graph and optimize its attention weights by introducing contextual information, which further improves the performance of event detection. Extensive experiments conducted on the standard ACE2005 benchmark dataset indicate that our method significantly outperforms the state-of-the-art methods and verifies the superiority of DualGAT over existing syntactic-based methods.

1 Introduction

Event Detection (ED) aims to identify event triggers from a given text and classify them into predefined event types (Chen et al., 2015), playing an important role in information extraction. As shown in Figure 1, an event detection method is expected to identify the event trigger "thrust" from the example sentence and classify it into the event type `Transport`. Event detection can be used to facilitate various natural language processing (NLP) applications, such as adverse drug event discovery (Wei et al., 2020; Liu et al., 2018a), court decision event identification (Filtz et al., 2020), financial event extraction (Zheng et al., 2019; Liang et al., 2020) and so on.

Various event detection methods have been explored, including traditional feature-based (Hong et al., 2011; Li et al., 2013) and deep learning methods (Chen et al., 2015; Nguyen et al., 2016). Due to the powerful feature representation extraction capability of deep neural networks, deep learning methods have outperformed traditional feature-based methods in most cases and have become popular. Most deep learning methods have recently paid more attention to exploiting syntactic relations in event detection. These methods adopt Graph Neural Networks (GNNs) such as Graph Convolutional Networks (GCNs) and Graph Attention Networks (GATs) to encode dependency trees to learn effective representations for the words. Since dependency trees convey rich linguistic information for ED, syntactic-based methods usually achieve better performance (Xie et al., 2021).

However, existing syntactic-based methods still have two shortcomings to be solved. First, dependency trees cannot always capture trigger-related salient information concisely, which may contain noisy dependency relations close to the root node and mislead event detection (Lai et al., 2020; Liu et al., 2021). As shown in Figure 1, the dependencies marked in red are incorrectly identified by the syntactic-based event detection methods as trigger-related hints for the `Attack` event. Since "troops" and the trigger candidate "striking" have a direct connection with the root node "distance", GNN-based methods are prone to pay more attention to them and predict incorrect triggers (Liu et al., 2021). It is worth noting that in this case, the marked blue dependencies that are closely related to the real trigger "thrust" should be exploited with more emphasis. Second, relying solely on syntactic

* Corresponding author

Proceedings of the 29th International Conference on Computational Linguistics, pages 1979–1989
October 12–17, 2022.

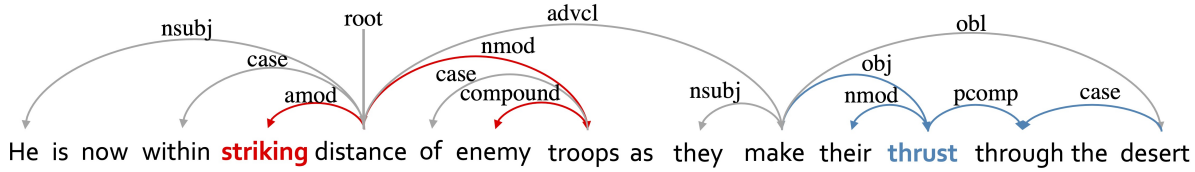

Figure 1: An event detection example with syntactic dependency relations. The "thrust" is the true trigger of `Transport` event. The "striking" is incorrectly identified as the trigger for an `Attack` event by some existing methods. The red arrow indicates a noisy dependency relation and blue arrow indicates a trigger-relevant relation.

dependency trees may not be sufficient for event detection. The parsing results of the existing dependency parser may contain incorrect or weakly correlated information, which will inevitably affect the performance of syntactic-based event detection methods due to possible error propagation. Moreover, syntactic dependency trees cannot provide all the linguistic knowledge needed for event detection.

To address the above problems, we propose a novel model named DualGAT, which makes full use of syntactic and semantic information to improve event detection performance. Inspired by aspect-based dependency parsing (Wang et al., 2020a), we construct a dual relational graph that contains both syntactic and semantic relations to capture trigger-relevant information. Empirically, only a small part of dependency relations in a sentence is task-aware (Zhang et al., 2018; He et al., 2018). Therefore, to reduce the influence of noisy relations, we prune the original syntactic dependencies that are not directly connected to the trigger candidates and reconstruct other connections between the remaining words of the sentence and the trigger candidates. In addition to syntactic information, we also introduce semantic relation information and make them rooted in the predicate of the sentence. Next, we adopt augmented relational graph attention networks to encode the graph to learn the root node's representations from syntactic and semantic views. Specially, we introduced contextual information to adjust the attention weights to mitigate the possible loss of information due to the introduction of the dependency parser. The experimental results on the standard ACE2005 benchmark dataset indicate that DualGAT outperforms the state-of-the-art methods by a large margin.

The main contributions of this work can be summarized as follows:

- We construct a dual relational graph that converges syntactic and semantic relations to the critical nodes in the graph, which can capture important event information from different perspectives and reduce information loss or noise caused by the syntactic parser.

- We adopt an augmented relational graph attention network to encode the graph and optimize its attention weights by introducing contextual information.

- Experimental results further verify the superiority of DualGAT over existing approaches. DualGAT achieves the 5% improvements in F1 score with state-of-the-art syntactic-based methods.

2 Related Work

Event detection has attracted extensive attention in recent years. Traditional event detection methods use hand-crafted linguistic features for event detection, such as lexical features, syntactic features, or entity features (Hong et al., 2011; Ahn, 2006). However, it is time-consuming to design these features and is not easy to adapt to other tasks or new domains.

With the rapid development of neural networks, a series of neural event detection methods have been proposed. Many researchers applied new learning strategies to event detection, such as leveraging the weakly-supervised learning strategy to generate more labeled data to improve the performance of event detection (Zeng et al., 2018; Yang et al., 2018). Wang et al. (2019) applied an adversarial training mechanism to obtain diverse and accurate training data. Lu et al. (2019) proposed a method based on knowledge distillation to achieve better performance on sparsely labeled triggers. Some recently proposed methods introduced extra knowledge to improve event detection via open-domain trigger knowledge (Tong et al., 2020), entity knowledge (Liu et al., 2019) and syntactic dependency relations (Yan et al., 2019).

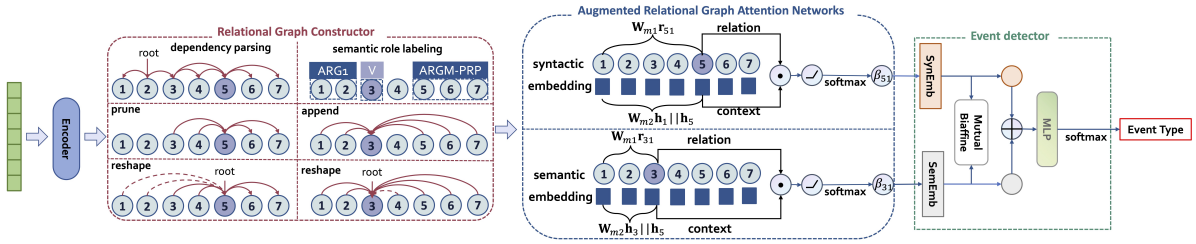

Figure 2: The architecture of our proposed dual relational graph attention networks for event detection. The 5-th node is the assumed trigger candidate and the 3-th is the predicate verb. $\mathbf{r}_{ij}$ denotes the relation embedding of e_{ij}.

The effectiveness of dependency relations has been verified in many natural language processing (NLP) tasks, such as sentiment analysis (Wang et al., 2020a) and relation extraction (He et al., 2018). Due to the rich syntactic and structure information, syntactic dependency relations also play an important role in event detection (Liu et al., 2018b). For example, Yan et al. (2019) exploited multi-order syntactic relations in sentences to obtain better representations of trigger words. Lv et al. (2021) integrated syntax and document information for better event detection. Cui et al. (2020) proposed a model to explore further the type information of dependency relations to capture task-aware knowledge. Since existing graph-based models introduced many trigger-agnostic representations, Lai et al. (2020) proposed to filter noisy information via a gating mechanism. Due to the effective combination of GNNs and syntactic dependency trees, these syntactic-based event detection methods achieved overall better performance than ordinary deep learning methods.

Although these works use similar syntactic information, few of them take a new perspective to reshape the original graph to facilitate the capture of event-relevant salient information. The original dependency tree contains rich structural and linguistic knowledge, but it may be inaccurate and contains event-irrelevant information. Besides, as far as we know, there are no syntactic-based approaches that explicitly use semantic information to complement syntactic information for event detection. To this end, we propose DualGAT, which takes into account both syntactic and semantic relations as well as contextual information for event detection.

3 Methodology

3.1 Model Overview

The overall architecture of DualGAT is shown in Figure 2, which consists of three major components: (1) *Relational Graph Constructor* (§3.2), which constructs the dual relational graph by dependency parsing and semantic role labeling; (2) *Augmented Relational Graph Attention Networks* (§3.3), which introduces additional contextual information into the adaptation of attention weight and encodes the dual relational graph to get the root node's representations from syntactic and semantic views; (3) *Event detector* (§3.4), which leverages the Biaffine module to exchange relevant features between syntactic and semantic representations and performs event detection.

3.2 Relational Graph Constructor

3.2.1 Dual Relational Graph

Existing methods are interfered with by noisy dependency relations and tend to learn trigger-agnostic representations (Lai et al., 2020). Since the trigger candidates are the focus of the event detection task, this noisy information irrelevant to the trigger words inevitably hurts the performance. Besides, the parsing results of the existing dependency parser may contain incorrect or weakly correlated information, which limits the performance of event detection.

Many works have proven that only a small part of the syntactic relations is task-aware (Zhang et al., 2018; He et al., 2018) and identifying trigger words is the core of event detection tasks. Thus, we believe that paying more attention to syntactic relations directly linked to trigger candidates can reduce the effect of erroneous syntactic relations. Besides, the complementary nature of semantic and syntactic knowledge has been exploited and validated in related NLP tasks, such as relations extraction (Bovi et al., 2015) and entity extraction (Chan and Roth, 2011). We believe that semantic relations can make up for the inadequacy of syntactic relations and reduce reliance on dependency parsers.

Motivated by the above idea, we first construct a dual relational graph structure that aggregates

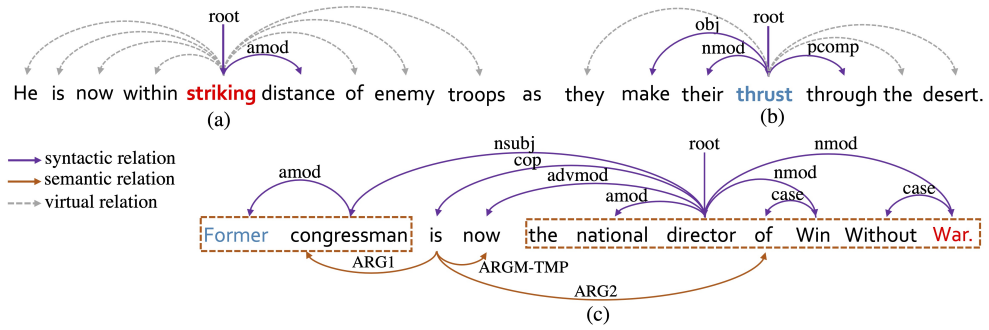

Figure 3: Three event detection examples to illustrate the advantages of the proposed dual relational graph. The correct trigger is marked blue and the incorrect one is marked red. Example (a) and example (b) are two dual relational graphs of one sentence that omit semantic relations. In example (c), the purple arrow is the original syntactic relation, and the orange arrow is the additional semantic relation of the dual relational graph.

Algorithm 1 The Construction Process of Dual Relational Graph.

Input:

The sentence $X = \{x_i \mid i \in [1, L]\}$ with a trigger candidate x_t and a predicate verb x_v.

Output:

Dual relational graph G;

Getting syntactic, semantic edge set D, A.
Construct initial dual relational graph G with L nodes.
for $i = 1$ to L **do**
  if e_{it} or e_{ti} in D **then**
    add e_{ti} in G.
  else
    $d = t - i + 1$, $e_{ti} = vir : d$.
    add e_{ti} in G
  end if
  if e_{iv} or e_{vi} in A **then**
    add e_{vi} in G.
  else
    $d = v - i + 1$, $e_{vi} = vir : d$.
    add e_{vi} in G
  end if
end for
return G;

syntactic relations to trigger candidates and additionally aggregates semantic relations to predicate verbs to improve the robustness of the graph. Algorithm 1 describes the construction process of the dual relational graph. We first leverage a dependency parser to obtain the original syntactic relations of a given sentence. Then we retain the syntactic relations directly connected to the trigger candidates and prune the remaining relations. To improve the robustness of the graph, we further replace the pruned relations with a virtual one classified by distance with trigger candidates. Specifically, the type of virtual relations is defined as $vir : d$, where d represents the distance between a word and the trigger candidate. Finally, we perform semantic role labeling to append semantic relations between other words and predicate verbs and do the same process as above.

The graph obtained after the above processing is the dual relational graph. It has two same levels of subgraphs: the syntactic relational subgraph, which converges syntactic information to the trigger candidates, and the semantic relational subgraph, which converges semantic information to the predicate verbs. Formally, we define the dual relational graph as $G = (V, E)$ associated with a edge type mapping function $\tau : E \rightarrow T_E$. V represents the set of word nodes in a sentence, and E represents the set of relational edges. T_E represents all types of relations including syntactic relations, semantic relations and virtual relations. The relational edge between word node i and word node j is defined as e_{ij}.

3.2.2 The Advantages of Dual Relational Graph

The dual relational graph has proprietary advantages in event detection. First, the dual relational graph is customized for every trigger candidate, which can reduce the introduction of noisy trigger-agnostic information. Second, the semantic relations rooted in predicate verbs can improve the robustness of the dual relational graph. Third, the dual relational graph structure facilitates the event detection model to capture task-aware information. We use three simple examples to illustrate the above advantages, as shown in Figure 3.

First, example (a) and example (b) are two customized dual relational graphs for different trigger candidates in the same sentence. The existing methods tend to identify "striking" as a trigger evoking an `Attack` event, since the strong bonding between "striking", "enemy", and "troops" in the original dependency tree. However, the dual relational graph of "striking" prunes the trigger-agnostic relations and clearly illustrates that "striking" is only an adjectival modifier (amod) of "distance". Moreover, the dual relational graph of "thrust" can facilitate ED models to capture its connections with "through" and "make" without the interference of noisy dependency relations in the original dependency tree. Thus, ED methods tend to identify "thrust" as a trigger evoking a `Transport` event rather than "striking".

Besides, example (c) can illustrate the importance of semantic relations. In example (c), the purple arrow is the original syntactic relation, and the orange arrow is the additional semantic relation of the dual relational graph. Based on original syntactic relations, ED methods tend to identify "War" as a trigger of an `Attack` event, since the strong bonding between "War" and "Win". Even pruning the trigger-agnostic syntactic relations, the correct trigger "Former" still cannot obtain much attention. However, the semantic relations can make the deep semantic information flow from "now" to "Former". The "ARGM-TMP" relation introduces timing information to help identify "Former" as a trigger for an `End-position` event.

Finally, as shown in examples (a) and (b), the dual relational graph does not have complex mutual interactions. The structure of the dual relational graph is clear and aggregated. It can reduce the interference of noisy interactions and the ED model's difficulty in capturing trigger-relevant information.

3.3 Augmented Relational Graph Attention Networks

To more effectively encode the dual relational graph for event detection, we propose an augmented relational graph attention network (AR-GAT) by introducing additional contextual information to encode graphs constructed from words in a sentence.

3.3.1 Graph Attention Network

Graph neural networks (Scarselli et al., 2009) have been widely used to encode dependency trees for event detection, as they can effectively capture relevant information based on an information aggregation scheme (Cao et al., 2021). In addition, much work has shown that graph convolutional networks (Schlichtkrull et al., 2018) cannot effectively leverage multi-hop relational information (Yan et al., 2019). Intuitively, the heart of the event detection task is to capture the relevant words with the trigger candidates. Thus, we apply graph attention networks (Velickovic et al., 2018) which can more efficiently leverage the relations between words to encode the dual relational graph.

Formally, given a dual relational graph G with L word nodes, and the set of neighborhood nodes of node i is defined as $\mathcal{N}_i$. The feature vector of node i at layer l is denoted as $\mathbf{h}_i^l$ $\left(\mathbf{h}_i^l \in \mathbb{R}^F\right)$, F is the dim of node features. For the node i at the layer l+1, the computation of multi-head graph attention networks can be defined as follows:

$$\mathbf{h}_{att_i}^{l+1} = \|_{k=1}^{K} \sigma\left(\sum_{j \in \mathcal{N}_i} \alpha_{ij}^{lk} \mathbf{W}_k^l \mathbf{h}_j^l\right) \quad (1)$$

$$\alpha_{ij}^{lk} = \frac{\exp(f_1(\mathbf{a}^T[\mathbf{W}_k^l \mathbf{h}_i^l \| \mathbf{W}_k^l \mathbf{h}_j^l]))}{\sum_{t \in \mathcal{N}_i} \exp(f_1(\mathbf{a}^T[\mathbf{W}_k^l \mathbf{h}_i^l \| \mathbf{W}_k^l \mathbf{h}_t^l]))} \quad (2)$$

where $\mathbf{h}_{att_i}^{l+1}$ is the attention head of node i at the layer l+1, $\mathbf{W}_k^l$ is a transformation matrix, $f_1(\cdot)$ is the function of LeakyReLU, $\mathbf{a}$ is a weight vector, K is the number of heads and $\|_{k=1}^{K} \mathbf{h}_k$ represents concatenation of vectors from $\mathbf{h}_1$ to $\mathbf{h}_K$.

3.3.2 AR-GAT

The relational graph attention networks (Wang et al., 2020a) extended original graph attention networks with additional heads to leverage the type information of edges. However, relational graph attention networks are not sufficiently compatible with encoding the dual relational graph. On the one hand, original syntactic and semantic relations that are initially generated may be wrong. On the other hand, in the construction process of the dual relational graph, reshaping and pruning may further lead to the propagation of errors originating from the parser. Thus, the relation heads are not sufficient to accurately control information flow from neighborhood nodes.

To overcome the above problems, we propose introducing additional contextual information from word nodes to control information flow more accurately. The performance of factorization machines has been proven in many tasks (Guo et al., 2017). Inspired by factorization machines, we employ an

inner product unit to combine contextual and type information.

Formally, the initial relation embedding matrix is defined as $\mathbf{W}_t \in \mathbb{R}^{N_t \times F}$, N_t is the number of relation type. For the node i at the layer l+1, the computation of multi-head augmented relational graph attention networks can be defined as follows:

$$\mathbf{h}_{rel_i}^{l+1} = \|_{m=1}^{M} \sigma \left(\sum_{j \in \mathcal{N}_i} \beta_{ij}^{lm} \mathbf{W}_m^l \mathbf{h}_j^l \right) \quad (3)$$

$$R_{ij}^m = (\mathbf{W}_{m1} f_2(\tau(e_{ij}), \mathbf{W}_t) + b_{m1}) \odot (\mathbf{W}_{m2}(\mathbf{h}_i \| \mathbf{h}_j) + b_{m2}) \quad (4)$$

$$g_{ij}^{lm} = \sigma(\text{relu}(R_{ij}^m \mathbf{W}_{m3} + b_{m3}) \mathbf{W}_{m4} + b_{m4}) \quad (5)$$

$$\beta_{ij}^{lm} = \frac{\exp\left(g_{ij}^{lm}\right)}{\sum_{j \in \mathcal{N}_i} \exp\left(g_{ij}^{lm}\right)} \quad (6)$$

where $\mathbf{h}_{rel_i}^{l+1}$ represents the augmented relational attention head of node i at layer l+1, M is the number of heads, $f_2(\tau(e_{ij}), \mathbf{W}_t)$ is a mapping function mapping edge e_{ij} into the corresponding relation embedding according to relation embedding matrix $\mathbf{W}_t$. The final representation of node i is computed by:

$$o_i^{l+1} = \mathbf{h}_{att_i}^{l+1} \| \mathbf{h}_{rel_i}^{l+1} \quad (7)$$

$$\mathbf{h}_i^{l+1} = \text{relu}\left(\mathbf{W}_{l+1} o_i^{l+1} + b_{l+1}\right) \quad (8)$$

3.4 Event Detector

We use BERT (Devlin et al., 2019) to obtain the word embedding of graph nodes, and the embedding of word x_i is defined as $\mathbf{h}_i^0$. We use $\mathbf{h}_t^0$ and $\mathbf{h}_v^0$ to denote the initial embedding of the trigger candidate node and the predicate verb node, respectively. Then we apply AR-GAT to encode two subgraphs of dual relational graph respectively and obtain their root representation $\mathbf{h}_t^l$ and $\mathbf{h}_v^l$. The $\mathbf{h}_t^l$ and $\mathbf{h}_v^l$ aggregate the syntactic and semantic information respectively.

To effectively exchange relevant features between these two types of information, we employ a mutual Biaffine transformation (Li et al., 2021b). The interaction process is:

$$\mathbf{h}_t^{l\prime} = \text{softmax}(\mathbf{h}_t^l \mathbf{W}_1 (\mathbf{h}_v^l)^{\mathrm{T}}) \mathbf{h}_v^l \quad (9)$$

$$\mathbf{h}_v^{l\prime} = \text{softmax}(\mathbf{h}_v^l \mathbf{W}_2 (\mathbf{h}_t^l)^{\mathrm{T}}) \mathbf{h}_t^l \quad (10)$$

where $\mathbf{W}_1$ and $\mathbf{W}_2$ are parameters.

Finally, we concatenate syntactic root $\mathbf{h}_t^{l\prime}$ and semantic root $\mathbf{h}_v^{l\prime}$ to obtain final feature representation:

$$\mathbf{x} = \mathbf{h}_t^{l\prime} \| \mathbf{h}_v^{l\prime} \quad (11)$$

Then the final feature representation $\mathbf{x}$ is fed into a fully connected layer and adopt softmax layer to get a final event type probability distribution:

$$p(t) = \text{softmax}\left(\mathbf{W}_p \mathbf{x} + b_p\right) \quad (12)$$

4 Experiments

4.1 Datasets and Evaluation Metrics

We evaluate our proposed model on the widely used standard benchmark dataset ACE2005. ACE2005 consists of 33 event types and contains 599 documents with 4090 event instances. Following the previous works (Chen et al., 2015; Wang et al., 2020b), we use the same data split for train, dev and test set. We adopt Precision (P), Recall (R) and micro F1 score (F1), which are the standard metrics for event detection, as the evaluation criteria for method performance. The three metrics are defined as follows:

$$P = \frac{N_C}{N_P} \quad (13)$$

$$R = \frac{N_C}{N_D} \quad (14)$$

$$F1 = \frac{2PR}{P+R} \quad (15)$$

where N_C, N_P, and N_D are the number of correctly predicted events, all predicted events, and all events in the dataset, respectively.

4.2 Parameter Settings

Our implementation [1] is based on the bert-base-uncased model, whose layer number is 12, hidden size is 768, and attention head number is 12. For the "Our w/o BERT" setting, we use Glove (Pennington et al., 2014) and BiLSTM to obtain word embedding. The dimension of both word embedding and relational embedding is 300. the hidden state size of AR-GAT and BiLSTM is set to 200. The number of attention heads is 6. The max sentence length is set to 128 by cutting longer sentences and padding shorter ones. We employ the Biaffine Parser (Dozat and Manning, 2017) and SRL BERT (Shi and Lin, 2019) for dependency

[1] Code is available at https://github.com/Macvh/DualGAT, which leverage the HuggingFace's Transformers library for loading pre-trained models.

parsing and semantic role labeling. The Adam algorithm (Kingma and Ba, 2015) is used to optimize model parameters and the dropout rate is set to 0.3 to avoid over-fitting. The max epoch is 30. In addition, we report the average performance of 5 trials with different random seeds for each experiment.

4.3 Baselines

We select various representative methods as baselines, including:

Syntactic based Methods: (1) **GCN-ED** uses GCN with entity mention based pooling method for event detection (Nguyen and Grishman, 2018). (2) **SA-GRCN** introduces a self-attention mechanism for better modeling word dependencies (Liu et al., 2021). (3) **EE-GCN** introduces additional typed dependency label information into GCNs (Cui et al., 2020). (4) **GatedGCN** proposes to filter noisy information via a gating mechanism (Lai et al., 2020).

External Knowledge based Methods: (1) **PLMEE** proposes a method to enlarge the scale of labeled data by editing prototypes and an evaluation method to screen out generated data (Yang et al., 2019). (2) **DNR** uses additional external knowledge to link predefined event types to each sentence to improve performance (Liao et al., 2021). (3) **SS-JDN** introduces statistical features to cooperate with the contextual features for event detection (Li et al., 2021a).

4.4 Overall Results

Experimental results are shown in Table 1, where "Our w/o BERT" denotes replacing BERT with BiLSTM in our model. The highest values are shown in bold. We can observe that our proposed DualGAT outperforms all the baselines on the ACE2005 dataset in terms of three metrics from Table 1. Particularly, DualGAT outperforms the syntactic-based methods by a large margin. It is worth mentioning that the performance of Our w/o BERT is even better than that of GatedGCN which used BERT. Compared with the next best method among all compared methods, DualGAT improves the F1, Precision, and Recall by 0.9%, 0.5%, and 1.3%, respectively. It proves the effectiveness of our proposed model for ED tasks. DualGAT fully considers the complementary of syntactic and semantic information and effectively captures the trigger-relevant information. Compared with existing syntactic-based methods, the advantages of DualGAT are more apparent. DualGAT is able to more effectively leverage the internal critical information of sentences for ED tasks. Besides, DualGAT is even better than three methods that use external knowledge. It indicates that existing ED methods do not fully exploit the internal information of sentences of original data that are worth further exploiting.

Type	Method	P	R	F1	BERT
Syn.	GCN-ED	77.9	68.8	73.1	-
	EE-GCN	76.7	78.6	77.6	-
	SA-GRCN	78.6	77.4	78.0	-
	GatedGCN	78.8	76.3	77.6	√
Exter.	SS-JDN	80.3	78.8	79.5	√
	PLMEE	81.0	80.4	80.7	√
	DNR	81.2	82.4	81.8	√
Ours	Our w/o BERT	79.1	80.8	80.0	-
	DualGAT	**81.7**	**83.7**	**82.7**	√

Table 1: Overall performance on ACE2005 dataset (%). "Syn." indicates syntactic dependency relations are used, "Exter." indicates external knowledge and resources are used.

Method	P	R	F1
DualGAT w/o reshape	78.1	75.3	76.7
DualGAT w/o syntactic	76.2	79.3	77.7
DualGAT w/o semantic	78.9	81.3	80.1
DualGAT w/o AR-GAT	80.3	82.1	81.2
DualGAT	**81.7**	**83.7**	**82.7**

Table 2: Experimental results of ablation study on ACE2005 dataset (%).

4.5 Ablation Study

To assess the effect of the dual relational graph and augmented relational graph attention networks, we further conduct several ablation studies. We design four variants of the proposed model:

DualGAT w/o reshape: to study whether the dual relational graph contributes to the performance improvement, we use the ordinary syntactic dependency tree to replace the dual relational graph.

DualGAT w/o syntactic: to prove the effectiveness of the syntactic relational subgraph, we remove the syntactic relational subgraph and only use the semantic relational subgraph.

DualGAT w/o semantic: to prove the effectiveness of the semantic relational subgraph, we remove the semantic relational subgraph and only use the syntactic relational subgraph.

DualGAT w/o AR-GAT: to prove the effectiveness of the augmented relational graph attention

networks, we use relational graph attention networks to encode the dual relational graph.

The results of the ablation study are shown in Table 2. From the results, we can observe that:

(1) DualGAT w/o semantic outperforms DualGAT w/o reshape by a large margin. It indicates that pruning and reshaping original syntactic relations are useful for event detection. The syntactic relational subgraph converges all trigger-relevant relations to the trigger candidates, which reduces the interference of noisy relations. It also indicates that a small part of dependency is task-aware, and encoding the entire dependency tree is unnecessary for event detection.

(2) In terms of performance degradation compared to DualGAT, the F1 score of DualGAT without syntactic relational subgraph drops more seriously than that without semantic relational subgraph. It indicates that syntactic relations converged on trigger candidates are more necessary in DualGAT. The reason may be that the syntactic relational subgraph establishes more effective correlations among trigger candidates and other words in a sentence.

(3) The DualGAT is improved by using the augmented relational graph attention networks and achieves 1.5% improvements in the F1 score. It indicates that the introduction of contextual information effectively captures key information between words. The dependency parsers do not always parse sentences correctly, which causes much loss of information. Thus, we use additional contextual information to get more accurate attention weights.

4.6 The Effect of Multiple Event Recognition

To verify the effectiveness of the dual relational graph customized for every trigger candidate, we study the performance of the proposed DualGAT for multiple event recognition. Following (Xie et al., 2021), we divide the test set into the "1/1" and "1/N" subsets and perform evaluation on the two subsets separately, where one sentence has only one event in the 1/1 subset but has multiple events in the 1/N subset. Moreover, one sentence may contain multiple event types in the 1/N subset. Figure 4 illustrates the performance (F1 score) of DualGAT w/o reshape and DualGAT.

As shown in Figure 4, our proposed DualGAT significantly outperforms DualGAT w/o reshape in three situations. DualGAT improves upon the DualGAT w/o reshape by 6.1% and 9.9% in 1/1 data split and 1/N data split, respectively. The multi-event scenario in a sentence confuses the event detection methods, resulting in poor performance. However, DualGAT improves the performance by using the dual relational graph. Since each trigger candidate has its particular dual relational graph, the dual relational graph can reduce the interference of irrelevant nodes and relations. The additional semantic relations further capture sentence-level information. Thus, our proposed DualGAT can alleviate the multi-event problem to a certain extent. The experimental results demonstrate that the dual relational graph is effective for the task of multiple event recognition.

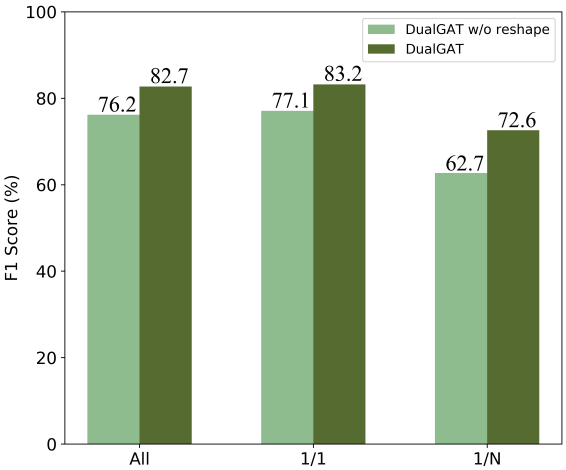

Figure 4: Performance of our method with ordinary syntactic dependency tree ("DualGAT w/o reshape") and the proposed dual relational graph ("DualGAT").

4.7 Semantic Relations Improve Robustness

We further study whether semantic relations can make the dual relational graphs less vulnerable to dependency parsing errors. Therefore, we conduct experiments based on two widely used dependency parsers: Stanford Parser (Chen and Manning, 2014) and Biaffine Parser (Dozat and Manning, 2017). The performance of the two parsers is shown in Table 3, measured by two widely used metrics UAS and LAS, of which higher is better. We use each of these two dependency parsers to construct the dual relational graph and evaluate the final performance of the proposed method.

The experimental results of DualGAT with different dependency parsers are shown in Table 4. From Table 4, we can find that the DualGAT with Biaffine parser achieves better performance in event detection since the dependency parsing performance of the Biaffine parser is better than Stanford Parser.

Parser	UAS	LAS
Stanford	94.10	91.49
Biaffine	**95.74**	**94.08**

Table 3: The performance of two dependency parsers. The results are from (Chen and Manning, 2014) and (Dozat and Manning, 2017) respectively.

Method	P	R	F1
DualGAT (Sta) w/o sem	76.1	78.6	77.3
DualGAT (Bia) w/o sem	78.9	81.3	80.1
DualGAT (Sta)	81.5	82.3	81.9
DualGAT (Bia)	**81.7**	**83.7**	**82.7**

Table 4: The performance of DualGAT using different parsers on ACE2005 dataset (%).

In the case of only syntactic relations, DualGAT with Biaffine parser improves upon the DualGAT with Stanford parser by 2.8%, 2.8% and 2.7% in terms of F1 score, Precision and Recall. However, compared with DualGAT (Bia), DualGAT (Sta) declines by only 0.8%, 0.2% and 1.4% in F1 score, Precision and Recall. It illustrates that semantic relations can resist interference with syntactic parser performance and improve the robustness of the dual relational graph. Semantic relations have complementarity with syntactic relations and sustain the performance of DualGAT in the case of syntactic relationship failure. Besides, it also implies that our proposed DualGAT can benefit from the advances in syntactic parsing and semantic parsing techniques.

5 Conclusion and Future Work

In this paper, we propose a simple yet effective model named DualGAT (Dual Relational Graph Attention Networks) to address the disadvantages of syntactic-based methods for event detection tasks. To facilitate the accurate capture of key information from different perspectives in a sentence, we devise a dual relational graph that aggregates syntactic and semantic relations to key nodes in the graph. To efficiently encode the dual relational graph, we propose augmented relational graph attention networks that introduce contextual information to compute more robust attention weights. Experimental results show that our proposed method achieves state-of-the-art performance.

We intend to explore several aspects of our work further in the future. First, we would improve the way semantic information is introduced. Second, we would develop a more effective method for fusing syntactic and semantic information. Third, we would explore the effect of augmented relational graph attention networks in other tasks.

Acknowledgements

We want to express gratitude to the anonymous reviewers for their hard work and kind comments. This work was supported by the National Social Science Fund of China under Grant No. 20BTQ068.

References

David Ahn. 2006. The stages of event extraction. In *Proceedings of the Workshop on Annotating and Reasoning about Time and Events*, pages 1–8.

Claudio Delli Bovi, Luca Telesca, and Roberto Navigli. 2015. Large-scale information extraction from textual definitions through deep syntactic and semantic analysis. *Transactions of the Association for Computational Linguistics*, 3:529–543.

Pengfei Cao, Xinyu Zuo, Yubo Chen, Kang Liu, Jun Zhao, Yuguang Chen, and Weihua Peng. 2021. Knowledge-enriched event causality identification via latent structure induction networks. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing*, pages 4862–4872.

Yee Seng Chan and Dan Roth. 2011. Exploiting syntactico-semantic structures for relation extraction. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics*, pages 551–560.

Danqi Chen and Christopher D. Manning. 2014. A fast and accurate dependency parser using neural networks. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing*, pages 740–750.

Yubo Chen, Liheng Xu, Kang Liu, Daojian Zeng, and Jun Zhao. 2015. Event extraction via dynamic multi-pooling convolutional neural networks. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing of the Asian Federation of Natural Language Processing*, pages 167–176.

Shiyao Cui, Bowen Yu, Tingwen Liu, Zhenyu Zhang, Xuebin Wang, and Jinqiao Shi. 2020. Edge-enhanced graph convolution networks for event detection with syntactic relation. In *Findings of the Association for Computational Linguistics*, pages 2329–2339.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 4171–4186.

Timothy Dozat and Christopher D. Manning. 2017. Deep biaffine attention for neural dependency parsing. In *Proceedings of the 5th International Conference on Learning Representations*, pages 1–8.

Erwin Filtz, María Navas-Loro, Cristiana Santos, Axel Polleres, and Sabrina Kirrane. 2020. Events matter: Extraction of events from court decisions. In *Proceedings of Legal Knowledge and Information Systems*, pages 33–42.

Huifeng Guo, Ruiming Tang, Yunming Ye, Zhenguo Li, and Xiuqiang He. 2017. Deepfm: A factorization-machine based neural network for CTR prediction. In *Proceedings of the Twenty-Sixth International Joint Conference on Artificial Intelligence*, pages 1725–1731.

Shexia He, Zuchao Li, Hai Zhao, and Hongxiao Bai. 2018. Syntax for semantic role labeling, to be, or not to be. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics*, pages 2061–2071.

Yu Hong, Jianfeng Zhang, Bin Ma, Jian-Min Yao, Guodong Zhou, and Qiaoming Zhu. 2011. Using cross-entity inference to improve event extraction. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, pages 1127–1136.

Diederik P. Kingma and Jimmy Ba. 2015. Adam: A method for stochastic optimization. In *Proceedings of the 3rd International Conference on Learning Representations*, pages 1–15.

Viet Dac Lai, Tuan Ngo Nguyen, and Thien Huu Nguyen. 2020. Event detection: Gate diversity and syntactic importance scores for graph convolution neural networks. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing*, pages 5405–5411.

Qi Li, Heng Ji, and Liang Huang. 2013. Joint event extraction via structured prediction with global features. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics*, pages 73–82.

Rui Li, Wenlin Zhao, Cheng Yang, and Sen Su. 2021a. Treasures outside contexts: Improving event detection via global statistics. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 2625–2635.

Ruifan Li, Hao Chen, Fangxiang Feng, Zhanyu Ma, Xiaojie Wang, and Eduard H. Hovy. 2021b. Dual graph convolutional networks for aspect-based sentiment analysis. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing*, pages 6319–6329.

Xin Liang, Dawei Cheng, Fangzhou Yang, Yifeng Luo, Weining Qian, and Aoying Zhou. 2020. F-HMTC: detecting financial events for investment decisions based on neural hierarchical multi-label text classification. In *Proceedings of the Twenty-Ninth International Joint Conference on Artificial Intelligence*, pages 4490–4496.

Jinzhi Liao, Xiang Zhao, Xinyi Li, Lingling Zhang, and Jiuyang Tang. 2021. Learning discriminative neural representations for event detection. In *Proceedings of the 44th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 644–653.

Anan Liu, Ning Xu, and Haozhe Liu. 2021. Self-attention graph residual convolutional networks for event detection with dependency relations. In *Findings of the Association for Computational Linguistics*, pages 302–311.

Jian Liu, Yubo Chen, and Kang Liu. 2019. Exploiting the ground-truth: An adversarial imitation based knowledge distillation approach for event detection. In *Proceedings of the AAAI Conference on Artificial Intelligence*, pages 6754–6761.

Jing Liu, Songzheng Zhao, and Gang Wang. 2018a. Ssel-ade: a semi-supervised ensemble learning framework for extracting adverse drug events from social media. *Artificial intelligence in medicine*, 84:34–49.

Xiao Liu, Zhunchen Luo, and Heyan Huang. 2018b. Jointly multiple events extraction via attention-based graph information aggregation. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 1247–1256.

Yaojie Lu, Hongyu Lin, Xianpei Han, and Le Sun. 2019. Distilling discrimination and generalization knowledge for event detection via delta-representation learning. In *Proceedings of the 57th Conference of the Association for Computational Linguistics*, pages 4366–4376.

Jianwei Lv, Zequn Zhang, Li Jin, Shuchao Li, Xiaoyu Li, Guangluan Xu, and Xian Sun. 2021. HGEED: hierarchical graph enhanced event detection. *Neurocomputing*, 453:141–150.

Thien Huu Nguyen, Kyunghyun Cho, and Ralph Grishman. 2016. Joint event extraction via recurrent neural networks. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 300–309.

Thien Huu Nguyen and Ralph Grishman. 2018. Graph convolutional networks with argument-aware pooling for event detection. In *Proceedings of the Thirty-Second AAAI Conference on Artificial Intelligence*, pages 5900–5907.

Jeffrey Pennington, Richard Socher, and Christopher D. Manning. 2014. Glove: Global vectors for word representation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing*, pages 1532–1543.

Franco Scarselli, Marco Gori, Ah Chung Tsoi, Markus Hagenbuchner, and Gabriele Monfardini. 2009. The graph neural network model. *IEEE transactions on neural networks*, 20(1):61–80.

Michael Sejr Schlichtkrull, Thomas N. Kipf, Peter Bloem, Rianne van den Berg, Ivan Titov, and Max Welling. 2018. Modeling relational data with graph convolutional networks. In *Proceedings of the 15th European Semantic Web Conference*, pages 593–607.

Peng Shi and Jimmy Lin. 2019. Simple bert models for relation extraction and semantic role labeling. *ArXiv*, abs/1904.05255.

Meihan Tong, Bin Xu, Shuai Wang, Yixin Cao, Lei Hou, Juanzi Li, and Jun Xie. 2020. Improving event detection via open-domain trigger knowledge. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 5887–5897.

Petar Velickovic, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Liò, and Yoshua Bengio. 2018. Graph attention networks. In *Proceedings of the 6th International Conference on Learning Representations*.

Kai Wang, Weizhou Shen, Yunyi Yang, Xiaojun Quan, and Rui Wang. 2020a. Relational graph attention network for aspect-based sentiment analysis. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 3229–3238.

Xiaozhi Wang, Xu Han, Zhiyuan Liu, Maosong Sun, and Peng Li. 2019. Adversarial training for weakly supervised event detection. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 998–1008.

Xiaozhi Wang, Ziqi Wang, Xu Han, Wangyi Jiang, Rong Han, Zhiyuan Liu, Juanzi Li, Peng Li, Yankai Lin, and Jie Zhou. 2020b. MAVEN: A massive general domain event detection dataset. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing*, pages 1652–1671.

Qiang Wei, Zongcheng Ji, Zhiheng Li, Jingcheng Du, Jingqi Wang, Jun Xu, Yang Xiang, Firat Tiryaki, Stephen Wu, Yaoyun Zhang, Cui Tao, and Hua Xu. 2020. A study of deep learning approaches for medication and adverse drug event extraction from clinical text. *Journal of the American Medical Informatics Association*, 27(1):13–21.

Jianye Xie, Haotong Sun, Junsheng Zhou, Weiguang Qu, and Xinyu Dai. 2021. Event detection as graph parsing. In *Findings of the Association for Computational Linguistics*, pages 1630–1640.

Haoran Yan, Xiaolong Jin, Xiangbin Meng, Jiafeng Guo, and Xueqi Cheng. 2019. Event detection with multi-order graph convolution and aggregated attention. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing*, pages 5765–5769.

Hang Yang, Yubo Chen, Kang Liu, Yang Xiao, and Jun Zhao. 2018. DCFEE: A document-level chinese financial event extraction system based on automatically labeled training data. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics-System Demonstrations*, pages 50–55.

Sen Yang, Dawei Feng, Linbo Qiao, Zhigang Kan, and Dongsheng Li. 2019. Exploring pre-trained language models for event extraction and generation. In *Proceedings of the 57th Conference of the Association for Computational Linguistics*, pages 5284–5294.

Ying Zeng, Yansong Feng, Rong Ma, Zheng Wang, Rui Yan, Chongde Shi, and Dongyan Zhao. 2018. Scale up event extraction learning via automatic training data generation. In *Proceedings of the AAAI Conference on Artificial Intelligence*, pages 6045–6052.

Yuhao Zhang, Peng Qi, and Christopher D. Manning. 2018. Graph convolution over pruned dependency trees improves relation extraction. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 2205–2215.

Shun Zheng, Wei Cao, Wei Xu, and Jiang Bian. 2019. Doc2edag: An end-to-end document-level framework for chinese financial event extraction. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing*, pages 337–346.

A Multi-Format Transfer Learning Model for Event Argument Extraction via Variational Information Bottleneck

Jie Zhou[1†*], Qi Zhang[2†], Qin Chen[2], Qi Zhang[1], Liang He[2], and Xuanjing Huang[1]

[1]School of Computer Science, Fudan Univerisity, Shanghai, China
[2]School of Computer Science and Technology, East China Normal University, Shanghai, China
{jie_zhou, qz, xjhuang}@fudan.edu.cn
qzhang@stu.ecnu.edu.cn; {qchen, lhe}@cs.ecnu.edu.cn

Abstract

Event argument extraction (EAE) aims to extract arguments with given roles from texts, which have been widely studied in natural language processing. Most previous works have achieved good performance in specific EAE datasets with dedicated neural architectures. Whereas, these architectures are usually difficult to adapt to new datasets/scenarios with various annotation schemas or formats. Furthermore, they rely on large-scale labeled data for training, which is unavailable due to the high labelling cost in most cases. In this paper, we propose a multi-format transfer learning model with variational information bottleneck, which makes use of the information especially the common knowledge in existing datasets for EAE in new datasets. Specifically, we introduce a shared-specific prompt framework to learn both format-shared and format-specific knowledge from datasets with different formats. In order to further absorb the common knowledge for EAE and eliminate the irrelevant noise, we integrate variational information bottleneck into our architecture to refine the shared representation. We conduct extensive experiments on three benchmark datasets, and obtain new state-of-the-art performance on EAE.

1 Introduction

Event Extraction (EE) has received widespread attention in recent years, which aims to obtain structured information (e.g., trigger, event types, arguments, and argument roles) from large unstructured text corpora (Lu et al., 2021; Zhang et al., 2022). Event argument extraction (EAE) plays a crucial role in EE. Recently, deep learning-based models have obtained tremendous success in this task. However, these methods rely on a large-scale labeled dataset for training, which is time-consuming and labor-intensive due to the complexity of event extraction.

ACE2005 Example	Conflict.Attack	
And to the south , **British**[Place] **forces**[Target] continue their **attack**[trigger] on targets around Basra .	trigger	attack
	Target	forces
	Place	Basra
	Attacker	-
	Victim	-
	Instrument	-

WIKIEVENTS Example	Conflict.Attack.DetonateExplode	
For example , Ms . Davis has identified a man whose photo matches that of a \" John Doe # 2 \" sought immediately after the Murrah **Building**[Target] **attack**[trigger]. He appears to be a Palestinian by the name of Hussain Hashem Al Hussaini ...	trigger	attack
	Target	Murrah Building
	Place	-
	Attacker	-
	ExplosiveDevice	-
	Instrument	-

Figure 1: An example with a different format.

In this paper, we aim to answer the question "Can we transfer the knowledge from the existing complex event extraction datasets with different formats?". There are several event extraction datasets, such as ACE 2005 (Doddington et al., 2004), RAMS (Ebner et al., 2020), and WikiEvents (Li et al., 2021). These datasets contain abundant event types and semantic roles that may possess overlap knowledge and help to improve the performance of new datasets or low-resource extraction. As shown in Figure 1, both ACE2005 and WikiEvents datasets contain the same "attack" event type with inconsistent names. Additionally, some shared argument roles (e.g., "Target", "Attacker", "Place", and "Instrument") are labeled in both two datasets. All this information shows that the event knowledge can be transferred between two datasets.

However, the transfer between different event argument extraction is a challenging task. **(C1)** One challenge is that the formats of various datasets are inconsistent due to the complex structure of event records. Thus, it is hard to find a unified model to extract arguments with different formats. More specifically, 1) Two datasets may have dif-

*Corresponding authors; †Equal contribution.

Proceedings of the 29th International Conference on Computational Linguistics, pages 1990–2000
October 12–17, 2022.

ferent event types, which have various argument structures; 2) The same event type or argument type in two datasets may have different names. For example, the event names are “Conflict Attack" and “Conflict Attack Detonate Explode" in ACE2005 and WikiEvents respectively (Figure 1); 3) The argument roles set of the same event type may be different in various datasets. For instance, the argument role “Victim" and “ExplosiveDevice" for event “Attack" only appear in ACE2005 and WikiEvents, respectively (Figure 1). **(C2)** The another challenge is that the annotation among different datasets may exist a gap, which brings noise for transfer learning. Two datasets may have significant semantic differences, as they may belong to different domains. In addition, the annotation guidelines may be contradictory among various datasets. Our experiments also show that merging two datasets simply may reduce the performance.

Previous works mainly regard the argument extraction as a sequence labeling, which can not transfer to new event argument types (Yang et al., 2018). Then, a machine reading comprehension problem (MRC) based model is proposed to extract the arguments using natural questions (Liu et al., 2020; Du and Cardie, 2020). Recently, prompt-learning (Schick and Schütze, 2020; Liu et al., 2021b) based models (Ma et al., 2022; Chen et al., 2020) and generation-based models (Chen et al., 2020; Du et al., 2021; Li et al., 2021) are utilized for event argument extraction. These studies inspire us to design a unified model that can extract arguments with different formats for EAE. Moreover, some researches investigate cross-lingual event extraction (Subburathinam et al., 2019) and zero-shot event extraction (Chen et al., 2020; Feng et al., 2020), which are under zero-shot setting. In other words, these studies train on the source language or domain and transfer it to the target domain, where the target domain has no training data. Different from them, we train our model on both the source and target datasets with different formats where the format-shared knowledge is essential.

To deal with the above challenges, we propose a multi-format transfer learning model for EAE via information bottleneck, denoted as `UnifiedEAE`, which can leverage all event extraction datasets with heterogeneous formats. First, we adopt a Shared-Specific Prompt (`SSP`) framework to capture format-shared and format-specific knowledge to extract arguments with different formats. Then, to better capture the format-shared representation, we incorporate the variational information bottleneck (VIB) into the format-shared model (`SharedVIB`). VIB has been widely used to forget the irrelevant information and retain the vital information for prediction (Li and Eisner, 2019; Tishby et al., 2000). We leverage it to enhance the model to learn the format-shared knowledge. We conduct a series of experiments on three publicly available datasets and obtain new state-of-the-art performance. Our `UnifiedEAE` can also improve the performance of low-resource EAE effectively. Furthermore, the results show that our model can capture the format-shared knowledge and forget the noise among various datasets.

In summary, the main contributions of this paper are summarized as follows.

- We design a unified architecture that can learn both the format-shared and format-specific knowledge from various EE datasets with heterogeneous formats.
- The information bottleneck technology is utilized to enhance the model to learn the format-shared knowledge among different datasets by eliminating the irrelevant information and reserving the format-shared knowledge.
- Extensive experiments on three datasets show the great advantages of our model. Also, our model performs well on low-resource event argument extraction.

2 Related Work

2.1 Event Argument Extraction

Event extraction can be split into two subtasks, event identification and event argument extraction (EAE) (Zhang et al., 2020; Chen et al., 2015; Lin et al., 2022). We focus on the EAE task, which aims to extract the arguments based on the given event type and trigger (Wei et al., 2021; Ma et al., 2022). Wei et al. (2021) added constraints with each argument role to take the interaction into account. Data augmentation is adopted for event argument extraction (Liu et al., 2021a). To avoid the error propagation and learn the relationships among the subtasks, end to end model performs two subtasks jointly (Zhang et al., 2019; Wadden et al., 2019; Li et al., 2021). Several studies regard event argument extraction as a machine reading comprehension problem (MRC), which extracts the arguments

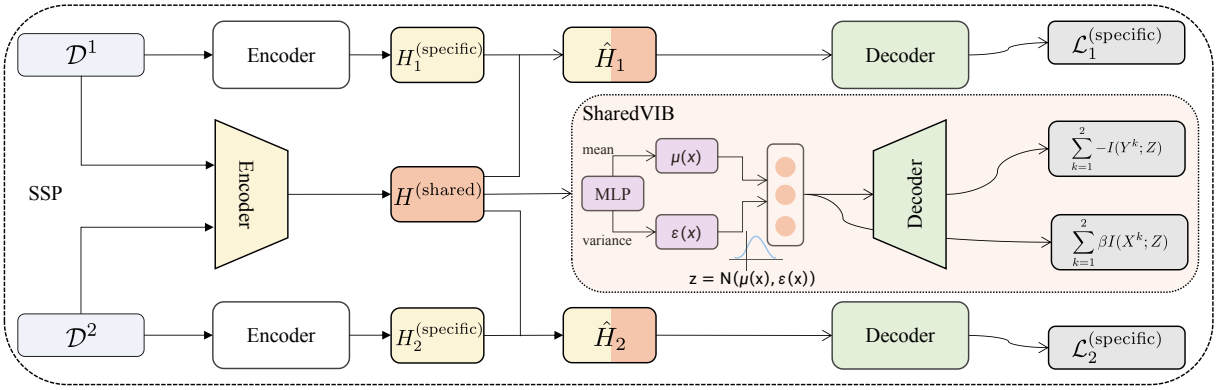

Figure 2: The framework of our `UnifiedEAE` model. To learn both the format-shared and format-specific representations (i.e., $\hat{H}_1$ and $\hat{H}_2$), we introduce a shared-specific prompt (`SSP`) model (white background). Then, we design a `SharedVIB` module (pink background) to better capture the format-shared representation (e.g., $H^{(\text{shared})}$) by forgetting format-specific information ($\sum_{k=1}^{2} \beta I(X^k; Z)$) and retaining format-shared knowledge ($\sum_{k=1}^{2} -I(Y^k; Z)$) via information bottleneck.

based on natural questions (Liu et al., 2020; Du and Cardie, 2020). Recently, prompt-learning (Schick and Schütze, 2020; Liu et al., 2021b) based models (Ma et al., 2022; Chen et al., 2020) and generation-based models (Chen et al., 2020; Du et al., 2021; Li et al., 2021) are utilized for event argument extraction. In this paper, we aim to transfer the knowledge of the existing event extraction datasets to the target dataset, which is not well studied since the complexity of this task.

2.2 Transfer Learning for NLP

To reduce the requirements of labeled data, transfer learning has attached great attention in the field of natural language processing (Liu et al., 2017; Ruder et al., 2019; Raffel et al., 2020; Zhou et al., 2020). Liu et al. (2017) proposed an adversarial multi-task learning framework to learn the shared and private representation. Cross-lingual event extraction aims to transfer the knowledge from the source language to the target language (Subburathinam et al., 2019). Zero-shot transfer learning is also explored on semantic role labeling (SRL) (Peng et al., 2016), event extraction (Chen et al., 2020; Feng et al., 2020), and abstract meaning representation (AMR) (Huang et al., 2018). Different from them, we focus on transfer learning among event argument extraction datasets with various complex formats where both the format-shared knowledge and format-specific knowledge are important.

2.3 Information Bottleneck

Recently, information bottleneck (IB) has been applied in NLP tasks, such as word cluster (Pereira et al., 1994), dependent parsing (Mahabadi et al., 2021), summarization (West et al., 2019), interpretability (Zhou et al., 2021). Li and Eisner (2019) use IB for compressing the hidden representations of words by removing the task-irrelevant information. Sun et al. (2021) adopted the IB principle for graph structure learning. Variational IB (VIB) is used as a regularization technique to improve the fine-tuning of pre-training language models in low-resource scenarios (Mahabadi et al., 2021). In this paper, we attempt to use VIB to constraint model to learn format-shared information for event argument extraction.

3 Methodology

To transfer the knowledge among the datasets with different formats, we propose a `UnifiedEAE` model for event argument extraction task (Figure 2). `UnifiedEAE` is based on a shared-specific prompt (`SSP`) architecture, which learns both the format-shared and format-specific knowledge from diverse datasets with multiple formats. Then, to enhance the model to learn the format-shared knowledge, we integrate variational information bottleneck into the format-shared model (`SharedVIB`) by removing the format irrelevant information and retaining format invariable knowledge.

Formally, given two event argument extraction datasets, denoted by $\mathcal{D}^1 = \{(X_i^1, Y_i^1)\}_{i=1}^{|\mathcal{D}^1|}$ and $\mathcal{D}^2 = \{(X_i^2, Y_i^2)\}_{i=1}^{|\mathcal{D}^2|}$ where $|\mathcal{D}^1|$ and $|\mathcal{D}^2|$ are the number of samples in dataset $\mathcal{D}^1$ and $\mathcal{D}^2$. For each sample $(X, Y) \in \mathcal{D}^1 or \mathcal{D}^2$, the input $X = \{s, e, t, R\}$ contains the sentence s, event

type e, and trigger word t, R denotes the set of event-specific role types, we aims to extract a set of span Y. For the i-th span in Y, $\text{span}_i^{\text{start}}$ and $\text{span}_i^{\text{end}}$ are the start and end indices of the span.

3.1 Shared-Specific Prompt

The shared-specific prompt (SSP) architecture aims to learn both format-shared and format-specific knowledge for EAE. This framework consists of three event argument extractors: two format-specific and one format-shared extractor, which are used to learn the format-specific and format-shared knowledge. We adopt a prompt-based model as the basic extractor to predict multi-format arguments.

Basic Prompt-based Extractor. Following the Ma et al. (2022), we use a BART (Lewis et al., 2020) based prompt model as an event argument extractor. This model consists of an encoder and decoder. The encoder is used to learn the event-aware sentence representation. Then we adopt a decoder model to extract all the argument spans jointly via a prompt template.

Encoder. To consider the position information of event, we insert special token "<t>" and "</t>" before and after the trigger t in the sentence s. Then we input it into BART to obtain the event-aware sentence representation H,

$$\begin{aligned} H_{\text{encoder}} &= \text{BART}_{\text{Encoder}}(s), \\ H &= \text{BART}_{\text{Decoder}}(s, H_{\text{encoder}}), \end{aligned} \tag{1}$$

Decoder. In the decoder, we use a prompt with slots to extract the argument roles at the same time. We use manual template from Li et al. (2021). For example, the prompt is "Person married Person at Place (and Place)" for event type "Life.Marry" in ACE2005. We aim to predict the argument spans for the four argument role slots. We input the prompt p to the BART decoder to obtain the prompt representation.

$$H_p = \text{BART}_{\text{decoder}}(p, H_{\text{encoder}})$$

For the i-th role slot in the prompt, we use the mean pooling of the corresponding tokens' representation from H_p as the role representation r_i. Then, we extract the argument span for role r_i by predicting the start and end index in the text.

$$\begin{aligned} p_i^{(\text{start})} &= \text{Softmax}(r_i w^{(\text{start})} H) \\ p_i^{(\text{end})} &= \text{Softmax}(r_i w^{(\text{end})} H) \end{aligned} \tag{2}$$

where $w^{(\text{start})}$ and $w^{(\text{end})}$ are the learnable parameters.

Finally, the cross-entropy between the predicted start/end probability $p_i^{(\text{start})}/p_i^{(\text{end})}$ and the ground truth,

$$\begin{aligned} \mathcal{L} = \sum_{X \in D} \sum_{i=1}^{|R|} &\text{CrossEntropy}(p_i^{(\text{start})}, \text{span}_i^{(\text{start})}) \\ &+\text{CrossEntropy}(p_i^{(\text{end})}, \text{span}_i^{(\text{end})}) \end{aligned} \tag{3}$$

where $\text{span}_i^{\text{start}}/\text{span}_i^{\text{end}}$ are the start and end index of the i-th argument role's span, $|R|$ is the length of roles set R in X.

For dataset $\mathcal{D}^1$ and $\mathcal{D}^2$, we use two independent prompt-based extractors to learn format-specific sentence representations $H_1^{(\text{specific})}$ and $H_2^{(\text{specific})}$ calculated by Equation 1. Then, the third extractor is adopted to learn the format-shared sentence representation $H^{(\text{shared})}$ among two datasets. To predict the event argument based on both format-specific and format-shared knowledge, we combine specific representation $H_k^{(\text{specific})}, k \in \{1, 2\}$ and shared representation $H^{(\text{shared})}$ via a gate mechanism (Hochreiter and Schmidhuber, 1997).

$$\begin{aligned} g_k &= \sigma\left(W_{g_k} \cdot \left[H_k^{(\text{specific})}, H^{(\text{shared})}\right] + b_{g_k}\right) \\ \hat{H}_k &= g_k * H_k^{(\text{specific})} + (1 - g_k) * H^{(\text{shared})} \end{aligned}$$

where W_{g_k}, b_{g_k} are the trainable parameters, σ is a sigmoid active function.

Then, we extract argument span by replacing H in Equation 2 with $\hat{H}_k$. In this way, we can predict the arguments based on both the format-specific and format-shared knowledge. Then, we obtain the format-specific loss $\mathcal{L}_{\text{SSP}} = \mathcal{L}_1^{(\text{specific})} + \mathcal{L}_2^{(\text{specific})}$, where $\mathcal{L}_1^{(\text{specific})}$ and $\mathcal{L}_2^{(\text{specific})}$ are the loss for dataset $\mathcal{D}^1$ and $\mathcal{D}^2$.

3.2 Shared Knowledge Learning via VIB

We hope the shared model in shared-specific prompt architecture to learn the format-shared knowledge while forgetting the format-specific knowledge. However, we do not add objectives to enhance the model to do this. Inspired by (Li and Eisner, 2019), we integrate variational information bottleneck (VIB) into our shared model (SharedVIB) to capture the format-shared knowledge while eliminating the format-specific information.

Particularly, the information bottleneck aims to learn a compressed representation Z, which maximizes mutual information with output Y and minimizes mutual information with input X. In this paper, we tend to let Z retain the information about $Y^k, k \in \{1, 2\}$ and remove the irrelevant information in $X^k, k \in \{1, 2\}$.

$$\sum_{k=1}^{2} \beta I(X^k; Z) - I(Y^k; Z)$$

The shared model performs both dataset $\mathcal{D}^1$ and $\mathcal{D}^2$ at the same time to learn the format-shared knowledge. Then, we let the model to forget the format-specific information in $X^k, k \in \{1, 2\}$ by minimizes mutual information between Z and $X^k, k \in \{1, 2\}$.

It is challenging to compute the mutual information $I(Y^k; Z)$ and $I(X^k; Z)$ directly. The same as (Li and Eisner, 2019), we use variational inference to compute a variational upper bound for $I(X^k; Z)$ as follow,

$$\overbrace{\mathop{\mathbb{E}}_{x}\left[\mathop{\mathbb{E}}_{z\sim p(z|x)}\left[\log \frac{p(z \mid x)}{q(z)}\right]\right]}^{\text{upper bound}} - \overbrace{\mathop{\mathbb{E}}_{x}\left[\mathop{\mathbb{E}}_{t\sim p(z|x)}\left[\log \frac{p(z \mid x)}{p(z)}\right]\right]}^{\mathrm{I}(X^k;Z)}$$
$$= \mathop{\mathbb{E}}_{x}\left[\mathrm{KL}\left(p(z) \| q(z)\right)\right] \geq 0$$

We optimize the upper bounder of $I(X^k; Z)$ to minimize it. We use reparameterzation method (Kingma and Welling, 2014) to sample Z from the latent distribution according to $p(z|x)$,

$$p(z \mid x) = \mathcal{N}\left(z \mid f^{\mu}(x), f^{\Sigma}(x)\right)$$

where $f^{\mu}(x) = H^{(shared)} \cdot W^{\mu}$ and $f^{\Sigma}(x) = H^{(shared)} \cdot W^{\Sigma}$ are the mean and variance of the latent Gaussian distribution, W^{μ} and W^{Σ} are the learnable parameters. Thus, we estimate $I(X^k; Z) = \mathrm{KL}\left(p(z \mid x) \| q(z)\right)$. For $q(z)$, we let it be a standard diagonal normal distribution.

For $I(Y^k; Z)$, we calculate the variational lower bound,

$$\overbrace{\mathop{\mathbb{E}}_{y,z\sim p}\left[\log \frac{p(y \mid z)}{p(y)}\right]}^{\mathrm{I}(Y^k;Z)} - \overbrace{\mathop{\mathbb{E}}_{y,z\sim p}\left[\log \frac{\psi(y \mid z)}{p(y)}\right]}^{\text{lower bound}}$$
$$= \mathop{\mathbb{E}}_{z\sim p}\left[\mathrm{KL}\left(p(y \mid z) \| \psi(y \mid z)\right)\right] \geq 0$$

Here, we use the decoder model in our shared-specific prompt (Section 3.1) as $\psi(y \mid z)$ by replacing H in Equation 2 with the sampled Z. Thus, the loss for optimizing $\psi(y \mid z)$ on $D^k, k \in \{1, 2\}$ is the same as Equation 3 by replacing the sampled Z with H Equation 2, denoted as $\mathcal{L}^{(shared)}$.

Thus, the loss function for `SharedVIB` is,

$$\mathcal{L}_{\text{SharedVIB}} = \sum_{k=1}^{2}\left(\mathcal{L}_k^{(\text{shared})} + \beta \sum_{X\in D^k} \mathrm{KL}\left(p(z \mid x) \| q(z)\right)\right)$$

Finally, the total loss for our `UnifiedEAE` is,

$$\mathcal{L} = \mathcal{L}_{\text{SSP}} + \mathcal{L}_{\text{SharedVIB}}$$

Table 1: Data statistics of RAMS and WikiEvents

Dataset	Split	#Doc	#Event	#Argument
RAMS	Train	3194	7329	17026
	Dev	399	924	2188
	Test	400	871	2023
WikiEvents	Train	206	3241	4542
	Dev	20	345	428
	Test	20	365	556

Table 2: Dataset statistics of ACE2005

Dataset	Split	#Sent	#Event	#Argument
ACE2005	Train	17,172	4202	4859
	Dev	923	450	605
	Test	832	403	576

4 Experimental Setups

4.1 Datasets

Our experiments are conducted on the three widely-used datasets for event argument extraction task: ACE2005 (Doddington et al., 2004), RAMS (Ebner et al., 2020) and WiKiEvents (Li et al., 2021). The ACE2005 dataset is a sentence-level extraction dataset that defines 33 different event types and 35 semantic roles. The split of training, validating, and testing sets is the same as (Wadden et al., 2019). The RAMS dataset focuses on a document-level argument extraction task, including 139 event types and 65 semantic roles. The WikiEvents dataset is another document-level argument extraction dataset, 246 documents are provided, with 50 event types and 59 argument roles. Our experiments are conducted under the annotations of their conventional arguments. The statistics of the datasets are listed in Table 1 and Table 2.

4.2 Evaluation Metric

Following Ma et al. (2022), we adopt two popular evaluation metrics. (1) Argument Identification F1

Table 3: The main results of event argument extraction. The best results are marked with **bold**.

	ACE2005		RAMS		WikiEvents		
	Args-I	Args-C	Args-I	Args-C	Args-I	Args-C	Head-C
FEAE	-	-	53.50	47.40	-	-	-
DocMRC	-	-	-	45.70	-	43.30	-
OneIE	65.90	59.20	-	-	-	-	-
EEQA	68.20	65.40	46.40	44.00	54.30	53.20	56.90
BART-Gen	59.60	55.00	50.90	44.90	47.50	41.70	44.20
EEQA-BART	69.60	67.70	49.40	46.30	60.30	57.10	61.40
PAIE	73.60	69.80	54.70	49.50	68.90	63.40	**66.50**
UnifiedEAE	**76.06**	**71.85**	**55.46**	**49.94**	**69.84**	**64.00**	66.30
UnifiedEAE (Zero-shot)	42.25	34.60	10.88	8.49	30.27	25.90	40.72
UnifiedEAE (Single)	72.77	68.82	53.32	48.29	68.31	63.40	66.16
UnifiedEAE (Multiple)	74.34	70.80	54.62	49.09	67.63	62.66	66.28

score (Arg-I): we consider an argument span is correctly identified when the predicted offset fits the golden-standard span. (2) Argument Classification F1 score (Arg-C): if both the span and the argument role type are matched with the golden standard, we consider the argument is correctly classified. For the WikiEvents dataset, we also additionally evaluate Argument Head F1 score (Head-C) that only considers the matching of the headword of an argument, the same as (Li et al., 2021).

4.3 Baselines

To investigate the effectiveness of our model, we compare our approach with the following state-of-the-art baseline models.

- ONEIE (Lin et al., 2020) is a joint neural model to extract the globally optimal IE result.
- BART-Gen (Li et al., 2021) proposes a document-level neural event argument extraction model by regarding this task as conditional generation based on event templates.
- EEQA (Du and Cardie, 2020) proposes an end-to-end model and translates event extraction task into a question answering (QA) task.
- FFAE (Wei et al., 2021) constructs the EAE task as a QA-based algorithm and uses the intra-event argument interaction to improve the performance.
- DocMRC (Liu et al., 2021a) utilizes implicit knowledge transfer and explicit data augmentation based on a QA-based method.
- EEQA-BART (Ma et al., 2022) replaces the BERT with BART for event extraction.
- PAIE (Ma et al., 2022) utilizes prompt tuning for extracting argument extraction so that it can take the best advantages of pre-trained language models.
- UnifiedEAE is our full model, which trains on two datasets and transfers to one of them. For UnifiedEAE (Zero-shot), we train our model on two datasets and test on the third dataset via format-shared extractor.
- UnifiedEAE (Multiple) trains on the merged dataset, which removes the SSP from our UnifiedEAE model. In other words, it is a basic prompt-based extractor in Section 3.1. Different from UnifiedEAE (Multiple), UnifiedEAE (Single) trains and tests on the same dataset without transferring.

4.4 Implementation Details

We initialize the weight in encoder-decoder architecture with pre-trained BART base models (Lewis et al., 2020). We use Adam optimizer with the learning rates of 2e-5. The max encoder sequence length is 500, and the max decoder length is 80. The dropout is 0.1. The reported results on the test set are based on the parameters that obtain the best performance on the development set.

5 Results and Analyses

To investigate the efficacy of UnifiedEAE model, we compare our model with the mainstream baselines (Section 5.1). Then, we do the ablation studies

Table 4: The performance of transfer learning between RAMS and WikiEvents.

	RAMS		WikiEvents		
	Args-I	Args-C	Args-I	Args-C	Head-C
UnifiedEAE	**55.05**	49.71	**67.68**	**62.79**	**68.26**
w/o SharedVIB	54.65	48.79	65.90	61.05	67.42
w/o SSP	54.87	**49.92**	63.60	59.27	67.41
UnifiedEAE (Single)	53.32	48.29	68.31	63.40	66.16

Table 5: The performance of transfer learning between ACE2005 and WikiEvents.

	ACE2005		WikiEvents		
	Args-I	Args-C	Args-I	Args-C	Head-C
UnifiedEAE	**76.06**	**71.85**	**69.84**	**64.00**	66.30
w/o SharedVIB	75.12	71.48	68.76	63.27	**67.76**
w/o SSP	74.34	70.80	67.63	62.66	66.28
UnifiedEAE (Single)	72.77	68.82	68.31	63.40	66.16

Table 6: The performance of transfer learning between ACE2005 and RAMS.

	ACE2005		RAMS	
	Args-I	Args-C	Args-I	Args-C
UnifiedEAE	**71.65**	**68.00**	**55.46**	49.94
w/o SharedVIB	67.59	62.96	55.12	**50.02**
w/o SSP	62.61	59.62	54.62	49.09
UnifiedEAE (Single)	72.77	68.82	53.32	48.29

to verify the performance of the parts consisting of our model from two views, the model structure and dataset transferring (Section 5.2). We also explore the effectiveness of transfer learning on low-resource EAE (Section 5.3) and case studies are given (Section 5.4).

5.1 Main Results

In this section, we compare our framework with several prior competitive baselines (Table 3). Note, for UnifiedEAE and UnifiedEAE (w/o SSP), we report the best results of transferring over each two datasets (e.g., ACE2005 and RAMS, ACE2005 and WikiEvents, RAMS and WikiEvents).

From the table, we find the following observations. **First**, we observe that our model consistently outperforms the state-of-the-art baseline in terms of Args-I and Args-C. Compared with the best baseline PAIE, our approach outperforms it with an improvement of 2.46% in terms of Args-I over ACE2005, which indicates the effectiveness of multi-format transfer learning. **Second**, our model can leverage the knowledge from other datasets effectively. Our UnifiedEAE model outperforms UnifiedEAE (w/o SSP), which trains a basic prompt-based extractor using merged data directly. Moreover, UnifiedEAE (Single) sometimes performs better than UnifiedEAE (w/o SSP). These indicate that merging two datasets directly for training may bring noise and lead to small gains or even drops. **Third**, we apply our transfer learning framework to implement zero-shot. It is capable of extracting new event argument roles that are unseen in the training phase using the format-shared model. From the results, we can observe that transferring between ACE2005 and WikiEvents achieves a good performance because they have many similar event types and argument roles.

5.2 Ablation Studies

We do the ablation studies to further investigate the effectiveness of the main components in our model from two perspectives, model structure and resource information. The results are shown in Table 4, 5 and 6.

From a model structure view, we remove the SharedVIB (w/o SharedVIB) and SSP (w/o SSP) from our model respectively. We observe that each component can help boost the performance of EAE. Particularly, **1)** Removing SSP will cause about four points decline in terms of argument identification and classification when transferring the knowledge of RAMS to WikiEvents (Row 1,3 in Table 4). This justifies that directly merging datasets may bring noises, which results in the degradation of test data. Our model can learn both the format-shared and format-specific knowledge, which improves the performance effectively. **2)** Our SharedVIB strategy can enhance the model to learn the format-shared knowledge. Removing SharedVIB from our model will reduce the performance in most cases. For example, UnifiedEAE obtains more than 4 points improvement compared with the one without SharedVIB when transferring RAMS to ACE2005.

To investigate the effectiveness of transfer learning among different resources, we evaluate our model on each two datasets. As we mentioned above, UnifiedEAE (Single) trains and tests on the same dataset and UnifiedEAE without SSP trains on the merged datasets. We can find that not all the transferring can improve the performance since it may contain noise for the target dataset. For example, transferring from RAMS to ACE2005 caused more than six points drop for both Args-I and Args-C compared with UnifiedEAE (Single) that only trains on ACE2005 (Row 3 and 4 in Table 6). Our model can reduce the influence of noise effectively by learning both format-shared

Train	Test	Business.Start-Org	Role	Shared	Specific	w/o SSP
ACE2005+ WikiEvents	ACE2005	**Founded**[trigger] by former mayor **Gholamhossein Karbaschi**[Agent] , **Hamshahri**[Org] was quick to become **Iran**[Place] 's biggest - selling daily with a circulation of 450,000 . It also built up healthy finances , carrying scores of pages of private advertisements daily.	Org	Hamshahri √	Hamshahri √	Gholamhossein Karbaschi ×
			Place	Iran √	Iran √	Iran √
			Agent	Gholamhossein Karbaschi √	Gholamhossein Karbaschi √	- ×

Train	Test	Personnel.Nominate	Role	Shared	Specific
RAMS+ WikiEvents	ACE2005	Suzanne I mean , I 'd like to s- -- I 'd like to see the **Greens**[Agent] **run**[trigger] David **Cobb**[Person] again.	Agent	Greens√	-
			Person	David Cobb√	-

Figure 3: Examples Visualization. We show the results of format-shared and format-specific extractors.

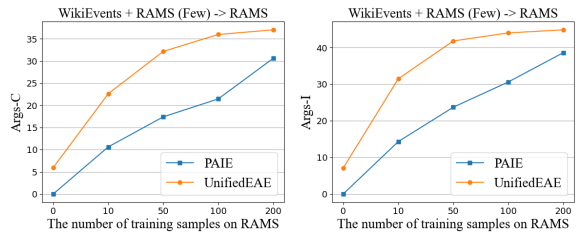

(a) Transfer from WikiEvents to RAMS (b) Transfer from WikiEvents to RAMS

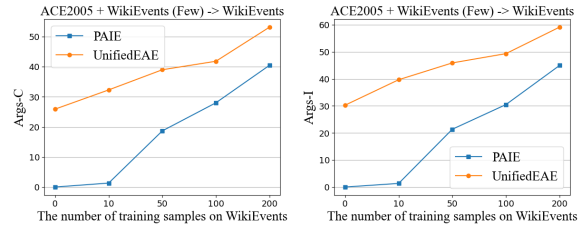

(c) Transfer from ACE2005 to WikiEvents (d) Transfer from ACE2005 to WikiEvents

Figure 4: The results of low-resource event argument extraction via transfer learning.

and format-specific knowledge.

5.3 Low-Resource EAE

Furthermore, we explore the performance of low-resource event argument extraction via transfer learning (Figure 4). We transfer the knowledge from the source dataset (e.g., WikiEvents) to the target dataset (e.g., RAMS) with few samples in the target dataset. In our experiments, we train our model with 0, 10, 50, 100, and 200 samples. From the results, we obtain the following observations. First, `UnifiedEAE` significantly outperforms the state-of-the-art PAIE model in terms of both Args-C and Args-I over two datasets. Particularly, our model achieves over 30 points with only ten examples on WikiEvents in terms of F1, while the PAIE model is almost not working. Second, `UnifiedEAE` obtains better performance with fewer samples compared with PAIE. `UnifiedEAE` uses ten examples and performs even better than PAIE with 100 examples. Third, `UnifiedEAE` with 200 samples achieves the comparable results with the existing baselines (e.g., BART-Gen, EEQA-BART) trained on full training datasets (3241 samples) on WikiEvents. All these findings indicate that our model captures the format-shared and format-specific knowledge and transfers the format-shared information effectively.

5.4 Case Studies

To make it easier to understand how our `UnifiedEAE` model works, we visualize two examples on ACE2005 in Figure 3. We find that our `UnifiedEAE` model transfers the knowledge effectively. 1) The format-shared module extracts the arguments correctly by learning the overlapping knowledge among multiple formats' datasets. However, `UnifiedEAE` (w/o `SSP`), which trains on the merging dataset directly, can not predict "Org" and "Agent". 2) We also train our model on RAMS and WikiEvents and test it on ACE2005 using a format-shared extractor, which is under a zero-shot setting. We find our format-shared model can extract the event roles for the event "Personnel.Nominate" though it does not appear in the training dataset.

6 Conclusions and Future Work

In this paper, we propose a unified event argument extraction (`UnifiedEAE`) model to transfer the knowledge among multi-format datasets. First, a shared-specific prompt architecture is introduced to extract the event arguments with multiple formats based on both format-shared and format-specific representations. Then, to enhance the

model to capture the format-shared knowledge effectively, we integrate the information bottleneck into our architecture. Variational information bottleneck is leveraged to eliminate the format-specific information and retain the format-shared knowledge. We conduct extensive experiments on three EAE datasets and compare our model with several strong baselines. The results show that our `UnifiedEAE` model outperforms the state-of-the-art baselines. Furthermore, the ablation studies show `SharedVIB` can capture the format-shared effectively. Our model also obtains good results on low-resource event argument extraction. In further work, we would like to adopt our model to other complex tasks, such as relation extraction, and named entity recognition.

Acknowledge

The authors wish to thank the reviewers for their helpful comments and suggestions. This work was partially funded by National Natural Science Foundation of China (No. 61976056, 62076069), Shanghai Municipal Science and Technology Major Project (No.2021SHZDZX0103).

References

Yubo Chen, Liheng Xu, Kang Liu, Daojian Zeng, and Jun Zhao. 2015. Event extraction via dynamic multi-pooling convolutional neural networks. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 167–176.

Yunmo Chen, Tongfei Chen, Seth Ebner, Aaron Steven White, and Benjamin Van Durme. 2020. Reading the manual: Event extraction as definition comprehension. In *Proceedings of the Fourth Workshop on Structured Prediction for NLP*, pages 74–83.

George R Doddington, Alexis Mitchell, Mark A Przybocki, Lance A Ramshaw, Stephanie M Strassel, and Ralph M Weischedel. 2004. The automatic content extraction (ace) program-tasks, data, and evaluation. In *Lrec*, volume 2, pages 837–840. Lisbon.

Xinya Du and Claire Cardie. 2020. Event extraction by answering (almost) natural questions. *arXiv preprint arXiv:2004.13625*.

Xinya Du, Alexander M Rush, and Claire Cardie. 2021. Grit: Generative role-filler transformers for document-level event entity extraction. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*, pages 634–644.

Seth Ebner, Patrick Xia, Ryan Culkin, Kyle Rawlins, and Benjamin Van Durme. 2020. Multi-sentence argument linking. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 8057–8077, Online. Association for Computational Linguistics.

Rui Feng, Jie Yuan, and Chao Zhang. 2020. Probing and fine-tuning reading comprehension models for few-shot event extraction. *arXiv preprint arXiv:2010.11325*.

Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long short-term memory. *Neural computation*, 9(8):1735–1780.

Lifu Huang, Heng Ji, Kyunghyun Cho, Ido Dagan, Sebastian Riedel, and Clare Voss. 2018. Zero-shot transfer learning for event extraction. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 2160–2170, Melbourne, Australia. Association for Computational Linguistics.

Diederik P. Kingma and Max Welling. 2014. Auto-encoding variational bayes. In *2nd International Conference on Learning Representations, ICLR 2014, Banff, AB, Canada, April 14-16, 2014, Conference Track Proceedings*.

Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Veselin Stoyanov, and Luke Zettlemoyer. 2020. BART: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7871–7880, Online. Association for Computational Linguistics.

Sha Li, Heng Ji, and Jiawei Han. 2021. Document-level event argument extraction by conditional generation. In *Proceedings of NAACL*, pages 894–908.

Xiang Lisa Li and Jason Eisner. 2019. Specializing word embeddings (for parsing) by information bottleneck. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 2744–2754.

Jiaju Lin, Qin Chen, Jie Zhou, Jian Jin, and Liang He. 2022. Cup: Curriculum learning based prompt tuning for implicit event argument extraction. In *Proceedings of the Thirty-First International Joint Conference on Artificial Intelligence, IJCAI-22*, pages 4245–4251. International Joint Conferences on Artificial Intelligence Organization.

Ying Lin, Heng Ji, Fei Huang, and Lingfei Wu. 2020. A joint neural model for information extraction with global features. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7999–8009.

Jian Liu, Yubo Chen, Kang Liu, Wei Bi, and Xiaojiang Liu. 2020. Event extraction as machine reading comprehension. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1641–1651.

Jian Liu, Yufeng Chen, and Jinan Xu. 2021a. Machine reading comprehension as data augmentation: A case study on implicit event argument extraction. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 2716–2725.

Pengfei Liu, Xipeng Qiu, and Xuanjing Huang. 2017. Adversarial multi-task learning for text classification. *arXiv preprint arXiv:1704.05742*.

Pengfei Liu, Weizhe Yuan, Jinlan Fu, Zhengbao Jiang, Hiroaki Hayashi, and Graham Neubig. 2021b. Pre-train, prompt, and predict: A systematic survey of prompting methods in natural language processing. *arXiv preprint arXiv:2107.13586*.

Yaojie Lu, Hongyu Lin, Jin Xu, Xianpei Han, Jialong Tang, Annan Li, Le Sun, Meng Liao, and Shaoyi Chen. 2021. Text2event: Controllable sequence-to-structure generation for end-to-end event extraction. *arXiv preprint arXiv:2106.09232*.

Yubo Ma, Zehao Wang, Yixin Cao, Mukai Li, Meiqi Chen, Kun Wang, and Jing Shao. 2022. Prompt for extraction? paie: Prompting argument interaction for event argument extraction. *arXiv preprint arXiv:2202.12109*.

Rabeeh Karimi Mahabadi, Yonatan Belinkov, and James Henderson. 2021. Variational information bottleneck for effective low-resource fine-tuning. *arXiv preprint arXiv:2106.05469*.

Haoruo Peng, Yangqiu Song, and Dan Roth. 2016. Event detection and co-reference with minimal supervision. In *Proceedings of the 2016 conference on empirical methods in natural language processing*, pages 392–402.

Fernando Pereira, Naftali Tishby, and Lillian Lee. 1994. Distributional clustering of english words. *arXiv preprint cmp-lg/9408011*.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J Liu. 2020. Exploring the limits of transfer learning with a unified text-to-text transformer. *Journal of Machine Learning Research*, 21:1–67.

Sebastian Ruder, Matthew E. Peters, Swabha Swayamdipta, and Thomas Wolf. 2019. Transfer learning in natural language processing. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Tutorials*, pages 15–18, Minneapolis, Minnesota. Association for Computational Linguistics.

Timo Schick and Hinrich Schütze. 2020. Exploiting cloze questions for few shot text classification and natural language inference. *arXiv preprint arXiv:2001.07676*.

Ananya Subburathinam, Di Lu, Heng Ji, Jonathan May, Shih-Fu Chang, Avirup Sil, and Clare Voss. 2019. Cross-lingual structure transfer for relation and event extraction. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 313–325, Hong Kong, China. Association for Computational Linguistics.

Qingyun Sun, Jianxin Li, Hao Peng, Jia Wu, Xingcheng Fu, Cheng Ji, and Philip S Yu. 2021. Graph structure learning with variational information bottleneck. *arXiv preprint arXiv:2112.08903*.

Naftali Tishby, Fernando C Pereira, and William Bialek. 2000. The information bottleneck method. *arXiv preprint physics/0004057*.

David Wadden, Ulme Wennberg, Yi Luan, and Hannaneh Hajishirzi. 2019. Entity, relation, and event extraction with contextualized span representations. In *Proceedings of EMNLP-IJCNLP*, pages 5784–5789.

Kaiwen Wei, Xian Sun, Zequn Zhang, Jingyuan Zhang, Guo Zhi, and Li Jin. 2021. Trigger is not sufficient: Exploiting frame-aware knowledge for implicit event argument extraction. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 4672–4682.

Peter West, Ari Holtzman, Jan Buys, and Yejin Choi. 2019. Bottlesum: Unsupervised and self-supervised sentence summarization using the information bottleneck principle. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 3752–3761.

Hang Yang, Yubo Chen, Kang Liu, Yang Xiao, and Jun Zhao. 2018. Dcfee: A document-level chinese financial event extraction system based on automatically labeled training data. In *Proceedings of ACL 2018, System Demonstrations*, pages 50–55.

Qi Zhang, Jie Zhou, Qin Chen, Qingchun Bai, and Liang He. 2022. Enhancing event-level sentiment analysis with structured arguments. In *Proceedings of the 45th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 1944–1949.

Tongtao Zhang, Heng Ji, and Avirup Sil. 2019. Joint entity and event extraction with generative adversarial imitation learning. *Data Intelligence*, 1(2):99–120.

Zhisong Zhang, Xiang Kong, Zhengzhong Liu, Xuezhe Ma, and Eduard Hovy. 2020. A two-step approach for implicit event argument detection. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7479–7485.

Jie Zhou, Junfeng Tian, Rui Wang, Yuanbin Wu, Wenming Xiao, and Liang He. 2020. SentiX: A sentiment-aware pre-trained model for cross-domain sentiment analysis. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 568–579, Barcelona, Spain (Online). International Committee on Computational Linguistics.

Jie Zhou, Yuanbin Wu, Qin Chen, Xuan-Jing Huang, and Liang He. 2021. Attending via both fine-tuning and compressing. In *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*, pages 2152–2161.

RSGT: Relational Structure Guided Temporal Relation Extraction

Jie Zhou* **Shenpo Dong*** **Hongkui Tu✉** **Xiaodong Wang** **Yong Dou**
College of Computer
National University of Defense Technology
{jiezhou,dsp,tuhongkui,xdwang,yongdou}@nudt.edu.cn

Abstract

Temporal relation extraction aims to extract temporal relations between event pairs, which is crucial for natural language understanding. Few efforts have been devoted to capturing the global features. In this paper, we propose **RSGT**: **R**elational **S**tructure **G**uided **T**emporal Relation Extraction to extract the relational structure features that can fit for both inter-sentence and intra-sentence relations. Specifically, we construct a syntactic-and-semantic-based graph to extract relational structures. Then we present a graph neural network based model to learn the representation of this graph. After that, an auxiliary temporal neighbor prediction task is used to fine-tune the encoder to get more comprehensive node representations. Finally, we apply a conflict detection and correction algorithm to adjust the wrongly predicted labels. Experiments on two well-known datasets, MATRES and TB-Dense, demonstrate the superiority of our method (**2.3%** F1 improvement on MATRES, **3.5%** F1 improvement on TB-Dense).

1 Introduction

Temporal relation extraction (TRE) is crucial for natural language understanding and can facilitate various downstream applications such as summarization (Zhou et al., 2010), question answering (Yu et al., 2017), and clinical diagnosis (Zhou et al., 2021). As shown in Figure 1, the goal of TRE is to determine the temporal order between an event pair (*BEFORE, AFTER*, etc.).

Most early methods were based on statistical machine learning (Mani et al., 2006; Chambers, 2013). In recent years, neural network based methods and large-scale pre-trained language models such as BERT (Devlin et al., 2018) have contributed to a substantial increase in the performance of TRE task (Ning et al., 2019; Wang et al., 2020).

S1: Former President Nicolas Sarkozy was **(e1,informed)** Thursday that he would face a formal investigation into whether he **(e2,abused)** the frailty of Liliane Bettencourt, to get funds for his 2007 presidential campaign.
S2: Mr. Sarkozy has **(e3,denied)** accepting illegal campaign funds from Ms. Bettencourt, either personally or through his party treasurer at the time, Eric Woerth, as **(e4,alleged)** by her former butler.

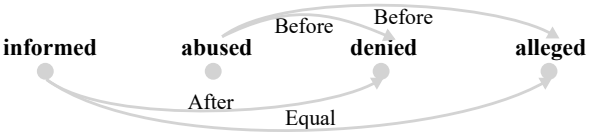

Figure 1: An example of temporal relation extraction. There are four events in these sentences. The graph below shows the pair-wise event temporal relationships.

However, these methods may ignore the global structure features which carry non-consecutive and long-distance semantics (Peng et al., 2018). This shortcoming is obvious in dealing with an event pair that the two events belong to different sentences (inter-sentences event pair), such as $< e_1, e_4 >$ in Figure 1. Few previous works differentiate inter-sentence event pairs from intra-sentence ones (where the two events appear in the same sentence). Thus, the performance may be impacted. For example, we test that there is a 5 accuracy points gap between intra-sentences and inter-sentence event pairs with the recent state-of-the-art method (Wen and Ji, 2021a).

To fill this gap, we aim to develop a structural features method that captures temporal semantic relations for both the inter-sentence and intra-sentence event pairs. Specifically, we adopt graph neural networks (GNNs), which have been proved to be effective in preserving global structure information of a graph in graph embeddings (Yao et al., 2019), to bridge the temporal relations.

Based on the above analysis, we present **RSGT**: **R**elational **S**tructure **G**uided **T**emporal Relation Extraction. To enable our model to learn more ef-

*These authors contributed equally to this work

Proceedings of the 29th International Conference on Computational Linguistics, pages 2001–2010
October 12–17, 2022.

fective representation for relational structures, we take the following strategies: First, to obtain more relational information, we create different types of connections for the graph nodes based on their syntactic and semantic information. Such connections are combined together to generate a rich relational graph. In particular, the node embeddings are obtained with the GGNN algorithm (Li et al., 2016). To avoid graph over-smoothing, RoBERTa (Liu et al., 2019) embeddings are concatenated with GGNN embeddings to make the final prediction.

Second, unlike most previous graph-based models which directly use the pre-trained language model as the node encoder, we present a task called temporal event neighbor prediction to fine-tune the encoder. This task aims to predict the neighbor node of event mentions from the relational graph. The fine-tuned encoder can help RSGT better understand the correlation between the relational structure and raw text. Ablation studies demonstrate that it can significantly boost efficiency.

Finally, we present a conflict detection and correction algorithm based on the transitivity rule of temporal relations to promote performance.

Experiments on two popular benchmarks, MATRES (Ning et al., 2018) and TB-Dense (Cassidy et al., 2014), show that RSGT outperforms the state-of-the-art methods (**2.3%** F1 points improvement on MATRES, **3.5%** F1 points improvement on TB-Dense). Meanwhile, we improve the accuracy of inter-sentence relations to the same level as intra-sentence relations.

2 Method

We formulate the TRE problem as a multi-class classification task. For a document D with n sentences $(S_1, S_2, ..., S_n)$, it can have multiple event mentions $E = (e_1, e_2, ..., e_m)$. The goal of TRE is to predict the temporal relation type between event pairs. For an event pair $< e_i, e_j >$, the input of our model is the sentence they belong to. In particular, if two events belong to different sentences (we call it inter-sentence event pair), two sentences $< S_i, S_j >$ are concatenated together as the input.

Our work RSGT involves five major parts: (i) Structure Generation to generate a relational-guided graph based on syntactic-and-semantic information, (ii) Temporal Event Neighbor Prediction to transform words into embedding vectors, (iii) Relational-guided Graph Model to predict temporal relations, (iv) Conflict Detect and Correct algorithm to revise temporal errors.

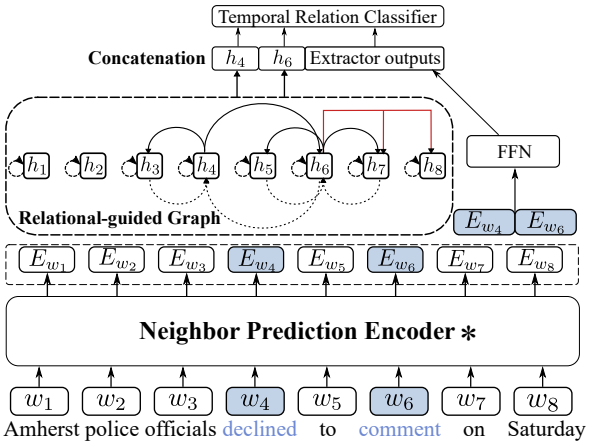

Figure 2: The illustrative architecture of the proposed Relational-guided Graph Model. Our goal is to extract the temporal relation of $< w_4, w_6 >$. In the relational-guided graph, black arcs mean syntactic-guided edges $\mathcal{E}_d$ and red arcs mean semantic-guided edges $\mathcal{E}_t$. * indicates a RoBERTa model fine-tuned on the Temporal Event Neighbor Prediction task.

2.1 Structure Generation

Building graphs is a feature selection process that can facilitate representation learning for the TRE problem. Given an input sentence S, our goal is to generate a relational graph $\mathcal{G} = \{\mathcal{N}, \mathcal{E}\}$ as the input of our graph neural network. Our relational graph is based on syntactic and semantic information extracted from S. The node set $\mathcal{N}$ and edge set $\mathcal{E}$ in $\mathcal{G}$ are constructed as follows strategies.

2.1.1 Nodes

The node set $\mathcal{N}$ should capture all objects related to temporal events. We take two types of nodes to make up the node set. The first type is from the original words $w_i \in S$. The second type is the event arguments extracted by the Semantic Role Labeling (SRL) model, which we will introduce in the semantic-guided edges section. Formally, let $W = \{w_1, w_2, ..., w_{|W|}\}$, $Arg = \{a_1, a_2, ..., a_{|Arg|}\}$ be the set of words and event arguments, respectively. Then the node set of the input sentence should consist of two parts: $\mathcal{N} = \{W \cup Arg\}$. After the generation of $\mathcal{G}$, nodes with no edges pointing to other nodes are removed from $\mathcal{N}$.

2.1.2 Syntactic-guided Edges

Dependency Parsing (DP) can examine the dependencies between the phrases of a sentence to determine its syntactic structure. As such, we apply the dependency parsing tree of the input sentence

to build syntactic-guided edges $\mathcal{E}_d$. For the dependency tree consisting of multiple head-dependent arcs, the arcs whose head is event mention are converted to edges $\mathcal{E}_{d\_along}$ as the solid black arcs in Figure 2. In addition, we assume that information flows not only along the syntactic dependency arcs, so we create edges $\mathcal{E}_{d\_rev}$ in the opposite direction as well (i.e., from dependents to heads). Following Kipf and Welling (2016), we also add self-edges for each nodes as $\mathcal{E}_{d\_loop}$. Therefore, syntactic-guided edges $\mathcal{E}_d$ contains three kinds of edges $\mathcal{E}_{d\_along}$, $\mathcal{E}_{d\_rev}$ and $\mathcal{E}_{d\_loop}$.

2.1.3 Semantic-guided Edges

We design semantic-guided edges $\mathcal{E}_t$ to fetch semantic information related to a temporal event. Specifically, we want to import an event extraction model that can extract event arguments based on event mentions. SRL-BERT (Shi and Lin, 2019) becomes our final choice because it not only meets our above requirements but also marks out the argument's types. As shown in the Figure 2's red arcs, arguments are connected to the event mentions as $\mathcal{E}_t$. SRL task assumes event mentions trigger the arguments, so we only consider unidirectional edges from event nodes. Some particular argument types, such as Temporal and Discourse, which can probably provide extra information to understand the temporal relation, are assigned to different edge types with higher weight.

2.2 Temporal Event Neighbor Prediction

In the graph model, we need to apply an encoder to transform each word $w_i \in S$ into a contextual represented vector for nodes. Most previous studies directly use pre-trained language models as the encoder. However, Chien et al. (2021) argues that these pre-trained language models ignore the correlations between graph topology and raw text features. Inspired by this work, we propose a task called Temporal Event Neighbor Prediction. Given a syntactic-guided graph $\mathcal{G}_d$ we construct, this task aims to distinguish whether a node is the neighbor of the event mention's node or not. We pick k words before and after per event mention respectively in the sentence, and they can form node pairs with its event mention.

Take the sentence in 2 as an example. Suppose we are using $k = 2$, so for the first event mentions w_4, we pick 4 words before and after w_4, which are $\{w_2, w_3, w_5, w_6\}$. Node pairs $< w_4, w_3 >, < w_4, w_6 >$ are neighbors, so their labels are 1. $< w_4, w_2 >, < w_4, w_5 >$ are not neighbors and their labels are 0. The second event mentions w_6 can be treated in the same way. The input of this task is each event-neighbor pair $< w_e, w_{nbr} >$ and its raw sentence.

To handle this task, we first apply RoBERTa to encode the sentences and extract the nodes' embeddings of $< w_e, w_{nbr} >$. The represented vector of two nodes then passes through a Feed-Forward Network (FFN) layer with a $tanh$ activation function, respectively. For the output of FFN layer h_e and h_{nbr}, we concatenate them together and apply Batch Normalization as the representation of node pair. Then a FFN layer with softmax is added for prediction. The model can be formalized as:

$$\begin{aligned} h_e &= \tanh(\mathbf{FFN}_1(\phi(w_e))) \\ h_{nbr} &= \tanh(\mathbf{FFN}_2(\phi(w_{nbr}))) \\ \hat{y_{nbr}} &= \mathrm{softmax}(\mathbf{FFN}_3(\mathbf{BN}[h_{nbr}; h_e])) \end{aligned} \tag{1}$$

where BN denotes Batch Normalization, and ϕ is the encoder that maps w to feature vectors. We adjust k to ensure that the distribution of labels is balanced. To make sure RoBERTa can maintain more topology information from the relational graph, the learning rate of RoBERTa is larger than other layers.

This task allows the encoder to understand not only the contextual information from the raw text but also the topology information from our relational graph $\mathcal{G}$. We select the model with the best accuracy in the validation set as the encoder. Then we apply this fine-tuned encoder to represent the node set $\mathcal{N}$. This task can be further extended to other graph-related models as an efficient way for the encoder's fine-tuning.

2.3 Relational-guided Graph Model

We have already generated a relational graph $\mathcal{G}$ and the represented vector x for each node. We apply Gated Graph Sequence Neural Networks (GGNN) to handle our relational graph. GGNN employs a gated recurrent unit (GRU) as a recurrent function, reducing the recurrence to a fixed number of steps. The advantage is that it no longer needs to constrain parameters to ensure convergence. We parse each sentence into the relational graph and use GGNN to digest this structural information. The forward process of GGNN is:

$$
\begin{aligned}
x_u &= \phi(w_u) \\
h_u^0 &= [x_u \| \mathbf{0}] \\
a_u^t &= \sum_{v \in \mathcal{N}(u)} W_{e_{uv}} h_v^t \\
h_u^{t+1} &= \mathrm{GRU}(a_u^t, h_u^t)
\end{aligned} \tag{2}
$$

where u denotes the current node and v denotes the neighbor node of u. ϕ is the fine-tuned encoder, and h_u^t denotes the t step hidden states of u.

As discussed in Chen et al. (2020), over-smoothing is a common issue faced by GNNs, which means that the representations of the graph nodes of different classes would become indistinguishable when stacking multiple layers. To avoid over-smoothing problems, the embeddings $< x_i, x_j >$ from fine-tuned RoBERTa are passed through a fully connected layer parallel with GGNN. For event pair $< x_i, x_j >$, the representation H_F from the fully connected layer is then concatenated with GGNN's final hidden states H_G. Concatenation may help us maintain some contextual information from RoBERTa encoder and increase the differentiation of event representations. In the end, we apply a two-layer FFN as classifier f and a BatchNorm layer for the final temporal prediction. The final output of event pair $< e_i, e_j >$ is:

$$
\hat{y}_{ij} = f(\mathbf{BN}[H_{G_i}; H_{G_j}; H_{F_i}; H_{F_j}]) \tag{3}
$$

The overall loss function to train our model is:

$$
\mathcal{L} = -\sum_{i,j} y_{ij}^* log(\hat{y}_{ij}) + \gamma \mathcal{L}_{reg} \tag{4}
$$

where y_{ij}^* is the gold labels of temporal relations and γ is a trade-off parameter for regularization techniques.

2.4 Conflict Detect and Correct

There exists a transitivity rule in temporal relationships. Take the events depicted in Figure 1 as an example. We consider the intra-sentence and inter-sentence event pairs relationships together and build the temporal diagram on the left side of the Figure 3. A transitivity rule could be explained as "e_2 happens before e_1, e_1 and e_4 occur simultaneously, then e_4 should happen after e_2". On the right side of Figure 3 is a counterexample. The red arrows can form a cycle, which indicates that at least one temporal relation edge violates the transitivity rule.

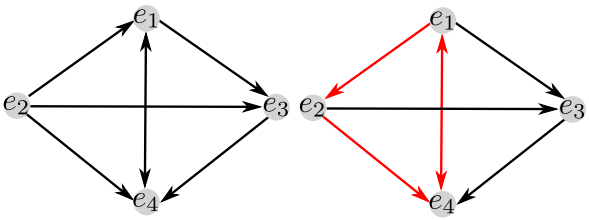

Figure 3: The example of transitivity rule in temporal relationship. Unidirectional arrows represent *BEFORE*, like $e_2 \rightarrow e_1$ refers to e_2 happens before e_1. Bidirectional arrows represent two events occurring simultaneously.

To take full advantage of this rule, we design an algorithm to find potential conflicts. From the output of the classifier f, we obtain a temporal relationship prediction $\hat{y}_{ij}$ for the event pair $< e_i, e_j >$. We can build a document-level temporal relational graph by collecting temporal relation predictions as edges and events as nodes from document D. For *BEFORE* relation of $< e_i, e_j >$, we add an edge from e_i to e_j. On the contrary, we add an edge from e_j to e_i for *AFTER*. We treat *EQUAL* as a bidirectional edge. Other temporal relations are ignored (e.g. *VAGUE*). Obviously, this graph should be a Directed Acyclic Graph (DAG). So our goal is to find the conflict cycles and correct them.

We re-implement the Johnson cycle algorithm (Johnson, 1975) as our temporal event conflict detection algorithm. It was presented to find all the elementary cycles of a directed graph, which time bounded by $O((n+e)(c+1))$ for n nodes, e edges and c elementary cycles.

Then we use algorithm 1 to detect and correct conflict. Basically, we:

1. Apply $conflict\_detect$ algorithm to find elementary cycles in the $edges$.
2. Pick the longest cycle from step 1 and initialize variables $cycle\_n$ as cycle's length, m_logit, m_edge as the smallest logit and its edge (lines 5-7).
3. Traverse all nodes in the $cycle$ and find the smallest logit (lowest probability of confidence $edge\_logit$). Store the start and end node of m_edge(lines 8-20).
4. Reverse the edge found in step 3 to solve the conflicts. Remove m_edge from the graph if it has been corrected twice. Go back to step 1 and repeat until the graph is a directed acyclic graph (lines 21-26).

Algorithm 1: Correct Algorithm

Input : $edges$
Output : Corrected edges

$revised$ = [] ;
while *True* **do**
  $cycles$ = conflict_detect($edges$);
  if *no cycles* **then** break;
  $cycle \leftarrow$ longest($cycles$);
  $cycle\_n \leftarrow$ length($cycle$);
  $m\_logit, m\_edge \leftarrow -1, (-1, -1)$;
  for *i in range*$(1, cycle\_n)$ **do**
    if $i! = cycle\_n$ **then**
      j = i+1 ;
    else
      j = 1;
    end
    $fr \leftarrow cycle[i]$;
    $to \leftarrow cycle[j]$;
    $edge\_logit = edges[now][to]$;
    if $m\_logit \leq edge\_logit$ **then**
      $m\_logit = edge\_logit$;
      $m\_edge = (fr, to)$
  end
  $fr, to \leftarrow m\_edge$;
  if fr, to *in* $revised$ **then**
    remove $edge_{fr,to}$
  $revised$. add(m_edge) ;
  reverse $edge_{fr,to}$ to $edge_{to,fr}$;
  $cycles \leftarrow$ collision detection (adj_{d_i})
end

This algorithm is concise and efficient, and it can be well adapted to the correction work of various datasets without training.

3 Experiments

3.1 Datasets

We conduct our experiments on two well-known benchmarks for the TRE task, MATRES(Ning et al., 2018) and TB-Dense(Cassidy et al., 2014). MATRES contains refined annotations on TimeBank(Pustejovsky et al., 2003) , AQUAINT and Platinum documents. It contains four types of temporal labels: *BEFORE, AFTER, EQUAL, VAGUE*. TB-Dense is a densely annotated dataset from TimeBank and TempEval(UzZaman et al., 2013). This dataset contains six label types. In addition to the four label types from MATRES, it has two more label types: *INCLUDES* and *IS_INCLUDED*. For compatible comparison, we apply the same data splits as in prior work for the considered datasets. The detailed statistics can be found in Table 1.

3.2 Evaluation Metrics

We adopt micro averaged precision, recall, and F1 scores as evaluation metrics following the previous works(Ning et al., 2018; Wen and Ji, 2021a; Cao et al., 2021). For the MATRES, *VAGUE* is considered to be non-temporal information and is excluded from the F1 calculation. For the TB-Dense, *VAGUE* is taken into consideration (i.e., all label types are seen as positive classes) so the metric should share the same precision, recall, and F1 value. We follow these different settings for our experiments to ensure fair comparisons.

Dataset	Train	Validation	Test	Labels
MATRES	10888	1852	837	a,b,e,v
TB-Dense	4032	629	1427	a,b,s,v,i,ii

Table 1: Data splits and relation pairs statistics. a: *AFTER*, b: *BEFORE*, e: *EQUAL*, s: *SIMULTANEOUS*, v: *VAGUE*, i: *INCLUDES*, ii: *IS_INCLUDED*.

3.3 Implement Details

The hyperparameters used in the experiment are listed. **Neighbor Prediction:** RoBERTa-large is adopted to encode the sentence. The learning rate for RoBERTa and FFN are 1e-5, 1e-4, respectively. **Syntactic Information:** We apply SpaCy [*] toolkit to build dependency trees based on input sentences. **Semantic Information:** The event arguments corresponding to each event mention are extracted from SRL-BERT. **Graphs Training:** AdamW with learning rate of 5e-6, β_1 = 0.9, β_2 = 0.999 and weight decay of 0.01 is used for optimization. We set the training epochs and batch size to 40 and 32, respectively. Besides, we exploit a dropout with a rate of 0.5 on the concatenated feature representations.

3.4 Baselines

We conduct experiments to compare our approach RSGT with the state-of-the-art models for TRE in each benchmark dataset as follows. Note that MATRES is a relatively new dataset, so we can hardly find more baselines that perform well on both MATRES and TB-Dense.

[*] https://spacy.io/

Dataset	Models	P	R	F1
MATRES	Siamese	66.6	60.8	63.0
	Constrained	72.1	80.8	76.2
	UAST	76.6	84.9	80.5
	SMTL	-	-	81.6
	Stack-Propagation	78.4	85.2	81.7
	RSGT	**82.2**	**85.8**	**84.0**
TB-Dense	Timelines	56.6	56.6	56.6
	UAST	64.3	64.3	64.3
	CTRL-PG	65.2	65.2	65.2
	RSGT	**68.7**	**68.7**	**68.7**

Table 2: Model performance on MATRES and TB-Dense. The performance improvement of RSGT over the baselines is significant with $p < 0.01$

MATRES For this dataset, the following baselines are chosen for comparison. (i) **Siamese** (Ning et al., 2019): A Siamese encoder of a temporal commonsense knowledge base, and global inference via integer linear programming (ILP). (ii) **Constrained** (Wang et al., 2020): A framework bridges temporal and subevent relation extraction tasks with a comprehensive set of logical constraints. (iii) **SMTL** (Ballesteros et al., 2020): A model relies on multi-task learning and self-training techniques. (iv) **Stack-Propagation** (Wen and Ji, 2021a): A Stack-Propagation framework to further incorporate predicted timestamp explicit for relation classification.

TB-Dense We use the following baselines for comparison. (i) **Timelines** (Vashishtha et al., 2019) A semantic framework for modeling fine-grained temporal relations and event duration that maps pairs of events to real-valued scales and constructs document-level event timelines. (ii) **UAST** (Cao et al., 2021) An uncertainty-aware self-training framework to quantify the model uncertainty. (iii) **CTRL-PG** (Zhou et al., 2021) A method with probabilistic soft logic Regularization and global inference at the document-level.

3.5 Overall Performance

The most important observation from Table 2 is that model **RSGT** has significantly outperformed all the baseline systems on both MATRES and TB-Dense. Thus evidently indicating the effectiveness of the proposed RSGT model for the TRE task. Compared with the previous SOTA method Stack-Propagation, which also uses RoBERTa, our RSGT has 2.3 % F1 improvement on the MATRES dataset.

	Intra-sentences			Inter-sentences		
	P	R	F1	P	R	F1
RoBERTa-F	81.6	81.9	81.7	77.0	78.7	78.0
Stack-Propagation	77.9	84.5	81.1	73.6	85.7	79.2
RSGT	**83.1**	**84.5**	**83.8**	**81.8**	**86.4**	**84.1**

Table 3: Performance of different models on **MATRES**.

For the more complex dataset TB-sense with six temporal relation types, RSGT also has a 2.6% F1 improvement over the previous SOTA method CTRL-PG. Overall, our method RSGT establishes a new state-of-the-art on two popular datasets of the TRE task.

3.6 Intra- and Inter-sentence

Inter-sentence event pairs make up a considerable proportion of the MATRES dataset (69.53% in the train set and 69.77% in the test set). Consequently, the performance on inter-sentence event pairs can significantly influence the overall performance. To explicitly demonstrate the effect of RSGT on the extraction of intra- and inter-sentence event pairs, we conduct a contrast experiment on the MATRES dataset. We attach a learnable fully-connected layer after RoBERTa as the baseline **RoBERTa-F**. The performance on the intra- and inter-sentences is shown in Figure 3. The previous SOTA method, Stack-Propagation, has a clear 4.3% gap in precision value between intra- and inter-sentences. As a comparison, we can observe an absolute F1 gain from RSGT, 2.7% and 4.9% on the intra-sentences and inter-sentences, respectively. Importantly, we successfully fill the performance gap between intra- and inter-sentence event pairs and improve their F1 result to the same level. These experiments show that the introduction of relational structure is of great help for inter-sentence temporal relations extraction.

3.7 Ablation Study

To illustrate the impact of each component in RSGT, we further conduct ablation studies with different configurations. Note that MATRES is a relatively new dataset, so we can hardly find more baselines that perform well both on MATRES and TB-Dense.

3.7.1 Effect of Neighbor Prediction

We propose the Neighbor Prediction task so that the encoder can learn the correlation between the relational graph's topology and raw text. In the

Model	P	R	F1
RSGT -w/o neighbor prediction	79.7	82.7	81.2
RSGT -w event prediction	69.7	79.2	74.1
RoBERTa	78.4	80.0	79.1
RSGT -w/o $\mathcal{E}_d$	80.5	84.7	82.5
RSGT -w/o $\mathcal{E}_t$	81.7	85.5	83.6
RSGT independent	81.0	84.8	82.8
RSGT	**82.2**	**85.8**	**84.0**

Table 4: Performance of different models on **MATRES**

MATRES dataset, the Neighbor Prediction task can reach 88.6% accuracy. In Table 4, **RSGT -w/o neighbor prediction** is RSGT excluding the Neighbor Prediction task, that is, using the original pre-trained RoBERTa model as encoder. As for **RSGT -w event prediction**, we replace the Neighbor Prediction task with a simple event extraction task to fine-tune the node encoder. Event extraction aims to extract event mentions from an input sentence. The result shows that: (1) The topological knowledge about relational graphs learned by neighbor prediction task can greatly improve the subsequent models. (2) Other types of knowledge, such as knowledge implied by the event extraction model, may not positively affect the TRE task.

3.7.2 Effect of Relational Structure Features

We examine the following ablated models to evaluate the effectiveness of different relational structure features in RSGT on the TRE task. (i) **RoBERTa** is a baseline with RoBERTa model and a fully-connected layer. (ii) **RSGT -w/o** $\mathcal{E}_d$ excludes the syntactic-guided edges. (iii) **RSGT -w/o** $\mathcal{E}_t$ excludes the semantic-guided edges. (iv) **RSGT independent** apply syntactic and semantics information to construct two independent graphs, respectively. At last, we average the embeddings of the two graphs.

The bottom half of Table 4 shows the performance of the above ablated models. We can observe that all the components can contribute to the proposed model RSGT as eliminating any of them degrades the performance in the F1 score. Apparently, the worse performance of **RSGT -** $\mathcal{E}_d$ model illustrates that syntactic information contributes a major improvement on TRE. And the **RSGT -** $\mathcal{E}_t$ model that removes semantic information slightly loses the performance of 0.4% F1. This is because the syntactic information contains more knowledge about the current event pair, and syntactic information may contain semantic information (event arguments) in some cases. Compared with **RSGT independent**, the independent graphs lack the interaction of all relational structure information. Instead, syntactic and semantic guided information should work together to form an interactive graph to enrich the relational structure obtained from RSGT.

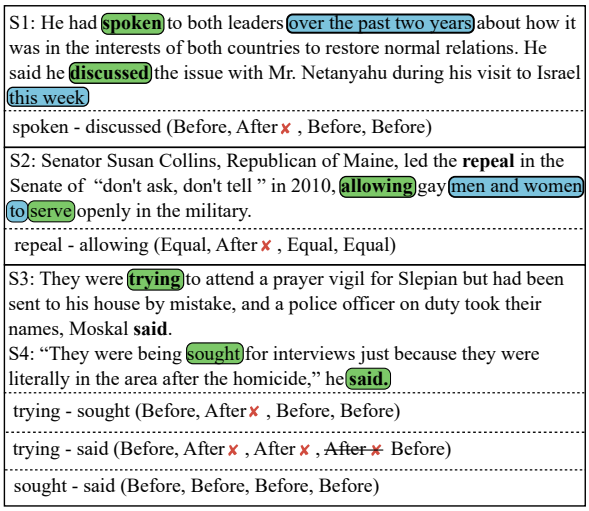

Figure 4: Case study. Event mentions and important relational structure are highlighted by green and blue respectively. Each line after sentence S has a structure like $< e_1, e_2 >$ (G,P_1,P_2,P_3), where e_1, e_2 is an event pair and G is the gold temporal label. P_1, P_2 and P_3 denotes prediction from Stack-Propagation (Wen and Ji, 2021b), RSGT-w/o conflict algorithm and RSGT respectively. Incorrect predictions are denoted by a red mark. Strikethrough means the prediction is corrected by our conflict detect and correct algorithm.

3.7.3 Effect of Conflict Detect and Correct

This algorithm is training-free and the time complexity is $O(n)$. Limited by the test set size, the improvement is slight (about 0.1%) on both MATRES and TB-dense datasets. Notes that the performance improvement from conflict detection gradually decreases with the training process. For example, it can bring a 4.3% average improvement in the first epoch, which means conflict detection can bring huge performance improvements in the early stage. We believe that it will play a more critical role in larger-scale datasets or real-world cases.

3.8 Case Study and Error Analysis

To promote a better understanding of our RSGT and guide potential research direction, we analyze three concrete examples in Figure 4. Each case has a pair of events, and the study results can be categorized into different types that are described

below:

Case 1. Sentence S1 contains a conversation event mentions "spoken" and a discussion event mentions "discussed". RSGT correctly predicts the temporal relations while Stack-Propagation fails. RSGT successfully extracts two temporal arguments from S1, enhancing the model's inference ability by providing the time of occurrence. Obviously, "over the past two years" has happened "this week". The previous model does not utilize semantic information, which leads to misclassification.

Case 2. The small proportion of *EQUAL* (about 3.6% in MATRES) makes temporal relationship prediction more challenging, as it can easily be confused with more common labels like *BEFORE* and *AFTER*. Sentence S3 contains two events, "allowing" and "serve". It seems like a simple task for a human. However, Stack-Propagation relies only on two event words and fails to recognize their interaction. We highlight some syntactic information extracted by RSGT. "allow someone to do something" is a typical relational structure that happens simultaneously. As a result, this relational structure makes the prediction much easier for RSGT.

Case 3. S3 and S4 show one intra-sentence and two inter-sentence event temporal relations. Our RSGT correctly classifies <trying, sought>, <sought, said> event pairs. For an inter-sentence event pair like <trying, said>, which is so hard that RSGT fails to predict its temporal relation initially, the conflict detect and correct algorithm can utilize the relationships between the other two event pairs to correct the result. In the directed graph built from the predictions, we obtain three edges (trying $\rightarrow$ sought), (trying $\leftarrow$ said), (sought $\rightarrow$ said). Obviously, this graph does not meet the DAG definition, and our algorithm reverses the edge with a minimum confidence score (trying $\leftarrow$ said) to correct it.

4 Related Work

Earlier efforts on TRE (temporal relation extraction) use statistical machine learning techniques (Support Vector Machine, Max entropy) and handcraft features (e.g Verhagen and Pustejovsky (2008) and Chambers (2013)). Recently, neural methods and large-scale pre-training language models have also achieved promising improvement (Nguyen and Grishman, 2015; Nguyen et al., 2016; Wang et al., 2020; Mathur et al., 2021). The early feature-based methods for TRE have explored different features and resources to improve the performance, including syntactic patterns and lexical features (Cheng and Miyao, 2017; Mirza and Tonelli, 2016). Unlike previous works, our approach RSGT takes account of relational structure features to induce more accurate representations.

A wave of research at the intersection of deep learning on graphs has influenced a variety of NLP tasks, including event extraction (Xu et al., 2021; Yan et al., 2019), relation extraction (Tran Phu and Nguyen, 2021; Su et al., 2022) and event argument extraction (Pouran Ben Veyseh et al., 2020). These graph-structured data can encode complicated relations between event pairs to infer temporal order. Our model is different from such related works in that we designed a relational structure guided graphs that are tailored to our TRE task. In addition, we introduce a novel Temporal Event Neighbor Prediction task for the fine-tuning of the node encoder.

5 Conclusion

This paper proposes RSGT to capture relational structure information for the temporal relation extraction task. The experimental results well demonstrate our model's effectiveness and superiority in both the overall datasets and the inter-sentence event pairs. Ablation experiments show that the relational graph model and *Temporal Event Neighbor Prediction* contribute greatly to RSGT's performance.

Our future work will focus on how to apply *Temporal Event Neighbor Prediction*, and *Conflict Detect and Correct Algorithm* to other tasks with rich relations such as Casual Relations (Caselli and Vossen, 2017). We believe these methods are promising in processing relational structure information from other relational extraction tasks.

References

Miguel Ballesteros, Rishita Anubhai, Shuai Wang, Nima Pourdamghani, Yogarshi Vyas, Jie Ma, Parminder Bhatia, Kathleen McKeown, and Yaser Al-Onaizan. 2020. Severing the edge between before and after: Neural architectures for temporal ordering of events. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 5412–5417, Online. Association for Computational Linguistics.

Pengfei Cao, Xinyu Zuo, Yubo Chen, Kang Liu, Jun Zhao, and Wei Bi. 2021. Uncertainty-aware self-

training for semi-supervised event temporal relation extraction. In *Proceedings of the 30th ACM International Conference on Information & Knowledge Management*, pages 2900–2904.

Tommaso Caselli and Piek Vossen. 2017. The event storyline corpus: A new benchmark for causal and temporal relation extraction. In *Proceedings of the Events and Stories in the News Workshop*, pages 77–86.

Taylor Cassidy, Bill McDowell, Nathanel Chambers, and Steven Bethard. 2014. An annotation framework for dense event ordering. Technical report, Carnegie-Mellon Univ Pittsburgh PA.

Nathanael Chambers. 2013. NavyTime: Event and time ordering from raw text. In *Second Joint Conference on Lexical and Computational Semantics (*SEM), Volume 2: Proceedings of the Seventh International Workshop on Semantic Evaluation (SemEval 2013)*, pages 73–77, Atlanta, Georgia, USA. Association for Computational Linguistics.

Deli Chen, Yankai Lin, Wei Li, Peng Li, Jie Zhou, and Xu Sun. 2020. Measuring and relieving the over-smoothing problem for graph neural networks from the topological view. In *Proceedings of the AAAI Conference on Artificial Intelligence*.

Fei Cheng and Yusuke Miyao. 2017. Classifying temporal relations by bidirectional LSTM over dependency paths. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 1–6, Vancouver, Canada. Association for Computational Linguistics.

Eli Chien, Wei-Cheng Chang, Cho-Jui Hsieh, Hsiang-Fu Yu, Jiong Zhang, Olgica Milenkovic, and Inderjit S Dhillon. 2021. Node feature extraction by self-supervised multi-scale neighborhood prediction. *arXiv preprint arXiv:2111.00064*.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*.

Donald B Johnson. 1975. Finding all the elementary circuits of a directed graph. *SIAM Journal on Computing*, 4(1):77–84.

Thomas N Kipf and Max Welling. 2016. Semi-supervised classification with graph convolutional networks. *arXiv preprint arXiv:1609.02907*.

Yujia Li, Richard Zemel, Marc Brockschmidt, and Daniel Tarlow. 2016. Gated Graph Sequence Neural Networks. In *Proceedings of ICLR'16*. Edition: Proceedings of ICLR'16.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Roberta: A robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692*.

Inderjeet Mani, Marc Verhagen, Ben Wellner, Chungmin Lee, and James Pustejovsky. 2006. Machine learning of temporal relations. In *Proceedings of the 21st International Conference on Computational Linguistics and 44th Annual Meeting of the Association for Computational Linguistics*, pages 753–760.

Puneet Mathur, Rajiv Jain, Franck Dernoncourt, Vlad Morariu, Quan Hung Tran, and Dinesh Manocha. 2021. TIMERS: Document-level temporal relation extraction. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 2: Short Papers)*, pages 524–533, Online. Association for Computational Linguistics.

Paramita Mirza and Sara Tonelli. 2016. CATENA: CAusal and TEmporal relation extraction from NAtural language texts. In *Proceedings of COLING 2016, the 26th International Conference on Computational Linguistics: Technical Papers*, pages 64–75, Osaka, Japan. The COLING 2016 Organizing Committee.

Thien Huu Nguyen, Kyunghyun Cho, and Ralph Grishman. 2016. Joint event extraction via recurrent neural networks. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 300–309, San Diego, California. Association for Computational Linguistics.

Thien Huu Nguyen and Ralph Grishman. 2015. Event detection and domain adaptation with convolutional neural networks. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 2: Short Papers)*, pages 365–371, Beijing, China. Association for Computational Linguistics.

Qiang Ning, Sanjay Subramanian, and Dan Roth. 2019. An improved neural baseline for temporal relation extraction. *arXiv preprint arXiv:1909.00429*.

Qiang Ning, Hao Wu, and Dan Roth. 2018. A multi-axis annotation scheme for event temporal relations. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1318–1328, Melbourne, Australia. Association for Computational Linguistics.

Hao Peng, Jianxin Li, Yu He, Yaopeng Liu, Mengjiao Bao, Lihong Wang, Yangqiu Song, and Qiang Yang. 2018. Large-scale hierarchical text classification with recursively regularized deep graph-cnn. In *Proceedings of the 2018 world wide web conference*, pages 1063–1072.

Amir Pouran Ben Veyseh, Tuan Ngo Nguyen, and Thien Huu Nguyen. 2020. Graph transformer networks with syntactic and semantic structures for event argument extraction. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 3651–3661, Online. Association for Computational Linguistics.

James Pustejovsky, Patrick Hanks, Roser Sauri, Andrew See, Robert Gaizauskas, Andrea Setzer, Dragomir Radev, Beth Sundheim, David Day, Lisa Ferro, et al. 2003. The timebank corpus. In *Corpus linguistics*, volume 2003, page 40. Lancaster, UK.

Peng Shi and Jimmy Lin. 2019. Simple bert models for relation extraction and semantic role labeling. *arXiv preprint arXiv:1904.05255*.

B. Y. Su, S. L. Hsu, K. Y. Lai, and A. Gupta. 2022. Temporal relation extraction with a graph-based deep biaffine attention model. *arXiv e-prints*.

Minh Tran Phu and Thien Huu Nguyen. 2021. Graph convolutional networks for event causality identification with rich document-level structures. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 3480–3490, Online. Association for Computational Linguistics.

Naushad UzZaman, Hector Llorens, Leon Derczynski, James Allen, Marc Verhagen, and James Pustejovsky. 2013. Semeval-2013 task 1: Tempeval-3: Evaluating time expressions, events, and temporal relations. In *Second Joint Conference on Lexical and Computational Semantics (* SEM), Volume 2: Proceedings of the Seventh International Workshop on Semantic Evaluation (SemEval 2013)*, pages 1–9.

Siddharth Vashishtha, Benjamin Van Durme, and Aaron Steven White. 2019. Fine-grained temporal relation extraction. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 2906–2919, Florence, Italy. Association for Computational Linguistics.

Marc Verhagen and James Pustejovsky. 2008. Temporal processing with the TARSQI toolkit. In *Coling 2008: Companion volume: Demonstrations*, pages 189–192, Manchester, UK. Coling 2008 Organizing Committee.

Haoyu Wang, Muhao Chen, Hongming Zhang, and Dan Roth. 2020. Joint constrained learning for event-event relation extraction. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 696–706, Online. Association for Computational Linguistics.

Haoyang Wen and Heng Ji. 2021a. Utilizing relative event time to enhance event-event temporal relation extraction. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 10431–10437.

Haoyang Wen and Heng Ji. 2021b. Utilizing relative event time to enhance event-event temporal relation extraction. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 10431–10437, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Mingzhou Xu, Liangyou Li, Derek F. Wong, Qun Liu, and Lidia S. Chao. 2021. Document graph for neural machine translation. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 8435–8448, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Haoran Yan, Xiaolong Jin, Xiangbin Meng, Jiafeng Guo, and Xueqi Cheng. 2019. Event detection with multi-order graph convolution and aggregated attention. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 5766–5770, Hong Kong, China. Association for Computational Linguistics.

Liang Yao, Chengsheng Mao, and Yuan Luo. 2019. Graph convolutional networks for text classification. In *Proceedings of the AAAI conference on artificial intelligence*.

Mo Yu, Wenpeng Yin, Kazi Saidul Hasan, Cicero dos Santos, Bing Xiang, and Bowen Zhou. 2017. Improved neural relation detection for knowledge base question answering. *arXiv preprint arXiv:1704.06194*.

Guodong Zhou, Longhua Qian, and Jianxi Fan. 2010. Tree kernel-based semantic relation extraction with rich syntactic and semantic information. *Information Sciences*, 180(8):1313–1325.

Yichao Zhou, Yu Yan, Rujun Han, J. Harry Caufield, Kai-Wei Chang, Yizhou Sun, Peipei Ping, and Wei Wang. 2021. Clinical Temporal Relation Extraction with Probabilistic Soft Logic Regularization and Global Inference. *Proceedings of the AAAI Conference on Artificial Intelligence*, 35(16):14647–14655.

Learning Hierarchy-Aware Quaternion Knowledge Graph Embeddings with Representing Relations as 3D Rotations

Jinfa Yang, Xianghua Ying*, Yongjie Shi,
Xin Tong, Ruibin Wang, Taiyan Chen, Bowei Xing
Key Laboratory of Machine Perception (MOE)
School of Intelligence Science and Technology, Peking University
{jinfayang, xhying, shiyongjie, xin_tong, robin_wang}@pku.edu.cn,
chenty@stu.pku.edu.cn, 2017xbw@pku.edu.cn

Abstract

Knowledge graph embedding aims to represent entities and relations as low-dimensional vectors, which is an effective way for predicting missing links. It is crucial for knowledge graph embedding models to model and infer various relation patterns, such as symmetry/antisymmetry. However, many existing approaches fail to model semantic hierarchies, which are common in the real world. We propose a new model called HRQE, which represents entities as pure quaternions. The relational embedding consists of two parts: (a) Using unit quaternions to represent the rotation part in 3D space, where the head entities are rotated by the corresponding relations through Hamilton product. (b) Using scale parameters to constrain the modulus of entities to make them have hierarchical distributions. To the best of our knowledge, HRQE is the first model that can encode symmetry/antisymmetry, inversion, composition, multiple relation patterns and learn semantic hierarchies simultaneously. Experimental results demonstrate the effectiveness of HRQE against some of the SOTA methods on four well-established knowledge graph completion benchmarks.

1 Introduction

Knowledge graphs represent human knowledge of the real world as structured triples— (head entity, relation, tail entity) also known as (subject, predicate, object). There are some outstanding knowledge graphs, such as WordNet (Miller, 1995), Freebase (Bollacker et al., 2008), DBpedia (Lehmann et al., 2015). They have gained widespread attention for their successful usage in various applications (*e.g.*, question answering, natural language processing, and recommendation systems). Although millions of entities and billions of facts exist in large-scale knowledge graphs, they still suffer from the incompleteness problem. Therefore, knowledge graph completion (also known as link prediction) which aims to predict missing links among entities based on the known triples has gained growing interest. Learning low-dimensional representations of entities and relations for Knowledge graphs is an effective solution for this task.

Learning knowledge graph embeddings in the complex space $\mathbb{C}$ or quaternion space $\mathbb{H}$ has been proven to be a highly effective inductive bias, largely owing to their ability to model connectivity patterns of the relations. For example, ComplEx (Trouillon et al., 2016), which infers new relational triplets with the asymmetrical Hermitian product can model the symmetry/antisymmetry patterns. RotatE (Sun et al., 2019), which represents entities as points in a complex space and relations as rotations, can model relation patterns including symmetry/antisymmetry, inversion, and composition. DualE (Cao et al., 2021), which combines rotation and translation in dual quaternion space can additionally model the multiple relations pattern. However, many existing models fail to model semantic hierarchies in knowledge graphs.

Semantic hierarchy is a ubiquitous property in knowledge graphs. For instance, WordNet contains the triple [arbor/cassia/palm, hypernym, tree], where "tree" is at a higher level than "arbor/cassia/palm" in the hierarchy. Freebase contains the triple [America, /location/location/contains, California/Los Angeles], where "California/Los Angeles" is at a lower level than "America" in the hierarchy. Although there exists some work that takes the hierarchy structures into account (Xie et al., 2016; Zhang et al., 2020), they usually require additional data to obtain the hierarchy information or cannot model various relation patterns. Therefore, it is still challenging to find an approach that is capable of modeling the various relation patterns and semantic hierarchy simultaneously.

*Corresponding Author

Proceedings of the 29th International Conference on Computational Linguistics, pages 2011–2023
October 12–17, 2022.

Model	Relation Patterns					Hierarchy
	Multiple	Symmetry	Antisymmetry	Inversion	Composition	-Aware
TransE (Bordes et al., 2013)	✗	✗	✓	✓	✓	✗
DistMult (Yang et al., 2015)	✗	✓	✗	✗	✗	✗
ComplEx (Trouillon et al., 2016)	✗	✓	✓	✓	✗	✗
RotatE (Sun et al., 2019)	✗	✓	✓	✓	✓	✗
QuatE (Zhang et al., 2019)	✗	✓	✓	✓	✗	✗
HAKE (Zhang et al., 2020)	✗	✓	✓	✓	✓	✓
DualE (Cao et al., 2021)	✓	✓	✓	✓	✓	✗
QuatRE (Nguyen et al., 2022)	✓	✓	✓	✓	✗	✗
RQE	✓	✓	✓	✓	✓	✗
HRQE	✓	✓	✓	✓	✓	✓

Table 1: The pattern modeling and hierarchy-aware abilities of several models

In this paper, we propose **R**otation Based **Q**uaternion Knowledge Graph **E**mbeddings (RQE) and its **H**ierarchy-aware extension HRQE. More concretely, we represent entities as pure quaternions with three imaginary components **i**, **j** and **k**. The relational embedding consists of two parts: (a) Using unit quaternions to represent the rotation part in 3D space, where the head entities Q_h are rotated by the corresponding relations through Hamilton product. (b) Using scale parameters to constrain the modulus of entities Q_h and Q_t to make them have hierarchical distributions.

To summarize, our contributions are as follows: 1) We propose a new framework called HRQE based on quaternion rotation. 2) To the best of our knowledge, HRQE is the first model that can encode symmetry/antisymmetry, inversion, composition, multiple relation patterns and learn semantic hierarchies simultaneously. 3) We conduct a series of theoretical and empirical analyses to show the strength of HRQE against some of the SOTA methods.

2 Related Work

2.1 Knowledge Graph Embedding Models

Roughly speaking, we can divide knowledge graph embedding models into translational distance models and semantic matching models. The former use distance-based score functions, while the latter use similarity-based ones.

Translational Distance Models. TransE (Bordes et al., 2013) is the most widely used translation distance constraint model. It assumes that entities and relations satisfy $head + relation \approx tail$. However, TransE cannot handle 1-1-N, N-1-1, and N-1-N relations well (Wang et al., 2014). TransH (Wang et al., 2014) is proposed to compensate for the shortcomings of TransE. It projects entities onto relation-specific hyperplanes. TransR (Lin et al., 2015) has a very similar idea to TransH, which introduces relation-specific spatial transformations instead of hyperplanes. TranSparse (Ji et al., 2016) simplifies TransR by forcing the projection matrix to be sparse. Moreover, RotatE (Sun et al., 2019) defines each relation as a rotation from the source entity to the target entity in a complex vector space, which can represent various relation patterns including symmetry/asymmetry, inversion and composition.

Semantic Matching Models. RESCAL (Nickel et al., 2011) is a tensor factorization model which represents each relation as a full-rank matrix and obtains score function by matrix multiplication. DistMult (Yang et al., 2015) simplifies RESCAL by restricting relation matrices to be diagonal. However, Distmult assumes that all relations are symmetric. ComplEx (Trouillon et al., 2016) extends DistMult to complex space, and uses conjugate-transpose to model asymmetric relations. QuatE (Zhang et al., 2019) extends the complex space into the quaternion space with two rotating surfaces. DualE (Cao et al., 2021) combines rotation and translation in dual quaternion space. ConvE (Dettmers et al., 2018) and InteractE (Vashishth et al., 2020) employ convolutional neural networks to build score functions.

2.2 The Ways to Model Hierarchy Structures

Another related problem is how to model hierarchy structures in knowledge graphs. Xie et al. (2016) propose TKRL, which requires additional hierarchical type information for entities. Zhang et al. (2018) use clustering algorithms to model the hierarchical relation structures. Zhang et al.

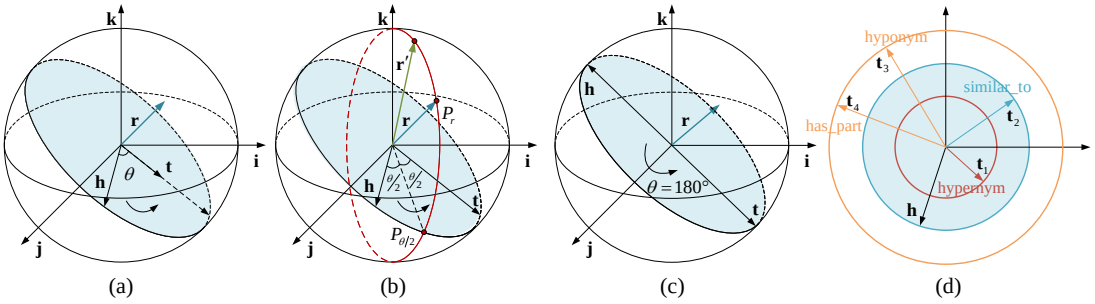

Figure 1: Illustrations of HRQE. (a) HRQE models r as rotation in 3D space. (b) Example of HRQE modeling multiple relations. (c) An example HRQE of modeling symmetric relations with $\theta = 180°$. (d) Example of HRQE modeling different levels of the hierarchy.

(2020) proposed HAKE, which maps entities into the polar coordinate system for hierarchy-aware. Inspired by HAKE, we project entities into 3D space and constrain their rotations and modulus with corresponding relations. In addition to learning the semantic hierarchy, we can better encode various relation patterns such as multiple relations.

3 Quaternion Background

A quaternion $Q \in \mathbb{H}$ is a hyper-complex number consisting of a real and three separate imaginary components (Hamilton, 1844), defined as $Q = a + b\mathbf{i} + c\mathbf{j} + d\mathbf{k}$, where $a, b, c, d \in \mathbb{R}$ and $\mathbf{i}, \mathbf{j}, \mathbf{k}$ are imaginary units. $\mathbf{i}, \mathbf{j}$ and $\mathbf{k}$ are satisfied with Hamilton's rules ($\mathbf{i}^2 = \mathbf{j}^2 = \mathbf{k}^2 = \mathbf{ijk} = -1$). And based on these rules, more non-commutative multiplication rules can be derived, such as $\mathbf{ij} = \mathbf{k}, \mathbf{ji} = -\mathbf{k}, \mathbf{jk} = \mathbf{i}, \mathbf{kj} = -\mathbf{i}, \mathbf{ki} = \mathbf{j}$, and $\mathbf{ik} = -\mathbf{j}$. Some widely used operations of quaternion algebra are introduced as follows:

Quaternion Conjugate: The conjugate of a quaternion Q is defined as $\bar{Q} = a - b\mathbf{i} - c\mathbf{j} - d\mathbf{k}$.

Quaternion Norm: The norm of a quaternion Q is defined as $|Q| = \sqrt{a^2 + b^2 + c^2 + d^2}$.

Pure Quaternion: A pure quaternion $Q \in \mathbb{H}_p$ is defined as a quaternion whose scalar part is zero. Usually, we convert the 3D space point (x, y, z) into a pure quaternion $Q = 0 + x\mathbf{i} + y\mathbf{j} + z\mathbf{k}$, x, y and $z \in \mathbb{R}$ for further quaternion operations.

Quaternion-Inner Product: The quaternion inner product between $Q_1 = a_1 + b_1\mathbf{i} + c_1\mathbf{j} + d_1\mathbf{k}$ and $Q_2 = a_2 + b_2\mathbf{i} + c_2\mathbf{j} + d_2\mathbf{k}$ is obtained by taking the inner products between corresponding scalar and imaginary components and returns a scalar $Q_1 \cdot Q_2 = \langle a_1, a_2 \rangle + \langle b_1, b_2 \rangle + \langle c_1, c_2 \rangle + \langle d_1, d_2 \rangle$.

Quaternion Multiplication (Hamilton Product): The quaternion multiplication is composed of all the standard multiplications of factors in quaternions and returns another quaternion, defined as:

$$\begin{aligned} Q_1 \otimes Q_2 = &(a_1a_2 - b_1b_2 - c_1c_2 - d_1d_2) \\ &+ (a_1b_2 + b_1a_2 + c_1d_2 - d_1c_2)\mathbf{i} \\ &+ (a_1c_2 - b_1d_2 + c_1a_2 + d_1b_2)\mathbf{j} \\ &+ (a_1d_2 + b_1c_2 - c_1b_2 + d_1a_2)\mathbf{k}. \end{aligned} \tag{1}$$

Quaternion Rotation: If $Q_r = \cos\frac{\theta}{2} + \sin\frac{\theta}{2}\mathbf{u}$, where $\mathbf{u} \in \mathbb{R}\mathbf{i} + \mathbb{R}\mathbf{j} + \mathbb{R}\mathbf{k}$ is a unit vector, the result of pure quaternion $Q = 0 + x\mathbf{i} + y\mathbf{j} + z\mathbf{k}$ rotating θ around the axis $\mathbf{u}$ is $Q' = 0 + x'\mathbf{i} + y'\mathbf{j} + z'\mathbf{k}$, then

$$Q' = Q_r \otimes Q \otimes \bar{Q}_r. \tag{2}$$

4 Method

In this section, we introduce our proposed model HRQE. First of all, we elaborate the details of our framework, which mainly consists of two parts: (1) rotate the head entity using the unit relation quaternion and score each triplet with inner product between the rotated head quaternion and the tail quaternion; (2) limit the norm of the head quaternion and the tail quaternion with the relation modulus part. After that, we provide a series of analyses to show the strength of our framework.

Symbol Description. Suppose that we have a knowledge graph $\mathcal{G}$ consisting of N entities and M relations. We formulate the all entity embeddings as a pure quaternion matrix $Q \in \mathbb{H}_p^{N \times k}$, where each row is an embedding vector for a specific entity of dimensionality k, and denote the relation embeddings as rotation part $W \in \mathbb{H}^{M \times k}$ and modulus part $w \in \mathbb{R}^{M \times k}$. Given a triplet (h, r, t), the embedding of head entity h is denoted as $Q_h = \{0 + x_h\mathbf{i} + y_h\mathbf{j} + z_h\mathbf{z} : x_h, y_h, z_h \in \mathbb{R}^k\}$ and the embedding of the tail entity $Q_t = \{0 + x_t\mathbf{i} + y_t\mathbf{j} + z_t\mathbf{z} : x_t, y_t, z_t \in \mathbb{R}^k\}$, where $Q_h, Q_t \in Q$.

Then we denote the relation r as rotation part $W_r = \{a_r + b_r\mathbf{i} + c_r\mathbf{j} + d_r\mathbf{z} : a_r, b_r, c_r, d_r \in \mathbb{R}^k\}$ and modulus part $w_r = \{e_r : e_r \in \mathbb{R}^k\}$, where $W_r \in W, w_r \in w$.

4.1 Hierarchy-Aware Rotation Quaternion Embeddings

The Rotation Part. We first normalize the relation quaternion W_r to a unit quaternion $W_r^{\triangleleft}$ to eliminate the scaling effect by dividing W_r by its norm:

$$W_r^{\triangleleft} = \frac{W_r}{|W_r|} = \frac{a_r + b_r\mathbf{i} + c_r\mathbf{j} + d_r\mathbf{k}}{a_r^2 + b_r^2 + c_r^2 + d_r^2}. \quad (3)$$

Secondly, we rotate the head entity Q_h by doing Hamilton product with $W_r^{\triangleleft}$ and $\bar{W}_r^{\triangleleft}$:

$$Q'_h(r'_h, x'_h, y'_h, z'_h) = W_r^{\triangleleft} \otimes Q_h \otimes \bar{W}_r^{\triangleleft}, \quad (4)$$

where $\otimes$ denotes the element-wise multiplication between two vectors. Then the rotation part scoring function $\phi_r(h, r, t)$ is defined by the quaternion inner product:

$$\phi_r(h, r, t) = Q'_h \cdot Q_t = \langle x'_h, x_t\rangle + \langle y'_h, y_t\rangle + \langle z'_h, z_t\rangle. \quad (5)$$

We separate the rotation part as an independent model RQE, which achieves impressive results (refer to Section 5).

The Modulus Part. As shown in Figure 1a, the rotation part allows the head entity to rotate in 3D space to approximate the tail entity. The modulo length of entities is used to represent the hierarchical distribution of entities. The modulus part of relations is used to measure the hierarchical difference between head and tail entities, which is beneficial for learning hierarchy-aware, see Section 4.2 for details. The modulus part scoring function $\phi_m(h, r, t)$ is defined as:

$$\begin{aligned}\phi_m(h,r,t) &= -\| w_r |Q_h| - |Q_t| \|_1 \\ &= -\left\| w_r \sqrt{x_h^2 + y_h^2 + z_h^2} - \sqrt{x_t^2 + y_t^2 + z_t^2} \right\|_1.\end{aligned} \quad (6)$$

Finally, The scoring function of HRQE is:

$$\phi(h, r, t) = \phi_r(h, r, t) + \lambda\phi_m(h, r, t), \quad (7)$$

where $\lambda \in \mathbb{R}$ is a parameter that learned by the model.

Loss Function. Following Trouillon et al. (2016), We formulate the task as a classification problem and adopt the cross-entropy loss as our loss function. Ω and $\Omega' = \mathcal{E} \times \mathcal{R} \times \mathcal{E} - \Omega$ are used to denote the set of observed triplets and the set of unobserved triplets, respectively. Moreover, we use the ℓ_2 norm with regularization rates λ_1 and λ_2 to regularize Q and W:

$$\begin{aligned}L = &\sum_{r(h,t)\in\Omega\cup\Omega^-} \log(1 + \exp(-Y_{hrt}\phi(h, r, t))) \\ &+ \lambda_1\|Q\|_2^2 + \lambda_2\|W\|_2^2,\end{aligned} \quad (8)$$

where $\Omega^- \subset \Omega'$ with negative sampling strategies such as uniform sampling, bernoulli sampling (Wang et al., 2014), and adversarial sampling (Sun et al., 2019). $Y_{hrt} \in \{-1, 1\}$ represents the corresponding label of the triplet (h, r, t). We optimize the loss function by utilizing Adagrad (Duchi et al., 2011).

4.2 Discussion

In this part, we discuss the theoretical properties of HRQE. We summarize several popular knowledge graph embedding models in Appendix A.5 and definitions of various relation patterns in Appendix A.1.

Capability in Modeling Multiple Relations. For multiple relations such as (A, classmate, B) and (A, neighbor, B) $\in \mathcal{G}$, HRQE allows multiple expressions for the relations when the head and tail entities are fixed. As shown in Figure 1b (here the modulus part is simplified as $w_r = 1$), the red arc passes through **r** vertex P_r and the angle bisector vertex $P_{\theta/2}$. $\mathbf{r}'$ with vertex on the red arc can also make **h** rotate to **t**. That is, $\exists r' \neq r$, (h, r, t) and $(h, r', t) \in \mathcal{G}$.

Capability in Modeling Symmetry /Antisymmetry, Inversion and Composition. The flexibility and representational power of quaternion rotation enable us to model various relation patterns at ease. Since HRQE degenerates to RQE when $\lambda = 0$, we mainly use RQE to discuss. When the rotation angle $\theta = [0°, 180°, 270°, 360°]$, RQE can model the symmetry pattern, and when $W_{r1}^{\triangleleft} = \bar{W}_{r2}^{\triangleleft}$, RQE can model the inversion pattern. The specific lemmas and proofs are as follows:

Lemma 1 *HRQE can infer the symmetry /antisymmetry pattern. (See proof in Appendix A.2)*

Models	WN18					FB15K				
	MR	MRR(%)	Hits(%) @10	Hits(%) @3	Hits(%) @1	MR	MRR	Hits(%) @10	Hits(%) @3	Hits(%) @1
TransE	-	49.5	94.3	88.8	11.3	-	46.3	74.9	57.8	28.7
DistMult	655	79.7	94.6	-	-	42.2	79.8	89.3	-	-
HolE	-	93.8	94.9	94.5	93.0	-	52.4	73.9	75.9	59.9
ComplEx	-	94.1	94.7	94.5	93.6	-	69.2	84.0	75.9	59.9
ConvE	374	94.3	95.6	94.6	93.5	51	65.7	83.1	72.3	55.8
R-GCN+	-	81.9	96.4	92.9	69.7	-	69.6	84.2	76.0	60.1
SimplE	-	94.2	94.7	94.4	93.9	-	72.7	83.8	77.3	66.0
NKGE	366	94.7	95.7	94.9	94.2	56	73	87.1	79.0	65.0
TorusE	-	94.7	95.4	95.0	94.3	-	73.3	83.2	77.1	67.4
RotatE	184	94.7	96.1	95.3	93.8	32	69.9	87.2	78.8	58.5
a-RotatE	309	94.9	95.9	95.2	94.4	40	79.7	88.4	83.0	74.6
QuatE	162	95.0	95.9	95.4	**94.5**	**17**	78.2	**90.0**	83.5	71.1
Rotat3D	214	95.1	96.1	95.3	**94.5**	39	78.9	88.7	83.5	72.8
RQE	117	**95.2**	96.3	**95.6**	**94.5**	23	**81.3**	89.2	**84.3**	**76.6**
HRQE	**72**	**95.2**	**96.5**	**95.6**	94.4	32	79.1	88.7	83.7	73.0

Table 2: Evaluation results on WN18, FB15k datasets. The best scores are in bold, while the second best scores are in underline.

Lemma 2 *HRQE can infer the inversion pattern. (See proof in Appendix A.3)*

Lemma 3 *HRQE can infer the composition pattern. (See proof in Appendix A.4)*

Capability in Modeling Hierarchy Structures. To model the semantic hierarchies of knowledge graphs, a knowledge graph embedding model must be capable of distinguishing entities in the following two categories. (a) Entities at the same level of the hierarchy. (*e.g.* "truck" and "lorry") (b) Entities at different levels of the hierarchy. (*e.g.* "mammal" and "cat") (Zhang et al., 2020). For HRQE, the rotate part can model the entities at the same level of the semantic hierarchy, and the modulus part can model the entities at different levels of the hierarchy. As shown in Figure 1d, entities have hierarchical distribution under different relations, and we simplify it into a 2D space for display. That is, HRQE maps entities into the 3D polar coordinate system, where the angular coordinates and the radial coordinates correspond to the rotate part and the modulus part, respectively.

These results are also summarized in Table 1. We can see that HRQE is the only model that can model and infer all types of relation patterns and hierarchy awareness.

5 Experiments and Results

5.1 Experimental Setup

We evaluate our proposed models on four widely used benchmarks, which are statistically summarized in Table 4.

Datasets: WN18 (Bordes et al., 2013) is extracted from WordNet (Miller, 1995), a database featuring lexical relations and conceptual-semantic between words. The dataset has many inverse relations and the mainly relation patterns are **symmetric/antisymmetric** and **inversion**. WN18RR (Dettmers et al., 2018) is a subset of WN18, with inverse relations removed. The main relation patterns are **symmetric/antisymmetric** and **composition**. In WN18 and WN18RR, most of the triples consist of hyponym and hypernym relations which make them tend to follow a strictly **hierarchical structure**. FB15K (Bordes et al., 2013) is extracted from Freebase (Bollacker et al., 2008), which is a large-scale knowledge graph containing general human knowledge. The key of link prediction on FB15k is to model and infer the **symmetry/antisymmetry** and **inversion** patterns. FB15K-237 (Toutanova and Chen, 2015) is a subset of FB15K, with inverse relations removed. Therefore, the key to link prediction on FB15K-237 boils down to model and infer **symmetrical/antisymmetric** and **composition** patterns.

Evaluation Protocol: For each triple (h, r, t) in the test dataset, we replace either the head entity h or the tail entity t with the total list of the embedding entities. Then, we base the score function to rank the candidate entities in descending order. The filtered setting is used to remove some correct results that appear in the training set or valida-

Models	WN18RR					FB15K-237				
	MR	MRR(%)	Hits(%) @10	Hits(%) @3	Hits(%) @1	MR	MRR	Hits(%) @10	Hits(%) @3	Hits(%) @1
TransE	3384	22.6	50.1	-	-	357	29.4	46.5	-	-
DistMult	5100	43	49	44	39	254	24.1	41.9	26.3	15.5
ComplEx	5261	44	51	46	41	339	24.7	42.8	27.5	15.8
ConvE	4187	43	52	44	40	244	32.5	50.1	35.6	23.7
InteractE	5202	46.3	52.8	-	43.0	172	35.4	53.5	-	26.3
RotatE	3277	47.0	56.5	48.8	42.2	185	29.7	48.0	32.8	20.5
a-RotatE	3340	47.6	57.1	49.2	42.8	177	33.8	53.3	37.5	24.1
QuatE	2314	48.8	58.2	50.8	43.8	87	34.8	55.0	38.2	24.8
ComplEx-N3	-	48.0	57.2	49.5	43.5	-	35.7	54.7	39.2	26.4
TuckER	-	47.0	52.6	48.2	44.3	-	35.8	54.4	39.4	26.6
MURP	-	47.5	55.4	48.7	43.6	-	33.6	52.1	37.0	24.5
RoTH	2293	49.1	58.6	51.1	44.1	-	34.4	53.5	38.0	24.6
Rotat3D	3328	48.9	57.9	50.5	44.2	165	34.7	54.3	38.5	25.0
HAKE	-	49.7	58.2	51.6	45.2	-	34.6	54.2	38.1	25.0
DualE	2270	49.2	58.4	51.3	44.4	91	36.5	55.9	40.0	26.8
QuatRE	1986	49.3	59.2	51.9	43.9	88	36.7	56.3	40.4	26.9
RQE	2043	49.7	59.1	51.7	44.8	**86**	36.9	56.8	40.4	27.3
HRQE	**1198**	**50.5**	**60.1**	**52.4**	**45.4**	89	**37.2**	**56.9**	**40.7**	**27.5**

Table 3: Evaluation results on WN18RR, FB15k-237 datasets. The best scores are in bold, while the second best scores are in underline.

Dataset	#En	#Re	#train	#valid	#test
WN18	40,943	18	141,442	5,000	5,000
WN18RR	40,943	11	86,835	3,034	3,134
FB15K	14,951	1,345	483,142	50,000	59,071
FB15k-237	14,541	237	272,115	17,535	20,466

Table 4: Number of entities, relations, and observed triplets in each split for benchmarks.

tion set but not in test set. We choose Mean Rank (MR), Mean Reciprocal Rank (MRR) and Hits at N (H@N) as the evaluation metrics. MR measures the average rank of all correct entities with a lower value representing better performance. MRR is the average inverse rank for correct entities. Hit@n measures the proportion of correct entities in the top n entities.

Baselines: We compare HRQE with a number of strong baselines. For Translational Distance Models, we reporte TransE (Bordes et al., 2013) , TorusE (Ebisu and Ichise, 2018), RotatE (Sun et al., 2019), Rotat3D (Gao et al., 2020), ROTH (Chami et al., 2020) and HAKE (Zhang et al., 2020); For Semantic Matching Models, we reporte DistMult (Yang et al., 2015), HolE (Nickel et al., 2016), ComplEx (Trouillon et al., 2016), ComplEx-N3 (Lacroix et al., 2018), SimplE (Kazemi and Poole, 2018), TuckER (Balažević et al., 2019), ConvE (Dettmers et al., 2018), R-GCN (Schlichtkrull et al., 2018), NKGE (ConvE based) (Wang et al., 2018), InteractE (Vashishth et al., 2020), QuatE (Zhang et al., 2019), DualE (Cao et al., 2021), and QuatRE (Nguyen et al., 2022).

Implementation Details: The best models are selected by early stopping on the validation set with Hits@10. The ranges of the hyperparameters for the grid search are set as follows: The embedding size k is selected in $\{100, 200, 300, 400, 500\}$. The regularization rates λ_1 and λ_2 are adjusted in $\{0, 0.01, 0.02, 0.03, 0.05, 0.1, 0.2, 0.3, 0.5\}$. The learning rate is chosen from $\{0.01, 0.02, 0.05, 0.1\}$, the number of negative triples sampled per training triple is selected from $\{1, 2, 5, 10\}$. In addition, we create $\{10, 100\}$ batches of training samples for the different datasets. We report RQE and HRQE with type constraints (Krompaß et al., 2015). The training strategies of self-adversarial negative sampling (Sun et al., 2019) and N3 regularization with reciprocal learning (Lacroix et al., 2018) are not used for RQE and HRQE. All hyper-parameters of our models are provided in the appendix A.6, and our code is available at https://github.com/Jinfa/HRQE.

5.2 Results

The empirical results on four datasets are reported in Table 2 and Table 3. HRQE performs extremely competitively compared to the existing state-of-the-art models across all metrics. As a rotation-

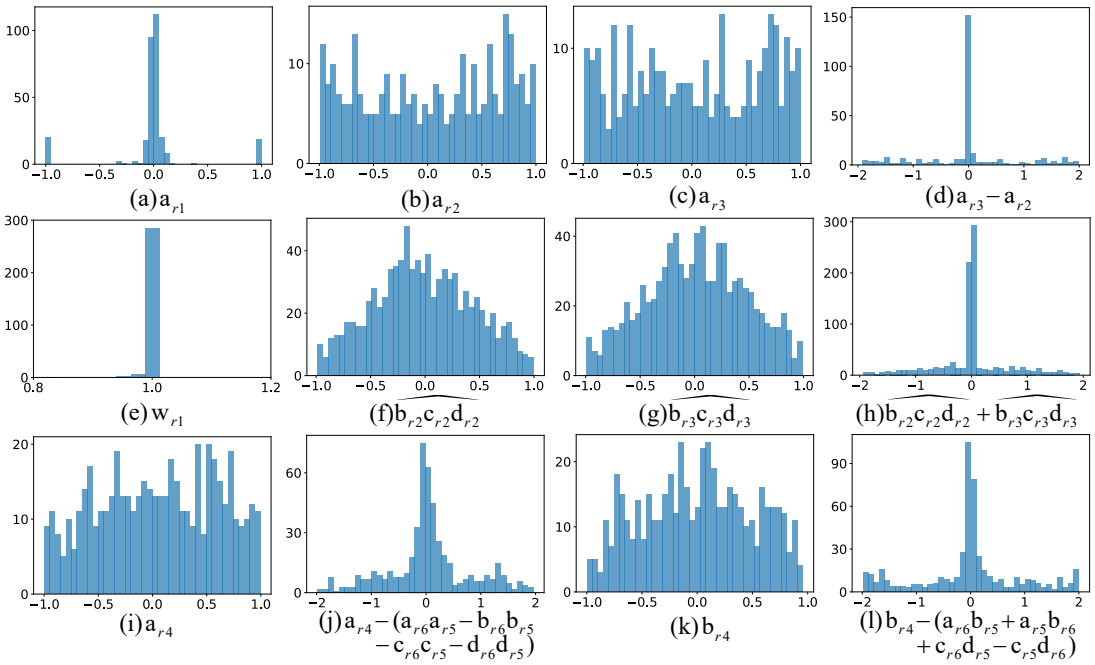

Figure 2: Histograms of relation embeddings for different relation patterns. The corresponding relation is as follows: r_1 is *similar_to*; r_2 is *has_part*; r_3 is *part_of*; r_4 is */location/administrative_division/capital /location/administrative_division_capital_relationship/capital*; r_5 is */location/hud_county_place/place*; r_6 is *base/areas/schema/administrative_area/capital*.

based model, HRQE outperforms the two representative rotation models RoatE, and Roata3D. As a hierarchy-sensitive model, HRQE outperforms the representative hierarchy-aware model HAKE. Also, we outperform other quaternion-valued models such as QuatE, DualE, and QuatRE.

On the WN18 dataset, HRQE outperforms all the baselines on all metrics except Hit@1. HRQE achieves slightly lower results on H@1 than QuatE and RotatE, but surpasses them on the other four metrics, especially on MR with a 56% improvement over QuatE. RQE outperforms HRQE on the FB15K dataset, while the results of them on the validation set are close with 87%. We speculate that excessive inverse relations in FB15K affect the expression of HRQE hierarchy-aware modules. The other recent models a-RotatE, QuatE, and Rotat3D achieve comparable results.

As shown in Table 3, HRQE achieves the best performance over existing state-of-the-art models on the two datasets where trivial inverse relations are removed. On WN18RR in which there are a number of symmetry relations, TransE cannot learn the symmetric relation pattern, so it performs not well. In contrast, the rotation family can achieve good results, and HRQE has further refreshed the performance to achieve the optimal. In addition, HRQE's performance on MR is also impressive, with a 48% improvement over QuatE. WN18 and WN18RR contain hyponym and hypernym relations which make them tend to follow a strictly hierarchical structure, and HRQE's performance demonstrates its ability to learn hierarchically. On FB15K-237, HRQE also achieved better results compared with the previous state-of-the-art models, which shows that HRQE can learn the composition relation pattern better.

Models	Prediction Head (MRR)		Prediction Tail (MRR)	
	1-1-1	1-N-1	1-1-1	1-N-1
TransE	44.3	48.4	45.6	46.9
RotatE	66.8	79.4	71.6	78.6
RQE	**67.8**	**87.7**	**72.5**	**89.5**

Table 5: Evaluation results of multiple relations on FB15k dataset.

5.3 Model Analysis

Analysis on Multiple Relations. In the test set of FB15K, 38121 are single-relation triples (1-1-1), and 20950 are multi-relation triples (1-N-1). To avoid the influence of the modulus part, we choose RQE as the comparison model. Table 5 shows that RQE can better deal with multi-relational triples than TransE and RotatE.

Visualize Some Typical Relation Patterns. To

Models	Prediction Head (Hits@10)				Prediction Tail (Hits@10)			
	1-1-1	1-1-N	N-1-1	N-1-N	1-1-1	1-1-N	N-1-1	N-1-N
QuatE	54.2	**66.4**	38.6	46.9	53.1	25.5	88.3	60.9
QuatRE	58.9	**66.4**	39.3	48.1	59.9	26.8	**88.9**	61.7
HRQE	**63.5**	**66.4**	**42.5**	**48.7**	**63.0**	**28.1**	**88.9**	**62.2**

Table 6: Evaluation results of complex relations on FB15k-237 dataset. The first two rows are taken from (Nguyen et al., 2022)

further verify the learned relation patterns, we visualize some examples. For **symmetry pattern**, HRQE encode with rotation angle $\theta = [0°, 180°, 360°]$ (correspondingly $a_{r1} = \cos\frac{\theta}{2} = [-1, 0, 1]$) and modulo weight $w_r = 1$ which are shown in Figure 2 a and e. For **inversion pattern**, we have $W^{\triangleleft}_{r2} = \bar{W}^{\triangleleft}_{r3}$ (correspondingly $a_{r2} - a_{r3} = 0$ and $\widehat{b_{r2}c_{r2}d_{r2}} + \widehat{b_{r3}c_{r3}d_{r3}} = 0$) , which are shown in Figure 2 b,c,d and f,g,h. For **composition pattern**, we have $W^{\triangleleft}_{r4} = W^{\triangleleft}_{r6} \otimes W^{\triangleleft}_{r5}$. We show the real part and the first imaginary part(correspondingly $a_{r4} - (a_{r6}a_{r5} - b_{r6}b_{r5} - c_{r6}c_{r5} - d_{r6}d_{r5}) = 0$ and $b_{r4} - (a_{r6}b_{r5} + a_{r5}b_{r6} + c_{r6}d_{r5} - c_{r5}d_{r6}) = 0$) in Figure 2 i, j, k and l. Table 7 summarizes the MRR for each relation on WN18RR, confirming the superior representation capability of HRQE in modelling different types of relation.

Analysis on Hierarchy-Aware. We plot the entity embeddings of two models: RQE and HRQE. Their entities are all pure quaternions. For an intuitive display, we project it to a 2D plane and display them in polar coordinates. The radius r of the polar coordinates is quaternion norm $|Q|$, and the angle is twice the angle between the entities and $\mathbf{i} + \mathbf{j} + \mathbf{k}$. Note that we use the logarithmic scale to better display the differences between entity embeddings. As all the moduli have values less than one, after applying the logarithm operation, the larger radii in the figures will actually represent smaller modulus. Compared with the tail entities, the head entities in Figure 3 a, b, and c are at lower levels, similar levels, and higher levels in the semantic hierarchy, respectively. We can see that there exist clear hierarchies in HRQE, which demonstrates that the modulus part in HRQE can help effectively model the semantic hierarchies.

Analysis on Complex Relations. We also conduct further investigation on the performance of HRQE on complex relations: 1-1-N, N-1-1, and N-1-N relations. We compare with QuatE and QuatRE (Nguyen et al., 2022). QuatRE adds two additional relational quaternions and quaternion mul-

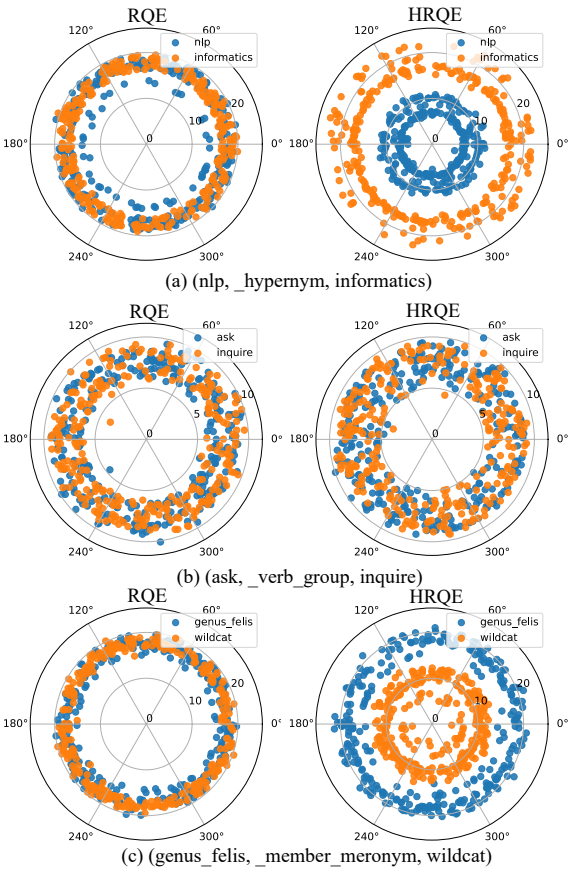

Figure 3: Visualization of the embeddings of several entity pairs from WN18RR dataset.

Relation Name	RotatE	QuatE	HRQE
hypernym	14.8	17.3	**19.3**
derivationally_related_form	94.7	95.3	**95.7**
instance_hypernym	31.8	36.4	**38.1**
also_see	58.5	62.9	**68.3**
member_meronym	23.2	23.2	**27.2**
synset_domain_topic_of	34.1	46.8	**48.4**
has_part	18.4	23.3	**24.3**
member_of_domain_usage	31.8	**44.1**	42.6
member_of_domain_region	20.0	19.3	**27.1**
verb_group	**94.3**	92.4	91.1
similar_to	**100.0**	**100.0**	**100.0**

Table 7: MRR for the models tested on each relation of WN18RR.

tiplies with the head and tail entities to improve QuatE's ability for handling complex relations. Table 6 shows the MRR and H@10 scores for predicting the head entities and then the tail entities with respect to each relation category on FB15k-237, wherein our HRQE outperforms QuatE and QuatRE on these relation categories.

6 Conclusion

To model various relation patterns and semantic hierarchies in knowledge graphs, we propose a novel knowledge graph embedding model HRQE, which maps entities into 3D space with rotation and modulo constraints. Empirical experimental evaluations on benchmark datasets show that our proposed HRQE significantly outperforms several existing state-of-the-art methods. Further investigation shows that HRQE is capable of modeling relations with various relation patterns and modeling entities at both different levels and the same levels in the semantic hierarchies.

Acknowledgement

This work was supported in part by National Key R&D Program of China Grand No. 2020YFB1708002, and NNSFC Grant No. 61971008.

References

Ivana Balažević, Carl Allen, and Timothy Hospedales. 2019. Tucker: Tensor factorization for knowledge graph completion. In *Proc. of Conference on Empirical Methods in Natural Language Processing and International Joint Conference on Natural Language Processing*, pages 5185–5194.

Kurt Bollacker, Colin Evans, Praveen Paritosh, Tim Sturge, and Jamie Taylor. 2008. Freebase: a collaboratively created graph database for structuring human knowledge. In *Proc. of ACM International Conference on Management of Data*, pages 1247–1250.

Antoine Bordes, Nicolas Usunier, Alberto Garcia-Duran, Jason Weston, and Oksana Yakhnenko. 2013. Translating embeddings for modeling multi-relational data. In *Proc. of Annual Conference on Neural Information Processing Systems*, pages 1–9.

Zongsheng Cao, Qianqian Xu, Zhiyong Yang, Xiaochun Cao, and Qingming Huang. 2021. Dual quaternion knowledge graph embeddings. In *Proc. of AAAI Conference on Artificial Intelligence*, volume 35, pages 6894–6902.

Ines Chami, Adva Wolf, Da-Cheng Juan, Frederic Sala, Sujith Ravi, and Christopher Ré. 2020. Low-dimensional hyperbolic knowledge graph embeddings. In *Proc. of Annual Meeting of the Association for Computational Linguistics*, pages 6901–6914.

Tim Dettmers, Pasquale Minervini, Pontus Stenetorp, and Sebastian Riedel. 2018. Convolutional 2d knowledge graph embeddings. In *Proc. of AAAI Conference on Artificial Intelligence*, volume 32, pages 1811–1818.

John Duchi, Elad Hazan, and Yoram Singer. 2011. Adaptive subgradient methods for online learning and stochastic optimization. *Journal of machine learning research*, 12(7).

Takuma Ebisu and Ryutaro Ichise. 2018. Toruse: Knowledge graph embedding on a lie group. In *Proc. of AAAI Conference on Artificial Intelligence*.

Chang Gao, Chengjie Sun, Lili Shan, Lei Lin, and Mingjiang Wang. 2020. Rotate3d: Representing relations as rotations in three-dimensional space for knowledge graph embedding. In *Proc. of ACM International Conference on Information & Knowledge Management*, pages 385–394.

William Rowan Hamilton. 1844. Lxxviii. on quaternions; or on a new system of imaginaries in algebra: To the editors of the philosophical magazine and journal. *The London, Edinburgh, and Dublin Philosophical Magazine and Journal of Science*, 25(169):489–495.

Guoliang Ji, Kang Liu, Shizhu He, and Jun Zhao. 2016. Knowledge graph completion with adaptive sparse transfer matrix. In *Proc. of AAAI Conference on Artificial Intelligence*, pages 985–991.

Seyed Mehran Kazemi and David Poole. 2018. Simple embedding for link prediction in knowledge graphs. *Proc. of Annual Conference on Neural Information Processing Systems*, 31.

Denis Krompaß, Stephan Baier, and Volker Tresp. 2015. Type-constrained representation learning in knowledge graphs. In *Proc. of International semantic web conference*, pages 640–655.

Timothée Lacroix, Nicolas Usunier, and Guillaume Obozinski. 2018. Canonical tensor decomposition for knowledge base completion. In *Proc. of International Conference on Machine Learning*, pages 2863–2872.

Jens Lehmann, Robert Isele, Max Jakob, Anja Jentzsch, Dimitris Kontokostas, Pablo N Mendes, Sebastian Hellmann, Mohamed Morsey, Patrick Van Kleef, Sören Auer, et al. 2015. Dbpedia–a large-scale, multilingual knowledge base extracted from wikipedia. *Semantic web*, 6(2):167–195.

Yankai Lin, Zhiyuan Liu, Maosong Sun, Yang Liu, and Xuan Zhu. 2015. Learning entity and relation embeddings for knowledge graph completion. In *Proc.*

of AAAI Conference on Artificial Intelligence, volume 29, pages 2181–2187.

George A Miller. 1995. Wordnet: a lexical database for english. *Communications of the ACM*, 38(11):39–41.

Dai Quoc Nguyen, Thanh Vu, Tu Dinh Nguyen, and Dinh Phung. 2022. Quatre: Relation-aware quaternions for knowledge graph embeddings. In *Companion Proc. of International World Wide Web Conferences 2022*.

Maximilian Nickel, Lorenzo Rosasco, and Tomaso Poggio. 2016. Holographic embeddings of knowledge graphs. In *Proc. of AAAI Conference on Artificial Intelligence*, volume 30, pages 1955–1961.

Maximilian Nickel, Volker Tresp, and Hans-Peter Kriegel. 2011. A three-way model for collective learning on multi-relational data. In *Proc. of International Conference on Machine Learning*, volume 11, pages 809–816.

Michael Schlichtkrull, Thomas N Kipf, Peter Bloem, Rianne Van Den Berg, Ivan Titov, and Max Welling. 2018. Modeling relational data with graph convolutional networks. In *Proc. of European Semantic Web Conference*, pages 593–607.

Zhiqing Sun, Zhi-Hong Deng, Jian-Yun Nie, and Jian Tang. 2019. Rotate: Knowledge graph embedding by relational rotation in complex space. In *Proc. of International Conference on Learning Representations*, pages 1–18.

Kristina Toutanova and Danqi Chen. 2015. Observed versus latent features for knowledge base and text inference. In *Proc. of Workshop on Continuous Vector Space Models and their Compositionality*, pages 57–66.

Théo Trouillon, Johannes Welbl, Sebastian Riedel, Éric Gaussier, and Guillaume Bouchard. 2016. Complex embeddings for simple link prediction. In *Proc. of International Conference on Machine Learning*, volume 48, pages 2071–2080.

Shikhar Vashishth, Soumya Sanyal, Vikram Nitin, Nilesh Agrawal, and Partha Talukdar. 2020. Interacte: Improving convolution-based knowledge graph embeddings by increasing feature interactions. In *Proc. of AAAI Conference on Artificial Intelligence*, volume 34, pages 3009–3016.

Kai Wang, Yu Liu, Xiujuan Xu, and Dan Lin. 2018. Knowledge graph embedding with entity neighbors and deep memory network. *arXiv preprint arXiv:1808.03752*.

Zhen Wang, Jianwen Zhang, Jianlin Feng, and Zheng Chen. 2014. Knowledge graph embedding by translating on hyperplanes. In *Proc. of AAAI Conference on Artificial Intelligence*, volume 28, pages 1112–1119.

Ruobing Xie, Zhiyuan Liu, Maosong Sun, et al. 2016. Representation learning of knowledge graphs with hierarchical types. In *Proc. of International Joint Conference on Artificial Intelligence*, volume 2016, pages 2965–2971.

Bishan Yang, Wen-tau Yih, Xiaodong He, Jianfeng Gao, and Li Deng. 2015. Embedding entities and relations for learning and inference in knowledge bases. In *Proc. of International Conference on Learning Representations*, pages 1–12.

Shuai Zhang, Yi Tay, Lina Yao, and Qi Liu. 2019. Quaternion knowledge graph embeddings. In *Proc. of Annual Conference on Neural Information Processing Systems*, volume 32, pages 2735–2745.

Zhanqiu Zhang, Jianyu Cai, Yongdong Zhang, and Jie Wang. 2020. Learning hierarchy-aware knowledge graph embeddings for link prediction. In *Proc. of AAAI Conference on Artificial Intelligence*, pages 3065–3072.

Zhao Zhang, Fuzhen Zhuang, Meng Qu, Fen Lin, and Qing He. 2018. Knowledge graph embedding with hierarchical relation structure. In *Proc. of Conference on Empirical Methods in Natural Language Processing*, pages 3198–3207.

A Appendix

A.1 Definitions of Different Relation Patterns

Definition 1 *Relations r_i are multiple if $\forall i \in 0, ..., m$, (h, r, t) can hold in knowledge graphs simultaneously. A clause with such form is a multiple relations pattern.*

Definition 2 *A relation r is symmetric (antisymmetry) if $\forall x, y$*

$$r(x, y) \Rightarrow r(y, x) \ (r(x, y) \Rightarrow \neg r(y, x)).$$

A clause with such form is a symmetry (antisymmetry) pattern.

Definition 3 *Relation r_1 is inverse to relation r_2 if $\forall x, y$*

$$r_2(x, y) \Rightarrow r_1(y, x).$$

A clause with such form is an inversion pattern.

Definition 4 *Relation r_1 is composed of relation r_2 and relation r_3 if $\forall x, y, z$*

$$r_2(x, y) \wedge r_3(y, z) \Rightarrow r_1(x, z).$$

A clause with such form is a composition pattern.

Definition 5 *For each relation r, we compute average number of tails per head (tphr) and average number of heads per tail (hptr). If tphr < 1.5 and hptr < 1.5, r is treated as 1-to-1; if tphr > 1.5 and hptr > 1.5, r is treated as a N-to-N; if tphr > 1.5 and hptr < 1.5, r is treated as 1-to-N. Clauses with such form are complex relations.*

A.2 Proof of Lemma 1

Proof of symmetry pattern. When $\theta = [0°, 180°, 360°]$, HRQE can represent a symmetric relationship, then $W_r^{\triangleleft} = \cos\frac{\theta}{2} + \sin\frac{\theta}{2}(q\mathbf{i} + u\mathbf{j} + v\mathbf{k})$, we need to prove:

$$W_r^{\triangleleft} \otimes Q_h \otimes \bar{W}_r^{\triangleleft} \cdot Q_t = W_r^{\triangleleft} \otimes Q_t \otimes \bar{W}_r^{\triangleleft} \cdot Q_h \quad (9)$$

For $\theta = [0°, 360°]$, we have $W_r^{\triangleleft} = \pm 1$. Firstly, we expand the left term:

$$\begin{aligned} &W_r^{\triangleleft} \otimes Q_h \otimes \bar{W}_r^{\triangleleft} \cdot Q_t \\ &= Q_h \cdot Q_t \\ &= \langle x_h, x_t \rangle + \langle y_h, y_t \rangle + \langle z_h, z_t \rangle. \end{aligned} \quad (10)$$

We then expand the right term:

$$\begin{aligned} &W_r^{\triangleleft} \otimes Q_t \otimes \bar{W}_r^{\triangleleft} \cdot Q_h \\ &= Q_t \cdot Q_h \\ &= \langle x_h, x_t \rangle + \langle y_h, y_t \rangle + \langle z_h, z_t \rangle. \end{aligned} \quad (11)$$

We can easily see that those two terms are equal.

For $\theta = [180°]$, we have $W_r^{\triangleleft} = q\mathbf{i} + u\mathbf{j} + v\mathbf{k}$. Firstly, we expand the left term:

$$\begin{aligned} &W_r^{\triangleleft} \otimes Q_h \otimes \bar{W}_r^{\triangleleft} \cdot Q_t \\ &= [((qq - uu - vv)x_h + 2(qu)y_h + 2(qv)z_h)\mathbf{i} \\ &+ (2(qu)x_h + (qq + uu - vv)y_h + 2(uv)z_h)\mathbf{j} \\ &+ (2(qv)x_h + 2(uv)y_h + (qq - uu + vv)z_h)\mathbf{k}] \\ &\cdot (x_t\mathbf{i} + y_t\mathbf{j} + z_t\mathbf{k}) \\ &= \langle x_h, (qq - uu - vv), x_t \rangle + \langle x_h, 2(qu), y_t \rangle \\ &+ \langle x_h, 2(qv), z_t \rangle + \langle y_h, 2(qu), x_t \rangle \\ &+ \langle y_h, (qq + uu - vv), y_t \rangle + \langle y_h, 2(uv), z_t \rangle \\ &+ \langle z_h, 2(qv), x_t \rangle + \langle z_h, 2(uv), y_t \rangle \\ &+ \langle z_h, (qq - uu + vv), z_t \rangle. \end{aligned} \quad (12)$$

We then expand the right term:

$$\begin{aligned} &W_r^{\triangleleft} \otimes Q_t \otimes \bar{W}_r^{\triangleleft} \cdot Q_h \\ &= [((qq - uu - vv)x_t + 2(qu)y_t + 2(qv)z_t)\mathbf{i} \\ &+ (2(qu)x_t + (qq + uu - vv)y_t + 2(uv)z_t)\mathbf{j} \\ &+ (2(qv)x_t + 2(uv)y_t + (qq - uu + vv)z_t)\mathbf{k}] \\ &\cdot (x_h\mathbf{i} + y_h\mathbf{j} + z_h\mathbf{k}) \\ &= \langle x_t, (qq - uu - vv), x_h \rangle + \langle x_t, 2(qu), y_h \rangle \\ &+ \langle x_t, 2(qv), z_h \rangle + \langle y_t, 2(qu), x_h \rangle \\ &+ \langle y_t, (qq + uu - vv), y_h \rangle + \langle y_t, 2(uv), z_h \rangle \\ &+ \langle z_t, 2(qv), x_h \rangle + \langle z_t, 2(uv), y_h \rangle \\ &+ \langle z_t, (qq - uu + vv), z_h \rangle. \end{aligned} \quad (13)$$

We can easily see that those two terms are equal.

Proof of antisymmetry pattern. In order to prove the antisymmetry pattern, we need to prove the following inequality when $\theta \neq [0°, 180°, 270°, 360°]$:

$$W_r^{\triangleleft} \otimes Q_h \otimes \bar{W}_r^{\triangleleft} \cdot Q_t \neq W_r^{\triangleleft} \otimes Q_t \otimes \bar{W}_r^{\triangleleft} \cdot Q_h \quad (14)$$

Firstly, we expand the left term:

$$\begin{aligned} &W_r^{\triangleleft} \otimes Q_h \otimes \bar{W}_r^{\triangleleft} \cdot Q_t \\ &= [((pp + qq - uu - vv)x_h + 2(-pv + qu)y_h \\ &+ 2(pu + qv)z_h)\mathbf{i} + (2(pv + qu)x_h \\ &+ (pp - qq + uu - vv)y_h + 2(-pq + uv)z_h)\mathbf{j} \\ &+ (2(-pu + qv)x_h + 2(pq + uv)y_h \\ &+ (pp - qq - uu + vv)z_h)\mathbf{k}] \\ &\cdot (x_t\mathbf{i} + y_t\mathbf{j} + z_t\mathbf{k}) \\ &= \langle x_h, (pp + qq - uu - vv), x_t \rangle \\ &+ \langle x_h, 2(pv + qu), y_t \rangle + \langle x_h, 2(-pu + qv), z_t \rangle \\ &+ \langle y_h, 2(-pv + qu), x_t \rangle \\ &+ \langle y_h, (pp - qq + uu - vv), y_t \rangle \\ &+ \langle y_h, 2(pq + uv), z_t \rangle + \langle z_h, 2(pu + qv), x_t \rangle \\ &+ \langle z_h, 2(-pq + uv), y_t \rangle \\ &+ \langle z_h, (pp - qq - uu + vv), z_t \rangle. \end{aligned} \quad (15)$$

We then expand the right term:

Model	Score Function $f_r(Q_h, Q_t)$	Parameters	O_{time}
TransE (Bordes et al., 2013)	$-\|(Q_h + W_r) - Q_t\|_{1/2}$	$Q_h, W_r, Q_t \in \mathbb{R}^k$	$O(k)$
Hole (Nickel et al., 2016)	$\langle W_r, Q_h \star Q_t \rangle$	$Q_h, W_r, Q_t \in \mathbb{R}^k$	$O(k\log(k))$
DistMult (Yang et al., 2015)	$\langle W_r, Q_h, Q_t \rangle$	$Q_h, W_r, Q_t \in \mathbb{R}^k$	$O(k)$
ComplEx (Trouillon et al., 2016)	$\text{Re}\left(\langle W_r, Q_h, \bar{Q}_t \rangle\right)$	$Q_h, W_r, Q_t \in \mathbb{C}^k$	$O(k)$
RotatE (Sun et al., 2019)	$-\|Q_h \circ W_r - Q_t\|_2$	$Q_h, W_r, Q_t \in \mathbb{C}^k, \|W_{ri}\| = 1$	$O(k)$
Rotate3D (Gao et al., 2020)	$-\|Q_h \odot W_r \times B_r - Q_t\|_2$	$Q_h, W_r, Q_t \in \mathbb{H}^k, B_r \in \mathbb{R}^k, \|W_{ri}\| = 1$	$O(k)$
QuatE (Zhang et al., 2019)	$Q_h \otimes W_r^{\triangleleft} \cdot Q_t$	$Q_h, W_r, Q_t \in \mathbb{H}^k$	$O(k)$
DualE (Cao et al., 2021)	$\left\langle Q_h \underline{\otimes} W_r^{\diamond}, Q_t \right\rangle$	$Q_h, W_r, Q_t \in \mathbb{H}_d^k$	$O(k)$
HAKE (Zhang et al., 2020)	$-\|Q_{hm} \circ W_m - Q_{tm}\|_2$ $-\lambda\|\sin((Q_{hp} + W_p - Q_{tp})/2)\|_1$	$Q_{hm}, Q_{tm} \in \mathbb{R}^k, W_m \in \mathbb{R}_+^k$ $Q_{hp}, W_p, Q_{tp} \in [0, \pi)^k, \lambda \in \mathbb{R}$	$O(k)$
RQE	$W_r^{\triangleleft} \otimes Q_h \otimes \bar{W}_r^{\triangleleft} \cdot Q_t$	$W_r \in \mathbb{H}^k, Q_h, Q_t \in \mathbb{H}_p^k$	$O(k)$
HRQE	$W_r^{\triangleleft} \otimes Q_h \otimes \bar{W}_r^{\triangleleft} \cdot Q_t$ $-\lambda\|w_r\|Q_h\| - \|Q_t\|\|_1$	$W_r \in \mathbb{H}^k, Q_h, Q_t \in \mathbb{H}_p^k$ $w_r \in \mathbb{R}^k, \lambda \in \mathbb{R}$	$O(k)$

Table 8: Scoring functions of state-of-the-art knowledge graph embedding models, along with their parameters, time complexity."$\star$" denotes the circular correlation operation; "$\circ$" denotes Hadmard (or element-wise) product; "$\otimes$" denotes Hamilton product.

$$
\begin{aligned}
&W_r^{\triangleleft} \otimes Q_t \otimes \bar{W}_r^{\triangleleft} \cdot Q_h \\
&= [((pp + qq - uu - vv)x_t + 2(-pv + qu)y_t \\
&+ 2(pu + qv)z_t)\mathbf{i} + (2(pv + qu)x_t \\
&+ (pp - qq + uu - vv)y_t + 2(-pq + uv)z_t)\mathbf{j} \\
&+ (2(-pu + qv)x_t + 2(pq + uv)y_t \\
&+ (pp - qq - uu + vv)z_t)\mathbf{k}] \\
&\cdot (x_h\mathbf{i} + y_h\mathbf{j} + z_h\mathbf{k}) \\
&= \langle x_t, (pp + qq - uu - vv), x_h \rangle \\
&+ \langle x_t, 2(pv + qu), y_h \rangle + \langle x_t, 2(-pu + qv), z_h \rangle \\
&+ \langle y_t, 2(-pv + qu), x_h \rangle \\
&+ \langle y_t, (pp - qq + uu - vv), y_h \rangle \\
&+ \langle y_t, 2(pq + uv), z_h \rangle + \langle z_t, 2(pu + qv), x_h \rangle \\
&+ \langle z_t, 2(-pq + uv), y_h \rangle \\
&+ \langle z_t, (pp - qq - uu + vv), z_h \rangle.
\end{aligned} \tag{16}
$$

We can easily see that those two terms are not equal as the signs for some terms are not the same.

A.3 Proof of Lemma 2

Proof of inversion pattern. To prove the inversion pattern, we need to prove that:

$$
W_r^{\triangleleft} \otimes Q_h \otimes \bar{W}_r^{\triangleleft} \cdot Q_t = \bar{W}_r^{\triangleleft} \otimes Q_t \otimes W_r^{\triangleleft} \cdot Q_h \tag{17}
$$

We expand the right term:

$$
\begin{aligned}
&\bar{W}_r^{\triangleleft} \otimes Q_t \otimes W_r^{\triangleleft} \cdot Q_h \\
&= [((pp + qq - uu - vv)x_t + 2(pv + qu)y_t \\
&+ 2(-pu + qv)z_t)\mathbf{i} + (2(-pv + qu)x_t \\
&+ (pp - qq + uu - vv)y_t + 2(pq + uv)z_t)\mathbf{j} \\
&+ (2(pu + qv)x_t + 2(-pq + uv)y_t \\
&+ (pp - qq - uu + vv)z_t)\mathbf{k}] \\
&\cdot (x_h\mathbf{i} + y_h\mathbf{j} + z_h\mathbf{k}) \\
&= \langle x_t, (pp + qq - uu - vv), x_h \rangle \\
&+ \langle x_t, 2(-pv + qu), y_h \rangle + \langle x_t, 2(pu + qv), z_h \rangle \\
&+ \langle y_t, 2(pv + qu), x_h \rangle \\
&+ \langle y_t, (pp - qq + uu - vv), y_h \rangle \\
&+ \langle y_t, 2(-pq + uv), z_h \rangle + \langle z_t, 2(-pu + qv), x_h \rangle \\
&+ \langle z_t, 2(pq + uv), y_h \rangle \\
&+ \langle z_t, (pp - qq - uu + vv), z_h \rangle.
\end{aligned} \tag{18}
$$

We can easily check the equality of these two terms.

A.4 Proof of Lemma 3

Proof of composition relation. For composition relations we can get that:

$$
\begin{aligned}
&W_{r3}^{\triangleleft} \otimes (W_{r2}^{\triangleleft} \otimes Q_h \otimes \bar{W}_{r2}^{\triangleleft}) \otimes \bar{W}_{r3}^{\triangleleft} \cdot Q_t \\
&= (W_{r3}^{\triangleleft} \otimes W_{r2}^{\triangleleft}) \otimes Q_h \otimes (\bar{W}_{r2}^{\triangleleft} \otimes \bar{W}_{r3}^{\triangleleft}) \cdot Q_t \\
&= W_{r1}^{\triangleleft} \otimes Q_h \otimes \bar{W}_{r1}^{\triangleleft} \cdot Q_t.
\end{aligned} \tag{19}
$$

A.5 Summary of Several Popular Knowledge Graph Embedding Models

Table 8 summarizes several popular knowledge graph embedding models, including scoring func-

tions, parameters, and time complexities. TransE, HolE, and DistMult use Euclidean embeddings, while ComplEx and RotatE operate in the complex space. QuatE, DualE (dual quaternion) and our models operate in the quaternion space. HAKE and our model HRQE can learn hierarchy-aware in knowledge graphs.

A.6 Parameter Settings

We list the best hyperparameter settings of RQE and HRQE w.r.t. the validation dataset on several benchmarks in Table 9 and Table 10.

Dataset	nB	k	λ_1	λ_2	neg
WN18	10	300	0.03	0.0	10
FB15K	100	400	0.05	0.0	10
WN18RR	10	300	0.3	0.3	2
FB15K237	100	500	0.5	0.01	10

Table 9: Hyperparameters for RQE

Dataset	nB	k	λ_1	λ_2	neg
WN18	10	300	0.05	0.01	10
FB15K	100	500	0.03	0.0	10
WN18RR	10	300	0.3	0.01	1
FB15K237	100	500	0.5	0.01	10

Table 10: Hyperparameters for HRQE

Two Languages Are Better Than One: Bilingual Enhancement For Chinese Named Entity Recognition

Jinzhong Ning[1], Zhihao Yang[1]*, Zhizheng Wang[1],
Yuanyuan Sun[1], Hongfei LIN[1], Jian Wang[1]
School of Computer Science and Technology, Dalian University of Technology
Jinzhong_Ning@mail.dlut.edu.cn, wzz0727@gmail.com
{yangzh,syuan,hflin,wangjian}@dlut.edu.cn

Abstract

Chinese Named Entity Recognition (NER) has continued to attract research attention. However, most existing studies only explore the internal features of the Chinese language but neglect other lingual modal features. Actually, as another modal knowledge of the Chinese language, English contains rich prompts about entities that can potentially be applied to improve the performance of Chinese NER. Therefore, in this study, we explore the bilingual enhancement for Chinese NER and propose a unified bilingual interaction module called the **A**dapted **C**ross-**T**ransformers with Global **S**parse Attention (**ACT-S**) to capture the interaction of bilingual information. We utilize a model built upon several different ACT-Ss to integrate the rich English information into the Chinese representation. Moreover, our model can learn the interaction of information between bilinguals (inter-features) and the dependency information within Chinese (intra-features). Compared with existing Chinese NER methods, our proposed model can better handle entities with complex structures. The English text that enhances the model is automatically generated by machine translation, avoiding high labour costs. Experimental results on four well-known benchmark datasets demonstrate the effectiveness and robustness of our proposed model.

1 Introduction

"One language sets you in a corridor for life. Two languages open every door along the way."

—Frank Smith, Psycholinguist

Named entity recognition (NER) is the task of determining spans and semantic categories of named entities such as organization (*ORG*), person (*PER*) and location (*LOC*) in given free text. As the cornerstone of a wide range of natural language processing tasks, NER plays an essential role in many downstream tasks, such as relation extraction (Zelenko et al., 2003) and question answer (Diefenbach et al., 2018).

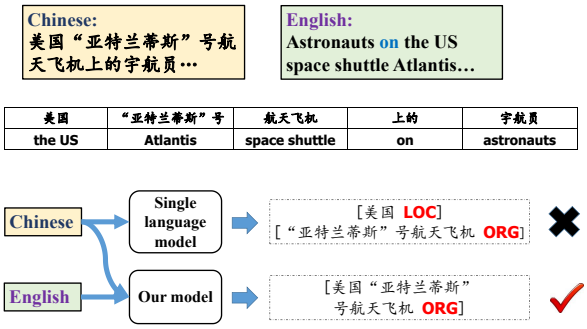

Figure 1: An example of Chinese NER via bilingual enhancement. The entity types of named entities are highlighted.

Compared with English NER, Chinese NER meets a series of challenges caused by the characteristics of Chinese. Aside from the lack of natural word boundary information, Chinese named entities usually vary significantly in length and have complex compositional structures (Dong et al., 2016). At the same time, the annotated data of Chinese NER is relatively scarce, and it is difficult and costly to annotate the data manually (Liu et al., 2022). Hence, without more annotated data, it is a promising approach to improve Chinese NER by leveraging external information resources, which has attracted more and more research attention.

One way to utilize external resources is to perform Chinese NER with bilingual constraints. Previous works (Che et al., 2013; Wang et al., 2013) have demonstrated that the joint use of bilingual information in Chinese and English can significantly improve performance in the Chinese NER task. In addition, incorporating vocabulary knowledge has also become a promising solution(Zhang and Yang, 2018; Li et al., 2020; Mengge et al., 2020). What's more, several studies were proposed to exploit information from other modalities to supplement the representation of Chinese text(Meng et al., 2019;

* Corresponding Author

Proceedings of the 29th International Conference on Computational Linguistics, pages 2024–2033
October 12–17, 2022.

Sun et al., 2021b; Wu et al., 2021; Sui et al., 2021).

However, despite the success of the above-mentioned methods in Chinese NER by introducing external information, these methods still have the following limitations. First, the external resources utilized by these methods are mainly obtained manually, which increases the cost significantly. Second, existing methods for boosting NER with bilingual constraints (Che et al., 2013; Wang et al., 2013) rely on bilingual word alignment information and both Chinese and English sentences need to be manually annotated, which limits its usage. Third, as the state-of-the-art approaches based on deep neural networks for Chinese NER, lexical enhancement methods and multimodal methods still fail to effectively handle entities with complex composition structures, which, however, are frequently observed in Chinese NER tasks. The example in Figure 1 illustrates one of such dilemmas in Chinese NER. In this example, due to the complex component structure, the ORG entity "美国'亚特兰蒂斯'号航天飞机(The US space shuttle Atlantis)" tends to be incorrectly labeled by the NER model as a LOC entity "美国(USA)" and an ORG entity "'亚特兰蒂斯'号航天飞机(Space shuttle Atlantis)". However, it is encouraging that the clues of the English expression, such as the preposition "on", will potentially alleviate this type of incorrect labeling that often occurs in Chinese NER tasks.

To address the above issues, in this work, we propose to boost the performance of Chinese NER with the unlabelled English text translated from the corresponding Chinese text. English texts are automatically generated through the publicly available neural machine translation API without any extra human labor all the way through. Besides, considering the Chinese texts and corresponding English translations as two different modalities, we perform bilingual enhancement of Chinese NER with multimodal NER approach. Furthermore, based on the fact that a word in the text will only be strongly correlated with a small fraction of words in the translated text, we propose a bilingual interaction enhancement model based on **A**dapted **C**ross-**T**ransformers with Global **S**parse Attention (short for **ACT-S**). The interaction of bilingual information as well as intra-linguistic interaction are taken into account in the our model.

The primary contributions of this work can be summarized as follows: (1) We improve the performance of Chinese NER by bilingual enhancement, based on the unlabeled translated English text automatically generated using the Neural Machine Translation API. To the best of our knowledge, this is the first end-to-end NER method that effectively exploits bilingual information. (2) We further propose the neural module called Adapted Cross-Transformers with Global Sparse Attention to simultaneously model bilingual interactions and inter-lingual interactions. So far as we know, it is the first attempt to use global sparse attention mechanisms for multimodal information interaction. (3) Experimental results on four Chinese NER datasets show that our proposed model achieves superior performance to other strong baseline models.

2 Related Work

2.1 Chinese NER with lexicon enhancement

In Chinese NER, a series of recent works focus on introducing lexical boundaries and semantic information by word matching. Zhang *et al.* (2018) proposed to introduce semantic and boundary information of the lexicon through the lattice structure in LSTM. Afterwards, some CNN-based NER lexical enhancement methods, such as LR-CNN (Gui et al., 2019a), were proposed. Graph neural networks have also been applied to Chinese NER word enhancement tasks, a typical one of which is LGN (Gui et al., 2019b). And Transformer-like encoders fusing lexical information are also used for Chinese NER tasks, including PLTE (Mengge et al., 2020) and FLAT (Li et al., 2020). In addition, some work (Ma et al., 2020; Liu et al., 2021) has proposed to fuse lexical information into the word embedding representation instead of integrating word information into the model encoder. However, the lexicon enhanced Chinese NER approach still cannot effectively deal with entities with complex composition structures. And our approach utilizes bilingual clues that can alleviate this problem in Chinese NER.

2.2 Multimodal NER

In recent years, with the development of multimodal information processing technology, the multimodal NER has emerged. In the field of English information extraction, existing multimodal NER works(Yu et al., 2020; Sun et al., 2021a; Zhang et al., 2021) have focused on using image clues to improve NER on Twitter. As for Chinese NER, introducing information from other modalities has

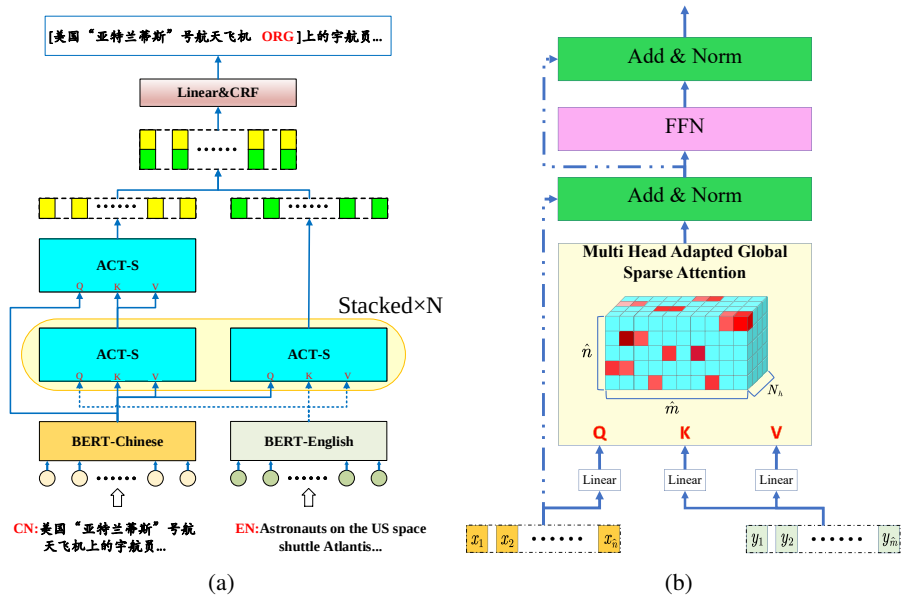

(a) (b)

Figure 2: (a) The overall architecture of our model. (b) The implementation details of ACT-S.

also become a promising solution. On the one hand, the multimodal information of Chinese characters is used to mining the semantics in the structure of Chinese characters(Sun et al., 2021b; Wu et al., 2021). On the other hand, multimodal information of the whole Chinese sentence, such as the audio content of the text (Sui et al., 2021), is also used to improve the word representation for Chinese NER. Compared with existing methods, the method we propose makes use of relative distance information while focusing on strongly correlated units during modal interactions.

2.3 Sequence labelling with bilingual clues or translation

Some previous works (Che et al., 2013; Wang et al., 2013) have demonstrated that constraints in bilingual parallel annotated corpora can be used to improve NER performance in two languages. But different from them, our proposed approach can take advantage of the hints in the unannotated English texts which are automatically translated by Neural machine translation tools. In addition, in cross-lingual sequence labelling tasks, translation methods(Mayhew et al., 2017; Fei et al., 2020; Zhen et al., 2021) are used to migrate annotation information from rich languages to low-resource languages. Unlike these previous works, the only resources used by our model in the additional language are texts with no annotation information. Furthermore, the model automatically learns all bilingual lexical alignment information with no assistance from any bilingual alignment tool.

3 Methodology

3.1 Overall Architecture

Task Formulation: Given a Chinese text $S_c = (cw_1, ..., cw_i, ..., cw_n)$ and its corresponding English translated text $E_c = (ew_1, ..., ew_l, ..., ew_m)$, where cw_i represents the i-th Chinese character and ew_l represents the l-th character of English translated text, the goal of the task is to utilize the information in the bilingual text to determine the spans and types of all named entities in the Chinese text. In this work, we formulate the task as a sequence labeling task. And the BMES (Beginning, Middle, End, Singleton)(Xue, 2003) tagging scheme is adopted.

The architecture of our proposed model is shown in Figure 2. In our model, we absorb the inspiration from the unified multimodal Transformer encoder widely used in vision-language tasks (Tsai et al., 2019; Yu et al., 2020). And similar to MECT (Wu et al., 2021), we introduce distance-aware and direction-aware components in Transformer attention. However, unlike previous works, we propose to use global sparse attention in the Cross-Transformer to reduce the noise in the information fusion process of the two modalities. Each part of the model is introduced in detail in the following sections.

3.2 Word Representations

Since many previous works have proven the effectiveness of the pre-trained language model in Named Entity Recognition task (Li et al., 2020; Wu et al., 2021), we use BERT (Devlin et al., 2019) as our contextualized representation encoder for both Chinese text and English translated text. To fit the BERT encoding procedure, we add two special symbols [CLS] and [SEP] at the beginning and end of the input sentence respectively, and we discard the representation vectors of [CLS] and [SEP] at the end of the BERT encoding computation. If a word is tokenized into several subwords by the Byte Pair Encoding (BPE) algorithm used in BERT, we found empirically that the model performs better when the average pooling method is used to merge the representations of the subwords that belong to the same word into a single representation vector. Thus, we can obtain the word representation generated using BERT:

$$(c_1, c_2, ..., c_n) = BERT_{-Ch}\left(cw_1, cw_2, ..., cw_n\right) \quad (1)$$

$$(e_1, e_2, ..., e_m) = BERT_{-En}\left(ew_1, ew_2, ..., ew_m\right) \quad (2)$$

3.3 Adapted Cross-Transformer with Global Sparse Attention

This section presents our first proposed Adapted Cross-Transformer with Global Sparse Attention (ACT-S) for bilingual information interaction in detail. As illustrated in Figure 2(a), several parameter independent ACT-Ss are used in our model for both inter- and intra-language interactions between bilinguals. The implementation details of ACT-S are shown in Figure 2(b).

Motivation: In multimodal tasks, the widely used cross-modal Transformers (Yu et al., 2020; Sui et al., 2021; Wu et al., 2021) typically use Softmax to normalize the cross-modal attention distribution of each head. As a result, in multi-head cross-modal attention, each unit is represented by all the units in the other modality by multiple different weighted averages. However, it is essential that a word is only associated with a small number of words in the other language in bilingual enhanced Chinese NER. Based on the above observations, we propose for the first time to incorporate the global sparse attention mechanism in a cross-modal Transformer to learn bilingual interactive word representations, which exclude the interference of irrelevant words in the other language. To our knowledge, it is the first time that global sparse attention is used for a multimodal task.

For inputs $X = (x_1, ..., x_u, ..., x_{\hat{n}})$ and $Y = (y_1, ..., y_v, ..., y_{\hat{m}})$, we treat $X \in \mathbb{R}^{\hat{n} \times d}$ as queries, and $Y \in \mathbb{R}^{\hat{m} \times d}$ as keys and values. The input Q, K, and V are obtained by linear transformation of X and Y:

$$Q^{(h)}, K^{(h)}, V^{(h)} = XW_q^{(h)}, YW_k^{(h)}, YW_v^{(h)} \quad (3)$$

where $h \in \{1, 2, ..., N_h\}$ is the index of the h-th attention head and N_h is the number of attention heads. $\left\{W_q^{(h)}, W_k^{(h)}, W_v^{(h)}\right\} \in \mathbb{R}^{d \times d_k}$ are learnable parameters and $d_k = \frac{d}{N_h}$.

To provide the attention mechanism in ACT-S with the ability of both distance perception and direction perception, we adopt the component of sensing relative distances similar to that in MECT(Wu et al., 2021) in the attention matrix computation process:

$$\begin{aligned} \widetilde{A}_{u,v}^{(h)} &= Q_u^{(h)}\left(K_v^{(h)}\right)^T + Q_u^{(h)} R_{u-v}^T \\ &+ K_v^{(h)} R_{u-v}^T + \alpha\left(K_v^{(h)}\right)^T + \beta R_{u-v}^T \end{aligned} \quad (4)$$

$$R_{u-v} = \left[..., \sin\left(\frac{u-v}{10^{6p/d_k}}\right), \cos\left(\frac{u-v}{10^{6p/d_k}}\right), ...\right] \quad (5)$$

where u is the index of the word in the target language and v is the token in the other language, Q_u, K_v is the query vector and key vector of word x_u, y_v respectively , $R_{u-v} \in \mathbb{R}^{d_k}$ is the relative position encoding, p in Eq. 5 is in the range $\left[0, \frac{d_k}{2}\right]$, $\alpha \in \mathbb{R}^{d_k}$ and $\beta \in \mathbb{R}^{d_k}$ are learnable parameters.

Different from MECT, we add a key bias term $K_v^{(h)} R_{u-v}^T$ to attention matrix to represent the bias of v-th token in the key sequence on certain relative distance and we empirically found that models perform better with it. And in this work, we consider that words in the target language are only relevant to a small number of words in another language. To learn a better representation of the target language words guided by the relevant words in another language. For the first time, we introduce sparse prior information to the global attention distribution via the top-k mask operation $Tkm(\cdot)$, which is formulated as follows:

$$Tkm(\widetilde{A}_{u,v}^{(h)}, k) = \begin{cases} \widetilde{A}_{u,v}^{(h)} & \text{if} \quad \widetilde{A}_{u,v}^{(h)} \in top(\widetilde{A}_{u,:}^{(h)}, k) \\ C & \text{if} \quad \widetilde{A}_{u,v}^{(h)} \notin top(\widetilde{A}_{u,:}^{(h)}, k) \end{cases} \quad (6)$$

Where k is a hyperparameter, masking constant $C \ll 0$, $A_{u,:} = Q_u^{(h)}\left(K^{(h)}\right)^T \in \mathbb{R}^{\hat{m}}$ contains the attention values between $Q_u^{(h)}$ and all keys in $K^{(h)}$,

$top(\widetilde{A}^{(h)}_{u,:}, k)$ is a set containing the largest k values in $\widetilde{A}^{(h)}_{u,:}$. The Adapted Global Sparse Attention matrix for the h-th attention head is then calculated as:

$$A^{(h)}_{u,v} = Tkm(\widetilde{A}^{(h)}_{u,v}, k) \tag{7}$$

And the attention score is calculated as follows:

$$Atten^{(h)} = soft\max\left(A^{(h)}\right)V^{(h)} \tag{8}$$

$$Atten = \left[Atten^{(1)}; ...; Atten^{(N_h)}\right]W^o \tag{9}$$

where $W^o \in \mathbb{R}^{d\times d}$ is a learnable parameter. Then, we stack the following sub-layers on top to obtain the text representation based on bilingual interaction:

$$\hat{O} = LN\left(X + Atten\right) \tag{10}$$

$$O = LN\left(\hat{O} + FFN\left(\hat{O}\right)\right) \tag{11}$$

where FFN is the feed-forward network, LN is the layer normalization. Both of these operations are consistent with those in vanilla Transformer (Vaswani et al., 2017).

3.4 Chinese representation with bilingual interaction

Given a Chinese text sequence $C = (c_1, c_2, ..., c_n)$ and an English text sequence $E = (e_1, e_2, ..., e_m)$, obtaining the Chinese representation of fused bilingual information requires two steps: inter-modal fusion and intra-modal fusion.

Inter-modal interaction: In order to learn a better representation of Chinese with the aid of information in the English text, we first employ ACT-Ss to get the English-Aware Chinese Representation. Similarly, to align each English word with its closely related Chinese characters, i.e., assigning high/low attention weights to its related/unrelated Chinese characters, we also use parameter independent ACT-S to gain the Chinese-Aware English Representation:

$$K^{(t)} = ACT\text{-}S\_cn(K^{(t-1)}, J^{(t-1)}) \tag{12}$$

$$J^{(t)} = ACT\text{-}S\_en(J^{(t-1)}, K^{(t-1)}) \tag{13}$$

where $t \in \{1, \cdots, N\}$, N denotes the number of interactions between bilingual modalities, $K^{(t)} = \left(k^{(t)}_1, k^{(t)}_2, ..., k^{(t)}_n\right)$, $J^{(t)} = \left(j^{(t)}_1, j^{(t)}_2, ..., j^{(t)}_m\right)$, $K^{(0)} = C$ and $J^{(0)} = E$.

Intra-modal interaction: In order to learn the dependencies between Chinese words, we use another ACT-S to obtain the Intra-Chinese Interaction Representation:

$$A = ACT\text{-}S\_in(C, J^{(N)}) \tag{14}$$

Then, we concatenate $A = (a_1, a_2, ..., a_n)$ and $J^{(N)}$ and input them into a fully connected layer to obtain the final hidden representations $H = (h_1, h_2, ..., h_n)$:

$$H = \left(a_1 \oplus j^{(N)}_1, ..., a_n \oplus j^{(N)}_n\right)W^F + b \tag{15}$$

where $\oplus$ denotes the concatenation operation, W^F and b are learnable parameters.

At last, we pass the final hidden representations H to a Conditional Random Field (CRF) (LAFFERTY, 2001) module.

4 Experiments

4.1 Experiment Settings

4.1.1 Datasets

We used four publicly available Chinese NER datasets, including Weibo NER (Peng and Dredze, 2015), Resume NER (Zhang and Yang, 2018), MSRA (Levow, 2006) and Ontonotes 4.0 (Weischedel et al., 2010). The corpus of MSRA and Ontonotes 4.0 comes from news, the corpus of Weibo comes from social media, and the corpus of Resume comes from the resume data in Sina Finance. The splitting methods and other pre-processing methods of datasets follow those in (Zhang and Yang, 2018; Li et al., 2020; Wu et al., 2021).

4.1.2 Text Translation

Neural machine translation (NMT) methods have achieved state-of-the-art performance for the translation of a wide range of language pairs(Vaswani et al., 2017). Therefore, automatic text translation with NMT is applicable. In this work, we employ Baidu Translation API[1] and Tencent Translation API[2] to automatically translate Chinese text into English, respectively, and both machine translation systems have achieved the highest BLUE score on the WMT Chinese-English task(Sun et al., 2019; Wang et al., 2021b) in recent years.

[1] https://fanyi-api.baidu.com/

[2] https://cloud.tencent.com/product/tmt

Models	Resources	Weibo			Resume			MSRA			Ontonotes 4.0		
		P(%)	R(%)	F1(%)	P(%)	R(%)	F1(%)	P(%)	R(%)	F1(%)	P(%)	R(%)	F1(%)
BERT-Tagger♦†	-	-	-	68.20	-	-	95.53	-	-	94.95	-	-	80.14
LSTM-CRF[BERT]♦*	-	68.21	68.38	68.29	95.11	96.01	95.56	95.33	95.04	95.18	79.92	80.56	80.24
TENER[BERT]♦*	-	67.19	69.49	68.32	94.98	96.20	95.59	95.39	95.44	95.41	78.69	82.21	80.41
FLAT[BERT]♦†	L	-	-	68.55	-	-	95.86	-	-	96.09	-	-	81.82
DyLex♦†	L	-	-	71.12	-	-	95.99	-	-	96.49	-	-	81.48
MECT[BERT]♦†	L+RC	-	-	70.43	-	-	95.98	-	-	96.24	-	-	82.57
Ours(N=2)♦*	T-T	**72.60**	**74.16**	**73.37**	96.29	**96.99**	**96.87**	**96.62**	96.82	96.72	83.95	83.77	83.86
Ours(N=2)♦*	T-B	72.57	73.95	73.25	**96.30**	96.91	96.60	96.59	**96.89**	**96.74**	**83.98**	**83.85**	**83.91**
BERT-Tagger▲†	-	67.12	66.88	67.33	96.12	95.45	95.78	94.43	93.86	94.14	78.01	80.35	79.16
LSTM-CRF[BERT]▲*	-	67.63	67.18	67.40	95.84	95.61	95.72	94.46	93.89	94.17	78.92	79.56	79.24
TENER[BERT]▲*	-	66.69	68.21	67.44	95.05	96.63	95.83	94.45	94.19	94.32	78.99	79.70	79.34
LEBERT▲†	L	-	-	70.75	-	-	96.08	-	-	95.70	-	-	82.08
PLTE[BERT]▲†	L	72.00	66.67	69.23	96.16	96.75	96.45	94.91	94.15	94.53	79.62	81.82	80.60
SLex-LSTM[BERT]▲†	L	70.94	67.02	70.50	96.08	96.13	96.11	95.75	95.10	95.42	83.41	82.21	82.81
Ours(N=2)▲*	T-T	**73.21**	**71.90**	**72.55**	96.21	**96.89**	96.55	96.35	95.91	96.13	83.24	**83.82**	83.53
Ours(N=2)▲*	T-B	73.15	71.84	72.48	**96.32**	96.87	**96.59**	**96.37**	**95.92**	**96.14**	**83.41**	83.71	**83.56**

Table 1: Main results. Bold marks the highest score. † marks results quoted directly from the original papers. ♦ marks results produced with BERT-wwm. ▲ marks results produced with BERT-base-Chinese. ∗ marks the results implemented in the fastNLP[3] framework. 'L' denotes using the lexicon resources. 'RC' denotes the radical information of Chinese. 'T-T' denotes using the bilingual information from the Tencent Translation API and 'T-B' denotes using the bilingual information from the Baidu Translation API.

4.1.3 Baseline Methods

To demonstrate the effectiveness of our proposed model, we compare it with several strong baseline models for Chinese NER: (1) **BERT-Tagger** (Devlin et al., 2019)(2) **LSTM-CRF[BERT]** (Huang et al., 2015) (3) **TENER[BERT]** (Yan et al., 2019). Besides, we also compare our method with the lexicon enhancement methods, which are the state-of-the-art methods for Chinese NER: (1) **LEBERT** (Liu et al., 2021) (2) **FLAT[BERT]** (Li et al., 2020) (3) **PLTE[BERT]** (Mengge et al., 2020)(4) **SoftLexicon-LSTM[BERT]** (Ma et al., 2020)(In this paper, we call Soft-Lexicon SLex for short.) (5)**DyLex** (Wang et al., 2021a) (6) **MECT[BERT]** (Wu et al., 2021).

4.1.4 Implement Details

The English word representations E are initialized with the cased BERT-base-English model pre-trained by Devlin et al. (2019). In addition, to make the comparison with the results of the baseline models convincing, we use BERT-base-Chinese(Devlin et al., 2019) and BERT-wwm(Cui et al., 2021) to initialize the Chinese word representations C separately to get different experimental results for comparison and fine-tuned during training. All the neural models are implemented with PyTorch and fastNLP[3]. More implementation details are described in Appendix A.

[3] https://github.com/fastnlp/fastNLP

4.2 Main Results

We compared our proposed method with the state of the art methods. The experimental results are reported in Table 1, which is divided into two blocks. The methods in the first block use the Chinese word representation from BERT-wwm. And the methods in the second block use the Chinese word representation from BERT-base-Chinese. Our model achieves a significant and consistent performance boost over current SOTA models on four Chinese NER datasets. From the results, we can observe that:

(1) In comparison with the methods without external resources (BERT-Tagger, LSTM-CRF[BERT] and TENER[BERT]), our model achieves a significant performance boost. Because our model makes use of the rich prompt information in the English text that help to determine the boundaries and types of entities. It demonstrates the significant effect of introducing bilingual information compared to just using the internal features of Chinese.

(2) Compared with the lexicon enhancement method using pre-trained BERT-wwm Chinese representation, our model has superior performance, i.e., +2.25 , +0.88, +0.25 ,+1.34 on Weibo, Resume, MSRA, Ontonotes4.0, respectively. When compared with baselines with BERT-base-Chinese, the performance of our model is still be competitive, i.e., +1.8, +0.14, +0.44, +0.75 on Weibo, Resume, MSRA, Ontonotes4.0, respectively. This verifies our claim that, compared with the exter-

Models	Weibo	Resume	MSRA	OntoNotes
Ours	73.37	96.87	96.72	83.86
-GS	71.68	96.09	96.02	82.72
-RA	72.23	96.18	96.14	82.81
-APW	73.28	96.80	96.65	83.79
-KB	73.25	96.75	96.64	83.81

Table 2: An ablation study of the proposed model. F1 scores were evaluated on the test sets. 'GS' denotes the Global Sparse operation. 'RA' denotes the relative distance-aware attention. 'APW' denotes the average pooling operation used to obtain the word-level representations. 'KB' denotes the key bias term in the attention matrix.

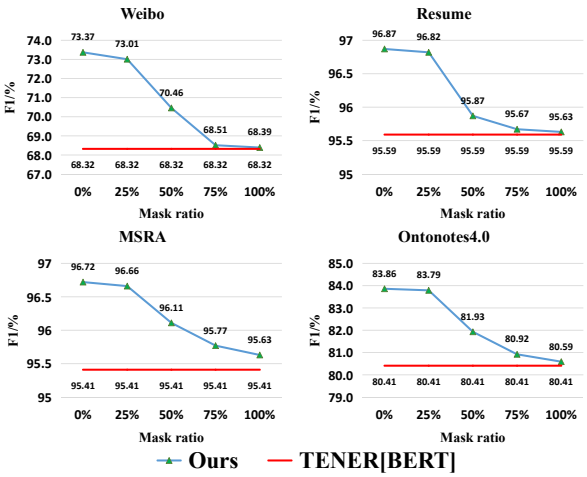

Figure 3: Impact of the machine translation quality. F1 scores were evaluated on the test sets.

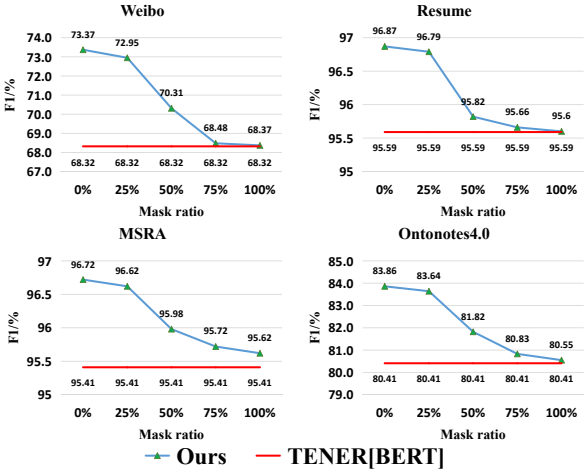

Figure 4: Impact of missing translated texts. F1 scores were evaluated on the test sets.

nal resources used by most Chinese NER models, incorporating the rich information in English is a promising way to improve the performance of Chinese NER tasks.

(3) All translated English texts used in this work are automatically generated from the publicly available machine translation API. Compared with the baseline models with external resources, our proposed bilingual enhanced approach improves performance and requires no artificially generated external knowledge. From another perspective, we believe that our proposed model transfers the knowledge in the neural machine translation model to enhance the Chinese NER model.

(4) Our proposed model could achieve state-of-the-art performance in both cases with English texts automatically translated by two different machine translation systems. This suggests that the performance improvement of our model is not dependent on a specific machine translation system.

(5) Even when the size of training set is small, such as Weibo NER, the performance improvement of our model over other baselines is still significant. This demonstrates that our proposed model is not data-hungry and has promising potential in low-resource NER scenarios.

5 Analysis and Discussion

5.1 Ablation Study

To study the contribution of the main components in our model, we conducted an ablation study on all four datasets. The results are reported in Table 2. And we can observe the following facts:

(1) To demonstrate the advantage of global sparse operation, we remove it from the model. The results show that, without global sparse operation, the performance of the model degrades severely, which indicates that there exist severe interference of irrelevant words during the process of bilingual interactions. The global sparse operation in ACT-S substantially improves the performance of our model, demonstrating its effectiveness while achieving higher interpretability for our method.

(2) The component of sensing relative distance in ACT-S has a significant positive impact on the performance of the proposed model, and the model without it shows a certain degree of performance degradation. This illustrates the validity of the relative distance-aware component in our model.

(3) We empirically found that the model performs better at word-level bilingual interactions than at subword-level.

(4) There are positive effects of introducing a key bias term into the cross-attention matrix.

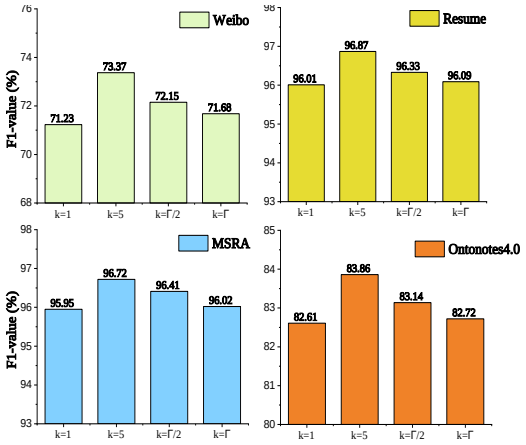

Figure 5: Influence of attention sparsity k. Γ is the length of the text sequence.

Method	NER results
Gold labels	[美国"亚特兰蒂斯号"航天飞机 ORG]上的宇航员... Astronauts on [the US space shuttle Atlantics ORG]...
BERT-Tagger	[美国 LOC]"亚特兰蒂斯号"航天飞机上的宇航员...
MECT	[美国 LOC]["亚特兰蒂斯号"航天飞机 ORG]上的宇航员...
Translation-1	Astronauts on the US space shuttle Atlantics...
Ours-1	[美国"亚特兰蒂斯号"航天飞机 ORG]上的宇航员...
Translation-2	Astronauts on space shuttle Atlantics of US...
Ours-2	[美国"亚特兰蒂斯号"航天飞机 ORG]上的宇航员...

Table 3: Example-1 of the NER results.

5.2 Impact of machine translation quality & missing translated texts

To illustrate the robustness of our model, we set up separate experiments to investigate the effect of machine translation quality and missing translated text on our model. We randomly replace words in the automatically translated English text with the mask token [MASK] in a fixed proportion to simulate a reduction in the machine translation quality. Similarly, we randomly replace the entire automatically translated English text with a fixed-length(average length of samples in the training set) sequence of the mask token [MASK] at a certain rate for all samples. As shown in Figure 3 and Figure 4, our model can achieve excellent performance even in cases where the translated text has little noise or is missing. And when the translated text is almost full of noise or even 100% missing, our model still outperforms TENER [BERT], a strong baseline that does not use any external resources, which demonstrates the robustness of our model. In addition, it demonstrates that the performance improvement of our model originates not only from bilingual resources but is also related to the model itself.

5.3 Impact of attention sparsity k.

We also conducted experiments to verify the impact of the attention sparsity control factor k in ACT-Ss. The results reported on the four datasets are shown in Figure 5. From the results, we can see that the inappropriate sparsity of global attention significantly degrades the performance of the model. If the global cross-attention matrix is too sparse, such as in the case of $k = 1$, the ACT-S module cannot learn sufficiently about the dependencies between bilingual texts. At the other extreme, if the attention matrix takes too many interactions between bilingual words into account, the model will not achieve the best performance either. This indicates the sparse nature of lexical dependencies between bilingual texts. Furthermore, the experimental results suggest that it is applicable and interpretable to introduce sparse attention rather than full attention in bilingual interactions.

5.4 Case Study

Table 3 illustrates one typical example where our proposed bilingual enhancement model successfully tackles the dilemma of the complex structure of entity composition in the Chinese NER task. Most of the existing Chinese NER methods utilize only the internal features of the Chinese language, which makes it difficult to tag entities with complex composition structures correctly. When rich cues in English are leveraged, this problem can be alleviated. In addition, it can be seen from these two cases that the clues in the different English translations are all beneficial for the correct labeling of complex entities.

6 Conclusion

In this paper, we propose an Adapted Cross-Transformers with Global Sparse Attention (ACT-S) module to explore bilingual interaction information to improve the performance of the Chinese NER task. Several parameter independent ACT-Ss are employed in our work to capture the rich information in both English and Chinese. We evaluate the proposed model on four Chinese NER datasets and the experimental results illustrate that our method achieves significant and consistent improvement compared to other baselines. In the fu-

ture, we will explore how to improve Chinese NER with features from languages other than English and extend our model to other sequence labelling tasks.

Acknowledgements

This research is supported by the Fundamental Research Funds for the Central Universities (No.DUT22ZD205). Finally, we would like to thank the anonymous reviewers for their valuable comments.

References

Wanxiang Che, Mengqiu Wang, Christopher D Manning, and Ting Liu. 2013. Named entity recognition with bilingual constraints. In *NAACL*, pages 52–62.

Yiming Cui, Wanxiang Che, Ting Liu, Bing Qin, and Ziqing Yang. 2021. Pre-training with whole word masking for chinese bert. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 29:3504–3514.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: pre-training of deep bidirectional transformers for language understanding. In *NAACL*, pages 4171–4186.

Dennis Diefenbach, Vanessa Lopez, Kamal Singh, and Pierre Maret. 2018. Core techniques of question answering systems over knowledge bases: a survey. *Knowledge and Information systems*, 55(3):529–569.

Chuanhai Dong, Jiajun Zhang, Chengqing Zong, Masanori Hattori, and Hui Di. 2016. Character-based LSTM-CRF with radical-level features for chinese named entity recognition. In *Natural Language Understanding and Intelligent Applications - 5th CCF Conference on Natural Language Processing and Chinese Computing, NLPCC 2016, and 24th International Conference on Computer Processing of Oriental Languages, ICCPOL 2016, Kunming, China, December 2-6, 2016, Proceedings*, volume 10102 of *Lecture Notes in Computer Science*, pages 239–250. Springer.

Hao Fei, Meishan Zhang, and Donghong Ji. 2020. Cross-lingual semantic role labeling with high-quality translated training corpus. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7014–7026.

Tao Gui, Ruotian Ma, Qi Zhang, Lujun Zhao, Yu-Gang Jiang, and Xuanjing Huang. 2019a. Cnn-based chinese ner with lexicon rethinking. In *IJCAI*.

Tao Gui, Yicheng Zou, Qi Zhang, Minlong Peng, Jinlan Fu, Zhongyu Wei, and Xuan-Jing Huang. 2019b. A lexicon-based graph neural network for chinese ner. In *EMNLP*, pages 1040–1050.

Zhiheng Huang, Wei Xu, and Kai Yu. 2015. Bidirectional lstm-crf models for sequence tagging. *arXiv preprint arXiv:1508.01991*.

JD LAFFERTY. 2001. Conditional random fields: probabilistic models for segmenting and labeling sequence data. In *ICML-2001*.

Gina-Anne Levow. 2006. The third international chinese language processing bakeoff: Word segmentation and named entity recognition. In *Proceedings of the Fifth SIGHAN Workshop on Chinese Language Processing*, pages 108–117.

Xiaonan Li, Hang Yan, Xipeng Qiu, and Xuanjing Huang. 2020. Flat: Chinese ner using flat-lattice transformer. In *ACL*.

Pan Liu, Yanming Guo, Fenglei Wang, and Guohui Li. 2022. Chinese named entity recognition: The state of the art. *Neurocomputing*, 473:37–53.

Wei Liu, Xiyan Fu, Yue Zhang, and Wenming Xiao. 2021. Lexicon enhanced chinese sequence labeling using BERT adapter. In *ACL*, pages 5847–5858.

Ruotian Ma, Minlong Peng, Qi Zhang, Zhongyu Wei, and Xuan-Jing Huang. 2020. Simplify the usage of lexicon in chinese ner. In *ACL*, pages 5951–5960.

Stephen Mayhew, Chen-Tse Tsai, and Dan Roth. 2017. Cheap translation for cross-lingual named entity recognition. In *Proceedings of the 2017 conference on empirical methods in natural language processing*, pages 2536–2545.

Yuxian Meng, Wei Wu, Fei Wang, Xiaoya Li, Ping Nie, Fan Yin, Muyu Li, Qinghong Han, Xiaofei Sun, and Jiwei Li. 2019. Glyce: Glyph-vectors for chinese character representations. *Advances in Neural Information Processing Systems*, 32:2746–2757.

Xue Mengge, Bowen Yu, Tingwen Liu, Yue Zhang, Erli Meng, and Bin Wang. 2020. Porous lattice transformer encoder for chinese ner. In *COLING*, pages 3831–3841.

Nanyun Peng and Mark Dredze. 2015. Named entity recognition for chinese social media with jointly trained embeddings. In *EMNLP*, pages 548–554.

Dianbo Sui, Zhengkun Tian, Yubo Chen, Kang Liu, and Jun Zhao. 2021. A large-scale chinese multimodal ner dataset with speech clues. In *ACL*, pages 2807–2818.

Lin Sun, Jiquan Wang, Kai Zhang, Yindu Su, and Fangsheng Weng. 2021a. Rpbert: A text-image relation propagation-based bert model for multimodal ner. In *AAAI*, pages 13860–13868.

Meng Sun, Bojian Jiang, Hao Xiong, Zhongjun He, Hua Wu, and Haifeng Wang. 2019. Baidu neural machine translation systems for wmt19. In *Proceedings of the Fourth Conference on Machine Translation (Volume 2: Shared Task Papers, Day 1)*, pages 374–381.

Zijun Sun, Xiaoya Li, Xiaofei Sun, Yuxian Meng, Xiang Ao, Qing He, Fei Wu, and Jiwei Li. 2021b. Chinesebert: Chinese pretraining enhanced by glyph and pinyin information. In *ACL*, pages 2065–2075.

Yao-Hung Hubert Tsai, Shaojie Bai, Paul Pu Liang, J. Zico Kolter, Louis-Philippe Morency, and Ruslan Salakhutdinov. 2019. Multimodal transformer for unaligned multimodal language sequences. In *ACL*, pages 6558–6569.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in neural information processing systems*, pages 5998–6008.

Baojun Wang, Zhao Zhang, Kun Xu, Guang-Yuan Hao, Yuyang Zhang, Lifeng Shang, Linlin Li, Xiao Chen, Xin Jiang, and Qun Liu. 2021a. Dylex: Incorporating dynamic lexicons into bert for sequence labeling. In *emnlp*, pages 2679–2693.

Longyue Wang, Mu Li, Fangxu Liu, Shuming Shi, Zhaopeng Tu, Xing Wang, Shuangzhi Wu, Jiali Zeng, and Wen Zhang. 2021b. Tencent translation system for the wmt21 news translation task. In *Proceedings of the Sixth Conference on Machine Translation*, pages 216–224.

Mengqiu Wang, Wanxiang Che, and Christopher Manning. 2013. Effective bilingual constraints for semi-supervised learning of named entity recognizers. In *AAAI*, volume 27.

Ralph Weischedel, Sameer Pradhan, Lance Ramshaw, et al. 2010. Ontonotes release 4.0.

Shuang Wu, Xiaoning Song, and Zhenhua Feng. 2021. Mect: Multi-metadata embedding based cross-transformer for chinese named entity recognition. In *ACL*, pages 1529–1539.

Nianwen Xue. 2003. Chinese word segmentation as character tagging. In *Int. J. Comput. Linguist.Chin. Lang. Process*, pages 29–48.

Hang Yan, Bocao Deng, Xiaonan Li, and Xipeng Qiu. 2019. Tener: adapting transformer encoder for named entity recognition. *arXiv preprint arXiv:1911.04474*.

Jianfei Yu, Jing Jiang, Li Yang, and Rui Xia. 2020. Improving multimodal named entity recognition via entity span detection with unified multimodal transformer. In *ACL*, pages 3342–3352.

Dmitry Zelenko, Chinatsu Aone, and Anthony Richardella. 2003. Kernel methods for relation extraction. *J. Mach. Learn. Res.*, 3:1083–1106.

Dong Zhang, Suzhong Wei, Shoushan Li, Hanqian Wu, Qiaoming Zhu, and Guodong Zhou. 2021. Multi-modal graph fusion for named entity recognition with targeted visual guidance. In *AAAI*, pages 14347–14355.

Yue Zhang and Jie Yang. 2018. Chinese ner using lattice lstm. In *ACL*, pages 1554–1564.

Ranran Zhen, Rui Wang, Guohong Fu, Chengguo Lv, and Meishan Zhang. 2021. Chinese opinion role labeling with corpus translation: A pivot study. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 10139–10149.

Read Extensively, Focus Smartly: A Cross-document Semantic Enhancement Method for Visual Documents NER

Jun Zhao[1]*, **Xin Zhao**[1]*, **Wenyu Zhan**[1], **Tao Gui**[2]†, **Qi Zhang**[1]†
Liang Qiao[3], **Zhanzhan Cheng**[3], **Shiliang Pu**[3]
[1]School of Computer Science, Shanghai Key Laboratory of Intelligent Information Processing, Fudan University, Shanghai, China
[2]Institute of Modern Languages and Linguistics, Fudan University
[3]Hikvision Research Institute, Shanghai, China
{zhaoj19,tgui,qz}@fudan.edu.cn,{zhaoxin21,wyzhan21}@m.fudan.edu.cn

Abstract

The introduction of multimodal information and pretraining technique significantly improves entity recognition from visually-rich documents. However, most of the existing methods pay unnecessary attention to irrelevant regions of the current document while ignoring the potentially valuable information in related documents. To deal with this problem, this work proposes a cross-document semantic enhancement method, which consists of two modules: 1) To prevent distractions from irrelevant regions in the current document, we design a learnable attention mask mechanism, which is used to adaptively filter redundant information in the current document. 2) To further enrich the entity-related context, we propose a cross-document information awareness technique, which enables the model to collect more evidence across documents to assist in prediction. The experimental results on two documents understanding benchmarks covering eight languages demonstrate that our method outperforms the SOTA methods.

1 Introduction

Visually-rich documents (VRDs) are the most common information carriers in the real world, such as newspapers, resumes, tickets, etc. Different from plain text data, the information in VRDs is encoded by multiple modalities including text, vision, and layout. Entity recognition from VRDs, as a key step for document understanding, is of utmost practical interest for many downstream applications such as business analysis (Xu et al., 2020), intelligent education (Kahraman et al., 2010), digital library (Kroll et al., 2021).

The pioneering explorations approach this task by either computer vision (Katti et al., 2018) or natural language processing (Lample et al., 2016) paradigm. However, these methods ignore the inherent multi-modality of VRDs and consequently the suboptimal results are achieved. To address this problem, graph neural network (Liu et al., 2019) and self-attention mechanism (Zhang et al., 2020) are introduced to capture the cross-modality interaction and achieve superior performance. Recently, inspired by the widespread success of large pretrained language models (Qiu et al., 2020), self-supervised pretraining techniques are leveraged to learn cross-modal knowledge from unlabeled documents (Xu et al., 2020, 2021a,b) and the amount of labeled data required for document understanding is greatly reduced.

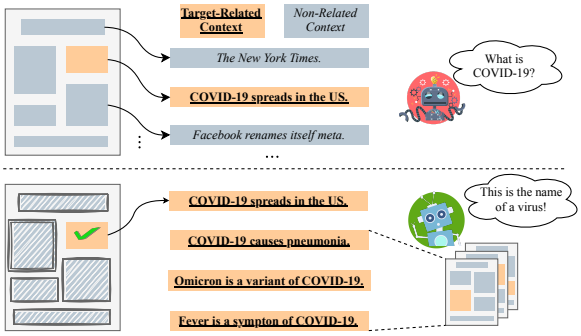

Figure 1: Redundant irrelevant information and insufficient entity-related information in current document make it difficult to extract entities accurately. We filter redundant information from irrelevant regions and expand the information source from a single to multiple related documents. Thereby, entity-related information is efficiently integrated and the extraction is improved.

However, the existing methods suffer from the following two main shortcomings. From **intra-document** perspective, most previous works model a document by encoding each token uniformly, ignoring the fact that the document is composed of several regions (e.g. paragraphs, tables, captions) that are not so closely related semantically (Binmakhashen and Mahmoud, 2019). As shown in Figure 1, the type of COVID-19 in

*Equal Contributions.
†Corresponding authors.

Proceedings of the 29th International Conference on Computational Linguistics, pages 2034–2043
October 12–17, 2022.

the orange region does not depend on the context in other gray regions. Unrestricted consideration of the whole document will not only distract the attention to local regions with stronger semantic associations (Guo et al., 2019b) but also increase the risk of fitting spurious features (also known as the *Shortcut* phenomenon (Geirhos et al., 2020)). From the **inter-document** perspective, the observed information is limited to the single document context, which may not be enough to accurately recognize entities. As shown in Figure 1, we can't judge whether COVID-19 is a piece of news or a virus only by virtue of "COVID-19 spreads in the US". Worse still, the insufficient context information can be further destroyed by the errors in character recognition when faced with low-quality documents (e.g. imaging blur, deformity) (Mou et al., 2020).

In this work, we propose a cross-document semantic enhancement method to enable the model to focus and integrate entity-related information across documents. Specifically, to prevent distractions from irrelevant regions in the current document, we design a learnable attention mask mechanism. Cross-region attentions are masked with a higher probability, so the model tends to make predictions using more reliable features from the local region. We introduce gumbel softmax (Jang et al., 2017) to solve the non-differentiable problem of discrete mask variables. To embrace sufficient context to assist prediction, we propose a cross-document information awareness technique. Inverted index (Knuth, 1997) is used to efficiently store and retrieve the contextualized representations of each token from each document. Cross-document attention acts on entity token representations to collect sufficient evidence to improve prediction.

Our contribution is threefold: 1) we propose a cross-document semantic enhancement method to enable the model to focus on the entity-related information across documents. The method filters the redundant information while expanding the information source; 2) our method can be regarded as a plug-in. It can be added to any document understanding model to improve prediction. 3) The experimental results on two datasets covering eight languages demonstrate that the proposed method outperforms the state-of-the-art methods;

2 Related Work

2.1 Multi-modal Named Entity Recognition

The approaches used to handle the task roughly fall into one of three directions. (1) **From single to multiple modality**. Due to the inherent multi-modality of visually-rich documents, early attempts from the perspective of computer vision (CV) (Katti et al., 2018) or natural language processing (NLP) (Lample et al., 2016; Ma et al., 2022; Zhao et al., 2021; Wang et al., 2022) can not achieve optimal performance. Therefore, artfully designed network architectures (Yu et al., 2021) and sophisticated mechanisms (Guo et al., 2019a) are used to fuse multimodal features for document understanding. (2) **From isolated to end-to-end optimization**. In classical document understanding methods, modules such as text recognition, image encoding, and information extraction are still optimized in isolation with different objective functions. To deal with this limitation and extract task-tailored features, end-to-end training frameworks are proposed (Zhang et al., 2020; Wang et al., 2021a), in which all modules are differentiable and optimized by a unified loss function. (3) **From supervised to pretraining paradigm**. Recently, Xu et al. (2020) extend the textual pretraining task to visual documents. Multiple well-designed pretraining objectives facilitate the interaction of multimodal information. With the help of learned generic document features, only a few samples would be sufficient to achieve SOTA accuracy. Different from all the above-mentioned methods, we rethink this task from the perspective of information integration and achieve the transition **from single-document uniform to cross-document selective information integration** to improve prediction.

2.2 Shortcut Phenomenon in Neural Network

Why should we selectively focus on local regions rather than dealing with each token of the whole document without distinction? The answer is the Shortcut phenomenon (Geirhos et al., 2020), which illustrates that neural networks always tend to fit training objectives in the simplest way. For example, suppose that in the training set, the word "Washington" in samples entitled "New York Times" all stand for place names. Global self-attention can easily learn this spurious feature. When "Washington" appears as a person's name, the model may completely ignore its context and

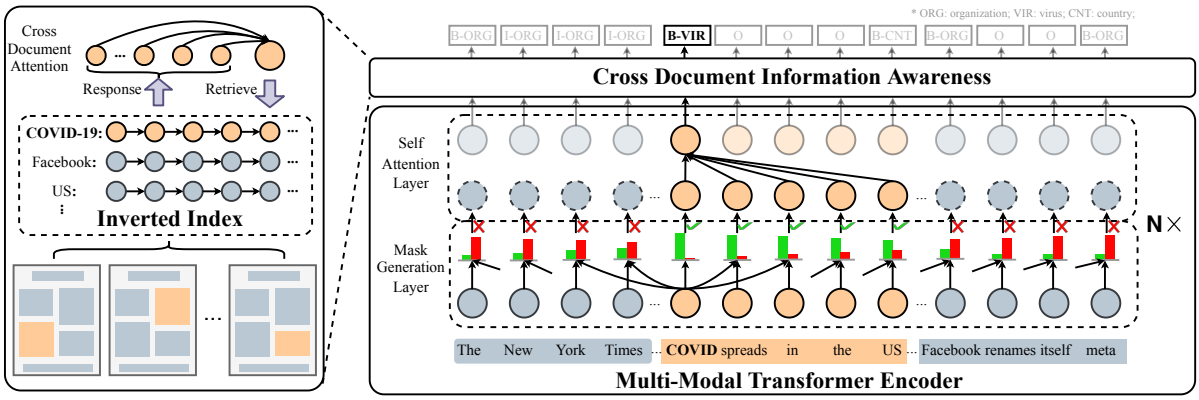

Figure 2: We take the extraction of COVID-19 as an example to illustrate the proposed method. First, a multimodal encoder is used to encode the context information of the current document. To filter the redundant irrelevant information (gray nodes) in the document, the mask generation layer is added to each layer of the encoder. Tokens farther away from COVID-19 will be masked with a higher probability during training (dotted node). Finally, we efficiently query COVID-19's representation in other documents by inverted index and a cross-document self-attention is used to integrate the information of COVID-19 in related documents.

directly make wrong predictions based on the title.

3 Approach

We propose a cross-document semantic enhancement method, which enables the model to make use of the cross-document context of the target entity and reduce the impact of noise from irrelevant regions of the current document.

The problem settings in this paper are formally stated as follows. Let $\mathcal{E} = \{e_i\}_{i=1,...,N_\mathcal{E}}$ denotes the predefined $N_\mathcal{E}$ entity types of interest and $\mathcal{C} = \{c_i\}_{i=1,...,N_c}$ denotes the entity label set derived by $\mathcal{E}$ according to the BIO scheme. Let $\mathcal{D} = \{(w_i, \boldsymbol{b}_i, v_i)\}_{i=1,...,N_\mathcal{D}}$ be a visually-rich document (VRDs), where w_i denotes token in the document. $\boldsymbol{b}_i = (x_i^0, y_i^0, x_i^1, y_i^1)$ denotes the bounding box (i.e. the position in the document) and v_i denotes image patch of w_i. Given a labeled set of VRDs $\mathcal{S} = \{\mathcal{D}_i\}_{i=1,...,N_\mathcal{S}}$, we target at learning a mapping function $\mathcal{F} : \mathcal{D} \rightarrow \{c_{w_i}\}_{i=1,...,N_\mathcal{D}}$. In other words, $\mathcal{F}$ assigns an entity label $c_{w_i} \in \mathcal{C}$ to each word w_i in the document to extract entities of interest.

3.1 Method Overview

We improve entity recognition from VRDs by the proposed cross-document semantic enhancement method, which enables the model to focus on the entity-related information across documents, rather than being distracted by irrelevant regions in the current document. As illustrated in Figure 2, the proposed method works as follows.

(1) We encode a visually-rich document $\mathcal{D} \in \mathcal{S}$ using a multi-modal transformer encoder implemented as the pretrained LayoutXLM (Xu et al., 2021b), which takes multi-modal information including text w_i, layout $\boldsymbol{b}_i$ and picture patch cut by $\boldsymbol{b}_i$ as input, and $\boldsymbol{h}_{i,j}$ denotes the output of layer j of the encoder. However, a document is usually composed of several regions that are not so closely related semantically. Consequently, unrestricted consideration of the whole document will not only distract the attention to local regions with stronger semantic associations but also can increase the risk of fitting spurious features.

(2) To prevent distractions from irrelevant regions in the current document, we design a learnable attention mask mechanism. Tokens that are farther away from the current token will be masked with a higher probability. Since mask operation is a discrete variable sampled from the binomial distribution, Gumbel softmax is introduced to realize end-to-end optimization of mask distribution. However, the current document context may not contain enough information to accurately classify the output of the encoder $\boldsymbol{h}_{i,N}$ to the true entity label c_{w_i}.

(3) To embrace sufficient context to improve prediction, we propose a cross-document information awareness technique. Each word w_i corresponds to a queue $\mathcal{Q}_i = \{\boldsymbol{h}_{i,N}^m\}_{m=1}^{|\mathcal{Q}|}$ that stores the contextualized representation of w_i in $|\mathcal{Q}|$ different documents. When encoding the current document $\mathcal{D}$, for each word $w_i \in \mathcal{D}$, we retrieve $\mathcal{Q}_i$ and obtain the final representation

$\boldsymbol{h}_i$ through the cross-document attention between $\boldsymbol{h}_{i,N}$ and each $\boldsymbol{h}^m_{i,N} \in \mathcal{Q}_i$. After that, $\boldsymbol{h}_{i,N}$ is updated to queue $\mathcal{Q}_i$. The lazy update ensures efficiency. Based on $\boldsymbol{h}_i$, we classify w_i into its corresponding entity label c_{w_i}.

3.2 Learnable Attention Mask Mechanism

In this section, we elaborate on the proposed learnable attention mask mechanism. Firstly, the pretrained encoder aims to integrate multi-modal inputs such as text, layout, and vision, and obtain the fixed-length representation of each token. To reduce the excessive attention to the irrelevant regions in the document during encoding, a mask sampled from the binomial distribution is applied to the original attention distribution, and the tokens farther away from the current position will be masked with a higher probability. Finally, the introduction of Gumbel relaxation solves the problem of non-differentiability of the discrete mask, which enables the model to learn the optimal mask distribution in an end-to-end manner.

3.2.1 Multi-Modal Transformer Encoder

The proposed cross-document semantic enhancement method is architecture-agnostic and can be added to any encoder architecture based on self attention mechanism. We adopt LayoutXLM (Xu et al., 2021b) as the implementation of our encoder $\boldsymbol{enc}(\cdot)$ because LayoutXLM is a multilingual encoder, which enables us to comprehensively demonstrate the effectiveness of the proposed method in different languages. We follow the way in LayoutXLM to generate input of the encoder. Specifically, given a visually-rich document $\mathcal{D} = \{(w_i, \boldsymbol{b}_i, v_i)\}_{i=1,...,N_\mathcal{D}}$, the multi-modal transformer encoder $\boldsymbol{enc}(\cdot)$ takes inputs from three different modalities, including text w_i, layout $\boldsymbol{b}_i$, and image patch v_i, which are mapped to text embedding $\boldsymbol{w}_i$, layout embedding $\boldsymbol{PE}_i$, and visual embedding $\boldsymbol{v}_i$, respectively. The text and visual embeddings are concatenated, then plus the layout embedding to get the input embedding $\boldsymbol{H}^0 = \{\boldsymbol{h_{1,0}}, ..., \boldsymbol{h_{N,0}}\} \in \mathcal{R}^{N\times d}$. Then, the intra-document self-attention transform $\boldsymbol{H}^0$ into the queries $\boldsymbol{Q}^0 \in \mathcal{R}^{N\times d}$, the keys $\boldsymbol{K}^0 \in \mathcal{R}^{N\times d}$, and the values $\boldsymbol{V}^0 \in \mathcal{R}^{N\times d}$. Finally, the output of the current layer is calculated as follows:

$$\boldsymbol{H}^{l+1} = ATT(\boldsymbol{Q}^l, \boldsymbol{K}^l)\boldsymbol{V}^l \tag{1}$$

$$ATT(\boldsymbol{Q}^l, \boldsymbol{K}^l) = \text{Softmax}(\frac{\boldsymbol{Q}^l \boldsymbol{K}^{l,\top}}{\sqrt{d}}). \tag{2}$$

The output of the last layer $\boldsymbol{H}^N$ is used as the input of the entity classifier. Although the global attention mechanism can model the interaction between multi-modal information, the redundant irrelevant information contained in the document reduces the attention of the model to the local regions with stronger semantic relevance, which leads to sub-optimal results.

3.2.2 Self-Attention with Mask Mechanism

In order to enhance the attention to local regions and reduce the risk of fitting spurious features, we carefully design a mask generation layer, which aims to select a more reliable subset from the input document as the basis for model prediction. Specifically, for the representation of each token $\boldsymbol{h}_{i,l} \in \boldsymbol{H}^l$ in layer l of encoder, the mask generation layer outputs a specific mask sequence $\boldsymbol{m} = (m_0, m_1, ..., m_N)$, $m_i \in \{0, 1\}$, where 0 and 1 here represent discard and select respectively. When calculating whether a contextual token will be discarded, we first calculate its distance ΔX on the horizontal axis and ΔY on the longitudinal axis with the current word. Then the mask is sampled from the following binomial distribution:

$$\begin{aligned} P_{\Delta X,Y}(m) &= m * \pi_{\Delta X,Y} \\ &\quad + (1-m) * (1 - \pi_{\Delta X,Y}) \end{aligned} \tag{3}$$

$$\pi_{\Delta X,Y} = e^{-[\alpha(\frac{\Delta X}{X_{MAX}}) + \beta(\frac{\Delta Y}{Y_{MAX}})]}, \tag{4}$$

where X_{MAX} and Y_{MAX} denote the maximum height and width of the document, respectively. α and β are the learnable parameters. In addition to relative positions, we also try to take region type into account. However, mainstream document understanding datasets lack region labels. Due to the domain gap, the pseudo region labels obtained by the existing layout analyzer are too noisy to use. It should be noted that equation 4 is easy to extend. We only need to add more terms to the exponential term to consider more influencing factors (e.g. when region labels are available)

We concatenate the mask sequences $\boldsymbol{m} \in \mathcal{R}^N$ corresponding to each token to get a mask matrix $\boldsymbol{M} \in \mathcal{R}^{N\times N}$, which is used to refine the original attention distribution of token i in layer l.

$$\boldsymbol{E} = \frac{\boldsymbol{Q}\boldsymbol{K}^\top}{\sqrt{d}} \tag{5}$$

$$ATT_i(\boldsymbol{M}, \boldsymbol{Q}, \boldsymbol{K}) = \frac{\boldsymbol{M}_i exp(\boldsymbol{E}_i)}{\sum_{i'=1}^{N} \boldsymbol{M}_{i'} exp(\boldsymbol{E}_{i'})}. \tag{6}$$

Based on the ATT obtained from equation 6, we calculate $\boldsymbol{h}_{i,l+1}$ according to equation 1. It should be **emphasized** that although tokens farther away from the current token will be masked with a higher probability, this does not mean that the model can never observe them. The purpose of the above mechanism is to make the model tend to make predictions using more reliable local features.

3.2.3 Gumbel Relaxation

To solve the non-differentiable problem of discrete mask operation, we introduce Gumbel softmax (Jang et al., 2017) to approximate the mask operation from the categorical distribution so that it can be optimized through backpropagation. Since we are dealing with a 2-class sampling problem, the original Gumbel softmax approximation is reduced to sigmoid-form as follows:

$$Gumbel(L) = \frac{exp((L+G_1)/\tau)}{exp((L+G_1)/\tau) + exp(G_2/\tau)}, \tag{7}$$

where $Gumbel(\cdot)$ is a continuous approximation of discrete mask m_i, L denotes $logits = log(\frac{p}{1-p})$ and p is calculated according to equation 3. G_1 and G_2 are two noises sampled from Gumbel distribution (Gumbel, 1954) independently. In addition, $\tau \in (0, +\infty)$ denotes the temperature parameter. With the decrease of τ, the approximate result $Gumbel(\cdot)$ will gradually tend to one hot. In the inference, we directly use $P_{\Delta X,Y}(m=1) = 0.5$ as the threshold of whether to mask a context word to ensure the consistency of inference results.

3.3 Cross-Document Information Awareness

For some hard cases, the context of the current document may not contain enough entity-related information. To embrace sufficient context to assist prediction, we propose the cross-document information awareness technique. Inverted index is introduced to deal with the fast retrieval of massive document context and cross-document attention is used to integrate entity-related information from different documents.

3.3.1 Efficient Retrieval Supported by Inverted Index

Efficient storage and retrieval of a large number of documents is a prerequisite for cross-document information awareness. To enable the model to efficiently retrieve different contexts containing the current token, we introduce the inverted index (Knuth, 1997) to manage the data efficiently. Specifically, each word w_i corresponds to a queue $\mathcal{Q}_i = \{\boldsymbol{h}_{i,N}^m\}_{m=1}^{|\mathcal{Q}|}$ that stores the contextualized representation of w_i in $|\mathcal{Q}|$ different documents. When the model queries the context of a token in other documents (e.g. COVID-19), we only need to return the queue corresponding to COVID-19 instead of traversing the entire dataset. However, as the model is updated, the contextualized representations stored in the queue will gradually become obsolete. It is obviously inefficient or even impossible to update the whole inverted index after the training of each batch. In order to solve this problem, we use the lazy update to maintain the queue. That is, after the current document $\mathcal{D}$ is encoded, we only update the representation $\boldsymbol{h}_{i,N}$ of each $w_i \in \mathcal{D}$ to the queue $\mathcal{Q}_i$. Finally, does storing these vectors incur excessive memory overhead? In fact, most words do not need to store more than 3 cross-document copies due to the long-tail effect. Words that appear many times are usually stop words, and it is useless to store too many of their representations. Therefore, we limit the maximum queue length to avoid useless storage. Overall, the queue size accounts for only about 5% to 10% of the encoder parameters.

3.3.2 Cross-Document Attention

Given the multi-modal embedding $\boldsymbol{h}_{i,N}$ of each word w_i outputted by layer N of encoder $\boldsymbol{enc}(\cdot)$, we integrate the information of different documents through the cross-document attention mechanism to assist prediction. First, we query the context representation queue $\mathcal{Q}_i = \{\boldsymbol{h}_{i,N}^m\}_{m=1}^{|\mathcal{Q}|} \in \mathcal{R}^{|\mathcal{Q}|\times d}$ corresponding to token w_i from the inverted index. Then $\boldsymbol{h}_{i,N}$ and $\boldsymbol{Q}$ are concatenated to get $\boldsymbol{H}^c \in \mathcal{R}^{(|\mathcal{Q}|+1)\times d}$, the input of cross-document attention. Then we transform $\boldsymbol{H}^c$ into the keys $\boldsymbol{K}^c \in \mathcal{R}^{(|\mathcal{Q}|+1)\times d}$, and the values $\boldsymbol{V}^c \in \mathcal{R}^{(|\mathcal{Q}|+1)\times d}$. We only calculate the query $\boldsymbol{q}^c$ of $\boldsymbol{h}_{i,N}$ for the efficiency of calculation. We obtain the representation $\boldsymbol{h}_i$ of the word w_i by integrating cross document information as follows:

$$\boldsymbol{h}_i = ATT(\boldsymbol{q}^c, \boldsymbol{K}^c)\boldsymbol{V}^c \tag{8}$$

$$ATT(\boldsymbol{q}^c, \boldsymbol{K}^c) = \text{Softmax}(\frac{\boldsymbol{q}^c \boldsymbol{K}^{c,\top}}{\sqrt{d}}). \tag{9}$$

Finally, a linear classifier $\boldsymbol{\eta}(\cdot) : \mathbb{R}^d \rightarrow \mathbb{R}^{N_c}$ optimized by cross entropy transforms $\boldsymbol{h}_i$ to its corresponding entity label $c_{w_i} \in \mathcal{C}$.

4 Experimental Setup

In this section, we describe the datasets for training and evaluating the proposed method. We also detail the baseline models for comparison. Finally, we clarify the implementation details and hyperparameter configuration of our method.

4.1 Datasets

The effectiveness of the proposed method is not limited to a particular language. We conducted experiments on two well-known document understanding datasets consisting of eight languages to show the universality of our method.

FUNSD (Jaume et al., 2019) is an English dataset for form understanding in noisy scanned documents. It consists of 199 manually labeled real documents from different fields such as marketing, advertising, and scientific reports. Each entity is annotated by a label (i.e., question, answer, header, or other) following the BIO schema, and a bounding box indicating the 2D position in the document. The dataset is split into 149 training forms and 50 testing forms.

XFUND (Xu et al., 2021b) is a multilingual form understanding dataset, which extends the FUNSD dataset to 7 other languages including Chinese, Japanese, Spanish, French, Italian, German, and Portuguese. Each language includes 199 forms, where the training set includes 149 forms and the test set includes 50 forms.

4.2 Comparison Methods

To evaluate the effectiveness of our method, we select the following SOTA multilingual NER models for comparison. The first two baseline only use textual modal as self-supervised signal, while multimodal pretraining is used in the last baseline and achieves SOTA in document understanding task.

XLM-RoBERTa (Conneau et al., 2020) is a Transformer-based masked language model pretrained on one hundred languages, with more than two terabytes of data.

InfoXLM (Chi et al., 2021) is a cross-lingual pretrained model based on an information-theoretic framework. It formulates pretraining as maximizing mutual information between multilingual multi-granularity texts.

LayoutXLM (Xu et al., 2021b) is a multimodal pretrained model, which takes the information of three modalities (text, layout, and image) as input. The carefully designed cross-modal alignment pretraining objectives improve the effectiveness of visually-rich document modeling.

4.3 Implementation Details

We use the AdamW (2019) as the optimizer, with a learning rate of $5e-5$ and batch size of 4 for all datasets. The length of the queue is selected among {5, 10, 15, 20} and experiments show that 10 is the best choice. We initialize α and β to be 0.2 and 1.0 respectively. τ is selected among {0.15, 0.25, 0.35} and we choose the best one. We use the base version for all the pretrained models. All experiments are conducted using an NVIDIA GeForce RTX 3090 with 24GB memory. All experimental results are the average of three runs based on the Pytorch framework.

5 Results and Analysis

In this section, we present the experimental results on two well-known document understanding datasets to show the effectiveness of our method.

5.1 Main Results

Table 1 reports model performance on FUNSD, and XFUND datasets, which shows that the proposed method achieves state-of-the-art results in eight different languages. For visually-rich document understanding, the key information is presented in multiple modalities, such as text, layout, vision. However, XLM-RoBERTa and InfoXLM only model a single textual modal, consequently underperforming the multi-modal LayoutXLM baseline and our method by a large margin. Benefitting from (1) the irrelevant redundant information filtering supported by attention mask mechanism and (2) valuable entity-related context provided by the cross-document information awareness technique, the model can efficiently integrate entity-related information to make predictions. As a result, the proposed method outperforms LayoutXLM in eight different languages. This also shows that the effectiveness of the proposed semantic enhancement method is not limited to a specific language.

Dataset	Language	XLM-RoBERTa (Conneau et al., 2020)			InfoXLM (Chi et al., 2021)			LayoutXLM (Xu et al., 2021b)			**Ours**		
		Prec.	Rec.	F_1	Prec.	Rec.	F_1	Prec.	Rec.	F_1	Prec.	Rec.	F_1
FUNSD	English	-	-	0.667	-	-	0.685	0.784	0.817	0.800	0.800	0.828	**0.814**
XFUND	Chinese	-	-	0.877	-	-	0.887	0.848	0.920	0.882	0.871	0.932	**0.901**
	Japanese	-	-	0.776	-	-	0.787	0.733	0.854	0.789	0.760	0.851	**0.803**
	Spanish	-	-	0.611	-	-	0.623	0.713	0.752	0.732	0.736	0.760	**0.748**
	French	-	-	0.674	-	-	0.702	0.762	0.794	0.778	0.778	0.807	**0.792**
	Italian	-	-	0.669	-	-	0.675	0.774	0.833	0.803	0.791	0.852	**0.820**
	German	-	-	0.681	-	-	0.706	0.771	0.824	0.797	0.783	0.836	**0.809**
	Portuguese	-	-	0.682	-	-	0.701	0.759	0.803	0.780	0.767	0.816	**0.790**
Avg.	ALL	-	-	0.705	-	-	0.721	0.768	0.825	0.795	**0.786**	**0.835**	**0.810**

Table 1: Main results on two well-known document understanding datasets.

Language	w/o. CD			w/o. AM			**Ours**		
	Prec.	Rec.	F_1	Prec.	Rec.	F_1	Prec.	Rec.	F_1
English	0.788	0.825	0.806	0.785	0.827	0.805	0.800	0.828	**0.814**
Chinese	0.868	0.934	0.900	0.869	0.928	0.897	0.871	0.932	**0.901**
Japanese	0.743	0.847	0.792	0.749	0.850	0.797	0.760	0.851	**0.803**
Spanish	0.718	0.753	0.735	0.717	0.755	0.736	0.736	0.760	**0.748**
French	0.763	0.802	0.782	0.767	0.811	0.789	0.778	0.807	**0.792**
Italian	0.785	0.844	0.814	0.776	0.846	0.809	0.791	0.852	**0.820**
German	0.779	0.820	0.799	0.766	0.826	0.795	0.783	0.836	**0.809**
Portuguese	0.757	0.809	0.782	0.766	0.808	0.787	0.767	0.816	**0.790**
Average	0.775	0.829	0.801	0.774	0.831	0.802	**0.786**	**0.835**	**0.810**

Table 2: Abalation study of our method.

5.2 Ablation Study

To study the contribution of each component in the proposed method, we conduct ablation experiments on the two datasets and display the results in Table 2. The results show that the model performance is degraded if the learnable attention mask (AM) is removed, indicating that the redundant information in the document will distract the model from focusing on the local regions with stronger semantic relevance. In addition, cross-document information awareness (CD) provides diverse contexts for key information extraction. Without CD, some hard cases where the entity-related information is insufficient can not be recognized accurately. Consequently, the overall performance will be negatively affected. It is worth noting that the proposed AM and CD are effective in all languages, which also shows the generality and practical value of the proposed method.

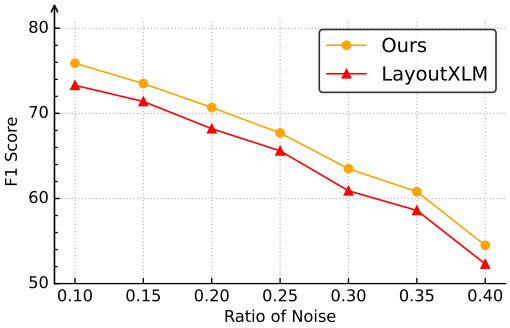

Figure 3: Model performance on the perturbated FUNSD dataset.

5.3 Robustness Analysis

In real-world applications, we inevitably encounter low-quality input documents. Due to various factors such as occlusion, focus and angular deformations, optical character recognition (OCR) of those documents often yield unsatisfactory results, which will negatively impact the information extraction. To exhaustively evaluate the robustness of the model in real scenarios, we apply TextFlint (Wang et al., 2021b), a robustness evaluation platform, to perturb the original dataset to simulate OCR errors in real-world applications. Specifically, we use OCR error transformation in TextFlint

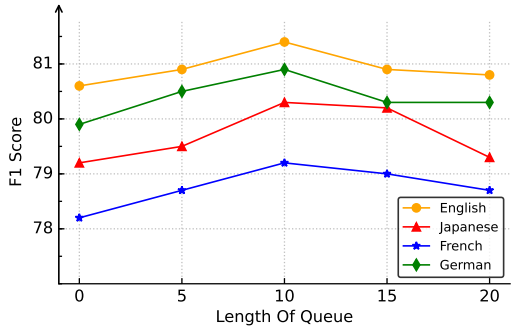

Figure 4: Model performance with different lengths of the queue in 4 randomly selected languages.

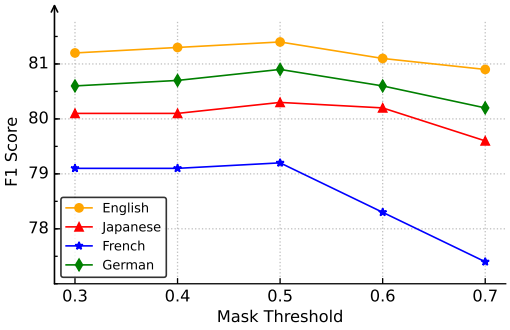

Figure 5: Model performance with different mask thresholds in 4 randomly selected languages.

to disturb the document. For each entity in the document, we perturb a certain proportion of words in its context and evaluate the prediction results. From Figure 3 we can see that the proposed method consistently outperforms baselines under the different ratios of OCR errors in context. Benefitting from the proposed cross-document information awareness technique, even if the context of entities in the current document is severely damaged, the model can still use the context information in related documents to assist in prediction. Therefore, the proposed model has better robustness in real scenarios.

5.4 Efficiency Analysis

Although the introduction of inverted index enables us to retrieve large-scale related documents efficiently, *a question worthy of discussion is whether storing the representations of each token will occupy too much memory space?* We answer the question by analyzing the maximum queue length L, a key parameter affecting memory consumption. From Figure 4 we can observe that optimal performance can be achieved by memorizing no more than 10 token representations of each target entity in related documents. In addition, nearly 70% of tokens in the two datasets appear no more than 3 times due to the long-tail token distribution. Therefore, the storage of cross-document information does not bring too high a memory overhead. In fact, too much information from other documents is not necessarily leading to better results. When the maximum queue length L exceeds 10, further increasing L will make a large number of meaningless stop words and some other outdated representations memorized, which leads to the decline of model performance.

5.5 Impact of Context Sparsity

In the inference, we directly use $P_{\Delta X,Y}(m = 1) = \theta$ as the threshold of whether to mask a context word to ensure the consistency of inference results. The threshold θ corresponds to the sparsity of the context that the model can observe. The larger the θ, the fewer tokens can be observed. To analyze the impact of context sparsity, we conduct experiments on four randomly selected languages. As can be seen from Fig. 5, initially, as the threshold increases, redundant noise from different regional contexts is continuously masked. The performance continues to improve, reaching the maximum when $\theta = 0.5$. After that, further increasing θ will cause more useful contexts in the same region to be masked, and the performance declines rapidly. This also confirms the view that useful context information is mainly distributed in local regions.

6 Conclusions

In this work, we introduce a cross-document semantic enhancement method to improve entity recognition from visually-rich documents. The proposed learnable attention mask mechanism effectively filters redundant irrelevant information in the current document, which reduces the risk of overfitting spurious features. Cross-document information awareness enriches sufficient entity-related context to improve predictions. The proposed method can be regarded as a plug-in, which can be added to any existing document understanding model and improve prediction. Experimental results show that the proposed method outperforms the existing state-of-the-art methods in documents of different languages and is more robust in real scenarios.

Acknowledgements

The authors wish to thank the anonymous reviewers for their helpful comments. This research was sponsored by Hikvision Cooperation Fund.

References

Galal M Binmakhashen and Sabri A Mahmoud. 2019. Document layout analysis: A comprehensive survey. *ACM Computing Surveys (CSUR)*, 52(6):1–36.

Zewen Chi, Li Dong, Furu Wei, Nan Yang, Saksham Singhal, Wenhui Wang, Xia Song, Xian-Ling Mao, Heyan Huang, and Ming Zhou. 2021. InfoXLM: An information-theoretic framework for cross-lingual language model pre-training. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 3576–3588, Online. Association for Computational Linguistics.

Alexis Conneau, Kartikay Khandelwal, Naman Goyal, Vishrav Chaudhary, Guillaume Wenzek, Francisco Guzmán, Edouard Grave, Myle Ott, Luke Zettlemoyer, and Veselin Stoyanov. 2020. Unsupervised cross-lingual representation learning at scale. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 8440–8451, Online. Association for Computational Linguistics.

Robert Geirhos, Jörn-Henrik Jacobsen, Claudio Michaelis, Richard Zemel, Wieland Brendel, Matthias Bethge, and Felix A Wichmann. 2020. Shortcut learning in deep neural networks. *Nature Machine Intelligence*, 2(11):665–673.

Emil Julius Gumbel. 1954. Statistical theory of extreme values and some practical applications : A series of lectures.

He Guo, Xiameng Qin, Jiaming Liu, Junyu Han, Jingtuo Liu, and Errui Ding. 2019a. Eaten: Entity-aware attention for single shot visual text extraction. In *2019 International Conference on Document Analysis and Recognition (ICDAR)*, pages 254–259. IEEE.

Maosheng Guo, Yu Zhang, and Ting Liu. 2019b. Gaussian transformer: A lightweight approach for natural language inference. In *National Conference on Artificial Intelligence*.

Eric Jang, Shixiang Gu, and Ben Poole. 2017. Categorical reparameterization with gumbel-softmax. In *5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Conference Track Proceedings*. OpenReview.net.

G. Jaume, H. Kemal Ekenel, and J. Thiran. 2019. Funsd: A dataset for form understanding in noisy scanned documents. In *2019 International Conference on Document Analysis and Recognition Workshops (ICDARW)*, volume 2, pages 1–6, Los Alamitos, CA, USA. IEEE Computer Society.

H. Tolga Kahraman, Seref Sagiroglu, and Ilhami Colak. 2010. Development of adaptive and intelligent web-based educational systems. In *2010 4th International Conference on Application of Information and Communication Technologies*, pages 1–5.

Anoop Raveendra Katti, Christian Reisswig, Cordula Guder, Sebastian Brarda, Steffen Bickel, Johannes Höhne, and Jean Baptiste Faddoul. 2018. Chargrid: Towards understanding 2d documents. *arXiv preprint arXiv:1809.08799*.

Donald E. Knuth. 1997. *The Art of Computer Programming, Vol. 1: Fundamental Algorithms*, third edition. Addison-Wesley, Reading, Mass.

Hermann Kroll, Jan Pirklbauer, Jan-Christoph Kalo, Morris Kunz, Johannes Ruthmann, and Wolf-Tilo Balke. 2021. Narrative query graphs for entity-interaction-aware document retrieval. In *Towards Open and Trustworthy Digital Societies*, pages 80–95, Cham. Springer International Publishing.

Guillaume Lample, Miguel Ballesteros, Sandeep Subramanian, Kazuya Kawakami, and Chris Dyer. 2016. Neural architectures for named entity recognition. *arXiv preprint arXiv:1603.01360*.

Xiaojing Liu, Feiyu Gao, Qiong Zhang, and Huasha Zhao. 2019. Graph convolution for multimodal information extraction from visually rich documents. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 2 (Industry Papers)*, pages 32–39, Minneapolis, Minnesota. Association for Computational Linguistics.

Ilya Loshchilov and Frank Hutter. 2019. Decoupled weight decay regularization. In *International Conference on Learning Representations*.

Ruotian Ma, Yiding Tan, Xin Zhou, Xuanting Chen, Di Liang, Sirui Wang, Wei Wu, and Tao Gui. 2022. Searching for optimal subword tokenization in cross-domain ner. In *Proceedings of the Thirty-First International Joint Conference on Artificial Intelligence, IJCAI-22*, pages 4289–4295. International Joint Conferences on Artificial Intelligence Organization. Main Track.

Yongqiang Mou, Lei Tan, Hui Yang, Jingying Chen, Leyuan Liu, Rui Yan, and Yaohong Huang. 2020. Plugnet: Degradation aware scene text recognition supervised by a pluggable super-resolution unit. In *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part XV 16*, pages 158–174. Springer.

Xipeng Qiu, Tianxiang Sun, Yige Xu, Yunfan Shao, Ning Dai, and Xuanjing Huang. 2020. Pre-trained models for natural language processing: A survey. *CoRR*, abs/2003.08271.

Jiapeng Wang, Chongyu Liu, Lianwen Jin, Guozhi Tang, Jiaxin Zhang, Shuaitao Zhang, Qianying Wang, Yaqiang Wu, and Mingxiang Cai. 2021a. Towards robust visual information extraction in real world: New dataset and novel solution. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 35, pages 2738–2745.

Xiao Wang, Shihan Dou, Limao Xiong, Yicheng Zou, Qi Zhang, Tao Gui, Liang Qiao, Zhanzhan Cheng, and Xuanjing Huang. 2022. MINER: Improving out-of-vocabulary named entity recognition from an information theoretic perspective. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 5590–5600, Dublin, Ireland. Association for Computational Linguistics.

Xiao Wang, Qin Liu, Tao Gui, Qi Zhang, Yicheng Zou, Xin Zhou, Jiacheng Ye, Yongxin Zhang, Rui Zheng, Zexiong Pang, Qinzhuo Wu, Zhengyan Li, Chong Zhang, Ruotian Ma, Zichu Fei, Ruijian Cai, Jun Zhao, Xingwu Hu, Zhiheng Yan, Yiding Tan, Yuan Hu, Qiyuan Bian, Zhihua Liu, Shan Qin, Bolin Zhu, Xiaoyu Xing, Jinlan Fu, Yue Zhang, Minlong Peng, Xiaoqing Zheng, Yaqian Zhou, Zhongyu Wei, Xipeng Qiu, and Xuanjing Huang. 2021b. TextFlint: Unified multilingual robustness evaluation toolkit for natural language processing. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing: System Demonstrations*, pages 347–355, Online. Association for Computational Linguistics.

Yang Xu, Yiheng Xu, Tengchao Lv, Lei Cui, Furu Wei, Guoxin Wang, Yijuan Lu, Dinei Florencio, Cha Zhang, Wanxiang Che, Min Zhang, and Lidong Zhou. 2021a. LayoutLMv2: Multi-modal pre-training for visually-rich document understanding. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 2579–2591, Online. Association for Computational Linguistics.

Yiheng Xu, Minghao Li, Lei Cui, Shaohan Huang, Furu Wei, and Ming Zhou. 2020. Layoutlm: Pre-training of text and layout for document image understanding. In *Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery amp; Data Mining*, KDD '20, page 1192–1200, New York, NY, USA. Association for Computing Machinery.

Yiheng Xu, Tengchao Lv, Lei Cui, Guoxin Wang, Yijuan Lu, Dinei Florêncio, Cha Zhang, and Furu Wei. 2021b. Layoutxlm: Multimodal pre-training for multilingual visually-rich document understanding. *CoRR*, abs/2104.08836.

Wenwen Yu, Ning Lu, Xianbiao Qi, Ping Gong, and Rong Xiao. 2021. Pick: Processing key information extraction from documents using improved graph learning-convolutional networks. In *2020 25th International Conference on Pattern Recognition (ICPR)*, pages 4363–4370. IEEE.

Peng Zhang, Yunlu Xu, Zhanzhan Cheng, Shiliang Pu, Jing Lu, Liang Qiao, Yi Niu, and Fei Wu. 2020. Trie: End-to-end text reading and information extraction for document understanding. In *Proceedings of the 28th ACM International Conference on Multimedia*, pages 1413–1422.

Jun Zhao, Tao Gui, Qi Zhang, and Yaqian Zhou. 2021. A relation-oriented clustering method for open relation extraction. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 9707–9718, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

STAD: Self-Training with Ambiguous Data for Low-Resource Relation Extraction

Junjie Yu[1] **Xing Wang**[2] **Jiangjiang Zhao**[3] **Chunjie Yang**[3] **Wenliang Chen**[1]
[1]Institute of Artificial Intelligence, School of Computer Science and Technology, Soochow University, China
[2]Tencent AI Lab, Shenzhen, China
[3]China Mobile Online Marketing and Services Center, China
jjyu@stu.suda.edu.cn, brightxwang@tencent.com, wlchen@suda.edu.cn
{zhaojiangjiang, yangchunjie}@cmos.chinamobile.com

Abstract

We present a simple yet effective self-training approach, named as STAD, for low-resource relation extraction. The approach first classifies the auto-annotated instances into two groups: confident instances and uncertain instances, according to the probabilities predicted by a teacher model. In contrast to most previous studies, which mainly only use the confident instances for self-training, we make use of the uncertain instances. To this end, we propose a method to identify ambiguous but useful instances from the uncertain instances and then divide the relations into candidate-label set and negative-label set for each ambiguous instance. Next, we propose a set-negative training method on the negative-label sets for the ambiguous instances and a positive training method for the confident instances. Finally, a joint-training method is proposed to build the final relation extraction system on all data. Experimental results on two widely used datasets SemEval2010 Task-8 and Re-TACRED with low-resource settings demonstrate that this new self-training approach indeed achieves significant and consistent improvements when comparing to several competitive self-training systems.[1]

1 Introduction

Relation Extraction (RE) is a fundamental task in Information Extraction, which aims to obtain a predefined semantic relation between two entities in a given sentence (Zhou et al., 2005). In recent years, fine-tuning on the downstream RE tasks with pre-trained models (Soares et al., 2019) has achieved significant progress with the rapid development of the "Pre-train and Fine-tune" Paradigm (Devlin et al., 2019) which leverages large-scale unlabeled data. However, RE still suffers from the data scarcity problem. For most RE tasks, due to the task-specific definition of relations, the lack of customized annotation data poses great challenge for the supervised RE (Hendrickx et al., 2010; Zhang et al., 2017). Meanwhile, manually labeling large-scale RE data is extremely time-consuming, expensive, and laborious. As an alternative, automatically building annotated data for RE attracts a lot of attention in the research community (Mintz et al., 2010; Luo et al., 2019; Yu et al., 2020).

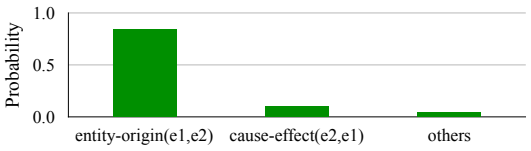

Most of birdtown's earliest residents hailed from eastern european countries.

(a) Confident instance

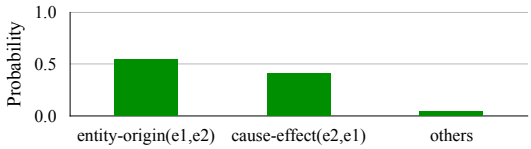

The taste is from the ginseng , not alcohol .

(b) Uncertain instance

Figure 1: Two examples of auto-annotated instances. For simplicity, we list two detailed relations and use "others" to represent the other relations.

Self-training is a simple and effective approach to build auto-annotated data (Xie et al., 2020; Du et al., 2021). The idea is to use a teacher model trained on human-annotated data to annotate the additional unlabeled data. Then the human-annotated data is combined with some instances selected from the auto-annotated data to train a student model. In this paper, we follow the self-training framework to improve **Low-Resource RE**, which is closer to practical situations where the task starts with a small seed set of human-annotated data.

In previous studies, researchers often select the auto-annotated instances with high confidence,

[1] Code is publicly available at https://github.com/jjyunlp/STAD

Proceedings of the 29th International Conference on Computational Linguistics, pages 2044–2054
October 12–17, 2022.

named as *confident instance*, and have achieved a certain success (Qian et al., 2009; Du et al., 2021). Figure 1(a) shows an example of confident instance, where the teacher model can easily classify it as relation "entity-origin(e1,e2)" with the clue offered by "hailed from". Therefore, we first follow this kind of self-training solutions to train the RE system. However, in the preliminary experiments, we find that *some relations might have similar expressions among instances*, which makes the teacher model confused. As a result, for the uncertain instances, the teacher model gives similar high probabilities to some relations or assigns low probabilities to all the relations. An example of uncertain instance is shown in Figure 1(b), where the teacher model predicts the instance as relation "entity-origin(e1,e2)" with a probability of 56%, as "cause-effect(e2,e1)" (the ground truth label) with 42%, and as other relations with only 2%. It is hard to distinguish between the first two relations as the expression "... is from ..." is often used for both. In previous studies, the uncertain instances are often discarded due to the confusion. However, we argue that ignoring all the uncertain instances may not be appropriate since they may contain useful information. For example, it is a good clue that the answer is one of the first two relations with a probability 98% for the instance in Figure 1(b).

Ideally, we would wish to make full use of all the auto-annotated instances to improve the RE system. But it is very hard due to the confusion problem. Therefore, we split the uncertain instances into two groups: *ambiguous set* and *hard set*. The ambiguous set contains the instances for which the teacher model predicts similar high probabilities on some relations, while the hard set contains those that the teacher model assigns low probabilities to all the relations. In this paper, we focus on the ambiguous set and propose an approach to use the ambiguous instances and the confident instances to improve the system of low-resource RE.

In our approach, we tackle two main issues when exploiting the ambiguous instances: 1) how to identify the candidate labels of ambiguous instances; 2) how to train a new model with the ambiguous instances. For the first issue, we adopt a probabilistic accumulation approach to obtain a set of relations, that is, a set of candidate labels containing the vast majority of probabilities, and then label ambiguous instances with the candidate labels in the set. To deal with the second issue, we make an assumption: *For the ambiguous instances, the teacher model does not know which relation is the exact answer, but it does know that #1) the answer is (with high probability) in the candidate-label set (likes the first two relations in Figure 1(b)) and #2) the answer is not one of the relations which are with very low probabilities (likes "others" in Figure 1(b)).* Under this assumption, we treat the ambiguous instances as partially-labeled instances (Yan and Guo, 2020), where the relations are divided into a candidate-label set and a negative-label set for each ambiguous instance. Then, we propose a novel set-negative training method to learn from the ambiguous instances by the negative-label set. In order to build the final relation extraction system, we further propose a joint training method which supports both positive and set-negative training.

Our main contributions are as follows:

- We propose STAD, a self-training framework, which supports learning from ambiguous data.
- We propose a method to classify the auto-annotated instances generated from teacher model into confident set, ambiguous set and hard set. The ambiguous instances are then treated as partially-labeled, which can reduce the effect of confused expressions. To our best knowledge, it is the first time that partial labeling is used to tag the auto-annotated instances in RE.
- In order to exploit the auto-annotated instances properly, we propose set-negative training for the ambiguous instances and positive training for the confident instances. The set-negative training method can utilize the information that the answer is not in the set of the relations which have very low probabilities predicted by the teacher model. And we further propose a joint training method to combine positive and set-negative training methods to build the final RE system.

To verify the effectiveness of our approach, we conduct experiments on two widely used datasets SemEval2010 Task-8 and Re-TACRED with low-resource settings. Experimental results show that our proposed approach significantly outperforms the conventional self-training system which only samples confident instances and other baseline systems.

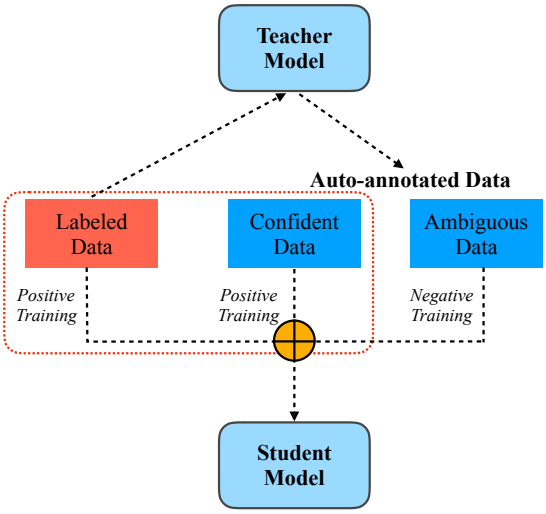

Figure 2: Framework of our approach. The red dotted rectangle is the conventional self-training approach that neglects the ambiguous data in auto-annotated data.

2 Our Approach

We first briefly introduce the relation extraction task as well as the self-training framework frequently used in the previous studies (Zhou et al., 2008). Then, we propose an algorithm to classify the auto-annotated instances into three groups: confident data, ambiguous data and hard data. Next, we introduce three tagging modes for ambiguous data. Finally, we propose the training method to use both the confident and ambiguous data. The framework of the proposed approach is shown in Figure 2.

2.1 Self-Training for Relation Extraction

2.1.1 Relation Extraction

Fine-tuning on a pre-trained model, e.g., BERT, with a task-specific classifier is a common practice for downstream NLP tasks (Devlin et al., 2019). Following Soares et al. (2019), the RE model is composed of a BERT encoder and a relation classification layer. Entity markers ('[E1] head entity [/E1]' and '[E2] tail entity [/E2]') are inserted into input tokens to wrap the entities. Concretely, the output representation of two start entity markers ('[E1]' and '[E2]') are concatenated as the representation of entities. Finally, the representations are used as input for the relation classification layer.

Formally, the output representation of an input instance x after BERT is $\mathbf{h} = \mathbf{h}_{[E1]} \oplus \mathbf{h}_{[E2]}$. Then, the output probability distribution for M relations $p = [p_1, p_2, ..., p_M]$ is computed by the relation classification layer:

$$p = f(x) = \text{Softmax}(\mathbf{W}\mathbf{h} + \mathbf{b}), \quad (1)$$

where $\mathbf{W}$ and $\mathbf{b}$ are model parameters.

During training, each instance x from the human-annotated data is labeled with a one-hot label vector y: a single 1 value for the ground-truth label and 0 values for other labels. Then, the positive training is performed to calculate the cross entropy loss:

$$\mathcal{L}_{PT}(f(x), y) = -\sum_{i=1}^{M} y_i \log p_i, \quad (2)$$

where M is the number of relations, and y_i and p_i are the label and prediction probability of ith relation, respectively.

2.1.2 Self-Training

Generally, in the self-training framework, unlabeled instances are labeled by the teacher model to build the auto-annotated data. As shown in Figure 2, the common-used flow of self-training is performed in the following steps: (1) use the human-annotated data to train a teacher model; (2) use the teacher model to conduct label prediction for unlabeled data; (3) select confident auto-annotated instances via a pre-defined probability threshold (described in Sec. 2.2) and the rest are uncertain instances; (4) combine confident auto-annotated data and human-annotated data to train a student model (the red dotted rectangle in Figure 2).

And in step (3), the remaining uncertain instances are considered to be useless. However, as described in Sec. 1, the uncertain instances (e.g., the example in Figure 1(b)) might contain useful information.

2.2 Instance Classification

As shown in Algorithm 1, we propose a method to classify the auto-annotated data into confident data (Line 14), ambiguous data (Line 16) and hard data (Line 18). In detail, considering a classification task with M classes, we first set a probability threshold T. After sorting the classes by the predicted probability from large to small, we dynamically accumulate the probability of top classes until the score is larger than T (Line 6-10 in Algorithm 1). Then, we move on to the classification of instances:

Confident instance. Using a probability threshold to select confident instances is a common practice in self-training. We also add an instance into the confident data when the highest prediction probability exceeds the threshold T, as the teacher

Algorithm 1 Instance Classification

Input: auto-annotated data $\mathbf{D_{auto}} = \{x, P, Y\}$ containing sentence x, prediction distribution P and its corresponding relations Y.
Hyper Parameter: probability threshold T.
Output: confident data $\mathbf{D_{con}}$, ambiguous data $\mathbf{D_{amb}}$ and hard data $\mathbf{D_{hard}}$

1: **for** $(x, P) \in \mathbf{D_{auto}}$ **do**
2: Let $score = 0.0$.
3: Let candidate-label set $C^+ = \{\}$
4: Sort P from large to small
5: Sort Y by the order in P
6: **for** $(p, y) \in P, Y$ **do**
7: $score \leftarrow score + p$.
8: Append y to C^+.
9: **if** $score > T$ **then**
10: **break**
11: **end if**
12: **end for**
13: **if** len(C^+) == 1 **then**
14: Append (x, C^+) to $\mathbf{D_{con}}$
15: **else if** len(C^+) $\leq M - 1$ **then**
16: Append (x, C^+) to $\mathbf{D_{amb}}$
17: **else**
18: Append x to $\mathbf{D_{hard}}$
19: **end if**
20: **end for**
21: **return** $\mathbf{D_{con}}, \mathbf{D_{amb}}, \mathbf{D_{hard}}$

makes a certain prediction. In Line 13-14 of Algorithm 1, instances with only one candidate label are added into the confident data $\mathbf{D_{con}}$.

Ambiguous instance. Besides the confident data, according to our observations, the teacher model may give relatively high probabilities to multiple relations. In order to make full use of ambiguous instances to improve the RE system, we adopt a greedy probability accumulation method to identify ambiguous data from uncertain data. Specifically, the size of candidate-label set is at least two and can be at most $M - 1$ (Line 15-16 in Algorithm 1). Obviously, as the size of candidate-label set increases, instances carry less information.

Hard instance. We also consider an extreme situation that the sum probability of top $M - 1$ relations is still lower than T. In other words, the lowest predicted probability is larger than $1 - T$. We treat such examples as hard instances because the teacher model is confused about all relations.

Mode	Ent-Ori	Cau-Eff	Others
Probability	0.56	0.42	0.02
Hard Label	1	0	0
Soft Label	0.56	0.42	0.02
Partial Label	1	1	0

Table 1: An example of three tagging modes with given predicted probability distribution.

2.3 Instance Label Tagging Mode

After identifying confident and ambiguous data from the auto-annotated data, we now have three training sets: a small seed set of human-annotated data $\mathbf{D_{hum}}$, a confident auto-annotated data $\mathbf{D_{con}}$, and an ambiguous auto-annotated data $\mathbf{D_{amb}}$. For the human-annotated data and confident data, we take the one-hot label vector as described in Sec. 2.1.1 to tag the data. For the ambiguous data, a variety of methods can be used to tag the instance. As shown in Table 1, given the probability distributions predicted by the teacher model, hard label mode assigns an exact label (the label with highest prediction probability) with a one-hot vector (Lee, 2013) while soft label mode (Xie et al., 2020) adopts probability distributions as labels.

In this work, we propose to use the partial label mode (Yan and Guo, 2020) to tag ambiguous instances. As the teacher model gives relatively high probabilities to relations in the candidate-label set C^+ (Line 3 in Algorithm 1), partial label mode assigns each ambiguous instance with a set of candidate relations and treats each candidate relation equally to form a multi-hot label vector. As the example shown in Table 1, the instance is labeled as [1, 1, 0] because the first two relations are in candidate-label set.

2.4 Model Training

As described in Sec. 2.1.1, we can directly utilize Eqn. 2 to train data tagged in the hard or soft label mode. In this section, we focus on training ambiguous data under partial label mode and describe how to train model on a mixed data.

2.4.1 Set-Negative Training

Inspired by negative training (Kim et al., 2019) which trains noisy data in a negative way by selecting a random label excepting the tagged label, we propose the set-negative training method to train ambiguous data. For the ambiguous data, as described in Sec. 2.2, we first split relations into two

sets: candidate-label set C^+ and negative-label set C^-, where C^- includes the relations that are not in C^+. With the Assumption #2 described in Sec. 1, we are confident that the answer is not in C^-. Hence, we randomly select one label from C^- as a negative label for each ambiguous instance in every iteration to update the model in a negative way.

Formally, the loss function for the ambiguous instances under set-negative training is:

$$\mathcal{L}_{NT}(f(x), y) = -\sum_{i=1}^{M} y_i \log(1 - p_i), \quad (3)$$

where the one-hot label y is dynamically changed by randomly selecting a negative label from C^- for every iteration.

2.4.2 Joint Training

After adopting the set-negative training for ambiguous data, another problem is how to train on $\mathbf{D_{hum}}$, $\mathbf{D_{con}}$, and $\mathbf{D_{amb}}$ simultaneously. Formally, we need to design a unified training framework to support both positive training (Eqn. 2) and set-negative training (Eqn. 3). To this end, we first introduce a flag variable z to represent whether current input instance is partially labeled or not:

$$z = \begin{cases} 1 & \text{if partially labeled,} \\ 0 & \text{others.} \end{cases} \quad (4)$$

Then, the following unified loss function can be directly used for joint training:

$$\mathcal{L}(f(x), y) = -\sum_{i=1}^{M} y_i \log |z - p_i|, \quad (5)$$

where $|*|$ is the absolute value. When the input instance is partially labeled, this function is equivalent to Eqn. 3, otherwise it is equivalent to Eqn. 2.

3 Experiments

In this section, we carry out experiments to show the effectiveness of the proposed approach. We first introduce settings of datasets and parameters. Then, we describe comparison systems used in this work. After that, we present the overall evaluation results, followed by further analysis.

3.1 Datasets

We conduct our experiments on two widely used relation extraction datasets: SemEval 2010 Task-8 (SemEval) and Re-TACRED. The brief information of two datasets are as follows:

- SemEval: A classical dataset in relation extraction which contains 10,717 annotated sentences covering 9 relations with two directions and one special relation "no_relation" (Hendrickx et al., 2010).
- Re-TACRED: A repaired version of TACRED (Zhang et al., 2017) proposed by Stoica et al. (2021) who re-annotated the data and refined relation definitions. In total, it contains 91,467 sentences covering 40 relations (also including a "no_relation" class).

To test on the **low-resource scenario**, we randomly sample 20 instances for each relation from the original training set as the human-annotated data and use the remaining instances as unlabeled data. [2] Meanwhile, we also rebuild the development set by sampling 10 instances for each relation from the original development set which is more realistic. Besides, the special "no_relation" type in the Re-TACRED is excluded as it occupies more than 66% of instances. The statistics of two low-resource datasets are shown in Table 2.

Data	Rel	Train	Dev	Test	Unlabel
SemEval	19	380	190	2,717	6,076
Re-TACRED	39	780	390	5,648	18,938

Table 2: Statistics of SemEval and Re-TACRED with the low-resource setting.

3.2 Parameter Settings

Relation Extraction Model Training To train the relation extraction model, we use the "Entity marker+Entity start state" architecture proposed in Soares et al. (2019) with $\texttt{BERT}_{\texttt{base}}$ (Devlin et al., 2019). During training, we follow Devlin et al. (2019) to select the learning rate among {2e-5, 3e-5, 5e-5}, batch size among {16, 32} and adopt the hyper parameters with the best performance on development set. For other parameters, we simply follow the settings used in Soares et al. (2019) to conduct experiments. We train the model in 20 epochs with early stop strategy to relieve the overfitting problem. Besides, we run each system

[2] For any relation contains less than 20 instances in the original training set, we directly repeat some instances to keep the data balanced.

#	System	Auto-annotated		Micro F1		
		$\mathbf{D_{con}}$	$\mathbf{D_{amb}}$	SemEval	Re-TACRED	Avg. Score
1	SUPERVISED	×	×	$77.3_{\pm 0.8}$	$77.5_{\pm 1.3}$	77.4
2	SELF-TRAINING	✓	×	$79.2_{\pm 1.5}$	$80.5_{\pm 0.4}$	79.9
3	HARD-LABEL	✓	✓	$78.0_{\pm 1.2}$	$78.2_{\pm 1.4}$	78.1
4	SOFT-LABEL	✓	✓	$79.1_{\pm 0.8}$	$79.8_{\pm 2.0}$	79.5
5	STAD (Ours.)	✓	✓	$\mathbf{80.0}_{\pm 1.0}$	$\mathbf{81.6}_{\pm 0.3}$	**80.8**

Table 3: Results on the test set of SemEval and Re-TACRED with low-resource settings in terms of micro F1 score. $\mathbf{D_{con}}$ and $\mathbf{D_{amb}}$ denote confident data and ambiguous data, respectively. We run experiment five times with different random seeds and report the mean of the micro F1 scores (Micro-F1) and its standard deviation score. Some additional experiments with random data samples are included in Appendix A.

five times with random seeds and report the mean of the micro F1 score (Micro-F1) with a standard deviation score.

Auto-Annotated Data Building The probability threshold is an important hyper parameter which is used to select confident data and determine the candidate-label sets of ambiguous data. In detail, we search probability threshold T among {0.95, 0.90, 0.85, 0.80} on development set to select the best confident data.

3.3 Comparison Systems

In order to make a fair comparison, all systems in this work use the BERT-based fine-tuning model proposed by Soares et al. (2019) as relation extraction model. Based on the difference of data parts and tagging strategies, we conduct following models for comparison.

SUPERVISED: A supervised baseline system for our relation extraction task (Soares et al., 2019). This system is trained only on human-annotated data. We also use this system as the teacher model to tag unlabeled data for the following approaches.

SELF-TRAINING: A common-used self-training method (Du et al., 2021), which combines human-annotated and confident data. We re-produce this self-training framework with the same relation extraction model of SUPERVISED system.

HARD-LABEL: A direct comparison of the system proposed by Lee (2013), which extends the SELF-TRAINING system to involve all auto-annotated data. We obtain the class with the highest prediction probability as the label for all auto-annotated data.

SOFT-LABEL: Another extension of SELF-TRAINING system which utilizes the remaining auto-annotated data by assigning probability distributions comes from teacher model as labels (Xie et al., 2020).

3.4 Overall Evaluation Results

Table 3 shows the overall evaluation results on SemEval and Re-TACRED with low-resource settings. Our observations are:

- System SELF-TRAINING significantly and consistently outperforms SUPERVISED with +2.5 on Micro-F1 (79.9 vs. 77.4) in average. The results indicate that self-training with sampling confident data is effective for relation extraction in low-resource scenarios.
- Systems HARD-LABEL and SOFT-LABEL employ the ambiguous data outperform the SUPERVISED system, but can not achieve improvement over the SELF-TRAINING system, demonstrating that it is challenging to achieve improvement with the ambiguous data.
- Our final system STAD significantly outperforms the SUPERVISED system on both datasets with +3.4 (80.8 vs. 77.4) in average. And it also performs better than the SELF-TRAINING system, demonstrating the the effectiveness and the versatility of the proposed approach.

3.5 Extremely Low-resource Scenario

In addition, we conduct additional experiments on **extremely low-resource scenario** with 15, 10 and 5 instances per relation to investigate the effect of the proposed approach. Experimental results are

System	Data Size			
	20	15	10	5
SUPERVISED	77.3	72.0	60.6	45.1
SELF-TRAINING	79.2	77.1	71.3	47.4
STAD	**80.0**	**78.1**	**72.6**	**53.5**
$\triangle_1$	2.7	6.1	12.0	8.4
$\triangle_2$	0.8	1.0	1.3	6.1

Table 4: Results on the test set of SemEval with low-resource settings in terms of Micro F1 score. $\triangle_1$ and $\triangle_2$ represent the absolute improvement when comparing ours with supervised baseline and self-training baseline, respectively.

System	Data Size			
	20	15	10	5
SUPERVISED	77.5	74.1	62.0	47.0
SELF-TRAINING	80.5	79.3	68.4	50.1
STAD	**81.6**	**81.1**	**72.3**	**60.0**
$\triangle_1$	4.1	7.0	10.3	13.0
$\triangle_2$	1.1	1.8	3.9	9.9

Table 5: Results on the test set of Re-TACRED with low-resource settings in terms of Micro F1 score. $\triangle_1$ and $\triangle_2$ represent the absolute improvement when comparing ours with supervised baseline and self-training baseline, respectively.

shown in Table 4 and Table 5. From the two tables, we can find that:

- The performance of SUPERVISED drops rapidly when the amount of training data decreases.
- SELF-TRAINING system works well on 20, 15 and 10, but it only achieves a slight improvement on 5. This indicates that it becomes hard to learn from confident data when teacher model is weak.
- The results of $\triangle_1$ and $\triangle_2$ show that our STAD system outperforms both SUPERVISED and SELF-TRAINING on all settings, especially for the improvement obtained on data size is 5 which indicating the robustness of our system. We think the reason is that when the teacher model becomes weak, there is more and more ambiguous data in the auto-annotated data.

3.6 Ablation Study

Method	Mirco F1
STAD	81.6
– Partial Labeling	-1.4
– Set-Negative Training	-2.3
– Both	-3.4

Table 6: An ablation study for using partial labeling and set-negative training on the test set of Re-TACRED.

To further analyze the effect of partial labeling and set-negative training on the ambiguous data in our proposed approach, we perform ablation studies on the low-resource Re-TACRED dataset and list the results in Table 6.

Without the partial labeling, we label the ambiguous data in the hard label mode and use the negative training to train the model, the performance of our proposed approach degrades by 1.4 F1 score. Without the set-negative training, we label the ambiguous data in partial label mode but with positive training. The strategy results in a drastic drop in F1 score. Removing both the partial labeling and set-negative training, the proposed system deals the ambiguous data in the same way as in HARD-LABEL system in Table 3 and suffers from performance degradation (78.2-81.6=-3.4).

These results demonstrate that the partial labeling and the set-negative training on the ambiguous data indeed contribute to the final performance.

3.7 Distribution of Auto-Annotated Data

Figure 3 shows the distribution of auto-annotated data under our probability accumulation method on Re-TACRED. From the figure we can find that although the number of confident data (the first bar, 44.5%) is much larger than the others, the sum of ambiguous data (others, 55.5%) is also considerable. This indicates that ignoring such a large number of ambiguous data is not appropriate and how to make full use of them is the key to improving self-training. In addition, we find that there is no hard instances in our experiments due to the large number of relation types and the setting of probability threshold.

3.8 Top-N Evaluation

Our method of modeling ambiguous data under negative training helps the model compress the scope of answers. Intuitively, there should be

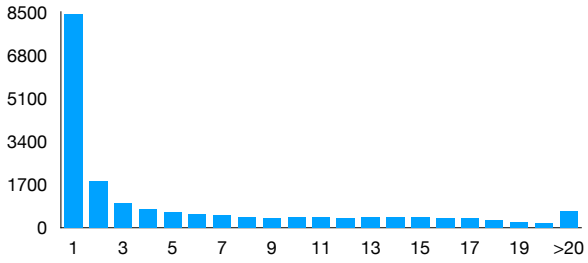

Figure 3: The number of auto-annotated instances according to size of candidate-label sets on Re-TACRED.

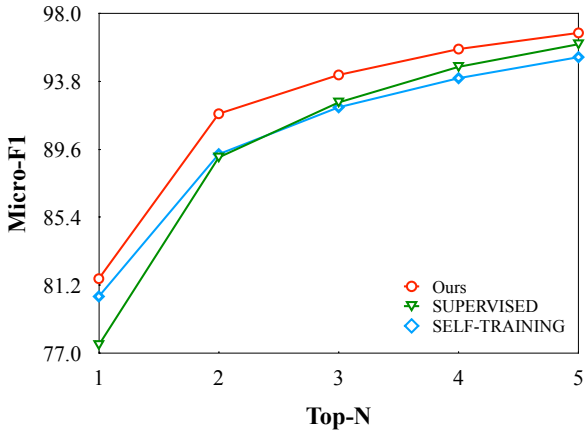

Figure 4: The top-N evaluation on Re-TACRED.

more significant improvements in top-N evaluation. Therefore, we evaluate top-n predictions on Re-TACRED. The results are presented in Figure 4. The figure shows that the SELF-TRAINING method shows a disadvantage in top-N evaluation as its performance gradually degrades when compared to the SUPERVISED baseline. After modeling ambiguous data in a partially annotated and negative training mode, however, our method achieves significant and consistent improvement. In conclusion, our method of modeling ambiguous data can learn the information contained in the ambiguous data.

System	Data Sampling		
	Data 1	Data 2	Data 3
SUPERVISED	$80.3_{\pm 1.6}$	$79.7_{\pm 1.3}$	$79.0_{\pm 1.5}$
SELF-TRAINING	$82.5_{\pm 0.6}$	$84.0_{\pm 1.5}$	$81.4_{\pm 1.7}$
STAD	$\mathbf{84.6}_{\pm 1.5}$	$\mathbf{85.3}_{\pm 1.0}$	$\mathbf{83.9}_{\pm 1.6}$

Table 7: Results on Re-TACRED with different data samples for training and validation sets on the low-resource setting.

3.9 Experiments with Random Data Samples on Re-TACRED

In order to further study the stability of our approach, we conduct extra experiments on Re-TACRED which contains much more data than SemEval. For each experiment, we randomly sample a new split of training and validation sets for the low-resource setting. We report the mean of the micro F1 scores and its standard deviation score of five runs with different random seeds.

3.10 Qualitative Results: Case Study

Table 8 shows examples of ambiguous data in Re-TACRED which are predicated by the teacher model. We present the sentence as well as its relations with two highest predicted probabilities. The teacher model gives relatively high probabilities to the two relations and fails to predicate the ground-truth relation. For example, the teacher model can not distinguish "per:city_of_birth" from "per:country_of_birth" for the first sentence, and the conventional self-training system does not adopt this instance to the training set. But in our approach, this instance is tagged in a partially labeled mode and is used as an ambiguous instance to train the student model.

4 Related Work

Self-Training. Self-training is one of the most commonly used approaches for exploiting unlabeled data (Scudder, 1965; Yarowsky, 1995; McClosky et al., 2006; Lee, 2013). With the development of neural network models and the growth of demand for labeled data, self-training has become a very active field in research. It is widely used to improve the performance of neural machine translation (Zhang and Zong, 2016; Jiao et al., 2020, 2021), question answering (Sachan and Xing, 2018) and low resource dependency parsing (Rotman and Reichart, 2019). In this work, we apply self-training to exploit unlabeled data for low-resource relation extraction. The main difference is our work focuses on the uncertain instances which are often neglected in the previous studies.

Low-Resource Relation Extraction Recently, low-resource relation extraction (LRE) has attracted much attention due to data scarcity problems. There are two main scenarios of LRE: low-shot RE and low-resource supervised RE. Low-shot RE, including few-shot RE (Han et al., 2018) and

Sentence	Relation	Probability
It 's website , lists [Abu Zubaydah]$_{e1}$'s birthplace as [Riyadh]$_{e2}$, Saudi Arabia	per:country_of_birth per:city_of_birth	0.55 0.40
[Parliament]$_{e2}$ speaker [Ali Larijani]$_{e1}$, Iran 's former chief nuclear negotiator.	per:title per:employee_of	0.79 0.16
[Paul Kim]$_{e1}$ (25) Currently lives in [Saratoga]$_{e2}$, CA	per:city_of_birth per:city_of_residence	0.74 0.20

Table 8: Examples of ambiguous data in Re-TACRED."e1" and "e2" are head entity and tail entity respectively. "Probability" is predicted probability given by the teacher model.

zero-shot RE (Levy et al., 2017), trains a model to identify unseen relation labels in a test set with a few support examples or even none of the examples. The low-resource supervised RE is designed to train a supervised model with small training data (Deng et al., 2022). In this work, we also focus on the low-resource supervised RE.

To solve the low-resource problem, one direction is to build a robust model which can make full use of small data and existing knowledge. For example, integrating semantic relation labels (Dong et al., 2021) and introducing knowledge graph (Zhang et al., 2019). Another direction is to enlarge annotated data automatically. Data augmentation is a widely used technology to enrich the data (Xu et al., 2016; Yu et al., 2020). Self-training is also a conventional way to obtain automatically annotated data (Rosenberg et al., 2005; Hu et al., 2021a,b).

Partial Label Learning. In this work, the definition of partial label is a candidate set of labels for an ambiguous instance in a multi-class classification task (Cour et al., 2011). This is different from that in sequence labeling tasks (Li et al., 2014) and multi-label multi-class classification tasks (Xie and Huang, 2018). In order to learn from partially labeled instances, various methods have been proposed to deal with the problem (Nguyen and Caruana, 2008; Cour et al., 2011). Recently, with the help of self-training, Feng and An (2019) proposes a self-guided retraining method to learn from partially labeled data. Besides, Yan and Guo (2020) also proposes to recalculate the confidence of labels in a candidate set by taking the current model as a teacher.

5 Conclusion

This paper proposes a novel self-training approach with ambiguous data (STAD) for the low-resource relation extraction, which fully uses the auto-annotated data. According to the probabilities predicted by the teacher model, we classify the auto-annotated data into three sets: confident set, ambiguous set and hard set. During training, we consider the ambiguous set, which is neglected by the previous studies, and adopt the set-negative training method to alleviate the noise problem. Experimental results show that our proposed system consistently outperforms the self-training systems.

Acknowledgements

This work was supported by the National Natural Science Foundation of China (61876115 and Grant No. 61936010), the Priority Academic Program Development of Jiangsu Higher Education Institutions. The corresponding authors are Wenliang Chen and Jiangjiang Zhao. We thank the anonymous reviewers for their constructive comments.

References

Timothee Cour, Ben Sapp, and Ben Taskar. 2011. Learning from partial labels. *JMLR*.

Shumin Deng, Ningyu Zhang, Hui Chen, Feiyu Xiong, Jeff Z Pan, and Huajun Chen. 2022. Knowledge extraction in low-resource scenarios: survey and perspective. *arXiv preprint arXiv:2202.08063*.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. Bert: Pre-training of deep bidirectional transformers for language understanding. pages 4171–4186.

Manqing Dong, Chunguang Pan, and Zhipeng Luo. 2021. Mapre: An effective semantic mapping approach for low-resource relation extraction. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 2694–2704.

Jingfei Du, Édouard Grave, Beliz Gunel, Vishrav Chaudhary, Onur Celebi, Michael Auli, Veselin Stoyanov,

and Alexis Conneau. 2021. Self-training improves pre-training for natural language understanding. In *NAACL*.

Lei Feng and Bo An. 2019. Partial label learning with self-guided retraining. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, pages 3542–3549.

Xu Han, Hao Zhu, Pengfei Yu, Ziyun Wang, Yuan Yao, Zhiyuan Liu, and Maosong Sun. 2018. Fewrel: A large-scale supervised few-shot relation classification dataset with state-of-the-art evaluation. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 4803–4809.

Iris Hendrickx, Su Nam Kim, Zornitsa Kozareva, Preslav Nakov, Diarmuid Ó Séaghdha, Sebastian Padó, Marco Pennacchiotti, Lorenza Romano, and Stan Szpakowicz. 2010. Semeval-2010 task 8: Multi-way classification of semantic relations between pairs of nominals. In *SemEval*.

Xuming Hu, Chenwei Zhang, Fukun Ma, Chenyao Liu, Lijie Wen, and S Yu Philip. 2021a. Semi-supervised relation extraction via incremental meta self-training. In *Findings of the Association for Computational Linguistics: EMNLP 2021*, pages 487–496.

Xuming Hu, Chenwei Zhang, Yawen Yang, Xiaohe Li, Li Lin, Lijie Wen, and S Yu Philip. 2021b. Gradient imitation reinforcement learning for low resource relation extraction. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 2737–2746.

Wenxiang Jiao, Xing Wang, Shilin He, Irwin King, Michael Lyu, and Zhaopeng Tu. 2020. Data rejuvenation: Exploiting inactive training examples for neural machine translation. In *EMNLP*.

Wenxiang Jiao, Xing Wang, Zhaopeng Tu, Shuming Shi, Michael Lyu, and Irwin King. 2021. Self-training sampling with monolingual data uncertainty for neural machine translation. In *ACL-IJCNLP*.

Youngdong Kim, Junho Yim, Juseung Yun, and Junmo Kim. 2019. Nlnl: Negative learning for noisy labels. In *ICCV*.

Dong-Hyun Lee. 2013. Pseudo-label: The simple and efficient semi-supervised learning method for deep neural networks. In *ICML*.

Omer Levy, Minjoon Seo, Eunsol Choi, and Luke Zettlemoyer. 2017. Zero-shot relation extraction via reading comprehension. In *Proceedings of the 21st Conference on Computational Natural Language Learning (CoNLL 2017)*, pages 333–342.

Zhenghua Li, Min Zhang, and Wenliang Chen. 2014. Ambiguity-aware ensemble training for semi-supervised dependency parsing. In *ACL*, pages 457–467.

Fan Luo, Ajay Nagesh, Rebecca Sharp, and Mihai Surdeanu. 2019. Semi-supervised teacher-student architecture for relation extraction. In *Workshop on Structured Prediction*, pages 29–37.

David McClosky, Eugene Charniak, and Mark Johnson. 2006. Effective self-training for parsing. In *HLT-NAACL*.

Mike Mintz, Steven Bills, Rion Snow, and Dan Jurafsky. 2010. Distant supervision for relation extraction without labeled data. In *ACL*.

Nam Nguyen and Rich Caruana. 2008. Classification with partial labels. In *KDD*, pages 551–559.

Longhua Qian, Guodong Zhou, Fang Kong, and Qiaoming Zhu. 2009. Semi-supervised learning for semantic relation classification using stratified sampling strategy. In *EMNLP*.

Chuck Rosenberg, Martial Hebert, and Henry Schneiderman. 2005. Semi-supervised self-training of object detection models.

Guy Rotman and Roi Reichart. 2019. Deep contextualized self-training for low resource dependency parsing. *TACL*.

Mrinmaya Sachan and Eric Xing. 2018. Self-training for jointly learning to ask and answer questions. In *NAACL*.

Henry Scudder. 1965. Probability of error of some adaptive pattern-recognition machines. *IEEE Transactions on Information Theory*.

Livio Baldini Soares, Nicholas FitzGerald, Jeffrey Ling, and Tom Kwiatkowski. 2019. Matching the blanks: Distributional similarity for relation learning. In *ACL*.

George Stoica, Emmanouil Antonios Platanios, and Barnabás Póczos. 2021. Re-tacred: Addressing shortcomings of the tacred dataset. In *AAAI*.

Ming-Kun Xie and Sheng-Jun Huang. 2018. Partial multi-label learning. In *AAAI*.

Qizhe Xie, Minh-Thang Luong, Eduard Hovy, and Quoc V Le. 2020. Self-training with noisy student improves imagenet classification. In *CVPR*.

Yan Xu, Ran Jia, Lili Mou, Ge Li, Yunchuan Chen, Yangyang Lu, and Zhi Jin. 2016. Improved relation classification by deep recurrent neural networks with data augmentation. In *Proceedings of COLING 2016, the 26th International Conference on Computational Linguistics: Technical Papers*, pages 1461–1470.

Yan Yan and Yuhong Guo. 2020. Partial label learning with batch label correction. In *AAAI*, volume 34, pages 6575–6582.

David Yarowsky. 1995. Unsupervised word sense disambiguation rivaling supervised methods. In *ACL*.

Junjie Yu, Tong Zhu, Wenliang Chen, Wei Zhang, and Min Zhang. 2020. Improving relation extraction with relational paraphrase sentences. In *COLING*, pages 1687–1698, Barcelona, Spain (Online). International Committee on Computational Linguistics.

Jiajun Zhang and Chengqing Zong. 2016. Exploiting source-side monolingual data in neural machine translation. In *EMNLP*.

Ningyu Zhang, Shumin Deng, Zhanlin Sun, Guanying Wang, Xi Chen, Wei Zhang, and Huajun Chen. 2019. Long-tail relation extraction via knowledge graph embeddings and graph convolution networks. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 3016–3025.

Yuhao Zhang, Victor Zhong, Danqi Chen, Gabor Angeli, and Christopher D Manning. 2017. Position-aware attention and supervised data improve slot filling. In *EMNLP*.

Guodong Zhou, Junhui Li, Longhua Qian, and Qiaoming Zhu. 2008. Semi-supervised learning for relation extraction. In *Proceedings of the Third International Joint Conference on Natural Language Processing: Volume-I*.

Guodong Zhou, Jian Su, Jie Zhang, and Min Zhang. 2005. Exploring various knowledge in relation extraction. In *Proceedings of the 43rd annual meeting of the association for computational linguistics*, pages 427–434.

Flat Multi-modal Interaction Transformer for Named Entity Recognition

Junyu Lu[♣♠] **Dixiang Zhang**[♣] **Jiaxing Zhang**[♠] **Pingjian Zhang**[♣*]
[♣]South China University of Technology
[♠]International Digital Economy Academy (IDEA)
lujunyu@idea.edu.cn
sedxzhang@mail.scut.edu.cn
pjzhang@scut.edu.cn

Abstract

Multi-modal named entity recognition (MNER) aims at identifying entity spans and recognizing their categories in social media posts with the aid of images. However, in dominant MNER approaches, the interaction of different modalities is usually carried out through the alternation of self-attention and cross-attention or over-reliance on the gating machine, which results in imprecise and biased correspondence between fine-grained semantic units of text and image. To address this issue, we propose a Flat Multi-modal Interaction Transformer (FMIT) for MNER. Specifically, we first utilize noun phrases in sentences and general domain words to obtain visual cues. Then, we transform the fine-grained semantic representation of the vision and text into a unified lattice structure and design a novel relative position encoding to match different modalities in Transformer. Meanwhile, we propose to leverage entity boundary detection as an auxiliary task to alleviate visual bias. Experiments show that our methods achieve the new state-of-the-art performance on two benchmark datasets.

1 Introduction

Named entity recognition (NER) is a fundamental task in the field of information extraction, which involves determining entity boundaries from free text and classifying them into pre-defined categories, such as person (PER), location (LOC), and organization (ORG) (Zhao et al., 2021b). Along with the rapid development of social media, multi-modal deep learning is widely applied in the structured extraction from massive multimedia news and web product information (Zhang et al., 2020; Ju et al., 2020). As an important research branch of NER, multi-modal named entity recognition (MNER) significantly extends the text-based NER by taking the {Sentence, Image} pair as inputs (Lu et al., 2018; Kruengkrai et al., 2020; Dosovitskiy et al., 2020; Lu and Zhang, 2022). Since the visual context associated with text content has been confirmed to help resolve the recognition of ambiguous multi-sense words and out-of-vocabulary words, MNER plays an important role in extracting entities from user-generated content on social media platforms such as Twitter (Li et al., 2015).

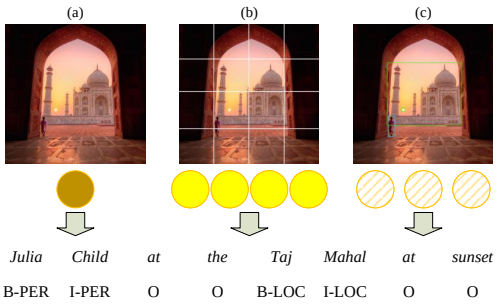

Figure 1: An example for multi-modal named entity recognition with different visual cues: (a) the whole image, (b) averagely segmented visual feature and (c) targeted visual objects.

It has been the core issue in MNER to fully exploit the effective visual information and suppress the interference information, which directly affects the model performance. To this end, there are three lines of methods to integrate visual information into NER. (1) The first line is to encode the whole image into a global feature vector (Figure 1(a)) for augmenting each word representation (Moon et al., 2018). (2) The second line is to divide the feature map extracted from the whole image into multiple regions averagely (Figure 1(b)), which is the most dominant method currently. The method guides the word to learn a vision-aware representation through co-attention and gating mechanism (Lu et al., 2018; Zhang et al., 2018), or uses a Transformer framework based on the combination of self-modal and cross-modal attention to interact textual and visual information (Yu et al., 2020; Sun et al., 2021). (3) The last line is to use noun phrases to detect the image bounding boxes (Figure 1(c)) and fuse fine-grained words and visual objects by graph neural

*Corresponding author

Proceedings of the 29th International Conference on Computational Linguistics, pages 2055–2064
October 12–17, 2022.

networks (GNN) (Zhang et al., 2021).

Despite the success, the above works may not precisely exploit the fine-grained semantic correspondence between semantic units in an input sentence-image pair. Specifically, as shown in Figure 1, we believe that the crux of the issue lies in two aspects. (1) The global clue provided by image (a) and regional feature maps provided by images (b) are both implicit and vague, which are difficult to fit into fine-grained words. Previous practice tends to map the visual and textual representations into different spaces and then fuse them adaptively. However, the indirectness of information interaction through cross-modal attention and gating mechanism will lead to asymmetry in information acquisition (Sun et al., 2021; Yu et al., 2020). (2) Specific visual objects derived from noun phrases are overly targeted, which makes some non-entities embodied in the images identified as entities incorrectly. Generally speaking, this kind of explicit information is helpful to identify some words as the correct entity type, such as "*the Taj Mahal*". However, the visual objects prominent in the image may easily misidentified as entities, such as "*sunset*".

To handle the aforementioned issues, we propose a novel Flat Multi-model Interaction Transformer for MNER. The key insight comes from the lattice structure in Chinese NER (Zhang and Yang, 2018; Li et al., 2020), where word sequence is used as additional information to enhance the character representation. To fully exploit the available visual information, we use a combination of two visual objects, extracted from the whole image and derived from the noun phrases. We first represent the input sentence and image with a unified flat lattice structure consisting of fine-grained semantic units. Each unit corresponds to a word or visual object and its position. Meanwhile, inspired by the strategies of position representation, we design an ingenious position encoding for our flat lattice structure (Shaw et al., 2018; Dai et al., 2019). In detail, we assign two positional indices for a unit: head position and tail position, by which we can correspond the visual object to the associated words span. Based on the flat lattice structure, we then resort to the fully-connected self-attention structure and long-distance dependencies modeling capability in Transformer (Vaswani et al., 2017) to build bridges in the interaction between self-modal and cross-modal units. Finally, we utilize the CRF decoder to obtain the predicted labels. To largely eliminate the bias of visual context, we further introduce the entity boundary detection (EBD) as an auxiliary task.

We conduct extensive experiments on Twitter 2015 and Twitter 2017 benchmark datasets. The state-of-the-art performance and efficiency demonstrate the effectiveness of our methods.

2 Related Work

Multi-modal NER. As an important role in many downstream NLP tasks, including information retrieval (Chen et al., 2015), relation extraction (Miwa and Bansal, 2016) and question answering system (Diefenbach et al., 2018), the text-based NER task has attracted much attention in the journalistic and social fields. Deep learning approaches such as CNN, LSTM, attention mechanism and pre-trained models have achieved significant success in NER, by which we can effectively uncover and combine the character, word and sentence information in text sequence (Ma and Hovy, 2016; Akbik et al., 2019; Luo et al., 2020). Influenced by the extensive applications of multi-modal learning in neural machine translation, visual question answering and emotion detection (Zhang et al., 2020; Yin et al., 2020; Gao et al., 2019), many researches have focused on constructing multi-modal NER datasets and exploring the methods to guide entity recognition using images. The main idea of these early methods is encoding the text through LSTM and the image through pre-trained CNN, then implicitly interacting the information of two modalities (Lu et al., 2018; Moon et al., 2018; Zhang et al., 2018). Recently, (Yu et al., 2020) leverage BERT to model text sequence and creatively design a multi-modal interaction module based on Transformer and gating mechanism to perform self-modal and cross-modal information interaction alternately. (Zhang et al., 2021) further represent the input sentence-image pair as a unified graph to capture the various semantic relationships and introduce an extended GNN to conduct graph encoding via multi-modal semantic interactions. Different from above studies, our approach aims at representing the fine-grained semantic units of the text and image as a unified flat lattice structure. We further design a novel relative position encoding strategy to directly capture the interaction between different modalities in Transformer.

Lattice Structure. Since word information is potentially useful for character-based Chinese NER task, the lattice structure is proposed for injecting the word information into the associated characters. Specifically, (Zhang and Yang, 2018) first proposes Lattice-LSTM to explicitly exploit word boundary information, in which the matched lexical words are integrated into characters via a directed acyclic graph. Later, to overcome the limitations of the lattice structure so that it can be flexibly exploited, (Sui et al., 2019) converts the lattice into graph and designs a collaborative graph network for encoding. (Zhao et al., 2021a) proposes a dynamic cross- and self-lattice attention network to model dense interactions over word-character pairs.

3 Method

Figure 2 illustrates the overall architecture of our FMIT, which contains three main components: (1) Unified flat lattice structure for representing the input sentence-image pairs. (2) Transformer Encoder with relative position encoding method for interacting multi-modal information. (3) Training with entity boundary detection as an auxiliary task.

Task Formulation. Given a sentence S and its associated image O as input, the goal of MNER is to extract a set of entities from S and classify each extracted entity into one of the pre-defined categories. As with most existing work in MNER, we formulate the task as a sequence labelling problem. Let $S = (w_1, w_2, ..., w_n)$ denote a sequence of input words, where w_i with $i = 1, 2, ...n$ denotes the ith word in the sentence and n represents the length of the sentence, and $Y = (y_1, y_2, ..., y_n)$ be the corresponding entity labels for all words, where $y_1 \in \mathcal{Y}$ and $\mathcal{Y}$ is the pre-defined label set with standard BIO schema (Sang and Veenstra, 1999). We also use $O = (o_1, o_2, ..., o_m)$ to denote a set of input visual objects of number m.

3.1 Unified Flat Lattice Structure

In this section, we take the sentence and image shown in Figure 1 as an example to describe how to extract features from them and represent them with a flat lattice structure.

Word Representations. Due to the capability of providing different representations for the same word in different contexts, we utilize pre-trained language model BERT (Devlin et al., 2019) as our sentence encoder. Following BERT, the input sentence is preprocessed by inserting the special token [CLS] and [SEP] at the beginning and the end positions, respectively. S is then fed to BERT encoder to obtain the vectorized representation $H_x = (x_1, x_2, ...x_n)$, where $x_i \in \mathbb{R}^{d_w}$ is the generated contextualized vector for w_i.

Visual Representations. To capture the visual objects in O, except for employing the whole image, we also need to derive additional visual objects from the text. Similar to (Yin et al., 2020), we use the constituency parsing tool in the Stanford parser to identity all noun phrases in the input sentence, and then apply a visual grounding toolkit (Yang et al., 2019) to detect bounding boxes (visual objects) for each noun phrase. Since it is difficult to use noun phrases merely to completely detect all potential visual objects in the image, according to the property of NER, we further introduce four general words of the pre-defined categories (i.e., miscellaneous, person, location and organization) to discover more relevant objects.

To extract meaningful feature representations from images, we leverage a pre-trained 152-layer ResNet (He et al., 2016) as a feature detector. We feed each visual object in O to the ResNet and take the last hidden layer output as vectorized representation $H_v = (v_1, v_2, ..., v_m)$, where $v_i \in \mathbb{R}^{d_v}$ is the generated visual representation for o_i.

Flat Lattice Construction. The flat lattice structure aims to integrate the intra-modal and inter-modal information in a unified space, and enhances information coupling through a unique positioning scheme. Before representing the words and visual objects in a uniform lattice cell, we introduce two non-linear transformations with ReLU activation function to project different representations onto the same dimension:

$$x_i^c = W_0(\text{ReLU}(W_1 x_i + b_1)) + b_0 \quad (1)$$

$$v_i^c = W_0(\text{ReLU}(W_2 v_i + b_2)) + b_0, \quad (2)$$

where $W_1 \in \mathbb{R}^{d \times d_w}$, $W_2 \in \mathbb{R}^{d \times d_v}$, $W_0 \in \mathbb{R}^{d \times d}$ are weight matrices, and b_1, b_2, b_0 are scaler bias. d is the dimension of unified representations of two modalities in the flat lattice.

To convert two independent sets of modalities to a flat lattice structure, we concatenate the aligned word representation and visual representation, and then flatten them into a unified sequence $E = ([CLS], x_1^c, ..., x_n^c, [SEP], v_1^c, ..., v_m^c)$. As shown in Figure 2, the flat lattice can be defined as a set of cells, and a cell corresponds to a fine-grained

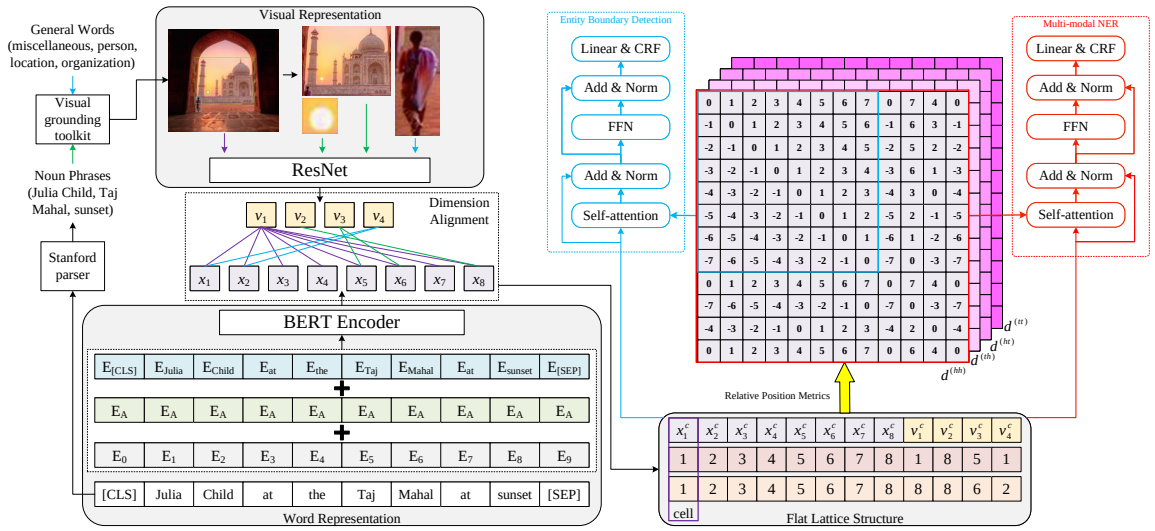

Figure 2: The overall architecture of our Flat Multi-modal Interaction Transformer (FMIT). On the right part, the blue-frame demonstrates the auxiliary task of entity boundary detection, and the red-frame demonstrates the main task of MNER.

semantic unit, a head and a tail. Specifically, for the word, its head and tail are equal, both indexes of the absolute position in E. For the visual object, when it is derived from a noun phrase, its head and tail are indexes of the first and last words of the corresponding noun phrase; in particular, when it is the whole image or derived from general words, we denote its head and tail as indexes of the first and last words of the sequence.

3.2 Flat Multi-modal Interaction Transformer

Relative Position Encoding for Flat Lattice Structure. The flat lattice structure consists of cells with different modalities and different position intervals in the visual modality. As illustrated on the red-frame of Figure 2, to leverage the Transformer framework to encode interactions among cells, we design a relative position encoding for the cells. Specifically, for two cells c_i and c_j in the lattice, we consider two kinds of relations between them: intra-modal and inter-modal. Instead of directly encoding input with absolute position as vanilla Transformer, we calculate a dense vector to represent relative position by continuous transformation of the head and tail information. In this way, we can not only capture the distance between arbitrary cells, but also model the relationship between different modalities. Let $head[i]$ and $tail[i]$ denote the head and tail position of cell c_i. We use four kinds of relative distance to indicate the position information between c_i and c_j. They can be calculated as:

$$d_{ij}^{(hh)} = head[i] - head[j] \quad (3)$$

$$d_{ij}^{(ht)} = head[i] - tail[j] \quad (4)$$

$$d_{ij}^{(th)} = tail[i] - head[j] \quad (5)$$

$$d_{ij}^{(tt)} = tail[i] - tail[j], \quad (6)$$

where $d_{ij}^{(ht)}$ denotes the distance between the head of c_i and tail of c_j, and $d_{ij}^{(hh)}$, $d_{ij}^{(th)}$, $d_{ij}^{(tt)}$ have similar meanings. To obtain the position encoding P_{pos} from distance value, we adopt sine and cosine functions of different frequencies as in (Vaswani et al., 2017):

$$P_{pos}^{2k} = \sin(pos/10000^{2k/d}) \quad (7)$$

$$P_{pos}^{2k+1} = \cos(pos/10000^{2k/d}), \quad (8)$$

where pos is $d_{ij}^{(hh)}$, $d_{ij}^{(th)}$, $d_{ij}^{(ht)}$ or $d_{ij}^{(tt)}$, and k is the index of dimension of position encoding. The Transformer has the same dimension d as flat lattice embeddings. Then, we concatenate the four distance position encodings and feed them into a non-linear transformation layer to get the final relative position encoding of cells:

$$R_{ij} = \mathrm{ReLU}(W_r(P_{d_{ij}^{(hh)}} \oplus P_{d_{ij}^{(ht)}} \oplus P_{d_{ij}^{(th)}} \oplus P_{d_{ij}^{(tt)}})), \quad (9)$$

where $W_r \in \mathbb{R}^{d \times 4d}$ is a learnable parameter, and $\oplus$ denotes the concatenation operation.

As mentioned in (Shaw et al., 2018), we think that commutativity of the vector inner dot will cause the loss of directionality in Transformer. Therefore, we further use a variant of self-attention (Dai et al., 2019) to leverage the relative position

encoding of different cells, and the attention score between the query q_i and key vector k_j of two semantic units can be calculated as following:

$$\begin{aligned} A_{i,j} =& E_{c_i}^T W_q^T W_{k,E} E_{c_j} + E_{c_i}^T W_q^T W_{k,R} R_{ij} \\ &+ u^T W_{k,E} E_{c_j} + v^T W_{k,R} R_{ij}, \end{aligned} \tag{10}$$

where E_{c_i} represents the flat lattice embedding of i-th cell from x_i^c and v_i^c, or the output of last Transformer layer. We collect the attention values and embeddings of all indexes, and denote them as A^* and E_c^*. Then, we perform self-attention over the sequence by $h = 8$ parallel heads individually and $d_{head} = d/h$ is the dimension of each head. We concatenate the results and transform them into the original dimension by a linear projection. The output of the Transformer is calculated as following:

$$Att_i = softmax(\frac{A^*}{\sqrt{d_{head}}})[E_c^* W_v]^T \tag{11}$$

$$MH\text{-}ATT = W_t[Att_1; ...; Att_z], \tag{12}$$

where W_q, $W_{k,R}$, $W_{k,E}$, $W_v \in \mathbb{R}^{d \times d_{head}}$ and $u, v \in \mathbb{R}^{d_{head}}$ are learnable parameters of each head, and they keep individual from different heads. $W_t \in \mathbb{R}^{d \times d}$ denotes the weight matrices for the multi-head attention. For simplicity, we omit the subsequent layers, which are the same as vanilla Transformer.

CRF Decoding. To increase model capacity and interaction frequency, we stack l Transformer layers to form a cascaded architecture. Finally, we only take the word presentation of the encoding output as $H_W \in \mathbb{R}^{n \times d}$, which is sent to the decoding layer for sequence labelling. Considering the dependency between successive labels, we model H_W jointly using a standard CRF layer. Let $Y^{'}$ denotes the set of all possible label sequences for input sentence X, the probability of the label sequence Y can be calculated as:

$$p(Y|S, O) = \frac{\prod_{n=1}^{N} \psi_n(y_{n-1}, y_n, H_W)}{\sum_{y' \in Y'} \prod_{n=1}^{N} \psi_n(y'_{n-1}, y'_n, H_W)}, \tag{13}$$

where $\psi_n(y_{n-1}, y_n, H_W) = \exp(W_{crf} H_W + b_{crf})$ is the scoring function, and W_{crf} and b_{crf} are the weight vector and bias.

3.3 Training with Entity Boundary Detection

Since the pre-trained ResNet is intended for the image classification task, its high-level representation may overemphasize the visual objects that are prominent in the image and misidentify them as named entities. To alleviate the bias, we propose to leverage a flat text-based Transformer for entity boundary detection based on the properties of positioning scheme in the flat lattice structure. The EBD task aims to detect the position of the head and tail of entities in the input sentence, which can eliminate the types guidance from visual objects and enhance the perception of boundary words. As illustrated on the blue-frame of Figure 2, the flat text-based Transformer is the same structure as FMIT, but only takes the words representation as input.

We remove the type information and decompose the subsequence $Z = (z_1, z_2, ..., z_n)$ from the labelling sequence Y, where $z_i \in \{B, E, O\}$ indicates whether the i-th position is the head, tail or neither of an entity. We employ the flat text-based Transformer to obtain its specific hidden representations as $T_W \in \mathbb{R}^{n \times d}$, followed by feeding it to another CRF layer to predict the probability of the label sequence Z given S as Eqn. (13):

$$p(Z|S) = \frac{\prod_{n=1}^{N} \psi_n(z_{n-1}, z_n, T_W)}{\sum_{z' \in Z'} \prod_{n=1}^{N} \psi_n(z'_{n-1}, z'_n, T_W)}, \tag{14}$$

In training phase, we linearly combine the loss function of the main MNER task and auxiliary EBD task, making the final training objective function by minimizing negative log-likelihood estimation as follows:

$$\mathcal{L} = -\sum(\log p(Y|S, O) + \lambda \log p(Z|S)), \tag{15}$$

Table 1: The statistics summary of two Twitter datasets.

Entity Type	Twitter-2015			Twitter-2017		
	Train	Dev	Test	Train	Dev	Test
Person	2217	552	1816	2943	626	621
Location	2091	522	1697	731	173	178
Organization	928	247	839	1674	375	395
Miscellaneous	940	225	726	701	150	157
Total	6176	1546	5078	6049	1324	1351
Num of Tweets	4000	1000	3257	3373	723	723

4 Experiments

We conduct experiments on two MNER datasets, comparing our Flat Multi-modal Interaction Transformer (FMIT) approach with a number of uni-modal and multi-modal approaches.

4.1 Datasets

We take two publicly benchmark Twitter datasets (Twitter-2015 and Twitter-2017) for MNER, which are provided by (Zhang et al., 2018) and (Lu et al., 2018), respectively. Each sample consists of a {Sentence, Image} pair. Since some samples lack image modality, we replace the missing images with a uniform empty image. Table1 shows the number of entities for each type and the size of train/dev/test data split.

Table 2: Performance of different competitive text-based and multi-modal approaches on two Twitter datasets.

Modality	Approaches	Twitter-2015							Twitter-2017						
		Single Type (F1)				Overall			Single Type (F1)				Overall		
		PER	LOC	ORG	MISC	P	R	F1	PER	LOC	ORG	MISC	P	R	F1
Text	CNN-BiLSTM-CRF	80.86	75.39	47.77	32.61	66.24	68.09	67.15	87.99	77.44	74.02	60.82	80.00	78.76	79.37
	HBiLSTM-CRF	82.34	76.83	51.59	32.52	70.32	68.05	69.17	87.91	78.57	76.67	59.32	82.69	78.16	80.37
	BERT	84.72	79.91	58.26	38.81	68.30	74.61	71.32	90.88	84.00	79.25	61.63	82.19	83.72	82.95
	BERT-CRF	84.74	80.51	60.27	37.29	69.22	74.59	71.81	90.25	83.05	81.13	62.21	83.32	83.57	83.44
Text + Image	VG-ATT	82.66	77.21	55.06	35.25	73.96	67.90	70.80	89.34	78.53	79.12	62.21	83.41	80.38	81.87
	Ada-Co-ATT	81.98	78.95	53.07	34.02	72.75	68.74	70.69	89.63	77.46	79.24	62.77	84.16	80.24	82.15
	UMT	85.24	81.58	63.03	39.45	71.67	75.23	73.41	91.56	84.73	82.24	70.10	85.28	85.34	85.31
	UMGF	84.26	83.17	62.45	42.42	74.49	75.21	74.85	91.92	85.22	83.13	69.83	86.54	84.50	85.51
	FMIT ($l = 1$)	84.83	83.19	62.64	41.13	74.18	75.03	74.60	91.75	85.06	82.38	69.84	85.55	85.29	85.42
	FMIT ($l = 3$)	**86.77**	83.93	**64.88**	42.97	75.11	**77.43**	**76.25**	**93.14**	**86.52**	83.93	70.90	**87.51**	86.08	**86.79**
	FMIT ($l = 6$)	86.45	**84.19**	64.35	**43.68**	**76.28**	75.67	75.97	93.04	85.94	**84.56**	**71.20**	86.80	**86.26**	86.53
	FMIT ($l = 12$)	85.79	83.91	62.87	41.55	74.92	75.63	75.27	92.61	86.03	83.34	70.78	86.32	85.50	85.91

Table 3: The relative inference-time speed and parameters of different models in the information interaction phrase.

Approaches	Speedup	Parameters
UMT	1.93×	403M
UMGF	1×	231M
FMIT ($l = 1$)	**27.73**×	**12M**
FMIT ($l = 3$)	10.17×	35M
FMIT ($l = 6$)	6.12×	69M

4.2 Baselines

For a comprehensive comparison, we mainly compare two groups of baselines with our approach.

The first group is the representative text-based approaches for NER: (1) **CNN-BiLSTM-CRF** (Ma and Hovy, 2016) and **HBiLSTM-CRF** (Lample et al., 2016), leverage both character-level information and BiLSTM-based word-level information. (2) **BERT** (Devlin et al., 2019) and **BERT-CRF**, a pre-trained multi-layer bidirectional Transformer encoder.

The second group is several competitive multi-modal approaches for MNER: (3) **VG-ATT** (Lu et al., 2018), based on **HBiLSTM-CRF** with the visual context, utilizes a visual attention model and a gate mechanism to mine implicit the word-aware visual information. (4) **Ada-Co-ATT** (Zhang et al., 2018), a multi-modal approach based on **CNN-BiLSTM-CRF**, designs an adaptive co-attention network to fuse word-guided visual representations and image-guided textual representations by a filtration gate. (5) **UMT** (Yu et al., 2020) empowers Transformer with a multi-modal interaction module to capture the inter-modality dynamics and incorporates the auxiliary entity span detection module. (6) **UMGF** (Zhang et al., 2021), the existing state-of-the-art approach for MNER, uses a unified multi-modal graph to capture the semantic relationships between the words and visual objects and stack multiple fusion layers to perform semantic interactions to learn node representations.

4.3 Experiment Results

We mainly adopt standard Precision (P), Recall (R) and F1-score (F1) to evaluate the overall performance on two Twitter MNER datasets and report the metric F1 for each single type. To demonstrate the effectiveness and robustness of **FMIT**, we conduct extensive experiments from self-domain and cross-domain scenarios.

Self-domain Scenario. Table 2 shows the performance comparison of our FMIT approach with different competitive text-based and multi-modal approaches in a self-domain scenario for MNER.

(1) Compared with the text-based approaches, the multi-modal approaches can generally achieve better performance than their corresponding unimodal baselines, which demonstrates that incorporating the visual information is motivating for NER in social media. For example, in the overall F1 on both datasets, **VG-ATT** outperforms **HBiLSTM-CRF** by 1.63% and 1.50%, respectively. Moreover, recent multi-modal approaches **UMT** and **UMGF** show significant performance improvements when replacing the sentence encoder with BERT and using the Transformer framework to interact textual and visual information. It further demonstrates that the self-attention in Transformer is more beneficial for feature fusion and filtering.

(2) Compared with **UMT** and **UMGF**, which utilize Transformer to model intra-modal and inter-modal information interactions and dynamically control the contribution of visual features through

Table 4: Performance comparison of **FMIT** and two existing state-of-the-art multi-modal approaches in cross-domain scenarios for generalization analysis.

Approaches	Twitter-2017→Twitter-2015							Twitter-2015→Twitter-2017						
	Single Type (F1)				Overall			Single Type (F1)				Overall		
	PER	LOC	ORG	MISC	P	R	F1	PER	LOC	ORG	MISC	P	R	F1
UMT	80.34	71.30	47.97	20.13	64.67	63.59	64.13	81.24	67.89	39.52	31.87	67.80	55.23	60.87
UMGF	79.62	71.94	49.48	20.24	**67.00**	62.81	66.21	81.83	**72.25**	41.20	32.00	69.88	56.92	62.74
FMIT($l = 3$)	**82.05**	**72.33**	**50.82**	**21.28**	66.72	**69.73**	**68.19**	**83.51**	71.96	**42.93**	**33.46**	**70.65**	**59.22**	**64.43**

Table 5: Ablation study of our FMIT.

Approaches	Twitter-2015							Twitter-2017						
	Single Type (F1)				Overall			Single Type (F1)				Overall		
	PER	LOC	ORG	MISC	P	R	F1	PER	LOC	ORG	MISC	P	R	F1
FMIT ($l = 3$)	**86.77**	**83.93**	**64.88**	**42.97**	**75.11**	77.43	**76.25**	**93.14**	**86.52**	**83.93**	**70.90**	**87.51**	86.08	**86.79**
w/o Obj	85.55	80.73	63.37	38.61	73.44	74.25	73.84	92.16	85.23	81.57	68.97	85.36	84.69	85.02
w/o Rel	84.13	79.95	62.64	38.86	72.87	73.38	73.12	91.06	85.36	81.24	67.83	84.29	84.76	84.52
w/o EBD	86.21	83.26	64.05	42.21	73.38	**77.95**	75.60	92.73	86.14	82.55	69.47	85.55	**86.67**	86.11

a gating mechanism, Our **FMIT** makes radical promotion in model structure and representations of different modalities. In the overall F1 on both datasets, our best model achieves state-of-the-art performance by obtaining 76.25% and 86.79% results, outperforming **UMT** by 2.84% and 1.48%, and outperforming **UMGF** by 1.40% and 1.28%, respectively.

(3) We compare the impact of the number of Transformer layers. It can be observed that when $l = 1$, **FMIT** achieves performance comparable to **UMT** and **UMGF**, both of which use the 12-layers Transformer framework. When $l = 3$ or $l = 6$, we reach state-of-the-art F1-score in all single types and overall F1-score, precision and recall metrics on both datasets. As shown in Table 3, our approach can achieve better performance with fewer parameters and higher efficiency. With only 12M parameters, the 1-layer **FMIT** is 14.37 times and 27.73 times faster than **UMT** and **UMGF** in information interaction phases, respectively. It demonstrates that flat lattice structure can more fully and directly establish interactions in both intra-modal and inter-modal simultaneously, making it possible to incorporate important visual information into entity recognition with only a few Transformer layers. Meanwhile, we speculate that the reason for the performance declines at 12-layer **FMIT** is the redundancy of unnecessary information.

Cross-domain Scenario. Due to the obvious differences in type distribution and data characteristics between the two Twitter datasets, we compare our **FMIT** approach and two existing state-of-the-art multi-modal approaches in cross-domain scenarios for generalization analysis. Twitter-2017→Twitter-2015 indicates that the model trained on Twitter-2017 is used to test Twitter-2015, and Twitter-2015→Twitter-2017 has similar meaning. As shown in Table 4, our approach outperforms **UMT** and **UMGF** by a large margin in most metrics. The potential reason for the excellent generalization may be that with the tight information coupling structure enables **FMIT** to learn the underlying features better.

4.4 Ablation Study

To investigate the influence of different factors of our proposed approach, we perform comparison between the 3-layer **FMIT** and its ablation approaches, concerning the entity boundary detection task and several critical components of the model. The results are reported in Table 5.

w/o Obj. Firstly, we replace the targeted visual object guidance with 7×7 average-segmented visual blocks, which can be obtained by feeding the whole image to ResNet (He et al., 2016) and taking the output of the last convolution layer. This approach completely ignores the correspondence of fine-grained units between different modalities, bringing in significant performance degradation.

w/o Rel. Secondly, we remove the relative position encoding for flat lattice structure and the positioning scheme of each cell. In this case, we only use self-attention in vanilla Transformer to conduct intra-modal and inter-modal fusions. We find that the overall F1 on both datasets decreases substantially by 3.13% and 2.27%, respectively, which indicates a critical role for coupling interactions between different modalities through position strategy.

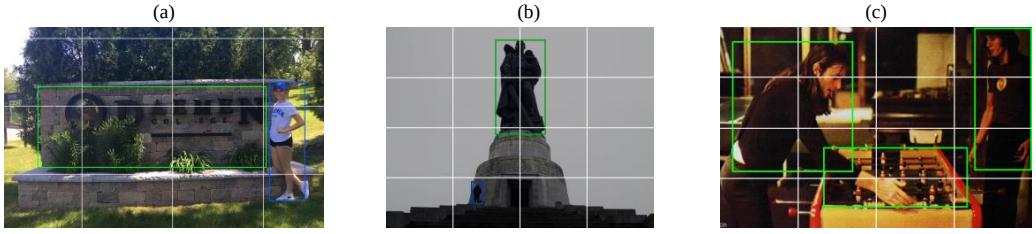

	(a)	(b)	(c)
	I've committed to play soccer at **[Daemen College ORG]**[1] in **[New York LOC]**[2]	**[The Soviet War Memorial MISC]**[1] at **[Treptower Park LOC]**[2], **[Berlin LOC]**[3]	**[David Gilmour PER]**[1] and **[Roger Waters PER]**[2] playing **[table football]**[3]
BERT-CRF	1-None ×, 2-LOC ✓	1-None ×, 2-LOC ✓, 3-LOC ✓	1-PER ✓, 2-PER ✓, 3-None ✓(0.495)
UMT	1-ORG ✓(0.362), 2-LOC ✓	1-PER ×, 2-LOC ✓, 3-LOC ✓	1-PER ✓, 2-None ×, 3-None ✓(0.386)
UMGF	1-ORG ✓(0.575), 2-LOC ✓	1-MISC ✓(0.623), 2-LOC ✓, 3-LOC ✓	1-PER ✓, 2-PER ✓, 3-MISC ×
FMIT	1-ORG ✓(0.867), 2-LOC ✓	1-MISC ✓(0.883), 2-LOC ✓, 3-LOC ✓	1-PER ✓, 2-PER ✓, 3-None ✓(0.713)

Figure 3: The first row shows several representative samples together with their manually labeled entities in the test set of two Twitter datasets, and the bottom four rows show predicted entities of different approaches on these test samples. The values in parentheses represent the confidence of the predicted label.

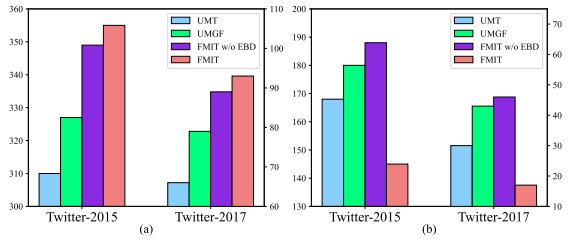

Figure 4: Predicted results statistics: (a) The number of entities (shown in y-axis) that are incorrectly predicted by **BERT-CRF**, but get corrected by each multi-modal approach; (b) The number of entities (shown in y-axis) that are correctly predicted by **BERT-CRF**, but wrongly predicted by each multi-modal approach.

w/o EBD. Discarding the entity boundary detection task and only retaining the main MNER task will lead to significant performance degradation in overall precision, while a slight increase in overall recall. The result is consistent with our hypothesis that the guidance of visual objects drives the corresponding words to be misjudged as entities, while the EBD auxiliary task can balance the play of visual objects and text itself.

4.5 Further Analysis

Case Study. To better understand the effective of our approach in incorporating visual information into the MNER task, we select a representative set of test samples to compare the prediction results of the 3-layer **FMIT** and other approaches.

First, from Figure 3(a), we can observe that the **BERT-CRF** fails to identify *Daemen College* due to the lack of guidance from visual context such as the plaque, while all the multi-modal approaches can accurately determine the entities by referring to specific visual regions.

Second, we can see from Figure 3(b) that **UMT** gives a wrong identification of the entity *The Soviet War Memorial*, probably because the segmented visual feature is fragmented, bringing in interference to type classification. On the contrary, **UMGF** and **FMIT** can accurately classify the entities into corresponding types with the guidance of targeted visual objects.

Third, as shown in Figure 3(c), **UMGF** erroneously identifies *table football* as an entity of MISC, which indicates that over-reliance on visual information will lead to emphasis bias. Therefore, **FMIT** corrects this bias by balancing the importance of text and vision with EBD task.

Finally, we find that compared with other approaches, **FMIT** can obtain higher confidence in predicted results. For example, for entity *Daemen College* in Figure 3(a), **FMIT** achieves a label confident of 0.867, substantially outperforming **UMGF**(0.575) and **UMT**(0.362). It indicates that our relative position encoding strategy and flat lattice structure can extract important information and couple different modalities more directly.

Statistic Study. To better appreciate the importance of the EBD auxiliary task, we count the number of entities that are correctly/wrongly predicted by **BERT-CRF**, but wrongly/correctly predicted by each multi-modal approach.

As shown in Figure 4, compared with other multi-modal methods, our **FMIT** can correctly identify more entities due to the powerful image-ware word representations. Moreover, it is clear that **FMIT** introduces fewer wrong entities with the help of EBD auxiliary task. It demonstrates that the well-designed EBD auxiliary task can greatly eliminate the visual bias brought by visual context and perform more efficiently than the span detection module proposed in **UMT**.

5 Conclusion

In this paper, we propose a novel Flat Multi-modal Interaction Transformer for MNER, which exploits flat lattice structure and relative position encoding to directly interact fine-grained semantic units between different modalities. Moreover, we put forward entity boundary detection as an auxiliary task to alleviate visual bias. We conduct extensive experiments on two MNER datasets, and experimental results demonstrate that our approach outperforms other text-based and multi-modal approaches.

References

Alan Akbik, Tanja Bergmann, and Roland Vollgraf. 2019. Pooled contextualized embeddings for named entity recognition. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 724–728.

Yubo Chen, Liheng Xu, Kang Liu, Daojian Zeng, and Jun Zhao. 2015. Event extraction via dynamic multi-pooling convolutional neural networks. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 167–176.

Zihang Dai, Zhilin Yang, Yiming Yang, Jaime G Carbonell, Quoc Le, and Ruslan Salakhutdinov. 2019. Transformer-xl: Attentive language models beyond a fixed-length context. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 2978–2988.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. Bert: Pre-training of deep bidirectional transformers for language understanding. In *Annual Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*.

Dennis Diefenbach, Vanessa Lopez, Kamal Singh, and Pierre Maret. 2018. Core techniques of question answering systems over knowledge bases: a survey. *Knowledge and Information Systems*, 55(3):529–569.

Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, et al. 2020. An image is worth 16x16 words: Transformers for image recognition at scale. In *International Conference on Learning Representations*.

Peng Gao, Zhengkai Jiang, Haoxuan You, Pan Lu, Steven CH Hoi, Xiaogang Wang, and Hongsheng Li. 2019. Dynamic fusion with intra-and inter-modality attention flow for visual question answering. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 6639–6648.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. 2016. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778.

Xincheng Ju, Dong Zhang, Junhui Li, and Guodong Zhou. 2020. Transformer-based label set generation for multi-modal multi-label emotion detection. In *Proceedings of the 28th ACM International Conference on Multimedia*, pages 512–520.

Canasai Kruengkrai, Thien Hai Nguyen, Sharifah Mahani Aljunied, and Lidong Bing. 2020. Improving low-resource named entity recognition using joint sentence and token labeling. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 5898–5905.

Guillaume Lample, Miguel Ballesteros, Sandeep Subramanian, Kazuya Kawakami, and Chris Dyer. 2016. Neural architectures for named entity recognition. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 260–270.

Chenliang Li, Aixin Sun, Jianshu Weng, and Qi He. 2015. Tweet segmentation and its application to named entity recognition. *IEEE Transactions on Knowledge and Data Engineering*, 2(27):558–570.

Xiaonan Li, Hang Yan, Xipeng Qiu, and Xuan-Jing Huang. 2020. Flat: Chinese ner using flat-lattice transformer. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 6836–6842.

Di Lu, Leonardo Neves, Vitor Carvalho, Ning Zhang, and Heng Ji. 2018. Visual attention model for name tagging in multimodal social media. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1990–1999.

Junyu Lu and Pingjian Zhang. 2022. Local context interaction-aware glyph-vectors for chinese sequence tagging. In *ICASSP 2022 - 2022 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 8152–8156.

Ying Luo, Fengshun Xiao, and Hai Zhao. 2020. Hierarchical contextualized representation for named entity recognition. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pages 8441–8448.

Xuezhe Ma and Eduard Hovy. 2016. End-to-end sequence labeling via bi-directional lstm-cnns-crf. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1064–1074.

Makoto Miwa and Mohit Bansal. 2016. End-to-end relation extraction using lstms on sequences and tree structures. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1105–1116.

Seungwhan Moon, Leonardo Neves, and Vitor Carvalho. 2018. Multimodal named entity recognition for short social media posts. In *NAACL*, pages 852–860.

Erik Tjong Kim Sang and Jorn Veenstra. 1999. Representing text chunks. In *Ninth Conference of the European Chapter of the Association for Computational Linguistics*, pages 173–179.

Peter Shaw, Jakob Uszkoreit, and Ashish Vaswani. 2018. Self-attention with relative position representations. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 2 (Short Papers)*, pages 464–468.

Dianbo Sui, Yubo Chen, Kang Liu, Jun Zhao, and Shengping Liu. 2019. Leverage lexical knowledge for chinese named entity recognition via collaborative graph network. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 3830–3840.

Lin Sun, Jiquan Wang, Kai Zhang, Yindu Su, and Fangsheng Weng. 2021. Rpbert: a text-image relation propagation-based bert model for multimodal ner. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 35, pages 13860–13868.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in neural information processing systems*, pages 5998–6008.

Zhengyuan Yang, Boqing Gong, Liwei Wang, Wenbing Huang, Dong Yu, and Jiebo Luo. 2019. A fast and accurate one-stage approach to visual grounding. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 4683–4693.

Yongjing Yin, Fandong Meng, Jinsong Su, Chulun Zhou, Zhengyuan Yang, Jie Zhou, and Jiebo Luo. 2020. A novel graph-based multi-modal fusion encoder for neural machine translation. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 3025–3035.

Jianfei Yu, Jing Jiang, Li Yang, and Rui Xia. 2020. Improving multimodal named entity recognition via entity span detection with unified multimodal transformer. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 3342–3352.

Dong Zhang, Xincheng Ju, Junhui Li, Shoushan Li, Qiaoming Zhu, and Guodong Zhou. 2020. Multi-modal multi-label emotion detection with modality and label dependence. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 3584–3593.

Dong Zhang, Suzhong Wei, Shoushan Li, Hanqian Wu, Qiaoming Zhu, and Guodong Zhou. 2021. Multi-modal graph fusion for named entity recognition with targeted visual guidance. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 35, pages 14347–14355.

Qi Zhang, Jinlan Fu, Xiaoyu Liu, and Xuanjing Huang. 2018. Adaptive co-attention network for named entity recognition in tweets. In *Thirty-Second AAAI Conference on Artificial Intelligence*.

Yue Zhang and Jie Yang. 2018. Chinese ner using lattice lstm. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1554–1564.

Shan Zhao, Minghao Hu, Zhiping Cai, Haiwen Chen, and Fang Liu. 2021a. Dynamic modeling cross-and self-lattice attention network for chinese ner. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 35, pages 14515–14523.

Shan Zhao, Minghao Hu, Zhiping Cai, and Fang Liu. 2021b. Modeling dense cross-modal interactions for joint entity-relation extraction. In *Proceedings of the Twenty-Ninth International Conference on International Joint Conferences on Artificial Intelligence*, pages 4032–4038.

A Implementation Details

For each uni-modal and multi-modal approach, the maximum length of the sequence input and batch size are respectively set to 128 and 16. For our FMIT approach, we utilize the pre-trained cased $BERT_{base}$ model with dimension of 768 to initial word representations H_x, and employ a pre-trained 152-layer ResNet with dimension of 2048 to initial the visual representations H_v. The parameters of both pre-trained models keep fine-tuned during training. After dimension aligned, the dimension d of both modalities are transformed into 512. The dropout rate and tradeoff rate λ are respectively set to 0.2 and 0.25. To train our model, we use Adam optimizer with a learning rate of 5e-5 for pre-trained models and 2e-4 for other parameters.

MetaSLRCL: A Self-Adaptive Learning Rate and Curriculum Learning Based Framework for Few-Shot Text Classification

Kailin Zhao, Xiaolong Jin*, Saiping Guan, Jiafeng Guo, Xueqi Cheng
CAS Key Laboratory of Network Data Science and Technology, Institute of Computing Technology, Chinese Academy of Sciences;
School of Computer Science and Technology, University of Chinese Academy of Sciences
{zhaokailin17z, jinxiaolong, guansaiping}@ict.ac.cn
{guojiafeng, cxq}@ict.ac.cn

Abstract

Due to the lack of labeled data in many realistic scenarios, a number of few-shot learning methods for text classification have been proposed, among which the meta learning based ones have recently attracted much attention. Such methods usually consist of a learner as the classifier and a meta learner for specializing the learner to different tasks. For the learner, learning rate is crucial to its performance. However, existing methods treat it as a hyper parameter and adjust it manually, which is time-consuming and laborious. Intuitively, for different tasks and neural network layers, the learning rates should be different and self-adaptive. For the meta learner, it requires a good generalization ability so as to quickly adapt to new tasks. Motivated by these issues, we propose a novel meta learning framework, called MetaSLRCL, for few-shot text classification. Specifically, we present a novel meta learning mechanism to obtain different learning rates for different tasks and neural network layers so as to enable the learner to quickly adapt to new training data. Moreover, we propose a task-oriented curriculum learning mechanism to help the meta learner achieve a better generalization ability by learning from different tasks with increasing difficulties. Extensive experiments on three benchmark datasets demonstrate the effectiveness of MetaSLRCL.

1 Introduction

Text classification is one of the most concerned tasks in Natural Language Processing (NLP). At present, most text classification methods are based on supervised learning with a large amount of labeled data. But there is not so much labeled data, even source data, in many scenarios (e.g., news classification in specific domains). Some distant supervision methods (Mintz et al., 2009) have thus been proposed to handle this problem. However, this kind of methods may add a large proportion of noisy training data (Zeng et al., 2014). Because of this, it is a big challenge for traditional supervised learning methods to work well in the scenarios with very limited training data. As a result, few-shot text classification has attracted much attention in recent years, where there are only a few labeled instances available for each class.

The concept of few-shot learning was formally put forward by (Li et al., 2003). They presented a method for learning from classes with few data, by incorporating generic knowledge which may be obtained from previously learned models of unrelated classes. The existing few-shot learning methods can be divided into three categories (Gao et al., 2019), namely, model fine-tuning based (e.g., (Howard and Ruder, 2018; Nakamura and Harada, 2019)), metric learning based (e.g., (Snell et al., 2017; Vinyals et al., 2016)), and meta learning based methods (e.g., (Finn et al., 2017; Munkhdalai and Yu, 2017)). In recent years, meta learning based methods have attracted lots of interests. However, they still suffer from some challenges.

A meta learning method is composed of a learner and a meta learner. For the learner, learning rate is crucial to its performance. Nevertheless, in existing methods, it is treated as a hyper parameter and needs to be adjusted manually, which is time-consuming and laborious. Intuitively, for different tasks and different neural network layers, their learning rates should be different. On the other hand, the present meta learning methods cannot be quickly generalized to new tasks (Zheng et al., 2021) and a good generalization ability to new tasks is necessary for the meta learner. And curriculum learning can help models obtain better generalization performance by guiding the training process towards better regions in the parameter space, i.e., into local minima of the descent procedure associated with good generalization (Bengio et al., 2009).

*Corresponding author.

Proceedings of the 29th International Conference on Computational Linguistics, pages 2065–2074
October 12–17, 2022.

For the above reasons, we propose a novel meta learning framework, called MetaSLRCL, for few-shot text classification, which contains two main mechanisms, i.e., Self-adaptive Learning Rates for the learner and a task-oriented Curriculum Learning mechanism for the meta learner. Our general contributions are three-fold. 1) We present a novel meta learning mechanism with self-adaptive learning rates, which enables different tasks and neural network layers to obtain different learning rates; 2) We introduce curriculum learning for the first time, to the best of our knowledge, into few-shot learning. Unlike traditional instance-oriented curriculum learning, the proposed task-oriented curriculum learning mechanism gradually learns from different tasks with increasing difficulties; 3) MetaSLRCL is evaluated with three typical types of text classification, i.e., relation classification, news classification and topic classification, on three benchmark datasets, namely, FewRel80, 20Newsgroup and DBPedia Ontology, respectively. Experimental results demonstrate its superior performance on all datasets.

2 Related Works

2.1 Few-shot Learning

Few-shot learning is to learn how to solve problems from few data. As aforesaid, the existing mainstream methods can be divided into three categories. The model fine-tuning based mbethods learn how to fine-tune general-purpose models to specialized tasks (Howard and Ruder, 2018; Nakamura and Harada, 2019). The metric learning based methods learn a semantic embedding space upon a distance function (Snell et al., 2017; Vinyals et al., 2016). The meta learning based methods learn a learning strategy to make them well adapt to new tasks (Finn et al., 2017; Munkhdalai and Yu, 2017). Furthermore, according to the different kinds of meta knowledge the meta learner learns, the meta learning based methods can be further divided into three sub-categories, i.e., initial parameter (Finn et al., 2017; Raghu et al., 2019; Jamal and Qi, 2019), hyper parameter (Wu et al., 2019) and optimizer based methods (Santoro et al., 2016; Munkhdalai and Yu, 2017). The initial parameter based methods learn parameter initialization for fast adaptation; The hyper parameter based methods learn a good hyper parameter setting for the learner; And, the optimizer based methods learn a meta-policy to update the parameters of the learner. Some methods of the hyper parameter based category in Computer Vision (CV) (e.g., MAML++ (Antoniou et al., 2019) and ALFA (Baik et al., 2020)) have explored to learn the learning rate. However, these methods usually consider from a single perspective, e.g., the network layer or loop perspective. Specifically, MAML++ learns the learning rate from the network layer perspective, while ALFA learns it from the loop perspective. Unlike them, this paper proposes a novel meta learning mechanism to self-adaptively obtain the learning rates of the learner, which allocates different learning rates for different tasks and neural network layers.

2.2 Curriculum Learning

Compared with the general paradigm of machine learning without distinction, curriculum learning is proposed to imitate the process of human learning (Bengio et al., 2009). It advocates that the model should start learning from easy instances and gradually advance to hard instances. Curriculum learning has been widely applied in many fields, e.g., CV (Guo et al., 2018; Jiang et al., 2014) and NLP (Platanios et al., 2019; Tay et al., 2019). Furthermore, curriculum learning can also be applied in other technical frameworks, e.g., reinforcement learning (Florensa et al., 2017; Narvekar et al., 2017; Ren et al., 2018), graph learning (Gong et al., 2019; Qu et al., 2018) and continual learning (Wu et al., 2021). In this paper, we extend the traditional instance-oriented curriculum learning to a task-oriented one, which gradually learns from different tasks with increasing difficulties.

3 Notations

In meta learning based few-shot text classification, two datasets are given: D_{train} and D_{test}, which have disjoint label sets. T tasks are sampled from D_{train} and the t-th task ($t \in [1, T]$), $Task_t$, consists of a support set S_t and a query set Q_t. Following the setting (Gao et al., 2019), we adopt C-way K-shot (hereinafter denoted as CwKs) for few-shot text classification, meaning S_t contains C classes and each class has K labeled instances. Thus, S_t can be formulated as $S_t = \{(x_t^i, y_t^i)\}_{i=1}^{C \times K}$, where x_t^i denotes the i-th piece of text in $Task_t$ and y_t^i is its class label. Furthermore, x_t^i contains M_t^i words (hereinafter simplified as M if not causing any confusion) and the m-th word ($m \in [1, M]$) in x_t^i denotes as $w_{t,m}^i$. Thus, $x_t^i = \{w_{t,m}^i\}_{m=1}^{M}$. x_t^i additionally includes a head entity h_t^i and a tail

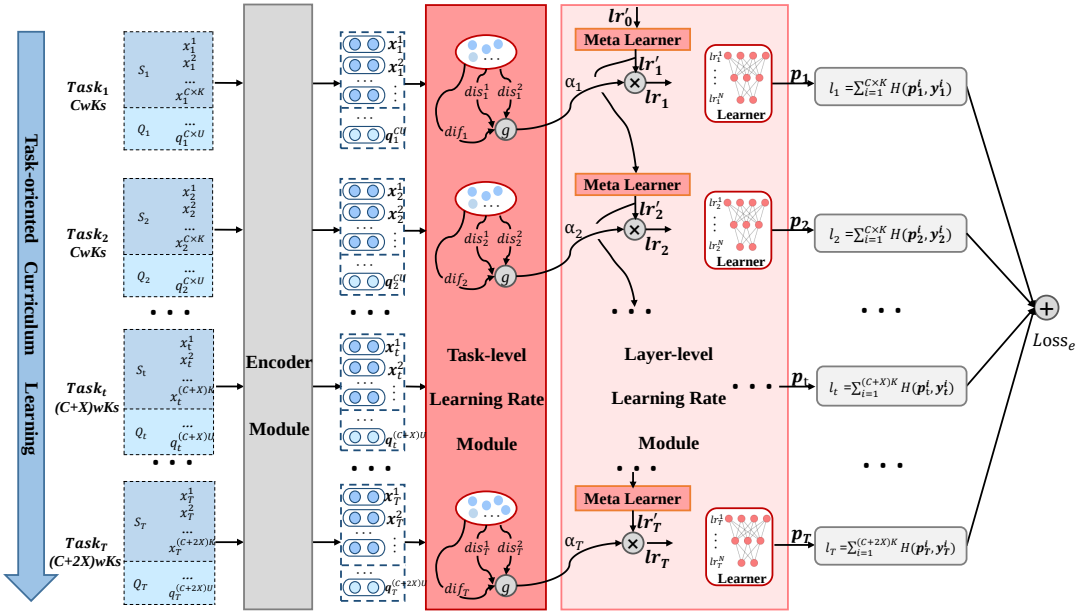

Figure 1: The diagram of the MetaSLRCL framework.

entity o_t^i in relation classification. Moreover, the query set Q_t contains U_t unlabeled instances for each class in S_t, where the i-th instance denotes q_t^i. Q_t can thus be formulated as $Q_t = \{q_t^i\}_{i=1}^{C \times U}$.

4 The MetaSLRCL Framework

MetaSLRCL is a generic framework, where few-shot learning models of different categories (i.e., model fine-tuning based, metric learning based, and meta learning based) can be adopted as the learner. As shown in Figure 1, MetaSLRCL consists of three modules coupled with a task-oriented curriculum learning mechanism.

The Encoder Module. This module maps the instances into the semantic space as embeddings via the encoder network.

The Task-level Learning Rate Module. This module calculates the task-level learning rate via the number of training classes and the distance between different instances in the support set.

The Layer-level Learning Rate Module. In this module, the layer-level learning rate is self-adaptively obtained based on the meta learning mechanism. This module contains two main parts: the learner as the classifier and the meta learner above the learner, which allocates learning rates for different network layers of the learner.

The Task-oriented Curriculum Learning Mechanism. It enables MetaSLRCL to gradually learn from tasks with more and more classes, thus with increasing difficulties, to make the meta learner achieve a better generalization ability.

4.1 The Encoder Module

The encoder module encodes each instance x_t^i into an embedding $\boldsymbol{x}_t^i$. This module consists of two parts, i.e., the embedding part and the encoding part.

In the embedding part, the semantic embeddings $\boldsymbol{w}_{t,m}^i$ for each word $w_{t,m}^i$ in x_t^i is obtained by looking up table. In this paper, we employ GloVe (Pennington et al., 2014) to obtain word embeddings for its fast training and remarkable performance even with small corpus. In the encoding part, the CNN encoder is employed because of its good performance and time efficiency to derive the instance embedding $\boldsymbol{x}_t^i$ of B dimension of x_t^i based on the word embeddings $\{\boldsymbol{w}_{t,m}^i\}_{m=1}^M$. CNN slides a conventional kernel with a window of size k, over the input embeddings to get the output hidden embeddings,

$$\boldsymbol{h}_{t,m}^i = \mathrm{Con}\left(\boldsymbol{w}_{t,m-\frac{k-1}{2}}^i, ..., \boldsymbol{w}_{t,m+\frac{k-1}{2}}^i\right), \quad (1)$$

where $\mathrm{Con}(\cdot)$ is a conventional operation.

A max pooling operation is then applied over these hidden embeddings to output the final instance embedding $\boldsymbol{x}_t^i$ as follows:

$$[\boldsymbol{x}_t^i]_b = max\left\{[\boldsymbol{h}_{t,1}^i]_b, ..., [\boldsymbol{h}_{t,M}]_b\right\}, \quad (2)$$

where $[\cdot]_b$ is the b-th value of a vector ($b \in [1, B]$).

4.2 The Task-level Learning Rate Module

This module is designed to self-adaptively get different learning rates for different tasks. In the context of few-shot learning, it is necessary for a model to converge within only a few steps (Finn et al., 2017). Intuitively, for easy tasks, large learning rates enable the model to converge fast. However, for hard tasks, relatively small learning rates are preferred so as to help the model carefully search for the optimal parameters in the complex search space. In this module, the difficulty of a task is defined as the learning difficulty, measured in terms of the number of training classes and the distance between different instances in the support set.

In more detail, the learning difficulty of a task is related to the number of classes in meta training. If the number, C, of training classes, of $Task_t$ is equal to that of its meta test classes, $C^{'}$, its difficulty coefficient dif_t is set to 1. If C is larger than $C^{'}$, indicating that it is a harder task, dif_t is increased. Otherwise, it is reduced. dif_t can be formally calculated as follows:

$$dif_t = 1 + \gamma \left(C - C^{'} \right), \tag{3}$$

where γ is a coefficient within $[0, 1]$.

The distance between different instances can be measured from two aspects, namely, the average intra-class distance dis_t^1 and the average inter-class distance dis_t^2. The closer the intra-class distance and the farther the inter-class distance, the easier the task. Both of them are measured by the Euclidean distance function $d(\cdot, \cdot)$. Specifically, dis_t^1 is calculated by

$$dis_t^1 = \frac{1}{D_t^1} \sum_{v=1}^{D_t^1} d\left(\boldsymbol{x}_t^i, \boldsymbol{x}_t^j \right), \tag{4}$$

where $\boldsymbol{x}_t^i$ and $\boldsymbol{x}_t^j$ $(i \neq j)$ belong to the same class; $D_t^1 = CK(K-1)/2$, denoting the number of pairs $(\boldsymbol{x}_t^i, \boldsymbol{x}_t^j)$. dis_t^2 is calculated as follows:

$$dis_t^2 = \frac{1}{D_t^2} \sum_{v=1}^{D_t^2} d\left(\boldsymbol{x}_t^i, \boldsymbol{x}_t^j \right), \tag{5}$$

where $\boldsymbol{x}_t^i$ and $\boldsymbol{x}_t^j$ belong to different classes and $D_t^2 = CK(C-1)K/2$. Therefore, the learning rate $\alpha_t^{'}$ of $Task_t$ can be calculated as

$$\alpha_t^{'} = \frac{dis_t^2}{dif_t \cdot dis_t^1}. \tag{6}$$

As aforesaid, larger learning rates are preferred for easier tasks. Therefore, Equation (6) means a larger $\alpha_t^{'}$ is obtained with dis_t^2 increasing, as well as dif_t and dis_t^1 decreasing, which indicates an easier task. Otherwise, a smaller $\alpha_t^{'}$ represents a harder task.

As the task-level learning rate is required to multiply the layer-level one in Equation (12), it should be larger than 1 for easier tasks and smaller than 1 for harder tasks. Therefore, we formulate the task-level learning rate $\alpha_t \in [\beta, 1+\beta]$ by function $g(\cdot)$ as

$$\alpha_t = g\left(\alpha_t^{'}\right) = nor\left(\alpha_t^{'}\right) + \beta, \tag{7}$$

where $nor(\cdot)$ is the min-max normalization function to normalize $\alpha_t^{'}$ between 0 and 1. In this paper, the bias β is set to 0.5.

4.3 The Layer-level Learning Rate Module

As aforementioned, this module contains a learner and a meta learner.

4.3.1 The Learner

In text classification, the learner is actually a classifier. Existing models of different categories can be employed as the learner, e.g., BERT (Kenton and Toutanova, 2019), PN (Snell et al., 2017) and MLMAN (Ye and Ling, 2019), which are pre-trained. By inputting the embedding $\boldsymbol{x}_t^i$, the learner with the learning rate $\boldsymbol{lr}_t$, which is obtained by Equation (12), outputs the predicted probability distribution, $\boldsymbol{p}_t^i$, to different classes. Formally, $\boldsymbol{p}_t^i$ is calculated as follows:

$$\boldsymbol{p}_t^i = Learner\left(\boldsymbol{x}_t^i, \boldsymbol{lr}_t \right). \tag{8}$$

The loss of the learner is defined as l_t, which is calculated by the cross entropy function $H(\cdot, \cdot)$ as

$$l_t = \sum_{i=1}^{C \times K} H\left(\boldsymbol{p}_t^i, \boldsymbol{y}_t^i \right), \tag{9}$$

where $\boldsymbol{y}_t^i$ is the ground truth distribution of $\boldsymbol{x}_t^i$ to different classes.

4.3.2 The Meta Learner

The meta learner allocates different learning rates for different network layers. Let θ be its parameters. Given the layer-level learning rate $\boldsymbol{lr}_{t-1}^{'}$ of N dimension corresponding to $Task_{t-1}$ of the learner, the hidden state $\boldsymbol{hs}_t$ of the meta learner to $Task_t$ is calculated upon $\boldsymbol{lr}_{t-1}^{'}$ and its last hidden state $\boldsymbol{hs}_{t-1}$ as

$$\boldsymbol{hs}_t = MetaLearner_\theta\left(\boldsymbol{hs}_{t-1}, \boldsymbol{lr}_{t-1}^{'} \right). \tag{10}$$

Then, the layer-level learning rate $\boldsymbol{lr}_t^{'}$ of $Task_t$ is obtained upon the state $\boldsymbol{hs}_t$ as

Algorithm 1 The Training Pro. of Meta Learning.

Given a set of labeled training data D_{train}
Init parameters of the meta leaner as θ
Given the initial learning rate $\boldsymbol{lr}_0^{'}$
For $e \rightarrow 1$ to E do:
  Given a pre-trained learner with $\boldsymbol{lr}_0^{'}$
  For $t \rightarrow 1$ to T do:
    Given a task $Task_t$ sampled from D_{train}
    $\boldsymbol{hs}_t \leftarrow MetaLearner_\theta\left(\boldsymbol{hs}_{t-1}, \boldsymbol{lr}_{t-1}^{'}\right)$
    $\boldsymbol{lr}_t^{'} \leftarrow \sigma\left(\boldsymbol{W}\boldsymbol{hs}_t + \boldsymbol{b}\right)$
    $\boldsymbol{lr}_t \leftarrow \alpha_t \boldsymbol{lr}_t^{'}$
    Train the learner with $\boldsymbol{lr}_t$ on $Task_t$ in one step
    Compute the loss l_t
    If $t = T$, calculate the loss $Loss_e$ by summing up l_t
  Update θ using $Loss_{e-1}$- $Loss_e$

$$\boldsymbol{lr}_t^{'} = \sigma\left(\boldsymbol{W}\boldsymbol{hs}_t + \boldsymbol{b}\right), \tag{11}$$

where $\boldsymbol{W}$ and $\boldsymbol{b}$ are parameters of a fully-connected layer and $\sigma(\cdot)$ is the Sigmoid activation function.

By multiplying the task-level learning rate α_t, the final learning rate is obtained as

$$\boldsymbol{lr}_t = \alpha_t \boldsymbol{lr}_t^{'}. \tag{12}$$

The loss of the meta learner in the e-th iteration ($e \in [1, E]$), $Loss_e$ is calculated by summing up the losses l_t of all tasks from the learner as

$$Loss_e = \sum_{t=1}^{T} l_t. \tag{13}$$

Finally, θ is updated by minimizing the difference between the loss in the last iteration and the current loss, which makes the meta learner converge faster, through applying gradient-based optimization. The training process of meta learning is shown in Algorithm 1.

4.4 The Task-oriented Curriculum Learning Mechanism

To get better generalization performance to new tasks, MetaSLRCL introduces a task-oriented curriculum learning mechanism to the meta training period. The original curriculum learning mechanism learns from instances with gradually increasing difficulties in a step-by-step manner. However, in the context of meta learning, we need to pay more attention to tasks with different difficulties. It is acknowledged that when the number of classes in a task increases, its difficulty increases accordingly. For example, a 10w1s task is harder than a 5w1s one. Therefore, a three-stage process with increasing difficulties is carried out with the number of classes increasing from C to C+X and further to C+2X (hereinafter denoting the process as C-(C+X)-(C+2X)), making the meta learner train tasks from easy to hard. Besides, a previous study (Munkhdalai and Yu, 2017) found that the models trained on harder tasks, but tested with relatively easier tasks may achieve better performance, as compared with those models which are trained and tested on tasks with the same difficulty configuration. Thus, in this paper we set that the average difficulty of tasks in the meta training period is always higher than that in the meta test period to get better performance in test tasks.

5 Experiments

5.1 Datasets and Evaluation Metrics

Parameters	Value
γ	0.1
β	0.5
k	3
word emb. dim.	50
max sentence length	40
hidden layer dim.	230
LSTM hidden size	100
initial learning rate	[$7e^{-3}$, $6e^{-3}$, $5e^{-3}$, $4e^{-3}$]
batch size	1
T	600
E	50
dropout	0.2

Table 1: The parameter setting in MetaSLRCL.

To verify the effectiveness of the MetaSLRCL framework, we conduct experiments on three different types of text classification, i.e., relation classification, news classification, and topic classification with three representative benchmark datasets. For relation classification, we choose a typical few-shot learning dataset, FewRel (Han et al., 2018). Note that the FewRel dataset used in this paper has only 80 classes, thus marked as FewRel80, because 20 classes of the original FewRel dataset for test are not publicly available. We randomly divide FewRel80 into three subsets containing 50, 10 and 20 classes for training, validation and test, respectively. For news classification, we choose the representative dataset, 20Newsgroup (Dadgar et al., 2016) with 20 news classes. As 20Newsgroup lacks standard splits in few-shot learning, we randomly divide it into subsets with 14 and 6 classes for training and test, respectively. For topic classification, the DBPedia Ontology (Zhang et al., 2015) dataset is a classic one with 14 topic classes. Similarly, we randomly partition it into 8 classes

Dataset: FewRel80					
Model		5w1s	5w5s	10w1s	10w5s
model fine-tuning based	BERT	0.5762	0.7109	0.5233	0.5480
	MetaSLRCL+BERT	**0.6347**	**0.7601**	**0.5672**	**0.5988**
metric learning based	PN_HATT	0.7319	0.8703	0.6114	0.7632
	MetaSLRCL+PN_HATT	**0.7675**	**0.8929**	**0.6507**	**0.8067**
meta learning based	MLMAN	0.7957	0.9119	0.6903	0.8516
	MetaSLRCL+MLMAN	**0.8182**	**0.9150**	**0.7084**	**0.8519**
Dataset: 20Newsgroup					
Model		3w1s	3w5s	6w1s	6w5s
model fine-tuning based	BERT	0.7417	0.8198	0.5876	0.7107
	MetaSLRCL+BERT	**0.7689**	**0.8476**	**0.6187**	**0.7426**
metric learning based	PN	0.8463	0.9614	0.7052	0.8887
	MetaSLRCL+PN	**0.8680**	**0.9843**	**0.7217**	**0.9264**
meta learning based	MAML	0.7612	0.8405	0.6143	0.7451
	MetaSLRCL+MAML	**0.7824**	**0.8599**	**0.6465**	**0.7738**
Dataset: DBPedia Ontology					
Model		3w1s	3w5s	6w1s	6w5s
model fine-tuning based	BERT	0.7609	0.8256	0.6118	0.7589
	MetaSLRCL+BERT	**0.7928**	**0.8598**	**0.6540**	**0.7990**
metric learning based	PN	0.8428	0.9520	0.7070	0.8896
	MetaSLRCL+PN	**0.8683**	**0.9799**	**0.7301**	**0.9104**
meta learning based	MAML	0.7778	0.8571	0.6434	0.8093
	MetaSLRCL+MAML	**0.8110**	**0.8911**	**0.6786**	**0.8359**

Table 2: The overall results on three benchmark datasets: FewRel80, 20Newsgroup and BDPedia Ontology.

and 6 classes for training and test, respectively.

We set up four configurations, namely, 5w1s, 5w5s, 10w1s and 1w5s, on FewRel80. Four settings are considered for the 20Newsgroup and DBPedia Ontology datasets, i.e., 3w1s, 3w5s, 6w1s and 6w5s. Following the previous study in (Obamuyide and Vlachos, 2019), average accuracy upon 5 runs is adopted as the evaluation metric.

5.2 Implementation Details and Parameters Setting

Table 1 presents the parameter setting of MetaSLRCL. For the encoder module, CNN is employed as the encoder and the word embeddings pre-trained in GloVe (Pennington et al., 2014) are adopted as the initial embeddings. More specifically, we choose the embedding set of GloVe trained on Wikipedia 2014 + Gigaword 5, which contains 6B tokens and 400K words. The word embeddings are of 50 dimensions. For the parameters of CNN, we follow the settings used in (Zeng et al., 2014). For the layer-level learning rate module, LSTM is selected as the meta learner, because of its simple implementation, fast training speed and remarkable performance. Furthermore, for the curriculum learning, we choose one setting with best performance on each dataset, specifically, 10-15-20 on FewRel80, 7-9-11 on 20Newsgroup and 5-6-7 on DBPedia Ontology.

5.3 Baseline Models

As MetaSLRCL is a generic framework, it can employ different few-shot learning models as its learner. Therefore, in the experiments, we adopt the representative and state-of-the-art (SOTA) models of the aforesaid different categories as the learner of MetaSLRCL in order to verify its effectiveness on different tasks. These models are also adopted as the baselines for performance comparison. It should be particularly mentioned that for the sake of space limitation, for each type of text classification and each category of the few-shot learning models, the experimental results of only the baseline models (e.g., MAML) with the best performance and their MetaSLRCL counterparts (e.g., MetaSLRCL+MAML) will be presented. More specifically, for relation classification, the baseline models include: 1) BERT-base-uncased (Kenton and Toutanova, 2019), a widely adopted model of model fine-tuning based category; 2) PN_HATT (Gao et al., 2019), the SOTA metric learning based model especially for relation classification; 3) MLMAN (Ye and Ling, 2019), the SOTA model in few-shot relation classification. For news classification and topic classifica-

Model		5w1s	5w5s	10w1s	10w5s
model fine-tuning based	MetaSLRCL+BERT	**0.6347**	**0.7601**	**0.5672**	**0.5988**
	SLR+BERT	0.6174	0.7456	0.5532	0.5851
	CL+BERT	0.5904	0.7263	0.5370	0.5615
metric learning based	MetaSLRCL+PN_HATT	**0.7675**	**0.8929**	**0.6507**	**0.8067**
	SLR+PN_HATT	0.7592	0.8831	0.6435	0.7982
	CL+PN_HATT	0.7380	0.8719	0.6152	0.7792
meta learning based	MetaSLRCL+MLMAN	**0.8182**	**0.9150**	**0.7084**	0.8519
	SLR+MLMAN	0.8103	0.9145	0.7059	**0.8541**
	CL+MLMAN	0.8167	0.9136	0.7042	0.8507

Table 3: The results of the ablation study on SLR and CL on FewRel80.

Model		5w1s	5w5s	10w1s	10w5s
model fine-tuning based	SLR+BERT	**0.6174**	**0.7456**	**0.5532**	**0.5851**
	SLRL+BERT	0.6145	0.7412	0.5509	0.5823
	SLRT+BERT	0.5771	0.7148	0.5261	0.5502
metric learning based	SLR+PN_HATT	**0.7592**	**0.8831**	**0.6435**	**0.7982**
	SLRL+PN_HATT	0.7578	0.8811	0.6414	0.7956
	SLRT+PN_HATT	0.7354	0.8723	0.6137	0.7648
meta learning based	SLR+MLMAN	**0.8103**	**0.9145**	**0.7059**	**0.8541**
	SLRL+MLMAN	0.8095	0.9139	0.7051	0.8537
	SLRT+MLMAN	0.7982	0.9125	0.6931	0.8522

Table 4: The results of the ablation study on SLRs on FewRel80.

tion, the baseline models are the same, including: 1) BERT-base-uncased, for the same reason; 2) PN (Snell et al., 2017), a widely adopted metric learning based model; 3) MAML (Finn et al., 2017), a widely adopted meta learning based model.

5.4 Main Results

Table 2 presents the main results, where we can see that all of the MetaSLRCL models with BERT, PN_HATT, MLMAN, PN and MAML as their learners consistently outperform those corresponding baselines on all datasets. The accuracy of the model fine-tuning based and metric learning based MetaSLRCL models increases by 4-6% and 2-4% on FewRel80, respectively. However, for MetaSLRCL+MLMAN, its performance is improved less than those of the former two categories; But it still achieves the best results. Moreover, all kinds of MetaSLRCL models are observed accuracy promotion by 2-4% compared to the baselines on the majority of few-shot tasks on 20Newsgroup and DBPedia Ontology. In short, these experimental results convincingly suggest that MetaSLRCL is effective for different tasks on different datasets and with different models.

5.5 Ablation Studies

5.5.1 SLR and CL in MetaSLRCL

In this subsection, we conduct ablation studies to investigate the effectiveness of both Self-adaptive Learning Rate (SLR) and Curriculum Learning (CL), as well as their impacts on the performance of MetaSLRCL. For the sake of space limitation, only the results on FewRel80 are presented. As shown in Table 3, the performance of all ablated models without SLR and CL consistently falls, except MLMAN on the 10w5s task. For each type of the models in this table, we adopt the same CL setting on different tasks, with which the MetaSLRCL enhanced model exhibits best performance on most of them. Therefore, for the MLMAN models, the 10-15-20 CL setting is selected, because under this CL setting the MetaSLRCL+MLMAN model achieves the best results on the 5w1s, 5w5s and 10w1s tasks. Nevertheless, on the 10w5s task, MetaSLRCL+MLMAN obtains its best performance with the CL setting of 15-20-25. For this reason, SLR+MLMAN exceptionally outperforms MetaSLRCL+MLMAN on the 10w5s task. The general results in Table 3 indicate that both SLR and CL contribute to the effectiveness of MetaSLRCL. Besides, it can be observed that SLR is more important to MetaSLRCL than CL, because of the larger performance improvement. Similar phenomena can be observed on the other datasets, 20Newsgroup and DBPedia Ontology.

5.5.2 SLRs for Tasks and Network Layers

In MetaSLRCL, SLR consists of two subsets, the Self-adaptive Learning Rates for different

Model	5w1s	5w5s
Adadelta+BERT	0.5825	0.7232
RMSProp+BERT	0.5887	0.7203
Adam+BERT	0.5943	0.7261
SLR+BERT	**0.6174**	**0.7456**
Adadelta+PN_HATT	0.7386	0.8612
RMSProp+PN_HATT	0.7327	0.8446
Adam+PN_HATT	0.7101	0.8300
SLR+PN_HATT	**0.7592**	**0.8831**
Adadelta+MLMAN	0.7995	0.9063
RMSProp+MLMAN	0.8007	0.9087
Adam+MLMAN	0.8027	0.9108
SLR+MLMAN	**0.8103**	**0.9145**

Table 5: The results of different models with SLR and other self-adaptive learning rate mechanisms on FewRel80.

Tasks (SLRT) and different neural network Layers (SLRL). As shown in Table 4, the performance of all models without SLRT and SLRL consistently decreases, indicating that both SLRT and SLRL are important to the effectiveness of SLR. However, the models with SLRL outperform those with SLRT. That means, although both task-level and layer-level learning rates work, the layer-level ones are more important and effective to the performance of models than their counterparts.

5.6 SLR Comparing with Other Self-Adaptive Learning Rate Methods

We also conduct some experiments to compare our SLR with other self-adaptive learning rate mechanisms, i.e., Adadelta (Zeiler, 2012), RMSProp (Hinton et al., 2012) and Adam (Kingma and Ba, 2014), on FewRel80. The parameters of these methods are tuned on our dataset. The experimental results are shown in Table 5. It can be noted that, the models with our SLR outperform all the others, which indicates that our SLR is more effective than the others. Moreover, as compared with Adadelta, the performance of RMSProp and Adam are unstable when coupled with different models, i.e., BERT, PN_HATT, and MLMAN. Differently, our SLR exhibits consistently the best performance in all cases, indicating that our SLR is more robust than the others when applied to different models.

As mentioned in Section 2, there have already been some models in CV, which explore self-adaptive learning rates, e.g., MAML++ and ALFA. We experimentally compare our SLR in the MetaSLRCL framework with them at the method level. The experimental results are shown in Table 6. Note that for fair comparison, the same initial learning rate as ours is adopted. As we can see, the accuracy of MAML++ and ALFA is lower than all of the MetaSLRCL models with our SLR. It suggests that although MAML++ and ALFA achieve superior performance in CV, our SLR outperforms them on text classification.

Model		5w1s	5w5s
CV	MAML++	0.5823	0.6954
	ALFA	0.6009	0.7137
Ours	SLR+BERT	0.6174	0.7456
	SLR+PN_HATT	0.7592	0.8831
	SLR+MLMAN	**0.8103**	**0.9145**

Table 6: The results of self-adaptive learning rate models in CV and our SLR on FewRel80.

Model	5w1s	5w5s
SLR+5-10-15+BERT	0.6285	0.7498
SLR+10-15-20+BERT	**0.6347**	**0.7601**
SLR+15-20-25+BERT	0.6315	0.7581
SLR+20-25-30+BERT	0.6239	0.7475
SLR+5-10-15+PN_HATT	0.7562	0.8836
SLR+10-15-20+PN_HATT	0.7565	**0.8929**
SLR+15-20-25+PN_HATT	**0.7675**	0.8877
SLR+20-25-30+PN_HATT	0.7645	0.8926
SLR+5-10-15+MLMAN	0.8102	0.9135
SLR+10-15-20+MLMAN	**0.8182**	0.9150
SLR+15-20-25+MLMAN	0.8133	**0.9161**
SLR+20-25-30+MLMAN	0.8046	0.9146

Table 7: The results of different CL settings on FewRel80.

5.7 Different CL Settings

We also conduct experiments to evaluate the impact of the CL mechanism. Specifically, we set up four training settings for each task on FewRel80, namely, 5-10-15, 10-15-20, 15-20-25 and 20-25-30. For the sake of space limitation, only results on 5w1s and 5w5s are shown in Table 7, which demonstrate that all the best results are obtained at two settings, 10-15-20 and 15-20-25. This may be due to the following reason: the 5-10-15 configuration is the simplest one, which does not reach the difficulty to get the best performance of a model, whilst the 20-25-30 configuration is too hard and the learner cannot be well trained at the training period and thus cannot work well at the test period.

Furthermore, four training settings, namely, 3-5-7, 5-7-9, 7-9-11 and 9-11-13 are examined on 20Newsgroup. Four training settings, i.e., 3-4-5, 4-5-6, 5-6-7 and 6-7-8 are also studied on DBPedia Ontology. Similar phenomena can be observed on these datasets. The results are not presented due to

space limitation.

6 Conclusion and Future Work

In this paper, we proposed a novel meta learning framework, called MetaSLRCL, for few-shot text classification. MetaSLRCL can self-adaptively obtain different learning rates for different tasks and different network layers. Moreover, a task-oriented curriculum learning mechanism is introduced into few-shot learning to achieve a better generalization ability for the meta learner. MetaSLRCL is evaluated with three typical types of text classification, relation classification, news classification and topic classification, on three benchmark datasets, namely, FewRel80, 20Newsgroup and DBPedia Ontology, respectively. Experimental results demonstrate superior performance of MetaSLRCL on all datasets. In the future, we will explore few-shot learning under the unbalance learning scenarios because they are ubiquitous in the real world.

Acknowledgments

This work is partially supported by the National Natural Science Foundation of China under grants U1911401, 62002341, and 61772501, the GFKJ Innovation Program, and the Lenovo-CAS Joint Lab Youth Scientist Project.

References

Antreas Antoniou, Harri Edwards, and Amos Storkey. 2019. How to train your maml. In *Seventh International Conference on Learning Representations*. 2.1

Sungyong Baik, Myungsub Choi, Janghoon Choi, Heewon Kim, and Kyoung Mu Lee. 2020. Meta-learning with adaptive hyperparameters. *Advances in Neural Information Processing Systems*, 33:20755–20765. 2.1

Yoshua Bengio, Jérôme Louradour, Ronan Collobert, and Jason Weston. 2009. Curriculum learning. In *Proceedings of the 26th annual international conference on machine learning*, pages 41–48. 1, 2.2

Seyyed Mohammad Hossein Dadgar, Mohammad Shirzad Araghi, and Morteza Mastery Farahani. 2016. A novel text mining approach based on tf-idf and support vector machine for news classification. In *2016 IEEE International Conference on Engineering and Technology (ICETECH)*, pages 112–116. IEEE. 5.1

Chelsea Finn, Pieter Abbeel, and Sergey Levine. 2017. Model-agnostic meta-learning for fast adaptation of deep networks. In *International Conference on Machine Learning*, pages 1126–1135. PMLR. 1, 2.1, 4.2, 5.3

Carlos Florensa, David Held, Markus Wulfmeier, Michael Zhang, and Pieter Abbeel. 2017. Reverse curriculum generation for reinforcement learning. In *Conference on robot learning*, pages 482–495. PMLR. 2.2

Tianyu Gao, Xu Han, Zhiyuan Liu, and Maosong Sun. 2019. Hybrid attention-based prototypical networks for noisy few-shot relation classification. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, pages 6407–6414. 1, 3, 5.3

Chen Gong, Jian Yang, and Dacheng Tao. 2019. Multi-modal curriculum learning over graphs. *ACM Transactions on Intelligent Systems and Technology (TIST)*, 10(4):1–25. 2.2

Sheng Guo, Weilin Huang, Haozhi Zhang, Chenfan Zhuang, Dengke Dong, Matthew R Scott, and Dinglong Huang. 2018. Curriculumnet: Weakly supervised learning from large-scale web images. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 135–150. 2.2

Xu Han, Hao Zhu, Pengfei Yu, Ziyun Wang, Yuan Yao, Zhiyuan Liu, and Maosong Sun. 2018. Fewrel: A large-scale supervised few-shot relation classification dataset with state-of-the-art evaluation. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 4803–4809. 5.1

Geoffrey Hinton, Nitish Srivastava, and Kevin Swersky. 2012. Rmsprop: Divide the gradient by a running average of its recent magnitude. *Neural networks for machine learning, Coursera lecture 6e*, page 13. 5.6

Jeremy Howard and Sebastian Ruder. 2018. Universal language model fine-tuning for text classification. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 328–339. 1, 2.1

Muhammad Abdullah Jamal and Guo-Jun Qi. 2019. Task agnostic meta-learning for few-shot learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 11719–11727. 2.1

Lu Jiang, Deyu Meng, Teruko Mitamura, and Alexander G Hauptmann. 2014. Easy samples first: Self-paced reranking for zero-example multimedia search. In *Proceedings of the 22nd ACM international conference on Multimedia*, pages 547–556. 2.2

Jacob Devlin Ming-Wei Chang Kenton and Lee Kristina Toutanova. 2019. Bert: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of NAACL-HLT*, pages 4171–4186. 4.3.1, 5.3

Diederik P Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. *arXiv e-prints*, pages arXiv–1412. 5.6

Fei-Fei Li et al. 2003. A bayesian approach to unsupervised one-shot learning of object categories. In *Proceedings Ninth IEEE International Conference on Computer Vision*, pages 1134–1141. IEEE. 1

Mike Mintz, Steven Bills, Rion Snow, and Dan Jurafsky. 2009. Distant supervision for relation extraction without labeled data. In *Proceedings of the Joint Conference of the 47th Annual Meeting of the ACL and the 4th International Joint Conference on Natural Language Processing of the AFNLP*, pages 1003–1011. 1

Tsendsuren Munkhdalai and Hong Yu. 2017. Meta networks. In *International Conference on Machine Learning*, pages 2554–2563. PMLR. 1, 2.1, 4.4

Akihiro Nakamura and Tatsuya Harada. 2019. Revisiting fine-tuning for few-shot learning. *arXiv preprint arXiv:1910.00216*. 1, 2.1

Sanmit Narvekar, Jivko Sinapov, and Peter Stone. 2017. Autonomous task sequencing for customized curriculum design in reinforcement learning. In *IJCAI*, pages 2536–2542. 2.2

Abiola Obamuyide and Andreas Vlachos. 2019. Model-agnostic meta-learning for relation classification with limited supervision. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 5873–5879. 5.1

Jeffrey Pennington, Richard Socher, and Christopher D Manning. 2014. Glove: Global vectors for word representation. In *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*, pages 1532–1543. 4.1, 5.2

Emmanouil Antonios Platanios, Otilia Stretcu, Graham Neubig, Barnabas Poczos, and Tom M Mitchell. 2019. Competence-based curriculum learning for neural machine translation. In *Proceedings of NAACL-HLT*, pages 1162–1172. 2.2

Meng Qu, Jian Tang, and Jiawei Han. 2018. Curriculum learning for heterogeneous star network embedding via deep reinforcement learning. In *Proceedings of the Eleventh ACM International Conference on Web Search and Data Mining*, pages 468–476. 2.2

Aniruddh Raghu, Maithra Raghu, Samy Bengio, and Oriol Vinyals. 2019. Rapid learning or feature reuse? towards understanding the effectiveness of maml. In *International Conference on Learning Representations*. 2.1

Zhipeng Ren, Daoyi Dong, Huaxiong Li, and Chunlin Chen. 2018. Self-paced prioritized curriculum learning with coverage penalty in deep reinforcement learning. *IEEE transactions on neural networks and learning systems*, 29(6):2216–2226. 2.2

Adam Santoro, Sergey Bartunov, Matthew Botvinick, Daan Wierstra, and Timothy Lillicrap. 2016. One-shot learning with memory-augmented neural networks. *arXiv preprint arXiv:1605.06065*. 2.1

Jake Snell, Kevin Swersky, and Richard Zemel. 2017. Prototypical networks for few-shot learning. In *Proceedings of the 31st International Conference on Neural Information Processing Systems*, pages 4080–4090. 1, 2.1, 4.3.1, 5.3

Yi Tay, Shuohang Wang, Anh Tuan Luu, Jie Fu, Minh C Phan, Xingdi Yuan, Jinfeng Rao, Siu Cheung Hui, and Aston Zhang. 2019. Simple and effective curriculum pointer-generator networks for reading comprehension over long narratives. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 4922–4931. 2.2

Oriol Vinyals, Charles Blundell, Timothy Lillicrap, Daan Wierstra, et al. 2016. Matching networks for one shot learning. *Advances in neural information processing systems*, 29:3630–3638. 1, 2.1

Jiawei Wu, Wenhan Xiong, and William Yang Wang. 2019. Learning to learn and predict: A meta-learning approach for multi-label classification. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 4354–4364. 2.1

Tongtong Wu, Xuekai Li, Yuan-Fang Li, Gholamreza Haffari, Guilin Qi, Yujin Zhu, and Guoqiang Xu. 2021. Curriculum-meta learning for order-robust continual relation extraction. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 35, pages 10363–10369. 2.2

Zhi-Xiu Ye and Zhen-Hua Ling. 2019. Multi-level matching and aggregation network for few-shot relation classification. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 2872–2881. 4.3.1, 5.3

Matthew D Zeiler. 2012. Adadelta: an adaptive learning rate method. *arXiv preprint arXiv:1212.5701*. 5.6

Daojian Zeng, Kang Liu, Siwei Lai, Guangyou Zhou, and Jun Zhao. 2014. Relation classification via convolutional deep neural network. In *Proceedings of COLING 2014, the 25th International Conference on Computational Linguistics: Technical Papers*, pages 2335–2344. 1, 5.2

Xiang Zhang, Junbo Zhao, and Yann LeCun. 2015. Character-level convolutional networks for text classification. *Advances in neural information processing systems*, 28:649–657. 5.1

Jianming Zheng, Fei Cai, Wanyu Chen, Wengqiang Lei, and Honghui Chen. 2021. Taxonomy-aware learning for few-shot event detection. In *Proceedings of the Web Conference 2021*, pages 3546–3557. 1

A Simple Temporal Information Matching Mechanism for Entity Alignment Between Temporal Knowledge Graphs

Li Cai[1,2], **Xin Mao**[1], **Meirong Ma**[3], **Hao Yuan**[3], **Jianchao Zhu**[3], **Man Lan**[1] *
[1]School of Computer Science and Technology, East China Normal University
[2]College of Computer Science and Technology, Guizhou University
[3]Shanghai Transsion Co., Ltd
caili2020stu@gmail.com, xmao@stu.ecnu.edu.cn
{meirong.ma, hao.yuan, jianchao.zhu}@transsion.com
mlan@cs.ecnu.edu.cn

Abstract

Entity alignment (EA) aims to find entities in different knowledge graphs (KGs) that refer to the same object in the real world. Recent studies incorporate temporal information to augment the representations of KGs. The existing methods for EA between temporal KGs (TKGs) utilize a time-aware attention mechanism to incorporate relational and temporal information into entity embeddings. The approaches outperform the previous methods by using temporal information. However, we believe that it is not necessary to learn the embeddings of temporal information in KGs since most TKGs have uniform temporal representations. Therefore, we propose a simple graph neural network (GNN) model combined with a temporal information matching mechanism, which achieves better performance with less time and fewer parameters. Furthermore, since alignment seeds are difficult to label in real-world applications, we also propose a method to generate unsupervised alignment seeds via the temporal information of TKG. Extensive experiments on public datasets indicate that our supervised method significantly outperforms the previous methods and the unsupervised one has competitive performance.

1 Introduction

Knowledge graphs (KGs) describe facts of the real world in a structured form of triples (h, r, t), where h represents the head entity, r represents the relation, t represents the tail entity. KGs have drawn great research attention from the academia (Lin et al., 2021; Ji et al., 2022) and have been widely used to enhance downstream applications such as question answering (Saxena et al., 2020; Qiu et al., 2020) and recommendation systems (Anelli et al., 2021; Zhou et al., 2020).

Recently, Wikidata (Vrandecic and Krötzsch, 2014) and YOGO2 (Hoffart et al., 2013) add temporal information to the relations to represent the KGs more accurately. In temporal knowledge graphs (TKGs), triples are extended to quadruples as (h, r, t, τ), where τ represents the timestamps. Figure 1 is an example of entity alignment (EA) between TKGs, where nodes represent entities, edges include relational and temporal information, and the quadruple *(Ronaldinho, playsfor, A.C.Milan, 2008-2010)* represents one of the facts.

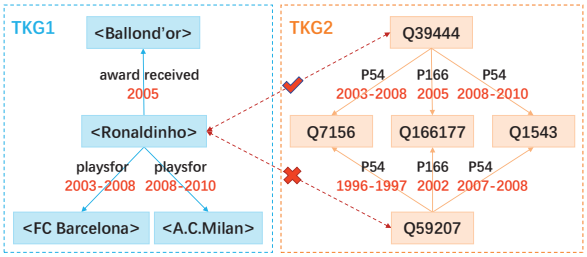

Figure 1: One sample of EA between TKGs. TKG1 is a sub-graph in YAGO. TKG2 is a sub-graph in WIKI, where the entity names corresponding to the items are as follows: *Q39444 (Ronaldinho), Q7156 (FC Barcelona), Q166177 (Ballon d'Or), Q1543 (A.C.Milan), Q59207 (Ronaldo), P54 (member of sports team), P166(award received).*

EA seeks to find the same entities in the real world from different KGs to promote knowledge fusion. Many embedding-based methods perform effective EA (Sun et al., 2020; Mao et al., 2022), which encode entities in a continuous embedding space and align entities based on the learned embeddings. Previous EA methods ignore the temporal information despite their successes, a key factor indicating when a fact occurred.

In order to utilize the time information in TKGs, TEA-GNN (Xu et al., 2021) first proposes a time-aware GNN for EA between TKGs, which incorporates the temporal information via a time-aware attention mechanism to learn the embeddings of entities. TREA (Xu et al., 2022) develops a temporal relational attention mechanism that introduces temporal embedding and relational embedding to entity embedding and achieves state-of-the-art (SOTA)

*Corresponding author

Proceedings of the 29th International Conference on Computational Linguistics, pages 2075–2086
October 12–17, 2022.

performance.

In different TKGs, equivalent entity and relation pairs are usually disrupted and have different representation forms. As shown in Figure 1, the entity of *"Ronaldinho"* is stored in the form of <*Ronaldinho*> in TKG1. But in TKG2, the same entity is stored as a pure id <*Q39444*>. The relation of *"plays for"* is stored in the form of <*playsfor*> and <*P54*> in the two TKGs respectively. Therefore, we need to map different representations into a unified vector space to infer whether they are similar. However, the temporal information is uniform across most TKGs. For example, the temporal information *"2005"* is the same in the two TKGs, and refers to the same year. So we believe it's unnecessary to learn additional temporal embeddings. In addition, different entities can be distinguished by their temporal information even though they have similar structures and relations. Such as <*Ronaldinho*> and *Q59207 (Ronaldo)* in Figure 1. In summary, we can directly use the temporal information for enhancing the prediction, rather than learning the temporal embeddings, which is redundant and time-consuming.

Based on this finding, we propose a **S**imple GNN model combined with a **T**emporal information matching mechanism for **E**ntity **A**lignment (STEA) between TKGs. Unlike the previous work, the model adopts a temporal information matching mechanism instead of learning temporal embeddings, which is effective and efficient. Our proposed model first takes a simple GNN to learn the embeddings of entities by using their structural and relational information, then compares the entities' temporal information, and finally aligns the entities by combining their embedding similarity and time similarity. Experimental results on four datasets show that our model significantly outperforms TREA (Xu et al., 2022). The improvement scores of Hits@1 are 3.4%, 3.6%, 2.2%, 12.3%, respectively.

Most existing methods of EA need alignment seeds to train the model and learn the embeddings of the entities. Finding the alignment seeds of different KGs in the real world is labor-intensive. Some unsupervised EA methods (Liu et al., 2021; Qi et al., 2021; Mao et al., 2021) automatically generate alignment seeds via the image or name of entities. Inspired by these methods, we believe that uniform temporal information can also be used to generate alignment seeds. In this paper, we assume that entities with the same temporal information have high similarity and calculate the time similarity of entities according to their temporal information. If two entities are both the unique nearest neighbor to each other, then the two entities are regarded as alignment seeds. Experimental results show that the proposed unsupervised method outperforms the previous SOTA models on two datasets and competes with our supervised method STEA.

The main contributions of this paper are summarized as follows:

(1) We propose a simple GNN model combined with a temporal information matching mechanism, achieving significant performance with fewer parameters and less time.

(2) By assuming that entities with the same temporal information have high similarity, we propose a simple strategy to generate alignment seeds by selecting the nearest neighbor.

(3) Extensive experiments on public datasets indicate that our supervised method outperforms all SOTA methods. The performance of using automatically generated alignment seeds is close to that of the supervised method.

2 Related Work

2.1 Translation-based Model

Translation-based models learn the embeddings of entities by the triples of KGs, which interpret relation as translation operations from its head to its tail, such as $\boldsymbol{h} + \boldsymbol{r} \approx \boldsymbol{t}$. MTransE (Chen et al., 2017) is the first model to use TransE (Bordes et al., 2013) for entity alignment which maps two KGs into different vector spaces. Entities with similar positions in two vector spaces are alignment pairs. In addition to using relational triples, JAPE (Sun et al., 2017) also utilizes the attribute features of entities to learn the representation of entities to improve the performance of EA. BootEA (Sun et al., 2018) proposes a bootstrapping process by adding label likely alignment entities into training data iteratively to promote the EA results. MultiKE (Hu et al., 2019) represents entities based on mutiaspects of KGs to enhance the alignment. These methods use the triples to align entities independently and lack utilization of the global structural information of KGs.

2.2 GNN-based Model

Due to the powerful ability of GNN to model the structure of KGs, many GNN-based models have been proposed and have achieved good performance on the task of EA. GCN-Align (Wang et al., 2018) is the first method to use GCN (Kipf and Welling, 2017) for entity alignment. GCN-Align combines the attribute features of entities with the structural features of the KG and uses GCN to map the entities to a low-dimensional vector space so that the equivalent entities are close to each other in the space. However, GCN-Align does not effectively utilize the relational features in the knowledge graph. MRAEA (Mao et al., 2020a) proposes a meta relation aware EA method which leverages meta relation-aware embedding and relation-aware self-attention to align the KGs. RREA (Mao et al., 2020b) designs a relational reflection transformation operation to preserve the similarity distributions of entities and integrate them into GNNs to facilitate EA. TEA-GNN (Xu et al., 2021) adopts a similar idea from RREA and utilizes additional temporal information for EA between TKGs. It learns the embeddings of entities by a time-aware attention mechanism to promote EA. TREA (Xu et al., 2022) uses a temporal relational attention mechanism to integrate relational and temporal features of links between nodes to enhance EA. In TKGs, the temporal information representing the same timestamps is uniform. We propose a simple GNN model combined with a temporal information matching mechanism, which is effective and efficient.

2.3 Unsupervised Methods for Alignment Seeds Generation

In recent years, some unsupervised methods have emerged to generate alignment seeds to solve the resource-consuming problem. IMUSE (He et al., 2019) uses both attribute triples and relation triples of KGs to collect seed with high text similarity. EVA (Liu et al., 2021) leverages the additional visual information of entities to create alignment seeds. PRASE (Qi et al., 2021) uses PARIS to obtain the alignment seeds by literal attributes of the entities. These approaches incorporate additional information such as visual and text information of entities to obtain the alignment seeds. Inspired by these methods, we propose a simple alignment seeds generation method by utilizing the temporal information of entities.

3 Problem Formulation

TKGs store the real-world knowledge in the form of quadruples (h, r, t, τ). A TKG is represented as $G = (E, R, T, Q)$, where E, R and T represent the sets of entities, relations and timestamps respectively, $Q \subset E \times R \times E \times T$ denotes the set of quadruples. Defining $G_1 = (E_1, R_1, T_1, Q_1)$ and $G_2 = (E_2, R_2, T_2, Q_2)$ to be two TKGs, $S = \{(e_{1_i}, e_{2_j}) | e_{1_i} \in E_1, e_{2_j} \in E_2\}$ is the set of alignment seeds between G_1 and G_2. Specifically, the timestamps in the two time set has been merged in a uniform time set $T^* = T_1 \cup T_2$. Therefore, the two TKGs can be renewed as $G_1 = (E_1, R_1, T^*, Q_1)$ and $G_2 = (E_2, R_2, T^*, Q_2)$ sharing the same set of timestamps. EA task aims to find new alignment entities set P based on the alignment seeds S between the two TKGs.

4 The Proposed Approach

This research proposes a simple GNN model combined with a temporal information matching mechanism for EA between TKGs using the entity's structural information, relational information, and temporal information. During the training phase, the simple GNN is used to learn entities' structural and relational embeddings. In the prediction stage, the model predicts the new alignment entities by combining the embedding similarity and temporal matching degree of entities. Experimental results show that our model achieves good performance by balancing the embedding similarity with the time similarity of entities. Figure 2 shows the framework of STEA.

4.1 Simple Graph Neural Network

The simple GNN model learns the embeddings of entities by using their structural and relational information. The model first fuses entities' structural features and relational features, then aggregates them through a simple graph convolution operation. Finally, it combines entity embeddings of different layers to obtain a global entity embedding. The sub-modules of the model include feature fusion, feature aggregation, and global embedding generation.

Feature Fusion

Some studies (Yang et al., 2019; Mao et al., 2020b) believe that both the structural information and the relational information are beneficial to the representation of entities. Therefore, we fuse the relational

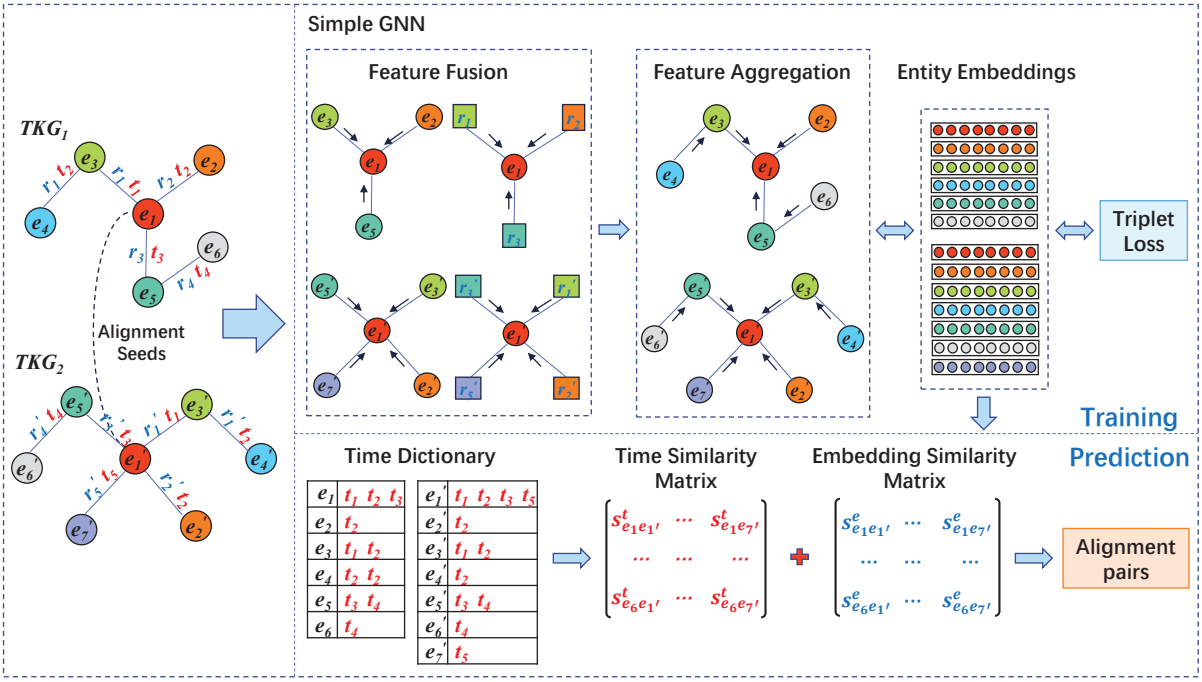

Figure 2: Framework of STEA. The solid line between entities represents the existent relations in TKGs, the dashed line represents the alignment seeds.

features and structural features to get semantic embeddings of entities. The structural feature and relational feature of entity e_i are calculate by the following equations:

$$\boldsymbol{h}^{e}_{e_i} = \frac{1}{|\mathcal{N}^{e}_{e_i}|} \sum_{e_j \in \mathcal{N}^{e}_{e_i}} \boldsymbol{h}_{e_j} \tag{1}$$

$$\boldsymbol{h}^{r}_{e_i} = \frac{1}{|\mathcal{N}^{r}_{e_i}|} \sum_{r_j \in \mathcal{N}^{r}_{e_i}} \boldsymbol{h}_{r_j} \tag{2}$$

where $\mathcal{N}^{e}_{e_i}$ represents the neighboring entity set of e_i, $\boldsymbol{h}_{e_i}$ is the randomly generated initialization embedding of e_i, $\mathcal{N}^{r}_{e_i}$ represents the set of the relations around entity e_i, $\boldsymbol{h}_{r_i}$ is the randomly generated initialization embedding of r_i.

The fused feature of entity e_i is obtained as follows:

$$\boldsymbol{h}^{(1)}_{e_i} = [\boldsymbol{h}^{e}_{e_i} || \boldsymbol{h}^{r}_{e_i}] \tag{3}$$

where $||$ represents the concatenate operation, $\boldsymbol{h}^{(1)}_{e_i}$ represents the fused embedding of e_i in layer 1.

Feature Aggregation

Each entity aggregates the fused features from its neighbors by a graph convolution operation. The aggregation can be performed efficiently by matrix operations. Let $\boldsymbol{H}^{(l)}$ represents the features of entities in layer l, $\boldsymbol{A}$ is the adjacency matrix with self-loops of TKGs, $\boldsymbol{D}$ is the diagonal node degree matrix, $\boldsymbol{H}^{(l+1)}$ is the embeddings of entities in layer $l+1$, The matrix operation of feature aggregation to get $\boldsymbol{H}^{(l+1)}$ is shown as follows:

$$\boldsymbol{H}^{(l+1)} = \sigma(\boldsymbol{D}^{-1}\boldsymbol{A}\boldsymbol{H}^{(l)}) \tag{4}$$

where $\sigma(\cdot)$ is a non-linear activation function like the ReLU.

Global-aware Embedding

The GNN can capture multi-hop neighborhood information. We concatenate the entity embedding from different layers to get the global-aware embedding. The final embedding of entity e_i is obtained as follows:

$$\boldsymbol{h}^{mul}_{e_i} = [\boldsymbol{h}^{(1)}_{e_i} || \boldsymbol{h}^{(2)}_{e_i} || ... || \boldsymbol{h}^{(L)}_{e_i}] \tag{5}$$

where $\boldsymbol{h}^{(1)}_{e_i}$ is the embedding of entity e_i in layer 1, $\boldsymbol{h}^{(L)}_{e_i}$ is the embedding of entity e_i in layer L, $||$ represents the concatenate operation.

4.2 Entity Alignment

The model first learns the embeddings of entities by training with the alignment seeds, then calculates the embedding similarity matrix of the entities and combines the time similarity matrix to predict the new alignment entities. Iterative learning is performed by adding new alignment entities to the training set to find more alignment entities.

Training

We use the following triplet loss function to train the model:

$$L = \sum_{(e_i,e_j)\in P} max(dist(\boldsymbol{e}_i, \boldsymbol{e}_j) - dist(\boldsymbol{e}'_i, \boldsymbol{e}'_j) + \lambda,\ 0) \quad (6)$$

where (e_i, e_j) represent the positive pair, (e'_i, e'_j) represent the negative pair by randomly replacing one of (e_i, e_j). λ represents the margin hyperparameter, $dist(\cdot)$ represent the Manhattan distance:

$$dist(\boldsymbol{e}_i, \boldsymbol{e}_j) = \|\boldsymbol{h}^{mul}_{\boldsymbol{e}_i} - \boldsymbol{h}^{mul}_{\boldsymbol{e}_j}\|_1 \quad (7)$$

Time Dictionary Construction

There are two types of temporal information in the datasets, one is the time point τ and another is time interval $[\tau_s, \tau_e]$, where τ_s denote the start time and τ_e denote the end time. We construct the time dictionary Dic of the entities by using the quadruples. If the quadruple is (h, r, t, τ), then append τ to the dictionary Dic_h of h and dictionary Dic_t of t. If the quadruple is $(h, r, t, [\tau_s, \tau_e])$, then append τ_s and τ_e to the dictionary Dic_h and Dic_t .

Temporal Information Matching Mechanism

We use the time dictionary mentioned above to obtain the time similarity matrix by calculating the matching degree of temporal information in the time dictionary of each entity. Specifically, for two entities $e_i \in G_1$ and $e_j \in G_2$, the time dictionaries of them are $Dic_{e_i} = [\tau^1_{e_i}, \tau^2_{e_i}, ..., \tau^m_{e_i}]$ and $Dic_{e_j} = [\tau^1_{e_j}, \tau^2_{e_j}, ..., \tau^n_{e_j}]$, where m and n represent the numbers of temporal information in Dic_{e_i} and Dic_{e_j} respectively. Let c denote the number of same items in Dic_{e_i} and Dic_{e_j}, the time similarity $s^t_{e_ie_j}$ of e_i and e_j is calculated as follows:

$$s^t_{e_ie_j} = \frac{c \times 2}{m + n} \quad (8)$$

For each entity in the two TKGs, by calculating the matching degree of temporal information from each other, we get the time similarity matrix of entities in different TKGs.

Prediction

We adopt a similar method with RREA (Mao et al., 2020b) which uses the Cross-domain Similarity Local Scaling (CSLS) (Lample et al., 2018) to predict the alignment entities. Since TKGs have additional temporal information, which is used to calculate the time similarity of entities, we predict alignment entities by combining the embedding similarity and time similarity of entities. The final similarity of entities can be expressed as follows:

$$s_{e_ie_j} = (1 - \alpha) \times s^e_{e_ie_j} + \alpha \times s^t_{e_ie_j} \quad (9)$$

where $s^e_{e_ie_j}$ is the embedding similarity of e_i and e_j, it can be simply calculated by the inner product of the embeddings of e_i and e_j. $s^t_{e_ie_j}$ is the time similarity of e_i and e_j, α is a balance hyperparameter.

Iterating Learning

We adopt the bi-directional iterative strategy proposed by MRAEA (Mao et al., 2020a). The difference from this method is that we also combine the temporal information to calculate the similarity between entities in addition to using entity embedding. If the entities e_i and e_j are mutually nearest neighbors, then the pair (e_i, e_j) is considered new alignment entities and will be added into the training set of the next iteration. Since an iteration contains many epochs of training (1200 in this paper), the pseudo-supervised seeds generated after one iteration help improve the model's performance.

4.3 Unsupervised Alignment Seeds Generation

In the real world, alignment seeds are often difficult to obtain due to the high cost of manual annotations. Assuming that the entities with the same temporal information in the two TKGs are the most similar, we propose a strategy to generate the alignment seeds using the temporal information of entities. The generated alignment seeds (e_i, e_j) should satisfy the following two criteria: (1) For e_i, there is and only one e_j that exactly matches the temporal information of e_i. (2) They are mutually and uniquely nearest neighbors of each other. The detailed alignment seeds generation process is described in Algorithm 1.

5 Experiments

5.1 Data sets

We experiment on four datasets for entity alignment between temporal knowledge graphs in English.

(1) **DICEWS** (Xu et al., 2021): This dataset is built from ICEWS05-15 (García-Durán et al., 2018) which contains events during 2005 to 2015. There are **DICEWS-1K** with 1K alignment seeds and **DICEWS-200** with 200 in it.

(2) **YAGO-WIKI50K** (Xu et al., 2021): This dataset extracts the top 50,000 entities according to

Dataset	$\|E_1\|$	$\|E_2\|$	$\|R_1\|$	$\|R_2\|$	$\|T\|$	$\|Q_1\|$	$\|Q_2\|$	$\|P\|$	$\|S\|$
DICEWS-1K	9,517	9,537	247	246	4,017	307,552	307,553	7,566	1,000
DICEWS-200	9,517	9,537	247	246	4,017	307,552	307,553	8,366	200
YAGO-WIKI50K-5K	49,629	49,222	11	30	245	221,050	317,814	44,172	5,000
YAGO-WIKI50K-1K	49,629	49,222	11	30	245	221,050	317,814	48,172	1,000

Table 1: Statistics of YOGO-WIKI50K and DICEWS. $|\cdot|$ represents the numbers.

Algorithm 1 Algorithm for alignment seeds generation

Input: time similarity matrix of entities $\boldsymbol{S}^t$.
Output: Alignment seeds set AS.

Let $AS = \{\}, index = \{\}$.
for $i \in \boldsymbol{S}^t[i]$ **do**
  if $(len(argmax(\boldsymbol{S}^t[i])) == 1)$ & $(max(\boldsymbol{S}^t[i]) == 1)$ **then**
    $index \leftarrow (i, argmax(\boldsymbol{S}^t[i])) \cup index$
  end if
end for
for $(a, b) \in index$ **do**
  if $(\boldsymbol{S}^t[a][b] == \boldsymbol{S}^t[b][a])$ **then**
    $AS \leftarrow (a, b) \cup AS$
  end if
end for
return AS

their frequencies in Wikidata from tkbc (Lacroix et al., 2020) which has temporal information, and then link the Wikidata entities to their equivalent YAGO entities by the mappings of Wikidata QIDs to YAGO instances [1]. There are two subsets **YAGO-WIKI50K-5K** and **YAGO-WIKI50K-1K** in the datasets. The only difference between the subsets is that one has 5K alignment seeds, and the other has 1K.

The statistics of these datasets are listed in Table 1.

5.2 Baselines

We compare our method with the following three groups of methods:

(1) **Translation-based Model**: MTransE (Chen et al., 2017), JAPE (Sun et al., 2017), BootEA (Sun et al., 2018). These methods are strong baseline based on TransE.

(2) **GNN-based Model**: GCN-Align (Wang et al., 2018), MRAEA (Mao et al., 2020a), RREA (Mao et al., 2020b). These are SOTA GNN-based models for EA between KGs without temporal information.

(3) **Time-aware Model**: TEA-GNN (Xu et al., 2021), TREA (Xu et al., 2022). These are SOTA GNN-based models for EA between TKGs.

The main results of these models reported in the paper are from TREA (Xu et al., 2022) except BootEA (Sun et al., 2018) and RREA (Mao et al., 2020b), because the source code of the two models uses an iterative strategy, while TREA uses a non-iterative version. The experiments of these two models are implemented based on their publicly available resource codes. All experiments are conducted on a workstation with a GeForce RTX 3090 GPU and an AMD EPYC 7502 32-Core Processor CPU and 128GB memory. The code and datasets are available online [2].

5.3 Experimental Setup

For a fair comparison, we use the fixed training set and validation set provided by TEA-GNN (Xu et al., 2021). TREA (Xu et al., 2022) uses the same datasets, but the source code is not yet provided. We also provide a non-iterative model STEA* to compare with the SOTA models.

Following convention, Hits@k ($k = 1, 10$) and mean reciprocal rank (MRR) are used as evaluation metrics. Hits@k reports the proportion of correct alignment pairs to the top k potential entities. In particular, Hits@1 represents accuracy. MRR is calculated as the average of the reciprocal ranks of the results. The higher the Hits@k and MRR, the better the performance.

For all datasets, we use a same default setting: the dimensionality for embedding $d = 100$; depth of GNN layers $L = 2$; margin $\gamma = 3$; dropout_rate $dr = 0.3$; balance factor $\alpha = 0.3$; iterations $k = 5$; The number of epochs is 1200 and RMSprop is adopted to optimize the model with learning rate set to 0.005. The reported performance is the average of five independent training runs.

[1] http://resources.mpi-inf.mpg.de/yago-naga/yago3.1/

[2] https://github.com/lcai2/STEA

Models	DICEWS-1K			DICEWS-200			YAGO-WIKI50K-5K			YAGO-WIKI50K-1K		
	MRR	Hits@1	Hits@10	MRR	Hits@1	Hits@10	MRR	Hits@1	Hits@10	MRR	Hits@1	Hits@10
MTransE	.150	.101	.241	.104	.067	.175	.332	.242	.477	.033	.012	.067
JAPE	.198	.144	.298	.138	.098	.210	.345	.271	.488	.157	.101	.262
AlignE	.593	.508	.751	.303	.222	.457	.800	.756	.883	.618	.565	.714
BootEA	.670	.598	.796	.614	.546	.737	-	-	-	-	-	-
GCN-Align	.291	.204	.466	.231	.165	.363	.581	.512	.711	.279	.217	.398
MRAEA	.745	.675	.870	.564	.476	.733	.848	.806	.913	.685	.623	.801
RREA	.840	.795	.917	.823	.773	.911	.913	.887	.955	.870	.836	.929
TEA-GNN	.911	.887	.947	.902	.876	.941	.909	.879	.961	.775	.723	.871
TREA	.933	.914	.966	.927	.910	.960	.958	.940	.989	.885	.840	.937
STEA*	.941	.928	.960	.941	.927	.961	.954	.935	.986	.916	.887	.966
STEA	**.954**	**.945**	**.967**	**.954**	**.943**	**.968**	**.974**	**.961**	**.992**	**.962**	**.943**	**.989**

Table 2: Experimental results of EA on DICEWS and YOGO-WIKI50K. AlignE is the non-iterative version of BootEA. - means the results are not obtained. STEA* represents the mothed without iteration. The best results are written in bold. Underline indicate the sub-optimal results.

5.4 Results and Analysis

Main Results

Table 2 show the main results of the experiment. It can be seen that STEA with iteration achieves the best results on all datasets compared to all methods on all the evaluation metrics. Especially compared with the SOTA method TREA for EA between TKGs, STEA exceeds by at least 2.2% on Hits@1. The improvement scores of Hits@1 are 3.4%, 3.6%, 2.2%, 12.3%, respectively. Without iteration, STEA* still outperforms TREA on most datasets and metrics, and the gaps in other experimental results are very tiny, at most no more than 0.06%. It indicates the effectiveness of combining the embedding similarity and time similarity to predict alignment entities. STEA achieves the best results and proves that iterative strategies could improve performance. Our method achieves remarkable performance using the simple GNN model combined with a temporal information matching mechanism.

Ablation Study

The STEA consists of three key components: (1) Relational Feature Fusion (RFF); (2) Global-Aware Representation (GAR); (3) Time Similarity Matrix (TSM). We remove these components from STEA individually to demonstrate their effectiveness.

Table 3 show the ablation study of the experiment. It can be seen from the table that the performance of STEA is degraded after removing a module. Each module contributes differently to the two datasets due to their different characteristics. Without relational feature fusion, EA performance drops slightly on YAGO-WIKI50K (STEA vs. STEA-RFF) because there are few relations in the datasets and decreases more on DICEWS with more relations. This shows the importance of relational features in heterogeneous TKGs with multiple relationships. STEA-GAR Remove concatenating the representations of different layers, Hits@1 declines by about 3% on DICEWS and 5% on YAGO-WIKI50K, validating the effectiveness of gathering the embedding of multi-hop neighborhood. Compared with STEA, without using the temporal information, STEA-TSM does hurt STEA in all metrics, indicating the importance of using the temporal matching mechanism. The performance drops more on DICEW than YAGO-WIKI50K, suggesting that the former is more relevant to temporal information.

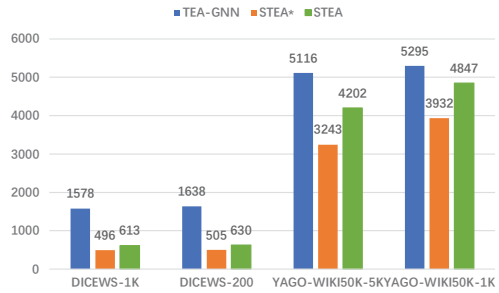

Figure 3: Time costs of methods (seconds).

Complexity analysis

Figure3 reports the overall time costs of our methods with TEA-GNN (Xu et al., 2021) on each dataset, including data loading, pre-processing, training, and evaluating. The results are obtained by directly running the source code provided by the author. As shown in Figure 3, the time cost of STEA* is the lowest. The time cost of STEA is higher than STEA* since using the iteration, but still less than TEA-GNN, which indicates the efficiency of our model.

Models	DICEWS-1K			DICEWS-200			YAGO-WIKI50K-5K			YAGO-WIKI50K-1K		
	MRR	Hits@1	Hits@10	MRR	Hits@1	Hits@10	MRR	Hits@1	Hits@10	MRR	Hits@1	Hits@10
STEA	**.954**	**.945**	**.967**	**.954**	**.943**	**.968**	**.974**	**.961**	**.992**	**.962**	**.943**	**.989**
STEA-RFF	.923	.910	.944	.924	.910	.946	.967	.953	.987	.948	.925	.981
STEA-GAR	.932	.916	.952	.934	.918	.956	.945	.918	.984	.927	.890	.977
STEA-TSM	.792	.741	.888	.742	.674	.863	.932	.910	.967	.881	.843	.939

Table 3: Ablation study of STEA on all datasets

Model	Number of Parameters
MTransE	$O((\|E\| + \|R\|) \times d)$
JAPE	$O((\|E\| + \|R\|) \times d)$
BootEA	$O((\|E\| + \|R\|) \times d)$
GCN-Align	$O(\|E\| \times d + d \times d \times L)$
MRAEA	$O((\|E\| + \|R\| \times 2) \times d)$
RREA	$O((\|E\| + \|R\| \times 2) \times d + 3 \times d \times L)$
TEA-GNN	$O((\|E\| + \|R\| \times 2 + \|T\|) \times d + 3 \times d \times L \times 2)$
TREA	$O((\|E\| + \|R\| \times 2 + \|T\|) \times d + 4 \times d \times L \times 2)$
STEA	$O((\|E\| + \|R\|) \times d)$

Table 4: The total number of parameters in the training phase of the models compared in the paper.

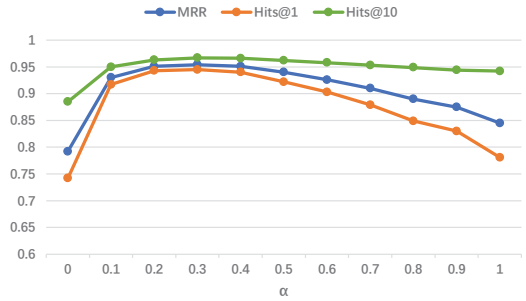

Figure 4: Experimental results with different balance factor α on DICEWS-1K.

The total number of parameters in the training phase of our model is equal to $(|E| + |R|) \times d$, which is less than the existing SOTA model. The number of parameters of each model compared in our experiments is shown in Table 4, where $|E|$ represents the total entities of the two TKGs, $|R|$ represents the total relations of the two TKGs, d represents the embedding dimension of the parameters, L represents the layers of GNN, $|T|$ represents the total temporal information of the two TKGs.

Hyper-parameter Analysis

In order to investigate the effects of hyper-parameters on the performance of STEA, we conduct the following experiments on DICEWS-1K: (1) The performance with different balance factors α. (2) The entity alignment results in different layers l. (3) The Hits@1 values with different embedding dimensions d.

The balance factor α is used to balance the weight of the entity's embedding similarity matrix and the time similarity matrix in our approach. We set it in range $0 \sim 1$ with interval 0.1. As shown in Figure 4, the performance of EA only using the embedded similarity matrix ($\alpha = 0$) is lower than that of only using the time similarity matrix ($\alpha = 1$). It shows that the temporal information in the dataset is highly matched. The model's performance is optimal when $\alpha = 0.3$ and similar in the range of $0.1 \sim 0.5$.

For more analysis about hyper-parameters layers l and dimension d, please refer to Appendix A.

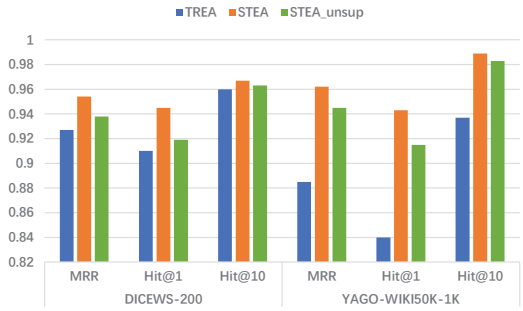

Figure 5: Experimental Results with unsupervised alignment seeds.

Unsupervised Experimental Results

We adopt the approach mentioned in section 4.3 to obtain the alignment seeds and conduct experiments on DICEWS-200 and YAGO-WIKI50K-1K to compare against the supervised method TREA (Xu et al., 2022) and STEA. The experimental results in Figue 5 show that our unsupervised method STEA$_{unsup}$ still outperforms TREA. The performance of STEA$_{unsup}$ on the two datasets is slightly lower than our supervised method STEA, indicating that only using the temporal matching degree of entities to obtain alignment seeds on the KGs may introduce noise.

Case Study

In order to study the strengths and weaknesses of our approach compared to previous works, we illustrate some cases that STEA predicts from the test sets of YOGO-WIKI50K-1K and DICEWS-1K compared to TEA-GNN. Details are described in Appendix B.

6 Conclusion

This paper proposes a simple GNN model combined with a temporal information matching mechanism for EA between TKGs, which achieves significant performance with less time and fewer parameters.

By assuming that entities with the same temporal information have high similarity, we propose a method to generate the alignment seeds by temporal information. The unsupervised method outperforms the previous models on two datasets and is competitive with our supervised method.

The paper does not discuss the cases where the KGs might employ different schemas for representing time. We will further explore this in future work.

7 Acknowledgements

This work has been supported by Pudong New Area Science Technology Development Fund.

References

Vito Walter Anelli, Tommaso Di Noia, Eugenio Di Sciascio, Antonio Ferrara, and Alberto Carlo Maria Mancino. 2021. Sparse feature factorization for recommender systems with knowledge graphs. In *RecSys '21: Fifteenth ACM Conference on Recommender Systems, Amsterdam, The Netherlands, 27 September 2021 - 1 October 2021*, pages 154–165. ACM.

Antoine Bordes, Nicolas Usunier, Alberto García-Durán, Jason Weston, and Oksana Yakhnenko. 2013. Translating embeddings for modeling multi-relational data. In *Advances in Neural Information Processing Systems 26: 27th Annual Conference on Neural Information Processing Systems 2013. Proceedings of a meeting held December 5-8, 2013, Lake Tahoe, Nevada, United States*, pages 2787–2795.

Muhao Chen, Yingtao Tian, Mohan Yang, and Carlo Zaniolo. 2017. Multilingual knowledge graph embeddings for cross-lingual knowledge alignment. In *Proceedings of the Twenty-Sixth International Joint Conference on Artificial Intelligence, IJCAI 2017, Melbourne, Australia, August 19-25, 2017*, pages 1511–1517. ijcai.org.

Alberto García-Durán, Sebastijan Dumancic, and Mathias Niepert. 2018. Learning sequence encoders for temporal knowledge graph completion. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing, Brussels, Belgium, October 31 - November 4, 2018*, pages 4816–4821. Association for Computational Linguistics.

Fuzhen He, Zhixu Li, Qiang Yang, An Liu, Guanfeng Liu, Pengpeng Zhao, Lei Zhao, Min Zhang, and Zhigang Chen. 2019. Unsupervised entity alignment using attribute triples and relation triples. In *Database Systems for Advanced Applications - 24th International Conference, DASFAA 2019, Chiang Mai, Thailand, April 22-25, 2019, Proceedings, Part I*, volume 11446 of *Lecture Notes in Computer Science*, pages 367–382. Springer.

Johannes Hoffart, Fabian M. Suchanek, Klaus Berberich, and Gerhard Weikum. 2013. YAGO2: A spatially and temporally enhanced knowledge base from wikipedia. *Artif. Intell.*, 194:28–61.

Wei Hu, Qingheng Zhang, Zequn Sun, and Jiacheng Huang. 2019. Multike: a multi-view knowledge graph embedding framework for entity alignment. In *Proceedings of the 14th International Workshop on Ontology Matching co-located with the 18th International Semantic Web Conference (ISWC 2019), Auckland, New Zealand, October 26, 2019*, volume 2536 of *CEUR Workshop Proceedings*, pages 189–190. CEUR-WS.org.

Shaoxiong Ji, Shirui Pan, Erik Cambria, Pekka Marttinen, and Philip S. Yu. 2022. A survey on knowledge graphs: Representation, acquisition, and applications. *IEEE Trans. Neural Networks Learn. Syst.*, 33(2):494–514.

Thomas N. Kipf and Max Welling. 2017. Semi-supervised classification with graph convolutional networks. In *5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Conference Track Proceedings*. OpenReview.net.

Timothée Lacroix, Guillaume Obozinski, and Nicolas Usunier. 2020. Tensor decompositions for temporal knowledge base completion. In *8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, April 26-30, 2020*. OpenReview.net.

Guillaume Lample, Alexis Conneau, Marc'Aurelio Ranzato, Ludovic Denoyer, and Hervé Jégou. 2018. Word translation without parallel data. In *6th International Conference on Learning Representations, ICLR 2018, Vancouver, BC, Canada, April 30 - May 3, 2018, Conference Track Proceedings*. OpenReview.net.

Mengqi Lin, Qing Liao, Yan Jia, and Ye Wang. 2021. Survey on knowledge graph embedding based on hyperbolic geometry. In *Sixth IEEE International Conference on Data Science in Cyberspace, DSC 2021, Shenzhen, China, October 9-11, 2021*, pages 152–158. IEEE.

Fangyu Liu, Muhao Chen, Dan Roth, and Nigel Collier. 2021. Visual pivoting for (unsupervised) entity alignment. In *Thirty-Fifth AAAI Conference on Artificial Intelligence, AAAI 2021, Thirty-Third Conference on Innovative Applications of Artificial Intelligence, IAAI 2021, The Eleventh Symposium on Educational*

Advances in Artificial Intelligence, EAAI 2021, Virtual Event, February 2-9, 2021, pages 4257–4266. AAAI Press.

Xin Mao, Meirong Ma, Hao Yuan, Jianchao Zhu, Zongyu Wang, Rui Xie, Wei Wu, and Man Lan. 2022. An effective and efficient entity alignment decoding algorithm via third-order tensor isomorphism. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), ACL 2022, Dublin, Ireland, May 22-27, 2022*, pages 5888–5898. Association for Computational Linguistics.

Xin Mao, Wenting Wang, Yuanbin Wu, and Man Lan. 2021. From alignment to assignment: Frustratingly simple unsupervised entity alignment. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing, EMNLP 2021, Virtual Event / Punta Cana, Dominican Republic, 7-11 November, 2021*, pages 2843–2853. Association for Computational Linguistics.

Xin Mao, Wenting Wang, Huimin Xu, Man Lan, and Yuanbin Wu. 2020a. MRAEA: an efficient and robust entity alignment approach for cross-lingual knowledge graph. In *WSDM '20: The Thirteenth ACM International Conference on Web Search and Data Mining, Houston, TX, USA, February 3-7, 2020*, pages 420–428. ACM.

Xin Mao, Wenting Wang, Huimin Xu, Yuanbin Wu, and Man Lan. 2020b. Relational reflection entity alignment. In *CIKM '20: The 29th ACM International Conference on Information and Knowledge Management, Virtual Event, Ireland, October 19-23, 2020*, pages 1095–1104. ACM.

Zhiyuan Qi, Ziheng Zhang, Jiaoyan Chen, Xi Chen, Yuejia Xiang, Ningyu Zhang, and Yefeng Zheng. 2021. Unsupervised knowledge graph alignment by probabilistic reasoning and semantic embedding. In *Proceedings of the Thirtieth International Joint Conference on Artificial Intelligence, IJCAI 2021, Virtual Event / Montreal, Canada, 19-27 August 2021*, pages 2019–2025. ijcai.org.

Yunqi Qiu, Yuanzhuo Wang, Xiaolong Jin, and Kun Zhang. 2020. Stepwise reasoning for multi-relation question answering over knowledge graph with weak supervision. In *WSDM '20: The Thirteenth ACM International Conference on Web Search and Data Mining, Houston, TX, USA, February 3-7, 2020*, pages 474–482. ACM.

Apoorv Saxena, Aditay Tripathi, and Partha P. Talukdar. 2020. Improving multi-hop question answering over knowledge graphs using knowledge base embeddings. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, ACL 2020, Online, July 5-10, 2020*, pages 4498–4507. Association for Computational Linguistics.

Zequn Sun, Wei Hu, and Chengkai Li. 2017. Cross-lingual entity alignment via joint attribute-preserving embedding. In *The Semantic Web - ISWC 2017 - 16th International Semantic Web Conference, Vienna, Austria, October 21-25, 2017, Proceedings, Part I*, volume 10587 of *Lecture Notes in Computer Science*, pages 628–644. Springer.

Zequn Sun, Wei Hu, Qingheng Zhang, and Yuzhong Qu. 2018. Bootstrapping entity alignment with knowledge graph embedding. In *Proceedings of the Twenty-Seventh International Joint Conference on Artificial Intelligence, IJCAI 2018, July 13-19, 2018, Stockholm, Sweden*, pages 4396–4402. ijcai.org.

Zequn Sun, Qingheng Zhang, Wei Hu, Chengming Wang, Muhao Chen, Farahnaz Akrami, and Chengkai Li. 2020. A benchmarking study of embedding-based entity alignment for knowledge graphs. *Proc. VLDB Endow.*, 13(11):2326–2340.

Denny Vrandecic and Markus Krötzsch. 2014. Wikidata: a free collaborative knowledgebase. *Commun. ACM*, 57(10):78–85.

Zhichun Wang, Qingsong Lv, Xiaohan Lan, and Yu Zhang. 2018. Cross-lingual knowledge graph alignment via graph convolutional networks. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing, Brussels, Belgium, October 31 - November 4, 2018*, pages 349–357. Association for Computational Linguistics.

Chengjin Xu, Fenglong Su, and Jens Lehmann. 2021. Time-aware graph neural network for entity alignment between temporal knowledge graphs. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing, EMNLP 2021, Virtual Event / Punta Cana, Dominican Republic, 7-11 November, 2021*, pages 8999–9010. Association for Computational Linguistics.

Chengjin Xu, Fenglong Su, Bo Xiong, and Jens Lehmann. 2022. Time-aware entity alignment using temporal relational attention. In *WWW '22: The ACM Web Conference 2022, Virtual Event, Lyon, France, April 25 - 29, 2022*, pages 788–797. ACM.

Hsiu-Wei Yang, Yanyan Zou, Peng Shi, Wei Lu, Jimmy Lin, and Xu Sun. 2019. Aligning cross-lingual entities with multi-aspect information. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing, EMNLP-IJCNLP 2019, Hong Kong, China, November 3-7, 2019*, pages 4430–4440. Association for Computational Linguistics.

Kun Zhou, Wayne Xin Zhao, Shuqing Bian, Yuanhang Zhou, Ji-Rong Wen, and Jingsong Yu. 2020. Improving conversational recommender systems via knowledge graph based semantic fusion. In *KDD '20: The 26th ACM SIGKDD Conference on Knowledge Discovery and Data Mining, Virtual Event, CA, USA, August 23-27, 2020*, pages 1006–1014. ACM.

	Entities	Similar Quadruples Involving the Entities
Entity to be Aligned (in YAGO)	<Neymar>	*<Neymar>,<playsFor>,<Brazil_national_under_17_football_team>,[2009,inf]* *<Neymar>,<playsFor>,<Brazil_national_under_20_football_team>,[2011,inf]* *<Neymar>,<playsFor>,<Brazil_national_under_23_football_team>,[2012,inf]* ...
Prediction in TEA-GNN (in WIKI)	Ademilson (Q2756361)	*Ademilson(Q2756361),member of sports team,Brazil_national_under_17_football_team(Q2402747),[2011,2011]* *Ademilson(Q2756361),member of sports team, Brazil_national_under_20_football_team(Q2308235),[2012,2013]* *Ademilson(Q2756361),member of sports team, Brazil_national_under_23_football_team(Q899189),[2014,inf]* ...
Prediction in STEA (in WIKI)	Neymar (Q142794)	*Neymar (Q142794),member of sports team,Brazil_national_under_17_football_team(Q2402747),[2009,2009]* *Neymar (Q142794),member of sports team, Brazil_national_under_20_football_team(Q2308235),[2011,2011]* *Neymar (Q142794),member of sports team, Brazil_national_under_23_football_team(Q899189),[2012,2012]* ...

Table 5: The case of different predictions in STEA and TEA-GNN.

	Entities	Similar Quadruples Involving the Entities
Entity to be Aligned (in $ICEWS_1$)	Bishop (India)	*Head of Government (India), Make statement, Bishop (India), 462*
Prediction in STEA (in $ICEWS_2$)	Electoral Alliance (India)	*Head of Government (India),Make statement,Electoral Alliance (India),462*
Entity to be Aligned (in YAGO)	<William_Travilla>	*<William_Travilla>,<isMarriedTo>,<Dona_Drake>,[1944,1989]*
Prediction in STEA (in WIKI)	Dona_Drake (Q3035962)	*Dona_Drake(Q3035962), spouse, William_Travilla(Q945402),[1944,1989]*

Table 6: The bad case of predictions in STEA.

A Analysis of Hyper-parameters

Figure 6 presents the entity alignment results with different layers of STEA on DICEWS-1K. It can be seen that the performance gap of the models with different layers is tiny, and STEA with 2 layers achieves the best performance. When stacking more layers, the performance does not improve. Increasing the layers only introduces more computation, not better performance.

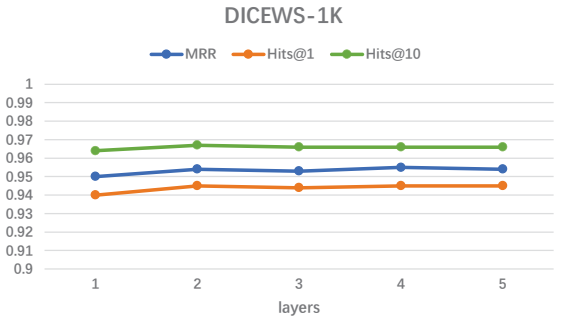

Figure 6: Entity alignment results with different layers of STEA on DICEWS-1K.

Figure 7 reports the Hits@1 performances with embedding dimension d from 50 to 400. The performance of STEA is not greatly affected by the dimension. As the dimension increases, the performance gain is small. Good performance is achieved even when dimension is 50. We chose 100 as the final dimension, which has high performance and small space.

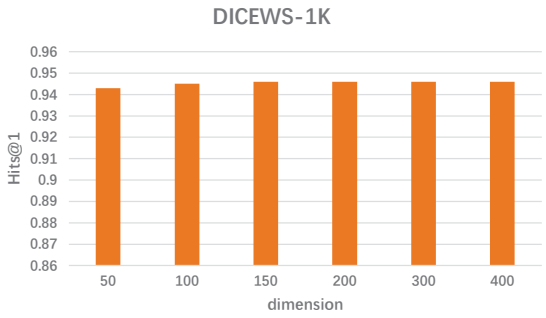

Figure 7: @1 performances of different dimensions on DICEWS-1K.

B Case Study

We illustrate some predictions of the STEA to analyze its advantages and problems. Table 5 shows an example that STEA gives a prediction different from TEA-GNN in YOGO-WIKI50K-1K. It can be seen from the case that TEA-GNN wrongly aligns *Neymar* and *Ademilson* from G_1 and G_2, because these two entities have similar connected links and temporal information in TKG1 and TKG2. Some links respective to these two entities have the same linked entities and relation types but different temporal information. The TEA-GNN wrongly identifies them as alignment entities by the time-aware attention mechanism. The temporal information matching mechanism of STEA can correctly distinguish these two entities since their timestamps are not equal.

Due to the temporal information matching mechanism, STEA will wrongly align entities with the same timestamps. There are some cases in Table 6. In the first case, entity *Bishop (India)* from G_1 and entity *Electoral Alliance (India)* from G_2 of DICEWS-1K are predicted as the alignment pairs because these two entities have the same links and temporal information in TKG1 and TKG2. STEA regards them as alignment entities due to their high similarity of structure and timestamps. In the second case, STEA mistakenly regards the entity <*William_Travilla*> in YAGO and the entity *Dona_Drake (Q3035962)* in WIKI as the same person since these two-person are a couple and get married on the same day. The model will misidentify these entities with symmetric relationships as alignment entities. TEA-GNN also has the same problem. It will be addressed in future work.

DCT-Centered Temporal Relation Extraction

Liang Wang, Peifeng Li * and **Sheng Xu**
School of Computer Science and Technology Soochow University, Jiangsu, China
docy@vip.qq.com, pfli@suda.edu.cn, sxu@stu.suda.edu.cn

Abstract

Most previous studies on temporal relation extraction focus on extracting temporal relations among events and suffer from the issue of different forms of events, timexes and Document Creation Time (DCT) in a document. Moreover, DCT can act as a hub to semantically connect the other events and timexes. Unfortunately, previous work cannot fully use such critical and helpful information. To address the above issues, we propose a unified DCT-centered Temporal Relation Extraction model DTRE to identify temporal relations among events, timexes and DCT. Specifically, we first introduce sentence-style DCT to unify the expressions of event, timex and DCT. Then, we apply a DCT-aware graph to obtain their contextual structural representations. Furthermore, we propose a DCT-anchoring multi-task framework to jointly predict three tasks of temporal relation extraction in a batch. Finally, we provide a DCT-guided global inference to further enhance the global consistency among different relations. Experimental results on three popular datasets TBD, TDD-man and TDD-Auto show that our DTRE outperforms several SOTA baselines on E-E, E-T and E-D significantly.

1 Introduction

Temporal relation extraction focuses on the occurrence order (TLINK) of event mentions, time expressions (timexes) and Document Creation Time (DCT). Most previous studies only focus on the event-centered tasks and consider three TLINKs: event-event (E-E), event-timex (E-T), and event-DCT (E-D). As a crucial component of relation extraction, temporal relation extraction can help many downstream NLP tasks, such as question answering (Ning et al., 2020), summarization (Noh et al., 2020) and timeline construction (Li et al., 2021).

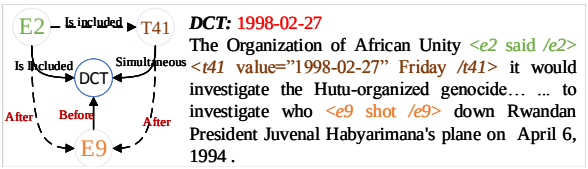

Figure 1: Examples of temporal relations among events, timex and DCT.

Most previous studies (Mathur et al., 2021; Liu et al., 2021) only focus on the single E-E task, ignoring the other E-T and E-D tasks. The main barricade is the hardness of combining the E-T or E-D task with the E-E task due to their different expression forms. Since most event mentions [1] are sentences or clauses, their rich information is helpful for a neural network model to identify the specific relation between two event mentions. However, timex and DCT (especially DCT) are only word-level or phrase-level tokens, and the information imbalance between events and timexes/DCT will lead to the issue that it is difficult for a neural network model to extract their correct temporal relation by a unified model.

As shown in Figure 1, identifying the temporal relation between the two long-distance event mentions *e2* and *e9* is challenging, even for humans. However, if we first recognize the *Is Included* and *Before* links of (*e2*, *DCT*) and (*e9*, *DCT*), then the *After* link of (*e2*, *e9*) will be much easier to obtain. Fortunately, identifying the temporal relation between event (or timex) and DCT is relatively simple for humans and pre-trained language models.

Since event, timex and DCT have different forms of expressions, most previous studies (Cheng and Miyao, 2017; Meng and Rumshisky, 2018) only focus on E-E or regarded E-E, E-T and E-D as three independent tasks, as we mentioned above. They often suffer from the issues of data scarcity

*Corresponding author

[1] An event mention refers to a phrase, clause or sentence within which an event is described.

Proceedings of the 29th International Conference on Computational Linguistics, pages 2087–2097
October 12–17, 2022.

and overfitting due to the small size of available datasets. Cheng et al. (2020) propose a model to fuse E-E, E-T and E-D into a unified model by introducing a learnable parameter-based DCT representation. However, they still suffer from two issues: 1) the different expressions of event, timex and DCT harm the information interaction among different tasks; 2) they ignore the importance of DCT to connect the events and timexes in a document.

To address the above issues, we propose a unified **D**CT-centered **T**emporal **R**elation **E**xtraction (**DTRE**) model to discover the temporal relations among events, timexes and DCT in a document. Specifically, we first introduce a sentence-style DCT representation to unify the expressive forms of event, timex, and DCT. Then, we generate a DCT-aware graph to obtain their contextual structural representations. Furthermore, we propose a DCT-anchoring multi-task learning framework to jointly predict three temporal relations (i.e., E-E, E-T and E-D). Finally, we provide a DCT-guided global inference mechanism to benefit from the high accuracies of the E-D and T-D tasks. Experimental results on three popular datasets TBD, TDD-man and TDD-Auto show that our DTRE outperforms several SOTA baselines on all three tasks E-E, E-T and E-D significantly. In summary, our contributions are as follows:

- We introduce a sentence-style DCT representation to unify the expressive forms of events, timexes and DCT;
- We propose a DCT-aware graph to obtain the contextual structural representations;
- We construct a DCT-anchoring multi-task learning framework to jointly predict three different temporal relations in one batch;
- We provide a DCT-guided global inference mechanism to further enhance the global consistency among different relations. In our future work, we will focus on constructing more effective DCT representation.

2 Related Work

TimeBank (Pustejovsky et al., 2003) is an early temporal relation corpus and its extended version TimeBank-Dense (TBD) (Cassidy et al., 2014) adopts a dense annotation scheme in a slide window within adjacent sentences, in which there are mainly five types of TLINKs: E-E, E-T, T-T, E-D and T-D. MATRES (Ning et al., 2018) only contains E-E relations and simplifies the relation labels with a higher inter-annotator agreement. TDDiscourse (Naik et al., 2019) is a discourse-level temporal relation dataset based on TBD, which also focuses on E-E temporal ordering.

Early work on temporal relation extraction (Chambers et al., 2007; Chambers and Jurafsky, 2008; Do et al., 2012; D'Souza and Ng, 2013; Chambers et al., 2014) focused on various linguistic features, including part-of-speech (POS), lexical and morphological features, dependency parsing information, etc. Recent work mainly focuses on the E-E task using neural networks. Liu et al. (2021) and Mathur et al. (2021) show that graph-based neural networks can help relieve informative sparsity between long-distance event mentions in discourse-level temporal relation extraction. Besides, a bunch of efforts focus on incorporating external resources to deal with the limited training resource, such as combining the pre-trained temporal-aware language model (Han et al., 2021), collecting the distantly-supervised examples (Zhao et al., 2021), applying the transfer learning methods to leverage complementary datasets (Ballesteros et al., 2020). Other methods seek to enhance global inference with structural constraints, i.e., relieving the transitive conflicts within temporal graphs (Ning et al., 2017; Han et al., 2019).

Only a few studies consider all three tasks. Early methods were mostly rule-based on event attributes (Chambers et al., 2014; Mirza and Tonelli, 2016), whose performance are deeply harmed by the vague relation, or simply transferred the neural architecture on E-E to E-D and E-T directly via input adjustment (Cheng and Miyao, 2017; Meng and Rumshisky, 2018). Motivated by the success of Multi-Task Deep Neural Network (Liu et al., 2019a) that leverages different supervised learning tasks with the shared contextual embeddings, Cheng et al. (2020) proposed a multi-category learning framework to joint E-E, E-T and E-D. Specially, they introduce a learnable vector to represent DCT as it does not explicitly occur in documents.

3 DTRE: DCT-Centered Temporal Relation Extraction

The temporal relations between event mentions are determined by their occurrence intervals (i.e., start and end points). However, in most real-world texts,

events' intervals are rarely explicitly mentioned, and then external knowledge or common sense is required for temporal reasoning. One important clue is "Had this event happened yet?" or "Is this a future event?", i.e., the E-D task, which is easier and helpful for the E-E and E-T tasks.

To fully exploit the DCT representation and its bridge function to connect events and timexes, we propose a DCT-centered Temporal Relation Extraction model DTRE to discover the temporal relations among events, timexes and DCT. Figure 2 illustrates the overview of our DTRE framework. We first combine the sentence-style DCT representation with the original document as the input of the pre-trained model (BERT or RoBERTa) to obtain the mention embeddings of the different types. Then, we build a DCT-Aware Graph DAG for each document to obtain the contextual structural representations of events, timexes, DCT, etc. Furthermore, we conduct a DCT-anchoring Multi-Task Learning framework DAML to jointly predict the three tasks of temporal relation extraction. Finally, we introduce a DCT-guided Global Inference mechanism DGI to our model according to the high accuracies of the E-D and T-D tasks.

3.1 Input Representation and Encoding

Different from most previous studies that only use event mentions as the input, we input a document with the annotated event mentions, timexes and DCT to our DTRE. Formally, the input is a document D consisting of a sentence set $S = \{s_1, \ldots, s_i, \ldots, s_k\}$, a token set $W = \{w_1, \ldots, w_i, \ldots, w_l\}$, an event set $E = \{e_1, \ldots, e_i, \ldots, e_m\}$, a timex set $T = \{t_1, \ldots, t_i, \ldots, t_n\}$, an entity set $V = \{v_1, \ldots, v_i, \ldots, v_p\}$ and a representation of the document creation time $DCT = \{t_{dct}\}$, where *k, l, m, n, p* represent the total number of sentences, tokens, event mentions, timexes and entity mentions in the document *D*, respectively. In this paper, we do not normalize the timexes and use their original values as the example in Figure 2.

Due to the different expression forms and the different amounts of tokens, it is a challenge to identify those E-T and E-D relations directly. Moreover, DCT does not explicitly occur in the document, making it hard to represent its semantics for temporal relation extraction. However, DCT is an anchor to connect those event mentions or timexes in a document-level temporal ordering graph. Hence, how to represent DCT is critical for our DTRE. To address this issue, we propose a novel DCT representation, which uses a generated sentence to express the token-level DCT. In detail, a sentence that contains DCT is used to represent DCT, and three forms (i.e., CreatN, CreatD, and CreatND) are shown in Table 1.

Type	DCT-indicator Sentence
CreatN	The document is **creating** *now*.
CreatD	The document is **creating** by {*date*}.
CreatND	The document is **creating** *now* by {*date*}.

Table 1: Various forms of DCT representation in sentence-style, where {*date*} is the specific DCT of a document.

The purpose of our sentence-style DCT representation is to make DCT have a similar sentence-based expression as events and timexes. Hence, DTRE can use a unified framework to extract the E-E, E-T and E-D relations simultaneously. Specifically, since most event triggers are verbs, we select "creating" to denote the occurrence of document creation. Moreover, timex has two types, i.e., absolute (e.g., "2022.10.10") and relative time (e.g., "yesterday") that are explicitly annotated in documents. Therefore, we also utilize date like "20221010" extracted from the raw corpus as well as "now" to denote DCT's value.

Finally, we insert the DCT-indicator sentence shown in Table 1 at the beginning of each document to form the input (an example is shown in Figure 2). Hence, all temporal mentions, i.e., events, timexes and DCT, explicitly occur in the input document. In this way we can establish the bridge of information interaction on DCT. In the input, we also use *DCT*, E_i and T_i to represent DCT, the *i*-th event mention, and timex, respectively.

Following previous work (Mathur et al., 2021; Liu et al., 2021) and for fair comparison in our evaluation, we also use BERT and RoBERTa as the pre-trained models to encode the input document and obtain the embeddings of the token set $\boldsymbol{H}_W = \{\boldsymbol{h}_{w1}, \ldots, \boldsymbol{h}_{wi}, \ldots, \boldsymbol{h}_{wl}\}$, the event set $\boldsymbol{H}_E = \{\boldsymbol{h}_{e1}, \ldots, \boldsymbol{h}_{ei}, \ldots, \boldsymbol{h}_{em}\}$, the timex set $\boldsymbol{H}_T = \{\boldsymbol{h}_{t1}, \ldots, \boldsymbol{h}_{ti}, \ldots, \boldsymbol{h}_{tn}\}$, the entity set $\boldsymbol{H}_V = \{\boldsymbol{h}_{v1}, \ldots, \boldsymbol{h}_{vi}, \ldots, \boldsymbol{h}_{vp}\}$, the sentence set $\boldsymbol{H}_S = \{\boldsymbol{h}_{s1}, \ldots, \boldsymbol{h}_{si}, \ldots, \boldsymbol{h}_{sk}\}$, the document $\boldsymbol{H}_D = \{\boldsymbol{h}_{[CLS]} | \boldsymbol{h}_{<s>}\}$, and the DCT set $\boldsymbol{H}_{DCT} = \{\boldsymbol{h}_{dct}\}$.

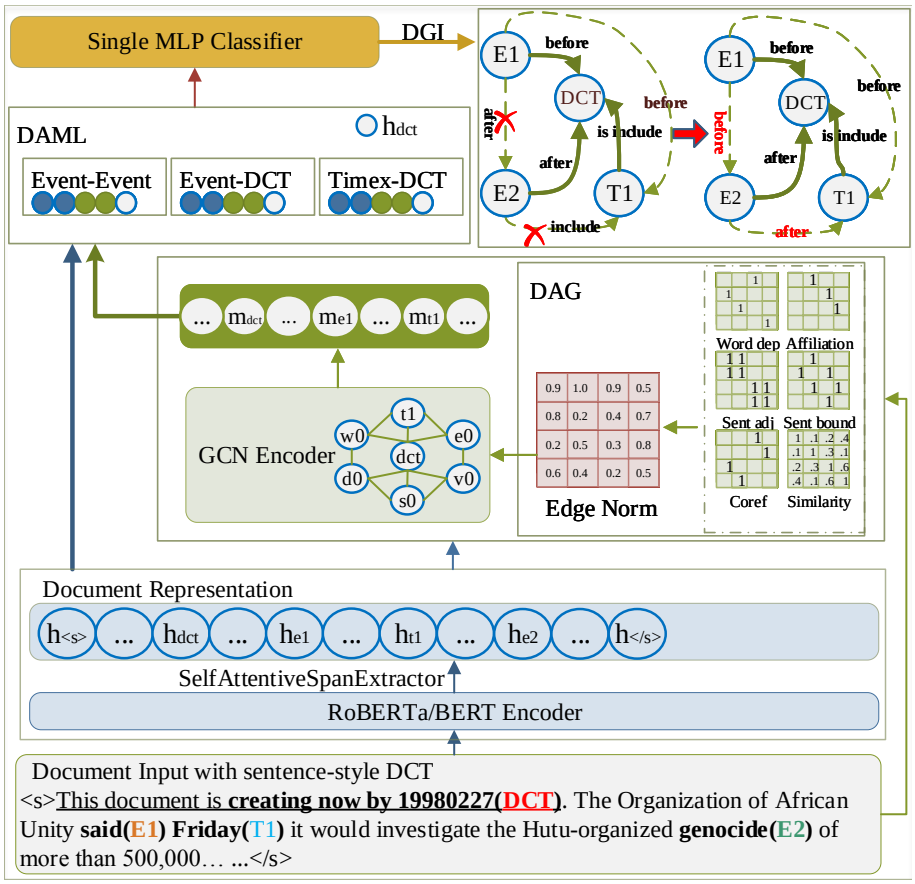

Figure 2: An overview of our proposed DTRE framework.

3.2 DCT-Aware Graph

To capture the structural and interactive information between the different types of temporal mentions, we introduce a DCT-aware graph $DAG = \{N, E\}$ to our model. In this way, we relieve the difference within three tasks to provide rich discourse-level temporal clues.

Our DAG is different from previous GCN models TIMERS (Mathur et al., 2021) and UCGraph (Liu et al., 2021). Specifically, TIMERS constructed three graphs (syntactic, time and rhetorical-aware graph) on events and timexes, while UCGraph built an uncertainty-guided graph on events. Different from their graph, our DAG is a DCT-aware fully-connected graph on events, timexes, and DCT. Moreover, our DAG is simpler than their graphs, because it does not need the edge prediction and optimization.

The node set N can be divided into three levels: token, mention, and discourse, i.e., the token $w_i \in W$, the entity mention $v_j \in V$, the event mention $e_k \in E$, the timex $t_l \in T$, the DCT t_{dct}, the sentence $s_r \in S$, and the document D. We use the embeddings $\boldsymbol{h}_{wi}, \boldsymbol{h}_{ei}, \boldsymbol{h}_{ti}$ and $\boldsymbol{h}_{dct}$ to represent the nodes w_i, e_i, t_i, and DCT, respectively. For nodes consisting of multiple tokens (e.g., entity mentions and sentences), a self-attention mechanism is applied over RoBERTa/BERT embeddings to obtain node representations following (Lee et al., 2017). For the document D, we take $\boldsymbol{h}_{[cls]}$ (BERT) or $\boldsymbol{h}_{<s>}$ (RoBERTa) as its representation. In addition, DAG is composed of six types of edges, i.e., the relations of the affiliation, sentence boundary, word dependency, sentence adjacency, entity coreference, and semantic similarity. We initialize six adjacency matrices ($\boldsymbol{A}^{af}$, $\boldsymbol{A}^{bd}$, $\boldsymbol{A}^{dp}$, $\boldsymbol{A}^{ad}$, $\boldsymbol{A}^{cf}$, $\boldsymbol{A}^{sm}$) to represent them in our graph DAG as follows, where $\boldsymbol{A}^{y} = \{a^{y}_{1,1}, ..., a^{y}_{i,j}, ..., a^{y}_{|N|,|N|}\}$ ($y \in \{af, bd, dp, ad, cf, sm\}$).

Affiliation. A token node w_i connects to its subordinate event/entity/timex node $o_j \in V \cup E \cup T$,and o_j connects to its respective sentence node $s_q \in S$. Each sentence node s_q connects to the document node D. Besides, if the *i*-th node in DAG connects to the *j*-th node, then we set $a^{af}_{i,j} = 1$; otherwise, $a^{af}_{i,j} = 0$, where $a^{af}_{i,j} \in \boldsymbol{A}^{af}$. In this way, we can capture structural information on the word, sentence and document levels.

Sentence Boundary. Entity and event mention pairs that occur in the same sentence usually own a strong relevance. If an event/entity/timex mention pair (o_i, o_j) occur in the same sentence, we set $a^{bd}_{i,j} = 1$; otherwise, $a^{bd}_{i,j} = 0$, where $a^{bd}_{i,j} \in \boldsymbol{A}^{bd}$.

Word Dependency. To encode the syntactic structure, two token nodes w_i and w_j are connected if they share a parent-child relation in a dependency tree, namely we set $a^{dp}_{i,j} = 1$; otherwise, $a^{dp}_{i,j} = 0$, where $a^{dp}_{i,j} \in \boldsymbol{A}^{dp}$.

Sentence Adjacency. If two sentence nodes s_i and s_j are adjacent, then we set $a^{ad}_{i,j} = 1$; otherwise, $a^{ad}_{i,j} = 0$, where $a^{ad}_{i,j} \in \boldsymbol{A}^{ad}$. In this way, the sentence ordering is retained in our DAG.

Entity Coreference. If two entity mention nodes v_i and v_j refer to the same real-world entity, then we set $a^{cf}_{i,j} = 1$; otherwise, $a^{cf}_{i,j} = 0$, where $a^{cf}_{i,j} \in \boldsymbol{A}^{cf}$. This type of edge can help identify the temporal relations between those long-distance event mention pairs.

Semantic Similarity. We compute the cosine similarity $c_{i,j} (0 < c_{i,j} \leq 1)$ between any two nodes n_i and n_j. We set $a^{sm}_{i,j} = c_{i,j}$, where $a^{sm}_{i,j} \in \boldsymbol{A}^{sm}$. In this way, we can capture the rich semantic information among events, timexes and DCT.

The above six matrices are sparse matrices and have the same dimensions. We apply an edge normalization step for the imputation of Graph Convolutional Network (GCN) after generating the above adjacency matrices as follows.

$$\begin{aligned} \boldsymbol{A} = \text{sigmoid}(&\boldsymbol{A}^{af} + \boldsymbol{A}^{bd} + \boldsymbol{A}^{dp} + \\ &\boldsymbol{A}^{ad} + \boldsymbol{A}^{cf} + \boldsymbol{A}^{sm}) \end{aligned} \tag{1}$$

Then the GCN model encodes the original node representations $\boldsymbol{H}_0 = \boldsymbol{H}_W \cup \boldsymbol{H}_V \cup \boldsymbol{H}_E \cup \boldsymbol{H}_T \cup \boldsymbol{H}_S \cup \boldsymbol{H}_D \cup \boldsymbol{H}_{DCT}$ and the adjacency matrix $\boldsymbol{A}$ through G layers as follows.

$$\boldsymbol{H}_I = \text{ReLU}(\boldsymbol{A} \cdot \boldsymbol{H}_{I-1} \cdot \mathbf{W}_I) \tag{2}$$

where $\mathbf{W}_I$ is the weight matrix for the I-th $(0 < I \leq G)$ layer. We denote the GCN out $\boldsymbol{H}_G = \{\boldsymbol{m}_1, \ldots, \boldsymbol{m}_{|N|}\}$.

3.3 DCT-Anchoring Multi-Task Learning

Most previous studies often suffer from data scarcity and overfitting due to the small size of the available datasets and the single E-E task. Cheng et al. (2020) proposed a multi-task learning model SEC that puts three tasks E-E, E-T and E-D into a batch to train, which addresses the issues. However, SEC still suffered from two other issues. The first is the different expressive forms of event, timex and DCT, making it difficult for a unified model to reveal the different types of temporal relations. The second is that they ignore the importance of DCT to connect the events and timexes in a document.

To address the above issues, we propose an efficient DCT-Anchoring Multi-task Learning (DAML) framework to unify the E-E, E-T and E-D tasks, which can enforce the events and timexes to pay more attention to their temporal order related to DCT, considering the highly credible E-D and T-D relations and their transitivity.

Firstly, our sentence-style DCT representation can not only make DCT occur in the document, but also erase the differences in expression among events, timexes and DCT. In this way, DAML can minimize the task distinction of E-E, E-T and E-D, and make them relatively close to each other. Hence, we train a single general classifier for all three tasks. Specifically, not like SEC that takes a fixed batch size, we organize all mention pairs in the same document into a single batch, which helps maintain global consistency in those densely annotated corpora for each prediction via the global relation anchored to DCT.

Thus, we represent each mention $n_i \in N$ by concatenating its original BERT or RoBERTa embedding $\boldsymbol{h}_i \in \boldsymbol{H}_0$ and GCN out $\boldsymbol{m}_i \in \boldsymbol{H}_G$, then the pair representation $\boldsymbol{d}_{i,j}$ for $n_i, n_j \in N$ is as follows.

$$\boldsymbol{d}_{i,j} = \text{concat}([\boldsymbol{h}_i; \boldsymbol{m}_i; \boldsymbol{h}_j; \boldsymbol{m}_j]) \tag{3}$$

Secondly, since DCT is the anchor to connect the relative events and timexes in a document, we incorporate the DCT representation $\boldsymbol{h}_{dct} \in \boldsymbol{H}_0$ into the classifier to enforce the E-E and E-T pairs noticing their temporal orders with DCT as follows.

$$P(r \mid n_i, n_j) = \text{softmax}(\text{MLP}([\boldsymbol{d}_{i,j}; \boldsymbol{h}_{dct}])) \tag{4}$$

where MLP is the single Multi-layer Perceptron classifier for all tasks. Then we calculate the cross-entropy loss for each task as follows.

$$\mathcal{L}_{\mathcal{T}} = -\sum_{n_i, n_j \in \mathcal{T}} \log P(r = r_{(n_i, n_j)} \mid n_i, n_j) \tag{5}$$

where $\mathcal{T} \in \{EE, ET, ED\}$ refers to one of the E-E, E-T and E-D tasks, and $r_{(n_i, n_j)}$ is the golden label for the mention pair (n_i, n_j). Finally, we combine the three task losses as follows, where α and β are trade-off parameters.

$$\mathcal{L} = \mathcal{L}_{EE} + \alpha \cdot \mathcal{L}_{ET} + \beta \cdot \mathcal{L}_{ED} \tag{6}$$

Dataset	E-E	E-T	E-D
TBD	6088	2001	1737
TDD-Man	6150	-	1221
TDD-Auto	38302	-	1221

Table 2: Statistics of the three datasets.

3.4 DCT-Guided Global Inference

Previous work applied different strategies for global consistent predictions, such as ILP constraints (Ning et al., 2017). Recent studies found that ILP constraints can improve consistency, while they maybe generally harm the F1 score (Liu et al., 2021).

In our multi-task framework DTRE, since the performance of E-D are high reliable (e.g., $F1 > 80$ on TBD and $F1 > 90$ on TDD), we propose a DCT-Guided Global Inference mechanism DGI to treat all of the E-D predictions as golden labels and use them to check whether those E-E and E-T instances obey transitivity in document-level. For example, if an event mention *e1* is before *DCT*, and *DCT* is before *e2*, then the label of *(e1, e2)* should be before. Specially, if the predicted label of *(e1, e2)* is vague, then we do not change it through DGI.

4 Experimentation

In this section, we first introduce the datasets and the experimental settings, and then report results on our proposed DTRE and baselines.

4.1 Datasets and Experimental Settings

We evaluate our DTRE on two popular datasets TimeBank-Dense (TBD) (Cassidy et al., 2014) and TDDiscourse (TDD) (Naik et al., 2019). TBD densely annotated 4 TLINKs (E-E, E-D, E-T and T-D) within an adjacent sentence slide window (as DCT does not explicitly occur in texts, each event or Timex has a temporal relation annotation with DCT). TBD has six types of labels, i.e. *before, after, include, is included, simultaneous*, and *vague*. There are 243 T-D instances (2%) in TBD, we do not distinguish T-D with E-D for simplicity in this paper. TDD is a discourse-level temporal ordering corpus and has five types of event temporal relations except for the vague relation in TBD, which makes the class distribution more balanced. TDD consists of two subsets: TDD-Man and TDD-Auto, which are manual and auto annotated, respectively. Since TDD does not annotate E-D relation, we additionally take the E-D examples in TBD (except those vague samples) into the training step, as TDD shares the same documents and event annotations with TBD. Table 2 shows the statistics of the three datasets.

We split the standard train/dev/test sets on TBD and TDD datasets following (Mathur et al., 2021) and report Precision (P), Recall (R), and micro-F1 scores. Since previous studies on TBD treat vague as none type (Liu et al., 2021) or positive type (Cheng et al., 2020) when calculating F1 scores, we report both (five types/six types) for fair comparison.

In DAG, we utilize SpaCy[2] to extract the entities and word dependency trees. Entity coreference resolution is obtained by neuralcoref[3] toolkit. We use cosine_similarity()[4] to obtain the cosine similarity between mentions. We tune all the hyperparameters on the development set. For the pre-trained encoder, we choose BERT-base and RoBERTa-large architecture following previous work for fair comparison. The number of MLP layer is set to 2, and the number of GCN layer is set to 2. The hidden dimension of GCN is set to 768 and 1024 for BERT and RoBERTa, respectively. The trade-off parameters α and β in Eq.6 are both set to 1.0.

4.2 Experimental Results

To evaluate the performance of our model DTRE, we conduct seven strong baselines for comparison as follows:

- **DP-RNN** (Cheng and Miyao, 2017): a model applied event pair's shortest dependency path (SDP) into Bi-LSTM, while it utilizes single event's DP branch for the E-D task;
- **GCL** (Meng and Rumshisky, 2018): a context-aware neural network with a uniform architecture for E-E, E-T and E-D;
- **SEC** (Cheng et al., 2020): a multi-task source event centric model that dynamically managed event representations across three TLINK types;
- **Rand** (Cheng et al., 2020): a multi-task model that RNN module is removed in **SEC**, which

[2] https://spacy.io/

[3] https://github.com/huggingface/neuralcoref/

[4] https://pytorch.org/docs/stable/nn.functional.html

Method	E-E	E-T	E-D
DP-RNN	-/52.9	-/47.1	-/54.6
GCL	-/57.0	-/48.7	-/48.9
SEC*	-/65.0	-/55.8	-/65.9
Rand*	63.0/61.4	60.2/54.8	75.9/65.2
ECONET†	66.8/-	-/-	-/-
TIMERS*	67.8/-	-/-	-/-
BERT*	62.2/59.7	49.4/49.0	73.8/69.4
RoBERTa†	62.4/59.8	51.5/49.8	76.3/72.1
DTRE*	69.2/68.4	64.9/62.1	77.7/73.6
DTRE†	**72.3/70.2**	**70.6/67.5**	**81.9/75.8**

Table 3: F1-score comparison of E-E, E-T and E-D on TBD. The figures before and after "/" refer to the results on five and six types, where "*" and "†" refer to the encoder BERT and RoBERTa, respectively.

uses randomly initialized learnable embeddings to represent DCT.

- **BERT**-based (Devlin et al., 2019) and **RoBERTa**-based (Liu et al., 2019b) Transformer: the models follow (Zhao et al., 2021) to conduct a pair-wise classification in which the E-D task utilizes our DCT representation;
- **UCGraph** (Liu et al., 2021): the first work introducing graph representation learning and uncertainty modeling to temporal relation extraction;
- **TIMERS** (Mathur et al., 2021): a graph-based method on the E-E task that merges syntactic, temporal, and rhetorical information;
- **ECONET** (Han et al., 2021): a pre-trained method on the E-E task using millions of raw temporal relative data.

Table 3 and Table 4 show the performance comparison of our model DTRE and the baselines on the datasets TBD, TDD-Man and TDD-Auto, respectively. It can be observed that our model DTRE outperforms all baselines on the three datasets significantly (significance test with $p < 0.05$).

Compared with the SOTA models TIMERS (E-E) and SEC (E-T and E-D) on TBD, DTRE improves the F1-score by 4.5, 11.7, and 9.9 on the three tasks E-E, E-T and E-D, respectively. Compared with the SOTA model TIMERS on TDD-Auto and TDD-Man, DTRE gains improvements (E-E) of 10.7 and 10.8 on F1 score, respectively.

	TDD-Man			TDD-Auto		
Method	P	R	F1	P	R	F1
UCGraph*	44.5	42.3	43.4	66.1	56.9	61.2
TIMERS*	43.7	46.7	45.5	64.3	72.7	71.1
BERT*	39.9	39.9	39.9	62.3	62.3	62.3
RoBERTa†	44.8	44.8	44.8	76.7	76.7	76.7
DTRE*	50.0	50.0	50.0	70.2	70.2	70.2
DTRE†	**56.3**	**56.3**	**56.3**	**81.8**	**81.8**	**81.8**

Table 4: Performance comparison of E-E on TDD-Man and TDD-Auto, where "*" and "†" refer to the encoder BERT and RoBERTa, respectively.

These results verify the effectiveness of our DTRE on extracting all kinds of temporal relations. Besides, compared with E-E of our DTRE in Table 3, E-T and E-D gain much higher improvements. This result further indicates that timexes and DCT are the critical clues for temporal relation extraction and our DCT representation is effective in DTRE.

In Table 3, the pre-trained models BERT and RoBERTa achieve similar performance on TBD, while RoBERTa outperforms BERT rapidly on both TDD-Man and TDD-Auto in Table 4. These results indicate that RoBERTa is better than BERT as encoder to extract the temporal relations among inter-sentence event mentions and RoBERTa works well on a large-scale training set (e.g., TDD-Auto).

5 Ablation Study

In this section, we conduct the ablation study of DTRE (RoBERTa-based) on TBD as examples. It is worth mentioning that BERT-based DTRE on TBD and TDD also show the similar results and we do not describe here for simplification.

5.1 Impacts of Different Modules

To verify the effectiveness of each module in DTRE, we conduct the experiments on the following variants and baseline:

- *w/o DAG*: we remove DAG and only use the original RoBERTa embeddings in pair representation;
- *w/o DAML*: we remove DAML and separately train each task;
- *w/o DGI*: we remove DGI from DTRE. The results are shown in Table 5.

Method	E-E	E-T	E-D
DTRE (RoBERTa)	72.3	70.6	81.8
w/o DAG	-3.5	-3.6	-1.6
w/o DAML	-5.4	-15.8	-2.5
w/o DGI	-0.6	-2.0	-

Table 5: F1 scores of DTRE and its variants on TBD.

Variant	E-E	E-T	E-D
Rand	63.0*	60.2*	75.9*
CreatN (*w/o* DCT_{feat})	64.2	62.3	74.3
CreatD (*w/o* DCT_{feat})	66.0	62.4	75.4
CreatND (*w/o* DCT_{feat})	66.3	63.7	76.8
CreatN + DCT_{feat}	65.8	64.5	76.3
CreatD + DCT_{feat}	66.5	65.2	78.1
CreatND+ DCT_{feat}	67.1	66.0	79.3

Table 6: Effects of different representation of DCT and the DCT-aware feature, where "+DCT_{feat}" denotes that adding $\boldsymbol{h}_{dct}$ in pair representation mentioned in Eq.4 and "*" denotes our re-implementations on five temporal types without the *vague* relation.

Resource	E-E	E-T	E-D
Single task	66.9	54.8	79.3
E-E&E-T	68.4	66.8	-
E-E&E-D	70.8	-	80.4
E-E&E-T&E-D	71.7	68.6	81.9

Table 7: Effect of training resources on TBD, where DGI is removed for fair comparison.

The results of DTRE and its variants on TBD are showed in Table 5. When we remove DAG (*w/o DAG*), the F1 scores of the E-E, E-T and E-D tasks decrease by 3.5, 3.6, and 1.6, respectively. This indicates the importance of the document structure for temporal reasoning, especially for the E-E and E-T tasks.

Removing DAML (*w/o DAML*) leads to the biggest performance deterioration for E-E, E-T and E-D by 5.4, 15.8, and 2.5, respectively. This indicates that the three tasks can complement each other in a unified framework. The significant improvements of E-E and E-T also show that utilizing DCT to anchor events and timexes is an effective way for temporal relation extraction, which can be regarded as a bridge to link event pair or event-timex pair.

Moreover, E-E and E-T tasks benefit from DCT-guided global inference (*w/o DGI*) with the gains of 0.6 and 2.0, although the transitivity is harmed by *vague* relation to some extent, which verifies that the E-D task can provide direct temporal clues for E-E and E-T.

5.2 Impacts of DCT Representations

Obviously, DCT can often provide explicit temporal information and be a bridge to link those events without temporal clues. To further analyze the impacts of the different DCT representations and our DCT-aware feature $\boldsymbol{h}_{dct}$ in pair representation (Eq.4), we adopt several DCT representation strategies in Table 1 and the results are shown in Table 6. Specifically, to compare with the existing multi-task learning model *Rand* (Cheng et al., 2020), we remove DAG and DGI from our DTRE, and only conduct the resource-shared multi-task learning with BERT-base settings for direct comparison.

As showed in Table 6, compared with *Rand*, our three strategies (*w/o* DCT_{feat}) improve the F1 scores of E-E and E-T significantly. Their randomly initialized learnable embeddings do not explicitly contain any DCT information, while our representation uses a sentence to let DCT explicitly occur in the document. This result indicates that DCT is an important hub to connect the events and timexes scattered in a document. Although our DCT-indicator sentences are simple, it also shows that all three strategies are effective, especially **CreatND** with the highest improvement.

We also introduce our DCT-aware feature $\boldsymbol{h}_{dct}$ to our model and the results in Table 6 indicate that it can boost all three tasks, especially E-D and E-T. In this way, we enforce the classifier to pay more attention to the related E-D relations when predicting E-E and E-T pairs and then can gain the improvement for all tasks.

5.3 Impacts of DAML

We also evaluate the impacts of DAML on different tasks. Intuitively, we remove one task from our DTRE (RoBERTa) and the results on TBD are shown in Table 7. It shows that the E-T and E-D tasks play an important role in our DTRE framework, which contributes the performance gains of 1.5 and 3.9 for E-E. Moreover, although the sample size of E-T is larger than that of E-D (2001

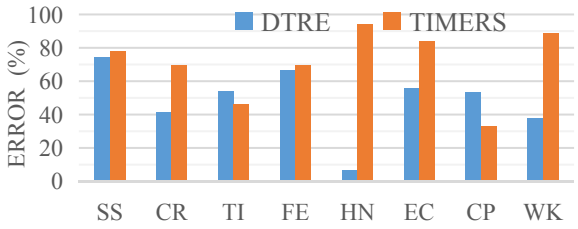

Figure 3: Error analysis on manually annotated phenomena in the test set of TDD-Man. SS: SingleSent, CR: Chain Reasoning, TI: Tense Indicator, FE: Future Events, HN: Hypothetical/ Negated, EC: Event Coreference, CP: Causal/ Prereq, WK: World Knowledge (detail definitions please refer to Naik et al. (2019))

vs 1494), E-D is better than E-T as the auxiliary task of E-E. This result indicates that E-D is more effective than E-T for this multi-task framework and verifies the core role of DCT in temporal relation extraction. Besides, with the help of E-E, E-T can significantly improve the F1-score by 12.0 and the reason is that the number of E-E instances are larger than that of E-T.

5.4 Error Analysis

To analyze the errors in our DTRE, we use the annotated cues (Naik et al., 2019) between events in TDD-Man and compare them with the SOTA model TIMERS (Mathur et al., 2021). Figure 3 shows the error percentages of eight cue types on TDD-Man. We can find out that our DTRE deals well with HN (Hypothetical/Negated), WK (World Knowledge) and CR (Chain Reasoning), while it suffers from SS (SingleSent) and FE (Future Events). The reason behind this is that SS and FE need more event-level semantics to predict temporal relation while our DTRE only focuses on using the novel DCT representation and the intrinsic relations among event, timex and DCT to boost all three temporal relation extraction tasks.

TIMERS suffers from TLINK pairs which depend on CR, HN, EC (Event Coreference) and WK, while our DTRE achieves significant progress on them. This result indicates that the document-level knowledge (e.g., DCT) is a core clue for temporal relation extraction and our DCT-anchoring multi-task framework regards the whole document as the input and can incorporate the document-level knowledge. However, TIMERS is better to deal with TI (Tense Indicator) and CP (Causal / Prereq), because it focuses on mining more semantic information inside E-E relations.

As for the errors in E-T and E-D, most of them come from two aspects: 1) there are no explicit temporal words or clues in events, and 2) some timexes do not express a specific duration or time point (e.g., "recently" and "a few years ago").

6 Conclusion

In this paper, we proposed a unified DCT-centered temporal relation extraction model DTRE to discover the relations among events, timexes and DCT. Specifically, we first introduce sentence-style DCT to unify the expressions of event, timex and DCT. Then, we apply a DCT-aware graph to obtain their contextual structural representations. Furthermore, we propose a DCT-anchoring multi-task framework to jointly predict three tasks of temporal relation extraction in a batch. Finally, we provide a DCT-guided global inference to further enhance the global consistency among different relations. Experimental results on three popular datasets show that our DTRE outperforms several SOTA baselines on E-E, E-T and E-D significantly. Our future work will focus on discovering effective graph structure and inference mechanism for temporal relation extraction.

Acknowledgements

The authors would like to thank the three anonymous reviewers for their comments on this paper. This research was supported by the National Natural Science Foundation of China (Nos. 61836007, 62276177 and 62006167), and Project Funded by the Priority Academic Program Development of Jiangsu Higher Education Institutions (PAPD).

References

Miguel Ballesteros, Rishita Anubhai, Shuai Wang, Nima Pourdamghani, Yogarshi Vyas, Jie Ma, Parminder Bhatia, Kathleen McKeown, and Yaser Al-Onaizan. 2020. Severing the edge between before and after: Neural architectures for temporal ordering of events. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 5412–5417.

Taylor Cassidy, Bill McDowell, Nathanael Chambers, and Steven Bethard. 2014. An annotation framework for dense event ordering. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 501–506.

Nathanael Chambers, Taylor Cassidy, Bill McDowell, and Steven Bethard. 2014. Dense event ordering

with a multi-pass architecture. *Transactions of the Association for Computational Linguistics*, 2:273–284.

Nathanael Chambers and Daniel Jurafsky. 2008. Jointly combining implicit constraints improves temporal ordering. In *Proceedings of the 2008 Conference on Empirical Methods in Natural Language Processing*, pages 698–706.

Nathanael Chambers, Shan Wang, and Dan Jurafsky. 2007. Classifying temporal relations between events. In *Proceedings of the 45th Annual Meeting of the Association for Computational Linguistics Companion Volume Proceedings of the Demo and Poster Sessions*, pages 173–176.

Fei Cheng, Masayuki Asahara, Ichiro Kobayashi, and Sadao Kurohashi. 2020. Dynamically updating event representations for temporal relation classification with multi-category learning. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 1352–1357.

Fei Cheng and Yusuke Miyao. 2017. Classifying temporal relations by bidirectional LSTM over dependency paths. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 1–6.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186.

Quang Do, Wei Lu, and Dan Roth. 2012. Joint inference for event timeline construction. In *Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning*, pages 677–687.

Jennifer D'Souza and Vincent Ng. 2013. Classifying temporal relations with rich linguistic knowledge. In *Proceedings of the 2013 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 918–927.

Rujun Han, Qiang Ning, and Nanyun Peng. 2019. Joint event and temporal relation extraction with shared representations and structured prediction. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 434–444.

Rujun Han, Xiang Ren, and Nanyun Peng. 2021. ECONET: Effective continual pretraining of language models for event temporal reasoning. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 5367–5380.

Kenton Lee, Luheng He, Mike Lewis, and Luke Zettlemoyer. 2017. End-to-end neural coreference resolution. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 188–197.

Manling Li, Tengfei Ma, Mo Yu, Lingfei Wu, Tian Gao, Heng Ji, and Kathleen McKeown. 2021. Timeline summarization based on event graph compression via time-aware optimal transport. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 6443–6456.

Jian Liu, Jinan Xu, Yufeng Chen, and Yujie Zhang. 2021. Discourse-level event temporal ordering with uncertainty-guided graph completion. In *Proceedings of the Thirtieth International Joint Conference on Artificial Intelligence, IJCAI-21*, pages 3871–3877.

Xiaodong Liu, Pengcheng He, Weizhu Chen, and Jianfeng Gao. 2019a. Multi-task deep neural networks for natural language understanding. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 4487–4496.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019b. Roberta: A robustly optimized bert pretraining approach. *arXiv*, abs/1907.11692.

Puneet Mathur, Rajiv Jain, Franck Dernoncourt, Vlad Morariu, Quan Hung Tran, and Dinesh Manocha. 2021. TIMERS: Document-level temporal relation extraction. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 2: Short Papers)*, pages 524–533.

Yuanliang Meng and Anna Rumshisky. 2018. Context-aware neural model for temporal information extraction. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 527–536.

Paramita Mirza and Sara Tonelli. 2016. CATENA: CAusal and TEmporal relation extraction from NAtural language texts. In *Proceedings of COLING 2016, the 26th International Conference on Computational Linguistics: Technical Papers*, pages 64–75.

Aakanksha Naik, Luke Breitfeller, and Carolyn Rose. 2019. TDDiscourse: A dataset for discourse-level temporal ordering of events. In *Proceedings of the 20th Annual SIGdial Meeting on Discourse and Dialogue*, pages 239–249.

Qiang Ning, Zhili Feng, and Dan Roth. 2017. A structured learning approach to temporal relation extraction. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 1027–1037.

Qiang Ning, Hao Wu, Rujun Han, Nanyun Peng, Matt Gardner, and Dan Roth. 2020. TORQUE: A reading comprehension dataset of temporal ordering questions. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1158–1172.

Qiang Ning, Hao Wu, and Dan Roth. 2018. A multi-axis annotation scheme for event temporal relations. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1318–1328.

Yunseok Noh, Yongmin Shin, Junmo Park, A-Yeong Kim, Su Jeong Choi, Hyun-Je Song, Seong-Bae Park, and Seyoung Park. 2020. Wire: An automated report generation system using topical and temporal summarization. In *Proceedings of the 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '20, page 2169–2172.

James Pustejovsky, Patrick Hanks, Roser Sauri, Andrew See, Robert Gaizauskas, Andrea Setzer, Dragomir Radev, Beth Sundheim, David Day, Lisa Ferro, et al. 2003. The timebank corpus. In *Corpus linguistics*, volume 2003, page 40. Lancaster, UK.

Xinyu Zhao, Shih-Ting Lin, and Greg Durrett. 2021. Effective distant supervision for temporal relation extraction. In *Proceedings of the Second Workshop on Domain Adaptation for NLP*, pages 195–203.

Document-level Biomedical Relation Extraction Based on Multi-Dimensional Fusion Information and Multi-Granularity Logical Reasoning

Lishuang Li * and **Ruiyuan Lian** † and **Hongbin Lu** and **Jingyao Tang**
School of Computer Science and Technology
Dalian University of Technology
Dalian, China
lils@dlut.edu.cn

Abstract

Document-level biomedical relation extraction (Bio-DocuRE) is an important branch of biomedical text mining that aims to automatically extract all relation facts from the biomedical text. Since there are a considerable number of relations in biomedical documents that need to be judged by other existing relations, logical reasoning has become a research hotspot in the past two years. However, current models with reasoning are single-granularity only based on one element information, ignoring the complementary fact of different granularity reasoning information. In addition, obtaining rich document information is a prerequisite for logical reasoning, but most of the previous models cannot sufficiently utilize document information, which limits the reasoning ability of the model. In this paper, we propose a novel Bio-DocuRE model called **FILR**, based on Multi-Dimensional **F**usion **I**nformation and Multi-Granularity **L**ogical **R**easoning. Specifically, FILR presents a multi-dimensional information fusion module MDIF to extract sufficient global document information. Then FILR proposes a multi-granularity reasoning module MGLR to obtain rich inference information through the reasoning of both entity-pairs and mention-pairs. We evaluate our FILR model on two widely used biomedical corpora CDR and GDA [1]. Experimental results show that FILR achieves state-of-the-art performance.

1 Introduction

Biomedical relation extraction(Bio-RE) is an important branch of biomedical text mining, which always draws researchers' attention. According to the different length of processing text, Bio-RE is generally divided into sentence-level RE and document-level RE. In sentence-level RE, all involved entities and relations are within a sentence.

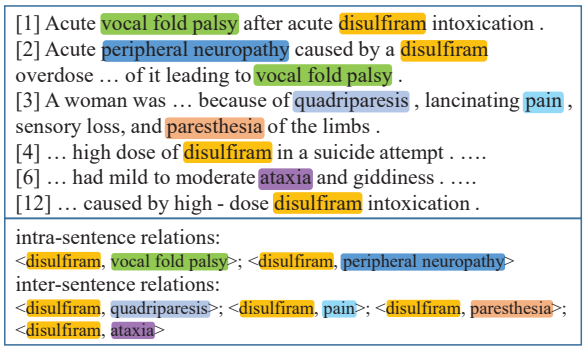

Figure 1: An example of Bio-DocuRE from the CDR dataset. We use the same color to denote the mentions of the same entity.

Document-level RE is more complex than its sentence counterpart. As shown in Fig. 1, in document-level biomedical relation extraction(Bio-DocuRE), the processed text is a whole document containing multiple sentences. "*disulfiram*" is a chemical and other entities are diseases. Each entity always has more than one mention, for example, "*disulfiram*" has four mentions in the document. In this example, there are two intra-sentence and four inter-sentence Chemidical Induce Disease(CID) relations, and the inter-sentence relation means that the two entities involved in a relation may span multiple sentences or even the entire document.

Early research (Guo et al., 2015; Jiang et al., 2016; Zheng et al., 2017; Xiong et al., 2019) efforts focus on sentence-level RE. However, in real-world scenarios, many relations are expressed in multiple sentences, and the sentence-level models fail to recognize them. Therefore, more and more attention has been paid to DocuRE (Zhou et al., 2016; Gu et al., 2017; Verga et al., 2018). Most existed Bio-DocuRE studies are sequence-based (Nguyen and Verspoor, 2018; Li et al., 2020; Jiang et al., 2020), graph-based (Tran et al., 2020; Zhang et al., 2020a; Wang et al., 2020; Li et al., 2021b) or transformer-based (Liu et al., 2020; Zhou et al., 2021; Xu et al., 2021) methods, which never con-

*Corresponding author
† The author contributes equally to the first author in this work.
[1] https://github.com/Luguo-ry/FILR

Proceedings of the 29th International Conference on Computational Linguistics, pages 2098–2107
October 12–17, 2022.

sider the problem of relation reasoning and have limited effect. In the past two years, researchers noticed that there are a lot of relations in biomedical text that need to be judged relying on other relevant relations. So, Bio-DocuRE methods integrating logical reasoning have become a research hotspot (Li et al., 2021a; Christopoulou et al., 2019; Nan et al., 2020; Zhang et al., 2020b; Zeng et al., 2021; Zhao et al., 2022).

Mention, entity, mention-pair, and entity-pair are four document elements in different granularity, and we notice that the purpose of logical reasoning is essential to capture the dependencies between certain document elements. Consequently, we divide the logical reasoning mechanisms in Bio-DocuRE into the following four granularity: mention-based, entity-based, mention-pair-based, and entity-pair-based. All current methods with reasoning only consider single-granularity, and they perform mention-based (Li et al., 2021a), or entity-based (Christopoulou et al., 2019; Nan et al., 2020; Zhang et al., 2020b; Zeng et al., 2021), or entity-pair-based (Zhao et al., 2022) reasoning. Mention-based and entity-based reasoning mainly focuses on local token-level interactions and results in limited effect. While entity-pair-based reasoning focuses on global entity-relation interaction and could provide direct guidance for document-level relation classification. Compared with entity-pair-based reasoning, mention-pair-based reasoning can capture the global dependencies among all mention-pairs and provide fine-grained mention-relation reasoning information, which complements each other with the coarse-grained entity-relation reasoning information. Regrettably, no study notices the importance of mention-pair-based reasoning in Bio-DocuRE so far. Meanwhile, the current single-granularity methods leads to insufficient reasoning. Therefore, in order to obtain the effective and sufficient reasoning information, we propose a multi-granularity logical reasoning module with both mention-pair-based and entity-pair-based reasoning.

In addition, adequately mining global document information is an important step for Bio-DocuRE and is also the premise of performing logical reasoning. However, most of the current models suffer from the insufficient utilization of the document information because they either obtain the document information from a single perspective (Li et al., 2021a; Christopoulou et al., 2019; Nan et al., 2020; Zhang et al., 2020b; Zeng et al., 2021; Zhao et al., 2022), or ignore the interaction and fusion of different document information (Zhang et al., 2020a; Wang et al., 2020; Xu et al., 2021). Both the global context and structural information of a document are important, which express the document from different views. Therefore, we extract both the global context and structural information of documents in this paper and propose a multi-dimensional information fusion module to fuse them.

Furthermore, in the process of obtaining the document global structural information, most current methods use GCN (Kipf and Welling, 2016) to iteratively update the document graph. However, GCN has been proved (Li et al., 2018; Luan et al., 2019; Zhao and Akoglu, 2019) to suffer from the over-smoothing problem when stacking the convolution layers, that is, the features of the nodes in graph would converge to similar values after deep iterations. Therefore, to relieve the GCN over-smoothing problem, we improve the commonly used document graph structure (Wang et al., 2020; Christopoulou et al., 2019; Li et al., 2021b) by designing a bridge node to capture the document global structural information more effectively within a finite number of iteration layers.

In this paper, to address the above issues, we present a novel Bio-DocuRE model called **FILR** based on multi-dimensional **F**usion **I**nformation and multi-granularity **L**ogical **R**easoning. We conduct experiments on two Bio-DocuRE datasets CDR (Li et al., 2016) and GDA (Wu et al., 2019). Our contributions can be summarized as follows:

- We propose a novel Bio-DocuRE model FILR. Experimental results on two benchmark datasets show that our model FILR can achieve state-of-the-art performance.
- We propose a multi-dimensional information fusion module called MDIF to adequately extract the global information of documents. Meanwhile, a bridge node is designed on the document graph to relieve the over-smoothing problem.
- We propose a multi-granularity logical reasoning module called MGLR to simultaneously conduct mention-pair-based and entity-pair-based reasoning.

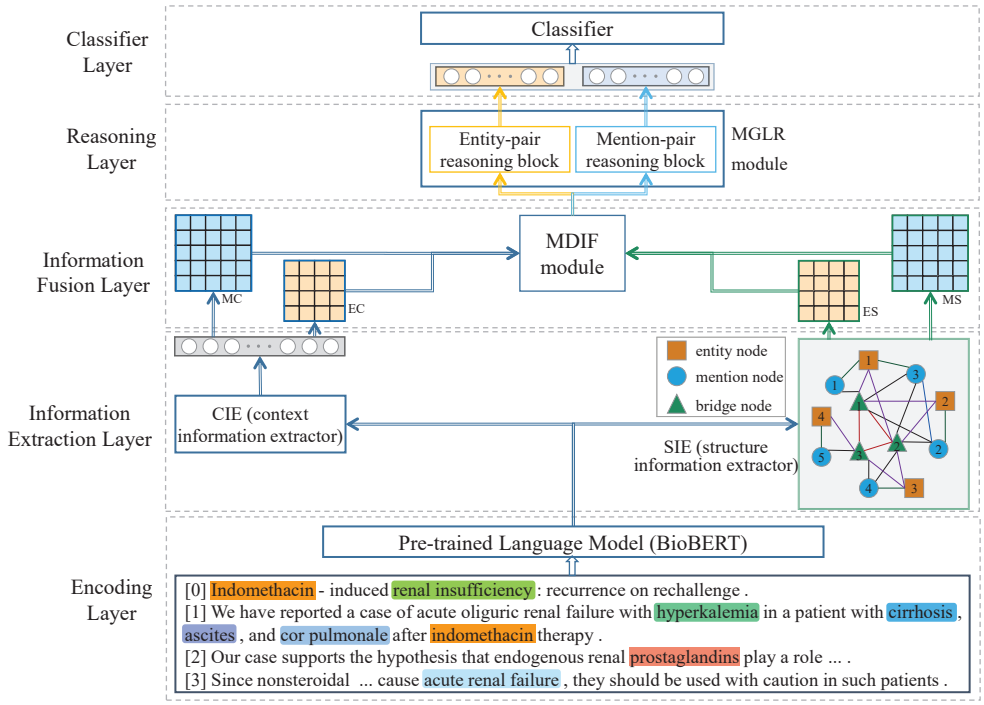

Figure 2: The overall framework of FILR, where MC means "mention-context map", EC means "entity-context map", ES means "entity-structure map" and MS means "mention-structure map".

2 Model

In this section, we describe the overall framework of the proposed model FILR in Fig. 2. FILR makes an effective use of biomedical documents information and conducts multi-granularity reasoning to improve the performance of Bio-DocuRE.

2.1 Task Definition

Document-level biomedical relation extraction task can be defined as a classification problem. Given an annotated document T and a set of entities $e_i (1 \leq i \leq N_e)$, our goal is to extract the relations between all target entity pairs (e_h, e_t). Note that in one document, an entity pair, also called a relation instance, usually owns more than one corresponding mention pairs because each entity e_i can occur multiple times by entity mentions $m_{ij} (1 \leq j \leq N_{e_i})$.

2.2 Encoding Layer

To better model the semantics of input document, FILR adopts BioBERT (Lee et al., 2020) as the document encoder, which is a domain-specific language representation model pre-trained on the large biomedical dataset for understanding complex biomedical texts. And its effectiveness has been proven surprisingly in various biomedical text mining tasks.

Let $D_T = [w_1, w_2, ..., w_N]$ be the input document T with N tokens. To better represent the mentions which usually span a few consecutive tokens in T, we first insert a special marker "&" at the start and end of mentions to mark the mention's span (Zhou et al., 2021). Then, the embedding representations of each token and marker "&" with rich semantic information can be got by the encoder. The embedding representation of T can be described by:

$$X_T = [x_1, x_2, ..., x_i, ..., x_{N'}], x_i \in R^{d_{emb}}, \quad (1)$$

where N' is the total number of tokens in T after inserting the marker, and d_{emb} denotes the embedding dim of words. In this paper, for each mention, we take the embedding of the start marker "&" as the mention embedding m_{ij}, and apply *logsumexp* (Jia et al., 2019) on the mentions referring to the same entity to obtain the entity embedding e_i:

$$e_i = log \sum_{j=1}^{N_{e_i}} exp(m_{ij}). \quad (2)$$

2.3 Information Extraction Layer

In this section, to obtain mention and entity representations that contain the global context and structural information of documents, we design two information extractors, respectively.

Context information extractor(CIE) Zhou et al. (2021) and Zhao et al. (2022) have proven the effectiveness of the pre-trained multi-head attention matrix $A \in R^{H \times N' \times N'}$ in aggregating the entity context information. So in this paper, we obtain the entity and mention representations with rich context information using similar methods. A_{kab} represents the attention from token a to token b in the kth attention head. For mention m_{ij}, we first take the attention from start marker "&" to all tokens as the mention attention and get the mention-context attention matrix $A^{m_{ij}} \in R^{1 \times N'}$ by averaging all attention heads. For entity e_i, we get the entity-context attention matrix $A^{e_i} \in R^{1 \times N'}$ by averaging all related mention-context attention. Then we obtain the entity representations E_c and mention representations M_c with context information by:

$$A_1^m = \frac{A^m}{1^T A^m}, A^m \in R^{N_m \times N'}, \quad (3)$$

$$M_c = A_1^m * X_T, M_c \in R^{N_m \times d_{emb}}, \quad (4)$$

$$A_1^e = \frac{A^e}{1^T A^e}, A^e \in R^{N_e \times N'}, \quad (5)$$

$$E_c = A_1^e * X_T, E_c \in R^{N_e \times d_{emb}}, \quad (6)$$

where N_m is the total number of mentions in the document, A^m and A^e are the attention matrix of all mentions and entities, respectively.

Structure information extractor(SIE) In this part, we improve the commonly used document graph structure to avoid the over-smoothing problem of GCN. The main idea is to make the nodes in the document graph cover a wider information field in the initial state so that the model can capture the global structure information of the document before the over-smoothing problem occurs. So we construct a document-level graph with three types of nodes: entity nodes, mention nodes, and bridge nodes. The bridge node contains information across two sentences, so it broadens the information field and retains the structure of sentences in the meantime. And it is more conducive to the interaction and aggregation of long-distance nodes' information. With this document graph, the shallow layers of GCN can also well model the structure of documents, which alleviates the over-smoothing problem of GCN to some extent.

Specifically, we let $n_{b_i} = [b_i; t_b], n_{b_i} \in R^{d_{emb}+d_t}$ as the representation of bridge node b_i, where b_i is defined by averaging the representation of all tokens in the linked two sentences: $b_i = avg_{x_k \in \{s_i, s_{i+1}\}}(x_k)$. Meanwhile, we let $n_{m_{ij}} = [m_{ij}; t_m]$ and $n_{e_i} = [e_i; t_e]$ as the representations of mention and entity nodes, respectively. $t_b, t_m, t_e \in R^{d_t}$ are type representations of nodes.

Then, we construct five types of edges: (1) Mention-Mention edges(MM): two mentions are connected with MM edge if they co-occur in a single sentence. (2) Mention-Bridge edges(MB): a mention node is connected to a bridge node with MB edge if this mention appears in the sentences linked with the bridge node. (3) Mention-Entity edge(ME): ME edge is added between a mention node and an entity node if the mention refers to the entity. (4) Entity-Bridge edge(EB): EB edge is added between an entity node and a bridge node if at least one mention of the entity is connected to the bridge node. (5) Bridge-Bridge edge(BB): BB edge is added between any two bridge nodes. In this paper, all edges are undirected.

After the document graph is constructed, R-GCN (Schlichtkrull et al., 2018) is applied on the graph to model the structure information of documents, and the iterative update process is defined as follows:

$$n_i^{l+1} = \sigma\left(\sum_{r \in R} \sum_{j \in N_i^r} \frac{1}{|N_i^r|} W_r^l n_j^l + W_0^l n_i^l\right), \quad (7)$$

where $\sigma(\cdot)$ is an activation function, W_r^l,W_0^l are the parameters for edge type r and self-connection in l^{th} layer. N_i^r is the set of neighbouring nodes connected to n_i by edge type r. R is the set of edge types, and n_i^{l+1} is the ith node representation resulted from the lth R-GCN layer.

After L-layer R-GCN is stacked, mention representations $M_s \in R^{N_m \times d_{hid}}$ and entity representations $E_s \in R^{N_e \times d_{hid}}$ with structural information can be obtained, where d_{hid} is the hidden size of R-GCN, and d_{hid} is equal to the d_{emb} in this paper.

2.4 Information Fusion Layer

In this section, we propose a multi-dimensional information fusion module called MDIF to fuse the global context and structure information of documents.

First, four relation maps M_C(mention-context map), E_C(entity-context map), M_S(mention-structure map), E_S(entity-structure map) are built

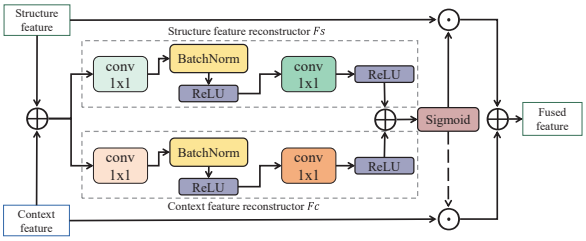

Figure 3: The structure of MDIF. $\oplus$ denotes addition and $\odot$ denotes the element-wise multiplication.

based on M_c, E_c, M_s and E_s as follows:

$$M_C = M_c \otimes M_c^T, M_C \in R^{d_{emb} \times N_m \times N_m}, \quad (8)$$

$$E_C = E_c \otimes E_c^T, E_C \in R^{d_{emb} \times N_e \times N_e}, \quad (9)$$

$$M_S = M_s \otimes M_s^T, M_S \in R^{d_{hid} \times N_m \times N_m}, \quad (10)$$

$$E_S = E_s \otimes E_s^T, E_S \in R^{d_{hid} \times N_e \times N_e}, \quad (11)$$

where $\otimes$ is the multiplication by Einstein summation convention, $*^T$ is the matrix transpose.

Then MDIF is proposed for Bio-DocuRE inspired by Dai et al. (Dai et al., 2021) in computer vision. Our goal is to obtain entity-pair and mention-pair representations with global context and structural information.

The structure of MDIF is shown in Fig. 3. It receives the context and structure features as input. First, these two features are fused to obtain the shallow fusion feature. Then, the refined context and structure features are re-extracted with two similar information reconstructors. Both of them are mainly composed of two layers of point-wise convolution (PWConv), which only exploits point-wise channel interactions for the shallow fusion feature. Next, the refined structure and context features are fused again to obtain the deep fusion feature, and the channel-aware weight is calculated by a sigmoid function. Finally, the original structure and context features are fused by the channel-aware weight. The entity-pair representations E_{sc} and mention-pair representations M_{sc} with rich fusion information can be expressed as:

$$E_{sc} = E_S \odot W_{sc}^e + E_C \odot (1 - W_{sc}^e), \quad (12)$$

$$M_{sc} = M_S \odot W_{sc}^m + M_C \odot (1 - W_{sc}^m), \quad (13)$$

$$W_{sc}^e = \delta(F_s^e(E_S \odot E_C) + F_c^e(E_S \odot E_C)), \quad (14)$$

$$W_{sc}^m = \delta(F_s^m(M_S \odot M_C) + F_c^m(M_S \odot M_C)), \quad (15)$$

where $E_{sc} \in R^{d_{emb} \times N_e \times N_e}$, and $M_{sc} \in R^{d_{emb} \times N_m \times N_m}$. $\odot$ denotes the element-wise multiplication. F_s^* and F_c^* are the structure and context feature reconstructors, respectively. $\delta(\cdot)$ is the $sigmoid$ function. W_{sc}^e and W_{sc}^m denote the channel-aware weights for entity-pair and mention-pair fusion, respectively. Entity-pairs and mention-pairs belong to different granularity of relation representations, so the parameters for entity-pairs and mention-pairs are independent and trained parallelly.

2.5 Reasoning Layer

In this section, we propose a multi-granularity logical reasoning module named MGLR based on CNN (Krizhevsky et al., 2012). There are two reasoning blocks in MGLR, one is the mention-pair reasoning block and the other is the entity-pair reasoning block. The structure of the two reasoning blocks is similar and the specific structure is shown in Fig. 4.

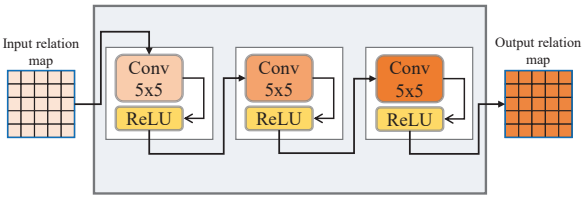

Figure 4: The structure of reasoning block.

The input of the mention-pair reasoning block is the mention-pair representations with rich global context and structure information M_{sc}. We apply the 2D convolution with kernel size 5×5 for reasoning, and use the ReLU activation function after each convolutional layer. In the mention-pair map, each mention-pair can interact with other pairs without the limitation of their position in the document, so the global interaction information of mention-pairs can be achieved only using one layer of convolution. While in the implementation process, we stack three convolution layers for deeper and more adequate fine-grained reasoning. We let M_{rs} as the representation of mention-relation reasoning information, it can be expressed as:

$$M_{rs} = Conv_n^M(M_{sc}), \quad (16)$$

where $Conv_n^M(\cdot)$ denotes the n-layers convolutional network for mention-pair-based reasoning and $M_{rs} \in R^{d_{emb} \times N_m \times N_m}$.

The structure of the entity-pair reasoning block is the same as counterpart mention-pair, and its input is the entity-pair representations with rich global context and structure information E_{sc}. The representation of entity-relation reasoning informa-

tion E_{rs} can be expressed as:

$$E_{rs} = Conv_n^E(E_{sc}), \quad (17)$$

where $Conv_n^E(\cdot)$ denotes the n-layers convolutional network for entity-pair-based reasoning and $E_{rs} \in R^{d_{emb} \times N_e \times N_e}$.

2.6 Classifier Layer

In this section, to classify the relations for a relation instance $r = (e_h, e_t)$, we need to get the representation of r.

First, let r_{rs}^e and r_{rs}^m as the entity-relation and mention-relation reasoning representation of r, respectively. r_{rs}^e is the representation of r in E_{rs}, and r_{rs}^m can be calculated by aggregating all mention-pairs referring to r:

$$r_{rs}^m = avg_{i \in e_h} \left(avg_{j \in e_t} \left(M_{rs[r;i;j]} \right) \right). \quad (18)$$

Then, r_{rs}^e and r_{rs}^m are concatenated as the reasoning representation of r: $r_{rs} = [r_{rs}^e; r_{rs}^m], r_{rs} \in R^{2d_{emb}}$.

Next, to enrich the relation representation, the entity-pair representation before reasoning and the entity representation obtained by R-GCN are also concatenated to represent the relation. The final relation representation can be expressed as:

$$r = [r_{rs}; r_{ht}; r_f], \quad (19)$$

where $r_{ht} = W_s[e_h^s; e_t^s] + b_s$, and $r_f \in R^{d_{emb}}$ denotes the entity-pair representation of r from E_{sc}. e_h^s and e_t^s are the entity representations of e_h and e_t from E_s. $W_s \in R^{2d_{emb} \times 2d_{emb}}$, and b_s are learnable parameters.

Finally, we use a feed-forward neural network to calculate the probability for each relation instance:

$$P(r|e_h, e_t) = \delta(W_b \sigma(W_a r + b_a) + b_b), \quad (20)$$

where W_*, b_* are learnable parameters, $\sigma(\cdot)$ is activation function, and $\delta(\cdot)$ is the $sigmoid$ function. Global thresholding does not consider the variations of model confidence in different instances that results in suboptimal performance, so we adopt an adaptive-thresholding loss as the classification loss to train our model following Zhou et al. (2021). Specifically, they introduce a threshold class TH, and push the logits of all positive classes to be higher than the TH class, and the logits of negative classes to be lower than it:

$$L = -\sum_{r \in P_T} log(\frac{exp(logit_r)}{\sum_{r' \in P_T \cup \{TH\}} exp(logit_{r'})}) - log(\frac{exp(logit_{TH})}{\sum_{r' \in N_T \cup \{TH\}} exp(logit_{r'})}), \quad (21)$$

where $logit$ is the output in the last layer before $sigmoid$ function.

3 Experiments

3.1 Datasets

We evaluate our FILT model on two widely used Bio-DocuRE datasets: CDR (Li et al., 2016) and GDA (Wu et al., 2019). The dataset statistics are shown in Table 1.

Table 1: The statistic results of CDR and GDA datasets.

Dataset		Docs	relations	N/A	Intra-	Inter-
CDR	Train	500	1038	4280	755	283
	Dev	500	1012	4136	766	246
	Test	500	1066	4270	763	303
GDA	Train	23353	36079	96399	30199	5880
	Dev	5839	8762	24362	7408	1354
	Test	1000	1502	3720	1273	229

3.2 Experimental Settings

FILR is developed using PyTorch and based on Huggingface's Transformers (Wolf et al., 2020). The experimental settings are the same on CDR and GDA. We use BioBERT-Base v1.1 (Lee et al., 2020) as the encoder with learning rate 3e-5, and train FILR with learning rate 1e-4 using 3-layers of R-GCN. In FILR, in order to satisfy the requirement for matrix dimensions when performing reasoning, we set batch-max-entity-number and batch-max-mention-number for each batch. And all entity-pairs and mention-pairs representations of documents in the same batch are aligned with padding value 0.

3.3 Results

The baseline models can be divided into non-reasoning models and reasoning models. And each of them can also be divided into sequence-, graph-, or transformer-based methods according to the way of encoding documents.

Results on CDR Dataset. The performances of FILR and baseline models on CDR dataset are shown in Table 2. We can find that FILR achieves 85.7%, 89.1%, and 77.2% in terms of overall, intra-, and inter-F1, which outperforms all existing models and achieves a new state-of-the-art on

Table 2: Experimental results of the FILR on CDR and GDA datasets. All results given in this table are from their related papers. "-" means the data not given in original papers.

Model		CDR F1(%)			GDA F1(%)		
		Overall	Intra-	Inter-	Overall	Intra-	Inter-
non-reasoning model							
CNN+ME+PP(Gu et al., 2017) (Gu et al., 2017)	Sequence-based	61.3	57.2	11.7	-	-	-
BRAN(Verga et al., 2018) (Verga et al., 2018)	Sequence-based	62.1	-	-	-	-	-
EncAttAgg(Jiang et al., 2020) (Jiang et al., 2020)	Sequence-based	64.9	-	-	-	-	-
EoGANE(Tran et al., 2020) (Tran et al., 2020)	Graph-based	66.1	70.7	53.5	82.8	86.3	58.6
GLRE(Wang et al., 2020) (Wang et al., 2020)	Graph-based	68.5	-	-	-	-	-
DAM-GAN(Li et al., 2021) (Li et al., 2021b)	Graph-based	68.6	78.8	56.2	83.6	86.9	63.5
ATLOP(Zhou et al., 2021) (Zhou et al., 2021)	Transformer-based	69.4	-	-	83.9	-	-
SSAN(Xu et al., 2021) (Xu et al., 2021)	Transformer-based	68.7	74.5	56.2	83.7	86.6	65.3
reasoning model							
MRN(Li et al., 2021) (Li et al., 2021a)	Sequence-based	65.9	70.4	54.2	82.9	86.1	53.5
EoG(Christopoulou et al., 2019) (Christopoulou et al., 2019)	Graph-based	63.6	68.2	50.9	81.5	85.2	51.1
LSR(Nan et al., 2020) (Nan et al., 2020)	Graph-based	64.8	68.9	53.1	82.2	85.4	53.1
DHG(Zhang et al., 2020) (Zhang et al., 2020b)	Graph-based	65.9	70.1	54.6	83.1	85.6	58.8
SIRE(Zeng et al., 2021) (Zeng et al., 2021)	Graph-based	70.8	-	-	**84.7**	-	-
CGM2IR(Zhao et al., 2022) (Zhao et al., 2022)	Transformer-based	73.8	79.2	55.1	**84.7**	**88.3**	59.0
FILR(ours)		**85.7**	**89.1**	**77.2**	**84.7**	87.2	**68.9**

CDR dataset. Compared with the previous state-of-the-art model CGM2IR, FILR achieves improvements of 11.9%, 9.9%, and 22.1% in overall, intra- and inter-F1, respectively, which can be attribut to the reasoning of global mention-pairs and the utilization of global structure information of documents. In general, FILR provides rich premise information for logical reasoning through the extraction and fusion of global document information. Meanwhile, the multi-granularity logical reasoning based on entity-pair and mention-pair can effectively model the global interaction among different granularity relation pairs, improving the effect distinctly in all evaluation metrics, especially the inter-sentence.

Results on GDA Dataset. As shown in Table 2, we can see that FILR achieves 84.7% in overall F1, which outperforms most of the existing models and achieves the same result as the state-of-the-art models on GDA dataset. It is important to note that the scale of GDA is nearly 50 times larger than CDR, so GDA is less sensitive to models and the improvements on it in all evaluation metrics are not significant. Compared with CGM2IR, on intra-F1 and inter-F1, we can observe that FILR is 1.1% lower than CGM2IR in intra-F1, but 9.9% higher in inter-F1, which shows that FILR mainly improves the performance of inter-sentence relations. However, since the number of inter-sentence relations only accounts for 16% of the total relations shown in Table 1, the increase in inter-sentence F1 scores fails to bring the same gain in overall F1 scores.

3.4 Ablation Study

Table 3: Ablation study of FILR on CDR and GDA. "o-$*$" refers to the model removing "$*$" module. "o-context" and "o-structure" are the models without using context and structure information of documents, respectively. In "o-MDIF-cat" model, MDIF module is replaced with concat operation. "o-e-r" refers to the model removing entity-pair-based reasoning block.

Model	CDR F1(%)			GDA F1(%)		
	Overall	Intra-	Inter-	Overall	Intra-	Inter-
FILR	85.7	89.1	77.2	84.7	87.2	68.9
o-context	83.4	87.1	75.0	84.0	87.3	63.8
o-structure	84.6	88.8	74.8	83.7	86.8	63.6
o-MDIF-cat	83.6	88.4	73.5	84.2	87.2	66.7
o-e-r	79.1	83.2	70.3	83.4	86.7	63.1
o-MGLR	70.1	74.8	60.1	83.1	86.1	64.2

We conduct a thorough ablation study as shown in Table 3 and Fig. 5 to study the effectiveness of our contributions: MDIF, MGLR modules and the improved document graph. From Table 3 and Fig. 5, we can observe that:

(1) Both global context and structure document information are important for Bio-DocuRE. And the MDIF module is able to integrate the two global documents information effectively by the channel-based weighted fusion, which brings at least 2.1% and 0.5% improvements in terms of the CDR and GDA dataset, respectively.

(2) Coarse-grained entity-pair-based reasoning can get the macro entity-relation reasoning information, which is necessary for the relationship inference of long-distance entity-pairs. At the same time, fine-grained mention-pair-based reasoning as the beneficial supplement to entity-pair-based rea-

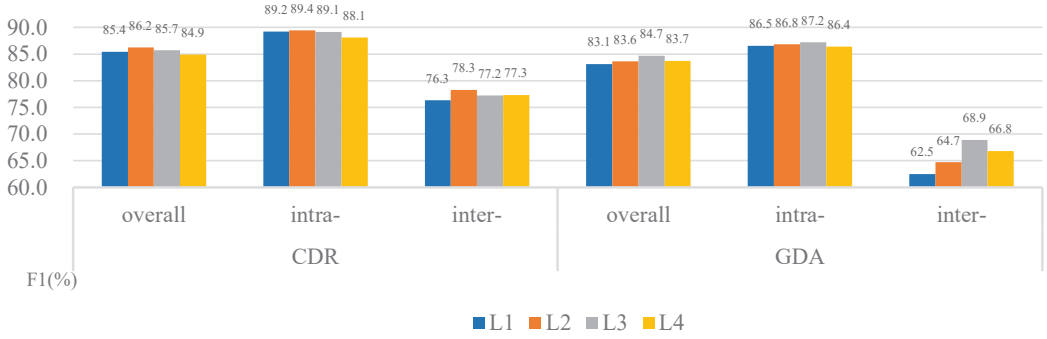

Figure 5: Ablation study of the effect of R-GCN layers.

soning, contributes evidently to both the intra- and inter-sentence relation extraction.

(3) As shown in Fig. 5, the best extraction results are achieved on CDR and GDA dataset with the R-GCN layers of 2 and 3, respectively. And with the increase of the number of RGCN layers, the decline of all evaluation metrics is relatively gentle. This result shows that the model finally achieves optimal performance after the convolution of two layers or three GCN layers, alleviating the over-smoothing problem as well as leveraging the advantages of GCN.

3.5 Case Study

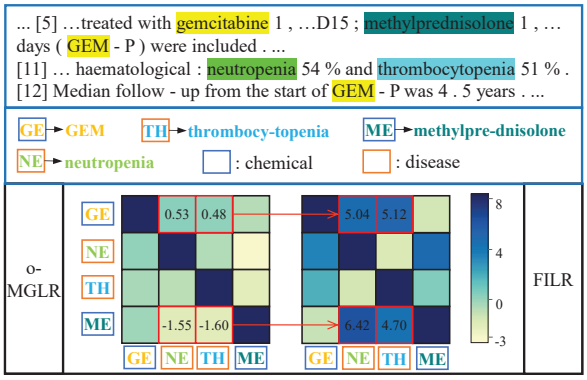

Figure 6: Case study of the multi-granularity reasoning module MGLR effectiveness on CDR dataset.

As shown in Fig. 6, we conduct a case study to further illustrate the effectiveness of the MGLR module. All values in Fig. 6 are the prediction scores of the given relation instances. In this example, *GEM* and *methylprednisolone* are chemicals, *neutropenia* and *thrombocytopenia* are diseases. The "o-MGLR" model incorrectly predicts the classification of relation instance <*methylprednisolone*, *neutropenia*> and <*methylprednisolone*, *thrombocytopenia*> with the prediction scores -1.55 and -1.6, respectively. And the prediction scores of <*GEM*, *neutropenia*> and <*GEM*, *thrombocytopenia*> are 0.53 and 0.48, which indicates the model has insufficient confidence. Whereas, FILR predicts all relation instances correctly. We can see from Fig. 6, the confidence of FILR has been improved evidently with the help of MGLR module. In summary, the MGLR module can capture the global interactions among all entity-pairs and mention-pairs through convolution operation, even if there is no explicit connection between them, which effectively improves the performance of Bio-DocuRE.

4 Related Work

Non-reasoning methods. Previous non-reasoning Bio-DocuRE approaches can be divided into: sequence-based, graph-based, and transformer-based methods. Sequence-based methods (Gu et al., 2017; Verga et al., 2018; Jiang et al., 2020) used CNN or LSTM to extract document-level relations. However, these sequence-based methods do not capture the long-distance dependencies of documents and have limited effect. In order to break through the limitations, researchers tried to construct document graphs (Tran et al., 2020; Wang et al., 2020; Li et al., 2021b). Then, with the development of pre-trained language models, Transformer showed powerful capabilities in capturing context information, and is increasingly used in Bio-DocuRE task (Zhou et al., 2021; Xu et al., 2021).

Reasoning methods. Many researchers have tried to introduce various reasoning mechanisms to reinforce the ability of model in intra- and inter-sentence relation extraction. Li et al. (2021a) proposed a mention-based module for multi-hop reasoning, and introduced a co-attention mechanism for global mention reasoning. Christopoulou et al. (2019) implemented inference with an iterative algorithm. Nan et al. (2020) and Zhang et al. (2020b)

constructed a document graph and used GCN on the graph to employ multi-hop reasoning. Zeng et al. (2021) designed a logical reasoning module that can cover more logical reasoning chains. Zhao et al. (2022) executed entity-pair level logical reasoning using GNN-based methods.

5 Conclusion

In this paper, we propose a novel Bio-DocuRE model called FILR. FILR first acquires the global context and structure information of documents. Then, the MDIF module is proposed to fuse the two document information effectively. Next, we propose the MGLR module to conduct entity-pair-based and mention-pair-based reasoning parallelly. Last, we evaluate our FILR model on two widely used biomedical datasets CDR and GDA and the experimental results show that FILR achieves state-of-the-art on two datasets.

Acknowledgements

This work is supported by grant from the National Natural Science Foundation of China (No. 62076048), the Science and Technology Innovation Foundation of Dalian (2020JJ26GX035).

References

Fenia Christopoulou, Makoto Miwa, and Sophia Ananiadou. 2019. Connecting the dots: Document-level neural relation extraction with edge-oriented graphs. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 4925–4936.

Yimian Dai, Fabian Gieseke, Stefan Oehmcke, Yiquan Wu, and Kobus Barnard. 2021. Attentional feature fusion. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, pages 3560–3569.

Jinghang Gu, Fuqing Sun, Longhua Qian, and Guodong Zhou. 2017. Chemical-induced disease relation extraction via convolutional neural network. *Database: The Journal of Biological Databases and Curation*, 2017:bax024.

Xiyue Guo, Tingting He, and Jie Yuan. 2015. Relation dictionary construction and rule learning for ppi extraction from biomedical literatures. In *2015 IEEE International Conference on Bioinformatics and Biomedicine (BIBM)*, pages 1133–1140. IEEE.

Robin Jia, Cliff Wong, and Hoifung Poon. 2019. Document-level n-ary relation extraction with multiscale representation learning. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 3693–3704.

Ping Jiang, Xian-Ling Mao, Binbin Bian, and Heyan Huang. 2020. Improving document-level relation extraction via contextualizing mention representations andweighting mention pairs. In *2020 IEEE International Conference on Knowledge Graph (ICKG)*, pages 305–312. IEEE.

Zhenchao Jiang, Lishuang Li, and Degen Huang. 2016. A general protein-protein interaction extraction architecture based on word representation and feature selection. *International Journal of Data Mining and Bioinformatics*, 14(3):276–291.

Thomas N Kipf and Max Welling. 2016. Semi-supervised classification with graph convolutional networks. *arXiv preprint arXiv:1609.02907*.

Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. 2012. Imagenet classification with deep convolutional neural networks. *Advances in neural information processing systems*, 25:1106–1114.

Jinhyuk Lee, Wonjin Yoon, Sungdong Kim, Donghyeon Kim, Sunkyu Kim, Chan Ho So, and Jaewoo Kang. 2020. Biobert: a pre-trained biomedical language representation model for biomedical text mining. *Bioinformatics*, 36(4):1234–1240.

Jiao Li, Yueping Sun, Robin J Johnson, Daniela Sciaky, Chih-Hsuan Wei, Robert Leaman, Allan Peter Davis, Carolyn J Mattingly, Thomas C Wiegers, and Zhiyong Lu. 2016. Biocreative v cdr task corpus: a resource for chemical disease relation extraction. *Database: The Journal of Biological Databases and Curation*, 2016:baw068.

Jingye Li, Kang Xu, Fei Li, Hao Fei, Yafeng Ren, and Donghong Ji. 2021a. Mrn: A locally and globally mention-based reasoning network for document-level relation extraction. In *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*, pages 1359–1370.

Lishuang Li, Ruiyuan Lian, and Hongbin Lu. 2021b. Document-level biomedical relation extraction with generative adversarial network and dual-attention multi-instance learning. In *2021 IEEE International Conference on Bioinformatics and Biomedicine (BIBM)*, pages 438–443. IEEE.

Qimai Li, Zhichao Han, and Xiao-Ming Wu. 2018. Deeper insights into graph convolutional networks for semi-supervised learning. In *Proceedings of the Thirty-Second AAAI Conference on Artificial Intelligence (AAAI-18)*, pages 3538–3545. Association for the Advancement of Artificial Intelligence.

Zhiheng Li, Zhihao Yang, Yang Xiang, Ling Luo, Yuanyuan Sun, and Hongfei Lin. 2020. Exploiting sequence labeling framework to extract document-level relations from biomedical texts. *BMC bioinformatics*, 21(1):1–14.

Xiaofeng Liu, Jianye Fan, Shoubin Dong, et al. 2020. Document-level biomedical relation extraction leveraging pretrained self-attention structure and entity replacement: Algorithm and pretreatment method validation study. *JMIR Medical Informatics*, 8(5):e17644–e17644.

Sitao Luan, Mingde Zhao, Xiao-Wen Chang, and Doina Precup. 2019. Break the ceiling: stronger multi-scale deep graph convolutional networks. In *Proceedings of the 33rd International Conference on Neural Information Processing Systems*, pages 10945–10955.

Guoshun Nan, Zhijiang Guo, Ivan Sekulić, and Wei Lu. 2020. Reasoning with latent structure refinement for document-level relation extraction. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 1546–1557.

Dat Quoc Nguyen and Karin Verspoor. 2018. Convolutional neural networks for chemical-disease relation extraction are improved with character-based word embeddings. *Proceedings of the BioNLP 2018 workshop*, pages 129–136.

Michael Schlichtkrull, Thomas N Kipf, Peter Bloem, Rianne van den Berg, Ivan Titov, and Max Welling. 2018. Modeling relational data with graph convolutional networks. In *European semantic web conference*, pages 593–607. Springer.

Hieu Minh Tran, Minh Trung Nguyen, and Thien Huu Nguyen. 2020. The dots have their values: exploiting the node-edge connections in graph-based neural models for document-level relation extraction. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 4561–4567.

Patrick Verga, Emma Strubell, and Andrew McCallum. 2018. Simultaneously self-attending to all mentions for full-abstract biological relation extraction. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 872–884.

Difeng Wang, Wei Hu, Ermei Cao, and Weijian Sun. 2020. Global-to-local neural networks for document-level relation extraction. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 3711–3721.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Rémi Louf, Morgan Funtowicz, et al. 2020. Transformers: State-of-the-art natural language processing. In *Proceedings of the 2020 conference on empirical methods in natural language processing: system demonstrations*, pages 38–45.

Ye Wu, Ruibang Luo, Henry Leung, Hing-Fung Ting, and Tak-Wah Lam. 2019. Renet: A deep learning approach for extracting gene-disease associations from literature. In *International Conference on Research in Computational Molecular Biology*, pages 272–284. Springer.

Wuti Xiong, Fei Li, Hong Yu, and Donghong Ji. 2019. Extracting drug-drug interactions with a dependency-based graph convolution neural network. In *2019 IEEE International Conference on Bioinformatics and Biomedicine (BIBM)*, pages 755–759. IEEE.

Benfeng Xu, Quan Wang, Yajuan Lyu, Yong Zhu, and Zhendong Mao. 2021. Entity structure within and throughout: Modeling mention dependencies for document-level relation extraction. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 35, pages 14149–14157.

Shuang Zeng, Yuting Wu, and Baobao Chang. 2021. Sire: Separate intra-and inter-sentential reasoning for document-level relation extraction. In *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*, pages 524–534.

Jinyong Zhang, Weizhong Zhao, Jincai Yang, Xingpeng Jiang, and Tingting He. 2020a. An effective framework for document-level chemical-induced disease relation extraction via fine-grained interaction between contexts. In *2020 IEEE International Conference on Bioinformatics and Biomedicine (BIBM)*, pages 391–396. IEEE.

Zhenyu Zhang, Bowen Yu, Xiaobo Shu, Tingwen Liu, Hengzhu Tang, Wang Yubin, and Li Guo. 2020b. Document-level relation extraction with dual-tier heterogeneous graph. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 1630–1641.

Chao Zhao, Daojian Zeng, Lu Xu, and Jianhua Dai. 2022. Document-level relation extraction with context guided mention integration and inter-pair reasoning. *arXiv preprint arXiv:2201.04826*.

Lingxiao Zhao and Leman Akoglu. 2019. Pairnorm: Tackling oversmoothing in gnns. *arXiv preprint arXiv:1909.12223*.

Wei Zheng, Hongfei Lin, Ling Luo, Zhehuan Zhao, Zhengguang Li, Yijia Zhang, Zhihao Yang, and Jian Wang. 2017. An attention-based effective neural model for drug-drug interactions extraction. *BMC bioinformatics*, 18(1):1–11.

Huiwei Zhou, Huijie Deng, Long Chen, Yunlong Yang, Chen Jia, and Degen Huang. 2016. Exploiting syntactic and semantics information for chemical–disease relation extraction. *Database: The Journal of Biological Databases and Curation*, 2016:baw048.

Wenxuan Zhou, Kevin Huang, Tengyu Ma, and Jing Huang. 2021. Document-level relation extraction with adaptive thresholding and localized context pooling. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 35, pages 14612–14620.

Simple yet Powerful: An Overlooked Architecture for Nested Named Entity Recognition

Matías Rojas[1,2], **Felipe Bravo-Marquez**[1,3,4], **and Jocelyn Dunstan**[2,5,6]
[1]Department of Computer Science, University of Chile.
[2]Center for Mathematical Modeling (CMM), University of Chile.
[3]National Center for Artificial Intelligence (CENIA), Chile.
[4]Millennium Institute for Foundational Research on Data (IMFD), Chile.
[5]Initiative for Data & Artificial Intelligence, University of Chile.
[6]Millenium Institute for Intelligent Healthcare Engineering (IHEALTH), ANID, Chile.

Abstract

Named Entity Recognition (NER) is an important task in Natural Language Processing that aims to identify text spans belonging to predefined categories. Traditional NER systems ignore nested entities, which are entities contained in other entity mentions. Although several methods have been proposed to address this case, most of them rely on complex task-specific structures and ignore potentially useful baselines for the task. We argue that this creates an overly optimistic impression of their performance. This paper revisits the Multiple LSTM-CRF (MLC) model, a simple, overlooked, yet powerful approach based on training independent sequence labeling models for each entity type. Extensive experiments with three nested NER corpora show that, regardless of the simplicity of this model, its performance is better or at least as well as more sophisticated methods. Furthermore, we show that the MLC architecture achieves state-of-the-art results in the Chilean Waiting List corpus by including pre-trained language models. In addition, we implemented an open-source library that computes task-specific metrics for nested NER. The results suggest that metrics used in previous work do not measure well the ability of a model to detect nested entities, while our metrics provide new evidence on how existing approaches handle the task.

1 Introduction

Named Entity Recognition (NER) is a widely studied task in Natural Language Processing (NLP) that seeks to identify text spans expressing references to predefined categories such as person names, locations, and organizations (Chinchor and Robinson, 1997). NER, or in general, the task of recognizing entity mentions[1], has drawn the attention of the community due to its relevance in several NLP applications. Nested Named Entity Recognition is a particular case of NER where entities are nested within each other (Finkel and Manning, 2009). Traditional NER models simplify the nested entities by keeping the outermost entity and removing the inner ones. This simplified problem is better known as flat NER and is commonly regarded as a sequence labeling problem since each token can be associated with at most one label. However, removing part of these entities could be a problem in model performance due to losing relevant information and inner dependencies.

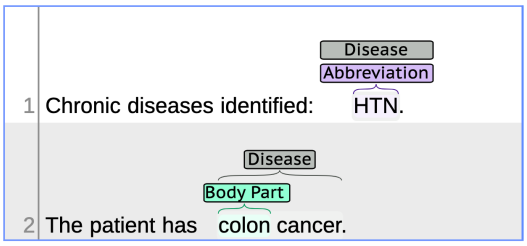

Figure 1: An example of a multi-label entity in the Chilean Waiting List corpus, followed by a nesting of different types. The annotation was translated from its original language.

Several methods have been proposed to address the nesting problem. Traditional approaches have focused on creating representations of nested entities through structures such as hypergraphs (Lu and Roth, 2015; Muis and Lu, 2017; Katiyar and Cardie, 2018; Wang and Lu, 2018). However, they usually suffer from heavy feature engineering, structural ambiguity, or complex models. Another category is region-based, which divides the problem into two sequential stages. First, the detection of entity boundaries, and then the assignment of entity types to these regions (Sohrab and Miwa, 2018; Zheng et al., 2019; Yu et al., 2020). One of the main drawbacks of this method is its high time complexity. There are also approaches that attempt to transform the nested NER task into a sequence labeling problem (Alex et al., 2007; Ju et al., 2018; Shibuya and Hovy, 2020). Although these studies have shown

[1]Mentions are defined as references to entities that could be named, nominal or pronominal (Florian et al., 2004).

Proceedings of the 29th International Conference on Computational Linguistics, pages 2108–2117
October 12–17, 2022.

competitive performance, most of them have three critical issues discussed below.

First, with the incorporation of large pre-trained language models, the standard LSTM-CRF (Lample et al., 2016) sequence labeling architecture received substantial improvements for flat NER tasks (Liu et al., 2018). However, little research has been conducted on adapting this architecture to nested NER using a single entity approach, i.e., training independent flat NER models for each entity type. This paper revisits this architecture, naming it Multiple LSTM-CRF (MLC). We show that this model yields very positive results despite the apparent simplicity, outperforming several recent approaches explicitly designed for nested entities.

Second, we note that most of the literature ignores the case in which the same text span is tagged with more than one entity type, as shown in Figure 1. This case is very common in the Chilean Waiting List corpus (Báez et al., 2020), and it was first noticed by Alex et al. (2007) but was not analyzed further. One of the main advantages of our architecture is that it addresses this problem.

Third, we argue that the way the community is evaluating this task does not adequately measure the effectiveness of a model at identifying nested entities. Specifically, the current metric calculates the micro F1-score over all entities, the same metric used in flat NER. Consequently, a model that performs well over flat entities but not nested ones may also obtain good results. To alleviate this problem, we first identify the different types of nesting by formalizing the task, and then we propose new task-specific metrics for these cases.

In summary, the main contributions of our work are the following:

- Due to the lack of an agreed-upon definition of nested NER, we introduce a formalization of the task by identifying the different types of nesting. In addition, we released an open-source library for computing new task-specific metrics for nested NER.

- We conduct an empirical study comparing several nested NER architectures in three datasets from different languages, with particular attention to the impact of using pre-trained language models and nesting metrics. Experimental results confirm the effectiveness of the MLC model, achieving state-of-the-art in the Chilean Waiting List corpus and competitive performance in the rest of the corpora.

2 Related Work

In recent years there has been a growing interest from the research community in nested NER. Several studies have been conducted to address nested entities, which can be mainly divided into three categories:

Region-based: These approaches divide the problem into two stages: identifying entity boundaries and then categorizing these regions. Sohrab and Miwa (2018) designed a model that enumerates all possible spans within a limited length and then used boundary and average internal token representation to predict entity types. Another region-based model was proposed by Zheng et al. (2019), which uses a sequence labeling layer to detect entity boundaries, and then classifies selected regions into their categorical types. Yu et al. (2020) used ideas from a biaffine model, scoring all possible start-end tokens in a sentence to predict nested entities. Although these methods have proven to be effective, they often suffer from high time complexity and fail to identify entities tagged with more than one type.

Structure-based: There have also been attempts to capture the structure of nested entities. Finkel and Manning (2009) represented each input sentence as a constituency tree of nested entities and used a CRF-based approach to predict entity types. Lu and Roth (2015) proposed a mention hypergraph representation to extract entity mentions. Next, Muis and Lu (2017) improved on previous work by modeling nested NER with mention separators and handcrafted features. Similarly, Katiyar and Cardie (2018) designed a directed hypergraph using LSTM features to learn the nesting structure. Wang et al. (2020) recursively introduce the embedding of tokens and regions into flat NER layers simulating the shape of a pyramid. However, these approaches usually suffer from spurious structures and structural ambiguities, as explained in Wang and Lu (2018).

Sequence labeling-based: Some studies report that sequence labeling methods can also perform well on this task. Early work mainly exploited the potential of conditional random fields (CRF). Alex et al. (2007) proposed three CRF-based methods to reduce the nested NER as several BIO tagging problems. Their best approach, called cascaded CRF, uses one model per entity type by using the output of the previous flat NER model as a feature for the current one. Ju et al. (2018) took advantage

of inner entity information to encourage outer entity recognition. They dynamically stacked LSTM-CRF layers predicting entities inside-to-outside until no entities were extracted. Straková et al. (2019) formulated the nested NER task as a sequence-to-sequence problem using an LSTM to decode entity types. Finally, using a recursive CRF-based method, Shibuya and Hovy (2020) recognized entities iteratively from outermost ones to inner ones. The MLC approach falls into this category by using a sequence labeling approach capable of handling both nested entities and entities tagged with more than one label.

3 Methods

3.1 Problem Definition

One of the main issues in our knowledge of nested NER is that the task definition has not been addressed in-depth, and clarification of the different nesting cases is needed. By analyzing several corpora with nested entities, we have identified the following nesting cases:

Multi-label entities (ME): This case has been little explored in the literature. As explained in Alex et al. (2007), it consists of entities tagged with more than one entity type. With the release of the Chilean Waiting List corpus, it is interesting to study this case since 10.75% of the entities are involved in this type of nesting. For example, the entity "HTN", which stands for hypertension, is tagged as a disease and an abbreviation.

Nested entities of different types (NDT): This is the most frequent type of nesting in nested NER datasets. It consists of an entity containing a shorter entity tagged with a different type. An example is "colon cancer", where a body part (colon) is contained in a disease.

Nested entities of the same type (NST): This case usually occurs when entities are originally represented by a hierarchy, which is later pruned to reduce the entity space, resulting in the merging of entities of different levels of granularity. Although it appears in most corpora, it is much more frequent in GENIA (Kim et al., 2003). For example, the DNA "Drosophila homeodomain" contains another DNA, "homeodomain".

To better understand these cases, we formally define what we mean by nested entities and the nested NER task.

Definition 1 (Nested entities) *Given an input sequence $X = \{x_1, x_2, ..., x_n\}$ of words, an entity Q is defined by a tuple (S_q, E_q, T_q), where S_q and $E_q \in [1, n]$ represents entity boundaries in X, and T_q in $\mathcal{E}$ (the entity space) corresponds to the entity type. Given two entities Q and R, we say that Q is nested in R if $S_r \leq S_q$ and $E_q \leq E_r$. The particular case of $S_q = S_r$ and $E_q = E_r$ corresponds to an entity with multiple labels.*

Definition 2 (Nested NER) *Given an input sequence $X = \{x_1, x_2, ..., x_n\}$, nested NER aims to correctly identify the boundaries for every entity Q in X and assign it the correct entity type from a predefined list of categories. This identification must be made for cases where nested entities are involved and when not.*

3.2 Model

With advances in deep learning, sequence labeling architectures have received substantial improvements in the NER task in recent years. Therefore, we decided to revisit a method that belongs to this category but, despite its effectiveness, has been little studied.

Multiple LSTM-CRF (MLC): This approach consists of training multiple flat NER models, one for each entity type. The predicted labels of the input sentences correspond to the union of the outputs of each of these models, thus retrieving both nested entities and entities tagged with multiple labels. The main advantage of this approach is that it can easily incorporate all the progress made for the flat NER task into the nested NER task.

The apparent simplicity of MLC would lead us to believe that it should be considered a natural baseline for any proposed architecture in nested NER. However, few papers have used this approach as a baseline (Muis and Lu, 2017; Lin et al., 2019; Fei et al., 2020) and their results were not competitive. We believe the problem lies in their failure to use the potential of recent advances in flat NER architectures, such as the addition of pre-trained language models to create contextualized embeddings.

Figure 2 shows an overview of the MLC model. Specifically, to create each flat NER module, we follow the LSTM-CRF approach proposed by Lample et al. (2016), one of the most widely used architectures for sequence labeling. To encode sentences, we use different combinations of embeddings in the stacked embedding layer. First, we concatenate domain-specific word embeddings with character embeddings retrieved from a bidi-

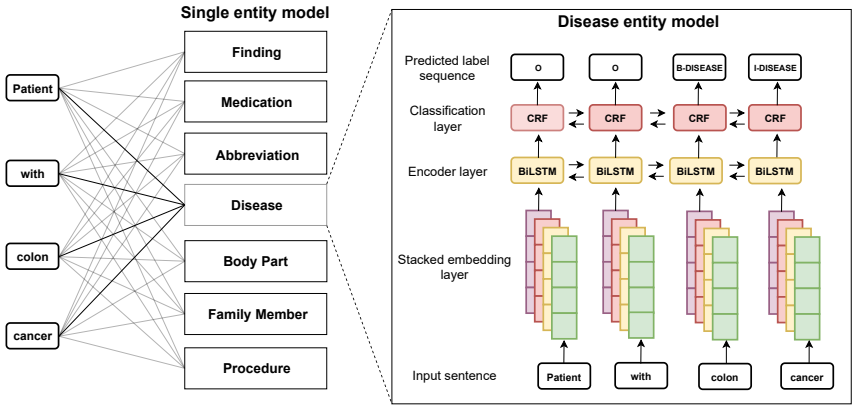

Figure 2: Overview of the MLC architecture, where each entity type has an associated flat NER model. The right side of the figure shows, as an example, the flat NER module for the Disease tag in the Chilean Waiting List dataset.

rectional character-level LSTM. Next, we enrich word representations by adding contextualized embeddings from Flair (Akbik et al., 2018) and BERT (Devlin et al., 2019), which have proven to be particularly effective on NER. The output is fed into a BiLSTM encoding layer to obtain long-contextual information. Finally, we use a CRF-loss and the Viterbi algorithm to decode the most likely tag sequence using the IOB2 tagging format.

4 Experiments

In this section, we present the datasets, baselines, and settings used in our experiments.

4.1 Datasets

Since most previous work on nested NER has been done in English datasets, we conducted our experiments with three corpora containing nested entities for three different languages and domains. The statistics for each corpus are shown in Table 1.

GENIA V3.02[2] **(Kim et al., 2003)** English biomedical corpus created from 2,000 MEDLINE abstracts. It comprises 36 fine-grained entity types and 55,740 entity mentions, of which 17.3% are involved in nesting. We followed the same setup as the previous work (Finkel and Manning, 2009; Lu and Roth, 2015; Zheng et al., 2019), collapsing sub-types into their five super-types, using the first 90% of the sentences for the training set and the remaining 10% in the test set.

GermEval 2014[3] **(Benikova et al., 2014)** German dataset sampled from German Wikipedia and German online news. It consists of 41,124 entity mentions, where 14.9% of them are involved in nesting. The corpus contains two levels of nesting and 12 entity types.

Chilean Waiting List[4] **(Báez et al., 2020)** Spanish clinical corpus created from real diagnoses of the Chilean healthcare system. It is composed of 43,730 entity mentions and seven entity types. From a nested NER point of view, it is a good resource since 46.7% of the entities are involved in nesting.

Studying previous work, we have noticed that comparisons between models are not entirely fair since the data partitions used vary between different papers. Therefore, for a fair comparison, in both the GENIA and GermEval datasets, we trained the models using the preprocessed version released in Zheng et al. (2019). In the case of the Chilean corpus, we used the public files released by the authors, which are already tokenized.

4.2 Baselines

We compare our results with several state-of-the-art models in GENIA and GermEval. Table 2 shows the different types of nesting that each of these baselines is capable of addressing. Based on the released source code, we have reproduced the following models to use as a reference for analyzing both traditional and task-specific metrics:

Pyramid is a structure-based architecture that recognizes entities in a bottom-up manner, from the shortest to the longest, assimilating the shape of a pyramid. It is currently the state-of-the-art method without using external supervision (Wang et al., 2020).

[2] http://www.geniaproject.org/genia-corpus/pos-annotation

[3] https://sites.google.com/site/germeval2014ner/data

[4] https://zenodo.org/record/5591011

	GENIA			GermEval			Chilean Waiting List		
	Train	Test	Dev	Train	Test	Dev	Train	Test	Dev
tokens	454,882	57,021	48,932	452,853	96,499	41,653	149,574	18,436	16,754
sentences	15,023	1,854	1,669	24,000	5,100	2,200	8,014	990	890
avg sent len	30.3	30.8	29.3	18.9	18.9	18.9	18.7	18.6	18.8
entities	45,929	5,474	4,337	31,545	6,693	2,886	35,480	4,289	3,971
avg entity len	2.9	2.9	3.1	1.4	1.4	1.5	2.6	2.7	2.6
nested entities (%)	17.0	20.6	16.8	15.0	14.7	14.1	46.4	45.9	46.7
nested entities	7,795	1,130	727	4,721	986	407	16,456	1,969	1,856
- different type	3,712	589	369	4,230	892	366	12,635	1,555	1,398
- same type	4,132	547	358	536	93	44	0	0	0
- multi-label entities	0	0	0	2	2	0	4,241	470	502

Table 1: Statistics of the datasets.

Model	ME	NDT	NST
Layered	✓	✓	✓
Exhaustive	✗	✓	✓
Boundary	✗	✓	✓
Biaffine	✗	✓	✓
Pyramid	✓	✓	✓
Recursive-CRF	✓	✓	✓
MLC	✓	✓	✗

Table 2: Nesting types identified by the architectures used in our experiments. Multi-label entities (ME), nesting of different types (NDT), and nesting of the same type (NST).

Recursive-CRF is a sequence labeling-based approach that extracts nested entities iteratively in an outside-to-inside way using a recursive CRF-based algorithm (Shibuya and Hovy, 2020).

Layered is a sequence labeling-based model designed to identify nested entities by dynamically stacking LSTM-CRF layers. It predicts entities in an inside-to-outside way until no more entities are extracted. (Ju et al., 2018).

Exhaustive is a region-based model that enumerates all possible regions as potential entity mentions and then classifies them into their entity types (Sohrab and Miwa, 2018).

Boundary is a region-based method that combines ideas from the Layered and Exhaustive models. It uses a BiLSTM layer to detect boundary-relevant regions and then uses these representations to predict categorical entity labels (Zheng et al., 2019).

Biaffine is a region-based architecture that leverages contextualized paragraph-level embeddings to create a Biaffine model. This approach scores candidate pairs of start and end tokens in a sequence and then classifies them into predefined categories using nested entities constraints (Yu et al., 2020).

4.3 Implementation Details

Pre-trained Word Embeddings. To encode sentences, we selected pre-trained word embeddings in the same domain of each corpus. For the experiments with GENIA, we used biomedical embeddings trained on MEDLINE abstracts (Chiu et al., 2016). In GermEval, we incorporated German FastText embeddings (Grave et al., 2018), and for the Chilean dataset, we used pre-trained embeddings from a large clinical corpus, which can be downloaded from here[5]. During the training process, the embeddings were not left static.

Contextual Word Embeddings. To study the impact of adding pre-trained language models, we used BERT (Devlin et al., 2019), and Flair (Akbik et al., 2018), which is a character-level language model. In the case of BERT, we did not perform fine-tuning. The embeddings were calculated by averaging the representations retrieved from hidden states. Since it uses WordPiece tokenization, we computed word embeddings using the average of subtoken embeddings.

A version of these models was available for all the languages and domains involved in our study, except for Spanish. Therefore, we added new language models in the Spanish clinical domain to the Flair framework (Rojas et al., 2022). We trained these models on the same corpus as the word embeddings used for the Chilean dataset, following the same settings and assumptions reported in the Flair paper. The model reached a final perplexity value of 1.61.

The Biaffine model computed the BERT embeddings using the paragraph-level context. Fu et al. (2021) explains that this method provides better performance in resolving correlations, so it is not an entirely fair comparison with models that use sentence-level context. For this reason, we do not

[5] http://doi.org/10.5281/zenodo.3924799

Parameter	Range	MLC
max epochs	[20, 100]	100
optimizer	[SGD, Adam, AdamW]	SGD
batch size	[8, 32]	16
learning rate	[0.0001, 0.1]	0.1
char emb dim	[20, 50]	25
dropout	[0.2, 0.8]	0.3
BiLSTM depth	[1, 3]	3
BiLSTM hidden size	[128, 512]	128

Table 3: Hyperparameter search space and the best values found for the MLC model.

make a comprehensive comparison with this model in terms of contextualized embeddings.

Parameters. We used a unified setting for all the experiments with MLC. The best hyperparameters were chosen by performing a random search over the range of values shown in Table 3, selecting the best configuration based on performance on the development set. To perform a fair comparison with our baselines, we used the best hyperparameters reported in their papers.

We trained the MLC architecture using the SGD optimizer to 100 epochs, with mini-batches of size 16 and a learning rate of 0.1. To control the overfitting problem, we employed a learning rate scheduler and an early stopping strategy. We also applied dropout regularization (Srivastava et al., 2014) after the embedding layer and BiLSTM. The MLC model was implemented using the Flair framework (Akbik et al., 2019), and the rest of the baselines were executed with the official code provided by the authors. All the experiments were performed using a Tesla V100 GPU. Training the MLC model on the Chilean Waiting List took 7 hours to get an idea of the computational cost of our approach. The source code of our system is freely available to reproduce the experiments[6].

4.4 Evaluation Metrics

Overall Performance. Performance was evaluated using precision, recall, and micro F1-score, which is the standard metric used in nested NER. An entity is considered correct when both entity types and boundaries are predicted correctly.

Nested Performance. Since flat entities are much more common than nested entities, the standard metric ends up confusing flat and nested results and, consequently, is not able to reflect well the ability of a model to detect nesting. To alleviate this issue, we analyze task-specific metrics proposed in previous work that adequately measure the model's ability to detect nested and non-nested entities. Precisely, we compute scores for the following cases: non-nested entities (m_{flat}), nested entities (m_{nested}), inner entities (m_{inner}) and outer entities (m_{outer}). We consider an entity nested if it contains any entity or is contained within another entity mention. Thus, the m_{nested} metric considers both m_{inner} and m_{outer} scores.

However, none of these metrics capture the ability of the models to recognize both inner and outer entities simultaneously. For this reason, and to demonstrate whether the choice of a model in a dataset depends on the types of nesting present, we compute a score for nesting ($m_{nesting}$) and on the different types of nesting described in the task formalization (m_{ME}, m_{NDT}, m_{NST}). A nesting is considered correct if both inner and outer entities are recognized correctly.

The above metrics are calculated using precision, recall, and micro F1-score, but we only report the last one for brevity. We emphasize that most of these metrics have not been used before in nested NER research. Therefore, we believe it is crucial to incorporate them in future work as it allows us to measure and differentiate the performance of models on nested and non-nested entities. Due to this, we implemented an open-source library[7] that computes these metrics.

4.5 Main Results

Table 4 shows the overall performance of the proposed model against baselines on three different datasets. Despite its simplicity, the MLC architecture outperforms existing state-of-the-art models on the Chilean Waiting List by +1.6 in terms of the F1 measure. By contrast, although state-of-the-art is not obtained in GENIA and GermEval, we can see that MLC outperforms many specialized nested NER architectures, thus being a competitive approach. One possible reason for the excellent performance is that we use one model per entity type, which means that the number of possible labels is only one per model, avoiding the problem of nested entities and making the classification step more straightforward compared to other architectures. Compared with the statistics in Table 1, we can conclude that it is more challenging to obtain good results when the corpora have entities of a

[6] https://github.com/matirojasg/nested-ner-mlc

[7] https://github.com/matirojasg/nestednereval

Model	GENIA			GermEval			Chilean Waiting List		
	P	R	F1	P	R	F1	P	R	F1
Layered	73.9	68.7	71.2	71.8	64.1	67.7	75.0	72.8	73.9
Exhaustive	74.1	69.7	71.8	78.6	64.6	70.9	76.3	71.7	68.2
Boundary	76.7	71.8	74.2	74.4	65.5	69.7	74.0	67.6	70.7
Pyramid	78.1	72.8	75.3	77.8	66.9	71.9	79.6	75.4	77.5
Biaffine	79.1	73.7	76.3	89.0	77.4	82.8	81.5	67.1	73.6
Recursive-CRF	75.8	75.2	75.5	85.1	78.2	81.5	75.1	77.2	76.1
MLC	77.6	74.2	75.8	86.8	77.2	81.7	77.7	78.3	78.0
LM-based									
Biaffine [BERT]	79.9	76.5	**78.1**	88.3	85.0	86.6	78.7	70.8	74.5
Recursive-CRF									
- Flair	77.1	78.0	77.6	83.4	82.9	83.2	78.0	79.9	78.9
- BERT	76.4	77.4	76.9	84.3	83.0	83.6	76.6	77.8	77.2
- Flair + BERT	77.4	76.8	77.1	84.8	82.1	83.4	77.1	77.9	77.5
Pyramid									
- Flair	77.8	75.6	76.7	83.4	80.0	81.7	80.1	77.2	78.6
- BERT	79.1	76.9	78.0	87.7	85.8	**86.7**	78.0	73.6	75.7
- Flair + BERT	80.4	75.0	77.6	87.7	84.4	86.0	78.5	77.2	77.9
MLC									
- Flair	80.1	75.2	77.6	85.3	82.4	83.8	80.6	80.5	**80.5**
- BERT	79.4	74.3	76.8	85.1	80.3	82.6	79.7	78.8	79.3
- Flair + BERT	78.8	75.2	75.5	84.7	80.1	82.3	79.9	78.1	79.0

Table 4: Overall results on three nested NER datasets.

more considerable length. This can be explained by the strict metric we are using, where the boundaries and the entity types are requested to match.

We further analyze the effect of adding pre-trained language models in our experiments. As we believed, all models benefit from incorporating contextual word embeddings, improving their performance compared to their base version. In GermEval, a general-purpose corpus, the language model that best improves the model's performance is BERT, while in the other corpora, it is Flair. Also, we can see that stacking Flair and BERT embeddings does not produce better results. We attribute this to the high dimensionality of these representations and the fact that the two language models were trained on different corpora.

Regarding the Chilean corpus, which contains the highest percentage of nested entities, we observe that the MLC model with Flair embeddings improves by +2.5 compared to its base version without pre-trained language models. This demonstrates the effectiveness of using Flair over BERT in this corpus. We suspect that it is due to the large number of misspelled and out-of-vocabulary words found in the unstructured clinical text. As pointed out in Akbik et al. (2018), handling these types of words is one of the main advantages when using its character-level language model.

Despite the promising results, we hypothesize that benchmarking against the standard nested NER metric may not be a good indicator of model performance on nesting since most of the entities are not nested. Therefore, we analyze the results using nested metrics.

4.6 Nested Results

In most cases, the revisited nested metrics presented in Table 5 are relatively consistent with results in Table 4. This means that models which obtain state-of-the-art using the standard metrics also perform well according to these metrics. For example, in the Chilean Waiting List, the best model (MLC) achieves the best results according to the m_{flat}, m_{inner}, m_{outer}, m_{nested} metrics, which is a remarkable result considering a large number of nestings present in this corpus. Another observation is that, unlike the other datasets, GENIA is more complex to recognize inner entities over the outermost ones. This finding could be helpful when designing future architectures for this corpus.

As expected, the models with better performance according to the standard metric are also associated with good results using the m_{flat} metric. However, this may not be a good indicator in the nested NER task since most of the entities in these corpora are not nested, and the proper performance on nestings is not reflected. This issue becomes much more evident when analyzing our proposed nesting metrics, presented in Table 6. Interestingly, we observe that the results are significantly lower than those for the previous metrics of Tables 4 and 5. This reveals the difficulty of correctly recognizing the

GENIA				
Model	m_{flat}	m_{nested}	m_{inner}	m_{outer}
Layered	73.2	62.3	42.9	79.8
Exhaustive	76.6	55.0	42.6	67.9
Boundary	77.4	59.5	42.0	75.6
Biaffine [BERT]	81.2	**65.8**	**49.3**	80.5
Pyramid [BERT]	81.1	65.2	46.1	**82.4**
Recursive-CRF [Flair]	**81.5**	62.3	46.9	77.4
MLC [Flair]	80.7	63.8	41.7	82.2

GermEval				
Model	m_{flat}	m_{nested}	m_{inner}	m_{outer}
Layered	68.8	60.9	62.0	59.7
Exhaustive	73.4	56.1	65.7	45.7
Boundary	70.9	54.5	54.1	55.0
Biaffine [BERT]	88.4	76.6	**78.1**	75.0
Pyramid [BERT]	**88.5**	**76.7**	77.3	**76.1**
Recursive-CRF [BERT]	85.5	73.0	74.9	71.0
MLC [Flair]	86.0	71.6	74.5	68.4

Chilean Waiting List				
Model	m_{flat}	m_{nested}	m_{inner}	m_{outer}
Layered	73.4	74.5	82.4	64.5
Exhaustive	71.7	63.8	71.5	53.4
Boundary	73.4	61.1	65.5	55.4
Biaffine [BERT]	76.2	72.5	75.2	69.2
Pyramid [Flair]	79.0	78.1	84.7	69.3
Recursive-CRF [Flair]	80.3	77.4	82.8	70.4
MLC [Flair]	**80.9**	**80.1**	**86.2**	**72.5**

Table 5: Results on nested and non-nested entities.

GENIA				
Model	$m_{nesting}$	m_{ME}	m_{NDT}	m_{NST}
Layered	26.2	-	41.7	9.7
Exhaustive	25.8	-	41.2	17.7
Boundary	26.6	-	40.5	17.8
Biaffine [BERT]	**34.5**	-	**51.9**	**22.9**
Pyramid [BERT]	33.4	-	49.5	20.9
Recursive-CRF [Flair]	31.5	-	49.1	19.4
MLC [Flair]	27.9	-	47.8	0

GermEval				
Model	$m_{nesting}$	m_{ME}	m_{NDT}	m_{NST}
Layered	37.3	-	40.4	16.2
Exhaustive	27.8	-	38.2	9.7
Boundary	21.2	-	25.5	7.8
Biaffine [BERT]	55.7	-	**64.3**	20.8
Pyramid [BERT]	**56.5**	-	63.8	21.4
Recursive-CRF [BERT]	51.1	-	58.9	**23.9**
MLC [Flair]	49.1	-	59.3	0

Chilean Waiting List				
Model	$m_{nesting}$	m_{ME}	m_{NDT}	m_{NST}
Layered	51.6	71.1	49.5	-
Exhaustive	28.4	0	41.7	-
Boundary	28.2	0	35.4	-
Biaffine [BERT]	41.8	0	55.1	-
Pyramid [Flair]	54.9	**73.7**	57.9	-
Recursive-CRF [Flair]	56.0	71.7	58.8	-
MLC [Flair]	**60.6**	72.5	**60.0**	-

Table 6: Our task-specific metrics. If columns have no results, it means that there was not a significant number of examples.

nesting cases. One possible reason for this low performance is that these metrics are strict, as inner and outer entities must be correctly predicted.

Although the selected baselines are designed to deal with nestings of the same type, their m_{NST} results in GENIA and GermEval are poor, while the results using the m_{NDT} metric are much higher. This suggests that NST is the most challenging case to identify for all models. Therefore, we believe that a model should not be prematurely discarded based on its limitation in handling a particular type of nesting. For example, although the MLC architecture cannot strictly identify the NST case in GENIA and GermEval, it obtains excellent results on the NDT case and the outermost entities involved in the NST. In contrast, concerning the m_{ME} metric, we note that the performance of the four models addressing this case is quite good, suggesting that it is not a complex case to recognize but still not taken into account when building nested NER models.

We highlight that in the Chilean corpus where the state-of-the-art is reached, almost half of the complete nestings ($m_{nesting}$) are correctly recognized, which is a reliable indicator of the performance of our model on the nested NER task. These results suggest that the MLC architecture should be considered in future state-of-the-art comparisons due to its effectiveness. Besides, we argue that there is still much work to be done in nested NER, as most models fail to simultaneously recognize the inner and outer entities of nestings, which is one of the main objectives of the task.

5 Conclusions and Future Work

This paper presented an effective but overlooked neural model for nested NER based on sequence labeling architectures. Specifically, we revisited the Multiple LSTM-CRF (MLC) approach, which uses a single flat NER model per entity type. We argue that this approach has not been analyzed in-depth since large pre-trained language models have not been incorporated. Our experimental results show that by adding a character-level language model to the MLC architecture, it achieves state-of-the-art in the Chilean Waiting List corpus. One of the main advantages of using this approach is that it can handle entities tagged with more than one entity type, barely addressed in previous works.

In addition, to alleviate some gaps found in current evaluation metrics, we implemented an open-source library that computes task-specific metrics for nested NER. The results according to these metrics are low, especially when it comes to recognizing complete nestings, i.e., inner and outer entities simultaneously. This finding shows that most models are better at identifying flat entities or

part of nested entities, which is not the primary goal of the task. We hope that our study will help raise awareness in the research community that overlooking intuitive models and using only standard metrics when evaluating a new complex solution can be misleading and create an overly optimistic impression of the new solution's performance.

Future directions include incorporating the hierarchical information of nested entities to improve the performance of our model. In addition, we plan to analyze two underexplored issues in the NER task: crossing and discontinuous entities. The first corresponds to cases where entities are not fully nested in other entities, but there is an overlap, and the second is when entities do not necessarily have consecutive tokens in the sentence.

Acknowledgements

This work was funded by ANID Chile: Basal Funds for Center of Excellence FB210005 (CMM) and FB210017 (CENIA); Millennium Science Initiative Program ICN17_002 (IMFD) and ICN2021_004 (iHealth); Fondecyt grants 11200290 and 11201250. We also acknowledge the U-Inicia VID Project UI-004/20. This research was partially supported by the supercomputing infrastructure of the NLHPC (ECM-02) and the Patagón supercomputer of Universidad Austral de Chile (FONDEQUIP EQM180042). We are also grateful from the help received from the reviewers.

References

Alan Akbik, Tanja Bergmann, Duncan Blythe, Kashif Rasul, Stefan Schweter, and Roland Vollgraf. 2019. Flair: An easy-to-use framework for state-of-the-art nlp. In *NAACL 2019, 2019 Annual Conference of the North American Chapter of the Association for Computational Linguistics (Demonstrations)*, pages 54–59.

Alan Akbik, Duncan Blythe, and Roland Vollgraf. 2018. Contextual string embeddings for sequence labeling. In *Proceedings of the 27th International Conference on Computational Linguistics*, pages 1638–1649, Santa Fe, New Mexico, USA. Association for Computational Linguistics.

Beatrice Alex, Barry Haddow, and Claire Grover. 2007. Recognising nested named entities in biomedical text. In *Biological, translational, and clinical language processing*, pages 65–72, Prague, Czech Republic. Association for Computational Linguistics.

Pablo Báez, Fabián Villena, Matías Rojas, Manuel Durán, and Jocelyn Dunstan. 2020. The Chilean waiting list corpus: a new resource for clinical named entity recognition in Spanish. In *Proceedings of the 3rd Clinical Natural Language Processing Workshop*, pages 291–300, Online. Association for Computational Linguistics.

Darina Benikova, Chris Biemann, and Marc Reznicek. 2014. NoSta-D named entity annotation for German: Guidelines and dataset. In *Proceedings of the Ninth International Conference on Language Resources and Evaluation (LREC'14)*, pages 2524–2531, Reykjavik, Iceland. European Language Resources Association (ELRA).

Nancy Chinchor and Patricia Robinson. 1997. Muc-7 named entity task definition. In *Proceedings of the 7th Conference on Message Understanding*, volume 29, pages 1–21.

Billy Chiu, Gamal Crichton, Anna Korhonen, and Sampo Pyysalo. 2016. How to train good word embeddings for biomedical NLP. In *Proceedings of the 15th Workshop on Biomedical Natural Language Processing*, pages 166–174, Berlin, Germany. Association for Computational Linguistics.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, Minneapolis, Minnesota. Association for Computational Linguistics.

Hao Fei, Yafeng Ren, and Donghong Ji. 2020. Dispatched attention with multi-task learning for nested mention recognition. *Information Sciences*, 513:241–251.

Jenny Rose Finkel and Christopher D. Manning. 2009. Nested named entity recognition. In *Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing*, pages 141–150, Singapore. Association for Computational Linguistics.

R. Florian, H. Hassan, A. Ittycheriah, H. Jing, N. Kambhatla, X. Luo, N. Nicolov, and S. Roukos. 2004. A statistical model for multilingual entity detection and tracking. In *Proceedings of the Human Language Technology Conference of the North American Chapter of the Association for Computational Linguistics: HLT-NAACL 2004*, pages 1–8, Boston, Massachusetts, USA. Association for Computational Linguistics.

Yao Fu, Chuanqi Tan, Mosha Chen, Songfang Huang, and Fei Huang. 2021. Nested named entity recognition with partially-observed treecrfs. *Proceedings of the AAAI Conference on Artificial Intelligence*, 35(14):12839–12847.

Edouard Grave, Piotr Bojanowski, Prakhar Gupta, Armand Joulin, and Tomas Mikolov. 2018. Learning

word vectors for 157 languages. In *Proceedings of the Eleventh International Conference on Language Resources and Evaluation (LREC 2018)*, Miyazaki, Japan. European Language Resources Association (ELRA).

Meizhi Ju, Makoto Miwa, and Sophia Ananiadou. 2018. A neural layered model for nested named entity recognition. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 1446–1459, New Orleans, Louisiana. Association for Computational Linguistics.

Arzoo Katiyar and Claire Cardie. 2018. Nested named entity recognition revisited. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 861–871, New Orleans, Louisiana. Association for Computational Linguistics.

Jin-Dong Kim, Tomoko Ohta, Yuka Tateisi, and Jun'ichi Tsujii. 2003. Genia corpus—a semantically annotated corpus for bio-textmining. *Bioinformatics (Oxford, England)*, 19 Suppl 1:i180–2.

Guillaume Lample, Miguel Ballesteros, Sandeep Subramanian, Kazuya Kawakami, and Chris Dyer. 2016. Neural architectures for named entity recognition. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 260–270, San Diego, California. Association for Computational Linguistics.

Hongyu Lin, Yaojie Lu, Xianpei Han, and Le Sun. 2019. Sequence-to-nuggets: Nested entity mention detection via anchor-region networks. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 5182–5192, Florence, Italy. Association for Computational Linguistics.

Liyuan Liu, Jingbo Shang, Xiang Ren, Frank F. Xu, Huan Gui, Jian Peng, and Jiawei Han. 2018. Empower sequence labeling with task-aware neural language model. In *Proceedings of the Thirty-Second AAAI Conference on Artificial Intelligence and Thirtieth Innovative Applications of Artificial Intelligence Conference and Eighth AAAI Symposium on Educational Advances in Artificial Intelligence*.

Wei Lu and Dan Roth. 2015. Joint mention extraction and classification with mention hypergraphs. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 857–867, Lisbon, Portugal. Association for Computational Linguistics.

Aldrian Obaja Muis and Wei Lu. 2017. Labeling gaps between words: Recognizing overlapping mentions with mention separators. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 2608–2618, Copenhagen, Denmark. Association for Computational Linguistics.

Matías Rojas, Jocelyn Dunstan, and Fabián Villena. 2022. Clinical flair: A pre-trained language model for Spanish clinical natural language processing. In *Proceedings of the 4th Clinical Natural Language Processing Workshop*, pages 87–92, Seattle, WA. Association for Computational Linguistics.

Takashi Shibuya and Eduard Hovy. 2020. Nested named entity recognition via second-best sequence learning and decoding. *Transactions of the Association for Computational Linguistics*, 8:605–620.

Mohammad Golam Sohrab and Makoto Miwa. 2018. Deep exhaustive model for nested named entity recognition. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 2843–2849, Brussels, Belgium. Association for Computational Linguistics.

Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. 2014. Dropout: A simple way to prevent neural networks from overfitting. *Journal of Machine Learning Research*, 15(56):1929–1958.

Jana Straková, Milan Straka, and Jan Hajic. 2019. Neural architectures for nested NER through linearization. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 5326–5331, Florence, Italy. Association for Computational Linguistics.

Bailin Wang and Wei Lu. 2018. Neural segmental hypergraphs for overlapping mention recognition. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 204–214, Brussels, Belgium. Association for Computational Linguistics.

Jue Wang, Lidan Shou, Ke Chen, and Gang Chen. 2020. Pyramid: A layered model for nested named entity recognition. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 5918–5928, Online. Association for Computational Linguistics.

Juntao Yu, Bernd Bohnet, and Massimo Poesio. 2020. Named entity recognition as dependency parsing. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 6470–6476, Online. Association for Computational Linguistics.

Changmeng Zheng, Yi Cai, Jingyun Xu, Ho-fung Leung, and Guandong Xu. 2019. A boundary-aware neural model for nested named entity recognition. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 357–366, Hong Kong, China. Association for Computational Linguistics.

ERGO: Event Relational Graph Transformer for Document-level Event Causality Identification

Meiqi Chen[1], **Yixin Cao**[2], **Kunquan Deng**[3],
Mukai Li[4], **Kun Wang**[4], **Jing Shao**[4], **Yan Zhang**[1]*
[1] Peking University [2] Singapore Management University
[3] Beihang University [4] SenseTime Research
meiqichen@stu.pku.edu.cn

Abstract

Document-level Event Causality Identification (DECI) aims to identify event-event causal relations in a document. Existing works usually build an event graph for global reasoning across multiple sentences. However, the edges between events have to be carefully designed through heuristic rules or external tools. In this paper, we propose a novel **E**vent **R**elational **G**raph Transf**O**rmer (ERGO) framework[1] for DECI, to ease the graph construction and improve it over the noisy edge issue. Different from conventional event graphs, we define a pair of events as a node and build a complete event relational graph without any prior knowledge or tools. This naturally formulates DECI as a node classification problem, and thus we capture the causation transitivity among event pairs via a graph transformer. Furthermore, we design a criss-cross constraint and an adaptive focal loss for the imbalanced classification, to alleviate the issues of false positives and false negatives. Extensive experiments on two benchmark datasets show that ERGO greatly outperforms previous state-of-the-art (SOTA) methods (12.8% F1 gains on average).

1 Introduction

Event Causality Identification (ECI) is the task of identifying if the occurrence of one event causes another in text. As shown in Figure 1, given the text "*... the outage$_2$ was caused by a terrestrial break in the fiber in Egypt ...*", where "*outage$_2$*" and "*break*" are event triggers, an ECI model should predict if they have a causal relation or not. Discovering causal relationships not only helps to deeply understand how the world progresses, but also is an important goal of empirical research in various areas, such as machine reading comprehension (Berant et al., 2014), question answering (Oh et al., 2016), future event forecasting (Hashimoto, 2019),

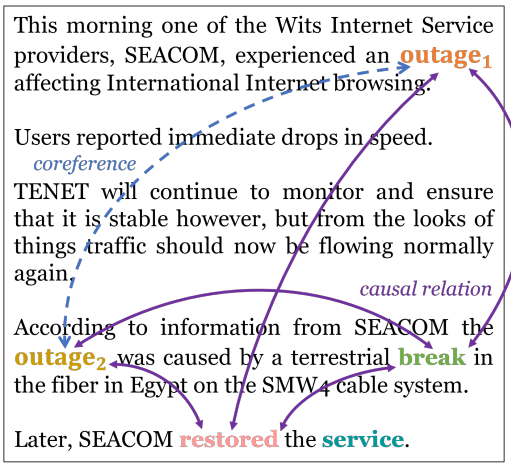

Figure 1: Example of DECI. Solid purple lines denote target causal relations.

and event knowledge graph construction (Ma et al., 2022b).

Causality is usually implicit in natural language (Copley and Martin, 2014), especially when events scatter in a document, a.k.a. Document-level ECI (DECI). Recent methods typically construct an event graph to assist the global inference across multiple sentences, where nodes are events and edges are their relations, such as linguistic dependency or adjacent contexts (Gao et al., 2019; Zhao et al., 2021). However, there are two major issues. First, the edges heavily rely on external tools and heuristic rules, which are not always reliable and may introduce noise (Tran Phu and Nguyen, 2021). Second, it is like a chicken-egg problem — to identify (causal) relations between events, you need to extract their relations to build the graph first.

In this paper, we propose a novel **E**vent **R**elational **G**raph Transf**O**rmer (ERGO) framework for DECI, which doesn't require any external tools and can effectively alleviate the noise issue. Different from conventional event graphs, the basic idea is to build an event relational graph that naturally converts ECI into a node classification problem, where each node denotes a pair of events,

*Corresponding author.
[1]https://github.com/chenmeiqii/ERGO.git

Proceedings of the 29th International Conference on Computational Linguistics, pages 2118–2128
October 12–17, 2022.

and all edges among nodes are initialized to capture potential causal chains, following the assumption "preserving transitivity of causation" (Paul et al., 2013). If event A causes event B and event B causes event C, then we have event A causes event C. That is, if the node of (A, C) receives positive predictions (i.e., causal relation) from nodes (A, B) and (B, C), it is positive, too. By contrast, if either/both of nodes (A, B) and (B, C) are negative (i.e., no relation), it is not necessary that (A, C) is negative, either. To this end, we leverage a graph transformer to model the graph and assign a lower weight to such uninformative edges, paying more attention to other paths or its own textual contexts.

Although the proposed graph directly models all event pairs for causality identification, it poses a great challenge of false positive and false negative issues. First, most of the event pairs have no causal relations. That is, negative nodes are dominant, and the imbalanced classification will easily confuse the model into false-negative predictions. Second, to ease the graph construction, we assume that all nodes can pass information with each other via the complete graph structure. While there are many spurious correlations between events, which can be incorrectly propagated to neighbor nodes, leading to severe false positives. For example, “treatment” and “death” frequently co-occur in the same document, but there is no causality between them since it is not “treatment” that causes “death”.

To address the above issues, we further design a *criss-cross constraint* and an adaptive focal loss. The criss-cross constraint simplifies the paths of each pair of events for global inference. Instead of a complete graph, we assume that there is an edge between two nodes, only if the two pairs of events share at least one event. Clearly, if they have no common event, there must be no direct causal effect between them. The adaptive focal loss re-weights positive and negative samples to tackle the imbalance issue. On the one hand, we leverage a weighting factor to balance two classes' training. On the other hand, we also introduce a scaling factor to focus more on difficult samples.

Our contributions can be summarized as follows:

- We propose to build an event relational graph without using any external tools to capture causal transitivity.
- We propose a novel framework ERGO that further alleviates false positive and false negative issues for DECI.
- Extensive experiments on two benchmark datasets indicate that ERGO greatly outperforms previous SOTA methods (12.8% F1 gains on average). We have also conducted both quantitative and qualitative analysis to better understand key components of ERGO. Furthermore, detailed error analysis provides insights into our approach and the task.

2 Related Work

ECI has attracted much attention in recent years. In terms of text corpus, there are mainly two types of methods: Sentence-level ECI (SECI) and DECI.

In the first research line, early methods usually design various features tailored for causal expressions, such as lexical and syntactic patterns (Riaz and Girju, 2013, 2014a,b), causality cues or markers (Riaz and Girju, 2010; Do et al., 2011; Hidey and McKeown, 2016), statistical information (Beamer and Girju, 2009; Hashimoto et al., 2014), and temporal patterns (Riaz and Girju, 2014a; Ning et al., 2018). Then, researchers resort to a large amount of labeled data to mitigate the efforts of feature engineering and to learn diverse causal expressions (Hu et al., 2017; Hashimoto, 2019). To alleviate the annotation cost, recent methods leverage Pre-trained Language Models (PLMs, e.g., BERT (Devlin et al., 2019)) for the ECI task and have achieved SOTA performance (Kadowaki et al., 2019; Liu et al., 2020; Zuo et al., 2020). To deal with implicit causal relations, Cao et al. (2021) incorporate the external knowledge from ConceptNet (Speer et al., 2017) for reasoning, which achieves promising results. Zuo et al. (2021a) learn context-specific causal patterns from external causal statements and incorporate them into a target ECI model. Zuo et al. (2021b) propose a data augmentation method to further solve the data lacking problem.

Along with the success of sentence-level natural language understanding, many tasks are extended to the entire document, such as relation extraction (Yao et al., 2019), natural language inference (Yin et al., 2021), and event argument extraction (Ma et al., 2022a). A concurrent and relevant work is (Tan et al., 2022), which also leverages focal loss for entity relation extraction. The difference is that the focal loss in (Tan et al., 2022) is used to make long-tail (positive) classes contribute more to the overall loss, while the focal loss in our ERGO tackles the imbalance issue of DECI task by focusing more on difficult samples. We further leverage a

weighting factor in the focal loss to balance two classes' training, which is not considered in (Tan et al., 2022). Moreover, in Section 4.6, we have given a more detailed analysis of the impact of adaptive focal loss on the DECI task.

Compared with SECI, DECI not only aggravates the lack of clear causal indicators but also poses a new challenge of cross-sentence inference. Gao et al. (2019) use Integer Linear Programming (ILP) to model the global causal structures; Zhao et al. (2021) proposes a document-level context-based graph inference mechanism to capture interaction among events; RichGCN (Tran Phu and Nguyen, 2021) constructs document-level interaction graphs and uses Graph Convolutional Network (GCN, Kipf and Welling (2017)) to capture relevant connections. However, the construction of the aforementioned global structure or graph requires sophisticated feature extraction or tools, which may introduce noise and mislead the model (Tran Phu and Nguyen, 2021). Compared with them, we formulate DECI as an efficient node classification framework, which could capture the global interactions among event pairs automatically, as well as alleviate the imbalanced and noisy issues.

3 Methodology

The goal of our proposed framework ERGO is to capture potential causal chains for document-level reasoning. There are three main components: (1) **Document Encoder** to encode the document and obtain contextualized representations of events as the inputs for the following components; (2) **Event Relational Graph Transformer** that models causal chain for global inference by building a handy event relational graph, where node features are from the Document Encoder and enhanced through propagation over the graph; and (3) **Classification with Adaptive Focal Loss** to predict if a node of event pair has causal relation or not based on their enhanced node features, with considering the imbalance issue.

3.1 Document Encoder

Given a document $\mathcal{D} = [x_t]_{t=1}^{L}$ (can be of any length L), the document encoder aims to output the contextualized document and event representations. We leverage a Pre-trained Language Model (PLM) as a base encoder to obtain the contextualized embeddings. Following conventions, we add special tokens at the start and end of $\mathcal{D}$ (e.g., "[CLS]" and "[SEP]" of BERT (Devlin et al., 2019)), and insert additional special tokens "<t>" and "</t>" at the start and end of all the events to mark event positions. Then, we have:

$$H = [h_1, h_2, ..., h_L] = \text{Encoder}([x_1, x_2, ..., x_L]), \quad (1)$$

where $h_i \in \mathbb{R}^d$ is the embedding of token x_i. We use the embedding of token "[CLS]" to represent the document and the embeddings of token "<t>" to represent the events.

In this paper, we choose pre-trained BERT (Devlin et al., 2019) and Longformer (Beltagy et al., 2020) as encoders for comparison. We handle documents longer than the limits of PLMs as follows.

BERT for Document Encoder To handle documents that are longer than 512 (BERT's original limit), we leverage a *dynamic window* to encode the entire document. Specifically, we divide $\mathcal{D}$ into several overlapping spans according to a specific step size and input them into BERT separately (details can be found in Section 4.2). Then, we find and average all the embeddings of token "[CLS]" or "<t>" of different spans to represent the whole document or each event, respectively.

Longformer for Document Encoder Longformer introduces a localized sliding window based attention mechanism (the default window size is 512) with little global attention to reduce computation and extend BERT for long documents. In our implementation, we apply its efficient local and global attention pattern. Specifically, we use global attention on the "<s>" token (Longformer uses "<s>" and "</s>" as the special start and end tokens, corresponding to BERT's "[CLS]" and "[SEP]"), and local attention on other tokens, which could build full sequence representations. The maximum document length allowed by Longformer is 4096, which is suitable for most documents. Therefore, we directly take the embedding of token "<s>" as document representation and embedding of token "<t>" as event representation.

3.2 Event Relational Graph Transformer

In this section, we first introduce how to construct the event relational graph, including the criss-cross constraint. Then, based on it, we leverage a Relational Graph Transformer (RGT) to capture the high-order interaction among event pairs and obtain enhanced event pair representations for the final classification.

3.2.1 Event Relational Graph Construction

Given all the events of document $\mathcal{D}$, we construct an event relational graph $\mathcal{G} = \{\mathcal{V}, \mathcal{E}\}$, where $\mathcal{V}$ is the set of nodes, $\mathcal{E}$ is the set of edges. We highlight the following differences of $\mathcal{G}$ from previous event graphs. First, for each node in $\mathcal{V}$, it refers to a different pair of events in $\mathcal{D}$, instead of a single event. Our motivation is to learn the relation of relations between events, i.e., the logic of causal transitivity, for higher-order reasoning. Second, for edges $\mathcal{E}$, we do not require any prior relations between events. Instead, we add all edges between any two nodes into $\mathcal{E}$. Thus, $\mathcal{G}$ is initialized as a complete graph.

Criss-cross Constraint. To simplify the graph structure and alleviate the negative impacts of false positives propagation, we introduce a criss-cross constraint. It assumes that there is an edge between two nodes, *only* if the two corresponding event pairs share at least one event. The basic idea behind this is that if two pairs of events have no common event, there must be no *direct* causal effect between them. Still, they can have causal interactions if there are some mediator events, and such causality takes effects conditioned on the mediator. For example in Figure 1, (1) the causality information of (*restore*, *service*) has no effect on predicting the causal relation of (*outage*$_1$, *break*). (2) the causality of (*outage*$_2$, *restored*) has a transitive effect on predicting the causal relation of (*outage*$_1$, *break*) if we know that (*restored*, causes, *break*) and (*outage*$_1$, *outage*$_2$) is coreference[2]. Note that the criss-cross constraint is not posed over the graph directly, which is different for each event pair. In Section 4.5, we show that such a simple and intuitive constraint brings considerable performance gains compared with using a complete graph.

3.2.2 Relational Graph Transformer

Node Embedding Initialization For global inference, we first initialize node feature vectors with event pair node embedding, which is based on the contextualized event embeddings by Equation (1). Formally, for event pair (e_1, e_2) and the corresponding contextual embeddings (h_{e_1}, h_{e_2}), their event pair node embedding is initialized by:

$$v_{e_{1,2}}^{(0)} = [h_{e_1} \| h_{e_2}], \tag{2}$$

where $\|$ denotes concatenation, 0 indicates the initial state for the following neural layers.

The event pair node embeddings represent the implicit relational information between two events, which enables us to integrate event pair representation learning and causal chain inference seamlessly, without any prior knowledge or tools. Clearly, better initial features of nodes will provide more discriminative signals from local textual contexts for classification. On the other hand, structural reasoning further improves the discriminative ability of node features by considering all event pairs globally, such that confident prediction shall help others via causality transitivity.

Each RGT layer l is closed to the transformer architecture proposed in (Vaswani et al., 2017). It takes a set of node embeddings $\mathbf{v}^{(l-1)} \in \mathbb{R}^{N \times d_{\text{in}}}$ as input, and outputs a new set of node embeddings: $\mathbf{v}^{(l)} \in \mathbb{R}^{N \times d_{\text{out}}}$, where N is the number of event pairs, d_{in} and d_{out} are the dimensions of input and output embeddings.

To better exploit the relational information from each neighbor to predict the causal relation of an event pair node i, we perform a shared self-attention mechanism to measure the importance of neighbor j to i:

$$\text{att}_{ij} = \frac{(v_i \mathbf{W}_q)(v_j \mathbf{W}_k)^T}{\sqrt{d_k}}, \tag{3}$$

where d_k is the hidden size, $\mathbf{W}_q, \mathbf{W}_k \in \mathbb{R}^{d_{\text{in}} \times d_k}$ are parameter weight matrices, $\sqrt{d_k}$ is a scaling factor (Vaswani et al., 2017). Thus, negative and uninformative nodes are expected to assign lower attention weights.

Then we normalize att_{ij} across all choices of j using a softmax function to make the importance more comparable:

$$\alpha_{ij} = \text{softmax}_j(\text{att}_{ij}) = \frac{\exp(\text{att}_{ij})}{\sum_{z \in \mathcal{N}_i} \exp(\text{att}_{iz})}, \tag{4}$$

where $\mathcal{N}_i$ are all the first order neighbors of node i.

To aggregate relational knowledge from the neighborhood information, we compute a weighted linear combination of the embeddings :

$$v_i^{(l)} = \sum_{j \in \mathcal{N}_i} \alpha_{ij} (v_j \mathbf{W}_v), \tag{5}$$

where $\mathbf{W}_v \in \mathbb{R}^{d_{\text{in}} \times d_k}$ is the parameter weight matrix. We also perform multi-head attention to

[2]In the datasets, coreference events have similar surface forms and thus can be implicitly captured by PLMs. We leave further coreference modeling in the future work.

jointly attend to information from different representation subspaces. Finally, the output embedding of node i can be represented as:

$$v_i^{(l)} = \Big(\Big\Vert_{c=1}^{C} \sum_{j \in \mathcal{N}_i} \alpha_{ij}(v_j \mathbf{W}_v) \Big) \mathbf{W}_o, \quad (6)$$

where $\|$ denotes concatenation, C is the number of heads. $\mathbf{W}_o \in \mathbb{R}^{C\dot{d}_k \times d_{\text{out}}}$ is the parameter weight matrix. By simultaneously computing embeddings for all the event pair nodes, a node embedding matrix $\mathbf{v}^{(l)} \in \mathbb{R}^{N \times d_{\text{out}}}$ is obtained. By stacking multiple layers, RGT could reach high-order connectivity and capture complex interactions.

Note that our framework is flexible to almost arbitrary Graph Neural Networks (GNNs). Here we leverage RGT for its powerful expressiveness. We also report results with GCN in Section 4.5.

3.3 Classification with Adaptive Focal Loss

Remember that we formulate DECI as a node classification task, which predicts the label of each node as either a positive or negative class. However, the number of negative samples during training far exceeds that of positives, leading to an imbalanced classification problem. What is worse, the dominant negatives contain many spurious correlations between events ("treatment" and "death" example in Section 1). How can we know the difficulties of sample prediction, so that ERGO can penalize them to alleviate false negatives for better performance?

To address this problem, we leverage an adaptive loss function for training, following focal loss (Lin et al., 2017). Specifically, we reshape the loss function to down-weight easy samples and thus focus on hard ones. Formally, a modulating factor is added to Cross-Entropy (CE) loss, with a predefined focusing hyper-parameter $\gamma \geq 0$, which is defined as:

$$\mathcal{L}_{\text{FL}} = -\sum_{e_i, e_j \in \mathcal{D}} (1 - p_{e_{i,j}})^{\gamma} \log(p_{e_{i,j}}). \quad (7)$$

where $p_{e_{i,j}}$ is the predicted probability of whether there is a causal relation between events e_i and e_j. $p_{e_{i,j}}$ is defined as follows:

$$p_{e_{i,j}} = \text{softmax}\left(\left[v_{e_{i,j}} || h_{[CLS]} \right] \mathbf{W}_p \right), \quad (8)$$

where $\mathbf{W}_p$ is the parameter weight matrix, $\|$ denotes concatenation. Here we concatenate embeddings of $h_{[CLS]}$ (of BERT) or $h_{<s>}$ (of Longformer) to each node in order to incorporate the global document representation for classification.

This scaling factor, $(1 - p_{e_{i,j}})^{\gamma}$, allows us to efficiently train on all event pairs by encouraging the model to focus on difficult samples, reducing false-negative predictions. For example, when a sample is misclassified and $p_{e_{i,j}}$ is small, the modulating factor is near 1, and the loss is unaffected. As $p_{e_{i,j}} \to 1$, the factor goes to 0 and the loss for well-classified examples is down-weighted. Therefore, the focusing parameter γ smoothly adjusts the rate at which easy examples are down-weighted. When $\gamma = 0$, $\mathcal{L}_{\text{FL}}$ is equivalent to CE loss, and with the increase of γ, the influence of the modulating factor also increases. We will give further discussion in Section 4.6.

Besides, we use an α-balanced variant of the focal loss, which introduces a weighting factor α in $[0, 1]$ for class "positive" and $1 - \alpha$ for class "negative". The value of α is related to the ratio of positive and negative samples. The final adaptive focal loss $\mathcal{L}_{\text{FL}_b}$ can be written as:

$$\mathcal{L}_{\text{FL}_\text{b}} = -\sum_{e_i, e_j \in \mathcal{D}} \alpha_{e_{i,j}} (1 - p_{e_{i,j}})^{\gamma} \log(p_{e_{i,j}}). \quad (9)$$

4 Experiments

4.1 Datasets and Evaluation Metrics

We evaluate our proposed method on two widely used datasets, EventStoryLine (version 0.9) (Caselli and Vossen, 2017) and Causal-TimeBank (Mirza, 2014).

EventStoryLine contains 22 topics, 258 documents, 5,334 events, 7,805 intra-sentence and 62,774 inter-sentence event pairs (1,770 and 3,885 of them are annotated with causal relations respectively). Following Gao et al. (2019) and (Tran Phu and Nguyen, 2021), we group documents according to their topics. Documents in the last two topics are used as the development data, and documents in the remaining 20 topics are employed for a 5-fold cross-validation.

Causal-TimeBank contains 184 documents, 6,813 events, and 318 of 7,608 event pairs are annotated with causal relations. Following (Liu et al., 2020), we employ a 10-fold cross-validation evaluation. Note that the number of inter-sentence event pairs in Causal-TimeBank is quite small (i.e., only 18 pairs), following (Tran Phu and Nguyen, 2021), we only evaluate ECI performance for intra-sentence event pairs on Causal-TimeBank.

Evaluation Metrics For evaluation, we adopt Precision (P), Recall (R), and F1-score (F1) as evaluation metrics, same as previous methods to ensure comparability.

4.2 Implementation Details

We implement our method based on Pytorch. We use uncased BERT-base (Devlin et al., 2019) or Longformser-base (Beltagy et al., 2020) as the document encoder. For the BERT-base document encoder, we set the dynamic window size to 256, and divide documents into several overlapping windows with a step size 32. We optimize our model with AdamW (Loshchilov and Hutter, 2019) using a learning rate of 0.00002 with a linear warm-up for the first 8% steps. We apply dropout (Srivastava et al., 2014) between layers and clip the gradients of model parameters to a max norm of 1.0. We perform early stopping based on the F1 score on the development set. We tune the hyper-parameters by grid search based on the development set performance: heads $C \in \{1, 2, \mathbf{4}, 8\}$ for the relational graph transformer model, dropout rate $\in \{0.1, \mathbf{0.2}, 0.3\}$, focusing parameter $\gamma \in \{0, 1, \mathbf{2}, 3\}$, and weighting factor $\alpha \in \{0.65, \mathbf{0.75}, 0.85\}$.

4.3 Baselines

We compare our proposed ERGO with various state-of-the-art SECI and DECI methods.

SECI Baselines (1) **KMMG** (Liu et al., 2020), which proposes a mention masking generalization method and use extenal knowledge databases. (2) **KnowDis** (Zuo et al., 2020), a knowledge enhanced distant data augmentation method to alleviate the data lacking problem. (3) **LSIN** (Cao et al., 2021), which constructs a descriptive graph to leverage external knowledge and has the current SOTA performance for intra-sentence ECI. (4) **LearnDA** (Zuo et al., 2021b), which uses knowledge bases to augment training data. (5) **CauSeRL** (Zuo et al., 2021a), which learns context-specific causal patterns from external causal statements for ECI.

DECI Baselines (1) **OP** (Caselli and Vossen, 2017), a dummy model that assigns causal relations to event pairs. (2) **LR+** and **LIP** (Gao et al., 2019), feature-based methods that construct document-level structures and use various types of resources. (3) **BERT (our implement)** a baseline method that leverages dynamic window and event marker techniques. (4) **RichGCN** (Tran Phu and Nguyen, 2021), which constructs document-level interaction graph and uses GCN to capture relevant connections. RichGCN has the current SOTA performance for inter-sentence ECI.

Model	EventStoryLine			Causal-TimeBank		
	P	R	F1	P	R	F1
OP	22.5	**98.6**	36.6	-	-	-
LR+	37.0	45.2	40.7	-	-	-
LIP	38.8	52.4	44.6	-	-	-
KMMG[○]	41.9	62.5	50.1	36.6	55.6	44.1
KnowDis[○]	39.7	66.5	49.7	42.3	60.5	49.8
LSIN[○]	47.9	58.1	52.5	51.5	56.2	53.7
LearnDA[○]	42.2	69.8	52.6	41.9	68.0	51.9
CauSeRL[○]	41.9	69.0	52.1	43.6	**68.1**	53.2
BERT[○]	47.8	57.2	52.1	47.6	55.1	51.1
RichGCN[○]	49.2	63.0	55.2	39.7	56.5	46.7
ERGO[○]	49.7	72.6	59.0	58.4	60.5	59.4
ERGO[◇]	**57.5**	72.0	**63.9**	**62.1**	61.3	**61.7**

Table 1: Model's intra-sentence performance on EventStoryLine and Causal-TimeBank, the best results are in **bold** and the second-best results are underlined. [○] and [◇] denote models that use pre-trained BERT-base and Longformer-base encoders, respectively. Overall, our ERGO outperforms previous SOTA models (with a significant test at the level of 0.05).

4.4 Overall Results

Since some baselines are evaluated only on EventStoryLine, the baselines used for EventStoryLine and Causal-TimeBank are different. Some baselines can not handle the inter-sentence scenarios in EventStoryLine. Thus we report the results of intra- and inter- sentence settings separately.

4.4.1 Intra-sentence Evaluation

From Table 1, we can observe that:

(1) ERGO outperforms all the baselines by a large margin on both datasets. Compared with SOTA methods, ERGO-BERT$_{\text{BASE}}$ achieves 6.9% improvements of F1-score on EventStoryLine, and 10.6% on Causal-TimeBank. This demonstrates the effectiveness of ERGO.

(2) The feature-based method OP achieves the highest Recall on EventStoryLine, which may be due to simply assigning causal relations by mimicking the textual order of presentation. This leads to many false positives and thus a low Precision.

(3) The usage of PLMs boosts performance. Using Longformer$_{\text{BASE}}$ as the encoder, ERGO achieves better results than ERGO-BERT$_{\text{BASE}}$, which also achieves new SOTA results. The reason may be: 1) Longformer continues pre-training from RoBERTa (Liu et al., 2019), which has been

Model	Inter-sentence			Intra + Inter		
	P	R	F1	P	R	F1
OP	8.4	**99.5**	15.6	10.5	**99.2**	19.0
LR+	25.2	48.1	33.1	27.9	47.2	35.1
LIP	35.1	48.2	40.6	36.2	49.5	41.9
BERT[○]	36.8	29.2	32.6	41.3	38.3	39.7
RichGCN[○]	39.2	45.7	42.2	42.6	51.3	46.6
ERGO[○]	43.2	48.8	45.8	46.3	50.1	48.1
ERGO[◇]	**51.6**	43.3	**47.1**	**48.6**	53.4	**50.9**

Table 2: Model's inter and (intra+inter)-sentence performance on EventStoryLine.

Model	Intra	Inter	Intra + Inter
ERGO[○]	59.0	45.8	48.1
ERGO_1[○]	56.6	43.5	45.6
ERGO_2[○]	56.2	41.8	44.6
ERGO_3[○]	58.3	43.6	47.3
ERGO[◇]	**63.9**	**47.1**	**50.9**
ERGO_1[◇]	61.3	44.7	47.1
ERGO_2[◇]	60.7	43.1	46.3
ERGO_3[◇]	62.6	45.9	49.1

Table 3: F1 Results of Ablation study on EventStoryLine, where ERGO_1 denotes ERGO w/ a complete graph, ERGO_2 denotes ERGO w/ GCN, ERGO_3 denotes ERGO w/o the focal factor.

found to outperform BERT on many tasks; 2) Longformer leverages an efficient local and global attention pattern, which is beneficial to capture longer contextual information for inference.

4.4.2 Inter-sentence Evaluation

From Table 2, we can observe that:

(1) ERGO greatly outperforms all the baselines under both inter- and (intra+inter)-sentence settings, especially in terms of Precision. This demonstrates that our ERGO can make better document-level inference via the event relational graph, while alleviating the negative impacts of false positives.

(2) The overall F1-score of inter-sentence setting is much lower than that of intra-sentence, which indicates the challenge of document-level ECI.

(3) The BERT baseline performs well on intra-sentence event pairs. However, it performs much worse than LIP, RichGCN, and ERGO under inter-sentence settings, which indicates that a document-level structure or graph is helpful to capture the global interactions for prediction.

4.5 Ablation Study

To analyze the main components of ERGO, we have the following variants, as shown in Table 3:

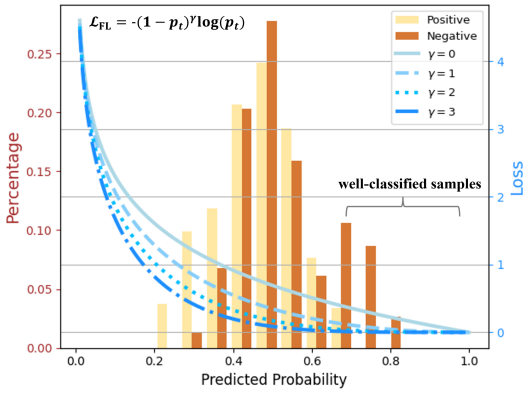

Figure 2: Distribution histogram of predicted probabilities of positive and negative event pairs and the visualized loss with focal parameter $\gamma = \{0, 1, 2, 3\}$.

(1) **w/ a complete graph**, which connects all the nodes in the event relational graph (without the criss-cross constraint mentioned in Section 3.2.1). Compared with the full ERGO model (both $\text{BERT}_{\text{BASE}}$ and $\text{Lonformer}_{\text{BASE}}$), ERGO (w/ a complete graph) clearly decreases the performance, which demonstrates the effectiveness of the handy design of criss-cross constraints.

(2) **w/ GCN**, which replaces the RGT in Section 3.2.2 with a well-known GNN model, GCN. It can be seen that **(i)** ERGO (w/ GCN) also performs better or competitive than other baselines. This indicates that our framework is flexible to other GNNs, and the main improvement comes from our new formulation of the ECI task. **(ii)** the full ERGO model clearly outperforms ERGO (w/ GCN), which validates the effectiveness of our RGT model.

(3) **w/o focal factor**, which sets the focusing parameter $\gamma = 0$ (in Section 3.3) and thus makes the focal loss degenerate into standard CE loss. Compared with the full ERGO model, ERGO (w/o focal factor) also decreases performance. This highlights the effectiveness of penalizing hard samples via an adaptive focal loss in the ECI task.

4.6 Dealing with the Imbalance Issue

In Figure 2, we show the distribution histogram of the predicted probability after the first training epoch for positive and negative samples, respectively (denoted by the bars). The predicted probability of x-axis reflects the difficulty of samples (i.e., the lower, the harder), and the curves denote loss — how much penalization on the corresponding samples during learning. From the histogram, we can find: **(1)** the model is less confident about

Teen Arrested$_1$ in Shooting of Hero Brooklyn Mom
Posted Oct 26, 2011 8:57 AM CDT
An 18-year-old gang member has **confessed** to killing a pregnant mom , who **died** on Friday as she **shielded** a group of children from bullets, but insisted he "did not mean to shoot the ladies," sources tell the New York Daily News.
In addition to Zurana Horton-who was a mother of 12-another mom and an 11-year-old girl were wounded by rooftop sniper Andrew Lopez, who told police his dozen rounds were intended for members of a rival gang.
Lopez has been **charged** with **murder**; his two half-brothers, 17 and 22, were also arrested$_2$.

No.	Event Pair	GT	BERT	ERGO
1	(Shooting, killing)	Yes	*No*	Yes
2	(killing, arrested$_2$)	Yes	*No*	Yes
3	(Shooting, Arrested$_1$)	Yes	Yes	Yes
4	(Arrested$_1$, arrested$_2$)	No	*Yes*	No
5	(Shooting, wounded)	Yes	*No*	Yes
6	(wounded, arrested$_2$)	No	No	No
7	(Shooting, arrested$_2$)	Yes	*No*	Yes
…	…	…	…	…

Figure 3: The case study of BERT baseline and our proposed ERGO, where "GT" denotes the ground truth class, and the right two columns are the output of BERT and ERGO (italic red color means wrong prediction). The thickness of arrows represents the size of attention values, and the bold green arrows show a possible reasoning path.

positives than negatives, i.e., the left-of-center distributed bars of positives. This matches our intuition that the imbalance issue brings a great challenge of false-negative predictions to ECI. **(2)** we visualize focal loss with γ values $\in \{0, 1, 2, 3\}$. The top solid blue curve ($\gamma = 0$) can be seen as the standard CE loss. As γ increases, the shape of focal loss moves to the bottom left corner. That is, the learning of ERGO pays more attention to hard samples. In practice, we find $\gamma = 2$ works best on both datasets, indicating that there is a balance between the focus on simple and difficult samples.

4.7 Case Study

In this section, we conduct a case study to further illustrate an intuitive impression of our proposed ERGO. As shown in Figure 3, We notice that: **(1)** BERT is good at sentence-level ECI (e.g., No.3 event pair), but fails at more complex cross-sentence cases (e.g., No.1, 2, 4, 5, 7). **(2)** By contrast, ERGO can make correct predictions by modeling the global interactions among event pairs. **(3)** Figure 3 shows 3 causal patterns that ERGO could cover: **(i) Transitivity** (No.1, 2, 7 event pairs): knowing both (*Shooting*, *killing*) and (*killing*, *arrested*$_2$) have causal relations, we could infer that (*Shooting*, *arrested*$_2$) has a causal relation; **(ii) Implicit Coreference Assistance** (No.3, 4, 7 event pairs) : Given that (*Shooting*, *Arrested*$_1$) has a causal relation and (*Arrested*$_1$, *arrested*$_2$) is coreference, we could infer that (*Shooting*, *arrested*$_2$) has a causal relation, even if the causal relation of (*Arrested*$_1$, *arrested*$_2$) is implicitly modeled. We attribute this to PLMs

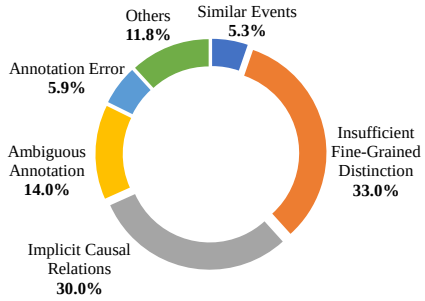

Figure 4: Statistics of Error Types.

that tend to capture coreference relations, such as similar tokens. A piece of supporting evidence is that BERT incorrectly predicts the coreferenced No.4 event pair with a causal relation. **(iii) Deconfounding Negatives** (No.5, 6, 7 event pairs): Knowing (*Shooting*, *wounded*) has a causal relation, although (*wounded*, *arrested*$_2$) does not has a causal relation, it is still possible that (*Shooting*, *arrested*$_2$) has a causal relation through other paths. Correspondingly, as shown in the bottom right, both (*Shooting*, *wounded*) and (*wounded*, *arrested*$_2$) are assigned with very low attention weights, blocking the propagation over these uninformative paths, to avoid the negative confounders contaminating causal transitivity.

4.8 Remaining Challenges

We randomly sample 20 documents of different topics from EventStoryLine, which contains 170 event pairs whose causal relations cannot be correctly predicted by our model. As shown in Figure 4, we manually categorize these pairs into different types and discuss the remaining challenges:

Insufficient Fine-Grained Distinction and Need to Extract Temporal Information (33%) For example, in the following document:

"...Dubai experienced a slight 'tremor' today, ***after*** *a more serious earthquake in Southern Iran, resulting in the evacuation of Emirates Towers and a few other scrapers..."*

The "*tremor*" happens in "*Dubai*" and the "*earthquake*" happens in "*Southern Iran*", they are two different events identified by the temporal indicator "*after*". ERGO incorrectly predicts that there is a causal relation in (*earthquake*, *evacuation*). Future work could consider joint extraction of causal and temporal relations within the document.

Events with Similar Semantics (5.3%) Take the following document as an example:

"...Kenneth Dorsey says the woman accused of killing two co-workers and critically injuring a third at the Kraft plant in Northeast Philly is a good person. And so were the two women she's accused of gunning down with a .357 Magnum, just minutes after she'd been suspended and escorted from the building..."

ERGO incorrectly predicts that there is a causal relation between "*killing*" and "*gunning down*". The reason is that "*killing*" and "*gunning down*" are actually coreference, which suggests a future direction in exploring related tasks.

Implicit Causal Relations (30%) ERGO still fails at many implicit causal relations. For example, the causal relation between "*killing*" and "*suspended*" in the aforementioned document. This is mainly because there are insufficient events for global reasoning and hard negatives bring noise. Clearly, commonsense reasoning will be helpful in this case, since "*suspended*" is an unexpected change that may bring some negative emotions.

Ambiguous Annotation (14%) This type denotes that ambiguous causality within some event pairs. For example, in the following document:

"... A Texas inmate escaped from a prison van near Houston after pulling a gun on two guards who were transporting him between prisons..."

We can think there is a causal relation between "*escaped*" and "*transporting*" because if there is no "*transporting*", the "*inmate*" will have no chance to "*escape*". However, we can also think that there is no causal relation between them because it is not "*transporting*" that directly causes "*escape*".

Finally, as shown in Figure 4, our statistics show that the other errors have to do with annotation errors (5.9%) and more complicated issues that cannot be categorized clearly ("Others", 11.8%).

5 Conclusion

In this paper, we regard DECI as a node classification task by constructing an event relational graph. We propose a novel Event Relational Graph Transformer (ERGO) framework that could capture potential causal chains and mitigate the false positive and false negative issues for DECI. Extensive experiments show great improvements of ERGO under both intra- and inter-sentence settings on two widely used benchmarks. Further analysis provides insights into our approach and the DECI task. In the future, we will consider introducing commonsense reasoning and auxiliary tasks to discover more reliable causality.

Acknowledgments

This work was supported by National Key Research and Development Program of China under Grant No. 2018AAA0101902, NSFC under Grant No. 61532001, the Singapore Ministry of Education (MOE) Academic Research Fund (AcRF) Tier 1 grant, and the RIE2020 Industry Alignment Fund – Industry Collaboration Projects (IAF-ICP) Funding Initiative, as well as cash and in-kind contribution from the industry partner(s).

References

Brandon Beamer and Roxana Girju. 2009. Using a bigram event model to predict causal potential. In *International Conference on Intelligent Text Processing and Computational Linguistics*, pages 430–441. Springer.

Iz Beltagy, Matthew E Peters, and Arman Cohan. 2020. Longformer: The long-document transformer. *ArXiv preprint*, abs/2004.05150.

Jonathan Berant, Vivek Srikumar, Pei-Chun Chen, Abby Vander Linden, Brittany Harding, Brad Huang, Peter Clark, and Christopher D. Manning. 2014. Modeling biological processes for reading comprehension. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1499–1510, Doha, Qatar. Association for Computational Linguistics.

Pengfei Cao, Xinyu Zuo, Yubo Chen, Kang Liu, Jun Zhao, Yuguang Chen, and Weihua Peng. 2021. Knowledge-enriched event causality identification via latent structure induction networks. In *Proceedings of the 59th ACL and the 11th IJCNLP (Volume 1:*

Long Papers), pages 4862–4872, Online. Association for Computational Linguistics.

Tommaso Caselli and Piek Vossen. 2017. The event StoryLine corpus: A new benchmark for causal and temporal relation extraction. In *Proceedings of the Events and Stories in the News Workshop*, pages 77–86, Vancouver, Canada. Association for Computational Linguistics.

Bridget Copley and Fabienne Martin. 2014. *Causation in grammatical structures*, volume 52. Oxford University Press.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 NAACL: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, Minneapolis, Minnesota. Association for Computational Linguistics.

Quang Do, Yee Seng Chan, and Dan Roth. 2011. Minimally supervised event causality identification. In *Proceedings of the 2011 Conference on Empirical Methods in Natural Language Processing*, pages 294–303, Edinburgh, Scotland, UK. Association for Computational Linguistics.

Lei Gao, Prafulla Kumar Choubey, and Ruihong Huang. 2019. Modeling document-level causal structures for event causal relation identification. In *Proceedings of the 2019 NAACL: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 1808–1817, Minneapolis, Minnesota. Association for Computational Linguistics.

Chikara Hashimoto. 2019. Weakly supervised multilingual causality extraction from Wikipedia. In *Proceedings of EMNLP-IJCNLP 2019*, pages 2988–2999, Hong Kong, China. Association for Computational Linguistics.

Chikara Hashimoto, Kentaro Torisawa, Julien Kloetzer, Motoki Sano, István Varga, Jong-Hoon Oh, and Yutaka Kidawara. 2014. Toward future scenario generation: Extracting event causality exploiting semantic relation, context, and association features. In *Proceedings of the 52nd ACL (Volume 1: Long Papers)*, pages 987–997, Baltimore, Maryland. Association for Computational Linguistics.

Christopher Hidey and Kathy McKeown. 2016. Identifying causal relations using parallel Wikipedia articles. In *Proceedings of the 54th ACL (Volume 1: Long Papers)*, pages 1424–1433, Berlin, Germany. Association for Computational Linguistics.

Zhichao Hu, Elahe Rahimtoroghi, and Marilyn Walker. 2017. Inference of fine-grained event causality from blogs and films. In *Proceedings of the Events and Stories in the News Workshop*, pages 52–58, Vancouver, Canada. Association for Computational Linguistics.

Kazuma Kadowaki, Ryu Iida, Kentaro Torisawa, Jong-Hoon Oh, and Julien Kloetzer. 2019. Event causality recognition exploiting multiple annotators' judgments and background knowledge. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 5816–5822, Hong Kong, China. Association for Computational Linguistics.

Thomas N. Kipf and Max Welling. 2017. Semi-supervised classification with graph convolutional networks. In *5th ICLR, 2017, Toulon, France, April 24-26, 2017, Conference Track Proceedings*. OpenReview.net.

Tsung-Yi Lin, Priya Goyal, Ross B. Girshick, Kaiming He, and Piotr Dollár. 2017. Focal loss for dense object detection. In *IEEE, ICCV 2017, Venice, Italy, October 22-29, 2017*, pages 2999–3007. IEEE Computer Society.

Jian Liu, Yubo Chen, and Jun Zhao. 2020. Knowledge enhanced event causality identification with mention masking generalizations. In *Proceedings of the Twenty-Ninth International Joint Conference on Artificial Intelligence, IJCAI 2020*, pages 3608–3614. ijcai.org.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Roberta: A robustly optimized bert pretraining approach. *ArXiv preprint*, abs/1907.11692.

Ilya Loshchilov and Frank Hutter. 2019. Decoupled weight decay regularization. In *7th ICLR 2019, New Orleans, LA, USA, May 6-9, 2019*. OpenReview.net.

Yubo Ma, Zehao Wang, Yixin Cao, Mukai Li, Meiqi Chen, Kun Wang, and Jing Shao. 2022a. Prompt for extraction? PAIE: Prompting argument interaction for event argument extraction. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 6759–6774, Dublin, Ireland. Association for Computational Linguistics.

Yubo Ma, Zehao Wang, Mukai Li, Yixin Cao, Meiqi Chen, Xinze Li, Wenqi Sun, Kunquan Deng, Kun Wang, Aixin Sun, and Jing Shao. 2022b. MMEKG: Multi-modal event knowledge graph towards universal representation across modalities. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics: System Demonstrations*, pages 231–239, Dublin, Ireland. Association for Computational Linguistics.

Paramita Mirza. 2014. Extracting temporal and causal relations between events. In *Proceedings of the ACL 2014 Student Research Workshop*, pages 10–17, Baltimore, Maryland, USA. Association for Computational Linguistics.

Qiang Ning, Zhili Feng, Hao Wu, and Dan Roth. 2018. Joint reasoning for temporal and causal relations. In *Proceedings of the 56th ACL (Volume 1: Long Papers)*, pages 2278–2288, Melbourne, Australia. Association for Computational Linguistics.

Jong-Hoon Oh, Kentaro Torisawa, Chikara Hashimoto, Ryu Iida, Masahiro Tanaka, and Julien Kloetzer. 2016. A semi-supervised learning approach to why-question answering. In *Proceedings of the Thirtieth AAAI Conference on Artificial Intelligence, February 12-17, 2016, Phoenix, Arizona, USA*, pages 3022–3029. AAAI Press.

Laurie Ann Paul, Ned Hall, and Edward Jonathan Hall. 2013. *Causation: A user's guide*. Oxford University Press.

Mehwish Riaz and Roxana Girju. 2010. Another look at causality: Discovering scenario-specific contingency relationships with no supervision. In *2010 IEEE Fourth International Conference on Semantic Computing*, pages 361–368. IEEE.

Mehwish Riaz and Roxana Girju. 2013. Toward a better understanding of causality between verbal events: Extraction and analysis of the causal power of verb-verb associations. In *Proceedings of the SIGDIAL 2013 Conference*, pages 21–30, Metz, France. Association for Computational Linguistics.

Mehwish Riaz and Roxana Girju. 2014a. In-depth exploitation of noun and verb semantics to identify causation in verb-noun pairs. In *Proceedings of the 15th Annual Meeting of the Special Interest Group on Discourse and Dialogue (SIGDIAL)*, pages 161–170, Philadelphia, PA, U.S.A. Association for Computational Linguistics.

Mehwish Riaz and Roxana Girju. 2014b. Recognizing causality in verb-noun pairs via noun and verb semantics. In *Proceedings of the EACL 2014 Workshop on Computational Approaches to Causality in Language (CAtoCL)*, pages 48–57, Gothenburg, Sweden. Association for Computational Linguistics.

Robyn Speer, Joshua Chin, and Catherine Havasi. 2017. Conceptnet 5.5: An open multilingual graph of general knowledge. In *Proceedings of the Thirty-First AAAI Conference on Artificial Intelligence, February 4-9, 2017, San Francisco, California, USA*, pages 4444–4451. AAAI Press.

Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. 2014. Dropout: a simple way to prevent neural networks from overfitting. *The journal of machine learning research*, 15(1):1929–1958.

Qingyu Tan, Ruidan He, Lidong Bing, and Hwee Tou Ng. 2022. Document-level relation extraction with adaptive focal loss and knowledge distillation. In *Findings of ACL*.

Minh Tran Phu and Thien Huu Nguyen. 2021. Graph convolutional networks for event causality identification with rich document-level structures. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 3480–3490, Online. Association for Computational Linguistics.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, December 4-9, 2017, Long Beach, CA, USA*, pages 5998–6008.

Yuan Yao, Deming Ye, Peng Li, Xu Han, Yankai Lin, Zhenghao Liu, Zhiyuan Liu, Lixin Huang, Jie Zhou, and Maosong Sun. 2019. DocRED: A large-scale document-level relation extraction dataset. In *Proceedings of the 57th ACL*, pages 764–777, Florence, Italy. Association for Computational Linguistics.

Wenpeng Yin, Dragomir Radev, and Caiming Xiong. 2021. DocNLI: A large-scale dataset for document-level natural language inference. In *Findings of ACL-IJCNLP 2021*, pages 4913–4922, Online. Association for Computational Linguistics.

Kun Zhao, Donghong Ji, Fazhi He, Yijiang Liu, and Yafeng Ren. 2021. Document-level event causality identification via graph inference mechanism. *Information Sciences*, 561:115–129.

Xinyu Zuo, Pengfei Cao, Yubo Chen, Kang Liu, Jun Zhao, Weihua Peng, and Yuguang Chen. 2021a. Improving event causality identification via self-supervised representation learning on external causal statement. In *Findings of ACL-IJCNLP 2021*, pages 2162–2172, Online. Association for Computational Linguistics.

Xinyu Zuo, Pengfei Cao, Yubo Chen, Kang Liu, Jun Zhao, Weihua Peng, and Yuguang Chen. 2021b. LearnDA: Learnable knowledge-guided data augmentation for event causality identification. In *Proceedings of the 59th ACL and the 11th IJCNLP (Volume 1: Long Papers)*, pages 3558–3571, Online. Association for Computational Linguistics.

Xinyu Zuo, Yubo Chen, Kang Liu, and Jun Zhao. 2020. KnowDis: Knowledge enhanced data augmentation for event causality detection via distant supervision. In *Proceedings of the 28th COLING*, pages 1544–1550, Barcelona, Spain (Online). International Committee on Computational Linguistics.

DRK: Discriminative Rule-based Knowledge for Relieving Prediction Confusions in Few-shot Relation Extraction

Mengru Wang, Jianming Zheng*, Fei Cai, Taihua Shao, Honghui Chen
Science and Technology on Information Systems Engineering Laboratory,
National University of Defense Technology, China
{wangmengru20,zhengjianming12,caifei08,shaotaihua13,chenhonghui}@nudt.edu.cn

Abstract

Few-shot relation extraction aims to identify the relation type between entities in a given text in the low-resource scenario. Albeit much progress, existing meta-learning methods still fall into **prediction confusions** owing to the limited inference ability over shallow text features. To relieve these confusions, this paper proposes a *discriminative rule-based knowledge* (DRK) method. Specifically, DRK adopts a logic-aware inference module to ease the word-overlap confusion, which introduces a logic rule to constrain the inference process, thereby avoiding the adverse effect of shallow text features. Also, DRK employs a discrimination finding module to alleviate the entity-type confusion, which explores distinguishable text features via a hierarchical contrastive learning. We conduct extensive experiments on four types of meta tasks and the results show promising improvements from DRK (6.0% accuracy gains on average). Besides, error analyses reveal the word-overlap and entity-type errors are the main courses of mispredictions in few-shot relation extraction.

1 Introduction

With the emergence of new relation types and ever-increasing annotation costs, traditional data-driven relation extraction (RE) methods cannot survive in this low-data regime (Wang et al., 2021; Chia et al.). Therefore, the task of few-shot relation extraction is proposed to cope with such a low-resource dilemma. In few-shot RE, meta-learning (ML) has been extensively employed and attained promising performance, the core of which is to learn the generalization ability from the data-rich classes to help predict the data-scarce classes (Han et al., 2018). These ML approaches can be roughly divided into two categories: basic ML only accessible to raw sentence text, e.g., prototype (Snell et al., 2017) and MAML (Finn et al., 2017), and knowledge-based ML with the additional external knowledge (Zheng et al., 2020a), e.g., TD-proto (Yang et al., 2020) and REGRAB (Qu et al., 2020).

Albeit much progress, the insufficient labeled data forces existing ML methods to make shallow inferences based on superficial text features, e.g., the word overlap (Utama et al., 2021) and the matched entity type (Brody et al., 2021). In this light, as shown in Fig.1, when the support instances of some relations exhibit massive word overlaps or have a matched entity-type pair, these ML methods inevitably get caught in the **prediction confusions** (Wang et al., 2020). Based on the taxonomy of similar pattern, prediction confusions can be further classified into the word-overlap confusion and the entity-type confusion as shown in Fig.1.

For the word-overlap confusion, intuitively, incorporating external knowledge, e.g., relation descriptions (Yang et al., 2020), an entity-level knowledge graph (KG) (Roy and Pan, 2021) or entity types (Sainz et al., 2021; Hao et al., 2019; Yang et al., 2021), into RE models can ease this semantic uncertainty to some extent. However, this simple incorporation (concatenation in the majority) just provides additional information, and cannot alter the fact that existing ML methods still process the shallow inferences. Even worse, the lengthened text may exceed the input limit and introduce noise, thereby degrading the model performance. Besides, the entity-type confusion can not be relieved with the introduction of external knowledge. Since external information generally contains entity type information, the simple introduction cannot work in relations whose entity-type pairs are matched in head and tail positions. Although the entity masking (Li et al., 2021) and some data augmentations (Brody et al., 2021) can increase the relation separability by changing the data format, these solutions still rely on the superficial features and fail to grasp the subtle semantic differences.

In this paper, we attempt to solve these two pre-

* Corresponding author.

Proceedings of the 29th International Conference on Computational Linguistics, pages 2129–2140
October 12–17, 2022.

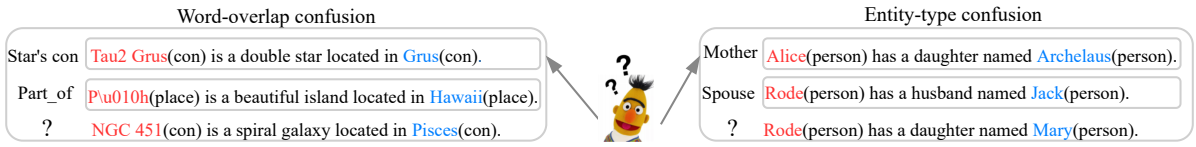

Figure 1: **Prediction confusions** in few-shot RE, including the word-overlap confusion (left) and the entity-type confusion (right). The word-overlap confusion has relations with the same grammatical structure and massive word overlaps but mismatched entity-type pair ("con-con" vs. "place-place"), while the entity-type confusion has relations with the same entity-type pair ("person-person" vs. "person-person"), where the head and tail entities are indicated in red and blue respectively and the word constellation is abbreviated to "con".

diction confusions by proposing a *discriminative rule-based knowledge* (DRK) method that consists of a logic-aware inference module and a discrimination finding module. Specifically, the logic-aware inference module relieves the word-overlap confusion by a rule-based incorporation of an ontology KG (Hao et al., 2019). Different from the simple concatenation in previous work , this module employs a logic rule (e.g., the "Star's con" relation happens when entity-type pair belongs to "con-con" rather than "place-place", shown in Fig.1) to constrain the model inference. In this way, the rule-based knowledge can not only mitigate the adverse effect of shallow text features but also provide a new inference direction. Unfortunately, this introduced rule is heavily dependent on the discrepancy of entity types, thereby losing its advantages in the entity-type confusion. To clear this confusion, the discrimination finding module is proposed, which employs a hierarchical contrastive learning strategy (the instance level and the category level) to further explore distinguishable text features.

We compare DRK with ML methods with and without external knowledge on four types of meta-tasks in FewRel 1.0 (Han et al., 2018). The results demonstrate: 1) DRK achieves significant improvements over the state-of-the-art baseline in terms of accuracy; 2) the logic-aware inference module is more effective than the discrimination finding one; 3) error analyses reveal that the word-overlap and entity-type confusions are the main error courses of mispredictions in few-shot RE, and DRK can effectively relieve these two prediction confusions.

Our key contributions are: 1) a *discriminative rule-based knowledge* (DRK) method for few-shot RE, which targets to relieve prediction confusions; 2) a logic-aware inference module for the word-overlap confusion by the rule-based knowledge incorporation, which opens a new inference direction for few-shot RE; 3) a discrimination finding module for the entity-type confusion by the hierarchical contrastive learning, which explores subtle semantic differences; 4) extensive experiments demonstrating the effectiveness of our proposals.

2 Related Work

2.1 Basic ML for Few-shot RE

In few-shot RE, basic ML generally infers relation type based on original text as the sole input, which can be roughly divided into two types, i.e., optimization-based ML and metric-based ML (Huang et al., 2021). Optimization-based ML focuses on finding good initialization points for parameters that can readily generalize to novel relation types within few gradient steps. For example, MAML adopts a model-agnostic gradient update strategy to produce good gradient stand points of parameters for novel relations (Finn et al., 2017). To reduce the computational complexity of MAML, Reptile only employs first-order derivatives to update parameters (Nichol et al., 2018). Metric-based ML aims to design a metric function that clearly measures the distance of instances in the embedding space. For instance, prototypical networks identify the relation labels by computing the similarity between query instances and the relation prototypes (Snell et al., 2017). Following this work, quantities of methods devote to improving the performance of prototype, e.g., Gao et al. (2019a) modify the representation of the prototype by highlighting the crucial instances and features, and Sun et al. (2019) redefine the prototype via a hierarchical attention scheme.

However, these ML methods can't make reliable inferences in the low-resource scenario, and easily fall into the prediction confusions. To clear these confusions, our proposal DRK introduces an ontology-level KG by the rule-based incorporation to avoid the adverse effect of shallow text inferences.

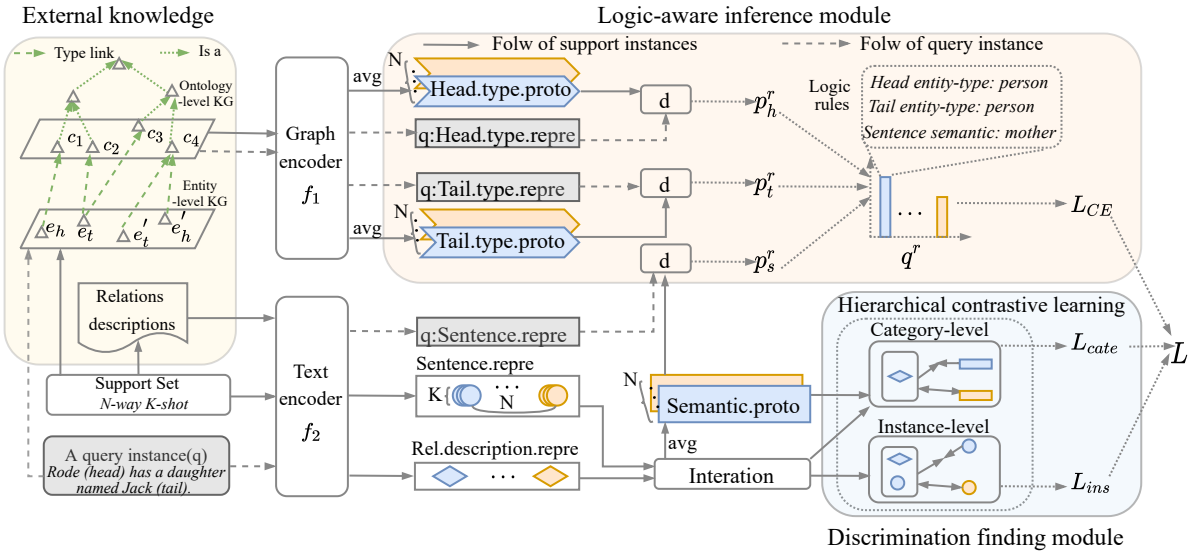

Figure 2: Framework of discriminative rule-based knowledge. For naming these process or submodules, "representation" and "prototype" are abbreviated as "repre" and "proto", respectively.

2.2 Knowledge-based ML for Few-shot RE

External knowledge has been widely employed in few-shot RE due to the abundant auxiliary semantic information. Based on the data structure, external knowledge can be divided into unstructured text and structured knowledge graph. For the unstructured text, most work focuses on leveraging relation and entity descriptions to enhance contextual semantic representation (Yang et al., 2020). Compared with the unstructured text, massive endeavors have been devoted to the structured knowledge graph. For example, Liu et al. (2020) inject entity-level triplets into text by a sentence tree, Roy and Pan (2021) integrate entity-level KG and text by several fusion techniques, and Yang et al. (2021) leverage an ontology-level knowledge graph to provide clues for the entity type and designed a fusion module based on self-attention to bridge the gap between the embeddings of text and the relation types. Besides, Sainz et al. (2021) annotate the entity type manually for each instance.

In essence, the above knowledge-based ML methods are still built on shallow inferences, suffering from prediction confusions. Our proposal adopts the logic-aware inference module and the discrimination finding module to clear the word-overlap and entity-type confusions, respectively.

3 Approaches

In this section, we first introduce the task definition as well as the framework of DRK in §3.1. Then, we detail the logic-aware inference module in §3.2 and the discrimination finding module in §3.3.

3.1 Task definition and model framework

RE. Formally, given a L_s-word instance s with the head and tail entities e_h and e_t, i.e., $s = \{w_1, \cdots, e_h, \cdots, e_t, \cdots, w_{L_s}\}$, the goal of RE is to correctly extract the relation triplet (e_h, e_t, r), where r is a relation label belonging to the predefined relation label set $\mathcal{R}$.

Few-shot RE. Following the typical N-way K-shot setting, a meta task consists of a support set $\mathcal{S}$ and a query set $\mathcal{Q}$. $\mathcal{S} = \{S_i\}_{i=1}^{N}$ has N novel relations $\mathcal{R} = \{r_i\}_{i=1}^{N}$, each relation r has $S_r = \{s_r^i\}_{i=1}^{K}$ containing K instances. Few-shot relation extraction targets to predict the relation label $r_Q \in \mathcal{R}$ of the query set $\mathcal{Q}$ based on the limited labeled data $\mathcal{S}$.

It is noteworthy that the few-shot RE models are trained on meta tasks sampled from instances of base relation labels then tested on meta tasks sampled from instances of novel relation labels(Zheng et al., 2021). And the base and novel relation labels are disjoint.

External Knowledge. Based on the data structure, external knowledge can be classified into the structured entity-level KG G_e, the structured ontology-level KG G_o and the unstructured knowledge G_t.

In specific, G_e consists of a set of relation triplets $\{(e_h, e_t, r) \in \mathcal{E} \times \mathcal{E} \times \mathcal{R}_e\}$, where $\mathcal{E}$ and $\mathcal{R}_e$ are the entity node set and the entity-level relation set, respectively. Similar to G_e, G_o can also be formu-

lated as $\{(e_h, e_t, r) \in \mathcal{C} \times \mathcal{C} \times \mathcal{R}_o\}$, where $\mathcal{C}$, $\mathcal{R}_o$ are the ontology node set and the ontology-level relation label set, respectively. Note that there exhibits an "instance of" relation $\hat{r}$ between $\mathcal{E}$ and $\mathcal{C}$, i.e., $\{(e_h, e_t, \hat{r}) \in \mathcal{E} \times \mathcal{C} \times \hat{r}\}$. For unstructured knowledge G_t, it describes each relation r from the given relation set $\mathcal{R}$ with a L_r-word sequence relation description $a_r = \{w_i\}_{i=1}^{L_r}$.

Framework of DRK. As shown in Fig 2, In each iteration step, the support set $\mathcal{S}$ and the query set $\mathcal{Q}$ are first fed into external knowledge to obtain their corresponding structured and unstructured information that are encoded by the graph encoder f_1 and the text encoder f_2, respectively. And then, the logic-aware inference module employs the prototypes on structured and unstructured information to get the head, tail and contextual semantic representations, which are further constrained by the logic rule to compute L_{CE}. Furthermore, the discrimination finding module leverages the category-level contrastive learning L_{cate} and the instance-level contrastive learning L_{ins} to mine the distinguishable text features.

3.2 Logic-aware inference module

This module employs structured and unstructured knowledge encoders to get the entity-type and contextual semantic representations, which are further constrained by the logic rule.

3.2.1 Structured knowledge encoder

Each entity belongs a set of entity types, e.g., the entity "Biden" stemmed from the "politician" and "person" types. Take the head entity e_h for example, the ontology KG G_o[1] is adopted to construct it's type set C_h:

$$C_h = B\left(e_h\right), \tag{1}$$

where B is a link function base on the "instance_of" relation between e_h and c_h. Then a graph neural network encoder f_1 is employed to obtain the head entity type representation $\mathbf{H}_{e_h}$ of e_h:

$$\mathbf{H}_{e_h} = \frac{1}{|C_h|} \sum_{c \in C_h} f_1\left(c\right), \tag{2}$$

where $|C_h|$ is the number of elements in the type set. For any relation r, its support instances S_r contain K instances, hence the head entity type prototype of the relation r can be formulated as:

$$\mathbf{m}_h^r = \frac{1}{K} \sum_{e_h \in S_r} \mathbf{H}_{e_h}. \tag{3}$$

Analogously, the tail entity type prototype $\mathbf{m}_t^r$ of the relation r can also be obtained. And, for any query instance q in the query set $\mathcal{Q}$, similar encoding process can be employed to get its head and tail entity type representations: $\mathbf{H}_h^q$ and $\mathbf{H}_t^q$.

3.2.2 Unstructured knowledge encoder

Given a relation r with support instances $\{s_r^i\}_{i=1}^K$, the unstructured knowledge G_t [2] is retrieved to obtain the relation description a_r. Then, the support instances and the relation description are respectively fed into a text encoder $f_2(\cdot)$ to get the instance representations $\{\mathbf{H}^0_{s_r^i} \in \mathbb{R}^{L_s \times d}\}_{i=1}^K$ and the relation-describing representations $\mathbf{H}_{a_r} \in \mathbb{R}^{L_r \times d}$, where d is the embedding dimension of f_2. To highlight the entities (Soares et al., 2019), the start tokens of the head and tail entities are concatenated to get the entity representations $\{\mathbf{H}^e_{s_r^i} \in \mathbb{R}^{1 \times 2d}\}_{i=1}^K$ that is different from Eq.(2).

Interaction. On the one hand, the relation description a_r elaborates the relation r, a_r can be utilized to further refine the instance representations. Taking the instance s_r^i as an example, its refined instance representation can be formulated as:

$$\begin{aligned} \mathbf{H}^{a_r}_{s_r^i} &= \sum_{j=1}^{L_s} \alpha_j \mathbf{H}^0_{s_r^i}[j:], \\ \boldsymbol{\alpha} &= soft\max\left(sum\left(\mathbf{H}^0_{s_r^i}\mathbf{H}^T_{a_r}\right)\right), \end{aligned} \tag{4}$$

where $sum(\cdot)$ is a row-wise summation function, and hence the attention weight $\boldsymbol{\alpha} = \{\alpha_j\}_{j=1}^{L_s}$ attends over the instance tokens. On the other hand, the relation description, as a highly-concise summary, also requires support instances to express its semantics. Specifically, for a specific relation r, its each support instance $\mathbf{H}^0_{s_r^i}$ attends its relation description a_r to obtain the instance-aware relation-describing representation $\mathbf{H}^{s_r^i}_{a_r}$:

$$\begin{aligned} \mathbf{H}^{s_r^i}_{a_r} &= \sum_{j=1}^{L_r} \beta_j \mathbf{H}_{a_r}[j:], \\ \boldsymbol{\beta} &= soft\max\left(sum\left(\mathbf{H}_{a_r}\left(\mathbf{H}^0_{s_r^i}\right)^T\right)\right), \end{aligned} \tag{5}$$

[1] The construct process of G_o can refer to https://github.com/imJiawen/KEFDA

[2] All the relation descriptions in G_t is shown in Table 1 of Appendix A.1

where $\boldsymbol{\beta} = \{\beta_j\}_{j=1}^{L_r}$ is the attention weight over the relation description tokens and $\mathbf{H}_{a_r}^{s_r^i}$ elaborates the instantiation traits of the relation r.

Contextual semantic representation. After the interaction between the instance text and the relation description, the final contextual semantic representation can be further formulated as:

$$\mathbf{H}_{s_r^i} = [\mathbf{H}_{s_r^i}^e; \mathbf{H}_{s_r^i}^{a_r}] + Mul(\mathbf{H}_{a_r}^{s_r^i}), \tag{6}$$

where $\mathbf{H}_{s_r^i}^e \in \mathbb{R}^{1\times 2d}$, $\mathbf{H}_{a_r}^{s_r^i} \in \mathbb{R}^{1\times d}$, $\mathbf{H}_{s_r^i}^{a_r} \in \mathbb{R}^{1\times d}$ and $\mathbf{H}_{s_r^i} \in \mathbb{R}^{1\times 3d}$. $Mul(\cdot)$ is a multi-layer perceptron that converts the dimension from d to 3d.

Similar to Eq.(3), the contextual semantic prototype of r can be obtained:

$$\mathbf{m}_s^r = \frac{1}{K}\sum_{i=1}^{K}\mathbf{H}_{s_r^i}. \tag{7}$$

Note that any instance q in the query set $\mathcal{Q}$ does not have the relation description. Therefore, the refined instance representations in Eq.(4) is replaced with the average pooling and the $Mul(\cdot)$ operation in Eq.(6) is removed when defining the contextual semantic representation of q, i.e., $\mathbf{H}_s^q$.

3.2.3 Logic rule

For RE, this logic rule is assumed as a set of conditions that should be occurred simultaneously (Han et al., 2021a). Taking the "Mother" relation in Fig.1 as an example, whether an instance expresses this relation must satisfy three conditions: 1) the head entity is the person type; 2) the tail entity is the person type; 3) the contextual semantics of this instance describes the "Mother" relation. Based on this logic rule, such three conditions can correspondingly be transformed into three probabilities.

Specifically, for a query instance q, the probability of the head entity type belonging to relation r is:

$$p_h^r = \frac{\exp\left(d\left(\mathbf{H}_h^q, \mathbf{m}_h^r\right)\right)}{\sum_{n=1}^{N}\exp\left(d\left(\mathbf{H}_h^q, \mathbf{m}_h^n\right)\right)}, \tag{8}$$

where $d(\cdot,\cdot)$ indicates the dot product. Analogous to Eq.(8), the probabilities of the tail entity and the contextual semantics belonging to the relation r can also be obtained, i.e., p_t^r and p_s^r. With these three probabilities, the final probability of q expressing the relation r is calculated as:

$$q^r = p_h^r \cdot p_t^r \cdot p_s^r. \tag{9}$$

Consequently, the cross entropy loss is used to optimize the RE parameters:

$$L_{CE} = -\sum_{q\in\mathcal{Q}}\log\left(q^r\right), \tag{10}$$

where the ground-truth relation label of the query instance q is r.

3.3 Discrimination finding module

Despite effective in the word-overlap confusion, the logic-aware inference module cannot handle the entity-type confusion with the matched entity-type pair. Then we propose a hierarchical contrastive learning (including the the instance level and the category level) to mine subtle differences of relation instances.

3.3.1 Instance-level contrastive learning

In a support set, we formulate the instance-level contrastive (He et al., 2020) as follows:

$$L_{\text{ins}} = \frac{-1}{NK^2}\sum_{r=1}^{N}\sum_{i=1}^{K} log\frac{\sum_{j=1}^{K} exp(\mathbf{H}_{s_r^i}\cdot\mathbf{H}_{s_r^j}/\tau_1)}{\sum_{r'\neq r}\sum_{k=1}^{K}(\mathbf{H}_{s_r^i}\cdot\mathbf{H}_{s_{r'}^k}/\tau_1)}, \tag{11}$$

where τ_1 is a temperature hyperparameter, $\mathbf{H}_{s_r^i}$ and $\mathbf{H}_{s_r^j}$ belonging to the same relation form positives and L_{ins} aims to pull the positives closer.

3.3.2 Category-level contrastive learning.

Similar to Eq.(3), the instance-aware relation-describing prototype of r can be obtained, i.e.,

$$\mathbf{m}_a^r = \frac{1}{K}\sum_{i=1}^{K}\mathbf{H}_{a_r}^{s_r^i}. \tag{12}$$

This prototype $\mathbf{m}_a^r$ summarizes the instance-aware traits of relation r. When encountering the prediction confusion, these traits can help RE models to distinguish the subtle differences of instances.

In a support set, the contextual semantic prototype $\mathbf{m}_s^r$ must be close to the belonged relation-describing prototype $\mathbf{m}_a^r$, and keep away from the others. Hence, the category-level contrastive learning is formulated as:

$$L_{cate} = \frac{-1}{N}\sum_{r=1}^{N}\log\frac{\exp\left(\mathbf{m}_s^r\cdot\mathbf{m}_a^r/\tau_2\right)}{\sum_{j\neq r}^{N}\exp\left(\mathbf{m}_s^r\cdot\mathbf{m}_a^j/\tau_2\right)}, \tag{13}$$

where τ_2 is a temperature hyperparameter.

In all, the overall training objective is:

$$L = L_{CE} + L_{ins} + L_{cate}, \tag{14}$$

Task	#Rel	#Ins	Len	Link(%)
Training	50	35000	25	98.35
validation	14	9800	24	98.46
testing	16	11200	24	98.70

Table 1: Statistics of FewRel 1.0. "#Rel" and "#Ins" denote the number of relations and instances, respectively. "Len" means the average token length of instances. "Link" demonstrates the probability of an entity linking to a corresponding type in G_o.

4 Experiments

4.1 Dataset

Experiments are conducted on FewRel 1.0 [3] (Han et al., 2018) that consists of 100 relations extracted from Wikipedia. Since the test set with 20 relations is unpublished, following previous work (Yang et al., 2020, 2021), we re-split the published 80 relations into 50, 14 and 16 for training, validation and testing, respectively. The statistics of the re-splitted dataset can refer to Tabel 1. The ontology-level KG we adopt in this paper comes from KEFDA [4]. Besides, the unstructured text knowledge for test data is represented in Table 5 in Appendix A.1.

4.2 Model Configuration

The model configurations are kept the same among all discussed models, including our proposal and the selected baselines. In detail, following (Han et al., 2018, 2021b), we use the classification accuracy to evaluate the performance of DRK on four typical meta tasks (Han et al., 2018): 5-way 1-shot, 5-way 5-shot, 10-way 1-shot and 10-way 5-shot. We use BERT$_{base}$ (Soares et al., 2019) as the text feature encoder f_2 and apply DistMult (Yang et al., 2015) as the graph neural network (Zheng et al., 2020b) encoder f_1. The embedding dimension of f_2 and f_1 is 768 and 256, respectively. We train and test each model with 20,000 and 10,000 iteration steps, respectively. Besides, we set the batch size to 4, the weight decay to 2×10^{-5}, the max sentence length L_s to 128, τ_1 to 1, and τ_2 to 1.

4.3 Baselines

As mentioned above, we introduce several basic ML and knowledge-based ML methods as baselines. In specific, the basic ML approaches contain **GNN** (Satorras and Estrach) that considers all the instances in a meta task as nodes in a graph, then leverages the label propagation to infer relations, **Snail** (Mishra et al., 2018) that combines temporal convolutions and soft attention to learn information from past experiences to predict relations, **Siamese** (Mishra et al.) that employs siamese neural networks to distinguish relations, **Proto** (Snell et al., 2017) that calculates the similarity between the query instance and the relation prototypes to predict relation types, and **BERT-PAIR** (Gao et al., 2019b) that concatenates the query instances with all labeled support instances as pair sequences, then identifies the relation type through the similarity in each pair. ; while the knowledge-based ML approaches include **ConceptFERE** [5] (Yang et al., 2021) that concatenates the entity type with sentence text by a self-attention fusion, and **KEFDA** (Zhang et al., 2021) that also concatenates the entity type and sentence text, and leverages a relation-meta learning network (Chen et al., 2019) to learn implicit relation matching. Note that all the baselines and our proposal use the same text feature and graph encoders for fair comparisons.

4.4 Overall Evaluation

To examine the few-shot relation extraction performance of our proposal as well as the baselines, we report the overall evaluation results on four types of meta tasks in Table 2.

Among all baselines, the knowledge-based ML ConceptFERE achieves the best performance with the average 1.11% improvement against BERT-PAIR (the best basic ML) on the four meta tasks. This performance advancement reflects the benefits brought by the external knowledge that provides auxiliary clues for relation extraction. Compared with ConceptFERE, our proposal DRK can further enhance the extraction performance with the respective improvements of 6.72%, 2.30%, 10.73% and 4.17% on the four meta task. Such model improvements prove the effectiveness of DRK by relieving the prediction confusions.

Clearly, as the shot number decreases, all discussed models get impaired. For example, comparing meta tasks on 5-way 1-shot and 5-way 5-shot, the best baseline ConceptFERE loses the relative 6.71% model performance (9.56% in 10-way 1-shot vs. 10-way 5-shot). These phenomena show that these models are sensitive to the shot number,

[3] https://github.com/thunlp/FewRel
[4] https://github.com/imJiawen/KEFDA
[5] We use the ConceptFERE(simple) version here to allow for the computation overheads in the training of "5-way 5-shot" and "10-way 5-shot" meta tasks.

Model	5-way 1-shot	5-way 5-shot	10-way 1-shot	10-way 5-shot
Snail	57.82	80.53	50.40	68.11
GNN	66.48	82.65	48.14	73.22
Siamese	81.29	88.18	71.00	81.12
Proto	78.59	88.99	64.07	81.80
BERT-PAIR	82.57	89.00	73.37	81.81
KEFDA	80.46	89.88	68.23	81.49
ConceptFERE†	84.28	90.34	74.00	81.82
DRK(Our)	**89.94**$^{\triangle}$	**92.42**$^{\triangle}$	**81.94**$^{\triangle}$	**85.23**$^{\triangle}$

Table 2: Few-shot relation extraction performance in terms of accuracy(%) on four types of meta tasks. † means the results are quoted from the original paper (Yang et al., 2021). Results of the best baseline and the best performer in each column are underlined and **boldfaced**, respectively. Statistical significance of pairwise differences of DRK vs. the best baseline is determined by a t-test ($^{\triangle}$ for $\alpha = 0.05$).

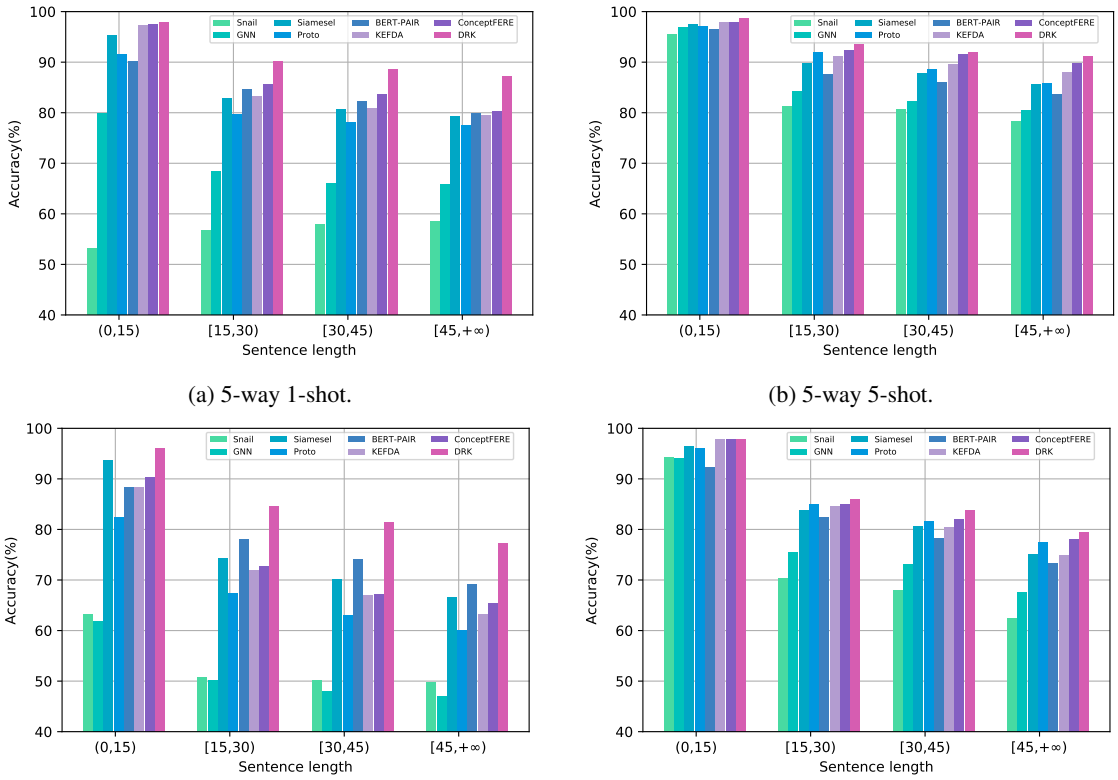

(a) 5-way 1-shot.

(b) 5-way 5-shot.

(c) 10-way 1-shot.

(d) 10-way 5-shot.

Figure 3: Effect on the performance of our proposal and baselines affected by the sentence length.

and prone to fail when the labeled data become fewer. In the same comparisons about the shot number, however, DRK only has 2.68% and 3.86% performance wastage, indicating that DRK is more robust than other methods in the low-data regime.

4.5 Impact of sentence length

To explore the performance of few-shot RE affected by the sentence length, we group the testing performance of our proposal and baselines based on the sentence length L_s. Considering the distribution of testing data, the sentence length is divided into four groups, i.e., $L_s \in (0, 15), [15, 30), [30, 45), [45, +\infty)$. With these sentence groups, the results on the 5-way 1-shot, 5-way 5-shot, 10-way 1-shot and 10-way 5-shot meta task are plotted in Fig.3. Then, we take the performance on the 5-way 1-shot meta task for instance to analyze the results.

In general, with the increase of sentence length,

Model	5-way 1-shot	5-way 5-shot	10-way 1-shot	10-way 5-shot
DRK	89.94	92.42	81.94	85.23
- logic&text	84.08	90.48	79.02	81.90
- logic&type	83.64↓	90.37↓	74.60↓	81.45↓
- instance	88.07	91.80	80.53	83.43
- category	85.22	90.63	79.83	81.63

Table 3: Ablation studies on DRK, where "-logic&text" ("-logic&type") means that the logic-aware inference module is replaced with prototypical networks that do not use entity type from ontology-level KG (the simple concatenation of incorporating entity type from ontology KG), while "-instance" ("-category") denotes the removal of the instance-level (category-level) contrastive learning, and the biggest drop in each column is appended ↓.

almost all models (except for Snail on 5-way 1-shot meta task in Fig.3a) exhibit a ever-decreasing trend in terms of the model performance. This downward trend can be attributed to the fact that short sentences are more concise than long sentences in the grammatic structure, and hence easier for RE models to understand their semantics. In particular, the steep performance degenerations in KEFDA and ConceptFERE shown in Fig.3a further confirm our hypothesis about the existing knowledge-based ML methods. Furthermore, the upward trend of Snail in Fig.3a may be due to the probable error in the low-shot setting. In other three meta tasks, Snail still keeps the decreasing trend.

Next, let's focus on our proposal DRK. Clearly, DRK still presents notable performance advantages over all baselines for each sentence group. For example, compared to the best baseline ConceptFERE on 5-way 1-shot meta task in Fig.3a, DRK achieves 0.47%, 4.51%, 4.90% and 6.96% model improvements in terms of accuracy when $L_s \in (0, 15), [15, 30), [30, 45), [45, +\infty)$, respectively. Interestingly, the improvement magnitude of DRK always keeps pace with the increase of sentence length. This increasing pattern can be explain by the fact that DRK can effectively extract distinguishable features from overlong sentences, thereby reducing the adverse effect of massive noises in sentences.

Similar findings can also be found on the 5-way 5-shot, 10-way 1-shot and 10-way 5-shot meta tasks. The ever-decreasing trend in the 5-way 1-shot also appears in these meta tasks. Comparing 1-shot and 5-shot meta tasks, all the methods achieve improvements for four sentence groups due to the increase of training data. Besides, compared to baselines, our proposal achieves the biggest improvements for four sentence groups in Fig.3c than on the other three meta tasks. This phenomenon shows that our proposal is robust and competitive on the most challenging 10-way 1-shot meta task.

4.6 Ablation Studies

In order to better understand the contributions of different components, the ablation studies are conducted on four types of meta tasks. In the ablation studies, we remove or replace some specific parts to measure their influence on DKR, which is marked with the notation"-". The ablation results are demonstrated in Table 3.

Clearly, the removal or replacement of components all leads to the model degeneration, proving the efficacy of each component. In particular, the biggest drop happens in "-logic&type", which reflects that the logic-aware inference module plays a key role in few-shot RE and simple concatenation in previous works probably cannot improve the model performance. These findings can be further verified in the comparison of "-logic&text" and "-logic&type", where the model incorporating the ontology-level KG ("-logic&type") loses the competitions against the model only using text ("-logic&text"). Besides, in the discrimination finding module, losing the category-level contrastive learning is more sever than DRK without the instance level one. The comparisons demonstrate that compared to the instance-level contrastive learning, the category-level one can better identify inter-class differences to help predict relations.

4.7 Error analyses

To further figure out the error causes, we conduct error analyses of our proposal and the best baseline ConceptFERE on the test set. In particular, we divide the causes of classification errors into there main categories: word-overlap (word), entity-type (entity) and others. For each test relation, we randomly select 5 instances (i.e., K=5) and 50

Relation	Correct	Word	Entity	Other	Top 2 confounders
constellation	0.98	0.02	0.00	0.00	follows(0.02)
	0.96	0.03	0.00	0.01	part of(0.02),located in body of water(0.02)
part of	0.50	0.35	0.08	0.07	follows(0.10)
	0.40	0.39	0.11	0.10	member(0.18), subject(0.14)
mother	0.92	0.00	0.08	0.00	child(0.06), spouse(0.02)
	0.52	0.00	0.48	0.00	child(0.42), spouse(0.06)
spouse	0.48	0.00	0.52	0.00	child(0.44), mother(0.08)
	0.36	0.02	0.72	0.00	child(0.34), mother(0.28)

Table 4: Rates of the correct predictions (Correct), the word-overlap error (Word), the entity-type error (Entity) and the other errors (Other) in DRK (shown in orange) and ConceptFERE (shown in blue). Top 2 confounders list the top two wrongly-predicted relations and their rates.

instances of this relation into the support set and the query set, respectively. And in the evaluating process, the support set is used to calculate the relation prototype, then predict the relation label in the query set based on the similarity to the relation prototype. Also, we repeat the process three times and reported the mean result. Note that the prediction rate in the query set is presented relation-by-relation. Based on this setting, the error analyses of some relations are listed in table 4, and the other test relations can refer to Appendix A.2.

As shown in table 4, the rate of correct prediction varies widely among different relations, especially for some confusion relations ("part of" and "spouse"), their performance are much lower than expected. This unexpected performance is stemmed from the "word" and "entity" error types. Besides, the top 2 confounders have similar expressions or the same entity-type pairs as the instances of the ground-truth label. These findings all verify our arguments that owing to the limited inference based on the shallow features, RE models easily fall into the prediction confusions.

Fortunately, DRK relieves the prediction confusion by reducing the rates of these two errors. Taking "spouse" as an example, DRK achieves obvious declines in the "word" and "entity" errors against ConceptFERE (reducing 0.02 and 0.20 rates, respectively). Furthermore, DRK lowers the number of confounders relations. For example, taking "constellation" for example, ConceptFERE has two confounders ("part of" and "located in body of water") while DRK only has one confounder ("follows"). The above phenomena demonstrate that our proposal can further boost RE performance for each relation by reducing these errors, rather than improving easy relations for good average performance as the baselines do.

5 Conclusion

This paper focuses on the prediction confusions in few-shot RE. To relieve these confusions, this paper develops a *discriminative rule-based knowledge* (DRK) method consisting of a logic-aware inference module and a discrimination finding module. Specifically, the first module relieves the word-overlap confusion through the rule-based knowledge incorporation and the other module alleviates the entity-type confusion by a hierarchical contrastive learning. Extensive experiments show the effectiveness of our proposal. As for future work, we plan to explore the confusion by the parameter-efficient prompt tuning (Lester et al., 2021; Liao et al., 2022).

References

Sam Brody, Sichao Wu, and Adrian Benton. 2021. Towards realistic few-shot relation extraction. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*.

Mingyang Chen, Wen Zhang, Wei Zhang, Qiang Chen, and Huajun Chen. 2019. Meta relational learning for few-shot link prediction in knowledge graphs. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing*.

Yew Ken Chia, Lidong Bing, Soujanya Poria, and Luo Si. Relationprompt: Leveraging prompts to generate synthetic data for zero-shot relation triplet extraction. In *Findings of the Association for Computational Linguistics*, pages 45–57.

Chelsea Finn, Pieter Abbeel, and Sergey Levine. 2017. Model-agnostic meta-learning for fast adaptation of deep networks. In *Proceedings of the 34th International Conference on Machine Learning*, volume 70.

Tianyu Gao, Xu Han, Zhiyuan Liu, and Maosong Sun. 2019a. Hybrid attention-based prototypical networks for noisy few-shot relation classification. In *The Thirty-Third AAAI Conference on Artificial Intelligence*.

Tianyu Gao, Xu Han, Hao Zhu, Zhiyuan Liu, Peng Li, Maosong Sun, and Jie Zhou. 2019b. Fewrel 2.0: Towards more challenging few-shot relation classification. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing*.

Xu Han, Weilin Zhao, Ning Ding, Zhiyuan Liu, and Maosong Sun. 2021a. PTR: prompt tuning with rules for text classification. *CoRR*, abs/2105.11259.

Xu Han, Hao Zhu, Pengfei Yu, Ziyun Wang, Yuan Yao, Zhiyuan Liu, and Maosong Sun. 2018. Fewrel: A large-scale supervised few-shot relation classification dataset with state-of-the-art evaluation. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*.

Yi Han, Linbo Qiao, Jianming Zheng, Zhigang Kan, Linhui Feng, Yifu Gao, Yu Tang, Qi Zhai, Dongsheng Li, and Xiangke Liao. 2021b. Multi-view interaction learning for few-shot relation classification. In *The 30th ACM International Conference on Information and Knowledge Management*.

Junheng Hao, Muhao Chen, Wenchao Yu, Yizhou Sun, and Wei Wang. 2019. Universal representation learning of knowledge bases by jointly embedding instances and ontological concepts. In *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, pages 1709–1719.

Keqing He, Jinchao Zhang, Yuanmeng Yan, Weiran Xu, Cheng Niu, and Jie Zhou. 2020. Contrastive zero-shot learning for cross-domain slot filling with adversarial attack. In *Proceedings of the 28th International Conference on Computational Linguistics*.

Weizhi Huang, Ming He, and Yongle Wang. 2021. A survey on meta-learning based few-shot classification. In *Machine Learning and Intelligent Communications - 6th EAI International Conference*, volume 438, pages 243–253.

Brian Lester, Rami Al-Rfou, and Noah Constant. 2021. The power of scale for parameter-efficient prompt tuning. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*.

Luoqiu Li, Xiang Chen, Hongbin Ye, Zhen Bi, Shumin Deng, Ningyu Zhang, and Huajun Chen. 2021. On robustness and bias analysis of bert-based relation extraction. In *Knowledge Graph and Semantic Computing: Knowledge Graph Empowers New Infrastructure Construction - 6th China Conference*.

Jinzhi Liao, Xiang Zhao, Jianming Zheng, Xinyi Li, Fei Cai, and Jiuyang Tang. 2022. PTAU: prompt tuning for attributing unanswerable questions. In *The 45th International ACM SIGIR Conference on Research and Development in Information Retrieval*.

Weijie Liu, Peng Zhou, Zhe Zhao, Zhiruo Wang, Qi Ju, Haotang Deng, and Ping Wang. 2020. K-BERT: enabling language representation with knowledge graph. In *The Thirty-Fourth AAAI Conference on Artificial Intelligence*.

Nikhil Mishra, Mostafa Rohaninejad, Xi Chen, and Pieter Abbeel. A simple neural attentive meta-learner. In *6th International Conference on Learning Representations*.

Nikhil Mishra, Mostafa Rohaninejad, Xi Chen, and Pieter Abbeel. 2018. A simple neural attentive meta-learner. In *6th International Conference on Learning Representations*.

Alex Nichol, Joshua Achiam, and John Schulman. 2018. On first-order meta-learning algorithms. *CoRR*, abs/1803.02999.

Meng Qu, Tianyu Gao, Louis-Pascal A. C. Xhonneux, and Jian Tang. 2020. Few-shot relation extraction via bayesian meta-learning on relation graphs. In *Proceedings of the 37th International Conference on Machine Learning*, volume 119, pages 7867–7876.

Arpita Roy and Shimei Pan. 2021. Incorporating medical knowledge in BERT for clinical relation extraction. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*.

Oscar Sainz, Oier Lopez de Lacalle, Gorka Labaka, Ander Barrena, and Eneko Agirre. 2021. Label verbalization and entailment for effective zero and few-shot relation extraction. In *Proceedings of the 2021*

Conference on Empirical Methods in Natural Language Processing.

Victor Garcia Satorras and Joan Bruna Estrach. Few-shot learning with graph neural networks. In *6th International Conference on Learning Representations*.

Jake Snell, Kevin Swersky, and Richard S. Zemel. 2017. Prototypical networks for few-shot learning. In *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems*, pages 4077–4087.

Livio Baldini Soares, Nicholas FitzGerald, Jeffrey Ling, and Tom Kwiatkowski. 2019. Matching the blanks: Distributional similarity for relation learning. In *Proceedings of the 57th Conference of the Association for Computational Linguistics*.

Shengli Sun, Qingfeng Sun, Kevin Zhou, and Tengchao Lv. 2019. Hierarchical attention prototypical networks for few-shot text classification. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing*.

Prasetya Ajie Utama, Nafise Sadat Moosavi, Victor Sanh, and Iryna Gurevych. 2021. Avoiding inference heuristics in few-shot prompt-based finetuning. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*.

Hailin Wang, Guoming Lu, Jin Yin, and Ke Qin. 2021. Relation extraction: A brief survey on deep neural network based methods. In *The 4th International Conference on Software Engineering and Information Management*.

Yingyao Wang, Junwei Bao, Guangyi Liu, Youzheng Wu, Xiaodong He, Bowen Zhou, and Tiejun Zhao. 2020. Learning to decouple relations: Few-shot relation classification with entity-guided attention and confusion-aware training. In *Proceedings of the 28th International Conference on Computational Linguistics*.

Bishan Yang, Wen-tau Yih, Xiaodong He, Jianfeng Gao, and Li Deng. 2015. Embedding entities and relations for learning and inference in knowledge bases. In *3rd International Conference on Learning Representations*.

Kaijia Yang, Nantao Zheng, Xinyu Dai, Liang He, Shujian Huang, and Jiajun Chen. 2020. Enhance prototypical network with text descriptions for few-shot relation classification. In *The 29th ACM International Conference on Information and Knowledge Management*.

Shan Yang, Yongfei Zhang, Guanglin Niu, Qinghua Zhao, and Shiliang Pu. 2021. Entity concept-enhanced few-shot relation extraction. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing*, pages 987–991.

Jiawen Zhang, Jiaqi Zhu, Yi Yang, Wandong Shi, Congcong Zhang, and Hongan Wang. 2021. Knowledge-enhanced domain adaptation in few-shot relation classification. In *The 27th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*.

Jianming Zheng, Fei Cai, and Honghui Chen. 2020a. Incorporating scenario knowledge into A unified fine-tuning architecture for event representation. In *Proceedings of the 43rd International ACM SIGIR conference on research and development in Information Retrieval*.

Jianming Zheng, Fei Cai, Wanyu Chen, Wengqiang Lei, and Honghui Chen. 2021. Taxonomy-aware learning for few-shot event detection. In *The Web Conference 2021, Virtual Event / Ljubljana, Slovenia, April 19-23*, pages 3546–3557.

Jianming Zheng, Fei Cai, Yanxiang Ling, and Honghui Chen. 2020b. Heterogeneous graph neural networks to predict what happen next. In *Proceedings of the 28th International Conference on Computational Linguistics*.

A Appendix

A.1 Dataset

We demonstrate some unstructured text knowledge, including the relation id,, name and description in Table 5.

A.2 Errors in test relations

Following the setting in section §4.7, the detail prediction performance of DRK and ConceptFERE on each test relation is shown in Table 6.

In general, our proposal outperforms the baseline for almost all test relations, and the observations from section §4.7 can also be found in Table 6. Besides, compared to the ConceptFERE that only precisely identifies one relation (P2094), our proposal can accurately identify four relations (P177, 2094, P412 and P413), which demonstrates our proposal has a competitive inference ability. However, our proposal underperforms the baseline for the relation P206. This unusual relation may be stemmed from an improper representation of entity in the linking process, thereby invalidating our proposed modules.

Id	Ralation name	Relation description
P155	follows	immediately prior item in a series of which the subject is a part.
P177	crosses	obstacle (body of water, road, ...) which this bridge crosses over or this tunnel goes under.
P206	located in or next to body of water	located in or next to body of water", "sea, lake or river".
P2094	competition class	official classification by a regulating body under which the subject (events, teams, participants, or equipment) qualifies for inclusion.
P25	mother	female parent of the subject. For stepmother, use stepparent.
P26	spouse	the subject has the object as their spouse (husband, wife, partner, etc.)
P361	part of	object of which the subject is a part (it's not useful to link objects which are themselves parts of other objects already listed as parts of the subject).
P364	original language of film or TV show	language in which a film or a performance work was originally created.
P40	child	subject has object as biological, foster, and/or adoptive child.
P410	military rank	military rank achieved by a person.
P412	voice type	person's voice type. expected values: soprano, mezzo-soprano, contralto, countertenor, tenor, baritone, bass (and derivatives).
P413	position played on team	position or specialism of a player on a team, e.g. Small Forward.
P463	member of	organization or club to which the subject belongs. Do not use for membership in ethnic or social groups, nor for holding a position such as a member of parliament.
P59	constellation	the area of the celestial sphere of which the subject is a part (from a scientific standpoint, not an astrological one).
P641	sport	sport in which the subject participates or belongs to.
P921	main subject	primary topic of a work.

Table 5: Relation descriptions for the test set with the relation id, name and description content.

Id	Correct		Top 2 confounders	
P155	0.88	0.86	P361(0.04), P463(0.04)	P361(0.04), P463(0.04)
P177	1.00	0.96	None	P361(0.02), P206(0.02)
P206	0.42	0.62	P177(0.38),P361(0.20)	P177(0.38)
P2094	1.00	1.00	None	None
P25	0.92	0.52	P40(0.06), P26(0.02)	P40(0.42), P26(0.06)
P26	0.48	0.36	P40(0.44), P25(0.08)	P25(0.34), P26(0.28)
P361	0.50	0.40	P463(0.20), P155(0.10)	P463(0.18), P921(0.14)
P364	1.00	0.98	None	P921(0.02)
P40	0.82	0.66	P26(0.06), P25(0.06)	P25(0.28), P26(0.06)
P410	0.98	0.98	P413(0.02)	P413(0.02)
P412	1.00	0.96	None	P413(0.04)
P413	1.00	0.82	None	P412(0.14), P2094(0.04)
P463	0.82	0.76	P361(0.10), P412(0.06)	P361(0.08), P26(0.04)
P59	0.98	0.96	P155(0.02)	P206(0.02), P361(0.02)
P641	0.48	0.82	P413(0.30), P2094(0.10)	P412(0.16), P364(0.02)
P921	0.52	0.40	P361(0.24), P641(0.06)	P463(0.18), P59(0.14)

Table 6: Rates of the correct prediction and top 2 confounders in our proposal DRK (shown in orange) and ConceptFERE (shown in blue)

DocQueryNet: Value Retrieval with Arbitrary Queries for Form-like Documents

Mingfei Gao*, Le Xue*, Chetan Ramaiah, Chen Xing, Ran Xu, Caiming Xiong
Salesforce Research, Palo Alto, USA
{mingfei.gao, lxue, cramaiah, cxing, ran.xu, cxiong}@salesforce.com

Abstract

We propose, DocQueryNet, a value retrieval method with arbitrary queries for form-like documents to reduce human effort of processing forms. Unlike previous methods that only address a fixed set of field items, our method predicts target value for an arbitrary query based on the understanding of the layout and semantics of a form. To further boost model performance, we propose a simple document language modeling (SimpleDLM) strategy to improve document understanding on large-scale model pre-training. Experimental results show that DocQueryNet outperforms previous designs significantly and the SimpleDLM further improves our performance on value retrieval by around 17% F1 score compared with the state-of-the-art pre-training method. *Code is available here*.

1 Introduction

Form-like documents are very commonly used in business workflows. However, tremendous forms are still processed manually everyday. When humans need to extract some relevant information from a form-like document, they proceed as conducting a value retrieval task with some text queries. For example in Figure 1, when humans process a form, they usually have a description of the information (query) that they want to extract (e.g., total page number). Then, they examine the form (usually an image or a PDF) carefully to locate the key (e.g., *NUMBER OF PAGES INCLUDING COVER SHEET* in Figure 1) that is most semantically similar to the query and finally infer the target value based on the localized key. This manual process costs a large amount of human efforts as the number of forms and queries increases. Automating information extraction from forms is important to alleviate this problem.

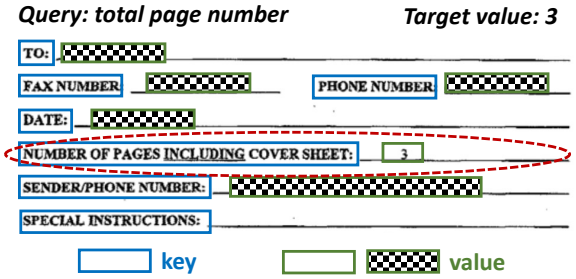

Figure 1: Illustration of the value retrieval with an arbitrary query. A user obtains the target value from a document based on a query of interest. Some values are decorated with a mosaic for privacy purposes.

Existing methods formulate the problem as sequence labeling (Xu et al., 2020b) or field extraction (Gao et al., 2022), where they define a fixed set of items of interest (referred as fields) and train models that only extract values of the pre-defined fields. There are at least two limitations of this formulation. First, forms are very diverse and it is impossible to cover all the items of interest using a fixed set of fields. Second, their models are very domain-specific and hard to be utilized for different form types. For example, an invoice field extractor may not be able to process resumes, since different fields are expected for these two form types.

To handle diverse queries with a unified model, we formulate the problem as value retrieval with arbitrary queries for form-like documents. Under such task formulation, users can extract values from a form by presenting variants of the corresponding keys as queries. We also set up a benchmark for the task by introducing a simple yet effective method, i.e., DocQueryNet. DocQueryNet takes an arbitrary query phrase and all the detected optical character recognition (OCR) words with their locations in the form as inputs. Then, we model the interactions between the query and the detected words from the document using a transformer-based architecture. The training objective encourages the

*Mingfei and Le contributed equally.

Proceedings of the 29th International Conference on Computational Linguistics, pages 2141–2146
October 12–17, 2022.

matching of the positive query-value pairs and discourages that of the negative ones. To further boost the performance, we present SimpleDLM, a simple document pre-training strategy that makes it more flexible to learn local geometric relations between words/tokens compared to existing pre-trained models. Experimental results show that DocQueryNet outperforms the baselines by a large margin under different settings. When initializing using SimpleDLM, our method is further improved largely by about 17% F1 score compared to initializing using a state-of-the-art (SOTA) pre-trained model, i.e., LayoutLM (Xu et al., 2020b).

2 Related Work

2.1 Information Extraction from Documents

Information extraction from documents is crucial for improving the efficiency of form processing and reducing human labor. Information extraction is often formulated as a field extraction task. Palm et al. 2019 propose an invoice field extractor by using an Attend, Copy, Parse architecture. Majumder et al. 2020 present a field-value pairing framework that learns the representations of fields and value candidates in the same feature space using metric learning. Nguyen et al. 2021 propose a span extraction approach to extract the start and end of a value for each field. Gao et al. 2022 introduce a field extraction system that can be trained with large-scale unlabeled documents. Xue et al. 2021 propose form transformations to mimic the variations of forms for robustness evaluation. Unlike previous methods that aim to extracting values for a pre-defined set of fields, our method targets at retrieving values for arbitrary queries.

2.2 Document Pre-training

Document pre-training is an effective strategy to improve document-related downstream tasks. Xu et al. 2020b propose LayoutLM that models interaction between texts and layouts in scanned documents using masked language modeling and image-text matching. Later, LayoutLMv2 (Xu et al., 2020a) introduces a spatial-aware self-attention mechanism to improve learning relative positional relationship among different text blocks. Most recently, Appalaraju et al. 2021 propose DocFormer that encourages the interaction between image and text modalities by adding an image reconstruction task. The existing pre-training methods perform well when applied to downstream tasks such as document classification and token sequence labeling. However, all of the above methods include the absolute 1-D positional embedding (the so called reading order) of the tokens in the inputs. Although this 1-D embedding is a helpful prior knowledge for a holistic understanding of a document, it hinders a model from learning rich geometric relationship among tokens, thus is not beneficial to our value retrieval task, where local geometric relationship between words is essential for prediction. Our method uses permutation-invariant positional encoding to improve model performance.

3 Our Approach

Problem Formulation. The inputs to the system are an expected key phrase as the query, $\mathbf{Q}_i$, and a document $\mathbf{D}_i$. $\mathbf{D}_i$ is represented using a set of OCR words $\{w_{i1}, w_{i2}, ..., w_{iN}\}$ and their bounding-box locations $\{b_{i1}, b_{i2}, ..., b_{iN}\}$ in the document. The input query phrase is tokenized to $\mathbf{Q}_i = \{q_{i1}, q_{i2}, ..., q_{iM}\}$. A value retrieval system reads the document and understands the layout and semantics. The goal is to pick a phrase from the OCR words as the value $\mathbf{V}_i$ for the input query, $\mathbf{Q}_i$. Since the modeling is within one document, we will omit the subscript i for simplicity.

3.1 Value Retrieval with Arbitrary Queries

To the best of our knowledge, there are no existing methods explicitly address value retrieval with arbitrary queries from scanned forms. However, it is possible to apply the recent pairing-based field extraction methods (Majumder et al., 2020; Nguyen et al., 2021) to this task with some modification. In previous design, they first embed fields using a text encoder, extract OCR word representation by modeling the interaction of OCR words using self-attention and then conduct the field-value pairing. To accommodate arbitrary queries, we can simply replace their field (in a fixed set) embedding with the embedding of an arbitrary query as shown in Figure 2 (previous method). There are two obvious drawbacks of these methods: (1) they model the interaction between query and OCR words in a shallow way using only simple fully connected layers and (2) they require an additional model of text encoder for query embedding which introduces extra computational cost. We set this previous design as our baseline. The implementation details are shown in the appendix.

In contrast, our DocQueryNet utilizes a unified

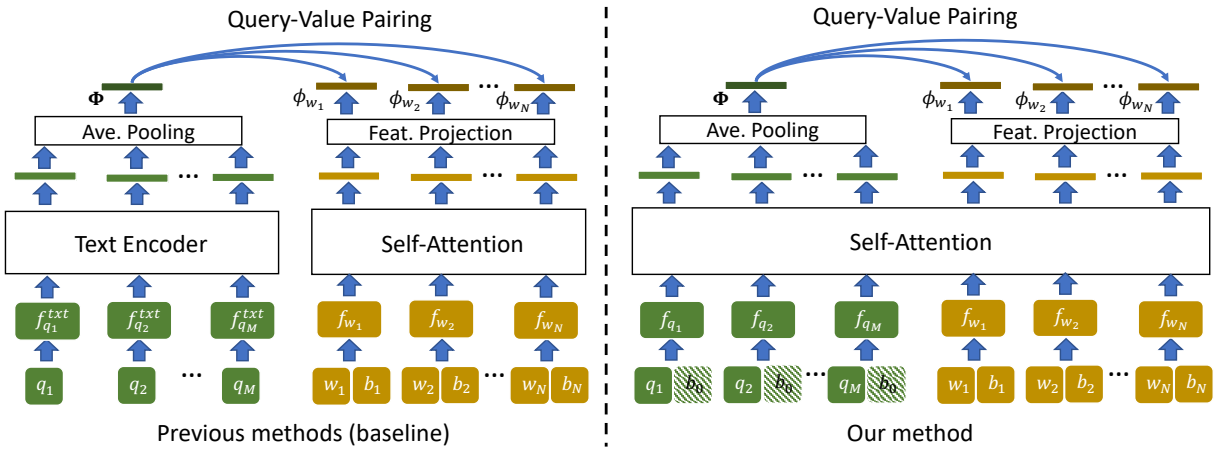

Figure 2: Comparison of previous methods and our method (see details in Section 3).

model to deeply model interactions among the query words and the OCR words. The direct inputs are a query $\{q_1, q_2, ..., q_M\}$ and OCR words $\{w_1, w_2, ..., w_N\}$ associated with their locations $\{b_1, b_2, ..., b_N\}$. We use a fixed dummy location b_0 for each query. Each query/OCR word is embedded as $f^{txt}_{w_j} \in \mathbb{R}^d$ and its location is encoded as $f^{loc}_{b_j} \in \mathbb{R}^d$, where d indicates the length of each vector (see Section 3.2 for details). The final embedding of each word (e.g., $\mathbf{f}_{q_j}$ for a query word or $\mathbf{f}_{w_j}$ for an OCR word) is the summation of its word embedding and location embedding.

A transformer is used to model the interactions among $\{\mathbf{f}_{q_1}, \mathbf{f}_{q_2}.., \mathbf{f}_{q_M}\}$ and $\{\mathbf{f}_{w_1}, \mathbf{f}_{w_2}, ..., \mathbf{f}_{w_N}\}$ via L self-attention layers. In the lth layer, the hidden representation of the jth token is updated following

$$\mathbf{h}^l_j = \text{Softmax}(\frac{\mathbf{h}^{l-1}_j \mathbf{H}^{l-1^T}}{\sqrt{d_h}}) \cdot \mathbf{H}^{l-1}, \quad (1)$$

where $\mathbf{H}^{l-1}$ indicates the representation of all tokens from the $(l-1)^{th}$ layer and d_h denotes the length of the hidden feature $\mathbf{h}^l_j$. $\mathbf{H}^0 = \{\mathbf{f}_{q_1}, \mathbf{f}_{q_2}, ..., \mathbf{f}_{q_M}, \mathbf{f}_{w_1}, \mathbf{f}_{w_2}..., \mathbf{f}_{w_N}\}$. In the final layer, $\mathbf{H}^L = \{\hat{\mathbf{f}}_{q_1}, \hat{\mathbf{f}}_{q_2}, ..., \hat{\mathbf{f}}_{q_M}, \hat{\mathbf{f}}_{w_1}, \hat{\mathbf{f}}_{w_2}..., \hat{\mathbf{f}}_{w_N}\}$. The self-attention mechanism deeply models the interactions among query words and OCR words.

We obtain the query phrase representation, Φ, using average pooling over $\{\hat{\mathbf{f}}_{q_1}, \hat{\mathbf{f}}_{q_2}, ..., \hat{\mathbf{f}}_{q_M}\}$. The final representation of w_j is obtained by $\phi_{w_j} = FC(\hat{\mathbf{f}}_{w_j})$, where FC indicates a fully connected layer. The likelihood score of w_j being a part of the target value of the query is obtained in Equation 2

$$\mathbf{s}_j = \text{Sigmoid}(\phi_{w_j} \cdot \Phi^T). \quad (2)$$

Our model is expected to learn (1) the layout and semantics of the document (2) the mapping between the input query phrase and the actual key texts in the document and (3) the geometric and semantic relationship between the key and value.

Model optimization. During training, each w_j is associated with a ground-truth label $y_j \in \{0, 1\}$, where 1 means this word is a part of the target value and 0 means it is not. The model is optimized using binary cross entropy loss as $\frac{1}{N}\sum_{j=1}^{N} y_j \text{logs}_j + (1 - y_j)(1 - \text{logs}_j)$.

Inference. Since the target value may contain multiple words, we group nearby OCR words horizontally as value candidates, $\{g_1, g_2, ..., g_D\}$, based on their locations using DBSCAN algorithm (Ester et al., 1996), where D is the number of grouped candidates. The value score of each candidate, g_r, is the maximum of all its covered words as $\mathbf{S}_r = \max_{w_j \in g_r} \mathbf{s}_j$, where, $w_j \in g_r$ indicates the OCR word w_j is a part of the grouped g_r. The value candidate with the highest score is used as the value prediction.

3.2 SimpleDLM for Document Pre-training

We introduce a Simple Document Language Modeling (SimpleDLM) method to encourage the understanding of the geometric relationship among words during pre-training.

The inputs to our pre-trained model are the OCR words $\{w_1, w_2, ..., w_N\}$ associated with their locations $\{b_1, b_2, ..., b_N\}$. The word/location embedding protocol and the transformer structure of our pre-trained model are the same as those of our value retrieval model such that the pre-trained model can be directly used for initializing the parameters of the value retrieval model. Specifically, the word embedding, $f^{txt}_{w_j}$, is constructed by using a simple

look up table. Previous works (Xu et al., 2020b,a; Appalaraju et al., 2021) require the input text sorted in the reading order, so that they can process texts of the document in a similar way of processing languages. They leverage this prior knowledge by adding the ranking of each word as the 1-D positional embedding in the final location embedding, $f_{b_j}^{loc}$. This 1-D embedding provides a holistic view of the geometric relationship of words. However, it introduces extra dependence on the OCR engines (most SOTA OCR engines do not have the capability of sorting detected OCR words in the reading order), and it also restricts the model from learning local geometric relations between words in a flexible way. To encourage a model to better learn the local geometric relations, we exclude the 1-D positional embedding and only encode the 2-D bounding-box location (top-left, bottom-right, width and height) of each word using a lookup table. For simplicity, we only use the masked language modeling as the pre-training objective. We show in Section 4 that our method improves the SOTA largely by using this simple pre-training strategy.

4 Experiments

The following datasets are used in our experiments.

IIT-CDIP (Lewis et al., 2006) is a large-scale unlabeled document dataset that contains more than 11 million scanned images. Following prior works (Xu et al., 2020b,a; Appalaraju et al., 2021), our model is pre-trained using this dataset.

FUNSD (Jaume et al., 2019) is a commonly used dataset for spatial layout analysis. It contains 199 scanned forms with 9,707 semantic entities annotated, where 149 samples are for training and 50 for testing. The semantic linking annotations for all the key-value pairs are provided in the dataset.

INV-CDIP (Gao et al., 2022) is document dataset which contains 350 real invoices for testing. This dataset has key-value pair annotations for 7 commonly used invoice fields including *invoice_number*, *purchase_order*, *invoice_date*, *due_date*, *amount_due*, *total_amount* and *total_tax*. We evaluate our model using this test set.

Settings. By default the annotated key texts of each dataset is used as the queries. The location of the keys are not used. Models are pre-trained on IIT-CDIP and fine-tuned on the train set of FUNSD. More implementation details are in the appendix.

Evaluation Metric. We use F1 score to evaluate models. Exact string matching between our predicted values and the ground-truth ones is used to count true positive, false positive and false negative. If a query has multiple value answers, a prediction is counted as correct if it equals one of them.

Model	Pretrain	Precision	Recall	F1
Baseline	Bert	31.7	32.0	31.9
	LayoutLM	41.8	42.1	41.9
	SimpleDLM	56.0	56.5	56.3
Ours	Bert	35.1	35.4	35.3
	LayoutLM	43.6	43.9	43.8
	SimpleDLM	**60.4**	**60.9**	**60.7**

Table 1: Comparisons on FUNSD when our model and baseline use different pre-trained models. **Ours** indicates **DocQueryNet**.

Model	Query	Precision	Recall	F1
Baseline	Exact Key	33.5	31.5	32.5
Ours		**50.5**	**47.6**	**49.0**
Baseline	Field Name	6.0	5.6	5.7
Ours		**21.2**	**19.9**	**20.5**

Table 2: Comparison in the transfer setting. Models are trained on FUNSD and evaluated on INV-CDIP. SimpleDLM is used for both methods. **Ours** indicates **DocQueryNet**.

Experimental Results. The comparisons between our DocQueryNet and the baseline when pre-trained using different approaches are shown in Table 1. The performance of both methods are improved largely with SimpleDLM. For the baseline, the F1 score is improved by 14.4% when replacing the LayoutLM with our SimpleDLM as the pre-trained model. Similarly, using SimpleDLM increases the F1 score of our method by 16.9% compared to using LayoutLM. Our method outperforms the baseline by 3-4% using different pre-trained models.

Transferring ability to another dataset is important in real-world applications. We measure this ability by directly evaluating the trained models using FUNSD to the test set of INV-CDIP in Table 2. As shown, when transferring to a new dataset, the performance of both our method and the baseline drops, compared to the numbers in Table 1. When using the exact key as the query, our method largely surpasses the baseline by 16.5% in F1 score.

In practice, we may not assume the input queries match exactly with the actual keys shown in a form. Here, we experiment using the field names directly as the queries. Using field names is a more conve-

nient way, since users don't need to design different queries that match keys for different forms. However, field names are more abstract, which makes the problem more challenging. When using the abstract field name as a query, our method achieves 20.5% F1 which is 14.8% better than our baseline.

Discussion. We use a dummy box to serve as the location of a query word, since there are no actual boxes for the query words in practice. This choice may make our model lose the sequential ordering information of a query phrase. To investigate this effect, we conduct an experiment and leverage the ordering information of query words by adding different positional ids to different words of a query in our model. The results show that this change decreases our performance from 60.7% to 60.3% in F1 score on FUNSD and from 49.0% to 47.6% in F1 score on INV-CDIP. It may be because that queries with the same meaning might be represented with words in different orders. For example, *total amount* could be referred to as *amount total*. Ignoring the ordering could help improve our model's generalization ability.

5 Conclusion

We introduce DocQueryNet, a framework for value retrieval with arbitrary queries for form-like documents. It takes a query and the detected OCR words from a document as inputs, models their interactions and predicts the best value corresponding to the input query. We also present SimpleDLM as a pre-training strategy to boost performance. Experimental results show that our method significantly outperforms previous designs in different settings.

6 Broader Impacts

This work is introduced to automate the information extraction from forms to improve document processing efficiency. It has positive impacts such as reducing human labor. However, reducing human labor may also cause negative consequences such as job loss or displacement, particularly amongst low-skilled labor who may be most in need of gainful employment. The negative impact is not specific to this work, but should be addressed broadly in the field of AI research.

References

Srikar Appalaraju, Bhavan Jasani, Bhargava Urala Kota, Yusheng Xie, and R Manmatha. 2021. Docformer: End-to-end transformer for document understanding. *arXiv preprint arXiv:2106.11539*.

Martin Ester, Hans-Peter Kriegel, Jörg Sander, Xiaowei Xu, et al. 1996. A density-based algorithm for discovering clusters in large spatial databases with noise. In *KDD*.

Mingfei Gao, Zeyuan Chen, Nikhil Naik, Kazuma Hashimoto, Caiming Xiong, and Ran Xu. 2022. Field extraction from forms with unlabeled data. *ACL Workshop*.

Guillaume Jaume, Hazim Kemal Ekenel, and Jean-Philippe Thiran. 2019. Funsd: A dataset for form understanding in noisy scanned documents. In *2019 International Conference on Document Analysis and Recognition Workshops (ICDARW)*, volume 2, pages 1–6. IEEE.

David Lewis, Gady Agam, Shlomo Argamon, Ophir Frieder, David Grossman, and Jefferson Heard. 2006. Building a test collection for complex document information processing. In *Proceedings of the 29th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 665–666.

Bodhisattwa Prasad Majumder, Navneet Potti, Sandeep Tata, James Bradley Wendt, Qi Zhao, and Marc Najork. 2020. Representation learning for information extraction from form-like documents. In *ACL*.

Tuan-Anh D Nguyen, Hieu M Vu, Nguyen Hong Son, and Minh-Tien Nguyen. 2021. A span extraction approach for information extraction on visually-rich documents. *arXiv preprint arXiv:2106.00978*.

Rasmus Berg Palm, Florian Laws, and Ole Winther. 2019. Attend, copy, parse end-to-end information extraction from documents. In *ICDAR*.

Yang Xu, Yiheng Xu, Tengchao Lv, Lei Cui, Furu Wei, Guoxin Wang, Yijuan Lu, Dinei Florencio, Cha Zhang, Wanxiang Che, et al. 2020a. Layoutlmv2: Multi-modal pre-training for visually-rich document understanding. *arXiv preprint arXiv:2012.14740*.

Yiheng Xu, Minghao Li, Lei Cui, Shaohan Huang, Furu Wei, and Ming Zhou. 2020b. Layoutlm: Pre-training of text and layout for document image understanding. In *Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, pages 1192–1200.

Le Xue, Mingfei Gao, Zeyuan Chen, Caiming Xiong, and Ran Xu. 2021. Robustness evaluation of transformer-based form field extractors via form attacks. *arXiv preprint arXiv:2110.04413*.

A Appendix

A.1 Implementation Details

Our code is implemented using Pytorch. We used Tesseract[1] to extract OCR words from documents for IIT-CDIP and INV-CDIP. Since FUNSD provides an official OCR annotation, we use it directly. The total number of query words M and the OCR words are different for different queries and documents. We keep $M + N = 512$ and pad with 0s when needed. We follow LayoutLM-base [2] to setup the structure of our transformers. The fully connected layer used for feature projection has 768 units. Each document is rescaled to [1000, 1000] and the dummy location, b_0, is set to [0, 0, 1000, 1000]. Adam is used as the optimizer. During pre-training, the learning rate is $5e^{-5}$. During fine-tuning, the learning rate is set to $3e^{-5}$ with weight decay equals to 0.9. SimpleDLM is initialized by LayoutLM and pre-trained on IIT-CDIP using 8 Nvidia A100 GPUs with a batch size of 36 for one epoch. We use a single A100 GPU for fine-tuning, where the training batch size is 8 and the total number of epochs is 45. In our experiments, all the pre-trained models including Bert, LayoutLM and SimpleDLM are base models. The pre-training and fine-tuning of our method take about 10 hours and 2 hours, respectively.

To perform a fair comparison, our method and baseline adopt the same experimental settings except for the interaction strategy. The baseline has the same transformer architecture and the feature projection layer as our method. The transformer takes OCR words with locations as inputs and obtain ϕ_{w_j} for each word w_j. And then, the query-value pairing score $\mathbf{s}_j$ is obtained by measuring the distance between the query representation, Φ, and ϕ_{w_j}. The query representation is obtained by average pooling over word embeddings extracted from a pre-trained Bert model.

[1] https://github.com/tesseract-ocr/tesseract (Apache License 2.0)

[2] https://github.com/microsoft/unilm/tree/master/layoutlm/deprecated (MIT License)

DoSEA: A Domain-specific Entity-aware Framework for Cross-Domain Named Entity Recogition

Minghao Tang[1,2], **Peng Zhang**[2,3]*, **Yongquan He**[4], **Yongxiu Xu**[1,2], **Chengpeng Chao**[1,2] and **Hongbo Xu**[1,2]
[1]Institute of Information Engineering, Chinese Academy of Sciences, Beijing, China
[2]School of Cyber Security, UCAS, Beijing, China
[3]School of Cyberspace Security, NJUST, Nanjing, China
[4]Meituan, Beijing, China
{tangminghao,hbxu}@iie.ac.cn, heyongquan@meituan.com

Abstract

Cross-domain named entity recognition aims to improve performance in a target domain with shared knowledge from a well-studied source domain. The previous sequence-labeling based method focuses on promoting model parameter sharing among domains. However, such a paradigm essentially ignores the domain-specific information and suffers from entity type conflicts. To address these issues, we propose a novel machine reading comprehension based framework, named DoSEA, which can identify domain-specific semantic differences and mitigate the subtype conflicts between domains. Concretely, we introduce an entity existence discrimination task and an entity-aware training setting, to recognize inconsistent entity annotations in the source domain and bring additional reference to better share information across domains. Experiments on six datasets prove the effectiveness of our DoSEA. Our source code can be obtained from https://github.com/mhtang1995/DoSEA.

1 Introduction

Named entity recognition(NER) is a fundamental task in natural language processing and has been extensively studied in various domains. However, acquiring a high performance NER model heavily relies on labor-intensive annotated data (Huang et al., 2015; Devlin et al., 2019). Thus, there is a growing interest in cross-domain NER, which aims to exploit the information on a well-studied source domain to improve the performance in a target domain (Pan and Yang, 2010). Following Daumé III (2007), we focus on the supervised cross-domain NER setting, which utilizes abundant annotated samples from the source domain and small annotated samples from the target domain.

Previous studies (Kim et al., 2015; Lin and Lu, 2018; Wang et al., 2018b; Jia and Zhang, 2020) typically treat cross-domain NER task as a sequence labeling problem, classifying each word as a type of entity. However, cross-domain NER is challenging due to the entity type difference between domains, since the target domain contains specific entity types. As Figure 1 shows, CrossNER Politics dataset (Liu et al., 2021) contains specific entity types(e.g., Event, Politician, Country and Political Party), which are not labeled in the CoNLL-2003 dataset (Sang and Meulder, 2003). Thus, the sequence-labeling based method commonly adopts separate model structures(CRF or softmax layer) for each domain, which primarily limits the parameter sharing across domains.

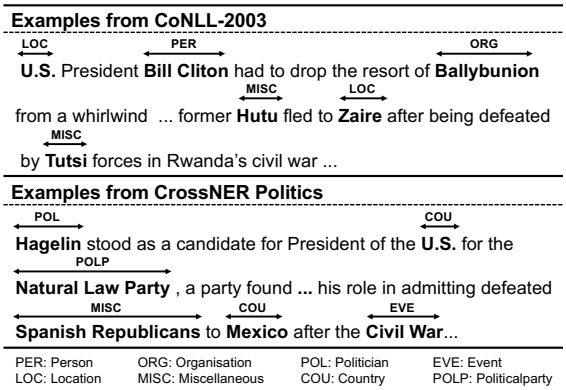

Figure 1: Examples from CoNLL-2003 and CrossNER Politics dataset.

A series of fine-tune methods (Lee et al., 2017; Lin and Lu, 2018) and multi-task learning methods (Wang et al., 2018b; Jia and Zhang, 2020) have been proposed for promoting parameter sharing. The fine-tune method first trains a model using source domain samples, then fine-tunes the model using target domain samples with an initialized label decoder. However, it depends on the sizes of target domain samples to learn a strong label decoder. Conversely, the multi-task learning method simultaneously trains a model for both domains under the jointly training strategy, and it essentially adds auxiliary tasks to facilitate parameter

*Peng Zhang is the corresponding author

Proceedings of the 29th International Conference on Computational Linguistics, pages 2147–2156
October 12–17, 2022.

sharing. Specifically, Wang et al. (2018b) adds KL-divergence in the features of identical entity types between each domain's CRF layer. Jia and Zhang (2020) models entity type features as separate cell states in a multi-cell compositional LSTM structure, then shares the same entity type's features across domains. However, they ignore the domain-specific information and suffer from entity type conflicts. For example, CoNLL-2003 and CrossNER Politics both contain "Location" entities, but the latter requires a distinction between "Location" and "Country", which is not considered in the former. In this case, previous work (Liu et al., 2021) indicates the subtype conflicts, since the cross-domain NER model may tend to classify "Country" as "Location" entities.

To this end, we propose a novel framework named Domain-specific Entity-aware(DoSEA) network for cross-domain NER, which aims to mitigate the negative impacts raised by the domain-specific entity types. Specifically, instead of assigning a separate model structure for each domain, our DoSEA formalizes NER as a machine reading comprehension (MRC) task (Li et al., 2020), which can naturally combine domain-related questions with annotated samples, to share all model parameters across domains. Moreover, we propose an entity existence discrimination (EED) task and an entity-aware training setting to recognize inconsistent entity annotations and handle the subtype conflicts. The EED task is designed to determine whether each type of entity exists in a sentence. As for the entity-aware training setting, we utilize the EED task to aware the presence of inconsistent annotated entities in source domain samples, then transform these entities to the target domain-specific entities by leveraging the explicit hierarchical relationship. The above procedures not only alleviate the subtype conflicts but also bring additional reference to better share information across domains. The main contributions of this paper are summarized as follows:

- We propose a novel framework named DoSEA for cross-domain NER, to handle the issues raised by the specific entity types from the target domain.
- In our method, we design an entity existence discrimination task and an entity-aware training setting to alleviate the subtype conflicts, which can identify the presence of each type of entity and transform the inconsistent annotated entities into hierarchical target domain-specific entities in source domain samples.
- Experimental results on six datasets show the superiority of our DoSEA over the state-of-the-art methods.

2 Related Work

MRC for NER Task. The goal of the machine reading comprehension(MRC) task is extracting answer spans from a sentence through a given question. There have been successful attempts to formalize other task as MRC task, such as NER (Li et al., 2020), relation extraction (Li et al., 2019) and event extraction (Liu et al., 2020a). As for the NER task, Li et al. (2020) first formulate NER as MRC to handle both flat and nested NER tasks. Xue et al. (2020) proposed a coarse-to-fine pre-training framework for NER task based on the MRC framework. Zhang et al. (2021) add MRC task in the training process of zero-resource NER task.

Cross-domain NER. Multi-task learning methods (Yang et al., 2017; Wang et al., 2018b; Jia et al., 2019; Jia and Zhang, 2020) have been popular in cross-domain NER, which is used to add the auxiliary task to improve performance. Jia et al. (2019) jointly trains the NER and LM tasks in a parameter generator network. Jia and Zhang (2020) proposed a multi-cell compositional LSTM structure for cross-domain NER, which models each entity type as a separate cell state. Fine-tune methods (Lee et al., 2017; Rodríguez et al., 2018; Cui et al., 2021) also show strong performance, which pre-trained a model in a source domain and then fine-tune the model in a target domain.

Some works try to achieve accurate transfer learning for cross-domain NER (Ruder and Plank, 2017; Wang et al., 2018a; Chen et al., 2021). Wang et al. (2018a) classifies source domain samples by similarity metrics and assigns different weights for training. Chen et al. (2021) proposes a data augmentation approach to transform the data representation across domains. A few works consider the relationship between entity types (Kim et al., 2015; Qu et al., 2016). Qu et al. (2016) considers the mentioned relationship between the source and target entity types, such as "Professor" and "Student". Compared with them, our proposed DoSEA is built on the MRC model, which aims to handle the issues raised by the specific entity types from the target domain.

3 Methodology

Figure 2 shows that DoSEA has three components, including a context encoder, a multi-task layer and an entity-aware training setting. The multi-task layer contains entity existence discrimination(EED) task and entity span prediction(ESP) task. In the entity-aware training setting, we adopt different training processes for source and target domains.

3.1 Problem formulation

Given a sentence $\mathbf{x} = (x_1, x_2, \cdots, x_n)$, where n denotes the word length of sentence $\mathbf{x}$. An entity $\mathbf{et}^t = (x^t_{start}, \cdots, x^t_{end})$ is a substring of $\mathbf{x}$ satisfying start$\leq$end, where t represents the entity type. Besides, we define $y_t \in \{0, 1\}$ as the ground-truth of whether the t type of entity exists in sentence $\mathbf{x}$.

Combining with Questions. Given sentence $\mathbf{x}^r$ from domain $r \in \{\mathcal{S}, \mathcal{T}\}$, we need to combine every entity query questions $\mathbf{Q}^r = \{\mathbf{q}^r_1, \mathbf{q}^r_2, ..., \mathbf{q}^r_m\}$ with sentence $\mathbf{x}^r$, where m denotes the number of entity types of domain r. The entity annotation guidelines are used as references to construct questions. In particular, we use questions from the target domain for common entity types between domains. Therefore, we obtain a set of quadruples $(\mathbf{q}^r_t, y^r_t, [\mathbf{et}^{t,r}_1, \cdots, \mathbf{et}^{t,r}_l], \mathbf{x}^r)$ in each domain $r \in \{\mathcal{S}, \mathcal{T}\}$, where l denotes the number of t type of entities in sentence $\mathbf{x}^r$.

Meanwhile, if source domain $\mathcal{S}$ contains the entity type which has a hierarchical relationship with the entity type specific to target domain $\mathcal{T}$, the questions of these entity subtypes are also combined with source domain sentence $\mathbf{x}^{\mathcal{S}}$ for entity-aware training. Thus, we obtain a set of quadruples $(\mathbf{q}^{\mathcal{T}}_{sub_t}, unknow, unknow, \mathbf{x}^{\mathcal{S}})$ in domain $\mathcal{S}$, where $\mathbf{q}^{\mathcal{T}}_{sub_t} \in \mathbf{Q}^{\mathcal{T}}$ denotes the question of entity subtype sub_t.

3.2 Context Encoder

Normally, we combine question $\mathbf{q}^r_t$ and sentence $\mathbf{x}^r$ as a string $\{\texttt{[CLS]}, \mathbf{q}^r_t, \texttt{[SEP]}, \mathbf{x}^r\}$, where `[CLS]` and `[SEP]` are special tokens. Then, the combined string is sent into the input embedding layer. We use BERT (Devlin et al., 2019) as the input embedding layer to generate the contextualized word embeddings. Since question $\mathbf{q}^r_t$ is a natural language sequence that may contain entity examples and disturb accurate entity span prediction, we only retain the sentence embeddings $\mathbf{V} = [\mathbf{v}^d_1, \mathbf{v}^d_2, ..., \mathbf{v}^d_n]$ for the next steps, where d is the output dimension of BERT. To encode sentence-level features, the retained embeddings are fed into a standard bi-directional LSTM layer (Graves and Schmidhuber, 2005). The hidden output of BiLSTM can be expressed as follows:

$$\begin{aligned} \overrightarrow{\mathbf{h}}_i &= \text{LSTM}(\overrightarrow{\mathbf{h}}_{i-1}, v^d_i) \\ \overleftarrow{\mathbf{h}}_i &= \text{LSTM}(\overleftarrow{\mathbf{h}}_{i+1}, v^d_i) \end{aligned} \tag{1}$$

where $\overrightarrow{\mathbf{h}}_i$ and $\overleftarrow{\mathbf{h}}_i$ denote the forward and backward output of BiLSTM. The final representation of a word is $\mathbf{h}_i = [\overrightarrow{\mathbf{h}}_i; \overleftarrow{\mathbf{h}}_i]$

3.3 Multi-task Layer

Given the sentence-level features, the purpose of the multi-task layer is to distinguish the existence of each type of entity and predict entity spans.

Entity Existence Discrimination. To model the relationship between sentence and various entity types, we introduce an entity existence discrimination task to identify whether a sentence contains each type of entity. Logically, this task is sensitive to the semantic feature and certain keywords of the sentence, then aggregates this information to make a particular prediction.

In general, the special characters `[CLS]` without semantic property is often used to represent the semantic features of the whole sentence. However, using the semantic feature alone to predict the existence of entities is not enough, because it lacks connections to the entity types. Therefore, we first use an entity type embedding layer to generate entity type features $\mathbf{E} = [\mathbf{e}_1, ..., \mathbf{e}_m]$. Then, to capture the relationship between sentence and entity types, the sentence features and entity type features are incorporated by leveraging the attention mechanism (Vaswani et al., 2017). Given sentence-level features $\mathbf{H} = [\mathbf{h}_1, \mathbf{h}_2, ..., \mathbf{h}_n]$, $\mathbf{h}^{ENT}_j$ represents the incorporated feature associating with entity type t as follow:

$$\mathbf{h}^{ENT}_t = \sum_i^n \alpha_{i,t} \mathbf{W}_v \mathbf{h}_i \tag{2}$$

$$\alpha_{i,t} = \frac{1}{z_t} (\mathbf{W}_q \mathbf{e}_t)^T \mathbf{W}_k \mathbf{h}_i \tag{3}$$

where $\mathbf{W}_q \in \mathbb{R}^{d \times d}$, $\mathbf{W}_k \in \mathbb{R}^{d \times d}$, $\mathbf{W}_v \in \mathbb{R}^{d \times d}$ are parameter matrices and z_t is the normalized factor:

$$z_t = \sum_i^n (\mathbf{W}_q \mathbf{e}_t)^T \mathbf{W}_k \mathbf{h}_i \tag{4}$$

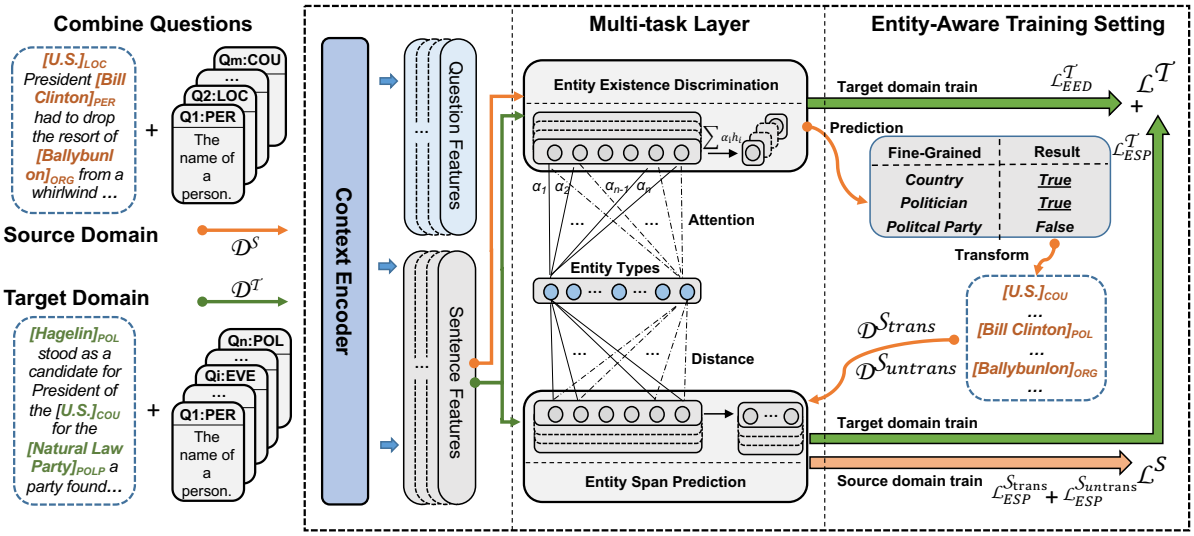

Figure 2: The Domain-specific entity-aware framework(DoSEA) for cross-domain NER. DoSEA has three components: a context encoder, a multi-task layer and an entity-aware training setting. There are different training processes for source and target domains.

The weight $\alpha_{i,t}$ reflects the degree of relevance between word x_i and entity type t. Finally, a contacted feature $\hat{\mathbf{h}} = [\mathbf{h}^{CLS}; \mathbf{h}^{ENT}]$ is fed into a softmax layer to predict the probability of the existence of entities queried by question q_j:

$$\hat{y}_t = \text{softmax}(\mathbf{W}_c\hat{\mathbf{h}} + \mathbf{b}_c) \tag{5}$$

where $\mathbf{W}_c \in \mathbb{R}^{d\times 2}$, $\mathbf{b}_c \in \mathbb{R}^2$ are parameter matrices.

Therefore, the cross-entropy loss for entity existence discrimination task is denoted as follows:

$$\mathcal{L}_{EED} = -\frac{1}{m}\sum_{t}^{m} y_t \log(p(\hat{y}_t)) \tag{6}$$

Entity Span Prediction. The classical MRC-NER model directly uses word embeddings to predict the start and end positions of entities. However, to enhance the task relationship between the EED and ESP tasks, the entity type embeddings $\mathbf{E} = [\mathbf{e}_1, ..., \mathbf{e}_m]$ are shared in both tasks.

Specifically, we improve the entity prediction task in that the start and end positions of entities are predicted by the absolute distance between word embeddings and entity type embeddings. The entity type embedding $\mathbf{e}_t$ is shared as follows:

$$\mathbf{h}_i^{'} = |\text{norm}(\mathbf{h}_i) - \mathbf{e}_t| \tag{7}$$

where $\text{norm}(\cdot)$ is instance normalization function (Ulyanov et al., 2016), and $|\cdot|$ means the absolute value. Therefore, $\mathbf{h}_i^{'}$ represents the absolute distance between word and entity type representations. To extract entity spans, $\mathbf{h}_i^{'}$ is fed into two softmax layers to predict the probability of each token being a start or end position of entities queried by question $\mathbf{q}_t$:

$$P_s(y_{i,t}^s|x_i) = \text{softmax}(\mathbf{W}_s\hat{\mathbf{h}_i^{'}} + \mathbf{b}_s) \tag{8}$$

$$P_e(y_{i,t}^e|x_i) = \text{softmax}(\mathbf{W}_e\hat{\mathbf{h}_i^{'}} + \mathbf{b}_e) \tag{9}$$

where $\mathbf{W}_s, \mathbf{W}_e \in \mathbb{R}^{d\times 2}$, $\mathbf{b}_s, \mathbf{b}_e \in \mathbb{R}^2$ are parameter matrices.

As mentioned above, the training samples are a set of quadruples $(\mathbf{q}_t, y_t, [\mathbf{et}_1^t, \cdots, \mathbf{et}_l^t], \mathbf{x}))$ for DoSEA. Meanwhile, entities $[\mathbf{et}_1^t, \cdots, \mathbf{et}_l^t]$ can be paired with two label sequences $[y_{1,t}^s, ..., y_{n,t}^s]$, $[y_{1,t}^e, ..., y_{n,t}^e]$, which represent the ground-truth label of each token x_i being the start position or end position of the entities queried by question $\mathbf{q}_t$. The cross-entropy losses of start and end positions prediction are denoted as follows:

$$\mathcal{L}_s = -\frac{1}{mn}\sum_{t}^{m}\sum_{i}^{n} y_{i,t}^s \log(P_s(\hat{y}_{i,t}^s|x_i)) \tag{10}$$

$$\mathcal{L}_e = -\frac{1}{mn}\sum_{t}^{m}\sum_{i}^{n} y_{i,t}^e \log(P_e(\hat{y}_{i,t}^e|x_i)) \tag{11}$$

The total loss of entity span prediction task is denoted as follows:

$$\mathcal{L}_{ESP} = \mathcal{L}_s + \mathcal{L}_e \tag{12}$$

3.4 Entity-aware Training Setting

As Figure 2 shows, considering the domain-specific entity types, we design different training processes for $\{\mathcal{S}, \mathcal{T}\}$ domains.

Target Domain Training. We adopt normally jointly training process for target domain $\mathcal{T}$, in which the EED task and the ESP task are training together. Therefore, the training loss for target domain $\mathcal{T}$ is defined as follows:

$$\mathcal{L}^{\mathcal{T}} = \mathcal{L}^{\mathcal{T}}_{ESP} + \gamma \mathcal{L}^{\mathcal{T}}_{EED} \tag{13}$$

where γ is auxiliary task weight.

Source Domain Training. In order to avoid learning inaccurate information, we don't train the EED task with samples from source domain $\mathcal{S}$. In contrast, given the source domain samples, the EED task is utilized to recognize the entities with inconsistent annotations between domains. Then we transform these inconsistent entities into hierarchical entities specific to target domain.

Specifically, we can obtian a set of supertype-subtype pair samples as discussed in the previous Section 3.1, in which a pair of the sample consists of $(\mathbf{q}^{\mathcal{S}}_{sup_t}, y^{\mathcal{S}}_{sup_t}, [\mathbf{et}_1^{sup_t,\mathcal{S}}, \cdots, \mathbf{et}_l^{sup_t,\mathcal{S}}], \mathbf{x}^{\mathcal{S}})$ and $(\mathbf{q}^{\mathcal{T}}_{sub_t}, unknow, unknow, \mathbf{x}^{\mathcal{S}})$. In the beginning, we send the subtype sample into the EED task for acquiring the existence prediction result $\hat{y}^{\mathcal{S}}_{sub_t}$. Then, we presume that if the sentence $\mathbf{x}^{\mathcal{S}}$ contains supertype entities and the hierarchical subtype entities are predicted to exist, these supertype entities can be transformed into subtype entities. Finally, we obtain the transformed samples to train the ESP task together, which consists of $(\mathbf{q}^{\mathcal{S}}_{sup_t}, None, \mathbf{x}^{\mathcal{S}})$ and $(\mathbf{q}^{\mathcal{T}}_{sub_t}, [\mathbf{et}_1^{sub_t,\mathcal{S}}, \cdots, \mathbf{et}_l^{sub_t,\mathcal{S}}], \mathbf{x}^{\mathcal{S}})$. The training loss for source domain $\mathcal{S}$ is defined as follows:

$$\mathcal{L}^{\mathcal{S}} = \mathcal{L}^{\mathcal{S}_{untrans}}_{ESP} + \delta \mathcal{L}^{\mathcal{S}_{trans}}_{ESP} \tag{14}$$

where δ is the data weight, and the source domain samples are divided into untransformed and transformed parts.

4 Experimental Settings

4.1 Datasets.

We take CoNLL-2003 dataset (Sang and Meulder, 2003) as the source domain for all experiments. We use CrossNER datasets (Liu et al., 2021) and MIT Movie Review dataset (Liu et al., 2013) as the target domain datasets. Statistics of these datasets are shown in Table 1.

Domain	Entity Type	Train.	Dev.	Test.
CoNLL-2003 Dataset				
Newswire	4	15.0K	3.5K	3.7K
CrossNER Datasets				
Politics	9	0.2K	0.5K	0.6K
Science	17	0.2K	0.5K	0.5K
Music	13	0.1K	0.4K	0.5K
Literature	11	0.1K	0.4K	0.4K
Artificial Intelligence	12	0.1K	0.4K	0.4K
MIT Movie Review Dataset				
Movie	12	9.7K	-	2.3K

Table 1: Statistics of datasets.

Hierarchical Entity Pairs. CoNLL-2003 is annotated with "Person", "Location", "Organization" and "Miscellaneous" entities. CrossNER datasets consist of five different domains: Politics, Science, Music, Literature and Artificial Intelligence(AI). Moreover, they all contain four overlapped entity types and hierarchical entity subtypes with CoNLL-2003. Especially, Politics domain contains "Politician", "Political Party" and "Country" entities. Science domain contains "Scientist", "University" and "Country" entities. Music domain contains "Artist", "Band" and "Country" entities. Literature domain contains "Writer" and "Country" entities. AI domain contains "Researcher", "University" and "Country" entities. Entity types in MIT Movie Review and CoNLL-2003 are non-overlapping, but MIT Movie Review contains "Actor", "Character" and "Director" entities which are subtypes of "Person" entities.

4.2 Baseline Methods

In the beginning, we consider a classical method, BiLSTM-CRF (Huang et al., 2015), which combined the bi-directional LSTM network and conditional random fields(CRF) for sequence labeling task. Based on this, there are two improved methods Coach (Liu et al., 2020b) and Multi-Cell LSTM (Jia and Zhang, 2020). Coach proposed a two-step approach for cross-domain NER, it first detects whether the tokens are entities or not, then predicts the specific entity types. Multi-Cell LSTM investigated a multi-cell compositional LSTM model structure, which models each entity type using a separate cell state.

Models	Pol.	Sci.	Mus.	Lite.	AI.	Mov.	Avg.
BiLSTM-CRF	53.89	49.12	43.65	41.87	43.18	77.52	51.54
BiLSTM-CRF-joint†	56.60	49.97	44.79	43.03	43.56	78.11	52.68
Coach†	61.50	52.09	51.66	48.35	45.15	78.59	56.22
BERT-Tagger†	66.56	63.73	66.59	59.95	50.37	79.37	64.43
BERT-Tagger-joint†	68.85	65.03	67.59	62.57	58.57	80.04	67.11
Multi-Cell LSTM(BERT)†	70.56	66.42	70.52	66.96	58.28	82.22	69.16
TemplateNER	65.41	62.93	64.67	64.55	57.64	78.56	65.62
MRC-NER	70.23	67.25	70.64	62.53	62.77	83.28	69.45
MRC-NER-joint	72.37	67.70	71.87	66.67	64.65	85.87	71.52
DoSEA w/o $\mathcal{L}_{EED}$+$\mathcal{L}_{ESP}^{S_{trans}}$	72.41	68.20	71.93	66.74	64.77	86.19	71.71
DoSEA w/o $\mathcal{L}_{EED}$	73.31	70.61	72.55	67.35	65.23	86.74	72.63
DoSEA w/o $\mathcal{L}_{ESP}^{S_{trans}}$	73.46	70.13	72.39	67.89	65.24	86.91	72.67
DoSEA(Ours)	**75.52***	**71.69***	**73.10***	**68.59***	**66.03***	**87.31***	**73.71***

Table 2: Cross-domain experiment results on six domain datasets compared to the baseline methods. † indicates the results on CrossNER datasets are from Liu et al. (2021). "joint" postfix means the model jointly trains on both domains. "w/o" is a abbreviation of "without".

We also compare BERT-based methods, including BERT-Tagger (Devlin et al., 2019) and Multi-Cell LSTM(BERT) method that leverages the outputs of BERT as contextualized word embeddings. As the basic model for our proposed framework, MRC-NER (Li et al., 2020) is considered as a baseline method too. In addition, we compare a fine-tune method named TemplateNER (Cui et al., 2021), which is a template-based method by using BART (Lewis et al., 2020) and also shows effectiveness in cross-domain NER. However, we don't compare our method with Liu et al. (2021), because they continue pre-training the language model BERT with abundant domain-related corpus, which is unfair to compare with each other.

4.3 Implementation details

For all methods, word embeddings are fine-tuned in the training process. When training BiLSTM-CRF and Coach, we use the word-level embedding from Pennington et al. (2014) and char-level embedding from Hashimoto et al. (2017) as the input layer. For BERT-based methods, we use the base-sized BERT pre-trained on the Wikipedia corpus to output contextualized word embeddings.

Since the size of training samples in CrossNER is far smaller than CoNLL-2003, we upsample the training samples in the target domain to keep the balance between domains. In the training step, we set the learning rate as 5e-5, entity type embedding dimension d_h as 768, task weight γ as 0.1 and data weight δ as 0.2.

Domain	Separately			Jointly		
	P	R	F1	P	R	F1
Pol.	83.16	77.82	80.40	85.44	80.31	**82.79**
Sci.	83.84	74.92	79.13	86.39	76.10	**80.92**
Mus.	75.86	71.56	73.65	78.71	75.79	**77.22**
Lite.	80.69	69.35	74.59	84.18	72.15	**77.71**
AI.	78.29	67.99	72.74	80.93	70.03	**75.09**
Mov.	90.87	88.21	89.52	92.56	89.66	**91.09**

Table 3: The entity existence discrimination task results on six domain datasets. "Separately" means the EED task is trained alone. "Jointly" means the EED task is jointly training with the ESP task.

5 Results and Discussion

5.1 Main Results

Cross-domain NER. Table 2 shows the overall performances of the proposed DoSEA against baseline methods on six domain datasets. Our proposed DoSEA significantly outperforms prior state-of-the-art methods on all target domains. To be specific, the F1 scores of DoSEA advance the previous best method by +3.15, +3.99, +1.23, +1.92 and +1.38 pp.(*e.g.*, percentage points.) on Politics, Science, Music, Literature, Artificial Intelligence and Movie domains, respectively.

Compared to the sequence-labeling based methods(e.g., BiLSTM-CRF, Coach, Multi-Cell LSTM and BERT-Tagger), results on the MRC-NER method show the best performance while jointly training across domains. We speculate the main

Model	ELE.	COU.	POLP.	POL.	EVE.	ORG.	LOC.	PER.	MISC.
BERT-Tagger-joint	92.07	69.46	78.09	67.75	45.64	63.12	68.55	13.37	47.11
MRC-NER-joint	92.43	70.46	81.48	69.49	**55.59**	69.49	72.55	49.67	50.03
DoSEA(Ours)	**92.61**	**72.00**	**81.51**	**74.66**	45.97	**72.92**	**81.26**	**58.90**	**50.13**

Table 4: Fine-grained comparisons on the Politics domain dataset. "ELE.", "COU.", "POLP.", "POL.", "EVE.", "ORG.", "LOC.", "PER." and "MISC." denote "Election", "Country", "Political Party", "Politician", "Event", "Organization", "Location", "Person" and "Miscellaneous", respectively.

reason is that the MRC-NER model parameters are all shared across domains, while the sequence-labeling based methods use an independent CRF layer for each domain. However, our method shows further performance improvement compared with the MRC-NER method. It demonstrates the effectiveness of our proposed DoSEA, which depth alleviates the subtype conflicts between each entity supertype-subtype pair among domains.

Auxiliary Task. We additionally analyze the performance on the entity existence discrimination task, which plays a crucial role in the entity-aware training setting. As shown in Table 3, the F1 scores of the EED task are 82.79, 80.92, 77.22, 77.71, 75.09 and 91.09 in Politics, Science, Music, Literature, Artificial Intelligence and Movie domains, respectively.

Although we only use annotated samples from the target domain to directly train the EED task, this task still achieves quite good performance compared to the results on cross-domain NER. Meanwhile, we also study the task relationship between the EED and ESP tasks, in which we don't share the entity type features and separately train the EED task. However, the performance suffers a significant decline in all target domains in that case. Therefore, the experimental results indicate the positive effect of the jointly training strategy and provide support for the entity-aware training setting.

5.2 Ablation Study

We conduct ablation studies to explore the effectiveness of each component in the DoSEA framework. To be specific, we consider three settings in the ablation study. (1) We first consider eliminating $\mathcal{L}_{ESP}^{S_{trans}}$ from Eq.14 when using source domain samples to train the DoSEA model. In this case, F1-scores overall target domains suffer a significant decline. (2) To explore the interaction between tasks, $\mathcal{L}_{EED}$ is removed from Eq.13 when using target domain samples to train the DoSEA model. In particular, we use the separate EED model to generate the prediction results about whether the domain-specific subtype entities exist in the source domain samples. Furthermore, the cross-domain NER results suffer a severe drop of about an average of 0.98 pp on the target domains. (3) When we both remove the $\mathcal{L}_{ESP}^{S_{trans}}$ and $\mathcal{L}_{EED}$, DoSEA obtains a similar performance as the MRC-NER method. Eventually, these empirical results suggest that each component in DoSEA is beneficial for cross-domain NER.

5.3 Fined-grained comparisons

To understand the performance of DoSEA at the entity type level, we make fine-grained comparisons on the Politics domain dataset. As mentioned above, we consider three hierarchical entity type pairs in the Politics domain. "Event" to "Election" entity type pair is not considered, because the source domain does not contain the "Event" entity type.

As shown in Table 4, the most interesting result is that the BERT-Tagger method has difficulty in identifying "Person" entities, although there are huge annotated samples for "Person" entities in the source domain. We speculate that the model structure with independent CRF layers seriously hinders the transfer of knowledge from the source domain to the target domain. However, the MRC-NER method achieves relatively high performance on "Person" entity type, which proves the advantages of MRC-NER which shares all model parameters across domains.

MRC-NER method achieves relatively higher performance on "Event" entity type than other methods. We found that the performance on "Event" entity fluctuated greatly during the whole training process. After training data statistic, we think the reason for the instability performance may be that the annotated samples for "Event" entity are very small, only 22 samples. However,

Sentence	In **India**, Prime Ministers **Indira Gandhi** and her son **Rajiv Gandhi** (neither of whom were related to **Mahatma Gandhi**, who was assassinated in 1948), were assassinated in 1984 and 1991 respectively.
Golden labels	**India**: *Country*; **Indira Gandhi**: *Politician*; **Rajiv Gandhi**: *Politician*; **Mahatma Gandhi**: *Politician*
BERT-Tagger-joint	**India**:[*B-location*]; **Indira Gandhi**:[*B-person I-politician*]; **Rajiv Gandhi**:[*B-person I-politician*]; **Mahatma Gandhi**: [*B-person I-person*]
MRC-NER-joint	**India**:[*Location*; *Country*]; **Indira Gandhi**:[*Person*; *Politician*]; **Rajiv Gandhi**:[*Person*; *Politician*] **Mahatma Gandhi**: *Person*
DoSEA(Ours)	**India**: *Country*; **Indira Gandhi**: *Politician*; **Rajiv Gandhi**: *Politician*; **Mahatma Gandhi**: *Politician*

Table 5: The results of an example from Politics domain test dataset.

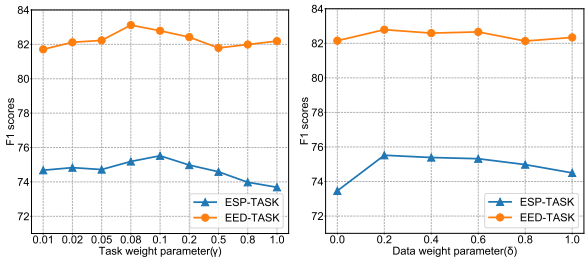

Figure 3: The impact of weight parameters γ and δ on the performance of Politics domain dataset.

our DoSEA accomplishes significant F1-score improvements over all the three hierarchical entity type pairs, demonstrating the effectiveness in reducing the subtype conflicts between domains and providing additional reference about the entity subtypes which are specific to the target domain.

5.4 Case study

Table 5 shows a case study comparing DoSEA with two baseline methods, which is more representative than the others. As we can observe, BERT-Tagger misidentifies `India` as a "Location" entity and also fails to recognize all "Politician" entities. These prediction results hurt the performance of both entity supertype and subtype, which shows a phenomenon of subtype conflict. Meanwhile, the results of the BERT-Tagger method also show label-level mistakes, which presents a challenge to completely identify the correct entities. Since the MRC-NER method can handle the nested entities, it identifies `India` as both a "Location" and "Country" entity, which causes performance degradation on "Location" entity. However, DoSEA correctly identifies all entities in the sentence, which well demonstrates how it can mitigate the subtype conflicts between entity types among different domains.

5.5 Hyperparameter Sensibility

We explore the impact of weight parameter γ in Eq.13 and δ in Eq.14 on Politics domain dataset.

Auxiliary Task Weight. Task weight γ affects the training process for the multi-task inference layer. From Figure 3, we can see that DoSEA keeps a stable F1-scores performance on both entity existence discrimination task and entity span prediction task when $\gamma > 0.01$ and $\gamma < 0.2$, suggesting the stability of the DoSEA. As γ continues to increase, the performance of entity prediction task began to decline, and the best γ parameter is 0.1.

Data Weight. Data weight δ controls how much transfer knowledge the DoSEA model should learn from the transformed subtype entities in the source domain. As we observed, δ have a relatively higher influence on F1 scores of the entity prediction task when $\delta \leq 0.2$ and $\delta \geq 0.6$. Therefore, the reasonable value range for the δ parameter is $\delta \geq 0.2$ and $\delta \leq 0.6$.

6 Conclusion

In this paper, we propose a novel framework named Domain-specific Entity-aware(DoSEA) for cross-domain NER and focus on the issues raised by the domain-specific entity types. Our framework is built on the MRC-NER task, which shares all model parameters across domains. Then, we introduce an entity existence discrimination task and an entity-aware training setting to alleviate the subtypes conflicts, which learns to transform the entities with inconsistent annotations into target domain-specific entities in source domain samples. Experiments show that DoSEA achieves new state-of-the-art performance over six cross-domain benchmarks under the jointly training strategy.

References

Shuguang Chen, Gustavo Aguilar, Leonardo Neves, and Thamar Solorio. 2021. Data augmentation for cross-domain named entity recognition. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 5346–5356. Association for Computational Linguistics.

Leyang Cui, Yu Wu, Jian Liu, Sen Yang, and Yue Zhang. 2021. Template-based named entity recognition using BART. In *Findings of the Association for Computational Linguistics: ACL/IJCNLP 2021*, pages 1835–1845. Association for Computational Linguistics.

Hal Daumé III. 2007. Frustratingly easy domain adaptation. In *Proceedings of the 45th Annual Meeting of the Association for Computational Linguistics*. Association for Computational Linguistics.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 4171–4186. Association for Computational Linguistics.

Alex Graves and Jürgen Schmidhuber. 2005. Framewise phoneme classification with bidirectional LSTM and other neural network architectures. *Neural Networks*, 18(5-6):602–610.

Kazuma Hashimoto, Caiming Xiong, Yoshimasa Tsuruoka, and Richard Socher. 2017. A joint many-task model: Growing a neural network for multiple NLP tasks. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 1923–1933. Association for Computational Linguistics.

Zhiheng Huang, Wei Xu, and Kai Yu. 2015. Bidirectional lstm-crf models for sequence tagging. *arXiv preprint arXiv:1508.01991*.

Chen Jia, Liang Xiao, and Yue Zhang. 2019. Cross-domain NER using cross-domain language modeling. In *Proceedings of the 57th Conference of the Association for Computational Linguistics*, pages 2464–2474. Association for Computational Linguistics.

Chen Jia and Yue Zhang. 2020. Multi-cell compositional LSTM for NER domain adaptation. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 5906–5917. Association for Computational Linguistics.

Young-Bum Kim, Karl Stratos, Ruhi Sarikaya, and Minwoo Jeong. 2015. New transfer learning techniques for disparate label sets. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing of the Asian Federation of Natural Language Processing*, pages 473–482. Association for Computer Linguistics.

Ji Young Lee, Franck Dernoncourt, and Peter Szolovits. 2017. Transfer learning for named-entity recognition with neural networks. *arXiv preprint arXiv:1705.06273*.

Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Veselin Stoyanov, and Luke Zettlemoyer. 2020. BART: denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7871–7880. Association for Computational Linguistics.

Xiaoya Li, Jingrong Feng, Yuxian Meng, Qinghong Han, Fei Wu, and Jiwei Li. 2020. A unified MRC framework for named entity recognition. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 5849–5859. Association for Computational Linguistics.

Xiaoya Li, Fan Yin, Zijun Sun, Xiayu Li, Arianna Yuan, Duo Chai, Mingxin Zhou, and Jiwei Li. 2019. Entity-relation extraction as multi-turn question answering. In *Proceedings of the 57th Conference of the Association for Computational Linguistics*, pages 1340–1350. Association for Computational Linguistics.

Bill Yuchen Lin and Wei Lu. 2018. Neural adaptation layers for cross-domain named entity recognition. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 2012–2022. Association for Computational Linguistics.

Jian Liu, Yubo Chen, Kang Liu, Wei Bi, and Xiaojiang Liu. 2020a. Event extraction as machine reading comprehension. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 2716–2725. Association for Computational Linguistics.

Jingjing Liu, Panupong Pasupat, Scott Cyphers, and James R. Glass. 2013. Asgard: A portable architecture for multilingual dialogue systems. In *IEEE International Conference on Acoustics, Speech and Signal Processing*, pages 8386–8390.

Zihan Liu, Genta Indra Winata, Peng Xu, and Pascale Fung. 2020b. Coach: A coarse-to-fine approach for cross-domain slot filling. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 19–25. Association for Computational Linguistics.

Zihan Liu, Yan Xu, Tiezheng Yu, Wenliang Dai, Ziwei Ji, Samuel Cahyawijaya, Andrea Madotto, and Pascale Fung. 2021. Crossner: Evaluating cross-domain named entity recognition. In *Thirty-Fifth AAAI Conference on Artificial Intelligence, AAAI 2021, Thirty-Third Conference on Innovative Applications of Artificial Intelligence, IAAI 2021, The Eleventh Symposium on Educational Advances in Artificial Intelligence*, pages 13452–13460.

Sinno Jialin Pan and Qiang Yang. 2010. A survey on transfer learning. *IEEE Trans. Knowl. Data Eng.*, 22(10):1345–1359.

Jeffrey Pennington, Richard Socher, and Christopher D. Manning. 2014. Glove: Global vectors for word representation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing*, pages 1532–1543. Association for Computational Linguistics.

Lizhen Qu, Gabriela Ferraro, Liyuan Zhou, Weiwei Hou, and Timothy Baldwin. 2016. Named entity recognition for novel types by transfer learning. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 899–905. Association for Computational Linguistics.

Juan Diego Rodríguez, Adam Caldwell, and Alexander Liu. 2018. Transfer learning for entity recognition of novel classes. In *Proceedings of the 27th International Conference on Computational Linguistics*, pages 1974–1985. Association for Computational Linguistics.

Sebastian Ruder and Barbara Plank. 2017. Learning to select data for transfer learning with bayesian optimization. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 372–382. Association for Computational Linguistics.

Erik F. Tjong Kim Sang and Fien De Meulder. 2003. Introduction to the conll-2003 shared task: Language-independent named entity recognition. In *Proceedings of the Seventh Conference on Natural Language Learning at HLT-NAACL 2003*, pages 142–147. Association for Computational Linguistics.

Dmitry Ulyanov, Andrea Vedaldi, and Victor Lempitsky. 2016. Instance normalization: The missing ingredient for fast stylization. *arXiv preprint arXiv:1607.08022*.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017*, pages 5998–6008.

Rui Wang, Masao Utiyama, Andrew M. Finch, Lemao Liu, Kehai Chen, and Eiichiro Sumita. 2018a. Sentence selection and weighting for neural machine translation domain adaptation. *ACM*, 26(10):1727–1741.

Zhenghui Wang, Yanru Qu, Liheng Chen, Jian Shen, Weinan Zhang, Shaodian Zhang, Yimei Gao, Gen Gu, Ken Chen, and Yong Yu. 2018b. Label-aware double transfer learning for cross-specialty medical named entity recognition. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 1–15. Association for Computational Linguistics.

Mengge Xue, Bowen Yu, Zhenyu Zhang, Tingwen Liu, Yue Zhang, and Bin Wang. 2020. Coarse-to-fine pre-training for named entity recognition. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing*, pages 6345–6354. Association for Computational Linguistics.

Zhilin Yang, Ruslan Salakhutdinov, and William W Cohen. 2017. Transfer learning for sequence tagging with hierarchical recurrent networks. *arXiv preprint arXiv:1703.06345*.

Ying Zhang, Fandong Meng, Yufeng Chen, Jinan Xu, and Jie Zhou. 2021. Target-oriented fine-tuning for zero-resource named entity recognition. In *Findings of the Association for Computational Linguistics: ACL/IJCNLP 2021*, pages 1603–1615. Association for Computational Linguistics.

Incremental Prompting: Episodic Memory Prompt for Lifelong Event Detection

Minqian Liu[♣], **Shiyu Chang**[♠], **Lifu Huang**[♣]
[♣]Virginia Tech, [♠]University of California Santa Barbara
[♣]{minqianliu,lifuh}@vt.edu, [♠]chang87@ucsb.edu

Abstract

Lifelong event detection aims to incrementally update a model with new event types and data while retaining the capability on previously learned old types. One critical challenge is that the model would catastrophically forget old types when continually trained on new data. In this paper, we introduce **E**pisodic **M**emory **P**rompts (**EMP**) to explicitly retain the learned task-specific knowledge. Our method adopts continuous prompt for each task and they are optimized to instruct the model prediction and learn event-specific representation. The EMPs learned in previous tasks are carried along with the model in subsequent tasks, and can serve as a memory module that keeps the old knowledge and transferring to new tasks. Experiment results demonstrate the effectiveness of our method. Furthermore, we also conduct a comprehensive analysis of the new and old event types in lifelong learning.[1]

1 Introduction

Class-incremental event detection (Cao et al., 2020; Yu et al., 2021) is a challenging setting in lifelong learning, where the model is incrementally updated on a continual stream of data for new event types while retaining the event detection capability for all the previously learned types. The main challenge of class-incremental event detection lies in the *catastrophic forgetting* problem, where the model's performance on previously learned types significantly drops after it is trained on new data. Recent studies (Lopez-Paz and Ranzato, 2017; Wang et al., 2019) have revealed that replaying stored samples of old classes can effectively alleviate the catastrophic forgetting issue. However, simply fine-tuning the entire model on the limited stored samples may result in overfitting, especially when the model has a huge set of parameters. How to effectively leverage the limited stored examples still remains an important question.

Prompt learning, which is to simply tune a template-based or continuous prompt appended to the input text while keeping all the other parameters freezed, has recently shown comparable or even better performance than fine-tuning the entire model in many NLP tasks (Brown et al., 2020; Li and Liang, 2021; Lester et al., 2021). It is especially flavored by lifelong learning since it only tunes a small amount of parameters. However, it is still non-trivial to equip prompts with the capability of retaining acquired knowledge and transferring to new tasks in the class-incremental setting.

In this work, we propose an incremental prompting framework that introduces **E**psodic **M**emory **P**rompts (**EMP**) to store and transfer the learned type-specific knowledge. At each training stage, we adopt a learnable prompt for each new event type added from the current task. The prompts are initialized with event type names and fine-tuned with the annotations from each task. To encourage the prompts to always carry and reflect type-specific information, we entangle the feature representation of each event mention with the type-specific prompts by optimizing its type distribution over them. After each training stage, we keep the learned prompts in the model and incorporate new prompts for next task. In this way, the acquired task-specific knowledge can be carried into subsequent tasks. Therefore, our EMP can be considered as a soft episodic memory that preserves the old knowledge and transfers it to new tasks. Our method does not require task identifiers at test time, which enables it to handle the challenging class-incremental setting. Our contributions can be summarized as follows:

• We propose **E**psodic **M**emory **P**rompts (**EMP**) which can explicitly carry previously learned knowledge to subsequent tasks for class-incremental event detection. Extensive experi-

[1]The source code is publicly available at https://github.com/VT-NLP/Incremental_Prompting.

Proceedings of the 29th International Conference on Computational Linguistics, pages 2157–2165
October 12–17, 2022.

ments validate the effectiveness of our method.

• To the best of our knowledge, we are the first to adopt prompting methods for class-incremental event detection. Our framework has the potential to be applied to other incremental learning tasks.

2 Problem Formulation

Given an input text $x_{1:L}$ and a set of target spans $\{(x_i, x_j)\}$ from it, an event detection model needs to assign each target span with an event type in the ontology or label it as *Other* if the span is not an event trigger. For class-incremental event detection, we aim to train a single model f_θ on a sequence of T tasks $\{\mathcal{D}_1, ..., \mathcal{D}_T\}$ that consist of non-overlapping event type sets $\{\mathcal{C}_1, ..., \mathcal{C}_T\}$[2]. In each t-th task, the model needs to classify each mention to any of the types that have seen so far $\mathcal{O}_t = \mathcal{C}_1 \bigcup ... \bigcup \mathcal{C}_t$. The training instances in each task $\mathcal{D}_t$ consist of tuples of an input text $x^t_{1:L}$, a target span $\bar{x}^t$, and its corresponding label $\mathbf{y}^t$ where $\mathbf{y}^t \in \mathcal{C}_t$. For convenience, the notations are for the t-th training stage by default unless denoted explicitly in the following parts of the paper.

3 Approach

3.1 Span-based Event Detection

Given an input sentence $x^t_{1:L}$ from task $\mathcal{D}_t$, we first encode it with BERT (Devlin et al., 2019) to obtain the contextual representations $\mathbf{x}^t_{1:L} = \text{BERT}(x^t_{1:L})$. Note that we freeze BERT's parameters in our method and all baselines. For each span $\bar{x}^t$, we concatenate its starting and ending token representations and feed them into a multilayer perceptron (MLP) to get the span representation $\mathbf{h}^t_{span}$. Then, we apply a linear layer on $\mathbf{h}^t_{span}$ to predict the type distribution of the span $p^t = linear(\mathbf{h}^t_{span})$. We use cross-entropy loss to train the model on $\mathcal{D}_t$:

$$\mathcal{L}_C = -\sum_{(\bar{x}^t, y^t) \in \mathcal{D}_t} \log p^t. \quad (1)$$

3.2 Episodic Memory Prompting

To overcome the catastrophic forgetting and exemplar memory overfitting issues, we design an incremental prompting approach with Episodic Memory Prompts (EMPs) to preserve the knowledge learned from each task and transfer to new tasks.

Given an incoming task $\mathcal{D}_t$ and its corresponding new event type set $\mathcal{C}_t = \{c^t_1, ..., c^t_{n_t}\}$, we first initialize a sequence of new *prompts* $\mathbf{C}^t = [\mathbf{c}^t_1, ..., \mathbf{c}^t_{n_t}]$ where $\mathbf{c}^t_i \in \mathbb{R}^{1\times e}$ is a type-specific prompt for type c^t_i, n_t is the number of event types in the t-th task. e is the embedding dimension size. In our experiments, we use the event type name to initialize each event type prompt $\mathbf{c}^t_i$ (see Appendix A for details). Note that we always preserve the prompts learned from previous tasks, thus the accumulated prompts until the t-th task are represented as $\mathbf{I}^t = [\mathbf{C}^1, ..., \mathbf{C}^t]$. Given a particular sentence $x^t_{1:L}$ from $\mathcal{D}_t$, we concatenate it with the accumulated prompts $\mathbf{I}^t$, encode the whole sequence with BERT, and obtain the sequence of contextual representations $[\tilde{\mathbf{x}}^t_{1:L}; \tilde{\mathbf{I}}^t]$, where $\tilde{\mathbf{x}}^t_{1:L}$ and $\tilde{\mathbf{I}}^t$ denote the sequence of contextual embeddings of $x^t_{1:L}$ and $\mathbf{I}^t$ respectively. $[;]$ is concatenation operation. Then, similar as Section 3.1, we obtain a representation $\tilde{\mathbf{h}}^t_{span}$ for each span based on $\tilde{\mathbf{x}}^t_i$, and predict the logits over all target event types $\tilde{p}^t = linear(\tilde{\mathbf{h}}^t_{span})$.

We expect the EMPs to be specific to the corresponding event types and preserve the knowledge of each event type from previous tasks. So we design an entangled prompt optimization strategy to entangle the feature representation of each span with the event type-specific prompts by computing an event type probability distribution over them. Specifically, given a span representation $\tilde{\mathbf{h}}^t_{span}$ and EMP representations $\tilde{\mathbf{I}}^t$, we compute the probability distribution over all prompts as $\tilde{p}^t_c = \text{MLP}(\tilde{\mathbf{I}}^t) \cdot \tilde{\mathbf{h}}^t_{span}$, where $\cdot$ is the dot product. Finally, we combine the original logits $\tilde{p}^t$ and $\tilde{p}^t_c$ to predict the event type label for each span:

$$\tilde{\mathcal{L}}_C = -\sum_{(\bar{x}^t, y^t) \in \mathcal{D}_t} \log (\tilde{p}^t + \tilde{p}^t_c). \quad (2)$$

At the end of each training stage, we keep the learned prompts from the current task $\mathbf{C}^t$ in the model, and then initialize a new prompt $\mathbf{C}^{t+1}$ for the next task and concatenate it with the previous accumulated prompts $\mathbf{I}^t$ incrementally: $\mathbf{I}^{t+1} = [\mathbf{I}^t; \mathbf{C}^{t+1}]$.

3.3 Lifelong Learning with Experience Replay and Knowledge Distillation

To alleviate the catastrophic forgetting issue, two strategies have been widely applied in many lifelong learning works (Rebuffi et al., 2017; Sun et al., 2020; Cao et al., 2020; Yu et al., 2021): (1) Experience Replay which is to repeatedly optimize the model on the stored previous data in subsequent tasks; and (2) Knowledge Distillation (KD)

[2]Though the type sets from all tasks contain *Other*, they have distinct meanings given different seen types.

that is to ensure the output probabilities and/or features from the current and previous models to be matched, respectively. We also adopt these two baselines to validate the compatibility of our method with other lifelong learning techniques.

Specifically, after training on $\mathcal{D}_t$, we apply the herding algorithm (Welling, 2009) to select 20 training samples for each type into the memory buffer, denoted as $\mathcal{M}$. Similar as Equation 2, the objective for experience replay is:

$$\mathcal{L}_{ER} = -\sum_{(\bar{x}^r, y^r)\in\mathcal{M}} \log\,(\tilde{p}^t + \tilde{p}^t_c). \quad (3)$$

For knowledge distillation, following (Cao et al., 2020), we apply both *prediction-level* and *feature-level* distillation. The objectives for prediction-level KD and feature-level KD are computed as:

$$\mathcal{L}_{PD} = -\sum_{(\bar{x}^r, y^r)\in\mathcal{M}} (\tilde{p}^{t-1} + \tilde{p}^{t-1}_c) \log\,((\tilde{p}^t + \tilde{p}^t_c)),$$

$$\mathcal{L}_{FD} = \sum_{(x^r, (x^r_i, x^r_j), y^r)\in\mathcal{M}} 1 - g(\bar{\mathbf{h}}^{t-1}_{span}, \bar{\mathbf{h}}^{t}_{span}),$$

where g is the cosine similarity function. $\bar{\mathbf{h}}^{t-1}_{span}$ and $\bar{\mathbf{h}}^{t}_{span}$ are l_2-normalized features from the model at $t-1$ and t stages, respectively.

Optimization We combine the multiple objectives with weighting factors α and β as follows:

$$\mathcal{L} = \tilde{\mathcal{L}}_C + \alpha\mathcal{L}_{ER} + \beta(\mathcal{L}_{PD} + \mathcal{L}_{FD}).$$

4 Experiments and Discussions

Experiment Settings We conduct experiments on two benchmark datasets: ACE05-EN (Doddington et al., 2004) and MAVEN (Wang et al., 2020), and construct the class-incremental datasets following the *oracle negative* setting in (Yu et al., 2021). We divided the ontology into 5 subsets with distinct event types, and then use them to constitute a sequence of 5 tasks denoted as $\mathcal{D}_{1:5}$. We use the same partition and task order permutations in (Yu et al., 2021). During the learning process from $\mathcal{D}_1$ to $\mathcal{D}_5$, we constantly test the model on the entire test set (which contains the whole ontology) and take the mentions of unseen event types as negative instances. More implementation details, including parameters, initialization of prompts as well as baselines are shown in Appendix A.

Baselines We consider the following baselines for comparison: (1) **BERT-ED**: simply trains the BERT based event detection model on new tasks without prompts, experience replay or knowledge distillation. It's the same as the span-based event detection baseline in Section 3.1. (2) **KCN** (Cao et al., 2020): use a prototype-based example sampling strategy and hierarchical distillation. As the original approach studied a different setting, we adapt their prediction-level and feature-level distillation as the baseline. (3) **KT** (Yu et al., 2021): transfer knowledge between old types and new types in two directions. (4) **iCaRL*** (Rebuffi et al., 2017): use nearest-mean-of-exemplars rules to perform classification combined with knowledge distillation. iCaRL adopts different strategies for classification, experience replay, and distillation. We thus directly report the result in (Yu et al., 2021) for reference. (5) **EEIL** (Castro et al., 2018): use an additional finetuning stage on the balanced dataset. (6) **BIC** (Wu et al., 2019): use a bias correction layer after the classification layer. (7) **Upperbound**: trains the same model on all types in the datasets jointly. For **iCaRL**, **EEIL**, and **BIC**, we use the same implementation in (Yu et al., 2021). For fair comparison, our approach and all baselines (except for the Upperbound baseline) are built upon **KCN** and use the same experience replay and knowledge distillation strategies described in Section 3.2. We set the exemplar buffer size as 20, and allow one exemplar instance to be used in each training batch instead of the whole memory set. Note that this replay setting is different from the one in (Yu et al., 2021), where we allow much less frequent exemplar replay, and thus our setting is more efficient, challenging, and realistic.

Results We present the main results in Table 1. We have following observations: (1) by comparing the performance of various approaches on Task 1 which are not affected by any catastrophic forgetting, our approach improves 4.1% F-score on MAVEN and 1.3% F-score on ACE05, demonstrating that by incorporating task-specific prompts, event detection itself can be significantly improved. EMPs even provide more improvement on MAVEN which contains a lot more event types than ACE05, suggesting the potential of incorporating EMPs for fine-grained event detection; (2) **KCN** can be viewed as an ablated version of our approach without EMPs. Our approach consistently outperforms **KCN** on almost all tasks on both datasets, demon-

	MAVEN					ACE05-EN				
Task	1	2	3	4	5	1	2	3	4	5
BERT-ED	63.51	39.99	33.36	23.83	22.69	58.30	43.96	38.02	21.53	25.71
iCaRL* (Rebuffi et al., 2017)	18.08	27.03	30.78	31.26	29.77	4.05	5.41	7.25	6.94	8.94
EEIL (Castro et al., 2018)	63.51	50.62	45.16	41.39	38.34	58.30	54.93	52.72	45.18	41.95
BIC (Wu et al., 2019)	63.51	46.69	39.15	31.69	30.47	58.30	45.73	43.28	35.70	30.80
KCN (Cao et al., 2020)	63.51	51.17	46.80	38.72	38.58	58.30	54.71	52.88	44.93	41.10
KT (Yu et al., 2021)	63.51	52.36	47.24	39.51	39.34	58.30	**55.41**	53.95	45.00	42.62
EMP (Ours)	**67.86**	**60.26**	**58.61**	**54.81**	**50.12**	**59.60**	53.19	**55.20**	**45.64**	**43.28**
Upperbound (Ours)	/	/	/	/	68.42	/	/	/	/	67.22

Table 1: Comparison between our approach and baselines in terms of micro F-1 (%) on 5 class-incremental tasks. We report the *averaged* results on 5 permutations of tasks so that the results are independent of randomness.

strating the effectiveness of EMPs on improving class-incremental event detection; (3) Comparing with **BERT-ED**, **KCN** adopts experience replay and knowledge distillation. Their performance gap verifies that these two strategies can dramatically alleviate catastrophic forgetting; (4) There is still a large gap between the current approaches and the upperbound, indicating that catastrophic forgetting still remains a very challenging problem. Note that the only difference in **EEIL**, **BIC**, **KCN**, and **KT** is the lifelong learning techniques they applied, thus these models have identical F-score on Task 1. We also analyze failed examples in Appendix B.

Analysis of New and Old Types in Lifelong Learning Figure 1 shows the F-score on old and new event types in each training stage for our approach and **KT** (Yu et al., 2021) on MAVEN. Our approach consistently outperforms **KT** by a large margin on both old types and new types, demonstrating that our EMPs effectively preserve learned knowledge from old event types and improve event detection when annotations are sufficient. Interestingly, comparing the F-score on new types in Task 1 and old types in Task 2, both methods improve the performance on the types of Task 1, indicating that both methods have the potential of leveraging indirect supervision to improve event detection.

Ablation Study We consider four ablated models based on our EMPs: (1) change the prompt initialization[3] from using event type name representations to using random distribution; (2) remove the entangled prompt optimization but still append the event type prompts to the end of each input sentence and apply Equation 1 only to detect the events; (3) remove the knowledge distillation loss $\mathcal{L}_{PD}$ and $\mathcal{L}_{FD}$; (4) use completely fixed prompts

[3]Appendix A shows the details of prompt initialization. We use the same initialization for the discrete prompt ablation.

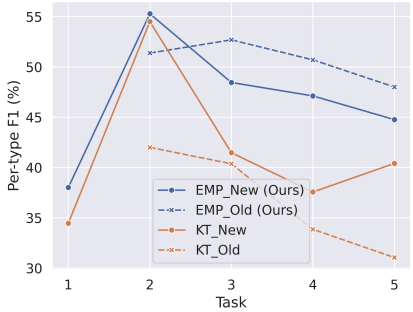

Figure 1: Per-type F1 on old types and new types in each lifelong task on one *randomly selected* permutation of the MAVEN dataset. The F-scores on old and new types reflect the ability to retain acquired knowledge and to learn new types, respectively. Best viewed in color.

to replace the trainable soft prompts. From Table 2, we observes that: (1) using event type names to initialize the prompts is helpful in most tasks; (2) both entangled prompt optimization and knowledge distillation can help alleviate catastrophic forgetting; (3) switching the continuous prompts to discrete prompts degrades the performance significantly, suggesting that the continuous prompts are generally more promising than discrete prompts.

Task	1	2	3	4	5
EMP (Ours)	**67.86**	**60.26**	**58.61**	**54.81**	**50.12**
- w/o EInit	66.73	58.99	57.63	53.98	49.33
- w/o EPO	67.04	59.02	57.79	53.72	49.05
- w/o KD	67.86	57.57	55.83	53.02	48.65
- Discrete	60.13	51.98	50.60	48.97	43.68

Table 2: Ablation study on event-specific prompt initialization (EInit), entangled prompt optimization (EPO), knowledge distillation (KD), and trainable soft prompts (Discrete) on MAVEN. We report the *averaged* results on 5 permutations of tasks.

Effect of Exemplar Buffer Size We conduct an analysis on the effect of exemplar buffer size. We explore the buffer size for each type in {0, 10, 20}. We use **KT** as the baseline when buffer size is 20

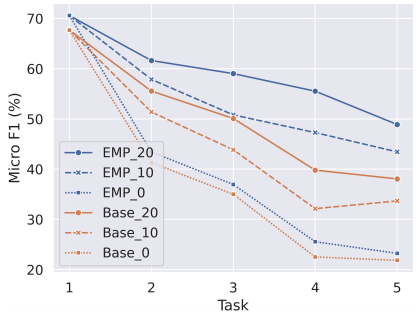

Figure 2: Performance with different buffer size in each task on one *randomly selected* permutation of MAVEN. Best viewed in color.

and 10. Note that when buffer size is 0, we do not adopt either experience replay or knowledge distillation and thus use **BERT-ED** as the baseline. We plot the results on Figure 2. We observed that: (1) Decreasing the buffer size for each type from 20 to 10 degrades the performance of both models. This indicates that reducing data diversity may result in the overfitting on example data, and thus deteriorates the performance; (2) Our method still outperforms the **KT** baseline when storing only half of history examples, which indicates our method is able to utilize the stored examples more effectively. (3) When the buffer size decreases to 0, the performance of both methods drops significantly. This shows that both approaches highly rely on the stored data to overcome the catastrophic forgetting problem. This calls for developing more advance techniques to reduce the dependence on stored examples, as storing past data could result in data leakage in real-world applications.

5 Related Work

Lifelong Event Detection Deep neural networks have shown state-of-the-art performance on supervised event detection (Nguyen et al., 2016; Feng et al., 2016; Zhang et al., 2017; Huang and Ji, 2020; Wang et al., 2021b). However, when moving to lifelong learning setting, their performance significantly drops (Kirkpatrick et al., 2017; Aljundi et al., 2019; Biesialska et al., 2020; Cui et al., 2021; Ke et al., 2021b; Madotto et al., 2021; Ke et al., 2021a; Feng et al., 2022). Though experience replay (Lopez-Paz and Ranzato, 2017; de Masson d'Autume et al., 2019; Guo et al., 2020; Han et al., 2020; Zhao et al., 2022) and knowledge distillation (Chuang et al., 2020; Cao et al., 2020) have shown to be effective in overcoming catastrophic forgetting, they highly rely on the stored data from old tasks, which is not the most realistic setting for lifelong learning.

Prompt Learning Conditioning on large-scale pre-trained language models, prompt learning (Brown et al., 2020; Lester et al., 2021; Liu et al., 2021; Wang et al., 2021a,c, 2022) has shown comparable performance as language model fine-tuning. Specific to lifelong learning, Qin and Joty (2021) use prompt tuning to train the model as a task solver and data generator for lifelong few-shot problem. Zhu et al. (2022) propose continual prompt tuning for dialogue state tracking. To the best of our knowledge, we are the first work to adopt prompt learning for class-incremental event detection.

6 Conclusion

We propose a novel Episodic Memory Prompting (EMP) framework for class-incremental event detection. During each training stage, EMP learns type-specific knowledge via a continuous prompt for each event type. The EMPs trained in previous tasks are kept in the model, such that the acquired task-specific knowledge can be transferred into the following new tasks. Experimental results validate the effectiveness of our method comparing with competitive baselines. Our extensive analysis shows that by employing EMPs, both event detection itself and the incremental learning capability of our approach are significantly improved.

Acknowledgements

We thank the anonymous reviewers and area chair for their valuable time and constructive comments, and the helpful discussions with Zhiyang Xu. We also thank the support from the Amazon Research Awards.

References

Rahaf Aljundi, Min Lin, Baptiste Goujaud, and Yoshua Bengio. 2019. Gradient based sample selection for online continual learning. In *Advances in Neural Information Processing Systems 32: Annual Conference on Neural Information Processing Systems 2019, NeurIPS 2019, December 8-14, 2019, Vancouver, BC, Canada*, pages 11816–11825.

Magdalena Biesialska, Katarzyna Biesialska, and Marta R. Costa-jussà. 2020. Continual lifelong learning in natural language processing: A survey. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 6523–6541,

Barcelona, Spain (Online). International Committee on Computational Linguistics.

Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel Ziegler, Jeffrey Wu, Clemens Winter, Chris Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. 2020. Language models are few-shot learners. In *Advances in Neural Information Processing Systems*, volume 33, pages 1877–1901. Curran Associates, Inc.

Pengfei Cao, Yubo Chen, Jun Zhao, and Taifeng Wang. 2020. Incremental event detection via knowledge consolidation networks. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 707–717, Online. Association for Computational Linguistics.

Francisco M. Castro, Manuel J. Marín-Jiménez, Nicolás Guil, Cordelia Schmid, and Karteek Alahari. 2018. End-to-end incremental learning. In *Computer Vision - ECCV 2018 - 15th European Conference, Munich, Germany, September 8-14, 2018, Proceedings, Part XII*, volume 11216, pages 241–257. Springer.

Yung-Sung Chuang, Shang-Yu Su, and Yun-Nung Chen. 2020. Lifelong language knowledge distillation. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 2914–2924, Online. Association for Computational Linguistics.

Li Cui, Deqing Yang, Jiaxin Yu, Chengwei Hu, Jiayang Cheng, Jingjie Yi, and Yanghua Xiao. 2021. Refining sample embeddings with relation prototypes to enhance continual relation extraction. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 232–243, Online. Association for Computational Linguistics.

Cyprien de Masson d'Autume, Sebastian Ruder, Lingpeng Kong, and Dani Yogatama. 2019. Episodic memory in lifelong language learning. In *Advances in Neural Information Processing Systems*, pages 13122–13131.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, Minneapolis, Minnesota. Association for Computational Linguistics.

George R Doddington, Alexis Mitchell, Mark A Przybocki, Lance A Ramshaw, Stephanie M Strassel, and Ralph M Weischedel. 2004. The automatic content extraction (ace) program-tasks, data, and evaluation. In *Lrec*, volume 2, pages 837–840. Lisbon.

Shaoxiong Feng, Xuancheng Ren, Kan Li, and Xu Sun. 2022. Hierarchical inductive transfer for continual dialogue learning. *CoRR*, abs/2203.10484.

Xiaocheng Feng, Lifu Huang, Duyu Tang, Heng Ji, Bing Qin, and Ting Liu. 2016. A language-independent neural network for event detection. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 66–71, Berlin, Germany. Association for Computational Linguistics.

Yunhui Guo, Mingrui Liu, Tianbao Yang, and Tajana Rosing. 2020. Improved schemes for episodic memory-based lifelong learning. In *Advances in Neural Information Processing Systems*.

Xu Han, Yi Dai, Tianyu Gao, Yankai Lin, Zhiyuan Liu, Peng Li, Maosong Sun, and Jie Zhou. 2020. Continual relation learning via episodic memory activation and reconsolidation. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 6429–6440, Online. Association for Computational Linguistics.

Lifu Huang and Heng Ji. 2020. Semi-supervised new event type induction and event detection. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 718–724.

Zixuan Ke, Bing Liu, Hu Xu, and Lei Shu. 2021a. CLASSIC: Continual and contrastive learning of aspect sentiment classification tasks. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 6871–6883, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Zixuan Ke, Hu Xu, and Bing Liu. 2021b. Adapting BERT for continual learning of a sequence of aspect sentiment classification tasks. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 4746–4755, Online. Association for Computational Linguistics.

James Kirkpatrick, Razvan Pascanu, Neil Rabinowitz, Joel Veness, Guillaume Desjardins, Andrei A Rusu, Kieran Milan, John Quan, Tiago Ramalho, Agnieszka Grabska-Barwinska, et al. 2017. Overcoming catastrophic forgetting in neural networks. *Proceedings of the national academy of sciences*, 114(13):3521–3526.

Brian Lester, Rami Al-Rfou, and Noah Constant. 2021. The power of scale for parameter-efficient prompt tuning. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*,

pages 3045–3059, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Xiang Lisa Li and Percy Liang. 2021. Prefix-tuning: Optimizing continuous prompts for generation. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 4582–4597, Online. Association for Computational Linguistics.

Xiao Liu, Kaixuan Ji, Yicheng Fu, Zhengxiao Du, Zhilin Yang, and Jie Tang. 2021. P-tuning v2: Prompt tuning can be comparable to fine-tuning universally across scales and tasks. *CoRR*, abs/2110.07602.

David Lopez-Paz and Marc'Aurelio Ranzato. 2017. Gradient episodic memory for continual learning. In *Advances in Neural Information Processing Systems*, pages 6467–6476.

Ilya Loshchilov and Frank Hutter. 2019. Decoupled weight decay regularization. In *7th International Conference on Learning Representations, ICLR 2019, New Orleans, LA, USA, May 6-9, 2019*. OpenReview.net.

Andrea Madotto, Zhaojiang Lin, Zhenpeng Zhou, Seungwhan Moon, Paul Crook, Bing Liu, Zhou Yu, Eunjoon Cho, Pascale Fung, and Zhiguang Wang. 2021. Continual learning in task-oriented dialogue systems. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 7452–7467, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Thien Huu Nguyen, Kyunghyun Cho, and Ralph Grishman. 2016. Joint event extraction via recurrent neural networks. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 300–309, San Diego, California. Association for Computational Linguistics.

Chengwei Qin and Shafiq Joty. 2021. LFPT5: A unified framework for lifelong few-shot language learning based on prompt tuning of T5. *CoRR*, abs/2110.07298.

Sylvestre-Alvise Rebuffi, Alexander Kolesnikov, Georg Sperl, and Christoph H. Lampert. 2017. icarl: Incremental classifier and representation learning. In *2017 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2017, Honolulu, HI, USA, July 21-26, 2017*, pages 5533–5542. IEEE Computer Society.

Jingyuan Sun, Shaonan Wang, Jiajun Zhang, and Chengqing Zong. 2020. Distill and replay for continual language learning. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 3569–3579, Barcelona, Spain (Online). International Committee on Computational Linguistics.

Chengyu Wang, Jianing Wang, Minghui Qiu, Jun Huang, and Ming Gao. 2021a. TransPrompt: Towards an automatic transferable prompting framework for few-shot text classification. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 2792–2802, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Hong Wang, Wenhan Xiong, Mo Yu, Xiaoxiao Guo, Shiyu Chang, and William Yang Wang. 2019. Sentence embedding alignment for lifelong relation extraction. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 796–806, Minneapolis, Minnesota. Association for Computational Linguistics.

Sijia Wang, Mo Yu, Shiyu Chang, Lichao Sun, and Lifu Huang. 2021b. Query and extract: Refining event extraction as type-oriented binary decoding. *arXiv preprint arXiv:2110.07476*.

Sijia Wang, Mo Yu, and Lifu Huang. 2022. The art of prompting: Event detection based on type specific prompts. *arXiv preprint arXiv:2204.07241*.

Xiaozhi Wang, Ziqi Wang, Xu Han, Wangyi Jiang, Rong Han, Zhiyuan Liu, Juanzi Li, Peng Li, Yankai Lin, and Jie Zhou. 2020. MAVEN: A Massive General Domain Event Detection Dataset. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1652–1671, Online. Association for Computational Linguistics.

Zifeng Wang, Zizhao Zhang, Chen-Yu Lee, Han Zhang, Ruoxi Sun, Xiaoqi Ren, Guolong Su, Vincent Perot, Jennifer G. Dy, and Tomas Pfister. 2021c. Learning to prompt for continual learning. *CoRR*, abs/2112.08654.

Max Welling. 2009. Herding dynamical weights to learn. In *Proceedings of the 26th Annual International Conference on Machine Learning*, ICML '09, page 1121–1128, New York, NY, USA. Association for Computing Machinery.

Yue Wu, Yinpeng Chen, Lijuan Wang, Yuancheng Ye, Zicheng Liu, Yandong Guo, and Yun Fu. 2019. Large scale incremental learning. In *IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2019, Long Beach, CA, USA, June 16-20, 2019*, pages 374–382. Computer Vision Foundation / IEEE.

Pengfei Yu, Heng Ji, and Prem Natarajan. 2021. Lifelong event detection with knowledge transfer. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 5278–5290, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Tongtao Zhang, Spencer Whitehead, Hanwang Zhang, Hongzhi Li, Joseph Ellis, Lifu Huang, Wei Liu, Heng

Ji, and Shih-Fu Chang. 2017. Improving event extraction via multimodal integration. In *Proceedings of the 25th ACM international conference on Multimedia*, pages 270–278.

Kang Zhao, Hua Xu, Jiangong Yang, and Kai Gao. 2022. Consistent representation learning for continual relation extraction. *CoRR*, abs/2203.02721.

Qi Zhu, Bing Li, Fei Mi, Xiaoyan Zhu, and Minlie Huang. 2022. Continual prompt tuning for dialog state tracking. *CoRR*, abs/2203.06654.

A Experimental Details

Implementation Details During training, we use AdamW (Loshchilov and Hutter, 2019) optimizer with the learning rate set to $1e-4$ and weight decay set to $1e-2$. Different from previous work (Yu et al., 2021), we set the batch size to 1 as we encode each sentence once and consider all target spans in the sentence at the same time. We adopt gradient accumulation with the step set to 8. As the number of batches is large, we apply a periodic replay and distillation strategy with the interval set to 10 to reduce computational cost. For each lifelong task $\mathcal{D}_t$, we set the maximum number of training epochs to 20. We adopt the early stopping strategy with patience 5, i.e., the training stops if the performance on the development set does not increase for 5 epochs. The temperature parameter used in prediction-level distillation is set to 2. The weighted factors for the loss function α and β are computed based on the number of learned event types and new types.

The parameters of each prompt in EMPs are initialized with the corresponding event type name. Specifically, there are three cases in the initialization: (1) If the type name is *single-token* and it is contained in BERT's vocabulary, we directly use the pre-trained embedding of this token to initialize the prompt; (2) If the type name is *multiple-token* and the tokens are contained in BERT's vocabulary, we take the average of the pre-trained embeddings of these tokens to initialize the prompt; (3) If the type name contains *Out-of-Vocabulary (OOV)* tokens, we replace the OOV tokens with the synonyms that are contained in BERT's vocabulary. It is worth noting that we randomly initialize the prompt for the *Other* type and keep updating it throughout all lifelong tasks. We leave how to incorporate more effective prior knowledge into prompts for future work.

B Failure Cases

We show some of typical failure cases in Table 3. We have following observations: (1) the first three examples illustrate the catastrophic forgetting problem in class-incremental event detection. While the model predicted correct event types right after it was trained on those types, it starts to predict wrong types in subsequent tasks. Interestingly, we observed that the model typically predicts the *Other* type or the types relevant to triggers (e.g., *Creating*) when forgetting occurs; (2) the 4th and 5th examples showed that the model sometimes keeps predicting the old types while it is supposed to predict new types in subsequent tasks. (3) the 7th example showed that the model can sometimes correct itself in subsequent tasks, which indicates the experience replay and knowledge distillation have the potentials of improving old types; (4) the last example indicates that in some cases, the model is interfered after trained on a task contained ambiguous types even though it predicts the correct type in all other tasks.

Text	Gold Event Type(s)	Predicted Event Type(s)
The Minnesota Territory itself was **formed** only in 1849 but the area had a rich history well before this.	Coming_to_be ($\mathcal{D}_2$)	$\underline{f_2}$: Coming_to_be; $f_{3:4}$: Other; f_5: Creating ($\mathcal{D}_5$)
He **informed** the Air France chief executive in writing "I did not believe the captain capable of qualifying in the 707."	Telling ($\mathcal{D}_3$)	$\underline{f_3}$: Telling; f_4: Other; f_5: Request ($\mathcal{D}_5$)
Unprepared for the attack, the Swedish attempted to **save**$_{[1]}$ their ships by cutting their anchor ropes and to **flee**$_{[2]}$.	[1] Rescuing ($\mathcal{D}_2$) [2] Escaping ($\mathcal{D}_3$)	[1] $\underline{f_{2:4}}$: Rescuing; f_5: Other [2] $\underline{f_{3:4}}$: Escaping; f_5: Other
After the uprising in Germany was **suppressed**, it flared briefly in several Swiss Cantons.	Control ($\mathcal{D}_3$)	$f_{3:5}$: Hindering ($\mathcal{D}_2$)
Brazilians and Chinese living in the region have been **evacuated**.	Escaping ($\mathcal{D}_3$)	$f_{3:5}$: Removing ($\mathcal{D}_1$)
A surveillance video of the incident was **released** by police four days after the shooting, on 26 November.	Releasing ($\mathcal{D}_3$)	$f_{3:4}$: Other; f_5: Publishing ($\mathcal{D}_5$)
Giral **agreed** to arm the trade unionists in defence of the Republic, and had 60, 000 rifles delivered to the CNT and UGT headquarters, although only 5, 000 were in working order.	Agree_or_refuse_to_act ($\mathcal{D}_4$)	f_4: Other; $\underline{f_5}$: Agree_or_refuse_to_act
Meanwhile, in the city, the Republican government had **reformed** under the leadership of socialist leader Francisco Largo Caballero.	Reforming_a_system ($\mathcal{D}_1$)	$\underline{f_{1,2,4,5}}$: Reforming_a_system; f_3: Change_of_leadership ($\mathcal{D}_3$)

Table 3: Failure analysis of our EMP on the first permutation of MAVEN. The targeted triggers are highlighted in **bold**. $\mathcal{D}_i$ after the event types indicate the type is introduced at i-th task. f_i indicates the model trained after i-th task. We highlight the models predicted the correct types with underline. For example, "Coming_to_be ($\mathcal{D}_2$)" indicates the *Coming_to_be* type is introduced at the 2nd task. "$f_{3:4}$: Other" indicates the models trained after the 3rd and 4th task both predict the *Other* type.

Recent Advances in Text-to-SQL: A Survey of What We Have and What We Expect

Naihao Deng
University of Michigan
dnaihao@umich.edu

Yulong Chen
Westlake University
yulongchen1010@gmail.com

Yue Zhang
Westlake University
yue.zhang@wias.org.cn

Abstract

Text-to-SQL has attracted attention from both the natural language processing and database communities because of its ability to convert the semantics in natural language into SQL queries and its practical application in building natural language interfaces to database systems. The major challenges in text-to-SQL lie in encoding the meaning of natural utterances, decoding to SQL queries, and translating the semantics between these two forms. These challenges have been addressed to different extents by the recent advances. However, there is still a lack of comprehensive surveys for this task. To this end, we review recent progress on text-to-SQL for datasets, methods, and evaluation and provide this systematic survey, addressing the aforementioned challenges and discussing potential future directions. We hope that this survey can serve as quick access to existing work and motivate future research. [1]

1 Introduction

The task of text-to-SQL is to convert natural utterances into SQL queries (Zhong et al., 2017; Yu et al., 2018c). Figure 1 shows an example. Given a user utterance "*What are the major cities in the state of Kansas?*", the system outputs a corresponding SQL that can be used for retrieving the answer from a database. It builds a natural language interface to the database (NLIDB) to help lay users access information in the database (Popescu et al., 2003; Li and Jagadish, 2014), inspiring research in human-computer interaction (Elgohary et al., 2020). Because the SQL query can be regarded as a semantic representation (Guo et al., 2020), text-to-SQL is also a representative task in semantic parsing, helping downstream applications such as question answering (Wang et al., 2020d). Thus, text-to-SQL has attracted researchers in the natural language processing (NLP) and the database (DB) community for decades (Codd, 1970; Hemphill et al., 1990; Dahl et al., 1994; Zelle and Mooney, 1996; Popescu et al., 2003; Bertomeu et al., 2006; Wang et al., 2020a; Scholak et al., 2021b).

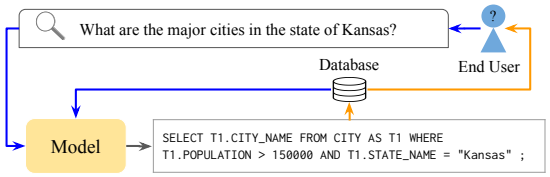

Figure 1: The framework for text-to-SQL systems. Given the database schema and user utterance, the system outputs a corresponding SQL query to query the database system for the result. Appendix B gives more text-to-SQL examples.

The challenges in text-to-SQL lie within three aspects: (1) extracting the meaning of natural utterances (encoding); (2) transforming the extracted meaning into another expression which is pragmatically equivalent to the NL meaning (translating) and; (3) producing the corresponding SQL queries (decoding). A wide range of methods has been investigated to address the technical challenges, from representation learning, intermediate structures, decoding, model structures, training objectives, and other perspectives. In addition, much work has been conducted on data resources and evaluation. However, relatively little work has been done in the literature to provide a comprehensive survey of the landscape. The only exceptions are (Katsogiannis-Meimarakis and Koutrika, 2021) and (Kalajdjieski et al., 2020), but they cover a limited scope. To this end, we aim to provide a systematic survey that involves a broader range of text-to-SQL research and addresses the aforementioned challenges.

In this paper, we survey the recent progress on text-to-SQL, from datasets (§ 2), methods (§ 3) to evaluation (§ 4) [2] and highlight potential direc-

[1]The Github Link for this survey is: https://github.com/text-to-sql-survey-coling22/text-to-sql-survey-coling22.github.io.

[2]Note that most work discussed in this paper is in English

Proceedings of the 29th International Conference on Computational Linguistics, pages 2166–2187
October 12–17, 2022.

Datasets	#Size	#DB	#D	#T/DB	Issues addressed	Sources for data
Spider (Yu et al., 2018c)	10,181	200	138	5.1	Domain generalization	College courses, DabaseAnswers, WikiSQL
WikiSQL (Zhong et al., 2017)	80,654	26,521	-	1	Data size	Wikipedia
Squall (Shi et al., 2020b)	11,468	1,679	-	1	Lexicon-level supervision	WikiTableQuestions
KaggleDBQA (Lee et al., 2021)	272	8	8	2.3	Domain generalization	Real web daabases
IMDB (Yaghmazadeh et al., 2017)	131	1	1	16	-	Internet Movie Database
Yelp (Yaghmazadeh et al., 2017)	128	1	1	7	-	Yelp website
Advising (Finegan-Dollak et al., 2018)	3,898	1	1	10	-	University of Michigan course information
MIMICSQL (Wang et al., 2020d)	10,000	1	1	5	-	Healthcare domain
SEDE (Hazoom et al., 2021)	12,023	1	1	29	SQL template diversity	Stack Exchange

Table 1: The statistic for recent text-to-SQL datasets. #Size, #DB, #D, and #T/DB represent the numbers of question-SQL pairs, databases, domains, and the averaged number of tables per domain, respectively. The "-" in the #D column indicates an unknown number of domains, and the "-" in the Issues Addressed indicates no specific issue addressed by the dataset. Datasets above and below the line are cross-domain and single-domain, respectively. The complete statistic is listed in Table 7 in Appendix C.

tions for future work (§ 5). Apendix A shows the topology for the text-to-SQL task.

2 Datasets

As shown in Table 1, existing text-to-SQL datasets can be classified into three categories: single-domain datasets, cross-domain datasets and others.

Single-Domain Datasets Single-domain text-to-SQL datasets typically collect question-SQL pairs for a single database in some real-world tasks, including early ones such as Academic (Li and Jagadish, 2014), Advising (Finegan-Dollak et al., 2018), ATIS (Price, 1990; Dahl et al., 1994), GeoQuery (Zelle and Mooney, 1996), Yelp and IMDB (Yaghmazadeh et al., 2017), Scholar (Iyer et al., 2017) and Restaurants (Tang and Mooney, 2000; Popescu et al., 2003), as well as recent ones such as SEDE (Hazoom et al., 2021), ESQL (Chen et al., 2021a) and MIMICSQL (Wang et al., 2020d).

These single-domain datasets, particularly the early ones, are usually limited in size, containing only a few hundred to a few thousand examples. Because of the limited size and similar SQL patterns in training and testing phases, text-to-SQL models that are trained on these single-domain datasets can achieve decent performance by simply memorizing the SQL patterns and fail to generalize to unseen SQL queries or SQL queries from other domains (Finegan-Dollak et al., 2018; Yu et al., 2018c). However, since these datasets are adapted from real-life applications, most of them contain domain knowledge (Gan et al., 2021b) and dataset conventions (Suhr et al., 2020). Thus, they are still valuable to evaluate models' ability to generalize to new domains and explore how to incorporate domain knowledge and dataset convention to model predictions.

Appendix B gives a detailed discussion on domain knowledge and dataset convention, and concrete text-to-SQL examples.

Large Scale Cross-domain Datasets Large cross-domain datasets such as WikiSQL (Zhong et al., 2017) and Spider (Yu et al., 2018c) are proposed to better evaluate deep neural models. WikiSQL uses tables extracted from Wikipedia and lets annotators paraphrase questions generated for the tables. Compared to other datasets, WikiSQL is an order of magnitude larger, containing 80,654 natural utterances in total (Zhong et al., 2017). However, WikiSQL contains only simple SQL queries, and only a single table is queried within each SQL query (Yu et al., 2018c).

Yu et al. (2018c) propose Spider, which contains 200 databases with an average of 5 tables for each database, to test models' performance on complicated unseen SQL queries and their ability to generalize to new domains. Furthermore, researchers expand Spider to study various issues of their inter-

unless otherwise specified.

est (Lei et al., 2020; Zeng et al., 2020; Gan et al., 2021b; Taniguchi et al., 2021; Gan et al., 2021a).

Besides, researchers build several large-scale text-to-SQL datasets in different languages such as CSpider (Min et al., 2019a), TableQA (Sun et al., 2020), DuSQL (Wang et al., 2020c) in Chinese, ViText2SQL (Tuan Nguyen et al., 2020) in Vietnamese, and PortugueseSpider (José and Cozman, 2021) in Portuguese. Given that human translation has shown to be more accurate than machine translation (Min et al., 2019a), these datasets are annotated mainly by human experts based on the English Spider dataset. These Spider-based datasets can serve as potential resources for multi-lingual text-to-SQL research.

Other Datasets Several context-dependent text-to-SQL datasets have been proposed, which involve user interactions with the text-to-SQL system in English (Price, 1990; Dahl et al., 1994; Yu et al., 2019a,b) and Chinese (Guo et al., 2021). In addition, researchers collect datasets to study questions in text-to-SQL being answerable or not (Zhang et al., 2020), lexicon-level mapping (Shi et al., 2020b) and cross-domain evaluation for real Web databases (Lee et al., 2021).

Appendix C.1 discusses more details about datasets mentioned in § 2.

3 Methods

Early text-to-SQL systems employ rule-based and template-based methods (Li and Jagadish, 2014; Mahmud et al., 2015), which is suitable for simple user queries and databases. However, with the progress in both DB and NLP communities, recent work focuses on more complex settings (Yu et al., 2018c). In these settings, deep models can be more useful because of their great feature representation ability and generalization ability.

In this survey, we focus on the deep learning methods primarily. We divide these methods employed in text-to-SQL research into Data Augmentation (§ 3.1), Encoding (§ 3.2), Decoding (§ 3.3), Learning Techniques (§ 3.4), and Miscellaneous (§ 3.5).

3.1 Data Augmentation

Data augmentation can help text-to-SQL models handle complex or unseen questions (Zhong et al., 2020b; Wang et al., 2021b), achieve state-of-the-art with less supervised data (Guo et al., 2018), and attain robustness towards different types of questions (Radhakrishnan et al., 2020) .

Typical data augmentation techniques involve paraphrasing questions and filling pre-defined templates for increasing data diversity. Iyer et al. (2017) use the Paraphrase Database (PPDB) (Ganitkevitch et al., 2013) to generate paraphrases for training questions. Appendix B gives an example of this augmentation method. Iyer et al. (2017) and Yu et al. (2018b) collect question-SQL templates and fill in them with DB schema. Researchers also employ neural models to generate natural utterances for sampled SQL queries to acquire more data. For instance, Li et al. (2020a) fine-tune pre-trained T5 model (Raffel et al., 2019) using SQL query as the input to predict natural utterance on WikiSQL, and then randomly synthesize SQL queries from tables in WikiSQL and use the tuned model to generate the corresponding natural utterance.

The quality of the augmented data is important because low-quality data can hurt the performance of the models (Wu et al., 2021). Various approaches have been exploited to improve the quality of the augmented data. After sampling SQL queries, Zhong et al. (2020b) employ an utterance generator to generate natural utterances and a semantic parser to convert the generated natural utterance to SQL queries. To filter out low-quality augmented data, Zhong et al. (2020b) only keep data that have the same generated SQL queries as the sampled ones. Wu et al. (2021) use a hierarchical SQL-to-question generation process to obtain high-quality data. Observing that there is a strong segment-level mapping between SQL queries and natural utterances, Wu et al. (2021) decompose SQL queries into several clauses, translate each clause into a sub-question, and then combine the sub-questions into a complete question.

To increase the diversity of the augmented data, Guo et al. (2018) incorporate a latent variable in their SQL-to-text model to encourage question diversity. Radhakrishnan et al. (2020) augment the WikiSQL dataset by simplifying and compressing questions to simulate the colloquial query behavior of end-users. Wang et al. (2021b) exploit a probabilistic context-free grammar (PCFG) to explicitly model the composition of SQL queries, encouraging sampling compositional SQL queries.

Methods	Adopted by	Applied datasets
Encode type	TypeSQL (Yu et al., 2018a)	WikiSQL
Graph-based	GNN (Bogin et al., 2019a)	Spider
Self-attention	RAT-SQL (Wang et al., 2020a)	Spider
Adapt PLM	SQLova (Hwang et al., 2019)	WikiSQL
Pre-training	TaBERT (Yin et al., 2020)	Spider

Table 2: Typical methods used for encoding in text-to-SQL. The full table of existing methods and more details are listed in Table 8 in Appendix D.

3.2 Encoding

Various methods have been adopted to address the challenges of representing the meaning of questions, representing the structure for DB schema, and linking the DB content to question. We group them into five categories, as shown in Table 2.

Encode Token Types To better encode keywords such as entities and numbers in questions, Yu et al. (2018a) assign a type to each word in the question, with a word being an entity from the knowledge graph, a column, or a number. Yu et al. (2018c) concatenate word embeddings and the corresponding type embeddings to feed into their model.

Graph-based Methods Since DB schemas contain rich structural information, graph-based methods are used to better encode such structures.

As summarized in § 2, datasets prior to Spider typically involve simple DBs that contain only one table or a single DB in both training and testing. As a result, modeling DB schema receives little attention. Because Spider contains complex and different DB in training and testing, Bogin et al. (2019a) propose to use graphs to represent the structure of the DB schemas. Specifically, Bogin et al. (2019a) use nodes to represent tables and columns, edges to represent relationships between tables and columns, such as tables containing columns, primary key, and foreign key constraints, and then use graph neural networks (GNNs) (Li et al., 2016) to encode the graph structure. In their subsequent work, Bogin et al. (2019b) use a graph convolutional network (GCN) to capture DB structures and a gated GCN to select the relevant DB information for SQL generation. RAT-SQL (Wang et al., 2020a) encodes more relationships for DB schemas such as "both columns are from the same table" in their graph.

Graphs have also been used to encode questions together with DB schema. Researchers have been using different types of graphs to capture the semantics in NL and facilitate linking between NL and table schema. Cao et al. (2021) adopt line graph (Gross et al., 2018) to capture multi-hop semantics by meta-path (e.g., an exact match for a question token and column, together with the column belonging to a table can form a 2-hop meta-path) and distinguish between local and non-local neighbors so that different tables and columns will be attended differently. SADGA (Cai et al., 2021) adopts the graph structure to provide a unified encoding for both natural utterances and DB schemas to help question-schema linking. Apart from the relations between entities in both questions and DB schema, the structure for DB schemas, S^2SQL (Hui et al., 2022) integrates syntax dependency among question tokens into the graph to improve model performance. To improve the generalization of the graph method for unseen domains, ShawdowGNN (Chen et al., 2021b) ignores names of tables or columns in the database and uses abstract schemas in the graph projection neural network to obtain delexicalized representations of questions and DB schemas.

Finally, graph-based techniques are also exploited in context-dependent text-to-SQL. For instance, IGSQL (Cai and Wan, 2020) uses a graph encoder to utilize historical information of DB schemas in the previous turns.

Self-attention Models using transformer-based encoder (He et al., 2019; Hwang et al., 2019; Xie et al., 2022) incorporate the original self-attention mechanism by default because it is the building block of the transformer structure.

RAT-SQL (Wang et al., 2020a) applies relation-aware self-attention, a modified version of self-attention (Vaswani et al., 2017), to leverage relations of tables and columns. DuoRAT (Scholak et al., 2021a) also adopts such a relation-aware self-attention in their encoder.

Adapt PLM Various methods have been proposed to leverage the knowledge in pre-trained language models (PLMs) and better align PLM with the text-to-SQL task. PLMs such as BERT (Devlin et al., 2019) are used to encode questions and DB schemas. The modus operandi is to input the concatenation of question words and schema words

to the BERT encoder (Hwang et al., 2019; Choi et al., 2021). Other methods adjust the embeddings by PLMs. On WikiSQL, for instance, X-SQL (He et al., 2019) replaces segment embeddings from the pre-trained encoder by column type embeddings. Guo and Gao (2019) encode two additional feature vectors for matching between question tokens and table cells as well as column names and concatenate them with BERT embeddings of questions and DB schemas.

HydraNet (Lyu et al., 2020) uses BERT to encode the question and an individual column, aligning with the tasks BERT is pre-trained on. After obtaining the BERT representations of all columns, Lyu et al. (2020) select top-ranked columns for SQL prediction. Liu et al. (2021b) train an auxiliary concept prediction module to predict which tables and columns correspond to the question. They detect important question tokens by detecting the largest drop in the confidence score caused by erasing that token in the question. Lastly, they train the PLM with a grounding module using the question tokens and the corresponding tables as well as columns. By empirical studies, Liu et al. (2021b) claim that their approach can awaken the latent grounding from PLM via this erase-and-predict technique.

Pre-training There have been various works proposing different pre-training objectives and using different pre-training data to better align the transformer-based encoder with the text-to-SQL task. For instance, TaBERT (Yin et al., 2020) uses tabular data for pre-training with objectives of masked column prediction and cell value recovery to pre-train BERT. Grappa (Yu et al., 2021) synthesizes question-SQL pairs over tables and pre-trains BERT with the objectives of masked language modeling (MLM) and predicting whether a column appears in the SQL query as well as what SQL operations are triggered. GAP (Shi et al., 2020a) pre-trains BART (Lewis et al., 2020) on synthesized text-to-SQL and tabular data with the objectives of MLM, column prediction, column recovery, and SQL generation.

3.3 Decoding

Various methods have been proposed for decoding to achieve a fine-grained and easier process for SQL generation and bridge the gap between natural language and SQL queries. As shown in Table 3, we group these methods into five main categories and other technologies.

Methods	Adopted by	Applied datasets
Tree	SyntaxSQLNet (Yu et al., 2018b)	Spider
Sketch	SQLNet (Xu et al., 2017)	WikiSQL
Bottom-up	SmBop (Rubin and Berant, 2021)	Spider
Attention	Wang et al. (2019)	WikiSQL
Copy	Wang et al. (2018a)	WikiSQL
IR	IRNet (Guo et al., 2019)	Spider
Others	Global-GCN Bogin et al. (2019b)	Spider
	Kelkar et al. (2020)	Spider

Table 3: Typical methods used for decoding in text-to-SQL. The full table and more details are listed in Table 9 in Appendix D. IR: Intermediate Representation.

Tree-based Seq2Tree (Dong and Lapata, 2016) employs a decoder that generates logical forms in a top-down manner. The components in the sub-tree are generated conditioned on their parents apart from the input question. Note that the syntax of the logical forms is implicitly learned from data for Seq2Tree. Similarly, Seq2AST (Yin and Neubig, 2017) uses an abstract syntax tree (AST) for decoding the target programming language, where the syntax is explicitly integrated with AST. Although both Seq2Tree (Dong and Lapata, 2016) and Seq2AST (Yin and Neubig, 2017) do not study text-to-SQL datasets, their uses of trees inspire tree-based decoding in text-to-SQL. SyntaxSQL-Net (Yu et al., 2018b) employs a tree-based decoding method specific to SQL syntax and recursively calls modules to predict different SQL components.

Sketch-based SQLNet (Xu et al., 2017) designs a sketch aligned with the SQL grammar, and SQL-Net only needs to fill in the slots in the sketch rather than predict both the output grammar and the content. Besides, the sketch captures the dependency of the predictions. Thus, the prediction of one slot is only conditioned on the slots it depends on, which avoids issues of the same SQL query with varied equivalent serializations. Dong and Lapata (2018) decompose the decoding into two stages, where the first decoder predicts a rough sketch, and the second decoder fills in the low-level details conditioned on the question and the sketch. Such coarse-to-fine decoding has also been adopted in other works such as IRNet (Guo et al., 2019). To address the complex SQL queries with

nested structures, RYANSQL (Choi et al., 2021) recursively yields SELECT statements and uses a sketch-based slot filling for each of the SELECT statements.

Bottom-up Both the tree-based and the sketch-based decoding mechanisms can be viewed as top-down decoding mechanisms. Rubin and Berant (2021) use a bottom-up decoding mechanism. Given K trees of height t, the decoder scores trees with height $t+1$ constructed by SQL grammar from the current beam, and K trees with the highest scores are kept. Then, a representation of the new K trees is generated and placed in the new beam.

Attention Mechanism To integrate the encoder-side information at decoding, an attention score is computed and multiplied with hidden vectors from the encoder to get the context vector, which is then used to generate an output token (Dong and Lapata, 2016; Zhong et al., 2017).

Variants of the attention mechanism have been used to better propagate the information encoded from questions and DB schemas to the decoder. SQLNet (Xu et al., 2017) designs column attention, where it uses hidden states from columns multiplied by embeddings for the question to calculate attention scores for a column given the question. Guo and Gao (2018) incorporate bi-attention over question and column names for SQL component selection. Wang et al. (2019) adopt a structured attention (Kim et al., 2017) by computing the marginal probabilities to fill in the slots in their generated abstract SQL queries. Duo-RAT (Scholak et al., 2021a) adopts the relation-aware self-attention mechanism in both its encoder and decoder. Other works that use sequence-to-sequence transformer-based models or decoder-only transformer-based models incorporate the self-attention mechanism by default (Scholak et al., 2021b; Xie et al., 2022).

Copy Mechanism Seq2AST (Yin and Neubig, 2017) and Seq2SQL (Zhong et al., 2017) employ the pointer network (Vinyals et al., 2015) to compute the probability of copying words from the input. Wang et al. (2018a) use types (e.g., columns, SQL operators, constant from questions) to explicitly restrict locations in the query to copy from and develop a new training objective to only copy from the first occurrence in the input. In addition, the copy mechanism is also adopted in context-dependent text-to-SQL task (Wang et al., 2020b).

Intermediate Representations Researchers use intermediate representations to bridge the gap between natural language and SQL queries. IncSQL (Shi et al., 2018) defines actions for different SQL components and let decoder decode actions instead of SQL queries. IRNet (Guo et al., 2019) introduces SemQL, an intermediate representation for SQL queries that can cover most of the challenging Spider benchmark. Specifically, SemQL removes the JOIN ON, FROM and GROUP BY clauses, merges HAVING and WHERE clause for SQL queries. ValueNet (Brunner and Stockinger, 2021) uses SemQL 2.0, which extends SemQL to include value representation. Based on SemQL, NatSQL (Gan et al., 2021c) removes the set operators [3]. Suhr et al. (2020) implement SemQL as a mapping from SQL to a representation with an under-specified FROM clause, which they call SQL^{UF}. Rubin and Berant (2021) employ a relational algebra augmented with SQL operators as the intermediate representations.

However, the intermediate representations are usually designed for a specific dataset and cannot be easily adapted to others (Suhr et al., 2020). To construct a more generalized intermediate representation, Herzig et al. (2021) propose to omit tokens in the SQL query that do not align to any phrase in the utterance.

Inspired by the success of text-to-SQL task, intermediate representations are also studied for SPARQL, another executable language for database systems (Saparina and Osokin, 2021; Herzig et al., 2021).

Others PICARD (Scholak et al., 2021b) and UniSAr (Dou et al., 2022) set constraints to the decoder to prevent generating invalid tokens. Several methods adopt an execution-guided decoding mechanism to exclude non-executable partial SQL queries from the output candidates (Wang et al., 2018b; Hwang et al., 2019). Global-GNN (Bogin et al., 2019b) employs a separately trained discriminative model to rerank the top-K SQL queries in the decoder's output beam, which is to reason about the complete SQL queries instead of considering each word and DB schemas in isolation. Similarly, Kelkar et al. (2020) train a separate discriminator to better search among candidate SQL

[3] The operators that combine the results of two or more SELECT statements, such as INTERSECT

queries. Xu et al. (2017); Yu et al. (2018b); Guo and Gao (2018); Lee (2019) use separate submodules to predict different SQL components, easing the difficulty of generating a complete SQL query. Chen et al. (2020b) employ a gate to select between the output sequence encoded for the question and the output sequence from the previous decoding steps at each step for SQL generation. Inspired by machine translation, Müller and Vlachos (2019) apply byte-pair encoding (BPE) (Sennrich et al., 2016) to compress SQL queries to shorter sequences guided by AST, reducing the difficulties in SQL generation.

3.4 Learning Techniques

Apart from end-to-end supervised learning, different learning techniques have been proposed to help text-to-SQL research. Here we summarize these learning techniques, each addressing a specific issue for the task.

Fully supervised Ni et al. (2020) adopt *active learning* to save human annotation. Yao et al. (2019, 2020); Li et al. (2020b) employ *interactive or imitation learning* to enhance text-to-SQL systems via interactions with end-users. Huang et al. (2018); Wang et al. (2021a); Chen et al. (2021a) adopt *meta-learning* (Finn et al., 2017) for domain generalization. Various *multi-task learning* settings have been proposed to improve text-to-SQL models via enhancing their abilities on some relevant tasks. Chang et al. (2020) set an auxiliary task of mapping between column and condition values. SeaD (Xuan et al., 2021) integrates two denoising objectives to help the model better encode information from the structural data. Hui et al. (2021b) integrate a task of learning the correspondence between questions and DB schemas. Shi et al. (2021) integrate a column classification task to classify which columns appear in the SQL query. McCann et al. (2018) and Xie et al. (2022) train their models with other semantic parsing tasks, which improves models' performance on text-to-SQL task.

Weakly supervised Seq2SQL (Zhong et al., 2017) use *reinforcement learning* to learn `WHERE` clause to allow different orders for components in `WHERE` clause. Liang et al. (2018) leverage memory buffer to reduce the variance of policy gradient estimates when applying reinforcement learning to text-to-SQL. Agarwal et al. (2019) use *meta-learning* and *Bayesian optimization* (Snoek et al., 2012) to learn an auxiliary reward to discount spurious SQL queries in SQL generation. Min et al. (2019b) model the possible SQL queries as a *discrete latent variable* and adopt a hard-*EM*-style parameter updates, letting their model take advantage of the possible pre-computed solutions.

3.5 Miscellaneous

In DB linking, BRIDGE (Lin et al., 2020) appends a representation for the DB cell values mentioned in the question to corresponding fields in the encoded sequence, which links the DB content to the question. Ma et al. (2020) employ an explicit extractor of slots mentioned in the question and then link them with DB schemas.

Model-wise, Finegan-Dollak et al. (2018) use a template-based model which copies slots from the question. Shaw et al. (2021) use a hybrid model which firstly uses a high precision grammar-based approach (NQG) to generate SQL queries, then uses T5 (Raffel et al., 2019) as a back-up if NQG fails. Yan et al. (2020) formulate submodule slot-filling as machine reading comprehension (MRC) task and apply BERT-based MRC models on it. Besides, DT-Fixup (Xu et al., 2021) designs an optimization approach for a deeper Transformer on small datasets for the text-to-SQL task.

In SQL generation, IncSQL (Shi et al., 2018) allows parsers to explore alternative correct action sequences to generate different SQL queries. Brunner and Stockinger (2021) search values in DB to insert values into SQL query.

For context-dependent text-to-SQL, researchers adopt techniques such as turn-level encoder and copy mechanism (Suhr et al., 2018; Zhang et al., 2019; Wang et al., 2020b), constrained decoding (Wang et al., 2020b), dynamic memory decay mechanism (Hui et al., 2021a), treating questions and SQL queries as two modalities, and using bimodal pre-trained models (Zheng et al., 2022).

4 Evaluation

Metrics Table 4 shows widely used automatic evaluation metrics for the text-to-SQL task. Early works evaluate SQL queries by comparing the database querying results executed from the predicted SQL query and the ground-truth (or gold) SQL query (Zelle and Mooney, 1996; Yaghmazadeh et al., 2017) or use *exact string match* to compare the predicted SQL query with the gold one query (Finegan-Dollak et al., 2018). However,

Metrics	Datasets	Errors
Naiive Execution Accuracy	GeoQuery, IMDB, Yelp, WikiSQL, etc	False positive
Exact String Match	Advising, WikiSQL, etc	False negative
Exact Set Match	Spider	False negative
Test Suite Accuracy (execution accuracy with generated databases)	Spider, GeoQuery, etc	False positive

Table 4: The summary of metrics, datasets that use these metrics, and their potential error cases.

execution accuracy can create false positives for semantically different SQL queries even if they yield the same execution results (Yu et al., 2018c). The exact string match can be too strict as two different strings can still have the same semantics (Zhong et al., 2020a). Aware of these issues, Yu et al. (2018c) adopt *exact set match* (ESM) in Spider, deciding the correctness of SQL queries by comparing the sub-clauses of SQL queries. Zhong et al. (2020a) generate databases that can distinguish the predicted SQL query and gold one. Both methods are used as official metrics on Spider.

Evaluation Setup Early single-domain datasets typically use the standard train/dev/test split (Iyer et al., 2017) by splitting the question-SQL pairs randomly. To evaluate generalization to unseen SQL queries within the current domain, Finegan-Dollak et al. (2018) propose SQL query split, where no SQL query is allowed to appear in more than one set among the train, dev, and test sets. Furthermore, Yu et al. (2018c) propose a database split, where the model does not see the databases in the test set in its training time. Other splitting methods also exist to help different research topics (Shaw et al., 2021; Chang et al., 2020).

5 Discussion and Future Directions

Ever since the LUNAR system (Woods et al., 1972; Woods, 1973), systems for retrieving DB information have witnessed an increasing amount of research interest and an enormous growth, especially in the field of text-to-SQL in the deep learning era. With the ever-increasing model performance on the WikiSQL and Spider leaderboards, one can be optimistic because models are becoming more sophisticated than ever. But there are still several challenges to overcome.

First, these sophisticated models suffer a great performance loss when tested against different text-to-SQL datasets from other domains (Suhr et al., 2020; Lee et al., 2021). It is unclear how to incorporate domain knowledge to the models trained on Spider and deploy these models efficiently on different domains, especially those with similar information stored in DB but slightly different DB schemas. Although large-scale datasets promote the cross-domain settings, question-SQL pairs from Spider are free from domain knowledge, ambiguity, or domain convention. Thus, *cross-domain text-to-SQL* needs to be studied in future research to build a practical cross-domain system that can handle real-world requests.

There are different *use cases in real-world scenarios*, which requires models to be robust to different settings and be smart to handle different user requests. For instance, the model trained with DB schemas can need to handle a corrupted table, or no table is provided in its practical use. Besides, the input from users can vary from the standard question input in Spider or WikiSQL, which poses challenges to models trained on these datasets. More user studies need to be done to study how well the current systems serve the end-users and the input pattern from the end-users. Apart from SQL queries, administrators can want to change DB schemas, where a system that can translate the natural language to such DB commands can be helpful. Also, although there are already works on text-to-SQL beyond English (Min et al., 2019a; Tuan Nguyen et al., 2020; José and Cozman, 2021), but we still lack a comprehensive study on multilingual text-to-SQL, which can be challenging but useful in real-life scenarios. Finally, it is important to build NLIDB for people with disabilities. Song et al. (2022) propose speech-to-SQL that translates voice input to SQL queries, which helps visually impaired end users. More work can be done to address various needs from the perspective of end-users, in particular, the needs from minorities.

Text-to-SQL research can also be integrated into *a larger scope of research*. Application-wise, Xu et al. (2020) develop a question answering system for the database, Chen et al. (2020a) generate task-oriented dialogue by retrieving knowledge from the database using the text-to-SQL model. An example of the possible directions is to employ the text-to-SQL model to query databases for fact-checking. Research-wise, Guo et al. (2020) compare SQL queries to other logical forms in semantic pars-

ing, Xie et al. (2022) include text-to-SQL as one of the tasks to achieve a generalized semantic parsing framework. The inter-relations between various logical forms in semantic parsing can be further studied. A generalized framework or a generalized model can come as the fruit for our semantic parsing community.

In hindsight, the development of text-to-SQL has been pushed by the innovation in the general ML/NLP community, such as LSTM (Hochreiter and Schmidhuber, 1997), self-attention (Vaswani et al., 2017), PLMs (Devlin et al., 2019), etc. Recently, *prompt learning* has achieved decent performance on various tasks, in particular, in the low-resource setting (Liu et al., 2021a). Such characteristics align well with the expectation of having a functional text-to-SQL model with a few training samples. Some recent works already explore applying prompt learning to the text-to-SQL task (Xie et al., 2022). The practical expectation for the text-to-SQL task is to deploy the model in different scenarios, requiring robustness across domains. However, prompt learning struggles with being robust, and the performance can be easily affected by the selected data. This misalignment encourages researchers to study how to employ prompt learning in the real-world text-to-SQL task, which can need further understanding of the cross-domain challenges for text-to-SQL.

Another line of research is to *evaluate these sophisticated text-to-SQL systems*. The typical measure is to evaluate the performance of the system on some existing datasets. As there are operational systems using NL input to perform tasks such as getting answers from database management system or building ontologies or playing some games, the performance of these systems can be measured by the diminution of the (human) time taken to get the searched information (Deng et al., 2021; Zhou et al., 2022). While there are context-dependent text-to-SQL datasets available (Yu et al., 2019a,b), researchers can draw inspirations from other fields of research (Zellers et al., 2021) to design interactive set-ups to evaluate text-to-SQL systems. Appendix E discusses tasks relevant to the task of text-to-SQL.

Acknowledgement

Yue Zhang is the corresponding author. We thank all reviewers for their insightful comments, and Rada Mihalcea, Siqi Shen, Winston Wu and Ian Stewart for proofreading and suggestions. The work is funded by the Zhejiang Province Key Project 2022SDXHDX0003.

References

Rishabh Agarwal, Chen Liang, Dale Schuurmans, and Mohammad Norouzi. 2019. Learning to generalize from sparse and underspecified rewards. In *Proceedings of the 36th International Conference on Machine Learning, ICML 2019, 9-15 June 2019, Long Beach, California, USA*, volume 97 of *Proceedings of Machine Learning Research*, pages 130–140. PMLR.

Núria Bertomeu, Hans Uszkoreit, Anette Frank, Hans-Ulrich Krieger, and Brigitte Jörg. 2006. Contextual phenomena and thematic relations in database QA dialogues: results from a Wizard-of-Oz experiment. In *Proceedings of the Interactive Question Answering Workshop at HLT-NAACL 2006*, pages 1–8, New York, NY, USA. Association for Computational Linguistics.

Shikhar Bharadwaj and Shirish Shevade. 2022. Efficient constituency tree based encoding for natural language to bash translation. In *Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 3159–3168.

Ben Bogin, Jonathan Berant, and Matt Gardner. 2019a. Representing schema structure with graph neural networks for text-to-SQL parsing. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 4560–4565, Florence, Italy. Association for Computational Linguistics.

Ben Bogin, Matt Gardner, and Jonathan Berant. 2019b. Global reasoning over database structures for text-to-SQL parsing. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 3659–3664, Hong Kong, China. Association for Computational Linguistics.

Sridevi Bonthu, S Rama Sree, and MHM Krishna Prasad. 2021. Text2PyCode: Machine translation of natural language intent to python source code. In *International Cross-Domain Conference for Machine Learning and Knowledge Extraction*, pages 51–60. Springer.

Ursin Brunner and Kurt Stockinger. 2021. Valuenet: A natural language-to-SQL system that learns from database information. In *2021 IEEE 37th International Conference on Data Engineering (ICDE)*, pages 2177–2182. IEEE.

Paweł Budzianowski, Tsung-Hsien Wen, Bo-Hsiang Tseng, Iñigo Casanueva, Stefan Ultes, Osman Ramadan, and Milica Gašić. 2018. MultiWOZ - a large-scale multi-domain Wizard-of-Oz dataset for task-oriented dialogue modelling. In *Proceedings of the*

2018 Conference on Empirical Methods in Natural Language Processing, pages 5016–5026, Brussels, Belgium. Association for Computational Linguistics.

Ruichu Cai, Jinjie Yuan, Boyan Xu, and Zhifeng Hao. 2021. SADGA: Structure-aware dual graph aggregation network for text-to-SQL. *Advances in Neural Information Processing Systems*, 34.

Yitao Cai and Xiaojun Wan. 2020. IGSQL: Database schema interaction graph based neural model for context-dependent text-to-SQL generation. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 6903–6912, Online. Association for Computational Linguistics.

Ruisheng Cao, Lu Chen, Zhi Chen, Yanbin Zhao, Su Zhu, and Kai Yu. 2021. LGESQL: Line graph enhanced text-to-SQL model with mixed local and non-local relations. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 2541–2555, Online. Association for Computational Linguistics.

Shuaichen Chang, Pengfei Liu, Yun Tang, Jing Huang, Xiaodong He, and Bowen Zhou. 2020. Zero-shot text-to-SQL learning with auxiliary task. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pages 7488–7495.

Chieh-Yang Chen, Pei-Hsin Wang, Shih-Chieh Chang, Da-Cheng Juan, Wei Wei, and Jia-Yu Pan. 2020a. AirConcierge: Generating task-oriented dialogue via efficient large-scale knowledge retrieval. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 884–897, Online. Association for Computational Linguistics.

Sanxing Chen, Aidan San, Xiaodong Liu, and Yangfeng Ji. 2020b. A tale of two linkings: Dynamically gating between schema linking and structural linking for text-to-SQL parsing. In *Proceedings COLING-2020, the 28th International Conference on Computational Linguistics*, pages 2900–2912, Barcelona, Spain (Online). Association for Computational Linguistics.

Yongrui Chen, Xinnan Guo, Chaojie Wang, Jian Qiu, Guilin Qi, Meng Wang, and Huiying Li. 2021a. Leveraging table content for zero-shot text-to-SQL with meta-learning. *ArXiv preprint*, abs/2109.05395.

Zhi Chen, Lu Chen, Yanbin Zhao, Ruisheng Cao, Zihan Xu, Su Zhu, and Kai Yu. 2021b. ShadowGNN: Graph projection neural network for text-to-SQL parser. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 5567–5577, Online. Association for Computational Linguistics.

DongHyun Choi, Myeong Cheol Shin, EungGyun Kim, and Dong Ryeol Shin. 2021. RYANSQL: Recursively applying sketch-based slot fillings for complex text-to-SQL in cross-domain databases. *Computational Linguistics*, 47(2):309–332.

E. F. Codd. 1970. A relational model of data for large shared data banks. *Commun. ACM*, 13(6):377–387.

Deborah A. Dahl, Madeleine Bates, Michael Brown, William Fisher, Kate Hunicke-Smith, David Pallett, Christine Pao, Alexander Rudnicky, and Elizabeth Shriberg. 1994. Expanding the scope of the ATIS task: The ATIS-3 corpus. In *Human Language Technology: Proceedings of a Workshop held at Plainsboro, New Jersey, March 8-11, 1994*.

Naihao Deng, Shuaichen Chang, Peng Shi, Tao Yu, and Rui Zhang. 2021. Prefix-to-SQL: Text-to-SQL generation from incomplete user questions. *arXiv preprint arXiv:2109.13066*.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, Minneapolis, Minnesota. Association for Computational Linguistics.

Li Dong and Mirella Lapata. 2016. Language to logical form with neural attention. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 33–43, Berlin, Germany. Association for Computational Linguistics.

Li Dong and Mirella Lapata. 2018. Coarse-to-fine decoding for neural semantic parsing. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 731–742, Melbourne, Australia. Association for Computational Linguistics.

Longxu Dou, Yan Gao, Mingyang Pan, Dingzirui Wang, Jian-Guang Lou, Wanxiang Che, and Dechen Zhan. 2022. UniSAr: A unified structure-aware autoregressive language model for text-to-SQL. *ArXiv preprint*, abs/2203.07781.

Ahmed Elgohary, Saghar Hosseini, and Ahmed Hassan Awadallah. 2020. Speak to your parser: Interactive text-to-SQL with natural language feedback. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 2065–2077, Online. Association for Computational Linguistics.

Catherine Finegan-Dollak, Jonathan K. Kummerfeld, Li Zhang, Karthik Ramanathan, Sesh Sadasivam, Rui Zhang, and Dragomir Radev. 2018. Improving text-to-SQL evaluation methodology. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 351–360, Melbourne, Australia. Association for Computational Linguistics.

Chelsea Finn, Pieter Abbeel, and Sergey Levine. 2017. Model-agnostic meta-learning for fast adaptation of deep networks. In *Proceedings of the 34th International Conference on Machine Learning, ICML 2017, Sydney, NSW, Australia, 6-11 August 2017*, volume 70 of *Proceedings of Machine Learning Research*, pages 1126–1135. PMLR.

Yujian Gan, Xinyun Chen, Qiuping Huang, Matthew Purver, John R. Woodward, Jinxia Xie, and Pengsheng Huang. 2021a. Towards robustness of text-to-SQL models against synonym substitution. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 2505–2515, Online. Association for Computational Linguistics.

Yujian Gan, Xinyun Chen, and Matthew Purver. 2021b. Exploring underexplored limitations of cross-domain text-to-SQL generalization. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 8926–8931, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Yujian Gan, Xinyun Chen, Jinxia Xie, Matthew Purver, John R. Woodward, John Drake, and Qiaofu Zhang. 2021c. Natural SQL: Making SQL easier to infer from natural language specifications. In *Findings of the Association for Computational Linguistics: EMNLP 2021*, pages 2030–2042, Punta Cana, Dominican Republic. Association for Computational Linguistics.

Juri Ganitkevitch, Benjamin Van Durme, and Chris Callison-Burch. 2013. PPDB: The paraphrase database. In *Proceedings of the 2013 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 758–764, Atlanta, Georgia. Association for Computational Linguistics.

Jonathan L Gross, Jay Yellen, and Mark Anderson. 2018. *Graph theory and its applications*. Chapman and Hall/CRC.

Daya Guo, Yibo Sun, Duyu Tang, Nan Duan, Jian Yin, Hong Chi, James Cao, Peng Chen, and Ming Zhou. 2018. Question generation from SQL queries improves neural semantic parsing. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 1597–1607, Brussels, Belgium. Association for Computational Linguistics.

Jiaqi Guo, Qian Liu, Jian-Guang Lou, Zhenwen Li, Xueqing Liu, Tao Xie, and Ting Liu. 2020. Benchmarking meaning representations in neural semantic parsing. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1520–1540, Online. Association for Computational Linguistics.

Jiaqi Guo, Ziliang Si, Yu Wang, Qian Liu, Ming Fan, Jian-Guang Lou, Zijiang Yang, and Ting Liu. 2021. Chase: A large-scale and pragmatic Chinese dataset for cross-database context-dependent text-to-SQL. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 2316–2331, Online. Association for Computational Linguistics.

Jiaqi Guo, Zecheng Zhan, Yan Gao, Yan Xiao, Jian-Guang Lou, Ting Liu, and Dongmei Zhang. 2019. Towards complex text-to-SQL in cross-domain database with intermediate representation. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 4524–4535, Florence, Italy. Association for Computational Linguistics.

Tong Guo and Huilin Gao. 2018. Bidirectional attention for SQL generation. *ArXiv preprint*, abs/1801.00076.

Tong Guo and Huilin Gao. 2019. Content enhanced BERT-based text-to-SQL generation. *ArXiv preprint*, abs/1910.07179.

Moshe Hazoom, Vibhor Malik, and Ben Bogin. 2021. Text-to-SQL in the wild: A naturally-occurring dataset based on stack exchange data. In *Proceedings of the 1st Workshop on Natural Language Processing for Programming (NLP4Prog 2021)*, pages 77–87, Online. Association for Computational Linguistics.

Pengcheng He, Yi Mao, Kaushik Chakrabarti, and Weizhu Chen. 2019. X-SQL: reinforce schema representation with context. *ArXiv preprint*, abs/1908.08113.

Charles T. Hemphill, John J. Godfrey, and George R. Doddington. 1990. The ATIS spoken language systems pilot corpus. In *Speech and Natural Language: Proceedings of a Workshop Held at Hidden Valley, Pennsylvania, June 24-27,1990*.

Jonathan Herzig, Peter Shaw, Ming-Wei Chang, Kelvin Guu, Panupong Pasupat, and Yuan Zhang. 2021. Unlocking compositional generalization in pre-trained models using intermediate representations. *ArXiv preprint*, abs/2104.07478.

Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long short-term memory. *Neural computation*, 9(8):1735–1780.

Po-Sen Huang, Chenglong Wang, Rishabh Singh, Wen-tau Yih, and Xiaodong He. 2018. Natural language to structured query generation via meta-learning. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 2 (Short Papers)*, pages 732–738, New Orleans, Louisiana. Association for Computational Linguistics.

Binyuan Hui, Ruiying Geng, Qiyu Ren, Binhua Li, Yongbin Li, Jian Sun, Fei Huang, Luo Si, Pengfei Zhu, and Xiaodan Zhu. 2021a. Dynamic hybrid relation network for cross-domain context-dependent semantic parsing. *ArXiv preprint*, abs/2101.01686.

Binyuan Hui, Ruiying Geng, Lihan Wang, Bowen Qin, Bowen Li, Jian Sun, and Yongbin Li. 2022. S^2SQL: Injecting syntax to question-schema interaction graph encoder for text-to-SQL parsers. *ArXiv preprint*, abs/2203.06958.

Binyuan Hui, Xiang Shi, Ruiying Geng, Binhua Li, Yongbin Li, Jian Sun, and Xiaodan Zhu. 2021b. Improving text-to-SQL with schema dependency learning. *ArXiv preprint*, abs/2103.04399.

Wonseok Hwang, Jinyeong Yim, Seunghyun Park, and Minjoon Seo. 2019. A comprehensive exploration on WikiSQL with table-aware word contextualization. *ArXiv preprint*, abs/1902.01069.

Srinivasan Iyer, Ioannis Konstas, Alvin Cheung, Jayant Krishnamurthy, and Luke Zettlemoyer. 2017. Learning a neural semantic parser from user feedback. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 963–973, Vancouver, Canada. Association for Computational Linguistics.

Marcelo Archanjo José and Fabio Gagliardi Cozman. 2021. mRAT-SQL+ GAP: A Portuguese text-to-SQL transformer. In *Brazilian Conference on Intelligent Systems*, pages 511–525. Springer.

Jovan Kalajdjieski, Martina Toshevska, and Frosina Stojanovska. 2020. Recent advances in SQL query generation: A survey. *ArXiv preprint*, abs/2005.07667.

George Katsogiannis-Meimarakis and Georgia Koutrika. 2021. A deep dive into deep learning approaches for text-to-SQL systems. In *Proceedings of the 2021 International Conference on Management of Data*, pages 2846–2851.

Amol Kelkar, Rohan Relan, Vaishali Bhardwaj, Saurabh Vaichal, Chandra Khatri, and Peter Relan. 2020. Bertrand-dr: Improving text-to-SQL using a discriminative re-ranker. *ArXiv preprint*, abs/2002.00557.

Yoon Kim, Carl Denton, Luong Hoang, and Alexander M. Rush. 2017. Structured attention networks. In *5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Conference Track Proceedings*. OpenReview.net.

Chia-Hsuan Lee, Oleksandr Polozov, and Matthew Richardson. 2021. KaggleDBQA: Realistic evaluation of text-to-SQL parsers. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 2261–2273, Online. Association for Computational Linguistics.

Dongjun Lee. 2019. Clause-wise and recursive decoding for complex and cross-domain text-to-SQL generation. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 6045–6051, Hong Kong, China. Association for Computational Linguistics.

Wenqiang Lei, Weixin Wang, Zhixin Ma, Tian Gan, Wei Lu, Min-Yen Kan, and Tat-Seng Chua. 2020. Re-examining the role of schema linking in text-to-SQL. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 6943–6954, Online. Association for Computational Linguistics.

Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Veselin Stoyanov, and Luke Zettlemoyer. 2020. BART: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7871–7880, Online. Association for Computational Linguistics.

Fei Li and Hosagrahar V Jagadish. 2014. Constructing an interactive natural language interface for relational databases. *Proceedings of the VLDB Endowment*, 8(1):73–84.

Ning Li, Bethany Keller, Mark Butler, and Daniel Cer. 2020a. SeqGenSQL–a robust sequence generation model for structured query language. *ArXiv preprint*, abs/2011.03836.

Yujia Li, Daniel Tarlow, Marc Brockschmidt, and Richard S. Zemel. 2016. Gated graph sequence neural networks. In *4th International Conference on Learning Representations, ICLR 2016, San Juan, Puerto Rico, May 2-4, 2016, Conference Track Proceedings*.

Yuntao Li, Bei Chen, Qian Liu, Yan Gao, Jian-Guang Lou, Yan Zhang, and Dongmei Zhang. 2020b. "what do you mean by that?" a parser-independent interactive approach for enhancing text-to-SQL. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 6913–6922, Online. Association for Computational Linguistics.

Chen Liang, Mohammad Norouzi, Jonathan Berant, Quoc V. Le, and Ni Lao. 2018. Memory augmented policy optimization for program synthesis and semantic parsing. In *Advances in Neural Information Processing Systems 31: Annual Conference on Neural Information Processing Systems 2018, NeurIPS 2018, December 3-8, 2018, Montréal, Canada*, pages 10015–10027.

Xi Victoria Lin, Richard Socher, and Caiming Xiong. 2020. Bridging textual and tabular data for cross-domain text-to-SQL semantic parsing. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 4870–4888, Online. Association for Computational Linguistics.

Pengfei Liu, Weizhe Yuan, Jinlan Fu, Zhengbao Jiang, Hiroaki Hayashi, and Graham Neubig. 2021a. Pre-train, prompt, and predict: A systematic survey of

prompting methods in natural language processing. *ArXiv preprint*, abs/2107.13586.

Qian Liu, Dejian Yang, Jiahui Zhang, Jiaqi Guo, Bin Zhou, and Jian-Guang Lou. 2021b. Awakening latent grounding from pretrained language models for semantic parsing. In *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*, pages 1174–1189, Online. Association for Computational Linguistics.

Qin Lyu, Kaushik Chakrabarti, Shobhit Hathi, Souvik Kundu, Jianwen Zhang, and Zheng Chen. 2020. Hybrid ranking network for text-to-SQL. *ArXiv preprint*, abs/2008.04759.

Jianqiang Ma, Zeyu Yan, Shuai Pang, Yang Zhang, and Jianping Shen. 2020. Mention extraction and linking for SQL query generation. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 6936–6942, Online. Association for Computational Linguistics.

Tanzim Mahmud, KM Azharul Hasan, Mahtab Ahmed, and Thwoi Hla Ching Chak. 2015. A rule based approach for NLP based query processing. In *2015 2nd International Conference on Electrical Information and Communication Technologies (EICT)*, pages 78–82. IEEE.

Bryan McCann, Nitish Shirish Keskar, Caiming Xiong, and Richard Socher. 2018. The natural language decathlon: Multitask learning as question answering. *ArXiv preprint*, abs/1806.08730.

Qingkai Min, Yuefeng Shi, and Yue Zhang. 2019a. A pilot study for Chinese SQL semantic parsing. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 3652–3658, Hong Kong, China. Association for Computational Linguistics.

Sewon Min, Danqi Chen, Hannaneh Hajishirzi, and Luke Zettlemoyer. 2019b. A discrete hard EM approach for weakly supervised question answering. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 2851–2864, Hong Kong, China. Association for Computational Linguistics.

Samuel Müller and Andreas Vlachos. 2019. Byte-pair encoding for text-to-SQL generation. *ArXiv preprint*, abs/1910.08962.

Ansong Ni, Pengcheng Yin, and Graham Neubig. 2020. Merging weak and active supervision for semantic parsing. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pages 8536–8543.

Peter Ochieng. 2020. Parot: Translating natural language to SPARQL. *Expert Systems with Applications: X*, 5:100024.

Panupong Pasupat and Percy Liang. 2015. Compositional semantic parsing on semi-structured tables. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 1470–1480, Beijing, China. Association for Computational Linguistics.

Ana-Maria Popescu, Oren Etzioni, and Henry Kautz. 2003. Towards a theory of natural language interfaces to databases. In *Proceedings of the 8th international conference on Intelligent user interfaces*, pages 149–157.

P. J. Price. 1990. Evaluation of spoken language systems: the ATIS domain. In *Speech and Natural Language: Proceedings of a Workshop Held at Hidden Valley, Pennsylvania, June 24-27,1990*.

Karthik Radhakrishnan, Arvind Srikantan, and Xi Victoria Lin. 2020. ColloQL: Robust cross-domain text-to-SQL over search queries. *ArXiv preprint*, abs/2010.09927.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J Liu. 2019. Exploring the limits of transfer learning with a unified text-to-text transformer. *ArXiv preprint*, abs/1910.10683.

Ohad Rubin and Jonathan Berant. 2021. SmBoP: Semi-autoregressive bottom-up semantic parsing. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 311–324, Online. Association for Computational Linguistics.

Irina Saparina and Anton Osokin. 2021. SPARQLing database queries from intermediate question decompositions. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 8984–8998, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Torsten Scholak, Raymond Li, Dzmitry Bahdanau, Harm de Vries, and Chris Pal. 2021a. DuoRAT: Towards simpler text-to-SQL models. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 1313–1321, Online. Association for Computational Linguistics.

Torsten Scholak, Nathan Schucher, and Dzmitry Bahdanau. 2021b. PICARD: Parsing incrementally for constrained auto-regressive decoding from language models. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 9895–9901, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Rico Sennrich, Barry Haddow, and Alexandra Birch. 2016. Neural machine translation of rare words with subword units. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1715–1725, Berlin, Germany. Association for Computational Linguistics.

Peter Shaw, Ming-Wei Chang, Panupong Pasupat, and Kristina Toutanova. 2021. Compositional generalization and natural language variation: Can a semantic parsing approach handle both? In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 922–938, Online. Association for Computational Linguistics.

Peng Shi, Patrick Ng, Zhiguo Wang, Henghui Zhu, Alexander Hanbo Li, Jun Wang, Cicero Nogueira dos Santos, and Bing Xiang. 2020a. Learning contextual representations for semantic parsing with generation-augmented pre-training. *ArXiv preprint*, abs/2012.10309.

Peng Shi, Tao Yu, Patrick Ng, and Zhiguo Wang. 2021. End-to-end cross-domain text-to-SQL semantic parsing with auxiliary task. *ArXiv preprint*, abs/2106.09588.

Tianze Shi, Kedar Tatwawadi, Kaushik Chakrabarti, Yi Mao, Oleksandr Polozov, and Weizhu Chen. 2018. IncSQL: Training incremental text-to-SQL parsers with non-deterministic oracles. *ArXiv preprint*, abs/1809.05054.

Tianze Shi, Chen Zhao, Jordan Boyd-Graber, Hal Daumé III, and Lillian Lee. 2020b. On the potential of lexico-logical alignments for semantic parsing to SQL queries. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 1849–1864, Online. Association for Computational Linguistics.

Jasper Snoek, Hugo Larochelle, and Ryan P. Adams. 2012. Practical bayesian optimization of machine learning algorithms. In *Advances in Neural Information Processing Systems 25: 26th Annual Conference on Neural Information Processing Systems 2012. Proceedings of a meeting held December 3-6, 2012, Lake Tahoe, Nevada, United States*, pages 2960–2968.

Yuanfeng Song, Raymond Chi-Wing Wong, Xuefang Zhao, and Di Jiang. 2022. Speech-to-SQL: Towards speech-driven SQL query generation from natural language question. *ArXiv preprint*, abs/2201.01209.

Alane Suhr, Ming-Wei Chang, Peter Shaw, and Kenton Lee. 2020. Exploring unexplored generalization challenges for cross-database semantic parsing. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 8372–8388, Online. Association for Computational Linguistics.

Alane Suhr, Srinivasan Iyer, and Yoav Artzi. 2018. Learning to map context-dependent sentences to executable formal queries. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 2238–2249, New Orleans, Louisiana. Association for Computational Linguistics.

Ningyuan Sun, Xuefeng Yang, and Yunfeng Liu. 2020. Tableqa: a large-scale Chinese text-to-SQL dataset for table-aware SQL generation. *ArXiv preprint*, abs/2006.06434.

Lappoon R. Tang and Raymond J. Mooney. 2000. Automated construction of database interfaces: Intergrating statistical and relational learning for semantic parsing. In *2000 Joint SIGDAT Conference on Empirical Methods in Natural Language Processing and Very Large Corpora*, pages 133–141, Hong Kong, China. Association for Computational Linguistics.

Yasufumi Taniguchi, Hiroki Nakayama, Kubo Takahiro, and Jun Suzuki. 2021. An investigation between schema linking and text-to-SQL performance. *ArXiv preprint*, abs/2102.01847.

Anh Tuan Nguyen, Mai Hoang Dao, and Dat Quoc Nguyen. 2020. A pilot study of text-to-SQL semantic parsing for Vietnamese. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 4079–4085, Online. Association for Computational Linguistics.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, December 4-9, 2017, Long Beach, CA, USA*, pages 5998–6008.

Oriol Vinyals, Meire Fortunato, and Navdeep Jaitly. 2015. Pointer networks. In *Advances in Neural Information Processing Systems 28: Annual Conference on Neural Information Processing Systems 2015, December 7-12, 2015, Montreal, Quebec, Canada*, pages 2692–2700.

Bailin Wang, Mirella Lapata, and Ivan Titov. 2021a. Meta-learning for domain generalization in semantic parsing. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 366–379, Online. Association for Computational Linguistics.

Bailin Wang, Richard Shin, Xiaodong Liu, Oleksandr Polozov, and Matthew Richardson. 2020a. RAT-SQL: Relation-aware schema encoding and linking for text-to-SQL parsers. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7567–7578, Online. Association for Computational Linguistics.

Bailin Wang, Ivan Titov, and Mirella Lapata. 2019. Learning semantic parsers from denotations with latent structured alignments and abstract programs. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 3774–3785, Hong Kong, China. Association for Computational Linguistics.

Bailin Wang, Wenpeng Yin, Xi Victoria Lin, and Caiming Xiong. 2021b. Learning to synthesize data for semantic parsing. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 2760–2766, Online. Association for Computational Linguistics.

Chenglong Wang, Marc Brockschmidt, and Rishabh Singh. 2018a. Pointing out SQL queries from text.

Chenglong Wang, Kedar Tatwawadi, Marc Brockschmidt, Po-Sen Huang, Yi Mao, Oleksandr Polozov, and Rishabh Singh. 2018b. Robust text-to-SQL generation with execution-guided decoding. *ArXiv preprint*, abs/1807.03100.

Huajie Wang, Mei Li, and Lei Chen. 2020b. PG-GSQL: Pointer-generator network with guide decoding for cross-domain context-dependent text-to-SQL generation. In *Proceedings of COLING-2022, the 28th International Conference on Computational Linguistics*, pages 370–380, Barcelona, Spain (Online). Association for Computational Linguistics.

Lijie Wang, Ao Zhang, Kun Wu, Ke Sun, Zhenghua Li, Hua Wu, Min Zhang, and Haifeng Wang. 2020c. DuSQL: A large-scale and pragmatic Chinese text-to-SQL dataset. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 6923–6935, Online. Association for Computational Linguistics.

Ping Wang, Tian Shi, and Chandan K. Reddy. 2020d. Text-to-SQL generation for question answering on electronic medical records. In *WWW '20: The Web Conference 2020, Taipei, Taiwan, April 20-24, 2020*, pages 350–361. ACM / IW3C2.

W. Woods, Ronald Kaplan, and Bonnie Webber. 1972. The lunar sciences natural language information system.

William A Woods. 1973. Progress in natural language understanding: an application to lunar geology. In *Proceedings of the June 4-8, 1973, national computer conference and exposition*, pages 441–450.

Kun Wu, Lijie Wang, Zhenghua Li, Ao Zhang, Xinyan Xiao, Hua Wu, Min Zhang, and Haifeng Wang. 2021. Data augmentation with hierarchical SQL-to-question generation for cross-domain text-to-SQL parsing. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 8974–8983, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Tianbao Xie, Chen Henry Wu, Peng Shi, Ruiqi Zhong, Torsten Scholak, Michihiro Yasunaga, Chien-Sheng Wu, Ming Zhong, Pengcheng Yin, Sida I Wang, et al. 2022. UnifiedSKG: Unifying and multi-tasking structured knowledge grounding with text-to-text language models. *ArXiv preprint*, abs/2201.05966.

Peng Xu, Dhruv Kumar, Wei Yang, Wenjie Zi, Keyi Tang, Chenyang Huang, Jackie Chi Kit Cheung, Simon J.D. Prince, and Yanshuai Cao. 2021. Optimizing deeper transformers on small datasets. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 2089–2102, Online. Association for Computational Linguistics.

Silei Xu, Sina Semnani, Giovanni Campagna, and Monica Lam. 2020. AutoQA: From databases to QA semantic parsers with only synthetic training data. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 422–434, Online. Association for Computational Linguistics.

Xiaojun Xu, Chang Liu, and Dawn Song. 2017. SQLNet: Generating structured queries from natural language without reinforcement learning. *ArXiv preprint*, abs/1711.04436.

Kuan Xuan, Yongbo Wang, Yongliang Wang, Zujie Wen, and Yang Dong. 2021. Sead: End-to-end text-to-SQL generation with schema-aware denoising. *ArXiv preprint*, abs/2105.07911.

Navid Yaghmazadeh, Yuepeng Wang, Isil Dillig, and Thomas Dillig. 2017. SQLizer: query synthesis from natural language. *Proceedings of the ACM on Programming Languages*, 1(OOPSLA):1–26.

Zeyu Yan, Jianqiang Ma, Yang Zhang, and Jianping Shen. 2020. SQL generation via machine reading comprehension. In *Proceedings of COLING-2022, the 28th International Conference on Computational Linguistics*, pages 350–356, Barcelona, Spain (Online). Association for Computational Linguistics.

Ziyu Yao, Yu Su, Huan Sun, and Wen-tau Yih. 2019. Model-based interactive semantic parsing: A unified framework and a text-to-SQL case study. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 5447–5458, Hong Kong, China. Association for Computational Linguistics.

Ziyu Yao, Yiqi Tang, Wen-tau Yih, Huan Sun, and Yu Su. 2020. An imitation game for learning semantic parsers from user interaction. In *Proceedings of the 2020 Conference on Empirical Methods*

in Natural Language Processing (EMNLP), pages 6883–6902, Online. Association for Computational Linguistics.

Xi Ye, Qiaochu Chen, Xinyu Wang, Isil Dillig, and Greg Durrett. 2020. Sketch-driven regular expression generation from natural language and examples. *Transactions of the Association for Computational Linguistics*, 8:679–694.

Pengcheng Yin and Graham Neubig. 2017. A syntactic neural model for general-purpose code generation. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 440–450, Vancouver, Canada. Association for Computational Linguistics.

Pengcheng Yin, Graham Neubig, Wen-tau Yih, and Sebastian Riedel. 2020. TaBERT: Pretraining for joint understanding of textual and tabular data. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 8413–8426, Online. Association for Computational Linguistics.

Tao Yu, Zifan Li, Zilin Zhang, Rui Zhang, and Dragomir Radev. 2018a. TypeSQL: Knowledge-based type-aware neural text-to-SQL generation. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 2 (Short Papers)*, pages 588–594, New Orleans, Louisiana. Association for Computational Linguistics.

Tao Yu, Chien-Sheng Wu, Xi Victoria Lin, Bailin Wang, Yi Chern Tan, Xinyi Yang, Dragomir R. Radev, Richard Socher, and Caiming Xiong. 2021. Grappa: Grammar-augmented pre-training for table semantic parsing. In *9th International Conference on Learning Representations, ICLR 2021, Virtual Event, Austria, May 3-7, 2021*. OpenReview.net.

Tao Yu, Michihiro Yasunaga, Kai Yang, Rui Zhang, Dongxu Wang, Zifan Li, and Dragomir Radev. 2018b. SyntaxSQLNet: Syntax tree networks for complex and cross-domain text-to-SQL task. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 1653–1663, Brussels, Belgium. Association for Computational Linguistics.

Tao Yu, Rui Zhang, Heyang Er, Suyi Li, Eric Xue, Bo Pang, Xi Victoria Lin, Yi Chern Tan, Tianze Shi, Zihan Li, Youxuan Jiang, Michihiro Yasunaga, Sungrok Shim, Tao Chen, Alexander Fabbri, Zifan Li, Luyao Chen, Yuwen Zhang, Shreya Dixit, Vincent Zhang, Caiming Xiong, Richard Socher, Walter Lasecki, and Dragomir Radev. 2019a. CoSQL: A conversational text-to-SQL challenge towards cross-domain natural language interfaces to databases. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 1962–1979, Hong Kong, China. Association for Computational Linguistics.

Tao Yu, Rui Zhang, Kai Yang, Michihiro Yasunaga, Dongxu Wang, Zifan Li, James Ma, Irene Li, Qingning Yao, Shanelle Roman, Zilin Zhang, and Dragomir Radev. 2018c. Spider: A large-scale human-labeled dataset for complex and cross-domain semantic parsing and text-to-SQL task. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 3911–3921, Brussels, Belgium. Association for Computational Linguistics.

Tao Yu, Rui Zhang, Michihiro Yasunaga, Yi Chern Tan, Xi Victoria Lin, Suyi Li, Heyang Er, Irene Li, Bo Pang, Tao Chen, Emily Ji, Shreya Dixit, David Proctor, Sungrok Shim, Jonathan Kraft, Vincent Zhang, Caiming Xiong, Richard Socher, and Dragomir Radev. 2019b. SParC: Cross-domain semantic parsing in context. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 4511–4523, Florence, Italy. Association for Computational Linguistics.

John M Zelle and Raymond J Mooney. 1996. Learning to parse database queries using inductive logic programming. In *Proceedings of the national conference on artificial intelligence*, pages 1050–1055.

Rowan Zellers, Ari Holtzman, Elizabeth Clark, Lianhui Qin, Ali Farhadi, and Yejin Choi. 2021. TuringAdvice: A generative and dynamic evaluation of language use. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 4856–4880, Online. Association for Computational Linguistics.

Jichuan Zeng, Xi Victoria Lin, Steven C.H. Hoi, Richard Socher, Caiming Xiong, Michael Lyu, and Irwin King. 2020. Photon: A robust cross-domain text-to-SQL system. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics: System Demonstrations*, pages 204–214, Online. Association for Computational Linguistics.

Rui Zhang, Tao Yu, Heyang Er, Sungrok Shim, Eric Xue, Xi Victoria Lin, Tianze Shi, Caiming Xiong, Richard Socher, and Dragomir Radev. 2019. Editing-based SQL query generation for cross-domain context-dependent questions. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 5338–5349, Hong Kong, China. Association for Computational Linguistics.

Yusen Zhang, Xiangyu Dong, Shuaichen Chang, Tao Yu, Peng Shi, and Rui Zhang. 2020. Did you ask a good question? a cross-domain question intention classification benchmark for text-to-SQL. *ArXiv preprint*, abs/2010.12634.

Yanzhao Zheng, Haibin Wang, Baohua Dong, Xingjun Wang, and Changshan Li. 2022. HIE-SQL: History information enhanced network for context-dependent text-to-SQL semantic parsing. *ArXiv preprint*, abs/2203.07376.

Ruiqi Zhong, Tao Yu, and Dan Klein. 2020a. Semantic evaluation for text-to-SQL with distilled test suites. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 396–411, Online. Association for Computational Linguistics.

Victor Zhong, Mike Lewis, Sida I. Wang, and Luke Zettlemoyer. 2020b. Grounded adaptation for zero-shot executable semantic parsing. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 6869–6882, Online. Association for Computational Linguistics.

Victor Zhong, Caiming Xiong, and Richard Socher. 2017. Seq2SQL: Generating structured queries from natural language using reinforcement learning. *ArXiv preprint*, abs/1709.00103.

Jiawei Zhou, Jason Eisner, Michael Newman, Emmanouil Antonios Platanios, and Sam Thomson. 2022. Online semantic parsing for latency reduction in task-oriented dialogue. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1554–1576, Dublin, Ireland. Association for Computational Linguistics.

A Topology for Text-to-SQL

Figure 5 shows the topology for the text-to-SQL task.

B Text-to-SQL Examples

B.1 Table and Database

Table 6 shows an example of the table in the database for Restaurants dataset. The domain for this dataset is restaurant information, where questions are typically about food type, restaurant location, etc.

There is a big difference in terms of how many tables a database has. For restaurants, there are 3 tables in the database, while there are 32 tables in ATIS (Suhr et al., 2020).

B.2 Domain Knowledge

Question: Will undergrads be okay to take 581 ?
SQL query:

```
SELECT DISTINCT T1.ADVISORY_REQUIREMENT ,
 T1.ENFORCED_REQUIREMENT , T1.NAME FROM
COURSE AS T1 WHERE T1.DEPARTMENT =
"EECS" AND T1.NUMBER = 581 ;
```

In Advising dataset, Department "EECS" is considered as domain knowledge where "581" in the utterance means a course in "EECS" department with course number "581".

B.3 Dataset Convention

Question: Give me some restaurants in alameda ?
SQL query:

```
SELECT T1.HOUSE_NUMBER ,
T2.NAME FROM LOCATION AS T1 , RESTAURANT
 AS T2 WHERE T1.CITY_NAME = "alameda"
AND T2.ID = T1.RESTAURANT_ID ;
```

In Restaurants dataset, when the user queries "restaurants", by dataset convention, the corresponding SQL query returns the column "HOUSE_NUMBER" and "NAME".

B.4 Text-to-SQL Templates

An example of the template for text-to-SQL pair used by Iyer et al. (2017) is as follows:
Question template: Get all <ENT1>.<NAME> having <ENT2>.<COL1>.<NAME> as <ENT2>.<COL1>.<TYPE>
SQL query template:

```
SELECT <ENT1>.<DEF> FROM JOIN_FROM(
<ENT1>, <ENT2>) WHERE JOIN_WHERE(<ENT1>,
<ENT2>) AND
<ENT2>.<COL1> = <ENT2>.<COL1>.<TYPE> ;
```

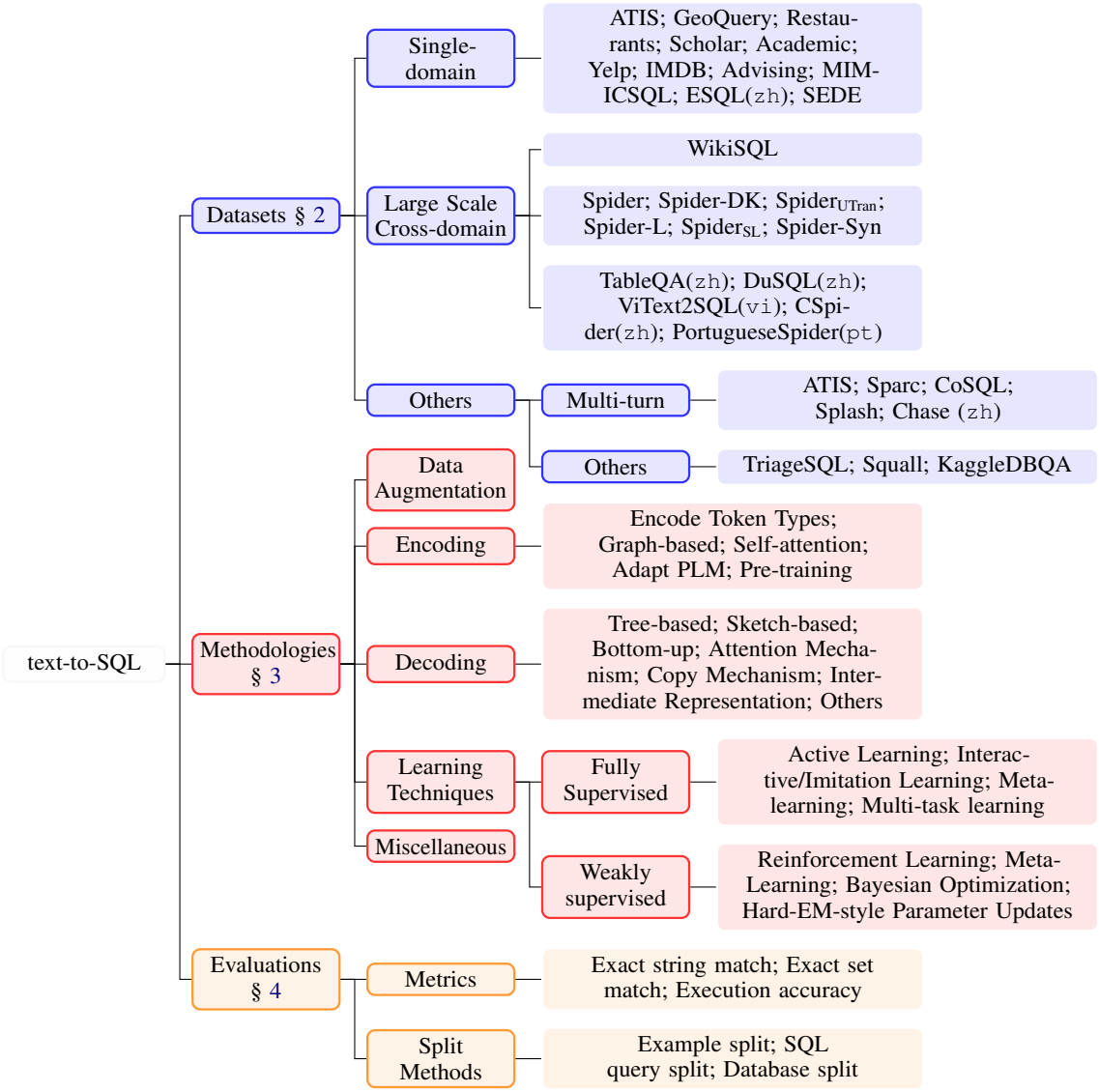

Table 5: Topology for text-to-SQL. Format adapted from Liu et al. (2021a).

CITY_NAME*	COUNTY	REGION
VARCHAR(255)	VARCHAR(255)	VARCHAR(255)
Alameda	Alameda County	Bay Area
Alamo	Contra Costa County	Bay Area
Albany	Alameda County	Bay Area
...	...	...

Table 6: Geography, one of the tables in Restaurants database. * denotes the primary key of this table. We only include 3 rows for demonstration purpose.

Generated question: Get all author having dataset as DATASET_TYPE
Generated SQL query:

```
SELECT author.authorId
FROM author , writes , paper ,
paperDataset , dataset WHERE author.
authorId = writes.authorId
AND writes.paperId = paper.paperId
AND paper.paperId = paperDataset.paperId
 AND paperDataset.datasetId = dataset.
datasetId AND dataset.datasetName =
DATASET_TYPE ;
```

, where they populate the slots in the templates with table and column names from the database schema, as well as join the corresponding tables accordingly.

An example of the PPDB (Ganitkevitch et al., 2013) paraphrasing is “thrown into jail” and “imprisoned”. The English portion of PPDB contains over 220 million paraphrasing pairs.

B.5 Complexity of Natural Language and SQL Query Pairs

In terms of the complexity for SQL queries, Finegan-Dollak et al. (2018) find that models perform better on shorter SQL queries than longer SQL queries, which indicates that shorter SQL queries are easier in general. Yu et al. (2018c) define the SQL hardness as the number of SQL components. The SQL query is harder when it contains more SQL keywords such as `GROUP BY` and nested subqueries. Yu et al. (2018c) gives some examples of SQL queries with different difficulty levels:

Easy:

```
SELECT COUNT(*)
FROM cars_data
WHERE cylinders > 4 ;
```

Medium:

```
SELECT T2.name, COUNT(*)
FROM concert AS T1 JOIN stadium AS T2 ON
 T1.stadium_id = T2.stadium_id GROUP
BY T1.stadium_id ;
```

Hard:

```
SELECT T1.country_name
FROM countries AS T1 JOIN continents AS
T2 ON T1.continent = T2.cont_id JOIN
car_makers AS T3 ON T1.country_id = T3.
country
WHERE T2.continent = 'Europe'
GROUP BY T1.country_name
HAVING COUNT(*) >= 3 ;
```

Extra Hard:

```
SELECT AVG(life_expectancy) FROM country
WHERE name NOT IN
   (SELECT T1.name
    FROM country AS T1 JOIN
    country_language AS T2
    ON T1.code = T2.country_code
    WHERE T2.language = "English"
      AND T2.is_official = "T") ;
```

In terms of the complexity of natural utterance, there is no qualitative measure of how hard the utterance is. Intuitively, models' performance can decrease when faced with longer questions from users. However, the information conveyed in longer sentences can be more complete, while there can be ambiguity in shorter sentences. Besides, there can be domain-specific phrases that confuse the model in both short and long utterances (Suhr et al., 2020). Thus, researchers need to consider various perspectives to determine the complexity of natural utterance.

C Text-to-SQL Datasets

Table 7 lists statistics for text-to-SQL datasets.

C.1 More Discussion on Text-to-SQL Datasets

CSpider (Min et al., 2019a), ViText2SQL (Tuan Nguyen et al., 2020) and José and Cozman (2021) translate all the English questions in Spider into Chinese, Vietnamese and Portuguese, respectively. TableQA (Sun et al., 2020) follows the data collection method from WikiSQL, while DuSQL (Wang et al., 2020c) follows Spider. Both TableQA and DuSQL collect Chinese utterance and SQL query pairs across different domains. Chen et al. (2021a) propose a Chinese domain-specific dataset, ESQL.

For multi-turn context-dependent text-to-SQL benchmarks, ATIS (Price, 1990; Dahl et al., 1994) includes user interactions with a SQL flight database in multiple turns. Sparc (Yu et al., 2019b) takes a further step to collect multi-turn interactions across 200 databases and 138 domains. However, both ATIS and Sparc assume all user questions can be mapped into SQL queries and do not include system responses. Later, inspired by task-oriented dialogue system (Budzianowski et al., 2018), Yu et al. (2019a) propose CoSQL. In CoSQL, the dialogue state is tracked by SQL. CoSQL includes three tasks of SQL-grounded dialogue state tracking to generate SQL queries from user's utterance, system response generation from query results, and user dialogue act prediction to detect and resolve ambiguous and unanswerable questions.

Besides, TriageSQL (Zhang et al., 2020) collects unanswerable questions other than natural utterance and SQL query pairs from Spider and WikiSQL, bringing up the challenge of distinguishing answerable questions from unanswerable ones in text-to-SQL systems.

D Encoding and Decoding Method

Table 8 and Table 9 show the encoding and decoding methods that have been discussed in § 3.2 and § 3.3, respectively.

E Other Related Tasks

Other tasks that are related to text-to-SQL include text-to-python (Bonthu et al., 2021), text-to-shell script/bash script (Bharadwaj and Shevade, 2022), text-to-regex (Ye et al., 2020), text-to-SPARQL (Ochieng, 2020), etc. They all take natural language queries as input and output different logical forms. Among these tasks, text-to-SPARQL is closest to text-to-SQL as both SPARQL and SQL can execute on database systems. Therefore, some

Datasets	#Size	#DB	#D	#T/DB	Issues addressed	Sources for data
Spider (Yu et al., 2018c)	10,181	200	138	5.1	Domain generalization	College courses, DabaseAnswers, WikiSQL
Spider-DK (Gan et al., 2021b)	535	10	-	4.8	Domain knowledge	Spider dev set
$\text{Spider}_{\text{Utran}}$ (Zeng et al., 2020)	15,023	200	138	5.1	Untranslatable questions	Spider + 5,330 untranslatable questions
Spider-L (Lei et al., 2020)	8,034	160	-	5.1	Schema linking	Spider train/dev
$\text{Spider}_{\text{SL}}$ (Taniguchi et al., 2021)	1,034	10	-	4.8	Schema linking	Spider dev set
Spider-Syn (Gan et al., 2021a)	8,034	160	-	5.1	Robustness	Spider train/dev
WikiSQL (Zhong et al., 2017)	80,654	26,521	-	1	Data size	Wikipedia
Squall (Shi et al., 2020b)	11,468	1,679	-	1	Lexicon-level supervision	WikiTableQuestions (Pasupat and Liang, 2015)
KaggleDBQA (Lee et al., 2021)	272	8	8	2.3	Domain generalization	Real web daabases
ATIS (Price, 1990; Dahl et al., 1994)	5,280	1	1	32	-	Flight-booking
GeoQuery (Zelle and Mooney, 1996)	877	1	1	6	-	US geography
Scholar (Iyer et al., 2017)	817	1	1	7	-	Academic publications
Academic (Li and Jagadish, 2014)	196	1	1	15	-	Microsoft Academic Search (MAS) database
IMDB (Yaghmazadeh et al., 2017)	131	1	1	16	-	Internet Movie Database
Yelp (Yaghmazadeh et al., 2017)	128	1	1	7	-	Yelp website
Advising (Finegan-Dollak et al., 2018)	3,898	1	1	10	-	University of Michigan course information
Restaurants (Tang and Mooney, 2000) (Popescu et al., 2003)	378	1	1	3	-	Restaurants
MIMICSQL (Wang et al., 2020d)	10,000	1	1	5	-	Healthcare domain
SEDE (Hazoom et al., 2021)	12,023	1	1	29	SQL template diversity	Stack Exchange

Table 7: Summarization for text-to-SQL datasets. #Size, #DB, #D, and #T/DB represent the number of question-SQL pairs, databases, domains, and tables per domain, respectively. We put "-" in the #D column because we do not know how many domains are in the Spider dev set and "-" in the Issues Addressed column because there is no specific issue addressed for the dataset. Datasets above and below the line are cross-domain and single-domain, respectively.

end-to-end models that take user queries as the input and output a sequence of logical forms can be applied to both tasks (Raffel et al., 2019). In contrast, methods (Xu et al., 2017) designed to take care of SQL natures cannot be directly applied to SPARQL, which requires carefully modification instead.

Methods	Adopted by	Applied datasets	Addressed challenges
Encode token type	TypeSQL (Yu et al., 2018a)	WikiSQL	Representing question meaning
Graph-based	GNN (Bogin et al., 2019a)	Spider	(1) Representing question and DB schemas in a structured way (2) Schema linking
	Global-GCN (Bogin et al., 2019b)	Spider	
	IGSQL (Cai and Wan, 2020)	Sparc, CoSQL	
	RAT-SQL (Wang et al., 2020a)	Spider	
	LEGSQL (Cao et al., 2021)	Spider	
	SADGA (Cai et al., 2021)	Spider	
	ShawdowGNN (Chen et al., 2021b)	Spider	
	S^2SQL (Hui et al., 2022)	Spider, Spider-Syn	
Self-attention	X-SQL (He et al., 2019)	WikiSQL	
	SQLova (Hwang et al., 2019)	WikiSQL	
	RAT-SQL (Wang et al., 2020a)	Spider	
	DuoRAT (Scholak et al., 2021a)	Spider	
	UnifiedSKG (Xie et al., 2022)	WikiSQL, Spider	
Adapt PLM	X-SQL (He et al., 2019)	WikiSQL	Leveraging external data to represent question and DB schemas
	SQLova (Hwang et al., 2019)	WikiSQL	
	Guo and Gao (2019)	WikiSQL	
	HydraNet (Lyu et al., 2020)	WikiSQL	
	Liu et al. (2021b), etc	Spider-L, SQUALL	
Pre-training	TaBERT (Yin et al., 2020)	Spider	
	GraPPA (Yu et al., 2021)	Spider	
	GAP (Shi et al., 2020a)	Spider	

Table 8: Methods used for encoding in text-to-SQL.

Methods		Adopted by	Applied datasets	Addressed challenges
Tree-based		Seq2Tree (Dong and Lapata, 2016)	-	Hierarchical decoding
		Seq2AST (Yin and Neubig, 2017)	-	
		SyntaxSQLNet (Yu et al., 2018b)	Spider	
Sketch-based		SQLNet (Xu et al., 2017)	WikiSQL	
		Dong and Lapata (2018)	WikiSQL	
		IRNet (Guo et al., 2019)	Spider	
		RYANSQL (Choi et al., 2021)	Spider	
Bottom-up		SmBop (Rubin and Berant, 2021)	Spider	
Attention Mechanism	Attention	Seq2Tree (Dong and Lapata, 2016)	-	Synthesizing information for decoding
		Seq2SQL (Zhong et al., 2017)	WikiSQL	
	Bi-attention Structured attention Relation-aware Self-attention	Guo and Gao (2018)	WikiSQL	
		Wang et al. (2019)	WikiSQL	
		DuoRAT (Scholak et al., 2021a)	Spider	
Copy Mechanism		Seq2AST (Yin and Neubig, 2017)	-	
		Seq2SQL (Zhong et al., 2017)	WikiSQL	
		Wang et al. (2018a)	WikiSQL	
		SeqGenSQL (Li et al., 2020a)	WikiSQL	
Intermediate Representation		IncSQL (Shi et al., 2018)	WikiSQL	Bridging the gap between natural language and SQL query
		IRNet (Guo et al., 2019)	Spider	
		Suhr et al. (2020)	Spider and others♠	
		Herzig et al. (2021)	GeoQuery, ATIS, Scholar	
		Gan et al. (2021c)	Spider	
		Brunner and Stockinger (2021)	Spider	
Others	Constrained decoding	UniSAr (Dou et al., 2022)	WikiSQL, Spide and others♡	Fine-grained decoding
		PICARD (Scholak et al., 2021b)	Spider, CoSQL	
	Execution-guided	SQLova (Hwang et al., 2019)	WikiSQL	
		Wang et al. (2018b)	WikiSQL	
	Discriminative re-ranking	Global-GCN (Bogin et al., 2019b)	Spider	SQL Ranking
		Kelkar et al. (2020)	Spider	
	Separate submodule	SQLNet (Xu et al., 2017)	WikiSQL	Easier decoding
		Guo and Gao (2018)	WikiSQL	
		Lee (2019)	Spider	
	BPE	Müller and Vlachos (2019)	Advising, ATIS, GeoQuery	
	Link gating	Chen et al. (2020b)	Spider	Synthesizing information for decoding

Table 9: Methods used for decoding in text-to-SQL. ♠: Academic, Advising, ATIS, GeoQuery, Yelp, IMDB, Scholar, Restaurants; ♡: TableQA DuSQL, CoSQL, Sparc, Chase.

An MRC Framework for Semantic Role Labeling

Nan Wang[1], **Jiwei Li**[2,3], **Yuxian Meng**[2]
Xiaofei Sun[2,3], **Han Qiu**[4], **Ziyao Wang**[5], **Guoyin Wang**[6] **and Jun He**[1*]
[1]Key Laboratory of Data Engineering and Knowledge Engineering of MOE,
School of Information, Renmin University of China
[2]Shannon.AI,[3]Zhejiang University, [4]Tsinghua University
[5]University of Leeds, [6]Amazon Alexa AI
{nanwang, hejun}@ruc.edu.cn, jiwei_li@shannonai.com

Abstract

Semantic Role Labeling (SRL) aims at recognizing the predicate-argument structure of a sentence and can be decomposed into two subtasks: predicate disambiguation and argument labeling. Prior work deals with these two tasks independently, which ignores the semantic connection between the two tasks. In this paper, we propose to use the machine reading comprehension (MRC) framework to bridge this gap. We formalize predicate disambiguation as multiple-choice machine reading comprehension, where the descriptions of candidate senses of a given predicate are used as options to select the correct sense. The chosen predicate sense is then used to determine the semantic roles for that predicate, and these semantic roles are used to construct the query for another MRC model for argument labeling. In this way, we are able to leverage both the predicate semantics and the semantic role semantics for argument labeling. We also propose to select a subset of all the possible semantic roles for computational efficiency. Experiments show that the proposed framework achieves state-of-the-art or comparable results to previous work. Code is available at https://github.com/ShannonAI/MRC-SRL.

1 Introduction

Semantic Role Labeling (SRL) aims at recognizing the predicate-argument structure of a sentence. The classic PropBank-style SRL includes two tasks: predicate disambiguation and argument labeling. Predicate disambiguation determines the specific meaning of a predicate in a given context and argument labeling identifies the arguments of the predicate and assign them with the corresponding semantic roles, where each argument is a text span in the sentence. PropBank defines two types of semantic roles for argument labeling: core roles and non-core roles (Bonial et al., 2010). Core roles are required roles that are in a close relation to the main verb in a sentence, such as *agent* and *patient*. There are seven core roles in PropBank: A0-A5 and AA. Non-core roles are modifiers, such as location (LOC) and time (TMP). The specific meanings of predicates and core roles are defined in the frame files. For example, for the sentence in Figure 1, the sense id of the predicate "beaten" is "beat.02", and its three arguments span are "The stock", "down" and "for two days", whose roles are respectively A1, A2, TMP. We can get the meaning of sense label "beat.02" and roles A1 and A2 from the frame file.

The stock has been beaten down for two days.
[A1] [beat.02][A2] [TMP]

sense id	beat.02	
sense	push, cause motion	
roles	A0	causer of motion
	A1	thing moving
	A2	direction, destination

Figure 1: An example of SRL. A0, A1 and A2 are *semantic roles* for the sense id "beat.02". The meanings of A0, A1 and A2 are respectively "causer of motion", "thing moving" and "direction, destination".

In traditional methods, predicate disambiguation and argument labeling are usually solved as two independent tasks. These works usually rely on feature-based methods (He et al., 2018b; Roth and Lapata, 2016; Che and Liu, 2010b)) for predicate disambiguation, and use span-based (Ouchi et al., 2018; He et al., 2018a; Li et al., 2019b) or BIO-based (He et al., 2017; Strubell et al., 2018; Shi and Lin, 2019) methods for argument labeling. These methods treat different predicate senses and argument roles as different class categories, and then solve them through classification. However, since these approaches ignore the semantic information of both predicate senses and argument roles, they are unable to establish the semantic connection be-

*Corresponding author.

Proceedings of the 29th International Conference on Computational Linguistics, pages 2188–2198
October 12–17, 2022.

Input Sentence
The stock has been < p> beaten </p> down for two days.

Multiple-Choice MRC for Predicate Disambiguation
Question: What is the sense of predicate "beaten"?
A. (Cause) pulsating motion that often makes sound
B. push, cause motion
C. win over some competitor
Answer: B

Extractive MRC for Argument Labeling
Question for A0: What are the arguments with meaning "causer of motion"?
Answer: No Answer
Question for A1: What are the arguments with meaning "thing moving"?
Answer: the stock
Question for A2: What are the arguments with meaning "direction, destination"?
Answer: down
Question for TMP: What are the time modifiers of predicate "beaten"?
Answer: for two days

Figure 2: An illustration of our MRC framework for Semantic Role Labeling. The meanings of predicate senses and argument roles are used for multiple-choice and extractive MRC, respectively.

tween the two tasks, i.e., argument roles are defined under predicate sense via the frame files. Some works (Cai et al., 2018; Conia and Navigli, 2020) jointly deal with these two tasks, but still cannot establish the semantic connection. We bridge this gap with an MRC framework, and we hope that the results from predicate disambiguation will contribute to argument labeling.

For PropBank-style semantic role labeling, although the specific meanings of predicate senses and argument roles are provided in the frame files, this information is seldom used due to its huge number and lack of effective ways to utilize it. Inspired by recent success in formulating non-MRC NLP tasks as MRC tasks (Levy et al., 2017; Li et al., 2020c), we propose an MRC framework for SRL, which can effectively utilize the semantic information provided by frame files. First, we transform the predicate disambiguation task into multiple-choice machine reading comprehension, where the descriptions of candidate predicate senses are used as options to select the correct sense. Then, we use the result of predicate disambiguation (i.e., the predicate sense) to determine the meaning of each core role with respect to the predicate. Lastly, we transform argument labeling into extractive machine reading comprehension, where the description of each semantic role is used to construct the query to extract the answer span within the input sentence, which serves as the argument we want. In addition, we also propose an additional module to select a subset of all possible semantic roles to improve computational efficiency. We provide an example (Figure 1) of the MRC framework in Figure 2.

We conduct experiments on CoNLL2005 (Carreras and Màrquez, 2005), CoNLL2009 (Hajič et al., 2009), and CoNLL2012 (Pradhan et al., 2013) benchmarks. Experimental results show that our model can achieve SOTA or comparable results to previous works on the three benchmarks.

2 Related Work

2.1 Semantic Role Labeling

Early semantic role labeling methods focused on feature engineering (Zhao et al., 2009; Pradhan et al., 2005). Recently, neural network based models have been studied and achieved promising performance. Collobert et al. (2011) proposed a unified neural network architecture and can avoid task-specific engineering. Zhou and Xu (2015) proposed to use BiLSTM as an end-to-end system for SRL. Tan et al. (2018) applied self-attention (Vaswani et al., 2017) mechanism to directly draw the global dependencies of the inputs. Shi and Lin (2019) presented a BERT (Devlin et al., 2019) based model for semantic role labeling. Jindal et al. (2020) propose a parameterized neighborhood memory adaptive method for SRL. Kalyanpur et al. (2020); Paolini et al. (2021); Blloshmi et al. (2021) cast SRL to a generative translation problem. Zhou et al. (2019); Marcheggiani and Titov (2020) incorporates syntactic information into SRL.

Some works also show that predicate disambiguation is helpful for argument labeling. Che et al. (2010) incorporated a word sense feature to improve the SRL performance. Che and Liu (2010a); Cai et al. (2018); Conia and Navigli (2020) jointly dealt with predicate disambiguation and argument labeling. These methods are different from ours and cannot use this semantic information of

the sense label and role label.

2.2 Machine Reading Comprehension

According to the type of the answer, machine reading comprehension can be divided into the following four categories: extractive (Rajpurkar et al., 2016), multiple-choice (Lai et al., 2017), close style (Onishi et al., 2016), and free-form (Nguyen et al., 2016). Related to our work are extractive and multiple-choice MRC. For extractive reading comprehension such as SQuAD (Rajpurkar et al., 2016), the answer is a span in the text, and the MRC model (Seo et al., 2017) gets the answer by predicting the probability that the word is start or end. Some datasets such as DROP (Dua et al., 2019) have answers that include multiple spans, and the answers can be obtained by using BIO tagging (Segal et al., 2019). For multiple-choice reading comprehension where the answer is one of several options, a method (Pan et al., 2019) is to calculate the score for each option and then select the option with the highest score.

2.3 Formalizing Non-MRC Tasks as MRC

Previous studies achieved great performance boost by applying the MRC framework to NER, dependency parsing and other non-MRC tasks. He et al. (2015) introduced the task of question-answer driven semantic role labeling without predefining an inventory of frames. Levy et al. (2017) showed that relation extraction can be reduced to answering simple reading comprehension questions. McCann et al. (2018) framed ten tasks as question answering. Li et al. (2020c) proposed to formulate named entity recognition as an MRC task. Other examples include joint entity relation extraction (Li et al., 2019a), coreference resolution (Wu et al., 2020), event extraction (Li et al., 2020a), entity linking (Gu et al., 2021), dependency parsing (Gan et al., 2021), text classification (Chai et al., 2020), etc.

Our approach to formalizing argument labeling as extractive MRC is similar to QA-SRL (He et al., 2015), but we focus on improving the performance of the model on Propbank-style SRL, while He et al. (2015) aims to provide a new SRL annotation paradigm, and He et al. (2015) neither uses the predicate sense definitions nor the argument role definitions provided in the frame files.

Algorithm 1 MRC framework for SRL

Input: sentence $x = \{x_1, ..., x_n\}$ with marked predicate p, frame files, annotation guidelines
Output: predicate sense $\hat{s}$, arguments $\boldsymbol{A}$

Get the lemma l of p using SpaCy
Get all the predicate senses $\boldsymbol{S_l}$ of l and the corresponding descriptions $\boldsymbol{D_l^s}$ from the frame files
for s_i in $\boldsymbol{S_l}$ **do**
    Get the description d_i^s of sense s_i from $\boldsymbol{D_l^s}$
    Concatenate d_i^s and x to get the input for RoBERTa
    Compute the score of s_i as the answer with Eq.(1)
end for
Select the highest scoring $\hat{s} \in \boldsymbol{S_l}$ as the predicate sense of p
Get the candidate argument roles $\boldsymbol{R_p}$ of p from the role prediction module
for r_i in $\boldsymbol{R_p}$ **do**
    if r_i is core role **then**
        Get the description d_i^r of role r_i from the frame files with $\hat{s}$
    else
        Get the description d_i^r of role r_i from the annotation guidelines
    end if
    Construct query q_i using d_i^r and p
    Concatenate q_i and x to get the input for RoBERTa
    Calculate the probability that each word in x belongs to the BIO tags
end for
Decode with non-overlap constraint to get the arguments $\boldsymbol{A}$ of p
return $\hat{s}$, $\boldsymbol{A}$

3 Method

3.1 Overview

An overview of our model is shown in Algorithm 1. Given a sentence $x = \{x_1, ..., x_n\}$ and the predicate p, the predicate disambiguation task is to determine the predicate sense $s \in \boldsymbol{S}$ of p, where $\boldsymbol{S}$ is the set of all predicate senses, and the argument labeling task is to find all the arguments $\boldsymbol{A} = \{a_1, ..., a_k\}$ of p, where $a_i \in \boldsymbol{A}$ is a text span in x, and assigning them the corresponding semantic roles.

Our framework consists of three modules: predicate disambiguation, role prediction, and argument

labeling, all of which use RoBERTa (Liu et al., 2019) as the backbone and use two special symbols < p> and </p> to mark the predicate p in the input sentence x. The predicate disambiguation module is intended to obtain the predicate sense of the predicate p. Note that we do not use the predicate sense for argument labeling directly, but only use it to get the meanings of the argument roles in the frame files. The role prediction module is to obtain the set of candidate roles for the predicate, and the main purpose of this module is to reduce the number of questions that need to be constructed when solving the argument labeling problem via an extractive MRC. The argument labeling module is used to obtain the arguments of the predicate, which is the core module in the whole framework.

3.2 Multiple-Choice MRC for Predicate Disambiguation

For the predicate disambiguation task, determining the sense label of the predicate involves two steps: identifying the lemma of the predicate, and determining the sense index of the predicate under this lemma. We use spaCy (Honnibal et al., 2020) to identify the lemma of the predicate. If the recognized lemma is not in the frame files, we use the lemma with the smallest edit distance of the predicate. After identifying the lemma, we can find all the senses defined under this lemma from the frame files, and then we choose the correct sense through multiple-choice reading comprehension.

Specifically, let $\boldsymbol{S_l}$ be all possible senses for the detected lemma. For each sense $s_i \in \boldsymbol{S_l}$, the corresponding sense description is d_i^s. We treat d_i^s as option, and the input for the RoBERTa is the concatenation of d_i^s and x. The confidence score of s_i as the correct sense is calculated by:

$$P(s_i = 1|d_i^s, x, p) = \text{sigmoid}(\text{FFN}_\text{p}(\boldsymbol{h^d})) \quad (1)$$

where $\boldsymbol{h^d}$ is the context representation of the first input token from RoBERTa and FFN_p is a single layer feedforward neural network. We train the model using the binary cross-entropy loss function. [1] During inference, we choose the sense with the highest probability score among all the sense options as the answer.

[1]We also tried to use softmax to get the probability of all senses, and then use the multi-class cross entropy loss for training, but we found the loss is unstable and hard to optimize.

3.3 Role Prediction

In semantic role labeling, most semantic roles do not have corresponding arguments given a specific input sentence. For example, in the CoNLL2005 dataset, there is a total number of 20 roles, but on average there are only 2.5 roles per predicate. Therefore, we use a role prediction module to avoid asking questions about impossible roles at the next argument labeling stage, reducing the amount of calculation required when using the MRC-based method.

Let $\boldsymbol{R}$ be the set of all semantic roles (in CoNLL 2005 the size of $\boldsymbol{R}$ is 20), the purpose of role prediction is to predict a set of possible roles $\boldsymbol{R_p} \subseteq \boldsymbol{R}$ for the predicate p. The input to RoBERTa is the sentence x with the marked predicate p. Let $\boldsymbol{h^r}$ be the context representation of the first token of the input sequence from RoBERTa, and $r_i \in \boldsymbol{R}$ is the i-th role of $\boldsymbol{R}$. We use the sigmoid function to calculate the probability that the predicate p has a role r_i:

$$P(r_i = 1|x, p) = \text{sigmoid}(\text{FFN}_{\text{r}_\text{i}}(\boldsymbol{h^r})) \quad (2)$$

where $\text{FFN}_{\text{r}_\text{i}}$ is a single layer feedforward neural network. We use the binary cross entropy loss function to train the model. During inference, we only keep up to λN roles with the highest probability score, where N is the number of predicates in the dataset. [2] Note that here we select the roles with the top λN probability scores on the whole dataset, not on the input sentence. And in the argument labeling module, we use the predicted roles from the role prediction module instead of the gold roles for training.

3.4 Extractive MRC for Argument Labeling

We formalize argument labeling as extractive reading comprehension, where the meaning of argument role is used to construct the query, and since the answer may contain multiple spans, we use BIO tagging to extract the arguments. [3] In ProbBank-style SRL, a role may be a norm role, a reference role, or a continuation role. A norm role is a standard role defined in the annotation guidelines, a

[2]An alternative strategy is to use a fixed threshold, which performs similarly to ours. But our strategy can directly get the number of argument roles, which helps to analyze the amount of computation needed in argument labeling.

[3]For dependency semantic role labeling, since pre-trained language models such as BERT split a word into multiple subwords, which is similar to span, BIO tagging is also applicable.

reference role is a reference to some other arguments, and a continuation role is a continuation phrase of a previously started argument. For example, role A1 may be N-A1 or R-A1, or C-A1. Since the subcategories of N/R/C do not contain semantic information, we do not encode such information into the query of the MRC model. We use BIO tagging to get the arguments of the predicate, and the set of BIO tags is

$$\boldsymbol{T} = \{B, I\} \times \{N, R, C\} \cup \{O\} \tag{3}$$

We use templates to construct the query of the MRC model. For core roles, our template is *"What are the X arguments of predicate Y with meaning Z?"*, where X is the role type, Y is the predicate, and Z is the description of role X in the frame files. For non-core roles, our template is *"What are the W modifiers of predicate Y?"*, where W is the specific meanings of non-core roles defined in the annotation guidelines.

Specifically, let q_i represent the query corresponding to the predicate p and the role $r_i \in \boldsymbol{R_p}$, the input of the MRC model is the concatenation of q_i and x. The context representation of x in the input $< q_i, x >$ pair is $\boldsymbol{h^{r_i}} = \{\boldsymbol{h_1^{r_i}}, ..., \boldsymbol{h_n^{r_i}}\}$, our goal is to predict $\boldsymbol{y}^{r_i} = \{y_1^{r_i}, ..., y_n^{r_i}\}$. Each $y_j^{r_i} \in y^{r_i}$ belongs to one tag in the tag set $\boldsymbol{T}$. For $y_j^{r_i}$, its probability distribution on BIO tag set is calculated by a softmax layer:

$$P(y_j^{r_i} = t|x, p, r_i) \propto \exp(\boldsymbol{W_t h_j^{r_i}} + \boldsymbol{b_t}) \tag{4}$$

where $t \in \boldsymbol{T}$ is a BIO tag, $\boldsymbol{W_t}$ and $\boldsymbol{b_t}$ are the corresponding parameters. We use multi-class cross entropy loss to train the model. And we use the method in section 3.5 to get the argument.

Note that at this stage, we use the predicate sense extracted at the predicate disambiguation stage to find the sense of each role selected at the role prediction stage. For example, suppose the predicate sense is "beat.02" and the semantic role is A0 as shown in Figure 1, we will immediately obtain the role's sense "causer of motion". In this way, the predicate sense can be leveraged for role sense detection, and thus further for semantic labeling, bridging the gap between the two tasks via a MRC framework.

3.5 Constrained Decoding

There are many global constraints in semantic role labeling (Punyakanok et al., 2008; Li et al., 2020b), such as all arguments of the predicate cannot overlap and each core role should appear at most once for each predicate. Our MRC approach does not directly model these constraints and can not guarantee that the obtained results satisfy these constraints. For simplicity, we only consider the non-overlap arguments constraint. The previous approach of using BIO tagging (He et al., 2017; Shi and Lin, 2019) to extract arguments can naturally model the non-overlap constraint, since each word in x can only belong to one of the BIO tags, there will be no overlapping words between the argument elements. But in our MRC-based BIO tagging method, since we have $\boldsymbol{R_p}$ roles, each word has at most $\boldsymbol{R_p}$ BIO tags. We implement the non-overlap constraint by mapping the local role-related BIO tag of each word into a global BIO tag set.

Specifically, for the sentence $x = \{x_1, ..., x_n\}$, the goal of constraint decoding is to obtain the corresponding tag sequence $y = \{y_1, ..., y_n\}$, where $y_j \in y$ belongs to the tag set $\boldsymbol{T_p}$:

$$\boldsymbol{T_p} = \boldsymbol{R_p} \times \{B, I\} \times \{N, R, C\} \cup \{O\} \tag{5}$$

For tag $t_p \in \boldsymbol{T_p}$, when it is a BI tag, it can be expressed as r_i-t, where $r_i \in \boldsymbol{R_p}$ and $t \in \boldsymbol{T}$. For BI tags, we add a role tag directly before the original BI tag. For example, the B-R tag of role A1 will be converted to A1-B-R, and then the score of the new tag is equal to the probability of the original tag:

$$\begin{aligned} s(y_j = t_p) &= s(y_j = r_i\text{-}t) \\ &= p(y_j^{r_i} = t) \end{aligned} \tag{6}$$

where $s(\cdot)$ is the score function. For O tags, we merge the O tags of different roles into one O tag, and the score of O tag after merging is the product of the O tag probabilities of all roles.

$$s(y_j = O) = \prod_{i=1}^{|\boldsymbol{R_p}|} p(y_j^{r_i} = O) \tag{7}$$

During inference, for each word x_i, its tag y_j is the highest scoring tag in the new BIO tag set $\boldsymbol{T_p}$.

$$y_j = \arg\max_{t_p \in T_p} s(y_j = t_p) \tag{8}$$

And we use the BIO tag sequence y to get all the arguments.

4 Experiments

4.1 Datasets

We conduct experiments on CoNLL2005 (Carreras and Màrquez, 2005), CoNLL2009 (Hajič

Model	Dev	WSJ	Brown
Shi and Zhang (2017)	-	93.4	82.4
Roth and Lapata (2016)	94.8	95.5	-
He et al. (2018b)	95.0	95.6	-
Shi and Lin (2019)+BERT	96.3	96.9	90.6
Ours+BERT	96.3	97.2	**91.9**
Ours+RoBERTa	**96.6**	**97.3**	91.3
Ours-semantics	96.2	96.7	89.9

Table 1: Predicate disambiguation results on CoNLL2009.

	CoNLL09 WSJ			CoNLL09 Brown		
Model	P	R	F1	P	R	F1
syntax-aware						
Cai and Lapata (2019)	91.7	90.8	91.2	83.2	81.9	82.5
Kasai et al. (2019)	90.3	90.0	90.2	81.0	80.5	80.8
Zhou et al. (2019)+BERT	91.2	91.2	91.2	85.7	86.1	85.9
Chen et al. (2022)+BERT	92.3	91.8	92.1	87.0	86.0	86.3
syntax-agnostic						
Li et al. (2019b)	89.6	91.2	90.4	81.7	81.4	81.5
Conia and Navigli (2020)+BERT	92.5	92.7	92.6	-	-	85.9
Shi and Lin (2019)+BERT	92.4	92.3	92.4	85.7	85.8	85.7
Jindal et al. (2020)+BERT	90.0	91.5	90.8	83.5	86.5	85.0
Ours+BERT	93.3	92.7	93.0	87.5	86.6	87.0
Ours+RoBERTa	**93.5**	**93.1**	**93.3**	**87.7**	**86.6**	**87.2**

Table 2: Argument labeling results on CoNLL2009.

et al., 2009) and CoNLL2012 (Pradhan et al., 2013) datasets. The CoNLL2005 and CoNLL2012 datasets are span-based SRL, where the arguments are constituents (spans) in the sentence, and the CoNLL2009 dataset is dependency-based SRL, where the arguments are syntactic heads. The CoNLL2005 dataset consists of sections of the Wall Street Journal part of the Penn TreeBank, where section 2-21 is used for training, section 24 is used for development, and section 23 is used for evaluation. In addition, it also includes three sections of the Brown corpus to test the robustness of the systems. The CoNLL2009 dataset uses the same corpus as CoNLL2005, but uses NomBank to extend the annotations. The CoNLL2012 dataset is extracted from the OntoNotes v5.0 corpus. The frame files are available as official resources in the three datasets and can be used by all systems.

4.2 Experiment Setup

For data preprocessing we follow (Li et al., 2019b). We use RoBERTa Large as the base encoder and we use two special symbols < p> and </p> to mark the predicate of the input sentence. We adopt Adam as optimizer, and the warmup rate is 0.05, the initial learning rate is 1e-5, the maximum number of epochs is 20, the number of tokens in each batch is 2048. λ is tuned on development set to ensure that the recall of the predicted roles is higher than 99%. All the experiments were conducted on a Tesla V100 GPU with 16GB memory.

During the parsing of the raw frame files, we found that there may be multiple sense and role definitions corresponding to one sense id, which may be caused by predicates with different part of speech or other reasons. For simplicity, we directly concatenate these different definitions, so that the final definition contains all possible cases and can be regarded as a more coarse-grained definition.

Predicate disambiguation is evaluated using accuracy, and argument labeling is evaluated using micro F1. The evaluation of argument labeling in CoNLL2009 also includes the results of predicate disambiguation, where the predicate sense is treated as a special kind of argument of a virtual root node.

4.3 Main Results

Predicate Disambiguation We evaluate the performances of predicate disambiguation on the CoNLL2009 dataset as previous work on the CoNLL2005 and CoNLL2012 datasets did not consider predicate disambiguation. The error of lemma recognition is also included in the final results. In Table 1, we report the experimental results of our method when using BERT and RoBERTa as encoders. The model using RoBERTa achieves the best results on the development set and on the in-domain test set (WSJ), and the model using BERT achieves the best results on the out-of-domain test set (Brown). The performances of BERT and RoBERTa on the development and brown test sets are opposite, which indicates that the evaluation on the development set does not fully reflect the model's generalization ability.

To investigate the impact of the sense description provided by the frame file, we also give the experimental results without using this semantic information in Table 1 ("-semantics"). In this setting, we also use RoBERTa, but the predicate sense description is replaced by the corresponding numeric label (e.g., "02" in "beat.02"). The experimental results show that the model performs worse when this semantic information is not available, especially in the out-of-domain Brown test set, where the accuracy decreases by 1.4%.

Argument Labeling Table 2 shows the results for dependency SRL, and Table 3 shows the ex-

Model	CoNLL05 WSJ			CoNLL05 Brown			CoNLL12 Test		
	P	R	F1	P	R	F1	P	R	F1
syntax-aware									
Zhou et al. (2019)+BERT	89.0	88.8	88.9	81.9	81.0	81.4	-	-	-
Mohammadshahi and Henderson (2021)+BERT	89.1	88.7	88.9	83.9	82.5	83.2	-	-	-
Xia et al. (2020)+RoBERTa	88.4	88.8	88.6	83.1	83.3	83.2	-	-	-
Marcheggiani and Titov (2020)+RoBERTa	87.7	88.1	88.0	80.5	80.7	80.6	86.5	87.1	86.8
syntax-agnostic									
Li et al. (2019b)	87.9	87.5	87.7	80.6	80.4	80.5	85.7	86.3	86.0
Conia and Navigli (2020)+BERT	-	-	-	-	-	-	86.9	87.7	87.3
Blloshmi et al. (2021)+BART	-	-	-	-	-	-	87.8	86.8	87.3
Shi and Lin (2019)+BERT	88.6	89.0	88.8	81.9	82.1	82.0	85.9	87.0	86.5
Jindal et al. (2020)+BERT	88.7	88.0	87.9	80.3	80.1	80.2	86.3	86.8	86.6
Paolini et al. (2021)+T5	-	-	89.3	-	-	82.0	-	-	87.7
Zhang et al. (2021)+RoBERTa	89.6	89.7	89.6	83.8	83.6	83.7	88.1	**88.6**	**88.3**
Ours+BERT	89.7	89.0	89.3	85.9	83.5	84.7	88.0	87.7	87.8
Ours+RoBERTa	**90.4**	**89.7**	**90.0**	**86.4**	**83.8**	**85.1**	**88.6**	87.9	88.3

Table 3: Argument labeling results on CoNLL2005 and CoNLL2012.

perimental results for span SRL. Since our method is syntax-agnostic, we first compare it with the syntax-agnostic methods. Compared with previous methods, our improvement on the in-domain WSJ test sets of CoNLL2005 and CoNLL2009 is 0.4 and 0.7, respectively, on the out-of-domain Brown test sets is 1.4 and 1.3, respectively, and we achieve comparable results on the CoNLL2012 test set. The out-of-domain Brown test set is used to test the robustness of the presented systems, and our method achieves greater improvement on this test set , which indicates that our approach is more robust than previous syntax-agnostic approaches because of the use of role semantics. The syntax-aware method (Mohammadshahi and Henderson, 2021) also performs better on the Brown test set compared to the syntax-agnostic methods (Shi and Lin, 2019), a similar phenomenon to ours. However, unlike the syntax-aware approach, our approach is syntax-agnostic and utilizes the semantic information provided in the frame files rather than the syntactic information of the sentence, and outperforms syntax-aware methods. This observation demonstrates that leveraging semantic information in frame files provides stronger robustness than syntax-aware methods for SRL.

5 Ablation studies

5.1 Effect of Predicate Disambiguation

Our framework uses a pipelined approach to connect the predicate disambiguation and the argument labeling task, so different predicate disambiguation accuracies may affect the results of argument labeling. Here we analyze the performance of the same argument labeling model with different predicate disambiguation accuracies. We obtain the results of different predicate disambiguation accuracies through randomly replacing part of the gold predicate senses with other predicate senses. Then we use the ordinarily trained argument labeling model to make predictions under different predicate disambiguation results. Figure 3 shows that the model performs monotonically worse as the predicate disambiguation accuracy decreases, so an accurate predicate disambiguation model is required to achieve improved semantic role labeling results

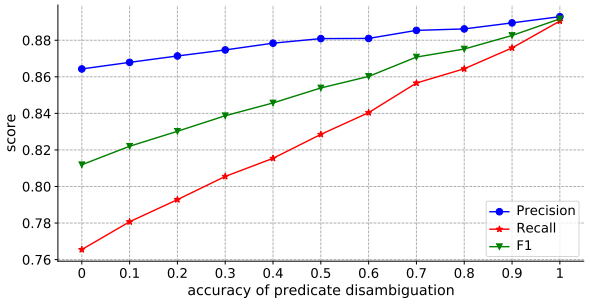

Figure 3: Experimental results on CoNLL 2005 development set with different predicate disambiguation accuracies, we use the argument labeling model trained in the main results.

Recall	0.90	0.93	0.96	0.99
F1	87.3	88.4	88.2	88.6

Table 4: Experimental results of different role prediction recall scores on CoNLL2005 development set.

5.2 Effect of Argument Role Semantics

We also study the performance of our MRC framework in the case where the query does not contain any semantics, and in this case, the query is replaced with a category label. We use RoBERTa-Base for our experiments. When role semantics is not considered, the F1 scores on the development set and the out-of-domain Brown test set of CoNLL2005 are 88.2 and 83.2, respectively. when role semantics is considered, the F1 scores on the development set and the out-of-domain Brown test set of CoNLL2005 are 88.5 and 83.8, respectively. The experimental results show that taking semantics into account performs better than not taking semantics into account, especially when the domains of the training and test sets are different. And this proves that the semantics of the argument roles is useful in our framework.

5.3 Effect of Role Prediction

Since role prediction is an upstream task of argument labeling, missing potential argument roles in the role prediction stage can lead to the error propagation problem. We mitigate this problem by ensuring that the recall of role prediction is higher than 99% and training the argument labeling model under the predicted roles. We use RoBERTa-Base to train the model under different role recalls. Table 4 shows the influence of different role prediction recall scores on argument labeling. It can be seen that when the recall is low, the F1 score of argument labeling will decrease significantly – 87.3 when recall is 0.90 versus 88.6 when recall is 0.99.

5.4 Low-Resource Scenarios

Our argument labeling model uses role semantic information, and this prior information may be helpful for model learning when the dataset is small, so we study the effect of different training data sizes. The baseline we compare is a naive BIO tagging model based on RoBERTa, since our MRC model can be seen as a simple BIO tagging model incorporating role semantic information. We use RoBERTa-Base, train on different percentages of CoNLL2005 training set, and then evaluate on CoNLL2005 development set. From Figure 4, we can see that the MRC BIO model performs better in low-resource scenarios than the naive BIO model that does not use role semantics.

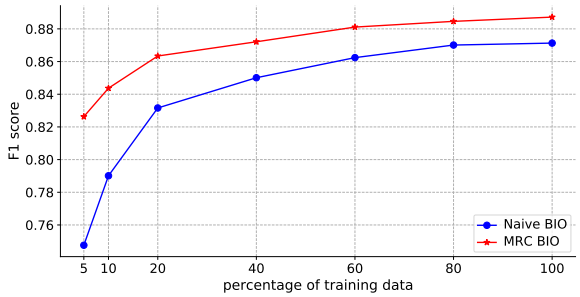

Figure 4: F1 score under different percentages of data on CoNLL2005 development set.

5.5 Computational Overhead

Our MRC framework performs better in robustness and low-resource scenarios due to the use of label semantic information provided in frame files, but to utilize this information, we need to encode all <label, sentence> pairs using a pre-trained model, which can be computationally intensive if the number of labels is large, we mitigate this problem by filtering impossible labels. [4] In predicate disambiguation, we use lemma to filter impossible predicate senses, and in argument labeling, we use an additional role prediction module to filter impossible roles.

Since the main computation in our framework is spent on the argument labeling module, here we give a rough analysis of the computational overhead it requires. In section 3.3, we select the λN roles with the highest probability scores in the dataset, which are used in the argument labeling module to construct queries, so λN reflects the amount of computation we need in the argument labeling module. When $\lambda = R$, this approach is equivalent to asking questions directly to all roles. In CoNLL2005, CoNLL2009, and CoNLL2012, the total number of semantic roles are 20, 20, 28, respectively, and the actual λs in the role prediction module are 5, 4.2, 5.5, respectively, which indicates that our model achieves approximately 4x, 4.8x and 5.1x speedups in CoNLL2005, CoNLL2009, and CoNLL2012 compared to asking questions directly to all roles.

[4] We also tried to decouple the label and sentence encoding to avoid encoding the same sentence multiple times, but it did not perform as well as the simple filtering strategy.

6 Conclusion

In this paper, we propose an MRC-based framework for semantic role labeling. We formalize predicate disambiguation as multiple-choice reading comprehension and argument labeling as extractive reading comprehension. Besides, we also propose a role prediction module to reduce the computation caused by considering all roles in the dataset for argument labeling. Experiments show that our framework can effectively utilize the semantic information provided in frame files and achieve competitive results.

Acknowledgement

We would like to thank anonymous reviewers for their comments and suggestions. This work is supported by the Key R & D Projects of the Ministry of Science and Technology (2020YFB2104100, 2020YFC0832500), National Natural Science Foundation of China (62172421, 62072459).

References

Rexhina Blloshmi, Simone Conia, Rocco Tripodi, and Roberto Navigli. 2021. Generating senses and roles: An end-to-end model for dependency- and span-based semantic role labeling. In *IJCAI*.

Claire Bonial, Olga Babko-Malaya, Jinho D Choi, Jena Hwang, and Martha Palmer. 2010. Propbank annotation guidelines. *Center for Computational Language and Education Research Institute of Cognitive Science University of Colorado at Boulder*.

Jiaxun Cai, Shexia He, Zuchao Li, and Hai Zhao. 2018. A full end-to-end semantic role labeler, syntactic-agnostic over syntactic-aware? In *Proceedings of the 27th International Conference on Computational Linguistics*, pages 2753–2765.

Rui Cai and Mirella Lapata. 2019. Semi-supervised semantic role labeling with cross-view training. In *EMNLP*.

Xavier Carreras and Lluís Màrquez. 2005. Introduction to the CoNLL-2005 shared task: Semantic role labeling. In *Proceedings of the ninth conference on computational natural language learning (CoNLL-2005)*, pages 152–164.

Duo Chai, Wei Wu, Qinghong Han, Fei Wu, and Jiwei Li. 2020. Description based text classification with reinforcement learning. In *International Conference on Machine Learning*, pages 1371–1382. PMLR.

Wanxiang Che and Ting Liu. 2010a. Jointly modeling WSD and SRL with Markov Logic. In *Proceedings of the 23rd International Conference on Computational Linguistics (Coling 2010)*, pages 161–169, Beijing, China. Coling 2010 Organizing Committee.

Wanxiang Che and Ting Liu. 2010b. Using word sense disambiguation for semantic role labeling. In *2010 4th International Universal Communication Symposium*, pages 167–174. IEEE.

Wanxiang Che, Ting Liu, and Yongqiang Li. 2010. Improving semantic role labeling with word sense. In *Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, pages 246–249.

Junjie Chen, Xiangheng He, and Yusuke Miyao. 2022. Modeling syntactic-semantic dependency correlations in semantic role labeling using mixture models. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 7959–7969, Dublin, Ireland. Association for Computational Linguistics.

Ronan Collobert, Jason Weston, Léon Bottou, Michael Karlen, Koray Kavukcuoglu, and Pavel Kuksa. 2011. Natural language processing (almost) from scratch. *Journal of machine learning research*, 12(ARTICLE):2493–2537.

Simone Conia and R. Navigli. 2020. Bridging the gap in multilingual semantic role labeling: a language-agnostic approach. In *COLING*.

J. Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. Bert: Pre-training of deep bidirectional transformers for language understanding. In *NAACL-HLT*.

Dheeru Dua, Yizhong Wang, Pradeep Dasigi, Gabriel Stanovsky, Sameer Singh, and Matt Gardner. 2019. Drop: A reading comprehension benchmark requiring discrete reasoning over paragraphs. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 2368–2378.

Leilei Gan, Yuxian Meng, Kun Kuang, Xiaofei Sun, Chun Fan, Fei Wu, and Jiwei Li. 2021. Dependency parsing as mrc-based span-span prediction. *arXiv preprint arXiv:2105.07654*.

Yingjie Gu, Xiaoye Qu, Z. Wang, Baoxing Huai, Nicholas Jing Yuan, and Xiaolin Gui. 2021. Read, retrospect, select: An mrc framework to short text entity linking. *ArXiv*, abs/2101.02394.

Jan Hajič, Massimiliano Ciaramita, Richard Johansson, Daisuke Kawahara, Maria Antònia Martí, Lluís Màrquez, Adam Meyers, Joakim Nivre, Sebastian Padó, Jan Štěpánek, Pavel Straňák, Mihai Surdeanu, Nianwen Xue, and Yi Zhang. 2009. The CoNLL-2009 shared task: Syntactic and semantic dependencies in multiple languages. In *Proceedings of*

the Thirteenth Conference on Computational Natural Language Learning (CoNLL 2009): Shared Task, pages 1–18, Boulder, Colorado. Association for Computational Linguistics.

Luheng He, Kenton Lee, Omer Levy, and Luke Zettlemoyer. 2018a. Jointly predicting predicates and arguments in neural semantic role labeling. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 364–369.

Luheng He, Kenton Lee, Mike Lewis, and Luke Zettlemoyer. 2017. Deep semantic role labeling: What works and what's next. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 473–483.

Luheng He, Mike Lewis, and Luke Zettlemoyer. 2015. Question-answer driven semantic role labeling: Using natural language to annotate natural language. In *Proceedings of the 2015 conference on empirical methods in natural language processing*, pages 643–653.

Shexia He, Zuchao Li, Hai Zhao, and Hongxiao Bai. 2018b. Syntax for semantic role labeling, to be, or not to be. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 2061–2071, Melbourne, Australia. Association for Computational Linguistics.

Matthew Honnibal, Ines Montani, Sofie Van Landeghem, and Adriane Boyd. 2020. spaCy: Industrial-strength Natural Language Processing in Python.

Ishan Jindal, Ranit Aharonov, Siddhartha Brahma, Huaiyu Zhu, and Yunyao Li. 2020. Improved semantic role labeling using parameterized neighborhood memory adaptation. *arXiv preprint arXiv:2011.14459*.

Aditya Kalyanpur, Or Biran, Tom Breloff, Jennifer Chu-Carroll, Ariel Diertani, Owen Rambow, and Mark Sammons. 2020. Open-domain frame semantic parsing using transformers. *arXiv preprint arXiv:2010.10998*.

Jungo Kasai, Dan Friedman, R. Frank, Dragomir R. Radev, and Owen Rambow. 2019. Syntax-aware neural semantic role labeling with supertags. In *NAACL*.

Guokun Lai, Qizhe Xie, Hanxiao Liu, Yiming Yang, and Eduard Hovy. 2017. Race: Large-scale reading comprehension dataset from examinations. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 785–794.

Omer Levy, Minjoon Seo, Eunsol Choi, and Luke Zettlemoyer. 2017. Zero-shot relation extraction via reading comprehension. In *Proceedings of the 21st Conference on Computational Natural Language Learning (CoNLL 2017)*, pages 333–342.

Fayuan Li, Wei-Hua Peng, Y. Chen, Q. Wang, Lu Pan, Yajuan Lyu, and Yong Zhu. 2020a. Event extraction as multi-turn question answering. In *EMNLP*.

Tao Li, Parth Anand Jawale, Martha Palmer, and Vivek Srikumar. 2020b. Structured tuning for semantic role labeling. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 8402–8412.

Xiaoya Li, J. Feng, Yuxian Meng, Qinghong Han, F. Wu, and J. Li. 2020c. A unified mrc framework for named entity recognition. *ArXiv*, abs/1910.11476.

Xiaoya Li, Fan Yin, Zijun Sun, Xiayu Li, Arianna Yuan, Duo Chai, Mingxin Zhou, and J. Li. 2019a. Entity-relation extraction as multi-turn question answering. In *ACL*.

Zuchao Li, Shexia He, Hai Zhao, Yiqing Zhang, Zhuosheng Zhang, Xi Zhou, and Xiang Zhou. 2019b. Dependency or span, end-to-end uniform semantic role labeling. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, pages 6730–6737.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Roberta: A robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692*.

Diego Marcheggiani and Ivan Titov. 2020. Graph convolutions over constituent trees for syntax-aware semantic role labeling. In *EMNLP*.

B. McCann, N. Keskar, Caiming Xiong, and R. Socher. 2018. The natural language decathlon: Multitask learning as question answering. *ArXiv*, abs/1806.08730.

Alireza Mohammadshahi and J. Henderson. 2021. Syntax-aware graph-to-graph transformer for semantic role labelling. *ArXiv*, abs/2104.07704.

Tri Nguyen, Mir Rosenberg, Xia Song, Jianfeng Gao, Saurabh Tiwary, Rangan Majumder, and Li Deng. 2016. Ms marco: A human generated machine reading comprehension dataset. In *CoCo@ NIPS*.

Takashi Onishi, Hai Wang, Mohit Bansal, Kevin Gimpel, and David McAllester. 2016. Who did what: A large-scale person-centered cloze dataset. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 2230–2235.

Hiroki Ouchi, Hiroyuki Shindo, and Yuji Matsumoto. 2018. A span selection model for semantic role labeling. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 1630–1642, Brussels, Belgium. Association for Computational Linguistics.

Xiaoman Pan, Kai Sun, Dian Yu, Jianshu Chen, Heng Ji, Claire Cardie, and Dong Yu. 2019. Improving question answering with external knowledge. *arXiv preprint arXiv:1902.00993*.

Giovanni Paolini, Ben Athiwaratkun, Jason Krone, Jie Ma, Alessandro Achille, Rishita Anubhai, Cicero Nogueira dos Santos, Bing Xiang, and Stefano Soatto. 2021. Structured prediction as translation between augmented natural languages. In *9th International Conference on Learning Representations, ICLR 2021*.

Sameer Pradhan, Alessandro Moschitti, Nianwen Xue, Hwee Tou Ng, Anders Björkelund, Olga Uryupina, Yuchen Zhang, and Zhi Zhong. 2013. Towards robust linguistic analysis using ontonotes. In *Proceedings of the Seventeenth Conference on Computational Natural Language Learning*, pages 143–152.

Sameer Pradhan, Wayne Ward, Kadri Hacioglu, James H Martin, and Dan Jurafsky. 2005. Semantic role labeling using different syntactic views. In *Proceedings of the 43rd Annual Meeting of the Association for Computational Linguistics (ACL'05)*, pages 581–588.

Vasin Punyakanok, Dan Roth, and Wen-tau Yih. 2008. The importance of syntactic parsing and inference in semantic role labeling. *Computational Linguistics*, 34(2):257–287.

Pranav Rajpurkar, Jian Zhang, Konstantin Lopyrev, and Percy Liang. 2016. Squad: 100,000+ questions for machine comprehension of text. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 2383–2392.

Michael Roth and Mirella Lapata. 2016. Neural semantic role labeling with dependency path embeddings. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1192–1202, Berlin, Germany. Association for Computational Linguistics.

Elad Segal, Avia Efrat, Mor Shoham, Amir Globerson, and Jonathan Berant. 2019. A simple and effective model for answering multi-span questions. *arXiv preprint arXiv:1909.13375*.

Minjoon Seo, Aniruddha Kembhavi, Ali Farhadi, and Hannaneh Hajishirzi. 2017. Bidirectional attention flow for machine comprehension. *ArXiv*, abs/1611.01603.

Peng Shi and Jimmy Lin. 2019. Simple bert models for relation extraction and semantic role labeling. *arXiv preprint arXiv:1904.05255*.

Peng Shi and Yue Zhang. 2017. Joint bi-affine parsing and semantic role labeling. In *2017 International Conference on Asian Language Processing (IALP)*, pages 338–341. IEEE.

Emma Strubell, Patrick Verga, Daniel Andor, David Weiss, and Andrew McCallum. 2018. Linguistically-informed self-attention for semantic role labeling. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 5027–5038, Brussels, Belgium. Association for Computational Linguistics.

Zhixing Tan, Mingxuan Wang, J. Xie, Y. Chen, and X. Shi. 2018. Deep semantic role labeling with self-attention. In *AAAI*.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, L. Kaiser, and Illia Polosukhin. 2017. Attention is all you need. *ArXiv*, abs/1706.03762.

W. Wu, F. Wang, Arianna Yuan, F. Wu, and J. Li. 2020. Corefqa: Coreference resolution as query-based span prediction. In *ACL*.

Qingrong Xia, Rui Wang, Zhenghua Li, Y. Zhang, and Min Zhang. 2020. Semantic role labeling with heterogeneous syntactic knowledge. In *COLING*.

Yu Zhang, Qingrong Xia, Shilin Zhou, Yong Jiang, Zhenghua Li, Guohong Fu, and Min Zhang. 2021. Semantic role labeling as dependency parsing: Exploring latent tree structures inside arguments. *arXiv preprint arXiv:2110.06865*.

Hai Zhao, Wenliang Chen, Kiyotaka Uchimoto, Kentaro Torisawa, et al. 2009. Multilingual dependency learning: Exploiting rich features for tagging syntactic and semantic dependencies. In *Proceedings of the Thirteenth Conference on Computational Natural Language Learning (CoNLL 2009): Shared Task*, pages 61–66.

Jie Zhou and W. Xu. 2015. End-to-end learning of semantic role labeling using recurrent neural networks. In *ACL*.

Junru Zhou, Zuchao Li, and Hai Zhao. 2019. Parsing all: Syntax and semantics, dependencies and spans. *arXiv preprint arXiv:1908.11522*.

PCBERT: Parent and Child BERT for Chinese Few-shot NER

Peichao Lai Feiyang Ye Lin Zhang Zhiwei Chen
Yanggeng Fu Yingjie Wu Yilei Wang
College of Computer and Data Science, Fuzhou University, China
yilei@fzu.edu.cn

Abstract

Achieving good performance on few-shot or zero-shot datasets has been a long-term challenge for NER. The conventional semantic transfer approaches on NER will decrease model performance when the semantic distribution is quite different, especially in Chinese few-shot NER. Recently, prompt-tuning has been thoroughly considered for low-resource tasks. But there is no effective prompt-tuning approach for Chinese few-shot NER. In this work, we propose a prompt-based Parent and Child BERT (PCBERT) for Chinese few-shot NER. To train an annotating model on high-resource datasets and then discover more implicit labels on low-resource datasets. We further design a label extension strategy to achieve label transferring from high-resource datasets. We evaluated our model on Weibo and the other three sampling Chinese NER datasets, and the experimental result demonstrates our approach's effectiveness in few-shot learning.

1 Introduction

NER is a fine-grained sequence labeling task, a slight change in each token will significantly impact the model results. A big challenge of NER is to enhance the performance in low-resource scenarios. There are some prior works (Yang et al., 2017; Lee et al., 2017; Abhishek et al., 2017) that demonstrate that transfer learning can improve the model performance. However, they all rely on similar semantic distribution between source and target datasets, and both datasets should contain rich annotated data. A significant difficulty of few-shot or zero-shot NER is the lack of annotated labels in practical application. Another challenge of Chinese NER is the implicit word boundary, which makes it difficult for the model to distinguish the entity boundary. The lexicon-based approach is a standard solution to solve the above issue. But the performance of traditional lexicon-based models in Chinese few-shot NER is still unsatisfactory.

Recently, prompt-tuning (Lester et al., 2021) on the pre-trained language models (PLMs) has been thoroughly considered for low-resource scenarios because the prompt-tuning process is highly consistent with the target task. Previous work (Cui et al., 2021; Ma et al., 2022; Chen et al., 2021) has demonstrated that prompt-tuning can more effectively enhance the model performance on few-shot NER compared with fine-tuning. However, when the semantic distribution is quite different, using prompt-tuning for semantic transfer learning will decrease model performance, which implies the semantic transfer is unsuitable for NER in the above situation. Besides, the implicit boundaries of Chinese words make the size of the prompt template uncertain and require a higher ability to judge its boundary. Moreover, using inappropriate prompt construction engineering on Chinese few-shot NER datasets can not improve model performance effectively but increases training time.

In this work, we introduce an enhanced lexicon feature and a prompt-based label transfer approach to address the above issues. We leverage the lexicon feature to enhance Chinese word boundary distinction ability in few-shot NER datasets. We further design a label extension strategy to achieve label transferring from high-resource datasets. We propose a Parent and Child BERT(PCBERT) model powered by a label lexicon adapter and a prompt-tuning component to integrate the lexicon feature and the implicit label feature. And it is worth noting that our implementation with a transformer encoder is more efficient than some decoding template-based approaches. We evaluated our model on Weibo(Peng and Dredze, 2015) and the other three samplings of Chinese NER datasets, and the experimental result demonstrates our approach's effectiveness in few-shot learning. Our model outperforms other related work in all experiments and achieves state-of-the-art F1 scores on Weibo.

Proceedings of the 29th International Conference on Computational Linguistics, pages 2199–2209
October 12–17, 2022.

The contributions of this work can be summarized as follows:

1. We introduce a label extension strategy to implement the label transfer learning in few-shot NER, which can effectively enhance the model performance.

2. We propose a new PCBERT model consisting of a P-BERT component and a C-BERT component to integrate the lexicon feature and the implicit label feature.

3. Experimental results verify that our approaches are suitable for Chinese few-shot NER transfer learning and achieve excellent performance on few-shot learning.

2 Preliminaries

2.1 Problems of Few-shot NER

In the few-shot NER tasks, given the high-resource source domain dataset $S = \{P_S, L_S\}$, where the $P_S = \left\{\left(X_S^1, Y_S^1\right), \ldots, \left(X_S^R, Y_S^R\right)\right\}$ is the set of input text and corresponding labels, and $L_S = \{l_1, \ldots, l_m\}$ is the set of entity label categories with size m. Then given the low-resource target domain dataset $T = \{P_T, L_T\}$, the task aims to enhance the model performance in the target domain dataset by utilizing the resources of the source domain dataset. However, the traditional NER transfer learning approaches face two main challenges: the semantic distribution difference between the source and target domains; the same category labels have different definitions in different datasets.

2.2 Label Extension Strategy

Formally, we denote $\mathcal{D}_P(X)$ to represent the semantic space of input text X, and $\mathcal{D}_\mathcal{E}(L)$ represent the semantic distribution that contains label $l \in L$. The correlation between model performance p and the semantic distribution can be explained as:

$$p \propto \frac{\mathcal{D}_P(X_S) \cap \mathcal{D}_P(X_T)}{\mathcal{D}_P(X_S) - \mathcal{D}_P(X_T)} \quad (1)$$

$$p \propto \frac{\mathcal{D}_\mathcal{E}(L_S) \cap \mathcal{D}_\mathcal{E}(L_T)}{\mathcal{D}_\mathcal{E}(L_S) - \mathcal{D}_\mathcal{E}(L_T)} \quad (2)$$

when the semantic space gap between the source domain and the target domain is large, the semantic intersection of S and T is quite limited compared with the semantic difference between S and T (i.e., $\mathcal{D}_P(X_S) \cap \mathcal{D}_P(X_T) \ll \mathcal{D}_P(X_S) - \mathcal{D}_P(X_T)$). The semantic deviation makes the pre-trained model more difficult to fine-tune than the uniform distribution model and even decreases performance in the target domain. Therefore, it is tough to carry out cross-domain semantic migration on few-shot NER datasets.

In this work, we use label extension to enrich the label features in T. As shown in Equation 2, $\mathcal{D}_\mathcal{E}(L_S) \cap \mathcal{D}_\mathcal{E}(L_T)$ represents the semantic distribution range that implicitly contains the intersection of L_S and L_T. It may include entity labels from S and does not exist in T, making label extension reasonable as T is a low-resource dataset with fewer labels. The label extension can be implemented with an annotation model with fully supervised training on S and annotating on T. However, some issues may impact the label extension accuracy. One is the annotation model performance; another is that the same category labels may explain the different meanings between S and T. These issues can be treated as label noise that affects the target task performance. To address the above issues, we adopt a prompt-based approach with a label fusion layer in our proposed model to reduce the influence of label noise.

3 Method

In this paper, we propose a two-stage model named PCBERT for Chinese few-shot NER, which consists of the Parent and the Child component. Both components are implemented with BERT (Devlin et al., 2019), and we defined them as P-BERT and C-BERT, respectively. The overall model structure of PCBERT is illustrated in Figure 1. The P-BERT is a prompt-based model to extract the implicit label extension features in the target dataset; the C-BERT is a lexicon-based model inspired by the LEBERT (Liu et al., 2021a) and further incorporates multi-label features of each lexicon. In the first stage, the P-BERT pre-trains on the label extension dataset. Then the P-BERT is set to be frozen in the second stage, providing label extension features to fine-tune the C-BERT. The structure and functionality are described in the following.

3.1 P-BERT

The primary function of P-BERT is prompt-tuning on the label extension dataset and providing prompt features for C-BERT. The label extension dataset is constructed by the method mentioned in Section 2.2. The inspiration for prompt-tuning comes from models like GPT-3 (Brown et al., 2020), and T5 (Raffel et al., 2020), which transform the tar-

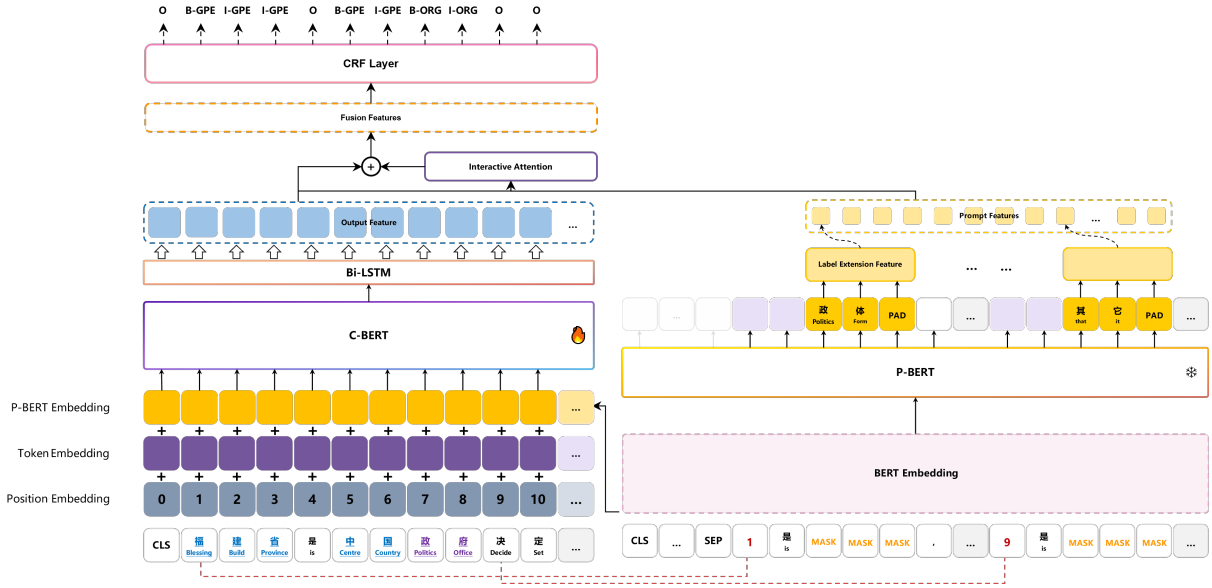

Figure 1: The overall structure of PCBERT. The P-BERT is trained on the label extension dataset in the prompt-tuning stage and provides label extension features for the C-BERT in the fine-tuning stage. While fine-tuning, the P-BERT is set to be frozen.

get task into text-to-text form and directly model text using PLMs. In this work, our prompt-tuning approach is designed toward the target task, consisting of a template function $TP(X, Y)$ that converts the raw input to prompt input. The label in the template input is a textual string instead of an entity category index, which helps leverage implicit knowledge from PLMs and reduces the influence of label noise in the label extension dataset.

We use vanilla BERT as P-BERT, each input $X = \{x_1, \ldots, x_n\}$ in the label extension dataset is converted into prompt input X_{prompt} with the $TP(X, Y)$. The prompt input consists of the following parts:

$$X_{prompt} = [\text{CLS}]\, X\, [\text{SEP}]\, TP\,(X, Y) \qquad (3)$$

where the first part of X_{prompt} is the origin input X, and the second part is label templates computed by the $TP(X, Y)$. [CLS] and [SEP] are the special token of BERT. Each label template follows the form as "$[Index]is[Z]$", where the index slot $[Index]$ indicates each token position in X, and the label slot $[Z]$ is the Chinese word that represents the label Y. Each label template is concatenated with a comma. Then, the label slot is padded to the same size by the tokenizer to adapt parallel training better and locate the output features. During prompt-tuning, the label slot of each input will be masked with the [MASK] token, and its task goal is to restore the masked label tokens. Then the loss function can be defined with the cross-entropy loss:

$$\mathcal{L}_{prompt} = -\sum_{i} z_i \log\left(p\left(\hat{\mathbf{z}}_\mathbf{i} \mid X\right)\right) \qquad (4)$$

where $z_i \in Z$ and $\hat{\mathbf{z}}_\mathbf{i}$ is the corresponding predicted token.

3.2 C-BERT

Chinese NER tasks are more challenging because the word boundary of sentences is not explicit. Many works (Sui et al., 2019; Li et al., 2020; Zhang and Yang, 2018) have demonstrated that leveraging lexicon information can effectively enhance the model performance. In few-shot NER, the lexicon information is vital in promoting the model to understand token-level semantic information. For each input sequence X, we construct a lexicon tree following the method of (Liu et al., 2021a). As shown in Figure 2, the lexicon set of token x_i can be embedded as $\omega_i = \{\omega_{i1}, \ldots \omega_{im}\}$, where $x_i \in \mathbb{R}^{1\times H}$,$\omega_i \in \mathbb{R}^{m\times H'}$, H is the hidden dimension of each token and H' is the hidden dimension of each word. Moreover, we further introduce a label set for each word. In this work, we adopt a BERT classifier model pre-trained on the high-resource dataset to predict top-k labels embeddings $L_{ij} = \left\{L_{ij}^1, \ldots, L_{ij}^k\right\}$ for ω_{ij}, where $L_{ij} \in \mathbb{R}^{k\times H^*}$, H^* is the hidden dimension of a

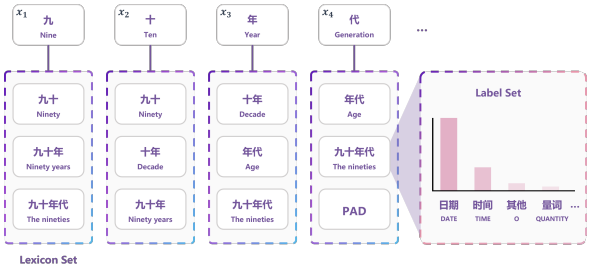

Figure 2: Each token x_i corresponds to a lexicon set, and each lexicon corresponds to a label set.

label string. It is worth noting that each lexicon comes from the external dictionary and is a subset of the input.

A variant of LEBERT is designed to serve as C-BERT in our implementation. As shown in Figure 1, C-BERT's word embedding is the sum of the P-BERT and its word embeddings. We propose a Label Lexicon Adapter (LLA) after the first encoder layer in C-BERT to leverage the lexicon and corresponding labels information. Figure 3. displays the detailed structure of C-BERT, where $H_1^o = \{h_1^o, \ldots, h_n^o\}$ is the set of original output hidden states in the first encoder layer, where the n is the length of the input sequence. In the LLA, the input contains the hidden states H^o from the first encoder layer; the lexicon set ω_i in each token position and corresponding top-k label embedding $L_i = \{L_{i1}, \ldots L_{im}\}$.

We use label attention to compute the relevance between multi-label and lexicon context features, and $\xi_{ij} = [h_i^o; \omega_{ij}]$ is the concatenation between word ω_{ij} and the hidden state h_i^o in position i. Then we transform the multi-label features to align the lexicon context features:

$$\tilde{L}_{ij} = \boldsymbol{W}_2^{\mathrm{L}} \left(\tanh\left(\boldsymbol{W}_1^{\mathrm{L}} L_{ij}^T + \boldsymbol{b}_1^{\mathrm{L}}\right)\right) + \boldsymbol{b}_2^{\mathrm{L}} \quad (5)$$

where $\boldsymbol{W}_1^{\mathbf{L}} \in \mathbb{R}^{(H'+H)\times H^*}$ and $\mathbf{W}_2^{\mathbf{L}} \in \mathbb{R}^{(H'+H)\times(H'+H)}$ are weight matrices; $\boldsymbol{b}_1^{\mathrm{L}}$, $\boldsymbol{b}_2^{\mathrm{L}}$ are biases. The label attention score can be calculated as:

$$\boldsymbol{\alpha}_{ij} = \mathrm{softmax}\left(\xi_{ij} \boldsymbol{W}_{attn}^{\mathrm{L}} \tilde{L}_{ij}\right) \quad (6)$$

where $\boldsymbol{W}_{\mathrm{attn}}^{\mathrm{L}} \in \mathbb{R}^{(H'+H)\times(H'+H)}$ is the label attention weight matrix. The multi-label features can be further computed by the weighted sum:

$$F_{ij}^{\mathrm{L}} = \frac{1}{k} \sum_{t=1}^{k} \alpha_{ij} \tilde{L}_{ij}^t \quad (7)$$

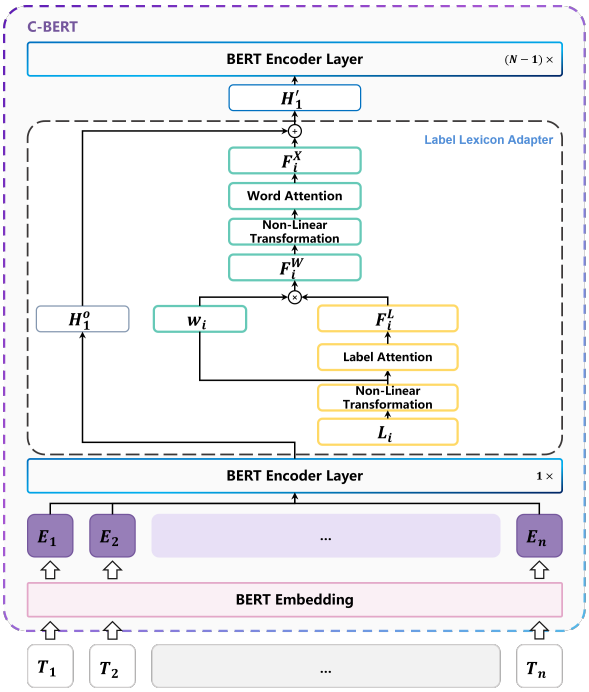

Figure 3: The implementation details of C-BERT.

We fuse features of lexicons with the corresponding label sets to enhance the lexicon representation, and the multi-label features can effectively alleviate the label noise from P-BERT:

$$F_{ij}^{\omega} = \left[\omega_{ij}; F_{ij}^{\mathrm{L}}\right] \quad (8)$$

The computed lexicon features F_i^{ω} are directly injected into the BERT following (Liu et al., 2021a) with the word attention, the lexicon information is calculated by:

$$\widetilde{\omega}_{ij} = \boldsymbol{W}_2^{\omega} \left(\tanh\left(\boldsymbol{W}_1^{\omega} F_{ij}^{\omega^T} + \boldsymbol{b}_1^{\omega}\right)\right) + \boldsymbol{b}_2^{\omega} \quad (9)$$

$$\beta_{ij} = \mathrm{softmax}\left(h_i^o \boldsymbol{W}_{attn}^{\omega} \widetilde{\omega}_{ij}\right) \quad (10)$$

$$F_{ij}^{\mathrm{X}} = \frac{1}{m} \sum_{j=1}^{m} \beta_{ij} \tilde{\omega}_{ij} \quad (11)$$

where $\boldsymbol{W}_1^{\omega} \in \mathbb{R}^{H\times(H'+H^*)}$, $\mathbf{W}_2^{\omega} \in \mathbb{R}^{H\times H}$ are weight matrices; $\boldsymbol{W}_{attn}^{\omega} \in \mathbb{R}^{H\times H}$ is the word attention weight matrix; and $\boldsymbol{b}_1^{\omega}$, $\boldsymbol{b}_2^{\omega}$ are biases.

Finally, the fusion features of each token are computed by:

$$H_1' = H_1^o + F^{\mathrm{X}} \quad (12)$$

3.3 Interactive Training

During fine-tuning, the primary function of P-BERT is to provide label extension features for C-BERT. We intercept the label templates part of the P-BERT output, and the label extension features $F_i^P = \{f_1, \ldots, f_d\}$ are the label slot part corresponding to each label template, where d is the max size of the label string. Then the prompt feature for each token is computed as:

$$P_i = \frac{1}{d} \sum_{j=1}^{d} f_j \tag{13}$$

We use a bidirectional LSTM (BiLSTM) model to enhance the timing information of C-BERT output:

$$H^{\mathrm{B}} = \mathrm{BiLSTM}\left(H_N^{\mathrm{o}}\right) \tag{14}$$

where $H_N^o = \left\{\tilde{h}_1, \ldots \tilde{h}_n\right\}$ is the C-BERT output hidden states.

To further mitigate the impact of the potential label noise, an interactive attention mechanism is applied to calculate the relevance between the output hidden states of BiLSTM $H^B = \left\{\hat{h}_1, \ldots \hat{h}_n\right\}$ and the prompt features P:

$$\gamma_i = \mathrm{softmax}\left(\hat{h}_i \boldsymbol{W}_{attn}^{\mathrm{P}} P_i^T\right) \tag{15}$$

$$\tilde{P}_i = \sum_{i=1}^{n} \gamma_i P_i \tag{16}$$

where $\boldsymbol{W}_{attn}^{P} \in \mathbb{R}^{H \times H}$ is the interactive attention weight matrix, and the fusion features φ can be calculated as:

$$\varphi_i = \left[\widehat{h}_i; \widetilde{P}_i\right] \tag{17}$$

Finally, fusion features are taken into a Conditional Random Field (CRF) layer and predict the label for each token. And the loss function of fine-tuning can be defined by minimizing the negative likelihood loss as:

$$\mathcal{L} = -\sum_{i} \log(p(\boldsymbol{Y} \mid \boldsymbol{X})) \tag{18}$$

4 Experiments

4.1 Datasets

We investigate the effectiveness of our model on four Chinese NER datasets. Including Weibo (Peng and Dredze, 2015), Ontonotes 5.0 (Weischedel et al., 2011), Resume (Zhang and Yang, 2018) and MSRA (Levow, 2006). The statistics of the target datasets are shown in Table 1, and we randomly sample a small train set from each original dataset during training to simulate the few-shot scene.

Table 1: The statistics of the target datasets.

Dataset	Train	Dev	Test	Entity Types
Weibo	1.4k	0.27k	0.27k	8
Ontonotes	15.7k	4.3k	4.3k	4
Resume	3.8k	0.46k	0.48k	8
MSRA	46.4k	-	4.4k	3

Table 2: The statistics of the high-resource dataset.

Subset	Train	Dev	Test	Entity Types
CLUENER	10.7k	1.34k	1.34k	10
CNERTA	38.5k	4.44k	4.44k	5
RenMinRiBao	50.7k	4.63k	4.63k	4
Others	27.0k	2.83k	2.83k	10
Sum	**126.9k**	**13.2k**	**13.2k**	**18**

Besides, we construct a high-resource dataset to implement the label extension. The high-resource dataset is integrated with multiple datasets, including CLUENER (Xu et al., 2020), CNERTA (Sui et al., 2021), RenMinRiBao (Xia et al., 2005), and datasets from unknown sources. The high-resource dataset covers plenty of data and labels, and it can accurately support the label expansion on the low-resource datasets. The statistics of the high-resource dataset are shown in Table 2.

4.2 Experimental Settings

We implement the PCBERT based on the Transformers (Wolf et al., 2020) BERT with 12 layers of transformer in this work. The encoder hidden dimension H of P-BERT and C-BERT is 768; the word embedding dimension of the lexicon H' and label string H^* are both set as 200.

We use the Adam optimizer in all experiments. Before training all the target datasets, we first train a pre-labeled model on the high-resource dataset to annotate the extension entity labels for each train set and generate the label extension train set. Then our P-BERT is trained on the label extension train set. The learning rate of prompt-tuning is set as 1e-4. During fine-tuning on the original train set, the P-BERT is set as frozen, and we use an initial learning rate of 1e-5 for the C-BERT and 1e-2 for other parameters. We sample the same size from all datasets for few-shot learning, the max sequence

length is set as 150, and we train a maximum epoch number of 20 in all datasets.

To evaluate our proposed PCBERT, we compare it with the following approaches:

BERT. (Devlin et al., 2019) The BERT model with a token classifier is the baseline of the BERT-based NER approach.

BERT-LC. Based on the vanilla BERT, we further add a BiLSTM-CRF layer behind the BERT output layer to better compare with our proposed PCBERT.

Lattice LSTM. (Zhang and Yang, 2018) A lexicon-based Chinese NER approach is implemented with a lattice-structure LSTM model.

FLAT. (Li et al., 2020) An enhanced lattice-structured NER approach. By constructing a flat structure Transformer to fully leverage the lattice information and utilize the parallelism of GPUs.

LEBERT. (Liu et al., 2021a) A lexicon enhanced the Chinese sequence labeling model. Integrating external lexicon knowledge into BERT with a Lexicon Adapter layer.

LEBERT-LC. Based on the vanilla LEBERT, we further add a BiLSTM layer behind the BERT output layer in LEBERT to better compare with our proposed PCBERT.

4.3 Overall Results

We randomly sample different samples from the dataset in Table 1 to simulate NER in the few-shot scenario. The train set sampling sizes K are 250, 500, 1000, and 1350 (the max size of Weibo is 1350), respectively. We use the standard F1-score evaluation metrics to compare the performance.

Table 3 illustrates the experimental results of the Chinese few-shot NER. Our model outperforms all related approaches when K is 250 and achieves the best result on all the samples of Weibo and Ontonotes. Besides, our model performance in Weibo at K=250 outperforms other approaches at K=1350, demonstrating that our approach achieves excellent performance on the few-shot dataset.

The experimental results also indicate that all models' performance in different datasets is quite different even under the same sample size. We speculate that it is related to the semantic environment quality of the dataset rather than the number of entity types. Furthermore, our PCBERT shows more significant advantages on Weibo and Resume datasets with worse semantic environment quality.

Table 3: The overall results on Chinese Few-shot NER.

Dataset	Methods	K=250	K=500	K=1000	K=1350
Weibo	BERT	56.42	62.21	61.27	61.21
	BERT-LC	65.10	71.14	72.03	72.45
	Lattice LSTM	40.37	49.54	53.80	58.27
	FLAT	51.42	56.95	58.70	64.27
	LEBERT	65.83	67.12	70.34	69.12
	LEBERT-LC	66.92	71.11	71.80	73.42
	PCBERT	**73.52**	**73.49**	**76.58**	**77.88**
Ontonotes	BERT	63.85	69.50	71.33	72.42
	BERT-LC	65.69	73.54	74.97	77.19
	Lattice LSTM	39.71	45.46	54.54	57.48
	FLAT	49.01	46.35	49.34	57.44
	LEBERT	69.48	69.01	73.78	74.84
	LEBERT-LC	70.26	69.89	73.83	76.01
	PCBERT	**74.42**	**75.62**	**78.33**	**81.52**
Resume	BERT	53.80	62.64	69.36	70.65
	BERT-LC	92.26	**94.66**	95.16	**96.41**
	Lattice LSTM	85.63	89.60	92.01	93.13
	FLAT	84.62	90.77	92.97	87.79
	LEBERT	89.15	92.56	94.02	95.19
	LEBERT-LC	91.60	93.03	**95.40**	95.16
	PCBERT	**93.42**	94.01	94.96	95.97
MSRA	BERT	68.44	72.28	81.21	82.28
	BERT-LC	79.01	83.13	87.84	89.32
	Lattice LSTM	54.69	63.61	74.27	76.31
	FLAT	59.62	70.20	80.79	64.95
	LEBERT	79.11	85.18	87.77	89.35
	LEBERT-LC	80.92	**86.09**	**88.11**	88.70
	PCBERT	**81.08**	85.25	87.88	**89.72**

4.4 Analysis and Discussion

Ablation Study

We analyze the impact of each module in our PCBERT by designing several experiments. Table 4 presents the performance comparison between PCBERT and other ablation models. First, we observe a performance decline when removing the P-BERT component, demonstrating that P-BERT plays a vital role in model performance. We then observe that its results outperform LEBERT and LEBERT-LC on Weibo and Ontonotes when K is less than or equal to 500, which verifies that multi-label features can improve the model performance in the few-shot scenario. Moreover, after removing the label extension strategy (LEA) by using the original annotated dataset to train the model, the performance also decreases, indicating that the label extension strategy is effective in our approach.

To further analyze the impact of the label extension strategy, we replace the label extension dataset with the high-resource dataset to train the P-BERT (LEB). The results in Table 4 show a severe model performance decrease when directly adopting the high-resource dataset for prompt-tuning. Furthermore, the phenomenon becomes more prominent when the sample size K becomes smaller. And we observed there are different decrease degrees in

Table 4: Results of the Ablation Study on Chinese Few-shot NER.

Dataset	Methods	K=250	K=500	K=1000	K=1350
Weibo	PCBERT	**73.52**	**73.49**	**76.58**	**77.88**
	-P-BERT	67.28	71.85	70.02	72.66
	-LEA	67.06	70.31	71.88	72.73
	-LEB	61.95	67.01	68.62	69.33
Ontonotes	PCBERT	**74.42**	**75.62**	**78.33**	**81.52**
	-P-BERT	72.94	72.42	72.55	74.66
	-LEA	69.13	72.10	74.24	72.62
	-LEB	62.23	66.07	68.86	70.09
Resume	PCBERT	**93.42**	94.01	**94.96**	**95.97**
	-P-BERT	91.18	92.99	94.41	95.41
	-LEA	91.28	**94.33**	**94.96**	95.55
	-LEB	87.17	91.64	92.97	93.96
MSRA	PCBERT	**81.08**	85.25	**87.88**	**89.72**
	-P-BERT	80.59	**85.50**	86.95	87.88
	-LEA	82.77	84.32	86.20	84.32
	-LEB	79.09	81.36	83.61	84.75

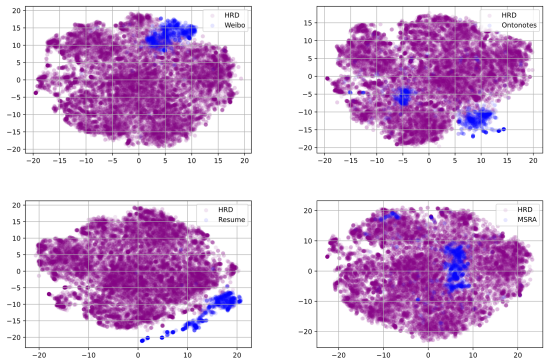

Figure 4: t-SNE visualization of each sampled train set and the high-resource dataset.

different datasets. For example, the performance on Weibo decreased by 11.57% (K=250), while MSRA decreased by only 1.99% (K=250). We present the visualization of semantic distribution between each sampled (K=1350) train set and the high-resource dataset (HRD) in Figure 4. The sentence-level representation is obtained from the BERT embedders onto a 2-dimensional space using t-SNE (Van der Maaten and Hinton, 2008). It can be concluded from Figure 4 and Table 4 that when the semantic space gap between the source dataset and the target dataset increases, transfer training directly from the source dataset will decrease the model performance.

Impact of Feature Injection

The results in Table 3 and Table 4 have demonstrated that the injected lexicon and multi-label features in C-BERT can effectively enhance the model performance. We speculate that multi-type lexicon or multi-label features injection can improve

Table 5: Comparison between LEBERT with random initial lexicon embeddings (LEBERT-RW) and original LEBERT.

Dataset	Methods	K=250	K=500	K=1000	K=1350
Weibo	LEBERT	65.83	67.12	70.34	69.12
	LEBERT-RW	64.08	67.16	68.89	70.42
Ontonotes	LEBERT	69.48	69.01	73.78	74.84
	LEBERT-RW	66.65	71.41	73.93	75.96
Resume	LEBERT	89.15	92.56	94.02	95.19
	LEBERT-RW	90.77	93.44	94.77	95.68
MSRA	LEBERT	79.11	85.18	87.77	89.35
	LEBERT-RW	79.34	83.83	88.74	88.59

the model's perception of fine-grained information and judgment of entity boundaries. Moreover, we further adopt LEBERT with random initial lexicon embeddings (LEBERT-RW) to compare the original LEBERT on four datasets. As shown in Table 5, the performance of LEBERT-RW is similar to LEBERT, which indicates that the boundary information introduced by feature injection is more critical to the model than the semantic distribution of the word embeddings.

Impact of Label Extension

To further analyze the impact of the label extension strategy, we evaluate the PCBERT performance when each extension label is removed from the label extension train set of Weibo (K=1350). Figure 6 illustrates the results, sorted in descending order according to each metric. We can conclude that, in most cases, removing an extension label will cause the model performance to decrease. It also shows that in the Weibo dataset, introducing any extension label will bring the final performance improvement in prompt-tuning, which indirectly indicates that our prompt-based PCBERT can effectively suppress the label extension noise.

Sentence Length

Figure 5 shows the F1-score trend of all baselines and PCBERT on the four datasets in Table 1 with the sampling size of 250. As shown in the results, we discover that PCBERT significantly improves performance in all sentence length intervals of the Weibo and Ontonotes datasets. Comparing the results of LEBERT and LEBERT-LC, it can be observed that adding the BiLSTM layer improves performance in the sampled Weibo and MSRA datasets. One potential reason is that the BiLSTM has a better awareness of directionality and short-distance information. To achieve more stable performances, we add the BiLSTM layer behind the

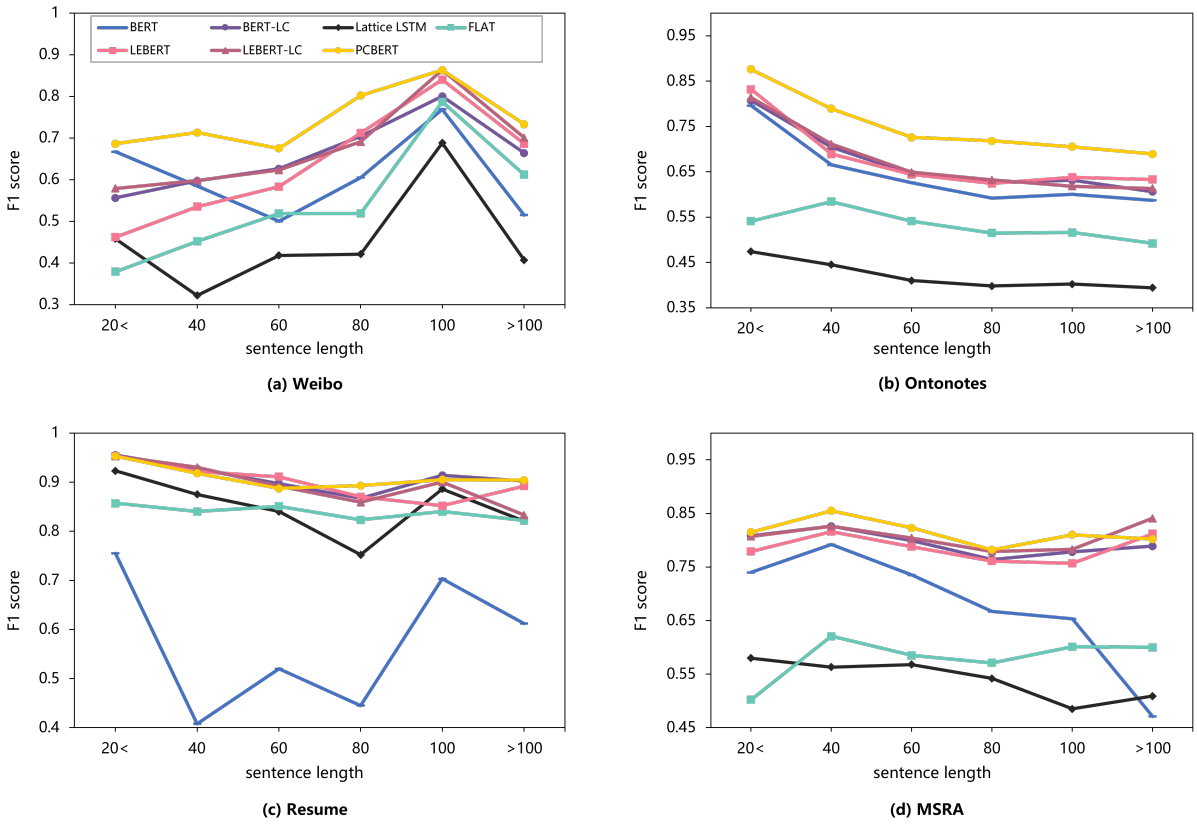

Figure 5: F1-scores against the sentence length.

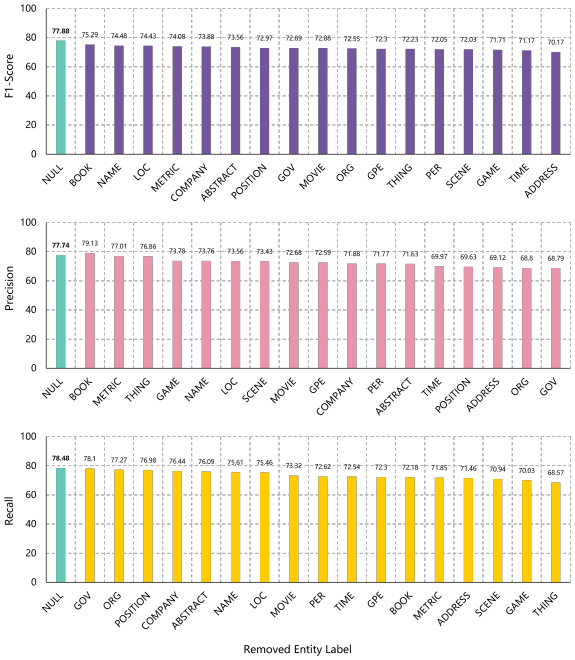

Figure 6: F1-score, Precision, and Recall comparison of PCBERT on the Weibo dataset when removing each extension entity label, where NULL indicates the original label extension train set.

C-BERT.

5 Related Works

Chinese NER

NER is a fine-grained sequence labeling task. With the advent of PLMs, the benchmark of Chinese NER has been dramatically improved. Pre-trained models based on large-scale corpus (Devlin et al., 2019; Lewis et al., 2020; Radford et al., 2019) provide excellent semantic representation for Chinese NER and are used by many works. Some work adds a softmax on PLMs (Yang, 2019) and achieves significant performance; others (Peters et al., 2018; Zheng et al., 2021; Nan et al., 2021) take PLMs as the backbone model to further enhance the original model performance.

Despite the remarkable achievements of PLMs, most existing models still need to be improved in judging Chinese word boundaries. Lexicon-based approaches (Zhang and Yang, 2018; Ma et al., 2020; Gui et al., 2019; Zhao et al., 2020) can effectively alleviate this issue. In particular, many lexicon-based works like Lex-BERT (Zhu and Cheung, 2021) need a high-quality vocabulary with entity-type information. (Zhang and Yang,

2018) proposed the Lattice LSTM approach to leverage all potential words in each segment and only need word vectors, which provided great inspiration for the later work. Recently many works (Xiao et al., 2019; Sperber et al., 2019; Zhang et al., 2019a,b) presented lattice-based transformers to promote parallel computing performance and fuse the PLMs representation into the model. However, most lattice-based transformers only fuse dictionary features in external input sequences without integrating them into the PLM structure. (Liu et al., 2021a) proposed LEBERT integrates lexicon knowledge into BERT layers and achieved state-of-the-art performance in multiple Chinese NER datasets.

Prompt-tuning

With the emergence of GPT-3 (Brown et al., 2020), the target-task-oriented pre-training form attracted a lot of attention (Schick and Schütze, 2021). Prompt-tuning (Lester et al., 2021) can be regarded as a new template-based pre-training paradigm. Unlike fine-tuning, the downstream task of prompt-tuning is homologous to pre-training. Prompt-tuning is more dependent on the prior distribution of the model, while fine-tuning is more dependent on the posterior distribution (Qiu et al., 2020).

Designing appropriate prompt templates for different tasks is crucial in prompt-tuning performance (Liu et al., 2021b). There is no universal template for all NLP tasks. (Jiang et al., 2020; Yuan et al., 2021; Haviv et al., 2021) proposed discrete prompts to disassemble and replace sentence components for text inference tasks; and (Gao et al., 2021; Ben-David et al., 2021) designed the generation prompt to build generated templates by automatically extracting semantic information from sentences.

In NER tasks, the model requires more specific semantic fine-grained information. Therefore, prompt templates construction approaches for other natural language understanding tasks can not work out well on NER tasks. (Ma et al., 2022) put forward a template-free approach to complete the entity template using the word vector mean of the same entity in the dataset. And (Chen et al., 2021) use an encoder-decoder model to translate the NER task into a prompt-based generation task.

6 Conclusion

In this paper, we propose a Parent and Child BERT for Chinese few-shot NER tasks and achieve state-of-the-art results on the Weibo dataset. Our model consists of P-BERT and C-BERT, where P-BERT is a prompt-based model for providing richer semantic information, and C-BERT is a lexicon-based model. The experimental results demonstrate that our PCBERT effectively improves the performance on the Chinese few-shot NER task. In the future, we will further analyze the performance improvement of label extension strategy in domain-specific datasets.

7 Acknowledgments

This work is supported by the Natural Science Foundation of Fujian Province, PR China (2022J01120).

References

Abhishek Abhishek, Ashish Anand, and Amit Awekar. 2017. Fine-grained entity type classification by jointly learning representations and label embeddings. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 1, Long Papers*, pages 797–807.

Eyal Ben-David, Nadav Oved, and Roi Reichart. 2021. Pada: A prompt-based autoregressive approach for adaptation to unseen domains. *arXiv preprint arXiv:2102.12206*.

Tom B. Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel M. Ziegler, Jeffrey Wu, Clemens Winter, Christopher Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. 2020. Language models are few-shot learners. In *Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020, December 6-12, 2020, virtual*.

Xiang Chen, Ningyu Zhang, Lei Li, Xin Xie, Shumin Deng, Chuanqi Tan, Fei Huang, Luo Si, and Huajun Chen. 2021. Lightner: A lightweight generative framework with prompt-guided attention for low-resource ner. *arXiv preprint arXiv:2109.00720*.

Leyang Cui, Yu Wu, Jian Liu, Sen Yang, and Yue Zhang. 2021. Template-based named entity recognition using BART. In *Findings of the Association for Computational Linguistics: ACL/IJCNLP 2021, Online*

Event, August 1-6, 2021, volume ACL/IJCNLP 2021 of *Findings of ACL*, pages 1835–1845. Association for Computational Linguistics.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. Bert: Pre-training of deep bidirectional transformers for language understanding. In *NAACL-HLT (1)*.

Tianyu Gao, Adam Fisch, and Danqi Chen. 2021. Making pre-trained language models better few-shot learners. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing, ACL/IJCNLP 2021, (Volume 1: Long Papers), Virtual Event, August 1-6, 2021*, pages 3816–3830. Association for Computational Linguistics.

Tao Gui, Ruotian Ma, Qi Zhang, Lujun Zhao, Yu-Gang Jiang, and Xuanjing Huang. 2019. Cnn-based chinese ner with lexicon rethinking. In *IJCAI*, pages 4982–4988.

Adi Haviv, Jonathan Berant, and Amir Globerson. 2021. Bertese: Learning to speak to bert. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*, pages 3618–3623.

Zhengbao Jiang, Frank F Xu, Jun Araki, and Graham Neubig. 2020. How can we know what language models know? *Transactions of the Association for Computational Linguistics*, 8:423–438.

Ji Young Lee, Franck Dernoncourt, and Peter Szolovits. 2017. Transfer learning for named-entity recognition with neural networks.

Brian Lester, Rami Al-Rfou, and Noah Constant. 2021. The power of scale for parameter-efficient prompt tuning. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing, EMNLP 2021, Virtual Event / Punta Cana, Dominican Republic, 7-11 November, 2021*, pages 3045–3059. Association for Computational Linguistics.

Gina-Anne Levow. 2006. The third international chinese language processing bakeoff: Word segmentation and named entity recognition. In *Proceedings of the Fifth SIGHAN Workshop on Chinese Language Processing*, pages 108–117.

Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Veselin Stoyanov, and Luke Zettlemoyer. 2020. Bart: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7871–7880.

Xiaonan Li, Hang Yan, Xipeng Qiu, and Xuan-Jing Huang. 2020. Flat: Chinese ner using flat-lattice transformer. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 6836–6842.

Wei Liu, Xiyan Fu, Yue Zhang, and Wenming Xiao. 2021a. Lexicon enhanced chinese sequence labelling using bert adapter. *arXiv preprint arXiv:2105.07148*.

Xiao Liu, Kaixuan Ji, Yicheng Fu, Zhengxiao Du, Zhilin Yang, and Jie Tang. 2021b. P-tuning v2: Prompt tuning can be comparable to fine-tuning universally across scales and tasks. *arXiv preprint arXiv:2110.07602*.

Ruotian Ma, Minlong Peng, Qi Zhang, Zhongyu Wei, and Xuan-Jing Huang. 2020. Simplify the usage of lexicon in chinese ner. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 5951–5960.

Ruotian Ma, Xin Zhou, Tao Gui, Yiding Tan, Linyang Li, Qi Zhang, and Xuanjing Huang. 2022. Template-free prompt tuning for few-shot NER. In *Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL 2022, Seattle, WA, United States, July 10-15, 2022*, pages 5721–5732. Association for Computational Linguistics.

Guoshun Nan, Jiaqi Zeng, Rui Qiao, Zhijiang Guo, and Wei Lu. 2021. Uncovering main causalities for long-tailed information extraction. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 9683–9695.

Nanyun Peng and Mark Dredze. 2015. Named entity recognition for chinese social media with jointly trained embeddings. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 548–554.

ME Peters, M Neumann, M Iyyer, M Gardner, C Clark, K Lee, and L Zettlemoyer. 2018. Deep contextualized word representations. naacl-hlt.

Xipeng Qiu, Tianxiang Sun, Yige Xu, Yunfan Shao, Ning Dai, and Xuanjing Huang. 2020. Pre-trained models for natural language processing: A survey. *Science China Technological Sciences*, pages 1–26.

Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, Ilya Sutskever, et al. 2019. Language models are unsupervised multitask learners. *OpenAI blog*, 1(8):9.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J. Liu. 2020. Exploring the limits of transfer learning with a unified text-to-text transformer. *Journal of Machine Learning Research*, 21(140):1–67.

Timo Schick and Hinrich Schütze. 2021. It's not just size that matters: Small language models are also few-shot learners. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 2339–2352.

Matthias Sperber, Graham Neubig, Ngoc-Quan Pham, and Alex Waibel. 2019. Self-attentional models for lattice inputs. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 1185–1197.

Dianbo Sui, Yubo Chen, Kang Liu, Jun Zhao, and Shengping Liu. 2019. Leverage lexical knowledge for chinese named entity recognition via collaborative graph network. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 3830–3840.

Dianbo Sui, Zhengkun Tian, Yubo Chen, Kang Liu, and Jun Zhao. 2021. A large-scale Chinese multimodal NER dataset with speech clues. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*.

Laurens Van der Maaten and Geoffrey Hinton. 2008. Visualizing data using t-sne. *Journal of machine learning research*, 9(11).

Ralph Weischedel, Sameer Pradhan, Lance Ramshaw, Martha Palmer, Nianwen Xue, Mitchell Marcus, Ann Taylor, Craig Greenberg, Eduard Hovy, Robert Belvin, et al. 2011. Ontonotes release 4.0. *LDC2011T03, Philadelphia, Penn.: Linguistic Data Consortium*.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Rémi Louf, Morgan Funtowicz, Joe Davison, Sam Shleifer, Patrick von Platen, Clara Ma, Yacine Jernite, Julien Plu, Canwen Xu, Teven Le Scao, Sylvain Gugger, Mariama Drame, Quentin Lhoest, and Alexander M. Rush. 2020. Transformers: State-of-the-art natural language processing. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 38–45, Online. Association for Computational Linguistics.

YingJu Xia, Hao Yu, and Fumihito Nishino. 2005. The Chinese named entity categorization based on the people's daily corpus. In *International Journal of Computational Linguistics & Chinese Language Processing, Volume 10, Number 4, December 2005: Special Issue on Selected Papers from CLSW-5*, pages 533–542.

Fengshun Xiao, Jiangtong Li, Hai Zhao, Rui Wang, and Kehai Chen. 2019. Lattice-based transformer encoder for neural machine translation. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 3090–3097.

Liang Xu, Qianqian Dong, Cong Yu, Yin Tian, Weitang Liu, Lu Li, and Xuanwei Zhang. 2020. Cluener2020: Fine-grained name entity recognition for chinese. *arXiv preprint arXiv:2001.04351*.

Haiqin Yang. 2019. Bert meets chinese word segmentation. *arXiv preprint arXiv:1909.09292*.

Zhilin Yang, Ruslan Salakhutdinov, and William W. Cohen. 2017. Transfer learning for sequence tagging with hierarchical recurrent networks. In *5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Conference Track Proceedings*. OpenReview.net.

Weizhe Yuan, Graham Neubig, and Pengfei Liu. 2021. Bartscore: Evaluating generated text as text generation. In *Advances in Neural Information Processing Systems 34: Annual Conference on Neural Information Processing Systems 2021, NeurIPS 2021, December 6-14, 2021, virtual*, pages 27263–27277.

Pei Zhang, Niyu Ge, Boxing Chen, and Kai Fan. 2019a. Lattice transformer for speech translation. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 6475–6484.

Pei Zhang, Niyu Ge, Boxing Chen, and Kai Fan. 2019b. Lattice transformer for speech translation. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 6475–6484.

Yue Zhang and Jie Yang. 2018. Chinese ner using lattice lstm. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1554–1564.

Xiaoyan Zhao, Min Yang, Qiang Qu, and Yang Sun. 2020. Improving neural chinese word segmentation with lexicon-enhanced adaptive attention. In *Proceedings of the 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 1953–1956.

Hengyi Zheng, Rui Wen, Xi Chen, Yifan Yang, Yunyan Zhang, Ziheng Zhang, Ningyu Zhang, Bin Qin, Xu Ming, and Yefeng Zheng. 2021. PRGC: potential relation and global correspondence based joint relational triple extraction. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing, ACL/IJCNLP 2021, (Volume 1: Long Papers), Virtual Event, August 1-6, 2021*, pages 6225–6235. Association for Computational Linguistics.

Wei Zhu and Daniel Cheung. 2021. Lex-bert: Enhancing bert based ner with lexicons. *arXiv preprint arXiv:2101.00396*.

Label Smoothing for Text Mining

Peiyang Liu[1,2,*], **Xiangyu Xi**[3,*], **Wei Ye**[1,†] and **Shikun Zhang**[1]
[1] National Engineering Research Center for Software Engineering, Peking University, Beijing, China,
[2] PX Securities, Shenzhen, China,
[3] Meituan, Beijing, China,
{liupeiyang, wye, zhangsk}@pku.edu.cn, xixiangyu@meituan.com

Abstract

Current text mining models are trained with 0-1 hard label that indicates whether an instance belongs to a class, ignoring rich information of the relevance degree. Soft label, which involved each label of more varying degrees than the hard label, is considered more suitable for describing instances. The process of generating soft labels from hard labels is defined as label smoothing (LS). Classical LS methods focus on universal data mining tasks so that they ignore the valuable text features in text mining tasks. This paper presents a novel keyword-based LS method to automatically generate soft labels from hard labels via exploiting the relevance between labels and text instances. Generated soft labels are then incorporated into existing models as auxiliary targets during the training stage, capable of improving models without adding any extra parameters. Results of extensive experiments on text classification and large-scale text retrieval datasets demonstrate that soft labels generated by our method contain rich knowledge of text features, improving the performance of corresponding models under both balanced and unbalanced settings.

1 Introduction

Instances in most text mining datasets are usually assigned by one label. Such label, called *hard label*, reflects the logical relationship between the label and the instance. In most cases, hard labels are utilized for training models that learn to discriminate between classes with training objectives to maximize the log probability of the correct class. However, there are various real-world text mining tasks where there are not just two possibilities whether an instance belongs to a specific class since instances may be involved with multiple labels of varying degrees. In such scenarios, soft label (Bahri and Jiang, 2021; Hong et al., 2022), which involves the explicit relative importance of each label, is a more reasonable description of an instance. For example, in the sentence S1 shown in the Table 1,[1] though both positive and negative expressions exist in the sentence and sentiment slightly inclines to be positive, annotators must give a solid positive as a hard label. The soft label (0.6,0.4) describes the instance more comprehensively than hard labels.

S1: It was **nice**. The cashier was **chipper**. Wish they would crank the A/C **a little more though**. Restroom was **a little dirty**.	
Hard Label	(1, 0)
Soft Label	(0.6, 0.4)

Table 1: An instance of sentiment classification with hard label and soft label. X denotes the label of being positive, while Y denotes negative in (X, Y). Positive and negative expressions are marked in red and blue, respectively.

Few works are investigating soft labels for text data in the text mining community, although the issue demonstrated in the above example is very common. Existing works all focus on analyzing the effectiveness of classical soft labels in universal data mining tasks (Müller et al., 2019; Wu et al., 2021; Nguyen et al., 2022). However, classical soft labels cannot leverage useful text features in text mining datasets. Meanwhile, in most text datasets, soft labels are not explicitly available. Therefore, automatically generating soft labels from hard labels based on text data characteristics is a fundamental problem worth exploring.

The key to generate informative soft labels for text instances is to exploit the semantic relevance between instances and labels accurately. We are inspired by the fact that a set of highly representative words can represent semantic information of labels and instances (e.g., S1 in Table 1) (Spärck Jones, 1972, 2004), we present a **K**ey**w**ord-based **L**abel **S**moothing method (**KWLS**), which primarily in-

[*] Co-first authors, contributed to this work equally
[†] Corresponding author

[1] A sentence from Yelp Review dataset.

Proceedings of the 29th International Conference on Computational Linguistics, pages 2210–2219
October 12–17, 2022.

volves four steps: (1) Keyword detection for labels, which computes correlations between words and labels; (2) Keyword detection for instances, which calculates the saliency of a word to represent a given instance; (3) Instance-label relevance detection, which determines the correlations between instances and labels; (4) Soft label generation, which generates soft labels for instances based on the instance-label relevance. The generated soft labels can be utilized as complementary targets during the training stage, containing knowledge of label correlation and introducing auxiliary supervision information to improve models.

To verify the effectiveness of our method, we conduct extensive experiments on the two most typical application scenarios in text mining: text classification and large-scale text retrieval. Experimental results demonstrate that models equipped with KWLS gain significant improvements over the original models, especially in the highly unbalanced large-scale text retrieval task (Liu et al., 2021c). To further analyze the ability of KWLS to deal with unbalanced problems, we construct various unbalanced datasets to simulate multifarious unbalanced problems in text mining. Experimental results on different unbalanced settings show that KWLS may bring extra supervised signals to facilitate model learning.

Our main contributions are listed as follows:

1. Previous studies focus on leveraging label smoothing on universal data mining tasks so that they ignore the valuable text features in text mining tasks. We propose a novel keyword-based LS method (KWLS) that automatically generates soft labels with rich knowledge of label correlation from hard labels in text data.

2. Generated soft labels can be incorporated as complementary targets, introducing auxiliary supervision information, capable of improving models without adding any extra parameters.

3.Experimental results show that the knowledge of label correlation characterized by KWLS is practical under balanced and unbalanced settings and more suitable for text mining tasks than classical label smoothing methods.

2 Related Work

Label smoothing, a form of output distribution regularization, prevents overfitting of a neural network by softening the ground-truth labels to penalize overconfident outputs (Li et al., 2020), has made tremendous achievements in many data mining fields. For example:

Müller et al. (2019) reveal that label smoothing improves model calibration and summarize several behaviors observed while training deep neural networks with label smoothing.

Chelombiev et al. (2019) propose an improved mutual information estimator based on binning and show the correlation between compression of softmax layer representations and generalization, which may explain why networks trained with label smoothing generalize so well.

The above works analyze the effectiveness of classical label smoothing methods in various fields. The two most classical label smoothing methods are follows:

Szegedy et al. (2016) first introduced label smoothing that improves accuracy by computing cross-entropy not with the “hard" targets from the dataset, but with a weighted mixture of these targets with the uniform distribution, and many state-of-the-art image classification models have incorporated label smoothing into training procedures ever since.

Zhang and Sabuncu (2020) regard self-distillation as a label smoothing method and propose a novel instance-specific label smoothing technique that promotes predictive diversity without the need for a separately trained teacher model.

However, these two methods mentioned above are proposed for universal data mining tasks so that they cannot leverage the valuable text features in text mining datasets. This paper proposes a KWLS method explicitly designed for text mining, which incorporates semantic relevance between labels and instances into soft labels. Experimental results demonstrate that our method is more suitable for text data.

It is worth noting that the primary purpose of LS is incorporating the possibility (or uncertainty) into the original hard label to facilitate model performances rather than generating the ground truth soft labels.

3 Task Description and Formulation

Given a training set $D = \{(x_i, \boldsymbol{l_i})|1 \leq i \leq N\}$ with N instances, x_i is a sequence of words $x_i = \{w_1, w_2, ..., w_n\}$ where w_1, w_2, and w_n are words in the sentence and n is the length of the sequence. The hard label vector of x_i is denoted by $\boldsymbol{l_i} = (l_i^1, l_i^2, ..., l_i^p)$, where $l_i^j \in \{0, 1\}$ denotes whether

label l^j describes x_i, where p is the total number of labels. Our task is to generate the soft label $\boldsymbol{d_i}$ of x_i where $\boldsymbol{d_i} = (d_i^1, d_i^2, ..., d_i^p)$ is a distribution vector. The training set D is thus transformed into $\mathcal{E} = \{(x_i, \boldsymbol{d_i}) | 1 \leq i \leq N\}$ which is available for further exploration.

The proposed KWLS method is introduced in the following section.

4 Keyword-Based Label Smoothing

To represent labels and instances by a set of highly representative words, we propose to first determine keywords for instances and labels then calculate correlations between them. Our method mainly involves the following four components.

4.1 Keyword Detection for Labels

This section illustrates how to detect keywords for each label. We propose to use **Word-Label Relevance (WLR)** based on TF-IDF (Term Frequency–Inverse Document Frequency) (Spärck Jones, 1972, 2004) and BM25 (Robertson et al., 2009) to estimate the importance of a word to a label, which is decided by two factors: *Word-Label Weight* and *Word-Label Correlation*.

Word-Label Correlation (WLC) based on TF (Term Frequency) reflects the saliency of a word to represent a given label. In traditional TF, if a word frequently appears in instances of a given label, the word will own a larger WLC to the label. However, if a label contains 200 occurrences of "COLING", is it twice as relevant as a label containing 100 occurrences? We could argue that if "COLING" occurs a large enough number of times, say 100, the label is almost certainly relevant, and any further mentions do not increase the likelihood of relevance. According to the observation by Robertson et al. (2009), when the word frequency becomes larger, the WLC will grow slower. We define WLC as follows:

$$
\begin{aligned}
&WLC_{w_j,m} = \frac{f_{w_j,X_m} * k_1}{f_{w_j,X_m} + K}, w_j \in C, \\
&K = k_1 * (1 - b + b * \frac{S_m}{S_{avg}}) \\
&S_m = \sum_{x_i \in X_m} |x_i| \\
&S_{avg} = \frac{1}{p} * \sum_{i}^{p} S_i \\
&X_m = \{x_i | x_i \in X, l_i^m = 1\}
\end{aligned}
\tag{1}
$$

where f_{w_j,X_m} equals the number of times w_j appears in X_m, X_m is the set where $x_i \in X$ and $l_i^m = 1$, and C is the corpus. k_1 is a positive hyperparameter, which is used to scale and control the f_{w_j,X_m}. If k_1 is set to close to 0, then f_{w_j,X_m} will be ignored. If k_1 is set to a large value, then the $WLC_{w_j,m}$ is equal to f_{w_j,X_m}. $|x_i|$ is the number of words in x_i, and S_m is the number of words in X_m. S_{avg} is the average number of words in all labels. b is another hyperparameter to control S_m to normalize word frequency, and $0 \leq b \leq 1$. When b is set to 1, the f_{w_j,X_m} will be fully scaled based on S_m. When b is set to 0, S_m is not taken into account in normalization. $WLC_{w_j,m}$ is the Word-Label Correlation between w_j and the m-th label.

Word-Label Weight (WLW) based on IDF (Inverse Document Frequency) reflects the ability in which a word can be used to discriminate different labels. If a word occurs in every label, the word will have a low WLW. In traditional IDF, if a word occurs once in every label, the WLW of the word will be set to 0. However, words can easily occur in every label at least once, which will make most of the words' WLW be set to 0 unreasonably. So that we modified the traditional IDF, and our WLW can be calculated as follows:

$$
\begin{aligned}
&WLW_{w_j} = \log \frac{p}{sum(\mathbf{F_{w_j,L}})} + 1 \\
&\mathbf{F_{w_j,L}} = \{F_{w_j,l^1}, ..., F_{w_j,l^m}, ..., F_{w_j,l^p}\} \\
&F_{w_j,l^m} = \begin{cases} 1, f_{w_j,X_m} \geq 1 \\ \\ 0, f_{w_j,X_m} < 1 \end{cases}
\end{aligned}
\tag{2}
$$

where $L = \{l^1, l^2, ..., l^p\}$ denote the finite set of labels and p is the number of possible labels. WLW_{w_j} is the weight of w_j to labels. $\mathbf{F_{w_j,L}}$ is a vector to represent whether w_j occurs in labels, where $F_{w_j,l^m} = 1$ means the word w_j occurs in the m-th label. Finally, WLR is computed as follows:

$$
WLR_{w_j,m} = WLW_{w_j} * WLC_{w_j,m} \tag{3}
$$

where $WLR_{w_j,m}$ is the Word-Label Relevance between w_j and the m-th label.

4.2 Keyword Detection for Instances

This section illustrates how to detect keywords for each instance. We propose to use **Word-Instance Relevance (WIR)** to estimate the importance of a word to an instance, which is decided by two

factors: *Word-Instance Weight* and *Word-Instance Correlation*.

Word-Instance Correlation (WIC) based on TF reflects the saliency of a word to represent a given instance. Similar to WLC, if a word frequently appears in a given instance, the word will own a larger WIC to the instance, and the correlation between word frequency and WIC is also nonlinear. We define WIC as follows:

$$WIC_{w_j,i} = \frac{f_{w_j,x_i} * k_2}{f_{w_j,x_i} + k_2}, w_j \in x_i \tag{4}$$

where f_{w_j,x_i} is the number of times w_j appears in x_i. k_2 is a positive hyperparameter, which is used to scale and control the word frequency of x_i. $WIC_{w_j,i}$ is the Word-Instance Correlation between w_j and the x_i.

Word-Instance Weight (WIW) based on IDF reflects the ability in which a word can be used to discriminate different instances. Similar to WLW, WIW can be calculated as follows:

$$\begin{aligned} WIW_{w_j} &= \log \frac{N}{sum(\mathbf{F_{w_j,X}})} + 1 \\ \mathbf{F_{w_j,X}} &= \{F_{w_j,x_1}, ..., F_{w_j,x_i}, ..., F_{w_j,x_N}\} \\ F_{w_j,x_i} &= \begin{cases} 1, f_{w_j,x_i} \geq 1 \\ \\ 0, f_{w_j,x_i} < 1 \end{cases} \end{aligned} \tag{5}$$

where $X = \{x_1, x_2, ..., x_N\}$ denote the set of instances and N is the size of X. WIW_{w_j} is the weight of w_j to instances. $\mathbf{F_{w_j,X}}$ is a vector to represent whether w_j occurs in instances, where $F_{w_j,x_i} = 1$ means the word w_j occurs in the i-th instance. Finally, WIR is computed as follows:

$$WIR_{w_j,i} = WIW_{w_j} * WIC_{w_j,i} \tag{6}$$

where $WIR_{w_j,i}$ is the Word-Instance Relevance between w_j and the i-th instance.

4.3 Instance-Label Relevance Detection

We propose **Instance-Label Relevance (ILR)** to represent correlations between instances and labels, which is calculated as follows:

$$\begin{aligned} ILR_{i,m} &= \frac{1}{X_{avg}} \sum_{w_j \in x_i} WLR_{w_j,m} * WIR_{w_j,i} \\ X_{avg} &= \frac{1}{N} \sum_{x_i \in X} |x_i| \end{aligned} \tag{7}$$

where X_{avg} is the average length of all instances. $ILR_{i,m}$ is the relevance between the i-th instance and the m-th label.

4.4 Soft Label Generation

After the ILR is computed, we generate the soft label by the softmax function (Elfadel and Jr., 1993):

$$\boldsymbol{d_i} = Softmax(\boldsymbol{ILR_i}) \tag{8}$$

4.5 Incorporating Soft Label

After soft labels are generated for each instance, we incorporate soft labels as auxiliary fitting targets, and the loss function can be defined as:

$$Loss = -\frac{\sum_{i=1}^{N} \sum_{j=1}^{p} [l_i^j \log(o_i^j) - \frac{\alpha}{p}(d_i^j - o_i^j)^2]}{2N(1+\alpha)^2} \tag{9}$$

where o_i^j is j-th dimension of i-th instance's output probability predicted by the model. α denotes the loss weight of soft label during the training stage. We use Mean Squared Error (MSE) as the loss function for soft labels, where α denotes the loss weight of soft label during the training stage. When α is set to 0, the model degenerates into the standard classifier.

5 Experiments and Results

5.1 Datasets

For text classification, the following four real-world datasets are used in the experiment.

AG News consists of news articles from the AG's corpus of news articles on the web about the four largest classes (Technology, Sports, Business, and World). The dataset contains 30,000 training and 1,900 testing examples for each class (Gulli, 2005; Del Corso et al., 2005).

DBpedia is a project aiming to extract structured content from the information created in the Wikipedia project (Lehmann et al., 2015). The DBpedia ontology dataset contains 560,000 training samples and 70,000 testing samples for each of 14 non-overlapping classes from DBpedia (Lehmann et al., 2015).

IMDb is a binary sentiment analysis dataset consisting of 50,000 reviews from the Internet Movie Database (IMDb) labeled positive or negative. The dataset contains an even number of positive and negative reviews (Maas et al., 2011).

Yelp is a subset of businesses, reviews, and user data for use in personal, educational, and academic purposes (Zhang et al., 2015).

DataSet	Model	Accuracy	Model	Accuracy	Model	Accuracy
AG News	BERT	94.08	TextCNN	88.88	LSTM	87.02
	+ LS-Classic	94.17	+ LS-Classic	89.37	+ LS-Classic	87.93
	+ LS-Distill	94.37	+ LS-Distill	89.56	+ LS-Distill	88.17
	+ LS-TFIDF	94.42	+ LS-TFIDF	90.11	+ LS-TFIDF	90.13
	+ KWLS	**95.13**	+ KWLS	**91.07**	+ KWLS	**90.85**
DBpedia	BERT	99.30	TextCNN	98.48	LSTM	98.71
	+ LS-Classic	99.35	+ LS-Classic	98.52	+ LS-Classic	98.77
	+ LS-Distill	99.40	+ LS-Distill	98.63	+ LS-Distill	98.80
	+ LS-TFIDF	99.41	+ LS-TFIDF	98.86	+ LS-TFIDF	98.84
	+ KWLS	**99.65**	+ KWLS	**99.12**	+ KWLS	**99.13**
IMDb	BERT	93.21	TextCNN	91.62	LSTM	91.03
	+ LS-Classic	93.29	+ LS-Classic	91.70	+ LS-Classic	91.26
	+ LS-Distill	93.47	+ LS-Distill	91.76	+ LS-Distill	91.36
	+ LS-TFIDF	93.65	+ LS-TFIDF	92.10	+ LS-TFIDF	91.73
	+ KWLS	**94.26**	+ KWLS	**92.43**	+ KWLS	**92.55**
Yelp	BERT	69.54	TextCNN	64.38	LSTM	64.03
	+ LS-Classic	69.68	+ LS-Classic	64.53	+ LS-Classic	64.12
	+ LS-Distill	69.70	+ LS-Distill	64.76	+ LS-Distill	64.27
	+ LS-TFIDF	69.81	+ LS-TFIDF	65.02	+ LS-TFIDF	64.92
	+ KWLS	**70.08**	+ KWLS	**65.89**	+ KWLS	**65.68**

Table 2: Experimental results of models with different targets in text classification datasets.

Metric	BERT					TextCNN					LSTM				
	O	C	D	T	K	O	C	D	T	K	O	C	D	T	K
R@1	35.27	37.46	37.62	41.03	**43.87**	33.12	33.18	34.97	36.42	**37.25**	32.43	33.12	34.13	35.41	**36.96**
R@10	48.48	52.62	55.89	61.29	**62.32**	41.99	43.50	43.87	47.81	**52.62**	40.27	41.06	41.28	45.89	**50.63**
R@50	68.21	71.11	72.54	78.45	**79.72**	62.56	62.74	67.46	71.61	**76.56**	60.42	63.09	64.44	69.35	**74.15**
R@100	78.85	79.52	80.50	83.89	**85.28**	71.41	72.01	73.71	76.50	**79.13**	70.26	72.05	72.91	75.32	**77.56**

Table 3: Recall@k on the large-scale retrieval dataset. Numbers are in percentage (%). O, C, D, T, and K represent the original model, LS-Classic, LS-Distill, LS-TFIDF, and KWLS separately.

For large-scale retrieval, we consider the Retrieval Question-Answering (ReQA) benchmark proposed by Ahmad et al. (2019). The dataset we selected is SQuAD, which is a reading comprehension dataset, consisting of questions posed by crowd workers on a set of Wikipedia articles, where the answer to every question is a segment of text, or span, from the corresponding reading passage, or the question might be unanswerable (Rajpurkar et al., 2016). Each entry of this dataset is a tuple (q, a, p), where q is the question, a is the answer span, and p is the evidence passage containing a. Following Ahmad et al. (2019), we split a passage into sentences $p = s_1 s_2 ... s_n$. For a query q, we need to retrieve the correct sentence from a candidate set consisting of sentences of all passages. A query-sentence pair (q, s) is labeled as 1 if s is the sentence containing the corresponding answer span and labeled as 0 otherwise. This problem is more challenging than retrieving the evidence passage only since the larger number of candidates to be retrieved.

For each dataset, we randomly split the train sets into train/dev sets at the ratio of 3:1. The test sets remain unchanged.[2] We apply four-fold cross-validation to do significance tests.

5.2 Baselines

Considering current text mining models can be split into three types, RNN-, CNN-, and Transformer-based models. We incorporate our method with the following three models, which are existing representative models widely used.

LSTM is the most widely used RNN-based deep neural network in text mining tasks (Liu et al., 2016).

TextCNN is the most famous CNN-based text mining baseline proposed by Kim (2014).

BERT is the most representative Transformer-based model in the text mining community (Devlin et al., 2018).

We also compare our methods with three alternative label smoothing methods.

LS-TFIDF is a variant of our method in which the soft label is generated based on TF-IDF.

LS-Classic is the most classical label smoothing method proposed by Szegedy et al. (2016), widely

[2]Note that all LS methods are only used in the training set.

DataSet	Model	Accuracy	Model	Accuracy	Model	Accuracy
AG News *	BERT	76.34	TextCNN	72.62	LSTM	72.06
	+ LS-Classic	76.82	+ LS-Classic	72.76	+ LS-Classic	72.38
	+ LS-Distill	76.92	+ LS-Distill	72.84	+ LS-Distill	72.81
	+ LS-TFIDF	78.12	+ LS-TFIDF	73.61	+ LS-TFIDF	73.87
	+ KWLS	**79.94**	+ KWLS	**74.45**	+ KWLS	**74.04**
DBpedia *	BERT	97.27	TextCNN	95.30	LSTM	95.28
	+ LS-Classic	97.60	+ LS-Classic	95.85	+ LS-Classic	95.44
	+ LS-Distill	98.14	+ LS-Distill	96.04	+ LS-Distill	95.53
	+ LS-TFIDF	98.25	+ LS-TFIDF	97.09	+ LS-TFIDF	96.85
	+ KWLS	**98.41**	+ KWLS	**97.21**	+ KWLS	**97.35**
IMDb *	BERT	50.02	TextCNN	48.73	LSTM	48.31
	+ LS-Classic	54.10	+ LS-Classic	50.59	+ LS-Classic	50.89
	+ LS-Distill	54.60	+ LS-Distill	52.86	+ LS-Distill	52.92
	+ LS-TFIDF	56.31	+ LS-TFIDF	55.49	+ LS-TFIDF	55.14
	+ KWLS	**60.62**	+ KWLS	**57.93**	+ KWLS	**57.81**
Yelp *	BERT	51.69	TextCNN	49.04	LSTM	48.19
	+ LS-Classic	52.40	+ LS-Classic	52.18	+ LS-Classic	51.36
	+ LS-Distill	54.52	+ LS-Distill	53.07	+ LS-Distill	52.48
	+ LS-TFIDF	56.54	+ LS-TFIDF	54.10	+ LS-TFIDF	53.14
	+ KWLS	**58.81**	+ KWLS	**56.91**	+ KWLS	**56.43**

Table 4: Experimental results of models with different targets in unbalanced text classification datasets. "*" denotes the under-sampled thus unbalanced datasets.

used in computer vision and natural language processing.

LS-Distill is another widely used label smoothing method in which the soft label set as predicting scores of the original model (Zhang and Sabuncu, 2020). This method is similar to self-distillation process in born-again networks (Furlanello et al., 2018) and widely used in knowledge distillation (Hinton et al., 2015; Liu et al., 2021b).

5.3 Effectiveness in Text Classification

To explore the effectiveness of our LS method, we conduct experiments to explore whether our method can improve the CNN-based model TextCNN, RNN-based model LSTM, and Transformer-based model BERT. For all of the above text classification tasks, we report the classification accuracy over the test set. Table 2 demonstrates the results,[3] from which we have following five observations:

1. All CNN-, RNN-, and Transformer-based models incorporating soft labels generated by all LS methods as auxiliary targets gain improvements over the original models in all tasks, which verifies the intuition of label smoothing.

2. LS-Classic only gains minor improvements over original models. The reason is that LS-Classic generates soft labels from the hard label in a brute way, which ignores rich knowledge from the instances.

[3]The experiment results in this paper are statistically significant with $p < 0.05$.

DataSet	Business	Technology	Sports	World
c=0	30000	30000	30000	30000
c=1	30000	30000	30000	300
c=2	30000	30000	300	300
c=3	30000	300	300	300
m=300	30000	300	300	300
m=200	30000	200	200	200
m=100	30000	100	100	100
m=50	30000	50	50	50
m=10	30000	10	10	10
m=1	30000	1	1	1
Test Set	1900	1900	1900	1900

Table 5: The experimental setting for investigation on effects of different degrees of unbalance.

3. The improvement of LS-TFIDF over original models shows that TF-IDF weights serve as beneficial prior knowledge to characterize soft labels.

4. LS-Distill also achieves notable enhancements. This observation is consistent with other knowledge distillation works (Hinton et al., 2015). The self-distillation process brings valuable "dark" knowledge (Furlanello et al., 2018) via the generated soft predicting scores even without utilizing term weights information.

5. Our KWLS has clear superiority over LS-Distill and LS-TFIDF among all datasets. Rather than predicting relevance score directly as LS-Distill, KWLS incorporates improved TF-IDF information into supervised signals. Therefore, the final generated soft label integrates explicit prior term weight knowledge, and some "dark" knowledge is produced during training. We believe that is the main reason behind this superiority.

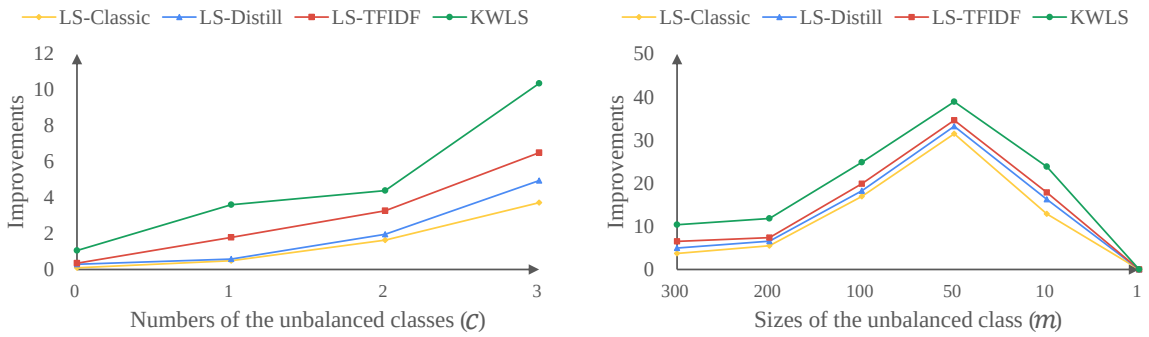

Figure 1: Experimental results of the investigation on the effect of different degrees of unbalance. The x-axis denotes the different settings of unbalance degrees. The y-axis denotes improvements (Accuracy) LS methods gain over the original model.

5.4 Effectiveness in Large-Scale Retrieval

As described in the above section, all LS methods improve the original model on common text classification tasks. To further explore the ability of LS to deal with text datasets, we evaluate LS methods on the more challenging large-scale retrieval task. Since the goal of retrieval is to capture the positives in the top-k results (Liu et al., 2021a), we select **Recall@k** as the evaluation metric.

The experimental results are shown in Table 3, from which we can observe that all LS methods perform exceptionally well for the large-scale retrieval task (especially the LS-TFIDF and KWLS). We can quickly guess the following two reasons for this phenomenon:

1. Note that the data collection process and human annotations of SQuAD are biased towards question-answer pairs with overlapping tokens (Rajpurkar et al., 2016). We can naturally expect that the generated soft label could better characterize query-document relevance degree in the SQuAD dataset due to the capability of term weighting LS methods to identify overlapped highly-representative tokens.

2. Another straightforward guess is that the large-scale retrieval task is highly unbalanced (Retrieve one result in large-scale candidates). For a class without adequate training instances, soft labels generated by LS methods will provide auxiliary supervision information from other categories, which may help the model identify the specific class better.

5.5 Effectiveness on unbalanced Datasets

As shown in the above section, LS methods perform exceptionally well in highly unbalanced large-scale retrieval tasks. To evaluate the effectiveness of LS methods on unbalanced datasets, for each task of text classification mentioned above, we manually remove some samples to simulate the unbalanced scenario. More specially, we undersample each class in the training set with the ratio of 1:100 and keep other classes unchanged. Then we can get n unbalanced training sets where n is the number of classes in the task. The test set remains unchanged (balanced). The average scores across the test set of models trained on n unbalanced training sets are used to evaluate each task. From the evaluation results in Table 4, we can see:

1. The inadequacy and unbalance of training data will significantly hinder the performance, especially for the IMDb dataset with binary labels.

2. Models with LS methods gain significant improvements over the original model on unbalanced datasets. For example, for the unbalanced IMDb dataset, the BERT with KWLS achieves **10.60** improvements in terms of accuracy.

Compared with balanced datasets, LS methods achieve more performance enhancement on unbalanced datasets, which verifies our assumption. For a data-lacking category, soft labels may provide auxiliary supervision information from other categories.

5.6 Effects of Different Degrees of Unbalance

To further explore the ability of LS methods to deal with various unbalance scenarios, we evaluate BERT equipped with LS methods on news classification datasets with multiple unbalanced settings. AG News, composed of 4 classes (Technology, Sports, Business, and World), is selected as the dataset for the experiment. As shown in Table 5, we set the training set unbalanced in two ways as follows:

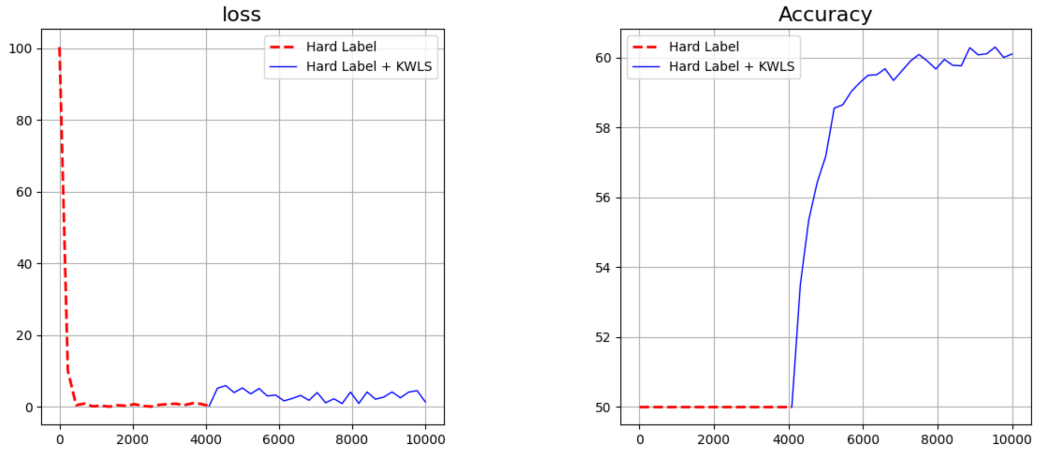

(a) Loss during the training step

(b) Accuracy in validate set during the training step

Figure 2: Loss and Accuracy during the training step in unbalanced IMDb* dataset.

1. To explore the effect of the number of unbalanced classes, we keep the instances of Business news unchanged and under-sample other c classes into the ratio of 1:100, where c is set to 0, 1, 2, and 3 respectively.

2. To explore the effect of the sizes of unbalanced classes, we keep the instances of Business news unchanged and under-sample sizes of other classes in training set to m, which is set to 300, 200, 100, 50, 10, and 1 respectively.

From the evaluation results in Figure 1, we can see that with the increment of c, the models with LS methods gain more improvements over the original models. Since bigger c means a more unbalanced training set, it is not strange for the improvements increment. The trends of relative improvement reveal that soft labels generated by LS methods play a more critical role in a more unbalanced situation. The reason is that the knowledge of label correlation in soft labels helps to discriminate classes with fewer instances.

Based on the most unbalanced setting ($c = 3$), we decrease m to further intensify the degree of unbalance. We can see that the relative improvement in performance brought by LS gradually increases as m becomes smaller until m is less than 50. The reason is that although LS methods can provide auxiliary supervision information from other classes to help models identify the data-lacking class, the supervised signal from the original class is still important, which may degrade dramatically in an extreme case with very few instances.

5.7 Effectiveness of KWLS

To further explore why KWLS can improve models' performances in unbalanced scenarios, we record losses of the hard label and our KWLS during the training step in the unbalanced IMDb* dataset, which is a binary sentiment classification task (positive or negative). We under-sample the positive instances in the training set with the ratio of 1:100 and keep negative instances unchanged. As shown in Figure 2, we first train our model only with the hard label. The loss decreases dramatically since there are few instances in the data-lacking category, and models will easy to fitting supervisory signals of instances in a mini-batch and predict all instances as positive. In the 4000-th step, we incorporate KWLS into the model training. We can see that models with KWLS will obtain extra supervisory signals from other categories, which will help models identify data-lacking categories.

6 Conclusion

We have presented our **K**ey**w**ord-based **L**abel **S**moothing method (**KWLS**) for text mining tasks and demonstrated it's usage and effect on model training. Unlike previous works that focus on universal data mining tasks, our method is explicitly designed for text mining, which incorporates semantic relevance between labels and instances into soft labels. Like other widely-used tricks of text mining, the technical design of KWLS is simple yet effective, making it extremely easy to be applied in different text mining tasks, as shown in experiments. Despite its simplicity, using soft labels generated by KWLS as an auxiliary training

target shows significant superiority in improving model performance, whether the data is balanced or unbalanced. Our codes are released on the Github.[4]

Ackowlegement

Thanks for the support of Xi Wang, Yunlai Hao, and Huihui Bai. We hope they will have a healthy, safe, and happy life. This research was supported by the National Key Research and Development Program of China (No. 2019YFB1405802).

References

Amin Ahmad, Noah Constant, Yinfei Yang, and Daniel Cer. 2019. Reqa: An evaluation for end-to-end answer retrieval models. In *MRQA@EMNLP*, pages 137–146.

Dara Bahri and Heinrich Jiang. 2021. Locally adaptive label smoothing improves predictive churn. In *Proceedings of the 38th International Conference on Machine Learning, ICML 2021, 18-24 July 2021, Virtual Event*, volume 139 of *Proceedings of Machine Learning Research*, pages 532–542. PMLR.

Ivan Chelombiev, Conor J. Houghton, and Cian O'Donnell. 2019. Adaptive estimators show information compression in deep neural networks. In *7th International Conference on Learning Representations, ICLR 2019, New Orleans, LA, USA, May 6-9, 2019*. OpenReview.net.

Gianna M Del Corso, Antonio Gulli, and Francesco Romani. 2005. Ranking a stream of news. In *WWW*, pages 97–106. ACM.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*.

Ibrahim M. Elfadel and John L. Wyatt Jr. 1993. The softmax nonlinearity: Derivation using statistical mechanics and useful properties as a multiterminal analog circuit element. In *Advances in Neural Information Processing Systems 6, [7th NIPS Conference, Denver, Colorado, USA, 1993]*, pages 882–887. Morgan Kaufmann.

Tommaso Furlanello, Zachary Chase Lipton, Michael Tschannen, Laurent Itti, and Anima Anandkumar. 2018. Born-again neural networks. In *ICML*, volume 80 of *Proceedings of Machine Learning Research*, pages 1602–1611. PMLR.

Antonio Gulli. 2005. The anatomy of a news search engine. In *Special interest tracks and posters of the 14th international conference on World Wide Web*, pages 880–881. ACM.

[4] https://github.com/PeiYangLiu/KWLS

Geoffrey Hinton, Oriol Vinyals, and Jeff Dean. 2015. Distilling the knowledge in a neural network. *stat*, 1050:9.

Seungbum Hong, Jihun Yoon, Min-Kook Choi, and Junmo Kim. 2022. Self-supervised knowledge transfer via loosely supervised auxiliary tasks. In *IEEE/CVF Winter Conference on Applications of Computer Vision, WACV 2022, Waikoloa, HI, USA, January 3-8, 2022*, pages 2947–2956. IEEE.

Yoon Kim. 2014. Convolutional neural networks for sentence classification. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing, EMNLP 2014, October 25-29, 2014, Doha, Qatar, A meeting of SIGDAT, a Special Interest Group of the ACL*, pages 1746–1751. ACL.

Jens Lehmann, Robert Isele, Max Jakob, Anja Jentzsch, Dimitris Kontokostas, Pablo N Mendes, Sebastian Hellmann, Mohamed Morsey, Patrick Van Kleef, Sören Auer, et al. 2015. Dbpedia–a large-scale, multilingual knowledge base extracted from wikipedia. *Semantic Web*, 6(2):167–195.

Weizhi Li, Gautam Dasarathy, and Visar Berisha. 2020. Regularization via structural label smoothing. In *The 23rd International Conference on Artificial Intelligence and Statistics, AISTATS 2020, 26-28 August 2020, Online [Palermo, Sicily, Italy]*, volume 108 of *Proceedings of Machine Learning Research*, pages 1453–1463. PMLR.

Peiyang Liu, Sen Wang, Xi Wang, Wei Ye, and Shikun Zhang. 2021a. QuadrupletBERT: An efficient model for embedding-based large-scale retrieval. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 3734–3739, Online. Association for Computational Linguistics.

Peiyang Liu, Xi Wang, Lin Wang, Wei Ye, Xiangyu Xi, and Shikun Zhang. 2021b. Distilling knowledge from BERT into simple fully connected neural networks for efficient vertical retrieval. In *CIKM '21: The 30th ACM International Conference on Information and Knowledge Management, Virtual Event, Queensland, Australia, November 1 - 5, 2021*, pages 3965–3975. ACM.

Peiyang Liu, Xi Wang, Sen Wang, Wei Ye, Xiangyu Xi, and Shikun Zhang. 2021c. Improving embedding-based large-scale retrieval via label enhancement. In *Findings of the Association for Computational Linguistics: EMNLP 2021, Virtual Event / Punta Cana, Dominican Republic, 16-20 November, 2021*, pages 133–142. Association for Computational Linguistics.

Pengfei Liu, Xipeng Qiu, and Xuanjing Huang. 2016. Recurrent neural network for text classification with multi-task learning. In *Proceedings of the Twenty-Fifth International Joint Conference on Artificial Intelligence, IJCAI 2016, New York, NY, USA, 9-15 July 2016*, pages 2873–2879. IJCAI/AAAI Press.

Andrew L Maas, Raymond E Daly, Peter T Pham, Dan Huang, Andrew Y Ng, and Christopher Potts. 2011. Learning word vectors for sentiment analysis. In *ACL*, pages 142–150. Association for Computational Linguistics.

Rafael Müller, Simon Kornblith, and Geoffrey E. Hinton. 2019. When does label smoothing help? In *Advances in Neural Information Processing Systems 32: Annual Conference on Neural Information Processing Systems 2019, NeurIPS 2019, December 8-14, 2019, Vancouver, BC, Canada*, pages 4696–4705.

Chuong H. Nguyen, Thuy C. Nguyen, Tuan N. Tang, and Nam L. H. Phan. 2022. Improving object detection by label assignment distillation. In *IEEE/CVF Winter Conference on Applications of Computer Vision, WACV 2022, Waikoloa, HI, USA, January 3-8, 2022*, pages 1322–1331. IEEE.

Pranav Rajpurkar, Jian Zhang, Konstantin Lopyrev, and Percy Liang. 2016. Squad: 100, 000+ questions for machine comprehension of text. In *EMNLP*, pages 2383–2392.

Stephen Robertson, Hugo Zaragoza, et al. 2009. The probabilistic relevance framework: Bm25 and beyond. *Foundations and Trends® in Information Retrieval*, 3(4):333–389.

Karen Spärck Jones. 1972. A statistical interpretation of term specificity and its application in retrieval. *Journal of documentation*, 28(1):11–21.

Karen Spärck Jones. 2004. Idf term weighting and ir research lessons. *Journal of documentation*, 60(5):521–523.

Christian Szegedy, Vincent Vanhoucke, Sergey Ioffe, Jonathon Shlens, and Zbigniew Wojna. 2016. Rethinking the inception architecture for computer vision. In *2016 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2016, Las Vegas, NV, USA, June 27-30, 2016*, pages 2818–2826. IEEE Computer Society.

Yichen Wu, Jun Shu, Qi Xie, Qian Zhao, and Deyu Meng. 2021. Learning to purify noisy labels via meta soft label corrector. In *Thirty-Fifth AAAI Conference on Artificial Intelligence, AAAI 2021, Thirty-Third Conference on Innovative Applications of Artificial Intelligence, IAAI 2021, The Eleventh Symposium on Educational Advances in Artificial Intelligence, EAAI 2021, Virtual Event, February 2-9, 2021*, pages 10388–10396. AAAI Press.

Xiang Zhang, Junbo Jake Zhao, and Yann LeCun. 2015. Character-level convolutional networks for text classification. In *Advances in Neural Information Processing Systems 28: Annual Conference on Neural Information Processing Systems 2015, December 7-12, 2015, Montreal, Quebec, Canada*, pages 649–657.

Zhilu Zhang and Mert R. Sabuncu. 2020. Self-distillation as instance-specific label smoothing. In *Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020, December 6-12, 2020, virtual*.

Diverse Multi-Answer Retrieval with Determinantal Point Processes

Poojitha Nandigam*, **Nikhil Rayaprolu***, **Manish Shrivastava**
Language Technologies Research Centre (LTRC)
International Institute of Information Technology, Hyderabad, India
poojitha.nandigam@research.iiit.ac.in, nikhil.rayaprolu@students.iiit.ac.in,
m.shrivastava@iiit.ac.in

Abstract

Often questions provided to open-domain question answering systems are ambiguous. Traditional QA systems that provide a single answer are incapable of answering ambiguous questions since the question may be interpreted in several ways and may have multiple distinct answers. In this paper, we address multi-answer retrieval which entails retrieving passages that can capture majority of the diverse answers to the question. We propose a re-ranking based approach using Determinantal point processes utilizing BERT as kernels. Our method jointly considers query-passage relevance and passage-passage correlation to retrieve passages that are both query-relevant and diverse. Results demonstrate that our re-ranking technique outperforms state-of-the-art method on the AmbigQA dataset.

1 Introduction

The objective of open-domain question answering is to provide answers to queries utilising a large collection of documents from the World Wide Web, Wikipedia etc. More than 50% of questions present in a widely used open-domain QA dataset (Natural Questions Kwiatkowski et al. (2019)) comprise of ambiguous questions (Min et al. (2020)). Questions that are ambiguous may be interpreted in a number of ways and as a result, they need various answers. In this paper, we concentrate on questions with multiple distinct answers.

Open domain question-answering systems are designed to generate answers from several data sources. Since similar information can be present across multiple data sources, it introduces a significant amount of redundancy. Traditional open-domain QA (Chen et al. (2017)) systems comprise of a Retriever, which retrieves passages relevant to the question. A passage retriever is primarily concerned with retrieving passages that are relevant to the query, and it does not address redundancy in the passages during retrieval. To be able to produce diverse answers to the question, the passages retrieved must be both relevant to the question and distinct from one another. After the retrieval stage, we introduce a novel re-ranking approach to handle redundant passages. As a result, the re-ranked passages would capture most of the diverse answers to the question. In this paper, we investigate the multi-answer retrieval task, which entails retrieving passages that can cover the distinct answers.

Re-ranking methods have been employed previously to improve the question answering accuracy significantly(Wang et al. (2019);Nogueira and Cho (2019); Min et al. (2021); Clark and Gardner (2017)). Min et al. (2021) tackles diverse multi-answer retrieval by proposing a re-ranker based on an auto-regressive framework in which each passage selected is dependent on the passages chosen at a previous time step.

Determinantal Point Processes (DPP) (Kulesza and Taskar (2012)) are probabilistic models that are effective at identifying diverse subsets of elements from a collection while preserving quality. DPP methods have proven effective in natural language processing tasks where there is a need for diverseness. Cho et al. (2019), Li et al. (2019), and Cho et al. (2020), Sharghi et al. (2017) have used DPPs to perform summarization by choosing salient but also diverse items to be included in the summaries. In this paper, we propose an unsupervised re-ranking technique for multi-answer retrieval utilising Determinantal point processes and BERT to model the kernels.

Our contributions can be summarized as follows:
1) We propose a re-ranking method based on determinantal point processes that focuses on diverse passage retrieval.
2) Since our approach is unsupervised, our method does not require a large amount of data unlike prior re-ranking methods(Min et al., 2021). Instead, we

*Equal contribution

Proceedings of the 29th International Conference on Computational Linguistics, pages 2220–2225
October 12–17, 2022.

rely on DPP to identify the most relevant passages to the question that are distinct from one another.

3) We demonstrate that our technique outperforms the state-of-the-art method on the AmbigQA dataset using MRECALL @ k metrics.

2 Related Work

Many open domain question answering systems(Chen et al. (2017); Yang et al. (2019); Izacard and Grave (2021); Guu et al. (2020); Lee et al. (2019)) adopt the *retriever-reader* method by retrieving the relevant documents and later applying neural techniques to predict the answer. The *retriever-reader* method was first proposed by Chen et al. (2017). DrQA(Chen et al. (2017)) uses Wikipedia as knowledge source and employs a sparse retrieval method using TF-IDF and a recurrent neural network to identify the answer spans. While Yang et al. (2019) adopts Anserini retriever(Yang et al. (2017)) using BM25 as the ranking function and BERT model (Devlin et al. (2018)) as the reader. Sparse retrieval based methods, such as TF-IDF and BM25, face challenges when retrieving relevant passages that do not match the question's exact terms. Dense retrieval-based approaches, on the other hand, overcome this problem by mapping each word into a vector space in which words with similar meanings tend to be closer together. ORQA (Lee et al. (2019)) and DPR (Karpukhin et al. (2020)) employ a question and passage encoder based on BERT and compute a relevance score. Using this relevance score, the retriever retrieves the most relevant documents from the corpus.

3 Determinantal Point Processes for Re-ranking

Re-ranker acts as a filter to pick a limited number of passages that can be used as input to generate answers to the questions. We formulate the task of passage re-ranking as a subset selection problem. Our objective is to choose a subset of passages (Y) of size k from the ground set $\mathcal{Y}$ comprising N passages that covers all of the answers to a given question q . DPP models a distribution on all the subsets of the ground set $\mathcal{Y}$ jointly considering the quality and diversity. A subset Y is drawn according to the probability distribution P.

$$P(Y; L) \propto det(L_Y) \tag{1}$$

$$P(Y; L) = \frac{\det(L_Y)}{\det(L + I)} \tag{2}$$

where I is the identity matrix, $L \in \mathbb{R}^{\mathsf{N}\times\mathsf{N}}$ is a positive semi-definite matrix referred as *L-ensemble*, $\det(.)$ denotes the determinant of a matrix, and L_y is the submatrix of L indexed by items in Y. L matrix jointly considers query-passage relevance as well as passage-passage correlation through eq. 3.

$$L_{ij} = Q(i, q) \cdot S(i, j) \cdot Q(j, q) \tag{3}$$

DPP focuses on two measures - quality and similarity (Fig 1). Quality score $Q(i, q)$ measures how salient the passage i is and whether it contains an answer to the question q. Similarity score $S(i, j)$ is computed between two passages i and j to incorporate diversity in the passages. DPP assigns a probability to a set Y proportional to the determinant of *L-ensemble* which may be interpreted geometrically as the volume of the parallepiped covered by the quality and similarity measures (Kulesza and Taskar (2012)). A diverse passage subset occupies more volume than a subset of similar passages, therefore DPP assigns higher probability to diverse and relevant passages rather than the most relevant and similar passages. If passages are relevant and diverse, then the passages can cover multiple distinct answers to the question.

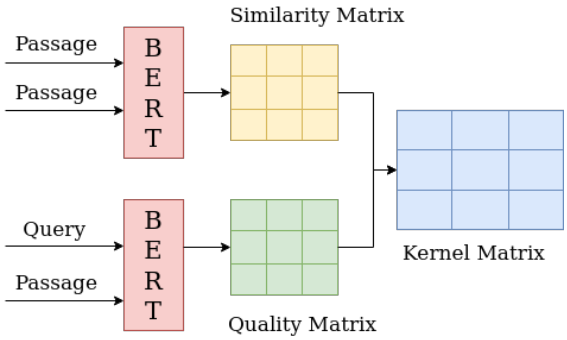

Figure 1: An overview of the proposed re-ranking method using DPP. A similarity score between the passages and a quality score between the question and passage are computed. These two scores are utilised to construct the DPP kernel matrix.

3.1 BERT for Similarity matrix

To compute the similarity scores, we use a pre-trained BERT model (Devlin et al. (2018); Reimers and Gurevych (2019)) to generate embeddings for every passage. The model takes the passage as

input and produces a 768 dimensional dense embedding. We use these embeddings to calculate the cosine similarity of all passages and compute a similarity matrix $S \in \mathbb{R}^{N \times N}$ for the whole passage set. All the values in the similarity matrix lie in the range of $[0, 1]$. If passages i and j are similar, the similarity value $S(i, j)$ lies closer to 1, if they are distinct, the value lies closer to 0, and if $i = j$, $S(i, j)$ becomes equal to 1.

$$S(i, j) = cosine\_sim(BERT_A(i), BERT_A(j)) \tag{4}$$

3.2 BERT for Quality matrix

We use a pretrained BERT model trained on MS MARCO (Nguyen et al. (2016)) for computing the Quality matrix. The model takes in a query and a passage and generates the quality score. Higher quality score indicates that the passage is most relevant to the query and therefore most likely to answer the query. Unlike for computing similarity matrix, we do not perform cosine similarity over the model's outputs to produce a score, instead, we use a BERT encoder that concatenates both query and passage and generates a score. The quality matrix $Q \in \mathbb{R}^{N \times N}$ is computed by performing the matrix multiplication of the scores $(N \times 1)$ with it's transpose resulting in $N \times N$ dimensioned vector . These quality scores are then normalized to lie between $[0, 1]$.

$$Q(i, j) = Norm(BERT_B([i; j])) \tag{5}$$

3.3 Sampling

Traditional DPP sampling algorithms have higher run-time complexity when L matrix is large. We apply an efficient sampling technique - BFGMInference (Li et al. (2019);Chen et al. (2018)). BFGMInference approximates a greedy approach to select a passage that maximizes the $det(L_Y)$ and adds it to the passage subset.

$$\begin{aligned} f(Y) &= \log\det(L_Y) \\ k &= \arg\max_{i \in \mathcal{Y} \setminus Y} f(Y \cup \{i\}) - f(Y) \end{aligned} \tag{6}$$

4 Experiments

In this section, we discuss about the passage retrieval method, the dataset we used in our experiments, the evaluation metric, and the results of our experiments.

4.1 Passage retrieval

Wikipedia is utilised as the corpus for retrieving passages for the questions. Each Wikipedia article is broken into multiple passages containing the same number of words. We retrieve query-relevant passages from Wikipedia using the Dense Passage Retriever (DPR) (Karpukhin et al. (2020); Lin et al. (2021)). DPR computes encodings for all passages extracted from the Wikipedia corpus and builds an index. The inner product of the query and passage encodings is used to determine the similarity scores between them. Passages with the highest scores are the ones that are most relevant to the query, and these passages are subsequently sent into the re-ranker as input.

4.2 Dataset

We evaluated our method on an open-domain question-answering dataset AmbigQA (Min et al. (2020)), which contains multiple-answer questions. The dataset was created from an anonymised collection of Google search queries submitted by users seeking information on different subjects. It consists of 14,042 question-answer pairs derived from the Natural questions dataset (Kwiatkowski et al. (2019)) and is split into train, validation, and test sets. Train set consists of 10,036 question-answer pairs, validation set consists of 2,002 examples, and test set consists of 4,042 examples.

4.3 Evaluation metric

MRECALL @ k (Min et al. (2021)) is used to evaluate the re-ranking of passages for questions with diverse answers. As per this metric, if a query has n answers, the k passages that are retrieved must cover all of the answers. If $n <= k$, all answers must be covered; if $n > k$, the passages retrieved must contain at least k answers. A retrieval is deemed successful if the passages retrieved include all or at least k of the answers to the query.

4.4 Results

We compare our technique to a few additional baselines, all of which were assessed using the MRECALL @ k metric on the AmbigQA dataset.

- **DPR**$^+$ Min et al. (2021) integrates REALM (Guu et al. (2020)) with DPR (Karpukhin et al. (2020)). As described in Section 4.1, DPR is a dense retrieval based technique that utilizes the FAISS library to retrieve the relevant documents. Encoders for the query and passage

Models	Top5	Top 10
	AmbigQA-Dev	AmbigQA-Dev
DPR$^+$(Min et al. (2021))	55.2/36.3	59.3/39.6
DPR$^+$ + Nogueira and Cho (2019)	63.4/43.1	65.8/46.4
JPR(Min et al. (2021))	64.8/45.2	67.1/48.2
QRR	62.0/42.3	70.8/57.6
DPP-R	**66.9/53.5**	**72.8/58.8**

Table 1: Performance of various models on AmbigQA dataset. Each row contains the MRECALL @ k metrics for single answer retrieval and multi-answer retrieval respectively.

are initialized using REALM and the DPR training method is followed.

- **DPR$^+$ + Nogueira and Cho (2019)** employs DPR$^+$ for the first stage of retrieval and the re-ranking method in Nogueira and Cho (2019) is applied on the retrieved passages.
- **JPR** Min et al. (2021) employs DPR$^+$ as the initial ranker and an auto-regressive framework is adopted as a re-ranker to generate diverse passages.
- **Query Relevance Re-ranking(QRR)** In this method, we first calculate the quality scores for each passage (described in section 3.2) and then we sort the passages based on these scores to pick the top-k passages. Here, similarity among the passages is not considered.
- **DPP-R** We employ our method described in section 3 to retrieve highly diverse and relevant passages.

We calculate the performance of diverse multi passage retrieval using the MRECALL @ k measure described in section 4.3. Evaluation on the AmbigQA dataset demonstrates that our approach outperforms existing re-ranking techniques. Our technique requires no human annotations for multi passage re-ranking while outperforming existing methods, as shown in Table 1. DPP is modelled to select a subset of high-quality and diverse passages, which contributes to the success of our method for this task. Experiments demonstrate that the DPP-based technique achieves promising results for retrieving passages containing diverse answers.

5 Discussion

Impact on QA system's performance: An Open domain question answering system's pipeline consists of three stages. 1) Retrieval 2) Re-ranking followed by 3) Answer extraction. Improvements in any of these stages significantly improve the overall system's ability to answer a question. Nogueira and Cho (2019), Min et al. (2021) have shown that the use of a re-ranker has led to end-to-end QA improvements. Based on the results presented in Table 1, the DPP method enhances re-ranking for both single and multi-answer questions. We believe that this improvement in re-ranking will also improve the overall performance of the end-to-end QA system.

Impact of diversity: DPP-R and JPR retrieve diverse passages utilising DPP and auto-regressive framework, respectively. Other approaches like QRR, retrieve just passages that are relevant to the query and do not tackle passage redundancy. We observe that our approach using DPP performs better than the QRR method. In order to re-rank, QRR simply considers how relevant a passage is to the query, and it retrieves the top-k passages with the highest relevance score for a given query. On the other hand, DPP-R takes into account how relevant the passage is to the query and also how similar passages are to each other, in order to eliminate redundant passages leading to diversity in the retrieved passages. DPP-R and JPR outperform other methods that do not emphasise diversity in multi-answer retrieval. For single answer retrieval, DPP-R and JPR have fared better than other methods, with the minor exception that QRR beats JPR in top-10 re-ranking. This demonstrates that diversity is an important aspect to consider during the re-ranking stage.

6 Conclusion

In this paper, we propose a DPP-based approach to improve the diverseness of the retrieved passages. We compare our method to the state-of-the-art method and outperform it by 3% (top 5), 8% (top 10) for single-answer questions, and 18% (top5) and 21% (top10) for multi-answer retrieval

on AmbigQA dataset.

References

Danqi Chen, Adam Fisch, Jason Weston, and Antoine Bordes. 2017. Reading Wikipedia to answer open-domain questions. In *ACL 2017 - 55th Annual Meeting of the Association for Computational Linguistics, Proceedings of the Conference (Long Papers)*, volume 1, pages 1870–1879.

Laming Chen, Guoxin Zhang, and Eric Zhou. 2018. Fast greedy map inference for determinantal point process to improve recommendation diversity. *Advances in Neural Information Processing Systems*, 31.

Sangwoo Cho, Logan Lebanoff, Hassan Foroosh, and Fei Liu. 2020. Improving the similarity measure of determinantal point processes for extractive multi-document summarization. *ACL 2019 - 57th Annual Meeting of the Association for Computational Linguistics, Proceedings of the Conference*, pages 1027–1038.

Sangwoo Cho, Chen Li, Dong Yu, Hassan Foroosh, and Fei Liu. 2019. Multi-document summarization with determinantal point processes and contextualized representations. *arXiv*.

Christopher Clark and Matt Gardner. 2017. Simple and effective multi-paragraph reading comprehension. *arXiv preprint arXiv:1710.10723*.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*.

Kelvin Guu, Kenton Lee, Zora Tung, Panupong Pasupat, and Ming Wei Chang. 2020. REALM: Retrieval-Augmented language model pre-training. In *37th International Conference on Machine Learning, ICML 2020*, volume PartF16814, pages 3887–3896.

Gautier Izacard and Edouard Grave. 2021. Leveraging passage retrieval with generative models for open domain question answering. In *EACL 2021 - 16th Conference of the European Chapter of the Association for Computational Linguistics, Proceedings of the Conference*, pages 874–880.

Vladimir Karpukhin, Barlas Oğuz, Sewon Min, Patrick Lewis, Ledell Wu, Sergey Edunov, Danqi Chen, and Wen tau Yih. 2020. Dense passage retrieval for open-domain question answering.

Alex Kulesza and Ben Taskar. 2012. Determinantal point processes for machine learning.

Tom Kwiatkowski, Jennimaria Palomaki, Olivia Redfield, Michael Collins, Ankur Parikh, Chris Alberti, Danielle Epstein, Illia Polosukhin, Matthew Kelcey, Jacob Devlin, Kenton Lee, Kristina N. Toutanova, Llion Jones, Ming-Wei Chang, Andrew Dai, Jakob Uszkoreit, Quoc Le, and Slav Petrov. 2019. Natural questions: a benchmark for question answering research. *Transactions of the Association of Computational Linguistics*.

Kenton Lee, Ming-Wei Chang, and Kristina Toutanova. 2019. Latent retrieval for weakly supervised open domain question answering. *arXiv preprint arXiv:1906.00300*.

Lei Li, Wei Liu, Marina Litvak, Natalia Vanetik, and Zuying Huang. 2019. In conclusion not repetition: Comprehensive abstractive summarization with diversified attention based on determinantal point processes. *CoNLL 2019 - 23rd Conference on Computational Natural Language Learning, Proceedings of the Conference*, pages 822–832.

Jimmy Lin, Xueguang Ma, Sheng-Chieh Lin, Jheng-Hong Yang, Ronak Pradeep, and Rodrigo Nogueira. 2021. Pyserini: An easy-to-use python toolkit to support replicable ir research with sparse and dense representations. *arXiv preprint arXiv:2102.10073*.

Sewon Min, Kenton Lee, Ming-Wei Chang, Kristina Toutanova, and Hannaneh Hajishirzi. 2021. Joint passage ranking for diverse multi-answer retrieval. *arXiv preprint arXiv:2104.08445*.

Sewon Min, Julian Michael, Hannaneh Hajishirzi, and Luke Zettlemoyer. 2020. Ambigqa: Answering ambiguous open-domain questions.

Tri Nguyen, Mir Rosenberg, Xia Song, Jianfeng Gao, Saurabh Tiwary, Rangan Majumder, and Li Deng. 2016. Ms marco: A human generated machine reading comprehension dataset. In *CoCo@ NIPS*.

Rodrigo Nogueira and Kyunghyun Cho. 2019. Passage re-ranking with bert. *arXiv preprint arXiv:1901.04085*.

Nils Reimers and Iryna Gurevych. 2019. Sentence-bert: Sentence embeddings using siamese bert-networks. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics.

Aidean Sharghi, Jacob S Laurel, and Boqing Gong. 2017. Query-focused video summarization: Dataset, evaluation, and a memory network based approach. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 4788–4797.

Zhiguo Wang, Patrick Ng, Xiaofei Ma, Ramesh Nallapati, and Bing Xiang. 2019. Multi-passage bert: A globally normalized bert model for open-domain question answering. *arXiv preprint arXiv:1908.08167*.

Peilin Yang, Hui Fang, and Jimmy Lin. 2017. Anserini: Enabling the use of lucene for information retrieval research. In *Proceedings of the 40th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 1253–1256.

Wei Yang, Yuqing Xie, Aileen Lin, Xingyu Li, Luchen Tan, Kun Xiong, Ming Li, and Jimmy Lin. 2019. End-to-end open-domain question answering with BERTserini. In *NAACL HLT 2019 - 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies - Proceedings of the Demonstrations Session*, pages 72–77.

Improving Deep Embedded Clustering via Learning Cluster-level Representations

Qing Yin[21], Zhihua Wang[34], Yunya Song[5], Yida Xu[6], Shuai Niu[2], Liang Bai[7], Yike Guo[2] and Xian Yang[1*]

[1] Alliance Manchester Business School, The University of Manchester
[2] Department of Computer Science, Hong Kong Baptist University
[3] Shanghai Institute for Advanced Study of Zhejiang University
[4] Shanghai AI Laboratory
[5] Department of Journalism, Hong Kong Baptist University
[6] Department of Mathematics, Hong Kong Baptist University
[7] School of Computer and Information Technology, Shanxi University

Abstract

Driven by recent advances in neural networks, various Deep Embedding Clustering (DEC) based short text clustering models are being developed. In these works, latent representation learning and text clustering are performed simultaneously. Although these methods are becoming increasingly popular, they use pure cluster-oriented objectives, which can produce meaningless representations. To alleviate this problem, several improvements have been developed to introduce additional learning objectives in the clustering process, such as models based on contrastive learning. However, existing efforts rely heavily on learning meaningful representations at the instance level. They have limited focus on learning global representations, which are necessary to capture the overall data structure at the cluster level. In this paper, we propose a novel DEC model, which we named the deep embedded clustering model with cluster-level representation learning (DEC-CRL) to jointly learn cluster and instance level representations. Here, we extend the embedded topic modelling approach to introduce reconstruction constraints to help learn cluster-level representations. Experimental results on real-world short text datasets demonstrate that our model produces meaningful clusters.

1 Introduction

Short Text Clustering has gained increasing attention in many real-world applications, such as event discovery (Atefeh and Khreich, 2015), spam detection (Wu and Liu, 2018), and sentiment analysis (Paltoglou and Thelwall, 2012). Unlike long texts, which can be represented as, for instance, term frequency inverse-document-frequency (TF-IDF) vectors in the clustering task, short texts cannot be encoded in the same manner. This is because the vector representation of short texts is highly sparse, making it difficult to measure the similarity between two sets of short texts (Xu et al., 2017). With this observation, deep clustering methods are being developed to encode raw short texts into latent representational space using neural networks and to detect clusters based on the learned representations. Deep clustering methods generally fall into two categories: the two-stage methods and the deep embedded clustering (DEC) methods. The two-stage methods (Zakaria et al., 2012; Tian et al., 2014; Vincent et al., 2010) assign data samples to different clusters after latent representations are learned and fixed, while the DEC methods (Xie et al., 2016) simultaneously learn latent representations and discover clusters via end-to-end training. Different from the two-stage methods, the DEC methods explicitly define the cluster-oriented loss to jointly map raw data into latent representations and acquire cluster assignments.

As discussed in (Jiang et al., 2016; Xie et al., 2016; Aljalbout et al., 2018), significant improvement in clustering performance can be achieved by learning better representations of texts. However, it has been increasingly found that purely cluster-oriented loss driven methods tend to generate meaningless representations (Guo et al., 2017). The semantic meaning of raw data cannot be preserved in the latent space, which would, in turn, deteriorate the performance of clustering. To tackle this problem, sequence-to-sequence (seq2seq) based reconstruction models have been widely used to learn general representations from texts in an unsupervised manner. For example, (Kiros et al., 2015) generated the text representations by predicting the context sentences of a given sentence. The work in (Brahma, 2018) learned text representations by predicting

*This is the corresponding author.

Proceedings of the 29th International Conference on Computational Linguistics, pages 2226–2236
October 12–17, 2022.

multiple future sentences based on the seq2seq model. Recently, instance-wise contrastive learning has achieved remarkable success in representation learning by adopting the contrastive loss along with the cluster-oriented loss (Li et al., 2021; Tsai et al., 2020; Van Gansbeke et al., 2020; Zhang et al., 2021). However, the aforementioned methods focus on optimizing the representation at the instance level. For example, the contrastive learning-based methods heavily rely on the instance discrimination (Li et al., 2021) such that their learning objectives (i.e. instance-wise loss) do not perfectly align with the ultimate goal of clustering. In the clustering task, the final clustering performance heavily relies on learning representations which are capable of reflecting the overall semantic structure of data. Instance-level representation learning methods cannot guarantee that the structure of data can be easily obtained through clustering.

In this paper, we aim to develop a novel DEC method which learns cluster-level as well as instance-level representations to better capture the semantic data structure for clustering. In our approach, the cluster-level representations are defined as the representations of cluster centres. Different from the SCCL model (Zhang et al., 2021) which learns centre representations without imposing any direct constraints, we adopt the reconstruction constraints (Ma et al., 2019) to encode the whole set of raw texts into latent representations of cluster centres and then use these centre representations to reconstruct the text data. As reconstructing the whole dataset from a limited number of cluster-level representations is quite challenging, we designed a cluster-level representation learning (CRL) module to help the representations of cluster centres participate in the process of reconstructing input instances. More specifically, we extend the idea of embedded topic modelling (ETM) (Dieng et al., 2020) to reconstruct words from the representations of topics in latent space. The representations of cluster centres will be integrated into the topic representations, which will then be learned by optimizing both ETM guided reconstruction and clustering objectives.

Our proposed model, named as deep embedded clustering model with cluster-level representation learning (DECCRL), consists of three modules: an instance-level encoding module that maps the original data inputs into latent representations; a cluster selection module that generates cluster labels; and a CRL model that learns the cluster-level representations through reconstruction. Our main contributions are summarized as follows:

- We develop a novel deep embedded clustering method to learn cluster-level as well as instance-level representations to better capture the data structure.
- We extend the idea of embedded topic modelling to impose reconstruction constraints to the cluster-level representations.
- The experimental results show that our method achieves the best clustering performance compared with current state-of-the-art short text clustering methods.

In the remainder of this paper, we first summarize related works in Section 2. We formulate the problem and explain our DECCRL model in Section 3. The experiments are introduced in Sections 4 and 5 and the paper is concluded in Section 6.

2 Related Work

2.1 Deep Embedded Clustering

Deep clustering (Xie et al., 2016; Wu et al., 2019) applies deep neural networks to transform raw inputs into latent representations, based on which clustering is performed. Traditional approaches derive latent representations first, and then clusters are detected (Zakaria et al., 2012; Tian et al., 2014; Vincent et al., 2010). However, latent representations learned by these approaches are not cluster-oriented in that they are learned before hand. The deep embedded clustering (DEC) methods are then developed to simultaneously generate latent representations and cluster assignments through the end-to-end training (Dosovitskiy et al., 2015; Caron et al., 2018; Asano et al., 2019; Ghasedi et al., 2019; Yang et al., 2020).

However, methods that purely depend on the cluster-oriented loss cannot well preserve the local structure of raw data and are likely to generate corrupted latent space (Guo et al., 2017). To address the above-mentioned problem, researchers recently introduced extra reconstruction modules along with the clustering model. For example, to cluster images, Jiang et al. (2016); Madiraju (2018); Yang et al. (2019) adopted auto-encoders to learn latent representations and simultaneously perform clustering using the latent representations from auto-

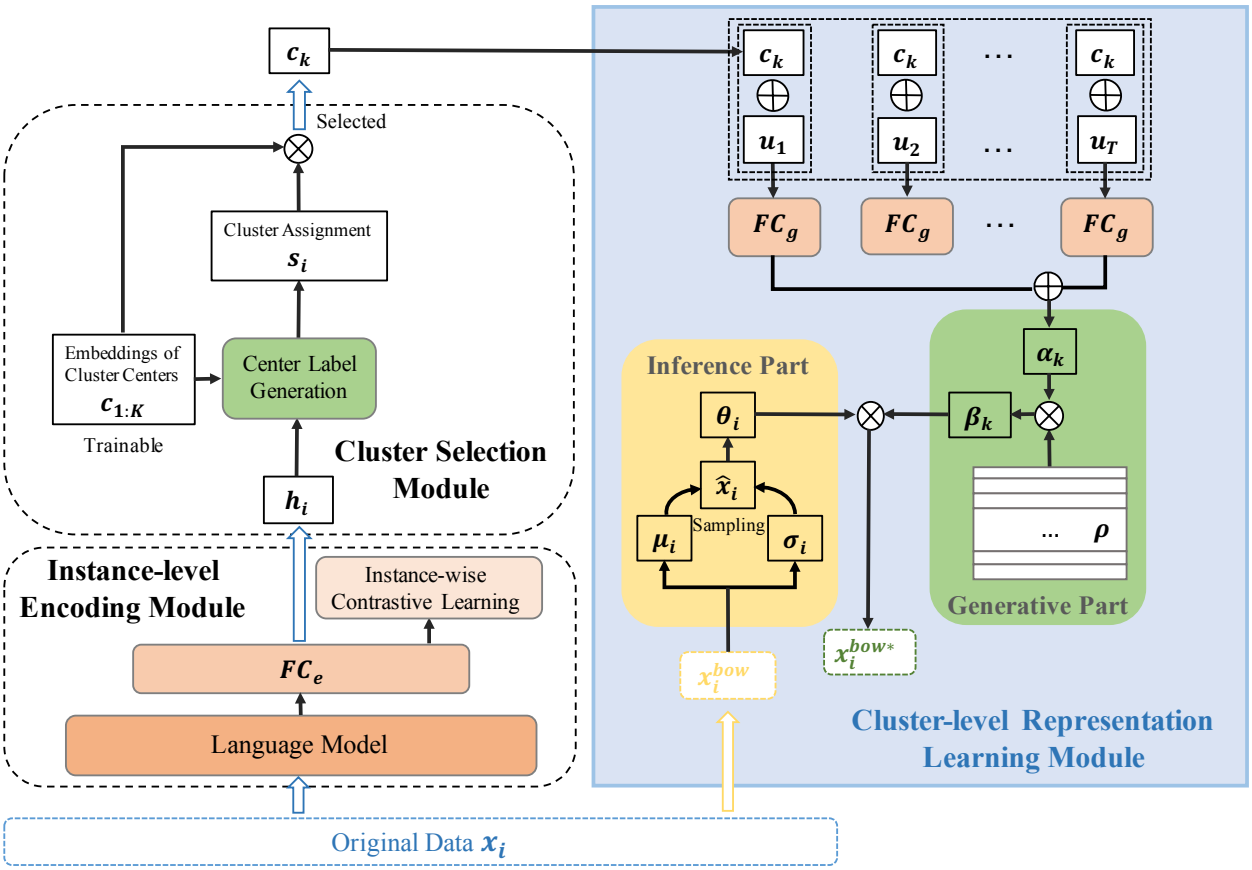

Figure 1: The overview of the proposed deep embedded clustering model with cluster-level representation learning (DECCRL). It contains three components: an instance-level encoding module, a cluster selection module and a cluster-level representation learning module based on the embedded topic modelling. Some key variables are described as follows: $\boldsymbol{c}_{1:K}$ contains embeddings of cluster centers; $\boldsymbol{\alpha}_k$ refers to embeddings of topics in the k-th cluster; $\boldsymbol{\theta}_i$ represents topic proportions of the i-th input text; $\boldsymbol{\rho}$ contains word embeddings of the vocabulary; $\boldsymbol{\beta}_k$ is the distribution over the vocabulary for topics within the k-th cluster.

encoders. To cluster time-series data, Ma et al. (2019) leveraged a seq2seq model to guide the generation of latent representations. To cluster text data, Zhao et al. (2021) utilized the idea of data reconstruction to reconstruct data in the latent space only other than the original space. In the same time, there are many seq2seq models have been used to assist text data clustering like (Kiros et al., 2015) generated the text representations by predicting the context sentences of a given sentence and the work in (Brahma, 2018) learned text representations by predicting multiple future sentences based on the seq2seq model.

2.2 Neural Topic Modelling in Various NLP tasks

With the development of neural networks, there has been a surge of methods that seek to combine deep neural networks with probabilistic topic models (Srivastava and Sutton, 2017; Cong et al., 2017; Zhang et al., 2018). Most of these methods used amortized inference and variational auto-encoder to reduce the dimension of the input data (Rezende et al., 2014; Dieng et al., 2020). For example, ETM (Dieng et al., 2020) is a neural topic model that uses word embeddings from Word2Vec (Mikolov et al., 2013). Neural topic modelling is not only used to learn hidden topics from a collection of texts, but has also been increasingly used to assist other NLP tasks. For example, See et al. (2017); Ailem et al. (2019); Wang et al. (2020) combined seq2seq models with topic models in the abstract generation task. (Dieng et al., 2016) incorporated topic modelling with the recurrent neural network to capture the long-range dependencies. Zeng et al. (2018); Wang and Yang (2020) integrated NTM with a memory network for short text classification. Tang et al. (2019); Wang et al. (2019) used neural topic modelling to assist text generation. Our work aims to improve clustering by incorporating ETM.

3 Method

The overview of the proposed deep embedded clustering model with cluster-level representation learning (DECCRL) is shown in Figure 1. It contains three components: an instance-level encoding module, a cluster selection module and a cluster-level representation learning module (CRL) based on the embedded topic modelling. The instance-level encoding module generates the latent representations of texts; the cluster selection module takes the outputs of the encoding module as the input to generate the cluster assignments; CRL attempts to optimise the overall structure of data by connecting the latent representations of cluster centres with the embedded topic model. More detailed descriptions of our method will be provided in the following.

3.1 The Instance-level Encoding Module

In the instance-level encoding module, we aim to generate optimised instance-level representation by contrastive learning. Suppose the inputs of the encoding module include: the original texts $\mathcal{B} = \{\boldsymbol{x}_i\}_{i=1}^{M}$ and its corresponding augmentation set $\mathcal{B}^a = \{(\tilde{\boldsymbol{x}}_{i_1}, \tilde{\boldsymbol{x}}_{i_2})\}_{i=1}^{M}$, where $\boldsymbol{x}_i$ is a sample of input texts, M is the number of samples, and $(\tilde{\boldsymbol{x}}_{i_1}, \tilde{\boldsymbol{x}}_{i_2})$ contains augmented versions of $\boldsymbol{x}_i$ to enable contrastive learning. We will apply the *Contextual Augmenter* (Kobayashi, 2018), which utilizes the pre-trained transformer-based models to find suitable words for synonym substitution (Kobayashi, 2018): Bertbase (Devlin et al., 2018) and Roberta (Liu et al., 2019) are used for generating $\tilde{\boldsymbol{x}}_{i_1}$ and $\tilde{\boldsymbol{x}}_{i_2}$, respectively.

As shown in the lower left part of Figure 1, a language model followed by a fully connected neural network FC_e are used to map the data from the original space $\mathcal{X}$ to latent space $\mathcal{H}$. Here, Sentence-BERT (Reimers and Gurevych, 2019) is chosen as the language model since it has fine-tuned BERT (Devlin et al., 2018) for better measuring sentence similarities which would suit short text clustering. The outputs of the encoding module can be represented as:

$$\begin{aligned} &\boldsymbol{h}_i = FC_e(SentenceBERT(\boldsymbol{x}_i)), \\ &\tilde{\boldsymbol{h}}_{i_j} = FC_e(SentenceBERT(\tilde{\boldsymbol{x}}_{i_j})), \\ &i \in \{1, ..., M\}, j \in \{1, 2\}. \end{aligned} \tag{1}$$

In order to optimise instance-level representations in latent space, we follow the work in (Zhang et al., 2021) and introduce the same contrasting module to DECCRL for leveraging the power of contrastive learning. For any $\boldsymbol{x}_i$ in $\mathcal{B}$, we refer to its augmented versions, $\tilde{\boldsymbol{x}}_{i_1}$ and $\tilde{\boldsymbol{x}}_{i_2}$ in $\mathcal{B}^a$, as the positive pair, while treating the other elements in sample pairs of $\mathcal{B}^a$ as negative instances. The contrasting module adopts fully connected neural networks to transform latent representations $\tilde{\boldsymbol{h}}_{i_j}$ into $\tilde{\boldsymbol{v}}_{i_j}$.

The contrastive loss is defined to make the positive samples closer and negative samples further apart from each other as follows:

$$l_{i_1}^{CL} = -log\frac{exp(sim(\tilde{\boldsymbol{v}}_{i_1}, \tilde{\boldsymbol{v}}_{i_2})/\tau)}{\sum_{m=1}^{M} \mathbb{1}_{m \neq i} \cdot exp(sim(\tilde{\boldsymbol{v}}_{i_1}, \tilde{\boldsymbol{v}}_{m_2})/\tau)}, \tag{2}$$

$$l_{i_2}^{CL} = -log\frac{exp(sim(\tilde{\boldsymbol{v}}_{i_2}, \tilde{\boldsymbol{v}}_{i_1})/\tau)}{\sum_{m=1}^{M} \mathbb{1}_{m \neq i} \cdot exp(sim(\tilde{\boldsymbol{v}}_{i_2}, \tilde{\boldsymbol{v}}_{m_1})/\tau)}, \tag{3}$$

where $\mathbb{1}_{m \neq i}$ is an indicator function, τ denotes the temperature parameter, and $sim(.)$ measures the cosine similarity between two vectors (Chen et al., 2020). The contrastive loss averaged across samples is:

$$\mathcal{L}_{CL} = \sum_{i=1}^{M} (l_{i_1}^{CL} + l_{i_2}^{CL})/M. \tag{4}$$

3.2 The Cluster Selection Module

The *cluster selection module* is designed to assign each data sample to a certain cluster as shown in the upper left of Figure 1. Assume there are K clusters, where each cluster can be characterized by its centroid $\boldsymbol{c}_k$ for $k \in \{1, ..., K\}$ in the latent space $\mathcal{H}$. Following the approach developed in Van der Maaten and Hinton (2008); Zhang et al. (2021), we calculate the probability of assigning the i-th input text $\boldsymbol{x}_i$ to the k-th cluster based on the Student's t-distribution as follows:

$$o_{ik} = \frac{(1 + \|\boldsymbol{h}_i - \boldsymbol{c}_k\|_2^2/\alpha)^{-\frac{\alpha+1}{2}}}{\sum_{k'=1}^{K}(1 + \|\boldsymbol{h}_i - \boldsymbol{c}_{k'}\|_2^2/\alpha)^{-\frac{\alpha+1}{2}}}, \tag{5}$$

where α is the degree of freedom of the Student's t-distribution, and $\boldsymbol{h}_i$ is the latent representation of $\boldsymbol{x}_i$ generated by the *instance-level encoding module* using Eq (1). A $softmax$ layer is then used to normalize o_{ik} as:

$$\boldsymbol{s}_i = softmax([o_{i,1}; ...; o_{i,K}]), \tag{6}$$

from which the cluster assignment of the i-th text sample can be sampled.

To optimize the estimation of the centroid $\boldsymbol{c}_k$ for each cluster, we utilize an auxiliary probability t_{ik} as discussed in Xie et al. (2016):

$$t_{ik} = \frac{o_{ik}^2/f_k}{\sum_{k'} o_{ik}^2/f_{k'}}, \tag{7}$$

where $f_k = \sum_{i=1}^{M} o_{ik}$ is the soft cluster frequency. To make the cluster assignment probability close to the auxiliary probability, we will minimize the KL divergence between them, which is defined as:

$$l_i^C = \sum_{k=1}^{K} t_{ik} log \frac{t_{ik}}{o_{ik}}, \tag{8}$$

The cluster-oriented loss averaged across M samples is:

$$\mathcal{L}_{Cluster} = \sum_{i=1}^{M} l_i^C / M. \tag{9}$$

3.3 The Cluster-level Representation Learning Module

In order to optimise the cluster-level representations by reconstructing, the cluster-level representation learning module (CRL) extend the idea of the embedded topic modelling (ETM) to reconstruct words from cluster center representations. The representations of cluster centroids $\boldsymbol{c}_k$ for $k \in \{1, ..., K\}$ obatined from the *cluster selection module* are used to generate latent representations of topics, from which the text data can be reconstructed. As shown in the right part of Figure 1, the CRL has two main parts: the generative part and the inference part, which will be explained in detail in the following paragraphs.

3.3.1 The Generative Part

Suppose L is the embedding length of vectors in the latent space $\mathcal{H}$. Let us denote the embeddings of words obtained from Word2Vec (Mikolov et al., 2013) as $\boldsymbol{\rho} = [\boldsymbol{\rho}_1, ..., \boldsymbol{\rho}_V]$, where $\boldsymbol{\rho}_v \in \mathbb{R}^{2L}$ is for the v-th word and V is the vocabulary size. The embedding of each topic t from the cluster k is represented as $\boldsymbol{\alpha}_k^t \in \mathbb{R}^{2L}$. In our model, the embeddings of topics from each cluster k are related to the cluster centroid $\boldsymbol{c}_k$ as follows:

$$\boldsymbol{\alpha}_k^t = FC_g(\boldsymbol{c}_k \oplus \boldsymbol{u}_t) \; for \; t \in \{1, .., T\}, \tag{10}$$

where T is the total number of topics, FC_g is a fully connected neural network, $\boldsymbol{u}_t \in \mathbb{R}^L$ is a trainable vector, and $\oplus$ is the concatenation operator. Using $\boldsymbol{\alpha}_k = [\boldsymbol{\alpha}_k^1, ..., \boldsymbol{\alpha}_k^T]$ and $\boldsymbol{\rho}$, the distribution over the vocabulary for topics within the k-th cluster can be obtained from:

$$\boldsymbol{\beta}_k = softmax((\boldsymbol{\rho})' \boldsymbol{\alpha}_k), \tag{11}$$

where $(.)'$ is the matrix transpose operator, and $\boldsymbol{\beta}_k \in \mathbb{R}^{V \times T}$ is a collection of simplexes achieved by computing the semantic similarity between topics and words.

For the i-th text, its topic proportions $\boldsymbol{\theta}_i$, indicating the prevalence of different topics in the text. Let $w_{i,n}$ denote the n-th word in the i-th text, whose topic assignment $z_{i,n}$ is assumed to be drawn from $z_{i,n} \sim Cat(\boldsymbol{\theta_i})$, where $Cat(.)$ denotes the categorical distribution. With $z_{i,n}$ and $\boldsymbol{\beta}_k$, the probability of observing $w_{i,n}$ is then:

$$p(w_{i,n}|\boldsymbol{\beta}_k, z_{i,n}) = Multi(\boldsymbol{\beta}_k^{z_{i,n}}), \tag{12}$$

where $Multi(.)$ is the Multinomial distribution and $\boldsymbol{\beta}_k^{z_{i,n}}$ is the $z_{i,n}$-th column of $\boldsymbol{\beta}_k$. Then, the log marginal likelihood of observing the i-th text can be represented as:

$$\begin{aligned} \log p(w_{i,n}|\hat{\boldsymbol{x}}_i) &= \sum_{z_{i,n}} \log p(w_{i,n}|\boldsymbol{\beta}_k^{z_{i,n}}) p(z_{i,n}|\hat{\boldsymbol{x}}_i) \\ &= \log \boldsymbol{\beta}_k \boldsymbol{\theta}_i, \end{aligned} \tag{13}$$

where $\boldsymbol{\theta}_i = softmax(\hat{\boldsymbol{x}}_i)$, and $\hat{\boldsymbol{x}}_i$ will be approximated from the BoW form of original text $\boldsymbol{x}_i$ as to be explained in the following paragraphs.

3.3.2 The Inference Part

Let us define the approximated distribution of $\hat{\boldsymbol{x}}_i$ as $q(\hat{\boldsymbol{x}}_i|\boldsymbol{x}_i^{bow*})$, where $\boldsymbol{x}_i^{bow*}$ is the normalized representation of BoWs data $\boldsymbol{x}_i^{bow}$. In the inference part, $\boldsymbol{x}_i^{bow*}$ is first passed through a fully connected neural network to get its latent representation, which is then fed into two parallel fully connected neural networks to get two vectors: $\boldsymbol{\mu}_i$ and $\boldsymbol{\sigma}_i$. By treating $\boldsymbol{\mu}_i$ and $\boldsymbol{\sigma}_i$ as the mean and standard deviation, $\hat{\boldsymbol{x}}_i$ can be sampled from:

$$\hat{\boldsymbol{x}}_i = \boldsymbol{\mu}_i + \boldsymbol{\epsilon} \cdot \boldsymbol{\sigma}_i, \tag{14}$$

where $\boldsymbol{\epsilon} \in \mathcal{N}(0, \mathbf{I})$. In our CRL, we choose the negative evidence lower bound as the cluster-level

representation loss:

$$\mathcal{L}_{RC} = -\mathbb{E}_{q(\hat{\boldsymbol{x}}_i|\boldsymbol{x}_i^{bow*})}[\sum_{n=1}^{N_i} \log p(w_{i,n}|\hat{\boldsymbol{x}}_i)] + D_{KL}[q(\hat{\boldsymbol{x}}_i|\boldsymbol{x}_i^{bow*})||p(\hat{\boldsymbol{x}}_i)], \quad (15)$$

where N_i is the number of words in the ith text, and $p(\hat{\boldsymbol{x}}_i)$ is the prior distribution of $\hat{\boldsymbol{x}}_i$, assumed to be normally distributed.

To train our model, the overall optimization objective is defined as:

$$\mathcal{L} = \mathcal{L}_{CL} + \lambda_c * \mathcal{L}_{Cluster} + \lambda_r * \mathcal{L}_{RC}, \quad (16)$$

where λ_c and λ_r are the weights of $\mathcal{L}_{Cluster}$ and $\mathcal{L}_{RC}$, respectively.

4 Experimental Setup

4.1 Dataset

We evaluate our model using three benchmark datasets. Table 1 briefly summarizes them with some details elaborated as follows.

- **AgNews** is a collection of news titles (Zhang and LeCun, 2015). In our experiment, we use a subset version from (Rakib et al., 2020), which contains 8,000 documents from four different categories. For performance evaluation, 6,400 documents are used for training while 1,600 documents are used for testing.
- **StackOverflow** is a subset of the challenge data released by Kaggle[1]. This dataset contains 20,000 documents, which can fall into 20 different categories (Xu et al., 2017). For model training and testing, 15,084 and 4,916 documents are used respectively.
- **Biomedical** is the challenge data published in BioASQ[2]. The version provided by Xu et al. (2017) is used in our experiment, which contains 20,000 paper titles from 20 categories.

4.2 Baseline

- **BoW & TF-IDF** (Zhang et al., 2021) together with the K-means clustering is used as a baseline method, where the length of BoW or TF-IDF vectors is set to 1,500.
- **STCC** (Xu et al., 2017) is a typical two-step deep clustering method. It first used Word2Vec (Mikolov et al., 2013) to embed words in the original text. The resulting word embeddings are fed into convolutional neural networks to get latent representations. Then, K-means is used to detect clusters using representations obtained from the previous step.
- **HAC-SD** (Rakib et al., 2020) introduces iterative classification to boost the performance of clustering. It considers outlier removal to generate outlier-free clusters for short texts. The outlier removed data is used to train a classification algorithm based on the cluster assignments.
- **SCCL** (Zhang et al., 2021) is a state-of-the-art deep embedded clustering method for short texts, which leverages the power of contrastive learning to improve clustering.

4.3 Settings

In our approach, we use the Adam optimizer (Kingma and Ba, 2014) with the batch size of 200. We choose *distilbert-base-nli-stsb-mean-tokens* for SentenceBERT and set the maximum input length to 32. Same as Zhang et al. (2021), we set α = 10.0 for the Biomedical dataset and α = 1.0 for the other datasets. The temperature parameter used in the contrasting module is 0.5. As used in the recent works for short text clustering (Xu et al., 2017; Rakib et al., 2020; Zhang et al., 2021), we adopt the clustering accuracy (ACC) (Xie et al., 2016) and the normalized mutual information (NMI) (Strehl and Ghosh, 2002) to show the performance of clustering models. For fair comparison, supervised pre-trained models are not applied. Since most existing works have pre-defined cluster numbers and reported results, we adopt this practice and follow their training/test protocols stated in their paper.

5 Experimental Result

5.1 Clustering performance compared with baselines

Table 2 shows the results of baseline methods along with our model on 3 benchmark datasets. The observations can be summarized as follows. First, the deep clustering methods, including STCC, SCCL and DECCRL, outperform conventional clustering methods which are based on BoW or TF-IDF for feature extraction and k-means for cluster detec-

[1]https://www.kaggle.com/c/predict-closed-questions-on-stackoverflow/download/train.zip

[2]http://participants-area.bioasq.org/

Table 1: Summary statistics of three benchmark datasets.

Dataset	# Docs	# Training	# Test	# Words	# Classes	# Average Length
AgNews	8,000	6,400	1,600	21,063	4	23
StackOverflow	20,000	15,084	4,916	10,941	20	8
Biomedical	20,000	15,583	4,417	18,244	20	13

Table 2: The short text clustering results for three benchmark text datasets. Our result are averaged over five random runs. The ACC and NMI values for baseline methods are directly obtained from (Zhang et al., 2021).

Models	AgNews Dataset		StackOverflow Dataset		Biomedical Dataset	
	ACC	NMI	ACC	NMI	ACC	NMI
BoW	27.6	2.6	18.5	14.0	14.3	9.2
TF-IDF	34.5	11.9	58.4	58.7	28.3	23.2
STCC	-	-	51.1	49.0	43.6	38.1
HAC-SD	81.8	54.6	64.8	59.5	40.1	33.5
SCCL(Zhang et al., 2021)	88.2	68.2	75.5	74.5	46.2	**41.5**
DECCRL	**88.9**	**69.2**	**82.3**	**76.7**	**47.0**	**41.5**

tion. HAC-SD, considering outlier removal, shows better performance than conventional k-means clustering approaches. Secondly, SCCL has shown better performance than the other baseline models, reflecting the need of introducing contrastive learning into the clustering models. More importantly, our model outperforms all other baseline models for all datasets, especially for the StackOverflow Dataset. Compared with SCCL, our model has achieved higher ACC and NMI values.

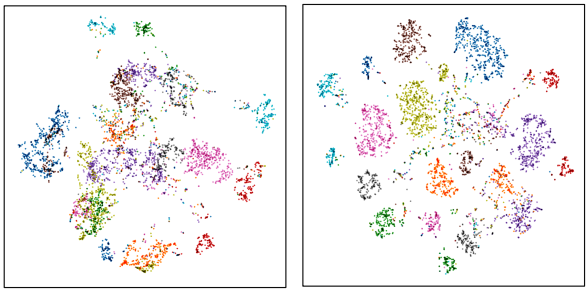

(a) (b)

Figure 2: The TSNE visualization of the latent representations for the StackOverflow dataset, where (a) is from-SCCL and (b) is from DECCRL. Each color indicates a ground-truth cluster category.

To further demonstrate the importance of introducing the CRL into the clustering model, we visualize the distribution of samples from a random subset of StackOverflow (n=4,916) using the t-SNE (Van der Maaten and Hinton, 2008) visualization algorithm. In Figure 2, samples are assigned to different colours based on their ground-truth categories, where the total number of categories is 20. By comparing the results from SCCL and DECCRL under the same settings as shown in Figure 2(a) and Figure 2(b) respectively, we can find that our model were able to learn representations that are more separable in the latent space. With this observation, it is more confident to predicate that the representations generated from our model would lead a better clustering results.

Table 3: The results of different representation guidance strategies. **AGN**, **SO** and **BIO** refer to the AgNews, StackOverflow and Biomedical datasets respectively.

Metric	Model	AGN	SO	BIO
NMI	Ours w/o CRL	59.2	74.0	28.0
	Ours w/o CRL w LSR	62.3	74.5	31.2
	Ours	**69.1**	**76.7**	**41.5**
ACC	Ours w/o CRL	81.5	78.8	29.4
	Ours w/o CRL w LSR	83.9	81.2	31.3
	Ours	**88.9**	**82.3**	**47.0**

5.2 The influence of cluster-level representation learning module

In this section, we investigate the performance of different cluster-level representation learning strategies, which are designed as follows:

- **Ours w/o CRL** – DECCRL without CRL. The model learns cluster-level representation without imposing any direct contraints.

- **Ours w/o CRL w LSR** – Replacing the CRL of DECCRL with a latent space reconstruction module (LSR). The L-dimensional latent representations are fed into a deep neural network, where an encoder generates $L/2$-dimensional vectors and a decoder returns L-dimensional vectors. The difference between the inputs and outputs of this network is considered as an extra loss for model training.

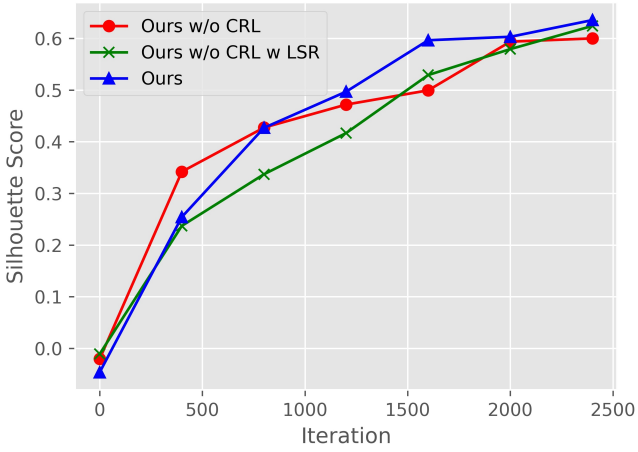

Figure 3: Silhouette scores from different cluster-level representation learning strategies during training process.

Table 3 shows the ACC and NMI values from modes with different cluster-level representation learning strategies. By comparing the performance of DECCRL with DECCRL w/o CRL, we find that the CRL has greatly improved for all three datasets. DECCRL w/o CRL w lSR, which constrains features at the latent level, has shown better performance than DECCRL w/o CRL but no better than DECCRL.

Without referring to the ground-truth labels of clusters, we use a pure clustering metric, Silhouette score (Rousseeuw, 1987), to investigate the performance of different cluster-level representation learning strategies during the training process as shown in Figure 3. The results from Figure 3 and Table 3 show that representations from all models seem to return clusters of similar characteristics (e.g., compactness and separation indicated by the Silhouette score). However, without adopting the proposed CRL, the learned representations cannot well preserve semantic information contained in the original text data such that the ACC and NMI scores generated using other methods are not as high as ours. Given the above observations, we find that using embedded topic modelling to guide the latent representation is a promising strategy.

Table 4: Selected clusters and their corresponding representative hidden topics.

Cluster Label	Representative Topics
osx	Topic1: ['terminal', 'mac', 'command', 'stdin'] Topic2: ['max', 'os', '**osx**', 'console'] Topic3: ['file', 'application', 'set', 'create']
excel	Topic1: ['data', 'xml', 'cell', 'table'] Topic2: ['**excel**', 'list', 'files', 'worksheet'] Topic3: ['file', 'create', 'application', 'xml']
oracle	Topic1: ['**oracle**', 'db', 'view', 'connection'] Topic2: ['sql', 'table', 'data', 'database'] Topic3: ['file', 'application', 'data', 'multiple']

5.3 Understanding clusters

This subsection shows that our embedded topic modelling based CRL does not only improve the clustering performance but can also be used to characterize each cluster using learned topics. Table 4 shows topics learned from three representative clusters from the StackOverflow Dataset whose cluster labels are: 'osx', 'excel' and 'oracle'. These labels are the ground-truth labels provided by the dataset, and indicate the meaning of each cluster at a coarse-grained level. To check whether the learned topics generated from our CRL are consistent with these cluster labels, Table 4 shows some selected topics (characterized by the top four key words) from each cluster. We can find that the class labels can be found as key words of topics. The learned topics can provide more detailed understanding of clusters. For example, for cluster 'oracle', its first topic indicates that 'oracle' might be a 'db' (database) having operations like 'view', 'connection' and 'access'. Apart from these observations, different clusters are also found to have topics with similar meaning. For example, the third topics from three clusters are all about 'file'.

5.4 Clustering performance with different topic numbers

We investigate the impacts of T (i.e. the total number of topics) on the clustering performance. Figure 4 (a) and (b) show the values of NMI and ACC for the three benchmark datasets with the topic number T chosen from $\{1, 5, 10, 15, 20, 25, 30\}$. For the BioMedical Dataset, we can see that the NMI and ACC values become stable when T exceeds 20. Therefore, in the experiment for this dataset, we set the topic number $T = 20$. For the AgNews

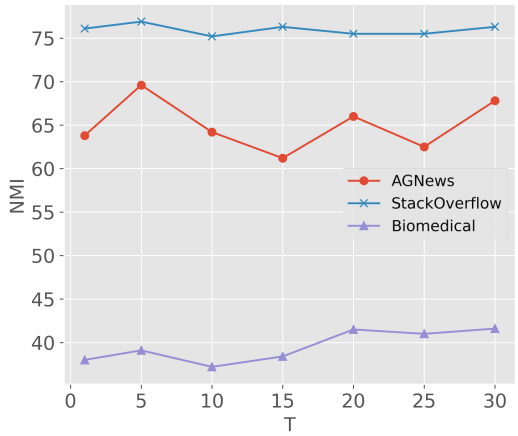

(a)

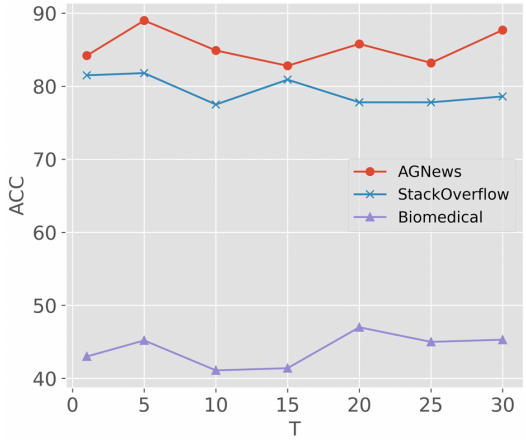

(b)

Figure 4: The clustering performance in terms of (a) NMI and (b) ACC with different topic number T.

and StackOverflow datasets, the highest NMI and ACC values are obtained at $T = 5$. Thus, we set the number of topics to 5 for these two datasets.

6 Conclusion

This paper proposes a deep embedded clustering method for short text clustering by developing a cluster-level representation learning module (CRL) to capture the overall structure of data and hence improve the clustering performance. Our model comprises three main parts: the instance-level encoding, the cluster selection, and CRL. To show the performance of our model, we utilize three benchmark datasets. The clustering performance has not only been quantitatively evaluated by ACC and NMI values but are also qualitatively assessed by case studies and visualization. The comparison of different cluster-level representation strategies shows the effectiveness of our CRL. The proposed model is expected to be generalizable to meet various text clustering challenges, not only limited to short texts. In the future, we will extend our model to capture dynamics changes of cluster centers that might evolve over time, where dynamic ETM learning smooth trajectories of topic embeddings can be considered. Another future research direction is to adopt non-parametric Bayesian approaches (e.g., Dirichlet process mixture model) to improve our clustering model so that the exact number of clusters does not need to be predefined.

7 Acknowledge

This work was supported by the National Key Research and Development Program of China (No. 2021ZD0113303).

References

Melissa Ailem, Bowen Zhang, and Fei Sha. 2019. Topic augmented generator for abstractive summarization. *arXiv preprint arXiv:1908.07026*.

Elie Aljalbout, Vladimir Golkov, Yawar Siddiqui, Maximilian Strobel, and Daniel Cremers. 2018. Clustering with deep learning: Taxonomy and new methods. *arXiv preprint arXiv:1801.07648*.

Yuki Markus Asano, Christian Rupprecht, and Andrea Vedaldi. 2019. Self-labelling via simultaneous clustering and representation learning. *arXiv preprint arXiv:1911.05371*.

Farzindar Atefeh and Wael Khreich. 2015. A survey of techniques for event detection in twitter. *Computational Intelligence*, 31(1):132–164.

Siddhartha Brahma. 2018. Unsupervised learning of sentence representations using sequence consistency. *arXiv preprint arXiv:1808.04217*.

Mathilde Caron, Piotr Bojanowski, Armand Joulin, and Matthijs Douze. 2018. Deep clustering for unsupervised learning of visual features. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 132–149.

Ting Chen, Simon Kornblith, Mohammad Norouzi, and Geoffrey Hinton. 2020. A simple framework for contrastive learning of visual representations. In *International conference on machine learning*, pages 1597–1607. PMLR.

Yulai Cong, Bo Chen, Hongwei Liu, and Mingyuan Zhou. 2017. Deep latent dirichlet allocation with topic-layer-adaptive stochastic gradient riemannian mcmc. In *International Conference on Machine Learning*, pages 864–873. PMLR.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. Bert: Pre-training of deep

bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*.

Adji B Dieng, Francisco JR Ruiz, and David M Blei. 2020. Topic modeling in embedding spaces. *Transactions of the Association for Computational Linguistics*, 8:439–453.

Adji B Dieng, Chong Wang, Jianfeng Gao, and John Paisley. 2016. Topicrnn: A recurrent neural network with long-range semantic dependency. *arXiv preprint arXiv:1611.01702*.

Alexey Dosovitskiy, Philipp Fischer, Jost Tobias Springenberg, Martin Riedmiller, and Thomas Brox. 2015. Discriminative unsupervised feature learning with exemplar convolutional neural networks. *IEEE transactions on pattern analysis and machine intelligence*, 38(9):1734–1747.

Kamran Ghasedi, Xiaoqian Wang, Cheng Deng, and Heng Huang. 2019. Balanced self-paced learning for generative adversarial clustering network. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 4391–4400.

Xifeng Guo, Long Gao, Xinwang Liu, and Jianping Yin. 2017. Improved deep embedded clustering with local structure preservation. In *Ijcai*, pages 1753–1759.

Zhuxi Jiang, Yin Zheng, Huachun Tan, Bangsheng Tang, and Hanning Zhou. 2016. Variational deep embedding: An unsupervised and generative approach to clustering. *arXiv preprint arXiv:1611.05148*.

Diederik P Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*.

Ryan Kiros, Yukun Zhu, Russ R Salakhutdinov, Richard Zemel, Raquel Urtasun, Antonio Torralba, and Sanja Fidler. 2015. Skip-thought vectors. *Advances in neural information processing systems*, 28.

Sosuke Kobayashi. 2018. Contextual augmentation: Data augmentation by words with paradigmatic relations. *arXiv preprint arXiv:1805.06201*.

Yunfan Li, Peng Hu, Zitao Liu, Dezhong Peng, Joey Tianyi Zhou, and Xi Peng. 2021. Contrastive clustering. In *2021 AAAI Conference on Artificial Intelligence (AAAI)*.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Roberta: A robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692*.

Qianli Ma, Jiawei Zheng, Sen Li, and Gary W Cottrell. 2019. Learning representations for time series clustering. *Advances in neural information processing systems*, 32:3781–3791.

Naveen Sai Madiraju. 2018. *Deep temporal clustering: Fully unsupervised learning of time-domain features*. Ph.D. thesis, Arizona State University.

Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013. Efficient estimation of word representations in vector space. *arXiv preprint arXiv:1301.3781*.

Georgios Paltoglou and Mike Thelwall. 2012. Twitter, myspace, digg: Unsupervised sentiment analysis in social media. *ACM Transactions on Intelligent Systems and Technology (TIST)*, 3(4):1–19.

Md Rashadul Hasan Rakib, Norbert Zeh, Magdalena Jankowska, and Evangelos Milios. 2020. Enhancement of short text clustering by iterative classification. In *International Conference on Applications of Natural Language to Information Systems*, pages 105–117. Springer.

Nils Reimers and Iryna Gurevych. 2019. Sentence-bert: Sentence embeddings using siamese bert-networks. *arXiv preprint arXiv:1908.10084*.

Danilo Jimenez Rezende, Shakir Mohamed, and Daan Wierstra. 2014. Stochastic backpropagation and approximate inference in deep generative models. In *International conference on machine learning*, pages 1278–1286. PMLR.

Peter J Rousseeuw. 1987. Silhouettes: a graphical aid to the interpretation and validation of cluster analysis. *Journal of computational and applied mathematics*, 20:53–65.

Abigail See, Peter J Liu, and Christopher D Manning. 2017. Get to the point: Summarization with pointer-generator networks. *arXiv preprint arXiv:1704.04368*.

Akash Srivastava and Charles Sutton. 2017. Autoencoding variational inference for topic models. *arXiv preprint arXiv:1703.01488*.

Alexander Strehl and Joydeep Ghosh. 2002. Cluster ensembles—a knowledge reuse framework for combining multiple partitions. *Journal of machine learning research*, 3(Dec):583–617.

Hongyin Tang, Miao Li, and Beihong Jin. 2019. A topic augmented text generation model: Joint learning of semantics and structural features. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 5090–5099.

Fei Tian, Bin Gao, Qing Cui, Enhong Chen, and Tie-Yan Liu. 2014. Learning deep representations for graph clustering. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 28.

Tsung Wei Tsai, Chongxuan Li, and Jun Zhu. 2020. Mice: Mixture of contrastive experts for unsupervised image clustering. In *International Conference on Learning Representations*.

Laurens Van der Maaten and Geoffrey Hinton. 2008. Visualizing data using t-sne. *Journal of machine learning research*, 9(11).

Wouter Van Gansbeke, Simon Vandenhende, Stamatios Georgoulis, Marc Proesmans, and Luc Van Gool. 2020. Scan: Learning to classify images without labels. In

European Conference on Computer Vision, pages 268–285. Springer.

Pascal Vincent, Hugo Larochelle, Isabelle Lajoie, Yoshua Bengio, Pierre-Antoine Manzagol, and Léon Bottou. 2010. Stacked denoising autoencoders: Learning useful representations in a deep network with a local denoising criterion. *Journal of machine learning research*, 11(12).

Wenlin Wang, Zhe Gan, Hongteng Xu, Ruiyi Zhang, Guoyin Wang, Dinghan Shen, Changyou Chen, and Lawrence Carin. 2019. Topic-guided variational autoencoders for text generation. *arXiv preprint arXiv:1903.07137*.

Xinyi Wang and Yi Yang. 2020. Neural topic model with attention for supervised learning. In *International Conference on Artificial Intelligence and Statistics*, pages 1147–1156. PMLR.

Zhengjue Wang, Zhibin Duan, Hao Zhang, Chaojie Wang, Long Tian, Bo Chen, and Mingyuan Zhou. 2020. Friendly topic assistant for transformer based abstractive summarization. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 485–497.

Jianlong Wu, Keyu Long, Fei Wang, Chen Qian, Cheng Li, Zhouchen Lin, and Hongbin Zha. 2019. Deep comprehensive correlation mining for image clustering. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 8150–8159.

Liang Wu and Huan Liu. 2018. Tracing fake-news footprints: Characterizing social media messages by how they propagate. In *Proceedings of the eleventh ACM international conference on Web Search and Data Mining*, pages 637–645.

Junyuan Xie, Ross Girshick, and Ali Farhadi. 2016. Unsupervised deep embedding for clustering analysis. In *International conference on machine learning*, pages 478–487. PMLR.

Jiaming Xu, Bo Xu, Peng Wang, Suncong Zheng, Guanhua Tian, and Jun Zhao. 2017. Self-taught convolutional neural networks for short text clustering. *Neural Networks*, 88:22–31.

Linxiao Yang, Ngai-Man Cheung, Jiaying Li, and Jun Fang. 2019. Deep clustering by gaussian mixture variational autoencoders with graph embedding. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 6440–6449.

Xu Yang, Cheng Deng, Kun Wei, Junchi Yan, and Wei Liu. 2020. Adversarial learning for robust deep clustering. *Advances in Neural Information Processing Systems*, 33.

Jesin Zakaria, Abdullah Mueen, and Eamonn Keogh. 2012. Clustering time series using unsupervised-shapelets. In *2012 IEEE 12th International Conference on Data Mining*, pages 785–794. IEEE.

Jichuan Zeng, Jing Li, Yan Song, Cuiyun Gao, Michael R Lyu, and Irwin King. 2018. Topic memory networks for short text classification. *arXiv preprint arXiv:1809.03664*.

Dejiao Zhang, Feng Nan, Xiaokai Wei, Shangwen Li, Henghui Zhu, Kathleen McKeown, Ramesh Nallapati, Andrew Arnold, and Bing Xiang. 2021. Supporting clustering with contrastive learning. *arXiv preprint arXiv:2103.12953*.

Hao Zhang, Bo Chen, Dandan Guo, and Mingyuan Zhou. 2018. Whai: Weibull hybrid autoencoding inference for deep topic modeling. *arXiv preprint arXiv:1803.01328*.

Xiang Zhang and Yann LeCun. 2015. Text understanding from scratch. *arXiv preprint arXiv:1502.01710*.

Jun Zhao, Tao Gui, Qi Zhang, and Yaqian Zhou. 2021. A relation-oriented clustering method for open relation extraction. *arXiv preprint arXiv:2109.07205*.

Decoupling Mixture-of-Graphs: Unseen Relational Learning for Knowledge Graph Completion by Fusing Ontology and Textual Experts

Ran Song[1,2], **Shizhu He**[3,4], **Suncong Zheng**[6], **Shengxiang Gao**[1,2] *, **Kang Liu**[3,4,5], **Zhengtao Yu**[1,2], and **Jun Zhao**[3,4]

[1] Faculty of Information Engineering and Automation, Kunming University of Science and Technology
[2] Yunnan Key Laboratory of Artificial Intelligence, Kunming, China
[3] Institute of Automation, Chinese Academy of Sciences, Beijing, China
[4] School of Artificial Intelligence, University of Chinese Academy of Science, Beijing, China
[5]Beijing Academy of Artificial Intelligence, Beijing, China [6] Tencent AI Lab
{song_ransr,junzhao123}@163.com, {shizhu.he,kliu}@nlpr.ia.ac.cn,
congzheng@tencent.com, {gaoshengxiang.yn,ztyu}@hotmail.com

Abstract

Knowledge Graph Embedding (KGE) has been proposed and successfully utilized for knowledge Graph Completion (KGC). But classic KGE paradigm often fail in unseen relation representations. Previous studies mainly utilize the textual descriptions of relations and its neighbor relations to represent unseen relations. In fact, the semantics of a relation can be expressed by three kinds of graphs: factual graph, ontology graph, textual description graph, and they can complement each other. A more common scenario in the real world is that seen and unseen relations appear at the same time. In this setting, the training set (only seen relations) and testing set (both seen and unseen relations) own different distributions. And the train-test inconsistency problem will make KGE methods easily overfit on seen relations and underperformance on unseen relations. In this paper, we propose decoupling mixture-of-graph experts (DMoG) for unseen relations learning, which could represent the unseen relations in the factual graph by fusing ontology and textual graphs, and decouple fusing space and reasoning space to alleviate overfitting for seen relations. The experiments on two unseen-only public datasets and a mixture dataset verify the effectiveness of the proposed method, which improves the state-of-the-art methods by 6.84% in Hits@10 on average.

1 Introduction

Knowledge Graphs (KGs) such as Freebase (Bollacker et al., 2008), DBpedia (Lehmann et al., 2015) and YAGO (Mahdisoltani et al., 2014) contain large amounts of entities, relations and facts, which can be used to support many NLP tasks. Knowledge-dependent tasks rely heavily on the coverage of KGs. And the incompleteness of those KGs is an urgent issue for its widespread utilization (Hogan et al., 2021). Therefore, knowledge graph embedding (KGE) (Arora, 2020; Ji et al., 2021) methods have been proposed and successfully applied to knowledge graph completion (KGC), which seek out potential facts inside KGs.

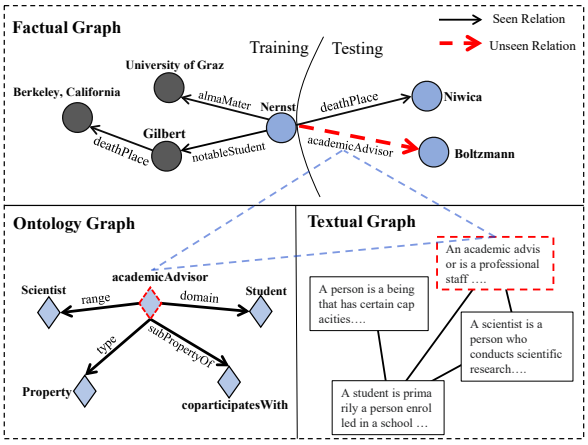

Figure 1: The semantics of a relation in a KG is expressed by three kinds of graphs: factual graph, ontology graph and textual graph. And the knowledge graph completion involving unseen relations in factual graph, which struggle in previous KGE methods, could be alleviated by utilizing their ontology and textual graphs.

In fact, most knowledge involves constantly emerging new entities and relations. And traditional KGE paradigm makes hard to deal with unseen entities and relations. For example, as illustrated in Figure 1 top, the model was trained on seen triples dataset, but required to answer the open query "*(Nernst, academicAdvisor, ?)*". Little research try to learning unseen relations representation mainly by utilizing textual description and neighboring seen relations (Qin et al., 2020; Geng et al., 2021; Zhang et al., 2020). And some datasets have been proposed to evaluate models generalizability for unseen relations. (Qin et al., 2020).

* Corresponding author

Proceedings of the 29th International Conference on Computational Linguistics, pages 2237–2246
October 12–17, 2022.

ZSGAN-KG (Qin et al., 2020) leverages a generative adversarial network to generate representations of unseen relations based on their textual descriptions. And OntoZSL (Geng et al., 2021) designs several functions to learn and fuse textual features, and then adapt a text-aware encoder to represent zero-shot entities and relations. GRL (Zhang et al., 2020) designs a classifier to select a neighboring seen relation to replace the unseen relation.

Although the above-mentioned methods can deal with unseen relations to some extent, they still have the following weaknesses: 1) Unstructured textual descriptions are incomplete and can only cover part of the semantics of relations. 2) It is often inaccurate and even noisy to use neighbor relations to represent an unseen relations. 3) Previous models are evaluated for unseen data only, but do not consider mixture dataset of seen and unseen relations. Therefore in this paper, we propose three questions for currently unseen relation methods: 1) What resources are accurate and abundant for unseen relations? 2) How to efficiently use resources to improve representations of unseen relations? 3) What challenges arise with models evaluated in mixture dataset of seen and unseen relations?

To answer above questions 1) and 2), we found that ontology, as an accurate resource, is worth considering. In fact, as shown in Figure 1, the semantics of a relation can be expressed by three different forms: 1 **factual graph** includes concrete relations between entities, 2 **ontology graph** describes high-level definitions for relations, and 3 **textual graph** contains textual descriptions of different relations. The factual graph is wide and links large amounts of entities by relations. Most KGE methods can represent KG effectively, but can not represent unseen relations. The ontology contains a high-level definition of entities and relations. In Figure 1, for relation *academicAdvisor*, the ontology graph means that *scientist have academic advisors who are students*. The textual descriptions are rich and contain different semantic information in natural language. In Figure 1, they can be assembled into a textual graph through words and sentences association. The above three graphs complete each other. We leverage ontology graph and textual graph to support factual graph to find unseen relations information.

To answer above the question 3), we found that seen and unseen relations appear at the same time in the real world. In this setting, the training set (only seen relations) and testing set (both seen and unseen relations) own different distributions. And the train-test inconsistency problem will make KGE methods easily overfit on seen relations and underperforman on unseen relations. Empirically, as the model converges in the training process the performance of seen relations gets higher, but the performance of unseen relations decreases. To overcome this issue, we are committed to making the learning of relation representations and factual reasoning in different spaces. That is, the relation representations by mixture-of-graphs is implemented on the fusion space, and fact prediction with the learned entities and relations is conducted on the reasoning space.

In this paper, we propose decoupling mixture-of-graph experts (DMoG) for unseen relations learning, which could represent the unseen relations of the factual graph by fusing ontology and textual graphs. And our method decouples fusion space and reasoning space to alleviate the overfitting on seen relations. Specifically, we collect different ontology graphs from official graph-based data, or we derive them from official dump data in other formats. To achieve the interactive information between seen and unseen relations, we leverage different GNNs to encode ontology and textual graphs. And we design different expert modules and mixture mechanism to fuse different graph information. Morevoer, we propose a transpose linear mapping to separate fusion space and reasoning space and alleviate overfitting.

We conducted extensive experiments on multiple benchmarks from public KGs such DBpedia and Wikidata. The proposed unseen relations learning method improves the state-of-the-art method by 3.68% in MRR and 6.15% in Hits@10 on average.

In short, our main contributions are as follows:

- We found that relations are expressed by factual graph, ontology graph and textual graph. Based on these observations, we propose mixture-of-graph (MoG) experts for unseen relations learning, which can represent unseen relations accurately and richly.
- We propose a decoupling strategy that alleviates the overfitting on seen relations during training. Trained KGE models effectively represent seen relations and maintain unseen relations performance.
- We implement our method with some main-

stream KGE methods. And the experimental results show that our method significantly improves the performance on the seen and unseen relations.

2 Related Work

2.1 Knowledge Graph Embedding

Recently, massive work focused on translation-based methods for knowledge graph completion (Zhang et al., 2021). The key issue of knowledge graph embedding is to learn low dimensional distributed embedding of entities and relations (Ji et al., 2021). The current KGE models can generally be categorized into translation-based models and similarity-based models. For KGE models: the pioneering model TransE (Bordes et al., 2013) embeds entities and relations as d-dimension vectors in same space, and makes vectors follow the translational principle $\mathbf{h} + \mathbf{r} = \mathbf{t}$. The subsequent work of TransE usually modifies the translational principle in different forms of relationship-specific spaces. And others translation-based models including TransR (Lin et al., 2015), TransD (Ji et al., 2015), TransAt (Qian et al., 2018) and RotatE (Sun et al., 2019) have been improved from the perspective of how entities can be better represented and translated. As for the similarity-based models, ComplEx (Trouillon et al., 2016) migrates DistMult in a complex space and offers comparable performance. However, previous embedding methods struggle in knowledge completion involving unseen relations.

2.2 Zero-shot Learning for KGC

Zero-shot learning describes tasks that given the prior knowledge (seen classes) and then transfer features from seen classes to unseen classes. Most works focus on computer vision such as image classification. In the area of knowledge graph completion, more studies focus on zero-shot entity learning which is devoted to deal with unseen entities. Some works leverage text and other auxiliary features to learn the entity representation(Xie et al., 2016; Shah et al., 2019). Some works design different models or strategies to aggregate neighbor seen entities for unseen entities (Wang et al., 2019; Albooyeh et al., 2020). Currently, inductive reasoning(Teru et al., 2020) completely disregards the symbol of entities and it means that all entities can be unseen entities. While few works consider zero-shot relation learning and model unseen relations. Few works take text-embedding spaces as semantic spaces of relation to represent unseen relations (Qin et al., 2020; Geng et al., 2021). And (Zhang et al., 2020) design a classifier-based method, which select an appropriate seen relation to replace the unseen relation. Our work focuses on unseen relations in knowledge graph completion and proposes a method that incorporates ontology graph and textual description to leaning the representations of unseen relations.

2.3 Ontology and Textual Information for KGE

The ontology is the definition and meta-information of KG, it is a core part of KG construction (Stevens et al., 2000). The massive KG relation facts are subject to frequent conflicts in the absence of ontological boundaries (Pasternack and Roth, 2013). A few studies focus on embedding techniques of cross-domain ontology and encode ontology from different perspectives (Chen et al., 2018; Gutiérrez-Basulto and Schockaert, 2018). Currently, some studies try to adapt ontology to enhance the representation of knowledge base. JOIE (Hao et al., 2019) employs both cross-view and intra-view modeling that learn on multiple facets of the knowledge base. For textual information, (Yao et al., 2019) propose to use pre-trained language models for knowledge graph completion. However, there are significant differences in the ontology of the knowledge base and knowledge graph. And some popular knowledge graphs do not distinguish between KB and KG (Ehrlinger and Wöß, 2016). Our work focuses on learning ontology representation for KGE involving unseen relations.

3 Knowledge Graph Embedding Models for KGC

KGC aims at scoring a triple (h, r, t) from KG $\mathcal{G} = (\mathcal{R}, \mathcal{E})$, where $r \in \mathcal{R}$ is relation and $h, t \in \mathcal{E}$ are entities. Traditional KGE models learn embedding matrix to translate head entity h to tail entity t through relation r. And different models have been proposed by mainly changing translating strategies. For example, TransE focuses on adding head entity and relation, which should be close to the corresponding tail entity with the scoring function by minimizes the score of a triple as follows:

$$s(h, r, t) = \| \mathbf{h} + \mathbf{r} - \mathbf{t} \|_2^2 \quad (1)$$

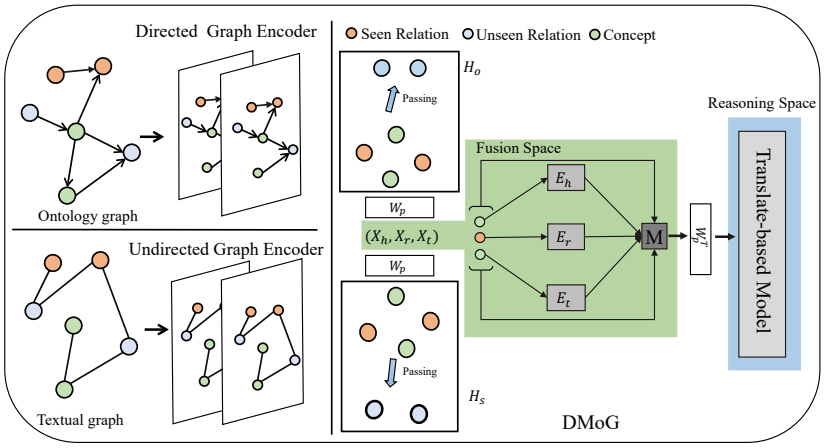

Figure 2: Our method leverages different GNN to capture ontology graph and textual graph nodes information and aggregate them by knowledge mixture of experts. By fusion ontology and textual features in fusion space (green), DMoG pools the representation of the relation in predicting a triplet fact in reasoning space (blue).

where $\mathbf{h}, \mathbf{r}, \mathbf{t} \in \mathbb{R}^d$, and d is dimension of embedding.

KGE models use the hinge loss function to effectively minimize the score. The loss function for a minibatch of labeled triples is defined as follows:

$$\mathcal{L}(\theta) = \sum_{(h,r,t)\in\mathcal{G}_b} [\gamma + f(h', r, t') - f(h, r, t)]_+ \tag{2}$$

where γ is a fixed margin, (h', r, t') is the negative fact that is commonly constructed by randomly replacing the head or tail entities from the true fact (h, r, t).

For evaluation, KGC is a link prediction task that aims to predict the missing h or t for a triple (h, r, t). Given the query $(h, r, ?)$, search the entity t that gets the minimum score with scoring function.

However, the embeddings (e.g., vectors) of all entities and relations must be initialized at the beginning for previous KGE models. If some relations r miss in training but appear in testing, they cannot be learned at all by the model. Therefore, in order to represent the unseen relations and conduct zero-shot relational learning, we consider to leverage multi-aspects information.

4 Decoupling Mixture-of-Graphs Experts

This section describes in detail our proposed approach. The framework is shown in Figure 2. Our method directly improves the effectiveness of previous KGE models for unseen relations by making rich and accurate their representations.

4.1 Framework

Our method mainly deals with three types of graphs: factual graph, ontology graph and textual graph. Factual graph is knowledge graph, following as the previous definition. Ontology is the backbone of KGs, which provide meta-descriptions to guide the knowledge graph construction and completion. Ontology is describe as directed graph $\mathcal{G}_o = (\mathcal{R}_o, \mathcal{E}_o)$, which uses meta-relations to associate ontology nodes (concepts and properties) (h_o, r_o, t_o). And the relations $\mathcal{R}$ and entities $\mathcal{E}$ of factual graph all find their own type in ontology. And relations have a unique mapping between the edges of factual graph and the nodes of ontology graph. Textual graph is undirected graph $\mathcal{G}_t = (\mathcal{R}_t, \mathcal{E}_t)$, the nodes are textual descriptions of concept and property and the edge is word embedding similarity between two nodes (h_t, r_t, t_t), and $0 < r_t < 1$.

4.2 Graph Construction

Ontology is stored in triples *(head, relation, tail)*, we take head and tail as node (indicates concept (type of entity) and property (type of relation) of factual graph) and relation (indicate meta-relations among concepts and properties) as edge. Based on the official released ontology file or the dump data, we can directly construct or build ontology graph by simple data filtering and format conversion.

For textual descriptions, we generate textual graph from textual descriptions or full names of concepts and properties. We want to find them associated as below:

$$A_t = \begin{cases} d(\mathbf{x}_i, \mathbf{x}_j), & if \quad d(\mathbf{x}_i, \mathbf{x}_j) > \varepsilon \\ 0, & otherwise \end{cases}$$

A_t is adjacency matrix of textual graph $\mathcal{G}_t$, $d(\cdot, \cdot)$ describes the cosine similarity function, ε is a threshold for connection between nodes. x_i is the word embedding of each node. Following previous work (Qin et al., 2020), Glove (Pennington et al., 2014) has higher performance than the pre-trained language model, and we use Glove to initialize word embeddings. The representation of a sentence is obtained by averaging its word embeddings.

4.3 Graph Encoder

The ontology can be represented as a directed attribute graph. Identically, the text descriptions of relations can be represented as an undirected graph. Our goal is to obtain the representation of unseen relations based on other seen nodes (concepts, properties and textual descriptions) in different graphs. Therefore, we encode ontology and textual graphs by graph neural network (GNN).

In the textual graph $\mathcal{G}_t$, the weight w_{ij} of each edge is the similarity between nodes. We consider the commonly used graph attention network (Veličković et al., 2017), but the attention value is replaced by edge weight w. The process as following:

$$\mathbf{h}_{s,i}^{(l+1)} = \sigma(\sum_{j \in \mathcal{N}_i} w_{ij} \mathbf{W}_s^l \mathbf{h}_{s,j}^l + \mathbf{W}_0^l \mathbf{h}_{s,i}) \quad (3)$$

where $\mathbf{h}_{s,i}^{(l+1)} \in \mathbb{R}^d$. The σ is sigmoid activation function. $\mathbf{W}_s^l$ is GAT weight. $\mathcal{N}_i$ denotes neighbor nodes of i. And the $\mathbf{h}_s^0$ for each node come from pretrained word embedding. To overcome the over-smooth problem of node representations, we add self-loop encoding for nodes. $\mathbf{W}_0^l$ is self-loop weight.

Similarly, ontology graph is directed graph, and each edge has its own type. We are inspired by RGCN (Schlichtkrull et al., 2018), a GNN model for relational (directed and labeled) multi-graph. To obtain the representations of concepts and properties, we use RGCN to get the representation of ontology nodes by aggregating neighborhoods nodes through different meta-relations, as follow:

$$\mathbf{h}_{o,i}^0 = e_o \mathbf{E}_o \quad (4)$$

$$\mathbf{h}_{o,i}^{(l+1)} = ReLU(\sum_{r \in \mathcal{R}_i} \sum_{j \in \mathcal{N}_i^r} \frac{1}{c_{i,r}} \mathbf{W}_r^{(l)} \mathbf{h}_{o,j}^{(l)} + \mathbf{W}_0^{(l)} \mathbf{h}_{o,i}^{(l)}) \quad (5)$$

$$\mathbf{h}_{o,i}^{(l+1)} = Norm\_Layer(\mathbf{h}_{o,i}^{(l+1)}) \quad (6)$$

where $e_o \in \mathcal{E}_o$ is node in ontology graph. $\mathbf{h}_{o,i}^{(l+1)} \in \mathbb{R}^d$ is hidden state of ontology node $h_{o,i}$ in the l-th layer, and d is dimension of layer's representation. $\mathcal{N}_i^r$ denotes the set of neighbor indices of node i under meta-relation $r_o \in \mathcal{R}_o$. $\mathbf{W}_r^{(l)}$ is relation parameters of meta-relation r which weight for node i neighboring node in l-th layer. $\mathbf{W}_0^{(l)}$ is self-loop weight for encoding self-node features. $c_{i,r}$ is a normalization constant that can either be learned or chosen in advance. ReLU is the activation function. We also use layer normalization to speed the training.

4.4 Decoupling Mixture of Graph Experts

We obtain the effective representation of the relation r from ontology space and textual space for the triple involving unseen relations and alleviate the overfitting on seen relations. For each factual triple (h, r, t), we can find ontology representation (h_o, r_o, t_o) and textual representation (h_t, r_t, t_t) from their space. We leverage adding operation to fuse two graphs information in another space, as show in:

$$\mathbf{x}_h = \mathbf{h}_o^L \mathbf{W}_p + \mathbf{h}_s^L \mathbf{W}_p \quad (7)$$

$$\mathbf{x}_r = \mathbf{r}_o^L \mathbf{W}_p + \mathbf{r}_s^L \mathbf{W}_p \quad (8)$$

$$\mathbf{x}_t = \mathbf{t}_o^L \mathbf{W}_p + \mathbf{t}_s^L \mathbf{W}_p \quad (9)$$

where $\mathbf{W}_p$ is transformation matrix, it transforms the multi representations into the fusion-aware space.

Based on the previous representations, we design aggregating strategies with mixture-of-graph experts to represent relations. Recently, the mixture of experts (Jordan and Jacobs, 1994; Shazeer et al., 2017; Fedus et al., 2021) has been widely used to capture features by different experts' views, and it can efficiently merge different features. For different knowledge roles (head, relation, tail), MoG can capture each role representation through ontology

and textual space. We define different expert networks E_h, E_r, E_t for the head, relation, tail, and a gating network M, proceeding as follows:

$$p_i = M(\mathbf{x_i}) \tag{10}$$

$$\mathbf{r} = \sum_{i \in (h,r,t)} p_i E_i(\mathbf{x}_i) \tag{11}$$

The expert networks and gating network are single-layer MLPs, and same dimension between input and output for expert networks. analyze the three roles individually and then vote to obtain the overall result.

After those processes, the representation r contain multiple information, but it still need to be put in fusion space V_f. We operate a inverse transformation to pull r back to reasoning space V_r from fusion space V_f, as follows:

$$\mathbf{r} = \mathbf{r}\mathbf{W}_p^T \tag{12}$$

where $\mathbf{W}_p^T$ transpose of a linear map, and we define $\mathbf{W}_p^T$ as square matrix to simplify calculations. It can be learned to satisfy two space bilinear forms, as follows:

$$\mathbf{W}_p = \min_{\substack{\mathbf{w}_p : V_r \to V_f \\ \mathbf{w}_p^T : V_f \to V_r}} \sum_i^N \mathcal{L}(h_i, r_i, t_i) \tag{13}$$

where N is seen dataset.

Following previous KGE models, we train our model with the margin-based ranking loss, and use a negative sampling loss function for effectively optimizing ranking loss :

$$\begin{aligned} \mathcal{L} = & - log\sigma(\gamma - f(h\mathbf{W}^E, \mathbf{r}, t\mathbf{W}^E)) \\ & - \sum_{i=1}^{n} \frac{1}{k} log\sigma(f(h_i'\mathbf{W}^E, \mathbf{r}, t_i'\mathbf{W}^E) - \gamma) \end{aligned} \tag{14}$$

where $\mathbf{W}^E$ indicates entity embedding, γ is a fixed margin value, σ is the sigmoid function, and (h_i', r, t_i') is the corresponding negative triple. The loss function can sample multiple negative triples for each positive triple at one minibatch.

5 Experiments

We conduct extensive experiments with KGC task on several public datasets, and mainly evaluate the performance of the proposed framework on zero-shot relational learning. We also verify the proposed decoupling strategy to prevent overfitting on seen relations. To directly demonstrate the effectiveness of our method, we show a visualization of seen and unseen relations.

5.1 Dataset

We select datasets from four public knowledge graphs, DBpedia, NELL, and Wikidata, to evaluate models on unseen relation learning. The current benchmark datasets contain only factual graph and not ontology graph. Therefore, we extract ontology from their origin websites[1][2]. Generally, we collect series ontology: **DBpedia** have human-created high-quality ontology, their have 17,663 triples, 7,966 nodes and 8 meta-relations. The ontology of **NELL** has 1,494 nodes, 6,907 triples and 14 meta-relations (e.g. *antisymmetric*, *mutexpredicates*). It should be noted that **Wikidata** has no official ontology, we collect 20,899 triples including 8,907 nodes and 604 meta-relations (e.g. *instance of* (P31), *see also* (P1659)) as their ontology from the released dump data [3].

Current zero-shot relational benchmarks focus entirely on inference on unseen relations. However, the seen and unseen relations should be be considered together. It requires that the model must be effective for seen relations and maintain unseen relation performance. Therefore, we propose DB100K-ZS from DB100K, which contains 383 seen relations and 77 unseen relations. We move 77 relations from training set to testing set based on DB100K. We select relations by frequency of appearing k, $k > 60$ and $k < 300$. Finally, we get training triples 540,570, seen validation triples 45,357, and seen testing triples 45,282 and unseen testing triples 13,420.

5.2 Evaluation Metrics

Triples in training data are utilized to learn KGE model, while those of validation and test dataset are respectively used to tune (hyper-parameters selection) and evaluate the model. The most typical KGC task is link prediction which aims to predict the missing h or t for a triple (h, r, t). We follow

[1] https://www.dbpedia.org/resources/ontology/

[2] http://rtw.ml.cmu.edu/resources/results/08m/NELL.08m.1115.ontology.csv.gz

[3] https://www.wikidata.org/wiki/Wikidata:Database_download

Model	NELL-ZS		Wiki-ZS		DB100K-ZS			
	UNSEEN		UNSEEN		UNSEEN		SEEN	
	MRR	H@10	MRR	H@10	MRR	H@10	MRR	H@10
DistMult	23.50	32.60	18.90	23.60	4.61	9.12	9.23	20.17
TransE	9.70	20.30	5.30	11.90	2.24	7.41	14.87	40.14
GRL(TransE) (Zhang et al., 2020)	-	-	-	-	5.15	13.12	15.12	41.51
$ZSGAN_{KG}$(DistMult) (Qin et al., 2020)	25.30	37.10	20.80	29.40	-	-	-	-
$ZSGAN_{KG}$(TransE) (Qin et al., 2020)	24.00	37.60	18.50	26.10	-	-	-	-
OntoZSL(DistMult) (Geng et al., 2021)	25.60	38.50	21.10	28.90	-	-	-	-
OntoZSL(TransE) (Geng et al., 2021)	25.00	39.90	18.40	26.50	-	-	-	-
DMoG(DistMult)	25.81	38.41	19.12	28.86	10.33	23.91	14.51	35.18
DMoG(TransE)	**30.49**	**49.11**	**23.18**	**31.13**	**23.31**	**40.79**	**27.37**	**52.07**

Table 1: Zero-shot relational learning results on NELL-ZS, Wiki-ZS and DB100K-ZS. SEEN is that relation of triples exist in training. UNSEEN is that relation of triples only in testing. **Bold** numbers denote the best results.

the setting (Sun et al., 2019) and create the query $(h, r, ?)$, and then find the ranking entities assigned by our proposed method and other KGE methods. We also apply bi-direction prediction that evaluate query $(h, r, ?)$ and $(?, r, t)$ for a test triple. The mean reciprocal rank (MRR) is computed as:

$$\frac{1}{2N_{Test}} \sum_{(h,r,t)\in Test} \left(\frac{1}{MR_{(h,r,?)}} + \frac{1}{MR_{(?,r,t)}}\right) \tag{15}$$

5.3 Implementation Details

In our experiments, we adopt the following KGE methods because of their effectiveness on link predictions. Our codes are based on (Sun et al., 2019) and adopt the PyTorch (Paszke et al., 2017) framework. For graph encoder, we used the implementation in the deep graph library (DGL). The initial word embedding is from GloVe (Pennington et al., 2014) and we set a similar threshold ε to 0.85. The entity embedding size is set to 100 for all KGE models. The GNN hidden size is set to 100, the number of layers is set to 2, and use self-loop for each node. We selected the hyperparameters corresponding to learning rate and batch size from {0.0001, 0.0005, 0.001} and {128, 256, 512, 1024}. And we use Adam to optimize all the parameters.

5.4 Results

The unseen relations denote that relation of the triples are in the test set, but they do not appear in the training set. Previous KGE models are transductive inference methods, and cannot deal with those relations. Table 1 shows the experimental results on NELL-ZS, WiKi-ZS and DB100K-ZS. The testing set of NELL-ZS and WiKi-ZS are all unseen relations (Qin et al., 2020), DB100K-ZS mix seen and unseen relations. Apparently, the newly constructed DB100K-ZS is more suitable for real-world applications.

To verify our method for unseen relation learning, we chose the latest proposed models for comparison. The GRL (Zhang et al., 2020) is the classifier-based method and hard to solve massive unseen relation. ZSGAN (Qin et al., 2020) and OntoZSL (Geng et al., 2021) always generate a representation for relation, therefore it cannot to keep traditional KGE methods performance in the seen dataset, and they do not work in DB100K-ZS.

From Table 1, our method performs better than other comparative methods in all evaluation metrics and on all three datasets. Our method increases MMR and Hits@10 by 3.86% and 6.15% for the previous state-of-the-art zero-shot method on NELL-ZS and Wiki-ZS. And our method can deal with seen and unseen relations at same time. For DB100K-ZS, DMoG not only improves the performance of unseen relations but goes beyond the base model on seen relations. We believe that the proposed model is more suitable to real world applications. In fact, Graph encoder effectively represents nodes from ontology graph and textual graph. And DMoG fully mixes different roles to extract the representation of unseen relations. The decoupling alleviates overfitting in training. The above three reasons are the key factors for our approach to achieve better results. In addition, our method could apply to any conteined in the ontology.

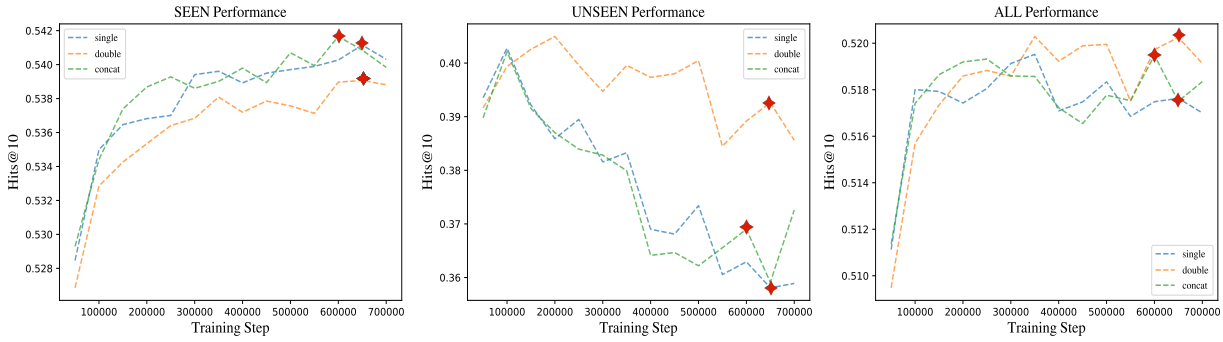

(a) (b) (c)

Figure 3: Hist@10 for seen (a), unseen (b) and all (c) relations performance in different training step. The red start denote the best performance of model in seen relations, and it still marks the same step in seen and all relations performance. In each figure, the single denote directly adding two representations. The double denote proposed decoupling strategy methods to separate the representations in different spaces. The concat denote concatenated two representations.

5.5 Alleviate Overfitting Experiment

In our setting, model trained by seen relations triples and can not get any information of unseen relations during training. And, we take early stopping through seen relations triples performance. The red star marks best checkpoint in seen performance but not all performance. As seen relations performance increases the unseen relations performance become lower, as show in Figure 3 (a, b). The reason is that model fit seen relations data and far away unseen relations latent representation. Therefore, we propose methods, which decouple fuse space and reasoning space, to alleviate the overfitting on seen relations. As show in Figure 3 (b), our method could make unseen performance decline more slowly compared to single space methods. While, our method harms seen performance little, due to its excellent unseen performance, it still has the best performance on whole seen+unseen performance, as show in Figure 3 (c).

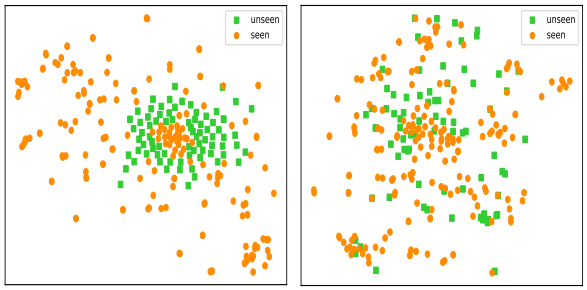

(a) TransE (b) DMoG(TransE)

Figure 4: Visualization for relation representations of DB100K-ZS testing set via t-SNE.

Model	DB100K-ZS			
	UNSEEN		SEEN	
	MRR	H@10	MRR	H@10
TransE	2.24	7.41	14.87	40.14
DMoG-T(TransE)	15.92	30.10	20.17	44.16
DMoG-O(TransE)	21.12	37.13	25.96	51.10
DMoG(TransE)	**23.31**	**40.79**	**27.37**	**52.07**

Table 2: The table shows the ablation experiment for using different information. "-T" denote only textual graph. "-O" denote only ontology graph.

5.6 Visualization of Relation Representations

In Figure 4, we show the visualization of relation representations via t-SNE. As show in Figure 4 (a), TransE can not represent unseen relations effectively, the unseen relation embeddings crowded in a cluster, which separate away seen relations space. However, our method can fully represent seen and unseen relations, the relation representations uniformly distributed in the same space, as show in Figure 4 (b).

5.7 Ablation Experiment

In order to further evaluate the effect of each module of the model, we design an ablation experiment for different graphs. As shown in Table 2, we can see that both ontology and textual graphs are helpful to KGC. DMoG enhances relations representation quality by fusing ontology and textual graph compared to single information. Further analysis showed that ontology graph is better than textual graph, formal language describe knowledge more accurately than natural language.

6 Conclusion

Our paper focuses on unseen relation representations of knowledge graph. We propose to utilize three different kinds of graphs to obtain representations of relation. And decoupling strategy alleviates the overfitting in training process. Experimental results demonstrate that our method significantly outperforms the existing state-of-the-art method on unseen relation learning.

7 Acknowledgements

The work is supported by the National Natural Science Foundation of China (U21B2027, 61972186, 61732005), the independent research project of National Laboratory of Pattern Recognition, the Youth Innovation Promotion Association CAS, the Program For Applied Basic Research of Yunnan Province in Key Areas (202002AD080001), Yunnan Provincial Major Science and Technology Special Plan Projects (No.202103AA080015) and CCF-DiDi GAIA Collaborative Research Funds for Young Scholars.

References

Marjan Albooyeh, Rishab Goel, and Seyed Mehran Kazemi. 2020. Out-of-sample representation learning for knowledge graphs. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: Findings*, pages 2657–2666.

Siddhant Arora. 2020. A survey on graph neural networks for knowledge graph completion. *arXiv preprint arXiv:2007.12374*.

Kurt Bollacker, Colin Evans, Praveen Paritosh, Tim Sturge, and Jamie Taylor. 2008. Freebase: a collaboratively created graph database for structuring human knowledge. In *Proceedings of SIGMOD*, pages 1247–1250. ACM.

Antoine Bordes, Nicolas Usunier, Alberto Garcia-Duran, Jason Weston, and Oksana Yakhnenko. 2013. Translating embeddings for modeling multi-relational data. In *Neural Information Processing Systems (NIPS)*, pages 1–9.

Muhao Chen, Yingtao Tian, Xuelu Chen, Zijun Xue, and Carlo Zaniolo. 2018. On2vec: Embedding-based relation prediction for ontology population. In *Proceedings of the 2018 SIAM International Conference on Data Mining*, pages 315–323. SIAM.

Lisa Ehrlinger and Wolfram Wöß. 2016. Towards a definition of knowledge graphs. *SEMANTiCS (Posters, Demos, SuCCESS)*, 48:1–4.

William Fedus, Barret Zoph, and Noam Shazeer. 2021. Switch transformers: Scaling to trillion parameter models with simple and efficient sparsity. *arXiv preprint arXiv:2101.03961*.

Yuxia Geng, Jiaoyan Chen, Zhuo Chen, Jeff Z Pan, Zhiquan Ye, Zonggang Yuan, Yantao Jia, and Huajun Chen. 2021. Ontozsl: Ontology-enhanced zero-shot learning. In *Proceedings of the Web Conference 2021*, pages 3325–3336.

Víctor Gutiérrez-Basulto and Steven Schockaert. 2018. From knowledge graph embedding to ontology embedding? an analysis of the compatibility between vector space representations and rules. *arXiv preprint arXiv:1805.10461*.

Junheng Hao, Muhao Chen, Wenchao Yu, Yizhou Sun, and Wei Wang. 2019. Universal representation learning of knowledge bases by jointly embedding instances and ontological concepts. In *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, pages 1709–1719.

Aidan Hogan, Eva Blomqvist, Michael Cochez, Claudia d'Amato, Gerard de Melo, Claudio Gutierrez, Sabrina Kirrane, José Emilio Labra Gayo, Roberto Navigli, Sebastian Neumaier, et al. 2021. Knowledge graphs. *Synthesis Lectures on Data, Semantics, and Knowledge*, 12(2):1–257.

Guoliang Ji, Shizhu He, Liheng Xu, Kang Liu, and Jun Zhao. 2015. Knowledge graph embedding via dynamic mapping matrix. In *Proceedings of the 53rd annual meeting of the association for computational linguistics and the 7th international joint conference on natural language processing (volume 1: Long papers)*, pages 687–696.

Shaoxiong Ji, Shirui Pan, Erik Cambria, Pekka Marttinen, and S Yu Philip. 2021. A survey on knowledge graphs: Representation, acquisition, and applications. *IEEE Transactions on Neural Networks and Learning Systems*.

Michael I Jordan and Robert A Jacobs. 1994. Hierarchical mixtures of experts and the em algorithm. *Neural computation*, 6(2):181–214.

Jens Lehmann, Robert Isele, Max Jakob, Anja Jentzsch, Dimitris Kontokostas, Pablo N Mendes, Sebastian Hellmann, Mohamed Morsey, Patrick Van Kleef, Sören Auer, et al. 2015. Dbpedia–a large-scale, multilingual knowledge base extracted from wikipedia. *Semantic web*, 6(2):167–195.

Yankai Lin, Zhiyuan Liu, Maosong Sun, Yang Liu, and Xuan Zhu. 2015. Learning entity and relation embeddings for knowledge graph completion. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 29.

Farzaneh Mahdisoltani, Joanna Biega, and Fabian Suchanek. 2014. Yago3: A knowledge base from multilingual wikipedias. In *7th biennial conference*

on innovative data systems research. CIDR Conference.

Jeff Pasternack and Dan Roth. 2013. Latent credibility analysis. In *Proceedings of the 22nd international conference on World Wide Web*, pages 1009–1020.

Adam Paszke, Sam Gross, Soumith Chintala, Gregory Chanan, Edward Yang, Zachary DeVito, Zeming Lin, Alban Desmaison, Luca Antiga, and Adam Lerer. 2017. Automatic differentiation in pytorch.

Jeffrey Pennington, Richard Socher, and Christopher D Manning. 2014. Glove: Global vectors for word representation. In *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*, pages 1532–1543.

Wei Qian, Cong Fu, Yu Zhu, Deng Cai, and Xiaofei He. 2018. Translating embeddings for knowledge graph completion with relation attention mechanism. In *IJCAI*, pages 4286–4292.

Pengda Qin, Xin Wang, Wenhu Chen, Chunyun Zhang, Weiran Xu, and William Yang Wang. 2020. Generative adversarial zero-shot relational learning for knowledge graphs. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pages 8673–8680.

Michael Schlichtkrull, Thomas N Kipf, Peter Bloem, Rianne Van Den Berg, Ivan Titov, and Max Welling. 2018. Modeling relational data with graph convolutional networks. In *European semantic web conference*, pages 593–607. Springer.

Haseeb Shah, Johannes Villmow, Adrian Ulges, Ulrich Schwanecke, and Faisal Shafait. 2019. An open-world extension to knowledge graph completion models. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, pages 3044–3051.

Noam Shazeer, Azalia Mirhoseini, Krzysztof Maziarz, Andy Davis, Quoc Le, Geoffrey Hinton, and Jeff Dean. 2017. Outrageously large neural networks: The sparsely-gated mixture-of-experts layer. *arXiv preprint arXiv:1701.06538*.

Robert Stevens, Carole A Goble, and Sean Bechhofer. 2000. Ontology-based knowledge representation for bioinformatics. *Briefings in bioinformatics*, 1(4):398–414.

Zhiqing Sun, Zhi-Hong Deng, Jian-Yun Nie, and Jian Tang. 2019. Rotate: Knowledge graph embedding by relational rotation in complex space. *arXiv preprint arXiv:1902.10197*.

Komal Teru, Etienne Denis, and Will Hamilton. 2020. Inductive relation prediction by subgraph reasoning. In *International Conference on Machine Learning*, pages 9448–9457. PMLR.

Théo Trouillon, Johannes Welbl, Sebastian Riedel, Éric Gaussier, and Guillaume Bouchard. 2016. Complex embeddings for simple link prediction. In *International Conference on Machine Learning*, pages 2071–2080. PMLR.

Petar Veličković, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Lio, and Yoshua Bengio. 2017. Graph attention networks. *arXiv preprint arXiv:1710.10903*.

Peifeng Wang, Jialong Han, Chenliang Li, and Rong Pan. 2019. Logic attention based neighborhood aggregation for inductive knowledge graph embedding. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, pages 7152–7159.

Ruobing Xie, Zhiyuan Liu, Jia Jia, Huanbo Luan, and Maosong Sun. 2016. Representation learning of knowledge graphs with entity descriptions. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 30.

Liang Yao, Chengsheng Mao, and Yuan Luo. 2019. Kg-bert: Bert for knowledge graph completion. *arXiv preprint arXiv:1909.03193*.

Jing Zhang, Bo Chen, Lingxi Zhang, Xirui Ke, and Haipeng Ding. 2021. Neural, symbolic and neural-symbolic reasoning on knowledge graphs. *AI Open*, 2:14–35.

Yao Zhang, Xu Zhang, Jun Wang, Hongru Liang, Wenqiang Lei, Zhe Sun, Adam Jatowt, and Zhenglu Yang. 2020. Generalized relation learning with semantic correlation awareness for link prediction. *CoRR*, abs/2012.11957.

CETA: A Consensus Enhanced Training Approach for Denoising in Distantly Supervised Relation Extraction

Ruri Liu[1]*, **Shasha Mo**[1]†, **Jianwei Niu**[2], **Shengda Fan**[1]
[1] School of Cyber Science and Technology, Beihang University, Beijing 100191, China
[2] State Key Laboratory of Virtual Reality Technology and Systems,
School of Computer Science and Engineering, Beihang University, Beijing 100191, China
{ruriliu, moshasha, niujianwei, fanshengda}@buaa.edu.cn

Abstract

Distantly supervised relation extraction aims to extract relational facts from texts but suffers from noisy instances. Existing methods usually select reliable sentences that rely on potential noisy labels, resulting in wrongly selecting many noisy training instances or underutilizing a large amount of valuable training data. This paper proposes a sentence-level DSRE method beyond typical instance selection approaches by preventing samples from falling into the wrong classification space on the feature space. Specifically, a theorem for denoising and the corresponding implementation, named Consensus Enhanced Training Approach (CETA), are proposed in this paper. By training the model with CETA, samples of different classes are separated, and samples of the same class are closely clustered in the feature space. Thus the model can easily establish the robust classification boundary to prevent noisy labels from biasing wrongly labeled samples into the wrong classification space. This process is achieved by enhancing the classification consensus between two discrepant classifiers and does not depend on any potential noisy labels, thus avoiding the above two limitations. Extensive experiments on widely-used benchmarks have demonstrated that CETA significantly outperforms the previous methods and achieves new state-of-the-art results.

1 Introduction

Relation Extraction (RE), which aims to identify the relation between two specific entities in the text, is a fundamental task in natural language processing. Most supervised RE methods demand large-scale labeled training data, which is difficult to acquire manually. To alleviate the problem, Distant Supervision (DS) is proposed by (Mintz et al., 2009) to automatically generate the labeled text corpus by aligning the plain texts and knowledge bases. For example, as shown in Figure 1, $[Steve\ jobs, born, America]$ is a relational triple in the knowledge base, DS annotates all sentences that contain the entity pair $(Steve\ jobs, America)$ as valid instances for relation $born$. However, DS inevitably introduces noisy labels when the sentences do not express the labeled relation (e.g., cases $S2$ and $S3$ in Figure 1). Hence, investigating a denoising method against noisy labels has become an urgent demand for Distantly Supervised Relation Extraction (DSRE), which aims to train the unbiased RE model under DS-built dataset.

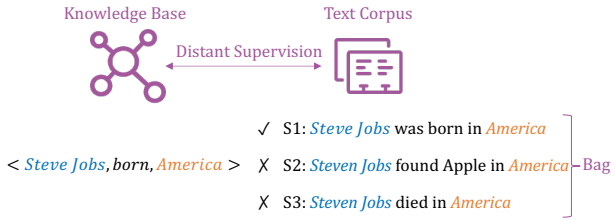

Figure 1: An example of annotating text corpus by distant supervision. $S2$ and $S3$ do not express the relation $born$ but still considered valid instances.

To alleviate the noise issue in DSRE, existing studies can be broadly classified into bag-level methods and sentence-level methods. The bag-level methods (Lin et al., 2016; Hu et al., 2019; Alt et al., 2019; Yuan et al., 2019) typically relax the relation label for each sentence to a bag, and then train the model by employing reliable bag-level representations. However, for bag-level methods, Feng et al. (2018); Jia et al. (2019) empirically verify that they cannot map each sentence to an explicit sentence label, resulting in inefficiency for sentence-level relation classification. From this perspective, several studies focus on sentence-level DSRE, which aims to select the reliable sentences for training and regard the sentence as a basic testing unit. Existing methods (Feng et al., 2018; Qin et al., 2018b; Han et al., 2018b; Zeng et al., 2018)

*The first two authors contributed equally.
†Corresponding authors.

Proceedings of the 29th International Conference on Computational Linguistics, pages 2247–2258
October 12–17, 2022.

usually apply reinforcement learning or adversarial learning to train sentence selector by receiving feedback from the manually crafted reward function, or train the model to start with the reliable sentences selected by frequent patterns (Jia et al., 2019). Finally, these methods select trustable sentences whose predicted labels are consistent with DS-annotated labels. However, these methods may be trapped by some common noisy instances whose model-predicted labels and DS-annotated labels are both wrong (Li et al., 2020b). Besides, the patterns of many correct sentences do not match the frequent patterns, resulting in much valuable information being discarded, limiting the capability of the trained model.

This work proposes a sentence-level DSRE method beyond typical instance selection approaches by preventing samples from falling into the wrong classification space on the feature space. Specifically, a theorem for denoising and the corresponding implementation, named Consensus Enhanced Training Approach (CETA), are proposed in this paper. By training the model with CETA, samples of different classes are separated, and samples of the same class are closely clustered in the feature space. As a result, the robust classification boundary can be easily established to prevent noisy labels from biasing samples into the wrong classification space. Compared with existing sentence-level DSRE methods, CETA performs denoising by enhancing the classification consensus between two discrepant classifiers within the model and does not depend on any potential noisy labels. Therefore, when dealing with noisy labels, CETA does not get trapped by common noisy instances. In addition, CETA enables the model to be trained on all data, and the effect of noisy labels is eliminated in the feature space instead of directly filtering sentences as in previous methods.

Contributions of this paper can be summarized as follows:

- This paper proposes and proves a theorem for denoising that enhancing the prediction consistency between two different classifiers in a model can reduce the impact of noisy instances.
- With the support of the proposed theorem, our proposed CETA facilitates the model to separate the samples of the different classes and cluster the samples of the same class. As a result, a robust classification boundary can be established to reduce the impact of noisy labels.
- Evaluations on widely-used datasets of DSRE demonstrate that CETA significantly outperforms the previous state-of-the-art models.

2 Related Work

We discuss two lines of related work as follows.

DSRE. DS is an effective approach to annotate texts, but suffers from the noisy labels. Most existing studies of DSRE are bag-level DSRE methods, which apply multi-instance learning to handle noisy sentences in each bag and train models by exploiting the constructed reliable bag-level representations. These methods usually utilize attention mechanisms to assign small weights to the potential noisy sentences in the bag (Lin et al., 2016; Han et al., 2018c; Alt et al., 2019; Hu et al., 2019; Yuan et al., 2019; Li et al., 2020a), apply adversarial training or reinforcement learning to remove the noisy sentences from the bag. (Zeng et al., 2015; Qin et al., 2018b; Han et al., 2018b; Shang et al., 2020) However, the studies (Feng et al., 2018; Jia et al., 2019) indicate that the bag-level DSRE methods are ineffective for sentence-level prediction.

This paper focus on sentence-level relation extraction. Some recent studies also regard the sentence as a basic training unit and perform denoising by applying reinforcement learning to select reliable instances based on the reward of noisy labels (Feng et al., 2018), building initial reliable sentences based on several manually defined frequent relation patterns (Jia et al., 2019), assigning the complementary labels cooperating with the negative training to filter noisy instances (Ma et al., 2021), and utilizing meta learning to exploit the extra clean reference data (Li et al., 2020b). Different from the previous works, our proposed method does not rely on the noisy labels, frequent relation patterns, and handles the noisy instances in the feature space without any extra clean reference data.

Supervised learning with noisy labels. In both computer vision and natural language processing, many methods have been proposed to train models with noisy labels and these methods can be broadly classified into: robust regularization (Krogh and Hertz, 1991; Müller et al., 2019; Qu et al., 2021; Zhou and Chen, 2021), robust loss function (Zhang and Sabuncu, 2018; Wang et al., 2019a), label reweighting (Chang et al., 2017; Wang et al., 2019b), noise filtering adaption layers (Goldberger and Ben-Reuven, 2016) and sample selection (Han et al., 2018a; Yu et al., 2019; Wei et al., 2020). In particu-

lar, The methods (Zhou and Chen, 2021; Wei et al., 2020) train two or more models simultaneously and regularize their predictions to be similar, which can be considered as consensus enhancement, but are computationally expensive and affected by the number of trained models. Different from these methods, our method achieves denoising by enhancing the consensus of predictions between two discrepant classifiers within a model, which is computationally friendly and guaranteed by a proven theorem.

3 Methodology

In this section, we start with the learning setup and present the objective function in conjunction with our proposed theorem: the generalization error bound for DSRE in subsection 3.1. Then, in subsection 3.2, we will introduce the details of our proposed CETA, which aims to implement the proposed theorem for denoising.

3.1 Generalization Error Bound for DSRE

Formally, the DS-built training set can be denoted as $\mathcal{D}^{cs} = \{(x_i, y_i^{cs})\}_{i=1}^{m} \in (\mathcal{X} \times \mathcal{Y})$, where $x_i \in \mathcal{X}$ and $y_i^{cs} \in \mathcal{Y}$. $\mathcal{X}$ represents the input instances. $\mathcal{Y}$ indicates the class labels. y^{cs} indicates that the label may be a clean label y^c or a noisy label y^s. The relation extraction model based on neural networks is usually composed of a sentence encoder and a classifier. The sentence encoder $g : \mathcal{X} \to \mathcal{Z}$ maps input instances $\mathcal{X}$ into feature space $\mathcal{Z}$. The classifier $f : \mathcal{Z} \to \mathcal{Y}$ establishes the classification boundary in the feature space and maps the features of the instances into labels $\mathcal{Y}$. In order to evaluate the performance of the model, we denote the clean test set as $\mathcal{D}^c = \{(x_i, y_i^c)\}_{i=1}^{n} \in (\mathcal{X} \times \mathcal{Y})$, where $x_i \in \mathcal{X}$ and $y_i^c \in \mathcal{Y}$. We use $\epsilon^c(f)$ to denote the expected error of the model. The goal of DSRE is to reduce $\epsilon^c(f)$ under the DS-built training set.

The supervised methods usually reduce the expected error based on the structural risk minimization whose basic theorem requires the training set and the test set to come from the same distribution. which is unsuitable for DSRE since the DS-built training set cannot be as clean as the test data. In this paper, the basic theorem is extended to DSRE, a generalization error bound to measure the expected error is theoretically proposed as follows[1].

Theorem 1. *Let g be a fixed representation function from $\mathcal{X}$ to $\mathcal{Z}$, $\mathcal{F}$ be the hypothesis class of Vapink Chervonenkis d. If a random sample of size m $\mathcal{Z}^{cs}$ is generated by applying g to a $\mathcal{D}^{cs}$ - i.i.d. for any $\sigma > 0$, with probability $1 - \sigma$, we have the following uniform generalization error bound for any classification functions $f \in \mathcal{F}$,*

$$\epsilon^c(f) \leq \hat{\epsilon^{cs}}(f) + \frac{1}{2} d_{F \triangle \mathcal{F}}(\mathcal{Z}^{cs}) + \lambda, \quad (1)$$

where

$$\hat{\epsilon^{cs}}(f) = \frac{1}{m} \sum_{i=1}^{m} \left| \hat{f}(z_i^{cs}) - y_i^{cs} \right|$$

$$d_{\mathcal{F}\Delta\mathcal{F}}(\mathcal{Z}^{cs}) = 2 \sup_{f', f'' \in \mathcal{F}} \left| \Pr\left[f'(z^{cs}) \neq f''(z^{cs}) \right] \right|$$

$$f^* = arg\min_{f \in \mathcal{F}} \epsilon^c(f) + \epsilon^{cs}(f)$$

$$\lambda = \epsilon^{cs}(f^*) + \epsilon^c(f^*) + \sqrt{\frac{4}{m}\left(d \log \frac{2em}{d} + \log \frac{4}{\delta} \right)}$$

Eq. (1) demonstrates that the expected error $\epsilon^c(f)$ can be bounded by using three terms. The corresponding explanations are as follows.

1. $\hat{\epsilon^{cs}}(f)$ is the empirical error of the DS-built training data $\mathcal{D}^{cs}$.

2. $d_{F \triangle \mathcal{F}}(\mathcal{Z}^{cs})$ is a key novelty of this theorem and can be viewed as the regularization term that represents the upper bound (sup) of the probability (Pr) that two classifiers f' and f'' category the feature z^{cs} into different classes.

3. λ indicates the shared error of the ideal joint hypothesis (f^*). λ is a constant and can be ignored during training stage.

Based on our proposed Theorem 1, a new denoising method is pointed out, that is, the expected error can be reduced by reducing the generalization error bound. In particular, if $d_{F \triangle \mathcal{F}}(\mathcal{Z}^{cs})$ is minimized, the feature z^{cs} will be classified into the same class by f' and f'' with a higher probability, which is equivalent to enhancing the consistency of model predictions by two discrepant classifiers.

3.2 Consensus Enhanced Training Approach

CETA aims to reduce the expected error $\epsilon^c(f)$ by minimizing $d_{F \triangle \mathcal{F}}(\mathcal{Z}^{cs})$ and $\hat{\epsilon^{cs}}(f)$ in the generalization error bound proposed in Theorem 1. We introduce the architecture of CETA and the optimization strategy of CETA in sequence.

[1]The detailed proof can be found in Appendix A.

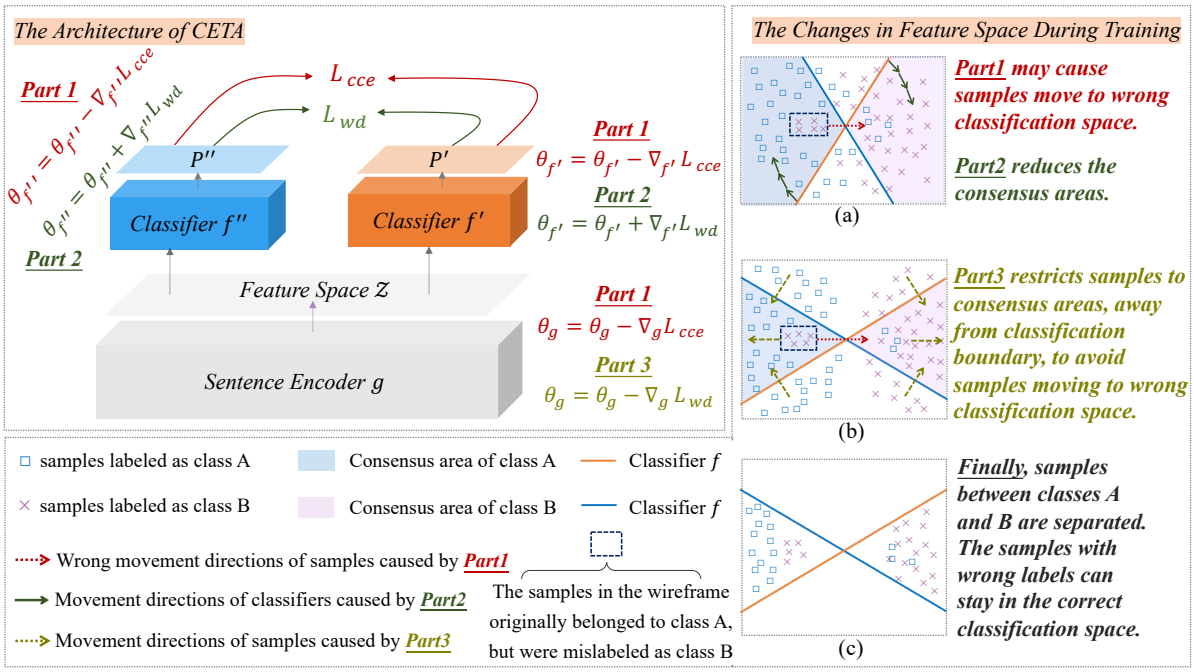

Figure 2: The left part is the architecture and optimization strategy of CETA. The right part is the corresponding changes in the feature space when the optimization algorithm acts on different components. $\mathcal{L}_{cce}$ is the categorical cross-entropy loss function defined in Eq. (2). $\mathcal{L}_{wd}$ is the divergence loss based on wasserstein distance defined in Eq. (3). The symbol Θ indicates the component's parameters of the model. The symbol ∇ denotes the component's gradient calculated by the corresponding loss. The consensus area refers to the area where two classifiers classify the sample into the same class.

.

3.2.1 Architecture of CETA

As shown in Figure 2, the architecture of CETA consists of two classifiers sharing an encoder. Given the input instance x, the encoder g transforms the x from the instance space $\mathcal{X}$ to the feature space $\mathcal{Z}$, the corresponding hidden feature vector is denoted as: $(z_1, z_2, \ldots z_{e_1} \ldots z_{e_2} \ldots z_L)$, where z_{e_1} and z_{e_2} are the feature vectors corresponding to the entities e_1 and e_2. We can obtain the instance representation $z = [z_{e_1}; z_{e_2}]$ for classification by concatenating z_{e_1} and z_{e_2}. In particular, CETA adds an auxiliary classifier, which is not only used to reduce the empirical loss $\hat{\epsilon^{cs}}(f)$, but also aims to use two classifiers to approximate f' and f'' in $d_{F\triangle\mathcal{F}}(\mathcal{Z}^{cs})$, and then combine the proposed optimization strategy to reduce $d_{F\triangle\mathcal{F}}(\mathcal{Z}^{cs})$.

3.2.2 Optimization Strategy of CETA

The optimization strategy of CETA can be broadly divided into three parts. The first part aims to reduce the empirical loss $\hat{\epsilon^{cs}}(f)$. The second part and the third part are combined to reduce the $d_{F\triangle\mathcal{F}}(\mathcal{Z}^{cs})$. The details are as follows.

Part 1. To reduce $\hat{\epsilon^{cs}}(f)$, we adopt the categorical cross-etropy function to calculate the classification loss $\mathcal{L}_{cce}$ of the noisy training set.

$$\mathcal{L}_{cce} = -\frac{1}{M}\sum_{i=1}^{M}\sum_{k=1}^{K}\mathbb{I}(y_i = k)\left[\log(p'_{ik}) + \log(p''_{ik})\right] \tag{2}$$

where M is the number of training instances, K is the number of relation classes. $\mathbb{I}(y_i = k)$ is an indicator function, which returns 1 when $y_i = k$, and 0 otherwise. The p'_{ik} and p''_{ik} are two probabilities that the instance i belongs to class k predicted by two classifiers f' and f'', respectively. As shown in Figure 2, we use $\mathcal{L}_{cce}$ to calculate the gradient on each component of the model. Then we update the parameters of each component to reduce the empirical error $\hat{\epsilon^{cs}}(f)$. In feature space, reducing $\hat{\epsilon^{cs}}(f)$ is equivalent to forcing the wrongly labeled samples to move to the wrong classification space, and the direction of the movement is indicated by the red arrow in the right part of Figure 2.

Part 2. To reduce $d_{F\triangle\mathcal{F}}(\mathcal{Z}^{cs})$, the goal of this part is to increase the discrepancy between the two classifiers, so as to approximate the $d_{F\triangle\mathcal{F}}(\mathcal{Z}^{cs})$ with two discrepant classifiers. Specifically, CETA first adopts the wasserstein distance (Kantorovich, 2006) to capture the discrepancy $\mathcal{L}_{wd}$ between $\mathcal{P}'$

and $\mathcal{P}''$ for measuring the classification discrepancy between two classifiers f' and f''. $\mathcal{P}'$ and $\mathcal{P}''$ are two probability distributions representing the probabilities of the samples being divided into different classes by f' and f'', respectively.

$$\mathcal{L}_{wd} = \underset{\gamma^* \in \Pi[\mathcal{P}', \mathcal{P}'']}{argmin} \mathbb{E}_{(p', p'') \sim \gamma} \|p\prime - p\prime\prime\| \quad (3)$$

$$\mathcal{P}' = f'(\mathcal{Z}^{cs}), \mathcal{P}'' = f''(\mathcal{Z}^{cs}) \quad (4)$$

where $\Pi[\mathcal{P}', \mathcal{P}'']$ indicates the set of all jointed distributions whose marginals are $\mathcal{P}'$ and $\mathcal{P}''$. Calculating $\mathcal{L}_{wd}$ can be regarded as finding the optimal $\gamma*$. $\gamma*$ can transform $\mathcal{P}'$ into $\mathcal{P}''$ with minimal modification.

Then, CETA uses $\mathcal{L}_{wd}$ to calculate the gradient $\nabla_{f'}\mathcal{L}_{wd}$, $\nabla_{f''}\mathcal{L}_{wd}$, and $\nabla_g\mathcal{L}_{wd}$ of two classifiers f', f'' and the sentence encoder g, respectively.

$$\theta_{f'} = \theta_{f'} + \nabla_{f'}\mathcal{L}_{wd} \quad (5)$$

$$\theta_{f''} = \theta_{f''} + \nabla_{f''}\mathcal{L}_{wd} \quad (6)$$

Since the $d_{F\triangle\mathcal{F}}(\mathcal{Z}^{cs})$ refers to the *upper bound* of the probability that two classifiers divide the sample into different classes. In order to approximate the *upper bound of classification discrepancy* of $d_{F\triangle\mathcal{F}}(\mathcal{Z}^{cs})$, we update the parameters $\theta_{f'}$ and $\theta_{f''}$ by executing Eq. (5) and Eq. (6). It can increase the classification discrepancy between the two classifiers. As shown in the right side of Figure 2, it can reduce the consensus area in the feature space. The consensus area refers to the area where two classifiers classify the sample into the same class. The smaller the consensus area, the greater the discrepancy between the classifiers. When the discrepancy between the two classifiers reaches a threshold value, the discrepancy between these two classifiers is approximately equal to $d_{F\triangle\mathcal{F}}(\mathcal{Z}^{cs})$.

Part 3. On the basis that the discrepancy between the two classifiers can well approximate $d_{F\triangle\mathcal{F}}(\mathcal{Z}^{cs})$, executing Eq. (7) can reduce the discrepancy between the two classifiers, which is equivalent to reducing $d_{F\triangle\mathcal{F}}(\mathcal{Z}^{cs})$. As shown in the right side of Figure 2, Eq. (7) is applied to the sentence encoder g, which can change the distribution of samples in the feature space. The samples encoded by g will enter the narrow consensus area and be pulled away from each other, only in this way the classification discrepancy between two classifiers can be reduced. After three parts of the optimization strategy, the samples of different classes are separated in the feature space, and the distance between clusters of different classes is enlarged. So that the wrongly labeled samples cannot be easily moved to the wrong classification space and stay in the original correct classification space.

$$\theta_g = \theta_g - \nabla_g\mathcal{L}_{wd} \quad (7)$$

The complete training steps of CETA are summarized in Algorithm 1. Besides, only sentence encoder g and the classifier f'' are adopted to predict the relation type during the inference procedure.

Algorithm 1 CETA Algorithm

Input: training sets $\mathcal{D}^{cs}$, β, learning rate η, sentence encoder θ_g , full connected layers as classifier $\theta_{f'}$ and $\theta_{f''}$, epoch T, iteration N
Output:$\theta_{f'}$, $\theta_{f''}$, θ_g

for $t = 1, 2, 3, ..., T$ **do**
  Shuffle training set $\mathcal{D}^{cs}$
  for $n = 1, 2, 3, ..., N$ **do**
    Fetch mini-batch $\bar{cs}$ from $\mathcal{D}^{cs}$
    Calculate $\mathcal{L}_{cce}$ and $\mathcal{L}_{wd}$ on $\bar{cs}$
    Update $\theta_{f'} \leftarrow \theta_{f'} - \nabla_{f'}\mathcal{L}_{cce}$
    Update $\theta_{f''} \leftarrow \theta_{f''} - \nabla_{f''}\mathcal{L}_{cce}$
    Update $\theta_g \leftarrow \theta_g - \nabla_g\mathcal{L}_{cce}$
    Update $\theta_{f'} \leftarrow \theta_{f'} + \nabla_{f'}\mathcal{L}_{wd}$
    Update $\theta_{f''} \leftarrow \theta_{f''} + \nabla_{f''}\mathcal{L}_{wd}$
    Update $\theta_g \leftarrow \theta_g - \beta\nabla_g\mathcal{L}_{wd}$
  end for
end for

4 Experiments

The experiments in this work are divided into two part. (1) The first part is the effectiveness study on sentence-level evaluation for our method and the compared methods. Many previous bag-level DSRE methods adopt held-out evaluation, where both training set and test sets are DS-built. However, the studies (Gao et al., 2021; Feng et al., 2018) have demonstrated that the bias is inevitably introduced into held-out evaluations since the DS-build test set is noisy. To provide more accurate and credible evaluations, this part of experiment follows most sentence-level DSRE methods that conduct sentence-level evaluations on benchmarks with clean test sets. (2) The second part is the ablation experiments, which adopts feature visualization to better illustrate the behaviors of our proposed CETA.

Benchmarks		NYT	KBP
#Label num		24	6
Train	Instances	371,461	151,091
	Positive	110,518	38,922
Test	Instances	2,164	4,168
	Positive	323	1,075

Table 1: Statistics of benchmarks. "Positive" means positive instances that are not labeled as "NA". "NA" indicates that the sample does not belong to any of the predefined relation labels.

4.1 Benchmarks

We evaluate our method on two widely-used DSRE benchmarks: NYT and KBP, and the dataset statistics are shown in Table 1.

NYT. This dataset is developed by Riedel et al. (2010) aligning New York Times corpus with the relation facts in Freebase. The origin training set and test set are both DS-built. To make the evaluation more precisely, we adopt the original training set and a widely-used manually annotated test set provided by Jia et al. (2019).

KBP. This dataset is constructed by Ling and Weld (2012) aligning English Wikipedia corpus with the relation facts in Freebase as training set. and the test set is built by utilizing the manually-annotated sentences from 2013 KBP (Ellis et al., 2012). However, some test relation types have no or only one training instance. Besides, this test set only contains 165 positive instances. To reduce the bias of evaluation, we utilize the other refined version of KBP proposed by Li et al. (2020b) to avoid the above problems. Our adopted test set has the same relation types with the training data, contains more positive instances, and keep the same proportion of positive instances as the training set.

4.2 Baseline Models

Our proposed CETA is a sentence-level DSRE method. For the fairness of the comparison, we compare with several strong DSRE methods. These compared methods can be categorized as: bag-level DSRE methods, sentence-level DSRE methods, sentence-level RE methods without denoising.

- **PCNN+ATT** (Lin et al., 2016) A bag-level DSRE method which employs the selective attention to alleviate noise.
- **PCNN+RA_BAG_ATT** (Ye and Ling, 2019) A bag-level DSRE method which utilizes inter-bag and intra-bag attentions to reduce the impact of noisy instances.
- **CNN+RL$_1$** (Qin et al., 2018b) A bag-level DSRE method which applies reinforcement learning to generate the false-positive indicator to recognize false positives, and redistribute the filtered data into the negative examples.
- **CNN+RL$_2$** (Feng et al., 2018) A sentence-level DSRE model which employs reinforcement learning to jointly train a RE model for relation classification and an instance selector for filtering the potential noisy instances.
- **PCNN+DSGAN** (Qin et al., 2018a) A sentence-level DSRE model which adopts adversarial learning to train a generator to recognize true positive instances, and then redistributes the remaining false positives to the negative set to obtain a new cleaned dataset.
- **ARNOR** (Jia et al., 2019) A sentence-level DSRE model which selects the reliable instances based on the reward of attention score on the selected patterns.
- **SENT** (Ma et al., 2021) A sentence-level DSRE model which filters noisy instances and relabeling based on negative training. It is the state-of-the-art method in sentence level.
- **CNN** (Zeng et al., 2014), **PCNN** (Zeng et al., 2015), **BiLSTM** (Zhang et al., 2015) and **BERT** (Devlin et al., 2019) are commonly-used models for RE without denoising methods.

4.3 Implementation Details

Our proposed CETA is a model-agnostic sentence-level DSRE method. We implement CETA using BiLSTM, PCNN, and BERT as sentence encoder, respectively[2].

When implemented with BiLSTM, our adopted word embedding are 50-dimensional Glove word embedding published by Lin et al. (2016). Besides, we utilize 50-dimension randomly initialized position and entity type embedding. The BiLSTM is single layer with hidden size 256 and optimized by Adam optimizer with the learning rate of 5e-4. All the adopted word embedding, position and

[2]The code and training scripts will be released at https://github.com/Ethan-RR/CETA

Model	NYT		
	Prec.	Rec.	F1
CNN*	35.75	64.54	46.01
PCNN*	36.06	64.86	46.35
BiLSTM*	35.52	67.41	46.53
BERT*	36.21	70.41	47.82
PCNN+ATT*	45.41	30.03	36.15
PCNN+RA_BAG_ATT*	56.76	50.60	53.50
CNN+RL_1*	39.41	61.61	48.07
CNN+RL_2*	40.23	63.78	49.34
BiLSTM+ARNOR*	65.23	56.79	60.90
BiLSTM+SENT*	71.22	59.75	64.99
BERT+BiLSTM+SENT*	76.34	63.66	69.42
BiLSTM+CETA	**71.34**	61.12	**65.83**
BERT+CETA	**63.98**	**69.13**	**66.45**
BERT+BiLSTM+CETA	76.29	**64.63**	**69.98**

Table 2: Main results of the sentence-level evaluation on NYT. Compared baselines include normal RE model (the first part of the table) and models for distant RE (the second part of the table). We run our experiment 5 times and report the average result. The results with * are reported in Ma et al. (2021).

Model	KBP		
	Prec.	Rec.	F1
PCNN*	56.12	33.38	41.75
BiLSTM †	57.10	47.06	51.48
PCNN+ATT*	72.65	29.24	41.69
PCNN+RL_1*	57.64	38.79	46.32
PCNN+DSGAN*	59.86	38.54	46.65
BiLSTM+ARNOR*	54.83	34.59	42.35
PCNN+SENT	59.98	32.78	42.39
BiLSTM+CETA	**59.74**	**56.15**	**57.89**
PCNN+CETA	58.72	**39.83**	**47.46**

Table 3: Main results of the sentence-level evaluation on KBP. We run our experiment 5 times and report the average result of our proposed method. The results with * and † are reported in Li et al. (2020b) and Li et al. (2022), respectively.

entity type embedding, the hyperparameters of BiLSTM and the optimizer are consistent with SENT. When implemented with PCNN, the size of position embeddings are 30 dimensions. The number of convolution filter for PCNN model is 230, and the filter window size is 3, which keeps the same with Li et al. (2020b). When implemented with BERT, we use *bert-base-uncased* as sentence encoder and apply AdamW optimizer with a learning rate of 2e-5. The above experimental setup is also applied to the compared RE model that utilizes BERT without denoising method.

We determine the best hyperparameters by grid search. Specifically, when training on the NYT and KBP datasets, we train the model for 10 epochs with a batch size of 256 when using BiLSTM and a batch size of 16 when using BERT. The optimal values of the scalar β for scaling gradients on the sentence encoder mentioned in Algorithm 1 are $\beta = 2.1$ for BiLSTM and $\beta = 4.7$ for BERT.

4.4 Sentence-Level Evaluation

we adopt the same evaluation metrics as the previous sentence-level DSRE method (Jia et al., 2019; Li et al., 2020b; Ma et al., 2021): Micro-Precision (Prec.), Micro-Recall (Rec.) and Micro-F1 (F1). The sentence-level evaluation results of our proposed CETA and other baselines are on NYT and KBP are shown in Table 2 and Table 3, respectively. We can observe that: (1) Our proposed CETA surpasses all baselines in F1 metrics on both KBP and NYT datasets when applying the same sentence encoder. (2) The results on the NYT dataset show that CETA has higher Rec. than both ARNOR and the current state-of-the-art SENT when LSTM is used as the basic sentence encoder. In addition, the results on the KBP dataset show that when PCNN is used as the basic sentence encoder, the Rec. of CETA is higher than that of other baselines, which shows that CETA can facilitate the model to fully exploit the training data. (3) The F1 metrics of ARNOR on the NYT dataset is significantly higher than that of RL_1, but on the KBP dataset, the F1 metrics of ARNOR is lower than that of RL_1. It shows that the performance of ARNOR is susceptible to different data distributions. Our proposed method consistently outperforms on both NYT and KBP datasets. We believe that the stability of the method is important for practical scenarios.

In addition, we conduct a hyperparameter-tuning study about the number of classifiers of CETA on NYT in Appendix B.

4.5 Ablation Study

The section 3.2 has demonstrated that training the model with CETA allows the model to separate

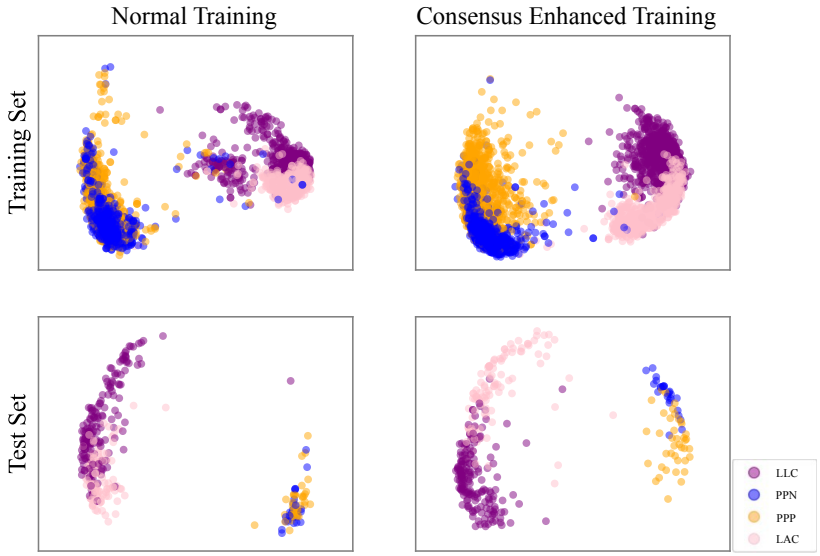

Figure 3: The visualization of instances' representations. The first column and second column show the representations produced by the sentence encoders trained by the *normal training method* and the *consensus enhanced training method*, respectively. The first row and second row show the results of 3750 and 410 random sampled instances from the noisy training set of and the clean test set of our adopted NYT, respectively.

samples between different classes, thereby reducing the possibility of mislabeled samples entering the wrong classification space, making it easier for the model to establish robust classification boundaries. To further verify this proposal, we conduct an ablation experiment that training our proposed CETA with two different training methods on the NYT dataset. The specific steps of this experiment are as follows.

The first training method only reduces the empirical error $\hat{\epsilon^{cs}}(f)$ by performing steps 6, 7, and 8 in our proposed Algorithm 1. We call the first training method *normal training method*. The second training method reduces both the empirical error $\hat{\epsilon^{cs}}(f)$ and the classification- consensus-related term $d_{F\triangle\mathcal{F}}(\mathcal{Z}^{cs})$ proposed in Theorem 1 by performing all steps in Algorithm 1. We call the second training method *consensus enhanced training method*. The *normal training method* and the *consensus enhanced training method* adopt the same experimental environment, relation extraction model (We adopt BERT model as sentence encoder), and hyperparameters. Second, we pick four main classes[3] of instances from the noisy training set and the clean test set of NYT. The randomly picked instances are mapped into representations by two sentence encoders that are trained by *normal training method* and *consensus enhanced training method*, respectively. Third, we adopt Principal Component Analysis (PCA) to reduce dimension of the representations and visualize these 2-dimension representations in Figure 3.

From the visualization of instances' representations plotted in Figure 3, we can observe that our proposed *consensus enhanced training method* facilitates model to closely cluster the representations of the same class's samples and clearly separate the representations of different classes' samples compared with the *normal training method*.

In addition, we perform sentence-level evaluation on the clean test set of NYT, and the model trained by *normal training method* achieves the following results: $Prec. = 36.21$, $Rec. = 70.41$, $F1 = 47.82$. The result of the model trained by *consensus enhanced training method* is as follows: $Prec. = 63.98$, $Rec. = 69.13$, $F1 = 66.45$. We can observe that the results obtained by the model trained by *consensus enhanced training method* lead across the board, strongly demonstrating the effectiveness of CETA.

5 Conclusion

This paper goes beyond the typical instance selection approaches, and focuses on handling the noisy labels in the feature space. A theorem for denoising and the corresponding implementation, named Consensus Enhanced Training Approach (CETA),

[3]Four selected classes are: (1) /Location/Location/Contains (LLC), (2) /People/Person/Nationality (PPN), (3) /People/Person/Place lived (PPP) and (4) /Location/Administrative division/Country (LAC), respectively.

are proposed in this paper. By training the model with CETA, samples of different classes are separated in the feature space. Thus the model can easily establish the robust classification boundary to prevent noisy labels from biasing wrongly labeled samples into the wrong classification space. Besides, CETA achieves denoising does not depend on any potential noisy labels. Therefore, CETA is not affected by common noisy instances. Extensive experiments on the widely-used benchmarks have demonstrated that our proposed CETA outperforms previous methods.

Acknowledgements

We thank anonymous reviewers for their responsible attitude and helpful comments. This work was supported by National Natural Science Foundation of China (62106013).

References

Christoph Alt, Marc Hübner, and Leonhard Hennig. 2019. Fine-tuning pre-trained transformer language models to distantly supervised relation extraction. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 1388–1398, Florence, Italy. Association for Computational Linguistics.

S. Ben-David, J. Blitzer, K. Crammer, A. Kulesza, F. Pereira, and J. W. Vaughan. 2010. A theory of learning from different domains. *Machine Learning*, 79(1-2):151–175.

Haw-Shiuan Chang, Erik Learned-Miller, and Andrew McCallum. 2017. Active bias: Training more accurate neural networks by emphasizing high variance samples. *Advances in Neural Information Processing Systems*, 30.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, Minneapolis, Minnesota. Association for Computational Linguistics.

Joe Ellis, Jeremy Getman, Justin Mott, Xuansong Li, Kira Griffitt, Stephanie Strassel, and Jonathan Wright. 2012. Linguistic resources for 2013 knowledge base population evaluations. *Theory and Applications of Categories*.

Jun Feng, Minlie Huang, Li Zhao, Yang Yang, and Xiaoyan Zhu. 2018. Reinforcement learning for relation classification from noisy data. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 32.

Tianyu Gao, Xu Han, Yuzhuo Bai, Keyue Qiu, Zhiyu Xie, Yankai Lin, Zhiyuan Liu, Peng Li, Maosong Sun, and Jie Zhou. 2021. Manual evaluation matters: Reviewing test protocols of distantly supervised relation extraction. In *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*, pages 1306–1318, Online. Association for Computational Linguistics.

Jacob Goldberger and Ehud Ben-Reuven. 2016. Training deep neural-networks using a noise adaptation layer.

Bo Han, Quanming Yao, Xingrui Yu, Gang Niu, Miao Xu, Weihua Hu, Ivor Tsang, and Masashi Sugiyama. 2018a. Co-teaching: Robust training of deep neural networks with extremely noisy labels. *Advances in neural information processing systems*, 31.

Xu Han, Zhiyuan Liu, and Maosong Sun. 2018b. Denoising distant supervision for relation extraction via instance-level adversarial training. *arXiv preprint arXiv:1805.10959*.

Xu Han, Zhiyuan Liu, and Maosong Sun. 2018c. Neural knowledge acquisition via mutual attention between knowledge graph and text. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 32.

Linmei Hu, Luhao Zhang, Chuan Shi, Liqiang Nie, Weili Guan, and Cheng Yang. 2019. Improving distantly-supervised relation extraction with joint label embedding. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 3812–3820.

Wei Jia, Dai Dai, Xinyan Xiao, and Hua Wu. 2019. Arnor: Attention regularization based noise reduction for distant supervision relation classification. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 1399–1408.

L. V. Kantorovich. 2006. On the translocation of masses. *Journal of Mathematical Sciences*, 133(4):1381–1382.

Anders Krogh and John Hertz. 1991. A simple weight decay can improve generalization. *Advances in neural information processing systems*, 4.

Yang Li, Guodong Long, Tao Shen, Tianyi Zhou, Lina Yao, Huan Huo, and Jing Jiang. 2020a. Self-attention enhanced selective gate with entity-aware embedding for distantly supervised relation extraction. In *Proceedings of the AAAI conference on artificial intelligence*, volume 34, pages 8269–8276.

Zhenzhen Li, Jian-Yun Nie, Yiping Song, Pan Du, and Dongsheng Li. 2022. Learning to classify relations between entities from noisy data - a meta instance reweighting approach. *Expert Systems with Applications*, 202:117113.

Zhenzhen Li, Jian-Yun Nie, Benyou Wang, Pan Du, Yuhan Zhang, Lixin Zou, and Dongsheng Li. 2020b. Meta-learning for neural relation classification with distant supervision. In *Proceedings of the 29th ACM International Conference on Information & Knowledge Management*, pages 815–824.

Y. Lin, S. Shen, Z. Liu, H. Luan, and M. Sun. 2016. Neural relation extraction with selective attention over instances. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*.

Xiao Ling and Daniel S Weld. 2012. Fine-grained entity recognition. In *Twenty-Sixth AAAI Conference on Artificial Intelligence*.

Ruotian Ma, Tao Gui, Linyang Li, Qi Zhang, Yaqian Zhou, and Xuanjing Huang. 2021. Sent: Sentence-level distant relation extraction via negative training. *arXiv preprint arXiv:2106.11566*.

M. Mintz, S. Bills, R. Snow, and D. Jurafsky. 2009. Distant supervision for relation extraction without labeled data. In *ACL 2009, Proceedings of the 47th Annual Meeting of the Association for Computational Linguistics and the 4th International Joint Conference on Natural Language Processing of the AFNLP, 2-7 August 2009, Singapore*.

Rafael Müller, Simon Kornblith, and Geoffrey E Hinton. 2019. When does label smoothing help? *Advances in neural information processing systems*, 32.

Pengda Qin, Weiran Xu, and William Yang Wang. 2018a. DSGAN: Generative adversarial training for distant supervision relation extraction. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 496–505, Melbourne, Australia. Association for Computational Linguistics.

Pengda Qin, Weiran Xu, and William Yang Wang. 2018b. Robust distant supervision relation extraction via deep reinforcement learning. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 2137–2147.

Yuntao Qu, Shasha Mo, and Jianwei Niu. 2021. Dat: Training deep networks robust to label-noise by matching the feature distributions. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 6821–6829.

Sebastian Riedel, Limin Yao, and Andrew McCallum. 2010. Modeling relations and their mentions without labeled text. In *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*, pages 148–163. Springer.

B Schölkopf, J. Platt, and T. Hofmann. Analysis of representations for domain adaptation. In *Advances in Neural Information Processing Systems 19: Proceedings of the 2006 Conference*.

Yuming Shang, He-Yan Huang, Xian-Ling Mao, Xin Sun, and Wei Wei. 2020. Are noisy sentences useless for distant supervised relation extraction? In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pages 8799–8806.

Vladimir N Vapnik. 1999. An overview of statistical learning theory. *IEEE transactions on neural networks*, 10(5):988–999.

Yisen Wang, Xingjun Ma, Zaiyi Chen, Yuan Luo, Jinfeng Yi, and James Bailey. 2019a. Symmetric cross entropy for robust learning with noisy labels. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 322–330.

Zihan Wang, Jingbo Shang, Liyuan Liu, Lihao Lu, Jiacheng Liu, and Jiawei Han. 2019b. Crossweigh: Training named entity tagger from imperfect annotations. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 5154–5163.

Hongxin Wei, Lei Feng, Xiangyu Chen, and Bo An. 2020. Combating noisy labels by agreement: A joint training method with co-regularization. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 13726–13735.

Zhi-Xiu Ye and Zhen-Hua Ling. 2019. Distant supervision relation extraction with intra-bag and inter-bag attentions. In *NAACL-HLT (1)*.

Xingrui Yu, Bo Han, Jiangchao Yao, Gang Niu, Ivor Tsang, and Masashi Sugiyama. 2019. How does disagreement help generalization against label corruption? In *International Conference on Machine Learning*, pages 7164–7173. PMLR.

Yujin Yuan, Liyuan Liu, Siliang Tang, Zhongfei Zhang, Yueting Zhuang, Shiliang Pu, Fei Wu, and Xiang Ren. 2019. Cross-relation cross-bag attention for distantly-supervised relation extraction. In *Proceedings of the AAAI conference on artificial intelligence*, volume 33, pages 419–426.

Daojian Zeng, Kang Liu, Yubo Chen, and Jun Zhao. 2015. Distant supervision for relation extraction via piecewise convolutional neural networks. In *Proceedings of the 2015 conference on empirical methods in natural language processing*, pages 1753–1762.

Daojian Zeng, Kang Liu, Siwei Lai, Guangyou Zhou, and Jun Zhao. 2014. Relation classification via convolutional deep neural network. In *Proceedings of COLING 2014, the 25th International Conference on Computational Linguistics: Technical Papers*, pages 2335–2344, Dublin, Ireland. Dublin City University and Association for Computational Linguistics.

Xiangrong Zeng, Shizhu He, Kang Liu, and Jun Zhao. 2018. Large scaled relation extraction with reinforcement learning. In *Thirty-Second AAAI Conference on Artificial Intelligence*.

Shu Zhang, Dequan Zheng, Xinchen Hu, and Ming Yang. 2015. Bidirectional long short-term memory networks for relation classification. In *Proceedings of the 29th Pacific Asia conference on language, information and computation*, pages 73–78.

Zhilu Zhang and Mert Sabuncu. 2018. Generalized cross entropy loss for training deep neural networks with noisy labels. *Advances in neural information processing systems*, 31.

Wenxuan Zhou and Muhao Chen. 2021. Learning from noisy labels for entity-centric information extraction. *arXiv preprint arXiv:2104.08656*.

A Appendix

In this section, we proof our proposed Theorem 1. For ease of reference, we restate Theorem 1.

Theorem 1: Let g be a fixed representation function from $\mathcal{X}$ to $\mathcal{Z}$, $\mathcal{F}$ be the hypothesis class of Vapink Chervonenkis d. If a random sample of size m $\mathcal{Z}^{cs}$ is generated by applying g to a $\mathcal{D}^{cs}$ - i.i.d. for any $\sigma > 0$, with probability $1-\sigma$, we have the following uniform generalization error bound for any feature classification functions $f \in \mathcal{F}$,

$$\epsilon^{c}(f) \leq \epsilon^{\hat{cs}}(f) + \frac{1}{2} d_{F\triangle\mathcal{F}}\left(\mathcal{Z}^{cs}\right) + \lambda \tag{8}$$

where

$$\epsilon^{\hat{cs}}(f) = \frac{1}{m}\sum_{i=1}^{m}\left|\hat{f}(z_i^{cs}) - y_i^{cs}\right| \tag{9}$$

$$d_{\mathcal{F}\Delta\mathcal{F}}(\mathcal{Z}^{cs}) = 2 \sup_{f^{'},f^{''}\in\mathcal{F}}\left|\Pr[f^{'}(z^{cs}) \neq f^{''}(z^{cs})]\right| \tag{10}$$

$$f^{*} = arg\min_{f\in\mathcal{F}} \epsilon^{c}(f) + \epsilon^{cs}(f) \tag{11}$$

$$\lambda = \epsilon^{cs}(f^{*}) + \epsilon^{c}(f^{*}) + \sqrt{\frac{4}{m}\left(d\log\frac{2em}{d} + \log\frac{4}{\delta}\right)} \tag{12}$$

Eq. (8) indicates the generalization error bound of $\epsilon^{c}(f)$. It demonstrates that the expected error $\epsilon^{c}(f)$ of the clean test set can be bounded by using three terms ($\epsilon^{\hat{cs}}(f)$, $d_{\mathcal{F}\Delta\mathcal{F}}(\mathcal{Z}^{cs})$ and λ), which are defined in Eq. (9), Eq. (10) and Eq. (12), respectively. The corresponding explanations for these three terms are as follows.

1. $\epsilon^{\hat{cs}}(f)$. This term is the empirical error of $\mathcal{D}^{cs}$.

2. $d_{F\triangle\mathcal{F}}\left(\mathcal{Z}^{cs}\right)$. This term represents the upper bound (sup) of the probability (Pr) that two classification functions $f^{'}$ and $f^{''}$ divide the feature z^{cs} of the same sample into different classes.

3. λ. This term is the shared error of the ideal joint hypothesis (f^{*}) proposed in Eq. (11). This term is a constant.

We begin with the following lemmas to prove the Theorem 1.

Lemma A. *Definition 1: Given two feature distribution $\mathcal{Z}^{s}$ and $\mathcal{Z}^{c}$ extracted by a fixed g, and a hypothesis class $\mathcal{F}$, a set of classifiers. Through a given classifier f, the divergence$\mathcal{F}\Delta\mathcal{F}$ between $\mathcal{Z}^{s}$ and $\mathcal{Z}^{c}$ is:*

$$\begin{aligned} & d_{\mathcal{F}\triangle\mathcal{F}}\left(\mathcal{Z}^{s},\mathcal{Z}^{c}\right) \\ &= 2\sup_{\eta\in\mathcal{F}\Delta\mathcal{F}}\left|\Pr_{z\sim\mathcal{Z}^{c}}[f^{'}(z)\neq f^{''}(z)] - \Pr_{z\sim\mathcal{Z}^{s}}[f^{'}(z)\neq f^{''}(z)]\right| \\ &= 2\sup_{\eta\in\mathcal{F}\Delta\mathcal{F}}\left|\Pr_{z\sim\mathcal{Z}^{s}}[z:\eta(z)=1] - \Pr_{z\sim\mathcal{Z}^{c}}[z:\eta(z)=1]\right| \end{aligned}$$

$$\begin{aligned} & \mathcal{F}\Delta\mathcal{F} = \{\eta : \eta\left(z^{*}\right) = 1\}, \oplus: \text{ XOR operator} \\ & z^{*} = \{z : f_1(z) \oplus f_2(z), f_1, f_2 \in \mathcal{F}\} \end{aligned}$$

where $\mathcal{Z}^{s} \subseteq \mathcal{Z}^{cs}$ and $\mathcal{Z}^{c} \subseteq \mathcal{Z}^{cs}$ are feature distribution of noisy data and clean data, respectively. Lemma A has been proposed and proved in (Ben-David et al., 2010).

Lemma B. *The upper bound of the probability $d_{F\triangle F}(\mathcal{Z}^{cs})$ that two classification functions $f^{'}$ and $f^{''}$ divide the feature z^{cs} of the same sample into different classes is equal to the upper bound of $d_{\mathcal{F}\triangle\mathcal{F}}(\mathcal{Z}^{s},\mathcal{Z}^{c})$.*

Proof. *Now we proof the Lemma B.*

$$\begin{aligned} & d_{\mathcal{F}\triangle\mathcal{F}}\left(\mathcal{Z}^{s},\mathcal{Z}^{c}\right) \\ &= 2\sup_{\eta\in\mathcal{F}\Delta\mathcal{F}}\left|\Pr_{z\sim\mathcal{Z}^{s}}[z:\eta(z)=1] - \Pr_{z\sim\mathcal{Z}^{c}}[z:\eta(z)=1]\right| \\ &\leq 2\sup_{\eta\in\mathcal{F}\Delta\mathcal{F}}\left|\Pr_{z\sim\mathcal{Z}^{s}}[z:\eta(z)=1] + \Pr_{z\sim\mathcal{Z}^{c}}[z:\eta(z)=1]\right| \\ &= 2\sup_{\eta\in\mathcal{F}\Delta\mathcal{F}}\left|\Pr_{z\sim\mathcal{Z}^{cs}}[z:\eta(z)=1]\right| \qquad (13) \\ &= 2\sup_{f^{'},f^{''}\in\mathcal{F}}\left|\Pr_{z\sim\mathcal{Z}^{cs}}[f^{'}(z)\neq f^{''}(z)]\right| \qquad (14) \end{aligned}$$

The Eq. (14) is equal to Eq. (10), which is the $d_{\mathcal{F}\triangle\mathcal{F}}(\mathcal{Z}^{cs})$, thus the $d_{F\triangle F}(\mathcal{Z}^{cs})$ is the upper bound of $d_{\mathcal{F}\triangle\mathcal{F}}(\mathcal{Z}^{s}, \mathcal{Z}^{c})$. Lemma B has been proved.

Proof. *Now we proof the Theorem 1.*

For a classifier f, let $\mathcal{Z}_f \subseteq \mathcal{Z}$ be the feature subset for whose characteristic function is f. The parallel notation $\mathcal{Z}_{f^*}$ and $\mathcal{Z}_f$ are used for classifier f^* and f. Through the feature subset, we make $\Pr_c[\mathcal{Z}_f \triangle \mathcal{Z}_{f^*}] = \Pr_{z\sim Z^c}[f(z) \neq f^*(z)]$, and the parallel notation $\Pr_{cs}$ is used.

$$\begin{aligned}
\epsilon^c(f) &\leq \epsilon^c(f^*) + \Pr_c[\mathcal{Z}_f \Delta \mathcal{Z}_{f^*}] && (15)\\
&\leq \epsilon^c(f^*) + \Pr_{cs}[\mathcal{Z}_f \triangle \mathcal{Z}_{f^*}] \\
&\quad + |\Pr_c[\mathcal{Z}_f \Delta \mathcal{Z}_{f^*}] - \Pr_{cs}[\mathcal{Z}_f \triangle \mathcal{Z}_{f^*}]| \\
&\leq \epsilon^c(f^*) + \epsilon^{cs}(f^*) + \epsilon^{cs}(f) \\
&\quad + |\Pr_c[\mathcal{Z}_f \Delta \mathcal{Z}_{f^*}] - \Pr_{cs}[\mathcal{Z}_f \Delta \mathcal{Z}_{f^*}]| && (16)\\
&= \epsilon^c(f^*) + \epsilon^{cs}(f^*) + \epsilon^{cs}(f) \\
&\quad + |\Pr_s[\mathcal{Z}_f \Delta \mathcal{Z}_{f^*}]| \\
&\leq \epsilon^c(f^*) + \epsilon^{cs}(f^*) + \epsilon^{cs}(f) \\
&\quad + \sup_{\hat{f}\in\mathcal{F}} |\Pr_s[\mathcal{Z}_f \Delta \mathcal{Z}_{\hat{f}}] + \Pr_c[\mathcal{Z}_f \Delta \mathcal{Z}_{\hat{f}}]| \\
&\leq \epsilon^c(f^*) + \epsilon^{cs}(f) + \frac{1}{2} d_{\mathcal{F}\Delta\mathcal{H}}(\mathcal{Z}^{cs})
\end{aligned}$$

Eq. (15) and Eq. (16) rely on the triangle inequality for classification error (Schölkopf et al.). Besides, according to the standard Vapnik-Chervonenkis theorem (Vapnik, 1999), the $\epsilon^{cs}(f)$ can be bounded by its empirical estimate:

$$\epsilon^{cs}(f) \leq \sqrt{\frac{4}{m}\left(d\log\frac{2em}{d} + \log\frac{4}{\delta}\right)} + \hat{\epsilon^{cs}}(f) \quad (17)$$

in summary:

$$\epsilon^c(f) \leq \hat{\epsilon^{cs}}(f) + \frac{1}{2} d_{F\triangle\mathcal{F}}(\mathcal{Z}^{cs}) + \lambda \quad (18)$$

Theorem 1 has been proved.

B Appendix

In this section, we conduct a hyperparameter tuning study on the number of classifiers for CETA. CETA is designed according to the denoising theorem, which states that the decomposition of the classifier helps to reduce the generalization error. To explore whether more classifiers can further improve the performance, we conduct the experiment of CETA implemented with two, three, and four classifiers on NYT based on BiLSTM, and the results are shown in Table 4.

Model	**NYT**			
	#Classifier	**Prec.**	**Rec.**	**F1**
BiLSTM	1	35.52	67.41	46.53
CETA+BiLSTM	2	71.34	61.12	65.83
CETA+BiLSTM	3	71.46	61.37	66.03
CETA+BiLSTM	4	71.49	61.39	66.05

Table 4: Main results of the sentence-level evaluation on NYT. #Classifier indicates the number of classifiers used to implement CETA.

The results demonstrate that CETA implemented by four classifiers is 0.34% higher than that of the two classifiers in F1 metrics. Besides, CETA implemented by three classifiers is 0.3% higher than that of the two classifiers in F1 metrics, indicating more classifiers can further improve the performance of CETA.

MEDDISTANT19: Towards an Accurate Benchmark for Broad-Coverage Biomedical Relation Extraction

Saadullah Amin[♣,△] * **Pasquale Minervini**[♠] * **David Chang**[♢]
Pontus Stenetorp[♠] **Günter Neumann**[♣,△]
[♣]German Research Center for Artificial Intelligence [♠]UCL Centre for Artificial Intelligence
[△]Saarland Informatics Campus, Saarland University [♢]Yale Center for Medical Informatics
{saadullah.amin,guenter.neumann}@dfki.de {p.minervini,p.stenetorp}@cs.ucl.ac.uk
david.chang@yale.edu

Abstract

Relation extraction in the biomedical domain is challenging due to the lack of labeled data and high annotation costs, needing domain experts. Distant supervision is commonly used to tackle the scarcity of annotated data by automatically pairing knowledge graph relationships with raw texts. Such a pipeline is prone to noise and has added challenges to scale for covering a large number of biomedical concepts. We investigated existing broad-coverage distantly supervised biomedical relation extraction benchmarks and found a significant overlap between training and test relationships ranging from 26% to 86%. Furthermore, we noticed several inconsistencies in the data construction process of these benchmarks, and where there is no train-test leakage, the focus is on interactions between narrower entity types. This work presents a more accurate benchmark MEDDISTANT19 for broad-coverage distantly supervised biomedical relation extraction that addresses these shortcomings and is obtained by aligning the MEDLINE abstracts with the widely used SNOMED Clinical Terms knowledge base. Lacking thorough evaluation with domain-specific language models, we also conduct experiments validating general domain relation extraction findings to biomedical relation extraction.

1 Introduction

Extracting structured knowledge from unstructured text is important for knowledge discovery and management. Biomedical literature and clinical narratives offer rich interactions between entities mentioned in the text (Craven and Kumlien, 1999; Xu and Wang, 2014), which can be helpful for applications such as bio-molecular information extraction, pharmacogenomics, and identifying drug-drug interactions (DDIs), among others (Luo et al., 2017).

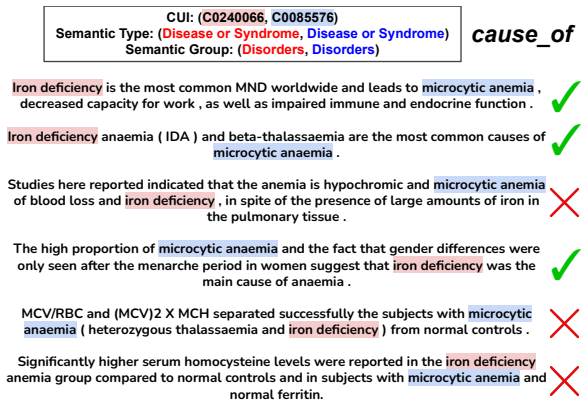

Figure 1: An example of a bag instance representing the UMLS concept pair (C0240066, C0085576) from the MEDDISTANT19 dataset, expressing the relation *cause_of*. In this example, three out of six sentences express the relation, while others are incorrect labels resulting from the distant supervision.

Manually annotating these relations for training supervised learning systems is an expensive and time-consuming process (Segura-Bedmar et al., 2011; Kilicoglu et al., 2011; Segura-Bedmar et al., 2013; Li et al., 2016), so the task often involves leveraging rule-based (Abacha and Zweigenbaum, 2011; Kilicoglu et al., 2020) and weakly supervised approaches (Peng et al., 2016; Dai et al., 2019).

To scale to a large number of biomedical entities, recent works have focused on broad-coverage relation extraction (Amin et al., 2020a; Xing et al., 2020; Hogan et al., 2021), where we investigated these benchmarks for possible train-test leakage of knowledge graph triples and found significant portions overlapping (Table 2). Such leakage impacts the model performance as it allows to score higher by simply memorizing the training relations rather than generalizing to new, previously unknown ones. We identify the sources of these issues as normalizing the textual form of concept mentions to their unique identifiers and improper handling of inverse relations. In contrast, more ac-

* *Equal contribution.*

Proceedings of the 29th International Conference on Computational Linguistics, pages 2259–2277
October 12–17, 2022.

Benchmark	Relations	No Train-Test Overlap	Broad-Coverage	Ontology
UMLS.v1 (Roller and Stevenson, 2014)	7	-	✗	UMLS
DTI (Hong et al., 2020)	6	✓	✗	DrugBank
UMLS.v2 (Amin et al., 2020a)	355	✗	✓	UMLS
BioRel (Xing et al., 2020)	125	✗	✓	NDFRT, NCI
UMLS.v3 (Hogan et al., 2021)	275	✗	✓	UMLS
TBGA (Marchesin and Silvello, 2022)	4	✓	✗	DisGeNET
MedDistant19	22	✓	✓	SNOMED CT

Table 1: The landscape of distantly supervised biomedical relation extraction (Bio-DSRE) benchmarks: all the existing broad-coverage datasets have corpus-level triples overlap between the train and test splits (Table 2), where the knowledge graph (KG) is also extracted from multiple ontologies. The DTI and TBGA benchmarks focus on harmonized ontology but are limited to drug-target interactions and gene-disease associations. In contrast, MEDDISTANT19 has a broader coverage of entities and their semantic types and is normalized to a single ontology, SNOMED CT, which has significant clinical relevance. We named the datasets from (Roller and Stevenson, 2014; Amin et al., 2020a; Hogan et al., 2021) to UMLS.v1/2/3 since the original works had no names. For UMLS.v1, there is no publicly available code to reconstruct the dataset; thus, the overlap information is missing.

curate benchmarks exist (Hong et al., 2020; Marchesin and Silvello, 2022) but focus on narrower types of interactions. To alleviate the broad-coverage benchmark issues and bridge this gap, we present a new benchmark MEDDISTANT19 which draws its knowledge graph from the widely used healthcare ontology SNOMED CT (Chang et al., 2020). Further, with the success of domain-specific pre-trained language models for biomedical and clinical tasks (Gu et al., 2021), and inspired by existing thorough relation extraction studies in the general domain (Peng et al., 2020; Alt et al., 2020; Gao et al., 2021), we conduct an extensive evaluation using MEDDISTANT19 for the biomedical domain.

2 Related Work

Relation Extraction (RE) is an important task in biomedical applications. Traditionally, supervised methods require large-scale annotated corpora, which is impractical to scale for broad-coverage biomedical relation extraction (Kilicoglu et al., 2011, 2020). Distant Supervision (DS) allows for the automated collection of noisy training examples by aligning a given knowledge base (KB) with a collection of text sources (Mintz et al., 2009). DS was used in recent works (Alt et al., 2019; Amin et al., 2020a) with pre-trained language models using Multi-Instance Learning (MIL) by creating *bags* of instances (Riedel et al., 2010) for corpus-level triple extraction.[1] In biomedical domain, Roller and Stevenson (2014) first proposed the use of the Unified Medical Language System (UMLS) Metathesaurus (Bodenreider, 2004) as a KB with PubMed (Canese and Weis, 2013) MEDLINE abstracts as text collection.

For broad-coverage tasks, Dai et al. (2019) implemented a knowledge-based attention mechanism (Han et al., 2018) for mutual learning with knowledge graph completion and entity type classification. Xing et al. (2020) introduced a large-scale BioRel benchmark focusing on drug-disease and gene-cancer interactions and showed significant performance using a comprehensive selection of baselines. Recent works focused on using domain-specific pre-trained language models for distantly supervised biomedical relation extraction (Bio-DSRE). Amin et al. (2020a) extended relation enriched sentence-level BERT (Wu and He, 2019) to handle bag-level MIL and demonstrated that preserving the direction of the KB relationships can denoise the training signal. They also outlined the steps to create a broad-coverage benchmark from UMLS. Following this, Hogan et al. (2021) introduced the concept of *abstractified* MIL (AMIL), by including different argument pairs belonging to the same semantic type pair in one bag, boosting performance on rare triples.

For domain-specific Bio-DSRE, Hong et al. (2020) introduced the BERE framework for latent tree learning and self-attention to use the semantic and syntactic information in the sentence for MIL. They also introduced a drug-target interactions (DTI) Bio-DSRE benchmark, suitable for drug repositioning, drawn from DrugBank

[1] RE is used to refer to two different tasks: sentence-level detection of relational instances and corpus-level triples extraction, a kind of knowledge graph completion or link prediction task (Amin et al., 2020b).

Triples	Train	Valid	Test
UMLS.v2	211,789	41,993 (26.7%)	89,486 (26.5%)
BioRel	39,969	17,815 (86.17%)	17,927 (86.37%)
UMLS.v3	23,163	2,643 (44.38%)	5,184 (40.12%)

Table 2: Training-test leakage we identified in the existing broad-coverage benchmarks. Numbers between parentheses show the percentage overlap of CUI triples.

(Wishart et al., 2018). Concurrent work of Marchesin and Silvello (2022) introduced a large-scale semi-automatically curated benchmark TGBA for gene-disease associations (GDA). TGBA uses DisGeNET (Piñero et al., 2020), which collects data on human genotype-phenotype relationships.

This work investigates recent results from the broad-coverage Bio-DSRE literature by probing the respective datasets for overlaps between training and test sets. Specifically, in UMLS, each concept is mapped to a *Concept Unique Identifier (CUI)*, and a given CUI might have different surface forms (Bodenreider, 2004), we thus probe for CUI-based KG triples leakage. Our results are shown in Table 2 for UMLS.v2 (Amin et al., 2020a), BioRel (Xing et al., 2020), and UMLS.v3 (Hogan et al., 2021). For UMLS.v2 and UMLS.v3, the triples use surface forms of CUIs rather than the CUIs themselves, which results in an overlap between training and test sets. For example, consider a relationship between a pair of UMLS entities (`C0013798`, `C0429028`). These two entities can appear in different forms within a text, such as (*electrocardiography*, *Q-T interval*), (*ECG*, *Q-T interval*), and (*EKG*, *Q-T interval*); each of these distinct pairs still refers to the same original pair (`C0013798`, `C0429028`). Amin et al. (2020a) claim no such text-based leakage, but when canonicalized to their CUIs, this results in leakage across the splits as reported in Table 2. In contrast, BioRel directly splits CUI triples without accounting for inverse relations that can also result in leakage (Chang et al., 2020). Since DSRE aims at corpus-level triples extraction, train-test triples leakage is problematic (see Table 3) compared to supervised sentence-level RE, where we aim to generalize to newer contexts.

We found no such overlap for DTI and TBGA, where the datasets used in (Roller and Stevenson, 2014; Dai et al., 2019) are not publicly available. Noting these shortcomings, we introduce a new and accurate benchmark MEDDISTANT19 for broad-coverage Bio-DSRE. Our benchmark utilizes clinically relevant SNOMED CT Knowledge Graph (Chang et al., 2020), extracted from the UMLS, that offers a careful selection of the concept types and is suitable for large-scale biomedical relation extraction. Table 1 summarizes the current landscape of Bio-DSRE benchmarks.

Model and Data	Original		Filtered	
	AUC	F1	AUC	F1
Amin et al. (2020a)	68.4	64.9	50.8	53.1
Hogan et al. (2021)†	82.6	77.6	11.8	19.8

Table 3: State-of-the-art Bio-DSRE language models were evaluated on the respective datasets before (Original) and after (Filtered) removing overlapping relationships. † Our re-run of the AMIL (Type L) model; original scores are 87.2 (AUC) and 81.2 (F1).

In supervised RE, ChemProt (Krallinger et al., 2017) and DDI-2013 (Herrero-Zazo et al., 2013) focus on multi-class interactions between chemical-protein and drug-drug respectively. EU-ADR (van Mulligen et al., 2012) and GAD (Bravo et al., 2015) focus on binary relations between genes and diseases, while CDR (Li et al., 2016) focuses on binary relations between chemicals and diseases.

3 Constructing the MedDistant19 Benchmark

Documents We used PubMed MEDLINE abstracts published up to 2019[2] as our text source, containing 32,151,899 abstracts. Following Hogan et al. (2021), we used SCISPACY [3] (Neumann et al., 2019) for sentence tokenization, resulting in 150,173,169 unique sentences. We further introduce the use of SCISPACY for linking entity mentions to their UMLS CUIs and filtering disabled concepts from UMLS, which resulted in entity-linked mentions at the sentence-level.

Named entity recognition (NER) and normalization were two primary sources of errors in biomedical RE, as shown in Kilicoglu et al. (2020). While SCISPACY is reasonably performant among other options for biomedical entity linking, it remains quite noisy in practice; e.g., Vashishth et al. (2021) showed that SCISPACY had only about a 50% accuracy on extracting concepts in benchmark datasets. Despite this being a limitation, using SCISPACY is better than relying on string matching alone (Dai et al., 2019; Amin et al., 2020a; Hogan et al., 2021).

[2] https://lhncbc.nlm.nih.gov/ii/information/MBR/Baselines/2019.html

[3] https://github.com/allenai/scispacy

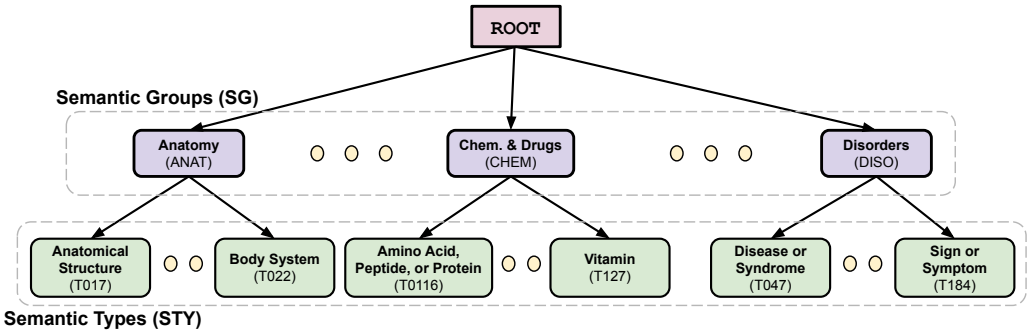

Figure 2: Type hierarchy in UMLS, where each concept is classified under a taxonomy. The *coarse-grained* and *fine-grained* entity types are referred to as Semantic Group (SG) and Semantic Type (STY) respectively.

Knowledge Base We use UMLS2019AB [4] as our primary knowledge source and apply a set of rules, resulting in a distilled and carefully reduced version of UMLS2019AB. The UMLS Metathesaurus (Bodenreider, 2004) covers concepts from 222 source vocabularies, thus being the most extensive ontology of biomedical concepts. However, covering all ontologies can be challenging, given the interchangeable nature of the concepts. For example, *programmed cell death 1 ligand 1* is an alias of concept `C1540292` in the HUGO Gene Nomenclature Committee ontology (Povey et al., 2001), and it is an alias of concept `C3272500` in the National Cancer Institute Thesaurus. This makes entity linking more challenging since a surface form can be linked to multiple entity identifiers and easier to have overlaps between training and test sets since the same fact may appear in both with different entity identifiers.

Furthermore, benchmark corpora for biomedical NER (Doğan et al., 2014; Li et al., 2016) and RE (Herrero-Zazo et al., 2013; Krallinger et al., 2017) focuses on specific entity types (e.g. diseases, chemicals, proteins), and are usually normalized to a single ontology (Kilicoglu et al., 2020). Following this trend, we also focus on a single vocabulary for Bio-DSRE. We use SNOMED CT, the most widely used clinical terminology worldwide for documentation and reporting in healthcare (Chang et al., 2020).

Since UMLS classifies each entity in a type taxonomy of semantic types (STY) and semantic groups (SG) (Fig. 2), this allows for narrowing the concepts of interest. Following Chang et al. (2020), we first consider 8 semantic groups in SNOMED CT: Anatomy (ANAT), Chemicals & Drugs (CHEM), Concepts & Ideas (CONC), Devices (DEVI), Disorders (DISO), Phenomena (PHEN), Physiology (PHYS), and Procedures (PROC). We then remove CONC and PHEN as they are far too general to be informative for Bio-DSRE. For a complete list of semantic types covered in MEDDISTANT19, see Table A.4. Similarly, each relation is categorized into a type and has a reciprocal relation in UMLS (Table A.3), which can result in train-test leakage (Dettmers et al., 2018).

These steps follow Chang et al. (2020), with the difference that we only consider relations of type *has relationship other than synonymous, narrower, or broader* (RO); this is consistent with prior works in Bio-DSRE. We also exclude uninformative relations, *same_as*, *possibly_equivalent_to*, *associated_with*, *temporally_related_to*, and ignore inverse relations as generally is the case in RE.

In addition, Chang et al. (2020) ensures that the validation and test set do not contain any new entities, making it a transductive learning setting where we assume all test entities are known beforehand. However, we are expected to extract relations between unseen entities in real-world applications of biomedical RE. To support this setup, we derive MEDDISTANT19 using an inductive KG split method proposed by Daza et al. (2021) (see Appendix A in their paper). Table 5 summarizes the statistics of the KGs used for alignment with the text. We use split ratios of 70%, 10%, and 20%. Relationships are defined between CUIs and have no overlap between training, validation, and test.

3.1 Knowledge-to-Text Alignment

We now describe the procedure for searching fact triples to match relational instances in text.

Let $\mathcal{E}$ and $\mathcal{R}$ respectively denote the set of UMLS CUIs and relation types, and let $\mathcal{G} \subseteq$

[4] https://download.nlm.nih.gov/umls/kss/2019AB/umls-2019AB-full.zip

Properties	Prior	MD19
approximate entity linking		✓
unique NA sentences		✓
inductive		✓
triples leakage	✓	
NA-type constraint		✓
NA-argument role constraint		✓

Table 4: MEDDISTANT19 (MD19) key data construction properties compared with the recent broad-coverage Bio-DSRE works.

Facts	Training	Validation	Testing
Inductive	261,797	48,641	97,861
Transductive	318,524	28,370	56,812

Table 5: The number of raw inductive and transductive SNOMED KG triples used for alignment with text.

Summary	Entities	Relations	STY	SG
	20,256	22	51	6
Split	**Instances**	**Facts**	**Bags**	**Inst. per Bag**	**NA (%)**
Train	450,071	5,455	88,861	5.06	90.0%
Valid	39,434	842	10,475	3.76	91.2%
Test	91,568	1,663	22,606	4.05	91.1%

Table 6: Summary statistics of the MEDDISTANT19 dataset using Inductive SNOMED KG split (Table 5). The number of relations includes the unknown relation type (NA).

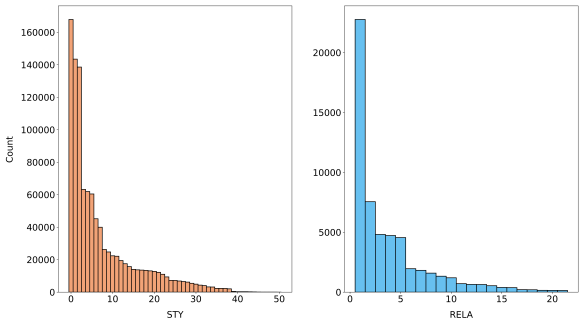

Figure 3: (*Left*) Entity distribution based on Semantic Types. (*Right*) Relations distribution.

$\mathcal{E} \times \mathcal{R} \times \mathcal{E}$ denote the set of relationships contained in UMLS. For producing a training-test split, we first create a set $\mathcal{G}^+ \subseteq \mathcal{E} \times \mathcal{E}$ of related entity pairs as:

$$\mathcal{G}^+ = \{(e_i, e_j) \mid \langle e_i, p, e_j \rangle \in \mathcal{G} \vee \langle e_j, p, e_i \rangle \in \mathcal{G}\}$$

Following this, we obtain a set of unrelated entity pairs by corrupting one of the entities in each pair in $\mathcal{G}^+$ and making sure it does not appear in $\mathcal{G}^+$, obtaining a new set $\mathcal{G}^- \subseteq \mathcal{E} \times \mathcal{E}$ of unrelated entities, defined as follows:

$$\mathcal{G}^- = \{(\overline{e_i}, e_j) \mid (e_i, e_j) \in \mathcal{G}^+ \wedge (\overline{e_i}, e_j) \notin \mathcal{G}^+\} \cup \{(e_i, \overline{e_j}) \mid (e_i, e_j) \in \mathcal{G}^+ \wedge (e_i, \overline{e_j}) \notin \mathcal{G}^+\}$$

During the corruption process, we enforce two constraints: 1) *type constraint* – the two entities appearing in each negative pair in $\mathcal{G}^-$ should belong to an entity type pair from $\mathcal{G}^+$, and 2) *role constraint* – the noisy *head* (*tail*) entity in negative pair must have appeared in *head* (*tail*) role from a pair in $\mathcal{G}^+$.

A naive choice for the negative group could be $\mathcal{G}^- = (\mathcal{E} \times \mathcal{E}) - \mathcal{G}^+$, for which the current approach is only a subset; however, enumerating all possible entity pairs can be infeasible if $|\mathcal{E}|$ is high. Furthermore, we do not assume the completeness of UMLS, and only derive a *fixed* sub-graph from the 2019 version subject to the constraints. This process is similar to Local-Closed World Assumption (LCWA, Dong et al., 2014; Nickel et al., 2016), in which a KG is assumed to be only locally complete: if we observed a triple for a specific entity $e_i \in \mathcal{E}$, then we assume that any non-existing relationship (e_i, e_j) denotes a false fact and include them in $\mathcal{G}^-$. Therefore, it is likely that if a triple emerges in a new PubMed article such that it violates the negative sampling assumptions, it will be considered a false negative. However, this amount is negligible due to intractable search space that scales with the size of the KG.

For each entity-linked sentence, we only consider those sentences that have SNOMED CT entities and have pairs in $\mathcal{G}^+$ and $\mathcal{G}^-$. Selected positive and negative pairs are mutually exclusive and have no overlap across splits. Since we only consider unique sentences associated with a pair, this makes for unique negative training instances, in contrast to Amin et al. (2020a), who considered generating positive and negative pairs from the same sentence. We define negative examples as relational sentences mentioning argument pairs with *unknown relation type* (NA), i.e. there might be a relationship, but the considered set of relations does not cover it. Our design choices are summarized in Table 4.

We also remove mention-level overlap across the splits and apply type-based mention pruning. Specifically, we pool mentions by type and remove the sentences which have the mention appearing more than 10,000 times. We selected the threshold based on manual inspection of frequent mentions

Model	Bag	Strategy	AUC	F1-micro	F1-macro	P@100	P@200	P@300	P@1k	P@2k
CNN	-	AVG	27.3	33.0	16.1	50.0	46.0	44.0	41.0	33.6
	-	ONE	30.4	36.7	18.2	67.0	58.5	52.6	43.5	34.4
	✓	AVG	30.4	36.2	19.8	70.0	58.0	56.0	46.0	35.5
	✓	ONE	34.6	40.4	17.8	77.0	72.5	67.6	50.0	37.3
	✓	ATT	35.0	40.1	19.8	78.0	73.5	68.6	51.4	36.4
PCNN	-	AVG	27.2	32.4	12.9	54.0	49.5	50.3	40.7	33.2
	-	ONE	29.8	36.7	16.2	66.0	55.5	52.3	44.4	34.2
	✓	AVG	29.6	37.3	20.5	59.0	50.5	50.0	47.0	35.9
	✓	ONE	28.6	36.5	18.1	66.0	65.0	62.0	44.7	33.7
	✓	ATT	32.5	38.2	14.4	71.0	71.0	67.3	49.0	35.2
GRU	-	AVG	42.7	47.4	27.8	78.0	74.0	76.0	59.2	42.7
	-	ONE	46.4	49.3	29.2	86.0	80.5	78.3	61.2	44.9
	✓	AVG	28.6	37.2	17.9	57.0	57.0	56.0	45.3	35.4
	✓	ONE	32.6	40.8	17.7	73.0	70.5	66.3	51.2	37.0
	✓	ATT	36.6	40.9	22.2	77.0	72.0	67.6	51.3	38.7
BERT	-	AVG	**79.8**	**76.1**	**65.3**	95.0	96.0	96.0	**90.2**	67.2
	-	ONE	79.3	**76.1**	64.7	93.0	94.0	94.0	89.2	**67.4**
	✓	AVG	78.3	73.1	51.1	**99.0**	**97.5**	**96.6**	87.8	66.0
	✓	ONE	67.0	55.7	44.4	89.0	90.5	91.0	78.7	57.8
	✓	ATT	64.6	56.4	42.7	89.0	87.5	85.6	75.4	57.9

Table 7: Baseline results for MEDDISTANT19.

in each semantic type, so the information loss is minimal. At the same time, we still removed generalized mentions such as *disease*, *drugs*, *temperature* etc. We provide a complete list of mentions removed by this step in Table A.2. Table 6 shows the final summary of MEDDISTANT19 using inductive split covering 20,256 entities with 51 types and 343 type pairs. Fig. 3 shows entity and relation plots, following a long-tail distribution.

4 Experiments

MEDDISTANT19 is released in a format that is compatible with the widely adopted RE framework OpenNRE (Han et al., 2019).[5] To report our results, we use the *corpus-level* Area Under the Precision-Recall (PR) curve (AUC), Micro-F1, Macro-F1, and Precision-at-k (P@k) with $k \in \{100, 200, 300, 1k, 2k\}$, and the *sentence-level* Precision, Recall, and F1. Due to the imbalanced nature of relational instances, following Gao et al. (2021), we report Macro-F1 values, and following Hogan et al. (2021), we report sentence-level RE results on relationships, including frequent and rare triples.

4.1 Baselines

Our baseline experiments largely follow the setup of Gao et al. (2021) with the addition of GRU models.[6] For sentence encoding, we use CNN (Liu et al., 2013), PCNN (Zeng et al., 2015), bidirectional GRU (Hong et al., 2020), and BERT (Devlin et al., 2019). We use GloVe (Pennington et al., 2014) and Word2Vec (Mikolov et al., 2013) for CNN/PCNN/GRU models and initialize BERT with BioBERT (Lee et al., 2020).

We trained our models both at *sentence-level* and at *bag-level*. In contrast, prior works only considered bag-level training for Bio-DSRE. The sentence-level setup is similar to standard RE (Wu and He, 2019), with the difference that the evaluation is conducted at the bag-level. We also consider different pooling strategies, namely average (AVG), which averages the representations of sentences in a bag, at least one (ONE, Zeng et al., 2015), which generates relation scores for each sentence in a bag, and then selects the top-scoring sentence, and attention (ATT), which learns an attention mechanism over the sentences within a bag.

Table 7 presents our main results. In all the cases, the BERT sentence encoder performed better than

[5] https://github.com/suamin/MedDistant19

[6] https://github.com/pminervini/meddistant-baselines

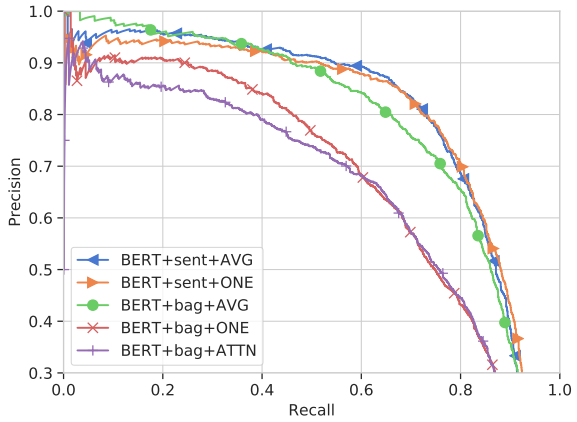

Figure 4: Precision-Recall curves for BERT baselines.

Model	1-1	1-M	M-1
BERT+bag+AVG	**66.6**	**48.3**	**66.6**
BERT+bag+ONE	52.6	33.2	47.1
BERT+bag+ATT	56.4	30.7	26.4

Table 8: Averaged F1-micro score on relation-specific category for *bag* pooling methods. The categories are defined using the *cardinality* of head and tail SGs.

Model	P	R	F1
All Triples			
BERT+sent+AVG	**0.79**	**0.65**	**0.71**
BERT+bag+AVG	0.72	0.64	0.68
Common Triples			
BERT+sent+AVG	**0.98**	**0.62**	**0.76**
BERT+bag+AVG	0.96	0.60	0.74
Rare Triples			
BERT+sent+AVG	**0.97**	0.70	0.82
BERT+bag+AVG	0.95	**0.73**	**0.83**

Table 9: Sentence-level RE comparing BERT baselines trained at bag and sentence-level with AVG pooling on Rare and Common subsets of MEDDISTANT19. The triples include NA relational instances.

others since pre-trained language models are effective for entity-centric transfer learning (Amin and Neumann, 2021), domain-specific fine-tuning (Amin et al., 2019), and can implicitly store relational knowledge during pre-training (Petroni et al., 2019). This trend is similar to the general domain, and the BERT-based experiments provide consistent baselines lacking in the prior works. Similar to the general domain (Gao et al., 2021), we find sentence-level training to perform better than the bag-level. However, BERT+bag+AVG had much better precision for the top-scoring triples at the expense of long-tail performance. At the sentence-level, those instances that have been correctly labeled by distant supervision (e.g. Fig. 1) provide enough learning signal, given the generalization abilities of LMs. However, the model is supposed to jointly learn from clean and noisy samples in bag-level training, thus limiting its overall performance. But, we do not find this trend for CNN/PCNN. Instead, the bag-level models performed slightly better except for GRU. We further plot Precision-Recall (PR) curves for BERT-based baselines in Fig. 4.

Pooling Strategies In all cases, AVG proved to be a better pooling strategy; this finding is consistent with prior works. Both Amin et al. (2020a) and Gao et al. (2021) found ATT to produce less accurate results with LMs, which we also find to hold true for MEDDISTANT19. To further study the impact of bag-level pooling strategies, we analyze the relation category-specific results. Following Chang et al. (2020), we grouped the relations based on cardinality, where the cardinality is defined as for a given relation type if the set of *head* or *tail* entities belongs to only one semantic group, then it has a cardinality one otherwise, M (many). The results are shown in Table 8 for bag-level BERT-based models with three pooling schemes. On average, models struggled the most with the 1-M category due to a lack of enough training signal to differentiate between heterogeneous entity types pooled over instances in a bag. While we would expect symmetric performance, to some extent, in 1-M and M-1 categories, the difference highlights that the KB-direction plays a role in Bio-DSRE, which previously has been used to de-noise the training signal (Amin et al., 2020a).

Long-Tail Performance Following Hogan et al. (2021), we also perform sentence-level triples evaluation of BERT-based encoders trained at sentence-level and bag-level. The authors divided the triples (including NA instances) into two categories: those with 8 or more sentences are defined as *common triples* and others as *rare triples*. Table 9 shows these results. We note that both training strategies performed comparably on rare triples with BERT+sent+AVG more precise than BERT+bag+AVG at the expense of low recall. However, we find a noticeable difference in com-

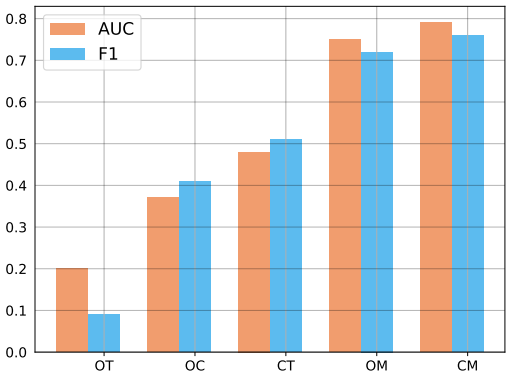

Figure 5: Ablation showing the effect of different text encoding methods with MEDDISTANT19.

mon triples where BERT+sent+AVG performed better. At the bag level, the model can overfit to certain type and mention heuristics, whereas sentence-level training allows more focus on context. The current state-of-the-art model from Hogan et al. (2021) creates a bag of instances by abstracting entity pairs belonging to the same semantic type pair into a single bag, thus producing heterogeneous bags. Due to such bag creation, it is not suited for sentence-level models.

4.2 Analysis

Context, Mention, or Type? RE models are known to heavily rely on information from entity mentions, most of which is type information, and existing datasets may leak shallow heuristics via entity mentions that can inflate the prediction results (Peng et al., 2020). To study the importance of mentions, contexts, and entity types in MEDDISTANT19, we take inspiration from (Peng et al., 2020; Han et al., 2020) and conduct an ablation of different text encoding methods. We consider entity mentions with special entity markers (Amin et al., 2020a) as the *Context + Mention* (CM) setting, which is common in RE with LMs. We then remove the context and only use mentions, the *Only Mention* (OM) setting, which reduces to KG-BERT (Yao et al., 2019) for relation prediction. We then only consider the context by replacing subject and object entities with special tokens, resulting in the *Only Context* (OC) setting. Lastly, we consider two type-based (STY) variations as *Only Type* (OT) and *Context + Type* (CT). We train the models at the sentence-level and evaluate them at the bag-level.

We observe in Fig. 5 that the CM method had the highest performance, but surprisingly, OM performed quite well. This highlights the ability of

Split	AUC	F1-micro	F1-macro
Inductive	**79.9**	**76.2**	65.4
Transductive	79.6	73.3	**65.9**

Table 10: BERT+sent+AVG performance on corpora created with an inductive and transductive set of triples.

LMs to memorize the facts and act as soft KBs (Petroni et al., 2019). This trend is also consistent with general-domain (Peng et al., 2020). The poor performance in the OC setting shows that the model struggles to understand the context, more pronounced in noise-prone distant RE than in supervised RE. Our CT setup can be seen as a sentence-level extrapolation of the AMIL model (Hogan et al., 2021), which struggles to perform better than the baseline (OM). However, comparing OC with CT, it is clear that the model benefits from type information as it can help constrain the space of the relations. Using only the type information had the least performance as the model fails to disambiguate between different entities belonging to the same type.

Inductive or Transductive? To study the impact of *transductive* and *inductive* splits (Table 5), we created another Bio-DSRE corpus using transductive train, validation, and test triples. The corpus generated differs from the inductive one, but it can offer insights into the model's ability to handle seen (*transductive*) and unseen (*inductive*) mentions. As shown in Table 10, the performance using inductive is slightly better than transductive for corpus-level extractions in terms of AUC. However, the F1-macro score is better for transductive. We conclude that the model can learn patterns that exploit mentions and type information to extrapolate to unseen mentions in the inductive setup.

Does Expert Knowledge Help? We now consider several pre-trained LMs with different knowledge capacities, specific to biomedical and clinical language understanding, to gain insights about the state-of-the-art encoders' performance and effectiveness on the MEDDISTANT19 benchmark.

We use BERT (Devlin et al., 2019) as baseline. We next consider only those pre-trained models trained with masked language modeling (MLM) objectives using domain-specific corpora. This includes ClinicalBERT (Alsentzer et al., 2019), BlueBERT (Peng et al., 2019), BioBERT (Lee et al., 2020), SciBERT (Beltagy et al., 2019), and PubMedBERT (Gu et al., 2021). We categorize these

Encoder	Knowledge Type					AUC
	Biomedical	*Clinical*	*Type*	*Triples*	*Synonyms*	
NON-EXPERT MODELS						
BERT						0.72
ClinicalBERT	✓	✓				0.73
BlueBERT	✓					0.78
SciBERT	✓					0.78
BioBERT	✓					0.79
PubMedBERT	✓					**0.80**
EXPERT KNOWLEDGE MODELS						
MedType	✓		✓			0.77
KeBioLM	✓			✓		**0.80**
UmlsBERT	✓	✓	✓			0.75
SapBERT	✓				✓	0.78

Table 11: Expert and non-expert pre-trained language models performance on MEDDISTANT19.

models as non-experts.

Secondly, we consider expert models that modify the MLM objective or introduce new pre-training tasks using external knowledge, such as UMLS. MedType (Vashishth et al., 2021), initialized with BioBERT, is pre-trained to predict semantic types. KeBioLM (Yuan et al., 2021), initialized with PubMedBERT, uses relational knowledge by initializing the entity embeddings with TransE (Bordes et al., 2013), improving entity-centric tasks, including RE. UmlsBERT (Michalopoulos et al., 2021), initialized with ClinicalBERT, modifies MLM to mask words belonging to the same CUI and further introduces semantic type embeddings. SapBERT (Liu et al., 2021), initialized with PubMedBERT, introduces a metric learning task for clustering synonyms together in an embedding space.

Table 11 shows the results of these sentence encoders fine-tuned on the MEDDISTANT19 dataset at sentence-level with AVG pooling. Without domain-specific knowledge, BERT performs slightly worse than the lowest-performing biomedical model, highlighting the presence of shallow heuristics in the data common to the general and biomedical domains. While domain-specific pre-training improves the results, similar to Gu et al. (2021), we find clinical LMs underperform on the biomedical RE task. There was no performance gap between BlueBERT, SciBERT, and BioBERT. However, PubMedBERT brought improvement, consistent with Gu et al. (2021).

For expert knowledge-based models, we noted a negative impact on performance. While we would expect type-based models, MedType and UmlsBERT, to bring improvement, their effect can be attributed to overfitting certain types and patterns. KeBioLM, initialized with PubMedBERT, has the same performance despite seeing the triples used in MEDDISTANT19 during pre-training, highlighting the difficulty of the Bio-DSRE. SapBERT, which uses the knowledge of synonyms, also hurt PubMedBERT's performance, suggesting that while synonyms can help in entity linking, RE is a more challenging task in noisy real-world scenarios.

5 Discussion

In the biomedical domain, health experts are often concerned with a particular type of interaction, for example, drug-target and gene-disease. However, the number of ontologies is constantly growing (222 in UMLS2019AB), thus a growing need for a more general purpose relation extraction benchmark. Broad-coverage benchmarks exist for biomedical entity linking, such as MedMentions (Mohan and Li, 2018), but they still lack many important concepts involved in relational learning. The research community has come up with several RE benchmarks (see Table 1), but the challenge remains as new entities, and relations emerge with the constant growth of biomedical literature. Hence, constructing a broad benchmark for biomedical RE is challenging due to domain requirements; nonetheless, having an accurate benchmark could offer a utility for future research. We supplement this discussion with Appendix D for a note on limitations.

Further, the train-test overlap highlights the need to systematically assess the proposed benchmarks for inconsistencies that can overestimate the model performance. Similar assessments have shown up in QA generalization where train-test overlap inflates the model performance (Liu et al., 2022). Related to RE generalization, Rosenman et al. (2020) exposed shallow heuristics while Taillé et al. (2021) showed that neural RE models could retain triples, primarily due to type hints. MEDDISTANT19 partially addresses these issues by an inductive setup that can offer insights into the generalization trend in biomedical RE using unseen entities.

6 Conclusion

In this work, we highlighted a need for an accurate broad-coverage benchmark for Bio-DSRE. We bridged this gap by utilizing SNOMED CT for constructing the benchmark and laying out the best practices. We thoroughly evaluated the benchmark with baselines and state-of-the-art, showing there is room to conduct further research.

Acknowledgments

The authors would like to thank the anonymous reviewers for their helpful feedback and William Hogan for assistance in providing data and code for AMIL experiments. The work was partially funded by the European Union (EU) Horizon 2020 research and innovation program through the projects Precise4Q (777107) and Clarify (875160), and the German Federal Ministry of Education and Research (BMBF) through the projects CoRA4NLP (01IW20010) and XAINES (01IW20005). The authors also acknowledge the computing resources provided by the DFKI and UCL.

Legal & Ethical Considerations

Does the dataset contain information that might be considered sensitive or confidential? (e.g. personally identifying information) We use PubMed MEDLINE abstracts (Canese and Weis, 2013)[7] that are publicly available and is distributed by National Library of Medicine (NLM). These texts are in the biomedical and clinical domains and are almost entirely in English. It is standard to use this corpus as a text source in several biomedical LMs (Gu et al., 2021). We cannot claim the guarantee that it does not contain any confidential or sensitive information e.g, it has clinical findings mentioned throughout the abstracts such as *A twenty-six-year-old male presented with high-grade fever*, which identifies the age and gender of a patient but not the identity. We did not perform a thorough analysis to distill such information since it is in the public domain.

References

Asma Ben Abacha and Pierre Zweigenbaum. 2011. Automatic extraction of semantic relations between medical entities: a rule based approach. *Journal of Biomedical Semantics*, 2(5):1–11.

Emily Alsentzer, John Murphy, William Boag, Wei-Hung Weng, Di Jindi, Tristan Naumann, and Matthew McDermott. 2019. Publicly available clinical BERT embeddings. In *Proceedings of the 2nd Clinical Natural Language Processing Workshop*, pages 72–78, Minneapolis, Minnesota, USA. Association for Computational Linguistics.

Christoph Alt, Aleksandra Gabryszak, and Leonhard Hennig. 2020. TACRED revisited: A thorough evaluation of the TACRED relation extraction task. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 1558–1569, Online. Association for Computational Linguistics.

Christoph Alt, Marc Hübner, and Leonhard Hennig. 2019. Fine-tuning pre-trained transformer language models to distantly supervised relation extraction. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 1388–1398, Florence, Italy. Association for Computational Linguistics.

Saadullah Amin, Katherine Ann Dunfield, Anna Vechkaeva, and Günter Neumann. 2020a. A Data-driven Approach for Noise Reduction in Distantly Supervised Biomedical Relation Extraction. In *Proceedings of the 19th SIGBioMed Workshop on Biomedical Language Processing*, pages 187–194, Online. Association for Computational Linguistics.

Saadullah Amin and Günter Neumann. 2021. T2NER: Transformers based Transfer Learning Framework for Named Entity Recognition. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: System Demonstrations*, pages 212–220. Association for Computational Linguistics.

Saadullah Amin, Günter Neumann, Katherine Ann Dunfield, Anna Vechkaeva, Kathryn Annette Chapman, and Morgan Kelly Wixted. 2019. MLT-DFKI at CLEF eHealth 2019: Multi-label Classification of ICD-10 Codes with BERT. In *Conference and Labs of the Evaluation Forum (Working Notes)*, pages 1–15.

Saadullah Amin, Stalin Varanasi, Katherine Ann Dunfield, and Günter Neumann. 2020b. LowFER: Low-rank Bilinear Pooling for Link Prediction. In *Proceedings of the 37th International Conference on Machine Learning*, volume 119 of *Proceedings of Machine Learning Research*, pages 257–268. PMLR.

Iz Beltagy, Kyle Lo, and Arman Cohan. 2019. SciBERT: A pretrained language model for scientific text. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 3615–3620, Hong Kong, China. Association for Computational Linguistics.

Olivier Bodenreider. 2004. The unified medical language system (UMLS): integrating biomedical terminology. *Nucleic acids research*, 32(suppl_1):D267–D270.

Antoine Bordes, Nicolas Usunier, Alberto García-Durán, Jason Weston, and Oksana Yakhnenko. 2013. Translating embeddings for modeling multi-relational data. In *Advances in Neural Information Processing Systems 26: 27th Annual Conference on Neural Information Processing Systems 2013. Proceedings of a meeting held December 5-8, 2013,*

[7] https://lhncbc.nlm.nih.gov/ii/information/MBR/Baselines/2019.html

Lake Tahoe, Nevada, United States, pages 2787–2795.

Àlex Bravo, Janet Piñero, Núria Queralt-Rosinach, Michael Rautschka, and Laura I. Furlong. 2015. Extraction of relations between genes and diseases from text and large-scale data analysis: implications for translational research. *BMC Bioinformatics*, 16(1):1–17.

Kathi Canese and Sarah Weis. 2013. PubMed: the bibliographic database. *The NCBI Handbook*, 2:1.

David Chang, Ivana Balažević, Carl Allen, Daniel Chawla, Cynthia Brandt, and Andrew Taylor. 2020. Benchmark and best practices for biomedical knowledge graph embeddings. In *Proceedings of the 19th SIGBioMed Workshop on Biomedical Language Processing*, pages 167–176, Online. Association for Computational Linguistics.

The Gene Ontology Consortium. 2018. The Gene Ontology Resource: 20 years and still GOing strong. *Nucleic Acids Research*, 47(D1):D330–D338.

Mark Craven and Johan Kumlien. 1999. Constructing biological knowledge bases by extracting information from text sources. In *Proceedings of the Seventh International Conference on Intelligent Systems for Molecular Biology*, page 77–86. AAAI Press.

Qin Dai, Naoya Inoue, Paul Reisert, Ryo Takahashi, and Kentaro Inui. 2019. Distantly supervised biomedical knowledge acquisition via knowledge graph based attention. In *Proceedings of the Workshop on Extracting Structured Knowledge from Scientific Publications*, pages 1–10, Minneapolis, Minnesota. Association for Computational Linguistics.

Daniel Daza, Michael Cochez, and Paul Groth. 2021. Inductive entity representations from text via link prediction. In *Proceedings of the Web Conference 2021*, WWW '21, page 798–808, New York, NY, USA. Association for Computing Machinery.

Tim Dettmers, Pasquale Minervini, Pontus Stenetorp, and Sebastian Riedel. 2018. Convolutional 2d knowledge graph embeddings. In *Proceedings of the Thirty-Second AAAI Conference on Artificial Intelligence, (AAAI-18), the 30th innovative Applications of Artificial Intelligence (IAAI-18), and the 8th AAAI Symposium on Educational Advances in Artificial Intelligence (EAAI-18), New Orleans, Louisiana, USA, February 2-7, 2018*, pages 1811–1818. AAAI Press.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, Minneapolis, Minnesota. Association for Computational Linguistics.

Rezarta Islamaj Doğan, Robert Leaman, and Zhiyong Lu. 2014. NCBI disease corpus: a resource for disease name recognition and concept normalization. *Journal of Biomedical Informatics*, 47:1–10.

Xin Dong, Evgeniy Gabrilovich, Geremy Heitz, Wilko Horn, Ni Lao, Kevin Murphy, Thomas Strohmann, Shaohua Sun, and Wei Zhang. 2014. Knowledge vault: a web-scale approach to probabilistic knowledge fusion. In *The 20th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, KDD '14, New York, NY, USA - August 24 - 27, 2014*, pages 601–610. ACM.

Tianyu Gao, Xu Han, Yuzhuo Bai, Keyue Qiu, Zhiyu Xie, Yankai Lin, Zhiyuan Liu, Peng Li, Maosong Sun, and Jie Zhou. 2021. Manual evaluation matters: Reviewing test protocols of distantly supervised relation extraction. In *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*, pages 1306–1318, Online. Association for Computational Linguistics.

Yu Gu, Robert Tinn, Hao Cheng, Michael Lucas, Naoto Usuyama, Xiaodong Liu, Tristan Naumann, Jianfeng Gao, and Hoifung Poon. 2021. Domain-specific language model pretraining for biomedical natural language processing. *ACM Transactions on Computing for Healthcare*, 3(1).

Xu Han, Tianyu Gao, Yankai Lin, Hao Peng, Yaoliang Yang, Chaojun Xiao, Zhiyuan Liu, Peng Li, Jie Zhou, and Maosong Sun. 2020. More data, more relations, more context and more openness: A review and outlook for relation extraction. In *Proceedings of the 1st Conference of the Asia-Pacific Chapter of the Association for Computational Linguistics and the 10th International Joint Conference on Natural Language Processing*, pages 745–758, Suzhou, China. Association for Computational Linguistics.

Xu Han, Tianyu Gao, Yuan Yao, Deming Ye, Zhiyuan Liu, and Maosong Sun. 2019. OpenNRE: An open and extensible toolkit for neural relation extraction. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP): System Demonstrations*, pages 169–174, Hong Kong, China. Association for Computational Linguistics.

Xu Han, Zhiyuan Liu, and Maosong Sun. 2018. Neural knowledge acquisition via mutual attention between knowledge graph and text. In *Proceedings of the Thirty-Second AAAI Conference on Artificial Intelligence, (AAAI-18), the 30th innovative Applications of Artificial Intelligence (IAAI-18), and the 8th AAAI Symposium on Educational Advances in Artificial Intelligence (EAAI-18), New Orleans, Louisiana, USA, February 2-7, 2018*, pages 4832–4839. AAAI Press.

María Herrero-Zazo, Isabel Segura-Bedmar, Paloma Martínez, and Thierry Declerck. 2013. The DDI corpus: An annotated corpus with pharmacological substances and drug-drug interactions. *Journal of Biomedical Informatics*, 46(5):914–920.

William P Hogan, Molly Huang, Yannis Katsis, Tyler Baldwin, Ho-Cheol Kim, Yoshiki Baeza, Andrew Bartko, and Chun-Nan Hsu. 2021. Abstractified Multi-instance Learning (AMIL) for Biomedical Relation Extraction. In *3rd Conference on Automated Knowledge Base Construction*.

Lixiang Hong, Jinjian Lin, Shuya Li, Fangping Wan, Hui Yang, Tao Jiang, Dan Zhao, and Jianyang Zeng. 2020. A novel machine learning framework for automated biomedical relation extraction from large-scale literature repositories. *Nature Machine Intelligence*, 2:347–355.

Halil Kilicoglu, Graciela Rosemblat, Marcelo Fiszman, and Thomas C Rindflesch. 2011. Constructing a semantic predication gold standard from the biomedical literature. *BMC Bioinformatics*, 12(1):486.

Halil Kilicoglu, Graciela Rosemblat, Marcelo Fiszman, and Dongwook Shin. 2020. Broad-coverage biomedical relation extraction with SemRep. *BMC Bioinformatics*, 21(1):188.

Diederik P. Kingma and Jimmy Ba. 2015. Adam: A method for stochastic optimization. In *ICLR (Poster)*.

Martin Krallinger, Obdulia Rabal, Saber A Akhondi, Martın Pérez Pérez, Jesús Santamaría, Gael Pérez Rodríguez, Georgios Tsatsaronis, and Ander Intxaurrondo. 2017. Overview of the BioCreative VI chemical-protein interaction Track. In *Proceedings of the sixth BioCreative challenge evaluation workshop*, volume 1, pages 141–146.

Jinhyuk Lee, Wonjin Yoon, Sungdong Kim, Donghyeon Kim, Sunkyu Kim, Chan Ho So, and Jaewoo Kang. 2020. BioBERT: a pre-trained biomedical language representation model for biomedical text mining. *Bioinformatics*, 36(4):1234–1240.

Jiao Li, Yueping Sun, Robin J. Johnson, Daniela Sciaky, Chih-Hsuan Wei, Robert Leaman, Allan Peter Davis, Carolyn J. Mattingly, Thomas C. Wiegers, and Zhiyong Lu. 2016. BioCreative V CDR task corpus: a resource for chemical disease relation extraction. *Database*, 2016. Baw068.

ChunYang Liu, WenBo Sun, WenHan Chao, and WanXiang Che. 2013. Convolution neural network for relation extraction. In *Advanced Data Mining and Applications*, pages 231–242, Berlin, Heidelberg. Springer Berlin Heidelberg.

Fangyu Liu, Ehsan Shareghi, Zaiqiao Meng, Marco Basaldella, and Nigel Collier. 2021. Self-alignment pretraining for biomedical entity representations. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 4228–4238, Online. Association for Computational Linguistics.

Linqing Liu, Patrick Lewis, Sebastian Riedel, and Pontus Stenetorp. 2022. Challenges in generalization in open domain question answering. In *Findings of the Association for Computational Linguistics: NAACL 2022*, pages 2014–2029, Seattle, United States. Association for Computational Linguistics.

Yuan Luo, Özlem Uzuner, and Peter Szolovits. 2017. Bridging semantics and syntax with graph algorithms—state-of-the-art of extracting biomedical relations. *Briefings in Bioinformatics*, 18(1):160–178.

Stefano Marchesin and Gianmaria Silvello. 2022. TBGA: a large-scale gene-disease association dataset for biomedical relation extraction. *BMC Bioinformatics*, 23(1):1–16.

George Michalopoulos, Yuanxin Wang, Hussam Kaka, Helen Chen, and Alexander Wong. 2021. UmlsBERT: Clinical domain knowledge augmentation of contextual embeddings using the Unified Medical Language System Metathesaurus. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 1744–1753, Online. Association for Computational Linguistics.

Tomás Mikolov, Ilya Sutskever, Kai Chen, Gregory S. Corrado, and Jeffrey Dean. 2013. Distributed representations of words and phrases and their compositionality. In *Advances in Neural Information Processing Systems 26: 27th Annual Conference on Neural Information Processing Systems 2013. Proceedings of a meeting held December 5-8, 2013, Lake Tahoe, Nevada, United States*, pages 3111–3119.

Mike Mintz, Steven Bills, Rion Snow, and Daniel Jurafsky. 2009. Distant supervision for relation extraction without labeled data. In *Proceedings of the Joint Conference of the 47th Annual Meeting of the ACL and the 4th International Joint Conference on Natural Language Processing of the AFNLP*, pages 1003–1011, Suntec, Singapore. Association for Computational Linguistics.

Sunil Mohan and Donghui Li. 2018. MedMentions: A Large Biomedical Corpus Annotated with UMLS Concepts. In *Automated Knowledge Base Construction (AKBC)*.

Stuart J Nelson, Kelly Zeng, John Kilbourne, Tammy Powell, and Robin Moore. 2011. Normalized names for clinical drugs: RxNorm at 6 years. *Journal of the American Medical Informatics Association*, 18(4):441–448.

Mark Neumann, Daniel King, Iz Beltagy, and Waleed Ammar. 2019. ScispaCy: Fast and robust models for biomedical natural language processing. In *Proceedings of the 18th BioNLP Workshop and Shared Task*, pages 319–327, Florence, Italy. Association for Computational Linguistics.

Maximilian Nickel, Kevin Murphy, Volker Tresp, and Evgeniy Gabrilovich. 2016. A review of relational machine learning for knowledge graphs. *Proc. IEEE*, 104(1):11–33.

Hao Peng, Tianyu Gao, Xu Han, Yankai Lin, Peng Li, Zhiyuan Liu, Maosong Sun, and Jie Zhou. 2020. Learning from Context or Names? An Empirical Study on Neural Relation Extraction. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 3661–3672, Online. Association for Computational Linguistics.

Yifan Peng, Chih-Hsuan Wei, and Zhiyong Lu. 2016. Improving chemical disease relation extraction with rich features and weakly labeled data. *Journal of Cheminformatics*, 8(1):1–12.

Yifan Peng, Shankai Yan, and Zhiyong Lu. 2019. Transfer learning in biomedical natural language processing: An evaluation of BERT and ELMo on ten benchmarking datasets. In *Proceedings of the 18th BioNLP Workshop and Shared Task*, pages 58–65, Florence, Italy. Association for Computational Linguistics.

Jeffrey Pennington, Richard Socher, and Christopher Manning. 2014. GloVe: Global vectors for word representation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1532–1543, Doha, Qatar. Association for Computational Linguistics.

Fabio Petroni, Tim Rocktäschel, Sebastian Riedel, Patrick Lewis, Anton Bakhtin, Yuxiang Wu, and Alexander Miller. 2019. Language models as knowledge bases? In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 2463–2473, Hong Kong, China. Association for Computational Linguistics.

Janet Piñero, Juan Manuel Ramírez-Anguita, Josep Saüch-Pitarch, Francesco Ronzano, Emilio Centeno, Ferran Sanz, and Laura I. Furlong. 2020. The DisGeNET knowledge platform for disease genomics: 2019 update. *Nucleic acids research*, 48(D1):D845–D855.

S Povey, R Lovering, E Bruford, M Wright, M Lush, and H Wain. 2001. The HUGO Gene Nomenclature Committee (HGNC). *Hum Genet*, 109(6):678–680.

Sebastian Riedel, Limin Yao, and Andrew McCallum. 2010. Modeling relations and their mentions without labeled text. In *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*, pages 148–163, Berlin, Heidelberg. Springer, Springer Berlin Heidelberg.

Roland Roller and Mark Stevenson. 2014. Self-supervised relation extraction using UMLS. In *International Conference of the Cross-Language Evaluation Forum for European Languages*, pages 116–127. Springer.

Shachar Rosenman, Alon Jacovi, and Yoav Goldberg. 2020. Exposing Shallow Heuristics of Relation Extraction Models with Challenge Data. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 3702–3710, Online. Association for Computational Linguistics.

Isabel Segura-Bedmar, Paloma Martínez, and María Herrero-Zazo. 2013. SemEval-2013 task 9 : Extraction of drug-drug interactions from biomedical texts (DDIExtraction 2013). In *Second Joint Conference on Lexical and Computational Semantics (*SEM), Volume 2: Proceedings of the Seventh International Workshop on Semantic Evaluation (SemEval 2013)*, pages 341–350, Atlanta, Georgia, USA. Association for Computational Linguistics.

Isabel Segura-Bedmar, Paloma Martínez Fernández, and Daniel Sánchez Cisneros. 2011. The 1st DDIExtraction-2011 challenge task: Extraction of Drug-Drug Interactions from biomedical texts. *CEUR Workshop Proceedings 761*.

Bruno Taillé, Vincent Guigue, Geoffrey Scoutheeten, and Patrick Gallinari. 2021. Separating retention from extraction in the evaluation of end-to-end Relation Extraction. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 10438–10449, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Erik M. van Mulligen, Annie Fourrier-Reglat, David Gurwitz, Mariam Molokhia, Ainhoa Nieto, Gianluca Trifiro, Jan A. Kors, and Laura I. Furlong. 2012. The EU-ADR corpus: annotated drugs, diseases, targets, and their relationships. *Journal of Biomedical Informatics*, 45(5):879–884.

Shikhar Vashishth, Denis Newman-Griffis, Rishabh Joshi, Ritam Dutt, and Carolyn P. Rosé. 2021. Improving broad-coverage medical entity linking with semantic type prediction and large-scale datasets. *Journal of Biomedical Informatics*, 121:103880.

David S Wishart, Yannick D Feunang, An C Guo, Elvis J Lo, Ana Marcu, Jason R Grant, Tanvir Sajed, Daniel Johnson, Carin Li, Zinat Sayeeda, et al. 2018. DrugBank 5.0: a major update to the DrugBank database for 2018. *Nucleic acids research*, 46(D1):D1074–D1082.

Shanchan Wu and Yifan He. 2019. Enriching pre-trained language model with entity information for relation classification. In *Proceedings of the 28th ACM International Conference on Information and Knowledge Management, CIKM 2019, Beijing, China, November 3-7, 2019*, pages 2361–2364. ACM.

Rui Xing, Jie Luo, and Tengwei Song. 2020. BioRel: towards large-scale biomedical relation extraction. *BMC Bioinformatics*, 21-S(16):543.

Rong Xu and QuanQiu Wang. 2014. Automatic construction of a large-scale and accurate drug-side-effect association knowledge base from biomedical literature. *Journal of Biomedical Informatics*, 51:191–199.

Liang Yao, Chengsheng Mao, and Yuan Luo. 2019. KG-BERT: BERT for knowledge graph completion. *ArXiv preprint*, abs/1909.03193.

Zheng Yuan, Yijia Liu, Chuanqi Tan, Songfang Huang, and Fei Huang. 2021. Improving biomedical pretrained language models with knowledge. In *Proceedings of the 20th Workshop on Biomedical Language Processing*, pages 180–190, Online. Association for Computational Linguistics.

Daojian Zeng, Kang Liu, Yubo Chen, and Jun Zhao. 2015. Distant supervision for relation extraction via piecewise convolutional neural networks. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 1753–1762, Lisbon, Portugal. Association for Computational Linguistics.

A UMLS

This section presents additional details about UMLS, including the final set of relations considered in MEDDISTANT19 (with their inverses obtained from the UMLS) and a complete list of semantic types (STY). Since, in relation extraction (RE), we are not interested in bidirectional extractions, therefore it is sufficient to only model one direction. Previous studies (Xing et al., 2020; Amin et al., 2020a; Hogan et al., 2021) fail to account the inverse relations, and with naive split, it can lead to train-test leakages. For more discussion on the relations in UMLS, including transitive closures, see Section 3.1 in Chang et al. (2020). We used UMLS2019AB to be consistent with the prior works.

A.1 UMLS Files

In UMLS (Bodenreider, 2004), a concept is provided with a unique identifier called Concept Unique Identifier (CUI), a term status (TS), and whether or not the term is preferred (TTY) in a given vocabulary, e.g., SNOMED CT. The concepts are stored in a file distributed by UMLS called `MRCONSO.RRF`.[8] Each concept further belongs to one or more semantic types (STY), provided in a file called `MRSTY.RRF`, with a type identifier TUI. There are 127 STY[9] in the UMLS2019AB version, which are mapped to 15 semantic groups (SG).[10]. The relationships between the concepts are organized in a multi-relational graph distributed in a file called `MRREL.RRF`[11]. The final set of relations considered in MEDDISTANT19 is presented in Table A.3.

Note that we only consider relations belonging to the *RO* (*has a relationship other than synonymous, narrower, or broader*) type, which is consistent with prior works. This consideration ignores relations such as *isa*, which defines hierarchy among relations.

[8] https://www.ncbi.nlm.nih.gov/books/NBK9685/table/ch03.T.concept_names_and_sources_file_mr/

[9] https://lhncbc.nlm.nih.gov/ii/tools/MetaMap/Docs/SemanticTypes_2018AB.txt

[10] https://lhncbc.nlm.nih.gov/ii/tools/MetaMap/Docs/SemGroups_2018.txt

[11] https://www.ncbi.nlm.nih.gov/books/NBK9685/table/ch03.T.related_concepts_file_mrrel_rrf/?report=objectonly

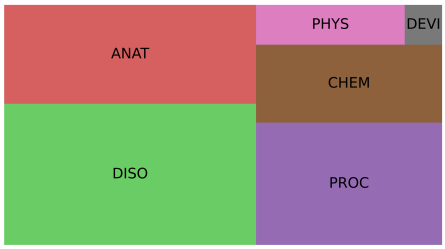

Figure A.1: Relative proportions of the entities present in MEDDISTANT19, based on the semantic groups.

A.2 Semantic Groups and Semantic Types

As we noted in Fig. 3, entities and relations follow a long-tail distribution. This has a major impact on the quality of the dataset created. For example, in the general domain, the standard benchmark NYT10 (Riedel et al., 2010) has more than half of the positive instances belonging to one relation type `/location/location/contains`. Fig. A.1 shows the relative proportions of the semantic groups in MEDDISTANT19.

Further, we used an inductive split set with 70, 10, and 20 proportions of train, validation, and test splits for constructing MEDDISTANT19. Below is an example instance from the dataset in OpenNRE (Han et al., 2019) format:

```
{
  "text": "In one patient who
  showed an increase of plasma
  prolactin level , associated
  with low testosterone and
  LH , a microadenoma
  of the pituitary gland
  ( prolactinoma ) was
  detected .",
  "h": {
    "id": "C0032005",
    "pos": [130, 145],
    "name": "pituitary gland"
  },
  "t": {
    "id": "C0033375",
    "pos": [148, 160],
    "name": "prolactinoma"
  },
  "relation": "finding_site_of"
}

/-----------------------------/
```

```
{
  "text": "Severe heart disease
  may result in cardiac cirrhosis
  in the elderly , with ascites
  and hepatomegaly .",
  "h": {
    "id": "C0018799",
    "pos": [7, 20],
    "name": "heart disease"
  },
  "t": {
    "id": "C0085699",
    "pos": [35, 52],
    "name": "cardiac cirrhosis"
  },
  "relation": "cause_of"
}

/-----------------------------/

{
  "text": "Complications
  closely associated to the
  osteosynthesis appeared
  only in instable
  fractures ( 7 % ) .",
  "h": {
    "id": "C0016658",
    "pos": [81, 90],
    "name": "fractures"
  },
  "t": {
    "id": "C0016642",
    "pos": [40, 54],
    "name": "osteosynthesis"
   },
   "relation":
   "direct_morphology_of"
}

/-----------------------------/

{
  "text": "Gluten proteins ,
  the culprits in celiac
  disease ( CD ) , show
  striking similarities in
  primary structure with
  human salivary proline-rich
  proteins ( PRPs ) .",
  "h": {
    "id": "C2362561",
    "pos": [0, 15],
    "name": "Gluten proteins"
  },
  "t": {
    "id": "C0007570",
    "pos": [34, 48],
    "name": "celiac disease"
  },
  "relation":
  "causative_agent_of"
}

/-----------------------------/

{
  "text": "Postherpetic
  neuralgia is an unfortunate
  aftermath of shingles ,
  and is most likely to
  develop , and most
  persistent , in elderly
  patients .",
  "h": {
    "id": "C0032768",
    "pos": [0, 22],
    "name": "Postherpetic
    neuralgia"
  },
  "t": {
    "id": "C0019360",
    "pos": [54, 62],
    "name": "shingles"
  },
  "relation": "occurs_after"
}
```


B UMLS License Agreement

To use the MEDDISTANT19 benchmark, the user must have signed the UMLS agreement[12]. The UMLS agreement requires those who use the UMLS (Bodenreider, 2004) to file a brief report once a year to summarize their use of the UMLS. It also requires acknowledging that the UMLS contains copyrighted material and that those restrictions are respected. The UMLS agreement requires users to agree to obtain agreements for *each* copyrighted source before its use within a commercial or production application.

[12] https://uts.nlm.nih.gov/license.html

C Risks

While our work does not have direct risk, we provide the dataset while asking users to respect the UMLS license before downloading it. This user agreement is needed to use our benchmark and to respect the source ontologies licenses. We provide this with the hope to accelerate reproducible research in Bio-DSRE by having ready-to-use corpora, with only the condition that the user has obtained the license. We provide users with this note and hope this will be respected. However, there is a risk that users may download the data and redistribute it without respecting the UMLS license. In case of such exploitation, we will add the UMLS authentication layer to protect data, where the user will be required to provide a UMLS API key, which will be validated, and only then will the data be allowed to be downloaded.

D Limitations

We provide several limitations of our work as presented in its current form. MEDDISTANT19 aims to introduce a new benchmark with good practices. However, it is still limited in its scope of ontologies considered. It also has a limited subset of relation types provided by UMLS. For example, the current benchmark does not include an important relation *may_treat*, because it is outside SNOMED CT. Since MEDDISTANT19 is focused on SNOMED CT, it lacks coverage of important protein-protein interactions, drug side-effects, and relations involving genes as provided by RxNorm (Nelson et al., 2011), Gene Ontology (Consortium, 2018), etc.

MEDDISTANT19 is automatically-created and susceptible to noise and thus needs to be approached carefully as a potential source for biomedical knowledge. While the dataset was not created to represent *true* biomedical knowledge, it has the potential to be treated as a reliable reference.

E Experimental Setup and Hyperparameters

We followed the experimental setup of Gao et al. (2021) for BERT-based experiments. Specifically, we used batch size 64, with a learning rate of 2e-5, maximum sequence length 128, and bag size 4. We used a single NVIDIA Tesla V100-32GB for BERT-based experiments. Each experiment took about 1.5hrs, with half an hour per epoch. We also attempted to perform a grid search for BERT experiments, but it was too expensive to continue; therefore, we abandoned those jobs. Since we only used the `base` models, they amount to 110 million parameters. During fine-tuning, we do not freeze any parts of the model.

Encoder	Bag Size	Batch Size	Embedding
CNN+sent+AVG	-	128	biowordvec
CNN+sent+ONE	-	128	biowordvec
CNN+bag+AVG	8	128	GloVe
CNN+bag+ONE	16	256	GloVe
CNN+bag+ATT	8	256	GloVe
PCNN+sent+AVG	-	128	biowordvec
PCNN+sent+ONE	-	128	biowordvec
PCNN+bag+AVG	4	128	GloVe
PCNN+bag+ONE	8	128	GloVe
PCNN+bag+ATT	8	128	GloVe
GRU+sent+AVG	-	128	biowordvec
GRU+sent+ONE	-	128	biowordvec
GRU+bag+AVG	8	128	biow2v
GRU+bag+ONE	16	256	GloVe
GRU+bag+ATT	16	128	GloVe

Table A.1: Best hyperparameters for CNN, PCNN, and GRU sentence encoders.

For CNN and PCNN, we performed grid search with Adam (Kingma and Ba, 2015) optimizer using learning rate 0.001 for 20 epochs with: batch size $\in \{128, 256\}$, bag size $\in \{4, 8, 16, 32\}$, 200-d word embeddings $\in$ {Word2Vec (Mikolov et al., 2013)[13], GloVe (Pennington et al., 2014)}, and with (test-time) pooling $\in \{\text{ONE}, \text{AVG}\}$ when using sentence-level training and pooling in $\{\text{ONE}, \text{AVG}, \text{ATT}\}$ when using bag-level training. We ran this job on a cluster with support for array jobs. These amounted to over 700 experiments and took 3 days. We fixed other hyperparameters from literature (Han et al., 2018), with position dimension set to 5, kernel size set to 3, and dropout set to 0.5. These are also default in OpenNRE (Han et al., 2019). The hyperparameters that had the most influence were batch size, bag size, and pre-trained word embeddings. All the experiments reported in this work are with a single run.

For sentence tokenization with ScispaCy, it took 9hrs with 32 CPUs (4GB each) and a batch size of 1024 to extract 151M sentences. Further, the ScispaCy entity linking job took about half TB of RAM with 72 CPUs (6GB each) with a batch size of 4096 with 40hrs of run-time to link 145M unique sentences.

[13] We used domain-specific word embeddings *biowordvec* and *biow2v* following Marchesin and Silvello (2022).

Semantic Type	10k-20k	20k-30k	$\geq$ 30k
Body Part, Organ, or Organ Component	*bladder, heart, retinal, lungs, spinal, kidneys, colon*	*eyes, lung, kidney, intestinal*	*liver, brain*
Organism Function	*death*	*period, blood pressure*	-
Body Location or Region	*head*	-	-
Therapeutic or Preventive Procedure	*injection, prevention, chemotherapy, application resection, infusion, treatments, therapeutic surgical treatment, CT, surgical, transplantation*	*stimulation, delivery*	*intervention, procedure, removal, operation*
Neoplastic Process	*cancer*	-	*tumor, tumors*
Disease or Syndrome	*obesity, disorder, disorders*	*diseases, stroke*	*disease, infection, condition, hypertension*
Laboratory Procedure	*test, erythrocytes*	-	*cells*
Diagnostic Procedure	*US, biopsy, ultrasound*	*MRI*	-
Finding	*lesion, interaction, mass, difficulty, dependent*	*abnormal*	*presence, positive, negative, severe, lesions*
Hormone	*insulin*	-	-
Biologically Active Substance	*amino acids, glucose, ATP*	*protein, proteins*	
Pharmacologic Substance	*medication*	-	*drugs, drug*
Injury or Poisoning	*strains*	*injury, exposure*	*damage*
Tissue	*tissue, bone marrow, tissues*	-	-
Organism Attribute	*male*	-	*temperature, age*
Immunologic Factor	*antibody, antibodies*	-	-
Health Care Activity	*investigations*	*examination*	*assessment*
Body Substance	*plasma, blood, skin*	-	-
Body System	-	*cardiovascular*	-
Mental Process	-	-	*concentrations, concentration*
Congenital Abnormality	-	*abnormalities*	-

Table A.2: Semantic types affected by type-based mention pruning with removed mentions placed in their respective frequency bins as discussed in Section 3.1.

Relation	Inverse Relation
finding_site_of	*has_finding_site*
associated_morphology_of	*has_associated_morphology*
method_of	*has_method*
interprets	*is_interpreted_by*
direct_procedure_site_of	*has_direct_procedure_site*
causative_agent_of	*has_causative_agent*
active_ingredient_of	*has_active_ingredient*
interpretation_of	*has_interpretation*
component_of	*has_component*
indirect_procedure_site_of	*has_indirect_procedure_site*
direct_morphology_of	*has_direct_morphology*
cause_of	*due_to*
direct_substance_of	*has_direct_substance*
uses_device	*device_used_by*
focus_of	*has_focus*
direct_device_of	*has_direct_device*
procedure_site_of	*has_procedure_site*
uses_substance	*substance_used_by*
associated_finding_of	*has_associated_finding*
occurs_after	*occurs_before*
is_modification_of	*has_modification*

Table A.3: *(Left)* 21 relations included in MEDDISTANT19, excluding NA relation. *(Right)* For completeness, we also include their inverse relations.

SG	TUI	Semantic Type
ANAT	T017	Anatomical Structure
	T029	Body Location or Region
	T023	Body Part, Organ, or Organ Component
	T030	Body Space or Junction
	T031	Body Substance
	T022	Body System
	T021	Fully Formed Anatomical Structure
	T024	Tissue
CHEM	T116	Amino Acid, Peptide, or Protein
	T195	Antibiotic
	T123	Biologically Active Substance
	T103	Chemical
	T200	Clinical Drug
	T196	Element, Ion, or Isotope
	T126	Enzyme
	T131	Hazardous or Poisonous Substance
	T125	Hormone
	T129	Immunologic Factor
	T130	Indicator, Reagent, or Diagnostic Aid
	T197	Inorganic Chemical
	T114	Nucleic Acid, Nucleoside, or Nucleotide
	T109	Organic Chemical
	T121	Pharmacologic Substance
	T192	Receptor
	T127	Vitamin
DEVI	T074	Medical Device
	T075	Research Device
DISO	T020	Acquired Abnormality
	T190	Anatomical Abnormality
	T049	Cell or Molecular Dysfunction
	T019	Congenital Abnormality
	T047	Disease or Syndrome
	T033	Finding
	T037	Injury or Poisoning
	T048	Mental or Behavioral Dysfunction
	T191	Neoplastic Process
	T046	Pathologic Function
	T184	Sign or Symptom
PHYS	T201	Clinical Attribute
	T041	Mental Process
	T032	Organism Attribute
	T040	Organism Function
	T042	Organ or Tissue Function
	T039	Physiologic Function
PROC	T060	Diagnostic Procedure
	T065	Educational Activity
	T058	Health Care Activity
	T059	Laboratory Procedure
	T063	Molecular Biology Research Technique
	T062	Research Activity
	T061	Therapeutic or Preventive Procedure

Table A.4: 51 semantic types (STY) along with their TUIs and semantic groups (SG) covered in MEDDISTANT19.

Decorrelate Irrelevant, Purify Relevant: Overcome Textual Spurious Correlations from a Feature Perspective

Shihan Dou[1*], **Rui Zheng**[1*], **Ting Wu**[1], **Songyang Gao**[1], **Junjie Shan**[3], **Qi Zhang**[12], **Yueming Wu**[4], **Xuanjing Huang**[1†]
[1] School of Computer Science, Fudan University, Shanghai, China
[2] Shanghai Key Laboratory of Intelligent Information Processing, Fudan University
[3] KTH Royal Institute of Technology, Stockholm, Sweden
[4] Nanyang Technological University, Singapore
{shdou21,tingwu21,gaosy21}@m.fudan.edu.cn
{rzheng20,qz,xjhuang}@fudan.edu.cn

Abstract

Natural language understanding (NLU) models tend to rely on spurious correlations (*i.e.*, dataset bias) to achieve high performance on in-distribution datasets but poor performance on out-of-distribution ones. Most of the existing debiasing methods often identify and weaken these samples with biased features (*i.e.*, superficial surface features that cause such spurious correlations). However, down-weighting these samples obstructs the model in learning from the non-biased parts of these samples. To tackle this challenge, in this paper, we propose to eliminate spurious correlations in a fine-grained manner from a feature space perspective. Specifically, we introduce Random Fourier Features and weighted re-sampling to decorrelate the dependencies between features to mitigate spurious correlations. After obtaining decorrelated features, we further design a mutual-information-based method to purify them, which forces the model to learn features that are more relevant to tasks. Extensive experiments on two well-studied NLU tasks demonstrate that our method is superior to other comparative approaches.

1 Introduction

Recently, researchers have found that the main reason why large-scale pre-trained language models perform well on NLU tasks is that they rely on *spurious correlations*, rather than capturing the language understanding for the intended task (Bender and Koller, 2020). These spurious correlations are also denoted as *dataset bias* in previous work (He et al., 2019; Clark et al., 2019): prediction rules that work for training examples but do not hold in general. In reality, a variety of spurious correlations appear in widely-used NLU benchmark datasets. For example, in natural language inference (NLI) tasks, McCoy et al. (2019) observe that models on the MNLI dataset (Williams et al., 2018) rely heavily on the features of word overlap to predict the entailment label blindly. Consequently, these models perform poorly on out-of-distribution (OOD) datasets where such correlations no longer hold (Nie et al., 2019).

To mitigate these spurious correlations, some existing debiasing works (Clark et al., 2019; He et al., 2019) prefer to train a *bias model* with known spurious correlations as prior knowledge to identify the samples without biased features. This trained *bias model* is used in the later stage to force the *main model* to learn from these samples. For better transferability, Utama et al. (2020b); Sanh et al. (2020) relax this basic assumption that spurious correlation is apriori by using a small part of the training dataset in the training phase of *bias model*. However, these methods are not end-to-end and their training procedures are complicated. Moreover, not all features in the samples with biased features are insignificant (Wen et al., 2021). These samples may still contain features that generalize to the real-world dataset, and weakening these samples obstructs the model in learning from the non-biased parts of these samples (Wen et al., 2021).

In this paper, unlike the above-mentioned methods, we propose an end-to-end method that can eliminate the spurious correlations in a fine-grained way[1]. Recently, some works (Marcus, 2018; Arjovsky et al., 2019) have demonstrated that spurious correlations are essentially caused by the subtle dependencies between irrelevant features (*i.e.*, the features that are irrelevant to a given label) and relevant features. According to this observation, we intend to eliminate spurious correlations by decor-

[*] Equal contribution.
[†] Corresponding author.

[1] Our code is available at https://github.com/Coling2022-DePro/DePro.

Proceedings of the 29th International Conference on Computational Linguistics, pages 2278–2287
October 12–17, 2022.

relating the dependencies between features in the feature space. However, those irrelevant features still exist in the feature space and may confuse the analysis capability of deep models. To achieve better performance, we further design another component to purify the decorrelated features in the feature space, which forces the model to learn useful local features (*i.e.*, features that are more relevant to tasks (Wang et al., 2020)). Specifically, we address two main challenges:

- *Challenge 1: How to eliminate dependencies among features in the feature space?*
- *Challenge 2: How to find the useful local features and purify the decorrelated global features with them?*

To address the first challenge, some previous works (Shen et al., 2020) try to decorrelate features under linear frameworks. However, these linear frameworks are not capable of dealing with nonlinear dependencies between features in the feature space. To further enhance the effectiveness of methods on decorrelating nonlinear dependencies, an ideal candidate is to use kernel methods to remap the original features to high-dimensional feature space. In this way, both linear and nonlinear dependencies can be decorrelated. Nevertheless, the mapping operator of the kernel function cannot be given explicitly. Therefore, we use Random Fourier Features (RFF) (Rahimi and Recht, 2007) to approximate the kernel method for the sake of computability. After completing high-dimensional feature reconstruction, we introduce weighted resampling to remove the dependencies between reconstructed features in the reconstructed feature space. To tackle the second challenge, we introduce a saliency-map-based method to identify the useful local features in the samples and design a mutual-information-based strategy to purify the decorrelated global features (*i.e.*, sentence representation) with these useful local features.

We evaluate our framework over two NLU tasks including Natural Language Inference and Fact Verification. Through the experimental results, we observe that feature decorrelation and feature purification are both useful for improving the generalization ability of deep neural models. Moreover, our method can achieve state-of-the-art performance on predicting out-of-distribution datasets compared with existing approaches. In summary, this paper makes the following contributions:

- We introduce a novel end-to-end framework that combines feature decorrelation with feature purification to strengthen the generalization ability of NLU models. The feature decorrelation phase is used to eliminate spurious correlations of features while the feature purification component is used to force the model to learn features that are more relevant to tasks.
- We conduct extensive experiments over several widely used benchmark datasets. The experimental results report that feature decorrelation and feature purification can both enhance the generalization ability of deep models. Also, the results suggest the synergistic effect between decorrelation and purification. After combining them, our proposed method outperforms the state-of-the-art methods.

2 Related Work

2.1 Spurious Correlations and Debiasing Methods

The performance of machine learning models on multiple natural language understanding benchmarks has achieved remarkable results. However, due to the presence of spurious surface lexical-syntactic features in the training phase, deep models perform poorly on out-of-distribution examples. These spurious properties are also known as spurious correlations or dataset biases. For example, McCoy et al. (2019) reports that models on the MNLI dataset (Williams et al., 2018) rely heavily on high word overlap to predict the entailment label. In fact, spurious correlations also exist in datasets of other NLU tasks such as multi-hop QA datasets (Wen et al., 2021). Deep models' excessive dependence on these spurious correlations can affect their generalization ability when testing on more challenging datasets.

In response to the problem of spurious correlations in datasets, many methods have been proposed to mitigate the impact. For example, Clark et al. (2019); He et al. (2019) propose a two-stage-based framework to reduce the model's dependence on known spurious correlations. They first train a bias-only model using known spurious correlations and then leverage it to guide the main model to distinguish biased examples. However, these approaches suffer from low transferability since they require prior knowledge about the spurious correla-

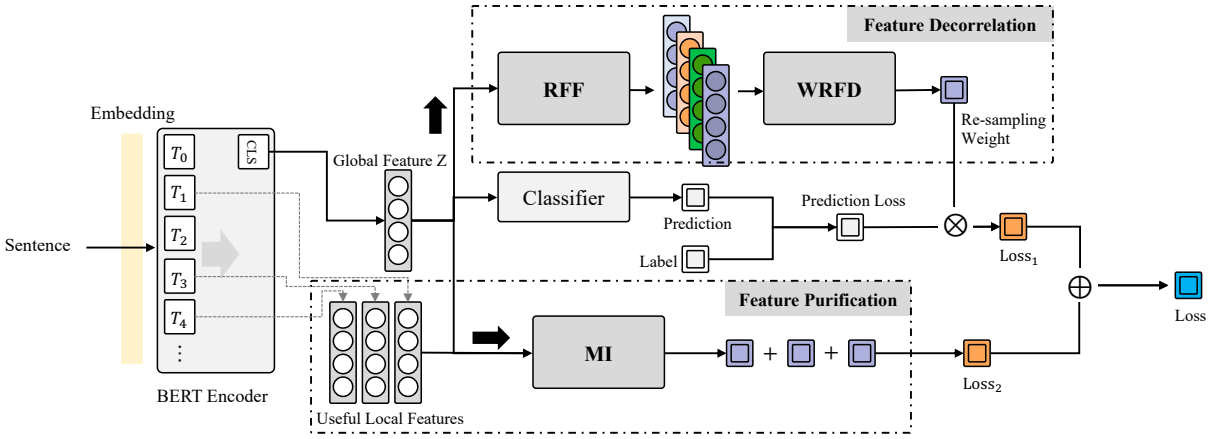

Figure 1: System architecture of *DePro*. RFF, WRFD, and MI refer to Random Fourier Features, Weighted Re-sampling for Feature Decorrelation, and Mutual Information, respectively.

tions in a dataset. To mitigate the issue, Utama et al. (2020b); Clark et al. (2020) tend to train a weak or shadow model as the bias-only model to provide guidance on discriminating biased data. However, these methods are not end-to-end and the training procedures of these methods are complicated.

2.2 Feature Decorrelation

Since the correlation between features can affect or even damage model predictions, several studies focus on eliminating this correlation during the training process. Zhang et al. (2017) propose a strategy that selects uncorrelated features in groups to decorrelate features. Shen et al. (2020) address this issue by re-weighting samples. However, these two methods can only remove the linear dependence between features which cannot tackle the complex nonlinear dependence between features. Bahng et al. (2020) propose to use the biased representations to generate a debiased representation. Although this method can decorrelate the nonlinear and linear dependence between features, it needs to artificially design the biased representation based on the known spurious correlations in the dataset. On the contrary, our method can remove all kinds of dependencies between the features and does not need to rely on prior knowledge.

3 Method

In this section, we introduce our proposed end-to-end framework namely *DePro*. Figure 1 presents the system architecture of *DePro* which mainly consists of two phases: feature decorrelation and feature purification. In the first phase, we introduce Random Fourier Features (RFF) (Rahimi and Recht, 2007) to map features from the original feature space to the reconstruction space. Then we use weighted re-sampling to remove the dependencies between reconstructed features. In the later phase, we purify the global sample features from an information theoretic perspective to further improve the generalization ability.

Notations $\mathcal{X}$, $\mathcal{Y}$, and $\mathcal{Z}$ denote the space of samples (*i.e.*, sentences), the space of labels, and the feature space, respectively. We use $f : \mathcal{X} \rightarrow \mathcal{Z}$ to denote the encoder function which can encode a sample into the feature space. The classifier function is denoted as $c : \mathcal{Z} \rightarrow \mathcal{Y}$, which can predict the sample to the corresponding label. Given a dataset $\mathcal{D}$ that consists of n pairs of sentences and labels $(X_i, Y_i)_{i\in[1,n]}$, with $X_i \in \mathcal{X}$ and $Y_i \in \mathcal{Y}$, the representation of X_i is denoted as $Z_i \in \mathcal{Z}$, and Z^i denotes the i-th variable in the feature space. For an input sentence $X_i = [X_i^1; X_i^2; \ldots; X_i^k]$, w_i denotes the re-sampling weight of this sentence X_i and we use $T_i = [T_i^1; T_i^2; \ldots; T_i^k]$ to denote the local feature of X_i in the encoder (*e.g.*, the output of BERT embedding layer).

3.1 Decorrelate Features of Feature Space

In this subsection, we mainly introduce our method of removing both the nonlinear and linear dependencies between features by using RFF and weighted re-sampling.

High-dimensional Feature Reconstruction via RFF

The kernel method can obtain mutually independent features by mapping them from the original feature space to Reproducing Kernel Hilbert Space (RKHS) (Alvarez et al., 2012) as follows:

$$\mathcal{K}(x,\cdot)=\sum_{i=1}^{\infty}\lambda_i\varphi_i(x)\varphi_i(\cdot)=\left(\sqrt{\lambda}_i\varphi_i(x),\cdots\right)_{\mathcal{H}} \quad (1)$$

Here $\mathcal{K}(\cdot,\cdot)$ is the mapping operator of a measurable, symmetric positive definite kernel function and $(\cdot)_{\mathcal{H}}$ is Hilbert-Schmidt space. However, the mapping operator $\mathcal{K}(x,\cdot)$ is implicit. In other words, the reconstructed features cannot be obtained explicitly. To mitigate this issue, we use Random Fourier Feature (RFF) (Rahimi and Recht, 2007), inspired by Zhang et al. (2021), to approximate the kernel function. The function space of Random Fourier Features is denoted as $\mathcal{H}$ with the following form:

$$\begin{aligned}\mathcal{H}=\{h:x\rightarrow\sqrt{2}\cos(\omega x+\phi)\mid\\ \omega\sim N(0,1),\phi\sim \mathrm{U}(0,2\pi)\}\end{aligned} \quad (2)$$

where ω and ϕ are sampled from any distribution.

For the i-th variable Z^i and the j-th variable Z^j of the feature space (Z^i and Z^j are represented by $\mathcal{A}$ and $\mathcal{B}$ for simplicity), we sample $n_{\mathcal{A}}$ and $n_{\mathcal{B}}$ mapping functions from $\mathcal{H}$ and denote them as $u=\{u_k\}_{k\in[1,n_{\mathcal{A}}]}$ and $v=\{v_k\}_{k\in[1,n_{\mathcal{B}}]}$. Thus, the reconstructed features $u(\mathcal{A})$ of feature A can be represented as Eq. (3) and $v(\mathcal{B})$ of feature B follows the same rule.

$$u(\mathcal{A})=\left(u_1(\mathcal{A}),\ldots,u_{n_{\mathcal{A}}}(\mathcal{A})\right),u_k(\cdot)\in\mathcal{H}_{\mathrm{RFF}},\forall k, \quad (3)$$

By mapping the two features A and B to the reconstructed space through RFF, only linear dependencies between $u(\mathcal{A})$ and $v(\mathcal{B})$ remain.

Weighted Re-sampling for Feature Decorrelation

We use cross-covariance operator Σ_{XY} to measure the independence between features as follows:

$$\begin{aligned}\langle g,\Sigma_{YX}f\rangle_{\mathcal{H}_2}=E_{XY}[f(X)g(Y)]-\\ E_X[f(X)]E_Y[g(Y)]\end{aligned} \quad (4)$$

Specifically, for $u(\mathcal{A})$ and $u(\mathcal{B})$, the cross-covariance Σ_{AB} between the distributions can be calculated by their unbiased empirical estimation with the following form:

$$\begin{aligned}\Sigma_{\mathcal{AB}}=\frac{1}{n-1}\sum_{i=1}^{n}\Big[\Big(u\left(\mathcal{A}_i\right)-\frac{1}{n}\sum_{j=1}^{n}u\left(\mathcal{A}_j\right)\Big)^T\cdot\\ \Big(v\left(\mathcal{B}_i\right)-\frac{1}{n}\sum_{j=1}^{n}v\left(B_j\right)\Big)\Big]\end{aligned} \quad (5)$$

Hilbert-Schmidt Independence Criterion (HSIC) (Gretton et al., 2007) uses the squared Hilbert-Schmidt norm of $\Sigma_{\mathcal{AB}}$ to test the independence of random variables. In the Euclidean space which the reconstructed space belongs to, Hilbert-Schmidt norm degenerates to the equivalent Frobenius norm (Zhang et al., 2021). Thus, we use Frobenius norm to calculate the linear correlation between the reconstructed features.

Suppose $P(\mathcal{A},\mathcal{B})$ is denoted as the joint distribution of features $\mathcal{A}$ and $\mathcal{B}$. Due to the complicated correlation between $\mathcal{A}$ and $\mathcal{B}$, $P(\mathcal{A},\mathcal{B})$ cannot be obtained by their respective marginal distributions, which means $P(\mathcal{A},\mathcal{B})\neq P(\mathcal{A})\cdot P(\mathcal{B})$. Inspired by the Acceptance-Rejection Sampling method (Naesseth et al., 2017) which reparameterizes the target distribution function from the standard normal distribution by introducing the proposal distribution, we use the normalized weight function instead of the rejection process to obtain a linearly independent weighted marginal distribution from the original complex joint distribution. Specifically, consider a probability density function with the independent marginal distributions of $\mathcal{A}$ and $\mathcal{B}$ as $\mathcal{Q}(\mathcal{A},\mathcal{B})=\mathcal{Q}(\mathcal{A})\cdot\mathcal{Q}(\mathcal{B})$, the $\mathcal{Q}$ can be fitted by the proposal distribution $\mathcal{P}$ and the normalized sampling weight is denoted as follows:

$$w(x)=\frac{\mathcal{Q}(\S)}{\tau\mathcal{P}(x)} \quad (6)$$

where $x\in\mathcal{H}(\mathcal{A},\mathcal{B})$ and τ is a normalization constant with the following form:

$$\tau^{-1}=\int_{x\in\mathcal{H}(\mathcal{A},\mathcal{B})}w(x)dx \quad (7)$$

Thus, the linear dependencies between reconstructed features can be removed by the normalized weight function as follows:

$$w(x)\cdot x(\mathcal{A},\mathcal{B})\sim\tau w(x)\cdot P(x)=\mathcal{Q}(\mathcal{A})\cdot\mathcal{Q}(\mathcal{B}) \quad (8)$$

In practice, we use the training dataset to learn the optimal sampling weights. Through Eq. (5) and Eq. (8), the cross-covariance with weighted re-sampling can be estimated as:

$$\begin{aligned}\tilde{\Sigma}_{\mathcal{AB};w}=\frac{1}{n-1}\sum_{i=1}^{n}\Big[\Big(w_iu\left(\mathcal{A}_i\right)-\frac{1}{n}\sum_{j=1}^{n}w_ju\left(\mathcal{A}_j\right)\Big)^T\cdot\\ \Big(w_iv\left(\mathcal{B}_i\right)-\frac{1}{n}\sum_{j=1}^{n}w_jv\left(\mathcal{B}_j\right)\Big)\Big]\end{aligned} \quad (9)$$

As aforementioned, we use Frobenius norm to measure the correlation between features (*i.e.*, $\left\|\tilde{\Sigma}_{\mathcal{AB};w}\right\|_F^2$). Thus, by optimizing w in the training process, both nonlinear and linear dependencies between features of the feature space can be eliminated. Specifically, the correlation between the two

variables Z^i and Z^j of the feature space is represented as $\left\|\tilde{\Sigma}_{Z^iZ^j;w}\right\|_F^2$. Therefore, the re-sampling weight w can be optimized as follows:

$$w^* = \arg\min_{w \in \mathcal{W}} \sum_{1 \le i < j \le m_Z} \left\|\tilde{\Sigma}_{Z^iZ^j;w}\right\|_F^2 \quad (10)$$

where $\mathcal{W} = \left\{w \in \mathbb{R}_+^n \mid \sum_{i=1}^n w_i = n\right\}$ and m_Z denotes the dimension of space $\mathcal{Z}$. We use a mini-batch to update the global weight repeatedly during the optimization process. Moreover, the optimization objective function for encoder f and classifier c can be expressed as:

$$f^*, c^* = \arg\min_{f,c} \sum_{i=1}^n w_i \mathcal{L}(c(f(X_i)), y_i) \quad (11)$$

where $\mathcal{L}(\cdot, \cdot)$ is the cross entropy loss function.

3.2 Feature Purification via Local Information

For better generalization, we propose to purify the decorrelated global features from an information theoretic perspective. Specifically, we find the useful local features by a saliency-map-based method and purify the decorrelated global features with these local features by mutual information (MI).

Inspired by Han et al. (2020), we measure the significance of all local features of the sentence by computing the absolute value of the partial derivative of loss w.r.t. these local features. The gradient of each local feature can be calculated as:

$$\mathcal{G}(T^i) = \nabla_{T^i} \ell\left(f(T), y\right) \quad (12)$$

where T^i is the i-th feature of the local features T. We consider the part of the local features with the smallest values as the useless local features (*e.g.*, stopwords and punctuation) which carry limited information and cannot be used to make predictions (Wang et al., 2020). Therefore, the information of such useless features should not be encoded into the global features of a sentence.

After feature filtering, we treat these remaining local features as useful local features that are significant to the label (Wang et al., 2020), and use them to purify the decorrelated sentence representation by mutual information. Specifically, by maximizing the mutual information between the useful local features and the decorrelated sentence representation, the useful features are retained and the useless features are compressed. In practice, we simply examine the ℓ_2 norm of the gradient $\mathcal{G}(T^i)$ of each local feature T^i. The optimization goal can be expressed as:

$$\arg\max_{f,c} \ \alpha \sum_{j=1}^M I(T^j; Z) \quad (13)$$

where α is a hyper-parameter to control the trade-off, T^j is the above-mentioned useful local semantic feature, and M is the number of remaining features. In addition, due to the intractability of computing MI, we use InfoNCE (Oord et al., 2018) as the lower bound of MI to approximate $I(T^j; Z)$.

Combining Eq. (11) and Eq. (13), the overall optimization goal can be as follows:

$$\begin{aligned} f^*, c^* = \arg\min_{f,c} \ & \sum_{i=1}^n (w_i \mathcal{L}(c(f(X_i)), Y_i) - \\ & \alpha \sum_{j=1}^M \hat{I}^{(\text{InfoNCE})}(f_T(X_i^j); f(X_i))) \end{aligned} \quad (14)$$

$$w^* = \arg\min_{w \in \mathcal{W}n} \sum_{1 \le i < j \le m_Z} \left\|\tilde{\Sigma}_{Z^iZ^j;w}\right\|_F^2 \quad (15)$$

where $f_T(\cdot)$ is the function (*i.e.*, the BERT embedding layer) that obtains the local features.

4 Experiments

In this section, we conduct extensive experiments to demonstrate (1) *DePro* outperforms the state-of-the-art comparative approaches; and (2) Both feature decorrelation and feature purification can improve the model's generalization ability.

4.1 Datasets

We conduct experiments on two well-studied NLU tasks including Natural Language Inference and Fact Verification to evaluate *DePro*. **Natural Language Inference** aims to infer the relationship between the premise and hypothesis. For this task, we use MNLI (Williams et al., 2018) as our ID data, MNLI-hard (Gururangan et al., 2018) and Heuristic Analysis for NLI Systems (HANS) (McCoy et al., 2019) as our OOD test set. **Fact Verification** aims to verify a claim by the evidence document. For this task, we use FEVER (Thorne et al., 2018) for ID evaluation and FEVER Symmetric (Schuster et al., 2019) (version 1&2) as our OOD test set.

Specifically, we report the main results and ablation studies on the test set and evaluate all the sensitivity analyses on the development set. However, for the MNLI dataset, only the train set and

Method	MNLI			FEVER		
	ID	MNLI-hard	HANS	ID	Symm. v1	Symm. v2
BERT-base	84.3	75.9	61.1	85.4	55.2	63.2
prior knowledge required						
Learned-Mixin + H (Clark et al., 2019)	84.2	-	65.8	83.3	60.4	64.9
Reg-conf (Utama et al., 2020a)	84.5	77.3	69.1	**86.4**	60.5	66.2
Reweight (Clark et al., 2019)	83.5	-	69.2	84.6	61.7	66.5
PoE + CE (He et al., 2019)	83.3	77.6	67.9	85.7	57.7	61.4
prior knowledge NOT required						
MCE (Clark et al., 2020)	83.3	77.6	64.4	-	-	-
Reg-conf (Utama et al., 2020b)	84.3	-	67.1	87.6	59.8	66.0
PoE (Sanh et al., 2020)	81.4	76.5	68.8	85.4	59.7	65.3
MoCaD (Xiong et al., 2021)	81.5	-	70.0	87.4	**65.7**	69.0
DePro (Our method)	83.2	**77.8**	**70.3**	84.5	65.2	**69.2**
w/o feature decorrelation	**84.7**	76.8	63.2	85.9	57.5	65.2
w/o feature purification	82.6	77.1	68.7	83.6	64.3	67.9

Table 1: Accuracy results on MNLI and FEVER, and out-of-distribution test sets MNLI-hard, HANS and FEVER Symmetric (version 1&2). We conduct the ablation study to further validate that our feature decorrelation and feature purification indeed improve the generalization ability. We compared 8 state-of-the-art debiasing methods including 4 debiasing methods with known bias and 4 debiasing methods with unknown bias. The hyper-parameters of BERT are identical for each model in the same dataset.

dev set are publicly available, but not the published test set. So we split 10 percent of training data into a dev set dedicated to picking hyper-parameters in order to avoid overfitting. And the original dev set of MNLI is used as the test set.

4.2 Implementation

Similar to the majority of current debiasing methods, we choose the uncased BERT-base model (Devlin et al., 2018) as our baseline. For all sentence-pair classification tasks, we concatenate the two sentences of one sentence pair into a single sequence and use the final-layer [CLS] embedding to represent the sentence representation. For BERT hyper-parameters, we use a batch size of 32, Adam optimizer with the learning rate $5e^{-5}$ for the MNLI dataset and $2e^{-5}$ for the FEVER dataset, respectively.

For feature decorrelation, we set the learning rate of weight to $1e^{-2}$ which decays with a rate of $1e^{-3}$ for the MNLI dataset, and the learning rate to $5e^{-2}$ which decays with a rate of $1e^{-3}$ for the FEVER dataset. For the parameter of Random Fourier Features dimension, we have verified through extensive experiments that our method can get the best performance on HANS, Symm. v1, and Symm. v2 when the RFF dimensions are four times, two times, and four times, respectively, that of the original feature space. For feature purification, α is set to $1e^{-4}$ to control the trade-off between feature decorrelation and feature purification.

Method w/o Prior Knowledge	End-to-End
MCE (Clark et al., 2019)	✖
Reg-conf (Utama et al., 2020b)	✖
PoE (Sanh et al., 2020)	✖
MoCaD (Xiong et al., 2021)	✖
DePro (Our method)	☑

Table 2: The structural details of state-of-the-art methods without the need for prior knowledge.

4.3 Experimental Results

Detection Performance

Table 1 shows the experimental results of *DePro* and comparative methods on the MNLI and FEVER datasets, respectively. Through the table, we can see that *DePro* can significantly improve the performance of the two NLU tasks and obtain state-of-the-art results on OOD datasets. Meanwhile, the

loss of *DePro* on ID datasets is not significant compared to other methods. Moreover, we also observe that the experimental results of the model under different random seeds have high variance, which has been demonstrated in previous works (Utama et al., 2020b). To mitigate this impact, we perform our experiments with five different seeds and report the average of these results.

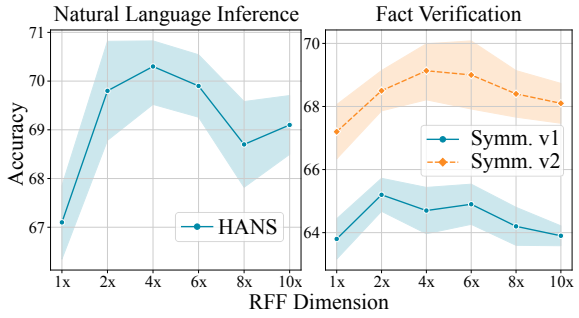

Figure 2: The results of *DePro* using different RFF dimensions. Meanwhile, the ratios of feature purification for HANS, Symm. v1, and Symm. v2 are 0.7, 0.7, and 0.6, respectively.

For the NLI task, compared to the baseline method (*i.e.*, Uncased BERT-base model), *DePro* improves by 1.9 and 9.2 percentage points on two OOD datasets MNLI-hard and HANS, respectively. The generalization ability of *DePro* on OOD datasets is also promising compared to other methods that introduce prior knowledge or unknown prior knowledge. For the Fact Verification task, *DePro* also has the best performance on the OOD dataset Symm. v2, with 6.0 percentage points higher than the accuracy of the BERT-base model. Meanwhile, the performance of our proposed method *DePro* is second only to MoCaD (Xiong et al., 2021), which is 0.5 percent lower when evaluated on Symm. v1. However, MoCaD is not an end-to-end method, but rather an improved version of the existing two-stage methods, as shown in Table 2. On the contrary, *DePro* is a complete end-to-end method, which is more flexible while preserving similar detection capabilities.

In conclusion, *DePro* outperforms the majority of state-of-the-art approaches on OOD datasets for two NLU tasks while the loss in ID datasets is acceptable.

Ablation Study

We also perform two ablation experiments to check whether feature decorrelation and purification can contribute to *DePro* or not. Through the results in Table 1, we find that feature decorrelation and purification can both boost the generalization ability of *DePro*. As aforementioned, the essence of spurious correlation is the subtle dependencies between relevant and irrelevant features. Therefore, after removing dependencies between features, we can mitigate the impact caused by spurious correlations, thus improving the model's generalization ability on OOD datasets. The results in Table 1 are consistent with this situation. On the other hand, if we directly perform feature purification on the original features, the model's performance on ID datasets can be enhanced. It is reasonable because feature purification can align the useful local features and the sentence representation, so that the representation generated by the model is more independent of useless local features, allowing the model to focus on the useful parts of the training data. After combining feature decorrelation with feature purification, *DePro* can achieve state-of-the-art performance on distinguishing samples in OOD datasets. Such results indicate that compared to aligning uncorrelated sentence representation, using feature purification on decorrelated representation enables sentence representation to better align the useful local features while staying away from the useless local features.

In conclusion, both feature decorrelation and feature purification can improve the detection ability, but if we can first remove the dependencies between features and then purify these decorrelated features, the generalization ability of the model can be improved to the level of state-of-the-art.

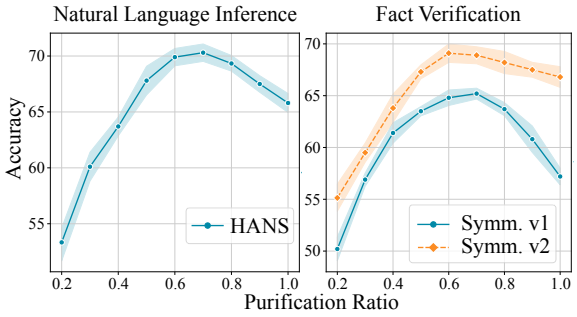

Figure 3: The results of *DePro* using different purification ratios. Meanwhile, the RFF dimensions for HANS, Symm. v1, and Symm. v2 are 4x, 2x, and 4x, respectively.

Sensitivity Analysis

In this part, we further explore the effect of the mapping dimension size of RFF and the degree of feature purification on the generalization ability of the

model. Specifically, we choose six different RFF dimensions and nine different purification ratios to commence our study. Due to the limited pages, we only show the corresponding experimental results of the best parameters in Figure 2 and Figure 3. For the NLI task, *DePro* performs the best when the RFF dimension is four times that of the original features and the top 70% of the features are used for purification. In addition, for the FEVER dataset, *DePro* can maintain the best results on Symm. v1 and Symm. v2 when the RFF dimensions are two times and four times, respectively, that of the original features and the top 70% and 60%, respectively, of the features are used for purification. Through these two figures, we see that the detection effect of *DePro* is different when choosing different RFF dimensions and different purification ratios. When the dimension is expanded to a certain number, the dependencies between features can be easily removed. At this point, when continuing to increase the dimension, it may bring additional overhead and impact, making the detection effect decrease instead.

For feature purification, if too many local features are removed, it can make the aligned sentence representation contain too little information. Moreover, if too many local features are purified, it may make the sentence representation contain too much useless information, so that the subsequent classifier cannot make predictions well based on the sentence representation.

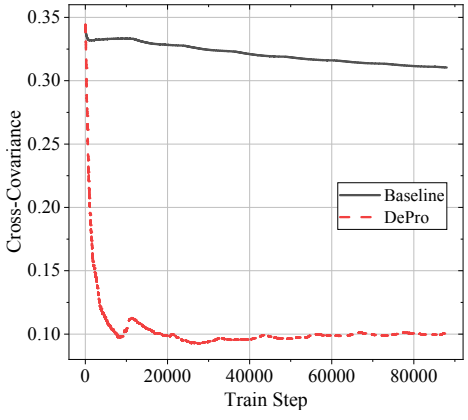

Figure 4: The mean of the correlations (*i.e.*, cross-covariance) between features at different iterations.

Decorrelation Study

Finally, we check whether feature decorrelation can remove the dependencies between features or not. Specifically, during the training phase, we record the mean of the correlations between features at different iterations. For the baseline experiment, we use the same RFF mapping functions to map the features to high-dimensional space. However, the reconstructed features are only used to calculate the cross-covariance, not to calculate the loss and optimize the parameters. Through the comparative results in Figure 4, we observe that the cross-covariance between features can be reduced as the number of iterations increases in *DePro*. However, in the baseline experiment, it barely decreases.

Overall, *DePro* can effectively remove dependencies between features. In this way, the spurious correlations can be mitigated at the feature level.

DePro	MNLI		FEVER	
	ID	HANS	ID	Symm. v2
With β-VAE	82.7	67.3	83.6	65.9
With RFF	**83.2**	**70.3**	**84.5**	**69.2**

Table 3: Evaluation results of the feature decorrelation phase leveraging Random Fourier Features (Rahimi and Recht, 2007) and β-VAE (Higgins et al., 2017) on two tasks, respectively.

4.4 Discussion

In this subsection, we primarily discuss two aspects: (1) Why we choose Random Fourier Features to decorrelate features in the feature decorrelation component; and (2) What distinguishes this work from prior works that use RFF to decorrelate features.

Many works (Rahimi and Recht, 2007; Zhang et al., 2021; Kingma and Welling, 2014) have been proposed to improve the generalization of the model by performing latent representation decorrelation learning. We compare the performance of two decorrelation methods RFF (Rahimi and Recht, 2007) and β-VAE (Higgins et al., 2017) in our model structure. The performance results are illustrated in Table 3, which shows that RFF outperforms β-VAE in our model both in ID and OOD datasets. In contrast to RFF, VAEs decorrelate the representation while compressing it, thus damaging the generalization ability. So we choose RFF to decorrelate the feature representation to obtain the uncompressed decorrelated representation, which benefits succeeding feature purification to distinguish useful from useless local features.

The distinction between *DePro* and other RFF-

based methods (Rahimi and Recht, 2007; Giffon et al., 2019; Zhang et al., 2021) is that our proposed method not only uses RFF for feature decorrelation but also combines two complementary approaches (*i.e.*, feature decorrelation and feature purification). These two methods are not mutually exclusive. In Section 4.3, we analyze the relationship between these two in detail, that is, the decorrelated features can be better purified, allowing the model to ignore more impurities when purifying useful features. Moreover, after feature decorrelation, feature purification can constrain the model to concentrate more on useful features rather than useless features.

5 Conclusion

In this paper, to improve the generalization ability of deep models on OOD datasets, we design an end-to-end framework called *DePro* which can eliminate spurious correlations and purify the decorrelated features. Extensive experiments on two well-studied NLU tasks demonstrate the synergistic effect between decorrelation and purification. After combining them, our method outperforms state-of-the-art methods in terms of effectiveness.

Acknowledgements

The authors would like to thank the anonymous reviewers for their helpful comments. We would also like to thank Feng Cheng, Haoxiang Jia, Wenxuan Li, and Yuhao Zhou for their help during the revision phase of the paper. This work was partially National Natural Science Foundation of China (No. 62076069, 61976056), Shanghai Municipal Science and Technology Major Project (No.2021SHZDZX0103).

References

Mauricio A Alvarez, Lorenzo Rosasco, Neil D Lawrence, et al. 2012. Kernels for vector-valued functions: A review. *Foundations and Trends® in Machine Learning*, 4(3):195–266.

Martin Arjovsky, Léon Bottou, Ishaan Gulrajani, and David Lopez-Paz. 2019. Invariant risk minimization. *arXiv preprint arXiv:1907.02893*.

Hyojin Bahng, Sanghyuk Chun, Sangdoo Yun, Jaegul Choo, and Seong Joon Oh. 2020. Learning de-biased representations with biased representations. In *International Conference on Machine Learning*, pages 528–539. PMLR.

Emily M Bender and Alexander Koller. 2020. Climbing towards nlu: On meaning, form, and understanding in the age of data. In *Proceedings of the 58th annual meeting of the association for computational linguistics*, pages 5185–5198.

Christopher Clark, Mark Yatskar, and Luke Zettlemoyer. 2019. Don't take the easy way out: Ensemble based methods for avoiding known dataset biases. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 4069–4082.

Christopher Clark, Mark Yatskar, and Luke Zettlemoyer. 2020. Learning to model and ignore dataset bias with mixed capacity ensembles. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 3031–3045.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*.

Luc Giffon, Stéphane Ayache, Thierry Artières, and Hachem Kadri. 2019. Deep networks with adaptive nyström approximation. In *2019 International Joint Conference on Neural Networks (IJCNN)*, pages 1–8. IEEE.

Arthur Gretton, Kenji Fukumizu, Choon Teo, Le Song, Bernhard Schölkopf, and Alex Smola. 2007. A kernel statistical test of independence. *Advances in neural information processing systems*, 20.

Suchin Gururangan, Swabha Swayamdipta, Omer Levy, Roy Schwartz, Samuel Bowman, and Noah A Smith. 2018. Annotation artifacts in natural language inference data. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 2 (Short Papers)*, pages 107–112.

Xiaochuang Han, Byron C Wallace, and Yulia Tsvetkov. 2020. Explaining black box predictions and unveiling data artifacts through influence functions. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 5553–5563.

He He, Sheng Zha, and Haohan Wang. 2019. Unlearn dataset bias in natural language inference by fitting the residual. In *Proceedings of the 2nd Workshop on Deep Learning Approaches for Low-Resource NLP (DeepLo 2019)*, pages 132–142.

Irina Higgins, Loïc Matthey, Arka Pal, Christopher P. Burgess, Xavier Glorot, Matthew M. Botvinick, Shakir Mohamed, and Alexander Lerchner. 2017. beta-vae: Learning basic visual concepts with a constrained variational framework. In *5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Conference Track Proceedings*. OpenReview.net.

Diederik P. Kingma and Max Welling. 2014. Auto-encoding variational bayes. In *2nd International Conference on Learning Representations, ICLR 2014, Banff, AB, Canada, April 14-16, 2014, Conference Track Proceedings*.

Gary Marcus. 2018. Deep learning: A critical appraisal. *arXiv preprint arXiv:1801.00631*.

Tom McCoy, Ellie Pavlick, and Tal Linzen. 2019. Right for the wrong reasons: Diagnosing syntactic heuristics in natural language inference. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 3428–3448.

Christian Naesseth, Francisco Ruiz, Scott Linderman, and David Blei. 2017. Reparameterization gradients through acceptance-rejection sampling algorithms. In *Artificial Intelligence and Statistics*, pages 489–498. PMLR.

Yixin Nie, Yicheng Wang, and Mohit Bansal. 2019. Analyzing compositionality-sensitivity of nli models. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, pages 6867–6874.

Aaron van den Oord, Yazhe Li, and Oriol Vinyals. 2018. Representation learning with contrastive predictive coding. *arXiv preprint arXiv:1807.03748*.

Ali Rahimi and Benjamin Recht. 2007. Random features for large-scale kernel machines. *Advances in neural information processing systems*, 20.

Victor Sanh, Thomas Wolf, Yonatan Belinkov, and Alexander M Rush. 2020. Learning from others' mistakes: Avoiding dataset biases without modeling them. *arXiv preprint arXiv:2012.01300*.

Tal Schuster, Darsh Shah, Yun Jie Serene Yeo, Daniel Roberto Filizzola Ortiz, Enrico Santus, and Regina Barzilay. 2019. Towards debiasing fact verification models. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 3419–3425.

Zheyan Shen, Peng Cui, Tong Zhang, and Kun Kunag. 2020. Stable learning via sample reweighting. In *Proceedings of the AAAI Conference on Artificial Intelligence*, pages 5692–5699.

James Thorne, Andreas Vlachos, Oana Cocarascu, Christos Christodoulopoulos, and Arpit Mittal. 2018. The fact extraction and verification (fever) shared task. *EMNLP 2018*, 80(29,775):1.

Prasetya Ajie Utama, Nafise Sadat Moosavi, and Iryna Gurevych. 2020a. Mind the trade-off: Debiasing nlu models without degrading the in-distribution performance. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 8717–8729.

Prasetya Ajie Utama, Nafise Sadat Moosavi, and Iryna Gurevych. 2020b. Towards debiasing nlu models from unknown biases. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 7597–7610.

Boxin Wang, Shuohang Wang, Yu Cheng, Zhe Gan, Ruoxi Jia, Bo Li, and Jingjing Liu. 2020. Infobert: Improving robustness of language models from an information theoretic perspective. In *International Conference on Learning Representations*.

Zhiquan Wen, Guanghui Xu, Mingkui Tan, Qingyao Wu, and Qi Wu. 2021. Debiased visual question answering from feature and sample perspectives. *Advances in Neural Information Processing Systems*, 34.

Adina Williams, Nikita Nangia, and Samuel Bowman. 2018. A broad-coverage challenge corpus for sentence understanding through inference. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 1112–1122.

Ruibin Xiong, Yimeng Chen, Liang Pang, Xueqi Cheng, Zhi-Ming Ma, and Yanyan Lan. 2021. Uncertainty calibration for ensemble-based debiasing methods. *Advances in Neural Information Processing Systems*, 34:13657–13669.

Xingxuan Zhang, Peng Cui, Renzhe Xu, Linjun Zhou, Yue He, and Zheyan Shen. 2021. Deep stable learning for out-of-distribution generalization. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 5372–5382.

Zhihong Zhang, Yiyang Tian, Lu Bai, Jianbing Xiahou, and Edwin Hancock. 2017. High-order covariate interacted lasso for feature selection. *Pattern Recognition Letters*, 87:139–146.

Event Causality Identification via Derivative Prompt Joint Learning

Shirong Shen[1] and **Heng Zhou**[1] and **Tongtong Wu**[1] and **Guilin Qi**[1*]
[1]School of Computer Science and Engineering, Southeast University, China
{ssr, zhouheng2020, wutong8023, gqi}@seu.edu.cn

Abstract

This paper studies event causality identification, which aims at predicting the causality relation for a pair of events in a sentence. Regarding event causality identification as a supervised classification task, most existing methods suffer from the problem of insufficient annotated data. In this paper, we propose a new derivative prompt joint learning model for event causality identification, which leverages potential causal knowledge in the pre-trained language model to tackle the data scarcity problem. Specifically, rather than external data or knowledge augmentation, we derive two relevant prompt tasks from event causality identification to enhance the model's ability to identify explicit and implicit causality. We evaluate our model on two benchmark datasets and the results show that our model has great advantages over previous methods.

1 Introduction

Event causality identification (ECI) is an important natural language processing (NLP) task, which aims at identifying causality between events in sentences. Event causality identification supports a variety of NLP applications, e.g., machine reading comprehension (Berant et al., 2014) and event prediction (Radinsky et al., 2012). (Berant et al., 2014; Radinsky et al., 2012). As shown in Figure 1, an ECI model identifies the causalities in sentences S_1 and S_2: (i) **practice** $\overset{cause}{\longrightarrow}$ **won** in S_1; (ii) **attack** $\overset{cause}{\longrightarrow}$ **killed** in S_2. The causality between events in a sentence mainly contains two types: *explicit causality* and *implicit causality*. For instance, the causality **practice** $\overset{cause}{\longrightarrow}$ **won** in S_1 is an explicit causality, which is triggered by the explicit cue words in the sentence. ECI models can take causal cue words as the shortcut for explicit causality identification. As a contrast, the causality

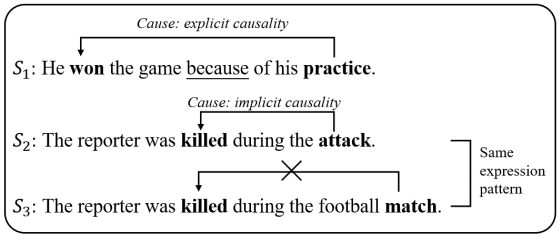

Figure 1: Examples of different causalities. S_1 contains *explicit causality* between **practice** and **won**. S_2 contains *implicit causality* between **killed** and **attack**. S_3 has the same patterns with S_2, but it does not contain any causality.

attack $\overset{cause}{\longrightarrow}$ **killed** in S_2 is an *implicit causality*, because none of explicit cue words is mentioned in S_2. And as shown in Figure 1, comparing S_2 and S_3, we may not always derive the existence of causality from two highly similar expressions without explicit cue words. In other words, *Implicit causality* must be inferred from the semantics and contexts of events.

Most existing methods regard ECI as a classification task, and train customized ECI models on annotated data by supervised learning (Cheng and Miyao, 2017; Choubey and Huang, 2017). However, the large-scale annotated datasets of ECI are relatively hard to collect, referring to that the so far largest dataset EventStoryLine (Caselli and Vossen, 2017) only contains 258 documents and 4316 sentences. Therefore, ECI models are challenged by the data scarcity problem in supervised learning. To address this problem, various methods have been proposed to leverage either the augmented dataset (Hashimoto, 2019) or external knowledge (Liu et al., 2020; Zuo et al., 2021b,a). Hashimoto (2019) exploit weakly supervised method to construct ECI datasets. Liu et al. (2020) and Zuo et al. (2021a) attempt to introduce external structure knowledge to identify causality. However, the model may fail to capture the differences between the explicit causality and

* Corresponding author.

Proceedings of the 29th International Conference on Computational Linguistics, pages 2288–2299
October 12–17, 2022.

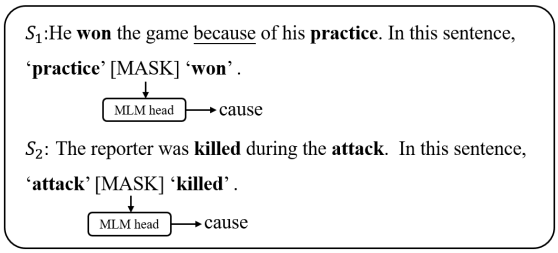

Figure 2: ECI can be converted into the form of predicting `[MASK]` by PLM.

implicit causality without modelling these different types respectively, especially when the ECI model is trained with only insufficient annotated dataset.

In this paper, we propose a new **D**rivative **P**rompt **J**oint **L**earning (**DPJL**) method for ECI, which identifies different causalities effectively without incorporating either more annotated instances or external knowledge. Firstly, as shown in Figure 2, we introduce a new prompt-based learning paradigm to ECI, i.e. converting ECI into a language modelling format and using pre-trained language model (PLM) to identity causalities. Since PLMs accumulated abundant knowledge (Jawahar et al., 2019; Yenicelik et al., 2020; Brown et al., 2020) through the self-supervised training on large-scale corpora, such a prompt-based paradigm may elicit the potential ECI ability of PLM to remedy the scarcity of annotated data. Then, we design two derivative prompt tasks for ECI to identify the explicit causality and implicit causality: (i) Causal cue Word Detection (CWD), which aims to detect the causal cue word of event pairs in a sentence; and (ii) Causal Event Detection (CED), which aims to detect the cause or effect of an event in a sentence. Intuitively, CWD is a straightforward way to identify *explicit causality*, and CED is helpful to identify *implicit causality* by capturing the semantic relevancy between the contextual events. Finally, given the above intuitions, we further propose a joint learning method for event causality identification enhanced by the tasks of CWD and CED. Note that the training data of derivative tasks are generated from ECI, without the cost of additional human annotation. Our contributions are summarized as following:

- We introduce a new prompt-based paradigm to ECI, and we propose a new derivative prompt joint learning method which remedies the problem caused by the scarcity of annotated data. As far as we know, this is the first time to use prompt-based method for ECI.
- We propose two new derivative tasks in the joint learning method, i.e., the causal cue word detection and causal event detection, to strengthen the ability of an ECI model in identifying the explicit causality and implicit causalities respectively. Note that, rather than using more human-annotation, the proposed two derivative tasks leverage the annotated instances modified from the dataset of ECI for training.
- We conduct extensive experiments on two benchmark datasets of ECI, in which our proposed method DPJL achieves the state-of-the-art performance with at least 11 percent F1 improvement on both benchmarks.

2 Related Work

Event Causality Identification Event Causality Identification (ECI) is a crucial information extraction task. Early causal identification methods include rule-based methods (Mirza et al., 2014; Riaz and Girju, 2013; Do et al., 2011) and statistics-based methods (Beamer and Girju, 2009). Recently, some benchmarks on the event causality have been released, e.g. Causal-TimeBank (Mirza and Tonelli, 2014), EventStoryLine Corpus (Caselli and Vossen, 2017) and BECAUSE (Dunietz et al., 2015). Based on these annotated datasets, a number of supervised learning-based methods of ECI have emerged (Kruengkrai et al., 2017; Kadowaki et al., 2019). However, the scale of labeled data in most datasets is relatively small. To solve this problem, Hashimoto (2019) exploited weakly supervised method to construct ECI datasets. Some methods introduce additional knowledge to strengthen the ECI model (Liu et al., 2020; Zuo et al., 2021b,a). Zuo et al. (2020) improved the performance of ECI with distantly supervised labeled training data. These methods introduce the pre-trained language model to generate the high-quality text coding required by the ECI model. But these methods ignore the potential ability of pre-training language model to identify the causality between events, and may fail to capture the differences between the explicit and implicit causalities in a low-resource scenario without modelling these two types of causalities respectively.

Prompt-based learning Recently, pre-trained language models like GPT (Radford and Narasimhan, 2018) , BERT (Devlin et al., 2019) , RoBERTa (Liu et al., 2019) and T5 (Raffel et al., 2020) can capture rich knowledge (Jawahar et al., 2019; Yenice-

lik et al., 2020) from massive unlabeled corpora. But there is a big gap between pre-training objectives and fine-tuning objectives, that is, downstream tasks still need to build task-specific models after PLMs, and use task-specific annotated data to fine-tune them. To solve this problem, prompt-based method (Brown et al., 2020) converts the downstream task into the same form as pre-training task. To better build task prompts, automatic search of discrete prompts (Gao et al., 2021), gradient-guided search (Shin et al., 2020) and continuous prompts (Li and Liang, 2021) are successively proposed. To the best of our knowledge, there is no work that uses prompt-based method for ECI task. However, only using prompt-based methods lacks the modeling of different causalities mentioned above.

3 Methodology

In this section, we first introduce problem definition of event causality identification. Then we will show the overview of our proposed model. After that, we introduce a prompt-based ECI method to elicit knowledge from PLMs. Then, we describe the details of two derivative prompt tasks and the joint learning method of derivative prompts. Finally, we introduce the training and prediction process of our model.

3.1 Problem Definition

Given $x = (S, (e_s, e_t))$ as an instance of ECI task, where S is a sentence and (e_s, e_t) is an event pair in S. $\mathcal{Y}$ is the set of causal labels indicating whether there is a causality between event pairs. For an instance x, the purpose of an ECI model is to predict the causal label $y \in \mathcal{Y}$ between e_s and e_t. The traditional approaches formulate ECI as a binary classification problem. In order to learn the feature of different causalities better, we set $\mathcal{Y} : \{Cause, Causedby, NA\}$, which respectively mean that e_s causes e_t, e_s is caused by e_t and there is no causality between (e_s, e_t). The output of our ECI model is a ternary vector corresponding to the probabilities of the three labels. In order to unify with the previous method, *Cause* or *Causedby* both indicate that there is a causality between events.

3.2 Overview

The overview of our approach is shown in Fig. 3. We convert the ECI task into a mask language prediction task, and use an excellent pre-trained masked language model (MLM) named RoBERTa (Liu et al., 2019) to encoder the input sequence and output prediction results. The reason we use MLM is that MLM can make good use of context information and we can flexibly design prompt templates for it. We design two derivative prompt tasks to make our model capture the different abilities of identifying different kinds of causalities. The prompts for all tasks are spliced after the input sentence as the input of RoBERTa. Finally, we use RoBERTa's MLM head to make predictions through a joint learning method.

3.3 Prompt-based Event Causality Identification

For a given instance $x = \{S, (e_s, e_t)\}$, the key to converting ECI task into MLM task is to construct an appropriate prompt template $\mathcal{T}_{ECI}(x)$ and determine the label words $\mathcal{V}$. $\mathcal{T}_{ECI}(x)$ spliced after the input sentence S is used to prompt the PLM to predict the causality between event pair (e_s, e_t). $\mathcal{V}$ refers to a set of words in the vocabulary of PLM that corresponds to the labels of the ECI task. A `[MASK]` token is placed into $\mathcal{T}_{ECI}(x)$ for PLM to fill the label words. There may be many kinds of templates as shown in Figure 2 for ECI, it is not sure which one is most suitable for ECI task. So we add some new learnable tokens to one template to make it dynamically adapt to the task during the model's training process. Since the words in PLM vocabulary may fail to represent the abundant semantic knowledge in causal labels, so we use three virtual words corresponding to three labels form $\mathcal{V}$ as in the previous work (Li and Liang, 2021). Finally, the prompt template and label words for ECI is formalized as follows:

$$\mathcal{T}_{ECI}(x) : In\ this\ sentence, \text{`}e_s\text{'}\ \texttt{<c>}\ \texttt{[MASK]}\ \texttt{</c>}\ \text{`}e_t\text{'}.\ \texttt{[SEP]}$$
$$\mathcal{V} : \{\texttt{Cause}, \texttt{Causedby}, \texttt{NA}\}$$

where `<c>`, `</c>` and virtual words in V are the new learnable tokens added into the vocabulary of PLM, `[SEP]` is the token indicating the end of the sentence. Each new token has an embedding with the same size as the embeddings of original word in the dictionary. Using `Cause` and `Causedby` allows the model to learn the directional features of the causality. We use an injective mapping $\mathcal{M} : \mathcal{Y} \rightarrow \mathcal{V}$ to connect causal labels to label words, each causal label is mapped to a label word with the same name. Then we expand

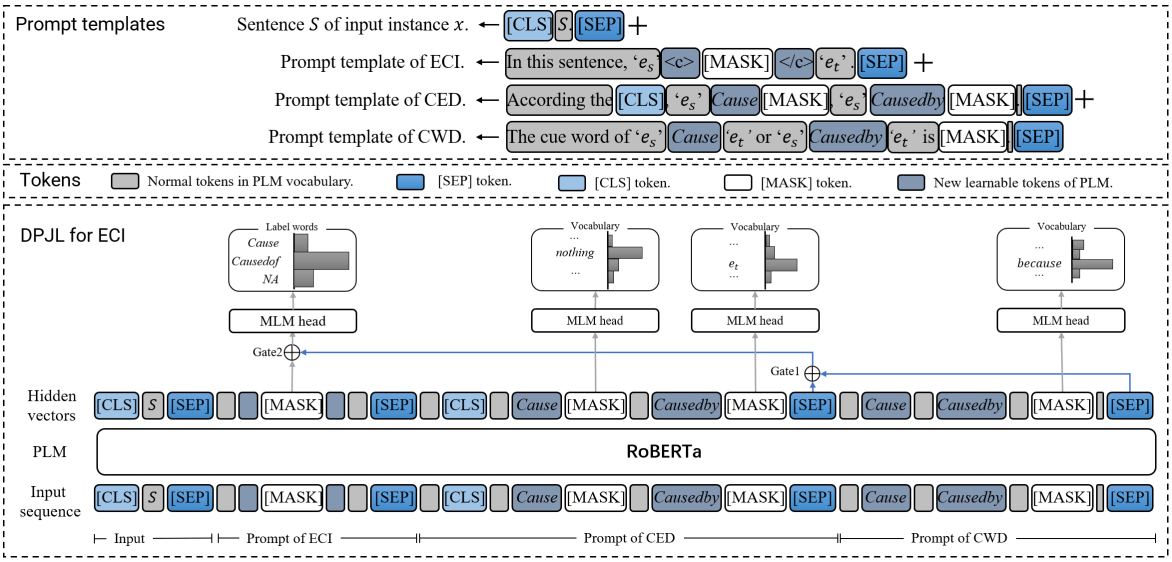

Figure 3: Overview of our derivative prompt joint learning (DPJL). The dashed box above shows the prompt template for each task. The dashed box in the middle shows the correspondence between different token and color blocks. The dashed box below shows the overall framework of DPJL.

the MLM head layer of PLM with the $\mathcal{V}$ and use the probability distribution over $\mathcal{V}$ at the `[MASK]` position as the probabilities of causal labels. Formally, for an instance x, we first splice `[CLS]` and `[SEP]` on both sides of S to fit the input form of RoBERTa. The probability of its causal label $p(y|x)$ is:

$$p_{MLM\_head}(\texttt{[MASK]} = \mathcal{M}(y|S' + \mathcal{T}_{ECI}(x)) \quad (1)$$

where p_{MLM_head} represents the probability distribution predicted by MLM head layer, '+' means sequence splicing, $S' =$ `[CLS]` S `[SEP]`.

3.4 Derivative Prompt Tasks

We design two derivative prompt tasks for ECI to elicit the abilities to identify explicit causality and implicit causality from PLM. For explicit causality, ECI model need the ability to detect causal cue words of given event pair. For implicit causality, ECI model need the ability to comprehensively analyze event semantics and context. So we design the following two derivative tasks, that is: (i) **Causal cue word detection (CWD)**: Given an instance $x = (S, (e_s, e_t))$, CWD aims to detect the cue word in S which triggers the causality between (e_s, e_t); (ii) **Causal event detection (CED)**: Given an instance $x = (S, e_s)$ where e_s is an event within sentence S, CED aims to detect the event in S which has a causality with e_s.

To elicit the corresponding abilities from PLM, we also set prompts for the two derivative prompt tasks as follows:

$\mathcal{T}_{CWD}(x)$: *The cue word of* 'e_s' `Cause` 'e_t' *or* 'e_s'`Causedby` 'e_t' *is* `[MASK].` `[SEP]`

$\mathcal{T}_{CED}(x)$: *According the* `[CLS]`, 'e_s' `Cause` `[MASK]`, 'e_s' `Causedby` `[MASK].` `[SEP]`

The targets of the two tasks are the specific words in the input sentence. If a `[MASK]` has no corresponding answer in the S, its target word is *nothing*, and if the answer word is longer than one word, the target is the first token of answer. The training data of derivative tasks are generated from the original dataset annotated for ECI, please refer to Section 4.2 for more details. We can splice the prompts of derivative task behind the input sentence and use the MLM head layer to predict the probability distribution over PLM's vocabulary at different masked positions as the results of these tasks. Because of the huge vocabulary space, it is possible for PLM to generate words that are not included in sentences. So that, we constrain the candidate vocabulary to the $\mathcal{V}_S$ by setting logits of extraneous words to -inf, where $\mathcal{V}_S$ contains *nothing* and the tokens in S.

3.5 Joint Learning

In this section, we will introduce the joint learning method for derivative tasks and ECI. Firstly,

as shown in Figure 3, we reuse the label words `Cause` and `Causedby` in the prompts of derivative tasks and we concatenate all prompts after S' as the input of RoBERTa (i.e. $S' + \mathcal{T}_{ECI}(x) + \mathcal{T}_{CED}(x) + \mathcal{T}_{CWD}(x)$), all tasks will be predicted and trained simultaneously. On the one hand, the three tasks can share semantics with each other by PLM. On the other hand, learnable label word embedding can be trained during the training of the derivative tasks. Secondly, we set up two gate units as the highway for building the association ECI and derivative tasks. Although the language model can share contextual information, due to the large parameter scale of it, the annotated samples of ECI may not be enough to build the association between ECI and derivative tasks. Specifically, we use the `[SEP]` tokens in $\mathcal{T}_{CED}(x)$ and $\mathcal{T}_{CED}(x)$ to represent the overall semantics of the two derivative prompts. Then we use a gate unit to integrate the semantic information of CWD and CED, and tune the hidden features of the `[MASK]` corresponding to ECI through another gate unit as follows.

$$\begin{aligned} g_1 &= \sigma(W_g^1[\mathbf{h}_{[SEP]}^{CWD}; \mathbf{h}_{[SEP]}^{CED}) \\ \widetilde{\mathbf{h}}_{[SEP]}^{CED} &= (1-g_1)\mathbf{h}_{[SEP]}^{CWD} + g_1\mathbf{h}_{[SEP]}^{CED} \\ g_2 &= \sigma(W_g^2[\mathbf{h}_{[MASK]}^{ECI}; \widetilde{\mathbf{h}}_{[SEP]}^{CED}]) \\ \widetilde{\mathbf{h}}_{[MASK]}^{ECI} &= (1-g_1)\mathbf{h}_{[MASK]}^{ECI} + g_1\widetilde{\mathbf{h}}_{[SEP]}^{CED} \end{aligned} \tag{2}$$

where $\mathbf{h}_{[SEP]}^{CWD}$ and $\mathbf{h}_{[SEP]}^{CED}$ means the hidden features of `[SEP]` in CWD prompts and CED prompts, $\mathbf{h}_{[MASK]}^{ECI}$ is the hidden feature of the `[MASK]` corresponding to ECI, each hidden feature is generated by RoBERTa. W_g^1 and W_g^2 are trainable parameters for gate units, σ is the sigmoid activation function maps variables between $(0, 1)$. $\widetilde{\mathbf{h}}_{[MASK]}^{ECI}$ will replace $\mathbf{h}_{[MASK]}^{ECI}$ as input to MLM head and predict the probability of label words via equation (1). This method effectively builds the link between ECI and derivative tasks without affecting the encoding process of PLM.

3.6 Training and Prediction

We perform supervised training on three tasks simultaneously. We use the cross entropy function to calculate the losses of all tasks, multiply the losses of derived tasks by 0.1, and add them to the losses of ECI as the objective function. In CED task, given a sample may have multiple golden answers, we calculate the averaged cross-entropy loss for each predicted answer. In addition, for each input instance, we have a 10% probability of filling the corresponding `[MASK]` positions with the correct answers of the derivative tasks to speed up the training process. In prediction stage, the target positions in the derivative tasks are all `[MASK]` tokens, and we only predict the probability distribution of the label words of the ECI task by Eq (1).

4 Experiments

Our experiments aim to verify (1) whether the prompt-based method can improve the generalization of the ECI model, and (2) whether the joint learning of derivative prompts can enhance the model's ability to identify different causalities. Our source code is available on Github[1]

4.1 Dataset and Metrics

We perform our method on two main benchmarks, including: EventStoryLine v0.9 (ESC) (Caselli and Vossen, 2017), which contains 258 documents, 4316 sentences, and 1770 causal event pairs; Causal-TimeBank (CTB) (Mirza and Tonelli, 2014) which contains 184 documents, 6813 events, and 318 causal event pairs. Same as previous methods (Gao et al., 2019; Zuo et al., 2021b,a), we use the last two topics of ESC as the development set for two datasets, and conduct 5-fold and 10-fold cross-validation on ESC and CTB respectively. For evaluation, we adopt Precision (P), Recall (R), and F1-score (F1) as evaluation metrics. All the results are the average of three independent experiments.

4.2 Experimental Settings

Training Details. In implementations, we use the RoBERTa[2] with an open pre-trained parameters[3] for our method, which has 12-layers, 768-hiddens, and 12-heads. Each of new tokens added in RoBERTa have 768-dimensional embedding. The size of W_g^1 and W_g^2 are 1536×1. We set the learning-rate of pre-trained parameters and new parameters as 1e-5 and 1e-4 respectively. We use the `causal signal` given in the annotated datasets of ECI as the cue word to construct the training data of CWD, and use the causal event pairs in the annotated datasets of ECI to construct the training data of CED. We adopt a negative sampling rate of 0.5 for training our model, and the batch size for training is 16. And we apply the early stop and AdamW gradient strategy to optimize all models.

[1] If the paper is accepted, a link to the code repository will be published.

[2] https://pytorch.org/hub/pytorch_fairseq_roberta/

[3] https://huggingface.co/roberta-base/tree/main

Compared Methods. We compare our methods with previous state-of-the-art works. For ESC, we prefer the following methods: **LSTM** (Cheng and Miyao, 2017), a dependency path based sequential model. **Seq** (Choubey and Huang, 2017), a sequence model with human designed feaures. **LR+** and **ILP** (Gao et al., 2019), ECI models adopt document structure. For CTB, we prefer the following methods: **RB** (Mirza and Tonelli, 2014), a rule-based system for ECI. **DD** (Mirza and Tonelli, 2014), a data driven machine learning based system. **VR-C** (Mirza, 2014), a verb rule based model with data filtering and gold causal signals enhancement.

In addition, we also compare SOTA methods based on pre-trained language models and introducing external knowledge: **MM** (Liu et al., 2020), the BERT-based SOTA method with mention masking generalization. **KnowDis** (Zuo et al., 2020), a distantly supervised method for ECI. **LearnDA** (Zuo et al., 2021b), a learnable knowledge-guided data augmentation method for ECI. **CauSeRL** (Zuo et al., 2021a), a self-supervised representation learning enhanced ECI method. For a fair comparison, we set up two baseline models based on RoBERTa: **RoBERTa-base**, a RoBERTa-base baseline, we use a linear classifier after RoBERTa for ECI, the input of the classifier is the hidden feature of target events. **Prompt-base**, a prompt-based baseline, our basic proposed ECI method mentioned in Section 3.3.

4.3 Main Results

Table 1 and Table 2 show the experimental results on ESC and CTB respectively. From these results:

Methods	P	R	F1
LSTM(Cheng and Miyao, 2017)	34.0	41.5	37.4
Seq(Choubey and Huang, 2017)	32.7	44.9	37.8
LR+(Gao et al., 2019)	37.0	45.2	40.7
ILP(Gao et al., 2019)	37.4	55.8	44.7
MM(Liu et al., 2020)	41.9	62.5	50.1
KnowDis(Zuo et al., 2020)	39.7	66.5	49.7
LearnDA(Zuo et al., 2021b)	42.2	69.8	52.6
CauSeRL(Zuo et al., 2021a)	41.9	69.0	52.1
RoBERTa-base(**ours**)	40.8	64.7	50.0*
Prompt-base(**ours**)	53.6	68.3	60.0*
DPJL(**ours**)	65.3	70.8	**67.9***

Table 1: Main results on ESC. * denotes a significant test at the level of 0.05.

(1) Our DPJL method outperforms all other ECI methods, and achieves the best F1 on both datasets, 67.9% on ESC and 64.6% on CTB respectively.

Methods	P	R	F1
RB(Mirza and Tonelli, 2014)	36.8	12.3	18.4
DD(Mirza and Tonelli, 2014)	67.3	22.6	33.9
VR-C(Mirza, 2014)	69.0	31.5	43.2
MM(Liu et al., 2020)	36.6	55.6	44.1
KnowDis(Zuo et al., 2020)	42.3	60.5	49.8
LearnDA(Zuo et al., 2021b)	41.9	68.0	51.9
CauSeRL(Zuo et al., 2021a)	43.6	68.1	53.2
RoBERTa-base(**ours**)	40.3	58.2	47.6*
Prompt-base(**ours**)	49.7	69.4	57.9*
DPJL(**ours**)	63.6	66.7	**64.6***

Table 2: Main results on CTB. * denotes a significant test at the level of 0.05.

Specifically, DPJL outperforms the previous SOTA method by at least 10 percentage points. It illustrated that prompt-based methods with derivative prompts joint learning can effectively elicit causal knowledge in PLMs, thus greatly improve the performance of the ECI model.

(2) The experimental results of KnowDis, LearnDA, and CauSeRL show that the introduction of different external knowledge and the method of introducing external knowledge can affect the performance of the ECI model. We note that the performance of Prompt-base and DPJL is higher than that of other knowledge-enhanced methods. It shows that eliciting causal knowledge from PLMs is more beneficial to ECI than the previous approach of introducing external knowledge. The reason may be that previous methods do not fill the gap between external knowledge and true causal representation well, and prompt-based method can directly convert the underlying causal knowledge of PLMs into the ability of causal identification.

(3) RoBERTa-base outperforms the methods without RoBERTa, which illustrates the superiority of RoBERTa. RoBERTa-base is not as good as LearnDA and CauSeRL, which illustrates that simply fine-tuning PLMs cannot completely cover the knowledge required for ECI. Prompt-base and DPJL outperform RoBERTa-base, indicating that prompt-based methods can elicit the potential of PLM to solve ECI better than fine-tuning.

(4) Comparing DPJL and Prompt-base, we notice that DPJL is significantly better than Prompt-base. It is illustrated that joint derivative prompts can elicit more useful knowledge from PLMs for ECI. In addition, the improvement of DPJL of ESC is more obvious than that of CTB. The reason is that ESC has more labeled data, which is beneficial

for training the label word embeddings in derivative prompts joint learning.

4.4 Ablation Experiment

To illustrate the effect of label words reuse and gate units in DPJL, we set up ablation experiments. Different experimental settings are indicated with subscripts, where $Full$ represents the full method of DPJL, $w/o.$ and $w/.$ in subscript mean *with* and *without* respectively, $reuse$ and $gate$ mean label words reuse and gate units of this paper. The ablation results are shown in Table 3 and Table 4. In addition, to verify the generalizability of each derivative prompt task in our method, we adopt two more additional datasets for further ablation studies, i.e., EventStoryLine v1.5 (ESC v1.5) (Caselli and Inel, 2018) [4] and BECAUSE (Dunietz et al., 2017) [5]. The specific experimental results are shown in the Appendix A.

Methods	P	R	F1	Δ
Prompt-base	53.6	68.3	60.0*	-
DPJL$_{w/o.reuse-w/o.gate}$	55.5	68.9	61.4*	+1.4
DPJL$_{w/.reuse-w/o.gate}$	59.9	69.3	64.5*	+4.5
DPJL$_{w/o.reuse-w/.gate}$	62.2	68.8	65.3*	+5.3
DPJL$_{Full}$	65.3	70.8	**67.9***	**+7.9**

Table 3: Ablation results on ESC. * denotes a significant test at the level of 0.05. **Δ** means the points higher than Prompt-base.

Methods	P	R	F1	Δ
Prompt-base	49.7	69.4	57.9*	-
DPJL$_{w.o./reuse-w/o.gate}$	50.2	70.4	58.6*	+0.7
DPJL$_{w./reuse-w/o.gate}$	52.1	71.2	60.1*	+2.2
DPJL$_{w.o./reuse-w/.gate}$	62.5	63.6	63.0*	+5.1
DPJL$_{Full}$	63.6	66.7	**64.6***	**+6.7**

Table 4: Ablation results on CTB. * denotes a significant test at the level of 0.05. **Δ** means the points higher than Prompt-base.

Effect of Derivative Prompts. Comparing Prompt-base with DPJL$_{w/o.reuse-w/o.gate}$, despite simply splicing the derivative prompt after the sentence, the model performance has improved. It illustrates that the derivative prompts contain the knowledge for identifying causality and this knowledge can assist Prompt-based ECI through context. In the same way, all the methods using derivative prompts are better than Prompt-base, indicating that adding derivative prompts is an effective method to improve prompt-based ECI model.

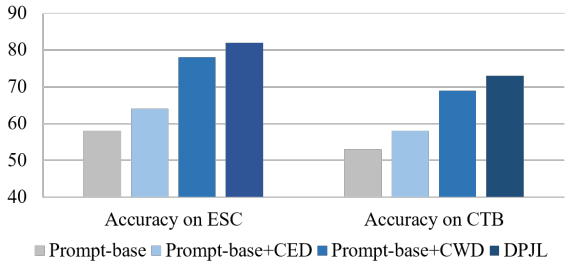

Figure 4: The accuracy of explicit set.

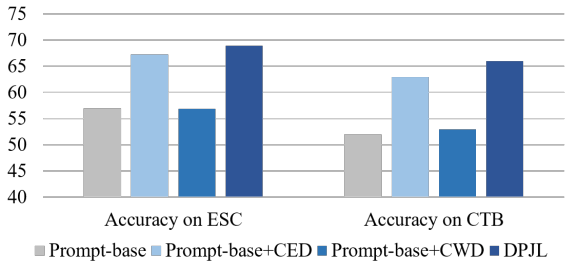

Figure 5: The accuracy of implicit set.

Effect of Gate Units. On both datasets, the methods with gate units outperform the methods without gate units whether the label words are reused in derivative prompts or not. This proves that gate units can provide a highway linking the ECI model and derivative prompts, which can better utilize the information in derivative prompts to improve the performance of causality identification in the case of a small number of training samples.

Effect of Label Word Reuse. On both ESC and CTB, the methods with label words reuse outperform the methods without label words reuse whether with or without gate units. It illustrates that reuse label words in derivative prompts can use the encoding process of the RoBERTa to strengthen the learning of the label word embeddings, and at the same time, the label words can obtain the semantic features of implicit and explicit causalities through derivative prompts. It is worth noting that the effect of label word reuse on CTB is not as obvious as that on ESC, which may be due to the fact that there are fewer training samples in CTB which cannot fully train the word embeddings of label words.

[4] http://github.com/tommasoc80/EventStoryLine
[5] http://github.com/duncanka/BECAUSE

Samples		Characteristics	Prompt-base	& CED	& CWD	DPJL
1	Iraq said it **invaded** Kuwait because of **disputes** over oil and money.	Simple causality	✓	✓	✓	✓
2	Fans and family **mourn** her **passing** , but Williams had a long , full life	Implicit causality	×	✓	×	✓
3	The general **strike** was staged as a **protest** against a new round of draconian austerity measures.	New cue word/Semantic association	×	✓	✓	✓
4	Mr. Potach notes older , more traditional groups like the Ku Klux Klan are also **opening** new chapters , thanks in part to their ability to **use** new technologies like the Internet .	New cue word	×	×	✓	✓

Figure 6: Case study. Bold words are target events, and underlined words indicate causal cue words. & CED and & CWD represent CED and CWD used in training, respectively.

4.5 Effect of Derivative Tasks on Implicit and Explicit Causalities

To illustrate the effect of derivative tasks on implicit and explicit causalities respectively, we divide the test data into implicit set and explicit set according to whether the test data contains causal cue words. The samples in implicit set contain implicit causalities and the samples in explicit set contain explicit causalities. Then we limit the kinds of derivative prompts in the model, where Prompt-base+CWD means we only use the prompts of CWD, Prompt-base+CED represents our method only using the prompts of CED. All methods with derivative prompts are consistent with DPJL except for the different types of derived tasks. We report the accuracy of these methods on both test sets.

As shown in Figure 4, joint CWD can significantly improve the accuracy of the model on explicit set, which proves that CWD can effectively elicit the ability of PLM to detect causal cue words, thereby improving the performance of the model in identifying explicit causalities. A similar phenomenon also occurs in Figure 5, the CED can effectively improve the performance of prompt-based ECI model on implicit set. It can also be found that, joint CED also helps to identify explicit causality to a certain extent, because CED not only enhances the understanding of event semantics, but also enhances the understanding of underlying causal expressions in context.

4.6 Case Study

In order to visually demonstrate the effectiveness of each of our derivative prompt joint learning method and the effect of each derivative task, we conducted a case study. As shown in Figure 6, case 1 is a simple sample of causality with a causal cue word. All methods can correctly identify the causality between target event pair. Case 2 and case 3 show that the CED task can effectively elicit the causal semantic knowledge of events in PLM, thus improving the effect of ECI between related events. However, in case 4, there is no strong semantic relationship between **opening** and **use**, and the causal cue words thanks did not appear in the training set, so the method only using CED can't correctly identify the causality between the event pairs. Case 3 and case 4 shows that CWD can elicit PLM's ability to identify causal cues, and then make the model show good generalization ability when new cue words appear. However, the method only with CWD can't correctly identify the causality in case 2, which shows that only using CWD cannot extract implicit causality well. DPJL can correctly extract all causality, which shows that our proposed method can strengthen the effect of ECI model extraction to identify explicit causality and implicit causality at same time by joint two derivative cue learning tasks.

Finally, the experiments verify that (1) the prompt-based method can effectively improve the generalization ability of the ECI model by eliciting causal knowledge in PLMs, and (2) the joint learning of derivative prompts can strengthen the model's ability to identify different causalities.

5 Conclusion and Future Work

In this paper, we first introduced a new prompt-based paradigm to event causality identification and proposed a new derivative prompt joint learning method, i.e, DPJL. The proposed method adopts two new derivative tasks, i.e., the causal cue word detection and causal event detection, to strengthen the ability of an ECI model in identifying the explicit causality and implicit causality respectively. The experimental results demonstrate that the proposed method achieves the state-of-the-art performance with at least 11 percent F1 improvement on both of two well-known benchmarks, i.e., EventStoryLine and Causal-TimeBank. Additionally, the

detailed analysis suggests the effectiveness of joint-learned prompt-based derivative tasks on performance improvement in downstream tasks. In the future, we will try knowledge-enhanced methods to construct both of derivative tasks and data for an ECI model, which may fill the gap between knowledge and samples.

6 Acknowledgement

Research in this paper was partially supported by the Natural Science Foundation of China (U21A20488).

References

Brandon Beamer and Roxana Girju. 2009. Using a bigram event model to predict causal potential. In *CICLing*.

Jonathan Berant, Vivek Srikumar, Pei-Chun Chen, Abby Vander Linden, Brittany Harding, Brad Huang, Peter Clark, and Christopher D. Manning. 2014. Modeling biological processes for reading comprehension. In *EMNLP*.

Tom B. Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, T. J. Henighan, Rewon Child, Aditya Ramesh, Daniel M. Ziegler, Jeff Wu, Clemens Winter, Christopher Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. 2020. Language models are few-shot learners. *ArXiv*, abs/2005.14165.

Tommaso Caselli and Oana Inel. 2018. Crowdsourcing storylines: Harnessing the crowd for causal relation annotation. In *EventStory@Coling*.

Tommaso Caselli and P. Vossen. 2017. The event storyline corpus: A new benchmark for causal and temporal relation extraction. In *NEWS@ACL*.

Fei Cheng and Yusuke Miyao. 2017. Classifying temporal relations by bidirectional lstm over dependency paths. In *ACL*.

Prafulla Kumar Choubey and Ruihong Huang. 2017. A sequential model for classifying temporal relations between intra-sentence events. In *EMNLP*.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. Bert: Pre-training of deep bidirectional transformers for language understanding. *ArXiv*, abs/1810.04805.

Quang Xuan Do, Yee Seng Chan, and Dan Roth. 2011. Minimally supervised event causality identification. In *EMNLP*.

Jesse Dunietz, Lori S. Levin, and Jaime G. Carbonell. 2015. Annotating causal language using corpus lexicography of constructions. In *LAW@NAACL-HLT*.

Jesse Dunietz, Lori S. Levin, and Jaime G. Carbonell. 2017. The because corpus 2.0: Annotating causality and overlapping relations. In *LAW@ACL*.

Lei Gao, Prafulla Kumar Choubey, and Ruihong Huang. 2019. Modeling document-level causal structures for event causal relation identification. In *NAACL*.

Tianyu Gao, Adam Fisch, and Danqi Chen. 2021. Making pre-trained language models better few-shot learners. *ArXiv*, abs/2012.15723.

Chikara Hashimoto. 2019. Weakly supervised multilingual causality extraction from wikipedia. In *EMNLP*.

Ganesh Jawahar, Benoît Sagot, and Djamé Seddah. 2019. What does bert learn about the structure of language? In *ACL*.

Kazuma Kadowaki, Ryu Iida, Kentaro Torisawa, Jong-Hoon Oh, and Julien Kloetzer. 2019. Event causality recognition exploiting multiple annotators' judgments and background knowledge. In *EMNLP*.

Canasai Kruengkrai, Kentaro Torisawa, Chikara Hashimoto, Julien Kloetzer, Jong-Hoon Oh, and Masahiro Tanaka. 2017. Improving event causality recognition with multiple background knowledge sources using multi-column convolutional neural networks. In *AAAI*.

Xiang Lisa Li and Percy Liang. 2021. Prefix-tuning: Optimizing continuous prompts for generation. *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, abs/2101.00190.

Jian Liu, Jian Liu, Yubo Chen, and Jun Zhao. 2020. Knowledge enhanced event causality identification with mention masking generalizations. In *IJCAI*.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Roberta: A robustly optimized bert pretraining approach. *ArXiv*, abs/1907.11692.

Paramita Mirza. 2014. Extracting temporal and causal relations between events. *ArXiv*, abs/1604.08120.

Paramita Mirza, R. Sprugnoli, Sara Tonelli, and Manuela Speranza. 2014. Annotating causality in the tempeval-3 corpus. In *EACL 2014*.

Paramita Mirza and Sara Tonelli. 2014. An analysis of causality between events and its relation to temporal information. In *COLING*.

Alec Radford and Karthik Narasimhan. 2018. Improving language understanding by generative pre-training.

Kira Radinsky, Sagie Davidovich, and Shaul Markovitch. 2012. Learning causality for news events prediction. In *WWW*.

Colin Raffel, Noam M. Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J. Liu. 2020. Exploring the limits of transfer learning with a unified text-to-text transformer. *ArXiv*, abs/1910.10683.

Mehwish Riaz and Roxana Girju. 2013. Toward a better understanding of causality between verbal events: Extraction and analysis of the causal power of verb-verb associations. In *SIGDIAL Conference*.

Taylor Shin, Yasaman Razeghi, Robert L Logan IV, Eric Wallace, and Sameer Singh. 2020. Eliciting knowledge from language models using automatically generated prompts. *ArXiv*, abs/2010.15980.

David Yenicelik, Florian Schmidt, and Yannic Kilcher. 2020. How does bert capture semantics? a closer look at polysemous words. In *BLACKBOXNLP*.

Xinyu Zuo, Pengfei Cao, Yubo Chen, Kang Liu, Jun Zhao, Weihua Peng, and Yuguang Chen. 2021a. Improving event causality identification via self-supervised representation learning on external causal statement. *ArXiv*, abs/2106.01654.

Xinyu Zuo, Pengfei Cao, Yubo Chen, Kang Liu, Jun Zhao, Weihua Peng, and Yuguang Chen. 2021b. Learnda: Learnable knowledge-guided data augmentation for event causality identification. In *ACL/IJCNLP*.

Xinyu Zuo, Yubo Chen, Kang Liu, and Jun Zhao. 2020. Knowdis: Knowledge enhanced data augmentation for event causality detection via distant supervision. In *COLING*.

A Appendix

We set up external ablation experiments to test the effect of different derivative tasks on EventStoryLine v1.5 (ESC v1.5) (Caselli and Inel, 2018) and BECAUSE (Dunietz et al., 2017). ESC v1.5 is an updated version of ESC v0.9, which contains 1,204 sentences and 7,778 event pairs, covering 22 news topics. The corpus is annotated by experts and crowd (Caselli and Inel, 2018). BECAUSE contains a total of 5380 sentences, and 1803 causal event pairs. The experimental setting is the same as that in Section 4.

We compare our proposed DPJL with the following methods: **RoBERTa-base**, a RoBERTa-base baseline which uses a linear classifier after RoBERTa for ECI. The input of the classifier is the hidden feature of target events; **Prompt-base**, a prompt-based baseline, which is our basic proposed ECI method mentioned in section 3.3; **Prompt-base+CED**, the method of joint learning with prompt-base method and CED; **Prompt-base+CWD**, the method of joint learning with prompt-base method and CWD. To show the real effect of CWD and CED, we use the label words reuse and gate units of this paper in **Prompt-base+CED** and **Prompt-base+CWD**. The experimental results on ESC v1.5 and BECAUSE are shown in Table 5 and Table 6.

Methods	P	R	F1	Δ
RoBERTa-base	53.6	64.3	59.3*	-
Prompt-base	64.0	64.6	64.3*	+5
Prompt-base+CWD	66.5	67.9	67.2*	+7.9
Prompt-base+CED	64.4	**70.4**	67.3*	+8.0
DPJL$_{Full}$	**76.9**	67.5	**71.9***	**+12.6**

Table 5: Experimental results on ESC v1.5. * denotes a significant test at the level of 0.05. **Δ** means the points higher than RoBERTa-base.

Methods	P	R	F1	Δ
RoBERTa-base	50.0	52.6	51.3*	-
Prompt-base	53.7	52.9	53.3*	+2
Prompt-base+CWD	52.9	**56.2**	54.5*	+3.2
Prompt-base+CED	**61.5**	52.6	56.7 *	+5.4
DPJL$_{Full}$	58.8	55.6	**57.1***	**+5.8**

Table 6: Experimental results on BECAUSE. * denotes a significant test at the level of 0.05. **Δ** means the points higher than RoBERTa-base.

DPJL achieves the highest F1 score on both datasets, which demonstrates the consistent effectiveness of our method. Prompt-based is superior to RoBERTa-base, which shows that prompt learning can better elicit causal knowledge in PLM than simple fine-tunig method in ECI task. The performance of Prompt-base+CWD and Prompt-base+CED is better than that of Prompt-based, which shows that joint both derivative tasks improve the model's ability to elicit the ability of ECI from PLM. Prompt-base+CED outperforms Prompt-base+CWD on BECAUSE, this may be because BECAUSE pays more attention to evaluating the ability of the model to identify implicit causality, while CED can help the ECI model to enhance the ability to identify the causal semantic association between events in PLM by predicting causal events. DPJL combines two kinds of derivative tasks, and the performance exceeds that of using only one kind of derivative task, which shows that both derivative tasks are meaningful to ECI and can complement each other.

The experimental results on these two datasets are consistent with the experimental results in the main paper. The results show that our method can adapt to more datasets, and further verifies the effectiveness of DPJL on ECI task.

Event Causality Extraction with Event Argument Correlations

Shiyao Cui[1,2] **Jiawei Sheng**[1,2] **Xin Cong**[1,2]
QuanGang Li[1,2*] **Tingwen Liu**[1,2] **Jinqiao Shi**[3,1]
[1]Institute of Information Engineering, Chinese Academy of Sciences. Beijing, China
[2]School of Cyber Security, University of Chinese Academy of Sciences. Beijing, China
[3]Beijing University of Posts and Telecommunications. Beijing, China
{cuishiyao, shengjiawei, congxin, liquangang}@iie.ac.cn
liutingwen@iie.ac.cn shijinqiao@bupt.edu.cn

Abstract

Event Causality Identification (ECI), which aims to detect whether a causality relation exists between two given textual events, is an important task for event causality understanding. However, the ECI task ignores crucial event structure and cause-effect causality component information, making it struggle for downstream applications. In this paper, we explore a novel task, namely Event Causality Extraction (ECE), aiming to extract the cause-effect event causality pairs with their structured event information from plain texts. The ECE task is more challenging since each event can contain multiple event arguments, posing fine-grained correlations between events to decide the cause-effect event pair. Hence, we propose a method with a dual grid tagging scheme to capture the intra- and inter-event argument correlations for ECE. Further, we devise a event type-enhanced model architecture to realize the dual grid tagging scheme. Experiments demonstrate the effectiveness of our method, and extensive analyses point out several future directions for ECE.

1 Introduction

Event causality (Liu et al., 2020; Cao et al., 2021) denotes an explicit causal relation between two events, constituting a specific cause-effect event pair. As shown in Figure 1, a causal relation exists between a `Price Rise` event (*The worldwide rise of oil prices*) and a `Cost Rise` event (*increases the cost of international shipping industry*). Understanding such event causality could facilitate various downstream applications including event forecasting (Hashimoto et al., 2014), intelligent search (Rudnik et al., 2019) and question answering (Costa et al., 2020), which is important for natural language understanding.

In recent years, it has aroused the research interest for Event Causality Identification (ECI) (Liu et al., 2020; Cao et al., 2021; Zuo et al., 2021a,b, 2020; Tran Phu and Nguyen, 2021), which aims to detect whether the causality exists between two given events. Despite of its success, there exist two issues that the ECI task fails to address. 1) **Event Structure Missing**, where each event in ECI is only expressed using a word or phrase which reflects its occurrence, but ignores the explicit event type and event arguments (i.e., entities which participate in the event). The absence of such event structure would lose valuable clues for understanding event causality. As shown in Figure 1, "oil" plays a `Product` role in a `Price Rise`-typed cause event, implying a consequent `Cost Rise`-typed effect event towards "shipping industry". 2) **Causality Component Missing**, where ECI only predicts the existence of causality between the given event pairs, ignoring to discriminate the specific cause/effect event causality component. Limited by these issues, ECI insufficiently explores the causality between events, which demands further promotion to the understanding of event causality.

The worldwide rise of oil prices increases the cost of international shipping industry and stimulates the demand for new energy such as Ammonia fuel.

No.	Causality Component	Event Type	Event Roles		
			Product	Region	Industry
(1)	Cause	Price Rise	oil	worldwide	None
	Effect	Cost Rise	None	international	shipping industry
(2)	Cause	Price Rise	oil	worldwide	None
	Effect	Demand Rise	new energy	None	Ammonia fuel

Figure 1: Illustration for ECE which takes the raw text as input, and outputs the structured event causality pair.

Motivated by discussion about event causality in CCKS (2021), we therefore formulate a task termed as **Event Causality Extraction (ECE)**. As Figure 1 shows, ECE aims to end-to-end extract the cause-effect event pairs with structured event information from plain texts. Comparing with ECI, ECE illustrates the event causality including the event structure, namely event types and arguments, and the specific cause-effect causality component,

*Corresponding Author

Proceedings of the 29th International Conference on Computational Linguistics, pages 2300–2312
October 12–17, 2022.

making it more informative to support the various downstream applications (Wang et al., 2021a).

Intuitively, ECE could be achieved by successively extracting the structured event and then classifying their causality relation. Unfortunately, such a paradigm would easily suffer from the redundant event-pair problem, where the causality-unrelated events would be inevitably extracted, confusing the causality decision. Another promising direction is to borrow ideas from relational triple extraction (RTE), which shares the similar task formulation. However, comparing with the entity-centric RTE task, the event-centric ECE raises new challenges: 1) **Intra-event Argument Correlations**. Specifically, ECE focuses on event, which is a structure maintaining interactive correlations among its arguments. For example in Figure 1, the argument "new energy" and "Ammonia fuel" in the `Demand Rise` event have strong semantic correlation. While RTE focuses on individual entity, thus simply adapting RTE models cannot capture such correlations to derive the event structure. 2) **Inter-event Argument Correlations**. Concretely, the event arguments involved in a cause/effect event pair usually show semantic correlations for causality deduction. As shown in Figure 1, event `Pricing_Rise` which occurs in "worldwide" `Region` could imply event `Cost_Rise` in "international" `Region`. It demonstrates that the inter-event argument correlations not only provide important clues to decide causality, but also benefit reliable cause/effect event extraction with mutual confirmation between the cause-effect pair.

In this paper, we propose an effective approach named **DualCor**, which explores both the intra-event and inter-event argument Correlations with a dual grid tagging scheme for ECE. Specifically, DualCor contains two grid tagging tables regarding event types and the input sentence, to respectively derive the event structures for cause and effect events. In each table, DualCor extracts structured event arguments according to different event types, naturally considering intra-event argument correlations. Further, when predicting the event arguments in the cause/effect table, DualCor also predicts their corresponding effect/cause event arguments, serving as auxiliary arguments to promote inter-event argument correlations. By confirming the auxiliary arguments in the other table, DualCor matches reliable cause-effect event pairs as predictions. To realize the above dual grid tagging scheme, we further devise a type-aware encoder, which refines textual representations with essential event type information to enhance argument prediction. We conduct the dual grid tagging on the type-aware textual representations to derive the final cause-effect event pair. Overall, our main contributions include:

(1) To promote the understanding to event causality, we formulate a new task named Event Causality Extraction (ECE), which succeeds ECI to push forward the research of event causality understanding.

(2) We propose a novel approach, DualCor, to exploit the intra-event and inter-event argument correlations for ECE, and present it as a baseline to inspire the following research.

(3) Experiments[1] on the ECE dataset reflect the effectiveness of DualCor, and extensive analyses show potential research directions for future works.

2 Related Works

This paper explores a novel ECE task, which aims to extract the cause-effect event pairs with structured event information from plain texts. Existing event-causality-related researches mostly focus on event causality identification, which predicts the causality for the previously given event pairs. They can be roughly categorized into three groups: (1) Early works exploit the linguistic features (Riaz and Girju, 2013; Gao et al., 2019), causal patterns (Hu et al., 2017; Do et al., 2011) and statistical causal associations (Riaz and Girju, 2014) to explore the causality between events. (2) Recent researchers (Liu et al., 2020; Cao et al., 2021; Zuo et al., 2021a,b, 2020) pay the major focus on incorporating external knowledge for causality identification with limited training data. (3) Different from works above which conduct ECI within a single sentence, Tran Phu and Nguyen (2021) focus on document-level ECI, where the given events scatter in multiple sentences. Despite of their success, they all suffer from the issues of **event structure missing** and **causality component missing**. To our best knowledge, ECE is the first to simultaneously derive the structured event information and explicit causality component, which could better support the downstream applications.

Other than ECE, Relational Triple Extraction (RTE) (Yu et al., 2020; Cong et al., 2022) has the similar task formulation, the ideas of which could actually be adapted for ECE. Concretely,

[1]Dataset and source code for implementation are available here: https://github.com/cuishiyao96/ECE

RTE detects entity pairs in a sentence and predicts pre-defined relation types between them. Existing approaches for RTE could be roughly categorized into two lines. (1) Traditional joint methods, which solve RTE through sequential interrelated steps via task decomposition (Wei et al., 2020; Yu et al., 2020; Cong et al., 2020) or sequence generation (Zeng et al., 2018; Nayak and Ng, 2020). Unfortunately, these methods all suffer from the **exposure bias** (Wang et al., 2020) problem due to the gap from training to inference between multiple steps. (2) Unified joint methods, which simultaneously derive the triplet entities and relations in one-stage without cascading between steps, and is thus free from the exposure bias. These methods solve RTE in either a sequence-labeling manner (Zheng et al., 2017) or grid-filling manner (Wang et al., 2021b, 2020). However, the entity-centric RTE methods seem to struggle for the event-centric task, since events present more complicated argument correlations either intra- and inter events.

3 Task Formulation

Event causality extraction (ECE) aims to derive the cause-effect event pairs from plain texts. Here, a cause-effect event pair contains a *Cause component* and an *Effect component*, where each component denotes an event with a specific *event type* and its *event arguments* with their *event roles*. Given a piece of text, an event causality extraction system is required to predict all the cause-effect event pairs from it as Figure 1 shows.

4 Dual Grid Tagging Scheme

This section introduces our proposed dual grid tagging scheme for the ECE, including the tagging scheme and its decoding strategy. The specific model implementation is introduced in Section. 5.

4.1 Tagging Scheme

In general, we construct two grid tagging tables respectively for the cause/effect events, where each table extracts all the possible events occurring in the sentence. Formally, given an n-token sentence and m predefined event types, we construct two $m \times n$ grid tables for the cause and effect events respectively. As shown in Figure 2, each row denotes arguments within the same event type, while each column denotes tags assigned to the token in the sentence based on the event type. For each entry in the tables, we fill it with a tag in the form of

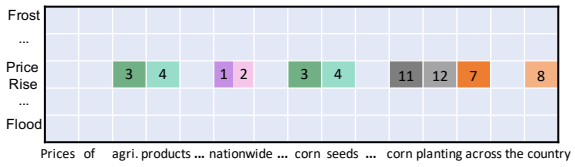

(a) Grid tagging for in the cause table.

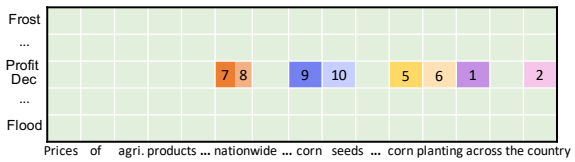

(b) Grid tagging in the effect table.

(c) Tags map.

Figure 2: Tagging scheme illustration with the sentence "Prices of agricultural products rose, but the nationwide soaring prices of corn seeds decreased the profit in corn planting across the country.", where the boundary-field is short as "S" and "E".

{`Cor-Rol-Bdy`} consisting of three fields, namely **correlation-field**, **role-field** and **boundary-field**:

(1) For the boundary-field: `Bdy` $\in$ {`Sta`, `End`}, we devise it to denote start and end position of argument spans. For example in Figure 2(a), we match the argument "corn seeds" by matching the `Cor-Rol-Sta` and `Cor-Rol-End` tags.

(2) For the role-field: `Rol`$\in\{$`Rol`$_i\}_i$ (i for role index), we devise it to denote the event role for each argument in an event, thus constituting an event structure. For example in Figure 2(a), we decide the argument "corn seeds" as a `Product`-role argument in `Price_Rising`-type event based on its `Cor-Product-Bdy` tag in `Price_Rising` row.

(3) For correlation-field: `Cor`$\in$ {`Intra`, `Inter`}, we devise it to denote event argument correlations in the cause-effect event pair. Specifically, `Intra` denotes the arguments belonging to the same event in the one causality component, while `Inter` denotes the arguments in the other causality component. For example, when predicting the cause event in the cause table, we predict not only the cause arguments (marked with `Intra`) with cause event type, but also the potential effect arguments (marked with `Inter`) as auxiliary arguments for mutual confirmation in causality pair matching. As Figure 2(a) shows, we not only predict argument

"corn seeds" with `Intra` for the `Price_Rise`-type cause event, but also predict "corn planting" with `Inter` tag as effect event argument. By matching argument "corn planting" with `Intra` tag in the effect table, we can derive a `Price_Rise`-typed and `Profit_Declination`-typed event pair.

Building upon the tagging scheme, the model can naturally extract causality event pairs with their arguments. Besides, the scheme learns event arguments for each type within separate type row, allowing the model to consider intra-event argument correlations with type-specific information. Moreover, the tagging scheme enforces the model to extract arguments in one causality component perceiving arguments in the other causality component, thus capturing inter-event argument correlations.

4.2 Decoding Strategy

Based upon the tagging scheme, we introduce the decoding strategy for the tagging results. Specifically, we decompose the process into three steps, including argument span decoding, event structure decoding and causality pair decoding. Appendix A also provides figure illustration to these three steps.

Step 1. Argument span decoding. To derive argument spans for cause/effect events, we adopt the nearest start-end match principle (Wei et al., 2020). Specifically, for entry tags having the same correlation-field and role-field in the same row, we match the start position to the nearest end position according to the position-field to obtain candidate argument spans. For example in Figure 2(a), this step ought to predict "agriculture products", "nationwide", "corn seeds", "corn planting" and "across the country" as candidate argument spans.

Step 2. Event structure decoding. To derive event structure for cause/effect events, we collect candidate argument spans attached to the same event type. Specifically, we merge the event arguments with correlation-field `Intra` belonging to the same row, resulting in structured candidate events. For example in Figure 2(a), given the candidate argument spans in Step 1, this step ought to select "agriculture products", "nationwide" and "corn seeds" with `Intra` tags as the candidate `Price_Rising`-type cause event arguments.

Step 3. Causality pair decoding. To derive causality pairs, we match inter-event correlated arguments between candidate cause and effect events. Specifically, we search the arguments co-occurring in both event tables simultaneously associating

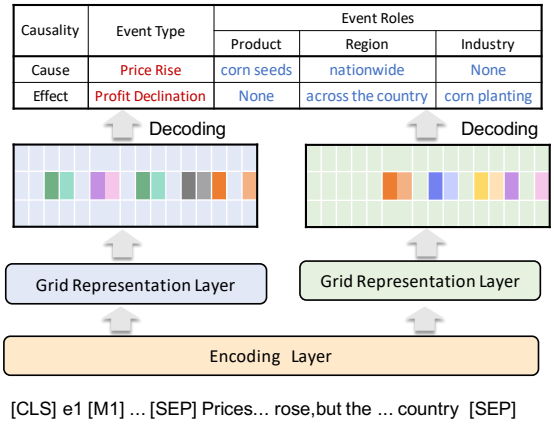

Figure 3: A toy illustration to our model architecture.

with correlation-field `Intra` and `Inter`, and then confirm cause-effect event arguments. For example in Figure 2(a), given the candidate event arguments in Step 2, this step ought to select "nationwide" and "corn seeds" as the true cause event arguments, since there also exist "nationwide" and "corn seeds" with `Inter` tags in the effect tables (Figure 2(b)). Similarly, this step also selects "corn planting", "across the country" as the arguments in the `Profit_Declination`-type effect events. Accordingly, it predicts the `Price_Rise`-type cause and `Profit_Declination`-type effect event pair as Figure 3 shows. Note that though "agriculture product" is also an event argument candidate of a `Price_Rise`-type event in Step 2, it is not included in the causality pair due to the absence of `Inter` correlation in the effect table.

5 Model

In this section, we introduce the model architecture to implement DualCor as Figure 3 shows.

5.1 Encoding Layer

This layer derives the contextualized representations of tokens in the sentence and event types. To facilitate the following event argument prediction, we intend to conduct **event type-aware encoding** which refines textual representations with event type information. Specifically, we concatenate the event types ahead of the sentence, and employ BERT (Devlin et al., 2019) for encoding thanks to its deep self-attention architectures (Vaswani et al., 2017). Supposing that a text consisting of n tokens $\{t_1, t_2, ..., t_n\}$ and m predefined event types $\{e_1, e_2, ..., e_m\}$ are given, the input sequence

is organized in the form as follows:

$$[\texttt{CLS}]\ \texttt{e}_1\ [\texttt{M}_1]\ \texttt{e}_2\ [\texttt{M}_2]...\ \texttt{e}_\texttt{m}\ [\texttt{M}_\texttt{m}]\ [\texttt{SEP}]\texttt{t}_1\ ...\ \texttt{t}_\texttt{n}\ [\texttt{SEP}] \tag{1}$$

where $[\texttt{M}_\texttt{j}]$ is the marker for the j_{th} event types e_j. We feed the input sequence into the encoder and use the output representations $\mathbf{H} = \mathbf{h}_1, \mathbf{h}_2, ..., \mathbf{h}_n$ corresponding to the sentence as token representations. Then, we gather representations of event type markers as event type representations, which is denoted as $\mathbf{E} = \mathbf{e}_1, \mathbf{e}_2, ..., \mathbf{e}_m$.

5.2 Grid Representation Layer

This section first details the function for producing entry representations, and then introduces how to apply it in both grid tables.

5.2.1 Semantic Fusion Function

Each entry in the grid respectively models the relation between one token and an event type for event argument deduction. For an entry connecting the j_{th} event type e_j and i_{th} token in the sentence, its representation $\mathbf{g}_{j,i}$ could be obtained via a fusion function ϕ by integrating the semantics of t_i and e_j as $\mathbf{g}_{j,i} = \phi(\mathbf{e}_j, \mathbf{h}_i)$. Intuitively, ϕ could be achieved in various semantic fusion ways including concatenation or addition. Considering that the same event argument span could play different role in different event types (Sheng et al., 2021), the decision of event arguments are conditioned on the event type. Hence, ϕ should imply the conditional dependency between event types and tokens. Accordingly, we adopt Conditional Layer Normalization (CLN) (Su, 2019) to implement ϕ. CLN is mostly based on the Layer Normalization (Ba et al., 2016), but it dynamically computes the gain γ and bias β based on the prior condition instead of directly deploying them as learnable parameters in neural networks. Given the event type representation $\mathbf{e}_j$ as condition and a token representation $\mathbf{h}_i$, the fusion function ϕ is achieved via CLN as:

$$\begin{aligned}\phi(\mathbf{e}_j, \mathbf{h}_i) &= \texttt{CLN}(\mathbf{e}_j, \mathbf{h}_i) = \gamma_j \odot (\frac{\mathbf{h}_i - \mu_i}{\sigma_i}) + \beta_j, \\ \gamma_j &= \mathbf{W}_\gamma \mathbf{e}_j + \mathbf{b}_\gamma, \beta_j = \mathbf{W}_\beta \mathbf{e}_j + \mathbf{b}_\beta,\end{aligned} \tag{2}$$

where $\mu_i \in \mathbb{R}$ and $\sigma_i \in \mathbb{R}$ are the mean and standard variance taken across the elements of $\mathbf{h}_i$, and $\gamma_j \in \mathbb{R}^d$ and $\beta_j \in \mathbb{R}^d$ are respectively the conditional gain and bias. In this way, the event type information is expressed as conditional information, and is thus integrated with token representations.

5.2.2 Grid Representation

We employ two semantic fusion functions, ϕ^c, ϕ^r, to respectively derive entry representations for the cause and effect grid table . Each semantic fusion function is implemented by a layer of CLN, and thus the entry representation is obtained as:

$$\begin{aligned}\mathbf{g}^c_{j,i} &= \phi^c(\mathbf{e}_j, \mathbf{h}_i) = \texttt{CLN}^c(\mathbf{e}_j, \mathbf{h}_i), \\ \mathbf{g}^r_{j,i} &= \phi^r(\mathbf{e}_j, \mathbf{h}_i) = \texttt{CLN}^r(\mathbf{e}_j, \mathbf{h}_i),\end{aligned} \tag{3}$$

where $\mathbf{g}^c_{j,i}$, $\mathbf{g}^r_{j,i}$ are respectively the entry representation in the cause and effect table for grid tagging.

5.3 Training and Inference

Since multiple tags could be simultaneously assigned towards (e_j, t_i) in each table, we conduct multi-label classification upon entry representations. Specifically, a fully-connected network predicts the probability of each tag for (e_j, t_i) as:

$$\mathbf{p}^{\mathcal{I}}_{j,i} = \texttt{sigmoid}(\mathbf{g}^{\mathcal{I}}_{j,i}\mathbf{W}^{\mathcal{I}} + \mathbf{b}^{\mathcal{I}}) \tag{4}$$

where $\mathcal{I} \in \{c, r\}$ is the symbol of grid field denoting the cause and effect grid table respectively, and each dimension of $\mathbf{p}^{\mathcal{I}}_{j,i}$ denotes the probability for a tag between (e_j, t_i). Consequently, we adapt Cross-Entropy loss as the loss function:

$$\mathcal{L}^{\mathcal{I}} = -\sum_{j=1}^{m}\sum_{i=1}^{n}\sum_{k \in C} \mathbb{I}(y^{\mathcal{I}}_{ji} = k)\texttt{log}(\mathbf{p}^{\mathcal{I}}_{j,i}[k]), \tag{5}$$

where C is the set of predefined tags, $\mathbf{p}^{\mathcal{I}}_{j,i}[k] \in [0, 1]$ is the predicted probability of tag k between (e_j, t_i) and $y^{\mathcal{I}}_{ji}$ is the ground truth tag between (e_j, t_i). $\mathbb{I}$ is a switching function which equals to 1 when $y^{\mathcal{I}}_{ji} = k$, otherwise 0. Following equation 5, we obtain losses from both grid tables, and aggregate them for the final training objective:

$$\mathcal{J}(\theta) = \mathcal{L}^c + \mathcal{L}^r. \tag{6}$$

For inference, $\mathbf{p}^{\mathcal{I}}_{j,i}$ is converted into tags whose probability overweights the scalar threshold $\tau^{\mathcal{I}} \in [0, 1]$, which is a manually tuned hyper-parameter.

6 Experiments

6.1 Dataset and Evaluation

Dataset We conduct experiments on the corpus (Tianchi, 2021) released by China Conference on Knowledge Graph and Semantic Computing 2021 (CCKS2021). The corpus comes from the

public news and reports, containing 7,000 sentences with an average length of 104 tokens. It annotates 15,816 events containing 7908 event causality pairs, covering 39 types of events and 3 types of event roles, namely `Product`, `Region` and `Industry`. To adapt this corpus into ECE task, we divide the corpus into training/validation/test set based on Cause-Effect event types. Specifically, CCKS2021 is divided into training/validation/test set with the proportion of 8 : 1 : 1. We rename the split dataset as ECE-CCKS.

Evaluation We evaluate our model using Precision (P), Recall (R) and Micro-F1 (F1) of three metrics. (1) **Event Argument Extraction (EAE) Metric** evaluates the model's ability to extract event arguments of interests. Like prior works (Yang et al., 2019), an argument is correctly predicted when its event type, span and event role simultaneously match the gold label. (2) **Cause-Effect Type (CET) Metric** measures whether both the predicted cause and effect event type match the golden answer. (3) **Event Causality Extraction (ECE) Metric** synthesizes the above two metrics, where an argument in ECE is correctly extracted when its predicted cause-effect event type, span and event role simultaneously meet the gold label.

6.2 Implementation Details

We employ $\text{BERT}_{\text{base}}$ (Devlin et al., 2019) as the encoder for our model and baselines. For DualCor, we manually tune all the hyper-parameters on the validation set. AdamW with learning rate of 3e-5 is adopted for model optimization. The model is trained 10 epoches with batch size of 8. The max length of sentence is 150 by padding shorter sentences and cutting longer ones. The threshold τ^c, τ^r are both set as 0.5.

6.3 Baselines

We employ a variety of baselines which could be classified into two streams.

Event-then-Causality methods. These methods first extract events from texts and then classify the causal relation. For event extraction, we choose three typical models. (1)**BERT-Softmax** (Devlin et al., 2019) adopts BERT to learn textual representations, and conducts sequence labeling for event extraction; (2) **BERT-CRF** utilizes conditional random field (CRF) to capture label dependencies upon the textual representations (Du and Cardie, 2020). (3) **DMBERT** (Wang et al., 2019) adopts dynamic multi-pooling (Chen et al., 2015) upon BERT to aggregate features for event extraction. (4) **PLMEE** (Yang et al., 2019) further adopts role-specific argument tagger upon BERT to solve the argument overlapping issue. After obtaining the events, we enumerate all possible cause-effect pairs and follow Zuo et al. (2021b) to build a Multilayer Perceptron classifier to decide the causality.

Event-with-Causality methods. Instead of separately deriving events and causality, these methods conduct event extraction with the causality taken into consideration and thus simultaneously derive the events and causality pair. To do this, we adapt three typical RTE methods as follows. (1) **Novel-tagging** designs a unified label space combining causality component (cause/effect), event types, event roles and argument boundary, and conducts ECE via sequence-labeling following Zheng et al. (2017). (2) **CasECE**, which is inspired by CasRel (Wei et al., 2020), first extracts the cause event, conditioned on which to derive the effect event. (3) **Pair-linking** works in a grid tagging manner following Wang et al. (2020). It first conducts event-type-level pair linking to derive the cause-effect event-type, which is then used as conditional information for token-pair linking to derive event arguments. Appendix B provides details about how we adapt these methods for ECE.

6.4 Main Results

We report the overall results in Table 1, and have observations as follows.

(1) The event-then-causality baselines generally produce weak performances, especially on the Precision indicator. The reason lies in that these methods extract events without considering the interest of causality. As a result, many causality-unrelated events are wrongly extracted, which would confuse the causality decision.

(2) Performances of the event-with-causality baselines are superior to the event-then-causality models, since the events are extracted with causality modeling, thus reducing the number of redundant events. However, their performances are still barely satisfactory, since the entity-oriented relation modeling strategy could not sufficiently to explore intra- and inter- correlations between events.

(3) DualCor achieves the best results among all baselines, we attribute this to that our designed dual grid tagging schema effectively explore the intra- and inter-event argument correlations. Despite of this, the overall ECE performance is far

	EAE(%)			CET(%)			ECE(%)		
	P	R	F1	P	R	F1	P	R	F1
BERT-softmax+Causality	32.55	35.11	33.78	49.61	64.20	55.97	30.47	31.52	30.99
BERT-CRF+Causality	35.52	34.10	34.79	53.22	60.95	56.82	31.02	31.28	31.15
DMBERT+Causality	34.27	38.18	36.12	52.87	**63.20**	57.58	30.08	34.93	32.33
PLMEE+Causality	34.22	40.70	37.18	58.11	60.20	59.13	29.98	41.14	34.69
Novel-tagging	**59.40**	28.47	38.49	49.79	61.70	55.11	51.52	26.75	35.22
CasECE	36.88	36.72	36.80	58.26	59.70	58.97	31.30	41.81	35.80
Pair-tagging	47.08	46.49	46.79	55.78	**62.95**	59.14	39.24	**47.69**	43.05
DualCor	58.05	**47.60**	**52.31**	**61.75**	58.19	**59.92**	**48.56**	44.85	**46.63**

Table 1: Overall results. The Wilcoxons test shows significant difference ($p<0.05$) between DualCor and baselines.

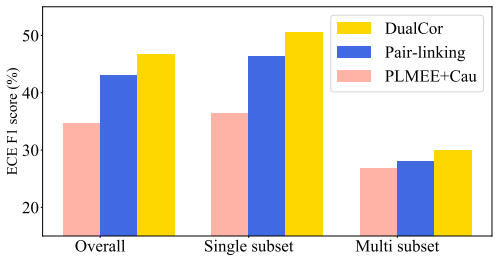

Figure 4: ECE performances on overall test set, Single and Multi subset. Appendix D shows detailed values.

Method	EAE	CET	ECE
DualCor	**52.50**	**61.60**	**47.58**
w/o Intra Cor	20.47	14.38	10.37
w/o Inter Cor	48.57	56.82	43.36
w/o type-aware encoding	47.69	56.00	43.16
$\phi \rightarrow$ Concatenation	51.39	59.56	45.88
$\phi \rightarrow$ Addition	51.96	61.08	46.83

Table 2: Ablation Study (F1%) on the validation set . Appendix E illustrates ablation on the test set.

from satisfactory. This reflects that ECE requires investigations from future works to improve it.

6.5 Single pair vs. Multi pairs

We notice that nearly 10% sentences in our dataset express multiple event causality pairs, and thus probe how the number of causality pairs influences the ECE performance. Specifically, we divide the test set into a **Single** subset where each sentence contains only one event causality pair, otherwise, **Multi** subset. Apart from DualCor, PLMEE+Causality (PLMEE+Cau in short) and Pair-linking are chosen as representatives for compared baselines, and we present their performances in Figure 4. Reading from the figure, we could see that (1) all models present a decreasing performance from Single to Multi subset, reflecting that ECE towards multiple causality pairs is much tricky. (2) Reasons for the weak performance on the Multi subset may be that the increasing number of causality pairs come from the increase of mentioned events, which demands more complicated inter-event correlations modeling (Sheng et al., 2022). (3) Since the performance on the Multi subset is obviously inferior to the overall and Single subset performances, we argue that **Multi-pairs** could be one great challenge which deserves investigation from future ECE works.

7 Analysis and Discussion

7.1 Ablation Study

To study how each module contributes to the performance, we ablate to DualCor on the validation set as Table 2 shows.

We probe the argument correlations via ablation to the tagging scheme. (1) w/o Intra-event argument correlations (Intra-Cor): To explore the necessity of Intra-Cor, we remove tags whose correlation-field are `Intra` in the tagging scheme. This leads to the sharp performance drops since Intra-Cor is the key to derive individual event from each grid. (2) w/o Inter-event argument correlations (Inter-Cor): To certify the effectiveness of Inter-Cor, we remove tags whose correlation-field are `Inter`. Without Inter-cor, the causality pairs are obtained by exhaustive enumeration between the cause and effect event which are individually derived from two tables. The ECE performance declines 4.42%, reflecting the importance of Inter-Cor. (3) We observe that the removing of either type of tags would hurt performances, verifying that these two correlations are both beneficial and functional for ECE.

We explore the influence of the model architecture via ablation to the encoding and grid representation layer. (1) w/o type-aware encoding: Instead of the collaborative encoding as Equation 1 shows,

Category	Example
Wrong Cause-Effect Type	Instance#1: The falling of stainless steel prices was caused by the drop in the cost of pure nickel. Gold: {Event type$_{\texttt{cause}}$: Cost Declination, event type$_{\texttt{effect}}$: Price Declination } Predicted: {Event type$_{\texttt{cause}}$: Price Declination, event type$_{\texttt{effect}}$: Price Declination }
Redundant Arguments	Instance#2: Feed prices rise across the country, reducing the profits in poultry industry. Gold: {Region$_{\texttt{cause}}$: across the country, Region$_{\texttt{effect}}$: None } Predicted: {Region$_{\texttt{cause}}$: across the country, Region$_{\texttt{effect}}$: across the country }
Missing Arguments	Instance#3: 50% of coke enterprises in Shanxi, Ningxia and 30% of those in Inner Mongolia have restricted their production, for which the coke output decreased. Gold: {Region$_{\texttt{reason}}$: Shanxi, Ningxia, *Inner Mongolia* } Predicted: {Region$_{\texttt{reason}}$: Shanxi, Ningxia }

Table 3: Error analysis, where we only present the associated event types and event arguments due to the space limitation. Appendix F provides the complete event causality pair for these three instances.

	#Para.	Training	Inference
DualCor	107.10M	18.6sents/s	38.9sents/s
Pair-linking	107.63M	6.4sents/s	19.4sents/s

Table 4: Efficiency comparison.

when we encode sentence using BERT while obtain event type embeddings by random initialization, the final performance declines by 4.42%. This manifests the importance of capturing semantic dependency between event types and each sentence. (2) $\phi \rightarrow$ Concatenation or Addition: To explore the impact of the semantic fusion function ϕ in Section 5.2.1, we respectively replace CLN as concatenation and addition. The performance degradation upon two variants signifies that CLN could better enhance token representations with event types, producing more expressive entry representations.

7.2 Efficiency Discussion

Since Pair-Linking also works in a grid tagging manner and achieves the comparable performance with DualCor, we discuss the efficiency of these two architectures from two aspects: parameter amount and running speed. For the sake of fairness, we run them on the same GPU server. Reading from Table 4, we notice that the amount of parameters of our model and Pair-Linking is roughly equal. We attribute this to that they both exploits the same basic encoding and grid representations learning strategy. However, we observe that the training and inference speeds of our model are respectively about 2.91 and 2.01 times faster than Pair-Linking. This is mainly because that the representation learning for two grids are carried in parallel in our model, while those of Pair-Linking are sequentially conducted. Considering analysis above, we could conclude that our model also maintains efficiency advantage over Pair-linking.

7.3 Error Analysis

To probe the drawbacks of DualCor and promote future works, we conduct error analysis towards 100 randomly selected failure instances. Here, we discuss three typical error types as Table 3 shows. (1) **Wrong Cause-Effect Type** refers to predicting the wrong combination of cause-effect event types as Instance#1. This error can severely hurt the final performances, since event arguments under the wrong cause-effect type would be regarded as false positive in ECE. We notice that almost 40% error cases of DualCor belong to this type, while that of Pair-linking is 32%. We attribute this to that our method mainly focus on correlations between event arguments, which lacks exact cause-effect modeling between event types, while the event-type-level pair linking in Pair-linking accounts for this. (2) **Redundant Arguments** denotes that the model predicts an argument which actually does not exist, as the redundant region for effect event in Instance#2. This kind of errors usually appear between the cause and effect event upon the same event role, which demonstrates the difficulty of deducing causality-specific event arguments. Though redundant arguments accounts for nearly 30% error cases of DualCor, it is almost 10% lower than that of Pair-linking. This reveals the importance of exploring intra- and inter- event argument correlations to discriminate the cause / effect event arguments. (3) **Missing Arguments** refers to that the model fails to predict the existed event argument, as the missed "Inner Mongolia" in Instance#3. We observe that it usually occurs for event roles which contains multiple event arguments, where more sophisticated modeling of intra-event arguments

correlations are required.

8 Conclusion

In this paper, we formulate a new task, Event Causality Extraction (ECE), which aims to extract the cause-effect event pairs with structured event information from plain texts. We propose a method based on an elaborately devised dual grid tagging scheme, which explores the intra- and inter-event argument correlations for the task. Experiment results prove the effectiveness of our method, and extensive analyses are conducted to point out several promising directions to inspire future works.

Acknowledgements

We would like to thank Bowen Yu and Jiangxia Cao for helpful discussions, support, and feedback on earlier versions of this work. We would also like to thank the anonymous reviewers for their insightful comments and suggestions. This work is supported by the National Key Research and Development Program of China (grant No.2021YFB3100600), the Strategic Priority Research Program of Chinese Academy of Sciences (grant No.XDC02040400) , the Youth Innovation Promotion Association of CAS (Grant No. 2021153).

References

Lei Jimmy Ba, Jamie Ryan Kiros, and Geoffrey E. Hinton. 2016. Layer normalization. *CoRR*.

Pengfei Cao, Xinyu Zuo, Yubo Chen, Kang Liu, Jun Zhao, Yuguang Chen, and Weihua Peng. 2021. Knowledge-enriched event causality identification via latent structure induction networks. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 4862–4872, Online. Association for Computational Linguistics.

CCKS. 2021. Joint extraction of financial events and causal relations.

Yubo Chen, Liheng Xu, Kang Liu, Daojian Zeng, and Jun Zhao. 2015. Event extraction via dynamic multi-pooling convolutional neural networks. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 167–176, Beijing, China. Association for Computational Linguistics.

Xin Cong, Jiawei Sheng, Shiyao Cui, Bowen Yu, Tingwen Liu, and Bin Wang. 2022. Relation-guided few-shot relational triple extraction. In *Proceedings of the 45th International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '22, page 2206–2213, New York, NY, USA. Association for Computing Machinery.

Xin Cong, Bowen Yu, Tingwen Liu, Shiyao Cui, Hengzhu Tang, and Bin Wang. 2020. Inductive unsupervised domain adaptation for few-shot classification via clustering. In *Machine Learning and Knowledge Discovery in Databases - European Conference, ECML PKDD 2020, Ghent, Belgium, September 14-18, 2020, Proceedings, Part II*, volume 12458 of *Lecture Notes in Computer Science*, pages 624–639. Springer.

Tarcisio Souza Costa, Simon Gottschalk, and Elena Demidova. 2020. Eventqa: A dataset for event-centric question answering over knowledge graphs. *Proceedings of the 29th ACM International Conference on Information and Knowledge Management*.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, Minneapolis, Minnesota. Association for Computational Linguistics.

Quang Do, Yee Seng Chan, and Dan Roth. 2011. Minimally supervised event causality identification. In *Proceedings of the 2011 Conference on Empirical Methods in Natural Language Processing*, pages 294–303, Edinburgh, Scotland, UK. Association for Computational Linguistics.

Xinya Du and Claire Cardie. 2020. Document-level event role filler extraction using multi-granularity contextualized encoding. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 8010–8020, Online. Association for Computational Linguistics.

Lei Gao, Prafulla Kumar Choubey, and Ruihong Huang. 2019. Modeling document-level causal structures for event causal relation identification. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 1808–1817, Minneapolis, Minnesota. Association for Computational Linguistics.

Chikara Hashimoto, Kentaro Torisawa, Julien Kloetzer, Motoki Sano, István Varga, Jong-Hoon Oh, and Yutaka Kidawara. 2014. Toward future scenario generation: Extracting event causality exploiting semantic relation, context, and association features. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 987–997, Baltimore, Maryland. Association for Computational Linguistics.

Zhichao Hu, Elahe Rahimtoroghi, and Marilyn Walker. 2017. Inference of fine-grained event causality from blogs and films. In *Proceedings of the Events and Stories in the News Workshop*, pages 52–58, Vancouver, Canada. Association for Computational Linguistics.

Jian Liu, Jian Liu, Yubo Chen, and Jun Zhao. 2020. Knowledge enhanced event causality identification with mention masking generalizations. In *IJCAI*.

Tapas Nayak and Hwee Tou Ng. 2020. Effective modeling of encoder-decoder architecture for joint entity and relation extraction. In *The Thirty-Fourth AAAI Conference on Artificial Intelligence, AAAI 2020, The Thirty-Second Innovative Applications of Artificial Intelligence Conference, IAAI 2020, The Tenth AAAI Symposium on Educational Advances in Artificial Intelligence, EAAI 2020, New York, NY, USA, February 7-12, 2020*, pages 8528–8535. AAAI Press.

Mehwish Riaz and Roxana Girju. 2013. Toward a better understanding of causality between verbal events: Extraction and analysis of the causal power of verb-verb associations. In *Proceedings of the SIGDIAL 2013 Conference*, pages 21–30, Metz, France. Association for Computational Linguistics.

Mehwish Riaz and Roxana Girju. 2014. In-depth exploitation of noun and verb semantics to identify causation in verb-noun pairs. In *Proceedings of the 15th Annual Meeting of the Special Interest Group on Discourse and Dialogue (SIGDIAL)*, pages 161–170, Philadelphia, PA, U.S.A. Association for Computational Linguistics.

Charlotte Rudnik, Thibault Ehrhart, Olivier Ferret, Denis Teyssou, Raphael Troncy, and Xavier Tannier. 2019. Searching news articles using an event knowledge graph leveraged by wikidata. *Companion Proceedings of The 2019 World Wide Web Conference*.

Jiawei Sheng, Shu Guo, Bowen Yu, Qian Li, Yiming Hei, Lihong Wang, Tingwen Liu, and Hongbo Xu. 2021. CasEE: A joint learning framework with cascade decoding for overlapping event extraction. In *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*, pages 164–174, Online. Association for Computational Linguistics.

Jiawei Sheng, Rui Sun, Shu Guo, Shiyao Cui, Jiangxia Cao, Lihong Wang, Tingwen Liu, and Hongbo Xu. 2022. Cored: Incorporating type-level and instance-level correlations for fine-grained event detection. In *SIGIR '22: The 45th International ACM SIGIR Conference on Research and Development in Information Retrieval, Madrid, Spain, July 11 - 15, 2022*, pages 1122–1132. ACM.

Jianlin Su. 2019. Conditional text generation based on conditional layer normalization.

Tianchi. 2021. Ccks2021 the dataset for financial event and causal relation extraction.

Minh Tran Phu and Thien Huu Nguyen. 2021. Graph convolutional networks for event causality identification with rich document-level structures. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 3480–3490, Online. Association for Computational Linguistics.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Ł ukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc.

Lihong Wang, Juwei Yue, Shu Guo, Jiawei Sheng, Qianren Mao, Zhenyu Chen, Shenghai Zhong, and Chen Li. 2021a. Multi-level connection enhanced representation learning for script event prediction. In *WWW '21: The Web Conference 2021, Virtual Event / Ljubljana, Slovenia, April 19-23, 2021*, pages 3524–3533. ACM / IW3C2.

Xiaozhi Wang, Xu Han, Zhiyuan Liu, Maosong Sun, and Peng Li. 2019. Adversarial training for weakly supervised event detection. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 998–1008, Minneapolis, Minnesota. Association for Computational Linguistics.

Yijun Wang, Changzhi Sun, Yuanbin Wu, Hao Zhou, Lei Li, and Junchi Yan. 2021b. UniRE: A unified label space for entity relation extraction. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 220–231, Online. Association for Computational Linguistics.

Yucheng Wang, Bowen Yu, Yueyang Zhang, Tingwen Liu, Hongsong Zhu, and Limin Sun. 2020. TPLinker: Single-stage joint extraction of entities and relations through token pair linking. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 1572–1582, Barcelona, Spain (Online). International Committee on Computational Linguistics.

Zhepei Wei, Jianlin Su, Yue Wang, Yuan Tian, and Yi Chang. 2020. A novel cascade binary tagging framework for relational triple extraction. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 1476–1488, Online. Association for Computational Linguistics.

Sen Yang, Dawei Feng, Linbo Qiao, Zhigang Kan, and Dongsheng Li. 2019. Exploring pre-trained language models for event extraction and generation. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 5284–5294, Florence, Italy. Association for Computational Linguistics.

Bowen Yu, Zhenyu Zhang, Xiaobo Shu, Tingwen Liu, Yubin Wang, Bin Wang, and Sujian Li. 2020. Joint extraction of entities and relations based on a novel decomposition strategy. In *ECAI 2020 - 24th European Conference on Artificial Intelligence, 29 August-8 September 2020, Santiago de Compostela, Spain, August 29 - September 8, 2020 - Including 10th Conference on Prestigious Applications of Artificial Intelligence (PAIS 2020)*, volume 325 of *Frontiers in Artificial Intelligence and Applications*, pages 2282–2289. IOS Press.

Xiangrong Zeng, Daojian Zeng, Shizhu He, Kang Liu, and Jun Zhao. 2018. Extracting relational facts by an end-to-end neural model with copy mechanism. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 506–514, Melbourne, Australia. Association for Computational Linguistics.

Suncong Zheng, Feng Wang, Hongyun Bao, Yuexing Hao, Peng Zhou, and Bo Xu. 2017. Joint extraction of entities and relations based on a novel tagging scheme. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1227–1236, Vancouver, Canada. Association for Computational Linguistics.

Xinyu Zuo, Pengfei Cao, Yubo Chen, Kang Liu, Jun Zhao, Weihua Peng, and Yuguang Chen. 2021a. Improving event causality identification via self-supervised representation learning on external causal statement. In *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*, pages 2162–2172, Online. Association for Computational Linguistics.

Xinyu Zuo, Pengfei Cao, Yubo Chen, Kang Liu, Jun Zhao, Weihua Peng, and Yuguang Chen. 2021b. LearnDA: Learnable knowledge-guided data augmentation for event causality identification. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 3558–3571, Online. Association for Computational Linguistics.

Xinyu Zuo, Yubo Chen, Kang Liu, and Jun Zhao. 2020. KnowDis: Knowledge enhanced data augmentation for event causality detection via distant supervision. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 1544–1550, Barcelona, Spain (Online). International Committee on Computational Linguistics.

A Decoding strategy

Step 1. Argument span decoding. In this stage, we derive argument spans for cause/effect events using the nearest start-end match principle (Wei et al., 2020). Specifically, for those entry tags having the same correlation-field and role-field in the same row, we match the start position to the nearest end position according to the position-field to obtain candidate argument spans Figure 5.(a).

Step 2. Event structure decoding. In this stage, we collect candidate argument spans attached to the same event types. Specifically, we merge the event arguments with correlation-field `Intra` belonging to the same row, resulting in structured candidate events in Figure 5.(b).

Step 3. Causality pair decoding. To derive causality pairs, we match inter-event correlated arguments between candidate cause and effect events. Specifically, we first obtain event argument with correlation-field `Inter` in each table as Figure 5.(c). Then, we search the arguments co-occurring in both event tables simultaneously associating with correlation-field `Intra` and `Inter`, and merge them to form the cause-effect pair in Figure 5.(d).

Table	Argument Spans
Cause	agriculture products, corn seeds, nationwide, corn planting, across the country
Effect	corn seeds, nationwide, corn planting, across the country

(a) Argument spans derived from Step 1.

Table	Event Type	Event Roles		
		Product	Region	Industry
Cause(Intra)	Price Rise	agriculture products corn seeds	nationwide	None
Effect(Intra)	Profit Declination	None	across the country	corn planting

(b) Event candidates derived from Step 2 via **Intra** field.

Table	Event Type	Event Roles		
		Product	Region	Industry
Cause(Inter)	Price Rise	None	across the country	corn planting
Effect(Inter)	Profit Declination	corn seeds	nationwide	None

(c) Inter-arguments derived via **Inter** field.

Causality	Event Type	Event Roles		
		Product	Region	Industry
Cause	Price Rise	corn seeds	nationwide	None
Effect	Prifit Declination	None	across the country	corn planting

(d) The cause-effect Event pair, which is derived by merging arguments which co-appear in both event tables with correlation-field Intra and Inter.

Figure 5: Detailed illustration to decoding strategy.

B Details about adapted baseliens

We detail the adaption of RTE methods to ECE.

(1) **Novel-tagging** is adapted from Zheng et al. (2017). It performs RTE through sequence labeling with a novel tagging scheme, which combines the label spaces of relation types and relation roles (subject and object), Similarly, we adopt a unified label space combining cause/effect, event type, event roles and argument boundary tag, namely $\{$`Causality-EventType-EventRole-Bdy`$\}$. Given the 2 types of causality components, 39 predefined event types, 3 predefined event roles and the Start/End boundary indicator, the capacity of the unified label space is $2 \times 39 \times 3 \times 2 = 468$. We employ the unified labels to tag the tokens in a sequence labeling manner with BERT+Softmax. Note that we only deploy BERT+Softmax for sequence labeling here, since the joint label space is too large for BERT-CRF to implement on our experiment devices.

(2) **CasECE** is adapted from CasRel (Wei et al., 2020), which conducts RTE by modeling the relations as functions mapping subject entity to object entity. Similarly, we regard the causality relation as the function which maps the cause event to the effect event. Following CasRel, we first extract the cause event, and then conditioned on it to derive the effect event. During this process, PLMEE (Yang et al., 2019) is utilized as the event extractor.

(3) **Pair-linking** is adapted from TPLinker (Wang et al., 2020), where RTE is formulated as a token pair linking problem which aligns the boundary tokens of entity pairs under each relation type. Similarly, we intend to respectively extract event arguments under specific cause-effect event types. Specifically, we first conduct event-type-level pair linking to derive the cause-effect event types. Then, we utilize CLN to refine textual representations enhanced with cause-effect event type pair information, and conduct token-pair-linking to extract the event arguments for the specific cause and effect event.

C Other encoders

We report performances of DualCor using different basic encoders in Table 5.

D Single pair vs. Multi pairs

We detail ECE performances on the overall test, Single and Multi subset in Table 6.

E Ablation on the test

We provide ablation study on the test set in Table 7.

F Error analysis

This section provides the complete instances for error analysis in Table 8.

	EAE(%)			CET(%)			ECE(%)		
	P	R	F1	P	R	F1	P	R	F1
DualCor$_{\text{BERT}_{\text{base}}}$	58.05	47.60	52.31	61.75	58.19	59.92	48.56	44.85	46.63
DualCor$_{\text{Roberta}_{\text{base}}}$	61.46	46.29	52.80	66.14	58.19	61.91	52.801	44.89	48.53
DualCor$_{\text{Roberta}_{\text{large}}}$	63.44	**50.48**	56.22	67.52	**62.70**	65.02	54.67	**49.80**	52.12
DualCor$_{\text{MacBERT}}$	**67.64**	49.19	**56.96**	**70.68**	60.95	**65.45**	**58.29**	48.02	**52.66**

Table 5: Overall results on the test set.

	Overall(%)			Single Subset(%)			Multi Subset(%)		
	P	R	F1	P	R	F1	P	R	F1
PLMEE+Causality	29.98	41.14	34.69	29.89	46.42	36.37	30.58	23.76	26.74
Pair-linking	39.24	**47.69**	43.05	40.31	**54.32**	46.28	32.15	**24.82**	28.03
DualCor	**48.64**	44.85	**46.67**	**49.39**	51.56	**50.46**	**43.65**	22.72	**29.89**

Table 6: ECE performances on the overall test set, Single subset and Multi subset.

Method	EAE	CET	ECE
DualCor	**52.36**	**59.96**	**46.67**
w/o Intra Cor	19.11	12.51	9.32
w/o Inter Cor	49.29	55.04	42.88
w/o type-aware encoding	47.21	54.72	41.79
$\phi \rightarrow$ Concatenation	50.91	57.99	44.04
$\phi \rightarrow$ Addition	51.52	58.90	45.05

Table 7: Ablation Study: F1% upon the three metrics on the test set.

Category	**Example**
Wrong Cause-Effect Type	Instance#1: The falling of stainless steel prices was caused by the drop in the cost of pure nickel. Gold: {Event type$_{\text{cause}}$: Cost Declination, Event type$_{\text{effect}}$: Price Declination, Product$_{\text{cause}}$: pure nickel, Product$_{\text{effect}}$: stainless steel, Region$_{\text{cause}}$: None, Industry$_{\text{effect}}$: None Industry$_{\text{cause}}$: None, Industry$_{\text{effect}}$: None } Predicted: {Event type$_{\text{cause}}$: Price Declination, event type$_{\text{effect}}$: Price Declination, Product$_{\text{cause}}$: pure nickel, Product$_{\text{effect}}$: stainless steel, Region$_{\text{cause}}$: None, Industry$_{\text{effect}}$: None Industry$_{\text{cause}}$: None, Industry$_{\text{effect}}$: None }
Redundant Arguments	Instance#2: Feed prices rise across the country, reducing the profits in poultry industry. Gold: { Event type$_{\text{cause}}$: Price Rise, Event type$_{\text{effect}}$: Profit Declination, Product$_{\text{cause}}$: feed, Product$_{\text{effect}}$: None, Region$_{\text{cause}}$: across the country, Region$_{\text{effect}}$: None, Industry$_{\text{cause}}$: None, Industry$_{\text{effect}}$: poultry industry } Predicted: { Event type$_{\text{cause}}$: Price Rising, Event type$_{\text{effect}}$: Profit Declination, Product$_{\text{cause}}$: feed, Product$_{\text{effect}}$: None, Region$_{\text{cause}}$: across the country, Region$_{\text{effect}}$: across the country, Industry$_{\text{cause}}$: None, Industry$_{\text{effect}}$: poultry industry }
Missing Arguments	Instance#3: 50% of coke enterprises in Shanxi, Ningxia and 30% of those in Inner Mongolia have restricted their production, for which the supply of coke output. Gold: {Event type$_{\text{cause}}$: Production Restriction, Event type$_{\text{effect}}$: Supply Reduction, Product$_{\text{cause}}$: coke, Product$_{\text{effect}}$: coke, Region$_{\text{cause}}$: Shanxi, Ningxia, *Inner Mongolia*, Region$_{\text{effect}}$: None, Industry$_{\text{cause}}$: None, Industry$_{\text{effect}}$: None } Predicted: {Event type$_{\text{cause}}$: Production Restriction, Event type$_{\text{effect}}$: Supply Reduction, Product$_{\text{cause}}$: coke, Product$_{\text{effect}}$: coke, Region$_{\text{cause}}$: Shanxi, Ningxia, Region$_{\text{effect}}$: None, Industry$_{\text{cause}}$: None, Industry$_{\text{effect}}$: None }

Table 8: The complete results of error analysis.

SCL-RAI: Span-based Contrastive Learning with Retrieval Augmented Inference for Unlabeled Entity Problem in NER

Shuzheng Si[1,2]*, **Shuang Zeng**[1,2]*, **Jiaxing Lin**[1,2], **Baobao Chang**[1]†
[1]Key Laboratory of Computational Linguistics, Peking University, MOE, China
[2]School of Software and Microelectronics, Peking University, China
{sishuzheng,jxlin}@stu.pku.edu.cn {zengs,chbb}@pku.edu.cn

Abstract

Named Entity Recognition is the task to locate and classify the entities in the text. However, *Unlabeled Entity Problem* in NER datasets seriously hinders the improvement of NER performance. This paper proposes **SCL-RAI** to cope with this problem. Firstly, we decrease the distance of span representations with the same label while increasing it for different ones via span-based contrastive learning, which relieves the ambiguity among entities and improves the robustness of the model over unlabeled entities. Then we propose retrieval augmented inference to mitigate the decision boundary shifting problem. Our method significantly outperforms the previous SOTA method by 4.21% and 8.64% F1-score on two real-world datasets.

1 Introduction

As a fundamental task in NLP, Named Entity Recognition aims to locate and classify named entities in the text. Due to the large-scale well-annotated datasets, deep-learning based methods (Li et al., 2022b; Devlin et al., 2019) have achieved great success. However, in real-world datasets, such as Ling and Weld (2012) with 112 fine-grained named entity tags, a large set of entity classes may cause inevitable missing annotations. Moreover, to obtain large NER datasets in practical scenarios, the distant supervision approach (Ren et al., 2015; Fries et al., 2017) may make this problem even worse, since the entity dictionary cannot cover all entities. Previous work (Li et al., 2021; Shang et al., 2018) find that this problem seriously hinders the performance of the NER model and name this problem as *Unlabeled Entity Problem*. As shown in Figure 1, the unlabeled second "NBA" may confuse model and introduce unnecessary noise.

To cope with this problem, several attempts from different perspectives have been proposed. Inspired by positive-unlabeled (PU) learning (Li and Liu, 2005), Peng et al. (2019) use a weighted loss to assign low weights to false negative words and build distinct binary classifiers for different entity types. However, they require prior information or heuristics (Li et al., 2022c) and the unlabeled entities still misguide the classifiers, bringing ambiguity among neighboring entities (Li et al., 2021). Yang et al. (2018); Jie et al. (2019) introduce the Partial CRF (Lafferty et al., 2001) to marginalize the instances that are consistent with the incomplete annotation. However, they require additional well-annotated corpus to get ground truth negative instances, which are usually unavailable in practice. Recently, Li et al. (2021) perform down-sampling among non-entity instances within annotation when computing loss function, in order to mitigate the misguidance from possible unlabeled entities. Li et al. (2022c) further propose a weighted and adaptive sampling distribution to introduce direction to real unlabeled entities when down-sampling. However, the inherent randomness of sampling strategy may still keep some unlabeled entities when computing loss then make the decision boundary biased (Li et al., 2022a). As shown in Figure 2, the learned decision boundary for training example containing unlabeled entity instances tends to shift from the expected boundary towards the entity side. The previous works do not consider this problem.

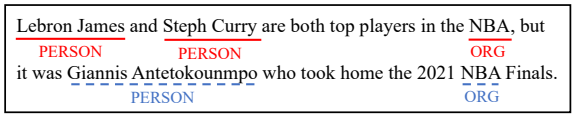

Figure 1: A toy case to show *Unlabeled Entity Problem*. The labeled entities are underlined with red lines while the unlabeled entities are underlined with dashed lines.

To deal with these challenges, this paper proposes the **S**pan-based **C**ontrastive **L**earning with **R**etrieval **A**ugmented **I**nference (**SCL-RAI**) to tackle *Unlabeled Entity Problem*, which mitigates

*Equal contribution.
†Corresponding author.

Proceedings of the 29th International Conference on Computational Linguistics, pages 2313–2318
October 12–17, 2022.

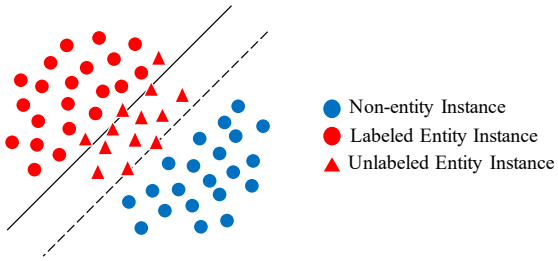

Figure 2: Illustration for decision boundary shifting phenomenon. The solid line is the learned boundary from datasets with unlabeled entities, the dashed line represents the expected boundary.

the limitations mentioned above, i.e., demanding additional corpus, ambiguity among neighboring entities and decision boundary shifting problem. Firstly, SCL-RAI tries to decrease the distance among span representations with the same labels while increasing it for different ones. Benefiting from our well-designed span-based contrastive learning, the ambiguity between entities is mitigated by the increased representation distance, so the model can capture the differences among different entity labels. We show in experiment that this contrastive learning objective also improves the model robustness under unlabeled entities. Furthermore, we propose Retrieval Augmented Inference to relieve decision boundary shifting phenomenon. It caches the center point representation for each entity type from the training set. Then, it computes a label distribution via cached representation and interpolates it with the distribution from the backbone NER model. Experiments on two real-world datasets show that SCL-RAI significantly outperforms previous SOTA methods.

2 Methodology

Our SCL-RAI consists of three modules: Span-based NER Model, Span-based Contrastive Learning, and Retrieval Augmented Inference.

2.1 Span-based NER Model

Span-based NER models have shown a strong ability to solve NER task, especially in flat NER and nested NER problem (Yu et al., 2020). For fair comparison, we follow Li et al. (2021, 2022c) on the design of Span-based NER model. Firstly, we use BERT (Devlin et al., 2019) as the text encoder to get the representations for words in sentence x:

$$[\mathbf{h}_1, \mathbf{h}_2, ..., \mathbf{h}_n] = BERT(x) \quad (1)$$

where h_i is the representation for word x_i. For each text span $s_{i,j}$ ranging from i-th word and j-th word in x, we get the span representation $\mathbf{s}_{i,j}$ as:

$$\mathbf{s}_{i,j} = \mathbf{h}_i \oplus \mathbf{h}_j \oplus (\mathbf{h}_i - \mathbf{h}_j) \oplus (\mathbf{h}_i \odot \mathbf{h}_j) \quad (2)$$

where $\oplus$ is the concatenation operation and $\odot$ is the element-wise product operation. Finally, we use a two-layer non-linear projection to obtain the entity label distribution for every span $s_{i,j}$:

$$\mathbf{r}_{i,j} = tanh(W\mathbf{s}_{i,j}) \quad (3)$$

$$\mathbf{o}_{i,j} = softmax(V\mathbf{r}_{i,j}) \quad (4)$$

where W and V are trainable parameter matrices.

And the probability of l-th gold entity label for span instance $s_{i,j}$ is $\mathbf{o}_{i,j,l}$:

$$\mathbf{o}_{i,j,l} = \mathbf{v}_l^T \mathbf{r}_{i,j} \quad (5)$$

We use cross entropy (CE) loss as our training objective:

$$loss_{ce} = \sum_{s_{i,j} \in D} -log(\mathbf{o}_{i,j,l}) \quad (6)$$

where D is the collection of all training instances.

2.2 Span-based Contrastive Learning

To mitigate the ambiguity among entities, SCL-RAI tries to pull span belonging to the same class together in embedding space, while simultaneously pushing apart clusters of span from different classes.

This way, the clusters in entity representation space could better distinguish different types of entities. To this end, we propose a novel span-based contrastive learning objective to mitigate the ambiguity problem among entities. Meanwhile, we find in our experiment that this contrastive learning objective could also improve the robustness of SCL-RAI under unlabeled entity noises.

For span-based NER model, we conduct contrastive learning within a batch of span instances D; We use the cosine similarity to represent the distance between the span representations of two instances $s_{i,j}$ and $s_{\hat{i},\hat{j}}$:

$$d_{s_{i,j}, s_{\hat{i},\hat{j}}} = \frac{\mathbf{r}_{i,j} \cdot \mathbf{r}_{\hat{i},\hat{j}}}{|\mathbf{r}_{i,j}||\mathbf{r}_{\hat{i},\hat{j}}|} \quad (7)$$

Then the span-based supervised contrastive learning loss function $loss_{scl}$ is defined as:

$$loss_{scl} = -\sum_{l \in L} \sum_{s_{i,j} \in D_l} \frac{1}{N_l - 1} \sum_{s_{\hat{i},\hat{j}} \in D_l} F(\mathbf{r}_{i,j}, \mathbf{r}_{\hat{i},\hat{j}}) \quad (8)$$

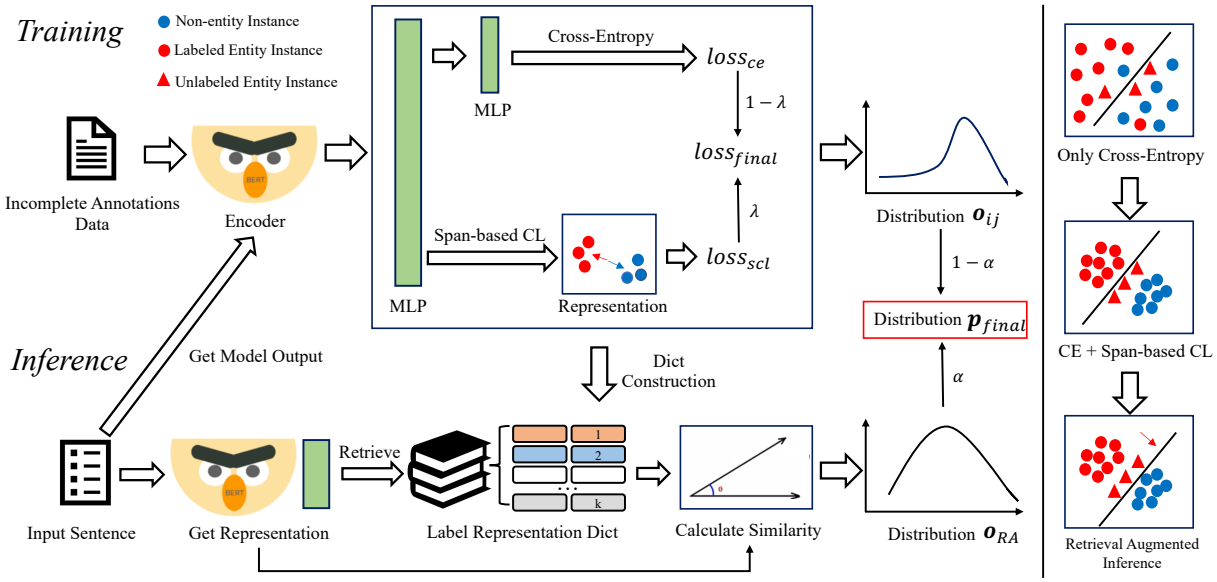

Figure 3: General architecture of SCL-RAI.

where L is the size of the entity label set; $(i,j) \neq (\hat{i},\hat{j})$; N_l is the total number of span instances with the same entity label l in the batch; D_l is the collection of all training span instance with l-th entity label. $F(\mathbf{r}_{i,j}, \mathbf{r}_{\hat{i},\hat{j}})$ is:

$$F(\mathbf{r}_{i,j}, \mathbf{r}_{\hat{i},\hat{j}}) = log\frac{exp(d_{s_{i,j},s_{\hat{i},\hat{j}}}/\tau)}{\sum_{s_{m,n}\in D_{\bar{l}}} exp(d_{s_{i,j},s_{m,n}}/\tau)} \tag{9}$$

where τ is the temperature. $D_{\bar{l}}$ is the collection of span instances not with l-th entity label.

This span-based supervised contrastive learning loss pushes the span representations of instances with the same entity labels closer and pushes the span representations of instances with the different entity labels farther. We confirm in our experiment that this contrastive learning objective indeed improves the model robustness under unlabeled entities, compared with previous works.

Then we combine the cross entropy loss and span-based contrastive learning loss to get our final loss function:

$$loss_{final} = (1-\lambda) * loss_{ce} + \lambda * loss_{scl} \tag{10}$$

where λ is a scalar hyperparameter.

2.3 Retrieval Augmented Inference

As we get the discriminative entity span representations via span-based contrastive Learning, we propose **R**etrieval **A**ugmented **I**nference (**RAI**) to facilitate the decoding process at the inference stage. As shown in Figure 3, RAI can be split into two parts: (i) Firstly, it generates a central point representation for each entity type from the training set and stores them in a dictionary $Dict$. (ii) It calculates the similarity between the representation of span to be predicted and each entity type representation in $Dict$ to get the retrieval augmented label distribution $\mathbf{o}_{RA}$, then interpolates the distribution $\mathbf{o}_{i,j,l}$ from span-based NER model with $\mathbf{o}_{RA}$ to get the final label distribution. For example, the second "NBA" in Figure 1 will get high similarity value with the the central point representation of the entity type "ORG", due to the similar context with other "ORG" entities in training set. So it could decrease the possible high probability of non-entity label from Span-based NER model and increase it of "ORG" entity label. This way, we can shift the learned decision boundary toward the expected boundary in Figure 2.

Dictionary Construction: The dictionary $Dict$ used in SCL-RAI consists of a set of key-value pairs. Each key is an entity type and the corresponding value is the calculated central point representation from the training set. After training the model, we could get the dictionary for storing representations of all entity tags:

$$Dict = \{K, V\} = \{(l, \mathbf{r}_l) | \forall l \in L\} \tag{11}$$

$$\mathbf{r}_l = \sum_{s_{m,n}\in T_l} \frac{1}{N_l} \mathbf{r}_{m,n} \tag{12}$$

where T_l is the collection of all training span instances with l-th entity label; N_l is the total number

of span instances with the label l in the training set.

Label Distribution Interpolation: At the same time, Span-based NER model outputs representation $\mathbf{r}_{i,j}$ for the span to be predicted and its label distribution $\mathbf{o}_{i,j}$. Then we calculate the cosine similarity between $\mathbf{r}_{i,j}$ each cached representation from $Dict$ to obtain a new label distribution, i.e., retrieval augmented label distribution $\mathbf{o}_{RA}$:

$$\mathbf{sim}_{(i,j)} = concat(\frac{\mathbf{r}_{i,j} \cdot \mathbf{r}_{l_1}}{|\mathbf{r}_{i,j}||\mathbf{r}_{l_1}|}, ..., \frac{\mathbf{r}_{i,j} \cdot \mathbf{r}_{l_L}}{|\mathbf{r}_{i,j}||\mathbf{r}_{l_L}|}) \quad (13)$$

$$\mathbf{o}_{RA} = softmax(\mathbf{sim}_{(i,j)}) \quad (14)$$

where L is the number of entity labels.

We then set the value of non-entity label in $\mathbf{o}_{RA}$ to 0:

$$\mathbf{o}_{RA}[v] = 0 \quad (15)$$

where v is the index for the non-entity. This ensures the similarity of non-entity label does not participate in interpolation.

Finally, we interpolate the distribution $\mathbf{o}_{i,j,l}$ from span-based NER model with $\mathbf{o}_{RA}$ to get the final label distribution $\mathbf{p}_{final}$:

$$\mathbf{p}_{final} = (1 - \alpha) * \mathbf{o}_{i,j} + \alpha * \mathbf{o}_{RA} \quad (16)$$

where α is a hyperparameter to makes a balance between two distributions.

3 Experiments

3.1 Experimental Settings

Following (Yang et al., 2018; Li et al., 2021, 2022c), we adopt EC and NEWS as our datasets. The training set of EC and NEWS both consist of two parts: (1) the well-annotated set $\mathcal{A}$; (2) the distantly supervised set $\mathcal{DS}$, which contains a large amount of incompletely annotated sentences. Therefore, NER models trained on EC or NEWS suffer from *Unlabeled Entity Problem*. The dev/test set used in two datasets are well-annotated to evaluate the performance of model trained on datasets containing label noise.

EC In the e-commerce domain (EC), there are five types of entities: Brand, Product, Model, Material, and Specification. It contains 2,400 sentences labeled by annotators. The well-annotated set $\mathcal{A}$ is split into three sets: 1,200 sentences for training, 400 for dev, and 800 for testing. Then Yang et al. (2018) collect a list of entities to construct a dictionary from the training data and perform distant supervision on raw data to get the distantly supervised set $\mathcal{DS}$, which contains 2,500 sentences.

Models	EC	NEWS
BERT-MRC	55.72	74.55
BERT-Biaffine Model	55.99	74.57
PU Learning	61.22	77.98
Partial CRF	60.08	78.38
Weighted Partial CRF	61.75	78.64
Vanilla Negative Sampling	66.17	85.39
Variant Negative Sampling	67.03	86.15
SCL-RAI	69.70	94.11
SCL-RAI+Vanilla Neg. Sampl.	**71.24**	**94.79**
- RAI	70.65 (-0.59)	93.71 (-1.08)
- SCL & RAI	66.17 (-5.07)	85.39 (-9.40)

Table 1: The F1-score results on two real-world datasets. "SCL" denotes Span-based Contrastive Learning and "RAI" denotes Retrieval Augmented Inference.

Models	$\mathcal{A}$	$\mathcal{A} + \mathcal{DS}$	Δ
Vanilla Negative Sampling	94.38	85.39	-8.99
SCL-RAI+Vanilla Neg. Sampl.	**95.33**	**94.79**	**-0.54**

Table 2: The robustness of SCL-RAI over unlabeled entities with different training set on NEWS dataset.

NEWS For news domain, Yang et al. (2018) use a NER data from MSRA (Levow, 2006). Yang et al. (2018) only keep entity type PERSON to get NEWS. Then (Yang et al., 2018) randomly select 3,000 sentences as training dataset, 3,328 as dev data, and 3,186 as testing data to get the well-annotated set $\mathcal{A}$. The rest set of MSRA is used as raw data, having 36,602 sentences. Yang et al. (2018) collect a list of person names from the training data. Then Yang et al. (2018) add additional names to the list. Finally, the list has 71,664 entries. Yang et al. (2018) perform distant supervision on raw data to obtain extra 3,722 sentences as the distantly supervised set $\mathcal{DS}$.

We adopt the same hyperparameter configurations for two datasets. We use Adam (Kingma and Ba, 2015) as optimizer with learning rate as 10^{-5} and bert-base as our encoder following Li et al. (2021, 2022c). The dimension of scoring layers W is set as 256. The scalar weighting hyperparameters λ and α are set as 0.1 and 0.5. The temperature parameter τ is set to 0.1. Since the label distribution is very unbalanced (most instances are non-entity), we also apply negative sampling and the same sampling rate as Li et al. (2021).

For evaluation, we use conlleval script[1] to compute the F1-score.

[1]https://www.clips.uantwerpen.be/conll2000/chunking/conlleval.txt.

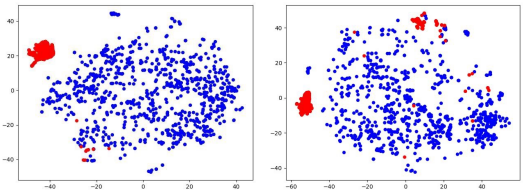

Figure 4: t-SNE plots of the representations on NEWS test set. CE+Span-based CL (left), CE only (right). Red dot denote entities and blue dot denote the non-entities.

3.2 Results and Analysis

We report the results from: (1) **Traditional NER methods**: BERT-MRC (Yu et al., 2020) and BERT-Biaffine (Yu et al., 2020) ; (2) **Recent Attempts on Unlabeled Entity Problem**: PU Learning (Peng et al., 2019), Partial CRF (Yang et al., 2018), Weighted Partial CRF (Jie et al., 2019), Vanilla Negative Sampling (Li et al., 2021), Variant Negative Sampling (Li et al., 2022c) and our **SCL-RAI**. Since our method is orthogonal to that of (Li et al., 2021), we also report the results of SCL-RAI with their negative sampling strategy "**SCL-RAI**+Vanilla Neg. Sampl." to get better results.

We report our results in Table 1. Firstly, traditional NER models perform poorly on real-world datasets. So the SOTA NER models on well-annotated datasets are not robust over the *Unlabeled Entity Problem*. Then, our method has achieved new state-of-the-art results on the two datasets. Compared with SOTA model (Li et al., 2022c), we achieve the improvements of 2.67% F1 on EC and 7.96% on NEWS. With the negative sampling strategy, we further get the improvements of 4.21% F1 on EC and 8.64% F1 on NEWS. The improvements shows that our model has a stronger ability to mitigate the noise from unlabeled entities.

To verify the effectiveness of SCL-RAI, we show ablation studies in Table 1. It is clear that Span-based Contrastive Learning and Retrieval Augmented Inference are both important to cope with *Unlabeled Entity Problem*. In Table 2 and Table 4, we show the robustness of our model over unlabeled entities on NEWS dataset. Our SCL-RAI can obtain less F1 degradation when introducing dataset $\mathcal{DS}$ with unlabeled entities. In Figure 4, we show t-SNE plots of the learned representations of 2000 instances on NEWS test set, comparing Cross Entropy (CE) with and without the Span-based CL term. We can clearly see that the Span-based CL term enforces more compact clustering of entities.

For span-based NER model, we also conduct our Span-based Contrastive Learning within a batch of span instances. Therefore, we test the robustness of Span-based Contrastive Learning for different batch sizes on EC. As shown in Table 3, we can clearly find that Span-based Contrastive Learning is robust for different batch sizes.

Batch Size	F1-score
8	70.78
16	71.24
32	70.87
64	70.79

Table 3: The Span-based Contrastive Learning robustness testing results on EC.

Variant	$\mathcal{A}$	$\mathcal{A}+\mathcal{DS}$	Δ
Vanilla Negative Sampling	76.82	66.17	-10.65
SCL-RAI	**76.44**	**71.24**	**-5.2**

Table 4: The robustness of SCL-RAI over unlabeled entities with different training set on EC dataset.

4 Conclusion

We propose the SCL-RAI to cope with *Unlabeled Entity Problem* in NER. Benefiting from our well-designed Span-based Contrastive Learning and Retrieval Augmented Inference, experiments on two real-world datasets show that SCL-RAI achieves more promising results than SOTA methods.

5 Acknowledgements

This paper is supported by the National Key R&D Program of China under Grand No.2018AAA0102003, the National Science Foundation of China under Grant No.61936012 and 61876004. Our code is now available at https://github.com/PKUnlp-icler/SCL-RAI.

References

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT 2019, Minneapolis, MN, USA, June 2-7, 2019, Volume 1 (Long and Short Papers)*, pages 4171–4186. Association for Computational Linguistics.

Jason A. Fries, Sen Wu, Alexander Ratner, and Christopher Ré. 2017. Swellshark: A generative model for biomedical named entity recognition without labeled data. *CoRR*, abs/1704.06360.

Zhanming Jie, Pengjun Xie, Wei Lu, Ruixue Ding, and Linlin Li. 2019. Better modeling of incomplete annotations for named entity recognition. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 729–734, Minneapolis, Minnesota. Association for Computational Linguistics.

Diederik P. Kingma and Jimmy Ba. 2015. Adam: A method for stochastic optimization. In *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*.

John D. Lafferty, Andrew McCallum, and Fernando C. N. Pereira. 2001. Conditional random fields: Probabilistic models for segmenting and labeling sequence data. In *Proceedings of the Eighteenth International Conference on Machine Learning (ICML 2001), Williams College, Williamstown, MA, USA, June 28 - July 1, 2001*, pages 282–289. Morgan Kaufmann.

Gina-Anne Levow. 2006. The third international Chinese language processing bakeoff: Word segmentation and named entity recognition. In *Proceedings of the Fifth SIGHAN Workshop on Chinese Language Processing*, pages 108–117, Sydney, Australia. Association for Computational Linguistics.

Changchun Li, Ximing Li, Lei Feng, and Jihong Ouyang. 2022a. Who is your right mixup partner in positive and unlabeled learning. In *International Conference on Learning Representations*.

Jingye Li, Hao Fei, Jiang Liu, Shengqiong Wu, Meishan Zhang, Chong Teng, Donghong Ji, and Fei Li. 2022b. Unified named entity recognition as word-word relation classification. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 36, pages 10965–10973.

Xiaoli Li and Bing Liu. 2005. Learning from positive and unlabeled examples with different data distributions. In *Machine Learning: ECML 2005, 16th European Conference on Machine Learning, Porto, Portugal, October 3-7, 2005, Proceedings*, volume 3720 of *Lecture Notes in Computer Science*, pages 218–229. Springer.

Yangming Li, Lemao Liu, and Shuming Shi. 2021. Empirical analysis of unlabeled entity problem in named entity recognition. In *9th International Conference on Learning Representations, ICLR 2021, Virtual Event, Austria, May 3-7, 2021*. OpenReview.net.

Yangming Li, Lemao Liu, and Shuming Shi. 2022c. Rethinking negative sampling for handling missing entity annotations. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 7188–7197, Dublin, Ireland. Association for Computational Linguistics.

Xiao Ling and Daniel S. Weld. 2012. Fine-grained entity recognition. In *Proceedings of the Twenty-Sixth AAAI Conference on Artificial Intelligence, July 22-26, 2012, Toronto, Ontario, Canada*. AAAI Press.

Minlong Peng, Xiaoyu Xing, Qi Zhang, Jinlan Fu, and Xuanjing Huang. 2019. Distantly supervised named entity recognition using positive-unlabeled learning. In *Proceedings of the 57th Conference of the Association for Computational Linguistics, ACL 2019, Florence, Italy, July 28- August 2, 2019, Volume 1: Long Papers*, pages 2409–2419. Association for Computational Linguistics.

Xiang Ren, Ahmed El-Kishky, Chi Wang, Fangbo Tao, Clare R. Voss, and Jiawei Han. 2015. Clustype: Effective entity recognition and typing by relation phrase-based clustering. In *Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, Sydney, NSW, Australia, August 10-13, 2015*, pages 995–1004. ACM.

Jingbo Shang, Jialu Liu, Meng Jiang, Xiang Ren, Clare R. Voss, and Jiawei Han. 2018. Automated phrase mining from massive text corpora. *IEEE Trans. Knowl. Data Eng.*, 30(10):1825–1837.

YaoSheng Yang, Wenliang Chen, Zhenghua Li, Zhengqiu He, and Min Zhang. 2018. Distantly supervised NER with partial annotation learning and reinforcement learning. In *Proceedings of the 27th International Conference on Computational Linguistics, COLING 2018, Santa Fe, New Mexico, USA, August 20-26, 2018*, pages 2159–2169. Association for Computational Linguistics.

Juntao Yu, Bernd Bohnet, and Massimo Poesio. 2020. Named entity recognition as dependency parsing. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 6470–6476, Online. Association for Computational Linguistics.

REDTab: A Relation Extraction Dataset for Knowledge Extraction from Web Tables

Siffi Singh [1], Alham Fikri Aji [2], Gaurav Singh [3,*], and Christos Christodoulopoulos [2]
[1]Amazon AWS AI, [2]Amazon Alexa AI
{siffis,afaji,chrchrs}@amazon.com
[3]Binance
gaurav.singh@binance.com

Abstract

Relational web-tables are significant sources of structural information that are widely used for relation extraction and population of facts into knowledge graphs. To transform the web-table data into knowledge, we need to identify the relations that exist between column pairs. Currently, there are only a handful of publicly available datasets with relations annotated against natural web-tables. Most datasets are constructed using synthetic tables that lack valuable metadata information, or are limited in size to be considered as a challenging evaluation set. In this paper, we present REDTab, the largest natural-table relation extraction dataset. We have annotated ~9K tables and ~22K column pairs using crowd sourced annotators from MTurk, which has 50x larger number of column pairs than the existing human-annotated benchmark. Our test set is specially designed to be challenging as observed in our experiment results using TaBERT.

1 Introduction

Web-tables contain a lot of knowledge (Cafarella et al., 2008; Wang and Hu, 2002) that can be utilized for various tasks such as question answering (Chakrabarti et al., 2020; Zhang et al., 2020), building knowledge graphs (Sekhavat et al., 2014; Wang et al., 2012), web-search (Sun et al., 2016; Kopliku et al., 2011), data-mining (Gatterbauer et al., 2007) and so on. Relation Extraction (RE) is one way to extract table information by capturing relations between two table columns (Figure 1). However, the current RE datasets available are either synthetic, or very small in size. In this paper, we propose a new human-annotated dataset for the evaluation of RE models.

The existing benchmark datasets for column pair RE are T2Dv2 and SemTab. However, T2Dv2 (Ritze and Bizer, 2017) only contains 236 tables and 435 column pairs annotated with 118 relations where about 50% of relations have only 1 column pair sample. The highest number of samples for a relation is only 28. Hence, due to its small size and less diversity in column pairs per relation, it is difficult to be treated as a reliable evaluation set. In contrast, the SemTab 2020 challenge (Jiménez-Ruiz et al., 2020a) provides a large benchmark dataset, but synthetically generated, which is arguably not representative of the real web-tables. Other table-based datasets (Limaye et al., 2010; Efthymiou et al., 2017; Zhang, 2017; Kacprzak et al., 2018) are designed for a binary relation extraction task.

REDTab[1] is a human annotated dataset consisting of 9,149 tables and 22,236 column pairs with 23 relations. The table was collected from Web Data Commons (WDC) (Lehmberg et al., 2016) relational tables for Music and Literature domains. These domains are two of most popular domains for question answering (Rajpurkar et al., 2016; Mihaylov et al., 2018; Kočiskỳ et al., 2018; Serban et al., 2016). The dataset also contains metadata such as table title, page title, and text before and after the table as extra context. The key differences between our proposed dataset and other benchmarks are shown in Table 1. We also provide benchmark results on REDTab using state-of-the-art table embedding approach called TaBERT (Yin et al., 2020). We get an F1-score of 0.58, which shows that our dataset is challenging and the performance on the test set can be further improved.

*Work done while at Amazon

[1]We publicly release the REDTab dataset at https://github.com/alexa/alexa-dataset-redtab.

Proceedings of the 29th International Conference on Computational Linguistics, pages 2319–2327
October 12–17, 2022.

Dataset	Tables	Col pairs	Relations	Annotation
T2Dv2	236	435	118	Human
SemTab	34,294	1,31,289	-	Synthetic
REDTab	**9,149**	**22,236**	23	**Human**

Table 1: Key Differences in REDTab over other RE benchmark datasets. The number of relations in SemTab is unknown as it is part of the blind test set and not provided by the organisers. Note that REDTab has 23 relations covering diverse set of column pairs for two domains.

2 Data Collection and Annotation

In this section, the construction of the REDTab dataset is described in detail. The discussion is divided into three subsections: Data Source, Annotation Protocol and Dataset Validation.

2.1 Data Source

REDTab dataset is constructed from Web Data Commons or WDC (Lehmberg et al., 2016) web-tables. Table selection is done by randomly sampling ~20k tables from WDC and manually identifying the suitable Music and Literature domain column headers. Based on these column headers, the tables for each domain are extracted from the entire WDC corpus. Approximately 5k tables are randomly sampled for both domains.

For each table, we also preserve metadata containing title, page title, as well as text before and after the table to provide more context. From the metadata information, we obtain the 'keyColumnIndex', that denotes the main column (referred to as Subject column). This Subject column will then be paired with other columns from the same table to form a column pair.

The relations in the column pairs are manually identified by inspecting the table headers. Our relation names hold more meaning than the relations from existing datasets. For example, instead of using "author", we use "is the author of", since it encodes more information than just "author". The full verb phrase construction encodes the argument order (Subject to the left, Object to the right), "Is" implies it is currently true, "the" implies uniqueness (compared to "is an author of"). It makes the relations far more human readable and interpretable without needing additional tooling to present the information to annotators. It also made the understanding connection to natural language questions easier with more naturalistic phrasing. The number of relations in our dataset is data-driven, this means, if we had encountered more relations in the data, we would have added them to the set of relations presented to annotators. Additionally, our dataset has higher number of relations per domain as compared to existing human-annotated datasets. Note that our dataset cannot be used to build a knowledge graph from scratch but rather to serve as an input to an information extraction system, that, with the right grounding (mapping), can also be used for knowledge graph construction. See Table 3 for examples of relation names. After filtering, there are 36 and 34 relevant relations in Music and Literature domains respectively.

[2] https://music.apple.com/ca/artist/jenny-silver/189196098/see-all?section=top-songs, accessed on Feb 16th 2022

2.2 Annotation Protocol

REDTab dataset is constructed by crowd source annotators from Amazon Mechanical Turk (MTurk) (Chen et al., 2011). The qualification requirement is set to locations of United States, United Kingdom, Australia, Ireland, Canada, and Singapore, and HIT Approval Rate of 98% with number of approved annotations as at-least 1000.

The annotation process is as follow. A table is shown to annotator. Then, the Subject column is paired with all other columns in that table. For each pair, the annotator must annotate the relation of the pair. The interface is shown in Figure 2. Let us consider an example, where <Name, Artist, Time, and Price> are the table columns and 'Name' is the Subject column. The annotators must annotate the relation between (Name - Artist), (Name - Time), and (Name - Price).

The table is displayed with the relation names specific to the table's domain. There is an additional 'UNKNOWN' relation that is provided to be selected in a scenario when the two column do not have a relation, or, when none of the given relations are applicable.

The annotator must also decide the directionality of the relation, i.e. whether it is 'Tinku' `<is a song performed by>` 'Holy Near & Inti Illimani', or 'Holy Near & Inti Illimani' `<is a song performed by>` 'Tinku' (as shown in Figure 2). Annotators are also provided with a space to leave their comments which has been useful to improve the guidelines and user experience.

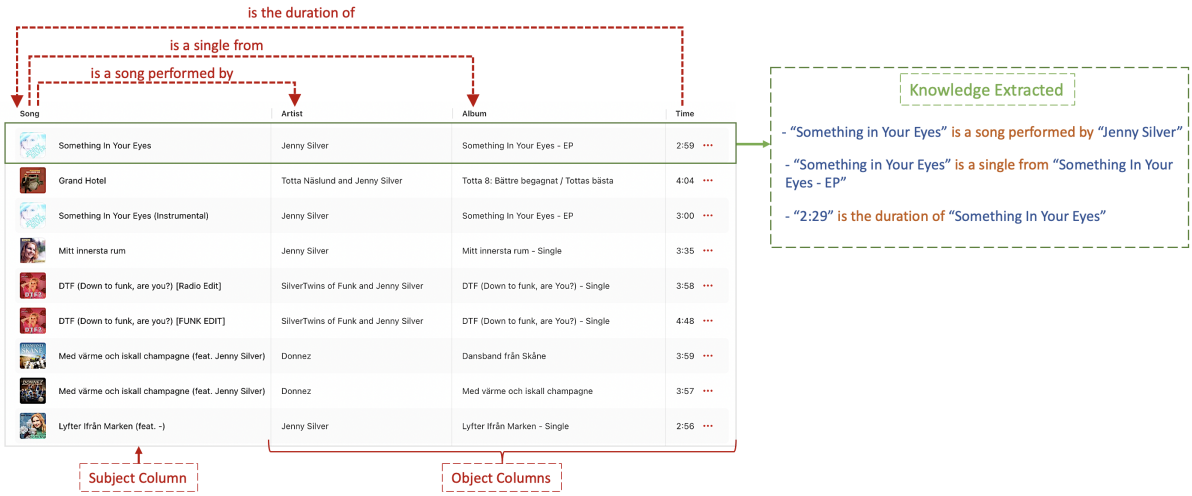

Figure 1: Example table[2] with columns 'Song', 'Artist', 'Album' and 'Time'. The main column, called the Subject column, is 'Song'. The relations between column pairs are labeled at the top. On the right, the extracted knowledge from one row of the table is shown.

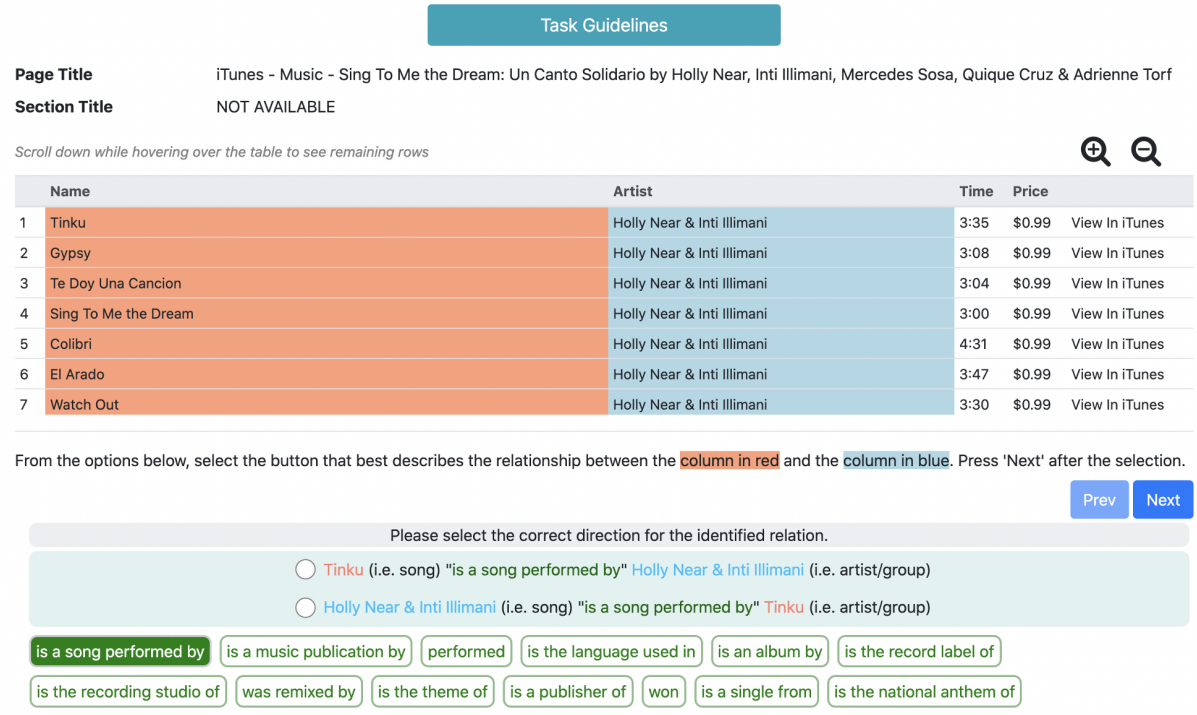

Figure 2: The annotation user interface for creating REDTab dataset. On top the 'Task Guidelines' are given. The 'Task Guidelines' is followed by page title, table title of the table, along with all the row values of table. A list of relations are provided, and upon selection of a relation (example shown 'is a song performed by'), the two options are provided showing the directionality of Subject column.

2.3 Dataset Validation

There are two main stages in the validation phase. The first stage is analysis of annotated data and the second stage is manual inspection to eliminate incorrect annotations.

Dataset Analysis: Annotations are carried out in several batches. Each table is annotated by 3 annotators. The first analysis is counting the number of annotated tables for each batch. This is followed by calculating the inter-annotator agreement using the Fleiss' kappa Score (McHugh, 2012). For Music,

the inter-annotator agreement is 0.75 whereas for Literature, the agreement is 0.40. The low agreement value in Literature is attributed to tables that are mistakenly identified into the Literature domain. For instance, 'Publisher' is a column header for tables in both Game and Literature domains. As 'Publisher' is also a column header used to filter tables as Literature, some of the tables from Game domain are mistakenly included during the filtering process. These tables create confusion for annotators as the remaining column pairs in the table cannot be identified with the given relations list.

In the final set, only those column pairs are included which have the same relation selected by at least two annotators. The co-occurrences of relations are then looked at to know which relation pairs are most commonly selected. For instance, 'is a publisher of' and 'is the record label of' have more than 50 co-occurrences in a batch of 1000 tables.

The annotated set consists of 10,284 tables and 39,514 column pairs. The number of selected relations is 52 as compared to 70 relations in the original set given to the annotators.

Dataset Cleanup: In this stage, the dataset is manually inspected. From the annotated tables, the header and subject column combination are collected. E.g. <Author, Title, Description, Date, Rank, 1>. This is called a 'template'. The given example is a table with five columns and a subject column of index '1'. Such a template is collected for all tables.

Considering these unique templates, a total of 1026 different sets are identified among 10,284 tables. The annotations for these tables are verified to have the correct relation and directionality. In the verification process, the contents of the table column are also taken into consideration. Using this approach, a mapping between a template and its corresponding annotation is created. The mapping is further used to update the annotations and create a clean dataset. The relations removed as part of cleanup are the ones which have incorrect annotations, or cannot be split into train-test set (see Section 3.1). The final dataset is created by collecting majority annotations, filtering the 'UNKNOWN' relation selections, and removing the incorrect annotations.

Dataset	# relations	# train	# test
REDTab-standard	23	4682	17554
REDTab-simple	10	4431	16176

Table 2: Statistics from the train and test set in REDTab Standard and REDTab simple dataset.

3 The Resulting Corpus

REDTab consists of 9,149 tables and 22,236 column-pairs annotated with 23 relations for Music and Literature domain. The overall time spent for annotation was ~300 hours and the total cost is ~6000 USD (i.e. 20 USD per hour). The number of tables in REDTab is an order of magnitude higher than T2Dv2 as shown in Table 1. Similarly, the number of column pairs in REDTab is two orders of magnitude higher than T2Dv2. Although, T2Dv2 has more relations, the number of column pair samples per relation is very small.

3.1 Train-Test Split

On both T2Dv2 and REDTab, it is observed that most of the time, the relation can be easily determined by looking at the column headers. For example, the column pair ('Title', 'Author') is annotated as `<is the author of>` all of the time. Only few column headers have some ambiguity to it. Therefore, if the train and test data is split randomly, the relation can be determined by simply memorizing the header. This hypothesis is tested and a simple header memorization is shown to achieve high accuracy without looking at any other context.

In order to construct a challenging test set that can be used to benchmark model's capability on understanding structured context, the training and testing data is split such that they do not share the same header pairs. For example, given a relation `<is a book about>`, the possible column header pairs are for example ('Title', 'Subjects'), ('Name', 'Description'), ('Title', 'Synopsis') etc. Data with ('Title', 'Subjects') are assigned as training whereas ('Name', 'Description') are assigned as testing. Specifically, for every relation, we group all data based on their header pair. However, some relations were annotated to only have 1 unique header pair, therefore these relations are excluded from the dataset.

3.2 REDTab dataset

Our final set consists of 23 relation classes, divided into training and testing split as shown in Table 3.

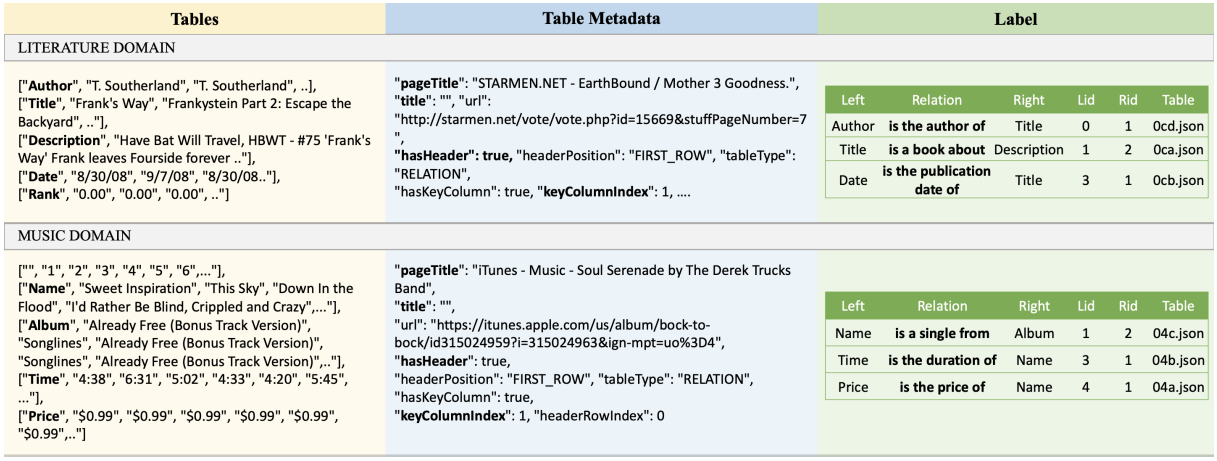

Figure 3: The first column shows the Table row data. The column headers are highlighted to show the column pairs. The second column is Table Metadata which includes 'keyColumnIndex' i.e. the index of the Subject column. In the first example, 'Title' is the Subject column with index 1. The third column shows the human-annotated relation label. The label has 'Left' i.e. left entity column, 'Relation' i.e. annotated relation and 'Right' i.e. right entity column. The result also contains the indexes of the two columns, direction information, and table filenames.

Note that some of the tail relation classes has minimal training or testing examples, which are the difficult samples. However, we argue that such cases reflect the real-world scenario. Therefore, we decide to release a simpler version of REDTab, which only focus on the top 10 most common relations. The statistics of our REDTab dataset can be seen in Table 2.

The train-test splits for both REDTab standard and simple category are publicly released for researchers to evaluate. An example annotation from REDTab can be seen in (Figure 3). The 'Left' and the 'Right' column is based on the directionality selected by annotators.

4 Experimental Setup

We use TaBERT (Yin et al., 2020) as a baseline in our dataset. TaBERT is a pre-trained language model that jointly learns representations for natural language sentences and tables. TaBERT represents each cell by the column name, column datatype and cell value, which is followed by using the Transformer (Vaswani et al., 2017; Devlin et al., 2018) model to generate row-level vectors. To allow for information flow across cells of different rows, TaBERT uses a vertical self-attention mechanism. TaBERT introduces two pre-training tasks, Masked Column Prediction and Cell Value Recovery to generate powerful contextualised representations. TaBERT has been tested only on semantic parsing tasks.

TaBERT model requires 4 inputs: Header, data, ID, and context. We feed the table header and row data as header and data respectively. The table title is assigned as the TaBERT ID, while the page title is assigned as the TaBERT context. We then concatenate the column embeddings for the current input, i.e. `concat(embedding of the left column, embedding of the right column)`. This is connected with a classification layer to predict the relation (see Figure 4).

Relation	# training	# testing
is a song performed by	1065	2702
is the author of	669	3322
is the price for	629	2197
is the duration of	549	2523
is the publication date of	427	2098
is the record label of	301	549
is a single from	271	1355
is a publisher of	202	430
is a book about	198	889
was written on	120	111
is an album by	81	624
is the composer of	47	17
is the genre of	42	175
is a narrative set in the location	39	248
is the isbn of	13	73
is the date of creation of	7	30
is the genre of music played by	5	3
is the number of pages in	4	35
is published as part of	4	39
is an edition of the published work	4	123
was released on an album at timepoint	2	3
is written in the language	2	7
wrote the lyrics to	1	1

Table 3: The relation names in REDTab standard dataset. We show the number of column pairs corresponding to each relation in the train and test set.

Model	REDTab-standard			REDTab-simple		
	P	R	F1	P	R	F1
Majority class	0.008	0.043	0.013	0.020	0.100	0.034
TaBERT base	0.628	0.621	0.580	0.876	0.900	0.883
TaBERT large	0.677	0.637	0.593	0.928	0.866	0.871

Table 4: Baseline performance showing macro precision, recall and F1-score, using REDTab dataset.

We experiment on both base and large TaBERT model. Our model is fine-tuned on 4 GPUs with a batch-size of 16 for 3 epochs. The model is trained with AdamW optimizer (Loshchilov and Hutter, 2019). Each experiment takes less than an hour to finish.

5 Result

5.1 Baseline

We start by exploring a TaBERT model as a baseline for both of our standard and simple REDTab set. Our baseline utilize all of the available context: table header, table data, and titles. We also introduce a "majority" baseline, where the model simply predicts the majority relation class all the time, which is "is the author of". The majority baseline is used as our lower-bound performance.

As shown in Table 4, our model achieved a F1 score of 0.58 on REDTab-standard, showing room for improvement for future studies on this dataset. Table 5 also shows the F1-score per relation. Better performance is achievable in REDTab-simple, since this set is designed to be easier compared to its standard counterpart. TaBERT large does not significantly outperform TaBERT base, despite using more computational resource. For further experiments, we use TaBERT base architecture.

5.2 Ablation Study

In this part, we perform an ablation study to see the effect of each data feature on performance. Specifically, we divide REDTab data into 3 contexts: 1) The header context, which includes the table header itself, 2) The row data context, which includes every row data of the table, and, 3) The title context, which includes the table and web page titles. This experiment uses REDTab-standard. Figure 4 illustrates the contexts.

Our baseline feeds all 3 contexts to TaBERT. Here, we remove one or more contexts by replacing it with empty strings before passing it to TaBERT. The experiment result can be seen in Table 6.

Header context is the most prominent feature.

Relation	P	R	F1
is a single from	0.95	1	0.97
is the duration of	0.83	0.98	0.9
is the price for	0.77	0.49	0.6
is the author of	0.74	0.88	0.8
is a book about	0.5	0	0
is the publication date of	0.98	1	0.99
is a publisher of	0.52	0.27	0.36
is a narrative set in the location	0.85	0.89	0.87
is a song performed by	1	0.98	0.99
is an album by	0.97	1	0.98
is the record label of	0.62	0.99	0.76
is an edition of- the published work	1	0.18	0.3
is the isbn of	0.97	0.99	0.98
is the genre of	0.7	0.71	0.71
was written on	0.99	0.66	0.79
is the composer of	0.89	0.94	0.91
is published as part of	0	0	0
is the date of creation of	0.01	0.2	0.02
is the number of pages in	0.51	1	0.68
is the genre of- music played by	0	0	0
is written in the language	0.58	1	0.74
was released on an album at timepoint	0	0	0
wrote the lyrics to	0.25	1	0.4

Table 5: Baseline performance showing precision, recall and F1-score per relation using TaBERT base on REDTab-standard dataset.

Our model can perform as good as the baseline only by looking at the table header (Header only). Similarly, removing the header significantly reduces the performance (No header). We argue that many table headers are representative of their content, therefore in most cases, it can be utilized to determine the relation of two columns without additional context.

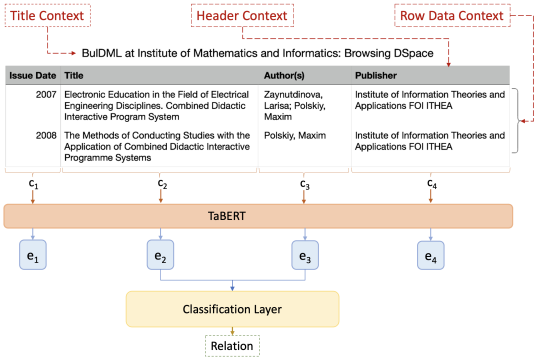

Figure 4: TaBERT for column-pair relation extraction. $\{c_1, c_2..\}$ are the columns, $\{e_1, e_2..\}$ are the TaBERT column embeddings for each column. The diagram shows title context, header context, and the row data content. Relation is classified between c_2 and c_3 in the table.

Config Name	Context			Performance		
	header	row-data	title	P	R	F1
Baseline	✓	✓	✓	0.628	0.621	0.580
Header only	✓			0.633	0.636	0.571
Row only		✓		0.417	0.318	0.290
Title only			✓	0.140	0.158	0.115
No header		✓	✓	0.392	0.367	0.336
No row	✓		✓	0.599	0.632	0.559
No title	✓	✓		0.664	0.644	0.615
Zero context				0.075	0.108	0.075

Table 6: Baseline results showing impact of metadata information on macro precision, recall and F1-scores for relation extraction using REDTab dataset.

We see a significant drop in performance (0.29 F1-score) when the model has to rely on row data alone. Therefore, it is challenging to understand a table data if we only see the data without additional context. Title context is the least prominent feature, as the model performed poorly (0.115 F1-score) if only the table and page titles are shown. In contrast, we gain performance over the baseline by removing the title context (No title), suggesting that the title feature might be too noisy if we have a better context such as table header or row data.

Interestingly, model with zero context (i.e. only see an empty table) can perform better than our lower-bound model (majority relation class). Therefore, the model is capable of guessing the relation even without any explicit context at all.

5.3 Column Ordering Bias

We also observe in our experiments that the table content can be indirectly inferred by the column location. For example, the first column is mostly expressing the main subject of the table, such as album name, whereas the later columns are mostly expressing the information of columns, such as price, or date. Therefore, the relation between two columns can be inferred by their position alone. For example, the relation between the 2nd and the 1st column is 'is a song performed by' 77% of the time, or the relation between the 3rd and the 1st column is 'is the price of' 68% of the time. We find that a model can exploit this implicit information, thus can gain some performance in zero context scenario.

To remove column ordering bias, we pre-process the data by shuffling the columns. As shown in Table 7. The zero-context model on shuffled columns has similar performance as majority.

Config Name	Performance		
	P	R	F1
Zero context	0.075	0.108	0.075
Zero context - shuffled column	0.018	0.052	0.025
Majority class	0.008	0.043	0.013

Table 7: Zero context performance on REDTab dataset by shuffling the column and removing the column ordering bias.

5.4 Discussion and Future Work

In REDTab, we see that some relations only have few examples, it is in fact reflective of the real-world scenario where there are column pairs which occur more frequently with one another, while some column pairs are uncommon and hence fewer tables contain these samples. Our baseline system is not handling these tail relation cases well, demonstrated by the low F1 score in REDTab-standard scenario. For future, we should investigate a model, that is capable of handling such tail cases. We have also observed that row data in tables alone cannot give good performance, and use of metadata improves results. In our experiments, we have explored the use of metadata such as headers, table title and page title. In future, we can explore models that can make effective use of other metadata information present with tables in the dataset (e.g. text before table) which might show further improvement in performance. In future, we want to add relation mappings to public knowledge graphs, expand REDTab dataset to include several other domains and cover diverse set of tables with variety of column pairs.

6 Related Work

We briefly describe most major benchmark datasets that contain relation annotations for web-tables.

SemTab Challenge

The SemTab challenge was launched to encourage comparisons between different automated table alignment techniques over a benchmark dataset. In 2019, SemTab (Jiménez-Ruiz et al., 2020a) released 3 tasks, namely: 1) Column-Type Annotation (CTA), 2) Cell-Entity Annotation (CEA), and 3) Columns-Property Annotation (CPA). They build their dataset by automatically generating labelled tables from facts stored in the the DBpedia (Auer et al., 2007) Knowledge Graph (KG). Despite its large size, the dataset consists of synthetically generated tables, which suffer from draw-

back such as: lack of associated metadata, noisiness, heterogeneity, incompleteness and ambiguity in the data generation process. These drawbacks are not unique to synthetic tables, but can cause them to significantly diverge from human generated web-tables. In 2020, they released a new dataset called Tough Table (2T) dataset (Jiménez-Ruiz et al., 2020b). It consists of a mix of small and large catalog-like tables to make the dataset more challenging. However, unlike other datasets, it also consists of real web-tables. Unfortunately though, it does not contain annotations for the relation extraction task (CPA). Also, a very small number of real web-tables are contained in this dataset, while the majority of other tables are still synthetic.

TURL

TURL (Deng et al., 2020) consists of Wikipedia tables annotated via distant supervision based on overlap of facts with Freebase KG (Bollacker et al., 2008). However, our own investigations revealed that these relations and tables were too easy, and therefore did not represent the real task of relation extraction from web-tables. In addition to our investigations, the model in the paper achieves extremely high F1 scores for the task, further evidence that the tables and relations are simple. Also, the method can not be used to create a larger dataset of more tables and relations, since the overlapping relations have already been exploited. Additionally, it would be harder to find direct overlaps of unpopular relations or facts in the Freebase KG.

T2Dv2

T2Dv2 (Ritze and Bizer, 2017) is a dataset of real web-tables manually labelled against DBpedia KG. To the best of our knowledge, it is the only other publicly available real web-tables dataset that was manually labelled. Of all the other web-tables datasets, this one is the closest to ours because it was manually created and consists of real web-tables. However, its one drawback is that it consists of only 236 web-tables with annotated relation. In contrast, our dataset consists of over 9K manually annotated web-tables. Therefore, our dataset is the largest publicly available and manually annotated dataset.

Additionally, there are other table alignment datasets created in the past (Limaye et al., 2010; Efthymiou et al., 2017). But these datasets either consists of very few tables or the tables were annotated using automated distant supervision techniques, making them unreliable for task evaluation. For example, Limaye et al. consists of 400 manually annotated Web-tables with entity, class, and property-level correspondences, where single cells are mapped to entities and Efthymiou et al. only mapped entities in tables to DBpedia KG.

7 Conclusion

We present REDTab, the largest natural web-tables dataset for column pair relation extraction. The dataset is annotated by crowd sourced annotators from MTurk. REDTab includes more than 9K tables and 22K column pairs making it the largest human-annotated relation extraction dataset to our knowledge. We evaluate state-of-the art table embedding model TaBERT and find that our dataset is challenging over the strategically created train-test split set. Our final set contains more diverse set of column pairs per relation that is ideal for testing models. Furthermore, our analysis shows that predicting relations solely based on table row data is challenging and use of some metadata information significantly improves the performance. We expect this dataset can contribute to facilitate further progress in the field of information extraction research.

References

Sören Auer, Christian Bizer, Georgi Kobilarov, Jens Lehmann, Richard Cyganiak, and Zachary Ives. 2007. Dbpedia: A nucleus for a web of open data. In *The semantic web*, pages 722–735. Springer.

Kurt Bollacker, Colin Evans, Praveen Paritosh, Tim Sturge, and Jamie Taylor. 2008. Freebase: a collaboratively created graph database for structuring human knowledge. In *Proceedings of the 2008 ACM SIGMOD international conference on Management of data*, pages 1247–1250.

Michael J Cafarella, Alon Halevy, Daisy Zhe Wang, Eugene Wu, and Yang Zhang. 2008. Webtables: exploring the power of tables on the web. *Proceedings of the VLDB Endowment*, 1(1):538–549.

Kaushik Chakrabarti, Zhimin Chen, Siamak Shakeri, and Guihong Cao. 2020. Open domain question answering using web tables. *arXiv preprint arXiv:2001.03272*.

Jenny J Chen, Natala J Menezes, Adam D Bradley, and T North. 2011. Opportunities for crowdsourcing research on amazon mechanical turk. *Interfaces*, 5(3):1.

Xiang Deng, Huan Sun, Alyssa Lees, You Wu, and Cong Yu. 2020. Turl: Table understanding

through representation learning. *arXiv preprint arXiv:2006.14806*.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*.

Vasilis Efthymiou, Oktie Hassanzadeh, Mariano Rodriguez-Muro, and Vassilis Christophides. 2017. Matching web tables with knowledge base entities: from entity lookups to entity embeddings. In *International Semantic Web Conference*, pages 260–277. Springer.

Wolfgang Gatterbauer, Paul Bohunsky, Marcus Herzog, Bernhard Krüpl, and Bernhard Pollak. 2007. Towards domain-independent information extraction from web tables. In *Proceedings of the 16th international conference on World Wide Web*, pages 71–80.

Ernesto Jiménez-Ruiz, Oktie Hassanzadeh, Vasilis Efthymiou, Jiaoyan Chen, and Kavitha Srinivas. 2020a. Semtab 2019: Resources to benchmark tabular data to knowledge graph matching systems. In *European Semantic Web Conference*, pages 514–530. Springer.

Ernesto Jiménez-Ruiz, Oktie Hassanzadeh, Vasilis Efthymiou, Jiaoyan Chen, Kavitha Srinivas, and Vincenzo Cutrona. 2020b. Results of semtab 2020. In *CEUR Workshop Proceedings*, volume 2775, pages 1–8.

Emilia Kacprzak, José M Giménez-García, Alessandro Piscopo, Laura Koesten, Luis-Daniel Ibáñez, Jeni Tennison, and Elena Simperl. 2018. Making sense of numerical data-semantic labelling of web tables. In *European Knowledge Acquisition Workshop*, pages 163–178. Springer.

Tomáš Kočiskỳ, Jonathan Schwarz, Phil Blunsom, Chris Dyer, Karl Moritz Hermann, Gábor Melis, and Edward Grefenstette. 2018. The narrativeqa reading comprehension challenge. *Transactions of the Association for Computational Linguistics*, 6:317–328.

Arlind Kopliku, Karen Pinel-Sauvagnat, and Mohand Boughanem. 2011. Attribute retrieval from relational web tables. In *International Symposium on String Processing and Information Retrieval*, pages 117–128. Springer.

Oliver Lehmberg, Dominique Ritze, Robert Meusel, and Christian Bizer. 2016. A large public corpus of web tables containing time and context metadata. In *Proceedings of the 25th International Conference Companion on World Wide Web*, pages 75–76.

Girija Limaye, Sunita Sarawagi, and Soumen Chakrabarti. 2010. Annotating and searching web tables using entities, types and relationships. *Proceedings of the VLDB Endowment*, 3(1-2):1338–1347.

Ilya Loshchilov and Frank Hutter. 2019. Decoupled weight decay regularization. In *ICLR*.

Mary L McHugh. 2012. Interrater reliability: the kappa statistic. *Biochemia medica*, 22(3):276–282.

Todor Mihaylov, Peter Clark, Tushar Khot, and Ashish Sabharwal. 2018. Can a suit of armor conduct electricity? a new dataset for open book question answering. *arXiv preprint arXiv:1809.02789*.

Pranav Rajpurkar, Jian Zhang, Konstantin Lopyrev, and Percy Liang. 2016. Squad: 100,000+ questions for machine comprehension of text. *arXiv preprint arXiv:1606.05250*.

Dominique Ritze and Christian Bizer. 2017. Matching web tables to dbpedia-a feature utility study. In *EDBT*.

Yoones A Sekhavat, Francesco Di Paolo, Denilson Barbosa, and Paolo Merialdo. 2014. Knowledge base augmentation using tabular data. In *LDOW*.

Iulian Vlad Serban, Alberto García-Durán, Caglar Gulcehre, Sungjin Ahn, Sarath Chandar, Aaron Courville, and Yoshua Bengio. 2016. Generating factoid questions with recurrent neural networks: The 30m factoid question-answer corpus. *arXiv preprint arXiv:1603.06807*.

Huan Sun, Hao Ma, Xiaodong He, Wen-tau Yih, Yu Su, and Xifeng Yan. 2016. Table cell search for question answering. In *Proceedings of the 25th International Conference on World Wide Web*, pages 771–782.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. *Advances in neural information processing systems*, 30.

Jingjing Wang, Haixun Wang, Zhongyuan Wang, and Kenny Q Zhu. 2012. Understanding tables on the web. In *International Conference on Conceptual Modeling*, pages 141–155. Springer.

Yalin Wang and Jianying Hu. 2002. Detecting tables in html documents. In *International Workshop on Document Analysis Systems*, pages 249–260. Springer.

Pengcheng Yin, Graham Neubig, Wen-tau Yih, and Sebastian Riedel. 2020. Tabert: Pretraining for joint understanding of textual and tabular data. *arXiv preprint arXiv:2005.08314*.

Xingyao Zhang, Linjun Shou, Jian Pei, Ming Gong, Lijie Wen, and Daxin Jiang. 2020. A graph representation of semi-structured data for web question answering. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 51–61.

Ziqi Zhang. 2017. Effective and efficient semantic table interpretation using tableminer+. *Semantic Web*, 8(6):921–957.

Automatic Keyphrase Generation by Incorporating Dual Copy Mechanisms in Sequence-to-Sequence Learning

Siyu Wang[1], Jianhui Jiang[1], Yao Huang[1] and Yin Wang[1,2]
[1]Gusu Laboratory of Materials, Suzhou, China
[2]Hongzhiwei Technology (Shanghai) Co., Ltd, Shanghai, China
{wangsiyu2022, jiangjianhui2021, huangyao2021}@gusulab.ac.cn
wangyin@hzwtech.com

Abstract

The keyphrase generation task is a challenging work that aims to generate a set of keyphrases for a piece of text. Many previous studies based on the sequence-to-sequence model were used to generate keyphrases, and they introduce a copy mechanism to achieve good results. However, we observed that most of the keyphrases are composed of some important words (seed words) in the source text, and if these words can be identified accurately and copied to create more keyphrases, the performance of the model might be improved. To address this challenge, we propose a DualCopyNet model, which introduces an additional sequence labeling layer for identifying seed words, and further copies the words for generating new keyphrases by dual copy mechanisms. Experimental results demonstrate that our model outperforms the baseline models and achieves an obvious performance improvement.

1 Introduction

A keyphrase is a short piece of text that summarizes and abstracts the main semantics of a long text (named "document" or "source text" in this study). High-quality keyphrase promotes readers to efficiently understand, summarize and access documents' content (Meng et al., 2017). Not only that, extracting high-quality keyphrase had been widely applied to many downstream tasks in Natural Language Processing and Data Mining, such as Information Retrieval (Jones and Staveley, 1999), Text Summarization (Zhang et al., 2004), Text Categorization (Hulth and Megyesi, 2006) and Opinion Expression Mining (Berend, 2011). Thus, how to automatically extract high-quality keyphrases has become popular research topics in recent decades (Augenstein et al., 2017; Kim et al., 2010). Due to accessibility of text data, many datasets from scientific articles are used as benchmarks for keyphrase extraction algorithms. Thus, our study also focuses on extracting keyphrases from the scientific articles.

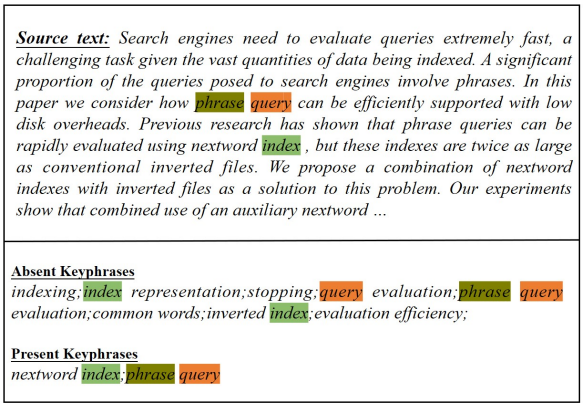

Figure 1: An example of keyphrases. The colored words (seed words) in the source text appear repeatedly in keyphrases.

Generally speaking, keyphrases can be divided into two categories: ***present keyphrase*** and ***absent keyphrase***. Present keyphrases are the explicit words that appear directly in source text, and vice versa for absent keyphrase. Many previous studies have focused on how to extract present keyphrases from documents. These studies (Campos et al., 2020; Hulth, 2003; Bougouin et al., 2014; Boudin, 2018; Bennani-Smires et al., 2018) consider the keyphrases extraction as a ranking task, which extracts a set of candidate phrases from the source text, and then selects keyphrases from the sorted candidates with the higher importance score. In recent years, some studies have also attempted to use deep learning methods for present keyphrase extraction. For example, Alzaidy et al. (2019) considered the present keyphrases extraction as a sequence labeling task by using LSTM-CRF model to label sequence, and obtained a better performance. Sun et al. (2021) used popular BERT (Devlin et al., 2019) model to extract present keyphrases. However, these methods were not expert in extracting absent keyphrases because the

Proceedings of the 29th International Conference on Computational Linguistics, pages 2328–2338
October 12–17, 2022.

source text has no absent keyphrase. To solve this problem, some sequence-to-sequence (Sutskever et al., 2014) based models were used to generate present/absent keyphrases, such as (Yuan et al., 2020; Chen et al., 2018; Meng et al., 2017; Chen et al., 2019; Ye et al., 2021). They first encoded the source text and then dynamically outputted corresponding present/absent keyphrases through a decoder. However, the decoder usually generates high-frequency words and ignores many out-of-vocabulary words, so it is unsatisfied in the task. To address this problem, Meng et al. (2017) incorporated a copy mechanism (Gu et al., 2016) in decoder to successfully predict out-of-vocabulary (OOV) words, namely OOV words copy mechanism. Besides, in many scientific articles, we observe that some words repeatedly appear in many keyphrases. For example, as shown in Figure 1, such as "*index*" appears in keyphrases "*nextword index*, *index representation*, *inverted index*", and "*query*" appears in keyphrases "*query evaluation*, *phrase query*" (these words are called seed words in this study). Therefore, if these seed words from source text can be identified and applied by the decoder to generate keyphrases as much as possible, the performance of the model will be greatly improved.

To address this challenge, we propose a novel sequence-to-sequence model (named DualCopyNet) to incorporate dual copy mechanisms for generating present/absent keyphrases. Since there is no labeled data, it is very difficult to directly extract the seed words from the source text. Thus, we try to extract present keyphrases as seed words. Specifically, besides a canonical encoder layer in DualCopyNet, we also introduce a sequence labeling layer for extracting present keyphrases (seed words). In addition, in the decoder layer, we introduce two kinds of copy mechanism. **Seed words copy mechanism**, it enables the decoder to generate phrases by selecting appropriate words from seed words. **OOV words copy mechanism**, it is a feasible solution that enables the decoder to predict OOV words by selecting appropriate words from the source text (Meng et al., 2017). Moreover, the decoder of DualCopyNet softly fuses the dual copy probability and generation probability through a gate mechanism to copy words (seed or OOV words) from the source text and generate words from the vocabulary. When training the model, we use a multi-task learning approach to optimize the primary task (generating keyphrases) and the auxiliary task (predicting seed words). Finally, we conduct experiments on four datasets. The results show that DualCopyNet has an obvious performance improvement in predicting present/absent keyphrases. The contributions of our paper are as follows:

- We introduce sequence labeling layer in the sequence-to-sequence architecture for predicting seed words and dynamically copy these words to generate more keyphrases.
- We design a novel decoder that incorporates dual copy mechanisms and uses a multi-task learning approach to optimize the model when generating present and absent keyphrases.
- On the four experimental datasets, our model outperforms most of the baseline models and obtains better results. Meanwhile, we demonstrate the positive effect of the dual copy mechanism by ablation study.

2 Related Work

2.1 Keyphrase Extraction

Many previous works (Hulth, 2003; Boudin, 2018; Witten et al., 1999; Bougouin et al., 2014) have been focusing on the study of keyphrases extraction. Generally, the extraction consists of two main steps: (1) Identifying candidate phrases by special hand-crafted rules (Hulth, 2003; Medelyan et al., 2009). (2) Sorting the candidate phrases to obtain keyphrases. For example, (Boudin, 2018; Bougouin et al., 2014; Campos et al., 2020; Mihalcea and Tarau, 2004) used an unsupervised approach to rank candidates. In recent years, some studies used a supervised approach for ranking, such as (Sun et al., 2021). And they achieved good results by introducing the BERT (Devlin et al., 2019) model. In addition, some studies considered keyphrase extraction as a sequence labeling task (Alzaidy et al., 2019). Although extraction-based methods obtained good results, they lacked an ability to predict absent keyphrases.

2.2 Keyphrase Generation

Due to previous methods' drawbacks for predicting absent keyphrases, Meng et al. (2017) first proposed a CopyRNN model to generate words from vocabulary and copy words from the source

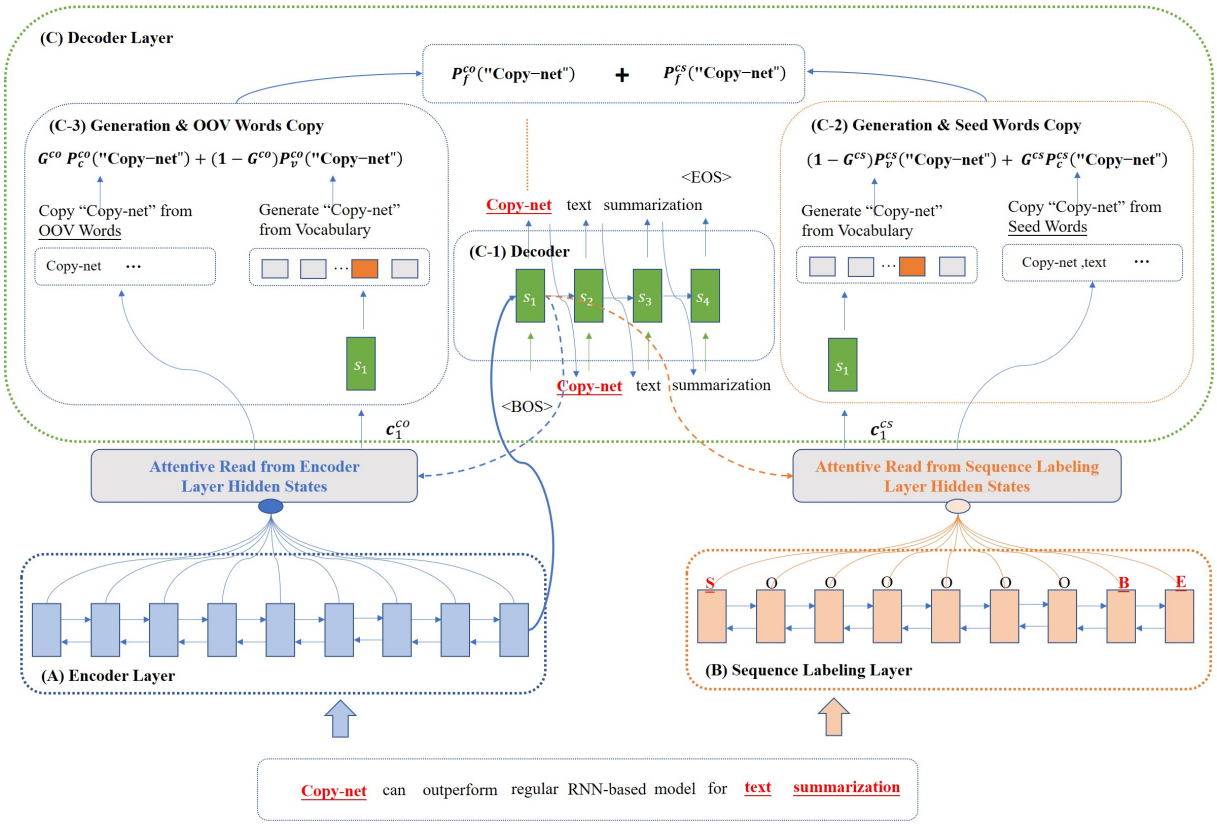

Figure 2: The overall structure of DualCopyNet, which includes three parts: **Encoder Layer**, **Sequence Labeling Layer** and **Decoder Layer**.

text. Subsequent studies have made many improvements on the basis of CopyRNN. (1) For problem with few training samples in certain domains, Ye and Wang (2018) proposed semi-supervised keyphrase generation method to leverage labeled data and large-scale unlabeled samples for learning. (2) Many studies ignored relationships among keyphrases, therefore Chen et al. (2018) proposed a new sequence-to-sequence architecture to capture correlation among keyphrases for generating new keyphrases. (3) Another problem is that the current researches ignored the leading role of the title. And Chen et al. (2019) realized the problem, then proposed a novel model named TGNet for keyphrase generation. (4) Different source texts should contain different number of keyphrases. Therefore, Yuan et al. (2020) proposed a recurrent generative model to generate multiple keyphrases by delimiter-separated sequences. (5) Keyphrases are inherently a disordered set rather than an ordered sequence, so Ye et al. (2021) proposed a new training paradigm ONE2SET to concatenate keyphrases without a predefined order.

In recent years, some new technologies have also been applied for keyphrase generation task, such as Reinforcement Learning (Chan et al., 2019) and Generative Adversarial Networks (GANs) (Swaminathan et al., 2020).

3 Methodology

3.1 Problem Definition

Given a keyphrase dataset that contains N data samples, and the ith is denoted as $(x^{(i)}, p^{(i)})$, where $x^{(i)}$ is a source text, $p^{(i)}$ is a set of keyphrases. $p^{(i)}$ contains M_i keyphrases and denotes as $p^{(i)} = (p^{(i,1)}, p^{(i,2)}, \ldots, p^{(i,M_i)})$, where $p^{(i,j)}$ is one of $p^{(i)}$. $x^{(i)}$ and $p^{(i,j)}$ are word sequences:

$$x^{(i)} = (x_1^{(i)}, x_2^{(i)}, \ldots, x_{l_{x^{(i)}}}^{(i)}), \quad (1)$$

$$p^{(i,j)} = (y_1^{(i,j)}, y_2^{(i,j)}, \ldots, y_{l_{p^{(i,j)}}}^{(i,j)}), \quad (2)$$

where $l_x^{(i)}$ and $l_p^{(i,j)}$ is length of $x^{(i)}$ and $p^{(i,j)}$ respectively. The keyphrase generation task is to generate a set of keyphrase $p^{(i)}$ from the source text $x^{(i)}$, namely maximizes the probability $\prod_{i=1}^{N} \prod_{j=1}^{M_i} P(p^{(i,j)}|x^{(i)})$.

3.2 DualCopyNet Architecture

In this section, we will introduce the proposed DualCopyNet in detail. The model is based on

the sequence-to-sequence framework (Sutskever et al., 2014) and the copy mechanism (Gu et al., 2016). The structure shown in Figure 2 includes three parts: **Encoder Layer**, **Sequence Labeling Layer** and **Decoder Layer**.

Specifically, we first feed the source text into an encoder layer and a sequence labeling layer to obtain corresponding contextual representations. Then, we send the representations to a decoder layer to produce keyphrases. For a better generation, we introduce a dual copy mechanisms namely seed words copy mechanism and OOV words copy mechanism in the decoder layer. After that, the decoder dynamically generates words from the vocabulary or copies useful words from the source text.

Encoder Layer. To better generate contextual representations from a text, DualCopyNet adopts bi-directional GRUs (Cho et al., 2014) to encode the source text. And the text is composed of word embedding, which is defined as follows:

$$\boldsymbol{X} = (\boldsymbol{w}_1, \boldsymbol{w}_2, \ldots, \boldsymbol{w}_n), \tag{3}$$

where $\boldsymbol{w}_i \in \mathbb{R}^{u_1}$ denotes the word embedding of the ith word in the source text. Let u_1 be the dimension of the word embedding and n be the length of the source text. Then, $\boldsymbol{X} \in \mathbb{R}^{n \times u_1}$ is sent into the encoder layer, we employ bi-directional GRUs (Bi-GRU) to read the text sequence from two directions and output the hidden state of each word as follows:

$$\begin{aligned} \overrightarrow{\boldsymbol{u}_i} &= \overrightarrow{GRU}(\boldsymbol{w}_i, \boldsymbol{u}_{i-1}), \\ \overleftarrow{\boldsymbol{u}_i} &= \overleftarrow{GRU}(\boldsymbol{w}_i, \boldsymbol{u}_{i+1}). \end{aligned} \tag{4}$$

We then concatenate $\overrightarrow{\boldsymbol{u}_i} \in \mathbb{R}^{u_2}$ and $\overleftarrow{\boldsymbol{u}_i} \in \mathbb{R}^{u_2}$ to get the hidden state $\boldsymbol{u}_i$ of the ith word, whose length is $2u_2$ and computed as follows:

$$\boldsymbol{u}_i = [\overrightarrow{\boldsymbol{u}_i}; \overleftarrow{\boldsymbol{u}_i}]. \tag{5}$$

Sequence Labeling Layer. As mentioned above, we observe that some seed words repeatedly appear in many keyphrases. Therefore, we expect to identify these seed words first and then dynamically copy them into the output of the decoder. Since there is no labeled data, it is very difficult to directly extract the seed words from the source text. Therefore, in this study, we consider identified present keyphrases as seed words and introduce a sequence labeling layer based on LSTM-CRF (Huang et al., 2015) in DualCopyNet to extract these seed words. Specifically, we first input the word embedding $\boldsymbol{x}_i \in \mathbb{R}^{u_1}$ at time step i into the bidirectional LSTM (Hochreiter and Schmidhuber, 1997) to obtain the hidden state $\boldsymbol{v}_i = [\overrightarrow{\boldsymbol{v}_i}; \overleftarrow{\boldsymbol{v}_i}]$ and derive the emission potential from $\boldsymbol{v}_i$. Meanwhile, an additional Conditional Random Field (CRF) (Lafferty et al., 2001) layer is employed to calculate the most probable tag for the each word. Then we use the BIESO scheme (*Begin, Intermediate, End, Single, Other*) to identify the seed words in the texts. For given a source text x, the conditional probability of a target tag^* is computed by:

$$P(tag^*|x) = \frac{\exp^{score(x, tag^*)}}{\sum_{\widetilde{tag}} \exp^{score(x, \widetilde{tag})}}, \tag{6}$$

where the function $score$ is defined as:

$$\begin{aligned} score(x, tag) = \sum_i log\phi_{emit}(i, tag_i) + \\ log\phi_{trans}(tag_{i-1} \rightarrow tag_i), \end{aligned} \tag{7}$$

where $\phi_{trans}(tag_{i-1} \rightarrow tag_i)$ is the transition score from tag_{i-1} to tag_i. $\phi_{emit}(i, tag_i)$ is the score of the tag_i for the ith input word and comes from the hidden state of the Bi-LSTM at timestep i.

Finally, the loss function of the sequence labeling layer is defined as:

$$\mathcal{L}^s = -log(P(tag^*|x)). \tag{8}$$

Decoder Layer. The source text has been encoded into two kinds of contextual representations through an encoder layer and a sequence labeling layer, respectively. Further, we adopt the decoder based on the attention mechanism to fuse the dual copy mechanisms to generate keyphrases. The decoder is created with one-way GRU. For each time step t, the GRU fuses the hidden state $\boldsymbol{s}_{t-1}$ and the word embedding $\boldsymbol{e}_{t-1}$ of the output word y_{t-1}, which is computed by:

$$\boldsymbol{s}_t = GRU(\boldsymbol{e}_{t-1}, \boldsymbol{s}_{t-1}), \tag{9}$$

where $t-1$ denotes previous time step, $\boldsymbol{e}_0$ is the embedding of the start token '<BOS>'.

Generation and Seed Words Copy. As mentioned above, the sequence labeling layer is adopted to predict seed words. Then we expect to dynamically copy these words into the output sequence for generating the keyphrases. Specifically,

we first use the decoder vector $\widetilde{\boldsymbol{s}_t^{cs}}$ to generate the next word, where $\widetilde{\boldsymbol{s}_t^{cs}} = [\boldsymbol{c}_t^{cs}; \boldsymbol{s}_t]$ is derived from concatenating $\boldsymbol{c}_t^{cs}$ and $\boldsymbol{s}_t$, and the context vector $\boldsymbol{c}_t^{cs}$ is computed as a weighted sum of hidden representations by the concatenate attention mechanism (Luong et al., 2015):

$$\boldsymbol{c}_t^{cs} = attn(\boldsymbol{s}_t, [\boldsymbol{v}_1, \boldsymbol{v}_2, \cdots \boldsymbol{v}_n], \boldsymbol{W}_{att}^{cs}). \quad (10)$$

Then, the probability of generating the word $P_v^{cs}(y_t)$ from vocabulary is computed by:

$$P_v^{cs}(y_t|y_{t-1}, x) = softmax(\boldsymbol{W}_{v2}(\boldsymbol{W}_{v1}\widetilde{\boldsymbol{s}_t^{cs}} + b_{v1}) + b_{v2}). \quad (11)$$

Furthermore, a copy mechanism (Gu et al., 2016) is adopted to efficiently extract the in-text information and strengthen the generation capability of our model. We first calculate a soft gate G_t^{cs} to dynamical select the way of output, that is whether generating from the vocabulary or copying from the seed words at time step t, which is defined as:

$$G_t^{cs} = sigmoid(\boldsymbol{W}_g^{cs}\boldsymbol{m}_t^{cs} + b_g^{cs}), \quad (12)$$

where $\boldsymbol{m}_t^{cs} = [\boldsymbol{e}_{t-1}; \widetilde{\boldsymbol{s}_t^{cs}}]$ is the concatenation of the embedding of previous output word $y_{(t-1)}$ and $\widetilde{\boldsymbol{s}_t^{cs}}$. Then, the probability of predicting a word y_t by copying from seed words or generating from vocabulary is defined as:

$$P_f^{cs}(y_t) = G_t^{cs}P_c^{cs}(y_t) + (1 - G_t^{cs})P_v^{cs}(y_t), \quad (13)$$

where $P_c^{cs}(y_t) = \sum_{i:x_i=y_t} a_{t,i}^{cs}$ is a probability of seed word copy for y_t . And $\sum_{i:x_i=y_t} a_{t,i}^{cs}$ is the normalized attention weight between $\boldsymbol{s}_t$ and sequence labeling layer hidden state $\boldsymbol{v}_i$.

Generation and OOV Words Copy. The probability of generating the word $P_v^{co}(y_t)$ for current step is computed by:

$$P_v^{co}(y_t|y_{t-1}, x) = softmax(\boldsymbol{W}_{v4}(\boldsymbol{W}_{v3}\widetilde{\boldsymbol{s}_t^{co}} + b_{v3}) + b_{v4}), \quad (14)$$

where $\widetilde{\boldsymbol{s}_t^{co}} = [\boldsymbol{c}_t^{co}; \boldsymbol{s}_t]$ and the context vector $\boldsymbol{c}_t^{co}$ is computed by:

$$\boldsymbol{c}_t^{co} = attn(\boldsymbol{s}_t, [\boldsymbol{u}_1, \boldsymbol{u}_2, \cdots \boldsymbol{u}_n], \boldsymbol{W}_{att}^{co}). \quad (15)$$

Then, the second soft gate G_t^{co} is computed by:

$$G_t^{co} = sigmoid(\boldsymbol{W}_g^{co}\boldsymbol{m}_t^{co} + b_g^{co}), \quad (16)$$

where $\boldsymbol{m}_t^{co} = [\boldsymbol{e}_{t-1}; \widetilde{\boldsymbol{s}_t^{co}}]$. Eventually, the probability of predicting a word y_t by copying from OOV words or generating from vocabulary is computed by:

$$P_f^{co}(y_t) = G_t^{co}P_c^{co}(y_t) + (1 - G_t^{co})P_v^{co}(y_t), \quad (17)$$

where $P_c^{co}(y_t) = \sum_{i:x_i=y_t} a_{t,i}^{co}$ is a probability of OOV word copy for y_t. And $\sum_{i:x_i=y_t} a_{t,i}^{co}$ is the normalized attention score between $\boldsymbol{s}_t$ and encoder hidden state $\boldsymbol{u}_i$.

In the end, the final probability distribution of predicting a word y_t is computed by summation of $P_f^{cs}(y_t)$ and $P_f^{co}(y_t)$:

$$P_f(y_t) = P_f^{cs}(y_t) + P_f^{co}(y_t). \quad (18)$$

Dataset	|Samples|	#KP	%A-KP
Inspec	500	9.79	26.42
NUS	211	10.81	45.36
SemEval	100	14.43	55.61
KP20k	20000	5.26	37.23

Table 1: Statistics for the four testing datasets. |Samples|: the number of samples on the dataset, #KP: the avg number of keyphrases, %A-KP: the proportion of absent keyphrase.

3.3 Training Loss

DualCopyNet is based on sequence-to-sequence structure and involves two tasks: keyphrase generation and sequence labeling. Thus, the loss function contains two parts: the sequence labeling layer introduces an additional CRF loss (equation 8), and the decoder layer adopts the negative log likelihood (NLL) loss, which is defined as:

$$\mathcal{L}^g = -\sum_{t=1}^{L_y} logP_f(y_t|y_{t-1}, x, \theta). \quad (19)$$

We define the overall loss function with the CRF loss and the NLL loss:

$$\mathcal{L} = \mathcal{L}^s + \mathcal{L}^g. \quad (20)$$

The loss is calculated as the average over mini batch. Finally, we use Adam (Kingma and Ba, 2014) to optimize the model.

4 Experimental Settings

In this section, we will describe the training and testing sets used for the experiments, then introduce the baseline models and evaluation metrics.

4.1 Dataset

We choose the largest public dataset KP20k (Meng et al., 2017) for training models, it contains a large number of high-quality academic papers mainly from the computer science field. The dataset has 527,830 articles for training and 20000 articles for validation.

Furthermore, we evaluate our model on four testing datasets widely adopted in previous works (Chen et al., 2018; Yuan et al., 2020; Chen et al., 2019; Ye et al., 2021; Swaminathan et al., 2020; Chen et al., 2020), including Inspect (Hulth, 2003), NUS (Nguyen and Kan, 2007), KP20k (Meng et al., 2017) and SemEval (Kim et al., 2010). Table 1 summarizes the statistics of each testing dataset.

4.2 Baselines

In our experiments, we choose the following keyphrase generation models as the baselines:

- catSeq (Yuan et al., 2020): An RNN-based attentional encoder-decoder model with copy mechanism.
- catSeqD (Yuan et al., 2020): catSeq augmented with orthogonal regularization and semantic coverage mechanism.
- catSeqCorr (Chen et al., 2018): a new sequence-to-sequence architecture for keyphrases generation, which captures correlation among multiple keyphrases in two ways.
- catSeqTG (Chen et al., 2019): a new sequence-to-sequence architecture for keyphrases generation, which explicitly considers the leading role of the title to the overall document main body.
- catSeq-RF (Chan et al., 2019): An extension of catSeq with RL-based finetuning, which introduces an adaptive reward function and encourages the model to generate both sufficient and accurate keyphrases.
- GAN-mr (Swaminathan et al., 2020): A novel model for keyphrase generation approach using Generative Adversarial Networks (GANs).
- ExHiRD-h (Chen et al., 2020): An exclusive hierarchical decoding model with a hard exclusion mechanism.

4.3 Implementation Details

The models catSeq, catSeqD, catSeqCorr, catSeqTG and catSeq-RF are implemented by (Chan et al., 2019). Followed by (Yuan et al., 2020; Chen et al., 2020; Swaminathan et al., 2020; Chan et al., 2019), when training our model, the ground-truth keyphrase sequence is the concatenation of present and absent keyphrases. Then the present keyphrases are sorted according to the initial orders arose in the document, and the absent keyphrases keep their original orders. Furthermore, we replace all digits with the symbol <digit> and define the vocabulary V with the most frequent words numbered 50,000.

The embedding size and hidden size of GRU, LSTM are set to 150; The batch size is 64 and learning rate is 0.0001; The gradient clipping is set to 1 and dropout is set to 0.1. The hyper-parameters are tuned on validation set. Early stopping is applied when the validation loss stops dropping three continuous evaluations. During testing, we set the maximum depth of the beam search as 6 and the beam size as 200. While on KP20k dataset, due to the large amount of test data, we set the beam size as 20. We implement the model using Pytorch (Paszke et al., 2019) and train the model using NVIDIA 3090TI and Ubuntu System.

4.4 Evaluation Metrics

Same as previous work (Chen et al., 2019; Swaminathan et al., 2020; Chan et al., 2019; Ye et al., 2021; Chen et al., 2020), we adopt the macro-averaged F1@5 and F1@M as the evaluation metrics. F1@M compares all keyphrases predicted by the model with the ground-truth to compute the F1 score. And F1@5 compares top 5 keyphrases predicted by the model with the ground-truth. Specifically, when the number of predictions is less than five, F1@5 will be the same as F1@M, so we must randomly append incorrect keyphrases to fill five predictions instead of directly using the original predictions. Furthermore, we also apply Porter Stemmer for preprocessing before comparisons.

5 Results and Analysis

5.1 Keyphrases Prediction

In this section, we will evaluate the performance of the model in predicting present keyphrase and absent keyphrase separately.

The performances of predicting present keyphrase are shown in Table 2. As we can see

Model	Inspec		NUS		SemEval		KP20k	
	F1@M	F1@5	F1@M	F1@5	F1@M	F1@5	F1@M	F1@5
catSeq	0.262	0.225	0.397	0.323	0.283	0.242	0.367	0.291
catSeqD	0.263	0.219	0.394	0.321	0.274	0.233	0.363	0.285
catSeqCorr	0.269	0.227	0.390	0.319	0.290	0.246	0.365	0.289
catSeqTG	0.270	0.229	0.393	0.325	0.290	0.246	0.366	0.292
catSeq-RF	0.300	0.250	**0.426**	0.364	0.327	0.285	**0.383**	0.310
GAN-mr	0.299	0.258	0.417	0.348	\	\	0.378	0.303
ExHiRD-h	0.291	0.253	\	\	0.335	0.284	0.374	0.311
DualCopyNet	**0.342**	**0.284**	0.395	**0.379**	**0.339**	**0.315**	0.337	**0.312**

Table 2: F1 of present keyphrases prediction on four datasets. The best/second results in each column are highlighted with bold/underline.

Model	Inspec		NUS		SemEval		KP20k	
	F1@M	F1@5	F1@M	F1@5	F1@M	F1@5	F1@M	F1@5
catSeq	0.008	0.004	0.028	0.016	0.028	0.020	0.032	0.015
catSeqD	0.011	0.007	0.024	0.014	0.024	0.016	0.031	0.015
catSeqCorr	0.009	0.005	0.024	0.014	0.026	0.018	0.032	0.015
catSeqTG	0.011	0.005	0.018	0.011	0.027	0.019	0.032	0.015
catSeq-RF	0.017	0.009	0.031	0.019	0.027	0.018	**0.047**	0.024
GAN-mr	0.019	**0.013**	0.038	0.026	\	\	0.045	**0.032**
ExHiRD-h	**0.022**	0.011	\	\	0.025	0.017	0.032	0.016
DualCopyNet	0.014	0.012	**0.055**	**0.038**	**0.029**	**0.023**	0.042	0.025

Table 3: F1 of absent keyphrases prediction on four datasets. The best/second results in each column are highlighted with bold/underline.

that DualCopyNet greatly outperforms the whole baseline models on F1@5, especially on Inspec, NUS and SemEval datasets, but there is only a slight improvement on the KP20k dataset. For F1@M, DualCopyNet also achieves the best results on the Inspec and SemEval datasets, outperforms all models, which demonstrates the effectiveness of our method.

Predicting absent keyphrases is a challenging task. As shown in Table 3, we can see that all models are poor in predicting absent keyphrases comparing to predicting present keyphrases. In this task, DualCopyNet achieves better performance on the NUS and SemEval datasets and outperforms all baseline models. While, the performance is slightly lower than some baseline models on Inspect and KP20k datasets.

Overall, the advantage of our model is more obvious on datasets containing more target keyphrases, such as NUS, SemEval and Inspec. Because the more keyphrases, the greater the number of seed words included, and the model can achieve a greater performance improvement. In contrast, as can be seen from Table 1, The average number of keyphrases on KP20k is much lower than the other three datasets. Therefore, our model does not achieve the best results on KP20k dataset.

5.2 Ablation Study

We conduct an ablation study to further analyze dual copy mechanisms and multi-task learning. First, we introduce three variants of DualCopyNet:

- SeqLabelingNet: We remove the decoder layer and only keep the sequence labeling layer.
- OOVCopyNet: We remove the seed words copy mechanism in the decoder layer, and only keep the OOV words copy mechanism.
- DualCopyNet^{nll}: This model is same as DualCopyNet. But we only use the negative log-likelihood loss (Equation 19) to optimize the model when training phase.

The present keyphrases prediction results of the ablation study are shown in Table 4. After adding the seed words copy mechanism, the performance of the models (DualCopyNet and DualCopyNet)

Model	Inspec			NUS			SemEval			KP20k		
	P	R	F1	P	R	F1	P	R	F1	P	R	F1
SeqLabelingNet	0.275	0.075	0.118	0.446	0.169	0.245	0.389	0.142	0.208	0.349	0.222	0.271
OOVCopyNet	0.407	0.273	0.327	0.353	0.402	0.375	0.285	0.353	0.315	0.258	0.473	0.334
DualCopyNet^{nll}	0.419	0.276	0.332	0.351	0.435	0.388	0.280	0.370	0.319	0.259	0.485	**0.338**
DualCopyNet	0.403	0.298	**0.342**	0.348	0.458	**0.395**	0.303	0.384	**0.339**	0.258	0.485	0.337

Table 4: Ablation study on the four datasets for present keyphrases predication. The P denotes Precision@M, the R denote Recall@M and the F1 denote F1@M. The best/second results in each column are highlighted with bold/underline.

Model	Inspec			NUS			SemEval			KP20k		
	P	R	F1	P	R	F1	P	R	F1	P	R	F1
OOVCopyNet	0.008	0.012	0.010	0.041	0.032	0.036	0.045	0.013	0.020	0.038	0.036	0.037
DualCopyNet^{nll}	0.013	0.019	**0.015**	0.048	0.039	0.043	0.049	0.025	**0.033**	0.040	0.044	**0.042**
DualCopyNet	0.012	0.015	0.014	0.063	0.048	**0.055**	0.062	0.019	0.029	0.040	0.044	**0.042**

Table 5: Ablation study on the four datasets for absent keyphrase predication. The best/second results in each column are highlighted with bold/underline.

on four datasets has been obviously improved, especially in Recall, which means the model can recall more keyphrases through the seed words copy mechanism, and this is also consistent with our expectations. Due to the KP20k dataset only contains few keyphrases, so there is only a slight improvement on F1. In addition, the precision of SeqLabelingNet achieves the best results on most of datasets, so it is reasonable and effective to identify present keyphrases as seed words and copy them into the decoder.

Since SeqLabelingNet cannot predict the absent keyphrase, there only remains three models. The experimental results are shown in Table 5. In the prediction of absent keyphrases, the models (DualCopyNet^{nll} and DualCopyNet) have obviously improved in Recall, Precision and F1 after adding the seed words copy mechanism. Furthermore, the models DualCopyNet and DualCopyNet^{nll} have the same structure, but DualCopyNet employs multi-task learning when training phase. As can be seen from Table 4 and Table 5, F1 of DualCopyNet outperforms DualCopyNet^{nll} on most of datasets. It proves that multi-task learning can effectively improve the performance of the model.

5.3 Ability to Generate Diverse Keyphrases

To investigate the model's ability of generating diverse keyphrases, we adopt NDCG (Wang et al., 2013) to evaluate models. NDCG is used to evaluate the diversity of generated text, which is widely

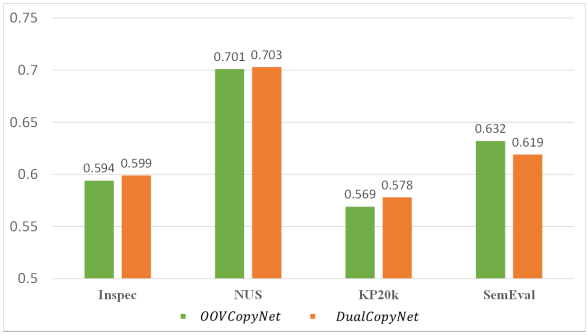

Figure 3: NDCG@10 metrics on four datasets.

used in text generation tasks (Habibi and Popescu-Belis, 2013) and information retrieval (Santos et al., 2013) tasks. The higher NDCG means the more diverse content that the model can generate. It works by penalizing redundant keyphrases and rewarding new keyphrases.

The results are summarized in Figure 3. Compared to OOVCopyNet, DualCopyNet achieves better NDCG@10 on three of the four datasets. Especially, there is an obvious improvement on the KP20k and Inspec datasets. It proves that the introduction of the seed words copy mechanism not only improves the performance of the model, but also generates more diverse phrases.

5.4 Case Study

In Figure 4, we show examples of keyphrases generated by SeqLabelingNet, OOVCopyNet and DualCopyNet respectively. After inputting the title and the abstract, we can see that DualCopyNet

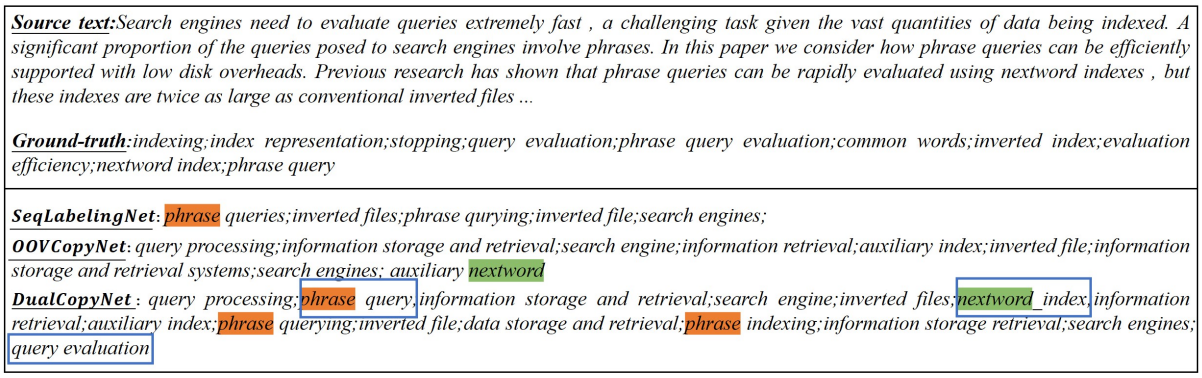

Figure 4: Case study. The keyphrases in the rectangle are truth keyphrases. Words remarked with green indicate copied from OOV words. Words remarked with orange indicate copied from seed words.

can generate more diverse keyphrases, the quantity even exceeds the ground-truth. Not only that, these generated keyphrases are generally reasonable and typical. Next, we can find that DualCopyNet copies the seed words "***phrase***" predicted by the sequence labeling layer to generate some new keyphrases, such as "***phrase indexing***". But the keyphrases generated by OOVCopyNet contain none of the above words, which proves the effectiveness of the seed words copy mechanism. On the other hand, DualCopyNet also generates keyphrases copying from OOV word "***nextword***". Finally, through the case study, it is proved that our model can well integrate the two copy mechanisms and effectively improve the performance of generating keyphrases.

6 Conclusions

In this paper, we propose a novel DualCopyNet for keyphrases generation. Based on the phenomenon of that many keyphrases are composed of seed words in the source text, we design dual copy mechanisms to precisely copy seed words and OOV words from the source text. Furthermore, aiming to obtain seed words, we introduce an additional sequence labeling layer and train the model with a multi-task learning. Finally, the experiments conducted on multiple datasets show our model's achievements are higher than most of baselines. Meanwhile, ablation experiments show a positive effect of the seed word copy mechanism and multi-task learning.

References

Rabah Alzaidy, Cornelia Caragea, and C Lee Giles. 2019. Bi-lstm-crf sequence labeling for keyphrase extraction from scholarly documents. In *The world wide web conference*, pages 2551–2557.

Isabelle Augenstein, Mrinal Das, Sebastian Riedel, Lakshmi Vikraman, and Andrew McCallum. 2017. Semeval 2017 task 10: Scienceie-extracting keyphrases and relations from scientific publications. In *Proceedings of the 11th International Workshop on Semantic Evaluation (SemEval-2017)*, pages 546–555.

Kamil Bennani-Smires, Claudiu Musat, Andreea Hossmann, Michael Baeriswyl, and Martin Jaggi. 2018. Simple unsupervised keyphrase extraction using sentence embeddings. In *Proceedings of the 22nd Conference on Computational Natural Language Learning*, pages 221–229.

Gábor Berend. 2011. Opinion expression mining by exploiting keyphrase extraction. In *Proceedings of 5th International Joint Conference on Natural Language Processing*, pages 1162–1170.

Florian Boudin. 2018. Unsupervised keyphrase extraction with multipartite graphs. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 2 (Short Papers)*, pages 667–672.

A Bougouin, F Boudin, and B Daille. 2014. Topicrank: Topic ranking for automatic keyphrase extraction. *Revue Traitement Automatique des Langues*, 55(1):4569.

Ricardo Campos, Vítor Mangaravite, Arian Pasquali, Alípio Jorge, Célia Nunes, and Adam Jatowt. 2020. Yake! keyword extraction from single documents using multiple local features. *Information Sciences*, 509:257–289.

Hou Pong Chan, Wang Chen, Lu Wang, and Irwin King. 2019. Neural keyphrase generation via reinforcement learning with adaptive rewards. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 2163–2174.

Jun Chen, Xiaoming Zhang, Yu Wu, Zhao Yan, and Zhoujun Li. 2018. Keyphrase generation with correlation constraints. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 4057–4066.

Wang Chen, Hou Pong Chan, Piji Li, and Irwin King. 2020. Exclusive hierarchical decoding for deep keyphrase generation. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 1095–1105.

Wang Chen, Yifan Gao, Jiani Zhang, Irwin King, and Michael R Lyu. 2019. Title-guided encoding for keyphrase generation. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, pages 6268–6275.

Kyunghyun Cho, Bart van Merriënboer, Dzmitry Bahdanau, and Yoshua Bengio. 2014. On the properties of neural machine translation: Encoder–decoder approaches. In *Proceedings of SSST-8, Eighth Workshop on Syntax, Semantics and Structure in Statistical Translation*, pages 103–111.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186.

Jiatao Gu, Zhengdong Lu, Hang Li, and Victor OK Li. 2016. Incorporating copying mechanism in sequence-to-sequence learning. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1631–1640.

Maryam Habibi and Andrei Popescu-Belis. 2013. Diverse keyword extraction from conversations. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 651–657.

Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long short-term memory. *Neural computation*, 9(8):1735–1780.

Zhiheng Huang, Wei Xu, and Kai Yu. 2015. Bidirectional lstm-crf models for sequence tagging. *arXiv preprint arXiv:1508.01991*.

Anette Hulth. 2003. Improved automatic keyword extraction given more linguistic knowledge. In *Proceedings of the 2003 conference on Empirical methods in natural language processing*, pages 216–223.

Anette Hulth and Beáta Megyesi. 2006. A study on automatically extracted keywords in text categorization. In *Proceedings of the 21st International Conference on Computational Linguistics and 44th Annual Meeting of the Association for Computational Linguistics*, pages 537–544.

Steve Jones and Mark S Staveley. 1999. Phrasier: a system for interactive document retrieval using keyphrases. In *Proceedings of the 22nd annual international ACM SIGIR conference on Research and development in information retrieval*, pages 160–167.

Su Nam Kim, Olena Medelyan, Min-Yen Kan, and Timothy Baldwin. 2010. SemEval-2010 task 5 : Automatic keyphrase extraction from scientific articles. In *Proceedings of the 5th International Workshop on Semantic Evaluation*, pages 21–26.

Diederik P Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*.

John Lafferty, Andrew McCallum, and Fernando CN Pereira. 2001. Conditional random fields: Probabilistic models for segmenting and labeling sequence data.

Minh-Thang Luong, Hieu Pham, and Christopher D Manning. 2015. Effective approaches to attention-based neural machine translation. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 1412–1421.

Olena Medelyan, Eibe Frank, and Ian H Witten. 2009. Human-competitive tagging using automatic keyphrase extraction. In *Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing*, pages 1318–1327.

Rui Meng, Sanqiang Zhao, Shuguang Han, Daqing He, Peter Brusilovsky, and Yu Chi. 2017. Deep keyphrase generation. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 582–592.

Rada Mihalcea and Paul Tarau. 2004. Textrank: Bringing order into text. In *Proceedings of the 2004 conference on empirical methods in natural language processing*, pages 404–411.

Thuy Dung Nguyen and Min-Yen Kan. 2007. Keyphrase extraction in scientific publications. In *International conference on Asian digital libraries*, pages 317–326.

Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, et al. 2019. Pytorch: An imperative style, high-performance deep learning library. *Advances in neural information processing systems*, 32.

Rodrygo LT Santos, Pablo Castells, Ismail Sengor Altingovde, and Fazli Can. 2013. Diversity and novelty in information retrieval. In *Proceedings of the 36th international ACM SIGIR conference on Research and development in information retrieval*, pages 1130–1130.

Si Sun, Zhenghao Liu, Chenyan Xiong, Zhiyuan Liu, and Jie Bao. 2021. Capturing global informativeness in open domain keyphrase extraction. In *CCF International Conference on Natural Language Processing and Chinese Computing*, pages 275–287.

Ilya Sutskever, Oriol Vinyals, and Quoc V. Le. 2014. Sequence to sequence learning with neural networks. In *Proceedings of the 27th International Conference*

on Neural Information Processing Systems - Volume 2, page 3104–3112.

Avinash Swaminathan, Haimin Zhang, Debanjan Mahata, Rakesh Gosangi, Rajiv Shah, and Amanda Stent. 2020. A preliminary exploration of gans for keyphrase generation. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 8021–8030.

Yining Wang, Liwei Wang, Yuanzhi Li, Di He, and Tie-Yan Liu. 2013. A theoretical analysis of ndcg type ranking measures. In *Conference on learning theory*, pages 25–54.

Ian H Witten, Gordon W Paynter, Eibe Frank, Carl Gutwin, and Craig G Nevill-Manning. 1999. Kea: Practical automatic keyphrase extraction. In *Proceedings of the fourth ACM conference on Digital libraries*, pages 254–255.

Hai Ye and Lu Wang. 2018. Semi-supervised learning for neural keyphrase generation. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 4142–4153.

Jiacheng Ye, Tao Gui, Yichao Luo, Yige Xu, and Qi Zhang. 2021. One2set: Generating diverse keyphrases as a set. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 4598–4608.

Xingdi Yuan, Tong Wang, Rui Meng, Khushboo Thaker, Peter Brusilovsky, Daqing He, and Adam Trischler. 2020. One size does not fit all: Generating and evaluating variable number of keyphrases. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7961–7975.

Yongzheng Zhang, Nur Zincir-Heywood, and Evangelos Milios. 2004. World wide web site summarization. *Web intelligence and agent systems: an international journal*, 2(1):39–53.

Dependency-aware Prototype Learning for Few-shot Relation Classification

Tianshu Yu [1,2], **Min Yang** [1]*, **Xiaoyan Zhao** [1]
[1] Shenzhen Institute of Advanced Technology, Chinese Academy of Sciences
[2] University of Chinese Academy of Sciences
ts.yu@siat.ac.cn, min.yang@siat.ac.cn, xy.zhao@siat.ac.cn

Abstract

Few-shot relation classification aims to classify the relation type between two given entities in a sentence by training with a few labeled instances for each relation. However, most of existing models fail to distinguish multiple relations that co-exist in one sentence. This paper presents a novel dependency-aware prototype learning (DAPL) method for few-shot relation classification. Concretely, we utilize dependency trees and shortest dependency paths (SDP) as structural information to complement the contextualized representations of input sentences by using the dependency-aware embedding as attention inputs to learn attentive sentence representations. In addition, we introduce a gate controlled update mechanism to update the dependency-aware representations according to the output of each network layer. Extensive experiments on the FewRel dataset show that DAPL achieves substantially better performance than strong baselines. For reproducibility, we will release our code and data upon the publication of this paper at https://github.com/publicstaticvo/DAPL.

1 Introduction

Relation classification, which aims to classify the relation between two entities in a sentence, is a fundamental task for information retrieval (Kadry and Dietz, 2017), knowledge graph construction (Shen et al., 2020; Ji et al., 2021) and question answering (Luo et al., 2018). Most of existing relation classification methods (Wang et al., 2016; Guo et al., 2019; Shen et al., 2020; Tian et al., 2021; Zhao et al., 2022a) focus on the supervised scenario where sufficient labeled training data is available. However, it is time-consuming and labor-intensive to collect large-scale labeled data in many real-world applications, especially in the low-resource settings (Geng et al., 2019, 2020; Fan et al., 2021; Zhao et al., 2022b).

Recently, few-shot relation classification (FSRC), which explores relation extraction methods by training with a few labeled examples in each relation, has become a hot research topic (Gao et al., 2019; Qu et al., 2020; Gao et al., 2020; Wang et al., 2020; Xu and Xiang, 2021; Ding et al., 2021; Fan et al., 2021). For instance, Han et al. (2018) introduce a large-scale FSRC dataset and implement several well-known few-shot learning techniques (Finn et al., 2017; Snell et al., 2017) for FSRC. Qu et al. (2020) propose a Bayesian meta learning approach for FSRC, which learns the posterior distributions of prototype vectors among different relations.

Despite the remarkable progress of FSRC methods, there is still a technical challenge which is not addressed well in prior work. Specifically, there can be multiple relations that co-exist in a sentence, while only one relation corresponds to the given entity pairs. The other existed relations may mislead the classifier to the wrong relation class, which is called the *misleading relation*. Taking Figure 1 as an example, the gold relation between two target entities "*Mitsubishi toppo*" and "*minica*" is "*derivative-model*" marked by the term "*derived from*", while most prior FSRC methods incorrectly predict the misleading relation "*products-producer*" marked by the term "*produced by*".

One possible solution is to leverage the dependency tree as auxiliary information to facilitate the representation learning. Recently, several studies have incorporated dependency tree into supervised relation classification models and obtained significant performance improvement (Sun et al., 2020; Yu et al., 2020; Pouran Ben Veyseh et al., 2020; Chen et al., 2021; Fan et al., 2021; Tian et al., 2021). However, few studies investigate the effectiveness of dependency trees in FSRC task. In addition, most existing works either solely focus on the terms that have direct dependency with target entities or involve redundant information by using

*Min Yang is corresponding author.

Proceedings of the 29th International Conference on Computational Linguistics, pages 2339–2345
October 12–17, 2022.

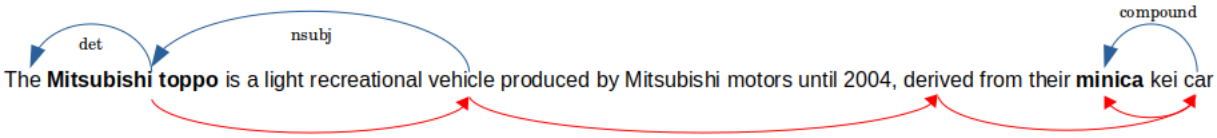

Figure 1: An example from the test set of FewRel. Previous models only focus on the dependency tree in blue color and ignore the SDP in red color which entails the ground-truth relation.

the entire dependency tree, failing to get other information such as shortest dependency paths (SDP) of two entities thus cannot tackle the misleading relation problem. For example, as illustrated in Figure 1, the SDP (marked as red lines) of the two entities can help the relation classification model obtain the correct relation "*derived from*". Therefore, it is necessary to fully exploit dependency information as auxiliary structural information, which can help identify useful terms and misleading terms via their relative positions to the given entities.

In this paper, we propose a novel dependency-aware prototype learning (DAPL) method for FSRC. DAPL is based on the framework of prototypical networks (Snell et al., 2017) with the BERT (Devlin et al., 2018) encoder, motivated by the effectiveness of prototypical networks in few-shot classification tasks. In our method, we leverage dependency trees as structural information to complement the contextualized representations of input sentences. Specifically, we assign each input token with a dependency label, according to whether the token is adjacent to the target entities or on the SDP between the two target entities. We highlight the tokens on the SDP by assigning a unified sdp dependency label for each token. Then, we convert these dependency labels into dependency embeddings, which are used as attention inputs of the contextualized sentence representations to learn dependency-aware sentence representations. Furthermore, we introduce a gate-controlled update mechanism to update the dependency-aware representations based on the output of each BERT layer, inspired by the effectiveness of the gate update functions in GRU (Cho et al., 2014). This mechanism provides a feedback to dependency states about whether they are reflecting the importance of each token and related to the ground truth labels.

The main contributions of this work are threefold:

- We propose a novel dependency-aware prototype learning method for FSRC, which fully exploit the dependency and contextualized information to alleviate the misleading relation problem and improve the overall performance of FSRC.
- We present a gate-controlled update mechanism to adaptively adjust the dependency-aware representations according to the output of each network layer.
- Experiments on a benchmark FSRC dataset (i.e., FewRel) show that our method outperforms the strong baselines by a noticeable margin.

2 Methodology

Problem Definition In the RC task, each instance consists of an input sequence x (including a input sentence z, a head entity e_1, a tail entity e_2) and a relation label y for the two entities. We adopt a typical N-way-K-shot setting for FSRC (Qu et al., 2020). Under N-way-K-shot configuration, the training data is further split into a support set $\mathcal{S}$ and a query set $\mathcal{Q}$ which have disjoint labels, where $\mathcal{S}$ contains N relation classes and each with K labeled examples. The goal of FSRC is to learn a model using D_{train}, which is then used to predict the relation y for each input x in testing set.

2.1 Dependency Labels

Given an input example $x \in D_{\text{train}}$, we denote its dependency tree as $\mathcal{G} = (V, E)$, where V contains the tokens in the sentence and E contains the set of edges (dependencies) of tokens. Each triplet $(w_i, w_j, d) \in E$ denotes that there is a dependency of type d between tokens w_i and w_j in x. Note that $\mathcal{G}$ is an undirected graph. Given the head entity e_1 and the tail entity e_2, we denote the set of all tokens on the SDP between e_1 and e_2 except themselves as P. We assign two dependency labels $l_i^{(1)}$ and $l_i^{(2)}$ to each token w_i of the sentence x, where $l_i^{(1)}$ and $l_i^{(2)}$ denote the dependency relations between the token w_i to the head entity and the tail entity respectively by the following four steps:

1. We initialize the $l_i^{(1)}$ and $l_i^{(2)}$ labels of each token as $none$.

2. The $l_i^{(1)}$ label of e_1 and the $l_i^{(2)}$ label of e_2 are set to $self$.

3. For each token $w_i \in P$ on SDP except e_1 and e_2, we set its $l_i^{(1)}$ and $l_i^{(2)}$ labels as sdp.

4. For each token $w_i \notin P$ that is not on SDP, we set $l_i^{(1)}$ to the corresponding dependency parsing type if $l_i^{(1)}$ is $none$ and w_i has an edge connected to e_1 on the dependency tree. We can get the $l_i^{(2)}$ label for e_2 in a similar way.

To better illustrate the above process, we take the sentence "*[CLS] the school <e1> master </e1> teaches the lesson with a <e2> steak </e2> [SEP]*" as an example. We show how the two labels of each token are obtained as follows:

1. We initialize the $l_i^{(1)}$ and $l_i^{(2)}$ labels of each token as $none$.

2. The $l_i^{(1)}$ labels of "<e1>", "master", "</e1>" and the $l_i^{(2)}$ labels of "<e2>", "steak", "</e2>" are assigned with $self$.

3. The dependency path between the two entities (i.e., "master" and "steak") is "master-teaches-steak", so both $l_i^{(1)}$ and $l_i^{(2)}$ labels of "teaches" are set as sdp.

4. For the remaining tokens, "the" and "school" are adjacent to "master" on the dependency tree, so the $l_i^{(1)}$ label of "the" is det, and the $l_i^{(1)}$ label of "school" is $compound$. Meanwhile, "with" and "a" are adjacent to "steak", so the $l_i^{(2)}$ label of "with" is $case$, and the $l_i^{(2)}$ label of "a" is det.

Afterwards, we use an embedding layer to convert the dependency labels $l_i^{(1)}$ and $l_i^{(2)}$ into dependency embeddings $\mathbf{d}_i^{(1)}$ and $\mathbf{d}_i^{(2)}$ with an embedding dimension of $d_h/2$, where d_h is the hidden vector size of the encoder. The dependency embedding $\mathbf{d}_i$ of each token w_i is formed by concatenating $\mathbf{d}_i^{(1)}$ and $\mathbf{d}_i^{(2)}$ together, whose length is d_h.

2.2 Dependency-aware Attention

Figure 2 shows the structure of our model DAPL. Our model takes each token representation $\{\mathbf{w}_i\}$ and dependency embedding $\{\mathbf{d}_i\}$ in the sentence as input. Inspired by the remarkable success of pre-trained language models (PLMs) on most of NLP tasks, we employ BERT (Devlin et al., 2018) as the basic framework of our model. To learn the importance of each token to the given entities, we modify the self-attention mechanism in original BERT by adding together the contextual representation and dependency representation when generating query and key matrices at the l-th layer:

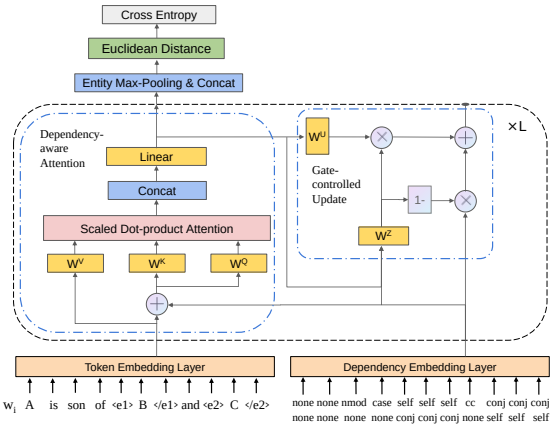

Figure 2: The overall structure of our DAPL.

$$Q^{(l)} = (\mathbf{h}_i^{(l-1)} + \mathbf{d}_i^{(l-1)})W_Q^{(l)} \quad (1)$$

$$K^{(l)} = (\mathbf{h}_i^{(l-1)} + \mathbf{d}_i^{(l-1)})W_K^{(l)} \quad (2)$$

$$V^{(l)} = \mathbf{h}_i^{(l-1)}W_V^{(l)} \quad (3)$$

$$\tilde{\mathbf{h}}^{(l)} = \text{softmax}\left(\frac{Q^{(l)}K^{(l)T}}{\sqrt{d_K}}\right)V^{(l)} \quad (4)$$

where $W_Q^{(l)}, W_K^{(l)}, W_V^{(l)} \in \mathbb{R}^{d_h \times d_h}$ are learnable attention weights in scaled dot-product attention. Here, $\mathbf{h}_i^{(0)} = \mathbf{w}_i$ and $\mathbf{d}_i^{(0)} = \mathbf{d}_i$. Then, a two-layer feed-forward neural network with a ReLU activation takes the weighted sum $\tilde{\mathbf{h}}^{(l)}$ as input to learn the output hidden states $\mathbf{h}^{(l)}$ at the l-th layer:

$$\mathbf{h}^{(l)} = \max(0, \tilde{\mathbf{h}}^{(l)}W_1^{(l)} + \mathbf{b}_1^{(l)})W_2^{(l)} + \mathbf{b}_2^{(l)} \quad (5)$$

where $W_1^{(l)}$, $W_2^{(l)}$, $\mathbf{b}_1^{(l)}$, $\mathbf{b}_2^{(l)}$ are learnable parameters in BERT.

2.3 Gate-controlled Update

We propose a gate-controlled update to the dependency states $\mathbf{d}_i^{(l-1)}$ at the end of each layer by using the previous dependency representation $\mathbf{d}_i^{(l-1)}$ and the output hidden states $\mathbf{h}_i^{(l)}$. Inspired by the Gate Recurrent Unit (GRU) (Cho et al., 2014), we devise an update gate and a control gate. Specifically, the

control gate is a single fully-connected layer with a sigmoid activate function, which is defined as:

$$\mathbf{z}_i^{(l)} = \text{sigmoid}([\mathbf{h}_i^{(l)}; \mathbf{d}_i^{(l-1)}]W_Z^{(l)}) \quad (6)$$

where $W_Z^{(l)} \in \mathbb{R}^{2d_h \times d_h}$ is a learnable parameter. The update gate is a single fully-connected layer with a tanh activate function, which is defined as:

$$\mathbf{u}_i^{(l)} = \tanh(\mathbf{h}_i^{(l)}W_U^{(l)}) \quad (7)$$

where $W_U^{(l)} \in \mathbb{R}^{d_h \times d_h}$ is a learnable parameter.

Finally, the output dependency representations are learned by considering the last dependency state $\mathbf{d}_i^{(l-1)}$ and the update gate output $\mathbf{u}_i^{(l)}$ under the control of $\mathbf{z}_i^{(l)}$:

$$\mathbf{d}_i^{(l)} = (1 - \mathbf{z}_i^l) \odot \mathbf{d}_i^{(l-1)} + \mathbf{z}_i^l \odot \mathbf{u}_i^{(l)} \quad (8)$$

where $\odot$ represents the element-wise product.

2.4 Relation Classification

We apply a max-pooling operation on the position spans of the head and tail entities, and get the head entity representation $\mathbf{h}_{e_1}^L$ and tail entity representation $\mathbf{h}_{e_2}^L$, where L denotes the number of layers in BERT. Then, we concatenate $\mathbf{h}_{e_1}$ and $\mathbf{h}_{e_2}$ as the representation $\mathbf{h}$ of each input instance.

Following the prototypical network (Snell et al., 2017), we compute a prototype for each relation class c as $\mathbf{p}_c = \frac{1}{K_c}\sum_{(x_{s_i}, y_{s_i}) \in \mathcal{S}_c} \mathbf{h}_{x_{s_i}}$, where $\mathcal{S}_c = \{(x_{s_i}, y_{s_i})\}_{i=1}^{K_c}$ denotes the support set that has class label c, $\mathbf{h}_{x_{s_i}}$ is the contextual representation of x_{s_i}, and K_c is the number of instances in $\mathcal{S}_c$. Given the query set $\mathcal{Q} = \{(x_{q_i}, y_{q_i})\}_{i=1}^{K_\mathcal{Q}}$ and a Euclidean distance function $d(\cdot)$, the prototypical network computes a distribution over classes for a query instance x_{q_i} based on a softmax over distances to the prototypes in the embedding space. Formally, we define the prototypical objective $\mathcal{L}_{\text{proto}}$ over the query set $\mathcal{Q}$ as follows:

$$\mathcal{L}_{\text{proto}} = -\frac{1}{K_\mathcal{Q}} \sum_{i=1}^{K_\mathcal{Q}} \log \frac{\exp(-d(\mathbf{h}_{x_{q_i}}, \mathbf{p}_{y_{q_i}}))}{\sum_{c=1}^{N} \exp(-d(\mathbf{h}_{x_{q_i}}, \mathbf{p}_c))} \quad (9)$$

where $K_\mathcal{Q}$ denotes the number of instances in $\mathcal{Q}$.

Inference Stage In inference phase, we compute the relation $\hat{y}_i$ of each input x_i in testing set as:

$$\hat{y}_i = \underset{c}{\text{argmin}}\, d(\mathbf{h}_{x_i}, \mathbf{p}_c),\ c \in [1, \ldots, N] \quad (10)$$

3 Experiments

3.1 Experimental Setup

Dataset We use the benchmark FSRC dataset FewRel (Han et al., 2018) to evaluate the effectiveness of our model. FewRel contains 100 different relations, with 64 relations in training set, 16 relations in validation set and 20 relations in testing set. For each type of relation, there are 700 different examples. Since the 20 testing relations are unpublished, we re-split the published 64 training relations into 50 relations and 14 relations for training and validation respectively, and employ the original validation set with 16 relations for testing, following previous studies (Yang et al., 2020).

Baselines We compare DAPL with several state-of-the-art baselines for FSRC, including **Proto** (Snell et al., 2017), **Proto-GAT** (Snell et al., 2017), **BERT-PAIR** (Gao et al., 2019), **CTEG** (Wang et al., 2020), **TD-Proto** (Yang et al., 2020), and a simple version of **ConceptFERE** (Yang et al., 2021) that involves an external concept database.

Implementation Details For the PLM, the proposed DAPL is implemented based on BERT$_{base}$ for all experiments. We conduct N-way-K-shot (denoted as N-w-K-s) to study the performance in different situations. Here, we adopt four different settings, i.e., 5-w-1-s, 5-w-5-s, 10-w-1-s, and 10-w-5-s. We tune the entire model and select the checkpoint with best validation performance. The maximum length of the sentence is 90. We follow Soares et al. (2019) to insert four special tokens <e1>, </e1>, <e2> and </e2> to mark the boundaries of the entities. The dependency trees are obtained using the external Standard CoreNLP Toolkit proposed by StanfordNLP. The size of the dependency embedding is 384. DAPL is optimized with AdamW (Loshchilov and Hutter, 2018) and warmup mechanism (Popel and Bojar, 2018).

3.2 Experimental Results

Overall Results We adopt classification accuracy as the evaluation metric. Table 1 reports the experimental results of our model and four baselines in four few-shot settings. Our DAPL model achieves significantly better performance than the baselines in all settings. Specifically, our method improves the best performance of baselines by 0.28%/1.76%/0.75%/3.34% under the four settings respectively. The performance gain of our method comes from the auxiliary dependency information.

Model	5 way 1 shot	5 way 5 shot	10 way 1 shot	10 way 5 shot
Proto	72.65	86.15	60.13	76.20
Proto-GAT	79.14	88.46	68.87	79.45
BERT-PAIR	85.66	89.48	76.84	81.76
ConceptFERE	84.28	90.34	74.00	81.82
CTEG	84.72	92.52	76.01	84.89
TD-Proto	84.76	92.38	74.32	85.92
DAPL (Ours)	**85.94**	**94.28**	**77.59**	**89.26**
DAPL w/o Gate	85.44	93.68	76.29	88.27
DAPL w/o SDP	85.30	93.10	76.04	87.43
DAPL w/o DT	85.06	92.46	75.13	86.54

Table 1: The main evaluation results and the ablation results on the test set.

Ground-truth	By DAPL	By CTEG	Input Example
Husband-Wife	Husband-Wife	Children-Parent	He was born in Kristiania as a son of Gerda Ring and Halfdan Christensen and brother of Bab Christensen.
Parent-Children	Parent-Children	Husband-Wife	On March 8,1852 he married Kapi'olani, daughter of Kūhiō Kalaniana'ole and Kinoiki Kekaulike.

Table 2: Prediction results on the test samples. We use the red and blue colors to indicate the head and tail entities.

Ablation Study To analyze the impact of different components in DAPL, we also conduct ablation test in terms of discarding the dependency tree (denoted as w/o DT), the SDP dependency label (denoted as w/o SDP) and the gate-controlled update mechanism (denoted as w/o Gate). The ablation test results are reported in Table 1. The accuracy scores decrease sharply when removing the dependency tree. This is within our expectation since the dependency tree provides rich information of entities and relations between them. Not surprisingly, combining all the factors achieves the best performance over the four experimental settings.

Case Study In Table 2, we provide a case study to illustrate the effectiveness of our model for alleviating the misleading relation problem qualitatively. Specifically, we provide two examples from the test set that are incorrectly predicted by CTEG while being correctly predicted by our method. By fully exploiting the auxiliary dependency information, our DAPL can correctly predict the correct relation even being disturbed by the misleading relation "*Husband-wife*". However, CTEG has a propensity to confuse the co-exist relations in a sentence, since the misleading terms are close to the given entities.

4 Conclusion

In this paper, we proposed a dependency-aware prototype learning method for FSRC, which leveraged dependency trees and shortest dependency paths as structural information to complement the contextualized representations of input sentences. A gate-controlled update mechanism was further devised to adaptively update the dependency-aware representations according to the output of each network layer. Experimental results showed that DAPL outperformed strong baselines for FSRC.

Acknowledgements

This work was partially supported by National Natural Science Foundation of China (61906185, 61876053), the Natural Science Foundation of Guangdong Province of China (No. 2019A1515011705), Youth Innovation Promotion Association of CAS China (No. 2020357), Shenzhen Science and Technology Innovation Program (Grant No. KQTD20190929172835662), Shenzhen Basic Research Foundation (No. JCYJ20210324115614039 and No. JCYJ20200109113441941).

References

Guimin Chen, Yuanhe Tian, Yan Song, and Xiang Wan. 2021. Relation extraction with type-aware map memories of word dependencies. In *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*, pages 2501–2512.

Kyunghyun Cho, Bart Van Merriënboer, Caglar Gulcehre, Dzmitry Bahdanau, Fethi Bougares, Holger

Schwenk, and Yoshua Bengio. 2014. Learning phrase representations using rnn encoder-decoder for statistical machine translation. *arXiv preprint arXiv:1406.1078*.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*.

Ning Ding, Xiaobin Wang, Yao Fu, Guangwei Xu, Rui Wang, Pengjun Xie, Ying Shen, Fei Huang, Hai-Tao Zheng, and Rui Zhang. 2021. Prototypical representation learning for relation extraction. *arXiv preprint arXiv:2103.11647*.

Siqi Fan, Binbin Zhang, Silin Zhou, Menghan Wang, and Ke Li. 2021. Few-shot relation extraction towards special interests. *Big Data Research*, 26:100273.

Chelsea Finn, Pieter Abbeel, and Sergey Levine. 2017. Model-agnostic meta-learning for fast adaptation of deep networks. In *International conference on machine learning*, pages 1126–1135. PMLR.

Tianyu Gao, Xu Han, Ruobing Xie, Zhiyuan Liu, Fen Lin, Leyu Lin, and Maosong Sun. 2020. Neural snowball for few-shot relation learning. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pages 7772–7779.

Tianyu Gao, Xu Han, Hao Zhu, Zhiyuan Liu, Peng Li, Maosong Sun, and Jie Zhou. 2019. Fewrel 2.0: Towards more challenging few-shot relation classification. *arXiv preprint arXiv:1910.07124*.

Ruiying Geng, Binhua Li, Yongbin Li, Jian Sun, and Xiaodan Zhu. 2020. Dynamic memory induction networks for few-shot text classification. In *ACL*.

Ruiying Geng, Binhua Li, Yongbin Li, Xiaodan Zhu, Ping Jian, and Jian Sun. 2019. Induction networks for few-shot text classification. In *EMNLP*.

Zhijiang Guo, Yan Zhang, and Wei Lu. 2019. Attention guided graph convolutional networks for relation extraction. *arXiv preprint arXiv:1906.07510*.

Xu Han, Hao Zhu, Pengfei Yu, Ziyun Wang, Yuan Yao, Zhiyuan Liu, and Maosong Sun. 2018. Fewrel: A large-scale supervised few-shot relation classification dataset with state-of-the-art evaluation. *arXiv preprint arXiv:1810.10147*.

Shaoxiong Ji, Shirui Pan, Erik Cambria, Pekka Marttinen, and S Yu Philip. 2021. A survey on knowledge graphs: Representation, acquisition, and applications. *IEEE Transactions on Neural Networks and Learning Systems*.

Amina Kadry and Laura Dietz. 2017. Open relation extraction for support passage retrieval: Merit and open issues. In *Proceedings of the 40th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 1149–1152.

Ilya Loshchilov and Frank Hutter. 2018. Fixing weight decay regularization in adam.

Kangqi Luo, Fengli Lin, Xusheng Luo, and Kenny Zhu. 2018. Knowledge base question answering via encoding of complex query graphs. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 2185–2194.

Martin Popel and Ondřej Bojar. 2018. Training tips for the transformer model. *arXiv preprint arXiv:1804.00247*.

Amir Pouran Ben Veyseh, Franck Dernoncourt, Dejing Dou, and Thien Huu Nguyen. 2020. Exploiting the syntax-model consistency for neural relation extraction. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*.

Meng Qu, Tianyu Gao, Louis-Pascal Xhonneux, and Jian Tang. 2020. Few-shot relation extraction via bayesian meta-learning on relation graphs. In *International Conference on Machine Learning*, pages 7867–7876. PMLR.

Ying Shen, Ning Ding, Hai-Tao Zheng, Yaliang Li, and Min Yang. 2020. Modeling relation paths for knowledge graph completion. *IEEE Transactions on Knowledge and Data Engineering*.

Jake Snell, Kevin Swersky, and Richard Zemel. 2017. Prototypical networks for few-shot learning. *Advances in neural information processing systems*, 30.

Livio Baldini Soares, Nicholas FitzGerald, Jeffrey Ling, and Tom Kwiatkowski. 2019. Matching the blanks: Distributional similarity for relation learning. *arXiv preprint arXiv:1906.03158*.

Kai Sun, Richong Zhang, Yongyi Mao, Samuel Mensah, and Xudong Liu. 2020. Relation extraction with convolutional network over learnable syntax-transport graph. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pages 8928–8935.

Yuanhe Tian, Guimin Chen, Yan Song, and Xiang Wan. 2021. Dependency-driven relation extraction with attentive graph convolutional networks. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 4458–4471.

Linlin Wang, Zhu Cao, Gerard De Melo, and Zhiyuan Liu. 2016. Relation classification via multi-level attention cnns. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1298–1307.

Yingyao Wang, Junwei Bao, Guangyi Liu, Youzheng Wu, Xiaodong He, Bowen Zhou, and Tiejun Zhao. 2020. Learning to decouple relations: Few-shot relation classification with entity-guided attention and confusion-aware training. *arXiv preprint arXiv:2010.10894*.

Shiyao Xu and Yang Xiang. 2021. Frog-gnn: multi-perspective aggregation based graph neural network for few-shot text classification. *Expert Systems with Applications*, 176:114795.

Kaijia Yang, Nantao Zheng, Xinyu Dai, Liang He, Shujian Huang, and Jiajun Chen. 2020. Enhance prototypical network with text descriptions for few-shot relation classification. In *Proceedings of the 29th ACM International Conference on Information & Knowledge Management*, pages 2273–2276.

Shan Yang, Yongfei Zhang, Guanglin Niu, Qinghua Zhao, and Shiliang Pu. 2021. Entity concept-enhanced few-shot relation extraction. *arXiv preprint arXiv:2106.02401*.

Bowen Yu, Xue Mengge, Zhenyu Zhang, Tingwen Liu, Wang Yubin, and Bin Wang. 2020. Learning to prune dependency trees with rethinking for neural relation extraction. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 3842–3852.

Xiaoyan Zhao, Min Yang, Qiang Qu, Ruifeng Xu, and Jieke Li. 2022a. Exploring privileged features for relation extraction with contrastive student-teacher learning. *TKDE*.

Yingxiu Zhao, Zhiliang Tian, Huaxiu Yao, Yinhe Zheng, Dongkyu Lee, Yiping Song, Jian Sun, and Nevin-L. Zhang. 2022b. Improving meta-learning for low-resource text classification and generation via memory imitation. In *ACL*.

MECI: A Multilingual Dataset for Event Causality Identification

[1]Viet Dac Lai, [1]Amir Pouran Ben Veyseh, [1]Minh Van Nguyen
[2]Franck Dernoncourt, [1]Thien Huu Nguyen
[1]Dept. of Computer Science, University of Oregon, OR, USA
[2]Adobe Research, Seattle, WA, USA
{vietl,apouranb,minhnv,thien}@cs.uoregon.edu
franck.dernoncourt@adobe.com

Abstract

Event Causality Identification (ECI) is the task of detecting causal relations between events mentioned in the text. Although this task has been extensively studied for English materials, it is under-explored for many other languages. A major reason for this issue is the lack of multilingual datasets that provide consistent annotations for event causality relations in multiple non-English languages. To address this issue, we introduce a new multilingual dataset for ECI, called MECI. The dataset employs consistent annotation guidelines for five typologically different languages, i.e., English, Danish, Spanish, Turkish, and Urdu. Our dataset thus enable a new research direction on cross-lingual transfer learning for ECI. Our extensive experiments demonstrate high quality for MECI that can provide ample research challenges and directions for future research. We will publicly release MECI to promote research on multilingual ECI. The dataset is available at https://github.com/nlp-uoregon/meci-dataset.

1 Introduction

Event Causality Identification (ECI) is an important Information Extraction (IE) task that aims to identify causal relations between event mentions in text. For example, in the sentence *"After **inspection** of his computer , officers **found** that he was interested..."*, a ECI system should detect a causal relation between two events *"**inspection**"* $\xrightarrow{cause}$ *"**found**"*. ECI can provide valuable information for various applications such as event timeline construction (Shahaf and Guestrin, 2010), question-answering (Oh et al., 2016), future event forecasting (Hashimoto, 2019), and machine reading comprehension (Berant et al., 2014).

Due to its applications, ECI has been extensively studied in the natural language processing community over the past decade. The vast majority of methods for ECI involve feature engineering models (Do et al., 2011; Hu and Walker, 2017; Hashimoto, 2019; Ning et al., 2018; Gao et al., 2019) and recent deep learning architectures (Kadowaki et al., 2019; Zuo et al., 2021b; Liu et al., 2021; Zuo et al., 2021a; Man et al., 2022a). As such, the creation of large annotated datasets, e.g., EventStoryLine (Caselli and Vossen, 2017), has been critical to the development of ECI study. However, existing datasets for ECI only annotate causal relations between event mentions in data of a single language, i.e., mainly for English (Caselli and Vossen, 2017; Cybulska and Vossen, 2014; O'Gorman et al., 2016). On the one hand, this leaves many other languages unexplored for ECI, posing an important question about the generalization ability of existing methods to other languages. For instance, Spanish, Danish, and Turkish are not covered in those separated datasets for ECI. Moreover, the current single-language datasets for ECI tend to employ different annotation guidelines that prevent their combination into a larger corpus and cross-lingual transfer learning research to train and evaluate models in different languages. In all, the annotation discrepancy and limited language coverage hinder the research and development of the ECI in various dimensions, necessitating a new dataset with broader coverage for ECI.

To address this issue, this paper introduces a Multilingual Event Causality Identification (MECI) dataset to standardize and foster future research in multilingual ECI. Particularly, we present a large-scale ECI dataset for five languages, i.e., English, Danish, Spanish, Turkish, and Urdu[1] that are annotated with the same annotation guideline to enable cross-lingual transfer learning evaluation for the first time. As such, four languages, i.e., Danish, Spanish, Turkish, and Urdu, are not explored in any of the existing datasets for ECI. To facilitate open access to the dataset, we obtain the texts from

[1]We will maintain the dataset and include more languages along the way.

Proceedings of the 29th International Conference on Computational Linguistics, pages 2346–2356
October 12–17, 2022.

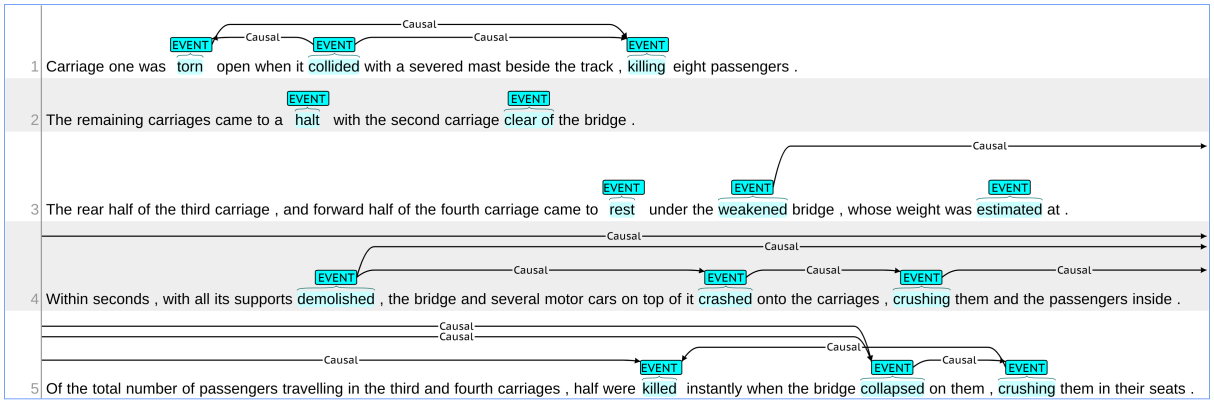

Figure 1: Our annotation interface for event causality identification.

Wikipedia for annotation in all examined languages. To make it consistent with prior research and benefit from the well-designed annotation guidelines of previous datasets, we inherit the event schema from the ACE 2005 dataset (Walker et al., 2006), and the causal event relation guideline from EventStoryLine (Caselli and Vossen, 2017) (with both explicit and implicit causal relations) during the annotation process. In total, our MECI dataset involves 46K events and 11K relations that are substantially larger than those in existing ECI datasets. Figure 1 illustrates our annotation interface in this work.

In addition, we evaluate the proposed MECI dataset using the state-of-the-art models for ECI. We investigate the challenges of MECI over all examined languages through the monolingual setting where the models are trained and evaluated in the same language. The experiments show that the performance of existing ECI models, even with large pre-trained language models (PLMs), is far from satisfactory; models for non-English languages generally perform poorer than their English counterparts. We also observe the importance of choosing language-specific or multilingual PLMs for ECI models as their effectiveness varies for different languages. Moreover, we evaluate the models in the zero-shot cross-lingual setting, where the models are trained on English data and tested on the data of the other languages. The experiment suggests transferability of ECI knowledge between English and Urdu while showing a significant performance drop in other language pairs. These results can serve as baselines for future studies on cross-lingual transfer learning for ECI. Finally, we report the analysis and challenges of the MECI dataset to provide insights for future ECI research. We will publicly release MECI to promote future studies in multilingual ECI.

2 Data Annotation

2.1 Annotation Scheme

Our goal is to annotate causal relations between event mentions in text. To this end, we define the annotation scheme for event mentions following the guidelines for the ACE 2005 dataset (Walker et al., 2006) for events, while the annotation guidelines for event causality relations are obtained from those for the EventStoryLine dataset (Caselli and Vossen, 2017). This allows us to inherit the well-designed documentation in such benchmark datasets and achieve consistency with prior research for ECI.

In particular, based on the ACE 2005 annotation guideline, an event in our dataset is either (1) an occurrence involving some participants, or (2) something that happens, or (3) a change of state. Event mentions/triggers are words/phrases in text that clearly evoke some event. As we are mainly interested in event causality relations, we only annotate event mention spans and do not include event types. To accommodate different languages, we allow event mentions/triggers to span multiple words in the sentences.

Next, for event causality relations, our annotation guideline follows the EventStoryLine dataset. In particular, a causal relation represents a directional relation between two events in which an event (CAUSE) causes another event (EFFECT) to happen or hold. This definition covers standard causal relations: cause, enablement, and prevention (Caselli and Vossen, 2017). In addition, similar to EventStoryLine, our dataset covers both explicit and implicit causality. Note that this is an extension from most prior annotation schema,

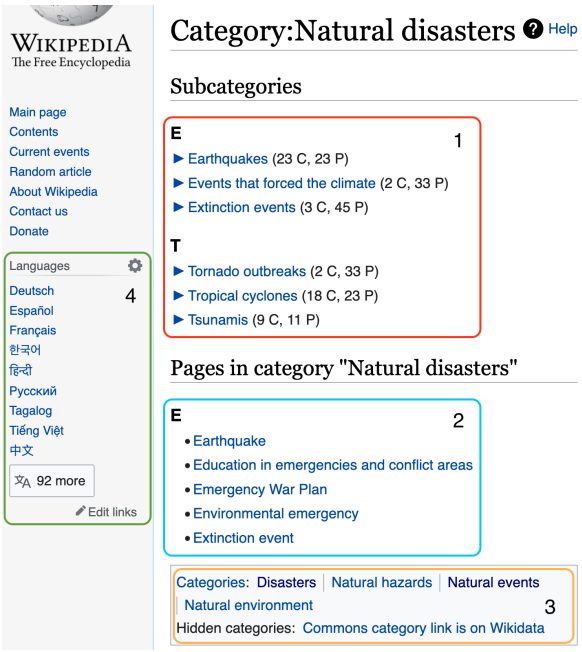

Figure 2: The Wikipedia category page of *Natural disasters* with its child categories (box 1), associated pages (2), parent categories (box 3), and interlink to the same category in other languages (box 4).

i.e., Causal-TimeBank (Mirza and Tonelli, 2014), RED (O'Gorman et al., 2016), BECauSe (Dunietz et al., 2017), that have only considered explicit relations covering the three causal concepts: *cause, enable,* and *prevent* through a verb-based lexicalization (Wolff, 2007). In our view, causality is a tool for humans to understand the world, and its existence is independent of the actual language for presentation (Neeleman et al., 2012). Hence, event causality relations might be established without explicit ground in the text. In other words, there are implicit causal relations between events that are not covered by the above lexicalization (Caselli and Vossen, 2017; Webber et al., 2019). To capture this important type of event causality relations, our annotation guideline is extended to cover implicit relations which require background knowledge, e.g., common-sense, domain-specific knowledge, for successful identification. Finally, similar to prior datasets, we annotate both intra- and inter-sentential causal relations between two events (Mirza and Tonelli, 2014; Caselli and Vossen, 2017).

2.2 Data Collection & Preparation

The documents for our MECI dataset are collected from Wikipedia for five topologically different languages, i.e., English, Danish, Spanish, Turkish, and Urdu. In particular, we focus on 5 topics: aviation accidents, railway accidents, natural disasters, conflicts, and economic crisis, to expect a high yield of events and event causality relations. Wikipedia organizes articles into a hierarchical graph of categories. A category is a group of articles sharing a topic that might be further split into finer subcategories as shown in Figure 2. Furthermore, the hierarchical category systems in Wikipedia for different languages are interconnected through interlinks between identical categories. Therefore, by exploiting the category systems and language interlinks, we are able to obtain Wikipedia articles of the same topics across many languages.

Given the list of five categories for the examined languages, we crawl all the articles associated with their category descendants (i.e., subcategories, subsubcategories) in the hierarchy up to the depth of 6. After this step, we obtain at least 1,000 articles per category for each language. The obtained articles are cleaned by removing format elements (i.e., lists, images, URLs, and markups) to retain only textual data. Afterward, the articles are split into sentences and tokenized into words by Trankit (Nguyen et al., 2021), a multilingual text processing tool with state-of-the-art performance. The detailed list of subcategory URLs will be included in the final dataset package.

Given an article, a direct method for data annotation for ECI is to ask the annotators to label all the event mention spans and event mention pairs with causal relations. However, as the number of event mention pairs in a document grows quadratically with respect to the number of event mentions, a long Wikipedia article can easily overwhelm the annotators, thus affecting the quality of the annotated data. To address the issue, we split the Wikipedia articles into smaller chunks that span five consecutive sentences for separate annotation, following prior practices (Mostafazadeh et al., 2016; Ebner et al., 2020). These chunks are called documents in our dataset. In this way, the annotators only need to consider a shorter context at a time to enhance the attention and quality of annotated data.

2.3 Human Annotation

To annotate the obtained documents, we hire annotators from `upwork.com`, a crow-sourcing platform with freelancers from all around the globe. We only consider candidates that are (1) native to the target language, (2) fluent in English, and

Language	Event	Relation
Danish	0.68	0.58
English	0.92	0.80
Spanish	0.84	0.66
Turkish	0.69	0.61
Urdu	0.65	0.75

Table 1: Kappa scores for the MECI dataset.

(3) highly approved among the Upwork employers. We can access this information from the annotators' profiles on the platform. The candidates are then given annotation guidelines and a test for performing both event annotation and event causality relation extraction tasks. The top two candidates are hired for each language. We use BRAT annotation tool for our annotation (Stenetorp et al., 2012) and illustrated in Figure 1.

Our annotation consists of two tasks, i.e., event mention annotation and event causal relation annotation. For each language, we annotate event causality relations over the outputs from event mention annotation (i.e., after event mention annotation has been completed and finalized for all documents). Given a sample of selected documents for a language, for each task, the two annotators for that language independently annotate event mentions/event causal relations for the documents. Afterward, the annotation conflicts will be presented to the annotators for further discussion and revision to produce the final version of annotated documents for the current task. This will help to ensure high agreement and consistency for our dataset.

2.4 Data Analysis

Table 1 presents our Kappa scores for annotation agreements of event mentions and event causality relations over different languages. Note that these scores are computed by comparing the independent annotations of the annotators over the documents before engaging in discussion to resolve conflicts. As can be seen, the scores are very close to either substantial or almost perfect agreement for all the tasks and languages, thus demonstrating the high quality of our created MECI dataset. We also find that non-English languages tend to have lower annotation agreement scores for both event mention and causality relation extraction tasks, thus highlighting the challenges of ECI for non-English languages and showing the importance of additional research for multilingual ECI.

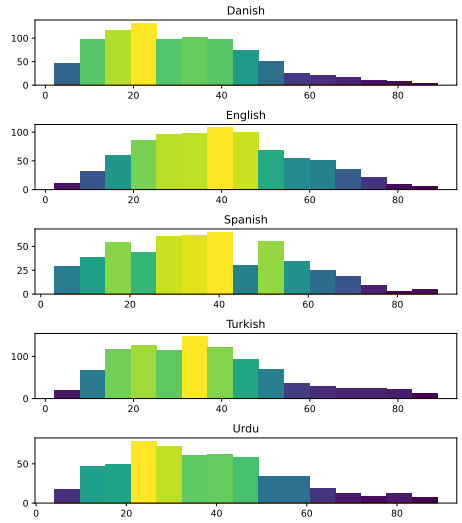

Figure 3: Distributions of distances between two event mentions with causal relations in MECI. Distances are measured via the number of words.

In addition, Table 2 show other statistics for our MECI dataset. Across five languages, each document contains an average of 13.0 event triggers, which account for 2.6 event triggers per sentence. This reveals a challenge of MECI for ECI models that might need to handle the ambiguity due to the overlap of the context of event mention pairs in both sentence and document levels. Furthermore, each document contains approximately 3.1 relations on average; however, there is a discrepancy in event causality relation density in documents among languages. In particular, English and Turkish represent a much denser level of event causality relations per document than other languages, especially Spanish and Urdu. As such, the divergences in the density of event causality relations (and event mentions) pose another robustness challenge for ECI models that should be able to bridge the gaps and transfer event causal knowledge across languages.

Finally, Figure 3 presents the distributions of distances between two event mentions with causal relations for five examined languages in MECI (the distances are counted via the number of words in between). There are several observations from the figure. First, for all the languages, a majority of event mentions are 10 to 50 words away from each other in the documents. This suggests diverse levels of context information between event mentions that an ECI model needs to capture to perform well for the languages in MECI. Second, there are clear divergences between the distance distributions of

causal event mention pairs over languages. For instance, the distances between event mentions for Danish and Urdu seem to be more distributed in the shorter ranges than those of English and Spanish. Such distribution differences require ECI models to introduce robust mechanisms to induce language-transferable representations for diverse causal contexts in cross-lingual learning for ECI.

Dataset Comparison: Table 2 also compares our MECI dataset with previous public datasets for ECI. Note that we focus on the datasets that explicitly consider causal relations between event mentions/triggers to make them comparable. It is clear from the table that our MECI dataset has a much larger scale with more event mentions, causal relations, and languages than all previous datasets for ECI. This will enable the training of larger models and a more comprehensive evaluation for ECI.

2.5 Challenges

Unlike most prior ECI datasets, our MECI dataset includes implicit causal relations, which allow causal relations to be derived from various implicit reasoning sources such as common-sense knowledge. This section illustrates some types of implicit reasoning for causal relations between events discovered in our dataset.

Implicit inference of causal cues: In the following example, considering two event mentions: *"derailed"* and *"running into"*, there is no triggering verb-based expression to signal the causal relationship between the two events. However, with the presence of the trailing comma between the two event mentions, our annotators can easily realize that the *"derail"* event is the cause of the *"running into"* event. As such, the annotators might have implicitly inferred the reduced relative clause "*which makes the train*" (presented in the brackets) between the two event mentions to make the causal decision. To this end, a model will also need to recognize such implicit reasoning cues based on the context to successfully perform ECI.

> *The Granville rail disaster ... when a crowded commuter train* ***derailed****, [~~which makes the train~~]* ***running into*** *the supports of a road bridge that ...*

Implicit transitivity: Consider three event mentions *"trouble"*, *"bail out"*, and *"killed"* in the following example. The ground text explicitly expresses the causal relation *"bail out"* $\xrightarrow{cause}$ *"killed"* via the adverb *"consequently"*. However, there is no clear signal of the causality between *"trouble"* and *"bail out"*, which requires common-sense knowledge to successfully recognize for the causal order of such events, i.e., *"trouble"* $\xrightarrow{cause}$ *"bail out"*. This increases the difficulty for identifying the causality *"trouble"* $\xrightarrow{cause}$ *"killed"*, which might entail transitivity reasoning between implicit and/or explicit causal relations, i.e., *"trouble"* $\xrightarrow{cause}$ *"bail out"* and *"bail out"* $\xrightarrow{cause}$ *"killed"*.

> *... when his Spitfire developed engine* ***trouble*** *between the islands of Skiathos and Skópelos over the Aegean Sea . He attempted to* ***bail out*** *of the aircraft, but his altitude was too low for his parachute to open, and he was consequently* ***killed****.*

3 Experiments

We randomly split the documents for each language in MECI into three separate parts with a ratio of 3/1/1 to serve as training, development, and test data respectively for experiments. To study the challenges of ECI presented in MECI, we evaluate the performance of the state-of-the-art models for ECI on this dataset. Each model will be comprehensively evaluated in the monolingual learning (i.e., trained and tested on data of the same language) and multilingual learning (i.e., trained and tested on the data of different language) settings with MECI.

3.1 ECI Models

We explore the following representative models for ECI in the literature:

PLM: This model is inherited from the BERT baseline in (Tran Phu and Nguyen, 2021). Given an input document D, this model concatenates the words from all sentences and sends it into a pre-trained language model, e.g., BERT (Devlin et al., 2019), to obtain representation vectors for each word-piece using the hidden vectors in the last transformer layer. Afterward, given the spans A and B for two event mentions e_A and e_B of interest in D, we compute the representations $\mathbf{r}_A$, $\mathbf{r}_B$ for the two event mentions by averaging the representation vectors of the word pieces within the corresponding spans A and B. Finally, we form an overall representation vector $\mathbf{r}_{A \to B} = [\mathbf{r}_A, \mathbf{r}_B, \mathbf{r}_A - \mathbf{r}_B, \mathbf{r}_A * \mathbf{r}_B]$ ($*$ is the element-wise multiplication operation) for ECI.

Dataset	Lang	#Documents	#Relations	#Events	Relation Type
Causal-TimeBank (Mirza et al., 2014)		100	318	11,000	Explicit
RED (O'Gorman et al., 2016)		95	*4,969	8,731	Explicit
BECauSE-2.0 (Dunietz et al., 2017)	English	118	1,803	-	Explicit
CaTeRS (Mostafazadeh et al., 2016)		320	488	2,708	Explicit, Implicit
EventStoryline (Caselli and Vossen, 2017)		258	5,519	7,275	Explicit, Implicit
MECI	Danish	519	1,377	6,909	Explicit, Implicit
	English	438	2,050	8,732	
	Spanish	746	1,312	11,839	
	Turkish	1,357	5,337	14,179	
	Urdu	531	979	4,975	
MECI (total)	Various	3591	11,055	46,634	Explicit, Implicit

Table 2: Comparison of public ECI datasets. #Relations indicates the number of causal relations in the datasets. * designates the numbers that include other event-event relations, i.e., temporal and hierarchical relations.

	Model	MECI English			EventStoryLine		
		P	R	F	P	R	F
BERT	PLM	35.6	44.9	39.7	27.3	35.3	30.8
	RichGCN	**48.1**	**69.5**	**56.8**	**42.6**	**51.3**	**46.6**

Table 3: Performance of models on MECI (English) and EventStoryLine datasets.

This vector will be fed into a feed-forward network with a sigmoid function in the end to predict the causal relationship between e_A and e_B in D.

RichGCN (Tran Phu and Nguyen, 2021): Similar to **PLM**, **RichGCN** employs a PLM to encode the entire input document and compute an overall representation vector $\mathbf{r}_{A \to B}$ for identifying the causal relationship between two given event mentions. To enhance representation learning, **RichGCN** also introduce several interaction graphs (with words and event mentions in the input document as the nodes) to capture relevant context information/interactions for the causal relationship between two event mentions. In particular, to adapt **RichGCN** to MECI with multiple languages, we implement four interaction graphs to represent an input document: (1) *Sentence Boundary Graph* where words or event mentions within each sentence in the document are connected to each other; (2) *Event Mention Span Graph* where words within each event mention span are connected to the event mention; (3) *Syntax-based Graph* where words within each sentence are connected to each other following the dependency tree structure of the sentence; and (4) *Semantic-based Graph* where words across the document are connected to each other; the weights for the connections are measured via the similarity between the word representations (computed from PLM). In **RichGCN**, each interaction graph is represented by an adjacency matrix. A final graph V to capture relevant connections for the two event mentions is formed by learning a linear combination of the adjacency matrices of the four graphs. Finally, the graph V is then sent into a Graph Convolutional Network (GCN) (Kipf and Welling, 2017; Nguyen and Grishman, 2018) to compute a richer representation for the two event mentions with more relevant context to perform ECI.

Know (Liu et al., 2021): By treating the event mentions as concepts in ConceptNet (Speer et al., 2017), **Know** retrieves related concepts and relations for the two input event mentions in our ECI problem from ConceptNet. The retrieved information is then used to augment the input text. As such, **Know** also utilizes a PLM to encode the augmented text to compute prediction representation for ECI. In addition, this model employs a masking mechanism to obtain event-agnostic context from input text, serving as another source of information to be encoded by the PLM and incorporated into representation learning for our task.

3.2 Experiment Setups

In the monolingual learning settings, for each language in MECI, we train the ECI models on the training data and evaluate model performance on the test data of the same language. We explore both multilingual PLMs, i.e., mBERT (Devlin et al., 2019) and XLMR (Conneau et al., 2020), and language-specific PLMs for the languages in MECI as the encoder for the ECI models in the experiments. In particular, we utilize the following language-specific PLMs that are available for MECI languages, i.e., BERT (Devlin et al., 2019)

	Model	English			Danish			Spanish			Turkish			Urdu		
		P	R	F	P	R	F	P	R	F	P	R	F	P	R	F
mBERT	PLM	38.4	46.0	41.9	25.2	26.6	25.9	43.9	41.5	42.7	36.2	48.7	41.6	31.9	34.3	33.0
	Know	35.8	56.7	43.9	25.8	36.0	30.1	39.7	38.3	39.0	39.7	46.9	43.0	36.7	35.3	36.0
	RichGCN	48.4	67.1	56.2	29.7	38.0	33.4	**51.2**	52.0	51.6	50.0	59.9	54.5	40.1	50.0	44.5
XLMR	PLM	48.7	59.9	53.7	**35.9**	36.2	36.0	50.6	49.1	49.9	44.0	59.4	50.5	40.4	43.2	41.8
	Know	39.3	42.6	40.9	31.4	11.4	16.7	39.9	28.4	33.2	36.5	46.7	41.0	**41.1**	22.2	28.9
	RichGCN	**50.6**	**68.0**	**58.1**	31.9	**50.0**	**38.9**	50.7	**55.0**	**52.8**	**50.5**	**64.6**	**56.7**	37.7	**56.0**	**45.1**

Table 4: Monolingual learning performance of ECI models on MECI with mBERT and XLMR.

	Model	English			Danish			Spanish			Turkish			Urdu		
		P	R	F	P	R	F	P	R	F	P	R	F	P	R	F
*	PLM	35.6	44.9	39.7	23.2	23.0	23.1	42.7	44.6	43.6	40.4	56.0	46.9	20.2	33.5	25.2
	RichGCN	**48.1**	**69.5**	**56.8**	**27.1**	**35.0**	**30.6**	**59.8**	**48.2**	**53.4**	**54.7**	**62.0**	**58.1**	**31.1**	**47.9**	**37.7**

Table 5: Monolingual learning performance of ECI models on MECI with language-specific PLMs.

for English; BotXO[2] for Danish, BETO (Cañete et al., 2020) for Spanish, BERTurk (Schweter, 2020) for Turkish, and UrduHack[3] for Urdu.

The support of multiple languages with the same annotation guideline for event causality relations in MECI allows us to perform cross-lingual transfer learning evaluation for ECI models. In particular, for cross-lingual settings, ECI models are trained on the training data of one language (the source language); however, they are evaluated on test data of new target languages. In the experiments, we treat English as the source language and other languages in MECI as the target languages for cross-lingual evaluation. To facilitate the prediction over multiple languages, we leverage the multilingual PLMs mBERT and XLMR in cross-lingual experiments.

Hyper-parameters: We employ the same hyper-parameters from the original works for the ECI models: **RichGCN** (Tran Phu and Nguyen, 2021), and **Know** (Liu et al., 2021) in the experiments. The multilingual NLP toolkit Trankit (Nguyen et al., 2021) is leveraged to obtain dependency trees for sentences in multiple languages for the **RichGCN** model. Also, we utilize the multilingual version of ConceptNet (Speer et al., 2017) to retrieve augmented information for **Know**. Finally, we employ the base versions for all the multilingual and monolingual PLMs considered in this work.

3.3 Results

Monolingual Performance: Table 4 shows the performance of the three ECI models on the monolingual learning settings across all the languages with the multilingual PLMs: mBERT and XLMR. Among the ECI models, we find that **RichGCN** maintains its top performance across all the languages and multilingual PLMs, thus demonstrating the effectiveness of its language-agnostic document structure to represent documents for ECI. Nonetheless, the best performance by **RichGCN** for English, Danish, Spanish, Turkish, and Urdu is 58.1, 38.9, 52.8, 56.7, and 45.1. These performance is far from being perfect, thus suggesting the challenges for ECI across languages and presenting ample research opportunities to improve the performance in the future. In addition, among the models, **Know** exhibits mixed performance with mBERT and worst performance with XLMR across languages. We attribute this phenomenon to the unstable quality of the concept retrieval with ConceptNet and context modification in **Know** that might exclude important causal context from the input texts to cause poor performance in different languages. Finally, comparing the multilingual PLMs, we find that XLMR performs significantly better than mBERT over all the languages with the **PLM** and **RichGCN** models, thus suggesting the benefits of XLMR for future ECI research.

Effects of language-specific PLMs: To better understand the effectiveness of PLMs for ECI, Table 5 reports the performance of **PLM** and **RichGCN** in the monolingual learning settings where language-specific PLMs for each language are employed as the encoder for the models. As can be seen, using the best model **RichGCN** and the best multilingual PLM XLMR as the anchors, ECI performance for English, Spanish and Turkish is very close with monolingual and multilingual PLMs (i.e., less than 2% difference in F1 scores). However, multilingual PLMs are substantially better than monolingual

[2]https://huggingface.co/Maltehb/danish-bert-botxo
[3]https://github.com/urduhack/urduhack

	Model	English → Danish			English → Spanish			English → Turkish			English → Urdu		
		P	R	F	P	R	F	P	R	F	P	R	F
mBERT	PLMd	12.4	35.4	18.4	11.4	63.3	19.3	21.5	47.6	29.6	17.0	44.2	24.6
	Know	7.8	**62.0**	13.8	7.2	**69.4**	13.0	20.4	55.5	29.9	14.2	61.5	23.0
	RichGCN	23.7	45.3	31.1	20.6	58.6	30.5	44.5	52.0	48.0	35.0	56.8	43.3
XLMR	PLM	20.1	59.2	30.1	16.0	66.4	25.8	36.1	**60.5**	45.2	25.7	**62.0**	36.3
	Know	13.3	42.1	20.3	10.4	47.3	17.1	25.8	57.6	35.7	19.3	54.5	28.5
	RichGCN	**28.5**	43.7	**34.5**	**22.7**	62.4	**33.3**	**46.4**	55.0	**50.3**	**38.6**	55.2	**45.5**

Table 6: Zero-shot cross-lingual learning performance on MECI using English as source language.

PLMs for Danish and Urdu (up to 7% difference in performance). This can be attributed to the lower resources in Danish and Urdu that hinder effective training for language-specific PLMs. With multilingual PLMs, such low-resource languages can benefit more from data in other languages to train multilingual PLMs.

Cross-lingual Performance: To investigate the transferability of ECI knowledge across languages, Table 6 presents the performance of the ECI models in the cross-lingual learning settings. Note that in these experiments English is the source languages while other languages are the targets. Among the three models, **RichGCN** is still the best performer across all target languages. However, the model's performance drops significantly for the three target languages Danish (by 4.4%), Spanish (by 19.5%), and Turkish (by 6.4%) compared to their monolingual performance with XLMR. This illustrates the challenges and necessity of further research on cross-lingual transfer learning for ECI that can now be enabled with our multilingual dataset.

Interestingly, compared to the monolingual settings, the performance on Urdu of **RichGCN** is slightly improved (by 0.4%) in the cross-lingual setting. One potential reason is due to the smallest size of the training data for Urdu in MECI that allows the larger English training data to train better models for Urdu test data. In addition, among the four target languages, we observe a wide range of cross-lingual performance from the model trained on English data, thus showing the diverse nature of data and languages in MECI for future research.

4 Related Work

As an important task in IE, ECI has attracted extensive research effort to develop effective models (Do et al., 2011; Hashimoto et al., 2014; Hidey and McKeown, 2016; Hu and Walker, 2017; Kadowaki et al., 2019; Zuo et al., 2020; Liu et al., 2021; Tran Phu and Nguyen, 2021; Man et al., 2022b). To support model development for ECI, several datasets have been introduced for this task, including PDTB (Prasad et al., 2008), Causal-TimeBank (Mirza, 2014), ECB (Cybulska and Vossen, 2014), Richer Event Description (O'Gorman et al., 2016), BeCause (Dunietz et al., 2017), and EventStoryLine (Caselli and Vossen, 2017), CaTeRS (Mostafazadeh et al., 2016). However, these previous work and datasets only focus on English data, presenting a strong demand for new research and datasets on other languages for ECI.

To this end, there are a few efforts on creating causality corpora for other languages, such as German (Rehbein and Ruppenhofer, 2020), Arabic (Sadek et al., 2018) and Persian (Rahimi and Shamsfard, 2021). However, these corpus consider not only event mentions, but also entities, clauses, and sentences, thus, not directly solving ECI as we do. In addition, most existing annotation efforts for ECI focus on explicit event causality relationships. EventStoryLine (Caselli and Vossen, 2017) and CaTerRS (Mostafazadeh et al., 2016) are the only two prior datasets that also explore implicit causal relationships between events. However, they do not provide annotation for multiple languages as we do in MECI. Finally, we also note recent efforts on creating multilingual datasets for other NLP tasks, including event detection (Pouran Ben Veyseh et al., 2022), natural language understanding (e.g., slot filling) (FitzGerald et al., 2022), and acronym extraction (Veyseh et al., 2022).

5 Conclusion

We present a new dataset for event causality identification in five different languages across diverse typologies. The dataset is annotated consistently for all languages, offering a large number of event mentions/causal relations and covering four languages that have not been explored in the prior ECI resources. Our extensive experiments and analysis

reveal the quality and challenges of our dataset for ECI. In addition, our dataset enables cross-lingual transfer learning research that is not possible with current resources for ECI. In the future, we plan to extend the dataset to include more languages such as Arabic and Hindi to broaden its coverage.

Ethical Considerations

In this work we present a dataset annotated over the publicly accessible articles of `wikipedia.org`. Complying with the discussion presented by Benton et al. (2017), research with human subject information is exempted from the required full Institutional Review Board (IRB) review if the data is already available from public sources (as with Wikipedia) or if the identity of the subjects cannot be recovered.

Acknowledgement

This research has been supported by the Army Research Office (ARO) grant W911NF-21-1-0112 and the NSF grant CNS-1747798 to the IU-CRC Center for Big Learning. This research is also based upon work supported by the Office of the Director of National Intelligence (ODNI), Intelligence Advanced Research Projects Activity (IARPA), via IARPA Contract No. 2019-19051600006 under the Better Extraction from Text Towards Enhanced Retrieval (BETTER) Program. The views and conclusions contained herein are those of the authors and should not be interpreted as necessarily representing the official policies, either expressed or implied, of ARO, ODNI, IARPA, the Department of Defense, or the U.S. Government. The U.S. Government is authorized to reproduce and distribute reprints for governmental purposes notwithstanding any copyright annotation therein. This document does not contain technology or technical data controlled under either the U.S. International Traffic in Arms Regulations or the U.S. Export Administration Regulations.

References

Adrian Benton, Glen Coppersmith, and Mark Dredze. 2017. Ethical research protocols for social media health research. In *Proceedings of the First ACL Workshop on Ethics in Natural Language Processing*, pages 94–102, Valencia, Spain. Association for Computational Linguistics.

Jonathan Berant, Vivek Srikumar, Pei-Chun Chen, Abby Vander Linden, Brittany Harding, Brad Huang, Peter Clark, and Christopher D. Manning. 2014. Modeling biological processes for reading comprehension. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1499–1510, Doha, Qatar. Association for Computational Linguistics.

Tommaso Caselli and Piek Vossen. 2017. The event StoryLine corpus: A new benchmark for causal and temporal relation extraction. In *Proceedings of the Events and Stories in the News Workshop*, pages 77–86, Vancouver, Canada. Association for Computational Linguistics.

José Cañete, Gabriel Chaperon, Rodrigo Fuentes, Jou-Hui Ho, Hojin Kang, and Jorge Pérez. 2020. Spanish pre-trained bert model and evaluation data. In *PML4DC at ICLR 2020*.

Alexis Conneau, Kartikay Khandelwal, Naman Goyal, Vishrav Chaudhary, Guillaume Wenzek, Francisco Guzmán, Edouard Grave, Myle Ott, Luke Zettlemoyer, and Veselin Stoyanov. 2020. Unsupervised cross-lingual representation learning at scale. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 8440–8451, Online. Association for Computational Linguistics.

Agata Cybulska and Piek Vossen. 2014. Guidelines for ecb+ annotation of events and their coreference.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, Minneapolis, Minnesota. Association for Computational Linguistics.

Quang Do, Yee Seng Chan, and Dan Roth. 2011. Minimally supervised event causality identification. In *Proceedings of the 2011 Conference on Empirical Methods in Natural Language Processing*, pages 294–303, Edinburgh, Scotland, UK. Association for Computational Linguistics.

Jesse Dunietz, Lori Levin, and Jaime Carbonell. 2017. The BECauSE corpus 2.0: Annotating causality and overlapping relations. In *Proceedings of the 11th Linguistic Annotation Workshop*, pages 95–104, Valencia, Spain. Association for Computational Linguistics.

Seth Ebner, Patrick Xia, Ryan Culkin, Kyle Rawlins, and Benjamin Van Durme. 2020. Multi-sentence argument linking. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 8057–8077, Online. Association for Computational Linguistics.

Jack FitzGerald, Christopher Hench, and Charith Peris et al. 2022. Massive: A 1m-example multilingual natural language understanding dataset with 51 typologically-diverse languages.

Lei Gao, Prafulla Kumar Choubey, and Ruihong Huang. 2019. Modeling document-level causal structures for event causal relation identification. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 1808–1817, Minneapolis, Minnesota. Association for Computational Linguistics.

Chikara Hashimoto. 2019. Weakly supervised multilingual causality extraction from Wikipedia. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 2988–2999, Hong Kong, China. Association for Computational Linguistics.

Chikara Hashimoto, Kentaro Torisawa, Julien Kloetzer, Motoki Sano, István Varga, Jong-Hoon Oh, and Yutaka Kidawara. 2014. Toward future scenario generation: Extracting event causality exploiting semantic relation, context, and association features. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 987–997, Baltimore, Maryland. Association for Computational Linguistics.

Christopher Hidey and Kathy McKeown. 2016. Identifying causal relations using parallel Wikipedia articles. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1424–1433, Berlin, Germany. Association for Computational Linguistics.

Zhichao Hu and Marilyn Walker. 2017. Inferring narrative causality between event pairs in films. In *Proceedings of the 18th Annual SIGdial Meeting on Discourse and Dialogue*, pages 342–351, Saarbrücken, Germany. Association for Computational Linguistics.

Kazuma Kadowaki, Ryu Iida, Kentaro Torisawa, Jong-Hoon Oh, and Julien Kloetzer. 2019. Event causality recognition exploiting multiple annotators' judgments and background knowledge. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 5816–5822, Hong Kong, China. Association for Computational Linguistics.

Thomas N. Kipf and Max Welling. 2017. Semi-supervised classification with graph convolutional networks. In *Proceedings of the 5th International Conference on Learning Representations*.

Jian Liu, Yubo Chen, and Jun Zhao. 2021. Knowledge enhanced event causality identification with mention masking generalizations. In *Proceedings of the Twenty-Ninth International Conference on International Joint Conferences on Artificial Intelligence*, pages 3608–3614.

Hieu Man, Nghia Trung Ngo, Linh Van Ngo, and Thien Huu Nguyen. 2022a. Selecting optimal context sentences for event-event relation extraction. In *Proceedings of the Association for the Advancement of Artificial Intelligence (AAAI)*.

Hieu Man, Minh Nguyen, and Thien Nguyen. 2022b. Event causality identification via generation of important context words. In *Proceedings of the 11th Joint Conference on Lexical and Computational Semantics*, pages 323–330, Seattle, Washington. Association for Computational Linguistics.

Paramita Mirza. 2014. Extracting temporal and causal relations between events. In *Proceedings of the ACL 2014 Student Research Workshop*, pages 10–17, Baltimore, Maryland, USA. Association for Computational Linguistics.

Paramita Mirza, Rachele Sprugnoli, Sara Tonelli, and Manuela Speranza. 2014. Annotating causality in the TempEval-3 corpus. In *Proceedings of the EACL 2014 Workshop on Computational Approaches to Causality in Language (CAtoCL)*, pages 10–19, Gothenburg, Sweden. Association for Computational Linguistics.

Paramita Mirza and Sara Tonelli. 2014. An analysis of causality between events and its relation to temporal information. In *Proceedings of COLING 2014, the 25th International Conference on Computational Linguistics: Technical Papers*, pages 2097–2106, Dublin, Ireland. Dublin City University and Association for Computational Linguistics.

Nasrin Mostafazadeh, Alyson Grealish, Nathanael Chambers, James Allen, and Lucy Vanderwende. 2016. CaTeRS: Causal and temporal relation scheme for semantic annotation of event structures. In *Proceedings of the Fourth Workshop on Events*, pages 51–61, San Diego, California. Association for Computational Linguistics.

Ad Neeleman, Hans Van de Koot, et al. 2012. The linguistic expression of causation. *The theta system: Argument structure at the interface*, 20.

Minh Van Nguyen, Viet Dac Lai, Amir Pouran Ben Veyseh, and Thien Huu Nguyen. 2021. Trankit: A lightweight transformer-based toolkit for multilingual natural language processing. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: System Demonstrations*, pages 80–90, Online. Association for Computational Linguistics.

Thien Huu Nguyen and Ralph Grishman. 2018. Graph convolutional networks with argument-aware pooling for event detection. In *Proceedings of the Association for the Advancement of Artificial Intelligence (AAAI)*.

Qiang Ning, Zhili Feng, Hao Wu, and Dan Roth. 2018. Joint reasoning for temporal and causal relations. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 2278–2288, Melbourne, Australia. Association for Computational Linguistics.

Tim O'Gorman, Kristin Wright-Bettner, and Martha Palmer. 2016. Richer event description: Integrating event coreference with temporal, causal and bridging annotation. In *Proceedings of the 2nd Workshop on Computing News Storylines (CNS 2016)*, pages 47–56, Austin, Texas. Association for Computational Linguistics.

Jong-Hoon Oh, Kentaro Torisawa, Chikara Hashimoto, Ryu Iida, Masahiro Tanaka, and Julien Kloetzer. 2016. A semi-supervised learning approach to why-question answering. In *Thirtieth AAAI Conference on Artificial Intelligence*.

Amir Pouran Ben Veyseh, Minh Van Nguyen, Franck Dernoncourt, and Thien Nguyen. 2022. MINION: a large-scale and diverse dataset for multilingual event detection. In *Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 2286–2299, Seattle, United States. Association for Computational Linguistics.

Rashmi Prasad, Nikhil Dinesh, Alan Lee, Eleni Miltsakaki, Livio Robaldo, Aravind Joshi, and Bonnie Webber. 2008. The penn discourse treebank 2.0. In *Proceedings of the Sixth International Conference on Language Resources and Evaluation (LREC'08)*.

Zeinab Rahimi and Mehrnoush Shamsfard. 2021. Persian causality corpus (percause) and the causality detection benchmark. *CoRR*, abs/2106.14165.

Ines Rehbein and Josef Ruppenhofer. 2020. A new resource for German causal language. In *Proceedings of the 12th Language Resources and Evaluation Conference*, pages 5968–5977, Marseille, France. European Language Resources Association.

Jawad Sadek, Farid Meziane, et al. 2018. Building a causation annotated corpus: the salford arabic causal bank-proclitics. In *Proceedings of the Eleventh International Conference on Language Resources and Evaluation (LREC 2018)*. European Language Resources Association (ELRA).

Stefan Schweter. 2020. Berturk - bert models for turkish.

Dafna Shahaf and Carlos Guestrin. 2010. Connecting the dots between news articles. In *Proceedings of the 16th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '10, page 623–632, New York, NY, USA. Association for Computing Machinery.

Robyn Speer, Joshua Chin, and Catherine Havasi. 2017. Conceptnet 5.5: An open multilingual graph of general knowledge. In *Thirty-first AAAI conference on artificial intelligence*.

Pontus Stenetorp, Sampo Pyysalo, Goran Topić, Tomoko Ohta, Sophia Ananiadou, and Jun'ichi Tsujii. 2012. brat: a web-based tool for NLP-assisted text annotation. In *Proceedings of the Demonstrations at the 13th Conference of the European Chapter of the Association for Computational Linguistics*, pages 102–107, Avignon, France. Association for Computational Linguistics.

Minh Tran Phu and Thien Huu Nguyen. 2021. Graph convolutional networks for event causality identification with rich document-level structures. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 3480–3490, Online. Association for Computational Linguistics.

Amir Pouran Ben Veyseh, Nicole Meister, Seunghyun Yoon, Rajiv Jain, Franck Dernoncourt, and Thien Huu Nguyen. 2022. Macronym: A large-scale dataset for multilingual and multi-domain acronym extraction. In *Proceedings of the 29th International Conference on Computational Linguistics*.

Christopher Walker, Stephanie Strassel, Julie Medero, and Kazuaki Maeda. 2006. Ace 2005 multilingual training corpus. In *Technical report, Linguistic Data Consortium*.

Bonnie Webber, Rashmi Prasad, Alan Lee, and Aravind Joshi. 2019. The penn discourse treebank 3.0 annotation manual. *Philadelphia, University of Pennsylvania*.

Phillip Wolff. 2007. Representing causation. *Journal of experimental psychology: General*, 136(1):82.

Xinyu Zuo, Pengfei Cao, Yubo Chen, Kang Liu, Jun Zhao, Weihua Peng, and Yuguang Chen. 2021a. Improving event causality identification via self-supervised representation learning on external causal statement. In *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*, pages 2162–2172, Online. Association for Computational Linguistics.

Xinyu Zuo, Pengfei Cao, Yubo Chen, Kang Liu, Jun Zhao, Weihua Peng, and Yuguang Chen. 2021b. LearnDA: Learnable knowledge-guided data augmentation for event causality identification. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 3558–3571, Online. Association for Computational Linguistics.

Xinyu Zuo, Yubo Chen, Kang Liu, and Jun Zhao. 2020. KnowDis: Knowledge enhanced data augmentation for event causality detection via distant supervision. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 1544–1550, Barcelona, Spain (Online). International Committee on Computational Linguistics.

Method Entity Extraction from Biomedical Texts

Waqar Bin Kalim
Department of Computer Science
The University of Western Ontario
London, Ontario, Canada
waqaar.199@gmail.com

Robert E. Mercer
Department of Computer Science
The University of Western Ontario
London, Ontario, Canada
mercer@csd.uwo.ca

Abstract

In the field of Natural Language Processing (NLP), extracting method entities from biomedical text has been a challenging task. Scientific research papers commonly consist of complex keywords and domain-specific terminologies, and new terminologies are continuously appearing. In this research, we find method terminologies in biomedical text using both rule-based and machine learning techniques. We first use linguistic features to extract method sentence candidates from a large corpus of biomedical text. Then, we construct a silver standard biomedical corpus composed of these sentences. With a rule-based method that makes use of the Stanza dependency parsing module, we label the method entities in these sentences. Using this silver standard corpus we train two machine learning algorithms to automatically extract method entities from biomedical text. Our results show that it is possible to develop machine learning models that can automatically extract method entities to a reasonable accuracy without the need for a gold standard dataset.

1 Introduction

Method Entity Extraction from unstructured biomedical text has been an important, yet challenging and somewhat under-examined, Natural Language Processing (NLP) task. Especially in the field of biomedicine, the automatic extraction of methodology names and terminologies is imperative. With thousands of research papers published each week, the biomedical research community is constantly creating new terminologies. As a result, it becomes difficult for researchers to find relevant information.

Wang et al. (2022) give a good discussion of method entities and provide the following definition: "named entities that represent specific methods". More specifically, "method entities in the academic literature are nouns or noun phrases representing the specific ways, means, and channels used to solve tasks or problems proposed by the authors, including sub-categories such as discipline-specific methods, software, models, algorithms, and metrics".

Biomedical method entities are a type of biomedical named entities. Biomedical named entity recognition (NER) is defined as the task of recognizing and categorizing entity names in the biomedical domain (Lee et al., 2004). As mentioned by Song et al. (2018), biomedical NER faces difficulties for various reasons. The first one is the increasing rate of newly created terminologies and keywords requiring new rules and patterns to be manually added to the rule-based methods, which can be a tedious and time-consuming task. Regarding biomedical method entity recognition, because it has been little studied (e.g., (Houngbo and Mercer, 2012)), a comprehensive definition of what constitutes a biomedical method entity beyond that given by Wang et al. (2022) still needs to be determined. Secondly, with information extraction tasks, the same words can have different meanings and significance in terms of the context.

In this study, our main contribution is the exploration of a rule-based approach to create a silver standard corpus annotated for method entities[1]. Regarding the rule-based approach, rules to extract method entities based on patterns of universal dependency relations (Universal Dependencies, 2014) between words will be designed. To evaluate the machine-made silver standard corpus, this corpus will be used as training data for two machine learning models, Conditional Random Fields (CRF) (Lafferty et al., 2001) and BiLSTM (Graves and Schmidhuber, 2005). CRF was used by Houngbo and Mercer (2012) and our CRF results are compared to their results as a baseline. The BiLSTM results indicate that a larger corpus will be needed

[1]The corpus and code to create it are available at https://github.com/waqarkalim/method-mention-extraction-from-biomedical-text

Proceedings of the 29th International Conference on Computational Linguistics, pages 2357–2362
October 12–17, 2022.

for neural learning models.

The structure of the remainder of this paper is as follows. In Section 2, we will review the background and related work. In Section 3, we will provide our research contributions. We will review the methodology of our research in Section 4. In Section 5, we will review the results. And in Section 6, we will conclude the paper and suggest directions for the continuation of this study.

2 Background and Related Work

Named Entity Recognition (NER) is an application of Natural Language Processing (NLP) where entities are tagged according to various semantic and syntactic rules. Surveys of research in the field of NER (Nadeau and Sekine, 2007; Wang et al., 2022) indicate that automatic NER extraction has received significant coverage in the past three decades. For the study reported in the current paper, we are interested in the extraction of a subcategory of named entities, method entities in particular, from biomedical literature. In the biomedical literature, extraction of named entities has tended to focus on biological entities (for example, Settles (2004) and Habibi et al. (2017)). Extraction of method entities, in particular, has received some attention, but the vast majority extract this type of named entity from non-biomedical literature (Wang et al., 2022). One exception is Houngbo and Mercer (2012) who deal with method entities in biomedical research articles. Lam et al. (2016) extract methodology terms as well as sleep disorder entities from biomedical literature that focusses on sleep disorders. Zhao et al. (2019) propose a new annotation scheme, manually annotate 3,088 resource citations (algorithms are the only methods included in these resources) found in biomedical and non-biomedical literature, and use BiLSTM as the machine learning method.

One aspect of this current work is to create a silver standard corpus for method entities in biomedical text. The automatic method to generate this corpus will use Stanza's dependency parser (Qi et al., 2020) with the biomedical packages (Zhang et al., 2021). This parser produces graph structures whose edges are labelled with universal dependencies (Universal Dependencies, 2014).

To evaluate using the silver standard corpus, we will use Conditional Random Fields (Lafferty et al., 2001) and BiLSTM (Graves and Schmidhuber, 2005), the machine learning techniques used in other studies (Houngbo and Mercer, 2012; Chiu and Nichols, 2016).

3 Research Contributions

Few researchers have addressed the question of automatic method entity extraction from biomedical text. Generating a human-labelled corpus is time-consuming. This study aims to address these issues.

1. We have created rule-based methods to use Stanza's universal dependencies while extracting a wider variety of method entities compared to Houngbo and Mercer (2012) and have successfully created an accurate silver standard corpus (precision score of 97.59) prepared from full text biomedical articles selected from the PubMed Central dataset with method entities annotated automatically.

2. By training on this silver corpus, we have improved on the performance benchmarks provided by Houngbo and Mercer (2012).

4 Methodology

Because Houngbo and Mercer (2012) is the only previous study that investigates the same problem, we use that work as the baseline and compare our results with those reported there.

In the initial stage of the study, we prepare a collection of sentences that contain mentions of method names, or "method sentences", using the method proposed by Houngbo and Mercer (2012). By employing the properties of anaphoric relations between sentences, we can collect the "method sentences" in a convenient and feasible manner. We collect these sentences by scanning through research papers using the Unix command grep and selecting some number of sentences that precede any sentence containing the words "this method". This approach successfully generates a corpus containing solely "method sentences".

After the corpus creation has been completed, the next stage involves utilizing linguistic rules and patterns to automatically label method entities. In this study, for tagging the entities, we will be using the IOB tagging format. In the IOB tagging scheme, every token is labelled as B-label if the token is the beginning of a named entity, I-label if it is inside a named entity but not the first token within the named entity, or O otherwise (Lample

et al., 2016). Leveraging rule-based methods allows for labelling the method mentions without the use of any pre-existing training data; additionally, a secondary benefit of this approach is the potential of introducing new rules and patterns based on the linguistic features of the terminologies. This step is an essential aspect of our research as it allows for implementing an accurate silver standard dataset.

After the rule-based methods have been applied, traditional and neural learning techniques can be explored in combination with the newly developed silver standard dataset. The primary benefit of utilizing a machine learning approach is the ability to generalize beyond the limits of the rules and patterns that are manually defined in the rule-based approach. In this stage of the research, we will explore two machine learning algorithms related to Natural Language Processing tasks: Conditional Random Field (CRF) models (Lafferty et al., 2001) and the Bidirectional Long Short Term Memory (BiLSTM) models (Graves and Schmidhuber, 2005). We opted for choosing these algorithms as CRF models are trained for sequence segmentation and labelling and BiLSTM models have been used in other named entity extraction research (Chiu and Nichols, 2016; Zhao et al., 2019).

5 Results

5.1 Silver Corpus Creation

In this study, we curated a collection of 2,839 biomedical research papers from which to derive our method entity silver corpus. Based on the findings reported by Torii and Vijay-Shanker (2005) that nearly all antecedents can be found within two sentences from the demonstrative anaphors, we used the technique suggested by Houngbo and Mercer (2012) and employed the anaphoric relations between sentences to find the "method sentence" candidates. So, to generate our corpus, we searched through our collection of papers for sentences that begin with the anaphor "This method" and then selected the three sentences that precede it for our corpus. By selecting three sentences rather than two, we achieve an extra layer of certainty that the selected sentences contain at least one method entity. As a result, we retrieved 10,974 potential "method sentences".

An example of a retrieval is:

Sentence 1: In tracheal samples, YCW increased concentrations of mucosal IgA compared to Control ($P < 0.05$).

Sentence 2: No significant differences were observed between Vaccine and Coccidiostat.

Sentence 3: The effect of different treatments on cell-mediated immune response was examined by the cutaneous basophilic hypersensitivity test.

Sentence 4: This method reveals the status of the T-cell response.

In this example, Sentence 4 contains the "this method" anaphor. The sentences found in the silver corpus would be Sentences 1, 2, and 3. Sentence 3 contains the antecedent "cutaneous basophilic hypersensitivity test" which is a method entity.

A manual investigation of these sentences suggests that most of the method entities in our corpus are sequences of words represented by the following examples:

- Tukey's biweight method
- naive KNN method
- 10-fold cross-validation test
- Roche Amplicor Cystic Fibrosis test
- bimolecular fluorescence complementation analysis
- Felsenstein's independent comparison method
- statistical total correlation spectroscopy analysis method
- MANOVA-based scoring method
- protein sequence Jukes-Cantor model
- utaneous basophilic hypersensitivity test

From these examples, we can observe how the rules and patterns to extract our method entities would look like. First, all of these mentions end with key suffixes (as observed by Houngbo and Mercer (2012)) that would correspond to most method entities: method, analysis, test, model, algorithm, etc. In addition, we have dependency-parsed the method mention candidate sentences with Stanza (Qi et al., 2020) using the biomedical and clinical model packages included in the Stanza toolkit (Zhang et al., 2021). Investigating these dependency parses, all of these method entities have at least one universal dependency (UD) compound relation, most of them have at least one amod UD relation, and some of them have at least one nmod:poss UD relation. A compound UD is a modifier that relates to a noun and itself is a noun, whereas an amod UD is an adjectival modifier that serves to modify a noun or pronoun but itself is an adjective. An nmod:poss UD is a modifier that serves to show possessives. After generating the corpus which contains 10,974 potential "method

sentences", we used linguistic rules and patterns to programmatically extract the method entities depending on the dependency relationships between the words. Based on these observations, we created three rules based solely on these UD relationships. These three rules were able to find 1338 method entities in our corpus in total; 629 for Rule 1, 680 for Rule 2, and 29 for Rule 3. The rules work as follows:

Rule 1: In a sentence, if there is a subtree with at least one compound relation, retrieve all the words between the first compound word to the last word of that subtree plus the subtree root (i.e., one of the key suffixes mentioned above) as a method entity.

Rule 2: In a sentence, if there is a subtree with exactly one compound relation and at least one amod relation, retrieve all the words between the first amod/compound word to the last word of that subtree plus the subtree root as a method entity.

Rule 3: In a sentence, if there is a subtree with exactly one nmod:poss relation, retrieve all the words between the nmod:poss word to the last word of that subtree plus the subtree root as a method entity.

The rules stated above are different from the rule used by Houngbo and Mercer (2012), a regular expression composed of POS tags and key suffixes, to find the method entities; whereas the rules in this study use universal dependencies provided by Stanza pre-trained on biomedical text.

With these rules, the rule-based model was able to achieve a precision score of 97.59, which is better than expected. Unfortunately, due to the sheer amount of data in our corpus, we were unable to manually determine the recall score, and accordingly, an F-1 score for our rule-based approach.

Using this rule-based model, we tag the extracted labels using IOB tagging in order to create our silver standard dataset which can be used in Section 5.2.

5.2 Machine-learning Approach

When the rule-based method to label the silver standard corpus has completed, we are now ready to investigate the machine-learning techniques. This study will investigate two machine learning models: 1) a traditional Conditional Random Field (CRF) model, and 2) a neural Bidirectional Long Short Term Memory (BiLSTM) model. Both are supervised machine learning models which require training data that is labelled with the feature to be learned. As our training dataset, we will use the silver standard corpus from Section 5.1.

System	P	R	F1
Conditional Random Field	83.58	85.49	84.53
Houngbo and Mercer (2012) CRF	81.80	75.00	78.26
BiLSTM	68.42	39.39	50.00

Table 1: Precision (P), Recall (R), and F1-Score (F1) for the Machine Learning Methods

CRF models are a framework for developing probabilistic models for segmenting and labelling sequence data (Lafferty et al., 2001). BiLSTM models (Graves and Schmidhuber, 2005) are a form of recurrent neural networks that can understand the context of a sentence quite well. BiLSTM models work well for NLP tasks as they can contextually scan through text in both forward and backward directions. For the word embeddings, the BiLSTM model uses BioWordVec, an open set of biomedical word embeddings that combines subword information learned from unlabeled biomedical text with a widely-used biomedical controlled vocabulary (Zhang et al., 2019).

Table 1 shows the results for each of the machine learning models. We observe from Table 1 that the highest performing machine learning model in this study (CRF) outperforms the machine learning model results of Houngbo and Mercer (2012) by a precision score of 1.78 pp, a recall score of 10.49 pp, and an F-1 score of 6.27 pp.

Our findings are based on inaccurate metrics, so the results should thus be treated with some caution. However, because this inaccuracy is due to the true positives being thought of as false positives, the actual precision and recall should be higher than what is displayed in Table 1. As an example, the CRF model produced 20 predictions that were labelled as not method entities in the Houngbo and Mercer (2012) gold-standard testing data, however, a manual check shows that 17 out of those 20 predictions actually are method entities.

The results for BiLSTM are lower than what would be predicted by other research that has used this neural architecture (Zhao et al., 2019). We believe that this is due to insufficient training samples. So, we did not investigate any other BiLSTM-based architectures, leaving this to future studies when we have built a larger silver corpus.

6 Conclusions and Future Work

In this paper, we explored various methodologies to automatically extract method entities from biomedical text. In the initial step, we created a corpus containing candidate method sentences using anaphoric relations. Next, we investigated a rule-based method using information provided by a dependency parse to IOB tag the method entities in the corpus. This silver standard corpus is the main contribution of our work and has been made publicly available. To evaluate the quality of this corpus, we trained two machine learning methods using this silver standard corpus to automatically extract method entities from biomedical text.

The evidence from this study shows that using a dependency parser that is pre-trained on biomedical vocabulary allows for precise extraction of method entities within the scope of the rules as shown in Section 5.1. Additionally, the results from Section 5.2 and Table 1 show how the CRF model outperforms the results from Houngbo and Mercer (2012) and show the potential of machine learning models to accurately generalize outside the scope of the rules defined in Section 5.1.

Our future work will include

1. improving on the anaphoric method to gather candidate method entity sentences.

2. creating a wider variety of rules and patterns for our rule-based approach to create a more comprehensive silver standard corpus. In this study we noticed phrases such as "a method that uses regular expressions to look for section headings" and *method phrase* followed by "developed by *one or more names*" (e.g., "Robust Multi-Array Analysis developed by Irizarry") which is like the possessive form in our Rule 3 above. These more complex patterns would be amenable to the dependency parse methodology. However, these patterns may not be amenable to an IOB-type annotation. And they include but do not end with one of the key suffixes (i.e., method, analysis, test, model, algorithm). So, this move may not be a simple extension of what was presented here.

3. developing a larger silver standard corpus. The BiLSTM results strongly suggest that the corpus is too small for neural learning methods.

4. detailing aspects of the silver standard corpus. Understanding what makes this corpus better than previous ones will inform further development of a definition of the text span of a method entity.

5. investigating more sophisticated machine learning models, such as a BiLSTM-CRF model and a BiLSTM-CNN-CRF model, to better evaluate the silver standard corpus and to improve the method entity performance beyond what has been achieved in previous works. Adding a CRF layer on top of a BiLSTM model, as well as adding a CNN and CRF layer on top of a BiLSTM model have proven to improve performance in a few sequence labelling problems.

Acknowledgements

This research is partially funded by The Natural Sciences and Engineering Research Council of Canada (NSERC) through a Discovery Grant to Robert E. Mercer. We also acknowledge the helpful comments provided by the reviewers.

References

Jason P. C. Chiu and Eric Nichols. 2016. Named entity recognition with Bidirectional LSTM-CNNs. *Transactions of the Association for Computational Linguistics*, 4:357–370.

Alex Graves and Jürgen Schmidhuber. 2005. Framewise phoneme classification with bidirectional LSTM and other neural network architectures. *Neural Networks*, 18(5–6):602–610.

Maryam Habibi, Leon Weber, Mariana Neves, David Luis Wiegandt, and Ulf Leser. 2017. Deep learning with word embeddings improves biomedical named entity recognition. *Bioinformatics*, 33(14):i37–i48.

Hospice Houngbo and Robert E. Mercer. 2012. Method mention extraction from scientific research papers. In *Proceedings of COLING 2012*, pages 1211–1222.

John D. Lafferty, Andrew McCallum, and Fernando C. N. Pereira. 2001. Conditional random fields: Probabilistic models for segmenting and labeling sequence data. In *Proceedings of the Eighteenth International Conference on Machine Learning (ICML '01)*, pages 282–289.

Calvin Lam, Fu-Chi Lai, Chia-Hui Wang, Mei-Hsin Lai, Nanl Hsu, and Min-Huey Chung. 2016. Text mining of journal articles for sleep disorder terminologies. *PLoS One*, 11(5):e0156031.

Guillaume Lample, Miguel Ballesteros, Sandeep Subramanian, Kazuya Kawakami, and Chris Dyer. 2016. Neural architectures for named entity recognition. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 260–270.

Ki-Joong Lee, Young-Sook Hwang, Seonho Kim, and Hae-Chang Rim. 2004. Biomedical named entity recognition using two-phase model based on SVMs. *Journal of Biomedical Informatics*, 37(6):436–447.

David Nadeau and Satoshi Sekine. 2007. A survey of named entity recognition and classification. *LingvisticæInvestigationes*, 30(1):3–26.

Peng Qi, Yuhao Zhang, Yuhui Zhang, Jason Bolton, and Christopher D. Manning. 2020. Stanza: A Python natural language processing toolkit for many human languages. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics: System Demonstrations*, pages 101–108.

Burr Settles. 2004. Biomedical named entity recognition using conditional random fields and rich feature sets. In *Proceedings of the International Joint Workshop on Natural Language Processing in Biomedicine and its Applications (JNLPBA '04)*, pages 104–107.

Hye-Jeong Song, Byeong-Cheol Jo, Chan-Younng Park, Jong-Dae Kim, and Yu-Seop Kim. 2018. Comparison of named entity recognition methodologies in biomedical documents. *BioMedical Engineering On-Line*, 17(158).

Manabu Torii and K. Vijay-Shanker. 2005. Anaphora resolution of demonstrative noun phrases in medline abstracts. In *Proceedings of the Conference Pacific Association for Computational Linguistics (PACLING)*, pages 332–339.

Universal Dependencies. 2014. Universal dependencies. https://universaldependencies.org. Accessed: 2021-04-30.

Yuzhuo Wang, Chengzhi Zhang, and Kai Li. 2022. A review on method entities in the academic literature: extraction, evaluation, and application. *Scientometrics*.

Yijia Zhang, Qingyu Chen, Zhihao Yang, and Zhiyong Lu. 2019. BioWordVec, improving biomedical word embeddings with subword information and MeSH. *Scientific Data*, 6, Article number: 52.

Yuhao Zhang, Yuhui Zhang, Peng Qi, Christopher D. Manning, and Curtis P. Langlotz. 2021. Biomedical and clinical English model packages for the Stanza Python NLP library. *Journal of the American Medical Informatics Association*, 28(9):1892–1899.

He Zhao, Zhunchen Luo, Chong Feng, and Yuming Ye. 2019. A context-based framework for resource citation classification in scientific literatures. In *Proceedings of the 42nd International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR'19)*, pages 1041–1044.

Optimal Partial Transport based Sentence Selection for Long-form Document Matching

Weijie Yu[1], **Liang Pang**[4], **Jun Xu**[2,3,*], **Bing Su**[2,3], **Zhenhua Dong**[5] and **Ji-Rong Wen**[2,3]
[1]School of Information, Renmin University of China
[2]Gaoling School of Artificial Intelligence, Renmin University of China
[3]Beijing Key Laboratory of Big Data Management and Analysis Methods
[4]Institute of Computing Technology, Chinese Academy of Sciences
[5]Noah's Ark Lab, Huawei
{yuweijie, junxu, bingsu, jrwen}@ruc.edu.cn, pangliang@ict.ac.cn, dongzhenhua@huawei.com

Abstract

One typical approach to long-form document matching is first conducting alignment between cross-document sentence pairs and then aggregating all of the sentence-level matching signals. However, this approach could be problematic because the alignment between documents is *partial* — despite two documents as a whole are well-matched, most of the sentences could still be dissimilar. Those dissimilar sentences lead to spurious sentence-level matching signals which may overwhelm the real ones, increasing the difficulties of learning the matching function. Therefore, accurately selecting the key sentences for document matching is becoming a urgent problem. To address this issue, we propose a novel matching approach that equips existing document matching models with an Optimal Partial Transport (OPT) based component, namely OPT-Match, which selects the sentences that play a major role in matching. Enjoying the partial transport properties of OPT, the selected key sentences can not only effectively enhance the matching accuracy, but also be explained as the rationales for the matching results. Extensive experiments on four publicly available datasets demonstrated that existing methods equipped with OPT-Match consistently outperformed the corresponding underlying methods. Evaluations also showed that the sentences selected by OPT-Match were consistent with human-provided rationales.

1 Introduction

Long-form document matching, which identifies the semantic relationship between a source document and a target document, has become a fundamental problem in both NLP and IR. Representative tasks include cite recommendation (Jiang et al., 2019) and plagiarism detection (Foltýnek et al., 2020) etc. For example, in cite recommendation, the document matching has been used to recommend existing papers to be cited in a new paper. In plagiarism detection, the document matching model has been used to determine whether a paper is plagiarized from another paper.

Document 1: ... *the attraction of medicinal and aromatic plants as worthy farm crops has grown due to the demand created by consumer interest in these plants for culinary, medicinal, and other anthropogenic applications* … an understanding of future opportunities in the medicinal and aromatic plant industry who 2003 is necessary to enable us growers to envision and invest in medicinal and aromatic crops that will meet market demands.

Document 2: *medicinal and aromatic plants are nowadays becoming the main constituents for medicines, colorings, preservatives, and fragrances*… although it is of high priority, the effect of slowgrowth conservation on the chemical profile of prestored medicinal plants after conservation was not extensively investigated .this research aimed to analyze in vitro stored thymbra spicata l. var. …

Figure 1: A pair of matched long-form documents from S2ORC dataset. Documents 1 focuses on the future opportunities in the medicinal and aromatic plant industry. Document 2 studies the vitro storage of spicata. Though most sentences are not similar, Document 1 cites Document 2 (matched) because they both take medicinal and aromatic plants as examples (the highlighted sentences).

Existing approaches to long-form document matching typically follow the paradigms developed for short-text (sentence) matching, i.e., they conduct matching based on all of the sentences in the documents. For example, Jiang et al. (2019); Pappagari et al. (2019); Zhou et al. (2020) map the documents into a latent semantic space from a hierarchical perspective (e.g., word, sentence, paragraph) and conduct matching in the semantic space. However, these methods ignore the fact that a long-form document usually contains multiple paragraphs and sentences, which convey complex and diverse semantics. Unlike short-text matching where almost every word-level matching signal matters, in long-form document matching, the alignment between a document pair is partial and a few matching signals

* Corresponding author

Proceedings of the 29th International Conference on Computational Linguistics, pages 2363–2373
October 12–17, 2022.

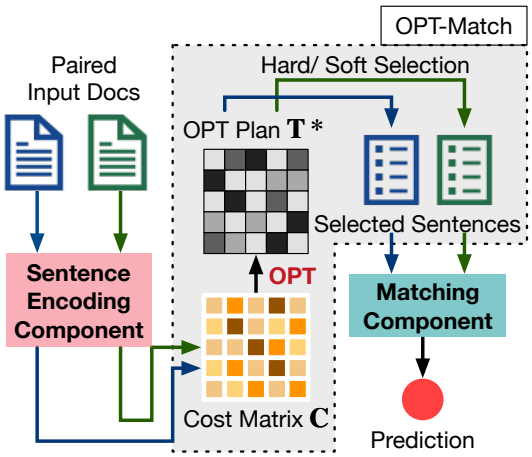

Figure 2: The architecture of OPT-Match. The sentence encoding component and the matching component are with existing methods (the underlying models).

between the key sentences can determine the matching result at the document level. Figure 1 illustrates a typical example: two matched paper abstracts (*Document 1* cites *Document 2*) from the Semantic Scholar Open Research Corpus (S2ORC) (Lo et al., 2020). These two abstracts are matched because of the highly matched signals between the highlighted key sentences. Other parts of the abstracts are not important for matching. This observation inspires us that it is unnecessary to force a matching model to aggregate all of the sentence pairs. More importantly, the introduced noise may overwhelm the matching signals between key sentences. Thus, identifying key sentences is becoming an essential step for document matching.

In this paper, we propose a novel Optimal Partial Transport (OPT) (Figalli, 2010) based sentence selection component for existing long-form document matching model, namely OPT-Match. As illustrated in Figure 2, OPT-Match models the document alignment as an OPT process by regarding two input documents (two sets of sentences) as two piles of earth. To achieve partial alignment, OPT-Match poses a limitation on how much the earth from one pile (*Document 1*) needs to be transported to the other pile (*Document 2*). Based on the cost matrix whose elements are defined as inverse (or negative) similarities between sentences from different documents, OPT-Match generates an optimal transport plan (matrix) (Benamou et al., 2015) which indicates the alignment of sentences. Therefore, key sentences can be extracted based on the transport plan. To make OPT-Match easily incorporated into existing document matching models, we provide two strategies to achieve the sentence selection.

Compared to existing OT-based methods (Kusner et al., 2015; Chen et al., 2018, 2019; Zhang et al., 2020), OPT-Match offers the following advantages in modeling the long-form document alignment: 1) OPT-Match models the partial nature of document alignment explicitly and flexibly, through limiting how much the mass to be transported; 2) OPT-Match allows the source and the target domains are not necessarily with the same total mass, which fits well with the phenomenon that the lengths of two documents may vary greatly. The OT-based methods, however, cannot take this into consideration; 3) OPT-Match is a model-agnostic approach, it can be easily plugged into a wide range of document matching models.

To summarize, this paper makes the following main contributions: (1) We highlight the critical importance of the key sentence selection in long-form document matching, which has not yet been thoroughly studied in existing models. (2) We propose a wide applicable component called OPT-Match, which selects key sentences for document matching by conducting partial document alignments. (3) We conducted extensive empirical studies on four large-scale publicly available datasets. The experimental results demonstrated that OPT-Match improved existing document matching models and the sentences selected by OPT-Match were consistent with human rationales.

2 Proposed OPT-Match Method

2.1 Problem Statement

Suppose that we are given a set of labeled data tuples: $\mathcal{D} = \{(X_i, Y_i, z_i)\}$, where the elements in the i-th training instance $X_i \in \mathcal{X}$, $Y_i \in \mathcal{Y}$, and $z_i \in \mathcal{Z}$ respectively denote the source document, the target document, and the label which indicates the semantic relationship of X_i and Y_i. Both the source and the target documents consist of a number of sentences, i.e., $X_i = [s_1^{X_i}, s_2^{X_i}, \ldots, s_M^{X_i}]$, $Y_i = [s_1^{Y_i}, s_2^{Y_i}, \ldots, s_N^{Y_i}]$, where the M and N denote the number of sentences in X_i and Y_i respectively. The learning objective of existing document matching models is $f : \mathcal{X} \times \mathcal{Y} \rightarrow \mathcal{Z}$, which takes all sentences in the input documents as input and outputs a prediction of the relationship between them. As the key idea of this paper, we aim at learning f' which selects the key sentences $S^X \subseteq X, S^Y \subseteq Y$ from input documents and con-

ducts matching based on those selected sentences rather than all sentences.

2.2 The Principle of Our Method

To learn f', we develop a novel sentence selection method from the viewpoint of optimal partial transport (OPT), which is an extension of optimal transport (OT). Originally, OT defines a distance between probability distributions. Given two probability distributions $\boldsymbol{\mu}$ and $\boldsymbol{\nu}$ which can be viewed as two piles of earth with equivalent mass, the optimal transport distance is defined as the minimum cost of turning one pile into the other, and the corresponding optimal transport plan provides a soft matching between two piles in a probabilistic way. Existing OT-based NLP studies (Kusner et al., 2015; Chen et al., 2018, 2019; Zhang et al., 2020) set the distributions $\boldsymbol{\mu}$ and $\boldsymbol{\nu}$ uniform, i.e., $\boldsymbol{\mu} = \frac{1}{M}\mathbf{1}_M$ and $\boldsymbol{\nu} = \frac{1}{N}\mathbf{1}_N$, where $\mathbf{1}_D$ represents the D-dimensional all-one vector, and accordingly, the optimal transport distance between them is:

$$\begin{aligned}\mathbf{T}^* &= \arg\min_{\mathbf{T}\in\Pi(\boldsymbol{\mu},\boldsymbol{\nu})} \mathbb{E}_{m,n\sim\mathbf{T}}[c(s_m^X, s_n^Y)] \\ &= \arg\min_{\mathbf{T}\in\Pi(\boldsymbol{\mu},\boldsymbol{\nu})} \sum_{m=1}^{M}\sum_{n=1}^{N} T_{mn}\cdot c(s_m^X, s_n^Y),\end{aligned} \tag{1}$$

where $\mathbf{T} \in \Pi(\boldsymbol{\mu},\boldsymbol{\nu}) = \{\mathbf{T} \in \mathbb{R}_+^{M\times N} | \mathbf{T}\mathbf{1}_N = \boldsymbol{\mu}, \mathbf{T}^\top\mathbf{1}_M = \boldsymbol{\nu}\}$ represents an arbitrary joint distribution of the sentences with marginals $\boldsymbol{\mu}$ and $\boldsymbol{\nu}$. $\mathbf{C} = [c(s_m^X, s_n^Y)] \in \mathbb{R}^{M\times N}$ is a sentence-level cost matrix, whose element $c(s_m^X, s_n^Y)$ measures the discrepancy between the two sentences. As shown in Eq. (1), OT corresponds to the minimum expectation of the sentence-level discrepancy, and thus shares a similar spirit with existing methods, which aggregate all sentence-level matching signals.

However, OT suffers from the following issues in modeling long-form document matching: 1) OT requires that $\boldsymbol{\mu}$ and $\boldsymbol{\nu}$ have identical total mass. This setting is unsuitable for document matching because the number of sentences in documents may vary greatly and the lengthy document contains more semantics in general. 2) OT requires the source points must exactly map to the targets. However, in document matching, only some key sentences from the source document align to that from the target document, and thus there should be only a fraction of mass from the source should be transported to the target. 3) OT aggregates all sentence-level aligning signals which inevitably involves noises to the matching. In this work, we solve these issues by modeling the sentence-level alignment as an OPT process (issue 1,2) and incorporating it into existing document matching methods as a sentence selection module (issue 3).

2.3 OPT-based Sentence-level Alignment

To fix the issue 1 and 2, we need to break the constraint that $\boldsymbol{\mu}$ and $\boldsymbol{\nu}$ must have the same total mass and limit the transporting mass, which leads to an OPT problem:

$$\mathbf{T}^* = \arg\min_{\mathbf{T}\in\Pi_{\leq}(\boldsymbol{\mu},\boldsymbol{\nu}),\mathbf{1}_M^\top\mathbf{T}\mathbf{1}_N=\epsilon} \langle \mathbf{T}, \mathbf{C}\rangle. \tag{2}$$

where $\mathbf{T} \in \Pi_{\leq}(\boldsymbol{\mu},\boldsymbol{\nu}) = \{\mathbf{T}\mathbf{1}_N \leq \boldsymbol{\mu}, \mathbf{T}^\top\mathbf{1}_M \leq \boldsymbol{\nu}\}$, $<,>$ represents the Frobenius dot-product. For the issue 1, considering that in OPT, $\boldsymbol{\mu}$ and $\boldsymbol{\nu}$ are not necessarily with the same total mass (Benamou et al., 2015) and usually the longer documents contains more semantics, we set a unit mass on each sentence, i.e., $\boldsymbol{\mu} = \mathbf{1}_M, \boldsymbol{\nu} = \mathbf{1}_N$. For the issue 2, as shown in Eq. (2), we set ϵ, indicating a proportion of total mass $\min(||\boldsymbol{\mu}||_1, ||\boldsymbol{\nu}||_1)$ to be transported, to control the degree of the document alignment. Intuitively, with the lower ϵ, OPT-Match focuses more on strongly aligned sentence pairs, while filtering out more spurious alignment signals.

To measure the discrepancy between two cross-document sentences, we define the cost matrix $\mathbf{C}$ in Eq. (2) based on the similarity between sentence pairs derived from (X, Y):

$$c(s_m^X, s_n^Y) = -\text{sim}(s_m^X, s_n^Y), \tag{3}$$

$\text{sim}(s_m^X, s_n^Y)$ is the similarity between s_m^X and s_n^Y. Intuitively, we expect more similar pairs of sentences to be transported at a lower cost, and thus can be more strongly aligned. In Eq. (3), various methods can be adopted to measure $\text{sim}(s_m^X, s_n^Y)$, leading to different kinds of cost, for example, the cosine similarity between sentence embeddings, and the overlapping words ratio after removing stop-words of (s_m^X, s_n^Y) (Mihalcea and Tarau, 2004). We respectively apply these two methods to soft selection strategy and hard selection strategy which we will introduce in Sec. 2.4 and Sec. 2.5.

To solve the OPT problem in Eq. (2), a number of algorithms have been proposed. As a representative method, Benamou et al. (2015) propose to add an entropic regularizer (Cuturi, 2013; Xie et al., 2018) and solve it with iterative Bregman projections (Bregman, 1967) and Dykstra algorithm (Dykstra, 1983). See also (Chizat et al., 2018; Zhou et al., 2020). For the fast approximation of OPT, an

entropic regularizer $E(\mathbf{T})$ (Cuturi, 2013) is added and the the optimal transport plan is

$$\mathbf{T}^* = \underset{\mathbf{T}\in\Pi_{\leq}(\boldsymbol{\mu},\boldsymbol{\nu}),\mathbf{1}_M^\top\mathbf{T}\mathbf{1}_N=\epsilon}{\arg\min} \langle \mathbf{T}\cdot\mathbf{C}\rangle + \lambda E(\mathbf{T}), \quad (4)$$

where λ is the trade-off coefficient. Thus, the optimal partial transport plan $\mathbf{T}^*$ can be iteratively calculated by Bregman-Dykstra iterations (Bregman, 1967; Dykstra, 1983; Benamou et al., 2015):

$$\begin{aligned}
\mathbf{T}_n^1 &= \text{diag}\left(\min\left(\frac{\boldsymbol{\mu}}{\mathbf{T}_{n-1}\mathbf{1}}, \mathbf{1}\right)\right)\mathbf{T}_{n-1}, \\
\mathbf{T}_n^2 &= \mathbf{T}_n^1 \cdot \text{diag}\left(\min\left(\frac{\boldsymbol{\nu}}{\mathbf{T}_n^{1\top}\mathbf{1}}, \mathbf{1}\right)\right), \\
\mathbf{T}_n &= \mathbf{T}_n^2 \cdot \frac{\epsilon}{\mathbf{1}^\top\mathbf{T}_n^2\mathbf{1}},
\end{aligned} \quad (5)$$

where $\mathbf{T}_0 = \exp(-\mathbf{C}/\lambda)$. $\mathbf{T}^*$ indicates the amount of probability mass moved from one pile of earth to the other under the constraint that limited mass should be transported. In the sentence alignment scenario, $\mathbf{T}^*$ can be regarded as the degree of the alignment between the source sentences and the target sentences in which only those strongly aligned sentences are be highlighted.

2.4 Sentence Selection

To fix the issue 3, we need to select sentences S^X, S^Y for matching according to $\mathbf{T}^*$. We provide two strategies to achieve that.

Hard Selection. We take an aggressive approach to filter out the noise in the document, that is, we select k sentences from the source and the target document respectively with the highest alignment in the optimal transport plan and discarding the rest of the sentences, where k is a hyper-parameter which stands for the desirable number of key sentences. Specifically, $\mathbf{T}^*$ is summed by rows, and the sentences in the source document corresponding to the top-k indexes of $\mathbf{T}^*\mathbf{1}_n$ are selected, then placed to S^X. Similarly, for the target document, S^Y is constructed based on the top-k indexes of $\mathbf{1}_n^\top\mathbf{T}^*$. Although this strategy is non-differentiable, it effectively filters out noise.

Soft Selection. To make the selection differentiable, we provide an alternative. Given $\mathbf{T}^*$ as the sampling probabilities, the key sentences are sampled using the Gumbel softmax (Jang et al., 2016), which provides a differentiable sampling process:

$$\begin{aligned}
u_i &\sim U(0,1), g_i = -\log(-\log(u_i)), \\
w_i &= \frac{\exp\left((\log(prob_i) + g_i)/\tau\right)}{\sum_j \exp\left((\log(prob_i) + g_i)/\tau\right)},
\end{aligned} \quad (6)$$

where $U(0,1)$ represents the uniform distribution between 0 and 1, and τ is a temperature hyperparameter, $prob_i$ represents the probability of choosing each sentence as the selected sentence. For the source document, $prob_i$ is normalized $\mathbf{T}^*\mathbf{1}_N$, for the target document, $prob_i$ is normalized $\mathbf{1}_M^\top\mathbf{T}^*$. Therefore, we obtain the selection weight w_i for each sentence, and the key sentence sets are $S^X = (w_1^X s_1^X, \cdots, w_M^X s_M^X)$ and $S^Y = (w_1^Y s_1^Y, \cdots, w_N^Y s_N^Y)$.

2.5 Combination with existing models

Till now, OPT-Match has extracted S^X and S^Y, a paired subset of sentences from input documents. As a widely applicable module, OPT-match can be easily combined with various document matching models. In this work, we take two representative methods as the underlying models for OPT-Match.

For models which hierarchically encode document (Jiang et al., 2019; Pappagari et al., 2019; Zhou et al., 2020), they suppose that documents present a hierarchical structure including words, sentences, and paragraphs. OPT-Match can be plugged at the sentence level, because hierarchical models explicitly represent all sentences in a document. For example, once the sentence representations are obtained, one can construct the cost matrix $\mathbf{C}$ based on the cosine similarity between sentence embeddings (Eq.3), then use the soft selection strategy. OPT-Match also can be used before sentence encoding by adopting the overlapping words ratio cost and the hard selection strategy.

For BERT and its variants (Devlin et al., 2019; Dai et al., 2019; Beltagy et al., 2020), since they focus on token-level interaction and do not explicitly generate sentence representations, OPT-Match can be regarded as a pre-processing to combining with BERT and its variants. One can use the overlapping words ratio as the OPT cost and select sentences using the hard selection strategy.

2.6 Training

As aforementioned, OPT-Match is a sentence selection module before matching, and thus the learning objective of OPT-Match equipped models is identical to its underlying models. In the underlying models, the cross-entropy loss which measures the discrepancy between the model's predictions and ground-truth labels is widely used for training:

$$\mathcal{L} = \sum_i \ell(M(S^{X_i}, S^{Y_i}), z_i) = \sum_i z_i \log p_i + (1-z_i)\log(1-p_i) \quad (7)$$

Algorithm 1 Training process of OPT-Mach based models.

Require: Training set $\mathcal{D} = \{(X_i, Y_i, z_i)\}_{i=1}^{N}$; mini-batch sizes n_b; learning rates η; Bregman-Dykstra iterations step $iter$; entropic regularizer coefficient λ; mass to be transported ϵ, number of selected sentences k, Gumbel temperature τ.

repeat
  ▷ OPT-Match component
  Sample mini-batch $\{(X_i, Y_i, z_i)\}_{i=1}^{n_b} \subseteq \mathcal{D}$
  Calculate the cost matrix $\mathbf{C}$ {Eq. (3)}
  $\mathbf{T} = \exp(-\mathbf{C}/\lambda)$
  for $t = 1$ to $iter$ **do**
    $\mathbf{T} \leftarrow \text{diag}\left(\min\left(\frac{\boldsymbol{\mu}}{\mathbf{T1}}, \mathbf{1}\right)\right)\mathbf{T}$
    $\mathbf{T} \leftarrow \mathbf{T} \cdot \text{diag}\left(\min\left(\frac{\boldsymbol{\nu}}{\mathbf{T}^\top\mathbf{1}}, \mathbf{1}\right)\right)$
    $\mathbf{T} \leftarrow \mathbf{T} \cdot \frac{\epsilon}{\mathbf{1}^\top\mathbf{T1}}$ {Eq. (5)}
  end for
  if Hard Selection **then**
    $S^X \leftarrow$ top k indexes (sentences) of $\mathbf{T1}_N$
    $S^Y \leftarrow$ top k indexes (sentences) of $\mathbf{1}_M^\top\mathbf{T}$
  else if Soft Selection **then**
    $S^X \leftarrow (w_1^X s_1^X, \cdots, w_n^X s_n^X)$ {Eq. (6)}
    $S^Y \leftarrow (w_1^Y s_1^Y, \cdots, w_m^Y s_m^Y)$ {Eq. (6)}
  end if
  ▷ Matching component M
  $\mathcal{L} = \sum_{i=1}^{n_b} \ell(M(S^X, S^Y; \boldsymbol{\Theta}), z_i)$ {Eq. (7)}
  $\boldsymbol{\Theta} \leftarrow \boldsymbol{\Theta} - \eta \bigtriangledown_{\boldsymbol{\Theta}} \mathcal{L}$
until convergence
return $\boldsymbol{\Theta}$

where z_i is the ground-truth label, $p_i = M(S^{X_i}, S^{Y_i})$ is the final matching prediction, and M represents the matching component. We include the training procedure of OPT-Match equipped matching models in Alg. 1.

3 Experiments

3.1 Experimental Setup

Datasets. The experiments are conducted on four large-scale publicly available long-form document matching datasets [1]. Table 1 provides the dataset statistics. Data splits and preprocessing for all datasets follow (Zhou et al., 2020).

Citation recommendations is a task to predict whether a paper cites another. In the experiments, AAN (Radev et al., 2013), OC (Bhagavatula et al., 2018), and S2ORC (Lo et al., 2020) are exploited for this task. Note that following the practice of (Zhou et al., 2020), we only use the paper abstract of AAN. S2ORC contains human annotations to indicate which sentences in the source document cite the target document. These annotations can be used to assess the quality of selected sentences by OPT-Match.

Table 1: Statistics of datasets. '**#Word**' denotes the average number of words per document, and '**#Sent.**' denotes the average number of sentences per document.

Dataset	Train	Dev	Test	#Word	#Sent.
PAN	17,968	2,908	2,906	1,569.7	47.4
S2ORC	152,000	19,000	19,000	263.7	9.3
AAN	106,592	13,324	13,324	122.7	4.9
OC	240,000	30,000	30,000	190.4	7.0

Plagiarism detection is a task to detect whether a text span in the source document plagiarizes a text span in the target document. PAN (Potthast et al., 2013) is used for this task. PAN also contains human annotations to indicate which sentences in the source document plagiarizes the target document.

Baselines. For document matching, we take two representative kinds of methods including the hierarchical models and the variants of BERT as the underlying model and compare the performance of these models with and without OPT-Match.

The hierarchical models include GRU-HAN (Jiang et al., 2019) which uses stacked GRU and attention networks to encode documents following the order of words, sentences, paragraphs, and documents; BERT-HAN (Pappagari et al., 2019) which replaces sentence encoder of GRU-HAN with BERT; GRU-HAN-CDA and BERT-HAN-CDA (Zhou et al., 2020) which add cross-document attention to GRU-HAN and BERT-HAN respectively. Following (Zhou et al., 2020), the attention scores of sentences are used to indicate the model's selection of key sentences.

Variants of BERT includes BERT (Devlin et al., 2019), Transformer-XL (Dai et al., 2019), and Longformer (Beltagy et al., 2020). For BERT, we choose the *'bert-base-uncased'* and truncate the first 510 words of the document. For Transformer-XL and Longformer, we choose *"transfoxl-wt103'* and *'allenai/longformer-base-4096'* respectively.

To further verify the impact of partial alignment between documents, we consider two OT-based method: (Kusner et al., 2015) which uses OT as a similarity function between sentences and (Swanson et al., 2020) which conducts sparse OT between sentences by adding dummy nodes[2].

[1] https://github.com/XuhuiZhou/CDA

[2] We consider the relaxed 1-to-k assignment which is suit-

Table 2: Experimental results on S2ORC, PAN, AAN and OC test sets. x-OPT denotes OPT-Match equipped model x. $(soft)$ and $(hard)$ indicate the soft selection strategy and the hard selection strategy respectively. † indicates the statistically significant difference between the model equipped with OPT-Match and the corresponding underlying model (p-value < 0.05).

	AAN		OC		S2ORC		PAN	
Models	**Acc.**	**F1**	**Acc.**	**F1**	**Acc.**	**F1**	**Acc.**	**F1**
GRU-HAN	68.01	67.23	84.46	82.26	82.36	83.28	75.70	75.88
GRU-HAN-OPT $(soft)$	69.87†	69.30†	**85.76**†	**85.89**†	83.82†	84.29†	76.28†	76.63†
GRU-HAN-OPT $(hard)$	**71.02**†	**70.91**†	85.49†	85.74†	**85.14**†	**85.65**†	**76.76**†	**77.03**†
GRU-HAN-CDA	74.51	74.81	88.71	88.96	88.91	89.92	77.04	78.23
GRU-HAN-CDA-OPT $(soft)$	75.88†	76.06†	**89.94**†	**90.11**†	89.15	89.84	78.10†	78.39
GRU-HAN-CDA-OPT $(hard)$	**76.96**†	**76.65**†	88.62	88.78	**89.72**†	**89.96**†	**78.52**†	**78.84**†
BERT-HAN	67.32	64.97	85.96	86.33	90.67	90.76	87.57	87.36
BERT-HAN-OPT $(soft)$	68.72†	68.98†	87.30†	87.44†	90.75	90.87	87.74	87.25
BERT-HAN-OPT $(hard)$	**70.57**†	**71.22**†	**88.21**†	**88.49**†	**91.40**†	**91.55**†	**88.12**†	**88.01**†
BERT-HAN-CDA	71.57	69.08	87.81	87.89	91.92	92.07	86.23	86.19
BERT-HAN-CDA-OPT $(soft)$	73.85†	73.42†	89.07†	89.01†	92.52†	92.61†	87.13†	86.89†
BERT-HAN-CDA-OPT $(hard)$	**75.56**†	**75.62**†	**90.58**†	**90.54**†	**92.74**†	**92.81**†	**87.61**†	**87.14**†
BERT	88.05	88.09	94.52	94.45	95.64	95.64	59.58	69.71
BERT-OPT $(hard)$	**89.31**†	**89.35**†	**95.06**†	**94.97**†	**96.85**†	**96.82**†	**89.09**†	**88.61**†
Transformer-XL	83.18	82.92	91.19	91.26	92.50	92.39	58.25	69.07
Transformer-XL-OPT $(hard)$	**85.03**†	**84.99**†	**92.37**†	**92.43**†	**93.80**†	**93.69**†	**80.28**†	**80.11**†
Longformer	88.01	88.29	93.46	93.51	96.02	96.07	56.82	69.78
Longformer-OPT $(hard)$	**88.92**†	**89.07**†	**94.88**†	**94.87**†	**96.61**†	**96.56**†	**82.68**†	**82.21**†

Evaluation Metrics. We use Accuracy and F1 as the evaluation metrics since all the datasets have binary labels for document matching. Following (Zhou et al., 2020), we use MRR as the evaluation metric for sentence selection since the sentence selection is regarded as a ranking task.

Implementation Details. All hyper-parameters in OPT-Match[3] are tuned using grid search on the validation set. The tuning range of hyperparameters are as follows: the proportion of mass to be transported ϵ in Eq. (2) is tuned among $\{0.25, 0.50, 0.75\}$; coefficient λ in Eq. (4) is tuned between $[0.5, 1.0]$; Gumbel temperature τ in Eq. (6) is tuned between $[0.5, 1.0]$. For OPT-Match equipped hierarchical models, the settings of optimizer, learning rate, batch size and hidden dimension are consistent with corresponding underlying models. For OPT-Match equipped BERT's variants, the fine-tuning optimizer is Adam (Kingma and Ba, 2014) with $\beta_1 = 0.9$, $\beta_2 = 0.999$, the learning rate is tuned between $[0.00001, 0.00005]$; the batch size is tuned between $[4, 8]$.

3.2 Matching performances

Table 2 reports the evaluation results of different models. All the methods are trained ten times and the averaged results are reported. Since BERT and its variants do not explicitly generate sentence representations, we only applied the word-overlap ratio cost and the hard selection versions of OPT-Match on these models. For a fair comparison, we follow (Swanson et al., 2020) and keep the number of selected sentences as 5 in the hard selection.

We summarize our observations from the results as follows: 1) in general, the models equipped with OPT-Match consistently outperform their corresponding underlying models across four datasets. The results verify the effectiveness of OPT-Match in terms of enhancing matching accuracy. 2) BERT and its variants such as Transformer-XL and Longformer achieve much worse performances on PAN, comparing to their good performances on other datasets. PAN has a large average document length (> 1500 words) and a large average number of sentences per document (> 40 sentences). The noise in the extremely long documents makes BERT and its variations perform poorly. OPT-Match equipped models, however, are not affected by the document length. It is because OPT-Match successfully filters out the noise in the document by selecting key sentences for matching. The results indicate that OPT-Match is more effective especially when the document length is extremely long; 3) Comparing the performances of OPT-Match in the soft selection version to that of in the hard selection version, the hard selection versions achieve better performances in most cases. We analyze the results, and the reason is that the soft selection is a weighting strategy, aiming at sampling sentences according to the optimal transport plan of OPT-Match. However, the soft selection cannot completely filter out the noise in documents, which often has negative impacts on the matching.

able for long-form document matching.

[3]The source code of OPT-Match is available at https://github.com/ruc-wjyu/OPT-Match

Table 3: The impact of degree of the document alignment ϵ in OPT-Match. BHC-OPT denotes BERT-HAN-CDA-OPT model in the soft selection and BERT-OPT is in the hard selection version.

	S2ORC		PAN	
Models	**Acc.**	**F1**	**Acc.**	**F1**
BHC-OT(Kusner et al., 2015)	91.46	91.50	85.90	85.98
BHC-SOT (Swanson et al., 2020)	91.91	91.96	86.55	86.32
BHC-OPT($\epsilon = 0.75$)	92.33	92.46	86.95	86.59
BHC-OPT ($\epsilon = 0.50$)	92.65	92.74	87.08	86.75
BHC-OPT ($\epsilon = 0.25$)	**92.74**	**92.81**	**87.61**	**87.14**
BERT-OT (Kusner et al., 2015)	95.27	95.31	66.28	65.47
BERT-SOT (Swanson et al., 2020)	96.05	96.13	87.86	87.95
BERT-OPT ($\epsilon = 0.75$)	96.79	96.75	88.02	87.44
BERT-OPT ($\epsilon = 0.50$)	96.77	96.76	88.67	88.24
BERT-OPT ($\epsilon = 0.25$)	**96.85**	**96.82**	**89.09**	**88.61**

3.3 Impact of Partial Alignment

To test the effects of the partial alignment between documents, we compare the performances of OPT-Match with different proportion of mass ϵ (degree of alignment) to two representative OT-based (full alignment) methods (Kusner et al., 2015; Swanson et al., 2020). We suppose that with the lower ϵ, OPT-Match focuses more on strongly aligned sentence pairs, while filtering out more spurious alignment signals. Please note that (Kusner et al., 2015) conduct traditional OT between documents which denotes the full alignment and (Swanson et al., 2020) add dummy sentences and conducts full alignment between real sentences and dummy ones in order to achieve partial alignment within real sentences. As illustrated in Table 3, we find that models with partial alignment ($\epsilon < 1$) always achieve better performances than that of the full alignment on S2ORC and PAN datasets. The results verify that the alignment between documents is partial. Moreover, we find that OPT-Match tends to achieve better performance with smaller ϵ. The results indicate that only a small fraction of strongly aligned sentences contributed to document matching, filtering out the noise sentences is helpful. In addition, although (Swanson et al., 2020) aims at partial alignment, the way of adding dummy nodes may not suit long-form documents matching because the document length varies greatly leading to many dummy nodes, and the alignment between dummy nodes and real sentences may dominate in the full alignment and overwhelm the real alignments between sentences.

3.4 Impact of the number of selected sentences.

We further conduct experiments to investigate how the number of selected sentences in OPT-Match affects the matching. Specifically, we configure

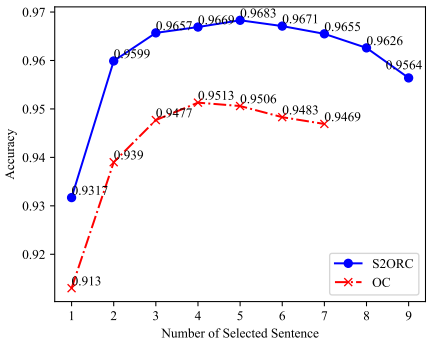

Figure 3: The impact of the number of selected sentence on OPT-Match. Results from BERT-OPT of the hard selection version on S2ORC and OC.

Table 4: Faithfulness evaluation. (all), $(selected)$, and $(all \setminus selected)$ respectively denotes using all sentences, using sentences selected by OPT-Match, and using the sentences not selected by OPT-Match as the inputs.

	S2ORC		PAN	
Models	**Acc.**	**F1**	**Acc.**	**F1**
BERT-HAN $(all \setminus selected)$	73.62	74.33	56.76	53.52
BERT-HAN (all)	90.67	90.76	77.04	78.23
BERT-HAN $(selected)$	**91.40**	**91.55**	**88.12**	**88.01**
BERT-HAN-CDA $(all \setminus selected)$	77.30	77.94	57.28	53.73
BERT-HAN-CDA (all)	91.92	92.07	86.23	86.19
BERT-HAN-CDA $(selected)$	**92.74**	**92.81**	**87.61**	**87.14**

BERT-OPT $(hard)$ to select a different number of sentences (from 1 to the average number of sentences in a document) and then conduct matching using these sentences. Figure 3 illustrates matching accuracy curves w.r.t. the number of selected sentences on the datasets of S2ORC and OC. We find that BERT-OPT shows a competitive performance when only 1 or 2 sentences are selected. The results show the effectiveness of OPT-Match in terms of accurately selecting sentences key to document matching. Additionally, the accuracy curves first steadily increase and reach the peak when 4 or 5 sentences are selected for matching. After that, the curves drop with more selected sentences. The phenomenon is intuitive and can be explained as follows: when the number of selected sentences is too small, the model needs more information from the selected sentences to infer the document-level matching. However, the number of selected sentences is greatly less than the number of all of the sentences in the document. Therefore, after some thresholds, the additional selected sentences become noisy, which causes the drop of the matching accuracy. The results verify our assumption that not all the sentences in the document contribute to the long-form document matching.

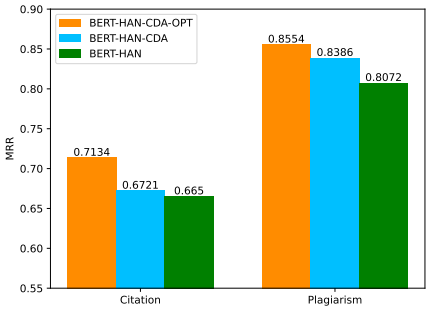

Figure 4: Plausibility comparison among BERT-HAN, BERT-HAN-CDA and BERT-HAN-CDA-OPT ($hard$) on S2ORC (Citation) and PAN (Plagiarism).

3.5 Selected Sentences as Rationales

To further assess the quality of the key sentence selected by OPT-Match, we regard the selected sentences as the rationales for the document-level matching prediction. Following (Strout et al., 2019; DeYoung et al., 2020), we adopt plausibility and faithfulness as the evaluation metrics. Plausibility measures how well the rationales provided by models align with human annotations, and faithfulness measures the degree to which the rationales influence the corresponding predictions.

Following the setting of (Zhou et al., 2020), we first compare the sentences selected by OPT-Match ($hard$) with human annotations in S2ORC and PAN. As shown in Figure 4, the sentences selected by OPT-Match are more consistent with human annotations compared with that of the sentence-level attention scores in BERT-HAN and BERT-CDA.

In terms of the faithfulness, we test the matching performance of BERT-HAN and BERT-HAN-CDA under three conditions: respectively using all sentences (denoted as (all)), using sentences selected by OPT-Match ($hard$) (denoted as ($selected$)), and using sentences except those selected by OPT-Match ($hard$) (denoted as $all \backslash selected$) as the model's input. From the results reported in Table 4, we find that the sentences selected by OPT-Match play a critical role in document matching, i.e., if the sentences selected by OPT-Match are removed from a model's input, the matching accuracy of the model drops dramatically. In addition, if all sentences are used as a model's input (note that the input still contains the sentences selected by OPT-Match), the predictive accuracy still drops to some extent because of the noise from other sentences. From the results, we conclude that OPT-Match is capable of accurately selecting the key sentences for document matching and filtering the noise.

Table 5: Training time (s/batch) on a single Nvidia Tesla V100 16GB. Batch size = 256 for both models.

Models	S2ORC	PAN
BERT-HAN-CDA	0.0992	0.2711
BERT-HAN-CDA-OPT ($soft$)	0.1004	0.2917
BERT-HAN-CDA-OPT ($hard$)	0.0934	0.1490

3.6 Time Complexity Analysis

Existing long-form document matching methods usually apply attention mechanism at the word level, which have a time complexity of $\mathcal{O}(N_{all}^2)$, where N_{all} denotes the number of tokens in the input document. For OPT-Match, the computing overhead mainly comes from the calculation of the optimal transport plan (line 6-9 in Algorithm 1). With the help of the entropic regularizer (Cuturi, 2013) and Bregman-Dykstra iterations (Bregman, 1967), the calculation of the optimal transport plan have a time complexity of $\mathcal{O}(N_s^2)$ (Benamou et al., 2015), where the N_s denotes the number of the sentences in the input document. Since usually the number of sentences is far less than the number of tokens in a document, i.e., $N_s \ll N_{all}$, we suppose the computational cost of OPT-Match is acceptable. In addition, if the word overlapping cost is applied, the hard selection version of OPT-Match can be used as a data pre-processing, therefore, OPT-Match does not increase the training time of the model. Also note that the input of the matching model equipped with OPT-Match is the selected sentences rather than all of the sentences. Therefore, the hard selection version of OPT-Match with word overlapping cost can effectively reduce the training time of the matching model. For the soft selection version of OPT-Match, although OPT-Match brings additional computational cost, considering OPT-Match is applied at the sentence level, the additional computation cost is not significant.

We further compare the average training time between BERT-HAN-CDA and BERT-HAN-CDA-OPT. From the results shown in Table 5, we can see that the soft selection version of OPT-Match brings an acceptable additional computation cost, while the hard selection version of OPT-Match effectively reduce the training time of the matching model.

4 Related Work

4.1 Long-form Document Matching

In long-form document matching, there are two representative methods in the literature — the hierarchical models and the variants of BERT.

The hierarchical models suppose that a document represents a hierarchical structure of words, sentences, paragraphs, and documents. Inspired by this idea, researchers exploit neural network to encode document in a hierarchical way. The representative method is (Jiang et al., 2019). This method first separately models input document pair as semantic vectors using RNN and Hierarchical Attention Network (Yang et al., 2016; Zhou et al., 2020). Then, the matching score is calculated by feeding the concatenation of the document vector to an MLP. Pappagari et al. (2019) improves (Jiang et al., 2019) by replacing the RNN-based encoder with the transformer-based encoder. Zhou et al. (2020) focuses on the interaction between documents and proposed the hierarchical cross-document attention to improve the document representation.

Although BERT has been dominated in the field of short-text matching, the quadratic time complexity of intrinsic attention mechanism makes BERT difficult to be applied in the long-form document matching. To tackle this issue. Dai et al. (2019) proposed Transformer-XL which consists of a segment-level recurrence mechanism and a positional encoding scheme. Transformer-XL enables learning dependency beyond a fixed length without disrupting temporal coherence. Beltagy et al. (2020) proposed dilated sliding windows attention which gradually increases the receptive field as the model goes deeper.

Recently, Pang et al. (2021) proposed to first filter out sentence-level noise from documents by applying PageRank on the sentence similarity graph and then plug PageRank into transformer layers to filter out word-level noise. Although these studies achieved improvement in document matching performances, they ignore the alignment between sentences in a document pair would be partial which inevitably introduced noises to the final matching.

4.2 OT in NLP

In recent years, OT have been widely studied in NLP. Kusner et al. (2015) formulated the distance between two sentences as an optimal transport problem and proposed Word Mover's Distance (WMD) which measures the dissimilarity between two text documents as the minimum amount of distance that words of one document need to transport to the words of another document. Yokoi et al. (2020) pointed out that the angle of semantic vectors is a good proxy for word similarity and proposed Word Rotator's Distance on top of WMD. Xu et al. (2018) proposed a Wasserstein method with a distillation mechanism, yielding joint learning of word embeddings and topics. Inspired from Order-Preserving OT (Su and Hua, 2017, 2019; Su et al., 2019), Liu et al. (2018) proposed to factorize sentence hierarchically based on Abstract Meaning Representation and obtain the reordered sentence representations. Then the semantic distance between a pair of text snippets can be solved by a penalized OT. Yu et al. (2020, 2022b) proposed to use OT to bridge the gap between heterogeous text pairs for sentence matching in asymmetrical domains. Chen et al. (2019); Zhang et al. (2020) respectively applied OT and OPT to sequence-to-sequence learning tasks such as text generation. (Li et al., 2019; Yu et al., 2022a) proposed to learn the similarity between texts using inverse optimal transport.

5 Conclusion

In this paper, we highlight the critical role of conducting partial alignment in long-form document matching. A novel key sentence selection component based on optimal partial transport is proposed, called OPT-Match. OPT-Match automatically selects key sentences for document matching, addressing the issue that not every sentence in one document can correspond to a sentence in another document. Moreover, OPT-Match can be easily incorporated into existing document matching models. Comprehensive experiments on four public datasets show that OPT-Match consistently outperformed its underlying document matching models. The empirical analysis also verifies that the sentences selected by OPT-Match are not only consistent with human-provided rationales but also contributed to document matching.

Acknowledgements

This work was funded by the National Key R&D Program of China (2019YFE0198200), the National Natural Science Foundation of China (61872338, 61832017, 62276248, 61906180, 61976206), Beijing Outstanding Young Scientist Program NO. BJJWZYJH012019100020098, Intelligent Social Governance Interdisciplinary Platform, Major Innovation & Planning Interdisciplinary Platform for the "Double-First Class" Initiative, Renmin University of China, and Public Policy and Decision-making Research Lab of Renmin University of China.

References

Iz Beltagy, Matthew E. Peters, and Arman Cohan. 2020. Longformer: The long-document transformer. *CoRR*, abs/2004.05150.

Jean-David Benamou, Guillaume Carlier, Marco Cuturi, Luca Nenna, and Gabriel Peyré. 2015. Iterative bregman projections for regularized transportation problems. *SIAM J. Sci. Comput.*, 37(2).

Chandra Bhagavatula, Sergey Feldman, Russell Power, and Waleed Ammar. 2018. Content-based citation recommendation. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 238–251, New Orleans, Louisiana. Association for Computational Linguistics.

Lev M Bregman. 1967. The relaxation method of finding the common point of convex sets and its application to the solution of problems in convex programming. *USSR computational mathematics and mathematical physics*, 7(3):200–217.

Liqun Chen, Shuyang Dai, Chenyang Tao, Haichao Zhang, Zhe Gan, Dinghan Shen, Yizhe Zhang, Guoyin Wang, Ruiyi Zhang, and Lawrence Carin. 2018. Adversarial text generation via feature-mover's distance. In *Advances in Neural Information Processing Systems 31: Annual Conference on Neural Information Processing Systems 2018, NeurIPS 2018, December 3-8, 2018, Montréal, Canada*, pages 4671–4682.

Liqun Chen, Yizhe Zhang, Ruiyi Zhang, Chenyang Tao, Zhe Gan, Haichao Zhang, Bai Li, Dinghan Shen, Changyou Chen, and Lawrence Carin. 2019. Improving sequence-to-sequence learning via optimal transport. In *7th International Conference on Learning Representations, ICLR 2019, New Orleans, LA, USA, May 6-9, 2019*. OpenReview.net.

Lenaic Chizat, Gabriel Peyré, Bernhard Schmitzer, and François-Xavier Vialard. 2018. Scaling algorithms for unbalanced optimal transport problems. *Mathematics of Computation*, 87(314):2563–2609.

Marco Cuturi. 2013. Sinkhorn distances: Lightspeed computation of optimal transport. In *Advances in Neural Information Processing Systems 26: 27th Annual Conference on Neural Information Processing Systems 2013. Proceedings of a meeting held December 5-8, 2013, Lake Tahoe, Nevada, United States*, pages 2292–2300.

Zihang Dai, Zhilin Yang, Yiming Yang, Jaime Carbonell, Quoc Le, and Ruslan Salakhutdinov. 2019. Transformer-XL: Attentive language models beyond a fixed-length context. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 2978–2988, Florence, Italy. Association for Computational Linguistics.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, Minneapolis, Minnesota. Association for Computational Linguistics.

Jay DeYoung, Sarthak Jain, Nazneen Fatema Rajani, Eric Lehman, Caiming Xiong, Richard Socher, and Byron C Wallace. 2020. Eraser: A benchmark to evaluate rationalized nlp models. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 4443–4458.

Richard L Dykstra. 1983. An algorithm for restricted least squares regression. *Journal of the American Statistical Association*, 78(384):837–842.

Alessio Figalli. 2010. The optimal partial transport problem. *Archive for rational mechanics and analysis*, 195(2):533–560.

Tomás Foltýnek, Norman Meuschke, and Bela Gipp. 2020. Academic plagiarism detection: A systematic literature review. *ACM Comput. Surv.*, 52(6):112:1–112:42.

Eric Jang, Shixiang Gu, and Ben Poole. 2016. Categorical reparameterization with gumbel-softmax. *arXiv preprint arXiv:1611.01144*.

Jyun-Yu Jiang, Mingyang Zhang, Cheng Li, Michael Bendersky, Nadav Golbandi, and Marc Najork. 2019. Semantic text matching for long-form documents. In *The World Wide Web Conference, WWW 2019, San Francisco, CA, USA, May 13-17, 2019*, pages 795–806. ACM.

Diederik P Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*.

Matt J. Kusner, Yu Sun, Nicholas I. Kolkin, and Kilian Q. Weinberger. 2015. From word embeddings to document distances. In *Proceedings of the 32nd International Conference on Machine Learning, ICML 2015, Lille, France, 6-11 July 2015*, volume 37 of *JMLR Workshop and Conference Proceedings*, pages 957–966. JMLR.org.

Ruilin Li, Xiaojing Ye, Haomin Zhou, and Hongyuan Zha. 2019. Learning to match via inverse optimal transport. *Journal of machine learning research*, 20.

Bang Liu, Ting Zhang, Fred X. Han, Di Niu, Kunfeng Lai, and Yu Xu. 2018. Matching natural language sentences with hierarchical sentence factorization. In *Proceedings of the 2018 World Wide Web Conference on World Wide Web, WWW 2018, Lyon, France, April 23-27, 2018*, pages 1237–1246. ACM.

Kyle Lo, Lucy Lu Wang, Mark Neumann, Rodney Kinney, and Daniel Weld. 2020. S2ORC: The semantic scholar open research corpus. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 4969–4983, Online. Association for Computational Linguistics.

Rada Mihalcea and Paul Tarau. 2004. Textrank: Bringing order into text. In *Proceedings of the 2004 conference on empirical methods in natural language processing*, pages 404–411.

Liang Pang, Yanyan Lan, and Xueqi Cheng. 2021. Match-ignition: Plugging pagerank into transformer for long-form text matching.

Raghavendra Pappagari, Piotr Zelasko, Jesús Villalba, Yishay Carmiel, and Najim Dehak. 2019. Hierarchical transformers for long document classification. In *2019 IEEE Automatic Speech Recognition and Understanding Workshop (ASRU)*, pages 838–844. IEEE.

Martin Potthast, Matthias Hagen, Tim Gollub, Martin Tippmann, Johannes Kiesel, Paolo Rosso, Efstathios Stamatatos, and Benno Stein. 2013. Overview of the 5th international competition on plagiarism detection. In *Working Notes for CLEF 2013 Conference , Valencia, Spain, September 23-26, 2013*, volume 1179 of *CEUR Workshop Proceedings*. CEUR-WS.org.

Dragomir R. Radev, Pradeep Muthukrishnan, Vahed Qazvinian, and Amjad Abu-Jbara. 2013. The ACL anthology network corpus. *Lang. Resour. Evaluation*, 47(4):919–944.

Julia Strout, Ye Zhang, and Raymond Mooney. 2019. Do human rationales improve machine explanations? In *Proceedings of the 2019 ACL Workshop BlackboxNLP: Analyzing and Interpreting Neural Networks for NLP*, pages 56–62, Florence, Italy. Association for Computational Linguistics.

Bing Su and Gang Hua. 2017. Order-preserving wasserstein distance for sequence matching. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.

Bing Su and Gang Hua. 2019. Order-preserving optimal transport for distances between sequences. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 41(12):2961–2974.

Bing Su, Jiahuan Zhou, and Ying Wu. 2019. Order-preserving wasserstein discriminant analysis. In *2019 IEEE/CVF International Conference on Computer Vision, ICCV 2019, Seoul, Korea (South), October 27 - November 2, 2019*, pages 9884–9893. IEEE.

Kyle Swanson, Lili Yu, and Tao Lei. 2020. Rationalizing text matching: Learning sparse alignments via optimal transport. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 5609–5626, Online. Association for Computational Linguistics.

Yujia Xie, Xiangfeng Wang, Ruijia Wang, and Hongyuan Zha. 2018. A fast proximal point method for wasserstein distance. *CoRR*, abs/1802.04307.

Hongteng Xu, Wenlin Wang, Wei Liu, and Lawrence Carin. 2018. Distilled wasserstein learning for word embedding and topic modeling. In *Advances in Neural Information Processing Systems*, volume 31. Curran Associates, Inc.

Zichao Yang, Diyi Yang, Chris Dyer, Xiaodong He, Alex Smola, and Eduard Hovy. 2016. Hierarchical attention networks for document classification. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 1480–1489, San Diego, California. Association for Computational Linguistics.

Sho Yokoi, Ryo Takahashi, Reina Akama, Jun Suzuki, and Kentaro Inui. 2020. Word rotator's distance. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 2944–2960, Online. Association for Computational Linguistics.

Weijie Yu, Zhongxiang Sun, Jun Xu, Zhenhua Dong, Xu Chen, Hongteng Xu, and Ji-Rong Wen. 2022a. Explainable legal case matching via inverse optimal transport-based rationale extraction. In *Proceedings of the 45th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 657–668.

Weijie Yu, Chen Xu, Jun Xu, Liang Pang, Xiaopeng Gao, Xiaozhao Wang, and Ji-Rong Wen. 2020. Wasserstein distance regularized sequence representation for text matching in asymmetrical domains. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 2985–2994, Online. Association for Computational Linguistics.

Weijie Yu, Chen Xu, Jun Xu, Liang Pang, and Ji-Rong Wen. 2022b. Distribution distance regularized sequence representation for text matching in asymmetrical domains. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 30:721–733.

Ruiyi Zhang, Changyou Chen, Xinyuan Zhang, Ke Bai, and Lawrence Carin. 2020. Semantic matching for sequence-to-sequence learning. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 212–222, Online. Association for Computational Linguistics.

Xuhui Zhou, Nikolaos Pappas, and Noah A. Smith. 2020. Multilevel text alignment with cross-document attention. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 5012–5025, Online. Association for Computational Linguistics.

LightNER: A Lightweight Tuning Paradigm for Low-resource NER via Pluggable Prompting

Xiang Chen[1,2]*, **Lei Li**[1,2]*, **Shumin Deng**[1,2], **Chuanqi Tan**[3], **Changliang Xu**[4]
Fei Huang[3], **Luo Si**[3], **Huajun Chen**[1,2], **Ningyu Zhang**[1,2,†]

[1]Zhejiang University & AZFT Joint Lab for Knowledge Engine, China
[2]Hangzhou Innovation Center, Zhejiang University, [3]Alibaba Group
[4]State Key Laboratory of Media Convergence Production Technology and Systems
{xiang_chen, leili21, 231sm, huajunsir, zhangningyu}@zju.edu.cn,
{chuanqi.tcq, f.huang, luo.si}@alibaba-inc.com, xu@shuwen.com

Abstract

Most NER methods rely on extensive labeled data for model training, which struggles in the low-resource scenarios with limited training data. Existing dominant approaches usually suffer from the challenge that the target domain has different label sets compared with a resource-rich source domain, which can be concluded as class transfer and domain transfer. In this paper, we propose a lightweight tuning paradigm for low-resource NER via pluggable prompting (LightNER). Specifically, we construct the unified learnable verbalizer of entity categories to generate the entity span sequence and entity categories without any label-specific classifiers, thus addressing the class transfer issue. We further propose a pluggable guidance module by incorporating learnable parameters into the self-attention layer as guidance, which can re-modulate the attention and adapt pre-trained weights. Note that we only tune those inserted module with the whole parameter of the pre-trained language model fixed, thus, making our approach lightweight and flexible for low-resource scenarios and can better transfer knowledge across domains. Experimental results show that LightNER can obtain comparable performance in the standard supervised setting and outperform strong baselines in low-resource settings[1].

1 Introduction

Named Entity Recognition (NER) has been a fundamental task of research within the Natural Language Processing (NLP) community. Mostly, the NER task is formulated as a sequence classification task, aiming to assign the labels to each entity in the input sequence. And those entity labels are all based on pre-defined categories, such as location, organization, person. The current

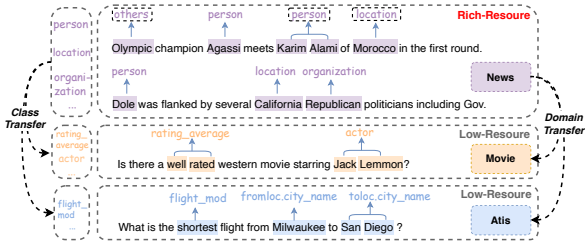

Figure 1: Examples of NER involved in **Class Transfer** and **Domain Transfer** in low-resource setting.

mature methodologies for handling NER is using Pre-trained language models (PLMs) (Devlin et al., 2019) equipped with several NER paradigms to perform extensive training process on large corpus, such as label-specific classifier paradigm (LC) (Strubell et al., 2017; Cui and Zhang, 2019), machine reading comprehension paradigm (MRC) (Yu et al., 2020) and unified generative paradigm (BartNER (Yan et al., 2021)). Unfortunately the resulting models are highly associated with seen categories, which often explicitly memorizing entity values (Agarwal et al., 2021), partially because the output layers require a consistent label set between training and testing. Note that these models require to build a new model from scratch to adapt to a target domain with new entity categories, thus, achieving unsatisfactory performance when the target labeled data is limited.

Unfortunately, this problem is prevalent in real-world application scenarios and draws attention to a challenging but practical research problem: low-resource NER, where the model is built to quickly identify new entities in a completely unseen target domain with only a few supporting samples in the new domain. Overall, low-resource NER (Wiseman and Stratos, 2019; Yang and Katiyar, 2020; Ziyadi et al., 2020) mainly faces two issues as shown in Figure 1: (1) **Class Transfer**. Entity categories can be different across rich-resource settings and low-resource settings. For example, source

* Equal Contribution.
† Corresponding Author.
[1]Code is in https://github.com/zjunlp/DeepKE/tree/main/example/ner/few-shot.

Proceedings of the 29th International Conference on Computational Linguistics, pages 2374–2387
October 12–17, 2022.

news domain contains the entity categories with "person" ,"location" , etc, while target movie domain adds new categories with "rating_average" and "actor". In this situation, the current mainstream method such as LC, MRC and BARTNER have to refactor a new model and train it from scratch, which is expensive, and unrealistic for real-world settings. (2) **Domain Transfer**. Compared with rich-resource settings, the low-resource setting may have a different textual domain. Intuitively, the sentence in news domain and atis domain contain the different grammar style and allegorical theme, which is not trivial to transfer the model fully trained in the source domain to target domain with few examples.

To address the issue of class transfer, we first reformulate the NER task from sequence labeling to a generative framework with a unified learnable verbalizer to realize **class transfer**. Considering different categories involve varying numbers of words as their descriptions, vanilla mainstream method that assign single classifier for entity may lose important label semantic information. Thus, we propose to construct a unified learnable verbalizer based on generative framework. Different from BartNER (Yan et al., 2021)) that has extra MLP layer in Encoder and classifier in Decoder, our method only contain the original architecture of pre-trained generative model by constructing constructing a unified learnable verbalizer for entity. Therefore, our approach can directly leverage any new or complicated entity types without modifying the network structure.

Recently, prompt-tuning (Schick and Schütze, 2021; Gao et al., 2021; Li and Liang, 2021; Liu et al., 2021c) has emerged to become surprisingly effective for the model adaptation of PLMs, especially in the low-resource setting. However, the prompt-tuning relies on reformulating the paradigm of downstream tasks into new tasks similar to MLM pre-training, which is not efficient for sequence labeling tasks such as NER. Inspired by the success of prompt-learning (Lester et al., 2021) in domain adaptation, we propose a lightweight tuning paradigm with pluggable guidance for NER (LightNER) to tackle the downsides of **domain transfer**. Specifically, we propose to incorporate learnable parameters into the self-attention layer in LMs and regard the parameters as knowledgeable guidance. In particular, we explore lightweight tuning with the pluggable guidance module to urge it to learn domain transfer ability and condition it at inference time.

In light of the limits of the existing techniques, we are interested in building a lightweight tuning framework fot low-resource NER with pluggable prompting. Notably, the modules in LightNER are extremely coupled and indispensable to each other. It is precisely because we design a generative model equipped with a decoupling space to solve the issue class transfer that the pluggable guidance module can realize domain knowledge transfer with lightweight tuning. In a nutshell, LightNER consists of the following contributions:

- We convert sequence labeling to the generative framework and construct decoupling space without any label-specific layers to solve the issue of class transfer. Therefore, the proposed method does not require to build a new model from scratch to adapt to a target domain with new entity categories.
- We propose to incorporate learnable parameters into the self-attention layers as pluggable guidance, which can be seamlessly plugged into the pre-trained generative models to conduct lightweight tuning with cross-domain and cross-task knowledge transfer ability. Therefore, LightNER doesn't need to maintain an LM for each target domain NER tasks and pay for expensive training services.
- We conduct extensive experiments on several benchmark datasets, and by tuning only little parameters, LightNER can achieve comparable results in standard supervised settings and yield promising performance in low-resource settings. Our results also suggest that LightNER has the potential towards cross domain zero-shot generation with pluggable guidance.

2 Related Work

2.1 Named Entity Recognition

The current dominant methods (Ma and Hovy, 2016; Liu et al., 2019; Zhang et al., 2020b; Liu et al., 2021a,b) treat NER as a sequence tagging problem with label-specific classifiers or CRF. Nevertheless, these works still need to modify the model architecture when facing new entity classes; the inability to solve the challenge of class transfer limits its efficiency and transferability, which is not

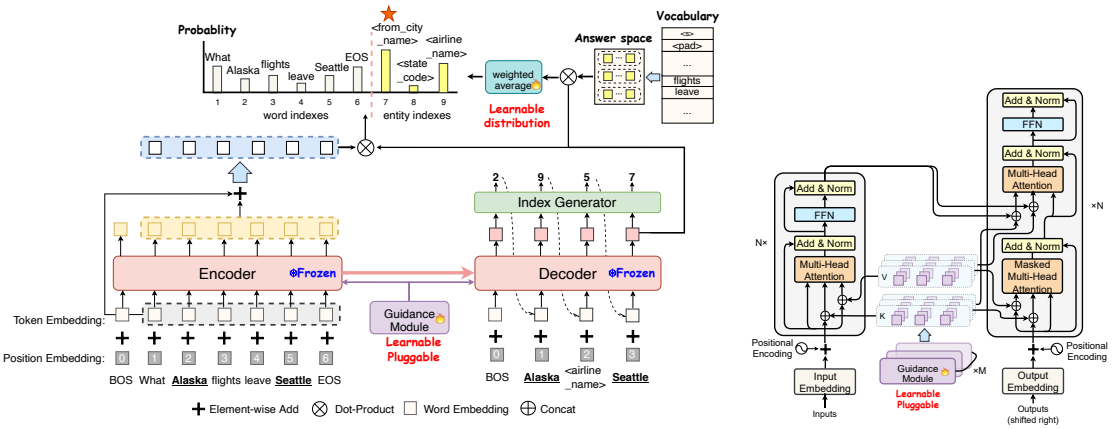

(a) Overview of LightNER.

(b) Attention with guidance.

Figure 2: Overview of our LightNER framework.

suitable for low-resource scenarios. Meanwhile, one crucial research line of low-resource NER is prototype-based methods, which involve meta-learning and have recently become popular few-shot learning approaches in the NER area. Most of the approaches (Fritzler et al., 2019; Wiseman and Stratos, 2019; Yang and Katiyar, 2020; Ziyadi et al., 2020; Henderson and Vulic, 2021; Hou et al., 2020; Lin et al., 2019; Ding et al., 2021) utilize the nearest-neighbor criterion to assign the entity type, which depends on similar patterns of entity between the source domain and the target domain without fully exploiting the potential of PLMs, behaving unsatisfactorily for cross-domain instances.

Recently, Cui et al. (2021) propose template-based BART for few-shot NER, which enumerates all $\hat{\mathbf{n}}$-gram possible spans in the sentence and fills them in the hand-crafted templates, classifying each candidate span based on the corresponding template scores. Different from their approach, our framework does not need template engineering and is more friendly with computation complexity.

2.2 Prompt Learning for PLMs

Since the emergence of GPT-3 (Brown et al., 2020), prompt-tuning has received considerable attention. A series of research work (Schick and Schütze, 2021; Schick et al., 2020; Shin et al., 2020; Han et al., 2021; Ben-David et al., 2022; Poth et al., 2021; Ben-David et al., 2022) have emerged, which implies that prompt-tuning can effectively stimulate knowledge from PLMs compared with standard fine-tuning, thus, inducing better performances on few-shot and cross-domain tasks. However, prompt learning mainly focuses on reformulating the downstream tasks' paradigm into completing a cloze task to bridge the gap between pre-training and fine-tuning, lacking an efficient method for NER and other sequence labeling tasks. Different from recent work of prompt tuning for NER (Zhou et al., 2021; Ma et al., 2022), we mainly focus on the issues of domain transfer and class transfer for low-resource NER.

2.3 Lightweight Learning for PLMs

Lightweight fine-tuning is performed to leverage the ability of PLMs with small trainable parameters. On the one hand, several studies consider removing or masking redundant parameters from PLMs (Frankle and Carbin, 2019; Sanh et al., 2020). On the other hand, some researchers (Guo et al., 2021; Zhang et al., 2020a) argue that extra trainable modules should be inserted into PLMs. As a typical approach, adapter-tuning (Houlsby et al., 2019) inserts task-specific layers (adapters) between each layer of PLMs; prefix-tuning (Li and Liang, 2021) prepends a sequence of continuous task-specific vectors to the inputs. However, adapter-tuning adds additional layers into the activation module of LMs, while this modification of the architecture is inconvenient to redeploy the model when switching to a new domain with unseen entity types. Meanwhile, the different label sets among domains make it impossible to transfer prefix-tuning to sequence labeling such as NER.

Apart from Adapter (Houlsby et al., 2019) and prefix-tuning (Li and Liang, 2021) aiming to conduct efficient finetuning, our approach focus on achieving efficient knowledge transfer through a pluggable paradigm.

3 Preliminaries

3.1 Low-resource NER

Given a rich-resource NER dataset $\mathbb{H} = \{(\mathbf{X}_1^H, \mathbf{Y}_1^H), ..., (\mathbf{X}_R^H, \mathbf{Y}_R^H)\}$, where the input is a text sequence of length n, $\mathbf{X}^H = \{x_1^H, \ldots, x_n^H\}$, we use $\mathbf{Y}^H = \{y_1^H, \ldots, y_n^H\}$ to denote corresponding labeling sequence of length n, and adopt $\mathcal{C}^H$ to represent the label set of the rich-resource dataset ($\forall y_i^H, y_j^H \in \mathcal{C}^H$). Traditional NER methods are trained in the standard supervised learning settings, which usually require many pairwise examples, *i.e.,* R is large. However, only a few labeled examples are available for each entity category in real-world applications due to the intensive annotation cost. This issue yields a challenging task of *low-resource* NER, in which given a low-resource NER dataset, $\mathbb{L} = \{(\mathbf{X}_1^L, \mathbf{Y}_1^L), ..., (\mathbf{X}_r^L, \mathbf{Y}_r^L)\}$, the number of labeled data in low-resource NER dataset is quite limited (i.e., $r \ll R$) compared with the rich-resource NER dataset. Regarding the issues of low resource and cross domain, the target entity categories $\mathcal{C}^L$ ($\forall l_i^L, l_j^L \in \mathcal{C}^L$) may be different from $\mathcal{C}^H$, which is challenging for model optimization.

3.2 Label-specific Classifier for NER

Traditional sequence labeling methods usually assign a label-specific classifier over the input sequence, which identifies named entities using BIO tags. A label-specific classifier with parameter $\boldsymbol{\theta} = \{\mathbf{W}_\mathcal{C}, \mathbf{b}_\mathcal{C}\}$ followed by a softmax layer is used to project the representation $\mathbf{h}$ into the label space. Formally, given $x_{1:n}$, the label-specific classifier method calculates:

$$\begin{aligned} \mathbf{h}_{1:n} &= \textsc{Encoder}(x_{1:n}), \\ q(\boldsymbol{y}|\boldsymbol{x}) &= \textsc{softmax}(\mathbf{h}_i \mathbf{W}_\mathcal{C} + \mathbf{b}_\mathcal{C}) \ (i \in [1, ..., n]), \end{aligned} \tag{1}$$

where $\mathbf{W}_\mathcal{C} \in \mathbb{R}^{d \times m}$, $\mathbf{b}_\mathcal{C} \in \mathbb{R}^m$ are trainable parameters and m is the numbers of entity categories. We adopt BERT (Devlin et al., 2019) and BART (Lewis et al., 2020) as our ENCODER to encoder the representation of text sequence, together with label-specific classifier layer, denoted as **LC-BERT** and **LC-BART** respectively.

4 Methodology

4.1 Task Formulation

Low-resource NER usually involves the class transfer, where new entity categories exist in target domains; however, the traditional sequence labeling method needs a label-specific output layer based on PLMs, hurting its generalization. Therefore, we reformulate the NER as a generative framework to maintain the consistency of architecture and enable the model to handle different entity types. For a given sentence X, we tokenize it into a sequence of tokens $X = \{x_1, x_2, ...x_n\}$. The NER task aims to provide the start and end index of an entity span, along with the entity type, represented by e, t in our framework, respectively. e is the index of tokens and $t \in$ {"*person*", "*organization*", ..., } is the set of entity types. Superscript ^{start} and ^{end} denote the start and end index of the corresponding entity token in the sequence. For the generative framework, the target sequence Y consists of multiple base prediction $p_i = \{e_i^{start}, e_i^{end}, t_i\}$ and $Y = \{p_1, p_2, ..., p_l\}$, where l denotes num of entities in X. We take a sequence of tokens X as input and hope to generate the target sequence Y as defined above. The input and output sequence starts and ends with special tokens "<s>" and "</s>". They should also be generated in Y, but we ignore them in equations for simplicity. Given a sequence of tokens X, the conditional probability is calculated as:

$$P(Y|X) = \prod_{t=1}^{3l} p(y_t|X, y_0, y_1, ..., y_{t-1}). \tag{2}$$

4.2 Generative Framework

To conduct **class transfer**, we adopt the seq2seq architecture with the pointer network to model the conditional probability $P(Y|X)$, where the conduction of pointer network is inspired by the See et al. (2017); Yan et al. (2021). Our generative module is shown in Figure 2, consisting of two components:

4.2.1 Encoder

The encoder is to encode X into the hidden representation space as a vector H_{en}.

$$H_{en} = \text{Encoder}(X) \tag{3}$$

where $H_{en} \in \mathcal{R}^{n \times d}$ and d is the hidden state dimension.

4.2.2 Decoder

The decoder part takes the encoder outputs H_{en} and previous decoder outputs $y_1, y_2, ..., y_{t-1}$ as inputs to decode y_t. $y_i{}_{i=1}^{t-1}$ indicates the token indexes; an index-to-token converter is applied for conversion.

$$\tilde{y}_i = \begin{cases} X_{y_i}, & \text{if } y_i \text{ is a pointer index} \\ C_{y_i - n}, & \text{if } y_i \text{ is a class index} \end{cases} \tag{4}$$

where $C = [c_1, c_2, ...c_m]$ is the set of entity categories (such as "Person", "Organazation", etc.), which are answer words corresponding to the entity category[2]. After this, we then get the last hidden state for y_t with the converted previous decoder outputs $[\tilde{y}_{i=1}^{t-1}]$.

$$h_t = \text{Decoder}(H_{en}; [\tilde{y}_{i=1}^{t-1}]) \tag{5}$$

where $h_t \in \mathcal{R}^d$; moreover, the probability distribution p_t of token y_t can be computed as follows:

$$\begin{aligned} E_{seq} &= \text{WordEmbed}(X), \\ \tilde{H}_{en} &= \alpha \cdot H_{en} + (1-\alpha) \cdot E_{seq}, \\ p_{seq} &= \tilde{H}_{en} \otimes h_t, \\ p_t &= \text{Softmax}([p_{seq}; p_{tag}]), \end{aligned} \tag{6}$$

where $E_{seq}, \tilde{H}_{en} \in \mathcal{R}^{n \times d}$; $\alpha \in \mathcal{R}$ is a hyper-parameter; p_{seq} and p_{tag} refer to the predicted logits on index of entity span and entity categories respectively; $p_t \in \mathcal{R}^{(n+m)}$ is the predicted probability distribution of y_t on all candidate indexes; $[\cdot\,;\cdot]$ denotes concatenation in the first dimension. In particular, the details of p_{tag} are in the following subsection.

4.3 Unified Learnable Verbalizer

As for the prediction of entity categories in NER, it is challenging to manually find appropriate tokens in the vocabulary to distinguish different entity types. Besides, some entity type may be complicated or very long in the specific target domain, such as $return\_date.month\_name$ in ATIS (Hakkani-Tür et al., 2016) and $restaurant\_name$ in MIT Restaurant (Liu et al., 2013).

To address the above issues in class transfer, we construct a unified learnable verbalizer containing multiple label words related to each entity class and leverage the weighted average approach for the utilization of the decoupling space $\mathcal{V}$. Concretely, we define a mapping $\mathcal{M}$ from the label space of entity categories $\mathcal{C}$ to the unified learnable verbalizer $\mathcal{V}$, i.e., $\mathcal{M}: \mathcal{C} \mapsto \mathcal{V}$. We utilize $\mathcal{V}_c$ to represent the subset of $\mathcal{V}$ that is mapped by a specific entity type c, $\mathcal{V} = \cup_{c \in \mathcal{C}} \mathcal{V}_c$. Take the above c = "return_date.month_name" as example, we set $\mathcal{V}_c$ = {"return","date","month","name"} according to decomposition of c. Since the direct average function may be biased, we adopt learnable weights β to average the logits of label words in answer space as the prediction logit:

$$\begin{aligned} E_{tag} &= \text{WordEmbed}(\mathcal{M}(\mathcal{C})), \\ p_{tag} &= Concat[\sum_{v \in \mathcal{V}_c} {\beta_v}^c * E_{tag}^c \otimes h_t] \end{aligned} \tag{7}$$

where ${\beta_v}^c$ denotes the weight of entity type c; $\sum_{v \in \mathcal{V}_c} {\beta_v}^c = 1$; $p_{tag} \in \mathcal{R}^m$. Through the construction of the unified learnable verbalizer, LightNER can perceive semantic knowledge in entity categories without modifying the PLM.

4.4 Pluggable Guidance Module

4.4.1 Parameterized Setting

Specifically, LightNER adds two sets of trainable embedding matrices $\{\phi^1, \phi^2, .., \phi^N\}$ for the encoder and decoder, respectively, and sets the number of transformer layers as N, where $\phi_\theta \in \mathbb{R}^{2 \times |P| \times d}$ (parameterized by θ), $|P|$ is the length of the prompt, d represents the $dim(h_t)$, and 2 indicates that ϕ is designed for the key and value. In our method, the LM parameters are fixed, and the prompt parameters θ and the learnable distribution of β are the only trainable parameters.

4.4.2 Pluggable Guidance Layer

LightNER inherits the architecture of the transformer (Vaswani et al., 2017), which is a stack of identical building blocks wrapped up with a feed-forward network, residual connection, and layer normalization. As a specific component, we introduce the pluggable guidance layer over the original query/key/value layer to achieve flexible and effective knowledge transfer. Given an input token sequence $X = \{x_1, x_2, ..., x_n\}$, following the above formulation, we can incorporate the representation of the guidance module into x with the calculation of self-attention. In each layer l, the input sequence representation $\boldsymbol{X}^l \in \mathbb{R}^{nd}$ is first projected into the query/key/value vector:

$$\boldsymbol{Q}^l = \boldsymbol{X}^l \boldsymbol{W}^Q, \boldsymbol{K}^l = \boldsymbol{X}^l \boldsymbol{W}^K, \boldsymbol{V}^l = \boldsymbol{X}^l \boldsymbol{W}^V, \tag{8}$$

where $\boldsymbol{W}_l^Q$, $\boldsymbol{W}_l^K$, $\boldsymbol{W}_l^V \in \mathbb{R}^{d \times d}$. Then, we can redefine the attention operation as:

$$Attention^l = softmax(\frac{\boldsymbol{Q}^l[\boldsymbol{\phi}_k^l; \boldsymbol{K}^l]^T}{\sqrt{d}})[\boldsymbol{\phi}_v^l; \boldsymbol{V}^l]. \tag{9}$$

Based on these representations of inputs and pluggable guidance module, we aggregate them and compute the attention scores to guide the final self-attention flow. Consequently, the guidance module can re-modulate the distribution of attention.

[2]The index of entity categories always starts after the pointer indexes of the given sequence, at $n+1$.

Traditional Models	P	R	F
Yang et al. (2018)	-	-	90.77
Ma and Hovy (2016)	-	-	91.21
Yamada et al. (2020)	-	-	**94.30**
Gui et al. (2020)	-	-	92.02
Li et al. (2020) †	92.47	93.27	92.87
Yu et al. (2020) ‡	**92.85**	92.15	92.50
LC-BERT	91.93	91.54	91.73
LC-BART	89.60	91.63	90.60
Few-shot Friendly Models	**P**	**R**	**F**
Wiseman and Stratos (2019)	-	-	89.94
Template (Cui et al., 2021)	90.51	93.34	91.90
LightNER	92.39	**93.48**	92.93

Table 1: Model performance on the CoNLL-2003 dataset . "†" indicates that we rerun their code with BERT-LARGE (Devlin et al., 2019). "‡" indicates our reproduction with only the sentence-level context. Although LUKE (Yamada et al., 2020) is pre-trained with a large entity-annotated corpus (Wikipedia), LightNER is highly competitive in rich resource settings even though it is designed for low-resource NER.

5 Experiments

We conduct extensive experiments in standard and low-resource settings. We use CoNLL-2003 (Sang and Meulder, 2003) as the rich-resource domain. Following the settings in Ziyadi et al. (2020) and Huang et al. (2020), we use the Massachusetts Institute of Technology (MIT) Restaurant Review (Liu et al., 2013), MIT Movie Review (Liu et al., 2013), and Airline Travel Information Systems (ATIS) (Hakkani-Tür et al., 2016) datasets as the cross-domain low-resource datasets[3]. Our experiments are evaluated in an exact match scenario, data analysis, and implementation details are presented in the Section Appendix A and B. We also provide the supplementary experimental result for the in-domain low-resource setting as shown in Section Appendix C.1.

5.1 Standard Supervised NER Setting

We adopt the CoNLL-2003 dataset to conduct experiments in the standard supervised settings. A comparison of the results of LightNER and the SOTA methods are listed in Table 1. Mainly, LC-BERT and LC-BART provide a strong baseline. We identify that even though LightNER is designed for the low-resource NER, it is highly competitive with the best-reported score in the rich-resource setting as well, indicating the effectiveness of our decoding strategy and guidance module.

[3]We do not conduct experiemnts on Few-NERD (Ding et al., 2021) since our setting follows (Ziyadi et al., 2020) which is different from the N-way K-shot settting.

5.2 Cross-Domain Low-resource NER Setting

In this section, we evaluate the model performance in the scenarios in which the target entity categories and textual style are specifically different from the source domain, and only limited labeled data are available for training. Precisely, we follow the setting in Cui et al. (2021) to sample a specific number of samples per entity category randomly as the training data in the target domain to simulate the cross-domain low-resource data scenarios. Table 2 lists the results of training models on the CoNLL-2003 dataset as a generic domain and its evaluations on other target domains. The results of LightNER are based on running the experiments five times on random samples and calculating the average of their scores.

Competitive Baselines We consider seven competitive approaches in our experiments. The *prototype-based methods*[4] primarily include the following: (i) *Neigh.Tag.* (Wiseman and Stratos, 2019); (ii) *Example-based NER* (Ziyadi et al., 2020); (iii) *Multi-prototype + NSP* (referred to as *MP-NSP*) is a SOTA prototype-based method reported in (Huang et al., 2020), utilizing noisy supervised pretraining. The *label-specific classifier* mainly include the following: (iv) *LC-BERT* and (v) *LC-BART* is the adoption of the label-specific classifiers on top of corresponding PLMs. Besides, (vi) *Template-based BART* (Cui et al., 2021) recently propose a template-based method for few-shot NER and (vii) *BERT-MRC* (Yu et al., 2020) propose to formulate NER as a machine reading comprehension (MRC) task, which is a strong SOTA model for NER. A summary comparison with baselines is shown in Section 4 of Appendix.

Train from Scratch on Target Domain We first consider direct training on the target domain from scratch without any available source domain data. However, prototype-based methods cannot be used in this setting. When compared to the LC-BART, LC-BERT, template-based BART and BERT-MRC, the results of our approach is consistently more persistent, indicating LightNER can better exploit few-shot data. Particularly, LightNER achieve an F1-score of 57.8% in 20-shot setting on MIT Movie, which is higher than the results of LC-BERT and template-based BART in 50-shot setting.

[4]Note that even if the prototype-based methods is training-free in the target domain, they are by no means equivalent to zero-shot setting, since prototype-based methods require labeled data in target domain as supporting examples.

Source	Methods	*MIT Movie*						*MIT Restaurant*						*ATIS*		
		10	20	50	100	200	500	10	20	50	100	200	500	10	20	50
None	LC-BERT	25.2	42.2	49.6	50.7	59.3	74.4	21.8	39.4	52.7	53.5	57.4	61.3	44.1	76.7	90.7
	LC-BART	10.2	27.5	44.2	47.5	54.2	64.1	6.3	8.5	51.3	52.2	56.3	60.2	42.0	72.7	87.5
	Template	37.3	48.5	52.2	56.3	62.0	74.9	46.0	57.1	58.7	60.1	62.8	65.0	71.7	79.4	92.6
	BERT-MRC†	18.7	48.3	55.5	62.5	80.2	82.1	12.3	37.1	53.5	63.9	65.5	70.4	35.3	63.2	90.2
	LightNER	**41.7**	**57.8**	**73.1**	**78.0**	**80.6**	**84.8**	**48.5**	**58.0**	**62.0**	**70.8**	**75.5**	**80.2**	**76.3**	**85.3**	**92.8**
CoNLL03	Neigh.Tag.	0.9	1.4	1.7	2.4	3.0	4.8	4.1	3.6	4.0	4.6	5.5	8.1	2.4	3.4	5.1
	Example.	29.2	29.6	30.4	30.2	30.0	29.6	25.2	26.1	26.8	26.2	25.7	25.1	22.9	16.5	22.2
	MP-NSP	36.4	36.8	38.0	38.2	35.4	38.3	46.1	48.2	49.6	49.6	50.0	50.1	71.2	74.8	76.0
	LC-BERT	28.3	45.2	50.0	52.4	60.7	76.8	27.2	40.9	56.3	57.4	58.6	75.3	53.9	78.5	92.2
	LC-BART	13.6	30.4	47,8	49.1	55.8	66.9	8.8	11.1	42.7	45.3	47.8	58.2	51.3	74.4	89.9
	Template	42.4	54.2	59.6	65.3	69.6	80.3	53.1	60.3	64.1	67.3	72.2	75.7	77.3	88.9	93.5
	BERT-MRC†	20.2	50.8	56.3	62.9	81.5	82.3	15.8	39.5	54.8	65.8	68.8	73.5	40.5	66.7	91.8
	LightNER	**62.9**	**75.6**	**78.8**	**82.2**	**84.5**	**85.7**	**58.1**	**67.4**	**69.5**	**73.7**	**78.4**	**81.1**	**86.9**	**89.4**	**93.9**

Table 2: Model performance (F1 score) in the cross-domain low-resource setting. "†" indicates that we rerun their public code in this setting. All of our experiments and baselines adopt large version of LMs.

Transfer Knowledge from a General Domain to Specific Domains We observe that the performance of prototype-based methods remains approximately the same as the number of labeled data increases, while LightNER continues to improve when the number of target-domain data increases. Table 2 shows that on all three target-domain datasets, LightNER significantly outperforms the other three types of baselines in the case of both 10 and 500 instances per entity type, From the perspective of quantifying the **knowledge transferred**, when the number of instances is 10, the performance of our model increase the F1-scores to 21.2%, 9.6%, and 10.6% on the MIT movie, MIT restaurant, and ATIS datasets, respectively, which are better than the results of knowledge transferred by *LC-BERT*. This demonstrates that our model is more successful in transferring the knowledge learned from the source domain.

Source	Methods	*MIT Restaurant*		
		10	20	50
None	Ours [BART]	48.5	58.0	62.0
	- pluggable module	**50.3**	**59.4**	**63.5**
	- unified learnable verbalizer	45.5	55.5	59.8
	Ours (Full-params Tuning)	49.5	59.0	62.8
	LC-BERT	21.8	39.4	52.7
	LC-BERT+[P-tuning]	24.9	41.2	53.5
	LC-BERT+[Adapter]	11.5	14.3	21.2
	Ours+[Adapter]	43.3	52.3	58.5
CoNLL03	Ours [BART]	**58.1**	**67.4**	**69.5**
	- pluggable module	54.5	64.2	67.8
	- unified learnable verbalizer	48.7	58.8	62.5
	Ours (Full-params Tuning)	53.7	63.5	66.9
	LC-BERT	27.2	40.9	56.3
	LC-BERT+[P-tuning]	30.3	46.8	58.2
	LC-BERT+[Adapter]	13.0	16.2	21.8
	Ours+[Adapter]	46.8	58.2	62.5

Table 3: Performance of Ablation and Variants Study.

5.3 Ablation and Comparison

As shown in the above experiments that our LightNER possess the outstanding ability of knowledge transfer in the low-resource setting, we demonstrate that the pluggable guidance module contributes to the cross-domain improvement. To this end, we ablate the pluggable module and unified learnable verbalizer to validate the effectiveness. - *pluggable module* indicates the entire parameter (100%) tuning without our proposed pluggable module. - *unified learnable verbalizer* donates our model only randomly assigns one token in the vocabulary to represent the type. From Table 3, we notice that only - *pluggable module* in the vanilla few-shot setting performs a little better than LightNER, but **decreases significantly in the cross-domain few-shot setting**. However, - *unified learnable verbalizer* drop both in the two settings. It further demonstrates that the design of the pluggable module is parameter-efficient and **beneficial for knowledge transfer**, while unified learnable verbalizer can **handle class transfer**, which is also essential for low-resource NER.

We further compare LightNER with several variants of our method: (i)*Ours (Full-params Tuning)*; (ii)*LC-BERT+[P-tuning]*: we set the length of continuous template words to be 10 for *P-tuning*; (iii)*LC-BERT+[Adapter]*; (iv)*Ours+[Adapter]*; **Firstly**, compared with LightNER, training all the parameters of our model merely improve a little in in vanilla few-shot setting, but drops significantly in cross-domain few-shot settings, which reveals that our pluggable module with LMs fixed is the vital for transferring knowledge across domains. **Secondly**, we observe that LC-BERT equipped with P-tuning achieves a few improvements both in vanilla few-shot and cross-domain few-shot settings. While Adapter makes performance drop significantly because LC-BERT cannot handle the class transfer, thus the few tuned parameters yield unsatisfactory performance. **Finally**, we replace

Methods	*NER → POS*			*POS → NER*		
	10	20	Full	10	20	Full
LC-BERT	44.3/46.2	53.7/54.3	91.4/91.7	37.9/38.3	48.4/48.6	91.7/91.3
LightNER	45.5/50.6	54.4/57.7	91.3/93.2	46.5/51.8	61.8/65.2	92.9/93.5

Table 4: Model performance in the cross-task setting. Number before and after "/" donate the F1 scores of training from scratch on target domain and transferring from source task to target task respectively.

Target	CoNLL			Movie			Restaurant		
Source	M	R	Mix	C	R	Mix	C	M	Mix
LC-BERT	0.2	0.4	0.0	0.5	0.3	0.0	0.3	0.2	0.0
Template	0.1	0.2	0.0	0.3	0.2	0.0	0.1	0.0	0.0
LightNER	**8.5**	**8.8**	**15.8**	**12.6**	**9.0**	**18.9**	**11.0**	**8.5**	**18.4**

Table 5: Cross-domain zero-shot performancefootnoteIn zero-shot setting, the weight of the unified learnable verbalizer is average operation.. C, M, and R refer to the dataset of CoNLL03, Movie, and Restaurant, respectively. The Mix column refers to the methods of averaging the parameters from the other two source domains (average the prompt for LightNER).

the pluggable module with the Adapter to validate the effectiveness of our module. The fact that *Ours+[Adapter]* performs significantly better than *LC-BERT+[Adapter]* demonstrates the superiority of our generative framework. Besides, *Ours+[Adapter]* behaves unsatisfactorily in a cross-domain low-resource setting, which reveals its poor ability of knowledge transfer for NER.

5.4 Detailed Model Analysis

The Transferability Across Task Although our LightNER is designed for NER, it is easy to generalize to other sequence tagging tasks without any modification network structure.

Thus, we try to train on full data of the source task, and then simply **load the pluggable guidance module** to further train the model on the target task. As shown in Table 4, we find LC-BERT has an extensive performance drop of all tasks in a cross-task setting, and we believe this is due to the task-specific classifier head hindering the generalization. The excellent performance in the cross-task setting proves that LightNER can adapt to other sequence labeling tasks and incredibly transfer knowledge across tasks.

Cross-Domain Zero-Shot Analysis with Mixed Guidance Parameters We leverage one dataset as the source domain and conduct the zero-shot experiments on target domains. From Table 5, we observe that our method can achieve F1-scores of approximately 10% in the cross-domain zero-shot setting, significantly higher than other methods. we further attempt to investigate the performance of mixing different pluggable guidance module. Specifically, we directly average the parameters of prompts from two source domains as a mixed prompt for the target domain and insert it into the generative framework to evaluate the target performance. From Table 5, we notice that mixed prompt achieves promising improvement, which is close to the addition of the results of the original two sources prompt-based model. We argue that this finding may also inspire future research directions of prompt-tuning and data augmentation.

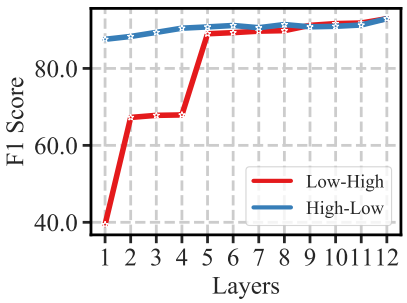

Figure 3: Performances on CoNLL03 as the layers of guidance module varies.

Low-high Layer vs. High-low Layer In the aforementioned experiments, we assign the pluggable guidance module to all layers in PLM. However, it is intuitive to investigate which layer is more sensitive with our approach. Intuitively, basic syntactic information may appear earlier in the PLM, while high-level semantic information emerges in higher-level layers (Clark et al., 2019). We conduct experiments by applying our pluggable module from the lowest to the highest layer and from the highest to the lowest layer separately. These two progressive methods are briefly denoted as low-high and high-low, respectively. As Figure 3 shows, the performance on CoNLL-2003 is close to the original result obtained after adding full-layer guidance module for tuning. This phenomenon also appears in the cross-domain few-shot setting. (Detailed results refer to Section 3 of Appendix C.2.). This proves that guidance module applied to higher layers of LMs can better stimulate knowledge from PLMs for downstream tasks more efficiently.

6 Conclusion and Future Work

In this paper, we propose a lightweight tuning paradigm for low-resource NER via pluggable

prompting (LightNER), which can accomplish the class transfer and domain transfer for low-resource NER without modifying the PLM's parameters and architecture. Note that we only tune pluggable guidance module with the whole parameter of the PLMs fixed, thus, making our approach lightweight and flexible for low-resource scenarios and can better transfer knowledge across domains and tasks. Experimental results reveal that LightNER can obtain competitive results in the rich-resource setting and outperform baselines in the low-resource setting. In the future, we plan to leverage knowledge graphs to enhance the pluggable guidance module for better knowledge transfer performance.

Acknowledgments

We want to express gratitude to the anonymous reviewers for their kind comments. This work was supported by National Natural Science Foundation of China (No.62206246, 91846204 and U19B2027), Zhejiang Provincial Natural Science Foundation of China (No. LGG22F030011), Ningbo Natural Science Foundation (2021J190), and Yongjiang Talent Introduction Programme (2021A-156-G).

Ethics/Broader Impact Statement

The ability to extract useful and important information from text, without the need for human input or control, has a wide range of practical and industrial applications. Named Entity Recognition is an important information extraction task that can benefit many NLP applications, e.g., information retrieval, dialog generation, and question answering. However, traditional fine-tuning-based approaches achieve poor performance in situations in which there are sparse data available. Our simple-yet-effective approach makes it possible to obtain better performance in these low-resource settings and we detail its advantages as:

(i) The few-shot setting is realistic (the number of labeled instance per class K can be any variable);

(ii) No necessity of prompt engineering;

(iii) Extensible to any pre-trained LMs (e.g., BERT or GPT-2).

Our motivation is to develop an applicable, generalizable lightweight tuning paradigm for the NLP community and our work is but a small step towards this direction.

References

Oshin Agarwal, Yinfei Yang, Byron C. Wallace, and Ani Nenkova. 2021. Interpretability analysis for named entity recognition to understand system predictions and how they can improve. *Comput. Linguistics*, 47(1):117–140.

Eyal Ben-David, Nadav Oved, and Roi Reichart. 2022. PADA: Example-based Prompt Learning for on-the-fly Adaptation to Unseen Domains. *Transactions of the Association for Computational Linguistics*, 10:414–433.

Tom B. Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel M. Ziegler, Jeffrey Wu, Clemens Winter, Christopher Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. 2020. Language models are few-shot learners. In *Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020, December 6-12, 2020, virtual*.

Kevin Clark, Urvashi Khandelwal, Omer Levy, and Christopher D Manning. 2019. What does bert look at? an analysis of bert's attention. In *Proceedings of the 2019 ACL Workshop BlackboxNLP: Analyzing and Interpreting Neural Networks for NLP*, pages 276–286.

Leyang Cui, Yu Wu, Jian Liu, Sen Yang, and Yue Zhang. 2021. Template-based named entity recognition using BART. In *Findings of the Association for Computational Linguistics: ACL/IJCNLP 2021, Online Event, August 1-6, 2021*, volume ACL/IJCNLP 2021 of *Findings of ACL*, pages 1835–1845. Association for Computational Linguistics.

Leyang Cui and Yue Zhang. 2019. Hierarchically-refined label attention network for sequence labeling. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing, EMNLP-IJCNLP 2019, Hong Kong, China, November 3-7, 2019*, pages 4113–4126. Association for Computational Linguistics.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT 2019, Minneapolis, MN, USA, June 2-7, 2019, Volume 1 (Long and Short Papers)*, pages 4171–4186. Association for Computational Linguistics.

Ning Ding, Guangwei Xu, Yulin Chen, Xiaobin Wang, Xu Han, Pengjun Xie, Haitao Zheng, and Zhiyuan Liu. 2021. Few-nerd: A few-shot named entity recognition dataset. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing, ACL/IJCNLP 2021, (Volume 1: Long Papers), Virtual Event, August 1-6, 2021*, pages 3198–3213. Association for Computational Linguistics.

Jonathan Frankle and Michael Carbin. 2019. The lottery ticket hypothesis: Finding sparse, trainable neural networks. In *7th International Conference on Learning Representations, ICLR 2019, New Orleans, LA, USA, May 6-9, 2019*. OpenReview.net.

Alexander Fritzler, Varvara Logacheva, and Maksim Kretov. 2019. Few-shot classification in named entity recognition task. In *Proceedings of the 34th ACM/SIGAPP Symposium on Applied Computing, SAC 2019, Limassol, Cyprus, April 8-12, 2019*, pages 993–1000. ACM.

Tianyu Gao, Adam Fisch, and Danqi Chen. 2021. Making pre-trained language models better few-shot learners. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing, ACL/IJCNLP 2021, (Volume 1: Long Papers), Virtual Event, August 1-6, 2021*, pages 3816–3830. Association for Computational Linguistics.

Tao Gui, Jiacheng Ye, Qi Zhang, Zhengyan Li, Zichu Fei, Yeyun Gong, and Xuanjing Huang. 2020. Uncertainty-aware label refinement for sequence labeling. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing, EMNLP 2020, Online, November 16-20, 2020*, pages 2316–2326. Association for Computational Linguistics.

Demi Guo, Alexander M. Rush, and Yoon Kim. 2021. Parameter-efficient transfer learning with diff pruning. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing, ACL/IJCNLP 2021, (Volume 1: Long Papers), Virtual Event, August 1-6, 2021*, pages 4884–4896. Association for Computational Linguistics.

Dilek Hakkani-Tür, Gökhan Tür, Asli Celikyilmaz, Yun-Nung Chen, Jianfeng Gao, Li Deng, and Ye-Yi Wang. 2016. Multi-domain joint semantic frame parsing using bi-directional RNN-LSTM. In *Interspeech 2016, 17th Annual Conference of the International Speech Communication Association, San Francisco, CA, USA, September 8-12, 2016*, pages 715–719. ISCA.

Xu Han, Weilin Zhao, Ning Ding, Zhiyuan Liu, and Maosong Sun. 2021. PTR: prompt tuning with rules for text classification. *CoRR*, abs/2105.11259.

Matthew Henderson and Ivan Vulic. 2021. Convex: Data-efficient and few-shot slot labeling. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT 2021, Online, June 6-11, 2021*, pages 3375–3389. Association for Computational Linguistics.

Yutai Hou, Wanxiang Che, Yongkui Lai, Zhihan Zhou, Yijia Liu, Han Liu, and Ting Liu. 2020. Few-shot slot tagging with collapsed dependency transfer and label-enhanced task-adaptive projection network. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, ACL 2020, Online, July 5-10, 2020*, pages 1381–1393. Association for Computational Linguistics.

Neil Houlsby, Andrei Giurgiu, Stanislaw Jastrzebski, Bruna Morrone, Quentin de Laroussilhe, Andrea Gesmundo, Mona Attariyan, and Sylvain Gelly. 2019. Parameter-efficient transfer learning for NLP. In *Proceedings of the 36th International Conference on Machine Learning, ICML 2019, 9-15 June 2019, Long Beach, California, USA*, volume 97 of *Proceedings of Machine Learning Research*, pages 2790–2799. PMLR.

Jiaxin Huang, Chunyuan Li, Krishan Subudhi, Damien Jose, Shobana Balakrishnan, Weizhu Chen, Baolin Peng, Jianfeng Gao, and Jiawei Han. 2020. Few-shot named entity recognition: A comprehensive study. *CoRR*, abs/2012.14978.

Brian Lester, Rami Al-Rfou, and Noah Constant. 2021. The power of scale for parameter-efficient prompt tuning. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing, EMNLP 2021, Virtual Event / Punta Cana, Dominican Republic, 7-11 November, 2021*, pages 3045–3059. Association for Computational Linguistics.

Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Veselin Stoyanov, and Luke Zettlemoyer. 2020. BART: denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, ACL 2020, Online, July 5-10, 2020*, pages 7871–7880. Association for Computational Linguistics.

Xiang Lisa Li and Percy Liang. 2021. Prefix-tuning: Optimizing continuous prompts for generation. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing, ACL/IJCNLP 2021, (Volume 1: Long Papers), Virtual Event, August 1-6, 2021*, pages 4582–4597. Association for Computational Linguistics.

Xiaoya Li, Jingrong Feng, Yuxian Meng, Qinghong Han, Fei Wu, and Jiwei Li. 2020. A unified MRC framework for named entity recognition. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, ACL 2020, Online,*

July 5-10, 2020, pages 5849–5859. Association for Computational Linguistics.

Hongyu Lin, Yaojie Lu, Xianpei Han, and Le Sun. 2019. Sequence-to-nuggets: Nested entity mention detection via anchor-region networks. In *Proceedings of the 57th Conference of the Association for Computational Linguistics, ACL 2019, Florence, Italy, July 28- August 2, 2019, Volume 1: Long Papers*, pages 5182–5192. Association for Computational Linguistics.

Jingjing Liu, Panupong Pasupat, Scott Cyphers, and James R. Glass. 2013. Asgard: A portable architecture for multilingual dialogue systems. In *IEEE International Conference on Acoustics, Speech and Signal Processing, ICASSP 2013, Vancouver, BC, Canada, May 26-31, 2013*, pages 8386–8390. IEEE.

Kun Liu, Yao Fu, Chuanqi Tan, Mosha Chen, Ningyu Zhang, Songfang Huang, and Sheng Gao. 2021a. Noisy-labeled NER with confidence estimation. In *Proceedings of NAACL*, pages 3437–3445. Association for Computational Linguistics.

Kun Liu, Yao Fu, Chuanqi Tan, Mosha Chen, Ningyu Zhang, Songfang Huang, and Sheng Gao. 2021b. Noisy-labeled NER with confidence estimation. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT 2021, Online, June 6-11, 2021*, pages 3437–3445. Association for Computational Linguistics.

Xiao Liu, Yanan Zheng, Zhengxiao Du, Ming Ding, Yujie Qian, Zhilin Yang, and Jie Tang. 2021c. GPT understands, too. *CoRR*, abs/2103.10385.

Yijin Liu, Fandong Meng, Jinchao Zhang, Jinan Xu, Yufeng Chen, and Jie Zhou. 2019. GCDT: A global context enhanced deep transition architecture for sequence labeling. In *Proceedings of the 57th Conference of the Association for Computational Linguistics, ACL 2019, Florence, Italy, July 28- August 2, 2019, Volume 1: Long Papers*, pages 2431–2441. Association for Computational Linguistics.

Ruotian Ma, Xin Zhou, Tao Gui, Yiding Tan, Linyang Li, Qi Zhang, and Xuanjing Huang. 2022. Template-free prompt tuning for few-shot NER. In *Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL 2022, Seattle, WA, United States, July 10-15, 2022*, pages 5721–5732. Association for Computational Linguistics.

Xuezhe Ma and Eduard H. Hovy. 2016. End-to-end sequence labeling via bi-directional lstm-cnns-crf. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics, ACL 2016, August 7-12, 2016, Berlin, Germany, Volume 1: Long Papers*. The Association for Computer Linguistics.

Clifton Poth, Jonas Pfeiffer, Andreas Rücklé, and Iryna Gurevych. 2021. What to pre-train on? efficient intermediate task selection. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing, EMNLP 2021, Virtual Event / Punta Cana, Dominican Republic, 7-11 November, 2021*, pages 10585–10605. Association for Computational Linguistics.

Erik F. Tjong Kim Sang and Fien De Meulder. 2003. Introduction to the conll-2003 shared task: Language-independent named entity recognition. In *Proceedings of the Seventh Conference on Natural Language Learning, CoNLL 2003, Held in cooperation with HLT-NAACL 2003, Edmonton, Canada, May 31 - June 1, 2003*, pages 142–147. ACL.

Victor Sanh, Thomas Wolf, and Alexander M. Rush. 2020. Movement pruning: Adaptive sparsity by fine-tuning. In *Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020, December 6-12, 2020, virtual*.

Timo Schick, Helmut Schmid, and Hinrich Schütze. 2020. Automatically identifying words that can serve as labels for few-shot text classification. In *Proceedings of the 28th International Conference on Computational Linguistics, COLING 2020, Barcelona, Spain (Online), December 8-13, 2020*, pages 5569–5578. International Committee on Computational Linguistics.

Timo Schick and Hinrich Schütze. 2021. Exploiting cloze-questions for few-shot text classification and natural language inference. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume, EACL 2021, Online, April 19 - 23, 2021*, pages 255–269. Association for Computational Linguistics.

Abigail See, Peter J. Liu, and Christopher D. Manning. 2017. Get to the point: Summarization with pointer-generator networks. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics, ACL 2017, Vancouver, Canada, July 30 - August 4, Volume 1: Long Papers*, pages 1073–1083. Association for Computational Linguistics.

Taylor Shin, Yasaman Razeghi, Robert L. Logan IV, Eric Wallace, and Sameer Singh. 2020. Autoprompt: Eliciting knowledge from language models with automatically generated prompts. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing, EMNLP 2020, Online, November 16-20, 2020*, pages 4222–4235. Association for Computational Linguistics.

Emma Strubell, Patrick Verga, David Belanger, and Andrew McCallum. 2017. Fast and accurate entity recognition with iterated dilated convolutions. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing, EMNLP 2017, Copenhagen, Denmark, September 9-11, 2017*, pages 2670–2680. Association for Computational Linguistics.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, December 4-9, 2017, Long Beach, CA, USA*, pages 5998–6008.

Sam Wiseman and Karl Stratos. 2019. Label-agnostic sequence labeling by copying nearest neighbors. In *Proceedings of the 57th Conference of the Association for Computational Linguistics, ACL 2019, Florence, Italy, July 28- August 2, 2019, Volume 1: Long Papers*, pages 5363–5369. Association for Computational Linguistics.

Ikuya Yamada, Akari Asai, Hiroyuki Shindo, Hideaki Takeda, and Yuji Matsumoto. 2020. LUKE: deep contextualized entity representations with entity-aware self-attention. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing, EMNLP 2020, Online, November 16-20, 2020*, pages 6442–6454. Association for Computational Linguistics.

Hang Yan, Tao Gui, Junqi Dai, Qipeng Guo, Zheng Zhang, and Xipeng Qiu. 2021. A unified generative framework for various NER subtasks. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing, ACL/IJCNLP 2021, (Volume 1: Long Papers), Virtual Event, August 1-6, 2021*, pages 5808–5822. Association for Computational Linguistics.

Jie Yang, Shuailong Liang, and Yue Zhang. 2018. Design challenges and misconceptions in neural sequence labeling. In *Proceedings of the 27th International Conference on Computational Linguistics, COLING 2018, Santa Fe, New Mexico, USA, August 20-26, 2018*, pages 3879–3889. Association for Computational Linguistics.

Yi Yang and Arzoo Katiyar. 2020. Simple and effective few-shot named entity recognition with structured nearest neighbor learning. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing, EMNLP 2020, Online, November 16-20, 2020*, pages 6365–6375. Association for Computational Linguistics.

Juntao Yu, Bernd Bohnet, and Massimo Poesio. 2020. Named entity recognition as dependency parsing. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, ACL 2020, Online, July 5-10, 2020*, pages 6470–6476. Association for Computational Linguistics.

Jeffrey O. Zhang, Alexander Sax, Amir Roshan Zamir, Leonidas J. Guibas, and Jitendra Malik. 2020a. Side-tuning: A baseline for network adaptation via additive side networks. In *Computer Vision - ECCV 2020 - 16th European Conference, Glasgow, UK, August 23-28, 2020, Proceedings, Part III*, volume 12348 of *Lecture Notes in Computer Science*, pages 698–714. Springer.

Ningyu Zhang, Shumin Deng, Zhen Bi, Haiyang Yu, Jiacheng Yang, Mosha Chen, Fei Huang, Wei Zhang, and Huajun Chen. 2020b. Openue: An open toolkit of universal extraction from text. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations, EMNLP 2020 - Demos, Online, November 16-20, 2020*, pages 1–8. Association for Computational Linguistics.

Xin Zhou, Ruotian Ma, Tao Gui, Yiding Tan, Qi Zhang, and Xuanjing Huang. 2021. Plug-tagger: A pluggable sequence labeling framework using language models. *CoRR*, abs/2110.07331.

Morteza Ziyadi, Yuting Sun, Abhishek Goswami, Jade Huang, and Weizhu Chen. 2020. Example-based named entity recognition. *CoRR*, abs/2008.10570.

A Detailed Statistics of Datasets

We take the standard split of CoNLL03 by following Sang and Meulder (2003), and splits MIT Movie Review, MIT Restaurant Review and ATIS by following Liu et al. (2013). Table 6 presents detailed statistics of our datasets. The standard precision, recall and F1 score are used for model evaluation.

Dataset	# Train	# Test	# Entity
CoNLL03	12.7k	3.2k	4
MIT Restaurant	7.6k	1.5k	8
MIT Review	7.8k	2k	12
ATIS	4.6k	850	79

Table 6: Statistic of datasets.

Models	PER	ORG	LOC*	MISC*	Overall
LC-BERT	76.25	75.32	61.55	59.35	68.12
LC-BART	75.70	73.59	58.70	57.30	66.82
Template	84.49	72.61	71.98	73.37	75.59
LightNER	**90.96**	**76.88**	**81.57**	**82.08**	**78.97**

Table 7: In-domain low-resource performance on the CoNLL-2003 dataset. * indicates the low-resource entity type.

B Experimental Details

This section details the training procedures and hyperparameters for each of the datasets. Considering the instability of the few-shot learning, we run each experiment 5 times on the random seed [1, 2, 49, 4321, 1234] and report the averaged performance. We utilize Pytorch to conduct experiments with 1 Nvidia 3090 GPUs. All optimizations are performed with the AdamW optimizer with a linear warmup of learning rate over the first 10% of

gradient updates to a maximum value, then linear decay over the remainder of the training. We set the hyper-parameter α as 0.5. And weight decay on all non-bias parameters is set to 0.01. We describe the details of the training hyper-parameters in the following sections.

B.1 Standard Supervised Setting

For all models, we fix the batch size as 16 and search for the learning rates in varied intervals [1e-5, 5e-5]. We train the model for 30 epochs and do evaluation after 20 epoch. We choose the model performing the best on the validation set and evaluate it on the test set.

B.2 Low-Resource Setting

We fix the batch size as 16 and search for the learning rates in varied intervals [3e-5, 5e-5]. We train the model for 30 epochs and do evaluation after 20 epoch. We choose the model performing the best on the validation set and evaluate it on the test set.

B.3 Cross-Task Setting

We fix the batch size as 8 and search for the learning rates in varied intervals [2e-5, 5e-5]. We train the model for 30 epochs and do evaluation after 25 epoch. We choose the model performing the best on the validation set and evaluate it on the test set.

C Supplementary Experimental Results

C.1 In-Domain Low-Resource NER Setting

Following (Cui et al., 2021), we construct few-shot learning scenarios on CoNLL-2003 by down-sampling, which limits the number of training instances for certain specific categories. Particularly, we choose " LOC" and "MISC" as the low-resource entities and "PER" and "ORG" as the rich-resource entities. The rich and low-resource entity categories have the same textual domain. Specifically, we downsample the CoNLL-2003 training set and generate 4,001 training instances, including 2,496 "PER," 3,763 "ORG," 50 "MISC," and 50 "LOC" entities. As shown in Table 7, our method outperforms other methods for both rich- and low-resource entity types. This proves that our proposed method has a more substantial performance for in-domain few-shot NER and demonstrates that it can effectively handle the class transfer, which is a challenging aspect in few-shot NER tasks.

C.2 The Performance in the Low-Resource Setting When the Prompt Layer Varies

Intuitively, basic syntactic information may appear earlier in the PLM, while high-level semantic information emerges in higher-level layers. Table 8 shows that the performance of prompts within highest 1 layer is better than lowest 1 layer overall, and the performance of highest 6 layers is close to the result of all 12 layers. This proves that prompts applied to higher layers of LMs can better stimulate knowledge from PLMs for downstream tasks more efficiently.

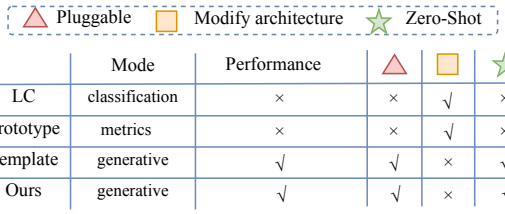

	Mode	Performance	△	□	☆
LC	classification	×	×	√	×
Prototype	metrics	×	×	√	×
Template	generative	√	√	×	√
Ours	generative	√	√	×	√

Figure 4: We show the formulations of different NER models and illustrate their corresponding strengths. Zero-shot refers to zero-shot learning ability; LC is short for label-specific classifier (vanilla sequence labeling)

Algorithm 1 Decoding Algorithm to Convert the Entity Index Sequence into Entity Spans

Require: n, the number of tokens in X; m, the number of entity types; target sequence $Y = [y_1, ..., y_{3l}]$, l is the number of entities; and we have $y_t \in [1, n+m]$

Ensure: Entity spans $E = \{(e_1^{start}, e_1^{end}, t_1), ..., (e_i^{start}, e_i^{end}, t_i)\}$

$E = \{\}, e = [], i = 1$
while $i <= 3l$ **do**
  $y_i = Y[i]$
  if $y_i > n$ **then**
    $E.add((e, C_{y_i - n}))$
    $e = []$
  else
    $e.append(y_i)$
  end if
  $i+ = 1$
end while
return E

C.3 Impact of Length of Guidance Module

We set the length of prompts as 10 in the above experiment and analyze whether the impact of the length of the prompt. From Figure 5, we notice that

Source	Methods	*MIT Movie*			*MIT Restaurant*			*ATIS*		
		10	20	50	10	20	50	10	20	50
None	Template	37.3	48.5	52.2	46.0	57.1	58.7	71.7	79.4	92.6
	LightNER(lowest 1 layer)	16.3	20.3	30.5	14.6	23.4	25.4	30.6	38.3	44.2
	LightNER(highest 1 layer)	29.5	38.4	45.5	35.4	45.3	50.5	60.1	69.8	78.7
	LightNER(highest 6 layers)	38.5	50.5	69.8	44.3	55.7	59.8	70.2	80.2	88.4
	LightNER(all 12 layers)	**41.7**	**57.8**	**73.1**	**48.5**	**58.0**	**62.0**	**76.3**	**85.3**	**92.8**
CoNLL03	Template	42.4	54.2	59.6	53.1	60.3	64.1	77.3	88.9	93.5
	LightNER(lowest 1 layer)	24.3	30.5	35.4	15.6	22.4	27.5	37.9	44.5	48.3
	LightNER(highest 1 layer)	44.6	59.3	74.3	39.4	45.2	51.7	59.7	68.5	79.2
	LightNER(highest 6 layers)	55.8	69.7	75.8	50.7	62.7	66.7	79.2	86.3	91.8
	LightNER(all 12 layers)	**62.9**	**75.6**	**78.8**	**58.1**	**67.4**	**69.5**	**86.9**	**89.4**	**93.9**

Table 8: Performances in cross-domain low-resource setting as the prompt layer varies.

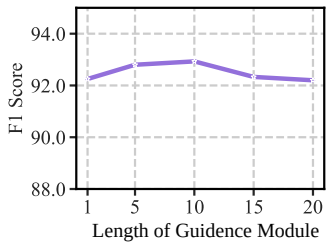

Figure 5: Performances on CoNLL03 as the length of guidance module varies.

a longer prompt implies more trainable parameters but does not guarantee more expressive power. It also reveals that our pluggable guidance module is stable; as the length changes, the performance fluctuation does not exceed 1%.

D Comprehensive Comparison

We carry out a comprehensive comparison with related methods as shown in Figure 4. For a given sequence, the computational complexity of our LightNER is $O\left(n^2 d\right)$ and Tamplate-based BART is $O\left(nm\hat{n} \cdot n^2 d\right)$,where d donates the dimension of the LMs; $\mathbf{n}, \mathbf{m}, \hat{\mathbf{n}}$ imply the length of input, number of entity classes and n-grams, respectively. Note that our approach does not need to enumerate all possible spans and construct templates, which is efficient than the Template-based method (Cui et al., 2021). Moreover, we only tune 2.2% parameters of the whole model (the tuned params divided by params of the LM), making it memory efficient during training.

E The decoding algorithm for converting process

The decoding algorithm for converting the predicted index sequence to entity spans is shown in Algorithm 1.

F Sampling strategy in low-resource setting

F.1 Cross-Domain

We simulate the cross-domain low-resource data scenarios by random sampling training instances from a large training set as the training data in the target domain. We use different numbers of instances for training, randomly sampling a number of instances per entity type (10, 20, 50, 100, 200, 500 instances per entity tag for MIT Movie and MIT restaurant, and 10, 20, 50 instances per entity tag for ATIS). For different instances per entity tag, we sample five times on the random seed [1, 2, 49, 4321, 1234] and report the averaged performance.

In order to alleviate the problem that an instance usually contains multiple entities, we first sort the entity tags according to the number of instances included. Then we sample instances in the sequence of the sorted order. After once sampling, we will update the status(the remaining number of instances to be sampled) of the entity tags. If an entity tag exceeds the limit after the sampling, discard this sampling.

F.2 Cross-Task

Since the CoNLL-2003 dataset contains both entity tag and POS information, we use the same sampling data in cross-task setting. For different instances per entity tag, we sample five times on the random seed [1, 2, 49, 4321, 1234] and report the averaged performance.

Cross-modal Contrastive Attention Model for Medical Report Generation

Xiao Song [1], Xiaodan Zhang[1], Junzhong Ji[1,*], Ying Liu[2], Pengxu Wei [3]
[1] Beijing University of Technology
[2] Peking University Third Hospital
[3] Sun Yat-sen University
xiaos@emails.bjut.edu.cn; {zhangxiaodan, jjz01}@bjut.edu.cn;
lyyulia@163.com; weipx3@mail.sysu.edu.cn

Abstract

Medical report automatic generation has gained increasing interest recently as a way to help radiologists write reports more efficiently. However, this image-to-text task is rather challenging due to the typical data biases: 1) Normal physiological structures dominate the images, with only tiny abnormalities; 2) Normal descriptions accordingly dominate the reports. Existing methods have attempted to solve these problems, but they neglect to exploit useful information from similar historical cases. In this paper, we propose a novel Cross-modal Contrastive Attention (CMCA) model to capture both visual and semantic information from similar cases, with mainly two modules: a Visual Contrastive Attention Module for refining the unique abnormal regions compared to the retrieved case images; a Cross-modal Attention Module for matching the positive semantic information from the case reports. Extensive experiments on two widely-used benchmarks, IU X-Ray and MIMIC-CXR, demonstrate that the proposed model outperforms the state-of-the-art methods on almost all metrics. Further analyses also validate that our proposed model is able to improve the reports with more accurate abnormal findings and richer descriptions.

1 Introduction

Medical report generation task in practice demands radiologists carefully examine details of images and write corresponding reports, which is time-consuming and technically rigorous. In addition, with the explosion of medical images, generating medical reports has increasingly become a tough burden for radiologists in clinical diagnosis and treatment. Thus, it is extremely desired for automatically generating medical reports from medical images, which has also attracted increasing attention especially in the Chest X-ray report generation.

Recently, the widely used Encoder-Decoder framework in image captioning task (Karpathy and Fei-Fei, 2015; You et al., 2016; Vaswani et al., 2017; Anderson et al., 2018) has been successfully inherited by medical report generation and has made great improvements (Jing et al., 2018; Zhang et al., 2017; Shin et al., 2016a; Wang et al., 2018; Li et al., 2019; Chen et al., 2020). Nevertheless, as shown in Figure 1, different from image captioning, medical report generation faces typical data biases which cause the failing of generating accurate descriptions: 1) the abnormal regions are usually tiny, rare and hard to be recognized in medical images with monotonous and homogeneous features (Guan et al., 2021; Li et al., 2018b; Guan et al., 2020); 2) the abnormal text descriptions are correspondingly rare in reports and the normal descriptions dominate the whole datasets (Shin et al., 2016b; Xue et al., 2018; Jing et al., 2019; Liu et al., 2021a,b).

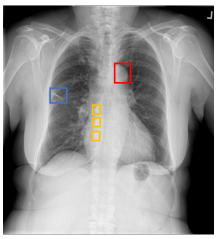

Ground Truth: There is scarring in the right mid and upper lung zone with surgical clips identified as well. There is no pleural effusion or pneumothorax. The heart is not significantly enlarged. There are atherosclerotic changes of the aorta. Arthritic changes of the skeletal structures are noted.

Baseline: The lungs are clear. There is no pleural effusion or pneumothorax. The heart and mediastinum are normal. The skeletal structures are normal.

CMCA: The lungs are clear. There is no pleural effusion or pneumothorax. The heart is not significantly enlarged. There are calcified mediastinal lymph. There are atherosclerotic changes of the aorta. Arthritic changes of the skeletal structures are noted.

Figure 1: One example of Chest X-ray image with the corresponding ground truth, Baseline (Vaswani et al., 2017) and our model generated reports. The abnormal regions and their corresponding descriptions are marked in same colors, showing serious data biases of this task.

To tackle these issues, Jing et al. (Jing et al., 2019) employed an auxiliary detector to identify abnormality terms. Liu et al. (Liu et al., 2021b) compared the input image with normal samples to distill the visual abnormal information. However, these approaches mainly probed abnormalities from images themselves or comparing with manual selected normal samples, without considering the importance of exploiting abnormal information from historical similar cases and making use of their visual and semantic information. Based

*Corresponding Author.

Proceedings of the 29th International Conference on Computational Linguistics, pages 2388–2397
October 12–17, 2022.

on the observation that similar images are more likely to have similar reports (Ramos et al., 2014), we presume that taking the most similar historical case as a contrastive reference will make models able to relieve the data biases and capture more critical visual and semantic information. Unfortunately, exploring positive semantic information from cases faces another challenge on cross-modal matching (Xu et al., 2020; Liang et al., 2021): the retrieved report which contains useful but noisy semantic information is hard to be aligned with the input image solely across the unmatched visual-semantic modalities.

In this paper, we propose a novel Cross-modal Contrastive Attention (CMCA) model to tackle the aforementioned problems. CMCA first retrieves the most similar case for the input image from a historical database, then generates a contrastive feature by enlarging the differences and maintaining the commons between the input image and the retrieved image. Subsequently, the contrastive feature is used to extract visually abnormal and semantically matched information through two modules: a Visual Contrastive Attention Module (VCAM) and a Cross-modal Attention Module (CAM). Specifically, VCAM extracts discriminative abnormal visual information from the contrastive feature, where the unique abnormal regions of input image are enhanced and similar regions are retained. CAM matches the positive semantic information from the retrieved report by aligning it with the contrastive feature, which builds interactions across the unmatched visual and semantic modalities. Finally, we propose a Parallel Attention Module (PAM) to further enhance the feature representation for generating accurate report. Extensive experiments on the widely-used benchmark IU X-Ray (Demner-Fushman et al., 2016) dataset and the largest public MIMIC-CXR (Johnson et al., 2019) dataset demonstrate that our model outperforms the state-of-the-art methods.

Overall, our contributions are as follows:

- We propose to take the most similar historical case as a contrastive reference to relieve the data biases for medical report generation.
- We propose a novel Cross-modal Contrastive Attention model to distill unique abnormal features for input image and match positive words from the case report.
- Extensive experimental results on the public IU X-Ray and MIMIC-CXR datasets demonstrate the effectiveness of the proposed model.
- We conduct analyses to validate the hypothesis that historical similar cases can significantly assist this task, and our CMCA model is able to generate reports with more accurate abnormal findings and richer descriptions.

2 Method

In this section, we introduce the background and the details of the proposed CMCA model in order.

2.1 Background

2.1.1 Overall Framework

Our CMCA model follows the Encoder-Decoder pipeline, as shown in Figure 2.

In Encoder: Firstly, the spatial visual feature X_I of input image I can be extracted through a DenseNet (Huang et al., 2017) network:

$$X_I = DenseNet(I), \quad (1)$$

Then, we introduce a Visual Attention Module (VAM), a Visual Contrastive Attention Module (VCAM) and a Cross-modal Attention Module (CAM) to respectively extract: 1) the visual attention feature V^a of X_I, 2) the visual contrastive attention feature V_c^a between X_I and the spatial visual feature of the retrieved similar case X_d, 3) the cross-modal attention feature Cr^a which matches the useful semantic information from the report of retrieved similar case R_d using the contrastive feature compared with X_I and X_d:

$$V^a = VAM(X_I), \quad (2)$$
$$V_c^a = VCAM(X_I, X_d), \quad (3)$$
$$Cr^a = CAM(X_I, X_d, R_d). \quad (4)$$

In Decoder: we propose a Parallel Attention Module (PAM) that allows the decoder model parallel grasping encoder features:

$$\tilde{Y} \leftarrow PAM(V^a, V_c^a, Cr^a). \quad (5)$$

where $\tilde{Y} = \{\tilde{y_1}, \tilde{y_2}, ..., \tilde{y_T}\}$ are word tokens of the generated report.

2.1.2 Basic Modules

Our proposed method is accomplished on stacks of identical Multi-head Attention (MHA) layers and Position-wised Feed-Forward Network (FFN) layers. The given input feature X are firstly converted into queries Q, keys K and values V:

$$Q = XW^Q, K = XW^K, V = XW^V, \quad (6)$$

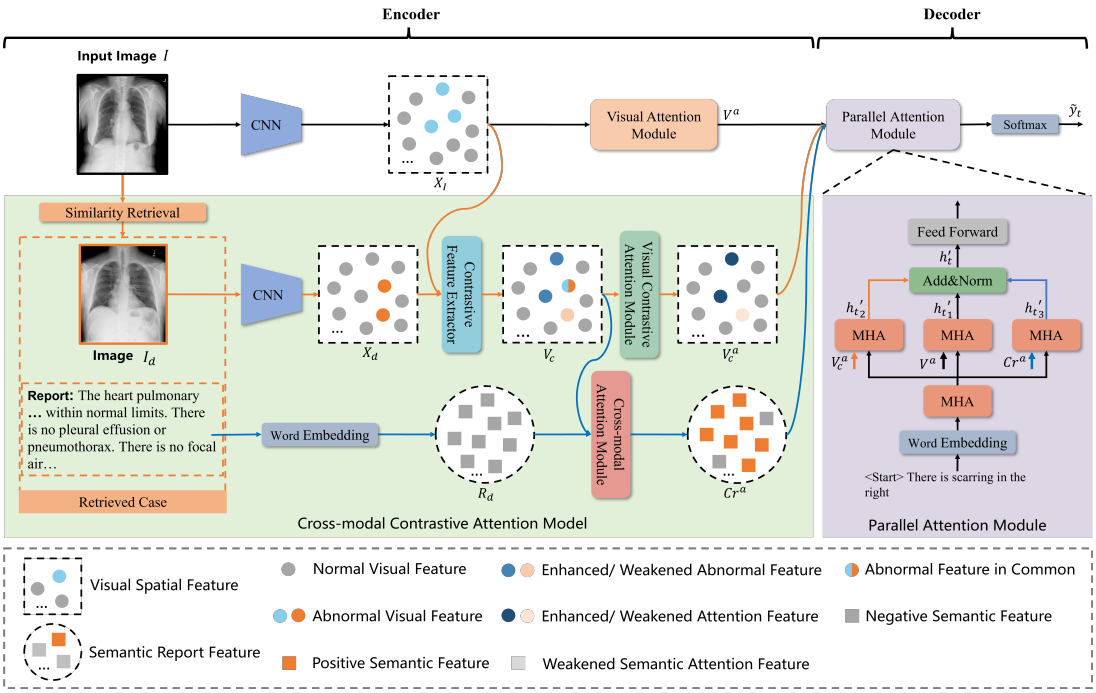

Figure 2: Overview of our proposed Cross-modal Contrastive Attention (CMCA) model for Medical Report Generation. In encoder, CMCA first retrieves the most similar image I_d, and generates a contrastive feature V_c through the contrastive feature extractor. Then, the Visual Contrastive Attention Module distills the unique abnormal features of I (blue circles), and the Cross-modal Attention Module matches the positive semantic information from the case report R_d. In decoder, the encoded features are integrated through a stack of Parallel Attention Modules to more effectively generate the final word $\tilde{y}_t$.

where $W^Q, W^K, W^V \in \mathbb{R}^{d\times d}$ are learned weights. In each MHA layer, the inputs are divided into h parallel attention heads, which allows the model to focus on the different representation sub-spaces of different positions jointly. And then the attention features of all heads can be calculated and concatenated as follows:

$$MHA(Q,K,V) = [head_1, ..., head_h]W^O, \quad (7)$$

$$head_i = Attn(QW_i^Q, KW_i^K, VW_i^V), \quad (8)$$

$$Attn(q,k,v) = softmax(\frac{qk^T}{\sqrt{d_k}})v. \quad (9)$$

where $W^O \in \mathbb{R}^{d\times d}$ and $W_i^Q, W_i^K, W_i^V \in \mathbb{R}^{d\times \frac{d}{h}}$ are learned parameters, and $[\cdot]$ indicates the concatenation operation.

Then, the FFN layer is applied as follows:

$$FFN(x) = max(0, xW_1 + b_1)W_2 + b_2. \quad (10)$$

where W_1, W_2 are learned parameters and b_1, b_2 are biases. There is also a residual connection around each of the MHA and FFN layers, followed by layer normalization.

2.1.3 Retrieval Case Database

For further retrieving the most similar historical cases of the input images, we create a Retrieval Case Database $\hat{DB}$. All of the records in $\hat{DB}$ are derived from the training sets, and each of them is a triplet consisting the global visual feature v_d of the image to be used for retrieval, the spatial visual feature X_d of the image to be used for calculating the contrastive feature compared to the input image, and the associated report R_d to be used for cross-modal alignment, which can be denoted by $< v_d, X_d, R_d >$. It can be noted that the spatial visual feature X_d is also extracted by DenseNet in Eq. 1, and the global visual feature v_d can be obtained by a average-pooling operation on X_d.

2.2 Cross-modal Contrastive Attention Model

To effectively extract critical abnormal visual features and positive semantic features from the input image and the most similar case, our CMCA model is proposed with mainly two modules: VCAM and CAM, both of which are applied based on the proposed simple-yet-effective contrastive feature, as shown in Figure 2.

Contrastive feature: To obtain the contrastive feature, we firstly retrieve the most similar historical case of the input image based on the historical Case Database $\hat{DB}$ established in Sec. 2.1.3 which includes the global and spatial visual features of case images and their corresponding reports, we

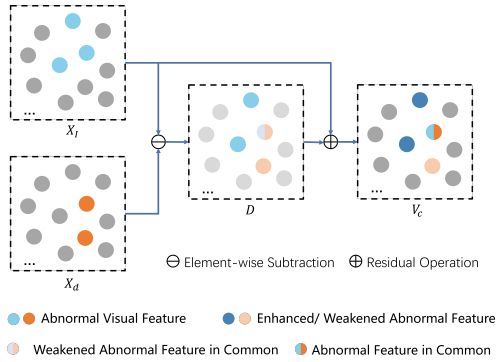

Figure 3: Structure of the proposed contrastive feature extractor. The circles in blue and orange correspond to the abnormal parts of the input image feature X_I and the retrieved image feature X_d, respectively.

retrieve the most similar case from $\hat{DB}$ by computing the highest cosine similarity between the global visual feature of input image v_I and historical image global features in case database $v_{\hat{DB}}$:

$$Case^{N_K} \leftarrow max(cosine(v_I, v_{\hat{DB}})). \quad (11)$$

where $Case^{N_K}$ denotes the retrieved N_K historical cases with the highest cosine similarity. It can be noted that we utilize each of the retrieved cases independently for the remaining task.

As shown in Figure 3, for one retrieved case $Case_d$, we calculate the contrastive feature V_c between the spatial visual features X_I and $X_d \in \mathbb{R}^{7\times7\times d}$ of the input image I and the case image I_d, the contrastive feature can be obtained through the following operations:

$$D = X_I - X_d, \quad (12)$$

$$V_c = D + X_I. \quad (13)$$

where D and $V_c \in \mathbb{R}^{7\times7\times d}$ denote the difference value of the spatial features and the final visual contrastive feature. The residual operation aims to tackle the problem of zero or negative value in D, and keep more representation of the input visual feature X_I for further cross-modal matching.

For further illustrating the ability of contrastive feature to distill the abnormal visual information, our explanations are as follows: As shown in Figure 3, we suppose the gray circles in X_I and X_d represent the normal portions in input and case images, respectively, while the colored circles represent the abnormal regions. Accordingly, there are four statuses in the contrastive feature V_c:

- The blue circles denote the regions of the input image I that are abnormal but normal in the case image I_d. As a result, the contrastive features of these regions can be expressed as $V_c^{blue} = X_I^{blue} + D^{blue}$, thus the distinctive abnormal regions of image I are reinforced.

- The orange circles denote the regions where I are abnormal but I_d are normal. The contrastive features decrease the distinctive abnormal regions of case I_d in orange, since $V_c^{orange} = X_I^{orange} + D^{orange}$ and the value of D^{orange} is negative.

- Identical regions are indicated by gray circles in two input spatial features. As a result, the differences of these parts are zero: $D^{gray} = 0$. The contrastive features of these regions are equivalent to the original spatial visual feature of I, which is indicated by $V_c^{gray} = X_I^{gray}$.

- The abnormal regions which occur in both I and I_d are the mixed color circles. As a result, the difference of these parts are also zero: $D^{mix} = 0$. And the contrastive features of these portions are also the original visual features of I, $V_c^{mix} = X_I^{blue}$.

In summary, the contrastive feature $V_c = \{V_c^{blue}, V_c^{orange}, V_c^{gray}, V_c^{mix}\}$ enhances the unique abnormal regions of the input image I, retains the identical regions, and weakens the unique abnormal regions of the case image I_d. Base on V_c, the following two modules respectively explain how to distill the unique abnormal visual representation of I and how to match the positive semantic features from the case report R_d.

2.2.1 Visual Contrastive Attention Module

The proposed VCAM aims to extract the unique abnormal regions of the input image I based on the calculated contrastive feature V_c.

Given the contrastive feature V_c, VCAM uses it to reconstruct an attention feature as follows:

$$V_c^a = FFN(MHA(V_c, V_c, V_c)). \quad (14)$$

In V_c, the unique abnormal regions (V_c^{blue}) of I are enhanced. Thus through the attention module, the generated visual contrastive attention feature V_c^a is able to more effectively represent the unique abnormal regions of the input image.

VCAM attends to represent the unique abnormal regions in the input images. However, there are still some important abnormal participants, such as the mixed region V_c^{mix} in Figure 3, ignored by VCAM. In addition, it is equally meaningful to match both normal and abnormal positive semantic words in

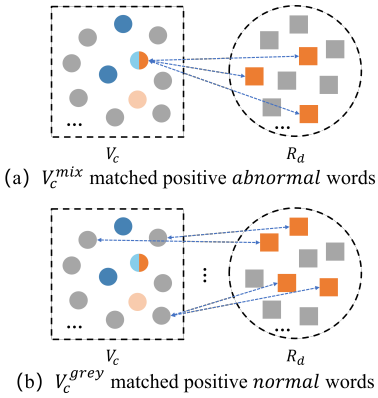

(a) V_c^{mix} matched positive *abnormal* words

(b) V_c^{grey} matched positive *normal* words

Figure 4: The cross-modal matching for the visual contrastive feature V_c and semantic feature of the retrieved report R_d. The mixed circles in V_c is the same abnormal regions of image I and I_d. The orange squares are positive semantic features.

the retrieved case report. Thus, the following module is proposed to solve these problems.

2.2.2 Cross-modal Attention Module

We propose CAM to align the contrastive feature V_c with the retrieved report R_d for exploring the positive semantic information from the retrieved case report R_d.

As shown in Figure 4, for better illustrating the ability of CAM, we hypothesize splitting case report features into positive and negative parts denoted by orange blocks and gray blocks in $R_d = \{R_d^{orange}, R_d^{gray}\}$. Given the contrastive features V_c and the case reports R_d, CAM feeds them into the following layers:

$$Cr^a = FFN(MHA(V_c, R_d, R_d)). \quad (15)$$

The negative words R_d^{gray} are unmatched because they are corresponding to the unique abnormal regions ($V_c{}^{blue}$ and $V_c{}^{orange}$) which has been enhanced or weakened in V_c. The positive words R_d^{orange} contain two statuses as follows:

- As shown in Figure 4 (a), the first status is that the abnormal positive words are corresponding to the same abnormal parts ($V_c{}^{mix}$) of I and I_d, which are retained in V_c and can be matched with CAM.
- As shown in Figure 4 (b), the second status is that the normal positive words are corresponding to the normal visual features ($V_c{}^{grey}$) which also retained in V_c and can be matched with CAM.

In short, CAM extracts both normal and abnormal positive semantic words from the retrieved semantic report R_d by building their cross-modal interactions with the visual contrastive feature V_c.

VCAM and CAM complement each other in representing visual and semantic information as well as settling the problems caused by data biases.

2.3 Parallel Attention Module

As shown in Figure 2, for each time step t, the decoder layer first takes the embedded previous words $y_{1:t-1}$ as the input of the MHA layers:

$$h_t = MHA(y_{1:t-1}, y_{1:t-1}, y_{1:t-1}), \quad (16)$$

Then, the obtained visual attention feature V^a, visual contrastive attention feature V_c^a and cross-modal attention feature Cr^a are fed into three MHA layers separately, which calculates the attention features in parallel:

$$h_t'^1 = MHA(h_t, V^a, V^a), \quad (17)$$
$$h_t'^2 = MHA(h_t, V_c^a, V_c^a,), \quad (18)$$
$$h_t'^3 = MHA(h_t, Cr^a, Cr^a), \quad (19)$$

The parallel operation would further enhance the decoding features for the three encoded attention feature. Thereafter, the three attention features are gathered as follows:

$$h_t' = h_t'^1 W_t^1 + h_t'^2 W_t^2 + h_t'^3 W_t^3, \quad (20)$$

where W_t^1, W_t^2 and W_t^3 are learned parameters.

Finally, the h_t' goes through a FFN layer and a linear layer followed by softmax activation to predict the current word:

$$\tilde{y_t} \leftarrow p_t = softmax(FFN(h_t')W_y + b_y). \quad (21)$$

where $\tilde{y_t}$ is the predicted word at current timestep, and W_y, b_y are learnable weight and bias.

3 Experiments

3.1 Datasets

We conduct experiments on two datasets to evaluate the effectiveness of our proposed model.

3.1.1 IU X-Ray

Indiana University Chest X-Ray Collection (IU X-Ray) (Demner-Fushman et al., 2016) is a widely used public radiography dataset which totally contains 7,470 Chest X-ray images and 3,955 reports. Following (Chen et al., 2020), we randomly split the dataset into train/validation/test sets by 7:1:2.

3.1.2 MIMIC-CXR

The new released MIMIC-CXR (Johnson et al., 2019) dataset is the largest dataset so far. It con-

Table 1: Comparison of the proposed Cross-modal Contrastive model with other state-of-the-art methods on the IU X-Ray dataset and MIMIC-CXR dataset. BLEU-n denotes the BLEU scores using n-grams.

Datasets	Methods	BLEU-1	BLEU-2	BLEU-3	BLEU-4	METEOR	ROUGE-L
IU X-Ray	CNN-RNN (Vinyals et al., 2015)	0.216	0.124	0.087	0.066	-	0.306
	AdaAtt (Lu et al., 2017)	0.220	0.127	0.089	0.068	-	0.308
	Att2in (Rennie et al., 2017)	0.224	0.129	0.089	0.068		0.308
	HRNN (Krause et al., 2017)	0.439	0.281	0.190	0.133	-	0.342
	CoAtt (Jing et al., 2018)	0.455	0.288	0.205	0.154	-	0.369
	HRGR-Agent (Li et al., 2018a)	0.438	0.298	0.208	0.151	-	0.322
	CMAS-RL (Jing et al., 2019)	0.464	0.301	0.210	0.154	-	0.362
	KERP (Li et al., 2019)	0.482	0.325	0.226	0.162	-	0.339
	R2Gen (Chen et al., 2020)	0.470	0.304	0.219	0.165	0.187	0.371
	CMN (Chen et al., 2021)	0.475	0.309	0.222	0.170	0.191	0.375
	CA (Liu et al., 2021b)	0.492	0.314	0.222	0.169	0.193	0.381
	Transformer (Vaswani et al., 2017)	0.396	0.254	0.179	0.135	0.164	0.342
	CMCA	**0.497**	**0.349**	**0.268**	**0.215**	**0.209**	**0.392**
MIMIC-CXR	CNN-RNN (Vinyals et al., 2015)	0.299	0.184	0.121	0.084	0.124	0.263
	AdaAtt (Lu et al., 2017)	0.299	0.185	0.124	0.088	0.118	0.266
	Att2in (Rennie et al., 2017)	0.325	0.203	0.136	0.096	0.134	0.276
	Top-Down (Anderson et al., 2018)	0.317	0.195	0.130	0.092	0.128	0.267
	R2Gen (Chen et al., 2020)	0.353	0.218	0.145	0.103	0.142	0.277
	CMN (Chen et al., 2021)	0.353	0.218	0.148	0.106	0.142	0.278
	CA (Liu et al., 2021b)	0.350	0.219	0.152	0.109	**0.151**	0.283
	Transformer (Vaswani et al., 2017)	0.314	0.192	0 .127	0.090	0.125	0.265
	CMCA	0.**360**	**0.227**	**0.156**	**0.117**	0.148	**0.287**

tains 473,057 Chest X-ray images and 206,563 reports. For fair comparison, we adopt the official split with 368,960 images and 222,758 reports for training, 2,991 images and 1,808 reports for validation, 5,159 images and 3,269 reports for testing.

For both datasets, we adopt and tokenize the *findings* section which has long sentences as the target reports and convert words into lower-cases.

3.2 Experimental Settings

3.2.1 Evaluation Metrics

We evaluate our proposed approach on the widely used metrics: BLEU (Papineni et al., 2002), ROUGE-L (Lin, 2004) and METEOR (Banerjee and Lavie, 2005). The metric scores are calculated by the standard image caption evaluation tool [1].

For clinical efficacy estimate, we employ the CheXpert labeling tool [2] proposed in (Irvin et al., 2019) to label our generated reports and the ground-truth reports in 14 different categories related to thoracic diseases and support devices. *Precision*, *Recall* and *F1* are taken as the evaluation metrics.

3.2.2 Implementation Details

We use the DenseNet-121 (Huang et al., 2017) pre-tained on ImageNet (Deng et al., 2009) and fine-tuned on CheXpert (Irvin et al., 2019) dataset to extract visual features of images in this paper. The dimension of each extracted visual feature map is set to 1024, and we then converted it to 512. In addition, following the previous works (Chen et al., 2020), we use the frontal and lateral view images as input and concatenate the features of two view images together for IU X-Ray dataset, and use single image as input for MIMIC-CXR dataset. For the proposed method, the dimension of our multi-head attention model is set to 512, and the number of heads is set to 8. And we set the number of layers to 3 for all modules. Moreover, the model is trained for 100 epochs under ADAM optimizer (Kingma and Ba, 2014). We set the initial learning rate to 1e-4 decaying by 0.99 per epoch. The beam size is set to 3 and we select top 5 similar cases for each input image.

Table 2: The comparison of the clinical efficacy metrics on MIMIC-CXR dataset, which measures the Precision, Recall and F1-score of the clinical abnormalities for the generated reports.

Methods	CE Metrics		
	Precision	Recall	F1-score
CNN-RNN (Vinyals et al., 2015)	0.249	0.203	0.204
AdaAtt (Lu et al., 2017)	0.268	0.186	0.181
Att2in (Rennie et al., 2017)	0.322	0.239	0.249
Top-Down (Anderson et al., 2018)	0.320	0.231	0.238
R2Gen (Chen et al., 2020)	0.333	0.273	0.276
CMN (Chen et al., 2021)	0.334	0.275	0.278
CA (Liu et al., 2021b)	0.352	**0.298**	0.303
CMCA	**0.444**	0.297	**0.356**

[1] https://github.com/tylin/coco-caption

[2] https://github.com/stanfordmlgroup/chexpert-labeler

Table 3: The comparison on the IU X-Ray dataset of the Baseline model (Vaswani et al., 2017) with the different components of our proposed method: VCAM, CAM, and PAM.

Methods	VCAM	CAM	PAM	BLEU-1	BLEU-2	BLEU-3	BLEU-4	METEOR	ROUGE-L
Baseline				0.396	0.254	0.179	0.135	0.164	0.342
(a)	✓			0.481	0.328	0.242	0.187	0.202	0.380
(b)		✓		0.470	0.305	0.215	0.160	0.186	0.384
(c)	✓	✓		0.484	0.335	0.248	0.194	0.203	0.381
CMCA	✓	✓	✓	**0.497**	**0.349**	**0.268**	**0.215**	**0.209**	**0.392**

3.3 Results

We take Transformer (Vaswani et al., 2017) with 3 layers for both encoder and decoder modules as the Baseline model. In addition, we compare our approach with the state-of-the-art medical report generation models, i.e., CoAtt (Jing et al., 2018), HRGR-Agent (Li et al., 2018a), CMAS-RL (Jing et al., 2019), KERP (Li et al., 2019), R2Gen (Chen et al., 2020), CMN (Chen et al., 2021) and CA (Liu et al., 2021b). And we also adopt image captioning methods, i.e., CNN-RNN (Vinyals et al., 2015), AdaAtt (Lu et al., 2017), Att2in (Rennie et al., 2017), Top-Down (Anderson et al., 2018), and the model designed for long sentence generation task: HRNN (Krause et al., 2017). We directly quote the results from the original papers for the comparison methods.

As shown in Table 1, our CMCA outperforms on almost all metrics on both datasets, which validates our hypothesis that historical similar cases can significantly assist medical report generation task, and CMCA is able to exploit useful visual and semantic information from similar cases for generating more accurate reports. In addition, the clinical efficacy metrics in Table 2 show that CMCA outperforms the state-of-the-art methods on almost all metrics especially on *precision* and *F1-score*, which indicates that more abnormal findings identified by CMCA are exact and our model greatly boosts the comprehensive performance on clinical efficacy.

4 Analysis

4.1 Quantitative Analysis

4.1.1 Effect of the Visual Contrastive Attention Module

VCAM is used to calculate the visual contrastive attention feature, which makes the model focus on the unique abnormal regions of the input image in the contrastive feature. Comparing with Baseline and (a) in Table 3, we can find that VCAM boosts the performance of Baseline model on all evaluation metrics. More encouragingly, comparing with the state-of-the-art methods in Table 1, VCAM achieves comparable performance on most of the metrics. We hypothesize that these performance gains may due to the contrastive feature which enhances the unique abnormal regions of the input image, and the following VCAM makes the model focus on the abnormal regions.

4.1.2 Effect of the Cross-modal Attention Module

To make full use of the semantic information of the retrieved case, we propose to align the contrastive feature with the retrieved report to match the positive words by CAM. Comparing the Baseline and (b) in Table 3, we can find that CAM also greatly improve the performance of baseline model on all evaluation metrics. In addition, it can be noted from (a) and (b) that CAM performs better on the ROUGE-L metric, which indicates that the longest common sub-sequence between the generated report and ground truth is longer. We assume this performance gain might because CAM catches more common semantic information between the input image and retrieved case.

4.1.3 Effect of the Parallel Attention Module

Based on the standard Transformer decoder layers, PAM decodes the attention features obtained from the Encoder in parallel, and then the parallel features are fused together to generate the final word. Comparing (c) with the full CMCA model in Table 3, we find that the decoder layers with parallel attention perform better than the normal ones. It verifies that the parallel attention can further extract discrimitive information from decoding the attention features separately.

In summary, both VCAM and CAM modules can boost the performance respectively. And the incorporation of the two modules can generate better results than each. Moreover, the combination of VCAM, CAM and PAM achieves the state-of-the-art performances on both neural language and clinical efficacy metrics.

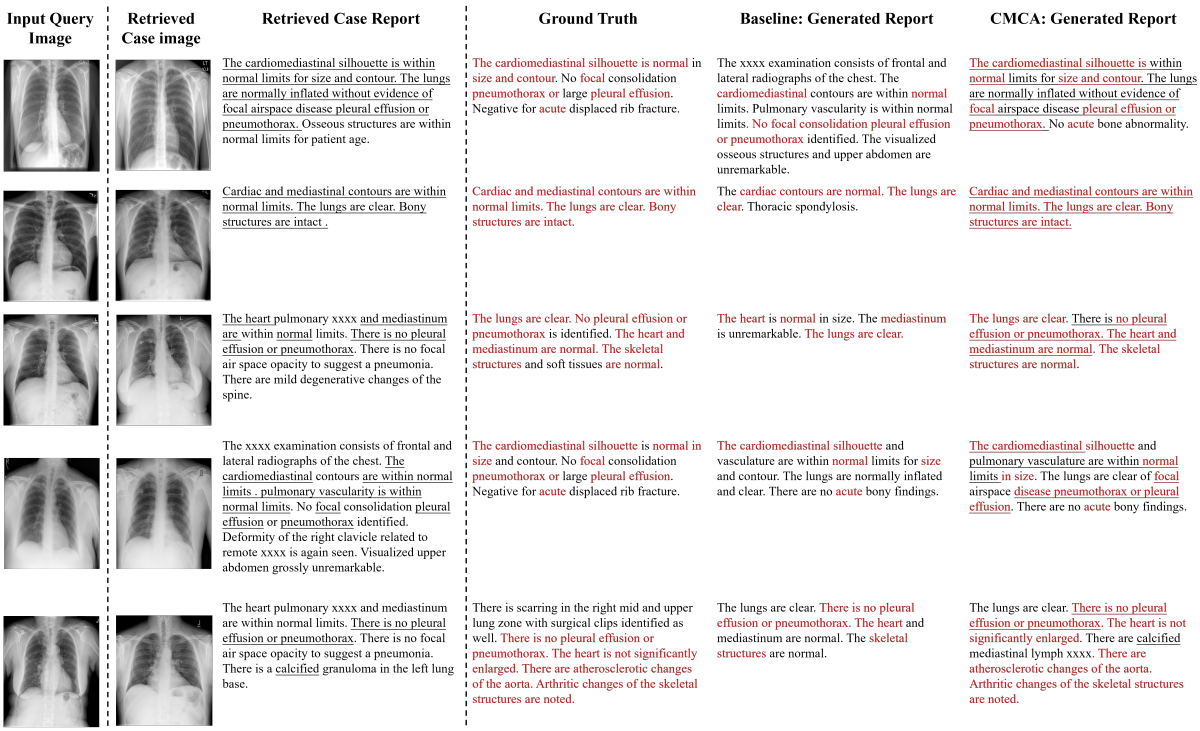

Figure 5: Visualization results of our proposed CMCA model on IU X-Ray dataset. The red fonts mark that our generated words match the ground truth, and the underlines mean that our generated words match the case reports.

Ground Truth: The lungs and pleural spaces show no acute abnormality. Heart size and pulmonary vascularity within normal limits.

Generated Report: The cardiomediastinal silhouette and vasculature are within normal limits for size and contour. The lungs are normally inflated and clear. Osseous structures are within normal limits for patient age.

Ground Truth: The cardiomediastinal silhouette is normal in size and contour. No focal consolidation pneumothorax or large pleural effusion. Negative for acute displaced rib fracture.

Generated Report: The cardiomediastinal silhouette and pulmonary vasculature are within normal limits in size. The lungs are clear of focal airspace disease pneumothorax or pleural effusion. There are no acute bony findings.

Ground Truth: The lungs are clear without evidence of focal airspace disease. There is no evidence of pneumothorax or large pleural effusion. The cardiac and mediastinal contours are within normal limits. The xxxx are unremarkable.

Generated Report: The cardiomediastinal silhouette is within normal limits for appearance. No focal areas of pulmonary consolidation. No pneumothorax . no pleural effusion. The thoracic spine appears intact. No acute displaced rib fractures.

Figure 6: Visualization of CMCA generated reports and the ground truth. The underlined texts indicate the words which correctly describe medical image information but absent in the ground truth reports.

4.2 Qualitative Analysis

In Figure 5, we visualize five examples in row to illustrate the effectiveness of our proposed model. For each row, the first column denotes the input image. The second and third columns show the retrieved case, which contains a visual similar image and the corresponding report. The fourth column is the ground truth report of the input image. The fifth column is the report generated by Baseline model and the last column is the generated report of our proposed CMCA. The red word indicates that the generated report matches the ground truth, and the underlined word means that the generated result matches the retrieved case report.

According to the underlined texts of the first two normal examples in Figure 5, our model adopts positive words from case report, and generates more accurate reports. The third and fourth examples are normal cases, but the retrieved cases are abnormal. It can be seen that CMCA employs the normal components and eliminates the abnormal ones in case reports, then generates other sentences according to the input images like *"the lungs are clear"* and *"the skeletal structures are normal"*. The fifth example and its retrieved case are both abnormal. Our result contains useful sentences from the case report, such as *"there is no pleural effusion or pneumothorax"*, and *"calcified"*. The visualization of the examples again verifies the effectiveness of our proposed CMCA model, which can select the useful semantic words from the retrieved reports.

In addition, as shown in Figure 6, the underlined texts show that our generated reports contain richer information than the ground truth. For example, in the first image, our model generates *"the cardiomediastinal silhouette and vasculature are within normal limits for size and contour"*, which is absent in the ground truth. In practice, radiologist might only write the most significant findings according to the images, while other pathological information might be neglected or incompletely recorded. Our proposed CMCA model can mitigate this problem and generate much richer reports.

5 Conclusion

In this paper, we propose a novel Cross-modal Contrastive Attention (CMCA) model to exploit the contrastive information from historical similar cases to tackle the data biases for medical report generation. CMCA contains two modules: the Visual Contrastive Attention Module that distills abnormal information of the input images, and the Cross-modal Attention Module that builds interactions of the unmatched cross-modalities. Extensive experimental results show that CMCA outperforms the state-of-the-art methods on almost all metrics. Further analyses verify the ability of CMCA in generating reports with more accurate abnormal findings and richer descriptions.

Acknowledgment

This work was supported in part by the National Natural Science Foundation of China under Grant 61906007 and 62276010, in part by the R&D Program of Beijing Municipal Education Commission under Grant KM202110005022 and KZ202210005009.

References

Peter Anderson, Xiaodong He, Chris Buehler, Damien Teney, Mark Johnson, Stephen Gould, and Lei Zhang. 2018. Bottom-up and top-down attention for image captioning and visual question answering. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 6077–6086.

Satanjeev Banerjee and Alon Lavie. 2005. Meteor: An automatic metric for mt evaluation with improved correlation with human judgments. In *Proceedings of the acl workshop on intrinsic and extrinsic evaluation measures for machine translation and/or summarization*, pages 65–72.

Zhihong Chen, Yaling Shen, Yan Song, and Xiang Wan. 2021. Cross-modal memory networks for radiology report generation. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing*, pages 5904–5914.

Zhihong Chen, Yan Song, Tsung-Hui Chang, and Xiang Wan. 2020. Generating radiology reports via memory-driven transformer. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing*, pages 1439–1449.

Dina Demner-Fushman, Marc D Kohli, Marc B Rosenman, Sonya E Shooshan, Laritza Rodriguez, Sameer Antani, George R Thoma, and Clement J McDonald. 2016. Preparing a collection of radiology examinations for distribution and retrieval. *Journal of the American Medical Informatics Association*, 23(2):304–310.

Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. 2009. Imagenet: A large-scale hierarchical image database. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 248–255.

Qingji Guan, Yaping Huang, Yawei Luo, Ping Liu, Mingliang Xu, and Yi Yang. 2021. Discriminative feature learning for thorax disease classification in chest x-ray images. *IEEE Transactions on Image Processing*, 30:2476–2487.

Qingji Guan, Yaping Huang, Zhun Zhong, Zhedong Zheng, Liang Zheng, and Yi Yang. 2020. Thorax disease classification with attention guided convolutional neural network. *Pattern Recognition Letters*, 131:38–45.

Gao Huang, Zhuang Liu, Laurens Van Der Maaten, and Kilian Q Weinberger. 2017. Densely connected convolutional networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 4700–4708.

Jeremy Irvin, Pranav Rajpurkar, Michael Ko, Yifan Yu, Silviana Ciurea-Ilcus, Chris Chute, Henrik Marklund, Behzad Haghgoo, Robyn Ball, Katie Shpanskaya, et al. 2019. Chexpert: A large chest radiograph dataset with uncertainty labels and expert comparison. In *Proceedings of the AAAI conference on artificial intelligence*.

Baoyu Jing, Zeya Wang, and Eric Xing. 2019. Show, describe and conclude: On exploiting the structure information of chest x-ray reports. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 6570–6580.

Baoyu Jing, Pengtao Xie, and Eric Xing. 2018. On the automatic generation of medical imaging reports. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics*, pages 2577–2586.

Alistair EW Johnson, Tom J Pollard, Nathaniel R Greenbaum, Matthew P Lungren, Chih-ying Deng, Yifan Peng, Zhiyong Lu, Roger G Mark, Seth J Berkowitz, and Steven Horng. 2019. Mimic-cxr-jpg, a large publicly available database of labeled chest radiographs. *arXiv preprint arXiv:1901.07042*.

Andrej Karpathy and Li Fei-Fei. 2015. Deep visual-semantic alignments for generating image descriptions. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 3128–3137.

D. Kingma and J. Ba. 2014. Adam: A method for stochastic optimization. *3rd International Conference on Learning Representations, ICLR 2015 - Conference Track Proceedings*.

Jonathan Krause, Justin Johnson, Ranjay Krishna, and Li Fei-Fei. 2017. A hierarchical approach for generating descriptive image paragraphs. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 317–325.

Christy Y Li, Xiaodan Liang, Zhiting Hu, and Eric P Xing. 2018a. Hybrid retrieval-generation reinforced agent for medical image report generation. In *Advances in neural information processing systems*, pages 1537–1547.

Christy Y Li, Xiaodan Liang, Zhiting Hu, and Eric P Xing. 2019. Knowledge-driven encode, retrieve, paraphrase for medical image report generation. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, pages 6666–6673.

Zhe Li, Chong Wang, Mei Han, Yuan Xue, Wei Wei, Li-Jia Li, and Li Fei-Fei. 2018b. Thoracic disease identification and localization with limited supervision. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 8290–8299.

Gongbo Liang, Connor Greenwell, Yu Zhang, Xin Xing, Xiaoqin Wang, Ramakanth Kavuluru, and Nathan Jacobs. 2021. Contrastive cross-modal pre-training: A general strategy for small sample medical imaging. *IEEE Journal of Biomedical and Health Informatics*.

Chin-Yew Lin. 2004. Rouge: A package for automatic evaluation of summaries. In *Text summarization branches out*, pages 74–81.

Fenglin Liu, Xian Wu, Shen Ge, Wei Fan, and Yuexian Zou. 2021a. Exploring and distilling posterior and prior knowledge for radiology report generation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 13753–13762.

Fenglin Liu, Changchang Yin, Xian Wu, Shen Ge, Ping Zhang, and Xu Sun. 2021b. Contrastive attention for automatic chest x-ray report generation. In *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*, pages 269–280.

Jiasen Lu, Caiming Xiong, Devi Parikh, and Richard Socher. 2017. Knowing when to look: Adaptive attention via a visual sentinel for image captioning. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 375–383.

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. Bleu: a method for automatic evaluation of machine translation. In *Proceedings of the 40th annual meeting of the Association for Computational Linguistics*, pages 311–318.

José Ramos, Thessa TJP Kockelkorn, Isabel Ramos, Rui Ramos, Jan Grutters, Max A Viergever, Bram van Ginneken, and Aurélio Campilho. 2014. Content-based image retrieval by metric learning from radiology reports: application to interstitial lung diseases. *IEEE Journal of Biomedical and Health Informatics*, 20(1):281–292.

Steven J Rennie, Etienne Marcheret, Youssef Mroueh, Jerret Ross, and Vaibhava Goel. 2017. Self-critical sequence training for image captioning. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 7008–7024.

Hoo-Chang Shin, Kirk Roberts, Le Lu, Dina Demner-Fushman, Jianhua Yao, and Ronald M Summers. 2016a. Learning to read chest x-rays: Recurrent neural cascade model for automated image annotation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2497–2506.

Hoo-Chang Shin, Kirk Roberts, Le Lu, Dina Demner-Fushman, Jianhua Yao, and Ronald M Summers. 2016b. Learning to read chest x-rays: Recurrent neural cascade model for automated image annotation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 2497–2506.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in neural information processing systems*, pages 5998–6008.

Oriol Vinyals, Alexander Toshev, Samy Bengio, and Dumitru Erhan. 2015. Show and tell: A neural image caption generator. In *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pages 3156 – 3164.

Xiaosong Wang, Yifan Peng, Le Lu, Zhiyong Lu, and Ronald M Summers. 2018. Tienet: Text-image embedding network for common thorax disease classification and reporting in chest x-rays. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 9049–9058.

Xing Xu, Tan Wang, Yang Yang, Lin Zuo, Fumin Shen, and Heng Tao Shen. 2020. Cross-modal attention with semantic consistence for image-text matching. *IEEE Transactions on Neural Networks and Learning Systems*, 31(12):5412–5425.

Yuan Xue, Tao Xu, L Rodney Long, Zhiyun Xue, Sameer Antani, George R Thoma, and Xiaolei Huang. 2018. Multimodal recurrent model with attention for automated radiology report generation. In *International Conference on Medical Image Computing and Computer-Assisted Intervention*, pages 457–466. Springer.

Quanzeng You, Hailin Jin, Zhaowen Wang, Chen Fang, and Jiebo Luo. 2016. Image captioning with semantic attention. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 4651–4659.

Zizhao Zhang, Yuanpu Xie, Fuyong Xing, Mason McGough, and Lin Yang. 2017. Mdnet: A semantically and visually interpretable medical image diagnosis network. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 6428–6436.

Domain-Specific NER via Retrieving Correlated Samples

Xin Zhang[1], **Yong Jiang, Xiaobin Wang, Xuming Hu**[2],
Yueheng Sun[3], **Pengjun Xie, Meishan Zhang**[4]*
[1]School of New Media and Communication, Tianjin University, China
[2]School of Software, Tsinghua University, China
[3]College of Intelligence and Computing, Tianjin University, China
[4]Institute of Computing and Intelligence, Harbin Institute of Technology (Shenzhen), China
hsinz@tju.edu.cn, hxm19@mails.tsinghua.edu.cn
jiangyong.ml@gmail.com, czwangxiaobin@foxmail.com
yhs@tju.edu.cn, xpjandy@gmail.com, zhangmeishan@hit.edu.cn

Abstract

Successful Machine Learning based Named Entity Recognition models could fail on texts from some special domains, for instance, Chinese addresses and e-commerce titles, where requires adequate background knowledge. Such texts are also difficult for human annotators. In fact, we can obtain some potentially helpful information from correlated texts, which have some common entities, to help the text understanding. Then, one can easily reason out the correct answer by referencing correlated samples. In this paper, we suggest enhancing NER models with correlated samples. We draw correlated samples by the sparse BM25 retriever from large-scale in-domain unlabeled data. To explicitly simulate the human reasoning process, we perform a training-free entity type calibrating by majority voting. To capture correlation features in the training stage, we suggest to model correlated samples by the transformer-based multi-instance cross-encoder. Empirical results on datasets of the above two domains show the efficacy of our methods.

1 Introduction

Named Entity Recognition (NER), which first locates entity positions and then labels their types sequentially, is a fundamental topic in both academia and industry (Li et al., 2022). Normal NER models consider the input samples to be independent of each other, learning the common intra-instance patterns and making predictions in a sequential way. This paradigm has shown surprising successes in decades, especially with the help of emerging deep learning (Shang et al., 2018; Zhang et al., 2018b; Liu et al., 2019; Luo et al., 2020; Lison et al., 2020; Fang et al., 2021; Meng et al., 2021).

However, learned models will fail at some hard cases, which would be inevitably encountered in real scenarios (Li et al., 2019; Ding et al., 2019). Figure 1 shows an example of the Chinese address domain. This kind of bad cases can not be easily solved by annotating more relevant training data[1]. For human annotators, this case is ambiguous as well if no extra information is given, for instance, we can not distinguish the type of "吉林 (Jilin)" without affixes "省 (Province)" or "市 (City)". This demonstrates that obtaining background knowledge and information is crucial to the text understanding.

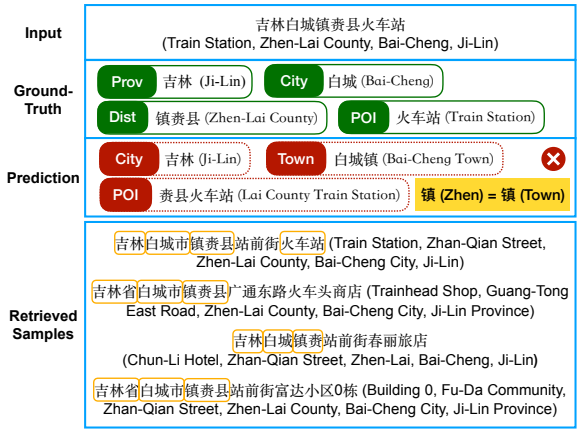

Figure 1: An address example with retrieved texts. The model incorrectly predicted "白城镇 (Baicheng Town)" and "赉县火车站 (Lai County Train Station)" because they match common patterns, i.e. "XX镇" (Xx Town) and "X县火车站" (X County Train Station). "吉林" is ambiguous, which is both a province and a city.

Learning from correlated or nested data is mainly studied in Machine Learning and Computer Vision (Dundar et al., 2007; Choi and Won, 2019; Choi et al., 2021). Images in sub-groups naturally show a high degree of correlation on both features and labels, and come with nested structures (Dundar et al., 2007; Choi and Won, 2019), such as different

*Corresponding author.

[1]Because this pattern is indeed correct in most cases. This problem also exists in models with internal larger datasets.

Proceedings of the 29th International Conference on Computational Linguistics, pages 2398–2404
October 12–17, 2022.

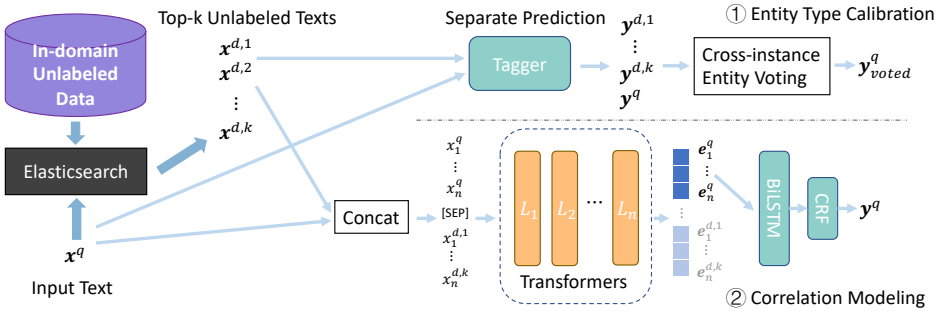

Figure 2: The overview of our suggested methods.

regions of interest could be drawn from the same objects. In the address and e-commerce domain, texts are also highly correlated in nature. For example, two addresses may belong to the same city or refer to the same location, e-commerce product titles could come from the same brand, or they are just the same product. In Figure 1, with correlated texts, annotators can infer that the "吉林 (Jilin)" and "白城 (Baicheng)" are short forms of "吉林省 (Jilin Province)" and "白城市 (Baicheng City)". Hence, we argue that correlated samples could offer sufficient disambiguation information for NER models as well. Such kind of inductive bias is seldom considered in previous NLP studies.

In this work, we propose to enhance NER models by modeling and inferencing with the correlated samples. We first draw the correlated samples from in-domain large-scale unlabeled data by the retrieval engine (Elasticsearch, 2022).[2] Then, we suggest two methods: (1) we perform an entity type calibrating by parallelly predicting the input text and all retrieved samples by the off-the-shelf NER model, and then aggregating the final labels by majority voting; (2) we propose to model the correlations by transformers via multi-instance cross-encoders to enhance the NER feature vectors.

To evaluate our methods, we conduct experiments on two open-access datasets (Inc., 2022; Ding et al., 2019) of the aforementioned two domains. We implement our methods based on a strong BiLSTM-CRF model with NEZHA (Wei et al., 2019) representation. Empirical results show that our methods outperform all baselines, and achieve promising results in the simulated low-resource setting. Finally, we present several analyses to understand our methods comprehensively.

[2]Recently, Wang et al. (2021) and Geng et al. (2022) studied retrieving external contexts from Google or Baidu for standard NER datasets, which is quite different from our idea of modeling correlated samples for specific domains.

2 Approach

2.1 Unlabeled Data Retrieval

In the address (resp. e-commerce) domain, some texts naturally possess entity co-reference relations, for instance, they may belong to the same city (resp. brand) or represent the same location (resp. product). We call such texts, which usually have entities with the same semantic but different expressions, **correlated samples**. Since these texts are highly structured and of limited vocabulary, showing a high degree of lexical overlap. We could draw correlated samples for a given text by taking it as a query and retrieving the domain-specific database with text similarity measurements.

We implement an efficient BM25 (Robertson and Walker, 1994) retriever by an off-the-shelf retrieval engine (Elasticsearch, 2022). For a cleaned large-scale in-domain unlabeled corpus, we create the Elasticsearch index by the build-in standard analyzer. Then, we can retrieve top-K samples by BM25 scores of an input text in nearly real-time.

2.2 Entity Type Calibrating

As shown in Figure 1, correlated sample can help the entity disambiguation. If this kind of entity appears in correlated samples, human annotators can decide its type by referring to answers of correlated samples. For NER models, we suggest achieving this process by entity-level (or span-level) majority voting. Concretely, we first use a model (e.g., baseline) to extract entities of the input text and each correlated sample parallelly, and then re-assign labels of shared entities by majority voting.

2.3 Correlation Modeling

To further capture sample correlations in the training time, we suggest modeling correlated samples by the cross-encoder (Reimers and Gurevych, 2019), letting transformers learn complex corre-

Method	Micro			Macro		
	P	R	F1	P	R	F1
Chinese Address						
Human	93.04	92.01	92.52	87.83	84.52	86.14
BC	85.56	83.90	84.72	82.20	76.48	79.24
NEZHA-BC	91.29	90.62	90.95	86.41	84.68	85.53
Entity-Voting†	91.67	91.00	91.34	86.70	84.83	85.76
Cross-Encoder†	**92.41**	**91.95**	**92.18**	**87.25**	**85.71**	**86.48**
Self-Training	91.57	91.02	91.29	86.65	85.37	86.01
Biaffine	91.35	90.25	90.80	86.32	84.59	85.45
Seq2set	89.43	87.69	88.55	83.89	80.12	81.96
Locate&Label	90.28	87.76	89.00	85.95	82.29	84.08
PIQN	90.27	87.83	89.03	86.04	80.28	83.06
E-commerce						
BC	65.31	62.54	63.90	58.88	50.38	54.30
NEZHA-BC	82.73	83.23	82.98	79.35	78.04	78.69
Entity-Voting†	82.83	83.33	83.08	79.56	78.19	78.87
Cross-Encoder†	**83.49**	83.74	**83.61**	**81.45**	**79.34**	**80.38**
Self-Training	81.51	**85.25**	83.34	78.89	79.29	79.09
Biaffine	81.91	84.06	82.97	80.14	79.05	79.59
Seq2set	82.77	81.65	82.21	81.39	76.44	78.84
Locate&Label	80.43	83.21	81.80	76.63	78.22	77.42
PIQN	83.43	82.54	82.98	81.23	75.60	78.31
BERT-CLS (2019)	77.06	80.65	78.81	-	-	-
MRC-NER (2020)	79.47	78.30	78.88	-	-	-
CoFEE-BERT (2020)	79.13	80.34	79.73	-	-	-
CoFEE-MRC (2020)	80.26	78.88	79.56	-	-	-

Table 1: Main results. † means statistically significant.

lation patterns among samples. Specifically, we concatenate the input text with retrieved samples by the separator (i.e., [SEP]), and then encode them by pretrained language models. Finally, only the contextual embeddings of the input text are fed into the NER tagger (here BiLSTM-CRF). With this simple strategy, NER models could benefit from the contrastive view between multiple correlated samples and understand the query instance better.

3 Experiments

3.1 Settings

Datasets. For the **Chinese Address** domain, we use the recently published dataset from CCKS competition (Inc., 2022). It is annotated by 21 classes of address elements and contains 8856, 1970, 4000 addresses for train, dev, and test sets. For the **E-commerce** domain, we use the dataset released by Ding et al. (2019). It is collected from e-commerce product titles and annotated by PROD (product) and BRAN (brand) types. It has 3983, 499, 498 sentences[3] for train, dev, and test sets. For our **retrieval-based** methods, we process and index our internal in-domain unlabeled data with Elasticsearch, obtaining 400M and 600M samples for the address and e-commerce domain, respectively.

Evaluation. We employ entity-level exact precision, recall, and F1-measure and report both micro and macro aggregations. All experiments of the same setting are conducted by 8 different random seeds. We test the best model of the devset, and the average scores are reported. We regard a result as statistically significant when the p-value is below 0.05 by the paired t-test with baseline NEZHA-BC.

Implementation. We choose the BiLSTM-CRF (Lample et al., 2016) to achieve NER task, and use NEZHA-base (Wei et al., 2019) as the embedding module. The BiLSTM hidden size is set to 384 for each direction. We apply the dropout (Srivastava et al., 2014) with probabilities 0.5 and 0.2 to NEZHA embeddings for address and e-commerce, and 0.2 to BiLSTM features. We set the batch size to 32 and use the AdamW (Loshchilov and Hutter, 2017) optimizer with a constant lr 1e-3 and 1e-5 to update BiLSTM-CRF and NEZHA parameters.

For the entity type calibrating, we use the top 100 and 50 retrieved samples for address and e-commerce, respectively. For the correlating modeling, we limit the max sample number to 12 and the max sequence length to 256.

Baselines. We denote the BiLSTM-CRF with random character embedding (resp. NEZHA) by **BC** (resp. **NEZHA-BC**). We implement several state-of-the-art methods, i.e., **Biaffine** (Yu et al., 2020), **Seq2set** (Tan et al., 2021), **Locate&Label** (Shen et al., 2021), **PIQN** (Shen et al., 2022). We also implement **Self-Training** based on NEZHA-BC and the unlabeled data of the same size as our cross-encoder. We include e-commerce results from Mengge et al. (2020) for comparison.

3.2 Main Results

As shown in Table 1, our training-free calibrating method consistently outperforms our implemented baselines on both datasets, which verifies our intuition that modeling the correlation between samples is important in processing domain-specific texts. By leveraging the retrieved samples in the training stage (Cross-Encoder), our approaches gain a significant performance boost. This indicates that these retrieved samples not only provide extended entity information (such as 白城⟶白城市), but also supply sufficient disambiguate sig-

[3]We remove a few sentences that are particularly long and do not contain entities.

	Method	100%	50%	20%	10%	5%	3%
Chinese Address							
Micro	NEZHA-BC	90.95	90.04	88.79	87.57	86.56	84.69
	Cross-encoder	92.18 ↑1.23	91.56 ↑1.52	90.33 ↑1.54	89.13 ↑1.56	88.61 ↑2.05	86.82 ↑2.13
Macro	NEZHA-BC	85.53	84.22	82.14	78.08	75.99	72.78
	Cross-encoder	86.48 ↑0.95	85.36 ↑1.14	83.43 ↑1.29	79.66 ↑1.58	78.31 ↑2.32	75.41 ↑2.63
E-commerce							
Micro	NEZHA-BC	82.98	81.54	79.52	77.89	75.80	73.86
	Cross-encoder	83.61 ↑0.63	82.21 ↑0.67	80.29 ↑0.77	78.99 ↑1.10	77.18 ↑1.38	74.60 ↑0.74
Macro	NEZHA-BC	78.69	77.03	75.17	72.86	69.63	67.11
	Cross-encoder	80.21 ↑1.52	78.21 ↑1.18	76.47 ↑1.30	74.28 ↑1.42	71.47 ↑1.84	68.03 ↑0.92

Table 2: Test F1 scores at various low-resource settings.

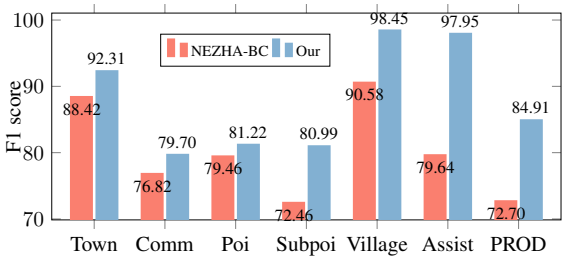

Figure 3: Test F1 of some hard entity types from main results, where their scores are less than the overall value.

nals for entity understanding (such as 镇赉县火车站 v.s 镇赉县站前街火车站). When compared with other recent state-of-the-art NER methods (Biaffine, Seq2set, Locate&Label, and PIQN), our approaches outperform them by a large margin. It is worth noting that our model outperforms the self-training (whose unlabeled corpus is in the same scale of samples we modeled), demonstrating that the correlation modeling is more effective. Then we plot detailed scores by categories whose F1 score is less than the overall F1 in Figure 3. All of these difficult categories are significantly improved, showing that the correlated samples are helpful.

Other Results. For the Chinese Address dataset, we also report the performance of human annotators without extra information provided. Notably, our approaches achieve comparable performance with humans, which empirically verifies that modeling text correlation with the retrieval perspective might have the possibility to simulate human expert annotations. For the E-commerce dataset, we also report other published results. Our NEZHA-BC is comparable with all the baseline implementations.

#Address	400M	40M	10M	4M	400k	100k
Micro F1	92.18	92.07	91.89	91.53	91.31	91.16

Table 3: Test F1 scores of our Cross-encoder in various sizes of unlabeled data for retrieval in address domain.

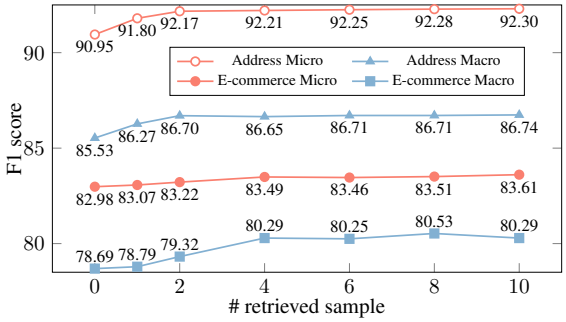

Figure 4: Test F1 scores of incooperating different correlated sample num by our cross-encoder.

3.3 Analysis

We conduct fine-grained analyses of cross-encoder.

Different Sizes of Labeled Data. Our idea essentially introduces extra in-domain data to the predictive models. Hence we can suppose that our methods will achieve larger improvements in the low-resource scenario. To verify this, we train the baseline and our cross-encoder in simulated smaller trainsets, which are sampled from the original trainset by different proportions. Table 2 demonstrates the test f1 scores of these two models in different settings. We can roughly say that the score difference increases as the sampling ratio decreases, which is in line with our intuition.

Different Sizes of Correlated Samples. In the above experiments, we limit the max sequence length of our cross-encoder to 256 for efficiency. Here we relax this constraint to investigate the influence of encoded sample num (from 0[4] to top 10 retrieved texts) in cross-encoder on both two domains. As shown in Figure 4, the performance increment is significant at the lower sample number. And adding more relatively low-ranking samples is of limited gains.

Different Sizes of Unlabeled Data. All of the previous experiments are based on the same large-scale in-domain unlabeled data, which almost reach the billion-level (400M and 600M samples for address and e-commerce, respectively). We also sample several smaller unlabeled corpus (i.e., 40M,

[4]The 0 samples cross-encoder degrade to the baseline NEZHA-BC.

Method	NEZHA-BC	Entity-Voting	Cross-Encoder
Seconds	14.73	500+	38.41

Table 4: Running times of different methods on the address domain testset, which has 4,000 texts.

10M, 4M, 400k, 100k) and re-train our cross-encoder. As shown in Table 3, with the size of unlabeled data declines, the retrieved samples are less relevant, the improvements of our model are lower. Interestingly, this experiments also could reflect the effect of **unlabeled data quality** to the performance of our cross-encoder. The higher the quality of the data, the more correlated samples can be retrieved. The behavior of low-quality unlabeled data is similar to the small size data.

Running Speed of Different Methods. Another key concern of our methods is the running speed. The entity-voting needs parallelly decode dozens of texts, and the cross-encoder will significantly enlarge the text length. We measured the running time of several methods on the testset of the address domain dataset. As demonstrated in Table 4, the entity-voting is truely slower than other methods in an order of magnitude. But the cross-encoder just took about twice as long as the baseline NEZHA-BC. This is because the most time-consuming part is the CRF, where the concatenated samples are droped before the CRF. So it can avoid the redundant decoding in the entity-voting, and has a higher running speed. Besides, the forward of pretrained language models are highly optimized.

3.4 Discussion

Retrieval-augmented models are showing state-of-the-art performance in many NLP tasks, such as Dialogue (Weston et al., 2018), Neural Machine Translation (Zhang et al., 2018a), Question Answering (Izacard and Grave, 2021), and Language Modeling (Guu et al., 2020; Yao et al., 2022; Borgeaud et al., 2022). Our work aims to model the internal correlation within sub-groups of samples. We first retrieve correlated sample groups for a given input by the off-the-shelf Elasticsearch engine. Then, we propose painlessly calibrating entity type and transformer-based correlation modeling, where the latter one is similar to Wang et al. (2021). Our recent work (Wang et al., 2022) also investigated retrieving knowledge from the Wikipedia, which can augment the context of NER inputs and shows significant improvements in SemEval-2022 Task 11 Multilingual NER.

This work could be further investigated with some more sophisticated techniques, such as example-based learning (Gao et al., 2021; Lee et al., 2022; Liu et al., 2022). Meanwhile, it also may help the NER task to extend to the low-resource and zero-shot scenarios (Meng et al., 2021; Zhang et al., 2021; Hu et al., 2021; Lu et al., 2022; Hu et al., 2020).

4 Conclusion

In this work, we investigated utilizing naturally correlated samples to improve current NER models on the Chinese address and e-commerce domain. We propose to retrieve correlated samples for the given text by the BM25 and elasticsearch engine. To explore the correlations in a light way, we suggest calibrating the predicted entity types by cross-instance entity voting. To further incorporate these correlated samples into model training, we use multi-instance cross-encoders to learn more complex correlations. Empirical results show that the painless entity type calibrating improved the performance to some extent, and modeling correlations by cross-encoders achieved the state-of-the-art performance. We hope this idea could benefit the similar scenario/domains of other tasks.

We will release our code and data at github.com/izhx/NER-unlabeled-data-retrieval to facilitate future research.

Acknowledgements

We thank all reviewers for their hard work. This research is supported by grants from the National Natural Science Foundation of China (No. 62176180).

Ethical Statement

All texts are anonymized.

References

Sebastian Borgeaud, Arthur Mensch, Jordan Hoffmann, Trevor Cai, Eliza Rutherford, Katie Millican, George van den Driessche, Jean-Baptiste Lespiau, Bogdan Damoc, Aidan Clark, Diego de Las Casas, Aurelia Guy, Jacob Menick, Roman Ring, Tom Hennigan, Saffron Huang, Loren Maggiore, Chris Jones, Albin Cassirer, Andy Brock, Michela Paganini, Geoffrey Irving, Oriol Vinyals, Simon Osindero, Karen Simonyan, Jack W. Rae, Erich Elsen, and Laurent Sifre. 2022. Improving language models by retrieving from trillions of tokens. In *Proc. of the ICML*, volume 162 of *Proceedings of Machine Learning Research*, pages 2206–2240. PMLR.

Youngwon Choi, Sungdong Lee, and Joong-Ho Won. 2021. Learning from nested data with ornstein auto-encoders. In *Proc. of the ICML*, volume 139 of *Proceedings of Machine Learning Research*, pages 1943–1952. PMLR.

Youngwon Choi and Joong-Ho Won. 2019. Ornstein auto-encoders. In *Proc. of the IJCAI*, pages 2172–2178. ijcai.org.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proc. of the NAACL*, pages 4171–4186, Minneapolis, Minnesota. Association for Computational Linguistics.

Ruixue Ding, Pengjun Xie, Xiaoyan Zhang, Wei Lu, Linlin Li, and Luo Si. 2019. A neural multi-digraph model for Chinese NER with gazetteers. In *Proc.of the ACL*, pages 1462–1467, Florence, Italy. Association for Computational Linguistics.

Murat Dundar, Balaji Krishnapuram, Jinbo Bi, and R. Bharat Rao. 2007. Learning classifiers when the training data is not IID. In *Proc. of the IJCAI*, pages 756–761.

Elasticsearch. 2022. Elasticsearch: The official distributed search & analytics engine.

Zheng Fang, Yanan Cao, Tai Li, Ruipeng Jia, Fang Fang, Yanmin Shang, and Yuhai Lu. 2021. TEBNER: Domain specific named entity recognition with type expanded boundary-aware network. In *Proc. of the EMNLP*, pages 198–207, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Tianyu Gao, Adam Fisch, and Danqi Chen. 2021. Making pre-trained language models better few-shot learners. In *Proc. of the ACL-IJCNLP*, pages 3816–3830, Online. Association for Computational Linguistics.

Zhichao Geng, Hang Yan, Zhangyue Yin, Chenxin An, and Xipeng Qiu. 2022. TURNER: the uncertainty-based retrieval framework for chinese NER. *CoRR*, abs/2202.09022.

Kelvin Guu, Kenton Lee, Zora Tung, Panupong Pasupat, and Ming-Wei Chang. 2020. Retrieval augmented language model pre-training. In *Proc. of the ICML*, volume 119 of *Proceedings of Machine Learning Research*, pages 3929–3938. PMLR.

Xuming Hu, Lijie Wen, Yusong Xu, Chenwei Zhang, and Philip Yu. 2020. SelfORE: Self-supervised relational feature learning for open relation extraction. In *Proc. of the EMNLP*, pages 3673–3682, Online. Association for Computational Linguistics.

Xuming Hu, Chenwei Zhang, Yawen Yang, Xiaohe Li, Li Lin, Lijie Wen, and Philip S. Yu. 2021. Gradient imitation reinforcement learning for low resource relation extraction. In *Proc. of the EMNLP*, pages 2737–2746, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Alibaba Inc. 2022. Ccks 2021 chinese address element parsing dataset.

Gautier Izacard and Edouard Grave. 2021. Leveraging passage retrieval with generative models for open domain question answering. In *Proc. of the EACL*, pages 874–880, Online. Association for Computational Linguistics.

Guillaume Lample, Miguel Ballesteros, Sandeep Subramanian, Kazuya Kawakami, and Chris Dyer. 2016. Neural architectures for named entity recognition. In *Proc. of the NAACL*, pages 260–270, San Diego, California. Association for Computational Linguistics.

Dong-Ho Lee, Akshen Kadakia, Kangmin Tan, Mahak Agarwal, Xinyu Feng, Takashi Shibuya, Ryosuke Mitani, Toshiyuki Sekiya, Jay Pujara, and Xiang Ren. 2022. Good examples make a faster learner: Simple demonstration-based learning for low-resource NER. In *Proc. of the ACL*, pages 2687–2700, Dublin, Ireland. Association for Computational Linguistics.

Hao Li, Wei Lu, Pengjun Xie, and Linlin Li. 2019. Neural Chinese address parsing. In *Proc. of the NAACL*, pages 3421–3431, Minneapolis, Minnesota. Association for Computational Linguistics.

Jing Li, Aixin Sun, Jianglei Han, and Chenliang Li. 2022. A survey on deep learning for named entity recognition. *IEEE Trans. Knowl. Data Eng.*, 34(1):50–70.

Xiaoya Li, Jingrong Feng, Yuxian Meng, Qinghong Han, Fei Wu, and Jiwei Li. 2020. A unified MRC framework for named entity recognition. In *Proc. of the ACL*, pages 5849–5859, Online. Association for Computational Linguistics.

Pierre Lison, Jeremy Barnes, Aliaksandr Hubin, and Samia Touileb. 2020. Named entity recognition without labelled data: A weak supervision approach. In *Proc. of the ACL*, pages 1518–1533, Online. Association for Computational Linguistics.

Shuliang Liu, Xuming Hu, Chenwei Zhang, Shu'ang Li, Lijie Wen, and Philip Yu. 2022. HiURE: Hierarchical exemplar contrastive learning for unsupervised relation extraction. In *Proc. of the NAACL*, pages 5970–5980, Seattle, United States. Association for Computational Linguistics.

Tianyu Liu, Jin-Ge Yao, and Chin-Yew Lin. 2019. Towards improving neural named entity recognition with gazetteers. In *Proc. of the ACL*, pages 5301–5307, Florence, Italy. Association for Computational Linguistics.

Ilya Loshchilov and Frank Hutter. 2017. Fixing weight decay regularization in adam. *CoRR*, abs/1711.05101.

Yaojie Lu, Qing Liu, Dai Dai, Xinyan Xiao, Hongyu Lin, Xianpei Han, Le Sun, and Hua Wu. 2022. Unified structure generation for universal information extraction. In *Proc. of the ACL*, pages 5755–5772, Dublin, Ireland. Association for Computational Linguistics.

Ying Luo, Fengshun Xiao, and Hai Zhao. 2020. Hierarchical contextualized representation for named entity recognition. In *Proc. of the AAAI*, pages 8441–8448. AAAI Press.

Yu Meng, Yunyi Zhang, Jiaxin Huang, Xuan Wang, Yu Zhang, Heng Ji, and Jiawei Han. 2021. Distantly-supervised named entity recognition with noise-robust learning and language model augmented self-training. In *Proc. of the EMNLP*, pages 10367–10378, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Xue Mengge, Bowen Yu, Zhenyu Zhang, Tingwen Liu, Yue Zhang, and Bin Wang. 2020. Coarse-to-Fine Pre-training for Named Entity Recognition. In *Proc. of the EMNLP*, pages 6345–6354, Online. Association for Computational Linguistics.

Nils Reimers and Iryna Gurevych. 2019. Sentence-BERT: Sentence embeddings using Siamese BERT-networks. In *Proc. of the EMNLP-IJCNLP*, pages 3982–3992, Hong Kong, China. Association for Computational Linguistics.

Stephen E. Robertson and Steve Walker. 1994. Some simple effective approximations to the 2-poisson model for probabilistic weighted retrieval. In *Proc. of the SIGIR*, pages 232–241. ACM/Springer.

Jingbo Shang, Liyuan Liu, Xiaotao Gu, Xiang Ren, Teng Ren, and Jiawei Han. 2018. Learning named entity tagger using domain-specific dictionary. In *Proc. of the EMNLP*, pages 2054–2064, Brussels, Belgium. Association for Computational Linguistics.

Yongliang Shen, Xinyin Ma, Zeqi Tan, Shuai Zhang, Wen Wang, and Weiming Lu. 2021. Locate and label: A two-stage identifier for nested named entity recognition. In *Proc. of the ACL*, pages 2782–2794, Online. Association for Computational Linguistics.

Yongliang Shen, Xiaobin Wang, Zeqi Tan, Guangwei Xu, Pengjun Xie, Fei Huang, Weiming Lu, and Yueting Zhuang. 2022. Parallel instance query network for named entity recognition. In *Proc. of the ACL*. Association for Computational Linguistics.

Nitish Srivastava, Geoffrey E. Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. 2014. Dropout: a simple way to prevent neural networks from overfitting. *J. Mach. Learn. Res.*, 15(1):1929–1958.

Zeqi Tan, Yongliang Shen, Shuai Zhang, Weiming Lu, and Yueting Zhuang. 2021. A sequence-to-set network for nested named entity recognition. In *Proc. of the IJCAI*, pages 3936–3942. ijcai.org.

Xinyu Wang, Yong Jiang, Nguyen Bach, Tao Wang, Zhongqiang Huang, Fei Huang, and Kewei Tu. 2021. Improving named entity recognition by external context retrieving and cooperative learning. In *Proc. of the ACL-IJCNLP*, pages 1800–1812, Online. Association for Computational Linguistics.

Xinyu Wang, Yongliang Shen, Jiong Cai, Tao Wang, Xiaobin Wang, Pengjun Xie, Fei Huang, Weiming Lu, Yueting Zhuang, Kewei Tu, Wei Lu, and Yong Jiang. 2022. DAMO-NLP at SemEval-2022 task 11: A knowledge-based system for multilingual named entity recognition. In *Proc. of the SemEval*, pages 1457–1468, Seattle, United States. Association for Computational Linguistics.

Junqiu Wei, Xiaozhe Ren, Xiaoguang Li, Wenyong Huang, Yi Liao, Yasheng Wang, Jiashu Lin, Xin Jiang, Xiao Chen, and Qun Liu. 2019. NEZHA: neural contextualized representation for chinese language understanding. *CoRR*, abs/1909.00204.

Jason Weston, Emily Dinan, and Alexander Miller. 2018. Retrieve and refine: Improved sequence generation models for dialogue. In *Proceedings of the 2018 EMNLP Workshop SCAI: The 2nd International Workshop on Search-Oriented Conversational AI*, pages 87–92, Brussels, Belgium. Association for Computational Linguistics.

Xingcheng Yao, Yanan Zheng, Xiaocong Yang, and Zhilin Yang. 2022. NLP from scratch without large-scale pretraining: A simple and efficient framework. In *Proc. of the ICML*, volume 162 of *Proceedings of Machine Learning Research*, pages 25438–25451. PMLR.

Juntao Yu, Bernd Bohnet, and Massimo Poesio. 2020. Named entity recognition as dependency parsing. In *Proc. of the ACL*, pages 6470–6476, Online. Association for Computational Linguistics.

Jingyi Zhang, Masao Utiyama, Eiichro Sumita, Graham Neubig, and Satoshi Nakamura. 2018a. Guiding neural machine translation with retrieved translation pieces. In *Proc. of the NAACL*, pages 1325–1335, New Orleans, Louisiana. Association for Computational Linguistics.

Qi Zhang, Jinlan Fu, Xiaoyu Liu, and Xuanjing Huang. 2018b. Adaptive co-attention network for named entity recognition in tweets. In *Proc. of the AAAI*, pages 5674–5681. AAAI Press.

Xin Zhang, Guangwei Xu, Yueheng Sun, Meishan Zhang, and Pengjun Xie. 2021. Crowdsourcing learning as domain adaptation: A case study on named entity recognition. In *Proc. of the ACL-IJCNLP*, pages 5558–5570, Online. Association for Computational Linguistics.

Type-enriched Hierarchical Contrastive Strategy for Fine-Grained Entity Typing

Xinyu Zuo, Haijin Liang, Ning Jing, Shuang Zeng, Zhou Fang and Yu Luo
Tencent Inc.
{xylonzuo,hodgeliang,shuangzeng,akirafang,yamiluo}@tencent.com
ning.jing.ustc@gmail.com

Abstract

Fine-grained entity typing (FET) aims to deduce specific semantic types of the entity mentions in text. Modern methods for FET mainly focus on learning what a certain type looks like. And few works directly model the type differences, that is, let models know the extent that one type is different from others. To alleviate this problem, we propose a type-enriched hierarchical contrastive strategy for FET. Our method can directly model the differences between hierarchical types and improve the ability to distinguish multi-grained similar types. On the one hand, we embed type into entity contexts to make type information directly perceptible. On the other hand, we design a constrained contrastive strategy on the hierarchical structure to directly model the type differences, which can simultaneously perceive the distinguishability between types at different granularity. Experimental results on three benchmarks, BBN, OntoNotes, and FIGER show that our method achieves significant performance on FET by effectively modeling type differences.

1 Introduction

Entity typing is a fundamental research problem in natural language processing (NLP), which aims to deduce the semantic types of the entity mentions in text. With the deepening of text understanding, the type sets of entities become more refined and ranging in from dozens (Hovy et al., 2006) to hundreds (Weischedel and Brunstein, 2005; Ling and Weld, 2012) or thousands (Choi et al., 2018). Therefore, fine-grained entity typing (FET) has gained more attention, which focuses on assigning more specific types to entities. For sentence in Figure 1, a FET system needs to assign a coarse-grained type *"/person"* and a fine-grained type *"/person/actor"* to the entity *"Vivien Leigh"*. The inferred fine-grained types could provide more specific prior knowledge for downstream NLP tasks, such as question answering (Lee et al., 2006) and entity linking (Leszczynski et al., 2022).

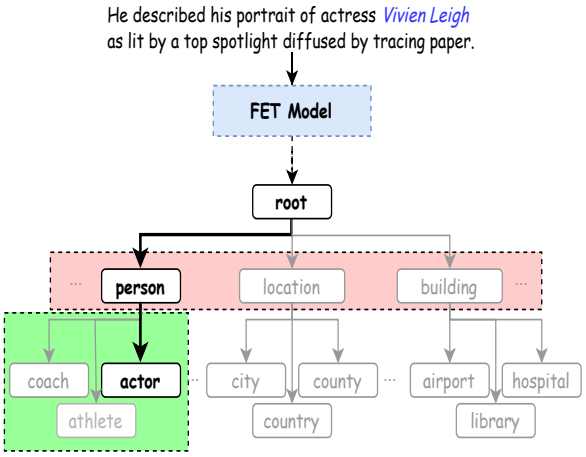

Figure 1: Example of fine-grained entity typing based on FIGER ontology. Green Box: the scope of visible types about *"person"* of the fine-grained contrastive strategy. Red Box: the scope of visible types of the coarse-grained contrastive strategy.

Considering the partial ontology of the FIGER dataset (Ling and Weld, 2012) in Figure 1, fine-grained entity types are often linked together in a hierarchical taxonomy, which makes the type boundaries increasingly blurred, especially for subtypes under the same coarse type. As shown in Figure 1, the fine-grained types *"coach"*, *"athlete"* and *"actor"*, all of which fall into the coarse-grained type *"person"*, are less differentiated.

In order to identify fine-grained types, prior work has concentrated on excavating more informative representations of types or entities, which benefiting from hand-crafted features (Ren et al., 2016), external resources (Onoe and Durrett, 2020; Li et al., 2022) or external pre-trained task (Xu et al., 2020). Most of them focus on learning *what a certain type looks like*, but few works have gone further to directly model the differences between types, that is, let models know *the extent that one*

Proceedings of the 29th International Conference on Computational Linguistics, pages 2405–2417
October 12–17, 2022.

type is different from others, which is more effective to distinguish among similar fine-grained types.

How to directly model the type differences? We argue that there are two key points, *Entity Type Awareness*, which refers to directly perceiving what type of entity is in the sentence, and *Type Differences Measure*, which refers to modeling how different the perceived type is from other types. For the first point, the intuitive idea is to expose the types directly, leading to a direct focus on what type the context represents. For the second point, heuristically, *direct is effective*, i.e., directly modeling which contexts represent the same types and which are different is the most efficient way to measure differences.

To this end, we propose a ty**P**e-enriched h**I**erarchical **CO**ntrastive stra**T**ey (**PICOT**) for fine-grained entity typing. Specifically, for *entity type awareness* with limited annotated data, inspired by prompt learning in entity typing (Ding et al., 2021), PICOT embeds the entity types in contexts via prompts to build type-rich expressions that guide the learning of correct types. Additionally, for *type differences measure*, PICOT takes a constrained contrastive strategy on hierarchical taxonomy to directly model the type differences from type-rich expressions. Concretely, as shown in Figure 1, PICOT is only concerned with the fine-grained types under the same coarse-grained type to learn the differences between fine-grained types. Similarly, PICOT is not concerned with what the fine-grained types are when distinguishing dissimilarities between coarse-grained types. Methodologically, PICOT learns the type differences at different granularity through type-rich expressions by limiting the scope of attention to types. Moreover, to further show models what a particular type is, we introduce a small number of type descriptions that directly expose richer type knowledge.

In experiments, we evaluate our model on three benchmarks. First, we concern with the standard evaluations and show that our model achieves the state-of-the-art performance on FET. Then we estimate the main components of PICOT. Finally, we do a visual analysis of the effectiveness of the type differentiation of PICOT.

In summary, the contributions are as follows:

- We propose a type-enriched hierarchical contrastive strategy (**PICOT**) for fine-grained entity typing. Our method can directly model the differences between hierarchical types and improve the ability to distinguish multi-grained similar types.
- First, we embed types into entity contexts to make type information directly perceptible. Then we design a constrained contrastive strategy on hierarchical taxonomy to directly model type differences at different granularities simultaneously.
- Experimental results on three benchmarks show that PICOT can achieve the SOTA performance on FET with limited annotated data.

2 Related Work

Entity Typing Named entity recognition (Tjong Kim Sang and De Meulder, 2003) and entity typing (Ling and Weld, 2012; Gillick et al., 2014) are fundamental research problems in NLP. Recently researchers pay more attention to fine-grained entity typing (FET) and ultra-fine entity typing (UFET) (Choi et al., 2018), which predict specific fine or ultra-fine types for given entities. To do so, obtaining more labeled data is the first research perspective, represented by distant supervision (Ling and Weld, 2012; Chen et al., 2019). With these, some researchers had focused on how to reduce noises in automatically labeled data (Gillick et al., 2014; Ren et al., 2016; Ren, 2020; Wu et al., 2019; Pan et al., 2022; Zhang et al., 2021b; Pang et al., 2022). Additionally, another key challenge is how to deal with hierarchical ontology. Most prior works regarded the hierarchical typing problem as a multi-label classification task and incorporated the hierarchical structure in different ways (Ren et al., 2016; Shimaoka et al., 2017; Xu and Barbosa, 2018; Murty et al., 2018; Chen et al., 2020b, 2022).

Some works attempted to mine more label information or better label representation. Abhishek et al. (2017) enhanced the label representation by sharing parameters; López and Strube (2020) embed types into a high-dimension; Xiong et al. (2019) introduced associated labels to enhance the label representation; Rabinovich and Klein (2017); Lin and Ji (2019) exploited co-occurrence structures and latent label representation; Additionally, several novel textual representations were applied to obtain richer entity contextual information, such as prompt based architecture (Ding et al., 2021) and box embeddings framework (Onoe et al., 2021).

Moreover, FET and UFET suffer from an obvious issue of the unseen types due to the lack of

annotated data. Therefore, a variety of paradigms were be studied to alleviate this issue (Huang et al., 2016; Ma et al., 2016; Obeidat et al., 2019; Zhou et al., 2018; Zhang et al., 2020; Chen et al., 2021). Moreover, some works further drew on different large-scale external data or knowledge to understand entity types (Onoe and Durrett, 2020; Xu et al., 2021; Dai et al., 2021; Li et al., 2022).

In summary, few prior works focus on directly modeling type differences. Therefore, this paper tries to let models know that one type is different from others without large-scale external resources. See Appendix B for more details.

Contrastive Learning Contrastive learning aims to further improve the model's ability to distinguish positive and negative examples, and has been a popular method for representation learning on computer vision tasks (Hjelm et al., 2018; Chen et al., 2020a; He et al., 2020). Recently, some researches have applied contrastive learning to natural language understanding tasks, aiming to obtain better text representations or to distinguish similar labels, such as the event causality identifier (Zuo et al., 2021), the contrastive self-supervised encoder (Fang and Xie, 2020), the supporting clustering framework (Zhang et al., 2021a), the abstractive summarization framework (Liu and Liu, 2021), the contrastive fine-tuning paradigm of pre-trained language for fine-grained text classification (Suresh and Ong, 2021), and so on. In this paper, we propose a constrained contrastive framework to directly model the hierarchical type differences.

Prompt Learning Prompt learning aims to leverage language prompts as contexts, and downstream tasks can be expressed as some cloze-style objectives similar to those pre-training objectives. Recently, a series of hand-crafted prompts have been widely used in natural language understanding (Liu et al., 2021b; Schick and Schütze, 2021; Feldman et al., 2019; Petroni et al., 2019; Trinh and Le, 2018; Ding et al., 2021). Moreover, to avoid expensive prompt design, automatic prompt has also been explored (Ren et al., 2016; Shin et al., 2020; Schick and Schütze, 2021), and some continuous prompts have also been proposed (Lester et al., 2021; Li and Liang, 2021). In this paper, we embed prompts to directly expose types in the entity contexts.

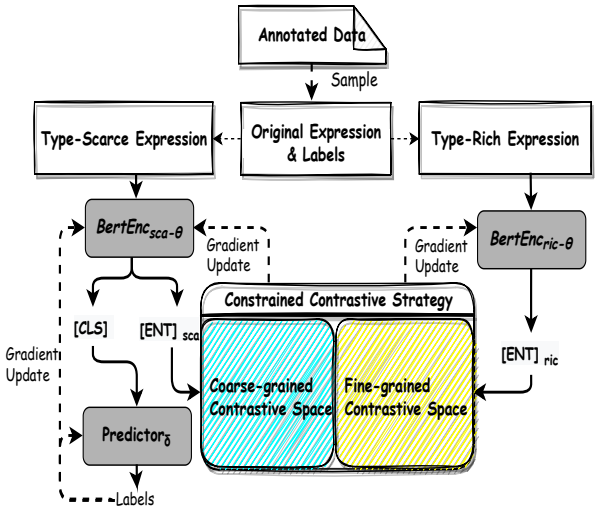

Figure 2: The framework of PICOT for FET (Sec. 4).

3 Problem Formulation

The input of fine-grained entity typing (FET) is a dataset $\mathcal{D} = \{x_1, x_2, ..., x_n\}$ with n sentences, a pre-defined hierarchical type ontology $\mathcal{Y}$, and each sentence x contains a marked entity e. A FET system is required to assign corresponding types to the given marked entity. Methodologically, for each input sequence $w_n = \{w_n^1, w_n^2, ..., w_n^t\}$ of the sentence x_n, FET aims to predict the correct multi-grained types $Y_n = \{y_n^1, y_n^2, ..., y_n^m\} \in \mathcal{Y}$ of the marked entity $e_n = \{w_n^l, ..., w_n^r\}$. For example in Figure 1, the correct type set of *"Vivien Leigh"* is $\{/person, /person/actor\}$, which contains a coarse-grained type "$/person$" and a fine-grained type "$/person/actor$".

4 Methodology

As shown in Figure 2, there are two key stages of PICOT for fine-grained entity typing.

- **Prompt-guided expression construction** (**ProExp**, Sec. 4.1). For *entity type awareness*, we construct two kinds of prompt-guided expressions, the *type-scarce expression* and the *type-rich expression*, which perceive the type patterns in context and expose type information directly, respectively.

- **Contrastive type knowledge transfer** (**ConTKT**, Sec. 4.2). For *type differences measure*, we propose a *constrained contrastive strategy* to directly model the differences among hierarchical types, and

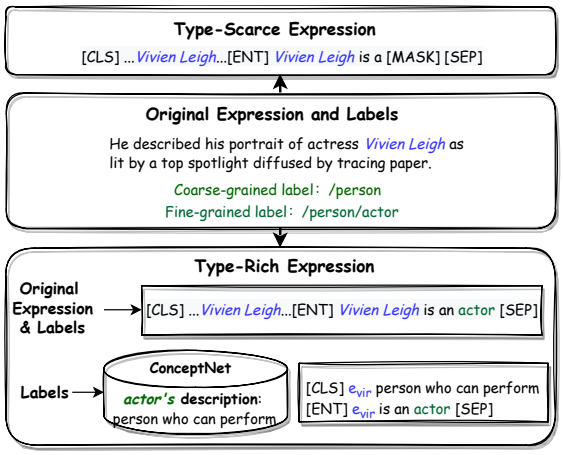

Figure 3: The illustration of prompt-guided expression construction (ProExp, Sec. 4.1).

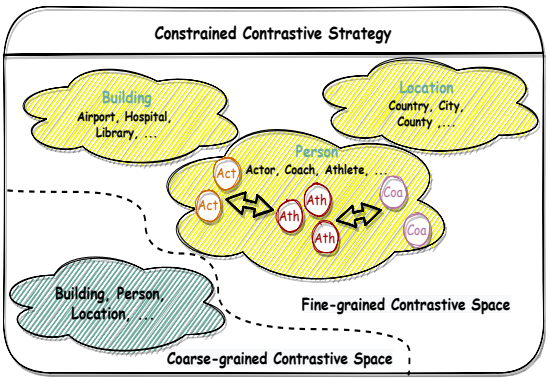

Figure 4: The illustration of constrained contrastive strategy in ConTKT (Sec. 4.2).

impart the type knowledge from type-rich expressions to predictor.

4.1 Prompt-guided Expression Construction (ProExp)

ProExp aims to convert the input sentences into type-scarce expressions and type-rich expressions based on entity-oriented prompts (Ding et al., 2021). The former could make models sensitive to type patterns in context, and the latter can be taken as the type knowledge resources based on type exposure.

Type-scarce Expression For each input sentence x_n, we construct type-scarce expression x_n^{ts} to guide the pre-trained language model (PLM, e.g. BERT (Devlin et al., 2019) used in this paper) encoder to efficiently exploit the entity contextual information, especially the type information. For simplicity, we choose declarative entity-oriented prompts to avoid grammatical errors.

Specifically, we first copy the marked entity e_n in x_n, then add a few conjunctions following the entity. Next, we add two specific words. One of them is *"[MASK]"* at the end of the expression, as a dummy for non-specific type. The other one is *"[ENT]"*, which bridges the original entity expression and prompt, and serves as an entry point for receiving type knowledge from type-rich expressions in ConTKT. The form of x_n^{ts} is as follows:

$$x_n^{ts} = x_n \textit{ [ENT] } e_n \textit{ is a [MASK].}$$

Type-rich Expression For each input sentence x_n, we also construct two kinds of type-rich expression x_n^{tr} as type knowledge resources for transfer in ConTKT when training. Intuitively, exposing the types directly to the context makes the expressions of entities type-aware.

Heuristically, fine-grained types contain both coarse- and fine- grained type properties. Therefore, we construct the type-rich expression x_n^{tr} of entity e_n by replacing the dummy type placeholder *"[MASK]"* in its type-scarce expression x_n^{ts} with its fine-grained types in Y_n[1]. Taking the entity in Figure 3 as an example, the form of x_n^{tr} is as follows:

$$x_n^{tr} = x_n \textit{ [ENT] } e_n \textit{ is an actor.}$$

Moreover, to better show what a particular type is, we introduce several descriptions (2 or 3) of each type from ConceptNet (Speer et al., 2017). Then we use them to directly expose richer type knowledge to construct extra type-rich expression x_{type}^{tr}. Specifically, for each fine-grained type, we replace x_n in x_n^{tr} with the combination of a virtual entity e_{vir} and one of descriptions. Taking the *actor* as an example:

$$x_{actor}^{tr} = e_{vir} \textit{ person who can perform [ENT] } e_{vir} \textit{ is an actor.}$$

4.2 Contrastive Type Knowledge Transfer (ConTKT)

ConTKT aims to directly model the differences among hierarchical types, and impart the type knowledge from type-rich expressions to predictor when training.

Expression Encoding We design two BERT encoders to encode two kinds of prompt-guided expressions for each entity e_n respectively. One is the encoder $BertEnc_{sca-\theta}$ for encoding type-scarce

[1]For entities with multiple fine-grained types, we concatenate the fine-grained types into one phrase by *"and"*.

expression x_n^{ts} when training and prediction, which digs out the type information of e_n in sentence x_n. Another is the encoder $BertEnc_{ric-\theta}$, which masters the type knowledge via encoding the type-rich expression x_n^{tr}.

Specifically, we first convert the x_n^{ts} and x_n^{tr} to the input sequences of two encoders respectively. Taking the example in Figure 3 as following:

$$\begin{aligned} w_n^{ts} =&[CLS], w_n^1, ..., w_n^t, [ENT], \\ & w_n^l, ..., w_n^r, is, a, [MASK], ., [SEP], \end{aligned} \tag{1}$$

$$\begin{aligned} w_n^{tr} =&[CLS], w_n^1, ..., w_n^t, [ENT], \\ & w_n^l, ..., w_n^r, is, an, actor, ., [SEP], \end{aligned} \tag{2}$$

$$\begin{aligned} w_{tr}^{actor} =&[CLS], des_{actor}^1, ..., des_{actor}^t, [ENT], \\ & w_{e_{vir}}, is, an, actor, ., [SEP]. \end{aligned} \tag{3}$$

where the des_{actor}^t is the token of type description and the $w_{e_{vir}}$ is the token of virtual entity e_{vir}.

After encoding, the representation $\boldsymbol{h}_n^{[CLS]_{ts}}$ of w_n^{ts} that encodes the contextual information of x_n^{ts} is used by predictor to predict types of entity e_n. Additionally, as mentioned above, the representation $\boldsymbol{h}_n^{[ENT]_{tr}}$ of w_n^{tr} is used as the exit of type knowledge contained in x_n^{tr}. Accordingly, the representation $\boldsymbol{h}_n^{[ENT]_{ts}}$ of w_n^{ts} is the entrance to receive the type knowledge from x_n^{tr} when training.

Constrained Contrastive Strategy We design a constrained contrastive strategy to directly model the hierarchical type differences based on prompt-guided expressions. Based on this, type knowledge is transferred to $BertEnc_{sca-\theta}$ from type-rich expressions by the contrast interaction between types at different granularities and the parameters sharing with $BertEnc_{ric-\theta}$.

Specifically, we only model the fine-grained type differences under the same coarse-grained type by bringing the same types closer while distancing different types. Likewise, for coarse-grained types, we do not care what the fine-grained types are in the same way. In this way, PICOT models the type differences between different granularities by limiting the scope of attention to types.

To specific, for one input batch $\mathcal{B} \subseteq \mathcal{D}$, there are two optimization objectives $\mathcal{L}_\theta^f$ and $\mathcal{L}_\theta^c$ to model the differences between fine-grained types and coarse-grained types respectively. And each optimization objective consists of two sub-optimization objectives, one for bringing the same types closer ($\mathcal{L}_\theta^{f+}$ and $\mathcal{L}_\theta^{c+}$), while another for distancing different types ($\mathcal{L}_\theta^{f-}$ and $\mathcal{L}_\theta^{c-}$):

$$\mathcal{L}_\theta^{f+} = \frac{1}{|\mathcal{Y}_\mathcal{B}^f|} \sum_{y \in \mathcal{Y}_\mathcal{B}^f} \frac{1}{2|\mathcal{B}^{f+}|} \sum_{x_i^y, x_j^y \in \mathcal{B}^{f+}}^{j \neq i} s(x_i^y, x_j^y), \tag{4}$$

$$\mathcal{L}_\theta^{f-} = -\frac{1}{2|\mathcal{B}^{f-}|} \sum_{x_i^{y_i}, x_j^{y_j} \in \mathcal{B}^{f-}}^{j \neq i, y_i \neq y_j} s(x_i^{y_i}, x_j^{y_j}), \tag{5}$$

$$\mathcal{L}_\theta^{c+} = \frac{1}{|\mathcal{Y}_\mathcal{B}^c|} \sum_{y \in \mathcal{Y}_\mathcal{B}^c} \frac{1}{2|\mathcal{B}^{c+}|} \sum_{x_i^y, x_j^y \in \mathcal{B}^{c+}}^{j \neq i} s(x_i^y, x_j^y), \tag{6}$$

$$\mathcal{L}_\theta^{c-} = -\frac{1}{2|\mathcal{B}^{c-}|} \sum_{x_i^{y_i}, x_j^{y_j} \in \mathcal{B}^{c-}}^{j \neq i, y_i \neq y_j} s(x_i^{y_i}, x_j^{y_j}), \tag{7}$$

$$s(x_i, x_j) = \lg \frac{e^{(dis(\boldsymbol{h}_i^E, \boldsymbol{h}_j^E)/\tau)}}{\sum_{x_i', x_j' \in \mathcal{B}*}^{j \neq i} e^{dis(\boldsymbol{h}_i^{E'}, \boldsymbol{h}_j^{E'})/\tau)}}, \tag{8}$$

where, take the $\mathcal{L}_\theta^f$ as illustration, $\mathcal{Y}_\mathcal{B}^f$ is the set of all fine-grained types in one batch, $y_\mathcal{B}^f$ is one of them. And $\mathcal{B}^{f+}$ consists of all x_n^{ts}, x_n^{tr} and x_{type}^{tr} with same fine-grained type $y \in \mathcal{Y}_\mathcal{B}^f$. Oppositely, $\mathcal{B}^{f-}$ consists of all x_n^{ts}, x_n^{tr} and x_{type}^{tr} with same coarse-grained type but different fine-grained types $y_i, y_j \in \mathcal{Y}_\mathcal{B}^f$. Moreover, s is the similarity between x_i and x_j, $\boldsymbol{h}_i^E$ and $\boldsymbol{h}_j^E$ are the $\boldsymbol{h}_n^{[ENT]}$ of x_i and x_j after encoding respectively, dis is the ℓ_2-distance function to measure the distance of two representation, τ is a temperature that adjusts the concentration level and $\mathcal{B}*$ is $\mathcal{B}^{f+}$ or $\mathcal{B}^{f-}$. Likewise, the optimization objectives are similar for $\mathcal{L}_\theta^c$.

Learning of FET After transferring, the $\boldsymbol{h}_n^{[CLS]_{ts}}$ of x_n^{ts} output by $BertEnc_{sca-\theta}$, which has learned types knowledge, is fed to the predictor to identify the types of the input e_n as following:

$$y_n^* \leftarrow MLP(\boldsymbol{h}_n^{[CLS]_{ts}}), \tag{9}$$

where y_n^* is the predicted types of e_n in x_n.

For training of two encoders and predictor, we add the constrained contrastive losses $\mathcal{L}_\theta^f$ and $\mathcal{L}_\theta^c$ to the classification loss $\mathcal{L}_\delta$. Finally, we minimize

Algorithm 1 Learning of PICOT for FET.

Require: type-scarce expression x^{ts}, type-rich expression x^{tr}, extra type-rich expression x^{tr}_{type}.
Training:
Stage: PROMPT-GUIDED EXPRESSION CONSTRUCTION
  for each batch $\mathcal{B} \in \mathcal{D}$ **do**
    for input entity e_n with its sentence $x_n \in \mathcal{B}$ **do**
      Construct type-scarce expression x^{ts}_n;
      Construct type-rich expression x^{tr}_n;
    end for

    for each fine-grained $type \in \mathcal{Y}_\mathcal{B}$ **do**
      Construct extra type-rich expression x^{tr}_{type};
    end for
  end for
end Stage:

Stage: CONTRASTIVE TYPE KNOWLEDGE TRANSFER
  for each batch $\mathcal{B} \in \mathcal{D}$ **do**
    for each fine-grained type y_f in $\mathcal{Y}_\mathcal{B}$ **do**
      Compute $\mathcal{L}^f_\theta$ in equation (4) and (5);
    end for

    for each coarse-grained type y_c in $\mathcal{Y}_\mathcal{B}$ **do**
      Compute $\mathcal{L}^c_\theta$ in equation (6) and (7);
    end for

    Compute batch classification loss $\mathcal{L}_\delta$ in (10);
    Compute $\mathcal{L}$ in equation (11);
    Stochastic gradient update θ and δ in (12);
  end for
end Stage:

the L and stochastic gradient update the θ and δ as Algorithm 1:

$$\mathcal{L}_\delta = BCEWithLogits(y^*_n, y_n), \tag{10}$$

$$\begin{aligned}\mathcal{L} = \mathcal{L}_\delta &+ \lambda_f(\mathcal{L}^{f+}_\theta + \mathcal{L}^{f-}_\theta) \\ &+ \lambda_c(\mathcal{L}^{c+}_\theta + \mathcal{L}^{c-}_\theta),\end{aligned} \tag{11}$$

$$\theta, \delta \leftarrow \eta \nabla \mathcal{L}, \tag{12}$$

where, λ_f and λ_c are the weights of $\mathcal{L}^f_\theta$ and $\mathcal{L}^c_\theta$ respectively, η is the learning rate.

5 Experiments

5.1 Experimental Setup

Datasets and Evaluation Metrics We conduct experiments on three standard FET datasets and follow the version processed and split by Onoe et al. (2021). (1) **BBN** (Weischedel and Brunstein, 2005), which contains 56 types and each type has a maximum type hierarchy level of 2; (2) **OntoNotes** (Gillick et al., 2014), which is sampled from the OntoNotes (Weischedel et al., 2013) corpus and re-annotated with 89 types in 3-level hierarchy. Additionally, we ignore the *other* type, which has no obvious meaning, and categorize it into two-level types; (3) **FIGER** (Ling and Weld, 2012), which contains 113 types and each type also has a maximum type hierarchy level of 2. Table 1 is the statistics on the coarse-grained and fine-grained types of three datasets. Moreover, we evaluate three datasets using the standard metrics: Macro F1 (Ma-F1) and Micro F1 (Mi-F1).

Datasets	#Coarse	#Fine	#Fine/Coarse
BBN	17	39	2.3
OntoNotes	20	68	3.4
FIGER	47	66	1.4

Table 1: Statistics on the coarse-grained and fine-grained types of three datasets.

Parameters Settings For a fair comparison, similar to Onoe et al. (2021), the BERT encoders are BERT-Large architecture[2], which has 24-layers, 1024-hiddens, and 16-heads. For parameters, we set the learning rate of η as 8e-6, and set the temperature τ of the contrastive loss as 0.1 tuned on the development set. Moreover, we also tune the batch size to 96 on the development set. The λ_f and λ_c are setted as 0.1/0.1/0.1 and 0.1/0.1/0.01/ for three datasets respectively. And we apply the early stop and AdamW gradient strategy to optimize all models. Additionally, to simulate constraints like the previous work (Chen et al., 2020b; Onoe et al., 2021), we use the same three simple rules to modify the model's predictions or training data on BBN datasets: (1) dropping *"person"* if *"organization"* exists, (2) dropping *"location"* if *"gpe"* exists, and (3) replacing *"facility"* by *"fac"*, since both the two tags appear in the training set but only *"fac"* in the test set. See Appendix C for more detailed settings.

Compared Methods Same as previous methods, we prefer the following models which use the same versions of three datasets and do not rely on large-scale external knowledge or resources as our compared methods[3]. (1) **Ren et al. (2016)**, a embedding method which separately models clean and noisy data with type hierarchy; (2) **Abhishek et al. (2017)**, a neural network model that jointly learns entities and their contexts representation; (3) **Zhang et al. (2018)**, a neural architecture which leverages both document and sentence level infor-

[2] https://github.com/google-research/bert

[3] There are different versions of three datasets exist.

Methods	BBN		OntoNotes		FIGER	
	Ma-F1	Mi-F1	Ma-F1	Mi-F1	Ma-F1	Mi-F1
Ren et al. (2016)	74.1	75.7	71.1	64.7	69.3	66.4
Abhishek et al. (2017)	74.1	75.7	68.5	63.3	78.0	74.9
Zhang et al. (2018)	75.7	75.1	72.1	66.5	78.7	75.5
Chen et al. (2020b) (exclusive)	63.2	61.0	72.4	67.2	82.6	**80.8**
Chen et al. (2020b) (undefined)	79.7	80.5	73.0	68.1	80.5	78.1
Lin and Ji (2019)	79.3	78.1	82.9*	77.3*	83.0	79.8
Onoe et al. (2021) (vector)	78.3	78.0	76.2	68.9	81.6	77.0
Onoe et al. (2021) (box)	78.7	78.0	77.3	70.9	79.4	75.0
Liu et al. (2021a)	-	-	77.6	71.8	-	-
PICOT	**81.8**	**82.2**	**78.7**	**72.1**	**84.7**	79.6

Table 2: Results on fine-grained entity typing. *: Not directly comparable since large-scale augmentated data is used. The results are tested for significance at the 0.05 level.

Methods	BBN		
	Ma-F1	Mi-F1	∇
PICOT (our)	**81.8**	**82.8**	-
w/o Exp.$_{tr_{des}}$	81.6	82.2	-0.2/-0.6
w/o Exp.$_{tr}$	81.4	81.7	-0.4/-1.1
w/o Exp.$_{ts\&tr}$	81.1	81.5	-0.7/-1.3
Previous SOTA	79.7	80.5	-

Table 3: Ablation results of the prompt-guided expression (ProExp, Sec. 4.1) of FET on BBN. w/o Exp.$_{tr_{des}}$ denotes a varietal PICOT that without extra type-rich expression when training; w/o Exp.$_{tr}$ denotes a varietal PICOT that without all type-rich expressions when training; w/o Exp.$_{ts\&tr}$ denotes a varietal PICOT that without all prompt-guided expressions when training.

Methods	BBN		
	Ma-F1	Mi-F1	∇
PICOT (our)	**81.8**	**82.8**	-
w/o ConTKT$_{coar}$	80.0	80.4	-1.8/-2.4
w/o ConTKT$_{fine}$	80.5	81.0	-1.3/-1.8
w/o ConTKT	80.2	80.7	-1.6/-2.1

Table 4: Ablation results of the contrastive type knowledge transfer (ConTKT, Sec. 4.2) of FET on BBN. w/o ConTKT$_{coar}$ denotes a varietal PICOT that removes coarse-grained contrastive loss; w/o ConTKT$_{fine}$ denotes a varietal PICOT that removes fine-grained contrastive loss; w/o ConTKT denotes a varietal PICOT that removes the whole ConTKT.

mation; (4) **Chen et al. (2020b) (exclusive)**, a classifier for hierarchical FET that embraces ontological structure with exclusive interpretations; (5) **Chen et al. (2020b) (undefined)**, a same classifier as (4) but with different undefined interpretations; (6) **Lin and Ji (2019)**, a FET model with a novel attention mechanism and a hybrid type classifier; (7) **Onoe et al. (2021) (vector)**, a vector-based model for FET; (8) **Onoe et al. (2021) (box)**, a box-based model for FET; (9) **Liu et al. (2021a)**, a FET model with extrinsic and intrinsic dependencies between labels. Moreover, all results of compared methods are directly copied from the previous papers.

5.2 Our Method vs. State-of-the-art Methods

Table 2 shows the results of FET on BBN, OntoNotes, and FIGER. From the results, we can observe that (see Appendix A for more results):

(1) On BBN, our PICOT outperforms all baselines and achieves the best performance on Macro F1 and Micro F1 values, which are 81.8% and 82.2%, outperforming the state-of-the-art by a margin of 2.1% and 1.7% respectively, which justifies its effectiveness. Moreover, among the three datasets, the BBN dataset has the least training data but the largest boost. This indicates that PICOT can effectively mine type information in limited labeled data by sensing type knowledge and directly modeling the differences between types.

(2) On OntoNotes, compared with the methods without large external data, our PICOT also achieves the best performance on Macro F1 and Micro F1 values, which are 78.7% and 72.1%, outperforming by a margin of 1.1% and 0.3% respectively. Although OntoNotes has three times more training data than BBN and can provide more type information to compared models, the proposed PICOT can still further improve the performance, which demonstrates the effectiveness of directly modeling type differences.

(3) On FIGER, the largest one in three datasets,

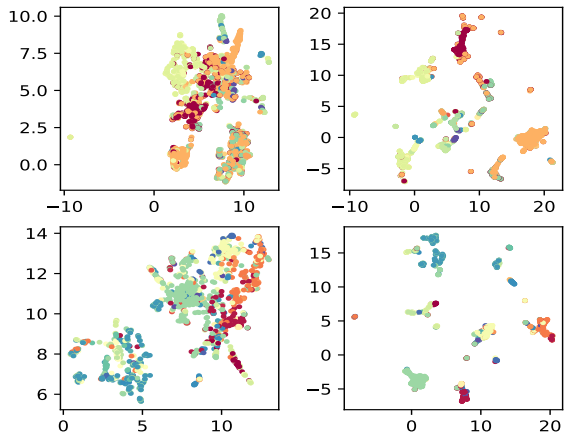

Figure 5: The visualization of type representation clustering without- (left) and with- (right) PICOT on development dataset. Specifically, top row is coarse-grained type clustering and bottom row is fine-grained type clustering. Each color represents a kind of type

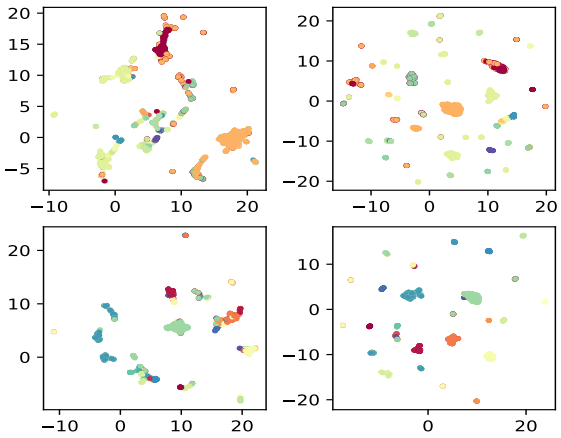

Figure 6: The visualization of type representation clustering of type-scarce (left) and type-rich (right) expressions on development dataset. Specifically, top row is coarse-grained type and bottom row is fine-grained type. Each color represents a kind of type

our PICOT outperforms the best compared method on Macro-F1 value by a margin of 1.7%, which further proves the effectiveness of PICOT in mining type differences with labeled data. It is worth noting that FIGER has a slightly lower performance on the Micro-F1 value due to the inconsistent of some test samples, in which only have fine-grained types (e.g., *"/organization/sports_team"* is present, but *"/organization"* is missing).

5.3 Effect of Prompt-guided Expression

We analyze the effect of the prompt-guided expression (ProExp, Sec. 4.1) on BBN dataset. As shown in Table 3, from the results, we can observe that: (1) after removing the type-rich expressions (***w/o Exp.***$_{tr_{des}}$ and ***w/o Exp.***$_{tr}$), the performance of FET significantly decreases. This proves that exposing the type information directly to the model can bring great help to determine entity types. (2) Comparing the ***w/o Exp.***$_{tr_{des}}$ with the ***previous SOTA***, we find that without introducing any external type descriptions, PICOT could also effectively mine type knowledge within the limited labeled data and improve the performance of FET. (3) Comparing the ***w/o Exp.***$_{tr_{des}}$ with ***w/o Exp.***$_{tr}$, just with a small amount of external types descriptions, our PICOT's type knowledge exposure and transfer framework also enhance the performance. (4) ***w/o Exp.***$_{ts\&tr}$ also achieves good results without any prompt-guided expressions, which also shows the effectiveness of the contrastive transfer strategy.

5.4 Effect of Contrastive Type Knowledge Transfer

We analyze the effect of the contrastive type knowledge transfer (ConTKT, Sec. 4.2) on the BBN dataset. As shown in Table 4, from the results, we can observe that: (1) after removing the ConTKT (***w/o ConTKT***), the performance of FET significantly decreases. This illustrates that the contrastive strategy can effectively improve the discrimination of similar types, which is important for FET. (2) Comparing ***w/o ConTKT***$_{coar}$, ***w/o ConTKT***$_{fine}$ and ***PICOT***, we find that both coarse-grained and fine-grained contrastive training play a key role in the measurement of type differences. (3) It is worth noting that, comparing ***w/o ConTKT***$_{coar}$ with ***w/o ConTKT***, we find that training with type-scarce expression without contrastive strategy works better than only using fine-grained type contrastive strategy. Meanwhile, coarse-grained contrastive training alone (w/o ***ConTKT***$_{fine}$) only give a small boost for FET. These indicate that only the combination of coarse-grained and fine-grained contrastive strategies can achieve the desired results. Specifically, coarse-grained contrast ensures the base performance while fine-grained contrast further improves the ability to discriminate types.

5.5 Visualization of the Effect of Type Distinguishing

To further illustrate the effect of PICOT, in Figure 5, we cluster the representations of *"[ENT]"* in the type-scarce expressions before and after train-

ing by UMAP downscaling (McInnes et al., 2018). The comparisons of the left and right subgraphs show that the differentiation of *"[ENT]"* representations, which is the entrance for type knowledge, is both greatly improved by PICOT among the coarse-grained and fine-grained types. This illustrates that PICOT can effectively improve the model's ability to discriminate against similar types.

As shown in Figure 6, to elucidate the effect of direct type exposure for type differentiation, we cluster the *"[CLS]"* representations of type-scarce and type-rich expressions, respectively. The comparisons show that the representation of type-rich expressions is more discriminative, especially for fine-grained types, which can effectively guide the model to identify types with high similarity.

6 Conclusion

We propose a type-enriched hierarchical contrastive strategy for fine-grained entity typing. Our method can directly model the differences between hierarchical types and improve the ability to distinguish multi-grained similar types. First, we embed types into entity contexts to make type information directly perceptible. Then we design a constrained contrastive strategy on hierarchical taxonomy to directly model type differences at different granularities simultaneously. Experimental results on three benchmarks show that PICOT can achieve state-of-the-art performance on FET with limited annotated data.

References

Abhishek Abhishek, Ashish Anand, and Amit Awekar. 2017. Fine-grained entity type classification by jointly learning representations and label embeddings. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 1, Long Papers*, pages 797–807, Valencia, Spain. Association for Computational Linguistics.

Bo Chen, Xiaotao Gu, Yufeng Hu, Siliang Tang, Guoping Hu, Yueting Zhuang, and Xiang Ren. 2019. Improving distantly-supervised entity typing with compact latent space clustering. *arXiv preprint arXiv:1904.06475*.

Ting Chen, Simon Kornblith, Mohammad Norouzi, and Geoffrey Hinton. 2020a. A simple framework for contrastive learning of visual representations. In *Proceedings of the 37th International Conference on Machine Learning*, volume 119 of *Proceedings of Machine Learning Research*, pages 1597–1607. PMLR.

Tongfei Chen, Yunmo Chen, and Benjamin Van Durme. 2020b. Hierarchical entity typing via multi-level learning to rank. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 8465–8475, Online. Association for Computational Linguistics.

Yi Chen, Jiayang Cheng, Haiyun Jiang, Lemao Liu, Haisong Zhang, Shuming Shi, and Ruifeng Xu. 2022. Learning from sibling mentions with scalable graph inference in fine-grained entity typing. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 2076–2087, Dublin, Ireland. Association for Computational Linguistics.

Yi Chen, Haiyun Jiang, Lemao Liu, Shuming Shi, Chuang Fan, Min Yang, and Ruifeng Xu. 2021. An empirical study on multiple information sources for zero-shot fine-grained entity typing. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 2668–2678, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Eunsol Choi, Omer Levy, Yejin Choi, and Luke Zettlemoyer. 2018. Ultra-fine entity typing. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 87–96, Melbourne, Australia. Association for Computational Linguistics.

Hongliang Dai, Yangqiu Song, and Haixun Wang. 2021. Ultra-fine entity typing with weak supervision from a masked language model. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 1790–1799, Online. Association for Computational Linguistics.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, Minneapolis, Minnesota. Association for Computational Linguistics.

Ning Ding, Yulin Chen, Xu Han, Guangwei Xu, Pengjun Xie, Hai-Tao Zheng, Zhiyuan Liu, Juanzi Li, and Hong-Gee Kim. 2021. Prompt-learning for fine-grained entity typing. *arXiv preprint arXiv:2108.10604*.

Hongchao Fang and Pengtao Xie. 2020. CERT: contrastive self-supervised learning for language understanding. *CoRR*, abs/2005.12766.

Joshua Feldman, Joe Davison, and Alexander M. Rush. 2019. Commonsense knowledge mining from pretrained models. *CoRR*, abs/1909.00505.

Dan Gillick, Nevena Lazic, Kuzman Ganchev, Jesse Kirchner, and David Huynh. 2014. Context-dependent fine-grained entity type tagging. *arXiv preprint arXiv:1412.1820*.

Kaiming He, Haoqi Fan, Yuxin Wu, Saining Xie, and Ross Girshick. 2020. Momentum contrast for unsupervised visual representation learning.

R Devon Hjelm, Alex Fedorov, Samuel Lavoie-Marchildon, Karan Grewal, Phil Bachman, Adam Trischler, and Yoshua Bengio. 2018. Learning deep representations by mutual information estimation and maximization. *arXiv preprint arXiv:1808.06670*.

Eduard Hovy, Mitch Marcus, Martha Palmer, Lance Ramshaw, and Ralph Weischedel. 2006. Ontonotes: the 90% solution. In *Proceedings of the human language technology conference of the NAACL, Companion Volume: Short Papers*, pages 57–60.

Lifu Huang, Jonathan May, Xiaoman Pan, and Heng Ji. 2016. Building a fine-grained entity typing system overnight for a new x (x= language, domain, genre). *arXiv preprint arXiv:1603.03112*.

Changki Lee, Yi-Gyu Hwang, Hyo-Jung Oh, Soojong Lim, Jeong Heo, Chung-Hee Lee, Hyeon-Jin Kim, Ji-Hyun Wang, and Myung-Gil Jang. 2006. Fine-grained named entity recognition using conditional random fields for question answering. In *Asia information retrieval symposium*, pages 581–587. Springer.

Brian Lester, Rami Al-Rfou, and Noah Constant. 2021. The power of scale for parameter-efficient prompt tuning. *CoRR*, abs/2104.08691.

Megan Leszczynski, Daniel Y Fu, Mayee F Chen, and Christopher Ré. 2022. Tabi: Type-aware bi-encoders for open-domain entity retrieval. *arXiv preprint arXiv:2204.08173*.

Bangzheng Li, Wenpeng Yin, and Muhao Chen. 2022. Ultra-fine entity typing with indirect supervision from natural language inference. *arXiv preprint arXiv:2202.06167*.

Xiang Lisa Li and Percy Liang. 2021. Prefix-tuning: Optimizing continuous prompts for generation. *CoRR*, abs/2101.00190.

Ying Lin and Heng Ji. 2019. An attentive fine-grained entity typing model with latent type representation. In *EMNLP-IJCNLP 2019*, pages 6197–6202, Hong Kong, China. Association for Computational Linguistics.

Xiao Ling and Daniel S Weld. 2012. Fine-grained entity recognition. In *Twenty-Sixth AAAI Conference on Artificial Intelligence*.

Qing Liu, Hongyu Lin, Xinyan Xiao, Xianpei Han, Le Sun, and Hua Wu. 2021a. Fine-grained entity typing via label reasoning. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 4611–4622, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Xiao Liu, Yanan Zheng, Zhengxiao Du, Ming Ding, Yujie Qian, Zhilin Yang, and Jie Tang. 2021b. GPT understands, too. *CoRR*, abs/2103.10385.

Yixin Liu and Pengfei Liu. 2021. SimCLS: A simple framework for contrastive learning of abstractive summarization. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 2: Short Papers)*, pages 1065–1072, Online. Association for Computational Linguistics.

Federico López and Michael Strube. 2020. A fully hyperbolic neural model for hierarchical multi-class classification. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 460–475, Online. Association for Computational Linguistics.

Yukun Ma, Erik Cambria, and Sa Gao. 2016. Label embedding for zero-shot fine-grained named entity typing. In *Proceedings of COLING 2016, the 26th International Conference on Computational Linguistics: Technical Papers*, pages 171–180, Osaka, Japan. The COLING 2016 Organizing Committee.

Leland McInnes, John Healy, and James Melville. 2018. Umap: Uniform manifold approximation and projection for dimension reduction.

Shikhar Murty, Patrick Verga, Luke Vilnis, Irena Radovanovic, and Andrew McCallum. 2018. Hierarchical losses and new resources for fine-grained entity typing and linking. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 97–109, Melbourne, Australia. Association for Computational Linguistics.

Rasha Obeidat, Xiaoli Fern, Hamed Shahbazi, and Prasad Tadepalli. 2019. Description-based zero-shot fine-grained entity typing. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 807–814, Minneapolis, Minnesota. Association for Computational Linguistics.

Yasumasa Onoe, Michael Boratko, Andrew McCallum, and Greg Durrett. 2021. Modeling fine-grained entity types with box embeddings. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 2051–2064, Online. Association for Computational Linguistics.

Yasumasa Onoe and Greg Durrett. 2020. Fine-grained entity typing for domain independent entity linking.

In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pages 8576–8583.

Weiran Pan, Wei Wei, and Feida Zhu. 2022. Automatic noisy label correction for fine-grained entity typing. *arXiv preprint arXiv:2205.03011*.

Kunyuan Pang, Haoyu Zhang, Jie Zhou, and Ting Wang. 2022. Divide and denoise: Learning from noisy labels in fine-grained entity typing with cluster-wise loss correction. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1997–2006, Dublin, Ireland. Association for Computational Linguistics.

Fabio Petroni, Tim Rocktäschel, Patrick S. H. Lewis, Anton Bakhtin, Yuxiang Wu, Alexander H. Miller, and Sebastian Riedel. 2019. Language models as knowledge bases? *CoRR*, abs/1909.01066.

Maxim Rabinovich and Dan Klein. 2017. Fine-grained entity typing with high-multiplicity assignments. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 330–334, Vancouver, Canada. Association for Computational Linguistics.

Quan Ren. 2020. Fine-grained entity typing with hierarchical inference. In *2020 IEEE 4th Information Technology, Networking, Electronic and Automation Control Conference (ITNEC)*, volume 1, pages 2552–2558. IEEE.

Xiang Ren, Wenqi He, Meng Qu, Lifu Huang, Heng Ji, and Jiawei Han. 2016. AFET: Automatic fine-grained entity typing by hierarchical partial-label embedding. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 1369–1378, Austin, Texas. Association for Computational Linguistics.

Timo Schick and Hinrich Schütze. 2021. Exploiting cloze-questions for few-shot text classification and natural language inference. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*, pages 255–269, Online. Association for Computational Linguistics.

Sonse Shimaoka, Pontus Stenetorp, Kentaro Inui, and Sebastian Riedel. 2017. Neural architectures for fine-grained entity type classification. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 1, Long Papers*, pages 1271–1280, Valencia, Spain. Association for Computational Linguistics.

Taylor Shin, Yasaman Razeghi, Robert L. Logan IV, Eric Wallace, and Sameer Singh. 2020. Autoprompt: Eliciting knowledge from language models with automatically generated prompts. *CoRR*, abs/2010.15980.

Robyn Speer, Joshua Chin, and Catherine Havasi. 2017. Conceptnet 5.5: An open multilingual graph of general knowledge. In *Thirty-first AAAI conference on artificial intelligence*.

Varsha Suresh and Desmond Ong. 2021. Not all negatives are equal: Label-aware contrastive loss for fine-grained text classification. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 4381–4394, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Erik F. Tjong Kim Sang and Fien De Meulder. 2003. Introduction to the CoNLL-2003 shared task: Language-independent named entity recognition. In *Proceedings of the Seventh Conference on Natural Language Learning at HLT-NAACL 2003*, pages 142–147.

Trieu H. Trinh and Quoc V. Le. 2018. A simple method for commonsense reasoning. *CoRR*, abs/1806.02847.

Ralph Weischedel and Ada Brunstein. 2005. Bbn pronoun coreference and entity type corpus. *Linguistic Data Consortium, Philadelphia*, 112.

Ralph Weischedel, Martha Palmer, Mitchell Marcus, Eduard Hovy, Sameer Pradhan, Lance Ramshaw, Nianwen Xue, Ann Taylor, Jeff Kaufman, Michelle Franchini, et al. 2013. Ontonotes release 5.0 ldc2013t19. *Linguistic Data Consortium, Philadelphia, PA*, 23.

Junshuang Wu, Richong Zhang, Yongyi Mao, Hongyu Guo, and Jinpeng Huai. 2019. Modeling noisy hierarchical types in fine-grained entity typing: A content-based weighting approach. In *IJCAI*, pages 5264–5270.

Wenhan Xiong, Jiawei Wu, Deren Lei, Mo Yu, Shiyu Chang, Xiaoxiao Guo, and William Yang Wang. 2019. Imposing label-relational inductive bias for extremely fine-grained entity typing. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 773–784, Minneapolis, Minnesota. Association for Computational Linguistics.

Peng Xu and Denilson Barbosa. 2018. Neural fine-grained entity type classification with hierarchy-aware loss. *arXiv preprint arXiv:1803.03378*.

Zenan Xu, Daya Guo, Duyu Tang, Qinliang Su, Linjun Shou, Ming Gong, Wanjun Zhong, Xiaojun Quan, Nan Duan, and Daxin Jiang. 2020. Syntax-enhanced pre-trained model. *arXiv preprint arXiv:2012.14116*.

Zenan Xu, Daya Guo, Duyu Tang, Qinliang Su, Linjun Shou, Ming Gong, Wanjun Zhong, Xiaojun Quan, Daxin Jiang, and Nan Duan. 2021. Syntax-enhanced

pre-trained model. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 5412–5422, Online. Association for Computational Linguistics.

Dejiao Zhang, Feng Nan, Xiaokai Wei, Shang-Wen Li, Henghui Zhu, Kathleen R. McKeown, Ramesh Nallapati, Andrew O. Arnold, and Bing Xiang. 2021a. Supporting clustering with contrastive learning. *CoRR*, abs/2103.12953.

Haoyu Zhang, Dingkun Long, Guangwei Xu, Muhua Zhu, Pengjun Xie, Fei Huang, and Ji Wang. 2021b. Learning with noise: improving distantly-supervised fine-grained entity typing via automatic relabeling. In *Proceedings of the Twenty-Ninth International Conference on International Joint Conferences on Artificial Intelligence*, pages 3808–3815.

Sheng Zhang, Kevin Duh, and Benjamin Van Durme. 2018. Fine-grained entity typing through increased discourse context and adaptive classification thresholds. In *Proceedings of the Seventh Joint Conference on Lexical and Computational Semantics*, pages 173–179, New Orleans, Louisiana. Association for Computational Linguistics.

Tao Zhang, Congying Xia, Chun-Ta Lu, and Philip Yu. 2020. MZET: Memory augmented zero-shot fine-grained named entity typing. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 77–87, Barcelona, Spain (Online). International Committee on Computational Linguistics.

Ben Zhou, Daniel Khashabi, Chen-Tse Tsai, and Dan Roth. 2018. Zero-shot open entity typing as type-compatible grounding. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 2065–2076, Brussels, Belgium. Association for Computational Linguistics.

Xinyu Zuo, Pengfei Cao, Yubo Chen, Kang Liu, Jun Zhao, Weihua Peng, and Yuguang Chen. 2021. Improving event causality identification via self-supervised representation learning on external causal statement. In *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*, pages 2162–2172, Online. Association for Computational Linguistics.

A Supplementary Experiment Results

A.1 Effect of the Weights of Different Contrastive Loss

Coarse / Fine	**0.01**	**0.1**	**0.5**
0.01	-	81.6	-
0.1	81.4	**81.8**	81.3
0.5	-	78.5	-

Table 5: Macro F1 of PICOT on BBN with different coarse- (λ_c) and fine-grained (λ_f) contrastive weights.

Coarse / Fine	**0.01**	**0.1**	**0.5**
0.01	-	82.1	-
0.1	82.0	**82.8**	81.8
0.5	-	79.9	-

Table 6: Micro F1 of PICOT on BBN with different coarse- (λ_c) and fine-grained (λ_f) contrastive weights.

To further explore the effect of contrastive loss of different granularities, we vary the weights of fine- and coarse-grained contrastive loss to observe the performance of PICOT on the BBN test set, respectively. As shown in Table 5 and 6, we can notice that the type knowledge is not fully migrated when the coarse-grained and fine-grained contrastive loss weights are too small, and overly affects the classification performance when the weights are too large. It is worth noting that excessive fine-grained contrastive loss weights significantly degrade the performance because many fine-grained types are not completely distinct, and some types could occur simultaneously. Therefore, excessive differentiation of fine-grained types will confuse models.

A.2 Effect of the *"[ENT]"* Position

Pos.	Ma-F1	Mi-F1
After *"[CLS]"*	80.7	81.2
Before prompt	**81.8**	**82.8**

Table 7: Performance of PICOT with different *"[ENT]"* positions.

As shown in Table 7, we further explore the effect of the position of *"[ENT]"* as type knowledge exit and entry in the input sequence on the PICOT

performance. From the results, we can see that placing *"[ENT]"* between the entity context and the type prompt allows for more efficient migration and reception of type knowledge.

B Supplementary Related Work

Named entity recognition (Tjong Kim Sang and De Meulder, 2003) and entity typing (Ling and Weld, 2012; Gillick et al., 2014) are fundamental research problems in NLP. Recently researchers pay more attention on fine-grained entity typing (FET) and ultra-fine entity typing (UFET) (Choi et al., 2018), which predicts specific fine or ultra-fine types for given entities. To do so, obtaining more labeled data is the first research perspective for FET, represented by the distant supervision annotation method (Ling and Weld, 2012). With these, some researches had focused on how to reduce noises in automatically labeled data, such as a heuristic constraint pruning approach (Gillick et al., 2014), a partial-label loss (Ren et al., 2016), a penalty optimization term (Ren, 2020), and a novel content-sensitive weighting schema (Wu et al., 2019).

Additionally, one key challenge is how to deal with hierarchical type ontology. Most prior works regarded the hierarchical typing problem as a multi-label classification task and incorporated the hierarchical structure in different ways. Ren et al. (2016) used a predefined label hierarchy to reduce noises; Shimaoka et al. (2017) proposed a hierarchical label encoding method; Xu and Barbosa (2018) employed a normalized hierarchical loss; Murty et al. (2018) learned a subtyping relation to constrain the type embedding; Chen et al. (2020b) designed a novel loss function to exploit label hierarchies.

Some work attempted to mine more label information or better label representation. Abhishek et al. (2017) enhanced the label representation by sharing parameters; López and Strube (2020) embed types into a high-dimension; Xiong et al. (2019) introduced associated labels to enhance the label representation; Rabinovich and Klein (2017) exploited co-occurrence structures during label set prediction; (Lin and Ji, 2019) reconstructed the co-occurrence structure via latent label representation; Liu et al. (2021a) reasoned fine-grained types by discovering label dependencies knowledge. Additionally, several novel textual representations were applied to obtain richer entity contextual information. Ding et al. (2021) investigated the application of prompt-learning to predict fine-grained entity types. Onoe et al. (2021) studied the box embeddings to capture hierarchies of types.

Moreover, FET and UFET suffer from an obvious issue of the unseen types due to the lack of annotated data. Therefore, a variety of paradigms were being studied to alleviate this issue, such as a hierarchical clustering model (Huang et al., 2016), a prototypical embedding method (Ma et al., 2016), a context-description matching model based on type descriptions from Wikipedia (Obeidat et al., 2019), a classifier based on Freebase types of its type-compatible, (Zhou et al., 2018), a novel framework which transfers the knowledge from seen types to the unseen ones (Zhang et al., 2020), and an empirical study on multiple auxiliary information (Chen et al., 2021). To further alleviate the lack of annotated data, some work draws on different large-scale external data or knowledge to understand the types of entities in the sentences. Onoe and Durrett (2020) used hyperlinked mentions in Wikipedia to distantly label large scale data and train an entity typing model; Xu et al. (2021) introduced a new pre-training task of predicting the syntactic distance in dependency tree based on large scale texts; Dai et al. (2021) automatically generated new ultra-fine entity typing data with labels; Li et al. (2022) presented LITE, a new approach that formulates entity typing as an NLI problem based on external data.

In summary, few prior works focus on directly modeling the differences between types. Therefore, this paper tries to let models know that one type is different from others without large-scale external resources.

C Main Experimental Environments and Other Parameters Settings

C.1 Experimental Environments

We deploy all models on a server with Tesla P40 GPU. Specifically, the configuration environment of the server is ubuntu 16.04, and our framework mainly depends on python 3.8.8 and Torch 1.11.

C.2 Other Parameters Settings

All the final hyper-parameters for evaluation are averaged after 3 independent tunings on the development set. Moreover, the three datasets BBN, OntoNotes, and FIGER achieve optimal results at the 20th/10th/5th epochs, which take half a day, one day, and two days, respectively.

This is an appendix.

Document-Level Relation Extraction via Pair-Aware and Entity-Enhanced Representation Learning

Xiusheng Huang[1,2], **Hang Yang**[1,2], **Yubo Chen**[1,2], **Jun Zhao**[1,2],
Kang Liu[1,2,4], **Weijian Sun**[3] and **Zuyu Zhao**[3]
[1]School of Artificial Intelligence, University of Chinese Academy of Sciences, Beijing, China
[2]National Laboratory of Pattern Recognition, Institute of Automation, CAS, Beijing, China
[3]Huawei Technologies Co., Ltd, Shenzhen, China
[4]Beijing Academy of Artificial Intelligence, Beijing, China
huangxiusheng2020@ia.ac.cn,
{hang.yang,yubo.chen,jzhao,kliu}@nlpr.ia.ac.cn,
{sunweijian,zhaozuyu1}@huawei.com

Abstract

Document-level relation extraction aims to recognize relations among multiple entity pairs from a whole piece of article. Recent methods achieve considerable performance but still suffer from two challenges: a) the relational entity pairs are sparse, b) the representation of entity pairs is insufficient. In this paper, we propose **P**air-**A**ware and **E**ntity-**E**nhanced(PAEE) model to solve the aforementioned two challenges. For the first challenge, we design a Pair-Aware Representation module to predict potential relational entity pairs, which constrains the relation extraction to the predicted entity pairs subset rather than all pairs; For the second, we introduce a Entity-Enhanced Representation module to assemble directional entity pairs and obtain a holistic understanding of the entire document. Experimental results show that our approach can obtain state-of-the-art performance on four benchmark datasets DocRED, DWIE, CDR and GDA.

1 Introduction

Relation extraction (RE) is a primary task in the field of information extraction, which aims to identify the relationships between two entities in a document. Previous works mainly focus on sentence-level relation extraction, i.e, recognizing the relationships between entities in a sentence. However, large amounts of relationships are expressed over multiple sentences in real-world applications. According to DocRED (Yao et al., 2019), above 40.7% of the relational facts can only be extracted from multiple sentences. Therefore, it requires the model to capture complex interactions among entities in the whole document. Previous work commonly referred to this problem as document-level relation extraction which has attracted much attention recently (Nan et al., 2020; Zhou et al., 2021; Zhang et al., 2021). Although the considerable performance of these methods, there are still two critical challenges in document-level RE to be addressed.

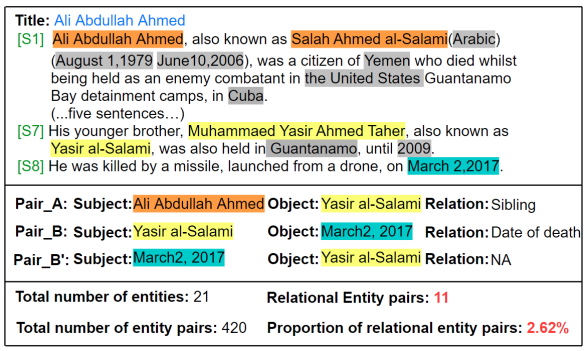

Fig. 1: An example with entity pairs and relations from DocRED. Entity mentions only involved in these relation instances are colored, other entities in the document are highlighted in grey.

The first challenge is how to identify relational entity pairs that are **sparse** in a document. Specifically, given a document with n entities, there will be $n(n-1)$ combinations of entities to classify. However, only a few entity pairs have predefined relationships. For example, as shown in Figure 1, this document contains 21 entities with 420 potential entity pairs. However, the number of relational entity pairs is only 11, accounting for 2.62 % of the total entity pairs. According to statistics, for DocRED (Yao et al., 2019) dataset, the proportion are 3.18% and 3.11% in the train set and dev set, respectively. To further explore the impact of sparsity on performance bottlenecks, we conduct a diagnostic experiment on DocRED dataset. Utilizing previous SOTA model ATLOP (Zhou et al., 2021), we divulge the information of whether existing a predefined relationship between the entities to the model. Specifically, we just concatenate a 0-1 variable on the original representation of entity pairs, where "1" represents the entity pair exists a predefined

Proceedings of the 29th International Conference on Computational Linguistics, pages 2418–2428
October 12–17, 2022.

relationship. Experimental results show that the F1 score reaches 93.50% in dev set which is 32.20% higher than normal setting. This demonstrates that the importance of identifying the relational entity pairs when facing the sparsity problem (Wang et al., 2019a).

The second challenge is how to effectively model the representations of entity pairs. There are commonly two characteristics for entity pairs. Firstly, the **entities-scattering**, which means the entities of an entity pair may scatter across multiple sentences. Figure 1 illustrates an example from the DocRED dataset. For Pair_A, the subject *Ali Abdullah Ahmed* and object *Yasir al-Salami* are distributed in different sentences ([S1] and [S7]), which requires model to capture the long-distance dependency among entities across sentences. Secondly, the **directivity** of entity pairs, which means that the relationships of entity pairs are directional. For example, the Pair_B and Pair_B′ in the Figure 1, their subject and object are opposite, and the relations of them are different. Therefore, this challenge requires the model to assemble directional entity pairs and obtain a holistic understanding of the cross-sentence context. To model the representation of entity pairs, most current approaches include graph-based methods and transformer-based methods. Specifically, some methods (Christopoulou et al., 2019; Nan et al., 2020; Wang et al., 2020) construct a document graph with structured attention, dependency structures or heuristics. Meanwhile, considering the transformer can capture long-distance information, some studies (Wang et al., 2019a; Tang et al., 2020; Zhou et al., 2021) directly apply pre-trained language models without introducing graph structures. However, they directly concatenate two entities together to obtain the representation of entity pair, without considering the directivity of entity pairs and modeling the representations of entity pairs adequately.

In this paper, we propose a **P**air-**A**ware and **E**ntity-**E**nhanced (PAEE) model for document-level RE. To deal with the sparsity of entity pairs, we propose the *Pair-Aware Representation*(PAR) module to identify potential relational entity pairs, which constrains the relation extraction to the predicted pairs subset rather than all pairs. Furthermore, to capture the global features of triples, PAR utilizes TNet (Papadopoulos et al., 2021) to model the relation between entity pairs, unlike previous methods, PAR designs a Sliding Window Filling Strategy for filling relation matrix, which enhances the interaction between entity pairs. To effectively model the representation of entity pairs, we focus on the global interactions among sentences and entities. Specifically, we propose a *Entity-Enhanced Representation*(EER) module. The EER first introduces a Representation-Enhanced Encoder to facilitate the interaction between all sentences and entities. In this way, EER obtains a holistic understanding of the entire document. Then, considering that the characteristics of entities as subjects and objects are different, especially in different relationship categories, EER utilizes a Cross Matching method to assemble directional entity pairs.

Experiments on four document-level relation extraction datasets, DocRED (Yao et al., 2019), DWIE (Zaporojets et al., 2021), CDR (Li et al., 2016) and GDA (Wu et al., 2019), demonstrate that our PAEE model significantly outperforms the state-of-the-art methods. To our best knowledge, we are first to consider the sparsity and the directivity of relational entity pairs for the task.

We summarize our contributions as follows:

- To alleviate the negative impact of sparsity, we propose Pair-Aware Representation(PAR) module, which promotes the interaction between entity pairs and accurately identifies potential relational entity pairs.
- To model the representation of entity pairs better, we propose Entity-Enhanced Representation(EER) module, which is based on a Representation-Enhanced Encoder to capture the global context for the scattered entities and a Cross Matching method to assemble directional entity pairs.
- We conduct experiments on four public document-level relation extraction datasets. Experimental results demonstrate that our PAEE model can achieves state-of-the-art performance compared with baselines.

2 Methodology

Before introducing our proposed approach for PAEE in this section, we first introduce the problem definition. Given a document d and a set of entities $\{e_i\}_{i=1}^{n}$, and there are $n(n-1)$ entity

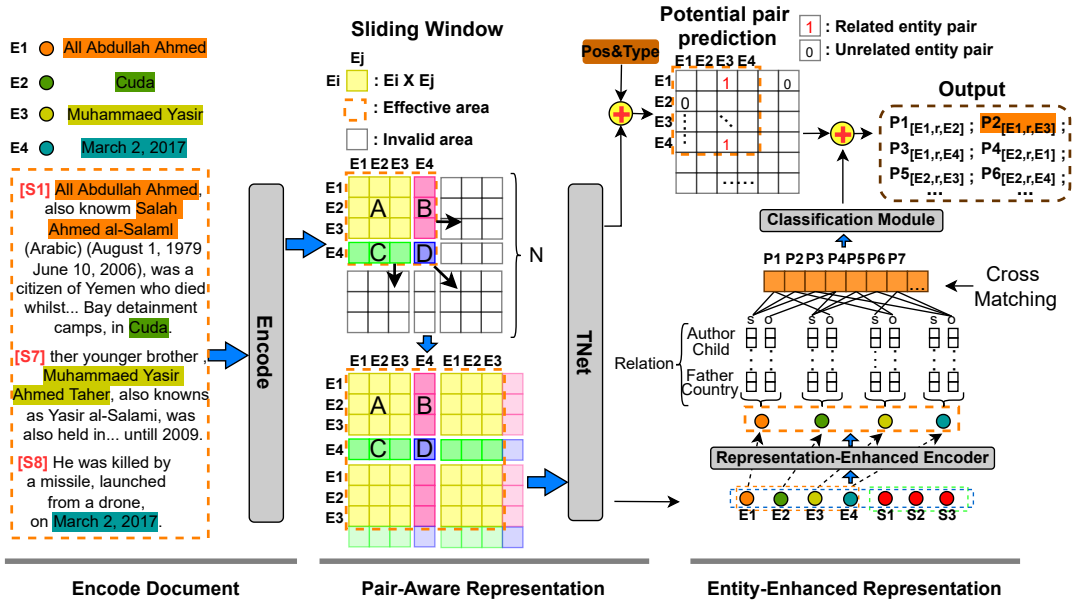

Fig. 2: The overall architecture of our PAEE. Given a document, PAEE will obtain the representation of entities with the Encoder module; Then, the Pair-Aware Representation module and Entity-Enhanced Representation module are designed to get the representation of entity pairs. Finally, PAEE will obtain the relations between entities with the Classification Module.

pairs in this document. The task of document-level relation extraction is to predict a subset of relations from $R \cup \{\text{NA}\}$ between the entity pairs $(e_s, e_o)_{s,o=1...n;s\neq o}$, where R is a pre-defined set of relationships, e_s, e_o are identified as subject and object entities, respectively. The entity pairs that do not express any relation are labeled NA. In addition, the model needs to predict the label of all entity pairs $(e_s, e_o)_{s,o=1...n;s\neq o}$ at the test time. To model relation extraction between e_s and e_o, we define a $N \times N$ matrix V, where entry $V_{s,o}$ indicates the relation type between e_s and e_o. Entities in V are arranged according to their first appearance in the document. Unlike Zhang et al. (2021), we utilize the sliding window filling strategy to fill matrix V, which can enhance the interaction between entity pairs and is beneficial to relation extraction.

2.1 Encoder

Given a document $D = [x_t]_{t=1}^{l}$ with l tokens, we insert special symbols " $< e >$ " and " $< /e >$ " to mark the entity positions at the start and end of mentions. It is adapted from the entity marker technique (Zhang et al., 2017; Shi and Lin, 2019; Soares et al., 2019). We leverage the pre-trained language model as an encoder to obtain the embedding as follows:

$$H = [h_1, h_2, ..., h_l] = \text{Encoder}([x_1, x_2, ..., x_l]) \quad (1)$$

where h_i is the embedding of the token x_i. Note that some documents are longer than 512, we thus leverage a *dynamic window* to encoder whole documents (Zhou et al., 2021). We take the embedding of " $< e >$ " at the start of mentions as the mention embeddings. Then, for an entity e_i with mentions $\{m_j^i\}_{j=1}^{N_{e_i}}$, we leverage a logsumexp pooling to obtain the entity embedding $\mathbf{e}_i$:

$$\mathbf{e}_i = \log \sum_{j=1}^{N_{e_i}} \exp(\mathbf{m}_j) \quad (2)$$

This pooling accumulates signals from mentions in the document. Compared with the mean pooling, the logsumexp pooling shows better performance in the experiment. We calculate the entity-level relation matrix based on entity-to-entity relevance. Specifically, we constructed a D-dimensional feature vector $\mathbf{V}(e_s, e_o)$ to capture the relevance between entities. Note that we add the position and type information of entities to enrich the vector $\mathbf{V}(e_s, e_o)$. For intra-sentential and inter-sentential entity pairs, their position captured by a 0-1 variable pos.

$$a_{(s,o)} = softmax(\sum_{i=1}^{K} A_i^s \cdot A_i^o),$$
$$\mathbf{V}(e_s, e_o) = W_1 \cdot H \cdot a_{(s,o)} \quad (3)$$

where W_1 is the learnable weight matrix, $a_{(s,o)}$ is the attention weight of last layer for entity-aware attention and A_i^s refers to the tokens' importance to the i-th entity, H is the contextual embedding in Eq.1. The K is the number of head in the transformer.

2.2 Pair-Aware Representation

In this section, we propose Pair-Aware Representation(PAR) module to enhance the interaction between entity pairs and identify potential entity pairs. We build the module base on existing BERT baselines (Zhou et al., 2021; Zhang et al., 2021) and integrate other techniques to further improve the performance.

Sliding Window Filling Strategy. To capture the relevance of entity pairs, we utilize TNet (Papadopoulos et al., 2021) to expand receptive field and learn more global and local information. The TNet is a novel multi-scale hard-attention architecture that constantly adjusts the number of elements to help us focus on the related entity pairs. We take the matrix $\mathbf{V} \in R^{N \times N \times D}$ as a D-channel variable and feed it into TNet, where N is the largest number of entities, counted from all the dataset samples. However, the number of entities annotated in each document is usually different and often less than N, thus, we propose a sliding window filling strategy to fill matrix before feeding matrix $\mathbf{V}$ into TNet.

$$\mathbf{V}^{'} = \text{Sliding}(\mathbf{V}),$$
$$\mathbf{Y} = \text{TNet}(W_2\mathbf{V}^{'}) \quad (4)$$

where $\mathbf{Y} \in R^{N \times N \times D^{'}}$ denotes the entity-level relation matrix. W_2 is the learnable weight matrix and $D^{'}$ is much smaller than D. As it shows in the Figure 2, the diagonal dots are far apart in the matrix $\mathbf{V}$, which makes their interaction poor (Ronneberger et al., 2015). Instead of previous zero filling (Ronneberger et al., 2015), we utilize the sliding window filling strategy to shorten the distance between entity pairs. Specifically, for the orange dashed window in matrix $\mathbf{V}$, we slide the window in three directions: transverse, longitudinal and oblique, then we will obtain a filled matrix $\mathbf{V}^{'}$. Furthermore, in the whole matrix $\mathbf{V}^{'}$, the spacing between dots that were originally far away was significantly shortened, which facilitates the interaction between them.

Potential Pair Prediction. This component is shown as a 0-1 distribution box in Figure 2, where "1" means potential relational entity pairs. Given a document which contains multiple entity pairs, different from previous works (Zhou et al., 2021; Zhang et al., 2021) which redundantly perform relationship classification to every entity pair, we utilize this module to predict potential relational entity pairs. Specifically, we utilize the average pooling operation (Lin et al., 2013) to obtain the representation $\mathbf{P}_{pair}$ of each entity pair, and then feed it into the binary classifier to get the potential entity pairs.

$$\mathbf{P}^{'}_{pair} = \kappa(\mathbf{P}_{pair}; \lambda; pos; sub_{emb}; obj_{emb}) \quad (5)$$

where κ and λ denote the binary classifier and threshold, sub_{emb} is the type embedding of subject in entity pairs, obj_{emb} is the type embedding of object in entity pairs. We model it as a binary classification task, and the corresponding entity pairs will be assigned with tag "1" if the probability exceeds a certain threshold λ or with tag "0" otherwise (as shown in Figure 2). By concatenating the classification results $\mathbf{P}^{'}_{pair}$ with matrix $\mathbf{Y}$ in Eq.4, we will obtain a entity-level relation matrix $\mathbf{Y}_{pair} \in R^{D^{'}+1}$ incorporating the information of candidate pairs.

2.3 Entity-Enhanced Representation

In this section, we propose a Entity-Enhanced Representation module to model the representation of entity pairs. Specifically, we introduce Representation-Enhanced Encoder to facilitate the interaction between all sentences and entities. Then, considering that the characteristics of entity as subject and object are different, especially in different relational categories, we propose Cross Matching method to assemble directional entity pairs.

Representation-Enhanced Encoder. To enable the awareness of document-level contexts for sentences and entities, we employ a Representation-Enhanced Encoder to facilitate the interaction between all sentences and entities. Formally, we can obtain the entity embedding $\mathbf{e}_i$ from Eq.2 and

the embedding $[h_1, h_2, ..., h_l]$ of every token in sentence S_i^l from Eq.1, where l is the sentence length. Hence the sentence embedding S_i can be obtained by a max-pooling operation over the token sequence representation. Then we employ the Transformer (Vaswani et al., 2017) module, Representation-Enhanced Encoder, as the encoder to obtain the document-aware embedding for sentences and entities. Note that we add the sentence representation with sentence position embeddings to inform the sentence order before feeding them into the Representation-Enhanced Encoder.

$$[\mathbf{H}^e; \mathbf{H}^s] = \text{RE-Encoder}(\mathbf{e}_1 ... \mathbf{e}_{N_e}; S_1 ... S_{N_s}) \quad (6)$$

where S_i is the local representation for i-th sentence and $\mathbf{e}_i$ is the representation for i-th entity. Utilizing the Representation-Enhanced Encoder, we can obtain the document-aware entities representation $\mathbf{H}^e \in R^{N_e \times D}$, N_e is the number of entities in a document.

Cross Matching. To extract the different features of entity as subject and object respectively, we utilize the Sub-Obj layer (a Linear Layer($N_e \times D$, $2 \times N_e \times N_c$)) for feature separation. Meanwhile, we map these features to each relationship category, which enhances the interaction between entities in each relationship. For the Sub-Obj layer, we set a corresponding loss (Appendix A.1) to learn that a single entity may have several relationships . The features of entity as subject and object in each relationship can be calculated as:

$$[F_{sub}; F_{obj}] = \text{Sub-Obj}(\mathbf{H}^e) \quad (7)$$

where $F_{sub}, F_{obj} \in R^{N_e \times N_c}$ denotes the features of entities as subjects and objects respectively, N_c is the number of relationship categories and N_e is the number of entities. Meanwhile, we concatenate these features with the representation $\mathbf{H}^e$ of entities, then we will obtain $\mathbf{e}_{sub}, \mathbf{e}_{obj} \in R^{N_e \times (D+N_c)}$, which are the representations of entities as subjects and objects respectively.

Classification Module. Given the entity embedding $\mathbf{e}_{sub}$ and $\mathbf{e}_{obj}$ with entity-level relation matrix $\mathbf{Y}_{pair}$ in section 2.2, we map them to hidden representations z with a feedforward neural network. Then we calculate the probability of relation r by bilinear function and sigmoid activation. Formally, we obtain:

Statistics/Datasets	DocRED	DWIE	CDR	GDA
# Train	3,053	602	500	23,353
# Dev	1,000	98	500	5,839
# Test	1,000	99	500	1,000
# Relation	97	65	2	2
Avg. # entity per Doc.	19.5	14	7.6	5.4
Avg. # Ment. per Ent.	1.4	1.6	2.7	3.3

Table 1: Statistics of the experimental datasets.

$$\begin{aligned} z_s &= \tanh(\mathrm{W_s} \cdot \mathrm{e_{sub}} + \mathbf{Y}_{\mathrm{s,o}}), \\ z_o &= \tanh(\mathrm{W_o} \cdot \mathrm{e_{obj}} + \mathbf{Y}_{\mathrm{s,o}}), \\ P(r \mid \mathbf{e}_{sub}, \mathbf{e}_{obj}) &= \sigma(z_s W_r z_o + b_r) \end{aligned} \quad (8)$$

where $\mathbf{Y}_{s,o}$ is the entity-pair representation of (s, o) in matrix $\mathbf{Y}_{pair}$, σ denotes the sigmoid function, $W_s \in R^{d \times d}$, $W_o \in R^{d \times d}$, $b \in R$, and $W_r \in R^{d \times d}$ are learnable parameters.

3 Experiments

3.1 Experimental Setup

Datasets. We evaluated our method on four document-level RE datasets. The statistical results of the datasets are shown in Table 2.

- **DocRED** (Yao et al., 2019) is a large-scale document-level relation extraction dataset. It is constructed from Wikipedia articles. DocRED contains 96 relationships and 3,053/1,000/1,000 instances for training, validating and test, respectively.
- **DWIE** (Zaporojets et al., 2021) is a document-level RE dataset after processing. This dataset has 700 documents for train and 99 documents for test. The training set is then randomly split into two parts: 602 documents for train and 98 for development.
- **CDR** (Li et al., 2016) is a relation extraction dataset in the biomedical domain, which is human-annotated and aims to predict the binary interactions between Chemical and Disease concepts.
- **GDA** (Wu et al., 2019) is a large-scale dataset in the biomedical domain, which aims to predict the binary interactions between Gene and Disease concepts.

Pretrained Transformers. We initialize PAEE with three different pretrained language models including BERT (Devlin et al., 2019), RoBERTa (Liu et al., 2019) and SciBERT (Beltagy et al., 2019).

Model	Dev			Test	
	Ign F1 (%)	F1 (%)	Rela (%)	Ign (%)	F1 (%)
BERT$_{\text{base}}$ (Wang et al., 2019b)	-	54.16	58.41	-	53.20
BERT-TS$_{\text{base}}$ (Wang et al., 2019a)	-	54.42	-	-	53.92
HIN-BERT$_{\text{base}}$ (Tang et al., 2020)	54.29	56.31	-	53.70	55.60
CorefBERT$_{\text{base}}$ (Ye et al., 2020)	55.32	57.51	-	54.54	56.96
SSAN-BERT$_{\text{base}}$ (Xu et al., 2021a)	57.03	59.19	68.37	56.06	58.41
ATLOP-BERT$_{\text{base}}$ (Zhou et al., 2021)	59.22	61.09	70.42	59.31	61.30
DocuNet-BERT$_{\text{base}}$ (Zhang et al., 2021)	59.86	61.28	70.55	59.45	61.42
PAEE-BERT$_{\text{base}}$ (Ours)	**60.38** (↑0.52)	**62.62** (↑1.34)	**74.61** (↑4.06)	**60.42** (↑0.97)	**62.98** (↑1.56)
BERT$_{\text{large}}$ (Ye et al., 2020)	56.67	58.83	67.42	56.47	58.69
CorefBERT$_{\text{large}}$ (Ye et al., 2020)	56.82	59.01	68.78	56.4	58.83
RoBERTa$_{\text{large}}$ (Ye et al., 2020)	57.14	59.22	69.23	57.51	59.62
CorefRoBERTa$_{\text{large}}$ (Ye et al., 2020)	57.35	59.43	69.77	57.9	60.25
SSAN-RoBERTa$_{\text{large}}$ (Xu et al., 2021a)	60.25	62.08	73.21	59.47	61.42
ATLOP-RoBERTa$_{\text{large}}$ (Zhou et al., 2021)	61.32	63.18	74.39	61.39	63.4
DocuNet-RoBERTa$_{\text{large}}$ (Zhang et al., 2021)	61.43	63.40	74.56	61.52	63.52
PAEE-RoBERTa$_{\text{large}}$ (Ours)	**62.44** (↑1.01)	**64.82** (↑1.42)	**79.02** (↑4.46)	**63.06** (↑1.54)	**65.09** (↑1.57)

Table 2: Main results on the development and test set of DocRED. We report the official test score on the CodaLab scoreboard with the best checkpoint on the development set. The performance of our method is followed by the improvements (↑) over the previous state-of-the-art method DocuNet.

- **BERT** employs a Transformer encoder to learn from large unlabeled text corpora and sub-word units to represent textual tokens, which contains 12 and 24 self-attention layers.
- **RoBERTa** is an improved version of BERT, which removes the Next Sentence Prediction task and adopts way larger text corpora as well as more training steps.
- **SciBERT** adopts the same model architecture as BERT, but is trained on scientific text instead. In this paper, we provide SciBERT-initialized PAEE on the two biomedical domain datasets **CDR** and **GDA**.

Implementation Detail. We used cased BERT-base, or RoBERTa-large as the encoder on DocRED and SciBERT-base on CDR and GDA. We use mixed-precision training (Micikevicius et al., 2018) based on the Apex library. Our model is optimized with AdamW (Loshchilov and Hutter, 2018) using learning rates $\in [2e{-}5, 3e{-}5, 5e{-}5, 1e{-}4]$, with a linear warmup (Goyal et al., 2018) for the first 6% steps followed by a linear decay to 0. We set the matrix size $N{=}42$ in the Figure 2 and $\lambda = 0.3$. We preprocess CDR and GDA dataset following Christopoulou et al. (2019). For GDA, we split 20% of the training set for development. For CDR, we merge the training set and dev set to train the final model after the best hyper-parameter is set. The calculation of loss will be provided in the appendix A.1. We report the mean and standard deviation of F1 on the development set by conducting 5 runs of training using different random seeds.

Evaluation. Our primary evaluation metric are **F1**, **Ign F1** (Yao et al., 2019) and **Rela**. **Ign F1** is computed by excluding relational facts that already appeared in the training set. It avoids information leakage from the training set. We propose **Rela** for evaluating the accuracy of identifying relational entity pairs. The prediction results of entity pairs are processed into two classification tasks. The relationship between entity pairs is divided into NA and non NA.

3.2 Experiment Results

We conduct experiments on four DocRE datasets to verify the effectiveness of our method PAEE.

Results on the DocRED Dataset. In the DocRED dataset, we compare PAEE with transformer-based models, including BERT$_{\text{base}}$ (Wang et al., 2019b), BERT-TS$_{\text{base}}$ (Wang et al., 2019a), CorefBERT$_{\text{base}}$ (Ye et al., 2020), HIN-BERT$_{\text{base}}$ (Tang et al., 2020), SSAN (Xu et al., 2021a) and ATLOP$_{\text{base}}$ on the DocRED dataset; and graph-based models, including GEDA (Li et al., 2020), LSR (Nan et al., 2020), GLRE (Wang et al., 2020), GAIN (Zeng et al., 2020), HeterGSAN (Xu et al., 2021b) and DocuNet (Zhang et al., 2021). Results in Table 2 shows that PAEE performs better than these methods. Our best model, PAEE built upon RoBERTa$_{\text{large}}$, is **+1.42 / +1.57 F1** better on dev/test set than DocuNet-RoBERTa$_{\text{base}}$

Model	Dev			Test		
	Ign F1 (%)	F1 (%)	Rela (%)	Ign (%)	F1 (%)	Rela (%)
CNN	37.65	47.73	56.43	34.65	46.14	55.83
LSTM	40.86	51.77	59.31	40.81	52.60	61.42
BiLSTM	40.46	51.92	59.49	42.03	54.47	64.78
GAIN	58.63	62.55	74.75	62.37	67.57	78.89
ATLOP	59.03	64.82	77.43	62.09	69.94	82.12
PAEE (Ours)	**62.05**(↑3.02)	**67.52**(↑2.70)	**82.01**(↑4.58)	**66.45**(↑4.36)	**73.10**(↑3.16)	**86.45**(↑4.33)

Table 3: Main results on the development and test set of DWIE. The performance of our method is followed by the improvements (↑) over the previous state-of-the-art method ATLOP.

Model	CDR	GDA
BRAN (Verga et al., 2018)	62.1	-
LSR (Nan et al., 2020)	64.8	82.2
DHG (Zhang et al., 2020)	65.9	83.1
GLRE (Wang et al., 2020)	68.5	-
SciBERT (Beltagy et al., 2019)	65.1	82.5
SSAN-SciBERT (Xu et al., 2021a)	68.7	83.7
ATLOP-SciBERT (Zhou et al., 2021)	69.4	83.9
DocuNet-SciBERT (Zhang et al., 2021)	76.3	85.3
PAEE-SciBERT	**78.2** (↑1.9)	**87.7** (↑2.4)

Table 4: Test F1 score (%) on CDR and GDA dataset. Our PAEE model with the SciBERT encoder outperforms the current state-of-the-art results. The performance of our method is followed by the improvements (↑) over the previous state-of-the-art method DocuNet.

Model	Ign F1	F1
PAEE-BERT$_{\text{base}}$	60.38	62.62
w/o PAR	57.67 (↓ 2.71)	59.61 (↓ 3.01)
w/o EER	59.57 (↓ 0.81)	61.53 (↓ 1.09)
w/o SW	59.72 (↓ 0.66)	61.72 (↓ 0.90)
w/o PPP	59.63 (↓ 0.75)	61.52 (↓ 1.10)

Table 5: Ablation study of PAEE on DocRED. We turn off different components of the model one at a time.

(Zhang et al., 2021), and obtains a new state-of-the-art(SOTA) result. Meanwhile, our method achieves **4.46**% improvements of Rela score on the DocRED dataset. The significant performance gain of our method over the baselines demonstrates that the proposed PAEE is very effective for this task.

Results on the DWIE Dataset. As show in Table 5, Our method improves upon the basic ATLOP model (Zhou et al., 2021) by **2.70**% and **3.16**% in term of F1 score on the Dev and Test sets of DWIE dataset, respectively. Meanwhile, our PAEE achieves **4.33**% improvements of Rela score. We attribute the improvements to that our method PAEE takes advantage of Pair-Aware Representation and Entity-Enhanced Representation, thus achieving superior performance than the previous model ATLOP.

Results on the Biomedical Datasets. In the biomedical datasets, we compare PAEE with baselines including: BRAN (Verga et al., 2018), LSR (Nan et al., 2020), DHG (Zhang et al., 2020), GLRE (Wang et al., 2020), SciBERT (Beltagy et al., 2019), SSAN (Xu et al., 2021a), ATLOP (Zhou et al., 2021) and DocuNet (Zhang et al., 2021). Following ATLOP (Zhou et al., 2021), we replace the encoder with SciBERT (Beltagy et al., 2019), which is pre-trained on the scientific publication corpora. Results in Table 4 shows that PAEE has achieved new state-of-the-art with the F1 score reached to **78.2%** and **87.7%** on CDR and GDA datasets.

3.3 Ablation Study

To show the efficacy of our proposed techniques, we conduct an ablation study experiment by turning off one component at a time. 1) w/o PAR, which removes the Pair-Aware Representation module; 2) w/o EER, which removes the Entity-Enhanced Representation module, we directly splice two entities as the representation of entity pairs; 3) w/o SW, which removes the Sliding Window strategy, the previous zero filling method is used to fill the whole relationship matrix; 4) w/o PPP, which removes the Potential Pair Prediction module. We present the results of ablation study in Table 5. From the results, we can observe that:

(1) **Effectiveness of Pair-Aware Representation.** When we remove the Pair-Aware Representation module from the PAEE, the F1 score drops by 3.01% on DocRED dataset. It proves the Pair-Aware Representation module is very effective for the task.

(2) **Effectiveness of Entity-Enhanced Representation.** Compared with the model removed Entity-Enhanced Representation module, our method PAEE achieves 1.09% improvements

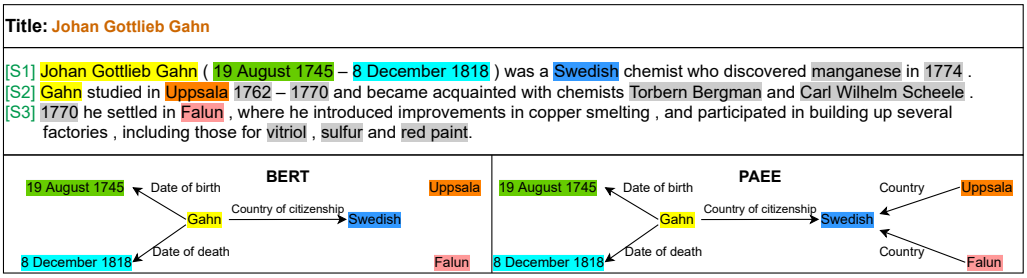

Fig. 3: Case study on our proposed PAEE and baseline model. Entity mentions only involved in these relation instances are colored, other entities in the document are high-lighted in grey. We utilize arrows to connect relational entity pairs.

Model	ACC	F1
$\text{BERT}_{\text{base}}$	48.83	54.16
CorefBERT	59.37	57.51
ATLOP	65.42	61.09
PAEE-$\text{BERT}_{\text{base}}$ (Our)	70.30 (↑4.88)	64.82 (↑3.73)

Table 6: The ACC means the accuracy of identifying relational entity pairs.

of F1 score on the DocRED dataset. It demonstrates that the EER module is able to effectively model the directivity of entity pairs.

(3) **Effectiveness of Sliding Window strategy.** Removing the SW, the performance drops significantly. Specifically, the F1 score drops from 62.62% to 61.72% on the DocRED dataset.

(4) **Effectiveness of Potential Pair Prediction.** When we remove the PPP module, the F1 score drops from 62.62% to 61.52%. It indicates that the performance of the model can be effectively improved by predicting potential relational entity pairs.

3.4 Discussion and Analysis

In order to explore whether the performance bottleneck of the model is effectively solved, we utilize experiments to analyze it.

The effect of PAEE on sparsity. To assess the effectiveness of PAEE on identifying relational entity pairs, we analyze it from contrast experiments, the experiments are based on the pre-training model $\text{BERT}_{\text{base}}$ and DocRED dataset. As show in Table 6, the ACC score if 70.3% which is 4.88% more than previous SOTA model ATLOP. This shows that PAEE model can effectively identify potential relational entity pairs.

The effect of Entity-Enhanced Representation(EER). To show that our EER can model the representation of entity pairs better, we divide the documents in dev set of DocRED into different groups by the proportion of relational entity pairs, and evaluate models trained with or without the EER. Experiment results are shown in Figure 4. We observe that for both models, their performance gets better when the proportion of relational entity pairs becomes larger, and the model w/ EER consistently outperforms the model w/o EER. This demonstrates that EER can model the representation of entity pairs better.

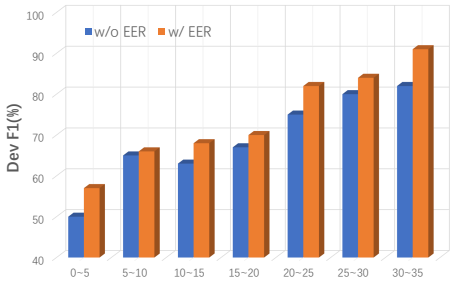

Fig. 4: Dev $F1$ score on DocRED. The x-axis refers to the proportion of relational entity pairs per document (Unit: %), the y-axis refers to the dev F1.

3.5 Case Study

We select a sample from the dev set of the DocRED dataset and conduct a case study to further illustrate the effectiveness of our model PAEE compared with the baseline. As shown in Figure 3, we notice that both $\text{BERT}_{\text{base}}$ and PAEE-$\text{BERT}_{\text{base}}$ can successfully extract the "$Country\ of\ citizenship$" relation between "$Gahn$" and "$Swedish$". However, only our PAEE-$\text{BERT}_{\text{base}}$ can deduce that the "$Country$" of "$Uppsala$" and "$Falun$" are same, namely "$Swedish$".

4 Conclusion

In this paper, we propose the **P**air-**A**ware and **E**ntity-**E**nhanced (PAEE) model. Specifically,

PAEE introduces Pair-Aware Representation(PAR) module to alleviate the negative impact of sparsity, which constrains the following relation extraction to the predicted entity pairs subset rather than all pairs. In addition, PAEE also designs Entity-Enhanced Representation(EER) module to assemble directional entity pairs and obtain holistic understanding of document. Experiments on four benchmark datasets DocRED, DWIE, CDA and GDA, show that PAEE outperforms the previous methods and obtains new state-of-the-art results.

5 Acknowledgements

This work is supported by the National Key Research and Development Program of China (No.2020AAA0106400), the National Natural Science Foundation of China (No.61922085, 61976211, 62176257). This work is also supported by the Strategic Priority Research Program of Chinese Academy of Sciences (Grant No.XDA27020200), the Youth Innovation Promotion Association CAS, and Yunnan Provincial Major Science and Technology Special Plan Projects (No.202202AD080004).

References

Iz Beltagy, Kyle Lo, and Arman Cohan. 2019. Scibert: A pretrained language model for scientific text. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 3615–3620.

Fenia Christopoulou, Makoto Miwa, and Sophia Ananiadou. 2019. Connecting the dots: Document-level neural relation extraction with edge-oriented graphs. *arXiv preprint arXiv:1909.00228*.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, Minneapolis, Minnesota. Association for Computational Linguistics.

Priya Goyal, Piotr Dollár, Ross Girshick, Pieter Noordhuis, Lukasz Wesolowski, Aapo Kyrola, Andrew Tulloch, Yangqing Jia, and Kaiming He. 2018. Accurate, large minibatch sgd: Training imagenet in 1 hour.

Bo Li, Wei Ye, Zhonghao Sheng, Rui Xie, Xiangyu Xi, and Shikun Zhang. 2020. Graph enhanced dual attention network for document-level relation extraction. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 1551–1560.

Jiao Li, Yueping Sun, Robin J Johnson, Daniela Sciaky, Chih-Hsuan Wei, Robert Leaman, Allan Peter Davis, Carolyn J Mattingly, Thomas C Wiegers, and Zhiyong Lu. 2016. Biocreative v cdr task corpus: a resource for chemical disease relation extraction. *Database*, 2016.

Min Lin, Qiang Chen, and Shuicheng Yan. 2013. Network in network. *arXiv preprint arXiv:1312.4400*.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Roberta: A robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692*.

Ilya Loshchilov and Frank Hutter. 2018. Decoupled weight decay regularization. In *International Conference on Learning Representations*.

Paulius Micikevicius, Sharan Narang, Jonah Alben, Gregory Diamos, Erich Elsen, David Garcia, Boris Ginsburg, Michael Houston, Oleksii Kuchaiev, Ganesh Venkatesh, et al. 2018. Mixed precision training. In *International Conference on Learning Representations*.

Guoshun Nan, Zhijiang Guo, Ivan Sekulić, and Wei Lu. 2020. Reasoning with latent structure refinement for document-level relation extraction. *arXiv preprint arXiv:2005.06312*.

Athanasios Papadopoulos, Pawel Korus, and Nasir Memon. 2021. Hard-attention for scalable image classification. *Advances in Neural Information Processing Systems*, 34.

Olaf Ronneberger, Philipp Fischer, and Thomas Brox. 2015. U-net: Convolutional networks for biomedical image segmentation. In *International Conference on Medical image computing and computer-assisted intervention*, pages 234–241. Springer.

Peng Shi and Jimmy Lin. 2019. Simple bert models for relation extraction and semantic role labeling. *arXiv preprint arXiv:1904.05255*.

Livio Baldini Soares, Nicholas Fitzgerald, Jeffrey Ling, and Tom Kwiatkowski. 2019. Matching the blanks: Distributional similarity for relation learning. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 2895–2905.

Yifan Sun, Changmao Cheng, Yuhan Zhang, Chi Zhang, Liang Zheng, Zhongdao Wang, and Yichen Wei. 2020. Circle loss: A unified perspective of pair similarity optimization. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 6398–6407.

Hengzhu Tang, Yanan Cao, Zhenyu Zhang, Jiangxia Cao, Fang Fang, Shi Wang, and Pengfei Yin. 2020. Hin: Hierarchical inference network for document-level relation extraction. *Advances in Knowledge Discovery and Data Mining*, 12084:197.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in neural information processing systems*, pages 5998–6008.

Patrick Verga, Emma Strubell, and Andrew McCallum. 2018. Simultaneously self-attending to all mentions for full-abstract biological relation extraction. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 872–884.

Difeng Wang, Wei Hu, Ermei Cao, and Weijian Sun. 2020. Global-to-local neural networks for document-level relation extraction. *arXiv preprint arXiv:2009.10359*.

Hong Wang, Christfried Focke, Rob Sylvester, Nilesh Mishra, and William Wang. 2019a. Fine-tune bert for docred with two-step process. *arXiv preprint arXiv:1909.11898*.

Hong Wang, Christfried Focke, Rob Sylvester, Nilesh Mishra, and William Wang. 2019b. Fine-tune bert for docred with two-step process.

Ye Wu, Ruibang Luo, Henry CM Leung, Hing-Fung Ting, and Tak-Wah Lam. 2019. Renet: A deep learning approach for extracting gene-disease associations from literature. In *International Conference on Research in Computational Molecular Biology*, pages 272–284. Springer.

Benfeng Xu, Quan Wang, Yajuan Lyu, Yong Zhu, and Zhendong Mao. 2021a. Entity structure within and throughout: Modeling mention dependencies for document-level relation extraction. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 35, pages 14149–14157.

Wang Xu, Kehai Chen, and Tiejun Zhao. 2021b. Document-level relation extraction with reconstruction. In *The 35th AAAI Conference on Artificial Intelligence (AAAI-21)*.

Yuan Yao, Deming Ye, Peng Li, Xu Han, Yankai Lin, Zhenghao Liu, Zhiyuan Liu, Lixin Huang, Jie Zhou, and Maosong Sun. 2019. Docred: A large-scale document-level relation extraction dataset. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 764–777.

Deming Ye, Yankai Lin, Jiaju Du, Zhenghao Liu, Peng Li, Maosong Sun, and Zhiyuan Liu. 2020. Coreferential reasoning learning for language representation. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 7170–7186.

Klim Zaporojets, Johannes Deleu, Chris Develder, and Thomas Demeester. 2021. Dwie: an entity-centric dataset for multi-task document-level information extraction.

Shuang Zeng, Runxin Xu, Baobao Chang, and Lei Li. 2020. Double graph based reasoning for document-level relation extraction. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1630–1640.

Ningyu Zhang, Xiang Chen, Xin Xie, Shumin Deng, Chuanqi Tan, Mosha Chen, Fei Huang, Luo Si, and Huajun Chen. 2021. Document-level relation extraction as semantic segmentation. *arXiv preprint arXiv:2106.03618*.

Yuhao Zhang, Victor Zhong, Danqi Chen, Gabor Angeli, and Christopher D Manning. 2017. Position-aware attention and supervised data improve slot filling. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 35–45.

Zhenyu Zhang, Bowen Yu, Xiaobo Shu, Tingwen Liu, Hengzhu Tang, Wang Yubin, and Li Guo. 2020. Document-level relation extraction with dual-tier heterogeneous graph. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 1630–1641.

Wenxuan Zhou, Kevin Huang, Tengyu Ma, and Jing Huang. 2021. Document-level relation extraction with adaptive thresholding and localized context pooling. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 35, pages 14612–14620.

A Appendix

A.1 Training Strategy

In the **Relationship Classification** stage, previous work (Wang et al., 2019b) observed that there is an imbalance relation distribution for RE (the relational entity pairs are sparse). To alleviate the negative impact of sparsity, Zhang et al. (2021) introduces a balanced softmax method inspired by the circle loss (Sun et al., 2020). Based on this, we design **Adaptive Softmax** loss, which introduces a addition threshold class TH, which is automatically learned in the same way as other classes. The class TH aims to separate positive classes and negative classes, hoping that the scores of the target category are all greater than s_{TH} and the scores of the non-target categories are all less than s_{TH}. Formally,

$$L_{rel} = \log(e^{s_{TH}} + \sum_{i \in \omega_{neg}} e^{s_i}) + \log(e^{-s_{TH}} + \sum_{i \in \omega_{pos}} e^{-s_i}) \tag{9}$$

In the **Potential Pair Prediction** stage, in order to match binary classification task, we design the loss as:

$$L_{pot} = -\frac{1}{n_p} \sum_{i=1}^{n_p} (y_i \log \mathbf{P}_{pair} + (1 - y_i) \log(1 - \mathbf{P}_{pair})) \tag{10}$$

where n_p is the size of full entity pairs set. In the **Cross Matching** stage, to capture the features of entities as subject and object respectively, we design the loss as:

$$L_{sub} = -\frac{1}{n_c n_e} \sum_{j=1}^{n_e} \sum_{i=1}^{n_c} (y_i \log F^j_{sub} + (1 - y_i) \log(1 - F^j_{sub})),$$

$$L_{obj} = -\frac{1}{n_c n_e} \sum_{j=1}^{n_e} \sum_{i=1}^{n_c} (y_i \log F^j_{obj} + (1 - y_i) \log(1 - F^j_{obj})) \tag{11}$$

where n_c is the size of full relation set, n_e is size of full entities set. The total loss is the sum of the above losses:

$$L_{total} = \alpha L_{rel} + \beta L_{pot} + \gamma \frac{L_{sub} + L_{obj}}{2} \tag{12}$$

Performance might be better by carefully tuning the weight of each sub-loss, but we just assign equal weights for simplicity (ie., $\alpha = \beta = \gamma = 1$).

Improving Zero-Shot Entity Linking Candidate Generation with Ultra-Fine Entity Type Information

Xuhui Sui[1], **Ying Zhang**[1*], **Kehui Song**[1], **Baohang Zhou**[1],
Guoqing Zhao[2], **Xin Wei**[2], **Xiaojie Yuan**[1]
[1] College of Computer Science, TKLNDST, Nankai University, China
[2] Mashang Consumer Finanace Co, Ltd
{suixuhui,zhangying,songkehui,zhoubaohang}@dbis.nankai.edu.cn
{guoqing.zhao02,xin.wei02}@msxf.com, yuanxj@nankai.edu.cn

Abstract

Entity linking, which aims at aligning ambiguous entity mentions to their referent entities in a knowledge base, plays a key role in multiple natural language processing tasks. Recently, zero-shot entity linking task has become a research hotspot, which links mentions to unseen entities to challenge the generalization ability. For this task, the training set and test set are from different domains, and thus entity linking models tend to be overfitting due to the tendency of memorizing the properties of entities that appear frequently in the training set. We argue that general ultra-fine-grained type information can help the linking models to learn contextual commonality and improve their generalization ability to tackle the overfitting problem. However, in the zero-shot entity linking setting, any type information is not available and entities are only identified by textual descriptions. Thus, we first extract the ultra-fine type information from the entity textual descriptions. Then, we propose a hierarchical multi-task model to improve the high-level zero-shot entity linking candidate generation task by utilizing the entity typing task as an auxiliary low-level task, which introduces extracted ultra-fine type information into the candidate generation task. Experimental results demonstrate the effectiveness of utilizing the ultra-fine entity type information and our proposed method achieves state-of-the-art performance.

1 Introduction

Entity linking (EL) is the task of assigning entity mentions in a text to corresponding entity records in a reference knowledge base. EL plays a key role in the language understanding pipeline, underlying a variety of downstream applications, such as information extraction (Hoffmann et al., 2011; Ji and Nothman, 2016), semantic search (Blanco et al., 2015) and question answering (Berant et al., 2013; Yih et al., 2015; Welbl et al., 2018). In general, EL consists of two phases: candidate generation which generates a set of candidates for each mention from millions of entities, and candidate ranking which retrieves the matched entity for each mention from the set of candidates. As the final results in EL are only generated from candidta sets, the accuracy of the whole EL task is limited by the candidate generation phase. Therefore, in this paper, we focus on the candidate generation phase to set a higher upper bound on EL accuracy.

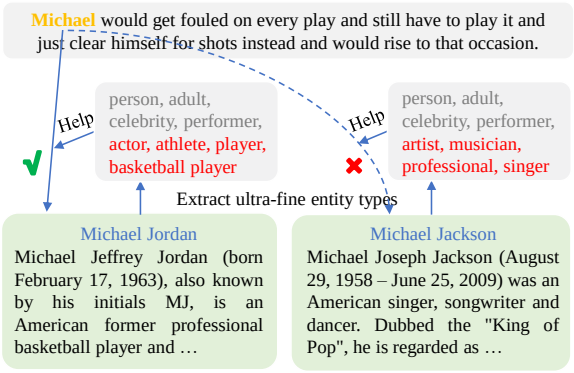

Figure 1: Examples of entity linking with general ultra-fine-grained entity type information. Different ultra-fine-grained types of the two entities are denoted in red.

Traditional EL approaches usually train models under the setting that linked entities in the test set are available in the training set. However, in many real-world scenarios, labeled data are not easily obtained in multiple domains. Thus, there is a need for EL models to have the capability of generalizing to new domains and new entities. To challenge the generalization ability, a zero-shot entity linking task (Logeswaran et al., 2019) has been proposed, where mentions need to be linked to unseen entities and only the textual information is available. For this task, the training and test sets share different distributions of entities, and thus entity linking models tend to be overfitting due to the tendency of memorizing the properties of entities that appear frequently in the training set.

*Corresponding author.

Proceedings of the 29th International Conference on Computational Linguistics, pages 2429–2437
October 12–17, 2022.

We argue that general ultra-fine-grained type information can help the linking models learn contextual commonality and improve their generalization ability to tackle the overfitting problem. If a linking model learns the contextual commonality of *athlete* related entities, it can use similar contextual information to correctly select entities of the same type. Examples of entity linking with general ultra-fine-grained entity type information are shown in Figure 1. A key observation is that the given ultra-fine entity types could have positive effect on the entity linking task. In this example, the type information can help the linking model link the *Michael* in the text to *Michael Jordan* [*actor, athlete, player, basketball player*] instead of *Michael Jackson* [*artist, musician, professional, singer*].

Therefore, in this paper, we try to introduce ultra-fine entity type information into the zero-shot entity linking candidate generation. However, in the zero-shot entity linking setting, any type information is not available and entities are only identified by textual descriptions. Thus, we first extract ultra-fine types from textual descriptions of each entity. In general, more fine-grained entity type information can better help the linking models learn contextual commonality and improve their generalization ability. Thus, we train an entity typing model by utilizing the Ultra-fine Entity Typing dataset (Choi et al., 2018) whose types are more fine-grained rather than other fine-grained entity typing datasets (e.g. FIGER (Ling and Weld, 2012) and OntoNotes (Gillick et al., 2014)), and use the model to extract ultra-fine types from each entity textual description. Then, we propose a hierarchical multi-task model, which jointly models the candidate generation task and ultra-fine entity typing task to introduce the type information extracted by the trained typing model into the candidate generation phase. The ultra-fine entity typing task is utilized as an auxiliary low-level task, providing corresponding type features for the high-level candidate generation task. Our primary motivation is to discover helpful training signals from ultra-fine-grained type information to ensure a more robust zero-shot entity linking candidate generation model.

To summarize, our major contributions are shown as follows:

- To the best of our knowledge, this work is the first to introduce fine-grained type information into zero-shot entity linking task. The fine-grained type information can help the linking models learn contextual commonality and improve their ability to generalize to new domains and unseen entities.
- We first extract ultra-fine entity types for each entity, without depending on additional manually annotated data. Then to take full advantage of extracted type information, we present a hierarchical multi-task model to improve the high-level zero-shot entity linking candidate generation task by utilizing the entity typing task as an auxiliary low-level task.
- Experimental results demonstrate the effectiveness of utilizing the ultra-fine entity type information and our proposed method achieves state-of-the-art performance.

2 Related Work

2.1 Zero-shot Entity Linking

Zero-shot entity linking (Logeswaran et al., 2019) has attracted significant interest from researchers in recent years. In this task, no mentions or entities in the test set have been observed during training and only descriptions of each entity are provided. It consists of two phases: candidate generation (Wu et al., 2020; Ristoski et al., 2021) and candidate ranking (Yao et al., 2020; Tang et al., 2021). In this paper, we focus on the candidate generation phase. (Logeswaran et al., 2019) is the first to formally propose the zero-shot entity linking task and use a traditional IR approach BM25 to generate candidates. BLINK (Wu et al., 2020) uses a bi-encoder architecture to encode mentions and descriptions of entities into dense space to generate candidates, which achieves state-of-the-art results. KG-ZESHEL (Ristoski et al., 2021) utilizes a knowledge graph to extend BLINK. Our work also extends BLINK by introducing auxiliary ultra-fine type information without depending on additional manually annotated data to improve the candidate generation task in zero-shot entity linking.

2.2 Entity Linking with Type Information

Entity typing refers to the act of assigning semantic types to mentions in the text. Fine-grained entity type information has been proven effective in the entity linking process. (Gupta et al., 2017) explores fine-grained entity typing for cross-domain entity linking. (Raiman and Raiman, 2018) proposes a type system to constrain the space in which mentions can be linked. (Onoe and Durrett, 2020)

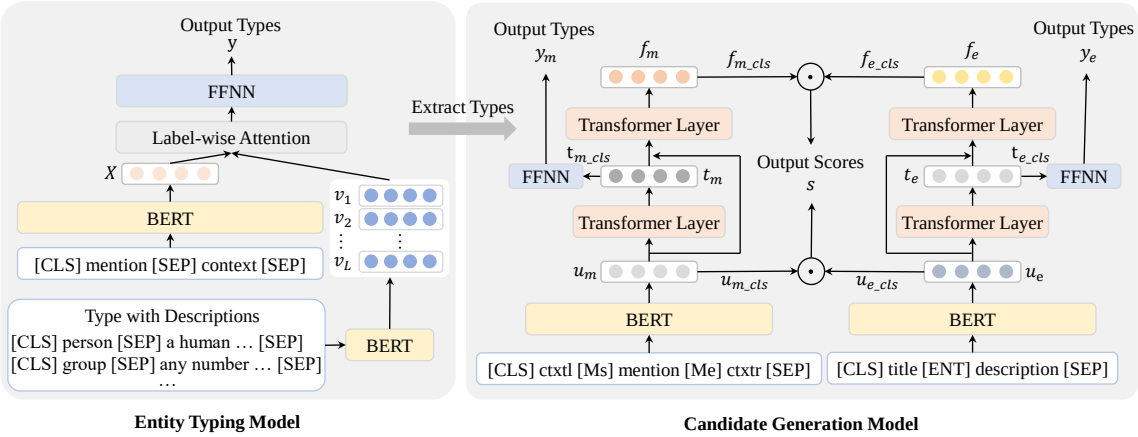

Figure 2: The overall architecture of our proposed model. It consists of two parts: an entity typing model to extract ultra-fine entity types for each entity and a hierarchical multi-task candidate generation model to generate candidate entities with the extracted type information.

converts the cross-domain entity linking task to a very fine-grained entity typing task to generalize across domains effectively. (Hou et al., 2020; Chen et al., 2020) create the semantic embedding for each entity by aggregating entity type embeddings. Inspired by these previous works, to the best of our knowledge, our work is the first to introduce fine-grained type information into zero-shot entity linking task. Also, considering the generalization ability challenge of the zero-shot entity linking task, inspired by (Sanh et al., 2019; Wiatrak and Iso-Sipilä, 2020), our proposed method introduces fine-grained type information in a hierarchical multi-task way to learn contextual commonality and improve the generalization ability.

3 Model

Figure 2 shows our proposed two-stage model, which consists of two parts: the entity typing model and the candidate generation model. In this section, we describe these two models in detail.

3.1 Entity Typing Model

The Entity Typing Model in Figure 2 presents the first part of our proposed model. The goal of this model is to extract ultra-fine entity types from textual description for each entity. Considering off-the-shelf entity typing models (e.g. (Onoe and Durrett, 2019; Onoe et al., 2021)) do not significantly outperform the BERT-based model (Onoe and Durrett, 2019), we simply modify the BERT-based model as our entity typing model.[1] In general, more fine-grained entity type information can better help the linking models learn contextual commonality and improve their generalization ability. Thus, we train the model by utilizing the Ultra-fine Entity Typing dataset (Choi et al., 2018). Then we use the model to extract ultra-fine type information for zero-shot entity linking dataset (Logeswaran et al., 2019). This process does not require any additional manually annotated data.

3.1.1 Label-wise Feature Extraction

Given a mention and its context, we input them to BERT (Devlin et al., 2019) as a sequence pair together with special start and separator tokens ([CLS] mention [SEP] context [SEP]) to extract features. It produces a matrix representation $X = [x_1, x_2, ..., x_n]$ to represent the mention-context pair, where $x_i \in \mathbb{R}^d$ is the word embedding vector for the i-th word, n is the length of the input pair and d is the dimension of hidden states of BERT.

Each entity type has a textual description. Considering that the model training and inference process are on different datasets, we make full use of the type descriptions to improve the generalization ability of the model. To utilize the type description, we input the type and its description to BERT in the form of [CLS] type [SEP] description [SEP]. The embeddings of all the possible labels $C = \{c_1, c_2, ..., c_L\}$ are represented to $V = [v_1, v_2, ..., v_L]$, where $v_i \in \mathbb{R}^d$ is the last hidden layer corresponding to the position of the [CLS] token for the i-th type. Note that the weights of the label embeddings V are fixed after being extracted.

A label-wise attention is utilized to learn individ-

[1] We leave the construction of a more effective entity typing model to future work.

ual representation for each label. The compatibility of label-word pairs is measured as follows:

$$G = VX^T$$

where $G \in \mathbb{R}^{L \times n}$. The attention scores for all the words of mention-context pair with regard to the l-th label are computed via the SoftMax function:

$$a_l = \text{SoftMax}(G_l)$$

where $G_l \in \mathbb{R}^n$ is the compatibility vector of each word and the l-th label. Intuitively, a_l extracts the most relevant information in X about the label l by using attention. Eventually, the label-wise representation is obtained by the weighted aggregation with the attention scores:

$$z_l = \sum_{i=1}^{n} a_{l_i} x_i$$

3.1.2 Multi-label Classification

For the l-th type, the binary prediction $\hat{y}_l$ is computed by: $\hat{y}_l = \text{FFNN}(z_l; \theta_{F_1})$. Each mention is usually associated with a set of types, and a multi-label training objective is required. Thus, the loss is a sum of binary cross-entropy losses over all types over all examples. Finally, we optimize a multi-label binary cross entropy objective:

$$\mathcal{L}_{type} = -\sum_{l=1}^{L} y_l \log \hat{y}_l + (1 - y_l) \log(1 - \hat{y}_l)$$

where y_l takes the value 1 if the l-th type applies to the current mention.

3.2 Candidate Generation Model

After training the Entity Typing Model, we use the model to extract ultra-fine entity types for each entity in the zero-shot entity linking dataset. Our candidate generation model extends BLINK (Wu et al., 2020) bi-encoder by introducing the ultra-fine type information and is shown in Figure 2 Candidate Generation Model, which utilizes a hierarchical multi-task way to jointly learn candidate generation task and ultra-fine entity typing task. The ultra-fine entity typing task is utilized as an auxiliary low-level task at the bottom layer, providing corresponding type features for the high-level candidate generation task at the top layer.

3.2.1 Feature Extraction

Followed (Wu et al., 2020), we use BERT (Devlin et al., 2019) to encode textual input of mentions and entities. The input of each mention $\mathcal{T}_m$ is constructed as follows:

[CLS] ctxtl [Ms] mention [Me] ctxtr [SEP]

where mention, ctxtl, ctxtr are the word-pieces tokens of the mention, context before and after the mention respectively, and [Ms], [Me] are special tokens to tag the mention. The input of each entity $\mathcal{T}_e$ is construct as follows:

[CLS] title [ENT] description [SEP]

where title, description are word-pieces tokens of entity title and description, and [ENT] is a special token to separate the entity title and its description.

Both the context and candidate entity are input to two independent BERT models and are encoded into vectors:

$$u_m = \text{BERT}(\mathcal{T}_m; \theta_{\text{BERT}_1}),\ u_e = \text{BERT}(\mathcal{T}_e; \theta_{\text{BERT}_2})$$

3.2.2 Low-level Entity Typing

The low-level task in our candidate generation model is the ultra-fine entity typing task. We follow (Zhu et al., 2020) to use a binary pairwise relation constraint between mention and each candidate entity. Briefly, a mention and its corresponding entity should share the same type distribution. Thus, the corresponding entity's ground truth types can also be the ground truth types of the mention. We utilize the extracted ultra-fine types from entity descriptions as the target labels of both mention type prediction and entity type prediction tasks. We use two independent transformer layers T as task specific encoders, which takes the extracted features u_m and u_e as input and outputs representations denoted as t_m and t_e:

$$t_m = T(u_m; \theta_{T_1}),\ t_e = T(u_e; \theta_{T_2})$$

Our mention and entity type prediction are similar to the Multi-label Classification in section 3.1.2. For each training pair (m_i, e_i), to predict the l-th type, the binary prediction $\hat{y}^l_{m_i}$ of the mention and $\hat{y}^l_{e_i}$ of the entity are computed respectively by: $\hat{y}^l_{m_i} = \text{FFNN}(t_{m\_cls_i}; \theta_{F_2})$, $\hat{y}^l_{e_i} = \text{FFNN}(t_{e\_cls_i}; \theta_{F_3})$, where $t_{m\_cls_i}$ and $t_{e\_cls_i}$ are the representations corresponding to the position of the [CLS] token of t_{m_i} and t_{e_i} respectively. The

losses of mention type prediction and entity type prediction are calculated as follows:

$$\mathcal{L}_{type\_m} = -\sum_{l=1}^{L} y^l \log \hat{y}^l_{m_i} + (1-y^l)\log(1-\hat{y}^l_{m_i})$$

$$\mathcal{L}_{type\_e} = -\sum_{l=1}^{L} y^l \log \hat{y}^l_{e_i} + (1-y^l)\log(1-\hat{y}^l_{e_i})$$

where y^l takes the value 1 if the l-th type is extracted from the description of the entity e_i, and e_i is the corresponding entity of the mention m_i.

3.2.3 High-level Candidate Generation

The high-level task in our candidate generation model is the zero-shot entity linking candidate generation task. It takes the average of the extracted features by BERT u_m, u_e and the low-level task encoder specific output t_m, t_e as the input. We utilizes another two independent transformer layers T as the task-specific encoders:

$$f_m = T(\frac{1}{2}(u_m + t_m); \theta_{T_3}),\ f_e = T(\frac{1}{2}(u_e + t_e); \theta_{T_4})$$

Finally, the score for a given mention m_i and a candidate entity e_i is calculated as the dot product of the corresponding vectors:

$$s(m_i, e_i) = f_{m\_cls_i} \cdot f_{e\_cls_i} + u_{m\_cls_i} \cdot u_{e\_cls_i}$$

where $f_{m\_cls_i}$, $f_{e\_cls_i}$, $u_{m\_cls_i}$ and $u_{e\_cls_i}$ are the representations corresponding to the position of the [CLS] token of f_{m_i}, f_{e_i}, u_{m_i} and u_{e_i} respectively.

Following (Wu et al., 2020), our model is trained on in-batch negatives. Within a batch, the corresponding entity of the mention is the positive sample while other entities in the batch are all negative samples of the mention. Thus, for the candidate scoring, the loss needs to maximize the score of the corresponding entity of the mention in the batch with respect to the other entities of the same batch. To achieve this, for each training pair (m_i, e_i) in a batch of B pairs, the loss is computed as:

$$\mathcal{L}_{(m_i,e_i)} = -s(m_i, e_i) + \log \sum_{j=1}^{B} \exp(s(m_i, e_j))$$

where e_i is the gold entity of the mention m_i.

3.2.4 Hierarchical Multi-task Training

Our model incorporates three objectives, one for candidate scoring and the others for the mention type prediction and the candidate entity type prediction. We jointly optimize these three objectives during our training process. The final loss of the candidate generation model is calculated as follows:

$$\mathcal{L} = \mathcal{L}_{(m_i,e_i)} + \mathcal{L}_{type\_m} + \mathcal{L}_{type\_e}$$

Set	World	Entities	Mentions
Training	American Football	31929	3898
	Doctor Who	40281	8334
	Fallout	16992	3286
	Final Fantasy	14044	6041
	Military	104520	13063
	Pro Wrestling	10133	1392
	StarWars	87056	11824
	World of Warcraft	27677	1437
Validation	Coronation Street	17809	1464
	Muppets	21344	2028
	Ice Hockey	28684	2233
	Elder Scrolls	21712	4275
Test	Forgotten Realms	15603	1200
	Lego	10076	1199
	Star Trek	34430	4227
	YuGiOh	10031	3374

Table 1: Overall statistics of the zero-shot entity linking dataset.

4 Experiments

In this section, we compare our proposed method to other state-of-the-art methods to demonstrate the effectiveness of our model. We first introduce the datasets we used and the implementation details of our model. Then we briefly introduce the baselines and present the overall performance of our model in comparison with others.

4.1 Datasets

We train the entity typing model on the Ultra-Fine Entity Typing dataset (Choi et al., 2018), which has 10331 labels and most of them are defined as free-form text phrases. Each type is marked as one of the three classes: *coarse, fine*, and *ultra-fine*. Note that this classification does not provide explicit hierarchies in the types, and all classes are treated equally during training.

We conduct our experiments mainly on the zero-shot entity linking dataset [2], which is proposed by (Logeswaran et al., 2019) and built using the documents on Wikia [3]. Table 1 shows the overall statistics of the dataset. In this dataset, the entities in the validation and test sets are from different

[2] https://github.com/lajanugen/zeshel
[3] https://www.wikia.com

Model	Forgotten Realms	Lego	Star Trek	YuGiOh	Macro Recall@64	Micro Recall@64
BM25	83.33	81.23	65.89	60.85	72.83	69.13
BLINK (base)*	90.67	89.99	82.45	71.40	83.63	80.61
BLINK (large)	–	–	–	–	–	82.06
BLINK (large)*	90.92	90.58	84.03	73.30	84.71	82.02
KG-ZESHEL	–	–	–	–	–	82.44
KG-ZESHEL*	91.25	90.40	84.43	73.77	84.96	82.35
Ours (base)	92.08	90.74	83.94	72.00	84.69	81.90
Ours (large)	**92.83**	**91.66**	**85.38**	**74.78**	**86.16**	**83.45**

Table 2: Recall@64 results on the test domains of the zero-shot entity linking dataset. Macro Recall@64 represents the average Recall@64 score of these four test domains. Micro Recall@64 represents the weighted average Recall@64 score of these four domains. * indicates the models are reproduced according to the implementation details in their papers and released codes for a more detailed analysis. We use (base) and (large) to indicate the version of the underlying pre-trained BERT model is BERT-base and BERT-large, respectively. All scores are averaged 5 runs using different random seeds, and our results over all baselines are statistically significant with $p < 0.05$ with the t-test. In the results, the highest values are in bold and the underlined ones are the second highest.

domains compared to the training set, allowing the performance evaluation on entire unseen entities. It uses 8 domains for training, 4 for validation, and 4 for test. The training set has 49,275 labeled mentions while the validation and test sets both have 10,000 unseen mentions.

The samples of this dataset are categorized into 4 categories by (Logeswaran et al., 2019), which are *High Overlap (HO)* whose mention string is identical to its gold entity title, *Multiple Categories (MC)* whose gold entity title is followed by a disambiguation phrase, *Ambiguous substring (AS)* whose mention string is a substring of its gold entity title, and *Low Overlap (LO)* are other mentions. According to the statistics of (Logeswaran et al., 2019), 5% of mentions are categorized as HO, 28% of mentions are MC, 8% of mentions belong to AS, and 59% of mentions are categorized as LO.

4.2 Implementation Details

In our experiments, we use BERT (Devlin et al., 2019) as our base model. The evaluation metric is the recall. We perform our experiments with 5 random seeds and report the average results. And we perform the t-test to demonstrate the statistical significance of our results.

For the entity typing model, the BERT we used to extract mention-context and type description is both the bert-base-uncased (Devlin et al., 2019). We set the maximum sequence length of the input text of mention-context and type description to be 128 and 80, respectively. In this setting, all tokens are covered. The batch size is 32, and the learning rate is 2e-5 with a linear learning rate decay schedule. We use ADAM (Kingma and Ba, 2015) optimization algorithm to optimize our model.

For the candidate generation model, followed (Wu et al., 2020), we use the bert-base-uncased and the bert-large-uncased (Devlin et al., 2019) models respectively. The maximum sequence length of the mention and entity is set to 128. The learning rate is 1e-5 and the batch size is 128. We also train the model by utilizing the ADAM. Our experimental code is available here [4].

4.3 Baselines

For the quantitative evaluation of our proposed model, we use the following state-of-the-art baseline methods for comparison. The first method is BM25 (Robertson and Zaragoza, 2009), which is a traditional IR approach and used by (Logeswaran et al., 2019). The second method is BLINK (Wu et al., 2020), which uses a bi-encoder architecture to encode mentions and entity descriptions into dense space to generate candidates. Our proposed model extends BLINK bi-encoder by introducing ultra-fine entity typing. Comparison results to BLINK could also be regarded as the ablation study to justify the advantage of our proposed model. The last method is KG-ZESHEL (Ristoski et al., 2021), which extends BLINK bi-encoder by utilizing a knowledge graph. Note that we reproduced the BLINK model and the KG-ZESHEL model for a more detailed analysis according to the implementation details in their papers and released codes.

4.4 Overall Performance

The recall@64 results for the candidate generation on the test domains of the zero-shot entity link-

[4] https://github.com/suixuhui/ETZEL

ing dataset are shown in Table 2. We can observe that our proposed model outperforms all baseline models in all test domains and on average. This is consistent with our main claim that our model can improve the performance of the zero-shot entity linking candidate generation task by utilizing the extracted ultra-fine entity type information. Our proposed method utilizes a hierarchical multi-task way to fully mine useful training signals from the low-level ultra-fine entity typing task to help the entity linking candidate generation models learn contextual commonality and improve their generalization ability. This is significant for the zero-shot entity linking setting to generalize the models to new domains and link unseen entities.

We observe that the bi-encoder-based methods perform better than the traditional IR approach BM25, which indicates the effectiveness of our chosen base model. The bi-encoder method achieves state-of-the-art results in the candidate generation task. This approach also allows fast, real-time inference, as the candidate representations can be cached. It can be expected that the BERT-large-based models work better than the BERT-base-based models, due to the larger pre-trained model which encodes more general knowledge. Despite this, we still find that our proposed model with the BERT-base version performs better than the baselines with the BERT-large version in some test domains (e.g., Forgotten Realms and Lego), which further indicates the effectiveness of our proposed method. In general, our proposed method outperforms the baseline approaches for 1.20% and 1.01% on Macro Recall@64 and Micro Recall@64 on the test set, respectively.

5 Analysis

5.1 Top-k Results

The micro recall@k results of the bi-encoder-based models (BLINK, KG-ZESHEL and our proposed model) for the candidate generation task on the test set are shown in Figure 3. We can find that our proposed model consistently outperforms the baseline methods for all k values. The improvement of our model compared to BLINK is pronounced, while the KG-ZESHEL slightly improves the performance of BLINK. This demonstrates the effectiveness of our proposed method. It can also be observed that our proposed model has a relatively significant improvement in the first few candidates (e.g., recall@1, recall@4) over BLINK, allowing for an improvement of more than 3%. This indicates the effectiveness of utilizing extracted ultra-fine entity type information and shows that the ultra-fine entity typing task has strong positive effect on the candidate generation task of zero-shot entity linking. However, as the number of candidates increases, the improvement becomes less and this phenomenon is foreseeable, because there is a tendency for all models to saturate as the number of candidates continues to increase.

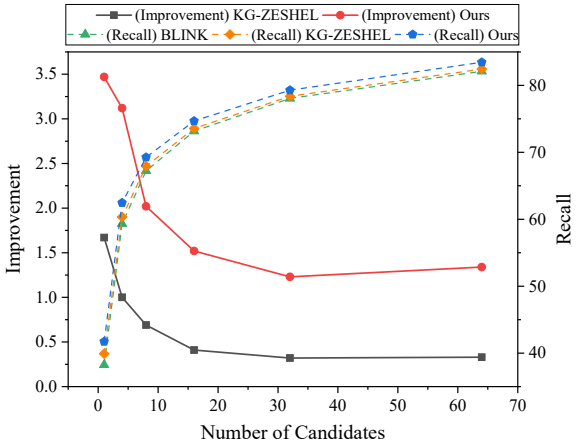

Figure 3: Top-k entity linking recall on test set of zero-shot entity linking dataset. Dashed lines indicate the recall of the three methods and solid lines indicate the relative improvement of our model and KG-ZESHEL compared to the BLINK. The results we choose to report are micro recall@1, recall@4, recall@8, recall@16, recall@32 and recall@64.

.

Model	HO	MC	AS	LO
BM25	**99.28**	72.54	88.03	54.37
BLINK	98.92	91.04	97.06	74.24
KG-ZESHEL	99.04	91.41	**97.69**	74.52
Ours	99.04	**93.69**	97.27	**75.42**

Table 3: Micro recall@64 scores on the category-specific test subsets including High Overlap (HO), Multiple Categories (MC), Ambiguous Substring (AS) and Low Overlap (LO).

5.2 Results on Category-specific

In addition, we analyze the results of zero-shot entity linking candidate generation models on different categories. There are four categories, and the details about these categories have been described in section 4.1. Table 3 shows the micro recall@64 scores on the category-specific test subsets of the zero-shot entity linking dataset. We find that BM25 outperforms all other methods including our pro-

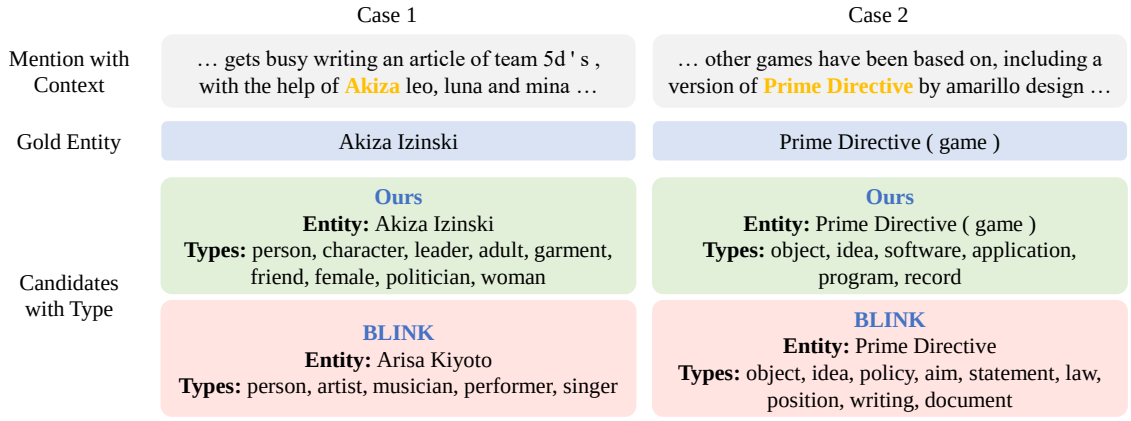

Figure 4: Examples of the compared candidate generation results of our proposed model and the baseline model (BLINK), which are selected from the test set of the zero-shot entity linking dataset. The mentions are denoted in yellow, the text in blue boxes are ground truth entities of the mentions, the candidate entity along with its type extracted by our model and BLINK under the setting of Recall@1 are highlighted in green and pink, respectively.

posed method in HO. This demonstrates the superiority of this traditional IR technique in dealing with cases where the words of the mention string and the entity title are highly overlapping. It can also be observed that KG-ZESHEL performs better than our model in AS, which suggests that using the knowledge graph can be a better choice in some cases. However, the performance of KG-ZESHEL in AS is only slightly higher than our model. Finally, we find that our proposed model improves more in MC and LO. We conjecture that these two categories require more complex reasoning according to the performance of candidate generation models in these two categories is much lower than in the other two categories. This indicates that our proposed model learns more contextual knowledge and has a more powerful reasoning ability to deal with the zero-shot entity linking candidate generation task by utilizing ultra-fine entity type information.

5.3 Case Study

To conduct qualitative analysis, Figure 4 shows two cases from the test set of the zero-shot entity linking dataset. In the case 1, the mention belongs to AS category, whose mention string is a substring of its gold entity title. The BLINK model will point to the entity *Arisa Kiyoto*, while our proposed model will point to the gold entity *Akiza Izinski*. Our model learns the contextual commonality of each type related entities during training by utilizing the ultra-fine entity typing task in a hierarchical multi-task way. At inference time, our model leverages the learned contextual commonality to improve the generalization ability. It will infer that there is some information of gold entity types in the mention with context in case 1 (e.g., *character, leader*, etc. instead of *artist, musician, singer*, etc.). This helps our model point to *Akiza Izinski* instead of *Arisa Kiyoto*. The same is true for case 2, whose gold entity title is followed by a disambiguation phrase and belongs to MC. Our model also infers that there is some information of some types(e.g., *software, application, program*, etc. instead of *policy, law, writing, document*, etc.), which helps our model point to *Prime Directive (game)* instead of *Prime Directive* like BLINK.

6 Conclusion

In this paper, we focus on the zero-shot entity linking task, which links mentions to unseen entities and only the textual information is available. This task challenges the generalization ability and often leads to a tendency of overfitting for entity linking models. To tackle the problem, we introduce the ultra-fine entity type information into the candidate generation phase of this task. Considering only entity description is available, we propose a two-stage model. We first extract ultra-fine entity types from each entity description, without depending on additional manually annotated data. Then we present a hierarchical multi-task model for jointly modeling candidate generation and ultra-fine entity typing, which can help the model to learn contextual commonality of types about the gold entity to improve the generalization ability. The experimental results demonstrate the effectiveness of utilizing the ultra-fine entity type information and our proposed method achieves state-of-the-art performance.

Acknowledgments

This research is supported by the Chinese Scientific and Technical Innovation Project 2030 (2018AAA0102100), the National Natural Science Foundation of China (No. 62272250, U1936206, U1903128).

References

Jonathan Berant, Andrew Chou, Roy Frostig, and Percy Liang. 2013. Semantic parsing on freebase from question-answer pairs. In *EMNLP*, pages 1533–1544.

Roi Blanco, Giuseppe Ottaviano, and Edgar Meij. 2015. Fast and space-efficient entity linking for queries. In *WSDM*, pages 179–188.

Shuang Chen, Jinpeng Wang, Feng Jiang, and Chin-Yew Lin. 2020. Improving entity linking by modeling latent entity type information. In *AAAI*, pages 7529–7537.

Eunsol Choi, Omer Levy, Yejin Choi, and Luke Zettlemoyer. 2018. Ultra-fine entity typing. In *ACL*, pages 87–96.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: pre-training of deep bidirectional transformers for language understanding. In *NAACL*, pages 4171–4186.

Dan Gillick, Nevena Lazic, Kuzman Ganchev, Jesse Kirchner, and David Huynh. 2014. Context-dependent fine-grained entity type tagging. *CoRR*, abs/1412.1820.

Nitish Gupta, Sameer Singh, and Dan Roth. 2017. Entity linking via joint encoding of types, descriptions, and context. In *EMNLP*, pages 2681–2690.

Raphael Hoffmann, Congle Zhang, Xiao Ling, Luke S. Zettlemoyer, and Daniel S. Weld. 2011. Knowledge-based weak supervision for information extraction of overlapping relations. In *ACL*, pages 541–550.

Feng Hou, Ruili Wang, Jun He, and Yi Zhou. 2020. Improving entity linking through semantic reinforced entity embeddings. In *ACL*, pages 6843–6848.

Heng Ji and Joel Nothman. 2016. Overview of TAC-KBP2016 tri-lingual EDL and its impact on end-to-end KBP. In *TAC*.

Diederik P. Kingma and Jimmy Ba. 2015. Adam: A method for stochastic optimization. In *ICLR*.

Xiao Ling and Daniel S. Weld. 2012. Fine-grained entity recognition. In *AAAI*.

Lajanugen Logeswaran, Ming-Wei Chang, Kenton Lee, Kristina Toutanova, Jacob Devlin, and Honglak Lee. 2019. Zero-shot entity linking by reading entity descriptions. In *ACL*, pages 3449–3460.

Yasumasa Onoe, Michael Boratko, Andrew McCallum, and Greg Durrett. 2021. Modeling fine-grained entity types with box embeddings. In *ACL*, pages 2051–2064.

Yasumasa Onoe and Greg Durrett. 2019. Learning to denoise distantly-labeled data for entity typing. In *NAACL*, pages 2407–2417.

Yasumasa Onoe and Greg Durrett. 2020. Fine-grained entity typing for domain independent entity linking. In *AAAI*, pages 8576–8583.

Jonathan Raiman and Olivier Raiman. 2018. Deeptype: Multilingual entity linking by neural type system evolution. In *AAAI*, pages 5406–5413. AAAI Press.

Petar Ristoski, Zhizhong Lin, and Qunzhi Zhou. 2021. KG-ZESHEL: knowledge graph-enhanced zero-shot entity linking. In *K-CAP*, pages 49–56.

Stephen E. Robertson and Hugo Zaragoza. 2009. The probabilistic relevance framework: BM25 and beyond. *Found. Trends Inf. Retr.*, 3(4):333–389.

Victor Sanh, Thomas Wolf, and Sebastian Ruder. 2019. A hierarchical multi-task approach for learning embeddings from semantic tasks. In *AAAI*, pages 6949–6956.

Hongyin Tang, Xingwu Sun, Beihong Jin, and Fuzheng Zhang. 2021. A bidirectional multi-paragraph reading model for zero-shot entity linking. In *AAAI*, pages 13889–13897.

Johannes Welbl, Pontus Stenetorp, and Sebastian Riedel. 2018. Constructing datasets for multi-hop reading comprehension across documents. *Trans. Assoc. Comput. Linguistics*, 6:287–302.

Maciej Wiatrak and Juha Iso-Sipilä. 2020. Simple hierarchical multi-task neural end-to-end entity linking for biomedical text. In *EMNLP*, pages 12–17.

Ledell Wu, Fabio Petroni, Martin Josifoski, Sebastian Riedel, and Luke Zettlemoyer. 2020. Scalable zero-shot entity linking with dense entity retrieval. In *EMNLP*, pages 6397–6407.

Zonghai Yao, Liangliang Cao, and Huapu Pan. 2020. Zero-shot entity linking with efficient long range sequence modeling. In *EMNLP*, pages 2517–2522.

Wen-tau Yih, Ming-Wei Chang, Xiaodong He, and Jianfeng Gao. 2015. Semantic parsing via staged query graph generation: Question answering with knowledge base. In *ACL*, pages 1321–1331.

Ming Zhu, Busra Celikkaya, Parminder Bhatia, and Chandan K. Reddy. 2020. LATTE: latent type modeling for biomedical entity linking. In *AAAI*, pages 9757–9764.

CofeNet: Context and Former-Label Enhanced Net for Complicated Quotation Extraction

Yequan Wang[1*], **Xiang Li**[2*], **Aixin Sun**[3], **Xuying Meng**[4], **Huaming Liao**[4], **Jiafeng Guo**[4]
[1]Beijing Academy of Artificial Intelligence, Beijing, China
[2]Alibaba Group, China
[3]School of Computer Science and Engineering, Nanyang Technological University, Singapore
[4]Institute of Computing Technology, Chinese Academy of Sciences, Beijing, China
tshwangyequan@gmail.com, yuanye.lx@alibaba-inc.com, axsun@ntu.edu.sg,
{mengxuying, lhm, guojiafeng}@ict.ac.cn

Abstract

Quotation extraction aims to extract quotations from written text. There are three components in a quotation: *source* refers to the holder of the quotation, *cue* is the trigger word(s), and *content* is the main body. Existing solutions for quotation extraction mainly utilize rule-based approaches and sequence labeling models. While rule-based approaches often lead to low recalls, sequence labeling models cannot well handle quotations with complicated structures. In this paper, we propose the **Co**ntext and **F**ormer-Label **E**nhanced **Net** (CofeNet) for quotation extraction. CofeNet is able to extract complicated quotations with components of variable lengths and complicated structures. On two public datasets (*i.e.,* PolNeAR and Riqua) and one proprietary dataset (*i.e.,* PoliticsZH), we show that our CofeNet achieves state-of-the-art performance on complicated quotation extraction.

1 Introduction

Quotation extraction aims to extract quotations from written text (Pouliquen et al., 2007). For example, given one instance shown in Figure 1, we extract the quotation with *source*: *some democrats*, *cue*: *privately express*, and *content*: *reservations about ...*. As a point of view, quotations provide opinions of the speaker, which is important for analyzing the speaker's stand. In general, quotation extraction is the first step before any further analysis, *e.g.,* speaker stand detection. In this paper, we focus on the extraction of the three quotation components.

As illustrated in the above example, the extraction of *content* component in a quotation is complicated and difficult due to three reasons: variable length, unclear boundary, and indistinguishable components. Specifically, the length of *content* can be over 10, or even more than 50 tokens. Moreover, *content* does not come with a regular pattern, which not only leads to a more unclear boundary of itself, but also affects the estimation of *source* and *cue*. For example, *content* in a quotation can be a complete instance with subject, predicate, and object. It is therefore hard to distinguish a noun (subject or object) representing the *source* or a part of *content*. Difficulty also exists in recognition of *cue* when tackling with a predicate, *e.g.,* verb. Thus, as *content* may contain another quotation, such a nesting structure further increases the difficulty of extracting quotations.

Yet for all the symbolism and feel-good value of such an appointment, *some democrats privately express reservations about entrusting a seat that could decide the balance of power in the closely divided senate to a candidate who has never won statewide, is considered less than dynamic and has been an anemic fundraiser.*

Figure 1: An example of quotations. Text spans with orange, green and gray denote *source*, *cue* and *content* respectively.

Many existing solutions for quotation extraction are rule-based methods (Pouliquen et al., 2007; Krestel et al., 2008; Elson and McKeown, 2010; Vu et al., 2018). Generally, quotations include direct quotations and indirect quotations. Quotation marks and their variants are clear; thus *content* can be extracted by using regular expressions. However, not all quoted texts are quotations. Meanwhile, not all quotations are quoted. Another popular rule-based approach is to recognize *cue* words, *e.g., speak(s)*. Similarly, not all *cue* words are related to quotations and vice versa. For both approaches, after recognizing *content* or *cue*, they usually search for the nearby noun as *source*. In short, rule-based methods only cover limited cases, leading to serious low recall problems.

Quotation extraction has also been formulated

[*]Indicates equal contribution

Proceedings of the 29th International Conference on Computational Linguistics, pages 2438–2449
October 12–17, 2022.

as a sequence labeling task. Pareti et al. (2013); Lee et al. (2020) directly adopt sequence labeling for quotation extraction. However, these solutions ignore the traits of quotations where lengths of quotation components are variable and structures of *content* are complicated. In general, *source* and *cue* components are short, *e.g.,* ≤ 3 tokens. However, *content* usually is over 10 tokens, or even more. Further, the complicated structure of *content* greatly reduces the performance of *content* extraction for sequence-labeling-based solutions.

In this paper, we propose **Co**ntext and **F**ormer-Label **E**nhanced **Net** (CofeNet) for quotation extraction. CofeNet is a novel architecture to extract quotations with variable-length and complicated-structured components. Our model is also capable of extracting both direct and indirect quotations.

CofeNet extracts quotations by utilizing dependent relations between sequenced texts. The model contains three components, *i.e., Text Encoder*, *Enhanced Cell*, and *Label Assigner*. Given a piece of text, the encoder encodes the instance and outputs the encoded hidden vectors. We design the Enhanced Cell module to study semantic representations of variable-length components with the utilization of contextual information. Specifically, the enhanced cell (i) uses a composer layer to enhance the input with the former labels (which are predicted by the former cells), the former words, the current word, and the latter words encoded by the encoder; and (ii) uses a gate layer and an attention layer to control and attend the corresponding input when predicting the label of the current word, at the level of element and vector respectively. Experimental results on two public datasets (*i.e.,* PolNeAR and Riqua) and one proprietary dataset (*i.e.,* PoliticsZH) show that our CofeNet achieves state-of-the-art performance on complicated quotation extraction.

2 Related Work

At first glance, quotation detection is a kind of "triplet" extraction, making the task similar to another two tasks, open information extraction (Angeli et al., 2015; Gashteovski et al., 2017) and semantic role labeling (Exner and Nugues, 2011). However, these three tasks have different focuses. Arguments extracted by semantic role labeling are event-related factors. OpenIE aims to output a structured representation of an instance in the form of binary or n-ary tuples, each of which consists of a predicate and several arguments. The extracted text spans in both tasks are typically short and less complicated, compared to the *content* in quotations. Because *content* extraction is the key challenge in quotation extraction, we will not further elaborate on semantic role labeling and OpenIE. Prior work on quotation extraction can be grouped into rule-based and sequence labeling methods.

2.1 Rule-based Methods

Extracting indirect quotations without clear boundaries is a challenging task, so early studies focus on rule-based methods to extract direct quotations (Pouliquen et al., 2007; Krestel et al., 2008; Elson and McKeown, 2010). In fact, rule-based methods perform well for marked texts, especially for direct quotations.

Pattern matching is a popular method in early studies. Pouliquen et al. (2007); Elson and McKeown (2010) identify *content*, *cue* and *source* by known quote-marks, pre-defined vocabulary, and rules of pattern recognition. The difference is that Elson and McKeown (2010) add machine learning methods to the quote attribution judgment so that they can process complex text. O'Keefe et al. (2012) use regular expressions to recognize quote-marks to extract components, then use sequence labeling to recognize quotation triplets.

Hand-built grammar is another popular rule-based method. Krestel et al. (2008) design a system by combining common verbs corresponding to *cue* and hand-built grammar to detect constructions that match six general lexical patterns. PICTOR (Schneider et al., 2010) utilizes context-free grammar to extract components of quotations.

2.2 Sequence Labeling Methods

Due to the development of deep learning, sequence-labeling-based approaches have attracted attention (Pareti et al., 2013; Lee et al., 2020). To identify the beginning of a quotation, Fernandes et al. (2011) use sequence labeling with features including part-of-speech and entity features generated by a guided transformation learning algorithm. Then they use regular expressions to recognize the *content* within quotations. Pareti et al. (2013) follow a similar idea but use CRF to decode the label. Lee et al. (2020) further use BERT to encode the text and CRF to decode the label on a non-public Chinese news dataset. However, these models cannot well handle quotations with complicated structures.

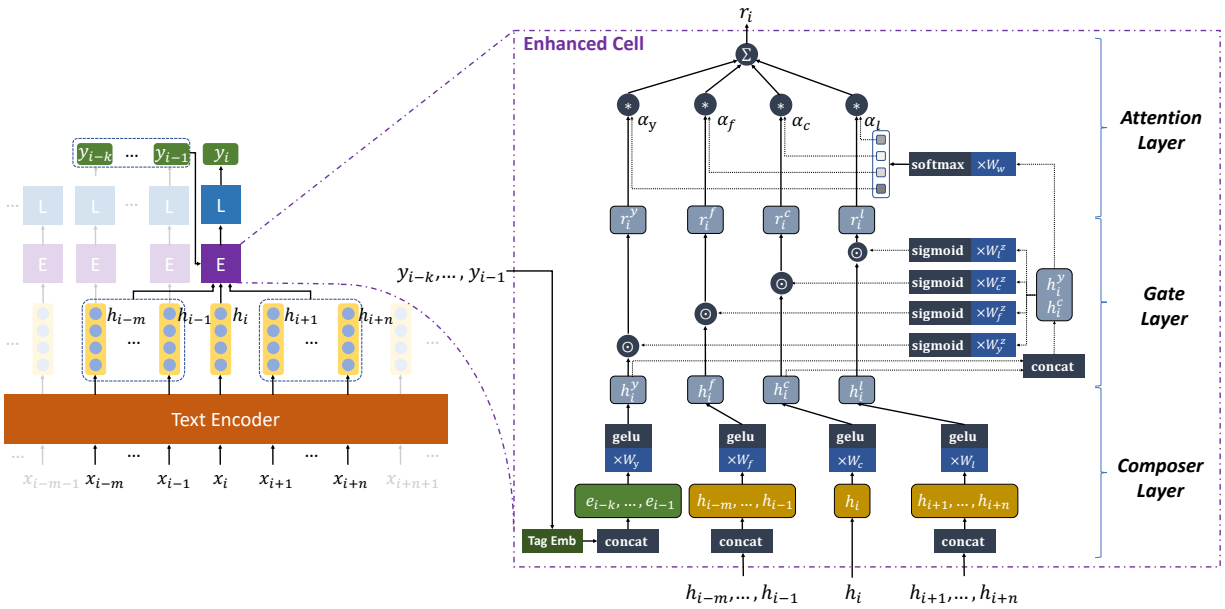

Figure 2: The architecture of CofeNet. Enhanced Cell is detailed on the right-hand side. (best viewed in color)

3 CofeNet Model

Figure 2 depicts the architecture of CofeNet. It consists of three modules: *Text Encoder*, *Enhanced Cell*, and *Label Assigner*. Text encoder is used to encode the input text to get hidden representations. Then, the enhanced cell is capable of building a representation considering the trait of quotations including variable-length and complicated-structured components. Last, the label assigner is to assign labels "B-source", "B-cue", "B-content", "I-source", "I-cue", "I-content" and "O", with BIO scheme.

3.1 Text Encoder

CofeNet is generic and can be realized by popular encoders such as LSTM (Hochreiter and Schmidhuber, 1997), CNN (Kim, 2014), Recursive Neural Network (Socher et al., 2011), and BERT (Devlin et al., 2019a). Unless otherwise specified, CofeNet denotes the model using BERT (Devlin et al., 2019b) as the encoder.

Given input text, hidden states of words are formulated by:

$$\{h_1, h_2, \ldots, h_N\} = \text{Encoder}(\{x_1, x_2, \ldots, x_N\}),$$

where, x_i is the i-th word of input, and Encoder denotes the Text Encoder. The hidden state h_i denotes the representation of i-th word x_i while encoding the preceding contexts of the position.

3.2 Enhanced Cell

As aforementioned, the challenge of quotation extraction is to extract the complicated-structured components with variable lengths. To this end, we design the enhanced cell with composer layer, gate layer, and attention layer, to study the semantic representations of variable-length components. At the same time, we also try to utilize contextual information and predicted labels.

Shown in Figure 2, the composer is used to reformat the input information to include the former labels $y_{i-k}, \ldots, y_{i-1}$, the former hidden states $h_{i-m}, \ldots, h_{i-1}$, the current state h_i, and the latter states $h_{i+1}, \ldots, h_{i+n}$. In this way, our model is able to consider a long span with different structures in a more coherent manner on top of encoded word representations. In general, the influence of different inflow information is different. To this end, we use a gate mechanism to control each element of input representations, and an attention mechanism to weigh the input representations at the vector level. Through the two mechanisms, we get a refined representation so that we could hold the complicated-structured and variable-length components of quotations. Next, we detail the workflow of the enhanced cell.

Composer Layer. The composer contains a label embedding unit and a linear unit to reformat the inflow information: the former labels $\{y_{i-k}, \ldots, y_{i-1}\}$, the former hidden states $\{h_{i-m}, \ldots, h_{i-1}\}$ of previous m words, the current state h_i of the current word x_i, and the latter states $\{h_{i+1}, \ldots, h_{i+n}\}$ of latter n words.

First, the enhanced cell contains a label embed-

ding unit, which is able to select the embedding of the given label, formulated by:

$$e_i = \text{Emb}(y_i), \tag{1}$$

where Emb denotes the mentioned label embedding unit. The predicted label of word i is y_i and the embedding of y_i is e_i. Taking the former k predicted labels into consideration, we get the former labels' representations $[e_{i-k}, \ldots, e_{i-1}]$ by concatenation, which is shown as a rectangle in green background, in the Enhanced Cell in Figure 2.

Intuitively, contextual information is important for us to predict the label of the current input word. We take the following context through simple but effective linear layers: the former predicted k labels, the former m words, the current word i, and the latter n words.

$$h_i^y = \text{GELU}([e_{i-k}, \ldots, e_{i-1}]W_y + b_y) \tag{2}$$
$$h_i^f = \text{GELU}([h_{i-m}, \ldots, h_{i-1}]W_f + b_f) \tag{3}$$
$$h_i^c = \text{GELU}(h_i W_c + b_c) \tag{4}$$
$$h_i^l = \text{GELU}([h_{i+1}, \ldots, h_{i+n}]W_l + b_l) \tag{5}$$

In the above formulation, the hidden states $\{h_{i-m}, \ldots, h_i, \ldots, h_{i+n}\}$ and label embeddings $\{e_{i-k}, \ldots, e_{i-1}\}$ are the input. W_y, W_f, W_c, W_l and b_y, b_f, b_c, b_l are the parameters of the linear layers. Here, we adopt GELU as the active function. $h_i^y, h_i^f, h_i^c, h_i^l$ denote the farther hidden states of the former labels, the former words, the current word and the latter words, respectively.

Gate Layer. The influence of different contexts is different. Hence, we use a gate mechanism to control the inflow hidden states at the element level. Inspired by Hochreiter and Schmidhuber (1997), we design a gate layer in the enhanced cell:

$$r_i^y = h_i^y \odot \text{sigmoid}([h_i^y, h_i^c]W_y^z + b_y^z) \tag{6}$$
$$r_i^f = h_i^f \odot \text{sigmoid}([h_i^y, h_i^c]W_f^z + b_f^z) \tag{7}$$
$$r_i^c = h_i^c \odot \text{sigmoid}([h_i^y, h_i^c]W_c^z + b_c^z) \tag{8}$$
$$r_i^l = h_i^l \odot \text{sigmoid}([h_i^y, h_i^c]W_l^z + b_l^z) \tag{9}$$

In the above formulation, r_i^y, r_i^f, r_i^c, and r_i^l denote the adjusted states of the former labels, the former words, the current word, and the latter word representation, respectively. The operation $\odot$ denotes element-wise product. W_y^z, W_f^z, W_c^z, W_l^z, and b_y^z, b_f^z, b_c^z, b_l^z are the parameters. We use sigmoid to adjust each element of the inflow representations.

Attention Layer. Inspired by Wang et al. (2016); Yang et al. (2016); Wang et al. (2018); Lin et al. (2019); Meng et al. (2022), we use an attention mechanism to attend the important part of r_i^y, r_i^f, r_i^c, and r_i^l. Since our target is to predict the label of the current word, we use the concatenation of h_i^y and h_i^c to attend the four vectors by

$$\alpha_y, \alpha_f, \alpha_c, \alpha_l = \text{softmax}([h_i^y, h_i^c]W_w + b_w), \tag{10}$$

where $\alpha_y, \alpha_f, \alpha_c$, and α_l are the weights for r_i^y, r_i^f, r_i^c, and r_i^l respectively. W_w and b_w are the parameters. In the attention layer, softmax function is used to calculate weights. Then, the current word representation r_i is obtained via:

$$r_i = \alpha_y r_i^y + \alpha_f r_i^f + \alpha_c r_i^c + \alpha_l r_i^l \tag{11}$$

To summarize, the Enhanced Cell uses the gate and attention layers with contextual information (*i.e.,* former labels, former words, current word, and latter words) to handle complicated-structured components with variable lengths. Specifically, to sense continuous span, we use attention layer by attending contextual information at the vector (macro) level, by using former labels, and the former, current, and latter word(s). Thus, the model avoids undesirable interruption within an instance. We also use the gate layer to control contextual information at the element (micro) level, especially former labels. Further, thanks to the ability of fine control, the gate layer is capable of avoiding illegal patterns, *e.g.,* "O" followed by "I-*".

3.3 Label Assigner

After getting the hidden representation of the current word, we use label assigner module to compute a probability distribution of the current label.

Briefly speaking, in label assigner, we use softmax classifier to calculate the distribution $\mathcal{P}_i$ of the current word i. Then argmax is used to assign a label of the current word. The two operations can be formulated as

$$\mathcal{P}_i = \text{softmax}(r_i W_p + b_p), \tag{12}$$
$$y_i = \text{argmax}(\mathcal{P}_i), \tag{13}$$

where W_p and b_p are the parameters.

3.4 Training Objective

The proposed CofeNet model could be trained in an end-to-end way by backpropagation. We adopt the cross-entropy objective function that has been

used in many studies (Tang et al., 2015; Wang et al., 2016, 2019).

Sequence Labeling Objective. Similar to sequence labeling tasks, we evaluate the label of all words for each given training instance. Recall that our objective is to predict the label of each word in the given instance. The unregularized objective L can be formulated as cross-entropy loss:

$$L(\theta) = -\sum_i \sum_j l_i^j \log(\mathcal{P}_i^j) \quad (14)$$

For a given training instance, l_i^j is the ground truth of label j for word i. Correspondingly, $\mathcal{P}_i^j$ is the probability of label j for word i. θ is the parameter set.

4 Experiment

We now evaluate the proposed CofeNet on two public datasets (*i.e.,* PolNeAR and Riqua), and one proprietary dataset (*i.e.,* PoliticsZH) against baselines. The implementation details and parameter settings are presented in Appendix A. On all datasets, we train the model with the training set, tune hyperparameters on the validation set, and report performance on the test set.

4.1 Datasets

PolNeAR. Political News Attribution Relations Corpus (PolNeAR) (Newell et al., 2018) is a corpus of news articles in English, on political candidates during US Presidential Election in November 2016. PolNeAR annotations are univocal, meaning that each word has only one label (*source*, *cue*, *content*, or none). The average number of tokens is 46.

Riqua. RIch QUotation Annotations (Riqua) (Papay and Padó, 2020) provides quotations, including interpersonal structure (speakers and addressees) for English literary. This corpus comprises 11 works of 19th-century literature that are manually annotated for direct and indirect quotations. Each instance, typically a sentence, is annotated with its *source*, *cue*, and *content*. The average number of tokens in this corpus is 129, longer than PolNeAR.

PoliticsZH. Chinese Political Discourse (PoliticsZH) contains politics and economics news collected from mainstream online media of China including Xinhua Net[1]. The news are in Chinese and the average length of input is 69 tokens, longer than PolNeAR but shorter than Riqua.

[1] http://news.cn/

Table 1: The statistics of three datasets. "Ave. len." refers to "Average length".

Dataset	Number of sentences			Ave. len. in tokens		
	Train	Valid	Test	Source	Cue	Content
PolNeAR	17,397	1,925	1,814	3.27	1.88	14.49
Riqua	1,604	208	105	1.38	1.08	20.65
PoliticsZH	10,754	1,344	1,345	3.08	1.80	43.47

Table 1 presents the statistics of the three datasets. We observe that the numbers of instances of PolNeAR and PoliticsZH are at the order of $10k$, and the Riqua is at $1k$. The length of *source* and *cue* is less than 5 tokens. The length of *content* is greater than 10, even 40 tokens. Note that for all three datasets, the length of *content* is much longer than *source* and *cue*.

4.2 Compared Methods

To provide a comprehensive evaluation, we experiment on both deep learning (*i.e.,* CNN, GRU, (Bi)LSTM, BERT, and BERT-CRF), and traditional methods (*i.e.,* Rule and CRF).

Rule. O'Keefe et al. (2012) uses rules including entity dictionary, reported speech verbs, and special flag characters to extract components of quotations.

CoreNLP. CoreNLP (Vu et al., 2018) contains quote extraction pipeline which deterministically picks out *source* and *content* from a text while ignoring *cue*.

CRF. Lafferty et al. (2001) present CRF to label sequence by building probabilistic models.

CNN. CNN (LeCun et al., 1995), a simple and parallelized model, can be independently adopted for sequence labeling tasks (Xu et al., 2018).

(Bi)LSTM. LSTM (Hochreiter and Schmidhuber, 1997) is able to exhibit dynamic temporal behavior due to its well-designed structure. We use it and its variants, *i.e.,* Bidirectional LSTM (BiLSTM).

GRU. GRU is a slightly more dramatic variation of LSTM (Cho et al., 2014).

BERT(-CRF). BERT is designed to pre-train deep bidirectional representations from unlabeled text by jointly conditioning on both left and right contexts (Devlin et al., 2019a).

4.3 Evaluation Metrics

The components of quotations are variable-length and complicated. As a result, it requires more spe-

Table 2: The $F1$ and J(accard) of methods on PolNeAR, Riqua and PoliticsZH datasets. The results marked with * are obtained by calling the CoreNLP toolkit package directly.

Dataset	Model	Source			Cue			Content		
		$F1$-E.	$F1$-B.	J	$F1$-E.	$F1$-B.	J	$F1$-E.	$F1$-B.	J
PolNeAR	Rule	10.7	13.0	8.8	22.8	25.3	14.4	5.6	10.5	6.1
	CoreNLP*	13.9	21.3	11.1	-	-	-	17.5	18.7	12.8
	CRF	50.6	56.2	42.1	53.4	63.3	44.1	28.6	50.9	42.3
	CNN	52.7	65.9	45.1	58.4	67.8	49.4	16.2	60.6	30.2
	GRU	46.5	58.2	36.7	59.1	68.1	48.8	51.3	65.0	51.3
	BiLSTM	64.1	74.4	56.8	63.3	72.6	55.1	53.4	67.3	53.7
	BERT	81.1	86.2	74.8	74.0	81.1	67.4	68.9	78.7	70.0
	CofeNet	**83.2**	**87.1**	**76.4**	**75.3**	**82.3**	**69.4**	**72.9**	**79.6**	**73.2**
Riqua	Rule	16.8	16.8	11.2	36.5	36.5	22.3	0.0	2.4	2.4
	CoreNLP*	22.8	22.8	17.9	-	-	-	63.8	63.8	46.9
	CRF	46.9	51.0	32.9	59.6	65.7	46.6	42.7	85.9	62.2
	CNN	52.7	59.1	39.6	85.2	85.2	74.2	45.2	95.4	58.5
	GRU	55.8	62.9	43.4	77.1	77.1	62.8	92.5	95.2	89.6
	BiLSTM	56.4	64.1	44.5	85.4	85.4	74.4	92.2	95.9	90.3
	BERT	74.5	77.9	62.4	88.9	88.9	80.0	94.3	96.6	92.9
	CofeNet	**81.8**	**84.3**	**72.6**	**89.2**	**89.2**	**80.4**	**94.4**	**97.1**	**94.1**
PoliticsZH	Rule	78.8	79.3	66.8	80.3	81.2	69.7	0.4	7.0	3.7
	CoreNLP*	38.1	39.5	24.3	-	-	-	0.2	2.2	4.3
	CRF	81.6	84.0	72.2	80.0	80.4	68.5	45.7	49.1	66.3
	CNN	82.5	87.8	76.5	81.4	83.6	72.1	35.0	74.5	46.7
	GRU	85.5	88.3	78.1	82.1	84.6	73.6	65.7	79.8	71.5
	BiLSTM	87.5	91.3	83.3	86.2	88.6	79.9	70.3	81.8	74.9
	BERT	92.6	93.7	88.2	89.5	90.8	84.0	73.7	83.6	84.4
	CofeNet	**93.7**	**94.4**	**89.8**	**90.3**	**91.1**	**85.4**	**78.0**	**86.9**	**88.7**

cific metrics. To this end, we evaluate the performance of models using our proposed "Jaccard", in addition to "Exact Match" and "Begin Match".

Exact Match. To measure the overall prediction at the instance level, we propose Exact Match index to quantify whether the multi-label prediction exactly matches the annotation. In the experiments, we use accuracy, precision, recall, and $F1$ to evaluate the exact match performance.

Begin Match. Exact match is harsh, especially long text span. Generally, the length of *source* and *cue* is short while the *content* is much longer. As a result, exact match is hard for *content*. To this end, we use begin match to evaluate only the beginning location for text span matching (Lee and Sun, 2019).

Jaccard. For text span matching, an important index is a ratio of the overlapping span over the total span. Thus we use "Jaccard" index to evaluate the performance of model in this aspect. Given the groundtruth text span $\mathcal{T}_g$ and its predicted text span $\mathcal{T}_p$, we can calculate the Jaccard index J through

$$J = \frac{|\mathcal{T}_p \cap \mathcal{T}_g|}{|\mathcal{T}_p \cup \mathcal{T}_g|}. \quad (15)$$

4.4 Main Results

Table 2 lists the $F1$ and J(accard) performance on the three datasets. In this table, the best results are in boldface and the second-best are underlined. We report results by exact match, begin match, and Jaccard, of all models for the three components of quotations. Here, $F1$-E. and $F1$-B. refer to the $F1$ based on exact match and begin match, respectively. The precision, recall and accuracy are shown in the page[2] due to space limitation. Our CofeNet model is listed in the last row of each dataset.

Table 2 shows that our CofeNet performs the best against all baselines. BERT achieves the second-best, followed by other deep-learning-based models. Note that due to the settled human-written rules, the performance of Rule and CoreNLP is not stable. For *source* and *cue*, on PoliticsZH, the performance is good due to more comprehensive rules. However, the rules on the other two datasets do not fit the domain well. As a comparison, *content* is on the opposite side. For *content*, the precision and recall of CoreNLP are 97.2 and 47.5 on Riqua dataset, which is better than PolNeAR. PoliticsZH dataset shows the worst performance. This is be-

[2] https://thuwyq.github.io/docs/cofenet-detail-exp.pdf

Table 3: The $F1$ and J of methods on PolNeAR. *B.L.* and *B.L.C.* denote BiLSTM and BiLSTM+CRF respectively.

Model	Source			Cue			Content		
	$F1$-E.	$F1$-B.	J	$F1$-E.	$F1$-B.	J	$F1$-E.	$F1$-B.	J
CNN	52.7	65.9	45.1	58.4	67.8	49.4	16.2	60.6	30.2
w. CRF	+8.3	**+4.1**	+8.0	**+4.3**	**+2.2**	**+3.6**	+25.8	+1.9	+19.3
w. Cofe	**+9.4**	+3.9	**+8.1**	+3.7	+2.1	+3.2	**+31.8**	**+3.1**	**+21.9**
GRU	46.5	58.2	36.7	59.1	68.1	48.8	51.3	65.0	51.3
w. CRF	+19.3	+13.7	+19.3	+6.2	+3.9	+6.8	+3.8	+0.8	**+6.2**
w. Cofe	**+20.5**	**+14.6**	**+19.7**	**+7.2**	**+4.6**	**+7.5**	**+6.9**	**+1.9**	+6.2
LSTM	46.1	56.4	35.7	58.6	67.5	47.9	50.4	65.5	50.8
w. CRF	+19.4	+14.7	+19.4	+6.4	+4.2	+6.7	+4.6	+0.3	+5.4
w. Cofe	**+21.8**	**+16.3**	**+20.9**	**+6.5**	**+4.3**	**+7.1**	**+7.6**	**+0.7**	**+6.0**
BiLSTM	64.1	74.4	56.8	63.3	72.6	55.1	53.4	67.3	53.7
w. CRF	+5.5	+1.3	+4.5	+3.4	+1.2	+2.6	+5.6	+2.1	+6.6
w. Cofe	**+7.1**	**+3.7**	**+7.0**	**+3.7**	**+1.3**	**+3.4**	**+8.8**	**+3.4**	**+9.1**
BERT	81.1	86.2	74.8	74.0	81.1	67.4	68.9	78.7	70.0
w. CRF	+1.1	+0.3	+0.8	+0.9	+0.9	+1.5	+2.1	+0.2	+2.8
w. CNN	-0.3	+0.6	+0.5	+0.0	+1.0	+1.2	+0.7	+0.3	+0.8
w. LSTM	+0.5	+0.4	+0.4	-0.3	0.0	+0.1	+2.0	+0.3	+1.0
w. *B.L.*	-0.6	-0.1	-0.5	-0.5	+0.7	+0.5	+0.7	-0.2	-0.6
w. *B.L.C.*	+1.4	+0.3	+1.2	**+1.4**	+0.9	+1.8	+2.9	+0.2	+2.4
w. Cofe	**+2.2**	**+0.9**	**+1.7**	+1.3	**+1.2**	**+2.0**	**+4.0**	**+1.0**	**+3.2**

cause CoreNLP uses quote marks to extract quotations. The number of direct quotations (*i.e.,* quoted *content*) on PolNeAR and Riqua is large, while the PoliticsZH is small. This shows that the rule-based methods cannot effectively identify indirect quotations.

The level of difficulty in extracting *source*, *cue*, and *content* is different. As a result, the performances of *source* and *cue* are better than the difficult *content*. This is expected because *content* is longer and complex in semantics. For example, the *content* may contain another *source*, *cue* and *content*. We design gate and attention mechanisms to fit those so that our model performs well.

4.5 Comparison with CRF and BERT

Comparison with CRF. CRF is a popular approach to handle sequence labeling problems, *e.g.,* NER (Ritter et al., 2011; Dong et al., 2016). We compare CofeNet with CRF by changing the encoder, *i.e., LSTM w. Cofe* denotes the Cofe using LSTM as text encoder. Recall that CofeNet specifically refers to the model using BERT as encoder, marked as *BERT w. Cofe* in Table 3. To make the comparison comprehensively and deeply, our comparisons between CRF and Cofe are based on various mainstream models including CNN, GRU, LSTM, BiLSTM, and BERT.

Table 3 details the comparison results on PolNeAR, and the results of the other two datasets are reported in the page[3]. (i) Results show that both Cofe and CRF perform better than basic models, and Cofe-based models perform better than CRF-based models. The comparison results suggest that our model architecture fits well with dependent sequence labeling tasks. As designed, the enhanced cell is capable of building the dependency relations of labels. (ii) Another interesting observation from the results is that if the basic model (*e.g.,* GRU) is simple, a larger improvement is achieved. On the contrary, the improvement over BERT is relatively small. It makes sense because the improvement is harder when the performance is already at a very high level. (iii) We also note that CofeNet performs better than CRF on all components of quotations.

Comparison with BERT. BERT based models are strong baselines for many tasks, particularly when there are clear patterns. The performance of models could be improved if we adopt a dependent encoding method based on BERT. To this end, based on BERT, we use decoders including CNN, LSTM, BiLSTM, BiLSTM+CRF in addition to CRF. The bottom area of Table 3 shows the results. Results show that the improvements of decoders including CNN, LSTM and BiLSTM are not significant than BiLSTM+CRF. Despite this, our CofeNet performs best. When meeting simple text span (*e.g.,*

[3] https://thuwyq.github.io/docs/cofenet-detail-exp.pdf

	B-source	I-source	B-cue	I-cue	B-content	I-content	O
<Start>	0.235	0.000	0.016	0.000	0.249	0.000	0.500
B-source	0.000	0.538	0.419	0.000	0.002	0.000	0.041
I-source	0.001	0.777	0.161	0.000	0.004	0.000	0.057
B-cue	0.054	0.000	0.000	0.380	0.342	0.000	0.225
I-cue	0.030	0.000	0.000	0.549	0.358	0.000	0.062
B-content	0.005	0.000	0.016	0.000	0.003	0.944	0.031
I-content	0.009	0.000	0.005	0.000	0.001	0.942	0.043
O	0.032	0.000	0.015	0.000	0.024	0.000	0.929

(a) The transition matrix of groundtruth

	B-source	I-source	B-cue	I-cue	B-content	I-content	O
<Start>	−0.006	0.000	−0.001	0.000	0.023	0.000	−0.016
B-source	0.000	0.022	−0.030	−0.001	0.002	0.000	0.006
I-source	−0.001	−0.006	−0.005	0.000	0.000	0.000	0.012
B-cue	0.006	0.000	0.000	−0.014	0.000	0.000	0.008
I-cue	0.003	0.000	0.000	0.025	−0.011	−0.003	−0.014
B-content	−0.001	0.000	0.002	0.000	0.002	−0.008	0.004
I-content	0.000	0.000	0.001	0.000	0.000	−0.001	0.000
O	0.000	0.000	0.003	0.000	0.003	0.000	−0.005

(b) The margin between groundtruth and CofeNet

Figure 3: The transition matrix and the margin of groundtruth and our model on PolNeAR.

Cue), the improvement of our proposed CofeNet is relatively small (1.3 point improvement, F1-Exact Match, on the Cue of PolNeAR dataset). When it comes to complex text span (*e.g.,* Content), our model shows large improvement over BERT model (4.0 points improvement, F1-Exact Match, on the Content of PolNeAR dataset).

From the comparisons, we demonstrate that our proposed CofeNet achieves the state-of-the-art performance on quotation extraction. To reveal the essence of CofeNet, we show the transition matrix of labels, the analysis on attention mechanism, and the ablation study in the next sections.

4.6 Label Transition Matrix

The probability transition matrix of labels reflects the particular features of *source*, *cue* and *content*. Thus we can use them to reveal the transition mechanism of labels. To this end, we calculate the label transition matrix of groundtruth, and the margin between groundtruth and CofeNet. Figure 3 depicts the detail on PolNeAR. In all subfigures, the column denotes the previous label and the row represents the current label. The value of Figure 3(a) denotes the transition probability of true labels, and the value of Figure 3(b) is the margin between the true and the predicted. As the word saying, "⟨Start⟩" denotes the location before the first word, "B-" and "I-" denote the beginning and the inside of the *source*, *cue* and *content*, respectively. "O" refers to the other words.

The transition matrix of groundtruth shown in Figure 3(a) reveals the statistics of the PolNeAR dataset. Recall that the key for quotation extraction

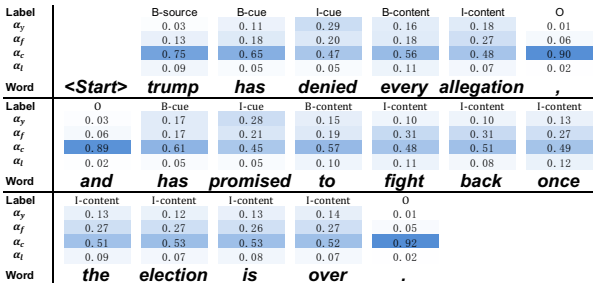

Figure 4: The attention weights of one test data from PolNeAR.

is the recognition of the "Begin". Hence, the margin of "Begin" is the compass for evaluating the performance. We find that the maximum absolute margin of "Begin" is -0.03, when the precious label is "B-source" and the current label is "B-cue". This is because the length of *source* is short, and *cue* word often follows *source* word closely. This proves that our model performs well even in difficult situations.

For BIO labeling scheme, the "I-source/cue/content" exists except the corresponding "B-*" exists. As a result, the transition value of "I-" could show the recognition ability of the model for those patterns. Also, Figure 3(b) shows almost all margins of those values are zeros. This reveals that our model could study those key patterns well.

4.7 Analysis on Attention Mechanism

In our design, the utilization of inflow information (*e.g.,* former labels, previous words, current word, and latter words) is the key for quotation extraction. Figure 4 shows the weights from the attention layer of one test instance in PolNeAR. To avoid the bias of a single case, we do a global prediction for all texts in the test dataset of PolNeAR attached in Appendix B. (i) The current word information has the largest weight, as expected. For the prediction of "I-source/cue/content", the former labels and former words information are the most important roles after the current word. It indicates that our model is capable of utilizing the former labels and sequence information as we designed. (ii) Another interesting observation is that the weights of the latter words' information for predicting "B/I-content" are about 0.1, which are greater than the other weights in α_l. As we mentioned before, the length of *content* is longer than *source* and *cue*, so the utilization of latter information improves the performance of long-span extraction more efficiently.

Table 4: Ablation study on PolNeAR dataset.

Model	Source			Cue			Content		
	$F1$-E.	$F1$-B.	J	$F1$-E.	$F1$-B.	J	$F1$-E.	$F1$-B.	J
CofeNet	83.2	87.1	76.4	75.3	82.3	69.4	72.9	79.6	73.2
w.o. g.m.	-1.0	-0.6	-0.9	-0.2	-0.2	-1.0	-0.8	-0.3	-1.2
w.o. a.m.	-0.9	-1.4	-1.5	-0.2	-1.0	-1.3	-1.2	-0.8	-1.3
w.o. f.l.	-2.4	-0.8	-1.5	-1.9	-0.5	-1.5	-2.5	-0.3	-2.7
w.o. f.w.	-0.9	-0.6	-1.1	-0.1	-0.3	-0.9	-1.3	-0.8	-1.1
w.o. c.w.	-2.0	-1.4	-2.0	-1.1	-1.0	-1.6	-1.4	-1.2	-1.2
w.o. l.w.	-1.0	-0.9	-1.2	-0.4	-0.4	-0.6	-1.7	-1.4	-1.0

4.8 Ablation Study

The CofeNet model uses gate mechanism $g.m.$ and attention mechanism $a.m.$ (see Section 3) to utilize information including former labels $f.l.$, former words $f.w.$, current word $c.w.$, and latter words $l.w.$. To study the effect of the two mechanisms and on the four information sources, we conduct ablation experiments on PolNeAR dataset.

Table 4 reports the results of this ablation study. (i) As expected, all mechanisms and information are useful for quotation extraction. For *content*, the Jaccard performance degrades at least 1.0 points after removing mechanisms or input information, which is similar to *source* and *cue*. As a comparison, the performance drop on $F1$-E. and $F1$-B. is significantly less than J. It is because the structure of *source* and *cue* is simpler than *content*. This phenomenon shows our CofeNet is particularly suitable for extracting quotations with long and complicated structures. (ii) When removing attention, larger drops on exact match are observed than removing gate. It reveals that attention is effective for begin match while gate prefers exact match. (iii) Further, we explore the performance of inflow information. The "w.o. f.w." on Table 4 shows that the former words' information is not so important for the prediction of *cue* because the *cue* is the shortest of all three components. The former label and the current word, the latter words are important for all of the components. It proves that the latter words' information is key for the recognition of *content*. This fits with our observations in Section 4.7.

5 Conclusion and Future Work

In this study, we design the CofeNet model for quotation extraction with variable-length span and complicated structure. The key idea of CofeNet model is to use gate and attention mechanisms to control the important information including former labels, former words, current word and latter words at the element and vector levels. Experiments show that the proposed model achieves the state-of-the-art performance on two public datasets PolNeAR and Riqua and one proprietary dataset PoliticsZH.

For quotation analysis, the extraction of quotation components is the first step. In our study, we split a long text into short texts to ensure that one instance contains one *source*, one *cue* and one *content*. Thus the recognition of quotation triplets from long text (*e.g.,* across instance) is one important future work. Another important direction is to go deep into the nesting phenomenon, which makes the recognition harder.

Acknowledgments

This work was supported by the National Key R&D Program of China (2020AAA0105200) and the National Science Foundation of China (NSFC No. 62106249, 61902382, 61972381).

References

Gabor Angeli, Melvin Jose Johnson Premkumar, and Christopher D. Manning. 2015. Leveraging linguistic structure for open domain information extraction. In Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers), pages 344–354, Beijing, China. Association for Computational Linguistics.

Kyunghyun Cho, Bart van Merriënboer, Caglar Gulcehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. 2014. Learning phrase representations using RNN encoder–decoder for statistical machine translation. In Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP), pages 1724–1734, Doha, Qatar. Association for Computational Linguistics.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019a. BERT: Pre-training of

deep bidirectional transformers for language understanding. In Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers), pages 4171–4186, Minneapolis, Minnesota. Association for Computational Linguistics.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019b. BERT: pre-training of deep bidirectional transformers for language understanding. In Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT 2019, Minneapolis, MN, USA, June 2-7, 2019, Volume 1 (Long and Short Papers), pages 4171–4186. Association for Computational Linguistics.

Chuanhai Dong, Jiajun Zhang, Chengqing Zong, Masanori Hattori, and Hui Di. 2016. Character-based LSTM-CRF with radical-level features for chinese named entity recognition. In Natural Language Understanding and Intelligent Applications - 5th CCF Conference on Natural Language Processing and Chinese Computing, NLPCC 2016, and 24th International Conference on Computer Processing of Oriental Languages, ICCPOL 2016, Kunming, China, December 2-6, 2016, Proceedings, volume 10102 of Lecture Notes in Computer Science, pages 239–250. Springer.

David K. Elson and Kathleen R. McKeown. 2010. Automatic attribution of quoted speech in literary narrative. In Proceedings of the Twenty-Fourth AAAI Conference on Artificial Intelligence, AAAI 2010, Atlanta, Georgia, USA, July 11-15, 2010. AAAI Press.

Peter Exner and Pierre Nugues. 2011. Using semantic role labeling to extract events from wikipedia. In Proceedings of the Workhop on Detection, Representation, and Exploitation of Events in the Semantic Web (DeRiVE 2011), Bonn, Germany, October 23, 2011, volume 779 of CEUR Workshop Proceedings, pages 38–47. CEUR-WS.org.

William Paulo Ducca Fernandes, Eduardo Motta, and Ruy Luiz Milidiú. 2011. Quotation extraction for Portuguese. In Proceedings of the 8th Brazilian Symposium in Information and Human Language Technology.

Kiril Gashteovski, Rainer Gemulla, and Luciano Del Corro. 2017. Minie: Minimizing facts in open information extraction. In Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing, EMNLP 2017, Copenhagen, Denmark, September 9-11, 2017, pages 2630–2640. Association for Computational Linguistics.

Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long short-term memory. Neural Comput., 9(8):1735–1780.

Yoon Kim. 2014. Convolutional neural networks for sentence classification. In Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP), pages 1746–1751, Doha, Qatar. Association for Computational Linguistics.

Diederik P. Kingma and Jimmy Ba. 2015. Adam: A method for stochastic optimization. In 3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings.

Ralf Krestel, Sabine Bergler, and René Witte. 2008. Minding the source: Automatic tagging of reported speech in newspaper articles. In Proceedings of the Sixth International Conference on Language Resources and Evaluation (LREC'08), Marrakech, Morocco. European Language Resources Association (ELRA).

John D. Lafferty, Andrew McCallum, and Fernando C. N. Pereira. 2001. Conditional random fields: Probabilistic models for segmenting and labeling sequence data. In Proceedings of the Eighteenth International Conference on Machine Learning (ICML 2001), Williams College, Williamstown, MA, USA, June 28 - July 1, 2001, pages 282–289. Morgan Kaufmann.

Yann LeCun, Yoshua Bengio, et al. 1995. Convolutional networks for images, speech, and time series. The handbook of brain theory and neural networks, 3361(10):1995.

Grace E. Lee and Aixin Sun. 2019. A study on agreement in PICO span annotations. In Proceedings of the 42nd International ACM SIGIR Conference on Research and Development in Information Retrieval, SIGIR 2019, Paris, France, July 21-25, 2019, pages 1149–1152. ACM.

Kuan-Lin Lee, Yu-Chung Cheng, Pai-Lin Chen, and Hen-Hsen Huang. 2020. Keeping their words: Direct and indirect chinese quote attribution from newspapers. In Companion of The 2020 Web Conference 2020, Taipei, Taiwan, April 20-24, 2020, pages 98–99. ACM / IW3C2.

Xi Lin, Yequan Wang, Xiaokui Xiao, Zengxiang Li, and Sourav S. Bhowmick. 2019. Path travel time estimation using attribute-related hybrid trajectories network. In Proceedings of the 28th ACM International Conference on Information and Knowledge Management, CIKM 2019, Beijing, China, November 3-7, 2019, pages 1973–1982. ACM.

Xuying Meng, Yequan Wang, Runxin Ma, Haitong Luo, Xiang Li, and Yujun Zhang. 2022. Packet representation learning for traffic classification. In KDD '22: The 28th ACM SIGKDD Conference on Knowledge Discovery and Data Mining, Washington, DC, USA, August 14 - 18, 2022, pages 3546–3554. ACM.

Edward Newell, Drew Margolin, and Derek Ruths. 2018. An attribution relations corpus for political news. In Proceedings of the Eleventh International Conference on Language Resources and Evaluation, LREC 2018, Miyazaki, Japan, May 7-12, 2018. European Language Resources Association (ELRA).

Timothy O'Keefe, Silvia Pareti, James R. Curran, Irena Koprinska, and Matthew Honnibal. 2012. A sequence labelling approach to quote attribution. In Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning, EMNLP-CoNLL 2012, July 12-14, 2012, Jeju Island, Korea, pages 790–799. ACL.

Timothy O'Keefe, Silvia Pareti, James R. Curran, Irena Koprinska, and Matthew Honnibal. 2012. A sequence labelling approach to quote attribution. In Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning, pages 790–799, Jeju Island, Korea. Association for Computational Linguistics.

Sean Papay and Sebastian Padó. 2020. Riqua: A corpus of rich quotation annotation for english literary text. In Proceedings of The 12th Language Resources and Evaluation Conference, LREC 2020, Marseille, France, May 11-16, 2020, pages 835–841. European Language Resources Association.

Silvia Pareti, Tim O'Keefe, Ioannis Konstas, James R. Curran, and Irena Koprinska. 2013. Automatically detecting and attributing indirect quotations. In Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing, pages 989–999, Seattle, Washington, USA. Association for Computational Linguistics.

Bruno Pouliquen, Ralf Steinberger, and Clive Best. 2007. Automatic detection of quotations in multilingual news. In Proceedings of Recent Advances in Natural Language Processing, pages 487–492.

Alan Ritter, Sam Clark, Mausam, and Oren Etzioni. 2011. Named entity recognition in tweets: An experimental study. In Proceedings of the 2011 Conference on Empirical Methods in Natural Language Processing, pages 1524–1534, Edinburgh, Scotland, UK. Association for Computational Linguistics.

Nathan Schneider, Rebecca Hwa, Philip Gianfortoni, Dipanjan Das, Michael Heilman, Alan W. Black, Frederick L. Crabbe, and Noah A. Smith. 2010. Visualizing topical quotations over time to understand news discourse.

Richard Socher, Cliff Chiung-Yu Lin, Andrew Y. Ng, and Christopher D. Manning. 2011. Parsing natural scenes and natural language with recursive neural networks. In Proceedings of the 28th International Conference on Machine Learning, ICML 2011, Bellevue, Washington, USA, June 28 - July 2, 2011, pages 129–136. Omnipress.

Duyu Tang, Bing Qin, and Ting Liu. 2015. Document modeling with gated recurrent neural network for sentiment classification. In Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing, pages 1422–1432, Lisbon, Portugal. Association for Computational Linguistics.

Thanh Vu, Dat Quoc Nguyen, Dai Quoc Nguyen, Mark Dras, and Mark Johnson. 2018. VnCoreNLP: A Vietnamese natural language processing toolkit. In Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Demonstrations, pages 56–60, New Orleans, Louisiana. Association for Computational Linguistics.

Yequan Wang, Minlie Huang, Xiaoyan Zhu, and Li Zhao. 2016. Attention-based LSTM for aspect-level sentiment classification. In Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing, pages 606–615, Austin, Texas. Association for Computational Linguistics.

Yequan Wang, Aixin Sun, Jialong Han, Ying Liu, and Xiaoyan Zhu. 2018. Sentiment analysis by capsules. In Proceedings of the 2018 World Wide Web Conference on World Wide Web, WWW 2018, Lyon, France, April 23-27, 2018, pages 1165–1174. ACM.

Yequan Wang, Aixin Sun, Minlie Huang, and Xiaoyan Zhu. 2019. Aspect-level sentiment analysis using as-capsules. In The World Wide Web Conference, WWW 2019, San Francisco, CA, USA, May 13-17, 2019, pages 2033–2044. ACM.

Hu Xu, Bing Liu, Lei Shu, and Philip S. Yu. 2018. Double embeddings and CNN-based sequence labeling for aspect extraction. In Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers), pages 592–598, Melbourne, Australia. Association for Computational Linguistics.

Zichao Yang, Diyi Yang, Chris Dyer, Xiaodong He, Alex Smola, and Eduard Hovy. 2016. Hierarchical attention networks for document classification. In Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, pages 1480–1489, San Diego, California. Association for Computational Linguistics.

Appendix

A Implementation Details

We list the implementation details of CofeNet.

Table 5: CofeNet-BERT experimental configuration on PolNeAR, Riqua and PoliticsZH datasets. The sampling ratio is the value selection ratio of the former label during training. The three values represent the proportions of truth label, predict label and random label.

Training hyperparameters	
Optimizer	Adam
Learning rate except BERT	1e-3
Learning rate of BERT	5e-5
The hyperparameters of BERT	
Encoder layer	12
Attention head	12
Hidden size	768
Intermediate size	3,072
The hyperparameters of CofeNet	
Hidden size	100
Label embedding	100
Number of Former labels k	1
Number of Former words n	3
Number of Latter words m	3

Table 5 lists the same settings for the two public datasets (*i.e.,* PolNeAR and Riqua) and our proprietary dataset (*i.e.,* PoliticsZH). The learning rate for model parameters except BERT are $1e-3$, and $5e-5$ for BERT. We use typical 12-layers BERT (known as *bert-base-uncased* [4]) as a basic encoder for the two English datasets. For the Chinese dataset PoliticsZH, we use *bert-base-chinese* [5]. The middle part of Table 5 shows the important hyperparameters of BERT. There are other hyperparamters for CofeNet except BERT related. The hidden sizes of word representation and label embedding are 100. The number of former labels, former words, and latter words is 1, 3, and 3, respectively. The different hyperparameter for CofeNet is the batch size due to the GPU memory limitation. During training, we set the batch sizes for PolNeAR, Riqua and PoliticsZH to 15, 15 and 16, respectively.

We use Adam (Kingma and Ba, 2015) as our optimization method. CofeNet is implemented on Pytorch (version 1.2.0). NLTK is used to segment text. For BERT model, we invoke the pytorch-transformers package (version 1.2.0). To ensure the reliability of experimental results, we use the same transformer package with the same initialization parameters in BERT, BERT-CRF and CofeNet.

[4]https://s3.amazonaws.com/models.huggingface.co/bert/bert-base-uncased-pytorch_model.bin

[5]https://s3.amazonaws.com/models.huggingface.co/bert/bert-base-chinese-pytorch_model.bin

	B-source	I-source	B-cue	I-cue	B-content	I-content	O
<Start>	0.045	-	0.115	-	0.119	-	0.059
B-source	-	0.143	0.119	0.236	-	-	0.037
I-source	0.021	0.166	0.121	0.201	0.145	0.172	0.030
B-cue	0.044	-	-	0.244	0.152	-	0.035
I-cue	0.053	-	-	0.233	0.163	0.158	0.048
B-content	0.031	-	0.094	-	0.106	0.105	0.032
I-content	0.022	-	0.085	0.178	0.080	0.096	0.021
O	0.056	-	0.170	0.290	0.136	0.199	0.138

(a) The weight α_y for former labels r_i^y

	B-source	I-source	B-cue	I-cue	B-content	I-content	O
<Start>	0.134	-	0.165	-	0.190	-	0.093
B-source	-	0.171	0.194	0.201	-	-	0.095
I-source	0.090	0.148	0.170	0.172	0.166	0.214	0.070
B-cue	0.164	-	-	0.200	0.203	-	0.088
I-cue	0.158	-	-	0.201	0.201	0.252	0.117
B-content	0.148	-	0.198	-	0.200	0.295	0.107
I-content	0.150	-	0.198	0.236	0.180	0.297	0.078
O	0.113	-	0.159	0.177	0.151	0.203	0.110

(b) The weight α_f for former words r_i^f

	B-source	I-source	B-cue	I-cue	B-content	I-content	O
<Start>	0.720	-	0.651	-	0.600	-	0.794
B-source	-	0.638	0.639	0.522	-	-	0.829
I-source	0.792	0.622	0.653	0.568	0.596	0.532	0.864
B-cue	0.682	-	-	0.488	0.552	-	0.843
I-cue	0.690	-	-	0.488	0.530	0.486	0.779
B-content	0.714	-	0.602	-	0.586	0.488	0.809
I-content	0.731	-	0.650	0.488	0.632	0.527	0.869
O	0.754	-	0.620	0.473	0.637	0.512	0.705

(c) The weight α_c for current word r_i^c

	B-source	I-source	B-cue	I-cue	B-content	I-content	O
<Start>	0.100	-	0.069	-	0.091	-	0.053
B-source	-	0.048	0.048	0.041	-	-	0.039
I-source	0.097	0.064	0.057	0.059	0.093	0.082	0.035
B-cue	0.110	-	-	0.067	0.093	-	0.034
I-cue	0.099	-	-	0.077	0.107	0.103	0.055
B-content	0.107	-	0.106	-	0.108	0.112	0.052
I-content	0.097	-	0.067	0.097	0.108	0.080	0.032
O	0.077	-	0.052	0.061	0.076	0.086	0.048

(d) The weight α_l for latter words r_i^l

Figure 5: The weights for hidden states on PolNeAR.

B Global Analysis on Attention Mechanism

In our design, the utilization of inflow information is the key for quotation extraction. Recall that the information includes the former labels, the previous words, the current word and the latter words. Hence, we use the attention to reveal the operating principle of the model. Figure 4 has shown the weights from the attention layer of one individual case from test set of PolNeAR dataset. To avoid the bias of a single case, we do a global prediction for all texts in test set of PolNeAR shown in Figure 5. The observations from Figure 5 are similar to that reported in Section 4.7, so we will not repeat them.

Supporting Medical Relation Extraction via Causality-Pruned Semantic Dependency Forest

Yifan Jin[1,2,*], **Jiangmeng Li**[1,2,*,†], **Zheng Lian**[1,2], **Chengbo Jiao**[3], **Xiaohui Hu**[1,2]
[1]University of Chinese Academy of Sciences
[2] Institute of Software Chinese Academy of Sciences
[3]University of Electronic Science and Technology of China
{yifan2020,jiangmeng2019,lianzheng2017,hxh}@iscas.ac.cn
chengbojiao@hotmail.com

Abstract

Medical Relation Extraction (MRE) task aims to extract relations between entities in medical texts. Traditional relation extraction methods achieve impressive success by exploring the syntactic information, e.g., dependency tree. However, the quality of the 1-best dependency tree for medical texts produced by an out-of-domain parser is relatively limited so that the performance of medical relation extraction method may degenerate. To this end, we propose a method to jointly model semantic and syntactic information from medical texts based on causal explanation theory. We generate dependency forests consisting of the semantic-embedded 1-best dependency tree. Then, a task-specific causal explainer is adopted to prune the dependency forests, which are further fed into a designed graph convolutional network to learn the corresponding representation for downstream task. Empirically, the various comparisons on benchmark medical datasets demonstrate the effectiveness of our model.

1 Introduction

Medical relation extraction (MRE) refers to identifying relations among entities from medical literature and reports. It plays a very important role in downstream tasks such as medical knowledge graph construction (Li et al., 2020; Rotmensch et al., 2017) and biomedical knowledge discovery (Quirk and Poon, 2016). On the other hand, as the number of medical literature increases, it becomes increasingly important to automatically discover the relation among entities in the literature (Peng et al., 2017).

The addition of syntactic structure has been demonstrated to be beneficial for various natural language processing tasks (Zaremoodi and Haffari, 2017; Zhou et al., 2005; Le and Zuidema, 2015). As a type of syntactic structure, the dependency tree capturing long-distance connections between words can indeed improve benchmark relation extraction methods (Tian et al., 2021; Chen et al., 2021; Zhang et al., 2018; Sun et al., 2020). We demonstrate an example in Figure 1. Specifically, the 1-best dependency tree of the sentence "***Aminopropylindenes*** derived from Grundmann's ketone as a novel chemotype of ***oxidosqualene cyclase*** inhibitors" in the CPR dataset. ***Aminopropylindenes*** and ***oxidosqualene cyclase*** are the entities, and the relation between them is "down regulator", denoted as "CPR:4".

However, in the medical field, the quality of the 1-best dependency tree generated by the out-of-domain parsers, e.g., parsers for the news domain, is relatively deficient. Generally, the main verb in a sentence is treated as the root node in the dependency tree, while, as the example shown in Figure 1, the entity ***Aminopropylindenes***, apparently a *noun*, is treated as the root node. To solve this problem, multiple methods with dependency forests have been proposed (Song et al., 2019; Jin et al., 2020; Guo et al., 2021). Such approaches focus on redesigning the parser or substituting the parser with a semantic encoder, but the semantic and dependency tree syntactic information is used in a biased manner. Furthermore, the causality between the edges in the dependency forests and the performance of the model is not explored by benchmark methods.

To this end, we propose a novel approach, namely **C**ausality-**P**runed semantic dependency forest **G**raph **C**onvolutional **N**etwork (CP-GCN). To acquire the dependency forests enriched with semantic and syntactic information in an unbiased manner, we first obtain the 1-best dependency tree, as the sentence syntactic information, which is generated by the out-of-domain parser, and then fuse the syntactic information with the semantic information by using a switch gate network. The seman-

[*]Equal contribution.
[†]Corresponding author.

Proceedings of the 29th International Conference on Computational Linguistics, pages 2450–2460
October 12–17, 2022.

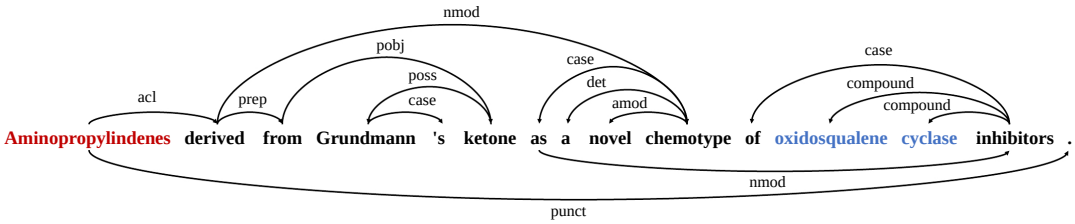

Figure 1: 1-best dependency tree for a biological sentence generated by the parser. ***Aminopropylindenes*** and ***oxidosqualene cyclase*** are the entities in the sentence.

tic information is captured in different representation subspaces using multi-head attention (Vaswani et al., 2017). To extract dependency forests' edges that are causally related to the MRE performance, we construct a causal explanation dataset based on Granger causality (Granger, 1969, 1980) and train a task-specific causal explainer. We then obtain task-specific explanations of the dependency forests generated by the trained explainer and prune the dependency forests by following the corresponding explanations, which aim to eliminate the task-irrelevant information from the dependency forests. The pruned dependency forests are encoded by DCGCNs (Guo et al., 2019b) for MRE task. Empirically, the comparisons demonstrate that CP-GCN achieves state-of-the-art on benchmark relation extraction tasks, e.g., for the sentence-level relation extraction task, our model obtains 67.3 and 92.9 scores on CPR and PGR, respectively. The **contributions** are summarized as follows:

- We propose an approach to generate dependency forests enriched with semantic and syntactic information in an unbiased manner.
- We propose a causal pruning approach to remove task-irrelevant information from the dependency forests, which is achieved by using a task-specific explainer trained on a causal explanation dataset for the target MRE task.
- CP-GCN achieves state-of-the-art on benchmark MRE datasets, and the ablation comparisons further support the effectiveness of each part of our model.

2 Related Work

2.1 Medical Relation Extraction

Previous work performs the MRE task by constructing the 1-best dependency tree of sentences (Peng et al., 2017; Song et al., 2018). However, the accuracy of the 1-best dependency tree generated by the out-of-domain parser is relatively low, resulting in a fall in MRE performance. Therefore, (Song et al., 2019) proposes to use dependency forests to solve this problem, which uses EDGEWISE and KBESTEISNER algorithm to pick edges to construct dependency forests. (Jin et al., 2020) encodes all effective dependency trees generated by a parser into dependency forests. (Guo et al., 2021) utilizes multi-head attention and Kirchhoff's Matrix-Tree Theorem (MMT) (Koo et al., 2007) to automatically generate latent dependency forests without the usage of any parser. In general, (Song et al., 2019) and (Jin et al., 2020) focus more on the syntactic information in the 1-best dependency tree generated by the out-of-domain parser, while (Guo et al., 2021) directly discards the syntactic information and focuses only on the semantic information.

2.2 Causal Explanation

Causal explanation is designed to explain the importance of each module in a machine learning model on the prediction, which receives increasing attention recently (Datta et al., 2016; Schwab and Karlen, 2019; Lin et al., 2021). There are several viable forms of causality, including Granger causality (Granger, 1969), causal Bayesian networks (Pearl, 1985), and structural causal models (Pearl, 2009). (Chattopadhyay et al., 2019) proposes an attribution method based on the first principles of causality. (Schwab and Karlen, 2019) models the explanation task of image deep learning models as a causal learning task and proposes a causal explanation model based on Granger causality. (Lin et al., 2021) proposes a framework for explaining graph neural networks using the first principles of Granger causality.

3 Preliminaries

3.1 Task Definition

Our task is to extract relation between entities in a sentence, focusing on both binary relation extraction and ternary relation extraction. Formally, the input to our task is a sentence $\boldsymbol{S} =$

$\{w_1, w_2, \ldots, w_n\}$ with n words and w_i denotes the i-th word in the sentence. $\boldsymbol{S}$ is annotated with entity mentions $\boldsymbol{E_1}$ and $\boldsymbol{E_2}$[1]. The output is the relation between entities from a predefined relation set $\boldsymbol{R} = \{r_1, r_2, \ldots, r_m\}$, where m denotes the number of relations.

3.2 Densely-Connected Graph Convolutional Networks

Graph Neural Network is a set of models that can effectively encode the information of graph structure, the classical models including Graph Attention Networks (GATs) (Velickovic et al., 2017), Graph Convolutional Networks (GCNs) (Kipf and Welling, 2016), etc. Densely-Connected Graph Convolutional Networks (DCGCNs) (Guo et al., 2019b) is a variant of GCNs, which introduces dense connections to GCNs. Thus being able to build multi-layer GCNs models with a large depth and learn richer information than the shallower GCNs models. More specifically, DCGCNs differs from GCNs in that the embedding of node v in the l-th layer receives information from all the preceding layers, which can be formulated as follows:

$$\boldsymbol{h}_v^{(l)} = \rho\left(\sum_{u \in \mathcal{N}(v)} \boldsymbol{W}^{(l)} \times \boldsymbol{g}_u^{(l)} + \boldsymbol{b}^{(l)}\right) \quad (1)$$

where $\times$ denotes matrix multiplication, $\boldsymbol{h}_v^{(l)}$ is the embedding of node v in the l-th layer, ρ is an activation function, $\mathcal{N}(v)$ denotes the neighbours of node v, $\boldsymbol{W}^{(l)}$ and $\boldsymbol{b}^{(l)}$ are the weight matrix and bias vector of the l-th layer respectively, and $\boldsymbol{g}_u^{(l)}$ indicates the information about node u from all the preceding layers. Mathematically, $\boldsymbol{g}_u^{(l)}$ can be calculated by concatenating the initial embedding $\boldsymbol{x}_u$ and the node embedding $\boldsymbol{h}_u^{(1)}; \ldots; \boldsymbol{h}_u^{(l-1)}$ produced in layer $1, \ldots, l-1$, respectively.

$$\boldsymbol{g}_u^{(l)} = [\boldsymbol{x}_u; \boldsymbol{h}_u^{(1)}; \ldots; \boldsymbol{h}_u^{(l-1)}] \quad (2)$$

3.3 Dependency Tree Generation

To construct the 1-best dependency tree, we use Standard CoreNLP Toolkits (SCT) (Manning et al., 2014) to obtain the dependency tree $\mathcal{T}$ for each input sentence $\boldsymbol{S}$ and represent $\mathcal{T}$ by a adjacency matrix $\mathbf{T} = (t_{i,j})_{n \times n}$[2], where $t_{i,j}$ is the dependency type (e.g., *dobj*) between w_i and w_j, e.g., $t_{i,j} = 0$ if the connection between w_i and w_j do *not* exist. Then, we encode $t_{i,j}$ to the corresponding embedding $c_{i,j}^t$ with a learnable matrix, and use $\mathbf{C} = (c_{i,j}^t)_{n \times n}$ to denote the syntactic matrix.

[1] $\boldsymbol{E_1}$, $\boldsymbol{E_2}$ and $\boldsymbol{E_3}$ for ternary relation extraction.

[2] The adjacency matrix $\mathbf{T}$ adds the self-loop of each word to the dependency tree $\mathcal{T}$ with the "self" dependency type and regards the dependencies between words as unoriented.

4 Methodology

In this section, we introduce our proposed CP-GCN model shown in Figure 2.

4.1 Causality-Pruned Semantic Dependency Forest Generator

In the medical domain, the quality of the 1-best dependency tree generated by the out-of-domain parsers is relatively deficient. Thus, we propose the Causality-pruned Semantic dependency Forest Generator (CSFG) to generate dependency forests enriched with syntactic and semantic information and derive task-relevant information from them.

4.1.1 Semantic Embedding Module

In order to construct dependency forests that combine both semantic and syntactic information in an unbiased manner, we propose a semantic embedding module to incorporate the semantic information of the sentence into the 1-best dependency tree.

Specifically, we model semantic information using the multi-head attention mechanism (Vaswani et al., 2017) with N heads, which captures the semantic relevance between words in a sentence. For the p-th head, we compute the semantic matrix $\mathbf{A}^p$ by using the query vector $\boldsymbol{Q}$ and the key vector $\boldsymbol{K}$:

$$\mathbf{A}^p = \frac{\left(\boldsymbol{Q} \times \boldsymbol{W^Q}\right) \times \left(\boldsymbol{K} \times \boldsymbol{W^K}\right)^\top}{\sqrt{d}} \quad (3)$$

where $\boldsymbol{W^Q}$ and $\boldsymbol{W^K}$ are learnable transformation matrices for $\boldsymbol{Q}$ and $\boldsymbol{K}$, respectively, and d is the dimension of $\boldsymbol{K}$.

Finally, the p-th dependency forest can be obtained by summing the syntactic matrix $\mathbf{C}$ and the semantic matrix $\mathbf{A}^p$ with a switch gate network and a softmax function:

$$\mathbf{F}^p = \text{softmax}\left((1-\alpha)\ \mathbf{A}^p + \alpha\, \mathbf{C}\right) \quad (4)$$

where $\alpha \in [0, 1]$ is a hyper-parameter to balance the syntactic matrix $\mathbf{C}$ and the semantic matrix $\mathbf{A}^p$, and $\mathbf{F}^p$ is the adjacency matrix of the p-th dependency forest.

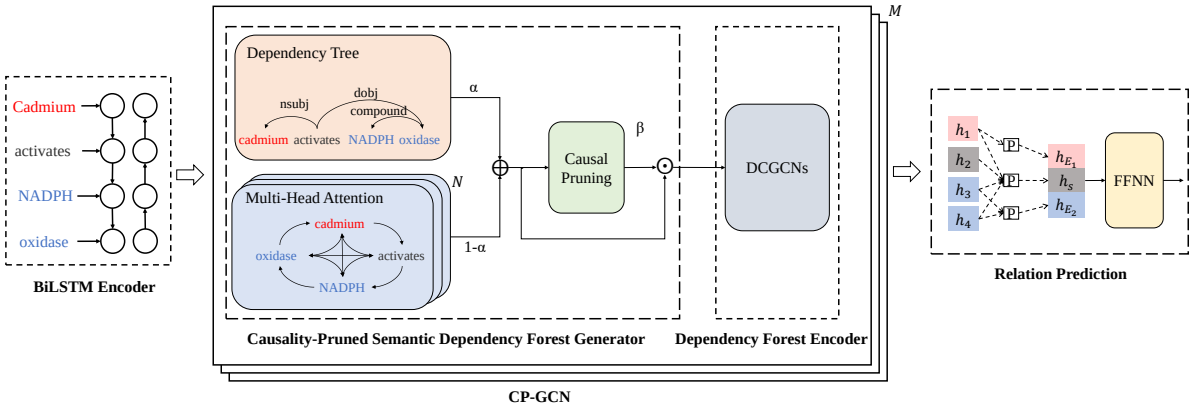

Figure 2: The overall architecture of CP-GCN with an example input sentence ("***Cadmium***" in red and "***NADPH oxidase***" in blue are two entities of the sentence). The model consists of three components: 1) BiLSTM Encoder obtains the sentence representation with a BiLSTM model. 2) CP-GCN is the main component of the model which contains M identical blocks, and each block contains two modules. Causality-Pruned Semantic Dependency Forest Generator combines the dependency tree and the multi-head attention with N heads to generate dependency forests and then prunes them using a task-specific causal explainer. Dependency Forest Encoder uses DCGCNs to encode the pruned dependency forests. 3) Relation Prediction module predicts relations using global and local max pooling and feedforward neural networks (FFNN).

4.1.2 Task-Specific Causal Pruning Module

In this part, our major objective is to extract dependency forests' edges that are causally related to the MRE performance. Inspired by (Lin et al., 2021), we propose a method consisting of three processes for pruning dependency forests based on Granger causality. The first two processes aim to train a task-specific causal explainer, which are illustrated in Figure 3. The causal pruning process prunes the dependency forests with the trained causal explainer.

Causal explanation generation process. This process is designed to construct a causal explanation dataset for a specific MRE task. Given a pre-trained MRE model denoted by $f_{\text{MRE}}(\cdot)$ and the gold-standard relation r of the sentence $\boldsymbol{S}$. We start by using the semantic embedding module of the pre-trained MRE model to generate $N * M$ dependency forests of the sentence $\boldsymbol{S}$, denoted by $\mathcal{G} = \{\boldsymbol{G}^1, \boldsymbol{G}^2, \ldots, \boldsymbol{G}^{N*M}\}$. For any dependency forest $\boldsymbol{G}^i$, it can be represented as $\boldsymbol{G}^i = (\mathbf{F}^i, \mathbf{H}_0)$, where $\mathbf{F}^i$ is the fully-connected adjacency matrix indicating the weights of the edges, and $\mathbf{H}_0$ is the matrix of node features, which is the same for each dependency forest. Then, we need to extract the top $\mathcal{K}$ edges from the dependency forests that are most relevant for predicting relation r. We implement this based on Granger causality.[3]

Specifically, we use $\mathcal{L}_{\mathcal{G}}$ to denote the model error of $f_{\text{MRE}}(\cdot)$ when taking the $N * M$ dependency forests into account, and $\mathcal{L}_{\mathcal{G}\setminus\{e_k\}}$ represents the model error excluding the edge e_k from each dependency forest. According to Granger causality, we can quantify the causal contribution of edge e_k to our MRE task by the change in model error after removing edge e_k:

$$\Delta_{e_k} = \mathcal{L}_{\mathcal{G}\setminus\{e_k\}} - \mathcal{L}_{\mathcal{G}} \tag{5}$$

where Δ_{e_k} represent the causal contribution of edge e_k.

To calculate $\mathcal{L}_{\mathcal{G}}$ and $\mathcal{L}_{\mathcal{G}\setminus\{e_k\}}$, we first take the $N * M$ dependency forests $\mathcal{G}$ and $\mathcal{G} \setminus \{e_k\}$ as the input to the pre-trained model, respectively, and obtain their corresponding outputs $r_{\mathcal{G}}$ and $r_{\mathcal{G}\setminus\{e_k\}}$:

$$r_{\mathcal{G}} = f_{\text{MRE}}\left(\mathbf{F}^1, \ldots, \mathbf{F}^{N*M}\right) \tag{6}$$

$$r_{\mathcal{G}\setminus\{e_k\}} = f_{\text{MRE}}\left(\mathbf{F}^1 \setminus \{e_k\}, \ldots, \mathbf{F}^{N*M} \setminus \{e_k\}\right) \tag{7}$$

We then use the cross-entropy loss function to measure the model error, denoted as **CE**.

$$\mathcal{L}_{\mathcal{G}} = \mathbf{CE}\left(r, r_{\mathcal{G}}\right) \tag{8}$$

$$\mathcal{L}_{\mathcal{G}\setminus\{e_k\}} = \mathbf{CE}\left(r, r_{\mathcal{G}\setminus\{e_k\}}\right) \tag{9}$$

Finally, we filter out the edges with the top $\mathcal{K}$ causal contributions to form the causal explanation. In summary, our causal explanation dataset

[3]Granger causality describes the causal relationships between two (or more) variables. Specifically, if we are better able to predict variable $\widetilde{y}$ using all information U than excluding information about variable x, which means that the variable x helps predict variable $\widetilde{y}$. Then we say that x Granger-causes $\widetilde{y}$ (Granger, 1980), denoted by $x \rightarrow \widetilde{y}$.

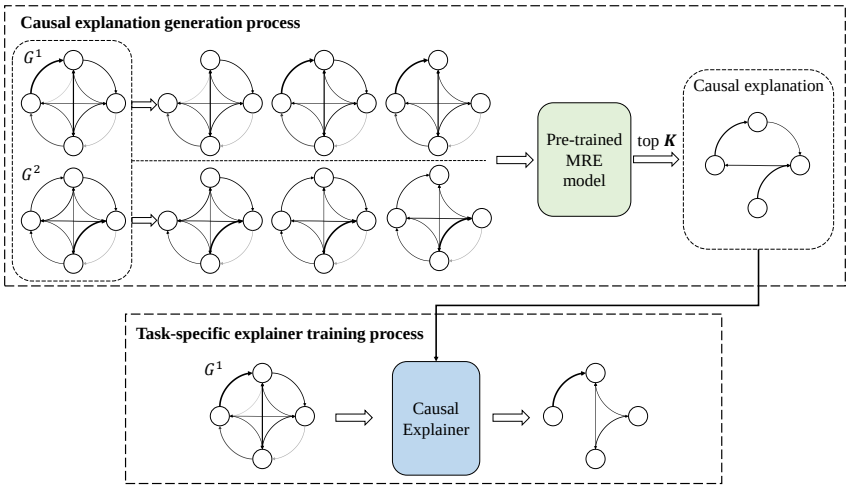

Figure 3: Illustration of training a task-specific causal explainer. Causal explanation generation process generates the causal explanations for the dependency forests using a pre-trained MRE model and the designed rules. Task-specific explainer training process trains a task-specific causal explainer with the *generated* causal explanation dataset.

is constructed with dependency forests and the corresponding causal explanations. Therefore, such a dataset is relevant to the specific MRE task.

Task-specific explainer training process. This process generates a task-specific explainer based on the causal explanation dataset. Following (Lin et al., 2021), we use an encoder-decoder architecture as the explainer. The encoder consists of several graph convolutional layers to aggregate information between neighbors in the dependency forest and learn node features. The decoder uses the inner product operation to obtain the explanation matrix. Specifically, the explanation matrix for $\boldsymbol{G}^i$ can be obtained by the explainer as:

$$\mathbf{X}^i = \sigma\left(f_{\text{GCN}}\left(\mathbf{F}^i, \mathbf{H}_0\right) \times f_{\text{GCN}}\left(\mathbf{F}^i, \mathbf{H}_0\right)^\top\right) \tag{10}$$

where $f_{\text{GCN}}(\cdot)$ denotes graph convolutional layers, $\mathbf{X}^i$ is the explanation matrix and each value in $\mathbf{X}^i$ represents the contribution of its corresponding edge to the prediction relation r, and σ is the activation function. **Causal pruning process.** Based on the pre-trained explainer, task-relevant explanation of the dependency forest can be obtained. Given the $\mathbf{F}^p$ calculated by Eq. 4 and the pre-trained explainer, the explanation matrix $\mathbf{X}^p$ corresponding to $\mathbf{F}^p$ can be calculated via Eq. 10. Causal pruning for $\mathbf{F}^p$ can be formulated as:

$$\hat{\mathbf{F}}^p = \text{softmax}\left(\mathbf{F}^p \odot (1 + \beta\, \mathbf{X}^p)\right) \tag{11}$$

where $\odot$ is the element-wise multiplication, and $\beta \in [0, 1]$ is a hyper-parameter to control the coefficient of the explanation matrix $\mathbf{X}^p$.

4.2 Dependency Forest Encoder

Given N pruned dependency forests, DCGCNs are used to encode information from the forest structure. For the p-th pruned dependency forest, which is represented by the adjacency matrix $\hat{\mathbf{F}}^p$. We use DCGCNs with L layers to aggregate information about neighbors in $\hat{\mathbf{F}}^p$, and the representation of node i at the l-th layer can be calculated as:

$$\boldsymbol{h}_{p_i}^{(l)} = \rho\left(\sum_j^n \hat{\mathbf{F}}_{ij}^p \left(\boldsymbol{W}_p^{(l)} \times \boldsymbol{g}_{p_j}^{(l)} + \boldsymbol{b}_p^{(l)}\right)\right) \tag{12}$$

where $\hat{\mathbf{F}}_{ij}^p$ denotes the weight between node i and node j in $\hat{\mathbf{F}}^p$. $\boldsymbol{g}_{p_j}^{(l)}$ denotes the information about node j in the p-th pruned dependency forest from all the preceding layers and can be obtained by the same way as Eq. 2.

Then, we concatenate the representations obtained from the N dependency forests and fuse them using a linear layer. This process can be formulated as follows:

$$\mathbf{H}_{\text{b}} = \text{Linear}\left([\mathbf{H}^1; \mathbf{H}^2; \ldots; \mathbf{H}^N]\right) \tag{13}$$

where $\mathbf{H}^i$ is the node representations obtained by DCGCNs for the i-th dependency forest, and $\mathbf{H}_{\text{b}}$ denotes the node representations of each block. M identical blocks are combined in the same way as above to obtain the final node representations for sentence $\boldsymbol{S}$, denoted as $\mathbf{H}$.

4.3 Relation Prediction

To predict the relations among entities, the max pooling mechanism is used. We obtain the global

	CPR	PGR
TRAIN	16107	11780
DEV	10030	-
TEST	14269	219

Table 1: The number of instances of CPR and PGR.

sentence representation $\boldsymbol{h_S}$ by applying the max pooling function to all the words in sentence $\boldsymbol{S}$:

$$\boldsymbol{h_S} = \text{MaxPooling}(\{\boldsymbol{h}_1, \ldots, \boldsymbol{h}_n\}) \quad (14)$$

where $\boldsymbol{h}_i$ is the feature vector of word w_i, and then obtain the representation of each entity by applying the max pooling function to the words that belongs to an entity mention (i.e., $\boldsymbol{E}_q$). Therefore, the entity representation of $\boldsymbol{E}_q$ can be obatined by:

$$\boldsymbol{h}_{\boldsymbol{E}_q} = \text{MaxPooling}(\{\boldsymbol{h}_i | w_i \in \boldsymbol{E}_q\}) \quad (15)$$

The sentence representation and entity representations are concatenated and fed into a feed-forward neural network (FFNN), and then we transform it into an m-dimensional vector $\boldsymbol{h_R}$ using a linear layer to make a prediction:

$$\boldsymbol{h_R} = \text{Linear}\left(\text{FFNN}\left([\boldsymbol{h_S}; \boldsymbol{h}_{\boldsymbol{E}_1}; \ldots; \boldsymbol{h}_{\boldsymbol{E}_\mathcal{Q}}]\right)\right) \quad (16)$$

where $\mathcal{Q}$ is 2 in the binary relational extraction task and is 3 in the ternary relational extraction task, m denotes the number of relations $\boldsymbol{R}$.

5 Experiment

5.1 Datasets

We evaluate our model on three datasets with two types of tasks: cross-sentence n-ary relation extraction and sentence-level relation extraction following (Guo et al., 2021).

For the cross-sentence n-ary relation extraction task, we use the dataset extracted by (Peng et al., 2017) based on PubMed. Most of the instances in this dataset contain multiple sentences, and the entities in the instances are cross-sentence. In detail, this dataset contains 6987 instances of ternary relations and 6087 instances of binary relations, each of them is divided into five folders according to (Song et al., 2018). The relation between entities in each instance belongs to one of the relation sets, including "resistance or non-response", "sensitivity", "response", "resistance", and "None". Following (Guo et al., 2021), we define two sub-tasks on this dataset: multi-class and binary-class relation extraction. For multi-class relation extraction, we keep the original dataset unchanged, and for binary-class relation extraction, we define the first four relations as "Yes" and the "None" as "No".

For the sentence-level relation extraction task, we use two datasets for Medical Relation Extraction, namely, BioCreative Vi CPR (CPR) (Krallinger et al., 2017) and Phenotype-Gene relation (PGR) (Sousa et al., 2019). CPR focuses on the relations between chemical components and human proteins, which contains six relation types ("CPR:3", "CPR:3", "CPR:4", "CPR:5", "CPR:6", "CPR:9", "None"). PGR focuses on whether human phenotypes are related to human genes, which contains two relation types ("TRUE" for related and "FALSE" for unrelated). The number of instances for train/dev/test sets of CPR and PGR datasets is shown in Table 1.

5.2 Implementation

During the causal explanation generation process, we use a pre-trained CP-GCN model *without* the task-specific causal pruning module as $f_{\text{MRE}}(\cdot)$ and choose 1/5 of the training set for the (Peng et al., 2017) dataset while the full training set for other datasets to generate the full dependency forests. Then, we set $\mathcal{K} = 20$ to construct causal explanation datasets.

For evaluation, we follow previous studies to use the test accuracy averaged over five cross validation folds for the cross-sentence n-ary task and F1 scores for the sentence-level task. Refer to the supplementary files for the details.

See Appendix A.1 for the hyper-parameter experiment on N, α, and β.

5.3 Results on Cross-Sentence N-Ary Relation Extraction Task

For the cross-sentence n-ary relation extraction task, We compare CP-GCN against two kinds of models and report the average test accuracies on the (Peng et al., 2017) dataset in table 2.

Tree: models use the 1-best dependency tree. DAG LSTM, GRN, and GCN(Full) use the full dependency tree directly, while GCN(Pruned) generates a pruned dependency tree with some rules (Zhang et al., 2018). Besides, DAG LSTM uses graph-structure LSTM to encode the dependency tree, while GRN and GCN use graph recurrent networks and graph convolutional networks, respectively.

Forest: models construct dependency forests. ACGCN treats a fully connected graph obtained by

Syntax Type	Model	Binary-class				Multi-class			
		Ternary		Binary		Ternary		Binary	
		Single	Cross	Single	Cross	Single	Cross	Single	Cross
Tree	DAG LSTM (Peng et al., 2017)	77.9	80.7	74.3	76.5	-	-	-	-
	GRN (Song et al., 2018)	80.3	83.2	83.5	83.6	-	71.7	-	71.7
	GCN(Full) (Zhang et al., 2018)	84.3	84.8	84.2	83.6	-	77.5	-	74.3
	GCN(Pruned) (Zhang et al., 2018)	85.8	85.8	83.8	83.7	-	78.1	-	73.6
Forest	AGGCN (Guo et al., 2019a)	87.1	87	85.2	85.6	-	79.7	-	77.4
	AGGCN* (Guo et al., 2019a)	86.3	87.2	86.3	85.8	77.7	78.7	77.7	77.3
	LF-GCN (Guo et al., 2021)	88	88.4	86.7	**87.1**	-	81.5	-	79.3
	LF-GCN* (Guo et al., 2021)	88.2	88.3	87	86.3	82.9	83.9	80	79.6
	AC-GCN (Qian et al., 2021)	88.8	88.8	86.8	86.5	-	84.6	-	**81**
	CP-GCN(ours)	**89.5**	**89.1**	**87.3**	86.5	**84.3**	**84.9**	**81**	80.1

Table 2: Average test accuracies on the (Peng et al., 2017) dataset for binary-class n-ary relation extraction and multi-class n-ary relation extraction. "Ternary" denotes drug-gene-mutation tuple and "Binary" denotes drug-mutation pair. "Single" means considering the instances within a single sentence, while "Cross" means considering all instances. Models with * indicate the accuracy of our reimplementation on their released implementation.

multi-head attention as a forest. LF-GCN automatically generates latent forests using multi-head attention and MMT. AC-GCN generates dependency forests with multi-head attention and encodes them with a 2D convolutional network.

As shown in Table 2, our proposed CP-GCN model achieves state-of-the-art performance in most settings. Specifically, the model using the pruned dependency tree performs better than those using the full dependency tree, suggesting that noisy information does exist in the 1-best dependency tree. In addition, the forest structure shows an advantage on this task, while CP-GCN surpasses the current state-of-the-art forest-structured model (AC-GCN) by 0.7 and 0.3 points on the binary-class ternary relation extraction task. The multi-class n-ary relation extraction task in (Peng et al., 2017) dataset is more challenging due to the unbalanced distribution of each relation, and CP-GCN can consistently achieve comparable performance.

5.4 Results on Sentence-Level Relation Extraction Task

For the sentence-level relation extraction task, we implement our approach on the CPR and PGR datasets and compare it against state-of-the-art models. We classify these models into three groups according to their syntax type.

None: models do not use tree or forest structures. Att-GRU adds a self-attention layer to GRU, and Bran uses a bi-affine self-attention model to capture interactions in sentences. BioBERT is a biomedical pre-trained language representation model.

Tree: models use the 1-best dependency tree. GCN, Tree-DDCNN, and Tree-GRN encode the full tree with GCN, DDCNN, and GRN, respectively. BO-LSTM prunes the tree, retaining only the shortest dependency path.

Syntax Type	Model	F1
None	Att-GRU (Liu et al., 2017)	49.5
	Bran (Verga et al., 2018)	50.8
Tree	GCN (Zhang et al., 2018)	52.2
	Tree-DDCNN (Jin et al., 2020)	50.3
	Tree-GRN (Jin et al., 2020)	51.4
Forest	Edgewise-GRN (Song et al., 2019)	53.4
	KBest-GRN (Song et al., 2019)	52.4
	AGGCN (Guo et al., 2019a)	56.7
	ForestFT-DDCNN (Jin et al., 2020)	55.7
	LF-GCN (Guo et al., 2021)	58.9
	AC-GCN (Qian et al., 2021)	65.8
	CP-GCN(ours)	**67.3**

Table 3: Main results on CPR.

Forest: models construct dependency forests. Edgewise-GRN chooses edges with weights greater than the pre-defined threshold to form the dependency forest. KBest-GRN constructs the forest by aggregating K-best trees. ForestFT-DDCNN generates forests with a learnable dependency parser.

The results of CPR and PGR datasets are shown in Table 3 and Table 4. CP-GCN achieves state-of-the-art performance on both datasets. F1 score increases by 1.5 and 0.5 on the CPR and PGR datasets, respectively. Compared to models with forest structure, CP-GCN performs significantly better than both models with a bias towards syntactic information (Edgewise-GRN, KBest-GRN, and ForestFT-DDCNN) and models using almost only semantic information (AGGCN, LF-GAN, and AC-GCN), which demonstrates the effectiveness of our proposed CSFG method.

5.5 Analysis and Discussion

Ablation study. To validate the effectiveness of the ingredients of CP-GCN, i.e., the semantic em-

Syntax Type	Model	F1
None	BioBERT (Lee et al., 2020)	67.2
Tree	BO-LSTM (Lamurias et al., 2019)	52.3
	GCN (Zhang et al., 2018)	81.3
	Tree-GRN (Jin et al., 2020)	78.9
Forest	Edgewise-GRN (Song et al., 2019)	83.6
	KBest-GRN (Song et al., 2019)	85.7
	AGGCN (Guo et al., 2019a)	89.3
	ForestFT-DDCNN (Jin et al., 2020)	89.3
	LF-GCN (Guo et al., 2021)	91.9
	AC-GCN (Qian et al., 2021)	92.4
	CP-GCN(ours)	**92.9**

Table 4: Main results on PGR.

Model	F1
CP-GCN	67.3
-semantic embedding module	66.7
-task-specific causal pruning module	65.7

Table 5: An ablation study for CP-GCN on CPR dataset.

bedding module and the task-specific causal pruning module, we conduct the ablation study on CPR. We train the complete CP-GCN, an ablation model without the semantic embedding module, and another ablation model without the task-specific causal pruning module, respectively. Our experimental results are reported in Table 5. We observe that the performance of the model dropped (compared with complete CP-GCN) regardless of which module is removed, suggesting that both modules can help construct dependency forests that are more conducive to predicting relation. Comparing these two modules, the removal of the task-specific causal pruning module has a greater impact on performance, which suggests that the proposed causal pruning method can effectively distinguish vital information from noise.

Performance against sentence length. Figure 4 compares the F1 scores of our CP-GCN model and the LF-GCN model (Guo et al., 2021) under different sentence lengths. Following (Guo et al., 2021), We divide the test set of CPR into three groups ((0,25], (25,50], >50) based on sentence length. In general, CP-GCN outperforms LF-GCN against various sentence lengths. Otherwise, our model achieves a significant improvement on the more challenging long sentences, which demonstrates the ability of our model to capture long-range dependencies. Moreover, the dependency forests of the long sentences are more sophisticated, thus indicating that the task-specific causal explainer is able to extract task-relevant information from the sophisticated graph structure.

Case study. To further validate the efficiency of CP-GCN, we conduct a case study on an example sentence "***Aspirin*** induced autophagy, a feature of ***mTOR*** inhibition", which can be correctly predicted by our model to be a "down regulator" relation between ***Aspirin***($\boldsymbol{E_1}$) and ***mTOR***($\boldsymbol{E_2}$). Figure 5(a) shows its 1-best dependency tree, and Figure 5(b) visualizes the top 10 edges with the highest causal weights in the pruned dependency forest generated by the proposed CSFG and the thicker lines referring to higher causal weights. In this example, the connection between "*induced*" and "*feature*" enhances in the pruned dependency forest, and we reckon the latent reason is that CP-GCN can capture richer semantic information. We observe that there exists a strong connection between "*autophagy*" and "*feature*" in the pruned dependency forest, which improves the prediction of the relation between ***Aspirin*** and ***mTOR***, supporting the effectiveness of the task-specific causal explainer.

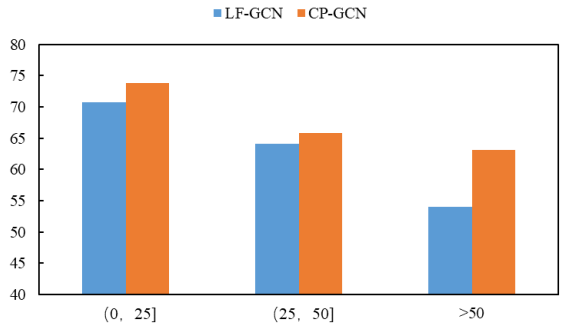

Figure 4: F1 scores against sentence length. The results on LF-GCN are reproduced based on its released implementation.

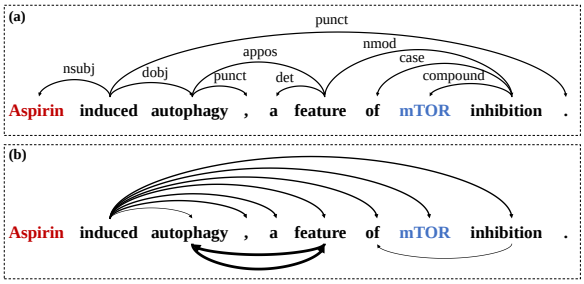

Figure 5: Visualizations of (a) 1-best dependency tree and (b) top 10 highest causal weight edges of the pruned dependency forest for the example input, where thicker lines denote the connections with higher causal weights.

6 Conclusion

In this paper, we introduce a novel approach for the medical relation extraction task, namely CP-GCN, which proposes a causality-pruned dependency forest enriched with semantic and syntactic information. We first construct dependency forests by incorporating semantic information into the dependency tree generated by the off-the-shelf parser.

Then, a task-specific causal explainer is trained to prune the dependency forests. Experiments on the benchmark medical datasets demonstrate the superiority of CP-GCN over the state-of-the-art methods for the medical relation extraction task.

Acknowledgements

This work is supported by the National Key Research and Development Program of China (No. 2019YFB1405100).

References

Aditya Chattopadhyay, Piyushi Manupriya, Anirban Sarkar, and Vineeth N Balasubramanian. 2019. Neural network attributions: A causal perspective. In *International Conference on Machine Learning*, pages 981–990. PMLR.

Guimin Chen, Yuanhe Tian, Yan Song, and Xiang Wan. 2021. Relation extraction with type-aware map memories of word dependencies. In *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*, pages 2501–2512.

Anupam Datta, Shayak Sen, and Yair Zick. 2016. Algorithmic transparency via quantitative input influence: Theory and experiments with learning systems. In *2016 IEEE symposium on security and privacy (SP)*, pages 598–617. IEEE.

Clive WJ Granger. 1969. Investigating causal relations by econometric models and cross-spectral methods. *Econometrica: journal of the Econometric Society*, pages 424–438.

Clive WJ Granger. 1980. Testing for causality: A personal viewpoint. *Journal of Economic Dynamics and control*, 2:329–352.

Zhijiang Guo, Guoshun Nan, Wei Lu, and Shay B Cohen. 2021. Learning latent forests for medical relation extraction. In *Proceedings of the Twenty-Ninth International Conference on International Joint Conferences on Artificial Intelligence*, pages 3651–3657.

Zhijiang Guo, Yan Zhang, and Wei Lu. 2019a. Attention guided graph convolutional networks for relation extraction. *arXiv preprint arXiv:1906.07510*.

Zhijiang Guo, Yan Zhang, Zhiyang Teng, and Wei Lu. 2019b. Densely connected graph convolutional networks for graph-to-sequence learning. *Transactions of the Association for Computational Linguistics*, 7:297–312.

Lifeng Jin, Linfeng Song, Yue Zhang, Kun Xu, Weiyun Ma, and Dong Yu. 2020. Relation extraction exploiting full dependency forests. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pages 8034–8041.

Thomas N Kipf and Max Welling. 2016. Semi-supervised classification with graph convolutional networks. *arXiv preprint arXiv:1609.02907*.

Terry Koo, Amir Globerson, Xavier Carreras Pérez, and Michael Collins. 2007. Structured prediction models via the matrix-tree theorem. In *Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning (EMNLP-CoNLL)*, pages 141–150.

Martin Krallinger, Obdulia Rabal, Saber A Akhondi, Martın Pérez Pérez, Jesús Santamaría, Gael Pérez Rodríguez, Georgios Tsatsaronis, Ander Intxaurrondo, José Antonio López, Umesh Nandal, et al. 2017. Overview of the biocreative vi chemical-protein interaction track. In *Proceedings of the sixth BioCreative challenge evaluation workshop*, volume 1, pages 141–146.

Andre Lamurias, Diana Sousa, Luka A Clarke, and Francisco M Couto. 2019. Bo-lstm: classifying relations via long short-term memory networks along biomedical ontologies. *BMC bioinformatics*, 20(1):1–12.

Phong Le and Willem Zuidema. 2015. The forest convolutional network: Compositional distributional semantics with a neural chart and without binarization. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 1155–1164.

Jinhyuk Lee, Wonjin Yoon, Sungdong Kim, Donghyeon Kim, Sunkyu Kim, Chan Ho So, and Jaewoo Kang. 2020. Biobert: a pre-trained biomedical language representation model for biomedical text mining. *Bioinformatics*, 36(4):1234–1240.

Linfeng Li, Peng Wang, Jun Yan, Yao Wang, Simin Li, Jinpeng Jiang, Zhe Sun, Buzhou Tang, Tsung-Hui Chang, Shenghui Wang, et al. 2020. Real-world data medical knowledge graph: construction and applications. *Artificial intelligence in medicine*, 103:101817.

Wanyu Lin, Hao Lan, and Baochun Li. 2021. Generative causal explanations for graph neural networks. In *International Conference on Machine Learning*, pages 6666–6679. PMLR.

Sijia Liu, Feichen Shen, Yanshan Wang, Majid Rastegar-Mojarad, Ravikumar Komandur Elayavilli, Vipin Chaudhary, and Hongfang Liu. 2017. Attention-based neural networks for chemical protein relation extraction. *Training*, 1020(25.247):4157.

Christopher D. Manning, Mihai Surdeanu, John Bauer, Jenny Rose Finkel, Steven Bethard, and David McClosky. 2014. The stanford corenlp natural language processing toolkit. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics, ACL 2014, June 22-27, 2014, Baltimore, MD, USA, System Demonstrations*, pages 55–60. The Association for Computer Linguistics.

Judea Pearl. 1985. Bayesian netwcrks: A model of self-activated memory for evidential reasoning. In *Proceedings of the 7th conference of the Cognitive Science Society, University of California, Irvine, CA, USA*, pages 15–17.

Judea Pearl. 2009. *Causality*. Cambridge university press.

Nanyun Peng, Hoifung Poon, Chris Quirk, Kristina Toutanova, and Wen-tau Yih. 2017. Cross-sentence n-ary relation extraction with graph lstms. *Transactions of the Association for Computational Linguistics*, 5:101–115.

Mengyuan Qian, Jian Wang, Hongfei Lin, Di Zhao, Yijia Zhang, Wentai Tang, and Zhihao Yang. 2021. Auto-learning convolution-based graph convolutional network for medical relation extraction. In *China Conference on Information Retrieval*, pages 195–207. Springer.

Chris Quirk and Hoifung Poon. 2016. Distant supervision for relation extraction beyond the sentence boundary. *arXiv preprint arXiv:1609.04873*.

Maya Rotmensch, Yoni Halpern, Abdulhakim Tlimat, Steven Horng, and David Sontag. 2017. Learning a health knowledge graph from electronic medical records. *Scientific reports*, 7(1):1–11.

Patrick Schwab and Walter Karlen. 2019. Cxplain: Causal explanations for model interpretation under uncertainty. *Advances in Neural Information Processing Systems*, 32.

Linfeng Song, Yue Zhang, Daniel Gildea, Mo Yu, Zhiguo Wang, and Jinsong Su. 2019. Leveraging dependency forest for neural medical relation extraction. *arXiv preprint arXiv:1911.04123*.

Linfeng Song, Yue Zhang, Zhiguo Wang, and Daniel Gildea. 2018. N-ary relation extraction using graph state lstm. *arXiv preprint arXiv:1808.09101*.

Diana Sousa, André Lamúrias, and Francisco M Couto. 2019. A silver standard corpus of human phenotype-gene relations. *arXiv preprint arXiv:1903.10728*.

Kai Sun, Richong Zhang, Yongyi Mao, Samuel Mensah, and Xudong Liu. 2020. Relation extraction with convolutional network over learnable syntax-transport graph. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pages 8928–8935.

Yuanhe Tian, Guimin Chen, Yan Song, and Xiang Wan. 2021. Dependency-driven relation extraction with attentive graph convolutional networks. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 4458–4471.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. *Advances in neural information processing systems*, 30.

Petar Velickovic, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Lio, and Yoshua Bengio. 2017. Graph attention networks. *stat*, 1050:20.

Patrick Verga, Emma Strubell, and Andrew McCallum. 2018. Simultaneously self-attending to all mentions for full-abstract biological relation extraction. *arXiv preprint arXiv:1802.10569*.

Poorya Zaremoodi and Gholamreza Haffari. 2017. Incorporating syntactic uncertainty in neural machine translation with forest-to-sequence model. *arXiv preprint arXiv:1711.07019*.

Yuhao Zhang, Peng Qi, and Christopher D Manning. 2018. Graph convolution over pruned dependency trees improves relation extraction. *arXiv preprint arXiv:1809.10185*.

GuoDong Zhou, Jian Su, Jie Zhang, and Min Zhang. 2005. Exploring various knowledge in relation extraction. In *Proceedings of the 43rd annual meeting of the association for computational linguistics (acl'05)*, pages 427–434.

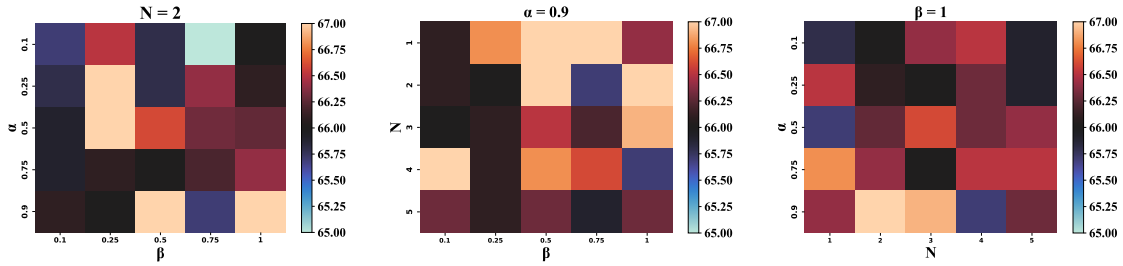

(a) (b) (c)

Figure 6: F1 scores with different hyper-parameter settings.

A Appendix

A.1 Hyper-Parameter Experiment

We perform several experiments on the CPR dataset to study the influence of the hyper-parameters in our proposed CP-GCN model, and the results are shown in Figure 6. The hyper-parameter α balances semantic and syntactic information in the dependency forest. The hyper-parameter β balances the impact of the task-specific causal pruning module. The hyper-parameter N represents the richness of semantic information. As (a), (b), and (c) are shown in Figure 6, our proposed CP-GCN model achieves comparable performance in most settings, which indicates the robustness of our model. Specifically, CP-GCN achieves the highest F1 score 67.3 with $N = 2, \alpha = 0.9$, and $\beta = 1$. As shown in Figure 6(c), when N decreases to 1, i.e. the semantic information decreases, CP-GCN performs best when the weight of the dependency tree, α, decreases as well, suggesting that our model is able to balance the syntactic and semantic information. As shown in Figure 6(a), when the weight of dependency tree α increases, CP-GCN performs best when the weight of task-specific causal pruning module β increases as well. This demonstrates that there is indeed some noise in the dependency tree and our proposed task-specific causal pruning module can remove task-irrelevant information.

Aspect-based Sentiment Analysis as Machine Reading Comprehension

Yifei Yang[1,2], **Hai Zhao**[1,2,*]
[1]Department of Computer Science and Engineering, Shanghai Jiao Tong University
[2]MoE Key Lab of Artificial Intelligence, AI Institute, Shanghai Jiao Tong University
{yifeiyang}@sjtu.edu.cn, zhaohai@cs.sjtu.edu.cn

Abstract

Existing studies typically handle aspect-based sentiment analysis by stacking multiple neural modules, which inevitably result in severe error propagation. Instead, we propose a novel end-to-end framework, **MRCOOL**: **MRC**-Pr**O**mpt m**O**de**L** framework, where numerous sentiment aspects are elicited by a machine reading comprehension (MRC) model and their corresponding sentiment polarities are classified in a prompt learning way. Experiments show that our end-to-end framework consistently yields promising results on widely-used benchmark datasets which significantly outperform existing state-of-the-art models or achieve comparable performance.[1]

1 Introduction

Compared with traditional sentence-level or document-level sentiment analysis tasks, aspect-based sentiment analysis (ABSA) requires finer grained analysis on the texts and extracting more detailed information (Liu, 2012; Pontiki et al., 2014a). ABSA contains many subtasks, such as aspect category detection, opinion term extraction (OE), etc. Aspect term extraction (AE) and aspect-level sentiment classification (SC) are two elemental subtasks of ABSA. AE means extracting the sentiment aspects from a given plain sentence and SC implies recognizing the sentiment polarities of the given aspects in a sentence. Combining the above two subtasks, aspect term extraction and sentiment classification (AESC) establishes the third fundamental subtask. The AE, SC and AESC for the sentence *Excellent food, although the interior could use some help.* are given in Figure 1.

In general, the existing mainstream approaches can be roughly divided into two brands. The first employs the two-stage method which first accomplishes the AE and is followed by another model to perform SC, thus achieving the AESC (Yu et al., 2018; Hu et al., 2019; Fan et al., 2019). The second tries to fulfill the three subtasks by a more unified methodology which extracts the terms and their corresponding polarities in a joint or interactive way (Liu et al., 2016; Wang et al., 2017; Li and Lam, 2017; Fan et al., 2018; He et al., 2019; Luo et al., 2019; Li et al., 2019; Peng et al., 2020; Chen and Qian, 2020; Chen et al., 2020; Wan et al., 2020). However, the above mentioned schemes perform ABSA task by stacking recurrent neural networks (RNN) or attention mechanisms and usually lead to too complex models.

Excellent food, although the interior could use some help.

AE	SC
• *food* • *interior*	• *food*-positive • *interior*-negative
Aspect Term Extraction and Sentiment Classification • (*food*, positive) • (*interior*, negative)	

Figure 1: An example of AE, SC and AESC. AE needs to identify the aspect terms *food* and *interior*. When given these two terms, SC should recognize the sentiment polarities of them as positive and negative separately. AESC is going to complete these two tasks from the given sentence.

In recent years, machine reading comprehension (Li et al., 2020; Liu et al., 2020; Su et al., 2020; Mao et al., 2021) quickly become a hot topic among various challenging natural language understanding tasks. Generally, MRC model may give a proper answer for a query based on a given passage. There are various types of MRC tasks according to the desired answer forms, among which span MRC or extractive MRC draw quite a lot of attention (Glass et al., 2019; Wu et al., 2019; Zhang et al., 2020). Most of the progress for MRC may be attributed to the latest pre-trained language models (PrLMs)

*Corresponding author. This work was supported by Key Projects of National Natural Science Foundation of China (U1836222 and 61733011).

[1]https://github.com/yangyifei729/MRC4absa

Proceedings of the 29th International Conference on Computational Linguistics, pages 2461–2471
October 12–17, 2022.

The waiters were very professional, courteous and attentive.

AE as MRC
Input to PrLM (before embedded)
[CLS] find the sentiment aspect terms in the text. [SEP] *The waiters were very professional, courteous and attentive.*
Output of MRC: *waiters*

SC as Prompt Learning
Input to PrLM (before embedded)
[CLS] *The waiters were very professional, courteous and attentive.* [SEP] I felt the *waiters* was [MASK].
[CLS] *The waiters were very professional, courteous and attentive.* [SEP] I [MASK] the *waiters*.
[CLS] *The waiters were very professional, courteous and attentive.* [SEP] The *waiters* made me feel [MASK].
[CLS] *The waiters were very professional, courteous and attentive.* [SEP] The *waiters* is [MASK].
Output of Prompt Learning: positive

Figure 2: An example of modeling AE as MRC task and SC as prompt learning task. This figure shows the final input of two subtasks to the PrLM.

such as BERT (Devlin et al., 2019). For enhancing multiple downstream tasks including MRC, PrLM may serve as a powerful enough encoder in the corresponding model for effectively capturing salient features from input text (they are passage and query in MRC).

Prompt learning is a natural manner to leverage the knowledge of PrLM which requires adapting the downstream tasks into a self-supervised learning task of the corresponding PrLM. For example, Chen et al. (2021) convert the relation extraction task to the masked language model (MLM) task of BERT and Sun et al. (2021) apply the next sentence prediction (NSP) task to carry out the downstream tasks. Even though MRC and prompt learning can take advantage of the knowledge of PrLM and facilitate the performance on downstream tasks, these paradigms still have obvious defects, (1) The query for MRC can severely inhibit the performance of downstream tasks, but the construction of query is currently based on templates or empiric which leads to huge labor costs and it is not guaranteed to find the best matched query. (2) For the prompt learning, after getting the output of PrLM, a standardized process is applying a verbalizer to project the original labels to the label words of the downstream task. When adopting MLM to perform tagging tasks, the current verbalizer selects the probability distribution of a few specific words from the output word-embedding of `[MASK]` token to determine the final prediction. This manner makes the verbalizer very sparse and can not make full use of the knowledge of PrLM.

To alleviate the above issues, we propose a novel end-to-end framework named **MRCOOL** to handle the AE, SC and AESC once for all. For AE, we model it as an MRC task and propose a query encoder to search for the possible latent optimal query in a continuous space. We treat SC as a prompt learning task and apply a concise MLP verbalizer to reduce the sparsity. Our experiments are conducted on three widely-used benchmark datasets. Results show that our framework outperforms the current methods or gets comparable performance.

2 MRCOOL Framework

2.1 Task Formulation

The three subtasks can be formulated as a tuple extraction task. Given an input sentence $X = \{x_1, x_2, ..., x_n\}$ of length n, the corresponding output is $Y = \{(a_1, p_1), (a_2, p_2), ..., (a_m, p_m)\}$ where a_i indicates an aspect and p_i represents its polarity.

Given a training dataset $\mathcal{D} = \{(X_1, Y_1), (X_2, Y_2), ..., (X_{|\mathcal{D}|}, Y_{|\mathcal{D}|})\}$, the purpose of our framework is to maximize the likelihood:

$$L(\mathcal{D}) = \prod_{i=1}^{|\mathcal{D}|} \prod_{(a_j, p_j) \in Y_i} P\left((a_j, p_j) \mid X_i\right) \quad (1)$$

The following section 2.2 and section 2.3 will introduce our MRC modeling for AE and prompt learning modeling for SC. An example of our modeling is given in Figure 2.

The extractive MRC task requires to extract the answer $A = \{a_1, a_2, ..., a_m\}$ from a passage $G = \{g_1, g_2, ..., g_n\}$ by answering a given query Q and each a_i in A is a span of G. The goal of a given model $\mathcal{M}$ is to maximize the likelihood:

$$P(A \mid G, Q) = \prod_{a_i \in A} P\left(a_i \mid G, Q\right) \quad (2)$$

In this paper, we model AE as an extractive MRC task. For a sentence $X = \{x_1, x_2, ..., x_n\}$ whose aspects are $A = \{a_1, a_2, ..., a_m\}$ where a_i is a span of X, we regard X as the passage G and each sentiment aspect a_i as a corresponding answer. Following the procedure of MRC, we desire to find each sentiment aspect a_i by asking model $\mathcal{M}$ a query Q. However, the query for the standard MRC task is given by the datasets, but AE datasets do not contain such an element. To let our task inputs compatible to the adopted MRC model, we construct a dedicated query Q. A large number of studies also have shown that the query has a significant impact on the performance of MRC no matter if such a query keeps a meaningful input or not. Following (Liu et al., 2021), we set a fixed-initialized query Q whose embedding can be optimized during the training process and search for the optimal best matching query in a continuous space for each sentence X.

2.2 Aspect Term Extraction as Machine Reading Comprehension

In detail, our MRC module takes a PrLM M as a backbone. The input X will be transformed into word embedding $W = \{w_1, w_2, ..., w_n\}$ by the embedding layer $\mathcal{E}$ of M. Then we initialize a query $Q = \{q_1, q_2, ..., q_m\}$. Since the alternative query can largely influence the performance of the MRC, we add an encoder module to more effectively capture the optimal potential query embedding. The query encoder module consists of a randomly initialized embedding layer $\mathcal{E}'$, a Bi-direction LSTM layer and a double-layer MLP activated by RELU function (Glorot et al., 2011). The embedding r_i of q_i can be formalized as:

$$r_i = \mathrm{MLP}\left(\left[\mathrm{BiLSTM}\left(\mathcal{E}'(q_{0:i})\right) : \mathrm{BiLSTM}\left(\mathcal{E}'(q_{i:m})\right)\right]\right) \quad (3)$$

Then we combine the encoded embedding of query and sentence to form the input sequence for M. The embeddings of two special tokens `[CLS]` and `[SEP]` are bound to be inserted in the sequence. The final input sequence is like:

$$\{\mathcal{E}([\texttt{CLS}]), r_1, r_2, ..., r_m, \mathcal{E}\left([\texttt{SEP}]\right), w_1, w_2, ..., w_n\}$$

After feeding the sequence into M, the context representation $S \in \mathbb{R}^{n \times \|\mathcal{V}\|}$ of W is the only output we need for the next steps, where the $\mathcal{V}$ is the vocabulary of PrLM.

Subsequently, we carry out the selection of aspects. We adopt two independent binary classifiers to predict whether a token is a start or end position of an aspect following (Li et al., 2020). For the start position prediction, we first project the S into the dimension of $\mathbb{R}^{n \times 2}$ by a learnable weight $O_{start} \in \mathbb{R}^{\|\mathcal{V}\| \times 2}$ and get S'. Then, we apply the softmax to each row of S' to form a probability distribution which indicates the probability of every token to be the start word of an aspect. The above process can be formalized as:

$$P_{start} = \mathrm{softmax}_{\text{each row}}\left(S \cdot O_{start}\right) \in \mathbb{R}^{n \times 2} \quad (4)$$

Following the same process, the probability distribution P_{end} of whether a token to be the end position of an aspect can be attained by the learnable weight $O_{end} \in \mathbb{R}^{\|\mathcal{V}\| \times 2}$.

The next step is to match the start position and end position in order to extract the final aspects. By the P_{start} and P_{end}, we can get the start as well as end positions simply according to the argmax function and store them into two sets, these are:

$$\begin{aligned} I_{start} &= \left\{ i \mid \mathrm{argmax}\left(P_{start}^{(i)}\right) = 1, i = 1, \cdots, n \right\} \\ J_{end} &= \left\{ j \mid \mathrm{argmax}\left(P_{end}^{(j)}\right) = 1, j = 1, \cdots, n \right\} \end{aligned} \quad (5)$$

The start position $i_{start} \in I_{start}$ means the $x_{i_{start}}$ should be the start token of an aspect and the $j_{end} \in J_{end}$ implies the end token $x_{i_{end}}$ of an aspect. Thus, we will train a sigmoid classifier to predict the match possibility of the i_{start}, j_{end} to be the boundary of one aspect, that is:

$$P_{i_{start}, j_{end}} = \mathrm{sigmoid}\left(m \cdot \mathrm{concat}\ \left(S_{i_{start}}, S_{j_{end}}\right)\right) \quad (6)$$

where $m \in \mathbb{R}^{1 \times 2\|\mathcal{V}\|}$.

During the training process, we leverage the CrossEntropy loss as below:

$$\begin{aligned} \mathcal{L}_{star} &= \mathrm{CrossEntropy}\left(P_{start}, G_{start}\right) \\ \mathcal{L}_{end} &= \mathrm{CrossEntropy}\left(P_{end}, G_{end}\right) \\ \mathcal{L}_{match} &= \mathrm{CrossEntropy}\left(P_{start,end}, G_{start,end}\right) \end{aligned} \quad (7)$$

where the G_{start}, G_{end} and $G_{start,end}$ represent the golden labels. The total loss of this MRC module is the weighted sum of the above three losses:

$$\mathcal{L}_{MRC} = \alpha\mathcal{L}_{start} + \beta\mathcal{L}_{end} + \gamma\mathcal{L}_{match} \quad (8)$$

where the $\alpha, \beta, \gamma \in [0, 1]$ are three hyper-parameters to control the contributions of each objective function.

2.3 Aspect-level Sentiment Classification as Prompt Learning

Incorporating the output from the MRC module, we model the SC subtask as prompt learning, which allows us to transfer a classification problem into the form of predicting the `[MASK]` token contained in a prompt sentence Q as a pre-specified word w. Namely, for a given text X and its label Y, the purpose of prompt learning model $\mathcal{M}$ is to maximize the likelihood:

$$P(Y \mid X) = P([\texttt{MASK}] = \texttt{w} \mid X, Q) \quad (9)$$

For a sentence X and one of its aspect a_j, we insert a_j into a pre-defined template $T = t_1, ..., a_j, ..., [\texttt{MASK}], ..., t_m$ to form a prompt Q. Then we feed the sequence `[CLS]` X `[SEP]` Q into a module $\mathcal{M}$ to measure the likelihood of a_j to be classified as polarity p_j:

$$P(p_j \mid a_j, X) = P([\texttt{MASK}] = \texttt{w} \mid [\texttt{CLS}]X[\texttt{SEP}]Q) \quad (10)$$

In detail, first of all, we need to construct the templates of prompts cautiously as the performance of prompt learning is very sensitive to the choice of prompts. Following (Seoh et al., 2021), we select four efficient prompt templates:

- T_0 =I felt the a_i was `[MASK]`.
- T_1 =I `[MASK]` the a_i.
- T_2 =The a_i made me feel `[MASK]`.
- T_3 =The a_i is `[MASK]`.

When given a sentence X and one of its aspect a_j, we fill $T_i (i \in \{1, 2, 3, 4\})$ to get four prompt Q_i and feed the sequence `[CLS]` X `[SEP]` Q_i into a PrLM, thus to attain its context representation C_i.

All we need is the representation $C_{i,[MASK]} \in \mathbb{R}^{\|\mathcal{V}\|}$ of `[MASK]` which contains the prediction information. To aggregate the prediction outcomes of the four prompts, we simply add them all up:

$$C_{[MASK]} = \sum_{i=0}^{3} C_{i,[MASK]} \quad (11)$$

The current methods (Seoh et al., 2021; Zhang et al., 2021) directly take the probabilities of predicting `[MASK]` as several representative words to extract final sentiment polarities. For instance, Seoh et al. (2021) map the probabilities of three words $\{good, bad, ok\}$ to the probabilities of three sentiment polarities $\{positive, negative, neutral\}$ towards the template T_0:

$$P([\texttt{MASK}] = good|X, a_i) = P(p_i = positive|X, a_i)$$
$$P([\texttt{MASK}] = bad|X, a_i) = P(p_i = negative|X, a_i)$$
$$P([\texttt{MASK}] = ok|X, a_i) = P(p_i = neutral|X, a_i)$$

This seems to be an extremely blunt approach, but we argue that it will increase the sparsity of the model and can not plausibly exert all the intelligence of $C_{[MASK]}$. Following such practice, we should exhaust all the tokens that can indicate sentiment to ameliorate this drawback. For example, if we want to predict the positive polarity, in addition to *good*, we also need to consider the words such as *nice*, *excellent*, *perfect*, etc. Obviously, such tedious work is not acceptable. In that case, we propose to utilize a double-layer MLP head activated by a RELU function to address the limitations mentioned above:

$$P(p_i|a_i, X) = \text{MLP}(C_{[MASK]}) \quad (12)$$

where the input dimension of MLP is $\|\mathcal{V}\|$ and the output dimension is set to 3 indicating the probability distribution over three polarities.

At this point, we merely need to follow the idea of the maximum likelihood method and apply CrossEntropy loss to calculate $\mathcal{L}_{PL}$ for prompt learning, thus training MLP while fine-tuning BERT.

2.4 Training

The above MRC model with prompt learning can be trained together as a multi-task learning. We aggregate the two losses together for conducting back propagation and the total loss can be formalized as:

$$\mathcal{L}_{Total} = \mathcal{L}_{MRC} + \mathcal{L}_{PL} \quad (13)$$

2.5 Inference

When given trained MRC model for inferencing the AE result, the start and end positions are separately decided according to I_{start} and J_{end} (Eq.5) firstly. The following sigmoid classifier (Eq.6) will detect

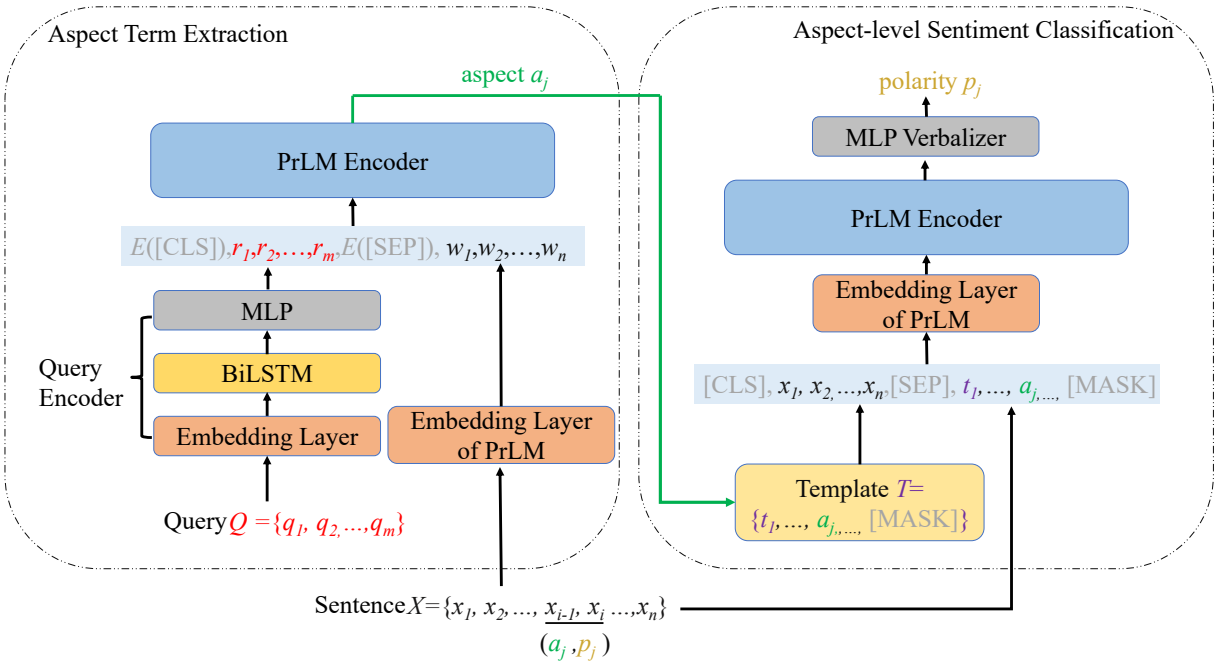

Figure 3: The architecture of our proposed **MRCOOL** framework.

	Lap14		Res14		Res15	
	#s	#a	#s	#a	#s	#a
train	3048	2373	3044	3699	1315	1199
test	800	654	800	1134	685	542

Table 1: The statistics of the three datasets (Wang et al., 2017). #s and #a denote the numbers of sentences and aspect terms.

the final start-end position combinations by I_{start} and J_{end}. As for inferencing the SC result, the final polarity $\hat{p}$ of an aspect is:

$$\begin{aligned} \hat{p} &= \text{argmax}\, P(p_i|X, a_i) \\ &= \text{argmax}\, \text{MLP}(C_{[MASK]}) \end{aligned} \quad (14)$$

For AESC, the above two inference processes are united to obtain the final result. When given a sentence X, our framework first inputs it into the MRC model and receives the candidate aspect terms set A. Then, each a_i in A is enumerated to construct four templates with its homologous X. The prompt learning module takes them in and outputs the polarity $\hat{p}$, thus we get the triplet (*sentence, aspect, polarity*) which is served as the result of AESC.

3 Experiments

3.1 Setup

Datasets We conduct experiments on three widely used benchmark datasets derived from SemEval 2014 (Pontiki et al., 2014a) and SemEval 2015 (Pontiki et al., 2014b). For each benchmark, the golden boundaries of aspect terms are labeled and the aspect terms are annotated with positive, negative, or neutral polarities. So AE, SC and AESC subtasks are all available. LAPTOP2014 (Lap14) contains the reviews of the products from the laptop domain. RESTAURANT2014 (Res14) and RESTAURANT2015 (Res15) give some comments on foods and dining halls. The training/test splits are fixed for three datasets and more details about them are shown in Table 1.

Metrics For all experiments, we adopt F1 score as evaluation metric following the previous researches. For AE and SC, a predicted aspect term or polarity is correct only if it matches the golden data. And for AESC, we regard it as a right prediction only if an aspect term and its corresponding polarity are both recognized accurately at the same time.

PrLM and Settings For the fair comparison, our selected PrLMs are consistent with the previous strong baselines. We apply the publicly available BERT-Base-Uncased and BERT-Large-Uncased models[2] with the vanilla parameters and sizes for our **MRCOOL** framework. We adopt AdamW optimizer with the learning rate of 2e-5 and warmup over the first 15% steps to train for 3 epochs. We use 30 epochs to train our framework.

[2]https://github.com/google-research/bert

	Lap14			Res14			Res15		
	AE	SC	AESC	AE	SC	AESC	AE	SC	AESC
IMN-BERT	77.35	75.56	61.73	84.06	75.67	70.72	69.90	70.10	60.22
SPAN-BERT	82.34	62.50	61.25	86.71	71.75	73.68	74.63	50.28	62.29
RACL-BERT	81.79	73.91	63.40	86.38	81.61	75.42	73.99	**74.91**	66.05
DUAL-MRC	82.51	75.97	65.94	86.60	**82.04**	75.95	75.08	73.59	65.08
BART-ABSA	82.52	**76.76**	67.37	87.07	75.56	73.56	75.48	73.91	**66.61**
MRCOOL	**86.50**	75.78	**69.47**	**88.31**	79.41	**77.12**	**77.35**	70.76	65.62

Table 2: Main results on three benchmark datasets for AE, SC and AESC. All results are measured by F_1. The state-of-the-art results are in bold.

The batch size is 16 and the α, β, γ for MRC are all set to 1/3. For each experiment, we train our framework multiple times with different random seeds. The average of the best three results is regarded as a final result. We conduct all experiments on one Nvidia Titan RTX GPU.

3.2 Baselines

We compare our proposed **MRCOOL** framework with the following methods on AE, SC and AESC subtasks:

RACL-BERT Chen and Qian (2020) propose a RACL framework which stacks multiple layers. They also propose a relation propagation approach to obtain interactive signals among different subtasks. With the BERT model, their framework achieves good performance on AE, SC and AESC.

IMN-BERT He et al. (2019) put forward an end-to-end multi-task learning model for AE, SC and AESC. They apply a mechanism of information transmission to enhance their model.

SPAN-BERT Hu et al. (2019) propose a pipeline model for AESC. They apply BERT as their backbone network and a multi-target extractor is used to detect the boundaries of the sentiment aspects. Then a polarity classifier recognizes the polarity for each aspect.

DUAL-MRC Mao et al. (2021) present a unified framework for AESC. Two BERT models are contained by their framework and two different MRC tasks are carried out by them separately. The left BERT recognizes the boundaries of aspect terms and the right BERT extracts their polarities.

BART-ABSA Yan et al. (2021) propose the current state-of-art model which can solve the AE, SC and AESC. It redefine each subtask as a sequence mixed by pointer indexes and sentiment class indexes. Then they convert all ABSA subtasks into a unified generative formulation. Finally, they use pre-trained sequence-to-sequence model BART to handle the subtasks in an end-to-end framework.

3.3 Main Results

Table 2 compares our results with other state-of-the-art approaches on three benchmark datasets. The results of AE are all obtained by MRC on BERT-large model. As for SC, we obtain the best results by taking the BERT-base model as the backbone of prompt learning for LAPTOP2014, RESTAURANT2015 datasets and the BERT-large model for RESTAURANT2014. More about the selection of different scale PrLM will be discussed in section 4.1. It is worth noting that our better results do not derive from the better PrLM, owing to DUAL-MRC (Mao et al., 2021) having already adopted the BERT-large model.

For the AE subtask, our MRC method has made good progress. We exceed the prior SOTA model by +3.98%, +1.24% and +1.87% on Lap14, Res14 and Res15 respectively. For the SC subtask, with our prompt learning method, we also achieve performance comparable to the best results before. For AESC subtask, our framework attains state-of-the-art performance by considerable margins over previous methods on Lap14 and Res14. Even if we do not obtain the SOTA on Res15, we also obtain a better result than most previous models.

The above results prove that our framework is very effective. We directly use the knowledge of the PrLM to avoid complex neural layers and feature engineering to attain SOTA or the results close to SOTA on multiple subtasks. The fine-tuning of PrLM is also very time-saving. With only 30 training epochs, our framework can get such a good

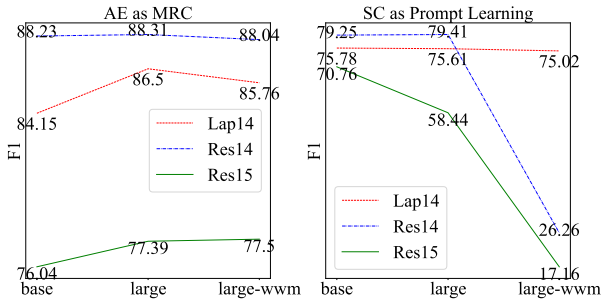

Figure 4: The F_1 change curve with the increment of the scale of PrLM on AE and SC.

	Lap14	Res14	Res15
w/o Query Encoder	85.24	87.35	77.05
Query Encoder	**86.50**	**88.31**	**77.35**

Table 3: Ablation study on Query Encoder for MRC. The BERT-large model is adopted. All results are F_1 for the AE subtask.

performance. The experimental results indicate the effectiveness and simplicity of our **MRCOOL** framework.

4 Ablation Study

4.1 Effect of different PrLM scale

Plenty of earlier studies have proved that the scale of the PrLM has a great influence on the performance of downstream tasks. Therefore, we adopt BERT-base, BERT-large and BERT-large-wwm models that derived from one series but of different scales to test our framework. The results are given in Figure 4.

For MRC, it can be speculated from the curve that the PrLM of different scales is relatively stable on the three datasets. Among three PrLM, the performance of BERT-large is usually better than BERT-base and BERT-large-wwm. For prompt learning, some not robust phenomena happen. BERT-base still performs well on the three datasets. But with the BERT-large, a significant loss of accuracy on the Res15 occurs. And BERT-large-wwm produces disastrous results both on Res14 and Res15. We check the training logs of these poorly experiments and find that the training loss usually reaches zero after 10 epochs but the testing loss still maintains a high value. This means that heavy overfitting could be triggered.

For these three benchmark datasets, BERT-large seems to be usually the best choice and it is definitely not true that a larger scale of PrLM leads to a better result.

	Lap14	Res14	Res15
w/o Prompt Encoder	**75.78**	**79.25**	**70.76**
Prompt Encoder	73.92	73.04	66.55

Table 4: Ablation study on Prompt Encoder for prompt learning. The BERT-large model is adopted. All results are F_1 for the SC subtask.

	Lap14	Res14	Res15
Selecting Words	74.40	78.95	69.45
Single-Layer MLP	74.69	79.21	69.29
Double-Layer MLP	**75.78**	**79.25**	**70.76**

Table 5: Ablation study on different verbalizer. Selecting words means the current dosage of selecting the probability distribution of a few of words. The BERT-large model is applied and all results are F_1 for the AE subtask.

4.2 Effect of Query/Prompt Encoder Module

We propose a query encoder module in this paper to find the potential optimal query for MRC. We also directly think of whether we can learn a potential optimal prompt template by the prompt encoder module whose structure is the same as the query encoder module. For example, we desire to find the optimal embedding of *I felt the* a_i *was* `[MASK]` in the template T_0. So we conduct ablation studies on the query encoder of MRC module and prompt encoder of the prompt learning module, respectively. The results are shown in the Table 3 and Table 4.

It can be indicated that the introduced query encoder has facilitated the performance of MRC, which is in line with the previous research that the query plays an important role in MRC and also supports the effectiveness of the query encoder module. But the prompt encoder harms to the prompt learning. The explanation we give is that we use four different templates for prompt learning and it is too difficult for a simple query encoder module to learn perfect embeddings for them all. We attempted to apply only one template and combine the query encoder module. The result shows that the query encoder module does boost the performance but such a manner is far not as good as the combination of four templates. Thus, we abandon the query encoder module in our framework.

4.3 Effect of MLP Verbalizer

In this paper, one of our major improvements to prompt learning is to desert the original verbalizer that selecting the probability distribution of minority specific words from the output word-embedding of `[MASK]` and we leverage an MLP with a RELU activation function to overcome the sparsity and loss of information. We respectively use the original method, a single-layer MLP and a double-layer MLP with RELU activation function for ablation study. Table 5 displays the result.

Experiments express that MLP does have a relatively large improvement in the effect of prompt learning and the double-layer MLP with RELU has a stronger ability to learn the probability distribution of three polarities. As a result, our proposed simple MLP verbalizer shows quite effective.

5 Related Work

Open-domain sentiment analysis or ABSA requires to extracting the aspect terms with their corresponding sentiment polarities in the open domain, which is an active research topic in recent years. Early studies treat them as two separate tasks and use some traditional algorithms such as Conditional Random Fields (CRF) to complete the task. Wang et al. (2016) apply recursive neural CRF to perform ABSA. Shu et al. (2017) use a lifelong learning CRF to extract the prior knowledge of past domains. With the rise of deep learning technology, more and more models based on neural networks have begun to emerge. (Poria et al., 2016; Xu et al., 2018; Shu et al., 2019; Wu et al., 2021) take convolutional neural network (CNN) to handle ABSA tasks. In addition, some researchers apply RNN and also make some progress (Wadawadagi and Pagi, 2018; Han et al., 2018; Luo et al., 2019; Zeng et al., 2019). In recent years, more studies have proposed diverse attention mechanisms to extract more knowledge from the text and boost the performance of the model. Wang et al. (2017) put forward a novel layer containing two attentions to extract sentiment aspects, opinions and polarities. Li et al. (2018) propose a history attention to exploit the opinion summary and the aspect detection history. Rida-E-Fatima et al. (2019) propose a deep learning-based multilayer dual-attention model to extract the mediate relationships between the aspects and opinions.

Recently, offering promising performance, PrLM has become an important and rapid development area in the field of natural language processing. However, minority studies directly utilize the knowledge of PrLM. Even if the current state-of-art model (Yan et al., 2021), it merely regards BART (Lewis et al., 2020) as a powerful seq-to-seq model without using its erudition. For extracting the knowledge from PrLM, MRC is an excellent solution. Li et al. (2020) convert named entity recognition into an MRC task and achieve the state-of-the-art. Mao et al. (2021) design a DUAL-MRC framework for ABSA and get the promising result. Gan et al. (2021) employ MRC to handle dependency parsing. Yu et al. (2021) and Chen et al. (2021a) respectively propose a self question-answering model and a bidirectional MRC model for ABSA, but they can not solve all three AE, SC and AESC tasks.

For prompt learning, many researchers regard it as a new learning paradigm along with the swift growth of PrLM and argue that it can effectively reel off the enlightenment of PrLM. (Chen et al., 2021b; Seoh et al., 2021) have applied prompt learning to named entity recognition and sentiment analysis respectively. Han et al. (2021) implement a sentence classification model by prompt learning and logic rules. On the selection and generation of prompt, Shin et al. (2020) propose an automatic prompt generation method and Liu et al. (2021) put forward a p-tuning idea to improve the effectiveness of prompt.

However, the above works do not pay attention to the query generation of MRC and the sparsity of the verbalizer of prompt learning. In this paper, we propose a **MRCOOL** framework for ABSA which designs a query encoder to improve the capability of MRC and a simple MLP verbalizer is used to reduce the sparsity of prompt learning.

6 Conclusion

In this paper, we propose a **MRCOOL** framework to handle AE, SC and AESC subtasks of ABSA in one shot through the process of MRC with prompt learning. In detail, we first model aspect extraction as an MRC task and then let the MRC module help aspect-level sentiment classification implemented in a prompt learning way so that we present an end-to-end framework to fulfill the complete task requirement of ABSA. The experimental results demonstrate the effectiveness of our proposed framework by providing consistent and general performance improvement over strong baselines. In

detail, our framework attains new state-of-the-art for AE subtask by considerable margins over previous methods on three datasets and reaches state-of-the-art performance for AESC on two datasets.

References

Shaowei Chen, Jie Liu, Yu Wang, Wenzheng Zhang, and Ziming Chi. 2020. Synchronous double-channel recurrent network for aspect-opinion pair extraction. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 6515–6524, Online. Association for Computational Linguistics.

Shaowei Chen, Yu Wang, Jie Liu, and Yuelin Wang. 2021a. Bidirectional machine reading comprehension for aspect sentiment triplet extraction. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 35, pages 12666–12674.

Xiang Chen, Ningyu Zhang, Lei Li, Xin Xie, Shumin Deng, Chuanqi Tan, Fei Huang, Luo Si, and Huajun Chen. 2021b. Lightner: A lightweight generative framework with prompt-guided attention for low-resource ner. *ArXiv preprint*, abs/2109.00720.

Xiang Chen, Ningyu Zhang, Xin Xie, Shumin Deng, Yunzhi Yao, Chuanqi Tan, Fei Huang, Luo Si, and Huajun Chen. 2021. KnowPrompt: Knowledge-aware Prompt-tuning with Synergistic Optimization for Relation Extraction. *arXiv e-prints*, page arXiv:2104.07650.

Zhuang Chen and Tieyun Qian. 2020. Relation-aware collaborative learning for unified aspect-based sentiment analysis. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 3685–3694, Online. Association for Computational Linguistics.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, Minneapolis, Minnesota. Association for Computational Linguistics.

Feifan Fan, Yansong Feng, and Dongyan Zhao. 2018. Multi-grained attention network for aspect-level sentiment classification. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 3433–3442, Brussels, Belgium. Association for Computational Linguistics.

Zhifang Fan, Zhen Wu, Xin-Yu Dai, Shujian Huang, and Jiajun Chen. 2019. Target-oriented opinion words extraction with target-fused neural sequence labeling. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 2509–2518, Minneapolis, Minnesota. Association for Computational Linguistics.

Leilei Gan, Yuxian Meng, Kun Kuang, Xiaofei Sun, Chun Fan, Fei Wu, and Jiwei Li. 2021. Dependency parsing as mrc-based span-span prediction. *ArXiv preprint*, abs/2105.07654.

Michael Glass, Alfio Gliozzo, Rishav Chakravarti, Anthony Ferritto, Lin Pan, GP Bhargav, Dinesh Garg, and Avirup Sil. 2019. Span selection pre-training for question answering. *arXiv preprint arXiv:1909.04120*.

Xavier Glorot, Antoine Bordes, and Yoshua Bengio. 2011. Deep sparse rectifier neural networks. In *Proceedings of the fourteenth international conference on artificial intelligence and statistics*, pages 315–323. JMLR Workshop and Conference Proceedings.

Xu Han, Weilin Zhao, Ning Ding, Zhiyuan Liu, and Maosong Sun. 2021. Ptr: Prompt tuning with rules for text classification. *ArXiv preprint*, abs/2105.11259.

Zhongming Han, Xin Jiang, Mengqi Li, Mengmei Zhang, and Dagao Duan. 2018. An integrated semantic-syntactic sbLSTM model for aspect specific opinion extraction. In *International Conference on Web Information Systems and Applications*, pages 191–199. Springer.

Ruidan He, Wee Sun Lee, Hwee Tou Ng, and Daniel Dahlmeier. 2019. An interactive multi-task learning network for end-to-end aspect-based sentiment analysis. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 504–515, Florence, Italy. Association for Computational Linguistics.

Minghao Hu, Yuxing Peng, Zhen Huang, Dongsheng Li, and Yiwei Lv. 2019. Open-domain targeted sentiment analysis via span-based extraction and classification. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 537–546, Florence, Italy. Association for Computational Linguistics.

Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Veselin Stoyanov, and Luke Zettlemoyer. 2020. BART: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7871–7880, Online. Association for Computational Linguistics.

Xiaoya Li, Jingrong Feng, Yuxian Meng, Qinghong Han, Fei Wu, and Jiwei Li. 2020. A unified MRC framework for named entity recognition. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 5849–5859, Online. Association for Computational Linguistics.

Xin Li, Lidong Bing, Piji Li, and Wai Lam. 2019. A unified model for opinion target extraction and target sentiment prediction. In *The Thirty-Third AAAI Conference on Artificial Intelligence, AAAI 2019, The Thirty-First Innovative Applications of Artificial Intelligence Conference, IAAI 2019, The Ninth AAAI Symposium on Educational Advances in Artificial Intelligence, EAAI 2019, Honolulu, Hawaii, USA, January 27 - February 1, 2019*, pages 6714–6721. AAAI Press.

Xin Li, Lidong Bing, Piji Li, Wai Lam, and Zhimou Yang. 2018. Aspect term extraction with history attention and selective transformation. In *Proceedings of the Twenty-Seventh International Joint Conference on Artificial Intelligence, IJCAI 2018, July 13-19, 2018, Stockholm, Sweden*, pages 4194–4200. ijcai.org.

Xin Li and Wai Lam. 2017. Deep multi-task learning for aspect term extraction with memory interaction. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 2886–2892, Copenhagen, Denmark. Association for Computational Linguistics.

Bing Liu. 2012. Sentiment analysis and opinion mining. *Synthesis lectures on human language technologies*, 5(1):1–167.

Jian Liu, Yubo Chen, Kang Liu, Wei Bi, and Xiaojiang Liu. 2020. Event extraction as machine reading comprehension. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1641–1651, Online. Association for Computational Linguistics.

Xiao Liu, Yanan Zheng, Zhengxiao Du, Ming Ding, Yujie Qian, Zhilin Yang, and Jie Tang. 2021. Gpt understands, too. *ArXiv preprint*, abs/2103.10385.

Yang Liu, Sujian Li, Xiaodong Zhang, and Zhifang Sui. 2016. Implicit discourse relation classification via multi-task neural networks. In *Proceedings of the Thirtieth AAAI Conference on Artificial Intelligence, February 12-17, 2016, Phoenix, Arizona, USA*, pages 2750–2756. AAAI Press.

Huaishao Luo, Tianrui Li, Bing Liu, and Junbo Zhang. 2019. DOER: Dual cross-shared RNN for aspect term-polarity co-extraction. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 591–601, Florence, Italy. Association for Computational Linguistics.

Yue Mao, Yi Shen, Chao Yu, and Longjun Cai. 2021. A joint training dual-mrc framework for aspect based sentiment analysis. *ArXiv preprint*, abs/2101.00816.

Haiyun Peng, Lu Xu, Lidong Bing, Fei Huang, Wei Lu, and Luo Si. 2020. Knowing what, how and why: A near complete solution for aspect-based sentiment analysis. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pages 8600–8607.

Maria Pontiki, Dimitris Galanis, John Pavlopoulos, Harris Papageorgiou, Ion Androutsopoulos, and Suresh Manandhar. 2014a. SemEval-2014 task 4: Aspect based sentiment analysis. In *Proceedings of the 8th International Workshop on Semantic Evaluation (SemEval 2014)*, pages 27–35, Dublin, Ireland. Association for Computational Linguistics.

Maria Pontiki, Dimitris Galanis, John Pavlopoulos, Harris Papageorgiou, Ion Androutsopoulos, and Suresh Manandhar. 2014b. SemEval-2014 task 4: Aspect based sentiment analysis. In *Proceedings of the 8th International Workshop on Semantic Evaluation (SemEval 2014)*, pages 27–35, Dublin, Ireland. Association for Computational Linguistics.

Soujanya Poria, Erik Cambria, and Alexander Gelbukh. 2016. Aspect extraction for opinion mining with a deep convolutional neural network. *Knowledge-Based Systems*, 108:42–49.

Syeda Rida-E-Fatima, Ali Javed, Ameen Banjar, Aun Irtaza, Hassan Dawood, Hussain Dawood, and Abdullah Alamri. 2019. A multi-layer dual attention deep learning model with refined word embeddings for aspect-based sentiment analysis. *IEEE Access*, 7:114795–114807.

Ronald Seoh, Ian Birle, Mrinal Tak, Haw-Shiuan Chang, Brian Pinette, and Alfred Hough. 2021. Open aspect target sentiment classification with natural language prompts. *ArXiv preprint*, abs/2109.03685.

Taylor Shin, Yasaman Razeghi, Robert L. Logan IV, Eric Wallace, and Sameer Singh. 2020. AutoPrompt: Eliciting Knowledge from Language Models with Automatically Generated Prompts. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 4222–4235, Online. Association for Computational Linguistics.

Lei Shu, Hu Xu, and Bing Liu. 2017. Lifelong learning CRF for supervised aspect extraction. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 148–154, Vancouver, Canada. Association for Computational Linguistics.

Lei Shu, Hu Xu, and Bing Liu. 2019. Controlled cnn-based sequence labeling for aspect extraction. *ArXiv preprint*, abs/1905.06407.

Chuandong Su, Fumiyo Fukumoto, Xiaoxi Huang, Jiyi Li, Rongbo Wang, and Zhiqun Chen. 2020. DeepMet: A reading comprehension paradigm for token-level metaphor detection. In *Proceedings of the Second Workshop on Figurative Language Processing*, pages 30–39, Online. Association for Computational Linguistics.

Yi Sun, Yu Zheng, Chao Hao, and Hangping Qiu. 2021. NSP-BERT: A Prompt-based Zero-Shot Learner Through an Original Pre-training Task–Next Sentence Prediction. *arXiv e-prints*, page arXiv:2109.03564.

Ramesh S Wadawadagi and Veerappa B Pagi. 2018. A deep recursive neural network model for fine-grained opinion classification. In *International Conference on Recent Trends in Image Processing and Pattern Recognition*, pages 607–621. Springer.

Hai Wan, Yufei Yang, Jianfeng Du, Yanan Liu, Kunxun Qi, and Jeff Z Pan. 2020. Target-aspect-sentiment joint detection for aspect-based sentiment analysis. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pages 9122–9129.

Wenya Wang, Sinno Jialin Pan, Daniel Dahlmeier, and Xiaokui Xiao. 2016. Recursive neural conditional random fields for aspect-based sentiment analysis. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 616–626, Austin, Texas. Association for Computational Linguistics.

Wenya Wang, Sinno Jialin Pan, Daniel Dahlmeier, and Xiaokui Xiao. 2017. Coupled multi-layer attentions for co-extraction of aspect and opinion terms. In *Proceedings of the Thirty-First AAAI Conference on Artificial Intelligence, February 4-9, 2017, San Francisco, California, USA*, pages 3316–3322. AAAI Press.

Chao Wu, Qingyu Xiong, Zhengyi Yang, Min Gao, Qiude Li, Yang Yu, Kaige Wang, and Qiwu Zhu. 2021. Residual attention and other aspects module for aspect-based sentiment analysis. *Neurocomputing*, 435:42–52.

Wei Wu, Fei Wang, Arianna Yuan, Fei Wu, and Jiwei Li. 2019. Coreference resolution as query-based span prediction. *arXiv preprint arXiv:1911.01746*.

Hu Xu, Bing Liu, Lei Shu, and Philip S. Yu. 2018. Double embeddings and CNN-based sequence labeling for aspect extraction. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 592–598, Melbourne, Australia. Association for Computational Linguistics.

Hang Yan, Junqi Dai, Tuo Ji, Xipeng Qiu, and Zheng Zhang. 2021. A unified generative framework for aspect-based sentiment analysis. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 2416–2429, Online. Association for Computational Linguistics.

Guoxin Yu, Jiwei Li, Ling Luo, Yuxian Meng, Xiang Ao, and Qing He. 2021. Self question-answering: Aspect-based sentiment analysis by role flipped machine reading comprehension. In *Findings of the Association for Computational Linguistics: EMNLP 2021*, pages 1331–1342.

Jianfei Yu, Jing Jiang, and Rui Xia. 2018. Global inference for aspect and opinion terms co-extraction based on multi-task neural networks. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 27(1):168–177.

Jiangfeng Zeng, Xiao Ma, and Ke Zhou. 2019. Enhancing attention-based LSTM with position context for aspect-level sentiment classification. *IEEE Access*, 7:20462–20471.

Wenxuan Zhang, Yang Deng, Xin Li, Yifei Yuan, Lidong Bing, and Wai Lam. 2021. Aspect sentiment quad prediction as paraphrase generation. *ArXiv preprint*, abs/2110.00796.

Zhuosheng Zhang, Junjie Yang, and Hai Zhao. 2020. Retrospective reader for machine reading comprehension. *arXiv preprint arXiv:2001.09694*.

Nested Named Entity Recognition as Corpus Aware Holistic Structure Parsing

Yifei Yang[1,2,†], **Zuchao Li**[1,2,†], **Hai Zhao**[1,2,*]
[1]Department of Computer Science and Engineering, Shanghai Jiao Tong University
[2]MoE Key Lab of Artificial Intelligence, AI Institute, Shanghai Jiao Tong University
{yifeiyang, charlee}@sjtu.edu.cn, zhaohai@cs.sjtu.edu.cn

Abstract

As a fundamental natural language processing task and one of core knowledge extraction techniques, named entity recognition (NER) is widely used to extract information from texts for downstream tasks. Nested NER is a branch of NER in which the named entities (NEs) are nested with each other. However, most of the previous studies on nested NER usually apply linear structure to model the nested NEs which are actually accommodated in a hierarchical structure. Thus in order to address this mismatch, this work models the full nested NEs in a sentence as a holistic structure, then we propose a holistic structure parsing algorithm to disclose the entire NEs once for all. Besides, there is no research on applying corpus-level information to NER currently. To make up for the loss of this information, we introduce Point-wise Mutual Information (PMI) and other frequency features from corpus-aware statistics for even better performance by holistic modeling from sentence-level to corpus-level. Experiments show that our model yields promising results on widely-used benchmarks which approach or even achieve state-of-the-art. Further empirical studies show that our proposed corpus-aware features can substantially improve NER domain adaptation, which demonstrates the surprising advantage of our proposed corpus-level holistic structure modeling.[1]

1 Introduction

Named Entity Recognition is to find predefined named entities such as locations, organizations or people in text, which usually serves as an upstream natural language processing (NLP) task (Huang et al., 2015; Lample et al., 2016; Ma and Hovy, 2016) and one of key knowledge extraction techniques in knowledge engineering.

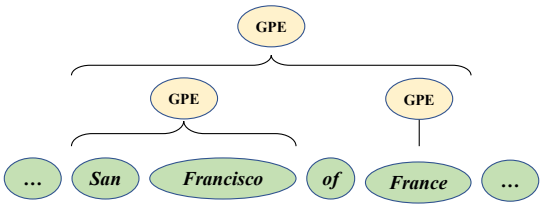

Figure 1: An example of nested named entities. The solid lines or brackets represents the start and end positions of a named entity. 'GPE' is a named entity category, representing geographic or political entities. *France* and *San Francisco* are nested by *the San Francisco of France* which forms a hierarchy structure.

Early works on NER (Liu et al., 2011; Ritter et al., 2011) mostly cope with only flat NEs that have no overlapped relationship with each other. However, as a common language phenomenon, nested NER appears universally in many corpora such as the field of biology or news events. An instance of overlapped NEs is shown in Figure 1. Since solving the Nested NER task will bring more contextual information than the Flat NER task and promotes downstream tasks, it recently has aroused great research interest (Luo and Zhao, 2020; Muis and Lu, 2017; Wang and Lu, 2018). Although researchers have made good progress on flat NER, existing studies nested NER has not achieved a more satisfactory result yet.

With the development of deep learning technology, nested NER has gradually developed from the methods based on handcraft features and traditional machine learning (Alex et al., 2007; Kumar et al., 2008) to the methods based on neural networks. In recent years, myriad studies apply neural network models to achieve state-of-the-art results.

From a perspective of deep learning, NER may be conveniently cast to a multi-class classification task or sequence labeling task (Fisher and Vlachos, 2019; Lample et al., 2016; Ma and Hovy,

* Corresponding author. † These authors made equal contribution to this paper. This work was supported by Key Projects of National Natural Science Foundation of China (U1836222 and 61733011).

[1]https://github.com/yangyifei729/NerAsParsing

Proceedings of the 29th International Conference on Computational Linguistics, pages 2472–2482
October 12–17, 2022.

2016). Nevertheless, such modeling ways which mostly suit for flat NER cannot well handle nested NEs. Instead, there are two categories of models for nested NER, Layered-based Model and Region-based Model. The former recognizes the hierarchical structure by stacking multiple Flat NER layers and the latter type enumerates all text spans to discover the possible named entity mentions.

However, all of the above methods still view all nested NEs as a sequence structure and identify all the possible tags one by one for each NE span, which leads to two obvious drawbacks in modeling capability, (1) These methods require multiple modules to cooperate and each module has to be able to recognize NEs from a complete sentence, which usually results in a too complicated model and time-consuming processing (Luo and Zhao, 2020; Wang et al., 2020). (2) When relying on multiple modules working together, these methods further impose substantial information transmission among different modules, which inevitably causes serious error propagation (Ju et al., 2018).

We address above limitation by modeling the nested NEs in a sentence as a holistic structure and propose a model which builds off work from recently constituency parsing algorithm (Kitaev and Klein, 2018). Although recent works (Lou et al., 2022; Yang and Tu, 2021) model nested NER with the similar idea, they ignore the global information in the corpus. Thus, we extend our idea of holistic structure modeling from sentence-level to corpus-level by introducing corpus-aware features according to the statistics of point-wise mutual information or frequency over dataset and aggregating the information to our model by span attention (Tian et al., 2020). The corpus-aware features hopefully enhance the current model by making up for such inability of the current representations.

Our experiments are conducted on three widely-used benchmark datasets following previous studies (Luo and Zhao, 2020; Wang et al., 2020). Results show that our model approaches or exceeds the current state-of-the-art with the newly-introduced methods. Our contribution can be summarized into three-fold:

- We take a holistic structure to model all nested named entities in a sentence and adopt a chart parsing algorithm to decode every levels of nested named entities once for all, avoiding the error propagation problem while offering a concise and natural solution for so complicated a task.
- We introduce corpus-aware features derived from PMI or frequency statistics to further boost model performance and enable the model a good domain adaptation capability.
- With the simplicity of our modeling idea, our proposed holistic structure parsing model yields performance on par with or surpassing the state-of-the-art on widely-used English benchmark datasets.

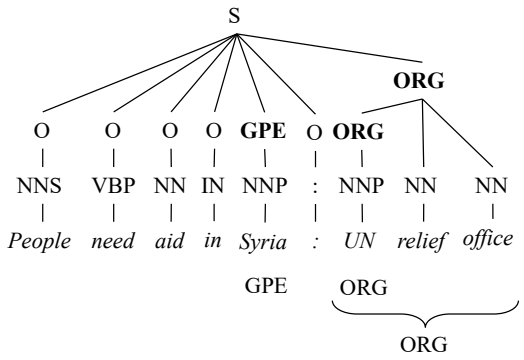

Figure 2: An example of nested NE tree. We use the tag 'O' to label the span that is not recognized as a named entity. The nested named entities form is transformed by removing the tag 'O'.

2 Model

2.1 Holistic Structure Modeling for Nested Named Entities

Given a sentence of n words $\mathcal{X} = \{x_1, x_2, ..., x_n\}$, we denote each named entity as a triplet (i, j, l), where i, j are the beginning and ending positions of a NE with label $l \in \mathcal{L}$, and $\mathcal{L}$ represents the set of NE types. Taking every levels of nested NE as constituent as syntactic parsing, all the labeled NE spans can form a tree structure like syntactic constituent parse tree (shown in Figure 2).

Our model assigns a real value score $s(i, j, l)$ to each labeled span. Then the score of a candidate tree $\mathcal{T}$ can be computed by the scores from all spans inside the tree:

$$s(\mathcal{T}) = \sum_{(i,j,l)\in\mathcal{T}} s(i, j, l). \quad (1)$$

Our model is trained and predicts the structure by selecting the tree $\widehat{\mathcal{T}}$ with the highest score by:

$$\widehat{\mathcal{T}} = \arg\max_{\mathcal{T}} s(\mathcal{T}). \quad (2)$$

Following (Gaddy et al., 2018; Taskar et al., 2005), we apply margin-based training for such a structured prediction problem. Our model is trained to satisfy the constraint:

$$s(\mathcal{T}^*) \geq s(\mathcal{T}) + \Delta(\mathcal{T}^*, \mathcal{T}), \tag{3}$$

where $\mathcal{T}^*$ is the golden tree and $\mathcal{T}$ covers all valid trees. Δ is the Hamming loss on labeled spans. Such a constraint guarantees two points: (1) The score of the golden tree is always greater than or equal to the candidate trees. (2) It enforces the model to make the tree structures with high scores close to the golden structure. And our objective function can be defined by hinge loss:

$$\max\left(0, \max_{\mathcal{T}}\left[s(\mathcal{T}) + \Delta\left(\mathcal{T}, \mathcal{T}^*\right)\right] - s\left(\mathcal{T}^*\right)\right).$$

2.2 Contextualized Encoding

Our model is implemented in an encoder-decoder framework. For the encoder, it encodes each word x_t in the input sentence $\mathcal{X} = \{x_1, x_2, ..., x_n\}$ into vectors h_t. Following (Fried et al., 2017; Kitaev and Klein, 2018), $\mathcal{X}$ will be transformed into word embedding $\mathcal{W} = \{w_1, w_2, ..., w_n\}$ by embedding model. In addition, we embed part-of-speech (POS) tag and word position in sentence as m_i and p_i respectively for x_i. We set these three embeddings to the same dimension and then add them together to generate the input vector $z_i = w_i + m_i + p_i$ for the encoder.

We adopt the Transformer (Vaswani et al., 2017) as our encoder which is composed of 8 modules of the same self-attention structure. Each structure contains a multi-head attention SubLayer_1 and a feed-forward SubLayer_2. After each sublayer, there will be a residual connection and a Layer Normalization. So when given an input x, the output y_2 of each module is:

$$y_2 = \text{LayerNorm}(y_1 + \text{SubLayer}_2(y_1)), \tag{4}$$

where LayerNorm denotes the Layer Normalization and y_1 is the output of the multi-head attention:

$$y_1 = \text{LayerNorm}(x + \text{SubLayer}_1(x)). \tag{5}$$

After 8 stacked modules, the final output h_t for word at position t is generated.

2.3 Decoder: Holistic Structure Parsing

Our decoder performs the needed holistic structure parsing for every levels of nested NEs in a sentence.

After obtaining the context-aware representation h_t for each word, the representation of NE span (i, j) is computed by:

$$r_{i,j} = h_j - h_i, \tag{6}$$

which assumes that the information of a subsequent step is generated by merging a previous state with the information of the span between them, so the difference between two steps can represent the span. In general, $s(i, j, \cdot)$ for span (i, j) over label set $\mathcal{L}$ is calculated by feeding $r_{i,j}$ into a multi-layer perceptrons (MLP), with ReLU activated function, which can be formalized by:

$$o_{i,j} = \text{ReLU}(\text{LayerNorm}(W_1 \cdot r_{i,j} + b_1)), \tag{7}$$

Finally, we have:

$$s(i, j, \cdot) = W_2 \cdot o_{i,j} + b_2, \tag{8}$$

where W_1, W_2 and b_1, b_2 are all trainable parameters.

We adopt CKY parsing algorithm (Younger, 1967) as our decoding algorithm to search for the highest score $s(i, j)$ for span (i, j). In details, to begin with the span $(i, i+1)$ of length 1, we only need to consider its label:

$$s^*(i, i+1) = \max_{l \in \mathcal{L}} s(i, i+1, l). \tag{9}$$

Then, we can extract the highest score $s^*(i, j)$ for longer span (i, j) in a recursive way by searching for the best matching label l and the best boundary k:

$$\begin{aligned} s^*(i, j) \quad &= \max_{l \in \mathcal{L}} s(i, j, l) \\ &+ \max_{i<k<j}[s^*(i, k) + s^*(k, j)]. \end{aligned} \tag{10}$$

In this way, for the current span, the best label and best split point are chosen separately to find the highest score.

According to the above steps, for an entire sentence, we use a bottom-up approach to find the highest score $s^*(0, n)$ which is the sum of the highest scores for its subtrees. The full structure for the entire sentence can be rehabilitated by traversing backpointers. Thus we parse the sentence as a holistic structure and disclose all nested named entities by giving the nested NE tree once for all.

As the same as classical chart parsing, the computational complexity for the CKY algorithm to parse a sentence of length n is $O(n^3)$, which is better than enumerating all possible spans in $O(n^4)$.

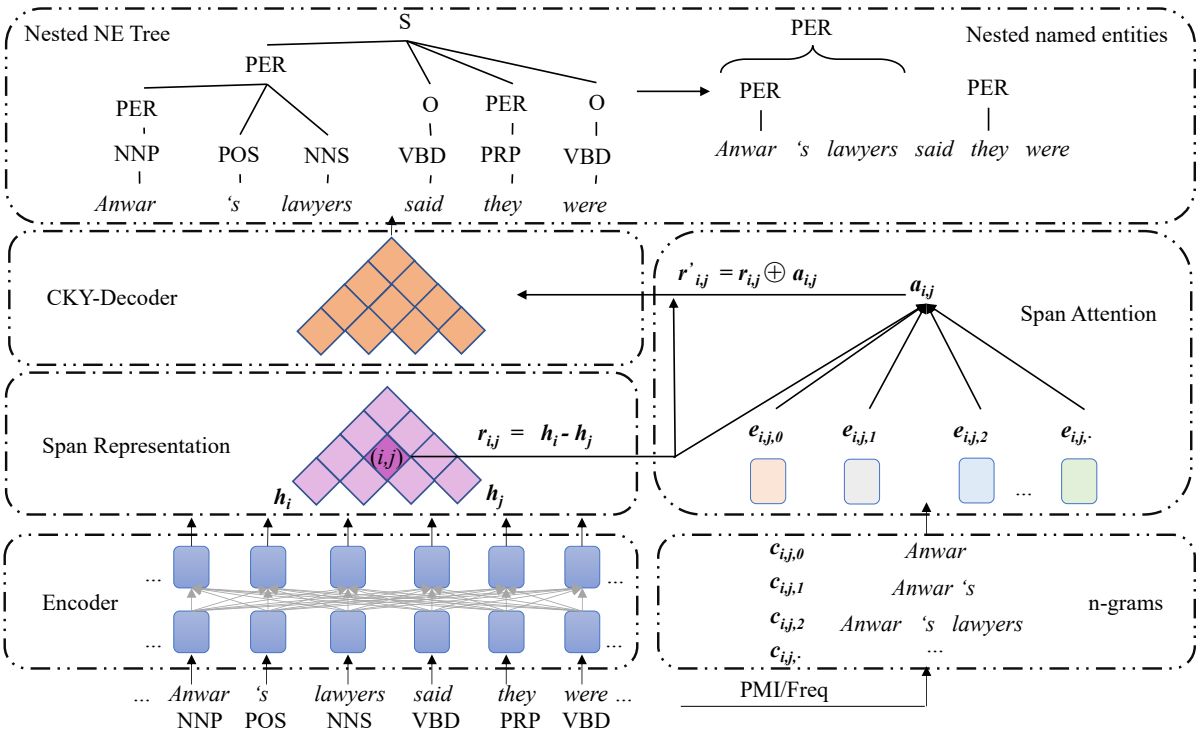

Figure 3: The architecture of our model, where the encoder converts the input sentence into a set of hidden states and spans are represented based on them. Then the span representation and the span attention calculated by n-gram are concatenated together and entered into CKY-decoder.

3 Corpus-aware Features

Though our proposed holistic structure parsing model at sentence-level hopefully captures the entire sentence for predicting all nested NEs, which is supposed to yield more powerful model capability than previous models, it still has the limitation similar with the previous ones which only learn the representation inside a sentence. Meanwhile, NEs quite sparsely distribute in all text data such as most NEs only occur for once in the entire corpus. Besides, there may be a huge domain difference among various corpora, which lets the model recognize new NEs even more difficultly. Thus, we further introduce corpus-aware clues to alleviate the above mentioned difficulties.

In detail, we consider corpus-level statistics including frequency and point-wise mutual information (PMI) over words and corresponding POS tags. Therefore, we consider the following statistics to form enhanced features, PMI on word (Word PMI), word frequency (Word Freq), PMI on POS tags (POS PMI) and on POS tags frequency (POS Freq).

Equipped with the corpus-aware features from a large amount of unlabeled corpus, the model may be enabled to effectively adapt to different domains by alleviating the difficulty caused by the sparsity of NEs distribution.

3.1 Point-wise Mutual Information and Frequency

The PMI between two adjacent tokens x',x'' is calculated as:

$$\mathbf{PMI}(x', x'') = \log \frac{p(x'x'')}{p(x')p(x'')} \quad (11)$$

We calculate the PMI between two adjacent words in turn and split a sentence from the low PMI positions to obtain multiple n-grams.

Similar with the above Word PMI processing, we can also count every n-grams (n<10) to record their frequency in the corpus, then perform sentence segmentation according to a pre-specified threshold t_{freq} following the same processing to obtain the results of Word Freq.

In view of the sparsity of n-gram occurrence, n-gram statistics like the above Word PMI or Word Freq may still encounter the issue of data sparsity and long-tail effect (Kordumova et al., 2016). Thus we turn to the more informative POS which usually has a limited sized POS tag set (Benajiba and Rosso, 2007; Diab, 2007).

To apply the POS information, we first perform POS tagging over every sentences in the corpus

and save the correspondence between the word and the POS tag. Then we segment the spans of POS tags according to PMI or frequency. Finally, we convert these spans back to their corresponding word n-grams.

3.2 Integration of Corpus-aware Features

Following (Tian et al., 2020), we incorporate corpus-aware features into our model through span attention scoring to update the original NE span score in Eq. (6).

Before training our model, we extract all the n-grams from the training set and development set by Word PMI, Word Freq, POS PMI, POS Freq and store them in a Lexicon $\mathcal{N}$. Given a sentence $\mathcal{X}$, we enumerate all the spans (i, j) in it and find the spans which are included by $\mathcal{N}$ to generate a set $\mathcal{C}_{i,j} = \{c_{i,j,1}, c_{i,j,2}, \cdots c_{i,j,v}, \cdots c_{i,j,m_{i,j}}\}$, where $m_{i,j}$ is the total amount of the spans. By span attention, each n-gram $c_{i,j,v}$ in $\mathcal{C}_{i,j}$ will be given an attention score by:

$$a_{i,j,v} = \frac{\exp(\mathbf{r}_{i,j}^{\top} \cdot \mathbf{e}_{i,j,v})}{\sum_{v=1}^{m_{i,j}} \exp(\mathbf{r}_{i,j}^{\top} \cdot \mathbf{e}_{i,j,v})}, \tag{12}$$

where $\mathbf{e}_{i,j,v} \in \mathbb{R}^{d_r}$ refers to the embedding of $c_{i,j,v}$. By the weighted average of n-gram embeddings, the resulted attention of span (i, j) is:

$$\mathbf{a}_{i,j} = \sum_{v=1}^{m_{i,j}} a_{i,j,v} \mathbf{e}_{i,j,v}. \tag{13}$$

As there are more short n-grams in the corpus than long n-grams, it is necessary to encourage these n-grams of various lengths in a balanced way, since there is no difference in the probability of being an entity. Thus, we split n-grams by their lengths into different categories, i.e., $\mathcal{C}_{i,j} = \{\mathcal{C}_{i,j,1}, \mathcal{C}_{i,j,2}, \cdots \mathcal{C}_{i,j,u}, \cdots \mathcal{C}_{i,j,n}\}$, where $u \in [1, n]$ indicates the n-gram length and weight them by each category following Eq. (12-13) to calculate $a_{i,j,v}^{(u)}$ and $\mathbf{a}_{i,j}^{(u)}$. Thus the final attention of the split n-grams is composed of the cascade all category attentions:

$$\mathbf{a}_{i,j} = \bigoplus_{1 \le u \le n} \delta_u \mathbf{a}_{i,j}^{(u)}, \tag{14}$$

where the weight of attentions from different categories is counterpoised by trainable parameter δ_u. After $\mathbf{a}_{i,j}$ is calculated for span (i, j), our model concatenates it with $r_{i,j}$ (generated by Eq. 6): $\mathbf{r}'_{i,j} = \mathbf{r}_{i,j} \oplus \mathbf{a}_{i,j} \in \mathbb{R}^{d_r \cdot (n+1)}$. Then $r'_{i,j}$ will be used to get $s(i, j, \cdot)$ by Eq. (7-8).

4 Experiments

4.1 Setup

We evaluate our approach on three nested named entity recognition benchmark datasets: GENIA, ACE2005 and KBP2017 datasets.

ACE2005 (Walker et al., 2006) contains 25% nested named entities and 7 entity types. We apply the same setup as (Katiyar and Cardie, 2018; Lu and Roth, 2015; Wang and Lu, 2018) by splitting the dataset into training/development/test sets by 8:1:1, respectively.

GENIA dataset (Kim et al., 2003) is based on the GENIAcorpus3.02p[2]. The dataset contains 10% nested mentions and 5 entity types. We follow the same train/dev/test as previous work (Finkel and Manning, 2009; Lu and Roth, 2015) and split first 81% as training set, subsequent 9% as development set, and last 10% as test set.

KBP2017 contains 19% nested entities. We evaluate our model on the Event Nugget Detection Evaluation dataset (LDC2017E55) following (Lin et al., 2019), and previous annotated datasets (LDC2015E29, LDC2015E68, LDC2016E31 and LDC2017E02) are added into training and development sets. We split the datasets into 866/20/167 documents for training, development and test.

We use Precision (P), recall (R) and F-score (F_1) to evaluate the predicted named entities and we consider a named entity to be correctly recognized only if its boundary and category are both recognized correctly.

4.2 Implementation

In our experiments, we use randomly initialized embedding, variants of BERT (Devlin et al., 2019) and XLNet$_{large}$ (Yang et al., 2019) to embed sentences. Following (Tian et al., 2020), we initialize all n-gram embeddings in the span attention module and match their dimensions with the hidden vectors from the encoder. During the training process, we use Adam optimizer with the learning rate 5e-5, 1e-5, and 5e-6. We also fine-tune dropout to attention, POS tag embedding and residual with the rate of 0.2, 0.4, 0.5. We select the model with the highest F_1 on the development set and evaluate it on the test set.

We set t_{pmi} to 0 to determine which words to form n-grams and 0.5 for POS tags since some POS tags often appear next to each other, but their

[2]http://www.geniaproject.org/genia-corpus/posannotation

Model	ACE 2005			GENIA			KBP 2017		
	P	R	F_1	P	R	F_1	P	R	F_1
Hyper-Graph (Katiyar and Cardie, 2018)	70.60	70.40	70.50	77.70	71.80	74.60	–	–	–
Seg-Graph (Wang and Lu, 2018)	76.80	72.30	74.50	–	–	–	–	–	–
ARN (Lin et al., 2019)	76.20	73.60	74.90	75.80	73.90	74.80	77.70	71.80	74.60
Merge-BERT (Fisher and Vlachos, 2019)	82.70	82.10	82.40	–	–	–	–	–	–
DYGIE (Luan et al., 2019)	–	–	82.90	–	–	76.20	–	–	–
Path-BERT (Shibuya and Hovy, 2020)	82.98	82.42	82.70	78.07	76.45	77.25	–	–	–
BERT-MRC (Li et al., 2020)	87.16	86.59	86.88	**85.18**	**81.12**	**83.75**	82.33	77.61	80.97
Seq2seq-BART (Yan et al., 2021)	83.16	86.38	84.74	78.57	79.30	78.93	–	–	–
Seq2set (Tan et al.)	87.48	86.63	87.00	82.31	78.66	80.40	84.91	83.04	83.90
LexicalizedParsing (Lou et al., 2022)	85.97	87.87	86.91	78.39	78.50	78.44	-	-	-
Triaffine-ALBERT (Yuan et al., 2021)	**88.88**	88.24	**88.56**	-	-	-	**89.42**	85.22	87.27
Ours	86.81	**88.70**	87.75	79.76	75.74	77.70	88.25	**87.10**	**87.67**
-Corpus-Feat.	84.65	87.19	85.90	79.74	74.13	76.83	86.62	86.61	86.62

Table 1: Results for nested NER tasks.

DATA	Pre-trained	Word PMI			Word Freq.			POS PMI			POS Freq.			-Corpus-Feat.		
		P	R	F_1	P	R	F_1	P	R	F_1	P	R	F_1	P	R	F_1
ACE2005	Random Init.	70.10	48.68	57.46*	73.35	46.10	56.62	72.13	49.31	**58.58**†	74.33	48.22	56.53	68.35	47.21	55.85
	$BERT_{base}$	82.27	83.15	**82.71***	80.94	83.05	81.98	81.47	82.61	82.04†	81.25	82.31	81.78	78.06	79.34	78.69
	$BERT_{wwm}$	84.05	85.47	**84.75***	83.42	85.29	84.34	83.42	85.29	84.34†	82.75	84.22	83.48	82.38	83.93	83.15
	$XLNet_{large}$	86.81	88.70	**87.75***	84.89	87.16	86.01	85.23	87.19	86.20†	84.95	87.40	86.16	84.65	87.19	85.90
GENIA	Random Init.	64.21	31.46	42.23*	71.05	29.44	41.63	61.85	43.97	51.40	73.03	47.28	**57.40**†	60.05	28.29	38.46
	$BERT_{base}$	78.71	73.44	**75.99***	77.60	72.43	74.92	77.75	73.57	75.60†	77.67	71.60	74.51	76.86	69.38	72.93
	$BERT_{wwm}$	79.07	74.92	**76.94***	78.79	74.26	76.46	78.38	74.76	76.53†	77.05	74.56	75.79	76.11	74.01	75.05
	$XLNet_{large}$	79.84	75.43	77.57	79.82	75.66	77.69*	79.76	75.74	**77.70**†	79.21	75.30	77.21	79.74	74.13	76.83
KBP2017	Random Init.	75.73	49.32	59.73*	76.14	40.76	53.10	72.41	45.56	55.93	77.74	50.54	**61.26**†	70.21	40.32	51.22
	$BERT_{base}$	83.45	80.22	81.80	82.95	81.18	**82.06***	82.42	80.10	81.24†	82.46	78.76	80.57	76.34	74.66	75.49
	$BERT_{wwm}$	85.04	83.31	84.17	85.80	84.00	**84.89***	85.56	82.83	84.17†	82.91	79.35	81.09	80.45	79.48	79.96
	$XLNet_{large}$	88.15	86.48	87.30*	87.96	86.41	87.18	88.25	87.10	**87.67**†	87.68	86.32	87.00	86.62	86.61	86.62

Table 2: Ablation study on the multiple corpus-aware features inducing approaches. * and † indicate the higher F_1 in Word PMI, Word Freq and POS PMI, POS Freq. Bold numbers indicate the highest F_1 in an experiment.

corresponding words are meaningless. Such as the POS tag pair (, VBD) has a high frequency of co-occurrence but the corresponding span (, had) is pointless, thus we set a higher threshold to filter this situation. Similarly, we set t_{freq} to 2 for words and 5 for POS tags.

4.3 Main Results

Table 1 compares our results with other state-of-the-art approaches on three nested NER benchmark datasets. By adding corpus-aware feature, our model has a significant improvement on KBP2017 which exceeds the state-of-the-art model (Yuan et al., 2021) by 1.88 R and 0.4 F_1. We also receive a comparable advancement on ACE2005 and outperform Yuan et al. (2021) by 0.46 R.

On the GENIA , our model performs relatively unsatisfactorily. But it is worth mentioning that the GENIA has less nested structure and is more close to flat NER, which indeed suits seq2seq (Tan et al.) modeling. And the rare biological nouns in the GENIA naturally fit for MRC method (Li et al., 2020) that can better utilize the prior knowledge of pre-trained language model. Apart from these two strong baselines, our method still outperforms better than most models on the GENIA.

After removing the corpus-aware features of each model while the other training parameters remaining unchanged, the results show that the F_1 of all models receive a drop greatly, which proves that our corpus-aware features indeed enhance the effect. Although without them, our method also surpasses the previous best results on the KBP2017 by a large margin and surmounts most preceding approaches on ACE2005 and GENIA, which confirms modeling nested NER as holistic structure is essentially feasible.

5 Ablation Study

5.1 Effect of different corpus-aware features

For exploring the effects of the four corpus-aware features with significance test, we carry out twelve sets of experiments. For each set of them, we combine different features and pre-trained language

models to conduct comparative analysis.

The results are shown in Table 2. For each dataset, we use a randomly initialized word embedding and three pre-training language models to experiment separately and each pre-training language model has been trained with four corpus-aware features. During the training process, we allow the model to fine-tune the pre-trained language model in order to get better result. In order to better verify the effect of them, we also conduct a set of controlled experiment ablating corpus-aware features as a comparison.

In the comparison between PMI and Freq, the PMI-based features devote 8 highest F_1 in all 12 experiments while the Freq-based features occupy the remaining 4 highest results. Namely, PMI-based features show more helpful than Freq-based features for both Word and POS types, which implies PMI is indeed a better statistical measure than the common Freq for building informative corpus-aware features. The result also shows that text-level co-occurrence knowledge has more information than vanilla frequency.

5.2 Role of POS Tags

The comparison of Word and POS is shown in Figure 3. The Word-based features attain 7 highest F_1 while the POS-based features gain the rest 5 highest results. This phenomenon displays that Word-based features are slightly better than POS-based features. This conclusion is also in line with common perception that words themselves provide more knowledge than their POS tags.

The performance obtained by removing corpus-aware features is always the calamity which likewise supports the significant improvement of our proposed corpus-aware features. From overall view, the Word PMI has the best lifting effect for our model, followed by POS PMI. Word Freq and POS Freq are slightly less effective while both of them share the similar effect. Meanwhile, the proposed corpus-aware features can be easily imposed into any models that require span extraction.

5.3 Domain Adaptation Effects

In order to verify that the corpus-aware features can enhance the domain adaptability of our model, we conduct cross-domain experiments between the three datasets. We find the best trained model on one dataset and evaluate the effect of this model on other datasets as our baseline (shown at the top in Table 4). Subsequently, we retrain the best trained model without changing any hyper-parameter setting and add external unlabeled data separately. In details, the model extracts n-gram from the external unlabeled data by leveraging the corpus-aware features and blends them into the lexicon N gleaned from the labeled training dataset. The changed lexicon N will be used to generate span attention and the domain adaptation ability of our model will be affected subsequently.

Method	P	R	F_1
Ours	88.25	87.10	87.67
-POS embedding	87.64	86.21	86.92
-POS PMI	86.62	86.61	86.62
-Both	86.31	85.89	86.10

Table 3: Ablation study on POS tags.

The lower part of Table 4 shows the results. It can be inferred that when a best trained model is joined a non-NER labeled dataset, the prediction effect of the model on this dataset will improve. The biggest improvement occurs when the KBP-trained and ACE-trained model are added GENIA non-NER labeled data. We deduce that the huge increments are because ACE and KBP both derive from the event domain while GENIA comes from a different biological domain. The model will perform poorly in a domain which is pretty different from the training corpus. Once the knowledge of that domain is compensated through the corpus-aware feature, the performance of the model in that domain will be greatly improved. This result verifies the effectiveness of our corpus-aware features for better domain adaptation.

We unexpectedly discover that the domain adaptation between ACE and KBP datasets has not made effective progress because they share the same domain. The corpus-aware features can not bring additional useful knowledge when the external unlabeled datasets are in the same domain with the training corpora. This inference is in line with our intuition and likewise demonstrates that our proposed corpus-aware features are able to extract information from different domains. We also find a counterintuitive fact that GENIA-trained model with KBP unlabeled data outperforms the GENIA-trained model with ACE unlabeled data on ACE. So we infer that the knowledge in the news domain brought by KBP unlabeled data suits well for this GENIA trained model and promotes its performance on another news domain ACE dataset.

In summary, the corpus-aware features can not

Model	ACE 2005			GENIA			KBP 2017		
	P	R	F_1	P	R	F_1	P	R	F_1
ACE-trained Model	86.81	88.70	87.75	24.28*	10.19*	14.36*	73.89	76.06	74.96
GENIA-trained Model	6.49*	0.10*	0.19*	79.76	75.74	77.70	51.66*	2.18*	4.18*
KBP-trained Model	73.49	71.78	72.62	5.88*	0.88*	1.54*	88.25	87.10	87.67
ACE-trained Model									
+ GENIA Unlabeled Data	86.51	87.96	87.23	27.36*	14.09*	**18.60***	74.40	74.60	74.50
+ KBP Unlabeled Data	85.72	87.50	86.60	22.50*	7.91*	11.71*	74.66	75.40	**75.03***
GENIA-trained Model									
+ ACE Unlabeled Data	50.00*	1.20*	**2.35***	79.38	74.80	77.02	37.64*	0.81*	1.59*
+ KBP Unlabeled Data	51.05*	2.53*	4.82*	78.13	74.01	76.01	52.02*	3.13*	**5.90***
KBP-trained Model									
+ ACE Unlabeled Data	73.87	71.85	**72.85**	16.60*	3.86*	6.26*	87.28	86.86	87.07
+ GENIA Unlabeled Data	74.19	70.91	72.51	17.01*	4.28*	**6.83***	86.46	84.79	85.61

Table 4: Results for domain adaptation experiments. * indicates unlabeled F_1 due to different named entity annotation standards. Bold numbers represent an improvement compared to the baselines.

only enhance the effect of the model but also can effectively improve its domain adaptation ability.

6 Related Work

- **Hypergraph model.** This approach is a common method applied for the nested named entities whose main idea is to apply hypergraph structure to represent nested NEs. Lu and Roth (2015) first proposed a hypergraph-based method by connecting multiple nodes with edges to represent nested NEs. Muis and Lu (2017) further employed multigraph representation and introduced a novel notion of mention separators to detect nested named entity. However, these methods are not intelligent enough due to the need to manually design explicit hypergraph.

- **Stacking layered model.** This is also a widespread approach to handle nested NER. Alex et al. (2007) stacked several neural layers to recognize the lower and higher level NEs separately. Fisher and Vlachos (2019) proposed a neural network that merges NEs or tokens to generate nested structure and labels them. Several recent researches applied multi-layer GCN to accomplish nested NER (Li et al., 2021; Luo and Zhao, 2020). These are practical methods, nonetheless, the existing models with a huge depth are computationally impractical.

- **Region-based model.** This method is another common method for nested NER, which enumerates all the possible subsequences and recognizes them. Sohrab and Miwa (2018) regarded all spans as potential NEs and used neural networks to recognize them. A layered model that enumerates all the potential subsequences while preserving the sequence structure was presented by (Wang et al., 2020). The biggest disadvantage of this method is that the model has computational complexity of at least $O(n^4)$ for the sentence of length n. Thus, it requires a lot of time overhead.

- **Parsing-based model.** Some concurrent work also model the nested NER as a parsing task. Lou et al. (2022) put forward a latent lexicalized parsing model for nested NER. Yang and Tu (2021) proposed a unified layer-by-layer bottom-up model for nested NER and constituency parsing. The formal additionally considers dependency syntax information, but its decoding speed is between $O(n^3)$ and $O(n^4)$, which is much lower than ours and not significantly better than our performance. The latter leverages the cascade decoding and suffers from the error propagation issue which does not exist in our model. Essentially different from us, none of them take into account the information of the corpus.

7 Conclusion

NER, so as to achieve the purpose of labeling all nested NEs once for all. On the basis of the proposed holistic sentence structure modeling, we further explore NER model enhancement from corpus-aware statistics with the hope of alleviating the serious sparsity issue of NEs. For this purpose, we extend our holistic structure modeling from sentence-level to corpus-level by offering multiple

corpus-aware features including Word Freq, Word PMI, POS Freq and POS PMI. The experimental results demonstrate the effectiveness of our proposed model by providing consistent and general performance improvement over strong baselines. In details, our model achieves performance approaching state-of-the-art for two datasets and reaches new state-of-the-art for one dataset. Last but not the least, our corpus-level holistic structure modeling shows surprising merit for effective domain adaptation.

References

Beatrice Alex, Barry Haddow, and Claire Grover. 2007. Recognising nested named entities in biomedical text. In *Biological, translational, and clinical language processing*, pages 65–72, Prague, Czech Republic. Association for Computational Linguistics.

Yassine Benajiba and Paolo Rosso. 2007. Anersys 2.0: Conquering the ner task for the arabic language by combining the maximum entropy with pos-tag information. In *IICAI*, pages 1814–1823.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, Minneapolis, Minnesota. Association for Computational Linguistics.

Mona Diab. 2007. Improved Arabic base phrase chunking with a new enriched POS tag set. In *Proceedings of the 2007 Workshop on Computational Approaches to Semitic Languages: Common Issues and Resources*, pages 89–96, Prague, Czech Republic. Association for Computational Linguistics.

Jenny Rose Finkel and Christopher D. Manning. 2009. Nested named entity recognition. In *Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing*, pages 141–150, Singapore. Association for Computational Linguistics.

Joseph Fisher and Andreas Vlachos. 2019. Merge and label: A novel neural network architecture for nested NER. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 5840–5850, Florence, Italy. Association for Computational Linguistics.

Daniel Fried, Mitchell Stern, and Dan Klein. 2017. Improving neural parsing by disentangling model combination and reranking effects. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 161–166, Vancouver, Canada. Association for Computational Linguistics.

David Gaddy, Mitchell Stern, and Dan Klein. 2018. What's going on in neural constituency parsers? an analysis. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 999–1010, New Orleans, Louisiana. Association for Computational Linguistics.

Zhiheng Huang, Wei Xu, and Kai Yu. 2015. Bidirectional lstm-crf models for sequence tagging. *arXiv preprint arXiv:1508.01991*.

Meizhi Ju, Makoto Miwa, and Sophia Ananiadou. 2018. A neural layered model for nested named entity recognition. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 1446–1459, New Orleans, Louisiana. Association for Computational Linguistics.

Arzoo Katiyar and Claire Cardie. 2018. Nested named entity recognition revisited. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 861–871, New Orleans, Louisiana. Association for Computational Linguistics.

J-D Kim, Tomoko Ohta, Yuka Tateisi, and Jun'ichi Tsujii. 2003. Genia corpus—a semantically annotated corpus for bio-textmining. *Bioinformatics*, 19(suppl_1):i180–i182.

Nikita Kitaev and Dan Klein. 2018. Constituency parsing with a self-attentive encoder. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 2676–2686, Melbourne, Australia. Association for Computational Linguistics.

Svetlana Kordumova, Jan van Gemert, and Cees GM Snoek. 2016. Exploring the long tail of social media tags. In *International Conference on Multimedia Modeling*, pages 51–62. Springer.

Sujan Kumar, Saha Sanjay, Chatterji Sandipan Dandapat, Sudeshna Sarkar, and Pabitra Mitra. 2008. A hybrid approach for named entity recognition in indian languages.

Guillaume Lample, Miguel Ballesteros, Sandeep Subramanian, Kazuya Kawakami, and Chris Dyer. 2016. Neural architectures for named entity recognition. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 260–270, San Diego, California. Association for Computational Linguistics.

Fei Li, ZhiChao Lin, Meishan Zhang, and Donghong Ji. 2021. A span-based model for joint overlapped and discontinuous named entity recognition. In *Proceedings of the 59th Annual Meeting of the Association for*

Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers), pages 4814–4828.

Xiaoya Li, Jingrong Feng, Yuxian Meng, Qinghong Han, Fei Wu, and Jiwei Li. 2020. A unified MRC framework for named entity recognition. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 5849–5859, Online. Association for Computational Linguistics.

Hongyu Lin, Yaojie Lu, Xianpei Han, and Le Sun. 2019. Sequence-to-nuggets: Nested entity mention detection via anchor-region networks. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 5182–5192, Florence, Italy. Association for Computational Linguistics.

Xiaohua Liu, Shaodian Zhang, Furu Wei, and Ming Zhou. 2011. Recognizing named entities in tweets. In *Proceedings of the 49th annual meeting of the association for computational linguistics: human language technologies*, pages 359–367.

Chao Lou, Songlin Yang, and Kewei Tu. 2022. Nested named entity recognition as latent lexicalized constituency parsing. *arXiv preprint arXiv:2203.04665*.

Wei Lu and Dan Roth. 2015. Joint mention extraction and classification with mention hypergraphs. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 857–867, Lisbon, Portugal. Association for Computational Linguistics.

Yi Luan, Dave Wadden, Luheng He, Amy Shah, Mari Ostendorf, and Hannaneh Hajishirzi. 2019. A general framework for information extraction using dynamic span graphs. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 3036–3046, Minneapolis, Minnesota. Association for Computational Linguistics.

Ying Luo and Hai Zhao. 2020. Bipartite flat-graph network for nested named entity recognition. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 6408–6418, Online. Association for Computational Linguistics.

Xuezhe Ma and Eduard Hovy. 2016. End-to-end sequence labeling via bi-directional LSTM-CNNs-CRF. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1064–1074, Berlin, Germany. Association for Computational Linguistics.

Aldrian Obaja Muis and Wei Lu. 2017. Labeling gaps between words: Recognizing overlapping mentions with mention separators. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 2608–2618, Copenhagen, Denmark. Association for Computational Linguistics.

Alan Ritter, Sam Clark, Oren Etzioni, et al. 2011. Named entity recognition in tweets: an experimental study. In *Proceedings of the 2011 conference on empirical methods in natural language processing*, pages 1524–1534.

Takashi Shibuya and Eduard Hovy. 2020. Nested named entity recognition via second-best sequence learning and decoding. *Transactions of the Association for Computational Linguistics*, 8:605–620.

Mohammad Golam Sohrab and Makoto Miwa. 2018. Deep exhaustive model for nested named entity recognition. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 2843–2849, Brussels, Belgium. Association for Computational Linguistics.

Zeqi Tan, Yongliang Shen, Shuai Zhang, Weiming Lu, and Yueting Zhuang. A sequence-to-set network for nested named entity recognition.

Ben Taskar, Vassil Chatalbashev, Daphne Koller, and Carlos Guestrin. 2005. Learning structured prediction models: A large margin approach. In *Proceedings of the 22nd international conference on Machine learning*, pages 896–903.

Yuanhe Tian, Yan Song, Fei Xia, and Tong Zhang. 2020. Improving constituency parsing with span attention. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 1691–1703, Online. Association for Computational Linguistics.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, December 4-9, 2017, Long Beach, CA, USA*, pages 5998–6008.

Christopher Walker, Stephanie Strassel, Julie Medero, and Kazuaki Maeda. 2006. Ace 2005 multilingual training corpus. *Linguistic Data Consortium, Philadelphia*, 57:45.

Bailin Wang and Wei Lu. 2018. Neural segmental hypergraphs for overlapping mention recognition. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 204–214, Brussels, Belgium. Association for Computational Linguistics.

Jue Wang, Lidan Shou, Ke Chen, and Gang Chen. 2020. Pyramid: A layered model for nested named entity recognition. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 5918–5928, Online. Association for Computational Linguistics.

Hang Yan, Tao Gui, Junqi Dai, Qipeng Guo, Zheng Zhang, and Xipeng Qiu. 2021. A unified generative framework for various ner subtasks. *arXiv preprint arXiv:2106.01223*.

Songlin Yang and Kewei Tu. 2021. Bottom-up constituency parsing and nested named entity recognition with pointer networks. *arXiv preprint arXiv:2110.05419*.

Zhilin Yang, Zihang Dai, Yiming Yang, Jaime G. Carbonell, Ruslan Salakhutdinov, and Quoc V. Le. 2019. Xlnet: Generalized autoregressive pretraining for language understanding. In *Advances in Neural Information Processing Systems 32: Annual Conference on Neural Information Processing Systems 2019, NeurIPS 2019, December 8-14, 2019, Vancouver, BC, Canada*, pages 5754–5764.

Daniel H Younger. 1967. Recognition and parsing of context-free languages in time n3. *Information and control*, 10(2):189–208.

Zheng Yuan, Chuanqi Tan, Songfang Huang, and Fei Huang. 2021. Fusing heterogeneous factors with triaffine mechanism for nested named entity recognition. *arXiv preprint arXiv:2110.07480*.

DESED: Dialogue-based Explanation for Sentence-level Event Detection

Yinyi Wei[1*†], **Shuaipeng Liu**[2*‡], **Jianwei Lv**[2],
Xiangyu Xi[2], **Hailei Yan**[2], **Wei Ye**[3‡], **Tong Mo**[1], **Fan Yang**[2], **Guanglu Wan**[2]
[1] Peking University
[2] Meituan Group, Beijing, China
[3] National Engineering Research Center for Software Engineering, Peking University
wyyy@pku.edu.cn, liushuaipeng@meituan.com, wye@pku.edu.cn

Abstract

Many recent sentence-level event detection efforts focus on enriching sentence semantics, e.g., via multi-task or prompt-based learning. Despite the promising performance, these methods commonly depend on label-extensive manual annotations or require domain expertise to design sophisticated templates and rules. This paper proposes a new paradigm, named dialogue-based explanation, to enhance sentence semantics for event detection. By saying dialogue-based explanation of an event, we mean explaining it through a consistent information-intensive dialogue, with the original event description as the start utterance. We propose three simple dialogue generation methods, whose outputs are then fed into a hybrid attention mechanism to characterize the complementary event semantics. Extensive experimental results on two event detection datasets verify the effectiveness of our method and suggest promising research opportunities in the dialogue-based explanation paradigm.

1 Introduction

Event detection (ED) is a crucial task in information extraction, which aims to identify event triggers (words or phrases that indicate events) and classify triggers into predefined event types. For example, we can identify the trigger *weddings* and classify it into *Marry* event type from the text "Giuliani regularly officiated at *weddings* while in office". Sentence-level event detection plays a dominant role in event detection and is significant for various downstream NLP tasks.

However, it is usually challenging to accurately detect events in a single sentence due to the limited information. Therefore, most prior methods on sentence-level event detection make improvements by enhancing sentence semantics, being divided into two categories. The first category mainly involves leveraging other information extraction tasks (e.g., named entity recognition and relation extraction) via multi-task learning (Wadden et al., 2019; Lin et al., 2020; Van Nguyen et al., 2021). However, these efforts highly depend on task-specific annotation, costing a vast amount of human effort. The other popular line of research exploits pretrained language models (PLMs), e.g., via prompt-based learning(Gao et al., 2021; Lee et al., 2021; Li et al., 2022; Hsu et al., 2022). MRC-based methods, which treat a task as a Machine Reading Comprehension task (Liu et al., 2020; Li et al., 2020; Du and Cardie, 2020), can also be regarded as a weaker version of prompt-based learning. One common bottleneck among these methods lies in their reliance on domain expertise and human efforts to devise sophisticated templates and rules.

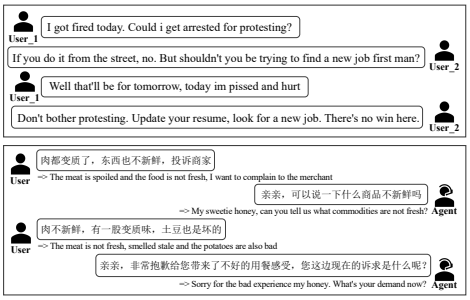

Figure 1: Two examples of dialogues from Reddit and FOSAED, respectively.

To enhance sentence semantics more effectively and efficiently, this work proposes to use generative models to generate contextual information for a sentence in the form of a dialogue, which consists of multiple utterances between different roles on a particular topic.

As two motivation examples, Figure 1 shows two real-world dialogues. In the utterance from User_1 in the first example, models are easily induced by *arrested* and *protesting* thus identifying *fired* as an *attack* event, but the subsequent utterances serve as

*Equal Contribution.
†Work was done when Yinyi Wei interned at Meituan.
‡Corresponding Author.

Proceedings of the 29th International Conference on Computational Linguistics, pages 2483–2493
October 12–17, 2022.

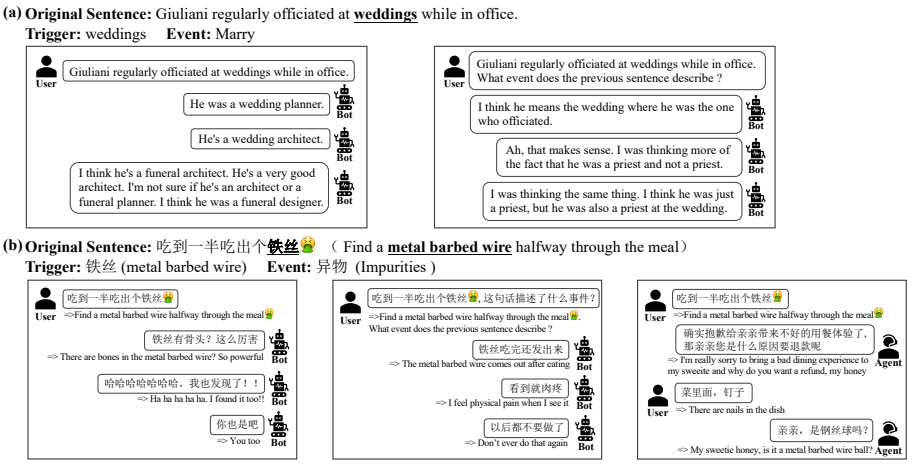

Figure 2: Examples of dialogue generation for a specific sentence with three methods: (1) Direct generation; (2) Generation with a prompt; (3) Further training and generation. Figure (a) shows the dialogue generation using method (1)(2) on ACE05-E$^+$. Figure (b) shows the dialogue generation using method (1)(2)(3) on FOSAED-R.

an explanation that *fired* is an *End-Position* event. In the second example, the dialogue provides clues about a natural association among multiple events, including physical feelings of user, food quality and complaints about the restaurant. Based on these two examples, we conjecture two main merits of dialogues over plain narrative texts in terms of enriching event context. On the one hand, a dialogue is more consistent with the original sentence (see Section 4.5 and 4.6). On the other hand, each utterance is an independent semantic unit requiring no additional segmentation, which is non-trivial for a plain text generated, e.g., by GPT-2. And more importantly, the interaction between these utterances provides room for refining the dialogue-based context. In this paper, we refer the generated dialogue for an event description to dialogue-based explanation and call our method **DESED**: **D**ialogue-base **E**xplanation for **S**entence-level **E**vent **D**etection.

In order to generate semantically rich dialogue-based explanation, we propose three methods based on pretrained dialogue GPTs (Radford et al., 2018, 2019): (1) direct generation on the original sentence; (2) generation with a prompt on the original sentence; (3) generation after further training on dialogue data in the same domain. The three methods are illustrated in Figure 2. Note that prompts we use are quite simple, and identical prompts can be used in our dialogue generation for different events and datasets. In contrast, the aforementioned prompt-based methods require redesigning templates and prompts, demanding expertise across different domains.

To exploit the information of generated dialogues, we then propose three methods: (1) token-level attention with the self-attention mechanism of PLMs; (2) utterance-level attention with an utterance gate; (3) hybrid attention combining the both. We conduct experiments on ACE2005 and another event detection dataset based on real-world data curated by ourselves. Experimentally, our method achieves competitive performance than previous multi-task and prompt-based works.

Our main contributions include:

- We propose dialogue-based explanation, a novel paradigm to enrich sentence semantics for event detection by generating a consistent dialogue on specific events.
- We propose three conceptually simple methods to generate dialogue-based explanation and design hybrid (token-level and utterance-level) attention mechanisms that demonstrate competitive results on two datasets.
- Our experiments reveal that compared with plain narrative contexts, dialogues are more consistent with original sentences and contain richer contextual knowledge for event detection, and appropriate prompts or dialogue data in a specific domain can guide pretrained models to generate better event-centric dialogues.

2 Related Work

2.1 Sentence-level Event Detection

To identify a trigger and classify the trigger into an event type from a sentence, traditional feature-based methods rely heavily on manually designing

features (Ahn, 2006; McClosky et al., 2011). With the development of deep learning, neural networks have been widely used in event detection. The most common usage for neural networks is token classification, which encodes and classifies each token with various neural methods (Chen et al., 2015; Nguyen et al., 2016; Sha et al., 2018). Furthermore, graph based (Liu et al., 2018; Yan et al., 2019), multi-task (Wadden et al., 2019; Lin et al., 2020; Van Nguyen et al., 2021; Lu et al., 2022), MRC-based (Liu et al., 2020; Li et al., 2020; Du and Cardie, 2020), Seq2Seq-based(Sequence-to-Sequence-based) (Lu et al., 2021; Hsu et al., 2022; Paolini et al., 2021) methods have also been introduced to sentence-level event detection.

2.2 Prompt-based Learning

Prompt-based learning aims to stimulate the knowledge of PLMs to serve downstream tasks (Schick and Schütze, 2021). Unidirectional language models (e.g. GPTs (Radford et al., 2018, 2019)), bidirectional language models (e.g. BERT (Kenton and Toutanova, 2019)) and hybrid language models (e.g. BART (Lewis et al., 2020)) can all be used as backbones. By retrieving similar instances in the training set or adding manual definitions of labels (Gao et al., 2021; Lee et al., 2021; Kumar and Talukdar, 2021), or by converting information extraction tasks to slot-filling tasks (Lu et al., 2021; Hsu et al., 2022; Li et al., 2022), prompt-based learning enables PLMs to have priori knowledge of a task, thus contributing to the final performance.

2.3 Generation-based Dialogue System

Generation-based dialogue system can generate a great diversity of responses which are not limited to the existing corpus (Chen et al., 2017). By making use of GPTs (Radford et al., 2018, 2019) and large amount of dialogue data, generation-based models can achieve excellent results on different languages (Zhang et al., 2020; Wang et al., 2020).

3 Methodology

In this section, we present our dialogue-based explanation for sentence-level event deteiction.

3.1 Task Description

In this paper, we formulate sentence-level event detection as a sequence labelling task using *BIO* tagging format. Given a trigger which evokes an event *EventType*. Each token is tagged as *B-EventType*, *I-EventType* or *O*, indicating the token is at the beginning, inside or outside of the trigger tokens.

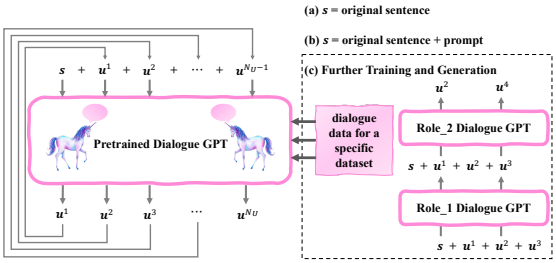

Figure 3: Illustration of dialogue generation methods and an example of dialogue generation with further training on two roles.

Formally, denote $\mathcal{S}$, $\mathcal{Y}$, $\mathcal{M}$ as instance set, label set and bidirectional language model. For a sentence instance $\boldsymbol{s} \in \mathcal{S}$, $\boldsymbol{s} = (s_0, s_1, \ldots, s_{N_s-1})$. In the general setting, representation $\boldsymbol{h} = \mathcal{M}(\boldsymbol{s})$, $\boldsymbol{h} \in \mathbb{R}^{N_s \times D}$, where D is the hidden size of $\mathcal{M}$. When using *BIO* tagging format, the set of all tags is $\mathcal{E}$, the total number of $\mathcal{E}$ is $|\mathcal{E}| = 2 \times |\mathcal{Y}| + 1$. To conduct sequence labelling, a weight matrix $\boldsymbol{W} \in \mathbb{R}^{D \times |\mathcal{E}|}$ and a bias term $\boldsymbol{b} \in \mathbb{R}^{|\mathcal{E}|}$ are introduced to classify each token representation into a tag in $\mathcal{E}$. The classification logits $\boldsymbol{p} = \boldsymbol{hW} + \boldsymbol{b}$, $\boldsymbol{p} \in \mathbb{R}^{N_s \times |\mathcal{E}|}$. The final labelling results $\boldsymbol{e} = \text{argmax}(\boldsymbol{p})$, $\boldsymbol{e} \in \mathbb{R}^{N_s}$,where e_i is the tag of s_i. The optimization objective is set to a cross entropy loss between classification logits $\boldsymbol{p}$ and golden tagging.

3.2 Dialogue Generation

A pretrained dialogue generation model $\mathcal{G}$ is used to generate dialogues. The overview of dialogue generation is shown in Figure 3.

3.2.1 Direct Generation

Given a sentence instance $\boldsymbol{s} = (s_0, s_1, \ldots, s_{N_s-1})$, the goal is to generate N_U utterances. $\boldsymbol{s}$ is firstly fed into $\mathcal{G}$ to obtain an utterance $\boldsymbol{u}^1$, $\boldsymbol{u}^1 = \mathcal{G}(\boldsymbol{s})$. Then $\boldsymbol{s}$ and $\boldsymbol{u}^1$ are concatenated as dialogue history which is fed into $\mathcal{G}$ to get a new response utterance $\boldsymbol{u}^2$, $\boldsymbol{u}^2 = \mathcal{G}(\boldsymbol{s} + \boldsymbol{u}^1)$. Circulating repeatedly, until $\boldsymbol{u}^{N_U}$ is obtained, $\boldsymbol{u}^{N_U} = \mathcal{G}(\boldsymbol{s} + \boldsymbol{u}^1 + \cdots + \boldsymbol{u}^{N_U-1})$.

3.2.2 Generation with a Prompt

To make the generated dialogue more focused on a particular topic, we propose to adding a straightforward prompt at the end of the original sentence, (e.g. *What event does the previous sentence describe?*), which means $\boldsymbol{s} = \boldsymbol{s} + \text{prompt}$. The procedure described in 3.2.1 is then repeated until N_U utterances are obtained.

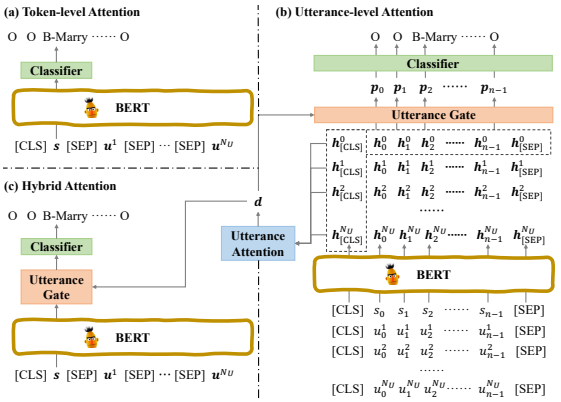

Figure 4: Different attention mechanisms of exploiting dialogue information. Figure (a) illustrates the token-level attention; Figure (b) illustrates the utterance-level attention; Figure (c) illustrates the hybrid attention.

3.2.3 Further Training and Generation

When dialogue data is provided for a dataset, further training can be carried out based on this data.

For the dialogue data with k roles, k different dialogue models are trained with role-specific responses in order to model the characteristics of different roles. When inferring, k different dialogue models are used alternatively to generate utterances from different roles. An example of dialogue generation on two roles is shown on the right of Figure 3.

3.3 Exploitation of Dialogue Information

We exploit generated dialogue information through different attention mechanisms based on sequence labelling. The overview is illustrated in Figure 4.

3.3.1 Token-level Attention

By encoding the concatenation of the original sentence and generated utterances simultaneously with a bidirectional language model $\mathcal{M}$, we can take advantage of the self-attention mechanism and the ability to capture long-range dependencies in $\mathcal{M}$.

Given a sentence instance $\boldsymbol{s}$ and generated utterances $\boldsymbol{u}^1, \ldots \boldsymbol{u}^{N_U}$, we use the separator token of $\mathcal{M}$ (e.g. [SEP] for BERT) to concatenate the original sentence and all utterances. Thus the combined input $\boldsymbol{c} = \boldsymbol{s}$ [SEP] $\boldsymbol{u}^1$ [SEP] ... [SEP] $\boldsymbol{u}^{N_U}$. After obtaining contextual representations by feeding $\boldsymbol{c}$ into $\mathcal{M}$, the token representations corresponding to $\boldsymbol{s}$ are classified into specific tags by a classifier.

3.3.2 Utterance-level Attention

Due to the uncertainty of $\mathcal{G}$, generated utterances may be disorganized and rambling. Directly combining and applying self-attention mechanism may introduce noise to the representation of the original sentence. We therefore propose to use an utterance attention mechanism and an utterance gate to integrate dialogue information into the representation of the original sentence.

Given a sentence instance $\boldsymbol{s}$ and generated utterance $\boldsymbol{u}^1, \ldots, \boldsymbol{u}^{N_U}$, assuming that the original sentence and all utterances are of length n. As shown in Figure 4(b), feeding them into $\mathcal{M}$, we can obtain representation $\boldsymbol{h} = (\boldsymbol{h}^0, \boldsymbol{h}^1, \ldots, \boldsymbol{h}^{N_U})$, where $\boldsymbol{h}^0$ is the representation of $\boldsymbol{s}$; $\boldsymbol{h}^j$, $j \geq 1$, is the representation of $\boldsymbol{u}^j$. For all $\boldsymbol{h}^i$, $i \geq 0$, $\boldsymbol{h}^i \in \mathbb{R}^D$.

An attention mechanism is applied to get a dialogue state $\boldsymbol{d}$ with the representation of [CLS] token $\boldsymbol{h}^i_{[\text{CLS}]}$ and learned attention weight α_i:

$$\boldsymbol{d} = \sum_{i=0}^{N_U} \alpha_i \boldsymbol{h}^i_{[\text{CLS}]},\ \boldsymbol{d} \in \mathbb{R}^D \tag{1}$$

$$\alpha_i = \frac{\exp(s_i)}{\sum_{j=0}^{N_U} \exp(s_j)} \tag{2}$$

$$s_i = \tanh\left(\boldsymbol{h}^0_{[\text{CLS}]} \cdot (\boldsymbol{W}_a \cdot (\boldsymbol{h}^i_{[\text{CLS}]})^T + \boldsymbol{b}_a)\right) \tag{3}$$

where $\boldsymbol{W}_a$ and $\boldsymbol{b}_a$ are the weight matrix and the bias term of a feed-forward neural network, s_i is the relevance score between the original sentence $\boldsymbol{s}$ and an generated utterance $\boldsymbol{u}^i$.

Knowing that $\boldsymbol{d}$ is the semantic abstraction of the whole dialogue, we further propose an utterance gate to fuse $\boldsymbol{d}$ into token representations of $\boldsymbol{s}$.

For the representation of the original sentence $\boldsymbol{h}^0 = (\boldsymbol{h}^0_0, \boldsymbol{h}^0_1, \ldots, \boldsymbol{h}^0_{n-1})$, the fused representation $\boldsymbol{p} = (\boldsymbol{p}_0, \boldsymbol{p}_1, \ldots, \boldsymbol{p}_{n-1})$ is computed as below:

$$\boldsymbol{p}_i = \boldsymbol{h}^0_i \parallel \boldsymbol{f}_i \tag{4}$$

$$\boldsymbol{f}_i = \theta_i \circ \boldsymbol{h}^0_i + (1 - \theta_i) \circ \boldsymbol{d} \tag{5}$$

$$\theta_i = \text{sigmoid}((\boldsymbol{h}^0_i \parallel \boldsymbol{d}) \cdot \boldsymbol{W}_g + b_g) \tag{6}$$

where $\parallel$ is the notation for the concatenation of two vectors, $\circ$ indicates scalar multiplication, $\boldsymbol{W}_g$ and b_g are the weight matrix and the bias term of a feed-forward neural network. θ can be seen as a dynamic threshold to determine how much dialogue information needs to be incorporated into token representations. A classifier is then applied on $\boldsymbol{p}$ to get the final tagging result.

3.3.3 Hybrid Attention

To cover different levels of attention, we propose to use attention mechanisms at both token-level and utterance-level. To get a representation $\boldsymbol{h}^c$ with token-level attention, combined sentence $\boldsymbol{c}$

Form	#Docs	#Sents
Labelled User Reviews	4,226	4,226
Unlabelled Conversations	7,155	309,295

Table 1: Statistics of FOSAED. We show the number of documents and sentences for different forms of data.

Dataset	Split	#Sents	#Events
ACE05-E$^+$	Train	19,216	4,419
	Dev	901	468
	Test	676	424
FOSAED-R	Train	3,380	3,893
	Dev	423	494
	Test	423	512

Table 2: Dataset statistics. We show the number of sentences and events for different splits.

is fed into $\mathcal{M}$: $\boldsymbol{h}^c = \mathcal{M}(\boldsymbol{c})$. Then the utterance attention mechanism and utterance gate are applied to compute the dialogue state $\boldsymbol{d}$ and fuse $\boldsymbol{d}$ into $\boldsymbol{h}^c$. Finally token classification is conducted on the fused representations corresponding to $\boldsymbol{s}$.

4 Experiments

4.1 Experimental Setup

4.1.1 Datasets and Evaluation Metrics

We evaluate on two event detection datasets, ACE2005 (Doddington et al., 2004) and FOSAED.

ACE2005, a collection of documents from a diversity of domains, is the most widely used dataset for event extraction. For data split and preprocessing, we follow Lin et al. (2020), which adds back pronouns and multi-token triggers. We use the English version which covers 8 event types and 33 event subtypes and refer to it as ACE05-E$^+$.

Aiming at evaluating DESED on a specific domain, we curate and propose a new dataset named FOSAED (Food Safety on User Reviews for Event Detection). FOSAED is a real-world Chinese event detection dataset, consisting of sentence-level user reviews (reviews posted by users about orders and restaurants) in the domain of food safety based on a leading e-commerce platform for food service. FOSAED focuses on 4 event types and 21 event subtypes. Each event type and event subtype correspond to a food safety issue (e.g. Abnormalities, Uncomfortable and Undercooked). To support further training, a number of unlabelled conversations are collected, which are in the same domain (i.e., food safety) as the user reviews. These conversations are dialogues between users and agents, and have two sources: text conversations (users communicate online with after-sale agents via text messages) and phone conversations (users communicate with after-sale agents via telephone). Statistics of FOSAED are shown in Table 1. We treat the conversations as the further training dialogue data and conduct event detection on the user reviews. The version is denoted as FOSAED-R.

Statistics of ACE05-E+ and FOSAED-R are shown in Table 2.

For evaluation, we use the same criteria in previous work (Li et al., 2013; Wadden et al., 2019; Lin et al., 2020) and report F1-scores in our experiments. **Trig-I**: A trigger is correctly identified if its offset match any of the gold triggers. **Trig-C**: The span of the trigger is correctly identified and its event type is also correctly classified.

4.1.2 Baselines

We compare DESED to baselines with multi-task learning and prompt-based learning. Specifically, we compare with: (1) **BILSTM+CRF**(Hochreiter and Schmidhuber, 1997; Lafferty et al., 2002), using a bi-directional long short-term memory network and a conditional random field layer; (2) **DMBERT**(Wang et al., 2019), using BERT and dynamic multi-pooling mechanism to assemble features; (3) **BERT**(Kenton and Toutanova, 2019), fine-tuning BERT for token classification; (4) **BERT_QA_TRIGGER**(Du and Cardie, 2020), converting event detection to a MRC task; (5) **OneIE**(Lin et al., 2020), a span-based model with multi-task learning; (6) **FourIE**(Van Nguyen et al., 2021), a span-based model using Graph Convolutional Networks with multi-task learning; (7) **Text2Event**(Lu et al., 2021), using a Seq2Seq model to generate a manually designed structure for each event; (8) **DEGREE**(Hsu et al., 2022), taking advantage of a Seq2Seq model with manually designed templates and prompts; (9) **PILED**(Li et al., 2022), using a prompt-based method to identify a event then adding event-specific demonstration to localize a trigger; (10) **TANL**(Paolini et al., 2021), treating multi-task as translation between augmented natural language and predicting structures with designed annotations; (11) **UIE**(Lu et al., 2022), using a unified text-to-structure generation with multi-task and prompt-based learning.

Category	Methods		ACE05-E$^+$		FOSAED-R	
			Trig-I	Trig-C	Trig-I	Trig-C
Basic	BiLSTM+CRF		72.9	69.3	71.5	70.8
	DMBERT		73.5	69.5	72.8	71.4
	BERT		73.4	70.5	73.6	71.5
MRC-based	BERT_QA_TRIGGER		74.6	71.5	72.9	71.8
Multi-task	OneIE*		75.6	72.8	-	-
	FourIE*		76.7	73.3	-	-
Prompt-based	Text2Event*		-	71.8	-	-
	DEGREE*		76.7	72.7	-	-
	PILED*		-	73.4	-	-
Multi-task and Prompt-based	TANL*		71.5	68.4	-	-
	UIE*		-	73.4	-	-
Dialogue-based Explanation	DESED	Direct Generation	76.2	72.3	**75.8**	74.3
		Generation with a Prompt	**76.9**	**73.5**	**75.8**	74.3
		Further Training	-	-	75.6	**74.4**

Table 3: Experimental results of sentence-level event detection on ACE05-E$^+$ and FOSAED-R (F1-score, %). The best results are in boldface. * indicates results cited from the original paper.

4.1.3 Implementation Details

For all experiments on sequence labelling, we select AdamW for optimization with a learning rate of 3e-5, weight decay of 5e-5, adam ϵ of 1e-8 and max gradient norm of 1.0. The max sequence length is set to the max token length in a batch, and the total max sequence length is set to 256 for ACE05-E$^+$ and 512 for FOSAED-R. We use a linear layer with a dropout rate of 0.3 for the classifier. Each model is trained for 10 epochs and choose the checkpoint with the best validation performance on the development set. For ACE05-E$^+$, we use a batch size of 4 and gradient accumulation step of 4, and BERT-large is applied as backbone. For FOSAED-R, we use a batch size of 4 and gradient accumulation step of 2, and BERT-base-Chinese is applied as backbone. We do all the experiments on NVIDIA Tesla V100. Our codes and datasets are released at https://github.com/Ydongd/DESED.

In order to generate grammatically correct and semantically rich dialogues, we use DialoGPT-large for ACE05-E$^+$ and CDial-GPT$_{\mathrm{LCCC-large}}$ for FOSAED-R as pretrained dialogue generation models. Four prompts are used to generate dialogues. We generate 1-5 utterances from an original sentence and report the best results. For further training on dialogue data, since there is no suitable and sufficient dialogue data in ACE05-E$^+$, we only conduct further training on FOSAED-R.

For unlabelled conversations in FOSAED, we first eliminate mechanical responses according to rules and merge consecutive utterances with the same role, then select the utterances with events (detected by a BERT model) and the next five responses from those utterances as the dialogue dataset which is used to train a user dialogue model and an agent dialogue model. For the user dialogue model, there is 36,395 dialogues in the training set and 4,678 dialogues in the development set; while for the agent dialogue model, there is 36,236 dialogues in the training set and 4,630 dialogues in the development set. When further training, we use a learning rate of 3e-5 and a max gradient norm of 1.0. We train the model for 10 epochs with 5000 warmup steps. The batch size is set to 8 and the gradient accumulation steps is set to 32, which is equivalent to a batch size of 256.

4.2 Main Results

From Table 3, we can see that DESED outperforms basic sequence labeling models (e.g., BiLSTM+CRF and DMBERT) as expected. Compared with the robust BERT token classification method, DESED also achieves improvements of 4.3% Trig-C F1 on ACE05-E$^+$ (73.5% v.s. 70.5%) and 4.1% Trig-C F1 on FOSAED-R (74.4% v.s. 71.5%). The superiority of DESED can also be easily observed by comparing it against a series of multi-task and prompt-based methods. These results prove the overall feasibility and effectiveness of our dialogue-based explanation paradigm.

On ACE05-E$^+$, generation with a prompt yields better results than direct generation. The possible

Generation	Att	ACE05-E$^+$		FOSAED-R	
		Trig-I	Trig-C	Trig-I	Trig-C
Direct	T	74.6	71.6	75.8	74.3
	U	74.9	71.8	75.0	73.4
	H	76.2	72.3	75.7	73.8
Prompt	T	75.2	72.3	75.1	73.7
	U	76.2	73.5	75.8	74.3
	H	76.9	73.3	74.3	72.9
Further	T	-	-	74.3	72.9
	U	-	-	74.9	73.5
	H	-	-	75.6	74.4

Table 4: Different attention mechanisms of DESED on ACE05-E$^+$ and FOSAED-R (F1-score, %). T, U and H denote token-level, utterance-level and hybrid attention mechanism respectively.

reason lies in that sentences in ACE05-E$^+$ are cut from documents, and many are unstructured, making it difficult to generate high-quality dialogues directly. While a clear and clarified prompt bridges the gap between unstructured sentences and generated utterances. On FOSAED-R, different methods produce similar results, as each user review is a complete and independent sentence.

4.3 Attention Mechanisms in DESED

Results of different attention mechanisms are shown in Table 4. Intuitively, more complex attention leads to better performance. However, this is not the case from the experimental results. There are two main reasons: firstly, generated dialogues have many noises and cannot simply be treated as standard contextual texts; secondly, there are differences in the training data for pretrained dialogue generation models. The English and Chinese datasets are constructed from Reddit comments and Weibo conversations, respectively. The latter has shorter utterances and more meaningless content, making the effects of our attention mechanisms vary across languages.

In particular, by applying generation with a prompt on ACE05-E$^+$, though the contents of dialogues are more focused on a topic, they also have some meaningless repetitive sentences which can not be seen as normal contextual texts. Applying self-attention to such contents would mess up token representations. For direct generation, the casualness and uncertainty of the generated contents make the influence of various attention mechanisms more consistent with our expectation.

On FOSAED-R, since user reviews are primarily informal texts, generated utterances may have jumbled characters and modal particles. And the nature of Weibo conversations make generated dialogues having some meaningless sentences. With direct generation, the sentence embeddings from [CLS] tokens may be useless, potentially making utterance attention impair performance. Generation with a prompt would yield consistent and coherent utterances, but the use of more attention mechanisms may confuse the model and make it more difficult to converge. When further training is conducted, generated dialogues are more domain-specific. However, as most of the utterances from the agent are less informative, it does not show a significant improvement in event detection.

Prompt	T	U	H
Prompt_1	72.3	72.6	73.0
Prompt_2	72.0	72.1	71.3
Prompt_3	71.6	73.5	71.8
Prompt_4	71.8	72.2	73.3

Table 5: Trig-C results(%) on ACE05-E$^+$ with different prompts to generate dialogues.

4.4 Effect of Different Prompts

We design four simple prompts to generate dialogues: (1) *What happened?* (2) *What happened in the previous sentence?* (3) *What event does the previous sentence describe?* (4) *Describe the event in the previous sentence.* The results on ACE05-E$^+$ are shown in Table 5.

Prompt_1 for generation and direct generation have the same trend under different attention mechanisms, as Prompt_1 is less topic-specific. However, it works better than direct generation. Prompt_2 and Prompt_3 work similarly, with Prompt_3 being slightly better than Prompt_2. Both of them add a phrase *in the previous sentence* to limit the scope of generated dialogues. Prompt_4 is the declarative form of Prompt_3, which imposes fewer constraints than the interrogative form.

4.5 Exploration of Generated Dialogues

To reveal the quality of generated dialogues and how the dialogue-based explanation impacts event detection, we heuristically design a feature ***p*(consistent)** to quantify the consistency of dialogues, which is defined as the percentage of generated dialogues consistent with the original sentences. This indicator intuitively specifies that if a sentence contains events, the generated dialogue

Generation	Indicator	ACE05-E$^+$	FOSAED-R
Direct	Length	54.6	62.1
	p(event)	11.9	19.5
	p(no-event)	93.2	72.2
	p(consistent)	58.0	30.7
Prompt_3	Length	60.9	79.2
	p(event)	21.2	24.0
	p(no-event)	80.4	71.1
	p(consistent)	54.7	34.0
Further	Length	-	134.6
	p(event)	-	41.1
	p(no-event)	-	26.7
	p(consistent)	-	38.1

Table 6: Heuristic exploration of different dialogue generation methods based on BERT and four indicators. The number of generated utterances is set to five.

Generation	Indicator	Context	Dialogue
Direct	Trig-C	70.6	70.9
	p(event)	22.5	11.9
	p(no-event)	50.4	93.2
	p(consistent)	38.3	58.0
Prompt_3	Trig-C	70.6	71.1
	p(event)	23.5	21.2
	p(no-event)	49.1	80.4
	p(consistent)	38.0	54.7

Table 7: Experiments of using plain narrative contexts or dialogues as additional information on ACE05-E$^+$. Five generated utterances are used for dialogue, and the number of generated tokens is set to the average token length of the five utterances for narrative contexts.

should contain all events in this sentence; if a sentence has no events, the generated dialogue would also has no events. The indicator can be divided into two sub-indicators ***p*(event)** and ***p*(no-event)**. p(event) indicates the number of generated dialogues containing all events in the original sentences as a percentage to the number of sentences with events. And p(no-event) indicates the number of generated dialogues having no events as a percentage to the number of sentences without events. We employ a BERT model to detect events in the generated dialogues consisting of five utterances. The average token length of generated dialogues is also used as a simple feature. It is noteworthy that these four indicators do not reflect true fluency of sentences and information intensity due to the inaccuracy of the BERT model, but they still provide a uniform quantitative metric for a relatively fair comparison.

Intuitively, a generation method producing more consistent dialogues should have higher scores on

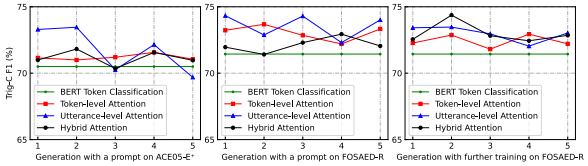

Figure 5: Effect of number of utterance on ACE05-E$^+$ with dialogue generation with a prompt and on FOSAED-R with dialogue generation with a prompt along with further training and generation.

p(consistent), p(event), and p(no-event). As reflected in Table 6, on ACE05-E$^+$, dialogues generated with Prompt_3 have higher p(event) and lower p(no-event) compared with dialogues generated directly. Since generation with a prompt can compensate for deficiencies in the structure and introduce prior knowledge from the prompt, it can generate more event-related dialogues, while more noise would be introduced. Combining the performance on event detection, we conclude that p(event) is a more crucial factor on the final results, however, a smaller p(no-event) may bring more noise, confusing the model and making it harder to converge. On FOSAED-R, generation with a prompt obtains better results on p(consistent) than generated directly, while generation after further training yields the best. Nevertheless, generation after further training significantly increases the average length of dialogues due to nonsensical utterances from an agent, which introduces unnecessary noise, thus giving limited improvement.

4.6 Comparison Between Dialogues and Narrative Contexts

To investigate how dialogue-based explanation differs from narrative contexts, we experiment by comparing the effects between using narrative contexts (generated by GPT-2) and dialogues on ACE05-E$^+$. Since the structure of contexts generated by GPT-2 is confusing and difficult to split for utilization, we only use token-level attention to derive results. For a dialogue, we use five generated utterances. For a narrative context, the number of generated tokens is set to the average token length of the five utterances.

As shown in Table 7, narrative contexts have slightly higher p(event) compared with dialogues, both generation directly and with a prompt, however, p(no-event) and p(consistent) are significantly lower. This observation suggests that narrative contexts are less consistent with original sentences.

Potential Issues	Generated Dialogues
Imbalanced Data in ACE05-E$^+$	"User: The Pakistani supreme court last year ***[aquitted | Justice:Acquit]*** Ayub Masih, who had denied uttering any word against Islam or the Prophet Mohammad. What happened?","Bot: He was sentenced to death.","Bot: He got life."
Multiple Events in ACE05-E$^+$	"User: Police ***[arrested | Justice:Arrest-Jail]*** her in Abilene, Texas, Saturday where she had ***[moved | Movement:Transport]*** with a friend June 2. What event does the previous sentence describe?", "Bot: I think it's a reference to the fact that she was arrested in Abillene, Texas, and then she was found in Abille, Texas.", "Bot: I thought it was a reference that she had been arrested in Texas, but was found at a different location."
Multiple Events in FOSAED-R	"用户:鱼都[没有煮熟|食品不熟]，一股[腥|感官异常]味，[恶心|身体不适]死了", "客服:有异味是吗亲亲,亲亲您方便把餐品照片发给商家吗", "用户:鱼鳞都没有刮干净,鱼腥味太重" ("User: The fish is ***[undercooked | Undercooked]*** and smells ***[fishy | Abnormalities]***, I feel ***[sick and disgusted | Uncomfortable]***","Agent: Is there a peculiar smell, my sweetie honey? Is it convenient for you to send the pictures of the food to the merchant","User: The fish scales are not even scraped clean, and the fishy smell is too strong")

Table 8: Case study for some potential issues in ACE05-E$^+$ and FOSAED-R, we compare DESED to BERT token classification based on original sentences as the baseline. The original sentence is the first utterance from User. The other utterances are generated. The format of the trigger and event is represented as ***[Trigger \| Event_Type]***, where color in red means that DESED can recognize but the baseline cannot, while color in black means that both the baseline and DESED can recognize.

The better performance using the generated dialogues also illustrates the superiority of dialogue-based explanation. Another advantage of dialogues over narrative contexts is that each utterance in a dialogue is an independent semantic unit that requires no additional segmentation, which is essential for various attention mechanisms.

4.7 Effect of Number of Utterance

Figure 5 shows the effect of number of utterance on ACE05-E$^+$ and FOSAED-R. On ACE05-E$^+$, we use Prompt_3. While on FOSAED-R, we use the prompt: 这句话描述了什么事件？ (which has the same English meaning as Prompt_3).

From the results, we can observe that: compared with the token-level attention, the utterance-level attention has a greater fluctuation on the number of utterance, and the hybrid attention is a fusion of them. Due to the randomness of dialogue generation, higher quality generated dialogues are more beneficial than dialogues with more utterances. After further training, the knowledge of the dialogue generation model is limited to a specific domain, thus having a smoother performance.

4.8 Case Study

We conduct a case study to further show the effectiveness of DESED intuitively in Table 8.

There exists data imbalance problem in ACE05-E$^+$ (e.g. Justice:Acquit only accounts for 1.1% of all events in the training set). Additional dialogue information can be utilized as an effective semantic complement for rare events. For a sentence with multiple events, while general methods may have difficulties capturing the association between events, DESED can further discover multiple events through generated dialogues.

5 Conclusion

In this paper, we propose a new paradigm, dialogue-based explanation, to enhance sentence semantics for sentence-level event detection. We propose three conceptually simple methods to generate dialogues for given original sentences, which concentrate on casual dialogues, focused dialogues and domain-specific dialogues respectively. To make effective use of generated dialogues, we design hybrid attention mechanisms at different levels of granularity. Extensive experiments and analyses show that DESED has promising performance on event detection. In the future, we are interested in generating dialogue-based explanation in a more controllable way and extending dialogue-based explanation to other tasks.

Acknowledgements

We would like to thank all reviewers for their insightful comments and valuable suggestions. This research is supported by the National Key Research and Development Program of China (No.2018YFB1403302).

References

David Ahn. 2006. The stages of event extraction. In *Proceedings of the Workshop on Annotating and Reasoning about Time and Events*.

Hongshen Chen, Xiaorui Liu, Dawei Yin, and Jiliang Tang. 2017. A survey on dialogue systems: Recent advances and new frontiers. *Acm Sigkdd Explorations Newsletter*, 19(2):25–35.

Yubo Chen, Liheng Xu, Kang Liu, Daojian Zeng, and Jun Zhao. 2015. Event extraction via dynamic multi-pooling convolutional neural networks. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (ACL-IJCNLP)*.

R. George Doddington, Alexis Mitchell, A. Mark Przybocki, A. Lance Ramshaw, Stephanie Strassel, and M. Ralph Weischedel. 2004. The automatic content extraction (ace) program - tasks, data, and evaluation. *Proceedings of the Fourth International Conference on Language Resources and Evaluation (LREC)*.

Xinya Du and Claire Cardie. 2020. Event extraction by answering (almost) natural questions. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*.

Tianyu Gao, Adam Fisch, and Danqi Chen. 2021. Making pre-trained language models better few-shot learners. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (ACL-IJCNLP)*.

Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long short-term memory. *Neural computation*, 9(8):1735–1780.

I-Hung Hsu, Kuan-Hao Huang, Elizabeth Boschee, Scott Miller, Prem Natarajan, Kai-Wei Chang, and Nanyun Peng. 2022. Degree: A data-efficient generation-based event extraction model. In *Proceedings of the 2022 Annual Conference of the North American Chapter of the Association for Computational Linguistics (NAACL)*.

Jacob Devlin Ming-Wei Chang Kenton and Lee Kristina Toutanova. 2019. Bert: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL-HLT)*.

Sawan Kumar and Partha Talukdar. 2021. Reordering examples helps during priming-based few-shot learning. In *Findings of the Association for Computational Linguistics: ACL-IJCNLP*.

D. John Lafferty, Andrew McCallum, and C. N. Fernando Pereira. 2002. Conditional random fields: Probabilistic models for segmenting and labeling sequence data. In *Proceedings of the 19th International Conference on Machine Learning (ICML)*.

Dong-Ho Lee, Mahak Agarwal, Akshen Kadakia, Jay Pujara, and Xiang Ren. 2021. Good examples make a faster learner: Simple demonstration-based learning for low-resource ner. *arXiv preprint arXiv:2110.08454*.

Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Veselin Stoyanov, and Luke Zettlemoyer. 2020. Bart: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics (ACL)*.

Fayuan Li, Weihua Peng, Yuguang Chen, Quan Wang, Lu Pan, Yajuan Lyu, and Yong Zhu. 2020. Event extraction as multi-turn question answering. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: Findings (EMNLP)*.

Qi Li, Heng Ji, and Liang Huang. 2013. Joint event extraction via structured prediction with global features. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics (ACL)*.

Sha Li, Liyuan Liu, Yiqing Xie, Heng Ji, and Jiawei Han. 2022. Piled: An identify-and-localize framework for few-shot event detection. *arXiv preprint arXiv:2202.07615*.

Ying Lin, Heng Ji, Fei Huang, and Lingfei Wu. 2020. A joint neural model for information extraction with global features. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics (ACL)*.

Jian Liu, Yubo Chen, Kang Liu, Wei Bi, and Xiaojiang Liu. 2020. Event extraction as machine reading comprehension. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*.

Xiao Liu, Zhunchen Luo, and He-Yan Huang. 2018. Jointly multiple events extraction via attention-based graph information aggregation. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing (EMNLP)*.

Yaojie Lu, Hongyu Lin, Jin Xu, Xianpei Han, Jialong Tang, Annan Li, Le Sun, Meng Liao, and Shaoyi Chen. 2021. Text2event: Controllable sequence-to-structure generation for end-to-end event extraction. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (ACL-IJCNLP)*.

Yaojie Lu, Qing Liu, Dai Dai, Xinyan Xiao, Hongyu Lin, Xianpei Han, Le Sun, and Hua Wu. 2022. Unified structure generation for universal information

extraction. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (ACL)*.

David McClosky, Mihai Surdeanu, and Christopher D Manning. 2011. Event extraction as dependency parsing. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies (ACL-HLT)*.

Thien Huu Nguyen, Kyunghyun Cho, and Ralph Grishman. 2016. Joint event extraction via recurrent neural networks. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL-HLT)*.

Giovanni Paolini, Ben Athiwaratkun, Jason Krone, Jie Ma, Alessandro Achille, Rishita Anubhai, Cicero Nogueira dos Santos, Bing Xiang, and Stefano Soatto. 2021. Structured prediction as translation between augmented natural languages. In *9th International Conference on Learning Representations (ICLR)*.

Alec Radford, Karthik Narasimhan, Tim Salimans, and Ilya Sutskever. 2018. Improving language understanding by generative pre-training.

Alec Radford, Jeff Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. 2019. Language models are unsupervised multitask learners.

Timo Schick and Hinrich Schütze. 2021. Exploiting cloze-questions for few-shot text classification and natural language inference. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics (EACL)*.

Lei Sha, Feng Qian, Baobao Chang, and Zhifang Sui. 2018. Jointly extracting event triggers and arguments by dependency-bridge rnn and tensor-based argument interaction. In *Thirty-Second AAAI Conference on Artificial Intelligence*.

Minh Van Nguyen, Viet Lai, and Thien Huu Nguyen. 2021. Cross-task instance representation interactions and label dependencies for joint information extraction with graph convolutional networks. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL-HLT)*.

David Wadden, Ulme Wennberg, Yi Luan, and Hannaneh Hajishirzi. 2019. Entity, relation, and event extraction with contextualized span representations. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*.

Xiaozhi Wang, Xu Han, Zhiyuan Liu, Maosong Sun, and Peng Li. 2019. Adversarial training for weakly supervised event detection. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL-HLT)*.

Yida Wang, Pei Ke, Yinhe Zheng, Kaili Huang, Yong Jiang, Xiaoyan Zhu, and Minlie Huang. 2020. A large-scale chinese short-text conversation dataset. In *CCF International Conference on Natural Language Processing and Chinese Computing (NLPCC)*.

Haoran Yan, Xiaolong Jin, Xiangbin Meng, Jiafeng Guo, and Xueqi Cheng. 2019. Event detection with multi-order graph convolution and aggregated attention. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*.

Yizhe Zhang, Siqi Sun, Michel Galley, Yen-Chun Chen, Chris Brockett, Xiang Gao, Jianfeng Gao, Jingjing Liu, and William B Dolan. 2020. Dialogpt: Large-scale generative pre-training for conversational response generation. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics: System Demonstrations (ACL-SD)*.

Data Augmentation for Few-Shot Knowledge Graph Completion from Hierarchical Perspective

Yuanzhou Yao [1,2], **Zhao Zhang** [1,3]*, **Yongjun Xu** [1] and **Chao Li** [3,1]*
[1] Institute of Computeing Technology, Chinese Academy of Sciences, Beijing, China
[2] University of Chinese Academy of Sciences, Beijing, China
[3] Zhejiang Lab, Hangzhou, China
{yaoyuanzhou21s, zhangzhao2021, xyj, lichao}@ict.ac.cn

Abstract

Few-shot knowledge graph completion (FKGC) has become a new research focus in the field of knowledge graphs in recent years, which aims to predict the missing links for relations that only have a few associative triples. Existing models attempt to solve the problem via learning entity and relation representations. However, the limited training data severely hinders the performance of existing models. To this end, we propose to solve the FKGC problem with the data augmentation technique. Specifically, we perform data augmentation from two perspectives, i.e., inter-task view and intra-task view. The former generates new tasks for FKGC, while the latter enriches the support or query set for an individual task. It is worth noting that the proposed framework can be applied to a number of existing FKGC models. Experimental evaluation on two public datasets indicates our model is capable of achieving substantial improvements over baselines.

1 Introduction

Knowledge graphs (KGs) are structured semantic knowledge bases used to describe concepts and their interrelationships in the physical world in symbolic form. Many KGs in the real world, such as Freebase (Bollacker et al., 2008), YAGO (Suchanek et al., 2007), WordNet (Miller, 1992), Wikidata (Vrandecic and Krötzsch, 2014) and NELL (Mitchell et al., 2018), consist of triple facts in the form of (head entity, relation, tail entity), e.g., (*Paris*, *capitalOf*, *France*) indicates that Paris is the capital of France. KGs have been introduced into various downstream tasks of NLP, such as question answering (Saxena et al., 2020), dialogue systems (He et al., 2017) and information extraction (Hoffmann et al., 2011), etc. The integrity of KG promotes the performance of downstream tasks. However, KGs in the real world are far from complete and comprehensive. Therefore, it is necessary to complete KGs by inferring new triple facts.

To complete KGs, most existing embedding-based KG completion models require adequate triples for each relation as training data, such as TransE (Bordes et al., 2013), RotatE (Sun et al., 2019) and ConvE (Dettmers et al., 2017). However, in reality, the number of triples for each relation conforms to a long-tail distribution (Xiong et al., 2018), i.e., only a small number of relations occur frequently, while most relations only occur a few times in a KG. This phenomenon hinders to learning reliable representations for infrequent relations and further degrades the KG completion performance.

This has motivated an emerging research topic named few-shot knowledge graph completion (FKGC), where one task is to predict the tail entity t in a query $(h, r, ?)$ given only a few entity pairs of the task relation r. GMatching (Xiong et al., 2018) is the first study on the FKGC task, which proposes the basic framework and problem formulation. FSRL (Zhang et al., 2020) and FAAN (Sheng et al., 2020) further improve the attention mechanism of the GMatching framework. MetaR (Chen et al., 2019) and GANA (Niu et al., 2021) adopt the meta-based paradigm in meta-learning as the basic architecture. Although the above methods achieve promising results for the FKGC problem, they still suffer from the limited training data for each relation. To this end, we propose to alleviate the above issue using the data augmentation technique.

Specifically, as shown in Figure 1, we aim to augment the data of each task within its own distribution, and densify the task distribution by providing interpolated tasks. Therefore, we propose to augment data from a hierarchical perspective. The inter-task view generates new tasks for the FKGC model. And the intra-task view provides

* Corresponding author: Zhao Zhang and Chao Li.

Proceedings of the 29th International Conference on Computational Linguistics, pages 2494–2503
October 12–17, 2022.

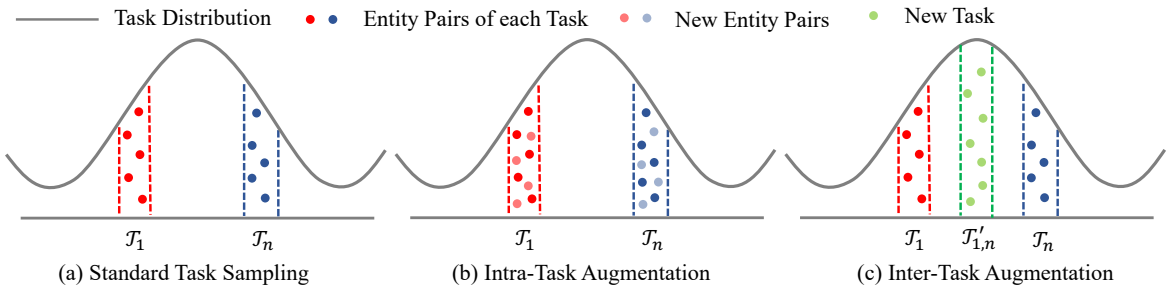

(a) Standard Task Sampling (b) Intra-Task Augmentation (c) Inter-Task Augmentation

Figure 1: Motivations behind data augmentation for FKGC. (a) Two tasks are sampled from the task distribution; (b) Intra-task augmentation methods that augment each task within its own distribution; (c) Inter-task augmentation densifies the task-level distribution by performing cross-task level interpolation or inversing task.

entity pairs for each individual task. This setting is capable of enriching luxuriant data and densifying the data distribution for FKGC models, which is beneficial to achieving better performances. We propose two data augmentation methods for each view to enhance the existing FKGC model. Particularly, the proposed technique is general and can be applied to a number of existing FKGC models. To the best of our knowledge, this is the first work to solve the FKGC task using the data augmentation technique. Finally, experimental results validate the effectiveness of the proposed method.

In a nutshell, we highlight our main contributions as follows,

- To solve the problem of limited training data, we propose to use the data augmentation technique for the FKGC problem. To the best of our knowledge, this is the first work that utilizes data augmentation for FKGC.
- To provide adequate data for the FKGC models, we propose to conduct data augmentation from hierarchical perspectives, i.e., intra-task perspective and inter-task perspective.
- Experimental results on benchmark datasets show the proposed method can be applied to various existing FKGC models and achieve substantial improvements over baselines competitors.

2 Background

In this section, we provide problem formulation and the settings of FKGC.

2.1 Problem Formulation

A Knowledge graph $\mathcal{G}$ is represented as a collection of triples $\{(h, r, t)\} = \mathcal{E} \times \mathcal{R} \times \mathcal{E}$, where $\mathcal{E}$ and $\mathcal{R}$ are the entity set and relation set, respectively. The task of knowledge graph completion falls into two categories: predicting the unknown relation r between the head entity and the tail entity $(h, ?, t)$, and predicting the missing entity t or h based on the head/tail entity and the relation $(h, r, ?)$ or $(?, r, t)$. In this paper, following previous FKGC work, we aim to predict the missing term in a given query $(h, r, ?)$. Unlike traditional knowledge graph completion task that requires abundant triples for the query relation during training, FKGC is only accessible to a few training triples when predicting the tail entity. Specifically, the goal of FKGC is to rank the true tail entities t_{true} higher than other candidate entities $\mathcal{C}_r$. Each relation r corresponds to a candidate entity set, which is constructed based on entity type constraints (Xiong et al., 2018; Toutanova et al., 2015). In the test phase, the corresponding candidate entities are ranked, and the groud truth tail entity is supposed to rank first among the candidates.

2.2 FKGC Settings

FKGC follows the standard few-shot learning settings, and the training data consists of a series of tasks. In FKGC, each task corresponds to a relation in KG $r \in \mathcal{R}_f$, where $\mathcal{R}_f$ is the few-shot relation set, and the rest of the relations in KG are background knowledge graph relations $\mathcal{R}_b$, which consist of high-frequency relations, $\mathcal{R}_f \cup \mathcal{R}_b = \mathcal{R}$ and $\mathcal{R}_f \cap \mathcal{R}_b = \phi$. The triples corresponding to each relation in R_b form the background knowledge graph G', which is mainly used for pre-training the representations of the entity set $\mathcal{E}$ and background knowledge graph relations $\mathcal{R}_b$. The head and tail entity pairs $\{(h_{k,i}, t_{k,i})\}$ of a few-shot relation constitutes a task. Each task $\mathcal{T}_k$ corresponds to one support set S_k and one query set Q_k, and a part of

the task is selected to form the meta-training set $\mathcal{T}_k \in T_{train}$.

Train Prase: The goal of FKGC is to rank all entities in the candidate entity set with S_k as reference, and the ground truth tail entity t_k should be higher than the other false entities t_{false}. We formulize the ranking loss function as $\mathcal{L}_\theta$ and θ denotes the model parameters, and the loss function is set to reflect the rank of the true tail entities in Q_k given S_k. The objective of training the FKGC model is defined as:

$$min_\theta \frac{1}{|T_t|} \sum_{\mathcal{T}_k^t \in T_t} \sum_{(h_{k,i},t_{k,i}) \in Q_k} \frac{L_\theta(t_{k,i}|h_{k,i}, S_k)}{|Q_k|} \tag{1}$$

where $|T_t|$ denotes the number of tasks in T_{train} and the $\mathcal{T}_k^t$ is sampled from the meta-training set T_{train}.

Test Prase: When training is complete and tail entity completion is performed, FKGC models will sample new tasks from the meta-test set $\mathcal{T}_{k'} \in T_{test}$ for prediction. Meta-test set T_{test} also has the support set $S_{k'}$ and query set $Q_{k'}$, which are defined in the same way as in meta-training. Similarly, each task corresponds to a relation in the meta-test relations $r_t \in R_{test}$ that does not appear in the training phase: $\mathcal{R}_{test} \cap \mathcal{R}_{train} = \phi$ and $\mathcal{R}_{train} \cup \mathcal{R}_{test} = \mathcal{R}_f$. These new relations only need to be predicted for tail entity $(t_{k',i}|h_{r',i}, \mathcal{C}_{k'})$ in $Q_{k'}$ with K triples of as $S_{k'}$ a reference.

3 Related Work

3.1 Data Augmentation Strategy

Data augmentation has been widely used to prevent deep neural networks from over-fitting to the training data (Bishop, 1995). Most of the traditional augmentation methods generate new data according to the mixed application transformation of data types or proposed target tasks (Cubuk et al., 2019), which can be independently applied to various data types and tasks, improving the generalization and robustness of deep neural networks. Input mixup (Zhang et al., 2017) linearly interpolates between two input data, and trains the model using mixed data with corresponding soft labels. Following this work, a variety of mixup methods for data augmentation have been proposed. Manifold mixup (Verma et al., 2018) applies the mixup strategy in the hidden feature space, and CutMix (Yun et al., 2019) proposes an image mixup method based on spatial copy and paste. Puzzle Mix (Kim et al., 2020) proposes a mixup method based on saliency and local statistics of the given data. MixSKD (Yang et al., 2022) incorporates Mixup with self-knowledge distillation into a unified framework to regularize the two image views. Most of these methods aim at the field of image processing. In this paper, we specially tailor the mixup strategy for the FKGC task.

3.2 FKGC models

Existing FKGC approaches fall into two categories: metric learning-based methods and meta learner-based methods. We outline the main structures of these two methods and describe them separately in the following

Metric learning-based methods. GMatching is the first research work on FKGC (Xiong et al., 2018), and it utilizes metric learning-based methods as the backbone and divides the model into three subparts: neighbor encoder, entity pairs encoder, and matching processor. *Neighbor encoder* is designed to enhance the representation of each entity with its local connections in the knowledge graph (one-hop neighbors).

Gmatching directly sums all neighbors on average, FSRL (Zhang et al., 2020) uses the attention mechanism (Veličković et al., 2017) to encoding neighbors, and FAAN (Sheng et al., 2020) leverages the relation in task R_f to introduce the adaptive attention network. The embedding of entities $\mathbf{h}, \mathbf{t} \in \mathbb{R}^d$ are then fed into the *entity pairs encoder* $\mathcal{F}_r$:

$$\mathbf{r}_{k,i}^q = \mathcal{F}_r(\mathbf{h}_{k,i}^q, \mathbf{t}_{k,i}^q), \quad \mathbf{r}_{k,.}^s = \mathcal{F}_r(S_k) \tag{2}$$

where $(\mathbf{h}_{k,i}^q, \mathbf{t}_{k,i}^q) \in Q_k$, the query relation $\mathbf{r}_{k,i}^q \in \mathbb{R}^d$ and support set relation $\mathbf{r}_{k,.}^s \in \mathbb{R}^d$ are then compared by *matching processor function*: $Score(\mathbf{r}_{k,i}^q, \mathbf{r}_{k,.}^s) = \mathcal{M}(\mathbf{r}_{k,i}^q, \mathbf{r}_{k,.}^s)$, since $\mathbf{r}_{k,i}^q$ and $\mathbf{r}_{k,.}^s$ represent the same task realtion r_k, their score should be as high as possible.

Meta learner-based methods. MetaR (Chen et al., 2019) is the first model to use the Meta learner-based method as backbone. In contrast to the standard gradient-based meta-learning, MetaR defined two kinds of meta information which are shared between support set and query set. It can be viewed as a bi-level optimization problem.

Formally, the bi-level optimization process can be formulated as:

$$\theta^* \leftarrow argmin_\theta \sum_{\mathcal{T}_k \in T_{train}} [\mathcal{L}_\theta(\mathbf{r}_{k,.}^{meta}, Q_k)]$$

$$s.t.\ \mathbf{r}^{meta}_{k,\cdot} = \mathbf{r}^{s}_{k,\cdot} - \eta \bigtriangledown_{\mathbf{r}^{s}_{k,\cdot}} \mathcal{L}_\theta(\mathbf{r}^{s}_{k,\cdot}, S_k) \quad (3)$$

Where $\mathbf{r}^{s}_{k,\cdot}$ is obtained by Equation 2; $\mathcal{L}_\theta$ and η denote the knowledge graph loss function and inner-loop learning rate. GANA (Niu et al., 2021) shares a similar idea with MetaR, but learns the relation-specific hyper-plane parameters to model complex relations.

4 Methodology

The section describes the details of the data augmentation for FKGC. It falls into two data augmentation methods from the task perspective: intra-task augmentation and inter-task augmentation. inter-task augmentation generates new tasks for the FKGC model, and the intra-task augmentation provides entity pairs for each individual task. We will describe how each of these data augmentation methods is applied to metric learning-based methods and meta learner-based methods.

4.1 Intra-Task Augmentation

Intra-task augmentation only enlarges the pool of triples to be sampled during training within each individual task, not the number of tasks. Since all entity pairs under the same task have the same relation, FKGC uses entity pairs to model few-shot relations $\mathcal{R}_f$. Assume $\mathbf{r}_{k,i}$ is the relation embedding vector modeled by the i-th entity pair $(h_{k,i}, t_{k,i})$ in the k-th task $\mathcal{T}_k$, and since both $\mathbf{r}_{k,i}$ and $\mathbf{r}_{k,j}$ belong to the same task, $\mathbf{r}_{k,j} \approx \mathbf{r}_{k,i}$. We consider the combination of different modeling vectors of the same relation can still represent this task relation: $(\mathbf{r}_{k,i}, \mathbf{r}_{k,j}) \approx \mathbf{r}_{k,j}$. Therefore mixing different entity pairs in the task after the entity pairs encoder can generate a new modeling vector. It is worth noting that what is generated is a new modeling vector belonging to this task relation instead of a new triple. In detail, the mixing strategy follows Manifold Mixup (Verma et al., 2019) where inputs and hidden representations are mixed up. A task contains a query set and a support set: $\mathcal{T}_k = (Q_k, S_k)$, so two types of intra-task augmentation can be derived according to the differences in augmenting settings:

4.1.1 Query Augmentation

Query augmentation enlarges the pool of evaluation data to be sampled during training. Since the structure of the two mainstream FKGC models is different (details in Section 3.2), we will introduce how query augmentation is applied to these two types of models.

Algorithm 1 The Process of Intra-Augmentation

Require: Meta-training set T_{train}, model parameter θ, outer-loop learning rate φ, inner-loop learning rate η, candidate set $\mathcal{C}$.

while not converge **do**
  Randomly sample a task $\mathcal{T}_k$ from T_{train}
  if Metric Learning-Based Methods **then**
    if Query Augmentation **then**
      $\theta = \theta - \varphi \mathbb{E}_{r_k^{new_q}} \bigtriangledown_\theta \mathcal{H}_\theta(\mathbf{r}^{new_q}_{k,i}, \mathbf{r}^{s}_{k,\cdot})$.
    else if Support Augmentation **then**
      $\theta = \theta - \varphi \mathbb{E}_{r_k^{q}} \bigtriangledown_\theta \mathcal{H}_\theta(\mathbf{r}^{q}_{k,i}, \mathbf{r}^{new_s}_{k,\cdot})$.
    end if
  else if Meta Learner-Based Methods **then**
    if Query Augmentation **then**
      $\theta = \theta - \varphi \bigtriangledown_\theta \mathcal{L}_\theta(\mathbf{r}^{meta}_{k,\cdot}, Q^{new}_k)$.
      $s.t.\ \mathbf{r}^{meta}_{k,\cdot} = \mathbf{r}^{s}_{k,\cdot} - \eta \bigtriangledown_{\mathbf{r}^{s}_{k,\cdot}} \mathcal{L}_\theta(\mathbf{r}^{s}_{k,\cdot}, S_k)$.
    else if Support Augmentation **then**
      $\theta = \theta - \varphi \bigtriangledown_\theta \mathcal{L}_\theta(\mathbf{r}^{meta'}_{k,\cdot}, Q_k)$.
      $s.t.\ \mathbf{r}^{meta'}_{k,\cdot} = \frac{1}{|S_k|} \sum_{i=0}^{|S_k|} \mathbf{r}^{meta'}_{k,i}$.
    end if
  end if
end while

Metric Learning-Based Methods try to learn generalizable metrics and the corresponding matching functions $\mathcal{M}(\cdot,\cdot)$ from a set of training tasks. Assume that $\mathbf{r}^{new_q}_{k,i}$ denotes a new modeling vector of query set, we can formulate this change on $\mathcal{M}$ as:

$$\begin{aligned} &\mathcal{M}(\mathbf{r}^{q}_{k,i}, \mathbf{r}^{s}_{k,\cdot}) := \mathcal{M}(\mathbf{r}^{new_q}_{k,i}, \mathbf{r}^{s}_{k,\cdot}) \\ &s.t.\ \ \mathbf{r}^{new_q}_{k,i} = \lambda \mathbf{r}^{q}_{k,i} + (1-\lambda)\mathbf{r}^{q}_{k,j} \end{aligned} \quad (4)$$

Where $\mathbf{r}^{q}_{k,i}$ and $\mathbf{r}^{q}_{k,j}$ are obtained by Eqn.2 and $\lambda \in [0,1]$ is sampled from a Beta distribution $Beta(\alpha, \beta)$. Then we construct a set of negative queries $Q^{neg}_k = \{(h_{k,i}, t^{-}_{k,i})\}$ by randomly corrupting the tail entity, where the false tail entity belongs to the task entity candidate set: $t^{-}_{k,i} \in \mathcal{C}_k$. The loss function is formally defined as:

$$\mathcal{H}_\theta(\mathbf{r}^{new_q}_{k,i}, \mathbf{r}^{s}_{k,\cdot}) = [\gamma + \mathcal{M}(\mathbf{r}^{new_q}_{k,i}, \mathbf{r}^{s}_{k,\cdot}) - \mathcal{M}(\mathbf{r}^{neg}_{k,i}, \mathbf{r}^{s}_{k,\cdot})]_+ \quad (5)$$

where $[x]_+ = max(0, x)$ is standard hinge loss, and γ is a margin separating positive and negative queries.

Meta learner-based methods are a bi-level optimization process; query augmentation for meta learner-based methods can improve the outer-loop optimization. Like metric learning-based methods,

we also construct a new query set, but due to the outer-loop optimization process does not encode the entity pairs of the query set into the relation vector, we directly mix up the original entity pair:

$$Q_k^{new} = \{(\lambda \mathbf{h}_{k,i}^q + (1-\lambda)\mathbf{h}_{k,j}^q, \lambda \mathbf{t}_{k,i}^q + (1-\lambda)\mathbf{t}_{k,j}^q)\} \tag{6}$$

where $(\mathbf{h}_{k,i}^q, \mathbf{t}_{k,i}^q)$, $(\mathbf{h}_{k,j}^q, \mathbf{t}_{k,j}^q) \in \mathcal{Q}_k$ and the Eqn.3 is reformulated as:

$$\theta^* \leftarrow argmin_\theta \sum_{\mathcal{T}_k \in T_{train}} [\mathcal{L}_\theta(\mathbf{r}_{k,\cdot}^{meta}, Q_k^{new})] \tag{7}$$

4.1.2 Support Augmentation

Support augmentation enlarges the pool of triples to be sampled for the support set, not to increase the value of $K = |S_k|$.

Metric Learning-Based Methods. Like the support augmentation, We also randomly sample two relation modeling vectors for mixup to generate a new support set and the Eqn.4 is reformulated as:

$$\begin{aligned} &\mathcal{M}(\mathbf{r}_k^q, \mathbf{r}_{k,\cdot}^s) := \mathcal{M}(\mathbf{r}_k^q, \mathbf{r}_{k,\cdot}^{new_s}) \\ &s.t.\ \mathbf{r}_{k,i}^{new_s} = \lambda \mathbf{r}_{k,i}^s + (1-\lambda)\mathbf{r}_{k,j}^s \end{aligned} \tag{8}$$

where the $\mathbf{r}_{k,i}^{new_s}$ is obtained by aggregating all $\mathbf{r}_{k,i}^{new_s}$ to represent the new support set relation and the loss function of support augmentation for metric learning-based methods is reformulated as: $\mathcal{H}_\theta(\mathbf{r}_k^q, \mathbf{r}_{k,\cdot}^{new_s})$.

Meta Learner-Based Methods. Support augmentation can be applied to support set in the inner-loop to fine-tuning the relation vector $\mathbf{r}_k^s$. This strategy enlarges the pool of fine-tuning data. Since both $\mathbf{r}_{k,i}^s$ and $\mathbf{r}_{k,j}^s$ represent the same task relation, their fine-tuning gradients with respect to the task relation should be consistent, Therefore, we mix the respective fine-tuned gradients of the two relation vector and apply the resulting gradient to $\mathbf{r}_{k,i}^s$.

$$\begin{aligned} \mathbf{r}_{k,i}^{meta'} &= \mathbf{r}_{k,i}^s - [\lambda G(\mathbf{r}_{k,i}^s) + (1-\lambda)G(\mathbf{r}_{k,j})] \\ s.t.\ G(\mathbf{r}_{k,j}) &= \bigtriangledown_{\mathbf{r}_{k,i}^s} \mathcal{L}_\theta[\mathbf{r}_{k,i}^s, (\mathbf{h}_{k,i}^s, \mathbf{t}_{k,i}^s)] \end{aligned} \tag{9}$$

where entity pair $(\mathbf{h}_{k,i}^s, \mathbf{t}_{k,i}^s) \in S_k$ and the $\mathbf{r}_{k,\cdot}^{meta'} = \frac{1}{|S_k|}\sum_{i=0}^{|S_k|} \mathbf{r}_{k,i}^{meta'}$ will be used as the relation vector of all entity pairs in the query set $(\mathbf{h}_{k,i}^q, \mathbf{r}_{k,\cdot}^{meta'}, \mathbf{t}_{k,j}^q)$, thus participating in the outer optimization of the model parameters. Changing $\mathbf{r}_{k,\cdot}^{meta}$ in Eqn.3 to $\mathbf{r}_{k,\cdot}^{meta'}$ is the outer-loop optimization process of support augmentation for meta methods.

4.2 Inter-Task Augmentation

Inter-task augmentation increases the number of tasks by creating new relations r'_k to enlarge the task pool of meta-training set T_{train}. To enlarge the value of $|T_{train}|$, we devise two task augmentation methods: inverse augmentation and interpolation augmentation.

4.2.1 Inverse Augmentation

FKGC models represent few-shot relation $\mathcal{R}_f$ using entity pairs, which consist of head and tail entities. Intuitively, flipping the head and tail entities to represent another relation can enrich the dataset, e.g., the triple *(Elon Musk,SonOf, Errol Musk)* can be flipped as *(Errol Musk, ParentOf, Elon Musk)*, where the entity pair *(Errol Musk, Elon Musk)* can represent a new relation *ParentOf*. When we generalize this augmentation to all tasks in the meta-training set, a new reversed meta-training set can be generated: $T'_{train} = \{\mathcal{T}'_1, \cdots, \mathcal{T}'_N\}$, where N is the number of tasks in T_{train} and $\mathcal{T}'_k = \{(t_{k,i}, , h_{k,i})\}_{|\mathcal{T}_k|}$. Merge the two meta-training sets to get a new larger meta-training set: $T_{train}^{new} = T'_{train} \cup T_{train}$. Therefore, the number of tasks of T'_{train} is twice that of the original train set T_{train}, and finally T'_{train} will replace T_{train} to participate in the training process.

4.2.2 Interpolation Augmentation

We think that the combination of two different relations can generate a new relation, such as $father + mother = grandma$. We adopt a mixup strategy for linear addition rather than direct combination: $\mathcal{T}'_{mix_{i,j}} = \lambda \mathcal{T}_i + (1-\lambda)\mathcal{T}_j$, which adjusts the weight of the two task relations in the new relation by λ. Since λ is obtained by sampling from the beta distribution $Beta(\alpha, \beta)$, the number of tasks in the meta-training set tends to be infinite in theory. When $\lambda = 0.5$, the mixup strategy is equivalent to a direct combination.

Metric Learning-Based Methods. Input the entity pairs of task i: $\mathcal{T}_i$ and task j: $\mathcal{T}_j$ into Eqn.8 respectively to obtain their corresponding relation modeling vectors, and mix up the relation vectors in these two tasks to generate a new task $r'_{mix_{i,j}}$. We can formulate this process as follows:

$$\begin{aligned} \mathbf{r}_{mix_{i,j},k}^{q'} &= \lambda \mathbf{r}_{i,k}^q + (1-\lambda)\mathbf{r}_{j,k}^q \\ \mathbf{r}_{mix_{i,j},k}^{s'} &= \lambda \mathbf{r}_{i,k}^s + (1-\lambda)\mathbf{r}_{j,k}^s \end{aligned} \tag{10}$$

Then we pass $\mathbf{r}_{mix_{i,j},k}^{q'}$ and $\mathbf{r}_{mix_{i,j},k}^{s'}$ through matching processor function to calculating the similarity

Algorithm 2 The Process of Inter-Augmentation

Require: Meta-training set T_{train}, inner-loop learning rate φ, Beta distribution parameters α, β, candidate set $\mathcal{C}$.

if Inverse Augmentation **then**
  $min_\theta \mathbb{E}_{\mathcal{T}'_k} \sum_{(h_{k,i},t_{k,i})\in Q'_k} \frac{L_\theta(t_{k,i}|h_{k,i},S'_k)}{|Q'_k|}$
  $s.t.\ \mathcal{T}'_k \in T^{new}_{train} = T'_{train} \cup T_{train}$
else if Interpolation Augmentation **then**
  while not converge **do**
    Sample two tasks $\mathcal{T}_i$, $\mathcal{T}_j$ from T_{train}.
    if Metric-Based Methods **then**
      $\theta = \theta - \varphi \bigtriangledown_\theta \mathcal{H}_\theta(\mathbf{r}^{q'}_{mix_{i,j},k}, \mathbf{r}^{s'}_{mix_{i,j},k})$.
    else if Meta-Based Methods **then**
      $\theta = \theta - \varphi \bigtriangledown_\theta \mathcal{L}_\theta(\mathbf{r}^{meta'}_{mix_{i,j},k}, Q'_{mix_{i,j}})$.
      $s.t.\ \mathbf{r}^{meta'}_{mix_{i,j},k} = \lambda \mathbf{r}^{meta}_{i,k} + (1-\lambda)\mathbf{r}^{meta}_{j,k}$.
    end if
  end while
end if

score of them: $\mathcal{M}(\mathbf{r}^{q'}_{mix_{i,j},k}, \mathbf{r}^{s'}_{mix_{i,j},k})$. Since they represent the same new task relation $r'_{mix_{i,j}}$, their score should be as high as possible.

Meta Learner-Based Methods are different from metric learning-based methods to generate a new task; it not only needs to mixup the relation vector of the support set in the two tasks: $\mathbf{r}^{meta'}_{mix_{i,j}} = \lambda \mathbf{r}^{meta}_{i,\cdot} + (1-\lambda)\mathbf{r}^{meta}_{j,\cdot}$, but also needs to generate a corresponding query set:

$$Q'_{mix_{i,j}} = \{(\lambda \mathbf{h}^q_{i,k} + (1-\lambda)\mathbf{h}^q_{j,k}, \lambda \mathbf{t}^q_{i,k} + (1-\lambda)\mathbf{t}^q_{j,k})\}_{i\neq j} \quad (11)$$

Substituting $\mathbf{r}^{meta'}_{mix_{i,j}}$ and $Q'_{mix_{i,j}}$ into Eqn.3 is the bi-level optimization process.

5 Experiments

5.1 Experimental Setup

5.1.1 Datasets.

We evaluate our augmentation methods on two public benchmark datasets: NELL-One and Wiki-One[1]. In these datasets, few-shot relations $\mathcal{R}_f$ that have more than 50 but less than 500 triples are selected to construct few-shot tasks. There are 67 tasks and 183 tasks in NELL-One and Wiki-One datasets respectively. Correspondingly, the partition 51/5/11 of the 67 tasks and the partition 133/16/34 of the 183 tasks are used for training/validation/test. Furthermore, the background knowledge graph $\mathcal{G}'$ except few-shot relations are used to pre-train entity vectors and $\mathcal{R}_b$ vectors. The statistic details of both datasets are shown in Table 2.

5.1.2 Comparison Methods.

In order to evaluate the effectiveness of our augmentation methods, We conduct experiments on three metric learning-based methods and two meta learner-based methods: GMatching, FSRL, FAAN and MetaR, GANA (model details in Section 3.2). All the above methods use the original datasets for training without data augmentation.

5.1.3 Implementation Details.

For all the models, we initialize the entity and relation embeddings by background knowledge graphs pre-trained on TransE, released by GMatching. The K-shot (K = 1, 5) support pairs are selected randomly and experimented for all the models. For a fair comparison, we run the official code and adopt the default hyperparameters for each baseline. GMatching and FSRL do not report the experimental results in the 5-shot case, but we can adopt the results reported by FAAN for these two models in the 5-shot case. Moreover, FSRL and FAAN do not report the experimental results in the 1-shot case, so we run their released code to get baseline results in the 1-shot case. we re-implement the GANA model to make a fair comparison. For MetaR, we choose both pre-train setting and in-train setting to evaluation our augmentation methods. We set $\alpha = 2$ and $\beta = 2$ in $Beta(\alpha, \beta)$ and the and the neighborhood's maximum size is fixed to 50 on both datasets. For other hyperparameters, we adopt the default value of their released code.

5.1.4 Evaluation Metrics.

To evaluate the performance of all models on our augmentation methods, which aims to rank the ground truth tail entity $t^q_{k,i}$ for each query among the task candidates $\mathcal{C}_k$. We report two standard evaluation metrics on both datasets: MRR and Hits@N. MRR is the mean reciprocal rank and Hits@N is the proportion of the ground truth entities ranked in the top N; in our experiments, we set N = 1, 5, 10 and the few-shot size is set to K=1, 3.

5.2 Experimental Results and Analysis

The MRR results of FKGC models with all augmentation methods on NELL-One and Wiki-One are shown in Table 1, we can conclude that:

1. Our augmentation methods applied to all baseline models improve their original MRR val-

[1]https://github.com/xwhan/One-shot-Relational-Learning

NELL-One	Methods	Shot	Vanilla	Intra-Task		Inter-Task	
				Query	Support	Inverse	Interpolation
Metric-based	Gmatching	1-shot	0.168	$0.185^{+0.017}$	$0.175^{+0.007}$	$0.179^{+0.011}$	$0.205^{+0.037}$
		5-shot	0.176	$0.191^{+0.015}$	$0.180^{+0.004}$	$0.196^{+0.020}$	$0.211^{+0.035}$
	FSRL	1-shot	0.148	$0.172^{+0.024}$	$0.164^{+0.016}$	$0.157^{+0.009}$	$0.179^{+0.031}$
		5-shot	0.153	$0.178^{+0.025}$	$0.165^{+0.012}$	$0.169^{+0.016}$	$0.185^{+0.032}$
	FAAN	1-shot	0.194	$0.231^{+0.037}$	$0.216^{+0.022}$	$0.209^{+0.015}$	$0.224^{+0.030}$
		5-shot	0.279	$0.304^{+0.025}$	$0.282^{+0.003}$	$0.284^{+0.005}$	$0.294^{+0.015}$
Meta-based	MetaR (Pre-Train)	1-shot	0.164	$0.204^{+0.040}$	$0.227^{+0.063}$	$0.217^{+0.053}$	$0.194^{+0.030}$
		5-shot	0.209	$0.224^{+0.015}$	$0.240^{+0.031}$	$0.233^{+0.024}$	$0.217^{+0.008}$
	MetaR (In-Train)	1-shot	0.250	$0.308^{+0.058}$	$0.319^{+0.069}$	$0.254^{+0.004}$	$0.266^{+0.016}$
		5-shot	0.261	$0.331^{+0.070}$	$0.332^{+0.071}$	$0.275^{+0.014}$	$0.307^{+0.046}$
	GANA	1-shot	0.254	$0.278^{+0.024}$	$0.291^{+0.037}$	$0.286^{+0.032}$	$0.261^{-0.007}$
		5-shot	0.314	$0.326^{+0.012}$	$0.342^{+0.028}$	$0.334^{+0.020}$	$0.318^{+0.004}$
WiKi-One	Methods	Shot	Vanilla	Intra-Task		Inter-Task	
				Query	Support	Inverse	Interpolation
Metric-based	Gmatching	1-shot	0.200	$0.234^{+0.034}$	$0.224^{+0.024}$	$0.218^{+0.018}$	$0.215^{+0.015}$
		5-shot	0.245	$0.278^{+0.033}$	$0.263^{+0.018}$	$0.261^{+0.016}$	$0.256^{+0.011}$
	FSRL	1-shot	0.128	$0.157^{+0.029}$	$0.155^{+0.027}$	$0.136^{+0.008}$	$0.147^{+0.019}$
		5-shot	0.158	$0.186^{+0.028}$	$0.176^{+0.018}$	$0.171^{+0.013}$	$0.165^{+0.007}$
	FAAN	1-shot	0.272	$0.301^{+0.029}$	$0.285^{+0.013}$	$0.289^{+0.017}$	$0.279^{+0.007}$
		5-shot	0.341	$0.358^{+0.025}$	$0.349^{+0.008}$	$0.353^{+0.012}$	$0.348^{+0.007}$
Meta-based	MetaR (Pre-Train)	1-shot	0.314	$0.328^{+0.014}$	$0.335^{+0.021}$	$0.325^{+0.011}$	$0.319^{+0.005}$
		5-shot	0.323	$0.334^{+0.011}$	$0.347^{+0.024}$	$0.328^{+0.005}$	$0.331^{+0.008}$
	MetaR (In-Train)	1-shot	0.193	$0.198^{+0.005}$	$0.207^{+0.014}$	$0.190^{-0.003}$	$0.184^{-0.009}$
		5-shot	0.221	$0.232^{+0.011}$	$0.239^{+0.018}$	$0.227^{+0.006}$	$0.209^{-0.012}$
	GANA	1-shot	0.261	$0.272^{+0.011}$	$0.286^{+0.025}$	$0.266^{+0.005}$	$0.273^{+0.012}$
		5-shot	0.322	$0.338^{+0.016}$	$0.342^{+0.020}$	$0.331^{+0.009}$	$0.327^{+0.005}$

Table 1: Evaluation MRR of FKGC models with all augmentation methods on NELL-One and Wiki-One.

Dataset	#Ent.	#Rel.	#Triples	#Tasks
NELL-One	68,545	358	181,109	67
WiKi-One	4,838,244	822	5,859,240	183

Table 2: Statistics of datasets. Each column respectively represents the number of entities, relations, triples and tasks.

ues on both datasets upon all metrics. The experimental results indicate that our augmentation methods are effective for improving the existing FKGC models.

2. After support augmentation on NELL-One data, MetaR (Pre-Train) has increased by 38.4% compared to the original model, which is the largest increase. On WiKi-One data, query augmentation improves the MRR value of FSRL by 22.7%. The improvement on NELL-One is larger than that on WiKi-One because the Wiki dataset is more extensive, so the improvement brought by data augmentation is limited.

3. On the NELL-One dataset, intra-task augmentation is better than inter-task augmentation on metric learning-based models, but the opposite is true on meta learner-based models. On the WiKi-One dataset, intra-task augmentation outperforms inter-augmentation on all FKGC models. We conjecture the reason lies in that the WiKi-One dataset has more tasks than NELL-One, therefore increasing the number of triples within a task is more effective than increasing the number of tasks.

5.3 Combining Augmentations

After studying each mode of data augmentation individually, we combine intra-task augmentation and inter-task augmentation to understand the interplay between these two levels of augmentation methods. We select the best-performing FAAN model among metric learning-based methods for experiments. As shown in Table 4, the augmented

Model: MetaR (Pre-Train)									
NELL-One		MRR		Hits@10		Hits@5		Hits@1	
		1-shot	5-shot	1-shot	5-shot	1-shot	5-shot	1-shot	5-shot
Vanilla		0.164	0.209	0.331	0.355	0.238	0.280	0.093	0.141
Intra-Task	Query	0.204	0.224	0.376	**0.383**	0.295	0.298	0.131	0.149
	Support	**0.227**	**0.240**	**0.380**	0.376	**0.303**	**0.323**	**0.161**	0.157
Inter-Task	Inverse	0.217	0.233	0.375	0.359	0.289	0.296	0.156	**0.172**
	Interpolation	0.194	0.217	0.379	0.359	0.284	0.289	0.100	0.158
Wiki-One		1-shot	5-shot	1-shot	5-shot	1-shot	5-shot	1-shot	5-shot
Vanilla		0.314	0.323	0.404	0.418	0.375	0.385	**0.266**	**0.270**
Intra-Task	Query	0.328	0.334	**0.515**	0.504	0.437	0.442	0.233	0.242
	Support	**0.335**	**0.347**	0.509	0.507	**0.447**	**0.451**	0.235	0.256
Inter-Task	Inverse	0.325	0.328	0.499	**0.513**	0.433	0.436	0.233	0.237
	Interpolation	0.319	0.331	0.500	0.509	0.426	0.444	0.223	0.235

Table 3: Evaluation results of MetaR (Pre-Train) with data augmentation on NELL-One and Wiki-One.

mode	1-shot	5-shot
FAAN	0.194	0.279
+Query, Inverse	$0.240^{+0.046}$	$0.307^{+0.028}$
+Query, Interp.	$0.225^{+0.031}$	$0.297^{+0.018}$
+Support, Inverse	$0.221^{+0.027}$	$0.288^{+0.009}$
+Support, Interp.	$0.217^{+0.023}$	$0.286^{+0.007}$

Table 4: MRR results of FAAN combining augmentations variants on NELL-One dataset. Interp. denote interpolation.

model outperforms the original one under all settings. Combined with Figure 2, we find the jointly augmented models achieve better results than models using only one augmentation method. It shows that the combination of inter-task augmentation and intra-task augmentation is able to further improve the results.

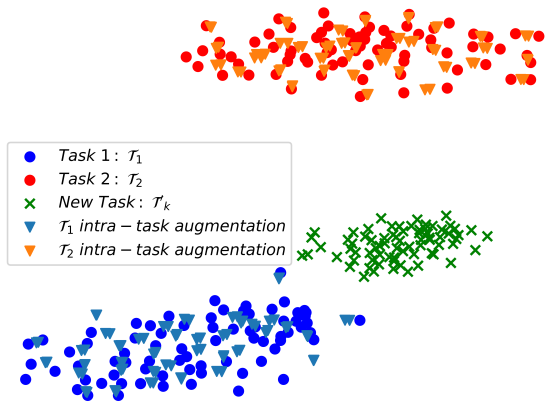

Figure 2: Visualization of relation vectors generated by different augmentation methods

5.4 Hits@N for case study

MetaR improves the most with all augmentation methods, and to get a complete picture of its performance; we further analyze it using Hits@N, which is summarized in Table 3. The augmentation methods bring the greatest improvement on Hits@5 and Hits@10, indicating that the augmentation methods mainly rely on the top-ranked recall to improve the overall MRR value. The Hit@N of the WiKi-One dataset is generally better than that of the NELL-One dataset under the same settings, because the former has more data to train.

5.5 Visualization

To better demonstrate the effectiveness of our augmentation methods, we visualize the new relation vectors generated by various augmentation methods in a 2-dimensional plane, i.e., using t-SNE (Van der Maaten and Hinton, 2008) for dimension reduction. As shown in figure 2, a new task is generated using interpolation augmentation, which can be well distinguished from other tasks and has a small intra-task distance. Intra-task augmentation for existing tasks can generate more relational vectors within the cluster. Therefore, the visualization results validate the effectiveness of our data augmentation method for FKGC.

5.6 Results on Different Relations

In addition to evaluating the augmented performance of all models, we also conduct experiments with FSRL on the NELL-One test data to evaluate the performance of each task relation. Table 5

reports the original performance of FSRL and the MRR after our augmentation methods. It can be seen from the table that no matter which augmentation method is used, the variance of the results is high in different task relations. The main reason for this is that the number of candidate entities is different, and large candidate sets make prediction difficult. Nonetheless, our augmentation methods outperform the baseline results on all relations, especially the interpretation augmentation method performs best on the FSRL model, indicating that our augmentation methods are robust to different task relations.

R-ID	Vanilla	Intra-task		Inter-task	
		Query	Support	Inver.	Interp.
1	0.975	0.982	0.980	0.982	**0.983**
2	0.064	0.072	0.068	0.070	**0.085**
3	0.472	0.601	0.602	0.595	**0.610**
4	0.005	0.008	0.007	0.008	**0.011**
5	0.210	0.242	0.232	0.268	**0.272**
6	0.045	0.048	0.047	0.049	**0.063**
7	0.141	0.163	0.149	0.156	**0.231**
8	0.118	0.121	0.123	**0.133**	0.128
9	0.561	0.562	0.550	0.566	**0.586**
10	0.009	0.011	0.010	0.012	**0.023**
11	0.373	0.397	0.378	0.394	**0.427**

Table 5: FSRL mrr results with 5-shot reference decomposed over different relations in NELL-One test dataset. R-ID denote relation id, Inver. and Interp. denotes Inverse augmentation and Interpolation augmentation.

6 Conclusion

To alleviate the limited data problem in the FKGC task. In this paper, we propose to utilize the data augmentation technique to enrich the training set for FKGC models. Specifically, we design the data augmentation method from hierarchical perspectives. The inter-task perspective generates new tasks for the FKGC task, while the intra-task perspective provides more entity pairs for each task. Furthermore, in order to fully perform data augmentation, we design two augmentation methods for each perspective, i.e., inverse augmentation and interpolation augmentation for the inter-task view, query augmentation and support augmentation for the intra-task view. Experimental results validate the effectiveness of the proposed method.

Acknowledgements

The research work is supported by the National Natural Science Foundation of China under Grant No.62206266. Zhao Zhang is supported by the China Postdoctoral Science Foundation under Grant No. 2021M703273.

References

Christopher M. Bishop. 1995. Training with noise is equivalent to tikhonov regularization. *Neural Computation*.

Kurt Bollacker, Colin Evans, Praveen Paritosh, Tim Sturge, and Jamie Taylor. 2008. Freebase: a collaboratively created graph database for structuring human knowledge. *international conference on management of data*.

Antoine Bordes, Nicolas Usunier, Alberto Garcia-Duran, Jason Weston, and Oksana Yakhnenko. 2013. Translating embeddings for modeling multi-relational data. *neural information processing systems*.

Mingyang Chen, Wen Zhang, Wei Zhang, Qiang Chen, and Huajun Chen. 2019. Meta relational learning for few-shot link prediction in knowledge graphs. *arXiv preprint arXiv:1909.01515*.

Ekin D. Cubuk, Barret Zoph, Dandelion Mane, Vijay K. Vasudevan, and Quoc V. Le. 2019. Autoaugment: Learning augmentation strategies from data. *computer vision and pattern recognition*.

Tim Dettmers, Pasquale Minervini, Pontus Stenetorp, and Sebastian Riedel. 2017. Convolutional 2d knowledge graph embeddings. *arXiv: Learning*.

He He, Anusha Balakrishnan, Mihail Eric, and Percy Liang. 2017. Learning symmetric collaborative dialogue agents with dynamic knowledge graph embeddings. *meeting of the association for computational linguistics*.

Raphael Hoffmann, Congle Zhang, Xiao Ling, Luke Zettlemoyer, and Daniel S. Weld. 2011. Knowledge-based weak supervision for information extraction of overlapping relations. *meeting of the association for computational linguistics*.

Jang-Hyun Kim, Wonho Choo, and Hyun Oh Song. 2020. Puzzle mix: Exploiting saliency and local statistics for optimal mixup.

George A. Miller. 1992. Wordnet: a lexical database for english. *human language technology*.

Tom M. Mitchell, William W. Cohen, Estevam R. Hruschka, Partha Pratim Talukdar, Bishan Yang, Justin Betteridge, Andrew Carlson, Bhavana Dalvi, Matt Gardner, Bryan Kisiel, Jayant Krishnamurthy, Ni Lao, Kathryn Mazaitis, T. Mohamed, Ndapandula Nakashole, Emmanouil Antonios Platanios, Alan Ritter, Mehdi Samadi, Burr Settles, Richard Wang, Derry Tanti Wijaya, Abhinav Gupta, Xinlei Chen, Abulhair Saparov, M. Greaves, and J. Welling. 2018.

Never-ending learning. *Communications of The ACM*.

Guanglin Niu, Yang Li, Chengguang Tang, Ruiying Geng, Jian Dai, Qiao Liu, Hao Wang, Jian Sun, Fei Huang, and Luo Si. 2021. Relational learning with gated and attentive neighbor aggregator for few-shot knowledge graph completion. In *Proceedings of the 44th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 213–222.

Apoorv Saxena, Aditay Tripathi, and Partha Pratim Talukdar. 2020. Improving multi-hop question answering over knowledge graphs using knowledge base embeddings. *meeting of the association for computational linguistics*.

Jiawei Sheng, Shu Guo, Zhenyu Chen, Juwei Yue, Lihong Wang, Tingwen Liu, and Hongbo Xu. 2020. Adaptive attentional network for few-shot knowledge graph completion. *arXiv preprint arXiv:2010.09638*.

Fabian M. Suchanek, Gjergji Kasneci, and Gerhard Weikum. 2007. Yago: a core of semantic knowledge. *the web conference*.

Zhiqing Sun, Zhi-Hong Deng, Jian-Yun Nie, and Jian Tang. 2019. Rotate: Knowledge graph embedding by relational rotation in complex space. *international conference on learning representations*.

Kristina Toutanova, Danqi Chen, Patrick Pantel, Hoifung Poon, Pallavi Choudhury, and Michael Gamon. 2015. Representing text for joint embedding of text and knowledge bases. In *Proceedings of the 2015 conference on empirical methods in natural language processing*, pages 1499–1509.

Laurens Van der Maaten and Geoffrey Hinton. 2008. Visualizing data using t-sne. *Journal of machine learning research*, 9(11).

Petar Veličković, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Lio, and Yoshua Bengio. 2017. Graph attention networks. *arXiv preprint arXiv:1710.10903*.

Vikas Verma, Alex Lamb, Christopher Beckham, Amir Najafi, Ioannis Mitliagkas, Aaron Courville, David Lopez-Paz, and Yoshua Bengio. 2018. Manifold mixup: Better representations by interpolating hidden states. *international conference on machine learning*.

Vikas Verma, Alex Lamb, Christopher Beckham, Amir Najafi, Ioannis Mitliagkas, David Lopez-Paz, and Yoshua Bengio. 2019. Manifold mixup: Better representations by interpolating hidden states. In *International Conference on Machine Learning*, pages 6438–6447. PMLR.

Denny Vrandecic and Markus Krötzsch. 2014. Wikidata: a free collaborative knowledgebase. *Communications of The ACM*.

Wenhan Xiong, Mo Yu, Shiyu Chang, Xiaoxiao Guo, and William Yang Wang. 2018. One-shot relational learning for knowledge graphs. *arXiv preprint arXiv:1808.09040*.

Chuanguang Yang, Zhulin An, Helong Zhou, Linhang Cai, Xiang Zhi, Jiwen Wu, Yongjun Xu, and Qian Zhang. 2022. Mixskd: Self-knowledge distillation from mixup for image recognition. In *European Conference on Computer Vision*.

Sangdoo Yun, Dongyoon Han, Seong Joon Oh, Sanghyuk Chun, Junsuk Choe, and Youngjoon Yoo. 2019. Cutmix: Regularization strategy to train strong classifiers with localizable features. *international conference on computer vision*.

Chuxu Zhang, Huaxiu Yao, Chao Huang, Meng Jiang, Zhenhui Li, and Nitesh V Chawla. 2020. Few-shot knowledge graph completion. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pages 3041–3048.

Hongyi Zhang, Moustapha Cisse, Yann N. Dauphin, and David Lopez-Paz. 2017. mixup: Beyond empirical risk minimization. *Learning*.

CLIO: Role-interactive Multi-event Head Attention Network for Document-level Event Extraction

Yubing Ren[1,2], Yanan Cao[1,2], Fang Fang[1,2*], Ping Guo[1,2]
Zheng Lin[1,2], Wei Ma[1,2*] and Yi Liu[3]
[1]Institute of Information Engineering, Chinese Academy of Sciences, Beijing, China
[2]School of Cyber Security, University of Chinese Academy of Sciences, Beijing, China
[3]National Computer Network Emergency Response Technical Team/Coordination Center of China
{renyubing,caoyanan,fangfang0703,guoping,linzheng,mawei}@iie.ac.cn
liuyi@cert.org.cn

Abstract

Transforming the large amounts of unstructured text on the Internet into structured event knowledge is a critical, yet unsolved goal of NLP, especially when addressing document-level text. Existing methods struggle in Document-level Event Extraction (DEE) due to its two intrinsic challenges: (a) Nested arguments, which means one argument is the substring of another one. (b) Multiple events, which indicates we should identify multiple events and assemble the arguments for them. In this paper, we propose a role-interactive multi-event head attention network (CLIO) to solve these two challenges jointly. The key idea is to map different events to multiple subspaces (i.e., multi-event head). In each event subspace, we draw the semantic representation of each role closer to its corresponding arguments, then we determine whether the current event exists. To further optimize event representation, we propose an event representation enhancing strategy to regularize pre-trained embedding space to be more isotropic. Our experiments on two widely used DEE datasets show that CLIO achieves consistent improvements over previous methods.

1 Introduction

Cognitive scientists believe that humans remember and understand reality primarily in terms of events (Shipley and Zacks, 2008). Event studies are justifiably popular in Natural Language Processing (NLP), such as Event Coreference Resolution, Event Causality Identification, and Event Extraction. Event extraction is the process of extracting structured event knowledge from unstructured text and can be divided into sentence-level and document-level. Sentence-level Event Extraction has demonstrated promising results in empirical evaluations. However, in real-world scenarios, a large number of event elements are expressed

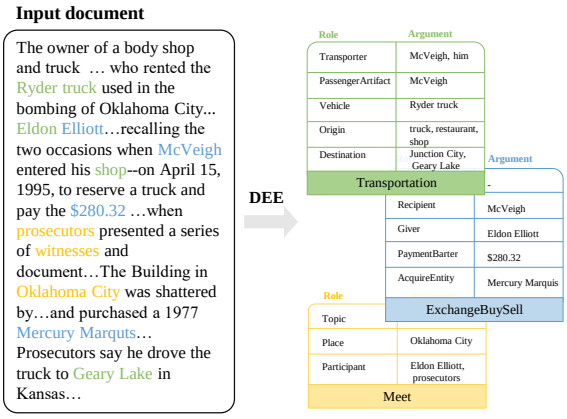

Figure 1: An illustration of DEE task. Different colored tabels indicate different event types. DEE needs to detect multiple event types and extract arguments for the roles of each event type.

across sentences. Document-level Event Extraction (DEE) is needed when we want to capture complete event information for the whole document. In contrast to SEE, increased text length brings more challenges, and DEE has still been underachieving.

Recently, researchers have shown an increased interest in DEE. Their works can be roughly divided into classification-based models (Zhang et al., 2020; Xu et al., 2021; Huang and Jia, 2021; Huang and Peng, 2021), tagging-based models (Yang et al., 2018; Du and Cardie, 2020), and generation-based models (Li et al., 2021; Yang et al., 2021; Du et al., 2021). The state-of-the-art approach (Liu et al., 2021) frames DEE as a machine reading comprehension task, assisted by two data augmentation regimes. Although scholars have made such valuable attempts in DEE, current methods still struggle in DEE due to the following crucial challenges:

Nested arguments: In a document, there are many nested arguments (i.e., one argument is the substring of another one) that belong to different roles. Figure 1 gives an example. In the "Trans-

*Fang Fang and Wei Ma are the co-corresponding authors

Proceedings of the 29th International Conference on Computational Linguistics, pages 2504–2514
October 12–17, 2022.

portation" event, "truck" (plays *Origin* role) and "Ryder truck" (plays *Vehicle* role) are nested event arguments. According to our statistics, 14.23% and 13.94% of documents in the WikiEvents (Li et al., 2021) and RAMS (Ebner et al., 2020) datasets have nested arguments, respectively. Unfortunately, these nested arguments can't be entirely identified by traditional tagging-based methods, which can not assign multiple labels to a token.

Multiple events: As shown in Figure 1, there are three kinds of events: "Transportation", "ExchangeBuySell", and "Meet" in a single document, and DEE should not only identify all events but also assign arguments to the corresponding events. The issue of multiple events is common in DEE (86.88% of documents in the WikiEvents involve multiple events). What's more, the arguments of these events are uniformly scattered across sentences, making it hard to achieve accurate arguments assembling. Previous works usually adopt a fixed document representation to detect all event types. However, different event types have different roles and arguments, and the emphasis of document representation should also be different.

For the nested arguments, which usually belong to different roles, the intuition is that we should extract arguments for each role independently. Assuming there are N roles in an event, we can perform N independent extractions by tagging arguments under each role. In this way, the argument substring "truck" of role "Origin" and the argument "Ryder truck" of role "Vehicle" can be identified at the same time. To address the challenge of multiple events, an intuitive way is to independently detect each event type and assemble arguments for it. For one event type, argument extraction can be simpler due to the decrease in roles. On the contrary, using role information specific to this event type can better detect the current event type. We argue that these two challenges can be solved jointly by mapping each event type to a specific subspace.

Analogy to multi-head attention (Vaswani et al., 2017), we propose a role-intera**C**tive mu**L**ti-event head attent**I**on netw**O**rk (CLIO) for DEE. The most critical part in CLIO is *Role-interactive Multi-event Head Attention* module, which can solve the aforementioned two challenges jointly. First, our attention module works in a role-centric way. That is to say, for each role, we extract all of its corresponding arguments independently. In this way, a token can be assigned multiple role labels, which can perfectly solve nested arguments problem. Second, our attention module assigns each event type a subspace by mapping it to each event head. In this way, we can independently detect each event type and assemble arguments for it, which can address the challenge of multiple events. In each event head, we use role information specific to this event to represent document. Such event-specific document representation eases the difficulty of detecting multiple events from a single document.

In summary, our contributions are as follows:

- We propose a role-interactive multi-event head attention network to handle the challenges of nested arguments and multiple events simultaneously.
- We conduct experiments on two widely used DEE datasets. Experimental results demonstrate that CLIO outperforms previous methods and has significant improvement when facing the vital challenges of DEE.

2 Methodology

We first describe the task formalization of DEE. Formally, given an input document comprised of m words $\mathcal{D} = \{w_i\}_{i=1}^{m}$, pre-defined event types $\mathcal{T} = \{t_i\}_{i=1}^{l}$, and role categorizies $\mathcal{R} = \{r_i\}_{i=1}^{n}$. The DEE task aims to extract one or more event records: $\{\text{event type} : t, r_1 : [a_1^1, a_1^2, ...], ..., r_i : [a_i^1, a_i^2, ...]\}$, where a_i^1 is the first argument of role r_i, and so on.

Figure 2 illustrates the architecture of CLIO, which consists of three key components: (1) Role-interactive Multi-event Head Attention, (2) Multiple Events Extraction, and (3) Event Representation Enhancing. *Role-interactive Multi-event Head Attention* module can solve the challenges of nested arguments and multiple events simultaneously.

2.1 Encoding

First, we construct an extended sequence $S = [\text{CLS}]\mathcal{D}[\text{SEP}]\mathcal{R}[\text{SEP}]$ by concatenating the document $\mathcal{D}$ and role set $\mathcal{R}$. Next, we use BERT (Devlin et al., 2019) with hidden size d to encode contextual embeddings of each word in the sequence S:

$$[\mathrm{H}_w, \mathrm{H}_r] = \mathrm{BERT}(S) \quad (1)$$

After this stage, we can obtain the word representation of document $\mathrm{H}_w \in \mathbb{R}^{m \times d}$ and role representation $\mathrm{H}_r \in \mathbb{R}^{n \times d}$. This stage makes a deep fusion

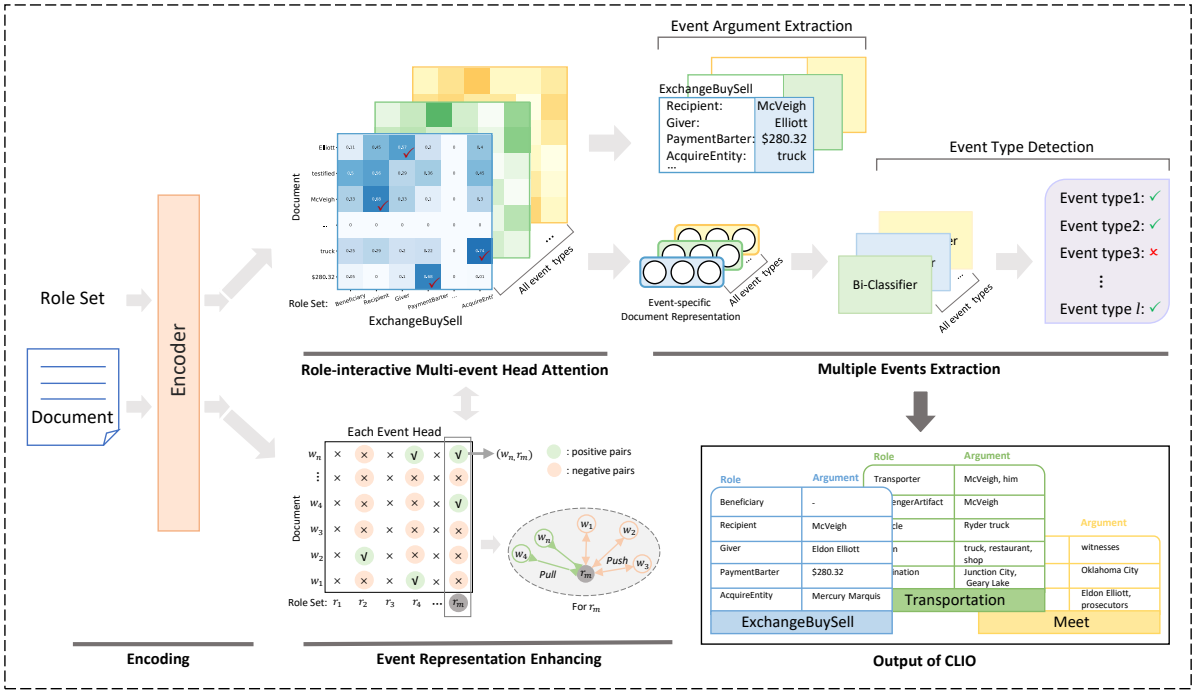

Figure 2: The overall architecture of CLIO. Role-interactive Multi-event Head Attention is designed to map each event type to a specific subspace. In each subspace, we compare the roles and words to measure the degree of relevance among them. Event Representation Enhancing is used to further optimize event representation. In Multiple Events Extraction, we perform the two subtasks of DEE.

between the document and roles by multi-head and multi-layer attention.

Norm-based Significance Score Intuitively, not every word in document is significant. So we introduce a norm-based significance score to measure the ability of words to express essential meaning based on the L2-Norm of word embedding. This feature of L2-Norm has already been proven by some promising works (Luhn, 1958; Chen et al., 2020; Liu et al., 2020).

We use the L2-Norm of word embeddings as the weight of them:

$$\mathrm{H}'_w = \|\mathrm{H}_w\|_2 \odot \mathrm{H}_w \tag{2}$$

where $\mathrm{H}'_w \in \mathbb{R}^{m \times d}$ is the weighted word embedding, $\odot$ means element-wise multiplication.

2.2 Role-interactive Multi-event Head Attention

In this step, the goal is to solve the challenges of nested arguments and multiple events simultaneously. We compare the role embeddings and word embeddings under each event type and select role-word pairs that have high semantic overlap as argument extraction results. We first consider a single event type, then extend it to all event types.

Role-interactive Event Attention

In each event type, we measure the degree of relevance between each role-word pair. We first project the original d-dimensional features of words and roles into a smaller dimension d' through two fully connected layers:

$$\begin{aligned} \tilde{\mathrm{H}}_w &= \mathrm{H}'_w \mathrm{W}_w + \mathrm{b}_w \\ \tilde{\mathrm{H}}_r &= \mathrm{H}_r \mathrm{W}_r + \mathrm{b}_r \end{aligned} \tag{3}$$

where $\mathrm{W}_w \in \mathbb{R}^{d \times d'}, \mathrm{b}_w \in \mathbb{R}^{d'}, \mathrm{W}_r \in \mathbb{R}^{d \times d'}, \mathrm{b}_r \in \mathbb{R}^{d'}$ are learnable parameters, $\tilde{\mathrm{H}}_w \in \mathbb{R}^{m \times d'}, \tilde{\mathrm{H}}_r \in \mathbb{R}^{n \times d'}$.

Then we apply concat attention (Luong et al., 2015) to measure the degree of relevance between word representation $\tilde{\mathrm{H}}_w$ and role representation $\tilde{\mathrm{H}}_r$. We indicate $\mathrm{S}_t(\tilde{\mathrm{H}}_w, \tilde{\mathrm{H}}_r)$ as the correlation intensity matrix of role-word pairs under the event type t:

$$\begin{aligned} \textsc{Score}_t(\tilde{\mathrm{H}}_w, \tilde{\mathrm{H}}_r) &= \tanh([\tilde{\mathrm{H}}_w; \tilde{\mathrm{H}}_r]\mathrm{W}_a) \cdot v_a \\ \mathrm{S}_t(\tilde{\mathrm{H}}_w, \tilde{\mathrm{H}}_r) &= \mathrm{sigmoid}(\textsc{Score}_t(\tilde{\mathrm{H}}_w, \tilde{\mathrm{H}}_r)) \end{aligned} \tag{4}$$

where $[\tilde{\mathrm{H}}_w; \tilde{\mathrm{H}}_r] \in \mathbb{R}^{m \times n \times 2d'}$, $\mathrm{W}_a \in \mathbb{R}^{2d' \times d'}$ and $v_a \in \mathbb{R}^{d'}$ are learnable parameters, $\textsc{Score}_t \in \mathbb{R}^{m \times n}, \mathrm{S}_t \in \mathbb{R}^{m \times n}, t \in \mathcal{T}$.

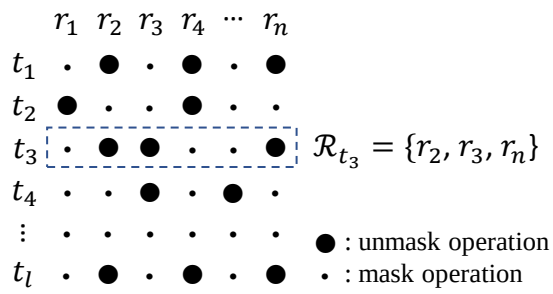

Figure 3: Event schema mask **M**. t and r denote event type and role type, respectively. In each event type, we mask those roles not in the pre-defined role set.

Multi-event Head Attention

We perform the above role-interactive event attention on all event heads in parallel, which can extract multiple events simultaneously. Formally, we stack the role-interactive event attentions under all event types to a multi-event head attention $\mathrm{S}_{\mathcal{T}}(\tilde{\mathrm{H}}_w, \tilde{\mathrm{H}}_r) \in \mathbb{R}^{l\times m\times n}$, where the number of heads l is the size of event types.

For the DEE dataset, each event type t_i has a pre-defined role set $\mathcal{R}_{t_i}$ [1]. We formalize it as the event schema mask **M** (see Figure 3):

$$\mathbf{M}_{t_i,r_j} = \begin{cases} 1, & \text{role } r_j \text{ in } \mathcal{R}_{t_i} \\ 0, & \text{role } r_j \text{ not in } \mathcal{R}_{t_i} \end{cases} \tag{5}$$

Through the event schema mask **M**, we decrease the number of roles to predict under each event type, leave each event type a unique role candidate set and make a difference among event heads. The final multi-event correlation intensity matrix $\mathrm{S}^{multi}(\tilde{\mathrm{H}}_w, \tilde{\mathrm{H}}_r)$ is caculated as:

$$\mathrm{S}^{multi}(\tilde{\mathrm{H}}_w, \tilde{\mathrm{H}}_r) = \mathrm{S}_{\mathcal{T}}(\tilde{\mathrm{H}}_w, \tilde{\mathrm{H}}_r) \odot \mathbf{M} \tag{6}$$

where $\mathrm{S}_{\mathcal{T}} \in \mathbb{R}^{l\times m\times n}, \mathbf{M} \in \mathbb{R}^{l\times n}, \mathrm{S}^{multi} \in \mathbb{R}^{l\times m\times n}$.

2.3 Multiple Events Extraction

Event Argument Extraction

The final multi-event correlation intensity matrix $\mathrm{S}^{multi}(\tilde{\mathrm{H}}_w, \tilde{\mathrm{H}}_r)$ (Eq. 6) contains probabilities for each role-word pair. We take those role-word pairs whose probabilities are higher than threshold δ_{EAE} as the argument extraction results under the current role.

We use cross-entropy between the predictions and golden labels to optimize our model:

$$\mathcal{L}_{\text{EAE}} = \text{CE}(\mathrm{S}^{multi}, \mathrm{Y}^{multi}) \tag{7}$$

[1]The pre-defined role sets are provided by DEE dataset.

where $\mathrm{Y}^{multi} \in \mathbb{R}^{l\times m\times n}$ is the ground truth label for the correlation matrix between a document and roles under the multi-event head.

Event-specific Document Representation

To better detect which event type is contained in the document, we construct event-specific document representation for each event head.

Given the role-interactive event attention $\textsc{Score}_t(\tilde{\mathrm{H}}_w, \tilde{\mathrm{H}}_r)$ (Eq. 4) under event type t, we first normalize $\textsc{Score}_t$ with respect to role, referred to as A. We obtain word representation H''_w specific to event type t by using A to weighted sum the roles $\tilde{\mathrm{H}}_r$. Through the mean pooling operation we obtain document representation H_D specific to event type t:

$$\begin{aligned} A &= \text{softmax}_r(\textsc{Score}_t(\tilde{\mathrm{H}}_w, \tilde{\mathrm{H}}_r) \odot \mathbf{M}_{t,:}) \\ \mathrm{H}''_w &= A\tilde{\mathrm{H}}_r \\ \mathrm{H}_D &= \frac{1}{m}\sum_{w=w_1}^{w_m} \mathrm{H}''_w \end{aligned} \tag{8}$$

where $\textsc{Score}_t \in \mathbb{R}^{m\times n}$, $\mathbf{M}_{t,:} \in \mathbb{R}^n$, $A \in \mathbb{R}^{m\times n}$, $\sum A_{i,:} = 1$, $\mathrm{H}''_w \in \mathbb{R}^{m\times d'}$, $\mathrm{H}_D \in \mathbb{R}^{d'}$.

Event Type Detection

We detect each event type based on the corresponding event-specific document representation. Concretely, we perform binary classification on H_D for event type t to get the probability P_t:

$$\mathrm{P}_t = \text{softmax}(\mathrm{H}_D\mathrm{W}_e) \tag{9}$$

where $\mathbf{W}_e \in \mathbb{R}^{d'\times 2}$ is learnable parameters, $\mathbf{P}_t \in \mathbb{R}^2$.

Then we expand P_t to the prediction of multiple events $\mathrm{P}^{event} \in \mathbb{R}^{l\times 2}$, which can identify multiple events simultaneously. We apply cross-entropy loss to update the model paremeters:

$$\mathcal{L}_{\text{ED}} = \text{CE}(\mathrm{P}^{event}, \mathrm{Y}^{event}) \tag{10}$$

where $\mathrm{Y}^{event} \in \mathbb{R}^{l\times 2}$ is the ground truth label for the event type.

2.4 Event Representation Enhancing

We find the language modeling of the above stages produces anisotropic word embeddings. So we apply intra-event contrastive learning to enhance event representation by regularizing pre-trained embedding space to be more isotropic. In DEE, we need to pull each role closer to its arguments (positives) while pushing each role away from other

words (negatives). Given a role, there are multiple arguments, i.e., there is more than one positive.

We apply an approach, proposed by (Hoffmann et al., 2022) based on InfoNCE, to include multiple positives. More specifically, for a role $\mathbf{h}_r \in \tilde{\mathrm{H}}_r$, words that are the arguments of role r form the set of positives $\mathcal{P}$, and words that are not the arguments of role r form the set of negatives $\mathcal{N}$, $\mathcal{P} \cup \mathcal{N} = \tilde{\mathrm{H}}_w$. To measure the similarity between a pair of features, we use the cosine similarity:

$$\mathrm{sim}(\mathbf{x}, \mathbf{y}) = \frac{\mathbf{x}^\top \mathbf{y}}{\|\mathbf{x}\| \cdot \|\mathbf{y}\|} \tag{11}$$

The training objective becomes:

$$\mathcal{L}_{\mathrm{CL}} = -\log \frac{\sum\limits_{\mathbf{p} \in \mathcal{P}} \exp(\frac{\mathrm{sim}(\mathbf{h}_r, \mathbf{p})}{\tau})}{\sum\limits_{\mathbf{p} \in \mathcal{P}} \exp(\frac{\mathrm{sim}(\mathbf{h}_r, \mathbf{p})}{\tau}) + \sum\limits_{\mathbf{n} \in \mathcal{N}} \exp(\frac{\mathrm{sim}(\mathbf{h}_r, \mathbf{n})}{\tau})} \tag{12}$$

where $\mathbf{h}_r \in \mathbb{R}^{d'}$ is the embedding of role r, $\mathbf{p} \in \mathbb{R}^{d'}$ is the argument embedding, and $\mathbf{n} \in \mathbb{R}^{d'}$ is the word embedding of the input document other than arguments , τ is a temperature hyperparameter.

2.5 Joint Learning

The overall loss function is divided into three parts: an event argument extraction loss $\mathcal{L}_{\mathrm{EAE}}$, an event type detection loss $\mathcal{L}_{\mathrm{ED}}$, and a contrastive loss $\mathcal{L}_{\mathrm{CL}}$. We let these three objectives learn jointly at the same speed and update model parameters together. We have the following training loss:

$$\mathcal{L} = \lambda_1 \mathcal{L}_{\mathrm{EAE}} + \lambda_2 \mathcal{L}_{\mathrm{ED}} + \lambda_3 \mathcal{L}_{\mathrm{CL}} \tag{13}$$

and λ_1, λ_2 are the weight dynamically adjusted with the training steps, where $\lambda_1 = \frac{1}{\mathcal{L}_{\mathrm{EAE}}}, \lambda_2 = \frac{1}{\mathcal{L}_{\mathrm{ED}}}$[2], λ_3 is hyperparameter.

3 Experiments

We evaluate our model's performance on the two commonly used DEE benchmarks and compare to prior work. Then we conduct an ablation study on how modules of our CLIO affect its performance on DEE task. We also conduct case study to analyze qualitatively the advantages and disadvantages of our model.

[2] λ_1 and λ_2 only take the value of $\mathcal{L}_{\mathrm{EAE}}$ and $\mathcal{L}_{\mathrm{ED}}$, which contain no gradient information.

3.1 Experimental Setup

Datasets. We conduct our experiments on two widely used document-level event extraction datasets: RAMS (Ebner et al., 2020) and WikiEvents (Li et al., 2021). RAMS provides 9,124 annotated examples from news based on 139 event types and 65 roles. WikiEvents provides 246 annotated documents from news based on 50 event types and 59 roles. According to our statistics, 13.94% of documents in the RAMS have nested arguments. 14.23% and 86.99% of documents in the WikiEvents involve nested arguments and multiple events, respectively.

Evaluation Metrics. Our results are reported as Precision (P), Recall (R) and F-measure (F-1) score. Our argument extraction results are based on the Exact Match criterion: the predicted argument span should match exactly the gold one. As an event type often includes multiple roles, we use micro-averaged role-level scores as the final DEE metric.

Baselines. For strictly consistent comparison, we involve the following strong baselines:

- BERT-CRF (Loshchilov and Hutter, 2018), which combines BERT with Condition Random Field (Lafferty et al., 2001), is the most popular method in tagging-based event extraction.
- SpanSel (Ebner et al., 2020), which is based on span ranking, enumerates each possible span in a document to identify the most likely event arguments.
- Head-Expand (Zhang et al., 2020), which achieves state-of-the-art performance on the RAMS. It first identifies the head of an argument and then expands its region.
- BART-Gen (Li et al., 2021), which bases on the unfilled template and a given context, frames the implicit EAE as conditional generation.
- DocMRC (Liu et al., 2021), which frames DEE as Machine Reading Comprehension task, assisted by two data augmentation regimes.

Experimental Settings. We adopt pretrained BERT (Devlin et al., 2019) (bert-base-cased for English dataset), which has 12 hidden layers, each

Methods	RAMS				WikiEvents			
	ED-F1	EAE-P	EAE-R	EAE-F1	ED-F1	EAE-P	EAE-R	EAE-F1
BERT-CRF (Loshchilov and Hutter, 2018)†	-	36.7	41.1	38.8	-	54.4	23.8	33.1
SpanSel (Ebner et al., 2020)†	-	38.0	38.4	38.2	-	**56.2**	26.2	35.7
Head-Expand (Zhang et al., 2020)†	-	-	-	40.1	-	55.4	25.4	34.8
BART-Gen (Li et al., 2021)†	-	20.7	30.3	24.6	-	14.2	7.8	10.1
DocMRC (Liu et al., 2021)	-	41.2	45.2	43.1	-	58.5	**30.5**	**40.1**
CLIO	**44.5**	**47.6**	**45.5**	**46.5**	**52.4**	48.7	29.5	36.8
w/o $\mathcal{L}_{\text{CL}}$	43.4	46.9	44.5	45.7	52.5	51.8	25.6	34.3
w/o norm	43.3	47.1	45.3	46.2	50	50.2	24.0	32.5

Table 1: The ED (Event Type Detection) and EAE (Event Argument Extraction) results of all models on the RAMS and WikiEvents datasets. Results marked † are from (Liu et al., 2021). DocMRC uses the expanded training data, **5 times** and **30 times** larger than RAMS and WikiEvents, to train its model. w/o $\mathcal{L}_{\text{CL}}$ and w/o norm denote we remove contrastive loss and norm-based significance weight respectively.

layer has 768 hidden units, and 12 attention heads. During training, we adopt mini-batch mechanism to train our model with batch size of 16, and the maximum training epoch is set to 100. We regularize our network using dropout, the dropout ratio of linear is 0.3. The initial learning rate is 2e-5 for BERT parameters and 2e-3 for other parameters. We trained all models with the AdamW optimizer (Loshchilov and Hutter, 2018). The warming up proportion for learning rate is 10%. Besides, the threshold δ_{EAE} for RAMS and WikiEvents is 0.6 and 0.65, respectively. We set temperature hyper-parameter τ as 0.07. The weight of contrastive loss for RAMS and WikiEvents is 0.5 and 0.05, respectively.

In addition, the implementation of baselines does not consider gold event types. The experiments on RAMS consider event trigger information. We apply dot attention to measure the degree of relevance between role and trigger, and then we use the probability as the weight of role embedding.

3.2 Main Results

Table 1 presents our main results. Since the baselines do not have the capability for event type detection, the value of ED-F1 is replaced by '-'. We think event type detection is an integral part of DEE, while previous methods did not consider it. From Table 1, we can see that CLIO has the capability for event type detection. Our model surpasses all previous methods with 46.5 EAE-F1 score on the RAMS benchmark. Compared with the DocMRC, which uses 5 times more training data, our approach on the RAMS benchmark can bring substantial improvements in EAE, 3.4 F1 points. In the WikiEvents benchmark, our CLIO shows 3.0% drop in EAE-F1 scores compared to DocMRC. The reason is that DocMRC applies data augmentation, which expands training data to 30 times larger than WikiEvents. So it shows great advantages on small-scale dataset WikiEvents (expanding the original 246 documents to 7,491). Our CLIO reaches competitive results using only **1/30 data** compared with DocMRC. Compared with the SpanSel, our approach on the WikiEvents benchmark can bring 1.1 points of improvement in EAE-F1.

	RAMS		WikiEvents	
	Subset-N	Subset-O	Subset-N	Subset-O
DocMRC	40.6	43.4	39.9	**41.3**
CLIO	**47.1**$_{\uparrow 6.5}$	**44.0**$_{\uparrow 0.6}$	**46.2**$_{\uparrow 6.3}$	32.6

Table 2: Overall EAE-F1 with nested argument handling. Subset-N is a nested subset, while Subset-O is a non-nested subset.

CLIO can extract nested arguments accurately. We conduct additional experiments to evaluate the capability of CLIO to extract nested arguments. The idea is to split the test data into two portions: documents with and without nested arguments (Subset-N and Subset-O). Table 2 shows the results. On the Subset-N, CLIO significantly outperforms DocMRC by 6.5 F1 and 6.3 F1 on the RAMS and WikiEvents, respectively. We conclude that CLIO achieves superior performance in both datasets largely because it solves the nested argument issue.

CLIO can handle complex multi-event scenarios. Figure 4 presents the additional experiment results. From (a), we can observe that as the number of event types increases, the performance of CLIO increases instead of decreases while the per-

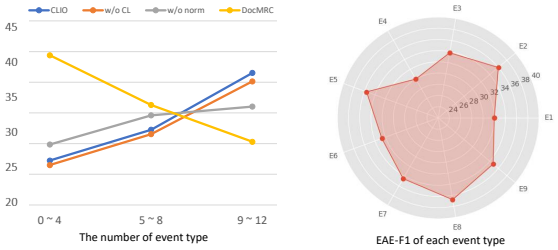

(a) (b)

Figure 4: (a) EAE-F1 scores on the WikiEvents with different numbers of event types. (b) Radar chart of the average EAE-F1. "E1, E2,... E9" are the event types CLIO identified. The red dot is the micro EAE-F1 on the role level.

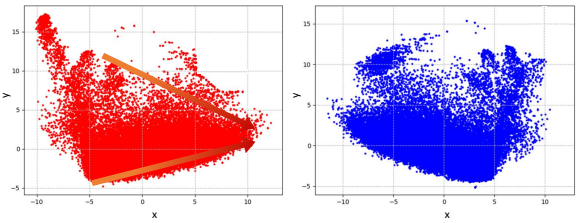

(a) CLIO w/o CL (b) CLIO

Figure 5: 2D visualization of projected word embeddings[3]. (a). Word embeddings trained by CLIO (without contrastive learning). (b). Word embeddings trained by CLIO.

formance of DocMRC decreases, which indicates CLIO can handle complex multi-event scenarios. In (b), we randomly choose a document from the test set of WikiEvents and calculate the EAE-F1 under each event type. We find the EAE-F1 values evenly distributed on all event types, which indicates CLIO has the capability to handle multiple events.

3.3 Ablation Study

We perform an ablation study to test how useful our event representation enhancing and norm-based significance weight. The results are shown in Table 1. Specifically, "w/o $\mathcal{L}_{\text{CL}}$" denotes contrastive loss is not considered in the joint learning, "w/o norm" means word embeddings are not weighted with norm significance score. In Figure 5, we compare the 2D visualization of word embeddings with or without contrastive learning. We can see that the event representation enhancing strategy can alleviate the representation degeneration problem and improve the isotropic properties of these representations.

[3]Note that we project the original word embeddings to a 2-dimensional vector space using principal component analysis (PCA) for the purpose of visualization.

3.4 Case Study

We present two examples from both datasets to illustrate the capability of CLIO. The examples are presented in Table 3, including input document and event extraction results. From Table 3, we find that CLIO can help DEE in two ways:

Handling nested arguments accurately In the first example, "U.S." and "U.S. officials" are nested arguments, belonging to the role "place" and the role "communicator" respectively. CLIO works in a role-centric way, which can extract both of them together and assign them to corresponding roles. This case demonstrates how role-interactive event attention can assign multiple role labels to each token and solve the challenges of nested arguments.

Handling complex multi-event scenarios In the second example, "E1", "E2", and "E3" indicate three different event types. In multi-event scenario, CLIO can not only identify all events but also assign arguments to the corresponding events. CLIO assigns each event type a subspace by mapping it to each event head, where the event-specific document representation eases the difficulty of detecting multiple events. This implies that CLIO is particularly helpful for the extraction of multi-event scenarios.

4 Related Work

Sentence-level Event Extraction SEE extracts the event trigger and its arguments from a single sentence. Researchers have made a lot of progress in this field. Li et al. (2013, 2015) employ various hand-designed features to extract event; (Nguyen and Grishman, 2015; Nguyen et al., 2016; Chen et al., 2015; Liu et al., 2017, 2018) use neural based models such as recurrent neural networks (Zaremba et al., 2014) and convolutional neural network (LeCun et al., 1998) to extract event. With the recent success of BERT (Devlin et al., 2019), pretrained language models have also been used for SEE (Wang et al., 2019b,c; Yang et al., 2019; Wadden et al., 2019; Tong et al., 2020; Wang et al., 2021; Lu et al., 2021; Liu et al., 2022). These approaches achieve remarkable performance in benchmarks such as ACE 2005 (Walker et al., 2005) and similar datasets (Ellis et al., 2015; Ji et al., 2016; Getman et al., 2017).

Document-level Event Extraction Different from SEE, DEE does not need to explicitly recognize event triggers. The goal of DEE is to iden-

Category	Example
Nested arguments	**Ex1.** From the media we discovered that some local authorities we approached coordinated their negative decision with the federal government...Reporters at the State Department 's daily press briefing on Friday asked if [[**U.S.**]$_{\text{place}}$**officials**]$_{\text{communicator}}$ had advised [**individual states**]$_{\text{recipient}}$ not to allow in Russian observers. (**Event type**: contact.requestadvise.correspondence)
Multiple events	**Ex2.** Japanese [**police**]$_{\text{E1-Jailer}}$ have arrested a [**man**]$_{\text{E1-Detainee}}$ who admitted to landing a drone with low-level radioactive sand on the roof of the prime minister's office...Tokyo metropolitan police said [**Yasuo Yamamoto**]$_{\text{E1-Detainee}}$, 40, turned himself in to authorities late Friday in Fukui in western Japan...The small [**drone**]$_{\text{E2-IdentifiedObject}}$ found Wednesday had traces of radiation and triggered fears of potential terrorist attacks using [**unmanned aerial devices**]$_{\text{E3-Instrument}}$...

Table 3: Case study on the RAMS (Ex1) and WikiEvents (Ex2) test sets. The bold text indicates the argument word. Predicted arguments are marked with [square brackets] span indicator. Ex2 includes multiple events, where E1: Justice.ArrestJailDetain, E2: Cognitive.IdentifyCategorize, E3: Conflict.Attack.

tify event types and extract arguments of roles from the whole document. On the task level, most of these works fall into three categories: (1) classification-based models (2) tagging-based models (3) generation-based models. Zhang et al. (2020); Xu et al. (2021); Huang and Jia (2021); Huang and Peng (2021) employ traditional classification paradigm to determine the event type, then they identify the arguments and classify the roles they play in an event; Yang et al. (2018); Du and Cardie (2020) use the sequence labeling model BiLSTM (Zhang et al., 2015) -CRF (Lafferty et al., 2001) to automatically extract events; Li et al. (2021) frame the problem as conditional generation. Yang et al. (2021) apply cross attention mechanism to extract structured events in a parallel manner. Above methods conduct experiments on MUC-4 (McLean, 1992), WikiEvents (Li et al., 2021), RAMS (Ebner et al., 2020), and Chinese financial dataset (Zheng et al., 2019).

Contrastive Learning In NLP, contrastive self-supervised learning has been widely used for learning better representations by contrasting positive pairs and negative pairs. The core idea is to concentrate positive samples while pushing apart negative samples. InfoNCE (Oord et al., 2018) is a frequently used objective function in contrastive learning. It maximizes the similarity of positive pairs and minimizes the similarity of negative pairs. More specifically, for a query q, a single positive p and a set of negatives $\mathcal{N} = \{n_1, ..., n_k\}$ is given. To measure the similarity between a pair of features, it uses the cosine similarity as the training objective:

$$\mathcal{L}_{\text{InfoNCE}} = -\log \frac{\exp(\frac{\text{sim}(q,p)}{\tau})}{\exp(\frac{\text{sim}(q,p)}{\tau}) + \sum_{n \in \mathcal{N}} \exp(\frac{\text{sim}(q,n)}{\tau})} \tag{14}$$

Anisotropy Gao et al. (2019); Wang et al. (2019a) have pointed out that language modeling usually produces an anisotropic word embedding space. This phenomenon is also observed in the pretrained Transformers like BERT, GPT-2, etc (Ethayarajh, 2019). Li et al. (2020) thinks that "anisotropic" means word embeddings occupy anarrow cone in the vector space. Through empirical analysis, we find that the word representations in documents have high cosine similarity between each other, which is known as anisotropic word embeddings. In a document, some words are event arguments while others are event-irrelevant, which means they should not learn similar word representations.

5 Conclusion

In this paper, we propose a role-interactive multi-event head attention network (CLIO) for DEE. By mapping different events to multiple subspaces, we decomposed DEE into multiple substeps to handle nested arguments and multiple events. To further optimize event representation, we apply an event representation enhancing strategy to regularize pretrained embedding space to be more isotropic. Experimental results show that CLIO can significantly outperform previous methods, especially when facing the specific challenges of DEE. In future work,

we would like to explore superior word representation specific to events.

Acknowledgements

This work is supported by the Youth Innovation Promotion Association of the Chinese Academy of Sciences (No.2018192), and the NSFC (No.61902394). We would like to thank the anonymous reviewers for their valuable comments.

References

Kehai Chen, Rui Wang, Masao Utiyama, and Eiichiro Sumita. 2020. Content word aware neural machine translation. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 358–364, Online. Association for Computational Linguistics.

Yubo Chen, Liheng Xu, Kang Liu, Daojian Zeng, and Jun Zhao. 2015. Event extraction via dynamic multi-pooling convolutional neural networks. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 167–176, Beijing, China. Association for Computational Linguistics.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, Minneapolis, Minnesota. Association for Computational Linguistics.

Xinya Du and Claire Cardie. 2020. Document-level event role filler extraction using multi-granularity contextualized encoding. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 8010–8020, Online. Association for Computational Linguistics.

Xinya Du, Alexander Rush, and Claire Cardie. 2021. Template filling with generative transformers. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 909–914, Online. Association for Computational Linguistics.

Seth Ebner, Patrick Xia, Ryan Culkin, Kyle Rawlins, and Benjamin Van Durme. 2020. Multi-sentence argument linking. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 8057–8077, Online. Association for Computational Linguistics.

Joe Ellis, Jeremy Getman, Dana Fore, Neil Kuster, Zhiyi Song, Ann Bies, and Stephanie M Strassel. 2015. Overview of linguistic resources for the tac kbp 2015 evaluations: Methodologies and results. In *TAC*.

Kawin Ethayarajh. 2019. How contextual are contextualized word representations? Comparing the geometry of BERT, ELMo, and GPT-2 embeddings. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 55–65, Hong Kong, China. Association for Computational Linguistics.

Jun Gao, Di He, Xu Tan, Tao Qin, Liwei Wang, and Tie-Yan Liu. 2019. Representation degeneration problem in training natural language generation models. *arXiv preprint arXiv:1907.12009*.

Jeremy Getman, Joe Ellis, Zhiyi Song, Jennifer Tracey, and Stephanie M Strassel. 2017. Overview of linguistic resources for the tac kbp 2017 evaluations: Methodologies and results. In *TAC*.

David T Hoffmann, Nadine Behrmann, Juergen Gall, Thomas Brox, and Mehdi Noroozi. 2022. Ranking info noise contrastive estimation: Boosting contrastive learning via ranked positives. *arXiv preprint arXiv:2201.11736*.

Kung-Hsiang Huang and Nanyun Peng. 2021. Document-level event extraction with efficient end-to-end learning of cross-event dependencies. In *Proceedings of the Third Workshop on Narrative Understanding*, pages 36–47, Virtual. Association for Computational Linguistics.

Yusheng Huang and Weijia Jia. 2021. Exploring sentence community for document-level event extraction. In *Findings of the Association for Computational Linguistics: EMNLP 2021*, pages 340–351, Punta Cana, Dominican Republic. Association for Computational Linguistics.

Heng Ji, Joel Nothman, H Trang Dang, and Sydney Informatics Hub. 2016. Overview of tac-kbp2016 tri-lingual edl and its impact on end-to-end cold-start kbp. *Proceedings of TAC*.

John Lafferty, Andrew McCallum, and Fernando CN Pereira. 2001. Conditional random fields: Probabilistic models for segmenting and labeling sequence data.

Yann LeCun, Léon Bottou, Yoshua Bengio, and Patrick Haffner. 1998. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324.

Bohan Li, Hao Zhou, Junxian He, Mingxuan Wang, Yiming Yang, and Lei Li. 2020. On the sentence embeddings from pre-trained language models. *arXiv preprint arXiv:2011.05864*.

Qi Li, Heng Ji, and Liang Huang. 2013. Joint event extraction via structured prediction with global features. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 73–82, Sofia, Bulgaria. Association for Computational Linguistics.

Sha Li, Heng Ji, and Jiawei Han. 2021. Document-level event argument extraction by conditional generation. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 894–908, Online. Association for Computational Linguistics.

Xiang Li, Thien Huu Nguyen, Kai Cao, and Ralph Grishman. 2015. Improving event detection with abstract meaning representation. In *Proceedings of the first workshop on computing news storylines*, pages 11–15.

Jian Liu, Yufeng Chen, and Jinan Xu. 2021. Machine reading comprehension as data augmentation: A case study on implicit event argument extraction. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 2716–2725, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Shulin Liu, Yubo Chen, Kang Liu, and Jun Zhao. 2017. Exploiting argument information to improve event detection via supervised attention mechanisms. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1789–1798, Vancouver, Canada. Association for Computational Linguistics.

Xiao Liu, Heyan Huang, Ge Shi, and Bo Wang. 2022. Dynamic prefix-tuning for generative template-based event extraction. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 5216–5228, Dublin, Ireland. Association for Computational Linguistics.

Xiao Liu, Zhunchen Luo, and Heyan Huang. 2018. Jointly multiple events extraction via attention-based graph information aggregation. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 1247–1256, Brussels, Belgium. Association for Computational Linguistics.

Xuebo Liu, Houtim Lai, Derek F. Wong, and Lidia S. Chao. 2020. Norm-based curriculum learning for neural machine translation. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 427–436, Online. Association for Computational Linguistics.

Ilya Loshchilov and Frank Hutter. 2018. Fixing weight decay regularization in adam.

Yaojie Lu, Hongyu Lin, Jin Xu, Xianpei Han, Jialong Tang, Annan Li, Le Sun, Meng Liao, and Shaoyi Chen. 2021. Text2Event: Controllable sequence-to-structure generation for end-to-end event extraction. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 2795–2806, Online. Association for Computational Linguistics.

Hans Peter Luhn. 1958. The automatic creation of literature abstracts. *IBM Journal of research and development*, 2(2):159–165.

Thang Luong, Hieu Pham, and Christopher D. Manning. 2015. Effective approaches to attention-based neural machine translation. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 1412–1421, Lisbon, Portugal. Association for Computational Linguistics.

Virginia McLean. 1992. Fourth message understanding conference (muc-4).

Thien Huu Nguyen, Kyunghyun Cho, and Ralph Grishman. 2016. Joint event extraction via recurrent neural networks. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 300–309, San Diego, California. Association for Computational Linguistics.

Thien Huu Nguyen and Ralph Grishman. 2015. Event detection and domain adaptation with convolutional neural networks. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 2: Short Papers)*, pages 365–371, Beijing, China. Association for Computational Linguistics.

Aaron van den Oord, Yazhe Li, and Oriol Vinyals. 2018. Representation learning with contrastive predictive coding. *arXiv preprint arXiv:1807.03748*.

Thomas F Shipley and Jeffrey M Zacks. 2008. *Understanding events: From perception to action*, volume 4. Oxford University Press.

Meihan Tong, Bin Xu, Shuai Wang, Yixin Cao, Lei Hou, Juanzi Li, and Jun Xie. 2020. Improving event detection via open-domain trigger knowledge. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 5887–5897, Online. Association for Computational Linguistics.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. *Advances in neural information processing systems*, 30.

David Wadden, Ulme Wennberg, Yi Luan, and Hannaneh Hajishirzi. 2019. Entity, relation, and event extraction with contextualized span representations. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the*

9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP), pages 5784–5789, Hong Kong, China. Association for Computational Linguistics.

Christopher Walker, Stephanie Strassel, Julie Medero, and Kazuaki Maeda. 2005. Ace 2005 multilingual training corpus-linguistic data consortium. *URL: https://catalog. ldc. upenn. edu/LDC2006T06*.

Lingxiao Wang, Jing Huang, Kevin Huang, Ziniu Hu, Guangtao Wang, and Quanquan Gu. 2019a. Improving neural language generation with spectrum control. In *International Conference on Learning Representations*.

Xiaozhi Wang, Xu Han, Zhiyuan Liu, Maosong Sun, and Peng Li. 2019b. Adversarial training for weakly supervised event detection. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 998–1008, Minneapolis, Minnesota. Association for Computational Linguistics.

Xiaozhi Wang, Ziqi Wang, Xu Han, Zhiyuan Liu, Juanzi Li, Peng Li, Maosong Sun, Jie Zhou, and Xiang Ren. 2019c. HMEAE: Hierarchical modular event argument extraction. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 5777–5783, Hong Kong, China. Association for Computational Linguistics.

Ziqi Wang, Xiaozhi Wang, Xu Han, Yankai Lin, Lei Hou, Zhiyuan Liu, Peng Li, Juanzi Li, and Jie Zhou. 2021. CLEVE: Contrastive Pre-training for Event Extraction. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 6283–6297, Online. Association for Computational Linguistics.

Runxin Xu, Tianyu Liu, Lei Li, and Baobao Chang. 2021. Document-level event extraction via heterogeneous graph-based interaction model with a tracker. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 3533–3546, Online. Association for Computational Linguistics.

Hang Yang, Yubo Chen, Kang Liu, Yang Xiao, and Jun Zhao. 2018. DCFEE: A document-level Chinese financial event extraction system based on automatically labeled training data. In *Proceedings of ACL 2018, System Demonstrations*, pages 50–55, Melbourne, Australia. Association for Computational Linguistics.

Hang Yang, Dianbo Sui, Yubo Chen, Kang Liu, Jun Zhao, and Taifeng Wang. 2021. Document-level event extraction via parallel prediction networks. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 6298–6308, Online. Association for Computational Linguistics.

Sen Yang, Dawei Feng, Linbo Qiao, Zhigang Kan, and Dongsheng Li. 2019. Exploring pre-trained language models for event extraction and generation. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 5284–5294, Florence, Italy. Association for Computational Linguistics.

Wojciech Zaremba, Ilya Sutskever, and Oriol Vinyals. 2014. Recurrent neural network regularization.

Shu Zhang, Dequan Zheng, Xinchen Hu, and Ming Yang. 2015. Bidirectional long short-term memory networks for relation classification. In *PACLIC*.

Zhisong Zhang, Xiang Kong, Zhengzhong Liu, Xuezhe Ma, and Eduard Hovy. 2020. A two-step approach for implicit event argument detection. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7479–7485, Online. Association for Computational Linguistics.

Shun Zheng, Wei Cao, Wei Xu, and Jiang Bian. 2019. Doc2EDAG: An end-to-end document-level framework for Chinese financial event extraction. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 337–346, Hong Kong, China. Association for Computational Linguistics.

COPNER: Contrastive Learning with Prompt Guiding for Few-shot Named Entity Recognition

Yucheng Huang[1] **Kai He**[1] **Yige Wang**[1] **Xianli Zhang**[1]
Tieliang Gong[1] **Rui Mao**[2] **Chen Li**[1*]
[1]School of Computer Science and Technology, Xi'an Jiaotong University
[2]School of Computer Science and Engineering, Nanyang Technological University
{huangyucheng, hk52025804, jihejue039, xlbryant}@stu.xjtu.edu.cn
{gongtl, cli}@xjtu.edu.cn, rui.mao@ntu.edu.sg

Abstract

Distance metric learning has become a popular solution for few-shot Named Entity Recognition (NER). The typical setup aims to learn a similarity metric for measuring the semantic similarity between test samples and referents, where each referent represents an entity class. The effect of this setup may, however, be compromised for two reasons. First, there is typically a limited optimization exerted on the representations of entity tokens after initing by pre-trained language models. Second, the referents may be far from representing corresponding entity classes due to the label scarcity in the few-shot setting. To address these challenges, we propose a novel approach named COntrastive learning with Prompt guiding for few-shot NER (COPNER). We introduce a novel prompt composed of class-specific words to COPNER to serve as 1) supervision signals for conducting contrastive learning to optimize token representations; 2) metric referents for distance-metric inference on test samples. Experimental results demonstrate that COPNER outperforms state-of-the-art models with a significant margin in most cases. Moreover, COPNER shows great potential in the zero-shot setting. The source code is available at: https://github.com/AndrewHYC/COPNER.

1 Introduction

As a fundamental task in Nature Language Process (NLP), Named Entity Recognition (NER) aims to identify the spans of text according to a pre-defined set of entity classes, such as person, organization and location (Sang and De Meulder, 2003). Many down-stream tasks heavily rely on these extracted entities, such as aspect-level sentiment (Mao and Li, 2021), intention recognition (Vedula et al., 2020), and knowledge graph construction (He et al., 2021). An enormous number of neural methods have shown promising ability on NER tasks (Chiu and Nichols, 2016; Yadav and Bethard, 2018; Li et al., 2020), whereas insufficient labeled data in different domains are still a significant challenging for the community. Considering obtaining full annotated data is labor-intensive and time-consuming, few-shot NER studies (Fritzler et al., 2019; Yang and Katiyar, 2020; Das et al., 2021; Cui et al., 2021; Ma et al., 2021) are raising more attention, which can alleviate annotation dependence and help neural methods transfer to other tasks easier.

Recently, Prompt Learning (PL) becomes a popular technology in NLP and shows great potential for dealing with few-shot issues (Liu et al., 2021b; Ding et al., 2021b; Chen et al., 2021; Mao et al., 2022). Typical prompt learning is designed for understanding sentence-level tasks by decoding a special marker of an input. However, it is challenging to adapt PL to token-level tasks, which needs to identify the class of each token. Inspired by PL, TemplateNER (Cui et al., 2021) applies manual templates for few-shot NER, which needs to enumerate all potential spans and forward propagate many times for each input, which is time-consuming.

Compared with TemplateNER, distance metirc-based approaches (Wiseman and Stratos, 2019; Yang and Katiyar, 2020; Ziyadi et al., 2020) are more popular and efficient in few-shot NER tasks. The key idea of this paradigm is to learn a similarity metric for measuring the semantic similarity between test samples and referents (e.g., prototypes or the nearest neighbors). The referents are usually derived from a few labeled samples through a pre-trained language model, e.g., BERT (Devlin et al., 2019). While being less costly, these methods are still limited in two aspects: Firstly, their ability to capture entity-class-related semantics is limited, because their main goal is to learn a suitable similarity function rather than optimizing the parameters of a sentence encoder for better entity representations. Secondly, as an anchored metric

*Corresponding author

Proceedings of the 29th International Conference on Computational Linguistics, pages 2515–2527
October 12–17, 2022.

referent is derived from only few labeled data, it is insufficient to properly represent the semantics of the corresponding entity class. This issue may severely impact the performance of a few-shot NER model in the inference phase.

To address these issues, we propose a novel approach named **CO**ntrastive learning with **P**rompt guiding for few-shot **NER** (COPNER). The core idea of COPNER is to leverage class-specific words (CWs) from natural language to serve as the agents of corresponding entity types. Specifically, the CWs are included by appending a prompt to the original input sentence. As shown in Figure 1(a), an original input sentence "`[BOS] Obama was born in 1961 [EOS]`" is concatenated with a prompt "`person date none`", where "`person`", "`date`" and "`none`" are the CWs for the entity classes of person, date, and non-entity, respectively. In the training phase, the representations of CWs are served as token-level supervision signals that guide the sentence encoder to pull the representations of tokens belonging to the same class, and also the representation of the anchored CW, to be closer. In this way, the sentence encoder can learn to capture the dependence between tokens for aligning the semantics of a mentioned entity with the semantics of the corresponding CW in the unified semantic space. In the inference phase, the representations of CWs are treated as metric referents for predicting entity classes. As the representations of CWs contain general and discriminates semantics at the scratch and are further trained to align with the corresponding entity set, they are more appropriate and stable than the referents derived from previous works. Further, we explore different methods of prompt construction, aiming to understand the effects of different forms of prompts.

We summarize existing few-shot NER research into three settings, including Cross-Label-Space, Domain Transfer, and In-Label-Space. COPNER is evaluated under all these settings. We conduct experiments on six NER datasets and COPNER largely outperforms state-of-the-art(SOTA) approaches in most cases, especially in complex scenarios. Specifically, compared with SOTA results in Few-NERD (INTRA) and Few-NERD (INTER), COPNER raises the F1 scores of 8.28% and 8.03% in these two fine-grained few-shot NER tasks. In Domain Transfer settings, COPNER also improves by 9.0% averaged F1 scores under 1-shot settings. Additionally, considering CWs can inherently carry the relevant category information, we explore the zero-shot ability of COPNER with satisfied results. The main contributions of this paper are summarized as follows:

- We propose a novel few-shot NER approach named COPNER, which combines contrastive learning and prompt guiding. By introducing prompts as supervision signals and metric referents, COPNER overcomes the problem that typical class referents cannot properly represent each category due to data scarcity, and enhances entity representations for better discrimination with contrastive learning.
- We detailly investigate the existing few-shot NER research and summarize them into three categories. COPNER is evaluated in all these categories and outperforms SOTAs in most cases. Further, we expand the boundaries of COPNER's ability to the zero-shot setting and also achieve satisfied results.

2 Task Formulation

NER is normally treated as a sequence labeling task. For each input sentence $X = \{x_1, x_2, ..., x_t\}$, NER models aim to assign each token x_i a label $y_i \in C$, where C is a predefined label set. The assigned label shows either the token is a part of a named entity or out of any entity classes.

For each entity class, few-shot NER tasks are only provided with very limited annotations as supervision for neural models. With a comprehensive survey, we summarize existing few-shot NER research into three settings, and COPNER is evaluated in all these settings to demonstrate its effectiveness and universality.

2.1 Cross-Label-Space Setting

For few-shot NER with Cross-Label-Space setting, there is a rich-resource dataset (source set $\mathbb{H}$) for training and a low-resource dataset (target set $\mathbb{L}$) for adapting and testing. Notably, the target label space $\mathcal{C}^L$ is totally different from the source label space $\mathcal{C}^H$, namely $\mathcal{C}^L \cap \mathcal{C}^H = \emptyset$. Usually, the Cross-Label-Space setting is combined with N-way K-shot setting, e.g., a support set consists of N entity classes and each class has K labeled examples. To deal with this setting, an NER model needs to be trained on $\mathbb{H}$ first, then adopts to a new label space using the support set of $\mathbb{L}$. It is challenging

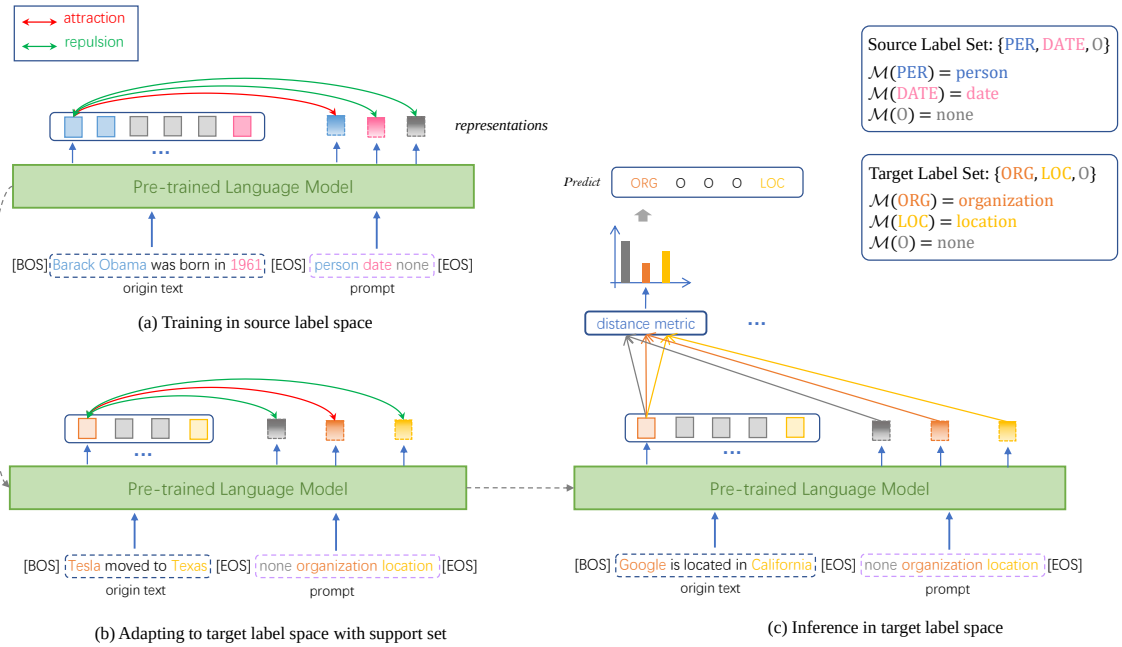

Figure 1: The illustration of the COPNER framework based on Contrastive Learning with Prompt guiding: (a) Training in the source label space {PER, DATE, O}. (b) Adapting to the target label space {ORG, LOC, O} with support set. (c) Inferring by comparing test tokens with class-specific words.

that COPNER needs to handle the problems of low resources and cross-label space.

2.2 Domain Transfer Setting

Similar to the Cross-Label-Space setting, there is also a rich-resource dataset (source set $\mathbb{H}$) for training and a low-resource dataset (target set $\mathbb{L}$) for adapting. The main difference is these two datasets come from different domains. For example, $\mathbb{H}$ can be news corpus while $\mathbb{L}$ comes from medical data. Besides, the label spaces of $\mathbb{H}$ and $\mathbb{L}$ can overlap. This setting needs COPNER to keep the domain transfer capability with limited annotations.

2.3 In-Label-Space Setting

Different from the previous two settings that have a rich-resource dataset, the In-Label-Space setting supposes that only a small number of labeled examples can be used for training. Specifically, a few-shot NER model is first trained on a dataset $\mathbb{D}^{train}$ with a label space $\mathcal{C}$, in which each entity class has only K samples. Then, the model is evaluated by a test set $\mathbb{D}^{test}$ with the same label space $\mathcal{C}$. It is a great challenge that COPNER needs to learn the NER task with only few training samples.

3 Methodology

The key idea of COPNER is to construct prompts with CWs for both the model training and inference. For each sentence, the COPNER concatenates a prompt to it and feeds the supplemented one into a pre-trained language model (PLM). Then, the PLM is trained in a contrastive learning fashion. In this process, the prompts play the role of the token-level anchors for constructing the positive pairs and negative pairs of contrastive learning. Finally, in the inference stage, the representations of CWs (in the prompt) are served as metric referents for predicting.

The whole process of COPNER is shown in Figure 1. We first train the used PLM in source label spaces. Next, the PLM is fine-tuned by a few support sets for adapting to new label spaces in evaluation tasks, e.g., Cross-Label-Space and Domain Transfer. Finally, a token-level distance-based metric classifier is employed to obtain the final results in the inference phase.

3.1 Prompt Guided Few-shot NER Model

First, our method constructs task-specific prompts for an employed dataset. A class-specific word mapping $\mathcal{M}$ is manually defined, then a unique class-specific word v_i is obtained for each en-

tity label $c_i \in C$ with $\mathcal{M}$, where C is the predefined label set. We use a simple yet effective method to develop $\mathcal{M}$, i.g. using class names of the given entities as CWs. For example, "LOC" is used as the label for location entities in most NER datasets, then the class-specific word "location" will be assigned to location entities following $\mathcal{M}(\texttt{LOC}) = \texttt{location}$. These CWs inherently contain the general semantic information of related entity classes and can avoid biases from limited labeled data.

In a specific few-shot task, COPNER maps each entity class $c_i \in C$ into a CWs v_i by $\mathcal{M}$ and concatenates these CWs to form a prompt. There are three ways of concatenating CWs in COPNER, which are described in Section 3.5.

Next, the generated prompt is appended to each input sentence X to form the extended input sequence $X' = \{x_1, x_2, \ldots, x_t, v_1, \ldots, v_n, v_{n+1}\}$[1], where t is the length of original input sentence and n is the number of entity class. Additionally, an extra class-specific word is added to denote non-entity class. Following, X' is fed into a PLM, e.g., BERT (Devlin et al., 2019), for generating contextualized representations. COPNER takes final hidden layer output as the representations of each token following:

$$\begin{aligned}\mathbf{H} &= [\mathbf{h}_1, ..., \mathbf{h}_t, \mathbf{h}'_1, ..., \mathbf{h}'_n] \\ &= \texttt{PLM}([x_1, .., x_t, v_1, ..., v_n])\end{aligned} \quad (1)$$

The representations of CWs in the prompt can be treated as guides for training the representations of tokens in the original sentence, which will detailed in the next section.

3.2 Training in Source Label Space

We employ an episode training strategy (Ding et al., 2021b) in COPNER, where a Greedy Sampling is adopted to randomly select an episode set $\mathcal{S}$ at each step. Specifically, the samples in an episode set $\mathcal{S}$ contain N entity classes (N way) with 1∼2 examples per class.

Centered with CWs in the prompt as class anchors, COPNER trains the used PLM to reduce representation distances between each token with its related CWs while pulling way with unrelated CWs. Specifically, for each extended input sequence $X' \in \mathcal{S}$, COPNER obtains the representation sequence $[\mathbf{h}_1, ..., \mathbf{h}_t, \mathbf{h}'_1, ..., \mathbf{h}'_{n+1}]$ by Eq. (1).

[1]For clarity, our formulates exclude the special marker "[BOS]" and "[EOS]". The detail utilization of these markers is shown in Fig 1.

Algorithm 1 Adapting process of COPNER

Input: $\mathcal{X}_{sup}$: support data; PLM: word encoder; γ: loss threshold;
Output: PLM

$\mathcal{L}_{prev} \in \mathbb{R}_+$ (arbitrary large);
$\mathcal{L}_{ft} = \mathcal{L}_{prev} - 1$;
repeat
  $\mathcal{L}_{prev} = \mathcal{L}_{ft}$;
  for all $(x_i, y_i) \in \mathcal{X}_{sup}$ **do**
    Calculate $l(x_i)$ as in Eq. (1) and Eq. (2);
  end for
  Calculate $\mathcal{L}_{ft}$ as in Eq. (3);
  Update PLM by back-propagation to reduce $\mathcal{L}_{ft}$;
until $(\mathcal{L}_{ft} > \mathcal{L}_{prev} \cap \mathcal{L}_{ft} < \gamma)$

Next, we construct positive and negative pairs of each X' for contrastive learning. Positive pairs are defined as (x_p, v_p), where x_p is the p_{th} token in X and v_p is the corresponding gold CW. Negative pairs are obtained by combining x_p and other unrelated CWs in the prompt.

Then, we can calculate the contrastive loss (Lin et al., 2021) with respect to x_p by:

$$\ell(x_p) = -\log \frac{\exp(-d(\mathbf{h}_p, \mathbf{h}'_p)/\tau)}{\sum_{q=1}^{n+1} \exp(-d(\mathbf{h}_p, \mathbf{h}'_q)/\tau)} \quad (2)$$

where τ denotes a temperature hyper-parameter proposed by Chen et al. (2020). As in previous metric-based works, we adopt the Euclidean distance as the similarity measure, which is calculated by: $d(\mathbf{h}_p, \mathbf{h}'_q) = ||\mathbf{h}_p - \mathbf{h}'_q||_2^2$. The total contrastive loss $\mathcal{L}$ of the episode $\mathcal{S}$ is calculate by:

$$\mathcal{L} = \frac{1}{|\mathcal{X}|} \sum_{x_i \in \mathcal{X}} \ell(x_i) \quad (3)$$

where $\mathcal{X}$ denotes the text token set in a sampled $\mathcal{S}$.

3.3 Adapting to Target Label Space

As mentioned in Section 2.1 and 2.2, when handling the settings of Cross-Label-Space and Domain Transfer, COPNER needs a certain extent of transferring ability. For such a reason, COPNER is fine-tuned with related support sets to adapt new label spaces after the training phase. This procedure is similar to the training stage, and the only difference is that the used data come from different label spaces or domains. Noticeably, such adapting process may make COPNER over-fit with the used adapting data, because these adapting data are usually with small numbers. Inspired by Das et al. (2021), we develop an early stopping criterion based on contrastive losses to alleviate the above problem. In particular, we add a hyper-parameter

γ as the loss threshold to prevent the model from not adapting enough or over-fitting. The complete adapting process with the early stopping criterion is illustrated in Algorithm 1.

3.4 Inferring from Metric Referents

In the inference phase, CWs in the prompt are regarded as metric referents to calculate distance with each token. We first obtain the representation of a extended test instance H^{test} following the rewritten Eq. (1) as

$$H^{test} = [\mathbf{h}_1, ..., \mathbf{h}_t, \mathbf{h}'_1, ..., \mathbf{h}'_n, \mathbf{h}'_{n+1}]. \quad (4)$$

For each token x_i, COPNER can find the nearest Metric Referent v_j in the PLM representation space, and the corresponding label c_j will be assigned with this token.

$$y_i^{test} = \underset{c_j}{\text{argmin}} ||\mathbf{h}_i - \mathbf{h}'_j||_2^2 \quad (5)$$

where $\mathbf{h}_i$ denotes each token representation and $\mathbf{h}'_j$ denotes each CW representation.

Alternatively, COPNER employs the Viterbi decoding algorithm. The used transition probabilities are calculated between three abstract NER tags (`O`, `I`, `I-Other`) on the training data. The emission probabilities are calculated by a `SoftMax` operation on the distance distributions between each test token and CWs during inference. These two probabilities are fed to a Viterbi decoder to obtain the final prediction. For more details, please refer to Structshot (Yang and Katiyar, 2020).

3.5 Prompt Construction

Liu et al. (2021b) shows that different forms of prompts have different effects on prompt-based approaches. For further explore these effects on COPNER, we propose three prompt construction methods. Figure 2 shows the examples for these methods and the details are introduced as:

- **Queue Prompt**: Directly combining the CWs in random order (the most intuitive way).
- **Partition Prompt**: Based Queue Prompt, extra special tokens "`[S]`" are used to separate each CWs. "`[S]`" only serves as a partition and does not have a specific meaning.
- **Continual Prompt**: This method employs continuous representations as special tokens to separate CWs. Similar with P-tuning (Liu et al., 2021b), the employed continuous representations are generated from an independent prompt encoder (two Bi-LSTM layers). By this way, we try to encode hidden associations between CWs into these separated markers.

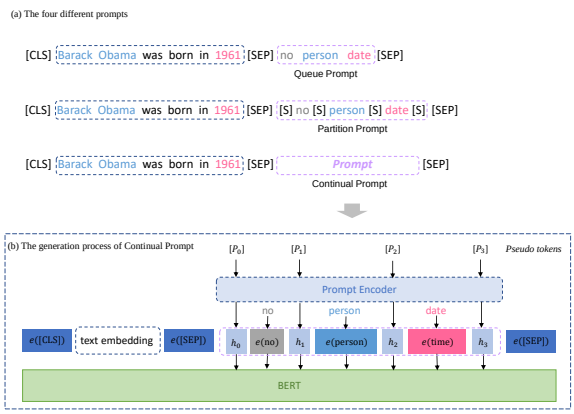

Figure 2: The illustration of prompt construction: (a) Three different forms of the prompt. (b) The process of generating continual prompts. A Prompt Encoder is employed to generate continual prompts, which is optimized with COPNER during training. $e(x)$ denotes the embedding of token x.

4 Experiment Setups

4.1 Datasets

COPNER is evaluated with six datasets, including OntoNotes 5.0 (Weischedel et al., 2012), WNUT'17 (Derczynski et al., 2017), I2B2'14 (Stubbs and Uzuner, 2015), CONLL'03 (Sang and De Meulder, 2003), MIT-Movie (Liu et al., 2013), and Few-NERD (Ding et al., 2021b). Among these datasets, the first five datasets come from the different domains, which correspond to the fields of general, social network, medical, newswire, and review, respectively. The last Few-NERD is the largest few-shot NER dataset, which contains INTRA and INTER two sub-settings, and a total of 66 fine-grained classes across 8 coarse-grained categories. Details of the datasets are shown in Table 6.

4.2 Baselines

(1) ProtoBERT is a popular few-shot method based on the prototypical network (Snell et al., 2017) with BERT (Devlin et al., 2019) as a backbone. **(2) NNShot** (Wiseman and Stratos, 2019) is a simple method based on token-level nearest neighbor classification. **(3) StructShot** (Yang and

Katiyar, 2020) adopts an additional Viterbi decoder based on NNShot. **(4) CONTaiNER** (Das et al., 2021) leverages contrastive learning to infer the distributional distance of Gaussian embeddings of entities. **(5) BERT-tagger** (Devlin et al., 2019) is a traditional BERT-based method which fine-tunes the BERT model with a label classifier. **(6) TemplateNER** (Cui et al., 2021) is a template-based approach, which enumerates all possible n-gram spans and classifies each of them. **(7) EntLM** (Ma et al., 2021) is a few-shot NER method which leverages an entity-oriented LM objective.

4.3 Evaluation on Three Settings

Cross-Label-Space Setting. For this setting, we evaluate COPNER with the Few-NERD dataset, which has two different tasks: Few-NERD (INTER) and Few-NERD (INTRA). For INTER, all the fine-grained entity classes are mutually disjoint in train, development, and test sets, while the coarse-grained categories are shared. For INTRA, the fine-grained entity classes in different sets belong to different coarse-grained categories. Few-NERD (INTRA) is more challenging due to the restrictions of sharing coarse-grained types. We evaluate COPNER 5000 episodes on the test set under each setting. As shown in Table 1 and Table 2, COPNER largely outperforms present related SOTAs in both tasks.

Domain Transfer Setting. This setting focuses on transferring an NER model to a new domain. Specifically, we train COPNER on the OntoNotes 5.0 dataset from the general domain and evaluate it on the test sets of CoNLL'03, WNUT'17, I2B2'14, which are from the newswire, social and medical fields, respectively. The support sets of the target domains are provided by Yang and Katiyar (2020). For each experiment, COPNER is fine-tuned on five support sets and the mean and standard deviation of F1 scores on the test set is reported. As shown in Table 3, COPNER also outperforms SOTAs in most cases, especially in the 1-shot setting.

In-Label-Space Setting. In this setting, we evaluate COPNER with two NER datasets from different domains: CoNLL'03 (Sang and De Meulder, 2003) and MIT-Movie (Liu et al., 2013). As introduced in Section 2.3, only K examples of each class are available for training. To explore the few-shot capability of COPNER with different size of training data, we try different K values from $\{5, 10, 20, 50\}$. For each K-shot values, COPNER is trained on three different sampled train sets provided by Ma et al. (2021), the mean and standard deviation of F1 scores on the test set is reported. As shown in Table 4, COPNER also outperforms state-of-the-art methods in most cases.

Table 1: F1 Scores (%) in Few-NERD (INTER). +Struct means using Viterbi Decoding. We color code each column as best and second best.

Model	5-way		10-way		Avg.
	1∼2 shot	5∼10 shot	1∼2 shot	5∼10 shot	
ProtoBERT	44.44	58.80	39.09	53.97	49.08
NNShot	54.29	50.56	46.98	50.00	50.46
StructShot	57.33	57.16	49.46	49.39	53.34
CONTaiNER	55.95	61.83	48.35	57.12	55.81
+Struct	56.10	61.90	48.36	57.13	55.87
COPNER	65.39	67.59	59.69	62.32	63.75
+Struct	65.98	67.70	59.56	62.37	63.90

Table 2: F1 Scores (%) in Few-NERD (INTRA). +Struct means using Viterbi Decoding. We color code each column as best and second best.

Model	5-way		10-way		Avg.
	1∼2 shot	5∼10 shot	1∼2 shot	5∼10 shot	
ProtoBERT	23.45	41.93	19.76	34.61	29.94
NNShot	31.01	35.74	21.88	27.67	29.08
StructShot	35.92	38.83	25.38	26.39	31.63
CONTaiNER	40.43	53.70	33.84	47.49	43.87
+Struct	40.40	53.71	33.82	47.51	43.86
COPNER	53.52	58.74	44.13	51.55	51.99
+Struct	54.26	58.84	44.26	51.18	52.14

5 Results and Discussion

In this section, we discuss the results of different few-shot NER settings and conduct experiments under the zero-shot setting. We also explore the effectiveness of different components of COPNER.

5.1 Overall Few-shot Results

The experimental results demonstrate that COPNER achieves a convincing improvement in all mentioned few-shot NER settings, and reaches SOTA performance to our best knowledge.

As shown in Table 1 and 2, COPNER outperforms the baselines by a large margin in the Cross-Label-Space setting. We observe a significant improvement in the 1-shot setting. A 1-shot sample may not give sufficient information about the target class distribution, which limits the performance of previous methods to a large extent. In contrast, CWs in COPNER carry class-related semantic information to ensure excellent performance.

As shown in Table 3, COPNER demonstrates strong domain transfer capability where support data are extremely limited. COPNER performs

Table 3: F1 scores (%) in Domain Transfer task. We report standard deviations from runs with five different support sets sampled by Yang and Katiyar (2020). +Struct means using Viterbi Decoding. We color code each column as best and second best.

Model	1 shot				5 shot			
	CoNLL	WNUT	I2B2	Avg.	CoNLL	WNUT	I2B2	Avg.
ProtoBERT	49.9±8.6	17.4±4.9	13.4±3.0	26.9	61.3±9.1	22.8±4.5	17.9±1.8	34.0
NNShot	61.2±10.4	22.7±7.4	15.3±1.6	33.1	74.1±2.3	27.3±5.4	22.0±1.5	41.1
StructShot	62.4±10.5	24.2±8.0	21.4±3.8	36.0	74.8±2.4	30.4±6.5	30.3±2.1	45.2
CONTaiNER	57.8±10.7	24.2±2.9	16.4±1.7	32.8	72.8±2.0	27.7±2.2	24.1±1.9	41.5
+Struct	61.2±10.7	27.5±1.9	21.5±1.7	36.7	75.8±2.7	32.5±3.8	36.7±2.1	48.3
COPNER	67.0±3.8	33.8±2.5	34.6±1.8	45.1	74.9±2.9	34.8±3.1	41.1±1.6	50.2
+Struct	66.5±2.1	34.9±1.8	35.8±1.3	45.7	74.6±3.1	34.2±2.6	43.7±1.5	50.8

Table 4: F1 scores (%) in In-Label-Space NER task. We report standard deviations from runs with three different support sets sampled by Ma et al. (2021). +Struct means using Viterbi Decoding. We color code each column as best and second best.

Model	CONLL					MIT-Movie				
	5 shot	10 shot	20 shot	50 shot	Avg.	5 shot	10 shot	20 shot	50 shot	Avg.
BERT-tagger	41.9±12.1	59.9±10.7	68.7±5.1	73.2±3.1	60.9	39.6±6.4	50.6±7.3	59.3±3.7	71.3±3.0	55.2
NNShot	42.3±8.9	59.2±11.7	66.9±6.1	72.6±3.4	60.3	39.0±5.5	50.5±6.1	58.9±3.5	71.2±2.9	54.9
StructShot	45.8±10.3	62.4±11.0	69.5±6.5	74.7±3.1	63.1	41.6±9.0	53.2±5.5	61.4±3.0	72.0±6.4	57.1
TemplateNER	43.0±6.2	57.9±5.7	66.4±6.1	72.7±2.1	60.0	46.0±3.9	49.3±3.4	59.1±0.4	65.1±0.2	54.9
EntLM	49.5±8.3	64.8±3.9	69.5±4.5	73.7±2.1	64.4	46.6±9.5	57.3±3.7	62.4±4.1	71.9±1.7	59.6
+Struct	51.3±7.7	66.9±3.0	71.2±3.9	74.8±1.9	66.1	49.2±8.9	59.2±4.0	63.9±3.7	73.0±1.8	61.3
COPNER	54.9±4.1	65.3±2.4	70.7±1.8	75.0±1.5	66.5	50.9±4.4	59.7±0.4	66.7±1.8	73.8±0.6	62.8
+Struct	54.2±7.9	66.2±2.9	71.8±1.8	77.0±1.4	67.3	50.1±3.6	61.9±1.4	68.9±2.4	74.6±0.3	63.9

significantly better than all previous methods, especially in the 1-shot setting. Specifically, COPNER raises of 5.8%, 7.4% and 14.3% F1 scores on the CoNLL, WNUT and I2B2 datasets, respectively.

As shown in Table 4, COPNER still achieves SOTA performance in most cases. In the In-Label-Space setting, the generalization ability of COPNER is examined by limiting the available training samples. Additionally, the standard deviations of F1 scores reported by COPNER are lower than those of other baselines, which indicates that our method is more stable than these baselines.

5.2 Zero-shot Learning

After trained on a rich-resource dataset, COPNER has learnt hidden contextual associations between CWs and tokens of input sentences. We aruge that these learnt contextual associations help COPNER to classify unseen entity categories even without any support data. Several experiments are conducted to demonstrate this idea. Specifically, for the Cross-Label-Space task and the Domain Transfer task, COPNER needs to make predictions on the target label spaces without any support set after training on the source label spaces. As shown in Table 5, COPNER can handle zero-shot NER tasks

Table 5: F1 Scores (%) in Zero-shot setting. +Struct means using Viterbi Decoding. We color code each column as best.

Model	Few-NERD				Domain Transfer		
	INTER		INTRA		CONLL	WNUT	I2B2
	5 way	10 way	5 way	10 way	-	-	-
COPNER	31.95	19.52	14.72	8.73	46.26	17.58	17.29
+Struct	33.97	20.92	16.06	9.64	49.39	17.41	17.47

with satisfied performance and the Viterbi decoding can further boost the performance. In the domain transfer tasks, COPNER under the zero-shot setting is even comparable to the prototypical network under the 1-shot setting.

5.3 Effect of Class-specific Words

Considering that the semantic information contained in class names can benefit entity encoding, the class names are selected as CWs introduced in Section 3.1. To demonstrate the effect of semantics of class names, we conduct experiments with the following variants of the CWs:

- Misleading words: We randomly swap the CWs between labels. For example, we assign "`location`" for "`PER`", "`person`" for "`ORG`", "`organization`" for "`LOC`".

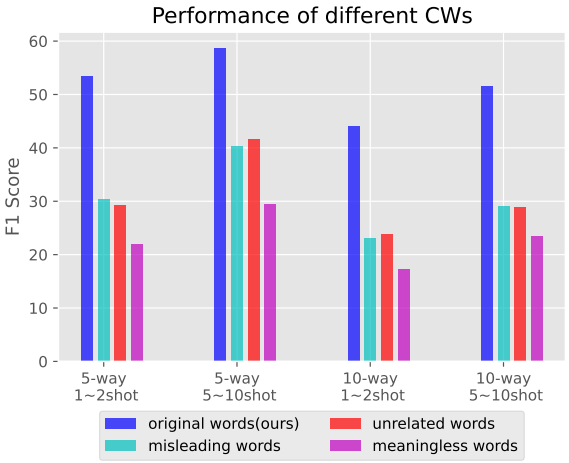

Figure 3: F1 Scores (%) in Few-NERD (INTRA) with the different variations of the CWs.

- Unrelated words: We randomly select some tokens from BERT's vocab as CWs, e.g., "`fully`", "`acoustic`" or "`new`", which are semantically unrelated to labels.

- Meaningless words: We use the tokens that are not used in BERT's vocab for CWs, such as "`[unused0]`", "`[unused1]`", etc. These tokens are semantically meaningless.

As shown in Figure 3, the performance of the three variants shows a significant decrease compared to the original words. The wrong semantic information in misleading words and unrelated words may mislead the entity representation learning leading to huge performance loss. It demonstrates that semantics matching the entity class is more effective as a class anchor. The further decrease in the performance of meaningless words further shows that semantic information is crucial in few-shot metric learning.

5.4 Influence of Prompts

In this subsection, we explore the influences of introducing prompts. In COPNER, prompts provide category-specific information during the representation calculation of tokens. To explore its impact, we conduct analytical experiments and the results are shown in Table 7 in Appendix. In addition to comparing the impact of three different prompt forms introduced in Section 3.5, we construct a baseline without adding any prompts: Fixed embedding Guiding (FG). More details are described in Appendix C. All three prompt-based methods are much better than FG, which indicates that the introduction of prompts can effectively improve the model capability. Both Partition Prompt and Continue Prompt achieve excellent performance, but the latter introduces additional parameters, so we use Partition Prompt in our main experiments.

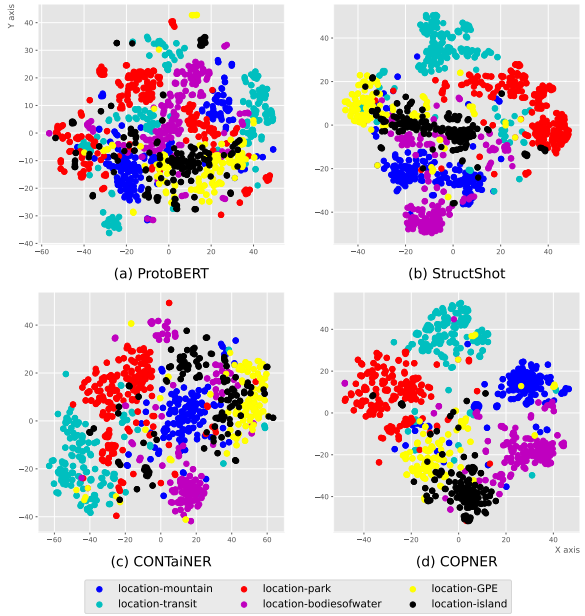

Figure 4: Two-dimensional t-SNE visualizations of the sampled 6 fine-grained classes from the `location` category. The embeddings are obtained from ProtoBERT, StructShot, CONTaiNER and COPNER, respectively.

5.5 Effectiveness Analysis

In this subsection, we summarize two main aspects to show the effectiveness of COPNER.

Enhanced Entity Representations. Figure 4 shows the two-dimension t-SNE visualization for the embeddings obtained from four different metric-based methods. Another one-dimension t-SNE visualization is shown as Figure 6 in Appendix D. As shown as these two Figures, COPNER results in better entity representations with greater differentiation of entity distributions across categories and more aggregation of similar entity distributions. More details are described in Appendix D.

Stable Metric Referents. We further investigate the metric results during inference. As shown in Figure 5(a), COPNER is more capable of distinguishing the positive pairs from the negative pairs with lower positive-negative distance ratios, which indicates that the nearest CW inference in COPNER has stronger category discrimination ability. COPNER is also the most stable and least influenced by support set differences, as shown in Figure 5(b). More details are described in Appendix E.

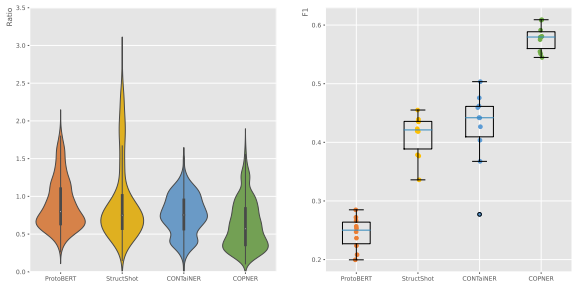

(a) positive-negative distance ratio (b) F1 scores with 10 different support sets

Figure 5: Effect of Metric Inference. (a) The ratio of the positive-pair distance to the mean negative-pair distance in inference of different models, (b) F1 scores of different models on 10 different support sets.

6 Related Work

Prompt Learning. GPT-3 (Brown et al., 2020) is the first try that used manual prompts in task prediction to improve performance. Then some ways to build templates manually were proposed (Petroni et al., 2019; Sanh et al., 2021). Automatic construction with different forms of templates was explored, including discrete templates (Shin et al., 2020; Jiang et al., 2020; Gao et al., 2020) and continuous prompts (Li and Liang, 2021; Liu et al., 2021b,a; Qin and Eisner, 2021; Han et al., 2021). A proper prompt can provide guidance information for downstream tasks. Therefore, the prompt is well suited for few-shot tasks where training data are scarce. A number of works introducing prompt learning to few-shot classification tasks have been proposed (Mao et al., 2018; Ding et al., 2021a; Chen et al., 2021; Madotto et al., 2021) in succession.

Few-shot NER. Generally, few-shot NER approaches can be categorized as metric-based and prompt-based. The former aims to calculate the similarity between test data and referents. Fritzler et al. (2019) applied the prototype network (Snell et al., 2017) to few-shot NER tasks. Inspired by the nearest neighbor inference (Wiseman and Stratos, 2019), Yang and Katiyar (2020) proposed Structshot, which used the Viterbi Decoder to capture label dependencies. Das et al. (2021) proposed CONTaiNER, which adopts Gaussian embeddings of tokens for the metric. Prompt-based methods leverage prompt learning to exploit the prior knowledge of pre-trained language models. Cui et al. (2021) proposed a time-consuming template-based BART for few-shot NER. Inspired by prompt tuning, Ma et al. (2021) proposed an entity-oriented method that fine-tuned the language model to predict class-related label words rather than the original words.

7 Conclusion

In this paper, we propose COPNER, a novel few-shot NER approach taking the advantage of contrastive learning and prompt guiding. COPNER achieves SOTA performance in few-shot NER settings by constructing prompts with CWs and exploiting the ability of contrastive learning to obtain enhanced representations and stable metric referents. COPNER can also handle zero-shot NER tasks. In the future, we will extend COPNER to more token-level few-shot classification tasks and further exploit its ability to handle zero-shot tasks.

Acknowledgements

This work has been supported by the Innovative Research Group of the National Natural Science Foundation of China (61721002); The Key Research and Development Program of Ningxia Hui Nationality Autonomous Region (2022BEG02025); The Key Research and Development Program of Shaanxi Province (2021GXLH-Z-095); The consulting research project of the Chinese Academy of Engineering (The Online and Offline Mixed Educational Service System for The Belt and Road Training in MOOC China); Project of China Knowledge Centre for Engineering Science and Technology; The innovation team from the Ministry of Education (IRT_17R86).

References

Tom B Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. 2020. Language models are few-shot learners. *arXiv preprint arXiv:2005.14165*.

Ting Chen, Simon Kornblith, Mohammad Norouzi, and Geoffrey Hinton. 2020. A simple framework for contrastive learning of visual representations. In *International conference on machine learning*, pages 1597–1607. PMLR.

Xiang Chen, Ningyu Zhang, Xin Xie, Shumin Deng, Yunzhi Yao, Chuanqi Tan, Fei Huang, Luo Si, and Huajun Chen. 2021. Knowprompt: Knowledge-aware prompt-tuning with synergistic optimization for relation extraction.

Jason PC Chiu and Eric Nichols. 2016. Named entity recognition with bidirectional lstm-cnns. *Transactions of the Association for Computational Linguistics*, 4:357–370.

Leyang Cui, Yu Wu, Jian Liu, Sen Yang, and Yue Zhang. 2021. Template-based named entity recognition using bart. *arXiv e-prints*, pages arXiv–2106.

Sarkar Snigdha Sarathi Das, Arzoo Katiyar, Rebecca J Passonneau, and Rui Zhang. 2021. Container: Few-shot named entity recognition via contrastive learning. *arXiv preprint arXiv:2109.07589*.

Leon Derczynski, Eric Nichols, Marieke van Erp, and Nut Limsopatham. 2017. Results of the wnut2017 shared task on novel and emerging entity recognition. In *Proceedings of the 3rd Workshop on Noisy User-generated Text*, pages 140–147.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. Bert: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186.

Ning Ding, Yulin Chen, Xu Han, Guangwei Xu, Pengjun Xie, Hai-Tao Zheng, Zhiyuan Liu, Juanzi Li, and Hong-Gee Kim. 2021a. Prompt-learning for fine-grained entity typing. *arXiv preprint arXiv:2108.10604*.

Ning Ding, Guangwei Xu, Yulin Chen, Xiaobin Wang, Xu Han, Pengjun Xie, Hai-Tao Zheng, and Zhiyuan Liu. 2021b. Few-nerd: A few-shot named entity recognition dataset. *arXiv preprint arXiv:2105.07464*.

Alexander Fritzler, Varvara Logacheva, and Maksim Kretov. 2019. Few-shot classification in named entity recognition task. In *Proceedings of the 34th ACM/SIGAPP Symposium on Applied Computing*, pages 993–1000.

Tianyu Gao, Adam Fisch, and Danqi Chen. 2020. Making pre-trained language models better few-shot learners. *arXiv preprint arXiv:2012.15723*.

Xu Han, Weilin Zhao, Ning Ding, Zhiyuan Liu, and Maosong Sun. 2021. Ptr: Prompt tuning with rules for text classification. *arXiv preprint arXiv:2105.11259*.

Kai He, Lixia Yao, JiaWei Zhang, Yufei Li, Chen Li, et al. 2021. Construction of genealogical knowledge graphs from obituaries: Multitask neural network extraction system. *Journal of Medical Internet Research*, 23(8):e25670.

Zhengbao Jiang, Frank F Xu, Jun Araki, and Graham Neubig. 2020. How can we know what language models know? *Transactions of the Association for Computational Linguistics*, 8:423–438.

Jing Li, Aixin Sun, Jianglei Han, and Chenliang Li. 2020. A survey on deep learning for named entity recognition. *IEEE Transactions on Knowledge and Data Engineering*, 34(1):50–70.

Xiang Lisa Li and Percy Liang. 2021. Prefix-tuning: Optimizing continuous prompts for generation. *arXiv preprint arXiv:2101.00190*.

Qika Lin, Jun Liu, Lingling Zhang, Yudai Pan, Xin Hu, Fangzhi Xu, and Hongwei Zeng. 2021. Contrastive graph representations for logical formulas embedding. *IEEE Transactions on Knowledge and Data Engineering*.

Jingjing Liu, Panupong Pasupat, Yining Wang, Scott Cyphers, and Jim Glass. 2013. Query understanding enhanced by hierarchical parsing structures. In *2013 IEEE Workshop on Automatic Speech Recognition and Understanding*, pages 72–77. IEEE.

Xiao Liu, Kaixuan Ji, Yicheng Fu, Zhengxiao Du, Zhilin Yang, and Jie Tang. 2021a. P-tuning v2: Prompt tuning can be comparable to fine-tuning universally across scales and tasks. *arXiv preprint arXiv:2110.07602*.

Xiao Liu, Yanan Zheng, Zhengxiao Du, Ming Ding, Yujie Qian, Zhilin Yang, and Jie Tang. 2021b. Gpt understands, too. *arXiv preprint arXiv:2103.10385*.

Ruotian Ma, Xin Zhou, Tao Gui, Yiding Tan, Qi Zhang, and Xuanjing Huang. 2021. Template-free prompt tuning for few-shot ner. *arXiv preprint arXiv:2109.13532*.

Andrea Madotto, Zhaojiang Lin, Genta Indra Winata, and Pascale Fung. 2021. Few-shot bot: Prompt-based learning for dialogue systems. *arXiv preprint arXiv:2110.08118*.

Rui Mao and Xiao Li. 2021. Bridging towers of multi-task learning with a gating mechanism for aspect-based sentiment analysis and sequential metaphor identification. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 35, pages 13534–13542.

Rui Mao, Xiao Li, Mengshi Ge, and Erik Cambria. 2022. MetaPro: A computational metaphor processing model for text pre-processing. *Information Fusion*, 86-87:30–43.

Rui Mao, Chenghua Lin, and Frank Guerin. 2018. Word embedding and WordNet based metaphor identification and interpretation. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1222–1231.

Fabio Petroni, Tim Rocktäschel, Sebastian Riedel, Patrick Lewis, Anton Bakhtin, Yuxiang Wu, and Alexander Miller. 2019. Language models as knowledge bases? In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 2463–2473.

Guanghui Qin and Jason Eisner. 2021. Learning how to ask: Querying lms with mixtures of soft prompts. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 5203–5212.

Erik Tjong Kim Sang and Fien De Meulder. 2003. Introduction to the conll-2003 shared task: Language-independent named entity recognition. In *Proceedings of the Seventh Conference on Natural Language Learning at HLT-NAACL 2003*, pages 142–147.

Victor Sanh, Albert Webson, Colin Raffel, Stephen H Bach, Lintang Sutawika, Zaid Alyafeai, Antoine Chaffin, Arnaud Stiegler, Teven Le Scao, Arun Raja, et al. 2021. Multitask prompted training enables zero-shot task generalization. *arXiv preprint arXiv:2110.08207*.

Taylor Shin, Yasaman Razeghi, Robert L Logan IV, Eric Wallace, and Sameer Singh. 2020. Eliciting knowledge from language models using automatically generated prompts. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 4222–4235.

Jake Snell, Kevin Swersky, and Richard Zemel. 2017. Prototypical networks for few-shot learning. In *Proceedings of the 31st International Conference on Neural Information Processing Systems*, pages 4080–4090.

Amber Stubbs and Özlem Uzuner. 2015. Annotating longitudinal clinical narratives for de-identification: The 2014 i2b2/uthealth corpus. *Journal of biomedical informatics*, 58:S20–S29.

Nikhita Vedula, Nedim Lipka, Pranav Maneriker, and Srinivasan Parthasarathy. 2020. Open intent extraction from natural language interactions. In *Proceedings of The Web Conference 2020*, pages 2009–2020.

Ralph Weischedel, Sameer Pradhan, Lance Ramshaw, Jeff Kaufman, Michelle Franchini, Mohammed El-Bachouti, Nianwen Xue, Martha Palmer, Jena D Hwang, Claire Bonial, et al. 2012. Ontonotes release 5.0.

Sam Wiseman and Karl Stratos. 2019. Label-agnostic sequence labeling by copying nearest neighbors. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 5363–5369.

Vikas Yadav and Steven Bethard. 2018. A survey on recent advances in named entity recognition from deep learning models. In *Proceedings of the 27th International Conference on Computational Linguistics*, pages 2145–2158.

Yi Yang and Arzoo Katiyar. 2020. Frustratingly simple few-shot named entity recognition with structured nearest neighbor learning. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 6365–6375.

Morteza Ziyadi, Yuting Sun, Abhishek Goswami, Jade Huang, and Weizhu Chen. 2020. Example-based named entity recognition. *arXiv e-prints*, pages arXiv–2008.

A Data statistics

In our experiments, we utilize a variety of different NER datasets to fully validate the capability of our proposed approach. A summary of these datasets is given in Table 6.

Table 6: Data statistics

Datasets	Domain	#Class	#Sent	#Entity
OntoNotes	General	18	76.7k	104.2k
WNUT'17	Social	6	5.7k	3.9k
I2B2'14	Medical	23	140.8k	29.2k
CoNLL'03	News	4	20.7k	35.1k
MIT-Movie	Review	12	12.2k	26.6k
Few-NERD	General	66	188.2k	491.7k

B Implementation Details

We use the "bert-base-uncased" pre-trained model as the word encoder in all of our experiments. For model training, the AdamW optimizer is employed with learning rate of 1e-4. We set batch size=16 and the loss temperature τ=0.05. As for prompt construction, we adopt Partition Prompt in our main experiments. The micro-F1 score is selected as the standard evaluation metric in all experiments.

Tagging Scheme For fair comparison, we adopt the IO tagging scheme following previous works, where I-type represents that all of the tokens are inside an entity, and O-type denotes all the other tokens.

C Effect of Prompts

In order to explore the effect of prompts, we add a baseline: Fixed embedding Guiding (FG). Specifically, we first obtain the embedding of each class-specific word from the last layer output of the "bert-base-uncased" pre-trained model. We then let these embeddings guide entity representations by contrastive learning. No prompt is expanded after the input text in FG and the embeddings of CWs are fixed during training. We conduct experiments on FG and the other three different prompts introduced in Section 3.5. The experimental results are shown in Table 7.

All the three prompt-based models outperform FG, indicating that prompts are effective in providing category-related information when models perform entity representation calculations. Despite their excellent results, three prompts have some different effects. The best model is Continual Prompt.

Table 7: F1 Scores(%) in Few-NERD (INTRA) with different prompts: QP, PP and CP mean to use Queue Prompt, Partition Prompt and Continual Prompt, respectively. And FG means to use the fixed embedding guiding. We color code each column as best and second best.

Model	5-way		10-way		Avg.
	1∼2 shot	5∼10 shot	1∼2 shot	5∼10 shot	
FG	43.55	51.85	37.49	48.79	45.42
QP	52.15	57.34	42.79	50.99	50.82
PP	53.52	58.74	44.13	51.55	51.99
CP	53.38	58.81	44.40	51.63	52.06

Whereas, it employs an extra Prompt Encoder to generate semantic linkage representations, which introduces additional training parameters. Partition Prompt achieves comparable performance to Continual Prompt, while introducing no additional parameters. This is the reason that we adopt it in our main experiments. The relatively poor performance of Queue Prompt indicates that putting the CWs together instead of separating them with some tokens has negative impacts on the model performance.

D t-SNE Visualization

From the Few-NERD (INTRA) test set, we randomly sample 100 examples from each of six fine-grained classes in the `location` category. Then, t-SNE is employed to project the entity representations obtained by the word encoder into two-dimensional and one-dimensional spaces. It is clear that the way of guiding entity representations by COPNER is more effective.

Figure 4 shows the two-dimensional visualization results of ProtoBERT, StructShot, CONTaiNER and COPNER, respectively. We can observe that: ProtoBERT has the weakest representation ability and fails to distinguish the entity representations of different classes; StructShot improves but tends to distribute the entity representations of the same class to multiple clusters; CONTaiNER further enhances entity representations, while it is still weak in some classes, such as location-bodiesofwater and location-island. COPNER performs the best and brings similar entities together as much as possible.

Figure 6 shows the one-dimensional visualization results of different entity classes. The higher aggregation of similar entity representations obtained from COPNER further visualizes the superiority of the way CWs guiding entity encoding.

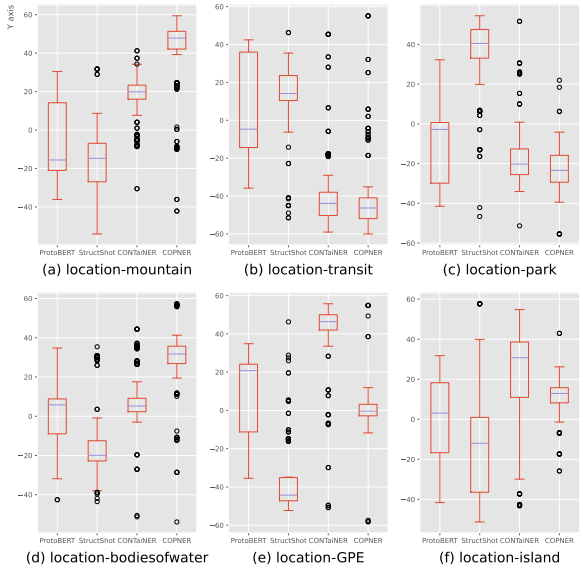

Figure 6: One-dimensional t-SNE visualizations of the sampled 6 fine-grained classes' examples from the `location` category. The 6 different classes are visualized separately.

E Effect of Metric Inference

In this sub-section, we investigate the effect of metric referents for different models during inference. We investigate the following two aspects: category discrimination ability and metric stability.

Category Discrimination Ability: In the distance-based inference, test examples are easily and correctly distinguished when they are closer to the positive metric referent and further away from the negative metric referents. We randomly sample a 1-shot support set containing six fine-grained classes of the location category from the Few-NERD (INTRA) test set, and then make predictions on the test examples sampled in Appendix D. Each test example forms a positive pair with the corresponding gold CW, and negative pairs with other CWs. We calculate the ratio of the positive-pair distance to the mean distance of negative pairs. As shown in Figure 5(a), COPNER is more capable of distinguishing the positive pair from the negative pairs with lower positive-negative distance ratios, which demonstrates that the nearest class-specific word inference in COPNER is more appropriate.

Metric Stability: In the distance-based metric approaches, the inference results of a good metric should be insensitive to different support sets. We randomly sampled 10 different 1-shot support sets from the Few-NERD (INTRA) test set. Each contains 6 fine-grained classes of the location category. Figure 5(b) demonstrates the prediction results of different models. The nearest class-specific word inference in COPNER is the most stable, while the nearest neighbor inference is sensitive to support set differences leading to large differences in prediction results.

F Class-specific Word Examples

Table 8 illustrates several CWs selected by COPNER, which are usually class names with class-specific semantics. For the classes with complex class names, we choose concise words with similar semantics.

Table 8: Several class-specific words selected by COPNER.

Few-NERD		**OntoNotes**	
#Class	#Class-specific word	#Class	#Class-specific word
location-bodiesofwater	water	ORG	organization
location-island	island	NORP	country
person-athlete	athlete	ORDINAL	number
person-director	director	WORK_OF_ART	art
organization-show	show	QUANTITY	quantity
organization-company	company	LAW	law
building-airport	airport	EVENT	event
building-hospital	hospital	CARDINAL	cardinal
art-painting	painting	LOC	location
art-film	film	FAC	facility
...	...	...	...
I2B2		**MIT-Movie**	
#Class	#Class-specific Word	#Class	#Class-specific Word
DATE	date	CHARACTER	character
PATIENT	patient	GENRE	genre
DOCTOR	doctor	TITLE	title
MEDICALRECORD	record	REVIEW	review
HOSPITAL	hospital	RATING	rating
IDNUM	id	YEAR	year
FAX	ip	ACTOR	actor
HEALTHPLAN	plan	DIRECTOR	director
ZIP	zip	SONG	song
STATE	state	RATINGS_AVG	score
...	...	...	...
WNUT		**CoNLL**	
#Class	#Class-specific Word	#Class	#Class-specific Word
location	location	ORG	organization
group	group	PER	person
corporation	company	LOC	location
person	person	MISC	miscellaneous
creative-work	creativity		
product	product		

Few Clean Instances Help Denoising Distant Supervision

Yufang Liu*[1], Ziyin Huang*[1], Yijun Wang[2],Changzhi Sun[3],
Man Lan[1], Yuanbin Wu[1], Xiaofeng Mou[4] and Ding Wang[4]
[1]School of Computer Science and Technology, East China Normal University
[2]Department of Computer Science and Engineering, Shanghai Jiao Tong University
[3]Bytedance AI Lab, [4]AI Innovation Center, Midea Group
{yfliu.antlp, zyhuang.cs}@gmail.com, ybwu@cs.ecnu.edu.cn

Abstract

Existing distantly supervised relation extractors usually rely on noisy data for both model training and evaluation, which may lead to garbage-in-garbage-out systems. To alleviate the problem, we study whether a small clean dataset could help improve the quality of distantly supervised models. We show that besides getting a more convincing evaluation of models, a small clean dataset also helps us to build more robust denoising models. Specifically, we propose a new criterion for clean instance selection based on influence functions. It collects sample-level evidence for recognizing good instances (which is more informative than loss-level evidence). We also propose a teacher-student mechanism for controlling purity of intermediate results when bootstrapping the clean set. The whole approach is model-agnostic and demonstrates strong performances on both denoising real (NYT) and synthetic noisy datasets.[1]

1 Introduction

Distant supervision was introduced to tackle the lacking training data problem in information extraction tasks (Mintz et al., 2009). By aligning relation triples in knowledge bases (KB) with free texts, it automatically builds labelled sentence instances and easily extends the scale of training set to hundreds of thousands samples. Due to this great scalability, distantly supervised entity relation extractors have been extensively studied in the past decade.

Like other weak signals, the major problem about these automatically generated datasets is label noise: not all aligned sentences carry the same semantic of a KB triple (e.g., not all sentences containing "Obama" and "United States" express a "born in" relation). Some applications (e.g., slot filling of the TAC KBP track (Ji and Grishman, 2011)) could be less affected with the help of *instance bags*, which only needs to seek one correct instance among a bag of aligned sentences. For a more general setting which aims to correctly detect relations on *individual sentences* (Miwa and Bansal, 2016; Sun et al., 2018; Wadden et al., 2019; Wang et al., 2020), however, the noisy labels make both learning and evaluation of models vulnerable: we may draw a flawed conclusion by using a dirty test set for a model learned with a dirty training set.

Many methods have been proposed to reduce noise labels (*denoise*) in distant supervision. For bag-level applications, models often rely on attention scores to either filter bad instances inside a bag (intra-bag attentions, Lin et al. 2016) or filter bags full of noisy instances (inter-bag attentions, Ye and Ling 2019). The dilemma there is that, while we expect attention scores to indicate correct labels, we have to train them to fit noisy labels since ground truth labels are noisy. The same difficulty also exists in recent instance-level denoising methods (Qin et al., 2018a,b) where the reward of denoising an instance is obtained by querying noisy labels. Therefore, not only the extraction models but also denoising models are questionable if only noisy labels are given.

In this paper, we would like to restate the importance of trustful data (*clean dataset*) in building large-scale information extraction systems. Specifically, if a *small* clean dataset ($\approx 10^2$ samples) is available, we ask whether the robustness of both the denoising model and final extraction model could be improved.

We start from training a relation classifier on the clean set and propose a new criterion to select good instances from the dirty set. The main idea is that if a testing instance is correctly labelled by distant supervision, some instances in the clean set should support it, and if we remove those support instances, prediction error of the testing instance

*These authors contributed equally.
[1]Our codes are publicly available at: https://github.com/Airuibadi/IF_DSRE.

Proceedings of the 29th International Conference on Computational Linguistics, pages 2528–2539
October 12–17, 2022.

will increase. Comparing with previous work, the criterion is based on perturbation analyses of classifiers instead of directly using output probabilities (scores) of classifiers. Our tool is *influence function* (IF; Cook and Weisberg 1982; Koh and Liang 2017) which can effectively approximate how a classifier's parameters change when removing a training point.

Next, to incrementally explore the dirty set, we compile our instance selection algorithm into a bootstrapping process: training a classifier on the current clean set, selecting new clean instances using the classifier and retraining the classifier on the updated clean set. The key challenge is how to control purity of those intermediate datasets: one noisy instance may bring more noisy instances. Existing works are either lack of such strategy, or use heuristic thresholds on classifiers or dataset size (Jia et al., 2019). Here, we propose a *teacher-student style* update for learning intermediate classifiers. It gradually controls the distance between the current model and history models by regularizing discrepancy of their predictions.

Our whole system could be deemed as a data preprocessing method. Comparing with in-model denoising method (e.g., attention scores), it outputs a new clean set which can be applied to any information extraction models (*model-agnostic*). We conduct experiments on both real distantly supervised datasets (NYT) and synthetic datasets (built on ACE05). The results demonstrate that besides effectively selecting good instances, the influence-function-based criterion can stratify noisy instances according their difficulties for prediction (or importances for a better extractor). We also find that the teacher-student update especially helps when the proportion of incorrectly labelled instances is large. Finally, when learned with clean sets built by our methods, we are able to achieve competitive extraction performances on *manually* labelled testing set.

2 Preliminary

Distantly Supervised Relation Classification Given an entity pair (e_h, e_t) and a sentence s containing the pair, we consider the task of determining whether the entity pair expresses certain relation $r \in R$, where R is the set of relation types (`None` indicates no relation). Denote $x = (s, e_h, e_t, r)$ to be an *instance*, $y \in \{0, 1\}$ to indicate whether x is positive or negative, and $D = \{(x_i, y_i)\}_{i=1}^{|D|}$ to be a set of labelled instances. For simplicity, we also define $z = (x, y)$.

In the distant supervision setting, instances in D are automatically obtained by aligning plain text and knowledge bases: for a KB triple (e_h, e_t, r), every sentence containing (e_h, e_t) is labelled with r. Obviously, D is a dirty set with both false positives (sentences don't match the semantic of r) and false negatives (sentences expressing relation r while been labelled with `None` due to incompleteness of KB). Here, we focus on false positives (much more serious in current datasets) and don't consider false negatives for its very low quantity. The denoising task is thus to find $D' \subset D$ containing correctly labelled instances (especially, positive instances).

Influence Function (Cook and Weisberg, 1982; Koh and Liang, 2017) provides a way to estimate how individual training instances influence a model. Typically, for a testing instance (x', y'), it efficiently answers the question that if a training instance (x, y) is removed how the model's prediction on (x', y') changes.

Denote $\mathcal{L}(z, \theta)$ to be a convex loss function of z with parameter θ, and $\hat{\theta} \triangleq \arg\min_\theta \frac{1}{n} \sum_{i=1}^{n} \mathcal{L}(z_i, \theta)$ to be the optimal model parameter learned on a training set (n is the set size). To study a training instance z's influence on $\hat{\theta}$, influence function considers an ϵ up-weight on z. Define $\hat{\theta}_{\epsilon,z} \triangleq \arg\min_\theta \frac{1}{n} \sum_{i=1}^{n} \mathcal{L}(z_i, \theta) + \epsilon \mathcal{L}(z, \theta)$. Therefore, when $\epsilon = -\frac{1}{n}$, $\hat{\theta}_{\epsilon,z}$ is the new model parameter after removing z from the training set.

The key idea of influence function is that, when ϵ is small (or training set size n is large), with the first order Taylor approximation, we can measure the difference between $\hat{\theta}$ and $\hat{\theta}_{\epsilon,z}$ without retraining the model,

$$\hat{\theta}_{\epsilon,z} - \hat{\theta} \approx -\epsilon H_{\hat{\theta}}^{-1} \nabla_\theta \mathcal{L}(z, \hat{\theta}) \triangleq \epsilon \mathcal{I}_{up,params}(z),$$

where $H_\theta = \frac{1}{n} \sum_{i=1}^{n} \nabla^2 \mathcal{L}(z_i, \theta)$ is the Hessian matrix of the original loss function. [2]

We can also get the change of the model's prediction on a testing instance z' by the chain rule,

$$\begin{aligned} &\mathcal{L}(z', \hat{\theta}_{\epsilon,z}) - \mathcal{L}(z', \hat{\theta}) \\ \approx &-\epsilon \nabla_\theta \mathcal{L}(z', \hat{\theta})^T H_{\hat{\theta}}^{-1} \nabla_\theta \mathcal{L}(z, \hat{\theta}) \triangleq \epsilon \mathcal{I}_{up,loss}(z, z'). \end{aligned}$$

We say z *supports* (or is *helpful* to) z' if removing z increases the testing loss of z', that is $\mathcal{S}(z, z') > 0$,

[2] We follow (Koh and Liang, 2017) using a stochastic estimation of H_θ^{-1} when computing influence functions.

where

$$\mathcal{S}(z, z') \triangleq -\frac{1}{n}\mathcal{I}_{up,loss}(z, z'). \quad (1)$$

We will see in the next section that the group of supporting instances is a key part in our denoising algorithms.

3 Utilities of Clean Sets

We study the task of picking out correct instances (D') from a distantly supervised dataset D (the dirty set). As discussed above, it is not easy for the denoising model to either correctly evaluate its results or receive the right learning signals if we only know noisy labels in D. Therefore, departing from previous works, we additionally require a small clean set C ($C \cap D = \emptyset$, $|D| \gg |C|$) which contains trustful annotations of instances (e.g., manually labelled). In our experiments, $|C| = 10$ is enough to bring significant improvement.

We build our denoising model based on a binary classifier $\hat{\theta}$, which aims to recognize truly labelled instances from D.[3] The classifier could be learned on D or C. For example, for an instance z, to test whether it is correctly labelled or not, a broadly applied principle is to query the classifier's confidence on predicting z's label: the lower loss $\mathcal{L}(z, \hat{\theta})$, the more possible z being correctly labelled.

Here, we go one step deeper: besides looking at the loss function, we could first seek high impact training samples on the classifier's drawing of $\mathcal{L}(z, \hat{\theta})$, and then collect evidence from them. For example, the more clean instances support z, the more possible z being correctly labelled. We are going to demonstrate that by probing the black-box classification process, we could build more explainable (yet effective) criteria for selecting instances.

First, from the computation of influence function, we can obtain a training instance's influence on a testing instances (Equation 1). Then, for a instance $z_d \in D$ (as discussed above, we mainly focus on positive $z_d = (x, y)$ where $y = 1$), we have two possible directions to derive a selection criterion.

- **Criterion 1**. We can train a classifier on D. z_d is correctly labelled if it supports $\hat{\theta}$'s prediction on the clean set C. Concretely, define $\mathcal{S}(z_d, \star) \triangleq \frac{1}{|C|}\sum_{z_c \in C}\mathcal{S}(z_d, z_c)$ to be the marginal $\mathcal{S}$ over the testing set,

$$\mathcal{S}(z_d, \star) > 0 \implies z_d \text{ is correct.} \quad (2)$$

- **Criterion 2**. We can also train a classifier on C. In this case, z_d is correctly labelled if $\hat{\theta}$'s prediction on z_d is supported by the instances in C. Define $\mathcal{S}(\diamond, z_d) \triangleq \frac{1}{|C|}\sum_{z_c \in C}\mathcal{S}(z_c, z_d)$ to be the marginal $\mathcal{S}$ over the training set,

$$\mathcal{S}(\diamond, z_d) > 0 \implies z_d \text{ is correct.} \quad (3)$$

Given the limited budget of clean instances C, the two methods are different in their way of using them. When taking C as the testing set (Criterion 1), we emphasize a valid feedback signal for evaluating the denoising model. On the other hand, when taking C as the training set (Criterion 2), we emphasize a clean learning signal for building the denoising model. We would like to discuss more on their pros and cons.[4]

For Criterion 1, as D is usually large enough, we could obtain a sufficiently learned classifier for denoising. More importantly, a large training set makes the estimation of influence function more reliable (Taylor expansion works on small ϵ). However, a good fitting of the dirty set could be a double-edged sword, especially when the proportion of wrongly labelled instances is large: we do get the influence function estimation right but it may not be applicable to our goal of denoising. We can first consider an ideal setting where all instances in D are true. In this case, Equation 2 is trustable since the ideal parameter $\hat{\theta}'$ is trustable, and it encodes the right information for detecting supporting relationship between the training and testing set. However, if a large part of D is false, the classifier $\hat{\theta}$ can diverge from the ideal $\hat{\theta}'$ severely, thus makes Equation 2 no longer true (e.g., a negative z_d could also satisfy the criterion as $\hat{\theta}$ is learned with noise).

Furthermore, we can have the following characterization of $|\mathcal{L}(z_d, \hat{\theta}') - \mathcal{L}(z_d, \hat{\theta})|$ if $L(z, \hat{\theta})$ is in the form of log-likelihood,

$$L(z, \hat{\theta}) = -\log p(y|x, \hat{\theta}) = -\log \frac{\exp\left(\hat{w}_y^\intercal h(x, \hat{\varphi})\right)}{Z}$$

[3] It is also possible to denoise by directly comparing similarities among instances (e.g., using patterns or sentence embeddings). While these methods are important, we mainly focus on classifier-based models whose settings are more analogous to semi-supervised learning or active learning. Comparing with them is beyond the scope of this paper.

[4] Similarly, we can also select the wrongly labelled instances by selecting the lowest influence function scores, we try to flip the labels and add them to the training set, but we find it barely working. The possible reason can be that these instances are positive instances for other entities or relations which adds too much noise for our classifier.

where $\hat{\theta} = [\hat{w}_0, \hat{w}_1, \hat{\varphi}]$, $\hat{w}_0, \hat{w}_1$ are class label embedding, $h(x, \hat{\varphi})$ is a learned representation of x (*encoder*), and Z is the normalizer.

Lemma 1. *Let $z = (x, y) \in D$, $z' = (x, y')$ be a relabelled z, and $\hat{\theta}_{z,z'}$ be the optimal model parameter after replacing z with z'. Denote τ_x to be the smallest singular value of $\nabla_\varphi h(x, \hat{\varphi})$. Then for any $z_d \in D$, up to $o(n^{-1})$, $|\mathcal{L}(z_d, \hat{\theta}_{z,z'}) - \mathcal{L}(z_d, \hat{\theta})|$ is lower bounded by*

$$\frac{c}{n}\left(\|h(x, \hat{\varphi})\| + \tau_x \|\hat{w}_y - \hat{w}_{y'}\|\right),$$

for some constant c. Proof is in Appendix A.

Therefore, if the classifier $\hat{\theta}$ fits well on the dirty set (in the sense of a large $\|\hat{w}_0 - \hat{w}_1\|$), $\mathcal{S}(z_d, z_c)$ calculated with $\hat{\theta}$ could be far away from its value being calculated with a clean training set (i.e., with $\hat{\theta}'$). For the case of multiple updates, since the group version of influence function may not faithfully reflect the change of parameters (Koh et al., 2019), we are not able to obtain similar results with Lemma 1. However, our empirical evaluations will show that performances of Criterion 1 is highly related to the proportion of clean instances in D.

For Criterion 2, comparing with training with dirty D, C contains trustful data, thus the implication relation in Equation 3 is clear after training on C. However, since the clean set is usually small, Criterion 2 takes the risk of under-fitting, which makes the prediction on $z_d \in D$ not sufficiently exploit structures of clean samples in C. Moreover, the estimation of influence function also becomes unstable on small datasets (i.e., ε is larger). In summary, instead of measuring a wrong $\mathcal{S}(z_d, z_c)$ with good accuracy (like Criterion 1), Criterion 2 may struggle with measuring the right $\mathcal{S}(z_c, z_d)$ with poor accuracy.

In the following section, we investigate bootstrapping methods to enlarge C incrementally. We hope that when the number of clean instances becomes larger, we could alleviate both under-fitting and poor estimation of influence function gradually.

4 Bootstrapping the Clean Set

Given a initial small clean set C_0 and a dirty set D_0,[5] our bootstrapping framework incrementally updates a denoising classifier $\hat{\theta}$. At iteration t, we first collect a fixed-size clean set $\tilde{C}$ by sampling from C_t. Second, a denoising classifier $\hat{\theta}$ is trained on the sampled set $\tilde{C}$, from which we can use influence-function-based scores (Equation 1) to evaluate each instance in D_t and choosing new clean instances D^c from D_t. Third, we update C_t and D_t by merging and excluding instances in D^c and retraining the denoising model again. As discussed above, how to control the purity of those intermediate clean sets is important (otherwise, we will face the same challenge as Criterion 1). We propose teacher-student style update for learning intermediate classifiers. It gradually controls the distance between the current model and history models by regularizing discrepancy of their predictions. We summarize the whole process in Algorithm 1. It is worth noting that the output of the bootstrapping process is a new clean set, on which we could build any relation classifier (i.e., *model-agnostic* denoising).

Denoising Classifier Our denoising model is a binary classification model. For each $x = (s, e_h, e_t, r)$, it predicts $y \in \{0, 1\}$. Here, we simply apply a softmax layer on a CNN encoder (the same setting of Lemma 1).[6] Specifically, $h(x, \varphi) = \text{CNN}(\mathbf{s}, \mathbf{p})$, where $\mathbf{s}$ contains embeddings of words in sentence s, and $\mathbf{p}$ contains position embeddings which indicates two entities e_h, e_t in the sentence (Zeng et al., 2014).

Sampling To obtain a fixed-size clean set $\tilde{C}$, we randomly sample instances from C_t with replacement. We keep $|\tilde{C}| = 200$ so that the influence function calculation is more efficient.

Fitting To fit the relation classifier on the sampled set $\tilde{C}$, our objective is to minimize

$$\hat{\theta} = \arg\min_\theta \sum_{z \in \tilde{C}} \mathcal{L}(z, \theta) \tag{4}$$

The parameters $\hat{\theta}$ are applied in calculating IF.

Evaluating After obtaining the parameters $\hat{\theta}$, to evaluate each instance $z_d \in D_t$, we define a score function by Criterion 2 as follows,

$$\mathcal{S}(\diamond, z_d) \triangleq \frac{1}{|\tilde{C}|} \sum_{z_c \in \tilde{C}} \mathcal{S}(z_d, z_c) \tag{5}$$

The score $\mathcal{S}(\diamond, z_d)$ is the average of clean training instances' influence on the test instance z_d.

[5] We use the subscript t to indicate the number of iterations. In some cases, we drop it for simplicity.

[6] The model could be any existing relation model. For simplicity, we select a simple CNN.

Algorithm 1 Bootstrapping Framework

Require: $C_0, D_0, t_{\max}, k$
Ensure: D^r
1: For the student model, initialize θ randomly
2: For the teacher model, initialize $\bar{\theta}$ with θ
3: **for** $t = 0 : t_{\max}$ **do**
4: Sample $\tilde{C}$ from C_t randomly
5: Fit θ on $\tilde{C}$ by Eq.9
6: Fit $\bar{\theta}$ by Eq. 10
7: Evaluate $\mathcal{S}(\diamond, z_d)\ \forall z_d \in D_t$ by Eq. 5
8: Select D^c by Eq. 6 and Eq. 7
9: Update C_t, D_t by Eq. 8
10: $D^r = C_{t_{\max}} \setminus C_0$

Selecting After obtaining the score for each instance $\mathcal{S}(\diamond, z_d)$, we can select the clean instances from D_t according $\mathcal{S}(\diamond, z_d) > 0$ (Criterion 2). In practice, we observe that adding a relaxation factor works better. Formally, we denote it as follows,

$$\tilde{D}_t^c = \{z_d \in D_t | \mathcal{S}(\diamond, z_d) + r > 0\} \quad (6)$$

where r is a positive number. In addition, we adopt a majority voting strategy: we consider not only the current iteration, but also the previous iterations to build the current cleaned set D^c, denoted as:

$$D^c = \left\{ z_d | \sum_{i=0}^{t} \mathbf{1}(z_d \in \tilde{D}_i^c) > k \right\} \quad (7)$$

where $\mathbf{1}(\cdot)$ is the indicator function and k is a hyper-parameter.

Updating Once we have the set D^c, we can update C_t and D_t with simple set operations, denoted as:

$$C_{t+1} = C_t \cup D^c, \quad D_{t+1} = D_t \setminus D^c \quad (8)$$

Teacher-student Mechanism Even though we have used an implicit majority voting to keep selected instances clean, affected by under-fitting and unstable estimation of influence function, error instances will inevitably enter the clean set. Considering our algorithm is based on bootstrapping, errors in previous rounds would have continuous impact on subsequent selection. As mentioned before, the model parameter θ is easily disturbed by wrong-label instances, with the error propagation, the θ would be rotten quickly. To avoid this case, we introduce a teacher-student mechanism (Tarvainen and Valpola, 2017).

Here,we deem θ as the student model, and use another set of model parameters $\bar{\theta}$ as the teacher model, In the fitting step, we add a consistency regularizer to Equation 4:

$$\hat{\theta} = \arg\min_{\theta} L(\tilde{C}, \theta) + \alpha \mathrm{KL}(q(*; \bar{\theta}) || p(*; \theta)) \quad (9)$$

where the q and p are outputs of teacher model and student model respectively, the KL-divergence provides a consistency loss and α is a hyper-parameter. Furthermore, the $\bar{\theta}$ would not be updated in Equation 9, we update it by exponentially moving average as commonly used in teacher-student method:

$$\bar{\theta}_t = \beta \bar{\theta}_{t-1} + (1 - \beta)\hat{\theta}_t \quad (10)$$

Teacher-student mechanism is seen as a regularization term of $\tilde{\theta}$ during fitting, and we neglect this term when calculating the influence function. After $t_{\max}$ times loop, we remove the seed set C_0 from the $C_{\max}$ and obtain our final result $D^r = C_{t_{\max}} \setminus C_0$. Then, we could train any model on D^r.

5 Experiment

5.1 Configurations

NYT The NYT dataset is a widely-used distant supervision benchmark, which is built by Riedel et al. (2010) and rearranged by Jia et al. (2019). The training set is annotated with distant supervision while both development set and test set are manually annotated. For this dataset, We set D to be the NYT training set, and C as to be the NYT development set. [7]

ACE05-N The ACE05-N dataset is a synthetic noisy dataset which adapted from ACE05 (Walker et al., 2006), a commonly used instance-level supervised dataset. We first add the same amount of negative instances (with `None` relation label) as the annotated instances, and then mix additional noisy instances with different ratio, which are flipped `None` instances. Detailed dataset specification could be found in the supplementary.

Settings The settings and implementation details are in Appendix B. We evaluate Precision, Recall, and F1 with micro-averaging in instance-level.

[7] Noting that there is no instance leakage in the following evaluation on development set: we have remove it from our obtained clean set (line 10 of Algorithm 1).

Encoder	Method	Dev			Test		
		Prec.	Rec.	F1	Prec.	Rec.	F1
CNN	RL	42.50	**71.62**	53.34	43.70	72.34	54.49
	Conf	**83.41**	56.03	67.03	58.09	58.09	67.75
	Cr1	81.33	43.94	57.06	73.49	43.62	54.75
	Cr2	76.82	61.54	68.34	**79.71**	60.48	**68.78**
	Cr2TS	76.80	62.10	**68.69**	75.36	**60.52**	67.13
PCNN	ATT	68.09	47.49	55.95	67.31	49.83	57.27
	Conf	82.18	57.23	67.47	**80.15**	58.48	67.62
	Cr1	78.38	47.64	59.26	76.63	52.19	56.85
	Cr2	75.94	**62.87**	68.78	78.71	**61.86**	**69.27**
	Cr2TS	**79.34**	61.14	**69.06**	79.60	59.62	68.17
BiLSTM	ARNOR	78.14*	59.82*	**67.77***	**79.70***	**62.30***	**69.93***
	Conf	80.37	55.82	65.88	79.46	56.43	65.99
	Cr1	**80.73**	55.28	62.06	69.28	54.16	60.79
	Cr2	72.39	**60.67**	66.01	72.04	61.85	66.56
	Cr2TS	77.38	60.00	67.59	74.90	58.65	65.78

Table 1: Comparison of our method and other baselines with different encoders. We denote Conf, Cr1, Cr2, Cr2TS as Confidence, Criterion 1, Criterion 2 and Criterion 2 with teacher-student update style. The code of ARNOR is not accessible now and we find it is hard to reproduce the reported performances.

5.2 Baselines

ATT (Lin et al., 2016) is a classical bag-level denoising method which tunes the attention weight of each instance in bags during training to alleviate the impact of noisy instances.

RL (Qin et al., 2018b) introduces reinforcement learning method to train a instance selector that could tell the noisy instances from the distant supervised training set.

ARNOR (Jia et al., 2019) embeds the relation pattern attention based on recurrent neural network into a bootstrapping framework.

Confidence We implement another baseline for fair comparison, which uses the trained model parameters $\hat{\theta}$ to select instances by confidence each iteration instead of influence function criteria. As a control, it also starts from a initial clean seed set.

5.3 Main Result

Table 1 lists overall performances on NYT dataset with different relation classification models (recall that our approach is model-agnostic). We compare the results of our method with several baselines. From the results, we find that,

- Comparing with prior methods, both Cr2 and Cr2TS achieve better or comparable performance with different encoders, which suggests that our model-agnostic method could effectively prevent RE model from noise data.
- Both Conf and Cr2 use the dev data as a reference, while Cr2 achieves superior performance. Thus, our method is a better strategy which makes use of limited clean set. We credit it to that the influence function could select better instances under the criterion 2.
- The results show that Cr1 performs much worse than Cr2 on both dev and test set. As we mentioned before, Cr1 uses dirty set as training set, leading to unreliable influence.
- The performance of Cr2TS is worse than that of Cr2, which shows the teacher-student mechanism has no advantage on this dataset. We guess it's relative to the noise ratio on dirty set, and further discuss in next section.

6 Analysis and Discussion

Validating influence function. The calculation of influence function is the key step of our method. Here we show the high correlation between the real influence (calculated by leave-one-out retraining) and estimated influence. From the experimental results (see Appendix E), we find that the correlation among high influential instances is 0.79, and 0.65 in all instances. The high correlation validate that influence function is reliable in perform instance perturbation analyses.

Bootstrapping process in detail. In this section, we study the performance change of four selecting strategies during the bootstrapping procedure, as show in Figure 1.

- The performance curves of four strategies are quite similar, which gradually rise to the peak at the beginning and then fall to the line of original noisy data. We think the main reason for this phenomenon is that these strategies add more clean instances into D_l^c in the early epochs, and inevitably select more and more noisy instances in the later training epochs.
- The curve of Cr2 and Cr2TS is higher than Conf, which suggests that the effectiveness of criterion 2. As expected, Cr1 fails in the later period, which is even inferior to training with original noisy data.

The impact of noise ratio. We conduct experiments with different ratio of noise data to verify the denoising ability of our method (Figure 2).

- Even the noise ratio is extreme high (90%), the Cr2 and Cr2TS is still stable. We think that

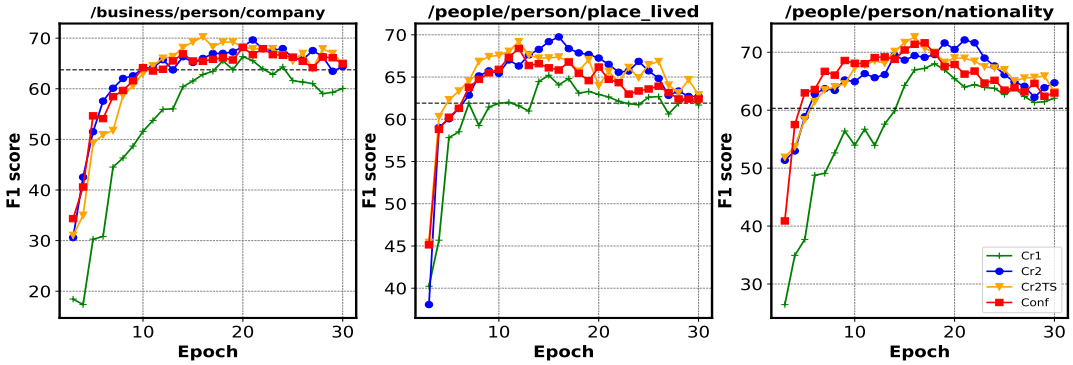

Figure 1: Bootstrapping result. Here we take 3 NYT relations as examples, we present performance change on dev during the bootstrapping. The black dash lines are the performance without denoising.

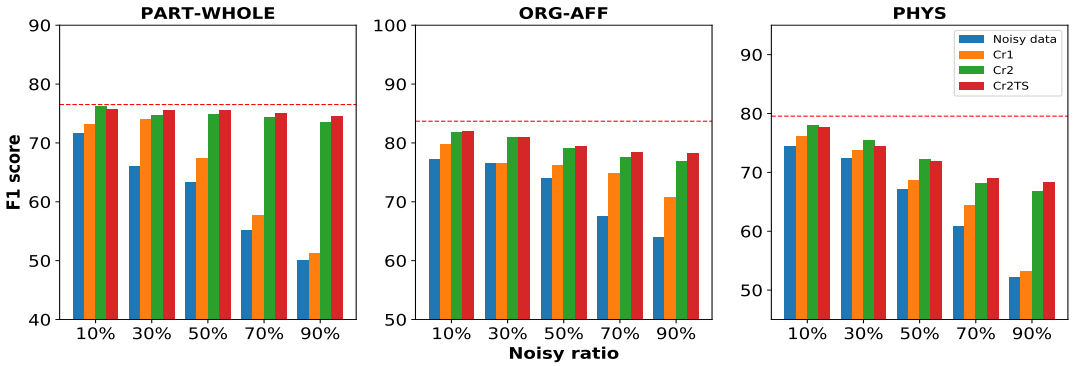

Figure 2: The denoising ability of our method with different noise ratio. Here we take three relation types from ACE05-N as examples. We evaluate their performances with the different ratio of noisy instances that range from 10% to 90% for each relation type. The red dash line indicates the performance without any noisy.

our methods take the most of the clean set to distinguish clean instances from the dirty set, so the damage from the noise instances in dirty set is quite slight.

- The `Cr1` would crash when the noise ratio beyond a certain level, as we analysis before, the higher noise ratio the train set has, the wider gap between estimated influence and the real influence of noisy instances.
- It is worth noting that `Cr2TS` would be better than `Cr2` in the case of the high noise ratio, which shows the effectiveness of teacher-student mechanism. The teacher-student mechanism has the advantage to help lower the lasting impact of misclassified instances in the previous iteration.

The impact of initial clean set size. Our method starts with an initial clean set, so we study the impact of the set size in Table 2.

- In general, the performance goes down with the number of clean instances decreasing. That is reasonable for that the limited clean seed set would suppress the methods to find more true positive instances.
- The performance of `Conf` drops sharply with few initial clean instances (10 instances). We guess that the method only considering confidence of instances is easily trapped into the limited clean set and hard to detect more clean instances.
- Both `Cr2` and `Cr2TS` show better robustness even the the size is extreme small. We believe that the key factor is the influence func-

Methods	10	30	50	ALL
Conf	31.43	34.28	35.89	36.95
Cr1	27.93	28.22	28.63	30.73
Cr2	35.1	36.23	37.42	37.87
Cr2TS	36.52	37.82	38.04	38.69

Table 2: We conduct our experiment on ACE05-N with 50% noise ratio to study the impact of initial clean set size. Note that the F1-score is 27.50 without any selecting strategy. Columns represents the result of using different number of instances as initial clean set. For each relation, we try to use 10, 30, 50 and all dev set as the intial clean set.

Reference:Jeffrey Katzenberg, chief executive of DreamWorks Animation, said...			
	Sentence	**TP/FP**	**Score**
Obvious TP	...Richard C.Notebaert, the cheif excutive of Qwest...	TP	3.46e-3
	...and Bruce Wassertein,the chairman and cheif excutive of Lazard	TP	1.73e-3
Hard instances	...last October, Ray Ozzie, chief technical officer, who joined Microsoft last year ...	TP	3.39e-5
	...Richard C.Noteaert, the company's chief executive, said Qwest spent..	FP	4.91e-5
Potential FP	Eric Foner is the De Witt Clinton professor of history at Columbia University and the author...	TP	-2.36e-3
	As Bruce Wasserstein left St. Regis Hotel in Manhattan on Tuesday afternoon after presenting Lazard's plan ...	FP	-4.37e-3

Table 3: An example of layered phenomenon of instances in the noisy dataset. We group instances by their scores calculated by `Cr2`.

tion, which considers more than confidence and is more practical to extend the scale of clean instances from a small start.

Stratification of instances. Table 3 presents a stratification of instances in the noisy dataset, which is our source of inspiration. There are three layers sorted by the score with Criterion 2.[8] The first layer contains instances with large positive score, which usually have a similar syntactic and semantic structure with the reference instance. These instances are true positive instances, and our method select them in every iteration. The second layer is made up of instances with score around zero. These instances are usually hard to tell whether they are noisy or not. The true positive instances in this layer could be discovered by extending clean set with bootstrapping. The last layer is formed by instances with large negative scores, which are quite different from the reference instance. Some of these instances are indeed noise, while some are still true positive instances but just not be supported by this reference instance. These true positive instances would be supported by other reference instances in clean set which selected by the average score in Criterion 2.

[8]We just take one reference instance as example, rather than the average of all reference instances in criterion 2.

7 Related Work

We focus on distant supervision relation extraction via influence function in this paper. For relation extraction, various neural networks like CNN (Zeng et al., 2014, 2015), RNN (Zhang et al., 2015) and Tree-GRU (He et al., 2018). Distant supervision provides a method to automatically label massive training data (Mintz et al., 2009), meanwhile, bringing excessive wrong label instances, so called noise, which stems the training.

To solve the noisy problem, people first take a multi-instance learning methods (Surdeanu et al., 2012; Lin et al., 2016; Ye and Ling, 2019), which puts instances with same entity pair into bags, to alleviate the impact of noisy instances. Then, to make training process closer to real-world application, people focus on instance-level denoising method. An instance-selector is utilized to pick out trustable instance, which is trained by reinforcement learning (Qin et al., 2018b; Feng et al., 2018) and adversarial learning (Qin et al., 2018a). The bootstrapping framework (Jia et al., 2019; Li et al., 2020) is also utilized to promote the ability of classification model gradually from a small seed. For influence function, which is commonly used in robust statistics (Cook and Weisberg, 1982), Koh and Liang (2017) introduce it in machine learning area. As a technology that is aiming to analyse the every training points' influence on model prediction, influence function is widely applied. Ren et al. (2020) apply influence function on weighting unlabeled data to promote semi-supervised learning. Xu and Kazantsev (2019) utilize influence function to designed an efficient strategy for active learning.

8 Conclusion

In this paper, we propose a model-agnostic denoise method for distant supervision relation extraction. We start from training a relation classifier on the clean set and propose a new criterion to select good instances from the noisy data. We leverage the criterion in a bootstrapping learning to extent the clean set iteratively. Further, we propose a teacher-student to control the update. Our method has a strong performance on NYT dataset and shows robustness under the high noise ratio circumstance or very limited size of initial clean set on the synthetic ACE05-N dataset.

9 Acknowledgments

We thank all anonymous reviewers for their constructive and helpful feedback. The corresponding author is Yuanbin Wu. This research was supported by NSFC (62076097) and a joint research fund from AI Innovation Center, Midea Group.

References

R. Dennis Cook and Sanford Weisberg. 1982. *Residuals and Influence in Regression*. New York: Chapman and Hall.

Jun Feng, Minlie Huang, Li Zhao, Yang Yang, and Xiaoyan Zhu. 2018. Reinforcement learning for relation classification from noisy data. pages 5779–5786.

Zhengqiu He, Wenliang Chen, Zhenghua Li, Meishan Zhang, Wei Zhang, and Min Zhang. 2018. See: Syntax-aware entity embedding for neural relation extraction.

Heng Ji and Ralph Grishman. 2011. Knowledge base population: Successful approaches and challenges. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, pages 1148–1158, Portland, Oregon, USA. Association for Computational Linguistics.

Wei Jia, Dai Dai, Xinyan Xiao, and Hua Wu. 2019. ARNOR: Attention regularization based noise reduction for distant supervision relation classification. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 1399–1408, Florence, Italy. Association for Computational Linguistics.

Pang Wei Koh, Kai-Siang Ang, Hubert H. K. Teo, and Percy Liang. 2019. On the accuracy of influence functions for measuring group effects. In *Advances in Neural Information Processing Systems 32: Annual Conference on Neural Information Processing Systems 2019, NeurIPS 2019, December 8-14, 2019, Vancouver, BC, Canada*, pages 5255–5265.

Pang Wei Koh and Percy Liang. 2017. Understanding black-box predictions via influence functions. In *Proceedings of the 34th International Conference on Machine Learning, ICML 2017, Sydney, NSW, Australia, 6-11 August 2017*, volume 70 of *Proceedings of Machine Learning Research*, pages 1885–1894. PMLR.

Zhenzhen Li, Jian-Yun Nie, Benyou Wang, Pan Du, Yuhan Zhang, Lixin Zou, and Dongsheng Li. 2020. Meta-learning for neural relation classification with distant supervision. *Proceedings of the 29th ACM International Conference on Information and Knowledge Management*.

Yankai Lin, Shiqi Shen, Zhiyuan Liu, Huanbo Luan, and Maosong Sun. 2016. Neural relation extraction with selective attention over instances. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 2124–2133, Berlin, Germany. Association for Computational Linguistics.

Mike Mintz, Steven Bills, Rion Snow, and Daniel Jurafsky. 2009. Distant supervision for relation extraction without labeled data. In *Proceedings of the Joint Conference of the 47th Annual Meeting of the ACL and the 4th International Joint Conference on Natural Language Processing of the AFNLP*, pages 1003–1011, Suntec, Singapore. Association for Computational Linguistics.

Makoto Miwa and Mohit Bansal. 2016. End-to-end relation extraction using LSTMs on sequences and tree structures. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1105–1116, Berlin, Germany. Association for Computational Linguistics.

Jeffrey Pennington, Richard Socher, and Christopher D. Manning. 2014. Glove: Global vectors for word representation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing, EMNLP 2014, October 25-29, 2014, Doha, Qatar, A meeting of SIGDAT, a Special Interest Group of the ACL*, pages 1532–1543. ACL.

Pengda Qin, Weiran Xu, and William Yang Wang. 2018a. DSGAN: Generative adversarial training for distant supervision relation extraction. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 496–505, Melbourne, Australia. Association for Computational Linguistics.

Pengda Qin, Weiran Xu, and William Yang Wang. 2018b. Robust distant supervision relation extraction via deep reinforcement learning. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 2137–2147, Melbourne, Australia. Association for Computational Linguistics.

Zhongzheng Ren, Raymond A. Yeh, and Alexander G. Schwing. 2020. Not all unlabeled data are equal: Learning to weight data in semi-supervised learning.

Sebastian Riedel, Limin Yao, and Andrew McCallum. 2010. Modeling relations and their mentions without labeled text. In *Machine Learning and Knowledge Discovery in Databases, European Conference, ECML PKDD 2010, Barcelona, Spain, September 20-24, 2010, Proceedings, Part III*, volume 6323 of *Lecture Notes in Computer Science*, pages 148–163. Springer.

Changzhi Sun, Yuanbin Wu, Man Lan, Shiliang Sun, Wenting Wang, Kuang-Chih Lee, and Kewen Wu. 2018. Extracting entities and relations with joint minimum risk training. In *Proceedings of the 2018*

Conference on Empirical Methods in Natural Language Processing, pages 2256–2265, Brussels, Belgium. Association for Computational Linguistics.

Mihai Surdeanu, Julie Tibshirani, Ramesh Nallapati, and Christopher D. Manning. 2012. Multi-instance multi-label learning for relation extraction. In *Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning*, pages 455–465, Jeju Island, Korea. Association for Computational Linguistics.

Antti Tarvainen and Harri Valpola. 2017. Mean teachers are better role models: Weight-averaged consistency targets improve semi-supervised deep learning results. In *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, December 4-9, 2017, Long Beach, CA, USA*, pages 1195–1204.

David Wadden, Ulme Wennberg, Yi Luan, and Hannaneh Hajishirzi. 2019. Entity, relation, and event extraction with contextualized span representations. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 5784–5789, Hong Kong, China. Association for Computational Linguistics.

Christopher Walker, Stephanie Strassel, Julie Medero, and Kazuaki Maeda. 2006. Ace 2005 multilingual training corpus. In *In Linguistic Data Consortium*.

Yijun Wang, Changzhi Sun, Yuanbin Wu, Junchi Yan, Peng Gao, and Guotong Xie. 2020. Pre-training entity relation encoder with intra-span and inter-span information. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1692–1705, Online. Association for Computational Linguistics.

Minjie Xu and Gary Kazantsev. 2019. Understanding goal-oriented active learning via influence functions.

Zhi-Xiu Ye and Zhen-Hua Ling. 2019. Distant supervision relation extraction with intra-bag and inter-bag attentions. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 2810–2819, Minneapolis, Minnesota. Association for Computational Linguistics.

Daojian Zeng, Kang Liu, Yubo Chen, and Jun Zhao. 2015. Distant supervision for relation extraction via piecewise convolutional neural networks. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 1753–1762, Lisbon, Portugal. Association for Computational Linguistics.

Daojian Zeng, Kang Liu, Siwei Lai, Guangyou Zhou, and Jun Zhao. 2014. Relation classification via convolutional deep neural network. In *Proceedings of COLING 2014, the 25th International Conference on Computational Linguistics: Technical Papers*, pages 2335–2344, Dublin, Ireland. Dublin City University and Association for Computational Linguistics.

Shu Zhang, Dequan Zheng, Xinchen Hu, and Ming Yang. 2015. Bidirectional long short-term memory networks for relation classification. In *Proceedings of the 29th Pacific Asia Conference on Language, Information and Computation*, pages 73–78, Shanghai, China.

Supplementary Materials for
Few Clean Instances Help Denoising Distant Supervision

A Proof of Lemma 1

Proof. Following the same first order Taylor approximation as influence function, up to $o(n^{-1})$, we can get the update of parameters after replacing z with z' (see Equation 3 of (Koh and Liang, 2017)),

$$\begin{aligned}\hat{\theta}_{z,z'} - \hat{\theta} &= n^{-1}\left(\mathcal{I}_{up,params}(z) - \mathcal{I}_{up,params}(z')\right) \\ &= n^{-1} H_{\hat{\theta}}^{-1}\left(\nabla_\theta \mathcal{L}(z,\hat{\theta}) - \nabla_\theta \mathcal{L}(z',\hat{\theta})\right).\end{aligned}$$

Since the set D is finite, we denote

$$c_1 \triangleq \arg\min_{(z,z')} \frac{|\mathcal{L}(z,\hat{\theta}_{z,z'}) - \mathcal{L}(z,\hat{\theta})|}{\|\hat{\theta}_{z,z'} - \hat{\theta}\|},$$

then

$$\begin{aligned}&|L(z_d,\hat{\theta}_{z,z'}) - L(z_d,\hat{\theta})| \geq c_1 \|\hat{\theta}_{z,z'} - \hat{\theta}\| \\ =&c_1 \|H_{\hat{\theta}}^{-1}\left(\nabla_\theta \mathcal{L}(z,\hat{\theta}) - \nabla_\theta \mathcal{L}(z',\hat{\theta})\right)\| \\ \geq&c_1 (n\sigma)^{-1} \|\nabla_\theta \mathcal{L}(z,\hat{\theta}) - \nabla_\theta \mathcal{L}(z',\hat{\theta})\|,\end{aligned}$$

where σ is the maximum singular value of the Hessian. Regarding the log-likelihood loss, $\nabla_\theta \mathcal{L}(z,\hat{\theta}) = \nabla_\theta \log Z - \nabla_\theta \hat{w}_y^\intercal h(x,\hat{\varphi})$, and the transpose of the second term is

$$\begin{matrix} & & w_y & & \varphi \\ (\,\mathbf{0}, & h^\intercal(x,\hat{\varphi}), & \hat{w}_y^\intercal \nabla_\varphi h(x,\hat{\varphi})\,). \end{matrix}$$

Hence,

$$\begin{aligned}&\|\nabla_\theta \mathcal{L}(z,\hat{\theta}) - \nabla_\theta \mathcal{L}(z',\hat{\theta})\| \\ =&\sqrt{2\|h(x,\varphi)\|^2 + \|\nabla_\varphi h(x,\varphi)^\intercal (w_y - w_{y'})\|^2} \\ \geq&\|h(x,\varphi)\| + \frac{\tau_x}{\sqrt{2}}\|w_y - w_{y'}\|.\end{aligned}$$

Let $c = c_1(\sigma\sqrt{2})^{-1}$, we get the lower bound. □

B Implementation details

For basic CNN model, the window size of the convolution layer is set to 3 and the number of the filter is set to 230. In bootstrapping procedure, the position embedding dimension of CNN is set to 1 and the word embedding is initialized with 100 dimensional pre-trained glove embedding (Pennington et al., 2014),[9] which is for making IF focuses more on semantic information. In training procedure, for fair comparison, the position embedding dimension of all models is set to 5, the word embedding dimension is set to 100 with random initialization and an entity type embedding. And for the PCNN model we have the same hype-parameters with Zeng et al. (2015). For majority vote,we set $k = 3$. In selection step of bootstrap, we at most select $n = \frac{1}{10}|D_t|$ instances. For teacher-student, the α is set to 1 and β is set to 0.9. To avoid the model just memorizes the entity pairs, we mask the entity words in both bootstrapping and training.

NYT	Training	Dev	Test
# Sentence	233038	1596	1596
# Instance	367596	4567	4484
# Positive instances	106653	975	1050

Table 4: Statistics on NYT dataset.

NYT	Training	Dev	Test
# /people/person/place_lived	7197	198	185
# /location/location/contains	51766	479	611
# /people/person/nationality	8079	117	91
# /business/person/company	5595	105	113
# /people/person/children	506	6	11
# /people/dec.../place_of_death	1936	8	14
# /location/country/capital	7690	14	15
# /business/company/founders	800	10	6
# /people/person/place_of_birth	3173	13	15
# /location/nei.../nei..._of	5553	6	7

Table 5: 10 relations on dev set. Our methods take the dev set as clean set to denoise.

C Detailed statistics on NYT dataset

Overall statistics on NYT datset are shown in Table 4. And the detailed statistics of each relations on dev set are in Table 5.

D Synthetic dataset ACE05-N

Table 6 shows the statistics on ACE05. Next we show how to construct our synthetic dataset

[9] Download from https://nlp.stanford.edu/projects/glove/.

ACE05-N	Training	Dev	Test
# GEN-AFF	290	73	55
# ORG-AFF	857	204	203
# PER-SOC	279	57	45
# PHYS	549	164	123
# PART-WHOLE	393	81	86
# ART	275	52	85
# NA	116572	27597	24363

Table 6: The statistics on ACE05.

NYT	Training	Dev	Test
# GEN-AFF	580	146	110
# ORG-AFF	1714	408	406
# PER-SOC	556	104	90
# PHYS	1098	328	246
# PART-WHOLE	786	162	172
# ART	550	104	170

Table 7: The statistics of ACE05 after adding NA.

ACE05-N, which takes two steps: adding NA and adding noise. The entity pair in a sentence that does not express any positive relation would be considered as "NA". The NA instances play two roles in our synthetic dataset: the negative instances and noisy.

Adding negative instances In DSRE, the noise come from the wrong-labelled negative instances. So the true negative instances are necessary for evaluating the denoise ability. In our synthetic dataset, we reconstruct the training, dev and test of each relation by adding the NA. The size of NA is same as the original size of training, dev and test. After that, the dataset would be changed to Table 7.

Adding noisy An instance that don't express relation r but labelled with r is a noise instance to the relation r. So the intentionally made noise is relabelling a NA instance to a positive label. In the experiment, we manually put a certain number of noise instance to poison the dataset. If we poison a train set with 50% noise, we mean that after poisoning, the noise ratio of this train set is 50%. For example, if the train set of *GEN-AFF* in Table 7 is poisoned with 50% noise, it would be put with 580 noise instances.

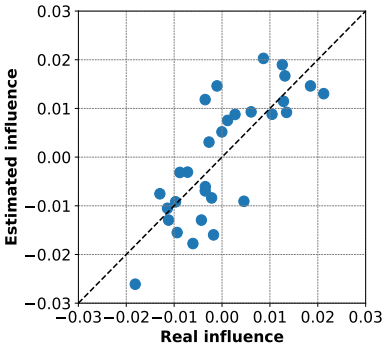

Figure 3: The correlation between estimated change of loss and real change of loss. We use a training set with 500 instances included two relation types and arbitrarily pick four instances as testing instances to validate computation of influence function. The picture shows 40 most influential points with their real difference in loss (obtained by 500 steps leave-one-out retraining).

E Validating influence function

The calculation of influence function is the key step of our method. Here we show the high correlation between the real influence (calculated by leave-one-out retraining) and estimated influence. From the experimental results (Figure 3), we find that the correlation among high influential instances is 0.79, and 0.65 in all instances. The high correlation validate that influence function is reliable in perform instance perturbation analyses.

SEE-Few: Seed, Expand and Entail for Few-shot Named Entity Recognition

Zeng Yang and **Linhai Zhang** and **Deyu Zhou***
School of Computer Science and Engineering, Key Laboratory of Computer Network and Information Integration, Ministry of Education, Southeast University, China
{yangzeng, lzhang472, d.zhou}@seu.edu.cn

Abstract

Few-shot named entity recognition (NER) aims at identifying named entities based on only few labeled instances. Current few-shot NER methods focus on leveraging existing datasets in the rich-resource domains which might fail in a training-from-scratch setting where no source-domain data is used. To tackle training-from-scratch setting, it is crucial to make full use of the annotation information (the boundaries and entity types). Therefore, in this paper, we propose a novel multi-task (Seed, Expand and Entail) learning framework, SEE-Few, for Few-shot NER without using source domain data. The seeding and expanding modules are responsible for providing as accurate candidate spans as possible for the entailing module. The entailing module reformulates span classification as a textual entailment task, leveraging both the contextual clues and entity type information. All the three modules share the same text encoder and are jointly learned. Experimental results on four benchmark datasets under the training-from-scratch setting show that the proposed method outperformed state-of-the-art few-shot NER methods with a large margin. Our code is available at https://github.com/unveiled-the-red-hat/SEE-Few.

1 Introduction

Named entity recognition (NER), focusing on identifying mention spans in text inputs and classifying them into the pre-defined entity categories, is a fundamental task in natural language processing and widely used in downstream tasks (Wang et al., 2019; Zhou et al., 2021; Peng et al., 2022). Supervised NER has been intensively studied and yielded significant progress, especially with the aid of pre-trained language models (Devlin et al., 2019; Li et al., 2020; Mengge et al., 2020; Yu et al., 2020; Shen et al., 2021; Li et al., 2021a; Chen and Kong, 2021). However, supervised NER relies on plenty of training data, which is not suitable for some specific situations with few training data.

Few-shot NER, aiming at recognizing entities based on few labeled instances, has attracted much attention in the research filed. Approaches for few-shot NER can be roughly divided into two categories, span-based and sequence-labeling-based methods. Span-based approaches enumerate text spans in input texts and classify each span based on its corresponding template score (Cui et al., 2021). Sequence-labeling-based approaches treat NER as a sequence labeling problem which assigns a tag for each token using the BIO or IO tagging scheme (Yang and Katiyar, 2020; Hou et al., 2020; Huang et al., 2021). Most of these span-based and sequence-labeling-based methods focus on leveraging existing datasets in the rich-resource domains to improve their performance in the low-resource domains. Unfortunately, the gap between the source domains and the target domains may hinder the performance of these methods (Pan and Yang, 2009; Cui et al., 2021). Moreover, these approaches might fail under the training-from-scratch setting where no source domain data is available.

Therefore, it is crucial to make full use of the in-domain annotations, which consist of two types of information: boundary information and entity type information. However, most the approaches mentioned above fail to fully utilize these information. (1) Most span-based methods simply enumerate all possible spans, ignoring the boundary information of named entities. As a large number of negative spans are generated, these approaches suffer from the bias, the tendency to classify named entities as non-entities. (2) Most sequence-labeling-based methods simply employ one-hot vectors to represent entity types while ignoring the prior knowledge of entity types.

To overcome the disadvantages mentioned above, firstly, inspired by three principles for weakly-supervised image segmentation, i.e. seed,

*Corresponding author.

Proceedings of the 29th International Conference on Computational Linguistics, pages 2540–2550
October 12–17, 2022.

expand and constrain (Kolesnikov and Lampert, 2016), we seed with relatively high-quality unigrams and bigrams in the texts, then expand them to extract the candidate spans as accurately as possible. Secondly, we cast span classification as textual entailment to naturally incorporate the entity type information. For example, to determine whether "J. K. Rowling" in "J. K. Rowling is a British author." is a PERSON entity or a non-entity, we treat "J. K. Rowling is a British author." as a premise, then construct "J. K. Rowling is a person." and "J. K. Rowling is not an entity." as hypotheses. In such way, span classification is converted into determining which hypothesis is true. Moreover, the size of training data is increased by such converting which is beneficial for few-shot settings.

In this paper, we propose SEE-Few, a novel multi-task learning framework (Seed, Expand and Entail) for Few-shot NER. The seeding and expanding modules are responsible for providing as accurate candidate spans as possible for the entailing module. Specifically, the seed selector chooses some unigrams and bigrams as seeds based on some metrics, e.g., the Intersection over Foreground. The expanding module takes a seed and the window around it into account and expands it to a candidate span. Compared with enumerating all possible n̂-gram spans, seeding and expanding can significantly reduce the number of candidate spans and alleviate the impact of negative spans in the subsequent span classification stage. The entailing module reformulates a span classification task as a textual entailment task, leveraging contextual clues and entity type information to determine whether a candidate span is an entity and what type of entity it is. All the three modules share the same text encoder and are jointly learned. Experiments were conducted on four NER datasets under training-from-scratch few-shot setting. Experimental results show that the proposed approach outperforms several state-of-the-art baselines.

The main contributions can be summarized as follows:

- A novel multi-task learning framework (Seed, Expand and Entail), SEE-Few, is proposed for few-shot NER without using source domain data. In specific, the seeding and expanding modules provide as accurate candidate spans as possible for the entailing module. The entailing module reformulates span classification as a textual entailment task, leveraging contextual clues and entity type information.
- Experiments were conducted on four NER datasets in training-from-scratch few-shot setting. Experimental results show that the proposed approach outperforms the state-of-the-art baselines by significant margins.

2 Related Work

2.1 Few-shot NER

Few-shot NER aims at recognizing entities based on only few labeled instances from each category. A few approaches have been proposed for few-shot NER. Methods based on prototypical network (Snell et al., 2017) require complex episode training (Fritzler et al., 2019; Hou et al., 2020). Yang and Katiyar (2020) abandon the complex meta-training and propose NNShot, a distance-based method with a simple nearest neighbor classifier. Huang et al. (2021) investigate three orthogonal schemes to improve the model generalization ability for few-shot NER. TemplateNER (Cui et al., 2021) enumerates all possible text spans in input text as candidate spans and classifies each span based on its corresponding template score. Ma et al. (2021) propose a template-free method to reformulate NER tasks as language modeling (LM) problems without any templates. Tong et al. (2021) propose to mine the undefined classes from miscellaneous other-class words, which also benefits few-shot NER. Ding et al. (2021) present Few-NERD, a large-scale human-annotated few-shot NER dataset to facilitate the research.

However, most of these studies follow the manner of episode training (Fritzler et al., 2019; Hou et al., 2020; Tong et al., 2021; Ding et al., 2021) or assume an available rich-resource source domain (Yang and Katiyar, 2020; Cui et al., 2021), which is in contrast to the real word application scenarios that only very limited labeled data is available for training and validation (Ma et al., 2021). EntLM (Ma et al., 2021) is implemented on training-from-scratch few-shot setting, but still needs distant supervision datasets for label word searching. The construction of distant supervision datasets requires additional expert knowledge. Some works study generating NER datasets automatically to reduce labeling costs (Kim et al., 2021; Li et al., 2021b). In this paper, we focus on the few-shot setting without source domain data

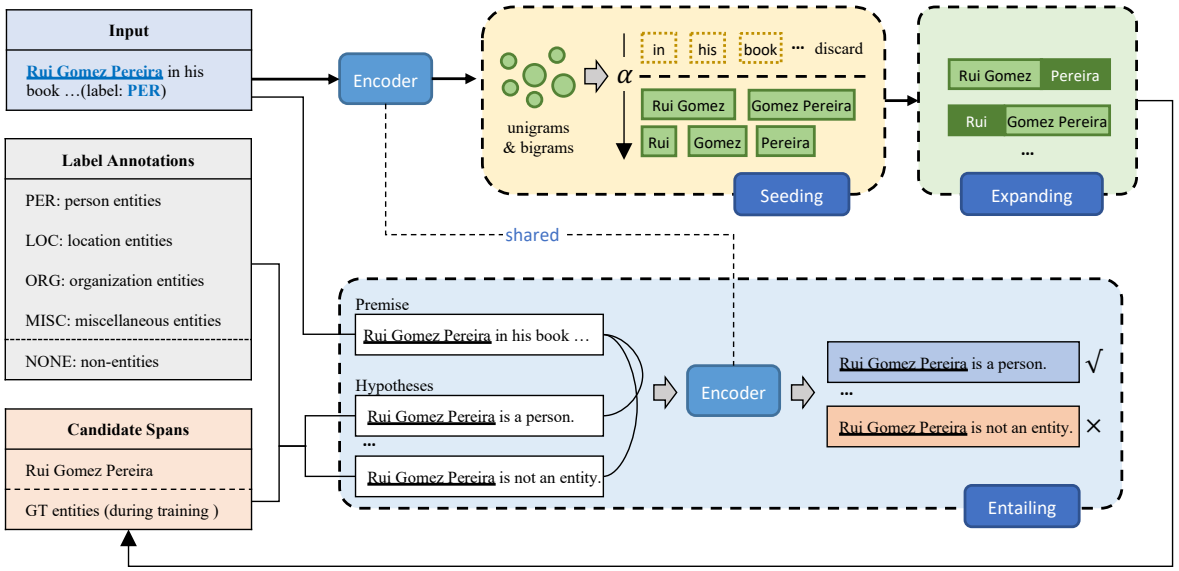

Figure 1: The architecture of the proposed approach, SEE-Few, which consists of three main modules: seeding, expanding, and entailing.

which makes minimal assumptions about available resources.

2.2 Three Principles for Weakly-Supervised Image Segmentation

Semantic image segmentation is a computer vision technique which aims at assigning a semantic class label to each pixel of an image. Kolesnikov and Lampert (2016) introduce three guiding principles for weakly-supervised semantic image segmentation: to seed with weak localization cues, to expand objects based on the information of possible classes in the image, and to constrain the segmentation with object boundaries.

3 Methodologies

3.1 Problem Setting

We decompose NER to two subtasks: span extraction and span classification. Given an input text $\mathbf{X} = \{x_1, \ldots, x_n\}$ as a sequence of tokens, a span starting from x_l and ending with x_r (i.e., $\{x_l, \ldots, x_r\}$) can be denote as $s = (l, r)$, where $1 \leq l \leq r \leq n$. The span extraction task is to obtain a candidate span set $\mathbf{C} = \{c_1, \ldots, c_m\}$ from the input text. Given an entity type set $\mathbf{T}^+ = \{t_1, \ldots, t_{v-1}\}$ and the candidate span set $\mathbf{C}$ produced by span extraction, the target of span classification is assign an entity category $t \in \mathbf{T}^+$ or the non-entity category to each candidate span. For convenience, we denote an entity type set including the non-entity type as $\mathbf{T} = \{t_1, \ldots, t_{v-1}, t_{none}\}$, where t_{none} represents the non-entity type and v is the size of $\mathbf{T}$.

3.2 The Architecture

Figure 1 illustrates the architecture of the proposed approach, SEE-Few, which consists of three main modules: seeding, expanding, and entailing. The input text will first be sent to the seeding module to generate informative seeds, then the seeds will be expanded to candidate spans in the expanding module, finally the candidate spans will be classified with an entailment task in the entailing module. We will discuss the details of each modules in the following sections.

3.2.1 Seeding

Given an input text $\mathbf{X} = \{x_1, \ldots, x_n\}$ consisting of n tokens, a unigram consists of one token and a bigram consists of two consecutive tokens. We denote the set of unigrams and bigrams in the input text as $\mathbf{S} = \{s_1, \ldots, s_{2n-1}\}$, where $s_i = (l_i, r_i)$ denotes i-th span, and l_i, r_i denote the left and right boundaries of the span respectively.

Seeding is to find the unigrams and bigrams that overlap with entities and have the potential to be expanded to named entities, which is important for the following seed expansion. It can be accomplished by constructing a seeding model and predicting the seed score for each candidate unigram

or bigram.

Firstly, we feed the input text into BERT to obtain the representation $h \in \mathbb{R}^{n \times d}$, where d is the dimension of the BERT hidden states. For the span $s_i = (l_i, r_i)$, its representation h_i^{seed} is the concatenation of the mean pooled span representation h_i^p and the representation of the `[CLS]` token $h^{[\texttt{CLS}]}$. The seed score is calculated as follows:

$$h_i^p = \text{MeanPooling}(h_{l_i}, \ldots, h_{r_i}) \quad (1)$$

$$h_i^{seed} = \text{Concat}(h_i^p, h^{[\texttt{CLS}]}) \quad (2)$$

$$p_i^{seed} = \text{Sigmoid}(\text{MLP}_\text{s}(h_i^{seed})) \quad (3)$$

where MLP denotes the multilayer perceptron with a GULE function in the last layer. We set the threshold α and select the span whose seed score is above α as a seed to expand.

To train the seeding model, we need to construct a dataset consisting of unigrams (bigrams) and their seeding scores. We construct the seed score based on Intersection over Foreground (IoF). Intersection over Union (IoU) is used to measure the overlap between objects in object detection which is defined as $\text{IoU}(A, B) = \frac{A \cap B}{A \cup B}$ in NER, where A and B are two spans (Chen et al., 2020; Shen et al., 2021). However, IoU is not suitable for the seeding stage. Considering an entity consisting of five words, e.g., "International Conference on Computational Linguistics", IoU between the bigram "International Conference" and the entity is 0.4, not significant. Intersection over Foreground (IoF) can be a better choice which is defined as $\text{IoF}(A, B) = \frac{A \cap B}{A}$, where A is the foreground (i.e., a unigram or bigram) and B is the background (i.e., a named entity). In the above example, IoF between "International Conference" and the entity is 1.0, indicating that it is part of the entity and has the potential to be expanded to the whole entity. We assign each s the IoF between it and its closed named entity as the ground-truth seed score $\hat{y}_i^{seed}$ which indicates the potential to be expanded to the whole entity.

3.2.2 Expanding

For named entities consisting of more than two words, the seeds generated in seeding stage are only part of them and need to be expanded to the whole entities. Expanding is a regression task to learn the boundary offsets $\hat{o}$ between a seed and the named entity closed to the seed.

Expanding is allowed to offset the left and right boundaries of the seed by up to λ, respectively, which means that the longest entity we can get is an entity of length $2 + 2\lambda$. Besides, expanding needs to consider a window around the seed in addition to the seed itself. For the seed $s_i = (l_i, r_i)$, the maximum expansion is denoted as:

$$s_i^{exp\_max} = (\min(1, l_i - \lambda), \max(n, r_i + \lambda)) \quad (4)$$

If we use $s_i^{exp\_max}$ as the window around s_i, it may not provide enough information to distinguish the boundaries for the maximum expansion. Thus, the window for s_i should be larger than $s_i^{exp\_max}$, defined as w_i:

$$w_i^l = \min(1, l_i - 2\lambda) \quad (5)$$

$$w_i^r = \max(n, r_i + 2\lambda) \quad (6)$$

$$w_i = (w_i^l, w_i^r) \quad (7)$$

We concatenate the mean pooled span representation h_i^p of seed s_i and the mean pooled span representation h_i^w of window w_i. Then the offsets o_i of left and right boundaries are calculated as follows:

$$h_i^w = \text{MeanPooling}(h_{w_i^l}, \ldots, h_{w_i^r}) \quad (8)$$

$$h_i^{exp} = \text{Concat}(h_i^p, h_i^w) \quad (9)$$

$$o_i = \lambda \cdot (2 \cdot \text{Sigmoid}(\text{MLP}_\text{e}(h_i^{exp})) - 1) \quad (10)$$

where $o_i \in \mathbb{R}^2$. The first element of o_i can be denoted as o_i^l, indicating the offset of the seed's left boundary. Likewise, the second element o_i^r indicates the offset of the seed's right boundary, and $o_i^l, o_i^r \in [-\lambda, \lambda]$. We can obtain the result of expanding, i.e., a candidate span with the new boundaries l_i' and r_i':

$$l_i' = \max(1, l_i + \left\lfloor o_i^l + \frac{1}{2} \right\rfloor) \quad (11)$$

$$r_i' = \min(n, r_i + \left\lfloor o_i^r + \frac{1}{2} \right\rfloor) \quad (12)$$

The duplicate results and invalid results that $l_i' > r_i'$ are discarded. At this point, a set of candidate spans are produced for span classification.

3.2.3 Entailing

The entailing module reformulates span classification as a textual entailment task, leveraging contextual clues and entity type information. To cast span classification as textual entailment, we need to construct textual entailment pairs. For the i-th candidate span c_i, the entailment pair is constructed as $(\mathbf{X}, \mathbf{E}_i^j)$, where $\mathbf{E}_i^j = \{c_i, \text{is}, \text{a}, t_j\}$ and $t_j \in \mathbf{T}$. Please refer to Appendix A for detailed templates used to construct entailment pairs. The entailment label $\hat{y}_{i,j}^{entail}$ for $(\mathbf{X}, \mathbf{E}_i^j)$ can be obtained by:

$$\hat{y}_{i,j}^{entail} = \begin{cases} \texttt{entail}, & \text{if } c_i \text{ belongs to } t_j \\ \texttt{not entail}, & \text{otherwise} \end{cases} \tag{13}$$

The entailment pair $(\mathbf{X}, \mathbf{E}_i^j)$ is fed into the shared text encoder to obtain the representation of the `[CLS]` token $h_{\mathbf{E}_{i,j}}^{\texttt{[CLS]}} \in \mathbb{R}^d$ and the binary textual entailment classification can be performed:

$$p_{i,j}^{entail} = \text{Softmax}(\text{MLP}_{\text{entail}}(h_{\mathbf{E}_{i,j}}^{\texttt{[CLS]}})) \tag{14}$$

To ensure that all ground-truth entities are learned, we add ground-truth entities to the candidate span set $\mathbf{C}$ during the training phase.

3.3 Training Objective

Both seeding and expanding are regression tasks, the seeding loss $\mathcal{L}_{seed}$ and expansion loss $\mathcal{L}_{exp}$ are defined as follows:

$$\mathcal{L}_{seed} = \sum_i \text{SmoothL1}\left(\hat{y}_i^{seed}, p_i^{seed}\right) \tag{15}$$

$$\mathcal{L}_{exp} = \sum_i \sum_{j \in \{l,r\}} \text{SmoothL1}\left(\hat{o}_i^j, o_i^j\right) \tag{16}$$

For the entailing module, since the number of instances with the `not entail` label is bigger than the number of instances with the `entail` label, we use focal loss (Lin et al., 2017) to solve the label imbalance problem:

$$FL(p, y) = \begin{cases} -(1-p_{i,j})^\gamma \log(p_{i,j}), & \text{if } y = 1 \\ -(p_{i,j})^\gamma \log(1-p_{i,j}), & \text{otherwise} \end{cases} \tag{17}$$

$$\mathcal{L}_{entail} = \sum_i \sum_j FL(p_{i,j}^{entail}, \hat{y}_{i,j}^{entail}) \tag{18}$$

where γ denotes the focusing parameter of focal loss.

The multi-task framework is trained by minimizing the combined loss defined as follows:

$$\mathcal{L} = \beta_1 \mathcal{L}_{seed} + \beta_2 \mathcal{L}_{exp} + \beta_3 \mathcal{L}_{entail} \tag{19}$$

where β_1, β_2 and β_3 are hyperparameters controlling the relative contribution of the respective loss term.

3.4 Entity Decoding

The entailing module will output an entailment score $p_{i,j}^{entail}$ for the entailment pair $(\mathbf{X}, \mathbf{E}_i^j)$, where $\mathbf{E}_i^j = \{c_i, \text{is}, \text{a}, t_j\}$, $c_i \in \mathbf{C}$ and $t_j \in \mathbf{T}$. We collect all entailment pairs associated with the candidate span c_i, then assign c_i the entity type with the highest entailment score. If two candidate spans have overlap, the span with a higher score will be selected as the final result.

4 Experiment Settings

4.1 Training-from-scratch Few-shot Settings

Different from most previous few-shot NER studies that assume source-domain data is available, we consider a training-from-scratch setting, which is more practical and challenging. Specifically, we assume only K examples for each entity class in the training set and validation set respectively, where $K \in \{5, 10, 20\}$.

4.2 Datasets Construction

For fair comparison, we manually construct the few-shot datasets. With $K \in \{5, 10, 20\}$, we follow the greedy sampling strategy in (Yang and Katiyar, 2020) to ensure the sample number K of each category. To make the experimental results more convincing and credible, we randomly sample 5 different groups of training sets and validation sets for each K. We employ these strategies on four NER datasets from different domains: CoNLL2003 dataset (Sang and De Meulder, 2003) in news domain, MIT-Restaurant dataset (Liu et al., 2013) in review domain, WikiGold dataset (Balasuriya et al., 2009) in general domain and Weibo dataset (He and Sun, 2016) in social media domain. Table 2 shows the statistics on these original datasets. The self-constructed datasets are public available with the code for reproducibility.

Datasets	Methods	$K=5$			$K=10$			$K=20$		
		P	R	F_1	P	R	F_1	P	R	F_1
CoNLL03	LC-BERT	42.83	30.72	$35.06_{(6.09)}$	50.36	52.13	$51.20_{(6.39)}$	56.33	63.85	$59.84_{(1.43)}$
	Prototype	38.26	43.14	$40.37_{(8.06)}$	45.08	64.02	$52.82_{(3.22)}$	43.94	69.72	$53.89_{(1.95)}$
	NNShot	32.11	38.42	$34.92_{(3.30)}$	34.10	40.98	$37.18_{(5.82)}$	38.43	47.85	$42.61_{(2.23)}$
	StructShot	30.04	21.33	$23.43_{(4.52)}$	38.62	19.72	$26.09_{(7.23)}$	44.96	28.59	$34.87_{(1.30)}$
	TemplateNER	26.90	23.46	$23.13_{(8.40)}$	44.51	43.99	$44.01_{(4.82)}$	52.16	56.46	$54.01_{(5.09)}$
	Ours	60.45	51.27	$\mathbf{55.21}_{(3.93)}$	66.19	58.68	$\mathbf{61.99}_{(1.73)}$	69.49	67.07	$\mathbf{68.21}_{(2.60)}$
MIT-Restaurant	LC-BERT	41.21	38.65	$39.88_{(3.79)}$	43.60	48.93	$46.08_{(3.75)}$	56.24	60.04	$58.07_{(1.50)}$
	Prototype	27.77	46.79	$34.84_{(1.63)}$	30.37	50.64	$37.97_{(2.29)}$	37.91	59.31	$46.25_{(1.62)}$
	NNShot	28.15	34.81	$31.11_{(2.30)}$	30.28	37.65	$33.56_{(1.48)}$	36.72	45.55	$40.66_{(1.26)}$
	StructShot	45.13	25.00	$31.93_{(4.32)}$	43.94	28.19	$34.30_{(2.56)}$	52.08	36.18	$42.69_{(1.12)}$
	TemplateNER	23.11	20.78	$21.53_{(4.66)}$	39.45	28.77	$32.71_{(8.14)}$	46.93	37.00	$41.26_{(6.80)}$
	Ours	53.08	39.47	$\mathbf{45.25}_{(3.18)}$	57.19	46.41	$\mathbf{51.20}_{(1.48)}$	64.79	57.22	$\mathbf{60.75}_{(2.07)}$
WikiGold	LC-BERT	36.02	8.02	$12.57_{(7.81)}$	43.13	8.95	$37.72_{(7.20)}$	50.68	50.73	$50.68_{(5.94)}$
	Prototype	20.55	21.46	$19.28_{(8.12)}$	23.31	45.21	$30.59_{(3.95)}$	27.31	56.22	$36.56_{(8.65)}$
	NNShot	27.81	34.16	$30.63_{(1.91)}$	26.36	37.92	$30.93_{(4.89)}$	28.33	39.07	$32.81_{(5.41)}$
	StructShot	49.00	13.37	$20.88_{(4.61)}$	43.21	14.19	$21.28_{(2.96)}$	43.51	15.94	$23.16_{(2.18)}$
	TemplateNER	18.45	19.45	$17.26_{(12.73)}$	38.33	45.37	$41.04_{(13.19)}$	57.39	56.00	$56.60_{(3.22)}$
	Ours	61.23	41.01	$\mathbf{48.87}_{(8.01)}$	63.36	48.74	$\mathbf{54.98}_{(3.24)}$	69.06	58.25	$\mathbf{63.19}_{(1.28)}$
Weibo	LC-BERT	36.93	26.32	$29.95_{(13.93)}$	46.49	53.19	$49.54_{(3.96)}$	54.27	58.53	$56.23_{(1.48)}$
	Prototype	14.32	37.68	$20.64_{(7.07)}$	21.27	59.42	$31.25_{(2.64)}$	21.27	59.42	$37.39_{(2.58)}$
	NNShot	4.64	10.57	$06.45_{(2.65)}$	6.58	13.73	$08.90_{(1.27)}$	11.77	26.61	$16.32_{(0.80)}$
	StructShot	16.77	1.53	$02.80_{(1.63)}$	38.48	3.21	$05.91_{(1.93)}$	52.05	5.93	$10.65_{(1.73)}$
	TemplateNER	4.12	16.70	$04.41_{(4.67)}$	5.12	27.27	$08.31_{(3.11)}$	10.70	29.57	$15.24_{(7.09)}$
	Ours	49.51	48.51	$\mathbf{48.67}_{(4.05)}$	55.12	57.65	$\mathbf{56.07}_{(1.62)}$	57.10	57.70	$\mathbf{57.21}_{(1.62)}$

Table 1: Performance comparison of SEE-Few and baselines on four datasets under different Ks.

Dataset	Domain	Language	# Class	# Train	# Test
CoNLL03	News	English	4	14,987	3,684
MIT-Restaurant	Review	English	8	7,660	1,521
WikiGold	General	English	4	1,017	339
Weibo	Social Media	Chinese	8	1,350	270

Table 2: Statistics on the original datasets used to construct our few-shot datasets.

4.3 Baselines

We compare the proposed model with five competitive baselines.

LC-BERT (Devlin et al., 2019) BERT with a linear classifier which is applied to project the contextualized representation of each token into the label space.

Prototype (Huang et al., 2021) A method based on prototypical network (Snell et al., 2017), represents the entity categories as vectors in the same representation space of individual tokens and utilizes the nearest neighbor criterion to assign the entity category.

NNShot and **StructShot** (Yang and Katiyar, 2020) NNShot is a metric-based few-shot NER method that leverages a nearest neighbor classifier for few-shot prediction. StructShot is based on NNShot and use the Viterbi algorithm for decoding predictions. These methods pre-train the model with a dataset from other rich-resource domain (source domain) which is unavailable in our training-from-scratch setting. We re-implement them and directly apply them on target domains.

TemplateNER (Cui et al., 2021) A template-based prompt learning method which fine-tunes BART (Lewis et al., 2020) to generate pre-defined templates filled by enumerating text spans from input texts.

4.4 Implementation Details

For the proposed model and all the baselines except TemplateNER, we implement them based on "bert-base-uncased" for English datasets and "bert-base-chinese" for Chinese datasets. TemplateNER uses BART-large (Lewis et al., 2020) as the backbone on English datasets and Chinese BART-large (Shao et al., 2021) as the backbone on Chinese datasets. For all the baselines, we use the recommended parameters provided by the original paper or the official implementation.

For the proposed model, the number of epochs is 35. The batch sizes of seeding and expanding are 1, and the batch sizes of entailing are 16, 16, 16, 8 on CoNLL03, MIT-Restaurant, WikiGold and Weibo, respectively. The threshold αs on CoNLL03, MIT-Restaurant, WikiGold and Weibo, are set to 0.5, 0.6,

Model	CoNLL03			MIT-Restaurant			WikiGold			Weibo		
	P	R	F_1	P	R	F_1	P	R	F_1	P	R	F_1
Full model	60.45	51.27	**55.21**	53.08	39.47	**45.25**	61.23	41.01	**48.87**	49.51	48.51	**48.67**
w/o seeding	63.20	45.46	52.36	53.45	38.11	44.15	61.79	36.66	45.68	56.46	40.72	47.23
w/o expanding	61.70	43.41	50.66	52.13	34.30	41.27	61.93	27.48	37.58	48.62	28.13	35.22
w/o entailing	50.93	41.15	45.01	46.60	31.43	37.13	69.39	28.47	40.02	38.32	22.58	27.14
w/o seed & exp	57.29	41.35	47.59	54.73	33.77	41.55	71.87	23.10	34.43	49.06	24.26	31.97
repl IoF with IoU	60.10	44.14	50.44	36.65	31.07	33.58	51.47	27.21	34.50	66.92	13.49	20.97

Table 3: Ablation study on 5-shot setting with the metrics of precision, recall and F1-score.

Model	CoNLL03		MIT-Restaurant		WikiGold		Weibo	
	$\|\mathbf{C}\|$	$\|\mathbf{C}\|$ / #Sen	$\|\mathbf{C}\|$	$\|\mathbf{C}\|$ / #Sen	$\|\mathbf{C}\|$	$\|\mathbf{C}\|$ / #Sen	$\|\mathbf{C}\|$	$\|\mathbf{C}\|$ / #Sen
Full model	13464 (×0.15)	3.65	7435 (×0.28)	4.48	1461 (×0.11)	4.31	1030 (×0.04)	3.81
w/o seeding	59053 (×0.66)	16.03	17780 (×0.66)	15.74	9924 (×0.72)	29.27	19772 (×0.67)	73.23
w/o expanding	15055 (×0.17)	4.09	8281 (×0.31)	5.11	1662 (×0.12)	4.90	2093 (×0.07)	7.75
w/o entailing	19350 (×0.22)	5.25	7625 (×0.28)	4.68	1656 (×0.12)	4.88	1272 (×0.04)	4.71
w/o seed & exp	89648 (×1.00)	24.33	26991 (×1.00)	17.75	13833 (×1.00)	40.81	29352 (×1.00)	108.71
repl IoF with IoU	11190 (×0.12)	3.04	4098 (×0.15)	1.76	490 (×0.04)	1.45	276 (×0.01)	1.02
#Entity / #Sent	1.53		2.07		2.15		1.55	

Table 4: Ablation study on 5-shot setting with entity-related statistics. $|\mathbf{C}|$ denotes the number of candidate spans during the testing phase. $|\mathbf{C}|$ / #Sen denotes the average number of candidate spans per sentence during the testing phase. #Entity / #Sent denotes the average number of named entities per sentence. $(\cdot)$ indicates the ratio to the number of candidate spans that produced by w/o seed & exp (e.g., the number of all unigrams and bigrams).

0.7, 0.7, respectively. λ is set to 5. The focusing parameter of focal loss γ is set to 2. β_1, β_2 and β_3 are set to 1, 1 and 1, respectively. The dropouts before the seeding, expanding and entailing are set with a rate of 0.5. The loss function is minimized using AdamW optimizer with a learning rate of 3e-05 and a linear warmup-decay learning rate schedule.

5 Experimental Results

5.1 Overall Results

Table 1 shows the performances of the proposed method and the baselines under different K-shot settings. From the table, we can observe that: (1) The proposed method performs consistently better than all the baseline methods. Specifically, the F1-scores of our model advance previous models by +18.72%, +18.24%, +14.84%, +5.37% on Weibo, WikiGold, CoNLL03 and MIT-Restaurant respectively on 5-shot setting, which verifies the effectiveness of our approach in exploiting few-shot data. (2) Compared to baselines, our method can achieve comparable performance with less training data. Specifically, our approach achieves an F1-score of 55.21% on CoNLL03 dataset on 5-shot setting, which is better than the result of TemplateNER on 20-shot setting.

5.2 Ablation Study

To validate the effectiveness of different components in our approach, we performed ablation experiments on 5-shot setting with a series of variants of SEE-Few. Table 3 shows the results with the metrics of precision, recall and F1-score, and Table 4 demonstrates how the number of candidate spans changes in ablations. The variants are as follows:

w/o seeding: removing the seeding module and directly enumerating all unigrams and bigrams as the seeds to expand. With the aid of expanding, this variant reduces 32.25% unigrams and bigrams on average. The recalls and F1-scores drop by 4.83% and 2.15% on average, respectively. The results show that the reduction of candidate span is mainly contributed by the seeding module and this variant suffers from the bias, the tendency to classify named entities as non-entities because of a large number of negative spans.

w/o expanding: removing the expanding module and directly using the seeds as candidate spans to entail. With the aid of seeding, this variant reduces 83.25% unigrams and bigrams on average. The recalls and F1-scores drop by 11.74% and 8.32% on average, respectively. Without expanding, this variant can not identify the entities whose lengths are greater than 2, achieving worse

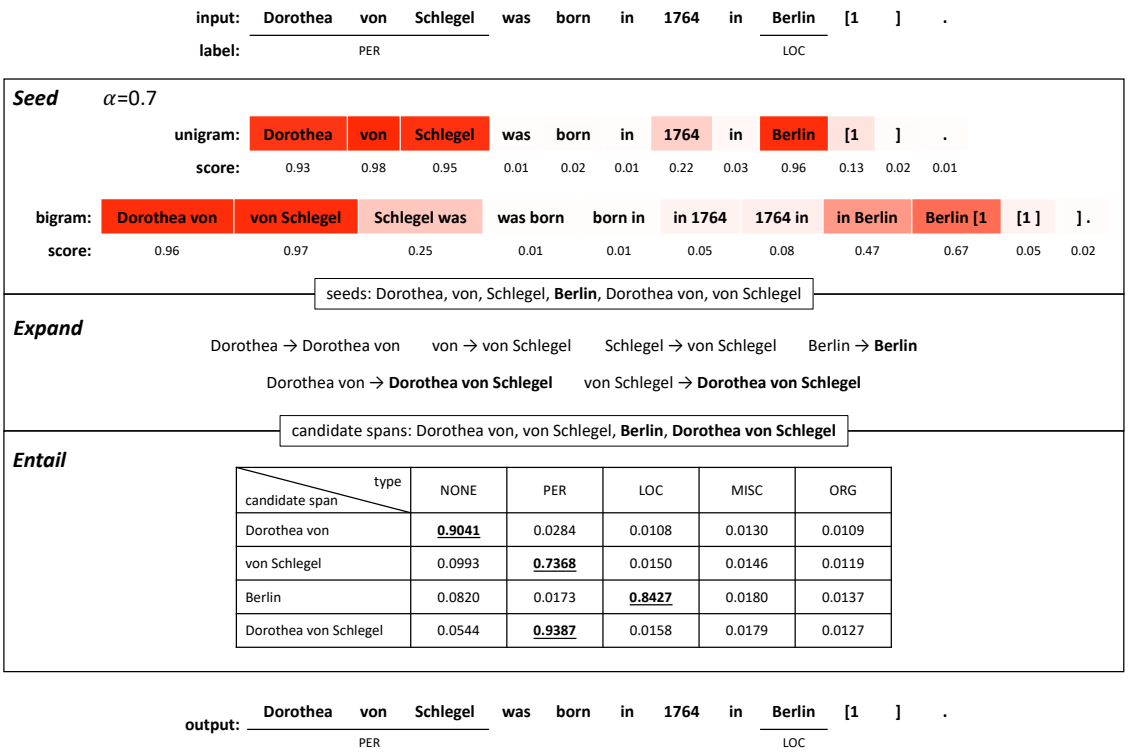

Figure 2: Case study. The result was predicted by SEE-Few trained on WikiGold, 5-shot setting.

performance.

w/o entailing: replacing the entailing module with a multi-class classifier. With the aid of seeding and expanding, this variant reduces 83.5% unigrams and bigrams on average. The F1-scores drop significantly by 12.17% on average, while dropping more sharply on datasets with fine-grained entity types (i.e., MIT-Restaurant and Weibo) than on datasets with coarse-grained entity types (i.e., CoNLL03 and WikiGold). The results demonstrate that the improvement comes from exploiting of contextual clues and label knowledge, and the entailing module can better distinguish different entity types than a multi-class classifier.

w/o seed & exp: removing both the seeding module and the expanding module in the same way as the w/o seeding and w/o seeding variants. This variant is equivalent to the entailing module classifying all the unigrams and bigrams into the entity categories or the non-entity category. In addition to a significant drop in performance, this variant is time consuming. It is on average 28.50 times slower than the full model on Weibo dataset.

repl IoF with IoU: using IoU as the ground-truth seed score instead of IoF during seeding, and keeping the threshold αs on CoNLL03, MIT-Restaurant, WikiGold and Weibo as 0.5, 0.6, 0.7, 0.7, respectively. The results demonstrate that IoF is a better choice to evaluate the qualities of unigrams and bigrams than IoU.

All the above experiments show the effectiveness of each component in our approach. Seeding and expanding can significantly reduce the number of candidate spans and alleviate the impact of negative spans in the subsequent span classification stage. The entailing module leverages contextual clues and entity type information benefiting span classification.

5.3 Case Study

Figure 2 shows an example of model predictions. We visualize the seed scores and observe that the unigrams and bigrams contained in the ground-truth entities are assigned with higher scores. The threshold α is set to 0.7 in the experiment, so "Dorothea", "von", "Schlegel", "Berlin", "Dorothea von" and "von Schlegel", totally 6 spans, are selected as seeds to expand. Among them, "Berlin" already hits the entity exactly, "Dorothea von" and "von Schlegel" are both expanded to a ground-truth entity "Dorothea von Schlegel". Other seeds are not expanded to the ground-truth entities, but do not lead an error in the final output, attributed to the success of the entailing module in deter-

mining "Dorothea von" is not an entity, and assigning a higher score to "Dorothea von Schlegel" with PER type than another candidate span (i.e., "von Schlegel") overlapping with "Dorothea von Schlegel". Considering that in a data-scarce scenario where error propagation is inevitable, our approach can still mitigate the impact of error propagation to a certain extent, which demonstrates the superiority of the proposed paradigm.

6 Conclusion

In this work, we propose a novel multi-task (Seed, Expand and Entail) learning framework, SEE-Few, for Few-shot NER without using source domain data. The seeding and expanding modules are responsible for providing as accurate candidate spans as possible for the entailing module. The entailing module reformulates span classification as a textual entailment task, leveraging both the contextual clues and entity type information. All the three modules share the same text encoder and are jointly learned. To investigate the effectiveness of the proposed method, extensive experiments are conducted under the training-from-scratch few-shot setting. The proposed method outperforms other state-of-the-art few-shot NER methods by a large margin. For future work, we will combine the framework with contrastive learning to effectively make use of limited data and further enhance the performance of few-shot NER.

Acknowledgements

The authors would like to thank the anonymous reviewers for the insightful comments. This work was funded by the National Natural Science Foundation of China (62176053).

References

Dominic Balasuriya, Nicky Ringland, Joel Nothman, Tara Murphy, and James R Curran. 2009. Named entity recognition in wikipedia. In *Proceedings of the 2009 Workshop on The People's Web Meets NLP: Collaboratively Constructed Semantic Resources (People's Web)*, pages 10–18.

Chun Chen and Fang Kong. 2021. Enhancing Entity Boundary Detection for Better Chinese Named Entity Recognition. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 2: Short Papers)*, pages 20–25, Online. Association for Computational Linguistics.

Yanping Chen, Lefei Wu, Liyuan Deng, Yongbin Qing, Ruizhang Huang, Qinghua Zheng, and Ping Chen. 2020. A boundary regression model for nested named entity recognition. *arXiv preprint arXiv:2011.14330*.

Leyang Cui, Yu Wu, Jian Liu, Sen Yang, and Yue Zhang. 2021. Template-based named entity recognition using bart.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. Bert: Pre-training of deep bidirectional transformers for language understanding.

Ning Ding, Guangwei Xu, Yulin Chen, Xiaobin Wang, Xu Han, Pengjun Xie, Haitao Zheng, and Zhiyuan Liu. 2021. Few-NERD: A few-shot named entity recognition dataset. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 3198–3213, Online. Association for Computational Linguistics.

Alexander Fritzler, Varvara Logacheva, and Maksim Kretov. 2019. Few-shot classification in named entity recognition task. In *Proceedings of the 34th ACM/SIGAPP Symposium on Applied Computing*, pages 993–1000.

Hangfeng He and Xu Sun. 2016. F-score driven max margin neural network for named entity recognition in chinese social media. *CoRR*, abs/1611.04234.

Yutai Hou, Wanxiang Che, Yongkui Lai, Zhihan Zhou, Yijia Liu, Han Liu, and Ting Liu. 2020. Few-shot slot tagging with collapsed dependency transfer and label-enhanced task-adaptive projection network. *arXiv preprint arXiv:2006.05702*.

Jiaxin Huang, Chunyuan Li, Krishan Subudhi, Damien Jose, Shobana Balakrishnan, Weizhu Chen, Baolin Peng, Jianfeng Gao, and Jiawei Han. 2021. Few-shot named entity recognition: A comprehensive study. *EMNLP*.

Hyunjae Kim, Jaehyo Yoo, Seunghyun Yoon, Jinhyuk Lee, and Jaewoo Kang. 2021. Simple questions generate named entity recognition datasets.

Alexander Kolesnikov and Christoph H Lampert. 2016. Seed, expand and constrain: Three principles for weakly-supervised image segmentation. In *European conference on computer vision*, pages 695–711. Springer.

Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Veselin Stoyanov, and Luke Zettlemoyer. 2020. BART: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7871–7880, Online. Association for Computational Linguistics.

Fei Li, Zheng Wang, Siu Cheung Hui, Lejian Liao, Dandan Song, Jing Xu, Guoxiu He, and Meihuizi Jia. 2021a. Modularized Interaction Network for Named Entity Recognition. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 200–209, Online. Association for Computational Linguistics.

Jiacheng Li, Haibo Ding, Jingbo Shang, Julian McAuley, and Zhe Feng. 2021b. Weakly supervised named entity tagging with learnable logical rules.

Xiaoya Li, Jingrong Feng, Yuxian Meng, Qinghong Han, Fei Wu, and Jiwei Li. 2020. A Unified MRC Framework for Named Entity Recognition. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 5849–5859, Online. Association for Computational Linguistics.

Tsung-Yi Lin, Priya Goyal, Ross Girshick, Kaiming He, and Piotr Dollár. 2017. Focal loss for dense object detection. In *2017 IEEE International Conference on Computer Vision (ICCV)*, pages 2999–3007.

Jingjing Liu, Panupong Pasupat, Scott Cyphers, and Jim Glass. 2013. Asgard: A portable architecture for multilingual dialogue systems. In *2013 IEEE International Conference on Acoustics, Speech and Signal Processing*, pages 8386–8390. IEEE.

Ruotian Ma, Xin Zhou, Tao Gui, Yiding Tan, Qi Zhang, and Xuanjing Huang. 2021. Template-free prompt tuning for few-shot ner.

Xue Mengge, Bowen Yu, Zhenyu Zhang, Tingwen Liu, Yue Zhang, and Bin Wang. 2020. Coarse-to-Fine Pretraining for Named Entity Recognition. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 6345–6354.

Sinno Jialin Pan and Qiang Yang. 2009. A survey on transfer learning. *IEEE Transactions on knowledge and data engineering*, 22(10):1345–1359.

Keqin Peng, Chuantao Yin, Wenge Rong, Chenghua Lin, Deyu Zhou, and Zhang Xiong. 2022. Named entity aware transfer learning for biomedical factoid question answering. *IEEE/ACM Transactions on Computational Biology and Bioinformatics*, 19(4):2365–2376.

Erik F Sang and Fien De Meulder. 2003. Introduction to the conll-2003 shared task: Language-independent named entity recognition. *arXiv preprint cs/0306050*.

Yunfan Shao, Zhichao Geng, Yitao Liu, Junqi Dai, Fei Yang, Li Zhe, Hujun Bao, and Xipeng Qiu. 2021. Cpt: A pre-trained unbalanced transformer for both chinese language understanding and generation. *arXiv preprint arXiv:2109.05729*.

Yongliang Shen, Xinyin Ma, Zeqi Tan, Shuai Zhang, Wen Wang, and Weiming Lu. 2021. Locate and label: A two-stage identifier for nested named entity recognition. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics*.

Jake Snell, Kevin Swersky, and Richard S Zemel. 2017. Prototypical networks for few-shot learning. *arXiv preprint arXiv:1703.05175*.

Meihan Tong, Shuai Wang, Bin Xu, Yixin Cao, Minghui Liu, Lei Hou, and Juanzi Li. 2021. Learning from miscellaneous other-class words for few-shot named entity recognition.

Rui Wang, Deyu Zhou, and Yulan He. 2019. Open event extraction from online text using a generative adversarial network. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 282–291, Hong Kong, China. Association for Computational Linguistics.

Yi Yang and Arzoo Katiyar. 2020. Simple and effective few-shot named entity recognition with structured nearest neighbor learning.

Juntao Yu, Bernd Bohnet, and Massimo Poesio. 2020. Named Entity Recognition as Dependency Parsing. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 6470–6476, Online. Association for Computational Linguistics.

Deyu Zhou, Kai Sun, Mingqi Hu, and Yulan He. 2021. Image generation from text with entity information fusion. *Knowledge-Based Systems*, 227:107200.

A Entity Types and Templates

In the span classification stage, the entailing module reformulates span classification as a textual entailment task to leverage contextual clues and entity type information. Table 5 shows the entity types in each dataset and corresponding natural language templates we use in our experiments. Besides, for English datasets, we use "is not an entity." as the template of the non-entity type. For Chinese datasets, we use "不是命名实体。" as the template of the non-entity type.

Dataset	Entity Type	Template
CoNLL03	PER	is a person.
	LOC	is a location.
	MISC	is a miscellaneous entity.
	ORG	is an organization.
MIT Restaurant	Hours	is a time.
	Rating	is the rating.
	Amenity	is an amenity.
	Price	is the price.
	Dish	is a dish.
	Location	is a location.
	Cuisine	is is a cuisine.
	Restaurant_Name	is a restaurant name.
WikiGold	PER	is a person.
	LOC	is a location.
	MISC	is a miscellaneous entity.
	ORG	is an organization.
Weibo	GPE.NAM	是城市、国家的特指。
	GPE.NOM	是城市、国家的泛指。
	LOC.NAM	是地名的特指。
	LOC.NOM	是地名的泛指。
	ORG.NAM	是组织名的特指。
	ORG.NOM	是组织名的泛指。
	PER.NAM	是人名的特指。
	PER.NOM	是人名的泛指。

Table 5: Entity types and their corresponding natural language templates.

Ruleformer: Context-aware Rule Mining over Knowledge Graph

Zezhong Xu[1] , **Peng Ye**[1] , **Hui Chen**[3], **Meng Zhao**[4]
Huajun Chen[1,2], **Wen Zhang**[1*]
[1]Zhejiang University & AZFT Joint Lab for Knowledge Engine, China
[2]Hangzhou Innovation Center, Zhejiang University
[3]Alibaba Group, [4]Huawei
{xuzezhong, yep, huajunsir, zhang.wen}@zju.edu.cn,
weidu.ch@alibaba-inc.com, zhaomeng57@huawei.com

Abstract

Rule mining is an effective approach for reasoning over knowledge graph (KG). Existing works mainly concentrate on mining rules. However, there might be several rules that could be applied for reasoning for one relation, and how to select appropriate rules for completion of different triples has not been discussed. In this paper, we propose to take the context information into consideration, which helps select suitable rules for the inference tasks. Based on this idea, we propose a transformer-based rule mining approach, **Ruleformer**[1]. It consists of two blocks: 1) an encoder extracting the context information from subgraph of head entities with modified attention mechanism, and 2) a decoder which aggregates the subgraph information from the encoder output and generates the probability of relations for each step of reasoning. The basic idea behind Ruleformer is regarding rule mining process as a sequence to sequence task. To make the subgraph a sequence input to the encoder and retain the graph structure, we devise a relational attention mechanism in Transformer. The experiment results show the necessity of considering these information in rule mining task and the effectiveness of our model.

1 Introduction

People built different Knowledge Graphs, such as Freebase (Bollacker et al., 2008), to store complex structured information and knowledge about abstract and real world. The facts in KG are usually represented in the form of triplets, e.g., *(New York, isCityOf, USA)*. KGs have been widely applied in various intelligent systems such as question answering (Yasunaga et al., 2021; Chen et al., 2022), recommender system(Guo et al., 2020; Wong et al., 2021; Zhang et al., 2021) and zero-shot learning(Geng et al., 2022; Chen et al., 2021b).

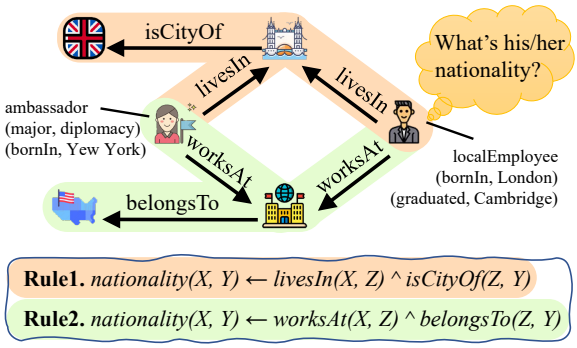

Figure 1: Rules in different context.

Although these KGs already contain a large number of relations and entities, they still suffer from the incompleteness of facts, whether constructed automatically or manually. In order to further expand KGs, many reasoning methods have been proposed to automated fact exploration, such as knowledge graph embedding (KGE) (Bordes et al., 2013; Trouillon et al., 2016; Dettmers et al., 2018; Sun et al., 2019; Zhang et al., 2019b), graph neural networks (Schlichtkrull et al., 2018), and rule mining methods (Ortona et al., 2018; Galárraga et al., 2013; Yang et al., 2017; Sadeghian et al., 2019; Zhang et al., 2019a). Compared with deep learning approaches like KGE, rule mining methods are preferred due to their interpretability for reasoning and robustness for domain knowledge transfer. To mine the structure and confidence of rules at the same time in a fast way, differentiable rule learning methods(Yang et al., 2017; Sadeghian et al., 2019) are introduced and attract many research interests.

The learning targets of existing methods is determining the confidence and structure of rules, according to which reasoning tasks are conducted. However, the uniform way of applying rules ignores the context of specific triplets, for which rules should be applied in different order.

Specifically, for a query $(h, r, ?)$, there might be multiple rules of relation r (not only one) that

* Corresponding Author.
[1]Source code of Ruleformer is available at https://github.com/zjukg/ruleformer.

Proceedings of the 29th International Conference on Computational Linguistics, pages 2551–2560
October 12–17, 2022.

could be used to get the target answer. For a specific query triplet, not all rules of r are convincing for reasoning, but how to select appropriate rule has not been studied. For example, to infer the missing fact, *(*`localEmployee/ambassador`*, nationality, ?)* as Figure 1 illustrated, previous rule mining methods only rely on the head relation *nationality*, so rules of *nationality* will be used in a fixed order. The two queries will get the same results, and whatever the order is, one of answers will be wrong. More precious rules can be observed if we pay attention to the head entity's identity, which is `localEmployee/ambassador`. Based on the observation, when the head entity is a `localEmployee`, the rule *nationality(X, Y)* $\leftarrow$ *livesIn(X, Z)* $\wedge$ *isCityOf(Z, Y)* may derive more accurate result. In another case, if we know the head entity is an `ambassador`, then rule *nationality(X, Y)* $\leftarrow$ *worksAt(X, Z)* $\wedge$ *belongsTo(Z, Y)* will be a better choice for inferring. This example shows that the order and choice of rules to be applied for reasoning will significantly affect the results, and context information could help determine proper order.

In this paper, we investigate making rule mining methods not only learn confidence and structures of rules but also learn to choose suitable rules based on the context of the head entity in completion tasks, which is challenging because of the diversity of context. We propose a Transformer-based model to aggregate the context around the head entity because of its excellent performance on information interaction. Essentially, the model consists of two blocks, 1) an encoder block aggregating the information of the head entity's subgraph and completing the information intersection of context, 2) a decoder block utilizing the aggregated entity embedding to generate the probability of relations for each step in differentiable rule mining process. Since Transformer framework is a sequence to sequence model, we design a converter that can turn graph structure into sequence. Moreover, to maintain the information of subgraph, we modify the attention mechanism of Transformer by adding the relation information to the attention calculation process, which we call relational attention mechanism.

Experimentally, we evaluate our model on several datasets (*UMLS* (Kok and Domingos, 2007), *FB15K-237* (Toutanova et al., 2015), *WN18RR* (Dettmers et al., 2018)) on link prediction and rule parsing task. The improvement in both experiments demonstrate the effectiveness of Ruleformer. A case study is also analyzed, and the case proves our assumption and the ability of Ruleformer to select suitable rules for different triples.

In summary, our contributions are as follows:

- We draw attention to the problem of mining and applying suitable rules depending on the specific context for reasoning task in KG.
- We propose a new model, Ruleformer, that can aggregate the information of subgraph and use the context to support the reasoning process.
- The experiment results prove that our model outperforms existing rule mining methods on link prediction task and rule quality assessment. It successfully selects suitable rules according to the exploitation of context.

2 Related work

2.1 Rule Learning Methods

The problem of learning rules over KG can be seen as a type of statistical relational learning (Koller et al., 2007). AMIE (Galárraga et al., 2013) concentrates on association rule mining with three operations, including dangling atom, instantiated atom and closing atom that add different type of atoms to incomplete rules and uses pre-defined evaluation metrics to prune incorrect rules. AMIE+ (Galárraga et al., 2015) revises the rule extending process and improves evaluation method based on AMIE.

Anyburl (Meilicke et al., 2019) proposed an framework that can mine rules in an effcient way. Based on the randomly sampled path, it replace some entities with variables and get rules.

Rudik (Ortona et al., 2018) can mines positive and negative rules. The positive rules can be used to infers new facts in KG, and the negative rules are useful for other tasks, like detecting erroneous triples.

Generally, conventional symbolic-based rule learning methods are built on effective search strategy, pruning techniques and pre-defined static evaluation indicators. The inference processes are transparent while they may suffer from large search space.

More recently, differentiable rule learning methods based on TensorLog (Cohen, 2016) are proposed, which can learn the confidence and structure of rule at the same time. Neural-LP (Yang et al.,

2017) use RNN to generate the possibilities of different relation for each step, and the parameters can be optimized in a differentiable way. Based on Neural-LP, DRUM (Sadeghian et al., 2019) is proposed, which use low rank approximation to get better results. Neural-Num-LP (Wang et al., 2019) extends Neural-LP to learn the numerical rules. Neural Logic Inductive Learning (Yang and Song, 2020) uses transformer structure to get the non-chain-like rules which To extend the diversity of mined rules.

Whether it's pure-symbolic or neural-symbolic method, none of these models consider the problem of utilizing the information about the head entity during the process of mining or using rules.

2.2 Embedding-based Models

A lot of previous works (Bordes et al., 2013; Trouillon et al., 2016; Dettmers et al., 2018; Sun et al., 2019) concentrate on the embedding-based paradigm. Most of them design a scoring function to get a value for a triplet in the embedding space. Despite their simplicity, embedding-based models achieved good performance on reasoning. These works mostly rely on simple triplets, which means they ignore the environment of the entities and relations.

Some other embedding-based methods consider context information during inferring process. PTransE (Lin et al., 2015) uses the path embedding from the source entity to target entity and the relation embedding to train the model jointly. (Das et al., 2017) further considers entities and entity type in the path. Except concentrating on the path, some methods (Schlichtkrull et al., 2018) utilize information from context with graph neural networks (GNN). HittER (Chen et al., 2021a) uses hierarchical Transformers to make contextualization based on a source entity's neighborhood because they think the network architecture of GNNs is too shallow.

These approaches prove contextual information which plays an important role in reasoning, and this aspires us to consider the environment of source entity in rule mining to some extent. Although the performance on link prediction task is good, these works mostly are based on vector computation, so the symbolic meaning and interpretability are missing while rule mining methods perform better in this regard.

3 Methodology

Knowledge Graph $\mathcal{G}$ is composed by a set of triplets like $\{(e_s, r, e_t)|r \in \mathcal{R}, e_s, \in \mathcal{E}, e_t \in \mathcal{E}\}$, where $\mathcal{E}$ is a countable set of entities and $\mathcal{R}$ is a set of relations, respectively. For the task of rule mining, all the triplets that belong to $\mathcal{G}$ are given to the model, and for each head relation r, the model is supposed to find meaningful rules that are interpretable and understandable for humans. The rule we want is in the following form:

$$r\left(X, Y\right) \leftarrow r_1\left(X, Z_1\right) \wedge ... \wedge r_T\left(Z_{T-1}, Y\right)$$

where T is the length of rule, r and r_i are relations belonging to $\mathcal{R}$, X, Y and Z_i are variables that can be replaced by specific entities and $r(X, Y)$ is a triplet. The triplet on the left of the arrow is called *head* of rule and the right part of the arrow is called rule *body*.

In this section, to provide an intuition about each part of our model, we introduce the details about Ruleformer. Firstly, an encoder is designed for converting the subgraph structure to a sequence, so that the context information of head entity can be input to the Transformer framework with relational attention mechanism which is used to retain the structure information of the graph completely. Then we present how our model is deployed to utilize the output of encoder to support the rule mining process and generate the target sequence that represents rule body. Finally, we propose a case-based rule parsing algorithm to get symbolic rules from the parameters.

3.1 Ruleformer

Now we introduce the subgraph encoding process of the model, and Figure 2 shows the details. The basic idea of our approach is to find rules for the same head relation and let the model has the ability to choose the suitable rule when the contextual environment of the head entity is different.

For a query (h, r, t), we assume that the subgraph around the head entity h in KG contains the potential knowledge needed for understanding the environment, so we first extract the subgraph around h. More precisely, let $\mathcal{S}_k(h)$ be the subgraph which contains the k-hop (shortest distance to h) neighbor of h in the KG and the set of edges connecting these entities.

As we stated before, KG is organized as a graph structure, while Transformer (Vaswani et al., 2017) is a seq2seq model, so we need to transfer

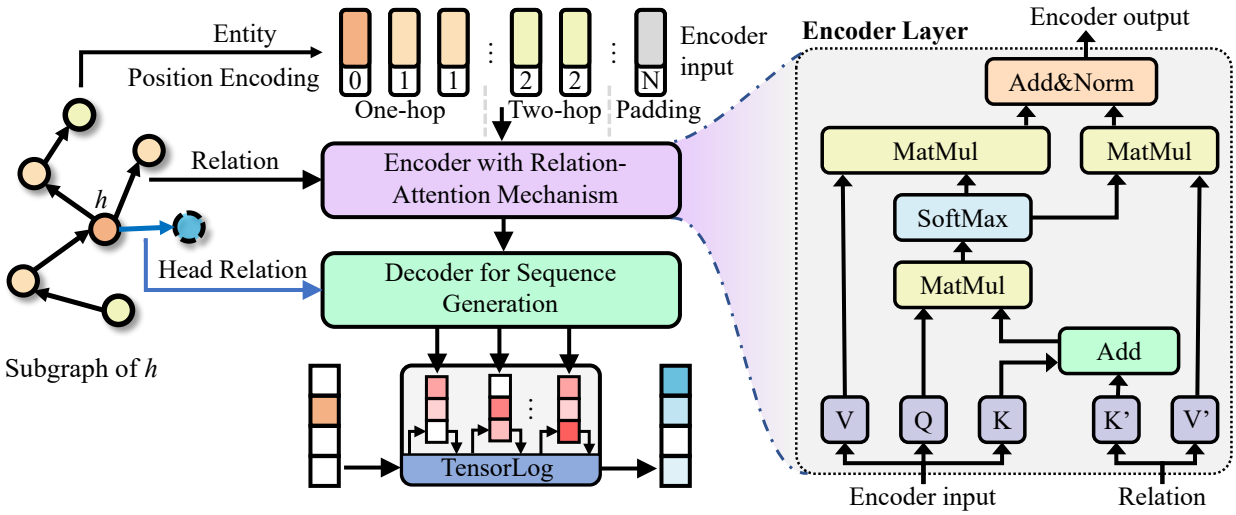

Figure 2: The framework of Ruleformer and details about relational attention mechanism.

the graph to a sequence. Specifically, the nodes in the subgraph are tokenized into a sequence, $S_{node} = [e_1, e_2, ...e_{num}, ...blank]$, where num is the number of entities in the subgraph, and a special token *blank* is needed for padding. As shown in the Figure 2, each node in the subgraph is extracted, and mapped to the initial embedding for entities. The shortest distance to the head entity h is also added as position embedding.

Meanwhile, the distribution of entity occurrence is heavy tailed and hence it is hard to learn appropriate representations for each entity. To alleviate this problem, we consider using the type of relations to help represent the entity. For each relation r, we define two randomly initialized embedding r^{dom} and r^{ran} representing the domain and range embedding of r, which can be learned during the training process. An entity e gets its representation x_e by addition of the type embedding of relations connecting to the entity and a randomly initialized embedding y_e as follows:

$$x_e = \sum_{i=1}^{|\mathcal{R}|} b_i^{dom} r_i^{dom} + \sum_{i=1}^{|\mathcal{R}|} b_i^{ran} r_i^{ran} + y_e \quad (1)$$

where b_i^{dom} and b_i^{ran} are the parameters which are determined by the numbers of different relation types after normalization, and can be given as:

$$b_i^{dom} = \frac{n_i^{dom}}{\sum_{j=1}^{|\mathcal{R}|} n_j^{dom}}, \quad b_i^{ran} = \frac{n_i^{ran}}{\sum_{j=1}^{|\mathcal{R}|} n_j^{ran}} \quad (2)$$

where n_i^{dom} and n_i^{ran} is the number of each type relation connecting to and connected from e, specifically.

Thus, the node sequence S_{node} in mapped to a sequence of embedding $S_e = [x_1, x_2, ...x_{num}, ...blank]$. Note that the above steps only use the entities in the subgraph without the edges between them. In order to utilize the edge information, we introduce relational attention mechanism. Different from the basic way of attention calculation in Transformer, we modify the steps to compute the attention a_{ij} between entity i and j with relations connectingin the following way:

$$a_{ij} = \frac{(x_i W^Q)(x_j W^K + \sum_{r=1}^{|\mathcal{R}|} k_r x_r W^{K'})}{\sqrt{d_k}} \quad (3)$$

where x_i and x_j are the embedding of entity i and j, respectively. x_r is the embedding for relation r which is different from r^{dom} and r^{ran}. W^Q and W^K are the query and key matrices for entity, while $W^{K'}$ is the key matrix for relation. d_k is the dimension of W^K. k_r is defined as $k_r = 1$ if $r(e_i, e_j) \in \mathcal{G}$, else $k_r = 0$.

Then the normalization step for attention is executed:

$$\alpha_{ij} = \frac{\exp a_{ij}}{\sum_{k=1}^{n} \exp a_{ik}} \quad (4)$$

Similarly, we add the relation to the value computation step:

$$z_i = \sum_{j=1}^{n} \alpha_{ij} (x_j W^V + \sum_{r=1}^{|\mathcal{R}|} k_r x_r W^{V'}) \quad (5)$$

where W^V is the value matrix for entity and $W^{V'}$ for relation. By doing this, the information of relations has been inserted into the encoding process.

The relational attention mechanism can ensure sufficient information exchange between each entity because the relations that clearly exist will be emphasized, while the relations that do not appear in the incomplete KG but may be correct will also be reflected because of the interaction process.

With the encoder output sequence S'_e, the decoder block does most of the lifting to aggregate the information together with the head relation. As we mentioned before, the rule mining process can be regarded as a sequence generation problem. For each step, the decoder generates the most suitable relation from $\mathcal{R}$, until the length of decoder output sequence S_r reaches the rule length T.

Specifically, the rule sequence S_r input to decoder starts with the head relations r's embedding x_r, which means $S_r^0 = x_r$. After cross attention calculation with S'_e, the decoder gets a vector which implies the next relation. With this vector, an MLP function is deployed, and we can get the probability ω_t^i of relation r_i in step t, which will be used in the reasoning process. The relation with the highest probability is chosen as the next relation added to S_r.

$$\omega_{t+1} = MLP(CrossAttention(S_r^t, S'_e)) \quad (6)$$

where $\omega_{t+1} \in \mathbb{R}^{|\mathcal{R}|\times 1}$ could be interpreted as the probabilities of all relations in step t. Let r_{t+1} be the relation with the max probability, then $S_r^{t+1} = [S_r^t, x_{r_{t+1}}]$.

We repeat this step T times, and get the complete rule body with length T. Moreover, considering that rule length can't be limited to a definite number T, we add a special relation *self-loop* which connects entities with themselves, and we finally remove this relation from the rule body so that we can get rules with length less than T.

There is still a problem that it's different from the task of machine translation that we don't have labels to judge if the generated relation is an appropriate choice for each step. To overcome this issue, we deploy the Tensorlog framework here to get the prediction results like Neural-LP (Yang et al., 2017) and DRUM (Sadeghian et al., 2019) to indirectly supply supervisory signal. Each entity e_i is represented as a one-hot vector $\mathbf{v}^{e_i} \in \{0,1\}^{|\mathcal{E}|}$, and each relation r_k is represented as an adjacent matrix $\mathbf{M}^{r_k} \in \{0,1\}^{|\mathcal{E}|\times|\mathcal{E}|}$, where $\mathbf{M}_{ij}^{r_k} = 1$ if $r_k(e_i, e_j) \in \mathcal{G}$, else $\mathbf{M}_{ij}^{r_k} = 0$.

Via applying the entity vector and relation matrix, path queries could be answered by expanding path as follows:

$$\mathbf{v}' = \mathbf{v}^{e_i}\mathbf{M}^{r_k} \quad (7)$$

Note that $\mathbf{v}'$ is a multi-hot vector that refers to several entities, which means these entities are connected to e_i via relation r_k.

For step t, the probability ω_t of all relations, which is generated in Equation 6, are used by applying the above step in an indirect way as Equation 8. Let $\mathbf{z}_t \in \mathbb{R}^{|\mathcal{E}|\times 1}$ be the vector representing the probability of all entities in step t, and it is a one-hot vector that represents the head entity if $t = 0$. With $\mathbf{z}_{t-1}$ after $t-1$ step inference, $\mathbf{z}_t$ can be computed as follow:

$$\mathbf{z}_t = \mathbf{z}_{t-1} \times \sum_{i=1}^{|\mathcal{R}|+1} \omega_t^i \mathbf{M}^{r_i} \quad (8)$$

The special relation $self-loop$ with an identity adjacency matrix with $\mathbf{M}^{r_{|\mathcal{R}|+1}} = I_{|\mathcal{E}|}$ is also considered. With this relation, the model can mine rule with length shorter than the length T. ω_t^i represents the probability of the relation r_i as the relation in rules at step t.

Finally, we get the $\mathbf{z}_T$, which is the final result after T steps reasoning. For triplet (h, r, t), the reasoning score is the similarity between the predicted vector $\mathbf{z}_T$ and the target entity vector $\mathbf{v}$:

$$\phi(t|h, r) = \mathbf{v} \cdot log[\mathbf{z}_T, \gamma]_+ \quad (9)$$

where $[\mathbf{x}, \gamma]_+$ denotes the maximum value between each element of $\mathbf{x}$ and γ. The objective function Ruleformer is:

$$\min \left(- \sum_{(h,r,t)\in\mathcal{G}} \phi(t|h, r) \right) \quad (10)$$

3.2 Rule Parsing

To decode symbolic rules from Ruleformer, we propose a rule parsing algorithm using the parameters learned from training process. The basic idea is to select appropriate relations with high weight. With different triplets and the context information fed to the model, Ruleformer may output different parameters, so even for the same relation, the rules mined might be different. Specifically, for a query (h, r, t), we recover possible rules via parameters α. In each step, we choose relations whose weights are over the threshold, and we will check if the entities in the previous step are linked with this relation. By performing this step cyclically until rule

length reaches the max length T, the confidence of each rule is computed by multiplying the weights of relations selected. output symbolic rules with high confidence. Finally, rules that may be useful in reasoning process for a triplet will be output. The detailed procedure is shown in Algorithm 1.

Then we summarize the number occurrences of each rule, and for each time a rule is applied, a confidence score will be calculated with ω and the final score is the average confidence of this rule in different cases. Specifically, a rule set is defined, and for each triplet (h, r, t), we apply Algorithm 1 to parse rules from it. Finally, the score will be calculated for all the rules in R.

Algorithm 1 Decode symbolic logical rules from model

Input: attention $\{\omega_t | t = 1, 2...T\}$ for each triplet
Initialize: $P = \{([P_r], [P_e], w)\}$, $P_r = \emptyset$, $P_e =$ head entity, w represents confidence;
Output: R

for $t = 1 : T$ **do**
  // Scale the attention
  $\omega_t = \omega_t / max(\omega_t)$
  for $([P_r, P_e], w) \in P$ **do**
    for $\omega_t^{r_p} \in \omega_t > thr$ **do**
      // Expand a new path if possible
      for $n \in \mathcal{E}$ can be linked with $P_e[-1]$ via r_p **do**
        // Compute the new confidence
        $w' = w \times \omega_t^{r_p}$
        add $([P_r + r_p], [P_e + n], w')$ to P
      end for
    end for
    // Remove the old path
    remove $([P_r, P_e], w)$ from P
  end for
end for
for $([P_r, P_e], w) \in P$ **do**
  // $R[r, p_r]$ is a list stores confidence scores
  add w to $R[r, p_r]$
end for

4 Experiment

4.1 Dataset

We conduct experiments on three different datasets which are introduced as follows, and Table 2 summarizes the data statistics. We count the average degree for each dataset, because the sparsity has an impact on the choice of subgraph.

- *UMLS* (Kok and Domingos, 2007): Unified Medical Language System, is a knowledge graph that brings together many health and biomedical vocabulary and standards.

- *FB15K-237* (Toutanova et al., 2015): This dataset is a subset from Freebase(Bollacker et al., 2008) and removes the inverse relation. It stores commonsense facts such as topics in movies, actors, awards, etc.

- *WN18RR* (Dettmers et al., 2018): WN18RR is a link prediction dataset created from WN18, which is a subset of WordNet. It's created to ensure that the evaluation dataset does not have inverse relation test leakage.

4.2 Experiment Setup

The experiments are implemented with Pytorch framework and are trained on RTX3090 GPU. ADAM optimizer is used for parameter tuning and the learning rate is set to 0.0001. Our model consists of the encoder and decoder block, which are two-layer Transformer and each layer has 6 heads by default. The dimension of entity is set to 200, as well as the position encoding dimension. We also try 6 heads and 3 layers in Transformer and larger dimension, which result in a little difference. Dropout is applied with a possibility $p = 0.1$. The γ used as threshold is set to be 10^{-20}.

Since the average degree is different for each dataset, to avoid the input sequence of entities being too long, the choice of subgraph is different, too. We extract one-hop subgraph for *FB15K-237*, two-hop subgraph for *WN18RR*. For *UMLS*, the shortest distance is the same as the rule length. Meanwhile, the maximum number of entities which is based on the sparsity of each dataset is different. Considering that relations with a large number of neighbors , like *hasGender*, don't contain much information relative to the huge number of entities, a max number of direct neighbor linked by one relation is set for each entity. For *UMLS*, we set the max number of context entities and max number of neighbors for each entity as 140 and 40 respectively. For *FB15K-237* and *WN18RR*, they are 70 and 40, and 40 and 10 respectively. If the number of context or neighbors exceeds our settings, we randomly select the same number of entities as the setting.

Methods	UMLS				FB15K-237				WN18RR			
	MRR	HIT			MRR	HIT			MRR	HIT		
		@1	@3	@10		@1	@3	@10		@1	@3	@10
TransE	.668	46.8	84.5	93.0	.294	-	-	46.5	.226	-	-	50.1
DistMult	.753	65.1	82.1	93.0	.241	15.5	26.3	41.9	.430	39.0	44.0	49.0
ComplEx	.829	74.8	89.7	96.1	.247	15.8	27.5	42.8	.440	41.0	46.0	51.0
ComplEx-N3	-	-	-	-	.370	-	-	**56.0**	**.480**	-	-	57.0
ConvE	.908	86.2	94.4	98.1	.325	23.7	35.6	50.1	.430	40.0	44.0	52.0
TuckER	-	-	-	-	**.358**	**26.6**	**39.4**	54.4	.470	**44.3**	48.2	52.6
RotatE	**.948**	**91.4**	**98.0**	**99.4**	.338	24.1	37.5	53.3	.476	42.8	**49.2**	**57.1**
PathRank	.197	14.7	25.6	37.6	.087	7.4	9.2	11.2	.189	17.1	20.0	22.5
Neural-LP(T=2)*	.751	63.0	84.7	94.0	.189	12.7	20.6	31.3	.371	35.9	37.4	39.6
Neural-LP(T=3)*	.735	62.7	82.0	92.3	.239	16.0	26.1	39.9	.425	39.4	43.2	49.2
DRUM(T=2)*	.791	64.5	92.7	96.8	.225	17.1	25.4	35.8	.379	36.8	38.5	40.9
DRUM(T=3)*	.784	64.3	91.2	97.2	.328	24.7	36.2	49.9	.441	41.2	45.6	51.6
M-Walk	-	-	-	-	.232	16.5	24.3	-	.437	41.4	44.5	-
Ruleformer(T=2)	.851	73.6	**96.6**	**98.8**	.237	17.4	25.7	36.0	.381	36.6	38.8	41.1
Ruleformer(T=3)	**.857**	**75.2**	95.8	98.4	**.342**	**25.5**	**37.4**	**51.3**	**.452**	**41.7**	**46.5**	**53.0**

Table 1: Link prediction results on dataset *UMLS*, *FB15K-237* and *WN18RR*. Note that for some algorithm, several entities may get the same score. Instead of computing the rank of the right answer as $m + 1$ where m is the number of entities with higher possibilities, we select a random rank for the right answer among the entities with the same possibility. Some methods reported their results in the previous setup in their original paper, and we rerun these methods with the same evaluation process in our way. The results with [*] is reported with our evaluation protocol.

Datasst	UMLS	FB15K-237	WN18RR
#Triplet	5,960	310,116	93,003
#Entity	135	14,541	40,943
#Relation	46	237	11
#Avg.deg	88.3	42.7	4.5

Table 2: Knowledge graph datasets statistics.

4.3 Link Prediction

For the link prediction task, each triplet (h, r, t) in test dataset and its inverse triplet (t, r^{-1}, h) are given to the model with the tail entity masked, and the goal is to predict the masked tail entity among all entities. Each candidate entity in KG will get a score according to the inference result, which is used to sort them, and the ground truth entities are filtered out of the ranking list. Here, we adopt the evaluation metrics the same as previous works (Bordes et al., 2013), Hit@1, Hit@3, Hit@10, and mean reciprocal rank(MRR).

We compare our model with some embedding methods including TransE (Bordes et al., 2013), DisMult (Yang et al., 2015), ConvE (Dettmers et al., 2018), TuckER (Balazevic et al., 2019) and RotatE (Sun et al., 2019), and rule mining methods like PathRank (Lao and Cohen, 2010), Neural-LP (Yang et al., 2017), DRUM (Sadeghian et al., 2019) and M-Walk (Shen et al., 2018). Compared to embedding methods, rule mining methods can provide the interpretability, which is an advantage to pure embedding methods. The results are shown in the Table 1.

As the Table shows, our model outperforms the other rule-based approaches on three datasets. On *UMLS*, the competitive results demonstrates the effectiveness of rule mining approaches. Ruleformer offers an absolute improvement in hit@1 about 10.9% (relatively 16.9%) compared against baseline on *UMLS*. On dataset *FB15K-237* and *WN18RR*, our model also gets the best results on rule mining methods which improves the overall results about 1.4% and 1.1% (relatively 4.3% and 2.3%), respectively. Ruleformer achieves better performance compared to other rule mining proves the effectiveness of our method, and we think this improvement is because other rule-based methods don't consider the head entity and its subgraph and confirms the correctness of our hypothesis.

4.4 Quality Assessment of Minded Rules

In order to have an objective evaluation of the mined rules, we adopt the Standard Confidence (SC) (Galárraga et al., 2013) to assess the rules

mined by different methods. Specifically, the average score of rules is calculated to show the quality. Given a list of rules which is ranked by their confidence calculated by their algorithm that is different from SC, we report the average SC of the top$K$ ($K = 50, 100, 200, 500$) rules. The rules of Neural-LP and DRUM are provided by their original code. The Standard Confidence $SC(\mathbf{B} \rightarrow r(X, Y))$ is calculated as follows:

$$\frac{\#(X, Y) : \exists Z_1 ... Z_m : \mathbf{B} \wedge r(X, Y)}{\#(X, Y) : \exists Z_1, \ldots, Z_m : \mathbf{B}}$$

where $\mathbf{B}$ represents the rule body, X, Y and Z are entity variables, and r is the head relation. This score regards facts that are not in KG as false, in another word, it implements a closed world setting.

Table 3 reports the results. We compare Ruleformer with other differentiable rule mining methods. The main difference between these methods and ours is that Ruleformer could parse the rule according to the environment, but others parse the same rules for each triplet with the same relation. There is a significant improvement in standard confidence with our model and algorithm. Specifically, the standard confidence on *UMLS* improves about 43.7% and on *FB15K-237*, the improvement is 25.5%. This is not only because the model generates suitable rules during training, but also because our parsing algorithm outputs the rules which can be mapped to an existing path in KG.

Methods	UMLS			FB15K-237		
	TOP			TOP		
	50	**100**	**200**	**50**	**200**	**500**
Neural-LP(T=2)	.228	.239	.221	.020	.044	.033
Neural-LP(T=2)	.104	.145	.153	.020	.031	.034
DRUM(T=2)	.400	.350	.303	.058	.036	.048
DRUM(T=3)	.340	.284	.202	.020	.039	.027
Ruleformer(T=2)	**.837**	**.793**	**.740**	.241	**.338**	**.310**
Ruleformer(T=3)	.680	.652	.573	**.313**	.322	.282

Table 3: Average confidence of top ranked rules on datasets *UMLS* and *FB15K-237*. The superscript [2] or [3] means with rule length 2 or 3, respectively.

4.5 Case Study

As we introduced, for different prediction tasks, we hope our model generate suitable rules in a better way with context of head entity into consideration rather than using them in the same order.

To verify the ability of generating appropriate rules of our model, we choose four test triplets from *WN18RR* that are shown in Figure 3.

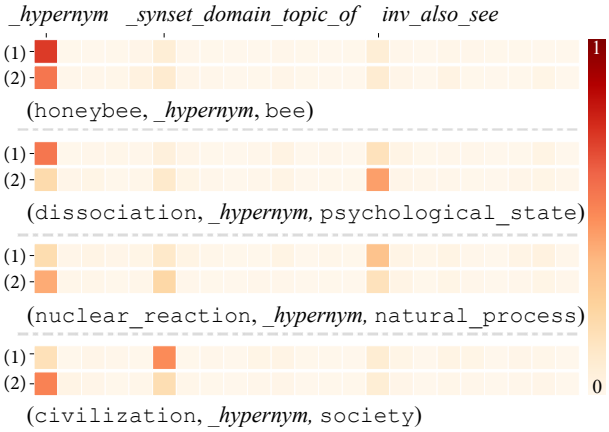

Figure 3: Decode output of four different triplets with the same head relation *_hypernym*. (1) and (2) means the first and second step.

Input these triplets into our model with rule length $T = 2$, and the output contains four sets of data which represent the probability of each relation for two steps in different cases. The heatmap in Figure 3 shows the output. Each row in the figure represents the probability for all relations in a step, and each pair of rows correspond to rules with different scores for the specific triplet which is showed below each subfigure. The darker the color is, the higher probability the relation has for current step. Note that we added inverse relations and *self-loop* so there are more than twice relations in the dataset. The relations with high probability are marked in the top of the figure.

As the figure shows, the four triplets have the same head relation `hypernym` which means a word that is more generic than a given word, while the head entities are different. for different triples with the same relation to be predicted, the probability of relations generated for reasoning in each step are not the same. Take the first case as an example, the rule with the highest confidence is '`hypernym(X, Z)` ← `hypernym(X, Y)` ∧ `hypernym(Y, Z)`'. While for the second example, the rule is '`hypernym(X, Z)` ← `hypernym(X, Y)` ∧ `inv_alsoSee(Y, Z)`' where `inv` means inverse relation. These different but all correct rules shows that our method successfully generate different rules and contextual information of triplets do have an impact on rule reasoning.

5 Conclusion

In this paper, we draw attention to using context to assist rule mining. We regard it as a sequence generation problem, and design a converter turning graph structure into a sequence. We propose a transformer-based model, Ruleformer, which utilizes the subgraph of head entity when learning rules over knowledge graph. The experiment results show that in a specific environment, our proposed model can mine and select different suitable rules. The performance on link prediction task and rule parsing also improves with our model and parsing algorithm. Future work may focus on a more effective way of rule mining and consider more complex forms of rules.

Acknowledgements

This work is funded by NSFCU19B2027/91846204.

References

Ivana Balazevic, Carl Allen, and Timothy M. Hospedales. 2019. Tucker: Tensor factorization for knowledge graph completion. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing, EMNLP-IJCNLP 2019, Hong Kong, China, November 3-7, 2019*, pages 5184–5193. Association for Computational Linguistics.

Kurt Bollacker, Colin Evans, Praveen Paritosh, Tim Sturge, and Jamie Taylor. 2008. Freebase: a collaboratively created graph database for structuring human knowledge. In *Proceedings of the 2008 ACM SIGMOD international conference on Management of data*, pages 1247–1250.

Antoine Bordes, Nicolas Usunier, Alberto Garcia-Duran, Jason Weston, and Oksana Yakhnenko. 2013. Translating embeddings for modeling multi-relational data. In *Advances in neural information processing systems*, pages 2787–2795.

Mingyang Chen, Wen Zhang, Yushan Zhu, Hongting Zhou, Zonggang Yuan, Changliang Xu, and Huajun Chen. 2022. Meta-knowledge transfer for inductive knowledge graph embedding. In *SIGIR*, pages 927–937. ACM.

Sanxing Chen, Xiaodong Liu, Jianfeng Gao, Jian Jiao, Ruofei Zhang, and Yangfeng Ji. 2021a. Hitter: Hierarchical transformers for knowledge graph embeddings. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing, EMNLP 2021, Virtual Event / Punta Cana, Dominican Republic, 7-11 November, 2021*, pages 10395–10407. Association for Computational Linguistics.

Zhuo Chen, Jiaoyan Chen, Yuxia Geng, Jeff Z. Pan, Zonggang Yuan, and Huajun Chen. 2021b. Zero-shot visual question answering using knowledge graph. In *ISWC*, volume 12922 of *Lecture Notes in Computer Science*, pages 146–162. Springer.

William W Cohen. 2016. Tensorlog: A differentiable deductive database. *arXiv preprint arXiv:1605.06523*.

Rajarshi Das, Arvind Neelakantan, David Belanger, and Andrew McCallum. 2017. Chains of reasoning over entities, relations, and text using recurrent neural networks. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics, EACL 2017, Valencia, Spain, April 3-7, 2017, Volume 1: Long Papers*, pages 132–141. Association for Computational Linguistics.

Tim Dettmers, Pasquale Minervini, Pontus Stenetorp, and Sebastian Riedel. 2018. Convolutional 2d knowledge graph embeddings. In *Thirty-Second AAAI Conference on Artificial Intelligence*.

Luis Galárraga, Christina Teflioudi, Katja Hose, and Fabian M Suchanek. 2015. Fast rule mining in ontological knowledge bases with amie+. *The VLDB Journal*, 24(6):707–730.

Luis Antonio Galárraga, Christina Teflioudi, Katja Hose, and Fabian Suchanek. 2013. Amie: association rule mining under incomplete evidence in ontological knowledge bases. In *Proceedings of the 22nd international conference on World Wide Web*, pages 413–422.

Yuxia Geng, Jiaoyan Chen, Wen Zhang, Yajing Xu, Zhuo Chen, Jeff Z. Pan, Yufeng Huang, Feiyu Xiong, and Huajun Chen. 2022. Disentangled ontology embedding for zero-shot learning. In *KDD*, pages 443–453. ACM.

Qingyu Guo, Fuzhen Zhuang, Chuan Qin, Hengshu Zhu, Xing Xie, Hui Xiong, and Qing He. 2020. A survey on knowledge graph-based recommender systems. *IEEE Transactions on Knowledge and Data Engineering*.

Stanley Kok and Pedro M. Domingos. 2007. Statistical predicate invention. In *ICML*, volume 227 of *ACM International Conference Proceeding Series*, pages 433–440. ACM.

Daphne Koller, Nir Friedman, Sašo Džeroski, Charles Sutton, Andrew McCallum, Avi Pfeffer, Pieter Abbeel, Ming-Fai Wong, Chris Meek, Jennifer Neville, et al. 2007. *Introduction to statistical relational learning*. MIT press.

Ni Lao and William W Cohen. 2010. Relational retrieval using a combination of path-constrained random walks. *Machine learning*, 81(1):53–67.

Yankai Lin, Zhiyuan Liu, Huan-Bo Luan, Maosong Sun, Siwei Rao, and Song Liu. 2015. Modeling relation paths for representation learning of knowledge

bases. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing, EMNLP 2015, Lisbon, Portugal, September 17-21, 2015*, pages 705–714. The Association for Computational Linguistics.

Christian Meilicke, Melisachew Wudage Chekol, Daniel Ruffinelli, and Heiner Stuckenschmidt. 2019. Anytime bottom-up rule learning for knowledge graph completion. In *IJCAI*, pages 3137–3143. ijcai.org.

Stefano Ortona, Venkata Vamsikrishna Meduri, and Paolo Papotti. 2018. Robust discovery of positive and negative rules in knowledge bases. In *2018 IEEE 34th International Conference on Data Engineering (ICDE)*, pages 1168–1179. IEEE.

Ali Sadeghian, Mohammadreza Armandpour, Patrick Ding, and Daisy Zhe Wang. 2019. DRUM: end-to-end differentiable rule mining on knowledge graphs. In *Advances in Neural Information Processing Systems 32: Annual Conference on Neural Information Processing Systems 2019, NeurIPS 2019, December 8-14, 2019, Vancouver, BC, Canada*, pages 15321–15331.

Michael Schlichtkrull, Thomas N Kipf, Peter Bloem, Rianne Van Den Berg, Ivan Titov, and Max Welling. 2018. Modeling relational data with graph convolutional networks. In *European Semantic Web Conference*, pages 593–607. Springer.

Yelong Shen, Jianshu Chen, Po-Sen Huang, Yuqing Guo, and Jianfeng Gao. 2018. M-walk: Learning to walk over graphs using monte carlo tree search. In *Advances in Neural Information Processing Systems 31: Annual Conference on Neural Information Processing Systems 2018, NeurIPS 2018, December 3-8, 2018, Montréal, Canada*, pages 6787–6798.

Zhiqing Sun, Zhi-Hong Deng, Jian-Yun Nie, and Jian Tang. 2019. Rotate: Knowledge graph embedding by relational rotation in complex space. In *7th International Conference on Learning Representations*.

Kristina Toutanova, Danqi Chen, Patrick Pantel, Hoifung Poon, Pallavi Choudhury, and Michael Gamon. 2015. Representing text for joint embedding of text and knowledge bases. In *EMNLP*, pages 1499–1509. The Association for Computational Linguistics.

Théo Trouillon, Johannes Welbl, Sebastian Riedel, Éric Gaussier, and Guillaume Bouchard. 2016. Complex embeddings for simple link prediction. In *International Conference on Machine Learning*, pages 2071–2080.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in neural information processing systems*, pages 5998–6008.

Po-Wei Wang, Daria Stepanova, Csaba Domokos, and J Zico Kolter. 2019. Differentiable learning of numerical rules in knowledge graphs. In *International Conference on Learning Representations*.

Chi-Man Wong, Fan Feng, Wen Zhang, Chi-Man Vong, Hui Chen, Yichi Zhang, Peng He, Huan Chen, Kun Zhao, and Huajun Chen. 2021. Improving conversational recommender system by pretraining billion-scale knowledge graph. In *ICDE*, pages 2607–2612. IEEE.

Bishan Yang, Wen-tau Yih, Xiaodong He, Jianfeng Gao, and Li Deng. 2015. Embedding entities and relations for learning and inference in knowledge bases. In *International Conference on Learning Representations*.

Fan Yang, Zhilin Yang, and William W. Cohen. 2017. Differentiable learning of logical rules for knowledge base reasoning. In *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, December 4-9, 2017, Long Beach, CA, USA*, pages 2319–2328.

Yuan Yang and Le Song. 2020. Learn to explain efficiently via neural logic inductive learning. In *8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, April 26-30, 2020*. OpenReview.net.

Michihiro Yasunaga, Hongyu Ren, Antoine Bosselut, Percy Liang, and Jure Leskovec. 2021. QA-GNN: reasoning with language models and knowledge graphs for question answering. In *NAACL-HLT*, pages 535–546. Association for Computational Linguistics.

Wen Zhang, Bibek Paudel, Liang Wang, Jiaoyan Chen, Hai Zhu, Wei Zhang, Abraham Bernstein, and Huajun Chen. 2019a. Iteratively learning embeddings and rules for knowledge graph reasoning. In *WWW*, pages 2366–2377. ACM.

Wen Zhang, Bibek Paudel, Wei Zhang, Abraham Bernstein, and Huajun Chen. 2019b. Interaction embeddings for prediction and explanation in knowledge graphs. In *WSDM*, pages 96–104. ACM.

Wen Zhang, Chi Man Wong, Ganqiang Ye, Bo Wen, Wei Zhang, and Huajun Chen. 2021. Billion-scale pre-trained e-commerce product knowledge graph model. In *ICDE*, pages 2476–2487. IEEE.

Are People Located in the Places They Mention in Their Tweets? A Multimodal Approach

Zhaomin Xiao
University of North Texas
zhaominxiao@my.unt.edu

Eduardo Blanco
University of Arizona
eduardoblanco@arizona.edu

Abstract

This paper introduces the problem of determining whether people are located in the places they mention in their tweets. In particular, we investigate the role of text and images to solve this challenging problem. We present a new corpus of tweets that contain both text and images. Our analyses show that this problem is multimodal at its core: human judgments depend on whether annotators have access to the text, the image, or both. Experimental results show that a neural architecture that combines both modalities yields better results. We also conduct an error analysis to provide insights into why and when each modality is beneficial.

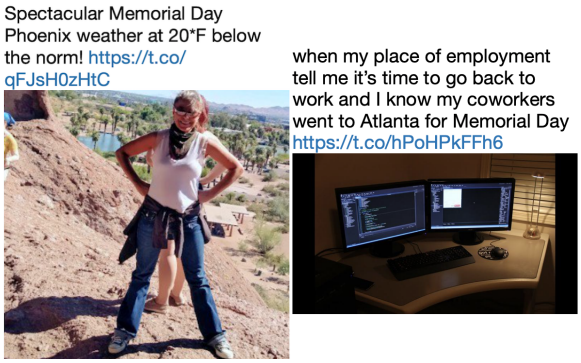

Figure 1: Examples of tweets in which the author is and is not located in the place mentioned in the tweet (*Phoenix* (left) and *Atlanta* (right) respectively).

1 Introduction

Twitter is a social network in which users post short messages known as tweets. While statistics vary depending on the source and publication time, official reports state that 187 million users logged in daily in the third quarter of 2020 (Twitter, 2020), and 500 million tweets were published worldwide on a daily basis in 2014—the last year the number was made public (Twitter, 2014). According to a recent report (Pew Research Center, 2019), 24% of all Americans use Twitter (45% between 18 and 24 years of age), and 46% of them use it at least once a day (26% more than once). Tweets contain not only text (including hashtags, links, emojis, etc.), but also multimedia content such as images and videos. Indeed, 42% of tweets have images (Lee, 2015), and marketing research reveals that having an image improves user engagement: 18% more click throughs, 89% more likes and 150% more retweets (Brandwatch, 2017).

When it comes to noisy user-generated content and spatial information, most previous work falls under two main topics: (a) named entity recognition (Baldwin et al., 2015) and disambiguation (Eshel et al., 2017), and (b) geolocation (Han et al., 2016). The former identifies, among others, location named entities and links them to a knowledge base without specifying who is there. The latter determines one location per user—even if it is not explicitly mentioned. For example, place of residence can be inferred, at least to a certain degree, from the locations of other users and language usage patterns. In this paper, we tackle a complimentary problem: to determine whether people are located in the places they mention in their tweets.

Extracting this kind of spatial information is challenging. First, people often mention places in their tweets even though they are not located there. Second, one must often rely on nuances in both the text and images to make a decision. Consider the tweets in Figure 1. The author of the tweet on the left was in *Phoenix* when the tweet was published. Note that the text alone could arguably be enough to conclude so, but the image provides additional evidence: the background is compatible with the Phoenix area (desert landscape, mountains, etc.), and the person in the picture is (most likely) enjoying the weather there during a short trip for Memorial Day. The author of the tweet on the right, on the other hand, was not in *Atlanta* when the tweet was published. In this example, the image together with the text provides evidence that the

Proceedings of the 29th International Conference on Computational Linguistics, pages 2561–2571
October 12–17, 2022.

author is working rather than enjoying Memorial Day in *Atlanta* with coworkers.

While the work presented here could be considered fundamental research, it opens the door to several applications. For example, emergency management systems could issue customized alerts to individuals who were, are, or are about to be located near a natural disaster. Similarly, eyewitness verification could benefit as the locations of people and the events they claim to witness must be compatible (within some temporal bounds).

The main contributions of this paper are:[1] (a) a corpus of 6,540 tweets with annotations indicating whether the author was in the places mentioned in the tweets; (b) analysis demonstrating that this is a multimodal problem: the ground truth changes depending on whether annotators have access to the text, the image, or both; (c) experimental results showing that taking into account both modalities is beneficial; and (d) qualitative analysis providing insights into (d.1) when are the text and image beneficial, and (d.2) the remaining sources of errors.

1.1 Ethical Considerations

Determining where people are located has the potential to open the door to malicious (or just unwanted) tracking and surveillance. For example, applications that track location data may turn around and sell that data, revealing someone's every movement—whether it is to a retail store, an abortion clinic, or a gay bar. Equally important, Twitter users may not be aware that their tweets can be used for research purposes (Fiesler and Proferes, 2018). We are not interested in tracking people or surveillance. Instead, we are interested in investigating the very definition of the problem and analyzing whether and how language and images complement each other.

In order to alleviate the issues above and preserve privacy, we implemented these safeguards. First, our corpus (a) only contains one tweet per user thus we do not enable user tracking or surveillance. Second, our analyses and experiments only take into account the text and image in a tweet—we do not take into account user information or any metadata. Third, we have designed a takedown request process via an online form following Mirowski et al. (2019).

[1]Corpus and code available at https://github.com/zhaomin1995/coling2022_repo

2 Connections to Related Work

Extracting spatial information from social media and tweets in particular has received substantial attention (Zheng et al., 2018). For example, the tasks of named entity recognition (i.e., identifying, among others, location named entities mentioned in text) and disambiguation (i.e., linking named entities to entries in a knowledge base) have been explored in this noisy user-generated domain (Ritter et al., 2011; Baldwin et al., 2015; Shen et al., 2013; Eshel et al., 2017). Unlike us, these efforts do not aim at determining spatial information about authors of tweets. As we shall see, people often mention places where they are *not* located thus identifying and disambiguating locations tell us what places people tweet about—not the places where they are located when they tweet.

Geolocating twitter users consists in assigning *one location* to a user (e.g., place of residence). Existing corpora calculate the ground truth (i.e., the location for each user) from the geotags attached to tweets. For example, GeoText (Eisenstein et al., 2010) and Twitter-US (Roller et al., 2012) select the geotag of the first geotagged tweet from each user, and Twitter-World (Han et al., 2012) and W-NUT'16 (Han et al., 2016) select the majority city after mapping geotags to city centers. State-of-the-art models take as their input a user's Twitter stream, and combine the text in the tweets, metadata and the social network structure with a neural architecture (Miura et al., 2017; Rahimi et al., 2017, 2018; Do et al., 2018). Unlike the work presented here, geolocating assigns one location per user thus it disregards that people participate in events and as a result their locations change. In this paper, we determine whether people are located in the places they mention in their tweets—even if they only mention the place once and regardless of how long and how often they are there.

More related to our work, Li and Sun (2014) determine whether people have visited, are currently at, or will soon visit points of interest (e.g., monuments, train stations). In their corpus, 47.3% of points of interest are invalid, resulting in little spatial information. More recently, Doudagiri et al. (2018) annotate whether people are located at the locations they tweet about (corpus size: 1,000 tweets), but they do not present experimental results. These two corpora were not publicly available at the time of writing. The work presented here complements these efforts. First, we target any city

mentioned in a tweet, not predefined points of interest. Second, we show that both text and images must be taken into account. Indeed, the ground truth changes depending on which modalities annotators have access to, and experimental results show that models benefit from both modalities. Third, we release a new corpus of 6,540 tweets.

Finally, we note that coupling language and vision has been proposed for, among others, machine translation (Huang et al., 2016) and spatial role labeling (Kordjamshidi et al., 2017). Within social media, some examples include determining the relationship between text and images (Vempala and Preoţiuc-Pietro, 2019), point-of-interest type prediction (Sánchez Villegas and Aletras, 2021), multimodal named entity recognition (Yu et al., 2020), named entity disambiguation (Moon et al., 2018), identifying fake news (Gupta et al., 2013), extracting possessions (Chinnappa et al., 2019), revealing demographic attributes (Sakaki et al., 2014), determining account types (Wijeratne et al., 2016), and detecting user groups (Balasuriya et al., 2016). Our work is inspired by these efforts, but to our knowledge we are the first to target spatial information about authors of tweets using both text and images.

3 A Corpus of Tweets and Spatial Information about the Authors

Our main goal is to understand what kind of spatial information one can infer between authors of tweets and the places they mention in their tweets. To our knowledge, we are the first to tackle this problem, so we create a new corpus. This allows us to explore whether human judgments change depending on whether annotators have access to the text, image or both (Section 4) as well as conduct experiments to automate the task (Section 5).

Collecting tweets We collected 10,000 tweets suitable for our purposes using the criteria below:

1. Each tweet contains both text and an image.
2. The text in each tweet:
   (a) is written in English and has at least five tokens;
   (b) mentions an event that occurred within 14 days of the tweet publication date; and
   (c) mentions a city.

We work with tweets that contain both text and images because we want to explore how spatial information depends on the interpretation of these modalities. We identify the language in which a tweet is written with *langdetect*[2] and spaCy (Honnibal et al., 2020). The list of events we consider include the following: *Christmas*, *Spring Break*, *Thanksgiving*, *Election Day*, *Labor Day*, *Memorial Day*, and *Veteran's Day*. Note that the Twitter search engine does not simply match keywords, thus small variations such as *#veteransday* are also matches. Finally, we use a list of the 100 most populous cities in the U.S.[3] This list includes large cities such as Los Angeles and Chicago as well as smaller cities such as Irving, TX and Richmond, VA (populations below 220,000).

We acknowledge that the events and cities we work with make our corpus US-centric. We believe, however, that the conclusions we reach are not US-centric. In particular, our analyses and experiments are not grounded on the specific events or cities that we work with. A corpus that covers all countries and events—assuming that doing so is possible—is outside the scope of this paper.

Annotation guidelines We aim at capturing spatial information intuitively understood by humans. To this end, we crowdsource human judgments from non-experts by asking a simple question. More specifically, we show crowdworkers one tweet at a time and ask them "Was the author of the tweet located in *city* when the tweet was published?," where *city* is one of the cities identified in the tweet during the collection process. Crowdworkers choose between two options:

- `yes`: the author of the tweet was in *city* when the tweet was published; or
- `no`: I cannot tell if the author of the tweet was in *city* when the tweet was published.

Note that `no` does not guarantee that the author was not in *city*, it rather indicates that the crowdworker cannot establish that the author was in *city*.

3.1 Annotation Process

We crowdsource annotations using Amazon Mechanical Turk. The annotation interface includes instructions and examples. Crowdworkers provide answers to the question above for one (tweet, city) pair before moving to the next one. The interface

[2] https://github.com/Mimino666/langdetect

[3] https://gist.github.com/Miserlou/11500b2345d3fe850c92

displays a screenshot of the tweet as shown on the Twitter's website (desktop version). Doing so ensures that special characters, symbols, and images are displayed properly.

We collected annotations in three independent phases: showing annotators (a) the original tweet (text and image) (b) only the text, and (c) only the image. There was no overlap between the crowdworkers involved in each phase to avoid potential biases. For example, we avoid the possibility that a crowdworker remembers the image in the original version of the tweet when the interface only displays the text. The three annotation phases allow us to analyze whether crowdworkers understand different spatial information if they cannot see the text or image in the original tweet. We created 30,000 annotation tasks (Human Intelligence Tasks in Mechanical Turk parlance; 3 versions per tweet), and crowdsource five annotations for each. The hourly pay ranges from $9 to $13 (the US federal minimum wage is $7.25).

3.2 Annotation Quality

Ensuring annotation quality is critical in any crowdsourcing effort. Our first defense is to recruit crowdworkers located in the United States and with previous approval rate above 95%. Additionally, we do not allow workers to continue working on our tasks if the average completion time per Human Intelligence Task in the past (i.e., the average time spent prior to submitting) is under 3 seconds. We decided on the minimum time required to complete our task based on observations during pilot annotations.

Our second defense is to collect five annotations per Human Intelligent Task and filter out bad annotations until we obtain *substantial* inter-annotator agreement. We do so using Multi-Annotator Competence Estimation (Hovy et al., 2013, MACE) and Krippendorff's α (Krippendorff, 2011). MACE is designed to rank annotators by their competence scores assessing their reliability. The adjudicated labels are determined based on these scores—the most frequent label is not always a good option. Krippendorff's α is a coefficient indicating inter-annotator agreement when several annotators complete different annotation tasks, as is common in crowdsourcing. $\alpha = 0$ indicates only the agreement expected by chance, and $\alpha = 1$ indicates that annotators always agree. Krippendorff's α at or above 0.6 are considered *substantial*, and above 0.8 (nearly) *perfect* (Artstein and Poesio, 2008).

		text		image	
		yes	no	yes	no
text + image	yes	74	26	91	9
	no	72	28	81	19

Table 1: Percentage of label changes depending on the information available to annotators. Many labels change if the text or image is unavailable, especially if the label when both are available is no (72% and 81%).

We ensure $\alpha \geq 0.6$ as follows:

1. Calculate the MACE score of all crowdworkers and sort them by decreasing MACE score.
2. While Krippendorff's $\alpha < 0.6$:
   (a) Drop all the annotations by the crowdworker with the lowest MACE score.
   (b) If a Human Intelligent Task is left without annotations, republish it.

We republish Human Intelligent Tasks (Step 2b) at most twice in order to keep the crowdsourcing costs reasonable. The final corpus consists of 6,540 annotated tweets with Krippendorff's $\alpha = 0.61$. In the rest of this paper, we work with these tweets.

4 Corpus Analysis

The 6,540 tweets in our corpus mention 96 unique cities. The most frequent cities are *Miami* (17% of tweets) and Chicago (6%); other cities account for at most 5% of tweets each. The tweets mention all the events we target (Section 3). The most common event is *Spring Break* (37% of tweets) followed by *Memorial Day* (27%). Other events account for between 5% and 10% of tweets except *Election Day*, which accounts for 3% of tweets.

4.1 Do labels depend on the information available to crowdworkers?

Yes, crowdworkers understand substantially different spatial information depending on whether we show them the original tweet (text and image), the text only, or the image only. The label distribution is as follows for each combination:

- text and image: yes: 51.09%, no: 48.91%
- only text: yes: 80.93%, no: 19.07%
- only image: yes: 69.74%, no: 30.26%

Note that the *right* label (i.e., the ground truth) is the one obtained when crowdworkers have access

Figure 2: Examples of annotations depending on what information is available to annotators (text, image, or both text and image). We only show the adjudicated label after adjudicating the crowdsourced labels. Annotations change substantially (Table 1); the image or text alone often misleads annotators (examples on the left and middle).

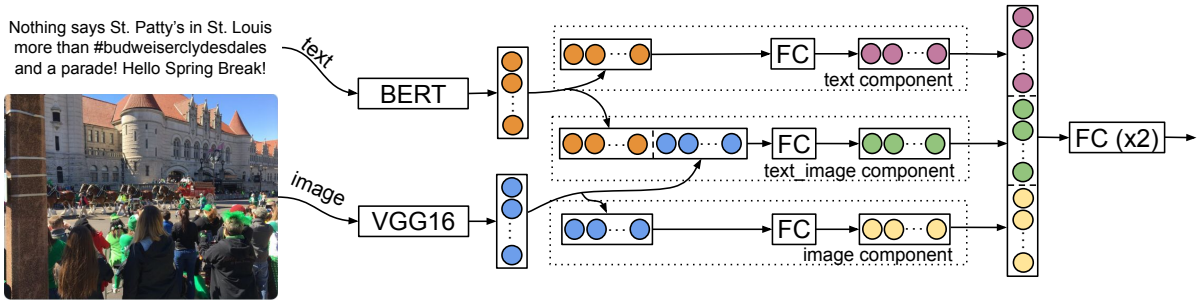

Figure 3: Neural network for determining whether people are located in the places they mention in their tweets. The output layer combines individual representations of the *text* and *image* (top and bottom, *text* and *image component*), and a joint representation of the text and image (middle, *text_image* component).

to both the text and image, as the author chose to publish a tweet including both text and an image.

Table 1 shows that when the right label is `yes` and we only show the text or image, crowdworkers most often do not change the label (74% and 91% respectively). When the right label is `no`, however, it is usually the case that crowdworkers are tricked when they are shown only the text or image (72% and 81% respectively). These percentages demonstrate that both the text and image must be taken into account to determine spatial information about the author of a tweet.

We show examples of annotation changes in Figure 2. In the example on the left, the image alone is insufficient to make any spatial inference between the author of the tweet and *Miami*. Indeed, it is hard to make any connection between *Miami* and the basketball court. The text alone ("Happy Spring Break from Miami"), however, is enough to understand that the author is in *Miami*. The tweet in the middle exemplifies how not having access to the image can trick annotators. When crowdworkers only have access to the text, they understand that the author was in *Raleigh* celebrating Memorial Day. When they are also shown the image, however, they realize that it is an advertisement and do not conclude that the author is in *Raleigh*. The tweet on the right shows an example in which the annotations do not change regardless of whether crowdworkers have access to the text, image, or both. The text indicates that the author is in *Chicago* ("Spring break in the chilly Chicago weather"), and the image also facilitates the same conclusion (cold weather, Cloud Gate in Chicago). Showing both the text and images provides further evidence to conclude that the author was in *Chicago* when the tweet was published.

5 Experiments and Results

Armed with our corpus (Section 3), we conduct experiments to automatically determine whether authors of tweets are located in the cities they mention in their tweets. We reduce the problem to a classification task. The input to the model is a (tweet, city) pair, and the output is a label indicating

	P	R	F1
text	0.65	0.66	0.65
image	0.64	0.65	0.64
text_image	0.62	0.68	0.65
text + image + text_image	**0.64**	**0.74**	**0.68**

Table 2: Results obtained with the full network (text + image + text_image) and individual components. Taking into account the three representations is beneficial.

whether the author of the tweet was located in the city when the tweet was published (`yes` or `no`). We create stratified training and test splits (80% / 20%), and reserve 20% of the training split for validation. If the tweet includes more than one image (it only applies to a handful of tweets), we only feed to the classifier the first image. Our models do not take into account network or user information. They make predictions based exclusively on the content of tweets (the text and image).

Neural Network Architecture We build a neural network consisting of three main components (Figure 3): a component to represent the text (top), a component to represent the image (bottom), and a component to jointly represent the text and image (middle). The three components use pretrained neural networks combined with a trainable fully connected layer to reduce the dimensionality of each representation individually (size: 512). Then, we concatenate the three representations (size: $3 \times 512 = 1536$) and apply two trainable fully connected layers (sizes: 512 and 2) to make the final prediction (`yes` or `no`). We use dropout (Srivastava et al., 2014) in the second-to-last fully connected layer (rate: 0.2). We tried different sizes for the fully connected layers during the tuning process, but we did not observe benefits.

The text component is BERT (Devlin et al., 2019) and the image component is VGG16 (Simonyan and Zisserman, 2014). We use the pretrained models released by HuggingFace (Wolf et al., 2020) and Pytorch (Paszke et al., 2019). We train the neural network for up to 100 epochs using the Adam optimizer (Kingma and Ba, 2014), categorical cross entropy as the loss function, and batch size 8. We stop the training process before 100 epochs if there is no improvement in the validation set for 10 epochs. We implement the neural network with PyTorch (Paszke et al., 2019).

Results Table 2 shows the results with the test split using several variations of the neural network: only the *text* component, only the *image* component, only the *text_image* component, and all of them. We observe that the three components by themselves obtain roughly the same results (F1: 0.64–0.65). Combining the three components, however, yields a slightly higher F1 (0.68), which is mostly due to an increase in Recall (0.74 vs. 0.65–0.68). These results show that the three components of the network are beneficial. In particular, incorporating the individual representations for the *text* and *image* in addition to the joint representation (*text_image*) is beneficial.

6 Qualitative Analysis

To better understand why and when the text and image are most beneficial, we perform a qualitative analysis of the errors made by each model. More specifically, we answer the following questions:

- When does the image complement the text?
- When does the text complement the text?
- When does the task remain challenging?

When does the image complement the text? We start the qualitative analysis providing insights into when is the image beneficial to solve the task. Table 3 exemplifies the most common errors made by the *text* component that are fixed by the full network (text + image + text_image).

The most frequent error that benefits from taking into account the image (38%) occurs when the image (apparently) does not have a connection with the location at hand. Instead, it (visually) depicts some event that (a) occurred in the location at hand and (b) is mentioned in the text. Consider the example on the left (Table 3). The text is about *tornadoes in Miami*, but the image is not a common Miami scene—it shows the destruction caused by the tornado. The text component alone is unable to make the connection, but the full network makes the connection and predicts that the author was in *Miami* when the tweet was published.

The second most common error fixed by the full network (31%) occurs when the tweet is an advertisement and the text component alone wrongly predicts `yes` (e.g., middle tweet in Table 3). In this case, taking into account the image allows the full network to identify the tweet as an advertisement and predict `no`. We note that crowdworkers generally annotate advertisements with `no` unless there is a connection between the author of the tweet and the location (e.g., *My Orlando Chapter Got Something For Ya! [...]*, right tweet in Table 4).

(38%) Image depicts key event	(31%) Advertisements	(14%) Image depicts location
#MIAMI TORNADOES: Some workers are paying it forward as paychecks continue even after businesses remain closed after the Memorial Day storms	Perfect time to gear up for summer travel #tumi #tumitravel #travel #summervacation #springbreak #houstonpremiumoutlets	CORONAVIRUS: The spread of the #coronavirus did not appear to dampen spring break plans for a lot of people at #Jacksonville beach today @ActionNewsJax @WOKVNews
Location: *Miami* Gold: `yes`, Predicted$_{\text{text}}$: `no`	Location: *Houston* Gold: `no`, Predicted$_{\text{text}}$: `yes`	Location: *Jacksonville* Gold: `yes`, Predicted$_{\text{text}}$: `no`

Table 3: Most common errors fixed by the full network compared to the network that only uses the text component.

(46%) Text describes key event	(27%) Text describes location	(10%) Advertisements
Arlington Spring Break Kids Camp Gets Wild https://t.co/qrFqcClRqH https://t.co/Jp5Wh1d2zb	My city better than yours! Period #Miami #MemorialDay https://t.co/0wZ4ibAw7r	🔥My Orlando Chapter Got Something For Ya! ATL Memorial Day Weekend Flash Sale Orlando Members Only!🇺🇸💎 Get Them Before There Gone!😈
Location: *Arlington* Gold: `yes`, Predicted$_{\text{image}}$: `no`	Location: *Miami* Gold: `yes`, Predicted$_{\text{image}}$: `no`	Location: *Orlando* Gold: `yes`, Predicted$_{\text{image}}$: `no`

Table 4: Most common errors fixed by the full network compared to the network that only uses the image component.

The third most common error that benefits from taking into account the image (14%) occurs when the image depicts a typical scene of the location at hand. For example, in the right tweet in Table 3, the picture depicts (presumably) *Jacksonville beach*.

When does the text complement the image? We continue the qualitative analysis providing insights into when is the text beneficial to solve the task. Table 4 exemplifies the most common errors made by the *image* component that are fixed by the full network (text + image + text_image).

The most frequent error (46%) occurs when (a) the image could have been taken in several places and (b) the text describes an event that occurred in the location at hand and is depicted in the image. The tweet on the left (Table 4) exemplifies this scenario. Indeed, the indoor picture could have been taken in many indoor spaces, but it shows an event described in the text (i.e., the Kids Camp).

The second most common error fixed by the full network (27%) occurs when (a) the image is compatible with the location at hand and (b) the text provides further evidence that the author was there. Consider the middle tweet in Table 4. The model that takes into account only the image fails to identify that the author was in *Miami*. Taking into account the text ("My city better than yours! Period #Miami [...]"), however, allows the full network to make the right prediction (`yes`).

The third most common error (10%) addressed

(56%) Missing Information	(24%) Advertisements	(8%) Sentence Fragments
Ever wondered what happens with all those flags you see in Arlington on #MemorialDay? https://t.co/GYHIGdtXaq https://t.co/qwaAC4NPLu	50% Off Memorial Day Fireworks Cruise Tickets at 200 N. Breakwater Access #chicago #memorial #fireworks #cruise #tickets #breakwater #access https://t.co/ElYj2JZiwV https://t.co/r6GaWQFSb6	Knuckle Up: On Today's Episode of Spring Break Miami (VIDEO) - https://t.co/tEJqwwFqBr https://t.co/8bX9juZ3CK 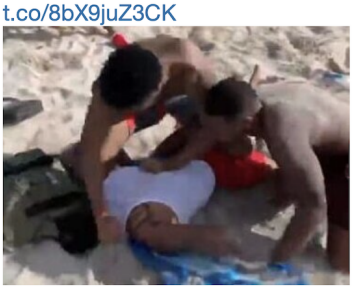
Location: *Arlington* Gold: `no`, Predicted$_{\text{full}}$: `yes`	Location: *Chicago* Gold: `no`, Predicted$_{\text{full}}$: `yes`	Location: *Miami* Gold: `yes`, Predicted$_{\text{full}}$: `no`

Table 5: Most common errors made by the full network (comparing with the ground truth).

when the text is taken into account are again advertisements. As is usual with screenshots and advertisement, the image component alone predicts `no`. Taking into account the text allows the full network to realize that the author most likely was in *Orlando* (*My Orlando Got Something for Ya! [. . .]*).

Which tweets remain challenging? We close the qualitative analysis with the most common errors made by the full network (Table 5). To do so, we look at the errors made by the full network. (text + image + text_image).

The most common error (56%) occurs when (a) neither the text nor image contains enough information to determine whether the author was in the location at hand, and (b) crowdworkers annotated the tweet with `no`. Consider the left tweet in Table 5. Crowdworkers did not indicate that the author was in *Arlington* (`no`), as there is no evidence that the author was there when the tweet was published. We hypothesize that the full network makes a connection between the flags in the image and "all those flags" from the text, and as a result, it predicts `yes`.

The second most common error (24%) are again advertisements. Consider the middle tweet in Table 5. Neither the text or image provide much evidence of the author being in *Chicago*, as indicated by the crowdworkers. The full network, however, predicts `yes`, most likely because it recognizes an urban environment in the picture.

Finally, the full network struggles when the text is not a complete sentence and the connection between text and image is rather nuanced. For example, the text in the tweet on the right (Table 5) is a sentence fragment, and the picture depicts a fight in a beach. The full network is unable to make the connection between (a) *Miami* and "the beach," and (b) the fight and the sentence fragment ("Knuckle Up: On Today's Episode of Spring Break [. . .]").

7 Conclusions

We have introduced the task of determining whether people are located in the places mentioned in their tweets. Going beyond named entity recognition and disambiguation, this problem is about figuring out whether the authors of tweets are located in the places mentioned their tweets. Our new corpus (6,540 tweets) shows that people often mention cities in their tweets even though they are not located there (48.9% of city mentions)—or at least there is not enough evidence in the tweet for crowdworkers to conclude so.

Importantly, we have shown that human judgments change substantially depending on whether crowdworkers have access to the text, the image, or both. These changes in human judgments indicate that when it comes to understanding spatial information about the authors of tweets, the text and images complement each other. To our knowledge, our corpus (Krippendorf's $\alpha = 0.61$) is the first to tackle this challenging problem.

Experimental results show that the task can be automated although our neural network obtains

modest results. In particular, coupling independent representations of the text and image (2 representations) with a joint representation of the text and image yields the best results. These empirical results mirror the observation that human judgments change depending on which modalities crowdworkers have access to. We have also presented a qualitative analysis providing insights into how the image and text complement each other. In summary, they are usually beneficial if they provide additional details about the location at hand or an event that occurred in the location at hand.

Acknowledgements

This research was made possible by funding from the NGA University Research Initiatives (NURI) and a Bloomberg Data Science Research Grant.

References

Ron Artstein and Massimo Poesio. 2008. Inter-coder agreement for computational linguistics. *Comput. Linguist.*, 34(4):555–596.

Lakshika Balasuriya, Sanjaya Wijeratne, Derek Doran, and Amit Sheth. 2016. Finding street gang members on twitter. In *Proceedings of the 2016 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining*, pages 685–692. IEEE Press.

Timothy Baldwin, Marie-Catherine de Marneffe, Bo Han, Young-Bum Kim, Alan Ritter, and Wei Xu. 2015. Shared tasks of the 2015 workshop on noisy user-generated text: Twitter lexical normalization and named entity recognition. In *Proceedings of the Workshop on Noisy User-generated Text*, pages 126–135, Beijing, China. Association for Computational Linguistics.

Brandwatch. 2017. 45 Incredible and Interesting Twitter Statistics. `https://www.brandwatch.com/blog/44-twitter-stats/`. Accessed: December 3, 2018.

Dhivya Chinnappa, Srikala Murugan, and Eduardo Blanco. 2019. Extracting possessions from social media: Images complement language. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 663–672, Hong Kong, China. Association for Computational Linguistics.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, Minneapolis, Minnesota. Association for Computational Linguistics.

Tien Huu Do, Duc Minh Nguyen, Evaggelia Tsiligianni, Bruno Cornelis, and Nikos Deligiannis. 2018. Twitter user geolocation using deep multiview learning. In *2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 6304–6308. IEEE.

Vivek Reddy Doudagiri, Alakananda Vempala, and Eduardo Blanco. 2018. Annotating if the authors of a tweet are located at the locations they tweet about. In *Proceedings of the Eleventh International Conference on Language Resources and Evaluation (LREC-2018)*.

Jacob Eisenstein, Brendan O'Connor, Noah A. Smith, and Eric P. Xing. 2010. A latent variable model for geographic lexical variation. In *Proceedings of the 2010 Conference on Empirical Methods in Natural Language Processing*, pages 1277–1287, Cambridge, MA. Association for Computational Linguistics.

Yotam Eshel, Noam Cohen, Kira Radinsky, Shaul Markovitch, Ikuya Yamada, and Omer Levy. 2017. Named entity disambiguation for noisy text. In *Proceedings of the 21st Conference on Computational Natural Language Learning (CoNLL 2017)*, pages 58–68, Vancouver, Canada. Association for Computational Linguistics.

Casey Fiesler and Nicholas Proferes. 2018. "participant" perceptions of twitter research ethics. *Social Media + Society*, 4(1):2056305118763366.

Aditi Gupta, Hemank Lamba, Ponnurangam Kumaraguru, and Anupam Joshi. 2013. Faking sandy: characterizing and identifying fake images on twitter during hurricane sandy. In *Proceedings of the 22nd international conference on World Wide Web*, pages 729–736. ACM.

Bo Han, Paul Cook, and Timothy Baldwin. 2012. Geolocation prediction in social media data by finding location indicative words. In *Proceedings of COLING 2012*, pages 1045–1062, Mumbai, India. The COLING 2012 Organizing Committee.

Bo Han, Afshin Rahimi, Leon Derczynski, and Timothy Baldwin. 2016. Twitter geolocation prediction shared task of the 2016 workshop on noisy user-generated text. In *Proceedings of the 2nd Workshop on Noisy User-generated Text (WNUT)*, pages 213–217, Osaka, Japan. The COLING 2016 Organizing Committee.

Matthew Honnibal, Ines Montani, Sofie Van Landeghem, and Adriane Boyd. 2020. spaCy: Industrial-strength Natural Language Processing in Python.

Dirk Hovy, Taylor Berg-Kirkpatrick, Ashish Vaswani, and Eduard Hovy. 2013. Learning whom to trust with MACE. In *Proceedings of the 2013 Conference of the North American Chapter of the Association*

for Computational Linguistics: Human Language Technologies, pages 1120–1130, Atlanta, Georgia. Association for Computational Linguistics.

Po-Yao Huang, Frederick Liu, Sz-Rung Shiang, Jean Oh, and Chris Dyer. 2016. Attention-based multimodal neural machine translation. In *Proceedings of the First Conference on Machine Translation*, pages 639–645, Berlin, Germany. Association for Computational Linguistics.

Diederik P. Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. Cite arxiv:1412.6980Comment: Published as a conference paper at the 3rd International Conference for Learning Representations, San Diego, 2015.

Parisa Kordjamshidi, Taher Rahgooy, Marie-Francine Moens, James Pustejovsky, Umar Manzoor, and Kirk Roberts. 2017. Clef 2017: Multimodal spatial role labeling (msprl) task overview. In *Experimental IR Meets Multilinguality, Multimodality, and Interaction*, pages 367–376, Cham. Springer International Publishing.

Klaus Krippendorff. 2011. Computing krippendorff's alpha-reliability. *Retrieved from https://repository.upenn.edu/asc_papers/43*.

Kevan Lee. 2015. What analyzing 1 million tweets taught us. https://thenextweb.com/socialmedia/2015/11/03/what-analyzing-1-million-tweets-taught-us/. Accessed: December 3, 2018.

Chenliang Li and Aixin Sun. 2014. Fine-grained location extraction from tweets with temporal awareness. In *Proceedings of the 37th international ACM SIGIR conference on Research & development in information retrieval*, pages 43–52. ACM.

Piotr Mirowski, Andras Banki-Horvath, Keith Anderson, Denis Teplyashin, Karl Moritz Hermann, Mateusz Malinowski, Matthew Koichi Grimes, Karen Simonyan, Koray Kavukcuoglu, Andrew Zisserman, and Raia Hadsell. 2019. The streetlearn environment and dataset.

Yasuhide Miura, Motoki Taniguchi, Tomoki Taniguchi, and Tomoko Ohkuma. 2017. Unifying text, metadata, and user network representations with a neural network for geolocation prediction. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1260–1272, Vancouver, Canada. Association for Computational Linguistics.

Seungwhan Moon, Leonardo Neves, and Vitor Carvalho. 2018. Multimodal named entity disambiguation for noisy social media posts. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 2000–2008, Melbourne, Australia. Association for Computational Linguistics.

Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Kopf, Edward Yang, Zachary DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. 2019. Pytorch: An imperative style, high-performance deep learning library. In *Advances in Neural Information Processing Systems 32*, pages 8024–8035. Curran Associates, Inc.

Pew Research Center. 2019. Social media use. https://www.pewresearch.org/fact-tank/2019/04/10/share-of-u-s-adults-using-social-media-including-facebook-is-mostly-unchanged-since-2018/.

Afshin Rahimi, Trevor Cohn, and Timothy Baldwin. 2017. A neural model for user geolocation and lexical dialectology. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 209–216, Vancouver, Canada. Association for Computational Linguistics.

Afshin Rahimi, Trevor Cohn, and Timothy Baldwin. 2018. Semi-supervised user geolocation via graph convolutional networks. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 2009–2019, Melbourne, Australia. Association for Computational Linguistics.

Alan Ritter, Sam Clark, Mausam, and Oren Etzioni. 2011. Named entity recognition in tweets: An experimental study. In *Proceedings of the 2011 Conference on Empirical Methods in Natural Language Processing*, pages 1524–1534, Edinburgh, Scotland, UK. Association for Computational Linguistics.

Stephen Roller, Michael Speriosu, Sarat Rallapalli, Benjamin Wing, and Jason Baldridge. 2012. Supervised text-based geolocation using language models on an adaptive grid. In *Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning*, pages 1500–1510, Jeju Island, Korea. Association for Computational Linguistics.

Shigeyuki Sakaki, Yasuhide Miura, Xiaojun Ma, Keigo Hattori, and Tomoko Ohkuma. 2014. Twitter user gender inference using combined analysis of text and image processing. In *Proceedings of the Third Workshop on Vision and Language*, pages 54–61.

Danae Sánchez Villegas and Nikolaos Aletras. 2021. Point-of-interest type prediction using text and images. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 7785–7797, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Wei Shen, Jianyong Wang, Ping Luo, and Min Wang. 2013. Linking named entities in tweets with knowledge base via user interest modeling. In *Proceedings of the 19th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '13, pages 68–76, New York, NY, USA. ACM.

Karen Simonyan and Andrew Zisserman. 2014. Very deep convolutional networks for large-scale image recognition. *CoRR*, abs/1409.1556.

Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. 2014. Dropout: A simple way to prevent neural networks from overfitting. *Journal of Machine Learning Research*, 15(56):1929–1958.

Twitter. 2014. The 2014 #yearontwitter. https://blog.twitter.com/official/en_us/a/2014/the-2014-yearontwitter.html.

Twitter. 2020. Q3 2020 letter to shareholders. https://s22.q4cdn.com/826641620/files/doc_financials/2020/q3/Q3-2020-Shareholder-Letter.pdf.

Alakananda Vempala and Daniel Preoţiuc-Pietro. 2019. Categorizing and inferring the relationship between the text and image of Twitter posts. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 2830–2840, Florence, Italy. Association for Computational Linguistics.

Sanjaya Wijeratne, Lakshika Balasuriya, Derek Doran, and Amit Sheth. 2016. Word embeddings to enhance twitter gang member profile identification. *arXiv preprint arXiv:1610.08597*.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Remi Louf, Morgan Funtowicz, Joe Davison, Sam Shleifer, Patrick von Platen, Clara Ma, Yacine Jernite, Julien Plu, Canwen Xu, Teven Le Scao, Sylvain Gugger, Mariama Drame, Quentin Lhoest, and Alexander Rush. 2020. Transformers: State-of-the-art natural language processing. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 38–45, Online. Association for Computational Linguistics.

Jianfei Yu, Jing Jiang, Li Yang, and Rui Xia. 2020. Improving multimodal named entity recognition via entity span detection with unified multimodal transformer. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 3342–3352, Online. Association for Computational Linguistics.

Xin Zheng, Jialong Han, and Aixin Sun. 2018. A survey of location prediction on twitter. *IEEE Transactions on Knowledge and Data Engineering*, 30(9):1652–1671.

Multi-modal Contrastive Representation Learning for Entity Alignment

Zhenxi Lin[1], **Ziheng Zhang**[1], **Meng Wang**[2], **Yinghui Shi**[3], **Xian Wu**[1], **Yefeng Zheng**[1]
[1]Tencent Jarvis Lab, Shenzhen, China
[2]School of Computer Science and Engineering, Southeast University, Nanjing, China
[3]School of Cyber Science and Engineering, Southeast University, Nanjing, China
{chalerislin,zihengzhang}@tencent.com
{meng.wang,shiyinghui}@seu.edu.cn
{kevinxwu,yefengzheng}@tencent.com

Abstract

Multi-modal entity alignment aims to identify equivalent entities between two different multi-modal knowledge graphs, which consist of structural triples and images associated with entities. Most previous works focus on how to utilize and encode information from different modalities, while it is not trivial to leverage multi-modal knowledge in entity alignment because of the modality heterogeneity. In this paper, we propose MCLEA, a **M**ulti-modal **C**ontrastive **L**earning based **E**ntity **A**lignment model, to obtain effective joint representations for multi-modal entity alignment. Different from previous works, MCLEA considers task-oriented modality and models the inter-modal relationships for each entity representation. In particular, MCLEA firstly learns multiple individual representations from multiple modalities, and then performs contrastive learning to jointly model intra-modal and inter-modal interactions. Extensive experimental results show that MCLEA outperforms state-of-the-art baselines on public datasets under both supervised and unsupervised settings.[1]

1 Introduction

Knowledge graphs (KGs) such as DBpedia (Lehmann et al., 2015) and YAGO (Mahdisoltani et al., 2015) employ the graph structure to represent real-world knowledge, where the concepts are represented as nodes and the relationships among concepts are represented as edges. KGs have been widely applied to knowledge-driven applications to boost their performance, like recommendation system (Cao et al., 2019b), information extraction (Han et al., 2018) and question answering (Lan et al., 2021). In recent years, an increasing amount of knowledge has been represented in multi-modal formats, such as MMKG (Liu et al., 2019) and Richpedia (Wang et al., 2020). These multi-modal KGs usually contain images as the visual modality, like profile photos, thumbnails, or posters. The augmented visual modality has shown the significant capability to improve KG-based applications (Chen et al., 2020a). It was also proven that the incorporation of visual modality can enhance the contextual semantics of entities and also achieve improved KG embeddings (Wang et al., 2021).

Due to the large scope of real-world knowledge, most KGs are often incomplete, and multiple different KGs are usually complementary to one another. As a result, integrating multiple KGs into a unified one can enlarge the knowledge coverage and can also assist in refining KG by discovering the potential flaws (Chen et al., 2020a). To integrate heterogeneous multi-modal knowledge graphs, the task of multi-modal entity alignment (MMEA) is therefore proposed, which aims to discover equivalent entities referring to the same real-world object. Several previous MMEA works have shown that the inclusion of visual modality in modeling helps to improve the performance of entity alignment. For instance, MMEA (Chen et al., 2020a) and EVA (Liu et al., 2021) proposed distinct multi-modal fusion modules to integrate entity representations from multiple modalities into joint embeddings and they achieved state-of-the-art performance.[2] However, these methods mainly utilize existing representations from different modalities, and customized representation learning for EA is not fully explored. In addition, existing methods only explore the use of diverse multi-modal representations to enhance the contextual embedding of entities, the inter-modal interactions are often neglected in modeling.

To address aforementioned problems, we propose MCLEA, a **M**ulti-modal **C**ontrastive **L**earning based **E**ntity **A**lignment model, which effectively

[1]The source code is available at https://github.com/lzxlin/MCLEA.

[2]To distinguish the model MMEA from the task MMEA, we use EA to denote multi-modal entity alignment for the rest of the paper.

Proceedings of the 29th International Conference on Computational Linguistics, pages 2572–2584
October 12–17, 2022.

integrates multi-modal information into joint representations for EA. MCLEA firstly utilizes multiple individual encoders to obtain modality-specific representations for each entity. The individually encoded information includes neighborhood structures, relations, attributes, surface forms (i.e., entity names), and images. Furthermore, we introduce contrastive learning into EA with intra-modal contrastive loss and inter-modal alignment loss. Specifically, intra-modal contrastive loss aims at distinguishing the embeddings of equivalent entities from the ones of other entities for each modality, thus generating more appropriate representations for EA. Inter-modal alignment loss, on the other hand, aims at modelling inter-modal interactions and reducing the gaps between modalities for each entity. With these two losses, MCLEA can learn discriminative cross-modal latent embeddings and ensure potentially equivalent entities close in the joint embedding space, regardless of the modality. MCLEA is also generic as it can support a wide variety of modalities. Moreover, it combines multiple losses and simultaneously learns multiple objectives using task-dependent uncertainty.

The contributions of this paper are three-fold: (*i*) We propose a novel method, called MCLEA, to embed information from different modalities into a unified vector space and then obtain discriminative entity representations based on contrastive learning for entity alignment. (*ii*) We propose two novel losses to explore intra-modal relationships and inter-modal interactions, ensuring that to-be-aligned entities between different KGs are semantically close with minimum gaps between modalities. (*iii*) We experimentally validate the effectiveness and superiority of MCLEA as it achieves state-of-the-art performance on several public datasets in both supervised and unsupervised settings. The overall results also suggest that our MCLEA is capable of learning more discriminative embedding space for multi-modal entity alignment.

2 Related Work

2.1 Multi-modal Knowledge Graph

While many efforts (Mahdisoltani et al., 2015; Lehmann et al., 2015) have been made to achieve large-scale KGs, there are just a few attempts to enrich KGs with multiple modalities. For example, MMKG (Liu et al., 2019) and Richpedia (Wang et al., 2020) utilized the rich visual resources (mainly images) to construct multi-modal knowledge graphs. Their target was to enrich the KG information via appending sufficient and diverse images to textual entities but they also brought challenges to the KG embedding methods. Unlike traditional KG embedding methods, multi-modal KG embedding methods model the textual and visual modalities at the same time (Zhang et al., 2019; Wang et al., 2021). For example, Zhang et al. (2019) proposed MKHAN to exploit multi-modal KGs with hierarchical attention networks on question answering, and Wang et al. (2021) proposed RSME to selectively incorporate visual information during the KG embedding learning process.

2.2 Multi-modal Entity Alignment

Recent studies for entity alignment mostly focused on exploring the symbolic similarities based on various features, including entity names (Wu et al., 2019; Zhang et al., 2019), attributes (Liu et al., 2020), descriptions (Zhang et al., 2019; Tang et al., 2020) and ontologies (Xiang et al., 2021). Most of them started with transforming entities from different KGs into a unified low-dimensional vector space by translation-based models or graph neural networks and then discovered their counterparts based on the similarity metrics of entity embeddings. Some surveys summarized that additional KG information, if appropriately encoded, could further improve the performance of EA methods (Sun et al., 2020b; Zhang et al., 2020). Some previous attempts even proposed to guide these embedding-based EA models with probabilistic reasoning (Qi et al., 2021). With such findings and the increasing popularity of multi-modal knowledge graphs, how to incorporate visual modality in EA, namely multi-modal entity alignment, has begun to draw research attention but the attempts are limited. The pioneer method PoE (Liu et al., 2019) combined the multi-modal features and measured the credibility of facts by matching the underlying semantics of entities. Afterward, MMEA (Chen et al., 2020a) was proposed to integrate knowledge from different modalities (relational, visual, and numerical) and obtain the joint entity representations. Another method termed EVA (Liu et al., 2021) leveraged visual knowledge and other auxiliary information to achieve EA in both supervised and unsupervised manner. Alternatively, the method HMEA (Guo et al., 2021) modeled structural and visual representations in the hyperbolic space, while Masked-MMEA (Shi et al., 2022)

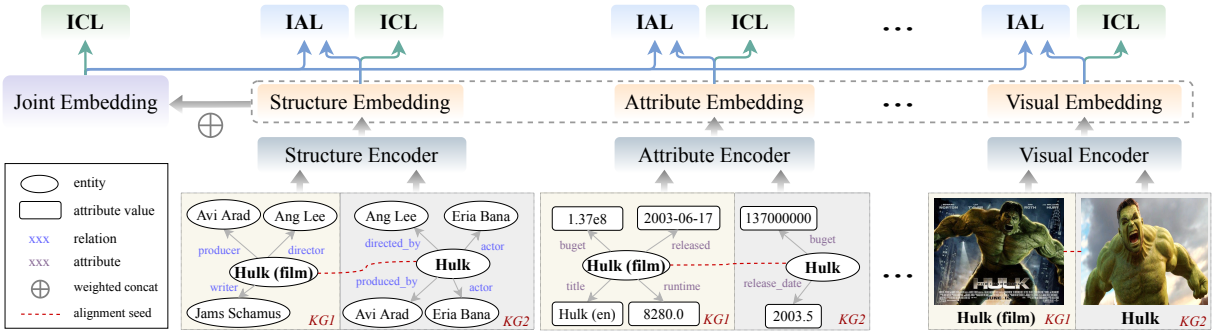

Figure 1: The overall architecture of MCLEA, which combines multiple modalities (§ 3.1) and learns through two proposed losses (§ 3.2), intra-modal contrastive loss (ICL) and inter-modal alignment loss (IAL).

discussed the impacts of visual modality and proposed to incorporate a selectively masking technique to filter potential visual noises. These methods mainly utilize multi-modal representations to enhance the contextual embedding of entities, nevertheless, customized entity representations for EA and inter-modal interactions are often neglected in modeling. Different from previous methods, our proposed MCLEA can learn both intra-modal and inter-modal dynamics simultaneously by the proposed contrastive objectives, expecting to learn more discriminative and abundant entity representation for EA.

3 Proposed Method

We start with the problem formulation and the notations. A multi-modal KG is denoted as $G = (E, R, A, V, T)$, where E, R, A, V, T are the sets of entities, relations, attributes, images, and triples, respectively. Given $G_1 = (E_1, R_1, A_1, V_1, T_1)$ and $G_2 = (E_2, R_2, A_2, V_2, T_2)$ as two KGs to be aligned, the aim of EA is to find aligned entity pairs $\mathcal{A} = \{(e_1, e_2)|e_1 \equiv e_2, e_1 \in E_1, e_2 \in E_2\}$, where we assume a small set of entity pairs $\mathcal{S}$ (seeds) are given as training data. The overall architecture of the proposed MCLEA is shown in Figure 1, and its primary components, multi-modal embeddings, and contrastive representation learning will be detailed in the following sections.

3.1 Multi-Modal Embeddings

Multi-modal KGs often depict various features with multiple modalities (or views), which are complementary to each other. We investigate different embeddings from different modalities for MCLEA, including neighborhood structures, relations, attributes, names (often termed as surface forms in previous work (Liu et al., 2021)), and images. Each modality is processed with an individual encoder network adapted to the nature of the signal. Furthermore, these uni-modal embeddings are aggregated with a simple weighted mechanism to form a joint embedding. Theoretically, MCLEA can support more modalities, e.g., numerical values (Chen et al., 2020a), which will be left in our future work.

3.1.1 Neighborhood Structure Embedding

The graph attention network (GAT) is a typical neural network that directly deals with structured data (Velickovic et al., 2018). Hence, we leverage GAT to model the structural information of G_1 and G_2, shown as "**Structure Encoder**" in Figure 1. Specifically, the hidden state $\boldsymbol{h}_i \in \mathbb{R}^d$ (d is the hidden size) of entity e_i by aggregating its one-hop neighbors $\mathcal{N}_i$ with self-loop is formulated as:

$$\boldsymbol{h}_i = \sigma\left(\sum_{j \in \mathcal{N}_i} \alpha_{ij} \boldsymbol{h}_j\right), \quad (1)$$

where $\boldsymbol{h}_j$ is the hidden state of entity e_j, $\sigma(\cdot)$ denotes the ReLU nonlinearity, and α_{ij} denotes the importance of entity e_j to entity e_i, which is calculated with the self-attention:

$$\alpha_{ij} = \frac{\exp\left(\eta\left(\mathbf{a}^T[\mathbf{W}\boldsymbol{h}_i \oplus \mathbf{W}\boldsymbol{h}_j]\right)\right)}{\sum_{u \in \mathcal{N}_i} \exp\left(\eta\left(\mathbf{a}^T[\mathbf{W}\boldsymbol{h}_i \oplus \mathbf{W}\boldsymbol{h}_u]\right)\right)}, \quad (2)$$

where $\mathbf{W} \in \mathbb{R}^{d \times d}$ denotes the weight matrix, $\mathbf{a} \in \mathbb{R}^{2d}$ is a learnable parameter, $\oplus$ is the concatenation operation and η is the LeakyReLU nonlinearity. Motivated by (Li et al., 2019), we restrict $\mathbf{W}$ to be a diagonal matrix to reduce computations, thus increasing the scalability of the model. To stabilize the learning process of self-attention (Velickovic et al., 2018), we perform K ($K = 2$) heads of independent attention of Eq. (1) in parallel, and concatenate these features to obtain the structure

embedding of entity e_i:

$$\boldsymbol{h}_i^g = \bigoplus_{k=1}^{K} \sigma\left(\sum_{j\in\mathcal{N}_i} \alpha_{ij}^k \boldsymbol{h}_j\right), \tag{3}$$

where α_{ij}^k is the normalized attention coefficient computed by the k-th attention. In practice, we apply a two-layer GAT model to aggregate the neighborhood information within multiple hops, and use the output of the last GAT layer as the neighborhood structure embedding.

3.1.2 Relation, Attribute, and Name Embeddings

Because the vanilla GAT operates on unlabeled graphs, it is unable to properly model relational information in multi-relational KGs. To alleviate this issue, we follow (Yang et al., 2019) and regard the relations of entity e_i as bag-of-words feature and feed it into a simple feed-forward layer to obtain the relation embedding $\boldsymbol{h}_i^r$. For the simplicity and consistency of MCLEA, we adopt the same approach for the attribute embedding $\boldsymbol{h}_i^a$ and the name embedding $\boldsymbol{h}_i^n$ for entity e_i. Therefore, these embeddings are calculated as:

$$\boldsymbol{h}_i^l = \mathbf{W}_l \boldsymbol{u}_i^l + \mathbf{b}_l,\ l \in \{r, a, n\}, \tag{4}$$

where $\mathbf{W}_l$ and $\mathbf{b}_l$ are learnable parameters, $\boldsymbol{u}_i^r$ is the bag-of-words relation feature, $\boldsymbol{u}_i^a$ is the bag-of-words attribute feature, and $\boldsymbol{u}_i^n$ is the name feature obtained by averaging the pre-trained GloVe (Pennington et al., 2014) vectors of name strings. To avoid the out-of-vocabulary challenges brought by the extensive proper nouns (e.g., person names) and the limited vocabulary of word vectors, we further incorporate the character bigrams (Mao et al., 2021) of entity names as auxiliary features for name embedding.

3.1.3 Visual Embedding

We adopt the pre-trained visual model (PVM), e.g., ResNet-152 (He et al., 2016), to learn visual embedding, shown as "**Visual Encoder**" in Figure 1. We feed the image v_i of entity e_i into the pre-trained visual model and use the final layer output before logits as the image feature. The feature is sent through a feed-forward layer to get the visual embedding:

$$\boldsymbol{h}_i^v = \mathbf{W}_v \cdot \mathrm{PVM}(v_i) + \mathbf{b}_v. \tag{5}$$

3.1.4 Joint Embedding

Next, we implement a simple weighted concatenation by integrating the multi-modal features into a single compact representation $\hat{\boldsymbol{h}}_i$ for entity e_i:

$$\hat{\boldsymbol{h}}_i = \bigoplus_{m\in\mathcal{M}} \left[\frac{\exp(w_m)}{\sum_{j\in\mathcal{M}}\exp(w_j)} \boldsymbol{h}_i^m\right], \tag{6}$$

where $\mathcal{M} = \{g, r, a, n, v\}$ and w_m is a trainable attention weight for the modality of m. L2-normalization is performed on the input embeddings before the weighted concatenation.

The current joint embeddings are coarse-grained and there are no interactions between modalities. Therefore, two training strategies are designed for learning the dynamics within (intra-) and between (inter-) modalities.

3.2 Contrastive Representation Learning

As the core of MCLEA, we propose two novel losses on the uni-modal and joint representations to sufficiently capture the dynamics within and between modalities while preserving semantic proximity and minimizing the modality gap.

3.2.1 Intra-modal Contrastive Loss (ICL)

Inspired by recent work on contrastive learning (CL) (Chen et al., 2020b; Khosla et al., 2020), we devise an intra-modal contrastive loss (ICL) that enforces the input embedding to respect the similarity of entities in the original embedding space. Meanwhile, ICL allows MCLEA to distinguish the embeddings of the same entities in different KGs from those of other entities for each modality.

Given that $\mathcal{S}$ can be naturally regarded as positive samples, whereas any non-aligned pairs can be regarded as negative samples due to the convention of 1-to-1 alignment constraint (Sun et al., 2018). Formally, for the i-th entity $e_1^i \in E_1$ of minibatch $\mathcal{B}$, the positive set is defined as $P_i = \{e_2^i | e_2^i \in E_2\}$, where (e_1^i, e_2^i) is an aligned pair. The negative set includes two types, inner-graph unaligned pairs from the source KG G_1 and cross-graph unaligned pairs from the target KG G_2, defined as $N_1^i = \{e_1^j | \forall e_1^j \in E_1, j \neq i\}$ and $N_2^i = \{e_2^j | \forall e_2^j \in E_2, j \neq i\}$, respectively. Both N_1^i and N_2^i come from minibatch $\mathcal{B}$. These two types of negative samples are designed to constrain the joint embedding space, in which the semantically similar entities from the same KG stay closeby and the aligned entities from two KGs map to proximate points. Overall, we define the alignment

probability distribution $q_m(e_1^i, e_2^i)$ of the modality of m for each positive pair (e_1^i, e_2^i) as:

$$q_m(e_1^i, e_2^i) = \frac{\delta_m(e_1^i, e_2^i)}{\delta_m(e_1^i, e_2^i) + \sum_{e_1^j \in N_1^i} \delta_m(e_1^i, e_1^j) + \sum_{e_2^j \in N_2^i} \delta_m(e_1^i, e_2^j)}, \quad (7)$$

where $\delta_m(u, v) = \exp(f_m(u)^T f_m(v)/\tau_1)$, $f_m(\cdot)$ is the encoder of the modality m, and τ_1 is a temperature parameter. Especially, L2-normalization is performed on the input feature embedding before computing the inner product. Notably, the distribution of Eq. (7) is directional and asymmetric for each input; the distribution for another direction is thus defined similarly for $q_m(e_2^i, e_1^i)$. The ICL can be formulated as:

$$\mathcal{L}_m^{\text{ICL}} = -\mathbb{E}_{i \in \mathcal{B}} \log \left[\frac{1}{2} (q_m(e_1^i, e_2^i) + q_m(e_2^i, e_1^i)) \right]. \quad (8)$$

We apply ICL on each modality separately and also on the combined multi-modal representation as specified in Eq. (6). ICL is performed in contrastive supervised learning to learn intra-modal dynamics for more discriminative boundaries for each modality in the embedding space.

3.2.2 Inter-modal Alignment Loss (IAL)

Since the embeddings of different modalities are separately trained with ICL, their representations are not aligned and it is difficult to model the complex interaction between modalities solely with the fusion module. To alleviate this, we further propose inter-modal alignment loss (IAL), which targets at reducing the gap between the output distribution over different modalities, so that the MCLEA can model inter-modal interactions and obtain more meaningful representations.

We consider the joint embedding as the comprehensive representation due to its fusion of multi-modal features; therefore, we attempt to transfer the knowledge from the joint embedding back to uni-modal embedding so that the uni-modal embedding could better utilize the complementary information from others. Concretely, we minimize the bidirectional KL divergence over the output distribution between joint embedding and uni-modal embedding:

$$\begin{aligned} \mathcal{L}_m^{\text{IAL}} = \mathbb{E}_{i \in \mathcal{B}} \frac{1}{2} [&\text{KL}(q_o'(e_1^i, e_2^i) \,||\, q_m'(e_1^i, e_2^i)) \\ &+ \text{KL}(q_o'(e_2^i, e_1^i) \,||\, q_m'(e_2^i, e_1^i))], \end{aligned} \quad (9)$$

where $q_o'(e_1^i, e_2^i)$, $q_o'(e_2^i, e_1^i)$ and $q_m'(e_1^i, e_2^i)$, $q_m'(e_2^i, e_1^i)$ represent the output predictions with two directions of joint embedding and the uni-modal embedding of modality m, respectively. Their calculation are similar to Eq. (7) but with a temperature parameter τ_2. We only back-propagate through $q_m'(e_1^i, e_2^i)$, $q_m'(e_2^i, e_1^i)$ in Eq. (9) as knowledge distillation (Hinton et al., 2015).

IAL aims at learning interactions between different modalities within each entity, which concentrates on aggregating the distribution of different modalities and thus reduces the modality gap. Some approaches (Zhang et al., 2019; Chen et al., 2020a) attempt to learn a common space by imposing alignment constraints on the features between different modalities, but they introduce noises due to semantic heterogeneity. Different from these approaches, we distill the useful knowledge from the output prediction of multi-modal representation to the uni-modal representation, while maintaining relatively modality-specific features of each modality.

3.3 Optimization Objective

The overall loss of the MCLEA is as follows,

$$\mathcal{L} = \mathcal{L}_o^{\text{ICL}} + \textstyle\sum_{m \in \mathcal{M}} \alpha_m \mathcal{L}_m^{\text{ICL}} + \sum_{m \in \mathcal{M}} \beta_m \mathcal{L}_m^{\text{IAL}}, \quad (10)$$

where $\mathcal{M} = \{g, r, a, n, v\}$, $\mathcal{L}_o^{\text{ICL}}$ denotes the ICL operated on joint embedding, α_m and β_m are hyper-parameters that balance the importance of losses. However, manually tuning these hyper-parameters is expensive and intractable. Instead, we treat MCLEA as a multi-task learning paradigm and then use homoscedastic uncertainty (Kendall et al., 2018) to weigh each loss automatically during model training. We adjust the relative weight of each task in the loss function by deriving a multi-task loss function based on maximizing the Gaussian likelihood with task-dependant uncertainty. Due to space limits, we only show the derived result and leave the detailed derivation process in the Appendix. The loss in Eq. (10) can be rewritten as:

$$\mathcal{L} = \mathcal{L}_o^{\text{ICL}} + \sum_{m \in \mathcal{M}} \left(\tfrac{1}{\alpha_m^2} \mathcal{L}_m^{\text{ICL}} + \tfrac{1}{\beta_m^2} \mathcal{L}_m^{\text{IAL}} + \log \alpha_m + \log \beta_m \right), \quad (11)$$

where α_m and β_m are automatically learned during training.

To overcome the lack of training data, we incorporate a bi-directional iterative strategy used in (Liu et al., 2021) to iteratively add new aligned seeds during training. In the inference, we use the cosine similarity metric between joint embeddings of entities to determine the counterparts of entities.

MCLEA can be extended to the unsupervised setting, in which the pseudo-alignment seeds are discovered based on feature similarities of entity names (Mao et al., 2020a; Ge et al., 2021) or entity images (Liu et al., 2021), accordingly resulting in different unsupervised versions of MCLEA.

4 Experiments

4.1 Experimental Setup

Datasets. Five EA datasets are adopted for evaluation, including three bilingual datasets ZH/JA/FR-EN versions of DBP15K (Liu et al., 2021) and two cross-KG datasets FB15K-DB15K/YAGO15K (Liu et al., 2019). The detailed dataset statistics are listed in Table 5 in the Appendix. Note that not all entities have corresponding images and for those without images, MCLEA would assign random vectors for the visual modality, as the setting of EVA (Liu et al., 2021). As for DBP15K, 30% reference entity alignments are given as $\mathcal{S}$ while as for cross-KG datasets, 20%, 50%, and 80% reference entity alignments are given (Liu et al., 2019).

Baselines. We compare the proposed MCLEA against 19 state-of-the-art EA methods, which can be classified into four categories: 1) *structure-based methods* that solely rely on structural information for aligning entities, including BootEA (Sun et al., 2018), MUGNN (Cao et al., 2019a), KECG (Li et al., 2019), NAEA (Zhu et al., 2019), and AliNet (Sun et al., 2020a); 2) *auxiliary-enhanced methods* that utilize auxiliary information to improve the performance, including MultiKE (Zhang et al., 2019), HMAN (Yang et al., 2019), RDGCN (Wu et al., 2019), AttrGNN (Liu et al., 2020), BERT-INT (Tang et al., 2020) and ERMC (Yang et al., 2021); 3) *multi-modal methods* that combine the multi-modal features to generate entity representations, including PoE (Liu et al., 2019), MMEA (Chen et al., 2020a), HMEA (Guo et al., 2021), and EVA (Liu et al., 2021); 4) *unsupervised methods*, including RREA (Mao et al., 2020b), MRAEA (Mao et al., 2020a), EASY (Ge et al., 2021), and SEU (Mao et al., 2021).

Implementation Details. The hidden size of each layer of GAT is 300, while the embedding size of the other modalities is 100. We use the AdamW optimizer with a learning rate of 5×10^{-4} to update the parameters. The number of training epochs is 1000 with early-stopping and the batch size is 512. The hyper-parameters τ_1, τ_2 are set to 0.1 and 4.0, respectively. To keep in line with previous works, we use the same entity name translations and word vectors provided by Xu et al. (2019) for bilingual datasets. As for cross-KG datasets, we do not consider surface forms for fair comparison. For visual embedding, we adopt the preprocessed image features provided by Liu et al. (2021) and Chen et al. (2020a) for bilingual datasets and cross-KG datasets, where the former uses ResNet-152 as the pre-trained backbone, while the latter uses VGG-16. Previous work has revealed that surface forms are quite helpful for entity alignment (Liu et al., 2020). For fair comparison, we divide the supervised methods on bilingual datasets into two groups based on whether surface forms are used, and we implement an MCLEA variant (MCLEA†) where the name embedding is removed. For the unsupervised setting, we implement two variants, MCLEA-V and MCLEA-N, which generate pseudo-alignment seeds based on the similarities of images and names, respectively.

Evaluation. We rank matching candidates of each to-be-aligned entity and use the metrics of Hits@1 (H@1), Hits@10 (H@10), and mean reciprocal rank (MRR). In the following tables, the best results are **in bold** with the second best results underlined, and "Improv. best %" denotes the relative improvement of MCLEA over the best baseline.

4.2 Overall Results

Table 1, Table 2, and Table 3 report the performance of MCLEA against different baselines on different datasets with different settings. Overall, MCLEA and its variants mostly perform the best across all the datasets on all the metrics.

Table 1 reports the performance of MCLEA against the supervised baselines on bilingual datasets in the settings of w/ and w/o surface forms. Compared with the first group without using surface forms, MCLEA† brings about 5.2% to 7.2% relative improvement in H@1 over the best baseline EVA. The superiority of MCLEA confirms that the proposed contrastive representation learning substantially promotes the performance. Specifically, compared with the second group with surface forms involvement, there are two notable observations. On one hand, MCLEA shows a clear improvement when combined with name embedding, suggesting that entity names provide useful clues for entity alignment, which has been revealed in

	Models	DBP15K$_{ZH-EN}$			DBP15K$_{JA-EN}$			DBP15K$_{FR-EN}$		
		H@1	H@10	MRR	H@1	H@10	MRR	H@1	H@10	MRR
w/o SF	BootEA (Sun et al., 2018)	.629	.847	.703	.622	.854	.701	.653	.874	.731
	KECG (Li et al., 2019)	.478	.835	.598	.490	.844	.610	.486	.851	.610
	MUGNN (Cao et al., 2019a)	.494	.844	.611	.501	.857	.621	.495	.870	.621
	NAEA (Zhu et al., 2019)	.650	.867	.720	.641	.873	.718	.673	.894	.752
	AliNet (Sun et al., 2020a)	.539	.826	.628	.549	.831	.645	.552	.852	.657
	EVA (Liu et al., 2021)	.761	.907	.814	.762	.913	.817	.793	.942	.847
	MCLEA† (Ours)	**.816**	**.948**	**.865**	**.812**	**.952**	**.865**	**.834**	**.975**	**.885**
	Improv. best %	7.2	4.5	6.3	6.6	4.3	5.9	5.2	3.5	4.5
w/ SF	MultiKE (Zhang et al., 2019)	.437	.516	.466	.570	.643	.596	.714	.761	.733
	HMAN (Yang et al., 2019)	.562	.851	–	.567	.969	–	.540	.871	–
	RDGCN (Wu et al., 2019)	.708	.846	–	.767	.895	–	.886	.957	–
	AttrGNN (Liu et al., 2020)	.777	.920	.829	.763	.909	.816	.942	.987	.959
	BERT-INT (Tang et al., 2020)	.968	.990	.977	.964	.991	.975	.992	.998	.995
	ERMC (Yang et al., 2021)	.903	.946	.899	.942	.944	.925	.962	.982	.973
	MCLEA (Ours)	**.972**	**.996**	**.981**	**.986**	**.999**	**.991**	**.997**	**1.00**	**.998**
	Improv. best %	0.4	0.6	0.4	2.3	0.8	1.6	0.5	0.2	0.3

Table 1: Comparative results of MCLEA without (w/o) and with (w/) surface forms (SF) against strong supervised methods on three bilingual datasets, and † denotes an MCLEA variant without name embedding.

previous work (Zhang et al., 2019; Liu et al., 2020; Ge et al., 2021). On the other hand, MCLEA still shows slightly better performance than the best baseline BERT-INT with 0.4% to 2.3% relative improvement in H@1 nevertheless with far fewer parameters (13M $vs.$ 110M). This also reveals that MCLEA can effectively model robust entity representations instead of attaching over-parameterized encoders. Noteworthily, BERT-INT relies heavily on entity descriptions to fine-tune BERT, but entity descriptions may not be available for every entity, and collecting them is labor-intensive, limiting the scope of its application.

Table 2 shows the comparison of multi-modal methods on two cross-KG datasets, which provides direct evidence of the effectiveness of MCLEA. When 20% training seeds are given, MCLEA outperforms the best baseline MMEA with 67.9% higher in H@1, 30.3% higher in H@10, and 49.6% higher in MRR. The performance gains are still significant when 50% and 80% alignment seeds are given. It is worth noting that the performance gains reach the highest in the 20% setting and MCLEA (20%) obtains comparable results to EVA (80%), indicating that MCLEA could better utilize the minimum number of alignment seeds to obtain effective representations. We also find that MMEA greatly outperforms EVA, we speculate that the cross-KG datasets are quite heterogeneous (w.r.t. the number of relations) compared to bilingual datasets, as shown in Table 5 in the Appendix, and the structure encoder of EVA struggles to model heterogeneous information and EVA cannot utilize the numerical knowledge in cross-KG datasets, which is well exploited in MMEA.

	Models	FB15K-DB15K			FB15K-YAGO15K		
		H@1	H@10	MRR	H@1	H@10	MRR
20%	PoE	.126	.251	.170	.113	.229	.154
	HMEA	.127	.369	–	.105	.313	–
	MMEA	.265	.541	.357	.234	.480	.317
	EVA∗	.134	.338	.201	.098	.276	.158
	MCLEA (Ours)	**.445**	**.705**	**.534**	**.388**	**.641**	**.474**
	Improv. best %	67.9	30.3	49.6	65.8	33.5	49.5
50%	PoE	.464	.658	.533	.347	.536	.414
	HMEA	.262	.581	–	.265	.581	–
	MMEA	.417	.703	.512	.403	.645	.486
	EVA∗	.223	.471	.307	.240	.477	.321
	MCLEA (Ours)	**.573**	**.800**	**.652**	**.543**	**.759**	**.616**
	Improv. best %	23.5	13.8	22.3	34.7	17.7	26.7
80%	PoE	.666	.820	.721	.573	.746	.635
	HMEA	.417	.786	–	.433	.801	–
	MMEA	.590	.869	.685	.598	**.839**	.682
	EVA∗	.370	.585	.444	.394	.613	.471
	MCLEA (Ours)	**.730**	**.883**	**.784**	**.653**	.835	**.715**
	Improv. best %	9.6	1.6	8.7	9.2	-0.4	4.8

Table 2: Experimental results on two cross-KG datasets where X% represents the percentage of reference entity alignments used for training. The symbol $*$ denotes the reproduced results.

When compared to the unsupervised methods in Table 3, both MCLEA variants perform slightly better than the best baseline with performance gain in H@1 varying from 0.7% to 6.7%. Note that using image (-V) or name (-N) similarities to produce seeds leads to almost identical results, demonstrating the effectiveness of such simple rules to enable MCLEA in the unsupervised setting.

Models	DBP15K$_{ZH-EN}$			DBP15K$_{JA-EN}$			DBP15K$_{FR-EN}$		
	H@1	H@10	MRR	H@1	H@10	MRR	H@1	H@10	MRR
MRAEA (Mao et al., 2020a)	.778	.832	–	.889	.927	–	.950	.970	–
RREA (Mao et al., 2020b)	.822	.964	–	.918	.978	–	.963	.992	–
EASY (Ge et al., 2021)	.898	.979	.930	.943	.990	.960	.980	.998	.990
SEU (Mao et al., 2021)	.900	.965	.924	.956	.991	.969	.988	.999	.992
MCLEA-V (Ours)	.959	**.995**	**.974**	.977	**.999**	.987	.990	**1.00**	.994
MCLEA-N (Ours)	**.960**	.994	**.974**	**.983**	**.999**	**.990**	**.995**	**1.00**	**.997**
Improv. best %	6.7	1.6	4.7	2.8	0.8	2.2	0.7	0.1	0.5

Table 3: Unsupervised experimental results on three bilingual datasets, where -V and -N denote different methods to generate pseudo-alignment seeds.

4.3 Model Analysis

Ablation study. The ablation experiments are performed on two bilingual datasets and the results are presented in Table 4. We first examine the individual contribution of different modalities. The removal of different modalities has varying degrees of performance drop, and entity names have shown the primary importance with the most significant drop, which is in line with the previous findings (Mao et al., 2020a; Ge et al., 2021). The structural information also shows its stable effectiveness across different datasets and other modalities make a slight contribution to MCLEA. Especially, visual information can play a more pronounced role in the absence of surface forms (Chen et al., 2020a; Liu et al., 2021). Furthermore, we inspect various training strategies in MCLEA. It dramatically degrades the performance when removing the ICL from MCLEA, which indicates the importance of ICL in learning the intra-modal proximity. The IAL learns the interdependence between different modalities and is also beneficial to our model. Training MCLEA without the iterative strategy and replacing the uncertainty mechanism with uniform weights (i.e., w/o uncertainty) also cause decreases in performance. Overall, the ablation experiments validate the involvement of these modalities and training strategies with empirical evidence.

Impact of hyper-parameters τ_1, τ_2. We investigate the effects of hyper-parameters τ_1, τ_2 on DBP15K$_{ZH-EN}$. As shown in Figure 2, different values of τ_1 have drastic effects on MCLEA, especially in terms of H@1 and MRR, which is because τ_1 controls the strength of penalties on hard negative samples and an appropriate τ_1 is conducive to learning discriminative entity embeddings. On the other hand, we observe lower variance in the performance w.r.t. τ_2 and the performance saturates when $\tau_2 = 4.0$. The KL divergence establishes the associations between different modalities, within which τ_2 regulates the softness of the alignment distribution produced by input embedding and transfers the generalization capability of the joint embedding to uni-modal embedding.

	Models	DBP15K$_{ZH-EN}$			DBP15K$_{JA-EN}$		
		H@1	H@10	MRR	H@1	H@10	MRR
	MCLEA	**.972**	**.996**	**.981**	**.986**	**.999**	**.991**
Modalities	*w/o* structure	.883	.956	.909	.947	.980	.959
	w/o relation	.967	.995	.978	.985	.999	.991
	w/o attribute	.961	.994	.974	.983	.999	.991
	w/o name	.816	.948	.865	.812	.952	.865
	w/o visual	.968	.994	.978	.985	.999	.991
Training	*w/o* ICL	.782	.892	.818	.813	.909	.844
	w/o IAL	.966	.995	.977	.980	.998	.987
	w/o iter. strategy	.942	.991	.960	.964	.995	.976
	w/o uncertainty	.969	.996	.980	.984	.999	.990

Table 4: Ablation study on two bilingual datasets.

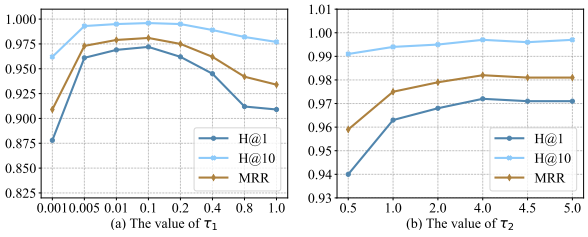

Figure 2: Performance comparison with different values of τ_1, τ_2.

Similarity Distribution of Representations. To investigate the effectiveness of entity representations, we experiment MCLEA with and without ICL/IAL on DBP15K$_{ZH-EN}$ and produce the visualization in Figure 3 by averaging the similarity distribution of the test entities and their predicted counterparts for different modalities. It can be observed that in every modality, especially in structure and name, it holds a high top-1 similarity and a large similarity variance. More importantly, it meets our expectation that contrastive learning

(ICL and IAL) enables more discriminative entity learning in the joint representations.

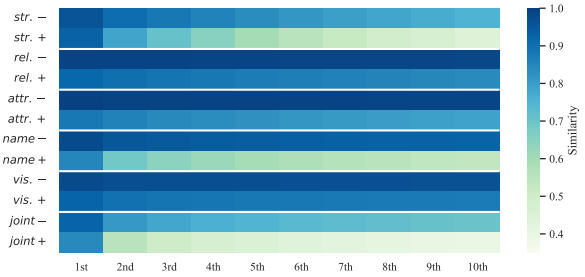

Figure 3: Similarity visualization of representations of test entities and their top-10 predicted counterparts. The vertical axis represents different modalities with (+) and without (−) ICL/IAL and the horizontal axis represents the index of ranked predictions.

5 Conclusion

This paper presented a novel method termed MCLEA to address the multi-modal entity alignment. MCLEA utilizes multi-modal information to obtain the joint entity representations and it is composed of two losses, intra-modal contrastive loss, and inter-modal alignment loss, to explore the intra-modal relationships and cross-modal interactions, respectively. We experimentally validated the state-of-the-art performance of MCLEA in several public datasets and its capability of learning more discriminative embedding space for entity alignment. For future work, we plan to explore more side information such as entity descriptions to boost alignment performance.

References

Yixin Cao, Zhiyuan Liu, Chengjiang Li, Zhiyuan Liu, Juanzi Li, and Tat-Seng Chua. 2019a. Multi-Channel Graph Neural Network for Entity Alignment. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 1452–1461, Florence, Italy. Association for Computational Linguistics.

Yixin Cao, Xiang Wang, Xiangnan He, Zikun Hu, and Tat-Seng Chua. 2019b. Unifying Knowledge Graph Learning and Recommendation: Towards a Better Understanding of User Preferences. In *The World Wide Web Conference, WWW 2019, San Francisco, CA, USA, May 13-17, 2019*, pages 151–161. ACM.

Liyi Chen, Zhi Li, Yijun Wang, Tong Xu, Zhefeng Wang, and Enhong Chen. 2020a. MMEA: Entity Alignment for Multi-modal Knowledge Graph. In *KSEM*.

Ting Chen, Simon Kornblith, Mohammad Norouzi, and Geoffrey E. Hinton. 2020b. A Simple Framework for Contrastive Learning of Visual Representations. In *Proceedings of the 37th International Conference on Machine Learning, ICML 2020, 13-18 July 2020, Virtual Event*, volume 119 of *Proceedings of Machine Learning Research*, pages 1597–1607. PMLR.

Congcong Ge, Xiaoze Liu, Lu Chen, Baihua Zheng, and Yunjun Gao. 2021. Make It Easy: An Effective End-to-End Entity Alignment Framework. In *Proceedings of the 44th International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '21, page 777–786, New York, NY, USA. Association for Computing Machinery.

Hao Guo, Jiuyang Tang, Weixin Zeng, Xiang Zhao, and Li Liu. 2021. Multi-modal Entity Alignment in Hyperbolic Space. *Neurocomputing*, 461:598–607.

Xu Han, Zhiyuan Liu, and Maosong Sun. 2018. Neural Knowledge Acquisition via Mutual Attention Between Knowledge Graph and Text. In *Proceedings of the Thirty-Second AAAI Conference on Artificial Intelligence, (AAAI-18), the 30th innovative Applications of Artificial Intelligence (IAAI-18), and the 8th AAAI Symposium on Educational Advances in Artificial Intelligence (EAAI-18), USA, February 2-7, 2018*, pages 4832–4839. AAAI Press.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. 2016. Deep Residual Learning for Image Recognition. In *2016 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2016, Las Vegas, NV, USA, June 27-30, 2016*, pages 770–778. IEEE Computer Society.

Geoffrey Hinton, Oriol Vinyals, and Jeff Dean. 2015. Distilling the Knowledge in a Neural Network. *ArXiv preprint*, abs/1503.02531.

Alex Kendall, Yarin Gal, and Roberto Cipolla. 2018. Multi-task Learning Using Uncertainty to Weigh Losses for Scene Geometry and Semantics. In *2018 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2018, USA, June 18-22, 2018*, pages 7482–7491. IEEE Computer Society.

Prannay Khosla, Piotr Teterwak, Chen Wang, Aaron Sarna, Yonglong Tian, Phillip Isola, Aaron Maschinot, Ce Liu, and Dilip Krishnan. 2020. Supervised Contrastive Learning. In *Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020, December 6-12, 2020, virtual*.

Yunshi Lan, Gaole He, Jinhao Jiang, Jing Jiang, Wayne Xin Zhao, and Ji-Rong Wen. 2021. A Survey on Complex Knowledge Base Question Answering: Methods, Challenges and Solutions. In *Proceedings of the Thirtieth International Joint Conference on Artificial Intelligence, IJCAI-21*, pages 4483–4491. International Joint Conferences on Artificial Intelligence Organization. Survey Track.

Jens Lehmann, Robert Isele, Max Jakob, Anja Jentzsch, Dimitris Kontokostas, Pablo N Mendes, Sebastian Hellmann, Mohamed Morsey, Patrick Van Kleef, Sören Auer, et al. 2015. DBpedia–A Large-scale, Multilingual Knowledge Base Extracted from Wikipedia. *Semantic Web*, 6(2).

Chengjiang Li, Yixin Cao, Lei Hou, Jiaxin Shi, Juanzi Li, and Tat-Seng Chua. 2019. Semi-supervised Entity Alignment via Joint Knowledge Embedding Model and Cross-graph Model. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 2723–2732, Hong Kong, China. Association for Computational Linguistics.

Fangyu Liu, Muhao Chen, Dan Roth, and Nigel Collier. 2021. Visual Pivoting for (Unsupervised) Entity Alignment. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 35, pages 4257–4266.

Ye Liu, Hui Li, Alberto Garcia-Duran, Mathias Niepert, Daniel Onoro-Rubio, and David S. Rosenblum. 2019. MMKG: Multi-modal Knowledge Graphs. In *The Semantic Web*, pages 459–474, Cham. Springer International Publishing.

Zhiyuan Liu, Yixin Cao, Liangming Pan, Juanzi Li, Zhiyuan Liu, and Tat-Seng Chua. 2020. Exploring and Evaluating Attributes, Values, and Structures for Entity Alignment. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 6355–6364, Online. Association for Computational Linguistics.

Farzaneh Mahdisoltani, Joanna Biega, and Fabian M. Suchanek. 2015. YAGO3: A Knowledge Base from Multilingual Wikipedias. In *CIDR 2015, Seventh Biennial Conference on Innovative Data Systems Research, Asilomar, CA, USA, January 4-7, 2015, Online Proceedings*. www.cidrdb.org.

Xin Mao, Wenting Wang, Yuanbin Wu, and Man Lan. 2021. From Alignment to Assignment: Frustratingly Simple Unsupervised Entity Alignment. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 2843–2853, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Xin Mao, Wenting Wang, Huimin Xu, Man Lan, and Yuanbin Wu. 2020a. MRAEA: An Efficient and Robust Entity Alignment Approach for Cross-lingual Knowledge Graph. In *WSDM '20: The Thirteenth ACM International Conference on Web Search and Data Mining, Houston, TX, USA, February 3-7, 2020*, pages 420–428. ACM.

Xin Mao, Wenting Wang, Huimin Xu, Yuanbin Wu, and Man Lan. 2020b. Relational Reflection Entity Alignment. In *CIKM '20: The 29th ACM International Conference on Information and Knowledge Management, Virtual Event, Ireland, October 19-23, 2020*, pages 1095–1104. ACM.

Jeffrey Pennington, Richard Socher, and Christopher Manning. 2014. GloVe: Global vectors for word representation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1532–1543, Doha, Qatar. Association for Computational Linguistics.

Zhiyuan Qi, Ziheng Zhang, Jiaoyan Chen, Xi Chen, Yuejia Xiang, Ningyu Zhang, and Yefeng Zheng. 2021. Unsupervised Knowledge Graph Alignment by Probabilistic Reasoning and Semantic Embedding. In *Proceedings of the Thirtieth International Joint Conference on Artificial Intelligence, IJCAI-21*, pages 2019–2025. International Joint Conferences on Artificial Intelligence Organization. Main Track.

Yinghui Shi, Meng Wang, Zhang Ziheng, Zhenxi Lin, and Yefeng Zheng. 2022. Probing the Impacts of Visual Context in Multimodal Entity Alignment. In *Web and Big Data - 6th International Joint Conference, APWeb-WAIM 2022*.

Zequn Sun, Wei Hu, Qingheng Zhang, and Yuzhong Qu. 2018. Bootstrapping Entity Alignment with Knowledge Graph Embedding. In *Proceedings of the Twenty-Seventh International Joint Conference on Artificial Intelligence, IJCAI 2018, July 13-19, 2018, Stockholm, Sweden*, pages 4396–4402. ijcai.org.

Zequn Sun, Chengming Wang, Wei Hu, Muhao Chen, Jian Dai, Wei Zhang, and Yuzhong Qu. 2020a. Knowledge Graph Alignment Network with Gated Multi-hop Neighborhood Aggregation. In *AAAI*, volume 34.

Zequn Sun, Qingheng Zhang, Wei Hu, Chengming Wang, Muhao Chen, Farahnaz Akrami, and Chengkai Li. 2020b. A Benchmarking Study of Embedding-based Entity Alignment for Knowledge Graphs. In *Proceedings of the 46th International Conference on Very Large Databases*, pages 2326–2340, Tokyo, Japen.

Xiaobin Tang, Jing Zhang, Bo Chen, Yang Yang, Hong Chen, and Cuiping Li. 2020. BERT-INT: A BERT-based Interaction Model For Knowledge Graph Alignment. In *Proceedings of the Twenty-Ninth International Joint Conference on Artificial Intelligence, IJCAI 2020*, pages 3174–3180. ijcai.org.

Petar Velickovic, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Liò, and Yoshua Bengio. 2018. Graph Attention Networks. In *6th International Conference on Learning Representations, ICLR 2018, Vancouver, BC, Canada, April 30 - May 3, 2018, Conference Track Proceedings*. OpenReview.net.

Meng Wang, Haofen Wang, Guilin Qi, and Qiushuo Zheng. 2020. Richpedia: A Large-scale, Comprehensive Multi-modal Knowledge Graph. *Big Data Research*, 22.

Meng Wang, Sen Wang, Han Yang, Zheng Zhang, Xi Chen, and Guilin Qi. 2021. Is Visual Context Really Helpful for Knowledge Graph? A Representation Learning Perspective. MM '21, page 2735–2743, New York, NY, USA. Association for Computing Machinery.

Yuting Wu, Xiao Liu, Yansong Feng, Zheng Wang, Rui Yan, and Dongyan Zhao. 2019. Relation-Aware Entity Alignment for Heterogeneous Knowledge Graphs. In *Proceedings of the Twenty-Eighth International Joint Conference on Artificial Intelligence, IJCAI 2019, Macao, China, August 10-16, 2019*, pages 5278–5284. ijcai.org.

Yuejia Xiang, Ziheng Zhang, Jiaoyan Chen, Xi Chen, Zhenxi Lin, and Yefeng Zheng. 2021. OntoEA: Ontology-guided Entity Alignment via Joint Knowledge Graph Embedding. In *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*, pages 1117–1128, Online. Association for Computational Linguistics.

Kun Xu, Liwei Wang, Mo Yu, Yansong Feng, Yan Song, Zhiguo Wang, and Dong Yu. 2019. Cross-lingual Knowledge Graph Alignment via Graph Matching Neural Network. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 3156–3161, Florence, Italy. Association for Computational Linguistics.

Hsiu-Wei Yang, Yanyan Zou, Peng Shi, Wei Lu, Jimmy Lin, and Xu Sun. 2019. Aligning Cross-Lingual Entities with Multi-aspect Information. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 4431–4441, Hong Kong, China. Association for Computational Linguistics.

Jinzhu Yang, Ding Wang, Wei Zhou, Wanhui Qian, Xin Wang, Jizhong Han, and Songlin Hu. 2021. Entity and Relation Matching Consensus for Entity Alignment. In *Proceedings of the 30th ACM International Conference on Information & Knowledge Management*, pages 2331–2341.

Qingheng Zhang, Zequn Sun, Wei Hu, Muhao Chen, Lingbing Guo, and Yuzhong Qu. 2019. Multi-view Knowledge Graph Embedding for Entity Alignment. In *Proceedings of the Twenty-Eighth International Joint Conference on Artificial Intelligence, IJCAI 2019, Macao, China, August 10-16, 2019*, pages 5429–5435. ijcai.org.

Ziheng Zhang, Hualuo Liu, Jiaoyan Chen, Xi Chen, Bo Liu, YueJia Xiang, and Yefeng Zheng. 2020. An Industry Evaluation of Embedding-based Entity Alignment. In *Proceedings of the 28th International Conference on Computational Linguistics: Industry Track*, pages 179–189, Online. International Committee on Computational Linguistics.

Qiannan Zhu, Xiaofei Zhou, Jia Wu, Jianlong Tan, and Li Guo. 2019. Neighborhood-Aware Attentional Representation for Multilingual Knowledge Graphs. In *Proceedings of the Twenty-Eighth International Joint Conference on Artificial Intelligence, IJCAI 2019, Macao, China, August 10-16, 2019*, pages 1943–1949. ijcai.org.

Appendix

A Derivation for Adaptively Weighted Multi-task Loss

In this section, we treat Eq. (10) as a multi-task loss function and combine multiple objectives using homoscedastic uncertainty (Kendall et al., 2018), allowing us to automatically learn the relative weights of each loss.

Firstly, the ICL can actually be regarded as a classification loss with negative log-likelihood, i.e., predicting whether two entities are equivalent. Here, we rewrite the loss function of ICL as follows (for simplicity, here we omit the modality index and the inner-graph negative samples, and only consider the unidirectional version):

$$\begin{aligned}\mathcal{L}^{\text{ICL}} &= -\mathbb{E}_{i\in\mathcal{B}} \log q(e_1^i, e_2^i) \\ &= -\mathbb{E}_{i\in\mathcal{B}} \log P\left(c=1|f^{\mathbf{W}}(e_1^i, e_2^i)\right),\end{aligned} \tag{12}$$

where $c=1$ means that the two input entities are equivalent, otherwise $c=0$; $f^{\mathbf{W}}(\cdot,\cdot)$ is the model output with parameter $\mathbf{W}$. Following (Kendall et al., 2018), we adapt the negative log-likelihood to squash a scaled version of the model output with an uncertainty scalar σ through a softmax function:

$$\begin{aligned}&-\log P\left(c=1|f^{\mathbf{W}}(e_1^i, e_2^i), \sigma\right) \\ &= -\log \text{Softmax}\left(\frac{1}{\sigma^2} f^{\mathbf{W}}(e_1^i, e_2^i)\right) \\ &= -\frac{1}{\sigma^2} f^{\mathbf{W}}(e_1^i, e_2^i) + \log \sum_{j\neq i} \exp\left(\frac{1}{\sigma^2} f^{\mathbf{W}}(e_1^i, e_2^j)\right),\end{aligned} \tag{13}$$

where e_2^j with $j \neq i$ is the cross-graph negative samples defined in the main paper.

Applying the same assumption in (Kendall et al., 2018):

$$\begin{aligned}&\frac{1}{\sigma}\sum_{j\neq i} \exp\left(\frac{1}{\sigma^2} f^{\mathbf{W}}(e_1^i, e_2^j)\right) \\ &\approx \left(\sum_{c'} \exp\left(f^{\mathbf{W}}(e_1^i, e_2^j)\right)\right)^{\frac{1}{\sigma^2}},\end{aligned} \tag{14}$$

we can simplify Eq. (13) to:

$$\begin{aligned}&-\log P\left(c=1|f^{\mathbf{W}}(e_1^i, e_2^i), \sigma\right) \\ &\approx -\frac{1}{\sigma^2} f^{\mathbf{W}}(e_1^i, e_2^i) + \frac{1}{\sigma^2}\log\sum_{j\neq i}\exp(f^{\mathbf{W}}(e_1^i, e_2^j)) + \log(\sigma) \\ &= -\frac{1}{\sigma^2}\log P\left(c=1|f^{\mathbf{W}}(e_1^i, e_2^i)\right) + \log(\sigma),\end{aligned} \tag{15}$$

where σ can be interpreted as the relative weight of the loss and automatically learned with stochastic gradient descent.

On the other hand, the IAL defines the KL divergence over the output distribution between joint embedding and uni-modal embedding (we omit the modality index and only consider the unidirectional version for simplicity):

$$\begin{aligned}\mathcal{L}^{\text{IAL}} &= \mathbb{E}_{i\in\mathcal{B}}\ \text{KL}(q'_o(e_1^i, e_2^i) \,||\, q'(e_1^i, e_2^i)) \\ &= \mathbb{E}_{i\in\mathcal{B}}\ q'_o(e_1^i, e_2^i)\log\frac{q'_o(e_1^i, e_2^i)}{q'(e_1^i, e_2^i)} \\ &= \mathbb{E}_{i\in\mathcal{B}}\ [q'_o(e_1^i, e_2^i)\log q'_o(e_1^i, e_2^i) \\ &\quad - q'_o(e_1^i, e_2^i)\log q'(e_1^i, e_2^i)],\end{aligned} \tag{16}$$

where $q'_o(e_1^i, e_2^i)$ and $q'(e_1^i, e_2^i)$ represent the output predictions of joint embedding and the uni-modal embedding, respectively. Since we only back-propagate through $q'(e_1^i, e_2^i)$ in Eq. (9), $\mathcal{L}^{\text{IAL}}$ is equivalent to calculating the cross-entropy loss over the two distributions:

$$\mathcal{L}^{\text{IAL}} = -q'_o(e_1^i, e_2^i)\log q'(e_1^i, e_2^i). \tag{17}$$

Dataset	KG	#Ent.	#Rel.	#Attr.	#Rel tr.	#Attr tr.	#Image	#Ref.
DBP15K$_{ZH-EN}$ (Liu et al., 2021)	ZH	19,388	1,701	8,111	70,414	248,035	15,912	15,000
	EN	19,572	1,323	7,173	95,142	343,218	14,125	
DBP15K$_{JA-EN}$ (Liu et al., 2021)	JA	19,814	1,299	5,882	77,214	248,991	12,739	15,000
	EN	19,780	1,153	6,066	93,484	320,616	13,741	
DBP15K$_{FR-EN}$ (Liu et al., 2021)	FR	19,661	903	4,547	105,998	273,825	14,174	15,000
	EN	19,993	1,208	6,422	115,722	351,094	13,858	
FB15K-DB15K (Liu et al., 2019)	FB15K	14,951	1,345	116	592,213	29,395	13,444	12,846
	DB15K	12,842	279	225	89,197	48,080	12,837	
FB15K-YAGO15K (Liu et al., 2019)	FB15K	14,951	1,345	116	592,213	29,395	13,444	11,199
	YAGO15K	15,404	32	7	122,886	23,532	11,194	

Table 5: Dataset Statistics.

Therefore, similar to ICL, we can automatically learn the relative weight of IAL for each modality through task-dependent uncertainty. As mentioned above, the total loss in Eq. (10) can be rewritten as:

$$\begin{aligned}\mathcal{L} = \mathcal{L}_o^{\text{ICL}} + \sum_{m\in\mathcal{M}} \left(\frac{1}{\alpha_m^2}\mathcal{L}_m^{\text{ICL}} + \log \alpha_m\right) \\ + \sum_{m\in\mathcal{M}} \left(\frac{1}{\beta_m^2}\mathcal{L}_m^{\text{IAL}} + \log \beta_m\right),\end{aligned} \tag{18}$$

where α_m, β_m are learnable parameters. Large α_m (β_m) will decrease the contribution of $\mathcal{L}_m^{\text{ICL}}$ ($\mathcal{L}_m^{\text{IAL}}$) for the m-th modality, whereas small α_m (β_m) will increase its contribution.

B Dataset Statistics

The detailed dataset statistics are listed in Table 5, including the number of entities (#Ent.), relations (#Rel.), attributes (#Attr.), number of relation triples (#Rel tr.) and attribute triples (#Attr tr.), number of images (#Image), and number of reference entity alignments (#Ref.). It is worth noting that not all entities have the associated images or the equivalent counterparts in the other KG.

Nonparametric Forest-Structured Neural Topic Modeling

Zhihong Zhang[*], **Xuewen Zhang**[*], **Yanghui Rao**[†]
School of Computer Science and Engineering, Sun Yat-sen University, Guangzhou, China
{zhangzhh33, zhangxw53}@mail2.sysu.edu.cn, raoyangh@mail.sysu.edu.cn

Abstract

Neural topic models have been widely used in discovering the latent semantics from a corpus. Recently, there are several researches on hierarchical neural topic models since the relationships among topics are valuable for data analysis and exploration. However, the existing hierarchical neural topic models are limited to generate a single topic tree. In this study, we present a nonparametric forest-structured neural topic model by firstly applying the self-attention mechanism to capture parent-child topic relationships, and then build a sparse directed acyclic graph to form a topic forest. Experiments indicate that our model can automatically learn a forest-structured topic hierarchy with indefinite numbers of trees and leaves, and significantly outperforms the baseline models on topic hierarchical rationality and affinity.

1 Introduction

Topic model has been widely used in modeling a collection of documents and encoding the text content to a low dimensional feature space. Traditional topic models can be divided into probabilistic graphical models and matrix factorization based methods. Probabilistic graphical models, such as Latent Dirichlet Allocation (LDA) (Blei et al., 2003b), rely on approximate approaches (e.g., variational inference and Gibbs sampling) with complex derivation or high computational costs to estimate parameters (Srivastava and Sutton, 2017). Matrix factorization based methods (Lee and Seung, 1999) can effectively decompose the document-word representation into two sub-matrices but are subject to a low stability (Chen et al., 2021b). Recently, neural topic models based on Neural Variational Inference (NVI) (Srivastava and Sutton, 2017; Miao et al., 2017; Chen et al., 2021b) have attracted great attention owing to the advantages of fast parameter inference and flexibility.

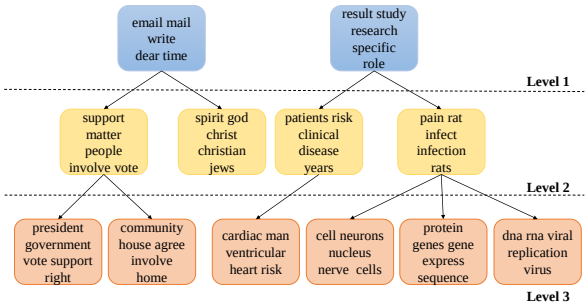

Figure 1: An example of forest-structured topics. Each topic is represented by 5 top words.

Despite some notable successes for neural topic models, most of the existing methods can only extract topics at the same level. This may cause confusions on identifying the hierarchical structure about the relationships among topics, which is valuable for data analysis and exploration in various domains (Paisley et al., 2015). To address this, a few neural topic models have been developed to model the hierarchical structure of topics (Isonuma et al., 2020; Chen et al., 2021b,a). The above methods, however, all assume that the hierarchical structure is a tree with a single root node. This is a significant limitation because for a real-world corpus, topics can be organized into several trees, where the structure of each tree is independent. As an illustration, Figure 1 shows two root topics on email and study. Topics at level 2 include email contents (i.e., politics and religion) and those describing the mode of study (i.e., patients and rats). Some of the topics at level 2 have several children topics which are very specific to distinguish the scope of politics (e.g., country and community) or focus of biology (e.g., cell, protein and genes), and others have only one child topic or none. It indicates that the real-world hierarchical topic structure is more likely to be a forest rather than a tree.

The forest-structured topic models still face challenges since the hierarchical structure of topics

[*]The first two authors contributed equally to this work.
[†]The corresponding author.

Proceedings of the 29th International Conference on Computational Linguistics, pages 2585–2597
October 12–17, 2022.

needs to be (1) rational—root topics are general and children topics are specific to their corresponding parent topics (Viegas et al., 2020); (2) affinitive—each topic is more similar to their children topics than topics from other parents (Kim et al., 2012); (3) diverse—the topic-word distributions associated with parents and children are distinguishable (Blei et al., 2003a); (4) flexible—children of each topic are automatically assigned (Kim et al., 2012) and topic numbers at each level are unbounded (i.e., nonparametric) (Chen et al., 2021b). Besides, the number of root topics, depth, and width of a forest structure are hard to be pre-defined.

In this work, we propose a nonparametric Forest-structured Neural Topic Model (nFNTM) to tackle these challenges, which firstly captures parent-child topic relationships based on the self-attention mechanism (Vaswani et al., 2017)[1], and then learns a sparse Directed Acyclic Graph (DAG) to build a topic forest by federating document-topic distributions and parent-child relationships based on NVI. To our best knowledge, the current topic models with a DAG structure are based on Bayesian learning (Li and McCallum, 2006; Mimno et al., 2007) or Non-negative Matrix Factorization (NMF) (Liu et al., 2018; Viegas et al., 2020), and there is no work under the NVI framework. To sum up, the main contributions are summarized as follows:

- We are the first to introduce the sparse DAG into neural topic modeling with the aim of learning a forest-structured topic hierarchy.
- We develop a self-attention mechanism to capture the relationships among topics.
- We evaluate nFNTM on three benchmark datasets. Empirical results indicate that our model significantly outperforms baselines.

2 Related Work

Traditional topic models, such as LDA (Blei et al., 2003b), are powerful tools for modeling text in an unsupervised fashion, while they lack the exploration of the relationship among topics. To overcome this issue, a tree-structured topic model named hLDA (Blei et al., 2003a) was first proposed. In hLDA, the nested Chinese Restaurant Process (nCRP) was used to generate a topic tree. To alleviate the single-path constraint assumed by nCRP, a nested Hierarchical Dirichlet Process (HDP), i.e., nHDP (Paisley et al., 2015) was developed, which provided the ability of cross-thematic borrowing while keeping general topic areas in separate subtrees. The nested Chinese Restaurant Franchise (nCRF) process developed in (Ahmed et al., 2013) combined the advantages of HDP (Teh et al., 2004) and nCRP. In (Kim et al., 2012), the recursive Chinese Restaurant Process (rCRP) was proposed to discover a hierarchical topic structure with unbounded depth and width.

These models based on the Chinese restaurant process can be effectively employed to discover the hierarchical topic structure by Bayesian learning, but the posterior inference method requires a high computational cost. The scalability of NMF-based methods (Liu et al., 2018; Viegas et al., 2020) is also quite limited. There is a new direction to build tree-structured topic models based on NVI due to its advantages of fast parameter inference and flexibility. A tree-structured neural topic model (TSNTM) (Isonuma et al., 2020) was proposed, which applied doubly-recurrent neural networks to parameterize topic distributions over a tree. But it lacked the ability of learning appropriate semantic embeddings for topics and relied on heuristic rules to update the tree structure. A nonparametric tree-structured neural topic model (nTSNTM) (Chen et al., 2021b) tackled these weaknesses by directly sampling the leaf topics and generating the paths from bottom up automatically. nTSNTM used a common stick-breaking construction to infer topic distributions from the leaf nodes to the root node and applied dependency matrices to keep track of the affiliations among topics. However, the dependency matrices which determine the topic hierarchy are neural weights between the network layers. It results in the structure (i.e., depth and width) of the tree can only be set in advance.

3 Methodology

3.1 Model Architecture

Our nFNTM consists of an encoder, a topic attention, and a decoder, as shown in Figure 2.

3.1.1 Encoder

Given a collection of documents, each document $\boldsymbol{d} \in \mathbb{R}^V$ is represented by Bag-of-Words (BoW), where V is the vocabulary size. For the encoder,

[1]We use the self-attention mechanism as it is effective to draw the global dependencies between input tokens with little reliance on the external information (Yao et al., 2021). Furthermore, it allows efficient computation by parallelization.

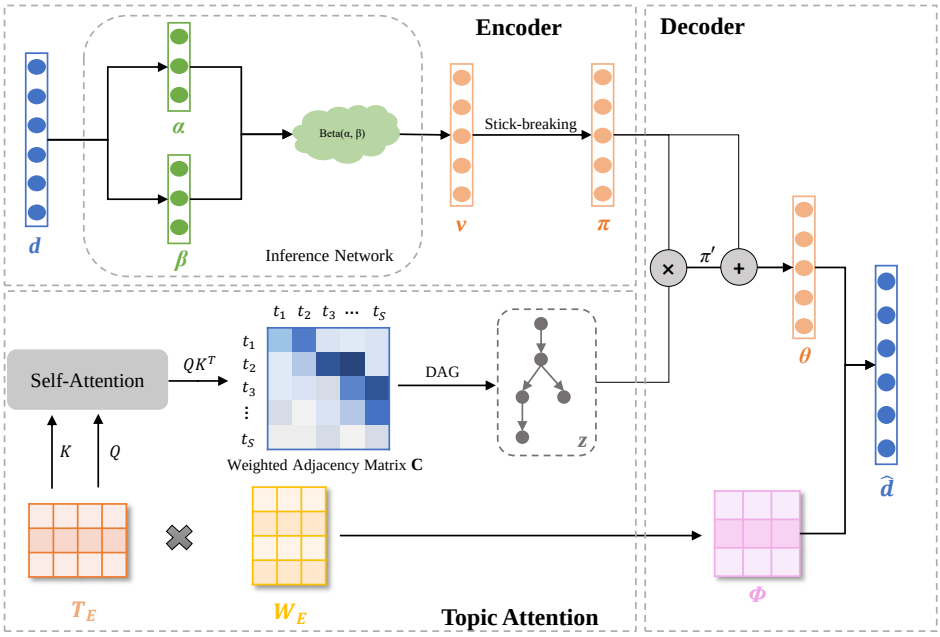

Figure 2: Structure of nFNTM.

we transform $\boldsymbol{d}$ into document-topic distribution $\boldsymbol{\pi}$ by the Stick-Breaking Process (SBP) (Ishwaran and James, 2001), which provides a solution to define atomic measures associated Bayesian nonparametric methods. Any almost sure (a.s.) discrete probability measure $\mathcal{P}$ is an SBP if it can be represented by:

$$\mathcal{P} = \sum_{i=1}^{\infty} \pi_i \delta_{x_i}, \quad \pi_i = \begin{cases} v_1 & i = 1, \\ v_i \prod_{t<i} (1 - v_t) & i > 1, \end{cases} \tag{1}$$

where $x_i \sim \mathcal{H}$, $\mathcal{H}$ is the base probability measure, δ_{x_i} is a discrete measure concentrated at x_i, $\boldsymbol{v} \sim \text{Beta}(\boldsymbol{\alpha}, \boldsymbol{\beta})$ with $\boldsymbol{\alpha}$ and $\boldsymbol{\beta}$ being the prior parameters, $\{\pi_i\}$ are random weights independent of $\mathcal{H}$ and satisfy $0 \leq \pi_i \leq 1$ and $\sum_{i=1}^{\infty} \pi_i = 1$. SBP specifies to the Dirichlet process if $\boldsymbol{v} \sim \text{Beta}\left(1, \hat{\boldsymbol{\beta}}\right)$, then the joint distribution over the infinite sequence of stick-breaking weights with concentration parameter $\hat{\boldsymbol{\beta}}$ is $\{\pi_i\} \sim \text{GEM}(\hat{\boldsymbol{\beta}})$ (Teh et al., 2004).

Note that the existing nTSNTM (Chen et al., 2021b) chooses the Kumaraswamy distribution (Kumaraswamy, 1980) to approximate the Beta distribution since it does not have a differentiable non-centered parametrization. However, the Beta distribution is a one-parameter subfamily of symmetric distributions and has more ways of generating the distribution via physical processes (Jones, 2009). To estimate the Beta distribution unbiasedly, we inference it through computing implicit reparameterization gradients and the details will be introduced in Section 3.3. As shown in Figure 2, we introduce an inference network to build the Beta distribution. We obtain $\boldsymbol{\alpha}$ and $\boldsymbol{\beta}$ by $\boldsymbol{\alpha} = l_\alpha(\boldsymbol{\eta})$ and $\boldsymbol{\beta} = l_\beta(\boldsymbol{\eta})$, where $\boldsymbol{\eta} = \text{MLP}(\boldsymbol{d})$ is the low-dimensional representation of $\boldsymbol{d}$, l_α and l_β are the single layer of linear connection, and MLP denotes a multilayer perceptron.

3.1.2 Topic Attention

We intend the model to attend on relationships among topics in a manner that the resultant attention is distributed according to the topics generated from the corpus. We employ topic embeddings $\mathbf{T_E} \in \mathbb{R}^{S \times d_t}$ to perform attention on topics, where $S \to \infty$ is the breaking number in SBP, i.e., the number of topics learned by SBP, and d_t is the dimension of topic embeddings. Considering that the relevance of each topic should be computed and learned independently, we regard each topic embedding as a subspace and project the input hidden representations to different topic embedding subspaces as follows:

$$\mathbf{Q} = \mathbf{T_E}\mathbf{W_Q}, \quad \mathbf{K} = \mathbf{T_E}\mathbf{W_K}, \tag{2}$$

where $\mathbf{W_Q} \in \mathbb{R}^{d_t \times d_r}$ and $\mathbf{W_K} \in \mathbb{R}^{d_t \times d_r}$ are trainable parameters, d_r is the dimension of each topic embedding subspace, $\mathbf{Q} \in \mathbb{R}^{S \times d_r}$ and $\mathbf{K} \in \mathbb{R}^{S \times d_r}$ denote the matrices of queries and keys. Based on queries and keys, we calculate a weighted adjacency matrix $\mathbf{C} \in \mathbb{R}^{S \times S}$, where each element

$C_{i,j}$ denotes the parent-child correlation degree of relevance between topics t_i and t_j[2]. To ensure that each element $C_{i,j}$ is discrete, a softmax function with temperature τ is applied (Hinton et al., 2015):

$$C_{i,j} = \frac{exp(\frac{\hat{C}_{i,j}}{\tau})}{\sum_{k=1}^{S} exp(\frac{\hat{C}_{i,k}}{\tau})}, \tag{3}$$

where $\hat{\mathbf{C}} = \mathbf{Q}\mathbf{K}^T$.

To further incorporate the parent-child correlations into document-topic distributions meanwhile maintaining the nonparametric characteristic from SBP, we firstly calculate the parent document-topic distribution by integrating the weighted adjacency matrix, i.e., $\boldsymbol{\pi}' = \boldsymbol{\pi} \times \mathbf{C}$. Particularly, as the weighted adjacency matrix has not captured parent-child correlations at the beginning, the document-topic distribution is initially represented by $\boldsymbol{\pi}$. With the learning of our self-attention module, the quality of the weighted adjacency matrix $\mathbf{C}$ is steadily enhanced, and $\boldsymbol{\pi}'$ becomes a valuable supplement of $\boldsymbol{\pi}$ due to strong parent-child topic relationships. Then, we generate the consolidated document-topic distribution by exploiting both $\boldsymbol{\pi}'$ and $\boldsymbol{\pi}$ as $\boldsymbol{\theta} = (1-\gamma)\boldsymbol{\pi}' + \gamma\boldsymbol{\pi}$, where γ is a decay coefficient.

3.1.3 Decoder

As the BoW document representations lack of the word relatedness information, we incorporate pre-trained word embeddings $\mathbf{W_E}$ (Viegas et al., 2020; Wu et al., 2020) into the network. For the decoder, we firstly obtain the topic-word distribution $\boldsymbol{\Phi} = \text{softmax}(\mathbf{T_E} \times \mathbf{W_E})$, where $\mathbf{W_E} \in \mathbb{R}^{d_t \times V}$ and $\boldsymbol{\Phi} \in \mathbb{R}^{S \times V}$. Then, we reconstruct document $\hat{\boldsymbol{d}}$ by combining document-topic distribution $\boldsymbol{\theta}$ with topic-word distribution $\boldsymbol{\Phi}$.

3.2 Topic Forest Hierarchy

To construct the topic forest, we build a forest hierarchy of topics from the weighted adjacency matrix $\mathbf{C}$ which contains the parent-child relationships among topics. However, the structure of weighted adjacency matrix $\mathbf{C}$ may not be a reasonable hierarchical structure, i.e., it may have loops if without constraints. To tackle this challenge, we propose to construct the structure of topics as a sparse DAG. Zheng et al. (2018) have proved that a weighted adjacency matrix $\mathbf{C}$ is a DAG if and only if $h(\mathbf{C}) = \text{tr}\left(e^{\mathbf{C}\circ\mathbf{C}}\right) - d = 0$, where $\circ$ is the Hadamard product and $e^{\mathbf{C}}$ is the matrix exponential of $\mathbf{C}$. We employ $h(\mathbf{C}) = 0$ with an augmented Lagrangian method to ensure the acyclicity of the weighted adjacency matrix.

Considering that the topic hierarchy is reasonable and children topics can be represented by their parent topics, we assume that the sum of children topics' document-topic distributions is similar to their parent topics' document-topic distributions. Accordingly, we learn the forest-structured topic hierarchy by minimizing the difference between documents reconstructed by $\boldsymbol{\pi} \times \boldsymbol{\Phi}$ and those reconstructed by their parent document-topic distributions (i.e., $\boldsymbol{\pi}' \times \boldsymbol{\Phi}$) under the constraint of $h(\mathbf{C}) = 0$ using the augmented Lagrangian method, as follows:

$$\begin{aligned} \min_{\mathbf{C}\in\mathbb{R}^{S\times S}} \quad & \mathcal{L}_C = \frac{1}{2}\|(\boldsymbol{\pi} - \boldsymbol{\pi}') \times \boldsymbol{\Phi}\|_F^2 \\ & + \frac{\rho}{2}|h(\mathbf{C})|^2 + \epsilon h(\mathbf{C}), \end{aligned} \tag{4}$$

where ρ is a penalty parameter, and ϵ is the Lagrange multiplier. We update parameters ρ and ϵ by following (Zheng et al., 2018), as follows: $\begin{cases} \rho_i = & 2\rho_{i-1} \\ \epsilon_i = & \epsilon_{i-1} + \rho h_{i-1} \end{cases}$, where $\rho_0 = 1$, $\epsilon_0 = 0$, i is a training epoch, and h is a constraint value.

The generative process of nFNTM is described as follows:

1. For each document $\boldsymbol{d}$:
   (a) Draw a topic proportion $\boldsymbol{\pi} \sim \text{GEM}(\hat{\boldsymbol{\beta}})$;
   (b) Get the weighted adjacency matrix $\mathbf{C}$;
   (c) Get the correlational topic distribution $\boldsymbol{\theta}$.
2. For each word $w_{d,n} \in \boldsymbol{d}$:
   (a) Draw a topic $z_{d,n} \sim \text{Mult}(\boldsymbol{\theta})$;
   (b) Obtain the topic-word distribution $\boldsymbol{\Phi}$;
   (c) Draw a word $w_{d,n} \sim \text{Mult}(\boldsymbol{\Phi}_{z_{d,n}})$.

3.3 Parameter Inference

We apply NVI to inference network parameters, which is proven to be efficient and flexible (Srivastava and Sutton, 2017; Miao et al., 2017; Isonuma et al., 2020; Chen et al., 2021b). The likelihood of each reconstructed document $\hat{\boldsymbol{d}}$ is estimated by $p\left(\hat{\boldsymbol{d}} \mid \boldsymbol{\theta}, \boldsymbol{\Phi}\right) = \sum_{\boldsymbol{z}} p\left(\hat{\boldsymbol{d}} \mid \boldsymbol{\Phi}_{\boldsymbol{z}}\right) p\left(\boldsymbol{z} \mid \boldsymbol{\theta}\right)$, where $\boldsymbol{z}$ is the topic assigned for each word in $\hat{\boldsymbol{d}}$. To maximize the log-likelihood, we derive the lower bound

[2]Different from (Vaswani et al., 2017), we capture the relationships among topics without using the value matrix. This is because we found experimentally that topics trained with value matrix tend to be the same, which might result from concentrating on several important topics.

as follows:

$$\begin{aligned}\mathcal{L}_B =& \mathbb{E}_{q(\boldsymbol{\theta},\boldsymbol{\Phi}|\boldsymbol{d})}\left[\log p\left(\hat{\boldsymbol{d}} \mid \boldsymbol{\theta},\boldsymbol{\Phi}\right)\right] \\ &- D_{KL}\left[q\left(\boldsymbol{\theta} \mid \boldsymbol{d}\right)\|p\left(\boldsymbol{\theta}\right)\right] - \mathcal{L}_C,\end{aligned} \tag{5}$$

where $\mathbb{E}_{q(\boldsymbol{\theta},\boldsymbol{\Phi}|\boldsymbol{d})}\left[\log p\left(\hat{\boldsymbol{d}} \mid \boldsymbol{\theta},\boldsymbol{\Phi}\right)\right]$ is the reconstruction loss, $D_{KL}\left[q\left(\boldsymbol{\theta} \mid \boldsymbol{d}\right)\|p\left(\boldsymbol{\theta}\right)\right]$ is the Kullback-Leibler (KL) divergence between the prior Beta distribution $p\left(\boldsymbol{\theta}\right)$ and the posterior Beta distribution $q\left(\boldsymbol{\theta} \mid \boldsymbol{d}\right)$. The KL divergence of two Beta distributions is given below:

$$\begin{aligned}&KL(Beta(\alpha_1,\beta_1)||Beta(\alpha_2,\beta_2)) = \\ &\quad \ln\Gamma(\alpha_2) + \ln\Gamma(\beta_2) + \ln\Gamma(\alpha_1+\beta_1) \\ &\quad - (\ln\Gamma(\alpha_1) + \ln\Gamma(\beta_1) + \ln\Gamma(\alpha_2+\beta_2)) \\ &\quad + (\alpha_1-\alpha_2)\digamma(\alpha_1) \\ &\quad + (\beta_1-\beta_2)\digamma(\beta_1) \\ &\quad + (\alpha_2+\beta_2-\alpha_1-\beta_1)\digamma(\alpha_1+\beta_1),\end{aligned} \tag{6}$$

where $\digamma(x) = \frac{d}{dx}\ln\digamma(x) = \frac{\digamma'(x)}{\digamma(x)}$.

Note that we transform $\boldsymbol{d}$ to the variational Beta distribution, thus $q\left(\boldsymbol{\theta} \mid \boldsymbol{d}\right)$ is derived by:

$$\begin{aligned}q\left(\boldsymbol{\theta} \mid \boldsymbol{d}\right) &= \text{Beta}(\boldsymbol{\theta} \mid \boldsymbol{\alpha},\boldsymbol{\beta})) \\ &= \text{Beta}(\boldsymbol{\theta} \mid l_\alpha\left(\text{MLP}(\boldsymbol{d})\right), l_\beta\left(\text{MLP}(\boldsymbol{d})\right)).\end{aligned} \tag{7}$$

To compute reparameterization gradients, the Beta samples are obtained from Gamma samples since the latter do not require inverting the standardization function (Figurnov et al., 2018): for $z_1 \sim \text{Gamma}(\boldsymbol{\alpha}, 1)$ and $z_2 \sim \text{Gamma}(\boldsymbol{\beta}, 1)$, it has $\frac{z_1}{z_1+z_2} \sim \text{Beta}(\boldsymbol{\alpha},\boldsymbol{\beta})$. The lower bound $\mathcal{L}_B$ is used to calculate gradients and parameters are updated by Adam (Kingma and Ba, 2015).

4 Experiment

4.1 Experimental Setup

Datasets To evaluate our nFNTM[3] comprehensively, we conduct experiments on three datasets which are observed to be hierarchical (Chen et al., 2021b): 20News (Miao et al., 2017), Reuters (Wu et al., 2020), and Wikitext-103 (Nan et al., 2019). According to (Liu et al., 2018; Viegas et al., 2020), a topic often comprises of some sub-topics and owns a hierarchical structure in Web directory of news (e.g., 20News and Reuters) and encyclopedia (e.g., Wikitext-103 which is extracted from Wikipedia). For these datasets, each document is associated with a manually-curated hierarchy of categories. Take 20News as an example, "rec.autos" and "rec.motor.cycles" are sub categories of "rec". Instead of relying on the prior coarse-grained categories, we build the hierarchical structure from a corpus at the fine-grained topic level, by automatically mining a set of representative words for each topic in a forest structure to help a user comprehend her/his interested topics.

All datasets have undergone a preprocessing of removing stop words and deleting low-frequency words. Table 1 shows the numbers of training and test documents, as well as the vocabulary size.

Dataset	#Docs(Train)	#Docs(Test)	Vocabulary size
20News	11,314	7,531	1,995
Reuters	7,769	3,019	2,000
Wikitext-103	28,472	120	20,000

Table 1: Statistics of datasets.

Baselines We employ four tree-structured topic models, including hLDA[4] (Blei et al., 2003a), rCRP[5] (Kim et al., 2012), TSNTM[6] (Isonuma et al., 2020), and nTSNTM[7] (Chen et al., 2021b), and three DAG-structured topic models, including hPAM (Mimno et al., 2007), HSOC (Liu et al., 2018), and CluHTM (Viegas et al., 2020) as baselines. For tree-structured baselines, the max-depth of topic tree is set to 3 by following (Isonuma et al., 2020). In addition, we adopt seven flat topic models for comparison, including parametric models of GSM[8] (Miao et al., 2017), GSB (Miao et al., 2017), NB-NTM[9] (Wu et al., 2020) and GNB-NTM[9] (Wu et al., 2020), and nonparametric models of HDP[10] (Teh et al., 2004), iTM-VAE[11] (Ning et al., 2020), and HiTM-VAE[11] (Ning et al., 2020). Except for GSB, we use the open source codes for all other baseline models.

Hyper-parameter Settings To ensure fair comparisons, we follow (Chen et al., 2021b) to set topic numbers to 50 and 200 for all parametric models. For non-parametric models based on SBP (i.e., iTM-VAE and nTSNTM), the maximum number of topics is set to 200, and the concentration

[3] https://github.com/Angr4Mainyu/nFNTM

[4] https://github.com/joewandy/hlda

[5] https://github.com/uilab-github/rCRP

[6] https://github.com/misonuma/tsntm

[7] https: //github.com/hostnlp/nTSNTM

[8] https://github.com/linkstrife/NVDM-GSM

[9] https://github.com/mxiny/NB-NTM

[10] https://github.com/arnim/HDP

[11] https://github.com/walkerning/itmvae_public

parameter $\hat{\beta}$ of the GEM distribution is set to 20. According to (Chen et al., 2021b), we select the topic with a total probability exceeding 95% as an active topic. Besides, the penalty parameter ρ is updated by $\rho = 2^x$, where x denotes the number of training epochs. The temperature τ changes exponentially from 5 to 1×10^{-4} to ensure the adjacency matrix to be sparse. The hidden layer size of the encoder is set to 256, which is consistent with other models. In the construction of the weighted adjacency matrix, we use an exponential change strategy to dynamically adjust the decay coefficient. This is because the quality of such a weighted adjacency matrix is gradually enhanced when learning the self-attention module. Finally, the decay coefficient γ changes exponentially from 1 to 0.5. We implement our model by pytorch and run it on a computer with NVIDIA 1080Ti and 128GB RAM.

4.2 Topic Hierarchy Analysis

To evaluate the rationality, affinity, and diversity of the topic hierarchy generated by different models, we adopt four metrics: topic specialization (Kim et al., 2012), Cross-Level Normalized Point-wise Mutual Information (CLNPMI) (Chen et al., 2021b), hierarchical affinity (Kim et al., 2012), and Topic Uniqueness (TU) (Nan et al., 2019). Key words of each topic are ranked by the topic-word matrix $\mathbf{\Phi}$ (Blei et al., 2003b).

Topic Hierarchical Rationality For the tree-structured topics, the topics closer to the root node will be more general, while topics closer to the leaf node will be more specific. Topic specialization score is to quantify this feature by computing the cosine similarity of the word distribution between each topic and the entire corpus. Since our forest-structured model generates several trees, we pick out all three-layer topic trees to obtain the average score for comparison. Figure 3 shows the topic specialization results, from which we can observe that our model outperforms baselines except for HSOC and CluHTM. Although the topic specializations of HSOC and CluHTM at different levels are close to 1, they still lack rationality since the root topic in a tree should be more general than others.

To measure the relationship between two connected topics, a metric of CLNPMI (Chen et al., 2021b) was proposed by calculating the average Normalized Point-wise Mutual Information (NPMI) score of every parent and its children topics, as follows: $CLNPMI\,(W_p, W_c) = \sum_{w_i \in W'_p} \sum_{w_j \in W'_c} \frac{\mathrm{NPMI}(w_i, w_j)}{|W'_p||W'_c|}$, where $W'_p = W_p - W_c$ and $W'_c = W_c - W_p$, in which W_p and W_c denote the top N words of a parent topic its child topic, respectively. A higher CLNPMI indicates that children topics are more coherent with their corresponding parent topics. We compare our nFNTM with hierarchical topic models mentioned above, and the CLNPMI results of all models are shown in Table 2. Note that the two NMF-based models (i.e., HSOC and CluHTM) had not converged after running for more than 48 hours on Wikitext-103, thus we did not include their results in the table. As shown in Table 2, our nFNTM achieves the best performance on all datasets.

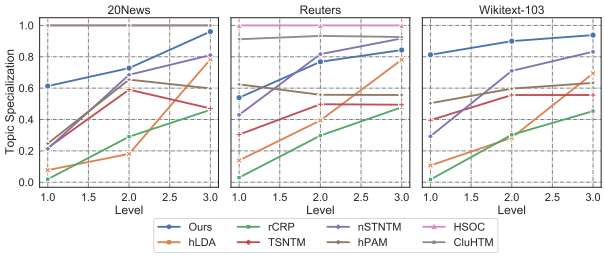

Figure 3: Topic specialization of different topics trees generated from the three datasets. A higher score with a growing trend means better performance.

Models	20News		Reuters		Wikitext-103	
	CLNPMI	*TU*	*CLNPMI*	*TU*	*CLNPMI*	*TU*
hLDA	0.065	0.051	0.050	0.447	0.063	0.597
rCRP	0.098	0.285	0.072	0.227	0.088	0.355
TSNTM	0.086	0.430	0.027	0.370	0.065	0.615
nTSNTM	0.109	0.745	0.102	0.708	0.113	0.730
hPAM	0.046	0.606	0.011	0.470	0.047	0.713
HSOC	0.128	0.231	0.047	0.211	-	-
CluHTM	0.123	0.116	0.016	0.117	-	-
Ours	**0.152**	**0.757**	**0.125**	**0.798**	**0.118**	**0.766**

Table 2: The average CLNPMI and TU scores of hierarchical topic models with top 5, 10, and 15 words for each topic. A higher score means better performance and the best scores are highlighted by boldface.

Topic Uniqueness To evaluate the diversity of hierarchical topics, we calculate the topic uniqueness by: $TU = \frac{Count(Set(W_{topN}))}{N \times S}$, where $Count(Set(W_{topN}))$ is the number of distinct words in top N words of all topics. A higher TU means that the generated topics are more diverse. Table 2 shows the TU results for all models, from which we can observe that our model generates more diverse topics than others. The baselines of HSOC and CluHTM perform quite poor since they need to preset up to 600 topics for convergence.

Topic Hierarchical Affinity A reasonable assumption for topic hierarchy is that topics with parent-child relationships show larger similarities in their topic-word distributions than topics without any parent-child relationship (Kim et al., 2012). According to (Kim et al., 2012), we firstly evaluate the similarity between parent-child topics and non-parent-child topics by computing the cosine similarity between their topic-word distributions. Then, the topic hierarchical affinity is measured according to the difference of those similarities.

Let $\mathbf{\Phi}(k)$ be a topic at level k, $\lambda(k)$ be children topics of $\mathbf{\Phi}(k)$, and $\bar{\lambda}(k)$ be non-children topics of $\mathbf{\Phi}(k)$. The topic hierarchical affinity metric compares the average cosine similarity $\mathrm{S}_{\lambda}(k)$ between $\mathbf{\Phi}(k)$ and all topics in $\lambda(k)$ against the average cosine similarity $\mathrm{S}_{\bar{\lambda}}(k)$ between $\mathbf{\Phi}(k)$ and all topics in $\bar{\lambda}(k)$. A large difference between $\mathrm{S}_{\lambda}(k)$ and $\mathrm{S}_{\bar{\lambda}}(k)$ indicates a good topic hierarchical affinity.

Since all topics between paired levels in hPAM are fully connected, it is impossible to clearly distinguish topics with parent-child and non-parent-child relationships. Thus, we exclude hPAM in this part. As shown in Figure 4, our model achieves high similarities between parent-child topics and low similarities between non-parent-child topics, indicating a good hierarchical affinity. Note that both HSOC and CluHTM are based on NMF, which rely on a predefined number of topic trees. To avoid missing potential topic sub-structures, these models need to set a large number of topic trees (e.g., 100) according to their default settings. Given superabundant topic trees, the similarity between non-parent-child topics will be underestimated due to the huge number of non-parent-child topics, resulting in a competitive hierarchical affinity. However, it leads to a very poor TU score for HSOC or CluHTM, as already shown in Table 2.

4.3 Topic Interpretability

We employ the NPMI score (Lau et al., 2014) to evaluate the interpretability of topics since this metric is shown to be close to human judgments (Lau et al., 2014). We extract top 5, 10, and 15 words for each topic and compute the average NPMI scores of all models over the three datasets. The higher the value of NPMI score, the more interpretable the generated topics is. The corresponding numbers of topics automatically determined by our nonparametric model on 20News, Reuters, and Wikitext-103 are 61, 76, and 121, respectively.

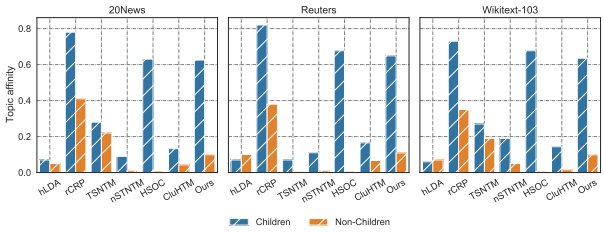

Figure 4: Topic hierarchical affinity results. For each associated topic, a larger difference of word distributions between children and non-children topics is better.

Table 3 shows the NPMI scores of topics generated by different models. Although our nFNTM outperforms hierarchical topic models by a large margin, it performs slightly worse than the top-performing flat topic model. The reason is that NPMI is the average score of all topics, which may be unfair for hierarchical topic models. Flat topic models treat topics independently and result in specific topics, while higher-level topics in hierarchical topic models are general with a lower NPMI score. Figure 5 shows two examples of topic trees generated by nFNTM on Wikitext-103 (left) and 20News (right), which focus on different topics, i.e., *politics* and *war* in the left tree, and *play*, *religion*, and *computer* in the right tree. Take the left tree as an illustration, the NPMI score of the root topic [*day hour people home left*] is obviously lower than that of the leaf topic [*force army battle attack war*], which affects the overall result. We can also observe that the topics closer to root are more general and those closer to leaves are more specific. Besides, children topics are related to parent topics, e.g., encryption is a child of computer, and law is a child of governance. These examples of topics validate the advantage of a forest structure when compared with a tree structure: the former can not only learn reasonable parent-child relationships among topics, but also generate a flexible topic hierarchy with unbounded depth and width.

4.4 Ablation Study

We perform ablation experiments on our model to validate the effectiveness of each module. Table 4 shows the ablation results of our nFNTM without three modules, where "Ours w/o SBP" denotes a parametric model with Gaussian distribution instead of SBP, "Ours w/o self-attention" means that the weighted adjacency matrix is randomly set rather than learned by the proposed self-attention mechanism, and "Ours w/o self-attention + DAG"

Datasets	20News		Reuters		Wikitext-103	
Model	50	200	50	200	50	200
GSM	0.211	0.165	0.198	0.155	0.214	0.217
GSB	0.231	0.191	0.152	0.136	0.229	0.131
NB-NTM	0.188	0.223	0.248	0.245	0.127	0.125
GNB-NTM	**0.240**	0.228	0.237	0.255	0.127	0.093
HDP	0.192		0.266		0.157	
iTM-VAE	0.195		0.201		0.184	
HiTM-VAE	0.237		**0.269**		0.233	
rCRP	0.186		0.206		0.201	
hLDA	0.221		0.185		0.186	
TSNTM	0.212		0.206		0.213	
nTSNTM	0.219		0.234		0.237	
hPAM	0.213		0.229		0.223	
HSOC	0.223		0.210		-	
CluHTM	0.219		0.161		-	
Ours	0.235		0.251		**0.240**	

Table 3: The average NPMI scores of different models using top 5, 10, and 15 words for each topic. A higher score means better performance and the best scores are highlighted by boldface.

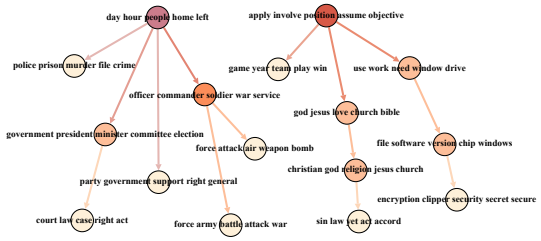

Figure 5: Two examples of topic trees generated by nFNTM on Wikitext-103 (left) and 20News (right). Each node represents a topic with top 5 words, and the arrow direction is from parent to children.

denotes reconstructing documents by their topic distributions generated from SBP. Note that there is a flat topic hierarchy without DAG, thus the CLNPMI metric can not be calculated. As shown in Table 4, nonparametric and self-attention modules are beneficial to generate a good topic hierarchy and achieve improvements in topic interpretability.

4.5 Concentration Parameter Evaluation

Here, we vary the values of concentration parameter $\hat{\beta}$ to validate the nonparametric property of our model on 20News. Figure 6 shows that the topic numbers of all nonparametric models are promoted by increasing the value of $\hat{\beta}$, which is reasonable since a larger $\hat{\beta}$ leads to a smoother distribution of SBP, and the smoother distribution generates more topic numbers than a denser distribution. It also demonstrates that our model generates more topics for a larger $\hat{\beta}$ since these topics are dispersed (Wu et al., 2020), and our model performs better than other models on approximating the nonparametric property of HDP.

Model	#Topics	NPMI	CLNPMI
Ours	61	**0.235**	**0.152**
Ours w/o SBP	50	0.223	0.135
Ours w/o self-attention	57	0.209	0.097
Ours w/o self-attention+DAG	48	0.203	-

Table 4: Ablation evaluation on 20News.

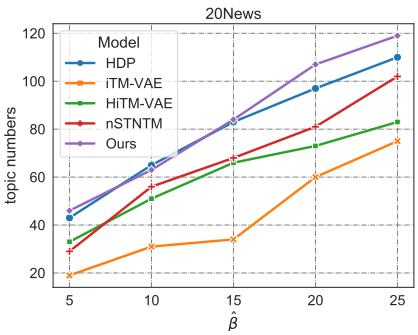

Figure 6: Topic numbers derived by different nonparametric models with various values of $\hat{\beta}$.

4.6 Forest-Structured Topic Visualization

In this part, we qualitatively analyze the rationality of the topic forest generated by our model. We determine parent-child topic relationships by the value of the weighted adjacency matrix $\mathbf{C}$. $C_{i,j} \approx 1$ means that child topic t_i connects to parent topic t_j. For clarity, we show an example of $\mathbf{C}$ and how to build a topic forest from it in Figure 7. Such a topic hierarchy is based on the assumption that a document can be decomposed into a weighted sum of multiple topics, where a topic can also be decomposed into a weighted sum of multiple sub-topics (Chen et al., 2021b).

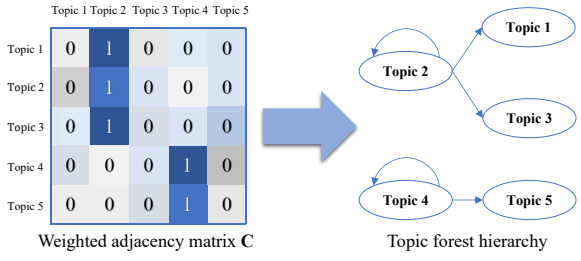

Figure 7: An example of building a topic forest from a weighted adjacency matrix.

Figure 8(a) shows the visualization of the weighted adjacency matrix generated from Reuters. Except for the dots on the diagonal, the rest of

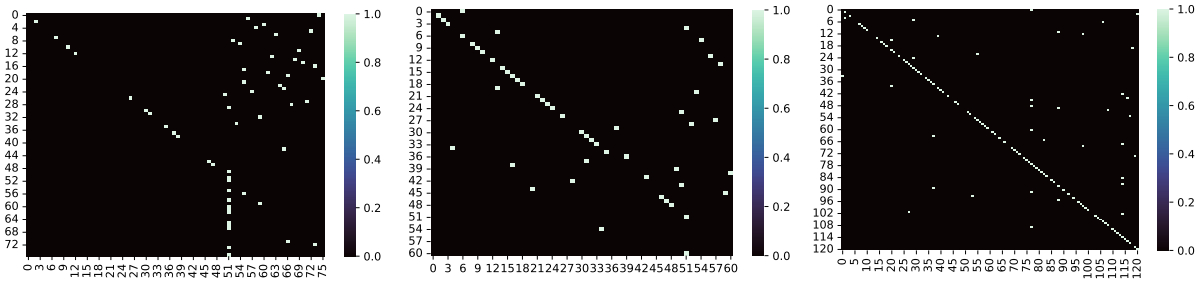

(a) Reuters (b) 20News (c) Wikitex-103

Figure 8: Visualization of the weighted adjacency matrix generated from (a) Reuters, (b) 20News, and (c) Wikitext-103, respectively. The white dot means that there is a correlation between two topics indexed in the row and column.

dots present a discrete distribution to meet the constraints of DAG. In addition, we observe that topic 51 is connected to various topics and becomes the parent topic of multiple topics, which is a general root topic of the whole corpus. Figures 8(b) and 8(c) present the weighted adjacency matrices generated by our model on 20News and Wikitext-103, respectively. The results also indicate that except for the dots on the diagonal, the rest of dots present a discrete distribution on both datasets, i.e., they meet the constraints of DAG.

Besides, we use Gephi[12] to visualize the topic forests generated by our model over 20News, Reuters, and Wikitext-103 in Figures 9, 10, and 11, where each node represents a topic with top 5 words and the arrow direction is from parent to children. The color of each node is determined by indegree. It indicates that the topic forest generated from Reuters contains a master tree whose root topic is about interviews. Furthermore, the topic forests generated from 20News and Wikitext-103 both consist of multiple trees, where the root topics mainly involve news and daily, respectively. All results indicate a rational hierarchical structure of topics obtained by our model, i.e., root topics are general and children topics are specific to their corresponding parents.

4.7 Speed Comparison

For speed comparison, we record the running time by taking the average cost of training each model for 5 times over 10k documents sampled from 20News. The running time for baseline methods are: hLDA - 16.74s , HSOC - 50.97s, CluHTM - 49.72s, and nTSNTM - 1.57s. Our model takes 1.35s, which is significantly faster than models based on Bayesian learning and NMF. Although nTSNTM achieves a competitive result, it applies Kumaraswamy distribution in parameter inference which slows down the model speed. Besides, such a tree-structured method is limited to local information and data sparsity along the hierarchy of topics (Viegas et al., 2020). In particular, the deeper the topic tree, the less coincident data remains. This deteriorates the quality of tree-structured methods inevitably. Compared with a single deep tree, the forest structure with several shallower trees could circumvent the above problem theoretically. Furthermore, with varied numbers of trees, a topic forest has a more flexible hierarchical structure and better adaptability than a fixed tree structure.

[12] http://gephi.org/

5 Conclusion

In this work, we present a forest-structured neural topic model named nFNTM. Particularly, the self-attention mechanism is applied to capture topic correlation and an augmented Lagrangian method is employed to build the topic forest under DAG constraints. By computing reparameterization of Gamma distribution, the SBP is adopted to obtain a nonparametric model. We empirically demonstrate that our nFNTM generates more rational, affinitive, and diverse topics than state-of-the-art hierarchical topic models, meanwhile achieves better topic interpretability than various baselines. Although flat topic models can generate distinguishable topics with larger NPMI scores than hierarchical ones, they treat all topics independently without explaining the corpus well to a user.

Acknowledgements

We are grateful to the reviewers for their constructive comments and suggestions. This work has been supported by the National Natural Science Foundation of China (61972426).

References

Amr Ahmed, Liangjie Hong, and Alexander Smola. 2013. Nested chinese restaurant franchise process: Applications to user tracking and document modeling. In *ICML*, pages 1426–1434.

David M. Blei, Thomas L. Griffiths, Michael I. Jordan, and Joshua B. Tenenbaum. 2003a. Hierarchical topic models and the nested chinese restaurant process. In *NIPS*, pages 17–24.

David M. Blei, Andrew Y. Ng, and Michael I. Jordan. 2003b. Latent dirichlet allocation. *Journal of Machine Learning Research*, 3:993–1022.

Ziye Chen, Cheng Ding, Yanghui Rao, Haoran Xie, Xiaohui Tao, Gary Cheng, and Fu Lee Wang. 2021a. Hierarchical neural topic modeling with manifold regularization. *World Wide Web*, 24:2139–2160.

Ziye Chen, Cheng Ding, Zusheng Zhang, Yanghui Rao, and Haoran Xie. 2021b. Tree-structured topic modeling with nonparametric neural variational inference. In *ACL/IJCNLP*, pages 2343–2353.

Mikhail Figurnov, Shakir Mohamed, and Andriy Mnih. 2018. Implicit reparameterization gradients. In *NeurIPS*, pages 439–450.

Geoffrey E. Hinton, Oriol Vinyals, and Jeffrey Dean. 2015. Distilling the knowledge in a neural network. *CoRR*.

Hemant Ishwaran and Lancelot F. James. 2001. Gibbs sampling methods for stick-breaking priors. *Journal of the American Statistical Association*, 96:161–173.

Masaru Isonuma, Junichiro Mori, Danushka Bollegala, and Ichiro Sakata. 2020. Tree-structured neural topic model. In *ACL*, pages 800–806.

M.C. Jones. 2009. Kumaraswamy's distribution: A beta-type distribution with some tractability advantages. *Statistical Methodology*, 6(1):70–81.

Joon Hee Kim, Dongwoo Kim, Suin Kim, and Alice H. Oh. 2012. Modeling topic hierarchies with the recursive chinese restaurant process. In *CIKM*, pages 783–792.

Diederik P. Kingma and Jimmy Ba. 2015. Adam: A method for stochastic optimization. In *ICLR*.

P. Kumaraswamy. 1980. A generalized probability density function for double-bounded random processes. *Journal of Hydrology*, 46(1):79–88.

Jey Han Lau, David Newman, and Timothy Baldwin. 2014. Machine reading tea leaves: Automatically evaluating topic coherence and topic model quality. In *EACL*, pages 530–539.

Daniel D Lee and H Sebastian Seung. 1999. Learning the parts of objects by non-negative matrix factorization. *Nature*, 401(6755):788–791.

Wei Li and Andrew McCallum. 2006. Pachinko allocation: Dag-structured mixture models of topic correlations. In *ICML*, pages 577–584.

Rui Liu, Xingguang Wang, Deqing Wang, Yuan Zuo, He Zhang, and Xianzhu Zheng. 2018. Topic splitting: A hierarchical topic model based on non-negative matrix factorization. *Journal of Systems Science and Systems Engineering*, 27(4):479–496.

Yishu Miao, Edward Grefenstette, and Phil Blunsom. 2017. Discovering discrete latent topics with neural variational inference. In *ICML*, pages 2410–2419.

David M. Mimno, Wei Li, and Andrew McCallum. 2007. Mixtures of hierarchical topics with pachinko allocation. In *ICML*, pages 633–640.

Feng Nan, Ran Ding, Ramesh Nallapati, and Bing Xiang. 2019. Topic modeling with wasserstein autoencoders. In *ACL*, pages 6345–6381.

Xuefei Ning, Yin Zheng, Zhuxi Jiang, Yu Wang, Huazhong Yang, Junzhou Huang, and Peilin Zhao. 2020. Nonparametric topic modeling with neural inference. *Neurocomputing*, 399:296–306.

John W. Paisley, Chong Wang, David M. Blei, and Michael I. Jordan. 2015. Nested hierarchical dirichlet processes. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 37(2):256–270.

Akash Srivastava and Charles Sutton. 2017. Autoencoding variational inference for topic models. In *ICLR*.

Yee Whye Teh, Michael I. Jordan, Matthew J. Beal, and David M. Blei. 2004. Sharing clusters among related groups: Hierarchical dirichlet processes. In *NIPS*, pages 1385–1392.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *NIPS*, pages 5998–6008.

Felipe Viegas, Washington Cunha, Christian Gomes, Antônio Pereira, Leonardo Rocha, and Marcos André Gonçalves. 2020. Cluhtm - semantic hierarchical topic modeling based on cluwords. In *ACL*, pages 8138–8150.

Jiemin Wu, Yanghui Rao, Zusheng Zhang, Haoran Xie, Qing Li, Fu Lee Wang, and Ziye Chen. 2020. Neural mixed counting models for dispersed topic discovery. In *ACL*, pages 6159–6169.

Shunyu Yao, Binghui Peng, Christos H. Papadimitriou, and Karthik Narasimhan. 2021. Self-attention networks can process bounded hierarchical languages. In *ACL/IJCNLP*, pages 3770–3785.

Xun Zheng, Bryon Aragam, Pradeep Ravikumar, and Eric P. Xing. 2018. Dags with NO TEARS: continuous optimization for structure learning. In *NeurIPS*, pages 9492–9503.

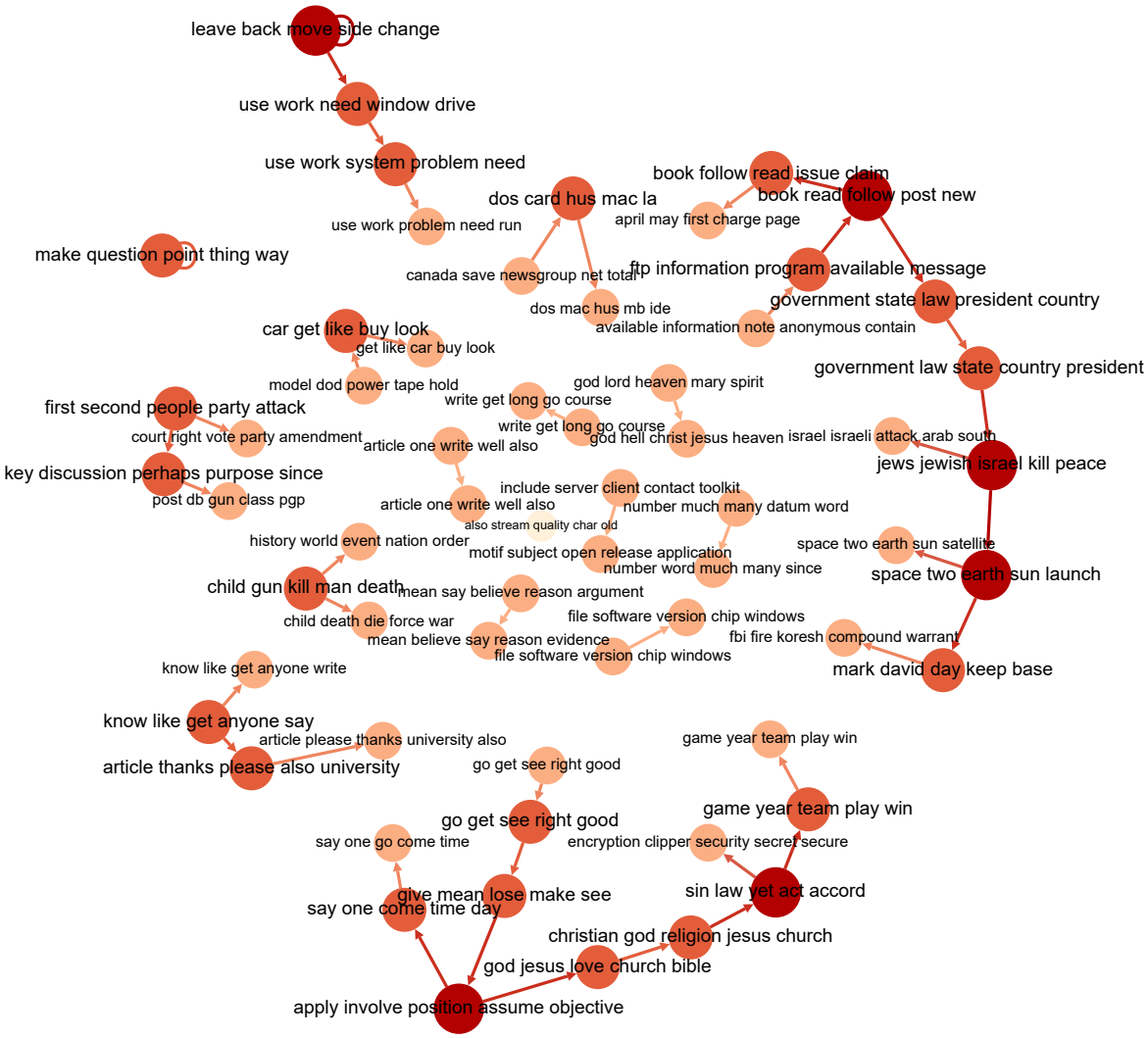

Figure 9: Visualization of the topic forest generated from 20News. Each node represents a topic and the arrow direction is from parent to children. There are about 8 trees in the forest, two of which have complex levels and structures, and two trees have only one root node. Take three of these trees as an illustration, the top 5 words of root topics and their children topics are (1) [know, like, get, anyone, say]⟶[article, thanks, please, also, university]; (2) [apply, involve, position, assume, objective]⟶[god, jesus, love, church, bible] and [game, year, team, play, win]; (3) [available, information, note, anonymous, contain]⟶[ftp, information, program, available, message]⟶[book, read, follow, post, new]. The root topics are general while children topics are specific.

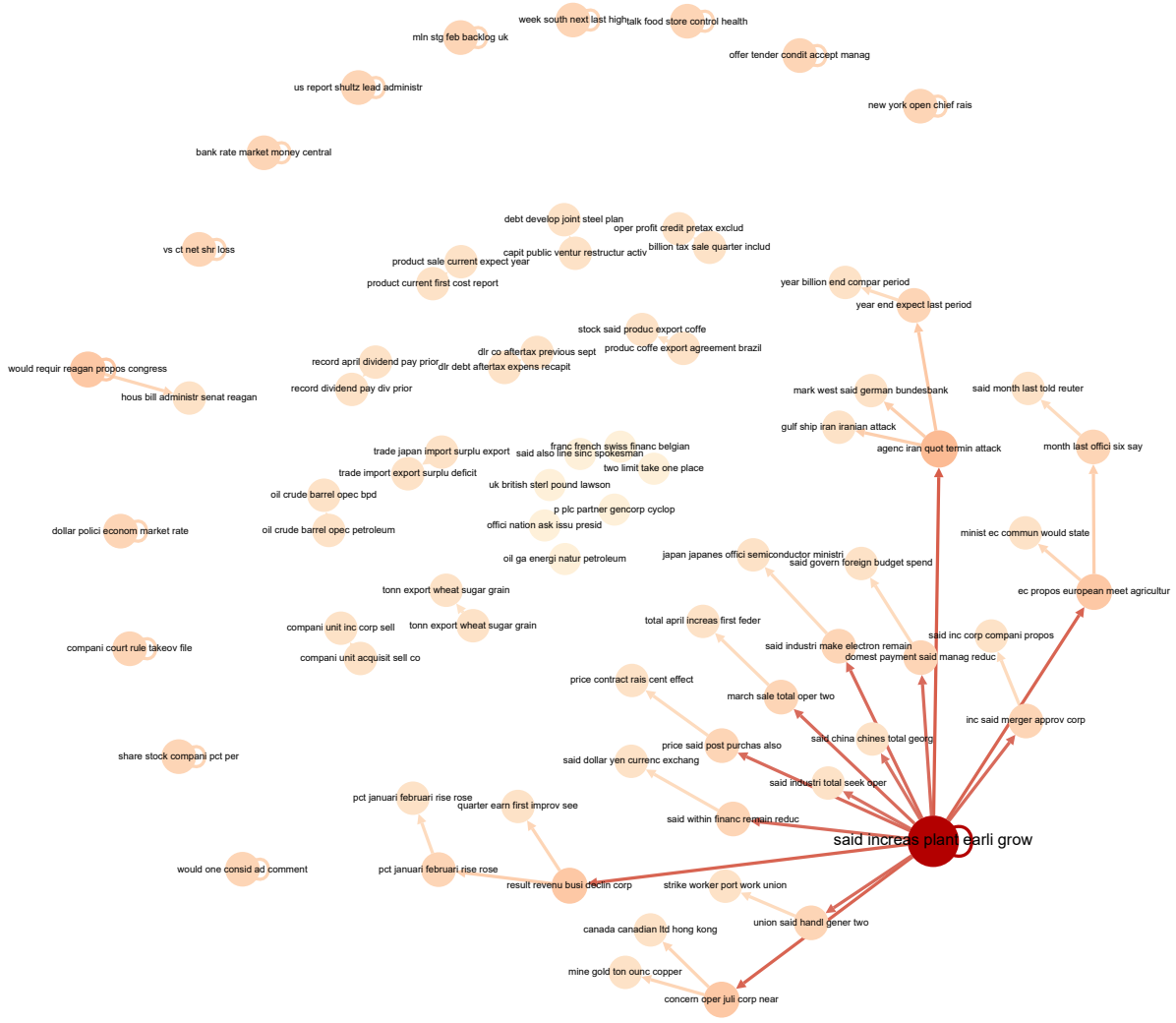

Figure 10: Visualization of the topic forest generated from Reuters, where a huge tree is surrounded by several trees with only two levels. The root topic of the largest tree contains words: [said, increase, plan, earli, grow], which are commonly used during interviews. The surrounding areas are related to the topics of various categories, including [mine, gold, ton, ounce, copper], [strike, worker, port, work, union], and [year, end, expect, last, period].

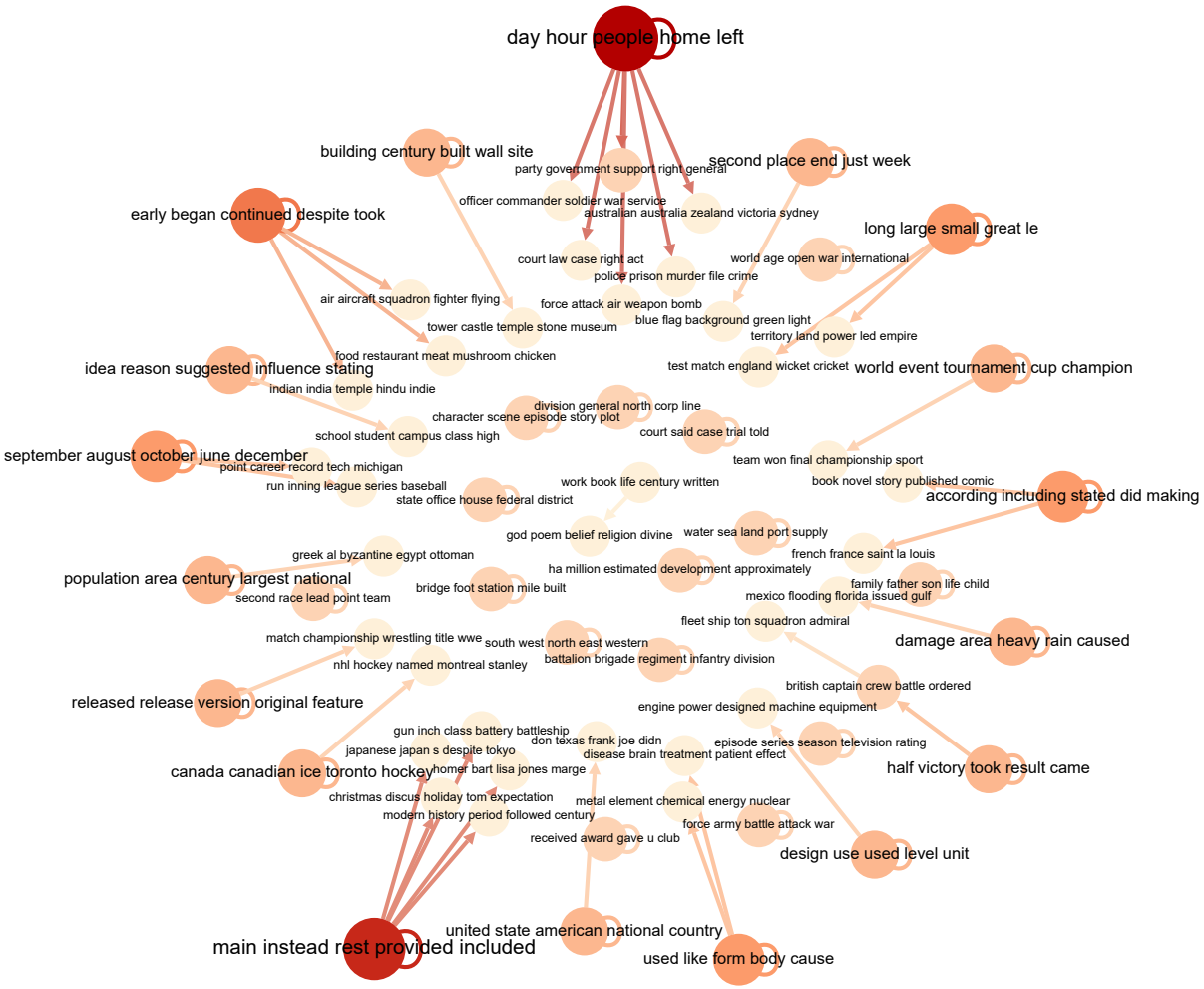

Figure 11: Visualization of the topic forest generated from Wikitext-103. Possibly due to a large variety of topics in this corpus, the result presents a more dispersed and shallower structure than others. The top words of two root topics and their children topics are (1) [day, hour, people, home, left]⟶[court, law, case, right, act] and [party, government, support, right, general]; (2) [main, instead, rest, provides, included]⟶[christmas, discus, holiday, tom, expectation] and [gun, inch, class, battery, battleship].

KGE-CL: Contrastive Learning of Tensor Decomposition Based Knowledge Graph Embeddings

Zhiping Luo[1,4*], **Wentao Xu**[1,4*], **Weiqing Liu**[2], **Jiang Bian**[2], **Jian Yin**[3,4†], **and Tie-Yan Liu**[2]
[1] School of Computer Science and Engineering, Sun Yat-sen University, Guangzhou, China
[2] Microsoft Research Asia, Beijing, China
[3] School of Artificial Intelligence, Sun Yat-sen University, Zhuhai, China
[4] Guangdong Key Laboratory of Big Data Analysis and Processing, Guangzhou, China
{luozhp7@mail2,xuwt6@mail2,issjyin@mail}.sysu.edu.cn
{weiqing.liu, jiang.bian, tyliu}@microsoft.com

Abstract

Learning the embeddings of knowledge graphs (KG) is vital in artificial intelligence, and can benefit various downstream applications, such as recommendation and question answering. In recent years, many research efforts have been proposed for knowledge graph embedding (KGE). However, most previous KGE methods ignore the semantic similarity between the related entities and entity-relation couples in different triples since they separately optimize each triple with the scoring function. To address this problem, we propose a simple yet efficient contrastive learning framework for tensor decomposition based (TDB) KGE, which can shorten the semantic distance of the related entities and entity-relation couples in different triples and thus improve the performance of KGE. We evaluate our proposed method on three standard KGE datasets: WN18RR, FB15k-237 and YAGO3-10. Our method can yield some new state-of-the-art results, achieving 51.2% MRR, 46.8% Hits@1 on the WN18RR dataset, 37.8% MRR, 28.6% Hits@1 on FB15k-237 dataset, and 59.1% MRR, 51.8% Hits@1 on the YAGO3-10 dataset. Source codes and data of this paper can be found at https://github.com/Wentao-Xu/KGE-CL.

1 Introduction

The knowledge graph (KG) stores a vast number of human knowledge in the format of triples. A triple (h, r, t) in a KG contains a head entity h, a tail entity t, and a relation r between h and t. The knowledge graph embedding (KGE) aims to project the massive interconnected entities and relations in a KG into vectors or matrices, which can preserve the semantic information of the triples. Learning the embeddings of KG can benefit various downstream artificial intelligence applications, such as question answering (Huang et al., 2019), machine reading comprehension (Yang and Mitchell, 2017), image classification (Marino et al., 2016), and personalized recommendation (Wang et al., 2018).

In general, most of the KGE methods would define a scoring function $f(h_i, r_j, t_k)$, and the training target of KGE is maximizing the score of a true triple (h_i, r_j, t_k) and minimizing the score of a false tripe (h_i, r_j, t_x). In this way, the trained embeddings of entities and relations in KG can preserve the intrinsic semantics of a true triple. We can mainly divide the existing KGE methods into two categories. The first category is the distance based (DB) methods, which use the Minkowski distance as scoring function to measure a triple's plausibility, include the TransE (Bordes et al., 2013), TransH (Wang et al., 2014), TransR (Lin et al., 2015), TransD (Ji et al., 2015) and TransG (Xiao et al., 2016). The other category is the tensor decomposition based (TDB) methods, which treat a KG as a third-order binary tensor and use the results of tensor decomposition as the representations of entities and relations. The TDB methods include the CP (Hitchcock, 1927), DistMult (Yang et al., 2015), RESCAL (Nickel et al., 2011) and ComplEx (Trouillon et al., 2016).

However, existing methods only capture the semantic connections among h, r and t in a same triple. For example, the TransE optimize the distance between $h + r$ and t, and the DistMult optimizie the dot product similarity among h, r and t. Therefore, they overlook the connections between the related entities and entity-relation couples in different triples. In a KG, some entities (entity-relation couples) that share the same entity-relation couple (entity) are in the same type and have semantic similarity. Capturing the semantic similarity of these entities or couples can improve the expressiveness of embeddings, which is the performance in capturing semantic information in KG (Dettmers et al., 2018; Xu et al.,

*The first two authors contributed equally.
†Corresponding author.

Proceedings of the 29th International Conference on Computational Linguistics, pages 2598–2607
October 12–17, 2022.

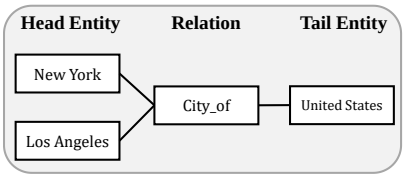

(a) Entities with the same couple.

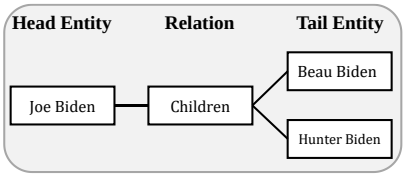

(b) Entities with the same couple.

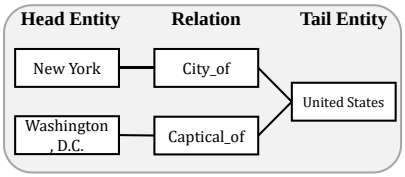

(c) Couples with the same tail entity United States.

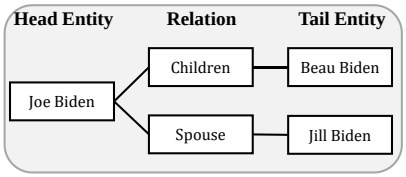

(d) Couples with the same head entity Joe Biden.

Figure 1: The examples of triples that share the same entities or entity-relation couples.

2020). For instance, in Figure 1 (a), the entities *New York* and *Los Angeles* share the same entity-relation couple (*City_of*, *United States*). Therefore, the entities *New York* and *Los Angeles* should have a similar embedding. Besides, in Figure 1 (c) the couples (*New York*, *City_of*) and (*Washington, D.C.*, *Captical_of*) share the same tail entity *United States*. Thus the representations of these two couples should also be similar.

To correlate the related entities and entity-relation couples in different triples, we propose a simple yet efficient contrastive learning framework called KGE-CL for KGE, which is quite general for existing TDB methods. We first construct the positive instances for those entities that share the same entity-relation couple and those entity-relation couples that share the same entity. For example, the positive instance of *Beau Biden* in Figure 1 (b) is the *Hunter Biden*, and the positive instance of (*Children*, *Beau Biden*) in Figure 1 (d) is (*Spouse*, *Jill Biden*). Then we calculate the contrastive loss on the original instance and the positive instances. Due to the design of the contrastive learning framework, we can also increase the distance between unrelated entities and couples. Finally, since each triple has four positive instances (corresponding to the four examples in Figure 1), we design a weighted contrastive loss to control the weights on different positive instances' loss flexibly.

We evaluate our KGE-CL method on the KG link prediction task using the standard WN18RR (Toutanova and Chen, 2015), FB15k-237 (Dettmers et al., 2018) and YAGO3-10 (Mahdisoltani et al., 2015) datasets. Our proposed method achieves new state-of-the-art results (SotA), obtaining 51.2% MRR, 46.8% Hits@1 on the WN18RR dataset, 37.8% MRR, 28.6% Hits@1 on the FB15k-237 dataset, and 59.1% MRR, 51.8% Hits@1 on the YAGO3-10 dataset. Moreover, We apply several experiments to further analyze the inner mechanism of our method. At last, to clearly explain why our method outperforms existing methods, we conduct the visualization of the KGE of our method and some compared methods using T-SNE (van der Maaten and Hinton, 2008).

In summary, this paper's contributions include:

- We propose KGE-CL, a simple yet efficient contrastive learning framework for TDB KGE. It can capture the semantic similarity of the related entities and entity-relation couples in different triples, thus improving the performance of KGE.
- Our proposed KGE-CL framework can also push the embeddings of unrelated entities and couples apart in the semantic space.
- The experiment results and analyses confirm the effectiveness of our KGE-CL method.

2 Related Work

In recent years, knowledge graph embedding (KGE) becomes a pretty hot research topic since its vital role in various downstream applications. We can categorize the existing KGE techniques into two categories: the distance based KGE and tensor decomposition based KGE.

Distance based (DB) methods describe relations as relational maps between head and tail entities. The TransE is a representative distance based method, which uses the relations as translations and its scoring function is: $f(h_i, r_j, t_k) = -||\mathbf{h}_i +$

$\mathbf{r}_j - \mathbf{t}_k||_2^2$. To improve the performance of TransE, many its variants that follow the same direction were proposed, such as the TransH (Wang et al., 2014), TransR (Lin et al., 2015), TransD (Ji et al., 2015), TranSparse (Ji et al., 2016), TransG (Xiao et al., 2016) and RotatE (Sun et al., 2019). However, the TransE and its extensions can not capture the semantic similarity between the related entities and entity-relation couples in different triples. For example, given two triples (h_1, r_1, t_1) and (h_1, r_1, t_2), TransE can only close the distance between $\mathbf{h}_1 + \mathbf{r}_1$ and $\mathbf{t}_1$ (or $\mathbf{t}_2$) in the same triple, it does not close the distance between $\mathbf{t}_1$ and $\mathbf{t}_2$, so the representations $\mathbf{t}_1$ and $\mathbf{t}_2$ can in different directions.

Tensor decomposition based (TDB) methods formulate the KGE task as a third-order binary tensor decomposition problem. RESCAL (Nickel et al., 2011) factorizes the j-th frontal slice of $\mathcal{X}$ as $\mathcal{X}_j \approx \mathbf{A}\mathbf{R}_j\mathbf{A}^\top$, in which embeddings of head and tail entities are from the same space. As the relation embeddings in RESCAL are matrices containing lots of parameters, RESCAL is easier to be overfitting and more difficult to train. DistMult (Yang et al., 2015) simplifies the matrix $\mathbf{R}_j$ in RESCAL to a diagonal matrix, while the RESCAL can only preserve the symmetry of relations, limiting its expressiveness. To model asymmetric relations, ComplEx (Trouillon et al., 2016) extends DistMult to complex embeddings and preserving the relations' symmetry in the real part and the asymmetry in the imaginary part. Moreover, the QuatE (Zhang et al., 2019) further extends the ComplEx to hypercomplex space to model more complicated relation properties, such as the inversion. All of the DistMult, ComplEx, and QuatE are the variants of CP decomposition (Hitchcock, 1927), which are in real, complex, and hypercomplex vector spaces, respectively. On the other hand, the TDB methods usually suffer from an overfitting problem; thus, some work is trying to improve the TDB methods from the aspect of regularizer, such as the N3 (Lacroix et al., 2018) and DURA (Zhang et al., 2020a) regularizers. These regularizers bring more significant improvements than the original squared Frobenius norm (L_2 norm) regularizer (Nickel et al., 2011; Yang et al., 2015; Trouillon et al., 2016). Nevertheless, the TDB methods can only capture the similarity among h, r and t in a same triple. For example, the ComplEx-DURA (Zhang et al., 2020a) mainly capture the semantics similarity between the couple (h_i, r_j)'s embedding $\mathbf{h}_i\mathbf{R}_j$ and tail entity t_k's embedding $\mathbf{t}_k$ in a sample triple.

Since both of the DB and TDB methods can not correctly capture the semantic similarity between related entities and couples in different triples, we propose our KGE-CL method to address the limitations of existing work.

3 Preliminaries

3.1 Knowledge Graph Embedding

Knowledge Graph Embedding (KGE) The knowledge graph embedding (KGE) is to learn the representations (may be real or complex vectors, matrices, and tensors) of the entities and relations. Its target is that the learned entities' and relations' embeddings can preserve the semantic information of the triples in knowledge graphs. Generally, the KGE methods define a scoring function $f(h_i, r_j, t_k)$ to score the corresponding triple (h_i, r_j, t_k), and the score measure the plausibility of triples.

Tensor Decomposition Based (TDB) KGE TDB methods like RESCAL (Nickel et al., 2011) and ComplEx (Trouillon et al., 2016), regard a KG as a third-order binary tensor $\mathcal{X} \in \{0,1\}^{|\mathcal{E}|\times|\mathcal{R}|\times|\mathcal{E}|}$. The (i, j, k) entry $\mathcal{X}_{ijk} = 1$ if (h_i, r_j, t_k) is a true triple otherwise $\mathcal{X}_{ijk} = 0$. The $\mathcal{X}_j$ denotes the j-th frontal slice of $\mathcal{X}$, that is, the corresponding matrix of the j-th relation. Generally, a TDB KGE model factorizes $\mathcal{X}_j$ as $\mathcal{X}_j \approx \mathbf{Re}\left(\mathbf{H}\mathbf{R}_j\overline{\mathbf{T}}^\top\right)$, where the i-th (k-th) row of $\mathbf{H}$ ($\mathbf{T}$) is $\mathbf{h}_i$ ($\mathbf{t}_k$), $\mathbf{R}_j$ is a matrix that represents relation r_j, $\mathbf{Re}\left(\cdot\right)$ and $\bar{\cdot}$ are the real part and the conjugate of a complex matrix, respectively. Then the scoring functions of TDB KGE methods is: $f(h_i, r_j, t_k) = \mathbf{Re}\left(\mathbf{h}_i\mathbf{R}_j\bar{\mathbf{t}}_k^\top\right)$. Note that the real part and the conjugate of a real matrix are itself. The goal of TDB models is to seek matrices $\mathbf{H}, \mathbf{R}_1, \ldots, \mathbf{R}_{|\mathcal{R}|}, \mathbf{T}$, such that $\mathbf{Re}\left(\mathbf{H}\mathbf{R}_j\overline{\mathbf{T}}^\top\right)$ can approximate $\mathcal{X}_j$. In this paper, we aim to improve the performance of existing TDB models, such as the RESCAL and ComplEx models.

3.2 Contrastive Learning

Contrastive learning is an efficient representation learning method that contrasts positive pairs against negative pairs (Hadsell et al., 2006; He et al., 2020; Chen et al., 2020; Khosla et al., 2020). The key idea of contrastive learning is pulling the semantically close pairs together and push apart the negative pairs. The unsupervised contrastive learning framework (Chen et al., 2020) would utilize the data augmentation to construct positive pairs to calculate the contrastive loss. The supervised contrastive

learning framework (Khosla et al., 2020) calculates the contrastive loss of all positive instances within the same mini-batch. Motivated by these exiting frameworks, we adopt the following function to calculate the contrastive loss between an instance z_i and its all positive instances z_i^+:

$$\mathrm{CL}\left(\mathbf{z}_i, \mathbf{z}_i^+\right) = \frac{-\sum_{z_i^+ \in P(i)} \log \frac{e^{\mathrm{sim}\left(\mathbf{z}_i, \mathbf{z}_i^+\right)/\tau}}{\sum_{z_j \in N(i)} e^{\mathrm{sim}\left(\mathbf{z}_i, \mathbf{z}_j\right)/\tau}}}{\mid P(i) \mid}, \tag{1}$$

where $\mathbf{z}_i$ and $\mathbf{z}_i^+$ are the representations of z_i and z_i^+, respectively. $P(i)$ is the set of all positive instances in the mini-batch, and $N(i)$ is the set of all negative instances in the batch. In our work, we define the negative instances as the instances that do not belong to the positive instances. $\mathrm{sim}(\mathbf{z}_i, \mathbf{z}_i^+) = \mathbf{z}_i \cdot \mathbf{z}_i^+$ is the dot product similarity.

4 Our Method: KGE-CL

In this section, we describe our KGE-CL method that utilize the contrastive learning to capture the semantic similarity of related entities and couples in different triples. Our method is very general and can be easily apply to arbitrary TDB methods. We can further name our KGE-CL method as RESCAL-CL or ComplEx-CL when we use the scoring function of RESCAL or ComplEx models. In this section, we firstly present the contrastive loss we designed for the KGE, then we introduce the training objective of our method.

4.1 Contrastive Loss of KGE

In this subsection, we elaborate on the contrastive loss that we designed for KGE.

Positive Instances The generation of positive instances z_i^+ for the instance z_i is vital in contrastive learning. Existing work in visual representation learning (Wu et al., 2018; Chen et al., 2020; Chen and He, 2020) used some data augmentation methods, such as cropping, color distortion, and rotation, to take two random transformations of the same images as z_i and z_i^+. Meanwhile, in NLP, some work (Wu et al., 2020; Meng et al., 2021) utilized other augmentation techniques like word deletion, reordering, and substitution. However, these data augmentation methods are not proper to the KGE. To capture the interactions between triples in a KG, we design a new approach to construct the positive instances for KGE. For a triple (h_i, r_j, t_k), the corresponding scoring function of TDB methods is:

$$\begin{aligned} f(h_i, r_j, t_k) &= \mathbf{Re}\left(\mathbf{h}_i \mathbf{R}_j \bar{\mathbf{t}}_k^\top\right) = \mathbf{Re}\left(\langle \mathbf{h}_i \mathbf{R}_j, \bar{\mathbf{t}}_k \rangle\right) \\ &= \mathbf{Re}\left(\langle \mathbf{h}_i, \bar{\mathbf{t}}_k \mathbf{R}_j^\top \rangle\right). \end{aligned} \tag{2}$$

We define $\langle \cdot, \cdot \rangle$ as the inner product of two real or complex vectors: $\langle \mathbf{u}, \mathbf{v} \rangle = \mathbf{u}\mathbf{v}^\top$. The $\mathbf{h}_i\mathbf{R}_j$ and $\bar{\mathbf{t}}_k\mathbf{R}_j^\top$ are the representations of the entity-relation couples (h_i, r_j) and (r_j, t_k), respectively. The Equation 2 means that we can firstly compute either the $\mathbf{h}_i\mathbf{R}_j$ or $\bar{\mathbf{t}}_k\mathbf{R}_j^\top$ in the scoring function. For a head entity h_i, we define its positive instances h_i^+ as those head entities that share the same relation and tail entity with h_i. Similarly, we define the tail entity t_k's positive instances t_k^+ with those tail entities that share the same head entity and relation with t_k. For the entity-relation couples (h_i, r_j) or (r_j, t_k), the corresponding positive instances $(h_i, r_j)^+$ or $(r_j, t_k)^+$ is those entity-relation couples that share the same tail entity with (h_i, r_j) or head entity with (r_j, t_k). The $(\mathbf{h}_i\mathbf{R}_j)^+$ and $(\bar{\mathbf{t}}_k\mathbf{R}_j^\top)^+$ are the representations of positive instances $(h_i, r_j)^+$ and $(r_j, t_k)^+$, respectively. Therefore, given a true triple (h_i, r_j, t_k), our method would construct four kinds of positive instances: h_i^+, t_k^+, $(h_i, r_j)^+$ and $(r_j, t_k)^+$, which are corresponding to the four examples in Figure 1. As mentioned in Section 3.2, for an instance h_i, we will use all of its positive instances h_i^+ in the same mini-batch. Besdies, there may be no positive instances in a mini-batch for some tripels. Therefore, we will add a postive instance for each triple in the mini-batch, where the added instance is randomly sampled from training set. We will study the effect of postive instances by experiment in Section 5.3.

Contrastive Loss Given a true triple (h_i, r_j, t_k), we use the Equation 1 to compute the contrastive loss of four types of positive instances, and the overall contrastive loss for a triple (h_i, r_j, t_k) is:

$$\begin{aligned} \mathcal{L}_c(h_i, r_j, t_k) &= \mathrm{CL}(\mathbf{h}_i, \mathbf{h}_i^+) + \mathrm{CL}(\mathbf{t}_k, \mathbf{t}_k^+) \\ &+ \mathrm{CL}(\mathbf{h}_i\mathbf{R}_j, (\mathbf{h}_i\mathbf{R}_j)^+) + \mathrm{CL}(\bar{\mathbf{t}}_k\mathbf{R}_j^\top, (\bar{\mathbf{t}}_k\mathbf{R}_j^\top)^+). \end{aligned} \tag{3}$$

Theoretical Analysis To study how the Equation 3 takes effect, we apply a theoretical analysis from the aspect of gradient. Taking the contrastive loss term $\mathrm{CL}(\mathbf{h}_i, \mathbf{h}_i^+)$ as an example, the gradient

Contrastive Loss	WN18RR				FB15k-237			
	MRR	H@1	H@3	H@10	MRR	H@1	H@3	H@10
$\mathrm{CL}(\mathbf{h}_i, \mathbf{h}_i^+)$	.509	.465	**.527**	.588	.376	.284	.412	.557
$\mathrm{CL}(\mathbf{t}_k, \mathbf{t}_k^+)$	.509	.468	.425	.587	.376	.285	.411	.558
$\mathrm{CL}(\mathbf{h}_i\mathbf{R}_j, (\mathbf{h}_i\mathbf{R}_j)^+)$	**.512**	**.469**	**.527**	**.595**	.374	.282	.410	.557
$\mathrm{CL}(\bar{\mathbf{t}}_k\mathbf{R}_j^\top, (\bar{\mathbf{t}}_k\mathbf{R}_j^\top)^+)$	.504	.462	.516	.580	**.378**	**.286**	**.414**	**.559**

Table 1: Link prediction results of RESCAL-CL that merely uses one specific contrastive loss term.

of $\mathrm{CL}(\mathbf{h}_i, \mathbf{h}_i^+)$ to embedding $\mathbf{h}_i$ is:

$$\frac{\partial \mathrm{CL}(\mathbf{h}_i, \mathbf{h}_i^+)}{\partial \mathbf{h}_i} = \frac{\sum_{h_i^+ \in P(i)} \partial \left(\mathbf{h}_i \cdot \mathbf{h}_i^+ / \tau - \log \sum_{h_j \in N(i)} e^{(\mathbf{h}_i \cdot \mathbf{h}_j / \tau)} \right)}{-|P(i)| \partial \mathbf{h}_i} = -\frac{\sum_{h_i^+ \in P(i)} \mathbf{h}_i^+}{\tau |P(i)|} + \frac{\sum_{h_j \in N(i)} \left(e^{(\mathbf{h}_i \cdot \mathbf{h}_j / \tau)} \mathbf{h}_j \right)}{\tau \sum_{h_j \in N(i)} e^{(\mathbf{h}_i \cdot \mathbf{h}_j / \tau)}}. \tag{4}$$

Then when we update the embedding $\mathbf{h}_i$ with the gradient $\frac{\partial \mathrm{CL}(\mathbf{h}_i, \mathbf{h}_i^+)}{\partial \mathbf{h}_i}$:

$$\mathbf{h}_i^{t+1} = \mathbf{h}_i^t - \eta \frac{\partial \mathrm{CL}(\mathbf{h}_i, \mathbf{h}_i^+)}{\partial \mathbf{h}_i}, \tag{5}$$

where η of learning rate for updating gradient. The Equation 4 and Equation 5 show that the embedding $\mathbf{h}_i^t$ would update to the direction of $\frac{\eta \sum_{h_i^+ \in P(i)} \mathbf{h}_i^+}{\tau |P(i)|}$, which is the mean value of positive instances' embeddings $\mathbf{h}_i^+$. Meanwhile, the $\mathbf{h}_i^t$ would also update away from the weighted value $\frac{\eta \sum_{h_j \in N(i)} \left(e^{(\mathbf{h}_i \cdot \mathbf{h}_j / \tau)} \mathbf{h}_j \right)}{\tau \sum_{h_j \in N(i)} e^{(\mathbf{h}_i \cdot \mathbf{h}_j / \tau)}}$ of negative instances $\mathbf{h}_j$. So our proposed contrastive loss $\mathcal{L}_c$ can not only pull the related entities and entity-relation couples together in the semantic space but also push the unrelated entities and couples apart.

Weighted Contrastive Loss There are four contrastive loss terms in Equation 3. We found that different contrastive loss terms have different effects on different knowledge graphs during our research process. This phenomenon happens may because different knowledge graphs have diverse graph properties, such as the ratio of the number of entities to the number of relations, the number of triples compared with the number of entities. Table 1 shows the results of RESCAL-CL that merely uses one specific contrastive loss term. In WN18RR dataset, using the term $\mathrm{CL}(\mathbf{h}_i\mathbf{R}_j, (\mathbf{h}_i\mathbf{R}_j)^+)$ achieves the highest results, while in FB15k-237 dataset, $\mathrm{CL}(\bar{\mathbf{t}}_k\mathbf{R}_j^\top, (\bar{\mathbf{t}}_k\mathbf{R}_j^\top)^+)$ is the best. Hence, we introduce a weighted contrastive loss, assigning a weight α_* for each contrastive loss term, and α_* is a hyper-parameter that can be flexibly tuned for a specific KG. The contrastive loss $\mathcal{L}_c^w(h_i, r_j, t_k)$ after adding weights α_* is:

$$\begin{aligned} \mathcal{L}_c^w(h_i, r_j, t_k) = & \alpha_h \mathrm{CL}(\mathbf{h}_i, \mathbf{h}_i^+) + \alpha_t \mathrm{CL}(\mathbf{t}_k, \mathbf{t}_k^+) \\ & + \alpha_{hr} \mathrm{CL}(\mathbf{h}_i\mathbf{R}_j, (\mathbf{h}_i\mathbf{R}_j)^+) \\ & + \alpha_{tr} \mathrm{CL}(\bar{\mathbf{t}}_k\mathbf{R}_j^\top, (\bar{\mathbf{t}}_k\mathbf{R}_j^\top)^+). \end{aligned} \tag{6}$$

4.2 Training Objective

Given a training triple (h_i, r_j, t_k) in a KG, the instantaneous loss of our framework on this triple is:

$$\mathcal{L}(h_i, r_j, t_k) = \mathcal{L}_s + \mathcal{L}_r + \mathcal{L}_c^w, \tag{7}$$

where $\mathcal{L}_s$ is the loss that measures the discrepancy between scoring function's output $f(h_i, r_j, t_k)$ and the label $\mathcal{X}_{ijk}$. $\mathcal{L}_r$ is the regularizer, and $\mathcal{L}_c^w$ is the weighted contrastive loss we introduced in Section 4.1. It should be noted that the additionally added postive instances in the mini-batch is only used to calculate the contrastive loss $\mathcal{L}_c^w$ and would not used to calculate the $\mathcal{L}_s$ and $\mathcal{L}_r$ losses.

$\mathcal{L}_s$ Loss Many previous efforts used the ranking losses (Bordes et al., 2013), binary logistic regression (Trouillon et al., 2016) or sampled multiclass log-loss (Kadlec et al., 2017) to calculate the distance between the scoring function's output and the triple's label. Since (Lacroix et al., 2018) had verified the competitiveness of the full multiclass log-loss, we utilize it as the $\mathcal{L}_s$ loss in Equation 7.

Regularizer Most of the previous work use the squared Frobenius norm (L_2 norm) regularizer (Nickel et al., 2011; Yang et al., 2015; Trouillon et al., 2016) in their object functions. More recently, some work proposed more efficient regularizers to prevent the overfitting of KGE, such as the N3 (Lacroix et al., 2018) and DURA (Zhang et al., 2020a) regularizers. Since the (Zhang et al., 2020a) had shown that DURA regularizer outperforms L2 and N3 regularizers, we use the DURA as the regularizer $\mathcal{L}_r$ in our work.

5 Experiment

In this section, we present thorough empirical studies to evaluate and analyze our proposed framework. We first introduce the experimental setting.

	WN18RR	FB15k-237	YAGO3-10
#Entity	40,943	14,541	123,182
#Relation	11	237	37
#Train	86,835	272,115	1,079,040
#Valid	3,034	17,535	5,000
#Test	3,134	20,466	5,000

Table 2: Statistics of the datasets.

Datasets	Methods	d	m	τ	α_h	α_t	α_{hr}	α_{tr}
WN18RR	RESCAL-CL	512	512	0.9	0	0	2.0	0
	ComplEx-CL	2000	2048	0.5	0	0	0	2.0
FB15k-237	RESCAL-CL	512	512	0.9	0	0	0	2.0
	ComplEx-CL	2000	2048	0.5	2.0	0	0	0
YAGO3-10	RESCAL-CL	512	512	0.9	0	0	0	1.0
	ComplEx-CL	2000	2048	0.5	0	1.0	0	0

Table 3: The selection of the hyper-parameters of RESCAL-CL and ComplEx-CL on different datasets.

Then we evaluate our framework's performance on the task of link prediction. Besides, we further analyze the details of our promotion by comparing our method with a baseline on the triples with different relations, and we also study the effect of positive instances to our framework. Finally, we visualize the embeddings of our method and some baselines to explain why our method outperforms baselines.

5.1 Exeprimental Setting

Dataset We use three standard KG datasets—WN18RR (Toutanova and Chen, 2015), FB15k-237 (Dettmers et al., 2018), and YAGO3-10 (Mahdisoltani et al., 2015) to evaluate the performance of KGE. We divide the datasets into training, validating, and testing sets using the same way of previous work. Table 2 shows the statistics of these datasets. WN18RR, FB15k-237, and YAGO3-10 are extracted from WN18 (Bordes et al., 2013), FB15k (Bordes et al., 2013), and YAGO3 (Mahdisoltani et al., 2015), respectively. Some previous work (Toutanova and Chen, 2015; Dettmers et al., 2018) indicated the test set leakage problem in WN18 and FB15k, where some test triplets may appear in the training dataset in the form of reciprocal relations. Therefore, they suggested using the WN18RR and FB15k-237 datasets to avoid the test set leakage problem.

Compared Methods We compare our KGE-CL method with existing state-of-the-art KGE methods, including CP (Hitchcock, 1927), RESCAL (Nickel et al., 2011), ComplEx (Trouillon et al., 2016), ConvE (Dettmers et al., 2018), RotatE (Sun et al., 2019), MuRP (Balazevic et al., 2019), HAKE (Zhang et al., 2020b), ComplEx-N3 (Lacroix et al., 2018), ROTH (Chami et al., 2020), REFE (Chami et al., 2020), CP-DURA (Zhang et al., 2020a), RESCAL-DURA (Zhang et al., 2020a) and ComplEx-DURA (Zhang et al., 2020a).

Implementation Details We implement our method base on the PyTorch library (Paszke et al., 2019), and run on all experiments with a single NVIDIA Tesla V100 GPU. We leverage Adagrad algorithm (Duchi et al., 2011) to optimize the objective function in Equation 7. We tune our model using the grid search to select the optimal hyper-parameters based on the performance on the validation dataset. We search the embedding size d in {256, 512, 1024} for RESCAL-CL and {200, 500, 1000, 2000} for ComplEx-CL. We search the temperature τ in Equation 1 in {0.3, 0.5, 0.7, 0.9, 1.0}. We search the weights α_h, α_t, α_{hr} and α_{tr} in Equation 6 in {0, 0.2, 0.4, 0.6, 0.8, 1.0, 1.5, 2.0, 2.5}. The best choices of hyper-parameters, the number of parameters, and the training time of RESCAL-CL and ComplEx-CL on each dataset are listed in Table 3. For a fair comparison, the RESCAL-CL and ComplEx-CL have the same embedding size as RESCAL-DURA and ComplEx-DURA, respectively. Besides, the batch size is 512 for RESCAL-CL and 200 for ComplEx-CL, and the learning rate η as 0.1 for all methods. On WN18RR, we set the number of training epochs as 50 for the ComplEx-CL and 200 for the RESCAL-CL. On FB15k-237 and YAGO3-10, the number of training epochs is 200 for all methods.

5.2 Main Results

We evaluate the performance of our framework on the link prediction task, which is a frequently-used task to evaluate the KGE. Specifically, we replace the head or tail entity of a true triple in the test set with other entities in the dataset and name these derived triples as *corrupted triples*. The link prediction task aims to score the original true triples higher than the corrupted ones. We rank the triples by the results of the scoring function.

The evaluation metrics we used in the link prediction are the MRR and Hits@N: 1) MRR: the mean reciprocal rank of original triples; 2) Hits@N: the percentage rate of original triples ranked at the top

Methods	WN18RR				FB15k-237				YAGO3-10			
	MRR	H@1	H@3	H@10	MRR	H@1	H@3	H@10	MRR	H@1	H@3	H@10
CP	.438	.414	.445	.485	.333	.247	.363	.508	.567	.494	.611	.698
RESCAL	.455	.419	.461	.493	.353	.264	.385	.528	.566	.490	.612	.701
ComplEx	.460	.428	.473	.522	.346	.256	.386	.525	.573	.500	.617	.703
ConvE	.43	.40	.44	.52	.325	.237	.356	.501	.44	.35	.49	.62
RotatE	.476	.428	.492	.571	.338	.241	.375	.533	.495	.402	.550	.670
MuRP	.481	.440	.495	.566	.335	.243	.367	.518	-	-		-
HAKE	.497	.452	.516	.582	.346	.250	.381	.542	.546	.462	.596	.694
ComplEx-N3	.491	.448	.505	.580	.366	.271	.403	.558	.577	.502	.619	.711
ROTH	.496	.449	.514	.586	.344	.246	.380	.535	.570	.495	.612	.706
REFE	.473	.430	.485	.561	.351	.256	.390	.541	.577	.503	.621	.712
CP-DURA	.478	.441	.497	.552	.367	.272	.402	.555	.582	.511	.623	.708
RESCAL-DURA	.498	.455	.514	.577	.368	.276	.402	.550	.579	.505	.619	.712
ComplEx-DURA	.491	.449	.504	.571	.371	.276	.408	.560	.584	.511	.628	.713
RESCAL-CL	**.512**	**.468**	**.531**	**.597**	**.378**	**.286**	**.414**	.559	.581	.507	.625	.713
ComplEx-CL	.505	.458	.522	.595	.377	.285	**.414**	**.564**	**.591**	**.518**	**.634**	**.722**

Table 4: Link prediction results on WN18RR, FB15k-237 and YAGO3-10 datasets. We take the results of CP, RESCAL, ComplEx, CP-DURA, RESCAL-DURA and ComplEx-DURA from the paper (Zhang et al., 2020a), and the results of other baselines are from their original papers.

N in prediction. For both metrics, we remove some of the corrupted triples that exist in datasets from the ranking results, which is also called *filtered* setting in (Bordes et al., 2013). For the metrics of Hits@N, we use Hits@1, Hits@3, and Hits@10.

Table 4 shows the results of link prediction on WN18RR, FB15K-237, and YAGO3-10 datasets. Our proposed method achieves the highest results on all datasets compared with the baselines. Specifically, the RESCAL-CL achieves evidently better results on the WN18RR dataset. The ComplEx-CL outperforms the compared methods in the YAGO3-10 dataset. On FB15k-237, the RESCAL-CL and ComplEx-CL are both better than the RESCAL-DURA and ComplEx-DURA, respectively. The results of RESCAL-CL and ComplEx-CL verify that correlating the entities and entity-relation couples in different triples can boost the performance of KGE.

5.3 Model Analysis

Analyzing the Improvements To further explore why our method outperforms existing state-of-the-art techniques, we compare our RESCAL-CL method with the RESCAL-DURA on the triples with different relations in WN18RR. Table 5 shows the results of the comparison, and we found that our RESCAL-CL is significantly better than the RESCAL-DURA in 9 out of the 11 relations, verifying that the promotion of our framework is extensive and not just on some specific relations.

Relations	#Train	#Test	RESCAL-DURA			RESCAL-CL		
			MRR	H@1	H@10	MRR	H@1	H@10
_similar_to	80	3	0.446	0.333	0.667	**0.756**	**0.667**	**1.000**
_verb_group	1138	39	0.930	0.885	0.974	0.934	**0.897**	0.974
*domain_usage	629	24	0.400	0.354	0.542	**0.447**	**0.396**	0.542
*domain_region	923	26	0.329	0.269	0.442	**0.360**	**0.289**	**0.500**
_member_meronym	7402	253	0.251	0.164	0.415	**0.318**	**0.221**	**0.506**
_has_part	4816	172	0.223	0.151	0.375	**0.245**	**0.174**	**0.384**
_hypernym	34796	1251	0.193	0.140	0.288	**0.204**	**0.152**	**0.296**
_instance_hypernym	2921	122	0.431	0.348	0.603	**0.461**	**0.369**	**0.631**
_synset_domain*	3116	114	0.405	0.347	0.522	**0.444**	**0.395**	**0.544**
*related_form	29715	1074	0.957	0.951	0.967	0.959	0.954	0.969
_also_see	1299	56	0.606	0.554	0.679	**0.621**	**0.571**	**0.696**

Table 5: MRR, Hit@1 and Hit@10 results of RESCAL-DURA and RESCAL-CL methods on the triples with different relations in WN18RR dataset. We use * to represent the abbreviation of some words in the relation names. #Train and #Test are the number of triples with the corresponding relations in the training and test set.

Effect of Positive Instances We apply an ablation study to verify the effect of positive instances. The variants of w/o Pos are the variants of RESCAL-CL and ComplEx-CL, which remove all original and additionally added positive instances in a mini-batch when calculating the contrastive loss. Table 6 shows the results of ablation study on WN18RR and FB15k-237 datasets. From Table 6 we know the positive instance of KGE can significantly improve the performance of KGE. Therefore, the positive instances we constructed are effective for the KGE.

Sparsity Analysis As mentioned by some other work (Zhang et al., 2020a), the sparsity (the number of zero entries) of embeddings can save the

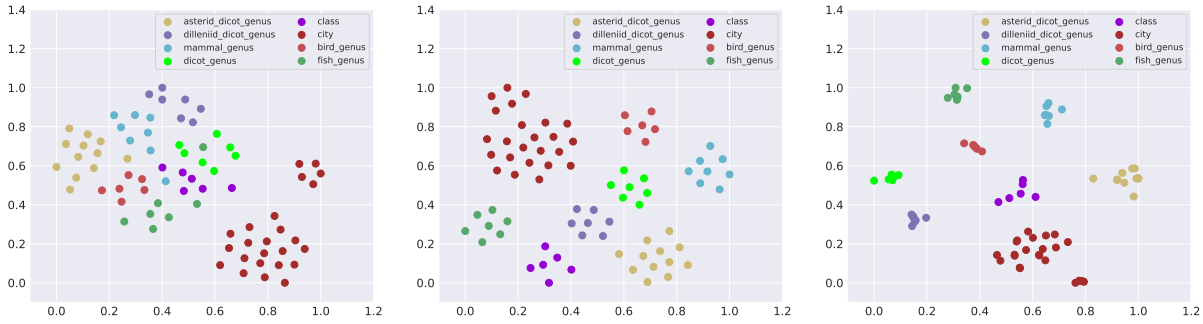

(a) RESCAL (b) RESCAL-DURA (c) RESCAL-CL

Figure 2: The visualization of the entity-relation couples' embeddings using T-SNE. A points represents a (h_i, r_j) couple, and the points with the same color are the couples that connected with the same tail entity.

Methods	Variants	WN18RR		FB15k-237	
		MRR	Hit@10	MRR	Hit@10
RESCAL -CL	w/o Pos	0.501	0.581	0.368	0.551
	w/ Pos	**0.512**	**0.597**	**0.378**	**0.559**
ComplEx -CL	w/o Pos	0.493	0.579	0.370	0.560
	w/ Pos	**0.505**	**0.595**	**0.377**	**0.564**

Table 6: Effect of Positive Instances.

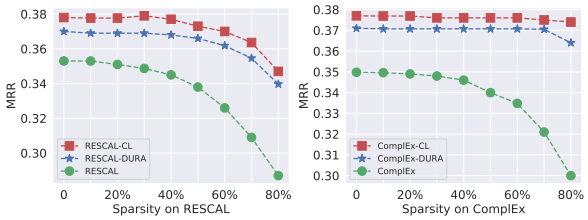

Figure 3: Effect of sparsity on FB15k-237.

storage usage of KG, which is vital for large scale real-world KGs. Therefore, we analyze the sparsity of embeddings trainng by our contrastive learning (CL) framework. We follow the setting in DURA (Zhang et al., 2020a), using a threshold λ to mask a proportion of entries as zero, and observe the MRR results under different proportions. Figure 3 shows the effect of embeddings' sparsity on FB15k-237. We can find that the embeddings trained by CL have better results than DURA in the sparse version, so CL can better reduce the storage usage of KG and benifit the real-world KGs.

5.4 Visualization

To make our method more explainable, we visualize the entity-relation couples via T-SNE (van der Maaten and Hinton, 2008). Specifically, we randomly pick up eight tail entities in WN18RR. We find out the triples with these tail entities in the test set, and extract (h_i, r_j) couples in these triples. We visualize these couples' embeddings trained by the RESCAL, RESCAL-DURA, and RESCAL-CL.

Figure 2 shows the results of visualization. The RESCAL method in Figure 2 (a) can not properly separate the couples with different tail entities. Compared with RESCAL, the RESCAL-DURA in Figure 2 (b) can relatively better separate the couples with different tail entities. However, since RESCAL-DURA can not capture the semantic similarity of couples with the same entity, the distribution of the couples connected with the same tail entity is still wide. Our RESCAL-CL can well split the couples in different types and shorten the distance of the couples connected with the same entity. Hence, our RESCAL-CL can better preserve the semantic information of the triples in knowledge graphs and has a higher performance.

6 Conclusion and Future Work

In this paper, we propose a simple yet efficient contrastive learning framework for TDB KGE to improve its performance. Compared with the previous work, our method can pull the related entities and entity-relation couples in different triples together in the semantic space and push the unrelated entities and couples apart. The experimental results on the standard datasets show that our method can achieve new state-of-the-art results. Our analyses further verify the effectiveness of our approach.

In the future, we plan to extend the critical insights of contrastive learning to distance based (DB) KGE methods and other representation learning problems in natural language processing.

Acknowledgments

Zhiping Luo, Wentao Xu and Jian Yin are supported by the National Natural Science Foundation of China (U1811264, U1811262, U1811261, U1911203, U2001211), Guangdong Basic and Applied Basic Research Foundation (2019B1515130001), Key-Area Research and Development Program of Guangdong Province (2018B010107005, 2020B0101100001).

References

Ivana Balazevic, Carl Allen, and Timothy Hospedales. 2019. Multi-relational poincaré graph embeddings. In *Advances in Neural Information Processing Systems 32*, pages 4463–4473.

Antoine Bordes, Nicolas Usunier, Alberto Garcia-Duran, Jason Weston, and Oksana Yakhnenko. 2013. Translating embeddings for modeling multi-relational data. In *Advances in Neural Information Processing Systems 26*, pages 2787–2795. Curran Associates, Inc.

Ines Chami, Adva Wolf, Da-Cheng Juan, Frederic Sala, Sujith Ravi, and Christopher Ré. 2020. Low-dimensional hyperbolic knowledge graph embeddings. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 6901–6914, Online. Association for Computational Linguistics.

Ting Chen, Simon Kornblith, Mohammad Norouzi, and Geoffrey Hinton. 2020. A simple framework for contrastive learning of visual representations. In *International conference on machine learning*, pages 1597–1607. PMLR.

Xinlei Chen and Kaiming He. 2020. Exploring simple siamese representation learning. *arXiv preprint arXiv:2011.10566*.

Tim Dettmers, Pasquale Minervini, Pontus Stenetorp, and Sebastian Riedel. 2018. Convolutional 2d knowledge graph embeddings. In *Proceedings of the 32th AAAI Conference on Artificial Intelligence*, pages 1811–1818. AAAI Press.

John Duchi, Elad Hazan, and Yoram Singer. 2011. Adaptive subgradient methods for online learning and stochastic optimization. *Journal of Machine Learning Research*, 12(Jul):2121–2159.

Raia Hadsell, Sumit Chopra, and Yann LeCun. 2006. Dimensionality reduction by learning an invariant mapping. In *2006 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'06)*, volume 2, pages 1735–1742. IEEE.

Kaiming He, Haoqi Fan, Yuxin Wu, Saining Xie, and Ross Girshick. 2020. Momentum contrast for unsupervised visual representation learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 9729–9738.

Frank L Hitchcock. 1927. The expression of a tensor or a polyadic as a sum of products. *Journal of Mathematics and Physics*, 6(1-4):164–189.

Xiao Huang, Jingyuan Zhang, Dingcheng Li, and Ping Li. 2019. Knowledge graph embedding based question answering. In *Proceedings of the Twelfth ACM International Conference on Web Search and Data Mining*, pages 105–113.

Sergey Ioffe and Christian Szegedy. 2015. Batch normalization: Accelerating deep network training by reducing internal covariate shift. In *International conference on machine learning*, pages 448–456. PMLR.

Guoliang Ji, Shizhu He, Liheng Xu, Kang Liu, and Jun Zhao. 2015. Knowledge graph embedding via dynamic mapping matrix. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 687–696, Beijing, China. Association for Computational Linguistics.

Guoliang Ji, Kang Liu, Shizhu He, and Jun Zhao. 2016. Knowledge graph completion with adaptive sparse transfer matrix. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 30.

Rudolf Kadlec, Ondrej Bajgar, and Jan Kleindienst. 2017. Knowledge base completion: Baselines strike back. *arXiv preprint arXiv:1705.10744*.

Prannay Khosla, Piotr Teterwak, Chen Wang, Aaron Sarna, Yonglong Tian, Phillip Isola, Aaron Maschinot, Ce Liu, and Dilip Krishnan. 2020. Supervised contrastive learning. *arXiv preprint arXiv:2004.11362*.

Timothee Lacroix, Nicolas Usunier, and Guillaume Obozinski. 2018. Canonical tensor decomposition for knowledge base completion. In *Proceedings of the 35th International Conference on Machine Learning*, pages 2863–2872. PMLR.

Yankai Lin, Zhiyuan Liu, Maosong Sun, Yang Liu, and Xuan Zhu. 2015. Learning entity and relation embeddings for knowledge graph completion. In *Proceedings of the Twenty-Ninth AAAI Conference on Artificial Intelligence*, page 2181–2187. AAAI Press.

Farzaneh Mahdisoltani, Joanna Biega, and Fabian M. Suchanek. 2015. YAGO3: A knowledge base from multilingual wikipedias. In *Seventh Biennial Conference on Innovative Data Systems Research*.

Kenneth Marino, Ruslan Salakhutdinov, and Abhinav Gupta. 2016. The more you know: Using knowledge graphs for image classification. *arXiv preprint arXiv:1612.04844*.

Yu Meng, Chenyan Xiong, Payal Bajaj, Saurabh Tiwary, Paul Bennett, Jiawei Han, and Xia Song. 2021. Coco-lm: Correcting and contrasting text sequences for language model pretraining. *arXiv preprint arXiv:2102.08473*.

Maximilian Nickel, Volker Tresp, and Hans-Peter Kriegel. 2011. A three-way model for collective learning on multi-relational data. In *Proceedings of the 28th International Conference on Machine Learning*, volume 11, pages 809–816. PMLR.

Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, et al. 2019. Pytorch: An imperative style, high-performance deep learning library. *arXiv preprint arXiv:1912.01703*.

Zhiqing Sun, Zhi-Hong Deng, Jian-Yun Nie, and Jian Tang. 2019. Rotate: Knowledge graph embedding by relational rotation in complex space. In *International Conference on Learning Representations*.

Kristina Toutanova and Danqi Chen. 2015. Observed versus latent features for knowledge base and text inference. In *3rd Workshop on Continuous Vector Space Models and Their Compositionality*.

Théo Trouillon, Johannes Welbl, Sebastian Riedel, Eric Gaussier, and Guillaume Bouchard. 2016. Complex embeddings for simple link prediction. In *Proceedings of The 33rd International Conference on Machine Learning*, volume 48, pages 2071–2080. PMLR.

Laurens van der Maaten and Geoffrey Hinton. 2008. Visualizing data using t-SNE. *Journal of Machine Learning Research*, 9:2579–2605.

Hongwei Wang, Fuzheng Zhang, Xing Xie, and Minyi Guo. 2018. Dkn: Deep knowledge-aware network for news recommendation. In *Proceedings of the 2018 world wide web conference*, pages 1835–1844.

Zhen Wang, Jianwen Zhang, Jianlin Feng, and Zheng Chen. 2014. Knowledge graph embedding by translating on hyperplanes. In *Proceedings of the Twenty-Eighth AAAI Conference on Artificial Intelligence*, page 1112–1119. AAAI Press.

Zhirong Wu, Yuanjun Xiong, Stella X Yu, and Dahua Lin. 2018. Unsupervised feature learning via non-parametric instance discrimination. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3733–3742.

Zhuofeng Wu, Sinong Wang, Jiatao Gu, Madian Khabsa, Fei Sun, and Hao Ma. 2020. Clear: Contrastive learning for sentence representation. *arXiv preprint arXiv:2012.15466*.

Han Xiao, Minlie Huang, and Xiaoyan Zhu. 2016. TransG : A generative model for knowledge graph embedding. In *ACL*.

Wentao Xu, Shun Zheng, Liang He, Bin Shao, Jian Yin, and Tie-Yan Liu. 2020. SEEK: Segmented embedding of knowledge graphs. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 3888–3897, Online. Association for Computational Linguistics.

Bishan Yang and Tom Mitchell. 2017. Leveraging knowledge bases in LSTMs for improving machine reading. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1436–1446, Vancouver, Canada. Association for Computational Linguistics.

Bishan Yang, Scott Wen-tau Yih, Xiaodong He, Jianfeng Gao, and Li Deng. 2015. Embedding entities and relations for learning and inference in knowledge bases. In *Proceedings of the International Conference on Learning Representations (ICLR) 2015*.

Shuai Zhang, Yi Tay, Lina Yao, and Qi Liu. 2019. Quaternion knowledge graph embeddings. In *Advances in Neural Information Processing Systems 32*, pages 2735–2745.

Zhanqiu Zhang, Jianyu Cai, and Jie Wang. 2020a. Duality-induced regularizer for tensor factorization based knowledge graph completion. *Advances in Neural Information Processing Systems*, 33.

Zhanqiu Zhang, Jianyu Cai, Yongdong Zhang, and Jie Wang. 2020b. Learning hierarchy-aware knowledge graph embeddings for link prediction. In *Proceedings of the Thirty-Fourth AAAI Conference on Artificial Intelligence*. AAAI Press.

A Coarse-to-fine Cascaded Evidence-Distillation Neural Network for Explainable Fake News Detection

Zhiwei Yang[1,2,3,5], **Jing Ma**[2,*], **Hechang Chen**[3,5,*], **Hongzhan Lin**[2], **Ziyang Luo**[2], **Yi Chang**[3,4,5,*]
[1] College of Computer Science and Technology, Jilin University, Changchun, China
[2] Department of Computer Science, Hong Kong Baptist University, Hong Kong, China
[3] School of Artificial Intelligence, [4] International Center of Future Science, Jilin University, China
[5] Key Laboratory of Symbolic Computation and Knowledge Engineering of Ministry of Education
yangzw18@mails.jlu.edu.cn, chenhc@jlu.edu.cn,
{majing, cszyluo, cshzlin}@comp.hkbu.edu.hk, yichang@jlu.edu.cn

Abstract

Existing fake news detection methods aim to classify a piece of news as true or false and provide veracity explanations, achieving remarkable performances. However, they often tailor automated solutions on manual fact-checked reports, suffering from limited news coverage and debunking delays. When a piece of news has not yet been fact-checked or debunked, certain amounts of relevant raw reports are usually disseminated on various media outlets, containing the wisdom of crowds to verify the news claim and explain its verdict. In this paper, we propose a novel Coarse-to-fine Cascaded Evidence-Distillation (CofCED) neural network for explainable fake news detection based on such raw reports, alleviating the dependency on fact-checked ones. Specifically, we first utilize a hierarchical encoder for web text representation, and then develop two cascaded selectors to select the most explainable sentences for verdicts on top of the selected top-K reports in a coarse-to-fine manner. Besides, we construct two explainable fake news datasets, which is publicly available. Experimental results demonstrate that our model significantly outperforms state-of-the-art detection baselines and generates high-quality explanations from diverse evaluation perspectives.

1 Introduction

During the COVID-19 pandemic, almost 80% of consumers in the United States received fake news, which has caused confusion and undermined public health efforts[1]. The proliferation of fake news has increased the demand for automatic fake news detection (Guo et al., 2022). To further clarify and explain detection results, explainable fake news detection has gained more importance recently, aiming to classify the truthfulness of a piece of news and generate veracity explanations[2] (Kotonya and Toni, 2020a). However, existing methods have a limitation in detecting fake news timely as they heavily relied on debunked reports of investigated journalism. Thus, it is urgent to develop explainable yet general methods to mitigate this issue.

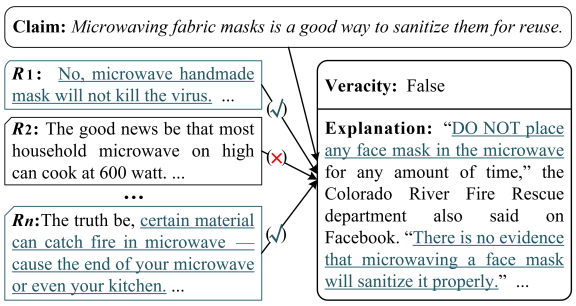

Figure 1: An example for veracity explanation generation. The underlined explanations can be semantically inferred from some relevant sentences in the reports R_1 and R_n. "R" denotes the raw report.

Many previous approaches detected fake news without any justifications (Wang, 2017; Ma et al., 2018). Recently, some explainable methods highlighted salient words or phrases in relevant reports as explanations (Popat et al., 2018; Wu et al., 2021), which lack readable complete sentences. To alleviate these issues, some methods aimed to extract salient sentences from relevant reports via attention mechanisms (Nie et al., 2019; Ma et al., 2019), or pre-trained extractive-abstractive summarization (Kotonya and Toni, 2020b), etc. As the human justification about veracity labels can significantly improve the performance of veracity prediction (Alhindi et al., 2018), Atanasova et al. (2020) proposed the first study on producing veracity explanations jointly with veracity prediction utilizing the debunked report released by fact-checking websites. However, such a debunked report is based on manual endeavors, thus prone to be coverage-limited and relatively inefficient.

A new study by MIT researchers suggests that crowds of laypeople reliably rate claims as effectively as fact-checkers do (Allen et al., 2021). To

*Corresponding authors.
[1] https://www.statista.com/topics/3251/fake-news
[2] Explanations and evidence are used interchangeably

Proceedings of the 29th International Conference on Computational Linguistics, pages 2608–2621
October 12–17, 2022.

use the wisdom of crowds, we assume that crowds of relevant raw reports (e.g., media reports, user comments, blogs, etc.) published by different media outlets contain evidence for effectively detecting fake news and explaining verdicts (Ma et al., 2019; Popat et al., 2018). As shown in Figure 1, given a false claim "*Microwaving fabric masks is a good way to sanitize them for reuse*", the check-worthy reports R_1 and R_n are selected from all reports $[R_1, R_2, \cdots, R_n]$ and then some evidential sentences (underlined) can be used to generate veracity explanations. In contrast, existing methods usually tailor models on one manual fact-checked article, rarely attempting to detect fake news based on raw reports.

To this end, we propose a general coarse-to-fine cascaded evidence-distillation (CofCED) network to detect fake news and explain verdicts directly using raw reports, mitigating the dependency on fact-checked reports. Specifically, we design a hierarchical encoder for text representation, and then we develop two coarse-to-fine cascaded selectors to distill explainable sentences on top of the selected top-K check-worthy reports. Our predictions of explainable sentences can be obtained by explicitly considering four features, i.e., claim relevance, richness, salience, and non-redundancy. Different from FEVER (Thorne et al., 2018) using human-crafted claims with credible Wikipedia articles, the claims in our task are real-world news containing some unreliable reports. Thus, detecting fake news on raw reports is much more challenging and significant than that in FEVER task.

Our contributions are as follows: **1)** To the best of our knowledge, we present the first study on explainable fake news detection directly utilizing the wisdom of crowds, alleviating the dependency on fact-checked reports; **2)** Our model has the advantage of revealing insight into the generation of veracity explanations from various perspectives; **3)** We construct two realistic datasets, i.e., RAWFC and LIAR-RAW, consisting of raw reports for each claim. Experimental results on benchmarks demonstrate the effectiveness of CofCED for detecting fake news and and explaining verdicts based on raw reports. Our resources are publicly available at https://github.com/Nicozwy/CofCED.

2 Related Work

We review prior works closely related to ours based on several surveys (Shu et al., 2017; Kotonya and Toni, 2020a).

Black-boxed fake news detection. Many existing studies on fake news detection achieved promising performances by incorporating claim metadata to facilitate the detection, such as user profiles (Wang, 2017; Long, 2017; Karimi et al., 2018). Besides, various deep learning methods have been proposed to capture report features, e.g., credibility (Popat et al., 2017), stances (Ma et al., 2018), writing styles (Potthast et al., 2018), extra knowledge (Dun et al., 2021), etc. Although these methods could improve the detection performance, they are lack of explainability on verdicts.

Explainable fake news detection. To address the above issue, many explainable methods on this task explored attention mechanisms to highlight salient words (Popat et al., 2018; Wu et al., 2021), news attributes (Yang et al., 2019), and suspicious users (Lu and Li, 2020), to obtain relevant evidence, providing a certain explainability. To improve the readability in word-level methods, there are some methods obtained evidential sentences using attention weights (Shu et al., 2019), semantic matching (Nie et al., 2019), and entailment (Ma et al., 2019). More recently, Atanasova et al. (2020) proposed the first study on directly producing veracity explanations using extractive summarization, and Kotonya and Toni (2020b) made use of extractive-abstractive summarization for explanation generation, independent of the veracity prediction. However, they significantly relied on the manual fact-checked report and rarely attempted to consider fine-grained features for this task. Thus, we utilize the wisdom of crowds for fake news detection based on raw reports, providing a highly explainable structure for explanation generation.

Datasets. For explainable fake news detection, FEVER (Thorne et al., 2018) was crafted merely from credible Wikipedia articles, and MultiFC (Augenstein et al., 2019) provided a real-world benchmark for multi-domain claims. While offering evidence labels, they do not contain veracity explanations. By contrast, LIAR-PLUS (Alhindi et al., 2018) extended on LIAR (Wang, 2017) and PUBHEALTH (Kotonya and Toni, 2020b) on the public health, providing manual explanations for explainable fake news detection. However, they only contain the manual fact-checked report that is relatively inefficient and coverage-limited. Thus, we constructed two datasets by collecting raw reports, which is more suitable and challenging for this task.

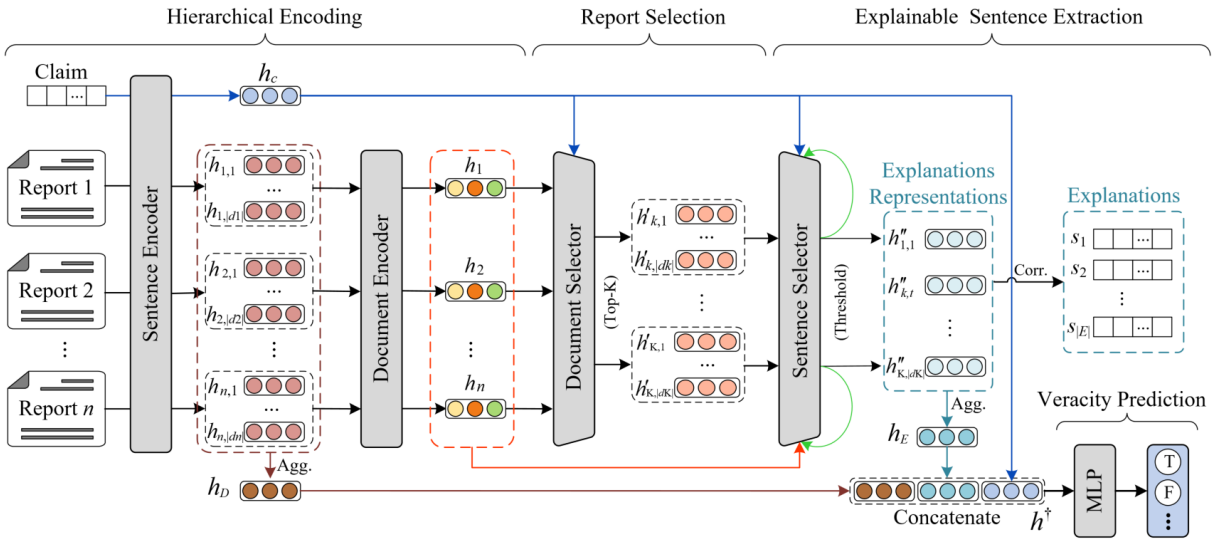

Figure 2: An overview of our proposed CofCED framework. The document selector and the sentence selector are used for selecting check-worthy reports (containing oracles) and oracles, respectively. "Agg." denotes aggregation and " Corr." denotes corresponding. We use different color to highlight different objects. Note that the green line denotes the last output of sentence selection for checking redundancy.

3 Problem Statement

Given a fake news dataset $\{C\}$, $C = (c, \mathcal{D})$ is a tuple representing a given claim c and its relevant raw reports $\mathcal{D} = \{d_i\}_{i=1}^{|\mathcal{D}|}$, where each $d_i = (s_{i,1}, s_{i,2}, \cdots, s_{i,|d_i|})$ denotes a relevant report consisted of a sequence of sentences and $|.|$ denotes the number of items. In the task of explainable fake news detection, each claim c is associated with a veracity y taking one of the class labels from $\{\text{True}, \text{False}, \cdots\}$, and each raw report d_i is associated with a binary label $y_i^d \in Y^d$ indicating that whether d_i contains explainable sentences (i.e., oracles). For each sentence $s_{i,j}$, $y_{i,j}^s \in Y^s$ is a binary label indicating that whether $s_{i,j}$ is one of the explainable sentences w.r.t. the gold justification.

We formulate this task as a multi-task learning problem by considering check-worthy report selection, explainable sentence extraction, and veracity prediction. Formally, $f : f(c, \mathcal{D}) \rightarrow (\hat{y}, \hat{Y}^d, \hat{Y}^s, \hat{E})$, where $\hat{E}$ denotes the veracity explanation (i.e., *evidence*) consisting of a set of predicted sentences (i.e., $\hat{y}_i^d = 1$ and $\hat{y}_{i,j}^s = 1$).

4 CofCED: The Proposed Method

Fig. 2 gives an overview of our proposed CofCED, which consists of four parts: hierarchical encoding, report selection, explainable sentence extraction, and veracity prediction.

4.1 Hierarchical Encoding

Given a word sequence of a claim or report sentence $T = (w_1 \cdots w_t \cdots w_{|T|})$, where $w_t \in \mathbb{R}^d$ is a d-dimensional vector initialized with a text encoder. Because words form a sentence and sentences form a report, we utilize a hierarchical encoding method for sentence and report representation in our model. Specifically, for sentence encoding, we use the special token "[CLS]" embedding from the final contextual layer of the pre-trained language model (Sanh et al., 2019) as the sentence representation. Thus, we obtain the sentence representation for a claim c and each sentence $s_{i,j}$ in a raw report d_i as $\mathbf{h}_c \in \mathbb{R}^d$ and $\mathbf{h}_{i,j} \in \mathbb{R}^d$, respectively.

For document encoding, we further adopt a document encoder consisting of a bidirectional LSTM (BiLSTM) (Rashkin et al., 2017) and a max-pooling layer to aggregate all salient sentence features as the representation of a report:

$$\tilde{\mathbf{h}}_{i,j} = \text{BiLSTM}(\mathbf{h}_{i,j}, \overrightarrow{\mathbf{h}}_{i,j-1}, \overleftarrow{\mathbf{h}}_{i,j-1}, \theta) \quad (1)$$

$$\mathbf{h}_i = \text{Max}([\tilde{\mathbf{h}}_{i,1}; \tilde{\mathbf{h}}_{i,2}; \cdots; \tilde{\mathbf{h}}_{i,|d_i|}]) \quad (2)$$

where $\tilde{\mathbf{h}}_{i,j} \in \mathbb{R}^d$ denotes the cross-sentence hidden state, and $\mathbf{h}_i \in \mathbb{R}^d$ denotes the representation of the report d_i. Max denotes the max pooling, [;] denotes concatenation, and θ denotes encoder parameters.

4.2 Report Selection

Since this task is formulated on massive raw reports, our model aims to automatically narrow

down the evidence extraction by ranking them and capturing the top ones for further analysis. Taking the claim in Fig. 1 as an example, there are n retrieved reports about "microwaving fabric masks" and the significant reports R_1 and R_n containing oracles (i.e., underlined sentences) are selected for veracity prediction and explanation generation.

To distill the check-worthy reports from massive reports $\mathcal{D}$ that are helpful for veracity prediction, we firstly develop a coarse-grained document selector by treating the claim as a query to find K most significant results. Then, global attention is utilized to obtain the significance score for each report d_i:

$$\alpha_{c\to\mathcal{D}} = \text{softmax}(\mathbf{H}_{\mathcal{D}} W_\alpha \mathbf{h}_c) \tag{3}$$

where $\mathbf{H}_{\mathcal{D}} = [\mathbf{h}_1; \mathbf{h}_2; \cdots; \mathbf{h}_{|\mathcal{D}|}]$ compacts all hidden vectors of reports and $W_\alpha \in \mathbb{R}^{d\times d}$ is a trainable parameter. We use $\alpha_{c\to\mathcal{D}}$ to rank all reports and select the top-K results as the check-worthy reports (i.e., $\hat{y}_i^d = \alpha_i(\alpha_i \geq \alpha_K)$ and otherwise $\hat{y}_i^d = 0(\alpha_i < \alpha_K)$). Note that the t-th sentence representation in the k-th selected report $d_k^{'}$ are denoted as $\mathbf{h}_{k,t}^{'} \in \{\mathbf{h}_{k,1}^{'}, \mathbf{h}_{i,2}^{'}, ..., \mathbf{h}_{k,|d_k^{'}|}^{'}\}$, and its document representation is denoted as $\mathbf{h}_k^{'}$, which are used for explainable sentence extraction.

4.3 Explainable Sentence Extraction

On top of selected reports, we treat explanation generation as a multi-document extractive summarization, where each report is visited sequentially for explainable sentences. Such reports are regarded as the wisdom of crowds when detecting a dubious claim. We assume that explainable sentences for verdicts should be claim-relevant, informative, salient, and non-redundant. Specifically, there may exist redundancy between reports because a report is generally self-contained and multiple raw reports are more likely to contain semantically irrelevant and redundant sentences (Ma et al., 2019).

In this paper, we develop a fine-grained sentence selector to extract explainable sentences from these check-worthy reports considering the following four features: 1) *claim relevance* measures the topic coverage of each sentence regarding the claim; 2) *richness* measures the content informativeness of each sentence containing evidence; 3) *salience* measures the significance of each sentence regarding the entire report; 4) *non-redundancy* measures the novelty of each sentence regarding previous selected explainable sentences. Therefore, we define a layer to predict the probability of each sentence that should be selected via integrating the four features as follows:

$$\begin{aligned} \text{P}(y_{k,t}^s = 1 | \mathbf{h}_c, \mathbf{h}_{k,t}^{'}, \mathbf{h}_k^{'}, \mathbf{h}_d) \\ = \sigma(\underbrace{\mathbf{h}_{k,t}^{'} W_c \mathbf{h}_c}_{(\textit{claim relevance})} + \underbrace{\mathbf{h}_{k,t}^{'} W_s}_{(\textit{richness})} \\ + \underbrace{\mathbf{h}_{k,t}^{'} W_r \mathbf{h}_k^{'}}_{(\textit{salience})} - \underbrace{\mathbf{h}_{k,t}^{'} W_d \mathbf{h}_d}_{(\textit{non-redundancy})}) \end{aligned} \tag{4}$$

where $y_{k,t}^s$ is a binary variable indicating whether the t-th sentence in the selected report $d_k^{'}$ should be selected as part of explanations $\hat{E}$, and W_* are trainable parameters. $\mathbf{h}_d$ is the redundancy vectors initialized with all zeros and updated by selected sentences in previously visited reports as follows:

$$\mathbf{h}_d = \tanh(\sum_t \mathbf{h}_{k-1,t}^{'} \cdot \text{P}(y_{k,t}^s = 1)) \tag{5}$$

Considering the number of report sentences, our model learns to select the explainable sentences with probabilities above a soft threshold $\varepsilon_k = 1/|d_k^{'}|$, i.e., $\text{P}(y_{k,t}^s = 1) > \varepsilon_k$, where $\text{P}(y_{k,t}^s = 1)$ is obtained by Eq. (4). Note that $\mathbf{h}_{k,t}^{''}$ is used to denote the sentence representation output from the explainable sentence selector.

4.4 Veracity Prediction

To enhance final veracity prediction, we further employ the extracted explanation as additional *evidence* besides the claims and all reports. Specifically, we aggregate the recognitions from such evidence and reports for a target claim, respectively, and then obtain the final representation by concatenating the claim representation, report representation, and explanation representation as follows:

$$\mathbf{h}_D = \text{Max}([\mathbf{h}_1; \mathbf{h}_2; \cdots; \mathbf{h}_{|\mathcal{D}|}]) \tag{6}$$

$$\mathbf{h}_E = \text{Max}([\mathbf{h}_1^{''}; \mathbf{h}_2^{''}; ...; \mathbf{h}_K^{''}]) \tag{7}$$

$$\mathbf{h}^\dagger = [\mathbf{h}_c; \mathbf{h}_D; \mathbf{h}_E] \tag{8}$$

where $\mathbf{h}_D$ denotes the integrated representation of all report sentences, $\mathbf{h}_E$ denotes the integrated representation of all explainable sentences. $\mathbf{h}^\dagger$ denotes the final representation for veracity prediction. K denotes a hyperparameter controlling the maximum number of selected reports. Similar to Eq. (2), $\mathbf{h}_k^{''} = \text{Max}([\mathbf{h}_{k,1}^{''}; \mathbf{h}_{k,2}^{''}; \cdots; \mathbf{h}_{k,|d_k^{'}|}^{''}])$ is the k-th report representation in the extracted explanations.

Finally, $\mathbf{h}^\dagger$ is fed into a multi-layer perceptron (MLP) layer to predict the veracity label as follows:

$$\hat{y} = \text{softmax}(\text{MLP}(\mathbf{h}^\dagger)) \tag{9}$$

4.5 Model Training

It is inefficient to train report selection, explainable sentence extraction, and veracity prediction independently, considering their implicit correlations and the pipeline for explainable fake news detection in the real world (Kotonya and Toni, 2020a). Thus, we jointly optimize these three sub-tasks in an end-to-end model. For model training, we minimize the overall loss $\mathcal{L}_{all}$ as follows:

$$\mathcal{L}_D = -\sum_i y_i^d \log(\hat{y}_i^d) \quad (10)$$

$$\mathcal{L}_S = -\sum_k \sum_t y_{k,t}^s \log(\hat{y}_{k,t}^s) \quad (11)$$

$$\mathcal{L}_C = -y\log(\hat{y}) \quad (12)$$

$$\mathcal{L}_{all} = \beta_D \mathcal{L}_D + \beta_S \mathcal{L}_S + \beta_C \mathcal{L}_C \quad (13)$$

where $\mathcal{L}_D$, $\mathcal{L}_S$, and $\mathcal{L}_C$ denote the cross-entropy loss for check-worthy report selection, explanation generation and veracity prediction tasks, respectively. y_i^d and $\hat{y}_i^d$ denote the gold and predicted label of reports, respectively. $y_{k,t}^s$, and $\hat{y}_{k,t}^s$ denote the ground truth and the predicted probability of the sentence for explanation, respectively. y and $\hat{y}$ denote the ground truth and predicted veracity probability of the claim, respectively. β denotes the trade-off parameter, controlling the task importance in our work. We can automatically assign β_D, β_S, and β_C with proper values using the adaptive strategy, rather than the grid search (see Appendix B).

5 Experiments

5.1 Datasets and Settings

To the best of our knowledge, there is no public dataset on raw reports available for this task. Thus, we collect two explainable datasets, i.e., RAWFC and LIAR-RAW, referring to two different fact-checking sites (i.e., Snopes[3] and Politifact[4]) for gold labels, respectively. For RAWFC, we constructed it from scratch by collecting the claims from Snopes and *relevant raw reports* by retrieving claim keywords. For LIAR-RAW, we extended the public dataset LIAR-PLUS (Alhindi et al., 2018) with *relevant raw reports*, containing *fine-grained* claims from Politifact. We process and separate these datasets into train/valid/test sets by 8:1:1 following the same setting in (Atanasova et al., 2020). More details are illustrated in Appendix A.

[3]www.snopes.com

[4]www.politifact.com

Dataset	RAWFC	LIAR-RAW
Claim	2,012	12,590
# pants-fire	-	1,013
# false	646	2,466
# barely-true	-	2,057
# half-true †	671	2,594
# mostly-true	-	2,439
# true	695	2,021
Veracity Label	3	6
Explain sentence		
# min	1	1
# max	110	209
# avg	18.4	4.1
Report per claim		
# min	1	1
# max	30	30
# avg	21.0	12.3
Sentence per report		
# min	1	1
# max	155	59
# avg	7.4	5.5

Table 1: Statistics of datasets. # half-true † is also denoted as # half in RAWFC. The number of oracles in datasets isn't pre-defined.

For experimental setup, we initialized word embeddings with the base uncased DistilBERT (Sanh et al., 2019) and $d = 768$ dimensions. The hidden size of LSTM is set to 384. We use Adam optimizer (Kingma and Ba, 2014) with a learning rate of 1e-5 and the mini-batch size is set to 1 to minimize joint cross-entropy loss. The maximum number K of selected reports for each claim is empirically set to 12 and 18 for RAWFC and LIAR-RAW, respectively. We use a soft threshold $\varepsilon_i = 1/|d_i'|$ for selection while empirically setting the maximum number of oracle sentences to 30 and 55 for RAWFC and LIAR-RAW, respectively. We set the dropout rate to 0.4 before final prediction and the maximum number of training epochs to 8. For evaluation, we employ macro-averaged precision (P), recall (R), and F1 score (macF1) for veracity prediction, and use ROUGE-N F1 score ($N \in \{1, 2, L\}$) and the human evaluation to evaluate the quality of explanations. *Note that fact-checked reports are not required during inference in our model.*

5.2 Veracity Prediction Performance

Table 2 compares veracity prediction results with the following strong baselines: 1) **SVM** (Pedregosa et al., 2011): This uses bag-of-words features to train SVM-based model for fake news detection; 2) **CNN** (Wang, 2017): This incorporates available metadata features to enhance representation learning; 3) **RNN** (Rashkin et al., 2017): This learns representation from word sequences without external resources; 4) **DeClarE** (Popat et al., 2018): This combines word embeddings from the claim,

Model	RAWFC			LIAR-RAW		
	P(%)	R(%)	macF1(%)	P(%)	R(%)	macF1(%)
SVM (Pedregosa et al., 2011)	32.33	32.51	31.71	15.78	15.92	15.34
CNN (Wang, 2017)	38.80	38.50	38.59	22.58	22.39	21.36
RNN (Rashkin et al., 2017)	41.35	42.09	40.39	24.36	21.20	20.79
DeClarE (Popat et al., 2018)	43.39	43.52	42.18	22.86	20.55	18.43
dEFEND (Shu et al., 2019)	44.93	43.26	44.07	23.09	18.56	17.51
SentHAN (Ma et al., 2019)	45.66	45.54	44.25	22.64	19.96	18.46
SBERT-FC (Kotonya and Toni, 2020b)	51.06	45.92	45.51	24.09	22.07	22.19
GenFE (Atanasova et al., 2020)	44.29	44.74	44.43	28.01	26.16	26.49
GenFE-MT (Atanasova et al., 2020)	45.64	45.27	45.08	18.55	19.90	15.15
CofCED	**52.99**	**50.99**	**51.07**	**29.48**	**29.55**	**28.93**

Table 2: Experimental results of veracity prediction merely using raw reports ($p < 0.05$ under t-test).

report, and source to access the credibility of the claim; 5) **dEFEND** (Shu et al., 2019): This utilizes GRU-based model for veracity prediction with explanations; 6) **SentHAN** (Ma et al., 2019): This represents each sentence based on sentence-level coherence and semantic conflicts with the claim; 7) **SBERT-FC** (Kotonya and Toni, 2020b): This uses SentenceBERT (SBERT) for encoding and detects fake news based on the top-K ranked sentences; 8) **GenFE/GenFE-MT** (Atanasova et al., 2020): This detects fake news independently or jointly with explanations in the multi-task set-up.

Table 2 demonstrates the detection performance of our proposed CofCED compared with existing strong baselines in terms of precision, recall and macro F1 (macF1). From this table, we can observe that CNN and RNN outperform SVM on both datasets, indicating that deep learning methods can better capture semantic and syntactic features from raw reports. By attentively aggregating multiple features from the claim, reports, and source to estimate the veracity, dEFEND, DeClarE and SentHAN achieve better performance on RAWFC but slightly worse results on LIAR-RAW, because fine-grained labels contained in LIAR-RAW make it more challenging.

SBERT-FC and GenFE outperform SentHAN and dEFEND on both datasets, demonstrating the superiority of pre-trained models. GenFE-MT performs better than GenFE on RAWFC, but much worse than other baselines on LIAR-RAW, implying the challenge of fine-grained fake news detection with explanation generation in the multi-task setting. Generally, CofCED consistently achieves much better performance on RAWFC and LIAR-RAW, demonstrating the superiority of CofCED in combining report selection, explainable sentence extraction and veracity prediction for fake news detection directly on raw reports, alleviating the dependency on fact-checked reports.

5.3 Ablation Study

To evaluate the impact of each component, we conduct ablation experiments for CofCED by removing the following key components: 1) **RS** denotes report selection; 2) **SE** denotes sentence selection; 3) **RS&SE** denotes RS and SE; 4) Four semantic features: **claim relevance**, **richness**, **salience**, and **non-redundancy**, for sentence selection.

As shown in Table 3, CofCED significantly outperforms CofCED w/o $*$ ($*$ indicates a component) on both datasets, demonstrating all components contribute to the effectiveness of CofCED in detecting fake news. Specifically, CofCED's performance significantly decreases without RS&SE because there is noise in raw reports, affecting the veracity prediction. CofCED w/o SE performs much worse than the others because irrelevant or redundant information contained in such reports may weaken the effect of evidence for detection; CofCED w/o RS also achieves worse performance than CofCED because noisy reports may affect sentence selection and model training. Furthermore, the performance of CofCED w/o claim relevance significantly decreases, highlighting the importance of selecting claim-relevant evidence for final prediction. CofCED outperforms CofCED without these four features for sentence selection, respectively, demonstrating they contribute to extracting explainable sentences for fake news detection from different perspectives.

5.4 Explanation Evaluation

Table 4 reports the ROUGE results of the extracted explanations regarding word overlapping. The ROUGE F1 score is employed to evaluate their qualities comparing with the following strong baselines: 1) **LEAD-N** (Nallapati et al., 2017): This uses the first N sentences as explanation and $N = 5$; 2) **Oracle** (Atanasova et al., 2020): This typically presents the best greedy approximation of the gold

Model	RAWFC			LIAR-RAW		
	P(%)	R(%)	macF1(%)	P(%)	R(%)	macF1(%)
CofCED w/o RS&SE	45.01	45.02	44.98	25.69	24.55	24.80
CofCED w/o SE	52.27	46.36	43.80	27.59	23.81	23.74
CofCED w/o RS	49.26	46.92	46.37	27.08	25.32	25.52
CofCED w/o non-redundancy	48.80	46.98	47.48	26.54	27.36	26.65
CofCED w/o salience	43.96	49.24	46.44	26.36	24.88	25.23
CofCED w/o richness	48.08	47.50	47.12	27.06	25.82	26.05
CofCED w/o claim relevance	45.66	45.25	45.28	26.42	24.01	24.88
CofCED	**52.99**	**50.99**	**51.07**	**29.48**	**29.55**	**28.93**

Table 3: Ablation study results of our veracity prediction on test sets; w/o denotes 'without'.

Model	RAWFC			LIAR-RAW		
	ROU-1	ROU-2	ROU-L	ROU-1	ROU-2	ROU-L
LEAD-N	19.52	4.54	17.26	9.84	0.40	7.20
Oracle	37.62	13.22	34.67	25.50	9.28	22.61
EXTABS (Kotonya and Toni, 2020b)	-	-	-	18.85	3.61	12.90
dEFEND (Shu et al., 2019)	19.95	5.08	17.21	17.03	3.26	11.42
GenFE-MT (Atanasova et al., 2020)	18.23	7.12	17.32	**23.08**	3.67	12.10
CofCED w/o non-redundancy	27.32	9.06	23.19	17.96	3.54	12.43
CofCED w/o salience	26.67	7.44	21.02	17.27	3.41	11.69
CofCED w/o richness	25.75	8.66	21.87	17.23	3.44	12.10
CofCED w/o claim relevance	25.56	8.07	20.73	17.08	3.31	11.25
CofCED w/o RS	26.64	8.96	22.69	17.51	**3.72**	**13.20**
CofCED	**27.62**	**9.32**	**23.57**	17.14	3.49	12.96

Table 4: ROUGE results of the generated explanation. ROU-N ($N \in \{1, 2, L\}$) denotes the ROUGE-N F1 score that evaluates the token overlap between the explanation and human justifications. RAWFC is not suitable for EXTABS because its gold justification is too long to train an abstractive-summarization model.

explanation with sentences extracted from reports; 3) **EXTABS** (Kotonya and Toni, 2020b): This uses extractive-abstractive summarization model pre-trained on extra news articles and summaries dataset before fine-tuning (Liu and Lapata, 2019); 4) **dEFEND**: This uses internal attention weights for explanations; 5) **GenFE-MT**: This incorporates explanation generation using pre-trained models.

Overall, CofCED achieves the state-of-the-art performance on RAWFC and comparable ROUGE scores with GenFE-MT on LIAR-RAW, suggesting that our CofCED can effectively distill explainable sentences that contributes to the final veracity prediction, as shown in Table 2. Specifically, the ROUGE results of LEAD-N and Oracle on RAWFC and LIAR-RAW indicate that generating explanations for fine-grained fake news detection is a more complex challenge. EXTABS obtains competitive results on LIAR-RAW due to additional news and summaries datasets for abstractive summarization but it cannot deal with long justifications. GenFE-MT performs much better than dEFEND on both datasets, indicating the advantage of pre-trained models in generating explanation from raw reports but failing to trade off both tasks regarding Table 2. For ablation results, we observe that some ablations of CofCED achieve slightly better ROUGE scores but much worse veracity predictions on LIAR-RAW, indicating these four features can effectively select explainable sentences to enhance fake news detection. Besides, CofCED performs better on RAWFC while only comparable on LIAR-RAW than CofCED w/o RS, implying that generating explanations for fine-grained veracity labels is much more challenging regarding word overlapping. We further conduct human evaluations as shown in Appendix D. In summary, our CofCED can effectively generate accurate explanations from raw reports and all components contribute to focusing on veracity prediction.

5.5 Case Study

For in-depth analysis, we further explore the process of CofCED in selecting explainable sentences. We normalized scores for each abstract feature, obtaining its overall probability for explaining detection results. As shown in Table 5, given a false claim about COVID-19, the top two sentences with higher overall scores refute the claim from different perspectives and the last two sentences with 0.3 and 0.2 overall scores contribute less to the veracity prediction. The separated terms, i.e., claim relevance, richness, salience, and non-redundancy, in Eq. (4) are clearly visualized for seeking the major factor responsible for the classification of each sentence. In addition to being a state-of-the-art method

Claim: *Dr. Tasuku Honjo said that COVID-19 was "man-made" at a lab in Wuhan, China.* [**Prediction:** False] **Explanation:** Honjo did not work at the Wuhan Institute of Virology, he did not say that COVID-19 was "invented" or "man-made," and the Twitter account posting similar claims does not belong to the Nobel Prize winner. In addition, this rumor is all based on the unfounded notion that COVID-19 was created as a bioweapon. (...)	Relevance	Richness	Salience	Non-redant	**Overall**	
[1] TOKYO, May 6 (Xinhua) – Japanese Nobel laureate Tasuku Honjo have refuted claim that China manufacture the novel coronavirus, say those rumor be "dangerously distract."	0.9	0.6	0.8	0.9	**0.9**	√
[2] Actually, the professor don't have a Twitter account.	0.7	0.5	0.6	0.9	**0.6**	√
[3] The 2018 Nobel laureate encourage Japanese authority to adopt a more proactive approach.	0.3	0.5	0.4	0.8	**0.3**	×
[4] China will have a big role to play. ...	0.2	0.2	0.1	0.7	**0.2**	×

Table 5: Our visualization of explanation extraction from raw reports. Each row is a sentence in raw reports. The score in the columns are normalized from each of the abstract features in Eq. (4), and the last column is the final probability explaining to detection results.

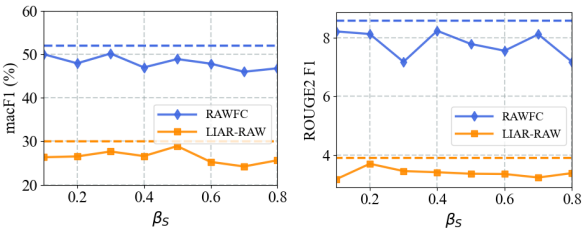

(a) Veracity prediction (b) Explanation generation

Figure 3: Results of CofCED under different values of the trade-off parameter β_S and $\beta_C = 1 - \beta_S$. The colored dashed horizontal lines denote the performance of CofCED with our adaptive weighting.

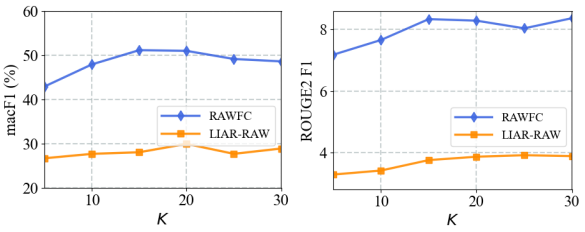

(a) Veracity prediction (b) Explanation generation

Figure 4: Results of CofCED under different values of the maximum number K for report selection.

for explainable fake news detection, CofCED has the additional superiority of being very explainable for sentence extraction. Thus, such visualization increases the transparency of the system and the credibility of generated explanations for verdicts.

5.6 Parameter Sensitivity Study

We further investigate the impact of the trade-off parameter β in Eq. (13) on CofCED using the grid search. For brevity, Fig. 3 only presents the results for a) veracity prediction and b) explanation generation on development sets when β_S varies and $\beta_D = 0.5$ is temporarily fixed. We also tried various $\beta_D \in [0.1, 0.8]$ and consistently achieved similar results. By varying the value of β_S from 0.1 to 0.8, our model achieves better performances on one task but poorer results on the other. This is because these tasks show different importance and priority for the final performance over time. By contrast, our CofCED with our proposed multi-task adaptive weighting (MAW) (i.e., the colored dashed horizontal lines) consistently achieves better performance. Thus, these results demonstrate that CofCED with MAW can effectively find better weights for explanation generation and veracity prediction in multi-task learning, alleviating the labor for the grid search for trade-off parameters.

To examine the impact of the maximum number of selected reports on CofCED, we conduct experiments by varying K while fixing other hyper-parameters on the development sets of RAWFC and LIAR-RAW. As shown in Fig. 4, we can see that too few raw reports generally cause performance reduction because the noise in the raw reports may impose the model training bias. Since too many raw reports will cause the out of memory problem, we empirically choose a proper value in this study, i.e., K is set to 12 and 18 for RAWFC and LIAR-RAW, respectively. Note that ε is a soft threshold that can be automatically assigned regarding the total number of report sentences.

6 Conclusion

We present a coarse-to-fine cascaded evidence-distillation (CofCED) neural network for explainable fake news detection that achieves the best detection performance and distills accurate veracity explanations directly from raw reports. Besides, CofCED has the additional advantage of being explainable in producing veracity explanations, explicitly considering the semantic features, e.g., claim relevance, richness, salience, and non-redundancy. Experimental results on real-world datasets demonstrate the effectiveness of CofCED for explainable fake news detection utilizing the wisdom of crowds, effectively mitigating the dependency on fact-checked reports.

Acknowledgments

This work is partially supported by National Natural Science Foundation of China through grants No.61976102, No.U19A2065, and No.61902145. This work is partly supported by the International Cooperation Project (20220402009GH) and Science & Technology Development Program (20210508060RQ), Jilin Province. This work is partly supported by HKBU One-off Tier 2 Start-up Grant (RCOFSGT2/20-21/SCI/004), Hong Kong RGC ECS (22200722).

References

Tariq Alhindi, Savvas Petridis, and Smaranda Muresan. 2018. Where is your evidence: improving fact-checking by justification modeling. In *FEVER*, pages 85–90.

Jennifer Allen, Antonio A Arechar, Gordon Pennycook, and David G Rand. 2021. Scaling up fact-checking using the wisdom of crowds. *Science advances*, 7(36):eabf4393.

Pepa Atanasova, Jakob Grue Simonsen, Christina Lioma, and Isabelle Augenstein. 2020. Generating fact checking explanations. In *ACL*, pages 7352–7364.

Isabelle Augenstein, Christina Lioma, Dongsheng Wang, Lucas Chaves Lima, Casper Hansen, Christian Hansen, and Jakob Grue Simonsen. 2019. Multifc: A real-world multi-domain dataset for evidence-based fact checking of claims. In *EMNLP-IJCNLP*, pages 4685–4697.

Rich Caruana. 1997. Multitask learning. *Machine learning*, 28(1):41–75.

Zhao Chen, Vijay Badrinarayanan, Chen-Yu Lee, and et al. 2018. Gradnorm: Gradient normalization for adaptive loss balancing in deep multitask networks. In *ICML*, pages 794–803. PMLR.

Yaqian Dun, Kefei Tu, Chen Chen, Chunyan Hou, and Xiaojie Yuan. 2021. Kan: Knowledge-aware attention network for fake news detection. In *Proc. AAAI Conf. Artif. Intell.*, volume 35, pages 81–89.

Zhijiang Guo, Michael Schlichtkrull, and Andreas Vlachos. 2022. A survey on automated fact-checking. *Transactions of the Association for Computational Linguistics*, 10:178–206.

Hamid Karimi, Proteek Roy, Sari Saba-Sadiya, and Jiliang Tang. 2018. Multi-source multi-class fake news detection. In *ICCL*, pages 1546–1557.

Diederik P Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. *arXiv:1412.6980*.

Neema Kotonya and Francesca Toni. 2020a. Explainable automated fact-checking: A survey. In *ICCL*, pages 5430–5443.

Neema Kotonya and Francesca Toni. 2020b. Explainable automated fact-checking for public health claims. In *EMNLP*, pages 7740–7754.

Chin-Yew Lin. 2004. Rouge: A package for automatic evaluation of summaries. In *Text summarization branches out*, pages 74–81.

Shikun Liu, Edward Johns, and Andrew J Davison. 2019. End-to-end multi-task learning with attention. In *CVPR*, pages 1871–1880.

Yang Liu and Mirella Lapata. 2019. Text summarization with pretrained encoders. In *EMNLP-IJCNLP*, pages 3730–3740.

Yunfei Long. 2017. Fake news detection through multi-perspective speaker profiles. In *ACL*.

Yi-Ju Lu and Cheng-Te Li. 2020. Gcan: Graph-aware co-attention networks for explainable fake news detection on social media. In *ACL*, pages 505–514.

Jing Ma, Wei Gao, Shafiq Joty, and et al. 2019. Sentence-level evidence embedding for claim verification with hierarchical attention networks. In *ACL*.

Jing Ma, Wei Gao, and Kam-Fai Wong. 2018. Detect rumor and stance jointly by neural multi-task learning. In *WWW*, pages 585–593.

Ramesh Nallapati, Feifei Zhai, and Bowen Zhou. 2017. Summarunner: A recurrent neural network based sequence model for extractive summarization of documents. In *AAAI*.

Yixin Nie, Haonan Chen, and Mohit Bansal. 2019. Combining fact extraction and verification with neural semantic matching networks. In *AAAI*, pages 6859–6866.

Fabian Pedregosa, Gaël Varoquaux, Alexandre Gramfort, Vincent Michel, Bertrand Thirion, Olivier Grisel, Mathieu Blondel, Peter Prettenhofer, Ron Weiss, Vincent Dubourg, et al. 2011. Scikit-learn: Machine learning in python. *JMLR*, 12:2825–2830.

Kashyap Popat, Subhabrata Mukherjee, Jannik Strötgen, and Gerhard Weikum. 2017. Where the truth lies: Explaining the credibility of emerging claims on the web and social media. In *WWW*, pages 1003–1012.

Kashyap Popat, Subhabrata Mukherjee, Andrew Yates, and Gerhard Weikum. 2018. Declare: Debunking fake news and false claims using evidence-aware deep learning. In *EMNLP*, pages 22–32.

Martin Potthast, Johannes Kiesel, Kevin Reinartz, Janek Bevendorff, and Benno Stein. 2018. A stylometric inquiry into hyperpartisan and fake news. In *ACL*, pages 231–240.

Hannah Rashkin, Eunsol Choi, Jin Yea Jang, Svitlana Volkova, and Yejin Choi. 2017. Truth of varying shades: Analyzing language in fake news and political fact-checking. In *EMNLP*, pages 2931–2937.

Nils Reimers and Iryna Gurevych. 2019. Sentence-bert: Sentence embeddings using siamese bert-networks. In *EMNLP-IJCNLP*, pages 3982–3992.

Victor Sanh, Lysandre Debut, Julien Chaumond, and Thomas Wolf. 2019. Distilbert, a distilled version of bert: smaller, faster, cheaper and lighter. *arXiv:1910.01108*.

Kai Shu, Limeng Cui, Suhang Wang, Dongwon Lee, and Huan Liu. 2019. defend: Explainable fake news detection. In *SIGKDD*, pages 395–405.

Kai Shu, Amy Sliva, Suhang Wang, Jiliang Tang, and Huan Liu. 2017. Fake news detection on social media: A data mining perspective. *SIGKDD*, 19(1):22–36.

James Thorne, Andreas Vlachos, Christos Christodoulopoulos, and Arpit Mittal. 2018. Fever: a large-scale dataset for fact extraction and verification. In *NAACL-HLT*, pages 809–819.

William Yang Wang. 2017. "liar, liar pants on fire": A new benchmark dataset for fake news detection. In *ACL (short)*, pages 422–426.

Lianwei Wu, Yuan Rao, Ling Sun, and Wangbo He. 2021. Evidence inference networks for interpretable claim verification. In *AAAI*, pages 14058–14066.

Fan Yang, Shiva K Pentyala, Sina Mohseni, Mengnan Du, Hao Yuan, Rhema Linder, Eric D Ragan, Shuiwang Ji, and Xia Hu. 2019. Xfake: Explainable fake news detector with visualizations. In *WWW*, pages 3600–3604.

Appendices

A Dataset Details

Existing benchmarks for explainable fake news detection collected official debunked reports written by journalists as evidence for fake news detection (Kotonya and Toni, 2020a), which is labor-intensive and relatively inefficient. However, debunked reports are not always available for breaking news and are mixed up with raw reports, which may contain more semantically irrelevant and redundant information. To the best of our knowledge, there is no available explainable dataset based on crowds of raw reports to detect fake news before official reports published. Thus, existing datasets are not suitable for most real-life scenarios, especially when the fact-checked reports are not always available. To address this issue, we collect two new datasets, i.e., RAWFC and LIAR-RAW, considering a more general situation of detecting and explaining fake news with relevant raw reports.

Note that we construct RAWFC and LIAR-RAW with gold labels referring to Snopes[5] and Politifact[6], respectively. RAWFC is constructed from scratch as follows and LIAR-RAW are extended with raw reports based on LIAR-PLUS (Alhindi et al., 2018). Besides, we pre-processed LIAR-RAW similar to RAWFC. The detailed statistics of datasets are shown in Table 1.

A.1 Data Collection and Processing.

We crawled claims with their veracity labels and relevant fact-checked reports that can be regarded as gold explanations from Snopes. For each claim, we extracted the claim-related keywords as the search query and used Google API to retrieve the top 30 relevant raw reports. To mitigate the dependency on fact-checked reports, we filtered out reports from fact-checking sites and removed the raw reports published after the publication time of the fact-checked report. We further removed the summary from the remaining articles and improved the quality of the dataset with data cleanings, e.g., removing reports containing less than 5 words or more than 3000 words. Finally, we standardized the original labels for 3-way classification: {*true*, *false*, *half*}, i.e., {true, correct attribute, mostly true} → *true*, { false, misattribute, mostly false } → *false*, {mixture, unproven } → *half*. Each sentence is annotated as evidence or not according to their similarities with the gold explanation, where we greedily extract sentences that achieve the high cosine similarity and ROUGE F1 score, referred to as *oracles*.

[5]www.snopes.com
[6]www.politifact.com

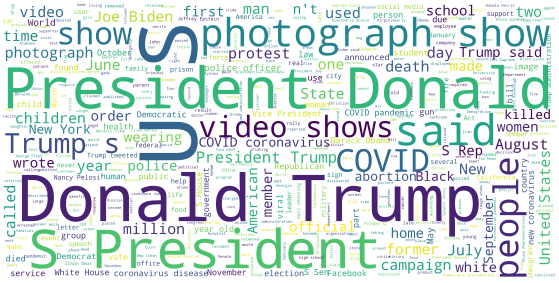

Figure A.1: The word cloud of our RAWFC.

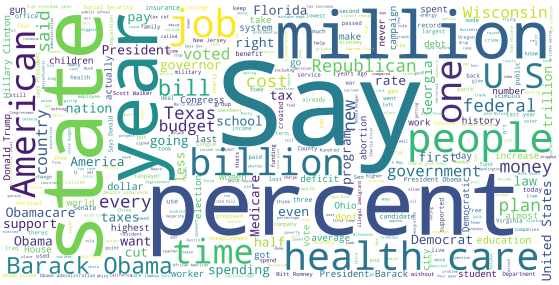

Figure A.2: The word cloud of our LIAR-RAW.

A.2 Evidential Sentence Annotation.

To help produce explanations from external raw reports, each sentence in the article is annotated as evidence or not. Different from selecting evidential sentences based merely on the ROUGE score (Lin, 2004) with gold explanations (Atanasova et al., 2020), we propose a more practical approach to annotate sentences according to both textual-level and semantic-level similarities.

For each candidate sentence, we adopt two metrics to assess whether it should be selected or not: 1) *ROUGE* measures the textual-level similarity regarding the gold explanation in terms of the n-gram overlap; and 2) *Cosine* measures the semantic similarity regarding the gold explanation. Formally, for a candidate sentence $s_{i,j} \in d_i = \{s_{i,j}\}_{j=1}^{|d_i|}$ and its corresponding explanation sentences set $E = \{e_1, e_2, ..., e_n\}$, we define the n-gram overlap function $f^{ROU}(s_{i,j}, e_i)$ and semantic similarity $f^{COS}(s_{i,j}, e_i)$ as follows:

$$f^{ROU}(s_{i,j}, e_i) = \frac{|n\text{-grams}(s_{i,j}) \cap n\text{-grams}(e_i)|}{|n\text{-grams}(e_i)|} \quad (A.1)$$

$$f^{COS}(s_{i,j}, e_i) = \cos(h_{s_{i,j}}, h_{e_i}), \quad (A.2)$$

Standardized Label	Train	Valid	Test
true	561	67	67
false	514	66	66
half	537	67	67

Table A.1: Label statistics of claims in RAWFC.

Fine-grained Label	Train	Valid	Test
pants-fire	812	115	86
false	1,958	259	249
barely-true	1,611	236	210
half-true	2,087	244	263
mostly-true	1,950	251	238
true	1,647	169	205

Table A.2: Label statistics of claims in LIAR-RAW.

where $h_{s_{i,j}}$ and h_{e_i} is the sentence representation encoded by SBERT (Reimers and Gurevych, 2019). We calculate the textual similarity in terms of *ROUGE-1*, *ROUGE-2*, and *ROUGE-L* F1 scores, respectively; we also calculate the semantic similarity in terms of *Cosine*. For sentence labeling, we empirically set the thresholds of *ROUGE-1*, *ROUGE-2*, and *ROUGE-L* F1 scores to 0.1, 0.0, and 0.1, respectively, and the threshold of *Cosine* to 0.6. Finally, we accepted the sentences that exceed all given thresholds as gold explanation sentences, i.e., *oracle*. The label statistics of claims in RAWFC and LIAR-RAW are displayed in Table 1 and Table A.2, respectively. Moreover, we also visualized their word clouds, as shown in Fig. A.1 and Fig. A.2, respectively.

B Multi-task Adaptive Weighting

Inspired by prior work (Chen et al., 2018; Liu et al., 2019), we further propose a simple yet effective remedy, namely Multi-task Adaptive Weighting (MAW), to automatically keep a dynamic balance among tasks for different benchmark datasets. We define the weighting function $\beta_k(t)$ as follows:

$$\beta_k(t) = \frac{N_k \exp[f_k(t)g(t)]}{\sum_i \exp[f_i(t)g(t)]} \tag{B.1}$$

$$f_k(t) = \frac{\mathcal{L}_k(t-1)}{\mathcal{L}_k(t-2)}, g(t) = \frac{\log(t-2)}{T} \tag{B.2}$$

where $\beta_k = \beta_k(t), k \in \{D, S, C\}$ and $f_k(t)$ represents the loss rate for task where t is an iteration step; $g(t)$ is a global function that can generate a growth value, contributing to an optimal balance between tasks, since a large T can result in a more even distribution between different tasks. $T = 8$ denotes an initial temperature to control the softness of task weighting similar to (Caruana, 1997). $N_k = 3$ indicates the total number of sub-tasks. We simply initialize $\beta_k = 0.5$ and update the average loss over each iteration.

Algorithm 1: CofCED

Input: A set of training instances $\{(c, \mathcal{D})\}$; Maximum selection number K; Thresholds ε.

Output: Veracity label $\hat{y}$; Check-worthy report labels $\hat{Y}^d$; Explainable sentence labels $\hat{Y}^s$; Generated Explanation $\hat{E}$

Initialize $\beta_D = \beta_S = \beta_C = 0.5$, if $t \leq 2$;
for *each instance* $(c, \{\{s_{i,j}\}_{j=1}^{|d_i|}\}_{i=1}^{|\mathcal{D}|})$ **do**
  {*Hierarchical Encoding*}
  $\mathbf{h}_c, \mathbf{h}_{i,j} \leftarrow$ DistilBERT;
  $\mathbf{h}_i \leftarrow$ Eq. (2);
  {*Task 1: Report Selection*}
  $\hat{y}_i^d, \{d_k^{'}\}_{k=1}^{K} \leftarrow K$; Eq. (3);
  $\mathbf{h}_D = \text{Max}([\mathbf{h}_1; \mathbf{h}_2; ...; \mathbf{h}_{|\mathcal{D}|}])$
  {*Task 2: Explainable Sentence Extraction*}
  for *each report* d_k *in* $\{d_k^{'}\}_{k=1}^{K}$ **do**
    $\hat{y}_{k,t}^s \leftarrow$ Eq. (4);
    $\{s_{k,t}\}_{t=1}^{|d_k^{'}|}, \{\mathbf{h}_{k,t}^{''}\}_{t=1}^{|d_k^{'}|} \leftarrow \hat{y}_{k,t}^s > \varepsilon_k$
  Explanations: $\hat{E} = \{\{s_{k,t}\}_{t=1}^{|d_k^{'}|}\}_{k=1}^{K}$,
  $\mathbf{h}_k^{''} = \text{Max}([\mathbf{h}_{k,1}^{''}; \mathbf{h}_{k,2}^{''}; \cdots; \mathbf{h}_{k,|d_k^{'}|}^{''}])$;
  $\mathbf{h}_E = \text{Max}([\mathbf{h}_1^{''}; \mathbf{h}_2^{''}; ...; \mathbf{h}_K^{''}])$;
  {*Task 3: Veracity Prediction*}
  $\mathbf{h}^{\dagger} = [\mathbf{h}_c; \mathbf{h}_D; \mathbf{h}_E]$
  Verdicts: $\hat{y} \leftarrow$ Eq. (9);
{*Multi-task Training*}
Optimize $\mathcal{L}_{all} = \beta_D \mathcal{L}_D + \beta_S \mathcal{L}_S + \beta_C \mathcal{L}_C \leftarrow$ Eq. (10,11,12);
Update $\beta_D, \beta_S, \beta_C$;

C CofCED Algorithm

Algorithm 1 shows our training procedure.

D Human Evaluation for Explanations

We also study the explanation quality by human evaluation referring to (Atanasova et al., 2020). Provided with three types of explanations, i.e., human justification, veracity explanation generated by CofCED, and the ones generated by GenFE-MT,

RAWFC			
Annotator	Gold	Exp-GenFE-MT	Exp-CofCED
<Informativeness>			
# 1	**1.38**	2.17	1.89
# 2	**1.63**	2.32	2.01
# 3	**1.24**	1.76	2.05
ALL	**1.42**	2.08	1.98
<Readability>			
# 1	**1.74**	1.98	1.81
# 2	**1.15**	1.76	1.63
# 3	**1.97**	2.35	2.07
ALL	**1.62**	2.03	1.84
<Overall>			
# 1	**1.54**	1.98	2.13
# 2	**1.43**	1.76	1.73
# 3	**1.60**	2.24	1.91
ALL	**1.52**	1.99	1.94
LIAR-RAW			
<Informativeness>			
# 1	**1.27**	1.91	1.82
# 2	**1.55**	2.09	1.63
# 3	**1.12**	1.72	1.46
ALL	**1.31**	1.91	1.64
<Readability>			
# 1	**1.13**	2.29	1.78
# 2	**1.38**	2.25	2.12
# 3	**1.24**	1.94	2.02
ALL	**1.25**	2.16	1.97
<Overall>			
# 1	**1.33**	1.96	1.68
# 2	**1.49**	2.12	1.94
# 3	**1.51**	2.35	2.08
ALL	**1.44**	2.14	1.90

Table C.1: Mean Average Ranks (MAR) of the explanations for each three evaluation criteria on RAWFC and LIAR-RAW, respectively. Gold denotes the explanations come from the justification, Exp-GenFE-MT denotes the explanations generated by GenFE-MT, and Exp-CofCED denotes the explanations generated by our CofCED. Best performances are shown in bold, and the second ones are underlined.

three English-speaking adult annotators were asked to rank them with 1–Good, 2–Medium, 3–Poor, according to three different criteria. To keep clear and simple, we use the following criteria:

- **Informativeness**. The explanation contains much evidential information that contributes to fake news detection.
- **Readability**. The explanation is easy to understand.
- **Overall**. The explanation is ranked based on their overall quality.

For the annotation settings, we randomly sample a set of 40 instances from the test set and prepare three candidate explanations without any other information about these explanations. All of annotators work independently.

Table C.1 shows the mean average results from the manual evaluation. We also compute Krippendorff's inter-annotator agreement (Atanasova et al., 2020) and obtain 0.37 for Informativeness, 0.43 for Readability, 0.31 for Overall. From the results, we can see that the human justification (Gold) achieves the best quality and our Exp-CofCED achieves better quality of explanations than Exp-GenFE-MT. These results suggest that the ROUGE results in Table 4 may be not sufficient for evaluating veracity explanations because the ROUGE score only accounts for word overlapping. Besides, the performance of veracity prediction in Table 2 also verifies the effectiveness of explanations in improving fake news detection. In summary, our proposed CofCED can significantly improve final fake news detection with overall better veracity explanations.

Dataset	P(%)	R(%)	macF1(%)
RAWFC	84.28	79.29	81.71
LIAR-RAW	14.98	61.06	24.06

Table E.1: Our results on report classification.

E Further Discussion

Table E.1 shows internal results about report classifications regarding precision, recall, and macro F1 score. Our model outperforms better on RAWFC than on LIAR-RAW, indicating that report classification for fine-grained claims is much challenging and further improving this part may contribute to explainable fake news detection. Similarly, Table E.2 shows internal results about explainable sentence classifications. Overall, our CofCED significantly outperforms GenFE-MT but only achieves comparable results on LIAR-RAW in terms of ROUGE scores (Table 4). This is probably because ROUGE scores w.r.t. word overlapping are not sufficient for evaluating the qualities of generated explanations. Thus, we further introduce human evaluation as a complementary measure.

F Example

Examples from RAWFC are shown in Table F.1.

Model	RAWFC			LIAR-RAW		
	P(%)	R(%)	macF1(%)	P(%)	R(%)	macF1(%)
GenFE-MT (Atanasova et al., 2020)	50.62	36.03	42.09	**43.83**	4.27	7.79
CofCED	**55.56**	**41.67**	**47.62**	14.29	**22.22**	**17.39**

Table E.2: Experimental results of explainable sentence classification regarding oracle sentences.

[**Label:** False] **Claim:** *U.S. Rep. Alexandria Ocasio-Cortez started "chain migration" deportation proceedings against First Lady Melania Trump and her parents.*
Explanation: Illegal immigration remained a top issue for U.S. President Donald Trump and continued to divide Americans in mid-2019, all the more so after Trump told several Democratic members of Congress of immigrant parentage, all but one of them born in the United States, they should "go back and help fix the totally broken and crime infested places from which they came." (...) This is simply not true. For context, "chain migration" is a term used to describe immigration procedures that allow adult U.S. citizens to obtain citizenship for foreign-born adult relatives. Reportedly, the first lady's parents secured their citizenship through just such a procedure — though we needn't belabor the point, because everything else in the story is fictional (Melania Trump's parents aren't named "Oedipus and Jezebel Beelzebub."
Raw Report Domain: *www.newsweek.com*
Content: The president have also be criticize for want to end "chain migration", a program that let U.S. citizen to sponsor immediate family member for legal residency, despite it be the program that Melania Trump use to put her parent Viktor and Amalija Knavs on a path to American citizenship. (...)
Raw Report Domain: *www.washingtonpost.com*
Content: Melania Trump' s parent be legal permanent resident, raise question about whether they rely on "chain migration" She enjoy put her personal mark on the historic home and have redesign the family live quarter. (...)
Raw Report Domain: *www.kbzk.com*
Content: Melania Trump's parent, Viktor and Amalija Knavs, also go through the immigration process, use the perjoratively call "chain migration" route the President have criticize. (...) A source with direct knowledge of Melania Trump's parent and their immigration status previously tell CNN that she have sponsor her parent for their green card, a status that allow them to live and work in the US indefinitely and pave the way for citizenship. (...)

[**Label:** True] **Claim:** *The snakehead fish can survive on land.*
Explanation: On Oct.10, 2019, many readers came across news stories about an invasive species of fish called the snakehead fish that had been discovered in Georgia. While these stories largely dealt with wildlife officials' attempts to eradicate the species, what caught the attention of most readers were brief mentions of this fish's unique ability to survive on land. CNN reported: A snakehead fish that survives on land was discovered in Georgia. Officials want it dead An invasive fish species that can breathe air and survive on land has been found in Georgia for the first time. And officials are warning anyone who comes into contact with the species to kill it immediately. The snakehead fish can truly survive on land. Here's a video of a snakehead in Thailand as it "walks," crawls, or wiggles its way back to the water.
Raw Report Domain: *www.cbsnews.com*
Content: Northern snakehead be invasive fish that can breathe air and survive for day on land. Lawrenceville, Georgia — Georgia's Department of Natural Resources have a message for angler: If you catch a northern snakehead, kill it immediately.
Raw Report Domain: *www.nytimes.com*
Content: Snakeheads can survive in freshwater and be describe a predator that can eat tiny animal, and travel across land, live out of water for several day. There have be no end to the creepy description of the snakehead fish, a slimy, toothy, large-jawed animal that can breathe on land and crawl like a snake, in the decade that it have pop up in freshwater lake, pond and river in the United States. (...)

Table F.1: Examples from RAWFC.

Document-level Event Factuality Identification via Machine Reading Comprehension Frameworks with Transfer Learning

Zhong Qian[1*], **Heng Zhang**[1], **Peifeng Li**[1,2], **Qiaoming Zhu**[1,2] and **Guodong Zhou**[1,2]
[1]School of Computer Science and Technology, Soochow University, Suzhou, China
[2]AI Research Institute, Soochow University, Suzhou, China
qianzhong@suda.edu.cn, 20204227055@stu.suda.edu.cn,
{pfli, qmzhu, gdzhou}@suda.edu.cn

Abstract

Document-level Event Factuality Identification (DEFI) predicts the factuality of a specific event based on a document from which the event can be derived, which is a fundamental and crucial task in Natural Language Processing (NLP). However, most previous studies only considered sentence-level task and did not adopt document-level knowledge. Moreover, they modelled DEFI as a typical text classification task depending on annotated information heavily, and limited to the task-specific corpus only, which resulted in data scarcity. To tackle these issues, we propose a new framework formulating DEFI as Machine Reading Comprehension (MRC) tasks considering both Span-Extraction (Ext) and Multiple-Choice (Mch). Our model does not employ any other explicit annotated information, and utilizes Transfer Learning (TL) to extract knowledge from universal large-scale MRC corpora for cross-domain data augmentation. The empirical results on DLEFM corpus demonstrate that the proposed model outperforms several state-of-the-arts.

1 Introduction

This paper focuses on Document-level Event Factuality Identification (DEFI) task, which is defined as identifying the factuality of a specific event based on a document from which the event is derived. As a sub-task in Event Factuality Identification (EFI), different from Sentence-level Event Factuality Identification (SEFI) focusing on just a single sentence, DEFI requires comprehensive understanding documents with regard to events.

Figure 1 illustrates an example of document-level event factuality. The current event is E1, i.e., "Barack Obama joins Joe Biden's cabinet", and the sentences S2-S6 contain the event mentions referring to E1. From Figure 1 we can know that: 1) Factuality of event mentions vary among sentences. S4, S5 and S6 negate E1 by the negative cues "not" and "denied", and commit to E1 as "certain negative/CT-". But some other sentences express different factuality with regard to E1. S2 evaluates E1 as "possible positive/PS+" according to the speculative cue "speculation", and S3 commits to E1 as "Underspecified/Uu" since the event mention is in the clause led by "if". However, the document-level factuality of E1 is unique, i.e., CT-. 2) In addition to E1, there are irrelevant mentions of other events in the document as well, e.g., the PS+ event "Susan Rice and Michèle Flournoy were considering for key posts under Biden"in S6, and the CT+ event "Barack Obama was elected president in 2008" in S7 , which may cause E1 to be identified as PS+ or CT+ falsely.

Event (E1): Barack Obama joins Joe Biden's cabinet.

Document: **(S1)** "I will help him in any ways that I can," Obama, said of his former vice president, in a new interview with CBS Sunday Morning's Gayle King. **(S2)** With the victory of former US Vice President Biden in the presidential election, there are speculations about **whether** Obama will return to the White House and serve in Biden's cabinet. **(S3)** King also asked **if** he would join Biden's cabinet. **(S4)** "I'm **not planning** to suddenly work on the White House staff **or** something," said Obama. **(S5)** He also jokingly responded, "There are **probably** some things I would **not** be doing, because Michelle would leave me. ... ..."**(S6)** Although Obama **denied** that he would take a position in Biden's cabinet, Susan Rice and Michèle Flournoy were among Obama administration veterans reportedly being **probably** considered for key posts under Biden. ... ... **(S7)** When Barack Obama was elected president in 2008, he became the first African American to hold the office.

Document-level Factuality: certain negative / CT-

Figure 1: An example of document-level event factuality. **Speculative cues** are **blue**, and **negative cues** are **red**.

Currently, most EFI studies limited to SEFI (Saurí and Pustejovsky, 2012; Rudinger et al., 2018; Qian et al., 2018a; Veyseh et al., 2019). While DEFI is still in its early stage, and previous work (Qian et al., 2019; Huang et al., 2019; Cao et al., 2021) usually regarded DEFI as a typical text classification task. Currently, DEFI is mainly faced

Proceedings of the 29th International Conference on Computational Linguistics, pages 2622–2632
October 12–17, 2022.

	+	-	u
CT/一定	CT+/一定发生	CT-/一定不发生	CTu/知道是否发生
PS/可能	PS+/可能发生	PS-/可能不发生	(NA)
U/未指定	(NA)	(NA)	Uu/未指定

Table 1: Event factuality values in English and Chinese.

with these limitations, i.e., 1) Most previous work on EFI only considered sentence-level task, i.e., SEFI, which means DEFI catches much less attention and has been in the preliminary phase; 2) Related models on DEFI required various annotated information, e.g., event triggers, speculative and negative cues, and cannot be applied to real world directly; 3) The performance of DEFI is limited by the scale of dataset, i.e., DLEF (Qian et al., 2019), the only available DEFI corpus, and related work did not adopt any form of data augmentation.

To address the above issues, we propose a new end-to-end paradigm for DEFI, i.e., Document-level Event Factuality identification via Machine Reading Comprehension Frameworks with Transfer Learning (DEFI-MRC-TL), casting DEFI into MRC tasks, and considering both Span-Extraction MRC (Ext-MRC) and Multiple-Choice MRC (Mch-MRC). To address the problem of data insufficiency, we adopt Transfer Learning (TL) as *Cross-Domain Data Augmentation*, which learns information from large-scale source datasets and applies it to the target dataset. Therefore, our model is comprised of two sub-models that can be denoted as Ext-TL and Mch-TL, respectively. Our MRC formulation is mainly inspired by recent studies formulating NLP tasks into MRC problems (McCann et al., 2018; Li et al., 2019; Du and Cardie, 2020; Li et al., 2020b; Liu et al., 2020). To sum up, the major contributions of our paper can be summarized as follows:

1) We propose a new framework for DEFI by formulating it as MRC tasks, and we consider both span-extraction and multiple-choice MRC.

2) We consider a transfer learning mechanism that trains our MRC model on large-scale MRC corpora (e.g., SQuAD2.0, RACE) and fine-tunes on the target dataset (i.e., DLEFM). To the best of our knowledge, this is the first DEFI model considering both MRC framework and transfer learning.

3) We construct the first MRC-style DEFI corpus, i.e., DLEFM, annotating both events and document-level event factuality. Empirical evaluations on DLEFM can prove the generalization and effectiveness of our MRC framework for end-to-end DEFI.

2 Approach

This section introduces our DEFI-MRC-TL model, where **Overview** (§2.1) gives the definitions of DEFI task, even factuality values, and transfer learning used by our model, then §2.2 and §2.3 present the data formalization and detailed structure of Ext-TL and Mch-TL model, respectively.

2.1 Overview

Document-level Event Factuality Identification can be defined as to identify a label $y \in Y$ for the event $\mathbb{E}$ (usually a sentence) based on a document $\mathbb{D}$, where Y is the set of event factuality values defined in Table 1 (Qian et al., 2019). Therefore, one sample $\mathbb{S}$ can be denoted as $\mathbb{S} = (y, \mathbb{E}, \mathbb{D})$.

Event Factuality Values are composed of modality and polarity (Saurí, 2008; Saurí and Pustejovsky, 2012). Modality conveys the certainty degree of events, including these values:

- **Certain/CT/**一定(不): It is *certain* that the event happens / does not happen.
- **Probable/PR/**很可能(不): It is *probable* that the event happens / does not happen.
- **Possible/PS/**可能(不): It is *possible* that the event happens / does not happen.
- **Underspecified/Uu/**未指定: The degree of certainty of the event is *unknown* or *uncommitted*.

while polarity expresses whether the event happens by the following values.

- **Positive / + /**正极性**/**发生：It is *certain* / *probable* / *possible* that the event happens.
- **Negative / - /**负极性**/**不发生：It is *certain* / *probable* / *possible* that the event does not happen.
- **Underspecified / u /**未指定: The polarity of the event is *unknown* or *uncommitted*.

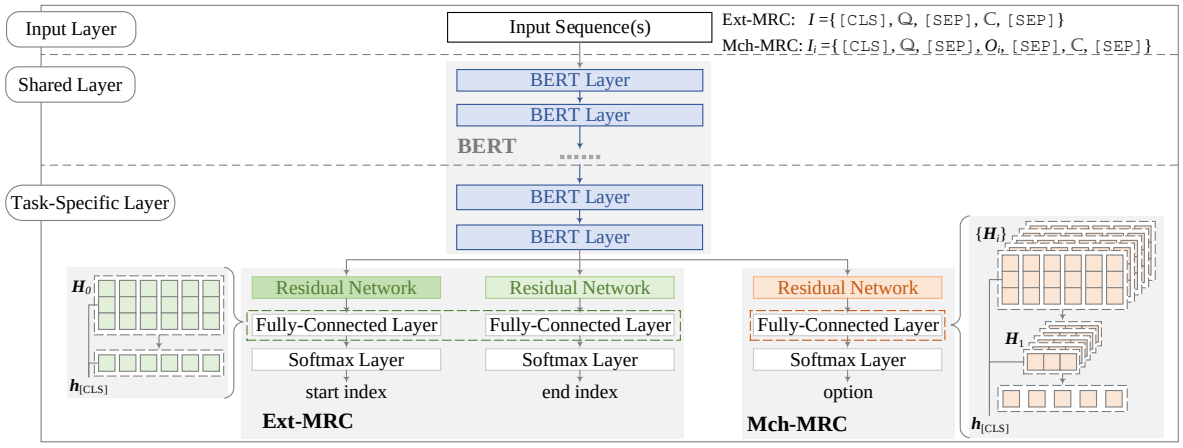

Figure 2: Overall architecture of our DEFI-MRC-TL models.

This paper uses the factuality values in Table 1 (Qian et al., 2019), where PSu and U+/- are not applicable (NA) semantically (Saurí and Pustejovsky, 2012). Compared with (Saurí and Pustejovsky, 2012), PS and PR are merged into PS due to similar semantics. Moreover, no event is annotated as CTu although applicable. Therefore, there are five applicable factuality values in DLEFM, i.e., Uu, PS-, CT-, PS+, CT+.

Transfer Learning is adopted as *Cross-Domain Data Augmentation* for DEFI, i.e., firstly learning knowledge from large-scale source datasets, and then applying it to the target datasets for fine-tuning. The architecture of DEFI-MRC-TL model is shown in Figure 2. With BERT (Devlin et al., 2019) as the backbone, our models consists of 1) **Shared Layer**, which is comprised of the first N_s layers of BERT, and optimized when training on the source datasets only; and 2) **Task-Specific Layers**, which contains the remaining layers of BERT and all the other networks in DEFI-MRC-TL, and is optimized when both training on the source datasets and fine-tuning on the target ones.

2.2 Ext-MRC Transfer Learning Model

We first give the data formalization of Ext-MRC model, and then describe the structure of Ext-TL.

Data Formalization for Ext-MRC. A sample $\mathbb{S}$ defined in §2.1 can be reformulated as a triple sample $\mathbb{S} = \{\mathbb{Q}, \mathbb{C}, \mathbb{A}\}$, where $\mathbb{Q} = \{q_0, \ldots, q_{|\mathbb{Q}|-1}\}$ is the *Question* that integrates the information of both event and candidate factuality values, $\mathbb{C} = \{c_0, \ldots, c_{|\mathbb{C}|-1}\}$ is the *Context* that refers to the document text $\mathbb{D}$ from which the event is derived, and $\mathbb{A} = \{a_0, \ldots, a_{|\mathbb{A}|-1}\} \subsetneq \mathbb{Q}$ is the *Answer* that is a sub-string of $\mathbb{Q}$, and each a_i belongs to $\mathbb{Q}$. To be specific, a $\mathbb{Q}$ in Ext-MRC can be denoted as:

- `What is the factuality of the event "`$\mathbb{E}$`", underspecified underspecified, possible negative, certain negative, possible positive, or certain positive?`

While for Chinese samples, $\mathbb{Q}$ is denoted as:

- `事件"`$\mathbb{E}$`"的事实性是未指定，可能不发生，一定不发生，可能发生，还是一定发生?`

where $\mathbb{E}$ is the event, and all the applicable factuality values are integrated into $\mathbb{Q}$. Therefore, Ext-MRC model extracts event factuality values from questions, rather than contexts.

Ext-TL Model. MRC-style input data is fed into MRC models defined below. To be in line with BERT, we concatenate the question $\mathbb{Q}$, the context $\mathbb{C}$, and the special token `[CLS]/[SEP]` as the input sequence I, which is fed into BERT:

$$I = \{[\texttt{CLS}], \mathbb{Q}, [\texttt{SEP}], \mathbb{C}, [\texttt{SEP}]\} \quad (1)$$

$$\boldsymbol{H}_0 = \text{BERT}(I) \quad (2)$$

where $\boldsymbol{H}_0 \in \mathbb{R}^{d \times N_L}$, d is the dimension of hidden states in BERT, and N_L is the length of I. During the phase of training on the source dataset, $\boldsymbol{H}_0$ is directly fed into softmax layer to compute the probability distributions of start and end indices.

In terms of fine-tuning on the target dataset, we have noticed several differences between typical MRC task (e.g., SQuAD) and our MRC framework for DEFI, i.e., 1) Instead of extracting answers from contexts $\mathbb{C}$, we extract answers from questions $\mathbb{Q}$,

and 2) Answers in typical MRC vary among documents. While in DEFI, the answer can only be one of the factuality values defined in Table 1.

Therefore, when fine-tuning the model, we utilize a variant of Residual Network (ResNet) (He et al., 2016) as the Adapter Network to encode $\boldsymbol{H}_0$ in order to bridge the information learned from source and target datasets. Actually, we use two ResNets to encode the information for the start and end indices of the answer separately:

$$\boldsymbol{H}_s = \text{ResNet}_s(\boldsymbol{H}_0) \quad (3)$$

$$\boldsymbol{H}_e = \text{ResNet}_e(\boldsymbol{H}_0) \quad (4)$$

where ResNet is composed of a stack of several residual layers, i.e., ResNet $= \{\text{ResLayer}\}$. For any input $\boldsymbol{U}_0$, each ResLayer is computed as:

$$\boldsymbol{U}_1 = \text{LN}(\text{Gelu}(\boldsymbol{W}_{r1}\boldsymbol{U}_0 + \boldsymbol{b}_{r1})) \quad (5)$$

$$\boldsymbol{U}_2 = \text{LN}(\text{Gelu}(\boldsymbol{W}_{r2}\boldsymbol{U}_1 + \boldsymbol{b}_{r2})) \quad (6)$$

$$\boldsymbol{U}_r = \boldsymbol{U}_2 + \boldsymbol{U}_0 \quad (7)$$

where $\boldsymbol{W}_{r1} \in \mathbb{R}^{h\times d}$, $\boldsymbol{b}_{r1} \in \mathbb{R}^h$, $\boldsymbol{W}_{r2} \in \mathbb{R}^{d\times h}$, and $\boldsymbol{b}_{r2} \in \mathbb{R}^d$ are parameters, h is the dimension of the hidden states in ResLayer, and LN is the Layer Normalization. Then, the probability of start and end index can be computed as:

$$\boldsymbol{p}_s = \text{softmax}(\boldsymbol{W}_s\boldsymbol{H}_s + \boldsymbol{b}_s) \quad (8)$$

$$\boldsymbol{p}_e = \text{softmax}(\boldsymbol{W}_e\boldsymbol{H}_e + \boldsymbol{b}_e) \quad (9)$$

Finally, the predicted start and end indices are obtained as:

$$\hat{i}_s = \{i|\text{argmax}(\boldsymbol{p}_s(i))\} \quad (10)$$

$$\hat{i}_e = \{i|\text{argmax}(\boldsymbol{p}_e(i))\} \quad (11)$$

where $i = 0, 1, \ldots, N_L$.

To ensure the generalization capability of Ext-MRC model on both source and target datasets, we do not discard the question $\mathbb{Q}$ or the context $\mathbb{C}$ when computing start and end index. The objective function $\mathcal{L}(\theta)$ of Ext-MRC is designed as:

$$\mathcal{L}_s(\theta) = -\frac{1}{N}\sum_{i=0}^{N-1}\log p_s(y_s^i|\theta) \quad (12)$$

$$\mathcal{L}_e(\theta) = -\frac{1}{N}\sum_{i=0}^{N-1}\log p_e(y_e^i|\theta) \quad (13)$$

$$\mathcal{L}(\theta) = \epsilon\mathcal{L}_s(\theta) + (1-\epsilon)\mathcal{L}_e(\theta) \quad (14)$$

where N is the number of samples, y_s^i and y_e^i are annotated start and end indices of the i-th sample, ϵ is the trade-off coefficient, and we set $\epsilon = 0.5$.

2.3 Mch-MRC Transfer Learning Model

Data Formalization for Mch-MRC. While in Mch-MRC, a sample $\mathbb{S}$ containing the event $\mathbb{E}$ can be represented as a quad sample, i.e., $\mathbb{S} = \{\mathbb{Q}, \mathbb{C}, \mathbb{O}, a\}$, where $\mathbb{Q} = \{q_0, \ldots, q_{|\mathbb{Q}|-1}\}$ is the *Question* that integrates the information of event, $\mathbb{C} = \{c_0, \ldots, c_{|\mathbb{C}|-1}\}$ is *Context* referring to the document $\mathbb{D}$ from which the event $\mathbb{E}$ is derived, $\mathbb{O} = \{O_0, \ldots, O_{|\mathbb{O}|-1}\}$ is the set of *Options*, and a is the *Answer* that is one of the options, i.e., $a \in \mathbb{O}$. Specifically, a $\mathbb{Q}$ in our Mch-MRC task can be denoted as:

- `What is the factuality of the event "`$\mathbb{E}$`"?`

For Chinese samples, a $\mathbb{Q}$ is denoted as:

- 事件"$\mathbb{E}$"的事实性是什么?

where $\mathbb{E}$ is the event, and $\mathbb{O}$ is the option set. For English samples, $\mathbb{O}$={Uu, PS-, CT-, PS+, CT+}, and for Chinese samples, $\mathbb{O}$={未指定/Uu, 可能不发生/PS-, 一定不发生/CT-, 可能发生/PS+, 一定发生/CT+} (Table 1).

Mch-TL Model. Similarly, our Mch-MRC model encodes each option O_i with the question $\mathbb{Q}$ and context $\mathbb{C}$ as previous work (Jin et al., 2020; Gu et al., 2021). Formally, given each option $O_i \in \mathbb{O}$, where $i = 0, \ldots, |\mathbb{O}| - 1$, we can obtain a set of input sequence $I = \{I_i\}_{i=0}^{|\mathbb{O}|-1}$ for BERT. Each I_i and its matrix representation are denoted as:

$$I_i = \{[\texttt{CLS}], \mathbb{Q}, [\texttt{SEP}], O_i, [\texttt{SEP}], \mathbb{C}, [\texttt{SEP}]\} \quad (15)$$

$$\boldsymbol{H}_i = \text{BERT}(I_i) \quad (16)$$

where BERT encodes $\{I_i\}$ as $\{\boldsymbol{H}_i\}$. For each $\boldsymbol{H}_i$, the state $\boldsymbol{h}^i_{[\texttt{CLS}]}$ of [CLS] is selected as the vector representation to make up $\boldsymbol{H}_1$:

$$\boldsymbol{H}_1 = \left\{\boldsymbol{h}^i_{[\texttt{CLS}]}\right\}_{i=0}^{|\mathbb{O}|-1} \quad (17)$$

where $\boldsymbol{H}_1 \in \mathbb{R}^{d\times N_o}$, and is fed into the softmax when training on the source datasets.

Similar to Ext-MRC model, during the phase of fine-tuning on the target dataset, we exploit residual networks as adapters to encode each $\boldsymbol{H}_i$ in Eq. 16 as well, since the text genre of the target dataset is different from that of the source dataset:

$$\tilde{\boldsymbol{H}}_i = \text{ResNet}(\boldsymbol{H}_i) \quad (18)$$

Then, we also use the hidden state of [CLS] $\tilde{\boldsymbol{h}}^i_{[\texttt{CLS}]}$ to denote each option O_i, and $\boldsymbol{H}_1$ is comprised of $\tilde{\boldsymbol{h}}^i_{[\texttt{CLS}]}$, where $i = 0, \ldots, |\mathbb{O}| - 1$ (Eq. 17).

Finally, we feed $\boldsymbol{H}_1$ into softmax layer to compute the probability distribution of each option O_i.

$$\boldsymbol{p}_o = \text{softmax}(\boldsymbol{W}_o \boldsymbol{H}_1 + \boldsymbol{b}_o) \tag{19}$$

where $\boldsymbol{W}_o \in \mathbb{R}^{1\times d}$ and $\boldsymbol{b}_o \in \mathbb{R}^1$ are parameters. The objective function is defined as:

$$\mathcal{L}(\theta) = -\frac{1}{N}\sum_{i=0}^{N-1} \log p_o(y_o^i|\theta) \tag{20}$$

where N is the number of samples, y_o^i is the annotated label of i-th sample.

3 Experimentation

This section first introduces **Experiment Settings** (§3.1), including target and source datasets, evaluation metrics, implementation details, and baselines. Then, experimental analysis focuses on the following aspects, i.e., 1) **Results and Analysis** (§3.2) discusses the comparisons of our model with baselines; 2) **Ablation** (§3.3) inspects the effectiveness of source datasets and networks in DEFI-MRC-TL model; 3) **Case Study** (§3.4) illustrates what our Ext-TL model can learn from several samples to reveal the internal mechanism of Ext-TL.

3.1 Experimental Settings

Target Dataset. DLEFM, whose formalization is defined in §2, is the target dataset to verify our DEFI-MRC-TL model, including two sub-corpora, i.e., English (DLEFM-E) and Chinese (DLEFM-C), whose statistics are shown in Table 2. The main differences between DLEFM and previous DLEF corpus lie in the following aspects:

Size. The sizes of Chinese sub-corpora are nearly the same in them. But DLEF contains more Chinese documents than English ones (4649 vs. 1727), which is less suitable to evaluate the performance on English texts. Hence, DLEFM annotates more English documents than DLEF.

Annotation. DLEF annotates not only event triggers and their sentence-level & document-level factuality, but also speculative and negative cues. And DLEFM further annotates ONE document-level event $\mathbb{E}$ based on event triggers in each document. Due to MRC framework, DLEFM corpus annotates questions for Ext-MRC (§2.2), while annotates questions and options for Mch-MRC (§2.3).

Task. Since DLEF can offer various annotated information, previous work (Qian et al., 2019; Huang et al., 2019) usually utilize annotated event triggers, sentence-level factuality, speculative and negative cues directly. While this paper aims to model DEFI as an end-to-end task, i.e., only considers questions, contexts, and options without any other explicit annotated information. Therefore, our model can apply to real-world applications directly.

	Uu	PS-	CT-	PS+	CT+	Total
English	38	46	671	594	3181	4530
Train	21	31	404	357	1905	2718
Develop	8	7	140	122	629	906
Test	9	8	127	115	647	906
Chinese	20	38	1358	860	2374	4650
Train	14	24	824	513	1415	2790
Develop	4	7	257	164	498	930
Test	2	7	277	183	461	930

Table 2: Dataset statistics of DLEFM.

Corpus	Language	Task	Used	Total
SQuAD2.0	English	Ext	20,000	130,217
NewsQA	English	Ext	20,000	103,960
RACE	English	Mch	20,000	87,866
CMRC2018	Chinese	Ext	10,111	10,111
C^3	Chinese	Mch	6,013	11,869

Table 3: MRC corpora used as source datasets, where Ext/Mch mean Ext-MRC/Mch-MRC, and "Used" & "Total" means used & total samples in training sets.

Source Datasets. For cross-domain data augmentation, the following corpora are selected as source datasets whose statistics are shown in Table 3: 1) **SQuAD2.0** (Rajpurkar et al., 2018) contains the existing SQuAD (Rajpurkar et al., 2016) collected from Wikipedia. 2) **NewsQA** (Trischler et al., 2017) collects news articles and highlights from CNN. 3) **RACE** (Lai et al., 2017) contains documents collected from the English exams for students. 4) **CMRC2018** (Cui et al., 2019) is composed of human-annotated questions on Chinese Wikipedia paragraphs. 5) $\mathbf{C}^3$ (Sun et al., 2020) is the first free-form Chinese multiple-choice MRC dataset sampled from Chinese examinations. According to the types, C^3 can be divided into C^3-Dialogue (C^3_D) and C^3-Mixed (C^3_M).

Evaluation. We focus on the performance of the three main categories of factuality values, i.e., CT-, PS+, CT+, since they occupy 98.15%/98.75% in DLEFM-E/DLEFM-C, respectively, and we do not consider the minor values (i.e., Uu and PS-) due to their small proportions as previous work (Saurí and Pustejovsky, 2012; Qian et al., 2018a, 2019). F1-score is used as the main evaluation metrics for each category of factuality values. To obtain the performance of all the values, macro- and micro-averaging F1 is also employed.

Models	CT-	PS+	CT+	Macro-Ave	Micro-Ave
LSTM-A	42.05 / 59.08	41.07 / 54.68	78.43 / 77.04	53.85 / 63.60	67.23 / 67.53
ULGN	45.87 / 61.07	43.05 / 49.58	81.87 / 76.49	56.93 / 62.38	70.55 / 66.27
BiDAF	50.66 / 67.84	49.75 / 60.94	81.06 / 81.21	60.49 / 69.99	72.86 / 73.23
BiDAF-TL	55.65 / 72.82	53.59 / 64.47	83.28 / 83.74	64.17 / 73.68	75.43 / 76.66
QANet	51.62 / 68.07	51.05 / 62.32	81.68 / 81.46	61.45 / 70.61	73.33 / 73.63
QANet-TL	55.81 / 72.91	53.25 / 65.05	83.38 / 83.39	64.14 / 73.78	75.61 / 76.58
BERT-B	54.22 / 71.11	53.11 / 63.11	81.30 / 81.55	62.88 / 71.92	74.18 / 74.84
Mch	54.09 / 71.86	52.39 / 63.66	82.01 / 81.48	62.83 / 72.33	74.42 / 75.16
Mch-TL	58.16 / 74.83	56.53 / 65.78	83.68 / 83.84	66.12 / 74.81	76.63 / 77.46
Ext	56.50 / 73.06	54.87 / 65.36	83.44 / 82.71	64.93 / 73.71	76.34 / 76.39
Ext-TL	**61.85 / 77.20**	**58.91 / 69.92**	**85.23 / 84.91**	**68.66 / 77.34**	**78.09 / 79.43**

Table 4: Performance of models on DEFI. Format: F1-scores for "English / Chinese" sub-corpus.

Models	Source Datasets	CT-	PS+	CT+	Macro-Ave	Micro-Ave
Ext-TL	SQuAD2.0	**61.85**	58.91	**85.23**	**68.66**	**78.09**
	NewsQA	60.68	**59.22**	84.85	68.25	77.87
Mch-TL	C^3-Dialogue	71.74	62.24	81.94	71.98	75.20
	C^3-Mixed	**74.83**	**65.78**	83.84	**74.81**	**77.46**
	C^3	74.04	64.03	**83.81**	73.96	76.62

Table 5: Performance of Ext-TL and Mch-TL on DEFI with difference source datasets. Format: F1-scores.

Implementation Details. BERT-Base version is chosen as the backbone of DEFI-MRC-TL model. We set 2 residual layers in the residual networks. The dimension of the hidden states of residual networks is set as 768. Adam (Kingma and Ba, 2015) is applied to optimize our model.

To take full advantage of knowledge learned from source datasets, we fine-tune as few BERT layers as possible on the target dataset. We observe that the performance on CT- and PS+ is very low (F1-score<10), or even no results can be obtained if only fine-tuning the last layer of BERT. Therefore, for those BERT-based transfer learning models, we fine-tune the last two layers of BERT (i.e., the shared layers contain $N_s = 10$ BERT layers) and all the layers of residual networks, and freeze other layers.

For all the models, we report the average evaluation metrics of the five rounds of experiments. For the training of TL models, each round adopts a subset that has a fixed size and is sampled from the source dataset randomly.

Baselines. For fair comparison with our DEFI-MRC-TL models, we employ the following models as baselines:

1) **LSTM-A** (Qian et al., 2019) uses dependency paths from cues to event triggers as syntactic features, and the sentences with event triggers as semantic features;

2) **ULGN** (Cao et al., 2021) is based on graph convolutional networks relying on event triggers;

3) **BiDAF** (Seo et al., 2017) employs bidirectional attention flow to obtain query-aware context representations;

4) **QANet** (Yu et al., 2018) adopts encoders consisting exclusively of convolution and self-attention;

5) **BERT-B** (BERT-Base) directly uses the event $\mathbb{E}$ and document $\mathbb{D}$ as the input sequence;

6) **Ext** & **Ext-TL** are Ext-MRC models, and **Mch** & **Mch-TL** are Mch-MRC models for DEFI. TL mean Transfer Learning is considered.

3.2 Results and Analysis

The comparisons of the performance of various models with our DEFI-MRC-TL model are summarized in Table 4. LSTM-A and ULGN get relatively lower results than other approaches, mainly due to the cascade errors, since we use raw texts as input for fair comparison, i.e., first extracting event triggers (F1=83.19%/79.65% for English/Chinese sub-corpus), speculative and negative cues (F1=68.80%/75.42%), then identifying document-level factuality.

BERT-B is a strong baseline compared with lightweighted models with simple structures, which can

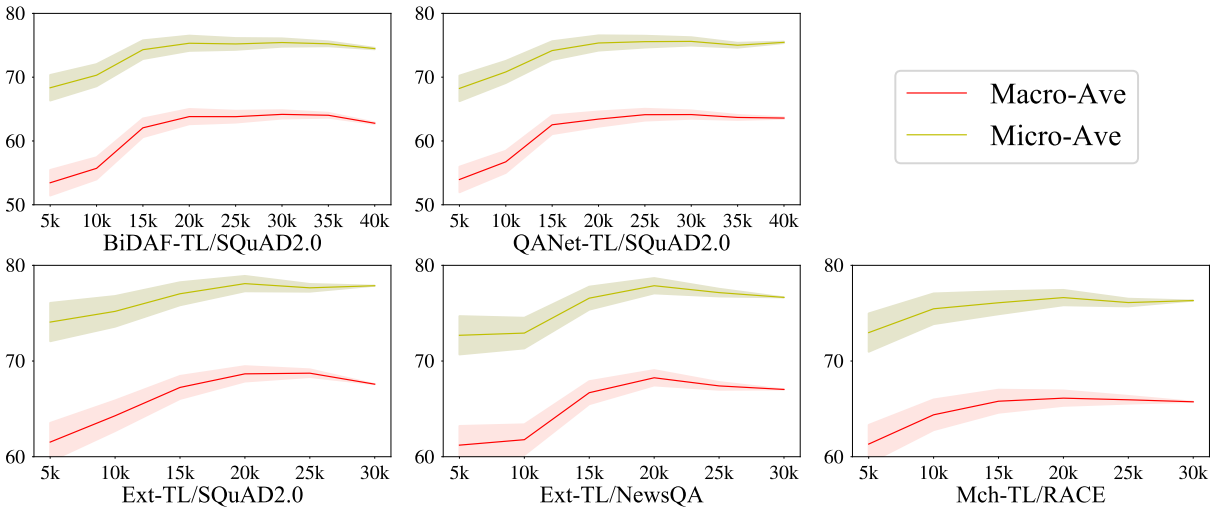

Figure 3: The test performance (F1-scores) w.r.t. the size of questions sampled from the source datasets for several transfer learning models on the English sub-corpus of DLEFM (i.e., DLEFM-E). For each sub-figure, the format of the label is "model/source dataset".

yield better results than LSTM-A and BiDAF when only fine-tuning on DLEFM, and achieve similar results with Mch.

Mch is slightly better than BERT-B. We argue that Mch can be regarded as a variant of text classification model integrating context knowledge with each option, whose input contains more information than BERT-B. Ext can get better results than BERT-B and Mch on both English and Chinese sub-corpora, which can validate the advantages of span-extraction MRC formulation for DEFI.

Table 4 demonstrates that all the TL models can achieve better performance than their corresponding original models that are only fine-tuned on target dataset, which can manifest the significance of cross-domain data augmentation using TL. For transfer learning models, BiDAF-TL and QANet-TL obtain lower performance than both Ext-TL and Mch-TL, since BiDAF and QANet have much fewer parameters than those BERT-based models. Ext-TL is superior to Mch-TL on DEFI. The main reason is that Ext-MRC models can extract meaningful and evidential texts for identifying document-level event factuality implicitly on both source and target dataset. Samples and analysis will be presented in §3.4 below.

Furthermore, Table 4 also shows that the performance on the Chinese sub-corpus is better than that on the English one, especial on CT- and PS+. The reason is that the factuality value distribution on the Chinese sub-corpus is more balanced than that on the English one.

3.3 Ablation

Text Genre of Source Dataset. We investigate the effects of source datasets with different text genres on the test performance on DLEFM, and present the performance of Ext-TL and Mch-TL w.r.t. difference source datasets in Table 5. Ext-TL achieves satisfactory performance employing either SQuAD2.0 or NewsQA as the source dataset, proving both of them can offer meaningful knowledge transferred to DEFI task. Moreover, using SQuAD2.0 obtains higher results than NewsQA, mainly owing to the high quality of Wikipedia articles with correct grammar and semantics.

Size of Source Dataset. Since the size of SQuAD2.0, NewsQA and RACE are quite large, we explore the relationship between the performance of DEFI on DLEFM-E and the scale of source dataset, and give the results in Figure 3. We can conclude that using too many samples from the source dataset can not lead to higher performance on the target dataset. For Ext-TL and Mch-TL, results can not be improved using more than 20k samples, or even degrades. It is mainly attributed to overfitting on source datasets when samples selected from them occupy too large quantities. Therefore, we adopt suitable amount of training samples from source datasets as shown in Table 3. For CMRC2018 and C^3, we leverage the whole training sets due to their limitations of the sizes.

In addition, we evaluate Mch-TL on Chinese sub-corpus. The source dataset C^3 consists of two sub-corpora, i.e, C^3_D and C^3_M. Different from all

Event (E1): Barack Obama joins Joe Biden's cabinet. [certain negative (CT-)/certain negative (CT-)]
Document: … …**whether** Obama will return to the White House and serve in Biden's cabinet. King also asked **if** he would join Biden's cabinet. "I'm **not planning** to suddenly work on the White House staff **or** something," said Obama. He also jokingly responded, "There are **probably** some things I would **not** be doing, ..."

Event (E2): New York City installs security barriers. [possible positive (PS+)/possible positive (PS+)]
Document: … … City Mayor Bill de Blasio announced the **plan** at a press conference held in Times Square on Tuesday. He said New York City **plans** to install 1,500 new security barriers in high-profile locations to guard against vehicle attacks and other terror-related incidents. ... ...

Event (E3): White House imposes new restrictions. [certain positive (CT+)/certain negative (CT-)]
Document: … …"Under **no** circumstances during a government shutdown will any government owned, rented, leased **or** chartered aircraft support any Congressional delegation, **without** the express written approval of the White House Chief of Staff," Russell Vought, … …

Event (E4): Hurricane Michael hits Florida. [possible positive (PS+)/positive, certain positive (non-applicable value, NA)]
Document: … … Hurricane Michael is **forecast** to strike Florida Panhandle in southeastern United States on Wednesday … … Hurricane Michael is currently centered about 360 miles (about 579 km) south of Panama City, Florida, and is moving north. … …

Figure 4: Spans extracted by Ext-TL are underlined. Questions are neglected when extracting spans in the documents to investigate the interpretability of Ext-TL. **Speculative cues** are **blue**, and **negative cues** are **red**. Format of labels: [Annotated/Predicted].

the other datasets considered in this paper, C^3_D is made up of dialogs. Moreover, the average context length of C^3_D (76.31) is obviously shorter than that of C^3_M (180.21) and Chinese sub-corpus in DLEFM (664.80). Hence, Mch-TL is not able to get higher results when employing the whole C^3_D, but can achieve the best performance on Chinese sub-corpus only using C^3_M as the source dataset.

Light-weighted solutions. Furtherly, in order to verify that TL models can benefit from adequate samples in the source datasets rather than only BERT-based models with large-scale and complicated structures, we also consider the light-weighted model, i.e., BiDAF and QANet based TL models, in Figure 3. QANet-TL outperforms BiDAF-TL, attributed to the more complicated attention in QANet. Compared with Ext-TL and Mch-TL, both BiDAF-TL and QANet-TL need more samples when training on the source dataset (i.e., SQuAD2.0) to reach the optimal performance, mainly due to the simpler structure of BiDAF and QANet than those BERT-based models. Both BiDAF-TL and QANet-TL are superior to BiDAF, BERT-B, and Mch, which can manifest the usefulness of transfer learning.

3.4 Case Study

As mentioned in §2, Ext-TL model discards neither questions nor contexts, and extracts answers (i.e., event factuality values) from the whole input sequence. To explore the interpretability of Ext-TL, we discard the questions in Equation 8 and 9, and extract text spans from the contexts.

As shown in Figure 4, Ext-TL identifies the correct values for events E1 and E2. In E1, the extracted span contains the mention "I'm not planning to suddenly work on the White House" that evaluates E1 as CT- according to the negative cue "not". While in E2, the extracted span commits to the event "New York City installs security barriers" as PS+ according to the speculative cue "plans".

However, E3 and E4 get wrong results. In term of E3, the extracted span contains no mention w.r.t. "White House imposes new restrictions", and another CT- event "Government aircraft support Congressional delegation" negated by "no" leads to the mistaken value of E3. While for E4, to identify it as PS+ correctly, we need to extract the event mention with speculative semantics, e.g., "Hurricane Michael is forecast to strike Florida Panhandle". But the actual span contains neither speculative nor negative semantics, extracting a non-applicable value that does not include "possible".

Therefore, these cases illustrate that correct identification of document-level event factuality relies on event mentions, speculative and negative information that governs the event.

4 Related Work

Event Factuality Identification started with SEFI, whose early work adopted rule-based models (Saurí and Pustejovsky, 2012), traditional machine learning models (de Marneffe et al., 2012; Lee et al., 2015), and hybrid models of them (Qian et al., 2018b). Recently, with the successful applications of neural networks in NLP, researchers focused on SEFI via neural networks, and captured information from sentences (He et al., 2017; Sheng et al.,

2019), sequential (Rudinger et al., 2018; Qian et al., 2018a) and graph-based (Veyseh et al., 2019) information from dependency trees, and furthurly more syntactic knowledge produced by generative adversarial networks (Qian et al., 2018a).

Compared with SEFI, DEFI remains at an initial stage, and limits to DLEF corpus (Qian et al., 2019). Previous studies (Qian et al., 2019; Huang et al., 2019) utilized multi-layer LSTM networks with attention to extract knowledge from dependency paths and sentences. Cao et al. (2021) learned local and global information of events by graph convolution networks. They relied on annotated information, e.g., event triggers, speculative and negative cues, and ignored data augmentation.

MRC/QA-Style Formulation has been widely utilized in NLP tasks over the past years, e.g, relation extraction (Li et al., 2019), named entity recognition (Li et al., 2020b), event extraction (Du and Cardie, 2020; Liu et al., 2020; Li et al., 2020a). To be specific, Li et al. (2020b) proposed a unified framework MRC handling both flat and nested NER tasks. Li et al. (2020a) designed MQAEE model casting event extraction into MRC problems to extract triggers and arguments successively. McCann et al. (2018) investigated MRC paradigms for ten tasks, including machine translation, sentiment analysis, semantic role labeling, etc.

Transfer Learning, or TL for short, is an effective technique for domain adaptation, and has achieved satisfactory results on various NLP applications, e.g., text classification (Houlsby et al., 2019; Stickland and Murray, 2019), sentiment classification (Fei and Li, 2020), neural machine translation (Aji et al., 2020), dialog system (Lin et al., 2020). Particularly, researchers also investigated TL for MRC/QA tasks. Kung et al. (2020) leveraged transfer learning to extract rationales through QA for zero-shot task transfer. Chung et al. (2018) explored both supervised and unsupervised transferability of knowledge learned among multiple-choice QA. Furthermore, some studies considered other TL paradigms, i.e., continual domain adaptation for domain drift in MRC (Su et al., 2020), and multi-task learning for QA (Wang et al., 2021; Lin et al., 2021) that is not dependent on specific domain of data.

5 Conclusion

This paper designs a novel framework formalizing Document-level Event Factuality Identification as MRC tasks, and considers both span-extraction and multiple-choice MRC. Furthermore, our model takes into account transfer learning as cross-domain data augmentation capturing extra knowledge from large-scale corpus in typical MRC. Experiments on DLEFM corpus demonstrate that our model can achieve state-of-the-art performance. In the future, we will explore cross-document event factuality identification and apply more effective data augmentation method.

Acknowledgements

The authors would like to thank the three anonymous reviewers for their comments on this paper. This research was supported by the National Natural Science Foundation of China (Nos. 62006167, 61836007 and 62276177), and Project Funded by the Priority Academic Program Development of Jiangsu Higher Education Institutions (PAPD).

References

Alham Fikri Aji, Nikolay Bogoychev, Kenneth Heafield, and Rico Sennrich. 2020. In neural machine translation, what does transfer learning transfer? In *Proceedings of ACL 2020*, pages 7701–7710. Association for Computational Linguistics.

Pengfei Cao, Yubo Chen, Yuqing Yang, Kang Liu, and Jun Zhao. 2021. Uncertain local-to-global networks for document-level event factuality identification. In *Proceedings of EMNLP 2021*, pages 2636–2645. Association for Computational Linguistics.

Yu-An Chung, Hung-yi Lee, and James R. Glass. 2018. Supervised and unsupervised transfer learning for question answering. In *Proceedings of NAACL-HLT 2018*, pages 1585–1594. Association for Computational Linguistics.

Yiming Cui, Ting Liu, Wanxiang Che, Li Xiao, Zhipeng Chen, Wentao Ma, Shijin Wang, and Guoping Hu. 2019. A span-extraction dataset for chinese machine reading comprehension. In *Proceedings of EMNLP-IJCNLP 2019*, pages 5882–5888. Association for Computational Linguistics.

Marie-Catherine de Marneffe, Christopher D. Manning, and Christopher Potts. 2012. Did it happen? the pragmatic complexity of veridicality assessment. *Comput. Linguistics*, 38(2):301–333.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: pre-training of deep bidirectional transformers for language understanding. In *NAACL-HLT 2019*, pages 4171–4186. Association for Computational Linguistics.

Xinya Du and Claire Cardie. 2020. Event extraction by answering (almost) natural questions. In *Proceedings of EMNLP 2020*, pages 671–683. Association for Computational Linguistics.

Hongliang Fei and Ping Li. 2020. Cross-lingual unsupervised sentiment classification with multi-view transfer learning. In *Proceedings of ACL 2020*, pages 5759–5771. Association for Computational Linguistics.

Yingjie Gu, Xiaoye Qu, Zhefeng Wang, Baoxing Huai, Nicholas Jing Yuan, and Xiaolin Gui. 2021. Read, retrospect, select: An MRC framework to short text entity linking. *CoRR*, abs/2101.02394.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. 2016. Deep residual learning for image recognition. In *Proceedings of CVPR 2016*, pages 770–778. IEEE Computer Society.

Tianxiong He, Peifeng Li, and Qiaoming Zhu. 2017. Identifying chinese event factuality with convolutional neural networks. In *Proceedings of CLSW 2017*, volume 10709 of *Lecture Notes in Computer Science*, pages 284–292. Springer.

Neil Houlsby, Andrei Giurgiu, Stanislaw Jastrzebski, Bruna Morrone, Quentin de Laroussilhe, Andrea Gesmundo, Mona Attariyan, and Sylvain Gelly. 2019. Parameter-efficient transfer learning for NLP. In *Proceedings of ICML 2019*, volume 97 of *Proceedings of Machine Learning Research*, pages 2790–2799. PMLR.

Rongtao Huang, Bowei Zou, Hongling Wang, Peifeng Li, and Guodong Zhou. 2019. Event factuality detection in discourse. In *Proceedings of NLPCC 2019*, volume 11839 of *Lecture Notes in Computer Science*, pages 404–414. Springer.

Di Jin, Shuyang Gao, Jiun-Yu Kao, Tagyoung Chung, and Dilek Hakkani-Tür. 2020. MMM: multi-stage multi-task learning for multi-choice reading comprehension. In *Proceedings of AAAI 2020*, pages 8010–8017. AAAI Press.

Diederik P. Kingma and Jimmy Ba. 2015. Adam: A method for stochastic optimization. In *Proceedings of ICLR 2015*.

Po-Nien Kung, Tse-Hsuan Yang, Yi-Cheng Chen, Sheng-Siang Yin, and Yun-Nung Chen. 2020. Zero-shot rationalization by multi-task transfer learning from question answering. In *Proceedings of EMNLP 2020*, pages 2187–2197. Association for Computational Linguistics.

Guokun Lai, Qizhe Xie, Hanxiao Liu, Yiming Yang, and Eduard H. Hovy. 2017. RACE: large-scale reading comprehension dataset from examinations. In *Proceedings of EMNLP 2017*, pages 785–794. Association for Computational Linguistics.

Kenton Lee, Yoav Artzi, Yejin Choi, and Luke Zettlemoyer. 2015. Event detection and factuality assessment with non-expert supervision. In *Proceedings of EMNLP 2015*, pages 1643–1648. The Association for Computational Linguistics.

Fayuan Li, Weihua Peng, Yuguang Chen, Quan Wang, Lu Pan, Yajuan Lyu, and Yong Zhu. 2020a. Event extraction as multi-turn question answering. In *Proceedings of EMNLP 2020*, pages 829–838. Association for Computational Linguistics.

Xiaoya Li, Jingrong Feng, Yuxian Meng, Qinghong Han, Fei Wu, and Jiwei Li. 2020b. A unified MRC framework for named entity recognition. In *Proceedings of ACL 2020*, pages 5849–5859. Association for Computational Linguistics.

Xiaoya Li, Fan Yin, Zijun Sun, Xiayu Li, Arianna Yuan, Duo Chai, Mingxin Zhou, and Jiwei Li. 2019. Entity-relation extraction as multi-turn question answering. In *Proceedings of ACL 2019*, pages 1340–1350. Association for Computational Linguistics.

Zhaojiang Lin, Andrea Madotto, Genta Indra Winata, and Pascale Fung. 2020. Mintl: Minimalist transfer learning for task-oriented dialogue systems. In *Proceedings of EMNLP 2020*, pages 3391–3405. Association for Computational Linguistics.

Zizheng Lin, Haowen Ke, Ngo-Yin Wong, Jiaxin Bai, Yangqiu Song, Huan Zhao, and Junpeng Ye. 2021. Multi-relational graph based heterogeneous multi-task learning in community question answering. In *Proceedings of CIKM 2021*.

Jian Liu, Yubo Chen, Kang Liu, Wei Bi, and Xiaojiang Liu. 2020. Event extraction as machine reading comprehension. In *Proceedings of EMNLP 2020*, pages 1641–1651. Association for Computational Linguistics.

Bryan McCann, Nitish Shirish Keskar, Caiming Xiong, and Richard Socher. 2018. The natural language decathlon: Multitask learning as question answering. *CoRR*, abs/1806.08730.

Zhong Qian, Peifeng Li, Yue Zhang, Guodong Zhou, and Qiaoming Zhu. 2018a. Event factuality identification via generative adversarial networks with auxiliary classification. In *Proceedings of IJCAI 2018*, pages 4293–4300. ijcai.org.

Zhong Qian, Peifeng Li, Guodong Zhou, and Qiaoming Zhu. 2018b. Event factuality identification via hybrid neural networks. In *Proceedings of ICONIP 2018*, volume 11305 of *Lecture Notes in Computer Science*, pages 335–347. Springer.

Zhong Qian, Peifeng Li, Qiaoming Zhu, and Guodong Zhou. 2019. Document-level event factuality identification via adversarial neural network. In *Proceedings of NAACL-HLT 2019*, pages 2799–2809. Association for Computational Linguistics.

Pranav Rajpurkar, Robin Jia, and Percy Liang. 2018. Know what you don't know: Unanswerable questions for squad. In *Proceedings of ACL 2018*, pages 784–789. Association for Computational Linguistics.

Pranav Rajpurkar, Jian Zhang, Konstantin Lopyrev, and Percy Liang. 2016. Squad: 100, 000+ questions for machine comprehension of text. In *Proceedings of EMNLP 2016*, pages 2383–2392. The Association for Computational Linguistics.

Rachel Rudinger, Aaron Steven White, and Benjamin Van Durme. 2018. Neural models of factuality. In *Proceedings of NAACL-HLT 2018*, pages 731–744. Association for Computational Linguistics.

Roser Saurí. 2008. *A Factuality Profiler for Eventualities in Text*. Ph.D. thesis, Brandeis University, Waltham, MA, USA.

Roser Saurí and James Pustejovsky. 2012. Are you sure that this happened? assessing the factuality degree of events in text. *Comput. Linguistics*, 38(2):261–299.

Min Joon Seo, Aniruddha Kembhavi, Ali Farhadi, and Hannaneh Hajishirzi. 2017. Bidirectional attention flow for machine comprehension. In *Proceedings of ICLR 2017*. OpenReview.net.

Jiaxuan Sheng, Bowei Zou, Zhengxian Gong, Yu Hong, and Guodong Zhou. 2019. Chinese event factuality detection. In *Proceedings of NLPCC 2019*, volume 11839 of *Lecture Notes in Computer Science*, pages 486–496. Springer.

Asa Cooper Stickland and Iain Murray. 2019. BERT and pals: Projected attention layers for efficient adaptation in multi-task learning. In *Proceedings of ICML 2019*, volume 97 of *Proceedings of Machine Learning Research*, pages 5986–5995. PMLR.

Lixin Su, Jiafeng Guo, Ruqing Zhang, Yixing Fan, Yanyan Lan, and Xueqi Cheng. 2020. Continual domain adaptation for machine reading comprehension. In *Proceedings of CIKM 2020*, pages 1395–1404. ACM.

Kai Sun, Dian Yu, Dong Yu, and Claire Cardie. 2020. Investigating prior knowledge for challenging chinese machine reading comprehension. *Trans. Assoc. Comput. Linguistics*, 8:141–155.

Adam Trischler, Tong Wang, Xingdi Yuan, Justin Harris, Alessandro Sordoni, Philip Bachman, and Kaheer Suleman. 2017. Newsqa: A machine comprehension dataset. In *Proceedings of the 2nd Workshop on Representation Learning for NLP, Rep4NLP@ACL 2017*, pages 191–200. Association for Computational Linguistics.

Amir Pouran Ben Veyseh, Thien Huu Nguyen, and Dejing Dou. 2019. Graph based neural networks for event factuality prediction using syntactic and semantic structures. In *Proceedings of ACL 2019*, pages 4393–4399. Association for Computational Linguistics.

Zhiguo Wang, Patrick Ng, Ramesh Nallapati, and Bing Xiang. 2021. Retrieval, re-ranking and multi-task learning for knowledge-base question answering. In *Proceedings of EACL 2021*.

Adams Wei Yu, David Dohan, Minh-Thang Luong, Rui Zhao, Kai Chen, Mohammad Norouzi, and Quoc V. Le. 2018. Qanet: Combining local convolution with global self-attention for reading comprehension. In *Proceedings of ICLR 2018*. OpenReview.net.

Unregulated Chinese-to-English Data Expansion Does NOT Work for Neural Event Detection

Zhongqiu Li, Yu Hong*, Jie Wang, Shiming He, Jianmin Yao, Guodong Zhou
School of Computer Science and Technology, Soochow University, China
{leosanqiu, tianxianer, jawn233, smhelpai}@gmail.com
{jyao, gdzhou}@suda.edu.cn

Abstract

We leverage cross-language data expansion and retraining to enhance neural Event Detection (abbr., ED) on English ACE corpus. Machine translation is utilized for expanding English training set of ED from that of Chinese. However, experimental results illustrate that such strategy actually results in performance degradation. The survey of translations suggests that the mistakenly-aligned triggers in the expanded data negatively influences the retraining process. We refer this phenomenon to "trigger falsification". To overcome the issue, we apply heuristic rules for regulating the expanded data, fixing the distracting samples that contain the falsified triggers. The supplementary experiments show that the rule-based regulation is beneficial, yielding the improvement of about 1.6% F1-score for ED. We additionally prove that, instead of transfer learning from the translated ED data, the straight data combination by random pouring surprisingly performs better.

1 Introduction

We tackle ED, a task of recognizing trigger words (triggers for short) that signal different types of events (Ahn, 2006). For example, the trigger "*meeting*" in (1) signals the CONTACT-MEET event.

(1) "*We have the transcript of the **meeting**.*"
Trigger: *meeting*
Type: CONTACT-MEET

The current study of ED, in general, utilizes neural classification models for determining the event types in the word-by-word manner, including the predefined ACE[1] event types and Non-trigger type. Correspondingly, supervised learning is applied to pursue the semantic-level distributed representations of words, so as to provide perceptible evidence for decoding event types.

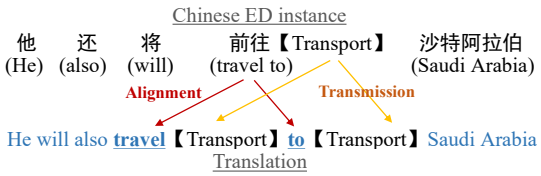

Figure 1: A case study of trigger designation and class-label transmission which is bridged by word alignment.

Expanding the training data is able to enhance the ED-oriented neural classification models. It is because there are a larger amount of knowledge and diverse pragmatic phenomena can be introduced into the supervised learning process, as claimed in different tasks of natural language processing. We childishly embrace this method, conducting expansion by translating Chinese ED corpus and pouring translations into the English training set, where word alignment (Sabet et al., 2020) is used for designating triggers in the translations.

What is beyond our expectation is that, however, cross-language data expansion actually results in performance degradation. We survey the translations which were used for expansion, and observe that a large number of low-quality instances were involved. The survey also suggests that the falsified triggers in such instances probably misled neural models during supervised learning. We provide an example in Figure 1 where Chinese event mention is taken from the publicly-shared ACE corpus of Chinese ED, and word alignment is carried out for trigger designation in English translation, as well as transmission of class label information. It can be found that the trigger "*to*" is mistakenly designated and labeled for event class (viz., TRANSPORT) due to inexact alignment, and undoubtedly it will cause severe misleading (in terms of the high occurrence frequency of "*to*" in English).

To overcome the issue, we explore a variety of easily-accessible rules to purify the translated ED instances (Section 2). Using the purified instances

*Corresponding author.
[1]https://catalog.ldc.upenn.edu/LDC2006T06

Proceedings of the 29th International Conference on Computational Linguistics, pages 2633–2638
October 12–17, 2022.

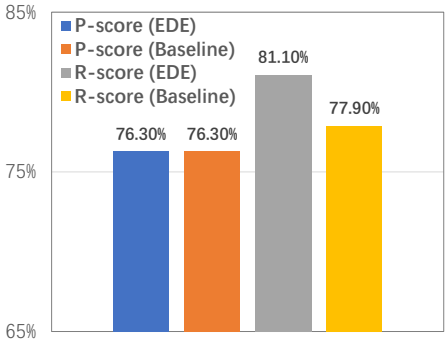

Figure 2: P and R-scores obtained by the baseline and ED𝔼♣, where RoBERTa-**base** is used.

as external data, we conduct retraining and transfer learning to strengthen the baseline ED models, including the ones which are grounded on basic and large RoBERTa (Liu et al., 2019), respectively (Section 3). Experimental results (Section 4) show that 1) the rule-based regulation helps to avoid performance degradation and yields substantial improvements, and 2) conventional expansion by combining datasets is beneficial while, on the contrary, transfer learning is less useful. We overview the related work in Section 5 before concluding this paper (Section 6).

2 Rule-based Purification Against Trigger Falsification

We apply Google translation toolkit[2] for translating Chinese event mentions into English, and use *SimAlign* (Sabet et al., 2020) to pursue the alignment between triggers in Chinese mentions and words in the corresponding translations. The aligned words are designated as triggers of translations and assigned with the manually-labeled event types in Chinese corpus.

Word alignment unavoidably falsify triggers in the translations. Therefore, we explore five heuristic rules to regulate the falsified triggers.

Unbinding prepositions It has been exhibited in Figure 1 that some prepositions (e.g., "*to*") are mistakenly designated as triggers due to inexact alignment, i.e., a Chinese trigger is aligned to the constituent that contains both verb and preposition. The number of prepositions that serve as triggers in translations is up to 326, occupying 8% of all the designated triggers. In the cases, we unbind verbs from prepositions, and designate the latter as `Non-trigger` words.

[2]https://translate.google.com

Unbinding participles In some cases, a single Chinese trigger is aligned to the present or past-participle phrase, where the participle that stands for an attributive is redundant for signaling a certain event type and, more seriously, it is common and generally leads a variety of word senses. For example, the past-participle "*opened*" in (2) is redundant. There are 38 participles found to be mistakenly designated as triggers, occupying about 1% of all the designated triggers. We repeal the designation.

(2) 坦克向两辆正常行驶的民用车辆开火
Translation: *Tanks opened fire on two normal civilian vehicles*
Chinese trigger: 开火; **Type**: `ATTACK`
Alignment: 开火="***opened*** *fire*"

Binary-choice exclusion Occasionally, a single English word is aligned with a pair of Chinese words, including not only a `Non-trigger` word but trigger. For example, both the `Non-trigger` word "提出" (i.e., "*bring*") and trigger "上诉" ("*lawsuit*") are aligned to the English word "*appealing*" in (3). In the cases, we exclude the `Non-trigger` type, but instead merely assign the concrete event type (such as `SUE` in (3)) to the aligned English word. There are 58 binary-choice cases occurred in the translations, occupying 1.4% of all the designated triggers.

(3) 我们正(提出)(上诉)
Translation: *We are appealing*
Chinese trigger: 上诉; **Type**: `SUE`
Alignment: (提出)(上诉)="*appealing*"

Correcting far-fetched triggers Before alignment, some Chinese triggers are segmented into formal characters or the ones holding less senses. As a result, the Chinese triggers are easily aligned with function words (prepositions and conjunctions) instead of content words in English. Grounded on the alignment results, the trigger designation method produces a series of far-fetched triggers. For example, the Chinese trigger "身中" (i.e., "*injured*") in (4) is mistakenly segmented into the characters "身" (i.e., *body*) and "中" ("*in*"), and the aligned preposition "*in*" is designated as the `INJURY` trigger. The number of English prepositions and conjunctions that were designated as triggers is up to 226, occupying 5.5%. We correct the errors by designating them as `Non-trigger` words.

(4) 发射了80发胡椒弹并(身中)约57发
Segmentation: (发射)(了)(80)(发)(胡椒)—

(弹)(并)(身)(中)(约)(57)(发)
Translation: *Fired 80 pepper bombs at him, with about 57 (in) his body*
Chinese trigger: 身中; **Type**: `INJURY`
Alignment: (身中)="*in*"

Skipping the omissions A large number of Chinese triggers fail to be aligned with any English word. For example, although the Chinese trigger "启用" is semantically equivalent to the English word "*opened*" in (5), the alignment is neglected. This results in the omission of triggers in translations. More seriously, the omitted triggers will be designated as `Non-trigger` word, and thus mislead classification models during training. Therefore, we skip the mentions in which trigger omission occurs. There are 426 cases of trigger omission found in the designation process, occupying 10.4% of all the Chinese triggers.

(5) 重新改建的勤务中心是在上午落成(启用)
Translation: *The remodeled service center was completed and (opened) in the morning*
Chinese trigger: 启用; **Type**: `Start-Org`
Alignment: (启用)="*None*"

3 Enhancing Classification Models

We use pretrained language models for ED, including RoBERTa-base and RoBERTa-large (Liu et al., 2019). RoBERTa-base is constructed by 12 transformer layers (Vaswani et al., 2017), each of which contains a 12-head attention network and 768 hidden states. RoBERTa-large is constructed by 24 transformer layers, each of which contains a 16-head attention network and 1,024 hidden states. The input of both RoBERTa models is a sentence no matter whether it appears as an event mention containing triggers. The maximum input length is set to 256 tokens, and padding is used if the input sentence fails to reach the length (Section 4.2 presents other hyperparameters). The initial word embeddings are obtained using look-up tables, and they are slightly strengthened by element-wise fusion with position embeddings. Besides, both RoBERTa models are connected with a linear fully-connected layer and Softmax layer (Bridle, 1990). For each word in the input sentence, the RoBERTa models conduct 34-class classification, towards not only the predefined 33 ACE event types but `Non-trigger` type.

We intend to enhance the classification models by transfer learning (Bengio, 2012) and data expansion (Journal and Alabert, 1989), using the translated Chinese ED corpus (**MT-ED** for short) as the external data. The aforementioned rule-based purification is utilized for regulating MT-ED. The considered models in experiments are as below:

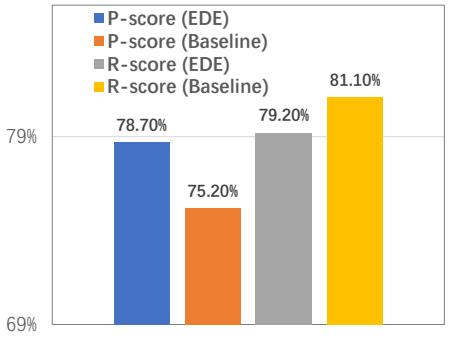

Figure 3: P and R-scores obtained by the baseline and ED$\mathbb{E}^{\clubsuit}$, where RoBERTa-**large** is used.

Baselines The baselines denotes the RoBERTa-based classifiers which are merely trained on the original training set. Such a training set contains ED instances that were split from the English corpus of the publicly shared ACE-2005 tasks.

ED$\mathbb{T}$ Transfer learning is used to enhance the RoBERTa-based classifiers. We first train the classifiers on MT-ED, and then train them on the original training set. Within the double-stage training process, the parameters obtained in the first stage (on MT-ED) are transferred to the second stage (on the original set). We refer the classifiers to ED$\mathbb{T}$s.

ED$\mathbb{E}$ We use MT-ED to expand the original training set by straight pouring, without any additional handling. Using the expanded data set, we train the RoBERTa-based classifiers from scratch. We refer the obtained classifiers to ED$\mathbb{E}$s.

4 Experimentation

4.1 Corpus and Evaluation Measure

We carry out experiments on the ACE-2005 benchmark dataset of English ED task, which comprises 599 documents. The documents contain about 5.2K manually-labeled triggers for 33 predefined event classes, and 280K `Non-trigger` words. We follow the common practice to set up the training, validation and test sets, which hold 529, 30, and 40 documents, respectively.

Besides, we use a set of Chinese ED instances which are taken from the ACE-2005 multilingual training corpus. Such data set comprises 633 documents scripted in Chinese, and involves about

3.3K triggers of 33 ACE event classes as well as 170K `Non-trigger` words. After purification, we collect 2.6K translated mentions, 2.6K triggers and about 218K `Non-trigger` words for building MT-ED. It is taken into consideration during transfer learning and data expansion (Section 3).

We evaluate all the considered classification models using the measure of Precision (P), Recall (R) and F1-score.

4.2 Hyperparameter Settings

The hyperparameters of both RoBERTa-base and RoBERTa-large are set as follows. The learning rate is set to 1e-5. We set epoch to 16 and batch size to 8. AdamW (Loshchilov and Hutter, 2017) optimizer is used where ε is set to 10e-8.

4.3 Results and Analysis

First, we examine the feasibility of cross-language data expansion for enhancing ED. The performance is indicated by ED$\mathbb{E}^*$ in Table 1, where the mark "*" denotes that ED$\mathbb{E}$ is trained on the unpurified MT-ED. It can be observed that, compared to the baseline, ED$\mathbb{E}^*$ obtains worse performance. By contrast, training ED$\mathbb{E}$ using the purified MT-ED produces substantial performance gains, as indicated by ED$\mathbb{E}^{\clubsuit}$ in Table 1. The test results reveal the necessity of data purification when MT-ED is combined with the original training set.

We compare RoBERTa-base to RoBERTa-large when different training sets are used, including the original training set, as well as the expanded version with the purified MT-ED. Table 1 shows P and R-scores they achieved, which are opposite to each other. Specifically, as indicated by *baseline* and ED$\mathbb{E}^{\clubsuit}$, RoBERTa-base achieves much higher R-score and slightly lower P-score when data expansion is used, but on the contrary, data expansion has exactly the opposite effect for RoBERTa-large. We also evaluate the performance of binary classification for triggers and `Non-trigger` words. Figure 2 shows the P and R-scores obtained by the baseline and ED$\mathbb{E}^{\clubsuit}$ when RoBERTa-base is used, while Figure 3 shows that of RoBERTa-large. It can be observed that ED$\mathbb{E}^{\clubsuit}$ achieves much higher R-score than baseline when RoBERTa-base is considered, but both of them achieve the same P-scores. On the contrary, the P and R-scores obtained when RoBERTa-large is considered change to be opposite states. The phenomena imply that the deeper neural networks like RoBERTa-large most probably overfit the common or homogeneous event instances in the original training set and MT-ED, though a small amount of novel knowledge within MT-ED is impervious to them.

RoBERTa-**base**	**P** (%)	**R** (%)	**F1** (%)
Baseline	**72.7**	74.2	73.4
ED$\mathbb{E}^*$	70.3	76.0	73.0
ED$\mathbb{E}^{\clubsuit}$	72.5	**76.9**	**74.6**$^{\triangle}$
RoBERTa-**large**	**P** (%)	**R** (%)	**F1** (%)
Baseline	72.9	**78.5**	75.6
ED$\mathbb{E}^*$	75.9	74.9	75.4
ED$\mathbb{E}^{\clubsuit}$	**76.2**	76.7	**76.5**$^{\triangle}$

Table 1: Performance of 34-class classification for ED when **data expansion** is used. The mark "*" denotes the use of unpurified MT-ED data for expansion, "$\clubsuit$" is that of purified, and "$\triangle$" indicates the significance level that p-value (Dror et al., 2018) is smaller than 0.05.

RoBERTa-**base**	**P** (%)	**R** (%)	**F1** (%)
ED$\mathbb{T}^{\clubsuit}$	69.7	76.7	73.0
ED$\mathbb{E}^{\clubsuit}$	**72.5**	**76.9**	**74.6**
RoBERTa-**large**	**P** (%)	**R** (%)	**F1** (%)
ED$\mathbb{T}^{\clubsuit}$	**77.6**	75.1	76.3
ED$\mathbb{E}^{\clubsuit}$	76.2	**76.7**	**76.5**

Table 2: Comparison between ED$\mathbb{E}^{\clubsuit}$ and ED$\mathbb{T}^{\clubsuit}$.

In a separate experiment, we compare the effect of data expansion to that of transfer learning, where ED$\mathbb{T}^{\clubsuit}$ and ED$\mathbb{E}^{\clubsuit}$ are considered. Table 2 shows the comparison results. It can be observed that ED$\mathbb{E}^{\clubsuit}$ outperforms ED$\mathbb{T}^{\clubsuit}$ for F1-score no matter what kind of RoBERTa (base or large) is used. Note that the scale of external data they take from MT-ED is the same. The comparison results suggest that asynchronous learning from exotic event knowledge to local contributes less to ED, compared to synchronous learning on the shuffled data.

5 Related Work

Conventional ED models rely heavily on elaborate feature engineering, such as that of context-independent features (Ji and Grishman, 2008), as well as cross-event (Liao and Grishman, 2010) and cross-entity (Hong et al., 2011) statistical features. In order to pursue the perception of deep event semantics, the current study concentrates on the utilization of neural networks, designing and developing a series of reliable neural ED models, including those which are grounded on CNN (Nguyen and Grishman, 2015), DMCNN (Chen et al., 2015), RNN (Nguyen et al., 2016), GAN (Hong et al.,

2018), GCN (Li et al., 2020) and VAE (Huang and Ji, 2020). Recently, the pretrained language models like BERT (Yang et al., 2019), RoBERTa (Wang et al., 2021) and AD-DMBERT (Wang et al., 2019) are used, yielding substantial improvements.

Data-driven enhancement strategies have been explored for ED, most of which are implemented by data augmentation. Yang et al. (2019) produce new ED instances by entity replacement. It is potentially effective to enhance entity-aware neural encoders for detecting events that hold entities. Tong et al. (2020) leverage knowledge distillation, which is beneficial for bringing open-domain knowledge into the understanding of local events. Veyseh et al. (2021) use GPT-2 to generate new training data. Teacher-student learning is applied for attenuating the effect of the generated noises.

6 Conclusion

We use cross-language data expansion to enhance neural ED models. Experimental results demonstrate that unregulated data expansion yields less improvement or even causes performance degradation. By contrast, data purification by simple heuristic rules produces substantial performance gains. In addition, it is proven that data expansion contributes more to ED then transfer learning.

Conducting multilingual data expansion potentially contributes to the enhancement of ED models. It is because diverse pragmatics in different languages and exotic event knowledge are informative for versatile encoding. However, it is challenging due to the lack of shareable purification rules among different languages for trigger alignment. Therefore, we will develop an automatic purification model that generalize well in different languages, where the encoding of syntactic information and reinforcement learning will be used.

Acknowledgements

The research is supported by National Key R&D Program of China (2020YFB1313601), National Science Foundation of China (62076174, 62076175).

References

David Ahn. 2006. The stages of event extraction. In *Proceedings of the Workshop on Annotating and Reasoning about Time and Events*, pages 1–8.

Yoshua Bengio. 2012. Deep learning of representations for unsupervised and transfer learning. In *Proceedings of ICML workshop on unsupervised and transfer learning*, pages 17–36. JMLR Workshop and Conference Proceedings.

John S Bridle. 1990. Training stochastic model recognition algorithms as networks can lead to maximum mutual information estimation of parameters. In *Advances in neural information processing systems*, pages 211–217.

Yubo Chen, Liheng Xu, Kang Liu, Daojian Zeng, and Jun Zhao. 2015. Event extraction via dynamic multi-pooling convolutional neural networks. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 167–176.

Rotem Dror, Gili Baumer, Segev Shlomov, and Roi Reichart. 2018. The hitchhiker's guide to testing statistical significance in natural language processing. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1383–1392.

Yu Hong, Jianfeng Zhang, Bin Ma, Jianmin Yao, Guodong Zhou, and Qiaoming Zhu. 2011. Using cross-entity inference to improve event extraction. In *Proceedings of the 49th annual meeting of the association for computational linguistics: human language technologies*, pages 1127–1136.

Yu Hong, Wenxuan Zhou, Jingli Zhang, Guodong Zhou, and Qiaoming Zhu. 2018. Self-regulation: Employing a generative adversarial network to improve event detection. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 515–526.

Lifu Huang and Heng Ji. 2020. Semi-supervised new event type induction and event detection. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 718–724.

Heng Ji and Ralph Grishman. 2008. Refining event extraction through cross-document inference. In *Proceedings of ACL-08: Hlt*, pages 254–262.

AG Journal and F Alabert. 1989. Non-gaussian data expansion in the earth sciences. *Terra Nova*, 1(2):123–134.

Manling Li, Qi Zeng, Ying Lin, Kyunghyun Cho, Heng Ji, Jonathan May, Nathanael Chambers, and Clare Voss. 2020. Connecting the dots: Event graph schema induction with path language modeling. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 684–695.

Shasha Liao and Ralph Grishman. 2010. Using document level cross-event inference to improve event

extraction. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics*, pages 789–797.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Roberta: A robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692*.

Ilya Loshchilov and Frank Hutter. 2017. Decoupled weight decay regularization. *arXiv preprint arXiv:1711.05101*.

Thien Huu Nguyen, Kyunghyun Cho, and Ralph Grishman. 2016. Joint event extraction via recurrent neural networks. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 300–309.

Thien Huu Nguyen and Ralph Grishman. 2015. Event detection and domain adaptation with convolutional neural networks. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 2: Short Papers)*, pages 365–371.

Masoud Jalili Sabet, Philipp Dufter, François Yvon, and Hinrich Schütze. 2020. Simalign: High quality word alignments without parallel training data using static and contextualized embeddings. *arXiv preprint arXiv:2004.08728*.

Meihan Tong, Bin Xu, Shuai Wang, Yixin Cao, Lei Hou, Juanzi Li, and Jun Xie. 2020. Improving event detection via open-domain trigger knowledge. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 5887–5897.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in neural information processing systems*, pages 5998–6008.

Amir Pouran Ben Veyseh, Viet Lai, Franck Dernoncourt, and Thien Huu Nguyen. 2021. Unleash gpt-2 power for event detection. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 6271–6282.

Xiaozhi Wang, Xu Han, Zhiyuan Liu, Maosong Sun, and Peng Li. 2019. Adversarial training for weakly supervised event detection. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 998–1008.

Ziqi Wang, Xiaozhi Wang, Xu Han, Yankai Lin, Lei Hou, Zhiyuan Liu, Peng Li, Juanzi Li, and Jie Zhou. 2021. Cleve: Contrastive pre-training for event extraction. *arXiv preprint arXiv:2105.14485*.

Sen Yang, Dawei Feng, Linbo Qiao, Zhigang Kan, and Dongsheng Li. 2019. Exploring pre-trained language models for event extraction and generation. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 5284–5294.

Finding Influential Instances for Distantly Supervised Relation Extraction

Zifeng Wang[1,2], **Rui Wen**[3], **Xi Chen**[3]*, **Shao-Lun Huang**[2], **Ningyu Zhang**[4], **Yefeng Zheng**[3]
[1]University of Illinois Urbana Champaign [2]TBSI, Tsinghua University
[3]Tencent [4]Zhejiang University

Abstract

Distant supervision (DS) is a strong way to expand the datasets for enhancing relation extraction (RE) models but often suffers from high label noise. Current works based on attention, reinforcement learning, or GAN are black-box models so they neither provide meaningful interpretation of sample selection in DS nor stability on different domains. On the contrary, this work proposes a novel model-agnostic instance sampling method for DS by influence function (IF), namely REIF. Our method identifies favorable/unfavorable instances in the bag based on IF, then does dynamic instance sampling. We design a fast influence sampling algorithm that reduces the computational complexity from $\mathcal{O}(mn)$ to $\mathcal{O}(1)$, with analyzing its robustness on the selected sampling function. Experiments show that by simply sampling the favorable instances during training, REIF is able to win over a series of baselines which have complicated architectures. We also demonstrate that REIF can support interpretable instance selection.

1 Introduction

To expand the training data for relation extraction (RE), distant supervision (DS) was proposed by Mintz et al. (2009) who assumed that if two entities are related in existing KBs, then all sentences contain both of them express this relation. However, this heuristic inevitably suffers from wrong labels (Takamatsu et al., 2012) and undermines model performance. For example, the sentence "*Bill Gates redefined the software industry, ... said Rob Glaser, a former Microsoft executive*" does not mention the relation *founder* but is still treated as a positive training sample in DS. Dealing with noisy instances in DS has been a focus in RE. There are three main genres in the literature: (1) incorporating an attention module (Lin et al., 2016) to allocate confidence level among instances in the same bag;

Bag Label	Instances	Influences ϕ	Probability π
/business /company /founders (Bill Gates, Microsoft)	*Bill Gates*, the founder of *Microsoft*, has indicates ···	-0.8	0.7
	··· as an investor, Allen, who founded *Microsoft* with *Bill Gates*, has ···	-0.2	0.5
	Bill Gates redefined the software industry ··· said Rob Glaser, a former *Microsoft* executive···	0.1	0.2

Figure 1: Finding influential instances within a bag via subsampling based on the calculated probability π. Note that here *negative* ϕ means a beneficial sample.

(2) using reinforcement learning (Qin et al., 2018b) for instance selection; and (3) leveraging adversarial training (Wu et al., 2017) to enhance the RE model's robustness against noise. However, they are either black-box models thus unable to provide meaningful interpretation of sample selection or sensitive to datasets. More importantly, none of them is theoretically guaranteed to truely reduce the "noise" from the dataset.

In this work, we propose to leverage influence function (IF) to evaluate instance quality then do instance selection for DS. Influence function is a powerful tool drawn from robust statistics (Huber, 2004). It is able to approximate the influence of a single data point on the whole model learned on the dataset. Creating to this merit, it has been successfully utilized for inspecting outliers (Boente et al., 2002) and denoising datasets (Wang et al., 2020) based on shallow machine learning models, e.g., logistic regression. Although Koh & Liang (2017) extends IF to interpreting deep networks, it is still elusive if it works for denoising datasets for deep networks. In this work, we develop the **R**elation **E**xtraction by **In**F**l**uence subsampling (REIF) framework, which aims for denoising DS for deep learning RE models.

The high-level idea of REIF is shown by Fig. 1. Each instance is assigned a quality measure ϕ, from which its sampling probability is obtained via the sampling function π. Accordingly, the better

* Corresponding author: jasonxchen@tencent.com

Proceedings of the 29th International Conference on Computational Linguistics, pages 2639–2650
October 12–17, 2022.

an instance's quality is, the more likely it is picked during training. We will explain the operational meaning of ϕ in Section 3.2. In a nutshell, the main contributions of this paper are

- We develop a novel IF-based denosing framework for DS RE, namely REIF, for denoising RE by sampling favorable training instances.
- An efficient implementation of REIF enables subsampling in $\mathcal{O}(1)$ complexity, instead of the $\mathcal{O}(mn)$ complexity without our implementation.
- Empirical experiments show REIF's superiority over other baselines, and we identify its capability to support interpretable instance selection for RE by a case study.[1]

2 Related Work

There are a series of works trying to address the noisy label difficulty in DS by multi-instance learning (MIL) (Hoffmann et al., 2011; Riedel et al., 2010; Surdeanu et al., 2012). MIL considers the training labels in *bag* level instead of instance level. Each bag contains at least one instance with the labeled relation while the exact label of each instance is unknown. As MIL being proved effective in relation extraction, it was firstly introduced to neural relation extraction by Zeng et al. (2015), where the piece-wise convolutional neural network (PCNN) was developed, and only one instance with the largest predicted probability was selected in each bag.

Later, attention (Lin et al., 2016; Zhou et al., 2018; Jia et al., 2019; Yuan et al., 2019; Ye and Ling, 2020; Zhou et al., 2021), reinforcement learning (Feng et al., 2018; Yang et al., 2018; Qin et al., 2018b; Chen et al., 2021), and adversarial training (Wu et al., 2017; Qin et al., 2018a; Han et al., 2018; Shi et al., 2018) have been proposed for further improvement. However, above works usually require intense trials in fine-tuning of the hyper-parameters in practice, or are not interpretable to human-beings. In this work, we propose a model-agnostic and interpretable instance selection method via IF, which is easy-to-use for most DL models without many hyperparameters to choose.

[1]Code is available in the supplementary materials.

3 Methodology

In this section, we elaborate on the major steps of REIF associated with the technical details and the theoretic foundation of measuring data quality by influences. Also, an analysis supporting our choice of sampling function is given.

3.1 Relation Extraction by Influence Subsampling

Our REIF is model-agnostic thus amenable to most DL models. Without loss of generality, we pick PCNN (Zeng et al., 2015) as the encoder for the input texts. The flowchart of our framework is shown in Fig. 2. It includes three main parts: 1) backbone model and 2) instance selection.

Backbone Model. Inputs of the encoder are raw sentences represented by indices of words, e.g., a sentence x_* with l words $x_* = \{x_{*,1}, \dots, x_{*,l}\}$. We transform them into dense real-valued representation vectors as $\boldsymbol{w}_* = \{\boldsymbol{w}_{*,1}, \dots, \boldsymbol{w}_{*,l}\}$, by concatenating the word embedding from $\boldsymbol{V} \in \mathbb{R}^{d^a \times |V|}$ (where $|V|$ denotes the size of the vocabulary and d^a is the dimension of word embedding) and position embedding with dimension d^p together. As there are two position embeddings, each word vector in $\boldsymbol{w}$ has dimension $d^a + 2 \times d^p$. Convolution layer processes the word representations as

$$\boldsymbol{x}_* = \mathrm{CNN}(\boldsymbol{w}_*). \tag{1}$$

The CNN model receives representation vectors $\boldsymbol{w}_*$ and outputs the processed feature vectors $\boldsymbol{x}_* \in \mathbb{R}^{d \times l}$. The probability for relation prediction, taking $\boldsymbol{x}_*$ as input, is given by

$$P(y = k|\boldsymbol{x}_*) = \frac{\exp(\boldsymbol{\beta}^{(k)\top} \boldsymbol{x}_*)}{\sum_{k'} \exp(\boldsymbol{\beta}^{(k')\top} \boldsymbol{x}_*)}, \tag{2}$$

where $\boldsymbol{\beta} = \{\boldsymbol{\beta}^{(1)} \dots \boldsymbol{\beta}^{(K)}\} \in \mathbb{R}^{d \times K}$ is the weight matrix of the last fully-connected layer; K is the total number of relations.

Dynamic Instance Sampling. One possible way to do sample selection by IF is *post-hoc*, i.e., it first samples from the full training set, then retrains the model on the subsamples. However, we argue it is unsuitable for DS. In post-hoc sampling, all instances are gathered together, hence the subsamples are dominated by majority relations with lots of training instances, resulting in severe class imbalance. In an extreme case, minority relations may completely disappear after subsampling.

On contrast, we propose dynamic instance sampling (DIS) which is executed within bags during

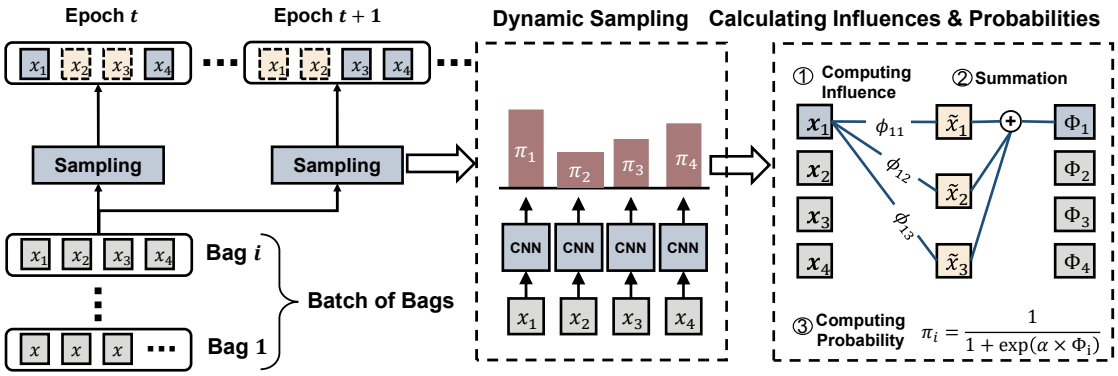

Figure 2: The flowchart of the instance-level subsampling method, where x is training sentence; $\tilde{x}$ is the validation sample; ϕ is the computed influence; and a dotted box means the instance is dropped after subsampling.

training. Given a bag $X = \{x_1, \ldots, x_n\}$ containing n sentences, we try to sample a subset X_{sub} with $|X_{sub}| < n$ from X. To this end, we calculate the influences Φ_i, $\forall i = 1, \ldots, n$, and sampling probabilities π_i are

$$\pi_i = \pi(\Phi_i) := \frac{1}{1 + \exp(\alpha \times \Phi_i)}, \tag{3}$$

where π_i is the probability of x_i being selected and α is a hyper-parameter. Consequently, the training objective function $J(\theta)$ is

$$J(\theta) = \frac{1}{|X_{sub}|} \sum_{x_i \in X_{sub}} \ell_i(\theta), \tag{4}$$

where $\ell(\theta)$ is the abbreviation of loss function $\ell(x, y; \theta)$ for notation simplicity.

3.2 Theoretic Foundation of Influence-based Sample Quality Measure

The core step of REIF is to measure the instance influence Φ. Intuitively, adverse instances, which cause model validation loss increasing, should be assigned low probability being sampled, and vice versa. We next present the property of Φ and substantiate this intuition in a rigorous way.

Consider a classification problem where we attempt to obtain a model $f_\theta : \mathcal{X} \to \mathcal{Y}$, which is parametrized by θ, that can make prediction from an input space $\mathcal{X}$ (e.g., sentences) to an output space $\mathcal{Y}$ (e.g., relations). Given a set of training data $\{x_i\}_{i=1}^n$ and the corresponding labels $\{y_i\}_{i=1}^n$, the optimal $\hat{\theta}$ defined by

$$\hat{\theta} := \arg\min_{\theta \in \Theta} \frac{1}{n} \sum_{i=1}^{n} \ell_i(\theta). \tag{5}$$

We evaluate the learned $f_{\hat{\theta}}$ on an additional validation set $\{(x_j^v, y_j^v)\}_{j=1}^m$ such as

$$L(\hat{\theta}) := \frac{1}{m} \sum_{j=1}^{m} \ell_j^v(\hat{\theta}) \tag{6}$$

where $\ell_j^v(\hat{\theta})$ is the validation loss on x_j^v.

In order to quantitatively measure the i-th training sample's influence over model's validation loss, we can perturb the training loss $\ell_i(\theta)$ by a small ϵ, then retrain a perturbed risk minimizer $\tilde{\theta}$ as

$$\tilde{\theta} := \arg\min_{\theta \in \Theta} \frac{1}{n} \sum_{i'=1}^{n} \ell_{i'}(\theta) + \epsilon \times \ell_i(\theta). \tag{7}$$

As a result, we are able to compute the validation loss change of the validation sample x_j^v by

$$\delta_j(\epsilon) := \ell_j^v(\tilde{\theta}) - \ell_j^v(\hat{\theta}). \tag{8}$$

It indicates to what extent x_i *influences* the prediction on x_j^v. If $\epsilon = -1/n$, according to Eq. (7), x_i's loss $\ell_i(\theta)$ is actually *removed* from the objective function. In this situation, $\delta_j(\epsilon) > 0$, i.e., $\ell_j^v(\tilde{\theta}) - \ell_j^v(\hat{\theta}) > 0$, implies that removing x_i causes the validation loss on x_j^v increasing, i.e.,

$$\delta_j\left(-\frac{1}{n}\right) > 0 \to x_i \text{ is good for } x_j^v. \tag{9}$$

The influence function $\phi_{i,j} := \phi(x_i, x_j^v; \hat{\theta})$ linearly approximate $\delta_j(\epsilon)$ by

$$\delta_j(\epsilon) = \ell_j^v(\tilde{\theta}) - \ell_j^v(\hat{\theta}) \simeq \epsilon \times \phi_{i,j}, \tag{10}$$

where the closed-form expression of ϕ is given in (Koh and Liang, 2017) as

$$\phi_{i,j} := -\nabla_\theta \ell_j^v(\hat{\theta})^\top H_{\hat{\theta}}^{-1} \nabla_\theta \ell_i(\hat{\theta}) \tag{11}$$

and $H_{\hat{\theta}} := \frac{1}{n} \sum_{i=1}^{n} \nabla_\theta^2 \ell_i(\hat{\theta})$ is the Hessian matrix.

In short, by Eq. (10), $\delta_j(-1/n) > 0$ is equivalent to $\phi_{i,j} < 0$. We can compute x_i's influence over the whole validation set by summation

$$\Phi_i = \sum_{j=1}^{m} \phi_{i,j} = -\sum_{j=1}^{m} \nabla_\theta \ell_j^{v\top}(\hat{\theta}) H_{\hat{\theta}}^{-1} \nabla_\theta \ell_i(\hat{\theta}). \tag{12}$$

Now, $\Phi_i < 0$ implies that x_i is *good* for the whole validation set. Also, if Φ_i is smaller, then x_i is more likely to be a favorable sample, and vice versa.

3.3 On Robustness of Sampling Functions

With the influence measure Φ, it seems that we can simply drop all *unfavorable* samples that have $\Phi > 0$. However, we argue that using 0 as the threshold usually results in failure to the out-of-sample test, due to its sensitivity to distribution shift. Instead, we take the measure of *probabilistic* sampling by designing a sampling function $\pi(\Phi) \in [0, 1]$. We give the reason of this choice based on the deviation of the induced validation loss by inaccurate estimate of influence. Let's denote the validation loss with inaccurate influence by $\ell^v(\tilde{\theta}; \hat{\Phi})$, thus

$$\Delta^2(L) := \frac{1}{m} \sum_{j=1}^{m} (\ell_j^v(\tilde{\theta}; \hat{\Phi}) - \ell_j^v(\tilde{\theta}))^2 \tag{13}$$

indicates the robustness of the model under $\hat{\Phi}$. We then give the following proposition on $\Delta^2(L)$ with respect to sampling function π. Proof can be found in Appendix A.

Proposition 1 (Robustness of Probabilistic Sampling under Inaccurate Influence)**.** *Let $\pi'(\Phi_i)$ be the derivative of $\pi(\cdot)$ function when taking Φ_i as its input, we have*

$$\sup_{\Phi, \hat{\Phi}} \Delta^2(L) = \gamma \sum_{i=1}^{n} (\pi(\hat{\Phi}_i) - \pi(\Phi_i))^2 \sum_{j=1}^{m} \phi_{i,j}^2 \tag{14}$$

$$\simeq \gamma \sum_{i=1}^{n} \Big((\hat{\Phi}_i - \Phi_i) \pi'(\Phi_i) \Big)^2 \sum_{j=1}^{m} \phi_{i,j}^2 \tag{15}$$

where γ is a constant.

It can be viewed that $\Delta^2(L)$ is controlled by the derivative of sampling function $\pi'(\Phi)$. For the sigmoid sampling in Eq. (3), it is easy to derive that

$$\pi'(\Phi) = -\alpha \pi(\Phi)(1 - \pi(\Phi)), \tag{16}$$

which means $\max |\pi'(\Phi)| = \frac{1}{4}\alpha$ when $\Phi = 0$. $\Delta^2(L)$ is hence controlled by the hyper-parameter α. When $|\Phi|$ increases, $|\pi'(\Phi)|$ reduces sharply, which ensures the variance's upper bound being tight all the time. By contrast, in deterministic sampling, $\Delta^2(L)$ is sensitive to inaccurate $\hat{\Phi}$ because it is "hard", or more rigorously, because $\Delta^2(L)$ is probably large due to large $|\pi(\Phi) - \pi(\hat{\Phi})|$ caused by an improper dropout threshold.

4 Efficient Implementation

Recap Eq. (12), computing Φ_i requires $\phi_{i,j}$ in Eq. (11) for $j = 1, \ldots, m$ on all validation samples. As a result, the computation of all $\{\Phi_i\}_{i=1}^n$ has $\mathcal{O}(mn)$ time complexity. Moreover, for DNNs with massive parameters, computing the layer-wise gradients $\nabla_\theta \ell(\theta)$ is intractable. These limitations prevent the use of IF from DL RE models. To address it, we here propose a rather efficient implementation of REIF. We demonstrate how to reduce the complexity of calculating influences from $\mathcal{O}(mn)$ to $\mathcal{O}(n)$, then to $\mathcal{O}(1)$. In addition, we show how to compute the influence function by stochastic estimation.

4.1 Computing Influences in Linear Time

We argue that in Eq. (12), it is unnecessary to calculate $\phi_{i,j}$ separately, since here we only care about their summations. Specifically, since the summation is only related to the subscript j, we can cast it to

$$\Phi_i = -\nabla_\theta \ell_i^\top(\hat{\theta}) H_{\hat{\theta}}^{-1} \sum_{j=1}^{m} \nabla_\theta \ell_j^v(\hat{\theta}) \tag{17}$$

$$= -\nabla_\theta \ell_i^\top(\hat{\theta}) H_{\hat{\theta}}^{-1} \nabla_\theta \sum_{j=1}^{m} \ell_j^v(\hat{\theta}) \tag{18}$$

$$= -m \nabla_\theta \ell_i^\top(\hat{\theta}) H_{\hat{\theta}}^{-1} \nabla_\theta L(\hat{\theta}), \tag{19}$$

where $L(\hat{\theta})$ comes from Eq. (6). By this derivation, we can calculate $L(\hat{\theta})$ rather than all $l_j(\hat{\theta})$, then take derivative of $L(\hat{\theta})$. Since $L(\hat{\theta})$ only needs to be calculated once and it is shared in calculating all Φ_is, this process only requires $\mathcal{O}(n)$ time, without loss of accuracy.

4.2 Linear Approximation for $\mathcal{O}(1)$ Complexity

$\nabla_\theta \ell(\hat{\theta})$ in Eq. (17) usually has complicated expression when $f_\theta(\cdot)$ is a neural network, hence the previous works implemented it by the auto-grad

systems like TensorFlow (Abadi et al., 2016) and PyTorch (Paszke et al., 2019). However, when the number of alternative training instances is large, even $\mathcal{O}(n)$ is not satisfactory enough, because additional differential operations need to be done on each $\ell_i(\hat{\theta})$ sequentially. Moreover, when faced with complex neural networks with massive parameters, computing the Hessian matrix $H_{\hat{\theta}}$ and its inversion is intractable. Considering these issues, we propose a linear approximation approach to reduce the complexity to $\mathcal{O}(1)$, and avoid operating on all parameters of the neural network.

Suppose the cross entropy loss function is used:

$$\ell(\theta) = -\sum_{k=1}^{K} \mathbb{I}\{y = k\} \log P(y = k|x; \theta) \quad (20)$$

where $\mathbb{I}(\cdot)$ is an indicator function. Let $\boldsymbol{y}, \hat{\boldsymbol{y}} \in \mathbb{R}^K$ be the one-hot label vector, e.g., $(1, 0, 0)^\top$, and prediction vector, e.g., $(0.8, 0.1, 0.1)^\top$, respectively. We replace $\nabla_\theta \ell(\theta)$ in Eq. (11) with the derivatives on $\boldsymbol{\beta}$ (the weight of the last fully-connected layer):

$$\nabla_\theta \ell(\theta) \Rightarrow \nabla_{\boldsymbol{\beta}} \ell(\theta) = (\hat{\boldsymbol{y}} - \boldsymbol{y})\boldsymbol{x}^\top \in \mathbb{R}^{d \times K} \quad (21)$$

where $\boldsymbol{x}$ is the input of the last fully-connected layer. This closed-form expression allows computing batch gradients in $\mathcal{O}(1)$ time. Although the calculated influence might be inaccurate, it is still reliable for measuring instances' *relative quality* in general. We will validate this claim in our experiments.

4.3 Algorithm

Algorithm 1 shows the details of REIF, please refer to Appendix B. It has two hyper-parameters: the sampling ratio r and the sigmoid sampling parameter α. The optimal value of r depends on quality of the dataset, since the higher quality it is, the more favorable instances it might have. Keeping $\alpha = 1$ is satisfactory in most scenarios.

In particular, on the line #14 of Algorithm 1, we compute the product between the inverse Hessian matrix and a gradient vector via the stochastic estimation procedure by Koh & Liang (Koh and Liang, 2017). Denoting the vector $\nabla_\theta L(\hat{\theta})$ by v, it first initializes the approximate inverse Hessian-Vector-Product (HVP) by $\tilde{H}_0^{-1} v \leftarrow v$, then repeatedly samples n_b training instances and updates as

$$\tilde{H}_t^{-1} v \leftarrow v + \left(I - \frac{1}{n_b} \sum \nabla_\theta^2 \ell(\hat{\theta}) \right) \tilde{H}_{t-1}^{-1} v \quad (22)$$

until $\tilde{H}_t^{-1} v$ converges. In our algorithm, we only need to do this once after each epoch, to get the precomputed inverse HVP $s = H_{\hat{\theta}}^{-1} \nabla_\theta L(\hat{\theta})$. Therefore, during training, we directly compute $\nabla_\theta \ell_i(\hat{\theta})$ for each instance according to Eq. (21), then multiply it with the precomputed s.

5 Experiments

We concentrate on the following research questions:

RQ1. How does our REIF perform as compared with classical baselines?

RQ2. How does the sampling ratio r influence the performance of the REIF?

RQ3. Does the sigmoid function lead to more robust sampling than the deterministic sampling?

RQ4. How does the proposed dynamic instance sampling perform compared with the post-hoc sampling using IF?

5.1 Datasets

In our experiments, we use two versions of widely used NYT datasets, the NYT-SMALL and NYT-LARGE. The small version is released in (Riedel et al., 2010), by aligning Freebase with the New York Times corpus. In particular, we use the filtered version of the NYT-SMALL released by (Zeng et al., 2015). The large version was released by (Lin et al., 2016). Data statistics can be found in Appendix C.

5.2 Experimental Setups

We pick PCNN (PCNN+ONE) (Zeng et al., 2015) as the backbone in our experiments, and include several baselines for comparison: the attention-based PCNN (PCNN+ATT) and the naive average method (PCNN+AVE) (Lin et al., 2016). Note that our REIF method is model-agnostic, hence it is applicable for other deep learning based backbones as well, e.g., CNN and RNN. Setups of models can be found in Appendix D.

We sample a clean validation set from training set by a rule-based approach used in (Jia et al., 2019), in order to obtain the inverse HVP required for calculating influences. The details of its establishment and discussions of this validation set can be found in Appendix E. During subsampling, we set $\alpha = 1$ and $r \in \{5\%, 10\%, 20\%, 30\%\}$[2] for our REIF.

[2]The ceiling function is used for rounding.

Table 1: P@N for relation extraction results, on NYT-SMALL and NYT-LARGE, where the best ones are in bold.

Dataset	NYT-SMALL				NYT-LARGE			
P@N (%)	100	200	300	Mean	100	200	300	Mean
PCNN + ONE	54.0	52.7	52.2	53.0	70.4	66.4	63.6	66.8
PCNN + AVE	52.7	50.8	47.3	50.3	73.0	71.2	67.8	70.6
PCNN + ATT	52.7	50.7	49.5	50.9	79.7	76.0	71.6	75.8
PCNN + REIF (Proposed)	**75.2**	**65.1**	**60.8**	**67.0**	**86.4**	**82.5**	**80.3**	**83.1**

Table 2: Prevision (%) of various DS methods using PCNN as backbones / other DS methods for different recalls (0.1, 0.2, 0.3) on NYT-LARGE. The results of cited methods are drawn from their papers, and the best are in bold.

PCNN	0.1	0.2	0.3	Mean
+ONE	64.7	57.1	48.9	56.9
+ATT	74.3	63.3	56.5	64.7
+ONE+soft-label (Liu et al., 2017)	71.6	62.5	54.1	62.7
+ATT+soft-label (Liu et al., 2017)	75.1	67.5	55.8	66.1
+ONE+DSGAN (Qin et al., 2018a)	65.5	57.2	50.0	57.6
+ATT+DSGAN (Qin et al., 2018a)	70.5	62.2	53.3	62.0
+PE+REINF (Zeng et al., 2018)	70.1	66.2	56.1	64.1
+ONE+RL (Qin et al., 2018b)	66.7	56.1	48.3	64.1
+ATT+RL (Qin et al., 2018b)	68.3	60.0	52.2	60.2
+ONE+ADV (Wu et al., 2017)	71.7	58.9	51.1	60.6
+ONE+AN (Han et al., 2018)	80.3	70.2	60.3	70.3
+ATT-RA+BAG-ATT (Ye and Ling, 2020)	78.8	68.9	62.1	69.9
+SATT (Zhou et al., 2021)	78.2	69.1	59.5	68.9
DISTRE (Alt et al., 2019)	65.2	64.4	60.9	63.5
RedSandT (Christou and Tsoumakas, 2021)	73.1	67.3	58.0	66.1
Trans-SA (Xiao et al., 2022)	74.1	67.2	57.9	66.4
PCNN+**REIF** (Ours)	**82.6**	**73.9**	60.9	**72.5**

5.3 Effects of Influence Subsampling (RQ1)

Fig. 3 shows the precision-recall curve in held-out evaluation of ONE, AVE, ATT, and our REIF, and Table 1 illustrates the corresponding P@N of all methods. Our REIF performs the best among all methods. In details, on NYT-SMALL, our REIF improves 14% over ONE, and 16.1% over ATT; on NYT-LARGE, the improvements are 14.1% and 5.1%, respectively, in terms of the mean P@N. Specifically, REIF only leverages part of instances during training, while ATT involves all instances but performs badly on NYT-SMALL, and ONE only picks one instance per bag. It means that neither picking too many nor too few instances gains satisfactory performance in distant supervision. On contrast, our REIF can detect and pick those favorable ones from the noisy dataset, thus achieving a better model. In distant supervision, our method is effective for achieving nice trade-off between efficiency and effectiveness. Moreover, we compare our method with many DS baselines, including adversarial training, reinforcement learning, attention, and GAN based methods, using the reported results. As shown in Table 2, REIF still is superior.

5.4 Effects of Sampling Ratio (RQ2)

We evaluate the performance of REIF with respect to different r by repeat experiments. Results are reported in Fig. 5. REIF keeps stable when sampling ratio ranges from 5% to 30%, such that adding more instances does not make much difference, which might be due to high noise in the NYT dataset, i.e., focusing on those favorable instances is enough for training a satisfactory RE model.

5.5 Effects of Sigmoid Sampling & Dynamic Sampling (RQ3, RQ4)

Our REIF is engaged with the proposed probabilistic sigmoid sampling and DIS, namely REIF+P+DIS. We would like to validate these two techniques compared with the deterministic sampling (REIF+D+DIS), and the post-hoc sampling (REIF+P+PH). Our main observations from Fig. 4

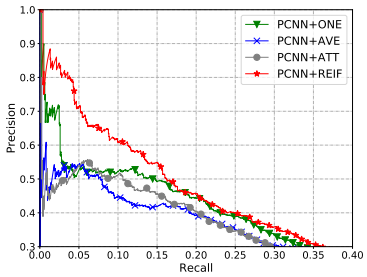

(a) NYT-SMALL

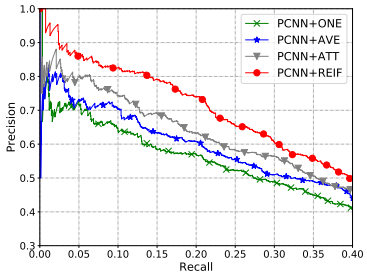

(b) NYT-LARGE

Figure 3: Aggregated precision-recall (P-R) curves obtained by PCNN+ONE, PCNN+AVE, PCNN+ATT, and the proposed PCNN+REIF on NYT-SMALL and NYT-LARGE datasets.

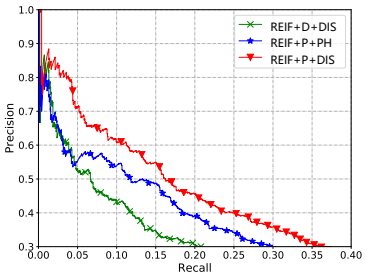

(a) NYT-SMALL

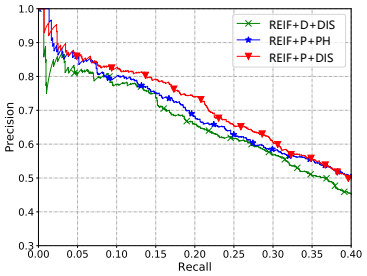

(b) NYT-LARGE

Figure 4: Precision-recall curve of compared REIF variants, where the REIF+P+DIS is the REIF with probabilistic sigmoid sampling and dynamic sampling, +D means deterministic sampling and +PH means post-hoc sampling.

are as follows:

(1) The probabilistic sigmoid sampling is crucial for robust subsampling, as the REIF+D+DIS performs the worst in both datasets. As mentioned in Proposition 1, drawbacks of REIF+D mainly come from the inaccurate estimate of influence $\hat{\Phi}$, due to the non-convexity of neural networks and the use of linear approximations. That is, we could not determine the instances that have $\hat{\Phi}$ around the threshold with very high confidence, e.g., deterministic ranking and selecting, since this causes high variance of the resulting test loss, as indicated by Eq. (14). By contrast, we should assign them similar probabilities to be sampled, as done in REIF+P, to avoid sharp variation of the test loss caused by inaccurate influences in deterministic selection.

(2) Our dynamic sampling method generally performs better than post-hoc sampling in DS, especially on the tail instances. When recall is high, REIF+DIS performs better on the minor relations, thus has higher precision than REIF+PH. In DIS, more minor relation instances are maintained, which facilitates the model's capacity of mining minor relation instances. Considering efficiency and the overall effectiveness, we shall prefer DIS in practice.

6 Manual Evaluation & Case Study

Held-out evaluation usually suffers from false negative examples in Freebase (Zeng et al., 2015). To further check our method, we perform manual evaluation by choosing the entity pairs which are labeled as "NA" but predicted a relation (not "NA") with high confidence. The top-k precisions are reported in Table 3, where the results of Mintz (Mintz et al., 2009), MultiR (Hoffmann et al., 2011), MIML (Surdeanu et al., 2012), PCNN+ONE (Zeng et al., 2015) and APCNN (Ji et al., 2017) are drawn from their papers. It could be seen our method outperforms baselines in extracting new facts from the false negative examples.

Fig. 6 reports an example of calculating influences that support instance selection. Picking a relation *children* as the example, influences and attention scores (Lin et al., 2016) are computed, from which we can identify that the influences quantitatively measure their individual quality. Recall in Section 3.2 that the smaller influences indicate better data quality. The first and the last instances are clearly right and wrong, respectively, in terms of indicating the relation *children* between their entities. By contrast, the second one tends to be right

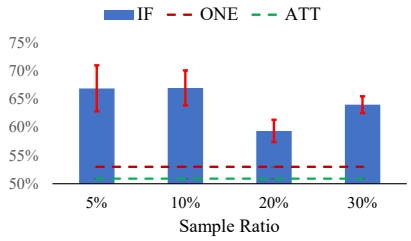

(a) NYT-SMALL

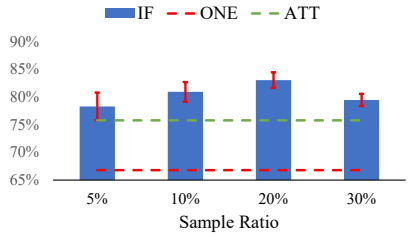

(b) NYT-LARGE

Figure 5: Mean P@N (average of P@100/200/300) varies with sampling ratio of REIF (IF) method. Red bar represents standard error by 5 times repeat experiments.

Instances	Influences	Att Scores
… because of *art rooney* , the legendary steelers ' owner … and they have continued to be a family under his oldest son , *dan rooney* .	-2.23E-02	1.11E-04
mother of *joseph* paula and walter eva, grandmother of david, lauren, *jacob*, miriam and leah .	-1.07E-04	2.61E-09
… the suspense novelists *mary higgins clark* and *carol higgins clark* signed books and posed for photographs for five hours …	1.50E-05	1.44E-07
… daughter jamie baldinger and her husband, *joseph*; son david goldring and his wife *rachel* …	7.81E-04	1.39E-09

Figure 6: Examples of influences calculated with the relation *children*, on NYT-LARGE. The words in bold are entities. The *Att Scores* (Lin et al., 2016) are standardized into $[0, 1]$ by softmax, and *Influence* is the smaller the better.

because it implies that *Joseph* is the parent of *Jacob*. Although two entities in the third instance are very similar, no evidence shows they are relatives. Therefore, sampling probabilities can be obtained via these influences for the further subsampling process.

7 Conclusion & Discussion

In this work, we proposed an efficient subsampling scheme to find the influential instances for DS, namely REIF. Our method is model-agnostic, therefore it can be engaged in the majority of RE models. REIF can be generalized to other tasks which also confront noisy data. For instance, in other weak supervision scenarios such as active learning, our method can be an effective approach to build data pipeline from data quality measure to data selection. We leave this as our future work.

Table 3: Precision values for the top 100, 200 and 500 via manual evaluation. Avg denotes the average of the former three columns. Best ones are in bold.

Accuracy (%)	Top 100	Top 200	Top 500	Avg
Mintz	77	71	55	67.7
MultiR	83	74	49	68.7
MIML	85	75	61	73.7
PCNN+ONE	86	80	69	78.3
APCNN	87	82	72	80.3
PCNN+ATT	86	81	70	79.0
PCNN+REIF	**88**	**84**	**76**	**82.7**

References

Martín Abadi, Ashish Agarwal, Paul Barham, Eugene Brevdo, Zhifeng Chen, Craig Citro, Greg S. Corrado, Andy Davis, Jeffrey Dean, Matthieu Devin, et al. 2016. TensorFlow: Large-scale machine learning on heterogeneous distributed systems. *arXiv preprint arXiv:1603.04467*.

Christoph Alt, Marc Hübner, and Leonhard Hennig. 2019. Fine-tuning pre-trained transformer language models to distantly supervised relation extraction. In *Annual Meeting of the Association for Computational Linguistics*, pages 1388–1398.

Graciela Boente, Ana M Pires, and Isabel M Rodrigues. 2002. Influence functions and outlier detection under the common principal components model: A robust approach. *Biometrika*, 89(4):861–875.

Jing Chen, Zhiqiang Guo, and Jie Yang. 2021. Distant supervision for relation extraction via noise filtering. In *International Conference on Machine Learning and Computing*, pages 361–367.

Despina Christou and Grigorios Tsoumakas. 2021. Improving distantly-supervised relation extraction through bert-based label and instance embeddings. *IEEE Access*, 9:62574–62582.

Jun Feng, Minlie Huang, Li Zhao, Yang Yang, and Xiaoyan Zhu. 2018. Reinforcement learning for relation classification from noisy data. In *Thirty-Second AAAI Conference on Artificial Intelligence*.

Xu Han, Zhiyuan Liu, and Maosong Sun. 2018. Denoising distant supervision for relation extraction via instance-level adversarial training. *arXiv preprint arXiv:1805.10959*.

Raphael Hoffmann, Congle Zhang, Xiao Ling, Luke Zettlemoyer, and Daniel S. Weld. 2011. Knowledge-based weak supervision for information extraction of overlapping relations. In *Annual Meeting of the Association for Computational Linguistics*, pages 541–550.

Peter J. Huber. 2004. *Robust Statistics*, volume 523. John Wiley & Sons.

Guoliang Ji, Kang Liu, Shizhu He, Jun Zhao, et al. 2017. Distant supervision for relation extraction with sentence-level attention and entity descriptions. In *AAAI Conference on Artificial Intelligence*, volume 3060.

Wei Jia, Dai Dai, Xinyan Xiao, and Hua Wu. 2019. ARNOR: Attention regularization based noise reduction for distant supervision relation classification. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 1399–1408.

Pang Wei Koh and Percy Liang. 2017. Understanding black-box predictions via influence functions. In *Proceedings of the 34th International Conference on Machine Learning-Volume 70*, pages 1885–1894.

Yankai Lin, Shiqi Shen, Zhiyuan Liu, Huanbo Luan, and Maosong Sun. 2016. Neural relation extraction with selective attention over instances. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics*, pages 2124–2133.

Tianyu Liu, Kexiang Wang, Baobao Chang, and Zhifang Sui. 2017. A soft-label method for noise-tolerant distantly supervised relation extraction. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 1790–1795.

Mike Mintz, Steven Bills, Rion Snow, and Dan Jurafsky. 2009. Distant supervision for relation extraction without labeled data. In *Proceedings of the 47th Annual Meeting of the Association for Computational Linguistics*, pages 1003–1011. Association for Computational Linguistics.

Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, et al. 2019. PyTorch: An imperative style, high-performance deep learning library. In *Advances in Neural Information Processing Systems*, pages 8024–8035.

Pengda Qin, Weiran Xu, and William Yang Wang. 2018a. DSGAN: Generative adversarial training for distant supervision relation extraction. *arXiv preprint arXiv:1805.09929*.

Pengda Qin, Weiran Xu, and William Yang Wang. 2018b. Robust distant supervision relation extraction via deep reinforcement learning. *arXiv preprint arXiv:1805.09927*.

Sebastian Riedel, Limin Yao, and Andrew McCallum. 2010. Modeling relations and their mentions without labeled text. In *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*, pages 148–163.

Ge Shi, Chong Feng, Lifu Huang, Boliang Zhang, Heng Ji, Lejian Liao, and He-Yan Huang. 2018. Genre separation network with adversarial training for cross-genre relation extraction. In *Conference on Empirical Methods in Natural Language Processing*, pages 1018–1023.

Mihai Surdeanu, Julie Tibshirani, Ramesh Nallapati, and Christopher D. Manning. 2012. Multi-instance multi-label learning for relation extraction. In *Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning*, pages 455–465.

Shingo Takamatsu, Issei Sato, and Hiroshi Nakagawa. 2012. Reducing wrong labels in distant supervision for relation extraction. In *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics*, pages 721–729.

Zifeng Wang, Hong Zhu, Zhenhua Dong, Xiuqiang He, and Shao-Lun Huang. 2020. Less is better: Unweighted data subsampling via influence function. In *Proceedings of the AAAI Conference on Artificial Intelligence*.

Yi Wu, David Bamman, and Stuart Russell. 2017. Adversarial training for relation extraction. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, pages 1778–1783.

Yan Xiao, Yaochu Jin, Ran Cheng, and Kuangrong Hao. 2022. Hybrid attention-based transformer block model for distant supervision relation extraction. *Neurocomputing*, 470:29–39.

Yaosheng Yang, Wenliang Chen, Zhenghua Li, Zhengqiu He, and Min Zhang. 2018. Distantly supervised NER with partial annotation learning and reinforcement learning. In *Proceedings of the 27th International Conference on Computational Linguistics*, pages 2159–2169.

Zhi-Xiu Ye and Zhen-Hua Ling. 2020. Distant supervision relation extraction with intra-bag and inter-bag attentions. In *Conference of the North American Chapter of the Association for Computational Linguistics*, pages 2810–2819.

Yujin Yuan, Liyuan Liu, Siliang Tang, Zhongfei Zhang, Yueting Zhuang, Shiliang Pu, Fei Wu, and Xiang Ren. 2019. Cross-relation cross-bag attention for distantly-supervised relation extraction. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, pages 419–426.

Matthew D Zeiler. 2012. ADADELTA: an adaptive learning rate method. *arXiv preprint arXiv:1212.5701*.

Daojian Zeng, Kang Liu, Yubo Chen, and Jun Zhao. 2015. Distant supervision for relation extraction via piecewise convolutional neural networks. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, pages 1753–1762.

Xiangrong Zeng, Shizhu He, Kang Liu, and Jun Zhao. 2018. Large scaled relation extraction with reinforcement learning. In *AAAI Conference on Artificial Intelligence*, pages 5658–5665.

Peng Zhou, Jiaming Xu, Zhenyu Qi, Hongyun Bao, Zhineng Chen, and Bo Xu. 2018. Distant supervision for relation extraction with hierarchical selective attention. *Neural Networks*, 108:240 – 247.

Yanru Zhou, Limin Pan, Chongyou Bai, Senlin Luo, and Zhouting Wu. 2021. Self-selective attention using correlation between instances for distant supervision relation extraction. *Neural Networks*, 142:213–220.

A Proof of Proposition 1

Proposition 1 (Robustness of Probabilistic Sampling under Inaccurate Influence)**.** *Let $\pi'(\Phi_i)$ be the derivative of $\pi(\cdot)$ function when taking Φ_i as its input, we have*

$$\begin{aligned}\sup_{\Phi,\hat{\Phi}} \Delta^2(L) &= \gamma \sum_{i=1}^{n} (\pi(\hat{\Phi}_i) - \pi(\Phi_i))^2 \sum_{j=1}^{m} \phi_{i,j}^2 \\ &\simeq \gamma \sum_{i=1}^{n} \Big((\hat{\Phi}_i - \Phi_i)\pi'(\Phi_i) \Big)^2 \sum_{j=1}^{m} \phi_{i,j}^2\end{aligned} \tag{A.1}$$

where γ is a constant.

Proof.

$$\Delta^2(L) \propto \sum_{j=1}^{m} (\ell_j^v(\tilde{\theta}; \hat{\Phi}) - \ell_j^v(\tilde{\theta}))^2 \tag{A.2}$$

$$= \sum_{j=1}^{m} (\ell_j^v(\tilde{\theta}; \hat{\Phi}) - \ell_j^v(\hat{\theta}) + \ell_j(\hat{\theta}) - \ell_j^v(\tilde{\theta}))^2 \tag{A.3}$$

$$\propto \sum_{j=1}^{m} \left(\sum_{i=1}^{n} \pi(\hat{\Phi}_i)\phi_{i,j} - \pi(\Phi_i)\phi_{i,j} \right)^2 \tag{A.4}$$

$$\leq \sum_{i=1}^{n} (\pi(\hat{\Phi}_i) - \pi(\Phi_i))^2 \sum_{j=1}^{m} \phi_{i,j}^2 \tag{A.5}$$

Eq. (A.4) is obtained by definition of probabilistic subsampling because

$$\begin{aligned}\ell_j^v(\tilde{\theta}) - \ell_j^v(\hat{\theta}) &\simeq \sum_{i=1}^{n} \epsilon_i \phi_{i,j} \\ &\propto \sum_{i=1}^{n} \pi(\Phi_i)\phi_{i,j}.\end{aligned} \tag{A.6}$$

Details can be referred to (Wang et al., 2020). Taking linear Taylor expansion of the $\pi(\hat{\Phi}_i) - \pi(\Phi_i)$ at the last line yields the final result. □

B Algorithm

C Dataset Statistics

D General Setups for Training PCNN

Following the configurations of previous works, we employ word2vec[3] to extract the word embeddings, to process the raw data. Parameters of PCNN are set according to (Zeng et al., 2015): window size $d^w = 3$, sentence embedding size $d^s = 230$, word dimension $d^a = 50$ and position dimension $d^p = 5$ for fair comparison. During training, we fix the batch size $B = 128$, dropout ratio $p = 0.5$, and use the ADADELTA (Zeiler, 2012) with parameters $\rho = 0.95$ and $\varepsilon = 10^{-6}$ for optimization. Since we find the default hyperparameters already lead superior performance of REIF, we did not make further tuning.

Algorithm 1 Finding Influential Instances for DS on RE by Influence Subsampling.

Input: Training and validation data $\mathcal{D}_{tr}, \mathcal{D}_{va}$; Hyper-parameters: r and α;

for epoch $t = 1 \to T$ **do**
  repeat
    Initialize the selected instances set $X_{sub} = \emptyset$;
    Sequentially sample a batch of bags $\{X_1, \ldots, X_B\}$ from $\mathcal{D}_{tr}$;
    for bag $b = 1 \to B$ **do**
      Obtain instance-level loss as $\vec{\ell} \leftarrow (\ell_1(\hat{\theta}_t), \ldots, \ell_{|X_b|}(\hat{\theta}_t))^\top$;
      Compute influences $\Phi_i \leftarrow s_t^\top \nabla_\theta \ell_i(\hat{\theta}_t)\ \forall i = 1, \ldots, |X_b|$;
      Compute sampling probability $\pi_i \leftarrow 1/(1 + \exp(\alpha \times \Phi_i))\ \forall i$;
      Sample $r \times |X_b|$ instances from X_b to get $\tilde{X}_b$, and $X_{sub} \leftarrow X_{sub} \cup \tilde{X}_b$;
    end for
    Update $\hat{\theta}_t$ using the selected subset X_{sub} by gradient descent;
  until going through all bags in $\mathcal{D}_{tr}$.
  Get validation loss by $L(\hat{\theta}_t) \leftarrow \frac{1}{m}\sum_{j=1}^{m} \ell_j^v(\hat{\theta}_t)$ on $\mathcal{D}_{va}$;
  Obtain $s_t \leftarrow H_t^{-1} \nabla_\theta L(\hat{\theta}_t)$ by stochastic estimation as done in Eq. (22);
end for

E Establishing the Validation Set

Due to lacking clean validation set, we utilize automatic selection similar to ARNOR (Jia et al., 2019). It takes top 10% high-frequency patterns of each relation as initial pattern, then takes max 5 new patterns in one loop for each relation in bootstrap procedure. We stop bootstrap until 10% training samples are involved. Our experiments demonstrate REIF can gain significantly from this automatically built validation set, although it is collected by heuristics and not absolutely clean.

[3] https://code.google.com/p/word2vec/

Table 1: Data statistics of used two NYT datasets. "# Pos", "# Ins", "# Rel": number of postive bags, instances and relations, respectively.

	NYT-SMALL		NYT-LARGE	
	Train	Test	Train	Test
# Bags	65,726	93,574	281,270	96,678
# Pos	4,266	1,732	18,252	1,950
# Ins	112,941	152,416	522,611	172,448
# Rel	26	26	53	53

A Simple Model for Distantly Supervised Relation Extraction

Ziqin Rao, Fangxiang Feng, Ruifan Li*, Xiaojie Wang
School of Artificial Intelligence, Beijing University of Posts and Telecommunications, China
{ziqin_rao, fxfeng, rfli, xjwang}@bupt.edu.cn

Abstract

Distantly supervised relation extraction is challenging due to the noise within data. Recent methods focus on exploiting bag representations based on deep neural networks with complex de-noising scheme to achieve remarkable performance. In this paper, we propose a simple but effective **B**ERT-based **G**raph convolutional network **M**odel (i.e., BGM). Our BGM comprises of an instance embedding module and a bag representation module. The instance embedding module uses a BERT-based pretrained language model to extract key information from each instance. The bag representaion module constructs the corresponding bag graph then apply a convolutional operation to obtain the bag representation. Our BGM model achieves a considerable improvement on two benchmark datasets, i.e., NYT10 and GDS[1].

1 Introduction

In the distant supervision relation extraction (DS-RE) setting, handling the noisy training data is a major challenge for downstream applications. To alleviate the severe noise problem in DS-RE, Riedel et al. (2010) incorporate the multi-instance learning (MIL) framework. Under this framework, the instances (i.e., sentences) for an identical entity pair are regarded as a bag. However, learning effective bag representations from the noisy data is a challenge resulting in unsatisfactory performance.

Recently, to obtain effective bag representations, various neural models are incorporated. Lin et al. (2016) propose a selective attention mechanism to capture relatively informative instances forming the bag representation. Vashishth et al. (2018) use graph convolution network (GCN) (Kipf and Welling, 2017) to encode syntactic information obtained from a dependency parser. However, the GCN's capacity in capturing the correlation among instances is not sufficiently explored.

Meanwhile, a prevalent trend is to use pretrained language models (PLMs) for various NLP tasks (Alberti et al., 2019; Tang et al., 2021; Wang et al., 2021; Li et al., 2021). PLMs work without explicit linguistic features and side-information like POS tags and entity types. Alt et al. (2019) use a PLM to incorporate more linguistic and semantic information. The correlations among instances are implicitly represented through a naïve attention mechanism. Very recently, Chen et al. (2021) propose PLMs combined with the contrastive instance learning (CIL) to build bag representations. CIL captures the correlation among instances through data augmentation with positive and negative pairs. These PLMs-based methods achieve a fabulous performance. Hence, a question arises, how about combining PLMs and GCNs to learn the instance correlations for bag representations?

Towards this goal, we propose an impressively simple model, i.e, BGM. Our BGM comprises of a PLM (e.g., BERT) and a concise GCN. The PLM brings accurate contextual representations for instances. The GCN whose nodes are instances encodes the correlation of instances in a bag through mutually aggregation. Thus, we finetune an off-the-shelf BERT with a GCN in an end-to-end fashion. With the obtained bag representation, the entity relation in a given sentence is then predicted.

Our contributions are twofold. **1**) We propose BGM model for DS-RE. Our BGM only comprises a PLM and a GCN but without any prior knowledge or data augmentation. **2**) Experimental results on two benchmark datasets show that BGM achieves a consistent improvement on performance.

2 Related Work

The noisy data is a major challenge in DS-RE for downstream applications. Previous works can be divided into two categories: PCNN-based methods

*Corresponding author.

[1]Code and datasets are available at https://github.com/ziqinrao.

Proceedings of the 29th International Conference on Computational Linguistics, pages 2651–2657
October 12–17, 2022.

and PLMs-based methods. The former methods use Piecewise Convolutional Neural Networks (i.e., PCNN) as the backbone to encode sentences. Thus, various methods are proposed to acquire effective bag representations. Lin et al. (2016) propose the selective attention mechanism over the bag's instances to use more informative instances. This method implicitly models the instance correlations. Subsequently, Liu et al. (2017) propose a model by providing a better supervision with soft labels as golden labels. Han et al. (2018) exploit a hierarchical attention paradigm to better capture valid instances. Combined with previous intra-bag attention, Ye and Ling (2019) design an inter-bag attention to obtain bag-group representations. Cao et al. (2021) build a co-occurrence graph to learn embeddings to enhance bag representations. Shang et al. (2022) employ a pattern-aware self-attention network to automatically discover relational patterns for pre-trained transformers.

Recently, PLMs-based methods achieve remarkable performance in various NLP tasks, including DS-RE. Alt et al. (2019) adopt PLMs to incorporate a great deal of commonsense knowledge. Christopoulou et al. (2021) focus on relational tokens using the sub-tree parsing and capture informative instances with fine-tuning BERT. Chen et al. (2021) combine PLMs with contrastive learning with data augmentation to improve the overall performance. Note that the three PLMs-based methods follow the soft attention mechanism adopted in (Lin et al., 2016). The attention mechanism uses the bag's target relation to emphasize the instances which better express the bag relation.

In this paper, we propose a simple but effective model BGM. The BGM uses a PLM and a GCN for DS-RE. To the best of our knowledge, we are the first to use GCN to learn the bag representation *directly over the instances*. Note that Vashishth et al. (2018) also use GCNs but for encoding the syntactic tree of instances. In addition, compared with (Lin et al., 2016), we do not follow their selective attention with bag's target relation. Our BGM uses a self-attention mechanism to capture the correlation among instances.

3 Method

Suppose that an instance bag $\mathcal{B}^{(e^h,e^t)} = \{s_1, s_2, ..., s_{n^s}\}$ contains n^s instances which all include the entity-pair (e^h, e^t). Each instance $s = \{x_1, x_2, ..., x_{n^w}\}$ contains n^w words. The DS-RE task aims to extract the specific relation between the entity-pair (e^h, e^t) (i.e., the head and tail entities). To this aim, we design the BGM model shown in Figure 1, which includes an embedding layer and a bag representation layer.

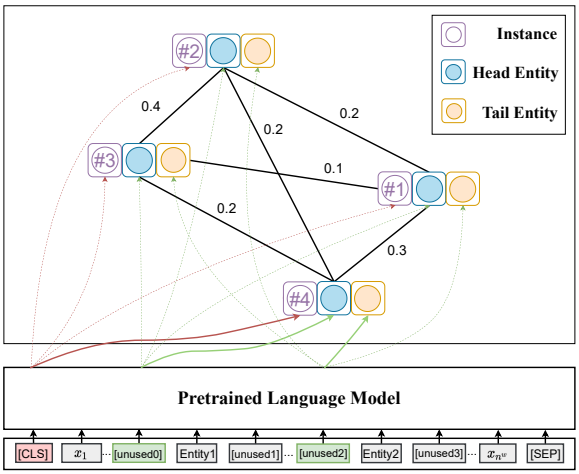

Figure 1: The overview of BGM of the embedding layer and the bag representation layer. Here, the BGM encodes a bag with four instances. For each instance, $[CLS]$, $[ununsed0]$ and $[unused2]$ are used.

3.1 Embedding Layer

To represent each instance in a bag, we use a BERT-based PLM as our embedding layer. Inspired by the role of $[CLS]$ in BERT, we adopt the $[unused*]$ to represent two involved entities. This effectively addresses the multi-word entities representation. Specifically, for each instance, the token sequence $\{[CLS], x_1, x_2, [unused0], e_h, [unused1], \cdots, [unused2], e_t, [unused3], x_{n^w}, [SEP]\}$ is fed into the BERT encoder. Then, three hidden states, i.e., $\{H_c, H_{u0}, H_{u2}\} \in \mathbb{R}^d$ which correspond to the tokens $[CLS]$, $[unused0]$, and $[unused2]$ are obtained for representing each instance and the corresponding two entities.

3.2 Bag Representation Layer

With the embedding layer, we then construct the bag graph $\mathcal{G} = \{\mathcal{V}, \mathcal{A}\}$ for each bag. The set of nodes $\mathcal{V}$ are initialized by concatenating the representations of instance and two entities, i.e., $[H_c; H_{u0}; H_{u2}]$. Thus, the dimension of node d_h equals $3d$. Moreover, the adjacency matrix $\mathcal{A} \in \mathbb{R}^{n^s \times n^s}$ is generated as,

$$\mathcal{A} = \text{softmax}\left(\frac{QW^Q \times \left(KW^K\right)^{\mathrm{T}}}{\sqrt{d_h}}\right) \quad (1)$$

where Q and K are both the concatenated representations of instances. Matrices W^Q and $W^K \in \mathbb{R}^{d_h \times d_h}$ are trainable parameters.

Our GCN is updated by applying the classical method (Kipf and Welling, 2017). Suppose that $H^{(l)}$ denotes the input matrix of nodes of the l-th layer. The computation of next layer is given as,

$$H^{(l+1)} = \rho\left(\widetilde{\mathcal{A}} H^{(l)} W^{(l)} + b^{(l)}\right) \quad (2)$$

where $W^{(l)} \in R^{d_h \times d_h}$ is the trainable weight matrix, $b^{(l)}$ is the bias vector and ρ is an activation function (e.g., ReLU). To maintain the instance's original semantics, a self-connection is added to each node, i.e., $\widetilde{\mathcal{A}} = \mathcal{A} + I$; I is an identity matrix. Thus, for the last graph layer L, the node representation $H^{(L)}$ is obtained, which is used for constructing a bag representation.

3.3 Relation Prediction

We apply an average pooling on the bag representation $H^{(L)}$. A linear layer followed by a softmax layer is used to predict the relation $\hat{r}$ as follows,

$$\hat{r} = \text{softmax}\left(\text{MLP}(\text{AvgPooling}(H^{(L)}))\right). \quad (3)$$

The BGM is trained using a classical cross-entropy loss with gradient descent optimization.

4 Experiment

4.1 Dataset and Metric

Two benchmark datasets[2] are used. **NYT10** (Riedel et al., 2010) is generated by aligning the FreeBase's instances with NYT News Corpus. This dataset contains 39,528 entities and supports 53 types of relations (NA for no relation is included). Following previous works, NYT10 is split into 466,876/55,167/172,448 instances for training/validation/testing.

GDS (Jat et al., 2017) is recently built using Google RE corpus[3]. To meet MIL's *expressed-at-least-once* assumption, at least one sentence for each bag is correctly labelled which makes automatically evaluation more credible. The GDS dataset is officially split into 11,297/1,864/5,663 for training/validation/testing.

Metrics. We adopt four metrics, including precision-recall curve (**PR**), area under curve (**AUC**), Precision@N (**P@N**) values, and Micro-F1 score (**F1**) for evaluation.

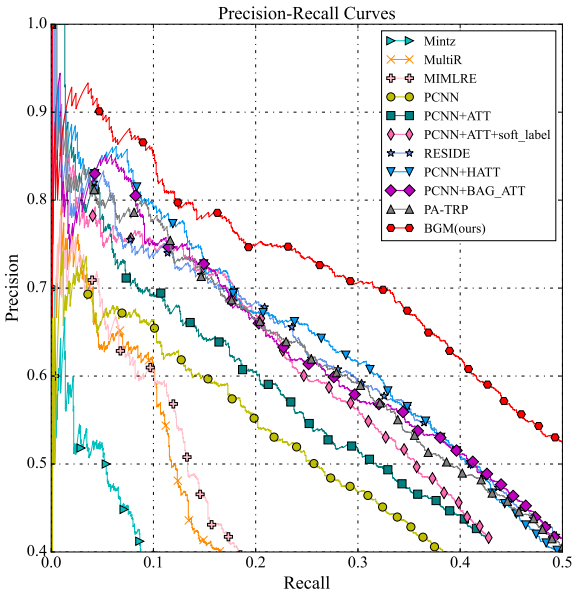

Figure 2: PR curves comparison on NYT10.

4.2 Implementations Detail

In our experiments, the BERT-base-uncased English version model[4] is used. The maximum of input sequence length of BERT is set to 120 and the hidden size is 768. The GCN has two layers. In addition, we apply drop-out rate p of 0.3 to GCN and 0.5 to all linear layers. The Adam optimizer (Kingma and Ba, 2015) is adopted to train the model with a learning rate of 2×10^{-5} and a batch size of 32 for up to 3 epochs. All experiments are conducted on an NVIDIA V100 GPU.

4.3 Baseline

We compare our BGM with the following baseline methods, including Mintz (Mintz et al., 2009), MultiR (Hoffmann et al., 2011), MIMLRE (Surdeanu et al., 2012), PCNN (Zeng et al., 2015), PCNN+ATT (Lin et al., 2016), PCNN+ATT+soft_label (Liu et al., 2017), BGWA (Jat et al., 2017), CNN+RL (Feng et al., 2018), DSGAN (Qin et al., 2018), RESIDE (Vashishth et al., 2018), PCNN+HATT (Han et al., 2018), PCNN+BAG_ATT (Ye and Ling, 2019), DISTRE (Alt et al., 2019), PA-TMR (Kuang et al., 2020), ToHRE (Yu et al., 2020), PA-TRP (Cao et al., 2021), SRKBP (Christopoulou et al., 2021), REDSandT (Christou and Tsoumakas,

[2]https://github.com/thunlp/OpenNRE

[3]https://research.googleblog.com/2013/04/50000-lessons-on-how-to-read-relation.html

[4]https://github.com/huggingface/transformers

Method	P@N							AUC
	100	200	300	500	1000	2000	MEAN	
PCNN+ATT (Lin et al., 2016)	73.0	68.0	67.3	63.6	53.3	40.0	60.9	34.1
BGWA (Jat et al., 2017)	76.0	74.0	-	-	-	-	-	36.7
CNN+RL (Feng et al., 2018)	79.0	73.0	-	-	-	-	-	37.4
DSGAN (Qin et al., 2018)	80.0	78.0	-	-	-	-	-	38.0
RESIDE (Vashishth et al., 2018)	81.8	75.4	74.3	69.7	59.3	45.0	67.6	41.5
PCNN+HATT (Han et al., 2018)	82.0	79.5	75.3	67.0	57.7	41.9	67.2	42.0
PCNN+BAG_ATT (Ye and Ling, 2019)	**91.8**	83.0	76.3	70.2	52.0	34.2	67.9	42.2
PA-TMR (Kuang et al., 2020)	83.0	79.0	-	-	-	-	-	43.7
ToHRE (Yu et al., 2020)	91.5	82.9	79.6	**74.8**	63.3	48.9	73.5	-
PA-TRP (Cao et al., 2021)	87.0	79.5	77.3	68.6	59.0	44.6	67.9	41.5
SRKBP (Christopoulou et al., 2021)	83.0	75.5	73.0	-	-	-	-	42.9
PSAN-RE (Shang et al., 2022)	79.2	71.1	66.8	65.9	60.4	48.1	65.2	43.8
DISTRE (Alt et al., 2019) ♯	68.0	67.0	65.3	65.0	60.2	47.9	62.2	42.2
REDSandT (Christou and Tsoumakas, 2021) ♯	78.0	-	73.0	67.6	-	-	-	42.9
CIL (Chen et al., 2021) ♯	90.1	86.1	**81.8**	-	-	-	-	50.8
Our BGM	90.3	**86.5**	80.0	74.6	**67.5**	**50.7**	**74.9**	**51.5**

Table 1: Comparison results on NYT10. The symbol ♯ denotes the PLMs-based methods.

2021), CIL (Chen et al., 2021), and PSAN-RE (Shang et al., 2022). Note that the three methods, including DISTRE, REDSandT, and CIL are PLMs-based.

4.4 Results and Analysis

On NYT10. Figure 2 plots the **PR curves** on the NYT10 dataset. We observe that compared with baseline models, our proposed BGM achieves better performance by a large margin. It means that our model could make full use of the training data and capture the critical information in noisy data.

Table 1 reports the **P@N** and **AUC** values on NYT10 dataset. We notice that our model achieves the best performance and outperforms the other baseline models in almost all metrics. Compared with the strong competitor CIL, our proposed model improves P@100 and P@200 by 0.2% and 0.5% respectively and improves AUC score by 1.4%. In addition, compared with ToHRE on P@MEAN, our model improves the score by 1.9%, i.e., 73.5 → 74.9 .

On GDS. Table 2 reports the results on GDS dataset. Our model achieves a comparable performance on the GDS dataset, in terms of P@100, P@200 and AUC, i.e., 100.0, 98.0 and 89.2. Our BGM achieves better performance compared with these baseline models.

Method	P@100	P@200	AUC
PCNN+ATT (Lin et al., 2016)	94.0	93.0	80.3
BGWA (Jat et al., 2017)	99.0	98.0	81.5
CNN+RL (Feng et al., 2018)	**100.0**	96.0	85.5
DSGAN (Qin et al., 2018)	99.0	97.0	84.5
RESIDE (Vashishth et al., 2018)	**100.0**	97.5	89.1
PCNN+HATT (Han et al., 2018)	99.0	97.0	85.4
PA-TMR (Kuang et al., 2020)	**100.0**	98.0	86.5
PA-TRP (Cao et al., 2021)	**100.0**	98.0	87.3
PSAN-RE (Shang et al., 2022)	97.0	**98.5**	**91.1**
Our BGM	**100.0**	98.0	89.2

Table 2: Comparison results on GDS.

Method	AUC	P@M	F1
Our BGM	**51.5**	**74.9**	**52.4**
BGM *w/o* GCN	46.7 (4.8↓)	68.3 (6.6↓)	51.6 (0.8↓)
BGM *w/o* EntCon	46.9 (4.6↓)	68.0 (6.9↓)	51.9 (0.5↓)

Table 3: Ablation study of our BGM on NYT10.

PLM	AUC	P@M	F1
Bert-based-uncased	51.5	**74.9**	52.4
Bert-based-cased	49.7	71.0	53.7
Bert-large-uncased	52.9	72.4	56.3
Distilbert-base-uncased	49.5	71.5	50.9
Xlnet-base-cased	47.5	67.1	53.1
Albert-based-v2	48.3	72.1	50.3
Roberta-base	48.8	68.5	52.7
Roberta-large	**53.2**	74.5	**57.1**

Table 4: Comparison of PLMs in BGM on NYT10.

4.5 Ablation Study

In Table 3, **BGM *w/o* GCN** degenerates our full BGM to a naïve BERT. BGM *w/o* GCN has a comparable performance. However, it fails to interrelate the instances for encoding bag representations, the overall performance drops, i.e., AUC ($51.5 \rightarrow 46.7$), P@Mean ($74.9 \rightarrow 68.3$) and F1 ($52.4 \rightarrow 51.6$). **BGM *w/o* EntCon** only uses the [CLS] token but not entity-aware instance representations. The AUC, P@M and F1 decrease. This shows that the entity information is essential for relation extraction. Moreover, to investigate the impact of various **PLMs** in our BGM, we replace the basic BERT-based with other representative PLMs. In Table 4, we observe that BGM with various PLMs as an embedding layer can achieve competitive performance. Due to larger parameters, Roberta-large achieves the best performance.

4.6 Case Study

We use two bags shown in Table 5 for case study on three methods, including BGM, BGM *w/o* GCN and BGM *w/o* EntCon. For **#1** bag, our BGM gives the relation, */location/country/administrative_divisions*. The other variants give the wrong relation, */location/location/contains*. The reason is that BGM *w/o* EntCon could not use the representations of key phrases "in the state of" in S3. Besides, BGM *w/o* GCN could not utilize the instance correlations in the bag. The captured information in S3 is not shared well with the other instances.

For **#2** bag, it expresses the relation */people/person/place_of_birth*. Our BGM *w/o* EntCon predicts the wrong relation, */people/person/place_lived*. In contrast, BGM and BGM *w/o* GCN can identify the golden truth. With the entity-aware instance representation, they capture the contextual information of entities embedded in "was born in". This helps the model focus on entities and capture the relation more effectively.

Bag	Instance
# 1	S1: ...she gazed at the work before her: ... and the landscape of the [Jalisco] region of [Mexico] .
	S2: ..., left his small ranch in the [Jalisco] region of [Mexico] for work in the promised land of the united states .
	S3: ...Italian real estate magnates who relocated to [Mexico] and built a series of sumptuous properties in the state of [Jalisco] that made it a magnet for the super-rich .
# 2	S1: ..., like Freddy Rodriguez's tribute to [Sammy Sosa] , who was born in the [Dominican Republic] , with a glass....

Table 5: Two bags from NYT10 for case study.

5 Discussion

Our BGM performs its calculation on the entire graph. The GCN layer calculates weights adopting the method of self-attention. In other words, the node features of the entire graph are updated after one calculation, and the learned parameters are not heavily related to the graph structure. Compared with other attention-based methods, we do not follow their selective attention with the bag's target relation. Our BGM uses a self-attention mechanism to capture the correlation among instances which are taken as the nodes of the graph. Therefore, GCN combined with self-attention is one of effective ways for the setting of DS-RE.

6 Conclusion

In this paper, we propose a simple but effective model, a.k.a. BGM based on PLMs and GCN for DS-RE. Each instance is represented using a BERT-based pre-trained language model. To capture the instance correlations, GCN for multiple instances within a bag is used. With this type of bag representation, a cross-entropy loss is applied for predicting the relation between entities. Extensive experiments on two benchmark datasets show the superior performance. In our future, we will investigate the hidden theory in-depth for better explainability of our BGM model. In addition, extending the BGM to dealing with the case of a single-instance bag is an interesting problem.

Acknowledgements

This work was supported in part by the National Key Research and Development Program of China under Grant 2019YFF0303300 and Subject II under Grant 2019YFF0303302, and in part by the National Natural Science Foundation of China under Grant 62076032. The authors would also like to thank the editor and anonymous reviewers for their valuable comments on improving the final version of this paper.

References

Chris Alberti, Kenton Lee, and Michael Collins. 2019. A bert baseline for the natural questions. *arXiv preprint arXiv:1901.08634*.

Christoph Alt, Marc Hübner, and Leonhard Hennig. 2019. Fine-tuning pre-trained transformer language models to distantly supervised relation extraction. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 1388–1398, Florence, Italy. Association for Computational Linguistics.

Yixin Cao, Jun Kuang, Ming Gao, Aoying Zhou, Yonggang Wen, and Tat-Seng Chua. 2021. Learning relation prototype from unlabeled texts for long-tail relation extraction. *IEEE Transactions on Knowledge and Data Engineering*, pages 1–1.

Tao Chen, Haizhou Shi, Siliang Tang, Zhigang Chen, Fei Wu, and Yueting Zhuang. 2021. CIL: Contrastive instance learning framework for distantly supervised relation extraction. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 6191–6200, Online. Association for Computational Linguistics.

Fenia Christopoulou, Makoto Miwa, and Sophia Ananiadou. 2021. Distantly supervised relation extraction with sentence reconstruction and knowledge base priors. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 11–26, Online. Association for Computational Linguistics.

Despina Christou and Grigorios Tsoumakas. 2021. Improving distantly-supervised relation extraction through bert-based label and instance embeddings. *IEEE Access*, 9:62574–62582.

Jun Feng, Minlie Huang, Li Zhao, Yang Yang, and Xiaoyan Zhu. 2018. Reinforcement learning for relation classification from noisy data. In *Proceedings of the AAAI conference on Artificial Intelligence*, volume 32.

Xu Han, Pengfei Yu, Zhiyuan Liu, Maosong Sun, and Peng Li. 2018. Hierarchical relation extraction with coarse-to-fine grained attention. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 2236–2245, Brussels, Belgium. Association for Computational Linguistics.

Raphael Hoffmann, Congle Zhang, Xiao Ling, Luke Zettlemoyer, and Daniel S. Weld. 2011. Knowledge-based weak supervision for information extraction of overlapping relations. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, pages 541–550, Portland, Oregon, USA. Association for Computational Linguistics.

Sharmistha Jat, Siddhesh Khandelwal, and Partha P. Talukdar. 2017. Improving distantly supervised relation extraction using word and entity based attention. In *6th Workshop on Automated Knowledge Base Construction, AKBC@NIPS 2017, Long Beach, California, USA, December 8, 2017*. OpenReview.net.

Diederik P. Kingma and Jimmy Ba. 2015. Adam: A method for stochastic optimization. In *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*.

Thomas N. Kipf and Max Welling. 2017. Semi-supervised classification with graph convolutional networks. In *5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Conference Track Proceedings*. OpenReview.net.

Jun Kuang, Yixin Cao, Jianbing Zheng, Xiangnan He, Ming Gao, and Aoying Zhou. 2020. Improving neural relation extraction with implicit mutual relations. In *36th IEEE International Conference on Data Engineering, ICDE 2020, Dallas, TX, USA, April 20-24, 2020*, pages 1021–1032. IEEE.

Zhengyan Li, Yicheng Zou, Chong Zhang, Qi Zhang, and Zhongyu Wei. 2021. Learning implicit sentiment in aspect-based sentiment analysis with supervised contrastive pre-training. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 246–256, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Yankai Lin, Shiqi Shen, Zhiyuan Liu, Huanbo Luan, and Maosong Sun. 2016. Neural relation extraction with selective attention over instances. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 2124–2133, Berlin, Germany. Association for Computational Linguistics.

Tianyu Liu, Kexiang Wang, Baobao Chang, and Zhifang Sui. 2017. A soft-label method for noise-tolerant distantly supervised relation extraction. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 1790–1795, Copenhagen, Denmark. Association for Computational Linguistics.

Mike Mintz, Steven Bills, Rion Snow, and Daniel Jurafsky. 2009. Distant supervision for relation extraction without labeled data. In *Proceedings of the Joint Conference of the 47th Annual Meeting of the ACL and the 4th International Joint Conference on Natural Language Processing of the AFNLP*, pages 1003–1011, Suntec, Singapore. Association for Computational Linguistics.

Pengda Qin, Weiran Xu, and William Yang Wang. 2018. DSGAN: Generative adversarial training for distant supervision relation extraction. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*,

pages 496–505, Melbourne, Australia. Association for Computational Linguistics.

Sebastian Riedel, Limin Yao, and Andrew McCallum. 2010. Modeling relations and their mentions without labeled text. In *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*, pages 148–163. Springer.

Yu-Ming Shang, Heyan Huang, Xin Sun, Wei Wei, and Xian-Ling Mao. 2022. A pattern-aware self-attention network for distant supervised relation extraction. *Information Sciences*, 584:269–279.

Mihai Surdeanu, Julie Tibshirani, Ramesh Nallapati, and Christopher D. Manning. 2012. Multi-instance multi-label learning for relation extraction. In *Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning*, pages 455–465, Jeju Island, Korea. Association for Computational Linguistics.

Hongyin Tang, Xingwu Sun, Beihong Jin, Jingang Wang, Fuzheng Zhang, and Wei Wu. 2021. Improving document representations by generating pseudo query embeddings for dense retrieval. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing, ACL/IJCNLP 2021, (Volume 1: Long Papers), Virtual Event, August 1-6, 2021*, pages 5054–5064. Association for Computational Linguistics.

Shikhar Vashishth, Rishabh Joshi, Sai Suman Prayaga, Chiranjib Bhattacharyya, and Partha Talukdar. 2018. RESIDE: Improving distantly-supervised neural relation extraction using side information. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 1257–1266, Brussels, Belgium. Association for Computational Linguistics.

Yijun Wang, Changzhi Sun, Yuanbin Wu, Hao Zhou, Lei Li, and Junchi Yan. 2021. UniRE: A unified label space for entity relation extraction. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics*. Association for Computational Linguistics.

Zhi-Xiu Ye and Zhen-Hua Ling. 2019. Distant supervision relation extraction with intra-bag and inter-bag attentions. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 2810–2819, Minneapolis, Minnesota. Association for Computational Linguistics.

Erxin Yu, Wenjuan Han, Yuan Tian, and Yi Chang. 2020. ToHRE: A top-down classification strategy with hierarchical bag representation for distantly supervised relation extraction. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 1665–1676, Barcelona, Spain (Online). International Committee on Computational Linguistics.

Daojian Zeng, Kang Liu, Yubo Chen, and Jun Zhao. 2015. Distant supervision for relation extraction via piecewise convolutional neural networks. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 1753–1762, Lisbon, Portugal. Association for Computational Linguistics.

Augmenting Legal Judgment Prediction with Contrastive Case Relations

Dugang Liu[1,2], Weihao Du[1,2], Lei Li[3], Weike Pan[1,2,*], Zhong Ming[1,2,*]
[1]Shenzhen University, Shenzhen, China
[2]Guangdong Laboratory of Artificial Intelligence and Digital Economy (SZ), Shenzhen, China
[3]Hong Kong Baptist University, Hong Kong, China
dugang.ldg@gmail.com, {panweike,mingz}@szu.edu.cn

Abstract

Existing legal judgment prediction methods usually only consider one single case fact description as input, which may not fully utilize the information in the data such as case relations and frequency. In this paper, we propose a new perspective that introduces some contrastive case relations to construct case triples as input, and a corresponding judgment prediction framework with case triples modeling (CTM). Our CTM can more effectively utilize beneficial information to refine the encoding and decoding processes through three customized modules, including the case triple module, the relational attention module, and the category decoder module. Finally, we conduct extensive experiments on two public datasets to verify the effectiveness of our CTM, including overall evaluation, compatibility analysis, ablation studies, analysis of gain source and visualization of case representations.

1 Introduction

As an important component of legal intelligence in civil law systems, legal judgment prediction (LJP) has received a lot of attention and research in recent years (Chalkidis et al., 2019; Zhong et al., 2020). Given a case fact description, LJP usually includes three sub-tasks, i.e., law article prediction, charge prediction and terms of penalty prediction for this case (Xiao et al., 2018), and an example of LJP is shown on the left side of Figure 1. As an auxiliary tool to serve legal practitioners and people without professional knowledge in law, a more accurate method for LJP is necessary.

The existing legal judgment prediction methods mainly include two lines of single-task modeling and multi-task modeling. The former usually focuses on targeted modeling of a certain sub-task, such as introducing some more advanced network architectures (Chen et al., 2019a; Le et al., 2020) or more sources of information (Luo et al., 2017; Hu et al., 2018; Chen et al., 2019b). The latter takes multiple sub-tasks as a whole and uses a multi-task learning (MTL) framework for unified modeling. The most representative methods in this line aim to design different decoding structures, including MTL (Zhong et al., 2018) that ignores the inter-task dependency, TopJudge (Zhong et al., 2018) that considers unidirectional topological dependency among sub-tasks, and MPBFN (Yang et al., 2019) that considers bidirectional topological dependency. In this paper, we focus on the line of multi-task learning because it is more aligned with practical applications.

Although the existing methods have shown promising results, as shown on the left side of Figure 1, most of them only consider the fact description of one single case as input when modeling. This form of modeling ignores the full utilization of the beneficial information contained in the data, such as the case relation and frequency information that might provide constraints for modeling. We believe this may have an adverse effect on the model and cause a performance bottleneck, such as cases with low-frequency law articles or charges suffer from insufficient training. As an example, we show in Figure 2 the accuracy of MPBFN on CAIL-small (Xiao et al., 2018) for law articles and charges of different frequencies. We can find that the accuracy drops significantly with decreasing frequency.

To more effectively utilize the beneficial information contained in the data, in this paper, we propose a new perspective that introduces some contrastive case relations to construct case triples as the input of the model. Specifically, we sample some similar and dissimilar cases for a current case through some carefully designed contrasting case relations, where these auxiliary cases will be beneficial to improve the performance of the model. An example of this new form of modeling is shown on

*Co-corresponding authors

Proceedings of the 29th International Conference on Computational Linguistics, pages 2658–2667
October 12–17, 2022.

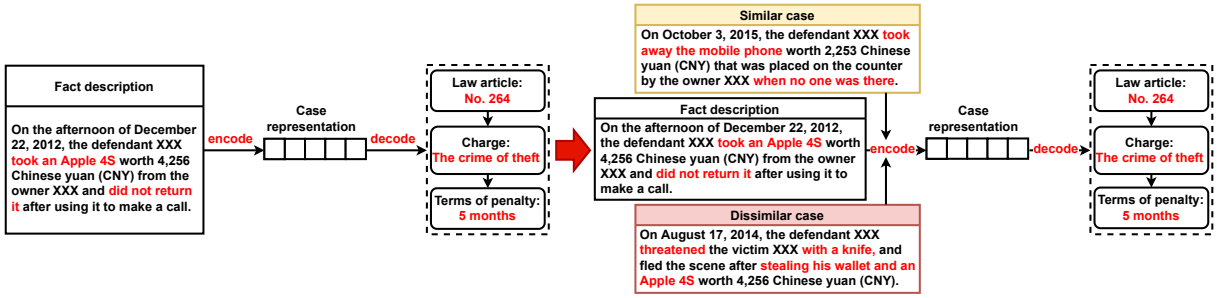

Figure 1: On the left is an illustrative example of legal judgment prediction (LJP), including a case fact description and the corresponding three prediction sub-tasks. On the right is an example of the proposed new form of modeling, where some contrastive case relations are introduced to construct case triples as input. Note that in this paper we focus on prediction of law articles and charges, since that of terms of penalty is known of high difficulty and variance.

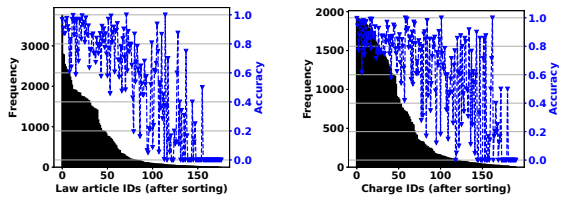

(a) (b)

Figure 2: The prediction accuracy of MPBFN on CAIL-small for each law article and charge. Note that the IDs on the horizontal axis have been sorted in a descending order of frequency.

the right side of Figure 1, and the importance of different phrases may be more accurately identified through this case triple.

We then propose a corresponding judgment prediction framework with case triple modeling (CTM) to mine information from the constructed case triples for improving the model. Specifically, our CTM adds three customized modules to the traditional encoder and decoder: 1) a case triple module samples two similar cases and one dissimilar case for an input case based on case labels and frequency information to form two case triples; 2) a relational attention module imposes a relational constraint on the obtained case triples to refine the encoding process; and 3) a category decoder module acts as a switch to select a corresponding decoding branch for a high-frequency or low-frequency case to further refine the decoding process.

It is intuitive that our CTM does not depend on a specific encoder or decoder. This means that our CTM can be easily integrated with some existing legal judgment prediction methods, and we will demonstrate its good compatibility in the experiments by combining our CTM with different encoder and decoder structures. In addition to this, we conduct other empirical studies on two public datasets to verify the effectiveness of our CTM, including overall performance evaluation, ablation studies, fine-grained performance evaluation and case representation analysis.

2 Related Work

In this section, we briefly review some related works on two research topics, including legal judgment prediction and case relations modeling.

Legal Judgment Prediction. Legal judgment prediction can be mainly summarized into two research lines. The first line focuses on the targeted modeling of a specific sub-task from the perspective of network architectures (Chen et al., 2019a; He et al., 2019; Le et al., 2020), available information sources (Luo et al., 2017; Hu et al., 2018; Chen et al., 2019b), and interpretability of the models (Jiang et al., 2018). The second line considers multiple sub-tasks as a whole and uses a multi-task learning framework for case modeling. The most representative methods are MTL (Zhong et al., 2018), TopJudge (Zhong et al., 2018) and MPBFN (Yang et al., 2019), in which three different decoding structures are considered respectively. Some recent works have designed some more sophisticated architectures based on them, especially in combination with some graph learning techniques and large-scale pre-trained models (Xu et al., 2020; Chen et al., 2020; Dong and Niu, 2021; Yue et al., 2021). Note that since the terms of penalty prediction is usually of higher difficulty and variance than the other two sub-tasks, we focus on law article prediction and charge prediction similar to (Bao et al., 2019; Chen et al., 2021).

Case Relations Modeling. The idea of case rela-

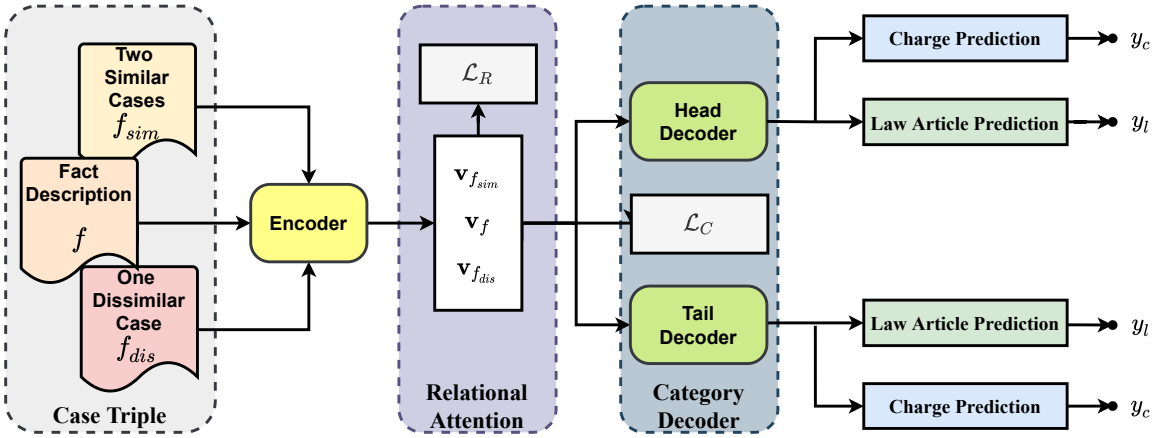

Figure 3: The architecture of a judgment prediction framework with our case triple modeling (CTM).

tions modeling is mainly applied to similar case matching (SCM) tasks in some recent studies on legal intelligence (Xiao et al., 2019; Peng et al., 2020; Hong et al., 2020). Unlike legal judgment prediction, this task is given a set of manually labeled case triples as training samples, where each triple contains two similar cases and one dissimilar case, and the goal is to learn a model that can identify those two similar ones. This task can be further relaxed to find some similar cases for a current case (Tang and Clematide, 2021; Ostendorff et al., 2021), which is important in the common law system. To the best of our knowledge, our work is the first to introduce a case triple structure to legal judgment prediction based on some case relations.

3 The Proposed Framework

3.1 Architecture

The judgment prediction framework with case triple modeling, or CTM for short, is shown in Figure 3. Note that similar to most works, we consider each case with only one law article label and one charge label for simplicity. Given a current case $f = [s; y_l, y_c, y_a]$, where $s = \{s_1, s_2, \ldots, s_n\}$ represents the fact description composed of sentences, y_l is the law article label, y_c is the charge label, and $y_a \in \{0, 1\}$ is a category label indicating whether the case is a high-frequency case or not. Note that a more specific description of high-frequency cases can be found in the case triple module in Sec 3.2. The case triple module samples two similar cases and one dissimilar case to construct the case triple (f, f_{sim}, f_{dis}) based on some contrastive case relations. Then, a constraint is imposed on the encoded representations corresponding to the case triple (i.e., $\mathbf{v}_f$, $\mathbf{v}_{f_{sim}}$ and $\mathbf{v}_{f_{dis}}$) in the relational attention module to refine the encoding process.

In the category decoder module, we first impose a classification constraint on the category label y_a to inform the model to which category the current encoded representation belongs, and then switch the decoder of the corresponding category branch to refine the decoding process. Finally, the model obtains the predicted label of each sub-task and compares it with the respective true label. The final optimization objective function of our CTM can be expressed as follows,

$$\min_{\theta} \mathcal{L}_{CTM} = \mathcal{L}_M + \mathcal{L}_R + \mathcal{L}_C + \lambda \|\theta\|, \quad (1)$$

where $\mathcal{L}_M$, $\mathcal{L}_R$, and $\mathcal{L}_C$ denote the prediction loss for multi-task learning, the constraint loss for the relational attention module and the loss for the category decoder module, respectively, and λ and $\|\theta\|$ are the tradeoff parameter and the regularization terms.

3.2 Training

In this section, we describe each module in detail based on the training process.

The Case Triple Module. We propose a concept called contrastive case relation that considers both labels and frequency information for constructing some case triples. Specifically, we use a threshold ϕ to pre-divide the labels of the law articles (and charges) into two sets of low-frequency $\mathcal{A}_l$ (or $\mathcal{A}_c$) and high-frequency $\mathcal{B}_l$ (or $\mathcal{B}_c$), where $\mathcal{A}_l$ (or $\mathcal{A}_c$) contains the labels with the lowest ϕ frequency and $\mathcal{B}_l$ (or $\mathcal{B}_c$) contains the remaining labels. For a case f, a similar case f^l_{sim} (or f^c_{sim}) on the law articles (or charges) is sampled from the candidate cases with the same law article (or charges) label. Then, a dissimilar case f_{dis} on the charges is sampled from

the candidate cases with different charge labels and the corresponding labels do not belong to $\mathcal{A}_c$. The additional constraint that the labels do not belong to $\mathcal{A}_c$ help cases with low-frequency charge labels to be more fully trained based on a large number of opposite references. Since the law articles can be regarded as the leaf nodes of charges in civil law systems, i.e., different charge labels must have different law article labels, we regard this dissimilar case on the charges as a shared dissimilar case, i.e., it is also regarded as a dissimilar case on the law article. This can reduce the number of cases that need to be encoded in a subsequent fact description encoder module to reduce the size of the model. Finally, we can obtain two types of case triples $\left(f, f^l_{sim}, f_{dis}\right)$ and $\left(f, f^c_{sim}, f_{dis}\right)$ for f.

Considering that when f is a high-frequency case, i.e., $y_l \in \mathcal{B}_l$ or $y_c \in \mathcal{B}_c$, the above two case triples can enhance the distinction between the high-frequency cases. When f is a low-frequency case, i.e., $y_l \in \mathcal{A}_l$ or $y_c \in \mathcal{A}_c$, these triples can improve its insufficient training and enhance the distinction between it and the high-frequency cases by introducing a large number of high-frequency cases as opposite references. For ease of understanding, we give an example of the sampling process in Figure 4.

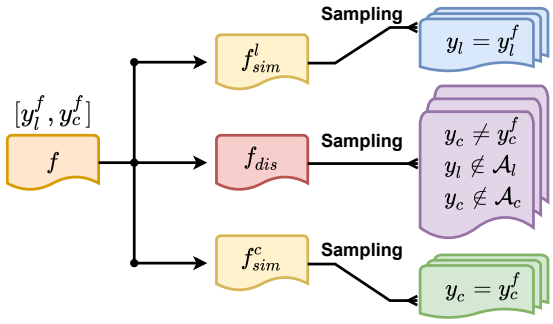

Figure 4: The schematic diagram of a sampling process, where the law article and charge labels of a case f are assumed to be y_l^f and y_c^f, respectively.

The Fact Description Encoder Module. After constructing the case triples, we need to encode the fact description of each case. Next, we use hierarchical Bi-GRU (Yang et al., 2016) as an example encoder[1], which has also been adopted in some recent works (Long et al., 2019; Xu et al., 2020; Ma et al., 2021). Specifically, let each sentence in the fact description be represented as $s_i = [w_{i,1}, w_{i,2}, \ldots, w_{i,m}]$, where $w_{i,j}$ represents the j-th word of sentence s_i, and m denotes the number of words, a word-level Bi-GRU will act on each sentence and output a corresponding representation (Yang et al., 2016),

$$\mathbf{h}_{i,j} = [\overrightarrow{\mathrm{GRU}}(\mathbf{w}_{i,j}), \overleftarrow{\mathrm{GRU}}(\mathbf{w}_{i,j})] \in \mathbb{R}^{d_w},$$
$$\alpha_{i,j} = \frac{\exp(\tanh(\mathbf{W}_w\mathbf{h}_{i,j} + \mathbf{b}_w)^\top \mathbf{u}_w)}{\sum_j \exp(\tanh(\mathbf{W}_w\mathbf{h}_{i,j} + \mathbf{b}_w)^\top \mathbf{u}_w)},$$
$$\mathbf{v}_{s_i} = \sum_{j=1}^{m} \alpha_{i,j}\mathbf{h}_{i,j},$$

where $\mathbf{w}_{i,j}$ represents an embedding vector of word $w_{i,j}$, $\mathbf{W}_w \in \mathbb{R}^{d_w \times d_w}$ is a weight matrix, $\mathbf{b}_w \in \mathbb{R}^{d_w}$ is a bias vector and $\mathbf{u}_w \in \mathbb{R}^{d_w}$ is a trainable context vector. Then, a sentence-level Bi-GRU will act on the representation sequence of the sentences, i.e., $[\mathbf{v}_{s_1}, \mathbf{v}_{s_2}, \ldots, \mathbf{v}_{s_n}]$, to obtain the encoded representation of case f (Yang et al., 2016),

$$\mathbf{h}_i = [\overrightarrow{\mathrm{GRU}}(\mathbf{v}_{s_i}), \overleftarrow{\mathrm{GRU}}(\mathbf{v}_{s_i})] \in \mathbb{R}^{d_s},$$
$$\alpha_i = \frac{\exp(\tanh(\mathbf{W}_s\mathbf{h}_i + \mathbf{b}_s)^\top \mathbf{u}_s)}{\sum_i \exp(\tanh(\mathbf{W}_s\mathbf{h}_i + \mathbf{b}_s)^\top \mathbf{u}_s)},$$
$$\mathbf{v}_f = \sum_{i=1}^{n} \alpha_i\mathbf{h}_i,$$

where the meaning of $\mathbf{W}_s$, $\mathbf{b}_s$ and $\mathbf{u}_s$ are similar to that of $\mathbf{W}_w$, $\mathbf{b}_w$ and $\mathbf{u}_w$, respectively. Similarly, we can also obtain the encoded representations of other cases in the case triples, i.e., $\mathbf{v}^l_{f_{sim}}$, $\mathbf{v}^c_{f_{sim}}$ and $\mathbf{v}_{f_{dis}}$.

The Relational Attention Module. To refine the encoding process by extracting beneficial case relation information from case triples, we first calculate the attention vectors between case f and its similar and dissimilar cases in the representation space, as well as the anchor attention to itself,

$$\mathbf{r}^l = \mathbf{W}^3_l(\sigma(\mathbf{W}^1_l\mathbf{v}_f + (\mathbf{W}^2_l\mathbf{v}_f + \mathbf{b}^2_l))),$$
$$\mathbf{r}^l_{sim} = \mathbf{W}^3_l(\sigma(\mathbf{W}^1_l\mathbf{v}_f + (\mathbf{W}^2_l\mathbf{v}^l_{f_{sim}} + \mathbf{b}^2_l))),$$
$$\mathbf{r}^l_{dis} = \mathbf{W}^3_l(\sigma(\mathbf{W}^1_l\mathbf{v}_f + (\mathbf{W}^2_l\mathbf{v}_{f_{dis}} + \mathbf{b}^2_l))),$$
$$\mathbf{r}^c = \mathbf{W}^3_c(\sigma(\mathbf{W}^1_c\mathbf{v}_f + (\mathbf{W}^2_c\mathbf{v}_f + \mathbf{b}^2_c))),$$
$$\mathbf{r}^c_{sim} = \mathbf{W}^3_c(\sigma(\mathbf{W}^1_c\mathbf{v}_f + (\mathbf{W}^2_c\mathbf{v}^c_{f_{sim}} + \mathbf{b}^2_c))),$$
$$\mathbf{r}^c_{dis} = \mathbf{W}^3_c(\sigma(\mathbf{W}^1_c\mathbf{v}_f + (\mathbf{W}^2_c\mathbf{v}_{f_{dis}} + \mathbf{b}^2_c))),$$

where $\mathbf{W}^1_l$, $\mathbf{W}^2_l$, $\mathbf{W}^3_l$ and $\mathbf{b}^2_l$ are weight matrices and bias vector for the first triple, the parameters for the second triple are similarly defined, and $\sigma(\cdot)$

[1]Note that the fact description encoder can be any existing encoder, and in the experiment section, we use a variety of encoders to verify the compatibility of CTM.

is the sigmoid activation function. Inspired by supervised contrastive learning (Schroff et al., 2015; Patro and Namboodiri, 2018), we impose a relational constraint on the two triples as an additional optimization objective,

$$\mathcal{L}_R = \max(0, \beta_l + \|\mathbf{r}^l - \mathbf{r}^l_{sim}\|_2^2 - \|\mathbf{r}^l - \mathbf{r}^l_{dis}\|_2^2) + \max(0, \beta_c + \|\mathbf{r}^c - \mathbf{r}^c_{sim}\|_2^2 - \|\mathbf{r}^c - \mathbf{r}^c_{dis}\|_2^2), \tag{2}$$

where β_l and β_c are weight parameters. An intuitive explanation for the relational attention module is that in the attention vector between two cases, a higher attention value means that this dimension plays a greater role in the similarity of the two cases. By imposing the relational constraints in Eq.(2), we can further reduce noise from the attention vector between the current case and the corresponding similar case, which contributes to the similarity between the current case and dissimilar case.

The Category Decoder Module. To further avoid the influence between the high-frequency and the low-frequency cases, we set up a decoder for each of them to refine the decoding process. Since it is difficult for the model to know the category information of the current encoded representation in practice, we first impose a classification constraint to encourage the model to identify the category information more accurately,

$$\mathcal{L}_C = \mathcal{L}\left(\hat{y}_a, y_a\right), \tag{3}$$

where $\hat{y}_a = softmax(\mathbf{W}^2_{ca} * relu(\mathbf{W}^1_{ca}\mathbf{v}_f) + \mathbf{b}^2_{ca})$, $\mathbf{W}^1_{ca}$, $\mathbf{W}^2_{ca}$ and $\mathbf{b}^2_{ca}$ are weight matrices and bias vector. After obtaining the category label of the current case, we select the corresponding branch to decode the encoded representation. Note that the decoders on both branches have the same structure. Next, we use a unidirectional topological dependency structure similar to TopJudge (Zhong et al., 2018) as an example decoder[2]. The decoding process can be described as follows,

$$\begin{bmatrix}\mathbf{h}_l \\ \mathbf{c}_l\end{bmatrix} = LSTMCell\left(\mathbf{v}_f, \begin{bmatrix}\overline{\mathbf{h}}_l \\ \overline{\mathbf{c}}_l\end{bmatrix}\right),$$
$$\begin{bmatrix}\overline{\mathbf{h}}_c \\ \overline{\mathbf{c}}_c\end{bmatrix} = \left(\mathbf{W}_{c,l}\begin{bmatrix}\mathbf{h}_l \\ \mathbf{c}_l\end{bmatrix}\right) + \mathbf{b}_{c,l},$$
$$\begin{bmatrix}\mathbf{h}_c \\ \mathbf{c}_c\end{bmatrix} = LSTMCell\left(\mathbf{v}_f, \begin{bmatrix}\overline{\mathbf{h}}_c \\ \overline{\mathbf{c}}_c\end{bmatrix}\right),$$

where $\overline{\mathbf{h}}_l$ and $\overline{\mathbf{c}}_l$ are the initial hidden state and memory cell of the law article prediction task, $\mathbf{W}_{c,l}$ and $\mathbf{b}_{c,l}$ are the transformation matrix and bias vector that convert the task to charge prediction, and $\mathbf{h}_l$ and $\mathbf{h}_c$ are the decoded representations for these two tasks.

The Judgment Prediction Module. After obtaining the decoded representation of the current case, we use a fully connected layer to obtain the prediction of two different sub-tasks and the loss of multi-task prediction,

$$\begin{aligned}\hat{y}_l &= softmax(\mathbf{W}^l_p\mathbf{h}_l + \mathbf{b}^l_p),\\ \hat{y}_c &= softmax(\mathbf{W}^c_p\mathbf{h}_c + \mathbf{b}^c_p),\\ \mathcal{L}_M &= \mathcal{L}(\hat{y}_l, y_l) + \mathcal{L}(\hat{y}_c, y_c),\end{aligned} \tag{4}$$

where $\mathbf{W}^l_p$, $\mathbf{b}^l_p$ and $\mathbf{W}^c_p$, $\mathbf{b}^c_p$ are the parameters of the respective prediction tasks.

4 Experiments

In this section, we first introduce the experimental setup, and then conduct extensive empirical studies and show the effectiveness of our CTM.

4.1 Experiment Setup

Datasets. We use the two most common benchmark datasets in our experiments, i.e., CAIL-small and CAIL-big (Xiao et al., 2018). Following the settings of most previous works, we remove the cases with fewer than 10 meaningful words, and do not consider cases associated with multiple law articles or charges (Yang et al., 2019; Xu et al., 2020; Yue et al., 2021). Note that for a more comprehensive evaluation, we do not additionally remove the cases that contain law articles or charges with a frequency of lower than 100 as they do. Also, since CAIL-big does not provide a validation set, we divide the original training set for training and verification at a ratio of 9:1. The statistics of the datasets are shown in Table 1.

Table 1: Statistics of the datasets, i.e., CAIL-small and CAIL-big, used in the experiments.

	CAIL-small	CAIL-big
Training Cases	105,059	1,432,826
Validation Cases	14,266	159,372
Test Cases	27,953	186,523
Law Articles	177	181
Charges	191	193

Implementation Details. The baselines considered in the experiments include three existing representative methods, i.e., MTL (Zhong et al., 2018),

[2]Note that in the experiment section, we use decoders with other dependencies to verify the compatibility of CTM.

TopJudge (Zhong et al., 2018), and MPBFN (Yang et al., 2019), and two recent state-of-the-art methods, i.e., LADAN (Xu et al., 2020) and NeurJudge (Yue et al., 2021), where LADAN can be integrated with the first three methods to obtain three variants. All baselines are implemented on TensorFlow 1.15[3], Keras 2.3.1[4] or PyTorch 1.9.1[5] by referring to the source code and parameter settings provided in (Xu et al., 2020; Yue et al., 2021)[6,7]. We use four metrics for performance evaluation, including accuracy (Acc.), macro-recall (MR), macro-precision (MP) and macro-F1 (F1).

After some preliminary experiments, we fix the values of some additional parameters of CTM to reduce the search space, i.e., ϕ, β_l and β_c are set to 60%, 0.5 and 0.3, respectively. For all the methods, we set the maximum number of iterations to 20, and search the best batch size from $\{32, 64, 128\}$ by evaluating the accuracy of the law article prediction on the validation set. We also adopt an early stopping mechanism with a patience of 5 to avoid overfitting to the training set. By setting a random seed from 0 to 7, we run each method for eight times on Intel(R) Xeon(R) E5-2698 with 8 Tesla V100 GPU and report their average results[8].

4.2 Overall Results

If not specified, we use hierarchical Bi-GRU as the default encoder for reporting results, and constrain the fact description of a case to contain up to 15 sentences, where each sentence contains up to 100 words (Yang et al., 2019; Xu et al., 2020). The comparison results between our CTM and the baselines are shown in Table 2. We can see that our CTM consistently outperforms all the baselines on all the metrics across the two datasets of CAIL-small and CAIL-big. Furthermore, by comparing the results of F1, we find that considering more complex decoding dependency structure (i.e., MPBFN) is more prone to misclassification of low-frequency cases, and LADAN and NeurJudge alleviate this problem to some extent by refining the encoding process. Unlike them, our CTM can significantly further improve the model performance by introducing the case triples and customized modules.

[3]https://www.tensorflow.org/

[4]https://keras.io/

[5]https://pytorch.org/

[6]https://github.com/prometheusXN/LADAN

[7]https://github.com/yuelinan/NeurJudge

[8]Note that the source codes are available at https://github.com/dgliu/COLING22_CTM

4.3 Compatibility Analysis

As described in Sec. 3, since our CTM does not depend on a specific encoder and decoder, it can be easily integrated with existing decision prediction methods. We first study the compatibility of our CTM under different encoder choices. In addition to the default hierarchical Bi-GRU, we consider two common encoder choices, i.e., TextCNN (Kim, 2014) and Lawformer (Xiao et al., 2021). For TextCNN, we set the size of each filter to 64 and the filter widths to $(2, 3, 4, 5)$. Since Lawformer is a pre-trained language model with Longformer (Beltagy et al., 2020) for legal long documents, we directly use their provided model[9] for fine-tuning. We compare our CTM variants with different encoders against their respective baselines, i.e., adding the same decoder as our CTM for different encoders. We report the results on our CAIL-small in Table 3, from which we can see that our CTM brings significant improvement in all cases.

Next, we explore the compatibility of our CTM on different decoding structures. In addition to the default unidirectional topological dependency similar to TopJudge, we consider two decoding structures, i.e., ignoring the intra-task dependency similar to MTL and the bidirectional topological dependency similar to MPBFN. We compare our CTM variants with different decoding structures against their respective baselines, i.e., prepending the same encoder as our CTM for different decoding structures. The results on CAIL-small are shown in Table 4, from which we can see that our CTM has a significant advantage in all cases.

4.4 Ablation Studies

Moreover, we conduct ablation studies of our CTM to analyze the role played by each proposed new module. Specifically, we first consider the removal of the category decoder module (denoted as 'w/o CD'), then consider using only the law article-based triple in the case triple module and relational attention module (denoted as 'w/o CD+DS'), and finally remove these two modules (denoted as 'w/o CD+CT+RA'). The results are shown in Table 5. We have the following observation: 1) By comparing 'w/o CD+DS' and 'w/o CD+CT+RA', the introduction of case triples is beneficial to the improvement of the model performance. 2) By comparing 'w/o CD' and w/o CD+DS', multi-case triples are more efficient than single-case triples. 3) By com-

[9]https://huggingface.co/xcjthu/Lawformer

Table 2: Comparison results between our CTM and the baselines, where the significantly best results ($p \leq 0.05$ via two sample t-test) are marked in bold. Note that the accuracy of law article prediction is the main evaluation metric.

Datasets	CAIL-small								CAIL-big							
Tasks	Law Articles (%)				Charges (%)				Law Articles (%)				Charges (%)			
Metrics	Acc.	MR	MP	F1	Acc.	MR	MP	F1	Acc.	MR	MP	F1	Acc.	MR	MP	F1
MTL	77.06	60.76	63.21	59.60	81.72	68.31	71.54	67.57	95.68	61.79	73.47	64.92	95.53	68.53	80.97	71.79
TopJudge	77.35	60.73	62.94	59.63	81.54	68.09	70.22	67.27	95.73	61.78	73.84	64.99	95.53	67.55	80.06	70.90
MPBFN	72.77	50.55	53.25	48.74	75.41	56.15	59.28	55.15	94.13	48.83	60.99	51.26	93.60	50.06	64.01	52.96
MTL-LADAN	77.95	62.62	64.91	61.25	82.84	71.01	73.74	70.24	95.98	64.41	76.01	67.63	95.86	71.15	82.78	74.57
TopJudge-LADAN	78.45	63.65	65.95	62.39	83.19	71.88	74.00	71.06	96.08	64.91	77.07	68.27	95.90	70.81	82.42	74.08
MPBFN-LADAN	75.49	56.26	59.54	55.04	78.75	63.25	66.26	62.46	95.16	54.44	66.31	56.93	94.64	56.09	70.93	59.18
NeurJudge	78.27	62.20	66.34	61.74	81.01	64.93	69.55	65.26	95.87	65.04	76.65	68.12	94.86	64.88	79.58	68.66
CTM	**81.10**	**69.42**	**68.37**	**66.59**	**87.03**	**77.85**	**76.61**	**75.64**	**96.57**	**74.08**	**77.55**	**74.46**	**96.41**	**79.81**	**83.23**	**80.34**

Table 3: Comparison results of our CTM variants with different encoders and their respective baselines.

Tasks	Law Articles (%)				Charges (%)			
Metrics	Acc.	MR	MP	F1	Acc.	MR	MP	F1
TextCNN	75.97	54.30	60.84	53.52	80.05	60.72	65.22	60.54
Text-CTM	**80.37**	**67.08**	**65.53**	**63.50**	**85.78**	**73.97**	**73.03**	**70.80**
BiGRU	77.35	60.73	62.94	59.63	81.54	68.09	70.22	67.27
BiGRU-CTM	**81.10**	**69.42**	**68.37**	**66.59**	**87.03**	**77.85**	**76.61**	**75.64**
Lawformer	81.94	73.77	72.68	71.46	87.24	81.58	81.26	79.80
Law-CTM	**84.12**	**74.63**	**76.56**	**73.83**	**89.82**	**81.79**	**83.40**	**81.05**

Table 4: Comparison results of our CTM variants with different decoding structures and their respective baselines.

Tasks	Law Articles (%)				Charges (%)			
Metrics	Acc.	MR	MP	F1	Acc.	MR	MP	F1
MTL	77.06	60.76	63.21	59.60	81.72	68.31	71.54	67.57
MTL-CTM	**80.85**	**69.76**	**68.22**	**66.72**	**86.94**	**78.23**	**77.18**	**75.83**
TopJudge	77.35	60.73	62.94	59.63	81.54	68.09	70.22	67.27
TopJudge-CTM	**81.10**	**69.42**	**68.37**	**66.59**	**87.03**	**77.85**	**76.61**	**75.64**
MPBFN	72.77	50.55	53.25	48.74	75.41	56.15	59.28	55.15
MPBFN-CTM	**78.72**	**62.36**	**61.71**	**59.51**	**82.86**	**70.74**	**70.33**	**68.42**

Table 5: Results of the ablation studies on CAIL-small.

Tasks	Law Articles (%)				Charges (%)			
Metrics	Acc.	MR	MP	F1	Acc.	MR	MP	F1
CTM	**81.10**	**69.42**	**68.37**	**66.59**	**87.03**	**77.85**	**76.61**	**75.64**
w/o CD	78.11	63.16	65.12	61.87	82.68	70.91	72.69	69.93
w/o CD+DS	77.77	62.90	64.59	61.66	82.60	70.66	72.63	69.90
w/o CD+CT+RA	77.35	60.73	62.94	59.63	81.54	68.09	70.22	67.27

Table 6: Average accuracies of our CTM variants and their respective baselines across different frequency groups.

Tasks	Law Articles (%)				Charges (%)			
Groups	H1	H2	L1	L2	H1	H2	L1	L2
MTL	89.30	81.06	53.73	10.86	88.53	83.60	68.75	44.86
MTL-CTM	**89.89**	**84.73**	**68.50**	**21.83**	**89.74**	**85.28**	**83.39**	**59.92**
TopJudge	89.01	78.60	43.04	6.26	89.16	80.47	60.13	27.63
TopJudge-CTM	**90.49**	**84.03**	**69.96**	**22.50**	**89.65**	**84.67**	**84.03**	**63.94**
MPBFN	85.94	76.78	40.82	3.64	**86.34**	78.46	58.08	24.62
MPBFN-CTM	**87.78**	**80.13**	**60.74**	**12.87**	86.17	**79.52**	**75.71**	**52.94**

paring CTM and 'w/o CD', the introduction of the category decoder module results in greater gains. This may be due to the fact that refining the encoding process alone is still limited by the biased decoder training, and it is more beneficial to the model by refining the encoding and decoding processes jointly. Overall, the three customized modules we propose are necessary and can cooperate to achieve significant performance improvement.

4.5 Analysis of Gain Sources

In order to have a deeper understanding of the source of the performance gain, we compare and analyze the accuracy of the three variants of our CTM and the baselines on law articles and charges with different frequencies. The results of this fine-grained evaluation on CAIL-small are shown in Figure 5, where the IDs on the horizontal axis are sorted in a descending order of frequency. In Table 6, we also report the average accuracies of the CTM variants and their respective baselines across four different frequency groups, i.e., the top 20% (H1), 20% to 40% (H2), 40% to 70% (L1) and the rest (L2) of the label frequencies. Combining the above results, we can find that the improvement of our CTM increases significantly with decreasing frequency, which verifies the effectiveness of the designed case triples, especially for the low-frequency cases.

4.6 Visualization of Case Representations

Finally, we analyze the source of performance gain from the perspective of model training, i.e., compare the case representations generated by the baselines and its improved version via our CTM. We take MPBFN and MPBFN-CTM as an example due to space limitation. Specifically, in the case sampling module, we have obtained the high-frequency and low-frequency subsets of the law articles and

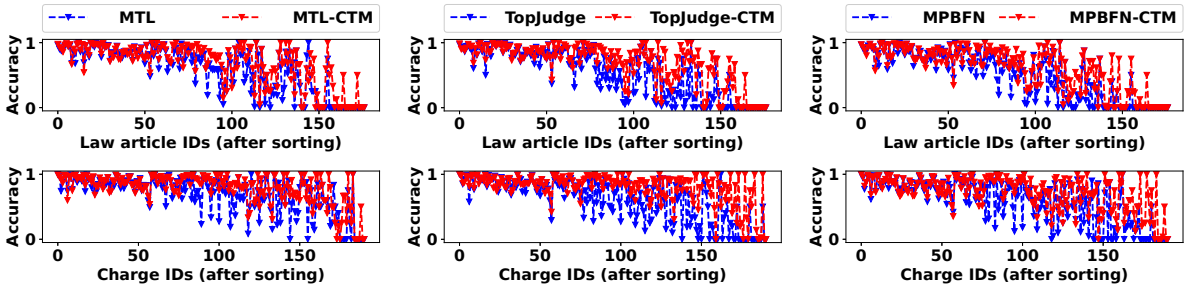

(a) (b) (c)

Figure 5: The prediction accuracy of our CTM variants and their respective baselines on law articles and charges with different frequencies from CAIL-small. Note that the IDs on the horizontal axis have been sorted in a descending order of frequency.

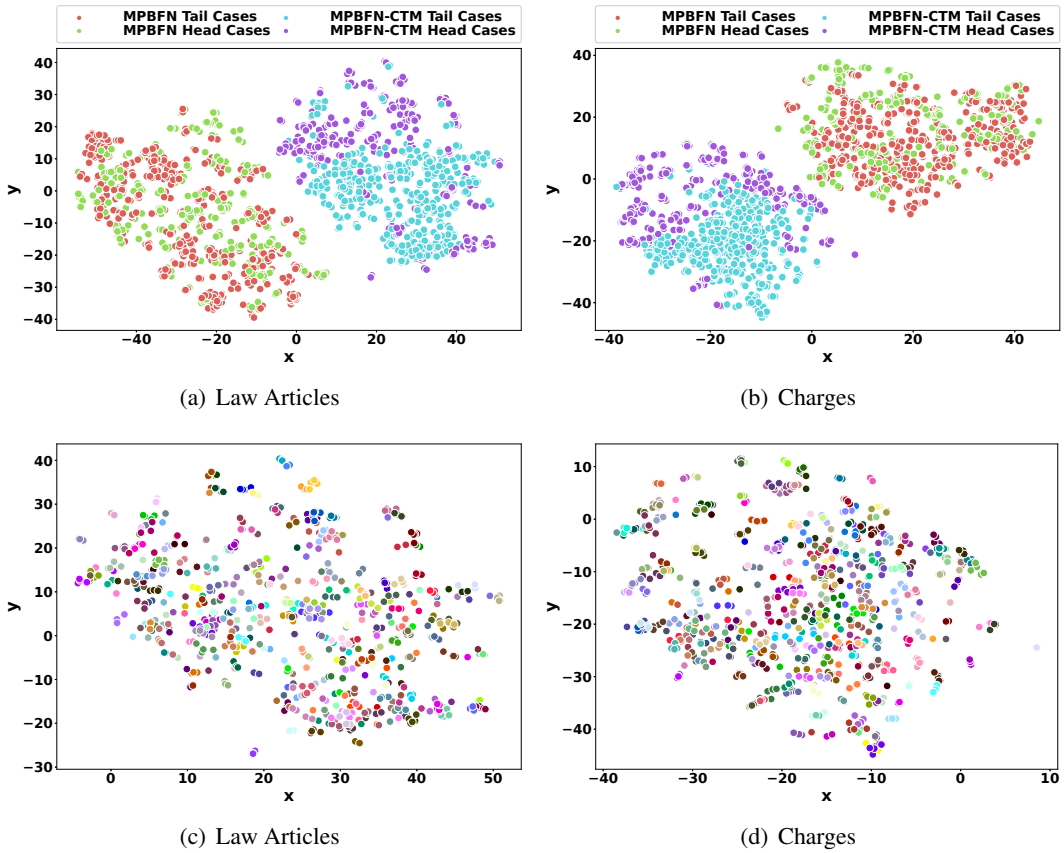

(a) Law Articles (b) Charges

(c) Law Articles (d) Charges

Figure 6: Visualization of the representations of some randomly sampled cases with high- and low-frequency law articles (a) and charges (b) on CAIL-small by MPBFN and MPBFN-CTM. The dots in (c) and (d) are fine-grained visualization of the representations obtained by MPBFN-CTM on each law article and charge, where the representations with the same law article or charge are clearly grouped.

charges. Then, we randomly sample 5 cases for each high-frequency (or low-frequency) law article and charge to construct their respective head (or tail) case sets. We respectively visualize the case representations generated by MPBFN and MPBFN-CTM on different sets.

The results are shown in Figure 6(a) and 6(b). We can find that the case representations generated by MPBFN have confusion on the head and tail sets (i.e., the green dots and red dots), and the case representations generated by MPBFN-CTM can cluster the head and tail sets separately and distinguish them effectively (i.e., the purple dots and blue dots). This clearly shows that the introduction of the case relations helps guide the encoder to learn the inter-class discrimination between the high-frequency and the low-frequency cases. We further present fine-grained visualization of the representations obtained by our CTM on each law article and each charge in Figure 6(c) and 6(d), respectively. As expected, we can see that most of the same law articles or charges, i.e., with the same colors, are clearly grouped.

5 Conclusions and Future Work

In this paper, we introduce some contrastive case relations to construct case triples as a new form of modeling, and propose a general judgment prediction framework with case triple modeling (CTM). Our CTM includes three new modules, i.e., a case sampling module for constructing case triples, a relational attention module for extracting information from case triples to refine the encoding process, and a category decoder module for refining the decoding process. Finally, we conduct extensive experiments on two public datasets and find that our CTM can effectively improve the performance of legal judgment prediction, especially for cases with low-frequency law articles or charges, and is also of good compatibility.

For future works, we plan to extend our CTM to more scenarios such as cases with multiple law articles or charges by further improving the corresponding case triple module and relational attention module. We are also interested in generalizing our CTM for prediction of the terms of penalty.

Acknowledgements

We thank the support of National Natural Science Foundation of China Nos. 61836005, 62272315 and 62172283.

References

Qiaoben Bao, Hongying Zan, Peiyuan Gong, Junyi Chen, and Yanghua Xiao. 2019. Charge prediction with legal attention. In *Proceedings of the 8th CCF International Conference on Natural Language Processing and Chinese Computing*, pages 447–458.

Iz Beltagy, Matthew E Peters, and Arman Cohan. 2020. Longformer: The long-document transformer. *arXiv preprint arXiv:2004.05150*.

Ilias Chalkidis, Ion Androutsopoulos, and Nikolaos Aletras. 2019. Neural legal judgment prediction in english. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 4317–4323.

Huajie Chen, Deng Cai, Wei Dai, Zehui Dai, and Yadong Ding. 2019a. Charge-based prison term prediction with deep gating network. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing*, pages 6362–6367.

Si Chen, Pengfei Wang, Wei Fang, Xingchen Deng, and Feng Zhang. 2019b. Learning to predict charges for judgment with legal graph. In *Proceedings of the 28th International Conference on Artificial Neural Networks*, pages 240–252.

Wenqing Chen, Jidong Tian, Liqiang Xiao, Hao He, and Yaohui Jin. 2020. Exploring logically dependent multi-task learning with causal inference. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing*, pages 2213–2225.

Yuh-Shyan Chen, Shin-Wei Chiang, and Meng-Luen Wu. 2021. A few-shot transfer learning approach using text-label embedding with legal attributes for law article prediction. *Applied Intelligence*, pages 1–19.

Qian Dong and Shuzi Niu. 2021. Legal judgment prediction via relational learning. In *Proceedings of the 44th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 983–992.

Congqing He, Li Peng, Yuquan Le, Jiawei He, and Xiangyu Zhu. 2019. SECaps: A sequence enhanced capsule model for charge prediction. In *Proceedings of the 28th International Conference on Artificial Neural Networks*, pages 227–239.

Zhilong Hong, Qifei Zhou, Rong Zhang, Weiping Li, and Tong Mo. 2020. Legal feature enhanced semantic matching network for similar case matching. In *Proceedings of the 2020 International Joint Conference on Neural Networks*, pages 1–8.

Zikun Hu, Xiang Li, Cunchao Tu, Zhiyuan Liu, and Maosong Sun. 2018. Few-shot charge prediction with discriminative legal attributes. In *Proceedings of the 27th International Conference on Computational Linguistics*, pages 487–498.

Xin Jiang, Hai Ye, Zhunchen Luo, Wenhan Chao, and Wenjia Ma. 2018. Interpretable rationale augmented charge prediction system. In *Proceedings of the 27th International Conference on Computational Linguistics: System Demonstrations*, pages 146–151.

Yoon Kim. 2014. Convolutional neural networks for sentence classification. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing*, pages 1746–1751.

Yuquan Le, Congqing He, Meng Chen, Youzheng Wu, Xiaodong He, and Bowen Zhou. 2020. Learning to predict charges for legal judgment via self-attentive capsule network. In *Proceedings of the 24th European Conference of Artificial Intelligence*, pages 1802–1809.

Shangbang Long, Cunchao Tu, Zhiyuan Liu, and Maosong Sun. 2019. Automatic judgment prediction via legal reading comprehension. In *China National Conference on Chinese Computational Linguistics*, pages 558–572.

Bingfeng Luo, Yansong Feng, Jianbo Xu, Xiang Zhang, and Dongyan Zhao. 2017. Learning to predict charges for criminal cases with legal basis. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 2727–2736.

Luyao Ma, Yating Zhang, Tianyi Wang, Xiaozhong Liu, Wei Ye, Changlong Sun, and Shikun Zhang. 2021. Legal judgment prediction with multi-stage case representation learning in the real court setting. In *Proceedings of the 44th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 993–1002.

Malte Ostendorff, Elliott Ash, Terry Ruas, Bela Gipp, Julian Moreno-Schneider, and Georg Rehm. 2021. Evaluating document representations for content-based legal literature recommendations. In *Proceedings of the 18th International Conference on Artificial Intelligence and Law*, pages 109–118.

Badri Patro and Vinay P Namboodiri. 2018. Differential attention for visual question answering. In *Proceedings of the 31st IEEE Conference on Computer Vision and Pattern Recognition*, pages 7680–7688.

Dunlu Peng, Jiyin Yang, and Jing Lu. 2020. Similar case matching with explicit knowledge-enhanced text representation. *Applied Soft Computing*, 95:106514.

Florian Schroff, Dmitry Kalenichenko, and James Philbin. 2015. FaceNet: A unified embedding for face recognition and clustering. In *Proceedings of the 28th IEEE Conference on Computer Vision and Pattern Recognition*, pages 815–823.

Li Tang and Simon Clematide. 2021. Searching for legal documents at paragraph level: Automating label generation and use of an extended attention mask for boosting neural models of semantic similarity. In *Proceedings of the Natural Legal Language Processing Workshop 2021*, pages 114–122.

Chaojun Xiao, Xueyu Hu, Zhiyuan Liu, Cunchao Tu, and Maosong Sun. 2021. Lawformer: A pre-trained language model for chinese legal long documents. *AI Open*, 2:79–84.

Chaojun Xiao, Haoxi Zhong, Zhipeng Guo, Cunchao Tu, Zhiyuan Liu, Maosong Sun, Yansong Feng, Xianpei Han, Zhen Hu, Heng Wang, et al. 2018. CAIL2018: A large-scale legal dataset for judgment prediction. *arXiv preprint arXiv:1807.02478*.

Chaojun Xiao, Haoxi Zhong, Zhipeng Guo, Cunchao Tu, Zhiyuan Liu, Maosong Sun, Tianyang Zhang, Xianpei Han, Zhen Hu, Heng Wang, et al. 2019. CAIL2019-SCM: A dataset of similar case matching in legal domain. *arXiv preprint arXiv:1911.08962*.

Nuo Xu, Pinghui Wang, Long Chen, Li Pan, Xiaoyan Wang, and Junzhou Zhao. 2020. Distinguish confusing law articles for legal judgment prediction. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 3086–3095.

Wenmian Yang, Weijia Jia, Xiaojie Zhou, and Yutao Luo. 2019. Legal judgment prediction via multi-perspective bi-feedback network. In *Proceedings of the 28th International Joint Conference on Artificial Intelligence*, pages 4085–4091.

Zichao Yang, Diyi Yang, Chris Dyer, Xiaodong He, Alex Smola, and Eduard Hovy. 2016. Hierarchical attention networks for document classification. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 1480–1489.

Linan Yue, Qi Liu, Binbin Jin, Han Wu, Kai Zhang, Yanqing An, Mingyue Cheng, Biao Yin, and Dayong Wu. 2021. NeurJudge: A circumstance-aware neural framework for legal judgment prediction. In *Proceedings of the 44th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 973–982.

Haoxi Zhong, Zhipeng Guo, Cunchao Tu, Chaojun Xiao, Zhiyuan Liu, and Maosong Sun. 2018. Legal judgment prediction via topological learning. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 3540–3549.

Haoxi Zhong, Chaojun Xiao, Cunchao Tu, Tianyang Zhang, Zhiyuan Liu, and Maosong Sun. 2020. How does NLP benefit legal system: A summary of legal artificial intelligence. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 5218–5230.

Constrained Regeneration for Cross-Lingual Query-Focused Extractive Summarization

Elsbeth Turcan[1], **David Wan**[2,*], **Faisal Ladhak**[1], **Petra Galuscakova**[3,†], **Sukanta Sen**[4,‡]
Svetlana Tchistiakova[4,‡], **Weijia Xu**[5], **Marine Carpuat**[5], **Kenneth Heafield**[4],
Douglas Oard[5] and **Kathleen McKeown**[1]
[1]Department of Computer Science, Columbia University;
[2]Department of Computer Science, University of North Carolina at Chapel Hill;
[3]Université Grenoble Alpes; [4]School of Informatics, The University of Edinburgh;
[5]Department of Computer Science & UMIACS, The University of Maryland
{eturcan, faisal}@cs.columbia.edu, davidwan@cs.unc.edu

Abstract

Query-focused summaries of foreign-language, retrieved documents can help a user understand whether a document is actually relevant to the query term. A standard approach to this problem is to first translate the source documents and then perform extractive summarization to find relevant snippets. However, in a cross-lingual setting, the query term does not necessarily appear in the translations of relevant documents. In this work, we show that constrained machine translation and constrained post-editing can improve human relevance judgments by including a query term in a summary when its translation appears in the source document. We also present several strategies for selecting only certain documents for regeneration which yield further improvements.

1 Introduction

Query-focused summarization creates an overview of a document which reflects how that document may be relevant to a provided query; such a task is useful for any search engine, such as for news articles or academic papers, where a user may want to search documents by a given query. In this paper, we further narrow the use case to one in which the user seeks a document containing a single specific query term (which may be a multi-word expression). For example, if the query is "dossier", the user is interested in finding information about a specific type of collection of files, as might exist in an intelligence investigation. A summary in the user's language can help them decide if a foreign language document is relevant.

Our work focuses on query-focused extractive summarization in a cross-lingual setting, where the summaries are generated in a language (here, English) different from the source language of the documents (here, Farsi, Kazakh, and Georgian). Because large summarization corpora do not exist in these languages, we follow a translate-then-summarize approach (Wan et al., 2010) in which we first apply machine translation (MT) to translate documents into English, a language with abundant summarization corpora, and then summarize the translated document; however, this introduces additional concerns. Translating a document once, before a query term is known, can lead to wording choices that are sub-optimal for any particular query term (e.g., if the Kazakh for "dossier" were translated as "file", and it may be unclear whether the specific meaning of dossier occurred in the source as opposed to other meanings of "file"). To address this, we present a *constrained regeneration* framework where we translate a document, summarize it with an extractive summarizer that uses evidence from the source language, and select a sentence to be regenerated under the constraint to include the requested query term if appropriate.

Our work is implemented within a pipeline that includes cross-lingual information retrieval (CLIR) followed by summarization of retrieved documents; in the latter step, a summary is generated for a document given a specific query term. Based on the intuition that seeing the query term in the summary is a strong signal of relevance to end users, we first present work on three types of constrained regeneration systems: Marian-C, a constrained version of Marian (Junczys-Dowmunt et al., 2018); EDITOR (Xu and Carpuat, 2021); and constrained automatic post-editing (Wan et al., 2020, cAPE). In initial experimentation, however, we found that these systems often insert the requested query term even in cases when the foreign document did not contain a suitable translation. To address this, we further in-

*Work performed while at Columbia University
†Work performed while at the University of Maryland
‡Work performed while at the University of Edinburgh

Proceedings of the 29th International Conference on Computational Linguistics, pages 2668–2680
October 12–17, 2022.

troduce document selection methods to determine when to apply regeneration and thus avoid inserting query terms inappropriately. We perform a human evaluation and show that the combined use of regeneration and document selection improve humans' ability to accurately distinguish relevant and irrelevant non-English documents by their generated English summaries.

Our approach combines complementary strengths of the three primary modules needed for cross-lingual query-focused summarization: CLIR excels at discovering cross-lingual mappings at the lexical level, neural MT produces complete sentences that are often very fluent, but sometimes at the expense of adequacy and term preservation, and summarization helps users assess relevance efficiently. The novelty of our approach lies in a tight integration of these components, exploiting CLIR to detect relevance, and combining summarization and selective regeneration of summary sentences to produce a human-useful summary.

Our contributions are as follows:

1. An approach to cross-lingual query-focused summarization using constrained regeneration to make it easier for humans to detect relevant documents.
2. A method of document selection enabling selective application of constrained regeneration to avoid over-generation of the query term.
3. Human evaluation in three different languages demonstrating that constrained MT performs better than constrained automatic post-editing for low-resource settings and that we can improve the end user's ability to identify relevant documents using our approach.

2 Background

2.1 Problem Definition

In this work, we operate in the setting of cross-lingual information retrieval and summarization. Our work focuses primarily on the summarization component of this problem, where we are given an English document D, composed of multiple sentences $s_1, s_2, ..., s_n$, and a search query q (which is a text string) and asked to generate a summary of D that condenses the information relevant to q. We apply extractive summarization, which means that our output summary S will be a subset of the sentences in D. In our setting, the English document D is actually a translation of a document F in another language, and the document-query pair $(\langle D, F\rangle, q)$ has been generated automatically by a CLIR system which was given q and a corpus (of length m) of source-language documents and their English translations $U = \{F_i, D_i\}_{0<i\leq m}$. This introduces some uncertainty as to whether D is always truly relevant to q. Moreover, the retrieval system uses a range of methods to deal with the mismatch between the q, D and F vocabulary, such as embeddings, n-best translations, query translation, and query expansion, and the retrieval thus does not guarantee that translations of the query terms occur in D even for the highly relevant documents.

This setup introduces our two main challenges. First, the initial translation of F into D was done without any query in mind, so it may contain synonyms or paraphrases of q, or it may have been incorrectly translated despite being relevant. Second, the generated summaries cannot always assume $\langle D, F\rangle$ is indeed relevant to q. Our goal is to generate summaries that contain the query q if and only if $\langle D, F\rangle$ is relevant to q without rerunning a large pipeline of CLIR and MT components.

2.2 Cross-Lingual Summarization Pipeline

Our system to translate from non-English documents and English query terms into English summaries is made up of several components developed by participants in the MATERIAL program[1]; its architecture can be seen in Figure 1. Documents are first translated from the source language into English using two different MT systems, Marian (Junczys-Dowmunt et al., 2018); and Google's multilingual neural MT (Google NMT) (Johnson et al., 2017).

The CLIR system, which retrieves relevant documents for a given query term and can work in tandem with MT, consists of a combination of 6 retrieval systems, including (1) statistical ranking (such as language models and BM25 (Robertson et al., 1995)), (2) neural ranking (Chen et al., 2021b), (3) re-ranking of both types, (4) stemming, (5) query expansion (using blind relevance feedback), and (6) document expansion (using DeepCT (Dai and Callan, 2019)). These systems were selected to perform optimally on each language and thus they differ for different languages. CLIR provides the ranking of the documents by relevance to the query, and also the cutoff point above which the documents should be relevant. This cut-

[1] https://www.iarpa.gov/research-programs/material

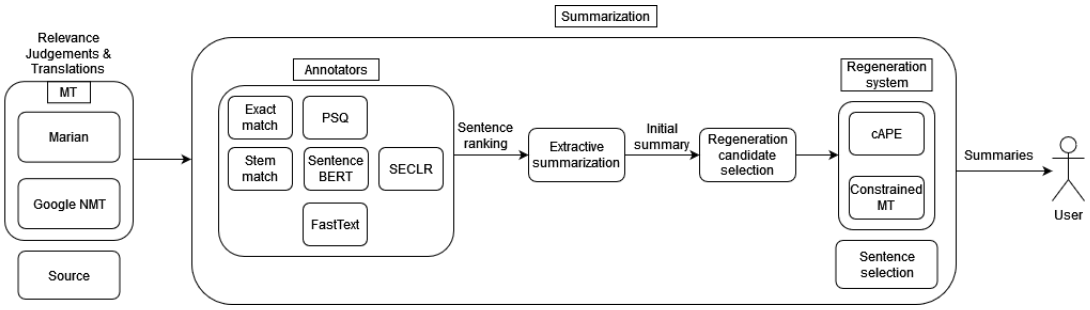

Figure 1: The architecture of our system pipeline as described in subsection 2.2.

System	Marian-C	Marian	EDITOR-C	EDITOR
fa→en	33.1	31.3	26.3	24.8
kk→en	30.2	28.0	20.5	20.5
ka→en	17.6	15.6	25.0	23.4

Table 1: BLEU scores of our constrained and unconstrained MT systems, computed using SacreBLEU (version string BLEU+case.mixed+numrefs.1+smooth.exp+tok.13a+version.1.5.1)

off uses an average of three estimates – the best ranked cutoff, sum-to-one cutoff and query specific threshold (Zhang et al., 2020) – and it is tuned to achieve an optimal F1 score.

Finally, our summarizer takes a given English document D and English query term q and generates an extractive summary S as relevant as possible to q using sentences from D. The summarizer contains several rankers which each rank all the sentences s_i of D from most to least relevant; these rankers include

1. a count of exact matches to q,
2. a count of stemmed matches to q using MorphAGram (Eskander et al., 2020),
3. mean cosine distance between the translated English sentence s_i and q using the 6B and 42B tokens GloVe embeddings (Pennington et al., 2014),
4. mean cosine distance between the source-language sentence and the English query term q using Probabilistic Structured Queries (PSQ) (Darwish and Oard, 2003),
5. mean cosine distance between the source-language sentence and the translated query term using FastText embeddings (Bojanowski et al., 2016) trained for the source language,
6. cosine distance between the translated English sentence s_i and English query term q using pretrained contextual Sentence-BERT embeddings (Reimers and Gurevych, 2019) based on RoBERTa-large (Liu et al., 2019), and
7. a cross-language sentence selector (Chen et al., 2021a, SECLR) that ranks directly using the sentences in the source language and the query term q.

We combine these rankings using the Borda count algorithm, a standard algorithm for unsupervised combination of rankers (Lillis, 2020; Aslam and Montague, 2001), to obtain a final relevance ranking for all the sentences in the document. We score the output of each MT system separately on each sentence to select the most appropriate translation for a given sentence, and then select the most relevant sentences to add to the summary until a fixed-length word budget is exhausted.

3 Our Models

Our approach to improve relevance judgments of summaries is based on (1) constrained regeneration models that encourage the inclusion of query terms in document translations, and (2) document and sentence selection models that identify documents where the inclusion of query terms is appropriate.

3.1 Constrained Regeneration

We experiment with three approaches that constrain the system to use the query term in the generated summaries: autoregressive MT (section 3.1.1), non-autoregressive MT (section 3.1.2), and automated post-editing (section 3.1.3). These approaches represent diverse state-of-the-art strategies to encourage rather than enforce the inclusion of query terms in translations (i.e., the query terms are soft rather than hard constraints). In this work, we experiment with soft constraints over hard constraints because of the intuition that soft constraints give our models the freedom to choose more natural synonyms and morphology as needed, and and based on empirical evidence that soft constraints result in more fluent and overall better translations (Xu and Carpuat, 2021).

3.1.1 Autoregressive Constrained Machine Translation: Marian-C

Marian-C is a constrained variant of the Marian system (Junczys-Dowmunt et al., 2018) trained on augmented synthetic data to encourage it to include English query terms in the translated English sentence.[2] Following Dinu et al. (2019),[3] we use a data augmentation technique to train our model to copy supplied query terms into its output. Augmentation simply consists in concatenating a query term to the source side of each training sample (with `|||`, a token of three pipe characters, as a delimiter). We create synthetic query terms for our parallel text by extracting random spans of target text of 1 to 3 words. We augment the data in this way with 75% probability. For the remaining 25%, we use the original training sample, to preserve the model's ability to translate when a query term is not available. During inference, the query q is simply appended to the source with the same delimiter.

3.1.2 Nonautoregressive Constrained Machine Translation: EDITOR

EDITOR takes the source sentence $\boldsymbol{x} = (x_1, x_2, ..., x_L)$ (where, in our case, $\boldsymbol{x} = s_i$) and optionally a sequence of constraint terms $\boldsymbol{C} = (c_1, ..., c_m)$ (here, $\boldsymbol{C} = q$) as inputs to generate a translation $\boldsymbol{y}$ that contains most of the constraint terms (Xu and Carpuat, 2021). The output is generated by iteratively editing an input sequence using repositioning, deletion and insertion operations. Constraints are seamlessly incorporated in decoding as the initial sequence $\boldsymbol{y}^0 = \boldsymbol{C}$ to be refined. They can thus be incorporated into the generated translation, or deleted, as the model sees fit. This process does not require custom training. An EDITOR model trained on a standard MT task can incorporate constraints in this way out of the box.

Table 1 shows that the resulting systems provide a wide range of quality levels as measured intrinsically by BLEU (Papineni et al., 2002). Farsi-English is evaluated on IWSLT 2012 and 2013 (Federico et al., 2012; Cettolo et al., 2013), Kazakh-English on WMT 2019 (Barrault et al., 2019), and Georgian-English on the MATERIAL ANALYSIS data described in section 4. Despite using similar parallel training sets, Marian performs better than EDITOR on Farsi (fa) and Kazakh (kk), while EDITOR outperforms Marian on Georgian (ka), reflecting independent system development processes that leverage monolingual data differently. Nevertheless, this provides a wide variety of translations that the summarization model can choose from.

3.1.3 Constrained Automatic Post-Editing

In contrast to MT systems that generate a new translation from scratch, constrained automatic post-editing (**cAPE**) edits the initial translation by incorporating desired words and fixing other potential errors. Following Wan et al. (2020), we use an autoregressive multi-source transformer model for this task. It takes as input the source sentence and the generated translation and outputs the corrected English sentence with the desired query term.

We generate synthetic post-editing triplets for training as follows. We use OPUS and ParaCrawl if available (section 4), resulting in 1.2M training examples for Kazakh-English, and 11.2M for Farsi. Each parallel sentence pair is augmented with an MT output from the relevant MT system (Marian and Google NMT). The original target plays the role of reference even though it was not generated by post-editing. We apply the same terminology set creation strategy and the same set of hyperparameters described in the original paper.

3.2 Document Selection

Due to the difficulty of cross-lingual retrieval and propagation of errors through the pipeline, we are likely to retrieve multiple documents that are not relevant to the given query. This is particularly problematic for regeneration, since the regeneration systems are optimized for including the constraints in their output, and thus may mislead users into judging summaries of irrelevant documents as relevant. Furthermore, regeneration adds additional computational overhead to the system, so we should run it only when we are relatively certain that a source document F is relevant to the query.

Therefore, in order to reduce the number of false positives, we add a document selection step that rescores the relevant documents by integrating scores from the CLIR system as well as the summarization system. In particular, we consider three values: (1) the document score from the SECLR query relevance component (Chen et al., 2021b) of the summarization system, (2) the CLIR system's document score, and (3) a binary variable that indicates whether the CLIR system's document score is above an F1 maximizing cutoff that was tuned on a development set (see Section 4). The new com-

[2] See Appendix A for training details.

[3] Different from Dinu et al. (2019), we do not use additional input factors.

posite score is simply the sum of those three values, all of which are bounded between zero and one.[4] We tune a threshold for the composite score using 100-fold cross-validation to achieve an optimal F1-score on the dev partition.[5] We then develop two systems to make use of this threshold.

+selection: This system presents documents selected for regeneration to human annotators using regenerated summaries and unselected documents using summaries that have not undergone regeneration. We expect that most unselected documents are not relevant, but this system favors high recall of the sort that may be valuable in applications like patent search or intelligence analysis.

+omission: This system assumes all unselected documents are irrelevant, and only asks human annotators for input on the regenerated summaries of documents that were selected. This is because some use cases may prefer higher precision at the cost of lower recall (for example, a casual searcher may prefer not to see irrelevant documents at all).

3.3 Sentence Selection

Once a document has been selected for regeneration, we use PSQ, a component of our CLIR model, to identify the sentences in the summary where it would be most appropriate to insert the query term. We rank each sentence by the maximum PSQ translation probability of any of its words with respect to the query term. We then select the sentence with highest rank (i.e., highest translation probability) to be regenerated; we break ties by the combined ranking of our other rankers as discussed in section 2.2, thus preferring sentences that also appear most conceptually related to the query term. In the event that no translation equivalent can be found through PSQ, regeneration would be aborted and the summary presented as originally created, but this never happens in our dataset.

4 Data

Machine Translation. The training corpora we use for our regeneration MT systems come from the WMT 2019 (Barrault et al., 2019), OPUS (Tiedemann, 2012), and MATERIAL-BUILD[6] parallel datasets for three languages: Farsi (FA), Kazakh (KK), and Georgian (KA). The dataset statistics are given in Table 3. We evaluate our full system on the MATERIAL text dataset consisting of source documents in the specified language as well as collection of English query terms.

Language	Collection	#documents	#queries
Farsi	ANALYSIS	388	221
	DEV	11,662	221
	EVAL	11,640	1264
Kazakh	ANALYSIS	388	400
	DEV	11,622	400
	EVAL	10,815	765
Georgian	ANALYSIS	388	412
	DEV	11,662	412
	EVAL	11,652	842

Table 2: Number of documents and queries for the MATERIAL dataset for evaluation.

	Corpus	#Sentence
	Farsi-English	
Para	OPUS	8.5M
	Hymers	22K
	Mizan	1M
	MATERIAL-BUILD	34K
	ParaCrawl	178K
	Lorelei	59K
	Kazakh-English	
Para	News Commentary	77K
	Wikititles	117K
	Kazakhtv	97K
	Crawl2019	495K
Mono	OPUS	131K
	News2019	20M
	News commentary.v15	608K
	Georgian-English	
Para	OPUS	1.7M
	Crawled	101K
	MATERIAL-BUILD	4K

Table 3: Parallel and monolingual corpora used in training the MT systems. The MATERIAL-BUILD corpus for Kazakh-English is the same as News Commentary.

Cross-Lingual Information Retrieval. The MATERIAL cross-lingual information retrieval dataset is divided into ANALYSIS, DEV, and EVAL, where ANALYSIS is intended for data statistics and examination, DEV for tuning and EVAL for test. The size of the splits are shown for each language in Table 2, and the structure of the data is similar to previous releases (Zavorin et al., 2020). This data includes, for each of our three languages, a separate collection of non-English news and blog documents, a separate collection of English query strings, and gold relevance annotations for each document-query pair within a language.

[4] We tried learning a logistic classifier; however, simply taking the sum of the scores performed similarly.

[5] We take the mean threshold over the 100 folds as the final threshold for our system.

[6] https://www.iarpa.gov/index.php/research-programs/material

5 Experiments

For each of our three languages, we draw a random sample of query-document pairs with a high likelihood of being relevant according to our trained CLIR system for that language. Then, for each query-document pair, we generate an extractive summary with no regeneration applied; this is our **baseline** system. We then apply each applicable constrained regeneration system to each summary independently, generating a new copy of the summary for each regeneration system.

We then submit each of these summaries to Amazon Mechanical Turk for human evaluation, the formulation of which is described in detail below in section 5.1. The result of the human evaluation is a relevance score for each summary–that is, for each $(q, \langle D, F\rangle,$ regeneration system$)$ triple. We can evaluate different regeneration systems at this stage by simply collecting the labels they are assigned and comparing them to the ground truth relevance labels. Finally, we apply our document selection methods; we select a subset of documents whose summaries should be regenerated according to our document selection threshold, and we use the collected scores from the regenerated and baseline variants to evaluate the +selection and +omission variants of the regeneration systems.

Our Farsi experiments are done on a random sample of 1000 query-document pairs from the documents returned by CLIR[7] for the MATERIAL EVAL partition, equally split between ground-truth irrelevant and relevant documents. These query-document pairs are selected such that the summaries the baseline system produced did not contain the query word, indicating an opportunity for regeneration systems to incorporate query terms. We repeat the experiments for Kazakh and Georgian similarly, using samples of 2000 documents for each language from their DEV partitions.

5.1 Human Evaluation

We evaluate our systems in an end-to-end fashion; in our setup, this means that we compare the ground-truth gold relevance label for each query-document pair with the relevance judgment assigned to that pair by human evaluators. The system we develop inherently includes a human in the loop, as its intended purpose is to allow a human to find documents relevant to an intended search term quickly and easily; therefore, we also involve human annotators in its evaluation.

[7]Documents returned by CLIR are those above a threshold that maximizes the Actual Query-Weighted Value (AQWV) that was learned on the dev partition). Details on this metric can be found at https://www.nist.gov/itl/iad/mig/iarpa-material-program.

Score	Precision	Recall	F1
Baseline	53.00	36.81	43.44
Baseline + omission	79.17	13.19	22.62
+cAPE	57.89	76.39	**65.87***
+cAPE +selection	59.79	53.01	56.20*
+cAPE +omission	**81.93**	29.40	43.27*
+Marian-C	52.58	70.83	60.36*
+Marian-C +selection	56.72	52.78	54.68
+Marian-C +omission	72.41	29.17	41.58
+EDITOR	50.22	**79.86**	61.66
+EDITOR +selection	57.89	53.47	55.60
+EDITOR +omission	75.44	29.86	42.79*

Table 4: **Farsi-English Document Relevance Evaluation**. Bold indicates the best score, and stars indicate statistically significant improvement over the baseline (by the approximate randomization test, $p < 0.05$).

For our human evaluation of our summaries, we asked workers on Amazon Mechanical Turk whether generated summaries were relevant to the given query term. We presented the summary, with any exact matches to the query term highlighted in a different color, to workers and asked them to rate the relevance on a five-point scale: {definitely irrelevant, probably irrelevant, unsure, probably relevant, definitely relevant}. For evaluation purposes, each worker's rating was binarized such that "probably relevant" and "definitely relevant" correspond to "relevant", and the others to "irrelevant". We asked three workers to evaluate each summary and aggregated their binarized judgments by majority vote, yielding a single final "relevant" or "irrelevant" human label for each query-summary pair. An example of the interface for this evaluation is included in Appendix B.

5.2 Evaluation Metrics

Our problem is a binary classification problem: a document-query pair is either relevant or irrelevant. We compare relevance judgements obtained during

Score	Precision	Recall	F1
Baseline	25.18	39.08	30.63
Baseline +omission	87.50	16.09	27.18*
+cAPE	25.48	75.86	38.15*
+cAPE +selection	30.61	51.72	38.46*
+cAPE +omission	**89.29**	28.74	43.48*
+Marian-C	21.52	**81.61**	34.05*
+Marian-C +selection	31.37	55.17	40.00
+Marian-C +omission	82.35	32.18	**46.28***
+EDITOR	24.90	68.97	36.59
+EDITOR +selection	26.76	43.68	33.19
+EDITOR +omission	78.26	20.69	32.73*

Table 5: **Kazakh-English Document Relevance Evaluation**. Bold indicates the best score, and stars indicate statistically significant improvement over the baseline (by the approximate randomization test, $p < 0.05$).

Score	Precision	Recall	F1
Baseline	14.35	29.25	19.25
Baseline +omission	30.43	13.21	18.42*
+cAPE	18.11	45.28	25.88
+cAPE +selection	17.02	37.74	23.46
+cAPE +omission	**35.38**	21.70	26.90*
+Marian-C	15.00	**62.26**	24.18*
+Marian-C +selection	18.31	49.06	26.67
+Marian-C +omission	30.70	33.02	**31.82***
+EDITOR	14.44	50.65	22.48
+EDITOR +selection	16.18	42.86	23.49
+EDITOR +omission	31.25	25.97	28.37

Table 6: **Georgian-English Document Relevance Evaluation**. Bold indicates the best score, and stars indicate statistically significant improvement over the baseline (by the approximate randomization test, $p < 0.05$).

human evaluation with reference judgments from the MATERIAL data, using the standard precision, recall and F_1 metrics. Reporting precision and recall independently provides important indicators of the incidence of false positives and false negatives respectively. A false positive represents a document that was not truly relevant to the query, but for which the generated summary falsely convinced the human annotators that it was relevant. Conversely, a false negative represents a relevant document whose summary failed to convey its relevance to the query (and thus human annotators judged it irrelevant). We hypothesize that the blind application of regeneration to even irrelevant documents is likely to decrease the false negative rate, but it may also increase the false positive rate.

We also note that the +selection and +omission systems can be evaluated for each regeneration system by replacing unselected documents' human evaluation with either the human evaluation of the original, non-regenerated document (+selection), or an automatic "irrelevant" judgment (+omission).

6 Results

The results of our experiments are shown in Table 4 (Farsi-English), Table 5 (Kazakh-English), and Table 6 (Georgian-English). Different result trends emerge for the high-resource (Farsi) and low-resource (Kazakh, Georgian) languages.

Beginning with Farsi, we see that applying regeneration via cAPE performs best, improving the F1 score by 20 points over the baseline; both constrained MT systems yield lesser but similar improvements. These improvements are due to dramatic increases in recall and similar precision as compared to the baseline, indicating that relevant documents are much more likely to be noticed and selected by human annotators. In Farsi, however, the additional layer of document selection is unhelpful, as it mitigates the recall too much without a large increase in precision; simply applying regeneration to every returned document-query pair performs best for Farsi.

For the low-resource languages, Kazakh and Georgian, applying regeneration via cAPE or Marian-C shows consistent and significant improvement over the baseline, with Marian-C performing best. EDITOR particularly improves recall over the baseline, but overall the improvements are not statistically significant. We see similar trends as in Farsi, where applying any form of regeneration increases recall and yields similar precision when not using document selection, leading to increased F1. When we include document selection as a pipeline

step before applying regeneration, however, precision also increases for all systems while retaining an improvement in recall (though not as large); as the vast majority of documents are irrelevant to any given query term, selection results in an overall net increase in F1 for cAPE and Marian-C. EDITOR, which is less aggressive in including its constraints, interacts poorly with document selection in Kazakh and yields reduced F1 under this setting. Finally, the +omission variant of document selection actually performs best overall because of how much it improves precision, although it does not increase recall as much as the +selection variant. Thus we see three variants of our systems (no selection, +selection, +omission) occupying different points on the precision-recall tradeoff in the low-resource setting.

We therefore see that for our low-resource languages, adding document selection to our regeneration improves the overall performance because it increases precision; the regeneration systems in these languages tend to take irrelevant documents and make their summaries appear relevant. However, in the case of our high-resource language, the improvements to precision afforded by document selection are minimal and do not balance out its diminished recall. We note that from Table 1, the performance of the base MT systems in Farsi is better than that for the low-resource languages, and correspondingly, the performance of our end-to-end system is best in Farsi, even for the baseline. Our hypothesis is that the documents returned by CLIR for Farsi are already relevant and high-quality compared to those in the low-resource languages; thus document selection helps identify relevant documents in low-resource languages but is not necessary for Farsi.

7 Related Work

Constrained Machine Translation. One of the crucial components of our system is the ability of the MT system to generate translations with specific terminology. Recent works use either constrained decoding, which modifies the decoding scheme to specify which words must be incorporated in the output (Post and Vilar, 2018; Hokamp and Liu, 2017; Hasler et al., 2018), or data augmentation techniques which incorporate the query term as an additional input in the training data (Dinu et al., 2019; Wan et al., 2020; Xu and Carpuat, 2021), avoiding the need to add overhead to the decoding scheme.

Cross-lingual Summarization. Prior work on cross-lingual summarization has mostly focused on two paradigms – summarize-then-translate (Lim et al., 2004; Orăsan and Chiorean, 2008; Wan et al., 2010) and translate-then-summarize (Leuski et al., 2003; Ouyang et al., 2019). The summarize-then-translate approach, however, requires a large amount of summarization training data in the source language (Ladhak et al., 2020), which makes them unsuitable for our setting since the source languages in our setting are low-resource. Prior work has shown that translate-then-summarize approaches are prone to error propagation (Ouyang et al., 2019; Ladhak et al., 2020), and propose methods to produce more fluent summaries. In our setting, having a fluent translation is not sufficient – we also need to have a translation with wording that is appropriate for the given input query. Therefore, in our work we focus on an integration of summarization with regeneration to more clearly indicate relevance.

Query-Focused Summarization. Query-focused summarization has been explored in both the single-document (Nema et al., 2017; Egonmwan et al., 2019; Ishigaki et al., 2020; Laskar et al., 2020; Xie et al., 2020; Zhong et al., 2021; Su et al., 2021) and multi-document setting (Feigenblat et al., 2017; Baumel et al., 2018). Prior work models this task as a question answering task, with the query being a question and the summary being similar to a terse answer to the question, sourced from the document. Unlike prior work, which has focused on monolingual settings, our work looks at query-focused summarization in the cross-lingual setting, where the query (and therefore the output summary) is in a different language than the source document.

8 Conclusion

We have presented a novel method of cross-lingual query-focused extractive summarization in which we apply regeneration to a generated summary in order to force inclusion of the query term when it appears in the source language document. We demonstrated large, significant improvements over the baseline in all cases through the addition of regeneration, showing increased recall and precision over the baseline. For our noisy low-resource languages, the combination of an aggressive constrained MT system and a document selection filter

additionally allows the benefits of including the query term in a relevant summary while avoiding creating new false positives. We experimented with three methods of constrained regeneration, which attempt to re-translate or edit a given sentence to include a given constraint: constrained automatic post-editing (cAPE), nonautoregressive MT (EDITOR), and our own implementation of autoregressive MT (Marian-C). For low resource languages, autoregressive MT consistently performed better, while for Farsi, cAPE was best. We believe this work opens the door to interesting future work experimenting with more complex varieties of document selection; with different, customized kinds of constrained regeneration; and with what types of languages benefit from these and other techniques.

9 Acknowledgements

This research is based upon work supported in part by the Office of the Director of National Intelligence (ODNI), Intelligence Advanced Research Projects Activity (IARPA), via contract FA8650-17-C-9117. The views and conclusions contained herein are those of the authors and should not be interpreted as necessarily representing the official policies, either expressed or implied, of ODNI, IARPA, or the U.S. Government. The U.S. Government is authorized to reproduce and distribute reprints for governmental purposes not withstanding any copyright annotation therein.

References

Javed A. Aslam and Mark Montague. 2001. Models for metasearch. In *Proceedings of the 24th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '01, page 276–284, New York, NY, USA. Association for Computing Machinery.

Loïc Barrault, Ondřej Bojar, Marta R. Costa-jussà, Christian Federmann, Mark Fishel, Yvette Graham, Barry Haddow, Matthias Huck, Philipp Koehn, Shervin Malmasi, Christof Monz, Mathias Müller, Santanu Pal, Matt Post, and Marcos Zampieri. 2019. Findings of the 2019 conference on machine translation (WMT19). In *Proceedings of the Fourth Conference on Machine Translation (Volume 2: Shared Task Papers, Day 1)*, pages 1–61, Florence, Italy. Association for Computational Linguistics.

Tal Baumel, Matan Eyal, and Michael Elhadad. 2018. Query focused abstractive summarization: Incorporating query relevance, multi-document coverage, and summary length constraints into seq2seq models. *ArXiv*, abs/1801.07704.

Piotr Bojanowski, Edouard Grave, Armand Joulin, and Tomas Mikolov. 2016. Enriching word vectors with subword information. *arXiv preprint arXiv:1607.04606*.

Mauro Cettolo, Jan Niehues, Sebastian Stüker, Luisa Bentivogli, and Marcello Federico. 2013. Report on the 10th IWSLT evaluation campaign. In *Proceedings of the 10th International Workshop on Spoken Language Translation: Evaluation Campaign*, Heidelberg, Germany.

Yanda Chen, Chris Kedzie, Suraj Nair, Petra Galuscakova, Rui Zhang, Douglas Oard, and Kathleen McKeown. 2021a. Cross-language sentence selection via data augmentation and rationale training. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 3881–3895, Online. Association for Computational Linguistics.

Yanda Chen, Chris Kedzie, Suraj Nair, Petra Galuscáková, Rui Zhang, Douglas W. Oard, and Kathleen R. McKeown. 2021b. Cross-language sentence selection via data augmentation and rationale training. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing, ACL/IJCNLP 2021, (Volume 1: Long Papers), Virtual Event, August 1-6, 2021*, pages 3881–3895. Association for Computational Linguistics.

Zhuyun Dai and Jamie Callan. 2019. Context-aware sentence/passage term importance estimation for first stage retrieval.

Kareem Darwish and Douglas W. Oard. 2003. Probabilistic structured query methods. In *Proceedings of the 26th Annual International ACM SIGIR Conference on Research and Development in Informaion Retrieval*, SIGIR '03, page 338–344, New York, NY, USA. Association for Computing Machinery.

Georgiana Dinu, Prashant Mathur, Marcello Federico, and Yaser Al-Onaizan. 2019. Training neural machine translation to apply terminology constraints. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 3063–3068, Florence, Italy. Association for Computational Linguistics.

Sergey Edunov, Myle Ott, Michael Auli, and David Grangier. 2018. Understanding back-translation at scale. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 489–500, Brussels, Belgium. Association for Computational Linguistics.

Elozino Egonmwan, Vittorio Castelli, and Md Arafat Sultan. 2019. Cross-task knowledge transfer for query-based text summarization. In *Proceedings of*

the 2nd Workshop on Machine Reading for Question Answering, pages 72–77, Hong Kong, China. Association for Computational Linguistics.

Ramy Eskander, Francesca Callejas, Elizabeth Nichols, Judith Klavans, and Smaranda Muresan. 2020. MorphAGram, evaluation and framework for unsupervised morphological segmentation. In *Proceedings of the 12th Language Resources and Evaluation Conference*, pages 7112–7122, Marseille, France. European Language Resources Association.

M. Federico, M. Cettolo, L. Bentivogli, M. Paul, and S. Stüker. 2012. Overview of the IWSLT 2012 evaluation campaign. In *Proceedings of the 9th International Workshop on Spoken Language Translation: Evaluation Campaign*, pages 12–33, Hong Kong, Table of contents.

Guy Feigenblat, Haggai Roitman, Odellia Boni, and David Konopnicki. 2017. Unsupervised query-focused multi-document summarization using the cross entropy method. In *Proceedings of the 40th International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '17, page 961–964, New York, NY, USA. Association for Computing Machinery.

Jiatao Gu, James Bradbury, Caiming Xiong, Victor OK Li, and Richard Socher. 2018. Non-autoregressive neural machine translation. In *International Conference on Learning Representations*.

Jiatao Gu, Changhan Wang, and Junbo Zhao. 2019. Levenshtein transformer. In *Advances in Neural Information Processing Systems 32*, pages 11181–11191. Curran Associates, Inc.

Eva Hasler, Adrià de Gispert, Gonzalo Iglesias, and Bill Byrne. 2018. Neural machine translation decoding with terminology constraints. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 2 (Short Papers)*, pages 506–512, New Orleans, Louisiana. Association for Computational Linguistics.

Chris Hokamp and Qun Liu. 2017. Lexically constrained decoding for sequence generation using grid beam search. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1535–1546, Vancouver, Canada. Association for Computational Linguistics.

Tatsuya Ishigaki, Hen-Hsen Huang, Hiroya Takamura, Hsin-Hsi Chen, and Manabu Okumura. 2020. Neural query-biased abstractive summarization using copying mechanism. In *Advances in Information Retrieval: 42nd European Conference on IR Research, ECIR 2020, Lisbon, Portugal, April 14–17, 2020, Proceedings, Part II*, page 174–181, Berlin, Heidelberg. Springer-Verlag.

Melvin Johnson, Mike Schuster, Quoc V. Le, Maxim Krikun, Yonghui Wu, Zhifeng Chen, Nikhil Thorat, Fernanda Viégas, Martin Wattenberg, Greg Corrado, Macduff Hughes, and Jeffrey Dean. 2017. Google's multilingual neural machine translation system: Enabling zero-shot translation. *Transactions of the Association for Computational Linguistics*, 5:339–351.

Marcin Junczys-Dowmunt, Roman Grundkiewicz, Tomasz Dwojak, Hieu Hoang, Kenneth Heafield, Tom Neckermann, Frank Seide, Ulrich Germann, Alham Fikri Aji, Nikolay Bogoychev, André F. T. Martins, and Alexandra Birch. 2018. Marian: Fast neural machine translation in C++. In *Proceedings of ACL 2018, System Demonstrations*, pages 116–121, Melbourne, Australia. Association for Computational Linguistics.

Diederik P. Kingma and Jimmy Ba. 2015. Adam: A method for stochastic optimization. In *Proceedings of the 3th International Conference on Learning Representations*, San Diego, CA, USA.

Philipp Koehn, Hieu Hoang, Alexandra Birch, Chris Callison-Burch, Marcello Federico, Nicola Bertoldi, Brooke Cowan, Wade Shen, Christine Moran, Richard Zens, Chris Dyer, Ondřej Bojar, Alexandra Constantin, and Evan Herbst. 2007. Moses: Open source toolkit for statistical machine translation. In *Proceedings of the 45th Annual Meeting of the Association for Computational Linguistics Companion Volume Proceedings of the Demo and Poster Sessions*, pages 177–180, Prague, Czech Republic. Association for Computational Linguistics.

Faisal Ladhak, Esin Durmus, Claire Cardie, and Kathleen McKeown. 2020. WikiLingua: A new benchmark dataset for cross-lingual abstractive summarization. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 4034–4048, Online. Association for Computational Linguistics.

Md Tahmid Rahman Laskar, Enamul Hoque, and Jimmy Huang. 2020. Query focused abstractive summarization via incorporating query relevance and transfer learning with transformer models. In *Advances in Artificial Intelligence*, pages 342–348, Cham. Springer International Publishing.

Anton Leuski, Chin-Yew Lin, Liang Zhou, Ulrich Germann, Franz Josef Och, and Eduard Hovy. 2003. Cross-lingual c*st*rd: English access to hindi information. *ACM Transactions on Asian Language Information Processing*, 2(3):245–269.

David Lillis. 2020. On the evaluation of data fusion for information retrieval. In *Forum for Information Retrieval Evaluation*, FIRE 2020, page 54–57, New York, NY, USA. Association for Computing Machinery.

Jung-Min Lim, In-Su Kang, and Jong-Hyeok Lee. 2004. Multi-document summarization using cross-language texts. In *NTCIR*.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Roberta: A robustly optimized BERT pretraining approach. *CoRR*, abs/1907.11692.

Preksha Nema, Mitesh M. Khapra, Anirban Laha, and Balaraman Ravindran. 2017. Diversity driven attention model for query-based abstractive summarization. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1063–1072, Vancouver, Canada. Association for Computational Linguistics.

Constantin Orăsan and Oana Andreea Chiorean. 2008. Evaluation of a cross-lingual Romanian-English multi-document summariser. In *Proceedings of the Sixth International Conference on Language Resources and Evaluation (LREC'08)*, Marrakech, Morocco. European Language Resources Association (ELRA).

Myle Ott, Sergey Edunov, Alexei Baevski, Angela Fan, Sam Gross, Nathan Ng, David Grangier, and Michael Auli. 2019. Fairseq: A fast, extensible toolkit for sequence modeling. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics (Demonstrations)*, pages 48–53, Minneapolis, Minnesota. Association for Computational Linguistics.

Jessica Ouyang, Boya Song, and Kathy McKeown. 2019. A robust abstractive system for cross-lingual summarization. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 2025–2031, Minneapolis, Minnesota. Association for Computational Linguistics.

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. BLEU: a method for automatic evaluation of machine translation. In *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics*, pages 311–318, Philadelphia, Pennsylvania, USA. Association for Computational Linguistics.

Jeffrey Pennington, Richard Socher, and Christopher Manning. 2014. GloVe: Global vectors for word representation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1532–1543, Doha, Qatar. Association for Computational Linguistics.

Matt Post. 2018. A call for clarity in reporting BLEU scores. In *Proceedings of the Third Conference on Machine Translation: Research Papers*, pages 186–191, Brussels, Belgium. Association for Computational Linguistics.

Matt Post and David Vilar. 2018. Fast lexically constrained decoding with dynamic beam allocation for neural machine translation. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 1314–1324, New Orleans, Louisiana. Association for Computational Linguistics.

Nils Reimers and Iryna Gurevych. 2019. Sentence-bert: Sentence embeddings using siamese bert-networks. *CoRR*, abs/1908.10084.

Stephen Robertson, S. Walker, S. Jones, M. M. Hancock-Beaulieu, and M. Gatford. 1995. Okapi at trec-3. In *Overview of the Third Text REtrieval Conference (TREC-3)*, pages 109–126. Gaithersburg, MD: NIST.

Rico Sennrich, Barry Haddow, and Alexandra Birch. 2016. Neural machine translation of rare words with subword units. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1715–1725, Berlin, Germany. Association for Computational Linguistics.

Dan Su, Tiezheng Yu, and Pascale Fung. 2021. Improve query focused abstractive summarization by incorporating answer relevance. In *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*, pages 3124–3131, Online. Association for Computational Linguistics.

Jörg Tiedemann. 2012. Parallel data, tools and interfaces in OPUS. In *Proceedings of the Eighth International Conference on Language Resources and Evaluation (LREC'12)*, Istanbul, Turkey. European Language Resources Association (ELRA).

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Ł ukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc.

David Wan, Chris Kedzie, Faisal Ladhak, Marine Carpuat, and Kathleen McKeown. 2020. Incorporating terminology constraints in automatic post-editing. In *Proceedings of the Fifth Conference on Machine Translation*, pages 1193–1204, Online. Association for Computational Linguistics.

Xiaojun Wan, Huiying Li, and Jianguo Xiao. 2010. Cross-language document summarization based on machine translation quality prediction. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics*, pages 917–926, Uppsala, Sweden. Association for Computational Linguistics.

Yujia Xie, Tianyi Zhou, Yi Mao, and Weizhu Chen. 2020. Conditional self-attention for query-based summarization. *arXiv preprint arXiv:2002.07338*.

Weijia Xu and Marine Carpuat. 2021. EDITOR: An Edit-Based Transformer with Repositioning for Neural Machine Translation with Soft Lexical Constraints. *Transactions of the Association for Computational Linguistics*, 9:311–328.

Ilya Zavorin, Aric Bills, Cassian Corey, Michelle Morrison, Audrey Tong, and Richard Tong. 2020. Corpora for cross-language information retrieval in six less-resourced languages. In *Proceedings of the workshop on Cross-Language Search and Summarization of Text and Speech (CLSSTS2020)*, pages 7–13, Marseille, France. European Language Resources Association.

Le Zhang, Damianos Karakos, William Hartmann, Manaj Srivastava, Lee Tarlin, David Akodes, Sanjay Krishna Gouda, Numra Bathool, Lingjun Zhao, Richard Schwartz Zhuolin Jiang, and John Makhoul. 2020. The 2019 BBN Cross-lingual Information Retrieval System. In *Proceedings of the Cross-Language Search and Summarization of Text and Speech Workshop*, pages 44–51.

Ming Zhong, Da Yin, Tao Yu, Ahmad Zaidi, Mutethia Mutuma, Rahul Jha, Ahmed Hassan Awadallah, Asli Celikyilmaz, Yang Liu, Xipeng Qiu, and Dragomir Radev. 2021. QMSum: A new benchmark for query-based multi-domain meeting summarization. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 5905–5921, Online. Association for Computational Linguistics.

A Constrained Machine Translation Training Frameworks

In the case of Marian-C, we train separate autoregressive unidirectional models for each of Farsi-English, Kazakh-English and Georgian-English. To train our models, we first preprocess the parallel data using Moses (Koehn et al., 2007) punctuation normalization, tokenization, and true-casing. We then create a shared byte-pair encoding vocabulary of 32k tokens following the method of Sennrich et al. (2016), and tokenize our parallel data. We train a Transformer-base model following the method of (Vaswani et al., 2017) using the Marian-NMT framework (Junczys-Dowmunt et al., 2018). The models are trained until BLEU score performance (Papineni et al., 2002; Post, 2018) on the validation set ceases to improve for 15 checkpoints. We use the English→X model to create backtranslations (Edunov et al., 2018) of our monolingual data, and train again on a concatenation of the parallel data and the backtranslations together, in the same way, to create our final X→English models.

In the case of EDITOR, we train separate unidirectional models for Farsi-English, Kazakh-English and Georgian-English using the same preprocessing steps as Marian-C except that we use a shared byte-pair encoding vocabulary of 20k tokens. We apply sequence-level knowledge distillation from autoregressive teacher models as widely used in non-autoregressive generation (Gu et al., 2018, 2019; Xu and Carpuat, 2021). We train a Transformer-base model (Vaswani et al., 2017) using fairseq (Ott et al., 2019). The models are trained using Adam (Kingma and Ba, 2015) with initial learning rate of 0.0005 for maximum 300,000 steps. We select the best checkpoint based on validation BLEU (Papineni et al., 2002).

B Amazon Mechanical Turk Interface

An example of our Amazon Mechanical Turk interface for human evaluation can be seen in Figure 2. Five such questions were presented in each Human Intelligence Task (HIT).

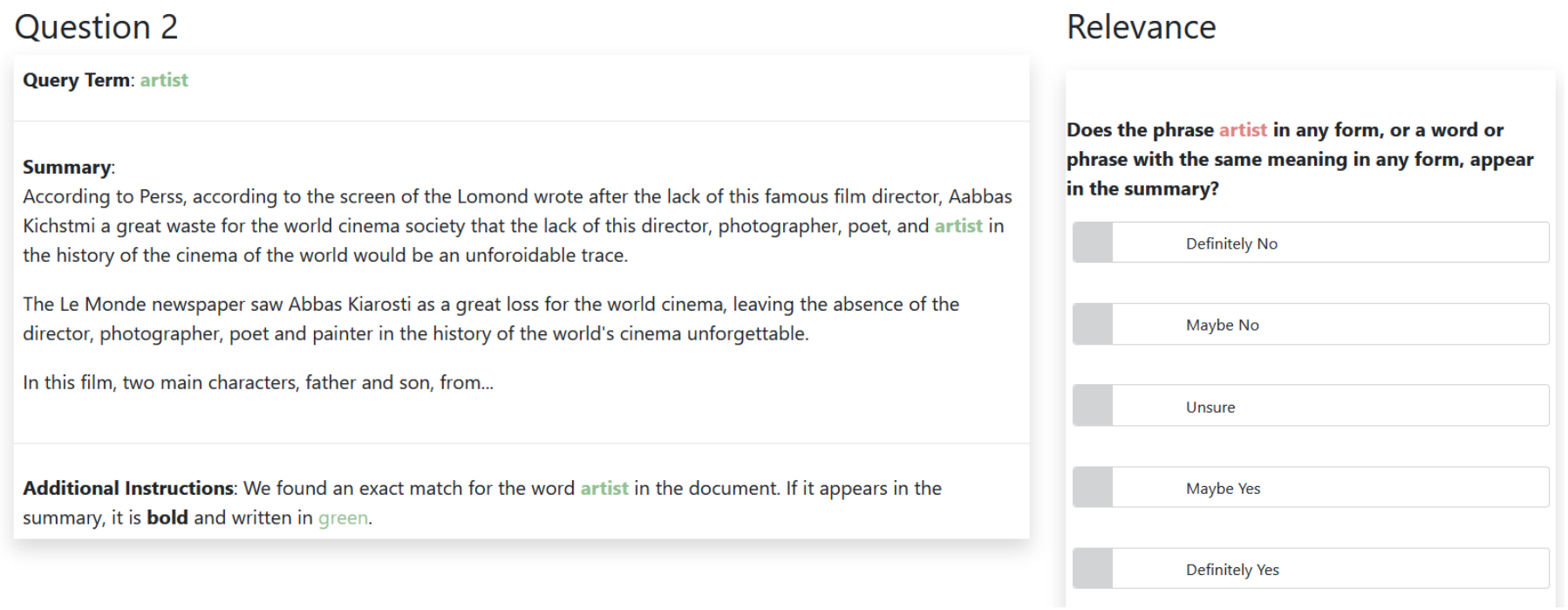

Figure 2: An example interface from our Amazon Mechanical Turk evaluation asking workers whether a given summary is relevant to the query. It includes highlighting of keywords and uses a 5-point scale to evaluate relevance.

Programmable Annotation with Diversed Heuristics and Data Denoising

†Ernie Chang, *Alex Marin, †Vera Demberg
†Dept. of Language Science and Technology, Saarland University
*Microsoft Corporation, Redmond, WA
cychang@coli.uni-saarland.de

Abstract

Neural natural language generation (NLG) and understanding (NLU) models are costly and require massive amounts of annotated data to be competitive. Recent data programming frameworks address this bottleneck by allowing human supervision to be provided as a set of labeling functions to construct generative models that synthesize weak labels at scale. However, these labeling functions are difficult to build from scratch for NLG/NLU models, as they often require complex rule sets to be specified. To this end, we propose a novel data programming framework that can jointly construct labeled data for language generation and understanding tasks – by allowing the annotators to modify an automatically-inferred alignment rule set between sequence labels and text, instead of writing rules from scratch. Further, to mitigate the effect of poor quality labels, we propose a dually-regularized denoising mechanism for optimizing the NLU and NLG models. On two benchmarks we show that the framework can generate high-quality data that comes within a 1.48 BLEU and 6.42 slot F1 of the 100% human-labeled data (42k instances) with just 100 labeled data samples – outperforming benchmark annotation frameworks and other semi-supervised approaches.

1 Introduction

Modern machine learning systems require large amounts of labeled data. For many applications, such labeled data is created by getting humans to explicitly label each training example. However, the standard labeling process that involves Wizard-of-Oz (Kelley, 1984) and other crowd-sourcing approaches (e.g. (Wen et al., 2017; Coucke et al., 2018; Budzianowski et al., 2018)) is restricted to the level of individual examples, and so are slow and static (Ratner et al., 2019). As a result, they are not only costly but require relabeling for any fine-grained domain revisions.

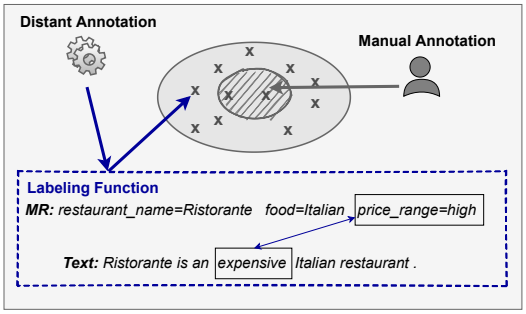

Figure 1: **Annotation scenario**: Each × represents a labeled data instance. The annotation framework allows to generalize from few human-labeled instances (inner) to large amounts of weakly labeled data (outer) by building rules (alignments) between sequence labels and text via labeling functions.

Data programming is a successful paradigm where humans provide low cost labeling rules written in programming languages to build imperfect training sets, which are then denoised for further improvements (Ratner et al., 2016a, 2017). However, two caveats exist: (1) Heuristic rules are costly to construct from scratch as exhaustive alignments between text and sequence labels have to be specified *manually* (Evensen et al., 2020), and especially strenuous for language generation that transforms meaning representations (MR) or slots into textual descriptions (e.g. (Reiter and Dale, 2000; Barzilay and Lapata, 2005)) where NLG is a one-to-many process (i.e.). (2) Labeling functions make decisions based on *discrete rules* that heavily limit the framework capability in making fine-grained decisions due to the lack of probabilistic information that guides the rule inference (Chatterjee et al., 2019).

To this end, we present a new data programming framework where language understanding and generation data can be *jointly labeled*. We argue that joint NLG/NLU annotation not only improves the overall data quality, but provide a greater degree of *compositionality* where semantic units such as slot-value pairs can be individually controlled. We tar-

Proceedings of the 29th International Conference on Computational Linguistics, pages 2681–2691
October 12–17, 2022.

get a weak supervision scenario (shown in Figure 1) consisting of small, high-quality expert-labeled data and a large set of unlabeled MR or text instances. In this framework, subject-matter experts are to use labeling rules to *modify* the automatically-inferred semantic alignments between MR and text, which are probabilistic rules that can be used for joint NLU/NLG labeling. The rules help to construct weak data via iterative denoising, before the dually-regularized NLU and NLG models can learn from the clean seed data to generate high-quality data. This work makes the following contributions:

- We introduce a novel annotation framework based on data programming that allows for joint labeling of language understanding and generation data. We validate the framework on two benchmarks and demonstrates its ability to create high quality data by expanding rules and then denoising the noisy data with dual regularization.

- We present a preliminary study to demonstrate the *compositionality* of the framework by showing that it can perform automatic domain revisions of MR slots without any relabeling efforts. This is especially beneficial in use of annotation tools where frequent data revisions are needed.

2 Related Work

Distant Supervision in Language Understanding. Learning with weak supervision is a well-studied area that is popularized by the rise of data-driven neural approaches (Ratner et al., 2017; Safranchik et al., 2020; Bach et al., 2017; Wu et al., 2018; Jiang et al., 2018). In particular, recent literature explore *knowledge distillation* from rules by either guiding the individual layers (Li and Srikumar, 2019) or training the model weights within constraints of the rule based system using a student and teacher model (Hu et al., 2016). Similarly, Snorkel and other techniques (Ratner et al., 2016b; Bach et al., 2017; Varma et al., 2019) rely on domain experts manually developing heuristics for noisy labels. However, these approaches are largely limited to NLU tasks and focus on providing discrete heuristics for tasks such as relations. Thus, our work serves to bridge this gap by (1) providing a way to more readily create texts, and (2) including probabilistic scores for labels.

Weak Supervision for Language Generation. Past works on semi-supervised learning consider settings with a large set of unlabeled data as in machine translation (Artetxe et al., 2017; Lample et al., 2017), or more relevantly the joint learning framework for training NLU and NLG (Tseng et al., 2020; Su et al., 2020; Schmitt and Schütze, 2019) that also considers a small labeled data. In particular, unsupervised statistical machine translation (e.g. (Artetxe et al., 2018; Lample et al., 2018)) utilizes statistical alignment models (Brown et al., 1993) that automatically infer explicit alignments between phrases in source and target sentences. Our work exploits this explicit alignment by treating them as a modifiable rule set[1], then using it to noisily synthesize weak data, allowing for annotation of NLU and NLG labels.

3 Annotation Framework Summary

Here we formally describe the joint annotation framework. Let X denote the set of meaning representation[2] (MR) instances and Y denote the text sequences. In our setting, we have (1) a seed dataset S which consists of k labeled pairs, and (2) a large unlabeled MR or text set U where $|U| \gg k > 0$. The annotation framework targets the creation of labeled samples $L = \{(x_1, y_1), \ldots, (x_n, y_n)\}$ where $x_t \in X$ is an MR instance and $y_t \in Y$ is its corresponding text. To do so, we construct a probabilistic rule set based on the seed data S (see §4) and outline this process in Figure 2. In particular, we draw the connection between the rule set *compositionality* and *coverage* to data denoising and highlight the framework advantages. The resulting rule set allows to create a large set of noisily-labeled data, where the framework then learns from the mixture of clean (seed) and noisy data in the process of data denoising (see §5) to create a higher quality set of data.

4 Rule Set Construction

We use the rule set to define the semantic alignment between MR and text. For instance, in a "cuisine" domain, the *fast food* slot can be aligned with slot value "Macdonald" and other related terms, and likewise for each value that might be associated with more than one slot. This relationship can

[1] This is similar to phrase table pruning (Zens et al., 2012; Galbrun, 2009).

[2] They can be seen as sequence labels.

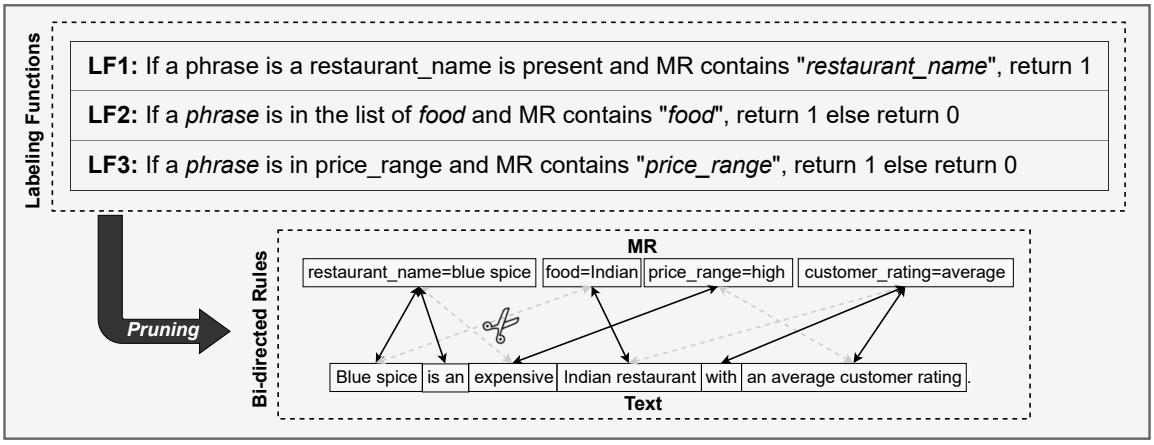

Figure 2: Depiction of the labeling process, where $\hat{x}$, $\hat{y}$ denote the noisy labels. Labeling functions are shown as textual descriptions, which are used to filter bi-directed rules below. In *bi-directed rules*, rules not removed by labeling functions are shown in **bold**. For more labeling functions see Appendix **??**.

be captured by the explicit many-to-many[3] rules derived from the MR-text pairs, which connects NLU and NLG as they are then explicitly linked with the alignments.

To build the rule set, we initialize the rule set in three steps: (1) We first generate new in-domain data using the pretrained language model (GPT-2) – this helps to increase data size and diversify the text distribution. (2) From this augmented data, we automatically initialize the rule set (phrase table) using a statistical alignment model. (3) Finally, we prompt the subject-matter experts to explicitly modify the alignments between phrases in MR and text with labeling functions, as these alignments help to refine the semantic relationships. Specifically, the aligned pairs are provided to experts where they are asked to write labeling functions to *prune* the non-matching phrases that contain incorrect semantic alignments (See Figure 2).

4.1 Diversifying Rule Set Diversity

To increase the diversity (and thus coverage) of the eventual annotation rule set generated, we perform an initial augmentation of the seed labeled data set S with additional weakly-labeled data, thus producing a larger but more diverse weakly-labeled dataset (L). We first generate additional MR via value swapping for each MR slot as in Chang et al. (2021a), then use GPT-2 (Radford et al., 2019) to perform conditional data-to-text generation as in Harkous et al. (2020); Mager et al. (2020). These works showed that fine-tuning GPT-2 on the joint distribution of MR and text for text-only generation yields decent performance. Given the sequential MR representation and a sentence in the seed labeled data, we maximize the joint probability $p_{\text{GPT-2}}(X, Y)$, where each sequence is concatenated into "[MR] $x_1 \cdots x_M$ [TEXT] $y_1 \cdots y_N$". The fine-tuned LM conditions on the augmented MR sample set to generate the in-domain text[4], and thus produces the augmented dataset with noisy texts. Similarly, for conditional MR generation with $p_{\text{GPT-2}}(Y, X)$, we apply the same process with text and MR flipped in the concatenation.

While the weak labels expands the seed data, *it creates noisy data by introducing false correlations between MR and text*. In what follows we discuss the creation of the rule set and the use of labeling functions that help to mitigate this noise.

4.2 Rule Set Initialization.

We extend the idea of a *phrase table* in statistical machine translation to be the rule set in our context: from the noisy augmented data L, we derive a rule set that is constructed based on the fertility-based[5] alignment model (GIZA++) (Och and Ney, 2003)[6] optimized using the EM algorithm (Dempster et al., 1977). This allows to obtain the semantic correspondences (or probabilistic rule set) $R_{i \to j} \subseteq \{(i, j, P_{i \to j}) : i = 1 \cdots |x|; j = 1 \cdots |y|\}$ where i and j refer to positions in flattened MR x and text sequence y, and $P_{i \to \hat{j}}$ is the probability of

[3]Relatedly, NLU is many-to-one and NLG is one-to-many (Tseng et al., 2020).

[4]We adopt the top-k random sampling with both $k = 2$ and $k = 15$ to encourage diversity and ensuring correct outputs (Radford et al., 2019)

[5]Fertility is defined as the number of words that correspond to a semantic unit.

[6]http://www.statmt.org/moses/giza/GIZA++.html

aligning the ith semantic unit in x to the jth unit in y. Each semantic correspondence can be seen as a bi-directed edge of a *rule* that connects semantic units (or phrases) in MR to phrases in text, where the granularity of each semantic unit is determined by the feature-based phrase-based alignment model (Brown et al., 1993). Thus, we can likewise induce $R_{j \to i} \subseteq \{(j, i, P_{j \to i})\}$ as the rule set that can be *derived for either NLU or NLG* inferences. We discuss the resolution of potential conflicts between rules in §5.

4.3 Building Labeling Functions

We ask a group of annotators to write Python code snippets within the time limit of an hour each. We denote these code snippets as labeling functions (LFs). Each labeling function follows basic rule relations, as shown in figure 2, and returns one of the possible values: $1, -1, 0$ ("valid"/"not valid"/"undetermined"). To prune the rule set, we apply the labeling functions to each rule, and judge whether the given rule is *strictly invalid* or not. Two types of LFs are designed: (1) *slot-specific LFs*: one LF is written for each slot identified in the MR, as all the decisions related to the slot can be grouped together. For certain slots, such as "restaurant_name", a basic dictionary of correct slot values is collected as to verify if mappings are correct. (2) *general LFs* help to eliminate false rules across all slots. One example is the rule that links a conjunction of text with a specific slot of the MR. Using the general LFs, such incorrect rules can be removed altogether. Rules marked as "undetermined" by the LFs are preserved and will be further evaluated by the denoising mechanism. For instance, the phrase "is a" may remain aligned with "restaurant_name" as no LF may have marked it as "not valid".

4.4 Rule Coverage and Compositionality

MRs are structured and compositional as they typically consist of attributes in the form of flat or tree-structured slot-value pairs (Balakrishnan et al., 2019). Here, we define the *compositionality* of a rule set as the average percentage of rules that correctly correlate MR and text over all slots. In our framework, rule sets can be manipulated to directly reflect high-level requirements in dialogue – such as the need to remove or add values to a slot in both MR and text for domain revisions. For slot removal, this can be done directly via the addition of labeling functions to remove specific attributes from MR and text altogether. An important trade-off exists between *rule coverage* and *compositionality* – higher rule coverage leads to lower compositionality, since a larger rule set tends to make more erroneous correlations between MR and text.

In what follows we describe the process of *data denoising*, in which the framework learns to utilize both the noisy and clean data.

5 Data Denoising

Label bias is a well-studied problem (Lafferty et al., 2001) where the frequency of some transitions will far outweigh the others even when they have roughly the same probability mass. Beneficially, the joint use of both NLU/NLG models helps to mutually shape the probability masses in token-label pairs. Figure 3 represents a simple finite-state model designed to map the two words "good" and "review" to their respective labels. Suppose that the observation sequence is "good review". From starting states 0 and 1, "good" matches both potential label transitions "price" and "rating", so the probability mass gets distributed rather equal among those two transitions.

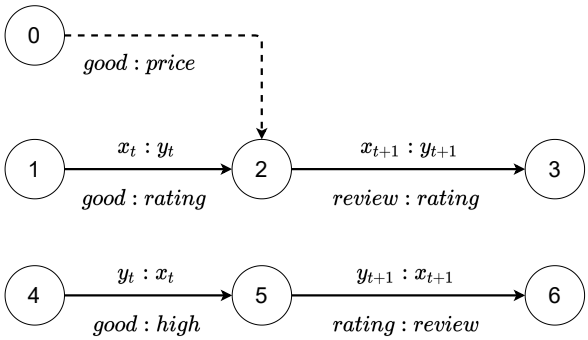

Figure 3: **Example of resolving label bias**: we place the token-label pairs (x_t:y_t) on transitions rather than states.

While the two paths will be roughly equally likely, but if one of the two labels is slightly more common in the training set (i.e. price), the transitions out of the start state will slightly prefer this corresponding transition, and yielding incorrect correlations. However, having both NLU and NLG models alleviate this bias since we now have two "versions" of the sequence in states $\{1, 2, 3\}$ and $\{4, 5, 6\}$ that can help to break the ties and shape the probability mass according to the sequential information.

Thus, we initialize noisy data with rules using the statistical *base models* and then training the *denoising models* to distill from the mixture of

noisy and clean data. This is done by iteratively creating noisy labels with rules and then refining these labeled data for higher quality ones. Further, we maintain both statistical NLU and NLG models for iterative joint labeling in the process.

5.1 Overview of the Denoising Mechanism

The denoising mechanism is achieved by first formulating the NLG model $p(y|x)$ as Brown et al. (1993):

$$p(y|x) \approx \sum_{j=1}^{m=|y|} \prod_{i=0}^{l=|x|} t(y_j|x_i)a(i|j,m,l) \quad (1)$$

where $t(\cdot)$ and $a(\cdot)$ are the translation and alignment probabilities learned from the seed data for x and y of lengths m and l respectively. Note that for the NLU model, we simply flip y and x in Eq. 1. Thus the formulation allows to estimate both statistical NLU and NLG models by adjusting the weights of log-linear combination of features[7] to optimize the evaluation metric (i.e. BLEU-4 (Papineni et al., 2002)) on the validation corpus via the minimum error rate training (MERT) (Och, 2003), which maximizes the BLEU-4 scores based on given inputs and their noisy labels.

In what follows we describe the training objectives to distill knowledge from the rule set into the base models (**Step-1**), then introduce clean data in **Step-2** to improve upon the data quality.

Step-1: Distillation from Noisy Rules. We optimize the parameters of the base version of the NLU and NLG models (**base models**) by alternately fixing the parameters of one model and optimizing the other model until convergence[8]: (1) MERT computes the optimal value for each model parameter and greedily selects data based on the generated candidate labels that leads to the largest gain in BLEU-4. (2) Then it noisily labels the MR samples with text or text with MR via the updated parameters at each iteration so as to obtain a better approximation of label candidates. The process of data denoising is functionally beneficial in reducing the label biases present in the imperfect labels.

Step-2: Adding Clean Data. The **denoising models** are trained with the expert-labeled seed data and the set of noisy data that were generated. However, the performance on the seed data is better than the pseudo-labels in the early rounds. This is anecdotally observed in both NLU and NLG models on various datasets and leads to potentially suboptimal performance (Shen and Sanghavi, 2019a,b). This motivates our proposal to reduce the total loss by using NLU and NLG models to select only a subset of data to train on – we filter out samples with large cross-entropy losses in early iterations *with replacement*, and train the models on the samples left after filtering. This serves to learn more effectively from both clean and corrupted data.

Specifically, we propose the use of *dual regularization (DR)* where $s_1, \ldots, s_n$ are the samples, θ are the model parameters:

$$\hat{\theta}^{(\text{DR})} = \arg\min_{\theta} \Big[\min_{\tilde{X} \subset X : |\tilde{X}| = \lfloor \beta * n \rfloor} \sum_{i \in \tilde{X}} L_{NLG_\theta(x_i)} \Big] + \arg\min_{\theta} \Big[\min_{\tilde{Y} \subset Y : |\tilde{Y}| = \lfloor \beta * n \rfloor} \sum_{i \in \tilde{Y}} L_{NLU_\theta(y_i)} \Big].$$

To find $\hat{\theta}^{(\text{DR})}$, we minimize over both the (a) sample subsets $\tilde{X}, \tilde{Y}$ given the ratio β and (b) the model parameters θ. In (a), the MR-text sample size is $\lfloor \beta * n \rfloor$, where $\beta = 0.1$[9] is the ratio of training samples to train on. $\tilde{X}, \tilde{Y}$ are the subsets of X, Y selected for the NLU and NLG models.

In this formulation, NLU and NLG models jointly select samples that are deemed to be less challenging for each other, before proceeding to learn from more challenging samples[10]. As such, the NLU and NLG models are kept to be similar in inference capabilities; this allows to select more suitable samples for each other due to the smaller degree of semantic misalignment.

6 Experiment Setting

We conduct experiments on the Weather (Balakrishnan et al., 2019) and E2E (Novikova et al., 2017b) datasets. Weather contains 25k instances of tree-structure annotations. E2E is a crowd-sourced dataset containing 50k instances in the restaurant domain. The NLU and NLG models are implemented in PyTorch (Paszke et al., 2019) with 2 Bi-LSTM layers and 100-dimensional token embeddings and Adam optimizer (Kingma and Ba, 2014) with an initial learning rate of 0.0002. The

[7] This includes the bidirectional *rule*, *lexical probabilities*, the *language model*, the *reordering model*, the *word penalty* and the *phrase penalty*.

[8] Following the training procedures in Artetxe et al. (2018).

[9] Following (Shen and Sanghavi, 2019b), we set β to be 1%, as it was shown to work well in generative models.

[10] It is related to *competence-based curriculum learning* (Platanios et al., 2019) where samples are selected based on their difficulties and the model competence.

	MR → Text (NLG)							Text → MR (NLU)						
Text-only	100	150	200	250	300	SR	VE	100	150	200	250	300	SR	VE
seq2seq	35.55	41.79	41.63	43.50	46.62	39.20	40.61	47.61	51.33	51.94	52.53	53.83	52.38	56.84
JUG	37.62	42.32	42.01	45.42	48.27	40.88	42.64	48.60	53.40	52.78	52.44	54.23	54.38	56.56
stats (iter-10)	47.49	48.10	48.68	49.71	50.09	47.72	55.48	51.87	54.30	56.43	55.67	56.52	55.88	59.43
stats w/ GPT2 (iter-10)	52.81	53.77	54.11	54.24	55.27	50.21	57.31	53.91	56.62	59.15	56.97	58.40	59.00	60.36
stats (iter-20)	54.09	56.72	56.25	57.87	57.46	53.24	57.30	54.78	59.83	60.16	58.39	61.96	61.50	62.81
+ step-1	54.33	56.28	57.39	57.62	58.51	57.49	57.28	59.11	60.03	61.43	61.00	62.93	63.89	64.79
+ seed data	56.71	58.68	**59.42**	60.63	61.98	60.75	60.48	60.25	63.75	**64.55**	63.21	63.97	66.95	66.59
+ seed data & step-2 (**Ours**)	**60.35**	**61.44**	59.10	**62.32**	**63.71**	**62.55**	**61.33**	**62.63**	**64.33**	64.34	**65.33**	**67.17**	**68.89**	**68.27**
	MR → Text (NLG)							Text → MR (NLU)						
MR-only	100	150	200	250	300	SR	VE	100	150	200	250	300	SR	VE
seq2seq	31.05	32.92	33.53	36.27	35.09	31.43	30.04	48.61	52.33	52.94	53.53	54.83	53.38	57.84
JUG	35.05	36.84	36.66	39.61	38.5	35.96	33.17	51.02	55.26	55.74	55.07	55.92	55.68	59.14
stats (iter-10)	38.70	40.08	40.42	43.18	41.76	42.36	40.35	53.74	56.95	58.43	57.28	58.78	57.77	60.61
stats w/ GPT2 (iter-10)	41.84	43.20	43.83	46.27	45.67	45.64	43.60	56.24	59.06	61.15	58.29	61.16	60.69	62.59
stats (iter-20)	45.69	46.40	46.84	49.82	49.52	49.17	46.87	57.46	61.26	62.21	60.36	63.98	62.58	64.88
+ step-1	49.66	50.40	50.74	53.09	53.26	52.40	49.93	60.14	62.69	63.38	62.71	65.53	65.15	66.19
+ seed data	53.60	**55.19**	54.32	56.24	57.03	55.92	53.15	62.16	**65.74**	65.65	64.93	66.86	67.99	68.41
+ seed data & step-2 (**Ours**)	**54.77**	54.92	**57.78**	**58.41**	**61.32**	**56.68**	**57.39**	**64.47**	65.38	**67.10**	**67.51**	**69.12**	**70.04**	**69.42**

Table 1: Ablation studies for text generation/NLG (BLEU-4) and slot filling/NLU (F1) on the E2E corpus with increasing amounts of manually-annotated data (100-300 samples). We show the performance increase to the base model initialized from the rule set (**stats**) as **GPT2** augmentation, statistical NLG/NLU models with distillation from **stats** (**step-1**), and dually-regularized sample selection (**step-2**) are added. Domain revisions are performed with 300 data instances. We train the following on the seed data for comparison: (1) a semi-supervised baseline, **JUG** (Tseng et al., 2020) and (2) a LSTM-based baseline (**seq2seq**).

		Slot Filling		Text Generation			
		F1(%)	Wrong	BLEU4	Naturalness	Wrong	Diversity
E2E	reference	-	-	-	4.51	0	53.89
	SLUG+100%	-	-	55.30	4.37	7	46.72
	JUG+100%	73.7	29	57.72	4.49	38	46.21
	seq2seq+100%	73.19	31	56.1	4.32	35	43.09
	GPT2	54.83	55	40.84	4.23	65	**44.55**
	Heuristic	62.81	39	53.08	3.82	**19**	31.37
	COACH	48.35	49	-	-	-	-
	seq2seq+Snorkel	60.71	43	-	-	-	-
	seq2seq+**Ours**	**66.42**	**36**	**54.62**	**4.39**	22	40.65
Weather	reference	-	-	-	4.30	0	40.97
	JUG+100%	67.09	11	51.43	3.30	14	33.61
	seq2seq+100%	66.43	8	46.29	4.10	9	35.74
	GPT2	36.51	46	34.01	**3.95**	35	**40.28**
	Heuristic	50.33	29	38.83	3.40	**16**	31.45
	COACH	46.21	32	-	-	-	-
	seq2seq+Snorkel	47.61	27	-	-	-	-
	seq2seq+**Ours**	**54.71**	**22**	**44.63**	3.80	23	36.76

Table 2: **Performance** and **human evaluation** comparing **Ours** with the benchmarks in the **text-only** scenario with 300 training samples evaluated on the test samples. We count the number of *wrong* slot-value pairs; and *naturalness* is based on average of 15 human ratings on a scale of 5. *Diversity* is the mean segmental type-token ratio (size=25) (Covington and McFall, 2010). **100%** indicates models trained on 100% manual annotation (also **highlighted in gray**). **A+B** indicates training the **A** model on the data generated by approach **B**.

scores are averaged over 10 random initialization runs. Two *subject-matter experts* are employed to construct labeling functions given labeled instances for 1-hour of labeling time. The labeling functions obtained were used for all subsequent scenarios.

Experimental Scenarios. We conduct experiments on two few-shot scenarios (see Table 1): **Text-only** consists of only unlabeled text; **MR-only** is given unlabeled MR alone. Both scenarios are given a small amount of clean, manual-annotated data consisting of MR-text pairs.

To demonstrate the framework's ability to perform *domain revisions* without relabeling, we explore two situations under the few-shot settings: (1) in slot removal (**SR**), we remove the "*customer rating*" slot by selecting from the original seed/dev/test sets. (2) in slot value enhancement (**VE**), we introduce additional restaurant names through relabeling the data. We release the data alongside our code. Note that we selected up to 300 training samples so as to simulate a low resource scenario. We display some examples of **SR** and **VE** in Figure 3.

We compare the performance of our framework with additional benchmark systems on both E2E and Weather datasets in Table 2. Note that our framework is only given the seed data and the additional unlabeled *text* or *MR* samples, while some models are trained with up to 100% of the data. The NLU benchmark systems include a baseline sequence-to-sequence model (**seq2seq**), and **COACH** (Liu et al., 2020) and a baseline data programming framework (**Snorkel**) (Ratner et al., 2017), both state-of-the-art systems on few-shot settings. For NLG, we included a **heuristic** labeler, a **GPT2** labeler (Harkous et al., 2020), and the high-performing SLUG (Juraska et al., 2018) on the E2E data. The **Heuristic** labeler was built on top of the labeling functions, but was extended to be a complete generative system. To compare with models capable of performing both NLG

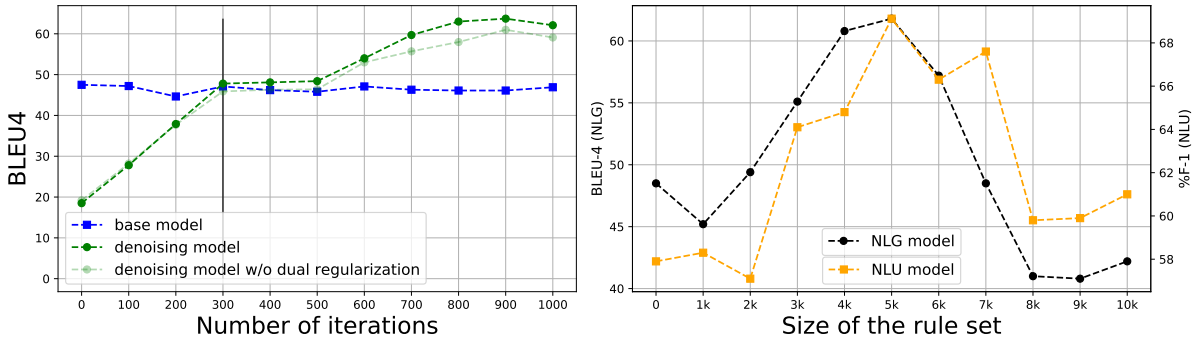

Figure 4: **Left:** This plot indicates the process of *knowledge distillation* (or gain in performance in BLEU-4) from the statistical *base models* to the *NLG model* as a function of the number of iterations. **Vertical line** indicates the addition of clean data. **Right:** we display the *modeling advantage* which showcase the improvement in performance as a function of the number of labeling functions. Both plots are based on 300 clean data.

MR & Text
Slot Removal (SR): **[MR]** name[the rice boat], food[italian], priceRange[cheap], area[riverside], familyFriendly[no], customerRating[average], near[express by holiday inn] **[Text]** the rice boat is located near express by holiday inn in riverside that serves italian food at a low price range .
Slot Value Enhancement (VE): **[MR]** name[the rice boat], food[italian], priceRange[cheap], area[riverside], familyFriendly[no], customerRating[average], near[cobalt lane 32092] **[Text]** the rice boat is located at the cobalt lane 32092, which is next to the riverside that serves italian food at a low price range .

Table 3: Samples of heuristically-based annotation for the revised domain for **SR** and **VE**.

and NLU, we include JUG (Tseng et al., 2020), which is a semi-supervised multi-task framework that allows to perform inference on both NLU and NLG. Table 2 contrasts the performance between **seq2seq** trained on 100% human annotation (**seq2seq+**100%) and of the data generated by our framework (**seq2seq+Ours**).

7 Results and Analysis

The results shown in tables 1 and 2 demonstrate the flexibility of our framework to perform annotation in both *text-only* and *MR-only* scenarios. Moreover, we see in Table 2 that, with as little as 300 data points, the framework is able to produce quality data[11] that allow the baseline **seq2seq** model to come close to the performance of the *same model* trained on full manual annotations; the combination reaches within 1.48 BLEU on NLG and 6.42% F1 score for NLU. Moreover, the framework produces high quality data that effectively mitigate the noise induced by automatic weak annotation, and manages to generate *natural* and *diverse* text for NLG purposes.

[11]The train-dev-test samples are 30-100-100 for the slot manipulation.

In Table 1, we first observe that adding GPT-2 augmentation does diversify the text and improves performance on both datasets maximally by 4.53 BLEU. The augmented system is used to initialize the base model's next iteration, and thus observe that base models can be iteratively enhanced. This is reflected across different sizes of seed data; the effect of iterative denoising is most prominent with seed data size$=$ 300. Next, we see that the use of denoising helps to further improve the models, as it allows to learn from both the base model's initialization and from the effect of noisy labeling. The base model, in this case, serves as the *teacher model* that guides the denoising models to iteratively improve. As the knowledge is completely distilled, we see that the denoised data performs slightly better than the base model (see left of Figure 4), having learned to search through the space of decoding for more optimal paths. We find that, after 20 iterations, the improvements become marginal for both datasets. Thus, we end our experiments at *iter=20*. We also include the expert-labeled (seed) data during denoising as an addition to the large set of pseudo-labeled (noisy) data. This brings about maximally a 3.47 BLEU (for NLG)

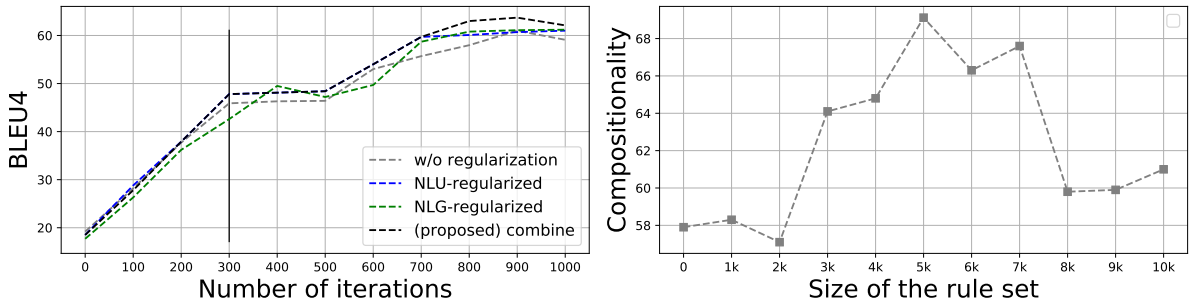

Figure 5: **Left:** This plot indicates the effect of dually-regularized sample selection. **Right:** The plot displays the trade-off between rule coverage and compositionality, where *compositionality* is defined as the average percentage of rules that correctly correlate MR and text over all slots.

and 3.72% F1 score improvement (for NLU) in **Text-only**, and 3.77 BLEU (for NLG) and 2.84% F1 score increase (for NLU) on the **MR-only** scenario. With the proposed dually-regularized sample selection, we further boost the performance by up to 4.29 BLEU and 2.58% F1 and an average of 2.37 BLEU improvements for NLG. This shows the efficacy of the proposed approach in modulating the effect of noisy and clean data.

Since the **Text-only** scenario consists of high-quality manually-labeled text, it generally performs better in NLG; similarly, in the **MR-only** scenario, NLU performance is generally better as MR samples are ground-truth labels. However, this effect is less prominent in the **MR-only** scenario as the difference between ground-truth MR samples and the weakly-labeled ones are often negligible.

Error Analysis. Word-level overlapping scores (BLEU-4) usually correlate rather poorly with human judgements on fluency and information accuracy (Reiter and Belz, 2009; Novikova et al., 2017a). Thus, we perform human evaluation on the E2E corpus on 100 sampled generation outputs, as seen in Table 2. We show that, with 100 instances, the denoising models yield significantly fewer *wrong* slot errors, while having more *natural* and *diverse* outputs. Moreover, we observe that benchmark systems (e.g. COACH) fail to generalize from the small seed data, and suffers heavily in terms of using the *wrong* facts (or slots).

8 Further Analysis

Analysis of Modeling Advantage. We further explore the relationship between performance and the number of labeling functions in the right plot of Figure 4. At one extreme, very few number of labeling functions will result in a very noisy set of rules, which leads to poorly labeled data. We find that, as the number of labeling functions grow, the *capability of the framework to denoise* the initial inferred rules improves. This continues until eventually the framework's denoising capability reaches its peak and starts to deteriorate – as some labeling functions eliminate a useful subset of alignments, as represented by the rule set.

Analysis of Dual Regularization. To analyze the process of sample selection during the training of denoising models, we experiment with selecting samples based on the cross-entropy loss from (1) NLU model, (2) NLG model or (3) the combined use of NLU and NLG models for sample selection, which is the proposed approach. We also compare them with performance without sample selection to show the contrast. In Figure 5 we show this comparison. We observe that selection based on either NLU or NLG model is not sufficient to match the performance of selection using both models. This shows that it is crucial to ensure that the NLG and NLU models learn at approximately the same rate, thereby allowing the semantic alignments induced from the base models to be preserved.

On *Rule Coverage* and *Compositionality* Trade-Offs. As discussed in section 4.4, we evaluate the framework limitations in composing semantic alignments (*compositionality*) as the number of rules becomes high. In Figure 5, we show that the number of rules influence the proportion of rules that correctly align MR and text, as indicated by the percentage of compositionality. In particular, with no labeling functions, the entire rule set is used and this leads to poor performance as most slots are being incorrectly aligned. As more labeling functions are introduced to reduce this rule set to its useful subset, the models begin to construct

better data with the right alignments. The labeling functions thus serve to remove the incorrect, low-impact rules, so that the high impact rules can play a greater role in constructing the necessary semantic alignments for both NLU and NLG – until too little rules remain to construct the base model.

9 Limitations

Overall, while we observe effectiveness in the proposed approach to recreate data, it remains to say that the constructed texts suffer from two main drawbacks: First, the diversity of the text is rather limited by the original seed set, which in turn constrains the data augmentation process that intend to enrich the text diversity. Second, the process of creating programmable labeling functions can indeed be a cumbersome process – it relies heavily on the adequate skill sets of the annotators who need to understand the target domain and basic scripting in order to proceed. It is then vital to ease the process of programming script writing, and reuse functions as much as possible to avoid overheads.

10 Conclusion and Future Work

In this paper, we show the efficacy of the framework where both NLU and NLG data can be jointly and automatically labeled to construct high quality data. We also demonstrate that the framework is receptive to the changes in MR slots, allowing for automatic domain revisions of MR and text data. Importantly, we observe that the success of the framework depends on finding the right balance between the number of labeling functions and the inherent level of compositionality of the data to be labeled. Thus, for future work we intend to focus on identifying the level of compositionality and predicting the threshold number of labeling functions necessary for decent performance, potentially manipulating the inherent graphical relationships (Hong et al., 2019). Moreover, the initial seed set in our experiments are assumed to be present, it is therefore necessary to first sample unlabeled data based on difficulty to annotate and the performance considerations (Chang et al., 2021b,c), before fine-tuning with pretrained language models which have strong priors for better quality data (Chang et al., 2022b,a).

Acknowledgements

This research was funded in part by the German Research Foundation (DFG) as part of SFB 248 "Foundations of Perspicuous Software Systems". We sincerely thank the anonymous reviewers for their insightful comments that helped us to improve this paper.

References

Mikel Artetxe, Gorka Labaka, and Eneko Agirre. 2018. Unsupervised statistical machine translation. *arXiv preprint arXiv:1809.01272*.

Mikel Artetxe, Gorka Labaka, Eneko Agirre, and Kyunghyun Cho. 2017. Unsupervised neural machine translation. *arXiv preprint arXiv:1710.11041*.

Stephen H Bach, Bryan He, Alexander Ratner, and Christopher Ré. 2017. Learning the structure of generative models without labeled data. *Proceedings of machine learning research*, 70:273.

Anusha Balakrishnan, Jinfeng Rao, Kartikeya Upasani, Michael White, and Rajen Subba. 2019. Constrained decoding for neural nlg from compositional representations in task-oriented dialogue. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 831–844.

Regina Barzilay and Mirella Lapata. 2005. Modeling local coherence: An entity-based approach. In *Proceedings of the 43rd Annual Meeting of the Association for Computational Linguistics (ACL'05)*, pages 141–148, Ann Arbor, Michigan. Association for Computational Linguistics.

Peter F Brown, Stephen A Della Pietra, Vincent J Della Pietra, and Robert L Mercer. 1993. The mathematics of statistical machine translation: Parameter estimation. *Computational linguistics*, 19(2):263–311.

Paweł Budzianowski, Tsung-Hsien Wen, Bo-Hsiang Tseng, Iñigo Casanueva, Stefan Ultes, Osman Ramadan, and Milica Gasic. 2018. Multiwoz-a large-scale multi-domain wizard-of-oz dataset for task-oriented dialogue modelling. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 5016–5026.

Ernie Chang, Jesujoba Alabi, David Adelani, and Vera Demberg. 2022a. Few-shot pidgin text adaptation via contrastive fine-tuning. In *Proceedings of COLING 2022*.

Ernie Chang, Vera Demberg, and Alex Marin. 2021a. Jointly improving language understanding and generation with quality-weighted weak supervision of automatic labeling. In *EACL 2021*.

Ernie Chang, Alex Marin, and Vera Demberg. 2022b. Improving zero-shot multilingual text generation via iterative distillation. *Proceedings of COLING 2022*.

Ernie Chang, Xiaoyu Shen, Alex Marin, and Vera Demberg. 2021b. The selectgen challenge: Finding the best training samples for few-shot neural text generation. In *Proceedings of the 14th INLG*, pages https–aclanthology.

Ernie Chang, Xiaoyu Shen, Hui-Syuan Yeh, and Vera Demberg. 2021c. On training instance selection for few-shot neural text generation. In *Proceedings of ACL 2021*.

Oishik Chatterjee, Ganesh Ramakrishnan, and Sunita Sarawagi. 2019. Data programming using continuous and quality-guided labeling functions. *arXiv preprint arXiv:1911.09860*.

Alice Coucke, Alaa Saade, Adrien Ball, Théodore Bluche, Alexandre Caulier, David Leroy, Clément Doumouro, Thibault Gisselbrecht, Francesco Caltagirone, Thibaut Lavril, et al. 2018. Snips voice platform: an embedded spoken language understanding system for private-by-design voice interfaces. *arXiv preprint arXiv:1805.10190*.

Michael A Covington and Joe D McFall. 2010. Cutting the gordian knot: The moving-average type–token ratio (mattr). *Journal of quantitative linguistics*, 17(2):94–100.

Arthur P Dempster, Nan M Laird, and Donald B Rubin. 1977. Maximum likelihood from incomplete data via the em algorithm. *Journal of the Royal Statistical Society: Series B (Methodological)*, 39(1):1–22.

Sara Evensen, Chang Ge, Dongjin Choi, and Çağatay Demiralp. 2020. Data programming by demonstration: A framework for interactively learning labeling functions. *arXiv preprint arXiv:2009.01444*.

Esther Galbrun. 2009. *Phrase table pruning for statistical machine translation*. Ph.D. thesis, University of Helsinki.

Hamza Harkous, Isabel Groves, and Amir Saffari. 2020. Have your text and use it too! end-to-end neural data-to-text generation with semantic fidelity. *arXiv preprint arXiv:2004.06577*.

Xudong Hong, Ernie Chang, and Vera Demberg. 2019. Improving language generation from feature-rich tree-structured data with relational graph convolutional encoders. In *Proceedings of the 2nd Workshop on Multilingual Surface Realisation (MSR 2019)*, pages 75–80.

Zhiting Hu, Xuezhe Ma, Zhengzhong Liu, Eduard Hovy, and Eric Xing. 2016. Harnessing deep neural networks with logic rules. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 2410–2420.

Lu Jiang, Zhengyuan Zhou, Thomas Leung, Li-Jia Li, and Li Fei-Fei. 2018. Mentornet: Learning data-driven curriculum for very deep neural networks on corrupted labels. In *International Conference on Machine Learning*, pages 2304–2313.

Juraj Juraska, Panagiotis Karagiannis, Kevin Bowden, and Marilyn Walker. 2018. A deep ensemble model with slot alignment for sequence-to-sequence natural language generation. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 152–162.

John F Kelley. 1984. An iterative design methodology for user-friendly natural language office information applications. *ACM Transactions on Information Systems (TOIS)*, 2(1):26–41.

Diederik P Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*.

John Lafferty, Andrew McCallum, and Fernando CN Pereira. 2001. Conditional random fields: Probabilistic models for segmenting and labeling sequence data.

Guillaume Lample, Alexis Conneau, Ludovic Denoyer, and Marc'Aurelio Ranzato. 2017. Unsupervised machine translation using monolingual corpora only. *arXiv preprint arXiv:1711.00043*.

Guillaume Lample, Myle Ott, Alexis Conneau, Ludovic Denoyer, and Marc'Aurelio Ranzato. 2018. Phrase-based & neural unsupervised machine translation. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 5039–5049.

Tao Li and Vivek Srikumar. 2019. Augmenting neural networks with first-order logic. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 292–302.

Zihan Liu, Genta Indra Winata, Peng Xu, and Pascale Fung. 2020. Coach: A coarse-to-fine approach for cross-domain slot filling. *arXiv preprint arXiv:2004.11727*.

Manuel Mager, Ramón Fernandez Astudillo, Tahira Naseem, Md Arafat Sultan, Young-Suk Lee, Radu Florian, and Salim Roukos. 2020. Gpt-too: A language-model-first approach for amr-to-text generation. *arXiv preprint arXiv:2005.09123*.

Jekaterina Novikova, Ondřej Dušek, Amanda Cercas Curry, and Verena Rieser. 2017a. Why we need new evaluation metrics for nlg. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 2241–2252.

Jekaterina Novikova, Ondřej Dušek, and Verena Rieser. 2017b. The e2e dataset: New challenges for end-to-end generation. In *Proceedings of the 18th Annual SIGdial Meeting on Discourse and Dialogue*, pages 201–206.

Franz Josef Och. 2003. Minimum error rate training in statistical machine translation. In *Proceedings of the 41st Annual Meeting of the Association for*

Computational Linguistics, pages 160–167, Sapporo, Japan. Association for Computational Linguistics.

Franz Josef Och and Hermann Ney. 2003. A systematic comparison of various statistical alignment models. *Computational linguistics*, 29(1):19–51.

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. Bleu: a method for automatic evaluation of machine translation. In *Proceedings of the 40th annual meeting on association for computational linguistics*, pages 311–318. Association for Computational Linguistics.

Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, et al. 2019. Pytorch: An imperative style, high-performance deep learning library. In *Advances in Neural Information Processing Systems*, pages 8024–8035.

Emmanouil Antonios Platanios, Otilia Stretcu, Graham Neubig, Barnabas Poczos, and Tom Mitchell. 2019. Competence-based curriculum learning for neural machine translation. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 1162–1172.

Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. 2019. Language models are unsupervised multitask learners. *OpenAI Blog*, 1(8).

Alexander Ratner, Stephen H Bach, Henry Ehrenberg, Jason Fries, Sen Wu, and Christopher Ré. 2017. Snorkel: Rapid training data creation with weak supervision. *Proceedings of the VLDB Endowment*, 11(3):269–282.

Alexander Ratner, Braden Hancock, Jared Dunnmon, Frederic Sala, Shreyash Pandey, and Christopher Ré. 2019. Training complex models with multi-task weak supervision. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, pages 4763–4771.

Alexander J Ratner, Christopher M De Sa, Sen Wu, Daniel Selsam, and Christopher Ré. 2016a. Data programming: Creating large training sets, quickly. In *Advances in Neural Information Processing Systems 29*.

Alexander J Ratner, Christopher M De Sa, Sen Wu, Daniel Selsam, and Christopher Ré. 2016b. Data programming: Creating large training sets, quickly. *Advances in neural information processing systems*, 29:3567–3575.

Ehud Reiter and Anja Belz. 2009. An investigation into the validity of some metrics for automatically evaluating natural language generation systems. *Computational Linguistics*, 35(4):529–558.

Ehud Reiter and Robert Dale. 2000. *Building natural language generation systems*. Cambridge university press.

Esteban Safranchik, Shiying Luo, Stephen H Bach, Elaheh Raisi, Stephen H Bach, Stephen H Bach, Daniel Rodriguez, Yintao Liu, Chong Luo, Haidong Shao, et al. 2020. Weakly supervised sequence tagging from noisy rules. In *AAAI*, pages 5570–5578.

Martin Schmitt and Hinrich Schütze. 2019. Unsupervised text generation from structured data. *arXiv preprint arXiv:1904.09447*.

Yanyao Shen and Sujay Sanghavi. 2019a. Iterative least trimmed squares for mixed linear regression. In *Advances in Neural Information Processing Systems*, volume 32, pages 6078–6088. Curran Associates, Inc.

Yanyao Shen and Sujay Sanghavi. 2019b. Learning with bad training data via iterative trimmed loss minimization. In *Proceedings of the 36th International Conference on Machine Learning*, pages 5739–5748.

Shang-Yu Su, Chao-Wei Huang, and Yun-Nung Chen. 2020. Towards unsupervised language understanding and generation by joint dual learning. *ACL*.

Bo-Hsiang Tseng, Jianpeng Cheng, Yimai Fang, and David Vandyke. 2020. A generative model for joint natural language understanding and generation. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 1795–1807.

Paroma Varma, Frederic Sala, Ann He, Alexander Ratner, and Christopher Ré. 2019. Learning dependency structures for weak supervision models. *arXiv preprint arXiv:1903.05844*.

Tsung-Hsien Wen, David Vandyke, Nikola Mrkšić, Milica Gasic, Lina M Rojas Barahona, Pei-Hao Su, Stefan Ultes, and Steve Young. 2017. A network-based end-to-end trainable task-oriented dialogue system. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 1, Long Papers*, pages 438–449.

Yu Wu, Wei Wu, Zhoujun Li, and Ming Zhou. 2018. Learning matching models with weak supervision for response selection in retrieval-based chatbots. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 420–425.

Richard Zens, Daisy Stanton, and Peng Xu. 2012. A systematic comparison of phrase table pruning techniques. In *Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning*, pages 972–983.

Text-to-Text Extraction and Verbalization of Biomedical Event Graphs

Giacomo Frisoni*, **Gianluca Moro*** and **Lorenzo Balzani***
Department of Computer Science and Engineering (DISI)
University of Bologna, Via dell'Università 50, I-47522 Cesena, Italy
{giacomo.frisoni,gianluca.moro}@unibo.it
balzanilo@icloud.com

Abstract

Biomedical events represent complex, graphical, and semantically rich interactions expressed in the scientific literature. Almost all contributions in the event realm orbit around semantic parsing, usually employing discriminative architectures and cumbersome multi-step pipelines limited to a small number of target interaction types. We present the first lightweight framework to solve both event extraction and event verbalization with a unified text-to-text approach, allowing us to fuse all the resources so far designed for different tasks. To this end, we present a new event graph linearization technique and release highly comprehensive event-text paired datasets, covering more than 150 event types from multiple biology subareas (English language). By streamlining parsing and generation to translations, we propose baseline transformer model results according to multiple biomedical text mining benchmarks and natural language generation metrics. Our extractive models achieve greater state-of-the-art performance than single-task competitors and show promising capabilities for the controlled generation of coherent natural language utterances from structured data.[1]

1 Introduction

In recent years, events have become an influential formalism for modeling complex relations mentioned within the text as semantic graphs (Frisoni et al., 2021, 2022). In bioinformatics, an event generally refers to an interaction between one or more biomedical entities (e.g., proteins, genes, diseases, drugs), each contributing with a specific role (e.g., Theme, Cause, Site). For instance, biomedical events include molecular reactions, organism-level outcomes, and adverse drug reactions. Their expressive power and flexibility have supported

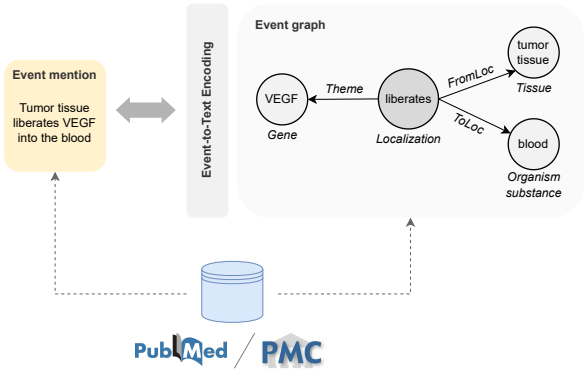

Figure 1: Illustration of an event graph and its textual mention from our datasets. All text-event pairs refer to human-crafted annotations above the biomedical literature (abstracts or full papers).

many practical applications like literature-based knowledge discovery (Wang et al., 2021b), biological network construction (Björne et al., 2010), diagnosis prediction (Zhang et al., 2020c), document summarization (Zhang et al., 2020b), and question answering (Berant et al., 2014).

Text-to-event (or event extraction, EE) and event-to-text (or event graph verbalization, EGV) systems effectively bridge natural language and symbolic representations. They provide a step towards decoupling concept units (*what to say*) from language competencies (*how to say it*) (Mel'čuk, 1973). Strongly linked to natural language understanding, EE is a fundamental task to automatically identify, monitor, and aggregate the relational knowledge disseminated within life science papers, speeding up medical progress and promoting discoveries. Yet, although much attention has been paid to EE, no research efforts have been directed to its inverse task, namely EGV. Even if under-explored, EGV targets the generation of informative text constrained on semantic graphs, holding a lot of potential in applications like conversational agents and summarization systems (Frisoni. et al., 2022; Moro and Ragazzi, 2022; Moro et al., 2022).

*Equal contribution.

[1]The data and the code to reproduce our baseline results are available at https://github.com/disi-unibo-nlp/bio-ee-egv

Proceedings of the 29th International Conference on Computational Linguistics, pages 2692–2710
October 12–17, 2022.

Most state-of-the-art (SOTA) approaches handle structured prediction by employing task-specific architectures and discriminative models. Ordinarily, they need to be adapted to the target events and their schema, or they are not flexible enough to work with different domains or analysis granularities (sentence- vs document-level). The extraction process is typically divided into subtasks, executed in a pipeline or joint manner, where the output of several classifiers needs to be integrated. Additionally, each task is often associated with its own output space, limiting knowledge sharing and multi-task learning (MTL). Classes (i.e., event, argument role, and entity types) are specified implicitly through numerical indices, and models contain no prior information about their meaning. Furthermore, modern deep learning solutions require a non-trivial amount of examples to train, but event annotations are expensive to produce—a discrepancy that results in multiple, stand-alone, and closed-domain datasets with few records and potentially overlapping labels (Miwa et al., 2013).

On a parallel track, transfer learning has been the pinnacle of the latest breakthroughs in natural language processing (NLP). Large pre-trained language models (PLMs) are powerful backbones that can be fine-tuned for different tasks to achieve impressive performance in wide-ranging applications (Kalyan et al., 2021). PLMs capture contextual information and latent linguistic/relational knowledge (Petroni et al., 2019; Roberts et al., 2020), incorporating syntax and semantics. In that sense, textual representations and conditional generative modeling can be seen as natural ways of encoding different events in a shared predictive space.

In this paper, we design a framework to solve both EE and EGV as text-to-text problems, thus leveraging SOTA PLMs and disposing of the need for complex and hardly adaptable architectures. Concretely, we propose a way to decompose events into text sequences, neatly preserving structure and labels. Above it, we present the Biomedical Text-to-Event (BIOT2E) and Event-to-Text (BIOE2T) datasets, two corpora of textualized biomedical event graphs paired with their mention. Precisely, we aggregate and preprocess gold annotations coming from 10 popular EE benchmarks, intending to systematize the community work matured with public evaluation programs and solving the low coverage issue. Among the exciting multimodal opportunities enabled by these datasets, we show that out-of-the-box transformer models can effectively learn text $\rightarrow$ event and event $\rightarrow$ text translations (Figure 1). We achieve this symmetry by using the same architecture for parsing and generation, as well as for all event instances, originally belonging to separate EE tasks with independent output spaces. To the best of our knowledge, this is the first study to handle such a variety of event schema without distinct models or additional task-specific modules. Our key contributions are the following:

1. We devise a novel event linearization with a consistent textual output format based on formal grammar (§3).

2. We introduce BIOT2E and BIOE2T, two large-scale biomedical event-text aligned datasets designed to frame the extraction and verbalization of general biomedical events as text-to-text tasks (§4).

3. We experiment EE, EGV, and MTL (§5 and §6). We demonstrate that autoregressive seq2seq models can achieve SOTA performance—previously attained only by discriminative solutions—while being much more flexible and scalable.

2 Related Work

Graph-Text Paired Data. Many graph-text paired datasets have sprung up. Nevertheless, annotating text or semantic graphs is expensive, especially for specific fields like biology. Most of the resources are domain-general and focus on knowledge graphs (KGs). Although there are datasets assembled by crowdsourced human annotators—such as WebNLG (Gardent et al., 2017), one common thread is using NLP tools and automatic alignment heuristics to forge silver pairs massively, e.g., mapping Wikipedia sentences to Wikidata triples (Elsahar et al., 2018; Agarwal et al., 2021) or Wikipedia paragraphs to Freebase subgraphs (Wang et al., 2021a). Predicate linkers and PLMs are already used to inherently construct KGs from the biomedical literature (Geleta et al., 2021), but the relations extracted for each document are generally not openly released. In contrast, we present the first datasets directly pairing scientific sentences to biomedical event graphs, usable as evaluation gold standards thanks to expert user provenance.

Event Extraction. In the NLP field, EE is placed within the more general information extraction (IE)

and structured prediction (SP) areas. Specifically, it aims to interpret and distill free-text chunks into structured, semantic, and fine-grained relations capturing an interplay between many different participants (entities or other events) usually subjected to a state change. EE requires to recognize *triggers* (text spans that clearly testify the occurrence of a real-world event), classify the type of the events for which they act as lawyers, detect involved 0-N *arguments* (entity mentions and corresponding classes, or sub-event triggers), predict their *semantic role*, and establish some optional event-level *modifiers*. For example, a Localization event is indicated in Figure 1 at "liberates", involving three bio-entities. Notably, end-to-end EE systems usually integrate named entity recognition (NER) and coreference resolution. Compared to the more traditional binary relation extraction, where the goal is deriving subject-relation-object triplets, EE is a more complex task that needs to deal with high-level linguistic phenomena, an avalanche of narrative styles, and syntactic constructions.

Early approaches tackled EE with pipeline architectures to decompose the problem in its sequential subtasks and independently train a classifier for each of them, also relying on gold-tagged entities to eventually ignore NER objectives. Historically, first attempts made use of pattern-based techniques (Cohen et al., 2009) or data-driven methods centered on generalizing classical machine learning algorithms to SP, including, among others, support vector machines (Miwa et al., 2012). More recently, joint and MTL architectures have gained popularity among researchers, training a single model on all EE sub- and linked-tasks simultaneously, benefitting from information sharing and mutually improving local predictions. Deep learning is the main architect of this transition, with many EE efforts rooted in transformers (Ramponi et al., 2020), convolutional (Björne and Salakoski, 2018), recurrent (Li et al., 2019), and graph (Zhao et al., 2021) neural networks (CNNs, RNNs, GNNs). Current SOTA EE solutions train end-to-end neural models on top of the features learned by domain-specific PLMs, such as SciBERT (Beltagy et al., 2019). In this line of work, DeepEventMine (Trieu et al., 2020) presently holds leading performance on most biomedical EE (BEE) benchmarks, with custom discriminative classification layers above SciBERT-encoded intra-sentence spans. Most BEE systems work within the sentence scope, not being able to scale to entire documents and facts with scattered arguments. Our framework is designed for joint EE, also including the NER subtask[2], and is not limited to sentence-level extraction in principle.

Data-to-Text. Data-to-text is the task of generating natural language text conditioned on source content provided in the form of structured data. Different GNNs have been proposed to better encode the input structure in the case of graphs, like Graph Transformers (Koncel-Kedziorski et al., 2019) and DualEnc for KG-triples ordering and verbalization (Zhao et al., 2020). On the other hand, recent works (Kale and Rastogi, 2020; Wang et al., 2021c; Agarwal et al., 2021) have favored seq2seq pre-trained models—with T5 (Raffel et al., 2020) as prominent example—which showcased better grammatical correctness and domain-shift robustness. To the extent of our knowledge, no prior research has attempted to verbalize event graphs. In this paper, we start from these heated evidences to fill the gap.

Seq2seq for Structured Prediction and Graph Verbalization. It has become increasingly popular to cast structured prediction problems as translations between natural languages, linearizing data when necessary and leveraging the transfer learning capacity of a transformer-based PLM. Text-to-text reframing has been applied to many contexts, from general NLP tasks (Raffel et al., 2020) and semantic role labeling (Blloshmi et al., 2021) to relation extraction (Huguet Cabot and Navigli, 2021). Closer to us, TANL (Paolini et al., 2021) and TEXT2EVENT (Lu et al., 2021b) are the only works carrying out this strategy on EE. However, the authors solely consider ACE2005 and ERE, two simplistic newswire datasets with a small type coverage and flat target structures. More importantly, TANL encodes event annotations in the form of augmented text, dividing EE into different subtasks with the lack of support for nested events, modifiers, or event overlapping, which are instead common in biology, thus being not directly applicable to our datasets. Instead, we solve BEE by generating the output graph at once, supporting complex structures and schema. Outside of our work, symmetric parsing and generation have been chiefly explored

[2]We do not predict relationships between gold entities, as is frequently assumed in other works dependent on extra input annotations or external NER tools that interrupt the backpropagation process. On the contrary, our models are directly trained to recognize target entities and classify their type as a fundamental subtask for the ultimate goal of end-to-end event extraction via text translation.

with AMRs (Konstas et al., 2017; Bevilacqua et al., 2021). The aforementioned publications highlight the relevance of seq2seq models. Not only do they exhibit strong performance, but they also lean on decoding mechanisms rather than predefined type sets, being easily extendable to new or unseen inputs. We also underline that, by conditioning future decoding on previous generations, they implicitly deal with dependencies among graph records (i.e., non-atomic extractions).

3 Event Linearization

Seq2seq models require that both the input and target be presented as a linear sequence of tokens. In this section, we describe our format design concept to reformulate event graphs as strings.

Events have an n-ary and potentially nested structure, with optional modifiers (e.g., "polarity", "certainty level") reshaping the described interaction. Like many other relational data, events can be conveniently and naturally modeled as rooted directed acyclic graphs (Frisoni et al., 2021). With this formalization, triggers and entities are nodes, while argument roles define edges. Each trigger, entity, and trigger-trigger/trigger-entity association is assigned to its type according to a predefined ontology[3].

We revisit the formulation by (Paolini et al., 2021) and put forward a formal event language designed to be easy and deterministically reversible to event graphs. While being more complex to learn, our linearization comes with the advantage of enclosing entire events in single expressions, reducing the overhead of generating different output sequences for trigger and argument annotations. Each node is surrounded by the special tokens `[` and `]`, which represent semantic structure indicators. Inside, a sequence of `|`-separated tags reports the text span, the type (described in natural words), and a list of `X=Y` relations, where `X` is the argument role and `Y` is the target trigger. Note that the same entity can be coupled to different events (triggers) with distinct roles (i.e., double tagging). The root trigger is a source vertex and has not incoming edges. Trigger nodes also specify event-level modifiers as additional `X=Y` assignments, in the form `Property=Value` (e.g., "Polarity=Negative"). So, the information on event components (i.e., nodes and their embedded interconnections) are all within `[...|...]` patterns, which can be nested in case of sub-events. The final string minimizes the number of tokens to be submitted or generated so as to make encoding/decoding more efficient. Event constituents are sorted by their order of appearance in the *.a2* for consistency.

We define a context-free grammar (§A.1) and test it with JFLAP (Rodger and Finley, 2006). Figure 2 depicts a practical example of textualized nested bio-event.

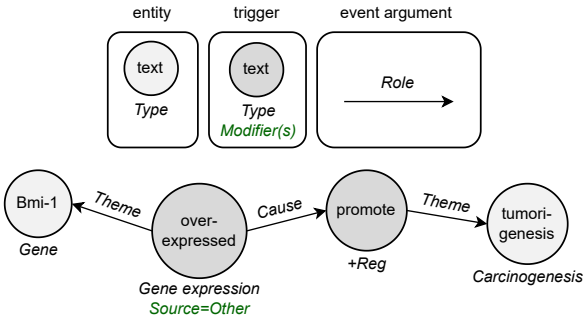

[over-expressed | *Gene_expression* | Source = Other]
[Bmi-1 | *Gene* | Theme = over-expressed]
[[promote | *Positive_regulation* | Cause = over-expressed]
[tumorigenesis | *Carcinogenesis* | Theme = promote]]

Figure 2: Example of textually linearized event graph.

4 Datasets

Based on §3, we build new corpora suitable for text- or graph-conditioned sequence modeling. Here, we present the construction process of BIOT2E and BIOE2T, together with their main properties.

4.1 Construction Process

4.1.1 Data Collection

Obtaining a large gold dataset of jointly annotated pairs of sentences and event graphs may require years of labor (Kim et al., 2008). We overcome this issue by combining the training sets of 10 influential real-world datasets originally designed for BEE, primarily derived from the ongoing BioNLP-ST series (Kim et al., 2019). Table 1 reports the characteristics of the seed datasets used for BIOE2T and BIOT2E construction[4]. These sources comprise seminal bioinformatics projects like GENIA (Kim et al., 2013a) and well-known tasks meeting

[3] In this paper, we refer to closed-domain EE settings. Please note that closed-domain EE exclusively searches for target events (e.g., positive/negative regulation, binding, carcinogenesis) with a defined schema. On the contrary, open-domain EE does not assume specific target types and aims to detect general events unsupervised, thereby being more limited.

[4] We intentionally focused on fusing existing benchmarks to build a highly-comprehensive biomedical evaluation gold standard (not possible with silver pairs forged with heuristics).

biologists' needs, including topics such as cancer genetics (Pyysalo et al., 2013) and infectious diseases (Pyysalo et al., 2011). Each focuses on a particular domain, differs in the annotation schema, and consists of human-curated event annotations on top of PubMed abstracts and full papers (English language). The reader should be aware that biomedical benchmarks generally support only two boolean modifiers—negation and speculation. On the flip side, modifiers are essential for a correct event interpretation, even with instances having the same triggers and arguments. Given the potential and uniqueness of modifiers in event data, we include GENIA-MK (Miwa et al., 2012) to manage more sophisticated forms of meta-knowledge during translation. Details on embraced modifiers are available in §A.2.

	Corpus	Domain(s)	#Documents	Annotation Schema
	Genia Event Corpus (GE08) (Kim et al., 2008)	Humans blood cells transcription factors	1,000 abstracts	35 entity types, 35 event types
BioNLP-ST'11	Genia Event 2011 (GE11) (Kim et al., 2012)	See GE08	1,210 abstracts, 14 full papers	2 entity types, 9 event types, 2 modifiers
BioNLP-ST'11	Epigenetics and Post-translational Modifications (EPI11) (Ohta et al., 2011)	Epigenetic change and common protein post-translational modifications	1,200 abstracts	2 entity types, 14 event types, 2 modifiers
BioNLP-ST'11	Infectious Diseases (ID11) (Pyysalo et al., 2011)	Two-component regulatory systems	30 full papers	5 entity types, 10 event types, 2 modifiers
	Multi-Level Event Extraction (MLEE) (Pyysalo et al., 2012)	Blood vessel development from the subcellular to the whole organism	262 abstracts	16 entity types, 19 event types
	GENIA-MK (Miwa et al., 2012)	See GE08	1,000 abstracts	35 entity types, 35 event types, 5 modifiers (+2 inferable)
BioNLP-ST'13	Genia Event 2013 (GE13) (Kim et al., 2013a)	See GE08	34 full papers	2 entity types, 13 event types, 2 modifiers
BioNLP-ST'13	Cancer Genetics (CG13) (Pyysalo et al., 2013)	Cancer biology	600 abstracts	18 entity types, 40 event types, 2 modifiers
BioNLP-ST'13	Pathway Curation (PC13) (Ohta et al., 2013)	Reactions, pathways, and curation	525 abstracts	4 entity types, 23 event types, 2 modifiers
BioNLP-ST'13	Gene Regulation Ontology (GRO13) (Kim et al., 2013b)	Human gene regulation and transcription	300 abstracts	174 entity types, 126 event types

Table 1: Summary of the biomedical event extraction corpora used for constructing BIOT2E and BIOE2T. All data is in public domain and licensed for research purposes.

4.1.2 Data Preprocessing, Filtering and Sampling

Annotations follow standoff *.a**, *.ann*, or *.xml* formats, where labels are connected to the text spans of the document through $(start, end)$ character offset pairs. We automatically produce the linearized version of each event graph by parsing and normalizing these files, otherwise having structure and labeling variants depending on the original dataset. For example, *.ann* files identify modifiers with "A" instead of "M"; GENIA-MK specifies the value of each property and not the active type only (e.g., "Speculation=True" versus "Speculation"); GRO13 supports the recognition of triggers or entities with scattered text spans (e.g., "RFX ... 3" → "RFX3"). Our encoding formalism constitutes a straightforward approach to control such nuances and unify all EE sources. To force a network to learn the connection between linguistic phenomena and event modifiers, we consistently report the latter in an expanded version, standardizing the names in case of inconsistencies (e.g., "Negation" → "Polarity=Negative"). We eliminate duplicate events, instances with annotation errors (e.g., references to undefined entities) or with nesting cycles. If multiple overlapping linearizations from different datasets correspond to the same event mention, we keep the longest and most complete one.

For BIOE2T (verbalization), we map each textualized graph with its mention. At this juncture, it is essential to clarify that, with the term "event mention", we refer to the complete sentences that describe all the components of a certain event and therefore contain all the offsets related to its triggers and arguments. Note that an event mention (generation target) can be longer than one sentence. Linearizations of events sharing the same text span (i.e., double tagging) are decomposed in multiple records. Poorly represented event types (less than three occurrences) are discarded. Similarly, single-node (trigger only) events are ignored since predicting entire sentences from such a little context would be unreasonable. Using stratified random sampling, we split data in training, validation, and test sets with a 90-5-5 proportion. We stratify on multiple variables: (i) the source dataset; (ii) the event type (the main one in case of nesting, i.e., the graph root); (iii) the event mention length.

BIOT2E (parsing) is specular, except to include a balanced number of negative examples, manage double tagging by concatenating linearized events, and not be filtered, thus enabling 1:N extractions. We map a PubMed sentence to a target linearization, if present, or to an empty string otherwise. Hence, we perform BEE at a sentence level, but we do not exclude the investigation of document granularities in future works thanks to efficient transformers (Tay et al., 2020)—e.g., LongT5 (Guo et al., 2022). By accommodating these steps, we treat originally

distinct tasks as different datasets of the more general BEE task.

4.2 Data Properties

We devote the last part of this section to quantitatively analyze the composition of BIOE2T and BIOT2E. Basic statistics are shown in Table 2. Note that the total number of unique event, entity, role, and modifier types are ∼170, ∼150, 19, and 6, respectively, considerably larger than those in previous standalone corpora (Frisoni et al., 2021). Figure 3 shows the distribution of event graph sizes and mention lengths, skewed with a long tail.

		Train		Valid		Test		All	
# Pairs		61,319		3,407		3,407		68,133	
		36,635		2,035		2,036		40,706	
# Event types		166	168	95	90	96	96	166	170
# Entity types		148	141	81	66	85	72	150	142
# Argument role types		19	19	15	16	15	14	19	19
# Modifier types		6	6	6	6	6	6	6	6
# Nodes per event	min	2	1	2	1	2	1	2	1
	mean	4.29	3.65	4.31	3.66	4.30	3.63	4.30	3.65
	max	35	25	35	20	31	19	35	25
# Modifiers per event	min	0	0	0	0	0	0	0	0
	mean	2.43	2.26	2.44	2.35	2.40	2.25	2.42	2.26
	max	5	5	5	5	5	5	5	5
# Sentences per event mention	min	1		1		1		1	
	mean	1.19		1.19		1.19		1.19	
	max	3		3		3		3	
# Tokens per event mention	min	6	2	8	2	11	2	6	2
	mean	58.55	38.57	58.61	37.67	58.66	39.18	58.56	38.56
	max	301	301	212	161	301	174	301	301
# Events per sentence	min	0		0		0		0	
	mean	1.40		1.39		1.43		1.40	
	max	28		15		24		28	

Table 2: Basic statistics about our BIOE2T and BIOT2E (blue text) datasets.

5 Experimental Setup

In this section, we provide the formal definition of text-to-event parsing and event-to-text generation. Then, describe the setup of the experiments we conducted to evaluate our framework in both tasks.

5.1 Tasks

We see event graph extraction and verbalization as bidirectional transduction tasks via conditional generation, similarly to machine translation.

Training an **event parser** means finding a set of parameters θ_P for a model f that predicts an event graph $\hat{e}$ given a text span s:

$$\hat{e} = \underset{e}{\operatorname{argmax}}\, f(e|s;\theta_P). \tag{1}$$

Training an **event mention generator** require finding a set of parameters θ_G for a model f that predicts a text span $\hat{s}$ given an event graph e:

$$\hat{s} = \underset{s}{\operatorname{argmax}}\, f(s|e;\theta_G). \tag{2}$$

In both cases, we use the same family of predictors f (i.e., architectural symmetry without dataset- or task-dependent modifications) by means of seq2seq models. We focus on two data settings: (i) multiple datasets for the same task (multi-dataset based on BIOT2E and BIOE2T with independent parameters θ_P and θ_G), and (ii) all datasets across different tasks (multi-task with shared parameters).

5.2 Models

Given the above-reframed definition of EE and EGV, we employ encoder-decoder architectures to autoregressively predict the target sequence y conditioned on the input sequence x:

$$p(y|x) = \prod_{i=1}^{|y|} p(y_i|y_{<i}, x), \tag{3}$$

where $y_{<i} = y_1 \dots y_{i-1}$ and $p(y_i|y_{<i}, x)$ is the probability over the target vocabulary $\mathcal{V}$ normalized by softmax$(\cdot)$. Because most of the tokens in linearized event representations are also natural language words, we investigate two PLMs with different capacities, aiming to reuse their general text and world knowledge: T5-Base (Raffel et al., 2020) and BART-Base (Lewis et al., 2020). Details about models, training, and hardware configurations are listed in §A.3. According to our literature review, T5 and BART are the two leading generative models adopted in this field. Basically, they are both transformer-based models (with a subword vocabulary) pre-trained on massive corpora through a denoising self-supervised task, i.e., reconstruction of artificially corrupted spans. T5 comes pre-trained also on a multi-task mixture of text-to-text supervised tasks, but none of these include language generation from structured data. As remarked by other researchers for AMR (Bevilacqua et al., 2021), we hypothesize that denoising pre-training is beneficial for EE and EGV. Linearized events can be seen as reordered and partially corrupted sentences that a model must reconstruct, and vice versa. Given a training dataset $\mathcal{D} = \{(x_1, y_1), \dots, (x_{|D|}, y_{|D|})\}$, the learning objective is the negative log-likelihood (teacher forcing):

$$\mathcal{L} = -\sum_{(x,y)\in\mathcal{D}} \log p(y|x,\theta). \tag{4}$$

5.3 Evaluation

Parsing. While training is based on a likelihood objective, we assess EE models using standard pre-

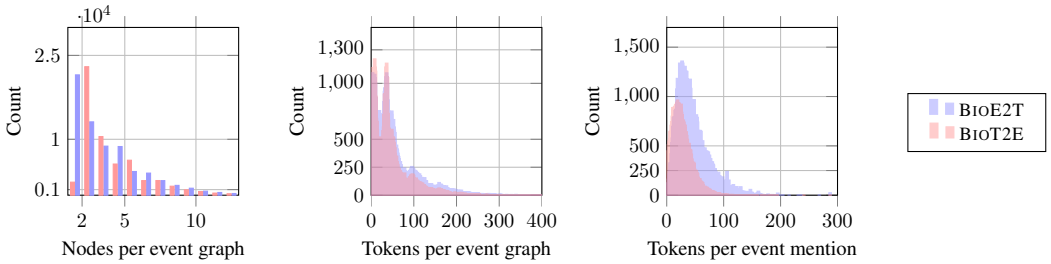

Figure 3: Distribution of instance origins, event graph sizes, and event mention lengths across our BIOE2T and BIOT2E datasets. Tokens refer to the T5 vocabulary.

cision, recall, and F1 scores according to the "approximate recursive matching" criterion (Kim et al., 2011) with string correspondence equality. Since our models stand on free generation, the derived event annotations are not accompanied by offset indices communicating their position in the original documents, preventing the "approximate span matching" relaxation. To avoid introducing error sources affecting results interpretation, we do not apply fragile heuristics such as likelihood-based class predictions (Paolini et al., 2021) or offset reconstruction (Lu et al., 2021b). In fact, (i) BIOT2E has a high type heterogeneity overhead; (ii) many sequences mention multiple events with the same trigger and scattering arguments, making difficult to assume that the matching argument-utterance is the one closest to the trigger.

Generation. To quantitatively compare predictions against ground truth literature sentences on the test set, we use a broad spectrum of natural language generation (NLG) evaluation metrics. We deepen them in §A.4 and refer the reader to (Celikyilmaz et al., 2020) for further details on their properties. In line with previous graph-to-text works, we include BLEURT (Sellam et al., 2020), a recent regression-based measure showing an higher correlation with human judgments than other simple yet widespread n-gram-overlap-based metrics.

6 Results

6.1 Event Extraction

Multi-dataset and Text-to-Text EE. Table 3 summarizes the BEE F1-scores of our end-to-end models trained on BIOT2E when evaluated on the validation set of the individual tasks[5]. We report complete precision and recall results in §A.5. Baseline systems have been assessed on the official BEE datasets—following a standard <*.txt*, *.a1*, *.a2*> structure—provided by each BioNLP shared task. They adopt discriminative architectures, meaning they train a distinct model for each task by relying on benchmark-specific event/entity/role target classes covered in *.a1* and *.a2* files. Despite conducting training on all tasks at once (unified thanks to a text-to-text format), we separately evaluate our models on each validation set, allowing for a fair comparison with the baseline. Thanks to knowledge sharing among several biomedical subareas and seq2seq, we significantly push the state-of-the-art on all the benchmarks, with T5 empirically producing better results than BART. Compared to the solutions previously known in the literature (Frisoni et al., 2021), our framework's main advantage is a higher recall and generalization capacity. We lay out a detailed error analysis in §A.6.

Low-resource. We experiment on the CG13 dataset, using only 1% to 10% of the training data (Figure 4, §A.3). F1-scores in such low-resource regime demonstrate that our framework is data-efficient compared to DeepEventMine, the SOTA discriminative model for BEE.

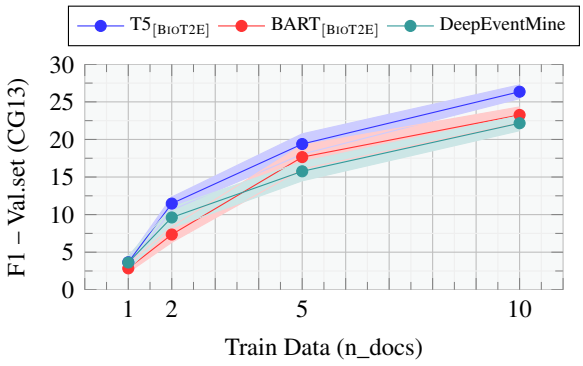

Figure 4: F1 comparison of our proposed models and DeepEventMine with train down-sampling on the CG13 validation set. Mean and standard deviation over 3 runs.

[5]Task organizers' servers for test set evaluation are currently non-available. Accessed on May 9th, 2022.

	Works \ Datasets	GE08	GE11	EPI11	ID11	MLEE	GE13	CG13	PC13	GRO13	GENIA-MK
single-task	Shared task winner	43.12†	55.90	56.41	50.10	–	50.74†	55.41†	51.10†	22.00†	–
	Trieu et al. (2020) w/o gold entities	–	56.64	55.81	50.10	51.73	45.95	54.27	50.53	–	–
	Abdulkadhar et al. (2021)	63.09†	61.74†	–	–	–	58.30†	–	–	–	61.58†
multi-task	*Ours*										
	T5-Base$_{[BIOT2E]}$	**70.74**	**73.62**	**84.43**	**84.13**	**79.91**	**81.18**	**80.10**	**83.19**	**81.91**	**83.24**
	BART-Base$_{[BIOT2E]}$	68.50	69.55	78.79	78.16	73.82	73.84	72.05	73.20	71.79	75.25

Table 3: F1-score (%) performance comparison on the validation set of the most significant biomedical event extraction tasks (eight BioNLP-STs, MLEE, and GENIA-MK). Top: original BioNLP-ST winning results and current SOTA neural systems (with per-task models); Bottom: proposed framework (with multi-task models). † indicates test set results if validation ones are unavailable. The highest scores are bolded. Both our models significantly outperform competitors (student t-test, p<0.05).

6.2 Event Graph Verbalization

NLG metrics. In Table 4, we show the event-to-text results achieved by T5 and BART on the overall BIOE2T test set. Since there are many ways to express the same symbolic concept, we use beam search at inference time to return all the different top beam sequences (i.e., multi-output). To give additional insights on generative performance, we apply metrics to all target-output pairs and not only to the one with the highest log-likelihood. T5-Base performs the best across all the NLG metrics, which—despite capturing different dimensions (grammatical correctness, fluency, informativeness, adequacy, etc.)—prove to be consistent with each other. Interestingly, we observe a relevant score gap between max-likelihood and max-score selection within a beam. This is strong evidence of the decoding strategy impact (often overlooked), also reinforcing the hypothesis that high quality human language does not follow a distribution of high probability next words (Holtzman et al., 2020). Moreover, it should be emphasized how this detachment is much more attenuated with evaluations closer to a semantic level. Sequences generated via beam search tend to be syntactically different (albeit moderate) but semantically similar, underlining the importance of metrics to grasp meaning preservation. From qualitative investigations, both models displays promising abilities in translating modifiers in elements of language (§A.7).

Graph Structure and Output Length Impact. Figure 5 shows the effect of the event graph size on verbalization, measured with BLEURT, abstractness (Gehrmann et al., 2019), and repetitiveness (Peyrard et al., 2017)[6]. In this experiment, we average the metric score for all the generated sequences and divide the results by the node number. We find that sequence quality increases as the event graph size increases, following a logarithmic function. This behavior is justified by the fact that BIOE2T, differently from other datasets like WebNLG, contains similar text lengths for various graph sizes. When the input is a larger event graph, the model has more contextual information to be leveraged during the generation, approaching the target syntactically and semantically. This thesis is also supported by the decline in abstraction, while repetitiveness is generally low and appears proportional to prediction length.

	T5$_{[BIOE2T]}$		BART$_{[BIOE2T]}$	
	MAX_L	MAX_S	MAX_L	MAX_S
BLEU	63.8	69.6 *(+5.8)*	53.1	59.6 *(+6.5)*
ROUGE-1	68.8	73.9 *(+5.1)*	60.0	65.6 *(+5.6)*
ROUGE-2	61.3	66.7 *(+5.4)*	49.8	55.7 *(+5.9)*
ROUGE-L	66.1	71.2 *(+5.1)*	56.2	61.8 *(+5.6)*
METEOR	66.6	72.1 *(+5.5)*	56.3	64.4 *(+6.1)*
BLEURT	68.9	73.5 *(+4.6)*	59.8	64.4 *(+4.6)*
NUBIA	65.2	73.1 *(+7.9)*	56.2	64.3 *(+8.1)*
BERTSCORE	94.1	95.0 *(+0.9)*	92.3	93.4 *(+1.1)*
BARTSCORE	-2.5	-1.7 *(+0.8)*	-2.31	-1.31 *(+1.0)*

Table 4: Event-to-text generation results on the BIOE2T test set. We show the average metric score considering the sequence with maximum likelihood (MAX_L) within a beam and the one obtainable by taking the sequence with maximum ground-truth-match according to the metric of interest (MAX_S). The gap between the two is shown in round brackets.

6.3 Multi-task Setting

Our method naturally allows us to train a single model on multiple datasets covering different NLP tasks, besides EE and EGV. In this setting, we use

[6] Abstractness: percentage of new n-grams in the predictions, compared to the references. Repetitiveness: average number of n-grams with at least one repetition in the generated sequences. We scan word-level unigrams.

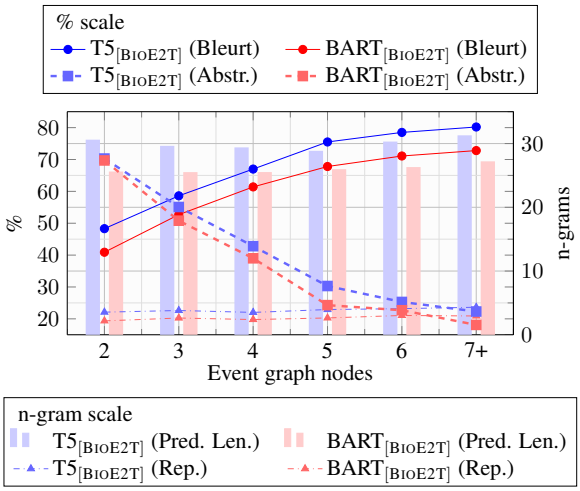

Figure 5: Average BLEURT score, abstractness, repetitiveness, and prediction length compared to the size of the event graph to condition on.

a task-specific prefix (e.g., "extract events:") to let the model know the requested transformation for each input. In particular, we inspect the performance gap on the PUBMED dataset (Cohan et al., 2018) for single document summarization, revealing a fairly advantage in terms of ROUGE (Table 5). Our intuition is that both EE (with its event-to-text back translation) and summarization tasks aim to distill salient information from massive text, providing complementary features for each other that can be beneficial for general NLP.

	T5-Base (R / P / F1)	**T5-Base Event-driven MTL** (R / P / F1)
ROUGE-1	27.24 / 58.11 / 32.97	33.46 / 47.94 / **39.41**
ROUGE-2	10.84 / 23.82 / 13.15	12.12 / 17.92 / **14.46**
ROUGE-L	18.08 / 40.29 / 22.03	22.19 / 32.01 / **26.21**

Table 5: Single document summarization performance on PUBMED test set w/o and w/ event-driven MTL.

7 Conclusion

This paper presented the first sequence-to-sequence framework for both biomedical event extraction and verbalization. Concretely, we proposed BIOT2E and BIOE2T, two highly comprehensive datasets with parallel text-event gold annotations, constructed through a novel linearization technique. By training autoregressive language models on them, we achieved an average F1-score of 0.81 on ten benchmarks, making considerable improvements over previously published work. In stark contrast with discriminative solutions, we employed the same architecture to perform previously distinct tasks, exploiting pre-trained knowledge and label semantics. Experimental results also proved the usefulness of (i) knowledge sharing between different biomedical spheres in event-based tasks, (ii) events in improving model understanding for NLP tasks in general, like document summarization. We hope that our contributions will lead to further progress in natural language understanding and generation as transfer learning becomes even more vital for graph-to-text and text-to-graph translations.

Future directions At the edge of our knowledge, this is the first work that proposes single deep neural models capable of effectively extracting (and back-translating to text) such a variety of biomedical events and their components. This high ontological coverage opens the door to numerous applications and research blueprints. Future work should tackle: (i) document-level granularities; (ii) prompting-based purely generative models (Ma et al., 2022); (iii) text⟷graph boosting approaches echoing autoencoders and Cycle-GT (Guo et al., 2020); (iv) few-shot learning; (v) event aggregation towards automatic corpus-level knowledge graph learning (Frisoni et al., 2020a; Frisoni and Moro, 2020; Frisoni et al., 2020c,b); (vi) conversion of events to logic and constrained decoding algorithms (Lu et al., 2021a); (vii) infusion of events in pre-trained language models for tasks like biomedical multi-document summarization (Moro et al., 2022) and information retrieval (Moro and Valgimigli, 2021).

8 Ethical Considerations

Largely pre-trained language models that we reference in our study might perpetuate and exacerbate biases and stereotypes hardwired in the training data, risking generating false or misleading information (Zhang et al., 2020a; Nadeem et al., 2021). Healthcare, in particular, requires strong guarantees about the factuality and reliability of predictions, but current state-of-the-art NLP solutions cannot establish such assurance. We acknowledge these issues and caution those who build on our framework to consider the aforementioned implications before deploying systems in the real world. Although automatic extraction of semantic relations from scientific documents is fundamental in the biomedical field, we do not encourage users to employ our models, like previous ones, for critical applications at present performance levels. No sensitive information is contained within our datasets,

which are derived from publicly and openly available PubMed articles. We honor and support the ACL Code of Ethics.

Acknowledgements

We would like to thank all the anonymous reviewers for their constructive feedback and valuable comments. We thank Paolo Italiani for training the DeepEventMine models and assisting us during low-resource experiments.

References

Sabenabanu Abdulkadhar, Balu Bhasuran, and Jeyakumar Natarajan. 2021. Multiscale laplacian graph kernel combined with lexico-syntactic patterns for biomedical event extraction from literature. *Knowledge and Information Systems*, 63(1):143–173.

Oshin Agarwal, Heming Ge, Siamak Shakeri, and Rami Al-Rfou. 2021. Knowledge graph based synthetic corpus generation for knowledge-enhanced language model pre-training. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 3554–3565, Online. Association for Computational Linguistics.

Iz Beltagy, Kyle Lo, and Arman Cohan. 2019. SciBERT: A pretrained language model for scientific text. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 3615–3620, Hong Kong, China. Association for Computational Linguistics.

Jonathan Berant, Vivek Srikumar, Pei-Chun Chen, Abby Vander Linden, Brittany Harding, Brad Huang, Peter Clark, and Christopher D. Manning. 2014. Modeling Biological Processes for Reading Comprehension. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1499–1510, Doha, Qatar. Association for Computational Linguistics.

Michele Bevilacqua, Rexhina Blloshmi, and Roberto Navigli. 2021. One SPRING to rule them both: Symmetric AMR semantic parsing and generation without a complex pipeline. In *Thirty-Fifth AAAI Conference on Artificial Intelligence, AAAI 2021, Thirty-Third Conference on Innovative Applications of Artificial Intelligence, IAAI 2021, The Eleventh Symposium on Educational Advances in Artificial Intelligence, EAAI 2021, Virtual Event, February 2-9, 2021*, pages 12564–12573. AAAI Press.

Jari Björne, Filip Ginter, Sampo Pyysalo, Jun'ichi Tsujii, and Tapio Salakoski. 2010. Complex event extraction at PubMed scale. *Bioinform.*, 26(12):382–390.

Jari Björne and Tapio Salakoski. 2018. Biomedical event extraction using convolutional neural networks and dependency parsing. In *Proceedings of the BioNLP 2018 workshop*, pages 98–108, Melbourne, Australia. Association for Computational Linguistics.

Rexhina Blloshmi, Simone Conia, Rocco Tripodi, and Roberto Navigli. 2021. Generating senses and roles: An end-to-end model for dependency- and span-based semantic role labeling. In *Proceedings of the Thirtieth International Joint Conference on Artificial Intelligence, IJCAI 2021, Virtual Event / Montreal, Canada, 19-27 August 2021*, pages 3786–3793. ijcai.org.

Asli Celikyilmaz, Elizabeth Clark, and Jianfeng Gao. 2020. Evaluation of text generation: A survey. *CoRR*, abs/2006.14799.

Arman Cohan, Franck Dernoncourt, Doo Soon Kim, Trung Bui, Seokhwan Kim, Walter Chang, and Nazli Goharian. 2018. A discourse-aware attention model for abstractive summarization of long documents. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 2 (Short Papers)*, pages 615–621, New Orleans, Louisiana. Association for Computational Linguistics.

K. Bretonnel Cohen, Karin Verspoor, Helen Johnson, Chris Roeder, Philip Ogren, William Baumgartner, Elizabeth White, and Lawrence Hunter. 2009. High-precision biological event extraction with a concept recognizer. In *Proceedings of the BioNLP 2009 Workshop Companion Volume for Shared Task*, pages 50–58, Boulder, Colorado. Association for Computational Linguistics.

Pierre Colombo, Chloé Clavel, and Pablo Piantanida. 2021. Infolm: A new metric to evaluate summarization & data2text generation. *CoRR*, abs/2112.01589.

Hady Elsahar, Pavlos Vougiouklis, Arslen Remaci, Christophe Gravier, Jonathon Hare, Frederique Laforest, and Elena Simperl. 2018. T-REx: A large scale alignment of natural language with knowledge base triples. In *Proceedings of the Eleventh International Conference on Language Resources and Evaluation (LREC 2018)*, Miyazaki, Japan. European Language Resources Association (ELRA).

Giacomo Frisoni., Paolo Italiani., Francesco Boschi., and Gianluca Moro. 2022. Enhancing biomedical scientific reviews summarization with graph-based factual evidence extracted from papers. In *DATA*, pages 168–179. INSTICC, SciTePress.

Giacomo Frisoni and Gianluca Moro. 2020. Phenomena Explanation from Text: Unsupervised Learning of Interpretable and Statistically Significant Knowledge. In *DATA (Revised Selected Papers)*, volume 1446, pages 293–318. Springer.

Giacomo Frisoni, Gianluca Moro, and Antonella Carbonaro. 2020a. Learning Interpretable and Statistically Significant Knowledge from Unlabeled Corpora of Social Text Messages: A Novel Methodology of Descriptive Text Mining. In *DATA 2020 - Proc. 9th Int. Conf. Data Science, Technol. and Appl.*, pages 121–134. SciTePress.

Giacomo Frisoni, Gianluca Moro, and Antonella Carbonaro. 2020b. Towards Rare Disease Knowledge Graph Learning from Social Posts of Patients. In *RiiForum*, pages 577–589. Springer.

Giacomo Frisoni, Gianluca Moro, and Antonella Carbonaro. 2020c. Unsupervised Descriptive Text Mining for Knowledge Graph Learning. In *IC3K 2020 - Proc. 12th Int. Joint Conf. Knowl. Discovery, Knowl. Eng. and Knowl. Manage.*, volume 1, pages 316–324. SciTePress.

Giacomo Frisoni, Gianluca Moro, and Antonella Carbonaro. 2021. A survey on event extraction for natural language understanding: Riding the biomedical literature wave. *IEEE Access*, 9:160721–160757.

Giacomo Frisoni, Gianluca Moro, Giulio Carlassare, and Antonella Carbonaro. 2022. Unsupervised event graph representation and similarity learning on biomedical literature. *Sensors*, 22(1):3.

Claire Gardent, Anastasia Shimorina, Shashi Narayan, and Laura Perez-Beltrachini. 2017. The WebNLG challenge: Generating text from RDF data. In *Proceedings of the 10th International Conference on Natural Language Generation*, pages 124–133, Santiago de Compostela, Spain. Association for Computational Linguistics.

Sebastian Gehrmann, Zachary Ziegler, and Alexander Rush. 2019. Generating abstractive summaries with finetuned language models. In *Proceedings of the 12th International Conference on Natural Language Generation*, pages 516–522, Tokyo, Japan. Association for Computational Linguistics.

David Geleta, Andriy Nikolov, Gavin Edwards, Anna Gogleva, Richard Jackson, Erik Jansson, Andrej Lamov, Sebastian Nilsson, Marina Pettersson, Vladimir Poroshin, Benedek Rozemberczki, Timothy Scrivener, Michael Ughetto, and Eliseo Papa. 2021. Biological insights knowledge graph: an integrated knowledge graph to support drug development. *bioRxiv*.

Mandy Guo, Joshua Ainslie, David C. Uthus, Santiago Ontañón, Jianmo Ni, Yun-Hsuan Sung, and Yinfei Yang. 2022. Longt5: Efficient text-to-text transformer for long sequences. In *NAACL-HLT (Findings)*, pages 724–736. Association for Computational Linguistics.

Qipeng Guo, Zhijing Jin, Xipeng Qiu, Weinan Zhang, David Wipf, and Zheng Zhang. 2020. CycleGT: Unsupervised graph-to-text and text-to-graph generation via cycle training. In *Proceedings of the 3rd International Workshop on Natural Language Generation from the Semantic Web (WebNLG+)*, pages 77–88, Dublin, Ireland (Virtual). Association for Computational Linguistics.

Ari Holtzman, Jan Buys, Li Du, Maxwell Forbes, and Yejin Choi. 2020. The curious case of neural text degeneration. In *8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, April 26-30, 2020*. OpenReview.net.

Pere-Lluís Huguet Cabot and Roberto Navigli. 2021. REBEL: Relation extraction by end-to-end language generation. In *Findings of the Association for Computational Linguistics: EMNLP 2021*, pages 2370–2381, Punta Cana, Dominican Republic. Association for Computational Linguistics.

Mihir Kale and Abhinav Rastogi. 2020. Text-to-text pre-training for data-to-text tasks. In *Proceedings of the 13th International Conference on Natural Language Generation*, pages 97–102, Dublin, Ireland. Association for Computational Linguistics.

Katikapalli Subramanyam Kalyan, Ajit Rajasekharan, and Sivanesan Sangeetha. 2021. AMMUS : A survey of transformer-based pretrained models in natural language processing. *CoRR*, abs/2108.05542.

Jin-Dong Kim, Claire Nédellec, Robert Bossy, and Louise Deléger, editors. 2019. *Proceedings of The 5th Workshop on BioNLP Open Shared Tasks*. Association for Computational Linguistics, Hong Kong, China.

Jin-Dong Kim, Ngan L. T. Nguyen, Yue Wang, Jun'ichi Tsujii, Toshihisa Takagi, and Akinori Yonezawa. 2012. The Genia Event and Protein Coreference tasks of the BioNLP Shared Task 2011. *BMC Bioinform.*, 13(S-11):S1.

Jin-Dong Kim, Tomoko Ohta, Kanae Oda, and Jun'ichi Tsujii. 2008. From Text to Pathway: Corpus Annotation for Knowledge Acquisition from Biomedical Literature. In *APBC*, volume 6 of *Advances in Bioinformatics and Computational Biology*, pages 165–176. Imperial College Press.

Jin-Dong Kim, Tomoko Ohta, Sampo Pyysalo, Yoshinobu Kano, and Jun'ichi Tsujii. 2011. Extracting bio-molecular events from literature - the bionlp'09 shared task. *Comput. Intell.*, 27(4):513–540.

Jin-Dong Kim, Yue Wang, and Yamamoto Yasunori. 2013a. The Genia event extraction shared task, 2013 edition - overview. In *Proceedings of the BioNLP Shared Task 2013 Workshop*, pages 8–15, Sofia, Bulgaria. Association for Computational Linguistics.

Jung-jae Kim, Xu Han, Vivian Lee, and Dietrich Rebholz-Schuhmann. 2013b. GRO task: Populating the gene regulation ontology with events and relations. In *Proceedings of the BioNLP Shared Task 2013 Workshop*, pages 50–57, Sofia, Bulgaria. Association for Computational Linguistics.

Rik Koncel-Kedziorski, Dhanush Bekal, Yi Luan, Mirella Lapata, and Hannaneh Hajishirzi. 2019. Text Generation from Knowledge Graphs with Graph Transformers. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 2284–2293, Minneapolis, Minnesota. Association for Computational Linguistics.

Ioannis Konstas, Srinivasan Iyer, Mark Yatskar, Yejin Choi, and Luke Zettlemoyer. 2017. Neural AMR: Sequence-to-sequence models for parsing and generation. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 146–157, Vancouver, Canada. Association for Computational Linguistics.

Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Veselin Stoyanov, and Luke Zettlemoyer. 2020. BART: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7871–7880, Online. Association for Computational Linguistics.

Diya Li, Lifu Huang, Heng Ji, and Jiawei Han. 2019. Biomedical event extraction based on knowledge-driven tree-LSTM. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 1421–1430, Minneapolis, Minnesota. Association for Computational Linguistics.

Ximing Lu, Peter West, Rowan Zellers, Ronan Le Bras, Chandra Bhagavatula, and Yejin Choi. 2021a. Neurologic decoding: (un)supervised neural text generation with predicate logic constraints. In *NAACL-HLT*, pages 4288–4299. Association for Computational Linguistics.

Yaojie Lu, Hongyu Lin, Jin Xu, Xianpei Han, Jialong Tang, Annan Li, Le Sun, Meng Liao, and Shaoyi Chen. 2021b. Text2Event: Controllable sequence-to-structure generation for end-to-end event extraction. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 2795–2806, Online. Association for Computational Linguistics.

Mingyu Derek Ma, Alex Taylor, Wei Wang, and Nanyun Peng. 2022. DICE: data-efficient clinical event extraction with generative models. *CoRR*, abs/2208.07989.

I. A. Mel'čuk. 1973. *Towards a Linguistic 'Meaning ↔ Text' Model*, pages 33–57. Springer Netherlands, Dordrecht.

Makoto Miwa, Sampo Pyysalo, Tomoko Ohta, and Sophia Ananiadou. 2013. Wide coverage biomedical event extraction using multiple partially overlapping corpora. *BMC Bioinform.*, 14:175.

Makoto Miwa, Paul Thompson, John McNaught, Douglas B. Kell, and Sophia Ananiadou. 2012. Extracting semantically enriched events from biomedical literature. *BMC Bioinform.*, 13:108.

Gianluca Moro and Luca Ragazzi. 2022. Semantic self-segmentation for abstractive summarization of long documents in low-resource regimes. In *AAAI*, pages 11085–11093. AAAI Press.

Gianluca Moro, Luca Ragazzi, Lorenzo Valgimigli, and Davide Freddi. 2022. Discriminative marginalized probabilistic neural method for multi-document summarization of medical literature. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 180–189, Dublin, Ireland. Association for Computational Linguistics.

Gianluca Moro and Lorenzo Valgimigli. 2021. Efficient self-supervised metric information retrieval: A bibliography based method applied to COVID literature. *Sensors*, 21(19).

Moin Nadeem, Anna Bethke, and Siva Reddy. 2021. StereoSet: Measuring stereotypical bias in pretrained language models. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 5356–5371, Online. Association for Computational Linguistics.

Tomoko Ohta, Sampo Pyysalo, Rafal Rak, Andrew Rowley, Hong-Woo Chun, Sung-Jae Jung, Sung-Pil Choi, Sophia Ananiadou, and Jun'ichi Tsujii. 2013. Overview of the pathway curation (PC) task of BioNLP shared task 2013. In *Proceedings of the BioNLP Shared Task 2013 Workshop*, pages 67–75, Sofia, Bulgaria. Association for Computational Linguistics.

Tomoko Ohta, Sampo Pyysalo, and Jun'ichi Tsujii. 2011. Overview of the epigenetics and post-translational modifications (EPI) task of BioNLP shared task 2011. In *Proceedings of BioNLP Shared Task 2011 Workshop*, pages 16–25, Portland, Oregon, USA. Association for Computational Linguistics.

Giovanni Paolini, Ben Athiwaratkun, Jason Krone, Jie Ma, Alessandro Achille, Rishita Anubhai, Cícero Nogueira dos Santos, Bing Xiang, and Stefano Soatto. 2021. Structured prediction as translation between augmented natural languages. In *9th International Conference on Learning Representations, ICLR 2021, Virtual Event, Austria, May 3-7, 2021*. OpenReview.net.

Fabio Petroni, Tim Rocktäschel, Sebastian Riedel, Patrick Lewis, Anton Bakhtin, Yuxiang Wu, and

Alexander Miller. 2019. Language Models as Knowledge Bases? In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 2463–2473, Hong Kong, China. Association for Computational Linguistics.

Maxime Peyrard, Teresa Botschen, and Iryna Gurevych. 2017. Learning to score system summaries for better content selection evaluation. In *Proceedings of the Workshop on New Frontiers in Summarization*, pages 74–84, Copenhagen, Denmark. Association for Computational Linguistics.

Sampo Pyysalo, Tomoko Ohta, and Sophia Ananiadou. 2013. Overview of the cancer genetics (CG) task of BioNLP shared task 2013. In *Proceedings of the BioNLP Shared Task 2013 Workshop*, pages 58–66, Sofia, Bulgaria. Association for Computational Linguistics.

Sampo Pyysalo, Tomoko Ohta, Makoto Miwa, Han-Cheol Cho, Junichi Tsujii, and Sophia Ananiadou. 2012. Event extraction across multiple levels of biological organization. *Bioinform.*, 28(18):575–581.

Sampo Pyysalo, Tomoko Ohta, Rafal Rak, Dan Sullivan, Chunhong Mao, Chunxia Wang, Bruno Sobral, Jun'ichi Tsujii, and Sophia Ananiadou. 2011. Overview of the infectious diseases (ID) task of BioNLP shared task 2011. In *Proceedings of BioNLP Shared Task 2011 Workshop*, pages 26–35, Portland, Oregon, USA. Association for Computational Linguistics.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J. Liu. 2020. Exploring the limits of transfer learning with a unified text-to-text transformer. *J. Mach. Learn. Res.*, 21:140:1–140:67.

Alan Ramponi, Rob van der Goot, Rosario Lombardo, and Barbara Plank. 2020. Biomedical event extraction as sequence labeling. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 5357–5367, Online. Association for Computational Linguistics.

Adam Roberts, Colin Raffel, and Noam Shazeer. 2020. How much knowledge can you pack into the parameters of a language model? In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 5418–5426, Online. Association for Computational Linguistics.

Susan H Rodger and Thomas W Finley. 2006. *JFLAP: an interactive formal languages and automata package*. Jones & Bartlett Learning.

Thibault Sellam, Dipanjan Das, and Ankur Parikh. 2020. BLEURT: Learning robust metrics for text generation. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7881–7892, Online. Association for Computational Linguistics.

Pranab Kumar Sen. 1968. Estimates of the regression coefficient based on kendall's tau. *Journal of the American statistical association*, 63(324):1379–1389.

Yi Tay, Mostafa Dehghani, Dara Bahri, and Donald Metzler. 2020. Efficient transformers: A survey. *CoRR*, abs/2009.06732.

Hai-Long Trieu, Thy Thy Tran, Anh-Khoa Duong Nguyen, Anh Nguyen, Makoto Miwa, and Sophia Ananiadou. 2020. DeepEventMine: end-to-end neural nested event extraction from biomedical texts. *Bioinform.*, 36(19):4910–4917.

Luyu Wang, Yujia Li, Ozlem Aslan, and Oriol Vinyals. 2021a. WikiGraphs: A Wikipedia text - knowledge graph paired dataset. In *Proceedings of the Fifteenth Workshop on Graph-Based Methods for Natural Language Processing (TextGraphs-15)*, pages 67–82, Mexico City, Mexico. Association for Computational Linguistics.

Qingyun Wang, Manling Li, and Xuan Wang et al. 2021b. COVID-19 literature knowledge graph construction and drug repurposing report generation. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies: Demonstrations, NAACL-HLT 2021, Online, June 6-11, 2021*, pages 66–77. Association for Computational Linguistics.

Qingyun Wang, Semih Yavuz, Xi Victoria Lin, Heng Ji, and Nazneen Rajani. 2021c. Stage-wise fine-tuning for graph-to-text generation. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing: Student Research Workshop*, pages 16–22, Online. Association for Computational Linguistics.

Zeqiu Wu, Michel Galley, Chris Brockett, Yizhe Zhang, Xiang Gao, Chris Quirk, Rik Koncel-Kedziorski, Jianfeng Gao, Hannaneh Hajishirzi, Mari Ostendorf, and Bill Dolan. 2021. A controllable model of grounded response generation. In *Thirty-Fifth AAAI Conference on Artificial Intelligence, AAAI 2021, Thirty-Third Conference on Innovative Applications of Artificial Intelligence, IAAI 2021, The Eleventh Symposium on Educational Advances in Artificial Intelligence, EAAI 2021, Virtual Event, February 2-9, 2021*, pages 14085–14093. AAAI Press.

Haoran Zhang, Amy X. Lu, Mohamed Abdalla, Matthew B. A. McDermott, and Marzyeh Ghassemi. 2020a. Hurtful words: quantifying biases in clinical contextual word embeddings. In *ACM CHIL '20: ACM Conference on Health, Inference, and Learning, Toronto, Ontario, Canada, April 2-4, 2020 [delayed]*, pages 110–120. ACM.

Junsheng Zhang, Kun Li, Changqing Yao, and Yunchuan Sun. 2020b. Event-based summarization method for scientific literature. *Personal and Ubiquitous Computing*, pages 1–10.

Tianran Zhang, Muhao Chen, and Alex A. T. Bui. 2020c. Diagnostic Prediction with Sequence-of-sets Representation Learning for Clinical Events. In *Artificial Intelligence in Medicine - 18th International Conference on Artificial Intelligence in Medicine, AIME 2020, Minneapolis, MN, USA, August 25-28, 2020, Proceedings*, volume 12299 of *Lecture Notes in Computer Science*, pages 348–358. Springer.

Chao Zhao, Marilyn Walker, and Snigdha Chaturvedi. 2020. Bridging the structural gap between encoding and decoding for data-to-text generation. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 2481–2491, Online. Association for Computational Linguistics.

Weizhong Zhao, Jinyong Zhang, Jincai Yang, Tingting He, Huifang Ma, and Zhixin Li. 2021. A novel joint biomedical event extraction framework via two-level modeling of documents. *Inf. Sci.*, 550:27–40.

A Appendix

A.1 Formal Event Grammar

Linearized events follow the formal context-free grammar orderly detailed in Table 6.

A.2 Insights on Event Modifiers

Table 7 recaps the event modifiers covered by our work, their meaning and possible values.

A.3 Training Details and Reproducibility

T5 and BART. We reimplemented T5-Base (∼220M parameters, 12-layers, 768-hidden, 12-heads) in Flax (T5X) starting from the Google Research codebase[7] and built our BART-Base (∼139M, 12-layers, 768-hidden, 16-heads) model in PyTorch using the HuggingFace's Transformers library[8]. For all variants, weights are initialized through the official checkpoints (C4 pre-training for T5). For verbalization, we set the maximum length for event mentions and linearized event graphs to 200 and 400, respectively. For parsing, we extended the linearization maximum length to 650. Instead, we used 1024 and 256 for single-document summarization input/output (truncated). We used BF16 mixed precision and a batch size of 16 (with gradient accumulation every 2 batches) for all models. We employed the Adam optimizer. Following (Raffel et al., 2020), T5 models are fine-tuned with a constant learning rate of 0.001, 1000 warmup steps, and a 0.1 dropout rate. For BART, we used default hyperparameters, but we did not penalize the model for the generation of repeated ngrams, e.g., multiple opening or closing brackets. We chose the best checkpoints based on the ROUGE score on the validation set; we found that it highly correlates with EE metrics (due to the extractive nature of the task). At prediction time, we used beam search with beam size 4 for generation and greedy decoding for parsing. We trained single-task T5 and BART models for 50 epochs (≈40 and ≈30 hours per full-training on BIOE2T and BIOT2E, respectively). The estimated[9] CO2 impact incurred by each model training belongs to the range [6.09, 8.12] kg (carbon footprint). Regarding T5 MTL, we prepended task-specific tags to the input records and performed 10 epochs using a mixture with 100% data proportion sampling for each task.

DeepEventMine. We reimplemented the training script (not released by the authors, accessed on January 16th, 2022), faithfully following the steps listed in the paper (Trieu et al., 2020). For comparison, we modified the original evaluation script to assess predictions without gold entities (not used by our framework).

Low-resource. As outlined in §6.1, we experimented on the CG13 dataset with only a limited portion of the training set available. We selected 1, 2, 5, and 10 PubMed abstracts with an average number of mentioned events. To account for the small dataset size, we fine-tuned on CG13 for a greater number of epochs, proportional to the size of each partition (50x, 25x, 10x, 5x). So, we trained T5 and BART for 2.500, 1.250, 500, and 250 epochs; DeepEventMine for 4.000, 2.000, 800, and 400 epochs. We performed 3 runs (each model being fine-tuned on the same 4 subsets of the training set and then evaluated on the entire validation set).

Hardware Setup. We ran each experiment on a workstation having two Nvidia GeForce RTX 3090 GPUs with 24GB of dedicated memory each, 64GB of RAM, and a Intel® Core™ i9-10900X CPU @ 3.70GHz.

A.4 NLG Evaluation Metrics

Metrics (default parameters and official repository implementation) are summarized in Table 8. As for BARTSCORE and BLEURT, we used the BLEURT-20 and BARTSCORE-CNNDM pre-trained models, respectively.

[7] https://github.com/google-research/t5x

[8] https://huggingface.co/transformers/model_doc/bart.html

[9] http://green-algorithms.org/

Symbol	Description
EV	event
T	trigger
A	argument
TST	text span trigger
TRG	trigger role group
TSE	text span entity
EVT	event type
MG	modifier group
M	modifier name
MV	modifier value
E	entity
ET	entity type
RG	role group
R	role

(a) Symbols in V

Symbol	Type Set
tst	^[A-Za-z0-9]+$
tse	^[A-Za-z0-9]+$
evt	$EventTypes$
m	$Modifiers$
mv	$ModifierValues$
et	$EntityTypes$
r	$RoleTypes$

(b) Symbols in Σ

Id	Rule	Id ↓	Rule ↓
1	$EV \rightarrow T\ A$	12	$E \rightarrow [TSE \mid ET \mid RG]$
2	$T \rightarrow [TST \mid EVT\ MG\ TRG]$	13	$RG \rightarrow RG \mid RG$
3	$MG \rightarrow MG\ MG$	14	$RG \rightarrow R = TST$
4	$MG \rightarrow \mid M = MV$	15	$TST \rightarrow tst$
5	$MG \rightarrow \epsilon$	16	$TSE \rightarrow tse$
6	$TRG \rightarrow TRG\ TRG$	17	$EVT \rightarrow evt$
7	$TRG \rightarrow \mid R = TST$	18	$ET \rightarrow et$
8	$TRG \rightarrow \epsilon$	19	$M \rightarrow m$
9	$A \rightarrow A\ A$	20	$MV \rightarrow mv$
10	$A \rightarrow E$	21	$R \rightarrow r$
11	$A \rightarrow EV$	22	$EV \rightarrow EV\,EV$

(c) Rules in R

Table 6: Formal definition of the event grammar $G = \langle V, \Sigma, R, EV \rangle$. V is the finite set of variables (a); Σ is the finite set of terminal symbols and therefore the alphabet of our event language (b); R is the finite set of production rules (c); EV is the start variable. As for (b), each symbol on the left belongs to the type set on the right, which depends on the dataset event schema. The only exception concerns tst and tse which are alphanumeric strings, reported as regex for notational simplicity. The pipe marker is intended as a character and not as a logic operator.

Modifier	Definition	Possible values
Polarity	The truth value of an event	Positive (*default*) Negative
Speculation	Whether an event is speculated or not	True, False (*default*)
Source	Origin of the knowledge expressed by an event	Current paper (*default*) Other
Manner	The intensity level of an event	High, Low, Neutral (*default*)
Certainty level	The confidence of an event being expressed	L1 (low confidence), L2 (not complete confidence) L3 (high confidence, *default*)
Knowledge type	The overarching information expressed by the event	Investigation, Observation, Analysis, Fact, Method, Other (*default*)

Table 7: Summary of the event modifiers in BIOT2E and BIOE2T.

Metric	U	S	Strategy	Model(s)
BLEU	✓		N-gram recall	–
ROUGE	✓		N-gram precision	–
METEOR	✓		N-gram overlap w/ synonym match	–
BERTSCORE	✓		Semantic similarity	BERT
BARTSCORE	✓		Conditioned generation for faithfulness, precision, and recall	BART
BLEURT		✓	Human score prediction	BERT
NUBIA		✓	Human score prediction	RoBERTa GPT-2

Table 8: Metrics applied for evaluating event graph verbalization performance. **U**: unsupervised, **S**: supervised, based on the need for human judgments to train. They belong to $[0, 1]$, with the exception of BARTSCORE, whose range is $]-\infty, 0]$. The higher the score, the more valid the hypothesis is.

A.5 Detailed Event Extraction Results

We report detailed event extraction performance for our models in Table 9.

A.6 Error Analysis

A.6.1 Event Extraction

We quantitatively classify errors into three broad categories: format, trigger, and argument errors. Further, we organize the latter two in fine-grained categories: under-prediction (i.e., expected but not predicted), over-prediction (i.e., predicted but not expected), and wrong type. Finally, we distinguish the target type, especially keeping track of multi-event outputs and nested (i.e., complex) events. Table 10 reports the proportions of error types we identified. We notice the most considerable fraction of errors is due to triggers. From a closer look, we found that over-predicted triggers are often linked to generic words used very frequently to indicate specific event types. For instance, similarly to what emerged in previous works (Ramponi et al., 2020), T5$_{[\text{BIOE2T}]}$ identifies a positive regulation event anchored at "activated" in the sentence: "Tax [...] maximally activated HTLV-I-LTR-CAT and kappa B-fos-CA" albeit the gold standard does not contain the event in this instance. However, we believe these errors are acceptable from a semantic point of view and sometimes highlight a low-annotation problem within the datasets. As for wrong trigger and argument types, the model tends to generate different but semantically equivalent labels, e.g., "sufficient to restore" instead of "restored", "Protein_molecule" instead of "Protein". This issue underlines the need for alternative automatic evaluation metrics operating at the semantic level. Format errors are less frequent, proving that the model can successfully manage bracket $[\ldots \mid \ldots]$ rules.

Works \ Datasets		GE08	GE11	EPI11	ID11	MLEE	GE13	CG13	PC13	GRO13	GENIA-MK
T5-Base$_{[\text{BIOT2E}]}$	R	67.71	74.57	92.28	90.28	71.56	73.18	70.83	75.68	74.84	75.79
	P	74.05	72.68	77.81	78.77	90.45	91.15	92.17	92.36	90.45	92.32
	F1	70.74	73.62	84.43	84.13	79.91	81.18	80.10	83.19	81.91	83.24
BART-Base$_{[\text{BIOT2E}]}$	R	65.23	70.82	87.66	84.22	66.84	66.91	64.79	67.15	66.82	69.62
	P	72.12	68.33	71.56	72.91	82.43	82.36	81.13	80.45	77.56	81.87
	F1	68.50	69.55	78.79	78.16	73.82	73.84	72.05	73.20	71.79	75.25

Table 9: Recall (R), Precision (P), and F1-score (%) performance of T5-Base$_{[\text{BIOT2E}]}$ and BART-Base$_{[\text{BIOT2E}]}$ on the validation set of the most significant biomedical event extraction tasks.

Error Type	Fraction		
	All	Nested	Multi-event
Format	5%	2%	3%
Trigger			
Under-prediction	17%	8%	6%
Over-prediction	28%	16%	5%
Wrong type	10%	3%	4%
Argument			
Under-prediction	13%	7%	4%
Over-prediction	23%	14%	5%
Wrong type	4%	2%	1%

Table 10: Quantitative event extraction error analysis of T5$_{[\text{BIOE2T}]}$. Average fraction values among the validation sets of all the ten datasets.

A.6.2 Event Graph Verbalization

To further assess the quality of the event-graph-controlled text, we conduct an in-depth human evaluation study for a manual scrutiny of error sources. Following previous works (Colombo et al., 2021), human raters are presented with the source graph, the predicted text, and the ground-truth. They are asked to judge the prediction along six quality criteria with binary rating.

- *Coverage*. Are all the information presented in the event graph included in the text?
- *Compliance*. Does the text contains only the information in the input event graph?
- *Correctness*. Are interactions modeled in the event graph correctly mentioned (correct roles and entity-linkage)?
- *Factuality*. Does the text contains only factual information?
- *Text Structure*. Is the text well-structured, grammatically correct and written in acceptable English?
- *Fluency*. Does the text progress naturally? Is it easy to understand? Is it a coherent whole?

Since the number of events is not balanced with respect the biomedicine subarea (see Table 1), we randomly sample eight graph-text pairs for each dataset composing the BIOE2T test set (80 in total). The evaluation is performed for T5$_{[\text{BIOE2T}]}$ and BART$_{[\text{BIOE2T}]}$. For each prediction, we collect scores from 3 expert evaluators and average them.

The average Kendall's coefficient (Sen, 1968) among all evaluators' inter-rater agreement is 0.86. Kendall's coefficient ranges from -1 to 1, indicating low to high association. Considering the subjectivity of the rating task, this number indicated high human agreement for the EGV task.

Table 11 summarizes the results. We first note a similar trend as in EE, with T5 outperforming BART on most quality axes. We observe that the generators mostly suffer from low compliance issues due to the verbalization of additional information not originally modeled in the input graph. We investigated the reason for this error, finding three main causes: (i) event mentions shared by multiple events, (ii) a low number of nodes, and (iii) superficial and often incomplete dataset annotations—especially on GE08 and ID11. Hence, the error is not attributable to a scarce expressive power of events as semantic representations. Notably, despite the frequent verbalization of further relationships, models generally do not produce fabricated facts, and the output quality is high. Compared to T5, BART is more a victim of hallucinations and tends to paraphrase the text more, mixing patterns seen during training.

A.7 Parsing and Generation Examples

Some input-output examples for the EE and EGV tasks are shown in Table 12 and Table 13, respectively. We emphasize that current end-to-end neural conversation models inherently lack the flexibility to impose semantic control in the response generation process (Wu et al., 2021), justifying the importance of EGV. This control is essential to ensure that users' semantic intents are satisfied and

Perspectives	Models	
	$\text{T5}_{[\text{BIOE2T}]}$	$\text{BART}_{[\text{BIOE2T}]}$
Coverage	**0.97**	0.94
Compliance	0.18	**0.22**
Correctness	**0.96**	0.91
Factuality	**0.99**	0.90
Text Structure	**0.97**	0.81
Fluency	**0.99**	0.83

Table 11: Human evaluation scores of the verbalized event graphs on a random sample of 80 instances from the test set. The highest are bolded.

to establish a degree of specificity on generated outputs. Following this line, modifiers offer the concrete opportunity of asking a model not only to verbalize an event but also to do it with a particular writing style. Multiple modifiers can be set at the same time (e.g., "H2A may not be methylated"), allowing great flexibility. EGV is also useful to collect rationales from language models more effectively, revealing what knowledge is stored in their parameters. The qualitative results obtained indicate that the event graphs can indeed steer the language model towards informative content following provided confidence measures or other lexical clue types.

Text	Extracted Event
We wanted to establish whether BMP-6 also could affect the viability of normal B cells.	*Ground_truth* [affect | Regulation | Speculation = True] [BMP-6 | Protein | Cause = affect] [normal B cells | Cell | Theme = affect] 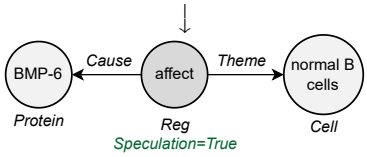 *T5*[BIOE2T] ✓ *BART*[BIOE2T] ✓
We analyzed the methylation status of hMLH1 and MGMT using methylation-specific polymerase chain reaction and DNA sequencing analysis.	*Ground_truth* [methylation | DNA_methylation | Speculation = True] [MLH1 | Protein | Theme = methylation] [methylation | DNA_methylation | Speculation = True] [MGMT | Protein | Theme = methylation] 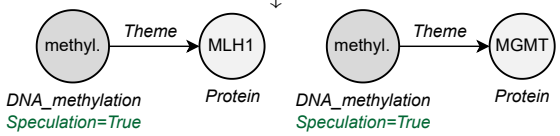 *T5*[BIOE2T] ✓ *BART*[BIOE2T] ✓
We found a lack of PKD1 expression.	*Ground_truth* [lack | Negative_regulation] [[expression | Gene_expression | Theme = lack] [PKD1 | Protein | Theme = expression]] 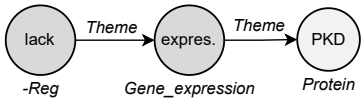 *T5*[BIOE2T] [lacking | Negative_regulation] [[expression | Gene_expression | Theme = lack] [PKD1 | Protein | Theme = expression]] *BART*[BIOE2T] ✓
Conversely, the murine kappa light chain enhancer motif (GGGGACTTTCCG) does not efficiently bind the nuclear p50p50 from tolerant murine P388 macrophages.	*Ground_truth* [bind | Binding | Polarity = Negative] [kappa light chain | Protein | Theme = bind] [p50 | Protein | Theme2 = bind] [GGGGACTTTCCG | Entity | Site = bind] 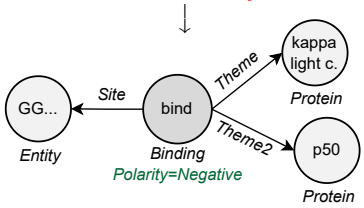 *T5*[BIOE2T] [bind | Binding | KT = Observation | Polarity = Negative | Source = Current] [murine kappa light chain enhancer motif | DNA_domain_or_region | Theme = bind] [p50p50 | Protein_complex | Theme = bind] [not | Negative-Cue | Cue = bind] *BART*[BIOE2T] [bind | Binding | Polarity = Negative] [kappa light chain | Protein | Theme = bind] [p50 | Protein_molecule | Theme2 = bind]

Table 12: Input-output event extraction examples. Modifiers' influence is highlighted in blue, while deviations (positive or negative) from the ground truth are in red. The check-mark symbol denotes a full-correspondence between the predicted text and the ground truth.

Event	Text
[Overexpression | Gene_expression] [Bax inhibitor-1 | Gene_or_gene_product | Theme = Overexpression] [[induces | Positive_regulation | Cause = Overexpression | Theme = transformation] [transformation | Cell_transformation] [NIH3T3 cells | Cell | AtLoc = transformation] [cell | Cell | Theme = transformation]] 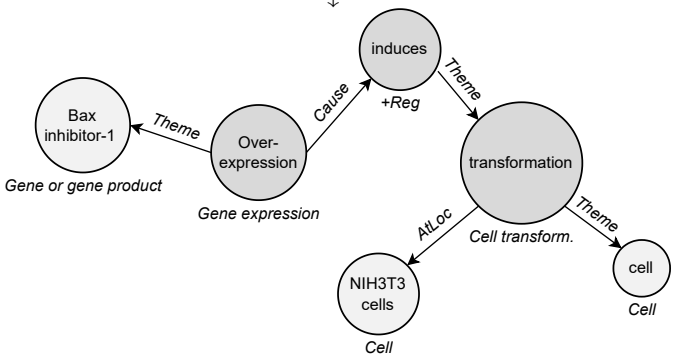	*Ground_truth* Overexpression of Bax inhibitor-1 (BI-1) induces cell transformation in NIH3T3 cells. *T5*[BIOE2T] ✓ *BART*[BIOE2T] ✓
[regulated | Regulation] [[Activation | Positive_regulation | KT = Observation | Theme = regulated] [NF-kappa B | Protein_molecule | Theme = Activation]] [phosphorylations | Other | Cause = regulated] 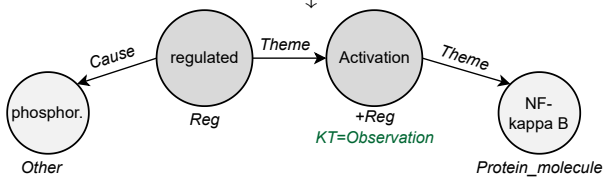	*Ground_truth* Activation of NF-kappa B in vivo is regulated by multiple phosphorylations. *T5*[BIOE2T] ✓ *BART*[BIOE2T] Activation of NF-kappa B is regulated by phosphorylations and rapid degradation of its inhibitor I kappa B alpha.
[inhibit | Negative_regulation] [DMI | Simple_chemical | Cause = inhibit] [[growth | Cell_proliferation | Theme = inhibit] [Ca3/7 cells | Cell | Theme = growth]] 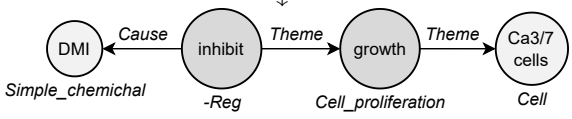	*Ground_truth* DMI is suggested to inhibit the growth of Ca3/7 cells. *T5*[BIOE2T] Therefore, we examined the effect of DMI on Ca3/7 cells, resistant to growth inhibition by the synthetic glucocorticoid fluocinolone acetonide (FA). *BART*[BIOE2T] The growth inhibition of Ca3/7 cells by DMI is neither a consequence nor the sole cause of growth arrest.
[involvement | Regulation | Speculation = True] [[progression | Development | Theme = involvement] [tumor | Cancer | Theme = progression]] [TGF-beta | Gene_or_gene_product | Cause = involvement] 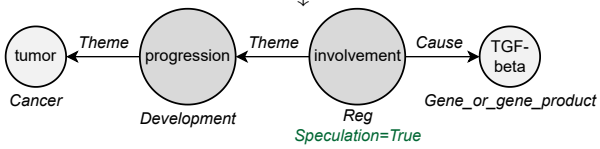	*Ground_truth* TGF-beta effects on angiogenesis, stroma formation, and immune function suggest its possible involvement in tumor progression. *T5*[BIOE2T] In the present study, we investigated the possible involvement of transforming growth factor beta (TGF-beta) in tumor progression. *BART*[BIOE2T] An understanding of the molecular basis of TGF-beta-mediated inhibition of angiogenesis and tumor progression will aid in the development of novel therapeutics for the treatment of cancer.

Table 13: Input-output event graph verbalization examples. Modifiers' influence is highlighted in blue. The check-mark symbol denotes a full-correspondence between the predicted text and the ground truth.

Multimodal Semi-supervised Learning for Disaster Tweet Classification

Iustin Sirbu
University Politehnica of Bucharest
iustin.sirbu@upb.ro

Tiberiu Sosea
University of Illinois Chicago
tsosea2@uic.edu

Cornelia Caragea
University of Illinois Chicago
cornelia@uic.edu

Doina Caragea
Kansas State University
dcaragea@ksu.edu

Traian Rebedea
University Politehnica of Bucharest
traian.rebedea@upb.ro

Abstract

During natural disasters, people often use social media platforms, such as Twitter, to post information about casualties and damage produced by disasters. This information can help relief authorities gain situational awareness in nearly real time, and enable them to quickly distribute resources where most needed. However, annotating data for this purpose can be burdensome, subjective and expensive. In this paper, we investigate how to leverage the copious amounts of unlabeled data generated on social media by disaster eyewitnesses and affected individuals during disaster events. To this end, we propose a semi-supervised learning approach to improve the performance of neural models on several multimodal disaster tweet classification tasks. Our approach shows significant improvements, obtaining up to 7.7% improvements in F-1 in low-data regimes and 1.9% when using the entire training data. We make our code and data publicly available.[1]

1 Introduction

The upswing of text and image sharing on social media platforms, such as Twitter, during mass emergency situations has led to numerous opportunities to gain timely access to valuable information that can help disaster relief authorities act quicker and more efficiently. Specifically, as a disaster unfolds, information shared on social media can provide insights into the infrastructure and utility damage, casualties, and missing people. Recent studies have focused on collecting and manually annotating disaster data with respect to such situational awareness categories, followed by training machine learning classifiers to automatically identify situational awareness information, useful for relief operations (Alam et al., 2018b; Ashktorab et al., 2014).

However, disaster events produce large amounts of user-generated data, of which only a small fraction can be annotated, due to the time-sensitive nature of the problem, together with high annotation costs, and also inherent subjectivity associated with annotating data (e.g., tweets).

To address this limitation, we propose a semi-supervised multimodal approach that can leverage the copious amounts of unlabeled data to improve the performance on various multimodal tasks. Specifically, we extend the *FixMatch* (Sohn et al., 2020) algorithm proposed for semi-supervised image classification to a multimodal setting. To account for subjective annotations and potentially overlapping labels, we use soft pseudo-labels instead of the original hard pseudo-labels. We apply the adapted *FixMatch* to the CrisisMMD labeled dataset and tasks (Alam et al., 2018b), to improve the performance of supervised baselines through the use of unlabeled data. We use 122K unlabeled tweets, containing both text and images, collected automatically using text queries about disasters that occurred during the year of 2017.

Experimental results show that our proposed approach produces performance improvements on all three CrisisMMD tasks in various data regimes. Notably, we obtain as much as 7.7% using as few as 250 examples per class and 1.9% improvement when using the entire data. To our knowledge, we are the first to propose a semi-supervised method for multimodal data using FixMatch and text-based searches for collecting a large unsupervised dataset. While our experiments focus on disaster tweets, the method can be easily generalized. Finally, we provide an extensive error analysis of our models. We analyze how the supervised model's predictions change with the introduction of unlabeled data and reinforce the importance of our improved version of *FixMatch*.

Our contributions are as follows:

(1) We extend *FixMatch* algorithm to a multimodal scenario and offer two extensions to the original approach relevant for text and multimodal

[1] https://github.com/iustinsirbu13/multimodal-ssl-for-disaster-tweet-classification

Proceedings of the 29th International Conference on Computational Linguistics, pages 2711–2723
October 12–17, 2022.

datasets. **(2)** We show that inexpensive unlabeled data gathered using text queries and basic preprocessing can be leveraged by our multimodal FixMatch to improve performance on 3 classification tasks. **(3)** We provide a detailed analysis into the predictions of the semi-supervised approaches, and compare them to their supervised counterparts.

2 Related Work

2.1 Semi-supervised learning

Semi-supervised learning combines labeled data with large amounts of unlabeled data during training to improve the performance of the models. *MixMatch* (Berthelot et al., 2019b) uses a sharpening technique and guesses low-entropy labels for augmented unlabeled data. Next, it employs *MixUp* (Zhang et al., 2017) to blend the labeled and unlabeled examples. *FixMatch* (Sohn et al., 2020) combines two standard semi-supervised techniques: consistency regularization (Rasmus et al., 2015; Sajjadi et al., 2016; Tarvainen and Valpola, 2017) and pseudo-labeling (Lee et al., 2013). The pseudo-labels are generated using the current model's predictions on weakly-augmented unlabeled images. Next, the model predicts the pseudo-labels for strongly augmented versions of the same images. *Noisy Student Training* (Xie et al., 2020) first trains a teacher model on the labeled data to predict pseudo-labels for the unlabeled examples. Next, it trains a larger student model on all the data (i.e. labeled and unlabeled) using augmentation and dropout. The teacher model is then replaced by the student, and the process is repeated until convergence.

Text and image SSL methods are usually tightly related. For example, Miyato et al. (2016) extends adversarial training from images (Miyato et al., 2015) to text. Specifically, the proposed approach leverages adversarial attacks for consistency regularization by identifying an optimal perturbation for each sample (instead of using random perturbations). *MixText* (Chen et al., 2020) adapts *MixMatch* for text and proposes replacing MixUp method with TMix, a newly introduced approach for interpolating texts in a hidden space. *Unsupervised Data Augmentation*, or *UDA* (Xie et al., 2019) has been shown to be effective both for texts and images. It uses common SSL techniques such as consistency regularization, sharpening and data filtering (i.e., confidence based masking and balancing), together with qualitative augmentations (i.e. RandAugment for images and back-translation for texts).

2.2 Disaster tweet classification

A significant body of research focuses on the benefits of social media information for improving disaster relief efforts. Some of these studies focus on learning from solely textual data (e.g., tweets) (Yin et al., 2012; Guan and Chen, 2014; Yuan and Liu, 2018; Imran et al., 2015; Kryvasheyeu et al., 2016; Li et al., 2018a; Enenkel et al., 2018; Alam et al., 2018a; Mazloom et al., 2019; Neppalli et al., 2018; Li et al., 2018b) including semi-supervised learning from text (Li et al., 2021, 2018c). Other studies focus on learning only from images (Lagerstrom et al., 2016; Alam et al., 2017; Li et al., 2019a; Chaudhuri and Bose, 2020; Alam et al., 2018d; Bica et al., 2017; Nguyen et al., 2017; Li et al., 2019b; Weber et al., 2020). However, many tweets posted during disasters contain both text and images, which, if studied jointly, can provide a better portrayal of the damage produced by disasters, or the needs of the affected individuals. Therefore, it is not surprising that multimodal models in the disaster space have recently started to gain popularity (Mouzannar et al., 2018; Rizk et al., 2019; Gautam et al., 2019; Nalluru et al., 2019; Agarwal et al., 2020; Abavisani et al., 2020; Li and Caragea, 2020; Hao and Wang, 2020; Ofli et al., 2020).

Sosea et al. (2021) leverages the image-text relationship to improve the performance of multimodal disaster tweet classification. Zou et al. (2021) proposes a framework containing separate feature extractors for each modality, followed by a procedure for fusing the two modalities. The approach proposed in Pranesh et al. (2021) is similar, however, the fusion is performed using an attention mechanism. Dinani and Caragea (2021) uses Capsule Networks to classify disaster images and identify the informativeness of a image. Alam et al. (2021) uses Noisy Student Training (Xie et al., 2020) and a multitasking setting to classify images from disaster tweets. Bidari (2021) proposed a weighting mechanism between predictions of a BERT (Devlin et al., 2018) model trained on text and a VGG16 (Simonyan and Zisserman, 2015) model trained on images.

These existing approaches, however, do not use the large amounts of unlabeled multimodal data generated during disasters. In this paper, we propose a semi-supervised approach to leverage this

(a) *This 4 BD/ 2 BA in Mora MUST be seen. Call, text or direct message me for more info!*

(b) *St. Augustine bed & breakfast picking up the pieces after Hurricane Irma*

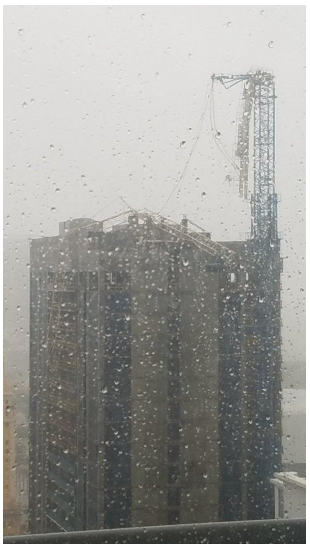

(c) *A huge crane just collapsed on top of building in down town Miami*

(d) *Irma update: Free roof help available*

(e) *Magnitude 6.1 aftershock hits Mexico as search for people and pets continues*

Figure 1: Examples of errors of the *MMBT* model that are corrected by *FixMatch* on the *Informativeness* and *Humanitarian* CrisisMMD tasks: **(a)** MMBT: *informative*; True: *not informative* **(b)** MMBT: *infrastructure and utility damage*; True: *not humanitarian* **(c)** MMBT: *not informative*; True: *informative* **(d)** MMBT: *infrastructure and utility damage*; True: *rescue, volunteering, or donation effort* **(e)** MMBT: *infrastructure and utility damage*; True: *rescue, volunteering, or donation effort*

data to improve the multimodal disaster tweet classification. Our approach extends *FixMatch* (originally proposed for image classification) to the multimodal setting and introduces two enhancements.

3 Methods

3.1 Baseline Modeling

We employ various single-modal and multi-modal models to compare our proposed approach. First, we experiment with an image-only model, *ResNet-152* (He et al., 2016), on top of which we add a linear layer for classification. Next, we use a *Multimodal Bitransformer* (*MMBT*) (Kiela et al., 2019) to leverage both the image and text for disaster tweet classification, as it already showed good results on this task (Sosea et al., 2021). We randomly crop and rescale the input images to 224x224, a common size for these types of networks, and also perform a standard horizontal flip and shift augmentation. We denote these approaches by *ResNet Aug* and *MMBT Aug*.

3.2 Semi-supervised learning

To leverage the large amounts of data generated during disaster events, we adapt the *FixMatch* (Sohn et al., 2020) algorithm to the multimodal setting. *FixMatch* obtains impressive performance on several Computer Vision tasks by combining consistency regularization (Sajjadi et al., 2016; Laine and Aila, 2016) and pseudo-labeling (McLachlan, 1975). *FixMatch* computes the overall loss l as a weighted sum of two loss terms $l = l_s + \lambda_u l_u$, where λ_u is a weighting parameter, l_s is the loss on labeled data, and l_u is the loss on unlabeled data. Specifically, in the multimodal setting, the labeled loss is defined as:

$$l_s = \frac{1}{B}\sum_{b=1}^{B} H(p_b, p_m(\alpha(x_b^{img}), \beta(x_b^{txt})))$$

where B is the batch size, H is the cross-entropy loss, p_b is the one-hot encoding of the true label of a multimodal tweet (x_b^{img}, x_b^{txt}), and p_m is the model's prediction (i.e., probability distribution over possible classes y) on a weakly augmented image, $\alpha(x_b^{img})$, and weakly augmented text, $\beta(x_b^{txt})$. The unlabeled loss is defined as:

$$l_u = \frac{1}{\mu B}\sum_{b=1}^{\mu B} \mathbb{1}_\tau(q_b) H(\hat{q_b}, p_m(\mathcal{A}(u_b^{img}), \mathcal{B}(u_b^{txt})))$$

where μ is the ratio between the number of labeled and unlabeled examples in a batch, and $q_b = p_m(\alpha(u_b^{img}), u_b^{txt})$ is the probability distribution over classes y, for the unlabeled example (u_b^{img}, u_b^{txt}). The function $\mathbb{1}_\tau(q_b)$ is used to filter out examples for which the prediction confidence, i.e., $\max_y(q_b)$, is less than a threshold, τ. For the remaining examples, the prediction is converted to a pseudo-label using $\hat{q_b} = \arg\max_y(q_b)$. Finally, the cross-entropy loss is computed between the one-hot encoding of this pseudo-label and the prediction of the model on a strongly augmented version of the current image, $\mathcal{A}(u_b^{img})$, and the corresponding augmented text, $\mathcal{B}(u_b^{txt})$. The strong augmentations for image use either RandAugment (Cubuk et al., 2020) or CTAugment (Berthelot et al., 2019a). For text augmentation we experiment with EDA (Wei and Zou, 2019) and back-translation (Edunov et al., 2018). We offer more details about our text augmentation methods in Subsection 4.3.

In this paper, we apply the *FixMatch* algorithm to our multimodal disaster domain, using *MMBT* as the base model. To understand the benefits of the multimodal representation, we also apply *FixMatch* on images only, using *ResNet-152* as the base model. We denote these methods by *MMBT FixMatch* and *ResNet FixMatch*, respectively.

3.3 FixMatch Enhancements

We propose two key enhancements to the unlabeled loss computation. First, we use *soft* pseudo-labels (q_b) instead of the hard labels ($\hat{q_b}$) used in the original paper:

$$l_u^{LS} = \frac{1}{\mu B} \sum_{b=1}^{\mu B} H(q_b, p_m(\mathcal{A}(u_b^{img}), \mathcal{B}(u_b^{txt})))$$

We argue that, in the disaster domain, there can be significant semantic overlap between two labels. For instance, in Figure 1e, which is labeled with *Rescue, volunteering, or donation effort* for the humanitarian task, there is a destroyed building in the background. By using soft labels, we can also incorporate information about the *Infrastructure and utility damage* class instead of stirring the model towards confidently predicting the example into the *Rescue, volunteering, or donation effort* class.

Second, we consider a variable weighting scheme for the loss, l. Originally, *FixMatch* employed a fixed weighting between the labeled and unlabeled loss (e.g., $\lambda_u = 1$). We argue that the predictions of the model during the first few epochs are not qualitative, hence using the predicted labels of unlabeled data can hurt the performance. To prevent that, we employ a linear growth of the unlabeled loss. Starting with 0 in the first epoch, we increase this loss in steps of 2 each epoch. Our loss becomes $l^{LS} = l_s + \lambda_u(t) l_u^{LS}$, where $\lambda_u(t) = 2t$, and t is the epoch number. We denote the corresponding *MMBT* semi-supervised model by *MMBT Fixmatch LS*, while the corresponding *ResNet-152* model is denoted by *Resnet Fixmatch LS*.

4 Experiments

4.1 Labeled Data

We evaluate our semi-supervised multimodal approach on CrisisMMD (Alam et al., 2018b), a multimodal Twitter dataset from natural disasters. The dataset contains $18,000$ tweets with both text and images extracted during disasters such as the Iraq-Iran Earthquakes or Hurricanes Irma, Harvey and Maria. CrisisMMD wfas manually labeled for three classification tasks: *(1) Informativeness:* A tweet is labeled as *Informative* or *Not Informative*, depending on whether the tweet is useful for humanitarian aid purposes or not useful. *(2) Humanitarian:* We use the 5-class version of this data (Ofli et al., 2020) to alleviate the skewed label distribution. *(3) Damage Assessment.* We use a 2-class version of this data, similar to prior works (Li et al., 2018d). Each tweet image is labeled as depicting *Damage* or *No Damage*.

Although a significant amount of prior work has focused on multimodal tweet classification on CrisisMMD, directly comparing our approach to these methods is challenging, mainly because of the use of different splits for training and evaluation or different setups of the tasks. For example, Sosea et al. (2021) uses different splits and a 3-class version of the humanitarian class. Zou et al. (2021) uses different splits as well and a 4-class version of the humanitarian task. Dinani and Caragea (2021) focuses on improving performance for specific disasters and the data is divided disaster-wise. To this end, in this paper we employ 2 setups. In the first setup (Subsection 5.1), we create our own splits which we release alongside our data. We show the number of examples from the train, development, and test sets for the 3 tasks in CrisisMMD in Table 1 and we provide the class distributions in Table 2. Moreover, to validate our approach against some prior work with publicly released splits, we also

experiment with the splits released by Ofli et al. (2020) (Subsection 5.2).

4.2 Unlabeled Data

We show that, by using text queries and preprocessing for collecting the unlabeled corpus, the performance of FixMatch can be improved even though the two datasets are not sampled from the same distribution. We used the Twitter Streaming API with a list of relevant keywords for the text in the training dataset. Then we selected $122k$ unique tweets containing both text and images that do not overlap with CrisisMMD.

The tweets were crawled from Twitter using the Twitter Streaming API (with keywords such as #hurricaneharvey, #harvey, #hurricane, #earthquake) during the following disasters that happened in 2017: Hurricane Harvey, Hurricane Irma, Hurricane Maria, Mexico Earthquake, and Chiapas Earthquake. This collection was filtered for disaster relevance using a Naive Bayes classifier trained on CrisisLexT6 (Olteanu et al., 2014) to ensure that it mostly contained tweets relevant to disasters. Subsequently, duplicate tweets, retweets and non-English tweets were removed. Finally, we selected only tweets that contained both an image and text.

In addition, we used several methods to clean and filter out duplicates between our dataset and CrisisMMD. This is done in order to make sure that test samples (from CrisisMMD) are not seen during training, not even as unlabeled examples (as part of our unlabeled dataset). First, we removed all retweets (tweets with the "RT" token), and normalized the texts removing characters repetitions (all consecutive identical characters of size > 2 are reduced to only 2 characters) and user mentions. Next, we removed duplicates using the drop_duplicates function from the pandas library.

The resulting unlabeled corpus will be made publicly available.

4.3 Data Augmentations

Data augmentations play a vital part in our FixMatch (Sohn et al., 2020) framework. Given the multimodal nature of our model, we experiment with both text and image augmentations. For image augmentations, we follow FixMatch and use a standard flip-and-shift as a weak augmentation and RandAugment (Cubuk et al., 2020) as strong augmentation. For the textual modality, we investigate two different techniques:

- Easy Data Augmentation (EDA) (Wei and Zou, 2019), which randomly applies 4 possible operators: synonym replacement, random insertion of a word, random swap of 2 words or a random deletion of a word. We used the EDA framework for applying these transformations on 10% of the words in each text.

- Backtranslation (Edunov et al., 2018) was used previously in UDA (Xie et al., 2019) and MixText (Chen et al., 2020). It consists of translating a sentence to another language and than back to the original language, aiming to obtain a new example different from the original text but keeping the same meaning. Inspired by MixText (Chen et al., 2020), we use FairSeq (Ott et al., 2019) with Russian as an intermediate language and random sampling with 0.9 temperature instead of beam search in order to ensure the diversity of the augmentations.

4.4 Experimental Setup

To separately assess the impact of using multimodal data and of introducing text augmentations, we conduct our experiments in two stages. First, to ensure a fair comparison with the ResNet-based models, which only use the image modality, we experimented with versions of MMBT-based models where no text augmentation is used ($\mathcal{B}$ is the identity function). Second, we analyze the impact of augmenting each modality separately or performing both text and image augmentations. We propose the following Fixmatch adaptations: **1)** $FixMatchLS_{img}$ solely augments the image, **2)** $FixMatchLS_{eda}$ only augments the text using EDA, **3)** $FixMatchLS_{img+eda}$ augments both modalities, using EDA for text augmentation, and **4)** $FixMatchLS_{img+bt}$ augments both modalities, using back-translation for text augmentation.

To test the limits of our approach, we also experiment with few labeled training examples ($250/500$) per class on the Informative task. (Subsection 5.3). All hyperparameters and model setups are available in Subsection 4.5. To attain statistically significant results, we ran each experiment 5 times and report the average of the results. We used 4 Nvidia V100 GPUs to train our models. One experiment takes roughly 20 hours to complete on a single GPU. To improve reproducibility, we will release the splits for each task alongside our code.

DATASET	SIZE	TRAIN	DEV	TEST
INFORMATIVE	13494	10795 (80%)	1349 (10%)	1350 (10%)
DAMAGE	6089	4262 (70%)	913 (15%)	914 (15%)
HUMANITARIAN	8079	6126 (75.8%)	998 (12.4%)	955 (11.8%)

Table 1: Data splits for each task.

DATASET	INFORMATIVE	DAMAGE	HUMANITARIAN
Labels	*uninformative* (55%) *informative* (45%)	*no damage* (70%) *damage* (30%)	*not humanitarian* (53%) *other relevant information* (22%) *rescue volunteering or donation effort* (15%) *infrastructure and utility damage* (9%) *affected individuals* (1%)

Table 2: Labels distribution for each task.

4.5 Hyperparameters

First, we tried to find the best *FixMatch* setup for our experiments (without our extension). To achieve this, we experimented with a variety of setups, by manually tuning the *FixMatch* hyperparameters and choosing the values that yield the best F1 score. The values that were tested for each parameter are detailed in Appendix A. The obtained values that we used in all the reported results are the following: ratio between unlabeled and labeled examples $\mu = 7$, weight of the unlabeled loss $\lambda_u = 1$, image size 224x224, dropout 0.2, exponential moving average (EMA) with decay 0.999, learning rate 10^{-5} with *ReduceOnPlateau* schedule and *Adam* optimizer, confidence threshold $\tau = 0.7$, batch size of 8 with 16 gradient accumulation steps. For image augmentation we used random horizontal flip as weak augmentation and *RandAugment* as strong augmentation in all our experiments.

We apply the best hyperparameters found for the classic *FixMatch* algorithm to our extended *FixMatch LS* version. Our changes are:

- we used *soft labels* instead of hard pseudo-labels for the unlabeled data;
- we used a *linear schedule* for the unlabeled loss weight λ_u.

Note that replacing pseudo labels with soft labels for the unlabeled data completely removes the confidence threshold parameter, τ. However, introducing the linear schedule $\lambda_u(t) = c * t$ for the unlabeled loss adds one extra parameter, c. This is the only hyperparameter tuned for *FixMatch LS*. We used $\lambda_u(t) = 2 * t$ in all the experiments.

5 Results

5.1 Our data split

As it can be seen in Table 3, our enhanced *FixMatch* models, which use soft-labels and a linear schedule for weighting the unlabeled loss, consistently outperform all the other models on all tasks. On the *Informative* task, *MMBT FixMatch LS* improves the F1 performance of the supervised *MMBT Aug* model by as much as 3.5%. Interestingly, on the *Humanitarian* task, the *MMBT FixMatch* approach, which uses hard labels and a constant loss weighting, obtains similar performance to *MMBT Aug*, which uses no unlabeled data. We attribute this to the nature of the humanitarian task, where the boundary between classes may not be well defined, i.e., an example annotated with class y_1 can exhibit characteristics specific to a different class y_2. We argue that the use of the "hard labeling" mechanism for these types of tasks can lead to poor model performance. On the other hand, the *MMBT FixMatch LS* manages to prevent this shortcoming, and obtains an F1 increase of 1% over the *MMBT Aug* model. Finally, on the *Damage* task, we observe that the *ResNet* and the *MMBT* perform similarly, which is not surprising, given that the examples in this task were annotated based only on the image in the tweet. However, similar to the *Informative* task, the best semi-supervised approach outperforms the other method by as much as 2.9% F1.

5.2 Official data split

Table 4 shows the improvement obtained for the best model so far (*MMBT FixMatch LS*) with the introduction of text augmentation. Here, in order to enable a fair comparison with other methods, we test our best performing approach, $FixMatchLS$ using the data splits introduced by Ofli et al. (2020).

MODEL	INFORMATIVE			DAMAGE			HUMANITARIAN		
	P	R	F1	P	R	F1	P	R	F1
RESNET AUG	0.767	0.767	0.766	0.861	0.863	0.858	0.804	0.812	0.806
RESNET FIXMATCH	0.793	0.793	0.793	0.886	0.887	0.886	0.820	0.820	0.816
RESNET FIXMATCH LS	0.804	0.804	0.804	**0.887**	**0.888**	**0.887**	0.829	0.825	0.819
MMBT AUG	0.786	0.785	0.785	0.865	0.867	0.865	0.865	0.862	0.863
MMBT FIXMATCH	0.808	0.806	0.806	0.882	0.882	0.882	0.865	0.865	0.864
MMBT FIXMATCH LS	**0.820**	**0.820**	**0.820**	0.885	0.882	0.883	**0.873**	**0.872**	**0.872**

Table 3: Results on CrisisMMD tasks using image augmentations - best results for each task are highlighted in **bold**.

MODEL	INFORMATIVE			HUMANITARIAN		
	P	R	F1	P	R	F1
OFLI ET AL. (2020) - TXT	0.810	0.810	0.809	0.700	0.700	0.677
OFLI ET AL. (2020) - IMG	0.831	0.833	0.832	0.764	0.768	0.763
OFLI ET AL. (2020)	0.841	0.840	0.842	0.785	0.780	0.783
BIDARI (2021)	-	-	-	0.860	0.830	0.840
PRANESH ET AL. (2021)	-	-	-	-	-	0.855
ALAM ET AL. (2021) - Noisy Student - (*) - IMG	0.878	0.878	0.876	0.786	0.783	0.783
DINANI AND CARAGEA (2021) (*) - IMG	0.838	0.843	0.837	-	-	-
ZOU ET AL. (2021) (*)	0.875	0.876	0.875	0.872	0.911	0.891
SOSEA ET AL. (2021)(*)	-	-	-	0.950	0.920	0.940
$MMBT(supervised)$	0.887	0.888	0.886	0.865	0.862	0.863
$FixMatchLS_{img}$	0.901	0.901	0.899	0.873	0.872	0.872
$FixMatchLS_{eda}$	0.897	0.896	0.894	0.878	0.877	0.877
$FixMatchLS_{img+eda}$	0.907	0.906	0.904	**0.885**	**0.881**	**0.881**
$FixMatchLS_{img+bt}$	**0.910**	**0.908**	**0.905**	0.880	0.879	0.878

Table 4: Comparison of proposed method with state of the art models on official split of CrisisMMD. An asterisk at the end (*) means that the paper uses different splits. Best results are highlighted in **bold**, for the official split only.

MODEL	INFORMATIVE 250/CLASS			INFORMATIVE 500/CLASS		
	P	R	F1	P	R	F1
$MMBT(supervised)$	0.666	0.667	0.666	0.713	0.704	0.705
$FixMatchLS_{img}$	0.695	0.688	0.689	0.741	0.730	0.730
$FixMatchLS_{eda}$	0.687	0.673	0.673	0.741	0.731	0.722
$FixMatchLS_{img+eda}$	0.701	0.702	0.701	0.759	0.756	0.756
$FixMatchLS_{img+bt}$	**0.744**	**0.742**	**0.743**	**0.772**	**0.759**	**0.760**

Table 5: Results on CrisisMMD, Informative task, with few labeled examples per class - best results are highlighted in **bold**.

IMAGE	MODEL	LABEL	
		informative	*not informative*
(a)	MMBT AUG	0.71	0.29
	FIXMATCH LS	0.09	0.91
(c)	MMBT AUG	0.24	0.76
	FIXMATCH LS	0.98	0.02

Table 6: Examples of predictions for the Informative Task

All the methods without an asterisk (*) in this table use the same data splits, so they are directly comparable to one another and to our approach. However, as the official splits for multimodal data in CrisisMMD (Alam et al., 2018c) were released in a subsequent work (Ofli et al., 2020), many approaches created their own splits, as we did in the Subsection 5.1. Although they are not directly comparable to us, especially because they use fewer classes for the humanitarian task, we also show some of these results in Table 4 and mark them with an asterisk (*).

First, we observe that our supervised $MMBT$ method performs better than the best comparable baselines (i.e., Ofli et al. (2020) for the Informative task and Pranesh et al. (2021) for the Humanitarian task). Second, we note that augmenting a single modality (i.e., either text or image) improves performance on both tasks, by 1.4% F1 on Humanitarian and 1.3% F1 on Informative. Critically, we obtain the best results when employing augmentations for both modalities simultaneously. Specifically, $FixMatchLS_{img+eda}$ outperforms both $FixMatchLS_{img}$ and $FixMatchLS_{eda}$. Third, we observe that the best text augmentation is task-dependent. For example, $FixMatchLS_{img+eda}$ performs better on the Humanitarian task, while $FixMatchLS_{img+bt}$ is the best method for the Informative task.

Finally, there are two baselines that report higher performance on the Humanitarian task, namely Zou et al. (2021) and Sosea et al. (2021). However, as previously explained, the results are not directly comparable, as they use different versions of the Humanitarian task, with 3 and 4 classes, respectively, instead of 5 classes, as introduced in Ofli et al. (2020), which makes the task a lot easier for them.

5.3 Low-data regimes

To test the limits of our approach, we also experiment with few labeled examples $(250/500)$ per class on the informative task, as shown in Table 5. We emphasize that our SSL methods perform substantially better than baselines in these low-resource settings. This is extremely valuable for disaster-related classification, where abundant data at the time of the disaster is hard-to-acquire. Specifically, our results show that, while augmenting the image is more important than augmenting the text in low-data regimes ($FixMatchLS_{img}$ performs better than $FixMatchLS_{eda}$), it is once again clear that augmenting both modalities is always the best option. Using back-translation instead of EDA gives the best results, obtaining up to 7.7% F1 improvement over the supervised approach.

We emphasize that all improvements of the enhanced *FixMatch* over baselines in this paper are statistically significant, according to a t-test with $p < 0.01$. These results show the feasibility of our proposed *FixMatch* variant: using *cheap to acquire* unlabeled data, we improve the performance of supervised models significantly.

5.4 Error Analysis

We investigate common errors of the models that use no unlabeled data, which are corrected by our *FixMatch* models. To this end, we first sample 20 such examples for each CrisisMMD task, followed by manually inspecting the output probabilities and the contents of the image and text. We show some examples together with the corresponding ground truths in Figure 1, and provide comparisons between predictions of the *MMBT Aug* and the *FixMatch LS* model in Tables 6 and 7.

We observed a few patterns. First, we spotted some erroneous predictions due to semantic disparities between the textual and the image modalities (i.e., the image and text pinpoint to different labels, hence the final label is subjective). An example is shown in Figure 1b. Second, we encountered a significant number of examples where the image modality is distorted, or contains noise. For instance, in Figure 1c, the photo contains perturbations (i.e., the rain drops) that hinder the capability to observe the main focus of the picture: a *collapsed huge crane*. Third, we observe some examples which contain characteristics specific to more than one class. In Figure 1e, even though the main focus of the tweet is on *Rescue and volunteering* efforts, the image also exhibits traits of the *Infrastructure and utility damage* class: a destroyed building.

Our proposed *FixMatch* variant is able to correct these types of errors. Moreover, the *FixMatch* model is confident in its predictions, usually assigning a probability over 90% to the correct class.

6 Limitations

While our approach provides significant improvements on all CrisisMMD tasks, we also have to acknowledge the limitations of the proposed method. As it generally is the case with semi-supervised approaches, the training time is significantly increased, as more data needs to be passed through

IMAGE	MODEL	LABEL				
		not hum.	*other*	*rescue*	*damage*	*affected*
(b)	MMBT AUG	0.36	0.06	0.04	0.51	0.09
	FIXMATCH LS	0.89	0.01	0.02	0.07	0.01
(d)	MMBT AUG	0.02	0.03	0.16	0.78	0.01
	FIXMATCH LS	0.03	0.03	0.90	0.01	0.03
(e)	MMBT AUG	0.01	0.01	0.02	0.95	0.01
	FIXMATCH LS	0.01	0.01	0.93	0.04	0.01

Table 7: Examples of predictions for the Humanitarian Task

the model until convergence, comparing to a supervised approach. Regarding our method of collecting unlabeled data by searching for relevant keywords, although it is generic and could be applied to datasets from other domains, it is limited for datasets containing tweets. For other types of datasets, obtaining a relevant unlabeled corpus in the same manner could be more challenging.

7 Conclusion

We extended *FixMatch* to multimodal data and proposed two improvements. We applied the improved *FixMatch* on three disaster-centric multimodal tweet classification tasks, and showed that the approach can leverage large unlabeled data to improve supervised model performance. Our semi-supervised approach is general enough and can be easily applied to other datasets, being at the same time very efficient as it does not add any inference complexity to the base model.

Acknowledgements

We thank our anonymous reviewers for their constructive feedback and suggestions. We also thank the National Science Foundation (NSF) for support from the research grants IIS-1741345 and IIS-1912887, UEFISCDI for support from the research grant EEA-RO- 2018-0496, and Amazon Web Services for computational resources used in this study. Any opinions, findings, and conclusions expressed here are those of the authors and do not necessarily reflect the views of NSF or UEFISCDI.

References

Mahdi Abavisani, Liwei Wu, Shengli Hu, Joel Tetreault, and Alejandro Jaimes. 2020. Multimodal categorization of crisis events in social media. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 14679–14689.

Mansi Agarwal, Maitree Leekha, Ramit Sawhney, and Rajiv Ratn Shah. 2020. Crisis-dias: Towards multimodal damage analysis-deployment, challenges and assessment. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pages 346–353.

Firoj Alam, Tanvirul Alam, Muhammad Imran, and Ferda Ofli. 2021. Robust training of social media image classification models for rapid disaster response. *arXiv preprint arXiv:2104.04184*.

Firoj Alam, Muhammad Imran, and Ferda Ofli. 2017. Image4act: Online social media image processing for disaster response. In *Proceedings of the 2017 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining 2017*, pages 601–604.

Firoj Alam, Shafiq Joty, and Muhammad Imran. 2018a. Graph based semi-supervised learning with convolution neural networks to classify crisis related tweets. In *Twelfth International AAAI Conference on Web and Social Media*.

Firoj Alam, Ferda Ofli, and Muhammad Imran. 2018b. Crisismmd: Multimodal twitter datasets from natural disasters. In *Proceedings of the 12th International AAAI Conference on Web and Social Media (ICWSM)*.

Firoj Alam, Ferda Ofli, and Muhammad Imran. 2018c. Crisismmd: Multimodal twitter datasets from natural disasters. In *Proc. of the International AAAI Conference on Web and Social Media*, ICWSM, Stanford, California, USA.

Firoj Alam, Ferda Ofli, and Muhammad Imran. 2018d. Processing social media images by combining human and machine computing during crises. *International Journal of Human-Computer Interaction*, 34(4):311–327.

Zahra Ashktorab, Christopher Brown, Manojit Nandi, and Aron Culotta. 2014. Tweedr: Mining twitter to inform disaster response. In *ISCRAM*, pages 269–272.

David Berthelot, Nicholas Carlini, Ekin D Cubuk, Alex Kurakin, Kihyuk Sohn, Han Zhang, and Colin Raffel. 2019a. Remixmatch: Semi-supervised learning with distribution alignment and augmentation anchoring. *arXiv preprint arXiv:1911.09785*.

David Berthelot, Nicholas Carlini, Ian Goodfellow, Nicolas Papernot, Avital Oliver, and Colin A Raffel. 2019b. Mixmatch: A holistic approach to semi-supervised learning. In *Advances in Neural Information Processing Systems*, pages 5049–5059.

Melissa Bica, Leysia Palen, and Chris Bopp. 2017. Visual representations of disaster. In *Proceedings of the 2017 ACM Conference on Computer Supported Cooperative Work and Social Computing*, CSCW '17, pages 1262–1276.

Sumit Bidari. 2021. *Categorization of Disaster Related Tweets using Multimodal Approach*. Ph.D. thesis, Pulchowk Campus.

Neha Chaudhuri and Indranil Bose. 2020. Exploring the role of deep neural networks for post-disaster decision support. *Decision Support Systems*, 130:113234.

Jiaao Chen, Zichao Yang, and Diyi Yang. 2020. Mixtext: Linguistically-informed interpolation of hidden space for semi-supervised text classification. *arXiv preprint arXiv:2004.12239*.

Ekin D Cubuk, Barret Zoph, Jonathon Shlens, and Quoc V Le. 2020. Randaugment: Practical automated data augmentation with a reduced search space. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops*, pages 702–703.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*.

Soudabeh Taghian Dinani and Doina Caragea. 2021. Disaster image classification using capsule networks. In *2021 International Joint Conference on Neural Networks (IJCNN)*, pages 1–8. IEEE.

Sergey Edunov, Myle Ott, Michael Auli, and David Grangier. 2018. Understanding back-translation at scale. *arXiv preprint arXiv:1808.09381*.

Markus Enenkel, Sofia Martinez Saenz, Denyse S Dookie, Lisette Braman, Nick Obradovich, and Yury Kryvasheyeu. 2018. Social media data analysis and feedback for advanced disaster risk management. In *Social Web in Emergency and Disaster Management*.

Akash Kumar Gautam, Luv Misra, Ajit Kumar, Kush Misra, Shashwat Aggarwal, and Rajiv Ratn Shah. 2019. Multimodal analysis of disaster tweets. In *2019 IEEE Fifth International Conference on Multimedia Big Data (BigMM)*, pages 94–103. IEEE.

Xiangyang Guan and Cynthia Chen. 2014. Using social media data to understand and assess disasters. *Natural hazards*, 74(2):837–850.

Haiyan Hao and Yan Wang. 2020. Leveraging multimodal social media data for rapid disaster damage assessment. *International Journal of Disaster Risk Reduction*, page 101760.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. 2016. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778.

Muhammad Imran, Carlos Castillo, Fernando Diaz, and Sarah Vieweg. 2015. Processing social media messages in mass emergency: A survey. *ACM Computing Surveys (CSUR)*, 47(4):67.

Douwe Kiela, Suvrat Bhooshan, Hamed Firooz, and Davide Testuggine. 2019. Supervised multimodal bitransformers for classifying images and text. *arXiv preprint arXiv:1909.02950*.

Yury Kryvasheyeu, Haohui Chen, Nick Obradovich, Esteban Moro, Pascal Van Hentenryck, James Fowler, and Manuel Cebrian. 2016. Rapid assessment of disaster damage using social media activity. *Science advances*, 2(3).

Ryan Lagerstrom, Yulia Arzhaeva, Piotr Szul, Oliver Obst, Robert Power, Bella Robinson, and Tomasz Bednarz. 2016. Image classification to support emergency situation awareness. *Frontiers in Robotics and AI*, 3:54.

Samuli Laine and Timo Aila. 2016. Temporal ensembling for semi-supervised learning. *arXiv preprint arXiv:1610.02242*.

Dong-Hyun Lee et al. 2013. Pseudo-label: The simple and efficient semi-supervised learning method for deep neural networks. In *Workshop on challenges in representation learning, ICML*, volume 3, page 896.

Hongmin Li, Doina Caragea, and Cornelia Caragea. 2021. Combining self-training with deep learning for disaster tweet classification. In *The 18th International Conference on Information Systems for Crisis Response and Management (ISCRAM 2021)*.

Hongmin Li, Doina Caragea, Cornelia Caragea, and Nic Herndon. 2018a. Disaster response aided by tweet classification with a domain adaptation approach. *Journal of Contingencies and Crisis Management*, 26(1):16–27.

Hongmin Li, Doina Caragea, Xukun Li, and Cornelia Caragea. 2018b. Comparison of word embeddings and sentence encodings as generalized representations for crisis tweet classification tasks. In *Proceedings of ISCRAM Asia Pacific*, page 13, Wellington, New Zealand.

Hongmin Li, Oleksandra Sopova, Doina Caragea, and Cornelia Caragea. 2018c. Domain adaptation for crisis data using correlation alignment and self-training. *International Journal of Information Systems for Crisis Response and Management (IJISCRAM)*, 10(4):1–20.

Xukun Li and Doina Caragea. 2020. Improving disaster-related tweet classification with a multimodal approach. In *ISCRAM 2020 Conference Proceedings–17th International Conference on Information Systems for Crisis Response and Management*.

Xukun Li, Doina Caragea, Cornelia Caragea, Muhammad Imran, and Ferda Ofli. 2019a. Identifying disaster damage images using a domain adaptation approach. In *Proceedings of the 16th International Conference on Information Systems for Crisis Response and Management (ISCRAM 2019)*, Valencia, Spain.

Xukun Li, Doina Caragea, Huaiyu Zhang, and Muhammad Imran. 2018d. Localizing and quantifying damage in social media images. In *2018 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining (ASONAM)*, pages 194–201. IEEE.

Xukun Li, Doina Caragea, Huaiyu Zhang, and Muhammad Imran. 2019b. Localizing and quantifying infrastructure damage using class activation mapping approaches. *Social Network Analysis and Mining*, 9(1):44.

Reza Mazloom, Hongmin Li, Doina Caragea, Cornelia Caragea, and Muhammad Imran. 2019. A hybrid domain adaptation approach for identifying crisis-relevant tweets. *International Journal of Information Systems for Crisis Response and Management (IJISCRAM)*, 11(2):1–19.

Geoffrey J McLachlan. 1975. Iterative reclassification procedure for constructing an asymptotically optimal rule of allocation in discriminant analysis. *Journal of the American Statistical Association*, 70(350):365–369.

Takeru Miyato, Andrew M Dai, and Ian Goodfellow. 2016. Adversarial training methods for semi-supervised text classification. *arXiv preprint arXiv:1605.07725*.

Takeru Miyato, Shin-ichi Maeda, Masanori Koyama, Ken Nakae, and Shin Ishii. 2015. Distributional smoothing with virtual adversarial training. *arXiv preprint arXiv:1507.00677*.

Hussein Mouzannar, Yara Rizk, and Mariette Awad. 2018. Damage identification in social media posts using multimodal deep learning. In *Proceedings of the 15th International Conference on Information Systems for Crisis Response and Management (ISCRAM 2018)*, Rochester, NY.

Ganesh Nalluru, Rahul Pandey, and Hemant Purohit. 2019. Relevancy classification of multimodal social media streams for emergency services. In *2019 IEEE International Conference on Smart Computing (SMARTCOMP)*, pages 121–125. IEEE.

Venkata Kishore Neppalli, Cornelia Caragea, and Doina Caragea. 2018. Deep neural networks versus naive bayes classifiers for identifying informative tweets during disasters. In *Proceedings of the 15th Annual Conference for Information Systems for Crisis Response and Management (ISCRAM)*.

Dat T Nguyen, Ferda Ofli, Muhammad Imran, and Prasenjit Mitra. 2017. Damage assessment from social media imagery data during disasters. In *Proceedings of the 2017 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining 2017*, pages 569–576. ACM.

Ferda Ofli, Firoj Alam, and Muhammad Imran. 2020. Analysis of social media data using multimodal deep learning for disaster response. In *17th International Conference on Information Systems for Crisis Response and Management*. ISCRAM.

Alexandra Olteanu, Carlos Castillo, Fernando Diaz, and Sarah Vieweg. 2014. Crisislex: A lexicon for collecting and filtering microblogged communications in crises.

Myle Ott, Sergey Edunov, Alexei Baevski, Angela Fan, Sam Gross, Nathan Ng, David Grangier, and Michael Auli. 2019. fairseq: A fast, extensible toolkit for sequence modeling. *arXiv preprint arXiv:1904.01038*.

Raj Ratn Pranesh, Ambesh Shekhar, and Anish Kumar. 2021. Exploring multimodal features and fusion strategies for analyzing disaster tweets.

Antti Rasmus, Harri Valpola, Mikko Honkala, Mathias Berglund, and Tapani Raiko. 2015. Semi-supervised learning with ladder networks. *arXiv preprint arXiv:1507.02672*.

Yara Rizk, Hadi Samer Jomaa, Mariette Awad, and Carlos Castillo. 2019. A computationally efficient multimodal classification approach of disaster-related twitter images. In *Proceedings of the 34th ACM/SIGAPP symposium on applied computing*, pages 2050–2059.

Mehdi Sajjadi, Mehran Javanmardi, and Tolga Tasdizen. 2016. Regularization with stochastic transformations and perturbations for deep semi-supervised learning. *Advances in neural information processing systems*, 29:1163–1171.

Karen Simonyan and Andrew Zisserman. 2015. Very deep convolutional networks for large-scale image recognition. In *Proceedings of ICLR*.

Kihyuk Sohn, David Berthelot, Chun-Liang Li, Zizhao Zhang, Nicholas Carlini, Ekin D Cubuk, Alex Kurakin, Han Zhang, and Colin Raffel. 2020. Fixmatch: Simplifying semi-supervised learning with consistency and confidence. *arXiv preprint arXiv:2001.07685*.

Tiberiu Sosea, Iustin Sirbu, Cornelia Caragea, Doina Caragea, and Traian Rebedea. 2021. Using the image-text relationship to improve multimodal disaster tweet classification. In *The 18th International Conference on Information Systems for Crisis Response and Management (ISCRAM 2021)*.

Antti Tarvainen and Harri Valpola. 2017. Mean teachers are better role models: Weight-averaged consistency targets improve semi-supervised deep learning results. *arXiv preprint arXiv:1703.01780*.

Ethan Weber, Nuria Marzo, Dim P Papadopoulos, Aritro Biswas, Agata Lapedriza, Ferda Ofli, Muhammad Imran, and Antonio Torralba. 2020. Detecting natural disasters, damage, and incidents in the wild. *arXiv preprint arXiv:2008.09188*.

Jason Wei and Kai Zou. 2019. Eda: Easy data augmentation techniques for boosting performance on text classification tasks. *arXiv preprint arXiv:1901.11196*.

Qizhe Xie, Zihang Dai, Eduard Hovy, Minh-Thang Luong, and Quoc V Le. 2019. Unsupervised data augmentation for consistency training. *arXiv preprint arXiv:1904.12848*.

Qizhe Xie, Minh-Thang Luong, Eduard Hovy, and Quoc V Le. 2020. Self-training with noisy student improves imagenet classification. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 10687–10698.

Jie Yin, Andrew Lampert, Mark Cameron, Bella Robinson, and Robert Power. 2012. Using social media to enhance emergency situation awareness. *IEEE intelligent systems*, (6):52–59.

Faxi Yuan and Rui Liu. 2018. Feasibility study of using crowdsourcing to identify critical affected areas for rapid damage assessment: Hurricane matthew case study. *International Journal of Disaster Risk Reduction*.

Hongyi Zhang, Moustapha Cisse, Yann N Dauphin, and David Lopez-Paz. 2017. mixup: Beyond empirical risk minimization. *arXiv preprint arXiv:1710.09412*.

Zhiqiang Zou, Hongyu Gan, Qunying Huang, Tianhui Cai, and Kai Cao. 2021. Disaster image classification by fusing multimodal social media data. *ISPRS International Journal of Geo-Information*, 10(10):636.

A Hyperparameters

First, we tried to find the best *FixMatch* setup for our experiments (without our extension). To achieve this, we experimented with a variety of setups, by manually tuning the *FixMatch* hyperparameters and choosing the values that yield the best F1 score:

- For the ratio μ between unlabeled and labeled examples we tried values from the set {3, 5, 7}. We observed that setting μ to 7 produced the best results. We did not try values bigger that 7 due to computation limitations. However, 7 is the reported best μ in the original *FixMatch* paper, too.
- For the weight of the unlabeled loss, λ_u, we experimented with values in the set $\{1, 10, 50, 100\}$, and obtained the best results with value 1 (similar to the original paper).
- For image preprocessing, we cropped and rescaled all images to 224x224 size. We also tried to reduce the size of the images to 96x96 to improve computational performance, but the results were heavily affected.
- For image augmentation we used random horizontal flip as weak augmentation and *RandAugment* as strong augmentation in all our experiments.
- Initially, the original paper used no dropout, but we observed that adding 0.2 dropout improved the results.
- Exponential moving average (EMA) with decay 0.999 was kept as in the original paper. We experimented with a smaller decay or without EMA, but this negatively impacted the performance.
- Instead of SGD and *cosine learning rate schedule*, we used *Adam* with a *ReduceOnPlateau schedule*, which improved results.
- We experimented with learning rates from the set $\{10^{-5}, 5 \times 10^{-5}, 10^{-4}\}$, and picked 10^{-5} as the optimal value.
- For the confidence threshold τ, we found that 0.7 was the best for our tasks. This is compatible with the value chosen in the original paper on the *ImageNet* dataset. We experimented with values in the set $\{0.5, 0.7, 0.85, 0.95\}$.
- Due to computation limitations, we used a batch size of 8 with 40 gradient accumulation steps in all our experiments.

We apply the best hyperparameters found for the classic *FixMatch* algorithm to our extended *FixMatch LS* version. Our changes are:

- we used *soft labels* instead of hard pseudo-labels for the unlabeled data
- we used a *linear schedule* for the unlabeled loss weight λ_u

Note that replacing pseudo labels with soft labels for the unlabeled data completely removes the confidence threshold parameter, τ. However, introducing the linear schedule $\lambda_u(t) = c * t$ for the unlabeled loss adds one extra parameter, c. This is the only hyperparameter tuned for *FixMatch LS*. After experimenting with values in the set $\{1, 2, 3\}$, we choose $\lambda_u(t) = 2 * t$ to be our weight in all the experiments.

In order to attain statistically significant results, we ran each experiment 5 times and report the average of the results. We used 4 Nvidia V100 GPUs to train our models. One experiment takes roughly 20 hours to complete on a single GPU.

Automated Essay Scoring via Pairwise Contrastive Regression

Jiayi Xie*, Kaiwei Cai*, Li Kong, Junsheng Zhou†, Weiguang Qu

NLP Lab, Department of Computer and Electronic Information
Nanjing Normal University, China
{ tammyjiayixie, ckwcccwkc }@gmail.com
kongli@nnu.edu.cn, { zhoujs, wgqu }@njnu.edu.cn

Abstract

Automated essay scoring (AES) involves the prediction of a score relating to the writing quality of an essay. Most existing works in AES utilize regression objectives or ranking objectives respectively. However, the two types of methods are highly complementary. To this end, in this paper we take inspiration from contrastive learning and propose a novel unified Neural Pairwise Contrastive Regression (NPCR) model in which both objectives are optimized simultaneously as a single loss. Specifically, we first design a neural pairwise ranking model to guarantee the global ranking order in a large list of essays, and then we further extend this pairwise ranking model to predict the relative scores between an input essay and several reference essays. Additionally, a multi-sample voting strategy is employed for inference. We use Quadratic Weighted Kappa to evaluate our model on the public Automated Student Assessment Prize (ASAP) dataset, and the experimental results demonstrate that NPCR outperforms previous methods by a large margin, achieving the state-of-the-art average performance for the AES task[1].

1 Introduction

Automated Essay Scoring (AES) is to evaluate the quality of essays and score automatically by using computer technologies. Notably, reasonable grading can solve problems that consume much time and require a lot of human effort. What's more, providing feedback to learners can promote self improvement. It is one of the most important applications of natural language processing (NLP) and is widely required in the educational field.

Most existing methods typically recast AES as a regression task, where the goal is to predict the score of an essay (Taghipour and Ng, 2016; Dong and Zhang, 2016; Dong et al., 2017; Tay et al., 2018). Although some promising results have been achieved, these regression-based models cannot exploit the labelling information in the training data efficiently and directly. Besides, another line of research treats AES as a preference ranking problem with learning-to-rank methods. Yannakoudakis et al. (2011) first proposed to rank the pair of documents by extracting features; later, Chen and He (2013) transformed this task into a listwise ranking problem. Cummins et al. (2016) also performed transfer learning to rank two essays that are constrained to be from the same prompt.

However, most existing works in AES utilize regression objectives or ranking objectives respectively. As a matter of fact, the two types of methods are highly complementary. On the one hand, only using regression models for AES cannot explicitly model score relationships between essays in the training data. On the other hand, only using ranking-based models could not guarantee accurate scores. In effect, in real-life situations, when a teacher evaluates and grades a student's essay, he usually first compares it with one or multiple exemplar essays as reference and then gives a specific score for it.

Recently, Yang et al. (2020) presents the first work to combine regression and ranking in the AES task by applying a multi-loss method that optimizes regression loss and ranking loss jointly with a simple dynamic combination strategy. Nevertheless, it is actually quite difficult to determine the combination weights to achieve the tradeoff between the two optimization objectives. Additionally, the proposed batch-wise learning based ranking model sacrifices the accuracy and only ranks essays in each batch.

To address the above problems, in this paper we explore a unified framework for the AES task where both the regression objective and the ranking

* Equal contribution.
† Corresponding author.
[1]The source code is available at https://github.com/CarryCKW/AES-NPCR.

Proceedings of the 29th International Conference on Computational Linguistics, pages 2724–2733
October 12–17, 2022.

objective are optimized simultaneously, with the goal of incorporating the merits of two popular AES solutions. The key challenge here is how to integrate two significantly different optimization objectives into a single model with a single loss.

To this end, we take inspiration from contrastive learning (Yu et al., 2021; Chen et al., 2020) and propose a novel unified Neural Pairwise Contrastive Regression (NPCR) model that jointly optimizes the two objectives in a principled way. In a nutshell, the goal of contrastive learning is to learn a better representation space (Chen et al., 2020). In particular, for two given essays, the distance between similar essays from the same category should be small while the distance between dissimilar essays should be large, and the semantic relationship can be reflected by measuring the distance in the representation space. Thus, under the contrastive learning framework, the proposed model aims to map the input essays into the representation space and calculates the differences between essays by the relative scores. Specifically, we first design a neural pairwise ranking model to guarantee the global ranking order in a large list of essays, and then we further extend this neural pairwise ranking model to predict the relative scores between an input essay and several reference essays. Additionally, a multi-sample voting strategy is adopted for the inference for every input test essay.

We use Quadratic Weighted Kappa to evaluate our model on the Automated Student Assessment Prize (ASAP) dataset, and the experimental results demonstrate that the proposed model outperforms previous methods by a large margin and establishes new state-of-the-art on this public benchmark.

In summary, the contributions of this work can be concluded as follows: (1) To the best of our knowledge, we make the first attempt to explore a unified framework for the AES task that performs regression and ranking optimization simultaneously; (2) We propose a neural pairwise ranking model for AES that guarantees the global ranking order in a large list of essays; (3) Experimental results on the public dataset ASAP show that the proposed approach not only achieves the state-of-the-art average performance but also obtains better performance on almost all prompts compared to all baselines.

2 Background

2.1 Task Description

Automated essay scoring systems are used in evaluating and scoring student essays written based on a given prompt. The performance of these systems is assessed by comparing their scores assigned to a set of essays to human-assigned gold standard scores. Since the output of AES systems is usually a real-valued number, the task is often addressed as a supervised machine learning task (mostly by regression or preference ranking).

2.2 The Multi-loss Method for the Combination of Regression and Ranking

In order to take advantage of the complementarity of regression loss and ranking loss, Yang et al. (2020) proposes a multi-loss objective to fine-tune the BERT model for the AES task by using a simple dynamic optimizing strategy as Formula 1. However, it is very difficult to determine the suitable combination weights to achieve the tradeoff between the two losses.

$$L = \tau_e \times L_m + (1 - \tau_e) \times L_r \quad (1)$$

where L_m is the regression objective, L_r is the result of the batchwise loss function, and τ_e is the parameter that vary with the number of epoch.

Besides, Yang et al. (2020) uses a batch-wise approach ListNet which ranks a list of essays each time and measures the accuracy between the predicted ranking list and the ground truth label. The major defect of this method is that it can only rank essays in a batch and cannot guarantee precise global order.

3 Pairwise Contrastive Regression for AES

3.1 Methodology Overview

Traditionally, most existing works formulate AES as a regression task, where the input is an essay and the output is a predicted score relating to the writing quality of the essay (Taghipour and Ng, 2016; Dong and Zhang, 2016; Tay et al., 2018). Formally, given the input essay e with the score label s, the regression problem is to predict the score $\hat{s}$ based on the quality of input essay:

$$\hat{s} = R_\theta(F_W(e)) \quad (2)$$

where R_θ and F_W are the regressor model and the feature extractor parameterized by θ and W, respectively.

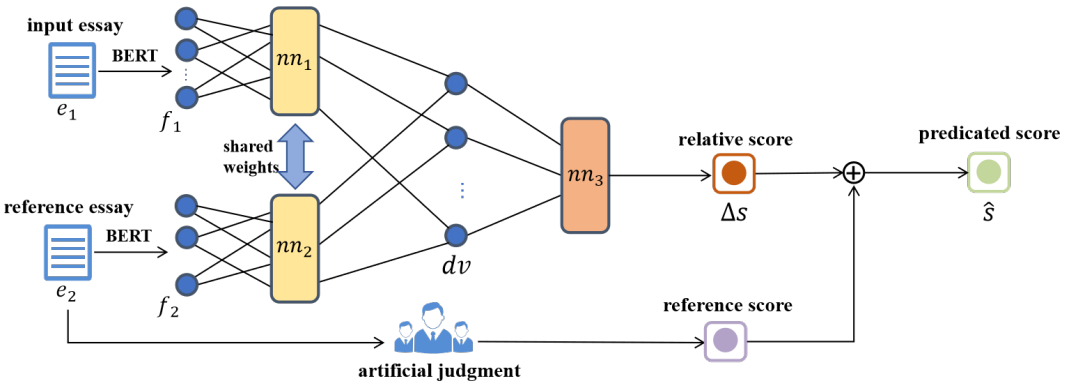

Figure 1: The overall framework of neural pairwise contrastive regression model for AES.

However, optimizing the regression objective alone is inadequate to make good use of the score label information in the training data. In contrast, the ranking-based methods could explicitly model score relationships between essays (Yannakoudakis et al., 2011). In order to take advantage of the complementarity of these two types of methods, we therefore propose to reformulate the AES problem as regressing relative score between the input and an exemplar. Let e_i denotes the input essay, and e_j denotes the reference essay with score label s_j, this regression problem can be re-written as:

$$\hat{s}_i = R_\theta(F_W(e_i, e_j)) + s_j \tag{3}$$

Note that the aim here is to predict the relative score, i.e. the difference of the scores between the input essay and a reference essay.

Technically speaking, the major challenge of successfully predicting the relative score lies in how to design effective regressor that takes as input a pair of essays rather than a single essay. In contrast to the single essay input, this regressor with the essay pair input should satisfy more characteristics, such as reflexivity and antisymmetry.

To achieve this, we propose a neural pairwise contrastive regression model for AES to predict the relative score. The overall framework of our method is illustrated in Figure 1. Methodologically, our pairwise contrastive regression model is actually a natural extension to a neural pairwise ranker for AES.

In order to clearly articulate our approach, in the following subsections, we first introduce the design of a neural pairwise ranker for AES in detail, and then we further extend it to a pairwise contrastive regressor to reach this goal.

3.2 Neural Pairwise Learning to Rank for AES

In this section, motivated by the DirectRanker (Köppel et al., 2019), we aim to design a neural pairwise ranking model for AES to predict a global ranking order given a large list of essays. To do this, given any two essays e_1 and e_2, we first define a partial order operator $e_1 \succeq e_2$ such that e_1 has higher score than e_2. In order to achieve a consistent and global order, this operator should satisfy three characteristics: *reflexivity*, *antisymmetry* and *transitivity*. Further, we use a ranking function $rf : \mathcal{F} \times \mathcal{F} \rightarrow \mathcal{R}$ over the feature space $\mathcal{F}$ to implement the operator:

$$x \succeq y :\Leftrightarrow rf(x, y) \geq 0, \ for\ x, y \in \mathcal{F} \tag{4}$$

Thus, the three characteristics of this operator can be defined through the function rf as follows:

(A) Reflexivity: $rf(x, x) = 0$

(B) Antisymmetry: $rf(x, y) = -rf(y, x)$

(C) Transitivity: $(rf(x, y) \geq 0 \wedge rf(y, z) \geq 0) \Rightarrow rf(x, z) \geq 0$

Particularly, to meet these requirements, the ranking function rf can be implemented by using a neural network with specific structure.

Firstly, in order to map an input essay e to low-dimensional vector space, we use BERT (Devlin et al., 2019) which can make full use of rich semantic information to obtain the text vector representation f:

$$h = BERT(e) \ \in \mathcal{R}^{r_h * |e|} \tag{5}$$

$$f = h_{[CLS]} \tag{6}$$

where h is the hidden representations, r_h is the dimension of the hidden state and $|e|$ represents the length of the input essay. The vector f, a hidden representation mapping to the special token $[CLS]$, is used as the text representation for the input essay e.

Next, as shown in Figure 1, the feature extraction part of the model includes two subnets nn_1 and nn_2 which are composed of multi-layer perceptron. The two subnets share the same structure and parameters like weights, biases, activation, etc. Then a difference vector of the two outputs from two subnets nn_1 and nn_2 can be simply calculated as follows:

$$dv = F_w(f_1) - F_w(f_2) \tag{7}$$

where f_1 and f_2 are the representation vectors of one essay pair, and F_w is the feature extractor parameterized by w.

After that, the difference vector dv is fed into the third subnet nn_3 which has only one output neuron. As shown by (Köppel et al., 2019), the antisymmetry can easily be guaranteed by choosing antisymmetric activation functions and removing the biases of the neuron.

In fact, it is easy to prove that the above three characteristics can be satisfied in our model. More specifically, we first utilize ϕ to define the antisymmetric activation function, i.e. $\phi(-x) = -\phi(x)$ for $\phi : \mathcal{R} \to \mathcal{R}$.

(I) The satisfaction of (II) means that (I) can be inferred.

$$rf(x, x) = -rf(x, x) \Rightarrow rf(x, x) \equiv 0 \tag{8}$$

(II) From the above mentioned, nn_1 and nn_2 have the consistent network structure, thus they employ the same function $g : \mathcal{F} \to \mathcal{R}^n$. Hence, for two input feature vectors $x, y \in \mathcal{F}$, (B) can be proved as follows:

$$\begin{aligned} rf(x, y) &= \phi[w(g(x) - g(y))] \\ &= -\phi[wg(y) - wg(x)] \\ &= -rf(y, x) \end{aligned} \tag{9}$$

where w is the weight vector.

(III) Assuming $x, y, z \in \mathcal{F}$, $rf(x, y) \geq 0$ and $rf(y, z) \geq 0$, the transitivity of the model can be testified by:

$$\begin{aligned} rf(x, z) &= \phi[w(g(x) - g(z))] \\ &= \phi[wg(x) - wg(y) + wg(y) - wg(z)] \\ &= rf(x, y) + rf(y, z) \geq 0 \end{aligned} \tag{10}$$

where w is the weight vector and g is defined as in (II). Hence, (C) is compliant.

3.3 Pairwise Contrastive Regression Model for AES

Next, we extend the neural pairwise ranking model illustrated in the previous section to form a pairwise contrastive regression model that predicts the relative score between an input essay and an reference essay. In fact, it is relatively straightforward to achieve this.

As the pairwise ranking model, the output of the third subnet nn_3 shown in Figure 1 is just a binary value. If we allow the output value of this subset to be a real value corresponding to a relative score and specify the second input essay as the reference sample, the converted model is actually a basic contrastive regression model. Then, the difference vector dv is fed into a fully connected neural network which consists of only one output neuron with antisymmetric activation and without a bias. Given the difference vector dv, the pairwise contrastive model can be simply defined as follows:

$$\Delta s = R_\theta(dv) \tag{11}$$

where Δs represents the relative score of any two essays and θ is the parameter of the regression model. This regression problem of the relative scores can be solved by minimizing the Mean Squared Error (MSE) loss that can be computed based on the predicted relative scores and the golden relative scores over the training data. The corresponding loss function for this pairwise contrastive regression is shown as follows:

$$L_r = \frac{1}{N}\sum_{i=1}^{N}(\Delta s_i - \Delta s_i^{'})^2 \tag{12}$$

where N refers to the total number of essay pairs, Δs_i and $\Delta s_i^{'}$ denote the predicted relative score and the golden relative score, respectively.

In principle, this contrastive regression model should satisfy three characteristics: *reflexivity*, *antisymmetry* and *accumulation*. Specifically, the *accumulation* can be defined as follows:

$$rf(x, y) + rf(y, z) = rf(x, z) \tag{13}$$

Obviously, both *reflexivity* and *antisymmetry* can easily be satisfied by adopting the same neural network architecture as in the previous section. Nevertheless, in theory, the *accumulation* in this contrastive regression model cannot be guaranteed by any neural network model itself. Thus, a learning goal of this contrastive regression model is to meet accumulation as much as possible.

With this end in view, how to select the essays pairs during training becomes critical. Therefore we design an effective selection strategy for constructing the training data. Particularly, we first arrange all essays in each prompt as a sequence according to the order in which they appear in the training data, then orderly pick every two adjacent essays in the sequence as a pair. Furthermore, an important additional step is imposed to cater the need of accumulation. To be specific, if we pick the essays pairs (e_i, e_j) and (e_j, e_k) as the training instances, we should also add the pair (e_i, e_k) into the training data in order to make the learned model meet the accumulation. Additionally, to make the input essay and the reference essay comparable, we tend to select the essays that shares the same prompt with the input essay as the references.

3.4 Inference

During inference, we employ a multi-sample voting strategy. Intuitively, the selected reference samples should be comparable to the input test essays. However, our dataset has eight prompts and different prompts have different relative score ranges. In order to solve the above problem, we select some sample essays which have the same prompt with the input essays.

Specifically, given an input essay e_{test}, we select M samples from the training datasets to construct M pairs using these M different samples $\{e^m_{train}\}^M_{m=1}$ whose scores are $\{s_{train}\}^M_{m=1}$. Then we will obtain M predicted scores and the final score of the input essay is the average of these M scores. The process of multi-sample voting can be summarized as follows:

$$\hat{s}^m_{test} = R_\theta(F_w(e_{test}, e^m_{train})) + s^m_{train} \quad (14)$$

$$\hat{s}_{test} = \frac{1}{M}\sum_{m=1}^{M} \hat{s}^m_{test},\ m = 1, 2, ..., M \quad (15)$$

where θ and w are the parameters of the pairwise contrastive regression model. $\hat{s}^m_{test}$ represents the m-th predicted score and $\hat{s}_{test}$ denotes the final predicted score of the input essay e_{test}.

Prompt ID	Essay Set Size	Original Score Range	Relative Score Range
1	1783	2-12	-10-10
2	1800	1-6	-5-5
3	1726	0-3	-3-3
4	1772	0-3	-3-3
5	1805	0-4	-4-4
6	1800	0-4	-4-4
7	1569	0-30	-30-30
8	723	0-60	-60-60

Table 1: The details of the ASAP dataset.

4 Experiments

4.1 Experimental Settings

4.1.1 Dataset

We use the widely used dataset ASAP (Automated Student Assessment Prize) for experimental evaluation. This comes from the competition which was organized and sponsored by the William and Flora Hewlett Foundation (Hewlett). This dataset contains eight prompts and has different genres and different number of essays, as described in Table 1. Following previous work, we also utilize 5-fold cross-validation to evaluate the model. In each run, we use 60%, 20% and 20% of the dataset for each prompt as training data, validation data and test set, which are provided by (Taghipour and Ng, 2016).

4.1.2 Evaluation Metrics

In this paper, we use the commonly used metric Quadratic Weighted Kappa (QWK) to measure the agreement between the artificial scores and the predicted results. Specially, let the essay set be scored on a scale of 1 to N, and the score from the expert is i while the predicted score of the model is j.

$$K = 1 - \frac{\sum_{i,j} w_{i,j} O_{i,j}}{\sum_{i,j} w_{i,j} E_{i,j}} \quad (16)$$

$$w_{i,j} = \frac{(i-j)^2}{(N-1)^2} \quad (17)$$

where w, O, E are matrices of weights, observed scores and expected scores, respectively. Furthermore, the value of $O_{i,j}$ represents the number of essays that receive a score i by the human rater and a score j by the AES system. And $E_{i,j}$ represents the outer product between two histogram vectors of the scores.

Dataset/Prompts									
Model	1	2	3	4	5	6	7	8	Avg
EASE(SVR)	0.781	0.630	0.621	0.749	0.782	0.771	0.727	0.534	0.699
EASE(BLRR)	0.761	0.621	0.606	0.742	0.784	0.775	0.730	0.617	0.705
ALL-MTL-cTAP (2016)	0.816	0.667	0.654	0.783	0.801	0.778	0.787	0.692	0.747
CNN+LSTM (2016)	0.821	0.688	0.694	0.805	0.807	0.819	0.808	0.644	0.761
LSTM-CNN-attent (2017)	0.822	0.682	0.672	0.814	0.803	0.811	0.801	0.705	0.764
SKIPFLOW (2018)	0.832	0.684	0.695	0.788	0.815	0.810	0.800	0.697	0.764
HISK+BOSWE (2018)	0.845	0.729	0.684	0.829	0.833	0.830	0.804	0.729	0.785
R^2BERT (2020)	0.817	0.719	0.698	0.845	0.841	0.847	**0.839**	0.744	0.794
NPCR	**0.856**	**0.750**	**0.756**	**0.851**	**0.847**	**0.858**	0.838	**0.779**	**0.817**

Table 2: The QWK evaluation scores on ASAP dataset, and the results of baselines are adapted from their original papers.

4.1.3 Implementation Details

Following previous work (Yang et al., 2020), we also use $BERT_{base}$ model for fair comparison. For tokenization and vocabulary, and we all use the preprocessing tools provided by the BERT model. For the limitation of our GPU memory, we set the max length of the essay is 512 words and the batch size is 5. We train our model for 80 epochs and select the best model based on the performance on the validation set. We use AdamW as our optimizer to train the model and the initial learning rate is set to $1e-5$. In addition, we normalize all relative scores to the range of $[0, 1]$ during training and the scores are rescaled back to the original score range for evaluation. Following previous work, we conduct the evaluation in prompt-specific fashion.

4.2 Overall Performance

In this section, we comprehensively compare our overall performance with the following state-of-the-art related methods that were evaluated on the dataset ASAP.

4.2.1 Baselines

EASE The major non deep learning system that we compare against is the Enhanced AI Scoring Engine (EASE). This system is publicly available and also achieved excellent results in the ASAP competition. Following previous works, we report the results of EASE with the settings of Support Vector Regression (SVR) and Bayesian Linear Ridge Regression (BLRR).

ALL-MTL-cTAP Cummins et al. (2016) used a constrained multi-task pairwise-preference learning method to achieve the representation of the essays.

CNN+LSTM Taghipour and Ng (2016) first designed a neural network model which used CNN for word sequence modeling and LSTM for text level modeling. Then the essay representation is achieved by mean of time pooling.

LSTM-CNN-attent Dong et al. (2017) proposed to use hierarchical neural networks with attention mechanism to extract features from sentences and documents.

SKIPFLOW Tay et al. (2018) proposed the model that considered neural coherence features within the context of an end-to-end neural framework to improve prediction.

HISK+BOSWE Cozma et al. (2018) combined string kernels and word embeddings to extract more semantic features and gained higher performance in both in-domain and cross-domain settings.

R^2BERT Yang et al. (2020) presented the first work that employed a multi-loss method to combine regression and ranking and to fine-tune BERT models in AES tasks.

4.2.2 Performance Comparison

Table 2 shows the overall performance comparison between our model and the above state-of-the-art AES models. From Table 2, we can see that our approach substantially improves the average QWK score by 2.3%, compared to the best baseline R^2BERT. It is worth noting that our model not only achieves the state-of-the-art average performance but also obtains better performance on almost all prompts compared to all baselines, which shows the superiority of the proposed pairwise contrastive regression model for AES.

Model	Avg QWK
R^2BERT-RegrOnly	0.768
NPCR-RegrOnly	0.770
R^2BERT-RankOnly	0.756
NPCR-RankOnly	0.796
NPCR	**0.817**

Table 3: Ablation studies on the use of the regression and ranking objectives in our model.

Model	Avg QWK
NPCR-Accu	0.800
NPCR-Group	0.802
NPCR	**0.817**

Table 4: Performance comparison on the choice of reference essays during training and inference.

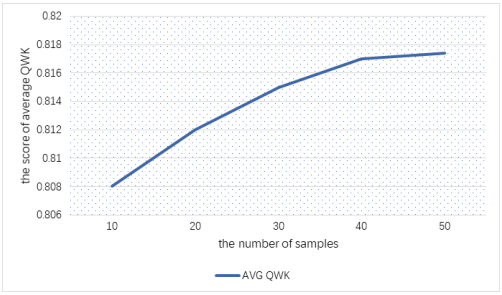

Figure 2: The performance curve varying with different number of reference essays.

4.3 Analysis

4.3.1 Effect of Contrastive Regression Learning

Unlike previous regression-based models for AES, our approach integrates regression with ranking within a contrastive regression framework. In this section, we evaluate the effect of exploiting contrastive regression by the ablation test.

As shown in the first row of Table 3, two baselines are presented for comparison. The first baseline R^2BERT-RegrOnly refers to the regression only version of R^2BERT (Yang et al., 2020). On the other hand, we also implement the second baseline NPCR-RegrOnly, which is the regression only version of our model NPCR by removing the contrastive learning from NPCR. More specifically, we first use BERT to obtain the representations of the input essays and then employ a fully connected layer with a sigmoid activation function to predict the scores. The results in Table 3 show that, compared the two baselines, our full model NPCR consistently improve QWK scores by 4.9% and by 4.7% respectively, which clearly indicates the importance of contrastive regression learning for our model.

4.3.2 Effect of Pairwise Ranking

In this section, we inspect the effect of our neural pairwise ranking model. In the second row of Table 3, the first baseline R^2BERT-RankOnly refers to the ranking only version of R^2BERT (Yang et al., 2020). Similarly, we also implement the second baseline NPCR-RankOnly, which is the ranking only version of our model NPCR by removing the score prediction part from NPCR. In detail, the output label of NPCR-RankOnly is a binary value which represents the priority relationship between any two essays. During inference, NPCR-RankOnly does not apply the multi-sample voting strategy, which means M is set to 1. After observing the results in Table 3, we can infer the following two implications: Firstly, the performance of NPCR-RankOnly is 4.0% better than R^2BERT-RankOnly, which shows that our neural pairwise ranking method is superior to the neural batchwise based ranking model in previous (Yang et al., 2020). Secondly, the large gap between our full model NPCR and the baseline NPCR-RankOnly clearly demonstrates the complementarity of the two methods.

4.3.3 Effect of the Strategy of Choosing Training Sample Pairs

In order to meet the accumulation of our model NPCR, we propose a sample selection strategy for building an effective training dataset, as illustrated in Section 3.3. In this section, we inspect the effect of this sample selection strategy. As a comparison, we also implement a baseline NPCR-Accu, which does not consider the accumulation while choosing the training sample pairs. That is to say, only the two adjacent essays in the given essay sequence are added into the training dataset. The results in Table 4 show that the average QWK score of NPCR is 1.7% better than the baseline NPCR-Accu.

4.3.4 Effect of the Strategy of Choosing Reference Essays

It is necessary to choose the number M in multi-sample voting strategy for inference. In this section, we investigate the relationship between the prediction performance and the number of reference essays. Figure 2 shows the results predicted with different number of reference essays. The per-

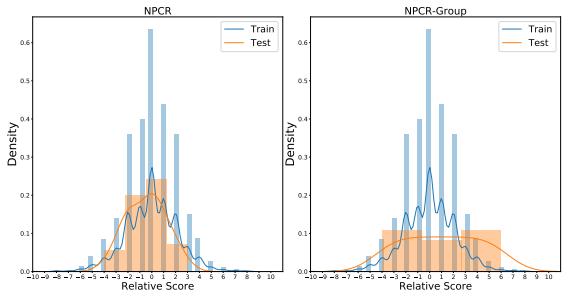

Figure 3: Realtive Score Distribution Histogram and Gaussian Kernel Density Estimation for model NPCR and NPCR-Group in Prompt 1.

Model	Avg QWK
NPCR-XLNet	0.816
NPCR-BERT	0.817
NPCR-RoBERTa	0.817

Table 5: Performance comparison of different pre-trained language models.

formance curve in Figure 2 demonstrates that the performance gradually improves as the number M increases, and then the performance growth tends to converge when M is greater than 40.

Furthermore, we consider the impact of different sample scores' distribution and devise a group-testing strategy to verify it. Concretely, we first divide the score range of training essays into M non-overlapping intervals (called $'groups'$), and then select M reference essays by picking only one essay from every group. From Table 4 we can see that NPCR has better performance than NPCR-Group that uses the group-testing strategy, indicating that randomly choosing samples is better than selecting samples from different score groups. For detailed reason, we generate Relative Socres Distribution Histograms and observe the Gaussian Kernel Density Estimates for the training and test essay pairs in all datasets, for instance Prompt 1 in Figure 3, in order to study the characteristics of the data distribution under different strategies of choosing reference essays. We can find that NPCR has better consistency w.r.t the distributions of relative scores in training and test data than NPCR-Group.

4.3.5 Comparison of Different Pre-trained Language Models

In this section we investigate the performance variation with different mainstream pre-trained language models including BERT, XLNet and RoBERTa. The experimental results in Table 5 show that the performance remains almost constant when changing the underlying pre-trained language model in our approach, which indicates that our AES model NPCR under the contrastive regression learning framework is relatively insensitive to the choice of pre-trained language models.

4.3.6 Computational Cost

In this section, we analyze the computational cost of the model NPCR. Compared with the previous work dealing with a single essay, our model NPCR really needs to take slightly more computational cost. However, in our model NPCR, the runtime of dealing with an essay pair is roughly similar to the cost of dealing with a single essay in the baselines, thus leading to the limitation of the increase of the computational cost. Hence, the number of essay pairs is a critical factor for analyzing the computational cost. In summary, during training, if the number of essays in training dataset is n, the number of essay pairs to be calculated in NPCR is less than $2*n$; during inference, if the number of essays in the test set is n, the number of essay pairs needed to be checked in NPCR should be $M*n$, where M is the number of reference essays. In effect, when we record the running time of model NPCR in Prompt 1 with GPU RTX3090Ti, the average training runtime is 90 seconds per epoch and the average inference runtime is 0.7 second per 40 essay pairs.

5 Related Work

Automated essay scoring systems have been deployed for high-stakes assessment since decades ago. The early approaches for AES mainly involved handcrafted feature based methods (Larkey, 1998; Chodorow and Burstein, 2004; Phandi et al., 2015; Zesch et al., 2015), while the recent studies have explored deep learning based methods to deliver state-of-the-art performance for this task.

In recent years, the mainstreams of AES methods typically formulate AES as a regression task. Multiple deep learning architectures based regression models for AES have been proposed. Taghipour and Ng (2016) presents the first neural network model for AES, which first uses the combination of CNN and LSTM to extract features of essays to generate text representation vectors and then apply a linear layer with sigmoid activation to map the vectors to valid scores. Dong and Zhang (2016) uses a hierarchical structure to automatically learn features from the word level and the sentence level.

Dong et al. (2017) further introduces the attention mechanism into the model and proves that CNN is more conducive to obtaining local features, while LSTM is more suitable for obtaining global features. Tay et al. (2018) proposes to consider neural coherence features as auxiliary features for prediction within an end-to-end neural framework. Recent advances in BERT (Devlin et al., 2019) model have inspired researchers to use pre-trained language model in AES (Rodriguez et al., 2019; Mim et al., 2019; Song et al., 2020).

Another line of research focuses on applying the learning to rank methods in AES tasks. Yannakoudakis et al. (2011) firstly formulate AES as a rank preference problem and then employ a pairwise ranking model RankSVM to rank two or more essays based on statistical features. Chen and He (2013) further utilizes the listwise ranking method to learn a ranking model based on linguistic features. Cummins et al. (2016) uses multi-task learning to address the problem of prompt adaptation by treating each prompt as a different task and introducing a constrained preference-ranking approach.

Recently, considering the complementarity of ranking and regression approaches, Yang et al. (2020) proposes a multi-loss method to combine regression and ranking in the AES task with a simple dynamic combination strategy.

6 Conclusion

In this paper, aiming to incorporate the merits of two popular AES solutions, we propose a novel unified model NPCR for AES which combines both regression and ranking objective in a principled way. Our approach is conceptually simple, however, the experimental results on the public dataset ASAP demonstrate that NPCR significantly outperforms previous approaches, advancing the state of the art in AES tasks.

In future work, we will explore more sophisticated neural feature extractors under the pairwise contrastive regression framework so that more powerful text features can be learned from the input essays, such as the hierarchical structure of a document, coherence features and so on.

References

Hongbo Chen and Ben He. 2013. Automated essay scoring by maximizing human-machine agreement. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1741–1752.

Ting Chen, Simon Kornblith, Mohammad Norouzi, and Geoffrey Hinton. 2020. A simple framework for contrastive learning of visual representations. In *Proceedings of the 37th International Conference on Machine Learning (ICML)*, pages 1597–1607.

Martin Chodorow and Jill Burstein. 2004. Beyond essay length: evaluating e-rater®'s performance on toefl® essays. *ETS Research Report Series*.

Mădălina Cozma, Andrei M Butnaru, and Radu Tudor Ionescu. 2018. Automated essay scoring with string kernels and word embeddings. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (ACL)*, pages 503–509.

Ronan Cummins, Meng Zhang, and Edward Briscoe. 2016. Constrained multi-task learning for automated essay scoring. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (ACL)*.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL-HLT)*, pages 4171–4186.

Fei Dong and Yue Zhang. 2016. Automatic features for essay scoring–an empirical study. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1072–1077.

Fei Dong, Yue Zhang, and Jie Yang. 2017. Attention-based recurrent convolutional neural network for automatic essay scoring. In *Proceedings of the 21st Conference on Computational Natural Language Learning (CoNLL)*, pages 153–162.

Marius Köppel, Alexander Segner, Martin Wagener, Lukas Pensel, Andreas Karwath, and Stefan Kramer. 2019. Pairwise learning to rank by neural networks revisited: Reconstruction, theoretical analysis and practical performance. In *Machine Learning and Knowledge Discovery in Databases- European Conference (ECML, PKDD)*, pages 237–252.

Leah S Larkey. 1998. Automatic essay grading using text categorization techniques. In *SIGIR '98: Proceedings of the 21st Annual International (ACM,SIGIR)*, pages 90–95.

Farjana Sultana Mim, Naoya Inoue, Paul Reisert, Hiroki Ouchi, and Kentaro Inui. 2019. Unsupervised learning of discourse-aware text representation for essay scoring. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics (ACL)*, pages 378–385.

Peter Phandi, Kian Ming A Chai, and Hwee Tou Ng. 2015. Flexible domain adaptation for automated essay scoring using correlated linear regression. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 431–439.

Pedro Uria Rodriguez, Amir Jafari, and Christopher M Ormerod. 2019. Language models and automated essay scoring. *CoRR*, abs/1909.09482.

Wei Song, Kai Zhang, Ruiji Fu, Lizhen Liu, Ting Liu, and Miaomiao Cheng. 2020. Multi-stage pre-training for automated chinese essay scoring. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 6723–6733.

Kaveh Taghipour and Hwee Tou Ng. 2016. A neural approach to automated essay scoring. In *Proceedings of the 2016 Conference on Empirical Methods in NaturalLanguage Processing (EMNLP)*, pages 1882–1891.

Yi Tay, Minh C Phan, Luu Anh Tuan, and Siu Cheung Hui. 2018. Skipflow: Incorporating neural coherence features for end-to-end automatic text scoring. In *Proceedings of the Thirty-Second AAAI Conference on Artificial Intelligence (AAAI)*, pages 5948–5955.

Ruosong Yang, Jiannong Cao, Zhiyuan Wen, Youzheng Wu, and Xiaodong He. 2020. Enhancing automated essay scoring performance via fine-tuning pre-trained language models with combination of regression and ranking. In *Findings of the Association for Computational Linguistics (EMNLP)*, pages 1560–1569.

Helen Yannakoudakis, Ted Briscoe, and Ben Medlock. 2011. A new dataset and method for automatically grading ESOL texts. In *the 49th Annual Meeting of the Association for Computational Linguistics (ACL)*, pages 180–189.

Xumin Yu, Yongming Rao, Wenliang Zhao, Jiwen Lu, and Jie Zhou. 2021. Group-aware contrastive regression for action quality assessment. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 7919–7928.

Torsten Zesch, Michael Wojatzki, and Dirk Scholten-Akoun. 2015. Task-independent features for automated essay grading. In *Proceedings of the Tenth Workshop on Innovative Use of NLP for Building Educational Applications (BEA@NAACL-HLT)*, pages 224–232.

Medical Question Understanding and Answering with Knowledge Grounding and Semantic Self-Supervision

Khalil Mrini[1], Harpreet Singh[1], Franck Dernoncourt[2], Seunghyun Yoon[2], Trung Bui[2], Walter Chang[2], Emilia Farcas[1], and Ndapa Nakashole[1]

[1]University of California, San Diego, La Jolla, CA 92093
{khalil, h1singh, efarcas, nnakashole}@ucsd.edu
[2]Adobe Research, San Jose, CA 95110
{franck.dernoncourt, syoon, bui, wachang}@adobe.com

Abstract

Current medical question answering systems have difficulty processing long, detailed and informally worded questions submitted by patients, called Consumer Health Questions (CHQs). To address this issue, we introduce a medical question understanding and answering system with knowledge grounding and semantic self-supervision. Our system is a pipeline that first summarizes a long, medical, user-written question, using a supervised summarization loss. Then, our system performs a two-step retrieval to return answers. The system first matches the summarized user question with an FAQ from a trusted medical knowledge base, and then retrieves a fixed number of relevant sentences from the corresponding answer document. In the absence of labels for question matching or answer relevance, we design 3 novel, self-supervised and semantically-guided losses. We evaluate our model against two strong retrieval-based question answering baselines. Evaluators ask their own questions and rate the answers retrieved by our baselines and own system according to their relevance. They find that our system retrieves more relevant answers, while achieving speeds 20 times faster. Our self-supervised losses also help the summarizer achieve higher scores in ROUGE, as well as in human evaluation metrics. We release our code to encourage further research.[1]

1 Introduction

Motivation. Users of medical question answering systems often write long questions, called Consumer Health Questions (CHQs). Several aspects of CHQs hinder the capacity of current question answering (QA) systems to process them: long medical questions may contain peripheral information like patient history (Roberts and Demner-Fushman, 2016) that are not necessary to retrieve relevant answers. Consumer health questions may also use a distinct vocabulary from the one used by medical providers to describe the same health concepts (Ben Abacha and Demner-Fushman, 2019a).

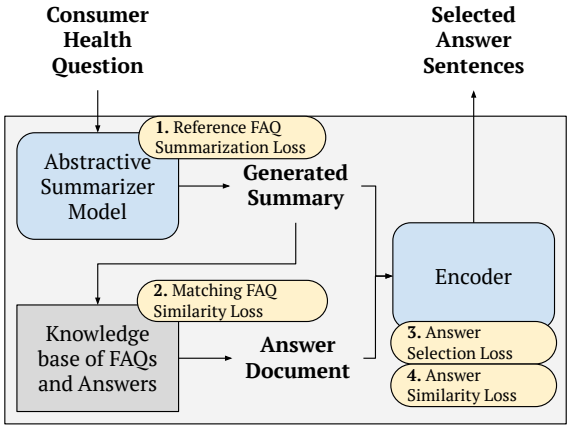

Figure 1: Overview of our proposed Consumer Health Question Understanding and Answering model. The input is a user question, called *Consumer Heath Question* (CHQ). The goal is to match the CHQ to relevant answer sentences associated with a *Frequently Asked Question* (FAQ) from a medical knowledge base.

A growing number of approaches attempt to enhance the processing of consumer health questions – or medical question understanding. These approaches include query relaxation (Ben Abacha and Zweigenbaum, 2015; Lei et al., 2020), question entailment (Ben Abacha and Demner-Fushman, 2016, 2019b; Agrawal et al., 2019), question summarization (Ben Abacha and Demner-Fushman, 2019a), and question similarity (Ben Abacha and Demner-Fushman, 2017; Yan and Li, 2018).

However, the above medical question understanding approaches stop short of retrieving answers after processing consumer health questions. The Medical Question Answering Task at TREC 2017 LiveQA (Ben Abacha et al., 2017) attempts to fill the gap by proposing the task of Consumer Health Question Answering. The goal is to retrieve relevant answers obtained using online search for

[1]Link: https://github.com/KhalilMrini/Medical-Question-Answering

Proceedings of the 29th International Conference on Computational Linguistics, pages 2734–2747
October 12–17, 2022.

the corresponding CHQ. As part of their participation in this task, Yang et al. (2017) find that online search engine queries introduce noise in performance, and that even collected and curated medical knowledge available offline can fare better.

Contributions. To enable the use of a curated medical knowledge base for answering long user questions, we introduce a novel, knowledge-grounded and semantically self-supervised system for Consumer Health Question Understanding and Answering (CHQUA). We tackle a challenging aspect of CHQUA: providing answers when no relevance labels are available. Our contributions are as follows:

(1) We propose an end-to-end pipeline, as shown in Figure 1, that takes as input a consumer health question, and trains a summarizer model to generate a short, formally worded question. We optimize a summarization training objective using the medical question summarization datasets.

(2) The medical knowledge base we use is separate from the question summarization datasets, and therefore we have no labels to indicate which knowledge base question matches a given consumer health question. We design a novel, semantically-guided self-supervised loss function to ground the generated summary with knowledge base FAQs, using semantic similarity as proxy to question matching. The Matching FAQ similarity loss helps the encoder pick the most semantically similar knowledge base question.

(3) The large medical knowledge base we use has no answer sentence relevance labels. We adapt to this scenario by designing two complementary self-supervised losses on the same encoder, and by considering semantic similarity as a proxy to relevance. The Answer Similarity loss pushes the model to distinguish between relevant and irrelevant answer sentences, whereas the Answer Selection loss works in a complementary way to push the model to select a given number of sentences.

Finally, we conduct an evaluation to compare the relevance of our system with two strong baselines of retrieval-based question answering. We ask evaluators to ask their own questions, and then perform a blind evaluation of the retrieved answers by each system. Seven evaluators find that our system retrieves more relevant answers compared to the two baselines, while achieving significantly faster processing speeds. We also find that the self-supervised losses help achieve better scores in ROUGE and human evaluation metrics. However, we find that the task remains challenging, with room for improvement. We release our code, model, and matched datasets to encourage further research in consumer health question understanding and answering.

2 Related Work

Consumer Health Question Answering. Ben Abacha et al. (2017) introduce the Medical QA shared task at TREC 2017 LiveQA, where the goal is to develop a consumer health question answering system. The training data is comprised of question-answer pairs. The questions are informally worded CHQs received by the U.S. National Library of Medicine (NLM). The answers are formally worded and come from websites of the U.S. National Institutes of Health or manually collected by librarians. The evaluation scores are given by humans, using a test set of CHQs and reference answers.

Many participating teams adopt a question matching approach, and train their models on question similarity datasets like the Quora question pair dataset (Iyer et al., 2017), or other datasets collected from community question answering websites. TODO (Mrini et al., 2021b)

In the MEDIQA 2019 Shared task, Ben Abacha and Demner-Fushman (2019a) introduce a differently defined consumer health question answering task. Here, the goal is to rank a given list of answers according to their relevance with regard to a CHQ. He et al. (2020) introduce a new disease knowledge infusion training procedure for BERT (Devlin et al., 2019) that scores well in this task.

Medical Question Answering. Medical QA approaches include translating questions to SPARQL queries (Ben Abacha and Zweigenbaum, 2012), semantic similarity between questions and candidate answers (Hao et al., 2019), knowledge representations (Terol et al., 2007; Goodwin and Harabagiu, 2017), ranking candidate answers (Ben Abacha et al., 2017, 2019), summarization of questions and/or answers (Ben Abacha et al., 2021; Mrini et al., 2021d,b,c), and medical entity linking (Basaldella et al., 2020; Mrini et al., 2022).

There is a variety of definitions for the task of medical QA and related sub-tasks in the literature. Hao et al. (2019) define medical QA as the task of finding the correct answer from a set of candidates and a body of evidence documents. They propose to work on two datasets: the National Medical Li-

censing Examination of China (NMLEC) (Shen et al., 2020), and Clinical Diagnosis based on Electronic Medical Records (CD-EMR), where the goal is to predict the correct diagnosis based on patient history.

Sharma et al. (2018) propose to tackle three kinds of medical questions found in the BioASQ challenge (Balikas et al., 2015): factoid questions where answers are single entities, list-type questions where answers are a set of entities, and yes/no questions.

Retrieval-based Question Answering. Recent methods for retrieval-based QA systems use contextual text embeddings to evaluate a candidate answer's relevance to a given question.

Tay et al. (2018) propose to use Multi-Cast Attention Networks (MCAN), a new attention mechanism, to model question-answer pairs.

Mrini et al. (2021e) introduce a recursive, tree-structured model that models sentences according to their syntactic tree. Their results show that tree structure sets a new state of the art in conventional, formally worded QA benchmarks like TrecQA and WikiQA (Yang et al., 2015), but does not fare well in informally worded, user-written datasets.

Karpukhin et al. (2020) introduce Dense Passage Retrieval (DPR): a dual-encoder based on BERT (Devlin et al., 2019), that predicts relevance scores of passages with regard to a question. DPR encoders are trained on the relevance of passages from datasets containing such labels, using a supervised negative log-likelihood loss based on the semantic similarity of questions and relevant passages.

Mao et al. (2021) modify the *query* part of retrieval-based QA: they propose to use language models to generate context for queries. They then feed the extended queries to retrieval systems, such as DPR or BM-25.

3 Problem Definition

We define knowledge-grounded Consumer Health Question Understanding and Answering (CHQUA) as the problem of retrieving a fixed number of answer sentences from a medical knowledge base that are the most relevant given a long and informal user question – called a Consumer Health Question (CHQ). There are three steps in CHQUA: question summarization, matching the summarized user question with a relevant FAQ from the knowledge base, and retrieval of the relevant answer sentences from the corresponding answer document.

Knowledge-grounded CHQUA is comprised of three elements used for training. First, the CHQ is the input of the task. Second, the Reference FAQ (Frequently Asked Question) is the golden or expert-written summary corresponding to the CHQ. Whereas the CHQ is a long and informally worded question, the reference FAQ is the corresponding short, one-sentence, formally worded question. At inference time, the reference FAQ is not available, and we will therefore use a summary generated by the model. Third, the medical knowledge base is comprised of FAQs, where each FAQ has a corresponding answer document with at least one sentence. FAQs in the knowledge base are also short, one-sentence, formally worded questions.

The goal of knowledge-grounded CHQUA is to find a set $\mathcal{R}$ of n relevant answer sentences, from a document comprised of answer sentences $\mathcal{A}_i$, such that $\mathcal{A}_i$ corresponds to question q_i from the knowledge base. We call q_i the retrieved or matching FAQ, such that q_i is the most similar question to the user's summarized question $\mathrm{q_u}$:

$$\mathrm{q}_i = \arg\max_{\mathrm{q} \in \mathcal{Q}} f(\mathrm{q}, \mathrm{q_u}) \tag{1}$$

where $\mathcal{Q}$ is the set of questions (FAQs) in the knowledge base, and f is a given similarity scoring function. $\mathrm{q_u}$ is the reference FAQ (during training) or a generated summary (during inference).

We find the set $\mathcal{R}$ of n relevant answer sentences such that it maximizes the relevance score with the user's summarized question $\mathrm{q_u}$:

$$\mathcal{R} = \arg\max_{\mathcal{R}' \subset \mathcal{A}_i} \sum_{\mathrm{a} \in \mathcal{R}'} g(\mathrm{a}, \mathrm{q_u}) \tag{2}$$

where a is an answer sentence, and g is a given relevance scoring function.

4 Our Pipeline

Our proposed pipeline for Consumer Health Question Understanding and Answering has three main components.

In the first step, our approach learns to *understand* the intent of user questions (CHQs) by summarizing them. We use an encoder-decoder-based summarization model for this step.

The second step is question matching, or the retrieval of the relevant FAQ from the knowledge base: we *ground* the generated summary to a medical knowledge base of FAQs and corresponding

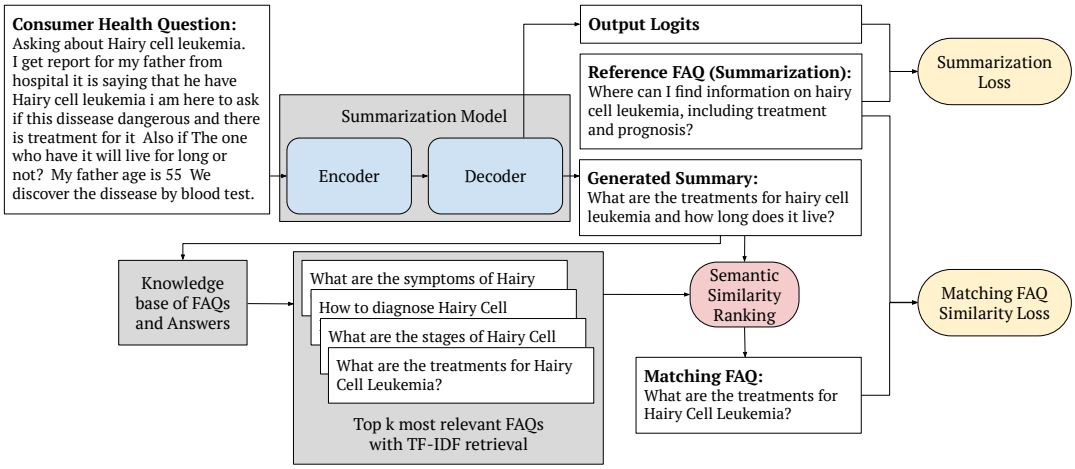

Figure 2: The Consumer Health Question (user question) is first summarized, and we then retrieve a relevant question from the knowledge base using the generated summary. The **top** half of the figure illustrates the first step: question understanding through summarization (§4.1). The **bottom** half of the figure illustrates the second step: question matching through self-supervised knowledge grounding (§4.2).

answer documents. As there are no question matching labels, we consider semantic similarity as a proxy to question matching, and we optimize a self-supervised similarity loss.

The third step is the retrieval of the relevant answer sentences: our model learns to *select* the top-k most relevant answer sentences from the matching answer document. To achieve this task in the absence of answer relevance labels, we consider semantic similarity as a proxy for relevance, and we optimize two novel, semantically-guided, and self-supervised loss functions. The first pushes the model to discriminate between relevant and irrelevant sentences, and the other pushes the model to consider only a fixed number of sentences as relevant.

We show an overview of the model and learning objectives in Figure 1. The entire pipeline is trained together, as the summarizer encoder is re-used to encode the questions and answer sentences.

4.1 Question Understanding through Summarization

Our work aims to flip the burden of question understanding on the question answering model. Instead of asking the user to shorten or reformulate their question, we train an encoder-decoder abstractive summarizer to shorten user questions. Figure 2 illustrates this part of the model.

At training time, we input a Consumer Health Question (CHQ) to the summarization model. The reference Frequently Asked Question (FAQ) is the corresponding shorter and formal question. Given a CHQ embedding $\mathbf{x}$ and the corresponding reference FAQ embedding $\mathbf{y_{ref}}$, the summarization loss is defined as the following negative log-likelihood objective:

$$\mathcal{L}_{\text{sum}} = -\log p(\mathbf{y}_{\text{ref}}|\mathbf{x};\theta) \quad (3)$$

4.2 Question Matching through Self-Supervised Knowledge Grounding

In the next step, we match the summarized user question with the most relevant FAQ from the medical knowledge base. We use semantic similarity as a proxy for question matching, in the absence of such labels.

The knowledge-grounding process is comprised of two steps. First, we use TF-IDF-weighted bag-of-word and n-gram vectors to get the top k most relevant FAQs from the knowledge base. This first step acts as a fast filter to extract a small subset of candidate FAQs. Our retrieval approach follows the retrieval methods commonly used in question answering systems (Chen et al., 2017; Dinan et al., 2018). Dinan et al. (2018) note that the retriever is a potentially learnable part of the model. In our case, using TF-IDF retrieval is computationally optimal and scalable given a large knowledge base with thousands of FAQs. We use a TF-IDF embedder fitted on all the FAQs of the knowledge base, as well as reference FAQs from the training set of the question summarization dataset.

The second step of knowledge-grounding is to

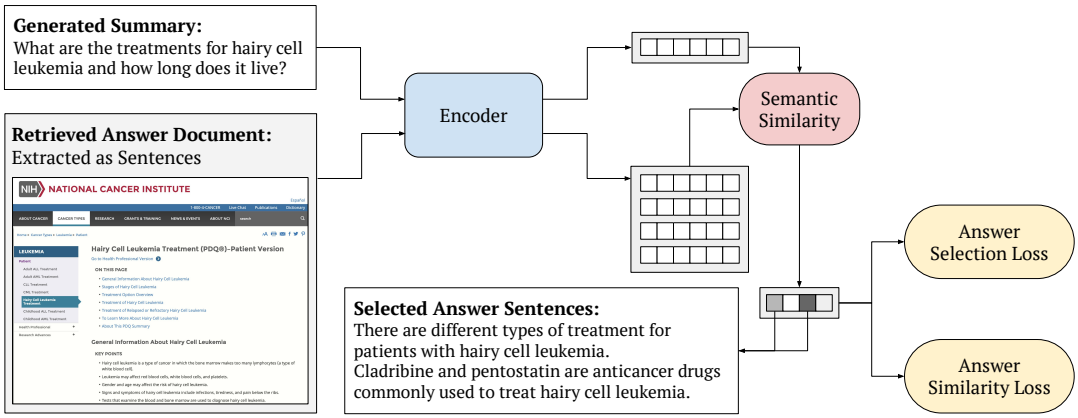

Figure 3: Illustration of the third step of our pipeline: answer retrieval through self-supervised similarity and selection losses (§4.3). Following the same example as Figure 2, our model encodes sentences from the retrieved answer document from the knowledge base, and compares them to the FAQ generated by the summarization model. We use the encoder of the summarization model to embed sentences.

rank the top k FAQs using semantic similarity. To get semantic embeddings of the generated summary and the corresponding top k most relevant FAQs from the knowledge base, we use the encoder of the summarization model. We take inspiration from the precision formula of BERTSCORE (Zhang et al., 2019), and compute the weighted semantic similarity score as follows:

$$\text{Sim}(\text{q}_\text{u}, \text{q}_i) = \sum_{w \in \mathcal{W}_\text{u}} \max_{w' \in \mathcal{W}_i} \frac{\text{idf}(w) \cdot \text{CosSim}(\mathbf{x}_w, \mathbf{x}_{w'})}{\sum_{w'' \in \mathcal{W}_\text{u}} \text{idf}(w'')} \quad (4)$$

where q_u is the reference FAQ (during training) or the generated summary (during inference), q_i is the i-th question from the top k most relevant FAQs, $\mathcal{W}_\text{u}$ and $\mathcal{W}_i$ are the corresponding sets of words, CosSim is the cosine similarity function, and $\text{idf}(w)$ is the inverse document frequency of the word w.

The matching FAQ is the knowledge base FAQ with the highest similarity score with q_u, as shown in the example in Figure 2. During training, the summarization model may produce low-quality or degenerate FAQs. For this reason, at training time, we choose to use the reference FAQ instead to compute the semantic similarity scores and find the matching FAQ. At test time, we only use the generated summary.

Since we are using different datasets for the question summarization and for the knowledge base, we have to reconcile the questions from the knowledge base and the reference questions. We propose to force the model to learn a representation space that does not distinguish between the reference FAQ and the most similar knowledge base FAQ. To accomplish this, we compute the matching FAQ similarity loss. Given the embedding of a summarization reference FAQ q_sum and the embedding of a matching FAQ q_mat, the matching FAQ similarity loss is defined as:

$$\mathcal{L}_\text{mat} = 1 - \text{ReLU}\left(\text{Sim}\left(\text{q}_\text{sum}, \text{q}_\text{mat}; \theta\right)\right) \quad (5)$$

4.3 Answer Retrieval through Self-Supervised Similarity and Selection Losses

After summarizing the user question and retrieving a relevant FAQ from the knowledge base, the next step is to retrieve relevant sentences from the corresponding answer document. In our setting, we need to retrieve a fixed number of sentences relevant to the user question. However, we have no labels for the answer sentences indicating relevance to the user question. We propose two complementary self-supervised learning objectives, that use semantic similarity as a proxy to relevance scoring, and satisfy the constraint of selecting a fixed number of answer sentences.

We show an overview of our answer retrieval approach in Figure 3. In the example of the figure, we show for simplicity a relatively short answer document with four sentences, from which the model chooses the two most relevant ones. In practice, there are close to ten sentences in answer documents.

We compute semantic similarity scores between the generated summary (for inference) or the reference FAQ (for training), and each of the sentences

of the retrieved answer document. We obtain the semantic embeddings of each sentence using the encoder of the summarization model. We then compute semantic similarity scores as shown in equation 4. Cosine similarity scores have values in the $[-1; 1]$ range. For a pair of sentences, a cosine similarity value closer to -1 means that the corresponding sentence embeddings are negatively correlated, or that the sentences have opposite meanings. A value closer to 0 means that the embeddings are not correlated, and that there is no particular semantic relation between the sentences. A value closer to 1 means that the sentence embeddings are positively correlated, and the sentences are close semantically. We consider that a sentence is relevant when the values are closer to 1, and irrelevant otherwise. For this reason, we apply a ReLU activation on the cosine similarity scores before feeding them to the loss functions.

We propose two learning objectives to achieve the self-supervised selection of relevant answer sentences. The semantic similarity loss pushes the model to increase its confidence in the relevance of answer sentences, whereas the answer selection loss pushes the model to select only a fixed number of sentences. The intuition for sharing the encoder with the summarization model, is that these two losses will enable the summarizer to absorb notions of relevance and semantic similarity.

Given the summarization reference FAQ $\mathrm{q_{sum}}$ and the i-th sentence of the retrieved answer document a_i, we compute the ReLU-activated semantic similarity score as follows:

$$S(\mathrm{q_{sum}}, \mathrm{a}_i; \theta) = \mathrm{ReLU}\left(\mathrm{Sim}\left(\mathrm{q_{sum}}, \mathrm{a}_i; \theta\right)\right) \quad (6)$$

We then define the semantic similarity loss $\mathcal{L}_{\mathrm{sim}}$ and the answer selection loss $\mathcal{L}_{\mathrm{sel}}$ as follows:

$$\mathcal{L}_{\mathrm{sim}} = \sum_{i=1}^{|\mathcal{A}|} S(\mathrm{q_{sum}}, \mathrm{a}_i; \theta) * (1 - S(\mathrm{q_{sum}}, \mathrm{a}_i; \theta)) \quad (7)$$

$$\mathcal{L}_{\mathrm{sel}} = \left| \min(n, |\mathcal{A}|) - \sum_{i=1}^{|\mathcal{A}|} S(\mathrm{q_{sum}}, \mathrm{a}_i; \theta) \right| \quad (8)$$

where $\mathcal{A}$ is the set of sentences in the retrieved answer document, and n is the fixed number of sentences to be retrieved.

DATASET SPLIT	TRAIN	DEV	TEST
MeQSum	405	50	50
HealthCareMagic	1,314	164	165

Table 1: Statistics of the medical dataset splits.

The semantic similarity loss $\mathcal{L}_{\mathrm{sim}}$ pushes the semantic similarity values to be either 1 (relevant) or 0 (irrelevant). In combination with $\mathcal{L}_{\mathrm{sim}}$, the answer selection loss pushes the model to only select up to n sentences to have semantic similarity values close to 1. Our system then outputs the sentences with the highest semantic similarity values in the order in which they appear in the answer document. Therefore, the particular semantic similarity ranking of the relevant sentences does not matter – it only matters that relevant sentences have the n highest values.

Finally, the learning objective $\mathcal{L}$ is as follows:

$$\mathcal{L} = \mathcal{L}_{\mathrm{sum}} + \lambda * \mathcal{L}_{\mathrm{mat}} + \gamma * (\mathcal{L}_{\mathrm{sim}} + \mathcal{L}_{\mathrm{sel}}) \quad (9)$$

where λ and γ are hyperparameters. We use only one weight for $\mathcal{L}_{\mathrm{sim}}$ and $\mathcal{L}_{\mathrm{sel}}$ as these two losses are complementary.

5 Experiments and Results

In this section, we evaluate our proposed pipeline for Consumer Health Question Understanding and Answering, and we propose to compare our proposed pipeline against two strong baselines. Seven medical experts judge the performance of our system and baselines by asking their own questions, and rating the relevance of the answers retrieved. Then, we analyze the results through the lens of summarization metrics, human evaluation, and computational speed.

5.1 Datasets

We use one medical knowledge base, MedQuAD (Ben Abacha and Demner-Fushman, 2019b), and two medical question summarization datasets: MeQSum (Ben Abacha and Demner-Fushman, 2019a) and HealthCareMagic (Zeng et al., 2020). All datasets are in English. We show dataset statistics in Table 1.

5.1.1 Dataset Details

MedQuAD is a large-scale Medical Question Answering Dataset. Ben Abacha and Demner-Fushman (2019b) collect trusted medical question-answer pairs by crawling them from 12 websites

of the U.S. National Institutes of Health (NIH). Each web page contains information about a health-related topic, like a disease or a drug. The authors automatically collect the question-answer pairs by composing handcrafted patterns adapted to each website based on document structure and section titles. They manually evaluate 1,721 CHQs to come up with automatic wording patterns for each of 36 question types. Therefore, even though answers are curated and written by medical experts, questions are automatically formulated and may have some noise.

We collect the publicly available (e.g. not copyrighted) question-answer pairs from the MedQuAD dataset[2]. We then use the NLTK sentence tokenizer (Bird, 2006) to split answer documents into sentences. We get 16,423 questions and 157,592 answer sentences, making for an average of 9.6 answer sentences for each question.

MeQSum (Ben Abacha and Demner-Fushman, 2019a) is a medical question summarization dataset released by the U.S. National Institutes of Health (NIH). It contains 1,000 consumer health questions summarized into FAQ-style single-sentence questions by medical experts.

HealthCareMagic is a medical dialogue dataset issued as part of the MedDialog dataset (Zeng et al., 2020)[3]. It is crawled from `HealthCareMagic.com`, an online healthcare service platform. This dataset includes first a formally worded, one-sentence question describing the intent of the patient question, followed by 2 long utterances: a CHQ from the patient that includes a description of the problem and a question, and then an answer from the doctor. To form a medical question summarization dataset, we consider the single-sentence descriptions as summaries of the patient's CHQ. We collect 226,405 question pairs.

5.1.2 Knowledge-based Filtering of Datasets

We conduct experiments for each of the two question summarization datasets, and we use MedQuAD as the underlying knowledge base in all experiments. For this reason, we decide to filter each of the question summarization datasets to reconcile their differences with MedQuAD.

We first fit a TF-IDF embedding model, similar to the one of (Dinan et al., 2018), on the reference FAQs of each question summarization dataset and the questions of MedQuAD. We then compute the dot products of the TF-IDF-weighted vectors for all possible pairs of summarization FAQs and MedQuAD questions. We assign a matching score $m(\mathrm{q_{sum}})$ to each summarization reference FAQ:

$$m(\mathrm{q_{sum}}) = \max_{\mathrm{q}' \in \mathcal{Q}_{\mathrm{MedQuAD}}} \mathrm{tfidf}(\mathrm{q_{sum}}) \cdot \mathrm{tfidf}(\mathrm{q}') \tag{10}$$

We manually evaluate the matching scores for each summarization dataset to set a cutoff matching score of filtering. This way, we obtain question summarization datasets where reference FAQs have matches in the medical knowledge base. Finally, we perform a random and rough 80/10/10 split for the train/dev/test sets. The dataset statistics are in the main paper.

5.2 Training Settings

We adopt the BART encoder-decoder model (Lewis et al., 2020), as it set a state of the art in abstractive summarization benchmarks. We train our model using the HuggingFace implementation (Wolf et al., 2020), on a learning rate of $2 \cdot 10^{-6}$. The question matching pool retrieved by TF-IDF is comprised of $k = 32$ knowledge base FAQs. Our answer selection loss $\mathcal{L}_{\mathrm{sel}}$ is optimized to select up to $n = 3$ sentences. We use $\lambda = 0.01$ and $\gamma = 0.01$ as weights for the self-supervised losses. The BART encoder is used for embedding sentences for question matching and answer selection.

We train for 50 epochs for MeQSum, and 20 epochs for HealthCareMagic. Each training epoch takes about 10 minutes for MeQSum, and about 35 minutes for HealthCareMagic. Inference takes 1 minute for the MeQSum test set and 3 minutes for the HealthCareMagic test set. The best checkpoint is selected based on the lowest loss value $\mathcal{L}$ on the dev set.

We use BART Large pre-trained on the CNN-Dailymail dataset, and each BART Large model contains 406 million parameters, as per the HuggingFace implementation.

5.3 Baselines

We propose the two following baselines in retrieval-based question answering: Dense Passage Retrieval (DPR) (Karpukhin et al., 2020), and Generation-Augmented Retrieval (GAR) (Mao et al., 2021). We adapt these two baselines to our case, and adopt BART-based pre-trained encoders.

[2] https://github.com/abachaa/MedQuAD

[3] https://github.com/UCSD-AI4H/Medical-Dialogue-System

SYSTEM	MeQSum	HealthCareMagic	Time/Query
DPR (Karpukhin et al., 2020)	1.42	1.73	47 seconds
GAR (Mao et al., 2021)	1.40	1.64	48 seconds
Ours	2.13	2.35	2 seconds

Table 2: Evaluation of the relevance (out of 5) of answers retrieved by our proposed system and two strong baselines for questions asked by seven evaluators. The systems trained on MeQSum are evaluated on 60 questions by 3 evaluators, and the ones trained on the larger HealthCareMagic dataset are evaluated on 80 questions by 4 evaluators. The column on the right shows the number of seconds it takes for a loaded system to retrieve the answer to a query.

Similarly to our own pipeline, we create a two-stage retrieval to get answers. The first stage encodes questions from the knowledge base, and retrieves the question that is most relevant to the query. The second stage encodes the corresponding answer document, and retrieves the three sentences that are most relevant to the query.

For DPR, the query is simply the user question. For GAR, we need to generate a context to add to the user question: we choose to add the summary of the user question as the context. We train a BART encoder to summarize user question, using the question summarization datasets.

Whereas our system's retrieval encoder is trained on our proposed self-supervised objectives, the retrieval encoders of the baselines are trained on Wikipedia for the task of retrieval-based question answering.

5.4 Do we retrieve relevant answers?

5.4.1 Evaluation Strategy

We hire seven annotators: four of which are medical doctors, and the remaining three hold degrees related to healthcare or immunology.

We ask the evaluators to first write user questions, and then evaluate the answers retrieved by our system and the two existing systems. Given that our medical knowledge base has limited questions, we ask the evaluators to limit their questions to the topics covered by the nine sources from which the knowledge base was extracted.

Then, we ask the evaluators to rate the relevance of the answers retrieved by each system independently, on a scale of 1 (not relevant) to 5 (relevant). The full description of scores given to the annotators is in the Appendix.

Each of the seven annotators wrote 20 questions, and each question gets three answers (one per system). We assign three annotators to the models trained on MeQSum, and four to the models trained on HealthCareMagic. The annotators rate answers only for the questions that they wrote themselves.

5.4.2 Results and Discussion

We show the results of the evaluations in Table 2. The first three columns show the averages of relevance scores that were given by annotators for all systems.

The results show that the evaluators have preferred our system's answers over the answers retrieved by the two baselines. Our system gets relevance scores that are 0.6 to 0.7 points higher, out of 5 on the relevance scale. An annotator commented that they find our system to be "*more organized and to-the-point than the rest of systems*."[4]

The two baselines seem to perform similarly to each other. This is likely due to the fact that the main difference between them is that the query is generation-augmented for GAR, whereas the query is simply the user question for DPR.

Overall, the relevance scores are on the lower side, as no system exceeds an average score of 2.5/5. This shows that consumer health question answering and understanding is a challenging task, especially since there are no labels to indicate whether an answer is relevant to a particular question, or which FAQ matches the user's intent.

In addition, the challenges of the task are also due to the limitations of the knowledge base. Some annotators noted that the retrieved answers were often not appropriate, or close to the topic but not answering the question. This is due to the fact that MedQuAD does not cover all possible illnesses and medical conditions that the users could ask about. Whereas a larger database would potentially solve coverage problems, it could be at the expense of the quality or verifiability of the answers. The MedQuAD dataset is at times noisy, and contains generic sentences that may not answer any question, or generic templates related to percentages of symptoms and how frequent they are.

[4] Annotators were not told that either system was ours or not. The systems were simply numbered for a blind evaluation.

CRITERIA	Fluency			Coherence			Informativeness			Correctness		
EVALUATION	Win	Lose	Tie	Win	Lose	Tie	Win	Lose	Tie	Win	Lose	Tie
MeQSum	11	5	28	10	6	28	12	3	29	12	4	28
HealthCareMagic	45	17	42	44	19	41	46	18	40	44	18	42

Table 3: Question Understanding evaluation: blind evaluation by 2 annotators of the generated summaries for the test set CHQs. A "Win" evaluation means that our model generates a better summary than the baseline summarizer.

DATASET	MeQSum			HealthCareMagic		
METRIC	R1	R2	RL	R1	R2	RL
GAR (Mao et al., 2021)	45.72	30.43	42.02	31.04	13.68	27.90
Ours	46.74	30.10	42.81	33.13	14.71	30.18

Table 4: Question Understanding evaluation: summarization results on test set (reference FAQs). The R1, R2 and RL metrics refer to the F1 scores of ROUGE-1, ROUGE-2 and ROUGE-L.

5.5 Computational Speed

We run our system on a single 11GB GPU, whereas the two baselines are each run on four 16GB GPUs. We show the average duration required to retrieve answers for a single query in the right column of Table 2.

We notice that, in addition to the higher relevance scores, the advantage of our system is that it is significantly (more than 20 times) faster compared to the two baselines. This is largely due to the fact that we limit to 32 the number of knowledge base questions that we encode and compare the query embedding to. In contrast, DPR and GAR encode all questions in the knowledge base. This is done at the beginning when loading the models, but the query similarity computation is done at each run, thereby lengthening the processing time.

5.6 Analysis of Question Understanding

An additional way that our system outperforms the two baselines could be through summarization. We evaluate the summarization of consumer health questions using the ROUGE metric (Lin, 2004). Our GAR baseline uses a BART model trained on the summarization loss only. We show the results in Table 4. We notice that sharing encoder parameters between the summarization loss and our proposed self-supervised losses generally increases ROUGE F1 scores across both datasets. For HealthCareMagic, score increases exceed 2 points in ROUGE-1 and ROUGE-L.

Given that ROUGE is notoriously unreliable, we hire two additional annotators on Upwork who are healthcare workers to judge the fluency, coherence, informativeness and correctness of generated summaries. We show the annotators the consumer health question (source text), the reference FAQ (target text) and two generated summaries. The annotators do not know which system generated which summary. We show the evaluation scores in Table 3. We remove repetitions of reference FAQs in the test sets put up for evaluation. The results confirm that our self-supervised losses increase the quality of generated summaries. Summaries generated with our model score more wins more often than losses on all four metrics, and score more wins than ties with the summarization-only baseline for HealthCareMagic.

6 Conclusions

We introduce an end-to-end pipeline for knowledge-grounded consumer health question answering and understanding (CHQUA). Our challenge is that we have no labels for question matching or answer relevance. We propose to use semantic similarity as a proxy for those labels, and we design three novel self-supervised losses: one works to match the user's summarized question to a knowledge base question, and the other two losses work complementarily to teach our model to select a fixed number of relevant answer sentences. We compare our proposed system against two strong baselines of retrieval-based question answering. We hire seven medical experts to ask their questions, and they find that our system provides more relevant answers. Our system also achieves processing times that are more than 20 times faster. Finally, we find that our proposed self-supervised losses enable the summarizer model to achieve higher scores in ROUGE and human evaluation metrics, compared to a summarization-only baseline. However, we find that this task remains challenging and that there is still room for improvement. We release our code and model to encourage further research.

Ethical Considerations

Our model is for medical question answering, but should be used with caution as it does not claim

to provide medical advice. Potential users of our system should be warned to not blindly trust the answers given to their medical questions. Potential users should always consult their physician for medical advice.

Each of our annotators spent between two and four hours on the task we gave them. Each annotator was compensated fairly for their work. We answered all of the annotators' questions about the task before they started. Hiring platform Upwork guarantees the payment, fair treatment and informed consent of our nine hired annotators through a mutually agreed-upon contract. The platform fee for Upwork was paid by us, and not deducted from the compensation of the annotators.

Acknowledgements

Khalil Mrini is supported by Adobe Research Unrestricted Gifts, and by an Amazon Research Award under his AWS AI proposal "*Learning Representations for Voice-Based Conversational Agents for Older Adults*". This work is part of the VOLI project (Mrini et al., 2021a; Johnson et al., 2020), and we gratefully acknowledge the award from NIH/NIA grant R56AG067393.

References

Anumeha Agrawal, Rosa Anil George, Selvan Suntiha Ravi, Sowmya Kamath, and Anand Kumar. 2019. Ars_nitk at mediqa 2019: analysing various methods for natural language inference, recognising question entailment and medical question answering system. In *Proceedings of the 18th BioNLP Workshop and Shared Task*, pages 533–540.

Georgios Balikas, Anastasia Krithara, Ioannis Partalas, and George Paliouras. 2015. Bioasq: A challenge on large-scale biomedical semantic indexing and question answering. In *Multimodal Retrieval in the Medical Domain*, pages 26–39, Cham. Springer International Publishing.

Marco Basaldella, Fangyu Liu, Ehsan Shareghi, and Nigel Collier. 2020. Cometa: A corpus for medical entity linking in the social media. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 3122–3137.

Asma Ben Abacha, Eugene Agichtein, Yuval Pinter, and Dina Demner-Fushman. 2017. Overview of the medical question answering task at trec 2017 liveqa. In *TREC*.

Asma Ben Abacha and Dina Demner-Fushman. 2016. Recognizing question entailment for medical question answering. In *AMIA Annual Symposium Proceedings*, volume 2016, page 310. American Medical Informatics Association.

Asma Ben Abacha and Dina Demner-Fushman. 2017. Nlm_nih at semeval-2017 task 3: from question entailment to question similarity for community question answering. In *Proceedings of the 11th International Workshop on Semantic Evaluation (SemEval-2017)*, pages 349–352.

Asma Ben Abacha and Dina Demner-Fushman. 2019a. On the summarization of consumer health questions. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 2228–2234.

Asma Ben Abacha and Dina Demner-Fushman. 2019b. A question-entailment approach to question answering. *BMC bioinformatics*, 20(1):511.

Asma Ben Abacha, Yassine Mrabet, Yuhao Zhang, Chaitanya Shivade, Curtis Langlotz, and Dina Demner-Fushman. 2021. Overview of the mediqa 2021 shared task on summarization in the medical domain. In *Proceedings of the 20th SIGBioMed Workshop on Biomedical Language Processing, NAACL-BioNLP 2021*. Association for Computational Linguistics.

Asma Ben Abacha, Chaitanya Shivade, and Dina Demner-Fushman. 2019. Overview of the mediqa 2019 shared task on textual inference, question entailment and question answering. In *Proceedings of the 18th BioNLP Workshop and Shared Task*, pages 370–379.

Asma Ben Abacha and Pierre Zweigenbaum. 2012. Medical question answering: Translating medical questions into sparql queries. In *Proceedings of the 2nd ACM SIGHIT International Health Informatics Symposium*, IHI '12, page 41–50, New York, NY, USA. Association for Computing Machinery.

Asma Ben Abacha and Pierre Zweigenbaum. 2015. Means: A medical question-answering system combining nlp techniques and semantic web technologies. *Information processing & management*, 51(5):570–594.

Steven Bird. 2006. Nltk: the natural language toolkit. In *Proceedings of the COLING/ACL 2006 Interactive Presentation Sessions*, pages 69–72.

Danqi Chen, Adam Fisch, Jason Weston, and Antoine Bordes. 2017. Reading wikipedia to answer open-domain questions. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1870–1879.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. Bert: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186.

Emily Dinan, Stephen Roller, Kurt Shuster, Angela Fan, Michael Auli, and Jason Weston. 2018. Wizard of wikipedia: Knowledge-powered conversational agents. In *International Conference on Learning Representations*.

Travis R. Goodwin and Sanda M. Harabagiu. 2017. Knowledge representations and inference techniques for medical question answering. *ACM Trans. Intell. Syst. Technol.*, 9(2).

Yu Hao, Xien Liu, Ji Wu, and Ping Lv. 2019. Exploiting sentence embedding for medical question answering. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, pages 938–945.

Yun He, Ziwei Zhu, Yin Zhang, Qin Chen, and James Caverlee. 2020. Infusing disease knowledge into bert for health question answering, medical inference and disease name recognition. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 4604–4614.

Shankar Iyer, Nikhil Dandekar, and Kornél Csernai. 2017. First quora dataset release: Question pairs.

Janet Johnson, Khalil Mrini, Allison Moore, Emilia Farkas, Ndapa Nkashole, Michael Hogarth, and Nadir Weibel. 2020. Voice-based conversational agents for older adults. In *Proceedings of the CHI 2020 Workshop on Conversational Agents for Health and Wellbeing, Honolulu, Hawaii*.

Vladimir Karpukhin, Barlas Oguz, Sewon Min, Patrick Lewis, Ledell Wu, Sergey Edunov, Danqi Chen, and Wen-tau Yih. 2020. Dense passage retrieval for open-domain question answering. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 6769–6781, Online. Association for Computational Linguistics.

Chuan Lei, Vasilis Efthymiou, Rebecca Geis, and Fatma Ozcan. 2020. Expanding query answers on medical knowledge bases. In *EDBT*, pages 567–578.

Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Veselin Stoyanov, and Luke Zettlemoyer. 2020. BART: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7871–7880, Online. Association for Computational Linguistics.

Chin-Yew Lin. 2004. Rouge: A package for automatic evaluation of summaries. In *Text summarization branches out*, pages 74–81.

Yuning Mao, Pengcheng He, Xiaodong Liu, Yelong Shen, Jianfeng Gao, Jiawei Han, and Weizhu Chen. 2021. Generation-augmented retrieval for open-domain question answering. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 4089–4100.

Khalil Mrini, Chen Chen, Ndapa Nakashole, Nadir Weibel, and Emilia Farcas. 2021a. Medical question understanding and answering for older adults. *The 3rd Southern California (SoCal) NLP Symposium*.

Khalil Mrini, Franck Dernoncourt, Walter Chang, Emilia Farcas, and Ndapa Nakashole. 2021b. Joint summarization-entailment optimization for consumer health question understanding. In *Proceedings of the Second Workshop on Natural Language Processing for Medical Conversations*, pages 58–65, Online. Association for Computational Linguistics.

Khalil Mrini, Franck Dernoncourt, Seunghyun Yoon, Trung Bui, Walter Chang, Emilia Farcas, and Ndapa Nakashole. 2021c. A gradually soft multi-task and data-augmented approach to medical question understanding. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 1505–1515, Online. Association for Computational Linguistics.

Khalil Mrini, Franck Dernoncourt, Seunghyun Yoon, Trung Bui, Walter Chang, Emilia Farcas, and Ndapa Nakashole. 2021d. UCSD-adobe at MEDIQA 2021: Transfer learning and answer sentence selection for medical summarization. In *Proceedings of the 20th Workshop on Biomedical Language Processing*, pages 257–262, Online. Association for Computational Linguistics.

Khalil Mrini, Emilia Farcas, and Ndapa Nakashole. 2021e. Recursive tree-structured self-attention for answer sentence selection. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 4651–4661, Online. Association for Computational Linguistics.

Khalil Mrini, Shaoliang Nie, Jiatao Gu, Sinong Wang, Maziar Sanjabi, and Hamed Firooz. 2022. Detection, disambiguation, re-ranking: Autoregressive entity linking as a multi-task problem. In *Findings of the Association for Computational Linguistics: ACL 2022*, pages 1972–1983, Dublin, Ireland. Association for Computational Linguistics.

Kirk Roberts and Dina Demner-Fushman. 2016. Interactive use of online health resources: a comparison of consumer and professional questions. *Journal of the American Medical Informatics Association*, 23(4):802–811.

Vasu Sharma, Nitish Kulkarni, Srividya Pranavi, Gabriel Bayomi, Eric Nyberg, and Teruko Mitamura. 2018. Bioama: towards an end to end biomedical question answering system. In *Proceedings of the BioNLP 2018 workshop*, pages 109–117.

Sheng Shen, Yaliang Li, Nan Du, Xian Wu, Yusheng Xie, Shen Ge, Tao Yang, Kai Wang, Xingzheng Liang, and Wei Fan. 2020. On the generation of

medical question-answer pairs. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pages 8822–8829.

Yi Tay, Luu Anh Tuan, and Siu Cheung Hui. 2018. Multi-cast attention networks for retrieval-based question answering and response prediction. *arXiv preprint arXiv:1806.00778*.

Rafael M Terol, Patricio Martínez-Barco, and Manuel Palomar. 2007. A knowledge based method for the medical question answering problem. *Computers in biology and medicine*, 37(10):1511–1521.

Thomas Wolf, Julien Chaumond, Lysandre Debut, Victor Sanh, Clement Delangue, Anthony Moi, Pierric Cistac, Morgan Funtowicz, Joe Davison, Sam Shleifer, et al. 2020. Transformers: State-of-the-art natural language processing. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 38–45.

Guokai Yan and Jianqiang Li. 2018. Medical question similarity calculation based on weighted domain dictionary. In *Proceedings of the 2018 International Conference on Big Data and Computing*, pages 104–107.

Yi Yang, Wen-tau Yih, and Christopher Meek. 2015. Wikiqa: A challenge dataset for open-domain question answering. In *Proceedings of the 2015 conference on empirical methods in natural language processing*, pages 2013–2018.

Yuan Yang, Jingcheng Yu, Ye Hu, Xiaoyao Xu, and Eric Nyberg. 2017. Cmu livemedqa at trec 2017 liveqa: A consumer health question answering system. *arXiv preprint arXiv:1711.05789*.

Guangtao Zeng, Wenmian Yang, Zeqian Ju, Yue Yang, Sicheng Wang, Ruisi Zhang, Meng Zhou, Jiaqi Zeng, Xiangyu Dong, Ruoyu Zhang, Hongchao Fang, Penghui Zhu, Shu Chen, and Pengtao Xie. 2020. MedDialog: Large-scale medical dialogue datasets. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 9241–9250, Online. Association for Computational Linguistics.

Tianyi Zhang, Varsha Kishore, Felix Wu, Kilian Q Weinberger, and Yoav Artzi. 2019. Bertscore: Evaluating text generation with bert. In *International Conference on Learning Representations*.

A Annotation Details

A.1 Topics Covered by the Knowledge Base

We ask the annotators to limit their questions to the nine sources of MedQUAD. The nine sources from which questions and answer documents are extracted are as follows:

- National Cancer Institute
- Genetic and Rare Diseases Information Center: various aspects of genetic/rare diseases
- Genetics Home Reference (GHR): consumer-oriented information about the effects of genetic variation on human health
- MedlinePlus Health Topics: information on symptoms, causes, treatment and prevention for diseases, health conditions, and wellness issues
- National Institute of Diabetes and Digestive and Kidney Diseases
- National Institute of Neurological Disorders and Stroke: neurological and stroke-related diseases
- NIHSeniorHealth: health and wellness information for older adults
- National Heart, Lung, and Blood Institute (NHLBI): diseases, tests, procedures, and other relevant topics on disorders of heart, lung, blood, and sleep
- Centers for Disease Control and Prevention (CDC)

A.2 Answer Relevance Scoring

We ask annotators to rate answers retrieved by our system and the two baselines according to the following criteria:

- Score of 1/5: The system's answer is completely irrelevant to the question, and does not even contain any concept related to the question.
- Score of 2/5: The system's answer mentions notions that are related to the question, but does not contain a word or concept mentioned in the question.

- Score of 3/5: The system's answer mentions one or more words or concepts from the question, but does not actually answer the question.
- Score of 4/5: The system's answer partially answers the question, mentions one or more words or concepts from the question, but does not fully answer the question.
- Score of 5/5: The system's answer fully answers the question.

A.3 Question Understanding

For question summarization, we evaluate the generated summaries on 4 criteria. We define these criteria for the two healthcare worker annotators as follows:

- Fluency: which generated FAQ is more grammatically correct, and easier to read and to understand?
- Coherence: which generated FAQ is better structured and more organized?
- Informativeness: which generated FAQ captures the most out of the concern of the patient who wrote the CHQ?
- Correctness: which generated FAQ is more factually correct given the CHQ?

A.4 Upwork

We ask annotators to work on Google docs that we share with them. We show in Figure 4 an example of a Google doc that we shared with an annotator (medical doctor) to ask their own question, and the answers we pasted for them to evaluate.

10. TB

    My grandmother was suffering from fever and cough for one months and I was taking care of her. Yesterday she was confirmed as tuberculosis positive and treatment was started. I am worried if I have any risk of being infected by her. However, I have no symptoms at all. Should I get tested for TB?

    System 1: The syndrome usually occurs in children younger than five years; although it has been reported in children up to 13 years. The syndrome is sporadic and non-hereditary. The course of PFAPA can be persistent for years before spontaneous, full resolution.

    System 2: How might periodic fever, aphthous stomatitis, pharyngitis and cervical adenitis be treated? Treatment options that have been successful in improving symptoms of this condition include: oral steroids (prednisone or prednisolone), tonsillectomy with adenoidectomy and cimetidine

    System 3: Tuberculosis (TB) is a disease that is spread through the air from one person to another. Tuberculin skin test: The TB skin test (also called the Mantoux tuberculin skin test) is performed by injecting a small amount of fluid (called tuberculin) into the skin in the lower part of the arm. IGRAs, unlike the TB skin tests, are not affected by prior BCG vaccination and are not expected to give a false-positive result in people who have received BCG.

Relevance Score for System 1: 1/5
Relevance Score for System 2: 1/5
Relevance Score for System 3: 3/5

Figure 4: Example of a Google document, where a hired annotator (medical doctor) asks a question, and rates the answers that we pasted once retrieved by our system and the two baselines.

A Progressive Framework for Role-Aware Rumor Resolution

Lei Chen[1], **Guanying Li**[2], **Zhongyu Wei**[1,3]*, **Yang Yang**[4], **Baohua Zhou**[2], **Qi Zhang**[5], **Xuanjing Huang**[5]
[1] School of Data Science, Fudan University, China
[2] School of Journalism, Fudan University, China
[3] Research Institute of Intelligent and Complex Systems, Fudan University, China
[4] College of Computer Science and Technology, Zhejiang University, China
[5] School of Computer Science, Fudan University, China
[1,3,5]{chenl18,zywei,qi_zhang,xjhuang}@fudan.edu.cn
[2]zhoubaohua@yeah.net;[4]yangya@zju.edu.cn

Abstract

Existing works on rumor resolution have shown great potential in recognizing word appearance and user participation. However, they ignore the intrinsic propagation mechanisms of rumors and present poor adaptive ability when unprecedented news emerges. To exploit the fine-grained rumor diffusion patterns and generalize rumor resolution methods, we formulate a predecessor task to identify triggering posts, and then exploit their characteristics to facilitate rumor verification. We design a tree-structured annotation interface and extend PHEME dataset with labels on the message level. Data analysis shows that triggers play a critical role in verifying rumors and present similar lingual patterns across irrelevant events. We propose a graph-based model considering the direction and interaction of information flow to implement role-aware rumor resolution. Experimental results demonstrate the effectiveness of our proposed model and progressive scheme.

1 Introduction

With the expansion of the Internet, online information tends to spread quickly and widely including fake news, misinformation and rumors, the last of which is defined as circulating stories unverifiable or deliberately false (DiFonzo and Bordia, 2007). Especially in current situation with infectious epidemics and intensive international relationships, researchers have witnessed more than 900% growth in the number of English fact-checks during the COVID-19 outbreak. (Brennen et al., 2020). Automatically verifying rumors has become an urgent need for individuals and society.

Conventional methods for rumor resolution depend on exploiting the evolutionary characteristics of content and spreaders (Kwon et al., 2013; Ma et al., 2015). Benefiting from various attention mechanisms, there is a growing tendency to retrieve evidential messages, indicative tokens and critical users to enhance interpretability (Ma et al., 2019; Lu and Li, 2020; Wu et al., 2020a). However, capturing patterns from historical records faces great challenges while transferring to unprecedented events as rumors evolve quickly and recur infrequently. Besides, extracting features of malicious users skips over the immediate indication and allusive roles of the content itself.

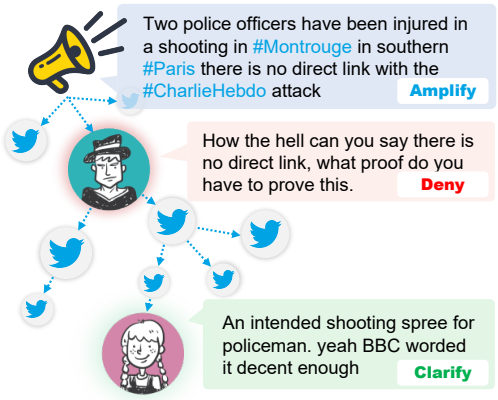

Figure 1: An illustration of rumor cascades and typical roles of messages helpful for rumor verification.

Recently, researchers are dedicated to investigating the intrinsic mechanism of rumor propagation rather than linguistic or rhetorical features. Vosoughi et al. (2018) reveal that it is the *information novelty* that stimulates discussion desire which makes rumors spread faster and deeper. Choi et al. (2020) manage to locate *echo chamber members* who tend to amplify rumor threads and find them responsible for viral propagation. N. Zehmakan and Galam (2020) divide rumor participants into 3 groups (*seeds* who adamantly convince the truth, *agnostics* who firmly reject and *others*) and analyze their roles during diffusion. Inspired by these sociological findings, we propose to explore and model different roles of messages as rumor evolves.

Figure 1 illustrates three types of messages with

* Corresponding author

Proceedings of the 29th International Conference on Computational Linguistics, pages 2748–2758
October 12–17, 2022.

triggering effects via a cascade instance drawn from PHEME dataset (Zubiaga et al., 2016; Kochkina et al., 2018). The source tweet seemingly reports objectively and *amplifies* discussion topics regard to a shooting event. After several rounds of retweet, someone presents an attitude of *denying* and asks for evidence. Finally, a user comes out to *clarify* the real condition and confirm the falseness of the source. Suchlike online discussion is widely existed, however, only a few messages present these critical roles and the majority are insignificant reposts and comments. Accordingly, our goal is to identify ***triggers***, i.e., messages that have prominent effects on rumor proliferation and dominate the judgment of cascade credibility. We also claim that identifying the *role-aware* propagation mode will contribute to rational and sound rumor verification.

To practice the idea, we formulate the task of trigger identification and annotate all the messages in PHEME to form a jointly labeled dataset. Based on the well-adopted graph learning methods, we further put forward the UGRN (**U**nsymmetric **G**raph **R**ecurrent **N**etworks) framework by additionally considering the direction and interaction of information flow to simulate trigger effect. Moreover, we devise the role-aware integration and warm-up strategy to facilitate rumor verification.

Our contributions are of three-folds:

- Following the propagation mechanism of rumors, we formulate trigger identification as a prepositive task of rumor verification and supplement message-level annotations to the PHEME dataset.
- What's more, we design a graph-based framework to jointly identify triggers and verify rumors by progressively modeling tree-structured rumor cascades.
- Taking advantage of our annotated dataset, extensive experiments are conducted and demonstrate the effectiveness of our model.

2 Related Work

As our research focuses on technically resolving rumors in social media, we relate it with existing computational rumor resolution methods and our motivation for applying graph neural networks.

2.1 Rumor Resolution

Since rumors online are enriched with multi-form data, early work concentrates on extracting prominent features from perspectives of content, user reliability and communication impact (Castillo et al., 2011; Kwon et al., 2013; Liu et al., 2015). With deeper understanding of rumor propagation, researchers endeavor to model the whole cascade considering it as time-series sequences (Ma et al., 2015, 2016; Yu et al., 2017), tree-structured diffusion networks (Ma et al., 2018b; Kumar and Carley, 2019; Wei et al., 2019; Ma and Gao, 2020; Li et al., 2020) or the combination of both (Sun et al., 2022).

A remarkable progress lately is to exploit stance information to enhance rumor verification. Kochkina et al. (2018) treat rumor detection, stance classification and rumor verification as a consistent pipeline and confirm the effectiveness of multi-task learning. Following studies explore various mechanism of parameter sharing (Ma et al., 2018a; Wu et al., 2019) and improve efficiency of data usage (Yu et al., 2020). However, annotating stances in all rumor cascades is labor-intensive thus existing corpus cannot reach a perfect match between stance and verification data, which requires superior multi-task training skills and still fails to explain how stance information instructs rumor verification.

Another trend is to excavate the explainability of rumor resolution networks. Ma et al. (2019) utilize hierarchical attention networks to locate evidential sentences. Lu and Li (2020) employ a graph-based co-attention model to capture the relevance between the source text and spreader behavior. Wu et al. (2020a) select suspicious retweets and apply co-attention mechanisms to explore their relationship with the source at token level. Although these approaches can identify unreliable words, sentences and users, they only practice in range-fixed and randomly-split corpus making them short of stability when faced with unknown events.

Recently, researchers attempt to model rumor cascades based on the widely-existed propagation mechanism. Wu and Rao (2020) employ gated mechanisms and devise adaptive interaction fusion networks to model the emotional associations and semantic conflicts which rationalize rumor verification. Chen et al. (2020a) utilize discrete variational autoencoders to model interaction between messages and capture their temporal evolution. Lin et al. (2021) design hierarchical graph attention networks to implement claim-guided rumor detection. Different from their work, our goal is to explicitly identify critical messages with triggering effects so that we construct a jointly labeled dataset.

What's more, we also investigate how triggers progressively facilitate rumor resolution.

2.2 Graph Neural Networks

With increasing complexity of data structure and ingenious construction of intrinsic relation, Graph Neural Networks (GNNs) have gained incremental popularity in modeling topological or tree-structured data (Wu et al., 2020b). Among all the variants, the basic skeleton Graph Convolutional Networks (GCNs) exploit structure information to aggregate and share features of neighbors which provides a rapid and effective solution for node classification, link prediction and community detection (Kipf and Welling, 2017). Following works mainly focus on refined aggregation of adjacent nodes, such as incorporating attention mechanism (Velickovic et al., 2018) and sampling neighbors to avoid over-smoothing and improve computation efficiency (Hamilton et al., 2017).

Nowadays, GNNs are extensively applied in the area of natural language processing, ranging from syntax-based machine translation (Bastings et al., 2017), knowledge-based question answering (Saxena et al., 2020) and aspect-level sentiment classification (Chen et al., 2020b). Despite the tree structure owned by rumor cascades, it is difficult to model cascades directly via GNNs. On one hand, propagation graphs present a high level of heterogeneity in which neighboring nodes usually possess different roles, while most GNNs are only effective for homogenous node classification by sharing mutual features. On the other hand, traditional ways of entire graph learning apply mean or attention pooling which is too coarse to capture evolutionary characteristics of rumor cascades. In this paper, we come up with an innovative way of message passing by inheriting the pioneering idea of Gated Graph Neural Networks (Li et al., 2016) that update node representation via gated recurrent unit, but also considering the direction and integration of information flow.

3 Task and Dataset

3.1 Task Formulation

The task of **rumor verification** is formulated as a supervised classification problem on the cascade level. Given a source tweet r_0, the tree-structured cascade can be constructed with its responsive tweets $\{r_1, r_2, ..., r_T\}$ following the retweet relationship while their textual, temporal and user-related features are available. The goal is to assess the veracity of the cascade by classifying $\mathcal{Y}^v$ into *true*, *false* or *unverified*.

In this paper, we propose a progressive framework to implement **trigger identification** during verification which aims at recognizing the role of each tweet $\mathcal{Y}_i^t$ as *amplify*, *deny*, *clarify* or *null*. *Amplify* indicates tweets that initiate new concerns or enlarge the discussion scale related to the social event. *Deny* means presenting doubt or rejection towards previous messages. *Clarify* introduces factual or substantial information. Other messages are left as *null* which means they are insignificant for rumor propagation or verification.

3.2 Dataset Construction

Our corpus is built on PHEME dataset released by Zubiaga et al. (2016) including TWITTER threads from 5 hot-debated social events. Although subsequently Zubiaga et al. (2018) expand the total event amount to 9, the additional cascades are small-scaled and extremely unbalanced, thus we only consider the original 5 events. They also supplement stance labels on the message level, but only 13% rumor cascades have been annotated limited by visualization technique and labor resources.

To implement role-aware rumor resolution, we annotate triggers for all the messages in rumor cascades. The main difference between two types of message-level labels is that stance just presents sentiment polarity towards the source tweet, while triggers imply their global roles for rumor evolution and are more context-sensitive.

The annotation process consists of three steps. First, We devise a tree-based annotation system containing textual information and propagation path[1]. We remove cascades that only contain source tweet and drop messages missing parents. Second, each cascade is sent to 3 undergraduates who need to read all the tweets in the cascade to understand how the circulating story develops, and then assign trigger labels to messages with critical roles. We only adopt the label if more than 2 people reach an agreement. Other messages are labeled as *null* to ensure the significance of triggering effects. Finally, we evaluate annotation quality with Fleiss's kappa coefficient (Fleiss, 1971) and achieve a moderate agreement of 0.515. Annotating triggers is challenging because social me-

[1] http://fudan-disc.com/project/annotation/propagation/demo.html

dia statements are full of abbreviations and slang words. The incompleteness of cascades caused by privacy restrictions also impedes global comprehension. Statistics of the extended dataset is shown in Table 1.

event	# of cas.	# of mes.	verify dist. (F:T:U)	trigger dist. (N:A:C:D)
CH	449	6110	114:187:148	4705:915:271:219
OS	467	6036	72:327:68	4793:868:254:121
SS	508	7832	76:378:54	5868:1050:471:443
FG	268	4516	8:9:251	3679:527:181:129
GW	237	2377	111:94:32	1762:388:147:80
All	1929	26871	381:995:553	20807:3748:1324:992

Table 1: Statistics of extended PHEME dataset. The abbreviation of different events is in short of *Charlie Hebdo*, *Ottawa Shooting*, *Sydney Siege*, *Ferguson Unrest*, *Germanwings Crash* respectively. The next two columns represents the amount of cascades and messages involved in different events. As for distribution of verification and trigger labels, capital letters stand for possible categories (F: *false*, T: *true*, U: *unverified*, N: *null*, A: *amplify*, C: *clarify*, D: *deny*).

3.3 Data Analysis

For purpose of exploring how triggers interact with neighbors and affect rumor proliferation, we analyze their contextual content continuity and capture their temporal characteristics as rumor develops.

Content Continuity. Since triggers are assumed to be more context-sensitive, we attempt to measure the similarity between successive messages using the ratio of overlapped word count to the longer sentence length. As propagation is irreversible, we differentiate the information flow either from parent nodes or child nodes. Specifically, after calculating the similarity of all the message pairs, we take the average of child posts as the similarity with children. Figure 2 shows the content continuity for different types of triggers in different events.

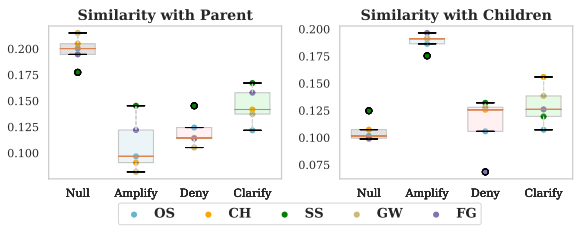

Figure 2: Context similarity for different types of triggers in different events. Scatter points represent the averaged context similarity for a certain kind of trigger in a specific event. Shapes of boxes depict the degree of trigger assimilation for different events.

It shows that triggers with different background tend to possess similar continuity property as the length of all the boxes is comparatively short. The type of *amplify* holds higher probability to bring information novelty compared with its parent and launch discussion associated with it, while the *null* type is totally on the contrary. Words in *deny* posts are less repeated in parent and child message. Posts of *clarify* present a moderate similarity with both parent and children partly because factual information is usually targeted towards the preceding content but also provides hints for further debate.

In addition, we can observe that the property of triggers is naturally endowed no matter what social event they are related to, and the same for their prior categorical distribution (shown in Table 1). Hence, we consider triggers hold higher transfer ability focusing on the universally existed propagation patterns instead of concrete topics or stories, which is helpful to debunk rumors nonexistent in history.

Temporal Variation. In order to investigate what role triggers play for verifying rumors, we calculate the amount of triggers in different diffusion stages. To ensure the amount declination is not from cascades rather small, we select 1,297 cascades whose conversation last for more than 30 minutes and count the number of different triggers in every 3 minutes. Then we count how many triggers of a certain type emerge in each evolution stage for every cascade. As shown in Figure 3, the y-axis represents the averaged number of specific triggers for each cascade in the time interval.

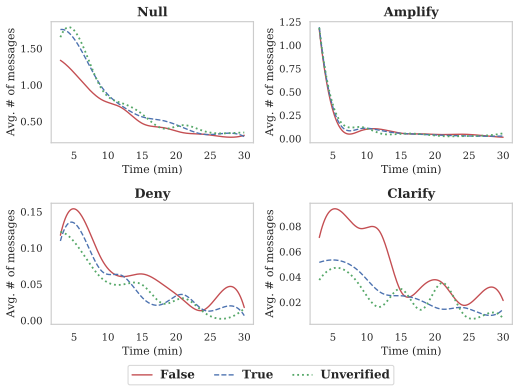

Figure 3: Temporal variation of trigger distribution. Each subgraph represents a certain kind of trigger. Different line styles stand for the category of rumors.

On the whole, the majority of discussion burst in early stages. Except the *amplify*, other types of triggers are distinguishable in different rumors.

Amount of *null* is relatively small in misinformation especially in early stages which means there exist more triggering posts arguing about cascade veracity. *Deny* and *clarify* appear more frequently in false rumors, while *clarify* takes longer time to fade away. With all the findings, we assume trigger identification as an effective way to promote verification and generalization for rumor resolution.

4 Proposed Model

Based on observations in previous section, we propose the **U**nsymmetric **G**raph **R**ecurrent **N**etworks (UGRN) to identify triggers and progressively verify rumors. Figure 4 illustrates the overall architecture which is composed of two components, the sharing GRN layers of two tasks and the trigger-aware prediction module.

4.1 Unsymmetric Graph Recurrent Networks

We use pretrained model to encode textual information for each tweet, and then decompose the propagation tree as two unsymmetric adjacency matrix to employ different GRN layer for interaction direction control.

Graph Initialization. Following the online conversational records, each rumor cascade can be constructed as a graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ where $\mathcal{V}$ represents the set of nodes (messages in this circumstance) and $\mathcal{E}$ represents the set of edges (retweet relationship). Node representation is initialized via the pretrained BERTweet (Nguyen et al., 2020) and fine-tuned afterwards. We directly take the final representation of `[CLS]` token $\boldsymbol{s}$ as the semantically meaningful features of sentences.

Structure Decomposition. After representing messages with pretrained model, node attributes $\mathbf{X} \in \mathbb{R}^{d \times |\mathcal{V}|}$ can be obtained by concatenating $\{\boldsymbol{s}_1, \boldsymbol{s}_2, ..., \boldsymbol{s}_{|\mathcal{V}|}\}$, where d is the dimension of sentence embedding and $|\mathcal{V}|$ represents the total amount of tweets in the cascade.

Generally, edges are represented with a symmetric adjacency matrix $\mathbf{A} \in \mathbb{R}^{|\mathcal{V}| \times |\mathcal{V}|}$, where $\mathbf{A}_{ij} = \mathbf{A}_{ji} = 1$ if there exists an edge between node i and j. However, implementing graph convolution in this way ignores the direction of information flow which is assumed prominent for classifying triggers (denoted in section 3.3). Therefore, we differentiate direction of information flow by decomposing the original adjacency matrix into two unsymmetric matrices $\mathbf{A}^p$ and $\mathbf{A}^c$, where $\mathbf{A}^p_{ij} = 1$ if child node i is connected with parent node j and $\mathbf{A}^c_{ij} = 1$ if parent i is connected with child j.

Since the decomposed adjacency matrix is sparse especially for tree-structured data, we add self-loops to the root (the source tweet) in $\mathbf{A}^p$ and all the leave nodes (the last tweet of propagation path) in $\mathbf{A}^c$ to ensure the sum for each row is larger than 0, thus can be divided for normalization. Figure 4 shows a concrete case to construct $\mathbf{A}^p$ and $\mathbf{A}^c$ for a specific cascade. Then we employ unsymmetric normalized transformation $\hat{\mathbf{A}}^p = (\mathbf{D}^p)^{-1}\mathbf{A}^p$ and $\hat{\mathbf{A}}^c = (\mathbf{D}^c)^{-1}\mathbf{A}^c$ to avoid value scale changing after graph convolution, where $\mathbf{D}^p, \mathbf{D}^c \in \mathbb{R}^{|\mathcal{V}| \times |\mathcal{V}|}$ are the diagonal degree matrices where $\mathbf{D}_{ii}$ equals to the row sum of the adjacency matrix.

Graph Reccurent Networks. Our GRN layer is based on the idea of Gated Graph Neural Networks (Li et al., 2016) but we also employ efficient way of graph convolution (Kipf and Welling, 2017). The GRNs can be extended to L layers, the lth GRN layer ($l \in [1, L]$) can be represented as follows.

First, we utilize the normalized unsymmetric adjacency matrix ($\hat{\mathbf{A}}^p$ for example) to aggregate neighbor information from the top-down direction to acquire the intermediate state for node i,

$$\boldsymbol{h}_{i,p}^{(l-1),p} = \sum_{j \in \{j | \hat{\mathbf{A}}^p_{ij} \neq 0\}} \hat{\mathbf{A}}^p_{ij} \boldsymbol{h}_j^{(l-1),p} \quad (1)$$

where $\boldsymbol{h}$ with a subscript of p means the intermediate information aggregated from parent and the right $\boldsymbol{h}$ without p as subscript is the final output of the $(l-1)^{\text{th}}$ GRN layer.

Then we employ the Long Short-Term Memory unit (Hochreiter and Schmidhuber, 1997) to recurrently implement graph convolution and obtain the output $\boldsymbol{h}_i^{l,p}$ of lth GRN layer,

$$\boldsymbol{f}_i^l = \sigma_g\left(\boldsymbol{W}_f \boldsymbol{h}_i^{(l-1),p} + U_f \boldsymbol{h}_{i,p}^{(l-1),p} + \boldsymbol{b}_f\right) \quad (2)$$

$$\boldsymbol{i}_i^l = \sigma_g\left(\boldsymbol{W}_i \boldsymbol{h}_i^{(l-1),p} + \boldsymbol{U}_i \boldsymbol{h}_{i,p}^{(l-1),p} + \boldsymbol{b}_i\right) \quad (3)$$

$$\boldsymbol{o}_i^l = \sigma_g\left(\boldsymbol{W}_o \boldsymbol{h}_i^{(l-1),p} + \boldsymbol{U}_o \boldsymbol{h}_{i,p}^{(l-1),p} + \boldsymbol{b}_o\right) \quad (4)$$

$$\tilde{\boldsymbol{c}}_i^l = \sigma_c\left(\boldsymbol{W}_c \boldsymbol{h}_i^{(l-1),p} + \boldsymbol{U}_c \boldsymbol{h}_{i,p}^{(l-1),p} + \boldsymbol{b}_c\right) \quad (5)$$

$$\boldsymbol{c}_i^l = \boldsymbol{f}_i^l \circ \boldsymbol{c}_i^{(l-1)} + \boldsymbol{i}_i^l \circ \tilde{\boldsymbol{c}}_i^l \quad (6)$$

$$\boldsymbol{h}_i^{l,p} = \boldsymbol{o}_i^l \circ \sigma_c\left(\boldsymbol{c}_i^l\right) \quad (7)$$

where $\boldsymbol{W} \in \mathbb{R}^{m \times h}$, $\boldsymbol{U} \in \mathbb{R}^{m \times h}$ and $\boldsymbol{b} \in \mathbb{R}^h$ (m is the input size and h is the hidden size) are weight

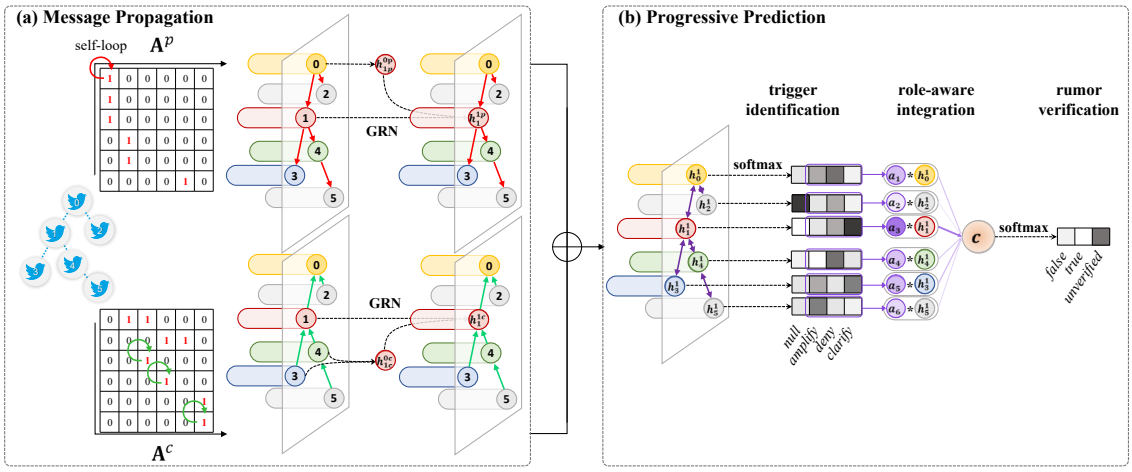

Figure 4: Overall architecture of our proposed model. The two squares on the left represent the decomposed adjacency matrices $\mathbf{A}^p$ and $\mathbf{A}^c$ that control the direction of information flow. Both of the two tasks share the unsymmetric GRN Layers. The updated node representation is used to predict trigger labels. Role-aware integration mechanism is then applied to acquire cascade representation and produce verification prediction.

matrices and bias vector, σ_g and σ_c represent sigmoid and hyperbolic tangent activation functions.

Computing $\boldsymbol{h}_i^{l,c}$ which integrates information from child nodes is identical. Afterwards, we concatenate the representation from parent and children to obtain updated node states $\boldsymbol{h}_i^1$, while 1 denotes that we only adopt one layer of GRN.

4.2 Progressive Prediction

After obtaining the node representation associated with message interaction, we implement node classification to identify triggers and then exploit trigger prediction to integrate nodes and make role-aware verification on cascade level.

Trigger Identification. We simply apply a Feed Forward Network (FFN) and softmax operator to classify each node.

$$\mathcal{Y}_i^t = \text{softmax}(\text{FFN}(\boldsymbol{h}_i^1)) \in \mathbb{R}^4 \tag{8}$$

The loss function of trigger identification is computed by cross-entropy criterion,

$$\mathcal{L}_t = -\frac{1}{|\mathcal{V}|}\sum_{i}^{|\mathcal{V}|}\sum_{j}^{L_t}\mathcal{Y}_i^{t,j}\log\hat{\mathcal{Y}}_i^{t,j} \tag{9}$$

where L_t is the number of trigger classes, $\hat{\mathcal{Y}}_i^{t,j}$ represents the ground-truth label of trigger.

Role-Aware Verification. Since we assume that triggers play an important role in rumor verification, we briefly design a trigger-informed and role-aware pooling mechanism that attends more to triggering posts when integrating the whole cascade. Intuitively, we calculate the weight of each post by dot product to weaken the impact of *null* messages.

$$a_i = \begin{bmatrix}0 & 1 & 1 & 1\end{bmatrix}\mathcal{Y}_i^t \tag{10}$$

Then we apply softmax to normalize attention weights in the cascade and sum up representations of all the nodes considering their role-aware weights to obtain the cascade representation $\boldsymbol{c}$.

$$\boldsymbol{c} = \sum_{i}^{|\mathcal{V}|} a_i \boldsymbol{h}_i^1 \tag{11}$$

Similarly, we make verify prediction and compute the loss function for verification,

$$\mathcal{Y}^v = \text{softmax}(\text{FFN}(\boldsymbol{c})) \in \mathbb{R}^3 \tag{12}$$

$$\mathcal{L}_v = -\sum_{j}^{L_v}\mathcal{Y}^{v,j}\log\hat{\mathcal{Y}}^{v,j} \tag{13}$$

where L_v is the number of verification classes, $\hat{\mathcal{Y}}^{v,j}$ represents the actual label.

Jointly Learning. We add two loss terms to obtain a joint loss function $\mathcal{L}$ for optimization.

$$\mathcal{L} = \mathcal{L}_t + \mathcal{L}_v \tag{14}$$

Moreover, we adopt a warm-up strategy that only reserves $\mathcal{L}_t$ in the first few rounds of training and then employs the overall loss $\mathcal{L}$ to test the validity of progressive learning.

5 Experiments

5.1 Experimental Setup

Data Split. Based on our dataset, we adopt 2 types of cross validation to compare performance and generalizaiton ability of different models. (1) *Random*: to split the dataset into train, validation and test set with a proportion of 8:1:1 randomly. (2) *LOEO*: to implement leave-one-event-out cross validation (Kochkina et al., 2018) which means to treat data equally drawn from a target event as test and validation set and leave others as train set. Although model performance is usually unsatisfactory to implement LOEO validation since semantics differs a lot between events, it is more representative of real world when unprecedented event emerges.

Model Comparison. We compare various models by replacing node updating module with following methods:

CNN: A CNN-based model (Yu et al., 2017) to extract informative local comment.

RNN: A RNN-based model (Ma et al., 2016) that treats rumor cascade as time series to capture dynamic signals.

TreeLSTM: A treeLSTM-based network (Kumar and Carley, 2019) to encode propatation tree.

TreeTrans: A model (Ma and Gao, 2020) using transformer to recursively update tree nodes.

GCN: A GCN-based model (Wei et al., 2019) first treating propagation trees as graphs.

GraphSage: A graph-based model (Li et al., 2020) that randomly samples neighbors to aggregate contextual information.

UGRN: Our proposed model.

Implementation Details. The network is trained with AdamW optimizer (Loshchilov and Hutter, 2017). Hyperparameters performing best in validation set are recorded for testing. The batch size (number of cascades) is set as 5. The hidden unit size for GRN is set as 300. We adopt initial learning rate of 8e-5, 2e-5 respectively for trigger classifier layers and others. The maximum number of training epochs is 100. We have made our extended dataset[2] and code[3] publicly available.

[2] http://fudan-disc.com/data/PHEME_trigger.zip

[3] https://github.com/lchen96/trigger_identification

5.2 Overall Performance

We implement the task of trigger identification and rumor verification to evaluate the performance of our proposed model, as shown in Table 2. Since these two tasks are both evil-balanced, we choose macro F1-score to compare model performance.

Method	Trigger		Verify	
	Random	LOEO	Random	LOEO
CNN	0.524	0.501	0.741	0.308
RNN	0.562	0.560	0.785	0.314
TreeLSTM	0.538	0.514	0.710	0.317
TreeTrans	0.541	0.511	0.714	0.314
GCN	0.548	0.542	0.772	0.322
GraphSage	0.549	0.561	0.781	0.304
UGRN	**0.574**	**0.570**	**0.819**	**0.346**

Table 2: Results of trigger identification and rumor verification. All the numerical values represent macro F1-score when adopting random or LOEO cross validation. The result of LOEO validation is the average of 5 folds. **Bold**: the best performance in each column.

It can be seen that our model can identify triggers more accurately and achieves the highest macro F1-score for verification. Considering the task of **trigger identification**, **CNN** model provides the baseline for trigger identification using pre-trained sentence representation for classification (since features are not updated on message level and pooling for nodes is not applied). Models with reccurent unit (**UGRN**, **RNN** and **TreeLSTM**) is more competitive for trigger identification. Compared with the other two graph-based models (**GCN** and **GraphSage**), our unsymmetric and recurrent framework can model conversational structures and learn high-quality representation of triggering posts while preserving the effective operation of graph convolution. As for the task of **rumor verification**, models all present a drastic decline faced with *LOEO* test. The direct reason probably lies in the extreme imbalance of verify labels between different events and the absence of semantic sharing.

Besides, by comparing the performance between different settings of cross validation, we find that trigger identification is more robust when coming cross underrepresented semantics as the decrease in LOEO setting is not as significant as the task of verification. The direct reason probably lies in the extreme imbalance of verify labels between different events, but we do expect trigger identification to have higher transfer ability which can facilitate handling unprecedented rumor cascades.

5.3 Ablation Study

To examine the effectiveness of key components of our **UGRN** framework, we perform ablation study by degrading the birectional graph-based node representation, as shown in Table 3. From bottom to top, we first substitute the concatenated node representation by only using parent aggregation (**UGRN-p**) or child aggregation (**UGRN-c**). The performance drops a lot when only considering information flow in one direction. Then we leave out the process of structure decomposition and directly use the symmetric adjacency matrix to apply recurrent graph convolution. The simplified model (**GRN**) can hardly distinguish triggers indicating it is valid to model information flow from different directions.

Component	Trigger		Verify	
	Random	LOEO	Random	LOEO
GRN	0.531	0.514	0.754	0.324
UGRN-c	0.541	0.522	0.768	0.321
UGRN-p	0.552	0.541	0.778	0.334
UGRN	**0.574**	**0.570**	**0.819**	**0.346**

Table 3: Ablation study on key components of UGRN. Presentation of result is the same with Table 2.

5.4 Trigger Role for Verification

In this paper, we propose three types of mechanisms to exploit trigger information for enhanced rumor verification, including parameter sharing, trigger-aware cascade pooling and warm-up of trigger identification. We examine the effect of these mechanisms to investigate the role of triggers.

Multi-Task Learning. We run our model on the two tasks separately to demonstrate the validity of multi-task learning. Table 4 shows the comparison between single-task and multi-task settings. As can be seen, the performance gain of multi-task learning is significant especially for the task of rumor verification which demonstrates the strong correlation between these two tasks and the rationality of capturing triggers.

Task	Trigger		Verify	
	Random	LOEO	Random	LOEO
Trigger	0.568	0.558	-	-
Verify	-	-	0.795	0.286
Multi-Task	**0.574**	**0.570**	**0.819**	**0.346**

Table 4: The effect of multi-task learning framework. Presentation of result is the same with Table 2.

Role-Aware Integration. Our model is designed to pay more attention to messages with triggering effect when implementing graph pooling. We replace the role-aware integration with general attention pooling to explore whether triggers can help verify rumors. Figure 5 shows the difference when converting the pooling strategy in different cascade modeling methods. Although the performance contrast is not as obvious as multi-task learning, among these 12 groups of experiments, 9 instances demonstrate role-aware integration is better than plain attention pooling which also covers that locating triggers and take full advantage of their semantics is favorable for rumor verification.

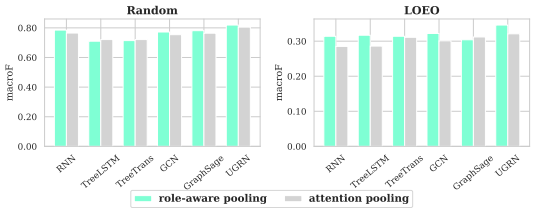

Figure 5: The effect of role-aware integration.

Trigger Warmup. During training, we adopt a warm-up strategy that only trains the network for trigger identification in the first few epochs. Consequently, we set various number of warm-up epochs to see the effectiveness of progressive learning. Figure 6 shows the impact of trigger warm-up in different validation settings. For random validation, 1 rounds of warm-up can slightly improve the verification performance but then the prediction precision drops a lot as warm-up epochs increase. However, the averaged performance of LOEO validation is steadily increasing with increment of warm-up epochs. Except when treating *Charlie Hebdo* as test event, other folds perform better with larger warm-up epochs. This is partly because the network tends to learn refined node representation under the supervision signal of triggers and provides better initialization for verification.

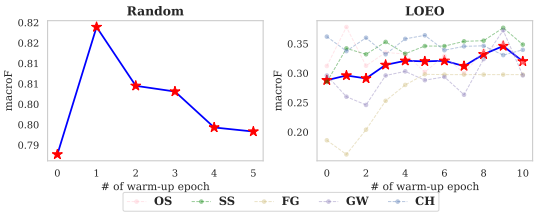

Figure 6: The effect of trigger warm-up strategy. Blue solid lines represent the averaged result and the dashed lines stand for results with different test event.

6 Conclusion and Future Work

In this paper, we propose the task of trigger identification to progressively resolve rumors. We extend PHEME dataset with annotations on message level. We design the framework of Unsymmetric Graph Reccurent Networks which significantly improves performance of two tasks. In the future, we would like to further model the relationship between triggers and rumor cascades.

Acknowledgement

This work is partially supported by Natural Science Foundation of China (No.6217020551, 71991471), National Social Science Foudation of China (No. 20ZDA060), Science and Technology Commission of Shanghai Municipality Grant (No.20dz1200600, 21QA1400600, 21511101000) and Zhejiang Lab (No. 2019KD0AD01).

References

Jasmijn Bastings, Ivan Titov, Wilker Aziz, Diego Marcheggiani, and Khalil Sima'an. 2017. Graph convolutional encoders for syntax-aware neural machine translation. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 1957–1967, Copenhagen, Denmark.

J Scott Brennen, Felix Simon, Philip N Howard, and Rasmus Kleis Nielsen. 2020. Types, sources, and claims of covid-19 misinformation. *Reuters Institute*, 7(3):1.

Carlos Castillo, Marcelo Mendoza, and Barbara Poblete. 2011. Information credibility on twitter. In *Proceedings of the 20th international conference on World wide web*, pages 675–684. ACM.

Lei Chen, Zhongyu Wei, Jing Li, Baohua Zhou, Qi Zhang, and Xuanjing Huang. 2020a. Modeling evolution of message interaction for rumor resolution. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 6377–6387, Barcelona, Spain (Online).

Xiao Chen, Changlong Sun, Jingjing Wang, Shoushan Li, Luo Si, Min Zhang, and Guodong Zhou. 2020b. Aspect sentiment classification with document-level sentiment preference modeling. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 3667–3677, Online.

Daejin Choi, Selin Chun, Hyunchul Oh, Jinyoung Han, et al. 2020. Rumor propagation is amplified by echo chambers in social media. *Scientific reports*, 10(1):1–10.

Nicholas DiFonzo and Prashant Bordia. 2007. *Rumor psychology: Social and organizational approaches.* American Psychological Association.

Joseph L Fleiss. 1971. Measuring nominal scale agreement among many raters. *Psychological bulletin*, 76(5):378.

William L. Hamilton, Zhitao Ying, and Jure Leskovec. 2017. Inductive representation learning on large graphs. In *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, December 4-9, 2017, Long Beach, CA, USA*, pages 1024–1034.

Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long short-term memory. *Neural computation*, 9(8):1735–1780.

Thomas N. Kipf and Max Welling. 2017. Semi-supervised classification with graph convolutional networks. In *5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Conference Track Proceedings*.

Elena Kochkina, Maria Liakata, and Arkaitz Zubiaga. 2018. All-in-one: Multi-task learning for rumour verification. *arXiv preprint arXiv:1806.03713*.

Sumeet Kumar and Kathleen M Carley. 2019. Tree lstms with convolution units to predict stance and rumor veracity in social media conversations. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 5047–5058.

Sejeong Kwon, Meeyoung Cha, Kyomin Jung, Wei Chen, and Yajun Wang. 2013. Prominent features of rumor propagation in online social media. In *2013 IEEE 13th international conference on data mining*, pages 1103–1108. IEEE.

Jiawen Li, Yudianto Sujana, and Hung-Yu Kao. 2020. Exploiting microblog conversation structures to detect rumors. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 5420–5429, Barcelona, Spain (Online). International Committee on Computational Linguistics.

Yujia Li, Richard Zemel, Marc Brockschmidt, and Daniel Tarlow. 2016. Gated graph sequence neural networks. In *Proceedings of ICLR'16*.

Hongzhan Lin, Jing Ma, Mingfei Cheng, Zhiwei Yang, Liangliang Chen, and Guang Chen. 2021. Rumor detection on twitter with claim-guided hierarchical graph attention networks. *arXiv preprint arXiv:2110.04522*.

Xiaomo Liu, Armineh Nourbakhsh, Quanzhi Li, Rui Fang, and Sameena Shah. 2015. Real-time rumor debunking on twitter. In *Proceedings of the 24th ACM International on Conference on Information and Knowledge Management*, pages 1867–1870. ACM.

Ilya Loshchilov and Frank Hutter. 2017. Decoupled weight decay regularization. *arXiv preprint arXiv:1711.05101*.

Yi-Ju Lu and Cheng-Te Li. 2020. Gcan: Graph-aware co-attention networks for explainable fake news detection on social media. Association for Computational Linguistics.

Jing Ma and Wei Gao. 2020. Debunking rumors on Twitter with tree transformer. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 5455–5466, Barcelona, Spain (Online). International Committee on Computational Linguistics.

Jing Ma, Wei Gao, Shafiq Joty, and Kam-Fai Wong. 2019. Sentence-level evidence embedding for claim verification with hierarchical attention networks. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 2561–2571, Florence, Italy.

Jing Ma, Wei Gao, Prasenjit Mitra, Sejeong Kwon, Bernard J Jansen, Kam-Fai Wong, and Meeyoung Cha. 2016. Detecting rumors from microblogs with recurrent neural networks. In *International Joint Conferences on Artificial Intelligence*, pages 3818–3824.

Jing Ma, Wei Gao, Zhongyu Wei, Yueming Lu, and Kam-Fai Wong. 2015. Detect rumors using time series of social context information on microblogging websites. In *Proceedings of the 24th ACM international on conference on information and knowledge management*, pages 1751–1754.

Jing Ma, Wei Gao, and Kam-Fai Wong. 2018a. Detect rumor and stance jointly by neural multi-task learning. In *Companion of the The Web Conference 2018 on The Web Conference 2018*, pages 585–593. International World Wide Web Conferences Steering Committee.

Jing Ma, Wei Gao, and Kam-Fai Wong. 2018b. Rumor detection on twitter with tree-structured recursive neural networks. Association for Computational Linguistics.

Ahad N. Zehmakan and Serge Galam. 2020. Rumor spreading: A trigger for proliferation or fading away. *Chaos: An Interdisciplinary Journal of Nonlinear Science*, 30(7):073122.

Dat Quoc Nguyen, Thanh Vu, and Anh Tuan Nguyen. 2020. BERTweet: A pre-trained language model for English Tweets. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 9–14.

Apoorv Saxena, Aditay Tripathi, and Partha Talukdar. 2020. Improving multi-hop question answering over knowledge graphs using knowledge base embeddings. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 4498–4507, Online.

Mengzhu Sun, Xi Zhang, Jiaqi Zheng, and Guixiang Ma. 2022. Ddgcn: Dual dynamic graph convolutional networks for rumor detection on social media.

Petar Velickovic, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Liò, and Yoshua Bengio. 2018. Graph attention networks. In *6th International Conference on Learning Representations, ICLR 2018, Vancouver, BC, Canada, April 30 - May 3, 2018, Conference Track Proceedings*.

Soroush Vosoughi, Deb Roy, and Sinan Aral. 2018. The spread of true and false news online. *Science*, 359(6380):1146–1151.

Penghui Wei, Nan Xu, and Wenji Mao. 2019. Modeling conversation structure and temporal dynamics for jointly predicting rumor stance and veracity. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 4787–4798, Hong Kong, China.

Lianwei Wu and Yuan Rao. 2020. Adaptive interaction fusion networks for fake news detection. *Frontiers in Artificial Intelligence and Applications*, 325:2220–2227.

Lianwei Wu, Yuan Rao, Haolin Jin, Ambreen Nazir, and Ling Sun. 2019. Different absorption from the same sharing: Sifted multi-task learning for fake news detection. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 4644–4653, Hong Kong, China.

Lianwei Wu, Yuan Rao, Yongqiang Zhao, Hao Liang, and Ambreen Nazir. 2020a. Dtca: Decision tree-based co-attention networks for explainable claim verification. Association for Computational Linguistics.

Zonghan Wu, Shirui Pan, Fengwen Chen, Guodong Long, Chengqi Zhang, and S Yu Philip. 2020b. A comprehensive survey on graph neural networks. *IEEE transactions on neural networks and learning systems*.

Feng Yu, Qiang Liu, Shu Wu, Liang Wang, Tieniu Tan, et al. 2017. A convolutional approach for misinformation identification.

Jianfei Yu, Jing Jiang, Ling Min Serena Khoo, Hai Leong Chieu, and Rui Xia. 2020. Coupled hierarchical transformer for stance-aware rumor verification in social media conversations. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1392–1401, Online.

Arkaitz Zubiaga, Elena Kochkina, Maria Liakata, Rob Procter, Michal Lukasik, Kalina Bontcheva, Trevor Cohn, and Isabelle Augenstein. 2018. Discourse-aware rumour stance classification in social media using sequential classifiers. *Information Processing & Management*, 54(2):273–290.

Arkaitz Zubiaga, Maria Liakata, and Rob Procter. 2016. Learning reporting dynamics during breaking news for rumour detection in social media. *arXiv preprint arXiv:1610.07363*.

Uncertainty-aware Propagation Structure Reconstruction for Fake News Detection

Lingwei Wei[1,2] and **Dou Hu**[1,2] and **Wei Zhou**[1*] and **Songlin Hu**[1,2]
[1] Institute of Information Engineering, Chinese Academy of Sciences
[2] School of Cyber Security, University of Chinese Academy of Sciences
{weilingwei, hudou, zhouwei, husonglin}@iie.ac.cn

Abstract

The widespread of fake news has detrimental societal effects. Recent works model information propagation as graph structure and aggregate structural features from user interactions for fake news detection. However, they usually neglect a broader propagation uncertainty issue, caused by some missing and unreliable interactions during actual spreading, and suffer from learning accurate and diverse structural properties. In this paper, we propose a novel dual graph-based model, Uncertainty-aware Propagation Structure Reconstruction (UPSR) for improving fake news detection. Specifically, after the original propagation modeling, we introduce propagation structure reconstruction to fully explore latent interactions in the actual propagation. We design a novel Gaussian Propagation Estimation to refine the original deterministic node representation by multiple Gaussian distributions and arise latent interactions with KL divergence between distributions in a multi-facet manner. Extensive experiments on two real-world datasets demonstrate the effectiveness and superiority of our model.

1 Introduction

Nowadays, *fake news*[1] has posed detrimental effects on individuals and society. For example, telecommunication towers were burned due to a conspiracy theory linking COVID-19 with 5G technology (Ahmed et al., 2020). To help mitigate the negative effects caused by fake news, it's critical to develop automatic methods to detect fake news.

Existing works generally leverage the user interactions (*e.g.,* retweet) and shared content in a social media conversation thread to detect fake news. The key principle behind such work is that users on social media share opinions, conjectures and evidence for checking fake news. Some studies (Ruchansky et al., 2017; Ma et al., 2016) flatten the conversation in a chronological order to catch linguistic and temporal features from the propagation sequence, which does not make better use of the network properties. Some works (Ma et al., 2018; Kumar and Carley, 2019; Khoo et al., 2020; Ma and Gao, 2020) build the conversation thread with a tree structure to capture the structural patterns from the interactions of information propagation. Driven by the success of graph neural networks (Kipf and Welling, 2017), recent methods (Bian et al., 2020; Hu et al., 2021; Lin et al., 2021) regard the conversation thread as a graph structure and aggregate informative neighbors to learn a good representation for detection.

However, most methods usually assume that the propagation structure is deterministic and complete at some point. In the real world, it is often the case that each sample describes a partial propagation structure that includes some missing and unreliable interactions due to various reasons such as personal privacy protection and profit-driven social bots (Shao et al., 2018). This fact contributes to the propagation uncertainty issue and makes it challenging to discover effective structural patterns for fake news detection. Wei et al. (2021) learned relational bias to alleviate the negative effect of unreliable interactions. But they only focus on explicit interactions between a tweet and its direct retweets. Thus, they still ignore some latent interactions that are not connected but may share similar stances that are useful to debunk false information. These vital but missing latent interactions in the social media conversation thread are also key to driving the propagation uncertainty issue. Thus, how to model the propagation uncertainty issue and learn effective structure-property is a practical research topic to enhance fake news detection.

An intuitive way is to reconstruct the original propagation structure to capture all possible interactions between posting nodes. We argue that, in

* Corresponding author.

[1]We adopts a broad definition, i.e., *fake news is false news where news broadly includes claims, statements, posts, among other types of information* (Zhou and Zafarani, 2020).

Proceedings of the 29th International Conference on Computational Linguistics, pages 2759–2768
October 12–17, 2022.

the propagation, many retweets that subconsciously promote each other (such as similar stances or emotions). Hu et al. (2021); Lin et al. (2021) have shown the positive gains of implicit interactions between sibling retweets from the same tweet. Beyond their assumptions, we make the attempt to investigate more potential interactions of all postings in the propagation structure, not limited to sibling retweets. Besides, previous works (Wei et al., 2021; Hu et al., 2021; Lin et al., 2021) usually measure interactions by learning deterministic embedding of each tweet, which may be insufficient to depict potential interactions accurately and comprehensively for uncertain propagation. Therefore, it is desirable to study potential interactions from multiple underlying facets, which can reflect their fuzzy stances, emotions, and other factors.

In this paper, we investigate a broader propagation uncertainty issue caused by missing and unreliable interactions. Towards this issue, we develop a novel dual graph-based model, named Uncertainty-aware Propagation Structure Reconstruction (UPSR), to adaptively learn accurate and diverse structural properties. Specifically, inspired by Chen et al. (2020), we first utilize deep graph convolutional networks to fully model long-range interactions in the original propagation. Then, instead of directly using deterministic node representations for reconstruction, we design a novel Gaussian Propagation Estimation to sample node representations from multiple Gaussian distributions where the covariance enables the model to reduce noisy interactions. We measure the Kullback-Leibler (KL) divergence between distributions in a multi-facet manner to update the propagation structure. Based on the reconstructed graph, we apply root-aware graph convolutional networks to aggregate features based on the learned latent interactions. UPSR's dual graph structure can not only learn accurate structural information in the original propagation but also capture diverse structural patterns in the reconstructed propagation. Finally, we exploit the dual-graph representation to identify fake news.

We conduct extensive experiments on two real-world public datasets. The experimental results show that UPSR significantly outperforms the state-of-the-art models, indicating the effectiveness for fake news detection. The core contributions of this paper are summarized as follows:

- To handle a broader propagation uncertainty issue caused by missing and unreliable relations, we propose a novel Uncertainty-aware Propagation Structure Reconstruction (UPSR) to learn accurate and diverse structural properties for fake news detection.
- We design a Gaussian Propagation Estimation (GPE) to reconstruct latent propagation structure by measuring KL divergence between different Gaussian distributions of retweets.
- We evaluate the model on two real-world benchmark datasets. Experimental results demonstrate the effectiveness and superiority of the proposed model.

2 Related Work

In the literature, some works (Jiang et al., 2019; Shu et al., 2019b; Mishra, 2020; Nguyen et al., 2020) leverage user characteristics to assist detection. As user information is not allowed recorded in many cases, we mainly focus on detecting fake news based on *text* and *propagation*.

Text-based fake news detection approaches (Mihalcea and Strapparava, 2009) emphasize investigating the truthfulness of news content by extracting its textual features. Early works relied on feature engineering to capture textual characteristics, *e.g.,* topic features (Castillo et al., 2011), writing styles and consistency (Popat, 2017; Potthast et al., 2018). After the emergence of deep learning, many works (Ma et al., 2016; Ruchansky et al., 2017; Karimi and Tang, 2019) apply various neural networks to automatically learn rich semantic or syntactic features from the source news and its retweets to detect fake news.

Propagation-based fake news detection approaches take advantage of the information related to the dissemination of a news article. Many empirical studies (Vosoughi et al., 2018; Jang et al., 2018) have shown that compared to real news, fake news has deeper propagation structures, and reaches a wider audience. Shu et al. (2019a) jointly learned the sequential effect of comments and correlation between source news and the corresponding comments. To capture structural propagation patterns, Ma et al. (2016) constructed a tree-structured neural network to model the propagation structure. Khoo et al. (2020) adopted Transformer (Vaswani et al., 2017) to learn long-distance interactions. Recently, Bian et al. (2020) regarded the propagation as a graph and applied two graph convolutional networks (GCNs) (Kipf and Welling, 2017)

to learn structural patterns from two distinct directed graphs. Hu et al. (2021); Lin et al. (2021) further explored multi-relational interactions in the propagation graph. Wei et al. (2021, 2022) focused on the propagation uncertainty and learned robust structural features.

Differences with Existing Models. 1) The aforementioned graph-based models (Bian et al., 2020; Hu et al., 2021) are shallow structure, limiting to explore latent interactions in a deeper propagation. Inspired by Chen et al. (2020), we stack more graph layers to explore long-range interactions in propagation. 2) Most approaches learn latent structural features on statics propagation trees/graphs. They may be disturbed by missing and unreliable behaviors easily, leading to a broader propagation uncertainty issue. This paper designs modules to reconstruct original propagation and explore more latent interactions from multiple facets.

3 Problem Statement

Formally, let $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ be a propagation structure, where $\mathcal{V} = \{r, c_1, ..., c_N\}$ is a set of nodes representing the source news r and its retweets $c_1, ..., c_N$. $\mathcal{E}$ refers to a set of explicit interactive behaviors, *e.g.,* retweet. Define the embedding of the source news r as $\mathbf{r} \in \mathbb{R}^{d_0}$, and that of a retweet $\mathbf{c}_i \in \mathbb{R}^{d_0}$, where d_0 is the dimensionality of textual features. Each propagation is annotated with a ground-truth label $y_i \in \{0, 1\}$.

We formulate the fake news detection problem as a binary classification problem, *i.e.,* each sample can be real ($y_i = 0$) or fake ($y_i = 1$). Fake news detection task can be seen as to learn a classifier f from the labeled set, *i.e.,* $f : \mathcal{G} \rightarrow y$.

4 The Proposed Model

In this section, we propose a novel dual graph-based model, **UPSR**, to fully model long-range dependencies in the original propagation and explore rich latent dependencies in the corresponding reconstructed propagation.

4.1 Overview

The overview architecture of UPSR is presented in Figure 1. Firstly, given the input text and propagation structure, we apply deep graph convolutions to learn long-range interactions in the original propagation. To better alleviate the propagation uncertainty issue, we design a Gaussian Propagation Estimation to reconstruct the propagation to discover more potential interactions. Then, based on the reconstructed propagation, we further aggregate node features with the guidance of latent connections. Finally, both node representations encoded in the original and latent propagation are concatenated for fake news classification.

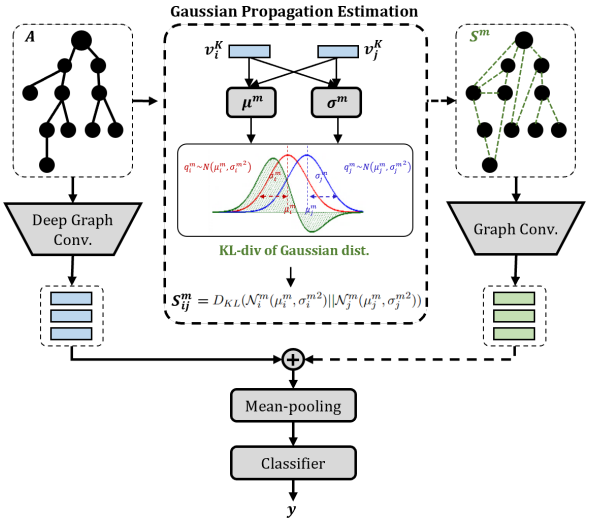

Figure 1: The overall architecture of UPSR.

4.2 Original Propagation Modeling

Vosoughi et al. (2018) have verified that fake news diffused significantly farther, deeper, and more broadly than the truth. Thus, modeling long-range interactions in the propagation are critical to differentiate fake news and true news. Inspired by (Chen et al., 2020), we develop a deep graph convolutional network to capture long-range interactions in the original propagation.

4.2.1 Graph Construction

First, we construct an undirected graph for each propagation structure to aggregate bi-directional interactions comprehensively. Formally, a propagation structure can be represented as an undirected graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$, where $\mathcal{V}$ denotes a set of tweet nodes including source news r and its retweets $c_1, ..., c_n$. $\mathcal{E}$ is a set of propagation behaviors. The edge weights are set to 1 if there is an edge between two nodes, *i.e.,* $\mathbf{A}_{ij} = 1$.

4.2.2 Learning Long-Range Interactions in the Original Propagation Graph

Chen et al. (2020) improved traditional graph convolutional networks by introducing the initial residual connection and an identity mapping to enable stack multiple graph layers, which has shown

promising performance on recent downstream applications (Hu et al., 2022). For information propagation, Vosoughi et al. (2018); Jang et al. (2018) have shown that compared to real news, fake news has deeper propagation structures, and reaches a wider audience. Therefore, we apply deep graph convolutional networks (Chen et al., 2020) on an undirected graph to fully capture this kind of long-range dependencies between two nodes in the original propagation.

Given the undirected graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$, the graph convolution at the k-th layer is defined as Eq. (1). A residual connection to the first layer $\mathbf{V}^{(0)}$ is added to the representation $\tilde{\mathbf{P}}\mathbf{V}^{(k)}$ and an identity mapping $\mathbf{I}$ is added to the weight matrix $\mathbf{W}_t^{(k)}$. $\mathbf{V}^{(0)}$ is initialized with the input embedding, *i.e.,* $\mathbf{V}^{(0)} = [\mathbf{r}, \mathbf{c}_1, ..., \mathbf{c}_N]$.

$$\mathbf{V}^{(k+1)} = \sigma\left(((1-\alpha_k)\tilde{\mathbf{P}}\mathbf{V}^{(k)} + \alpha_k\mathbf{V}^{(0)})((1-\beta_k)\mathbf{I}_n + \beta_k\mathbf{W}_t^{(k)})\right), \tag{1}$$

where $\tilde{\mathbf{P}} = (\mathbf{D}+\mathbf{I})^{-1/2}(\mathbf{A}+\mathbf{I})(\mathbf{D}+\mathbf{I})^{-1/2}$ is the renormalized graph Laplacian matrix (Kipf and Welling, 2017). $\mathbf{A}$ is the original adjacency matrix of $\mathcal{G}$. $\mathbf{D}$ is the diagonal degree matrix, and $\mathbf{I}$ is the identity matrix. α_k, β_k are two hyperparameters. In experiments, $\alpha_k = 0.1$ to make node representations consist of at least a fraction of the input features even if we stack many layers. Let $\beta_k = \log(\frac{\eta}{k} + 1)$ to ensure the decay of the weight matrix adaptively increases when stacking more layers. η is also a hyperparameter. $\mathbf{W}_t^{(k)}$ is the k-th weight matrix. σ denotes the activation function.

Based on the above modifications, we can stack many graph layers to capture long distant connections in the original propagation and provide more accurate node representations for the subsequent reconstructed propagation modeling. We denote the number of graph layers as K and final node representations as $\mathbf{V}^{(K)} = \{\mathbf{v}_r^{(K)}, \mathbf{v}_1^{(K)}, ..., \mathbf{v}_N^{(K)}\}$.

4.3 Reconstructed Propagation Modeling

To explore diverse structural patterns, we reconstruct the original propagation for finding more latent interactions and then encode the reconstructed propagation graph for improving detection.

4.3.1 Gaussian Propagation Estimation

We design a Gaussian Propagation Estimation (GPE) to reconstruct the original propagation from multiple facets. Instead of directly measuring the original deterministic embedding of each tweet, the GPE module generate samples stochastic node representations from multiple Gaussian distributions. It can depict potential interactions accurately and comprehensively for uncertain propagation.

Formally, given the deterministic embedding $\mathbf{v}_i^{(K)}$ of each node v_i, the uncertainty-aware node representations is defined as distributional estimation parameterised with estimated mean $\boldsymbol{\mu}_i^m$ and estimated variance $\boldsymbol{\sigma}_i^m$,

$$\begin{aligned}\{\boldsymbol{\mu}_i^1, \boldsymbol{\mu}_i^2, ..., \boldsymbol{\mu}_i^M\} &= g_\theta(\mathbf{v}_i^{(K)}) \\ \{\boldsymbol{\sigma}_i^1, \boldsymbol{\sigma}_i^2, ..., \boldsymbol{\sigma}_i^M\} &= \phi(g'_\theta(\mathbf{v}_i^{(K)})),\end{aligned} \tag{2}$$

where M is a parameter representing the number of facets to estimate uncertain effects of nodes. g_θ and g'_θ are two trainable neural networks such as a multilayer perception (MLP). ϕ is a non-linear activation function. $\{\boldsymbol{\sigma}^1, \boldsymbol{\sigma}^2, ..., \boldsymbol{\sigma}^M\}$ indicate the uncertainty of tweets which impacts others in a multi-facet manner. Then, the node representations $\mathbf{Q}^m = \{\mathbf{q}_r^m, \mathbf{q}_1^m, ..., \mathbf{q}_N^m\}$ at the m-th view latent propagation can be sampled from $\mathcal{N}_i^m(\boldsymbol{\mu}_i^m, \boldsymbol{\sigma}_i^{m2})$,

$$\mathbf{q}_i^m = \boldsymbol{\mu}_i^m + \epsilon\boldsymbol{\sigma}_i^m, \epsilon \in \mathcal{N}(\mathbf{0}, \mathbf{I}). \tag{3}$$

Then, GPE measures the latent interactions between nodes with KL divergence between distributions from multiple underlying facets. The edge weight between node v_i and node v_j on the m-th view reconstructed graph is computed as,

$$\mathbf{S}_{ij}^m = D_{KL}(\mathcal{N}_i^m(\boldsymbol{\mu}_i^m, \boldsymbol{\sigma}_i^{m2}) || \mathcal{N}_j^m(\mu_j^m, \boldsymbol{\sigma}_j^{m2})). \tag{4}$$

According to the above computations, we can obtain multi-view refined node representations $\{\mathbf{Q}^1, \mathbf{Q}^2, ..., \mathbf{Q}^M\}$ and the corresponding adjacent matrices $\{\mathbf{S}^1, \mathbf{S}^2, ..., \mathbf{S}^M\}$. They enable the model to learn uncertain effects of nodes in multiple reconstructed directed graphs.

4.3.2 Re-Learning Potential Interactions in the Reconstructed Propagation Graph

Based on these reconstructed graphs, we further apply two-layer graph convolutions to capture different potential interactions between two tweets. The message-passing is defined as,

$$\mathbf{U}^m = \sigma\left(\hat{\mathbf{S}}^m(\sigma\left(\hat{\mathbf{S}}^m\mathbf{Q}^m\mathbf{W}_g^{(0)}\right))\mathbf{W}_g^{(1)}\right), \tag{5}$$

where $\hat{\mathbf{S}}$ represents the normalization of adjacency matrix $\mathbf{S}$. $\mathbf{W}_g^{(0)}$ and $\mathbf{W}_g^{(1)}$ are learnable parameter matrices in the first and second graph layer.

Inspired by Bian et al. (2020), we concatenate hidden feature vectors of each node with that of the root node after each graph convolution operation to emphasize the vital role of source news in the propagation. Then, the final representation of nodes in the reconstructed graph is computed as,

$$\mathbf{Z} = \mathbf{W}_z[\mathbf{U}^1; \mathbf{U}^2; ...; \mathbf{U}^M] + \mathbf{b}_z, \quad (6)$$

where $\mathbf{W}_z$ and $\mathbf{b}_z$ are trainable parameters.

Through the above dual graph structure, we can not only learn long-range interactions in the original propagation but also capture potential interactions between uncertain tweets.

We aggregate node representations in the graph to form the graph representations. Given node representations $\mathbf{V}$ in the original propagation and node representations $\mathbf{Z}$ in the reconstructed graph, the graph representation is computed as,

$$\mathbf{O} = meanpooling([\mathbf{V}; \mathbf{Z}]), \quad (7)$$

where $meanpooling(\cdot)$ refers to the mean-pooling aggregating function.

4.4 Fake News Detection and Training

Based on the concatenation of two distinct graph representations, label probabilities of all classes can be defined by a full connection layer and a softmax function, *i.e.,*

$$\hat{\mathbf{y}} = softmax\left(\mathbf{W}_o\mathbf{O} + \mathbf{b}_o\right), \quad (8)$$

where $\mathbf{W}_o$ and $\mathbf{b}_o$ are learnable parameter matrices.

We optimize the fake news classification loss function calculated by the cross-entropy criterion, i.e.,

$$\mathcal{L} = -\mathbf{y}\log(\hat{\mathbf{y}}) - (1-\mathbf{y})\log(1-\hat{\mathbf{y}}), \quad (9)$$

where $\mathbf{y}$ is the ground-truth label and $\hat{\mathbf{y}}$ is the prediction distribution.

5 Experiments

In this section, we experimentally evaluate the performance of our proposed model for fake news detection.

5.1 Datasets

The dataset statistics are shown in Table 1. **PolitiFact** and **GossipCop** datasets are released by FakeNewsNet (Shu et al., 2020). Samples are collected from *PolitiFact*[2] and *GossipCop*[3], which are two websites for fact-checking political and celebrity news, respectively. We follow the same procedure as Shu et al. (2019a) to split each dataset, *i.e.,* randomly choose 75% of the news as the training data while keeping the rest as the test data.

Dataset	PolitiFact	GossipCop
# News	314	5,464
# True News	157	2,732
# Fake News	157	2,732
# Retweets	40,740	308,798
# Avg. Nodes per Graph	131	58
# Avg. Breadth per Graph	73.62	44.35
# Avg. Depth per Graph	3.75	2.51

Table 1: The statistics of two benchmark datasets.

5.2 Experimental Setups

Since the fake news detection is a classification task, we choose accuracy (Acc), prevision (P), recall (R), and macro-average F1 scores (F1) to measure the performance of each model.

All experiments are conducted on a single GeForce RTX 3080Ti. For the input features of text contents, we follow (Dou et al., 2021) and consider 300-dimensional word2vec vectors (Mikolov et al., 2013), which are pretrained on a large corpus with 680k words by spaCy (Honnibal and Montani, 2017), *i.e.,* $d_0 = 300$. The dimension of hidden vectors is set to 64. We train all models via backpropagation and a wildly used stochastic gradient descent named Adam (Kingma and Ba, 2015). The learning rate is set to 0.001 and 0.0005 for *PolitiFact* and *GossipCop*, respectively. The training process is iterated upon 200 epochs and early stopping (Yuan et al., 2007) is applied when the validation loss stops decreasing by 10 epochs. The final result is the average performance over 5 repeats.

5.3 Comparison Methods

Text-based fake news detection methods include: **mGRU** (Ma et al., 2016) uses an RNN to capture temporal-linguistic patterns recognized from sequences of retweets. **CSI** (Ruchansky et al., 2017) learns the sequential retweet features by employing an LSTM. *Propagation-based* fake news detection methods include: **GCNFN** (Monti et al., 2019) models the propagation structure as a graph

[2] https://www.politifact.com/
[3] https://www.gossipcop.com/

Method	PolitiFact				GossipCop			
	Acc	P	R	F1	Acc	P	R	F1
mGRU (Ma et al., 2016)	0.754	0.800	0.666	0.744	0.859	0.845	0.881	0.859
CSI (Ruchansky et al., 2017)	0.734	0.672	0.550	0.688	0.866	0.892	0.840	0.866
GAT (Velickovic et al., 2018)	0.861	0.848	0.883	0.853	0.958	0.957	0.959	0.957
GCNFN (Monti et al., 2019)	0.856	0.862	0.851	0.849	0.886	0.892	0.881	0.883
PLAN (Khoo et al., 2020)	0.868	0.861	0.879	0.858	0.962	0.960	0.945	0.953
BiGCN (Bian et al., 2020)	0.861	0.865	0.877	0.853	0.959	0.959	0.959	0.958
RumorGCN (Hu et al., 2021)	0.891	0.901	0.875	0.888	0.968	0.965	0.971	0.968
EBGCN (Wei et al., 2021)	0.896	0.898	0.909	0.891	0.964	0.966	0.962	0.963
UPSR	**0.914**	**0.911**	**0.917**	**0.910**	**0.977**	**0.980**	**0.974**	**0.976**

Table 2: Model performance for fake news detection on *PolitiFact* and *GossipCop*. The best result is in bold-face.

and uses GCN to encode the propagation graph. We implemented the model by removing profile information for fair comparison. **GAT** (Velickovic et al., 2018) applies graph attention networks to encode the propagation structure. **PLAN** (Khoo et al., 2020) uses the multi-head attention mechanism to model long-distance interaction in the propagation structure. **BiGCN** (Bian et al., 2020) employs two GCNs to model the propagation graph and dispersion graph. **RumorGCN** (Hu et al., 2021) learns multi-relational dependencies from the propagation by using Relational GCNs. **EBGCN** (Wei et al., 2021), a graph-based model, focuses on the uncertainty issue in the propagation structure from a probability perspective.

5.4 Fake News Detection Results

The overall performance for fake news detection is reported in Table 2. From them, we we have the following key observations:

1) Text-based methods achieve inferior performance than propagation-based methods. It indicates that propagation patterns are more beneficial to detect fake news since fake news publishers always deliberately distort the text content of news. 2) PLAN captures long-range interactions in the propagation sequence with attention modules and obtains moderate results, even outperforming some shallow graph-based models. However, they still could not effectively distill latent interactions hidden in the propagation sequence and thus obtain limited performance. 3) EBGCN and RumorGCN achieve sub-optimal performance on PolitiFact and GossipCop, respectively. It makes sense as RumorGCN considers potential interactions from sibling nodes; while EBGCN explores robust interactions in an adjusted propagation tree, which can provide more effective structural information for detection. Nevertheless, their shallow networks make it hard to model long-distance interactions in the propagation, and thus they cannot be adaptive for news that has a deeper propagation structure. 4) Our UPSR yields consistently better performance than all the baselines on both datasets. The benefit mainly comes in two-fold. First, deep graph convolutions enable the model to focus on long-range interactions in the original propagation modeling. Second, UPSR further encodes the reconstructed propagation based on uncertainty-aware node representations, which can effectively capture more potential interactions between retweets and learn diverse structural patterns for detection.

Methods	PolitiFact		GossipCop	
	Acc	F1	Acc	F1
UPSR	**0.914**	**0.910**	**0.977**	**0.976**
- w/o Root	0.891	0.886	0.974	0.973
- w/o GPE	0.904	0.894	0.972	0.961
- w/o OPM	0.828	0.817	0.975	0.974
- w/o RPM	0.873	0.867	0.962	0.961
UPSR$_{\text{GCN}}$	0.891	0.886	0.972	0.971
UPSR$_{\text{GAT}}$	0.899	0.894	0.973	0.973
UPSR$_{\text{BiGCN}}$	0.886	0.880	0.974	0.973

Table 3: Results of ablation study and component analysis. The best result is in bold-face.

6 Discussion

In this section, we conduct more experiemtns to further understand the performance of UPSR.

6.1 Ablation Study

We conduct an ablation study to evaluate key components in UPSR. 1) **w/o Root** indicates that encod-

ing the reconstructed propagation graph does not explicitly consider the influence of source news. 2) **w/o GPE** removes Gaussian Propagation Estimation module and measures cosine-similarity between two node embedding. 3) **w/o OPM** refers to removing the original propagation modeling and directly reconstructing the propagation according to input textual features. 4) **w/o RPM** is removing the whole reconstructed propagation modeling.

The results of the ablation study are shown in the first block of Table 3. The full model yields the best performance in terms of accuracy and F1 score. 1) Without the consideration of source news influence in the reconstructed propagation modeling, the performance of **w/o Root** slightly reduces on both datasets, showing the vital role of source news in the propagation. 2) **w/o GPE** is obviously inferior to the full model, verifying that estimating propagation structure with multiple facets can successfully adapt to the uncertain effect of retweets and enable to derive accurate potential interactions. 3) When removing the complete reconstructed propagation modeling, **w/o RPM** obtains the inferior performance in terms of two evaluation metrics, which proves the effectiveness of the propagation reconstruction. 4) After removing the original propagation modeling, the performance of **w/o OPM** also drops significantly. This is intuitive since learning from explicit interactions between retweets in the original propagation could lead to relatively comprehensive representations, which enables GPE to explore more effective interactions.

6.2 Comparison with Different Original Propagation Modeling Modules

We further replace the deep graph convolutional network in the original propagation modeling with the following alternatives. 1) **UPSR**$_{\text{GCN}}$ adopts vanilla two-layer GCNs (Kipf and Welling, 2017) to model the original propagation. 2) **UPSR**$_{\text{GAT}}$ replaces with vanilla two-layer GATs (Velickovic et al., 2018). 3) **UPSR**$_{\text{BiGCN}}$ follows (Bian et al., 2020) to apply bi-directional GCNs .

The results are reported in the second block of Table 3. The degradation performance of these variants indicates the superiority of our model, which can capture long-range interactions in the propagation by stacking multiple graph convolutions. Besides, **UPSR** and its variants **UPSR**$_{\text{GCN}}$, **UPSR**$_{\text{GAT}}$, **UPSR**$_{\text{BiGCN}}$ consistently outperform the corresponding single graph models on both

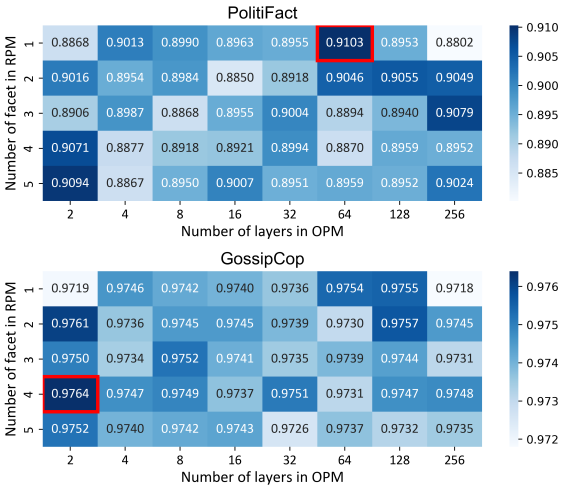

Figure 2: F1 scores against different hyperparameters.

datasets. The reason is that the dual graph framework can not only learn interactions in the original propagation but also capture potential interactions between uncertain tweets.

6.3 Parameter Analysis

Figure 2 explores the performance of UPSR against two vital parameters, *i.e.,* different numbers of layers in the original propagation modeling (OPM), and different numbers of facets in the reconstructed propagation modeling (RPM).

Effect of Graph Layers in Original Propagation Modeling. To investigate whether our model can benefit from the multi-layer propagation in the original propagation modeling, we vary the number of graph convolutional layers in the range of {2, 4, 8, 16, 32, 64, 128, 256}. The best setting is 64 and 2 on PolitiFact and GossipCop, respectively. Propagation structures are deeper on PolitiFact and thus more graph layers are needed to capture long-range interactions between nodes. The continual increase of the layer number even harms the performance. This might be caused by the overfitting issue.

Effect of Number of Facets in Reconstructed Propagation Modeling. To investigate whether our model can benefit from the multi-facet estimation for uncertainty, we vary the number of facets in the range of {1, 2, 3, 4, 5}. The optimal setting is 1 and 4 on PolitiFact and GossipCop datasets, respectively. These results indicate that estimating nodes from multiple facets is more profitable for detecting celebrity-related fake news, which can boost to capture latent interactions between two nodes sufficiently. Besides, dependencies between retweets under celebrity news may be more com-

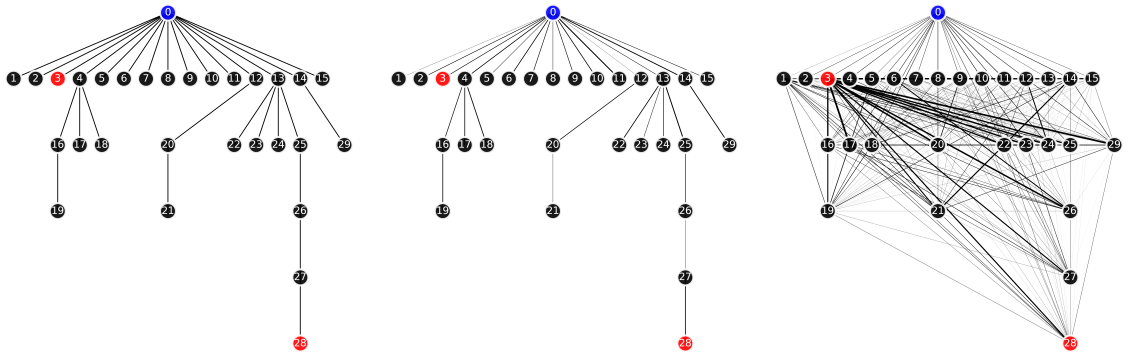

(a) Original Propagation (b) Adjusted by EBGCN (c) Reconstructed by UPSR

Figure 3: A case study of fake news on *PolitiFact*, which is missed by BiGCN and EBGCN but detected by UPSR. Node 0 refers to the source news and other nodes are its retweets. The breadth of the propagation is 15 and the depth of the propagation is 5. The edge width represents the weight of interactions.

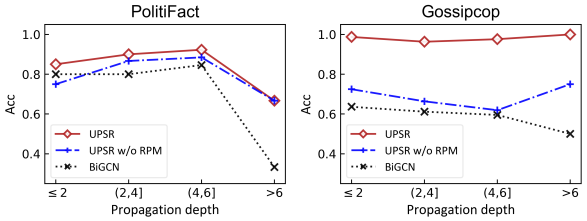

Figure 4: Performance on propagation structures with different depths. Y-axis refers to the accuracy score.

plex and more facets need to be considered.

6.4 Propagation Depth Analysis

Figure 4 shows performance on propagation structures with different depths. From the figure, the performance of BiGCN for detecting deeper propagation clearly decreases on both datasets. This reveals that fake news detection is more challenging with the deeper propagation which usually reflects vital potential interactions between users. Compared with BiGCN, UPSR and its variant obtain better performance in recognizing deeper propagation. This indicates that the original propagation modeling can effectively capture longer-range interactions in the original propagation for fake news detection. Moreover, UPSR achieves a considerable improvement over almost any range of propagation depth. We speculate, through estimating uncertain effects of retweets to reconstructing the original propagation, UPSR can further capture more potential interactions between two nodes and learn better representations for detection. Thus, UPSR is not sensitive to propagation depth and can be adaptive for both shallow and deep propagation.

6.5 Case Study

Figure 3 visualizes a propagation structure of a piece of *fake* news from PolitiFact. The news is misclassified by BiGCN and EBGCN but is detected by our model successfully.

Previous shallow graph networks (*e.g.,* BiGCN, EBGCN) would ignore the distant connections such as the interaction between node 3 and 28 and can only capture local structural propagation information. Through reconstructing the original propagation, UPSR alleviates this issue to some extent and aggregates more effective information in the graph via reconstructed edges between two distant nodes. Besides, compared with Figure 3(b) and 3(c), EBGCN dealt with noisy edges by adaptively adjusting weights of explicit edges. However, they solely focus on explicit edges and limit the message-passing in the graph. Different from their model, UPSR not only is robust to these noisy edges but also captures more valuable potential interactions between nodes to improve detection.

7 Conclusion

This paper has studied a broader propagation uncertainty issue in fake news detection. We propose a novel Uncertainty-aware Propagation Structure Reconstruction (UPSR) to jointly model long-range and potential interactions in the uncertain propagation. Gaussian Propagation Estimation (GPE) is developed to reconstruct latent propagation by adapting the inherent uncertain effect of retweets in the propagation. Experiments conducted on two real-world benchmarks have shown that UPSR outperforms recent detection methods.

In the future, we will focus on improving the detection performance of our model in scenarios where training propagation data is limited.

Acknowledgments

We thank our anonymous reviewers for their helpful comments. This work was supported by the National Natural Science Foundation of China under Grant No.6210071416.

References

Wasim Ahmed, Josep Vidal-Alaball, Joseph Downing, Francesc López Seguí, et al. 2020. Covid-19 and the 5g conspiracy theory: social network analysis of twitter data. *Journal of medical internet research*, 22(5):e19458.

Tian Bian, Xi Xiao, Tingyang Xu, Peilin Zhao, Wenbing Huang, Yu Rong, and Junzhou Huang. 2020. Rumor detection on social media with bi-directional graph convolutional networks. In *AAAI*, pages 549–556.

Carlos Castillo, Marcelo Mendoza, and Barbara Poblete. 2011. Information credibility on twitter. In *WWW*, pages 675–684. ACM.

Ming Chen, Zhewei Wei, Zengfeng Huang, Bolin Ding, and Yaliang Li. 2020. Simple and deep graph convolutional networks. In *ICML*, volume 119 of *Proceedings of Machine Learning Research*, pages 1725–1735. PMLR.

Yingtong Dou, Kai Shu, Congying Xia, Philip S. Yu, and Lichao Sun. 2021. User preference-aware fake news detection. In *SIGIR*, pages 2051–2055. ACM.

Matthew Honnibal and Ines Montani. 2017. spacy 2: Natural language understanding with bloom embeddings, convolutional neural networks and incremental parsing. *To appear*, 7(1).

Dou Hu, Xiaolong Hou, Lingwei Wei, Lian-Xin Jiang, and Yang Mo. 2022. MM-DFN: multimodal dynamic fusion network for emotion recognition in conversations. In *ICASSP*, pages 7037–7041. IEEE.

Dou Hu, Lingwei Wei, Wei Zhou, Xiaoyong Huai, Jizhong Han, and Songlin Hu. 2021. A rumor detection approach based on multi-relational propagation tree. *Journal of Computer Research and Development*, 58(7):1395–1411.

S. Mo Jang, Tieming Geng, Jo-Yun Queenie Li, Ruofan Xia, Chin-Tser Huang, Hwalbin Kim, and Jijun Tang. 2018. A computational approach for examining the roots and spreading patterns of fake news: Evolution tree analysis. *Comput. Hum. Behav.*, 84:103–113.

Shengyi Jiang, Xiaoting Chen, Liming Zhang, Sutong Chen, and Haonan Liu. 2019. User-characteristic enhanced model for fake news detection in social media. In *NLPCC (1)*, volume 11838 of *Lecture Notes in Computer Science*, pages 634–646. Springer.

Hamid Karimi and Jiliang Tang. 2019. Learning hierarchical discourse-level structure for fake news detection. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 3432–3442. Association for Computational Linguistics.

Ling Min Serena Khoo, Hai Leong Chieu, Zhong Qian, and Jing Jiang. 2020. Interpretable rumor detection in microblogs by attending to user interactions. In *AAAI*, pages 8783–8790. AAAI Press.

Diederik P. Kingma and Jimmy Ba. 2015. Adam: A method for stochastic optimization. In *ICLR*.

Thomas N. Kipf and Max Welling. 2017. Semi-supervised classification with graph convolutional networks. In *ICLR (Poster)*. OpenReview.net.

Sumeet Kumar and Kathleen M. Carley. 2019. Tree lstms with convolution units to predict stance and rumor veracity in social media conversations. In *ACL (1)*, pages 5047–5058.

Hongzhan Lin, Jing Ma, Mingfei Cheng, Zhiwei Yang, Liangliang Chen, and Guang Chen. 2021. Rumor detection on twitter with claim-guided hierarchical graph attention networks. In *EMNLP (1)*, pages 10035–10047. Association for Computational Linguistics.

Jing Ma and Wei Gao. 2020. Debunking rumors on twitter with tree transformer. In *COLING*, pages 5455–5466. International Committee on Computational Linguistics.

Jing Ma, Wei Gao, Prasenjit Mitra, Sejeong Kwon, Bernard J. Jansen, Kam-Fai Wong, and Meeyoung Cha. 2016. Detecting rumors from microblogs with recurrent neural networks. In *IJCAI*, pages 3818–3824. IJCAI/AAAI Press.

Jing Ma, Wei Gao, and Kam-Fai Wong. 2018. Rumor detection on twitter with tree-structured recursive neural networks. In *ACL (1)*, pages 1980–1989. Association for Computational Linguistics.

Rada Mihalcea and Carlo Strapparava. 2009. The lie detector: Explorations in the automatic recognition of deceptive language. In *ACL/IJCNLP (2)*, pages 309–312. The Association for Computer Linguistics.

Tomás Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013. Efficient estimation of word representations in vector space. In *ICLR (Workshop Poster)*.

Rahul Mishra. 2020. Fake news detection using higher-order user to user mutual-attention progression in propagation paths. In *CVPR Workshops*, pages 2775–2783. Computer Vision Foundation / IEEE.

Federico Monti, Fabrizio Frasca, Davide Eynard, Damon Mannion, and Michael M. Bronstein. 2019. Fake news detection on social media using geometric deep learning. In *ICLR (Workshop)*.

Van-Hoang Nguyen, Kazunari Sugiyama, Preslav Nakov, and Min-Yen Kan. 2020. FANG: leveraging social context for fake news detection using graph representation. In *CIKM*, pages 1165–1174. ACM.

Kashyap Popat. 2017. Assessing the credibility of claims on the web. In *WWW (Companion Volume)*, pages 735–739. ACM.

Martin Potthast, Johannes Kiesel, Kevin Reinartz, Janek Bevendorff, and Benno Stein. 2018. A stylometric inquiry into hyperpartisan and fake news. In *ACL (1)*, pages 231–240. Association for Computational Linguistics.

Natali Ruchansky, Sungyong Seo, and Yan Liu. 2017. CSI: A hybrid deep model for fake news detection. In *CIKM*, pages 797–806. ACM.

Chengcheng Shao, Giovanni Luca Ciampaglia, Onur Varol, Kai-Cheng Yang, Alessandro Flammini, and Filippo Menczer. 2018. The spread of low-credibility content by social bots. *Nature communications*, 9(1):1–9.

Kai Shu, Limeng Cui, Suhang Wang, Dongwon Lee, and Huan Liu. 2019a. defend: Explainable fake news detection. In *KDD*, pages 395–405. ACM.

Kai Shu, Deepak Mahudeswaran, Suhang Wang, Dongwon Lee, and Huan Liu. 2020. Fakenewsnet: A data repository with news content, social context, and spatiotemporal information for studying fake news on social media. *Big Data*, 8(3):171–188.

Kai Shu, Xinyi Zhou, Suhang Wang, Reza Zafarani, and Huan Liu. 2019b. The role of user profiles for fake news detection. In *ASONAM*, pages 436–439. ACM.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *NIPS*, pages 5998–6008.

Petar Velickovic, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Liò, and Yoshua Bengio. 2018. Graph attention networks. In *ICLR (Poster)*. OpenReview.net.

Soroush Vosoughi, Deb Roy, and Sinan Aral. 2018. The spread of true and false news online. *Science*, 359(6380):1146–1151.

Lingwei Wei, Dou Hu, Wei Zhou, Xin Wang, and Songlin Hu. 2022. Modeling the uncertainty of information propagation for rumor detection: A neuro-fuzzy approach. *IEEE Transactions on Neural Networks and Learning Systems*.

Lingwei Wei, Dou Hu, Wei Zhou, Zhaojuan Yue, and Songlin Hu. 2021. Towards propagation uncertainty: Edge-enhanced bayesian graph convolutional networks for rumor detection. In *ACL/IJCNLP (1)*, pages 3845–3854. Association for Computational Linguistics.

Yao Yuan, Lorenzo Rosasco, and Andrea Caponnetto. 2007. On early stopping in gradient descent learning. *Constructive Approximation*, 26(2):289 – 315.

Xinyi Zhou and Reza Zafarani. 2020. A survey of fake news: Fundamental theories, detection methods, and opportunities. *ACM Comput. Surv.*, 53(5):109:1–109:40.

A Unified Propagation Forest-based Framework for Fake News Detection

Lingwei Wei[1,2] and **Dou Hu**[1,2] and **Yantong Lai**[1,2] and **Wei Zhou**[1*] and **Songlin Hu**[1,2]
[1] Institute of Information Engineering, Chinese Academy of Sciences
[2] School of Cyber Security, University of Chinese Academy of Sciences
{weilingwei, hudou, laiyantong, zhouwei, husonglin}@iie.ac.cn

Abstract

Fake news's quick propagation on social media brings severe social ramifications and economic damage. Previous fake news detection usually learn semantic and structural patterns within a single target propagation tree. However, they are usually limited in narrow signals since they do not consider latent information cross other propagation trees. Motivated by a common phenomenon that most fake news is published around a specific hot event/topic, this paper develops a new concept of propagation forest to naturally combine propagation trees in a semantic-aware clustering. We propose a novel Unified Propagation Forest-based framework (UniPF) to fully explore latent correlations between propagation trees to improve fake news detection. Besides, we design a root-induced training strategy, which encourages representations of propagation trees to be closer to their prototypical root nodes. Extensive experiments on four benchmarks consistently suggest the effectiveness and scalability of UniPF.

1 Introduction

Recently, social media platforms have facilitated information dissemination greatly. Nevertheless, they quicken the proliferation of fake news as well due to the lack of authoritative regulators (Zhou et al., 2021). Its extensive dissemination would trigger great panic in society and severely impair the public and individuals (Difonzo and Bordia, 2007; Jin et al., 2017; Jankowski et al., 2020). To keep social media a healthy environment, it is desirable and socially beneficial to detect fake news.

Among previous works on automatic fake news detection, textual news material is utilized by almost all extant studies on fake news detection (Ma et al., 2016; Ruchansky et al., 2017; Popat, 2017; Potthast et al., 2018; Zhou et al., 2020). However, because fake news is purposefully designed to deceive readers by imitating actual news, detecting and distinguishing them solely from news material is challenging (Afroz et al., 2012; Shu et al., 2020a). Hence, an increasing number of works (Ma et al., 2016; Bian et al., 2020; Hu et al., 2021; Song et al., 2021; Wei et al., 2021) have been devoted to learning potential propagation patterns by investigating relationships among tweets for each news article.

However, most approaches usually consider that propagation trees are independent and ignore latent correlations across different propagation trees, which are supportive for identifying fake news for two-fold reason. 1) Most fake news is usually published deliberately around a specific hot event / topic and then widely disseminated in reality (Friggeri et al., 2014; Nourbakhsh et al., 2015). Hence, potential semantical connections may exist across them. 2) Some spreaders were social bots that are manipulated by a malicious group (Shu et al., 2020a). These deliberate and organized behaviors during propagation may lead to similar structural patterns across two propagation trees. We believe that capturing these vital **semantic** and **structural** characteristics across propagation trees is beneficial to understanding the target propagation tree, accordingly to make more accurate detection.

Yuan et al. (2019) built a global user-tweet heterogeneous graph according to similar participants but ignored semantic relations. Huang et al. (2020) introduced word nodes and constructed a user-tweet-word graph to capture fine-grained semantic relations between source news. However, these fine-grained (e.g., word-level) correlations between news contents may compensate for semantic information to some extent since some fine-grained words may suffer from the polysemy problem (Neelakantan et al., 2014). Given two examples, the real news is *Donald Trump: ... Senate I believe really wants to get something done because Obamacare is dead,...Obamacare is absolutely dead.*; the fake news is *Donald Trump was pronounced dead this morning following what some are describing as a violent heart attack....*

* Corresponding author.

Proceedings of the 29th International Conference on Computational Linguistics, pages 2769–2779
October 12–17, 2022.

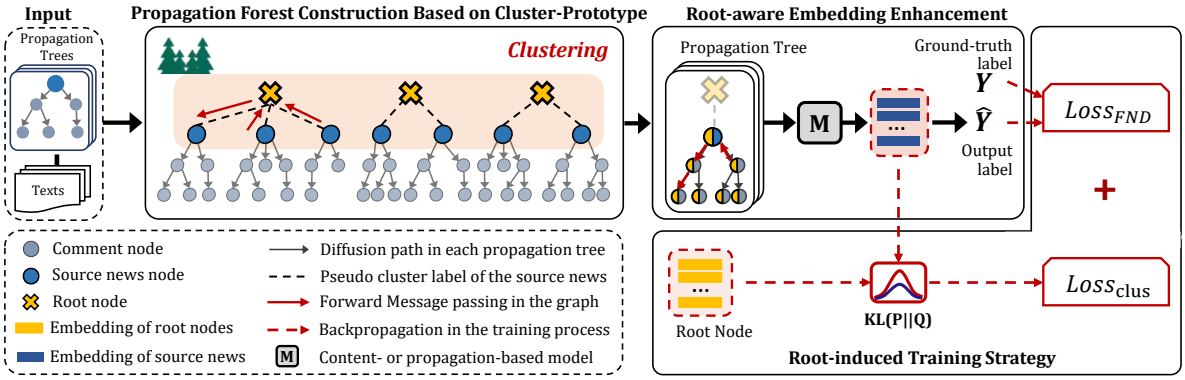

Figure 1: Overall architecture of Unified Propagation Forest-based (UniPF) framework.

Both *Donald Trump* and *dead* were mentioned in both news, but sentence-level semantics are obviously different. The former news shows Donald Trump saying that Obamacare is abolished. But the latter falsely reported Donald Trump was pronounced dead. Thus, there still lacks a unified coarse-grained paradigm that considers effective semantic and structural correlations cross propagation trees simultaneously for detection.

In this paper, we develop a new concept of *Propagation Forest* that combines all propagation trees to explore latent semantic and structural correlations between propagation trees. Under the propagation forest, we propose a general **Uni**fied **P**ropagation **F**orest-based framework (***UniPF***) to enhance target sample's embedding by exploring latent semantic and structural correlations between propagation trees. Root nodes in the propagation forest are generated according to prototypes (Snell et al., 2017) in a clustering manner. A prototype is a typical embedding for a cluster of related propagation trees. These root nodes can be interpreted as coarse-grained topics or events on social media in reality. Through aggregating neighbor features in the propagation forest, potential patterns implied in similar sub-propagation trees can be captured by performing a high-level transformation of the propagation forest. Besides, a new root-induced training strategy is designed to enhance the quality of the propagation forest by modeling the consistency of generated prototypical root nodes and observed propagation trees. UniPF is of great scalability and can easily be extended in any propagation-based or content-based approaches. We conduct experiments on four benchmarks to assess our framework. The results consistently suggest that UniPF can successfully strengthen fake news detection performance whilst being scalable.

The following are main contributions: 1) We develop a novel concept of *Propagation Forest* that bridges all propagation trees in a semantic-clustering manner. 2) We propose a general UniPF, to enhance fake news detection by fully exploring latent both semantic and structural correlations between similar propagation trees in the generated propagation forest. 3) We devise a root-induced training strategy to guarantee high-quality of prototypical roots in propagation forest. 4) Experiments on four benchmarks consistently demonstrate the scalability and effectiveness of UniPF.

2 Problem Statement

Let $\mathcal{D} = \{(x_i, G_i^{\text{tree}})\}_{i=1}^{|\mathcal{D}|}$ be fake news detection dataset with $|\mathcal{D}|$ samples. Each sample includes a specific source news x_i and its unique propagation tree G_i^{tree}. Text-based fake news detection techniques mainly use x; while propagation-based techniques use x and G^{tree}. $G_i^{\text{tree}} = (V_i^{\text{tree}}, E_i^{\text{tree}})$ refers to the corresponding propagation tree of i-th source news. $V_i^{\text{tree}} = \{x_i\} \cup \mathcal{C}_i$ is a set of nodes representing the source news x_i and comments $\mathcal{C}_i$. E_i^{tree} refers to a set of directed edges based on anonymous propagation behaviors, e.g., retweet or comment. Define the embedding of the source news x_i as $\mathbf{x}_i \in \mathbb{R}^{d_0}$, and that of a comment $c_{ij} \in \mathcal{C}_i$ as $\mathbf{c}_{ij} \in \mathbb{R}^{d_0}$, where d_0 is the dimensionality of textual features. Each sample is annotated with a ground-truth label $y_i \in \{0, 1\}$. We formulate the fake news detection problem as a binary classification problem, i.e., each sample can be real ($y_i = 0$) or fake ($y_i = 1$), and learn a classifier f from the labeled set, *i.e.*, $f : \mathcal{D} \to \mathcal{Y}$.

3 Methodology

This section offers Unified Propagation Forest-based (UniPF) framework to boost fake news detec-

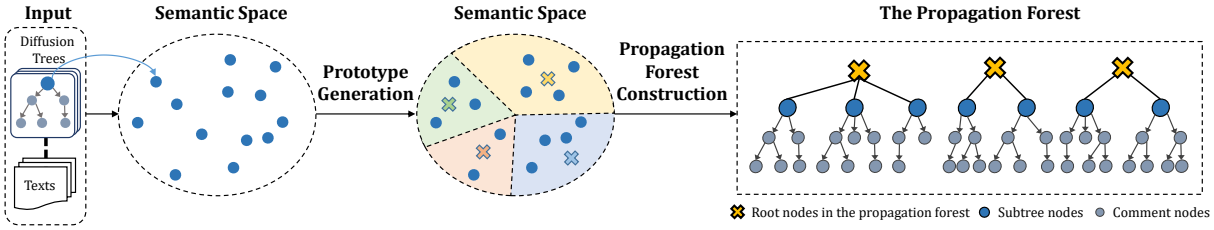

Figure 2: The workflow of constructing a propagation forest based on the input.

tion by examining latent correlations across propagation trees in a semantic-clustering manner. Figure 1 depicts an overview of UniPF framework. It consists of three key components, propagation forest construction, prototype-aware embedding enhancement, and root-induced training strategy.

3.1 Propagation Forest Construction Based on Cluster-Prototype

Given training samples, we perform semantic-aware prototype clustering to generate root nodes, which are *ancestors* for propagation subtrees and are representative for propagation trees with a similar structure as well as semantics. According to pseudo labels of clustering, a propagation forest is developed to combine all propagation trees.

3.1.1 Prototype Generation

Since most fake news is published around a specific event or hot topic and is widely disseminated. it is intuitive to find root nodes via clustering. The goal of this module is to group the entire propagation trees to generate prototypical root nodes of the propagation forest in a clustering manner. Inspired by K-Means (Arthur and Vassilvitskii, 2007), we perform semantic clustering to find several prototypes. The number of clusters K is the only parameter required by the algorithm.

The workflow of constructing a propagation forest based on the input is shown in Figure 2. Specifically, at the preliminary stage, each source news of propagation tree is projected into a unified semantic space. Then, we exploit the clustering tool to process the semantic representations of samples and obtain their pseudo clustering labels to build the propagation forest.

Given the textual embedding $\mathbf{X} = [\mathbf{x}_1, ..., \mathbf{x}_{|\mathcal{D}|}]$ of samples, we find K prototypes and assign propagation trees to a prototype so as to minimize the potential function ϕ. And K prototypes can be denoted as $\{\mathbf{m}_1, \mathbf{m}_2, ..., \mathbf{m}_K\}$.

$$\phi = \frac{\sum_{z=1}^{Z} \|\mathbf{x}_z - \mathbf{m}\|^2}{Z \log(Z + \beta)}, \tag{1}$$

where $\mathbf{m} \in \mathbb{R}^{d_0}$ refers to a prototype. β is a smooth variable that prohibits minor clusters from receiving an excessively high ϕ. Z is the number of data points covered by each prototype. We minimize the function in two distinct principles: 1) summation of squared distance between a target propagation tree and a cluster's nucleus (i.e., prototype) is minimal; 2) each cluster covers more key-points, namely a higher value of Z. Within clusters, the less variance there is, the more uniform the data points are.

3.1.2 Propagation Forest Construction

Based on the above generated prototypes and propagation trees, we build a propagation forest to combine propagation trees. The propagation forest is formulated as an undirected graph structure, denoted as $G^{\text{Forest}} = (V^{\text{Forest}}, E^{\text{Forest}})$, where V^{Forest} indicates a node set in G^{Forest} and E^{Forest} refers to a undirected edge set in G^{Forest}.

Nodes. There are three node types in propagation forest graph. $V^{\text{Forest}} = V^S \cup V^R \cup V^C$, where 1) source news nodes represent the source news of a propagation tree, denoted as $V^S = \{x_1, ..., x_N\}$; 2) root nodes represent prototypes, which can be interpreted as latent topics or events on social media, denoted as $V^R = \{m_1, ..., m_K\}$; 3) comment nodes refer to subsequent retweets given the source news, $V^C = \{\mathcal{C}_i\}_{i=1}^{N}$, where $\mathcal{C}_i$ is the set of comment nodes of the i-th propagation tree.

Edges. There are two types of edges in the propagation forest graph. 1) For connections between source news nodes and root nodes, we define undirected edges based on pseudo cluster labels, i.e., each propagation tree is connected with the subordinate cluster. The edge weights are defined as the probability that the source news is assigned to the root node. 2) For connections between source

news nodes and comment nodes, and connections between two comment nodes, we define directed edges based on retweet relations. For the i-th propagation tree, if the j-th comment c_{ij} retweeted the source news x_i, there will be an directed edge $x_i \rightarrow c_{ij}$; if the q-th comment c_{iq} retweeted the j-th comment c_{ij}, there will be an directed edge $c_{ij} \rightarrow c_{iq}$. The edge weights are set to 1 if there is a directed edge between two nodes.

3.2 Root-aware Embedding Enhancement

Based on the propagation forest, we perform message passing between root nodes and source news nodes with graph-based transformations to explore latent correlations cross similar propagation trees. Then, we enhance the original message passing in the target propagation tree to boost the understanding of information propagation with the shared semantics and structure from other propagation trees.

3.2.1 Modeling Latent Correlations in the Propagation Forest

We transform the propagation forest using a differentiable message passing method to explore rich correlations between propagation trees. Motivated by graph convolutional networks (Kipf and Welling, 2017), $\mathbf{v}_i^{(1)}$ was calculated for source news node v_i in the first layer by aggregating neighborhood information (i.e., neighbors indicated by the subordinate prototype root nodes) using the transformation,

$$\mathbf{v}_i^{(1)} = \sigma\Big(\sum_{j \in \mathcal{N}_i} a_{ij}\mathbf{W}^{(1)}\mathbf{v}_j^{(0)} + a_{ii}\mathbf{W}_0^{(1)}\mathbf{v}_i^{(0)}\Big), \quad \text{for } i = 1, ..., N+K, \tag{2}$$

where $\mathbf{v}^{(0)}$ is initialized with $\mathbf{m}$ for root nodes and $\mathbf{x}$ for other nodes. $\mathbf{W}^{(1)}$ and $\mathbf{W}_0^{(1)}$ are trainable parameters. σ is ReLU activation function. $\mathcal{N}_i$ denotes neighbouring indices of node v_i.

Based on the output, another neighborhood-based transformation is applied on source news nodes to integrate shared features from the root nodes. The computations can be defined as,

$$\mathbf{v}_i^{(2)} = \sigma\Big(\sum_{j \in \mathcal{N}_i} a_{ij}\mathbf{W}^{(2)}\mathbf{v}_j^{(1)} + a_{ii}\mathbf{W}_0^{(2)}\mathbf{v}_i^{(1)}\Big), \quad \text{for } i = 1, ..., N+K, \tag{3}$$

where $\mathbf{W}^{(2)}$ and $\mathbf{W}_0^{(2)}$ are learnable parameters. This stack of transformations effectively accumulates a normalized sum of information from similar propagation trees in the propagation forest.

In this way, latent correlations can be captured by extracting and aggregating effective information from the node's neighbors. These root nodes can not only transmit semantic and structural features of adjacent samples, but also further integrate features of similar clusters.

3.2.2 Improving Fake News Detection

Next, we exploit the embedding of source news $\mathbf{v}_i^{(2)}$ learned by the above network to enhance the embedding in the target propagation tree. To highlight the impact of the source news during propagation, we make a concatenation and apply a fully connected layer to compute the enhanced embedding of source news $\mathbf{x}_i'$ and comments $\mathbf{C}_i' = [\mathbf{c}_{i1}', \mathbf{c}_{i2}', ..., \mathbf{c}_{iN}']$.

$$\begin{aligned} \mathbf{x}_i' &= \mathbf{W}_e[\mathbf{x}_i; \mathbf{v}_i^{(2)}] + \mathbf{b}_e, \\ \mathbf{c}_{ij}' &= \mathbf{W}_e[\mathbf{c}_{ij}; \mathbf{v}_i^{(2)}] + \mathbf{b}_e, \end{aligned} \tag{4}$$

where $\mathbf{W}_e$ denotes the transform matrix and $\mathbf{b}_e$ denotes the bias term. The representation of the i-th propagation tree can be further encoded by the existing fake news detection models.

$$\mathbf{u}_i = \text{Model}(\mathbf{x}_i', \mathbf{C}_i', G_i^{\text{tree}}), \tag{5}$$

where $\text{Model}(\cdot)$ refers to a base detection model, given textual content and its propagation tree. Then, the label probability $\hat{\mathbf{y}}$ is computed as:

$$\hat{\mathbf{y}}_i = \text{Softmax}(\mathbf{W}_u\mathbf{u}_i + \mathbf{b}_u), \tag{6}$$

where $\mathbf{W}_u$ and $\mathbf{b}_u$ are learnable parameter matrices.

3.3 Root-induced Training Strategy

To guarantee the quality of generated root hubs, we design the root-induced training strategy to constraint the consistency of the representation of the target propagation tree to its prototypical root node in the propagation forest. The strategy can take the "confident" clustering assignments as soft labels, and be assist to guide the optimizing procedure.

3.3.1 Supervised Classification Loss

We minimize the fake news classification loss calculated by the Cross-entropy criterion. That is,

$$\mathcal{L}_{\text{FND}} = -\mathbf{y}\log(\hat{\mathbf{y}}) - (1-\mathbf{y})\log(1-\hat{\mathbf{y}}), \tag{7}$$

where $\mathbf{y}$ indicates the ground-truth label and $\hat{\mathbf{y}}$ represents the prediction label.

3.3.2 Unsupervised Consistency Loss

To improve the embedding of prototypes, inspired by Xie et al. (2016), we examine the similarity between propagation tree representation $\mathbf{u}$ and prototype $\mathbf{m}$ via a Student's t-distribution.

$$\mathbf{q}_{ik} = \frac{(1 + \|\mathbf{u}_i - \mathbf{m}_k\|^2/\alpha)^{-\frac{\alpha+1}{2}}}{\sum_k^K (1 + \|\mathbf{u}_i - \mathbf{m}_k\|^2/\alpha)^{-\frac{\alpha+1}{2}}}, \quad (8)$$

where α are degrees of freedom of Student's t-distribution and $\mathbf{q}_{ik}$ can be interpreted as the probability of assigning the i-th sample to prototype m_k, which is a soft assignment. This option can accommodate a variety of scaled clusters while still being computationally efficient.

Then, leveraging propagation tree nodes, we progressively update root nodes by gaining knowledge from high confidence predictions. As a consequence, we outline our goal as a Kullback-Leibler divergence between smooth assignments $\mathbf{q}$ and accessory distribution $\mathbf{p}$:

$$\mathcal{L}_{\text{Clus}} = \text{KL}(\mathbf{P}\|\mathbf{Q}) = \sum_i \sum_k \mathbf{p}_{ik} \log \frac{\mathbf{p}_{ik}}{\mathbf{q}_{ik}}. \quad (9)$$

Considering $\mathbf{q}$ are smooth assignments, using softer probabilistic objectives $\mathbf{p}$ appears more natural and adaptable. The auxiliary distribution $\mathbf{p}$ possesses have three attributes: 1) improve the purity of center clusters; 2) concentrate on highly relevant propagation trees; 3) standardize contribution of each centroid to reduce potential negative risk that larger-scale clusters may obfuscate latent feature space. By increasing $\mathbf{q}_i$ to the second power and then standardizing by cluster size, $\mathbf{p}_i$ is derived:

$$\mathbf{p}_{ik} = \frac{\mathbf{q}_{ik}^2/\sum_i \mathbf{q}_{ik}}{\sum_{k'} \mathbf{q}_{ik'}^2/\sum_i \mathbf{q}_{ik'}}. \quad (10)$$

3.3.3 Joint Training Procedure

To recap, we optimize UniPF framework during training through reducing supervised cross-entropy objective of labeled data $\mathcal{L}_{\text{FND}}$ and unsupervised consistency objective of unlabeled root nodes $\mathcal{L}_{\text{Clus}}$,

$$\Theta^* = \arg\min_{\Theta} \mathcal{L}_{\text{FND}} + \lambda \mathcal{L}_{\text{Clus}}, \quad (11)$$

where λ is a trade-off hyper-parameter. Θ is all trainable parameters of the model.

4 Experiments

In this section, we experimentally evaluate the performance of our proposed UniPF.

Dataset	Total	# Real News	# Fake News
PolitiFact	314	157	157
GossipCop	5,464	2,732	2,732
Twitter15	712	372	370
Twitter16	410	205	205

Table 1: The statistics of four public datasets.

4.1 Experimental Setup

Datasets We conduct experiments on four public benchmarks. The dataset statistics are shown in Table 1. **PolitiFact** and **GossipCop** are released by FakeNewsNet (Shu et al., 2020a). Samples are collected from *PolitiFact*[1] and *GossipCop*[2], which are two websites for fact-checking political and celebrity news, respectively. Each sample contains text content of source news and comments, and diffusion relations between anonymous posts. The ground truth label is obtained according to journalists and domain experts. We follow Shu et al. (2019a) and randomly select 75% of news as training data and the remaining as test data. **Twitter15** and **Twitter16**, released by Ma et al. (2017), contain a collection of source news on *Twitter*[3] in 2015 (Liu et al., 2015) and 2016 (Ma et al., 2016), respectively. Each sample contains text content of source news and comments, and diffusion relations between comments. Each sample is tagged as non-rumor, false rumor, true rumor, or unverified rumor based on veracity tags in rumor debunking websites. Following Lu and Li (2020), we choose only "true" and "fake" labels as the ground truth. Following Bian et al. (2020); Wei et al. (2021), we conduct 5-fold cross validation for the two datasets.

Comparison Methods *Text*-based fake news detection methods include: **mGRU** (Ma et al., 2016) uses an recurrent neural network to capture sequential features from user comments. **CSI** (Ruchansky et al., 2017) learns the sequential retweet features by employing an LSTM network. *Propagation*-based fake news detection methods include: **GCNFN** (Monti et al., 2019) models the propagation tree as a graph and uses GCN to encode propagation graphs. **BiGCN** (Bian et al., 2020) employs two distinct GCNs to model propagation directed graph and dispersion directed graph, respectively. **RumorGCN** (Hu et al., 2021) learns multi-relational dependencies from the propagation

[1] https://www.politifact.com/
[2] https://www.gossipcop.com/
[3] https://twitter.com/

	Method	PolitiFact				GossipCop			
		Acc	P	R	F1	Acc	P	R	F1
Text-based	mGRU	0.754	0.800	0.666	0.744	0.859	0.845	0.881	0.859
	UniPF-mGRU	**0.772**$^{\uparrow}$	**0.846**$^{\uparrow}$	**0.733**$^{\uparrow}$	**0.771**$^{\uparrow}$	**0.905**$^{\uparrow}$	**0.894**$^{\uparrow}$	**0.926**$^{\uparrow}$	**0.905**$^{\uparrow}$
	CSI	0.734	0.672	0.550	0.688	0.866	0.892	0.840	0.866
	UniPF-CSI	**0.760**$^{\uparrow}$	**0.783**$^{\uparrow}$	**0.800**$^{\uparrow}$	**0.754**$^{\uparrow}$	**0.933**$^{\uparrow}$	**0.932**$^{\uparrow}$	**0.933**$^{\uparrow}$	**0.932**$^{\uparrow}$
Propagation-based	GCNFN	0.856	0.862	0.851	0.849	0.886	0.892	0.881	0.883
	UniPF-GCNFN	**0.886**$^{\uparrow}$	**0.902**$^{\uparrow}$	**0.867**	**0.882**$^{\uparrow}$	**0.958**$^{\uparrow}$	**0.963**$^{\uparrow}$	**0.954**$^{\uparrow}$	**0.958**$^{\uparrow}$
	BiGCN	0.861	0.865	0.877	0.853	0.959	0.959	0.959	0.958
	UniPF-BiGCN	**0.906**$^{\uparrow}$	**0.901**$^{\uparrow}$	**0.906**$^{\uparrow}$	**0.904**$^{\uparrow}$	**0.968**$^{\uparrow}$	**0.968**$^{\uparrow}$	**0.969**$^{\uparrow}$	**0.968**$^{\uparrow}$
	RumorGCN	0.891	0.901	0.875	0.888	0.968	0.965	0.971	0.968
	UniPF-RumorGCN	**0.899**$^{\uparrow}$	**0.911**$^{\uparrow}$	**0.881**$^{\uparrow}$	**0.895**$^{\uparrow}$	**0.968**	**0.986**$^{\uparrow}$	**0.952**	**0.969**
	EBGCN	0.896	0.898	0.909	0.891	0.964	0.966	0.962	0.963
	UniPF-EBGCN	**0.911**$^{\uparrow}$	**0.912**$^{\uparrow}$	**0.904**	**0.909**$^{\uparrow}$	**0.966**$^{\uparrow}$	**0.965**	**0.973**$^{\uparrow}$	**0.966**$^{\uparrow}$

Table 2: Model performance for fake news detection on *PolitiFact* and *GossipCop* datasets. "↑" marks superior results compared to the corresponding base model, which are significant at level $p < 0.05$ based on t-test.

	Method	Twitter15				Twitter16			
		Acc	P	R	F1	Acc	P	R	F1
Text-based	mGRU	0.881	0.888	0.860	0.880	0.893	0.906	0.874	0.893
	UniPF-mGRU	**0.923**$^{\uparrow}$	**0.931**$^{\uparrow}$	**0.912**$^{\uparrow}$	**0.923**$^{\uparrow}$	**0.928**$^{\uparrow}$	**0.925**$^{\uparrow}$	**0.928**$^{\uparrow}$	**0.927**$^{\uparrow}$
	CSI	0.911	0.926	0.900	0.919	0.914	0.912	0.886	0.912
	UniPF-CSI	**0.931**$^{\uparrow}$	**0.939**$^{\uparrow}$	**0.921**$^{\uparrow}$	**0.929**$^{\uparrow}$	**0.927**$^{\uparrow}$	**0.954**$^{\uparrow}$	**0.911**$^{\uparrow}$	**0.926**$^{\uparrow}$
Propagation-based	GCNFN	0.927	0.920	0.936	0.927	0.926	0.927	0.927	0.926
	UniPF-GCNFN	**0.938**$^{\uparrow}$	**0.956**$^{\uparrow}$	**0.917**	**0.938**$^{\uparrow}$	**0.938**$^{\uparrow}$	**0.929**$^{\uparrow}$	**0.951**$^{\uparrow}$	**0.938**$^{\uparrow}$
	BiGCN	0.942	0.950	0.934	0.942	0.936	0.937	0.936	0.936
	UniPF-BiGCN	**0.952**$^{\uparrow}$	**0.959**$^{\uparrow}$	**0.945**$^{\uparrow}$	**0.952**$^{\uparrow}$	**0.955**$^{\uparrow}$	**0.962**$^{\uparrow}$	**0.951**$^{\uparrow}$	**0.955**$^{\uparrow}$
	RumorGCN	0.952	0.951	0.948	0.952	0.945	0.944	0.944	0.945
	UniPF-RumorGCN	**0.959**$^{\uparrow}$	**0.936**	**0.987**$^{\uparrow}$	**0.959**$^{\uparrow}$	**0.956**$^{\uparrow}$	**0.956**$^{\uparrow}$	**0.948**$^{\uparrow}$	**0.952**$^{\uparrow}$
	EBGCN	0.949	0.947	0.945	0.949	0.943	0.953	0.937	0.944
	UniPF-EBGCN	**0.955**$^{\uparrow}$	**0.971**$^{\uparrow}$	**0.941**	**0.955**$^{\uparrow}$	**0.963**$^{\uparrow}$	**0.951**$^{\uparrow}$	**0.976**$^{\uparrow}$	**0.963**$^{\uparrow}$

Table 3: Model performance for fake news detection on *Twitter15* and *Twitter16* datasets. "↑" marks superior results compared to the corresponding base model, which are significant at level $p < 0.05$ based on t-test.

tree by using relational GCNs. **EBGCN** (Wei et al., 2021), a graph-based model to learn propagation uncertainty in a probability manner, encodes propagation tree with edge-enhanced Bayesian networks.

Evaluation Metrics Since fake news detection is formulated as a classification problem, we evaluate models with four commonly used metrics: Accuracy (Acc), Precision (P), Recall (R), and F1 scores.

Implementation All experiments are conducted on a single Ge-Force RTX 3080Ti. For *PolitiFact* and *GossipCop*, we follow Dou et al. (2021) and consider 300-dimensional word2vec vectors (Mikolov et al., 2013) to initialize word embeddings, which are pretrained on a large corpus with 680k words by spaCy (Honnibal and Montani, 2017). For *Twitter15* and *Twitter16*, following Ma et al. (2018); Bian et al. (2020), we consider TF-IDF features as a text encoder. We train models via back-propagation and Adam optimizer (Kingma and Ba, 2015). When validation loss stops reducing by 10 epochs, the training will be stopped earlier (Yuan et al., 2007). The maximum value of iterations is 200.

4.2 Results of Fake News Detection

Table 2 and Table 3 show the results of fake news detection on four datasets. Through employing UniPF, the performance of all baselines is improved to different extents, which shows the effectiveness of the framework. Several conclusions can be drawn as follows:

1) Propagation-based methods are often superior to those only using text information. This indicates the importance of learning structural features for fake news detection. As fake news publishers may

Method	PolitiFact				GossipCop			
	Acc	P	R	F1	Acc	P	R	F1
UniPF-BiGCN	**0.906**	**0.901**	**0.906**	**0.904**	**0.968**	**0.968**	**0.969**	**0.968**
- w/o SC-G	0.868	0.864	0.867	0.866	0.963	0.959	0.968	0.963
- w/o RT	0.889	0.869	0.913	0.886	0.967	0.968	0.967	0.966
UniPF$_{\text{RS-G}}$-BiGCN	0.871	0.871	0.871	0.869	0.963	0.964	0.964	0.963
- w/o RT	0.868	0.866	0.857	0.865	0.951	0.944	0.956	0.951

Table 4: The ablation study results on *PolitiFact* and *GossipCop* datasets.

Method	Twitter15				Twitter16			
	Acc	P	R	F1	Acc	P	R	F1
UniPF-BiGCN	**0.952**	**0.959**	**0.945**	**0.952**	**0.955**	**0.962**	**0.951**	**0.955**
- w/o SC-G	0.951	0.952	0.951	0.950	0.947	0.953	0.942	0.946
- w/o RT	0.949	0.956	0.942	0.949	0.952	0.956	0.948	0.951
UniPF$_{\text{RS-G}}$-BiGCN	0.945	0.950	0.939	0.945	0.951	0.930	0.976	0.951
- w/o RT	0.939	0.945	0.934	0.939	0.949	0.951	0.951	0.950

Table 5: The ablation study results on *Twitter15* and *Twitter16* datasets.

deliberately rub off on the heated topic or disguise themselves by imitating other users, it is challenging to learn informative indicators from noisy texts.

2) In text-based fake news detection methods, UniPF significantly boosts existing models' performance in terms of most metrics. It suggests that fully exploiting deep semantic correlations between similar propagation trees can provide positive auxiliary information to detect fake news. UniPF formulates propagation forest as graph and retrieves semantic-related patterns from textual contents of other propagation trees. These patterns can purified noisy fine-grained correlations obtained by original base model and accordingly learn more easily discernible semantic features for detection.

3) In propagation-based fake news detection models, UniPF framework can also consistently improve the corresponding base models. These base models only capture structural features from the target propagation tree. UniPF effectively explores semantic-structural correlations between propagation trees with the guidance of prototypical root nodes in the propagation forest, thus more discriminative patterns are injected into the representation of the target sample.

4.3 Ablation Study

We perform ablation studies by comparing with: **w/o SC-G** refers to removing the graph-based transformations and learning embedding of root nodes with a simple fully-connected layer. **w/o RT** refers to Root-induced Training strategy is removed in the training process. **UniPF$_{\text{RS-G}}$** performs Randomly-Sampling to generate root nodes and apply Graph-based transformations to explore deep correlations in the propagation forest.

The results are shown in Table 4 and Table 5. The full UniPF consistently yields the best performance, which shows the effectiveness of the proposed framework for enhancing the detection performance.

1) UniPF w/o SC-G removing graph-based transformations is inferior to UniPF. It shows the embedding of the target sample can be enhanced by explicitly modeling latent correlations in the propagation forest.

2) After removing the root-induced unsupervised clustering loss, results of both UniPF w/o RT and UniPF$_{\text{RS-G}}$ w/o RT are reduced. Both variants generate prototypical root nodes in a semantic-clustering or random-sampling way. It implies the efficacy of our training strategy, which ensures the quality of prototypical root nodes. Accordingly, more effective semantic-structural features can be injected to boost the detection performance.

3) UniPF$_{\text{RS-G}}$ ignoring semantic-aware clustering achieves poor performance on four datasets, which shows the effectiveness of semantic clustering in our model. The fact reveals more beneficial correlations can be explored between semantically similar propagation trees.

4.4 Parameter Analysis

Figure 3 shows the performance against different numbers of root nodes K in the propagation forest. From results, UniPF is influenced by the number of

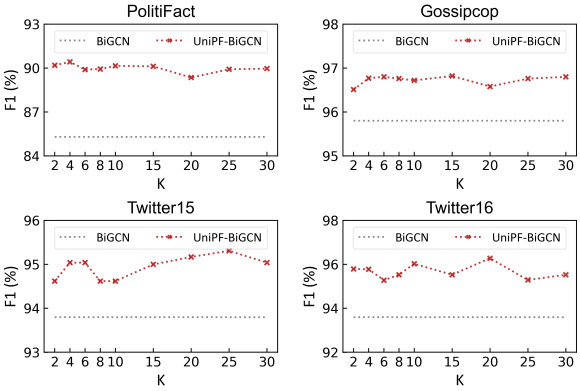

Figure 3: F1 score against different values of K.

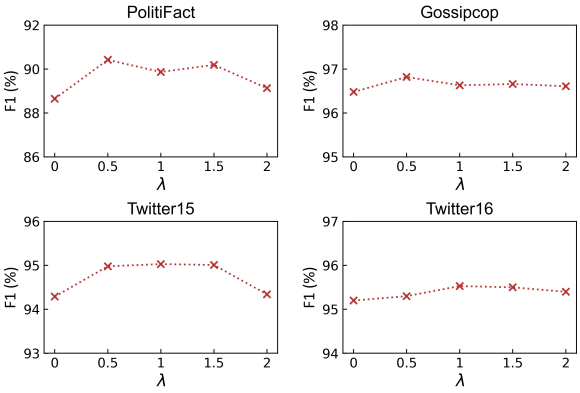

Figure 4: F1 score against different values of λ.

root nodes since these root nodes act as propagation tree's ancestor in the forest and play a pivot role to incorporate correlations between similar propagation trees. Under too smaller number of root nodes, some redundant or noisy features may be introduced at the same time; while a larger number of root nodes also hurts the performance since assigning more root nodes means propagation trees are independent. The optimal value is 4, 15, 25, and 20 on *PolitiFact*, *GossipCop*, *Twitter15* and *Twitter16* datasets, respectively.

Figure 4 reports the performance against different values of λ, a trade-off hyper-parameter of the unsupervised node-induced clustering loss in the root-induced training strategy. Performance first gets better with the increase of λ. The improvements demonstrate the effectiveness of the root-induced training strategy, which can further ensure high-quality root nodes in the propagation forest. The best setting of 0.5, 0.5, 1, and 1 on *PolitiFact*, *GossipCop*, *Twitter15* and *Twitter16* datasets, respectively. However, performance declines when applying a large value of λ. Because the model with too large λ would pay more attention to semantic clustering based on all propagation trees and cannot effectively exploit structural information of individual propagation trees.

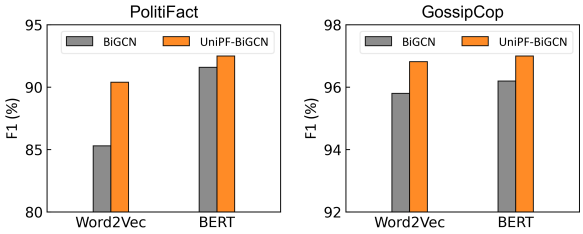

Figure 5: Performance with different textual features on *PolitiFact* and *GossipCop* datasets.

Model	Number of Parameters	Average Time
BiGCN	427.5k	2.35s
UniPF-BiGCN	575.2k	2.47s

Table 6: Efficiency analysis on the *PolitiFact* dataset.

4.5 Feature Analysis

BERT (Devlin et al., 2019) has shown powerful ablity to encode textual features and been applied in many NLP tasks (Hu et al., 2022a,b). To investigate the robustness and universality of UniPF, we follow Dou et al. (2021) and evaluate the performance against different feature extractors to explore that given different textual features, whether UniPF can still fully explore latent correlations between propagation trees. Figure 5 shows the fake news detection performance using two different textual features, i.e., Word2Vec (Mikolov et al., 2013) and BERT (Devlin et al., 2019).

From the results, the proposed framework can effectively boost the performance of the baseline model whether BERT or Word2Vec is used to initialize textual features, which reveals the robustness and effectiveness of the proposed UniPF framework. Besides, textual features extracted by BERT are better than Word2Vec. The fact shows contextual features can be captured by the pre-trained model from a large corpus.

4.6 Efficiency Analysis

As shown in Table 6, after employing UniPF, the number of model parameters is 34.5% more than that of BiGCN. The time difference is about 0.12s. Although UniPF would introduce a small number of parameters, the average spent time of each epoch is similar to the base model. We argue it is worthwhile to cost a very small amount of computing resources to provide better results.

5 Related Work

Some works exploit user information to detect fake news such as user credibility (Yuan et al., 2020), user profiles (Shu et al., 2019b) and user social relations (Shu et al., 2019a). However, due to user privacy protection, user's information is not allowed recorded on social media. This paper focuses on detecting fake news based on *anonymous propagation*. We review related works in two groups: content- and propagation-based detection methods.

5.1 Content-based Detection Methods

Content-based detection approaches investigate the truthfulness of news content by extracting its textual features. Early works (Castillo et al., 2011; Zhao et al., 2015; Popat, 2017; Potthast et al., 2018; Ajao et al., 2019) reply on feature engineering to capture textual attributes. After the emergence of deep learning, many works (Ma et al., 2016; Ruchansky et al., 2017; Shu et al., 2019a; Zhou et al., 2020; Kaliyar et al., 2021) apply various neural networks to automatically learn semantic features from source content and its comments to detect fake news. In reality, fake news is usually published around a specific event/topic, leading to potentially similar semantic characteristics. However, most of them usually extract fine-grained features from words but may suffer from polysemy problem and are not effective to learn shared semantic features. This paper explores potential semantics via generating cluster-prototype and integrating information under the propagation forest, fully exploring shared and identifiable semantic features to boost detection.

Understanding complicated dissemination patterns from propagation trees is also critical since this gives valuable hints into the discovery of fake new. Prior studies (Castillo et al., 2011; Vosoughi et al., 2018; Zhao et al., 2021) focus on several propagation-related criteria such as total number of nodes in a propagation tree, propagation tree depth and breadth. Within a deep learning framework, (Ma et al., 2016; Shu et al., 2020b; Khoo et al., 2020) learn the representation of the target propagation tree with neural networks to capture geometrical spreading patterns. More recently, many mainstream studies (Hu et al., 2019; Dong et al., 2019; Nguyen et al., 2020; Bian et al., 2020; Hu et al., 2021; Silva et al., 2020; Wei et al., 2021, 2022; Silva et al., 2021) regard the propagation tree as a graph structure and apply various graph-based techniques to learn richer structural features. Although obtaining promising results to some extent, they usually assume propagation trees are independent and ignore vital high-level correlations among propagation. To alleviate this issue, we develop the propagation forest and propose a generic framework to fully explore latent semantic and structural correlations across propagation.

6 Conclusion

In this paper, we investigate deep correlations between propagation trees and propose a generic framework UniPF for improving fake news detection. UniPF builds a propagation forest to naturally combine propagation trees in a semantic-clustering manner. The representation of the target sample is enhanced with shared patterns from similar propagation trees and prototypical root nodes. Besides, a node-induced training strategy is designed for guaranteeing the consistency of the representation of propagation trees and newly prototypical root nodes. Experiments on four datasets consistently prove the scalability and effectiveness of UniPF.

For future work, we will explore more available information (such as images) for perfecting propagation forest to enhance detection.

Acknowledgments

We thank our anonymous reviewers for their helpful comments. This work was supported by the National Natural Science Foundation of China under Grant No.6210071416.

References

Sadia Afroz, Michael Brennan, and Rachel Greenstadt. 2012. Detecting hoaxes, frauds, and deception in writing style online. In *IEEE Symposium on Security and Privacy*, pages 461–475. IEEE Computer Society.

Oluwaseun Ajao, Deepayan Bhowmik, and Shahrzad Zargari. 2019. Sentiment aware fake news detection on online social networks. In *ICASSP*, pages 2507–2511. IEEE.

David Arthur and Sergei Vassilvitskii. 2007. k-means++: the advantages of careful seeding. In *SODA*, pages 1027–1035. SIAM.

Tian Bian, Xi Xiao, Tingyang Xu, Peilin Zhao, Wenbing Huang, Yu Rong, and Junzhou Huang. 2020. Rumor detection on social media with bi-directional graph convolutional networks. In *AAAI*, pages 549–556. AAAI Press.

Carlos Castillo, Marcelo Mendoza, and Barbara Poblete. 2011. Information credibility on twitter. In *WWW*, pages 675–684. ACM.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: pre-training of deep bidirectional transformers for language understanding. In *NAACL-HLT*, pages 4171–4186. Association for Computational Linguistics.

N. Difonzo and P. Bordia. 2007. Rumor, gossip and urban legends. *Diogenes*, 54(1):19–35.

Ming Dong, Bolong Zheng, Nguyen Quoc Viet Hung, Han Su, and GuoHui Li. 2019. Multiple rumor source detection with graph convolutional networks. In *CIKM*, pages 569–578. ACM.

Yingtong Dou, Kai Shu, Congying Xia, Philip S. Yu, and Lichao Sun. 2021. User preference-aware fake news detection. In *SIGIR*, pages 2051–2055. ACM.

Adrien Friggeri, Lada Adamic, Dean Eckles, and Justin Cheng. 2014. Rumor cascades.

Matthew Honnibal and Ines Montani. 2017. spacy 2: Natural language understanding with bloom embeddings, convolutional neural networks and incremental parsing. *To appear*, 7(1).

Dou Hu, Xiaolong Hou, Lingwei Wei, Lian-Xin Jiang, and Yang Mo. 2022a. MM-DFN: multimodal dynamic fusion network for emotion recognition in conversations. In *ICASSP*, pages 7037–7041. IEEE.

Dou Hu, Lingwei Wei, Wei Zhou, Xiaoyong Huai, Jizhong Han, and Songlin Hu. 2021. A rumor detection approach based on multi-relational propagation tree. *Journal of Computer Research and Development*, 58(7):1395–1411.

Dou Hu, Mengyuan Zhou, Xiyang Du, Mengfei Yuan, Jin Zhi, Lian-Xin Jiang, Yang Mo, and Xiaofeng Shi. 2022b. PALI-NLP at semeval-2022 task 4: Discriminative fine-tuning of transformers for patronizing and condescending language detection. In *SemEval@NAACL*, pages 335–343. Association for Computational Linguistics.

Guoyong Hu, Ye Ding, Shuhan Qi, Xuan Wang, and Qing Liao. 2019. Multi-depth graph convolutional networks for fake news detection. In *NLPCC*, volume 11838 of *Lecture Notes in Computer Science*, pages 698–710. Springer.

Qi Huang, Junshuai Yu, Jia Wu, and Bin Wang. 2020. Heterogeneous graph attention networks for early detection of rumors on twitter. In *IJCNN*, pages 1–8.

Jaroslaw Jankowski, Piotr Bartkow, Patryk Pazura, and Kamil Bortko. 2020. Evaluation of the costs of delayed campaigns for limiting the spread of negative content, panic and rumours in complex networks. In *ICCS*, volume 12140 of *Lecture Notes in Computer Science*, pages 291–304. Springer.

Zhiwei Jin, Juan Cao, Han Guo, Yongdong Zhang, Yu Wang, and Jiebo Luo. 2017. Detection and analysis of 2016 US presidential election related rumors on twitter. In *SBP-BRiMS*, volume 10354 of *Lecture Notes in Computer Science*, pages 14–24. Springer.

Rohit Kumar Kaliyar, Anurag Goswami, and Pratik Narang. 2021. Fakebert: Fake news detection in social media with a bert-based deep learning approach. *Multim. Tools Appl.*, 80(8):11765–11788.

Ling Min Serena Khoo, Hai Leong Chieu, Zhong Qian, and Jing Jiang. 2020. Interpretable rumor detection in microblogs by attending to user interactions. In *AAAI*, pages 8783–8790. AAAI Press.

Diederik P. Kingma and Jimmy Ba. 2015. Adam: A method for stochastic optimization. In *ICLR*.

Thomas N. Kipf and Max Welling. 2017. Semi-supervised classification with graph convolutional networks. In *ICLR*. OpenReview.net.

Xiaomo Liu, Armineh Nourbakhsh, Quanzhi Li, Rui Fang, and Sameena Shah. 2015. Real-time rumor debunking on twitter. In *CIKM*, pages 1867–1870. ACM.

Yi-Ju Lu and Cheng-Te Li. 2020. GCAN: graph-aware co-attention networks for explainable fake news detection on social media. In *ACL*, pages 505–514. Association for Computational Linguistics.

Jing Ma, Wei Gao, Prasenjit Mitra, Sejeong Kwon, Bernard J. Jansen, Kam-Fai Wong, and Meeyoung Cha. 2016. Detecting rumors from microblogs with recurrent neural networks. In *IJCAI*, pages 3818–3824. IJCAI/AAAI Press.

Jing Ma, Wei Gao, and Kam-Fai Wong. 2017. Detect rumors in microblog posts using propagation structure via kernel learning. In *ACL*, pages 708–717. Association for Computational Linguistics.

Jing Ma, Wei Gao, and Kam-Fai Wong. 2018. Rumor detection on twitter with tree-structured recursive neural networks. In *ACL*, pages 1980–1989. Association for Computational Linguistics.

Tomás Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013. Efficient estimation of word representations in vector space. In *ICLR (Workshop Poster)*.

Federico Monti, Fabrizio Frasca, Davide Eynard, Damon Mannion, and Michael M. Bronstein. 2019. Fake news detection on social media using geometric deep learning. In *ICLR (Workshop)*.

Arvind Neelakantan, Jeevan Shankar, Alexandre Passos, and Andrew McCallum. 2014. Efficient non-parametric estimation of multiple embeddings per word in vector space. In *EMNLP*, pages 1059–1069.

Van-Hoang Nguyen, Kazunari Sugiyama, Preslav Nakov, and Min-Yen Kan. 2020. FANG: leveraging social context for fake news detection using graph representation. In *CIKM*, pages 1165–1174. ACM.

Armineh Nourbakhsh, Xiaomo Liu, Sameena Shah, Rui Fang, Mohammad Mahdi Ghassemi, and Quanzhi Li. 2015. Newsworthy rumor events: A case study of twitter. In *ICDM Workshops*, pages 27–32. IEEE Computer Society.

Kashyap Popat. 2017. Assessing the credibility of claims on the web. In *Proceedings of the 26th International Conference on World Wide Web Companion*, pages 735–739. ACM.

Martin Potthast, Johannes Kiesel, Kevin Reinartz, Janek Bevendorff, and Benno Stein. 2018. A stylometric inquiry into hyperpartisan and fake news. In *ACL*, pages 231–240.

Natali Ruchansky, Sungyong Seo, and Yan Liu. 2017. CSI: A hybrid deep model for fake news detection. In *CIKM*, pages 797–806. ACM.

Kai Shu, Limeng Cui, Suhang Wang, Dongwon Lee, and Huan Liu. 2019a. defend: Explainable fake news detection. In *KDD*, pages 395–405. ACM.

Kai Shu, Deepak Mahudeswaran, Suhang Wang, Dongwon Lee, and Huan Liu. 2020a. Fakenewsnet: A data repository with news content, social context, and spatiotemporal information for studying fake news on social media. *Big Data*, 8(3):171–188.

Kai Shu, Deepak Mahudeswaran, Suhang Wang, and Huan Liu. 2020b. Hierarchical propagation networks for fake news detection: Investigation and exploitation. In *ICWSM*, pages 626–637.

Kai Shu, Xinyi Zhou, Suhang Wang, Reza Zafarani, and Huan Liu. 2019b. The role of user profiles for fake news detection. In *International Conference on Advances in Social Networks Analysis and Mining*, pages 436–439. ACM.

Amila Silva, Yi Han, Ling Luo, Shanika Karunasekera, and Christopher Leckie. 2020. Embedding partial propagation network for fake news early detection. In *CIKM (Workshops)*, volume 2699 of *CEUR Workshop Proceedings*. CEUR-WS.org.

Amila Silva, Yi Han, Ling Luo, Shanika Karunasekera, and Christopher Leckie. 2021. Propagation2vec: Embedding partial propagation networks for explainable fake news early detection. *Inf. Process. Manag.*, 58(5):102618.

Jake Snell, Kevin Swersky, and Richard S. Zemel. 2017. Prototypical networks for few-shot learning. In *NIPS*, pages 4077–4087.

Chenguang Song, Kai Shu, and Bin Wu. 2021. Temporally evolving graph neural network for fake news detection. *Inf. Process. Manag.*, 58(6):102712.

Soroush Vosoughi, Deb Roy, and Sinan Aral. 2018. The spread of true and false news online. *Science*, 359(6380):1146–1151.

Lingwei Wei, Dou Hu, Wei Zhou, Xin Wang, and Songlin Hu. 2022. Modeling the uncertainty of information propagation for rumor detection: A neuro-fuzzy approach. *IEEE Transactions on Neural Networks and Learning Systems*.

Lingwei Wei, Dou Hu, Wei Zhou, Zhaojuan Yue, and Songlin Hu. 2021. Towards propagation uncertainty: Edge-enhanced bayesian graph convolutional networks for rumor detection. In *ACL/IJCNLP*, pages 3845–3854.

Junyuan Xie, Ross B. Girshick, and Ali Farhadi. 2016. Unsupervised deep embedding for clustering analysis. In *ICML*, volume 48 of *JMLR Workshop and Conference Proceedings*, pages 478–487. JMLR.org.

Chunyuan Yuan, Qianwen Ma, Wei Zhou, Jizhong Han, and Songlin Hu. 2019. Jointly embedding the local and global relations of heterogeneous graph for rumor detection. In *ICDM*, pages 796–805. IEEE.

Chunyuan Yuan, Qianwen Ma, Wei Zhou, Jizhong Han, and Songlin Hu. 2020. Early detection of fake news by utilizing the credibility of news, publishers, and users based on weakly supervised learning. In *COLING*, pages 5444–5454. International Committee on Computational Linguistics.

Yao Yuan, Lorenzo Rosasco, and Andrea Caponnetto. 2007. On early stopping in gradient descent learning. *Constructive Approximation*, 26(2):289 – 315.

Yuehua Zhao, Jingwei Da, and Jiaqi Yan. 2021. Detecting health misinformation in online health communities: Incorporating behavioral features into machine learning based approaches. *Inf. Process. Manag.*, 58(1):102390.

Zhe Zhao, Paul Resnick, and Qiaozhu Mei. 2015. Enquiring minds: Early detection of rumors in social media from enquiry posts. In *WWW*, pages 1395–1405. ACM.

Cheng Zhou, Kai Li, and Yanhong Lu. 2021. Linguistic characteristics and the dissemination of misinformation in social media: The moderating effect of information richness. *Inf. Process. Manag.*, 58(6):102679.

Xinyi Zhou, Jindi Wu, and Reza Zafarani. 2020. SAFE: similarity-aware multi-modal fake news detection. In *PAKDD*, volume 12085 of *Lecture Notes in Computer Science*, pages 354–367. Springer.

CLoSE: Contrastive Learning of Subframe Embeddings for Political Bias Classification of News Media

Michelle YoungJin Kim and **Kristen Marie Johnson**
Computer Science and Engineering, Michigan State University, USA
{kimmic16,kristenj}@msu.edu

Abstract

Framing is a political strategy in which journalists and politicians emphasize certain aspects of a societal issue in order to influence and sway public opinion. Frameworks for detecting framing in news articles or social media posts are critical in understanding the spread of biased information in our society. In this paper, we propose CLoSE, a multi-task BERT-based model which uses contrastive learning to embed indicators of frames from news articles in order to predict political bias. We evaluate the performance of our proposed model on subframes and political bias classification tasks. We also demonstrate the model's classification accuracy on zero-shot and few-shot learning tasks, providing a promising avenue for framing detection in unlabeled data.

1 Introduction

News media coverage shapes our attitudes, emotions, and decisions toward public issues (Iyengar, 1994; Pan and Kosicki, 1993; Jensen et al., 2014). Research shows that people's perceptions of news can be manipulated by presenting the same story with different expressions. For example, participants of a study found a terrorist attack caused by "al-Qaeda and associated radical Islamic groups" considerably more threatening than a terrorist attack by "domestic rebel separatist groups," which is an equivalent paraphrase (Kapuściński and Richards, 2016). Hence, studies on the influences of different presentations of issues, or the effects of *framing*, play an essential role in understanding political discourse.

Framing refers to emphasizing desired aspects of an issue to promote and amplify a particular perspective (Entman, 1993). By selecting and thus elevating the salience of a specific angle of a topic, media sources can present the topic through their choice of frames to induce particular attributes and judgments among the public. Framing is widely researched on various topics, from its effects on public opinion on political issues such as the U.S. anti-nuclear war movement (Entman and Rojecki, 1993) and stem cell research (Nisbet et al., 2003) to the economic impact of framing (Liu and Pennington-Gray, 2015; Van Dalen et al., 2017).

In this work, we study framing detection for three politically polarized issues in the U.S. news media: abortion, gun control, and immigration. We focus on framing in the news discourse to understand a discrepancy in media consumption and its influence on political bias polarization. With the increase of news media outlets and social media platforms, the public is overwhelmed with a flood of information. Unfortunately, the increase of news sources does not yield the sharing of information across political views, often developing into biased echo chambers on social media platforms. In fact, there are stark differences in the social media and news sources that liberals and conservatives use and trust. Fox News is the primary news source for nearly half of conservatives, while NPR, MSNBC, and the New York Times are the most trusted news sources for liberals (Mitchell et al., 2014).

Machine learning techniques have been applied to detect and analyze political frames (Card et al., 2015; Guo et al., 2016; Johnson et al., 2017a; Bhatia et al., 2021). In such framing analyses, the performance of a framing detection model is tested by predicting the political bias of an article or the political affiliation or stance of a politician's tweet or speech. However, we suggest incorporating the political bias information into the development of the actual frame detection model. We focus on general liberal and conservative bias rather than the specific stances politicians may take on issues.

In this paper, we propose CLoSE, a framing embedding model that jointly learns to predict political bias. CLoSE fine-tunes BERT variants with a contrastive learning objective to generate (sub)frame embeddings for a given input sentence. Then we

Proceedings of the 29th International Conference on Computational Linguistics, pages 2780–2793
October 12–17, 2022.

add a prediction loss to classify the political bias (liberal or conservative) of the embedded text.

For the embedding task, we use contrastive learning to place embeddings of the same *subframe* closer together. Subframes are fine-grained subcategorizations of the general political frames of Boydstun et al. (2014). The topic-specific lexicons of subframes are the *subframe indicators* used to identify specific framing language within a chosen topic. We use the subframe indicators to identify texts with frames and construct the Framing Triplet Dataset. This dataset, built explicitly for a contrastive objective, consists of a triplet of an anchor sentence, its positive sample, and its negative sample, and the political bias label of the anchor sentence. The contrastive learning objective reduces the distance between the anchor sentence and its positive sample, which belongs to the same subframe, while increasing the distance to its negative sample that belongs to a different subframe.

CLoSE outperforms the baseline models in both the subframe and political bias classification tasks. The results also show that the contexts of subframes improve the performance of the political bias classifier. Further, our pre-trained model accomplishes superior performance in zero-shot and few-shot settings. Finally, we design a topic modeling method for the subframe embeddings that clusters nearby embeddings, extracts statistically significant words with class-based TF-IDF (Grootendorst, 2022), and merges clusters with high overlapping words.

To summarize, our main contributions are as follows: (1) We collect and release the Triplet Framing Dataset[1], a triplet of sentences that include subframe indicators. (2) We propose CLoSE, contrastive learning of subframe embeddings model that jointly generates embeddings and predicts political bias of framed texts. The experiments demonstrate our method's performance on subframe and political bias classification tasks and investigate the effectiveness of political bias information on subframe detection and vice versa. (3) The experiments show that our pre-trained model performs competitively on zero-shot political bias classification and few-shot subframe classification tasks. Namely, the pre-trained model can predict the political bias of articles with previously unseen topics and predict subframe groups with a limited quantity of labeled data. This greatly reduces the cost of data annotation for framing and bias tasks.

[1]Code and data available at https://github.com/MSU-NLP-CSS/CLoSE_framing.

2 Related Work

Framing is a powerful political strategy that is used to influence public opinion. Hence identifying what and how frames are used is a critical task in political communications. Framing in news media and social networks has been studied to analyze political polarization (Johnson and Goldwasser, 2016; Tsur et al., 2015; Tourni et al., 2021). However, annotating data for framing is challenging due to the nuanced language of frames across political issues.

To overcome this challenge, computational social scientists follow a topic-specific codebook to manually annotate documents with frames (Terkildsen and Schnell, 1997; Baumgartner et al., 2008; Card et al., 2015). The most commonly used codebook is the Policy Frames Codebook (Boydstun et al., 2014), which proposes a general coding scheme for fifteen high-level frames across policy issues. Based on this codebook, Card et al. (2015) collected and released the Media Frames Corpus (MFC). The MFC contains more than 20,000 news articles on three policy issues: immigration, smoking, and same-sex marriage.

Following the release of MFC, the first large-scale open-source frames dataset, many researchers studied and analyzed frames by leveraging the MFC. Johnson et al. (2017b) use political tweets to extract phrases that frequently appear in each of the frames and propose a framing detection model that uses both linguistic features and the extracted ideological phrases. Field et al. (2018) extend the U.S. policy frames to Russian policy frames and analyze 13 years of Russian news articles. They derive the lexicons of each frame from the MFC and translate them to Russian to generate Russian framing lexicons. Huguet Cabot et al. (2020) propose a multi-task model that learns metaphor, framing, and emotion in political discourse and uses the MFC to predict frames. Although our method also jointly models political bias and framing, our approach differs in that the main task of our model is to embed language used in subframes with contrastive learning.

Additional research has narrowed down the general policy frames to suggest issue-specific frames. Johnson et al. (2017a) add Twitter-specific frames to the policy frames and annotate and analyze the tweets of U.S. politicians. Roy and Goldwasser (2020) build topic-specific lexicons, which they de-

fine as *subframe indicators*, by using an embedding model to generalize the MFC lexicon for analyzing media bias. We utilize their subframe indicators to create our Framing Triplet Dataset. Our proposed model also includes an embedding model that generalizes text containing subframe indicators. However, we integrate political bias information into our model directly by adding bias classification as an auxiliary task.

Other approaches for detecting frames are based on topic modeling algorithms. Latent topics of a given corpus are extracted with a topic modeling algorithm, often Latent Dirichlet Allocation (LDA) (Blei et al., 2003). LDA yields statistically significant words of each topic, which are used as candidate indicators for defining frames. Bhatia et al. (2021) provide an open-sourced tool for analyzing frames in multilingual texts. Given text inputs, their web-based system sends the LDA topic modeling output to a user's email so that the user can decide and label frames on the given topics. However, we note that the output of LDA is a list of keywords in each topic, *not* frame. The topic-based words are helpful guidance in framing annotations and exploratory analysis but are not appropriate data to be used for supervised framing analysis.

3 Framing Triplet Dataset

We introduce the Framing Triplet Dataset, which contains 25,627 news articles on three politically polarized topics: abortion, gun control, and immigration. We extend the hyper-partisan news dataset (Kiesel et al., 2019) and the dataset by Roy and Goldwasser (2020), which consists of news articles on the three topics until 2019. Open-source crawlers `pygooglenew` and `news-please` were used to collect recent articles from 2020 to 2022. We query the Google News RSS feed with keywords of each topic and crawl the articles. The keywords are "abortion" for the abortion data, "gun" for the gun control data, and "immigrant" and "immigration" for the immigration data. Then we assign the political bias label of each article according to Media Bias Fact Check [2], the largest crowdsourced media bias resource on more than 4,500 media sources and journalists. The Media Bias Fact Check categorizes each media source into one of the following five biases: "left bias," "left-center bias," "least bias," "right-center bias," and "right bias." For our study, given the partisan divide of current U.S. politics, we only consider articles from "left bias" and "right bias" media. In total, our dataset includes 12,911 left-biased articles and 12,761 right-biased articles. The detailed statistics of the dataset can be found in Table 1.

After labeling the political bias of the collected articles, we extract headlines and sentences in articles that contain *subframe indicators*. The subframe indicators, suggested by Roy and Goldwasser (2020), are the issue-specific subclassifications of the policy frames by Boydstun et al. (2014). These subframe indicators are collected via the following three steps. First, the top-250 unigrams for each of the 15 policy frames are gathered based on their Pointwise Mutual Information (PMI) scores (Church and Hanks, 1990). Then, each paragraph in the article is annotated with the policy frames based on the number of unigram matches. Finally, repeating phrases in the annotated paragraphs are grouped to form subframes, which represent topic-specific lexicons. They defined 20 subframes for the topic of abortion, 22 subframes for the topic of immigration, and 19 subframes for the topic of gun control. The list of subframes can be found in Appendix A. We refer to Roy and Goldwasser (2020) for the full list of subframe indicators.

Finally, we construct triplets with the extracted sentences. Given an anchor sentence s_a, we define its positive sample s_p as a sentence with subframes from the same subframe group and its negative sample s_n as a sentence with subframes from a different subframe group. For example, the sentence "The first backlash to *the Roe decision* came primarily from groups representing U.S. Catholics." contains a subframe indicator "Roe decision" that belongs to the subframe "Roe v. Wade." Its positive sample (s_p) should be a sentence with a subframe indicator of that same subframe, "Roe v. Wade." Its negative example (s_n) will be a sentence with a subframe indicator from a different, random subframe group such as "Birth Control." The triplet is formed as (s_a, s_p, s_n). An example of the triplet data is shown in Figure 2.

3.1 Data Analysis

We analyze the usage of subframes in the Framing Triplet Dataset by extracting the top-3 subframes across time and issues. The results for the abortion subdataset are shown in Table 2. (See Appendix B for the results on the gun control and immigration subdatasets.) Across the topics, we notice the trend

[2] `mediabiasfactcheck.com`

	News(#)	Left(#)	Right(#)	Sent.(#)	Left S.(#)	Right S.(#)	Time Span
Abortion	8,061	4,518	3,543	10,725	8,032	2,693	1984-2022
Gun control	9,476	4,238	5,238	8,138	3,918	4,220	2000-2022
Immigration	8,090	4,155	3,935	13,269	8,228	5,041	1996-2022

Table 1: Statistics of the Framing Triplet Dataset. For each topic, news articles are collected, their biases are labeled according to Media Bias Fact Check, and sentences containing subframe indicators are extracted. *Left S.* are sentences extracted from left-biased news articles. *Right S.* are extracted from right-biased news articles.

Years		Liberal	Conservative
2020-2022	Top-1	Roe v. Wade	Roe v. Wade
	Top-2	Abortion Funding	Abortion Funding
	Top-3	Birth Control	Birth Control
2018-2020	Top-1	Roe v. Wade	Roe v. Wade
	Top-2	Birth Control	Pregnancy Centers
	Top-3	Abortion Funding	Abort. Prov. Economy
2016-2018	Top-1	Birth Control	Pregnancy Centers
	Top-2	Health Care	Abort. Prov. Economy
	Top-3	Roe v. Wade	Abortion Funding
2010-2016	Top-1	Abortion Funding	Birth Control
	Top-2	Birth Control	Abort. Prov. Economy
	Top-3	⋆	Planned Parenthood
-2010	Top-1	Birth Control	Roe v. Wade
	Top-2	Roe v. Wade	Right of Human Life
	Top-3	⋆	Abortion Funding

⋆ Potentially offensive or triggering language has been omitted.

Table 2: Top-3 Subframe Indicators of the Abortion Triplet Dataset. Each row indicates the most frequently used subframe indicators for liberal and conservative biased media within the specified time frame.

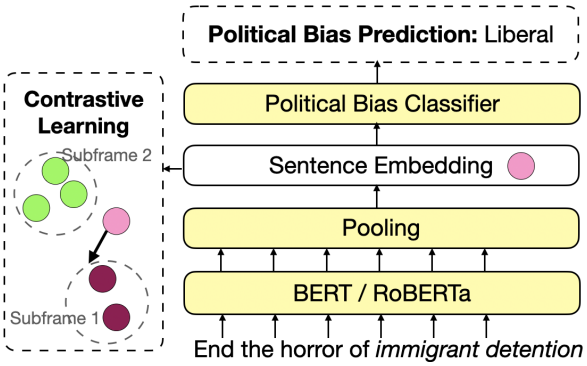

Figure 1: Architecture of CLoSE. Our model consists of a BERT- or RoBERTa-based encoder whose output is fed through a pooling layer to generate sentence embeddings via a contrastive learning objective. These embeddings are then used in the final political bias classification task. This example shows an input sentence and its corresponding embedding (pink circle), which is most similar to other embedded sentences of the Subframe 1's group. The final prediction here is liberal.

that the most used subframes begin to overlap as time passes. Before 2018, the top-3 subframes used by the liberal and conservative media significantly differed. Namely, the subframe indicators that the media focused on to frame people's opinions on abortion are different based on the media's political bias. However, the most used subframes of the left and right news are intersecting more recently. From 2018 to 2020, the most used subframe of both liberal and conservative media was "Roe v. Wade." The top-3 subframes from 2020 to 2022 of both views are also identical. Similarly, from 2020 to 2022, the top-3 subframes for the gun control subdataset, and the top-2 subframes for the immigration subdataset, of both liberal and conservative media are identical. These results show that in recent years, news media with opposing political biases cover common issues, sometimes with similar framing language. Despite this similar coverage, political polarization in the U.S. remains dominant (Doherty et al., 2021). Hence, we need a methodology that captures not only which issues the media chooses to spotlight for coverage but also how they frame these issues for public perception. By jointly learning to predict political bias from the sentence embeddings, our proposed method aims to capture the different contexts of the opposing media.

4 Methodology

We propose CLoSE: a multi-task learning model that jointly learns to embed sentence framing language and predict political bias. As shown in Figure 1, the model consists of a BERT-based or RoBERTa-based encoder, followed by a pooling layer that generates a sentence embedding, which feeds into a classifier for political bias prediction. We adopt a contrastive learning objective so that a sentence with subframe indicators close to other sentences of the same subframe and far from those of different subframes will be similarly reflected in the embedding space.

Embedding via pooling. Given a sentence s of tokens $\{t_1, \cdots, t_n\}$, we embed the sentence using BERT or RoBERTa to get a sequence of token embeddings $\{e(t_1), \cdots, e(t_n)\}$. Then a pooling operation is applied to the token embeddings to

derive a fixed size sentence embedding $e(s)$. There are three possible pooling strategies: (1) taking the output of the CLS token, (2) computing the mean of all output vectors, and (3) computing a max-over-time of the output vectors. CLoSE employs the default pooling operation of mean strategy, which has shown the best experimental results on semantic textual similarity and natural language inference tasks (Reimers and Gurevych, 2019). Thus, the final sentence embedding is $e(s) = \frac{1}{n}\sum_{i=1}^{n} e(t_i)$.

Learn subframes with a contrastive learning objective. We apply a contrastive learning objective to capture the framing embedding space. To do so, we fine-tune the BERT-based (or RoBERTa-based) encoder to encourage embeddings that will fall within the same subframe group to be closer and the embeddings within a different subframe group to be more distant from a given anchor sentence's embedding. Every anchor sentence s_a has a positive sentence s_p and a negative sentence s_n, corresponding to the triplets within the Framing Triplet Dataset. For example, as shown in Figure 2, the anchor sentence has a subframe indicator "Roe decision" (shown in bold), which belongs to the subframe group "Roe v. Wade." While its positive sentence has subframe indicators "overturns Roe" and "Roe v. Wade" that belong to the same subframe group, the negative sentence has a subframe indicator "March for Life" that belongs to a different subframe group "Pro-Life."

Given a triplet of sentences (s_a, s_p, s_n), the contrastive learning loss is computed as follows:

$$\mathcal{L}_c = \max(0, \epsilon - \|e(s_a) - e(s_n)\| + \|e(s_a) - e(s_p)\|)$$

where the margin ϵ is a hyperparameter, and $\|\cdot\|$ is the Euclidean distance. $e(s_a), e(s_p)$, and $e(s_n)$ are an embedding of the anchor sentence s_a, its positive sentence s_p, and its negative sentence s_n, respectively. We set ϵ as 1 for the experiments.

Predict political bias. The proposed model adds a binary classifier to the output of the BERT-based encoder to predict the political bias of an input (anchor) sentence. The sentence embedding of this anchor sentence is given as an input to the classifier, which predicts whether the sentence is from a liberal (left) or conservative (right) news source. Given an anchor sentence s_i, we predict a stance label $y_i \in \{0, 1\}$. We use binary cross-entropy as the loss function:

$$\mathcal{L}_{\text{BCE}} = -\frac{1}{N}\sum_{i=1}^{N} y_i \cdot \log(p(y_i)) + (1 - y_i) \cdot \log(1 - p(y_i))$$

where N is the number of data in a batch. In our experimental studies, the classifier is a single-layer feedforward neural network with Dropout (Srivastava et al., 2014) and ReLU activation (Nair and Hinton, 2010).

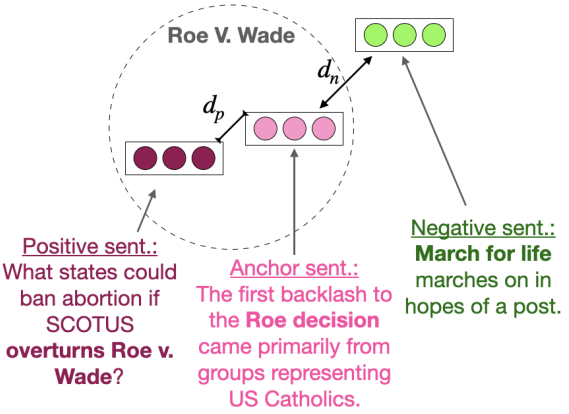

Figure 2: Visualization of the Contrastive Learning Objective. The contrastive learning objective pushes the anchor sentence closer to its positive sentence (decreasing distance to positive example d_p) and farther from its negative sentence (increasing distance from d_n).

Jointly learn framing and political bias. Finally, we combine the learned losses in order to build an embedding space that can group subframes as well as separate the contexts of opposing political stances, i.e., distinguish between liberal and conservative biased news. The final loss that computes the weighted sum of the contrastive learning loss L_c and the classification loss L_{BCE} is:

$$\mathcal{L} = (1 - \alpha) \cdot \mathcal{L}_c + \alpha \cdot \mathcal{L}_{\text{BCE}} \quad (1)$$

where α is a hyperparameter. The default value of α is 0.5.

5 Experimental Setup

We evaluate the performance of our proposed method on the tasks of (1) political bias classification and (2) subframe detection. For the first task, a model makes a binary prediction of whether a given text has a liberal or conservative perspective. We compare the classification performance of our proposed model to the baseline models. Further,

an ablation study on the classification objective is performed, and the classification performance of the pre-trained models on unseen topic data is studied. As for the second task, we test the performance of subframe classification and analyze the effect of the bias classification task on the subframe classification. Finally, we propose a novel topic modeling method for the output embeddings and show experimental results.

5.1 Training Details

We leverage the pre-trained BERT and RoBERTa models as the encoder for our proposed model. We use the following pre-trained models from Hugging Face: `bert-base-cased`, `bert-base-uncased`, and `roberta-base`. They are trained on English texts with 12 layers, 768 hidden units, and 12 attention heads. We use Adam (Kingma and Ba, 2015) optimizer and set learning rates to $\{1e-5, 2e-5\}$, an epoch to 5, and a dropout rate to 0.5. For all models, PyTorch was used for implementation. All experiments are conducted on an Nvidia Quatro RTX 5000, 16 GB memory GPU in a machine with Intel(R) Xeon(R) Silver 4214 CPU @ 2.20GHz.

5.2 Baseline Models

We compare the performance on the political bias classification task with the following baselines.

$\text{BERT}_{\text{cased}}$: A pre-trained BERT with a single-layer feedforward neural network. The BERT encoder is pre-trained with case-sensitive English texts.

$\text{BERT}_{\text{uncased}}$: A pre-trained BERT with a single-layer feedforward neural network. The BERT encoder is pre-trained with case-insensitive English texts.

RoBERTa: A pre-trained RoBERTa model with a single-layer feedforward neural network.

6 Results & Discussion

6.1 Political Bias Classification

We compare the performance of CLoSE against the baseline models described in Section 5.2 on the task of political bias classification. The results are reported in Table 3. Our modeling approach outperforms the baseline models for both gun control and immigration bias classification and showed competitive results for abortion classification. This indicates that jointly modeling the embeddings with a contrastive learning objective improves bias classification.

	Abortion	**Gun**	**Immig.**
$\text{BERT}_{\text{cased}}$	0.790	0.698	0.801
$\text{BERT}_{\text{uncased}}$	0.788	0.722	0.824
RoBERTa	**0.831**	0.720	0.827
$\text{CLoSE}_{\text{BERT-cased}}$	0.821	0.705	**0.841**
$\text{CLoSE}_{\text{BERT-uncased}}$	0.803	0.716	0.818
$\text{CLoSE}_{\text{RoBERTa}}$	0.758	**0.724**	0.829

Table 3: F_1 Scores for Political Bias Classification. Scores in bold indicate the best performing model for that topic.

	Abortion	**Gun**	**Immig.**
$\text{CLoSE}_{\text{Abort.}}$	—	0.722	0.738
$\text{CLoSE}_{\text{Gun}}$	0.749	—	0.735
$\text{CLoSE}_{\text{Immig.}}$	0.747	0.686	—

Table 4: F_1 Scores for Zero-Shot Political Bias Classification. Cells with — indicate the topic subdataset the model was trained on.

$\alpha =$	0	0.5	1
Abortion	0.297	0.821	0.787
Gun Control	0.328	0.705	0.514
Immigration	0.184	0.841	0.734

Table 5: Ablation Study. We compute F_1 scores of the political bias classification, dependent on the values of α. For $\alpha = 0$, only the contrastive learning objective is used. For $\alpha = 1$, only the political bias classification loss is calculated. For brevity, other α values are omitted but reflect similar patterns.

Interestingly, our pre-trained model also performs well in a zero-shot learning setting. That is, our model can predict the political bias of articles from unseen topics. As shown in Table 4, the model $\text{CLoSE}_{\text{Abort.}}$ trained on only the abortion subdataset of the Triplet Dataset is able to predict the political stance of articles from the gun control and immigration subdatasets with F_1 scores above 0.7. Furthermore, the F_1 score for predicting bias of the gun control topic is 0.722. This score is extremely close to the best classification score on the gun subdataset by $\text{CLoSE}_{\text{RoBERTa}}$, as shown in Table 3. The performance of our modeling approach in this zero-shot learning setting indicates the model's ability to learn and transfer latent framing indicators for political bias detection across topics.

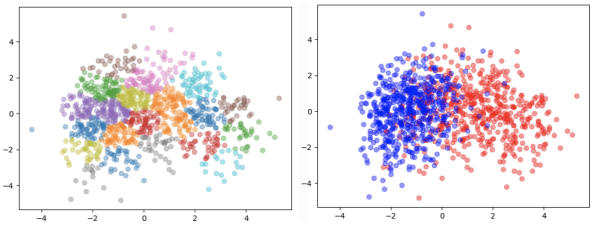

(a) Subframes (b) Political Bias

Figure 3: Visualization of the Embeddings for Immigration. Subfigure (a) shows the clustering of subframe groups. Subfigure (b) is labeled based on the political bias of each article.

	Abortion	**Gun**	**Immig.**
$\text{BERT}_{\text{cased}}$	0.879	0.761	0.734
$\text{CLoSE}_{\text{BERT-cased}}$	0.848	0.853	0.905

Table 6: F_1 Scores of the Subframe Classification Task.

6.2 Ablation Study

To further verify the usefulness of our joint learning objective, we compute F_1 scores of the political bias classification dependent on the values of the α hyperparameter of Equation 1. For $\alpha = 0.5$, we compute the sum of the contrastive objective and classification loss. When $\alpha = 0$, the loss function becomes the contrastive learning objective. On the other hand, when $\alpha = 1$, only the political bias classification loss is considered. The results are shown in Table 5.

We notice that when only the contrastive loss is applied, the model performs poorly on the political bias classification task. As expected, the performance improves when only the political bias classification loss is used (i.e., when $\alpha = 1$). The interesting observation is that when we use *both* the contrastive loss and the bias classification loss, as we have proposed in this work, the F_1 scores are the highest. These results show that incorporating subframe indicators indeed improves political bias classification.

6.3 Subframe Classification

We evaluate whether our proposed model can correctly predict the subframe group of a given sentence. The results are shown in Table 6. We add a classifier to the pre-trained BERT-based encoder and predict the subframe label of a given input. We use $\text{CLoSE}_{\text{BERT-cased}}$, which demonstrated competitive performance across the data in the classification task, as our pre-trained encoder and compare it to the baseline model $\text{BERT}_{\text{cased}}$. For both

		$\text{BERT}_{\text{cased}}$		$\text{CLoSE}_{\text{BERT-cased}}$	
		Acc.	F_1	*Acc.*	F_1
Abort.	50	0.232	0.232	0.300	0.300
	100	0.454	0.454	0.641	0.642
Gun	50	0.265	0.270	0.485	0.486
	100	0.666	0.679	0.756	0.759

Table 7: Accuracy and F_1 Scores of Subframe Classification in Few-shot Learning Setting. CLoSE, pre-trained on immigration data, is fine-tuned on a limited number (50 or 100 triplets) of abortion or gun control data.

	Original Data	$\text{CLoSE}_{\text{BERT-cased}}$
Abortion	0.413 (0.294)	0.347 (0.225)
Gun	0.303 (0.189)	0.198 (0.182)
Immig.	0.490 (0.297)	0.383 (0.281)

Table 8: Average Differences of Subframe Usage Between Political Ideologies. Standard deviations are shown in parentheses.

gun control and immigration subdatasets, our pre-trained model significantly outperforms the baseline model. For the abortion data, the F_1 score of the baseline model is higher; yet the difference in scores was not statistically significant.

Table 7 shows that our model outperforms the baseline model even in few-shot learning. For this experiment, $\text{CLoSE}_{\text{BERT-cased}}$, pre-trained on the immigration subdataset, is fine-tuned with a limited number of unseen data to predict subframes. The size of abortion and gun data is limited to 50 and 100 triplets. For both topics, our model's accuracy and F_1 scores are higher than the baseline, when the data size is 50 and 100 triplets. This result shows that our pre-trained model only requires a small amount of labeled data for fine-tuning, which improves the scalability of data annotation.

In order to understand the influence of political bias on framing, we compute the average differences in subframe usage between biases. Our contrastive objective encourages CLoSE to separate different subframe groups farther apart. Simultaneously, binary political bias information is given as context to the embedding space. We would like to verify whether the bias objective distinguishes amongst subframes that are used by both ideologies in similar usage percentages. For each subframe group, we measure the percentage of its usage by the liberal and conservative media sources and compute the average of the difference in those

percentages. We then use the subframe group classifications of CLoSE and compare them to the true subframe group labels of the original data.

The results are shown in Table 8. The average differences of our model were *smaller* than those of the original data across the datasets. Namely, the overlapping usage of subframes between ideologies is more clearly observed in our model.

Figures 3a and 3b are the visualizations of the embeddings of the immigration data. Principal Component Analysis (PCA) IS applied to reduce dimensions, and the reduced embeddings ARE plotted and labeled with color. Figure 3a displays the clustering of subframe groups, and Figure 3b shows the distribution of the liberal and conservative biased embeddings. Importantly, Figure 3b shows the intersection of subframes between political bias, which aligns with the experimental results of Table 8. The embedding visualizations of the abortion and gun control data can be found in Appendix C.

6.4 Topic Modeling for Frame Extension

Lastly, we propose a topic modeling method that leverages the subframe embeddings output by our proposed CLoSE model to predict new indicators which can be used to extend or generalize the subframes. First, we cluster the framing embeddings, the outputs of our model, with k-means clustering. Then, we use class-based TF-IDF to cluster the framing embeddings and to find topics (Grootendorst, 2022). TF-IDF is a classic keyword extraction method that combines term frequency and inverse document frequency (Joachims, 1996). The TF-IDF score of a token t in a document D is:

$$W_{t,D} = tf_{t,D} \cdot \log \frac{N}{df_t},$$

where $tf_{t,D}$ is the frequency of the token t in the document D, df_t is the total number of documents that contain t, and N is the total number of documents. The class-based TF-IDF generalizes TF-IDF to clusters by considering all documents in a cluster as a single document. It is defined as:

$$W_{t,C} = tf_{t,C} \cdot \log(1 + \frac{A}{tf_t}),$$

where $tf_{t,C}$ is the frequency of the token t in a cluster C, tf_t is the frequency of t across all clusters, and A is the average number of tokens per cluster.

Finally, we merge the clusters with high overlap of keywords, or tokens with high class-based TF-IDF scores. Suppose there are c clusters $C = \{C_1, C_2, \cdots, C_c\}$. For a cluster C_i, we rank tokens according to their class-based TF-IDF scores and choose the top-k number of tokens. Then, we compute the overlap of the tokens across clusters:

$$s(C_i, C_j) = \frac{\text{count}(C_i \cap C_j)}{|C_i|},$$

where $i \neq j$. If $s(C_i, C_j)$ is greater than a threshold β, we merge C_i and C_j. For our experiments, the default values are $k = 50$ and $\beta = 0.4$.

1	reproductive, health, justice, women, freedom
2	federal, funding, abortions, Hyde, Amendment
3	⋆
4	pregnancy, crisis, centers, Pro-Life, women
5	Wade, Roe, overturn, supreme, court
6	rights, anti-abortion, religious, catholic, abortion
7	Affordable, Care, Act, health, insurance
8	control, birth, health, save, prescription
9	Planned, Parenthood, selling, unborn, videos
10	life, unborn, right, baby, child
11	March, Life, Washington, national, Trump
12	industry, abortion, giant, business, profit
13	fetal, tissue, research, illegally, selling
14	Hobby, Lobby, access, coverage, insurance

⋆ Potentially offensive or triggering language has been omitted.

Table 9: Topics of Subframe Embeddings for Abortion.

Table 9 shows the experimental results on the abortion data. We observe the subframe indicators of the original dataset as keywords. For instance, topic 9 has the subframe indicators: "Planned Parenthood." Interestingly, our topic modeling method is able to identify important words that are *not* in the subframe lexicons. Topic 2 has known indicators from the subframe "Abortion Funding" as well as new indicators "Hyde" and "Amendment."[3] This indicates CLoSE's ability to identify new subframe indicators which would allow the development of new frames or unsupervised frame detection. Appendix D presents experimental results for the topics and subdatasets of gun control and immigration.

7 Conclusion

We have presented CLoSE, a framework that incorporates subframes and political bias using contrastive learning and multi-task learning. Our proposed joint objective outperforms baseline models

[3]The Hyde Amendment, which was passed in 1976 by the House, is a legislative provision that prohibits the use of federal funds for performing abortions.

in predicting subframes and political bias. Further, our pre-trained model adapts to zero-shot learning of political bias and few-shot learning of subframes. Finally, we propose a topic modeling method for the subframe embeddings and extract a list of keywords, which can be helpful for future subframe extensions and annotations. We plan to extend this work to embed general policy frames for study in unsupervised settings.

References

Frank R Baumgartner, Suzanna L De Boef, and Amber E Boydstun. 2008. *The decline of the death penalty and the discovery of innocence*. Cambridge University Press.

Vibhu Bhatia, Vidya Prasad Akavoor, Sejin Paik, Lei Guo, Mona Jalal, Alyssa Smith, David Assefa Tofu, Edward Edberg Halim, Yimeng Sun, Margrit Betke, Prakash Ishwar, and Derry Tanti Wijaya. 2021. OpenFraming: Open-sourced tool for computational framing analysis of multilingual data. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 242–250, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

David M Blei, Andrew Y Ng, and Michael I Jordan. 2003. Latent dirichlet allocation. *the Journal of machine Learning research*, 3:993–1022.

Amber E Boydstun, Dallas Card, Justin Gross, Paul Resnick, and Noah A Smith. 2014. Tracking the development of media frames within and across policy issues.

Dallas Card, Amber E. Boydstun, Justin H. Gross, Philip Resnik, and Noah A. Smith. 2015. The media frames corpus: Annotations of frames across issues. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 2: Short Papers)*, pages 438–444, Beijing, China. Association for Computational Linguistics.

Kenneth Church and Patrick Hanks. 1990. Word association norms, mutual information, and lexicography. *Computational linguistics*, 16(1):22–29.

Carroll Doherty, Jocelyn Kiley, Nida Asheer, and Calvin Jordan. 2021. Beyond red vs. blue: The political typology. *Pew Research Center*.

Robert M Entman. 1993. Framing: Towards clarification of a fractured paradigm. *McQuail's reader in mass communication theory*, pages 390–397.

Robert M Entman and Andrew Rojecki. 1993. Freezing out the public: Elite and media framing of the us anti-nuclear movement.

Anjalie Field, Doron Kliger, Shuly Wintner, Jennifer Pan, Dan Jurafsky, and Yulia Tsvetkov. 2018. Framing and agenda-setting in Russian news: a computational analysis of intricate political strategies. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 3570–3580, Brussels, Belgium. Association for Computational Linguistics.

Maarten Grootendorst. 2022. Bertopic: Neural topic modeling with a class-based tf-idf procedure. *arXiv preprint arXiv:2203.05794*.

Lei Guo, Chris J Vargo, Zixuan Pan, Weicong Ding, and Prakash Ishwar. 2016. Big social data analytics in journalism and mass communication: Comparing dictionary-based text analysis and unsupervised topic modeling. *Journalism & Mass Communication Quarterly*, 93(2):332–359.

Pere-Lluís Huguet Cabot, Verna Dankers, David Abadi, Agneta Fischer, and Ekaterina Shutova. 2020. The Pragmatics behind Politics: Modelling Metaphor, Framing and Emotion in Political Discourse. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 4479–4488, Online. Association for Computational Linguistics.

Shanto Iyengar. 1994. *Is anyone responsible?: How television frames political issues*. University of Chicago Press.

Jakob D Jensen, Courtney L Scherr, Natasha Brown, Christina Jones, Katheryn Christy, and Ryan J Hurley. 2014. Public estimates of cancer frequency: cancer incidence perceptions mirror distorted media depictions. *Journal of health communication*, 19(5):609–624.

Thorsten Joachims. 1996. A probabilistic analysis of the rocchio algorithm with tfidf for text categorization. Technical report, Carnegie-mellon univ pittsburgh pa dept of computer science.

Kristen Johnson and Dan Goldwasser. 2016. "all I know about politics is what I read in Twitter": Weakly supervised models for extracting politicians' stances from Twitter. In *Proceedings of COLING 2016, the 26th International Conference on Computational Linguistics: Technical Papers*, pages 2966–2977, Osaka, Japan. The COLING 2016 Organizing Committee.

Kristen Johnson, Di Jin, and Dan Goldwasser. 2017a. Leveraging behavioral and social information for weakly supervised collective classification of political discourse on Twitter. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 741–752, Vancouver, Canada. Association for Computational Linguistics.

Kristen Johnson, I-Ta Lee, and Dan Goldwasser. 2017b. Ideological phrase indicators for classification of political discourse framing on Twitter. In *Proceedings of the Second Workshop on NLP and Computational*

Social Science, pages 90–99, Vancouver, Canada. Association for Computational Linguistics.

Grzegorz Kapuściński and Barry Richards. 2016. News framing effects on destination risk perception. *Tourism Management*, 57:234–244.

Johannes Kiesel, Maria Mestre, Rishabh Shukla, Emmanuel Vincent, Payam Adineh, David Corney, Benno Stein, and Martin Potthast. 2019. SemEval-2019 task 4: Hyperpartisan news detection. In *Proceedings of the 13th International Workshop on Semantic Evaluation*, pages 829–839, Minneapolis, Minnesota, USA. Association for Computational Linguistics.

Diederik P. Kingma and Jimmy Ba. 2015. Adam: A method for stochastic optimization. In *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*.

Bingjie Liu and Lori Pennington-Gray. 2015. Bed bugs bite the hospitality industry? a framing analysis of bed bug news coverage. *Tourism Management*, 48:33–42.

Amy Mitchell, Jeffrey Gottfried, Jocelyn Kiley, and Katerina Eva Matsa. 2014. Political polarization & media habits.

Vinod Nair and Geoffrey E Hinton. 2010. Rectified linear units improve restricted boltzmann machines. In *Icml*.

Matthew C Nisbet, Dominique Brossard, and Adrianne Kroepsch. 2003. Framing science: The stem cell controversy in an age of press/politics. *Harvard International Journal of Press/Politics*, 8(2):36–70.

Zhongdang Pan and Gerald M Kosicki. 1993. Framing analysis: An approach to news discourse. *Political communication*, 10(1):55–75.

Nils Reimers and Iryna Gurevych. 2019. Sentence-BERT: Sentence embeddings using Siamese BERT-networks. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 3982–3992, Hong Kong, China. Association for Computational Linguistics.

Shamik Roy and Dan Goldwasser. 2020. Weakly supervised learning of nuanced frames for analyzing polarization in news media. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 7698–7716, Online. Association for Computational Linguistics.

Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. 2014. Dropout: a simple way to prevent neural networks from overfitting. *The journal of machine learning research*, 15(1):1929–1958.

Nayda Terkildsen and Frauke Schnell. 1997. How media frames move public opinion: An analysis of the women's movement. *Political research quarterly*, 50(4):879–900.

Isidora Tourni, Lei Guo, Taufiq Husada Daryanto, Fabian Zhafransyah, Edward Edberg Halim, Mona Jalal, Boqi Chen, Sha Lai, Hengchang Hu, Margrit Betke, Prakash Ishwar, and Derry Tanti Wijaya. 2021. Detecting frames in news headlines and lead images in U.S. gun violence coverage. In *Findings of the Association for Computational Linguistics: EMNLP 2021*, pages 4037–4050, Punta Cana, Dominican Republic. Association for Computational Linguistics.

Oren Tsur, Dan Calacci, and David Lazer. 2015. A frame of mind: Using statistical models for detection of framing and agenda setting campaigns. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 1629–1638, Beijing, China. Association for Computational Linguistics.

Arjen Van Dalen, Claes de Vreese, and Erik Albæk. 2017. Economic news through the magnifying glass: How the media cover economic boom and bust. *Journalism Studies*, 18(7):890–909.

Appendix A Subframes

Table 10 presents the list of subframes for each topic. As subframes are issue-specific subclassifications of the general policy frames, the frames are written in bold, and subframes that fall under a corresponding frame are displayed below the frame. This table is from Roy and Goldwasser (2020).

Appendix B Top-3 Subframes

Table 11 shows the top-3 subframes used across the period for the topic of gun control. Similarly, Table 12 is the results for the topic of immigration. High overlapping of subframes used by liberal and conservative media is observed in both data. As for the gun control data, all top-3 subframes from the liberal and conservative articles from 2020 to 2022 are identical. Similarly, the top-2 subframes from the liberal and conservative news from 2020 to 2022 are the same for the immigration data. We did not extract the top-3 subframes of the immigration data before 2010 because all the articles in the immigration dataset were from liberal media sources.

Appendix C Embedding Visualization

Figures 4 and 5 are the embedding visualizations of the abortion data and gun control data, respectively. The dimensions of the output embeddings of our proposed method are reduced with PCA, and the reduced embeddings are mapped to 2D plots. Subfigure (a) is labeled with color to show the clustering of subframes. Subfigure (b) displays the liberal embeddings in blue and the conservative embeddings in red. As we observed in the embedding plots of the abortion data, Figures 4b and 5b display a high percentage of overlap between the subframes used by liberal and conservative articles.

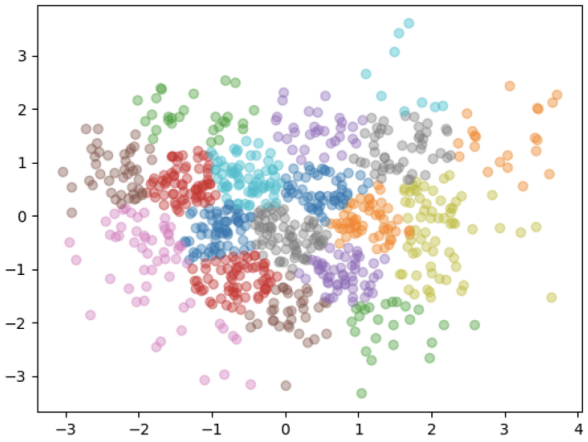

(a) Subframes

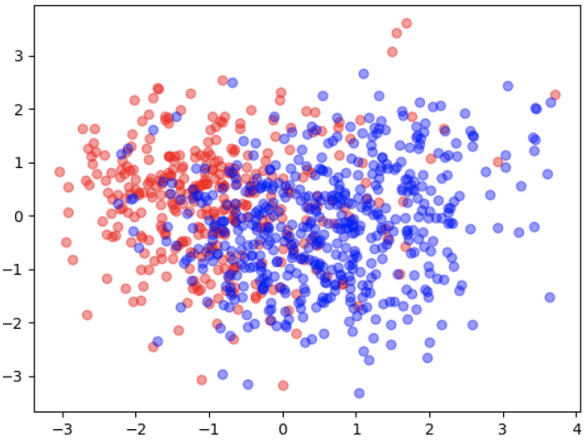

(b) Political Bias

Figure 4: Visualization of the Embeddings of the Abortion Dataset. Subfigure (a) shows the clustering of subframe groups. Subfigure (b) is labeled based on the political bias of each article.

Appendix D Topic Modeling of Framing Embeddings

Table 13 shows the topics of the subframe embeddings of the gun control data, and Table 14 shows those for the immigration data. After merging the topics with high overlapping keywords, there are 12 and 15 topics for the gun control and immigration data, respectively.

As for the topics of the gun control data, names and locations related to mass shootings are observed in topics 10 and 11. In topic 6, the words "Islamic" and "York" from New York appear along

Abortion	Gun Control	Immigration
Economic:	**Economic**:	**Economic**:
- Health Care	- Gun Buyback	- Minimum Wage
- Abort. Provider	Program	- Salary Stagnation
- Abortion Funding	- Gun Business	- Wealth Gap
Fairness & Equality:	**Capacity & Resources:**	- Cheap Labor Availability
- Reproduction Right	- School Safety	- Taxpayer Money
- Right of Human Life	**Cultural Identity:**	**Crime & Punishment:**
Legality, Constitution	- White Identity	- Deportation: Illegal
ality, Jurisdiction:	- Person of Color Identity	Immigrants
- Hobby Lobby	**Legality, Constitution**	- Deportation: In General
- Late Term Abortion	**ality, Jurisdiction:**	- Detention
- Roe V. Wade	- Ban on Handgun	- Terrorism
Crime & Punishment:	- Second Amendment	**Security & Defense:**
- Stem Cell Research	- Concealed Carry	- Border Protection
- Sale of Fetal Tissue	Reciprocity Act	**Legality, Const., Juri.**
- Sexual Assault Victims	- Gun Control to	- Asylum
Health & Safety:	Restrain Violence	- Refugee
- Birth Control	**Crime & Punishment:**	- Birth citizenship &
Morality:	- Illegal Gun	14th Amendment
- Sanctity of Life	- Gun Show Loophole	**Policy Pres. & Eval.:**
- Women Freedom	**Security & Defense**	- Amnesty
Quality of Life:	- Background Check	- Dream Act
- Planned Parenthood	- Terrorist Attack	- Family Separation
- Pregnancy Centers	**Health & Safety:**	Policy
- Life protection	- Gun Research	- DACA
Public Sentiment:	- Mental Health	**Fairness & Equality:**
- Pro-Life	- Gun Homicide	- Racism & Xenophobia
- Anti-Abortion	**Policy Pres. & Eval.:**	- Merit Based Immigration
- Pro-Choice	- Assault Weapon	- Human Right
	Morality:	**Cultural Identity:**
	- Right to Self-Defense	- Racial Identity
	- Stop Gun Crime	- Born identity

Table 10: Subframes of the Three Topics: Abortion, Gun Control and Immigration. Frames are written in bold.

		Top-1	Top-2	Top-3
2020-2022	Left	Gun Business Industry	Conc. Carry Recip. Act	Second Amendment
	Right	Gun Business Industry	Conc. Carry Recip. Act	Second Amendment
2018-2020	Left	Background Check	School Safety	Assault Weapon
	Right	Background Check	Assault Weapon	Second Amendment
2016-2018	Left	Gun Business Industry	Terrorist Attack	Conc. Carry Recip. Act
	Right	Gun Business Industry	Terrorist Attack	Assault Weapon
2010-2016	Left	Gun Business Industry	Background Check	Mental Health
	Right	Terrorist Attack	Gun Show Loophole	Gun Business Industry
-2010	Left	Second Amendment	Gun Business Industry	Terrorist Attack
	Right	Gun Business Industry	Illegal Gun	Second Amendment

Table 11: Top-3 Subframe Indicators of the Gun Control Triplet Dataset.

		Top-1	Top-2	Top-3
2020-2022	Left	Asylum	Detention	Human Rights
	Right	Asylum	Detention	Racial Identity
2018-2020	Left	Asylum	DACA	Human Rights
	Right	Asylum	DACA	Amnesty
2016-2018	Left	DACA	Human Rights	Racial Identity
	Right	DACA	Amnesty	Dream Act
2010-2016	Left	DACA	Human Rights	Detention
	Right	Birth Cit. & 14th Amen.	DACA	Dream Act

Table 12: Top-3 Subframe Indicators of the Immigration Triplet Dataset.

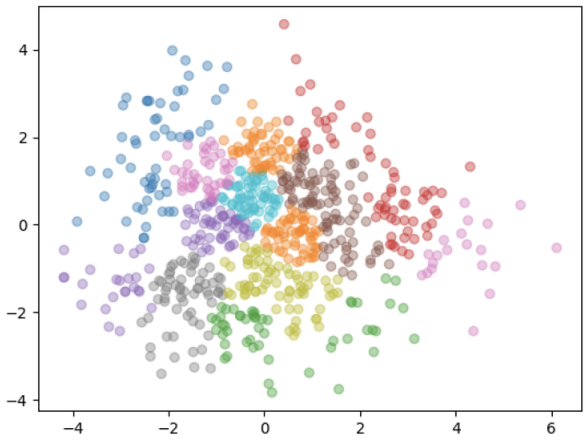

(a) Subframes

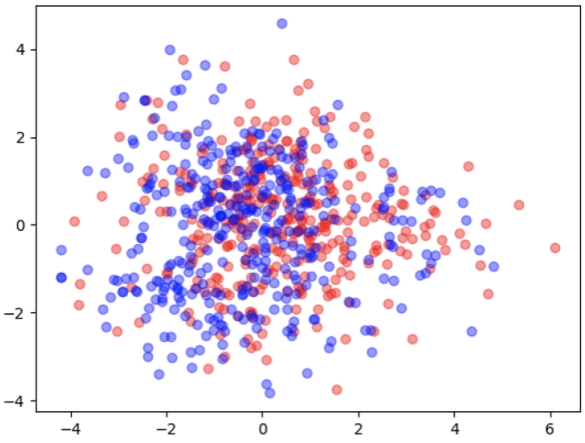

(b) Political Bias

Figure 5: Visualization of the Embeddings of the Gun Control Dataset. Subfigure (a) shows the clustering of subframe groups. Subfigure (b) is labeled based on the political bias of each article.

with the subframe indicators related to "terrorism," which imply the co-ocurrence of a specific religion with a place.

Similarly, Table 13 shows specific names and places such as "Germany," "Mexico," and "ACLU," which is an acronym for the American Civil Liberties Union that fights for civil rights, including the rights of immigrants. The names of the two former presidents are also found: "Trump" in topic 14 along with the keywords "build" and "wall" and "Obama" in a keyword "Obama-era" in topic 5 with keywords associated with the Dream Act.

1	Amendment, Second, rights, protect, individual
2	background, checks, NICS, FBI, criminal
3	industry, manufacturers, business, selling
4	rifles, gun, firearms, semiautomatic, automatic
5	illegal, guns, possessions, convicted, dealers
6	terrorism, terrorist, Islamic, York, Trump
7	ill, mentally, illness, violence, mental
8	abusers, shootings, victims, health, age
9	ban, handgun, Concealed, Carry, Reciprocity
10	school, violence, students, Parkland, Florida
11	contempt, Orlando, Mateen, Black, Americans
12	Buyback, program, deaths, restrictions, mandatory

Table 13: Topics of Subframe Embeddings of the Gun Control Data.

1	detention, center, facilities, ICE, children
2	Amnesty, plan, seekers, refugee, Germany
3	Families, Separation, Policy, Trump, parents
4	DACA, Democrats, childhood, arrivals, program
5	Act, Dream, dreamers, children, Obama-era
6	asylum, migrants, foreign, illegal, caravan
7	rights, human, Civil, Liberties, Union, ACLU
8	black, brown, undocumented, immigrants, racial
9	labor, cheap, workers, wages, manual
10	illegal, deported, alien, deportation, persons
11	white, identity, supremacist, nationalist, racist
12	terrorist, threat, border, terrorism, terror
13	birthright, citizenship, Amendment, 14th, Anton
14	build, wall, Trump, Republican, Mexico
15	minimum, wage, inequlaity, income, raise

Table 14: Topics of Subframe Embeddings of the Immigration Data.

Appendix E Ethical Considerations

The Framing Triplet Dataset is an extension of the existing public dataset by Kiesel et al. (2019) and Roy and Goldwasser (2020). Additionally collected documents are also from news texts that are free to the public. Hence, the text corpus does not contain private or sensitive information.

The code for the proposed method is open to the public and can be used to study the impact of news framing on public opinion. We do not anticipate any significant risks of deployment. Still we urge users not to use this research for malicious intentions.

Grammatical Error Correction: Are We There Yet?

Muhammad Reza Qorib and **Hwee Tou Ng**
Department of Computer Science, National University of Singapore
{mrqorib, nght}@comp.nus.edu.sg

Abstract

There has been much recent progress in natural language processing, and grammatical error correction (GEC) is no exception. We found that state-of-the-art GEC systems (T5 and GECToR) outperform humans by a wide margin on the CoNLL-2014 test set, a benchmark GEC test corpus, as measured by the standard $F_{0.5}$ evaluation metric. However, a careful examination of their outputs reveals that there are still classes of errors that they fail to correct. This suggests that creating new test data that more accurately measure the true performance of GEC systems constitutes important future work.

1 Introduction

Grammatical Error Correction (GEC) is a task that has many real-world applications, such as proofreading, assisting language learners (Knutsson et al., 2003; Chollampatt et al., 2016; Nadejde and Tetreault, 2019), assisting children with developmental language disorder (Balthazar et al., 2020), etc. In order to measure the accuracy of a GEC system, multiple evaluation metrics have been proposed in the past. Ever since the CoNLL-2014 shared task (Ng et al., 2014), the $F_{0.5}$ metric has been used as the standard evaluation metric for GEC. The $F_{0.5}$ score has been found to have a better correlation with human judgment compared to F_1 and other metrics (Grundkiewicz et al., 2015; Napoles et al., 2015; Chollampatt and Ng, 2018).

As $F_{0.5}$ is a reference-based metric, its computation relies on comparing a system's output sentences to human's corrected sentences (called references henceforth). Typically, there are multiple ways to correct an input sentence. For example, the CoNLL-2014 (Ng et al., 2014) official test set contains 2 references for each input sentence, and the BEA-2019 (Bryant et al., 2019) test set contains 5 references. Choshen and Abend (2018) even predict that a short sentence may have more than 1,000 valid corrections. Thus, this evaluation scheme tends to underestimate the performance of a GEC system. Rozovskaya and Roth (2021) have reported that GEC systems can obtain higher scores if evaluated using references that are closer to a system's outputs.

The limited references mean that humans may also not reach 100% $F_{0.5}$ performance, because their corrections need not be the same as the references. Bryant and Ng (2015) was the first to attempt to measure human performance on the CoNLL-2014 test set by adding 8 new references for each sentence, so that each sentence has 10 references. Then, they calculate each annotator's performance by comparing his corrections to the other 9 annotators. The average $F_{0.5}$ score of 72.58% from all annotators is then considered as the estimated human-level performance. To compare a GEC system's performance against human performance, a similar procedure is applied by taking the average of the system's performance on the 10 sets of 9-annotator references.

Recently, much progress has been made in natural language processing. For example, for some question answering datasets (such as SQuAD (Rajpurkar et al., 2016) and CoQA (Reddy et al., 2019)) and natural language understanding test suites (SuperGLUE (Wang et al., 2019)), superhuman performance has been reported where the performance of the best NLP system exceeds human performance. A natural question to ask is: how does the performance of the latest state-of-the-art (SOTA) GEC systems compare to human performance? In this paper, we first attempt to answer this question. We found that state-of-the-art GEC systems (T5 (Rothe et al., 2021) and GECToR (Omelianchuk et al., 2020)) outperform humans by a wide margin on the CoNLL-2014 test set, as measured by the $F_{0.5}$ metric. However, a careful examination of their outputs reveals that there are still classes of errors that they fail to cor-

Proceedings of the 29th International Conference on Computational Linguistics, pages 2794–2800
October 12–17, 2022.

rect. This suggests that creating new test data that more accurately measure the true performance of GEC systems constitutes important future work.

2 Are We There Yet?

At the time of writing this paper, the best published GEC system using a sequence-to-sequence approach is T5-XXL, and the best published GEC system using a sequence-tagging approach is the ensemble of GECToR XLNet and GECToR RoBERTa. In this paper, we use both systems, but with T5-XXL replaced by T5-Large due to resource constraints, since T5-Large has 770M parameters while T5-XXL has 11B parameters. Our evaluation is based on the scheme proposed by (Bryant and Ng, 2015), using 10 sets of human annotations of the CoNLL-2014 official test set. The $F_{0.5}$ scores of T5 and GECToR are shown in Table 1. We find that both systems outperform humans by a wide margin – T5 outperforms humans by 6.05 points and GECToR outperforms humans by 8.33 points.

Model	$F_{0.5}$ Score
Human	72.58
T5	78.63
GECToR	**80.91**

Table 1: The performance of SOTA GEC systems in comparison to human performance.

However, even though the scores in Table 1 show that SOTA GEC systems have outperformed humans, when we check the distribution of per-sentence $F_{0.5}$ scores, we find that both T5 and GECToR generate sentences with a zero $F_{0.5}$ score even comparing to 9 references. Specifically, on the CoNLL-2014 test set, T5 (GECToR) completely fails to correct 9.1% (12.8%) of the sentences (Figure 1). On the BEA-2019 development set, the proportion is higher, with 27% of the sentences resulting in a zero $F_{0.5}$ score for T5, and 29.9% for GECToR (Figure 2)[1].

3 Current Weaknesses

To understand what causes the low scores of the top-performing GEC systems, we examine a sample of 100 sentences from the systems' outputs on the BEA-2019 development set, starting from sentences with the lowest $F_{0.5}$ score. We found that even though T5 and GECToR generally produced good corrections, they also sometimes made obvious mistakes that humans will not make.

3.1 Unnatural Phrases

Source	The first place was gotting by us .
T5	The first place was **got** by us .
GECToR	The first place was gotting by us .
Target	**We won first place** .

Table 2: An example of GEC systems failing to fix unnatural phrases.

In the example in Table 2, GECToR completely fails to produce any corrections, with the misspelling uncorrected. T5 successfully makes the sentence grammatical by applying the appropriate edit, but it fails to make it sound natural. Similarly, both systems also fail to correct the sentence in Table 3. One possible reason is that this kind of sentences where the target sentence corrects unnatural phrases happen rarely in the training data, since the human annotators are expected to make minimal edits to make the sentence grammatically correct, instead of making the sentence sounds more natural, like the goal of the JFLEG dataset (Napoles et al., 2017).

Source	I like personality with childlike , so I like children .
T5	I like personality with childlike , so I like children .
GECToR	I like personality with childlike , so I like children .
Target	I like **childlike people**, so I like children .

Table 3: Another example of GEC systems failing to fix unnatural phrases.

3.2 Sentence Structure

In the example in Table 4, the systems fail to detect that the object of the sentence consists of multiple items. GEC systems sometimes fail to detect the subject-verb relationship when the subject or object is in a form of a complex phrase instead of a single word. This observation is in line with Mita and Yanaka (2021) who reported that the standard Transformer-based GEC model has difficulties in fixing subject-verb agreement on error patterns that do not appear in the training data, even in simple settings with limited vocabulary and syntax.

[1]However, note that since the BEA-2019 development set has only one reference per sentence, there is a greater chance of underestimating system performance.

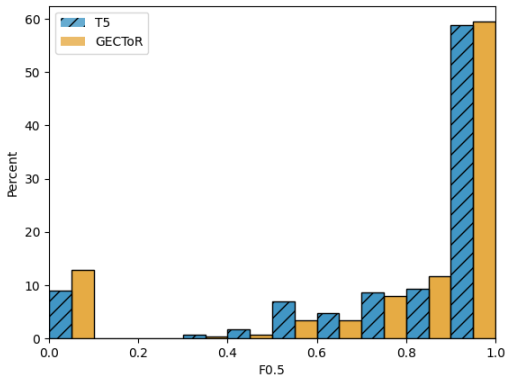

Figure 1: Distribution of per-sentence $F_{0.5}$ scores on the CoNLL-2014 test set.

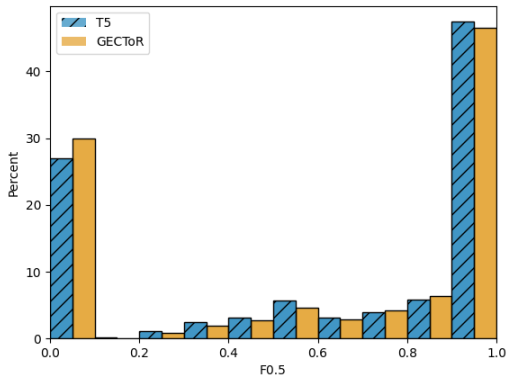

Figure 2: Distribution of per-sentence $F_{0.5}$ scores on the BEA-2019 development set.

Source	There are a little kitchen , a great bedroom , a bathroom with shower but without bath and a cool living - room .
T5	There **is** a little kitchen , a great bedroom , a bathroom with shower but without bath and a cool living - room .
GECToR	There **is** a little kitchen , a great bedroom , a bathroom with shower but without bath and a cool living - room .
Target	There are a little kitchen , a great bedroom , a bathroom with shower but without bath and a cool living - room .

Table 4: An example of GEC systems failing to detect the sentence structure.

Source	The implications for the beef industry could be rather serious , where everybody to boycott beef products .
T5	The implications for the beef industry could be rather serious , where everybody **boycotts** beef products .
GECToR	The implications for the beef industry could be rather serious , where everybody to boycott beef products .
Target	The implications for the beef industry could be rather serious , **were** everybody to boycott beef products .

Table 5: An example of GEC systems failing to understand a sentence's meaning.

3.3 Sentence Comprehension

The example in Table 5 requires a full comprehension of the sentence to make the right correction. Another example in Table 6 also shows that the GEC systems fail to understand that getting along with his or her companion is what the writer wants to convey, given the context of wanting to enjoy a trip.

3.4 Error Rates

On the BEA-2019 development set, we analyze the effect of error rate (the percentage of erroneous tokens in a source sentence) on GEC systems' per-sentence $F_{0.5}$ score. The error rate is computed as the number of tokens in a source sentence that are to be deleted or substituted by the edits in the gold reference, divided by the total number of tokens in the sentence. Next, we remove the outliers, which are sentences with error rates $\pm$ 3 standard deviations from the mean, and we end up removing 1.64% of the sentences. The relationship between the per-sentence $F_{0.5}$ score of a sentence and its error rate (rounded to the nearest 0.05) is presented in Figure 3. We observe that GEC systems perform better on sentences with low error rates (less than 0.05).

3.5 Long Sentences

We also analyze the effect of sentence length on a GEC model's performance. On the BEA-2019 development set, we count the number of words in each source sentence and obtain the per-sentence $F_{0.5}$ score of the sentence. Then, we group the sentences based on sentence length in step size of 5 words (0–4 words, 5–9 words, etc). We apply the same outlier removal procedure in Section 3.4, eliminating 1.25% of the sentences.

We observe that top GEC systems have difficulties in generating accurate corrections for long sen-

Source	In order to enjoy a trip to Mexico I suggest that the traveler find a manner to get alone with his or her companion.
T5	In order to enjoy a trip to Mexico **,** I suggest that the traveler find a **way** to get alone with his or her companion.
GECToR	In order to enjoy a trip to Mexico **,** I suggest that the traveler find a manner to get alone with his or her companion.
Target	In order to enjoy a trip to Mexico **,** I suggest that the traveler **finds** a **way** to get **along** with his or her companion.

Table 6: Another example of GEC systems failing to understand a sentence's meaning.

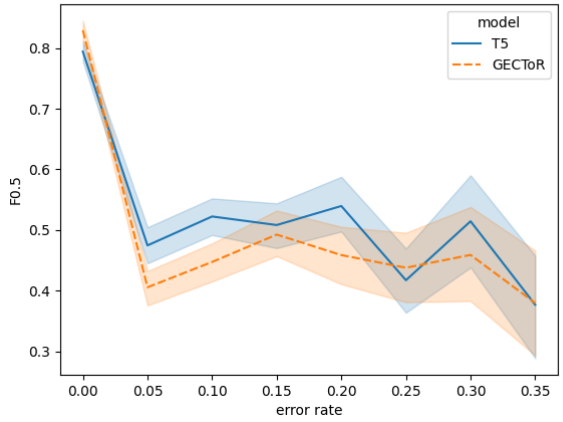

Figure 3: Per-sentence $F_{0.5}$ scores of top GEC models for different error rates. The straight line segments connect the mean values and the shaded region denotes the variance.

tences. As shown in Figure 4, the per-sentence $F_{0.5}$ shows a downward trend with increasing sentence length.

3.6 Cross-Sentence Context

Most of the current GEC systems are working at the sentence level. Cross-sentence GEC has not been given enough attention, and only a few papers have pursued this direction (Chollampatt et al., 2019; Yuan and Bryant, 2021). However, some sentences indisputably require cross-sentence context to correctly fix them. For example, the correction for the sentence in Table 7 requires knowing the context to realize that there is a misspelling in the sentence. Typical errors that require cross-sentence context to correct include pronoun agreement and tense agreement (Table 8). However, the errors can

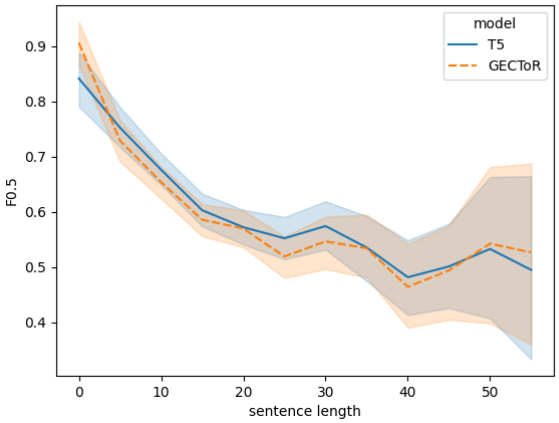

Figure 4: Per-sentence $F_{0.5}$ scores of top GEC models for different sentence lengths. The straight line segments connect the mean values and the shaded region denotes the variance.

be of many kinds, in the form of sentence structure, word choice, misspelling, and many more.

3.7 Adapting to a Different Domain

The common GEC training corpora, such as NUCLE (Dahlmeier et al., 2013), FCE (Yannakoudakis et al., 2011), Lang-8 (Mizumoto et al., 2011), and W&I+LOCNESS (Granger, 1998; Yannakoudakis et al., 2018), and GEC test corpora, such as CoNLL-2014, BEA-2019, and JFLEG, originated from essays written by English as a second language (ESL) authors. Those datasets are more situated in an academic setting. To know how GEC models perform in a different domain, we evaluate the models on the CWEB datasets (Flachs et al., 2020) without additional training (Table 9)

The CWEB dataset is made from website data to represent language correction "in the wild". The CWEB dataset consists of two subsets: CWEB-S which has source sentences from government, educational institution, and museum websites, and CWEB-G from general websites. When tested on CWEB, the $F_{0.5}$ scores of both T5 and GECToR drop drastically from their CoNLL-2014 test scores, especially for T5. These reductions indicate that current top GEC models are not robust to domain change, even though GEC models are expected to be able to correct all kinds of sentences.

4 Moving Forward

As we have seen examples of how SOTA GEC models fail to correct sentences that are easy for hu-

Context	Is there a future for Privately owned cars ? To be honest , I am not sure . Although privately oned cars are more and more popular , ...
Source	I would say that most probably private care are not " sustainables " in the long term , ...
T5	I would say that most probably private care **is** not " **sustainable** " in the long term , ...
GECToR	I would say that **,** most probably **,** private care **is** not " **sustainable** " in the long term , ...
Target	I would say that **,** most probably **,** private **cars** are not " **sustainable** " in the long term , ...

Table 7: An example of a sentence that requires cross-sentence context to correct.

Context	For example there are girl she is in my class she is beautiful . I love her look . Her eyes looks as the sun .
Source	But she is ignored me .
T5	But she ~~is~~ ignored me .
GECToR	But she ~~is~~ ignored me .
Target	But she ~~is~~ **ignores** me ..

Table 8: Another example of a sentence that requires cross-sentence context to correct.

mans, we can conclude that GEC models have not actually outperformed humans in practice. However, the current top GEC systems already reach high scores on the standard benchmarks. Thus, we argue that it is important to create a new test set that contains more sentences that pose challenges to SOTA GEC models but can be easily corrected by humans. The test set may emphasize on sentences with complex noun phrase subject/object, sentences that require full comprehension to be corrected, sentences that contain many grammatical errors, sentences that are long, sentences that require cross-sentence context, etc. With a harder test set, we can then more clearly assess how far we are from considering GEC as a solved task.

5 Conclusion

In this paper, we have reported that the top GEC systems using the sequence-to-sequence approach (T5) and sequence-tagging approach (GECToR) have produced $F_{0.5}$ scores that exceed that of humans, as measured by Bryant and Ng (2015). Based on qualitative analysis of their outputs, we conclude that even though GECToR and T5 achieve $F_{0.5}$ scores higher than that of human, they have not outperformed humans in practice as they still fail to correct sentences that can be easily corrected by humans. We also report our qualitative analysis on the weaknesses of current top GEC models, which point to directions for future research. Lastly, we argue that a new test set that emphasizes harder sentences is needed to evaluate the progress of GEC as a field.

Test data	T5	GECToR
CoNLL-2014	65.07	66.05
CWEB-G Dev	38.91	40.79
CWEB-G Test	39.80	42.67
CWEB-S Dev	27.37	37.63
CWEB-S Test	28.51	33.07

Table 9: The performance of T5 and GECToR on the CoNLL-2014 test set and CWEB.

Acknowledgements

This research is supported by the National Research Foundation, Singapore under its AI Singapore Programme (AISG Award No: AISG-RP-2019-014). The computational work for this article was partially performed on resources of the National Supercomputing Centre, Singapore (https://www.nscc.sg).

References

Catherine H Balthazar, Susan Ebbels, and Rob Zwitserlood. 2020. Explicit grammatical intervention for developmental language disorder: Three approaches. *Language, Speech, and Hearing Services in Schools*, 51:226–246.

Christopher Bryant, Mariano Felice, Øistein E. Andersen, and Ted Briscoe. 2019. The BEA-2019 shared task on grammatical error correction. In *Proceedings of the Fourteenth Workshop on Innovative Use of NLP for Building Educational Applications*, pages 52–75, Florence, Italy. Association for Computational Linguistics.

Christopher Bryant and Hwee Tou Ng. 2015. How far are we from fully automatic high quality grammatical error correction? In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 697–707, Beijing, China. Association for Computational Linguistics.

Shamil Chollampatt, Duc Tam Hoang, and Hwee Tou Ng. 2016. Adapting grammatical error correction based on the native language of writers with neural network joint models. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 1901–1911, Austin, Texas. Association for Computational Linguistics.

Shamil Chollampatt and Hwee Tou Ng. 2018. A reassessment of reference-based grammatical error correction metrics. In *Proceedings of the 27th International Conference on Computational Linguistics*, pages 2730–2741, Santa Fe, New Mexico, USA. Association for Computational Linguistics.

Shamil Chollampatt, Weiqi Wang, and Hwee Tou Ng. 2019. Cross-sentence grammatical error correction. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 435–445, Florence, Italy. Association for Computational Linguistics.

Leshem Choshen and Omri Abend. 2018. Inherent biases in reference-based evaluation for grammatical error correction. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 632–642, Melbourne, Australia. Association for Computational Linguistics.

Daniel Dahlmeier, Hwee Tou Ng, and Siew Mei Wu. 2013. Building a large annotated corpus of learner English: The NUS corpus of learner English. In *Proceedings of the Eighth Workshop on Innovative Use of NLP for Building Educational Applications*, pages 22–31, Atlanta, Georgia. Association for Computational Linguistics.

Simon Flachs, Ophélie Lacroix, Helen Yannakoudakis, Marek Rei, and Anders Søgaard. 2020. Grammatical error correction in low error density domains: A new benchmark and analyses. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 8467–8478, Online. Association for Computational Linguistics.

Sylviane Granger. 1998. The computer learner corpus: A versatile new source of data for SLA research. *Learner English on Computer*, pages 3–18.

Roman Grundkiewicz, Marcin Junczys-Dowmunt, and Edward Gillian. 2015. Human evaluation of grammatical error correction systems. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 461–470, Lisbon, Portugal. Association for Computational Linguistics.

Ola Knutsson, Teresa Cerrato Pargman, and Kerstin Severinson Eklundh. 2003. Transforming grammar checking technology into a learning environment for second language writing. In *Proceedings of the HLT-NAACL 03 Workshop on Building Educational Applications Using Natural Language Processing*, pages 38–45.

Masato Mita and Hitomi Yanaka. 2021. Do grammatical error correction models realize grammatical generalization? In *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*, pages 4554–4561, Online. Association for Computational Linguistics.

Tomoya Mizumoto, Mamoru Komachi, Masaaki Nagata, and Yuji Matsumoto. 2011. Mining revision log of language learning SNS for automated Japanese error correction of second language learners. In *Proceedings of 5th International Joint Conference on Natural Language Processing*, pages 147–155, Chiang Mai, Thailand. Asian Federation of Natural Language Processing.

Maria Nadejde and Joel Tetreault. 2019. Personalizing grammatical error correction: Adaptation to proficiency level and L1. In *Proceedings of the 5th Workshop on Noisy User-generated Text (W-NUT 2019)*, pages 27–33, Hong Kong, China. Association for Computational Linguistics.

Courtney Napoles, Keisuke Sakaguchi, Matt Post, and Joel Tetreault. 2015. Ground truth for grammatical error correction metrics. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 2: Short Papers)*, pages 588–593, Beijing, China. Association for Computational Linguistics.

Courtney Napoles, Keisuke Sakaguchi, and Joel Tetreault. 2017. JFLEG: A fluency corpus and benchmark for grammatical error correction. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 2, Short Papers*, pages 229–234, Valencia, Spain. Association for Computational Linguistics.

Hwee Tou Ng, Siew Mei Wu, Ted Briscoe, Christian Hadiwinoto, Raymond Hendy Susanto, and Christopher Bryant. 2014. The CoNLL-2014 shared task on grammatical error correction. In *Proceedings of the Eighteenth Conference on Computational Natural Language Learning: Shared Task*, pages 1–14, Baltimore, Maryland. Association for Computational Linguistics.

Kostiantyn Omelianchuk, Vitaliy Atrasevych, Artem Chernodub, and Oleksandr Skurzhanskyi. 2020. GECToR – grammatical error correction: Tag, not rewrite. In *Proceedings of the Fifteenth Workshop on Innovative Use of NLP for Building Educational Applications*, pages 163–170, Seattle, WA, USA → Online. Association for Computational Linguistics.

Pranav Rajpurkar, Jian Zhang, Konstantin Lopyrev, and Percy Liang. 2016. SQuAD: 100,000+ questions for machine comprehension of text. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 2383–2392, Austin, Texas. Association for Computational Linguistics.

Siva Reddy, Danqi Chen, and Christopher D. Manning. 2019. CoQA: A conversational question answering

challenge. *Transactions of the Association for Computational Linguistics*, 7:249–266.

Sascha Rothe, Jonathan Mallinson, Eric Malmi, Sebastian Krause, and Aliaksei Severyn. 2021. A simple recipe for multilingual grammatical error correction. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 2: Short Papers)*, pages 702–707, Online. Association for Computational Linguistics.

Alla Rozovskaya and Dan Roth. 2021. How good (really) are grammatical error correction systems? In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*, pages 2686–2698, Online. Association for Computational Linguistics.

Alex Wang, Yada Pruksachatkun, Nikita Nangia, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel Bowman. 2019. SuperGLUE: a stickier benchmark for general-purpose language understanding systems. In *Advances in Neural Information Processing Systems*, volume 32, pages 3261–3275.

Helen Yannakoudakis, Øistein E Andersen, Ardeshir Geranpayeh, Ted Briscoe, and Diane Nicholls. 2018. Developing an automated writing placement system for ESL learners. *Applied Measurement in Education*, 31(3):251–267.

Helen Yannakoudakis, Ted Briscoe, and Ben Medlock. 2011. A new dataset and method for automatically grading ESOL texts. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, pages 180–189, Portland, Oregon, USA. Association for Computational Linguistics.

Zheng Yuan and Christopher Bryant. 2021. Document-level grammatical error correction. In *Proceedings of the 16th Workshop on Innovative Use of NLP for Building Educational Applications*, pages 75–84, Online. Association for Computational Linguistics.

CXR data annotation and classification with pre-trained language models

Nina Zhou[1], Ai Ti Aw[1], Zhuohan Liu[1],
Cher Heng Tan[2], Yonghan Ting[2], Wenxiang Chen[2], and Sim Zheng Ting[3]
[1]Institute for Infocomm Research(I2R), Singapore,
[2]Tan Toch Seng Hospital (TTSH), Singapore, [3]MOH Holdings (MOHH), Singapore
{zhoun,aaiti,Liu_zhuohan}@.i2r.a-star.edu.sg
{cher_heng_tan,Yonghan_Ting,chen_wen_xiang}@.ttsh.com.sg
Jordan.sim@mohh.com.sg

Abstract

Clinical data annotation has been one of the major obstacles for applying machine learning approaches in clinical NLP. Open-source tools such as NegBio and CheXpert are usually designed on data from specific institutions, which limit their applications to other institutions due to the differences in writing style, structure, language use as well as label definition. In this paper, we propose a new weak supervision annotation framework with two improvements compared to existing annotation frameworks: 1) we propose to select representative samples for efficient manual annotation; 2) we propose to auto-annotate the remaining samples, both leveraging on a self-trained sentence encoder. This framework also provides a function for identifying inconsistent annotation errors. The utility of our proposed weak supervision annotation framework is applicable to any given data annotation task, and it provides an efficient form of sample selection and data auto-annotation with better classification results for real applications.

1 Introduction

Previous work (Wang et al., 2019) has showed clinical text classification can significantly benefit from supervised learning approaches when annotated data is available. However, annotation of clinical data is extremely expensive and time consuming since annotation needs human experts' domain knowledge. To circumvent this, some rule-based methods have been developed using expert knowledge, e.g., NegBio (Peng et al., 2018) and CheXpert (Irvin et al., 2019) using rule labeler to automatically detect the presence of observations in radiology reports. McDermott et al. (2020) mentioned that CheXpert is computationally slow, and its output is non-differentiable, so they proposed to train a BERT based classier (CheXpert++) based on the output of CheXpert. Likewise, Smit et al. (2020) proposed to combine automatic labelers with expert knowledge by first fine-tuning BERT classifier on output of CheXpert and then on a small set of expert annotations augmented with automated backtranslation.

While annotation is data and task specific (Irena et al., 2020), it is further complicated by the differences in writing style, structure, language use such as the vocabularies and phrase variability among different institutes and different countries. For example, phrases "airspace changes" and "infective change(s)" are commonly seen in our local data to describe pneumonia, but rarely seen in MIMIC data (Alistair et al., 2019). In addition, label definitions vary among institutes. For example, CheXpert classifies the sentence "suspicious for pneumonia" as "pneumonia uncertainty" based on their rule definition, while our clinicians/radiologists would consider it as an implication of "pneumonia positive". These differences limit the application of open-source tools on different data, and further limit the applications of methods (McDermott et al., 2020; Smit et al., 2021) which heavily rely on open-source tools.

Some active learning (Chen et al., 2015) and interactive learning methods (Wang et al., 2017) have been commonly used for reducing the experts' annotation burden. Chen et al. (2015)

Proceedings of the 29th International Conference on Computational Linguistics, pages 2801–2811
October 12–17, 2022.

proposed uncertainty-based and diversity-based sampling to annotate clinical NERs. Both sampling approaches adopt random sampling and longest sentence sampling, for comparison, to build the initial set for manual annotation. For subsequent annotation and updating, the uncertainty-based sampling relies on model's predictions while diversity-based selects samples based on pair-wise sentence similarity. Pair-wise sentence similarity is calculated based on individual words, syntax or clinical concepts, with the aim to select samples with lower similarity to annotated samples in the initial set. Wang et al. (2017) proposed an interactive learning method, ReQ-ReC, which is very similar to the uncertainty-based sampling. The method leverages on human experts' domain knowledge to build a list of sense-specific contextual words and use them to search for related sentences to form the initial annotation set. For subsequent sampling, it is based on the model's prediction too. The more ambiguous samples will be selected for annotation.

For the above approaches, it is not efficient in practice since they incur many rounds of model retraining (Chen et al., 2015) and multiple cycles of annotation by experts. Time taken for human annotation is normally affected more by the duration of the annotation cycle than by the sample sizes as experts are not readily available, especially in the clinical domain. Pair-wise sentence similarity is limited by using words, syntax or extracted clinical concepts to represent sentence since they cannot capture the semantic meaning of the whole sentence. In our data, it is very common to have sentences with the same clinical concepts annotated differently due to negation or speculation. In addition, distribution and the number of samples selected for initial annotation affects the performance of the model, which will then affect the prediction quality of the remaining samples.

To effectively select samples for initial annotation and avoid multiple training cycles and annotation by human experts, in this paper, we propose a new weak supervision annotation framework to overcome the retraining and multiple annotation process. Within the proposed framework, we adopt deep neural networks (DNN) for sample selection and text classification, which can fill the gap of using DNNs in active learning for text classification (Schroder et al., 2020). Our work has the following contributions:

1) We propose a generic weak supervision data annotation framework which relies on sentence embedding for sample selection, error checking and auto-annotation.
2) We propose to select representative samples through sentence clustering to kick start the human annotation process, which is a more efficient approach than random selection and longest sentence selection (Chen et al., 2015).
3) We show that our proposed annotation and training approach achieves better performance and requires fewer number of annotated samples.

2 Related Works

Supervised Sentence Encoder. Earlier sentence encoders are trained in supervised way. InferSent (Conneau et al., 2017) is trained on Stanford natural language inference (SNLI) data with three labels. Universal Sentence Encoder (Cer et al., 2018) augments unsupervised learning on labelled SNLI dataset for improved performance. Reimers et al. (2019) proposed SBERT, built by adding a Siamese network on top of BERT model and then fine tuning on NLI data sets. Their experimental results show that SBERT achieves much better results compared to InferSent and Universal Sentence Encoder on STS tasks (Reimers et al., 2019). For other tasks and domains, retraining SBERT on domain sentence pairs with labels are also preferred.

Unsupervised Sentence Encoder. As to unsupervised approaches, with the advent of pretrained language models (PLMs), Devlin et al. (2018) tried to get sentence embeddings from BERT by either averaging the vectors obtained from the last layer or using [CLS] token. Recently, Wang et al. (2021) proposed the transformer-based sequential denoising auto-Encoder (TSDAE) method by exploiting the encoder-decoder structure of transformer. During training, the encoder converts corrupted sentences into fixed-sized vectors and the decoder reconstructs the original sentences from this fixed-sized vectors. To make reconstruction as good as possible, the sentence embedding from the encoder must well represent the semantic

meaning of the sentences. At inference, only the encoder will be used for generating sentence embeddings.

Sentence Textual Similarity (STS) in clinical domain. In clinical domain, there are also related work for evaluating sentence similarity. Mahajan et al. (2020) proposed an iterative intermediate training (IIT) approach for calculating clinical STS by using multi-task learning (MTL). The final system attains promising results for clinical STS tasks by integrating module of Clinical BERT with other language models (BioBERT, MT-DNN, RoBERTa). But the method is not efficient for training sentence encoder as it involves high computation cost for the various pairs of regression tasks due to many possible combinations. Wang et al. (2020b) proposed to take advantage of general domain STS dataset and a small-scale in-domain training data to achieve an impressive result for clinical STS task.

In this paper, we present both supervised and unsupervised sentence encoder training methods in our weak supervised framework for CXR data annotation (See Section 3.2). We evaluate both sentence encoders using a small set of CXR data with pathology labels for a multi-label classification task. We show that our proposed weak supervision framework is effective for semi-supervised data annotation.

3 Methodology

Diseases / Sentences	CXR diseases annotation			
	CAT 1	CAT 2	CAT 3	CAT 4
atelectasis is seen at the right lower zone with ***vague air-space changes.***	u	×	×	+
the heart size cannot be accurately assessed in this projection but ***appears to be enlarged. no obvious consolidation is seen.***	-	×	+	×

Table 1. Annotation examples of our CXR data. '×' indicates the pathology is not mentioned

Our annotation task is to label each sentence in the CXR report into four pathologies, mainly pneumonia (CAT1), pneumothorax (CAT2), cardiomegaly (CAT3) and other diseases (CAT4) as illustrated in Table 1. For each pathology, we further label it as being 'positive'(+), 'negative'(-), or 'uncertain'(u). One sentence may have more than one pathology with one pathology as positive, while describing another pathology as negative or uncertain. If there are no pathologies described, we label it as 'no findings'.

3.1 The Proposed Framework

Our proposed framework is depicted in Figure 1. This proposed new weak supervision annotation framework is applicable to any data annotation task. It exploits an efficient sentence encoder to get high quality sentence embeddings. Using these embeddings, 1) we perform unsupervised clustering to obtain the natural grouping of data based on its distribution. 2) We then select representative sentences for human annotation from each sentence cluster using sematic similarity score. 3) Auto error checking is then performed on the human annotations to reduce bias and inconsistences caused by human errors. 4) We then perform automatic annotation on the remaining data in each cluster by measuring their semantic similarity with the annotated samples. Using this approach, we are able to obtain a set of high-quality data for our classification task.

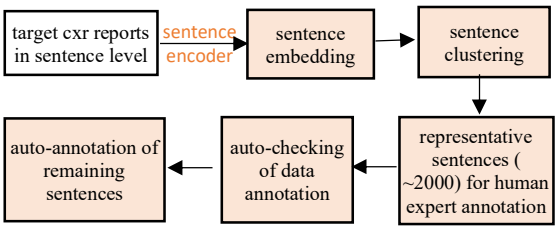

Figure 1. Proposed weak supervision annotation framework. Sentence Encoder is on our CXR data. Auto-checking checks the inconsistence and bias in the human annotation.

3.2 Sample Selection Strategy

We used HDBScan as our clustering method and divide the clusters into noisy clusters and clean clusters. We define the first cluster as *Noisy Cluster* as HDBScan always put all sentences it cannot group into this cluster. This noisy cluster has many variations on sentence length and keywords (mentions). For example, in Cluster 1 of Table 2, sentence 1 mentions both *cardiomegaly* (enlarged heart) and *pleural effusion* (water in the lung), which are keywords used in CAT3 and CAT4 categories. Sentence 2 is much shorter with only the keyword "*cardiac*"

without further information. As the ambiguity is high in this pool, it would be a good source of sentences for human annotation. In our experiment, we selected all samples from this noisy cluster for expert annotation.

We treat all other clusters as clean clusters and further divided them into *Clean-Relevant* and *Clean-Irrelevant*. *Clean-Irrelevant* (see Table 2) *refers* to clusters where most sentences do not mention keywords in any pathologies, and therefore are easily annotated, so we just choose 1 or 2 samples for expert annotation. *Clean-Relevant* clusters normally group sentences with the same pathology, but some sentences may describe more than one or different pathology with similar writing pattern. We selected more samples from them for expert annotation to have a more effective training dataset.

Cluster 1: (*Noisy Custer*) 1. comparing with the previous x-ray dated 0/0/0 findings, cardiac size cannot be completely assessed on the given projection there is increasing homogeneous opacification of the right hemithorax, likely in keeping with underlying pleural effusion. 2. suggest correlation cardiac size cannot be assessed on this suboptimal study 3. nipple markers are noted. … …
Cluster 2: (*Clean-Relevant*) 1. cardiac size is enlarged with perihilar vascular prominence. 2. cardiac silhouette appears enlarged with prominent hilar vessels and upper lobe diversion 3. cardiac size is enlarged with mild perihilar vascular congestion and bilateral perihilar air space shadows. ……
Cluster 3: (*Clean-Irrelevant*) 1. previous image done on 16 July 2016 is reviewed. 2. findings were noted at time of reporting. 3. comparison made with previous study dated 17 Apr. 2013. ……

Table 2. Examples of the *Noisy, Clean-Relevant, Clean-Irrelevant* clusters

Our data selection strategy can be illustrated in *Equation (1)*. Assuming there are N clean clusters and C_i indicates the i-th cluster ($i = 1,2 \dots N$), n_i indicates the number of samples to be selected from C_i. We first rank sentences based on their length for each cluster. Then we select l_i samples comprising the longest one ($n_{longest}$), the shortest one ($n_{shortest}$) and the one with medium length (n_{medium}). For each sentence in $l_i$samples, we use our trained sentence encoder to compute the cosine similarity between it with the rest of the sentences and select the least similar sentences to obtain n_i samples for human annotation.

$$\begin{aligned} &If\ C_i\ is\ Clean-Irrelvant: \quad n_i = 1\ or\ 2; \\ &If\ C_i\ is\ Clean-Relevant: \quad n_i = 2 * l_i; \\ &l_i = n_{longest} + n_{shortest} + n_{medium} \end{aligned} \tag{1}$$

here $n_{longest}$, $n_{shortest}$ and n_{medium} are used to control the length distribution in our training data.

In our experiments, we tried $n_{\{longest,shortest,medium\}} = 1,2,3$, and find out n=2 works best by obtaining enough representative samples selected for efficient expert annotation. There are samples showing that within a cluster, the longer the sentence, the more pathologies it tends to describe. For example, the third sample in cluster 2 describe 'pneumonia' (keyword: '*air space shadows*') and 'other diseases' (keyword: '*vascular congestion'*) besides 'pneumothorax' (keyword: '*cardiac size'* or '*cardiac silhouette*') as the first and second samples do. This is consistent with the assumption observed by Chen et al. (2015) in their clinical NER task too.

3.3 Inconsistent Error Checking and Auto-annotation

We also use the self-trained supervised sentence encoder to assist us in checking the inconsistencies among the human annotations via pairwise sentence comparison among the annotated sentences. If two sentences have high semantic similarity but different labels, we will flag out to the radiologists for label confirmation.

With this high-quality set of human annotated samples, we can auto-annotate the remaining sentences using the same sentence encoder. This is done by assigning the sentence with the same label of the human annotated sentence with highest similarity score. During this automatic labelling process, we set a threshold to reject auto labelling. The rejected sample will be sent for human annotation. A label is assigned only if the cosine similarity value between two sentences is

more than 0.9. This threshold has been proved to be efficient in our experiment.

3.4 CXR Sentence Encoder

A good sentence encoder is important for our annotation framework on sample selection, error checking, and auto-annotation. We use supervised SBERT (Reimers et al., 2019) and unsupervised TSDAE (Wang et al., 2021) for sentence encoder training in our experiments (Figure 2).

To obtain our supervised sentence encoder, we utilize sentence transformer[1] for sentence encoder training and testing. We first do a domain adapted fine-tuning on a pretrained language model using our in-house CXR and MIMIC data (Alistair et al., 2019) based on Transformer [2] with default parameters. We use Roberta-large as the pretrained model as it performs significantly better than other pretrained models based on our experiments. We further train this fine-tuned model on large Semantic Textual Similarity (STS) tasks using SBERT architecture and apply a pooling operation to our fine-tuned language model to get a fixed size sentence embedding output. Different from Wang *et al* (2020b), we include both general STS and clinical STS in the training and validation. Besides STS tasks, we also study the effect of fine-tuning on Natural Language Inference (NLI) for our data annotation task.

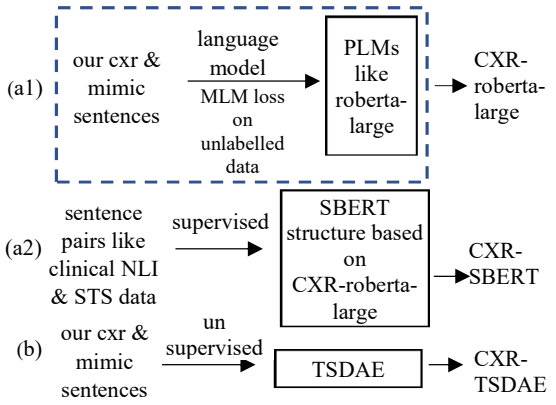

Figure 2. *Supervised Sentence Encoder Training*: (a1) to get domain adapted model (CXR-Roberta-large) through behavior fine tuning on pretrained model; (a2) further trains the adapted model (CXR-Roberta-large) to obtain a supervised sentence encoder (CXR-SBERT) on labelled clinical NLI or STS data. *Unsupervised Sentence Encoder Training*: (b) train unsupervised sentence encoder on unannotated CXR sentences and Roberta-large.

The unsupervised TSDAE training is shown in Figure 2 (b). The data used for domain adaptation in (a1) is also used to train the TSDAE sentence encoder. We use Roberta-large as the base model and train 10 epochs with the batch size of 16 in our experiment. As to other parameters, we use the default values as Wang et al. (2021) used.

3.5 CXR Report Classification

We frame our CXR disease labelling task as a multi-label classification task. We adopt one-hot encoding label strategy. For each sentence, we use one-hot vector with dimension 13 for the labelling. The first 12 dimensions refer to the labeling of the four diseases, each of which has three dimensions namely '1,0,0' '0,1,0'", '0,0,1' to indicate 'disease positive', 'disease negative', and 'disease uncertain' respectively. The last dimension refers to the labelling of 'no_findings', we use '1' to indicate no disease mentioned and '0' for one or more diseases mentioned. For example, if the sentence has pneumonia positive and other disease uncertainty, its one-hot vector label will be [1,0,0,0,0,0,0,0,0,0,0,1,0]. This gives us a maximum of $(3^4 + 1)$ labels, with label values within one disease mutually exclusive to each other and label values among different diseases independent to each other. Let $X = \{X_1, X_2, \dots, X_m\}$ denotes the input space, $Y = \{Y_1, Y_2, \dots, Y_c\}$ denotes the finite set of labels, in which c is the number of labels, $Y_i = [L_1, L_2 \dots, L_{13}]$ and $L_i \in \{0,1\}$. Our classification task is to build a multi-label classifier H that maps an instance x to its associated set of labels: $H(x) = P(y|x)$, in which $x \in X$, $y \in Y$.

In this multi-label classification, the prediction may be partially correct, which we do not consider it as a correct prediction. We use precision, recall and F-score to measure the performance for each disease, and use accuracy to measure the performance on sentence or report level.

4 Experiments

4.1 Data Processing

We use 8 years of CXR reports extracted from our local hospitals. All data have been anonymized

[1] https://github.com/UKPLab/sentence-transformers

[2] https://github.com/huggingface/transformers

and annotation is performed by professional radiologists with more than 13 years of experience who are native English speakers.

We segment all reports into sentences by using NLTK sentence tokenization tool (Bird *et al.* 2009). The length of the sentences varies from 1 to 52 words with an average of 10 words per sentence. We filter out all single word sentences as they do not have much context to infer pathology. We also filter out sentences describing other body parts (e.g., Abdomen, Supine, Neck, Back) as they are not related to our pathology. We obtain 12,350 sentences at the end after removing duplicates.

The data used for unsupervised TSDAE training and supervised SBERT training are summarized in Table 3. MIMIC CXR (Alistair et al., 2019) and in-house CXR are used to fine-tune Roberta-Large and TSDAE on the clinical domain. The data used to fine-tune the SBERT sentence encoders include (i) Combined NLI comprising MEDNLI (Romanov et al., 2018), general domain NLI (StanfordNLI (Bowman et al., 2015) and multi-genre NLI (Nangia *et al*., 2017); (ii) STS-B (STS Benchmark, Cer et al., 2017) consists of a mixture of news, captions, and forums; (iii) SemEval STS 2012-2016 (Agirre et al., 2012-2016) for semantic textual similarity tasks; (iv) Clinical STS 2018 (v) Clinical STS 2019 (Wang et al., 2018 & 2020c) are the only available STS data in clinical domain. We use all the 5 datasets for SBERT training.

data	train/dev/test	average length
data for domain adaptation and TSDAE		
in-house CXR	12350	10
MIMIC CXR	248k	14
data for SBERT sentence encoder		
combined NLI	956k	11.4
STS-B	5749/1500/1379	10.2
STS 2012-2016	23,778	12
clinicalSTS 2018	749/318	25.4
clinicalSTS 2019	1641/412	19.3

Table 3. Data used for sentence encoder training

4.2 Experiments & Results

4.2.1 Sentence Encoder

We studied our proposed framework using three supervised sentence encoders and one unsupervised sentence encoder. All three supervised sentence encoders are based on CXR-SBERT but trained on different datasets. The open-source sentence encoder "sts-robert-large" from sentence-transforms[1] is used as the baseline in our experiment. The settings for the supervised sentence encoder are descried below.

1) CXR-SBERT-nli-stsb: train CXR-SBERT on both NLI and STS-B, followed by continuous training on STS 2012-2016.
2) CXR-SBERT-nli-sts: train CXR-SBERT on both NLI and STS-B, followed by continuous training on all data from STS 2012-2016, clinicalSTS 2018 and Clinical STS2019.
3) CXR_SBERT-sts: train CXR-SBERT on STS-B and STS 2012-2016, followed by continuous training on Clinical STS 2018 and clinical STS 2019.

For training on NLI data set, we use classification objective function with the mean pooling strategy and the mean squared error loss. We set num_epoch as 4 and batch_size as 16. We use Adam optimizer with learning rate 2e−5, and a linear learning rate warm-up over 10% of the training data. For training on STS data, we use regression objective function with mean pooling strategy and cosine similarity loss (Reimbers *et al* 2019). We set num_epoch as 5 and batch_size as 16. Other parameters are the same as Reimers *et al* (2019).

Sentence encoder	Accuracy
sts-roberta-large	93.33
CXR-SBERT-nli-stsb	94.33
CXR-SBERT-nli-sts	**98.33**
CXR-SBERT-sts	**98.33**
CXR-TSDAE	96.67

Table 4. Sentence encoders' comparison on auto-annotation performance on a small CXR data set.

To select a suitable sentence encoder for our task, we use 360 CXR sentences from clean clusters generated by each sentence encoder for experts' annotation. We use 60 as training data and 300 as test data. The 60 sentences are distributed across clusters and are used for auto-annotating the 300 sentences.

The results are shown in Table 4. From the result, we can see that sentence encoder CXR-SBERT-sts and CXR-SBERT-nli-sts produce better results than other encoders, which means the encoders trained on large STS data generate better sentence embeddings on our data. The

model CXR-SBERT-nli-stsb has better performance than the out-of-the-box model (sts-roberta-large), which means domain adapted fine tuning is helpful. The unsupervised sentence encoder CXR-TSDAE performs well but is not so good as SBERT encoder CXR-SBERT-sts.

4.2.2 Semi Auto-annotation Strategy

With good quality sentence embeddings obtained, we leverage HDBscan with Umap[3] for clustering. We set parameters n_neighbors as 200 and n_components as 500 (original data dimension is 1024) for Umap. We set HDBscan parameters min_samples as 10, min_cluster_size as 30. Using the above setting, we generate 94 clusters for 12 thousand sentences (See Appendix A.1 for parameters setting).

We follow the selection criterion in Section 3.2 to select different number of samples for annotation (see Table 6). The selected samples are first annotated by two professional radiologists and checked by a third annotator. After human annotation, we conduct label auto-checking through similarity values. Some inconsistent annotations could be found during auto-checking due to human bias or mistake. These inconsistencies are verified by human annotators again and the verified annotated data is added to the reference for automatic annotation of the remaining sentences by measuring the similarity scores of the remaining sentences with these references.

Categories	(+)	(-)	(u)	total
pneumonia	949	489	79	1517
penumothorax	900	294	76	1210
cardiomeglay	1511	520	775	2806
other diseases	3850	974	132	4956
no_finiding	3436			

Table 5. The in-house data statistics, in which "(+) / (-) / (u)" indicates positive/negative/uncertainty of each pathology.

A total of 11,114 sentence samples are annotated using our proposed framework, during which 150 confusing samples were sent for further verification by our annotators. The final data statistics is shown in Table 5. Most of the data (88%) include only one pathology, while 12% of data include multiple pathologies.

We perform analysis on the size of human annotation samples with respect to the accuracy of auto-annotation of the remaining data. From the results shown in Table 6, we can see the annotation accuracy increased from 69.98% to 90.05% by annotating about 19.8% of data, from a test set of 1,236 reports.

#sents selected (% data)	Auto annotation Accuracy
876 (7.8%)	69.98%
1236 (11.12%)	81.07%
1656 (14.9%)	85.60%
2200 (19.80%)	90.05%

Table 6. Performance of auto-annotation performance on different number of samples selected from further annotation. '%data' indicates the percentage of data selected from 11,114 sentence samples.

4.2.3 Pathology classification

In the experiment, we used SimpleTransformer[4] library for the multi-label classifier training and testing. We used train and evaluation batch of 8, epoch of 3, learning rate of 4e-5 and threshold of 0.5. For other parameters, we use the default values. We split the annotated sentence samples into train data 9879, dev (1235) and test data (1236). The experiment results are shown in Table 7. We can see that most of the categories have f-score at around or more than 95% except two uncertainty cases (pneumonia and others). This model also has been recently tested on 988 reports from local hospital with an accuracy of 98.1% on report level. This successful application of this framework on CXR data boosts our confidence on its application on other annotation tasks (See Appendix A.2).

5 Discussion

5.1 The Robustness of the classifier

We observe that although the SBERT sentence encoder was trained on out-domain labelled STS data, it performs better than the unsupervised TSDAE sentence encoder. The result is consistent with the observation in Schick et al. (2021), who mentioned and demonstrated supervised sentence encoders perform better than unsupervised

[3] https://umap-learn.readthedocs.io/en/latest/clustering.html#umap-enhanced-clustering

[4] https://github.com/ThilinaRajapakse/simpletransformers

sentence encoders. Our auto-annotation experiments demonstrate very good performance on our CXR data. One possible reason can be due to the characteristics of our data. Our data is in sentence level and some sentences are very similar (not much variation), so clustering can generate some very clean clusters which make the annotation inside those clusters very efficient; Besides, the average length of our data is about 10 words, which possibly captures a good embedding for the sentence representation.

Category	precision	recall	f-score
pneumonia (+)	93.30	98	95.59
pneumonia (-)	94.70	94.74	94.74
pneumonia (u)	85.71	75.00	78.00
pneumothorax(+)	97.92	96.91	97.41
pneumothorax(-)	86.96	95.24	94.05
pneumothorax(u)	100	100	100
cardiomegaly (+)	96.27	99.36	97.79
cardiomegaly(-)	100	98.15	99.07
cardiomegaly (u)	98.55	94.44	96.45
others (+)	90.41	94.83	92.57
others (-)	90.70	92.86	91.77
others(u)	77.78	50	60.87
no_finding	97.80	93.93	95.36

Table 7. The classification comparison on each pathology, in which "(+) / (-) / (u)" indicates positive/negative/uncertainty of each pathology.

Though sentence encoder could potentially be used as an annotation tool, it has limitations. It cannot differentiate multi-labels because it treats multi-label as one power set of labels. A classifier trained on annotated data is much more robust than a sentence encoder for label assignments because during classifier training, the weights of the classifier in multiple layers are iteratively updated to represent a sentence in a more accurate way. On the other hand, a sentence encoder converts each sentence to a vector in a fixed way and treats each dimension of sentence embeddings equally (Reimers et al., 2019). Therefore, we fine-tuned cxr-roberta-large on the annotated data to obtain a more robust classifier (see Section 4.2.3).

5.2. The Effectiveness of our method

To demonstrate the effectiveness of our proposed annotation method, we compare our method with the uncertainty-based sampling method with the initial set selected randomly and based on sentence length (Chen et al., 2015). The reasons of using the two baselines of random sampling and sentence length-based sampling are because 1) although random sampling is simple and straightforward, it performs competitive to most sophisticated strategies (Schroder et al., 2020); 2)The length strategy is a data driven strategy, which is simple and has been tested to be effective for medical data (Chen et al., 2015), and it is slightly better than random sampling for medical data.

We use the total number of samples selected for manual annotation as an evaluation measure for the annotation efforts among different methods. Note that other methods will incur additional time for updating model and waiting time of more cycles of experts' annotation. In this experiment, we used the same library and setting as Section 4.2.3. For data setup, we have 10k+ data for annotation experiments and 2000 data used for model evaluation. For the batch sizes used in active learning, we use batch size of N= [100, 300, 600, 1000, 1500, 2200, 2800, 3600, 4500, 5500, 6700, 10350] for 12 iterations in our experiment, for the comparison among different annotation methods (see Figure 3).

From Figure 3, we can see that with more data for training, the performances of all models increase. The performance increases faster when the number of samples is less than 2000. After 2000 samples, the increasing trend slows down. Our proposed method demonstrates better and faster performance improvement than other two methods as our method selects more representative sentences for human annotation and model training. The auto-checking further assists us to check for errors and control our data quality. The sampling strategy based on longest sentences are better than the random selection, but when more and more samples are selected for training, the gap becomes smaller. Our experiment shows our proposed approach requires only 2200 manually annotated samples to perform auto-annotation of the remaining samples to reach the best performance where other methods need to manually annotate almost all the sentences to reach around 95.06% accuracy. This can contribute to our auto-annotation strategy with auto-checking process which reduces human annotation bias and errors and has a positive impact on the quality of the annotation data.

6 Conclusion

We propose a semi-supervised annotation scheme which avoids multiple model re-training and expert annotation which is applicable to CXR text data and other domain data annotation (Appendix A.2). Within the framework, we investigate a gap mentioned by Schroder et al. (2020) by using fine-tuning-based models in active learning for text classification. We utilize a self-trained sentence encoder for effective sample selection through clustering, error auto-detection and sample auto-annotation. Based on the annotated data, we further fine-tune a pre-trained language model to obtain a robust classifier which demonstrates high performance on CXR data disease detection. This method greatly improves data annotation efficiency and relieves human annotation burden.

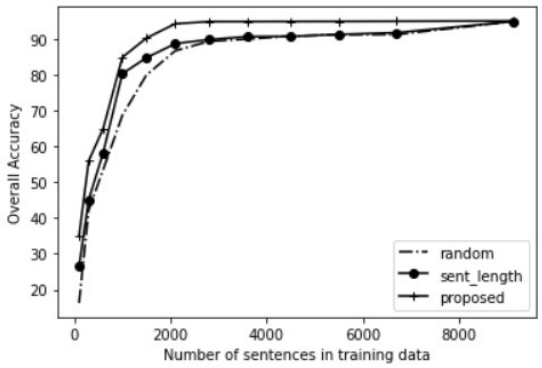

Figure 3. Comparison of different annotation methods

References

Emily Alsentzer, John Murphy, William Boag, Wei-Hung Weng, Di Jindi, Tristan Naumann, Matthew McDermott, *Publicly Available Clinical BERT Embeddings,* Proceedings of the 2nd Clinical Natural Language Processing Workshop, pp.72-78, 2019

Eneko Agirre, Carmen Banea, Daniel Cer, Mona Diab, Aitor Gonzalez-Agirre, Rada Mihalcea, German Rigau, and Janyce Wiebe. 2016. *Semeval-2016 task 1: Semantic textual similarity, monolingual and cross-lingual evaluation*. In Proceedings of the 10th International Workshop on Semantic Evaluation (SemEval-2016), pages 497–511.

Eneko Agirre, Carmen Banea, Claire Cardie, Daniel Cer, Mona Diab, Aitor Gonzalez-Agirre, Weiwei Guo, Inigo Lopez-Gazpio, Montse Maritxalar, Rada Mihalcea, et al. 2015. *Semeval-2015 task 2: Semantic textual similarity*, *english, spanish and pilot on interpretability*. In Proceedings of the 9th international workshop on semantic evaluation (SemEval 2015), pages 252–263.

Eneko Agirre, Carmen Banea, Claire Cardie, Daniel Cer, Mona Diab, Aitor Gonzalez-Agirre, Weiwei Guo, Rada Mihalcea, German Rigau, and Janyce Wiebe. 2014. *Semeval-2014 task 10: Multilingual semantic textual similarity*. In Proceedings of the 8th international workshop on semantic evaluation (SemEval 2014), pages 81–91.

Eneko Agirre, Daniel Cer, Mona Diab, Aitor GonzalezAgirre, and Weiwei Guo. 2013. *Semeval-2013 shared task: Semantic textual similarity*. In Second Joint Conference on Lexical and Computational Semantics (* SEM), Volume 1: Proceedings of the Main Conference and the Shared Task: Semantic Textual Similarity, pages 32–43.

Eneko Agirre, Mona Diab, Daniel Cer, and Aitor Gonzalez-Agirre. 2012. *Semeval-2012 task 6: A pilot on semantic textual similarity*. In Proceedings of the First Joint Conference on Lexical and Computational Semantics-Volume 1: Proceedings of the main conference and the shared task, and Volume 2: Proceedings of the Sixth International Workshop on Semantic Evaluation, pages 385–393. Association for Computational Linguistics.

Steven Bird, Edward Loper and Ewan Klein 2009, *Natural Language Processing with Python*. O'Reilly Media Inc.

Zalan Bodo, Zsolt Minier, Lehel Csató, Active Learning with Clustering, JMLR: Workshop and Conference Proceedings, 2011.

Samuel R. Bowman, Gabor Angeli, Christopher Potts, and Christopher D. Manning. 2015. *A large annotated corpus for learning natural language inference*. In Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing (EMNLP).

Daniel Cer, Mona Diab, Eneko Agirre, Iigo LopezGazpio, and Lucia Specia. 2017. *SemEval-2017 Task 1: Semantic Textual Similarity Multilingual and Crosslingual Focused Evaluation*. SemEval-2017, pages 1–14, Vancouver, Canada

Daniel Cer, et al., 2018, *Universal Sentence Encoder*, https://arxiv.org/pdf/1803.11175.pdf

Yukun Chen,Thomas A. Lasko, Qiaozhu Mei, Joshua C. Denny, and Hua Xu, *A Study of Active Learning Methods for Named Entity Recognition in Clinical Text*, J Biomed Inform. 2015 Dec; 58: 11–18.

Alexis Conneau, Douwe Kiela, Holger Schwenk, Lo¨ıcBarrault, and Antoine Bordes. 2017. *Supervised Learning of Universal Sentence Representations from Natural Language Inference Data*. In Proceedings of the 2017 Conference on

Empirical Methods in Natural Language Processing, pages 670–680,

Alexis Conneau and Douwe Kiela. 2018. *SentEval: An Evaluation Toolkit for Universal Sentence Representation*s. arXiv preprint arXiv:1803.05449.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. *BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding*. arXiv preprint arXiv:1810.04805.

Suchin Gururangan et al., 2020. *Don't stop pretraining: Adapt language models to domains and tasks*. ACL 2020, pages 8342–8360.

Jeremy Irvin et al., *CheXpert: A Large Chest Radiograph Dataset with Uncertainty Labels and Expert Comparison*. AAAI 2019

Alistair E. W. Johnson et al., 2019. *MIMIC-CXR-JPG, A large publicly available database of labeled chest radiographs*, https://arxiv.org

Bohan Li, Hao Zhou, Junxian He, Mingxuan Wang, Yiming Yang, and Lei Li. 2020. *On the sentence embeddings from pre-trained language models*. EMNLP 2020, pages 9119–9130.

Diwakar Mahajan, et al., *Identification of Semantically Similar Sentences in Clinical Notes: Iterative Intermediate Training Using Multi-Task Learning,* JMIR Med Inform. 2020 Nov; 8(11): e22508

L. McInnes, J. Healy, S. Astels, *HDBSCAN: Hierarchical density based clustering* In: Journal of Open Source Software, The Open Journal, volume 2, number 11. 2017

Matthew B. A., McDermott, et al., *Chexpert++: Approximating the chexpert labeler for speed, differentiability, and probabilistic output*. MLHC 2020.

Nikita Nangia, Adina Williams, Angeliki Lazaridou, Samuel R. Bowman, *The RepEval 2017 Shared Task: Multi-Genre Natural Language Inference with Sentence Representations.* Proceedings of the 2nd Workshop on Evaluating Vector-Space Representations for NLP, pages 1–10. 2017

Hieu T. Nguyen and Arnold Smeulders, Active Learning Using Pre-clustering, ICML 2004.

Peng Y, Wang X, Lu L, Bagheri M, Summers R, Lu Z. *NegBio: a high-performance tool for negation and uncertainty detection in radiology reports*. AMIA Summits on Translational Science Proceedings. 2018;2017:188.

Pedregosa *et al.*, *Scikit-learn: Machine Learning in Python*, JMLR 12, pp. 2825-2830, 2011.

Jason Phang, Thibault Fevry, Samuel R. Bowman, *Sentence Encoders on STILTs: Supplementary Training on Intermediate Labeled-data Tasks*, https://arxiv.org/pdf/1811.01088.pdf

Nils Reimers and Iryna Gurevych, *Sentence-BERT: Sentence Embeddings using Siamese BERT-Networks*, EMNLP 2019

Alexey Romanov and Chaitanya Shivade. Lessons from natural language inference in the clinical domain. arXiv preprint arXiv:1808.06752, 2018.

Timo Schick and Hinrich Schütze, Generating Datasets with Pretrained Language Models, https://arxiv.org/pdf/2104.07540.pdf, EMNLP 2021

Christopher Schröder, Andreas Niekler, A Survey of Active Learning for Text Classification using Deep Neural Networks, 2020. https://arxiv.org/pdf/2008.07267.pdf

Akshay Smit et al., *CheXbert: Combining Automatic Labelers and Expert Annotations for Accurate Radiology Report Labeling Using BERT*, EMNLP2020, pp 1500-1519

Irena Spasic and Goran Nenadic, *Clinical text data in machine learning: systematic review*, JMIR Med Inform. 2020 Mar; 8(3): e17984

C. Shivade, *MedNLI - A Natural Language Inference Dataset For The Clinical Domain* (version 1.0.0). PhysioNet. https://doi.org/10.13026/C2RS98.2019

Bin Wang and C-C Jay Kuo, (2020a). *A Sentence Embedding Method By Dissecting BERT-based Word Models*, IEEE/ACM Transactions on Audio, Speech, and Language Processing,vol 28, 2146-2157.

Kexin Wang, Nils Reimers, Iryna Gurevych, TSDAE: Using Transformer-based Sequential Denoising Auto-Encoder for Unsupervised Sentence Embedding Learning, EMNLP2021.

Yuxia Wang, Karin Verspoor, Timothy Baldwin, (2020b) *Learning from Unlabelled Data for Clinical Semantic Textual Similarity*, Proceedings of the 3rd Clinical Natural Language Processing Workshop ACL 2020, pages 227–233.

Yue Wang, Kai Zheng, Hua Xu, and Qiaozhu Mei, *Clinical Word Sense Disambiguation with Interactive Search and Classification*, AMIA Annu Symp Proc. 2017; 2017: 2062–2071.

Wang Y, et al., *A clinical text classification paradigm using weak supervision and deep representation*, BMC Medical Informatics and Decision Making 2019.

Y. Wang, Afzal N, Liu S, Rastegar-Mojarad M, Wang L, Shen F, Fu S, Liu H. *Overview of the BioCreative/OHNLP Challenge 2018 Task 2: Clinical Semantic Textual Similarity*. Proceedings of the BioCreative/OHNLP Challenge. 2018.

Y. Wang et al., (2020c) *The 2019 n2c2/OHNLP Track on Clinical Semantic Textual Similarity: Overview*. JMIR Medical Informatics 2020, 8(11), p.e23375.

Yan Zhang et al., *An Unsupervised Sentence Embedding Method by Mutual Information Maximization*, https://arxiv.org/pdf/2009.12061.pdf, MNLP2020

A Appendix

A.1 HDBSCAN & UMAP parameters setting

We had performed extensive clustering experiments on this CXR data. The UMAP is for dimension reduction and n_neighbors and n_components are two of the most important parameters. The bigger n_neighbors, it will look at more global manifold structure. If there are no ground truth labels, it is hard to know which values are best for those two parameters. And those values may change for different data.

HDBSCAN has two important parameters min_cluster_size and min_samples, which can cause quite different clustering if we change them in a wide range. The large min_cluster_size, more data points will be rejected. Our strategy is to obtain the initial results using the default parameters and then adjust the values within a range to get good clustering. This clustering has been tried on another set of finance data and compared with K-means which needs to have the number of clusters specified first. The clusters generated by HDBSCAN with UMAP is much more sensible and preferred by clients.

A.2 Application of the proposed method on anther data

We have used the method on another financial in-house data set for topic classification. The number of sentences used for training is 5044, and the average length of sentences is around 11, with maximum length 48 and minimum length 1. The number of clusters generated is 24. Through the auto error checking and auto-annotation, we achieved satisfactory classification result.

uChecker: Masked Pretrained Language Models as Unsupervised Chinese Spelling Checkers

Piji Li
College of Computer Science and Technology,
Nanjing University of Aeronautics and Astronautics
MIIT Key Laboratory of Pattern Analysis and Machine Intelligence
Nanjing, Jiangsu, China
pjli@nuaa.edu.cn

Abstract

The task of Chinese Spelling Check (CSC) is aiming to detect and correct spelling errors that can be found in the text. While manually annotating a high-quality dataset is expensive and time-consuming, thus the scale of the training dataset is usually very small (e.g., SIGHAN15[1] only contains 2339 samples for training), therefore supervised-learning based models usually suffer the data sparsity limitation and over-fitting issue, especially in the era of big language models. In this paper, we are dedicated to investigating the **unsupervised** paradigm to address the CSC problem and we propose a framework named **uChecker** to conduct unsupervised spelling error detection and correction. Masked pretrained language models such as BERT are introduced as the backbone model considering their powerful language diagnosis capability. Benefiting from the various and flexible MASKing operations, we propose a Confusionset-guided masking strategy to fine-train the masked language model to further improve the performance of unsupervised detection and correction. Experimental results on standard datasets demonstrate the effectiveness of our proposed model uChecker in terms of character-level and sentence-level Accuracy, Precision, Recall, and F1-Measure on tasks of spelling error detection and correction respectively.

1 Introduction

Chinese Spelling Check (CSC) is a crucial and essential task in the area of natural language processing. It aims to detect and correct spelling errors in the Chinese text (Chang, 1995; Wang et al., 2020b). Generally, sequence translation (Wang et al., 2018; Ge et al., 2018; Wang et al., 2019, 2020a; Kaneko et al., 2020) and sequence tagging (Omelianchuk et al., 2020; Liang et al., 2020; Mallinson et al., 2020; Parnow et al., 2021) are the two most typical technical paradigms to tackle the problem. Benefiting from the development of pretraining techniques, many researchers fine-tune the pretrained language models such as BERT (Devlin et al., 2019) on the task of CSC and obtain encouraging performance (Zhao et al., 2019; Hong et al., 2019; Zhang et al., 2020; Liu et al., 2021; Li et al., 2021; Huang et al., 2021; Guo et al., 2021; Zhang et al., 2021; Li and Shi, 2021; Dai et al., 2022). Meanwhile, it should be emphasized that almost all of the above mentioned models are trained via the **supervised learning** paradigm.

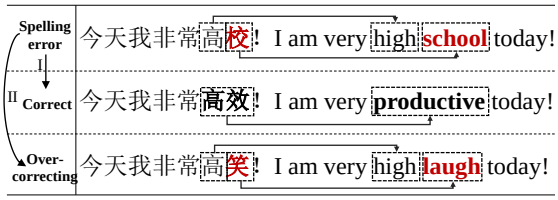

Figure 1: Illustration for the task of Chinese spelling check (operation path I) as well as the over-fitting phenomenon existing in the current supervised learning based models (operation path II).

However, during the investigating stage about those newly typical state-of-the-art models, we observe some spiny and serious phenomenons: (1) Occasionally those models may generate some special **over-correcting** results. As shown in Figure 1, operation path **I** is the regular spelling error detection and correction path, while operation path **II** is also observable in the inference stage where the models can detect the errors correctly but rectify them using some other error tokens in the correction stage. (2) The spelling error detection and correction performance will drop dramatically when those models did not see the spelling error cases in the training dataset or the text are from different genres and domains. This issue tells us that the **generalization capability** of those models are limited and need to be enhanced.

Then what are the causes of these phenomenons?

[1] http://ir.itc.ntnu.edu.tw/lre/sighan8csc.html

Proceedings of the 29th International Conference on Computational Linguistics, pages 2812–2822
October 12–17, 2022.

Corpus	#Train	#ErrTrain	AvgLen	#Test	#ErrTest	AvgLen
SIGHAN13	350	350	49.2	1,000	996	74.1
SIGHAN14	6,526	3,432	49.7	1,062	529	50.1
SIGHAN15	3,174	2,339	30.0	1,100	550	30.5

Table 1: Statistics of the SIGHAN series datasets.

Since some of the models are already strong enough (which are constructed based on big pretrained models), then we shift our eyes to the data perspective. In real practical scenarios, natural human-labeled spelling error corpus are difficult and expensive to obtain. Although some works such as Wang et al. (2018) employ OCR and ASR based techniques to automatically synthetic the paired samples by replacing the correct tokens using visually or phonologically similar characters, obviously, the constructed data is unrealistic and far from the real and objective scenarios. Therefore, actually, the scale of the typical corpus for the task of Chinese spelling check is very small. Considering that almost all the research works have used SIGHAN series datasets (Tseng et al., 2015) to train and evaluate their algorithms, we conduct counting on those three corpora, and the statistics results are shown in Table 1. From the results we can observe that there are only 2k∼3k sentences with spelling errors in the training dataset and really far from the practical requirements.

Thus, sticking to train the supervised learning models based on those scale-limited resources might not be a wise direction. Therefore, in this paper, we are dedicated to exploring **unsupervised** frameworks to conduct Chinese spelling error detection and correction. Fortunately, masked pretrained language models such as BERT (Devlin et al., 2019), RoBERTa (Liu et al., 2019), ELECTRA (Clark et al., 2020), etc. can satisfy the needs of detecting and correcting spelling errors in an unsupervised manner. First, the masked training strategy is naturally a convenient and perfect shortcut for us to conduct token-grained detection and correction. For example, we can mask any token and predict it based on the bi-directional context to see if the current token is appropriate or not. Second, the pretrained language models are usually trained using large-scale corpora, thus the language diagnosis capability is very strong. Intuitively, these models can also guarantee the generalization capability considering the corpora may contain text from a wide range of domains and genres.

Therefore, based on the masked pretrained language models, we propose a framework named **uChecker** to conduct unsupervised spelling error detection and correction respectively. uChecker is a two-stage framework and it will detect the text token-by-token first and then correct the abnormal tokens. Models such as BERT are introduced as the backbone model. Inspired by the previous works (Wang et al., 2019; Liu et al., 2021), benefiting from the various and flexible masking operations, we also introduce a confusionset-guided masking strategy to fine-train the masked language model to further improve the performance of unsupervised detection and correction. Though uChecker is a two-stage framework, we design an elegant method to let the information pass BERT only once to guarantee the time efficiency. Moreover, interestingly, in unsupervised settings, we experimentally find the performance of error detection is crucial to the global and general performance. It means that the correction capability of the pretrained language models are strong enough, then the key-point is how to improve the performance of detection. Therefore, in uChecker, we also design several algorithms to improve the performance of seplling error detection. Yasunaga et al. (2021) employ GPT2-like models to conduct unsupervised English grammatical error correction which also verifies the feasible of our direction.

In summary, our contributions are as follows:

- We propose an unsupervised framework named **uChecker** to conduct Chinese spelling error detection and correction.
- Benefiting from flexible masking operations, we introduce s confusionset-guided masking strategy to fine-train BERT to further improve the performance.
- We experimentally find that error detection is crucial to the global SCS performance. Therefore, we also design some algorithms to improve the capability of error detection.
- Extensive experiments on several benchmark datasets demonstrate the effectiveness of the proposed approach. And the results also show that uChecker can even outperform some strong supervised models.

Rate	Strategy	今天我非常高效！
20%:80%	<MASK>	今天我非常高<MASK>！
20%:10%	Random	今天我非常高语！
20%:10%	Unchange	今天我非常高效！

Figure 2: Illustration of the masking strategies of BERT during the pretraining stage.

2 Background: Masked Language Models

BERT (Devlin et al., 2019) is the most typical masked language model, and it is regarded as the backbone model of our proposed framework uChecker, therefore in this section we introduce the technical details of this model, especially the masking strategies.

BERT is constructed based on the model of Transformer (Vaswani et al., 2017). After preparing the input samples, an embedding layer and a stack of Transformer layers are followed to conduct the bi-directional semantic modeling. Specifically, for the input, we first obtain the representations by summing the word embeddings with the positional embeddings:

$$\mathbf{H}_t^0 = \mathbf{E}_{w_t} + \mathbf{E}_{p_t} \tag{1}$$

where 0 is the layer index and t is the state index. $\mathbf{E}_w$ and $\mathbf{E}_p$ are the embedding vectors for tokens and positions, respectively. Then the obtained embedding vectors $\mathbf{H}^0$ are fed into several Transformer layers. Multi-head self-attention is used to conduct bidirectional representation learning:

$$\begin{aligned}
\mathbf{H}_t^1 &= \textsc{Ln}\left(\textsc{Ffn}(\mathbf{H}_t^1) + \mathbf{H}_t^1\right) \\
\mathbf{H}_t^1 &= \textsc{Ln}\left(\textsc{Slf-Att}(\mathbf{Q}_t^0, \mathbf{K}^0, \mathbf{V}^0) + \mathbf{H}_t^0\right) \\
\mathbf{Q}^0 &= \mathbf{H}^0\mathbf{W}^Q \\
\mathbf{K}^0, \mathbf{V}^0 &= \mathbf{H}^0\mathbf{W}^K, \mathbf{H}^0\mathbf{W}^V
\end{aligned} \tag{2}$$

where $\textsc{Slf-Att}(\cdot)$, $\textsc{Ln}(\cdot)$, and $\textsc{Ffn}(\cdot)$ represent self-attention mechanism, layer normalization, and feed-forward network respectively (Vaswani et al., 2017). After L Transformer layers, we obtain the final output representation vectors $\mathbf{H}^L \in \mathbb{R}^{T\times d}$, where T is the input sequence length and d is the vector dimension.

The masking strategies used in BERT are shown in Figure 2. There are 20% tokens will be masked, and among them there are 80% tokens are replaced with a special symbol such as <MASK>, and 10% are replaced with a random token, and the left 10% keep unchanged. What should be emphasized here is the ***random replacing operation*** which plays an crucial role in the following model designs about the unsupervised detection and correction as well as the confusionset-guided fine-training.

Finally, a linear function g with $\mathrm{softmax}$ activation is used to predict the masked token x_t via:

$$p\left(x_t | x_{\leq t-1}, x_{\geq t+1}\right) = \mathrm{softmax}\left(g\left(\mathbf{h}_t\right)\right) \tag{3}$$

3 The Proposed uChecker Framework

3.1 Overview

Figure 3 depicts the basic components of our proposed framework uChecker. The backbone model is masked language model, say BERT. Note that all the parameters of BERT are frozen. Input is an incorrect sentence $X = (x_1, x_2, \ldots, x_T)$ which contains spelling errors, where x_i denotes each token (Chinese character) in the sentence, and T is the length of X. The objective of the task Chinese spelling check is to detect and correct all errors in X and obtain a new sentence $Y = (y_1, y_2, \ldots, y_{T'})$. Benefiting from the various and flexible masking operations, we introduce the confusionset-guided masking strategy to fine-train BERT to further improve the performance of CSC. We also design several algorithms such as unsupervised detection (UnsupDetection), supervised detection (SupDetection), and Ensemble of UnsupDetection and SupDetection to improve the performance of error detection to further improve the overall performance.

3.2 Unsupervised Spelling Error Detection

Given a pretrained BERT model, for a sentence $X = (x_1, x_2, \ldots, x_T)$ which need to be checked, the preconceived first guess is to mask the tokens one each time from left to right diagonally, as shown in Figure 4, and then input the masked sequence X' into BERT to conduct prediction. Assume token x_t is masked, the predicted distribution at position t is $p(x'_t)$, then we can obtain the probability of the corresponding original input token x_t by:

$$p^u_{x_t} = p_{x'_t = x_t}(x'_t) \tag{4}$$

Intuitively, if x_t is just the correct token, then $p^u_{x_t}$ will be very large (say 0.99). Otherwise, error may hide in this position. Therefore, the simple

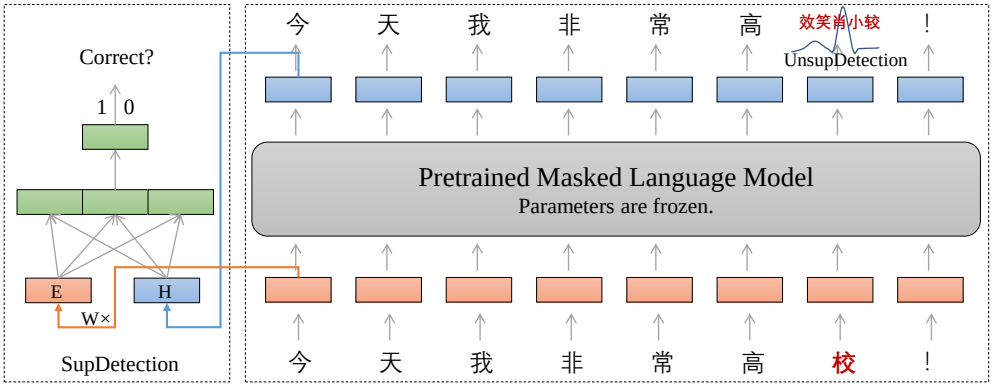

Figure 3: The proposed uChecker framework for unsupervised Chinese spelling error detection and correction.

<CLS><MASK>天我非常高校！ <SEP>
<CLS>今<MASK>我非常高校！ <SEP>
<CLS>今天<MASK>非常高校！ <SEP>
……
<CLS>今天我非常高<MASK>！ <SEP>
<CLS>今天我非常高校<MASK> <SEP>

Figure 4: Illustration of the first guess **diagonal masking strategy** for unsupervised error detection and correction, which is actually **time-consuming and not necessary**.

approach to conduct detection is to find a threshold θ^u and diagnose the results by:

$$error_t = \begin{cases} 1, & \text{if } p^u_{x_t} < \theta^u \\ 0, & otherwise \end{cases} \quad (5)$$

where $error_t = 1$ means that token x_t is not correct in sentence X.

However, although diagonal masking strategy is a natural way to conduct token diagnose, considering that for each sentence with length T, we need prepare T sequences to feed BERT to conduct prediction, which is a time-consuming procedure with low efficiency, and it is also difficult to be deployed and executed concurrently. Recall the masking strategies shown in Figure 2, besides replacing the tokens with <MASK> symbol, *BERT also uses **random tokens** to conduct masking*. Therefore, we do not need to rigidly obey the <MASK> based masking approach. On the contrary, we can briefly regard the potential error tokens as the random masking strategy and just feed the original input sentence X into the BERT model to conduct probability estimation. And this approach can execute with high concurrency because we can process hundreds or even thousands of sentences in a batch based on the parallel computing capability of GPUs. Moreover, since random masking is feasible, then *how about conduct random masking using the corresponding tokens from the **confusionset***? This is the **inspiration** of confusionset-guided fine-training strategy which will be introduced in the following sections.

For each sentence X, the spelling error detection stage will return a list containing the indices of the wrong tokens $\mathcal{I}^e = [2, 5, i, \dots]$, ranked by the predicted corresponding probability in an ascending order. The order indicates that the most worse token will be corrected firstly.

3.3 Unsupervised Spelling Error Correction

Given the detected wrong token indices $\mathcal{I}^e$, the unsupervised error correction component then scans the list and chooses the most appropriate tokens from the probability distribution to conduct correction. Specifically, for any index $i \in \mathcal{I}^e$, the predicted distribution is $p(x'_i)$, then we can straightforwardly select the token with the largest score as the correct result:

$$x_j = \text{argmax}_j \ p_{x'_i = x_j}(x'_i) \quad (6)$$

The operation of unsupervised spelling error correction is simple, therefore the correction performance will completely depend on the capability of the pretrained backbone language models.

Confusionset-guided Token Selection Due to the special input methods such as Pinyin and Wubi, many Chinese characters are similar either in phonology or morphology. There are about 76% of Chinese spelling errors belong to phonological similarity error and 46% belong to visual similarity error (Liu et al., 2011). Intuitively, incorporating the Confusionset with the token selection procedure may improve the performance. Therefore, we

Algorithm 1: Confusionset-guided Token Correction

Data: Wrong token indices $\mathcal{I}^e$; The predicted distributions for all the positions $\mathcal{P}$; The predefined confusionset $\mathcal{C}$ (hashmap<string, list>).
Result: The correct tokens Y for $\mathcal{I}^e$.

```
Y = [] ;
for i ∈ I^e do
    W_i = top_k(P_i);
    for w_i ∈ W_i do
        if w_i ∈ C(x_i) then
            Y.insert(w_i);
            break;
        end
        Y.insert(W_i[0]);
    end
end
```

further build a simple confusionset-guided token selection approach as shown in Algorithm 1.

Specifically, for each detected index i, we fist fetch the top_k tokens according to the distribution $\mathcal{P}_i$). Then if the top_k tokens are also from the corresponding confusionset, then we get the result. Otherwise, we still select the best predicted result.

3.4 Self-Supervised Spelling Error Detection

Surprisingly and interestingly, during the experiments stage, we find that **the performance of error detection plays an crucial role** in affecting the global checking performance. Therefore, improving the capability of error detection can benefit the whole system. But, there is a precondition that we cannot adjust the original backbone BERT parameters because unsupervised error correction is the essential component of our uChecker framework, and we do not want to let the BERT parameters collapse to some special areas or domains. So the BERT parameters need to be frozen when designing the error detection strategies.

As shown in Figure 3, after examining the model carefully, we create a **smart but straightforward self-supervised detection method** to tackle the problem. The basic observation is that, based on the masked language models, the information in the output hidden states $\mathbf{H}$ will be more closer to the true tokens because the model will use them ($\mathbf{H}$) to conduct masked prediction (Eq. 3). Moreover, the information in the embedding layer $\mathbf{E}$ also contains the token information. Then we assume that for any correct token x_i and error token x_j, pair $(\mathbf{e}_i, \mathbf{h}_i)$ for normal token holds a more tight relationship than the pair $(\mathbf{e}_j, \mathbf{h}_j)$ for wrong token, where $\mathbf{e}$ is the learnt token embedding and $\mathbf{h}$ is the output layer of BERT:

$$\mathcal{M}(\mathbf{e}_i, \mathbf{h}_i) > \mathcal{M}(\mathbf{e}_j, \mathbf{h}_j) \tag{7}$$

where $\mathcal{M}$ is metric to represent the interaction relationship, and here we use the following calculations to conduct the interaction modeling:

$$\mathbf{h}_i^s = \mathbf{W}_s(\mathbf{e}_i'; \mathbf{h}_i; \mathbf{e}_i' \odot \mathbf{h}_i; |\mathbf{e}_i' - \mathbf{h}_i|) + \mathbf{b}_s \tag{8}$$

where ; is the concatenation operation and $\mathbf{e}_i'$ is a transformation of $\mathbf{e}_i$ using:

$$\mathbf{e}_i' = \mathbf{W}_e(\mathbf{e}_i) + \mathbf{b}_e \tag{9}$$

This transformation is **essential and cannot be ignored** because that $\mathbf{e}$ and $\mathbf{h}$ are in different vector space. *Otherwise the training will not converge*.

We use cross entropy as the optimization objective:

$$\begin{aligned} \mathbf{y}_i^s &= softmax(\mathbf{h}_i^s) \\ \mathcal{L}^{\mathrm{s}} &= -\sum_{i=0}^{1} \log P^{\mathrm{s}}(\mathbf{y}_i^t | \mathbf{h}_i^s) \end{aligned} \tag{10}$$

For the self-supervised learning, we still employ the masked training strategy to conduct training, where we assign the label for random masking is 1 (position with errors), and 0 for those unchanged positions. Let $p_{x_t}^s = \mathbf{y}_t^s[1]$ be the self-supervised probability of error, then we also set up a threshold θ^s to conduct diagnose as well:

$$error_t = \begin{cases} 1, & \text{if } p_{x_t}^s >= \theta^s \\ 0, & otherwise \end{cases} \tag{11}$$

where $error_t = 1$ means that token x_t is not correct in sentence X.

Note that BERT parameters are **frozen** during the self-supervised learning procedure, therefore we only conduct optimization for a small group of parameters in Eq. 8 and Eq. 9, which is a light-scale training stage.

3.5 Ensemble Detection Methods

Obvious, we can collect all the detected error positions ($\mathcal{I}_u^e$ and $\mathcal{I}_s^e$) by unsupervised and self-supervised detectors respectively, which we name it ensemble detection operation.

TestSet	Model	Detection			Correction		
		PREC.	REC.	F1	PREC.	REC.	F1
SIGHAN13	**Supervised Methods**						
	LMC (Xie et al., 2015)	79.8	50.0	61.5	77.6	22.7	35.1
	Hybird (Wang et al., 2018)	54.0	69.3	60.7	-	-	52.1
	Confusionset (Wang et al., 2019)	66.8	73.1	69.8	71.5	59.5	69.9
	SpellGCN (Cheng et al., 2020)	82.6	88.9	85.7	98.4	88.4	93.1
	Unsupervised Methods						
	uChecker (Sec.3)	81.6	**93.0**	**86.9**	95.8	93.1	94.4
	w/o self-supervised detection	83.3	90.3	86.7	96.6	**93.2**	**96.8**
	w/o confusionset	**84.3**	89.0	86.6	89.8	86.4	88.1
SIGHAN14	**Supervised Methods**						
	LMC (Xie et al., 2015)	56.4	34.8	43.0	71.1	50.2	58.8
	Hybird (Wang et al., 2018)	51.9	66.2	58.2	-	-	56.1
	Confusionset (Wang et al., 2019)	63.2	82.5	71.6	79.3	68.9	73.7
	SpellGCN (Cheng et al., 2020)	83.6	78.6	81.0	97.2	76.4	85.5
	Unsupervised Methods						
	uChecker (Sec.3)	75.9	73.3	74.6	91.7	**84.9**	85.0
	w/o self-supervised detection	72.4	66.1	69.2	92.9	81.4	**86.8**
	w/o confusionset	78.0	68.5	72.9	84.3	78.2	78.3
SIGHAN15	**Supervised Methods**						
	LMC (Xie et al., 2015)	56.4	34.8	43.0	71.1	50.2	58.8
	Hybird (Wang et al., 2018)	56.6	69.4	62.3	-	-	57.1
	Confusionset (Wang et al., 2019)	66.8	73.1	69.8	71.5	59.5	69.9
	SpellGCN (Cheng et al., 2020)	88.9	87.7	88.3	95.7	83.9	89.4
	PLOME (Liu et al., 2021)	94.5	87.4	90.8	97.2	84.3	90.3
	Unsupervised Methods						
	uChecker (Sec.3)	85.6	79.7	82.6	91.6	**84.8**	88.1
	w/o self-supervised detection	75.8	71.3	73.5	92.6	84.5	88.4
	w/o confusionset	87.4	75.9	81.2	84.6	77.7	81.0

Table 2: The character-level performance on both detection and correction level. *We notice that character-level detection performance of scrips from Hong et al. (2019) and Wang et al. (2019) are same. But the correction performance is different. Usually the scrip from Wang et al. (2019) is used to conduct the correction evaluation.

3.6 Confusionset-Guided Fine-Training

As mentioned in Section 3.2, inspired by the random masking strategy in the pretraining stage of BERT, we tailor design a confusionset-guided random masking strategy where the target token x_t will probably be replaced using its corresponding tokens in the confusionset $\mathcal{C}(x_t)$. The masking rate will also be adjusted slightly. Recently we find that confusionset-guided fine-training strategy has been deployed in some related works (Liu et al., 2021; Guo et al., 2021).

After fine-training, we can use the new BERT model to conduct self-supervised/unsupervised detection and unsupervised correction.

4 Experimental Setup

4.1 Settings

The core technical components of our proposed uChecker is a pre-trained Chinese BERT-base model (Devlin et al., 2019). The most important parameters in our framework are the two thresholds θ^u and θ^s and we set them to be 0.1 and 0.4 for unsupervised detection and supervised detection respectively. For the supervised error detection training, Adam optimizer (Kingma and Ba, 2015) is used to conduct the parameter learning and partial of the training dataset from SIGHAN series are employed as the trainset.

4.2 Datasets

The overall statistic information of the datasets used in our experiments are depicted in Table 1. As did in the previous works, we also conduct evaluation on those three datasets: SIGHAN13, SIGHAN14, and SIGHAN15 (Tseng et al., 2015)[2].

4.3 Comparison Methods

Considering that we did not notice some typical unsupervised methods with good results. Therefore, in this Section we introduce several classical and stage-of-the-art supervised approaches for comparisons.

HanSpeller++ employs Hidden Markov Model

[2] http://ir.itc.ntnu.edu.tw/lre/sighan8csc.html

TrainSet	Model	Detection				Correction			
		ACC.	PREC.	REC.	F1	ACC.	PREC.	REC.	F1
SIGHAN13	**Supervised Methods**								
	FASPell (Hong et al., 2019)	-	76.2	63.2	69.1	-	73.1	60.5	66.2
	SpellGCN (Cheng et al., 2020)	-	80.1	74.4	77.2	-	78.3	72.7	75.4
	DCN (Wang et al., 2021)	-	86.8	79.6	83.0	-	84.7	77.7	81.0
	Unsupervised Methods								
	uChecker (Sec.3, Ours)	73.4	75.4	73.4	74.4	70.8	72.6	70.8	71.7
	w/o self-supervised detection	73.9	78.0	73.7	75.8	72.0	75.9	71.8	73.8
	w/o confusionset	73.5	78.2	73.3	75.7	65.5	69.4	65.1	67.2
SIGHAN14	**Supervised Methods**								
	FASPell (Hong et al., 2019)	-	61.0	53.5	57.0	-	59.4	52.0	55.4
	SpellGCN (Cheng et al., 2020)	-	65.1	69.5	67.2	-	63.1	67.2	65.3
	DCN (Wang et al., 2021)	-	67.4	70.4	68.9	-	65.8	68.7	67.2
	Unsupervised Methods								
	uChecker (Sec.3, Ours)	73.3	61.7	61.5	61.6	71.3	57.6	57.5	57.6
	w/o self-supervised detection	68.4	55.3	52.1	53.7	66.7	51.6	48.7	50.1
	w/o confusionset	72.5	62.3	57.3	59.7	58.3	52.9	48.7	50.7
SIGHAN15	**Supervised Methods**								
	*FASPell (Hong et al., 2019)	74.2	67.6	60.0	63.5	73.7	66.6	59.1	62.6
	*Confusionset (Wang et al., 2019)	-	66.8	73.1	69.8	-	71.5	59.5	64.9
	*SoftMask-BERT (Zhang et al., 2020)	80.9	73.7	73.2	73.5	77.4	66.7	66.2	66.4
	*Chunk (Bao et al., 2020)	76.8	88.1	62.0	72.8	74.6	87.3	57.6	69.4
	SpellGCN (Cheng et al., 2020)	-	74.8	80.7	77.7	-	72.1	77.7	75.9
	DCN (Wang et al., 2021)	-	77.1	80.9	79.0	-	74.5	78.2	76.3
	Unsupervised Methods								
	uChecker (Sec.3, Ours)	82.2	75.4	72.0	73.7	79.9	70.6	67.3	68.9
	w/o self-supervised detection	74.0	65.7	61.1	63.3	72.6	62.5	58.1	60.2
	w/o confusionset	81.4	76.2	68.5	72.1	76.5	65.1	58.5	61.6

Table 3: The sentence-level performance on both detection and correction level. Evaluation script is from Hong et al. (2019). * indicates the supervised methods which our unsupervised methods can outperform.

with a reranking strategy to conduct the prediction (Zhang et al., 2015).

LMC presents a model based on joint bi-gram and tri-gram LM and Chinese word segmentation (Xie et al., 2015).

Hybrid utilizes LSTM-based seq2seq framework to conduct generation (Wang et al., 2018) and **Confusionset** introduces a copy mechanism into seq2seq framework (Wang et al., 2019).

FASPell incorporates BERT into the seq2seq for better performance (Hong et al., 2019).

SoftMask-BERT firstly conducts error detection using a GRU-based model and then incorporating the predicted results with the BERT model using a soft-masked strategy (Zhang et al., 2020). Note that the best results of **SoftMask-BERT** are obtained after pre-training on a large-scale dataset with 500M paired samples.

SpellGCN proposes to incorporate phonological and visual similarity knowledge into language models via a specialized graph convolutional network (Cheng et al., 2020).

Chunk proposes a chunk-based decoding method with global optimization to correct single character and multi-character word typos in a unified framework (Bao et al., 2020).

PLOME also employs a confusionset to conduct training of BERT. Besides character prediction, PLOME also introduces pronunciation prediction to learn the misspelled knowledge on phonic level (Liu et al., 2021).

DCN generates the candidate Chinese characters via a Pinyin Enhanced Candidate Generator and then utilizes an attention-based network to model the dependencies between two adjacent Chinese characters (Wang et al., 2021).

4.4 Evaluation Metrics

Following the above mentioned works, we employ character-level and sentence-level **Accuracy**, **Precision**, **Recall**, and **F1-Measure** as the automatic metrics to evaluate the performance of all systems. Besides the official java-based evaluation toolkit (sentence-level) (Tseng et al., 2015)[3], as did in the previous works, we also report and compare the results evaluated by the tools from FASPell (character-level and sentence-level) (Hong et al.,

[3] http://nlp.ee.ncu.edu.tw/resource/csc.html

Parameter	Value	Detection				Correction			
		ACC.	PREC.	REC.	F1	ACC.	PREC.	REC.	F1
θ^s	0.2	76.3	99.0	76.4	86.3	65.5	98.9	65.4	78.7
	0.4	76.8	99.3	76.8	86.6	66.4	99.2	66.2	**79.4**
	0.5	75.3	99.3	75.3	85.6	64.9	99.1	64.6	78.3
	0.6	72.2	99.5	71.9	83.4	62.5	99.4	62.0	76.4
	0.8	61.2	99.4	60.7	75.3	54.6	99.3	54.8	69.8
θ^u	0.0001	21.9	99.1	20.2	33.5	21.9	99.1	20.2	33.5
	0.01	44.3	99.6	43.2	60.2	41.9	99.6	41.7	57.8
	0.1	53.3	98.2	53.0	68.9	49.7	98.1	49.4	**65.7**
	0.5	53.1	97.9	53.0	68.8	48.2	97.7	48.1	64.5
	0.9	44.2	96.7	44.3	60.8	38.4	96.2	38.4	54.9

Table 4: Parameter tuning on devset of SIGHAN2015 (sentence-level). Due to the limited computing resource, we only conduct parameter tuning independently. Finally, we let $\theta^u = 0.1$ and $\theta^s = 0.4$.

2019)[4] and Confusionset (character-level) (Wang et al., 2019)[5].

5 Results and Discussions

5.1 Main Results

Character-level Evaluation Table 2 depicts the evaluation results on character-level on the datasets of SIGHAN13, SIGHAN14, and SIGHAN15. It is obvious most of the baseline methods are published recently and their performance are very strong. More importantly, almost all of the models are supervised learning based approaches and some of them are even trained using external large-scale datasets. Nevertheless, **surprisingly**, our proposed unsupervised framework uChecker has obtained comparable or even better results than those strong baseline methods. Moreover, during the investigation about the evaluation methods, we notice that character-level detection performance of scrips from Hong et al. (2019) and Wang et al. (2019) are same. But the correction performance is different. Usually Wang et al. (2019) is used to conduct the correction evaluation and results reporting.

Sentence-level Evaluation Figure 3 depicts the evaluation results in sentence-level on those three datasets. Evaluation script is employed from Hong et al. (2019) in order to align the results and to conduct comparing fairly. We also find that the official evaluation tool will output large values though the predicted results are same. We are trying to figure out the reasons.

From Table 3 we can observe that our proposed unsupervised framework uChecker has obtained comparable or even better results than those strong baseline methods in the sentence-level as well.

5.2 Parameter Tuning

Considering that the proposed unsupervised model uChecker is simple and straightforward, there are only two hyperparameters in our framework, θ^u and θ^s, which are the threshold values to conduct spelling diagnosis for unsupervised detectors and supervised detectors respectively. And we only need to tune those two parameters. The tuning is conducted on the validation set of SIGHAN2015. Due to the limited computing resource, we only conduct tuning independently. Finally, we let $\theta^u = 0.1$ and $\theta^s = 0.4$.

5.3 Performance on small datasets

It is surprising that uChecker outperforms all the strong supervised baselines on datasets SIGHAN13 in character-level evaluation, as shown in Figure 2. After investigations we believe that the main reason is that the scale of trainset of SIGHAN13 (350) is much smaller than the other two corpora (6,526 and 3,174). This interesting phenomenon also verify the advantages of the unsupervised learning based methods, especially for the task of CSC which is very difficult for collecting real labelled datasets.

5.4 Ablation Analysis

In the main results tables Table 2 and Table 3, we also provide the results of our model uChecker without the components of self-supervised detection and confusionset guided fine-training and decoding. Generally, the experimental results demonstrate that the corresponding components can indeed improve the performance.

[4] https://github.com/iqiyi/FASPell

[5] https://github.com/sunnyqiny/Confusionset-guided-Pointer-Networks-for-Chinese-Spelling-Check

6 Conclusion

In this paper, we propose a framework named **uChecker** to conduct unsupervised spelling error detection and correction. Masked pretrained language models such as BERT are introduced as the backbone model. We also propose a confusionset-guided masking strategy to fine-train the model to further improve the performance. Experimental results on standard datasets demonstrate the effectiveness of our proposed model uChecker.

Acknowledgement

We thank the anonymous reviewers whose suggestions helped clarify this work. This research is supported by the National Natural Science Foundation of China (No.62106105), the CCF-Tencent Open Research Fund (No.RAGR20220122), the Scientific Research Starting Foundation of Nanjing University of Aeronautics and Astronautics (No.YQR21022), and the High Performance Computing Platform of Nanjing University of Aeronautics and Astronautics.

References

Zuyi Bao, Chen Li, and Rui Wang. 2020. Chunk-based chinese spelling check with global optimization. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: Findings, EMNLP 2020, Online Event, 16-20 November 2020*, pages 2031–2040. Association for Computational Linguistics.

Chao-Huang Chang. 1995. A new approach for automatic chinese spelling correction. In *Proceedings of Natural Language Processing Pacific Rim Symposium*, pages 278–283. Citeseer.

Xingyi Cheng, Weidi Xu, Kunlong Chen, Shaohua Jiang, Feng Wang, Taifeng Wang, Wei Chu, and Yuan Qi. 2020. Spellgcn: Incorporating phonological and visual similarities into language models for chinese spelling check. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, ACL 2020, Online, July 5-10, 2020*, pages 871–881. Association for Computational Linguistics.

Kevin Clark, Minh-Thang Luong, Quoc V. Le, and Christopher D. Manning. 2020. ELECTRA: pre-training text encoders as discriminators rather than generators. In *8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, April 26-30, 2020*. OpenReview.net.

Yong Dai, Linyang Li, Cong Zhou, Zhangyin Feng, Enbo Zhao, Xipeng Qiu, Piji Li, and Duyu Tang. 2022. "is whole word masking always better for Chinese BERT?": Probing on Chinese grammatical error correction. In *Findings of the Association for Computational Linguistics: ACL 2022*, pages 1–8, Dublin, Ireland. Association for Computational Linguistics.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT 2019, Minneapolis, MN, USA, June 2-7, 2019, Volume 1 (Long and Short Papers)*, pages 4171–4186. Association for Computational Linguistics.

Tao Ge, Furu Wei, and Ming Zhou. 2018. Reaching human-level performance in automatic grammatical error correction: An empirical study. *CoRR*, abs/1807.01270.

Zhao Guo, Yuan Ni, Keqiang Wang, Wei Zhu, and Guotong Xie. 2021. Global attention decoder for chinese spelling error correction. In *Findings of the Association for Computational Linguistics: ACL/IJCNLP 2021, Online Event, August 1-6, 2021*, volume ACL/IJCNLP 2021 of *Findings of ACL*, pages 1419–1428. Association for Computational Linguistics.

Yuzhong Hong, Xianguo Yu, Neng He, Nan Liu, and Junhui Liu. 2019. Faspell: A fast, adaptable, simple, powerful chinese spell checker based on dae-decoder paradigm. In *Proceedings of the 5th Workshop on Noisy User-generated Text, W-NUT@EMNLP 2019, Hong Kong, China, November 4, 2019*, pages 160–169. Association for Computational Linguistics.

Li Huang, Junjie Li, Weiwei Jiang, Zhiyu Zhang, Minchuan Chen, Shaojun Wang, and Jing Xiao. 2021. Phmospell: Phonological and morphological knowledge guided chinese spelling check. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing, ACL/IJCNLP 2021, (Volume 1: Long Papers), Virtual Event, August 1-6, 2021*, pages 5958–5967. Association for Computational Linguistics.

Masahiro Kaneko, Masato Mita, Shun Kiyono, Jun Suzuki, and Kentaro Inui. 2020. Encoder-decoder models can benefit from pre-trained masked language models in grammatical error correction. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, ACL 2020, Online, July 5-10, 2020*, pages 4248–4254. Association for Computational Linguistics.

Diederik P. Kingma and Jimmy Ba. 2015. Adam: A method for stochastic optimization. In *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*.

Chong Li, Cenyuan Zhang, Xiaoqing Zheng, and Xuanjing Huang. 2021. Exploration and exploitation: Two ways to improve chinese spelling correction models.

In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing, ACL/IJCNLP 2021, (Volume 2: Short Papers), Virtual Event, August 1-6, 2021*, pages 441–446. Association for Computational Linguistics.

Piji Li and Shuming Shi. 2021. Tail-to-tail non-autoregressive sequence prediction for chinese grammatical error correction. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing, ACL/IJCNLP 2021, (Volume 1: Long Papers), Virtual Event, August 1-6, 2021*, pages 4973–4984. Association for Computational Linguistics.

Deng Liang, Chen Zheng, Lei Guo, Xin Cui, Xiuzhang Xiong, Hengqiao Rong, and Jinpeng Dong. 2020. BERT enhanced neural machine translation and sequence tagging model for Chinese grammatical error diagnosis. In *Proceedings of the 6th Workshop on Natural Language Processing Techniques for Educational Applications*, pages 57–66, Suzhou, China. Association for Computational Linguistics.

C.-L. Liu, M.-H. Lai, K.-W. Tien, Y.-H. Chuang, S.-H. Wu, and C.-Y. Lee. 2011. Visually and phonologically similar characters in incorrect chinese words: Analyses, identification, and applications. *ACM Trans. Asian Lang. Inf. Process.*, 10(2):10:1–10:39.

Shulin Liu, Tao Yang, Tianchi Yue, Feng Zhang, and Di Wang. 2021. PLOME: pre-training with misspelled knowledge for chinese spelling correction. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing, ACL/IJCNLP 2021, (Volume 1: Long Papers), Virtual Event, August 1-6, 2021*, pages 2991–3000. Association for Computational Linguistics.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Roberta: A robustly optimized BERT pretraining approach. *CoRR*, abs/1907.11692.

Jonathan Mallinson, Aliaksei Severyn, Eric Malmi, and Guillermo Garrido. 2020. FELIX: flexible text editing through tagging and insertion. In *Findings of the Association for Computational Linguistics: EMNLP 2020, Online Event, 16-20 November 2020*, volume EMNLP 2020 of *Findings of ACL*, pages 1244–1255. Association for Computational Linguistics.

Kostiantyn Omelianchuk, Vitaliy Atrasevych, Artem N. Chernodub, and Oleksandr Skurzhanskyi. 2020. Gector - grammatical error correction: Tag, not rewrite. In *Proceedings of the Fifteenth Workshop on Innovative Use of NLP for Building Educational Applications, BEA@ACL 2020, Online, July 10, 2020*, pages 163–170. Association for Computational Linguistics.

Kevin Parnow, Zuchao Li, and Hai Zhao. 2021. Grammatical error correction as gan-like sequence labeling. In *Findings of the Association for Computational Linguistics: ACL/IJCNLP 2021, Online Event, August 1-6, 2021*, volume ACL/IJCNLP 2021 of *Findings of ACL*, pages 3284–3290. Association for Computational Linguistics.

Yuen-Hsien Tseng, Lung-Hao Lee, Li-Ping Chang, and Hsin-Hsi Chen. 2015. Introduction to SIGHAN 2015 bake-off for chinese spelling check. In *Proceedings of the Eighth SIGHAN Workshop on Chinese Language Processing, SIGHAN@IJCNLP 2015, Beijing, China, July 30-31, 2015*, pages 32–37. Association for Computational Linguistics.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, December 4-9, 2017, Long Beach, CA, USA*, pages 5998–6008.

Baoxin Wang, Wanxiang Che, Dayong Wu, Shijin Wang, Guoping Hu, and Ting Liu. 2021. Dynamic connected networks for chinese spelling check. In *Findings of the Association for Computational Linguistics: ACL/IJCNLP 2021, Online Event, August 1-6, 2021*, volume ACL/IJCNLP 2021 of *Findings of ACL*, pages 2437–2446. Association for Computational Linguistics.

Dingmin Wang, Yan Song, Jing Li, Jialong Han, and Haisong Zhang. 2018. A hybrid approach to automatic corpus generation for chinese spelling check. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing, Brussels, Belgium, October 31 - November 4, 2018*, pages 2517–2527. Association for Computational Linguistics.

Dingmin Wang, Yi Tay, and Li Zhong. 2019. Confusionset-guided pointer networks for chinese spelling check. In *Proceedings of the 57th Conference of the Association for Computational Linguistics, ACL 2019, Florence, Italy, July 28- August 2, 2019, Volume 1: Long Papers*, pages 5780–5785. Association for Computational Linguistics.

Hongfei Wang, Michiki Kurosawa, Satoru Katsumata, and Mamoru Komachi. 2020a. Chinese grammatical correction using bert-based pre-trained model. In *Proceedings of the 1st Conference of the Asia-Pacific Chapter of the Association for Computational Linguistics and the 10th International Joint Conference on Natural Language Processing, AACL/IJCNLP 2020, Suzhou, China, December 4-7, 2020*, pages 163–168. Association for Computational Linguistics.

Yu Wang, Yuelin Wang, Jie Liu, and Zhuo Liu. 2020b. A comprehensive survey of grammar error correction. *CoRR*, abs/2005.06600.

Weijian Xie, Peijie Huang, Xinrui Zhang, Kaiduo Hong, Qiang Huang, Bingzhou Chen, and Lei Huang. 2015.

Chinese spelling check system based on n-gram model. In *Proceedings of the Eighth SIGHAN Workshop on Chinese Language Processing*, pages 128–136, Beijing, China. Association for Computational Linguistics.

Michihiro Yasunaga, Jure Leskovec, and Percy Liang. 2021. Lm-critic: Language models for unsupervised grammatical error correction. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing, EMNLP 2021, Virtual Event / Punta Cana, Dominican Republic, 7-11 November, 2021*, pages 7752–7763. Association for Computational Linguistics.

Ruiqing Zhang, Chao Pang, Chuanqiang Zhang, Shuohuan Wang, Zhongjun He, Yu Sun, Hua Wu, and Haifeng Wang. 2021. Correcting chinese spelling errors with phonetic pre-training. In *Findings of the Association for Computational Linguistics: ACL/IJCNLP 2021, Online Event, August 1-6, 2021*, volume ACL/IJCNLP 2021 of *Findings of ACL*, pages 2250–2261. Association for Computational Linguistics.

Shaohua Zhang, Haoran Huang, Jicong Liu, and Hang Li. 2020. Spelling error correction with soft-masked BERT. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, ACL 2020, Online, July 5-10, 2020*, pages 882–890. Association for Computational Linguistics.

Shuiyuan Zhang, Jinhua Xiong, Jianpeng Hou, Qiao Zhang, and Xueqi Cheng. 2015. Hanspeller++: A unified framework for chinese spelling correction. In *Proceedings of the Eighth SIGHAN Workshop on Chinese Language Processing, SIGHAN@IJCNLP 2015, Beijing, China, July 30-31, 2015*, pages 38–45. Association for Computational Linguistics.

Wei Zhao, Liang Wang, Kewei Shen, Ruoyu Jia, and Jingming Liu. 2019. Improving grammatical error correction via pre-training a copy-augmented architecture with unlabeled data. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT 2019, Minneapolis, MN, USA, June 2-7, 2019, Volume 1 (Long and Short Papers)*, pages 156–165. Association for Computational Linguistics.

Boosting Deep CTR Prediction with a Plug-and-Play Pre-trainer for News Recommendation

Qijiong Liu
The Hong Kong Polytechnic University
jyonn.liu@connect.polyu.hk

Jieming Zhu
Huawei Noah's Ark Lab
jiemingzhu@ieee.org

Quanyu Dai
Huawei Noah's Ark Lab
quanyu.dai@connect.polyu.hk

Xiaoming Wu*
The Hong Kong Polytechnic University
xiao-ming.wu@polyu.edu.hk

Abstract

Understanding textual content is critical to improving the quality of news recommendation. To achieve this goal, recent studies have proposed to apply pre-trained language models (PLMs) such as BERT for semantic-enhanced news recommendation. Despite their great success in offline evaluation, it is challenging to apply such large PLMs in real-time ranking tasks due to the stringent latency requirements in model updating and inference. To bridge this gap, we propose a plug-and-play pre-trainer, namely PREC, to learn both user and news encoders through multi-task pre-training. Instead of directly leveraging sophisticated PLMs for end-to-end inference, we focus on how to use the cached user and item representations to boost the performance of traditional ID-based models for click-through-rate prediction. This enables efficient online inference as well as compatibility to the widely-used models in industry, which would significantly ease the practical deployment. We validate the effectiveness of PREC through both offline evaluation on public datasets and online A/B testing in an industrial system.

1 Introduction

Personalized news recommendation has become a ubiquitous channel in various online applications, such as Google News and MSN News, which helps users discover their interested news information. To deal with the massive amounts (usually millions) of daily news, industrial recommender systems usually apply a multi-stage recommendation pipeline as illustrated in Figure 1. It generally involves two phases, matching and ranking (Covington et al., 2016). The matching phase first generates hundreds or thousands of news candidates from multiple channels (e.g., popularity-based channel, content-based channel (Wu et al., 2019a,b), and collaborative filtering channel (Linden et al., 2003; Sedhain et al., 2014)). Subsequently, the ranking phase employs a news ranking task, such as click-through rate (CTR) prediction, to rank the candidate news and finally return the top dozens of news to the user interface.

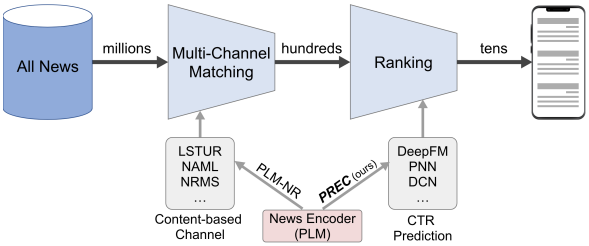

Figure 1: A multi-stage pipeline for news recommendation.

The matching and ranking phases have different goals and requirements. Concretely, the matching phase aims for efficient and high-recall candidate retrieval from millions of news corpus. Therefore, most studies construct two-tower networks (e.g., NPA (An et al., 2019), LSTUR (Wu et al., 2019a), and NRMS (Wu et al., 2019b)) to learn user representations and item representations separately, and then apply simple dot product to compute similarity scores. As such, both user and item representations can be pre-computed offline and cached online for fast nearest neighbour search (e.g., using Faiss (Johnson et al., 2019)). In contrast, the ranking phase targets at accurate scoring of each candidate item based on CTR prediction and thus requires networks to capture complex feature interactions between users and news (e.g., DeepFM (Guo et al., 2018), PNN (Qu et al., 2016), and DCN (Wang et al., 2017)). These widely adopted CTR prediction models in industry are usually small (e.g., $3 \sim 5$ layers) to meet the latency requirements for online inference and enable frequent model updates (e.g., hourly or daily).

To leverage powerful pre-trained language models (PLMs) to better capture the semantics of news

* Corresponding author.

Proceedings of the 29th International Conference on Computational Linguistics, pages 2823–2833
October 12–17, 2022.

content, some recent studies (Zhang et al., 2021a; Wu et al., 2021b,a) propose the use of PLMs for news recommendation. For example, as depicted in Figure 1, PLM-NR (Wu et al., 2021a) replaces original news encoders (e.g., CNN in Krizhevsky et al. (2012) and multi-head attention in Vaswani et al. (2017)) with PLMs such as BERT (Devlin et al., 2019) to empower existing content-based matching models (e.g., LSTUR (An et al., 2019), NAML (Wu et al., 2019a), and NRMS (Wu et al., 2019b)). Yet, these studies typically adopt the common pretrain-finetune strategy to train PLM-based news encoder and user encoder jointly, which is extremely time-consuming given the massive amount of click data. As a concrete example illustrated in Figure 2, we estimate the updating time of the end-to-end model and the hierarchically decoupled model on MIND (Wu et al., 2020). Finetuning "PLM-NR + NRMS" with the MIND click data (4M clicks with 1M users and 100K news) takes 2800 minutes for one epoch. This might be tolerable during matching as reported in Wu et al. (2021a), where user and item representations are pre-computed offline, but is absolutely unacceptable for CTR prediction tasks.

In this paper, we explore the use of PLMs for CTR prediction tasks and propose a plug-and-play pre-trainer, namely PREC, to learn both user and news encoders through multi-task pre-training. In particular, our PREC model has the following key advantages: 1) This is the first work to integrate both news pre-trainer and user pre-trainer for recommendation. The former fuses multi-view news features (e.g., title, abstract, category) for representation learning, while the latter learns user representations from historical interaction sequences depending on the learned news representations and iter-dependencies among them. 2) The PREC model is constructed in a hierarchical decoupled manner, bringing rich news contents and deep user interests to CTR models while meeting the requirements of both inference and updating efficiency. As shown in Figure 2, PREC's model updating time only reaches about 50 minutes, which is 50+ times faster than PLM-NR under the same setting. 3) The decoupled design makes PREC easily compatible with various existing CTR prediction models (e.g., PNN (Qu et al., 2016), DCN (Wang et al., 2017), DeepFM (Guo et al., 2018)), where user and news representations learned from PREC could be used as features or embedding initialization for downstream models. This allows PREC to be easily deployed in industrial recommender systems.

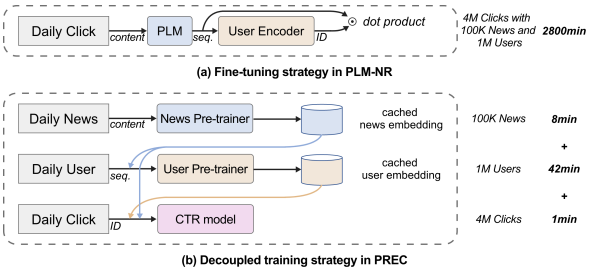

Figure 2: Model updating strategy of PLM-NR-like model (left) and our PREC-like model (right) on the MIND dataset. The "content", "seq", and "ID" symbols respectively denote news content, user browsing sequence, and ID-based features, which are fed to the following model. Please refer to subsubsection 4.1.2 for detailed settings.

We conduct extensive experiments to validate the effectiveness of PREC on two open benchmark datasets for news recommendation, MIND (Wu et al., 2020) and Adressa (Gulla et al., 2017). The experiments show that our pre-trainer PREC can consistently boost the performance of existing lightweight CTR prediction models, even outperforming some state-of-the-art heavyweight models. In addition, we have deployed PREC in the production system and an online A/B test shows that PREC leads to a 2.4% improvement on the overall CTR metric.

2 Related Work

News recommendation has become increasingly popular in recent years. Models based on deep neural networks (Wang et al., 2018; Wu et al., 2019b) have been proposed to learn news and user representations by leveraging CNN (Krizhevsky et al., 2012) or attention mechanism (Vaswani et al., 2017). These end-to-end models focus on solving recommendation task and typically have limited ability in semantic understanding. Most recently, PLMs such as BERT (Devlin et al., 2019) have shown promising results in news recommendation (Zhang et al., 2021a; Wu et al., 2021c; Li et al., 2022). Some works (Wu et al., 2021a,b; Yu et al., 2021) leverage PLMs to learn news embeddings by exploring out-domain knowledge and learning news semantics. These models can be deployed in matching stage, but not compatible to the ranking stage due to intolerant model updating time. Some use PLMs to learn user representations (Wu et al., 2021c; Sun et al., 2019; Chen et al., 2019) by mod-

eling user interests with historical news sequences. Besides, we notice that SpeedyFeed (Xiao et al., 2022) is presented to semi-decouple PLM-based recommender models for faster training, but the model updating time is still too long.

While pre-trained models have shown great promise in content understanding, lightweight CTR models (Wang et al., 2017; Guo et al., 2018; Lian et al., 2018) still dominate industrial applications due to its high efficiency, where PLMs cannot be deployed due to its large size. Our proposed pre-trainer can work seamlessly with these industrial CTR models and provide them semantic news embeddings and user interest knowledge with almost no overhead in inference time and deployment cost.

Model Pre-Training. The pioneer works (Mikolov et al., 2013; Pennington et al., 2014) introduced pre-training for natural language understanding by providing a static representation for each word. Later, some methods (Peters et al., 2018; Devlin et al., 2019) proposed to dynamically generate word representations according to the context, which effectively resolves the ambiguity problem. Very recently, the huge success of BERT has led to a surge of interest in pre-training models. A line of works (Zhou et al., 2020; Xie et al., 2020; Wu et al., 2021a,c) have proposed to use pre-training for new recommendation. However, the pre-training tasks for news recommendation are under-explored. In this paper, we propose multiple tasks for multi-view news and user pre-training.

3 The Proposed Method

In this section, we present our proposed cascaded pre-trainer for news recommendation (PREC). We first introduce the model framework, pre-training tasks and pre-training strategy. Then, we describe how to apply the pre-trainer for downstream tasks.

3.1 Model Framework

Figure 3 presents the model framework of our proposed cascaded pre-trainer PREC, which consists of a news pre-trainer and a user pre-trainer. Both news and users contain multiple feature sets, i.e., multiple views. For example, a news has title, entities and abstract, and a user is usually characterized by her/his historical browsed news, location, personal information, and so on. To capture the rich semantics from these multi-view data, PREC is designed to capture both intra-view and inter-view knowledge based on Transformer, with the news pre-trainer aimed at learning the deep semantic meanings of the textual content while the user pre-trainer targeting at capturing the inherent user interests. Since browsing history is one of the most significant views of users, the user pre-trainer relies on the news embeddings learned from the news pre-trainer. Thus, our model is built and trained in a cascaded manner.

3.1.1 News Pre-trainer

We assume each news contains n views (e.g., title, entities, abstract) and each view contains a sequence of tokens. We use $\boldsymbol{t}_i = \left[t_{i,1}, ..., t_{i,|\boldsymbol{t}_i|}\right]$ to represent the token sequence of the i-th view, where $i \in \{1, ..., n\}$ and $|\boldsymbol{t}_i|$ is the total length of the view. We propose a news pre-trainer to comprehend the deep semantic meanings of news by taking a concatenated sequence of the multi-view content as input. Following BERT (Devlin et al., 2019), the input sequence starts with a special token $\langle$CLS$\rangle$, and views are separated by another special token $\langle$SEP$\rangle$. For simplicity, these two special tokens are omitted in the following description. After concatenating n views, the input token sequence is represented as $\boldsymbol{t} = \left[t_{1,1}, ..., t_{1,|t_1|}, ..., t_{n,1}, ..., t_{n,|t_n|}\right]$, and the token embedding can be obtained through an embedding layer as follows:

$$\boldsymbol{E}^t_{token} = [\boldsymbol{e}^t_1, \boldsymbol{e}^t_2, ..., \boldsymbol{e}^t_s] \in \mathbb{R}^{s \times d}, \tag{1}$$

where $s = \sum_i^n |\boldsymbol{t}_i|$ is the sequence length and d is the embedding dimension. We also use position embedding to encode the positional information of a token, and view embedding to distinguish between different views. We denote the position embedding and view embedding of a news as $\boldsymbol{E}^t_{pos}$ and $\boldsymbol{E}^t_{view}$, respectively. For a given token, the input representation is obtained by summing up the corresponding token, position embedding and view embedding. Then, given a news, the integrated input representation of the news pre-trainer is obtained as follows:

$$\boldsymbol{E}^t = \boldsymbol{E}^t_{token} + \boldsymbol{E}^t_{pos} + \boldsymbol{E}^t_{view}. \tag{2}$$

The news pre-trainer consists of two sub-modules, news transformer and news aggregator. The news transformer first learns the refined token representations $\bar{\boldsymbol{E}}^t$ of news content through multiple Transformer layers, and then the news aggregator combines the token representations into a unified news representation $\bar{\boldsymbol{t}}$. The learned news

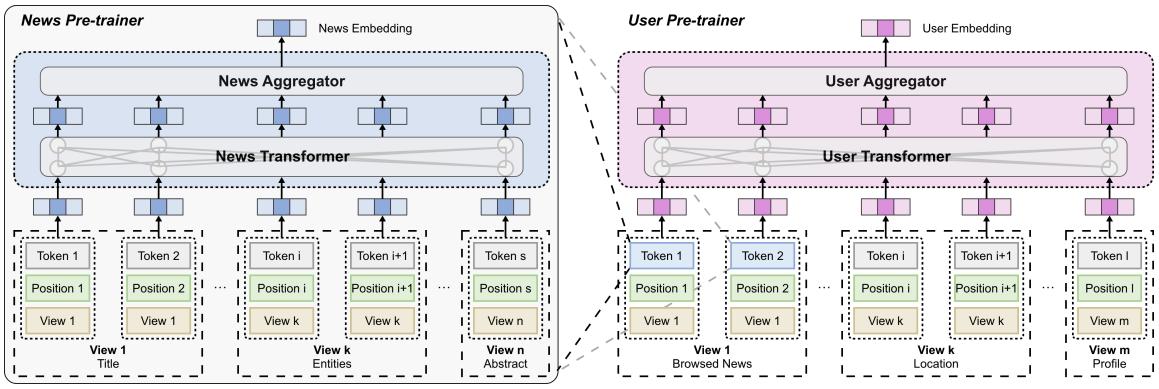

Figure 3: The architecture of our PREC model.

representation will be used for news token embedding in the user pre-trainer and the training of downstream CTR models.

3.1.2 User Pre-trainer

Similarly, users are characterized by m views, e.g., historical browsed news, location, and user profile. We use $\boldsymbol{u}_i = [u_{i,1}, ..., u_{i,|\boldsymbol{u}_i|}]$ to represent the i-th view of a user, where $i \in \{1, ..., m\}$, and $|\boldsymbol{u}_i|$ is the total length of the view. User pre-trainer has a similar model framework as the news pre-trainer, which consists of an embedding layer, multiple Transformer layers, and a aggregator. It takes a sequence of concatenated multi-view user information as input, which is represented as $\boldsymbol{u} = [u_{1,1}, ..., u_{1,|u_1|}, ..., u_{m,1}, ..., u_{m,|u_m|}]$. Firstly, the input token sequence of a user is transformed into three embedding sequences, including token, position, and view sequences, through an embedding layer. Specifically, the token embedding sequence is represented as follows:

$$\boldsymbol{E}^u_{token} = [\boldsymbol{e}^u_1, \boldsymbol{e}^u_2, ..., \boldsymbol{e}^u_l] \in \mathbb{R}^{l \times d}, \quad (3)$$

where $l = \sum_i^m |\boldsymbol{u}_i|$ is the sequence length. Similarly, we denote the corresponding position embedding and view embedding of a user as $\boldsymbol{E}^u_{pos}$ and $\boldsymbol{E}^u_{view}$, respectively. Then, the integrated input user representation for the user pre-trainer is obtained by summing the three types of embeddings:

$$\boldsymbol{E}^u = \boldsymbol{E}^u_{token} + \boldsymbol{E}^u_{pos} + \boldsymbol{E}^u_{view}. \quad (4)$$

As demonstrated by existing work (An et al., 2019; Wu et al., 2019b), users' historical browsed news is key to discover user interests. For training efficiency, we initialize the token embeddings of user browsed news with the learned news embeddings from the news pre-trainer, and keep them fixed during training.

Then, the user transformer explores users' potential interests through multiple Transformer layers and produces the refined token representations $\bar{\boldsymbol{E}}^u$, and the user aggregator fuses the refined representations into a single user representation $\bar{\boldsymbol{u}}$. Note that we use average pooling as our news and user aggregators as in Zhang et al. (2021a).

3.2 Pre-training Tasks

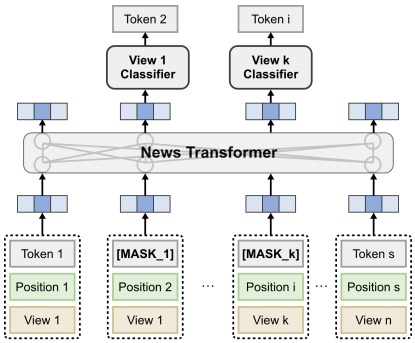

(a) Masked news token prediction.

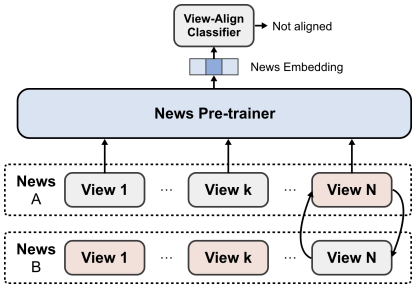

(b) News view alignment.

Figure 4: Two pre-training tasks in PREC.

Masked News Token Prediction (MNTP) is designed to learn context-aware token representations. The framework is shown in Figure 4(a). In this task, we first randomly sample a portion (15% in our experiments) of tokens in each sequence $\boldsymbol{t}$ and

replace them with other tokens. Specifically, for each sampled token, there is an 80% probability of being replaced with a unique identifier $\langle$`MASK_i`$\rangle$, a 10% possibility of being substituted with a random token from the vocabulary, and another 10% probability of remaining unchanged. We then conduct a classification task on the sampled tokens with a multi-layer perceptron classifier by taking the corresponding refined token embedding from the news transformer as input. Thus, the MNTP task is to minimize the following negative log-likelihood loss:

$$L_{mntp}(\boldsymbol{\theta}, \boldsymbol{\beta}_1) = -\mathbb{E}_{\boldsymbol{t}\sim D^t} \log P(t_{i,j}|\boldsymbol{t}_{\backslash(i,j)}, \boldsymbol{\theta}, \boldsymbol{\beta}_1), \tag{5}$$

where $\boldsymbol{\theta}$ denotes the model parameters of the news transformer and aggregator, $\boldsymbol{\beta}_1$ denotes the model parameters of the classifier, D^t is the training set of the news pre-trainer, and $P(t_{i,j}|\boldsymbol{t}_{\backslash(i,j)}; \boldsymbol{\theta}, \boldsymbol{\beta}_1)$ is the probability of correctly predicting the current token $t_{i,j}$ given other tokens $\boldsymbol{t}_{\backslash(i,j)}$.

News View Alignment (NVA) aims to capture inter-view relations. The framework is presented in Figure 4(b). In this task, we manually construct out-of-alignment news by replacing some views of the considered news with those of a randomly sampled one, and treat them as negative samples. More precisely, for a piece of news, it will undergo one of the following two operations randomly: (a) remains unchanged (labeled as 1); (b) constructed as out-of-alignment news (labeled as 0). Then, the processed token sequence $\boldsymbol{t}'$ is fed into the news pre-trainer to generate the unified news representation $\bar{\boldsymbol{t}}$. We perform a binary classification task with $\bar{\boldsymbol{t}}$ as input by optimizing the following cross-entropy loss:

$$L_{nva}(\boldsymbol{\theta}, \boldsymbol{\beta}_2) = -\mathbb{E}_{\boldsymbol{t}\sim D^t} \log P(l|\boldsymbol{t}', \boldsymbol{\theta}, \boldsymbol{\beta}_2), \tag{6}$$

where $\boldsymbol{\beta}_2$ denotes the model parameters of the binary classifier, $l \in \{0, 1\}$ represents the types of news samples, and $\boldsymbol{t}'$ is the reconstructed news after the random operations.

News Category Prediction (NCP) is proposed to capture global knowledge of news. The extracted news embedding $\bar{\boldsymbol{t}}$ is leveraged to predict the category of the considered news. The loss function of this task is as follows:

$$L_{ncp}(\boldsymbol{\theta}, \boldsymbol{\beta}_3) = -\mathbb{E}_{\boldsymbol{t}\sim D^t} \log P(c|\boldsymbol{t}, \boldsymbol{\theta}, \boldsymbol{\beta}_3), \tag{7}$$

where $\boldsymbol{\beta}_3$ denotes the model parameters of the category classifier, and c is the news category.

We also design **Masked User Token Prediction (MUTP)** task and **User View Alignment (UVA)** task for the user pre-trainer, which are similar to the MNTP task and the NVA task, respectively. The loss functions of the MUTP task and the UVA task are as follows:

$$L_{mutp}(\boldsymbol{\phi}) = -\mathbb{E}_{\boldsymbol{u}\sim D^u} \log P(u_{i,j}|\boldsymbol{u}_{\backslash(i,j)}, \boldsymbol{\phi}, \boldsymbol{\varphi}_1), \tag{8}$$

$$L_{uva}(\boldsymbol{\phi}) = -\mathbb{E}_{\boldsymbol{u}\sim D^u} \log P(l|\boldsymbol{u}', \boldsymbol{\phi}, \boldsymbol{\varphi}_2), \tag{9}$$

where $\boldsymbol{\phi}$ denotes the model parameters of the user transformer and aggregator, $\boldsymbol{\varphi}_1$ and $\boldsymbol{\varphi}_2$ denote the model parameters of the classifier for the MUTP task and the NUA task respectively, D^u is the training set of the user pre-trainer, (i, j) is the index of the masked token, and $l \in \{0, 1\}$ represents whether the views of the reconstructed token sequence $\boldsymbol{u}'$ are aligned or not.

3.3 Pre-training Strategy

Stage One. The performance of the user pre-trainer heavily relies on the quality of news embeddings, thus we first conduct pre-training of the news pre-trainer to obtain informative news embeddings. In this stage, MNTP, NVA, and NCP tasks are performed to capture intra-view and inter-view semantic meanings of news, and the overall loss is formulated as:

$$L_{NP} = \omega_{mntp} L_{mntp} + \omega_{nva} L_{nva} + \omega_{ncp} L_{ncp}, \tag{10}$$

where ω_{mntp}, ω_{nva}, and ω_{ncp} are hyper-parameters for balancing the losses of different tasks.

Stage Two. After obtaining news embeddings from the news pre-trainer, we perform the MUTP and UVA tasks to optimize the user pre-trainer with the following loss:

$$L_{UP} = \omega_{mutp} L_{mutp} + \omega_{uva} L_{uva}, \tag{11}$$

where ω_{mutp} and ω_{uva} are hyper-parameters for balancing the losses of the two tasks.

This two-stage pre-training strategy has the following advantages. 1) It greatly simplifies the optimization of PREC by decoupling the optimization of the news pre-trainer and user pre-trainer. 2) It allows to update the two pre-trainers in an asynchronous manner, which nicely satisfies the requirements of real industrial scenarios where user representations are likely to be updated more frequently due to the constant change of user interests.

Table 1: Dataset statistics after preprocessing.

Number	MIND		Adressa	
	small	large	1Week	4Week
#News	65,238	104,151	81,018	81,018
#Users	94,057	750,434	214,464	400,279
#Clicks	347,727	3,958,501	354,181	877,605
#Samples	8,584,442	97,592,931	1,770,905	4,388,025

3.4 CTR Prediction with PREC

The learned news and user representations from PREC are leveraged to boost the performance of deep CTR prediction models. Generally, deep CTR models, such as PNN (Qu et al., 2016), DeepFM (Guo et al., 2018), and DCN (Wang et al., 2017), take multiple user-side (e.g., *user id*, *browsed news ids*, *browsed news title sequence*) and news-side (e.g., *news id*, *title*) features as input. These raw features are usually first transformed into multi-field categorical format through a feature engineering module before being fed to a CTR model. Then, the input features are transformed into continuous embedding vectors through an embedding layer of the model (Zhang et al., 2021b). We use the pre-trained representations to initialize the embedding vectors of news id and user id to enrich the feature inputs of CTR models. Note that we keep the pre-trained news and user representations fixed during model training and add additional transformation matrices to project them into the same space of the embedding vectors of other features. The two transformation matrices are part of trainable model parameters and updated during optimization.

Table 2: Comparison results between baseline models and PREC-boosted models on the MIND-small and Adressa-1Week datasets.

MIND-small				
Method	DCN	DCN$_{+NP}$	DCN$_{+NP+UP}$	Improv.
AUC	65.07	66.63	**67.47**	3.69%
MRR	33.12	33.62	**34.88**	5.31%
nDCG@5	34.02	34.95	**36.20**	6.41%
nDCG@10	40.06	41.23	**42.33**	5.67%
Method	PNN	PNN$_{+NP}$	PNN$_{+NP+UP}$	Improv.
AUC	61.80	65.24	**66.55**	7.69%
MRR	30.11	32.42	**33.18**	10.20%
nDCG@5	30.65	30.02	**34.86**	13.74%
nDCG@10	36.93	40.14	**40.89**	10.72%
Method	DeepFM	DeepFM$_{+NP}$	DeepFM$_{+NP+UP}$	Improv.
AUC	61.86	65.46	**65.98**	6.66%
MRR	29.77	33.26	**33.42**	12.26%
nDCG@5	30.15	34.31	**34.92**	15.82%
nDCG@10	36.74	40.73	**40.94**	11.43%
Adressa-1Week				
Method	DCN	DCN$_{+NP}$	DCN$_{+NP+UP}$	Improv.
AUC	75.86	82.39	**83.78**	10.44%
MRR	93.26	95.07	**95.61**	2.52%
nDCG@5	95.30	96.60	**97.31**	2.11%
nDCG@10	95.51	96.81	**97.32**	1.90%
Method	PNN	PNN$_{+NP}$	PNN$_{+NP+UP}$	Improv.
AUC	71.21	79.64	**81.92**	15.04%
MRR	92.05	94.06	**95.09**	3.30%
nDCG@5	94.67	95.14	**96.80**	2.25%
nDCG@10	94.83	95.74	**96.89**	2.17%
Method	DeepFM	DeepFM$_{+NP}$	DeepFM$_{+NP+UP}$	Improv.
AUC	73.40	79.64	**81.14**	10.54%
MRR	92.43	94.29	**94.68**	2.43%
nDCG@5	94.43	95.99	**96.07**	1.74%
nDCG@10	94.83	96.22	**96.42**	1.68%

4 Experiments

4.1 Experimental Settings

We conduct experiments on two large real-world news recommendation datasets: MIND (including small and large versions) (Wu et al., 2020) and Adressa (including the 1-week and 4-week versions) (Gulla et al., 2017). Our experiments include two phases: pre-training and plug-and-play. For the MIND dataset, we use news title and abstract as news views, news category as the label in the NCP task, and user history as user view. As users' profiles are missing, we omit the UVA task on MIND. We set $\omega_{mntp} = \omega_{nva} = \omega_{ncp} = \omega_{mutp} = 1, \omega_{uva} = 0$. For the Adressa dataset, we take the news title, description, and keywords as news view and use user history, location (country, region, and city), and device information as user view. Since news category information is missing, we omit the NCP task on Adressa. We set $\omega_{mntp} = \omega_{nva} = \omega_{mutp} = \omega_{uva} = 1, \omega_{ncp} = 0$. For the pre-training phase, we first split the news set into 4:1 as the training set and validation set (used for early-stopping) respectively, and use them to train the news pre-trainer. We then split the user set in the same way and train the user pre-trainer. In the plug-and-play phase for CTR prediction, we follow the same splitting strategy as in Wu et al. (2020) on the MIND dataset. For the Adressa dataset, we perform 1:4 negative sampling since it has only positive interaction samples. For Adressa-1Week,

Table 3: Performance comparison among different approaches.

Method		End-to-end							Pretrain-finetune PLM		Plug-and-play PLM	
		PNN	DeepFM	DCN	NAML	LSTUR	NRMS	FIM	UNBERT	$NRMS_{PLM\text{-}NR}$	DCN_{BERT}	DCN_{PREC} (ours)
MIND-small	AUC	61.80	61.86	65.07	**66.12**	65.87	65.63	65.34	**67.62**	/	66.12	**67.47**
	MRR	30.11	29.77	**33.12**	31.53	30.78	30.96	30.64	**31.72**		33.53	**34.88**
	nDCG@5	30.65	30.15	34.02	**34.88**	33.95	34.13	33.61	**34.75**		34.77	**36.20**
	nDCG@10	36.93	36.74	40.06	**41.09**	40.15	40.52	40.16	**41.02**		40.95	**42.33**
MIND-large	AUC	66.33	67.00	66.52	66.46	67.08	67.66	**67.87**	**70.68**	70.64	68.46	**69.12**
	MRR	32.32	32.77	32.32	32.75	32.36	33.25	**33.46**	**35.68**	35.39	34.05	**34.47**
	nDCG@5	35.11	35.59	35.08	35.66	35.15	36.28	**36.53**	**39.13**	38.71	37.17	**37.70**
	nDCG@10	40.83	41.30	40.83	41.40	40.93	41.98	**42.21**	**44.78**	44.38	42.82	**43.35**
		PNN	DeepFM	DCN	AutoInt	xDeepFM	FiBiNET	GNUD			DCN_{BERT}	DCN_{PREC} (ours)
Adressa-1Week	AUC	71.21	73.40	75.86	71.80	**75.99**	71.21	72.03	/	/	77.30	**83.78**
	MRR	92.05	92.43	**93.26**	91.90	93.18	91.08	92.24			93.66	**95.61**
	nDCG@5	94.67	94.43	**95.30**	93.74	95.02	91.19	94.78			95.70	**97.31**
	nDCG@10	94.83	94.83	**95.51**	94.31	95.35	92.82	94.91			95.83	**97.32**
Adressa-4Week	AUC	68.81	70.21	**71.55**	69.41	71.08	66.71	69.03	/	/	74.18	**79.09**
	MRR	90.71	90.97	**91.80**	91.21	91.33	89.99	91.62			92.47	**93.90**
	nDCG@5	92.35	91.51	94.03	93.54	92.76	91.42	**95.28**			94.59	**95.67**
	nDCG@10	93.03	92.72	94.32	93.86	93.47	92.30	**95.04**			94.70	**95.86**

we use the first 5 days' interaction data as user history, the 6-th day's data as training set, and the 7-th day's data for validation (by randomly sampling 20% data) and testing (the remaining 80% data), respectively. For Adressa-4Week, we construct user history with the first 24 days' data; the following 2 days' data are used as the training set, and the 20% and 80% of the last 2 days' data are used for validation and testing, respectively. The statistics of the datasets are summarized in Table 1.

4.1.1 Evaluation Protocols

We follow common practice Wu et al. (2020, 2021a) to evaluate the effectiveness of our method with the widely-used ranking metrics: AUC, MRR, nDCG@5, and nDCG@10.

4.1.2 Implementation Details

We tokenize the title, abstract, and text descriptions in MIND and Adressa with the vocabulary provided by BERT and NordicBERT[1], respectively. Note that entities and locations are special words, so we do not tokenize them. We follow the setting of BERT (Devlin et al., 2019) and output 768-dimensional user and news vectors. For the news pre-trainer, we use 3 Transformer layers. For the user pre-trainer, we use 6 Transformer layers. We initialize the parameters as in BERT wherever possible. We apply the open-source FuxiCTR (Zhu et al., 2021) library for implementing downstream CTR prediction tasks. We release the source code for reproducibility[2].

As illustrated in Figure 2, we make a fair comparison of the running time of $NRMS_{PLM\text{-}NR}$ and DCN_{PREC}. The number of Transformer layers, attention heads, and dimension size of BERT (in PLM-NR), news and user pre-trainer (in PREC) are set to 12, 12, and 768, respectively. The news title and user sequence length are set to 50 and 25 respectively. We record the model updating time on a single NVIDIA GeForce RTX 3090 device.

4.1.3 Compared Models

We take 13 existing models as our baselines, including typical CTR prediction models (i.e., DCN (Wang et al., 2017), DeepFM (Guo et al., 2018), PNN (Qu et al., 2016) and xDeepFM (Lian et al., 2018), AutoInt (Song et al., 2019), FiBiNET (Huang et al., 2019)), neural news recommendation models (i.e., LSTUR (An et al., 2019), NAML (Wu et al., 2019a), NRMS (Wu et al., 2019b), GNUD (Hu et al., 2020), FIM (Wang et al., 2020)), and PLM-based models (i.e., UNBERT (Zhang et al., 2021a), PLM-NR (Wu et al., 2021a)). We also compare with the naive-BERT model with 3 Transformer layers (same as ours) to extract the news representations for DCN, denoted as DCN_{BERT}. In this setting, only the title view is used and the masked language modeling task is applied in news content pre-training. The repre-

[1] https://github.com/certainlyio/nordic_bert

[2] https://github.com/Jyonn/PREC

Table 4: Ablation results on the Adressa-1Week dataset. * indicates that only one mask identifier ⟨MASK⟩ is used for different views. The *exp.* column indicates experiments with different settings.

	View in NP			NP		UP		Method	
exp.	title	desc	key	MNTP	NVA	MUTP	UVA	DCN	PNN
a	-	-	-	-	-	-	-	75.86	71.21
b	✓	✓	✓	✓	-	-	-	80.06	77.82
c	✓	✓	✓	✓	✓	-	-	82.39	79.64
d	✓	✓	✓	✓	✓	✓	-	82.95	80.73
e	✓	✓	✓	✓	✓	✓	✓	83.78	81.92
f	✓	-	-	✓	-	-	-	77.30	74.77
g	✓	✓	-	✓	-	-	-	78.84	76.08
h	✓	✓	✓	*	-	-	-	79.47	77.25

sentations generated by the naive-BERT model are fixed and not fine-tuned in the downstream task. In particular, we apply PREC on DCN, which is widely used in industry, to validate its effectiveness and efficiency. We also test PREC on DeepFM and PNN to demonstrate its broad applicability.

4.2 Performance Comparison

Table 2 shows the comparison between the baseline models and PREC- boosted models, where $_{+NP}$ means news embeddings are generated by the news pre-trainer and $_{+UP}$ means user embeddings are generated by the user pre-trainer. We can observe that using pre-trained news and user representations can enhance the performance of each downstream CTR model. Table 3 provides the comprehensive results of different models on four datasets. The results show that: 1) Our PREC-based model achieves competitive results with the pretrain-finetune PLM-based models, which achieve state-of-the-art performance due to the fine-tuning process and cannot be deployed in the ranking stage. 2) With the pre-trained representations as input features, the lightweight DCN model can outperform sophisticated neural news recommendation models such as FIM and GNUD, showing the effectiveness of our approach.

4.3 Ablation Study

As demonstrated in Table 4, we explore different variants during PREC pre-training. We find that 1) based on *exp. a to e*, each pre-training task makes a considerable improvement; 2) based on *exp. bfg*, each view offers better comprehension to the news pre-trainer; 3) compared with applying one mask identifier (*exp. h*), employing the view-specific mask identifier (*exp. b*) achieves better performance. We also conduct experiments on the effect of the number of Transformer layers. As depicted in Table 5, the performance improves when the number of layers increases, since more Transformer layers can capture deeper semantic meanings. However, the pre-training time cost also grows. Hence, we choose 3 layers as a trade-off between performance and efficiency.

Table 5: Influence of the Transformer layers of the news pre-trainer on the Adressa-1Week dataset. Time (h) denotes the pre-training time in hours.

Layers of NP	1	3	6	9	12
AUC	81.34	82.39	82.60	83.02	83.66
Time (h)	12.0	13.3	14.0	15.9	18.3
Improv. per Layer	-	**0.525**	0.252	0.210	0.211

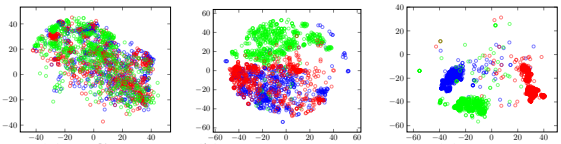

(a) DCN (b) *NP w/o NCP* (c) *NP*

Figure 5: Visualization of the news embeddings of the MIND-small dataset. Different color indicates different news category.

4.4 Visualization of News Embeddings

We use t-SNE (Van der Maaten and Hinton, 2008) to visualize the news embeddings learned by different settings. As depicted in Figure 5, we randomly select three categories and observe the embedding distribution. It can be seen that the news embeddings learned by the DCN model (Figure 5(a)) are scattered while the news features obtained by the news pre-trainer (Figure 5(b)) are clustered. When the NCP task is adopted Figure 5(c), the news features are more clustered.

4.5 Online A/B Testing

We have deployed our news pre-trainer in Huawei's news recommender system, serving millions of users daily, to perform an online A/B test. CTR prediction is applied in the ranking phase of recommendation, which takes tens of user features (e.g., city, tags clicked in recent 1/3/7 days) and news features (e.g., category, topic, tags, entities) as input and outputs the predicted click probability of each

user-news pair. The base model deployed online is an optimized variant of the DeepFM model (Guo et al., 2018) called FINAL and has a stringent latency requirement (less than 50ms) for each request. To improve the model performance, yet keep the efficiency of model inference, we apply the PREC pre-trainer to obtain cached representation vectors of users and news. Then, these vectors are used to initialize the embedding layers for training the ranking model. The decoupling allows asynchronous updates of both modules: the PREC pre-trainer is updated on a daily basis while the ranking model is updated on a minute level to adapt to new data quickly. During the one-week online A/B test, the PREC-boosted ranking model has achieved an average improvement of 2.4% in CTR over the baseline, which is significant in our application scenario.

5 Conclusion

In this paper, we have developed a plug-and-play pre-trainer called PREC to boost the performance of traditional ID-based CTR models for news recommendation. PREC is built on Transformer layers, utilizes multi-view features of news and users, and is trained with tailored pre-training tasks to learn semantic news and user representations. Aside from its ability in content understanding, PREC can be easily deployed in industrial recommender systems to improve CTR prediction. Offline experiments on public benchmark datasets and online A/B testing in industrial recommender systems demonstrate the effectiveness and efficiency of PREC.

Acknowledgments

The authors would like to thank the anonymous reviewers for their helpful comments.

References

Mingxiao An, Fangzhao Wu, Chuhan Wu, Kun Zhang, Zheng Liu, and Xing Xie. 2019. Neural news recommendation with long- and short-term user representations. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 336–345, Florence, Italy. Association for Computational Linguistics.

Xusong Chen, Dong Liu, Chenyi Lei, Rui Li, Zheng-Jun Zha, and Zhiwei Xiong. 2019. Bert4sessrec: Content-based video relevance prediction with bidirectional encoder representations from transformer. In *Proceedings of the 27th ACM International Conference on Multimedia*, pages 2597–2601.

Paul Covington, Jay Adams, and Emre Sargin. 2016. Deep neural networks for youtube recommendations. In *Proceedings of the 10th ACM conference on recommender systems*, pages 191–198.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. Bert: Pre-training of deep bidirectional transformers for language understanding. In *NAACL-HLT*.

Jon Atle Gulla, Lemei Zhang, Peng Liu, Özlem Özgöbek, and Xiaomeng Su. 2017. The adressa dataset for news recommendation. In *Proceedings of the international conference on web intelligence*, pages 1042–1048.

Huifeng Guo, Ruiming Tang, Yunming Ye, Zhenguo Li, et al. 2018. Deepfm: An end-to-end wide & deep learning framework for ctr prediction. In *International Joint Conferences on Artificial Intelligence*.

Linmei Hu, Siyong Xu, Chen Li, Cheng Yang, Chuan Shi, Nan Duan, Xing Xie, and Ming Zhou. 2020. Graph neural news recommendation with unsupervised preference disentanglement. In *Proceedings of the 58th annual meeting of the association for computational linguistics*, pages 4255–4264.

Tongwen Huang, Zhiqi Zhang, and Junlin Zhang. 2019. Fibinet: combining feature importance and bilinear feature interaction for click-through rate prediction. In *Proceedings of the 13th ACM Conference on Recommender Systems*, pages 169–177.

Jeff Johnson, Matthijs Douze, and Hervé Jégou. 2019. Billion-scale similarity search with GPUs. *IEEE Transactions on Big Data*, 7(3):535–547.

Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. 2012. Imagenet classification with deep convolutional neural networks. *Advances in neural information processing systems*, 25.

Jian Li, Jieming Zhu, Qiwei Bi, Guohao Cai, Lifeng Shang, Zhenhua Dong, Xin Jiang, and Qun Liu. 2022. Miner: Multi-interest matching network for news recommendation. In *Findings of the Association for Computational Linguistics: ACL 2022*, pages 343–352.

Jianxun Lian, Xiaohuan Zhou, Fuzheng Zhang, Zhongxia Chen, Xing Xie, and Guangzhong Sun. 2018. xdeepfm: Combining explicit and implicit feature interactions for recommender systems. In *Proceedings of the 24th ACM SIGKDD international conference on knowledge discovery & data mining*, pages 1754–1763.

Greg Linden, Brent Smith, and Jeremy York. 2003. Amazon. com recommendations: Item-to-item collaborative filtering. *IEEE Internet computing*, 7(1):76–80.

Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013. Efficient estimation of word representations in vector space. In *The International Conference on Learning Representations*.

Jeffrey Pennington, Richard Socher, and Christopher D Manning. 2014. Glove: Global vectors for word representation. In *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*, pages 1532–1543.

Matthew E. Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer. 2018. Deep contextualized word representations. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 2227–2237, New Orleans, Louisiana. Association for Computational Linguistics.

Yanru Qu, Han Cai, Kan Ren, Weinan Zhang, Yong Yu, Ying Wen, and Jun Wang. 2016. Product-based neural networks for user response prediction. In *2016 IEEE 16th International Conference on Data Mining (ICDM)*, pages 1149–1154. IEEE.

Suvash Sedhain, Scott Sanner, Darius Braziunas, Lexing Xie, and Jordan Christensen. 2014. Social collaborative filtering for cold-start recommendations. In *Proceedings of the 8th ACM Conference on Recommender systems*, pages 345–348.

Weiping Song, Chence Shi, Zhiping Xiao, Zhijian Duan, Yewen Xu, Ming Zhang, and Jian Tang. 2019. Autoint: Automatic feature interaction learning via self-attentive neural networks. In *Proceedings of the 28th ACM International Conference on Information and Knowledge Management*, pages 1161–1170.

Fei Sun, Jun Liu, Jian Wu, Changhua Pei, Xiao Lin, Wenwu Ou, and Peng Jiang. 2019. Bert4rec: Sequential recommendation with bidirectional encoder representations from transformer. In *Proceedings of the 28th ACM international conference on information and knowledge management*, pages 1441–1450.

Laurens Van der Maaten and Geoffrey Hinton. 2008. Visualizing data using t-sne. *Journal of machine learning research*, 9(11).

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. *Advances in neural information processing systems*, 30.

Heyuan Wang, Fangzhao Wu, Zheng Liu, and Xing Xie. 2020. Fine-grained interest matching for neural news recommendation. In *Proceedings of the 58th annual meeting of the association for computational linguistics*, pages 836–845.

Hongwei Wang, Fuzheng Zhang, Xing Xie, and Minyi Guo. 2018. Dkn: Deep knowledge-aware network for news recommendation. In *Proceedings of the 10th international conference on World Wide Web*.

Ruoxi Wang, Bin Fu, Gang Fu, and Mingliang Wang. 2017. Deep amp; cross network for ad click predictions. In *Proceedings of the ADKDD'17*, ADKDD'17, New York, NY, USA. Association for Computing Machinery.

Chuhan Wu, Fangzhao Wu, Mingxiao An, Jianqiang Huang, et al. 2019a. Neural news recommendation with attentive multi-view learning. In *International Joint Conferences on Artificial Intelligence*.

Chuhan Wu, Fangzhao Wu, Suyu Ge, Tao Qi, Yongfeng Huang, and Xing Xie. 2019b. Neural news recommendation with multi-head self-attention. In *Proceedings of the 2019 conference on empirical methods in natural language processing and the 9th international joint conference on natural language processing (EMNLP-IJCNLP)*, pages 6389–6394.

Chuhan Wu, Fangzhao Wu, Tao Qi, and Yongfeng Huang. 2021a. Empowering news recommendation with pre-trained language models. In *Proceedings of the 44th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 1652–1656.

Chuhan Wu, Fangzhao Wu, Yang Yu, Tao Qi, Yongfeng Huang, and Qi Liu. 2021b. NewsBERT: Distilling pre-trained language model for intelligent news application. In *Findings of the Association for Computational Linguistics: EMNLP 2021*, pages 3285–3295, Punta Cana, Dominican Republic. Association for Computational Linguistics.

Chuhan Wu, Fangzhao Wu, Yang Yu, Tao Qi, Yongfeng Huang, and Xing Xie. 2021c. Userbert: Contrastive user model pre-training. *arXiv preprint arXiv:2109.01274*.

Fangzhao Wu, Ying Qiao, Jiun-Hung Chen, Chuhan Wu, Tao Qi, Jianxun Lian, Danyang Liu, Xing Xie, Jianfeng Gao, Winnie Wu, et al. 2020. Mind: A large-scale dataset for news recommendation. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 3597–3606.

Shitao Xiao, Zheng Liu, Yingxia Shao, Tao Di, Bhuvan Middha, Fangzhao Wu, and Xing Xie. 2022. Training large-scale news recommenders with pretrained language models in the loop. In *Proceedings of the 28th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, pages 4215–4225.

Xu Xie, Fei Sun, Zhaoyang Liu, Jinyang Gao, Bolin Ding, and Bin Cui. 2020. Contrastive pre-training for sequential recommendation. *arXiv preprint arXiv:2010.14395*.

Yang Yu, Fangzhao Wu, Chuhan Wu, Jingwei Yi, Tao Qi, and Qi Liu. 2021. Tiny-newsrec: Efficient and effective plm-based news recommendation. In *ArXiv*.

Qi Zhang, Jingjie Li, Qinglin Jia, Chuyuan Wang, et al. 2021a. Unbert: User-news matching bert for news recommendation. In *International Joint Conferences on Artificial Intelligence*.

Weinan Zhang, Jiarui Qin, Wei Guo, Ruiming Tang, and Xiuqiang He. 2021b. Deep learning for click-through rate estimation. In *International Joint Conferences on Artificial Intelligence*.

Kun Zhou, Hui Wang, Wayne Xin Zhao, Yutao Zhu, Sirui Wang, Fuzheng Zhang, Zhongyuan Wang, and Ji-Rong Wen. 2020. S3-rec: Self-supervised learning for sequential recommendation with mutual information maximization. In *Proceedings of the 29th ACM International Conference on Information & Knowledge Management*, pages 1893–1902.

Jieming Zhu, Jinyang Liu, Shuai Yang, Qi Zhang, and Xiuqiang He. 2021. Open benchmarking for click-through rate prediction. In *Proceedings of the 30th ACM International Conference on Information & Knowledge Management*, pages 2759–2769.

Improving Fake News Detection of Influential Domain via Domain- and Instance-Level Transfer

Qiong Nan[1,2], Danding Wang[1], Yongchun Zhu[1,2], Qiang Sheng[1,2]
Yuhui Shi[1,2], Juan Cao[1,2]*, Jintao Li[1]
[1]Key Laboratory of Intelligent Information Processing,
Institute of Computing Technology, Chinese Academy of Sciences
[2]University of Chinese Academy of Sciences
{nanqiong19z,wangdanding,zhuyongchun18s}@ict.ac.cn
{shengqiang18z,caojuan,jtli}@ict.ac.cn
shiyuhui221@mails.ucas.ac.cn

Abstract

Both real and fake news in various domains, such as politics, health, and entertainment are spread via online social media every day, necessitating fake news detection for multiple domains. Among them, fake news in specific domains like politics and health has more serious potential negative impacts on the real world (e.g., the infodemic led by COVID-19 misinformation). Previous studies focus on multi-domain fake news detection, by equally mining and modeling the correlation between domains. However, these multi-domain methods suffer from a seesaw problem: the performance of some domains is often improved at the cost of hurting the performance of other domains, which could lead to an unsatisfying performance in specific domains. To address this issue, we propose a Domain- and Instance-level Transfer Framework for Fake News Detection (DITFEND), which could improve the performance of specific target domains. To transfer coarse-grained domain-level knowledge, we train a general model with data of all domains from the meta-learning perspective. To transfer fine-grained instance-level knowledge and adapt the general model to a target domain, we train a language model on the target domain to evaluate the transferability of each data instance in source domains and re-weigh each instance's contribution. Offline experiments on two datasets demonstrate the effectiveness of DITFEND. Online experiments show that DITFEND brings additional improvements over the base models in a real-world scenario.

1 Introduction

With the rapid popularization of the Internet, more and more people tend to acquire news through social media platforms, such as Weibo[1] and Twitter[2]. Due to the above phenomenon, fake news has spread widely all over the world. During the 2016 U.S. presidential election campaign, the top 20 frequently discussed fake election stories generated 8,711,000 engagements on Facebook, which is larger than the total of 7,367,000 for the top 20 most-discussed election stories posted by 19 major news websites (Silverman, 2016). The wide spread of fake news may break the authenticity balance of the news ecosystem (Shu et al., 2017), and it not only misleads many people but also leads to the social mobs and social panic (Chen, 2020). Therefore, fake news detection is of significant importance.

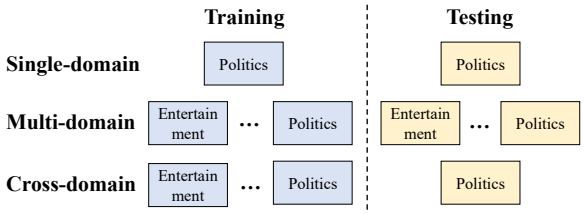

Figure 1: Illustration of the differences among single-domain, multi-domain, and cross-domain (**ours**) fake news detection.

Real-world news platforms categorize news pieces by topic into various domains, e.g., politics, health, and entertainment. Although it is necessary to detect fake news in every domain, the societal effects of fake news from some specific domains are more serious. For example, during the U.S. election, fake news in the political domain may have dictated election results (Allcott and Gentzkow, 2017), and Vosoughi et al. (2018) find that false political news travels deeper more quickly and more broadly, reaches more people, and is more vital than other categories of false information. In the COVID-19 infodemic (Bursztyn et al., 2020), thousands of fake news pieces have caused social panic (Chen, 2020) and weakened the effect of pandemic countermeasures (Bursztyn et al., 2020). Therefore, it is important to detect fake news for these specific news domains, e.g., politics (Wang, 2017; Jin et al., 2017), health (Bang

*Corresponding author
[1]https://weibo.com
[2]https://twitter.com

Proceedings of the 29th International Conference on Computational Linguistics, pages 2834–2848
October 12–17, 2022.

et al., 2021; Shi et al., 2020). Some researchers pay attention to fake news detection in the influential news domains by modeling each domain separately (Wang, 2017; Allcott and Gentzkow, 2017; Bovet and Makse, 2019; Dai et al., 2020; Cui et al., 2020; Zhou et al., 2020; Shang et al., 2022). However, a single-domain dataset could only contain limited data, e.g., the PolitiFact dataset of the political domain only has 948 samples (Shu et al., 2020). Fortunately, news pieces in different domains are correlated (Nan et al., 2021; Silva et al., 2021), so it is promising to exploit data of other domains to improve performance for a target domain.

Recently, some works (Wang et al., 2018; Silva et al., 2021; Nan et al., 2021) simultaneously model various domains to improve the overall performance of all domains. However, these multi-domain methods suffer from a serious seesaw phenomenon which could cause the performance of some domains to be improved at the cost of hurting the performance of other domains (Tang et al., 2020). For example, EANN (Wang et al., 2018) performs quite well in *Military* (with f1-score of 0.9274), whereas the results is not satisfactory in *Politics* (with f1-score of 0.8705) (Nan et al., 2021). Moreover, it is hard to guide these multi-domain models to improve the performance of a specific target domain due to the lack of a target-oriented design. In this paper, we focus on exploiting news pieces of other domains to improve the detection performance of a certain target domain, called **cross-domain fake news detection**, which can bring additional gains for target domains compared to multi-domain methods. The differences among single-domain, multi-domain, and cross-domain fake news detection are shown in Figure 1. To solve cross-domain fake news detection, we adopt two key ideas:

Transfer domain-level knowledge. It is necessary to transfer knowledge from multiple domains because news pieces in different domains are correlated. Moreover, to alleviate the seesaw phenomenon, improving the generalization ability of a multi-domain model, which can adapt fast to the target domain is necessary.

Transfer instance-level knowledge. The transferability varies from instance to instance. For example, "*A politician claimed COVID-19 is less lethal than flu*" and "*New York Officials welcome immigrants, legal or illegal*" are two different news pieces of politics. The former is more relevant to health while the latter is irrelevant. In other words, the former is more transferable. Therefore, it is important to quantify the transferability of source instances, in order to decrease the impact of irrelevant instances.

Along this line, we propose a Domain- and Instance-level Transfer framework for Fake News Detection (DITFEND). To transfer domain-level knowledge, we exploit data from multiple domains to train a general model from the meta-learning perspective, which contains common knowledge and can adapt fast to a specific domain. To transfer instance-level knowledge, we first learn a domain-adaptive language model which is endowed with characteristics of the target domain. To weigh the contribution to the target domain of every instance in source domains, we adopt an index, **perplexity** of the domain-adaptive language model, to quantify the transferability of these instances. Because in information theory perplexity can measure how well a probability model predicts an instance, in other words, low perplexity on a news piece indicates that the instance is highly related to the knowledge contained in the language model. Finally, we adapt the general model with instances from the target domain and weighted instances from source domains, in order to achieve satisfying performance on the target domain.

The main contributions of this paper can be summarized as follows:

- We investigate the importance of cross-domain fake news detection with multiple sources for target domains for the first time.
- We propose a Domain- and Instance-level Transfer Framework to improve fake news detection of target domains.
- We evaluate our proposed DITFEND on both English and Chinese real-word fake news datasets, and experiments demonstrate the effectiveness of DITFEND.

2 Related Work

2.1 Fake News Detection

Fake news detection aims at automatically classifying a news piece as real or fake. Existing methods can be generally grouped into two clusters: social-context-based methods and content-based methods (Shu et al., 2017). For social-context-based methods, some analyze propagation patterns to mine structural signals for fake news de-

tection (Jin et al., 2014; Liu and Wu, 2018; Shu et al., 2019; Mosallanezhad et al., 2022; Naumzik and Feuerriegel, 2022), others use the wisdom of crowds, such as emotion and stance, to detect fake news (Zhang et al., 2021; Jin et al., 2016), and Sheng et al. (2022) captures the environmental signals to detect fake news posts. For content-based methods, some extract evidence from external sources (Vlachos and Riedel, 2014; Shi and Weninger, 2016; Chen et al., 2022; Xu et al., 2022; Sheng et al., 2021), while others only analyze news itself and focus on better constructing features (Przybyla, 2020; Wang et al., 2018), which is within the scope of our research.

Since real-world news platforms categorize news pieces by topic into various domains, some researchers pay attention to the fake news detection performance of each domain, especially for that with serious societal effects. Some only take one specific domain into consideration and perform single-domain fake news detection (Wang, 2017; Allcott and Gentzkow, 2017; Bovet and Makse, 2019; Dai et al., 2020; Cui et al., 2020; Zhou et al., 2020; Shang et al., 2022), however, they ignore useful information from other domains. Some methods simultaneously model various domains to improve the overall performance of all domains (multi-domain fake news detection) (Wang et al., 2018; Silva et al., 2021; Nan et al., 2021; Zhu et al., 2022), however, due to the seesaw phenomenon, the performance on some target domains suffer from degeneration. Huang et al. (2021) and Mosallanezhad et al. (2022) propose to use a domain adaptation strategy for cross-domain fake news detection, however, they only transfer knowledge from one single source domain to ensure the model's detection performance on the target domain. However, in the practical scenario, news pieces from multiple source domains are inherently correlated with the target domain. It is straightforward to combine all source domains into one single domain and perform cross-domain fake news detection as (Huang et al., 2021) and (Mosallanezhad et al., 2022) do, but the improvement may not be significant. Hence, it is necessary to find a better way to make full use of all source domains to improve the performance of a target domain.

2.2 Transfer Learning

Transfer learning aims to leverage knowledge from a source domain to improve the learning performance or minimize the number of labeled examples required in a target domain (Pan and Yang, 2009; Zhuang et al., 2020). Recently, transfer learning has been widely adopted for natural language processing (Devlin et al., 2019; Liu et al., 2019; Lewis et al., 2020), e.g., sentiment classification (Peng et al., 2018), neural machine translation (Kim et al., 2019), style transfer (Yang et al., 2018). Meanwhile, meta learning serves as a paradigm that can be used to improve transfer learning problems (Hospedales et al., 2021). Benefit from its ability to integrate prior experience as well as generality to all domains, meta learning has been widely adopted in many applications (Li et al., 2018; Wang et al., 2021). In this paper, we propose a novel transfer framework based on meta-learning for cross-domain fake news detection.

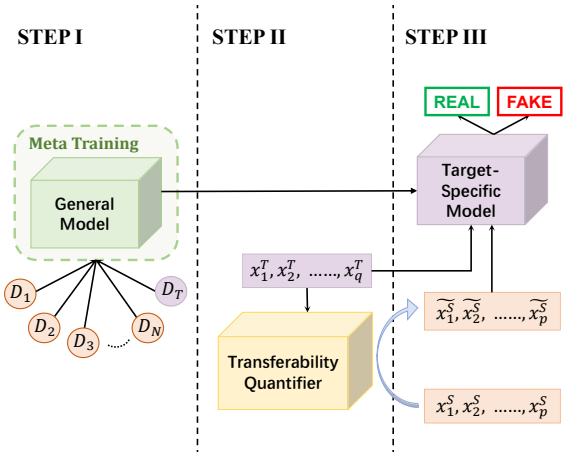

Figure 2: The overall framework of our DITFEND model. Step I is the general model training procedure, step II is the transferability quantifying procedure, and step III is the target domain adaptation procedure. The three steps correspond to Section 3.2, 3.3 and 3.4 respectively.

3 DITFEND: Domain- and Instance-Level Transfer for Fake News Detection

3.1 Problem Statement

In the scenario of cross-domain fake news detection, there is one target domain $\mathcal{D}_T$ and a collection of N source domains $\mathcal{D}_S = \{\mathcal{D}_1, \mathcal{D}_2, ..., \mathcal{D}_N\}$. For each domain, we have a collection of news pieces. For a piece of news P, we pad it with additional characters $[CLS]$ and $[SEP]$. Then we tokenize the sentence into m tokens $\{t_1, t_2, ..., t_m\}$, and encode it to get the corresponding embedding vector $\boldsymbol{e}$. To distinguish between different domains, we denote $\boldsymbol{e_T}$ as the embedding vector of the target domain

Algorithm 1 General Model Training Procedure.

Input: Given N source domains and one target domain.
Input: The base model $\mathcal{M}(\theta)$.
Input: Learning rate α and β.

randomly initialize θ
while not coverage **do**:
  sample batch of training tasks $\{\mathcal{T}_1, ..., \mathcal{T}_n\}$
  for all $\mathcal{T}_d \in \{\mathcal{T}_1, ..., \mathcal{T}_n\}$ **do**:
    $\mathcal{T}_d$ contains two disjoint sets $\mathcal{D}_d^s$ and $\mathcal{D}_d^q$
    evaluate loss $\mathcal{L}_d^s(\theta)$ with $\mathcal{D}_d^s$
    compute updated parameter $\theta_d = \theta - \alpha \frac{\partial \mathcal{L}_d^s(\theta)}{\partial \theta}$
    evaluate loss $\mathcal{L}_d^q(\theta_d)$ with $\mathcal{D}_d^q$
  end
  update $\theta = \theta - \beta \sum_{\mathcal{T}_d \in \{\mathcal{T}_1, ..., \mathcal{T}_n\}} \frac{\partial \mathcal{L}_d^q(\theta_d)}{\partial \theta}$
end

and $\boldsymbol{e_S} \in \{\boldsymbol{e_1}, \boldsymbol{e_2}, ..., \boldsymbol{e_N}\}$ of source domains.

The core idea of our framework is to leverage news pieces from all domains to train a target-adaptive fake news detector. To achieve this, in the first stage, we train a general model with news pieces from all domains, such that the model can alleviate the seesaw phenomenon (i.e., domain-level transfer); In the second stage, we evaluate and quantify the transferability of instances from source domains; And in the third stage, we adapt the general model to the target domain. The overall framework is shown in Figure 2.

3.2 General Model Training

For domain-level transfer, we take the advantage of meta-learning and train a general model $\mathcal{M}(\theta)$ to aggregate the knowledge from all the domains.

In each iteration of the parameter update, we draw a batch of training tasks $\{\mathcal{T}_1, ..., \mathcal{T}_n\}$, and for each task $\mathcal{T}_d \in \{\mathcal{T}_1, ..., \mathcal{T}_n\}$, we divide it into two disjoint sets: a support set $\mathcal{D}_d^s$ and a query set $\mathcal{D}_d^q$. The model is trained with samples from support set $\mathcal{D}_d^s$ and feedback with the corresponding loss $\mathcal{L}_d^s$ from $\mathcal{D}_d^s$:

$$\mathcal{L}_d^s(\theta) = \frac{1}{m_s} \sum_{i=1}^{m_s} -y_i \log \hat{y}_i - (1 - y_i) \log(1 - \hat{y}_i), \tag{1}$$

where m_s is the number of the data in the current support set. We adopt cross-entropy loss because it is widely used as the optimization goal for binary fake news detection (Wang et al., 2018; Silva et al., 2021; Nan et al., 2021). And then we use $\mathcal{L}_d^s$ to optimize the parameter of the current task $\mathcal{T}_d$ via gradient descent:

$$\theta_d = \theta - \alpha \frac{\partial \mathcal{L}_d^s(\theta)}{\partial \theta}, \tag{2}$$

where α is the learning rate of the training process within each task, and θ_d is the optimized model parameters for the current task $\mathcal{T}_d$.

And then the model is tested on samples from the current query set $\mathcal{D}_d^q$. The model is improved by considering how the test error $\mathcal{L}_d^q$ on $\mathcal{D}_d^q$ changes with respect to the parameters:

$$\mathcal{L}_d^q(\theta_d) = \frac{1}{m_q} \sum_{i=1}^{m_q} -y_i \log \hat{y}_i - (1 - y_i) \log(1 - \hat{y}_i), \tag{3}$$

where m_q is the number of data in the current query set.

In effect, the testing error on $\mathcal{D}_d^q$ serves as the training error of the meta-learning process.

After we loop over all tasks in $\{\mathcal{T}_1, ..., \mathcal{T}_n\}$, the base model's parameters θ can be updated as follows:

$$\theta = \theta - \beta \nabla_\theta \sum_{i=1}^{n} \mathcal{L}_i^q, \tag{4}$$

where β is the learning rate of the meta-learning process. The procedure of general model training is summarized in Alg 1.

3.3 Transferability Quantifying

In order to endow the language model with target domain knowledge, we perform the Masked Language Modeling task on the target domain to get a domain-adaptive language model. This second phase of training the language model can bring significant performance improvement on the following task based on the language model (Gururangan et al., 2020). And we utilize the domain-adaptive language model to evaluate the transferability of source instances.

Domain-adaptive Language Model Training. Let $\mathcal{D}_T$ be a dataset of news pieces from the target domain T, where P_T is a news piece in $\mathcal{D}_T$ containing n tokens $\{w_1, ..., w_n\}$. We replace 15% tokens in the input sequence with the $[MASK]$ token, a random token, or the original token. The model is required to predict the masked tokens based on the other tokens in P_T. The training objective is to minimize the cross-entropy loss of the language model in predicting the masked tokens of the target data:

$$\min_{\theta_m} \sum_{x \in D_t} \mathcal{L}_m(x, \theta_m), \tag{5}$$

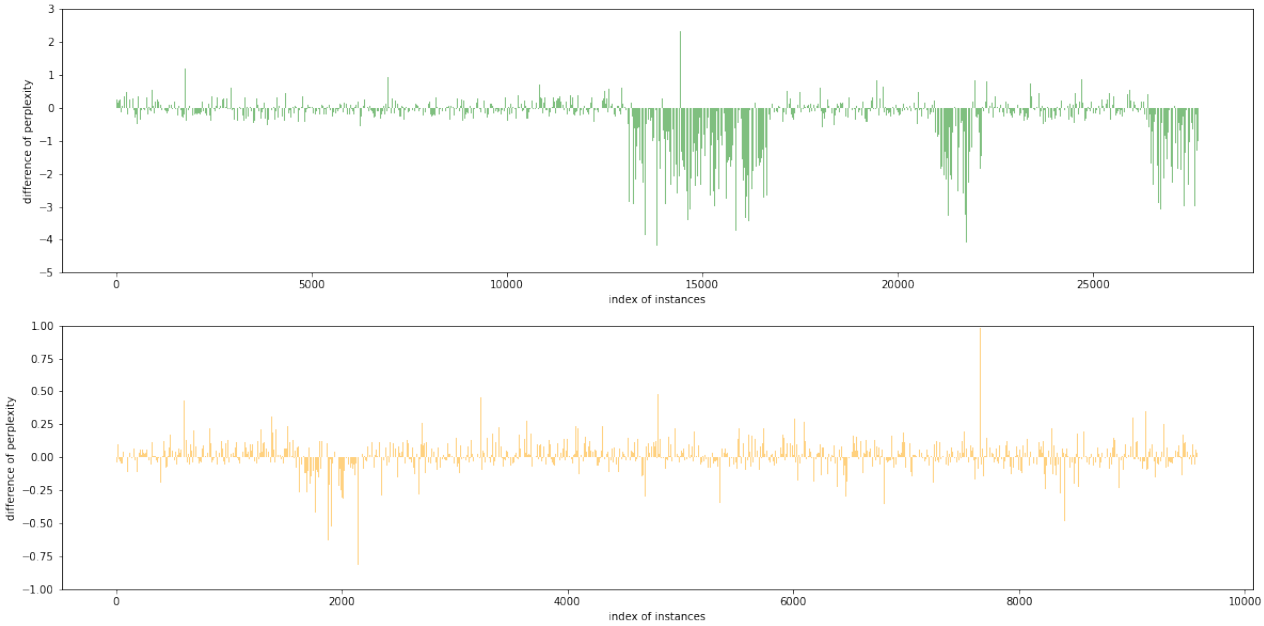

Figure 3: The D-value of different target language models' perplexity on the same batch of source instances (Top: Chinese dataset; Bottom: English dataset).

where θ_m is the parameters of the language model (*bert-base-uncased* for English and *bert-base-chinese* for Chinese).

After performing the Masked Language Modeling task on the target domain, we can effectively endow the corresponding language model with knowledge of the target domain.

Perplexity is an indicator that can measure how well a language model predicts a sentence, i.e., the higher the probability of prediction is, the better the language model is, and the lower the language model's perplexity is. Many researchers use it to evaluate language models (Belinkov and Glass, 2019; Yogatama et al., 2018). We exploit perplexity from another perspective – the better a language model can predict a sentence, the better this sentence fits the knowledge endowed in the language model. Therefore, it is intuitive to compute the perplexity of the language model on a given sample to quantify the sample's transferability, i.e., the lower perplexity indicates stronger transferability.

Let $\mathcal{D}_S$ indicate one of the source datasets, where P_S is a news piece in $\mathcal{D}_S$ containing m tokens $\{w_0, w_2, ..., w_{m-1}\}$. We pad it with two tokens, i.e., $[CLS]$ and $[SEP]$. A masked sentence P_{mask} is generated by masking a word/character in the sentence as follows:

$$\mathcal{P}_{mask} =< [CLS], ..., w_{i-1}, [MASK], w_{i+1}, ..., [SEP] >, \tag{6}$$

where $0 \leq i \leq m-1$. Then the target-adaptive language model is utilized to predict the probability of the right words w_i in the $[MASK]$ position:

$$prob(w_i) = MLM(\mathcal{P}, w_i). \tag{7}$$

After we calculated the probability of all the words in the sentence, the perplexity of the language model on the whole sentence is calculated as follows:

$$pp = \sqrt[N]{\prod_{i=0}^{N-1} \frac{1}{prob(w_i)}}. \tag{8}$$

To show the difference in the target-adaptive language model when assigning different target domains, we compute the difference value of different target-adaptive language models' perplexity on source instances. And we visualize the distribution in Figure 3 in detail. From the distribution, we can see that there exists an obvious difference in both the Chinese and English datasets.

Finally, we quantify the transferability of instances from source domains as follows:

$$w = 1/pp, \tag{9}$$

where w is the indicator of the transferability of the sample. In this way, we quantify the transferability of source instances, i.e., assigning bigger weights to samples with lower perplexity.

3.4 Target Domain Adaptation

In this section, we aim to adapt the general fake news detection model to the target domain. In Section 3.3, we have assigned each sample in source domains with a weight value to indicate its transferability. To make full use of source domain samples, we re-weigh them based on their transferability, along with the target domain samples to train the general model.

We exploit the general model trained via meta-learning (Section 3.2), and optimize it by minimizing the cross-entropy loss as follows:

$$\begin{aligned}\mathcal{L}^{ce}(y,\hat{y}) &= -ylog\hat{y} - (1-y)log(1-\hat{y}) \\ \mathcal{L} &= \mathbb{E}_{(x,y)\sim p_s(x,y)} w(x)\mathcal{L}^{ce}(y,\hat{y}) \\ &+ \mathbb{E}_{(x,y)\sim p_t(x,y)} \mathcal{L}^{ce}(y,\hat{y}),\end{aligned} \tag{10}$$

where y is the ground truth, $\hat{y}$ is the predicted label, and $w(x)$ is the indicator of the transferability of the instance x obtained in Section 3.3.

4 Experiments

In this section, we aim to answer the following evaluation questions:

- **EQ1:** Can DITFEND improve the performance of a target domain when coordinates with different base models?
- **EQ2:** How effective are domain-level transfer and instance-level transfer?
- **EQ3:** Can DITFEND perform well in a real-world fake news detection scenario?

4.1 Datasets

We evaluate DITFEND on both Chinese and English datasets. And the statistics of datasets are listed in Tables 1 and 2.

Table 1: Statistics of the Chinese dataset

Domain	Science	Military	Edu.	Disaster	Politics
# Fake	93	222	248	591	546
# Real	143	121	243	185	306
Total	236	343	491	776	852
Domain	**Health**	**Finance**	**Ent.**	**Society**	**All**
# Fake	515	362	440	1,471	4,488
# Real	485	959	1,000	1,198	4,640
Total	1,000	1,321	1,440	2,669	9,128

English Dataset. We combine two datasets (PolitiFact and GossipCop) in FakeNewsNet (Shu et al., 2020) and COVID (Li et al., 2020) into an English dataset, which contains three domains, namely PolitiFact, GossipCop and COVID.

Table 2: Statistics of the English Dataset

Domain	PolitiFact	GossipCop	COVID	All
# Fake	420	4,947	1,317	7,483
# Real	528	16,694	4,750	22,864
Total	948	21,641	6,067	30,347

Chinese Dataset (Nan et al., 2021). multi-domain fake news detection dataset collected from Sina Weibo. There are 9 domains in total, which is Science, Military, Education, Disaster, Politics, Health, Finance, Entertainment, and Society. In our experiments, in order to evaluate the impact of the domain number, we sample 3 domains (Politics, Health, Entertainment) and 6 domains (Education, Disaster, Health, Finance, Entertainment, Society) from the Chinese dataset to construct the other two multi-domain datasets, i.e., Chinese 3-domain and Chinese 6-domain.

4.2 Baseline Models

In this paper, we focus on textual content-based fake news detection. Technically, our DITFEND framework can coordinate with any text-based models that produce post-representation. Thus, we use 8 representative text-based models as our base models, which can be divided into three groups: (1) text classification models: **TextCNN** (Kim, 2014), **BiGRU** (Ma et al., 2016), and **RoBERTa** (Liu et al., 2019; Cui et al., 2019); (2) transfer learning models: **MMOE** (Ma et al., 2018) and **MOSE** (Qin et al., 2020); (3) multi-domain fake news detection models: **EANN**$_{\text{T}}$ (Wang et al., 2018), **EDDFN** (Silva et al., 2021), and **MDFEND** (Nan et al., 2021). Details about the base models are as follows:

TextCNN (Kim, 2014): Convolutional Neural Networks (CNNs) have been proven to gain remarkably strong performance on the task of text classification. In our experiment, we use filter windows of 1, 2, 3, 5, 10 with 64 feature maps each.

BiGRU (Ma et al., 2016): It is a widely used models in natural language processing applications. Different from (Ma et al., 2016), we treat each piece of news as a sequential input to a one-layer BiGRU model.

RoBERTa (Liu et al., 2019; Cui et al., 2019): It is a robustly optimized BERT (Devlin et al., 2019) pre-trained model. We utilize it to encode tokens of news content and feed the extracted embedding

Table 3: Performance comparison of base models with and without DITFEND on English dataset and Chinese 3-domain dataset.

Method	English dataset						Ch-3 dataset					
	target: COVID			target: Politifact			target: Health			target: Politics		
	F1	AUC	Acc	F1	AUC	Acc	F1	AUC	Acc	F1	AUC	Acc
BiGRU	0.7448	0.9114	0.8606	0.7339	0.8213	0.7375	0.8577	0.9367	0.8580	0.8384	0.9032	0.8637
+*DITFEND*	**0.9219**	**0.9890**	**0.9501**	**0.8476**	**0.9043**	**0.8477**	**0.8792**	**0.9548**	**0.8795**	**0.8528**	**0.9261**	**0.8772**
TextCNN	0.8322	0.9397	0.8955	0.7040	0.8046	0.7064	0.8716	0.9552	0.8720	**0.8579**	0.9067	**0.8859**
+*DITFEND*	**0.8642**	**0.9706**	**0.9184**	**0.7913**	**0.8740**	**0.7913**	**0.8878**	**0.9582**	**0.8880**	0.8563	**0.9127**	0.8807
RoBERTa	0.9014	0.9770	0.9377	0.7967	0.9078	0.7989	0.8955	0.9641	0.8955	0.8300	0.8948	0.8628
+*DITFEND*	**0.9360**	**0.9901**	**0.9578**	**0.8608**	**0.9183**	**0.8609**	**0.9105**	**0.9708**	**0.9105**	**0.8445**	**0.9085**	**0.8725**
EANNT	0.8836	0.9751	0.9282	0.7558	0.8612	0.7584	0.9189	**0.9787**	0.9190	0.8405	0.9074	0.8690
+*DITFEND*	**0.8883**	**0.9825**	**0.9310**	**0.8040**	**0.9074**	**0.8046**	**0.9219**	0.9760	**0.9220**	**0.8574**	**0.9136**	**0.8854**
MMOE	**0.9379**	0.9883	0.9588	0.8477	0.9408	0.8486	**0.9215**	0.9639	**0.9215**	**0.8779**	0.9388	**0.8982**
+*DITFEND*	0.9361	**0.9911**	**0.9600**	**0.8613**	**0.9515**	**0.8615**	0.9034	**0.9657**	0.9035	0.8523	**0.9398**	0.8766
MOSE	0.9326	0.9879	0.9588	0.8576	0.9447	0.8590	0.9023	0.9683	0.9025	0.8564	0.9138	0.8795
+*DITFEND*	**0.9586**	**0.9880**	**0.9712**	**0.8732**	**0.9553**	**0.8732**	**0.9069**	**0.9711**	**0.9070**	**0.8642**	**0.9220**	**0.8865**
EDDFN	0.9306	0.9891	0.9547	0.8505	0.9432	0.8509	0.9235	**0.9735**	0.9235	0.8440	0.9207	0.8702
+*DITFEND*	**0.9401**	**0.9912**	**0.9600**	**0.8720**	**0.9466**	**0.8725**	**0.9245**	0.9731	**0.9245**	**0.8486**	**0.9251**	**0.8731**
MDFEND	0.9331	0.9874	0.9565	0.8473	0.9391	0.8485	0.9419	0.9855	0.9420	0.8555	0.9259	0.8854
+*DITFEND*	**0.9485**	**0.9934**	**0.9700**	**0.8589**	**0.9500**	**0.8593**	**0.9530**	**0.9856**	**0.9530**	**0.8663**	**0.9368**	**0.8895**

into an MLP to obtain the final prediction. *roberta-base-chinese* and *roberta-base-uncased* (Liu et al., 2019; Cui et al., 2019) are exploited for Chinese and English datasets, respectively.

EANNT (Wang et al., 2018): It is a model that aims to learn event-agnostic features, which uses a TextCNN module for text representation and adopts an auxiliary event discriminator for adversarial learning. In our experiments, we only use the textual branch, and rather than event-level, we perform domain-level adversarial training (i.e., use the discriminator to classify different domains) to learn domain-shared features.

MMOE (Ma et al., 2018) and **MOSE** (Qin et al., 2020): These two models are proposed for multi-task learning. In our experiments, we assume that fake news in different domains are different tasks, and use the two models for multi-domain modeling.

EDDFN (Silva et al., 2021): It is proposed for multi-domain fake news detection, which models different domains implicitly and jointly preserves domain-specific and cross-domain knowledge.

MDFEND (Nan et al., 2021): It is a multi-domain fake news detection model, which utilizes a domain gate to aggregate multiple representations extracted by mixture-of-experts for multi-domain fake news detection.

4.3 Experiment Settings

Training Procedure. I. We use samples from all domains to train a general model from the meta-learning perspective. **II.** We use samples from the target domain to train a domain-adaptive language model via *Masked Language Modeling* task. **III.** We calculate the perplexity of the domain-adaptive language model on samples from source domains, which is used to re-weigh the samples. **IV.** We use samples from the target domain and source domains (assigned with the corresponding weights) to train the general model to adapt to the target domain.

Target domain assignment. In practice, fake news pieces of Politics, Health, and Finance have more serious influence. For Ch-3 and Ch-9 dataset, we choose *Politics* and *Health* as the target domain, respectively. For Ch-6 dataset, we choose *Health* and *Finance* as the target domain,respectively. For English dataset, we choose *PolitiFact* and *COVID* as the target domain, respectively.

Evaluation Metrics. We treat the fake news detection problem as a binary classification task. We report macro F1 score (F1), accuracy (Acc), and Area Under ROC (AUC). For the online tests, where the number of fake news is much lower than real news, we should detect fake news as accurately as possible without misclassifying real news. Thus, we further report standardized partial AUC ($\text{SPAUC}_{FPR \leq 0.1}$).

Implementation Details. We limit the max length of the sentence to 170 tokens for Chinese, and 300 tokens for English. Following the settings in (Nan et al., 2021), we tokenized sentences with jieba[3] (for Chinese), nltk (Bird, 2006) (for English), and embed them with Word2Vec (Le and

[3]https://github.com/fxsjy/ jieba

Mikolov, 2014; Mikolov et al., 2013) for BiGRU and TextCNN. We use RoBERTa and the corresponding tokenizer for other models. For the two embedding types, we fix the dimension of embeddings to 768 for RoBERTa (Devlin et al., 2019) and 200 for Word2Vec (Le and Mikolov, 2014; Mikolov et al., 2013). For a fair comparison, we set the same hyperparameters for all base models. The MLP (Multi-Layer Perceptron) module used in these models contains one dense layer (384 hidden units).

4.4 Performance Comparison (EQ1)

To answer **EQ1**, the DITFEND framework coordinates with each base model and compares with it respectively on both the Chinese dataset and English dataset. Experiment results are shown in Table 3 and Table 7 (Appendix A.1). The experimental results further reveal several insightful observations:

- DITFEND can coordinate with various base models. From the results compared with these base models, DITFEND can mostly bring improvements of the fake news detection performance on target domains, which indicates DITFEND has satisfying compatibility.
- The improvement on the English dataset is bigger than the Chinese dataset. The main reason could be that the English dataset contains more samples than Chinese dataset, which can bring benefits for knowledge transfer. During target domain adaptation, the model can learn more transferable information about the target domain.
- We find that for the same target domains (Health and Politics) in Ch-3 dataset and Ch-9 dataset, the performance of the target domain in Ch-9 dataset exceeds the former with most base models, while on the other hand, for the same target domain (Health) in Ch-6 and Ch-9, the performances are too close to call, which indicates that more domains could bring more transferable knowledge to some extent, but the number of domain is not proportionate with performance.

4.5 Analysis (EQ2)

To answer **EQ2**, we conduct both an ablation study and a case study. In the ablation study, we evaluate the effectiveness of domain-level transfer and instance-level transfer respectively; In case study, we aim to intuitively illustrate that our transferability quantifying strategy can pick out related source news pieces according to one specific domain.

Table 4: Ablation study on Chinese 9-domain dataset with RoBERTa and EDDFN.

Methods	target: Health			target: Politics		
	F1	AUC	Acc	F1	AUC	Acc
RoBERTa	0.9090	0.9611	0.9090	0.8366	0.9034	0.8637
+DITFEND	**0.9115**	**0.9739**	**0.9115**	**0.8775**	**0.9199**	**0.8982**
w/o meta	0.9096	0.9654	0.9087	0.8557	0.9144	0.8795
w/o sources	0.9100	0.9712	0.9100	0.8687	0.9183	0.8928
EDDFN	0.9379	0.9807	0.9380	0.8478	0.9292	0.8754
+DITFEND	**0.9399**	**0.9821**	**0.9400**	**0.8507**	**0.9308**	0.8772
w/o meta	0.9380	0.9810	0.9380	0.8480	0.9295	0.8760
w/o sources	0.9385	0.9815	0.9383	0.8482	0.9302	**0.8798**

4.5.1 Ablation Study

We evaluate two ablation experiment groups based on two representative fake news detection base models to evaluate the effectiveness of different modules in DITFEND framework. Table 4 shows the experimental results. To evaluate the effectiveness of domain-level transfer, we replace the meta-training procedure with classical training and use the weighted source data and target data to train the general model to adapt to the target domain. From the results, we can see that w/o *meta-training* performs worse than the whole DITFEND framework but better than the original model. To evaluate the effectiveness of instance-level transfer, we abandon instances from source domains when performing domain adaptation. The ablation results show that without instances from source domains, performance on target domains drops a little bit. Comparing the degree of decline of the two ablation experiments, we can also conclude that meta-training procedure brings more boost compared to instance from source domains.

4.5.2 Case Study

To further verify whether our transferability quantifier can pick out useful samples from source domains effectively, we conduct a case study on both the Chinese dataset and the English dataset. For each dataset, we have two assigned target domains (i.e., *Health* and *Politics* for Chinese, *COVID* and *PolitiFact* for English). According to Section 3.2, each instance has two different transferability indicators depending on two different target domains. Therefore, we use the corresponding indicator to choose samples relevant to the target domain, which can illustrate the interpretability of

instance-level transfer to some extent. Representative samples are listed in Table 5. Taking *Politics* domain (as the target domain) in the Chinese dataset for example, we successfully pick out some instances from other domains, which contain some knowledge related to *Politics*.

Table 5: Some representative examples selected depending on Transferability Quantifier. Words or phrases underlined are some indicators related to the target domain.

Target: Health
I. The team will undergo nucleic acid tests in the coming days as they resume individual training. [entertainment] II. No one was injured when a test carriage derailed. [disaster] III. Bask in inflatable dolls caused police cars and ambulances all out in broad daylight. [society]
Target: Finance
I. It has become the first Tencent smart medical joint innovation base in China. [health] II. The central bank claimed to accelerate finance-technology regulatory framework. [science] III. There are barriers to electronic use for older people in digital lifestyle. [society]
Target: Politics
I. Trump adviser urges Taiwan to come forward with advanced weapons program. [military] II. Shandong provincial government and Tencent signed a strategic cooperation agreement. [Finance] III. Beijing municipal government announces the hukou score to the society every year. [society]

4.6 Online Tests (EQ3)

To verify the real benefits of DITFEND bringing to the online system, we conduct online testing experiments. Different from the offline datasets, this online dataset is much more skewed (30,977 real: 774 fake $\approx$ 40:1). The testing set contains all news pieces in a whole month (ranging from 2021/10/10 to 2021/11/10), and the training set contains news pieces published before 2021/10/10. Both the training set and the test set come from the same online system[4]. The DITFEND coordinates with two online baselines: TextCNN and EANN, respectively. In real-world scenarios, the number of fake news items is much smaller than real news items, which means that we should detect as many fake news items as possible without misclassifying real news items. In other words, the objective is to improve the True Positive Rate (TPR) on the basis of a low False Positive Rate (FPR). Thus, beyond AUC and F1, we follow (McClish, 1989; Zhu et al., 2020) and adopt standardized partial AUC ($\text{SPAUC}_{\text{FPR}\leq 0.1}$). The online results in Table 6 demonstrate that DITFEND achieves a significant improvement on AUC and SPAUC against the baselines, with greater improvements in SPAUC.

Table 6: Online test performance. The results below are a relative improvement over the two baseline models.

Improvement on	SPAUC	AUC
TextCNN	+2.07%	+1.40%
EANN	+3.40%	+2.90%

5 Conclusion and Future Work

Conclusion. In this paper, we propose DITFEND, a Domain- and Instance-level Transfer framework to improve fake news detection in an influential domain. For domain-level transfer, we adopt meta-learning to learn common knowledge across domains, in which way we can learn a general model; For instance-level transfer, we transform the problem of transferability evaluation to the task of character/word prediction by a language model endowed with target knowledge. Experiments on the Chinese dataset and English dataset demonstrate the effectiveness of our DITFEND framework over several fake news detection models. Ablation studies and case studies further evaluate the effectiveness and interpretability of each module in DITFEND. Online testing results verified the practical performance of our DITFEND framework.

Future Work. (1) In this work, the domain label of the target domain is unknown in advance. We will investigate how to handle the situation when the domain label of the target domain and/or source domains is unknown; (2) Temporal distribution shift is more challenging, and we plan to investigate how to adapt fake news detection models to the data in the future; (3) We will explore how to select source domains based on the analysis of transferability between different domains.

Acknowledgements

The research work is supported by the National Key Research and Development Program of China (2021AAA0140203), the National Natural Science Foundation of China (62203425), the China Postdoctoral Science Foundation (2022TQ0344), and the International Postdoctoral Exchange Fellowship Program by Office of China Postdoc Council (YJ20220198).

[4]http://www.newsverify.com/os/index.html

References

Hunt Allcott and Matthew Gentzkow. 2017. Social media and fake news in the 2016 election. *Journal of economic perspectives*, 31(2):211–36.

Yejin Bang, Etsuko Ishii, Samuel Cahyawijaya, Ziwei Ji, and Pascale Fung. 2021. Model generalization on covid-19 fake news detection. In *International Workshop on Combating Online Hostile Posts in Regional Languages during Emergency Situation*, pages 128–140. Springer.

Yonatan Belinkov and James Glass. 2019. Analysis methods in neural language processing: A survey. *Transactions of the Association for Computational Linguistics*, 7:49–72.

Steven Bird. 2006. NLTK: The Natural Language Toolkit.

Alexandre Bovet and Hernán A Makse. 2019. Influence of fake news in twitter during the 2016 us presidential election. *Nature communications*, 10(1):1–14.

Leonardo Bursztyn, Aakaash Rao, Christopher P Roth, and David H Yanagizawa-Drott. 2020. Misinformation during a pandemic. Technical report, National Bureau of Economic Research.

Qingqing Chen. 2020. Coronavirus rumors trigger irrational behaviors among chinese netizens. https://www.globaltimes.cn/content/1178157.shtml. 2020.

Zhendong Chen, Siu Cheung Hui, Fuzhen Zhuang, Lejian Liao, Fei Li, Meihuizi Jia, and Jiaqi Li. 2022. Evidencenet: Evidence fusion network for fact verification. In *Proceedings of the ACM Web Conference 2022*, pages 2636–2645.

Limeng Cui, Haeseung Seo, Maryam Tabar, Fenglong Ma, Suhang Wang, and Dongwon Lee. 2020. DETERRENT: Knowledge guided graph attention network for detecting healthcare misinformation. In *Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, pages 492–502. Association for Computing Machinery.

Yiming Cui, Wanxiang Che, Ting Liu, Bing Qin, Ziqing Yang, Shijin Wang, and Guoping Hu. 2019. Pre-training with whole word masking for chinese bert. *arXiv preprint arXiv:1906.08101*.

Enyan Dai, Yiwei Sun, and Suhang Wang. 2020. Ginger cannot cure cancer: Battling fake health news with a comprehensive data repository. In *Proceedings of the International AAAI Conference on Web and Social Media*, volume 14, pages 853–862.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, Minneapolis, Minnesota. Association for Computational Linguistics.

Suchin Gururangan, Ana Marasović, Swabha Swayamdipta, Kyle Lo, Iz Beltagy, Doug Downey, and Noah A Smith. 2020. Don't stop pretraining: Adapt language models to domains and tasks. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 8342–8360.

Timothy M Hospedales, Antreas Antoniou, Paul Micaelli, and Amos J Storkey. 2021. Meta-learning in neural networks: A survey. *IEEE Transactions on Pattern Analysis and Machine Intelligence*.

Yinqiu Huang, Min Gao, Jia Wang, and Kai Shu. 2021. Dafd: Domain adaptation framework for fake news detection. In *International Conference on Neural Information Processing*, pages 305–316. Springer.

Zhiwei Jin, Juan Cao, Han Guo, Yongdong Zhang, Yu Wang, and Jiebo Luo. 2017. Detection and analysis of 2016 US presidential election related rumors on twitter. In *International conference on social computing, behavioral-cultural modeling and prediction and behavior representation in modeling and simulation*, pages 14–24. Springer.

Zhiwei Jin, Juan Cao, Yu-Gang Jiang, and Yongdong Zhang. 2014. News credibility evaluation on microblog with a hierarchical propagation model. In *2014 IEEE international conference on data mining*, pages 230–239. IEEE.

Zhiwei Jin, Juan Cao, Yongdong Zhang, and Jiebo Luo. 2016. News verification by exploiting conflicting social viewpoints in microblogs. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 30.

Yoon Kim. 2014. Convolutional neural networks for sentence classification. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1746–1751. Association for Computational Linguistics.

Yunsu Kim, Yingbo Gao, and Hermann Ney. 2019. Effective cross-lingual transfer of neural machine translation models without shared vocabularies. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 1246–1257.

Quoc Le and Tomas Mikolov. 2014. Distributed representations of sentences and documents. In *International conference on machine learning*, pages 1188–1196. PMLR.

Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Veselin Stoyanov, and Luke Zettlemoyer. 2020. Bart:

Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7871–7880.

Da Li, Yongxin Yang, Yi-Zhe Song, and Timothy M Hospedales. 2018. Learning to generalize: Meta-learning for domain generalization. In *Proceedings of the Thirty-Second AAAI Conference on Artificial Intelligence and Thirtieth Innovative Applications of Artificial Intelligence Conference and Eighth AAAI Symposium on Educational Advances in Artificial Intelligence*, pages 3490–3497.

Yichuan Li, Bohan Jiang, Kai Shu, and Huan Liu. 2020. Mm-covid: A multilingual and multimodal data repository for combating covid-19 disinformation. *arXiv preprint arXiv:2011.04088*.

Yang Liu and Yi-Fang Brook Wu. 2018. Early detection of fake news on social media through propagation path classification with recurrent and convolutional networks. In *Thirty-second AAAI Conference on Artificial Intelligence*.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Roberta: A robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692*.

Jiaqi Ma, Zhe Zhao, Xinyang Yi, Jilin Chen, Lichan Hong, and Ed H Chi. 2018. Modeling task relationships in multi-task learning with multi-gate mixture-of-experts. In *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, pages 1930–1939.

Jing Ma, Wei Gao, Prasenjit Mitra, Sejeong Kwon, Bernard J Jansen, Kam-Fai Wong, and Meeyoung Cha. 2016. Detecting rumors from microblogs with recurrent neural networks. In *Proceedings of the Twenty-Fifth International Joint Conference on Artificial Intelligence*, pages 3818–3824. AAAI Press.

Donna Katzman McClish. 1989. Analyzing a portion of the roc curve. *Medical decision making*, 9(3):190–195.

Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013. Efficient estimation of word representations in vector space. *arXiv preprint arXiv:1301.3781*.

Ahmadreza Mosallanezhad, Mansooreh Karami, Kai Shu, Michelle V Mancenido, and Huan Liu. 2022. Domain adaptive fake news detection via reinforcement learning. In *Proceedings of the ACM Web Conference 2022*, pages 3632–3640.

Qiong Nan, Juan Cao, Yongchun Zhu, Yanyan Wang, and Jintao Li. 2021. MDFEND: Multi-domain fake news detection. In *Proceedings of the 30th ACM International Conference on Information and Knowledge Management*. Association for Computing Machinery.

Christof Naumzik and Stefan Feuerriegel. 2022. Detecting false rumors from retweet dynamics on social media. In *Proceedings of the ACM Web Conference 2022*, pages 2798–2809.

Sinno Jialin Pan and Qiang Yang. 2009. A survey on transfer learning. *IEEE Transactions on knowledge and data engineering*, 22(10):1345–1359.

Minlong Peng, Qi Zhang, Yu-gang Jiang, and Xuan-Jing Huang. 2018. Cross-domain sentiment classification with target domain specific information. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 2505–2513.

Piotr Przybyla. 2020. Capturing the style of fake news. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pages 490–497. AAAI Press.

Zhen Qin, Yicheng Cheng, Zhe Zhao, Zhe Chen, Donald Metzler, and Jingzheng Qin. 2020. Multitask mixture of sequential experts for user activity streams. In *Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, pages 3083–3091.

Lanyu Shang, Ziyi Kou, Yang Zhang, and Dong Wang. 2022. A duo-generative approach to explainable multimodal covid-19 misinformation detection. In *Proceedings of the ACM Web Conference 2022*, pages 3623–3631.

Qiang Sheng, Juan Cao, Xueyao Zhang, Rundong Li, Danding Wang, and Yongchun Zhu. 2022. Zoom out and observe: News environment perception for fake news detection. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 4543–4556, Dublin, Ireland. Association for Computational Linguistics.

Qiang Sheng, Xueyao Zhang, Juan Cao, and Lei Zhong. 2021. Integrating pattern-and fact-based fake news detection via model preference learning. In *Proceedings of the 30th ACM International Conference on Information & Knowledge Management*, pages 1640–1650, Virtual Event, Queensland, Australia. Association for Computing Machinery.

Anqi Shi, Zheng Qu, Qingyao Jia, and Chen Lyu. 2020. Rumor detection of covid-19 pandemic on online social networks. In *2020 IEEE/ACM Symposium on Edge Computing (SEC)*, pages 376–381. IEEE.

Baoxu Shi and Tim Weninger. 2016. Fact checking in heterogeneous information networks. In *Proceedings of the 25th International Conference Companion on World Wide Web*, pages 101–102.

Kai Shu, Limeng Cui, Suhang Wang, Dongwon Lee, and Huan Liu. 2019. dEFEND: Explainable Fake News Detection. In *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, pages 395–405, Anchorage, AK, USA. Association for Computing Machinery.

Kai Shu, Deepak Mahudeswaran, Suhang Wang, Dongwon Lee, and Huan Liu. 2020. Fakenewsnet: A data repository with news content, social context, and spatiotemporal information for studying fake news on social media. *Big data*, 8(3):171–188.

Kai Shu, Amy Sliva, Suhang Wang, Jiliang Tang, and Huan Liu. 2017. Fake news detection on social media: A data mining perspective. *ACM SIGKDD Explorations Newsletter*, 19(1):22–36.

Amila Silva, Ling Luo, Shanika Karunasekera, and Christopher Leckie. 2021. Embracing domain differences in fake news: Cross-domain fake news detection using multi-modal data. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 35, pages 557–565.

Craig Silverman. 2016. This analysis shows how viral fake election news stories outperformed real news on facebook. Accessed: November, 2016.

Hongyan Tang, Junning Liu, Ming Zhao, and Xudong Gong. 2020. Progressive layered extraction (ple): A novel multi-task learning (mtl) model for personalized recommendations. In *Fourteenth ACM Conference on Recommender Systems*, pages 269–278.

Andreas Vlachos and Sebastian Riedel. 2014. Fact checking: Task definition and dataset construction. In *Proceedings of the ACL 2014 workshop on language technologies and computational social science*, pages 18–22.

Soroush Vosoughi, Deb Roy, and Sinan Aral. 2018. The spread of true and false news online. *Science*, 359(6380):1146–1151.

William Yang Wang. 2017. "liar, liar pants on fire": A new benchmark dataset for fake news detection. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 422–426.

Yaqing Wang, Fenglong Ma, Zhiwei Jin, Ye Yuan, Guangxu Xun, Kishlay Jha, Lu Su, and Jing Gao. 2018. EANN: Event adversarial neural networks for multi-modal fake news detection. In *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, pages 849–857, London, United Kingdom. Association for Computing Machinery.

Yaqing Wang, Fenglong Ma, Haoyu Wang, Kishlay Jha, and Jing Gao. 2021. Multimodal emergent fake news detection via meta neural process networks. In *Proceedings of the 27th ACM SIGKDD Conference on Knowledge Discovery & Data Mining*, pages 3708–3716.

Weizhi Xu, Junfei Wu, Qiang Liu, Shu Wu, and Liang Wang. 2022. Evidence-aware fake news detection with graph neural networks. In *Proceedings of the ACM Web Conference 2022*, pages 2501–2510.

Zichao Yang, Zhiting Hu, Chris Dyer, Eric P Xing, and Taylor Berg-Kirkpatrick. 2018. Unsupervised text style transfer using language models as discriminators. In *Proceedings of the 32nd International Conference on Neural Information Processing Systems*, pages 7298–7309.

Dani Yogatama, Yishu Miao, Gabor Melis, Wang Ling, Adhiguna Kuncoro, Chris Dyer, and Phil Blunsom. 2018. Memory architectures in recurrent neural network language models. In *International Conference on Learning Representations*.

Xueyao Zhang, Juan Cao, Xirong Li, Qiang Sheng, Lei Zhong, and Kai Shu. 2021. Mining dual emotion for fake news detection. In *Proceedings of the Web Conference 2021*, pages 3465–3476, Ljubljana, Slovenia. Association for Computing Machinery.

Xinyi Zhou, Apurva Mulay, Emilio Ferrara, and Reza Zafarani. 2020. Recovery: A multimodal repository for covid-19 news credibility research. In *Proceedings of the 29th ACM International Conference on Information & Knowledge Management*, pages 3205–3212.

Yongchun Zhu, Qiang Sheng, Juan Cao, Qiong Nan, Kai Shu, Minghui Wu, Jindong Wang, and Fuzhen Zhuang. 2022. Memory-guided multi-view multi-domain fake news detection. *IEEE Transactions on Knowledge and Data Engineering*.

Yongchun Zhu, Dongbo Xi, Bowen Song, Fuzhen Zhuang, Shuai Chen, Xi Gu, and Qing He. 2020. Modeling users' behavior sequences with hierarchical explainable network for cross-domain fraud detection. In *Proceedings of The Web Conference 2020*, pages 928–938.

Fuzhen Zhuang, Zhiyuan Qi, Keyu Duan, Dongbo Xi, Yongchun Zhu, Hengshu Zhu, Hui Xiong, and Qing He. 2020. A comprehensive survey on transfer learning. *Proceedings of the IEEE*, 109(1):43–76.

A Supplementary Experiments

A.1 Experiments on Chinese 6-domain and Chinese 9-domain Dataset

We list our experimental results on the Chinese 6-domain and Chinese 9-domain datasets in Table 7.

A.2 Experiments on Other Target Domains

In this section, we set other domains, which have not been chosen as target domains in Section 4, and evaluate the performance of the proposed DITFEND on these domains. Experimental results are shown in Table 8.

Table 7: Performance comparison of base models with and without DITFEND on Chinese 6-domain dataset and Chinese 9-domain dataset.

Method	Ch-6 dataset						Ch-9 dataset					
	target: Health			target: Finance			target: Health			target: Politics		
	F1	AUC	Acc	F1	AUC	Acc	F1	AUC	Acc	F1	AUC	Acc
BiGRU	0.8626	0.9442	0.8630	0.8254	0.9281	0.8642	**0.8868**	0.9574	**0.8870**	0.8356	0.9119	0.8590
+*DITFEND*	**0.8991**	**0.9676**	**0.8992**	**0.8431**	**0.9343**	**0.8755**	0.8789	**0.9630**	0.8794	**0.8749**	**0.9334**	**0.8925**
TextCNN	0.8832	**0.9670**	0.8835	**0.8646**	**0.9498**	**0.8943**	0.8768	0.9556	0.8770	0.8561	0.9225	0.8813
+*DITFEND*	**0.8922**	0.9631	**0.8925**	0.8581	0.9436	0.8887	**0.9093**	**0.9742**	**0.9094**	**0.8768**	**0.9235**	**0.8962**
RoBERTa	0.9100	0.9644	0.9100	0.8700	**0.9553**	0.8989	0.9090	0.9611	0.9090	0.8366	0.9034	0.8637
+*DITFEND*	**0.9175**	**0.9769**	**0.9175**	**0.8768**	0.9538	**0.9044**	**0.9115**	**0.9739**	**0.9115**	**0.8775**	**0.9199**	**0.8982**
EANN	0.9150	0.9761	0.9150	**0.8621**	**0.9483**	0.8906	0.9150	0.9762	0.9150	0.8705	0.9176	0.8918
+*DITFEND*	**0.9224**	**0.9767**	**0.9225**	0.8538	0.9466	**0.8918**	**0.9250**	**0.9797**	**0.9250**	**0.8822**	**0.9367**	**0.9035**
MMOE	0.9260	0.9754	0.9260	0.8546	0.9501	0.8887	**0.9364**	**0.9774**	**0.9365**	0.8620	0.9314	0.8842
+*DITFEND*	**0.9271**	**0.9756**	**0.9271**	**0.8611**	**0.9563**	**0.8906**	0.9199	0.9655	0.9200	**0.8633**	**0.9357**	**0.8860**
MOSE	0.9118	0.9720	0.9120	**0.8639**	0.9500	**0.8921**	**0.9179**	0.9700	**0.9160**	0.8673	0.9388	0.8918
+*DITFEND*	**0.9200**	**0.9735**	**0.9200**	0.8502	**0.9533**	0.8830	0.9050	**0.9725**	0.9050	**0.8681**	**0.9422**	**0.8972**
EDDFN	0.9280	0.9774	0.9280	0.8456	0.9436	0.8830	0.9379	0.9807	0.9380	0.8478	0.9292	0.8754
+*DITFEND*	**0.9350**	**0.9821**	**0.9350**	**0.8801**	**0.9434**	**0.9019**	**0.9399**	**0.9821**	**0.9400**	**0.8507**	**0.9308**	**0.8772**
MDFEND	0.9430	0.9851	0.9430	0.8749	0.9610	0.9023	0.9425	0.9846	0.9425	0.8774	0.9370	0.8994
+*DITFEND*	**0.9451**	**0.9876**	**0.9450**	**0.8800**	**0.9649**	**0.9100**	**0.9500**	**0.9864**	**0.9500**	**0.8986**	**0.9541**	**0.9181**

Table 8: Performance comparison (f1-score) of base models with and without DITFEND when assigning other target domains on Chinese 9-domain dataset.

target domain	science	military	Edu.	Disaster	Politics	Health	Finance	Ent.	Society
BiGRU	0.7269	0.8724	0.8138	0.7935	0.8356	**0.8868**	0.8291	0.8629	**0.8485**
+*DITFEND*	**0.8061**	**0.8604**	**0.9089**	**0.8017**	**0.8749**	0.8789	**0.8665**	**0.8840**	0.8483
TextCNN	0.7254	0.8839	0.8362	0.8222	0.8561	0.8768	0.8638	0.8456	0.8540
+*DITFEND*	**0.7283**	**0.8882**	**0.8829**	**0.8301**	**0.8768**	**0.9093**	**0.8665**	**0.8840**	**0.8769**
RoBERTa	0.7777	0.9072	0.8331	0.8512	0.8366	0.9090	0.8735	0.8769	0.8577
+*DITFEND*	**0.8107**	**0.9129**	**0.8384**	**0.8574**	**0.8775**	**0.9115**	**0.8889**	**0.8889**	**0.8881**
EANN$_T$	0.8225	0.9274	**0.8624**	0.8666	0.8705	0.9150	0.8710	0.8957	**0.8877**
+*DITFEND*	**0.8271**	**0.9419**	0.8582	**0.9060**	**0.8833**	**0.9250**	**0.8833**	**0.9100**	0.8874
MMOE	**0.8755**	0.9112	0.8706	0.8770	0.8620	**0.9364**	0.8567	0.8886	0.8750
+*DITFEND*	0.8587	**0.9186**	**0.8769**	**0.8836**	**0.8792**	0.9350	**0.8786**	**0.8923**	**0.8895**
MOSE	0.8502	0.8858	0.8815	**0.8672**	0.8808	0.9179	0.8672	0.8913	0.8729
+*DITFEND*	**0.8656**	**0.9121**	**0.8918**	0.8627	**0.8897**	**0.9200**	**0.8846**	**0.8987**	**0.8832**
EDDFN	**0.8186**	**0.9137**	0.8676	0.8786	0.8478	0.9379	0.8636	0.8832	0.8689
+*DITFEND*	0.8061	0.8834	**0.8887**	**0.8855**	**0.8492**	**0.9399**	**0.8647**	**0.8889**	**0.8754**
MDFEND	0.8301	0.9389	0.8917	0.9003	0.8865	0.9400	0.8951	0.9066	0.8980
+*DITFEND*	**0.8426**	**0.9419**	**0.9091**	**0.9145**	**0.8895**	**0.9550**	**0.8984**	**0.9100**	**0.8991**

A.3 Experiments on New Target Domains

In this section, we perform experiments to exclude target domain data when training the general model, in order to testify how our proposed framework performs when the target domain is unseen during general model training procedure.

We show the experimental results in Table 9 and Table 10. From the experimental results, we can find that our framework DITFEND can bring significant improvements to all domains.

Table 9: Performance comparison of base models with and without DITFEND on English dataset and Chinese 3-domain dataset when the target domain is unseen during meta-training procedure.

Method	English dataset						Ch-3 dataset					
	target: COVID			target: Politifact			target: Health			target: Politics		
	F1	AUC	Acc	F1	AUC	Acc	F1	AUC	Acc	F1	AUC	Acc
BiGRU	0.7448	0.9114	0.8606	0.7339	0.8213	0.7375	0.8577	0.9367	0.8580	0.8384	0.9032	0.8637
+*DITFEND*	**0.8949**	**0.9650**	**0.8950**	**0.8406**	**0.9052**	**0.8406**	**0.8746**	**0.9609**	**0.8750**	**0.8633**	**0.9316**	**0.8889**
TextCNN	0.8322	0.9397	0.8955	0.7040	0.8046	0.7064	0.8716	0.9552	0.8720	0.8579	0.9067	**0.8859**
+*DITFEND*	**0.8549**	**0.9658**	**0.9042**	**0.7851**	**0.8632**	**0.7850**	**0.8949**	**0.9650**	**0.8950**	**0.8587**	**0.9615**	0.8810
RoBERTa	0.9014	0.9770	0.9377	0.7967	0.9078	0.7989	0.8955	0.9641	0.8955	0.8300	0.8948	0.8628
+*DITFEND*	**0.9305**	**0.9800**	**0.9509**	**0.8493**	**0.9335**	**0.8497**	**0.9250**	**0.9743**	**0.9250**	**0.8302**	**0.9027**	**0.8655**
EANNT	0.8836	0.9751	0.9282	0.7558	0.8612	0.7584	0.9189	**0.9787**	0.9190	0.8405	0.9074	0.8690
+*DITFEND*	**0.9507**	**0.9896**	**0.9667**	**0.8664**	**0.9597**	**0.8671**	**0.9200**	0.9768	**0.9200**	**0.8615**	**0.9177**	**0.8889**
MMOE	0.9379	0.9883	**0.9588**	0.8477	0.9408	0.8486	0.9215	0.9679	0.9215	**0.8779**	**0.9388**	**0.8982**
+*DITFEND*	**0.9356**	**0.9907**	0.9586	**0.8542**	**0.9482**	**0.8555**	**0.9299**	**0.9735**	**0.9300**	0.8659	0.9331	0.8896
MOSE	0.9326	0.9879	0.9588	0.8576	0.9447	0.8590	**0.9023**	0.9683	**0.9025**	0.8564	**0.9138**	0.8795
+*DITFEND*	**0.9547**	**0.9889**	**0.9692**	**0.8725**	**0.9528**	**0.8728**	0.9010	**0.9685**	0.9012	**0.8568**	0.9137	**0.8812**
EDDFN	0.9306	0.9891	0.9547	0.8505	0.9432	0.8509	**0.9235**	0.9735	**0.9235**	0.8440	0.9207	**0.8702**
+*DITFEND*	**0.9383**	**0.9899**	**0.9586**	**0.8721**	**0.9466**	**0.8728**	0.9150	**0.9762**	0.9150	**0.8441**	**0.9337**	0.8663
MDFEND	0.9331	0.9874	0.9565	0.8473	0.9391	0.8485	0.9419	0.9855	0.9420	**0.8555**	0.9259	0.8854
+*DITFEND*	**0.9482**	**0.9932**	**0.9659**	**0.8599**	**0.9503**	**0.8613**	**0.9550**	**0.9877**	**0.9550**	0.8507	**0.9442**	**0.8772**

Table 10: Performance comparison of base models with and without DITFEND on Chinese 6-domain dataset and Chinese 9-domain dataset when the target domain is unseen during meta-training procedure.

Method	Ch-6 dataset						Ch-9 dataset					
	target: Health			target: Finance			target: Health			target: Politics		
	F1	AUC	Acc	F1	AUC	Acc	F1	AUC	Acc	F1	AUC	Acc
BiGRU	0.8626	0.9442	0.8630	0.8254	0.9281	0.8642	**0.8868**	0.9574	**0.8870**	0.8356	0.9119	0.8590
+*DITFEND*	**0.8875**	**0.9548**	**0.8875**	**0.8400**	**0.9295**	**0.8710**	0.8700	**0.9612**	0.8715	**0.8541**	**0.9305**	**0.8816**
TextCNN	0.8832	**0.9670**	0.8835	**0.8646**	**0.9498**	**0.8943**	0.8768	0.9556	0.8770	0.8561	0.9225	0.8813
+*DITFEND*	**0.8901**	0.9608	**0.8902**	0.8547	0.9431	0.8814	**0.8854**	**0.9715**	**0.8855**	**0.8650**	**0.9230**	**0.8879**
RoBERTa	0.9100	0.9644	0.9100	0.8700	**0.9553**	0.8989	0.9090	0.9611	0.9090	0.8366	0.9034	0.8637
+*DITFEND*	**0.9249**	**0.9666**	**0.9250**	**0.8781**	0.9347	**0.9057**	**0.9149**	**0.9723**	**0.9150**	**0.8999**	**0.9294**	**0.9181**
EANNT	0.9150	0.9761	0.9150	0.8621	0.9483	0.8906	0.9150	**0.9762**	0.9150	0.8705	0.9176	0.8918
+*DITFEND*	**0.9184**	**0.9753**	**0.9184**	**0.8789**	**0.9649**	**0.9094**	**0.9196**	0.9676	**0.9196**	**0.8733**	**0.9178**	**0.8940**
MMOE	**0.9260**	**0.9754**	**0.9260**	0.8546	0.9501	0.8887	**0.9364**	**0.9774**	**0.9365**	**0.8620**	0.9314	0.8842
+*DITFEND*	0.9145	0.9732	0.9150	**0.8548**	**0.9503**	**0.8888**	0.9200	0.9663	0.9200	0.8496	**0.9459**	**0.8843**
MOSE	**0.9118**	0.9720	**0.9120**	0.8639	0.9500	0.8921	**0.9179**	0.9700	**0.9160**	**0.8673**	**0.9388**	**0.8918**
+*DITFEND*	0.9110	**0.9725**	0.9112	**0.8640**	**0.9512**	**0.8925**	0.9048	**0.9715**	0.9049	0.8540	0.9386	0.8713
EDDFN	0.9280	**0.9774**	0.9280	0.8456	0.9436	0.8830	0.9379	**0.9807**	0.9380	**0.8478**	**0.9292**	**0.8754**
+*DITFEND*	**0.9299**	0.9745	**0.9300**	**0.8471**	**0.9497**	**0.8868**	**0.9399**	0.9768	**0.9400**	0.8240	0.8951	0.8596
MDFEND	0.9430	0.9851	0.9430	0.8749	**0.9610**	0.9023	0.9425	0.9846	0.9425	**0.8774**	0.9370	**0.8994**
+*DITFEND*	**0.9450**	**0.9900**	**0.9450**	**0.8792**	0.9486	**0.9057**	**0.9600**	**0.9892**	**0.9600**	0.8713	**0.9452**	0.8947

Student Surpasses Teacher: Imitation Attack for Black-Box NLP APIs

Qiongkai Xu[*1] **Xuanli He**,[2] **Lingjuan Lyu**,[3] **Lizhen Qu**,[2] **Gholamreza Haffari**[2]
[1]The University of Melbourne, [2]Monash University, [3]Sony AI
Qiongkai.Xu@unimelb.edu.au, Lingjuan.Lv@sony.com
{xuanli.he1,Lizhen.Qu,Gholamreza.Haffari}@monash.edu

Abstract

Machine-learning-as-a-service (MLaaS) has attracted millions of users to their splendid large-scale models. Although published as black-box APIs, the valuable models behind these services are still vulnerable to imitation attacks. Recently, a series of works have demonstrated that attackers manage to steal or extract the victim models. Nonetheless, none of the previous stolen models can outperform the original black-box APIs. In this work, we conduct unsupervised domain adaptation and multi-victim ensemble to showing that attackers could potentially surpass victims, which is beyond previous understanding of model extraction. Extensive experiments on both benchmark datasets and real-world APIs validate that the imitators can succeed in outperforming the original black-box models on transferred domains. We consider our work as a milestone in the research of imitation attack, especially on NLP APIs, as the superior performance could influence the defense or even publishing strategy of API providers.

1 Introduction

Task-oriented NLP APIs have received tremendous success, partly due to commercial cloud services (Krishna et al., 2019; Wallace et al., 2020). The enormous commercial benefit allures other companies or individual users to extract the backend decision models of these successful APIs. Some recent works have demonstrated that many existing NLP APIs can be locally imitated or extracted (Krishna et al., 2019; Wallace et al., 2020; Guo et al., 2022), violating the intellectual property (IP) of NLP APIs. Equipped with recent advanced models pre-trained on large-scale corpus, it is getting easier to train a decent attack model with limited training samples retrieved from victims (He et al., 2021a).

Despite their success, there are still a series of restrictions on current attack paradigm. The first restriction is that attackers and victims are trained and evaluated under the same domain. Although such setting simplifies the comparison of utility and fidelity between victim and attack models, it is less likely to be the case in real world. Generally, attackers and victims do not, at least not willing to, share their in-house datasets to the public. Moreover, the competition between attackers and victims lies in their performance on the application scenarios (transferred domains), which vary by customers not API providers. The second restriction is that traditional model extraction only refers a single victim model, and none of the previous works manage to build the extracted models that can surpass the original black-box APIs (Tramèr et al., 2016; Krishna et al., 2019; Wallace et al., 2020). Aggregating the strength from diverse victims, however, can potentially benefit the performance of attackers. This may further enable the attack model to acquire a superior performance, in the best case, even surpassing all the corresponding victim models.

Based on the above analysis, we are motivated to address their restrictions by *i)* conducting imitation attacks on a transferred domain and *ii)* utilizing multiple victim ensemble. Our approach integrates unsupervised domain adaptation and model ensemble into the imitation attack. We conduct experiments on two representative NLP tasks, namely sentiment classification (Lyu et al., 2020) and machine translation (Wallace et al., 2020). We investigate the attack performance on both the locally simulated victim models and publicly available commercial NLP APIs. Our results demonstrate that the attackers could potentially achieve better performance than victims in the transferred domains and utilizing multiple victim models further improves the performance of the attack models. For those target domains that are relatively far from victims' source domains, *e.g.*, sentiment analysis

[*] Most of the work was finished when the first author was at ANU and Data61 CSIRO.

Proceedings of the 29th International Conference on Computational Linguistics, pages 2849–2860
October 12–17, 2022.

models on movie review domain to financial document domain, the performance improvement of the imitation model could be further amplified.

Overall, our empirical findings exacerbate the potential risk concerns of public API services, as malicious companies could provide a better service in their specific domains by integrating several publicly available APIs. Moreover, the new services generally could cost far less than any of the original victim services or the wage paid to human annotators. The imitation attack not only infringes the IP of victim companies by misusing the predictions of their APIs, but also potentially corrupts the MLaaS market by publishing new APIs, with higher performance but lower price. We believe that explicitly exposing the new imitation attack paradigm would arouse significant attentions in the research community and encourage companies to reconsider their strategies of publishing API services.

2 Related Work

2.1 Model Imitation Attack

Model imitation attack (also referred to as model "extraction" or "stealing") has been studied for simple classification tasks (Tramèr et al., 2016), vision tasks (Orekondy et al., 2019), NLP tasks (Krishna et al., 2019; Wallace et al., 2020), and *etc*. Generally, model imitation attack aims to reconstruct a local copy or to steal the functionality of a black-box API. If the reconstruction is successful, the attacker has effectively stolen the intellectual property. Previous works on imitation attack (Krishna et al., 2019; Wallace et al., 2020) mainly focus on how to imitate a model with performance approximate to the victim API in the source domain. Whether a more powerful attacker can steal a model that is better than any victim API in new domains is largely unexplored.

2.2 Unsupervised Domain Adaption

Domain adaptation is a task of adapting a pre-trained model on a source domain to a target domain. It can be fulfilled in two mechanisms: supervised adaptation and unsupervised adaptation. The former can achieve an outstanding performance with a small amount of in-domain labeled data (Daumé III, 2007; Ben-David et al., 2007). In contrast, the unsupervised domain adaptation (UDA) (Miller, 2019; Ganin and Lempitsky, 2015) does not require ground-truth labels in the target domain, hence is more challenging but attractive.

This work is under the umbrella of UDA family (Wang et al., 2021; Miller, 2019). We differentiate our work from other UDA works in terms of the intent. Other UDA works aim to improve the models in both source domains and target domains simultaneously, while our proposed imitation attack mainly focuses on optimizing the adapted attack models merely in target domains. We exploit UDA from a dark side, where one can leverage domain adaptation to violate IP of commercial APIs, but benefit from such violations.

2.3 Ensemble for Knowledge Distillation

Model imitation attack is related to knowledge distillation (KD) (Hinton et al., 2015). Emergent knowledge distillation (Buciluǎ et al., 2006; Hinton et al., 2015) aims to transfer knowledge from a large teacher model to a small student model for the sake of model compression. By encouraging the student model to approximate the behaviors of the teacher model, the student is able to imitate the teacher's behavior with minor quality loss but more efficient inference (Furlanello et al., 2018; Dvornik et al., 2019; Anil et al., 2018).

However, model imitation differs from distillation in terms of data usage. Particularly, the victim's (*i.e.*, teacher's) training data is usually unknown as the victim API is deployed in a black-box manner. Thus, malicious users tend to consider unlabeled queries more likely to be collected from other domains (Xie et al., 2020) or generated from pre-trained generative models (Ravuri and Vinyals, 2019; Kumar et al., 2020; Besnier et al., 2020). Our work considers a more realistic setting, as victim and attack models are under different domains and express distinctive interests.

On the other hand, with the development of multi-task learning, multi-teacher distillation has been proposed, which is targeted at distilling multiple single-task models into a single multi-task model (Tan et al., 2019; Clark et al., 2019; Saleh et al., 2020). These lines of works lie in the spirit of improving model performance and encouraging parameter reduction via knowledge distillation.

The dichotomy between our work and previous works is we are interested in an ensemble distillation of single-task models, where all teachers and the student work on the same task, but could be in different domain. The gist of this approach aims to leverage the collective wisdom to obtain a better student outperforming its teachers, which is sim-

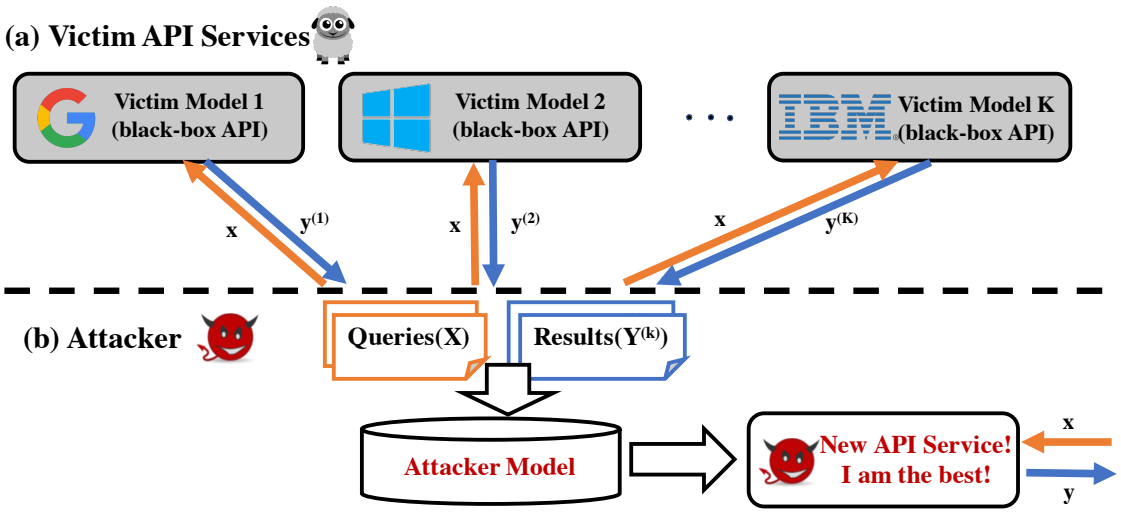

Figure 1: The workflow of the imitation attack and its harm. The attacker labels queries (x) using multiple victim APIs, and then trains an attack model on the resulting data. Finally, the attacker could publish a new API service, which could become a competitor of the victim services, thus infringing intellectual properties and eroding MLaaS market shares of the victim services.

ilar to the ensemble learning (Opitz and Maclin, 1999). However, ensemble learning focuses more on the aggregation of the predictions of different models at the inference stage, whereas our proposed method not only gathers the inputs and their predictions, but also trains a new model on the input-prediction pairs.

3 Imitation Attack Paradigm

In this section, we first explain the problem and motivation of imitation attack (IMA) under domain adaptation setting. Then, we connect our IMA practice with a corresponding domain adaptation theory (Ben-David et al., 2010; Wang et al., 2021). After that, we introduce a multi-victim ensemble methodology for IMA. Finally, we explain the rationality of a family of existing defense technologies under the domain adaptation theory.

3.1 Problem Statement

In the real world, attackers may be interested in their own new business, *e.g.*, a new classification or machine translation system in a new domain. Generally, the attackers possess their own training samples, $\mathcal{X} = \{x_1, x_2, \cdots, x_N\}$, while the oracle labels $\mathcal{Y} = \{y_1, y_2, \cdots, y_N\}$ of these samples are not yet labeled. In order to train a model with the least cost of annotation, attackers access the publicly available commercial APIs for the target task $\mathcal{T}$. Moreover, the attackers could inquire multiple APIs for further performance improvement. The underlying models of the attacked APIs are K victim models, $\mathcal{V} = \{h_v^{(1)}, h_v^{(2)}, \cdots, h_v^{(K)}\}$. As illustrated in Figure 1, our attack can be formulated as a two-step process:

1. The attacker queries the k-th victim model $h_v^{(k)}$ with samples in $\mathcal{X}$, and retrieves corresponding labels $\mathcal{Y}^{(k)} = \{h_v^{(k)}(x_1), h_v^{(k)}(x_2), \cdots, h_v^{(k)}(x_N)\}$.

2. The attacker learns an imitation (attack) model $h_a(x)$ based on the queries and concatenated retrieved labels, $\langle \mathcal{X}, \mathcal{Y}^{(k)} \rangle|_{k=1}^{K}$.

As the attacking process involves teaching a local model by imitating the behavior of the victim models, we put this attack process under the umbrella of imitation attack. The motivation of such attack paradigm is twofold:

- Firstly, querying commercial APIs generally costs far less than hiring human annotators. Therefore, the price for training the attack models using imitated labels is competitive in the market;

- Secondly, the attackers are potentially able to outperform the victims in terms of utility. As demonstrated in Section 5, domain adaptation and multi-victim ensemble further enhance the attacker performance.

We believe that both price and performance advantages would lure companies or individuals to imitation attack.

3.2 Training for Imitation Attack

In imitation attack, the attacker utilizes the labels from victim APIs for model training. Given a victim model $h_v(x)$, the attacker model $h_a(x)$ imitates the behavior of victim model by minimizing the prediction error on the target domain $\mathcal{T}$,

$$\min_{h_a} \mathbb{E}_{x\sim\mathcal{T}} \left[\mathcal{L}(h_a(x), h_v(x))\right]. \quad (1)$$

On the other hand, we assume that the victim model has learned from oracle annotations y in another source domain $\mathcal{S}$, although those labels are never used for training attacker in our IMA. The loss for training victim model is

$$\min_{h_v} \mathbb{E}_{x\sim\mathcal{S}} \left[\mathcal{L}(h_v(x), y)\right], \quad (2)$$

where $\mathcal{L}(\cdot,\cdot)$ is the loss function. In practice, we use cross entropy as loss functions for training both the victim and attack models. Note that jointly optimizing Equation 1 and Equation 2 can derive the unsupervised domain adaptation loss in (Miller, 2019; Ganin and Lempitsky, 2015). However, our approach optimizes victim and attack models in absolutely separate steps, as victim models and their training processes are black-box to the attackers.

3.3 Imitation Attack as Domain Adaptation

We connect our new IMA paradigm with domain adaptation theory. The error of the attacker is measured by attacker risk $\epsilon_a(h, h_a) := \mathbb{E}_{x\sim\mathcal{T}}[|h(x) - h_a(x)|]$. According to domain adaptation theory (Ben-David et al., 2010; Wang et al., 2021), the upper bound of the attacker risk is

$$\begin{aligned}\epsilon_a(h, h_a) \le \epsilon_v(h, h_v) + 2d(P^s_X, P^t_X) \quad (3)\\ + \min_{\mathcal{T}\in\{P^s_X, P^t_X\}} \mathbb{E}_{x\sim\mathcal{T}} \left[|h_v(x) - h_a(x)|\right],\end{aligned}$$

where $d(P^s_X, P^t_X)$ is the total variation between the distributions of source and target domains, which is a constant determined by domains of the datasets used by the victim and attacker. The first term, $\epsilon_v(h, h_v) := \mathbb{E}_{x\sim\mathcal{S}}[|h(x) - h_v(x)|]$, with oracle decisions $y = h(x)$, is the victim risk on the source domain. It is optimized during the training of victim models as Equation 2. The last term is associated with imitation training objective as Equation 1. Therefore, our imitation attack under domain adaptation setting is actually optimizing the upper bound of the attacker risk $\epsilon_a(h, h_a)$.

3.4 Multiple Victim Ensemble

Another approach to achieving further performance improvement is to integrate the results from multiple APIs. This strategy is well motivated in real-world imitation attack, as many cloud computing companies share similar APIs for main-stream NLP tasks, e.g, Google Cloud and Microsoft Azure both support sentiment classification and machine translation. Attackers can improve their performances by learning from multiple victim APIs. In more detail, given independent K victim models $h^{(k)}_v(x)$, attackers can take the average advantage of all victim models by,

$$\min_{h_a} \frac{1}{K} \sum_{k=1}^{K} \mathbb{E}_{x\sim\mathcal{T}}[\mathcal{L}(h_a(x), h^{(k)}_v(x))]. \quad (4)$$

According to ensemble theories (Breiman, 2001; Bauer and Kohavi, 1999), the lower generalization error of an ensemble model depends on *i)* better performance of the individual models, and *ii)* lower correlation between them. In real-world, companies are actually *i)* targeting on API with better performance, and *ii)* using their own private training datasets. The effort of these companies towards superior API performance unfortunately exacerbates these two factors for a successful ensemble model as an attacker.

3.5 Defending Imitation Attack

Some existing IMA defense strategies (He et al., 2021a) slightly distort the predictions of victim models to reduce the performance of attackers, by replacing $h_v(x)$ with noisy $h'_v(x)$. The variance introduced by distortion is $\Delta(h_v, h'_v) := \mathbb{E}_{x\sim\mathcal{S}}[|h_v(x) - h'_v(x)|]$. As the distortion should not destroy the utility of victim model, the variance should be bounded by a small constant δ, *i.e.*, $\Delta(h_v, h'_v) \le \delta$. The victim risk, the first term in (3), is relaxed to $\epsilon_v(h, h'_v)$, where

$$\epsilon_v(h, h'_v) \le \epsilon_v(h, h_v) + \Delta(h_v, h'_v). \quad (5)$$

Therefore, the gentle distortion of victim outputs results in a more relaxed upper bound for optimization, which could potentially lead to better results on defending the imitation attack.

4 Experimental Setup

4.1 Tasks and Datasets

In this paper, we focus on two essential NLP tasks, classification and machine translation, both

of which are predicting discrete outputs, and cross entropy loss is used as the objective in optimization. For classification tasks, as APIs provide continuous scores of the predictions, we consider the settings of soft label (using prediction scores) and hard label (using predicted categories). In machine translation, the translation result is a sequence of discrete words, which are considered as sequential hard labels. Classification and translation tasks are evaluated by accuracy(%) (Schütze et al., 2008) and BLEU (Papineni et al., 2002), respectively. The datasets used in our imitation attack experiment are summarized in Table 1.

Sentiment Classification. Imdb Movie Review (**IMDB**) (Maas et al., 2011) is a large-scale movie review dataset for sentiment analysis. Stanford Sentiment (**SST**) (Socher et al., 2013) is another movie review dataset with relatively shorter text than **IMDB**. Financial Sentiment (**FST**) (Malo et al., 2014) provides sentiment labels on economic texts in finance domain. We use **IMDB** to train local victim models and consider **SST** and **FST** as target domains in attack.

Machine Translation. We consider German (De) to English (En) translation as our testbed. We first study the attack performance on the local models trained on a general domain. Specifically, we use WMT14 (Bojar et al., 2014) to train the victim models. Then, we investigate the imitation attack on **Law** and **Koran** domains from OPUS (Tiedemann, 2012). We utilize Moses[1] to pre-process all corpora, and keep the text cased. A 32K BPE vocabulary (Sennrich et al., 2016) is applied to all datasets.

4.2 Victim Models

Locally Simulated NLP Services. We first use models trained on our local server as local services. Our models are trained on datasets in source domain, *i.e.*, **IMDB** for sentiment analysis and **WMT14** for machine translation. We vary BERT (Devlin et al., 2019) and RoBERTa (Liu et al., 2019) as pre-trained models for classification. Transformer-base (TF-base) and Transformer-big (TF-big) (Vaswani et al., 2017) are used as machine translation architectures, more details about hyper-parameter settings are described in Appendix A.

Commercial NLP API Services. To investigate the performance of imitation attack on real-world commercial NLP APIs, we query and retrieve the results of victim APIs for both sentiment analysis and machine translation. **Google Cloud** API[2] and **IBM Cloud** API[3] are inquired for sentiment analysis. **Google Trans**[4] and **Bing Trans**[5] are used as translation APIs. In this setting, we assume that different companies have various choices in datasets, domains, model architectures and training strategies. These settings are invisible to the attackers.

4.3 Imitation Attack Setup

For imitation attack, different from Wallace et al. (2020), we leverage datasets in other domains rather than those used for training victim models. The rationale behind this setting is that *i)* the owners of APIs tend to use in-house dataset, which is difficult for attacker to access; *ii)* attackers are more interested in their own private dataset, which is also not visible to others. Therefore, our setting is closer to the real-world attack scenario. The attack models are trained based on the labels retrieved from victim models, and tested on the labels from the human-annotated sets. We consider **SST** and **FST** as target domains for sentiment analysis. For machine translation, we use **Law** and **Koran**. In attack, we use BERT for classification. Regarding machine translation, Transformer-base is used for simulating local APIs, while mBART (Liu et al., 2020) for experiments on commercial APIs[6]. We also investigate the ensemble models, by concatenating all the outputs retrieved from multiple victim models in training.

5 Experimental Results

In this section, we analyze our experimental results. Our experiments are designed to answer the following research questions (**RQ**s),

- **RQ1**: Are the attack models able to outperform the victim models in new domains?
- **RQ2**: Will the ensemble of victim APIs improve the performance of the attack models?
- **RQ3**: Do traditional defense methods help APIs reduce the performance of attackers in our domain adaptation setting?

[1] https://github.com/moses-smt/mosesdecoder

[2] https://cloud.google.com

[3] https://cloud.ibm.com

[4] https://translate.google.com/

[5] https://www.bing.com/translator

[6] We observe that mBART works better on attacking commercial APIs than Transformers in our preliminary experiments.

Dataset	#Train	#Dev	#Test	Task	Domain
IMDB	25,000	25,000	N/A	Sentiment Classification	Movie Review (long)
SST	6,920	872	1,821	Sentiment Classification	Movie Review (short)
FST	1,413	159	396	Sentiment Classification	Finance Document
WMT14	4.5M	3,000	N/A	Machine Translation	General
JRC-Acquis	2M	1,000	1,000	Machine Translation	Law
Tanzil	579k	1,000	1,000	Machine Translation	Koran

Table 1: Statistic of sentiment classification and machine translation datasets, with number of samples in train, dev and test sets. The task name and corresponding domain for each dataset are also included.

	Sentiment Classification				Machine Translation			
Model	**Label**	**Arch.**	**SST**	**FST**	**Label**	**Arch.**	**Law**	**Koran**
In Domain	SST / FST	BERT	91.49	97.72	Law / Koran	TF base	38.52	19.49
Victim 1	IMDB	BERT	87.92	74.94	WMT14	TF-base	23.33	9.82
Victim 2	IMDB	RoBERTa	89.40	80.00	WMT14	TF-big	24.33	10.33
Attack_{s1}	Victim 1	BERT	90.13	83.59	Victim 1	TF-base	23.82	10.04
Attack_{s2}	Victim 2	BERT	90.72	88.76	Victim 2	TF-base	25.48	10.30
Attack_{m}	Victim 1+2	BERT	**91.57**	**90.53**	Victim 1+2	TF-base	**25.74**	**10.48**

Table 2: Experimental results of imitation attack on single and multiple victim models with settings, labels used for training and model architectures (Arch.). Oracle models are trained on the human-annotated datasets in target domain and victim models are trained on corpora from source domain. For all attack experiments, we report the mean results over 5 runs. Attackers using single victim and multiple victims are indicated as Attack_s and Attack_m. *In domain* models are trained on human labeled training sets in the target domain, which could arguably be considered as upper bounds of those attack models. This experiment focuses on comparing Victim and Attack.

Locally Simulated Experiments. We first conduct the experiments of imitation attack on local models, shown in Table 2. The models trained on oracle human annotated datasets are much better than the victim models, as the later ones are trained on other domains. All our attack models outperform the original victim models for both classification and translation tasks. We attribute this performance improvement to unsupervised domain adaptation. Ensemble models consistently work better than each of the single model[7]. For SST, although using the same architecture, the attack model trained on the ensemble of two victims surprisingly outperforms the model supervised by oracle labels. This result also outperforms some competitive supervised baselines (Tang et al., 2019; McCann et al., 2017; Zhou et al., 2016). This observation suggests that, in some scenarios, it is possible to achieve decent results only based on some open APIs, without relatively more expensive human annotations. As a result, some annotators could lose their working opportunities of labeling new datasets, and some API services might lose their market share in new domains or tasks.

Experiments on Commercial APIs. We then demonstrate the vulnerability of some real-world commercial APIs to our IMA approach, in Table 3. For classification task, the attacker uses soft labels, as *i)* these APIs provide such scores and *ii)* attackers using soft labels achieve better performance than hard attacks in our defense experiments (see Table 5). For machine translation, only hard label could be used, as we can only have token sequences without their perplexity scores from commercial APIs. In all attacks on classification and translation APIs, the attackers manage to achieve significantly better results than the corresponding victim models, with frighteningly low costs. Combining two commercial APIs generally improves the performance of the attackers.We observe that commercial APIs work quite poor on FST, as it belongs to a more professional domain. However, the performance of the attacker catches up significantly on FST and achieves an averaged accuracy of 89.82%, given poor competitors (victims) both with accuracy less than 70%. Google+Bing on Law is the only ensem-

[7]More discussion on averaging strategies for ensemble the classification APIs is discussed in Appendix B.

Task	# Queries	API	Cost	Victim	Attacker	Improv.
SST	9,613	**Google Cloud**	$5	84.62	88.26 ± 0.22	+3.64
		IBM Cloud	Free	87.26	89.17 ± 0.33	+1.91
		Google+IBM	$5	-	89.75 ± 0.58	
FST	1,968	**Google Cloud**	Free	68.35	83.85 ± 1.05	+15.50
		IBM Cloud	Free	58.73	85.01 ± 0.81	+26.28
		Google+IBM	Free	-	89.82 ± 0.81	
Law	2M	**Google Trans**	$6,822	30.43	31.99 ± 0.05	+1.56
		Bing Trans	$3,396	34.22	35.45 ± 0.09	+1.23
		Google+Bing	$10,218	-	34.94 ± 0.11	
Koran	579k	**Google Trans**	$1,211	14.31	14.63 ± 0.06	+0.32
		Bing Trans	$590	13.24	13.71 ± 0.05	+0.47
		Google+Bing	$1,801	-	15.25 ± 0.09	

Table 3: A comparison of the commercial APIs (Victims) with attackers. The improvement (Improv.) of attackers over victims is given by rows of single models. The estimated cost of the API is based on the price queried in a single day. For all attack experiments, we report mean and standard deviations of the results over 5 runs.

ble model that fails to surpass all the single model. We attribute this to the fact that Bing Trans and its attack model has already achieved a decent result on Law, outperforming Google Trans with a gap of 3% to 4%.

Estimated Attack Costs. In Table 3, we also estimate the cost of querying the commercial APIs as victim models. We find the costs are quite affordable to many companies or even individuals, given the benefit of obtaining high-quality in-domain classification and MT systems. The price is in accordance with retrieving the results in a day, using a single month budget of a single account. The price could be further decreased, by registering more accounts or using more time. On the other hand, we estimate that the costs for human to annotate the datasets are $480.65 (**SST**), $98.4 (**FST**), $1.6M (**Law**), and $463k (**Koran**), if we hire annotators from Amazon Mechanical Turk. The price is decided as 0.05 cents for each classification sample and 0.8 dollar for each translation sample[8]. Although the price is arguably floating, we give a preliminary overview on the costs by human annotators. We find that the cost of human annotation could be 20 to 150 times higher than querying APIs. This could become another motivation for attackers to learn from APIs, instead of human annotators.

Impact of Model Ensemble. We conduct an ablation study on the potential performance improvement of attacking multiple victim models, in Table 4. The source domain training datasets are equally separated into 4 disjoint subsets, coined A, B, C and D. Then, we train 4 corresponding local victim models, model A to D, respectively. Model Full uses the victim models trained on the complete training datasets in the source domains. The attacker chooses to utilize the combined results of these victim models for training, *e.g.*, Model A+B ensembles knowledge from victim A and B. We use BERT for classification and Transformer-base for machine translation in this study. Ensemble models with more victims generally improve the attacker performance on SST and Law. The ensemble of multiple weaker models can catch up with those of full model.

		SST		Law	
Model	#V	Victim	Attack	Victim	Attack
Model A	1	85.39	87.37	20.75	22.60
Model B	1	84.51	86.22	21.01	21.67
Model C	1	86.44	87.70	20.79	21.76
Model D	1	86.60	87.31	20.68	21.80
Model A+B	2	-	88.30	-	22.87
Model A+B+C+D	4	-	89.18	-	22.97
Model Full	1	87.92	88.58	23.33	23.82

Table 4: The comparison of imitation attacks with different model ensembles.

Defense Strategies.

We compare two possible defense strategies on classification tasks in Table 5. We consider per-

[8]In our preliminary experiment, annotators manage to finish about 10 classification and 1.5 translation annotations. The corresponding wages are about $30/hr and $32/hr, higher than the minimum wage in USA.

Model	SST					FST				
	Soft	Hard	P 0.1	P 0.2	P 0.5	Soft	Hard	P 0.1	P 0.2	P 0.5
Victim 1	87.92	87.92	87.94	86.79	78.06	74.94	74.94	75.24	69.27	58.33
Victim 2	89.40	89.40	88.79	87.73	80.71	80.00	80.00	78.08	76.46	65.42
Attack_{s1}	90.44	88.58	90.23	90.07	87.98	82.03	80.25	83.65	83.75	76.41
Attack_{s2}	90.12	90.12	90.49	90.47	88.72	87.85	85.57	88.86	85.16	82.03
Attack_{m}	91.82	90.66	91.44	91.20	90.15	88.86	87.09	89.27	87.34	86.33

Table 5: The comparison of imitation attack results given victims with various defense strategies, soft label (Soft) to hard label (Hard) and noise perturbation (P) with variance (σ).

turbing the original soft outputs by hard labeling or adding Gaussian noise $\boldsymbol{n} \sim \mathcal{N}(0, \sigma^2)$, as

$$h'_v(x) = h_v(x) + \boldsymbol{n}.$$

The models utilizing hard labels manage to consistently reduce the performance of the attackers who use soft labels. Then we compare models trained on labels with various perturbation (P) by sampling random noise from Gaussian distribution with variance $\sigma \in \{0.1, 0.2, 0.5\}$ (Tramèr et al., 2016). These experiments allow the noise to flip the outputs of the original victim models, and more experiments on perturbation without influencing utility of victim models are provided in Appendix C. We observe that gently disturbing the outputs of victim models could crack down the attackers to some extent and larger noise indicates better defense. However, the harm to victims is generally larger than those to attackers. We attribute this to noise reduction in training attack model. Overall, our new IMA calls for more effective defense methods in future.

6 Discussion

We consider our imitation attack approach has achieved outperforming results that challenge the current understanding of IMA. As a result, API publishing strategies and defense methodologies should be converted accordingly.

Suggested Actions. As in domain adaptation settings, IMA manages to achieve superb performance, while attacking models are not able to outperform victims in the same domain (Krishna et al., 2019; Wallace et al., 2020), we suggest API services could cover more domains to eliminate the potential performance gain from UDA. Simply harming the utility of victim models seems not to be a wise choice for service providers, but merely providing hard labels without probability scores could avoid more superior attack performance. Adjusting the pricing strategies of publishing API services or adding watermark to API outputs (He et al., 2021b) could be plausible solutions to avoid the illegal stealing of the precious APIs of industrial companies.

Ethical Concerns. We recognize that our work could be used for malicious purposes, for example, a competing company may adopt our attack to steal a better model for commercial benefits, thus eroding the market shares of other business competitors. However, the main purpose of our work is to help commercial cloud services and regulators raise awareness of model theft, and reconsider how to deploy NLP APIs in a safe manner to avoid being stolen and exceeded. In order to minimize the potential negative influence of our work, *i)* we will delete our models and retrieved results on our local server; *ii)* we will report the vulnerability to the API providers.

Follow-up Attacks. Recent works have demonstrated that the extracted model could be used as a reconnaissance step to facilitate later attacks (He et al., 2021a; Krishna et al., 2019; Wallace et al., 2020). For instance, the adversary could use the extracted model to facilitate private information inference about the training data of the victim model, or to construct adversarial examples that force the victim model to make incorrect predictions. We leave follow-up attacks that can leverage our imitated model with better performance to future works.

7 Conclusion

We demonstrate a powerful imitation attack paradigm which may produce better attack models that surpasses the imitated models including the real-world NLP APIs via domain adaptation and ensemble. We believe such achievements would potentially influence the price decision and publishing strategies of primary NLP services. We also take the first step of grounding our new attacking approach with unsupervised domain adaptation the-

ory and model ensemble. More broadly, we hope to arouse prominent concerns on security and privacy of API services in NLP applications. In future, we plan to further explore the possibility of detecting and defending the imitation attack.

References

Rohan Anil, Gabriel Pereyra, Alexandre Passos, Robert Ormandi, George E Dahl, and Geoffrey E Hinton. 2018. Large scale distributed neural network training through online distillation. *arXiv preprint arXiv:1804.03235*.

Eric Bauer and Ron Kohavi. 1999. An empirical comparison of voting classification algorithms: Bagging, boosting, and variants. *Machine learning*, 36(1):105–139.

Shai Ben-David, John Blitzer, Koby Crammer, Alex Kulesza, Fernando Pereira, and Jennifer Wortman Vaughan. 2010. A theory of learning from different domains. *Machine learning*, 79(1):151–175.

Shai Ben-David, John Blitzer, Koby Crammer, Fernando Pereira, et al. 2007. Analysis of representations for domain adaptation. *Advances in neural information processing systems*, 19:137.

V. Besnier, H. Jain, A. Bursuc, M. Cord, and P. Pérez. 2020. This dataset does not exist: Training models from generated images. In *ICASSP 2020 - 2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 1–5.

Ondrej Bojar, Christian Buck, Christian Federmann, Barry Haddow, Philipp Koehn, Johannes Leveling, Christof Monz, Pavel Pecina, Matt Post, Herve Saint-Amand, Radu Soricut, Lucia Specia, and Aleš Tamchyna. 2014. Findings of the 2014 workshop on statistical machine translation. In *Proceedings of the Ninth Workshop on Statistical Machine Translation*, pages 12–58, Baltimore, Maryland, USA. Association for Computational Linguistics.

Leo Breiman. 2001. Random forests. *Machine learning*, 45(1):5–32.

Cristian Buciluǎ, Rich Caruana, and Alexandru Niculescu-Mizil. 2006. Model compression. In *Proceedings of KDD*, pages 535–541. ACM.

Kevin Clark, Minh-Thang Luong, Urvashi Khandelwal, Christopher D Manning, and Quoc Le. 2019. Bam! born-again multi-task networks for natural language understanding. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 5931–5937.

Hal Daumé III. 2007. Frustratingly easy domain adaptation. In *Proceedings of the 45th Annual Meeting of the Association of Computational Linguistics*, pages 256–263, Prague, Czech Republic. Association for Computational Linguistics.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. Bert: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186.

Nikita Dvornik, Cordelia Schmid, and Julien Mairal. 2019. Diversity with cooperation: Ensemble methods for few-shot classification. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 3723–3731.

Tommaso Furlanello, Zachary C Lipton, Michael Tschannen, Laurent Itti, and Anima Anandkumar. 2018. Born again neural networks. *arXiv preprint arXiv:1805.04770*.

Yaroslav Ganin and Victor Lempitsky. 2015. Unsupervised domain adaptation by backpropagation. In *International conference on machine learning*, pages 1180–1189. PMLR.

Shangwei Guo, Chunlong Xie, Jiwei Li, Lingjuan Lyu, and Tianwei Zhang. 2022. Threats to pre-trained language models: Survey and taxonomy. *arXiv preprint arXiv:2202.06862*.

Xuanli He, Lingjuan Lyu, Qiongkai Xu, and Lichao Sun. 2021a. Model extraction and adversarial transferability, your bert is vulnerable! *arXiv preprint arXiv:2103.10013*.

Xuanli He, Qiongkai Xu, Lingjuan Lyu, Fangzhao Wu, and Chenguang Wang. 2021b. Protecting intellectual property of language generation apis with lexical watermark. *arXiv preprint arXiv:2112.02701*.

Geoffrey Hinton, Oriol Vinyals, and Jeff Dean. 2015. Distilling the knowledge in a neural network. *arXiv preprint arXiv:1503.02531*.

Kalpesh Krishna, Gaurav Singh Tomar, Ankur P Parikh, Nicolas Papernot, and Mohit Iyyer. 2019. Thieves on sesame street! model extraction of bert-based apis. In *International Conference on Learning Representations*.

Varun Kumar, Ashutosh Choudhary, and Eunah Cho. 2020. Data augmentation using pre-trained transformer models. *arXiv preprint arXiv:2003.02245*.

Yinhan Liu, Jiatao Gu, Naman Goyal, Xian Li, Sergey Edunov, Marjan Ghazvininejad, Mike Lewis, and Luke Zettlemoyer. 2020. Multilingual denoising pre-training for neural machine translation. *arXiv preprint arXiv:2001.08210*.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Roberta: A robustly optimized bert pretraining approach.

Lingjuan Lyu, Xuanli He, and Yitong Li. 2020. Differentially private representation for nlp: Formal guarantee and an empirical study on privacy and fairness. In *EMNLP 2020, Findings*.

Andrew L. Maas, Raymond E. Daly, Peter T. Pham, Dan Huang, Andrew Y. Ng, and Christopher Potts. 2011. Learning word vectors for sentiment analysis. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, pages 142–150, Portland, Oregon, USA. Association for Computational Linguistics.

Pekka Malo, Ankur Sinha, Pekka Korhonen, Jyrki Wallenius, and Pyry Takala. 2014. Good debt or bad debt: Detecting semantic orientations in economic texts. *Journal of the Association for Information Science and Technology*, 65(4):782–796.

Bryan McCann, James Bradbury, Caiming Xiong, and Richard Socher. 2017. Learned in translation: contextualized word vectors. In *Proceedings of the 31st International Conference on Neural Information Processing Systems*, pages 6297–6308.

Timothy Miller. 2019. Simplified neural unsupervised domain adaptation. In *Proceedings of the conference. Association for Computational Linguistics. North American Chapter. Meeting*, volume 2019, page 414. NIH Public Access.

David Opitz and Richard Maclin. 1999. Popular ensemble methods: An empirical study. *Journal of artificial intelligence research*, 11:169–198.

Tribhuvanesh Orekondy, Bernt Schiele, and Mario Fritz. 2019. Knockoff nets: Stealing functionality of black-box models. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 4954–4963.

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. Bleu: a method for automatic evaluation of machine translation. In *Proceedings of the 40th annual meeting of the Association for Computational Linguistics*, pages 311–318.

Suman Ravuri and Oriol Vinyals. 2019. Classification accuracy score for conditional generative models. In *Advances in Neural Information Processing Systems*, pages 12268–12279.

Fahimeh Saleh, Wray Buntine, and Gholamreza Haffari. 2020. Collective wisdom: Improving low-resource neural machine translation using adaptive knowledge distillation. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 3413–3421.

Hinrich Schütze, Christopher D Manning, and Prabhakar Raghavan. 2008. *Introduction to information retrieval*, volume 39. Cambridge University Press Cambridge.

Rico Sennrich, Barry Haddow, and Alexandra Birch. 2016. Neural machine translation of rare words with subword units. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1715–1725.

Richard Socher, Alex Perelygin, Jean Wu, Jason Chuang, Christopher D Manning, Andrew Y Ng, and Christopher Potts. 2013. Recursive deep models for semantic compositionality over a sentiment treebank. In *Proceedings of the 2013 conference on empirical methods in natural language processing*, pages 1631–1642.

Xu Tan, Yi Ren, Di He, Tao Qin, Zhou Zhao, and Tie-Yan Liu. 2019. Multilingual neural machine translation with knowledge distillation. *arXiv preprint arXiv:1902.10461*.

Raphael Tang, Yao Lu, Linqing Liu, Lili Mou, Olga Vechtomova, and Jimmy Lin. 2019. Distilling task-specific knowledge from bert into simple neural networks. *arXiv preprint arXiv:1903.12136*.

Jörg Tiedemann. 2012. Parallel data, tools and interfaces in opus. In *Proceedings of the Eighth International Conference on Language Resources and Evaluation (LREC'12)*, pages 2214–2218.

Florian Tramèr, Fan Zhang, Ari Juels, Michael K Reiter, and Thomas Ristenpart. 2016. Stealing machine learning models via prediction apis. In *25th {USENIX} Security Symposium ({USENIX} Security 16)*, pages 601–618.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in neural information processing systems*, pages 5998–6008.

Eric Wallace, Mitchell Stern, and Dawn Song. 2020. Imitation attacks and defenses for black-box machine translation systems. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 5531–5546.

Jindong Wang, Cuiling Lan, Chang Liu, Yidong Ouyang, and Tao Qin. 2021. Generalizing to unseen domains: A survey on domain generalization. *arXiv preprint arXiv:2103.03097*.

Qizhe Xie, Minh-Thang Luong, Eduard Hovy, and Quoc V Le. 2020. Self-training with noisy student improves imagenet classification. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 10687–10698.

Peng Zhou, Zhenyu Qi, Suncong Zheng, Jiaming Xu, Hongyun Bao, and Bo Xu. 2016. Text classification improved by integrating bidirectional lstm with two-dimensional max pooling. In *Proceedings of COLING 2016, the 26th International Conference on Computational Linguistics: Technical Papers*, pages 3485–3495.

A Hyper-Parameter Settings

In order to have a fair and consistent comparison of experiments, we utilize the same hyper-parameters for the same task, as demonstrated in Table 6. They are decided by our preliminary experiments on target domains. All experiments could be conducted on a single RTX 6000 GPU.

	Sent.	MT
Learning rate	1e-05	5e-04
Batch size	16 sentences	32k tokens
Optimizer	Adam	Adam
Epoch	50	40
Max length	256	1024
Warm-up	-	4000 steps

Table 6: Hyper-parameter used for sentiment analysis (Sent.) and machine translation (MT).

B Comparison of Ensemble Strategy

In this section, we compare two ensemble methods for sentiment classification, i) concatenating the training samples (Concat.) and ii) averaging the prediction scores (Avg.). Both ensemble strategies are competitive to the other, as demonstrated in Table 7. As we are not able to acquire scores of each token from MT APIs, we cannot average the results of MT. In our paper, to have a consistent comparison, Concate. is used in all our ensemble experiments on both classification and translation tasks.

Method	Ensemble	Accuracy
BERT+RoBERTa	Concat.	91.57 ± 0.27
BERT+RoBERTa	Avg.	91.58 ± 0.39
Google+IBM	Concat.	89.75 ± 0.58
Google+IBM	Avg.	89.76 ± 0.42

(a) Ensemble results on SST.

Method	Ensemble	Accuracy
BERT+RoBERTa	Concat.	90.53 ± 0.38
BERT+RoBERTa	Avg.	90.89 ± 0.48
Google+IBM	Concat.	89.82 ± 0.81
Google+IBM	Avg.	88.86 ± 1.22

(b) Ensemble results on FST.

Table 7: The comparison of imitation attack on multiple victims using concatenate samples (Concat.) and average scores (Avg.).

C Comparison of Defense Strategies

In this section, we discuss the perturbation methods. Given an input sentence x, the probability score by the victim model is $h_v(x)$. To compare the influence of API performance on attack model, we sample a noise vector $\boldsymbol{n}$ from Gaussian distribution with a variance of σ, *i.e.*, $\boldsymbol{n} \sim \mathcal{N}(0, \sigma^2)$. The perturbed prediction function is calculated as:

$$h'_v(x) = h_v(x) + \boldsymbol{n}$$

It is worth noting that original victim model prediction is $y = \text{argmax}(h_v(x))$, therefore, injecting $\boldsymbol{n}$ could lead to different prediction $y' = \text{argmax}(h'_v(x))$. Consequently, the utility of victim model can be corrupted as demonstrated in Table 5. However, such compromise can cause finance and reputation losses to the API providers in real-world. To avoid these adverse effects, API providers can adopt a label-preserved policy, where the injected noise $\boldsymbol{n}$ should not flip the originally predicted labels. In other words, another noise will be sampled, if currently sampled noise changes the prediction of the original model. The results of such defense strategy is shown in Table 8. As this setting is more conservative, the performance of this defense is in between the performance of hard label and soft label.

	SST					FST				
Model	Soft	Hard	P 0.1	P 0.2	P 0.5	Soft	Hard	P 0.1	P 0.2	P 0.5
Victim 1			87.92					74.94		
Victim 2			89.40					80.00		
Attack$_{s1}$	90.44	88.58	90.48	89.96	89.35	82.03	80.25	82.84	81.92	81.72
Attack$_{s2}$	90.12	90.12	90.30	90.02	89.23	87.85	85.57	88.10	87.29	84.41
Attack$_{m}$	91.82	90.66	91.24	91.38	91.11	88.86	87.09	89.37	88.25	87.59

Table 8: The comparison of imitation attack results given victims with various defense strategies, soft label (Soft) to hard label (Hard) and noise perturbation (P) with variance (σ). The predicted labels of victim models are not flipped in the experiment.

Combining Compressions for Multiplicative Size Scaling on Natural Language Tasks

Rajiv Movva[*123], **Jinhao Lei**[*2], **Shayne Longpre**[23], **Ajay Gupta**[2], and **Chris DuBois**[2]

[*]Equal Contribution
[1]Cornell Tech, rm868@cornell.edu
[2]Apple, {jlei2, ajay_gupta2, cdubois}@apple.com
[3]Massachusetts Institute of Technology, slongpre@mit.edu

Abstract

Quantization, knowledge distillation, and magnitude pruning are among the most popular methods for neural network compression in NLP. Independently, these methods reduce model size and can accelerate inference, but their relative benefit and combinatorial interactions have not been rigorously studied. For each of the eight possible subsets of these techniques, we compare accuracy vs. model size tradeoffs across six BERT architecture sizes and eight GLUE tasks. We find that quantization and distillation consistently provide greater benefit than pruning. Surprisingly, except for the pair of pruning and quantization, using multiple methods together rarely yields diminishing returns. Instead, we observe complementary and super-multiplicative reductions to model size. Our work quantitatively demonstrates that combining compression methods can synergistically reduce model size, and that practitioners should prioritize (1) quantization, (2) knowledge distillation, and (3) pruning to maximize accuracy vs. model size tradeoffs.

1 Introduction

As increasingly large models dominate Natural Language Processing (NLP) benchmarks, model compression techniques have grown in popularity (Gupta and Agrawal, 2020; Rogers et al., 2020; Ganesh et al., 2021). For example, quantization (Shen et al., 2020; Zafrir et al., 2019; Jacob et al., 2018) lowers bit precision of network weights to reduce memory usage and accelerate inference (Krashinsky et al., 2020). Knowledge distillation (KD; Hinton et al. (2015)), which trains a student neural network using the logits (or representations) of a teacher network, is used widely to transfer knowledge to smaller models (Sanh et al., 2019; Jiao et al., 2020; Sun et al., 2019, 2020). Pruning identifies weights which can be omitted at test time without significantly degrading performance. Some pruning methods remove individual weights according to magnitudes or other heuristics (Gordon et al., 2020; Chen et al., 2020; Sanh et al., 2020), while others remove structured blocks of weights or entire attention heads (Wang et al., 2020; Hou et al., 2020; Voita et al., 2019; Michel et al., 2019).

Recent work has begun combining these compression methods for improved results. Sanh et al. (2020), Zhang et al. (2020), and Bai et al. (2021) have used knowledge distillation with pruning or low-bit quantization to fine-tune BERT. As practitioners look to combine methods more generally, new research is needed to compare their empirical value and study interactions. This work addresses the questions: (1) Which popular compression methods or combinations of methods are usually most effective? (2) When combining methods, are their benefits complementary or diminishing?

We address these questions by computing accuracy vs. model size tradeoff curves for six pretrained BERT sizes fine-tuned on eight GLUE tasks (Wang et al., 2019b), applying each of eight possible subsets of quantization-aware-training (QAT), knowledge distillation (KD), and magnitude pruning (MP). Our main findings are as follows:

1. When methods are applied independently, QAT yields best accuracy-compression tradeoffs, followed by KD and then MP.
2. Strikingly, we observe no diminishing returns when combining KD with QAT or MP. Instead, KD mitigates the loss in accuracy caused by either method, thereby super-multiplicatively reducing model size.
3. When used together, QAT and MP amplify each other's individual accuracy losses. However, combining all three methods (*i.e.*, also using KD) preserves accuracy, allowing 18x and 11x compression for BERT-LARGE and BASE respectively.

Proceedings of the 29th International Conference on Computational Linguistics, pages 2861–2872
October 12–17, 2022.

2 Methods

In our work, we study three common model compressions: quantization-aware-training, knowledge distillation, and magnitude pruning. We prioritize performant, broadly applicable approaches with accessible implementations, so our findings are most useful to practictioners. Hyperparameters and additional method details are in Appendices A & C.

BERT Architecture Sizes. We test each compression combination across six different BERT architecture sizes, seeing as they may have different compressibilities. These pretrained models are taken from Turc et al. (2019): LARGE (367 million params), BASE (134M), MEDIUM (57M), SMALL (45M), MINI (19M), TINY (8M). Including a range of sizes makes our findings relevant to practitioners or deployment settings without resources for architectures like LARGE or BASE. Additionally, using smaller model sizes is a practical baseline to compare our compression methods against.

Quantization-Aware-Training (QAT). While most neural networks use 32-bit floats for weights and activations, recent work has shown promise for lower precisions. 16-bit floats cause no accuracy loss for most architectures (Das et al., 2018), and Zafrir et al. (2019) show that, with quantization-aware-training (QAT), 8-bit integer (INT8) BERT mostly preserves GLUE accuracy. The INT8 model is nearly 4x smaller, and can achieve 2.4-4.0x inference acceleration with appropriate hardware (Kim et al., 2021a). As lower precisions harm accuracy significantly (Shen et al., 2020), we use an 8-bit BERT with the QAT scheme described by Zafrir et al. (2019), recapped in the Appendix.

Knowledge Distillation (KD). In KD, we fine-tune a small student model by optimizing its weights to mimic the outputs of a teacher model. We use a common, simple variant of KD, emulating Turc et al. (2019): we use a BERT-LARGE fine-tuned for three epochs on the GLUE task as the teacher, and the student is trained to minimize KL-divergence between its predicted probabilities and the teacher's.

To further improve the utility of KD, we adopt Jiao et al. (2020)'s approach of data augmentation (DA) for GLUE training datasets. This technique helps for all tasks, especially the smaller ones (*e.g.* MRPC, RTE). Each example is copied 10, 20, or 30 times (more copies for smaller tasks), and each copy has some of its words replaced with synonyms (*i.e.* words with closest GLoVe embeddings). Many of the copies have altered meanings, but the teacher is able to adapt by making different predictions. Before running all of our experiments, we ran a few trials (on MRPC and QNLI) to confirm that DA helped with distillation but not without, and so we only used DA for the KD experiments.

Magnitude Pruning (MP). Several pruning methods have been used in NLP (Hoefler et al., 2021). We use unstructured weight pruning, which can achieve higher sparsities than structured pruning (Renda et al., 2020), and has comparatively standard implementations.

Magnitude pruning masks the weights with lowest magnitudes to achieve a target sparsity. As in Sanh et al. (2020), we iteratively prune weights at a linear schedule during training after some warmup steps. Sanh et al. (2020) also propose *movement pruning*, in which weights are pruned according to their gradients during fine-tuning. We found that movement pruning performs worse than magnitude at moderate sparsities (40-60%), when accuracy is retained (corroborated by Sanh et al. (2020)). As we target accuracy-preserving pruning, we use magnitude pruning for the experiments in this work. We prune either 40% or 60% of encoder weights only, as pruning embedding weights significantly damages accuracy (Yu et al., 2020).

3 Results

Experiments For six BERT architecture sizes and eight GLUE tasks[1], we tested every possible subset of compression methods: no compression (Baseline), QAT, KD, MP, QAT+KD, QAT+MP, KD+MP, and QAT+KD+MP. For each of 576 experiment settings, we log the max GLUE development set accuracy across twelve hyperparameter configurations and five repetitions of each configuration. In Figure 1, we plot mean GLUE accuracy across all eight tasks on the y-axis against decreasing model size on the x-axis, for each compression combination. The curves without pruning include six points, one for each architecture size from LARGE to TINY. With pruning, there are twice as many points, as each architecture is pruned to either 40% or 60% encoder sparsity. Results split by task are available in Appendix Figure A1.

Individually, QAT and KD are most effective. For all architectures, QAT (blue) reduces model

[1] We excluded CoLA and WNLI to reduce experimental burden and due to issues with WNLI (Wang et al., 2019a).

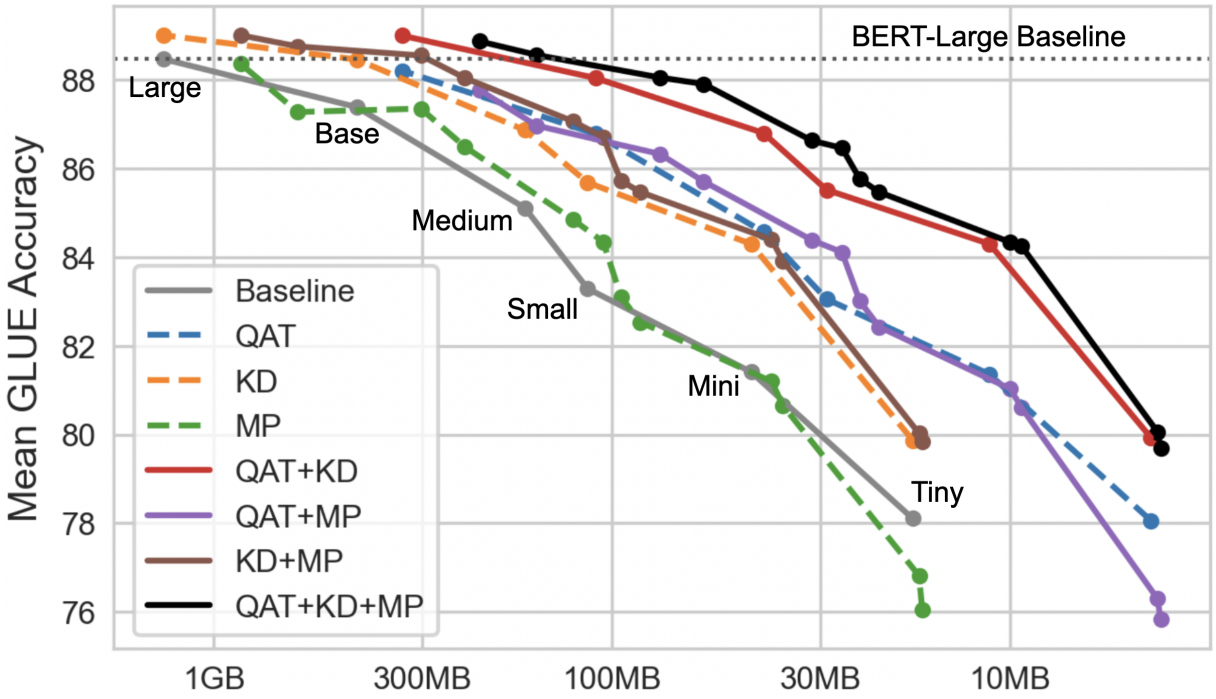

Figure 1: Mean GLUE accuracy vs. **decreasing** model size, with curves plotted for each compression combination. The different points for each curve represent the different BERT architecture sizes, from LARGE down to TINY.

size by 4× while minimally reducing accuracy: the largest drop is −0.6% for BERT-BASE (supporting Zafrir et al. (2019)), while other architecture sizes are nearly unaffected. KD (orange) does not reduce a given architecture's size, but instead yields a consistent boost to accuracy, especially for smaller architectures. This upward shift on the accuracy-model size curve means that larger models can be downsized more effectively: *e.g.*, LARGE's baseline accuracy is matched by the KD version of BASE, and KD SMALL outperforms baseline MEDIUM. Compared to QAT or KD, MP (green) is only modestly helpful. Typically, 40% of encoder weights can be removed without much impact, but pruning 60% (*i.e.*, the second point in each set of two) degrades accuracy. Also, removing 40% of encoder weights corresponds to a $< 40\%$ model size reduction because we do not prune embedding weights. Therefore, while MP can be helpful for LARGE and BASE, it cannot significantly compress small architectures, which have a higher percentage of their weights in embeddings.

Used together, KD mitigates accuracy losses from both QAT and MP. Moving to combinations of compression methods, we find that the most successful pair combines QAT and KD (red), which yields QAT's 4× memory reduction while retaining the improved accuracy over baseline from KD. Meanwhile, QAT+MP (purple) does poorly: though LARGE and BASE can prune 40% of weights while retaining accuracy, when they are pruned *and* quantized, they have lower accuracy than when they are only quantized. This result suggests that pruning specifically damages accuracy with quantization: practitioners should expect additive (or worse) accuracy degradation when combining QAT and MP. On the other hand, with KD+MP (brown), 40% weights can be pruned while retaining the accuracy boost from KD, for all architectures. Thus, KD *mitigates* accuracy losses from both MP and QAT. This result still holds when we combine all three methods (black). With QAT+KD+MP, 40% of encoder weights for LARGE and 60% for BASE (and smaller) can be removed while matching the accuracy of QAT+KD. In our experiments, KD completely mitigates the compounding losses from QAT+MP and even *improves* accuracy. KD enables deeper compression when practitioners combine methods.

Combining methods yields *super-multiplicative* compression ratios. Building on our qualitative findings, we were interested in quantitative estimates for how much each method allows us to compress each architecture size. So, we compute *compression ratios*, *i.e.*, the maximal size reduction factor possible while preserving accuracy to within 0.5% of baseline. For example, baseline LARGE (1341 MB) yields 92.8% accuracy on QNLI, while

	Size	QAT	KD	MP	QAT+KD	QAT+MP	KD+MP	QAT+KD+MP
LARGE	1341	3.6×	3.4×	1.7×	12.6×	3.9×	5.8×	18.1×
BASE	438	2.6×	1.8×	1.7×	5.9×	3.1×	2.8×	11.1×
MEDIUM	166	2.9×	2.0×	1.2×	8.0×	4.0×	3.5×	14.1×
SMALL	115	3.6×	2.4×	1.2×	9.4×	4.7×	3.6×	14.1×
MINI	45	3.7×	1.2×	1.1×	4.7×	3.4×	1.8×	7.0×
TINY	18	3.4×	1.0×	0.7×	4.0×	2.2×	1.0×	3.9×

Table 1: Ratios, averaged across all GLUE tasks, measuring the maximum possible size reduction factor of a certain architecture while within 0.5% of baseline accuracy. Uncompressed sizes are listed in **megabytes (MB)**.

BASE with QAT, KD, and 40% MP (76 MB) is the smallest model within 0.5% of that, at 92.6% accuracy. Therefore, BERT-LARGE's compression ratio on QNLI is $\frac{1341}{76} = 17.6\times$, and averaging this value across tasks yields a net ratio of 18.1× (top-right, Table 1). We similarly compute ratios for all architectures and compression combinations.

As before, QAT usually retains accuracy and yields a 4× size reduction. However, because there are a few tasks for which QAT causes a $> 0.5\%$ accuracy drop[2], the task-averaged compression ratios for LARGE and BASE of 3.6× and 2.6×. KD also has high mean compression ratios, because it often boosts small architectures' accuracies to match larger baseline architectures. MP yields 1.7× compression (from 40% pruning) for LARGE and BASE, and even less for the smaller architectures.

When combined, we observe synergistic compression between QAT and KD. We might expect strong diminishing returns from combining methods, but even in their absence, we would expect independent compression ratios to multiply. Strikingly, though, we often see *super-multiplicative* model size reductions with QAT+KD: *e.g.* BASE, $5.9 > 2.6 \cdot 1.8 = 4.7$; MEDIUM: $8.0 > 2.9 \cdot 2.0 = 5.8$. For KD+MP, the ratios multiply for LARGE and BASE. Also, while pruning was originally ineffective for MEDIUM and smaller, KD+MP appears to make pruning more effective, again with super-multiplicative compression (e.g. for Medium: $3.5 > 2.0 \cdot 1.2 = 2.4$). These findings show that KD mitigates the drop in accuracy from quantization (Zhang et al., 2020) and pruning (Sanh et al., 2020), supporting qualitative findings from prior literature. This mitigation effect, combined with the accuracy boost that KD provides, yields joint compressions that are more effective than the sum of their parts. Remarkably, with all three compressions, we still see super-multiplicative scaling for BASE and smaller (*e.g.*, BASE: $11.1 > 2.6 \cdot 1.8 \cdot 1.7 = 8.0$; MEDIUM: $14.1 > 2.9 \cdot 2.0 \cdot 1.2 = 7.0$). At 18.1×, LARGE is most compressible, but actually has a sub-multiplicative ratio, perhaps because the individual compressions already work very well.

[2] Those tasks would be considered to have 1× compression.

4 Discussion and Future Work

We examine the relative benefits and interactions of three widely-used compression methods in NLP: knowledge distillation, INT8 quantization-aware-training, and magnitude pruning. Though multiple techniques are increasingly used simultaneously, little work has studied the empirical interactions between them.

Across six architecture sizes and eight GLUE tasks, we find that INT8 quantization is most beneficial, and combining quantization with KD mitigates any of its occasional accuracy drops. Quantization and pruning yield diminishing returns, with the two methods exacerbating accuracy losses when used together. However, when all three methods are combined, KD restores accuracy above baseline, yielding 18× and 11× net compression for BERT-LARGE and BASE. We quantitatively confirm that KD's model size improvements are complementary (often super-multiplicative) with quantization and pruning. Given these benefits, using larger, more recent GLUE winners as KD teachers may yield further gains (*e.g.* ERNIE, Sun et al. (2021)). We also hope that our observed benefits inspire authors of new compression techniques to evaluate complementarity with existing methods.

In future work, we hope to measure accuracy vs. inference time tradeoffs, and compare these results to our findings on model size. Though there is work on accelerated inference with INT8 quantization (Kim et al., 2021b) or pruning (Pool et al., 2021), it is not clear how these speedups stack. Profiling compression combinations on specialized hardware would be an informative avenue to explore next.

Broader Impacts

As NLP models dramatically scale in size, they rely increasingly on specialized hardware (*e.g.* GPUs, TPUs) to train and deploy them. The manufacturing and energy consumption involved in the usage of such devices imposes a significant carbon footprint (Strubell et al., 2019; Gupta et al., 2021). Model compression is part of the broader movement towards "Green AI", in which researchers develop more energy-efficient models with similar task accuracies in order to reduce usage of compute-hungry hardware (Schwartz et al., 2020).

By careful empirical study of how to optimally combine compression methods, we believe our work takes further steps towards Green AI. In particular, rather than proposing a single architecture that achieves a specific tradeoff, we equip practitioners with a set of principles to apply depending on their needs, hopefully increasing uptake of at least some subset of compression methods. Applying our insights to widely-used deployment settings (speech-to-text, search, *etc.*) could significantly reduce AI's consumptive footprint.

Though model compression methods can dramatically mitigate deep learning's carbon footprint, they may also create opportunities for harm (Suresh and Guttag, 2021). Specifically, recent work shows that pruning damages accuracy for minority classes in the training dataset (Hooker et al., 2020), and that pruning may change model behavior even when accuracy is preserved (Movva and Zhao, 2020). Such predictive disparities can lead to algorithmic harms: *e.g., representational* harms for language models, or *allocational* harms for certain downstream task predictors (Blodgett et al., 2020). More work is needed to systematically characterize the relationship between compression and algorithmic harms.

References

Haoli Bai, Wei Zhang, Lu Hou, Lifeng Shang, Jin Jin, Xin Jiang, Qun Liu, Michael Lyu, and Irwin King. 2021. BinaryBERT: Pushing the limit of BERT quantization. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 4334–4348, Online. Association for Computational Linguistics.

Su Lin Blodgett, Solon Barocas, Hal Daumé III, and Hanna Wallach. 2020. Language (Technology) is Power: A Critical Survey of "Bias" in NLP. *arXiv:2005.14050 [cs]*. ArXiv: 2005.14050.

Tianlong Chen, Jonathan Frankle, Shiyu Chang, Sijia Liu, Yang Zhang, Zhangyang Wang, and Michael Carbin. 2020. The lottery ticket hypothesis for pre-trained bert networks. In *Advances in Neural Information Processing Systems*, volume 33, pages 15834–15846. Curran Associates, Inc.

Dipankar Das, Naveen Mellempudi, Dheevatsa Mudigere, Dhiraj D. Kalamkar, Sasikanth Avancha, Kunal Banerjee, Srinivas Sridharan, Karthik Vaidyanathan, Bharat Kaul, Evangelos Georganas, Alexander Heinecke, Pradeep Dubey, Jesús Corbal, Nikita Shustrov, Roman Dubtsov, Evarist Fomenko, and Vadim O. Pirogov. 2018. Mixed precision training of convolutional neural networks using integer operations. *CoRR*, abs/1802.00930.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. BERT: pre-training of deep bidirectional transformers for language understanding. *CoRR*, abs/1810.04805.

Prakhar Ganesh, Yao Chen, Xin Lou, Mohammad Ali Khan, Yin Yang, Hassan Sajjad, Preslav Nakov, Deming Chen, and Marianne Winslett. 2021. Compressing large-scale transformer-based models: A case study on bert. *Transactions of the Association for Computational Linguistics*, 9:1061–1080.

Mitchell A. Gordon, Kevin Duh, and Nicholas Andrews. 2020. Compressing BERT: Studying the Effects of Weight Pruning on Transfer Learning. *arXiv:2002.08307 [cs]*. ArXiv: 2002.08307.

Manish Gupta and Puneet Agrawal. 2020. Compression of deep learning models for text: A survey. *arXiv preprint arXiv:2008.05221*.

Udit Gupta, Young Geun Kim, Sylvia Lee, Jordan Tse, Hsien-Hsin S. Lee, Gu-Yeon Wei, David Brooks, and Carole-Jean Wu. 2021. Chasing Carbon: The Elusive Environmental Footprint of Computing. In *2021 IEEE International Symposium on High-Performance Computer Architecture (HPCA)*, pages 854–867. ISSN: 2378-203X.

Geoffrey Hinton, Oriol Vinyals, and Jeff Dean. 2015. Distilling the knowledge in a neural network. *arXiv preprint arXiv:1503.02531*.

Torsten Hoefler, Dan Alistarh, Tal Ben-Nun, Nikoli Dryden, and Alexandra Peste. 2021. Sparsity in Deep Learning: Pruning and growth for efficient inference and training in neural networks. *arXiv:2102.00554 [cs]*. ArXiv: 2102.00554.

Sara Hooker, Nyalleng Moorosi, Gregory Clark, Samy Bengio, and Emily Denton. 2020. Characterising Bias in Compressed Models. *arXiv:2010.03058 [cs]*. ArXiv: 2010.03058.

Lu Hou, Zhiqi Huang, Lifeng Shang, Xin Jiang, Xiao Chen, and Qun Liu. 2020. Dynabert: Dynamic bert with adaptive width and depth. In *Advances in Neural Information Processing Systems*, volume 33, pages 9782–9793. Curran Associates, Inc.

Benoit Jacob, Skirmantas Kligys, Bo Chen, Menglong Zhu, Matthew Tang, Andrew Howard, Hartwig Adam, and Dmitry Kalenichenko. 2017. Quantization and Training of Neural Networks for Efficient Integer-Arithmetic-Only Inference. *arXiv:1712.05877 [cs, stat]*. ArXiv: 1712.05877.

Benoit Jacob, Skirmantas Kligys, Bo Chen, Menglong Zhu, Matthew Tang, Andrew Howard, Hartwig Adam, and Dmitry Kalenichenko. 2018. Quantization and training of neural networks for efficient integer-arithmetic-only inference. In *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 2704–2713. IEEE Computer Society.

Xiaoqi Jiao, Yichun Yin, Lifeng Shang, Xin Jiang, Xiao Chen, Linlin Li, Fang Wang, and Qun Liu. 2020. Tinybert: Distilling bert for natural language understanding. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: Findings*, pages 4163–4174.

Sehoon Kim, Amir Gholami, Zhewei Yao, Michael W. Mahoney, and Kurt Keutzer. 2021a. I-BERT: Integer-only BERT Quantization. *arXiv:2101.01321 [cs]*. ArXiv: 2101.01321.

Sehoon Kim, Amir Gholami, Zhewei Yao, Michael W Mahoney, and Kurt Keutzer. 2021b. I-bert: Integer-only bert quantization. *arXiv preprint arXiv:2101.01321*.

Ronny Krashinsky, Oliver Giroux, Stephen Jones, Nick Stam, and Sridhar Ramaswamy. 2020. NVIDIA Ampere Architecture In-Depth.

Paul Michel, Omer Levy, and Graham Neubig. 2019. Are Sixteen Heads Really Better than One? *arXiv:1905.10650 [cs]*. ArXiv: 1905.10650.

Rajiv Movva and Jason Y. Zhao. 2020. Dissecting Lottery Ticket Transformers: Structural and Behavioral Study of Sparse Neural Machine Translation. *arXiv:2009.13270 [cs, stat]*. ArXiv: 2009.13270.

Jeffrey Pennington, Richard Socher, and Christopher Manning. 2014. GloVe: Global vectors for word representation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1532–1543, Doha, Qatar. Association for Computational Linguistics.

Jeff Pool, Abhishek Sawarkar, and Jay Rodge. 2021. Accelerating inference with sparsity using the nvidia ampere architecture and nvidia tensorrt. *NVIDIA Developer Blog*.

Alex Renda, Jonathan Frankle, and Michael Carbin. 2020. Comparing Rewinding and Fine-tuning in Neural Network Pruning. *arXiv:2003.02389 [cs, stat]*. ArXiv: 2003.02389.

Anna Rogers, Olga Kovaleva, and Anna Rumshisky. 2020. A primer in bertology: What we know about how bert works. *Transactions of the Association for Computational Linguistics*, 8:842–866.

Victor Sanh, Lysandre Debut, Julien Chaumond, and Thomas Wolf. 2019. Distilbert, a distilled version of bert: smaller, faster, cheaper and lighter. *arXiv preprint arXiv:1910.01108*.

Victor Sanh, Thomas Wolf, and Alexander Rush. 2020. Movement pruning: Adaptive sparsity by fine-tuning. In *Advances in Neural Information Processing Systems*, volume 33, pages 20378–20389.

Roy Schwartz, Jesse Dodge, Noah A. Smith, and Oren Etzioni. 2020. Green AI. *Communications of the ACM*, 63(12):54–63.

Sheng Shen, Zhen Dong, Jiayu Ye, Linjian Ma, Zhewei Yao, Amir Gholami, Michael W Mahoney, and Kurt Keutzer. 2020. Q-bert: Hessian based ultra low precision quantization of bert. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pages 8815–8821.

Emma Strubell, Ananya Ganesh, and Andrew McCallum. 2019. Energy and Policy Considerations for Deep Learning in NLP. *arXiv:1906.02243 [cs]*. ArXiv: 1906.02243.

Siqi Sun, Yu Cheng, Zhe Gan, and Jingjing Liu. 2019. Patient knowledge distillation for bert model compression. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 4323–4332.

Yu Sun, Shuohuan Wang, Shikun Feng, Siyu Ding, Chao Pang, Junyuan Shang, Jiaxiang Liu, Xuyi Chen, Yanbin Zhao, Yuxiang Lu, Weixin Liu, Zhihua Wu, Weibao Gong, Jianzhong Liang, Zhizhou Shang, Peng Sun, Wei Liu, Xuan Ouyang, Dianhai Yu, Hao Tian, Hua Wu, and Haifeng Wang. 2021. ERNIE 3.0: Large-scale Knowledge Enhanced Pre-training for Language Understanding and Generation. *arXiv:2107.02137 [cs]*. ArXiv: 2107.02137.

Zhiqing Sun, Hongkun Yu, Xiaodan Song, Renjie Liu, Yiming Yang, and Denny Zhou. 2020. MobileBERT: a compact task-agnostic BERT for resource-limited devices. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 2158–2170, Online. Association for Computational Linguistics.

Harini Suresh and John Guttag. 2021. A Framework for Understanding Sources of Harm throughout the Machine Learning Life Cycle. In *Equity and Access in Algorithms, Mechanisms, and Optimization*,

EAAMO '21, pages 1–9, New York, NY, USA. Association for Computing Machinery.

Iulia Turc, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. Well-Read Students Learn Better: On the Importance of Pre-training Compact Models. *arXiv:1908.08962 [cs]*. ArXiv: 1908.08962.

Elena Voita, David Talbot, Fedor Moiseev, Rico Sennrich, and Ivan Titov. 2019. Analyzing Multi-Head Self-Attention: Specialized Heads Do the Heavy Lifting, the Rest Can Be Pruned. *arXiv:1905.09418 [cs]*. ArXiv: 1905.09418.

Alex Wang, Jan Hula, Patrick Xia, Raghavendra Pappagari, R. Thomas McCoy, Roma Patel, Najoung Kim, Ian Tenney, Yinghui Huang, Katherin Yu, Shuning Jin, Berlin Chen, Benjamin Van Durme, Edouard Grave, Ellie Pavlick, and Samuel R. Bowman. 2019a. Can you tell me how to get past sesame street? sentence-level pretraining beyond language modeling. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 4465–4476, Florence, Italy. Association for Computational Linguistics.

Alex Wang, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel R. Bowman. 2019b. GLUE: A multi-task benchmark and analysis platform for natural language understanding. In the Proceedings of ICLR.

Ziheng Wang, Jeremy Wohlwend, and Tao Lei. 2020. Structured Pruning of Large Language Models. *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 6151–6162. ArXiv: 1910.04732.

Haonan Yu, Sergey Edunov, Yuandong Tian, and Ari S. Morcos. 2020. Playing the lottery with rewards and multiple languages: lottery tickets in RL and NLP. *arXiv:1906.02768 [cs, stat]*. ArXiv: 1906.02768.

Ofir Zafrir, Guy Boudoukh, Peter Izsak, and Moshe Wasserblat. 2019. Q8bert: Quantized 8bit bert. *arXiv preprint arXiv:1910.06188*.

Wei Zhang, Lu Hou, Yichun Yin, Lifeng Shang, Xiao Chen, Xin Jiang, and Qun Liu. 2020. Ternarybert: Distillation-aware ultra-low bit bert. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 509–521.

A Experiment Details

A.1 Experiment Breakdown

In total, we tested 576 experimental conditions, each of which involves fine-tuning a model on a GLUE task. We used eight GLUE tasks: SST-2, MRPC, STS-B, QQP, MNLI, MNLI-MM, QNLI, and RTE. We excluded CoLA and WNLI from the pruning experiments to reduce the computational burden (more on the compute budget below), and because there are some known issues with WNLI that make it difficult for a fair evaluation (Wang et al., 2019a).

We used six architecture sizes (details about the architectures are in Table 2), and eight subsets of compression methods (including the baseline of no compression). Note that, when we combine compression methods, there is no concept of order, because all methods function simultaneously and independently: QAT simply adds additional operations after each Linear layer, KD only modifies the loss, and MP gradually masks more weights. Therefore, these eight subsets are exhaustive.

There were twice as many pruning experiments as non-pruning experiments, since each pruning experiment tested two different sparsity levels[3]. So, there were 4 compression subsets without pruning, 4 compression subsets with 40% pruning, and 4 compression subsets with 60% pruning. Overall, there were $6 \cdot 8 \cdot (3 \cdot 4) = 576$ experimental conditions.

A.2 Training & Hyperparameters

For each experiment, we started by initializing the BERT architecture with pretrained weights from Turc et al. (2019). We fine-tuned for 3 epochs (on either the base or augmented GLUE training set, depending on whether we were performing a distillation experiment). As is standard for BERT fine-tuning on GLUE, batch size and LR can have a significant effect on the results (Devlin et al., 2018; Turc et al., 2019). For each experimental condition, we tested three batch sizes ({8, 16, 32}) and four learning rates ({1e-5, 2e-5, 3e-5, 4e-5} for LARGE and BASE; {3e-5, 5e-5, 0.0001, 0.0003} for MEDIUM/SMALL/MINI/TINY). For each hyperparameter combination, we performed five repetitions, so there were $3 \cdot 4 \cdot 5 = 60$ total training runs per experimental condition. We report max accuracy across these 60 runs rather than taking an average, as the BERT training on GLUE can be unstable and lead to poor results a high fraction of the time (Devlin et al., 2018). We use the public GLUE development sets rather than the official test sets, since it wouldn't have been feasible to make thousands of submissions to the GLUE testing portal.

Overall, then, we performed $576 \cdot 60 = 34560$ fine-tuning experiments. This was feasible because the smaller architectures could be fine-tuned quickly (from an hour for MEDIUM to a few minutes for TINY, on the largest GLUE tasks). We performed all experiments on NVIDIA V100 GPUs, and all told, we would estimate approximately 75K GPU hours were necessary for our experiments. As we set a rough budget of 100K GPU hours, this was the reason why we had to make decisions like excluding two GLUE tasks (CoLA, WNLI), not performing data augmentation for non-distillation experiments, and not performing pretraining distillation; any of these decisions would have ballooned our experimental burden. We recognize the privilege of having had access to as much GPU time as we did, and hope that other researchers can benefit from this thorough empirical analysis.

B Task-Specific Results

In Figure A1, we plot similar curves to Figure 1, split by each task. The data from Figure 1 are replicated in the top-left plot. Tasks are ordered by size (reading left-to-right and then top-to-bottom). Most tasks have concordant trends with the curves for average GLUE performance as discussed in the main text, but there are a couple exceptions.

First, for magnitude pruning, we find that some tasks experience *worse* accuracy-size tradeoffs than baseline when pruned to 40% of weights remaining; this was typically not the case when all task accuracies are averaged. Specifically, for some tasks (*e.g.* SST-2, STS-B, MRPC), pruning significantly degrades accuracy below baseline. Smaller BERT architectures are especially harmed, since they are already under-parameterized compared to BERT-LARGE and BASE. This finding agrees with Chen et al. (2020), who find that these particular GLUE tasks are least prunable while preserving accuracy (they also use magnitude pruning, but

[3] Actually, we tested three sparsities (also including 80%), but we only show experiments from 40% and 60% in the main text. This was because 80% sparse models generally performed poorly, and fell outside the accuracy-model size frontier, so they did not affect our results – which focused on the best possible tradeoffs for each method. We show these additional results in Appendix C.3.

Architecture	# Layers	Hidden Dim.	Params (Millions)	Size (MB)	Avg GLUE
LARGE	24	1024	367	1341	88.47
BASE	12	768	134	438	87.39
MEDIUM	8	512	57	166	85.11
SMALL	4	512	45	115	83.29
MINI	4	256	19	45	81.41
TINY	2	128	8	18	78.10

Table 2: Information on the different BERT architecture sizes we use in our experiments, with pretrained versions of each size downloaded from (Turc et al., 2019) (in accordance with their license). The "Avg GLUE" column is the mean GLUE accuracy across the eight tasks included in our experiments.

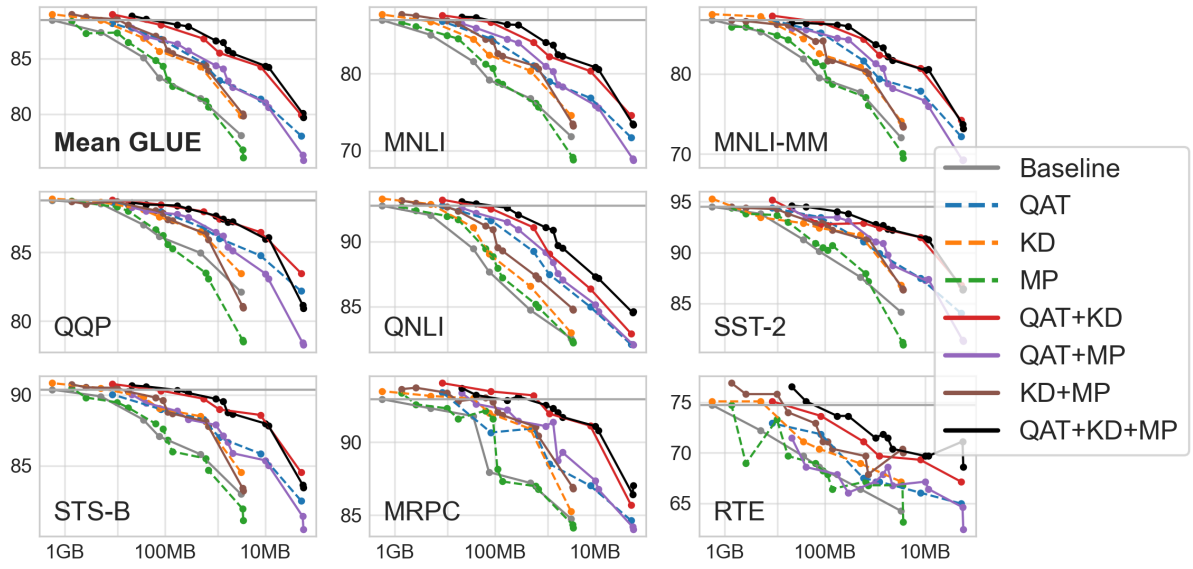

Figure A1: For each task, dev set accuracy vs. decreasing model size (from >1GB down to 10MB), with curves plotted for each compression combination. Tasks are ordered by training dataset size (left-to-right). The grey horizontal line in each plot is the baseline accuracy for BERT-LARGE.

a more compute-intensive version, allowing for slightly higher sparsities than we report).

Second, we find that for MRPC and RTE, the curves appear noisier, making some trends hard to discern. This is for two reasons: one, the tasks have the smallest training dataset, so they tend to degrade more in response to pruning (Chen et al., 2020). Second, we empirically found that these tasks varied more in their accuracy from run-to-run than other tasks (perhaps also because of their smaller training datasets). Thus, the true trends for different compression methods may be obscured by lower-confidence accuracy metrics.

C Implementation Details

C.1 Quantization-Aware-Training

In this work, we use quantization-aware-training (QAT) rather than naive post-training quantization (PTQ). PTQ quantizes weights after training and can significantly increase error due to the loss of precision. Recently, QAT has been more common, in which the effects of weight quantization are simulated during training with fake quantization operations (Jacob et al., 2017). Therefore, at inference time, the model's weights are better tailored to accommodate a reduced precision.

We specifically quantize the embedding and linear modules in our BERT architecture to use INT8 weights, following the symmetric linear quantization scheme from Q8BERT (Zafrir et al., 2019). The following quantization operation is applied to weights and activations, with scaling factor S and max value M, to quantize a value x:

$$\text{Quantize}(x \mid S, M) = \text{Clamp}(\lfloor x \cdot S \rceil, -M, M),$$

where $\lfloor \cdot \rceil$ is the integer rounding function, and

$\text{Clamp}(\cdot, -M, M)$ maps out-of-range values to $-M$ or M. M is determined by the number of bits; with 8 bits, for example, we have up to 256 possible quantization levels, so $M = 127$. Following Zafrir et al. (2019), the scaling factor S is set so that the largest possible value for a weight or activation matrix gets quantized to M. Thus, for a weight matrix W, the scale S^W is given by

$$S^W = \frac{M}{\max |W|}.$$

For activations x from a given layer L, the scale factor S^x is computed as an exponential moving average of the max activation value during training,

$$S^x = \frac{M}{\text{EMA}(\max_L |x|)}.$$

For quantization-aware-training, we add fake quantization ops to the model's weight matrices and activations during the training forward pass, therefore simulating the effect of quantization on each layer's output. However, $\text{Quantize}(\cdot)$ is not differentiable due to the rounding operation, so the backward pass simply ignores the quantization op using the straight-through-estimator: $\partial x^q / \partial x = \vec{1}$. We model our fake quantization ops off the implementation in Intel's `nlp-architect`[4] repository, authored by Zafrir et al. (2019) and others.

C.2 Knowledge Distillation

Knowledge Distillation (KD) aims to transfer the knowledge from a large teacher model into a smaller student model: ideally, our student's predictions will emulate the teacher's, but with reduced compute cost. Formally, models trained with KD learn to minimize $\mathcal{L}_{\text{KD}}$, the difference between the teacher's and student's functions f^T and f^S, across the training set $\mathcal{X}$:

$$\mathcal{L}_{\text{KD}} = \sum_{x \in \mathcal{X}} L\left(f^T(x), f^S(x)\right).$$

The functions $f^T(\cdot)$ and $f^S(\cdot)$ include the final output probabilities, and $L(\cdot)$ measures the cross entropy between the student and teacher predicted probabilities (Sanh et al., 2019).

While some approaches perform distillation during BERT pretraining, we only distill during task fine-tuning, which is also common. Focusing on fine-tuning was necessary to make our experimental search space tractable, since BERT pretraining can take multiple orders of magnitude more compute than fine-tuning. Task-specific distillation is also more critical to preserving accuracy than pretraining distillation (Jiao et al., 2020). We follow Jiao et al. (2020) in augmenting the GLUE datasets by copying examples and replacing words with synonyms. By running an ablation, they find that augmentation is useful for all tasks, and especially ones with less data (*i.e.,* CoLA and MRPC benefit much more than MNLI). They use different augmentation factors for each dataset, either scaling up the size by 10, 20, or 30 times. We use the same values in our work, copied here: {MNLI: 10, QQP: 10, QNLI: 20, SST-2: 20, STS-B: 30, MRPC: 30, RTE: 30}.

We directly used their script, `data_augmentation.py`, from the `TinyBERT`[5] Github repository. For each GLUE training dataset, this script generates an expanded file in the same format, except with multiple words replaced with synonyms (*i.e.*, L2 nearest neighbors from GLoVe embeddings (Pennington et al., 2014)). We then take our teacher model (fine-tuned BERT-LARGE on the same GLUE task) and generate predicted probabilities for every example in the augmented dataset, which includes the original sentences and their synonym-replacements. Note that, when using multiple synonyms, some of the sentences change meaning, leading to a substantially different prediction than the original sentence. This is another reason (in addition to little observable change in performance and our compute budget) that we did not use augmentation for the experiments which did not use distillation.

C.3 Pruning

We use the magnitude pruning setup from the `nn_pruning` Github repository[6] (Sanh et al., 2020). Importantly, because the `nn_pruning` implementation of a "pruned" model stores the weight values along with a dictionary of weights to be masked, the real model sizes on disk are not smaller. **So, the pruned model sizes that we report are theoretical rather than actual.** That said,

[4] `https://github.com/IntelLabs/nlp-architect`

[5] `https://github.com/yinmingjun/TinyBERT`; no license visible on Github.

[6] `https://github.com/huggingface/nn_pruning`; license allows commercial and private use.

it would be easy to attain a true size reduction if, for example, the weights were changed to zeroes and the model file was gzipped.

As in the movement pruning approach, we prune gradually throughout training. We prune to three possible sparsity levels: 40%, 60%, or 80% sparsity. Specifically, weights are masked on a linear schedule after 5000 warmup steps, meaning that a constant number of weights are masked at each step in order to reach the target sparsity by the end of training.

For most architectures, the 80% pruning results were very poor: they caused significant accuracy degradation, so they did not yield accuracy-model size tradeoffs that were on the optimal frontier. These results did not affect our conclusions, so we removed the 80% sparse points from Figure 1. We display these results here in the Appendix (Figure A2), by showing the same plots as Figure A1, but with the 80% sparse points added to the curves. There is often a steep accuracy dropoff from 60% to 80% sparsity, especially for the smaller tasks (STS-B, MRPC, RTE).

We acknowledge that there are some marginal improvements on magnitude pruning or other forms of weight pruning that may yield better results for some architectures or tasks. For example, Chen et al. (2020) use iterative magnitude pruning with multiple rounds of full training to achieve higher than 40% sparsities without accuracy loss. However, the goal of our work is not necessarily to achieve the largest model size reductions possible, but rather to understand how methods interact; therefore, we think our conclusions on magnitude pruning would hold even with slight modifications to the method.

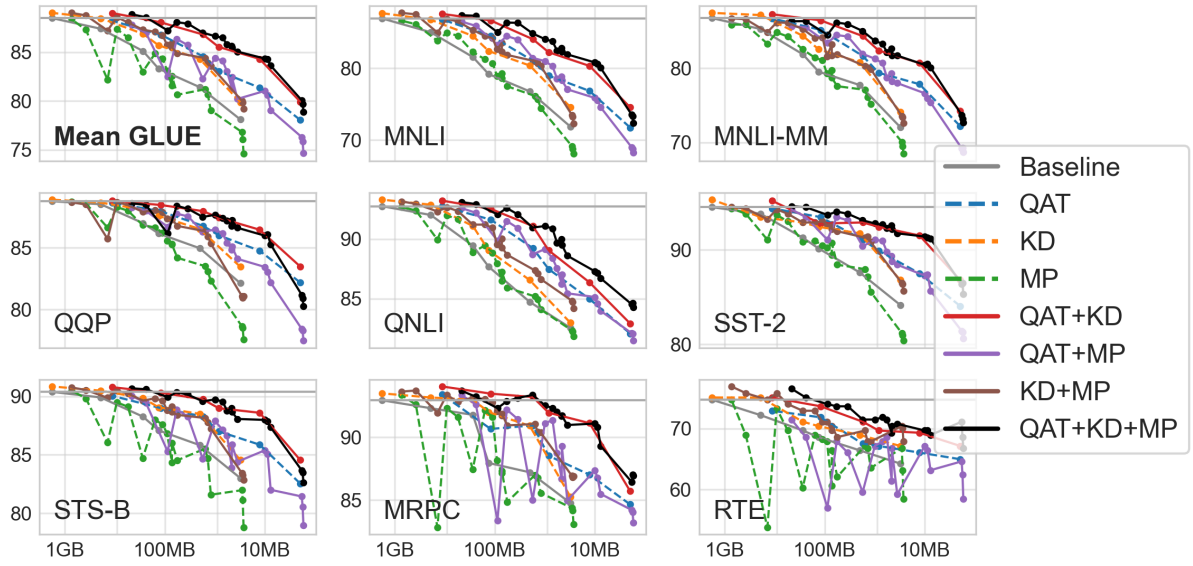

Figure A2: Accuracy vs. decreasing model size; same as Figure A1, but with the 80% pruning experiments also included (*i.e.*, 20% of weights remaining). There is a large dropoff when 80% of weights are pruned compared to 60%, especially for smaller tasks.

PlugAT: A Plug and Play Module to Defend against Textual Adversarial Attack

Rui Zheng[1*], **Rong Bao**[1,2*], **Qin Liu**[3], **Tao Gui**[4†], **Qi Zhang**[1,5†], **Xuanjing Huang**[1], **Rui Xie**[6], **Wei Wu**[5]
[1] School of Computer Science, Fudan University [2] Ant Group
[3] Viterbi School of Engineering, University of Southern California
[4] Institute of Modern Languages and Linguistics, Fudan University, Shanghai, China
[5] Shanghai Collaborative Innovation Center of Intelligent Visual Computing
[6] Meituan Inc., Beijing, China
{rzheng20,tgui,qz,xjhuang}@fudan.edu.cn
rbao22@m.fudan.edu.cn qliu4174@usc.edu

Abstract

Adversarial training, which minimizes the loss of adversarially perturbed examples, has received considerable attention. However, these methods require modifying all model parameters and optimizing the model from scratch, which is parameter inefficient and unfriendly to the already deployed models. As an alternative, we propose a pluggable defense module PlugAT, to provide robust predictions by adding a few trainable parameters to the model inputs while keeping the original model frozen. To reduce the potential side effects of using defense modules, we further propose a novel forgetting restricted adversarial training, which filters out bad adversarial examples that impair the performance of original ones. The PlugAT-equipped BERT model substantially improves robustness over several strong baselines on various text classification tasks, whilst training only 9.1% parameters. We observe that defense modules trained under the same model architecture have domain adaptation ability between similar text classification datasets.

1 Introduction

Deep neural networks have achieved great success in many fields, but they can be easily fooled by adversarial examples crafted by imperceptible perturbations on their normal counterparts (Goodfellow et al., 2015). Recent studies have shown that this phenomenon is widespread in NLP tasks (Jia and Liang, 2017; Liang et al., 2018; Wallace et al., 2019). In response to adversarial attackers, various adversarial defense methods are proposed to improve model robustness while maintaining high accuracy on both clean and adversarial examples (Dinan et al., 2019; Wang et al., 2021; Zheng et al., 2022; Liu et al., 2022).

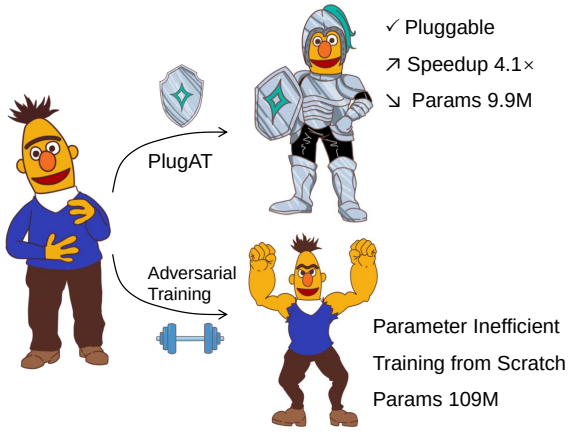

Figure 1: Comparison of adversarial training and our proposed PlugAT. Adversarial training requires training the model from scratch and modifying all model parameters, which is time-consuming and unfriendly for the deployed model. The proposed defense module improve the robustnesss of models in an extensible and efficient manner.

Among them, adversarial training is generally regarded as one of the strongest defense methods (Madry et al., 2018).

A major drawback of adversarial learning based methods is their high computational cost, as they require multi-step gradient descent to generate adversarial examples (Andriushchenko and Flammarion, 2020). The total time consumption of adversarial training is much more than that of standard training, and therefore some recent works have attempted to reduce the computational burden of adversarial training. FastAT (Wong et al., 2020) uses weaker and cheaper adversarial examples to replace strong ones and demonstrates that extremely weak adversarial training is capable of learning robust models. YOPO (Zhang et al., 2019) limits the number of forward and backward propagation without hurting the performance of the

* Equal contribution.
† Corresponding authors.

Proceedings of the 29th International Conference on Computational Linguistics, pages 2873–2882
October 12–17, 2022.

trained model. FreeAT (Shafahi et al., 2019) and FreeLB (Zhu et al., 2020) leverage "free" strategies to generate diversified adversarial examples at a negligible additional cost compared to standard training.

These efficient approaches mainly focus on reducing the cost of generating adversarial examples, which comes at the cost of training the model from scratch with the parameters entirely modified. The prohibitively huge training demand poses a severe challenge for the deployment of practical robust NLP systems. We seek to mitigate this problem by learning external modules to achieve robust results. This way, we only need to train and load a small number of robustness-specific parameters without retraining the entire model, greatly improving the ease of use.

In this work, we propose PlugAT, a plug-and-play module for transformer-based pre-trained language models (PLMs) to defend against textual adversarial attacks. The defense module consists of layerwise trainable parameters that are prepended to the input sequence of each PLM layer. By optimizing the defense module with adversarial training, the model is guided to response with robust outputs without updating its parameters. To alleviate the possible damage caused by training adversarial examples on the performance of original examples, we propose a novel forgetting restricted adversarial training, which filter out "aggressive" adversarial examples. PlugAT-equipped BERT has promising robust performance on text classification tasks while updating only 9.1% robustness-specific parameters, reducing GPU time by about half compared to state-of-the-art efficient adversarial training methods. In addition, we prove that defense module has the domain adaptation capability and can work when transferred to similar tasks.Our codes are publicly available at *Github*[1]. The main contributions of our work are summarized as follows:

- A plug-and-play module PlugAT is proposed to improve robustness of the deployed models in an extensible and efficient manner.
- We propose the forgetting constrained adversarial training to mitigate performance degradation of the original examples caused by training "aggressive" adversarial examples.
- We verify the effectiveness of the defense module under three adversarial attacks and enrich more potential applications of adversarial training.

[1]https://github.com/ruizheng20/PlugAT

2 Related Work

2.1 Adversarial Attack

For the purpose of exploring robustness, adversarial attack has been extensively studied for continuous data of images (Goodfellow et al., 2015; Madry et al., 2018) as well as discrete data of texts (Li et al., 2018; Ren et al., 2019; Li et al., 2020), with the latter aspect being more challenging than the former. Textual attacks typically generate explicit adversarial examples by swapping the components of sentences into their counterparts, be it high in similarity semantically (Ren et al., 2019) or in terms of embedding (Li et al., 2020). TextFooler (Jin et al., 2020) and TextBugger (Li et al., 2018) leverages genetic algorithms to search for word-level substitution that is semantically similar and grammatically correct. To improve the success rate of this kind of attack, Li *et al.* (2020) repeat the process of searching and substituting until a successful attack. In this work, we demonstrate the effectiveness of our proposed method on the mentioned adversarial attacks.

2.2 Adversarial Training

To improve models' robustness against attacks, adversarial defence has also attracted increasing interests, the one of most powerful methods is adversarial training. These methods generally incorporate a min-max optimization between the adversarial perturbations and the models by limiting embedding-level perturbations to Frobenius normalization balls, such as PGD (Madry et al., 2018) and FreeLB (Zhu et al., 2020), or to token-level normalization balls as implemented in TAVAT (Li and Qiu, 2020). Since training a model from scratch is time-consuming and effort-taking, there has been a series of recent efforts to try to improve the efficiency of adversarial training. Fast adversarial training (Wong et al., 2020) uses weaker and cheaper adversarial examples to replace strong ones and achieves comparable performance. Free adversarial training (Shafahi et al., 2019) recycles the gradient information computed when updating model parameters to generate adversarial examples for free. These two methods eliminate the overhead cost of generating adversarial examples. Zhang *et*

al. (2019) restrict most of the forward and back propagation within the first layer of the network during adversary updates limits, which reduces the total number of propagation and largely saves GPU time. A more practical question is that adversarial training needs to optimize all model parameters, which is inefficient and unfriendly for the deployed model in industrial applications.

2.3 Lightweight method

Researchers have long been studying how to efficiently transfer pre-trained models to downstream tasks (Ye et al., 2021; Ma et al., 2022). Adapter-tuning (Houlsby et al., 2019) inserts small task-specific trainable modules between the layers of pre-trained language models, and only trains these adapter modules while keeping the original network frozen. Similarly , Side-tuning (Zhang et al., 2020) trains a lightweight "side" network that can be fused with a pre-trained network through summation. Being even more lightweight, Prefix-Tuning (Li and Liang, 2021) maintains comparable performance while keeping model parameters frozen and tuning only 0.1% of the parameters to optimize a small continuous task-specific vector. These methods have been proven to reduce catastrophic forgetting in downstream tasks, and because the parameters are optimized in a limited space, they are more robust to adversarial attacks (Han et al., 2021). We take inspiration from these works and study how to use trainable modules to conduct adversarial training in an efficient and extensible method.

3 Methodology

In this section, we first detail the adversarial training on PLMs and then introduce our proposed plug-and-play defense module, PlugAT, which helps PLMs to get rid of optimizing all model parameters from scratch when performing adversarial training. Finally, we design a novel adversarial training method based on gradient alignment constraints and avoid performance degradation on clean examples.

3.1 Preliminaries

For a classification dataset $\mathcal{D} = \{\mathcal{X}, \mathcal{Y}\}$, where $\mathcal{X}$ is the set of input examples and $\mathcal{Y}$ is the set of label classes, we have a Transformer-based PLM with parameters θ, such as BERT to learn a mapping function $f : \mathcal{X} \rightarrow \mathcal{Y}$. Given an input instance $X \in \mathcal{X}$ that consists of several tokens $X = \{x_1, \ldots, x_{|X|}\} \in \mathcal{X}$, $Y \in \mathcal{Y}$ is the label, the PLM first converts it into the embeddings:

$$Z_0 = \mathrm{Embed}(X) \in \mathbb{R}^{|X| \times d}, \tag{1}$$

where d is hidden size and $\mathrm{Embed}(\cdot)$ denotes the input embedding layer, then the hidden states are encoded by l-th Transformer layer:

$$Z_l = \mathrm{Trans}_l(Z_{l-1}) \in \mathbb{R}^{|X| \times d}, \tag{2}$$

where $\mathrm{Trans}_l(\cdot)$ is l-th Transformer layer. Finally, the hidden states Z_L in last layer L is used to decode Y.

Adversarial Training. Adversarial training is a method for hardening classifiers against adversarial attacks and involves training the network on adversarially perturbed inputs. The adversarial training solves a min-max optimization problem as follows:

$$\min_{\theta} \mathbb{E}_{(X,Y)\sim\mathcal{D}} \max_{\|Z'_0 - Z_0\|_F \leq \epsilon} \mathcal{L}(f(X', \theta), Y), \tag{3}$$

where X' is an adversarial example of X and ϵ is the maximum perturbation bound. The *inner maximization* problem is to find an adversarial example within the ϵ-ball centered at X in the embedding space ($\|Z'_0 - Z_0\|_F \leq \epsilon$) that maximizes the classification loss. PGD applies the K-step stochastic gradient descent to search for the perturbation δ (Madry et al., 2018):

$$\delta_{k+1} = \prod_{\|\delta\| \leq \epsilon} \left(\delta_k + \eta \frac{g(\delta_k)}{\|g(\delta_k)\|} \right), \tag{4}$$

where $g(\delta_k) = \nabla_\delta \mathcal{L}(f(X + \delta_k, \theta), Y)$, δ_k is the perturbation in k-th step and $\prod_{\|\delta\| \leq \epsilon}(\cdot)$ projects the perturbation back onto the Frobenius normalization ball. Then robust training optimizes the network on adversarially perturbed input $X' = X + \delta_K$. And the objective of the *outer minimization* problem is to optimize the model parameters so that the loss on the adversarial examples is minimized.

In traditional adversarial training, all PLM parameters θ need to participate in the optimization starting from pre-training weights. It leads to the following limitations in practice: 1) it is not cost effective to retrain a large-scale deployed model for additional robustness; 2) it requires more training iterations (typically 3 times or more) compared to fine-tuning, so optimizing all model parameters is time-consuming and parametrically inefficient.

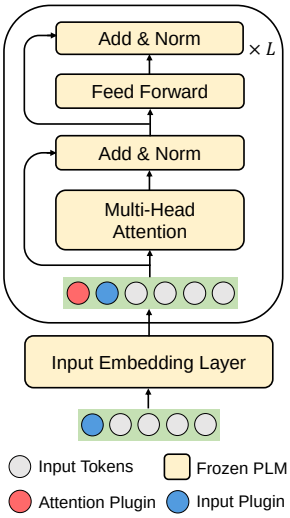

Figure 2: Architecture of the defense module and its integration with the Transformer-based model. **Input plugin** can be regarded as "virtual inputs", which is added before the original input tokens. **Attention plugin** is inserted into all multi-head self-attention layers. Then we conduct adversarial training on defense modules to improve robustness.

3.2 PlugAT

We propose a plug-and-play adversarial defense module as an efficient alternative for the deployed model without modifying its structure and parameters. Inspired by recent prompt-based approaches, we design the defense module $\mathcal{P}_\phi$ with parameters ϕ, which is a series of tunable vectors that are added before inputs of all Transformer layers to guide the model to make correct predictions in the face of adversarial examples. Thus, the inputs of the first Transformer layer (i.e., outputs of the input embedding layer) is:

$$Z_1^* = \text{Embed}(P_0; Z_0) \in \mathbb{R}^{(N+|X|)\times d}, \quad (5)$$

where $P_0 \in \mathbb{R}^{N\times d}$ denotes the part of $\mathcal{P}_\phi$ in the input embedding layer, N is the length of P_0, and $\mathcal{P}_\phi = \{P_0, P_1, \ldots, P_{L-1}\}$. Then the modified hidden states Z_l^* output by l-th layer is:

$$Z_l^* = \text{Trans}_l(P_{l-1}; Z_{l-1}^*[i > N]) \in \mathbb{R}^{(N+|X|)\times d}, \quad (6)$$

where $Z_{l-1}^*[i > N]$ is the hidden states at positions larger than N, which is the hidden states of original input tokens. However, previous work found these prefix parameters are difficult to optimize, which confirms similar observations in our experimental section. To tackle this problem, the structure of $\mathcal{P}_\phi$ consists of two parts: input plugin $\mathcal{I}_\phi$ and attention plugin $\mathcal{A}_\phi = \{A_0, \ldots, A_{L-1}\}$. They are parameters that affect the model behaviour globally and locally, respectively. The architecture of PlugAT is illustrated in Figure 2.

Input Plugin. Input plugin can be regarded as "virtual input tokens" that is added before original input embeddings, and its effects will propagate upward to all Transformer layers. Thus, the augmented input embeddings in (5) can be expanded as:

$$Z_0^* = \text{Embed}(\mathcal{I}_\phi; X) \in \mathbb{R}^{(N_I+|X|)\times d}, \quad (7)$$

where $\mathcal{I}_\phi$ is the "virtual input tokens" of length N_I. We optimize the parameters of the $\mathcal{I}_\phi$ only in the embedding layer, while their hidden states in the upper layers are automatically encoded by PLMs. This means that $\mathcal{I}_\phi$ acts as a part of the original inputs, providing global semantic information to the model, which makes the optimization of $\mathcal{P}_\phi$ more stable.

Attention Plugin. Attention plugin is composed of feature vectors inserted into all multi-head self-attention layers. The hidden states of l-th Transformer layer is encoded as:

$$Z_l^* = \text{Trans}_l(A_{l-1}; Z_{l-1}^*[i > N_A]) \in \mathbb{R}^{(N+|X|)\times d}, \quad (8)$$

where $N = N_I + N_A$ and N_A is the length of attention plugin. The parameter of $\mathcal{A}_\phi$ are independent at each layer, which avoids long-range dependencies and introduces more trainable parameters. The optimization of A_l is sensitive to the learning rate and initialization, thus A_l is reparameterized by a two-layer perceptron as shown below:

$$A_l = W_2(\tanh(W_1 A_l' + b_1) + b_2), \quad (9)$$

where $W_1 \in \mathbb{R}^{d'\times d}$, $W_2 \in \mathbb{R}^{N_A\times d'}$, $b_1 \in \mathbb{R}^{d'}$, $b_2 \in \mathbb{R}^{N_A\times d}$ and $A_l' \in \mathbb{R}^d$ are trainable parameters.

3.3 Forgetting of Clean Examples

Deep networks do not perform well in the sequential continual learning setting because they forget the past learned tasks after learning new ones. Therefore, it is necessary to overcome the problem of forgetting clean samples caused by using PlugAT in the deployed model. Our goal is to understand the effect of training adversarial examples $\{X_1', \ldots, X_{|\mathcal{D}|}'\}$ on their clean counterparts $\{X_1, \ldots, X_{|\mathcal{D}|}\}$. We follow a similar analysis procedure used in the influence function to implement it (Koh and Liang, 2017). First,

the parameter changes induced from the training of adversarial example can be calculated by reweighting (X', Y) with some λ: $\hat{\phi}_{\lambda,X'} \triangleq \arg\min_\phi \frac{1}{|\mathcal{D}|}\sum_{i=1}^{|\mathcal{D}|}\mathcal{L}(X'_i,\phi) + \lambda\mathcal{L}(X',\phi)$ where $\mathcal{L}(X'_i,\phi)$ is a shorthand of $\mathcal{L}(f(X'_i,\phi),Y_i)$. Atkinson et al. (1983) show that the influence of reweighting X' on the parameters $\hat{\phi}$ is

$$\frac{d\hat{\phi}_{\lambda,X'}}{d\lambda}\Big|_{\lambda=0} = -H_{\hat{\phi}}^{-1}\nabla_\phi\mathcal{L}(X',\phi), \tag{10}$$

where $H_{\hat{\phi}} \triangleq \frac{1}{|\mathcal{D}|}\sum_{i=1}^{|\mathcal{D}|}\nabla_\phi^2\mathcal{L}(X'_i,\hat{\phi})$ is the positive definite Hessian matrix. Second, we apply the chain rule to measure the influence of reweighting X' on the loss of X:

$$\begin{aligned}\mathcal{F}(X',X) &\triangleq \frac{d\mathcal{L}(X'_i,\hat{\phi}_{\lambda,X'})}{d\lambda}\Big|_{\lambda=0}\\ &= \nabla_\phi\mathcal{L}(X,\hat{\phi})^T\frac{d\hat{\phi}_{\lambda,X'}}{d\lambda}\Big|_{\lambda=0}\\ &= -\nabla_\phi\mathcal{L}(X,\hat{\phi})^T H_{\hat{\phi}}^{-1}\nabla_\phi\mathcal{L}(X',\hat{\phi}).\end{aligned} \tag{11}$$

$\mathcal{F}(X',X) > 0$ means the training process of the adversarial example X' will impair the performance of the clean one X. Pezeshkpour *et al.* (2021) show that Hessian information is unnecessary and $\mathcal{F}(X',X)$ can be approximated as $\mathcal{F}(X',X) \approx -\nabla_\phi\mathcal{L}(X,\hat{\phi})^T\mathcal{L}(X',\hat{\phi})$. For computational reasons, we use the subset of parameters corresponding to the last linear layer to compute $\mathcal{F}(X',X)$.

By combining the above constraints with adversarial training in (3), we propose the forgetting restricted adversarial training (FRAT) for our defense module:

$$\begin{aligned}&\min_\phi \mathbb{E}_{(X,Y)\sim\mathcal{D}}\max_{\|Z'_0-Z_0\|_F\le\epsilon}\mathcal{L}(f(X',\phi),Y),\\ &\text{s.t. } \mathcal{F}(X',X)\le 0.\end{aligned} \tag{12}$$

We use only the parameters of proposed defense module to optimize the above objectives, while keeping the original model frozen.

Method: PlugAT

Input: Training dataset $\mathcal{D} = \{(X_i, Y_i)\}_{i=1}^M$, perturbation bound ϵ, initialization of perturbation σ, ascent step size η, ascent steps K, a deployed model f_θ, learning rate τ, defense module $\mathcal{P}_\phi$

Output: defense module parameters ϕ

Initialize $\mathcal{D}_\phi$
for epoch$= 1, \ldots, N_{ep}$ **do**
  for minibatch $B \subset \mathcal{P}$ **do**
    $\delta_0 \leftarrow \frac{1}{N_\delta}\text{Uniform}(-\sigma,\sigma)$
    for $k = 0, \ldots, K-1$ **do**
      $\mathbf{g}_k \leftarrow \nabla_\delta\mathcal{L}(f_\theta(X+\delta,\phi),Y)$
      $\delta_k \leftarrow \prod_{\|\delta\|_F\le\epsilon}(\delta_k + \eta\cdot\mathbf{g}_k/\|\mathbf{g}_k\|_F)$
      if $\mathcal{F}(X+\delta_k, X) > 0$ **do**
        $\delta_k \leftarrow \delta_{k-1}$
        break
      end if
    end for
    $\phi \leftarrow \phi - \tau\nabla_\phi\mathcal{L}(f_\theta(X+\delta_k,\phi),Y)$
  end for
end for
return: ϕ

4 Experimental Setup

We first introduce the experimental setup, including datasets, baselines and adversarial attackers involved in our experiments.

4.1 Datasets

We compare the proposed method with baselines on two widely applied benchmark corpora: IMDB dataset (Maas et al., 2011) and SST-2 dataset (Socher et al., 2013). Collected from online movie reviews, IMDB contains long samples created by users, while SST samples are shorter (with an average length of 19 words) and show a higher diversity.

4.2 Baseline Methods

We select four adversarial defence methods as baselines for a thorough comparison, including three methods of adversarial training, one from information-theoretic perspective, and one of adversarial data augmentation.

PGD (Madry et al., 2018): A gradient-based adversarial training method that uses multiple projected gradient ascent steps to find the adversarial perturbations, which is then leveraged to update the model parameters.

FreeLB (Zhu et al., 2020): A similar gradient-based method with PGD while accumulating the gradients and then back propagate once, resulting in a faster process of searching for more effective perturbations.

TAVAT (Li and Qiu, 2020): An token-aware

adversarial training method that crafts fine-grained perturbations by constraining them into a token-level normalization ball.

InfoBERT (Wang et al., 2021): A learning framework for robust fine-tuning from an information-theoretic perspective, where two mutual-information-based regularizers are used for model training.

4.3 Adversarial Attack Methods

Three widely accepted attack methods are used to verify the ability of our proposed method against baselines.

TextBugger (Li et al., 2018): An adversarial attack method that is applicable in both white-box and black-box scenarios. Important words or sentences (in white- or black-box scenario respectively) are first discovered, and then an optimal perturbation is chosen from the generated perturbations, or, for the black-box situation, a scoring function is leveraged to find important words to manipulate.

BERT-Attack (Li et al., 2020): A method using BERT to generate adversarial text, and thus the generated adversarial examples are fluent and semantically preserved.

TextFooler (Jin et al., 2020): A comprehensive attack paradigm that creates adversarial examples that maintain human prediction consistency, semantic similarity, and language fluency. The important words for the target model are first identified and are then replaced with words, which are semantically similar and grammatically correct, until the prediction is altered.

4.4 Evaluation Metrics

The evaluation metrics adopted in our experimental analyses are listed as follows. For a robust defense method, higher accuracy under attack as well as query times is expected.

Clean Accuracy (Clean%): The accuracy on the clean test dataset. **Accuracy under Attack (Aua%)**: The model's prediction accuracy facing specific adversarial attacks. **Number of Queries (#Query)** The average number of times the attacker queries the model, which means the more average query number is, the harder the defense model is to be compromised. **Trainable Params**: The number of parameters to be optimized. **Speed UP**: The training speedups re reported against PGD and we evaluate all the adversarial defense methods on an NVIDIA RTX3090 GPU.

4.5 Implementation Details

Our implementation of PlugAT is mainly based on BERT and FreeLB, so most of the hyperparameter settings are based on these methods.[2] We use AdamW as our optimizers with the learning rate $1e^{-5}$, a batch size $\in \{16, 32\}$ and a linear learning rate decay schedule with a warm-up of 0.1. The dropout rate is set to 0.1 for all task-specific layers. For the defense module, $N_A = 35$, $N_i = 5$ and $d' = 512$. We set the adversarial perturbation step as 2, the perturbation step size as 0.03, the constrain bound of perturbations as 0.1, the initial magnitudes of perturbations as 0.05. The maximum number of epochs is set to 10. Since the results of the SST-2 test set need to be submitted to the GLUE evaluation server and cannot be used for adversarial attacks, the results of SST-2 are tested on the development set.

We implement three adversarial attack methods using TextAttack framework and follow the default parameter settings.[3] We adopt a "free" strategy similar to FreeLB (Zhu et al., 2020) to solve the inner maximization problem in (12) and obtain more adversarial examples. The clean accuracy (**(Clean%)**) is tested on the whole test/dev set. Other adversarial robustness evaluation metrics (e.g., **Aua**% and **#Query**) are evaluated on the dev set for SST-2 and 1000 randomly selected test samples for IMDB. We choose models trained during the last period to compare the robustness. All experiments are conducted using NVIDIA RTX3090 GPUs.

4.6 Main Results

Defense Effectiveness. As illustrated in Table 1, BERT equipped with our module with forgetting restricted adversarial training has seen an increase in robustness against attacks while maintaining clean accuracy. The adversarial training based approachs, e.g., FreeLB, achieve higher clean accuracy than the baseline and our proposed method, and their improvement in robustness is also very insignificant. Generally, PlugAT lifts BERT's accuracy under attack from 4.9% to 20.8% on SST-2 and 12.2% to 36.1% on IMDB when attacked by TextFooler, which outperforms all of the compared methods. The improvements indicate that our proposed defense module indeed helps BERT achieve greater robustness against attacks

[2] https://github.com/huggingface/transformers

[3] https://github.com/QData/TextAttack

Dataset	Methods	Trainable Params	Speed Up	Clean%	TextFooler		BERT-ATTACK		TextBugger	
					Aua%	#Query	Aua%	#Query	Aua%	#Query
SST-2	Fine-tune	109M	–	92.5	4.9	98.5	2.9	114.2	26.3	48.6
	PGD	109M	1×	92.9	16.5	122.3	11.8	156.8	43.4	56.1
	FreeLB	109M	2.1×	**93.1**	14.4	123.8	10.2	154.6	42.4	54.9
	TAVAT	109M	2.8×	93.1	13.5	117.8	9.9	147.9	38.5	49.7
	InfoBERT	109M	1.3×	92.1	18.3	121.4	14.4	162.3	40.3	51.2
	PlugAT	**9.9M**	**4.1×**	92.6	**20.8**	**126.5**	**15.6**	**162.4**	**41.8**	**56.3**
IMDB	Fine-tune	109M	–	94.2	12.2	1209.8	7.8	1572.2	25.8	783.2
	PGD	109M	1×	94.6	34.5	1616.0	28.6	2454.7	43.5	957.3
	FreeLB	109M	3.4×	94.7	27.2	1479.1	22.6	1954.7	36.0	907.3
	TAVAT	109M	4.1×	94.3	21.2	1417.7	17.6	1825.4	32.9	842.6
	InfoBERT	109M	1.2×	**95.2**	32.4	1572.2	26.0	2326.0	43.6	969.8
	PlugAT	**9.9M**	**6.1×**	93.9	**36.1**	**1624.9**	**32.8**	**2777.1**	**44.7**	**1045.3**

Table 1: Main results of the adversarial robustness evaluation on two text classification datasets. The defense module **PlugAT** achieves better robustness with fewer trainable parameters and less GPU time. The best performance is marked in bold.

compared with a vanilla BERT with the same kind of adversarial training while maintaining the clean accuracy to a large extent. Moreover, our proposed method helps BERT to undergo adversarial training by learning only 9.1% parameters, and the computational cost is also outstanding among the baseline methods.

Dataset		Methods	Clean%	TextFooler	
Source	Target			Aua%	#Query
SST-2	IMDB	Fine-tune	90.8	2.0	564.1
		FreeLB	89.2	22.5	874.2
		PGD	89.9	20.0	827.8
		PlugAT	**93.8**	**34.0**	**1536.0**
IMDB	SST-2	Fine-tune	86.7	3.2	87.7
		FreeLB	87.8	4.5	93.5
		PGD	88.5	5.4	100.8
		PlugAT	**91.4**	**10.3**	**113.1**

Table 2: Experimental results when the models are trained on the **source** dataset and then transferred to the **target** dataset for testing. BERT (fine-tuned on the **target** dataset) equipped with our module (trained on the **source** dataset) improves robustness to a large extent and with minor damage to the clean accuracy.

Transferability. The proposed method is competitive even in challenging scenarios, with the extra robustness obtained by our defense module being able to generalize across datasets. To get a clear view of this phenomenon, we conduct transferability experiments on IMDB and SST-2 datasets, which is reported in Table 2. The results indicate that the robustness of BERT with PlugAT trained on one dataset can transfer to the other to a degree. To be more specific, BERT with the defense module trained on SST-2 dataset transfers its ability of defending to IMDB and vice versa, while still outperforming the BERT trained on each dataset in all of the three metrics. This suggests that the defense module shows strong potential in transferability, the resource of which is left for a more thorough investigation of future work. We believe that the reason for the above phenomenon may be that these two datasets are similar. Another possible explanation is that due to the transferability of adversarial examples, the adversarial examples generated by adversarial training may be similar on SST-2 and IMDB. These adversarial examples may make the model more robust on both SST-2 and IMDB.

4.7 Intrinsic Evaluation

Several variants that affect the performance of our method are compared, including the length of input or attention plugins. Besides, we conduct ablation experiments to study the contribution of each component in the proposed method.

Effect of Attention Plugin. A longer attention plugin introduces more trainable parameters, enabling stronger expressive power. The lengths of 5 to 45 with an interval of 5 are experimented. As reported in Figure 3, **Aua%** increases as the length increases up to a threshold (about 35) and then a slight performance drop occurs.

Effect of Input Plugin. We find that the input

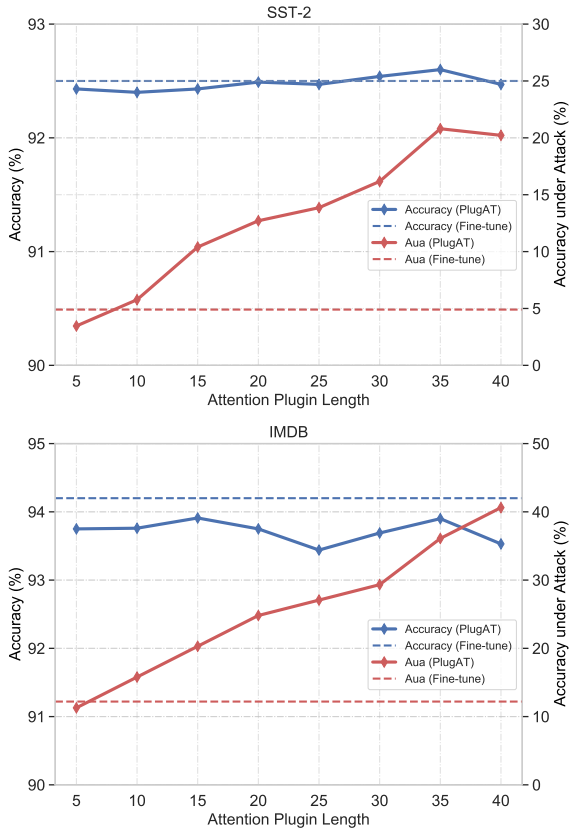

Figure 3: Length of attention plugins vs. performance on SST-2 and IMDB datasets. Performance increases up to a threshold 35 and then a slight performance drop occurs.

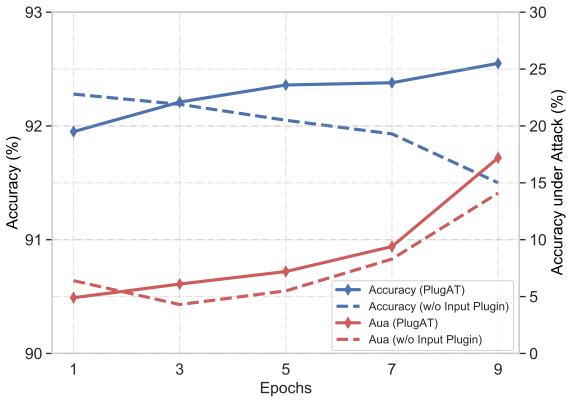

Figure 4: Full vs. without input plugin. The input plugin enables the training process to be more stable and reach a better performance.

Methods	Clean%	TextFooler	
		Aua%	#Query
PlugAT	92.6	20.8	126.5
w/o FRAT	91.5	21.5	128.2
w/o Input Plugin	92.1	16.7	114.6
w/o Attention Plugin	91.9	6.3	101.2

Table 3: Ablation study of PlugAT. We can observe that forgetting restricted adversarial training (FRAT) can effectively mitigate forgetting on clean data.

plugin makes the training process more stable and reduces the performance fluctuations. Figure 4 shows that generally, the existence of input plugin enables the deployed model to initialize better with a starting point at a higher accuracy both on clean data and under attack.

Ablation Study. The results of ablation experiments are demonstrated in Table 3. The results show that the model's clean accuracy decreases when forgetting restricted adversarial training is removed, indicating that forgetting restricted adversarial training can effectively mitigate forgetting on clean data. As we showed before, the global information provided by the input plugin makes the training more stable and slightly improves the performance. The attention plugin, on the other hand, provides more local information, making the model more expressive.

5 Conclusion

In this paper, we focus on improving adversarial training in the NLP field. We propose PlugAT, a plug-and-play module, as an effective solution in terms of lifting model's robustness while preserving its clean accuracy as well as cutting down on the overall computational cost. It is discovered that despite learning $10\times$ fewer parameters than conventional adversarial training, the proposed defense module also shows the potential in transferability. Empirical evaluations on two widely used benchmark datasets demonstrate that training models with the defense module costs half the GPU time while outperforming the existing adversarial training methods. Extensive experiments demonstrate that our proposed module can significantly improve the model robustness and transfer the performance across different datasets effectively. Besides, we hope our work could inspire more research on improving adversarial training in NLP.

Acknowledgements

The authors wish to thank the anonymous reviewers for their helpful comments. This work was partially funded by National Natural Science Foundation of China (No. 62076069, 61976056), Shanghai

Municipal Science and Technology Major Project (No.2021SHZDZX0103). This research was supported by Meituan and Ant Group through Ant Research Intern Program.

References

Maksym Andriushchenko and Nicolas Flammarion. 2020. Understanding and improving fast adversarial training. In *Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020, December 6-12, 2020, virtual*.

Anthony C. Atkinson, R. Dennis Cook, and Sanford Weisberg. 1983. Residuals and influence in regression. *Biometrics*, 39:818.

Emily Dinan, Samuel Humeau, Bharath Chintagunta, and Jason Weston. 2019. Build it break it fix it for dialogue safety: Robustness from adversarial human attack. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 4537–4546, Hong Kong, China. Association for Computational Linguistics.

Ian J. Goodfellow, Jonathon Shlens, and Christian Szegedy. 2015. Explaining and harnessing adversarial examples. In *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*.

Wenjuan Han, Bo Pang, and Ying Nian Wu. 2021. Robust transfer learning with pretrained language models through adapters. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 2: Short Papers)*, pages 854–861, Online. Association for Computational Linguistics.

Neil Houlsby, Andrei Giurgiu, Stanislaw Jastrzebski, Bruna Morrone, Quentin de Laroussilhe, Andrea Gesmundo, Mona Attariyan, and Sylvain Gelly. 2019. Parameter-efficient transfer learning for NLP. In *Proceedings of the 36th International Conference on Machine Learning, ICML 2019, 9-15 June 2019, Long Beach, California, USA*, volume 97 of *Proceedings of Machine Learning Research*, pages 2790–2799. PMLR.

Robin Jia and Percy Liang. 2017. Adversarial examples for evaluating reading comprehension systems. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 2021–2031, Copenhagen, Denmark. Association for Computational Linguistics.

Di Jin, Zhijing Jin, Joey Tianyi Zhou, and Peter Szolovits. 2020. Is BERT really robust? A strong baseline for natural language attack on text classification and entailment. In *The Thirty-Fourth AAAI Conference on Artificial Intelligence, AAAI 2020, The Thirty-Second Innovative Applications of Artificial Intelligence Conference, IAAI 2020, The Tenth AAAI Symposium on Educational Advances in Artificial Intelligence, EAAI 2020, New York, NY, USA, February 7-12, 2020*, pages 8018–8025. AAAI Press.

Pang Wei Koh and Percy Liang. 2017. Understanding black-box predictions via influence functions. In *Proceedings of the 34th International Conference on Machine Learning, ICML 2017, Sydney, NSW, Australia, 6-11 August 2017*, volume 70 of *Proceedings of Machine Learning Research*, pages 1885–1894. PMLR.

Jinfeng Li, Shouling Ji, Tianyu Du, Bo Li, and Ting Wang. 2018. Textbugger: Generating adversarial text against real-world applications. *ArXiv preprint*, abs/1812.05271.

Linyang Li, Ruotian Ma, Qipeng Guo, Xiangyang Xue, and Xipeng Qiu. 2020. BERT-ATTACK: Adversarial attack against BERT using BERT. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 6193–6202, Online. Association for Computational Linguistics.

Linyang Li and Xipeng Qiu. 2020. Tavat: Token-aware virtual adversarial training for language understanding. *ArXiv preprint*, abs/2004.14543.

Xiang Lisa Li and Percy Liang. 2021. Prefix-tuning: Optimizing continuous prompts for generation. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 4582–4597, Online. Association for Computational Linguistics.

Bin Liang, Hongcheng Li, Miaoqiang Su, Pan Bian, Xirong Li, and Wenchang Shi. 2018. Deep text classification can be fooled. In *Proceedings of the Twenty-Seventh International Joint Conference on Artificial Intelligence, IJCAI 2018, July 13-19, 2018, Stockholm, Sweden*, pages 4208–4215. ijcai.org.

Qin Liu, Rui Zheng, Bao Rong, Jingyi Liu, ZhiHua Liu, Zhanzhan Cheng, Liang Qiao, Tao Gui, Qi Zhang, and Xuanjing Huang. 2022. Flooding-X: Improving BERT's resistance to adversarial attacks via loss-restricted fine-tuning. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 5634–5644, Dublin, Ireland. Association for Computational Linguistics.

Ruotian Ma, Xin Zhou, Tao Gui, Yiding Tan, Linyang Li, Qi Zhang, and Xuanjing Huang. 2022. Template-free prompt tuning for few-shot NER. In *Proceedings of the 2022 Conference of the North American*

Chapter of the Association for Computational Linguistics: Human Language Technologies, pages 5721–5732, Seattle, United States. Association for Computational Linguistics.

Andrew L. Maas, Raymond E. Daly, Peter T. Pham, Dan Huang, Andrew Y. Ng, and Christopher Potts. 2011. Learning word vectors for sentiment analysis. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, pages 142–150, Portland, Oregon, USA. Association for Computational Linguistics.

Aleksander Madry, Aleksandar Makelov, Ludwig Schmidt, Dimitris Tsipras, and Adrian Vladu. 2018. Towards deep learning models resistant to adversarial attacks. In *6th International Conference on Learning Representations, ICLR 2018, Vancouver, BC, Canada, April 30 - May 3, 2018, Conference Track Proceedings*. OpenReview.net.

Pouya Pezeshkpour, Sarthak Jain, Byron Wallace, and Sameer Singh. 2021. An empirical comparison of instance attribution methods for NLP. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 967–975, Online. Association for Computational Linguistics.

Shuhuai Ren, Yihe Deng, Kun He, and Wanxiang Che. 2019. Generating natural language adversarial examples through probability weighted word saliency. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 1085–1097, Florence, Italy. Association for Computational Linguistics.

Ali Shafahi, Mahyar Najibi, Amin Ghiasi, Zheng Xu, John P. Dickerson, Christoph Studer, Larry S. Davis, Gavin Taylor, and Tom Goldstein. 2019. Adversarial training for free! In *Advances in Neural Information Processing Systems 32: Annual Conference on Neural Information Processing Systems 2019, NeurIPS 2019, December 8-14, 2019, Vancouver, BC, Canada*, pages 3353–3364.

Richard Socher, Alex Perelygin, Jean Wu, Jason Chuang, Christopher D. Manning, Andrew Ng, and Christopher Potts. 2013. Recursive deep models for semantic compositionality over a sentiment treebank. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*, pages 1631–1642, Seattle, Washington, USA. Association for Computational Linguistics.

Eric Wallace, Shi Feng, Nikhil Kandpal, Matt Gardner, and Sameer Singh. 2019. Universal adversarial triggers for attacking and analyzing NLP. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 2153–2162, Hong Kong, China. Association for Computational Linguistics.

Boxin Wang, Shuohang Wang, Yu Cheng, Zhe Gan, Ruoxi Jia, Bo Li, and Jingjing Liu. 2021. Infobert: Improving robustness of language models from an information theoretic perspective. In *9th International Conference on Learning Representations, ICLR 2021, Virtual Event, Austria, May 3-7, 2021*. OpenReview.net.

Eric Wong, Leslie Rice, and J. Zico Kolter. 2020. Fast is better than free: Revisiting adversarial training. In *8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, April 26-30, 2020*. OpenReview.net.

Jiacheng Ye, Tao Gui, Yichao Luo, Yige Xu, and Qi Zhang. 2021. One2Set: Generating diverse keyphrases as a set. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 4598–4608, Online. Association for Computational Linguistics.

Dinghuai Zhang, Tianyuan Zhang, Yiping Lu, Zhanxing Zhu, and Bin Dong. 2019. You only propagate once: Accelerating adversarial training via maximal principle. In *Advances in Neural Information Processing Systems 32: Annual Conference on Neural Information Processing Systems 2019, NeurIPS 2019, December 8-14, 2019, Vancouver, BC, Canada*, pages 227–238.

Jeffrey O Zhang, Alexander Sax, Amir Zamir, Leonidas Guibas, and Jitendra Malik. 2020. Side-tuning: A baseline for network adaptation via additive side networks. In *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part III 16*, pages 698–714. Springer.

Rui Zheng, Bao Rong, Yuhao Zhou, Di Liang, Sirui Wang, Wei Wu, Tao Gui, Qi Zhang, and Xuanjing Huang. 2022. Robust lottery tickets for pre-trained language models. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 2211–2224, Dublin, Ireland. Association for Computational Linguistics.

Chen Zhu, Yu Cheng, Zhe Gan, Siqi Sun, Tom Goldstein, and Jingjing Liu. 2020. Freelb: Enhanced adversarial training for natural language understanding. In *8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, April 26-30, 2020*. OpenReview.net.

Automatic ICD coding exploiting discourse structure and reconciled code embeddings

Shurui Zhang[1], **Bozheng Zhang**[1], **Fuxin Zhang**[1], **Bo Sang**[1] **and Wanchun Yang** [*21]
[1]MsunHealth Co., LTD
[2]Shandong Jiaotong University
{zhang65821, oscarzbz, zfx050621, kangtasz}@gmail.com
yangwanchun82@gmail.com

Abstract

The International Classification of Diseases (ICD) is the foundation of global health statistics and epidemiology. The ICD is designed to translate health conditions into alphanumeric codes. A number of approaches have been proposed for automatic ICD coding, since manual coding is labor-intensive and there is a global shortage of healthcare workers. However, existing studies did not exploit the discourse structure of clinical notes, which provides rich contextual information for code assignment. In this paper, we exploit the discourse structure by leveraging section type classification and section type embeddings. We also focus on the class-imbalanced problem and the heterogeneous writing style between clinical notes and ICD code definitions. The proposed reconciled embedding approach is able to tackle them simultaneously. Experimental results on the MIMIC dataset show that our model outperforms all previous state-of-the-art models by a large margin. The source code is available at https://github.com/discnet2022/discnet

1 Introduction

The International Classification of Diseases (ICD) is a classification system maintained by the World Health Organization. The system is designed to map health conditions to pre-defined ICD codes, allowing the world to share healthcare data in a consistent way between different regions. The foundation of global health statistics and epidemiology is based on the ICD.

The ICD coding task (as shown in table 1) is usually performed by professional coders. Coders review the whole clinical documents and manually assign the most appropriate codes. However, manual coding is labor-intensive, expensive, and error-prone. The approximate cost of ICD coding is estimated to be about $25 billion per year in the US (Lang, 2007).

Clinical Document	**History of Present Illness**: A 62-year-old male with Type II diabetes mellitus, coronary artery disease, hypertension, chronic kidney disease... **Past Medical History**: Hypertension Type II Diabetes Mellitus *s/p cervical laminoplasty*... **Brief Hospital Course**: ... **Discharge Diagnosis**: Anasarca Heart failure with restrictive physiology...
Assigned ICD codes	428.31 *Diastolic heart failure* 584.9 *Acute renal failure, unspecified* 427.32 *Atrial flutter* ...

Table 1: An illustration of ICD coding task

In recent years, deep learning approaches have demonstrated promising results on ICD coding. Some of these studies improved clinical document representation by leveraging Convolutional Neural Networks (CNN) (Mullenbach et al., 2018; Xie et al., 2019). The others improved ICD code representation by exploiting the dependencies between codes (Xie and Xing, 2018; Vu et al., 2020; Cao et al., 2020). However, these approaches entail limitations. Firstly, they ignore the discourse structure of clinical documents. Secondly, most of these approaches did not consider the writing style discrepancies between ICD code descriptions and relevant clinical documents related to the codes. Thirdly, most of these approaches did not consider the class imbalanced problem of the label spaces.

Why is the discourse structure important? Medical professionals prepare clinical documents in different sections. The sections convey discourse-level information and follow rhetorical moves of argumentation (Teufel et al., 1999). Such as "History of Present Illness", "Past Medical History", followed by "Hospital Course", etc. The health conditions that appear in different sections may contribute differently to code assignments. For

*Corresponding author.

Proceedings of the 29th International Conference on Computational Linguistics, pages 2883–2891
October 12–17, 2022.

example, in table 1, the *s/p cervical laminoplasty* in the past medical history is not related to the current hospitalization and does not contribute to code assignment. In such cases, omitting discourse-level information may mislead the coding task. The identification of the discourse structure can also benefit word sense disambiguation. For example, the acronym *BS* probably signifies *blood sugar* in the laboratory test section, but more likely signifies *breath sounds* in the physical examination section (Li et al., 2010). Therefore the meaning of a health condition must be considered from a discourse-level point of view.

The heterogeneity between ICD code descriptions and relevant clinical documents. Each ICD code is associated with a code description. For example, the code description of 414.01 is *Coronary atherosclerosis of native coronary artery*. A code description provides a formal definition of an ICD code. On the contrary, clinical documents that are written by physicians usually in an informal way, accompanied with telegraphic phrases and abbreviations. For example, *Coronary artery disease* is denoted by *CAD*. The writing style is highly heterogeneous between the code descriptions and relevant clinical documents.

The class imbalanced problem. Most of the recently proposed methods are based on a per-label attention mechanism that was initially proposed by Mullenbach et al. (2018). In this setting, the attention parameters for each label can be considered as the representation for each ICD code, which are learned from relevant segments in clinical documents (hereinafter referred to as "relevant documents") that are highlighted by the attention mechanism. However, the label frequency follows a highly skewed distribution. About 50% of the codes have less than 5 occurrences. In such a case, it is difficult to learn decent representations for instance-scarce codes. Considering the nature of code descriptions and the label distribution, we argue that instance-scarce code representations are supposed to learn more from code descriptions, since code descriptions are the essential definitions of ICD codes. On the contrary, instance-rich code representations are supposed to learn more from relevant documents, since relevant documents provide various expressions of each code.

In this paper, we design a novel neural architecture for automatic ICD coding given unstructured clinical documents:

- To the best of our knowledge, our work is the first to incorporate discourse-level features into automatic ICD coding. Our proposed **Disc**ourse **N**et (DiscNet) exploits discourse-level features by utilizing section type embeddings. In addition, we combine word-level features and sentence-level features for better expressive power.
- We propose a **R**econciled **E**mbedding (RE) approach to learn ICD code representations, mitigating the class imbalanced problem while reconciling the heterogeneity between code descriptions and relevant clinical documents.
- Experimental results on the MIMIC-III dataset (Johnson et al., 2016) show that our method outperforms all previous state-of-the-art methods across evaluation metrics by a large margin.

2 Related Works

Recently released automatic ICD coding approaches are mainly based on deep learning and performed on unstructured clinical documents. Baumel et al. (2018) proposed a possibility to exploit discourse structure, which inspired our work. Mullenbach et al. (2018) proposed a convolutional attention model and outperformed existing state-of-the-art methods (Baumel et al., 2018). Li and Yu (2020) and Xie et al. (2019) improved the convolutional attention model by exploiting multi-scale features. However, it is challenging for a CNN-based model to capture long-term dependencies in a document.

Discourse analysis is a task to model language phenomena that go beyond the individual sentences (Joty et al., 2019). There are few relevant works that focus on discourse analysis in the clinical domain. Li et al. (2010) focused on the discourse analysis of clinical notes and performed argumentative zoning (Teufel et al., 1999) using a hidden markov model. Denny et al. (2009) leveraged NLP techniques to categorize section headers in clinical documents.

To reconcile the heterogeneous writing styles of diagnosis descriptions and ICD code descriptions, Xie and Xing (2018) proposed an adversarial learning approach, which inspired our work.

Some studies worked on addressing the class imbalanced problem. Mullenbach et al. (2018)

proposed a regularization method using embedded code descriptions to improve the performance on infrequent codes. However, the method worsened the average performance on the MIMIC-III dataset. Some methods improved the performance on infrequent codes by modeling the hierarchical structure of ICD codes (Xie et al., 2019; Vu et al., 2020). Zhou et al. (2021) leveraged an interactive shared representation network to alleviate the long-tail problem.

3 Method

We propose a novel neural architecture for automatic ICD coding given unstructured discharge summaries. A discharge summary from an Electronic Health Record (EHR) is an unstructured clinical document that outlines the details of a hospital stay. We partition clinical documents into sections and exploit the discourse structure by leveraging section type embeddings. The ICD code representations are learned using a reconciled embedding approach. Finally, we use a dot production to predict the codes.

3.1 Discourse Net

Discourse Net (DiscNet) exploits discourse-level features, word-level features, and sentence-level features to learn multi-granularity clinical document representations as shown in Figure 1.

3.1.1 Section Type Embeddings

Clinical documents usually contain multiple sections with nonstandardized section headings. We partition a document into sections by identifying the locations of section headings using regular expressions. Terms that clinicians use to label sections are ambiguous and various, e.g. *past medical history* might appear as *pmh*. Due to various writing conventions, we extracted more than 10,000 distinct headings. We chose the top 100 most frequent headings as known section types since they accounted for 93% of the total heading occurrence. We map each section to known section types using a naive bayes classifier based on TF-IDF vectorized section contents. Concretely, each section content is converted to a TF-IDF vector. Then a naive bayes classifier is trained using known section types as labels. Finally, the trained naive bayes model map each section to known section types. We initialize an embedding matrix for know section types: $\boldsymbol{S} = \{\boldsymbol{s}_1, \boldsymbol{s}_2, ..., \boldsymbol{s}_{100}\}$. Where each $\boldsymbol{s}$ is a d dimensional vector, representing a known section type.

3.1.2 Input Layer

The input word sequence is mapped into an embedding space using pre-trained word embeddings. The word embeddings of size $d = 100$ are pre-trained on the training set of the MIMIC-III dataset using the word2vec CBOW method (Mikolov et al., 2013). The word embedding sequence is denoted as $\boldsymbol{D} = \{\boldsymbol{w}_1, \boldsymbol{w}_2, ..., \boldsymbol{w}_{|\boldsymbol{D}|}\}$, where $\boldsymbol{w} \in \mathbb{R}^d$ denotes a word vector, $|\boldsymbol{D}|$ denotes the number of words. The input embeddings are the sum of word embeddings and section type embeddings, which can be denoted as $\boldsymbol{E} = \{\boldsymbol{e}_1, \boldsymbol{e}_2, ..., \boldsymbol{e}_{|\boldsymbol{D}|}\}$. Each $\boldsymbol{e}_i$ is the sum of a word embedding $\boldsymbol{w}_i$ and the associated section type embedding $\boldsymbol{s}_j$.

3.1.3 Multi-Granularity Representations

$\boldsymbol{E}$ is a combination of word-level features and discourse-level features. Besides that, we carry sentence-level features for better expressive power. We use a bidirectional GRU (Chung et al., 2014) to model the sequential structure of $\boldsymbol{E}$:

$$\overrightarrow{\boldsymbol{H}}, \overleftarrow{\boldsymbol{H}} = \text{BiGRU}\left(\boldsymbol{E}\right), \quad \boldsymbol{H} = \left(\overrightarrow{\boldsymbol{H}} \parallel \overleftarrow{\boldsymbol{H}}\right) \boldsymbol{W}_1 \tag{1}$$

Where $\parallel$ denotes concatenation. $\boldsymbol{W}_1 \in \mathbb{R}^{2d \times d}$ is a trainable weight matrix to project the dimensionality of the forward and backward hidden states from $2d$ to d.

Let $\boldsymbol{C} \in \mathbb{R}^{|\boldsymbol{C}| \times d}$ denote the ICD code representations obtained through the Reconciled Embedding (RE) approach (refer to subsection 3.2). $|\boldsymbol{C}|$ is the number of distinct ICD codes. $\boldsymbol{C}$ is used as attention parameters to interact with document representations. A per-label attention mechanism (Bahdanau et al., 2014) is applied to re-express a document with respect to each code.

$$\begin{gathered} \boldsymbol{Z} = \boldsymbol{H}\boldsymbol{C}^{\mathrm{T}}, \quad \boldsymbol{\alpha} = \text{Softmax}_1\left(\boldsymbol{Z}\right) \\ \boldsymbol{V}^{\text{word}} = \boldsymbol{\alpha}^{\mathrm{T}}\boldsymbol{H} \end{gathered} \tag{2}$$

T denotes matrix transposition. $\boldsymbol{\alpha} \in \mathbb{R}^{|\boldsymbol{D}| \times |\boldsymbol{C}|}$ are attention weights of a document representation associated with each code. Softmax_1 is applied to the first dimension of $\boldsymbol{Z}$, ensuring the distribution over locations in a document sum to 1. $\boldsymbol{V}^{\text{word}} \in \mathbb{R}^{|\boldsymbol{C}| \times d}$ is the code-specific document representations at word level.

We concatenate the hidden state of $\overrightarrow{\boldsymbol{H}}$ at the end position of a sentence and the hidden state of $\overleftarrow{\boldsymbol{H}}$ at the start position of a sentence to embed a sentence.

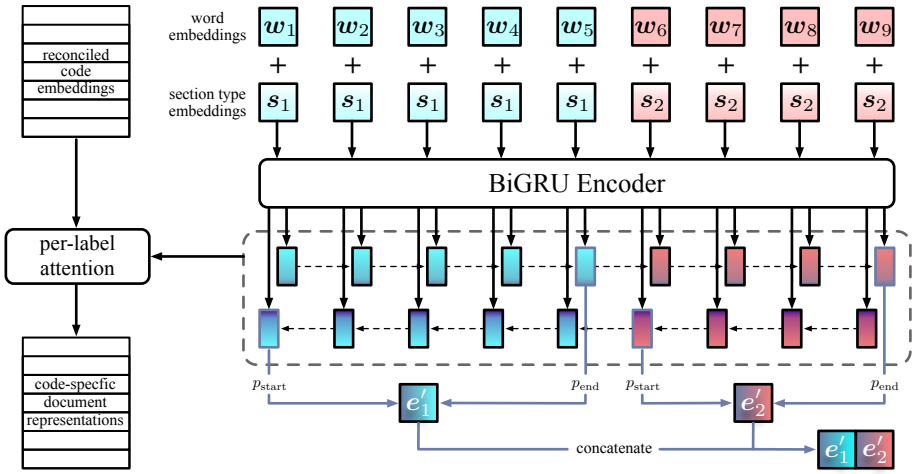

Figure 1: An illustration of how DiscNet works. The sentence embeddings $\boldsymbol{e}'_1$ and $\boldsymbol{e}'_2$ correspond to the input word embeddings $\{\boldsymbol{w}^l_1, ..., \boldsymbol{w}^l_5\}$ and $\{\boldsymbol{w}^l_6, ..., \boldsymbol{w}^l_9\}$ respectively.

Then we concatenate all sentence embeddings:

$$\begin{aligned} \boldsymbol{e}' &= \left(\overrightarrow{\boldsymbol{H}}[p_{\text{end}}] \parallel \overleftarrow{\boldsymbol{H}}[p_{\text{start}}]\right) \boldsymbol{W}_2 \\ \boldsymbol{E}' &= \boldsymbol{e}'_1 \parallel \boldsymbol{e}'_2 \parallel ... \parallel \boldsymbol{e}'_{|P|} \end{aligned} \tag{3}$$

Where $\boldsymbol{E}'$ represents sentence-level features. $\boldsymbol{W}_2 \in \mathbb{R}^{2d \times d}$ is a weight matrix. $|P|$ denotes the number of sentences. We follow the computation step of equation 1 and equation 2 with newly initialized network parameters to obtain $\boldsymbol{V}^{\text{sen}}$, which is the code-specific document representations at sentence level. Finally, we concatenate $\boldsymbol{V}^{\text{word}}$ and $\boldsymbol{V}^{\text{sen}}$. Then a Max Pooling is applied over the level dimension to obtain the condensed code-specific document representations:

$$\boldsymbol{V} = \text{MaxPooling}\left(\boldsymbol{V}^{\text{word}} \parallel \boldsymbol{V}^{\text{sen}}\right) \tag{4}$$

$\boldsymbol{V} \in \mathbb{R}^{|\boldsymbol{C}| \times d}$ is the condensed code-specific document representations.

3.2 Reconciled Embedding (RE)

We focus on the class-imbalanced problem and the heterogeneity between code definitions and relevant documents. The proposed RE approach is designed to reconcile them simultaneously.

3.2.1 Reconciling the heterogeneity

We initialize a new bidirectional GRU to encode the word embeddings of code descriptions. Similar to equation 1, the final hidden states of both directions are extracted and projected. Let $\tilde{\boldsymbol{C}} = \{\tilde{\boldsymbol{c}}_1, \tilde{\boldsymbol{c}}_2, ..., \tilde{\boldsymbol{c}}_{|\boldsymbol{C}|}\} \in \mathbb{R}^{|\boldsymbol{C}| \times d}$ denote the encoded code descriptions. We design a gate mechanism as follows:

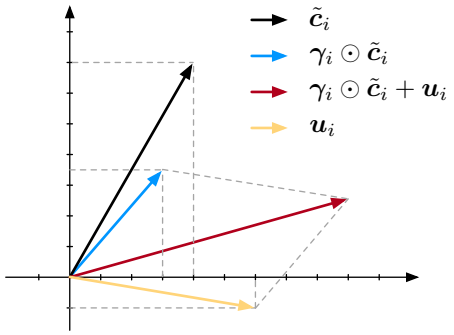

Figure 2: An example of equation 6 in two dimensional space. $\boldsymbol{\gamma}_i$ is set to $[0.75, 0.5]^\top$, $\tilde{\boldsymbol{c}}_i$ is set to $[4.0, 7.0]^\top$, $\boldsymbol{u}_i$ is set to $[6.0, -1.0]^\top$.

$$\begin{aligned} \boldsymbol{Q} &= \text{ReLU}\left(\left(\tilde{\boldsymbol{C}} \parallel \boldsymbol{U}\right) \boldsymbol{W}_3\right) \\ \boldsymbol{\Gamma} &= \text{sigmoid}\left(\boldsymbol{Q}\boldsymbol{W}_4\right) \end{aligned} \tag{5}$$

$\boldsymbol{U} = \{\boldsymbol{u}_1, \boldsymbol{u}_2, ..., \boldsymbol{u}_{|\boldsymbol{C}|}\} \in \mathbb{R}^{|\boldsymbol{C}| \times d}$ is trainable code-specific attention parameters, which are supposed to learn from relevant documents. $\tilde{\boldsymbol{C}}$ and $\boldsymbol{U}$ are concatenated at the last dimension, followed by a linear projection and a ReLU non-linearity. $\boldsymbol{W}_3 \in \mathbb{R}^{2d \times d}$ and $\boldsymbol{W}_4 \in \mathbb{R}^{d \times d}$ are trainable weight matrices. $\boldsymbol{\Gamma} = \{\boldsymbol{\gamma}_1, \boldsymbol{\gamma}_2, ..., \boldsymbol{\gamma}_{|\boldsymbol{C}|}\} \in \mathbb{R}^{|\boldsymbol{C}| \times d}$ is the gate vectors to adjust $\tilde{\boldsymbol{C}}$:

$$\boldsymbol{C} = \text{LayerNorm}\left(\tanh\left(\left(\boldsymbol{\Gamma} \odot \tilde{\boldsymbol{C}} + \boldsymbol{U}\right) \boldsymbol{W}_5\right)\right) \tag{6}$$

Where tanh is the hyperbolic tangent function. LayerNorm denotes layer normalization (Ba et al., 2016). $\odot$ denotes hadamard product. $\boldsymbol{W}_5 \in \mathbb{R}^{d \times d}$ is a trainable weight matrix. $\boldsymbol{C} \in \mathbb{R}^{|\boldsymbol{C}| \times d}$ is the

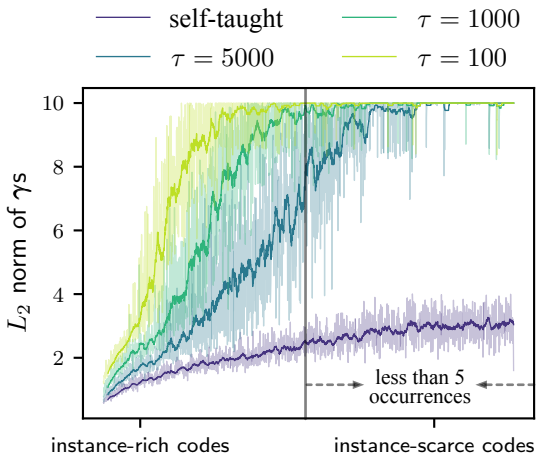

Figure 3: L_2 norm of γs with respect to each code sorted by code occurrences in descending order. A moving average smoothing is applied with a window size equal to 100. A smaller norm indicates less information flow from code descriptions.

reconciled code embeddings to interact with document representations. An example of equation 6 in two dimensional space is given in figure 2.

We suggest the function of the gate mechanism is twofold: Firstly, the element-wise addition of $\tilde{\boldsymbol{C}}$ and $\boldsymbol{U}$ is able to aggregate semantics both from code descriptions and relevant documents. Considering their heterogeneous nature, we leverage the gate vectors $\boldsymbol{\Gamma}$ to adjust $\tilde{\boldsymbol{C}}$, reconciling the semantic discrepancies between $\tilde{\boldsymbol{C}}$ and $\boldsymbol{U}$ during the element-wise addition. Each $\boldsymbol{\gamma}_i$ is able to tune both the length and the direction of $\tilde{\boldsymbol{c}}_i$ within the same quadrant (as shown in figure 2). A naive example is shown in figure 2. Secondly, each $\boldsymbol{\gamma}_i$ is able to scale the information flow from $\tilde{\boldsymbol{c}}_i$. We assume for rare classes, the norm of $\boldsymbol{\gamma}$ is required to be greater than for frequent classes, since it is necessary to learn more from the encoded code definitions for rare classes.

3.2.2 Reconciling the class-imbalance

We assume taking more information from code descriptions than relevant documents could benefit the representation learning of rare codes. But for frequent code, it could be beneficial to take more information from relevant documents than code descriptions. To examine our assumption, we plot the L_2 norm of the trained gate vectors (i.e. $\{\boldsymbol{\gamma}_1, \boldsymbol{\gamma}_2, ..., \boldsymbol{\gamma}_{|\boldsymbol{C}|}\}$) with respect to code occurrences sorted in descending order as shown in figure 3. The observation supports our assumption. The "self-taught" plot in figure 3 indicates that the proportion of the information transferred is related to the code distribution. The $\boldsymbol{\gamma}$s has learned to transfer more information from code descriptions to rare codes.

To better cope with the class imbalanced problem, we have found imposing a regularization on $\boldsymbol{\gamma}$s according to code distribution to be beneficial. We regularize the L_2 norm of each $\boldsymbol{\gamma}$ according to code distribution. The regularization encourages the norm of rare codes' $\boldsymbol{\gamma}$s to be large, forcing them to learn more from the essential definitions in code descriptions. Meanwhile, there's less regularization imposed on frequent codes since they can learn their heterogeneous expressions from relevant documents. Let $K \in \{k_1, k_2, ..., k_{|\boldsymbol{C}|}\}$ denote the number of occurrences of each code. The regularization term is computed as follows:

$$k'_i = \left(\frac{\max(K) - k_i}{\max(K) - \min(K)} \right)^{\tau}$$
$$L_{reg} = -\sum_{i}^{|C|} k'_i \|\boldsymbol{\gamma}_i\|_2 \tag{7}$$

Firstly, a min-max normalization is applied to rescale the range of K in [0, 1]. k'_i is the regularization weights associated with each code. τ is a rescaling hyperparameter. As shown in figure 3, the smaller τ is, the greater the regularization on frequent codes.

3.3 Output Layer

Firstly, $\boldsymbol{V}$ is fed to a linear layer followed by a ReLU non-linearity. Then a dot product of $\boldsymbol{C}$ and $\boldsymbol{V}$ is applied. Finally, a sigmoid activation function is applied to obtain the probability vector:

$$\boldsymbol{V} = \text{ReLU}(\boldsymbol{V}\boldsymbol{W}_6), \quad \hat{\boldsymbol{y}} = \text{sigmoid}\left(\boldsymbol{C}\boldsymbol{V}^{\text{T}}\right) \tag{8}$$

Let $\boldsymbol{y}$ denote the label vector. The code assignment task is treated as a multi-label classification problem. The training objective is to minimize the binary cross-entropy loss and the regularization term from subsection 3.2:

$$\text{Loss}(\boldsymbol{x}, \boldsymbol{y}, \boldsymbol{\theta}) = \text{CrossEntropy}(\boldsymbol{y}, \hat{\boldsymbol{y}}) + \lambda L_{reg} \tag{9}$$

Where $\boldsymbol{x}$ denotes input word tokens and $\boldsymbol{\theta}$ denotes all trainable parameters. λ is a hyperparameter.

Model	AUROC		F1		P@k	
	Macro	Micro	Macro	Micro	8	15
CAML* (Mullenbach et al., 2018)	0.895	0.986	0.088	0.539	0.709	0.561
DR-CAML* (Mullenbach et al., 2018)	0.897	0.985	0.086	0.529	0.690	0.548
MultiResCNN* (Li and Yu, 2020)	0.910	0.986	0.085	0.552	0.734	0.584
HyperCore* (Cao et al., 2020)	0.930	0.989	0.090	0.551	0.722	0.579
DiscNet+RE* (Ours)	**0.945**	**0.991**	**0.137**	**0.579**	**0.760**	**0.608**
MSATT-KG (Xie et al., 2019)	0.910	0.992	0.090	0.553	0.728	0.581
LAAT (Vu et al., 2020)	0.919	0.988	0.099	0.575	0.738	0.591
JointLAAT (Vu et al., 2020)	0.921	0.988	0.107	0.575	0.735	0.590
ISD (Zhou et al., 2021)	0.938	0.990	0.119	0.559	0.745	-
DiscNet+RE (Ours)	**0.956**	**0.993**	**0.140**	**0.588**	**0.765**	**0.614**

Table 2: Experimental results on the MIMIC-III full test set. Models with "*" are under a length limitation of 2,500. Models without "*" are under a length limitation of 4,000. We ran our model 5 times and averaged the scores.

4 Experiments

4.1 Dataset and Preprocessing

We use publicly available and widely studied MIMIC-III dataset (Johnson et al., 2016), which is an extension of the MIMIC-II dataset (Saeed et al., 2011). The dataset comprises de-identified EHR associated with over 40,000 ICU admissions. We follow the well studied MIMIC-III full setting that was initially proposed by Mullenbach et al. (2018), which consists of 8,929 ICD codes, 47,719, 1,631, and 3,372 discharge summaries for training, development, and testing respectively.

To better investigate the performance of our method on codes with different sample sizes, we divide the test set into head, body, and tail subsets. Each subset has the same number of discharge summaries as in the MIMIC-III full test set but with different range of codes. In the head subset there are 1446 distinct codes with sample size greater than or qual to 50. In the body subset there are 1779 distinct codes with sample size less than 50 and greater than 5. In the tail subset there are 860 distinct codes with sample size less than or equal to 5. There are only 4085 distinct codes in total present in the MIMIC-III full test set.

We tokenize the text, then lowercase and lemmatize the words. All numbers are replaced with a "NUM" token. We perform sentence segmentation using spaCy library[1].

4.2 Evaluation Metrics

For a complete comparison with previous studies, we use macro-averaged and micro-averaged F1, macro-averaged and micro-averaged AUC (area under the receiver operating characteristic curve) and P@k (precision at k).

4.3 Hyper-parameter Tuning and Training

The model is trained using Adam optimizer (Kingma and Ba, 2014) and the initial learning rate is set to 0.0005, the batch size is set to 12. The d-dimensional word embeddings are trainable. A dropout mechanism (Srivastava et al., 2014) is applied after each BiGRU with a dropout probability of 0.2. We notice that the model with section type embeddings is more prone to overfitting. Therefore a dropout mechanism is applied on the section type embeddings with a dropout probability of 0.5. τ is set to 1,000, and λ is set to 0.0001.

4.4 Baselines

Some studies truncated the input text to a maximum length of 2,500, the others to a maximum length of 4,000. We have noticed that there are performance differences between different length limitation settings. For a fair comparison, we conducted the experiments under the length limitations of 2,500 and 4,000, then report the results separately.

CAML: The first per-label attention based model for automatic ICD coding preposed by Mullenbach et al. (2018).

DR-CAML: An extension of CAML which incorporates the code descriptions to improve the performance on rarely observed codes. However, DR-CAML performed worse on most metrics than CAML.

MSATT-KG: The MSATT-KG (Xie et al., 2019) approach employed a graph convolutional neural network to capture the hierarchical relationships among codes, alleviating the class imbalanced problem. The study achieved SOTA performance.

HyperCore: Proposed by Cao et al. (2020), which

[1]https://spacy.io/

Model	AUROC		F1		P@k		Macro F1		
	Macro	Micro	Macro	Micro	8	15	head	body	tail
BiGRU	0.904	0.986	0.097	0.562	0.734	0.581	0.369	0.164	0.052
BiGRU+discourse	0.919	0.988	0.115	0.583	0.752	0.601	0.408	0.211	0.063
only DiscNet	0.919	0.988	0.119	0.583	0.757	0.605	0.419	0.216	0.064
only $RE_{\text{self-taught}}$	0.942	0.990	0.126	0.567	0.750	0.595	0.400	0.214	0.086
DiscNet+$RE_{\text{self-taught}}$	0.943	0.990	0.134	0.575	0.756	0.603	0.420	0.235	0.097
DiscNet+$RE_{\text{constant-}\gamma s}$	0.938	0.990	0.129	0.574	0.757	0.603	0.419	0.225	0.082
DiscNet+$RE_{\tau=100}$	**0.947**	**0.991**	0.132	0.578	0.756	0.606	0.420	**0.241**	**0.106**
DiscNet+$RE_{\tau=1000}$	0.945	**0.991**	**0.137**	**0.579**	**0.760**	**0.608**	**0.425**	0.240	**0.106**
DiscNet+$RE_{\tau=5000}$	0.946	**0.991**	0.132	**0.579**	0.756	0.604	0.419	0.229	0.100

Table 3: Ablation results on the MIMIC-III full test set under a length limitation of 2500. We ran each model 5 times and averaged the scores.

can jointly exploit code hierarchy and code co-occurrence. The approach outperformed all existing baseline models.

MultiResCNN: The MultiResCNN (Li and Yu, 2020) leveraged a multi-filter convolutional layer to capture various text patterns.

LAAT: A label attention model was proposed by Vu et al. (2020). LAAT outperformed all existing baseline models.

JointLAAT: An extension of LAAT, which leveraged a hierarchical joint learning mechanism to handle the class imbalanced problem.

ISD: The ISD approach (Zhou et al., 2021) leveraged an interactive shared representation network to alleviate the long-tail problem.

4.5 Compared with Baselines

Table 2 show the experimental results on the MIMIC-III full dataset. Our model outperformed all baseline models across all evaluation metrics and achieves new state-of-the-art results. It is worth noting that the macro-AUROC and macro-F1 were improved by 1.8% and 2.1% compared with the best baseline model. The improvements indicate our model is more robust to infrequent code assignments, since macro-averaging highlights the performance of infrequent classes. Meanwhile, the micro-F1, P@8, and P@15 were improved by 2.9%, 2%, and 2.4% respectively. The results suggest our model improves both macro-averaging and micro-averaging measurements simultaneously.

4.6 Ablation study

We performed an ablation study as shown in table 3 and in figure 4. In order to investigate the effectiveness of our methods on codes with different sample sizes, we use macro-F1 to evaluate each ablation experiment on the head, body, tail subsets of the MIMIC-III test set (refer to subsection 4.1)[2]. For the BiGRU setting, we use a BiGRU to model the input word embeddings, followed by a per-label attention to perform classification. For the BiGRU+discourse setting, we add section type embeddings to the BiGRU setting, achieving 1.8% improvement on Macro-F1 and 2.1% on Micro-F1, significantly improved Macro-F1 on the head subset by 3.9%. The improvements demonstrate the effectiveness of exploiting discourse-level features. For the only DiscNet setting, we add sentence-level features to the BiGRU+discourse setting, The minor improvements indicate the effectiveness of sentence-level features. The only

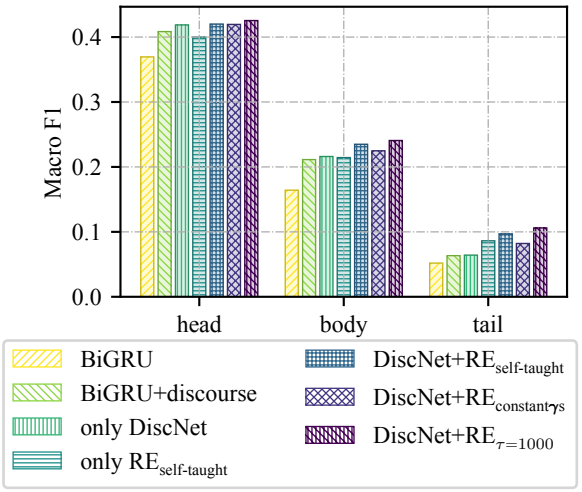

Figure 4: Ablation experiments on head, body and tail subsets demonstrate the improvements of DiscNet on frequent codes and the RE approach on infrequent codes.

[2]In the data splitting setting of Mullenbach et al. (2018), there are only 4,085 distinct codes out of 8,929 present in the test set. The F1 score of a non-appearing code is evaluated to 0. To better compare the F1 score of the head, body, and tail subsets. We evaluate only the 4,085 codes that are present in the test set.

	with section type embeddings	w/o section type embeddings
Case 1	**Brief Hospital Course**: #VB: She had *increased bleeding* ... leading up to presentation... Anemia: Ms. On HD#6, her *hematocrit trended down* to 25 ... ... was transfused 2 units of *red blood cells*... **Discharge Diagnosis**: *Anemia*...	**Past Medical History**: 1.Uterine fibroids 2.*Anemia*, iron-deficiency... **History of Present Illness**: ... with history of *anemia* secondary... **Brief Hospital Course**: ... her *hematocrit trended down* to 25... **Discharge Diagnosis**: *Anemia*...
	285.1 *Acute posthemorrhagic anemia*	~~285.9 *Unspecified anemia*~~
Case 2	**"self-taught"**	**constant γs**
	Smoked 30 yrs, 2 ppd, quit on... ... Patient recently *quit smoking*... We encouraged to continue *smoking abstinence*... He *quit smoking* two week ago due to shortness of breath...	We continued Lisinopril 5mg PO daily... Spiculated pulmonary lesions: consider infectious/inflammatory/neoplastic... In the right, pulmonary artery embolus...
	V15.82 prediction score: 0.70 *personal history of tobacco use*	V15.82 prediction score: 0.02 *personal history of tobacco use*

Table 4: Each example contains a predicted ICD code and relevant document with high attention score.

$\text{RE}_{\text{self-taught}}$ setting, namely the "self-taught" model in figure 3, achieving 3.8% improvement on Macro-AUC, 2.9% on Macro-F1 and 3.4% on Macro-F1 on the tail subset compared to the BiGRU model. The significant improvements demonstrate the effectiveness of RE, particularly on rare codes. For DiscNet+$\text{RE}_{\text{constant}\gamma\text{s}}$ setting, all γs are set equal to 1. The performance drop indicates the effectiveness of the gate mechanism in 3.2.1. We experimented with different τ values and $\tau = 1000$ yields the best results, which demonstrates the effectiveness of the regularization approach in 3.2.2.

4.7 Case Study

To better understand the effectiveness of our approaches, we give examples shown in table 4. We investigate the relevant documents with high attention scores associated with a predicted ICD code. For the first case, the type of *Anemia* is not specified in the discharge diagnosis. The model with section type embeddings correctly linked *bleeding* in the "Brief Hospital Course" section to *anemia*. On the contrary, the unspecified *Anemia* that appears in the "Past Medical History" and the "history of present illness" mislead the baseline model to *285.9 unspecified anemia*. The second example illustrates the impacts of the heterogeneity between code descriptions and relevant documents. The code description *personal history of tobacco use* and relevant document (*smoked, smoking abstinence, and etc*) are literally very different. The "self-taught" model has successfully linked them together. In contrast, the model with constant γs, namely all γs are set equal to 1, failed to highlight any meaningful relevant document.

5 Conclusion

This paper proposed a novel neural architecture for automatic ICD coding. We leverage section type embeddings to make our model discourse-aware. We focus on the class imbalanced problem and the heterogeneity between code definitions and relevant documents. The proposed Reconciled Embedding approach tackled them simultaneously. We achieve state-of-the-art performance on the widely-studied MIMIC-III dataset. DiscNet can be applied to all texts with a discourse structure, but not limited to clinical texts. The proposed reconciled embedding approach can be applied in scenarios where there is auxiliary information associated with labels.

References

Jimmy Lei Ba, Jamie Ryan Kiros, and Geoffrey E Hinton. 2016. Layer normalization. *arXiv preprint arXiv:1607.06450*.

Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. 2014. Neural machine translation by jointly learning to align and translate. *arXiv preprint arXiv:1409.0473*.

Tal Baumel, Jumana Nassour-Kassis, Raphael Cohen, Michael Elhadad, and Noémie Elhadad. 2018. Multi-label classification of patient notes: case study on icd code assignment. In *Workshops at the thirty-second AAAI conference on artificial intelligence*.

Pengfei Cao, Yubo Chen, Kang Liu, Jun Zhao, Shengping Liu, and Weifeng Chong. 2020. Hypercore: Hyperbolic and co-graph representation for automatic

icd coding. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 3105–3114.

Junyoung Chung, Caglar Gulcehre, KyungHyun Cho, and Yoshua Bengio. 2014. Empirical evaluation of gated recurrent neural networks on sequence modeling. *arXiv preprint arXiv:1412.3555*.

Joshua C Denny, Anderson Spickard III, Kevin B Johnson, Neeraja B Peterson, Josh F Peterson, and Randolph A Miller. 2009. Evaluation of a method to identify and categorize section headers in clinical documents. *Journal of the American Medical Informatics Association*, 16(6):806–815.

Alistair EW Johnson, Tom J Pollard, Lu Shen, H Lehman Li-Wei, Mengling Feng, Mohammad Ghassemi, Benjamin Moody, Peter Szolovits, Leo Anthony Celi, and Roger G Mark. 2016. Mimic-iii, a freely accessible critical care database. *Scientific data*, 3(1):1–9.

Shafiq Joty, Giuseppe Carenini, Raymond Ng, and Gabriel Murray. 2019. Discourse analysis and its applications. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics: Tutorial Abstracts*, pages 12–17.

Diederik P Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*.

Dee Lang. 2007. Consultant report-natural language processing in the health care industry. *Cincinnati Children's Hospital Medical Center, Winter*, 6.

Fei Li and Hong Yu. 2020. Icd coding from clinical text using multi-filter residual convolutional neural network. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pages 8180–8187.

Ying Li, Sharon Lipsky Gorman, and Noémie Elhadad. 2010. Section classification in clinical notes using supervised hidden markov model. In *Proceedings of the 1st ACM International Health Informatics Symposium*, pages 744–750.

Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. 2013. Distributed representations of words and phrases and their compositionality. In *Advances in neural information processing systems*, pages 3111–3119.

James Mullenbach, Sarah Wiegreffe, Jon Duke, Jimeng Sun, and Jacob Eisenstein. 2018. Explainable prediction of medical codes from clinical text. *arXiv preprint arXiv:1802.05695*.

Mohammed Saeed, Mauricio Villarroel, Andrew T Reisner, Gari Clifford, Li-Wei Lehman, George Moody, Thomas Heldt, Tin H Kyaw, Benjamin Moody, and Roger G Mark. 2011. Multiparameter intelligent monitoring in intensive care ii (mimic-ii): a public-access intensive care unit database. *Critical care medicine*, 39(5):952.

Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. 2014. Dropout: a simple way to prevent neural networks from overfitting. *The journal of machine learning research*, 15(1):1929–1958.

Simone Teufel, Jean Carletta, and Marc Moens. 1999. An annotation scheme for discourse-level argumentation in research articles. In *Ninth Conference of the European Chapter of the Association for Computational Linguistics*, pages 110–117.

Thanh Vu, Dat Quoc Nguyen, and Anthony Nguyen. 2020. A label attention model for icd coding from clinical text. *arXiv preprint arXiv:2007.06351*.

Pengtao Xie and Eric Xing. 2018. A neural architecture for automated icd coding. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1066–1076.

Xiancheng Xie, Yun Xiong, Philip S Yu, and Yangyong Zhu. 2019. Ehr coding with multi-scale feature attention and structured knowledge graph propagation. In *Proceedings of the 28th ACM International Conference on Information and Knowledge Management*, pages 649–658.

Tong Zhou, Pengfei Cao, Yubo Chen, Kang Liu, Jun Zhao, Kun Niu, Weifeng Chong, and Shengping Liu. 2021. Automatic icd coding via interactive shared representation networks with self-distillation mechanism. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 5948–5957.

Towards Summarizing Healthcare Questions in Low-Resource Setting

Shweta Yadav and **Cornelia Caragea**
Department of Computer Science,
University of Illinois at Chicago, Illinois
{*shwetay, cornelia*}@uic.edu

Abstract

The current advancement in abstractive document summarization depends to a large extent on a considerable amount of human-annotated datasets. However, the creation of large-scale datasets is often not feasible in closed domains, such as medical and healthcare domains, where human annotation requires domain expertise. This paper presents a novel data selection strategy to generate diverse and semantic questions in a low-resource setting with the aim to summarize healthcare questions. Our method exploits the concept of guided semantic-overlap and diversity-based objective functions to optimally select the informative and diverse set of synthetic samples for data augmentation. Our extensive experiments on benchmark healthcare question summarization datasets demonstrate the effectiveness of our proposed data selection strategy by achieving new state-of-the-art results. Our human evaluation shows that our method generates diverse, fluent, and informative summarized questions.

1 Introduction

Online health information search is becoming conventional for more and more consumers every day. A recent survey showed that on average eight million people in the United States seek health-related information on the Internet every day[1]. One challenge towards assisting consumers in their healthcare information search is automatic question understanding. Generally consumers' questions are overly descriptive and include several peripheral information (as shown in Figure-1), which are not necessary to answer questions. Therefore, in this study we tackle the task of consumer health question understanding by summarizing the question.

Automatic text summarization is a non-trivial task in Natural Language Processing (NLP) that aims to generate human-readable, concise text containing salient information of the original document. The recent development in large-scale neural language models (Devlin et al., 2019; Raffel et al., 2020) have led to significant performance on several abstractive summarization task. However, their accuracy is partially due to the availability of large-scale human-annotated training data. Moreover, some domains such as biomedical and medical require domain experts to create high-quality training datasets, which is tedious to create at a large-scale level.

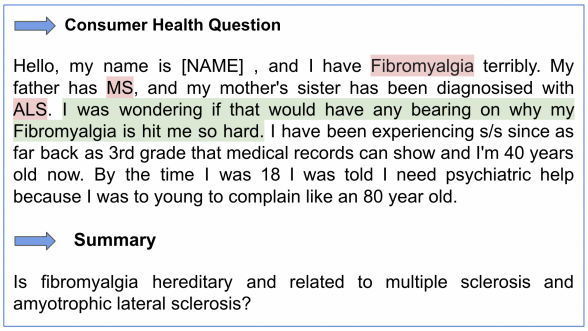

Figure 1: The highlighted text shows important key aspects of the question which need to be considered while generating the summary.

A potential solution that has shown effectiveness in other generation and translation tasks is to augment the large-scale synthetically generated samples with a human-annotated training set. However, a limited study focused on data selection strategy in summarization, particularly for abstractive summarization. The majority of the traditional data selection methods are based on word replacement that mainly generates a synthetic sentence by changing one or multiple words with their synonyms (Zhang et al., 2015) or with a language model predicted words (Kobayashi, 2018). However, these methods make minor changes to the original sentence and therefore fall short of generating a diversified sentence.

To address this research gap, we present a novel data selection strategy for abstractive consumer

[1] https://pewrsr.ch/3l6m3mv

Proceedings of the 29th International Conference on Computational Linguistics, pages 2892–2905
October 12–17, 2022.

health question (CHQ) summarization task. Inspired by the success of the round-trip translation (RTT) (Hoang et al., 2018) – a process of translating the sentences to a pivot language and then back translating to the original language, we aim to explore the effect of RTT as a data augmentation method in CHQ summarization. However, not all the data samples obtained from RTT are diverse and can contain redundant information.

Towards this, we enhance the capability of RTT by devising multiple optimal data selection strategies to select diverse and informative questions, which leads to the significant performance improvement of the CHQ summarization system. Our first data selection strategy **Frechét Question Distance** (**FQD**) is based on Frechét distance (Dowson and Landau, 1982), which measures the distribution distance between the gold and round-trip translated question. The FQD ensures that questions having near similar or very different distributions should not be selected as additional data to train the summarization system. We propose **Precision Recall Question Distance** (**PRQD**) as our second data selection strategy, which disentangles the question distributions divergence into two components: *precision* and *recall*. These two components ensure that the selected additional data brings diversity to the whole training dataset. It is achieved by finding the trade-off between precision and recall of the distributions of the gold and round-trip translated questions. Our final data selection strategy **Question Semantic Volume** (**QSV**) is based on maximizing the semantic area formed by the points obtained from the semantic representation of the questions. The QSV aims to select the questions which maximize the semantic area leading to the selection of the additional questions which are non-redundant and diverse in nature.

We evaluated the effect of the additional data generated using the RTT and our proposed data selection objective measures on benchmark CHQ summarization dataset and two low-resource open domain datasets. We assess the role of each objective measure in RTT based data selection technique using five different pivot languages. Our results show that the RTT-based data selection method helps to improve the performance of the summarization system. We summarize the contribution of the work as follows:

1. We explored the role of the RTT-based data selection technique on CHQ summarization by experimenting with five different pivot languages.
2. We introduced the semantic-volume and diversity-based data selection objective measure in RTT to optimally select the diverse and informative synthetic questions.
3. Our unsupervised method achieves state-of-the-art performance on benchmark consumer healthcare question summarization datasets. Further, our human analysis confirms the effectiveness of our proposed approach in generating fluent and informative summary.

2 Related Work

Neural Abstractive Summarization: The recent advancement of neural networks models, particularly sequence-to-sequence (seq2seq) (Sutskever et al., 2014) models, attention mechanism (Bahdanau et al., 2015), copy mechanism (Gu et al., 2016), coverage mechanism (See et al., 2017) has propelled the development of efficient abstractive summarization approaches on numerous open-domain datasets. Several other methods have utilized the reinforcement learning (RL) (Paulus et al., 2018; Pasunuru and Bansal, 2018; Zhang and Bansal, 2019) to guide the models to generate faithful summaries. Recently, several studies have investigated the pre-trained language models in the abstractive summarization task (Qi et al., 2020; Liu and Lapata, 2019) and have achieved the state-of-the-art performance. Besides the supervised models, various other unsupervised approaches have utilized variational autoencoders for automatic summarization (Laban et al., 2020; Bražinskas et al., 2020; Baziotis et al., 2019).

Consumer Health Question (CHQ) Summarization: While major progress has been made in open-domain abstractive summarization, CHQ summarization is a relatively new task. Ben Abacha and Demner-Fushman (2019) defined the task of summarizing CHQ and introduced a benchmark dataset containing 1000 consumer questions summaries. Recently, a first shared task was organized by Ben Abacha et al. (2021) with the task of summarizing consumer health questions, radiology reports, and multi-document answers. The majority of the works (Lee et al., 2021; He et al., 2021; Sarrouti et al., 2021; Sänger et al., 2021) used pre-trained language models, ensemble approaches, and knowledge-based methods for the CHQ summarization task. A few other new methods (Yadav

et al., 2021a) have enhanced the capability of transformer model by inducing the latent knowledge. In the literature, several works have explored the concept of RTT in machine translation (Nguyen-Son et al., 2021), sentence construction (Zhou et al., 2021), and style-transfer (Zhang et al., 2020b).

Our works advances the existing studies in the consumer health question summarization by proposing an unsupervised framework to optimally select the diverse and information RTT questions, which leads to significant improvement without the need of additional labelled data.

3 Methods

3.1 Background

Given a consumer health question $Q = \{q_1, q_2, \ldots, q_M\}$, the goal of this task is to generate a summarized question $S = \{s_1, s_2, \ldots, s_N\}$ that contains the key information of the original question. Towards this, we build our question summarization model over the Transformer-based seq2seq (Vaswani et al., 2017) architecture. It aims to learn the conditional likelihood $p(S|Q) = \prod_{t=1}^{t=N} p(s_t|s_{<t}, Q)$, where, $s_{<t}$ denotes all generated target tokens before s_t. We utilized the pre-trained ProphetNet (Qi et al., 2020), as the strong base model to summarize the questions. We choose ProphetNet as it is specifically designed for sequence-to-sequence training and it has shown near state-of-the-art results on language generation and CHQ summarization task (Yadav et al., 2021a).

3.2 CHQ Summarization with Round-trip Translation

To train an effective neural network model for language generation tasks, the requirement of sufficient training data is indispensable. Synthetic data augmentation is a way to mitigate the data scarcity issue. It helps the model to reduce the brute-force memorization and also introduce a regularization effect.

In the literature, existing works (Yu et al., 2018; Xie et al., 2020) have shown that the RTT-based data augmentation methods create diverse samples while preserving the semantics. Inspired by these studies, we perform RTT to generate the paraphrases of the source CHQ that could lead to a better summarization system. In order to avoid the noise and keeping the fact intact, we did not paraphrase the gold summarized questions.

Specifically, for a given original dataset $\mathcal{D}_{orig} = \{(Q_i, S_i) \mid i = 1, 2, \ldots, L\}$, we translate the source CHQ $Q_i \in \mathcal{D}_{orig}$ into a non-English pivot language (xx) to obtain $\mathcal{D}^{en \rightarrow xx} = \{(Q_i^{xx}, S_i) \mid i = 1, 2, \ldots, L\}$ using the Google translation. We then back-translate the $\mathcal{D}^{en \rightarrow xx}$ to English and obtained $\mathcal{D}^{xx \rightarrow en}$. The final dataset is obtained from forward ($\mathcal{D}^{en \rightarrow xx}$) followed by the backward ($\mathcal{D}^{xx \rightarrow en}$) translation as:

$$\mathcal{D}_{rtt}^{en \leftrightarrow xx} = \{(\hat{Q}_i, S_i) \mid i = 1, 2, \ldots, L\} \quad (1)$$

Further, to enhance the model's generalization ability, we enrich the original training dataset $\mathcal{D}_{orig}$ with the additional RTT-based generated data $\mathcal{D}_{rtt}^{en \leftrightarrow xx}$. We call this as the augmented dataset $\mathcal{D}_{aug}^{en \leftrightarrow xx}$:

$$\mathcal{D}_{aug}^{en \leftrightarrow xx} = \mathcal{D}_{orig} \cup \mathcal{D}_{rtt}^{en \leftrightarrow xx} \quad (2)$$

3.3 Data Selection Objective Measures

We define three different data selection objective measures: **(i)** Frechét Question Distance, **(ii)** Precision Recall Question Distance, and **(iii)** Question Semantic Volume, that assess both diversity and quality of the RTT question by assigning low scores to less informative questions (*i.e.*, questions having factual errors and lacking salient medical information as present in the original question) or have low-diversity.

In the literature, there are few metrics like BLEU, Self-BLEU, Negative Log-Likelihood (NLL) that individually account for quality and diversity in the generated text. Alihosseini et al. (2019) shows that these metrics neglect either the quality (in the case of Self-BLEU) or the coverage (in metrics like BLEU, NLL). Thus, it is necessary to have a measure that could jointly consider both quality-diversity in the generated text. We argue that the distribution distance between the semantic representations of the round-trip generated question and the original question can be used simultaneously to select the diverse and informative round-trip generated question.

Given a gold question Q and round-trip generated question $\hat{Q}$, we first extract the question semantic representations h_Q and $h_{\hat{Q}}$ from a Transformer-based (Vaswani et al., 2017) pre-trained language model, which encodes the contextual information of the questions. Unlike the other work (Xiang et al., 2021), where the BERT has been used to derive fixed-size sentence embedding, we follow the idea of sentence-BERT (Reimers

and Gurevych, 2019) which uses the siamese and triplet networks (Schroff et al., 2015) to update the weights such that the generated semantically similar question representations are close in vector space. Towards this, we utilized the pre-trained `MPNet` (Song et al., 2020) model, which is fine-tuned using the siamese and triplet networks as discussed in Reimers and Gurevych (2019). We obtain the semantic representation of the questions from fine-tuned `MPNet` as:

$$h_Q = \texttt{MPNet}(q_1, q_2, \ldots, q_M)$$
$$h_{\hat{Q}} = \texttt{MPNet}(\hat{q}_1, \hat{q}_2, \ldots, \hat{q}_{\hat{M}}) \quad (3)$$

In the following sub-sections, we use the semantic representation of the questions to devise multiple objective measures to select the diverse and informative round-trip generated question.

3.3.1 Frechét Question Distance

Heusel et al. (2017) introduced the metric Fréchet Inception Distance (FID) to evaluate the performance of the Generative Adversarial Networks (Goodfellow et al., 2014) based image generation models. FID is based on the Fréchet distance (Dowson and Landau, 1982) and is used to measure the similarity of generated images to real ones. Inspired by FID, we introduce FQD, which measures the distributional distance between the semantic representation of the gold question and the round-trip generated question. We assume that question semantic representations follow the multi-dimensional Gaussian distribution with first two moments: *mean* and *covariance*. The distance between these two Gaussian distributions is measured by the Fréchet distance.

Let the semantic representation h_Q of the gold question follow the Gaussian: $h_Q \sim \mathcal{N}(\boldsymbol{\mu}_q, \boldsymbol{\Sigma}_q)$ with mean $\boldsymbol{\mu}_q$ and co-variance matrix $\boldsymbol{\Sigma}_q$. Similarly, let the semantic representation of the round-trip question follow: $h_{\hat{Q}} \sim \mathcal{N}(\boldsymbol{\mu}_{\hat{q}}, \boldsymbol{\Sigma}_{\hat{q}})$. The Frechét Question Distance between Q and $\hat{Q}$ is computed as follows:

$$d_{\text{FQD}}(Q, \hat{Q}) = \|\boldsymbol{\mu}_q - \boldsymbol{\mu}_{\hat{q}}\|_2^2 + \text{Tr}(\boldsymbol{\Sigma}_q + \boldsymbol{\Sigma}_{\hat{q}} - 2(\boldsymbol{\Sigma}_q \boldsymbol{\Sigma}_{\hat{q}})^{1/2}) \quad (4)$$

where $Tr(X)$ is the trace of matrix X. To produce a uniform FQD score, we linearly scale the $d_{\text{FQD}}(Q, \hat{Q})$ in the range $[0, 1]$ using the following min-max normalization:

$$FQD(Q, \hat{Q}) = \frac{d_{\text{FQD}}(Q, \hat{Q}) - min(d_{\text{FQD}})}{max(d_{\text{FQD}}) - min(d_{\text{FQD}})} \quad (5)$$

where $min(d_{\text{FQD}})$ and $max(d_{\text{FQD}})$ represent the minimum and maximum FQD in the dataset. When the distribution of gold question is close to the distribution of the round-trip generated question, the FQD score is close to zero. In order to have the diverse, informative, and non-redundant samples in the training set, one does not need to include the round-trip generated questions whose FQD scores with gold questions are either low (near same question) or high (entirely different questions). Toward this, we aim to select the round-trip generated questions such that FQD score with gold questions is found to be in an optimal range. Given the round-trip generated questions $\mathcal{D}_{rtt}^{en \leftrightarrow xx}$ with pivot language (xx), we select a subset of the questions as follows:

$$\mathcal{D}_{\texttt{rtt+fqd}}^{en \leftrightarrow xx} = \{(\hat{Q}_i, S_i) \mid \mu_1 < FQD(Q_i, \hat{Q}_i) < \mu_2\} \quad (6)$$

where μ_1 and μ_2 are hyper-parameters (i.e., the optimal threshold) chosen based on the performance of CHQ summarization system on the validation dataset.

3.3.2 Precision Recall Question Distance

Inspired by the work of Sajjadi et al. (2018), which uses the notion of precision and recall to compare the reference and hypothesis distribution, we propose our second objective measure Precision Recall Question Distance. Similar to the FQD, it measures the distributional distance between semantic representations of the gold and round-trip generated questions; however, it does not require estimating the moments of the probability distributions. Intuitively *precision* measures how much of $h_{\hat{Q}}$ can be generated by a portion of h_Q. In contrast, *recall* measures how much of h_Q can be generated by a portion of $h_{\hat{Q}}$. Hence, the precision and recall should be high for the approximately same question distributions, whereas, if the question distributions are disjoint in nature, the precision and recall will be zero. Therefore, we aim to select the RTT questions whose precision and recall lies between the optimal range to ensure diversity. To compute PRQD, we follow the algorithm proposed by Sajjadi et al. (2018), which is based on the precision-recall distance (PRD) curve. Toward this, we compute pairs of precision $prec(\alpha)$ and recall $rec(\alpha)$ for an equiangular grid of values of α.

$$prec(\alpha) = \sum_{\boldsymbol{v} \in \mathcal{V}} \min(\alpha h_Q(\boldsymbol{v}), h_{\hat{Q}}(\boldsymbol{v}))$$
$$rec(\alpha) = \sum_{\boldsymbol{v} \in \mathcal{V}} \min\left(h_Q(\boldsymbol{v}), \frac{h_{\hat{Q}}(\boldsymbol{v})}{\alpha}\right) \quad (7)$$

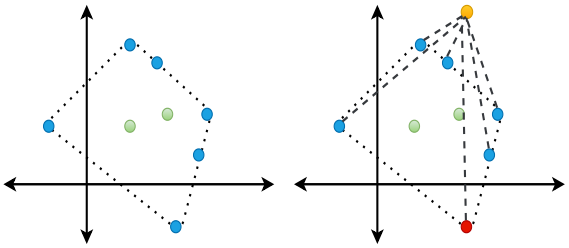

Figure 2: Question semantic volume maximization using convex hull. The ● and ● are the selected and non-selected candidates RTT questions using convex hull. The **left** side figure shows the toy-example of the convex hull. The **right** side figure shows the selected RTT question ● with respect to the gold question ●.

where h_Q and $h_{\hat{Q}}$ probability distributions are defined on a finite state space $\mathcal{V}$. In order to compute a single-value metric, we compute the F1-score corresponding to each α and select the maximum F1-score as the PRQD distance $d_{\text{PRQD}}(\hat{Q}, Q)$ as follows:

$$d_{\text{PRQD}}(\hat{Q}, Q) = \max \left\{ \frac{2 * prec(\alpha) * rec(\alpha)}{prec(\alpha) + rec(\alpha)} \;\middle|\; \alpha \in \Lambda \right\} \tag{8}$$

where $\Lambda = \{\tan(\frac{i}{p+1}\frac{\pi}{2}) | i = 1, \cdots, p\}$ and $p \in \mathbb{N}$ refers to the angular resolution, which is a hyper-parameter. Similar to FQD, we linearly scale the $d_{\text{PRQD}}(\hat{Q}, Q)$ in the range of $[0, 1]$ following Eq. 5 and obtained the normalized score $PRQD(\hat{Q}, Q)$.

Given the round-trip generated questions $\mathcal{D}_{rtt}^{en \leftrightarrow xx}$ with pivot language (xx), we select a subset of questions as follows:

$$\mathcal{D}_{\texttt{rtt+prqd}}^{en \leftrightarrow xx} = \{(\hat{Q}_i, S_i) \mid \beta_1 < PRQD(\hat{Q}_i, Q_i) < \beta_2\} \tag{9}$$

where β_1 and β_2 are the optimal thresholds which are chosen similar to μ_1 and μ_2.

3.3.3 Question Semantic Volume

Existing work in the literature (Yogatama et al., 2015) shows that the sentences which maximize the semantic volume in a distributed semantic space are the most diverse and have least redundant sentences. Motivated by this, first, we aim to find the most diverse and least redundant round-trip generated questions from the pool of RTT questions generated by considering different pivot languages. Later, we devise a simple yet effective measure to quantify the candidate RTT questions with respect to the gold questions in terms of their semantic distance. Specifically, for the given gold question Q and a set of K RTT generated questions $\{\hat{Q}_1, \hat{Q}_2, \ldots, \hat{Q}_K\}$, first, we extract (*cf.* Eq. 3) the semantic representation h_Q for gold question and each RTT questions $\{h_{\hat{Q}_1}, h_{\hat{Q}_2}, \ldots, h_{\hat{Q}_K}\}$ and form a data matrix $H \in \mathcal{R}^{(K+1)\times d}$. Later, we perform the linear dimensionality reduction using Principal Component Analysis to project the data matrix H to a lower dimensional space and obtain the transformed data matrix $\overline{H} \in \mathcal{R}^{(K+1)\times 2}$. In order to find and compare the most diverse round-trip candidate questions, we exclude the point corresponding to the gold question from $\overline{H}$. To find a convex maximum volume, we find the convex hull using the Quickhull algorithm (Barber et al., 1996) as follows:

$$\{p_1, p_2, \ldots, p_C\} = \texttt{ConvexHull}(\overline{h}_1, \overline{h}_2, \ldots, \overline{h}_K) \tag{10}$$

The convex hull are the smallest convex set that includes all points $\overline{h}_1, \overline{h}_2, \ldots, \overline{h}_K$. The points $\{p_1, p_2, \ldots, p_C\}$ are the vertices of the convex hull. It also guarantees to obtain the maximum semantic area with the selected points. Intuitively, it selects the RTT questions which are diverse in nature.

However, the vertices of the convex hull do not reduce the redundant points over the convex hull, and it lacks the notion of semantic distance from the point representing the gold question. Due to this, it usually selects the redundant round-trip generated questions (*cf.* Figure 2). To tackle this, first, we compute the euclidean distance $d(p_g, p_i)$ between the point p_g representing the gold question and each point p_i from the vertices of convex hull. Then, we only select the farthest apart round-trip question $\hat{Q}_j$ to include in the dataset if their semantic point in vector space represented by p_j is greater than an optimal threshold.

$$\begin{aligned} D &= \{d(p_g, p_i) \mid i = 1, 2, \ldots, C\} \\ p_j &= \underset{p_1, p_2, \ldots p_C}{\arg\max}(D) \end{aligned} \tag{11}$$

Finally, we select the optimal subset of the questions as follows:

$$\mathcal{D}_{\texttt{rtt+qsv}} = \{(\hat{Q}_j, S_j) \mid d(p_g, p_j) > \lambda\} \tag{12}$$

where λ is an optimal threshold and chosen based on the performance of CHQ summarization on the validation dataset.

4 Experiments

4.1 Datasets

We experimented with a benchmark CHQ summarization dataset (MEQSUM) (Ben Abacha and

Demner-Fushman, 2019). The MEQSUM dataset consists of domain-expert labeled 1000 question-summary pairs. The dataset is derived from de-identified consumer health questions (CHQs) received by the U.S. National Library of Medicine, National Institute of Health. Similar to the Ben Abacha and Demner-Fushman (2019), we augmented additional $4,655$ pairs of medical questions and shorter questions obtained from (Ely et al., 2000) to the original MEQSUM dataset. We use $5,055$ question-summary pairs as a training dataset, 100 sample pairs for validation, and 500 sample pairs for testing. We also experimented on an additional test collection containing 100 question-summary pairs released in BioNLP 2021 MEDIQA-QS shared task challenge (Ben Abacha et al., 2021) that has the same training set as the MEQSUM dataset.

We evaluated the performance of the proposed models using ROUGE (Lin, 2004). Following the existing works (Fabbri et al., 2021; Yadav et al., 2022b; Gliwa et al., 2019; Yadav et al., 2022a, 2021b), we reported the Rouge-1, Rouge-2, and Rouge-L. Additional implementation details are in the **Appendix**.

4.2 Experimental Setups

We design the following experiments to assess and compare the role of round-trip translation and the proposed data selection objective measures.

1. **Original Data**: We trained the question summarization system with the gold-standard training dataset ($\mathcal{D}_{orig}$) which consist of source question and target summary and evaluated the performance on the test dataset.
2. **RTT**: We augmented the RTT questions with the original data and obtained $\mathcal{D}_{aug}^{en \leftrightarrow xx}$ (*cf.* Eq. 2). We performed this experiment with five different languages (*xx*): Spanish (*es*), German (*de*), Japanese (*ja*), Chinese Simplified (*zh-CN*), and Chinese Traditional (*zh-TW*).
3. **RTT + FQD**: We utilize the FQD based objective measure to select the optimal subset of RTT synthetic questions. The selected synthetic questions ($\mathcal{D}_{\texttt{rtt+fqd}}^{en \leftrightarrow xx}$) with the original questions ($\mathcal{D}_{orig}$) are used to train the question summarization system.
4. **RTT + PRQD**: We use the PRQD based objective measure to select the optimal subset of round-trip translated synthetic questions. Similar to the **RTT + FQD**, we use selected synthetic questions ($\mathcal{D}_{\texttt{rtt+prqd}}^{en \leftrightarrow xx}$) along with the original questions ($\mathcal{D}_{orig}$) to train the question summarization system.
5. **RTT + QSV**: With this experimental setup, we select the optimal subset from round-trip translated synthetic questions based on question semantic volume obtained from the five different languages. We train the system with $\mathcal{D}_{\texttt{rtt+qsv}}$ dataset (*cf.* Eq. 12) along with the original questions ($\mathcal{D}_{orig}$).

4.3 Results

We report the results on MEQSUM datasets in Table 1. The results shows that our proposed method outperforms all the baselines in terms of Rouge-1, Rouge-2 and Rouge-L metrics on MEQSUM. Additionally, we also compared our proposed methods with the state-of-the-art techniques on MEQSUM. As evident from Table 1, our method outperforms the array of existing approaches on both datasets (in term of Rouge-L) without the need for any additional human-annotated training dataset. On MEQSUM, Mrini et al. (2021) obtained the best performance in terms of Rouge-1 and Rouge-2. It is to be noted that (Mrini et al., 2021) performed experiments on large-scale datasets from various healthcare forums which are restricted for data sharing and crawling. Therefore, to not breach the privacy concern of users, we did not considered those datasets for our experiments.

To understand the role of different data selection method, we carried out a deep analysis of the results (*cf.* Table- 2 and Table 7 in Appendix) both in terms of the performance (Rouge-1, Rouge-2, and Rouge-L) and the number of training samples selected. The results show that augmenting data via RTT significantly improves the performance of the model on all the three metrics. Especially with Frechét Question Distance, we achieve the highest Rouge-1, Rouge-2, and Rouge-L scores 46.59, 29.33, and 49.68 respectively. We also observe a similar gain on all the other language pairs with FQD. The FQD proved to be better amongst all the measures as it consider the semantic distance between the gold question and RTT generated question in the distributional space compared to the PRQD which computes a more abstractive distance. The PRQD based objective measure also achieve significant performance improvement over RTT in all five languages. Our final semantic-volume-based objective measure obtained the improvement of $2.35/3.01/2.61$ on Rouge-1/Rouge-2/Rouge-L

Methods	Rouge-1	Rouge-2	Rouge-L
Baseline Methods			
Seq2Seq (Sutskever et al., 2014)	25.28	14.39	24.64
Pointer Generator (PG) (See et al., 2017)	32.41	19.37	36.53
BertSumm (Liu and Lapata, 2019)	26.24	16.20	30.59
T5 (Raffel et al., 2020)	38.92	21.29	40.56
PEGASUS (Zhang et al., 2020a)	39.06	20.18	42.05
$\text{BART}_{\text{LARGE}}$ (Lewis et al., 2020)	42.30	24.83	43.74
ProphetNet (Qi et al., 2020)	43.87	25.99	46.52
State-of-the-art on CHQ Summarization			
PG + Data Augmentation (Ben Abacha and Demner-Fushman, 2019)	44.16	27.64	42.78
BART + Data-Augmented Joint Learning (Mrini et al., 2021)	48.50	29.70	44.90
ProphetNet + RL rewards (Yadav et al., 2021a)	45.52	27.54	48.19
Proposed Method (RTT+FQD)	46.59	29.33	49.68

Table 1: Comparison of our proposed method with the SOTA and other existing methods on the MEQSUM.

Method	Rouge-1	Rouge-2	Rouge-L	% of Additional Samples
Original Data	43.87	25.99	46.52	–
RTT	44.67	27.68	47.34	100
RTT+FQD	46.59	29.33	49.68	13.16
RTT+PRQD	45.48	27.74	48.61	75.31
RTT+QSV	46.22	29	49.13	2.00

Table 2: Performance of proposed methods (best on *es* language) on MEQSUM. The results for remaining languages can be found in **Appendix** (Table 7).

points over the original data based experiment.

Our second set of experiments analyzed the number of training samples selected by different objectives measures. It can be visualized from Table 2 that QSV outperforms the other selection measures by selecting only 2% of the RTT samples and obtaining 46.22, 29 and 49.13 values of Rouge-1, Rouge-2, and Rouge-L respectively. With FQD, we observed a little higher improvement on *de* language and reported 46.5, 29.53, and 49.45 values for Rouge-1, Rouge-2, and Rouge-L by selecting 6.85% of RTT samples.

We also evaluated the performance of our proposed objective measures on the MEDIQA-QS test dataset. Since, the official training data for MEDIQA-QS was MEQSUM annotated questions, we used the best-performing system (across each language) developed on MEQSUM to evaluate the performance (*cf.* Table-3) of each objective measures on MEDIQA-QS test set. The results shows that our proposed approach outperforms the existing methods in terms of Rouge-1 and Rouge-L. This confirms our data selection measures ensure the training samples are diverse in nature which leads to enhanced learning capability of the summarization model.

Methods	Rouge-1	Rouge-2	Rouge-L
Baseline Methods			
T5 (Raffel et al., 2020)	29.6	10.7	26.7
PEGASUS (Zhang et al., 2020a)	31.2	11.8	28.1
BART (Lewis et al., 2020)	28.6	9.8	25.8
ProphetNet (Qi et al., 2020)	30.3	11.1	26.5
Existing Methods			
Adversarial Training (Xu et al., 2021)	34.03	13.98	29.62
Transfer Learning (Lee et al., 2021)	33.52	15.97	30.90
Generative Transformers (Sänger et al., 2021)	33.40	15.99	31.49
Knowledge-based Method (He et al., 2021)	35.14	16.08	31.31
Proposed Methods			
RTT	35.40	15.00	30.80
RTT+FQD	36.80	15.30	32.10
RTT+PRQD	36.20	15.10	31.60
RTT+QSV	36.50	15.40	32.00

Table 3: Comparison of our proposed methods with the best performing models on the MEDIQA-QS test set.

4.4 Discussion

The results thus satisfy our two major claims: **(i)** The data generated using the RTT helps to improve the performance of the CHQ summarization model by a significant margin, and **(ii)** our proposed diversity and semantic-volume-based objective measures are highly effective in filtering out redundant and undesirable RTT questions, which makes the augmented data more informative and helpful in further improving the performance of the system. Amongst all the objective measure QSV select least amount of RTT samples, it is because QSV follow the two-steps (hull formation, maximizing the distance from gold summary) process to evaluate the informative and diverse samples. We analyze the 82.3% samples was excluded at the first step as they do not form the hull.

From the obtained results, FQD can be chosen among the proposed three objective measures. Although the use of FQD does not lead to selection of least training RTT samples, the results obtained by FQD are consistent and are very near to the optimal solution across all the languages. The complexity of FQD lies in estimating the mean and co-variance of the Gaussian. For the PRQD computation, we need to compute multiple precision and recall to form the PRD curve. The computation of precision and recall is computationally intense as the samples should be compared based on statistical regularities, which requires to obtain the histogram over the k-means clustering of the union of two semantic representations as discussed in Sajjadi et al. (2018). For the QSV, we need to obtained multiple (K) round-trip translated questions followed by their 2-d projection using PCA which requires $\mathcal{O}(2 * d^2)$.

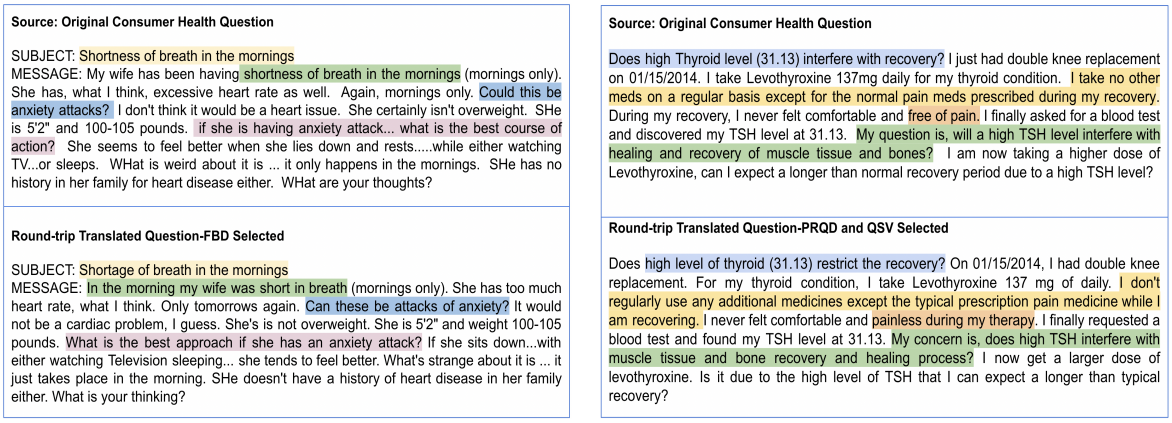

Figure 3: Selected RTT questions with the FQD, PRQD and QSV objective measures.

Thereafter, we need to obtain the convex hull of the projection which requires $\mathcal{O}(KlogK)$. Thus, the PRQD and QSV are computationally intense objectives compare to the FQD.

4.5 Evaluation on Healthcare Answer Retrieval Task

To determine whether the summarized questions can help in improving the answer retrieval performance, we performed experiments on the LiveQA 2017 test set (Abacha et al., 2017), consisting of 104 medical questions from the National Library of Medicine (NLM). The task aims to retrieve a correct answer to each medical question. Towards this, we used our best-performing method (FQD) on the CHQ summarization task to generate a summary for the LiveQA questions. We utilized the answer retrieval model developed in Yadav et al. (2022a) to retrieve the answer from the MedQuad collection[2]. We used the judgment scores[3] established by the LiveQA shared task to judge the quality of retrieved answers: "Correct and Complete Answer" (4), "Correct but Incomplete" (3), "Incorrect but Related" (2) and "Incorrect" (1). We excluded those questions for which the top answer's judgment score was unavailable. In this process, we evaluated common 48 questions for which human judgment scores were available across original questions, model-generated summarized questions and human-generated summarized questions.

Results We used the official evaluation metrics proposed by the LiveQA shared task to compare the performance of answer retrieval using the original versus summarized questions. Please note these metrics evaluate the first retrieved answer for each test question:

- avgScore(0-3): the average score for test questions by transferring 1-4 level grades to 0-3 scores. This is the main score to rank the LiveQA systems.
- succ@k: the number of questions with a score k or above ($k = \{2, 3, 4\}$) divided by the total number of questions in test set.
- prec@k: the number of questions with a score k or above ($k = \{2, 3, 4\}$) divided by the number of questions answered by the system.

Table 4 shows the results obtained by the QA system using: **(i)** the original questions, **(ii)** the summarized questions by FQD, and **(iii)** expert-created reference summaries as reported in (Ben Abacha and Demner-Fushman, 2019).

Measures	Original Questions	Human Generated Reference Summaries	FQD Generated Summarized Questions
avgScore(0-3)	0.384	0.557	0.48
succ@2+	0.23	0.336	0.288
succ@3+	0.115	0.144	0.144
succ@4+	0.038	0.076	0.048
prec@2+	0.5	0.72	0.62
prec@3+	0.25	0.312	0.312
prec@4+	0.083	0.016	0.104

Table 4: Evaluation of the answers retrieved using the original, human-generated, model-generated summaries based on the LiveQA metrics.

The results show that summarizing the CHQ can significantly improve the performance of the IR/QA system in retrieving relevant answers from the collection of curated answers. We also observe that the performance of the IR/QA model using the automatically summarized questions by our proposed approach is close to the performance achieved using the manually created reference summaries.

[2] https://github.com/abachaa/MedQuAD

[3] https://github.com/abachaa/MedQuAD(QA-TestSet)

Methods	RTT Questions			Generated Summary-MEQSUM				Generated Summary-MEDIQA-QS			
	Diversity (DI)	Informative (INF)	Factually Correct (FC)	Incorrect	Acceptable	Perfect	Fluent	Incorrect	Acceptable	Perfect	Fluent
ProphetNet	NA	NA	NA	23	18.5	8.5	23	25	17.5	7.5	26
FQD	38.5	41.5	41	9	12.5	28.5	37	11.5	10.5	28	35
PRQD	35.5	39.5	38.5	11.5	15.5	23	35	12	12	26	30
QSV	37	41.5	40.5	10	13	27	37	9.5	11	29.5	33

Table 5: Human evaluation on selected (50×5 languages) RTT questions. The metrics (DI, INF, FC) shows the average numbers of questions qualified for a given metric across all the 5 languages. The evaluation metrics (*Incorrect*, *Acceptable*, *Perfect*, *Fluent*) shows the average numbers of questions qualified for a given metric.

Original Question-I: SUBJECT: health MESSAGE: I have been bleeding since 2010 and I have been having sharp pain on my left stomach since 2014 and my stomach is so big and I feel weak I have don a lot of test and nothing was seen. What could be wrong with me? And how can I conquer?

Reference: What are the causes of abdominal pain and swelling?

Proposed Approach: What are the causes and treatment for abdominal pain?

Original Question-II: SUBJECT: EPI 743 MESSAGE: My son, His age 4 month discovered it leigh disease infected from the mother side. and we have full family history with the leigh disease. my Daughter she lived for 7 years with the same disease, we have her Hospital reports, it is confirmed leigh Disease. Kindly, if there is any hope for my son with EPI743 treatment, and we are appreciate to accept him in the treatment study. i have a full reports for my son and the MRI... Hope get your help ASAP because he is in the first stage. and we have a we are all hope he will be better.

Reference: Is EPI743 an effective treatment for leigh syndrome?

Proposed Approach: Can leigh disease be treated with EPI743?

Table 6: Generated summaries on the MEQSUM. Example-I shows an acceptable summary and model capability of generating novel words (*"abdominal pain"*) without being present in the original question. The second example shows a semantically correct summary.

4.6 Human Analysis

To understand the role of each data selection measures, we conducted human analysis on randomly selected 50×5 languages samples from RTT datasets. A set of 2 annotators experts in medical informatics evaluated the selected questions on the basis of *diversity*, *informativeness*, and *factual consistency* to measure **(1)** whether the RTT questions have novel n-grams, **(2)** whether the semantics of the original question was retained in the RTT questions and **(3)** whether the salient medical information were present in the selected RTT questions. We also instructed annotators to annotate the generated summaries into one of the following categories: '*Incorrect*', '*Acceptable*', and '*Perfect*' and also report the whether the summary was '*fluent*' or not. We reported the detailed quantitative analysis in Table 5. The results shows that FQD outperforms the other objective measures in terms of selecting more diverse and factually correct questions. Figure 3 shows the de-identified CHQ selected by the different objective measures. In our second analysis on the generated summary (*cf.* Table 5), we again observed the superiority of defined objective measures over the ProphetNet model (trained without the augmented data). This confirms the effectiveness of data selection objective measures that enhance the model learning ability by introducing diverse and informative questions (*cf.* Table 6), leading to the higher proportions of perfect summaries. We also conducted error analysis on generated summaries and identified two main source of errors: **(i)** the original questions consists of multiple sub-questions, and, **(ii)** if the question focus (medical entities) are not transformed into correct medical terminologies.

5 Conclusion

This work propose novel data selection strategy based on the concept of round-trip translation for consumer health question summarization. We devised three major data selection objective measures: FQD, PRQD and QSV based on the distributional distance to optimally select the diverse and informative samples from the pool of round-trip translated data. Extensive experiments show that proposed methods can effectively improve the performance without any additional labelled data. We also achieves new state-of-the-art results on benchmark consumer healthcare question summarization datasets. In future, we plan to explore these objective measures on other resource-scarce tasks.

References

Asma Ben Abacha, Eugene Agichtein, Yuval Pinter, and Dina Demner-Fushman. 2017. Overview of the medical question answering task at trec 2017 liveqa. In *TREC*.

Danial Alihosseini, Ehsan Montahaei, and Mahdieh Soleymani Baghshah. 2019. Jointly measuring diversity and quality in text generation models. In *Proceedings of the Workshop on Methods for Optimizing and*

Evaluating Neural Language Generation, pages 90–98.

Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. 2015. Neural machine translation by jointly learning to align and translate. In *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*.

C Bradford Barber, David P Dobkin, and Hannu Huhdanpaa. 1996. The quickhull algorithm for convex hulls. *ACM Transactions on Mathematical Software (TOMS)*, 22(4):469–483.

Christos Baziotis, Ion Androutsopoulos, Ioannis Konstas, and Alexandros Potamianos. 2019. SEQˆ3: Differentiable sequence-to-sequence-to-sequence autoencoder for unsupervised abstractive sentence compression. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 673–681, Minneapolis, Minnesota. Association for Computational Linguistics.

Asma Ben Abacha and Dina Demner-Fushman. 2019. On the role of question summarization and information source restriction in consumer health question answering. *AMIA Summits on Translational Science Proceedings*, 2019:117.

Asma Ben Abacha and Dina Demner-Fushman. 2019. On the summarization of consumer health questions. In *Proceedings of the 57th Conference of the Association for Computational Linguistics, ACL 2019, Florence, Italy, July 28- August 2, 2019, Volume 1: Long Papers*, pages 2228–2234. Association for Computational Linguistics.

Asma Ben Abacha, Yassine Mrabet, Yuhao Zhang, Chaitanya Shivade, Curtis Langlotz, and Dina Demner-Fushman. 2021. Overview of the MEDIQA 2021 shared task on summarization in the medical domain. In *Proceedings of the 20th Workshop on Biomedical Language Processing*, pages 74–85, Online. Association for Computational Linguistics.

Arthur Bražinskas, Mirella Lapata, and Ivan Titov. 2020. Few-shot learning for opinion summarization. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 4119–4135, Online. Association for Computational Linguistics.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, Minneapolis, Minnesota. Association for Computational Linguistics.

DC Dowson and BV Landau. 1982. The fréchet distance between multivariate normal distributions. *Journal of multivariate analysis*, 12(3):450–455.

John W Ely, Jerome A Osheroff, Paul N Gorman, Mark H Ebell, M Lee Chambliss, Eric A Pifer, and P Zoe Stavri. 2000. A taxonomy of generic clinical questions: classification study. *Bmj*, 321(7258):429–432.

Alexander Richard Fabbri, Faiaz Rahman, Imad Rizvi, Borui Wang, Haoran Li, Yashar Mehdad, and Dragomir Radev. 2021. Convosumm: Conversation summarization benchmark and improved abstractive summarization with argument mining. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 6866–6880.

Bogdan Gliwa, Iwona Mochol, Maciej Biesek, and Aleksander Wawer. 2019. Samsum corpus: A human-annotated dialogue dataset for abstractive summarization. *arXiv preprint arXiv:1911.12237*.

Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. 2014. Generative adversarial nets. volume 27.

Jiatao Gu, Zhengdong Lu, Hang Li, and Victor O.K. Li. 2016. Incorporating copying mechanism in sequence-to-sequence learning. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1631–1640, Berlin, Germany. Association for Computational Linguistics.

Yifan He, Mosha Chen, and Songfang Huang. 2021. damo_nlp at MEDIQA 2021: Knowledge-based preprocessing and coverage-oriented reranking for medical question summarization. In *Proceedings of the 20th Workshop on Biomedical Language Processing*, pages 112–118, Online. Association for Computational Linguistics.

Martin Heusel, Hubert Ramsauer, Thomas Unterthiner, Bernhard Nessler, and Sepp Hochreiter. 2017. Gans trained by a two time-scale update rule converge to a local nash equilibrium. In *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc.

Vu Cong Duy Hoang, Philipp Koehn, Gholamreza Haffari, and Trevor Cohn. 2018. Iterative back-translation for neural machine translation. In *Proceedings of the 2nd Workshop on Neural Machine Translation and Generation*, pages 18–24.

Diederik P. Kingma and Jimmy Ba. 2015. Adam: A method for stochastic optimization. In *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*.

Sosuke Kobayashi. 2018. Contextual augmentation: Data augmentation by words with paradigmatic relations. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 2 (Short Papers)*, pages 452–457, New Orleans, Louisiana. Association for Computational Linguistics.

Philippe Laban, Andrew Hsi, John Canny, and Marti A. Hearst. 2020. The summary loop: Learning to write abstractive summaries without examples. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 5135–5150, Online. Association for Computational Linguistics.

Lung-Hao Lee, Po-Han Chen, Yu-Xiang Zeng, Po-Lei Lee, and Kuo-Kai Shyu. 2021. NCUEE-NLP at MEDIQA 2021: Health question summarization using PEGASUS transformers. In *Proceedings of the 20th Workshop on Biomedical Language Processing*, pages 268–272, Online. Association for Computational Linguistics.

Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Veselin Stoyanov, and Luke Zettlemoyer. 2020. BART: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7871–7880, Online. Association for Computational Linguistics.

Chin-Yew Lin. 2004. ROUGE: A package for automatic evaluation of summaries. In *Text Summarization Branches Out*, pages 74–81, Barcelona, Spain. Association for Computational Linguistics.

Yang Liu and Mirella Lapata. 2019. Text summarization with pretrained encoders. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 3730–3740, Hong Kong, China. Association for Computational Linguistics.

Ilya Loshchilov and Frank Hutter. 2017. SGDR: stochastic gradient descent with warm restarts. In *5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Conference Track Proceedings*. OpenReview.net.

Khalil Mrini, Franck Dernoncourt, Walter Chang, Emilia Farcas, and Ndapandula Nakashole. 2021. Joint summarization-entailment optimization for consumer health question understanding. In *Proceedings of the Second Workshop on Natural Language Processing for Medical Conversations*, pages 58–65.

Hoang-Quoc Nguyen-Son, Tran Thao, Seira Hidano, Ishita Gupta, and Shinsaku Kiyomoto. 2021. Machine translated text detection through text similarity with round-trip translation. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 5792–5797.

Ramakanth Pasunuru and Mohit Bansal. 2018. Multi-reward reinforced summarization with saliency and entailment. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 2 (Short Papers)*, pages 646–653, New Orleans, Louisiana. Association for Computational Linguistics.

Romain Paulus, Caiming Xiong, and Richard Socher. 2018. A deep reinforced model for abstractive summarization. In *6th International Conference on Learning Representations, ICLR 2018, Vancouver, BC, Canada, April 30 - May 3, 2018, Conference Track Proceedings*.

F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. 2011. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830.

Weizhen Qi, Yu Yan, Yeyun Gong, Dayiheng Liu, Nan Duan, Jiusheng Chen, Ruofei Zhang, and Ming Zhou. 2020. ProphetNet: Predicting future n-gram for sequence-to-SequencePre-training. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 2401–2410, Online. Association for Computational Linguistics.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J Liu. 2020. Exploring the limits of transfer learning with a unified text-to-text transformer. *Journal of Machine Learning Research*, 21:1–67.

Nils Reimers and Iryna Gurevych. 2019. Sentence-bert: Sentence embeddings using siamese bert-networks. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 3982–3992.

Mehdi S. M. Sajjadi, Olivier Bachem, Mario Lucic, Olivier Bousquet, and Sylvain Gelly. 2018. Assessing generative models via precision and recall. In *Advances in Neural Information Processing Systems*, volume 31. Curran Associates, Inc.

Mario Sänger, Leon Weber, and Ulf Leser. 2021. WBI at MEDIQA 2021: Summarizing consumer health questions with generative transformers. In *Proceedings of the 20th Workshop on Biomedical Language Processing*, pages 86–95, Online. Association for Computational Linguistics.

Mourad Sarrouti, Deepak Gupta, Asma Ben Abacha, and Dina Demner-Fushman. 2021. Nlm at bioasq synergy 2021: Deep learning-based methods for

biomedical semantic question answering about covid-19.

Florian Schroff, Dmitry Kalenichenko, and James Philbin. 2015. Facenet: A unified embedding for face recognition and clustering. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 815–823.

Abigail See, Peter J. Liu, and Christopher D. Manning. 2017. Get to the point: Summarization with pointer-generator networks. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1073–1083. Association for Computational Linguistics.

Kaitao Song, Xu Tan, Tao Qin, Jianfeng Lu, and Tie-Yan Liu. 2020. Mpnet: Masked and permuted pre-training for language understanding. In *NeurIPS 2020*. ACM.

Ilya Sutskever, Oriol Vinyals, and Quoc V Le. 2014. Sequence to sequence learning with neural networks. In *Advances in Neural Information Processing Systems 27*, pages 3104–3112. Curran Associates, Inc.

Yuqing Tang, Chau Tran, Xian Li, Peng-Jen Chen, Naman Goyal, Vishrav Chaudhary, Jiatao Gu, and Angela Fan. 2021. Multilingual translation from denoising pre-training. In *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*, pages 3450–3466, Online. Association for Computational Linguistics.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in neural information processing systems*, pages 5998–6008.

Pauli Virtanen, Ralf Gommers, Travis E Oliphant, Matt Haberland, Tyler Reddy, David Cournapeau, Evgeni Burovski, Pearu Peterson, Warren Weckesser, Jonathan Bright, et al. 2020. Scipy 1.0: fundamental algorithms for scientific computing in python. *Nature methods*, 17(3):261–272.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Remi Louf, Morgan Funtowicz, Joe Davison, Sam Shleifer, Patrick von Platen, Clara Ma, Yacine Jernite, Julien Plu, Canwen Xu, Teven Le Scao, Sylvain Gugger, Mariama Drame, Quentin Lhoest, and Alexander Rush. 2020. Transformers: State-of-the-art natural language processing. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 38–45, Online. Association for Computational Linguistics.

Jiannan Xiang, Yahui Liu, Deng Cai, Huayang Li, Defu Lian, and Lemao Liu. 2021. Assessing dialogue systems with distribution distances. In *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*, pages 2192–2198, Online. Association for Computational Linguistics.

Qizhe Xie, Zihang Dai, Eduard Hovy, Thang Luong, and Quoc Le. 2020. Unsupervised data augmentation for consistency training. *Advances in Neural Information Processing Systems*, 33.

Liwen Xu, Yan Zhang, Lei Hong, Yi Cai, and Szui Sung. 2021. Chichealth@ mediqa 2021: Exploring the limits of pre-trained seq2seq models for medical summarization. In *Proceedings of the 20th Workshop on Biomedical Language Processing*, pages 263–267.

Shweta Yadav, Deepak Gupta, Asma Ben Abacha, and Dina Demner-Fushman. 2022a. Question-aware transformer models for consumer health question summarization. *Journal of Biomedical Informatics*, 128:104040.

Shweta Yadav, Deepak Gupta, Asma Ben Abacha, and Dina Demner-Fushman. 2021a. Reinforcement learning for abstractive question summarization with question-aware semantic rewards. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing*, pages 249–255.

Shweta Yadav, Deepak Gupta, and Dina Demner-Fushman. 2022b. Chq-summ: A dataset for consumer healthcare question summarization. *arXiv preprint arXiv:2206.06581*.

Shweta Yadav, Mourad Sarrouti, and Deepak Gupta. 2021b. Nlm at mediqa 2021: Transfer learning-based approaches for consumer question and multi-answer summarization. In *Proceedings of the 20th Workshop on Biomedical Language Processing*, pages 291–301.

Dani Yogatama, Fei Liu, and Noah A. Smith. 2015. Extractive summarization by maximizing semantic volume. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 1961–1966, Lisbon, Portugal. Association for Computational Linguistics.

Adams Wei Yu, David Dohan, Minh-Thang Luong, Rui Zhao, Kai Chen, Mohammad Norouzi, and Quoc V Le. 2018. Qanet: Combining local convolution with global self-attention for reading comprehension. In *International Conference on Learning Representations*.

Jingqing Zhang, Yao Zhao, Mohammad Saleh, and Peter Liu. 2020a. PEGASUS: Pre-training with extracted gap-sentences for abstractive summarization. In *Proceedings of the 37th International Conference on Machine Learning*, volume 119 of *Proceedings of Machine Learning Research*, pages 11328–11339. PMLR.

Shiyue Zhang and Mohit Bansal. 2019. Addressing semantic drift in question generation for semi-supervised question answering. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing*

(EMNLP-IJCNLP), pages 2495–2509, Hong Kong, China. Association for Computational Linguistics.

Xiang Zhang, Junbo Zhao, and Yann LeCun. 2015. Character-level convolutional networks for text classification. *Advances in neural information processing systems*, 28:649–657.

Yi Zhang, Tao Ge, and Xu Sun. 2020b. Parallel data augmentation for formality style transfer. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 3221–3228, Online. Association for Computational Linguistics.

Wangchunshu Zhou, Tao Ge, Canwen Xu, Ke Xu, and Furu Wei. 2021. Improving sequence-to-sequence pre-training via sequence span rewriting. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 571–582, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

A Experiments

A.1 Implementation Details

The pre-trained large uncased version[4] of ProphetNet is used as the base encoder-decoder model. We use the fine-tuned verison[5] of MPNet from Huggingface (Wolf et al., 2020) to extract the semantic representation of the questios. We use the Google translation[6] to translate the question into pivot language and then back-translate them into English. To decode the summary, we use beam search algorithm with beam size 4. We fine-tuned the summarization models on the respective training dataset for 15 epochs. The length of maximum original questions and summarized questions are set to 120 and 20, respectively. We choose the optimal value of pairs of hyper-parameters (μ_1, μ_2), (β_1, β_2) and λ using the grid search. We found the optimal pairs (μ_1, μ_2)=$\{(0.17, 0.4),\ (0.25, 0.35),\ (0.05, 0.23),\ (0.19, 0.3),\ (0.04, 0.17)\}$, (β_1, β_2)=$\{(0.3, 0.85),\ (0.3, 0.6),\ (0.3, 0.6),\ (0.55, 0.85),\ (0.4, 0.85)\}$ on languages *es*, *de*, *ja*, *zh-CN* and *zh-TW*, respectively. We obtain the 0.8 as optimal value of λ. To compute PRQD, we follow the official implementation[7] with the hyper-parameter value $p = 1001$. We obtained the first two principal components using the scikit-learn[8] library (Pedregosa et al., 2011). The convex hull is computed using the Sci-py Qhull[9] library (Virtanen et al., 2020). To compute Rouge, we use the py-rouge implementation[10].

To update the model parameters, we used Adam (Kingma and Ba, 2015) optimization algorithm with the learning rate of $7e-5$ in all the experiments. We also used the cosine annealing (Loshchilov and Hutter, 2017) based learning rate decay scheduler, where the learning rate decreases linearly from the initial learning rate in the optimizer to 0.

We have checked for the software usage agreements. The licence details of the used software are as follows: ProphetNet and MPNet Huggingface (Apache-2.0 License), Google Translate (Apache-2.0 License), scikit-learn (BSD-3-Clause License), scipy (BSD-3-Clause License).

Computing Infrastructure: We performed all the experiments on a single NVIDIA Tesla V100 GPU having GPU memory of 32GB.

Average Run Time: The average runtime (for each epoch) to fine-tuned the ProphetNet model on original and RTT augmented datasets are recorded as 10.4 and 20.5 minutes respectively. For the FQD, PRQD and QSV objective based methods the average run time range between 11.5 and 17.2 minutes. It depends upon the number of samples selected for a particular pivot language.

Number of Parameters: The ProphetNet model has 391.32 million parameters. Since, we used the same model for all our experiments there fore we have the same 391.32 million parameters in all variants of the proposed methods.

A.2 Experimental Setups

A.3 Limitation

In this study, we evaluated the model generated summary using Rouge-1, Rouge-2 and Rouge-L metrics. However, these automatic evaluation metrics do not fully capture the nuances of what should or should not be included in a consumer question summary. Although we have performed human evaluation on a subset of summary, it has to be val-

[4] https://huggingface.co/microsoft/prophetnet-large-uncased

[5] https://huggingface.co/sentence-transformers/paraphrase-mpnet-base-v2

[6] We also performed the initial experiment with `mbart-large-50-many-to-many-mmt` (Tang et al., 2021) and found that Google's translation quality was much better than mBART.

[7] https://github.com/msmsajjadi/precision-recall-distributions

[8] https://bit.ly/3DPdjeR

[9] http://www.qhull.org/

[10] https://pypi.org/project/py-rouge/

Method		Rouge-1	Rouge-2	Rouge-L	% of Additional Samples
Original Data		43.87	25.99	46.52	–
es	RTT	44.67	27.68	47.34	100
	RTT+FQD	46.59	29.33	49.68	13.16
	RTT+PRQD	45.48	27.74	48.61	75.31
de	RTT	45.43	29	48.41	100
	RTT+FQD	46.5	29.53	49.45	6.85
	RTT+PRQD	46.38	29.47	49.4	71.27
ja	RTT	45.86	27.8	48.32	100
	RTT+FQD	46.17	29.39	49.48	59.06
	RTT+PRQD	46.02	28.2	49.26	81.34
zh-CN	RTT	44.81	27.71	47.75	100
	RTT+FQD	45.75	28.04	48.71	17.55
	RTT+PRQD	45.66	28.54	48.6	11.36
zh-TW	RTT	45.23	27.76	48.12	100
	RTT+FQD	46.13	28.45	49.16	51.46
	RTT+PRQD	45.88	27.69	48.66	67.02
RTT+QSV		46.22	29	49.13	2.00

Table 7: Performance comparison across all the languages on the proposed methods.

idated by clinical expert on a larger representative collection.

A.4 Potential Risk

The ProphetNet pre-trained language model used in this study are not checked for social bias and diversity. It may not be the representative of the whole world population and may contains region, community, race or gender specific biases.

A.5 Ethics / Impact Statement

Our project involves publicly available datasets of consumer health questions. It does not involve any direct interaction with any individuals or their personally identifiable data and does not meet the Federal definition for human subjects research, specifically: “a systematic investigation designed to contribute to generalizable knowledge" and “research involving interaction with the individual or obtains personally identifiable private information about an individual."

Doc-GCN: Heterogeneous Graph Convolutional Networks for Document Layout Analysis

Siwen Luo[1]*, **Yihao Ding** [1*], **Siqu Long**[1], **Josiah Poon**[1], **Soyeon Caren Han**[1, 2†]
[1]The University of Sydney, [2] The University of Western Australia
{siwen.luo, yihao.ding, josiah.poon, caren.han}@sydney.edu.au,
caren.han@uwa.edu.au, slon6753@uni.sydney.edu.au

Abstract

Recognizing the layout of unstructured digital documents is crucial when parsing the documents into the structured, machine-readable format for downstream applications. Recent studies in Document Layout Analysis usually rely on visual cues to understand documents while ignoring other information, such as contextual information or the relationships between document layout components, which are vital to boost better layout analysis performance. Our Doc-GCN presents an effective way to harmonize and integrate heterogeneous aspects for Document Layout Analysis. We construct different graphs to capture the four main features aspects of document layout components, including syntactic, semantic, density, and appearance features. Then, we apply graph convolutional networks to enhance each aspect of features and apply the node-level pooling for integration. Finally, we concatenate features of all aspects and feed them into the 2-layer MLPs for document layout component classification. Our Doc-GCN achieves state-of-the-art results on three widely used DLA datasets: PubLayNet, FUNSD, and DocBank. The code will be released at https://github.com/adlnlp/doc_gcn

1 Introduction

Digital documents (incl. Scanned Document Images and PDF files) are popular and convenient for storing written textual information, so almost 2.5 trillion documents worldwide are available in the digital format (Zhong et al., 2019). However, it is challenging to automatically recognize the layout and components of these unstructured digital documents and extract meaningful information using this format. For example, the financial office would require the scanned document image after their client signed. It is then crucial to recognize and extract the form component, such as the form title, person name, and the date the document is signed. This task is widely called Document Layout Analysis (DLA). The DLA task aims at understanding the documents from either 1) the physical analysis by detecting the document structure and the boundaries of each layout region or 2) the logical analysis by categorizing the detected layout components (segments) into the predefined document element classes, such as Title, Date, Author, and Figure (Binmakhashen and Mahmoud, 2019). In this research, we focus on the logical DLA task to classify the different layout components of PDF documents by understanding the relationships between components. Traditional deep learning-based DLA approaches mainly focus on processing visual features of layout components (Soto and Yoo, 2019; Augusto Borges Oliveira and Palhares Viana, 2017; Li et al., 2020a) using CNN-based models. Some recent studies started to use texts to solve the problem with the support of semantic information for each layout component (Li et al., 2020b; Xu et al., 2020). However, applying visual and textual features is not enough to analyze the characteristics and relations of document layout components in order to classify them. In this paper, we try to fill this gap by defining and proposing: 1) Layout Components Characteristic Representation and 2) Relation Representation between components.

The first question would be *'What would be the best aspects to represent the characteristics of different document layout components?'*. The text density/sparsity in each document component is a valuable feature. For example, a *paragraph* is more dense and usually contains more texts than a *table*. Moreover, syntactic information can be a key characteristic. It is obvious that a *title* mainly consists of noun phrases only, whereas a paragraph contains more sentences with the noun and verb phrases. Hence, in this research, apart from the common visual features and semantic text features

*Co-First Authors
†Corresponding Author (caren.han@sydney.edu.au)

Proceedings of the 29th International Conference on Computational Linguistics, pages 2906–2916
October 12–17, 2022.

used by previous works, we propose the four major aspects, including text density, components appearance, syntactic and semantic information of textual contents, of each component in order to let the model conduct more comprehensive learning of the Documents properties. Another question would be *'How to represent the relation among document layout components?'* Most DLA studies did not apply multiple aspects of features. Even if applied (Soto and Yoo, 2019; Xu et al., 2020), those features are integrated via simple concatenations and do not consider the influences of relationships between layout components on the classification performances. The characteristics of each component/segment are not enough to analyze the whole document layout and its corresponding components. Assume a 'text' component is detected based on its characteristics, but it is more accurate to classify it as the 'figure caption' if it is right above or below a figure. Thus, capturing and encoding the relationships between layout components is crucial for better layout analysis.

In this paper, we propose a novel Heterogeneous Graph Convolutional Network (GCN)-based DLA model, Doc-GCN, on a document page level, taking the document layout component (segment) as nodes in the graph and encoding the relative positional and structural relations between layout components. We first construct six different graphs, each encoding one aspect of features among the layout components' syntactic, semantic, text density, and visual features. The syntactic and density aspects have two graph variants based on the different node embedding initialization methods. We use the GCN to update the node embedding by learning from its intimate neighbors, and the node-level pooling is then applied to integrate the graph variants. We concatenate the updated features of each aspect, getting the final layout component representation for the final classification of layout component types.

In summary, the contributions of our work are as follows:

- To the best of our knowledge, this is the first attempt to apply heterogeneous aspects of Document Layout Analysis.
- Doc-GCN is the first to propose using multi-aspect Graph Convolutional Networks for harmonizing the characteristics and relationships among document layout components (segments).
- Doc-GCN achieved the state-of-the-art performance on three widely used DLA datasets, PubLayNet, FUNSD, and DocBank.

2 Related Work

2.1 Document Layout Analysis

In the 1990s, rule-based methods (Klink and Kieninger, 2001; Fisher et al., 1990; Niyogi and Srihari, 1986) were widely used for the DLA tasks until the rise of deep learning. Zhong et al. (2019) used Faster RCNN (Ren et al., 2015) and Mask RCNN (He et al., 2017) as the basic deep learning models for DLA task. Recent works have extensions. Soto and Yoo (2019) added the size of the proposed Region of Interest (ROI) and normalized page number as the additional contextual information to the pooled feature vectors for both classification and regression network of Faster RCNN. Augusto Borges Oliveira and Palhares Viana (2017) proposed the 1-D CNN with the parallel operation of horizontal and vertical projection. Li et al. (2020a) proposed a model based on the Feature Pyramid Networks (FPN) object detector to solve the cross-domain document layout classifications.

Some DLA works integrated not only visual features but also textual features. Xu et al. (2020) proposed a pretrained model that integrated each token's positional and text embeddings with the corresponding image embeddings. Xu et al. (2021) then applied additional pretraining tasks to enhance the multi-modality interactions further and used a spatial-aware attention mechanism to capture the relative positional relationship between different layout components. Gu et al. (2021) proposed the pretraining framework with a cross-attention transformer to boost the more substantial alignment between visual and textual features for each document element region. Li et al. (2020b) proposed a new dataset on the token-level where each token is annotated into a layout element class and experimented with this dataset with pretrained language models: BERT (Devlin et al., 2019) and RoBERTa (Liu et al., 2019) by inputting the sequence of token embeddings with the corresponding bounding boxes. Zhang et al. (2021) used a relation module upon the multimodal representations integrated from vision and semantic features to detect the relations between different components.

However, understanding the vision and semantic aspects of individual objects/regions is not enough to analyze the document layout and components.

It is critical to consider the influences of relationships between document layout components. This is very similar to the trend that we can see in the most visual-language tasks (Long et al., 2022a). For example, the table caption should be around the table, and each component should be semantically related. Besides, it is also important to provide the better interpretation of the DLA prediction semantically, which has been considered as lot of NLP tasks (Luo et al., 2021).

2.2 Graph Convolutional Networks

Graph Convolutional Networks (GCN) (Kipf and Welling, 2016) is a type of Graph Neural Network which applies convolution over graph structures, and it has been applied to many Natural Language Processing (NLP) tasks. For example, TextGCN (Yao et al., 2019) and MEGCN (Wang et al., 2022a) focus on the text classification by representing words and documents as graph nodes, and TensorGCN (Liu et al., 2020b) captures their relations in different aspects, including semantic, syntactic, and sequential aspects. D-GCN (Chen et al., 2020) performs sentiment analysis jointly with aspect extraction for graph-based modeling. Hier-GCN (Cai et al., 2020) proposes a hierarchical GCN to model the inner- and inter-relations among multiple aspect categories and sentiments. InducT-GCN (Wang et al., 2022b) enables the inductive GCN learning model, which improves the performance and reduces the time complexicity.

Such graph-based approach is also widely applied to multimodal tasks, especially for visual question answering (Huang et al., 2020; Luo et al., 2020), text-to-image generation (Johnson et al., 2018; Han et al., 2020), and text-image matching (Liu et al., 2020a; Long et al., 2022b). It receives lots of attention in converting multiple modalities and aspects into structured graphs and enhancing joint learning. Some document-based analysis works, such as document dating (Vashishth et al., 2018), apply the GCN-based document dating approach by jointly exploiting the document's syntactic and temporal graph structures. (Wang et al., 2022c) also uses two-stage GCN classifiers for line splitting and clustering for paragraph recognition in documents. In this work, we apply GCN to the DLA task by joint-learning different aspects of document layouts and capturing the relationship between layout components.

3 Doc-GCN

We propose a graph-based network Doc-GCN to encode and integrate the different aspects of document layout components. For each PDF page with N document layout components, we construct six different graphs: $\mathcal{G}_{den1}$, $\mathcal{G}_{den2}$, $\mathcal{G}_{appr}$, $\mathcal{G}_{syn1}$, $\mathcal{G}_{syn2}$ and $\mathcal{G}_{semc}$, capturing the features of all the document layout components in the page from four aspects: density, appearance, syntactic and semantic. Each graph $\mathcal{G}_i = (\mathcal{V}_i, \mathcal{E}_i, \mathcal{A}_i)$ consists of a set of nodes $\mathcal{V}_i$ ($|\mathcal{V}_i| = N$), a set of edges $\mathcal{E}_i$, and an adjacency matrix $\mathcal{A}_i \in \mathbb{R}^{N \times N}$. We regard each layout component of the page as a node $v_n \in \mathcal{V}_i$ in a graph. We segment each layout component by its bounding box. Different node embedding initialization and edge connection methods are applied for different graphs to match the characteristics of different feature aspects and capture different node relationships. The Graph Convolution Network (GCN) is then trained to update the node embeddings by learning from the neighbors.

3.1 Preliminaries

GCN (Kipf and Welling, 2016) is a convolutional neural network that operates directly on a graph to update the node embeddings by learning from the neighbors of each node. Given an initial node embedding matrix $\mathcal{H}^0 \in \mathcal{R}^{N \times d_0}$ consisting of N node features of size d_0, GCN will conduct the propagation through layers based on the rule in Equation 1.

$$\mathcal{H}^{l+1} = f\left(\mathcal{H}^l, \mathcal{A}\right) = \sigma\left(\hat{\mathcal{A}}\mathcal{H}^l W^l\right) \quad (1)$$

The node embedding matrix will be updated from $\mathcal{H}^l \in \mathbb{R}^{N \times d_l}$ to $\mathcal{H}^{l+1} \in \mathbb{R}^{N \times d_{l+1}}$ after every GCN layer, where $l = 1, 2, ..., L$ indicates the layer number. $\hat{\mathcal{A}} = \widetilde{\mathcal{D}}^{-\frac{1}{2}}\widetilde{\mathcal{A}}\widetilde{\mathcal{D}}^{\frac{1}{2}}$ represents the normalized symmetric adjacency matrix where $\widetilde{\mathcal{A}} = \mathcal{A} + \mathcal{I}$ and $\mathcal{I}$ is the identity matrix for self-connection inclusion. $\widetilde{\mathcal{D}}$ is the diagonal node degree matrix with $\widetilde{\mathcal{D}}(i,i) = \Sigma_j \widetilde{\mathcal{A}}(i,j)$ and $W^l \in \mathcal{R}^{d_l \times d_{l+1}}$ is the trainable weight matrix associated with the l-th layer. σ denotes the activation function that can be different for different GCN layers.

3.2 Graphs Construction

3.2.1 Density-aspect Graph[1]

Text density can be a valuable characteristic in distinguishing different document layout components.

[1]Details of density and appearance graph construction procedure can be found in Appendix A.1

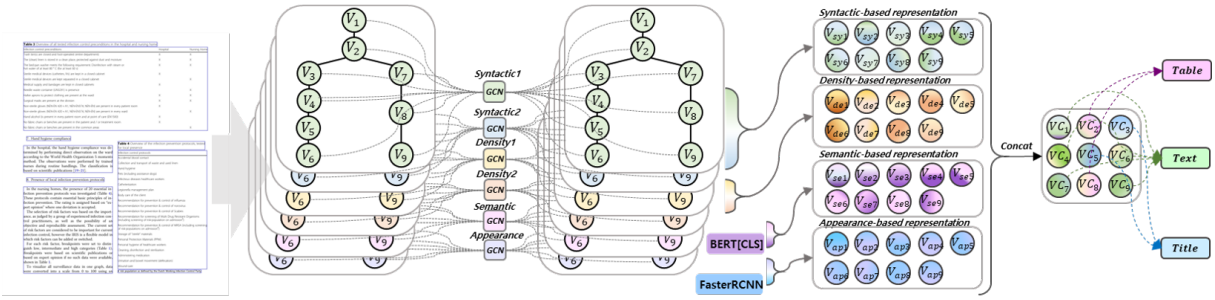

Figure 1: The overall architecture of the proposed Doc-GCN for document layout component classification.

For example, a paragraph usually contains more compact texts than a table with text in a sparser distribution. We construct two density-aspect graphs $\mathcal{G}_{den1}$ and $\mathcal{G}_{den2}$ to encode the text density features of each layout component based on the text density ratio and the absolute character numbers respectively.

We calculate the text density ratio for each layout component as the division between the number of character-level tokens it contains and the area size of its bounding box as in Equation 2:

$$\text{Ratio}_{density} = \frac{\text{\#tokens}}{\text{Area size of bbox}} \quad (2)$$

Given a bounding box $bbox_n$ of a layout component with the top left coordinates (x_1^n, y_1^n) and the bottom right coordinates (x_2^n, y_2^n), the area size is calculated as $(x_2^n - x_1^n) \times (y_2^n - y_1^n)$. We use the positional encoding approach (Vaswani et al., 2017) to project the density ratio and the absolute character numbers of each layout component into a higher dimension $d_0 = 768$ to get the initialized node embedding for each density graph. The initial node embeddings of all N layout components in the document page form the initial node embedding matrices of $\mathcal{G}_{den1}$ and $\mathcal{G}_{den2}$. As per Equation 3, t denotes the value of density ratio or the absolute character numbers of each node, and i is the ith dimension of $\overrightarrow{f_n}$. The value at ith dimension would change along with the odevity of i for any offset k.

$$\overrightarrow{f_n^i} = \begin{cases} sin(\frac{t}{10000^{2k/d_S}}), i = 2k \\ cos(\frac{t}{10000^{2k/d_S}}), i = 2k+1 \end{cases} \quad (3)$$

We connect each node with its closest two neighbors with the smallest gap distance between their corresponding boxes. The edge weight is set to be the inverse distance value to emphasize the positional relationship between closer nodes. For a node v_n with bounding box $bbox_n$ ($[(x_1^n, y_1^n), (x_2^n, y_2^n)]$), we calculate its vertical distance values with other bounding box $bbox_m$ ($[(x_1^m, y_1^m), (x_2^m, y_2^m)]$) that is vertically under it by $D_V = |y_1^m - y_2^n|$ or $D_V = |y_2^m - y_1^n|$ for bounding box that is vertically above it. This yields a set of distance values $\{D_V^1, ..., D_V^m\}$, we connect the node v_n with other two nodes that have the two smallest values D_V. For a two-column PDF page, in addition to the set of D_V, we also calculate the horizontal distance value $D_H = \left|x_1^j - x_2^n\right|$ between $bbox_n$ and a horizontally aligned bounding box $bbox_j$ with coordinates $\left[\left(x_1^j, y_1^j\right), \left(x_2^j, y_2^j\right)\right]$ that has the smallest vertical gap distance. We then connect the node v_n with the other two nodes with the smallest distance values among the set of D_V and D_H.

3.2.2 Appearance-aspect Graph

To learn and encode the appearance properties, such as the color and font size, of layout components in a PDF page, we use a pretrained Faster-RCNN model to extract the appearance feature $\overrightarrow{f_{appr}}$ based on the bounding box of layout component as the initial embedding for each node in $\mathcal{G}_{appr}$. We apply the same method of the edge connection and edge value setup as that for the density-aspect graphs in Section 3.2.1.

3.2.3 Syntactic-aspect Graph[2]

To comprehensively encode the syntactic features, we use the constituency parser (Kitaev and Klein, 2018) to extract both the first-level and second-level syntactic parse of the texts in each layout component and construct two variants of the syntactic-aspect graphs: $\mathcal{G}_{syn1}$ and $\mathcal{G}_{syn2}$ respectively. The first-level parse for each layout component contains only one syntactic symbol, while the second-level parse could be a sequence of different syntactic

[2]Details of syntactic and semantic graph generation can be found in Appendix A.2

symbols. For example, for an article title *'Vitrification preserves chromatin integrity, bioenergy potential and oxidative parameters in mouse embryos'*, the first-level parse is *(S)* and the second-level parse is a sequence of *(NP, VP)*. We use the same positional encoding as in Equation 3 to project a 768-dim vector $\overrightarrow{S_w}$ for each syntactic symbol based on their unique indexes, resulting in a sequence of syntactic embeddings $\overrightarrow{S_1}, \overrightarrow{S_2}, ..., \overrightarrow{S_w}$ for the sequence of parse for each layout component. We then feed this syntactic embedding sequence to a single-layer Bi-LSTM, and extract the last hidden state as the initialized node embedding for $\mathcal{G}_{syn2}$. Since there is only one syntactic symbol for the texts in each layout component for the first-level parse, we pad the single symbol embedding to the sequence of length T for Bi-LSTM input, and get the initialized node embeddings for $\mathcal{G}_{syn1}$. For both $\mathcal{G}_{syn1}$ and $\mathcal{G}_{syn2}$, we connect every two nodes in the graph and set binary edge value $\{0, 1\}$ based on the existence of a parent-child relationship between every two layout components.

For the training and validation, we use the parent-child relations extracted from the document source files provided in the datasets. We apply the OCR detection to the cropped image of each document layout component. Based on the detected OCR tokens and texts in document source files, we use fuzzy string matching and a reading order assigning method proposed by (Ding et al., 2022) to map each layout component with the corresponding element in the *XML*/ LaTeX source files for PubLayNet/DocBank and identify the parent-child relations based on the hierarchical structure embodied in these source files. We then train a transformer-based relation prediction model on the training and validation set utilizing those extracted parent-child relations and predict the parent-child relations for the test set [3].

3.2.4 Semantic-aspect Graph

We use the pretrained BERT model (Devlin et al., 2019) to encode the semantic features of each layout component to construct $\mathcal{G}_{semc}$ and extract the hidden state of the special token $[CLS]$, $\overrightarrow{f_{semc}}$, as the initial node embedding of $\mathcal{G}_{semc}$. We apply the same method for the edge connection and edge value setup as for the syntactic-aspect graphs in Section 3.2.3.

[3]Details of the parent-child relation extraction and transformer-based relation prediction model are provided in Appendix A.2 and Appendix A.3

3.3 Graph Embedding Learning

After the graph construction, we apply the GCN learning on each graph to update the node representations that preserve the four aspects of the layout components' properties by learning and integrating information from the neighbor nodes. For each graph $\mathcal{G}_i^p, i \in \{den1, den2, appr, syn1, syn2, semc\}$ where p denotes the individual PDF page of the dataset, there is an associated initial node embedding matrix $\mathcal{H}_i^0 \in \mathbb{R}^{N \times d_0}$, where $i \in \{den1, den2, appr, syn1, syn2, semc\}$ as defined in Section 3.2.

We feed these initialized node embedding matrices to GCN and update the weight matrices by optimizing the category prediction of each component node following the propagation rules as in Equation 1. After the training, we again feed each graph $\mathcal{G}_i^p$ into the trained GCN and extract the hidden layer node representations out, resulting in an updated node embedding matrix $\mathcal{O}_i^p \in \mathcal{R}^{N \times d}$ for each graph $\mathcal{G}_i^p$, where N denotes the number of components in this PDF page and d is the dimension of each updated node embedding.

3.4 Multi-aspect Classification

To synthesize the six graphs to four graphs with each representing one aspect of the layout components, we apply a node-level max pooling P_m over the updated node embedding matrix $\mathcal{O}_i^p \in \mathcal{R}^{N \times d}$ of the graph variants for each type of the graphs, yielding the new node embedding matrix $\mathcal{O}_a^p$ of each aspect a, where $a \in \{density, appearance, syntactic, semantic\}$.

$$\mathcal{O}_a^p = \begin{cases} P_m(O_{syn1}^p, O_{syn2}^p), a = syntactic \\ P_m(O_{semc}^p, FC(\mathcal{H}_{semc\prime}^{0,p})), a = semantic \\ P_m(O_{den1}^p, O_{den2}^p), a = density \\ P_m(O_{appr}^p, FC(\mathcal{H}_{appr}^{0,p})), a = appearance \end{cases} \tag{4}$$

Specifically, as per Equation 4, for syntactic-aspect and density-aspect graphs, we conduct the max pooling over the learned node representations O_{syn1}^p (O_{den1}^p) and O_{syn2}^p (O_{den2}^p). For semantic-aspect and appearance-aspect graphs that consists of only one graph variant, we apply the node-level max pooling over the learned node embedding matrix O_{appr}^p and O_{semc}^p respectively with the initial node embedding matrix $\mathcal{H}_{appr}^{0,p}$ and $\mathcal{H}_{semc\prime}^{0,p}$ that contains the fine-tuned $\overrightarrow{f_{semc}}$, where $\mathcal{H}_{appr}^{0,p} \in \mathbb{R}^{N \times d_0}$ and $\mathcal{H}_{semc\prime}^{0,p} \in \mathbb{R}^{N \times d_0}$ are projected into a d-dimensional features via fully connected layer FC

first. We then concatenate each of the ultimate aspects features $\mathcal{O}_a^p$ and feed it to the 2-layer MLPs for the final classification of each document layout component in each PDF page. Our classifier is optimized based on the standard $CrossEntropy$.

4 Evaluation Setup

4.1 Datasets

We conducted our experiments on three publicly available document layout analysis datasets: PubLayNet (Zhong et al., 2019), FUNSD (Jaume et al., 2019) and DocBank (Li et al., 2020b) to evaluate our model applicability. We adopted the same train/val/test split as the original dataset.[4]

PubLayNet annotates five different categories of layout components: *Text, Title, List, Table* and *Figure*, for the 358,353 PDF document images in total (94%/3%/3%) collected from the PubMed.

FUNSD is a much smaller dataset extracted from the RVL-CDIP dataset (Harley et al., 2015) and contains 199 scanned PDF pages (75%/0/25%) of survey forms with only 4 different types of layout components: *Header, Question, Answer* and *Other*.

DocBank has more sophisticated annotations for layout components than PubLayNet. It contains 500K PDF Document pages in total (80%/10%/10%)[5] with 13 different categories: *Abstract, Author, Caption, Date, Equation, Figure, Footer, List, Paragraph, Reference, Section, Table* and *Title*. The PDF files of DocBank are collected from arXiv.com with their LaTeX source.

Both **PubLayNet** and **FUNSD** include document page images, so it requires OCR[6] in order to extract the texts for each layout component. We consider those two datasets as Image-based DLA. However, **DocBank** datasets contain PDF files with text source, which contains the text for each document, so we consider this as PDF file-based DLA. We test our Doc-GCN model with both Image-based and PDF-based DLA datasets.

4.2 Baselines

We compare our model with four widely-used Document Layout Analysis baselines.

Faster-RCNN (Ren et al., 2015) is an object detection model that unifies the region proposal network and the Fast R-CNN to extract the visual features from the proposed object regions for image object classification. For the FUNSD and DocBank, we fine-tuned the Faster-RCNN pretrained on the ImageNet with our training set and evaluated the test results. For the PubLayNet, we directly apply their Pretrained Faster-RCNN [7].

BERT (Devlin et al., 2019) is a language model which regards the DLA task as a text classification task by predicting the category of each layout component based on their sequence of text contents.

RoBERTa (Liu et al., 2019) has the same structure as BERT but is pretrained longer on the larger corpus that contains longer sequences. RoBERTa also applies a more dynamic masking pattern for masked language model tasks for pre-training.

LayoutLM (Xu et al., 2020) is a pretrained model that uses the BERT architecture to jointly learn visual aspects (position) and textual features of document layouts.

4.3 Implementation Details

For node features of Semantic graph and Appearance graph, we extract $[CLS]$ encoding ($dim = 768$) from pre-trained $BERT_{BASE}$. The Faster-RCNN with ResNet-101 pre-trained on ImageNet is used to extract the visual features ($dim = 2048$) to initialize the node embedding of the appearance graph. We set $T = 16$ for the sequence padding in syntactic graph construction. All the graphs are trained on 2-layer GCNs for 10 epochs with Adam optimizer using a learning rate of 1×10^{-4} for Semantic and Syntactic Graph and a learning rate of 0.001 for the other two graphs. We use $ReLu$ as the activation function for the first GCN layer, from which we extract our learned node representations. For final classification, we train the multi-aspect classifier with Adam optimizer using the learning rate of 2×10^{-5}, dropout of 0.1 and $Tanh$ as activation function. We trained the model using Intel(R) Xeon(R) CPU @ 2.00GHz and NVIDIA Tesla K80 24GB, which took around 15 hours, 72 hours and 8 mins[8] for training PubLayNet, DocBank and FUNSD respectively. The number of trainable parameters is 126,401,796.

[4] The ratio of official data split can be found in each dataset description (train/val/test)

[5] DocBank only provides the data split ratio and the entire dataset. We split the entire dataset using the same split ratio.

[6] We applied Google Vision API to PubLayNet dataset, and directly used OCR result from FUNSD dataset

[7] PubLayNet Faster-RCNN: https://github.com/ibm-aur-nlp/PubLayNet/

[8] With more aspects covered, Doc-GCN training is still slightly smaller than other baselines recorded in each paper

	#	%	Faster-RCNN	BERT-base	RoBERTa-base	LayoutLM-base	Ours (Doc-GCN)
Text	91024	72.87	96.82	97.74	97.90	97.94	**99.18**
Title	19343	15.48	92.57	96.23	96.21	96.44	**98.11**
List	4913	3.93	49.51	76.73	79.08	76.21	**88.11**
Table	5018	4.02	95.49	91.61	91.30	91.30	**98.16**
Figure	4619	3.70	96.87	77.10	76.01	75.17	**98.71**
Overall	124917	100	96.96	95.96	96.08	96.03	**98.63**

Table 1: F1 comparison for each component category of **PubLayNet** test set. The number(#) and percentage (%) of each component is listed. The best models are bolded, and Doc-GCN always achieved the best for each component.

	#	%	Faster-RCNN	BERT-base	RoBERTa-base	LayoutLM-base	Ours (Doc-GCN)
Question	1077	46.18	74.40	87.23	87.02	87.08	**89.32**
Answer	821	35.21	69.26	82.44	84.11	81.72	**88.81**
Header	122	5.23	50.57	38.89	45.98	44.90	**61.96**
Other	312	13.38	65.12	57.33	56.20	56.19	**72.76**
Overall	2332	100	65.12	79.02	81.69	78.85	**85.49**

Table 2: F1 comparison for each component of **FUNSD** test set. The number(#) and percentage (%) of each component is listed. The best models are bolded, and Doc-GCN always achieved the best for each component.

	#	%	Faster-RCNN	BERT-base	RoBERTa-base	LayoutLM-base	Ours (Doc-GCN)
Abstract	420	0.70	0	70.04	60.61	58.27	**78.69**
Author	484	0.80	0	72.56	**82.01**	69.07	79.46
Caption	1840	3.05	4.44	77.33	76.48	74.78	**87.38**
Date	87	0.14	0	67.76	88.74	85.35	**91.02**
Equation	11846	19.66	81.80	85.92	86.08	86.00	**90.06**
Figure	1650	2.74	68.74	**100**	**100**	**100**	99.97
Footer	529	0.88	0	69.08	68.67	65.56	**84.48**
List	958	1.59	0	54.72	55.01	50.43	**65.79**
Paragraph	35496	58.92	83.69	89.44	89.69	89.05	**96.50**
Reference	1237	2.05	0	88.51	88.30	**89.61**	88.83
Section	4891	8.12	78.52	84.04	84.79	83.90	**95.93**
Table	525	0.87	0	49.29	51.79	49.17	**59.06**
Title	286	0.47	0	51.49	68.58	55.53	**84.85**
Overall	60249	100	71.02	86.65	86.97	86.16	**91.83**

Table 3: F1 comparison for each component of **DocBank** test set. The number(#) and percentage (%) of each component is listed. The best models are bolded, and Doc-GCN always achieved the best/the second best.

5 Results

5.1 Performance Comparison

We compared the performance of our proposed Doc-GCN with the baseline models on the test set of PubLayNet, FUNSD, and DocBank. The breakdown F1 scores on each layout component of the three datasets are provided in Table 1, 2 and 3 respectively.

We can see from the overall F1 scores that Doc-GCN outperforms all the baseline models on all datasets. It achieves a 98.63% F1 score in PubLayNet, which is 1.67% higher than the pre-trained Faster-RCNN. The performances of Faster-RCNN on FUNSD and DocBank are also much less competitive than models utilizing semantic features, which indicates the critical role of semantic information in the DLA task. Compared to BERT and RoBERTa, which only uses the semantic features, the F1 scores of our model are 2.55%, 3.8%, 4.86% higher than RoBERTa and 2.67%, 6.47%, 5.18% higher than BERT on PubLayNet, FUNSD and DocBank respectively. Especially for FUNSD, where there are only 144 training PDF pages, our model still achieves around 85%. Such results also prove the effectiveness of our multiple aspects of document layout component features for low-resource DLA datasets. Though LayoutLM also utilizes dual-aspect, visual aspect (position), and textual features, our Doc-GCN produces better

				PubLayNet			FUNSD			DocBank		
Syn	***Sem***	***Dens***	***Appr***	Precision	Recall	F1	Precision	Recall	F1	Precision	Recall	F1
O	X	X	X	67.81	72.92	63.45	54.80	49.36	39.68	40.95	46.01	42.40
X	O	X	X	96.70	96.37	96.48	81.68	82.33	81.84	88.68	88.54	88.61
X	X	O	X	50.65	71.17	59.18	45.91	48.93	43.52	34.71	58.90	43.68
X	X	X	O	96.53	95.95	96.15	70.82	70.88	70.63	87.18	85.91	86.27
X	O	X	O	98.42	98.36	98.38	84.45	84.99	84.41	91.80	91.79	91.66
X	O	O	O	98.49	98.45	98.46	84.55	84.31	84.42	91.95	**91.94**	91.74
O	O	X	O	98.61	98.58	98.59	84.85	84.99	84.85	92.00	91.77	91.68
O	O	O	O	**98.64**	**98.65**	**98.63**	**85.54**	**85.55**	**85.49**	**92.07**	**91.94**	**91.83**

Table 4: F1 comparison of using different aspects of graph features on the validation set of three datasets. *Syn*, *Sem*, *Dens* and *Appr* stand for the Syntactic-based, Semantic-based, Density-based and Appearance-based graphs respectively. "O" and "X" refer to the existence and absence of corresponding graph features for the classification. The second best is underlined.

overall F1 scores on all datasets. The superior performance of Doc-GCN demonstrates the effectiveness of the multiple aspects we applied. It indicates the positive contribution of graph representations and the joint GCN learning for integrating multiple features in the DLA task.

Furthermore, Doc-GCN shows significant performance improvement for some important layout component detection for each domain. It produce a better result in all components in PubLayNet, including the *Title*, *Table*, *Figure*, and the *List* that contains the structural text information, as is demonstrated in Table 1. Similar patterns can be also observed in Table 2 for FUNSD, especially with those important components of the forms: *Header* (section title - e.g. the aim of the form) and *Answer* (value - e.g. the content filled by the form user). For DocBank in Table 3, it shows that Doc-GCN works well with the essential components of scientific academic publication, including *Abstract*, *Date*, *Section*, *Title*. Those components are generally extracted for understanding the content of documents.

5.2 Comparison of Aspect Variants

To inspect the effectiveness of each aspect feature captured by Doc-GCN, we further compare its test performances by applying different combinations of aspect features. As seen in Table 4, using only semantic or appearance features already resulted in around 96% and 88% in F1 scores on PubLayNet and DocBank, respectively. For FUNSD, using only appearance features results in 70.63% in the F1 score, and using only semantic features is around 10% higher. The combination of semantic and appearance features improves the F1 scores to 98.38%, 84.41%, and 91.66% for PubLayNet, FUNSD, and DocBank, respectively. The semantic and appearance features seem to dominate the model's performance in the DLA task, but the syntactic and density features also positively contribute to the performance. By adding syntactic features to the semantic and appearance features, the F1 scores on PubLayNet and FUNSD improve to 98.59% and 84.85%, respectively. Further, adding density features results in the best performances of 98.63% and 85.49%, respectively. For DocBank, adding density features improves the F1 score from 91.66% to 91.74% and finally reaches the best performance of 91.83% after further including the syntactic features. Though the density features and syntactic features contribute differently in the cases of different datasets, it is evident that the utilization of both density features and syntactic features is effective for performance improvement in the DLA task. Such results also indicate that our proposed aspects and their representations are practical for a more comprehensive representation of the characteristics of document layout components.

	PubLayNet	**FUNSD**	**DocBank**
Bert-L	97.06	80.86	86.42
RoBERTa-L	96.15	79.47	86.67
LayoutLM-L	96.94	78.90	83.21
Doc-GCN-L	**98.22**	**85.40**	**90.07**

Table 5: Performances comparison (F1 score%) based on large (L) pretrained models.

5.3 Impact of Pretrained Model Variants

To evaluate the effects of base and large pretrained models, we also tested the performances of

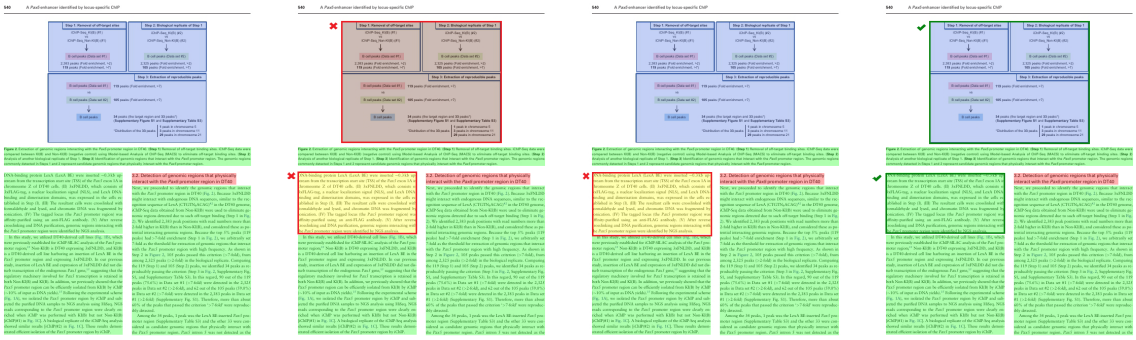

(a) **Ground Truth** (b) **RoBERTa-base** (c) **Faster-RCNN** (d) **Doc-GCN**

Figure 2: Example output of Top3 models for a PubLayNet page. The color of layout component labels are: Text, Title, List, Table, Figure. Our Doc-GCN classified all layout components accurately.

	PubLayNet	FUNSD	DocBank
Min Pooling	**98.67**	82.77	90.27
Avg Pooling	97.90	82.04	89.04
Max Pooling	98.63	**85.49**	**91.83**

Table 6: Performances comparison (F1 score%) of different node-level pooling methods on the test set.

large BERT, RoBERTa and LayoutLM over three datasets. For a fair comparison, we deployed Doc-GCN Large, shown as Doc-GCN-L in Table 5, which uses the same graph construction and representations as the original Doc-GCN except that the large pretrained BERT is used for generating the semantic-aspect graph features.

The result shows that large models have similar performances as base models. Nevertheless, our Doc-GCN-L still outperforms BERT-L, RoBERTa-L and LayoutLM-L on all three datasets: PubLayNet, FUNSD, and DocBank.

5.4 Impact of Pooling Variants

We applied the node-level pooling to integrate the node features over the two graph variants as the final features for each aspect representation. We compared Minimum, Average, and Max Pooling and used the method with the best results. From Table 6, we can see that for FUNSD and DocBank, Max Pooling resulted in the best performances. Especially for the FUNSD dataset, Max Pooling achieved almost 3% higher in F1 score compared to the results of using Minimum Pooling and Average Pooling. For PubLayNet, Minimum Pooling results in the best performance but is only 0.03% higher than Max Pooling. Hence, we used Max Pooling as the ultimate pooling method in our Doc-GCN.

5.5 Case Study[9]

We visualized the sample results for the top 3 models on a document page of PubLayNet in Figure 2. We can see that both RoBERTa and Faster-RCNN have wrongly recognized a *Text* into *List*, whereas our Doc-GCN has accurately recognized all components. This case further proves that simply considering the semantic or visual information is hard to distinguish the *List* and *Text*, indicating the importance of capturing the structural relationships between layout components for better performance.

6 Conclusion[10]

We successfully handled the importance of the DLA task, Document Component/Segment Classification. It focused on extracting important information (i.e., Title, Author, Date, Form Answer) from the digital documents, including Scanned Document Images and PDF files. We propose a heterogeneous graph-based DLA model, Doc-GCN, which integrates text density, appearance, and syntactic and semantic properties of the Document layout components to generate comprehensive representations of documents. The graph structure also enables the model to capture and learn the relationships between the layout components when making predictions. Our model outperforms all baselines on three publicly available DLA datasets. We strongly hope that Doc-GCN will motivate and provide insights into the future integration of different modalities and aspects for the logical document layout analysis task.

[9]More qualitative analysis for each of the three datasets can be found in Appendix B, and the superiority of understanding multiple aspects will be highlighted there.

[10]Appendix: https://github.com/adlnlp/doc_gcn/tree/main/appendix

References

Dario Augusto Borges Oliveira and Matheus Palhares Viana. 2017. Fast cnn-based document layout analysis. In *Proceedings of the IEEE International Conference on Computer Vision Workshops*, pages 1173–1180.

Galal M Binmakhashen and Sabri A Mahmoud. 2019. Document layout analysis: a comprehensive survey. *ACM Computing Surveys (CSUR)*, 52(6):1–36.

Hongjie Cai, Yaofeng Tu, Xiangsheng Zhou, Jianfei Yu, and Rui Xia. 2020. Aspect-category based sentiment analysis with hierarchical graph convolutional network. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 833–843.

Guimin Chen, Yuanhe Tian, and Yan Song. 2020. Joint aspect extraction and sentiment analysis with directional graph convolutional networks. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 272–279.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. Bert: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186.

Yihao Ding, Zhe Huang, Runlin Wang, YanHang Zhang, Xianru Chen, Yuzhong Ma, Hyunsuk Chung, and Soyeon Caren Han. 2022. V-doc: Visual questions answers with documents. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 21492–21498.

James L Fisher, Stuart C Hinds, and Donald P D'Amato. 1990. A rule-based system for document image segmentation. In *[1990] Proceedings. 10th International Conference on Pattern Recognition*, volume 1, pages 567–572. IEEE.

Jiuxiang Gu, Jason Kuen, Vlad Morariu, Handong Zhao, Rajiv Jain, Nikolaos Barmpalios, Ani Nenkova, and Tong Sun. 2021. Unidoc: Unified pretraining framework for document understanding. *Advances in Neural Information Processing Systems*, 34.

Caren Han, Siqu Long, Siwen Luo, Kunze Wang, and Josiah Poon. 2020. VICTR: Visual information captured text representation for text-to-vision multimodal tasks. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 3107–3117.

Adam W Harley, Alex Ufkes, and Konstantinos G Derpanis. 2015. Evaluation of deep convolutional nets for document image classification and retrieval. In *2015 13th International Conference on Document Analysis and Recognition (ICDAR)*, pages 991–995. IEEE.

Kaiming He, Georgia Gkioxari, Piotr Dollár, and Ross Girshick. 2017. Mask r-cnn. In *Proceedings of the IEEE international conference on computer vision*, pages 2961–2969.

Qingbao Huang, Jielong Wei, Yi Cai, Changmeng Zheng, Junying Chen, Ho-fung Leung, and Qing Li. 2020. Aligned dual channel graph convolutional network for visual question answering. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7166–7176.

Guillaume Jaume, Hazim Kemal Ekenel, and Jean-Philippe Thiran. 2019. Funsd: A dataset for form understanding in noisy scanned documents. In *2019 International Conference on Document Analysis and Recognition Workshops (ICDARW)*, volume 2, pages 1–6. IEEE.

Justin Johnson, Agrim Gupta, and Li Fei-Fei. 2018. Image generation from scene graphs. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1219–1228.

Thomas N Kipf and Max Welling. 2016. Semi-supervised classification with graph convolutional networks. *ICLR*.

Nikita Kitaev and Dan Klein. 2018. Constituency parsing with a self-attentive encoder. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 2676–2686.

Stefan Klink and Thomas Kieninger. 2001. Rule-based document structure understanding with a fuzzy combination of layout and textual features. *International Journal on Document Analysis and Recognition*, 4(1):18–26.

Kai Li, Curtis Wigington, Chris Tensmeyer, Handong Zhao, Nikolaos Barmpalios, Vlad I Morariu, Varun Manjunatha, Tong Sun, and Yun Fu. 2020a. Cross-domain document object detection: Benchmark suite and method. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 12915–12924.

Minghao Li, Yiheng Xu, Lei Cui, Shaohan Huang, Furu Wei, Zhoujun Li, and Ming Zhou. 2020b. Docbank: A benchmark dataset for document layout analysis. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 949–960.

Chunxiao Liu, Zhendong Mao, Tianzhu Zhang, Hongtao Xie, Bin Wang, and Yongdong Zhang. 2020a. Graph structured network for image-text matching. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 10921–10930.

Xien Liu, Xinxin You, Xiao Zhang, Ji Wu, and Ping Lv. 2020b. Tensor graph convolutional networks for text classification. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pages 8409–8416.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Roberta: A robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692*.

Siqu Long, Feiqi Cao, Soyeon Caren Han, and Haiqin Yang. 2022a. Vision-and-language pretrained models: A survey. In *Proceedings of the Thirty-First International Joint Conference on Artificial Intelligence, IJCAI-22*, pages 5530–5537. International Joint Conferences on Artificial Intelligence Organization. Survey Track.

Siqu Long, Soyeon Caren Han, Xiaojun Wan, and Josiah Poon. 2022b. Gradual: Graph-based dual-modal representation for image-text matching. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, pages 3459–3468.

Siwen Luo, Soyeon Caren Han, Kaiyuan Sun, and Josiah Poon. 2020. Rexup: I reason, i extract, i update with structured compositional reasoning for visual question answering. In *International Conference on Neural Information Processing*, pages 520–532. Springer.

Siwen Luo, Hamish Ivison, Caren Han, and Josiah Poon. 2021. Local interpretations for explainable natural language processing: A survey. *arXiv preprint arXiv:2103.11072*.

Debashish Niyogi and Sargur N Srihari. 1986. A rule-based system for document understanding. In *Proceedings of the Fifth AAAI National Conference on Artificial Intelligence*, pages 789–793.

Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun. 2015. Faster r-cnn: towards real-time object detection with region proposal networks. In *Proceedings of the 28th International Conference on Neural Information Processing Systems-Volume 1*, pages 91–99.

Carlos Soto and Shinjae Yoo. 2019. Visual detection with context for document layout analysis. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 3455–3461.

Shikhar Vashishth, Shib Sankar Dasgupta, Swayambhu Nath Ray, and Partha Talukdar. 2018. Dating documents using graph convolution networks. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1605–1615.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Proceedings of the 31st International Conference on Neural Information Processing Systems*, pages 6000–6010.

Kunze Wang, Caren Han, Siqu Long, and Josiah Poon. 2022a. Me-gcn: Multi-dimensional edge-embedded graph convolutional networks for semi-supervised text classification. In *ICLR 2022 Workshop on Deep Learning on Graphs for Natural Language Processing*.

Kunze Wang, Soyeon Caren Han, and Josiah Poon. 2022b. Induct-gcn: Inductive graph convolutional networks for text classification. In *26TH International Conference on Pattern Recognition (ICPR)*.

Renshen Wang, Yasuhisa Fujii, and Ashok C Popat. 2022c. Post-ocr paragraph recognition by graph convolutional networks. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, pages 493–502.

Yang Xu, Yiheng Xu, Tengchao Lv, Lei Cui, Furu Wei, Guoxin Wang, Yijuan Lu, Dinei Florencio, Cha Zhang, Wanxiang Che, Min Zhang, and Lidong Zhou. 2021. LayoutLMv2: Multi-modal pre-training for visually-rich document understanding. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 2579–2591, Online. Association for Computational Linguistics.

Yiheng Xu, Minghao Li, Lei Cui, Shaohan Huang, Furu Wei, and Ming Zhou. 2020. Layoutlm: Pre-training of text and layout for document image understanding. In *Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, pages 1192–1200.

Liang Yao, Chengsheng Mao, and Yuan Luo. 2019. Graph convolutional networks for text classification. In *Proceedings of the AAAI conference on artificial intelligence*, volume 33, pages 7370–7377.

Peng Zhang, Can Li, Liang Qiao, Zhanzhan Cheng, Shiliang Pu, Yi Niu, and Fei Wu. 2021. Vsr: A unified framework for document layout analysis combining vision, semantics and relations. In *Document Analysis and Recognition – ICDAR 2021*, pages 115–130, Cham. Springer International Publishing.

Xu Zhong, Jianbin Tang, and Antonio Jimeno Yepes. 2019. Publaynet: largest dataset ever for document layout analysis. In *2019 International Conference on Document Analysis and Recognition (ICDAR)*, pages 1015–1022. IEEE.

Analytic Automated Essay Scoring based on Deep Neural Networks Integrating Multidimensional Item Response Theory

Takumi Shibata
The University
of Electro-Communications
shibata@ai.lab.uec.ac.jp

Masaki Uto
The University
of Electro-Communications
uto@ai.lab.uec.ac.jp

Abstract

Essay exams have been attracting attention as a way of measuring the higher-order abilities of examinees, but they have two major drawbacks in that grading them is expensive and raises questions about fairness. As an approach to overcome these problems, automated essay scoring (AES) is in increasing need. Many AES models based on deep neural networks have been proposed in recent years and have achieved high accuracy, but most of these models are designed to predict only a single overall score. However, to provide detailed feedback in practical situations, we often require not only the overall score but also analytic scores corresponding to various aspects of the essay. Several neural AES models that can predict both the analytic scores and the overall score have also been proposed for this very purpose. However, conventional models are designed to have complex neural architectures for each analytic score, which makes interpreting the score prediction difficult. To improve the interpretability of the prediction while maintaining scoring accuracy, we propose a new neural model for automated analytic scoring that integrates a multidimensional item response theory model, which is a popular psychometric model.

1 Introduction

Rapid changes in society in recent years have led to an increased need for cultivating and assessing not only knowledge and skills but also practical abilities, such as expression skills, logical thinking, and creativity (Erguvan and Aksu Dunya, 2020; Uto, 2021a). Essay exams are one of the test formats that aim to evaluate these abilities, and consequently, they have been used in various educational and assessment settings (Erguvan and Aksu Dunya, 2020; Hussein et al., 2019). However, essay exams have two considerable drawbacks in the time and monetary costs required to grade them (Taghipour and Ng, 2016). Furthermore, it is difficult to ensure consistently fair and reliable evaluation due to subjective influences on the part of the rater (Uto and Ueno, 2020; Saal et al., 1980). Automated Essay Scoring (AES) has been attracting attention as a method for resolving these difficulties (Dong and Zhang, 2016; Taghipour and Ng, 2016).

Conventional AES systems can be broadly classified into two categories (Hussein et al., 2019): those that take a feature-engineering approach and those that take a neural approach. The feature-engineering approach, which has traditionally been the greater used of the two, utilizes a statistical or machine learning model with pre-defined handcrafted features (e.g. Attali and Burstein, 2006; Chen and He, 2013; Phandi et al., 2015; Dascalu et al., 2017; Hastings et al., 2018; Yao et al., 2019). The neural approach, on the other hand, which has become popular recently, uses deep neural networks to extract features automatically from texts (e.g. Alikaniotis et al., 2016; Taghipour and Ng, 2016; Dong and Zhang, 2016; Tay et al., 2018; Dong et al., 2017; Farag et al., 2018; Jin et al., 2018; Uto et al., 2020; Rodriguez et al., 2019; Uto et al., 2020; Ridley et al., 2020; Uto, 2021c). In this study, we focus on the neural approach because of the high accuracy it has achieved in many prior studies.

Most neural AES studies have focused on holistic scoring (Ridley et al., 2021; Ke and Ng, 2019), which provides a single overall score for each essay. However, to provide richer feedback, especially when essay exams are used for educational purposes, we often require not only the overall score but also analytic scores corresponding to various aspects of the essay, such as *content*, *organization*, and *word choice* (Hussein et al., 2020). Several AES models that can predict these analytic scores along with the overall score have recently been proposed for this purpose (Mathias and Bhattacharyya, 2020; Hussein et al., 2020; Mim et al., 2019; Ridley et al., 2021). From here on, we will refer to such

Proceedings of the 29th International Conference on Computational Linguistics, pages 2917–2926
October 12–17, 2022.

models as analytic AES models.

Mathias and Bhattacharyya (2020) proposed an early neural analytic AES model that took the simple approach of separately applying a conventional holistic scoring model (Dong et al., 2017) to each analytic score. Then, Hussein et al. (2020) proposed a multi-output model in which the output layers are branched by the number of analytic scores and the other layers are shared. One of the more recent models is a multi-output model proposed by Ridley et al. (2021) that has a complex deep neural architecture as the output layer for each analytic score. Although this model produces state-of-the-art accuracy, it has some problems in terms of interpretability.

1. It has a complex neural architecture for each analytic score, decreasing the interpretability of the prediction.

2. In general, analytic scores are designed to measure latent abilities in examinees that a test developer wishes to evaluate (Uto, 2021b). However, this model ignores the existence of an ability scale, further restricting the interpretability of the score prediction.

To resolve these problems, we propose to extend a conventional analytic AES model by integrating it with an item response theory (IRT) (Lord, 1980) model, a well-known psychometric model. Specifically, we extend the multi-output model of Ridley et al. (2021) by replacing the complex output layers for each analytic score with a multidimensional IRT model (Yao and Schwarz, 2006). The advantages of the proposed model are as follows.

1. The output IRT layer is explained by only three types of parameters: the discriminatory power and difficulty corresponding to each analytic score and the latent ability of each examinee. These allow us to better interpret the reasoning behind score predictions.

2. Investigating an optimal number of ability dimensions in the multidimensional IRT model layer and analyzing the estimated parameters allows us to interpret the ability scale implied by the multiple analytic scores.

In this study, we used benchmark datasets that have been widely used in analytic AES research to conduct experiments that evaluated the effectiveness of our model. They showed that our model offers reasonably interpretable parameters without significantly degrading scoring accuracy. Furthermore, an interesting finding from our experiment was that, although the benchmark dataset consisted of many analytic scores for each essay, only one or two latent abilities were measured by those multiple scores.

Note that a similar AES framework combining deep neural networks and IRT was recently proposed (Uto and Okano, 2021). However, they used IRT to improve the quality of training data by mitigating rater effects, so their research objective was completely different from the one we focus on in this study.

2 Conventional analytic AES model

This section introduces the conventional analytic AES model proposed by Ridley et al. (2021), which we use as a baseline model. The architecture of this model is displayed on the left side of Figure 1.

This model takes in an essay from examinee n and outputs multiple analytic scores $\{\hat{y}_{nm} \mid m \in \{1, 2, \ldots, M\}\}$, where $\hat{y}_{nm}$ is the m-th analytic score and M is the total number of analytic scores. An essay from examinee n is defined as a word sequence $\{w_{nsl} \mid s \in \{1, 2, \ldots, S\}, l \in \{1, 2, \ldots, l_s\}\}$, where w_{nsl} is the l-th word in the s-th sentence of examinee n's essay, S is the number of the sentences in the essay, and l_s is the number of words in the s-th sentence. Note that in our paper, we regard the overall score as one of the analytic scores.

The model consists of two types of layers: a *shared layer* and an *item-specific layer*. The shared layer receives the word sequence in each sentence and produces a sentence-level distributed representation through an embedding layer, a convolutional layer, and an attention pooling layer (Dong et al., 2017). The sequence for the sentence-level distributed representation is used in the item-specific layer.

The item-specific layer consists of the same number of heads as the number of analytic scoring items, which are evaluation items corresponding to analytic scores, such as *content*, *organization*, and *word choice*. An item-specific layer for an analytic scoring item receives the sequence of the sentence-level distributed representation and produces a score value for the corresponding scoring item. Specifically, the input sequence is first processed through a recurrent neural network (RNN),

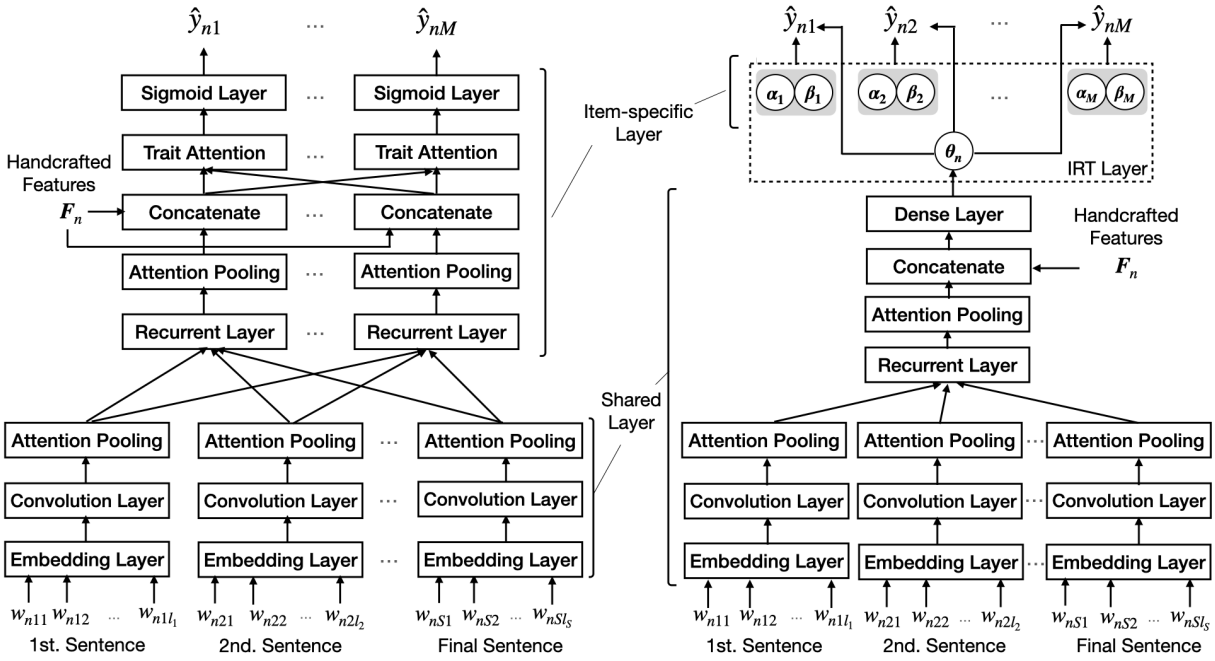

Figure 1: Architecture of a conventional analytic scoring model (left) and our model (right).

one in which the long short-term memory (Hochreiter and Schmidhuber, 1997) was used as the RNN. Then, an output sequence from the RNN layer is aggregated into a fixed-length hidden vector through an attention pooling layer (Dong et al., 2017). The hidden vector is concatenated with a manually designed feature vector $\boldsymbol{F}_n$, and the concatenated vector $\boldsymbol{h}_{nm}$ is input to the trait attention layer. For capturing the relation between analytic scoring items, the trait attention layer is defined as

$$a_{nmt} = \frac{\exp(\boldsymbol{h}_{nm} \cdot \boldsymbol{h}_{nt})}{\sum_{\substack{t=1 \\ t \neq m}}^{M} \exp(\boldsymbol{h}_{nm} \cdot \boldsymbol{h}_{nt})}, \ \forall t, \forall m, t \neq m \tag{1}$$

$$\boldsymbol{x}_{nm} = \sum_{\substack{t=1 \\ t \neq m}}^{M} a_{nmt} \boldsymbol{h}_{nt} \tag{2}$$

$$\tilde{\boldsymbol{x}}_{nm} = \mathrm{Concat}(\boldsymbol{x}_{nm}, \boldsymbol{h}_{nm}). \tag{3}$$

Finally, a linear layer with the sigmoid activation maps $\tilde{\boldsymbol{x}}_{nm}$, a trait attention output vector, to the prediction score $\hat{y}_{nm}$:

$$\hat{y}_{nm} = \sigma(\boldsymbol{W}_m \tilde{\boldsymbol{x}}_{nm} + b_m), \tag{4}$$

where σ is the sigmoid function, $\boldsymbol{W}_m$ is a weight vector, and b_m is a bias value. Note that this model uses a sigmoid function to predict scores, so $\hat{y}_{nm}$ takes values between 0 and 1. Thus, in the score prediction phase, the output scores must be linearly transformed to the original score scale.

This model is trained through a back-propagation algorithm using the Mean Squared Error (MSE) as a loss function. This is given by

$$\mathcal{L}_{MSE} = \frac{1}{NM} \sum_{n=1}^{N} \sum_{m=1}^{M} (\hat{y}_{nm} - y_{nm})^2, \tag{5}$$

where N is the number of essays and y_{nm} is the gold-standard score of examinee n for the m-th analytic scoring item. The gold-standard scores y_{nm} must be linearly transformed into the range between 0 and 1.

Note that Ridley et al. (2021) input the part-of-speech (POS) tags instead of the words themselves when applying the model to cross-prompt scoring tasks. However, we use word sequences as input because they are expected to be more accurate for the prompt-specific scoring tasks used in this study.

As previously mentioned, this model has a complex architecture for each analytic score, making it difficult to interpret the score prediction. Our main focus is to use IRT to increase the interpretability of score prediction.

3 Item Response Theory

IRT (Lord, 1980) is a popular psychometric model that has been widely used for making measurements in educational and psychological research.

Typical IRT models define the probability that an examinee will receive a certain score on a test item as a function of the examinee's latent ability and the item's characteristic parameters, such as the discrimination and difficulty parameters. Of the various existing IRT models, we employ a multidimensional generalized partial credit model (M-GPCM) (Yao and Schwarz, 2006), a representative multidimensional polytomous IRT model that can be applied to ordinal score data and can examine multidimensional latent abilities for each examinee.

If we regard IRT parameters for test items as those for analytic scoring items following the approach in previous studies (Uto, 2021b), then M-GPCM defines the probability that examinee n will receive score k for the m-th analytic scoring item as

$$P_{nmk} = \frac{\exp(k\boldsymbol{\alpha}_m^T\boldsymbol{\theta}_n + \sum_{u=1}^{k}\beta_{mu})}{\sum_{v=1}^{K_m}\exp(v\boldsymbol{\alpha}_m^T\boldsymbol{\theta}_n + \sum_{u=1}^{v}\beta_{mu})}, \quad (6)$$

where $\boldsymbol{\theta}_n = (\theta_{n1}, \theta_{n2}, \ldots, \theta_{nd})$ is the d-dimensional latent ability of examinee n, $\boldsymbol{\alpha}_m = (\alpha_{m1}, \alpha_{m2}, \ldots, \alpha_{md})$ is a d-dimensional discrimination parameter for analytic scoring item m, β_{mu} is a step parameter denoting the difficulty of the transition between scores $u-1$ and u in item m, and K_m is the number of possible scores for the m-th item. Here, $\beta_{m1} = 0 : \forall m$ is assumed for model identification.

All of these model parameters, $\boldsymbol{\theta}_n$, $\boldsymbol{\alpha}_m$, and β_{mu}, can be estimated from a collection of observed scores. These parameters are clearly interpretable, as will be explained in sections 4.3 and 5.3.

4 Proposed Model

We propose an analytic AES model that incorporates the M-GPCM mentioned in the previous section. The architecture of this model is displayed on the right side of Figure 1.

As Figure 1 shows, our model and the conventional model share the same layers from the input to the concatenate layer. Specifically, in both models, each sentence in an essay is fed to the embedding layer, convolution layer, and attention pooling layer, and then a sequence of the sentence-level distributed representation vectors is transformed into a fixed-length vector through the recurrent layer and the attention pooling layer. Finally, the concatenate layer creates an essay-level vector $\boldsymbol{h_n}$ by combining the output from the attention pooling layer and the handcrafted feature vector $\boldsymbol{F}_n$.

The main differences between the models occur after the concatenate layer. Given the essay-level vector $\boldsymbol{h_n}$, our model obtains the latent ability vector $\boldsymbol{\theta}_n$, which is used in the subsequent M-GPCM layer, by applying a dense layer given by

$$\boldsymbol{\theta}_n = \boldsymbol{W}\boldsymbol{h}_n + \boldsymbol{b}, \quad (7)$$

where $\boldsymbol{W}$ is a weights matrix and $\boldsymbol{b}$ is a bias vector. The latent ability $\boldsymbol{\theta}_n$ is input to the M-GPCM defined in Eq. (6) to obtain the score probabilities for each analytic scoring item m. Our model then uses the obtained probability P_{nmk} to predict the analytic scores.

4.1 Model Training

We train our model using the following Categorical Cross-Entropy (CCE) as a loss function:

$$\mathcal{L}_{CCE} = -\frac{1}{NM}\sum_{n=1}^{N}\sum_{m=1}^{M}\sum_{k=1}^{K_m} y_{nmk}\log(P_{nmk}). \quad (8)$$

We use this because the output IRT layer gives the probability distribution over score categories P_{nmk}. Note that during the training process, our model simultaneously estimates the IRT parameters, namely $\boldsymbol{\theta}_n$, $\boldsymbol{\alpha}_m$, and $\boldsymbol{\beta}_m = (\beta_{m1}, \beta_{m2}, \ldots, \beta_{mK_m})$, and the parameters in the other layers in an end-to-end manner.

The hyper-parameters in our model are the same as those in the conventional model (Ridley et al., 2021), and we use the RMSProp Optimizer (Dauphin et al., 2015) with a learning rate of 0.001. Furthermore, since IRT generally assumes a normal distribution for $\boldsymbol{\theta}_n$, we apply an L2-regularization for $\boldsymbol{\theta}_n$ so that its distribution closes to a normal distribution with mean zero.

4.2 Score Prediction

We have the following two options for predicting a score based on the output score probabilities P_{nmk}.

- Argmax score: $\arg\max_k P_{nmk}$.
- Expected score: $\sum_{k=1}^{K_m} kP_{nmk}$.

We compare these two options in the experiments discussed in section 5.2.

Table 1: Summary of the ASAP and ASAP++ dataset: *Org* refers to organization, *WC* to word choice, *SF* to sentence fluency, *Conv* to conventions, *PA* to prompt adherence, *Lang* to language, and *Narr* to narrativity.

Prompt	Num Essays	Mean Length	Analytic Scoring Items	Score Range	
				Overall	Analytic
1	1783	350	Overall, Content, Org, WC, SF, Conv	2-12	1-6
2	1800	350	Overall, Content, Org, WC, SF, Conv	1-6	1-6
3	1726	150	Overall, Content, PA, Lang, Narr	0-3	0-3
4	1772	150	Overall, Content, PA, Lang, Narr	0-3	0-3
5	1805	150	Overall, Content, PA, Lang, Narr	0-4	0-4
6	1800	150	Overall, Content, PA, Lang, Narr	0-4	0-4
7	1569	250	Overall, Content, Org, Conv, Style	0-30	0-6
8	723	650	Overall, Content, Org, WC, SF, Conv, Voice	0-60	2-12

4.3 Interpretability of our model

As explained in section 3, the M-GPCM consists of three types of trainable parameters: both the discrimination parameters $\boldsymbol{\alpha}_m$ and the difficulty parameters $\boldsymbol{\beta}_m$ for each analytic scoring item and the latent examinee ability parameter $\boldsymbol{\theta}_n$.

The discrimination parameter $\boldsymbol{\alpha}_m$ provides information on how well each analytic scoring item distinguishes examinee ability, whereas the difficulty parameter $\boldsymbol{\beta}_m$ reflects how difficult examinees find each score category for the m-th analytic scoring item to be. The examinee ability parameter $\boldsymbol{\theta}_n$ represents the ability level of each examinee. Section 5.3 shows an example of the interpretation of these parameters.

Furthermore, our model enables us to perform an analysis of the optimal number of ability dimensions assumed under multiple analytic scores by comparing its performance with different numbers of dimensions. For example, if the score prediction performance of our model is maximized when two ability dimensions are assumed, then we can interpret this as indicating that the given analytic scoring items measure a two-dimensional latent ability of examinees. We can also interpret what each ability dimension measures by analyzing the multidimensional discrimination parameter $\boldsymbol{\alpha}_m$. Section 5.3 gives an example of how the ability dimension can be interpreted.

Our model predicts the scores by using the output IRT layer with the IRT parameters mentioned above. Thus, interpreting these parameters allows us to understand how the model determines analytic scores for a given essay.

5 Experiments

In this section, we discuss how the effectiveness of our model was evaluated through experiments using real-word data.

5.1 Real-word data

In our experiments, we used real-word data from the Automated Student Assessment Prize (ASAP)[1] and the ASAP++ (Mathias and Bhattacharyya, 2018).

The ASAP was introduced in the Kaggle competition and has since been widely used in AES research. The ASAP dataset consists of examinees' essays for eight different prompts and scores for them. Only an overall score is given for prompts 1 through 6, while some analytic scores are given in addition to the overall score for prompts 7 and 8. The ASAP++, a dataset designed to supplement ASAP, offers analytic scores for prompts 1 through 6.

Table 1 gives a summary of the ASAP with the ASAP++ dataset.

5.2 Evaluation of our model

Using the ASAP with the ASAP++ dataset, we evaluated the scoring accuracy of our model while varying the number of ability dimensions from 1 to 3 and compared the results to those from the conventional baseline model described in section 2. The scoring accuracy was independently evaluated for each prompt through a 5-fold cross validation using the Quadratic Weighted Kappa (QWK), which is used in AES studies. Concretely, we evaluated the QWK score for each analytic scoring item and then calculated the average QWK score for each prompt. We examined two input types in this experiment: a word sequence and a POS tag sequence. We used Glove (Pennington et al., 2014), a pre-trained word embedding, in the embedding layer for models using word sequences as inputs. Furthermore, in our model, we evaluated the two types of prediction scores, the argmax scores and the expected scores, explained in section 4.2.

Table 2 and Table 3 show the results obtained when the expected scores and the argmax scores

[1] https://www.kaggle.com/c/asap-aes

Table 2: QWK scores for our model with the expected scores and the conventional model.

Input	Model	Prompts 1	2	3	4	5	6	7	8	Avg.	p-value 1dim	2dim	3dim
POS	Conventional	**0.688**	**0.632**	0.610	**0.680**	0.686	**0.684**	**0.694**	0.548	**0.653**	0.460	0.169	0.767
	Proposed-1dim	0.662	0.605	**0.623**	0.663	**0.693**	0.670	0.640	0.542	0.637	-	1.000	1.000
	Proposed-2dim	0.671	0.627	0.608	0.657	0.680	0.669	0.669	**0.555**	0.642	-	-	1.000
	Proposed-3dim	0.678	0.629	0.615	0.643	0.691	0.677	0.682	0.544	0.645	-	-	-
Word	Conventional	**0.685**	**0.655**	**0.660**	0.720	**0.706**	**0.750**	0.694	0.568	**0.680**	0.009	0.699	0.014
	Proposed-1dim	0.656	0.617	0.620	0.713	0.689	0.731	0.638	0.549	0.652	-	0.180	0.378
	Proposed-2dim	0.666	0.631	0.637	**0.722**	0.699	0.732	**0.704**	**0.576**	0.671	-	-	1.000
	Proposed-3dim	0.679	0.633	0.642	0.704	0.698	0.734	0.696	0.553	0.667	-	-	-

Table 3: QWK scores for our model with the argmax scores and the conventional model.

Input	Model	Prompts 1	2	3	4	5	6	7	8	Avg.	p-value 1dim	2dim	3dim
POS	Conventional	**0.688**	0.632	0.610	**0.680**	**0.686**	0.684	**0.694**	**0.548**	**0.653**	0.253	0.469	0.420
	Proposed-1dim	0.651	0.616	0.620	0.670	0.682	0.685	0.619	0.480	0.628	-	0.053	0.755
	Proposed-2dim	0.661	0.608	0.629	0.670	0.679	0.675	0.620	0.445	0.623	-	-	1.000
	Proposed-3dim	0.636	**0.633**	**0.634**	0.656	0.685	**0.694**	0.636	0.471	0.631	-	-	-
Word	Conventional	**0.685**	**0.655**	**0.660**	0.720	**0.706**	**0.750**	**0.694**	**0.568**	**0.680**	0.080	0.100	0.090
	Proposed-1dim	0.641	0.625	0.646	0.718	0.690	0.737	0.637	0.464	0.645	-	1.000	1.000
	Proposed-2dim	0.636	0.620	0.656	**0.721**	0.692	0.736	0.675	0.486	0.653	-	-	1.000
	Proposed-3dim	0.656	0.630	0.656	0.712	0.696	0.734	0.687	0.472	0.655	-	-	-

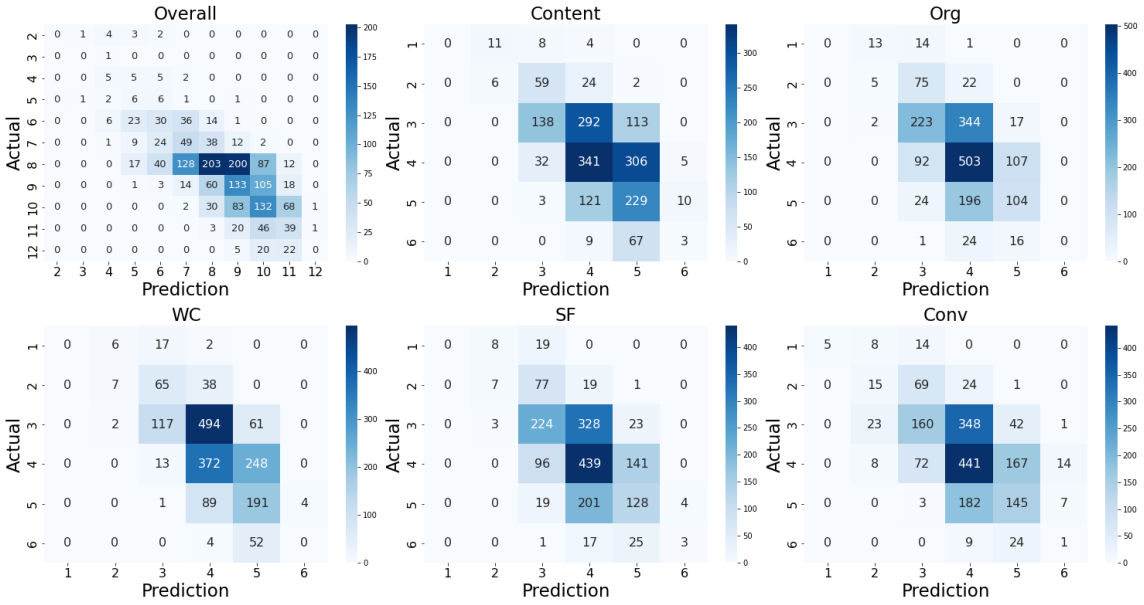

Figure 2: Confusion matrices between gold-standard scores and the expected scores from our model for prompt 1.

were used in our model, respectively. Note that the results for the conventional model are the same in both of these tables, and the highest QWK scores for each setting are shown in bold.

At first, comparing the input types suggests that the word input shows higher averaged performances in all settings. Ridley et al. (2021) used the POS tag input assuming cross-prompt tasks, as noted in section 2, whereas our experiment suggests that the word input is better for prompt-specific tasks.

Next, comparing Tables 2 and 3 shows that using the expected scores with our model tended to produce better results than when the argmax scores were used. Figure 2 shows the confusion matrices between the gold-standard scores and the expected scores given by our model for all of the analytic scoring items associated with prompt 1. According to this figure, the diagonal components of the matrices are responsive, indicating that the scores are predicted relatively well.

Finally, comparing variants of our model with different numbers of ability dimensions shows that the two- and three-dimensional models tended to outperform the one-dimensional model, although the differences are relatively small. Moreover, although the conventional model had the highest average performance, the degradations in performance of our model were small overall. We performed Bonferroni's multiple comparison test to quantitatively measure whether there were significant differences in the average QWK scores among the

Table 4: IRT parameters for analytic scoring items estimated by the one-dimensional variant of our model.

	α_1	β_{12}	β_{13}	β_{14}	β_{15}	β_{16}	β_{17}	β_{18}	β_{19}	$\beta_{1\ 10}$	$\beta_{1\ 11}$
Overall	1.15	-2.61	-3.32	-3.00	-3.79	-1.48	-2.14	0.85	0.87	2.47	2.84
Content	2.01	-4.71	-3.85	-0.56	2.02	4.20	-	-	-	-	-
Org	1.88	-4.45	-3.59	-0.32	2.25	4.75	-	-	-	-	-
WC	2.09	-4.71	-3.76	0.06	2.51	4.59	-	-	-	-	-
SF	2.06	-4.74	-3.71	-0.36	2.20	4.82	-	-	-	-	-
Conv	2.01	-4.66	-3.56	-0.36	2.29	5.01	-	-	-	-	-

Table 5: IRT parameters for analytic scoring items estimated by the two-dimensional variant of our model.

	α_{11}	α_{12}	β_{12}	β_{13}	β_{14}	β_{15}	β_{16}	β_{17}	β_{18}	β_{19}	$\beta_{1\ 10}$	$\beta_{1\ 11}$
Overall	1.81	0.14	-2.56	-3.53	-3.26	-3.75	-1.59	-2.24	0.95	1.09	2.90	3.19
Content	1.54	1.38	-4.80	-3.98	-0.60	2.04	4.35	-	-	-	-	-
Org	1.23	1.41	-4.38	-3.58	-0.39	2.18	4.65	-	-	-	-	-
WC	1.38	1.63	-5.10	-3.87	-0.01	2.55	4.73	-	-	-	-	-
SF	1.10	1.90	-4.73	-3.93	-0.49	2.21	5.02	-	-	-	-	-
Conv	1.04	1.95	-4.93	-3.87	-0.54	2.36	5.29	-	-	-	-	-

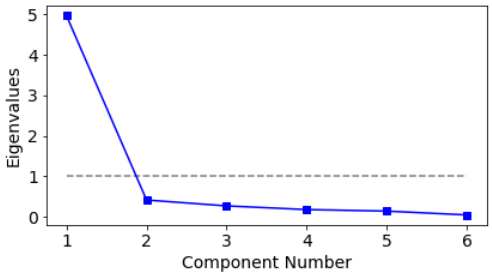

Figure 3: Scree plot for prompt 1.

models. The results are given in the p-value column of Table 2. The p-values indicate that there was no difference at the 5% significance level between the conventional model and our model with optimal dimension. This result is surprising because the scoring accuracy remains even though the item-specific layers in our model are described by significantly fewer parameters than the conventional model. Thus, we can conclude that our model does not lead to a significant decrease in the scoring accuracy.

5.3 Interpretation of our model

In this subsection, we explain how we interpreted the predictions from our model.

We first examined the optimal number of ability dimensions. In IRT studies, principal component analysis (PCA) is generally used for investigating the optimal number of dimensions (Nunnally and Bernstein, 1994; Bjorner et al., 2003). For this reason, Figure 3 shows the eigenvalues obtained by PCA for different numbers of dimensions in prompt 1; the horizontal axis shows the number of dimensions (component), and the vertical axis indicates the eigenvalue. A significant decrease in the eigenvalues occurs at the point where the component number is 2, suggesting that the ability dimension assumed under the data for the multiple analytic scores in prompt 1 is only explainable with a one-dimensional ability scale. Other prompts yielded the same results. Note that, as explained in the previous section, the one-dimensional model shows slightly lower QWK scores than the two- or three-dimensional models, so if prediction accuracy is a priority, then the two-dimensional model may be a better choice. Thus, we will now explain the interpretation of our model when one and two dimensions are assumed.

Tables 4 and 5 show the IRT-layer parameters for the analytic scoring items estimated with the one- and two-dimensional variants of our model, respectively. Only the results for prompt 1 are given here as an example.

The discrimination parameters provide information for interpreting how well the analytic scoring items measure examinees' abilities and what each ability dimension measures. For example, according to Table 4, the *overall* item has a lower discrimination value than the other analytic scoring items, suggesting that the *overall* item is relatively inaccurate for measuring a one-dimensional latent ability constructed by the multiple analytic scoring items. This also suggests the possibility that the ability measured by the *overall* item might differ from that of the other items, something that can be confirmed from the discrimination parameters in the two-dimensional model shown in Table 5. Specifically, the *overall* item in Table 5 has a large discrimination value in the first dimension but an extremely small value in the second dimension, whereas the other analytic scoring items have large discrimination values in the second dimension. Furthermore, taking a closer look at the other analytic scoring items, we can see that the *content* item

Table 6: Examples of examinees' latent abilities and the predicted scores estimated by our model.

Examinee n	Ability θ_n	Predicted Scores						
		Overall	*Content*	*Org*	*WC*	*SF*	*Conv*	Avg.
4	0.14	8	4	4	4	4	4	4.67
27	2.21	11	6	5	6	5	5	6.33
31	-2.01	6	2	2	2	2	2	2.67
916	-4.19	2	1	1	1	1	1	1.17
1651	3.40	12	6	6	6	6	6	7.00

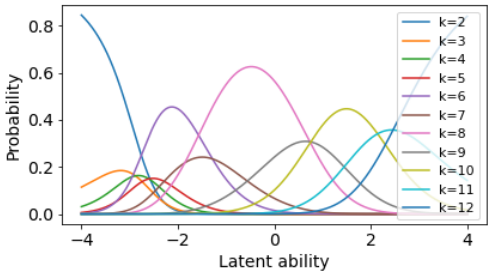

Figure 4: ICCs for the *overall* score.

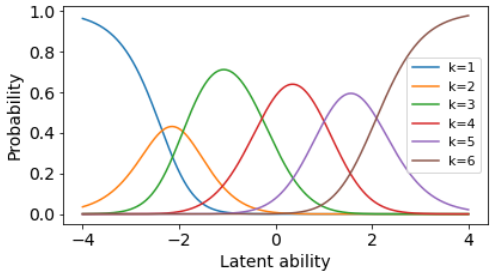

Figure 5: ICCs for the *content* score.

is like the *overall* item in that it has a higher discrimination value for the first dimension than for the second dimension, while the other items have higher discrimination values for the second dimension. These results suggest that the first ability dimension measures the overall ability relating to the skills for enriching content in an essay, while the second dimension measures the ability shared among *organization*, *word choice*, *sentence fluency*, and *convention*, which would make it a technical writing ability.

Furthermore, the difficulty parameters show how the score categories are obtained for each analytic scoring item. For instance, Figures 4 and 5 show item characteristic curves (ICC), which illustrate the probabilistic curve based on Eq. (6), for the *overall* and *content* items under the one-dimensional setting. In these figures, the horizontal axis indicates the latent ability of the examinees, and the vertical axis indicates the probability P_{nmk}. Note that the horizontal axis shows values for ability θ around zero because, as was explained in section 4.1, the distribution of the ability estimates follows a normal distribution with zero mean. Figures 4 and 5 show that examinees with a higher ability have a greater probability of obtaining a high score. Moreover, scores of 2, 6, 8, 10, 11, and 12 for the *overall* item are likely to be used while scores of 3, 4, 5, and 7 tend to be avoided. In the *content* item, a score of 2 tends to be avoided slightly. It is in this way that the difficulty parameters enable us to make an interpretation of how the score categories are used for the analytic scoring items. Note that although we highlighted the one-dimensional model results here, the difficulty parameters in the one- and two-dimensional models are similar and, thus, provide similar interpretations.

Our model predicts analytic scores based on these characteristics of the analytic scoring items and on estimations of the examinees' abilities. Table 6 shows examples of examinees' latent abilities and the predicted analytic scores estimated by the one-dimensional variant of our model. Table 6 indicates that our model tends to provide higher scores for essays written by examinees with higher abilities. Furthermore, comparing Table 6 with Figures 4 and 5, we can confirm that the predicted scores for the *overall* and *content* items follow the ICCs reasonably well. For example, examinee 4, who had a nearly zero value of θ_n, obtained an overall score of 8 and a content score of 4. The ICCs show high response probabilities for these scores around $\theta_n = 0$.

These results demonstrate that our model enables us to interpret predictions that are based on the IRT-layer model parameters.

6 Conclusions

In this study, we proposed a new neural-based analytic AES model that incorporates a multidimensional IRT model. Through experiments using the well-known benchmark datasets ASAP and ASAP++, we demonstrated that, compared to the latest conventional model, our model succeeds in improving interpretability without significantly losing performance.

Our experiments also suggested that one- or two-dimensional abilities can sufficiently explain the

multiple analytic scores, including the overall score. This is an important finding suggesting that the analytic scoring items in the dataset may fail to measure multiple aspects of ability. This is undesirable because the objective of analytic scoring is to evaluate multiple aspects of ability.

Future studies will be required to evaluate our model using various datasets, including other benchmark datasets. Moreover, another challenge to address in future work is to develop an extension of our model for cross-prompt tasks.

Acknowledgements

This work was supported by JSPS KAKENHI Grant Numbers 19H05663, 20K20817, and 21H00898.

References

Dimitrios Alikaniotis, Helen Yannakoudakis, and Marek Rei. 2016. Automatic text scoring using neural networks. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 715–725. Association for Computational Linguistics.

Yigal Attali and Jill Burstein. 2006. Automated essay scoring with e-rater® v.2. *The Journal of Technology, Learning and Assessment*, 4(3):1–30.

Jakob B Bjorner, Mark Kosinski, and John E Ware Jr. 2003. The feasibility of applying item response theory to measures of migraine impact: A re-analysis of three clinical studies. *Quality of Life Research*, 12(8):887–902.

Hongbo Chen and Ben He. 2013. Automated essay scoring by maximizing human-machine agreement. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*, pages 1741–1752.

Mihai Dascalu, Wim Westera, Stefan Ruseti, Stefan Trausan-Matu, and Hub Kurvers. 2017. Readerbench learns dutch: building a comprehensive automated essay scoring system for dutch language. In *International Conference on Artificial Intelligence in Education*, pages 52–63. Springer.

Yann Dauphin, Harm de Vries, and Yoshua Bengio. 2015. Equilibrated adaptive learning rates for non-convex optimization. In *Advances in Neural Information Processing Systems*, volume 28, pages 1504–1512. Curran Associates, Inc.

Fei Dong and Yue Zhang. 2016. Automatic features for essay scoring–an empirical study. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 1072–1077.

Fei Dong, Yue Zhang, and Jie Yang. 2017. Attention-based recurrent convolutional neural network for automatic essay scoring. In *Proceedings of the 21st Conference on Computational Natural Language Learning*, pages 153–162.

Inan Deniz Erguvan and Beyza Aksu Dunya. 2020. Analyzing rater severity in a freshman composition course using many facet rasch measurement. *Language Testing in Asia*, 10(1):1–20.

Youmna Farag, Helen Yannakoudakis, and Ted Briscoe. 2018. Neural automated essay scoring and coherence modeling for adversarially crafted input. arXiv.

Peter Hastings, Simon Hughes, and M Anne Britt. 2018. Active learning for improving machine learning of student explanatory essays. In *International Conference on Artificial Intelligence in Education*, pages 140–153. Springer.

Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long Short-Term Memory. *Neural Computation*, 9(8):1735–1780.

Mohamed A. Hussein, Hesham A. Hassan, and Mohammad Nassef. 2020. A trait-based deep learning automated essay scoring system with adaptive feedback. *International Journal of Advanced Computer Science and Applications*, 11(5):287–293.

Mohamed Abdellatif Hussein, Hesham Hassan, and Mohammad Nassef. 2019. Automated language essay scoring systems: A literature review. *PeerJ Computer Science*, 5:e208.

Cancan Jin, Ben He, Kai Hui, and Le Sun. 2018. Tdnn: a two-stage deep neural network for prompt-independent automated essay scoring. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1088–1097.

Zixuan Ke and Vincent Ng. 2019. Automated essay scoring: A survey of the state of the art. In *IJCAI*, volume 19, pages 6300–6308.

F.M. Lord. 1980. *Applications of item response theory to practical testing problems*. Erlbaum Associates.

Sandeep Mathias and Pushpak Bhattacharyya. 2018. ASAP++: Enriching the ASAP automated essay grading dataset with essay attribute scores. Proceedings of the Eleventh International Conference on Language Resources and Evaluation.

Sandeep Mathias and Pushpak Bhattacharyya. 2020. Can neural networks automatically score essay traits? In *Proceedings of the Fifteenth Workshop on Innovative Use of NLP for Building Educational Applications*, pages 85–91.

Farjana Sultana Mim, Naoya Inoue, Paul Reisert, Hiroki Ouchi, and Kentaro Inui. 2019. Unsupervised learning of discourse-aware text representation for essay

scoring. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics: Student Research Workshop*, pages 378–385.

J.C. Nunnally and I.H. Bernstein. 1994. The assessment of reliability. *Psychometric Theory*, 3:248–292.

Jeffrey Pennington, Richard Socher, and Christopher D Manning. 2014. Glove: Global vectors for word representation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing*, pages 1532–1543.

Peter Phandi, Kian Ming A Chai, and Hwee Tou Ng. 2015. Flexible domain adaptation for automated essay scoring using correlated linear regression. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 431–439.

Robert Ridley, Liang He, Xin-yu Dai, Shujian Huang, and Jiajun Chen. 2021. Automated cross-prompt scoring of essay traits. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 35, pages 13745–13753.

Robert Ridley, Liang He, Xinyu Dai, Shujian Huang, and Jiajun Chen. 2020. Prompt agnostic essay scorer: A domain generalization approach to cross-prompt automated essay scoring. arXiv.

Pedro Uria Rodriguez, Amir Jafari, and Christopher M Ormerod. 2019. Language models and automated essay scoring. arXiv.

Frank E Saal, Ronald G Downey, and Mary A Lahey. 1980. Rating the ratings: Assessing the psychometric quality of rating data. *Psychological bulletin*, 88(2):413.

Kaveh Taghipour and Hwee Tou Ng. 2016. A neural approach to automated essay scoring. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 1882–1891.

Yi Tay, Minh C Phan, Luu Anh Tuan, and Siu Cheung Hui. 2018. Skipflow: Incorporating neural coherence features for end-to-end automatic text scoring. In *Thirty-Second AAAI Conference on Artificial Intelligence*, pages 5948–5955.

Masaki Uto. 2021a. Accuracy of performance-test linking based on a many-facet rasch model. *Behavior Research Methods*, 53(4):1440–1454.

Masaki Uto. 2021b. A multidimensional generalized many-facet rasch model for rubric-based performance assessment. *Behaviormetrika*, 48(2):425–457.

Masaki Uto. 2021c. A review of deep-neural automated essay scoring models. *Behaviormetrika*, 48(2):1–26.

Masaki Uto and Masashi Okano. 2021. Learning automated essay scoring models using item-response-theory-based scores to decrease effects of rater biases. *IEEE Transactions on Learning Technologies*, 14(6):763–776.

Masaki Uto and Maomi Ueno. 2020. A generalized many-facet rasch model and its bayesian estimation using hamiltonian monte carlo. *Behaviormetrika*, 47(2):469–496.

Masaki Uto, Yikuan Xie, and Maomi Ueno. 2020. Neural automated essay scoring incorporating handcrafted features. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 6077–6088.

Lihua Yao and Richard D. Schwarz. 2006. A multidimensional partial credit model with associated item and test statistics: An application to mixed-format tests. *Applied Psychological Measurement*, 30(6):469–492.

Lili Yao, Shelby J Haberman, and Mo Zhang. 2019. Prediction of writing true scores in automated scoring of essays by best linear predictors and penalized best linear predictors. *ETS Research Report Series*, 2019(1):1–27.

DP-Rewrite: Towards Reproducibility and Transparency in Differentially Private Text Rewriting

Timour Igamberdiev[1] and **Thomas Arnold**[2] and **Ivan Habernal**[1]
[1]Trustworthy Human Language Technologies
[2]Ubiquitous Knowledge Processing Lab
Department of Computer Science, Technical University of Darmstadt
{timour.igamberdiev,ivan.habernal}@tu-darmstadt.de
arnold@ukp.informatik.tu-darmstadt.de
www.trusthlt.org www.ukp.tu-darmstadt.de

Abstract

Text rewriting with differential privacy (DP) provides concrete theoretical guarantees for protecting the privacy of individuals in textual documents. In practice, existing systems may lack the means to validate their privacy-preserving claims, leading to problems of transparency and reproducibility. We introduce `DP-Rewrite`, an open-source framework for differentially private text rewriting which aims to solve these problems by being modular, extensible, and highly customizable. Our system incorporates a variety of downstream datasets, models, pre-training procedures, and evaluation metrics to provide a flexible way to lead and validate private text rewriting research. To demonstrate our software in practice, we provide a set of experiments as a case study on the ADePT DP text rewriting system, detecting a privacy leak in its pre-training approach. Our system is publicly available, and we hope that it will help the community to make DP text rewriting research more accessible and transparent.

1 Introduction

Protecting the privacy of individuals has been gaining attention in NLP. One particular setup is text rewriting using local differential privacy (DP) (Dwork and Roth, 2013), which provides probabilistic guarantees of 'how much' privacy can be lost in the worst case if an individual gives us their piece of text that has been rewritten with DP. For instance, given a text "I want to fly from Newark to Cleveland on Friday", the system may rewrite it as "Flights from Los Angeles to Houston this week". Only a few recent works have touched on this challenging topic. For example, Krishna et al. (2021) proposed ADePT: A text rewriting system based on the Laplace mechanism. However, it turned out that their DP method was formally flawed (Habernal, 2021). We also see another recent approach, DP-VAE (Weggenmann et al., 2022), which shows results that look surprisingly good for the level of guaranteed privacy. However, neither ADePT nor DP-VAE published their source codes, so the community has no means to perform any empirical checks to validate the privacy-preserving claims. Therefore, the lack of transparency and reproducibility is the main obstacle to the accountability of DP text-rewriting systems.

We asked whether an open, modular, easily extensible, and highly customizable framework for differentially private text rewriting could help the community gain further insight into the utility and potential pitfalls of such systems. We hypothesize that by integrating various downstream datasets, models, pre-training procedures, and evaluation metrics into one software package, we improve the transparency, accountability, and reproducibility of research in differentially private text rewriting.

Our main contributions are twofold. First, this demo paper presents `DP-Rewrite`, an open-source framework for differentially private text rewriting experiments. It includes a correct reimplementation of ADePT as a baseline, integrates pre-training on several datasets, and allows us to easily perform downstream experiments with varying privacy guarantees by adjusting the privacy budget ε. Second, `DP-Rewrite` allows us to easily detect another privacy leak in the approach proposed in ADePT, namely in the pre-training strategy of the autoencoder, with the system memorizing the input data. We demonstrate this in detail as a use case of `DP-Rewrite` in Section 4.[1]

2 Related Work

Although the problem of simple data redaction is a widely researched field with several promising approaches (Hill et al., 2016; Lison et al., 2021),

[1]Our project is available at https://github.com/trusthlt/dp-rewrite.

Proceedings of the 29th International Conference on Computational Linguistics, pages 2927–2933
October 12–17, 2022.

the related problem of private text transformation is still largely unexplored.

We only briefly sketch the main ideas of local differentially private algorithms in text rewriting. For a longer introduction to DP see, e.g., Habernal (2022); Senge et al. (2021); Igamberdiev and Habernal (2022). Let $x, x' \in \mathcal{X}$ be two data points such as texts or vectors, each belonging to a different person. In DP terminology, x and x' are neighboring datasets, as they differ by one person (Desfontaines and Pejó, 2020). A (local) DP algorithm $\mathcal{M} : \mathcal{X} \to \mathcal{Y}$ is a function that takes any single data point $x \in \mathcal{X}$ and produces its 'privatized' version $y \in \mathcal{Y}$ which might be an arbitrary object, such as a text or a vector. Privatization is achieved by introducing randomness in $\mathcal{M}$. As a result, $(\varepsilon, 0)$-local DP guarantees that for any two neighboring datasets x, x' and any output y

$$\ln \left[\frac{\Pr(\mathcal{M}(x) = y)}{\Pr(\mathcal{M}(x') = y)} \right] \leq \varepsilon \qquad (1)$$

where ε is the privacy budget; the lower, the better privacy is guaranteed. If a text rewriting algorithm satisfies the local DP, it limits the probability of revealing the true text x after observing the privatized text y.

Krishna et al. (2021) proposed ADePT, a DP text rewriting system. It consists of an auto-encoder that learns a compressed latent representation of text, and a DP rewriter that uses the trained auto-encoder, adds Laplace noise to the latent representation vector, and generates the privatized text. Due to a formal error in the scale of the Laplace noise, ADePT violated differential privacy (Habernal, 2021).

Bo et al. (2021) proposed a text rewriting approach that generates words from a latent representation while adding DP noise. However, unlike holistic text rewriting with DP, perturbing text only at the word level does not protect against privacy attacks (Mattern et al., 2022).

Even more current, Weggenmann et al. (2022) proposed an end-to-end approach to text anonymization using a DP autoencoder, claiming to produce coherent texts of high privacy standards. However, several key aspects of the experiments lack a detailed description, while their results look surprisingly good. Since the code base is not public, we cannot reproduce or reimplement their approach, and we cannot prove or refute our suspicions.

3 Description of software

The goal of our system is to provide a seamless way to perform differentially private text rewriting, both on existing and custom datasets. A user can either load a dataset that we provide out-of-the-box, or use a custom one. In addition, we want to make it fast and convenient to run experiments for existing methodologies in DP text rewriting (e.g. ADePT), as well as the ability to integrate novel approaches. For this, we have a general *autoencoder* class based on which out-of-the-box and custom models are built. In this sense, our software is designed to be open and modular, where the researcher can swap out existing components to run a variety of experiments, as well as use the software for one's own privatized text rewriting needs.

The core architecture of our system can be seen in Figure 1. We divide the experiments into three distinct **modes**, *pre-training*, *rewriting*, and *downstream*. For all three, the pipeline begins with a dataloader which can either be a dataset provided in the framework, or a custom dataset specified by the user. Additionally, a rewritten dataset can be loaded for downstream experiments. The loaded dataset is then preprocessed according to user-specified parameters and the user's selected model, split into different procedures depending on the model type (e.g. RNN-based, transformer-based). The model is then initialized, optionally from an existing checkpoint. At this point, the main experiment is run based on the specified mode, either (1) pre-training the autoencoder, (2) using an existing checkpoint to rewrite the dataset, or (3) running a downstream model on an original or rewritten dataset. For each mode, a variety of evaluations are available, such as BLEU (Papineni et al., 2002) and BERTScore (Zhang et al., 2019) for pre-training and rewriting, and various classification metrics (e.g. F_1 score) for downstream experiments. The differential privacy component is incorporated during the rewriting phase for systems such as ADePT, although our framework also allows to incorporate it during the pre-training stage.

4 Case study

We present here a case study that demonstrates the process of using our framework and provides insights into the ADePT system, for which we provide an implementation in the software. Our goal is to investigate the difference in rewritten texts and downstream evaluations when we pre-

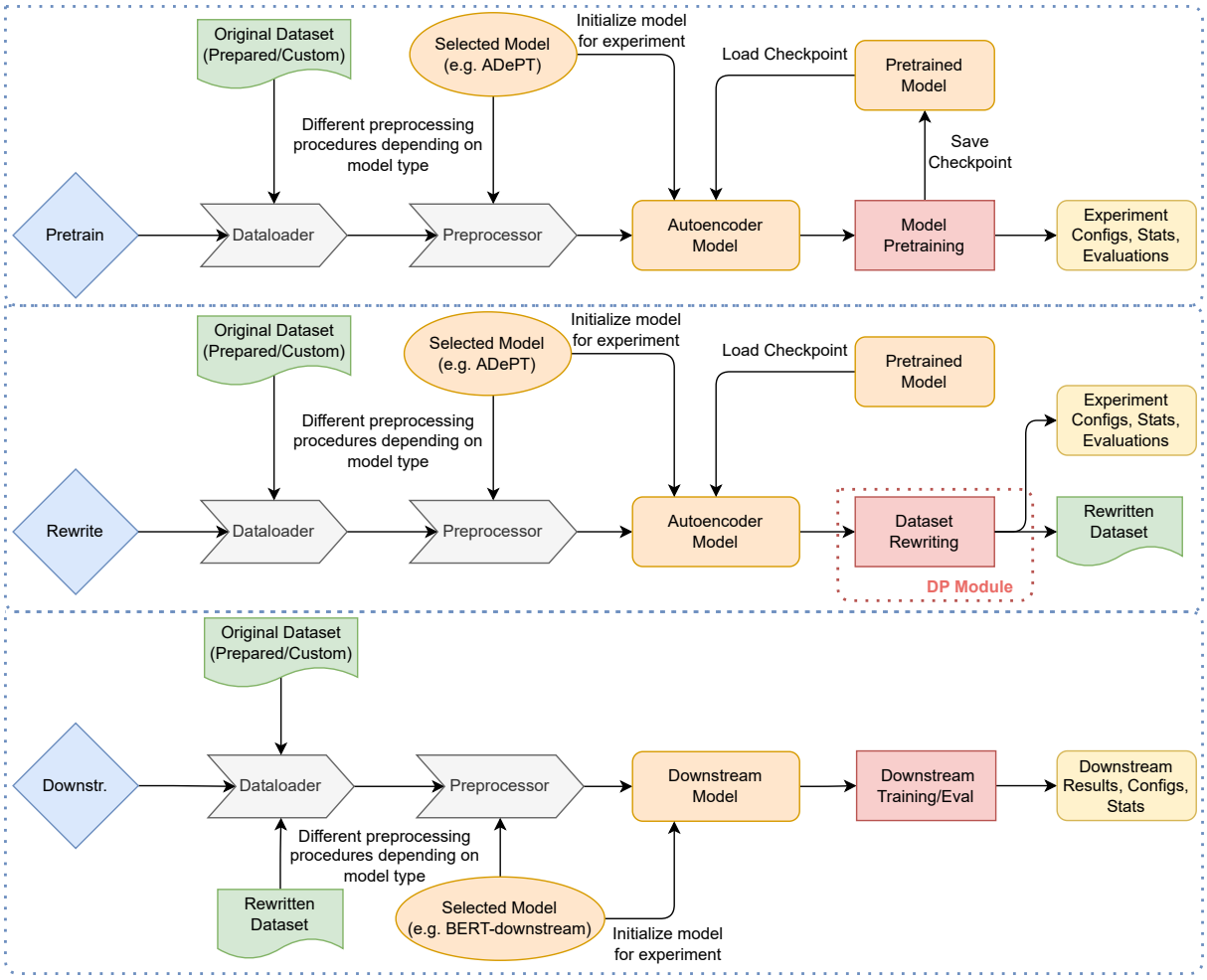

Figure 1: Overall structure of DP-Rewrite. Colors represent groupings of similar components. Blue: Experiment mode. Grey: Dataset preparation. Green: Datasets (original/rewritten). Orange: Model-related components. Red: Main experiment loop. Yellow: Additional experiment outputs.

train an autoencoder on one dataset and use this to rewrite another dataset. If we notice a lot of tokens from the dataset used for pre-training in the rewritten dataset, as well as comparatively higher downstream scores when pre-training and rewriting on the same dataset, then we can be certain of another form of privacy leakage in ADePT.

4.1 Datasets

As in Krishna et al. (2021), we use the ATIS (Dahl et al., 1994) and Snips (Coucke et al., 2018) datasets to conduct experiments on an intent classification task in English. For both datasets, we use the same train/validation/test split provided by Goo et al. (2018), with 4,478 training, 500 validation and 893 test examples for ATIS, and 13,084 training, 700 validation and 700 test examples for Snips. There are a total of 26 intent labels in ATIS and 7 in Snips.

4.2 Implementation

We start our experiment pipeline by pre-training two models, one on ATIS (1) and the other on Snips (2), in both cases using the training split. For pre-training, we set the vocabulary to the maximum number of words from the training set. As in ADePT, we do not incorporate a differential privacy component during pre-training, although we clip encoder hidden representations with a clipping constant value of 5. We limit sequence lengths to a maximum of 20 tokens, pre-training for 200 epochs with a learning rate of 0.003. In contrast to ADePT, we do not use the ℓ_2 norm for clipping due to issues in privacy guarantees outlined by Habernal (2021) and instead follow the suggested fix for the method by clipping using the ℓ_1 norm.

We then use these two models for rewriting, applying both pre-trained models for rewriting the training and validation partitions of ATIS and

Snips, resulting in four rewriting settings in total. For each setting, we rewrite using five ε values, $\infty, 1000, 100, 10$, and 1. We use the same clipping constant value of 5 as in pre-training.

See Appendix B for details of the downstream experiment setup.

4.3 Results and analysis

Our results can be seen in Figure 2. We observe the main patterns as follows. First and most importantly, datasets rewritten using a model that was pre-trained on the same dataset generally show better downstream results than datasets rewritten using a model pre-trained on a different dataset. For instance, at $\varepsilon = 1,000$, rewritten Snips from a model pre-trained on Snips has an F_1 score of 0.94, while rewritten Snips from a model pre-trained on ATIS has only 0.20. In fact, this is true even at $\varepsilon = \infty$ (non-private setting), without any added noise (e.g. 0.94 F_1 pre-trained Snips, rewritten Snips vs. 0.18 F_1 pre-trained ATIS, rewritten Snips), since for the latter case the model ends up rewriting the dataset that was pre-trained on, having memorized many of its examples. This can be seen in Figure 3, where the rewritten sentences appear to have no resemblance to the original dataset used for rewriting, but are very similar to the data used for pre-training.

Next, as expected, the results decrease for all configurations as the privacy budget ε decreases, except for rewritten ATIS from a model pre-trained on Snips, where results are low for all ε values, probably due to the same reasons as shown in Figure 3. At the lower ε values of 10 and 1, performance is very low for all configurations. Since there is so much noise added to the encoder hidden representations, the utility of ADePT's rewriting is severely diminished, for any data inputs.

Finally, compared to running downstream experiments on the original dataset, Snips rewritten with a model pre-trained on Snips shows about the same results at high ε values (e.g. 0.94 F_1 pre-trained Snips, rewritten Snips vs. 0.95 F_1 original Snips for $\varepsilon = \infty$). ATIS rewritten with a model pre-trained on ATIS shows lower results in this case (e.g. 0.73 F_1 pre-trained ATIS, rewritten ATIS vs. 0.87 F_1 original ATIS for $\varepsilon = \infty$). We speculate that since ATIS is a smaller dataset, there are less data points to effectively pre-train ADePT for the autoencoding task. We additionally report random and majority baselines in Appendix A on the original datasets for comparison.

We have thus shown that, despite fixing the theoretical privacy guarantees of ADePT, the pre-training procedure still results in privacy leakage, with rewritten datasets exposing a lot of information from the dataset used for pre-training. As a result, downstream performance is inflated if the datasets for pre-training and rewriting are the same.

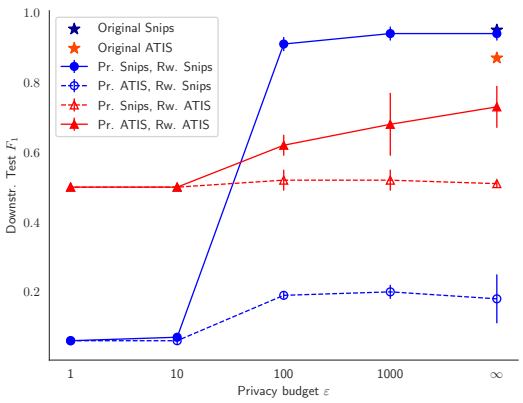

Figure 2: Downstream macro-averaged F_1 results for case study experiments with pre-trained and rewritten Snips/ATIS datasets, as well as comparisons with results on the original datasets ("Original Snips" and "Original ATIS"). Lower ε corresponds to better privacy.

Snips rewritten from ATIS $\varepsilon = 1000$	
Original Snips doc.	listen to westbam alumb allergic on google music
Rewritten Snips doc.	how many people fly on after a turboprop airport
ATIS doc. similar	how many people fly on a turboprop airport
ATIS rewritten from Snips $\varepsilon = 1000$	
Original ATIS doc.	what flights leave from phoenix
Rewritten ATIS doc.	start playing my disney spring
Snips doc. similar	start playing my disney playlist

Figure 3: Sample rewritten texts showing memorization by ADePT model when pre-training and rewriting on different datasets. For a given document in the original dataset ("Original Snips/ATIS doc."), its rewritten version by the model ("Rewritten Snips/ATIS doc.") has no resemblance to it, but is very similar to another document from the pre-trained dataset ("ATIS/Snips doc. similar").

5 Conclusion

We introduced `DP-Rewrite`, an open-source framework for differentially private text rewriting experiments. We have demonstrated a sample use-case for our framework, which allows us to detect privacy leakage in the pre-training procedure of

the ADePT system, an example of how the modular and customizable nature of the software allows for transparency and reproducibility in DP text-rewriting research. `DP-Rewrite` is continuing to be under active development, and we are incorporating new datasets and private text rewriting methodologies as they are released. We welcome feedback from the community.

Acknowledgements

The independent research group TrustHLT is supported by the Hessian Ministry of Higher Education, Research, Science and the Arts. This project was partly supported by the National Research Center for Applied Cybersecurity ATHENE. Thanks to Johannes Gaese and Jonas Rikowski who contributed in the initial implementation phases.

References

Haohan Bo, Steven H. H. Ding, Benjamin C. M. Fung, and Farkhund Iqbal. 2021. ER-AE: Differentially private text generation for authorship anonymization. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 3997–4007, Online. Association for Computational Linguistics.

Alice Coucke, Alaa Saade, Adrien Ball, Théodore Bluche, Alexandre Caulier, David Leroy, Clément Doumouro, Thibault Gisselbrecht, Francesco Caltagirone, Thibaut Lavril, et al. 2018. Snips voice platform: an embedded spoken language understanding system for private-by-design voice interfaces. *arXiv preprint arXiv:1805.10190*.

Deborah A Dahl, Madeleine Bates, Michael K Brown, William M Fisher, Kate Hunicke-Smith, David S Pallett, Christine Pao, Alexander Rudnicky, and Elizabeth Shriberg. 1994. Expanding the scope of the atis task: The atis-3 corpus. In *Human Language Technology: Proceedings of a Workshop held at Plainsboro, New Jersey, March 8-11, 1994*.

Damien Desfontaines and Balázs Pejó. 2020. SoK: Differential privacies. *Proceedings on Privacy Enhancing Technologies*, 2020(2):288–313.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*.

Cynthia Dwork and Aaron Roth. 2013. The Algorithmic Foundations of Differential Privacy. *Foundations and Trends® in Theoretical Computer Science*, 9(3-4):211–407.

Chih-Wen Goo, Guang Gao, Yun-Kai Hsu, Chih-Li Huo, Tsung-Chieh Chen, Keng-Wei Hsu, and Yun-Nung Chen. 2018. Slot-gated modeling for joint slot filling and intent prediction. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 2 (Short Papers)*, pages 753–757.

Ivan Habernal. 2021. When differential privacy meets NLP: The devil is in the detail. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 1522–1528, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Ivan Habernal. 2022. How reparametrization trick broke differentially-private text representation learning. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 771–777, Dublin, Ireland. Association for Computational Linguistics.

Steven Hill, Zhimin Zhou, Lawrence K Saul, and Hovav Shacham. 2016. On the (in) effectiveness of mosaicing and blurring as tools for document redaction. *Proc. Priv. Enhancing Technol.*, 2016(4):403–417.

Timour Igamberdiev and Ivan Habernal. 2022. Privacy-Preserving Graph Convolutional Networks for Text Classification. In *Proceedings of the Language Resources and Evaluation Conference*, pages 338–350, Marseille, France. European Language Resources Association.

Satyapriya Krishna, Rahul Gupta, and Christophe Dupuy. 2021. ADePT: Auto-encoder based differentially private text transformation. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*, pages 2435–2439, Online. Association for Computational Linguistics.

Pierre Lison, Ildikó Pilán, David Sanchez, Montserrat Batet, and Lilja Øvrelid. 2021. Anonymisation models for text data: State of the art, challenges and future directions. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 4188–4203, Online. Association for Computational Linguistics.

Justus Mattern, Benjamin Weggenmann, and Florian Kerschbaum. 2022. The Limits of Word Level Differential Privacy. *arXiv preprint*.

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. Bleu: a method for automatic evaluation of machine translation. In *Proceedings of the 40th annual meeting of the Association for Computational Linguistics*, pages 311–318.

Manuel Senge, Timour Igamberdiev, and Ivan Habernal. 2021. One size does not fit all: Investigating strategies for differentially-private learning across NLP tasks. *arXiv preprint*.

Benjamin Weggenmann, Valentin Rublack, Michael Andrejczuk, Justus Mattern, and Florian Kerschbaum. 2022. DP-VAE: Human-Readable Text Anonymization for Online Reviews with Differentially Private Variational Autoencoders. In *Proceedings of the ACM Web Conference 2022*, pages 721–731, Virtual Event. ACM.

Tianyi Zhang, Varsha Kishore, Felix Wu, Kilian Q Weinberger, and Yoav Artzi. 2019. Bertscore: Evaluating text generation with bert. *arXiv preprint arXiv:1904.09675*.

A Detailed results of the case study

Pretr. Dat.	Rewr. Dat.	ε	Test F_1
Snips	Snips	∞	0.94 (0.02)
Snips	Snips	1,000	0.94 (0.02)
Snips	Snips	100	0.91 (0.02)
Snips	Snips	10	0.07 (0.01)
Snips	Snips	1	0.06 (0.00)
ATIS	Snips	∞	0.18 (0.07)
ATIS	Snips	1,000	0.20 (0.02)
ATIS	Snips	100	0.19 (0.01)
ATIS	Snips	10	0.06 (0.01)
ATIS	Snips	1	0.06 (0.01)
Snips	ATIS	∞	0.51 (0.01)
Snips	ATIS	1,000	0.52 (0.03)
Snips	ATIS	100	0.52 (0.03)
Snips	ATIS	10	0.50 (0.01)
Snips	ATIS	1	0.50 (0.01)
ATIS	ATIS	∞	0.73 (0.06)
ATIS	ATIS	1,000	0.68 (0.09)
ATIS	ATIS	100	0.62 (0.03)
ATIS	ATIS	10	0.50 (0.01)
ATIS	ATIS	1	0.50 (0.01)
Snips Orig.			0.95 (0.01)
ATIS Orig.			0.87 (0.03)
Snips Rand.			0.14
ATIS Rand.			0.01
Snips Maj.			0.03
ATIS Maj.			0.13

Table 1: Downstream macro-averaged F_1 results for case study experiments with pre-trained and rewritten Snips/ATIS datasets. We additionally show results on the original datasets, as well as random and majority baselines. Test F_1 shown as "mean (standard deviation)" over five runs with different random seeds. Lower ε corresponds to better privacy.

B Downstream experiment setup

For downstream experiments, we use a pre-trained BERT model (Devlin et al., 2018), with an additional feedforward layer that takes the mean of the last hidden states as input and predicts the output label. We use the rewritten training and validation sets for each configuration, and the original test sets for final evaluation. We run each configuration with five different random seeds and report the mean and standard deviation.

Harnessing Abstractive Summarization for Fact-Checked Claim Detection

Varad Bhatnagar[1], Diptesh Kanojia[2,3], Kameswari Chebrolu[1]
[1]Department of Computer Science and Engineering, IIT Bombay, India
[2]Surrey Institute for People-Centred AI, [3]Department of Computer Science
[2,3]University of Surrey, United Kingdom
[1]{varadhbhatnagar,chebrolu}@cse.iitb.ac.in
[2,3]d.kanojia@surrey.ac.uk

Abstract

Social media platforms have become new battlegrounds for anti-social elements, with misinformation being the weapon of choice. Fact-checking organizations try to debunk as many claims as possible while staying true to their journalistic processes but cannot cope with its rapid dissemination. We believe that the solution lies in partial automation of the fact-checking life cycle, saving human time for tasks which require high cognition. We propose a new workflow for efficiently detecting previously fact-checked claims that uses abstractive summarization to generate crisp queries. These queries can then be executed on a general-purpose retrieval system associated with a collection of previously fact-checked claims. We curate an abstractive text summarization dataset comprising noisy claims from Twitter and their gold summaries. It is shown that retrieval performance improves 2x by using popular out-of-the-box summarization models and 3x by fine-tuning them on the accompanying dataset compared to verbatim querying. Our approach achieves Recall@*5* and MRR of 35% and 0.3, compared to baseline values of 10% and 0.1, respectively. Our dataset, code, and models are available publicly here.

1 Introduction

Social media is increasingly used for business, entertainment, and political discourse, thus, encouraging users to produce and consume large volumes of information that may not always be accurate. Due to a lack of digital awareness, the masses often believe and forward such disputed claims in their social circles. Such spread of misinformation often culminates in incidents which cause damage to life and property. It is well documented that misinformation is used as a tool by political agents to slander their opposition (Allcott and Gentzkow, 2017) and influence the opinion of the masses. It becomes furthermore dangerous when such claims pertain to religious beliefs, often leading to violence and mob lynchings[1]. In the era of COVID-19, unverified medical advice has also been circulated on social media (Shahi et al., 2021) which has already led to various health hazards.

Social media platforms have undertaken concerted efforts to tackle the fake news epidemic by enforcing strict policies to weed out unverified and sensitive content and ban habitual offenders. Journalistic organizations such as Alt News[2], Factly[3], Boom Live[4] and Snopes[5] among others are also fighting this problem by publishing fact-checking articles investigating the veracity of viral dubious claims. These articles detail the journalistic procedures followed to fact-check the claim along with suitable references.

Numerous researchers are working on AI-based solutions for fact-checking claims. Many datasets (Thorne et al., 2018; Sathe et al., 2020; Fan et al., 2020; Schuster et al., 2021) have been released to train models which can automate subtasks such as claim verification, evidence retrieval and assigning a verdict in a fact-checking workflow. A critical and insufficiently researched step in the fact-checking workflow is- *detecting whether a claim has been fact-checked previously*. This is a repetitive task with immense scope for automation, shrinking the turnaround time for a claim and ensuring that human efforts are put to better use on tasks involving higher cognition, such as assigning a verdict. In literature, learning-to-rank models (Shaar et al., 2020; Vo and Lee, 2020; Mansour et al., 2022) have been proposed for this step, which on being queried, produce a ranked list of results from a closed dataset of previously verified claims. Dozens of fact-checking articles are being published every hour around the world. It is

[1]Article on Mob Lynching: Washington Post
[2]AltNews: Website
[3]Factly: Website
[4]Boom Live: Website
[5]Snopes: Website

Proceedings of the 29th International Conference on Computational Linguistics, pages 2934–2945
October 12–17, 2022.

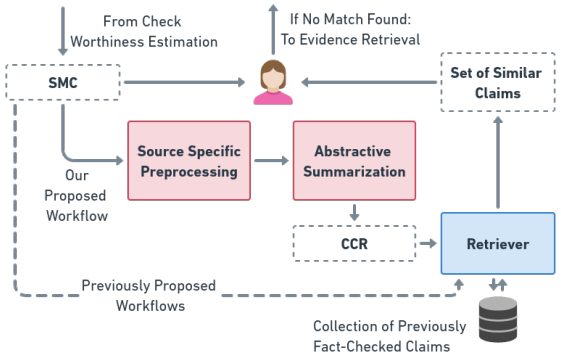

Figure 1: **Proposed Workflow**. In this work, we use Fact Check Explorer as a black box Retriever. The workflow proposed by previous works is denoted by a dotted path.

difficult for journalistic organizations to maintain such a large real-time collection of fact-checked articles and claims, thus, making such an approach infeasible in real-world scenarios.

We propose a novel workflow (as shown in Figure 1) for detecting previously fact-checked claims. In this work, we use Google's *Fact Check Explorer*, a cross-publisher, cross-language search engine for previously fact-checked articles, as a black box retriever. As social media platforms contribute a great deal to spreading misinformation, we deal with naturally occurring textual claims on Twitter in this study as opposed to artificial, well defined and structured claims (Thorne et al., 2018; Aly et al., 2021). Instead of querying using verbatim claims, which are noisy, it is proposed that abstractive text summarization be used as a precursor to querying to generate clear, succinct queries capturing the claim in a minimum number of words. Figure 1 represents a subpart of the complete fact-checking pipeline (Barrón-Cedeno et al., 2020) with our proposed changes in red. *CCR* and *SMC* are defined in Section 3. In literature, no dataset exists for abstractive summarization of tweets, and no attempts have been made to address this problem using the *Fact Check Explorer* to the best of our knowledge. Our contributions can be distilled into the following:

1. **Workflow**: A novel workflow for detecting previously-fact checked claims at scale.
2. **Dataset**: An abstractive summarization dataset[6] for tweets in the Indian context.
3. **Models**: Popular and large pre-trained abstractive summarization models, fine-tuned under supervision on this data, which can be used for other purposes involving tweet summarization.
4. **Experimental Study and Analysis**: We also perform quantitative and qualitative analysis for various outputs in our proposed workflow, including an analysis of generated summaries.

The rest of this paper is organized as follows: Section 2 discusses related work, Section 3 presents the dataset, Section 4 discusses the approach, evaluation metrics and the experimental setup. The results are presented and analysed in Section 5 followed by Section 6 which concludes the work and proposes future research directions. Section 8 and 9 discuss the ethical considerations and limitations.

2 Related Work

Sharma et al. (2019) describe the menace of misinformation on the Internet and summarize mitigation techniques and available datasets in this domain. Available intelligent technologies to assist the process of fact-checking are surveyed by Nakov et al. (2021a). This work highlights the partial overlap between current research endeavours and fact-checkers desiderata over the life cycle of a claim in a fact-checking pipeline.

A general-purpose four-step automatic fact-checking pipeline is presented by Barrón-Cedeno et al. (2020). The task of determining if a claim has been previously fact-checked is the second step in the pipeline. This problem is addressed in a series of open challenges (Shaar et al., 2021c) at *Checkthat!* workshop (Barrón-Cedeno et al., 2020; Nakov et al., 2021b, 2022) as part of CLEF [7]. Shaar et al. (2020) collect, annotate and release datasets of claim pairs and evidence sets, sourced from Politifact[8] and Snopes for solving this task. They develop and demonstrate the robustness of BM25 and BERT (Devlin et al., 2019) based learning to rank models on their dataset for this task. Vo and Lee (2020); Mansour et al. (2022) also propose variants of a ranking approach to solve this problem. Further, Shaar et al. (2021b) work with data from political debates and model the context of a claim and illustrate the positive impact this has in determining if it has been previously fact-checked. Shaar et al. (2021a) publish a dataset and develop a system for detecting all previously fact-checked claims in a lengthy document.

[6]Data and models are made available here: https://github.com/varadhbhatnagar/FC-Claim-Det/

[7]CLEF: Website

[8]Politifact: Website

Text summarization has been used to enable verdict explainability in automatic fact-checking (Mishra et al., 2020; Stammbach and Ash, 2020) but it hasn't been used for denoising tweets, to the best of our knowledge.

Tchechmedjiev et al. (2019) publish the *ClaimsKG* Knowledge Graph, containing 28K fact-checked claims and their metadata such as sources, truth value and entities. Structured queries can be executed on this knowledge graph, enabling exploration and information discovery. However, it does not provide any mechanism to check if a claim has been previously fact-checked. *Fact Check Explorer*[9] is a tool developed by Google which provides browsing and searching capability for already fact-checked articles which have the ClaimReview Schema[10] embedded. There are performance limitations associated with this tool in the face of long and complex queries.

3 Dataset

The following terms are defined for lucid perusal of this work:

1. **Social Media Claim (SMC):** A social media post (tweet, in this work) containing a claim in need of fact-checking. It is analogous to the output of the first step (*check worthiness estimation*) in the automatic fact-checking pipeline presented by Barrón-Cedeno et al. (2020).
2. **Fact Checked Article (FCA):** An article published by a fact-checking organization accepting or refuting a claim[11].
3. **Summary of Claim Review (SCR):** A short summary of the claim added by the publishing organization as part of the ClaimReview Schema associated with every FCA. Our use of the term SCR is the same as *VerClaim* coined by Shaar et al. (2020).
4. **Condensed Claim Representation (CCR):** A summary of the SMC generated using trained models.

3.1 Dataset Curation

In this work, we focus on FCAs published by Indian organizations between 2018 and 2022. FCAs from the following IFCN[12] certified organizations: 1) Alt News, 2) BoomLive, 3) India Today, 4) The Logical Indian, 5) The Quint, 6) Factchecker, 7) FactCrescendo, 8) Vishwas News, 9) PolitiFact, 10) Snopes, and 11) Factcheck.org, were retrieved. In order to make our dataset diverse, some FCAs from the USA based fact-checkers are also included, which shows that this workflow can be generalized.

Twint[13] is used to crawl Twitter, looking for URLs of the organizations mentioned above, in the comment threads of tweets. This resulted in a coarse collection of *<Tweet, SCR>* pairs. Those pairs with tweets in languages other than English and tweets containing only image/video content are discarded. We perform annotation on this collection, keeping two aspects in mind: (1) the tweet should contain a claim, and (2) it should be textually summarizable to the corresponding SCR. URL removal from SMCs followed by pairwise de-duplication is performed at this stage, resulting in our final dataset, a collection of *<SMC, SCR>* pairs, which can be used for training abstractive text summarization models. *The final dataset only contains <SMC,SCR> pairs where both are in English.*

Key world and Indian events have been covered as part of this dataset, such as the onset of COVID-19 and subsequent immunisation, the Taliban takeover of Afghanistan, Indian General Elections 2019 and US Presidential Elections 2020. Our annotation process is detailed below.

3.1.1 Annotation Details

Two trained annotators were tasked with annotating every *<Tweet,SCR>* pair from the coarse collection (Subsection 3.1). Three categorical attributes viz. Tweet language, SCR language, category and one boolean attribute *viz.* 'Summarizability' had to be populated for each pair.

The annotators were provided instructions to mark a pair as 'summarizable' only when the SCR is a condensed version of the tweet and named entity coverage is more than 50%. For deciding entity coverage, the annotators were allowed to take cues from the mentions and hashtags in the tweet. As majority of the FCA Publishers we dealt with are Indian, a lot of tweets and SCRs were in Indian languages such as Hindi, Hindi transliterated in English, Tamil, Telugu, and some other Indian languages. Any such instances were pruned from our dataset.

[9] Fact Check Explorer: Web Search
[10] ClaimReview Schema
[11] Example FCA
[12] IFCN: Website
[13] Twint: Github Repository

To understand the motivation behind these SMCs, our annotators were also requested to categorize them into classes like a) Politics, b) Crime and Terrorism, c) World, d) Entertainment, e) Technology, f) Food, g) Religion, h) Sports, i) Health, j) Education, k) Business, l) Environment, and m) Other (miscellaneous). Though not relevant to this work, nor a part of the final dataset, we collect and annotate this data as well for further research.

We observe an inter-annotator agreement of 92% within the annotations provided by both.

3.2 Dataset Statistics

Data Entity	**Count**		
<*SMC,SCR*> pairs	567		
Unique SMC	531		
Unique SCR	369		
FCA Source Country			
India	93%		
US	7%		
Median Length	**Chars**	**Words**	
SMC	193	33	
SCR	70	11	
Data Sets	**Cosine Similarity Threshold**		
	0.25	**0.5**	**0.75**
NP	59%	12%	2%
P-H-M	61%	13%	3%
Snopes (Shaar et al., 2020)	50%	8%	1%

Table 1: **Dataset Statistics and Complexity Analysis**. NP and P-H-M are defined in Subsection 4.1.

The statistics of our final dataset, comprising of 567 unique <*SMC, SCR*> pairs are presented in Table 1. Owning to several tweets and several FCAs about the same underlying event, 1:1 correspondence is not observed in the dataset, as evident from the first section in this table. Due to the 280 character limit imposed on tweets by Twitter, the SMCs are not arbitrarily long, with a median length of 33 words and the SCRs are observed to be very short, with a median length of 11 words. Similar to (Shaar et al., 2020), the complexity of the task is analyzed by reporting the word-level TF-IDF weighted cosine similarity for <*SMC, SCR*> pairs. Since our dataset supports summarization, cosine similarity is higher compared to the Snopes dataset by (Shaar et al., 2020), as expected. Figure 2 presents the FCA Source and SMC Topic distribution. 46% of the SMCs are political or religious, which is no surprise as these sensitive topics polarise opinion very easily. A large chunk of SMCs are health-related, owning to misinformation surrounding the COVID-19 immunization and mass hysteria.

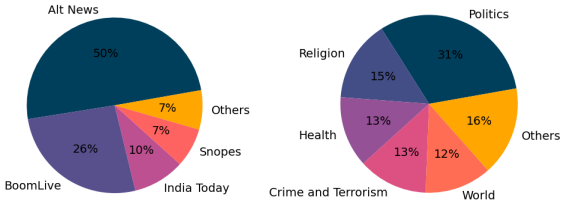

(a) FCA Sources (b) SMC Topics

Figure 2: **Dataset Distribution**

4 Our Approach

SMCs are very noisy in nature due to the inherent way people interact on social media and micro-blogging platforms. On Twitter, tweets are bounded by a character limit, forcing people to use slang and abbreviations to communicate effectively. It also allows for mentions and hashtags to be embedded in tweets to encourage inter-user interaction. Using these noisy SMCs verbatim (as done by Shaar et al. (2020)) to check if they have been previously fact-checked is challenging, as the retrieval module has to do all the heavy lifting for which it is not equipped.

In this work, it is hypothesized that a system which extracts queryable content from SMCs by dealing with its syntactic and semantic aspects before querying the retrieval module should perform better than verbatim querying. Keeping in mind the small scale at which fact-checking organizations work and the continuously growing collection of FCAs, Google's *Fact Check Explorer* is used as a retriever for previously fact-checked claims instead of a closed collection of verified claims. The *Fact Check Explorer* indexes the latest FCAs across the world and provides easy to use search APIs for free, which support filtering based on publisher and language, among other features. Various text preprocessing techniques on SMCs are experimented with before using state-of-the-art abstractive text summarization models to generate corresponding CCRs. These CCRs are then used to query the retriever. These techniques and the models used are detailed in the following subsections. Our prefer-

ence for abstractive summarization over extractive summarization arises because of two reasons; the SMCs are noisy and unlikely to contain query-able spans and due to the recent progress in abstractive summarization research (Lewis et al., 2019; Zhang et al., 2020; Raffel et al., 2020; Xiao et al., 2021).

This proposed workflow is generic in nature as <*SMC, SCR*> pairs collected from other microblogging platforms (using our curation methodology) can be used to train summarization models after applying text pre-processing techniques specific to that platform. These models can also generate queries for open-domain evidence retrieval, which is the next step in a fact-checking pipeline. It is also futuristic in the sense that a generative module can replace the text summarization module to support multimodal SMCs. However, this work is kept limited to textual SMCs due to a lack of suitable labelled data and the absence of a reliable equivalent of *Fact Check Explorer* for joint text, image and video search.

4.1 Twitter Specific Preprocessing

Most social media platforms encourage inter-user interaction by allowing 'mentioning' other users in a post. Typically, some form of notification goes to the user being mentioned, getting his attention on the post content. It is also used as a way for tagging people to establish their presence in photos and videos. Hashtags are metadata tags which allow cross referencing of content by topic or theme. They typically identify with some event or social movement, allowing users to discover and associate with trending content. Both hashtags and mentions are available on Twitter along with emojis, which are smileys embedded in text, providing emotional cues to the reader.

We experiment with these three aspects of a tweet. Most search engines do not deal with Emojis, hence we replace them with a constant to form the P+MRep set. Hashtags and Mentions provide rich signals about named entities, hence it is important to incorporate them in the input in some way. Upon manual analysis of the data, it was seen that a lot of tweets mentioned users who were unrelated to the content in the tweet. Some recurring instances of this phenomena that we came across, were fact-checking requests mentioning many journalists and organizations and political tweets mentioning prominent members of the opposition political party and prominent believers of the opposite ideology. It was observed that hashtags were also used in a similar manner. Another observation was the existence of runs of space separated hashtags and mentions and their occurrence at the beginning or end of the tweet, signifying the preference of users to separate actual tweet content from these meta tags. These signals led us to create sets of data where mention and hashtag runs are removed except the first member in each run (P-MRR-HRR). Further, some users used organization related twitter handles or twitter handles in other languages like Hindi. To deal with this, we replace these by their original names on Twitter to get the P-MRR-HRR+MRep set. With a clear intuition behind such preprocessing, we now describe what Twitter specific text preprocessing techniques are applied to SMCs to produce the following <*SMC,SCR*> sets, from the final dataset:

1. **Verbatim (NP)**: SMCs are used verbatim.
2. **Preprocessed (P)**: Symbols for hashtags(#) and mentions(@), emojis, punctuation and redundant are removed, followed by lowercasing of SMCs.
3. **Pre-processed with Emojis Replaced (P+ERep)**: Emojis are replaced by the string *$EMOJI$* in addition to techniques used in P.
4. **Pre-processed with Hashtags and Mentions Removed (P-H-M)**: All hashtags and mentions are removed in addition to techniques used in P. Subsets with only hashtag removal **(P-H)** and only mention removal **(P-M)** are also created.
5. **Pre-processed with Mention and Hashtag Run Removed (P-MRR-HRR)**: Run of hashtags and mentions are removed, except the first entity in each run, in addition to techniques used in P.
6. **Pre-processed with Mention and Hashtag Run Removed and Mentions Replaced (P-MRR-HRR+MRep)**: The remaining mentioned handles in P-MRR-HRR are replaced by their official names from Twitter.

4.2 Summarization Models

For summarization, the following models were experimented with:

1. **Truncate *k***: A naive summarizer which truncates a SMC to the first k space-separated tokens. It is used as a baseline to show gains by more complex models.

2. **T5**: A transformer-based architecture by Raffel et al. (2020) that uses a text to text approach for all tasks. It is pre-trained on a multi-task mixture of supervised and unsupervised tasks such as denoising on the high quality C4 corpus, sentiment analysis, natural language inference and question answering, among others. To make the model cope with this multi-task training, a task-specific prefix is added to the input sentence.
3. **BART**: A transformer-based sequence to sequence model by Lewis et al. (2019) which incorporates the bidirectional encoder of BERT (Devlin et al., 2019) and the left-to-right autoregressive decoder of GPT (Radford and Narasimhan, 2018; Radford and Wu, 2019), pre-trained in denoising autoencoder style. It works well for downstream tasks involving text generation.
4. **PEGASUS**: A transformer-based sequence to sequence model by Zhang et al. (2020) which uses a self-supervised pre-training objective called gap-sentence generation, aimed at optimizing downstream abstractive summarization tasks. In gap-sentence generation, important sentences in a document are masked, and the transformer model is asked to predict those sentences. PEGASUS shows impressive performance even with a small number of samples during fine-tuning.

4.3 Decoding Strategies

Decoding strategies define how text should be generated by models that support language generation. Based on the end application, the model may be expected to generate text that can be lengthy, short, non-repetitive, interesting, surprising, and so on. Our application requires the output to be short and crisp. In this work, we experiment with Greedy, Beam Search, Top k and Top p (Holtzman et al., 2019) decoding strategies.

4.4 Evaluation Metrics

Recall@k and Mean Reciprocal Rank (MRR) are reported for all experiments, as is the norm in retrieval tasks. While checking if a claim was previously fact-checked, fact-checkers would not want to look beyond the first few results. Keeping this in mind, Recall@*5* is used as the primary metric for comparing retrieval performance. Figure 3 shows the variation in Recall@k with increasing value of k. The sharp bend at Recall@*5*, subsequent plateauing also motivated us to report this metric for retrieval. Also, it is practically feasible for a human fact-checker to go through 5 results per claim rather than 10 or 20 results.

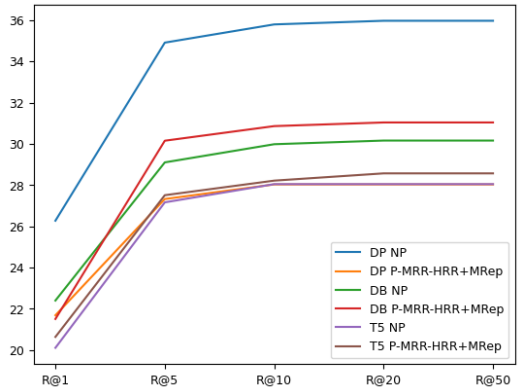

Figure 3: **Recall Plateauing for Decoding Strategies**

For evaluating the quality of the summary generated, word-level TF-IDF weighted cosine similarity between SMCs and CCRs and between SCRs and CCRs is reported. BLEU4 (Papineni et al., 2002).

4.5 Experiment Setup

In the experiments, the performance of summarization models in both out-of-the-box settings and through fine-tuning, *i.e.,* training on the task under supervision are compared. For the fine-tuning experiments, 5-fold cross-validation is performed on the data; and mean values along with the standard deviation are observed. Other experiments are performed on the entire data without any splits as no parameter learning is involved.

All experiments performed with the help of Transformer-based architectures in Table 2 use Beam Search decoder with a beam size of 6 and the maximum token length of a generated sequence, set to 15 with early-stopping enabled. For Truncate k experiments, k=11 is set. We arrive at these constants by looking at the median summary length provided in Table 1 and giving some leeway to transformer models as they operate on sub-word vocabularies. Hugging Face[14] implementations of the models mentioned in Subsection 4.2 are used for all experiments involving transformers. For the PEGASUS and BART experiments, we use the distilled versions released by Shleifer and Rush (2020) using the shrink and fine-tune approach on the CNN dataset. The 16-layer-encoder 4-layer-

[14] Hugging Face: Website

	No Summarization				Summarization using Out of the Box Models						Summarization using Fine Tuned Models					
Preprocessing Strategies	**None**		**Truncate11**		**T5**		**D BART**		**D PEGASUS**		**T5**		**D BART**		**D PEGASUS**	
	R@5	**MRR**	**R@5**	**MRR**	**R@5**	**MRR**	**R@5**	**MRR**	**R@5**	**MRR**	**R@5**	**MRR**	**R@5**	**MRR**	**R@5**	**MRR**
NP	9.52	.09	17.28	.14	15.52	.13	20.46	.17	**22.40**	**.19**	27.16 ±2.55	.23 ±.02	29.10 ±3.15	.26 ±.02	**34.91 ±5.91**	**.30 ±.05**
P	11.99	.11	18.34	.15	17.64	.15	17.99	.14	21.52	.17	28.38 ±6.55	.24 ±.05	28.21 ±6.88	.24 ±.06	27.69 ±2.63	.24 ±.02
-H	12.70	.12	17.99	.15	17.28	.15	17.64	.14	20.46	.16	24.87 ±5.13	.21 ±.04	30.15 ±6.08	.26 ±.06	29.61 ±6.59	.25 ±.05
-M	13.05	.12	18.69	.15	18.34	.15	17.64	.14	21.16	.17	26.80 ±5.33	.23 ±.03	*30.15 ±4.98*	.26 ±.04	29.10 ±2.45	.25 ±.02
-H-M	13.93	.13	**18.87**	**.15**	17.81	.15	17.28	.14	20.28	.16	26.46 ±4.69	.23 ±.03	27.51 ±4.09	.23 ±.04	26.28 ±2.36	.23 ±.02
+ERep	10.58	.10	17.46	.14	16.58	.14	17.99	.15	22.05	.18	27.51 ±6.13	.23 ±.05	28.20 ±5.89	.25 ±.05	30.15 ±5.72	*.27 ±.05*
-MRR-HRR	12.35	.11	17.81	.15	17.46	.15	17.81	.14	21.69	.18	28.75 ±6.38	.25 ±.05	27.33 ±5.94	.23 ±.05	25.92 ±1.98	.22 ±.01
-MRR-HRR +MRep	12.70	.12	**18.87**	**.15**	18.34	.15	17.81	.14	21.87	.18	27.51 ±5.43	.24 ±.04	30.15 ±5.98	.25 ±.05	27.32 ±3.87	.24 ±.03
Skyline	**63.85**	**.55**														

Table 2: **Retrieval Results** (Subsection 5.1). D BART and D PEGASUS stand for Distilled BART and Distilled PEGASUS respectively, and Recall@*5* is represented by R@5. All preprocessing strategies prefixed with '+' or '-' are applied on top of the P set.

decoder version of distilled PEGASUS[15] and 12-layer-encoder 6-layer-decoder version of distilled BART[16] are used. The base version of T5[17] is used for all experiments involving T5. The number of trainable parameters are 220M, 300M and 370M in T5-base, Distilled BART and Distilled PEGASUS, respectively. Since *Fact Check Explorer* is an ever-growing and evolving system, CCRs are generated for all experiments first, and then retrieval queries are run, ensuring consistency across results. For retrieval, the API documentation[18] is followed and retrieved URLs are compared with normalized (removing redirection/parameters accompanying the URL) URLs associated with an FCA.

5 Results and Discussion

This section discusses the results obtained at various stages of our workflow.

5.1 Retrieval

We present the retrieval results in Table 2. From top to bottom, the SMC pre-processing (Section 3) complexity increases and from left to right, the complexity of the summarization models (Subsection 4.2) increases. For the skyline numbers, *Fact Check Explorer* is queried using the gold SCRs, giving 63.85 Recall@*5* and 0.55 MRR. We observe two evident trends via experimentation- (1) the performance gain in using an out-of-the-box summarization model, as compared to no summarization, and (2) the benefit of learning under supervision on our labelled dataset, indicated by the sharp gain in performance of fine-tuned models as compared to the corresponding out of the box models. Since the PEGASUS model is pre-trained with an objective to boost abstractive summarization performance, it works quite well out-of-the-box, giving a 2x increase in performance compared to no summarization. The best performing model is Distilled PEGASUS, fine-tuned on our dataset (without any pre-processing), as exhibited by a Recall@*5* of 34.91 and MRR of 0.3, which is more than 3x improvement over verbatim querying.

We use three different summarization strategies- (1) No Summarization, (2) Summarization using out-of-the-box Models, and (3) Summarization using fine-tuned models as shown in Table 2. We separately highlight the best performance in the table itself in each of these cases. In the no summarization experiments, we observe that complex pre-processing techniques lead to a performance gain, as indicated by the best Recall@*5* and MRR of 18.87 and 0.15 on dealing with mentions and hashtags (for both P-H-M and P-MRR-HRR+MRep). Among the out-of-the-box experiments, it is seen that Distilled PEGASUS comfortably outperforms T5 and Distilled BART, with the best Recall@*5* and MRR being 22.4 and 0.19, respectively. Highly parameterized models like BART and PEGASUS do not benefit from input pre-processing.

The gap between the skyline numbers and the best performing model can be attributed to the fact that most models are pre-trained on document level summarization datasets such as CNN/Daily Mail (Nallapati et al., 2016) and Huge News (Zhang et al., 2020). Hence, they struggle with summarizing short input text.

5.2 Summarization Quality

The quality of CCRs is reported in Table 3. We compare CCRs with SMCs and SCRs on two metrics (1) Word level TF-IDF Weighted Cosine Simi-

[15] Distilled PEGASUS Model
[16] Distilled BART Model
[17] T5-base Model
[18] Fact Check Explorer: API Documentation

Experiment	n-Gram	Cosine Similarity Threshold			BLEU4
		0.25	0.5	0.75	
E1 SMC vs CCR	1	80%	26%	2%	-
	2	83%	30%	2%	-
E1 SCR vs CCR	1	76%	38%	14%	39.7
	2	73%	38%	13%	39.3
E2 SMC vs CCR	1	83%	26%	4%	-
	2	85%	29%	3%	-
E2 SCR vs CCR	1	68%	31%	11%	38.9
	2	69%	33%	13%	39.2

Table 3: **Summarization Quality Analysis** (Subsection 5.2). E1 and E2 stand for Distilled PEGASUS with NP and P-MRR-HRR+MRep experiments respectively (as described in Section 4) and n-Gram stands for the value of n in n-grams not appearing more than once in beam search decoding.

larity and (2) BLEU4. Since BLEU4 is generally reported between reference and generated sequences, it does not make sense to report it for SMC vs CCR rows. *We observe high BLEU4 scores for the CCRs* signifying that our approach can generate valid summaries, as can also be seen in Table 4. For both E1 and E2, our BLEU4 scores are approaching (approx.) 40. On comparing SMC vs CCR cosine similarities for both experiments with the cosine similarity of SMC vs SCR (last section in Table 1), we find higher values for all thresholds indicating that *our generated summaries are significantly similar to the tweets as compared to the gold summaries provided.*

5.3 Decoding

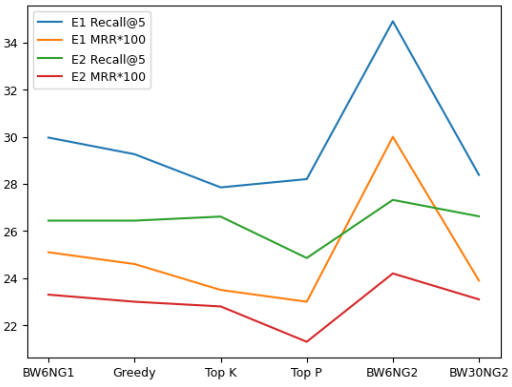

Figure 4: **Decoding Strategy Comparison** (Subsection 5.3). MRR values are multiplied by 100 for better visualization. BWkNGn corresponds to beam search decoding with beam width k and no n-grams appearing more than once in the generated output. k is set to 50 in Top k and p is set to 0.92 in Top p.

Figure 4 shows the variation in retrieval results on using different decoding strategies. Definitions for E1 and E2 follow from Table 3.

As observed from this figure, BW*6*NG*2* seems to be the best performing decoding strategy. Hence, this strategy is used for all experiments in Table 2. BW*6*NG*1* also seems to be a good alternative, but the *1*-gram constraint makes the queries very terse and grammatically inconsistent (observed manually). Greedy, Top k and Top p strategies are not competitive for such a task.

5.4 Larger Language Models

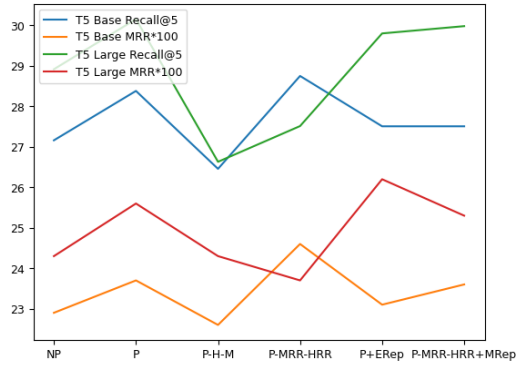

Figure 5: **Effect of Larger Language Models on Retrieval Metrics**

We study the variation in performance using even larger models such as T5 Large[19], which has 770M parameters, three times that of T5-base. CCRs generated by the larger model perform better on both retrieval metrics across a variety of preprocessed SMCs, but the performance gain is not significant. It is offset by a longer training time and heavy compute requirements leading to considerable cost overheads. Since this is not a study of large generative models and given the modest resources owned by most fact-checking organizations, we do not explore any larger language models such as T5-3B and T5-11B, which have 3 Billion and 11 Billion parameters, respectively.

5.5 CCR Quality

Table 4 lists a few CCRs generated by the best performing model, also listing the corresponding SMCs and SCRs. The model successfully extracts the core claim from the SMCs and ignores tokens like mentions and hashtags that have no contribution to the core claim. Owning to the constraints placed on length, it is seen that the generated CCRs are succinct and context-independent. They seem to be paraphrases of the gold SCRs, making them

[19] T5-large Model

#	SMC	SCR	CCR
1	Congratulations to Uttarakhand CM for becoming the first CM ever to charge stranded citizens for rescue operations! Helicopter rides will now be chargeable during rescue operations in Uttarakhand. And if you can't pay, you may safely die. #AchheDin #BJP	Passengers in Uttarkhand to be charged for rescue operations	Uttarakhand CM has charged stranded citizens for helicopter rides during rescue operations
2	@AltNews We are getting various WhatsApp forward regarding as Corona has been emerges only due to 5G testing in world. Please put some light, seems ,it is only a brain shit.	5G radiation is the cause behind the second wave of coronavirus pandemic in India	Coronavirus outbreak due to 5G testing
3	This woman in Afghanistan was killed by Taliban for not wearing the proper cloth. #Afghanistan #Taliban @cnn @FoxNews @BBCWorld	Video shows a woman being shot in the head by Taliban in Afghanistan for not dressing appropriately	Woman killed by Taliban for not wearing proper cloth
4	"India is ranked 102nd in the global hunger index, out of 117 countries. We are ranked in between Niger & Sierra Leone. We are the lowest ranked South Asian country. Bangladesh is ranked 88th and Pakistan 94th. They have only recently overtaken us. Our rank was 55,only 5 years ago"	India's ranking in Global Hunger Index (GHI) has fallen from 55 in 2014 to 102 in 2019	India ranked 102nd in the global hunger index
5	"Oxygen donated from Saudi and relabelled in india by Reliance, Share this with your contacts in Saudi and make this viral .. Let the world know the cheapness of this PM "	Oxygen sent from Saudi Arabia is being distributed in the name of Reliance	Reliance taking credit for oxygen supplied by Saudi Arabia

Table 4: **SMCs and SCRs from the Dataset with corresponding CCRs** (Subsection 5.5).

good candidates for querying the retrieval system. It is also seen that our model finds factual inputs which require reasoning, difficult to deal with. For instance, #4 in Table 4 requires a model to understand that going from rank 55 to 102 in the Global Hunger Index is a fall and not a rise. Our workflow does not expect the underlying language model to understand and reason, and this workflow only requires the generation of a valid summary.

6 Conclusion and Future Work

In this work, a new workflow for detecting previously fact-checked claims is proposed. This workflow uses text summarization as an intermediate step before retrieval module invocation. Clean and crisp summaries thus generated are then used for querying a retrieval system. To this end, a first-of-its-kind tweet summarization dataset in the Indian context to train such models is curated and released under the CC-BY-NC-SA 4.0 license. The performance gained on using popular out-of-the-box and fine-tuned summarization models before querying the *Fact Check Explorer* is demonstrated, and discussed with qualitative samples. Various popular decoding strategies are compared, and the implication of using larger pre-trained models is explored. *The aim of this work is to aid in the creation of general-purpose and performant modules which can speed up a fact-checking pipeline by equipping fact-checkers with the tools to fight misinformation at a large scale.*

In future, we would also like to perform this task in a more general context for news items from various countries, extending our work in a multilingual scenario. Also, named entities are crucial in drafting a good query for any retrieval system. Generating summaries based on the Named Entities (Zhang et al., 2020) found in SMCs is a promising avenue to explore. We do not take tweet threads into account as our focus is SMCs by users and not replies or comments to those SMCs, however, this can be an interesting future direction. Other controlled text generation (Keskar et al., 2019; Chan et al., 2021) techniques can also be explored to extract the maximum information from noisy SMCs. Better pre-training objectives for abstractive summarization on noisy text can lead to efficient out-of-the-box models for this task.

Most Indian fact-checking organizations in Section 3.1.1 also publish FCAs in regional languages such as Hindi, Tamil and Telugu. Twitter conversations, spreading misinformation in other pure and transliterated Indic languages are voluminous. Cross-lingual summarization research (Zhu et al., 2019) would go a long way in fighting misinformation in a holistic manner.

7 Acknowledgements

We acknowledge the kind support provided by IMPRINT-2, a technology development initiative of MHRD and DST, Government of India.

8 Ethical Considerations

To the best of our knowledge, no code of ethics was violated throughout the experiments performed for this study. We report all hyper-parameters and other technical details necessary to reproduce our results, and release the code and dataset curated via this work. We perform our experiments with the help of various language models which may contain biases as discussed by Weidinger et al. (2021). However, we believe that our workflow and methodology are solid and apply to any social media fake news setting. Any quantitative results reported by us are reproducible, subject to the ever growing number of articles indexed by the Fact Check Explorer (reported in Section 4.5). However, the qualitative results (like generated summaries) are an outcome of computational models that does not represent our personal views. We do not include any identifying information in the data that we use for our experiments and ensure that the dataset release will follow anonymization of any such information.

We would like to state that this dataset is collected in a recent real-world setting (raw social media claims from 2018-2022) and *no attempt has been made by us to subdue tweets on certain topics and promote others*. More precisely, we freely assigned the tweets to our annotators without any domain/topic specificity, however, they were required to label the tweet from a list of categories (Section 3.1.1) to collect more information.

9 Limitations

We believe there is a limitation to our work, *i.e.,* **The limited size of this dataset**; which can be attributed to following reasons:

- Most fact-checking organisations (covered in this work) emerged post-2017.
- Our data curation relies on a large number of users replying to potentially misinformative tweets. This user behaviour is limited by social network usage, awareness and internet proliferation for a particular language, region or country.
- The "manual pruning" step while curating the tweet level summarization dataset was a very time/effort-intensive process. For e.g., around 5000 coarse <*Tweet,SCR*> pairs were manually pruned to get the final dataset containing 567 <*SMC,SCR*> pairs, implying a rejection rate close to 90%.

References

Hunt Allcott and Matthew Gentzkow. 2017. Social media and fake news in the 2016 election. Working Paper 23089, National Bureau of Economic Research.

Rami Aly, Zhijiang Guo, Michael Sejr Schlichtkrull, James Thorne, Andreas Vlachos, Christos Christodoulopoulos, Oana Cocarascu, and Arpit Mittal. 2021. The fact extraction and VERification over unstructured and structured information (FEVEROUS) shared task. In *Proceedings of the Fourth Workshop on Fact Extraction and VERification (FEVER)*, pages 1–13, Dominican Republic. Association for Computational Linguistics.

Alberto Barrón-Cedeno, Tamer Elsayed, Preslav Nakov, Giovanni Da San Martino, Maram Hasanain, Reem Suwaileh, and Fatima Haouari. 2020. Checkthat! at clef 2020: Enabling the automatic identification and verification of claims in social media. In *European Conference on Information Retrieval*, pages 499–507. Springer.

Alvin Chan, Yew-Soon Ong, Bill Pung, Aston Zhang, and Jie Fu. 2021. Cocon: A self-supervised approach for controlled text generation. In *International Conference on Learning Representations*.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, Minneapolis, Minnesota. Association for Computational Linguistics.

Angela Fan, Aleksandra Piktus, Fabio Petroni, Guillaume Wenzek, Marzieh Saeidi, Andreas Vlachos, Antoine Bordes, and Sebastian Riedel. 2020. Generating fact checking briefs. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 7147–7161, Online. Association for Computational Linguistics.

Ari Holtzman, Jan Buys, Li Du, Maxwell Forbes, and Yejin Choi. 2019. The curious case of neural text degeneration. *arXiv preprint arXiv:1904.09751*.

Nitish Shirish Keskar, Bryan McCann, Lav R Varshney, Caiming Xiong, and Richard Socher. 2019. Ctrl: A conditional transformer language model for controllable generation. *arXiv preprint arXiv:1909.05858*.

Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Ves Stoyanov, and Luke Zettlemoyer. 2019. Bart: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. *arXiv preprint arXiv:1910.13461*.

Watheq Mansour, Tamer Elsayed, and Abdulaziz Al-Ali. 2022. Did i see it before? detecting previously-checked claims over twitter. In *European Conference on Information Retrieval*, pages 367–381. Springer.

Rahul Mishra, Dhruv Gupta, and Markus Leippold. 2020. Generating fact checking summaries for web claims. In *EMNLP W-NUT 2020: Conference on Empirical Methods in Natural Language Processing (EMNLP)*.

Preslav Nakov, Alberto Barrón-Cedeño, Giovanni Da San Martino, Firoj Alam, Julia Maria Struß, Thomas Mandl, Rubén Míguez, Tommaso Caselli, Mucahid Kutlu, Wajdi Zaghouani, et al. 2022. The clef-2022 checkthat! lab on fighting the covid-19 infodemic and fake news detection. In *European Conference on Information Retrieval*, pages 416–428. Springer.

Preslav Nakov, David Corney, Maram Hasanain, Firoj Alam, Tamer Elsayed, Alberto Barr'on-Cedeno, Paolo Papotti, Shaden Shaar, and Giovanni Da San Martino. 2021a. Automated fact-checking for assisting human fact-checkers. In *IJCAI*.

Preslav Nakov, Giovanni Da San Martino, Tamer Elsayed, Alberto Barrón-Cedeno, Rubén Míguez, Shaden Shaar, Firoj Alam, Fatima Haouari, Maram Hasanain, Nikolay Babulkov, et al. 2021b. The clef-2021 checkthat! lab on detecting check-worthy claims, previously fact-checked claims, and fake news. In *European Conference on Information Retrieval*, pages 639–649. Springer.

Ramesh Nallapati, Bowen Zhou, Cicero dos Santos, Çağlar Gu̇lçehre, and Bing Xiang. 2016. Abstractive text summarization using sequence-to-sequence RNNs and beyond. In *Proceedings of The 20th SIGNLL Conference on Computational Natural Language Learning*, pages 280–290, Berlin, Germany. Association for Computational Linguistics.

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. Bleu: a method for automatic evaluation of machine translation. In *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics*, pages 311–318, Philadelphia, Pennsylvania, USA. Association for Computational Linguistics.

Alec Radford and Karthik Narasimhan. 2018. Improving language understanding by generative pre-training. In *Published by OpenAI*.

Alec Radford and Jeffrey Wu. 2019. Rewon child, david luan, dario amodei, and ilya sutskever. 2019. *Language models are unsupervised multitask learners. OpenAI Blog*, 1(8):9.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J. Liu. 2020. Exploring the limits of transfer learning with a unified text-to-text transformer. *Journal of Machine Learning Research*, 21(140):1–67.

Aalok Sathe, Salar Ather, Tuan Manh Le, Nathan Perry, and Joonsuk Park. 2020. Automated fact-checking of claims from Wikipedia. In *Proceedings of the 12th Language Resources and Evaluation Conference*, pages 6874–6882, Marseille, France. European Language Resources Association.

Tal Schuster, Adam Fisch, and Regina Barzilay. 2021. Get your vitamin C! robust fact verification with contrastive evidence. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 624–643, Online. Association for Computational Linguistics.

Shaden Shaar, Firoj Alam, Giovanni Da San Martino, and Preslav Nakov. 2021a. Assisting the human fact-checkers: Detecting all previously fact-checked claims in a document. *arXiv preprint arXiv:2109.07410*.

Shaden Shaar, Firoj Alam, Giovanni Da San Martino, and Preslav Nakov. 2021b. The role of context in detecting previously fact-checked claims. *arXiv preprint arXiv:2104.07423*.

Shaden Shaar, Nikolay Babulkov, Giovanni Da San Martino, and Preslav Nakov. 2020. That is a known lie: Detecting previously fact-checked claims. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 3607–3618, Online. Association for Computational Linguistics.

Shaden Shaar, Fatima Haouari, Watheq Mansour, Maram Hasanain, Nikolay Babulkov, Firoj Alam, Giovanni Da San Martino, Tamer Elsayed, and Preslav Nakov. 2021c. Overview of the CLEF-2021 CheckThat! lab task 2 on detecting previously fact-checked claims in tweets and political debates. In *Working Notes of CLEF 2021—Conference and Labs of the Evaluation Forum*, CLEF '2021, Bucharest, Romania (online).

Gautam Kishore Shahi, Anne Dirkson, and Tim A. Majchrzak. 2021. An exploratory study of covid-19 misinformation on twitter. *Online Social Networks and Media*, 22:100104.

Karishma Sharma, Feng Qian, He Jiang, Natali Ruchansky, Ming Zhang, and Yan Liu. 2019. Combating fake news: A survey on identification and mitigation techniques. *ACM Transactions on Intelligent Systems and Technology (TIST)*, 10(3):1–42.

Sam Shleifer and Alexander M Rush. 2020. Pre-trained summarization distillation. *arXiv preprint arXiv:2010.13002*.

Dominik Stammbach and Elliott Ash. 2020. e-fever: Explanations and summaries for automated fact checking. *Proceedings of the 2020 Truth and Trust Online (TTO 2020)*, pages 32–43.

Andon Tchechmedjiev, Pavlos Fafalios, Katarina Boland, Malo Gasquet, Matthäus Zloch, Benjamin Zapilko, Stefan Dietze, and Konstantin Todorov. 2019. Claimskg: a knowledge graph of fact-checked claims. In *International Semantic Web Conference*, pages 309–324. Springer.

James Thorne, Andreas Vlachos, Christos Christodoulopoulos, and Arpit Mittal. 2018. FEVER: a large-scale dataset for fact extraction and VERification. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 809–819, New Orleans, Louisiana. Association for Computational Linguistics.

Nguyen Vo and Kyumin Lee. 2020. Where are the facts? searching for fact-checked information to alleviate the spread of fake news. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 7717–7731, Online. Association for Computational Linguistics.

Laura Weidinger, John Mellor, Maribeth Rauh, Conor Griffin, Jonathan Uesato, Po-Sen Huang, Myra Cheng, Mia Glaese, Borja Balle, Atoosa Kasirzadeh, Zac Kenton, Sasha Brown, Will Hawkins, Tom Stepleton, Courtney Biles, Abeba Birhane, Julia Haas, Laura Rimell, Lisa Anne Hendricks, William S. Isaac, Sean Legassick, Geoffrey Irving, and Iason Gabriel. 2021. Ethical and social risks of harm from language models. *CoRR*, abs/2112.04359.

Dongling Xiao, Han Zhang, Yukun Li, Yu Sun, Hao Tian, Hua Wu, and Haifeng Wang. 2021. Ernie-gen: An enhanced multi-flow pre-training and fine-tuning framework for natural language generation. In *Proceedings of the Twenty-Ninth International Joint Conference on Artificial Intelligence*, IJCAI'20.

Jingqing Zhang, Yao Zhao, Mohammad Saleh, and Peter Liu. 2020. Pegasus: Pre-training with extracted gap-sentences for abstractive summarization. In *International Conference on Machine Learning*, pages 11328–11339. PMLR.

Junnan Zhu, Qian Wang, Yining Wang, Yu Zhou, Jiajun Zhang, Shaonan Wang, and Chengqing Zong. 2019. NCLS: Neural cross-lingual summarization. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 3054–3064, Hong Kong, China. Association for Computational Linguistics.

Learning to Generate Explanation from e-Hospital Services for Medical Suggestion

Wei-Lin Chen,[1] **An-Zi Yen,**[2] **Hen-Hsen Huang,**[3] **Hsin-Hsi Chen**[1]
[1]Department of Computer Science and Information Engineering, National Taiwan University, Taiwan
[2]Department of Computer Science, National Yang Ming Chiao Tung University, Taiwan
[3]Institute of Information Science, Academia Sinica, Taiwan
wlchen@nlg.csie.ntu.edu.tw, azyen@nycu.edu.tw,
hhhuang@iis.sinica.edu.tw, hhchen@ntu.edu.tw

Abstract

Explaining the reasoning of neural models has attracted attention in recent years. Providing highly-accessible and comprehensible explanations in natural language is useful for humans to understand the model's prediction results. In this work, we present a pilot study to investigate explanation generation with a narrative and causal structure for the scenario of health consulting. Our model generates a medical suggestion regarding the patient's concern and provides an explanation as the outline of the reasoning. To align the generated explanation with the suggestion, we propose a novel discourse-aware mechanism with multi-task learning. Experimental results show that our model achieves promising performances in both quantitative and human evaluation.

1 Introduction

Neural models have shown remarkable success in various tasks, however, simply offering the predictions may not satisfy the requirement of end-users. Understanding how the decision has been reached by the model is essential in real-world applications. To provide a meaningful, human-comprehensible explanation, presenting it in natural language is a proper fashion. Note that simply present the explanation as a shopping list or fragments of text-highlight is not an ideal way. Humans prefer to read a text composed of a narrative structure, organized by the discourse relations elaborating the causality between the model input and output (Reiter, 2019). In this work, we propose a novel health consultancy model which can provide medical suggestions accompanied with natural language explanations learned from medical specialists to help humans make decisions in their daily life.

As many people are eager for help in addressing their health concerns, a model is necessitated to be capable of not only providing suggestions regarding their concerns, but also explaining the suggestions, alleviating their worries before visiting

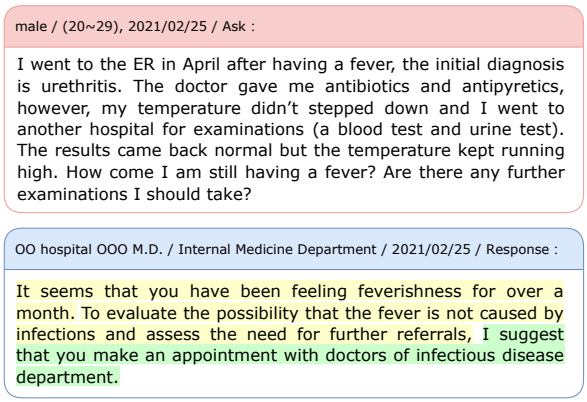

Figure 1: A Data Instance from the Health Consultancy Website

the doctors. Taking Figure 1 as an example, given a question asked by a patient, the physician answers it by explaining what disease might cause the symptoms mentioned in the question and suggests which medical specialty the patient should seek. In this example, the response has a clear narrative structure, where the explanation and suggestion are denoted by the yellow and green highlights, respectively. First, the patient's concern is addressed (e.g., having a fever for over a month). Then, before giving the suggestion, the physician explains the causality between the concerns and the suggestion. Recently, Explainable Artificial Intelligence (XAI), which aims at explaining how the decision is reached by the machine learning model (Ribeiro et al., 2016; Mullenbach et al., 2018; Pezeshkpour et al., 2019), has been gaining attentions in the research community, including works that provide textual explanations (Wu and Mooney, 2019; Rajani et al., 2019; Brahman et al., 2021). Generally, generating textual explanation can be regarded as a natural language generation task. Typically, most works collect explanations by asking annotators to write free-text sentences. Since human annotation is expensive, labor-intensive, and time-consuming, especially in domains where expertise is needed,

Proceedings of the 29th International Conference on Computational Linguistics, pages 2946–2951
October 12–17, 2022.

an alternative way is to construct synthesized explanations by designing rules to exploit information from other datasets as explanations (Li et al., 2018). Although previous works demonstrate that introducing an auxiliary generation task to explain the prediction enables performance improvement, two main issues remain to be tackled: **(1)** The explanations are annotation artifacts (Gururangan et al., 2018) since the annotators typically write vanilla and trivial description (Lei et al., 2020), resulting in a lack of linguistic variety (Parikh et al., 2020) that leads the model prone to overfit on annotator characteristics (Geva et al., 2019). **(2)** Whether the model faithfully explains the suggestion is still an open question (Jacovi and Goldberg, 2020). Producing an explanation just mimicking the way humans would say is impractical in fields involving high-stake scenarios.

To address the aforementioned two issues, our work is based on real-world data. We collect 86,399 question answering (QA) pairs [1] from an online health consultancy website called Taiwan e-Hospital,[2] which allows users to ask questions regarding their health conditions and physicians will respond to their concerns. The linguistic diversity from multiple users is greater than free-text produced by a few crowd-workers, and the explanation within an answer is more natural than handcrafted annotation. Then we propose pilot models to generate response consisting of explanation and suggestion according to the question. Note that our approach can be easily generalized since it is language and domain independent. The contributions of our work are summarized as follows:

1. We show a pilot study on health consulting with professional explanations.
2. We propose a novel discourse-aware mechanism that aligns the generated explanation with the suggestion.
3. Both qualitative and human evaluation show that our discourse-aware model achieves promising performances on suggestion and explanation generation.

2 Dataset

As mentioned in Section 1, we construct our dataset by crawling approximately 86k QA pairs from Taiwan e-Hospital website, where an answer is a free-text response written by the physician, containing the explanation and suggestion as Figure 1 shows. An ideal instance would be a triple of (q, s, e), where q is the question asked by the patient, and s and e are the suggestion and the explanation responded by the physician. However, the crawled raw text often carries greeting terms, salutations, and personal information, such as names of the patients and doctors, which are noise for our task and should be pruned. To gather the desired (q, s, e), we propose a rule-based keyword matching method to extract text snippets that belong to the suggestion and explanation, defined as follows.

- **Suggestion:** The suggested action regarding the patient's concerns, such as whether to seek medical attention, the department for making an appointment with, or the follow-up examination to undergo.
- **Explanation:** The text describing why a physician gives the suggestion. Generally, it includes medical knowledge to address the patient's concerns.

The details of our method are shown as follows.

Step1: We define a set of keyphrases $\mathcal{G}$ that belongs to greeting terms or salutations by regular expressions. Given a sequence of sentences $X = (x_1, x_2, ..., x_n)$ in the response, the i-th sentence x_i that contains a word $w \in \mathcal{G}$ will be filtered. Afterwards, a sequence of sentences $X^{'}$ is obtained, where $1 \leq |X^{'}| \leq n$.

Step2: We find that the sentences belonging to suggestions usually contain certain keywords such as "suggest", "recommend", and the name of the department to "seek". Hence, to identify whether a sentence $x \in X^{'}$ belongs to a suggestion, we manually collect a set of keywords $\mathcal{K}$ from several responses. Then, the sentences in $X^{'}$ that contain a word $w \in \mathcal{K}$ are regarded as the suggestion, and the remaining sentences are considered as the explanation.

Step3: In addition to preparing the (q, s, e) triples, we also construct binary labels $(0/1)$ from s, denoting whether the patient should receive medical assistance based on his/her health condition. Taking Figure 1 as an example, it would be labeled as 1 since the doctor suggests the user to make an

[1] The dataset for this paper is available at https://github.com/ntunlplab/tw-eH

[2] https://sp1.hso.mohw.gov.tw/doctor/

Setting	Input	Output
R_1	[user's question]	[doctor's response]
R_2	Suggest: [user's question] Explain: [user's question]	[doctor's suggestion] [doctor's explanation]
R_3	Suggest: [user's question] Explain: [user's question]	[doctor's suggestion] [doctor's explanation] [binary label for medical assistance]

Table 1: Three Input-Output Settings in the Experiments

Step1	Step2	Step3
0.82	0.80	0.81

Table 2: Evaluation Results of Regular Expressions in the Three Steps

appointment. In our expectation, this label, which indicates how serious the health risk the patient is facing, can play a useful auxiliary task. We compose a set of patterns to identify whether the physician suggests the patient to seek medical attention (e.g., mentioning a medical department in the suggestion).

The processed result from each step, namely R_1, R_2, and R_3, is utilized as references for our different proposed methods described in Section 3, i.e., *mT5* (3.1), *MTL mT5* (3.2), and *DMTL mT5* (3.3), respectively. The details of the dataset formats are shown in Table 1.

To validate the results of our regular expressions in Steps 1, 2, and 3, we randomly sample 100 instances from R_1, R_2, and R_3, respectively. And by checking the correctness of these instances with human evaluation described as follows, we can assess the quality of the regular expressions. For Steps 1 and 2, an instance is considered incorrect if it contains sentences that should be filtered or missing sentences that should be retained. That is, the regular expression admits or filters the wrong sentences. Otherwise, it is considered as correct. For Step 3, the correctness of an instance is determined by checking if the binary label l is the same as whether the physician suggests the patient to seek medical attention or not. The results are measured by $\frac{\#\text{ correct instances}}{100}$ and presented in Table 2. And Table 3 shows the statistics of the top-10 departments ranked by the number of QA pairs, where Exp. and Sug. indicate Explanation and Suggestion, respectively.

		Avg # sent.	
Department	**# QA pairs**	**Exp.**	**Sug.**
Gynecology & Obstetrics	11,676	7.28	1.45
Gastroenterology	7,497	6.36	1.56
Dermatology	7,487	5.93	1.38
Urology	6,895	8.55	1.42
Orthopedics	6,870	6.60	1.47
General Surgery	5,966	7.10	1.51
Ophthalmology	5,086	9.86	1.44
Otorhinolaryngology	5,010	7.54	1.60
Psychiatry	4,489	13.78	1.72
Dentistry	3,998	6.84	1.38

Table 3: Statistics of the Top-10 Departments with Average Numbers of Explanation (Exp.) and Suggestion (Sug.) Sentences

3 Methodology

In this section, we introduce our models for the task of suggestion and explanation generation. As the references of suggestions and explanations are collected by handcrafted rules without human annotation, the models are weakly supervised. Given a question q, our goal is to learn a generator $\mathbf{gen}(\cdot)$ to generate a textual response with a narrative structure consisting of a suggestion s and an explanation e. Three $\mathbf{gen}(\cdot)$ models are described as follows.

3.1 The mT5 Model

We adopt the pre-trained multilingual T5-base model (Xue et al., 2021), which casts natural language processing problems in an unified "text-to-text" form with great flexibility. The input and output (label) data are the patient's question and the corresponding response R_1, i.e., the result of Step 1 in Section 2. Since R_1 does not explicitly extract s and e from the given responses, the generated results of *mT5* do not distinguish the suggestions and the explanations. The pre-trained mT5-base model

Method	Full Response			Suggestion			Explanation		
	R-1	R-2	R-L	R-1	R-2	R-L	R-1	R-2	R-L
mT5	20.335	6.844	17.135	–	–	–	–	–	–
MTL mT5	21.470	7.429	19.383	22.559	7.615	21.764	20.691	7.296	17.679
DMTL mT5	**22.176**	**7.619**	**20.096**	**22.717**	**7.840**	**21.893**	**21.789**	**7.461**	**18.811**

Table 4: Results of Suggestion and Explanation Generation, Reported in ROUGE-1 (R-1), ROUGE-2 (R-2), and ROUGE-L (R-L)

also serves as the backbone for the following two $\mathbf{gen}(\cdot)$s.

3.2 The mT5 Model with Multitask Learning

Since *mT5* does not always generate suggestions and explanations as expected, i.e., the generated response would sometimes contain only suggestion or only explanation, we implement a multi-task learning mT5-base model, *MTL mT5*, to address this issue. To train the *MTL mT5*, we use R_2 as the output data. For input data, we add a prefix text, "Suggest:" or "Explain:", to specify which task the model should perform. Concretely, given q and a response (s, e) in R_2, the two formats of (input, output) data are ("Suggest: q", s) and ("Explain: q", e). In this way, the *MTL mT5* model can generate suggestions and explanations explicitly.

3.3 Discourse-aware MTL mT5 (DMTL mT5)

With the multi-task setting, given an input question q, both the suggestion s and the corresponding explanation e are generated. Ideally, the user can assess the need to seek medical attention based on the model's generated suggestion accompanied with the explanation. However, if the explanation cannot support the suggestion, i.e., they are not related, the user would be confused, so that s/he may decrease the degree of confidence to the system.

To mitigate this problem, we use the dataset R_3, which contains the binary labels indicating whether the patient needs to receive medical assistance. We propose a discourse-aware mechanism into the *MTL mT5* model by introducing a new objective function with a weighted parameter to focus on generating the correct binary label l, where $l \in \{0, 1\}$. We view l as a concise summary of the suggestion, and assume that the model would generate explanations aligned with suggestions in order to predict this binary label l. Specifically, considering a sequence of n words with a binary label $Y = (y_1, y_2, ..., y_n, l)$ output by *DMTL mT5* and the reference sequence $\hat{Y} = (\hat{y}_1, ..., \hat{y}_n, \hat{l})$, the weighted cross-entropy loss ψ of the task of explanation generation is computed as follows:

$$\psi = \left(\sum_{i=1}^{n} \mathcal{L}(y_i, \hat{y}_i) \right) + \alpha \times \left(\mathcal{L}(l, \hat{l}) \right)$$

where $\mathcal{L}$ denotes the cross entropy loss, and α is a hyper-parameter for the weighted loss function. We set $\alpha = 1.1$ by tuning with the validation set.

Note that the main purpose of the binary label l is to provide loss signals encouraging the generated explanations to align with the suggestions. For inference, l is not exposed to end-users, that is, we conduct post-processing to trimmed l from the generated explanation.

4 Experiments and Discussions

We conduct both quantitative and qualitative evaluations to compare the generated suggestions and explanations of our proposed methods. The dataset are randomly split into train, validation, and test sets by the ratio 8:1:1 (69,119, 8,640, 8,640), where every instance is a QA pair. And we adopt teacher-forcing strategy (Williams and Zipser, 1989) with the cross-entropy loss as the objective function for optimizing all models, i.e., *mT5*, *MTL mT5* and *DMTL mT5*. The results reported in this section are conducted on the test set.

4.1 Quantitative Evaluation

The ROUGE-1, ROUGE-2, and ROUGE-L scores (Lin, 2004) between the generated responses and the reference responses are shown in Table 4, denoted as R-1, R-2, R-L, respectively. Since *mT5* does not generate s and e individually, we combine the generated s and e from the multi-task learning models, i.e., *MTL mT5* and *Discourse-aware MTL mT5*, as one full response to compare across three models. As shown in Table 4, the multi-task learning models outperform the *mT5* model. It confirms

Method	Relevan.	Suggest.	Explan.
mT5	3.64	3.09	2.00
MTL mT5	3.75	3.52	2.21
DMTL mT5	**3.87**	**3.56**	**2.42**

Table 5: Results of Human Evaluation

that explicitly learning how to generate suggestions and explanations is a proper fashion. Furthermore, the *Discourse-aware MTL mT5* outperforms other models in all ROUGE metrics. It shows that introducing weighted loss benefits the explanation generation as well as the suggestion generation.

We also measure the statistical significance level with the sampling-based bootstrap test, following the guidelines of (Dror et al., 2018). We compare the *DMTL mT5* with *mT5* and *MTL mT5* on the full response, and *DMTL mT5* significantly outperforms the other models at $p < 0.05$. To further measure the qualities of the generated suggestions and explanations, we also conduct qualitative human evaluation in Section 4.2.

4.2 Human Evaluation

For human evaluation, we invite a group of physicians and randomly sample 100 instances from the test set, where each instance is assigned to two physicians to assess the following three aspects:

1. **Relevance**: whether the generated response is related to the patient's question.
2. **Correctness of suggestion**: whether the generated suggestion is correct.
3. **Correctness of explanation**: whether the generated explanation can explain the generated suggestion and help patients understand the reason why such a suggestion is given.

Note that for the multi-tasking methods, we present the generated suggestion and explanation of each instance jointly as one response to the physicians. Each aspect is ranging from zero (does not meet the given aspect) to five (totally meets the given aspect).

The evaluation results are reported in Table 5, where Relevan., Suggest., and Explan. correspond to the three aspects described above, respectively. The *Discourse-aware MTL mT5* achieves the highest scores in all aspects, suggesting that the discourse-aware mechanism enables the model to generate explanations more aligned with suggestions, and makes the response more relevant to the question. Compared to the single task learning *mT5*, *MTL mT5* and *Discourse-aware MTL mT5* obtain higher scores on "Correctness of Suggestion" and "Correctness of Explanation", indicating that multi-task learning makes the model more attend on learning information benefiting both tasks. Overall, our proposed models achieve promising performances on generating suggestions by learning from the QA pairs only. However, the scores obtained on "Correctness of Explanation" are lower than half of the full score. This might indicate that generating correct explanations is still challenging due to the lack of medical knowledge.

5 Conclusion

This paper proposes a discourse-aware generative model based on multi-task learning to generate narrative structured responses consisting of suggestions and explanations to the questions. Experimental results show that our model with the discourse-aware mechanism outperforms baseline models on both quantitative and qualitative evaluations. However, based on the human evaluation results, there is still ample room for improvement on providing medical explanations. As the correctness of explanation is still relatively lower than our expectation. On the other hand, without integrating explicit medical knowledge, there exists potential risks of producing unfaithful results. In the future, we plan to incorporate external domain knowledge, e.g., medical knowledge base, into the model to generate enriched and faithful explanations that are not only relevant to suggestions, but also contain correct information.

Acknowledgements

This research was partially supported by Ministry of Science and Technology, Taiwan, under grants MOST 110-2221-E-002-128-MY3 and MOST 110-2634-F-002-050-.

References

Faeze Brahman, Vered Shwartz, Rachel Rudinger, and Yejin Choi. 2021. Learning to rationalize for nonmonotonic reasoning with distant supervision. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 35, pages 12592–12601.

Rotem Dror, Gili Baumer, Segev Shlomov, and Roi Reichart. 2018. The hitchhiker's guide to testing sta-

tistical significance in natural language processing. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1383–1392, Melbourne, Australia. Association for Computational Linguistics.

Mor Geva, Yoav Goldberg, and Jonathan Berant. 2019. Are we modeling the task or the annotator? an investigation of annotator bias in natural language understanding datasets. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 1161–1166, Hong Kong, China. Association for Computational Linguistics.

Suchin Gururangan, Swabha Swayamdipta, Omer Levy, Roy Schwartz, Samuel Bowman, and Noah A Smith. 2018. Annotation artifacts in natural language inference data. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 2 (Short Papers)*, pages 107–112, New Orleans, Louisiana. Association for Computational Linguistics.

Alon Jacovi and Yoav Goldberg. 2020. Towards faithfully interpretable nlp systems: How should we define and evaluate faithfulness? In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 4198–4205, Online. Association for Computational Linguistics.

Jie Lei, Licheng Yu, Tamara Berg, and Mohit Bansal. 2020. What is more likely to happen next? video-and-language future event prediction. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 8769–8784, Online. Association for Computational Linguistics.

Qing Li, Qingyi Tao, Shafiq Joty, Jianfei Cai, and Jiebo Luo. 2018. Vqa-e: Explaining, elaborating, and enhancing your answers for visual questions. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 552–567.

Chin-Yew Lin. 2004. ROUGE: A package for automatic evaluation of summaries. In *Text Summarization Branches Out*, pages 74–81, Barcelona, Spain. Association for Computational Linguistics.

James Mullenbach, Sarah Wiegreffe, Jon Duke, Jimeng Sun, and Jacob Eisenstein. 2018. Explainable prediction of medical codes from clinical text. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 1101–1111, New Orleans, Louisiana. Association for Computational Linguistics.

Ankur Parikh, Xuezhi Wang, Sebastian Gehrmann, Manaal Faruqui, Bhuwan Dhingra, Diyi Yang, and Dipanjan Das. 2020. ToTTo: A controlled table-to-text generation dataset. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1173–1186, Online. Association for Computational Linguistics.

Pouya Pezeshkpour, Yifan Tian, and Sameer Singh. 2019. Investigating robustness and interpretability of link prediction via adversarial modifications. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 3336–3347, Minneapolis, Minnesota. Association for Computational Linguistics.

Nazneen Fatema Rajani, Bryan McCann, Caiming Xiong, and Richard Socher. 2019. Explain yourself! leveraging language models for commonsense reasoning. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 4932–4942, Florence, Italy. Association for Computational Linguistics.

Ehud Reiter. 2019. Natural language generation challenges for explainable ai. In *Proceedings of the 1st Workshop on Interactive Natural Language Technology for Explainable Artificial Intelligence (NL4XAI 2019)*, pages 3–7. Association for Computational Linguistics.

Marco Tulio Ribeiro, Sameer Singh, and Carlos Guestrin. 2016. " why should i trust you?" explaining the predictions of any classifier. In *Proceedings of the 22nd ACM SIGKDD international conference on knowledge discovery and data mining*, pages 1135–1144.

Ronald J Williams and David Zipser. 1989. A learning algorithm for continually running fully recurrent neural networks. *Neural computation*, 1(2):270–280.

Jialin Wu and Raymond Mooney. 2019. Faithful multimodal explanation for visual question answering. In *Proceedings of the 2019 ACL Workshop BlackboxNLP: Analyzing and Interpreting Neural Networks for NLP*, pages 103–112, Florence, Italy. Association for Computational Linguistics.

Linting Xue, Noah Constant, Adam Roberts, Mihir Kale, Rami Al-Rfou, Aditya Siddhant, Aditya Barua, and Colin Raffel. 2021. mt5: A massively multilingual pre-trained text-to-text transformer. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 483–498, Online. Association for Computational Linguistics.

DeltaNet: Conditional Medical Report Generation for COVID-19 Diagnosis

Xian Wu[1], Shuxin Yang[2,5], Zhaopeng Qiu[1], Shen Ge[1], Yangtian Yan[1], Xingwang Wu[3]
Yefeng Zheng[1], S. Kevin Zhou[4], Li Xiao[2,5]*
[1]Tencent Jarvis Lab [2]Institute of Computing Technology of the Chinese Academy of Sciences
[3]The First Affiliated Hospital of Anhui Medical University
[4]University of Science and Technology of China
[5]University of Chinese Academy of Sciences
{kevinxwu, shuxinyang, zhaopengqiu, shenge, yangtianyan, yefengzheng}@tencent.com;
duobi2004@126.com; s.kevin.zhou@gmail.com;xiaoli@ict.ac.cn

Abstract

Fast screening and diagnosis are critical in COVID-19 patient treatment. In addition to the gold standard RT-PCR, radiological imaging like X-ray and CT also works as an important means in patient screening and follow-up. However, due to the excessive number of patients, writing reports becomes a heavy burden for radiologists. To reduce the workload of radiologists, we propose DeltaNet to generate medical reports automatically. Different from typical image captioning approaches that generate reports with an encoder and a decoder, DeltaNet applies a conditional generation process. In particular, given a medical image, DeltaNet employs three steps to generate a report: 1) first retrieving related medical reports, i.e., the historical reports from the same or similar patients; 2) then comparing retrieved images and current image to find the differences; 3) finally generating a new report to accommodate identified differences based on the conditional report. We evaluate DeltaNet on a COVID-19 dataset, where DeltaNet outperforms state-of-the-art approaches. Besides COVID-19, the proposed DeltaNet can be applied to other diseases as well. We validate its generalization capabilities on the public IU-Xray and MIMIC-CXR datasets for chest-related diseases.

1 Introduction

Since December 2019, the world has been suffering from a serious health crisis: the outbreak of COVID-19 (Wang et al., 2020). Fast screening and diagnosis is critical in COVID-19 patient treatment. In clinical practice, the Reverse Transcription Polymerase Chain Reaction (RT-PCR) is recognized as the golden standard (Zu et al., 2020). However, due to high false-negative rate and shortage of equipment (Fang et al., 2020; Ng et al., 2020), medical imaging like X-ray and Computed Tomography (CT) (Rubin et al., 2020) also works as an alternative means in COVID-19 diagnosis and treatment which generates more timely results than RT-PCR and helps in evaluating the severity degree of COVID-19.

Given medical images of COVID-19 patients, radiologists need to write relatively long reports to address the impressions and findings. Considering the large volume of COVID-19 patients and potentially infected population, writing medical reports becomes a heavy burden for radiologists. Furthermore, due to the varied expertise of radiologists, some abnormalities in medical images may be ignored and thus not included in the final reports. To alleviate the heavy workload and aid less experienced radiologists, automatically generating medical reports becomes a critical task.

Due to its importance in clinical practice, automatically generating medical reports has attracted extensive research interests in recent years (Zhou et al., 2021). Existing works mainly follow the image captioning approaches and employ an encoder-decoder process. In the encoding stage, the visual features are extracted from medical images via a CNN; in the decoding stage, the reports are generated sequentially via an RNN. Such a two-step framework has been proven effective in generating general image captions, such as MS COCO. However, when applied to medical images, it may have following two problems: 1) visual bias: for most cases, the abnormal regions only occupy a small fraction of the entire medical images, thus visual features of normal regions dominate the extracted visual embedding, and abnormal regions are difficult to identify; 2) textual bias: in current medical reports, the majority of transcription focuses on describing the normal regions which distract the model from abnormal regions in training.

To address two problems mentioned above, we propose DeltaNet, which is customized for medical images. Different from encoder-decoder frameworks that generate reports from a single medical

*Corresponding author

Proceedings of the 29th International Conference on Computational Linguistics, pages 2952–2961
October 12–17, 2022.

image, DeltaNet introduces a retrieve-update process which consists of the following three steps: 1) Retrieval: DeltaNet firstly retrieves conditional medical images and reports from medical records. For patients who've already been examined before, we directly obtain his previous medical images and reports. For patients examined for the first time, we retrieve the medical images and reports with similar visual features from other patients; 2) Comparison: DeltaNet compares the embedding difference between two medical images to capture the visual difference; 3) Based on the identified visual difference and the retrieved reports, DeltaNet conditionally generates the final report for current medical image.

To prove the effectiveness of the proposed DeltaNet, we collect a dataset of COVID-19 patients. For each patient, we manage to collect all the historical medical images and the corresponding reports that cover the complete treatment process. The experimental results show that the proposed DeltaNet outperforms the baselines, including both the general image captioning approaches and the existing medical report generation works. Besides generating reports, DeltaNet can also highlight the differences between previous and current medical images and the corresponding updated reports, which provides a clear explanation for the generated report. In addition to COVID-19, the proposed DeltaNet can also be applied to medical report generation of other diseases. We evaluate the generalization ability of DeltaNet on IU-Xray and MIMIC-CXR datasets. DeltaNet consistently outperforms baselines.

2 Related Works

2.1 Image Captioning

Image captioning aims to provide a short descriptive sentence for a given image, which has recently received extensive research interests (Vinyals et al., 2015; Xu et al., 2015; Anderson et al., 2018; Huang et al., 2019). Typical image captioning models, e.g., (Anderson et al., 2018; Huang et al., 2019), adopt the encoder-decoder framework to accomplish the image captioning task, where the encoder extracts the visual representations from the images, and the decoder transforms the acquired visual features to texts. Different from typical image captioning approaches, DeltaNet applies a retrieve-update process, consisting of retrieval, comparison, and generation steps to generate reliable and robust reports.

2.2 Medical Report Generation

Medical report generation generally produces longer texts than typical image captions (Zhou et al., 2019). Employing the typical image caption encoder-decoder framework, Jing et al. (2018) proposed to use a two-level hierarchical LSTM to deal with long reports, with the top-level handling topic generation and the bottom-level generating texts according to the currently selected topic; Wang et al. (2018); Xue and Huang (2019); Yuan et al. (2019) used attention mechanism to drive the encoder and the decoder to emphasize on more informative words or visual regions, resulting in improved performance; Li et al. (2018, 2019); Zhang et al. (2020); Liu et al. (2021) introduced external information such as template or knowledge graph to guide the generation of medical reports; Syeda-Mahmood et al. (2020) invited four clinicians to manually examine 220,000 reports and build a taxonomy of 11,000 unique terms for developing an automatic labeling algorithm. Then, they built and customized similar reports from a large report database by fine-grained labels as the generated report. However, for diseases not labeled in advance, this method may not fit well. Some other works (Wang et al., 2018) brought in auxiliary tasks to improve report generation, usually requiring extra expert labeling. Our work uses a new retrieve-update process to effectively generate reports automatically conditioned on historical reports. Besides, our method enhances the model by only using the information from the training dataset, avoiding any expert labeling or external information.

3 Model

In this section, we introduce the proposed DeltaNet. Firstly, we formulate the conditional medical report generation problem; secondly, we describe the basic encoder-decoder based generation model; then, we propose DeltaNet, a conditional medical report generation model; finally, we further extend DeltaNet to exploit multiple conditional reports.

3.1 Problem Formulation

We use a quadruple $q = \{I, I_c, R, R_c\}$ to refer to an input instance, in which I denotes the input medical image and $R = \{w_1, w_2, \ldots, w_N\}$ denotes the corresponding report to be generated. In this manner, medical report generation can be formulated as estimating the parameters of the con-

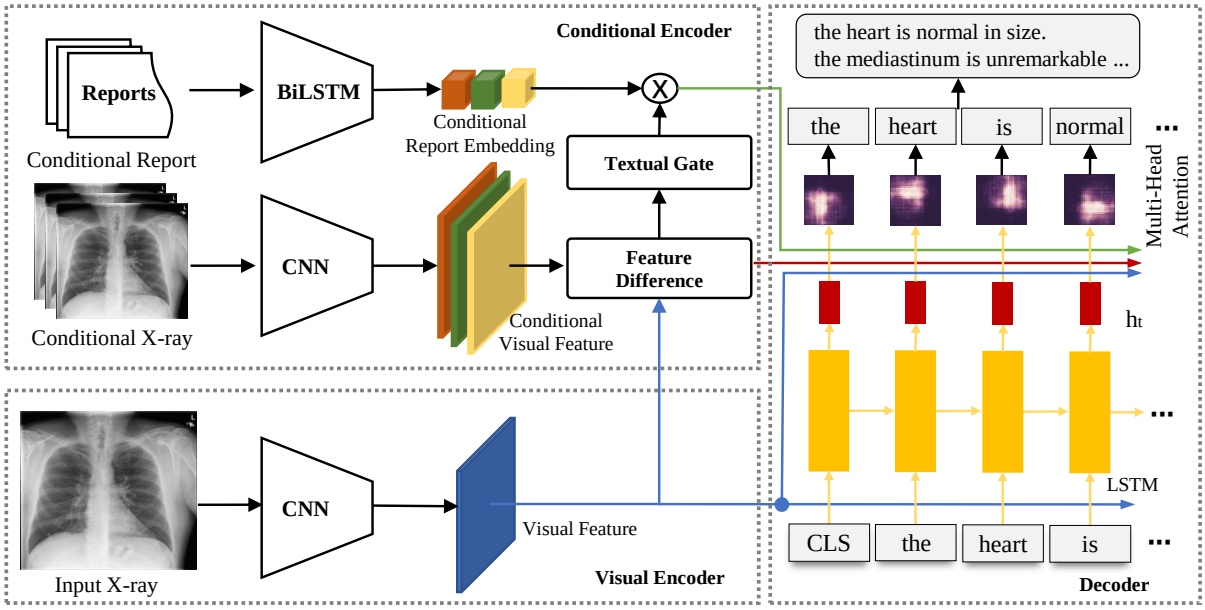

Figure 1: Framework of the proposed DeltaNet model including three major component: visual encoder, conditional encoder and decoder. The visual encoder extracts features from input X-ray image. The conditional encoder extracts features from conditional X-ray image, and embeds the conditional report by a BiLSTM, and acquires the difference between input x-ray and conditional X-ray. The decoder decodes all features, difference and embeddings to generate output report.

ditional probability $P(R|I)$. In this paper, we introduce a conditional image I_c and a conditional report $R_c = \{w_1, w_2, \ldots, w_{N_c}\}$. For patients who have historical medical images, we select their previous medical image and report as I_c and R_c respectively; as for patients without historical reports, we retrieve the most similar medical image I_c and report R_c from the pre-built medical report repository. Then the conditional medical report generation problem can be formulated as $P(R|I, I_c, R_c)$.

3.2 Basic Model

Many existing medical report generation works follow the image captioning approaches and employ the encoder-decoder two-step manner. Typically, a Convolution Neural Network (CNN) is introduced as the encoder, which extracts visual features from the input medical image. In this paper, we extract the output of the last convolution layer of the visual encoder following a linear projection layer as visual features:

$$V = \text{CNN}(I), \tag{1}$$

where $V \in \mathbb{R}^{K \times D}$, K denotes the size of feature maps and D denotes the number of feature maps; in this paper, K is set to 49 and D is set to 512.

After acquiring the visual features V, the next step is to generate medical reports which we refer as the decoding process. In the decoding process, we use LSTM to generate medical report sequentially. At each timestamp, we firstly acquire current hidden state:

$$h_t = \text{LSTM}(w_{t-1}, h_{t-1}), \tag{2}$$

where $h_0 = \mathbf{0}$ and $h_t \in \mathbb{R}^{1 \times D}$.

After acquiring h_t, we attend it to the extracted visual features. In Eq.(3), we use h_t as query and V as both the key and value:

$$a_t = \text{MHA}(h_t, V, V), \tag{3}$$

where $\text{MHA}(\cdot, \cdot, \cdot)$ refers to the multi-head attention function proposed in (Vaswani et al., 2017). The attention embedding $a_t \in \mathbb{R}^{1 \times D}$ can be regarded as attended visual features given the current hidden state.

Finally, we combine the hidden state h_t and the attention embedding a_t to estimate the probability distribution of generating w_t. The probability distribution can be formulated as:

$$P(w_t|w_1, \ldots, w_{t-1}, V) = \sigma([h_t; a_t]W_p), \tag{4}$$

where $W_p \in \mathbb{R}^{2D \times E}$ is a learnable linear projection and E is vocabulary size.

3.3 Conditional Generation Model

In Section 3.2, we adopt the two-step encoder-decoder framework to generate medical reports. Although such a two-step approach has been proven effective in general image captioning, e.g., MSCOCO, due to the unavailability of large-scale labeled data, it is difficult to generate an accurate yet fluent report when applying to the medical report generation problem.

Observing the fact that during a complete treatment process, one patient usually have been examined for multiple times, therefore he may have multiple medical images. For example, the COVID-19 patients may first take a chest X-ray examination for diagnosis and take several more X-ray to track the severity progresses of COVID-19 during the treatment. We review the medical reports from the same patient and find that the consecutive reports share the majority of the content and only differ in the disease progresses. Therefore, a natural thought is to generate the medical report in a conditional manner, not only from the current input medical image but also from the historical images and reports. For first time patients without previous reports, we can also select reports with similar visual appearances from other patients. For all medical images, we extract their visual features with Eq.(5). Then given an input image, we retrieve the most similar images according to the cosine distance of visual features. Let I_c and R_c denote the conditional image and report respectively, then conditional medical report generation problem can be formulated as $P(R|I, I_c, R_c)$. Here we propose DeltaNet as the medical report's conditional generation model.

DeltaNet model first extracts the visual features V_c and textual features T_c from the conditional image and report.

$$V_c = \text{CNN}(I_c), \tag{5}$$

$$T_c = \text{BiLSTM}(R_c), \tag{6}$$

.

As demonstrated in (Wu et al., 2018), the subtraction of visual features is an efficient operation to acquire the difference between features. Thus DeltaNet acquires the varied visual features between V and V_c using:

$$\Delta V_c = V - V_c, \tag{7}$$

In the decoding process, at each timestamp, after generating the hidden state h_t with Eq.(2), we attend h_t to both the varied visual features of medical images and the textual features from the conditional report as follows:

$$s_t = \text{MHA}(h_t, \Delta V_c, \Delta V_c), \tag{8}$$

$$c_t = \text{MHA}(h_t, T_c, T_c), \tag{9}$$

Finally, besides the h_t and a_t, we utilize s_t and c_t to estimate the probability distribution of generating w_t. The probability distribution can be formulated as:

$$\begin{aligned} P(w_t|w_1, \ldots, w_{t-1}, V) \\ = \sigma([h_t; a_t; s_t; c_t]W_p), \end{aligned} \tag{10}$$

where W_p is a learnable linear projection in the shape of $4D \times E$ and E is vocabulary size.

3.4 Multiple Conditional Generation Model

In the previous subsection, we mainly deal with the cases that the patient only has one conditional report. However, some patients have multiple historical reports. For first time patients without previous reports, we can also retrieve multiple reports according to the visual similarity from other patients. In this subsection, we extend the proposed DeltaNet to fit for multiple conditional reports.

For each input image, we assume that it has L conditional images and reports, we denote them with two sets $\{I_c^{(1)}, I_c^{(2)}, \ldots, I_c^{(L)}\}$ and $\{R_c^{(1)}, R_c^{(2)}, \ldots, R_c^{(L)}\}$. DeltaNet firstly extracts the visual features and textual features from conditional images and reports by two encoders:

$$V_c^{(i)} = \text{CNN}(I_c^{(i)}), \tag{11}$$

$$T_c^{(i)} = \text{BiLSTM}(R_c^{(i)}), \tag{12}$$

Then DeltaNet acquires varied visual features:

$$\Delta V_c^{(i)} = V - V_c^{(i)}. \tag{13}$$

As shown in Figure 1, for each textual feature $T_c^{(i)}$ extracted from the conditional report, we add a textual gate to control its contribution since the contribution of each report is different. Here we introduce a gate weight g_i as follows:

$$g^{(i)} = \sigma(W_v V + W_c V_c^{(i)} + b_i), \tag{14}$$

where W_v and W_c are learnable linear projections. The $g^{(i)}$ takes visual features of both input image and conditional image into consideration which is used to re-weight the textual features:

$$\hat{T}_c^{(i)} = g^{(i)} * T_c^{(i)}. \tag{15}$$

DeltaNet further concatenate varied visual features and weighted textual features in a row-wise manner:

$$\Delta V_c = [\Delta V_c^{(1)}; \ldots; \Delta V_c^{(L)}], \quad (16)$$

$$T_c = [\hat{T}_c^{(1)}; \ldots; \hat{T}_c^{(L)}], \quad (17)$$

where $\Delta V_c \in \mathbb{R}^{KL \times D}$ and $T_c \in \mathbb{R}^{N_c L \times D}$.

At each timestamp, DeltaNet use multi-head attention to generate the subtractive visual attend feature s_t and the context embedding c_t.

$$s_t = \text{MHA}(h_t, \Delta V_c, \Delta V_c), \quad (18)$$

$$c_t = \text{MHA}(h_t, T_c, T_c), \quad (19)$$

where both s_t and c_t are in the shape of $1 \times D$.

Then in each time step, the word w_t is generated according to the probability in Eq.(20).

$$P(w_t|w_1, \ldots, w_{t-1}, V) = \sigma([h_t; a_t; s_t; c_t]W_p). \quad (20)$$

4 Experiments

In this section, we evaluate the proposed DeltaNet from three perspectives: (1) Whether incorporating conditional reports can bring in performance gain in medical report generation; (2) Whether increasing the number of conditional reports can further improve the performance; (3) Which will perform better, the conditional reports that are acquired from the historical reports of patients themselves or retrieved from other patients according to visual similarity.

4.1 Implementation Details

For first time patients without historical reports, we retrieve conditional reports via the embedding similarity of their corresponding images. The embeddings are acquired from an encoder pre-trained on the ChestX-Ray14 dataset (Wang et al., 2017). Since the lengths of conditional reports are different, we use zero-padding for each conditional case to pad each report to a fixed length. We adopt ResNet-152 as our visual encoder for both current input image and conditional images, a two-layer bidirectional LSTM as conditional report encoder, and a single-layer LSTM as medical report decoder. The dimension of the visual feature and hidden states are set to 512. We adopt the Adam optimizer with an initial learning rate of 5e-4 and a mini-batch size of 32. We train the model with cross entropy loss for 100 epochs and early stop strategy is adopted.

4.2 Datasets and Settings

We introduce three data sets COVID-19, IU-Xray (Demner-Fushman et al., 2016) and MIMIC-CXR (Johnson et al., 2019) to evaluate the proposed DeltaNet.

The COVID-19 dataset includes 1,261 exams (including both images and reports) from 1,085 patients from mobile field hospitals used in the COVID-19 pandemic. The private information has been removed during the data collection process. Each case includes both the X-ray images and the corresponding reports. The max, median, and mean length of the reports are 180, 69, and 72 words, respectively. Thus the report to be generated are relatively lengthy. The ratio of patients with only one visit is 57.48%. Among these 1,261 cases, in terms of disease severity, 112 cases are severe, 1,113 cases are general, 30 cases are mild, and 6 cases are not labeled. More than 166 patients have more than 2 reports; we select the latest one as the prediction target and select the second latest one as the conditional report. We split these 166 patients in the ratio of 7:1:2, that is 116 for training, 16 for validating, and 34 for testing. These three groups have no overlap in patients.

IU-Xray (Demner-Fushman et al., 2016) is a public data set which includes 3,955 radiology reports and 7,470 frontal- and lateral-view chest X-ray images. We follow the split of (Chen et al., 2020) which divides the entire data set into train/validation/test in the ratio 7:1:2.

MIMIC-CXR (Johnson et al., 2019) is the largest dataset for the medical report generation, which contains 377,110 chest X-ray images and 227,827 free-text radiology reports. The dataset contains multi-view images, and we select frontal view images in this work.

For the COVID-19 data set, the conditional report is selected from the previous one of the same patient. As to the IU-Xray data set, we are unable to acquire the historical reports for a patient. Therefore we first acquire the visual feature for each image via a CNN based model pre-trained on the ChestX-Ray14 dataset. Then for each image, we select the conditional reports via cosine similarity of visual feature. For the case in the train/validate/test groups, we select their conditional reports only from the training group. In this case, we avoid the situation that the report to be generated appears as the conditional reports in the training data set. As a result, the label leakage is prevented.

Table 1: The performance of baselines and the proposed DeltaNet on IU-Xray and MIMIC dataset. The conditional reports are extracted from reports of other patients according to visual similarity. For the baseline methods, we cite their performance reported in their papers and reported in (Li et al., 2018) and (Jing et al., 2018). The *Basic* refers to the one introduced in Section 3.2.

Dataset	Model	BLEU-1	BLEU-2	BLEU-3	BLEU-4	CIDEr	ROUGE-L
IU-Xray	S&T (Vinyals et al., 2015)	0.216	0.124	0.087	0.066	0.294	0.306
	SA&T (Xu et al., 2015)	0.399	0.251	0.168	0.118	0.302	0.323
	AdaAtt (Lu et al., 2017)	0.220	0.127	0.089	0.068	0.295	0.308
	TieNet (Wang et al., 2018)	0.286	0.160	0.104	0.074	/	0.226
	CoAtt (Jing et al., 2018)	0.455	0.288	0.205	0.154	0.277	0.369
	R2Gen (Chen et al., 2020)	0.470	0.304	0.219	0.165	/	0.371
	PPKED (Liu et al., 2021)	0.483	0.315	0.224	0.168	0.351	0.376
	Basic	0.417	0.264	0.184	0.138	0.467	0.343
	DeltaNet-1C	0.470	0.307	0.224	0.175	**0.853**	0.369
	DeltaNet-3C	**0.485**	**0.324**	**0.238**	**0.184**	0.802	**0.379**
MIMIC	S&T (Vinyals et al., 2015)	0.256	0.157	0.102	0.070	0.063	0.249
	SA&T (Xu et al., 2015)	0.304	0.177	0.112	0.077	0.083	0.249
	AdaAtt (Lu et al., 2017)	0.311	0.178	0.111	0.075	0.084	0.246
	BU&TD (Anderson et al., 2018)	0.280	0.169	0.108	0.074	0.073	0.250
	R2Gen (Chen et al., 2020)	0.353	0.218	0.145	0.103	/	0.277
	PPKED (Liu et al., 2021)	0.360	0.224	0.149	0.106	/	**0.284**
	Basic	0.335	0.206	0.138	0.100	0.156	0.263
	DeltaNet-1C	0.355	0.221	0.152	0.113	0.220	0.279
	DeltaNet-3C	**0.361**	**0.225**	**0.154**	**0.114**	**0.281**	0.277

For the MIMIC-CXR data set, we exclude the patients without reports. Then we generate two data sets to evaluate two types of conditional report respectively: 1) MIMIC: this data set is the same as original MIMIC-CXR dataset. We follow standard split provided by (Johnson et al., 2019) to divide the entire data set into train/validate/test and the conditional cases are retrieved from the training set using visual feature similarity; 2) MIMIC-Multi-Visit: we first select the patients with more than three reports as candidates. We choose their latest reports as the prediction target and select three most recent previous reports as the conditional reports. The offical training/validation/testing sets split of MIMIC-CXR is 222,758/1,808/3,269, respectively. From the orignial MIMIC-CXR data set, we select the patients with more than 3 visits, resulting the data set MIMIC-Multi-Visit of 11,978/83/165, respectively. To conduct a fair comparison, during the training of the basic model and existing models, we append the conditional reports into their training set. In this manner, the proposed DeltaNet will not include extra labeled data in training.

We select the popular metrics for natural language generation (NLG) tasks and a specific clinical metric for evaluation. The NLG metrics include BLEU-n (Papineni et al., 2002), CIDEr (Vedantam et al., 2015), and ROUGE-L (Lin, 2004) score. The results are computed by MS-COCO caption evaluation tool [1] automatically. The clinical efficacy (CE) metric is proposed by R2gen (Chen et al., 2020) to quantify the precision, recall, and f1 score of medical terminology described in reference and generated reports. Because the IU-Xray dataset does not provide such labels, we only report CE metric on the MIMIC-CXR dataset. For consistency, we employ the CheXpert (Irvin et al., 2019) to extract the labels from generated reports.

4.3 Quantitative Results

Baselines. The Tables display the comparison results between existing works and the proposed DeltaNet. Among the baselines, S&T (Vinyals et al., 2015), SA&T (Xu et al., 2015), AdaAtt (Lu et al., 2017) and BU&TD (Anderson et al., 2018) belong to general image captioning approaches; while CoAtt (Jing et al., 2018), TieNet (Wang et al., 2018), R2Gen (Chen et al., 2020) and PPKED (Liu et al., 2021) focus on the specific medical report generation task. Since MIMIC-Multi-Visit only includes patients with more than three images, and COVID-19 is a private data set, we cannot directly cite the results of existing models. Therefore, we re-train existing models on these two data sets. For R2Gen, we directly use the implementation [2] provided by its authors; for CoAtt, we use the third party implementation [3]. For other works, we re-implement them ourselves; The *Basis Model* refers to the one described in *Basic Model* section; The

[1] https://github.com/tylin/coco-caption

[2] https://github.com/cuhksz-nlp/R2Gen

[3] https://github.com/ZexinYan/Medical-Report-Generation

Table 2: The performance of baselines and the proposed DeltaNet on COVID-19 and MIMIC-Multi-Visit data sets. The conditional reports are extracted from patients' own historical reports.

Dataset	Model	BLEU-1	BLEU-2	BLEU-3	BLEU-4	CIDEr	ROUGE-L
COVID-19	S&T (Vinyals et al., 2015)	0.604	0.573	0.546	0.523	0.148	0.639
	SA&T (Xu et al., 2015)	0.619	0.586	0.557	0.534	0.167	**0.642**
	AdaAtt (Lu et al., 2017)	0.617	0.583	0.553	0.529	0.150	0.625
	BU&TD (Anderson et al., 2018)	0.600	0.563	0.531	0.504	0.120	0.603
	R2Gen (Chen et al., 2020)	0.610	0.576	0.547	0.523	0.176	0.624
	Basic	0.614	0.566	0.530	0.503	0.256	0.610
	DeltaNet	**0.664**	**0.622**	**0.588**	**0.561**	**0.273**	0.635
MIMIC-Multi-Visit	S&T (Vinyals et al., 2015)	0.300	0.176	0.111	0.074	0.066	0.246
	SA&T (Xu et al., 2015)	0.284	0.170	0.114	0.080	0.046	0.234
	AdaAtt (Lu et al., 2017)	0.310	0.180	0.114	0.075	0.068	0.240
	BU&TD (Anderson et al., 2018)	0.284	0.170	0.111	0.079	0.089	0.253
	R2Gen (Chen et al., 2020)	0.303	0.193	0.131	0.095	0.148	0.266
	Basic	0.325	0.196	0.129	0.091	0.102	0.248
	DeltaNet-1C	0.358	0.217	0.144	0.103	0.248	**0.270**
	DeltaNet-3C	**0.371**	**0.228**	**0.152**	**0.107**	**0.301**	0.264

DeltaNet and *DeltaNet-1C* refers to the one introduced in *Conditional Generation Model* section; The *DeltaNet-3C* refers to the one introduced in *Multiple Conditional Generation Model* section with three conditional reports.

Conditional Reports from Other Patients. We evaluate DeltaNet with conditional reports retrieved by the visual feature similarity from other patients. Here we conduct experiments on IU-Xray and MIMIC data sets. Since these two data sets are public data sets and the split of train/validate/test is in the standard manner, for existing works, we directly cite the reported performance. As shown in Table 1, *DeltaNet* consistently outperforms the *Basic Model*. This demonstrates that besides using the historical reports of the same patient as the conditional report, selecting similar reports as conditional reports can also boost the performance. As shown in Table 1 and Table 3, for the existing methods, the proposed DeltaNet with three conditional reports outperforms almost all baselines on both NLG and clinical efficiency metrics. As to the number of conditional reports, we evaluate DeltaNet with 1 and 3 conditional reports. As shown in both Table 2 and Table 1, the inclusion of more conditional reports generally increases the performance.

Conditional Reports from the Patients Themselves. As shown in Table 2, *DeltaNet* significantly outperforms the *Basic Model* which proves the effectiveness of introducing conditional reports in medical report generation. Furthermore, the proposed *DeltaNet* outperforms state-of-the-art approaches almost all the metrics.

Table 3: The performance of clinical efficiency.

Dataset	Model	Precision	Recall	F1
MIMIC	S&T	0.084	0.066	0.072
	SA&T	0.181	0.134	0.144
	AdaAtt	0.265	0.178	0.197
	BU&TD	0.166	0.121	0.133
	R2Gen	0.333	0.273	0.276
	DeltaNet-1C	0.460	0.353	0.376
	DeltaNet-3C	**0.470**	**0.399**	**0.406**

4.4 Ablation Study

Self vs. Other Patients. Here we conduct an experiment to compare the performance of two types of conditional reports. We use MIMIC-CXR dataset and select the patients with $\geq$2 reports. For each patient, we select two most recent reports, that is 49,180 images from 24,590 patients. The method *Self* denotes that the conditional report is acquired from the same patient; the method *Others* denotes that the conditional report is retrieved via visual feature similarity. To be fair, the conditional reports of the *Self* methods are added into the training set of the *Others* method. As shown in Table 4, the BLEU-4 is 0.114 vs. 0.101. The conditional report from the same patient outperforms the one from other patients.

Table 4: The performance of two types of conditional reports.

Conditional Reports	BLEU-3	BLEU-4	CIDE-r	ROUGE-L
Others	0.140	0.101	0.179	0.278
Self	**0.158**	**0.114**	**0.231**	**0.289**

Effectiveness of Conditional Image and Report. Compared with the basic model, our model utilizes two additional features I_c and R_c. To analyze the

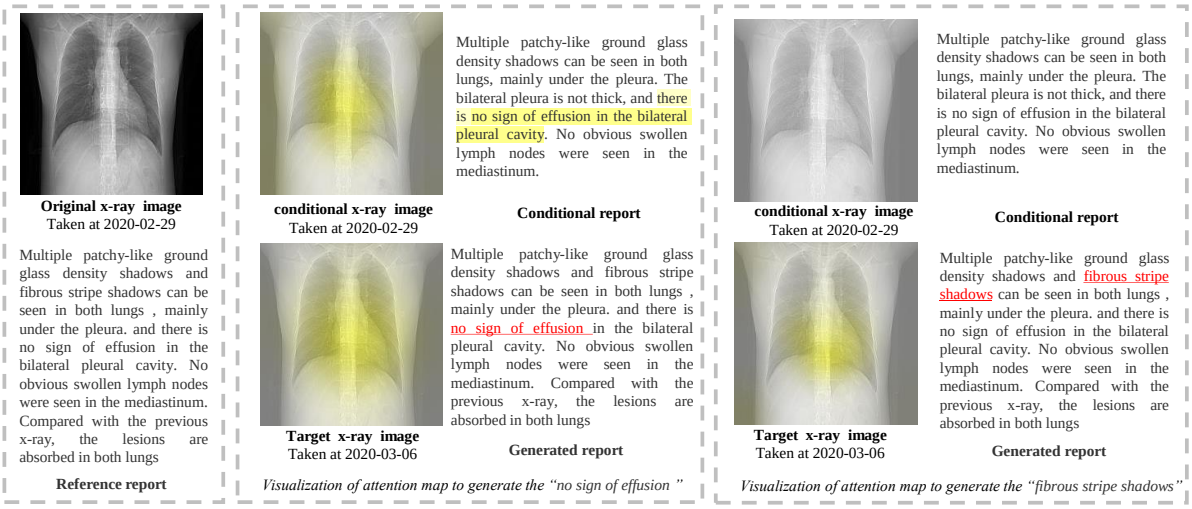

Figure 2: Visualizations of attention map on conditional/target image and report when DeltaNet generates the highlight words.

contribution of each feature, we design two models which separately concatenate I_c or R_c with visual features. As shown in first two rows of Table 5, the model with conditional reports outperforms the model with conditional images, which shows that the conditional reports are more effective than conditional images. To analyze the effectiveness of image feature subtraction in Eq.(7), we design a model which directly concatenates both with visual features. As shown in "$I_c + R_c$" and ours rows of Table 5, the model directly concatenates with both features is lower than DeltaNet, which shows the benefit by combining current input and conditional input by feature subtraction and cross attention in DeltaNet.

Table 5: The contribution of conditional image and report. I_c and R_c refer to the features of conditional images and reports. $I_c + R_c$ refers to feature concatenation.

Dataset	Model	BLEU-3	BLEU-4	ROUGE-L
IU-Xray	I_c	0.137	0.103	0.281
	R_c	0.176	0.134	0.310
	I_c+R_c	0.208	0.162	0.329
	Ours	**0.224**	**0.175**	**0.369**
MIMIC-Multi-Visit	I_c	0.108	0.075	0.222
	R_c	0.126	0.090	0.238
	I_c+R_c	0.127	0.091	0.242
	Ours	**0.144**	**0.103**	**0.270**
COVID-19	I_c	0.481	0.456	0.572
	R_c	0.507	0.482	0.595
	I_c+R_c	0.531	0.508	0.619
	Ours	**0.588**	**0.561**	**0.635**

4.5 Case Study

In this section, we demonstrate the effectiveness of DeltaNet with a case study of a COVID-19 patient. Here we select a patient who has two consecutive X-ray examinations during COVID treatment. This patient is in mild COVID-19 severity. We use the report taken on Feb 29, 2020 as the conditional report to generate the report taken on March 6, 2020.

On the left part of Figure 2, we visualize the attention heat map to generate the phrase "no sign of effusion". The finding of "no sign of effusion" remains the same in both conditional and current reports. Therefore, generating this phrase attends to both current and conditional medical images. It also attends to the corresponding phrase in the conditional reports. As a result, the combination of the conditional report and both images enable the correct generation of this phrase.

On the right of Figure 2, we visualize the attention heat map to generate the phrase "fibrous stripe shadows". Since this is a new finding which does not exist in a conditional report, it only attends to the current medical image. The proposed DeltaNet can learn the difference between conditional and current images and correspondingly generate this new finding in the target report.

5 Conclusion and Future Works

In this paper, we targeted to automate the medical report generation. Different from typical encoder-decoder framework, we proposed a conditional generation model DeltaNet. For patients with historical

reports, we combined the input image with the historical report to generate reports. For first time patients without historical reports, we retrieved visual similar reports from other patients as conditional reports. We proved the advantage of the proposed DeltaNet over state-of-the-art approaches on the IU-Xray, MIMIC-CXR and COVID-19 datasets.

In this paper, for new patients without historical reports, we retrieve the conditional reports only via visual similarity. However, more features like age, gender and diseases could be useful in selecting conditional reports. In addition, we may attempt to combine historical reports from the same patient and simliar reports from other patients together to futher improve the performance.

References

Peter Anderson, Xiaodong He, Chris Buehler, Damien Teney, Mark Johnson, Stephen Gould, and Lei Zhang. 2018. Bottom-up and top-down attention for image captioning and VQA. In *CVPR*.

Zhihong Chen, Yan Song, Tsung-Hui Chang, and Xiang Wan. 2020. Generating Radiology Reports via Memory-driven Transformer. In *EMNLP*.

Dina Demner-Fushman, Marc D. Kohli, Marc B. Rosenman, Sonya E. Shooshan, Laritza Rodriguez, Sameer Antani, George R. Thoma, and Clement J. McDonald. 2016. Preparing a collection of radiology examinations for distribution and retrieval. *Journal of the American Medical Informatics Association*, 23(2):304–310.

Yicheng Fang, Huangqi Zhang, Jicheng Xie, Minjie Lin, Lingjun Ying, Peipei Pang, and Wenbin Ji. 2020. Sensitivity of Chest CT for COVID-19: Comparison to RT-PCR. *Radiology*, pages 200–432.

Lun Huang, Wenmin Wang, Jie Chen, and Xiaoyong Wei. 2019. Attention on attention for image captioning. In *ICCV*.

Jeremy Irvin, Pranav Rajpurkar, Michael Ko, Yifan Yu, Silviana Ciurea-Ilcus, Chris Chute, Henrik Marklund, Behzad Haghgoo, Robyn L. Ball, Katie S. Shpanskaya, Jayne Seekins, David A. Mong, Safwan S. Halabi, Jesse K. Sandberg, Ricky Jones, David B. Larson, Curtis P. Langlotz, Bhavik N. Patel, Matthew P. Lungren, and Andrew Y. Ng. 2019. Chexpert: A large chest radiograph dataset with uncertainty labels and expert comparison. In *AAAI*.

Baoyu Jing, Pengtao Xie, and Eric P. Xing. 2018. On the automatic generation of medical imaging reports. In *ACL*.

Alistair E. W. Johnson, Tom J Pollard, Seth J Berkowitz, Nathaniel R Greenbaum, Matthew P Lungren, Chih-ying Deng, Roger G Mark, and Steven Horng. 2019. MIMIC-CXR, a de-identified publicly available database of chest radiographs with free-text reports. *Scientific Data*, 6(1):317.

Christy Y. Li, Xiaodan Liang, Zhiting Hu, and Eric P. Xing. 2019. Knowledge-Driven Encode, Retrieve, Paraphrase for Medical Image Report Generation. In *AAAI*.

Yuan Li, Xiaodan Liang, Zhiting Hu, and Eric P. Xing. 2018. Hybrid Retrieval-Generation Reinforced Agent for Medical Image Report Generation. In *NeurIPS*.

Chin-Yew Lin. 2004. Rouge: A package for automatic evaluation of summaries. In *Text summarization branches out*, pages 74–81.

Fenglin Liu, Xian Wu, Shen Ge, Wei Fan, and Yuexian Zou. 2021. Exploring and distilling posterior and prior knowledge for radiology report generation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 13753–13762.

Jiasen Lu, Caiming Xiong, Devi Parikh, and Richard Socher. 2017. Knowing when to look: Adaptive attention via a visual sentinel for image captioning. In *CVPR*.

Ming-Yen Ng, Elaine YP Lee, Jin Yang, Fangfang Yang, Xia Li, Hongxia Wang, Macy Mei-sze Lui, Christine Shing-Yen Lo, Barry Leung, Pek-Lan Khong, et al. 2020. Imaging profile of the covid-19 infection: radiologic findings and literature review. *Radiology: Cardiothoracic Imaging*, 2(1):e200034.

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. Bleu: a method for automatic evaluation of machine translation. In *Proceedings of the 40th annual meeting of the Association for Computational Linguistics*, pages 311–318.

Geoffrey D. Rubin, Linda B. Haramati, Jeffrey P. Kanne, Neil W. Schluger, and Athol U. Wells. 2020. The role of chest imaging in patient management during the covid-19 pandemic: A multinational consensus statement from the fleischner society. *Radiology*, 296(1):201365.

Tanveer F. Syeda-Mahmood, Ken C. L. Wong, Yaniv Gur, Joy T. Wu, Ashutosh Jadhav, Satyananda Kashyap, Alexandros Karargyris, Anup Pillai, Arjun Sharma, Ali Bin Syed, Orest B. Boyko, and Mehdi Moradi. 2020. Chest X-Ray Report Generation Through Fine-Grained Label Learning. In *MICCAI*.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *NIPS*.

Ramakrishna Vedantam, C Lawrence Zitnick, and Devi Parikh. 2015. Cider: Consensus-based image description evaluation. In *Proceedings of the IEEE*

conference on computer vision and pattern recognition, pages 4566–4575.

Oriol Vinyals, Alexander Toshev, Samy Bengio, and Dumitru Erhan. 2015. Show and tell: A neural image caption generator. In *CVPR*.

C. Wang, P. W. Horby, F. G. Hayden, and G. F. Gao. 2020. A novel coronavirus outbreak of global health concern. *The Lancet*, 395(10223):470–473.

Xiaosong Wang, Yifan Peng, Le Lu, Zhiyong Lu, Mohammadhadi Bagheri, and Ronald M. Summers. 2017. Chest x-ray8: Hospital-scale chest x-ray database and benchmarks on weakly-supervised classification and localization of common thorax diseases. In *CVPR*.

Xiaosong Wang, Yifan Peng, Le Lu, Zhiyong Lu, and Ronald M. Summers. 2018. Tienet: Text-image embedding network for common thorax disease classification and reporting in chest x-rays. In *CVPR*.

Chenfei Wu, Jinlai Liu, Xiaojie Wang, and Xuan Dong. 2018. Object-difference attention: A simple relational attention for visual question answering. In *MM 2018 - Proceedings of the 2018 ACM Multimedia Conference*, volume 1, pages 519–527.

Kelvin Xu, Jimmy Ba, Ryan Kiros, Kyunghyun Cho, Aaron Courville, Ruslan Salakhudinov, Rich Zemel, and Yoshua Bengio. 2015. Show, attend and tell: Neural image caption generation with visual attention. In *ICML*.

Yuan Xue and Xiaolei Huang. 2019. Improved disease classification in chest x-rays with transferred features from report generation. In *IPMI*.

Jianbo Yuan, Haofu Liao, Rui Luo, and Jiebo Luo. 2019. Automatic Radiology Report Generation Based on Multi-view Image Fusion and Medical Concept Enrichment. In *MICCAI 2019*, volume 11769 LNCS, pages 721–729.

Yixiao Zhang, Xiaosong Wang, Ziyue Xu, Qihang Yu, Alan Yuille, and Daguang Xu. 2020. When Radiology Report Generation Meets Knowledge Graph. *Proceedings of the AAAI Conference on Artificial Intelligence*, 34(07):12910–12917.

S Kevin Zhou, Hayit Greenspan, Christos Davatzikos, James S Duncan, Bram Van Ginneken, Anant Madabhushi, Jerry L Prince, Daniel Rueckert, and Ronald M Summers. 2021. A review of deep learning in medical imaging: Imaging traits, technology trends, case studies with progress highlights, and future promises. *Proceedings of the IEEE*.

S Kevin Zhou, Daniel Rueckert, and Gabor Fichtinger. 2019. *Handbook of medical image computing and computer assisted intervention*. Academic Press.

Zi Yue Zu, Meng Di Jiang, Peng Peng Xu, Wen Chen, and Long Jiang Zhang. 2020. Coronavirus disease 2019 (covid-19): A perspective from china. *Radiology*, 296(2):200490.

A Appendix

A.1 Ethical Impact

This work aims to provide efficient and accurate radiology reports to assist radiologists rather than replace radiologists. It benefits the development of human health. We conduct experiments on COVID-19, IU-Xray, and MIMIC-CXR datasets. All personal information was de-identified. We have removed the information related to the data collection. All necessary permissions have been obtained and the appropriate institutional forms have been archived.

A.2 Limitation

In this paper, our method applies a conditional generation process that generates medical reports based on historical medical records. There are still some limitations. First, the generated medical report only contains the diseases in the historical medical records. Second, our method only contains visual and textual information and lacks other examination results (e.g., blood examination and indication) and medical knowledge. We will explore how to involve more features to improve the performance of the proposed method.

A.3 Potential Risks

Given a lot of medical images, it can automatically generate radiology reports. The radiologists only need to make revisions rather than write a new report from scratch. However, inexperienced radiologists may rely on it. Therefore, it is necessary to take additional measures to avoid the abuse of our model.

MCS: An In-battle Commentary System for MOBA Games

Xiaofeng Qi, Chao Li, Zhongping Liang, Jigang Liu, Cheng Zhang, Yuanxin Wei, Lin Yuan, Guang Yang, Lanxiao Huang, Min Li
Tencent Timi L1 Studio, Chengdu, China
sealqi,supeli,skylerliang,jigangliu,donyzhang,winsterwei,
tayloryuan,mikoyang,jackiehuang,konthigali@tencent.com

Abstract

This paper introduces a generative system for in-battle real-time commentary in mobile MOBA games. Event commentary is important for battles in MOBA games, which is applicable to a wide range of scenarios like live streaming, e-sports commentary and combat information analysis. The system takes real-time match statistics and events as input, and an effective transform method is designed to convert match statistics and utterances into consistent encoding space. This paper presents the general framework and implementation details of the proposed system, and provides experimental results on large-scale real-world match data.

1 Introduction

In recent years, MOBA (Multiplayer Online Battle Arena) games have been popular all over the world. As a popular mobile MOBA game, Honor of Kings has a DAU of over 100 million, and has become an official sport in the 2022 Hangzhou Asian Games. In MOBA games, several players are divided into two teams and carry out a battle against each other. AI reseach in MOBA games has received many attention (Ye et al., 2020, 2022). Real-time commentary plays an important role in live streaming of e-sports competitions, assists the audience in the comprehension of game situation and events, expanding the presence of MOBA games.

In-battle commentary aims at generating coherent and informative natural language descriptions based on game situation, match statistics and player behaviour, as illustrated in Figure 1. Similar systems have already been widely applied in real-world applications such as automated news broadcast, television sports commentary, online video description, etc (Chen and Mooney, 2008; Wiseman et al., 2017; Nie et al., 2018; Puduppully et al., 2019; Kale and Rastogi, 2020). In this paper, we focus on the scenario of live e-sports commentary

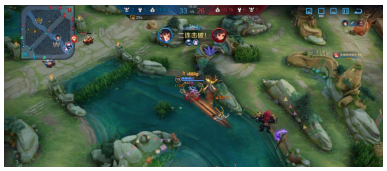

Hero	Kills	Deaths	Assists	···	Gold	Gold (variation)
Shangxiang Sun	12	4	7	···	14028	16.23/second
Que Bian	3	2	20	···	13428	19.53/second
···	···	···	···	···	···	···
Ying Yun	8	6	8	···	15186	3.73/second
Yuanfang Li	9	7	8	···	12186	3.73/second
···	···	···	···	···	···	···

Killer hero: Shangxiang Sun
Be killed hero: Yuanfang Li
Event: double kill

Commentary:
孙尚香拿下双杀，厉害！
Shangxiang Sun accomplished double kill, awesome!

Figure 1: The in-battle commentary generation task.

in mobile MOBA games and introduce MCS: an in-battle commentary generation system [1][2].

Automatic commentary is grounded in in-battle match statistics and events, regarded as a data-to-text generation task. Recently, many large-scale language models have been proposed and promoted the advance in natural language generation (Radford et al., 2018; Devlin et al., 2019; Radford et al., 2019). These pre-trained language models are also beneficial for data-to-text tasks (Kale and Rastogi, 2020), in this work, we employ OpenAI's GPT-2 model as the basic generator, and build a neural pipelined approach for commentary generation.

Although recent works have shown promising results regarding live comment generation (Chen and Mooney, 2008; Wiseman et al., 2017; Nikolova et al., 2017; Dušek et al., 2018; Ma et al., 2019; Zhang et al., 2020), there are still many challenges for constructing a commentary generation model for MOBA games: (1) Previous studies incorporated static data like final scores, while live e-sports commentary uses changing match statistics as the game goes on. Current statistics as well as history statistics are both vital for text generation. For example, the narrowing of advantages may cause anxiety, while overturn from deficit can be

[1]This work is licensed under a Creative Commons Attribution 4.0 International License. License details: http://creativecommons.org/licenses/by/4.0/.

[2]Demonstration video: https://youtu.be/G0lKZKd7eco.

Proceedings of the 29th International Conference on Computational Linguistics, pages 2962–2967
October 12–17, 2022.

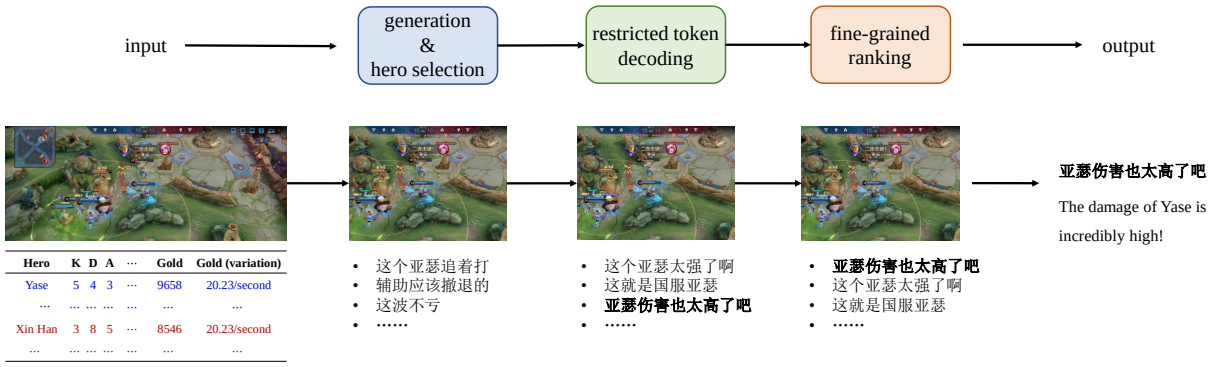

Figure 2: Pipeline of the proposed MCS, including three stages: initial commentary generation, restricted token decoding and fine-grained ranking.

a morale booster. Therefore, the problem of commentary generation employing dynamic structured data needs to be tackled. (2) Neural network based text generators are claimed to be unreliable in the consistency and accuracy of generated texts. However, commentary generation is highly correlated with in-battle data, which requires high conformity in the generated sentence and input data. In many approaches, retrieval methods are used instead of generative methods. How to solve the problem of generative methods and improve reliability remains challenging.

The proposed MCS has the following advantages: (1) We apply GPT-2 model into data-to-text task, and design a hero selection loss to focus on the target hero behaviour for commentary. (2) Previous studies on sports commentary generation incorporated only static data like final scores, while the proposed MCS uses dynamic match statistics during generation, as current and history statistics are both vital for e-sports commentary. (3) Commentary is highly correlated with in-battle data, which requires high conformity in the generated sentence and input data. A restricted token decoding method is employed to improve generation reliability.

2 System Framework

We first introduce the general framework of the proposed MCS system. The system consists of three modules in a pipeline, as presented in Figure 2. Taking in-battle statistics as input, the system produces a descriptive sentence based on current and history data.

2.1 Initial commentary generation

The first module focuses on generating initial texts: in this stage, several candidate sentences are generated in an end-to-end manner, used as inputs for further polishing. All statistics of the ten heros as well as critical events are included in network input. The critical event is defined by a triplet including event type, active hero, and passive hero. For instance, single killing is a typical event that occurs frequently in MOBA games, in which the active hero is the hero kills someone, and the passive hero is the killed one. Figure 3 shows two examples of events. Beside all hero statistics, critical events are also included in network input as most commentaries focus on the heros involved in these events.

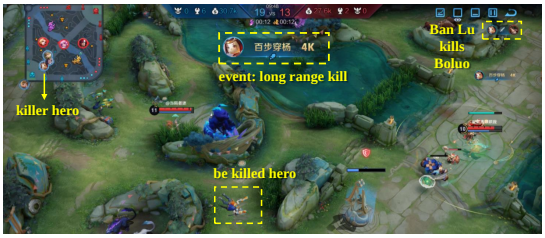

(a) Long range kill by hero Ban Lu.

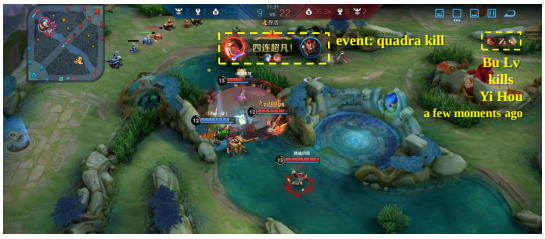

(b) Bu Lv quadra kill.

Figure 3: Examples of events. (a) A critical event of long range kill, active hero: Ban Lu, passive hero: Boluo; (b) A critical event of quadra kill, active hero: Bu Lv, passive hero: Wei Dian.

2.2 Restricted token decoding

The second module performs restricted decoding based on the initial text generator. Restricted generation can increase the reliability of neural lan-

(a) Data panel in game.

Data	Description
Hero ID	Unique identifier of the hero
Kills	Number of kills of the hero
Kills (variation)	The variation of kills in the last 30 seconds
Deaths	Number of deaths of the hero
Deaths (variation)	The variation of deaths in the last 30 seconds
Assists	Number of assists of the hero
Assists (variation)	The variation of assists in the last 30 seconds
Gold	In-game virtual currency income of the hero
Gold (variation)	The variation of Gold in the last 30 seconds

(b) Data and description.

Figure 4: Match statistics in game. Left: the kills/deaths/assists and other attributes are displayed on the data panel. Right: the data used as system input and description.

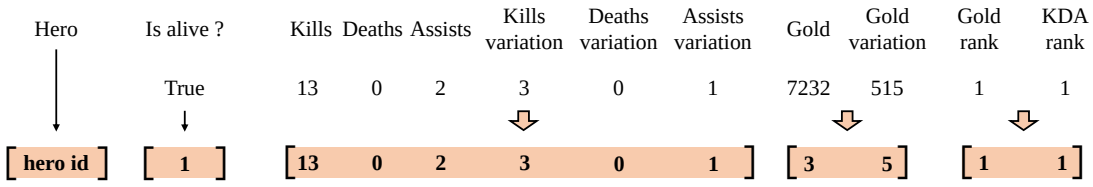

Figure 5: Serialization of structured data of a hero.

guage generation (Elder et al., 2020; Dathathri et al., 2020). Inspired by retrieval text generation, we construct event-specific corpus from the collected dataset, which is used in the token decoding process. With this approach employed, the generated sentences are expected to be controlled within a reasonable range around the corresponding event.

2.3 Fine-grained ranking

The final module conducts fine-grained ranking on sentences generated through the previous two modules. A pairwise scoring model is constructed following textual matching methods, to avoid topic shift and destruction of text fluency, the scoring model is trained against topic irrelevant commentaries and faulty sentences. After ranking, the commentary achieving the highest score serves as final output of the MCS system.

3 System Implementation

In this section, the implementation details of the proposed three modules are provided. In general, the network architecture are constructed following the pipeline of language models. As described in Figure 4, features are serialized and fed as the inputs, the network predicts a target hero ID and generates a comment as the outputs.

3.1 Initial commentary generation

For the initial commentary generator, the network input is composed of critical events and match statistics of the ten heros. A triplet including event type and two hero IDs is used to represent the event, as described in Section 2.1. For match statistics, the data entries used in the proposed MCS includes Hero ID, Kills, Deaths, Assists, etc, as presented in Figure 4. Since the statistics are changing over time and the changes are important in indicating hero status and highlight events, we also include variations of the above data entries in network input. For example, a hero with high kills and gold usually means that it can cause high damage in battle. If the kills of a hero rise rapidly in a short time, it is likely that the hero has outstanding performance and human commentators will comment on it.

To adapt the structured data to a sequential input of the network, we design a method to transform hero attributes into decimal digits. For each hero, a unique identifier is assigned and placed at the beginning of its data sequence, followed by a token indicating whether the hero is alive or not. The other indicators are discretized by dividing into segments, Figure 5 illustrates an example of data serialization. Kills, Deaths, Assists and the variations are encoded by 600 digits, 100 for each, as they are usually less than 100 in a single match. In extreme cases where the indicators exceed 100, the real digits are replaced by 100 to avoid confusion. In a single match, the Gold of each hero usually varies from 0 to 40000, it is discretized by dividing into 20 segments, 2000 golds for one segment. The variation of Gold is encoded in a similar way, each segment represents 90 golds per minute, and the maximum variation is set to 4500 golds per minute.

At the end of the sequence, we use two extra tokens to represent the ranking of Gold and KDA of the correspoding hero among all 10 heros. The data sequence of all 10 heros are connected by a special token "[SEP]", and another special token "[EOD]" is added after the last data block to separate text from data sequence.

In general, the network architecture is illustrated in Figure 6. Previous studies have proven that pre-trained language models can improve data-to-text generation (Peng et al., 2020; Chen et al., 2020; Kale and Rastogi, 2020), we use Open-AI's GPT-2 model to build the basic text generator. The GPT-2 model used in this work was pre-trained using Chinese copora LCCC (Large-scale Cleaned Chinese Conversation (Wang et al., 2020). In addition to standard language model loss, we design a hero selection loss to focus on the match data of the target hero, which is helpful to increase generation consistency.

3.2 Restricted token decoding

In this stage, the collected dataset is divided into event-specific groups by a triplet retrieval key. The retrieval key is composed of event type, active hero ID, and passive hero ID. During generation, the triplet retrieval key can be extracted from in-battle statistics, and the corresponding event-specific corpus can be retrieved. In retrieval text generation approaches, a selection strategy is designed to choose a proper sentence from the corpus as the final result. In the proposed MCS system, we employ the event-specific corpus to restrict the sentences generated by neural networks.

The Plug and Play Language Model (PPLM) (Dathathri et al., 2020) module is employed to control the token decoding process, restricting the generated texts around the corresponding event. Basically, the restriction is accomplished by updating the last hidden layer features h_t:

$$h_t \leftarrow h_t - \nabla \sum_{i=1}^{|corpus|} log(p(w_i|h_t)) \qquad (1)$$

where $|corpus|$ represents the size of a event-specific corpus and w_i denotes a word in the corpus. After a few iterations of updating, the hidden features are softly adjusted towards decoding the words in the event-specific corpus.

Table 1: Experimental results, best results are indicated in bold. The upper two rows are results of existing methods, the bottom rows are results of our method.

	ROUGE-1	ROUGE-2	ROUGE-L	Validity
Seq2Seq (Sutskever et al., 2014)	0.1306	0.0032	0.1296	0.69
GPT-2 (Radford et al., 2019)	0.1568	0.0102	0.1491	0.85
Initial commentary generator	0.1921	0.0111	0.1903	-
Restricted token decoding	0.2025	0.0140	0.2001	-
Fine-grained ranking	**0.2028**	**0.0144**	**0.2006**	**0.91**

3.3 Fine-grained ranking

The final stage performs a ranking on the generated candidate sentences. We build a pairwise ranking network on top of the GPT-2 generator. Taking two sentences as input, the GPT-2 outputs two encoding vectors respectively. These encodings are then fed into the ranking network, and a score is derived which indicates the relative order of the two sentences. After all candidate sentences are processed, the best sentence is regarded as the final commentary.

4 Materials and Results

The structured data and commentary history of 30 million matches in professional leagues and live broadcasts of Honor of Kings were collected and used to train and evaluate the proposed system. Perplexity is used to evaluate the fluency of generated sentences, ROUGE-1, ROUGE-2 and ROUGE-L are used to measure the distance between generated sentences and ground truth. We also calculate the generation validity by human evaluation.

The experimental results are presented in Table 1. Quantitatively, the system achieves an averaged perplexity of 1.27. Comparing the generated sentences with those written by human commentators among the collected data, the ROUGE-1, ROUGE-2 and ROUGE-L scores are 0.1921, 0.0111 and 0.1903 for the initial commentary generation, respectively. Combined with target hero prediction and restricted token decoding, the metrics improved to 0.2025, 0.0140 and 0.2001. The final ROUGE scores are 0.2028, 0.0144 and 0.2006 employing fine-grained ranking. Besides statistical indicators, we also carried out human evaluation of the system, and the generated commentaries achieve a validity of 0.91. This means that the system has the potential to be used in real-world applications. Our system is compared with two mainstream generation approaches: the RNN-based Seq2Seq (Sutskever et al., 2014) and the transformer-based standard GPT-2 (Radford et al., 2019), and outperforms both of them.

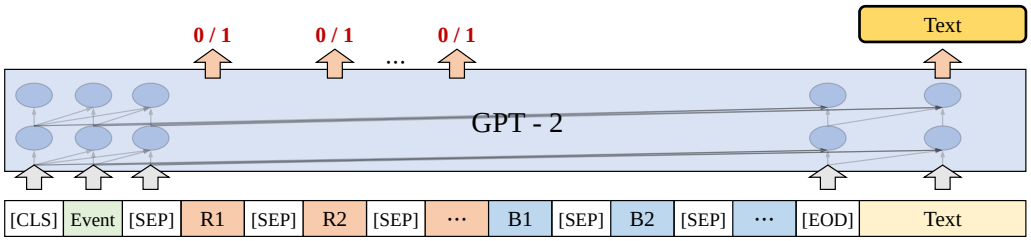

Figure 6: Architecture of the generation network. The network is based on GPT-2 model. The beginning of network outputs accomplish target hero selection, the remaining represents the generated tokens. "Event" denotes the triplet of critical event, "R1" - "R5" represents the data of the 5 heros on the red side, "B1" - "B5" represents the data of the 5 heros on the blue side.

Figure 7 shows two cases of the generated commentaries.

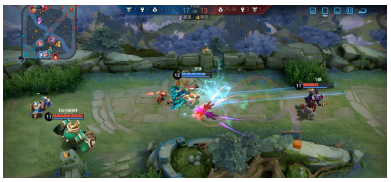

Commentary:
这个大可惜呀。
The ULT misses, what a great pity!

(a) The ULT of Ganjiang Moye misses, a few moments later, he is killed by Mozi.

Commentary:
这是真的太细节。
The player's control is full of precise details.

(b) Anqila draws first blood by killing Master Fu.

Figure 7: The examples of commentaries generated by the proposed system.

5 Conclusion

We propose MCS, a model for in-battle real-time commentary generation in MOBA games. MOBA is a type of games popular all over the world, commentary plays an important role in game comprehension for audience. The proposed MCS is designed with the ability of analyzing real-time match statistics and generating coherent descriptive contents. It has a wide range of applications such as e-sports commentary, combat information analysis, etc, and has the potential to promote game development and improve match environment.

References

David L. Chen and Raymond J. Mooney. 2008. Learning to sportscast: A test of grounded language acquisition. In *Proceedings of the 25th International Conference on Machine Learning*, page 128–135. Association for Computing Machinery.

Zhiyu Chen, Harini Eavani, Wenhu Chen, Yinyin Liu, and William Yang Wang. 2020. Few-shot NLG with pre-trained language model. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 183–190, Online. Association for Computational Linguistics.

Sumanth Dathathri, Andrea Madotto, Janice Lan, Jane Hung, Eric Frank, Piero Molino, Jason Yosinski, and Rosanne Liu. 2020. Plug and play language models: A simple approach to controlled text generation. In *International Conference on Learning Representations*.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. Bert: Pre-training of deep bidirectional transformers for language understanding. In *NAACL-HLT*, pages 4171–4186.

Ondřej Dušek, Jekaterina Novikova, and Verena Rieser. 2018. Findings of the e2e nlg challenge. In *Proceedings of the 11th International Conference on Natural Language Generation*, pages 322–328.

Henry Elder, Alexander O'Connor, and Jennifer Foster. 2020. How to make neural natural language generation as reliable as templates in task-oriented dialogue. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 2877–2888, Online. Association for Computational Linguistics.

Mihir Kale and Abhinav Rastogi. 2020. Text-to-text pre-training for data-to-text tasks. In *Proceedings of the 13th International Conference on Natural Language Generation*, pages 97–102. Association for Computational Linguistics.

Shuming Ma, Lei Cui, Damai Dai, Furu Wei, and Xu Sun. 2019. Livebot: Generating live video comments based on visual and textual contexts. *Proceedings of the AAAI Conference on Artificial Intelligence*, 33(01):6810–6817.

Feng Nie, Jinpeng Wang, Jin-Ge Yao, Rong Pan, and Chin-Yew Lin. 2018. Operation-guided neural networks for high fidelity data-to-text generation. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing, Brussels, Belgium, October 31 - November 4, 2018*, pages

3879–3889. Association for Computational Linguistics.

Jekaterina Novikova, Ondřej Dušek, and Verena Rieser. 2017. The e2e dataset: New challenges for end-to-end generation. In *Proceedings of the 18th Annual SIGdial Meeting on Discourse and Dialogue*, pages 201–206.

Baolin Peng, Chenguang Zhu, Chunyuan Li, Xiujun Li, Jinchao Li, Michael Zeng, and Jianfeng Gao. 2020. Few-shot natural language generation for task-oriented dialog. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 172–182, Online. Association for Computational Linguistics.

Ratish Puduppully, Li Dong, and Mirella Lapata. 2019. Data-to-text generation with content selection and planning. In *The Thirty-Third AAAI Conference on Artificial Intelligence, AAAI 2019*, pages 6908–6915. AAAI Press.

Alec Radford, Karthik Narasimhan, Tim Salimans, and Ilya Sutskever. 2018. Improving language understanding by generative pre-training. *OpenAI Blog*.

Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, Ilya Sutskever, et al. 2019. Language models are unsupervised multitask learners. *OpenAI Blog*.

Ilya Sutskever, Oriol Vinyals, and Quoc V Le. 2014. Sequence to sequence learning with neural networks. *Advances in neural information processing systems*, 27.

Yida Wang, Pei Ke, Yinhe Zheng, Kaili Huang, Yong Jiang, Xiaoyan Zhu, and Minlie Huang. 2020. A large-scale chinese short-text conversation dataset. In *NLPCC*.

Sam Wiseman, Stuart Shieber, and Alexander Rush. 2017. Challenges in data-to-document generation. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 2253–2263. Association for Computational Linguistics.

Deheng Ye, Guibin Chen, Wen Zhang, Sheng Chen, Bo Yuan, Bo Liu, Jia Chen, Zhao Liu, Fuhao Qiu, Hongsheng Yu, Yinyuting Yin, Bei Shi, Liang Wang, Tengfei Shi, Qiang Fu, Wei Yang, Lanxiao Huang, and Wei Liu. 2020. Towards playing full moba games with deep reinforcement learning. In *NeurIPS*.

Deheng Ye, Guibin Chen, Peilin Zhao, Fuhao Qiu, Bo Yuan, Wen Zhang, Sheng Chen, Mingfei Sun, Xiaoqian Li, Siqin Li, Jing Liang, Zhenjie Lian, Bei Shi, Liang Wang, Tengfei Shi, Qiang Fu, Wei Yang, and Lanxiao Huang. 2022. Supervised learning achieves human-level performance in moba games: A case study of honor of kings. *IEEE Transactions on Neural Networks and Learning Systems*, 33(3):908–918.

Shengyu Zhang, Ziqi Tan, Zhou Zhao, Jin Yu, Kun Kuang, Tan Jiang, Jingren Zhou, Hongxia Yang, and Fei Wu. 2020. Comprehensive information integration modeling framework for video titling. In *Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, page 2744–2754, New York, NY, USA. Association for Computing Machinery.

A Two Stage Adaptation Framework for Frame Detection via Prompt Learning

Xinyi Mou[1], **Zhongyu Wei**[1,2]*, **Changjian Jiang**[3], **Jiajie Peng**[4]
[1]School of Data Science, Fudan University, China
[2]Research Institute of Intelligent and Complex Systems, Fudan University, China
[3]School of International Relations & Public Affairs, Fudan University, China
[4]School of Computer Science, Northwestern Polytechnical University, China
{xymou20,zywei,Changjian}@fudan.edu.cn
jiajiepeng@nwpu.edu.cn

Abstract

Framing is a communication strategy to bias discussion by selecting and emphasizing. Frame detection aims to automatically analyze framing strategy. Previous works on frame detection mainly focus on a single scenario or issue, ignoring the special characteristics of frame detection that new events emerge continuously and policy agenda changes dynamically. To better deal with various context and frame typologies across different issues, we propose a two-stage adaptation framework. In the framing domain adaptation from pre-training stage, we design two tasks based on pivots and prompts to learn a transferable encoder, verbalizer, and prompts. In the downstream scenario generalization stage, the transferable components are applied to new issues and label sets. Experiment results demonstrate the effectiveness of our framework in different scenarios. Also, it shows superiority both in full-resource and low-resource conditions.

1 Introduction

Framing is a communication strategy, used to bias the discussion toward a specific stance, by selecting particular aspects of reality and making them more salient (Entman, 1993; Liu et al., 2019a). It is widely adopted by politicians, the media, and the voting public to seek support, express opinions, and advance political agendas (Levendusky, 2013), thus having important implications for public opinion understanding and policy decision-making (Mendelsohn et al., 2021). Boydstun et al. propose 15 generic frame dimensions based on policy agenda, including *economic*, *morality*, and so on, paving the way for frame analysis. In order to analyze the framing strategy automatically, researchers explore the task of frame detection.

Given a piece of statement related to a topic, frame detection aims to recognize which dimensions of frames are employed. Formally, it's a multi-label classification task with a pre-defined label set. Existing researches for automatic frame detection explore different supervised methods including feature-based machine learning, deep neural networks, and fine-tuning pre-trained models (Card et al., 2016; Naderi and Hirst, 2017; Liu et al., 2019b). Despite the success and contribution made by previous studies, these methods ignore the special characteristics of frame detection that events develop quickly and political focus of agendas changes frequently.

Issue	Source	# label	Example	Frame
abortion	tweets	14	Women have the **freedom** to make their own **choices** about their reproductive health care. #AbortionLaw protects the **safety** of women.	Fairness & Equality Health & Safety
gun	news	9	Florida shooter a **troubled loner** with **white supremacist ties**.	Mental Health Race/Ethnicity
gun	debates	5	The **2nd amend** does not say people need guns for self defense !	Legality, Constitutionality and Jurisprudence

Figure 1: A brief landscape of frame detection. These three samples are taken from **twitter** (Johnson et al., 2017), **gvfc** (Liu et al., 2019a), **fora** (Hartmann et al., 2019) respectively.

We demonstrate the landscape of frame detection in Figure 1 with examples picked from three existing datasets and three challenges stand out. (1) Dynamic nature of languages: framing is used in complex scenarios of quite different styles of language expressions, i.e., coming from different sources and discussing various issues. Previous methods didn't consider learning from existing issues, thus once new issues emerge, data annotation and model training must be repeated. (2) Diverse categories of frame dimensions: frames are defined from different perspectives that can be general like *fairness & equality* or issue-specific like *mental health*. (3) Variance of frames across issues: la-

*Corresponding author.

Proceedings of the 29th International Conference on Computational Linguistics, pages 2968–2978
October 12–17, 2022.

bel set of frames changes for different issues. The complex label system means frame detection can not be easily transferred, unlike traditional tasks including sentiment analysis and stance detection that shares a common label set. In order to address these three challenges and mitigate the limitations of existing research, we propose to develop a generalized framework for frame detection that obtains robust language modeling capability, possesses background knowledge of framing strategy, and can be adapted to new issues easily. Recently, progress in prompt learning provides a mechanism to exploit pre-trained models by task-specific cloze or prefix prompts. Previous research shows its effectiveness in few-shot and zero-shot learning (Schick and Schütze, 2020b; Liu et al., 2021a). Motivated by the success of prompt learning on low-resource sentence classification, we aim to design a prompting framework for frame detection.

In this paper, we propose a two-stage adaptation framework for frame detection based on prompt learning that optimizes prompts and a pre-trained language model for the general domain of framing and applies it to new scenarios. In the framing domain adaptation from pre-training stage, we train a pivot-based encoder on top of the pre-trained language model based on a generic corpus with a general framing label set. Then, we design a shared prompt and several issue-specific prompts and learn their parameters by prompt learning and adversarial training to acquire transferable prompts and a verbalizer. In the downstream scenario generalization stage, we adopt the transferable components to new issues and scenarios. Our contributions are mainly three-fold:

- We propose a generalized prompting framework for frame detection that can deal with different scenarios, issues, and typologies, bridging the gap of cross-issue generalization missing in previous work.
- Different from previous transfer learning, our framework does not train the target data with the source data together, but only reuses the parameters, ensuring data security in the real environment. It is more flexible since data related to politics can be confidential and sensitive in some countries.
- We present the largest study of frame detection, covering 5 datasets from 3 different scenarios. Experiment results show the effectiveness of our framework in both full-source and low-resource situations.

2 Framing Domain Adaptation from Pre-training

Given a sentence x and a set of frame labels $\mathcal{F}$, we aim to detect which frames are used. It is a multi-label classification problem. Since we hope the model can deal with various issues and label sets, it's necessary to urge the model to learn general and transferable knowledge related to frames. In this stage, namely framing domain adaptation from pre-training, we design two tasks to achieve this goal. A generic corpus $\mathcal{C}$ is used as an anchor corpus for the training in this stage.

2.1 Transferable Pivot-based Encoder

Primarily, we design a masked-pivot prediction task (Task1 in Figure 2) to learn a transferable encoder, to capture issue-unrelated features of frames. The process is divided into two steps.

Frame Pivots Generation According to Field et al., some indicators are informative for frame detection, e.g., *cost*, *wage*, *economy* for *economic* frame. We believe these indicators are similar to transferable sentiment words in sentiment classification, which can serve as pivots. We generate frame pivots by mutual information (Church and Hanks, 1990). For a given frame F in $\mathcal{C}$, we calculate point-wise mutual information (PMI) for each word by:

$$I(F, w) = \log \frac{P(F, w)}{P(F)P(w)} = \log \frac{P(w \mid F)}{P(w)} \quad (1)$$

where $P(w \mid F)$ is estimated as $\frac{Count(w)}{Count(allwords)}$ by taking all texts annotated with F, while $P(w)$ is similarly computed using the entire corpus. Words that occur in fewer than 0.5% or more than 98% of documents will be discarded. Finally, we reserve top K words with highest PMI score for each frame, where K is a hyper-parameter.

MLM Training We optimize an encoder on top of a pre-trained language model based on modeling the relationship between pivots and non-pivots following (Li et al., 2020; Ben-David et al., 2020). Concretely, we employ a pre-trained BERT (Devlin et al., 2019), optimized by an MLM objective. As shown in Task1 in Figure 2, given a document, we randomly mask pivots with probability p_v and mask non-pivots with probability p_n, instead of

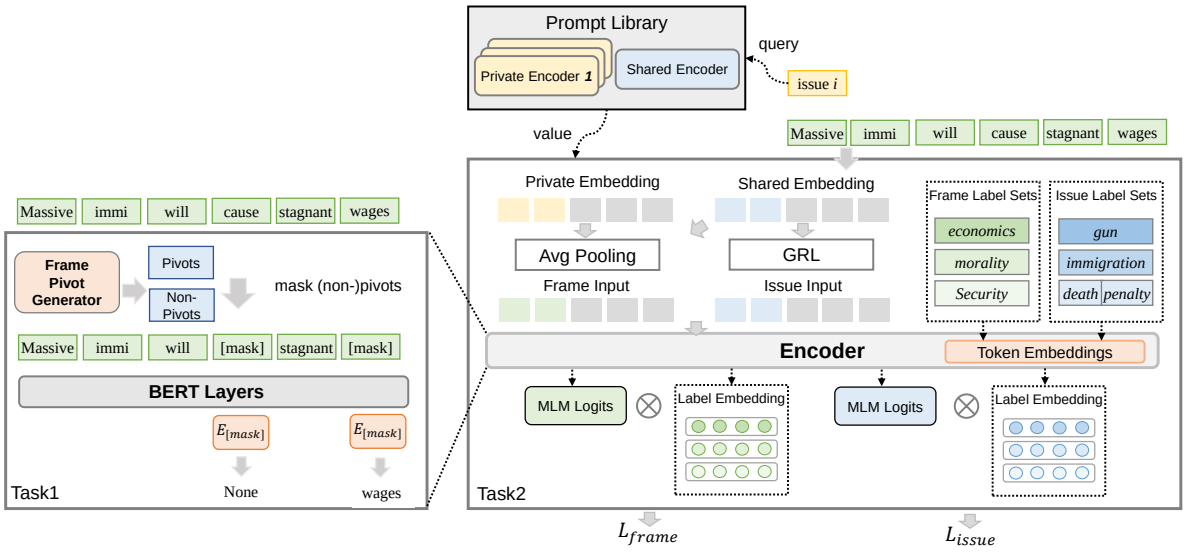

Figure 2: The framework architecture in framing domain adaptation from pre-training stage. In masked-pivot prediction task(left), we train a transferable encoder. In prompt learning-based frame detection task(right), we further optimize the prompts, encoder and verbalizer.

masking each token with the same probability. Like what is done in BERT, for the chosen (non-)pivot words, most of the time they are changed to the mask token, with a small probability of being unchanged. Besides, we don't predict the token on the entire vocabulary size, but only focus on whether the masked token is a pivot or not and which pivot it is. Therefore, it's a classification task with $p+1$ classes, where p is the number of unique pivots. Then, masked cross-entropy is used to calculate the loss for masked token prediction.

$$\mathcal{L}_{MLM} = \frac{1}{\sum m_i} \sum_{i=1}^{n} m_i \cdot L\left(\hat{\mathbf{y}}_i^p, \mathbf{y}_i^p\right) \quad (2)$$

where $m_i \in \{0, 1\}$ indicates whether the token is masked, and $L\left(\hat{\mathbf{y}}_i^p, \mathbf{y}_i^p\right)$ denotes the cross entropy loss for pivot prediction.

2.2 Transferable Prompts and Verbalizer

To pre-train transferable prompts and verbalizer, we adopt the second task, prompt learning-based frame detection (Task2 in Figure 2). Each component will be described in detail as follows.

Prompt Template To retrieve the knowledge encoded in the pre-trained language model, we design a template to wrap the original input text into a prompt, by discrete natural language or continuous parameters (Liu et al., 2021a,c). To avoid troublesome prompt engineering, we use continuous prompt embeddings. Inspired by Wang et al., for prompt configuration, we build a private prompt template for each issue, and a shared prompt template for all the issues, constituting the prompt library in Figure 2. For each input text x of issue i, the issue-specific template and the shared template will respectively convert the input as:

$$t^{(i)}(x) = w_1^{(i)}, ..., w_m^{(i)}, x, w_{m+1}^{(i)}, ..., w_M^{(i)}, w_{[mask]} \quad (3)$$

$$t^{s}(x) = w_1^{s}, ..., w_m^{s}, x, w_{m+1}^{s}, ..., w_M^{s}, w_{[mask]} \quad (4)$$

where w_m is prompt pseudo token (Liu et al., 2021c), M is the number of prompt pseudo tokens. A bidirectional LSTM (Hochreiter and Schmidhuber, 1997) with multi-layer perceptrons are applied as prompt encoders. We can get prompt embedding by:

$$PE^{(i)}(x) = MLP(BiLSTM(t^{(i)}(x))) \quad (5)$$

and the shared template will get:

$$PE^{s}(x) = MLP(BiLSTM(t^{s}(x))) \quad (6)$$

Then, the average of both prompts will result in the final prompt embedding, which is a sequence as the input of the encoder for frame detection, as shown in Figure 2.

Verbalizer Learning for Frames In prompt learning, a mapping from words to labels, e.g.,

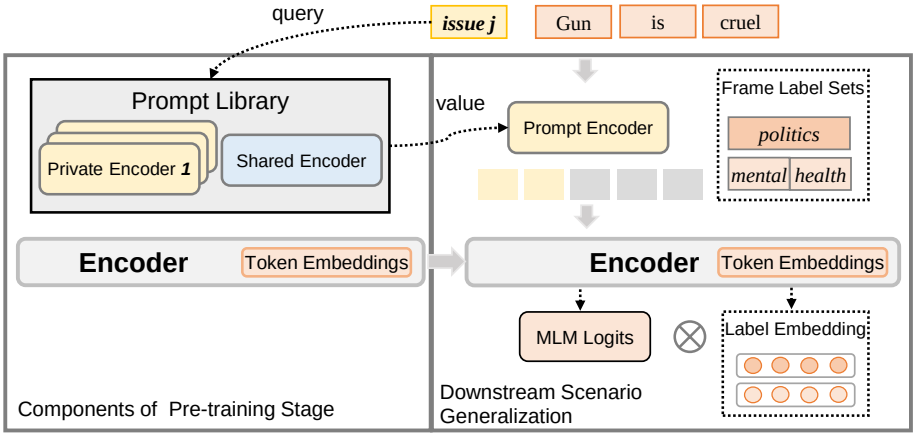

Figure 3: Downstream Scenario Generalization Stage: reuse the transferable components acquired in the framing domain adaptation from pre-training stage.

good, *great* for label *positive*, which is also called verbalizer, is needed for final prediction. However, manual label word selection needs domain-expert knowledge and is defective in dealing with answers of different lengths. Moreover, it's difficult to express the meaning of a frame label by single words, e.g., neither word *school* nor *safety* is a proper answer to a cloze problem about *school safety*. To overcome this problem, we attempt to use the average of the token embeddings to get label embedding. One of the biggest advantages of doing this is the implicit flexibility and transferability of label embedding. For example, once we learned token embedding for label *health* in the pre-training stage, it's beneficial to better understand label *mental health* in a new dataset, since they share the common token *health*. Specifically, we tokenize the original label, extract corresponding token embeddings and take the average as the final label representation:

$$E_l = \frac{1}{n_l}\sum_{i=1}^{n_l} TE(token_i) \tag{7}$$

where n_l is the number of tokens of label l, and $TE(token_i)$ represents token embedding of $token_i$. Then, we calculate the dot product of MLM output with each label embedding. Finally, a binary-cross entropy loss (BCE) is applied for each label. Thus, loss for frame detection is:

$$\mathcal{L}_{frame} = -\frac{1}{N}\sum_{j}^{N}\sum_{k}^{F} y_j^f(k)\log \hat{y}_j^f(k) \tag{8}$$

where $y_j^f(k) \in \{0,1\}$, N is the number of samples, and F is the number of unique frame labels.

Adversarial Training for Prompts In order to learn a shared prompt that can be easily transferred to new issues, we hope the shared prompt pays attention to cross-issue features related to frames, rather than any specific issue or topic content. To achieve this goal, we design an issue adversarial task to make the shared prompt cannot distinguish issues. Concretely, as shown in Figure 2, we use the shared template only to wrap each input text, and we input this issue prompt to the encoder to predict the corresponding issue. Here, a Gradient Reversal Layer (GRL) (Ganin and Lempitsky, 2015) is involved, before the input is sent to the encoder. During the backpropagation, it reverses the gradient by multiplying a negative scalar. For this single-label prediction task, the target is to minimize the cross-entropy loss:

$$\mathcal{L}_{issue} = -\frac{1}{N}\sum_{j}^{N}\sum_{k}^{I} y_j^i(k)\log \hat{y}_j^i(k) \tag{9}$$

where where $y_j^i(k) \in \{0,1\}$, N is the number of samples, and I is the number of unique issues in the generic corpus $\mathcal{C}$.

In this case, we train the model using following loss:

$$\mathcal{L} = \mathcal{L}_{frame} + \lambda\mathcal{L}_{issue} \tag{10}$$

where λ is a hyper-parameter controlling the weight of different losses.

3 Downstream Scenario Generalization

After the pre-training stage, we get a set of issue-specific prompt encoders, a shared prompt encoder, and an encoder that can be adapted to downstream scenarios. In the generalization stage, we reuse these transferable components for frame detection in new scenarios. As shown in Figure 3, for issues that have been "seen" in the generic corpus $\mathcal{C}$, we initialize the prompt encoder by the corresponding private prompt encoder in the prompt library, and for those unseen issues, the shared prompt encoder is applied for generalization. And encoder along with its token embeddings (used for verbalizer) in the previous stage is also adopted here.

4 Datasets

Based on review of previous work on frame detection, we get datasets available mainly from three resources: (1) news media (articles), (2) social media, (3) debates and statements. Five datasets are listed.

mfc News articles of 15 general frame annotation, constructed by Card et al.. The last version covers 6 issues-*climate change*, *death penalty*, *gun control*, *immigration*, *same-sex marriage* and *tobacco*, with 5,199 articles for each issue on average.

gvfc (Liu et al., 2019a) 1,300 headlines of news articles on *gun violence*, annotated with 9 issue-specific frames.

twitter (Johnson et al., 2017) 2,050 tweets posted by 40 politicians, annotated with 17 general frames (among which 14 are the same with **mfc**, and 3 designed specifically for tweets). 5 issues are included-*abortion*, *aca*, *immigration*, *isis* and *lgbt*. We also treat each as an independent dataset, to verify effectiveness of our framework on unseen new issues like *abortion*.

immi (Mendelsohn et al., 2021) tweets about *immigration*, published by the public, among which 2,325 are annotated with general frames and 1,375 are annotated with issue-specific frames.

fora (Hartmann et al., 2019) 868 arguments on online discussion fora, annotated with 5 most frequent general frames, covering 4 topics.

Following Johnson et al., we drop *other* category in **mfc** and **immi**, and the overview of the datasets is shown in Table 1. Considering that **mfc** is the largest existing corpus with the generic setting of frames, it is used in the framing domain adaptation from pre-training stage. Note that, any other corpus can also be used as the anchor in our framework.

Datasets	Typology	#Label	Size
mfc	general	14	31,960
gvfc	issue-specific	9	1,300
twitter	general	17	2,050
immi	general	14	2,325
	issue-specific	11	1,375
fora	general	5	868

Table 1: Statistics of listed datasets.

5 Experiment and Results

5.1 Experiment Setup

Models for Comparison. We compare our model with some state-of-art methods for frame detection as well as some prompt learning and transfer learning methods.

- Bi-LSTM (Naderi and Hirst, 2017) with initialization by GloVe word embeddings (Pennington et al., 2014).
- Bi-GRU (Naderi and Hirst, 2017) with initialization by GloVe word embeddings.
- FT, fine-tune a pre-trained language model (PLM). (Mendelsohn et al., 2021)
- MP, use a manually-designed prompt and verbalizer to implement prompt learning on specific datasets. We use manual prompt $T(X) = X.\ It\ emphasizes\ [MASK].$, according to definition of framing.
- P-tuning, use vanilla ptuning (Liu et al., 2021c) prompt to implement prompt learning on specific datasets and use label embedding in our framework as verbalizer.
- P-tuning v2 (Liu et al., 2021b), adds prefix into each layer of the PLM and use a fully-connected layer for classification.
- FT-mtl (Sun et al., 2019), fine-tunes PLM by multi-task learning, where frame detection on **mfc** dataset serves as the auxiliary task.
- FT-adv (Hu et al., 2019), on the basis of FT-mtl, it fine-tunes PLM with an additional adversarial domain loss, where domains are issues in **mfc** dataset.

Method	gvfc		twitter		immi-general		immi-specific		fora	
	MiF	MaF	MiF	MaF	MiF	MaF	MiF	MaF	MiF	MaF
Bi-LSTM	66.52	59.92	49.26	38.00	46.22	39.84	33.13	21.67	69.87	68.17
Bi-GRU	71.49	60.58	46.25	32.82	42.72	35.98	33.65	19.39	75.29	74.00
FT	83.09	77.29	65.37	56.98	74.38	67.51	60.19	45.71	80.94	81.65
MP	82.38	74.80	62.73	50.27	70.58	60.01	58.11	41.36	78.13	77.89
P-tuning	84.17	77.09	63.38	55.45	74.06	67.23	60.32	46.20	78.64	78.94
P-tuning v2	82.87	76.12	62.31	51.55	71.52	63.87	60.21	**51.41**	78.07	78.63
FT-mtl	84.51	73.02	62.02	52.75	71.54	60.05	62.58	44.84	78.53	79.41
FT-adv	81.50	72.20	59.91	42.36	71.89	63.89	60.94	39.01	75.82	76.22
FT-meta	81.69	75.03	62.25	51.14	68.58	61.22	61.24	46.87	76.43	75.77
Ours	**85.25**	**80.36**	**66.39**	**60.56**	**75.07**	**68.15**	**62.59**	50.66	**81.26**	**82.61**

Table 2: Results in full-resource experiments across datasets. (**Bold**: the best performance in the column)

Method	aca		abortion		isis	
	MiF	MaF	MiF	MaF	MiF	MaF
Bi-LSTM	28.33	8.01	41.60	9.52	38.85	17.46
Bi-GRU	35.55	15.92	36.97	17.89	55.01	25.41
FT	55.29	**30.06**	42.84	15.04	53.91	26.96
MP	54.88	24.74	46.57	13.72	57.16	20.15
P-tuning	54.68	28.18	47.36	17.70	60.32	33.55
P-tuning v2	56.34	26.37	44.44	14.32	**68.04**	29.21
FT-mtl	37.23	14.63	34.28	7.22	67.41	25.11
FT-adv	40.61	13.55	34.89	8.91	64.58	23.66
FT-meta	39.92	13.57	35.71	10.09	64.41	24.61
Ours	**56.72**	29.18	**53.03**	**19.74**	65.19	**36.04**

Table 3: Results in full-resource training on unseen issues in **twitter** dataset. (**Bold**: the best performance in the column)

- FT-meta (Wang et al., 2020), where typical instances of each issues in **mfc** dataset are used to train a meta learner, and the meta learner is further fine-tuned by specific new datasets.

- Ours, method proposed in this paper.

Experiment Settings (1) **Full-resource setting**: all training data are used. (2) **Zero-shot setting**: the training set and validation set are not available. Models are required to detect frames directly on the test set. Since recurrent neural networks and fine-tuning are not applicable, we only compare the prompting-based methods. (3) **Few-shot setting**: N-Way K-shot setup is applied where N is the number of classes in each dataset, and for each class, we take K samples for training and validation respectively. K is set to 2, 4, and 8 in our experiments.

Evaluation Metrics. Since each text can have more than one frame, this prediction task is a multi-label classification task. Results are reported in terms of micro- and macro-F1 score.

5.2 Implementation Details

For pre-trained language model involved in our framework and baselines, we use `bert-base-uncased` (Devlin et al., 2019). For pivot generation, we reserve top 50 frame indicators for each frame dimension. For masking probability, we set $p_v = 0.5$ and $p_n = 0.1$, following (Ben-David et al., 2020). We set training batch size = 16, learning rate of encoder=2e-5, learning rate of prompt encoder = 1e-2, λ = 1. Early stopping strategy is applied to avoid over-fitting. Empirically, we find the best threshold of multi-label classification by validation set for each model respectively. Our implementation[1] is partially based on OpenPrompt (Ding et al., 2022).

5.3 Overall Performance

Results for supervised learning across different methods are reported in Table 2. We can find: (1) Our method has shown improvement compared to all other baselines. Results across different datasets indicate that our method can handle frames of different topics and typologies in a unified framework. (2) Previous study (Vu et al., 2021) has shown that it's not easy for prompt tuning to surpass fine-tuning when using a small PLM like BERT. Benefiting from the pre-training stage, we can achieve comparable results to vanilla fine-tuning. (3) Traditional transfer methods don't perform as well as expected. It demonstrates that it's not easy to optimize multiple objectives due to the gaps between datasets. Encoding knowledge of auxiliary data

[1]Codes are publicly available at https://github.com/xymou/Frame_Detection

Method	Avg MiF	Avg MaF
Full Implementation	**74.11**	**68.47**
:w/o pivot-based encoder	72.64	66.69
:w/o issue adversarial training	73.18	66.09
:w/o both	72.55	65.59

Table 4: Results for ablation study. (**Bold**: the best performance in the column)

into PLM in the pre-training stage is an alternative.

Table 3 presents the results individually learned on unseen issues in **twitter** dataset, where training data is less. The results show that learned prompts can be well adapted to unseen issues.

5.4 Ablation study

We implement an ablation study to verify the effectiveness of pivot-based encoder and adversarial training for prompts. When we test without a pivot-based encoder, we use an ordinary BERT backbone. The average MiF and MaF of all datasets in ablation study are in Table 4. It is shown that both components contribute to the framework. This indicates that motivating both encoder and prompts to learn shared, transferable information across issues is crucial for frame detection.

5.5 Zero-shot and Few-shot Analysis

Method	gvfc	twitter	immi-general	immi-specific	fora
MP	43.03	34.48	59.08	19.92	67.21
P-Tuning	43.51	35.71	57.58	20.03	67.60
Ours	**44.65**	**37.44**	**60.30**	**20.28**	**68.96**

Table 5: P@1 in zero-shot experiments. We only compare prompt-based methods since neural networks and fine-tuning with random initialization is not applicable. (**Bold**: the best performance in the column)

Table 5 shows results in zero-shot setting. We didn't report the performance of neural networks, fine-tuning, and P-tuning v2 because a randomly initialized classifier may not produce reasonable results. Since validation data is not available, comparing F1 scores of different models using a random threshold may be unfair, we only illustrate P@1(precision at one). It indicates that prompting methods without fine-tuning can already induce reasonable predictions. Furthermore, with framing domain pre-training, our method shows effectiveness in mining frame-related knowledge in pre-trained language models.

Table 6 shows results in few-shot setting. Overall, prompting methods outperform fine-tuning by a large margin. Meanwhile, our method shows superiority in most datasets. Surprisingly, MP performs best in **immi**-specific. Several factors may account for this result. (1) With a manual template and verbalizer, MP has the least additional parameters and thus has the least risk of over-fitting. (2) Since the labels have some uncommon words like *humanitarian*, having less overlap with those in the pre-training stage, the label embedding verbalizer for this dataset is not well initialized in the generalization stage. Once we replace the verbalizer with a manual mapping, performance can be improved by about 3%. (3) Some data in **immi** is singly annotated without consensus-coding, so the potential noise brings more randomness to few-shot training.

6 Discussion

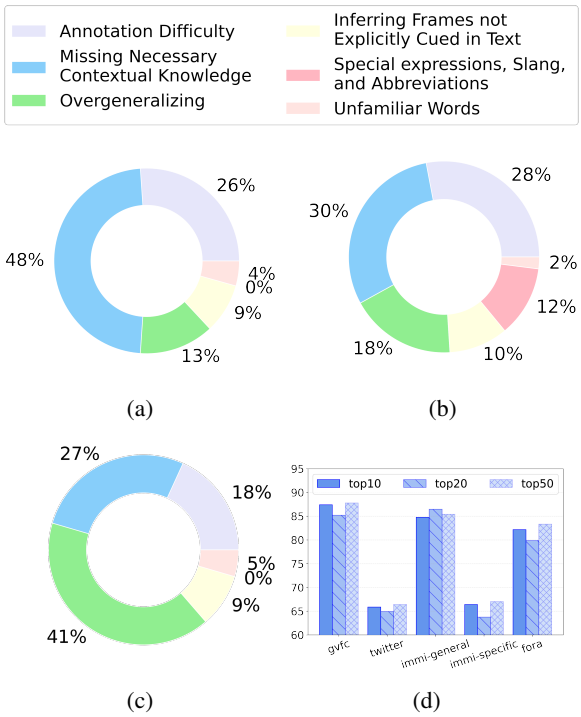

Figure 4: (a)error type proportion in **gvfc**; (b)error type proportion in **twitter** and **immi**; (c)error type proportion in **fora**; (d)P@1 across three setups with a varying number of pivots. (Best viewed in color.)

6.1 Analysis on the Number of Pivots

Here, we explore impact of different number of pivots. We reserve top 10, 20 and 50 frame indicators for each frame in Sec 2.1. Figure 4d presents our results. It is observed that the performance is stable across pivot numbers, especially for datasets having labels with more overlap to **mfc** dataset, i.e., **twitter** and **immi**-general.

Shot	Method	gvfc		twitter		immi-general		immi-specific		fora	
		MiF	MaF	MiF	MaF	MiF	MaF	MiF	MaF	MiF	MaF
2	FT	28.06	22.44	14.59	11.75	29.92	28.53	19.52	18.85	35.69	34.85
	MP	35.86	25.93	26.32	20.79	26.10	18.62	**33.63**	**25.49**	52.55	52.18
	P-tuning	42.73	27.06	29.40	26.39	**52.44**	**49.43**	21.32	15.36	64.22	64.26
	Ours	**43.66**	**29.65**	**29.45**	**26.61**	52.09	47.60	22.91	17.38	**65.92**	**64.85**
4	FT	30.27	25.39	17.12	14.17	33.16	29.13	20.54	18.36	36.54	35.82
	MP	53.05	40.82	37.06	26.99	36.89	26.55	**32.78**	**23.84**	51.09	50.31
	P-tuning	48.49	36.27	35.69	27.08	57.34	52.02	24.24	14.76	66.85	**67.04**
	Ours	**53.18**	**41.03**	**38.90**	**29.33**	**58.44**	**52.86**	26.39	19.34	**66.99**	66.20
8	FT	47.58	40.53	21.74	16.48	41.45	36.46	23.51	19.87	38.27	37.47
	MP	57.88	**52.80**	42.13	31.91	37.11	25.79	**36.44**	**28.19**	65.03	64.40
	P-tuning	56.83	48.88	39.91	33.03	60.20	54.48	31.33	21.02	68.43	67.66
	Ours	**58.46**	50.14	**42.93**	**35.56**	**60.35**	**54.99**	31.97	23.55	**68.49**	**67.97**

Table 6: Results in few-shot experiments, where we randomly sample 2,4,8 samples of each class for training. (**Bold**: the best performance in the column)

6.2 Error Analysis

We identify prediction errors by analyzing 100 random samples for each scenario. On the basis of (Mendelsohn et al., 2021), we add several additional types, and all error types included are shown in the table in Appendix A. Using the sampled instances, we also counted the proportion of each error type to all errors and get Figure 4a, 4b and 4c. We notice that *missing necessary contextual knowledge* is a common challenge, where key lexicons like names of politicians are cued, but the model lacks real-world knowledge or meta-data to make the right induction. External knowledge may be useful to deal with this issue. Also, *overgeneralizing* is a tricky problem, where informative words appearing in different contexts may mislead models. Compared to other models, ours is mainly trapped by this problem, since we have focused on the pivots, which sometimes can be misleading in highly-related frames like *legality, constitutionality and jurisdiction* and *crime and punishment*. Besides, there are scenario-specific challenges. For social media where expression is informal and diverse, slang and abbreviations like hashtags are useful but models may not make full use of them.

7 Related Work

Framing (Entman, 1993) is a communication strategy widely adopted by news media (Card et al., 2015) and politicians (Mou et al., 2021). Most researches on frames focus on news media. Boydstun et al. firstly conclude 15 frames that cross-cut issues from the policy agenda and use them to analyze news articles on 3 issues. Based on this, Card et al. construct Media Frame Corpus (mfc), one of the first large-scale datasets of frame annotations on news articles. This corpus is used to detect frames by dirichlet persona model (Card et al., 2016), deep recurrent neural networks (Naderi and Hirst, 2017) and lexicon analysis(Field et al., 2018). (Liu et al., 2019a) curated Gun Violence Frame Corpus (gvfc), which contains news headlines. They fine-tune BERT (Devlin et al., 2019) for prediction. Frames have also been studied on tweets and statements of politicians and the public. Johnson et al. annotated some tweets of politicians and use the Probabilistic Soft Logic model to detect frames. Roy et al. use a similar method to identify Morality frames in tweets. Since framing is an important communication strategy, frame detection in online fora and debates also aroused researchers' interest (Hartmann et al., 2019; Ajjour et al., 2019; Heinisch and Cimiano, 2021).

According to Boydstun et al., there are mainly two frame schemas, i.e., general and issue-specific. Although most work focuses on general frame prediction, some studies specialize in frames designed for a certain issue. Mendelsohn et al. fine-tune RoBERTa (Liu et al., 2019b) for analysis of different typologies of frames in immigration. Liu et al. conclude 9 issue-specific frames for gun violence issue.

Prompt learning aims to wrap the original input text using a template with a cloze or prefix prompt, and then the language model is used to fill the unfilled information to obtain a final string, which will be mapped into labels by a verbalizer (Schick and Schütze, 2020a; Liu et al., 2021a). This paradigm performs well on few-shot and zero-shot settings (Brown et al., 2020). Prompts can be manually

designed or learned with differentiable parameters (Liu et al., 2021c). Despite progress in prompt learning, few studies explore the transferability of prompts. Recently, (Vu et al., 2021; Wang et al., 2021) verify the effectiveness of reusing prompts of similar tasks to realize task transfer.

Massive work on cross-domain applies domain adversarial approaches to learn domain-invariant features (Ganin and Lempitsky, 2015; Ganin et al., 2016). Some researches on cross-domain sentiment classification extract common shared features, called pivots, which are frequent in both source and target domains and are prominent (Ziser and Reichart, 2018; Ben-David et al., 2020). Learning pivots and non-pivots help capture features for the task, rather than a specific domain.

8 Conclusion and Future Work

In this paper, we propose a general framework for frame detection that can handle various issues and frames. With the help of domain adaptation techniques, we enable both the encoder and prompts to learn transferable knowledge related to frames, thus yielding improvement on several datasets. Taking advantage of prompt learning, the framework can also deal with low-resource scenarios. In the future, we plan to conduct experiments on other formulations of framing analysis, e.g., diffusion of frames.

Acknowledgement

This work is partially supported by Natural Science Foundation of China (No.6217020551, 71991471), National Social Science Foudation of China (No. 20ZDA060), Science and Technology Commission of Shanghai Municipality Grant (No.20dz1200600, 21QA1400600, 21511101000) and Zhejiang Lab (No. 2019KD0AD01).

References

Yamen Ajjour, Milad Alshomary, et al. 2019. Modeling frames in argumentation. In *Proc. of EMNLP*.

Eyal Ben-David, Carmel Rabinovitz, and Roi Reichart. 2020. Perl: Pivot-based domain adaptation for pre-trained deep contextualized embedding models. *TACL*.

Amber E Boydstun, Dallas Card, et al. 2014. Tracking the development of media frames within and across policy issues. *Technical report, University of California, Davis.*

Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. 2020. Language models are few-shot learners. *Advances in neural information processing systems*, 33:1877–1901.

Dallas Card, Amber Boydstun, et al. 2015. The media frames corpus: Annotations of frames across issues. In *Proc. of ACL*.

Dallas Card, Justin H Gross, Amber Boydstun, and Noah A Smith. 2016. Analyzing framing through the casts of characters in the news. In *Proc. of EMNLP*.

Kenneth Church and Patrick Hanks. 1990. Word association norms, mutual information, and lexicography. *CL*.

Jacob Devlin, Ming-Wei Chang, et al. 2019. Bert: Pre-training of deep bidirectional transformers for language understanding. In *Proc. of NAACL*.

Ning Ding, Shengding Hu, Weilin Zhao, Yulin Chen, Zhiyuan Liu, Haitao Zheng, and Maosong Sun. 2022. OpenPrompt: An open-source framework for prompt-learning. In *Proc. of ACL: System Demonstrations*.

Robert M. Entman. 1993. Framing: Toward clarification of a fractured paradigm. *Journal of Communication*.

Anjalie Field, Doron Kliger, et al. 2018. Framing and agenda-setting in russian news: a computational analysis of intricate political strategies. In *Proc. of EMNLP*.

Yaroslav Ganin and Victor Lempitsky. 2015. Unsupervised domain adaptation by backpropagation. In *Proc. of ICML*.

Yaroslav Ganin, Evgeniya Ustinova, Hana Ajakan, et al. 2016. Domain-adversarial training of neural networks. *JMLR*.

Mareike Hartmann, Tallulah Jansen, et al. 2019. Issue framing in online discussion fora. In *Proc. of NAACL*.

Philipp Heinisch and Philipp Cimiano. 2021. A multi-task approach to argument frame classification at variable granularity levels. *it-Information Technology*.

Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long short-term memory. *Neural computation*.

Mengting Hu, Yike Wu, Shiwan Zhao, Honglei Guo, Renhong Cheng, and Zhong Su. 2019. Domain-invariant feature distillation for cross-domain sentiment classification. In *Proc. of EMNLP*.

Kristen Johnson, Di Jin, and Dan Goldwasser. 2017. Leveraging behavioral and social information for weakly supervised collective classification of political discourse on twitter. In *Proc. of ACL*.

Matthew S Levendusky. 2013. Why do partisan media polarize viewers? *American Journal of Political Science*.

Liang Li, Weirui Ye, Mingsheng Long, et al. 2020. Simultaneous learning of pivots and representations for cross-domain sentiment classification. In *Proc. of AAAI*.

Pengfei Liu, Weizhe Yuan, Jinlan Fu, Zhengbao Jiang, Hiroaki Hayashi, and Graham Neubig. 2021a. Pre-train, prompt, and predict: A systematic survey of prompting methods in natural language processing. *arXiv preprint arXiv:2107.13586*.

Siyi Liu, Lei Guo, et al. 2019a. Detecting frames in news headlines and its application to analyzing news framing trends surrounding us gun violence. In *Proc. of CoNLL*.

Xiao Liu, Kaixuan Ji, Yicheng Fu, Zhengxiao Du, Zhilin Yang, and Jie Tang. 2021b. P-tuning v2: Prompt tuning can be comparable to fine-tuning universally across scales and tasks. *arXiv preprint arXiv:2110.07602*.

Xiao Liu, Yanan Zheng, Zhengxiao Du, Ming Ding, Yujie Qian, Zhilin Yang, and Jie Tang. 2021c. Gpt understands, too. *arXiv preprint arXiv:2103.10385*.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019b. Roberta: A robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692*.

Julia Mendelsohn, Ceren Budak, and David Jurgens. 2021. Modeling framing in immigration discourse on social media. In *Proc. of NAACL*.

Xinyi Mou, Zhongyu Wei, Lei Chen, Shangyi Ning, Yancheng He, Changjian Jiang, and Xuanjing Huang. 2021. Align voting behavior with public statements for legislator representation learning. In *Proc. of ACL*.

Nona Naderi and Graeme Hirst. 2017. Classifying frames at the sentence level in news articles. *Policy*.

Jeffrey Pennington, Richard Socher, and Christopher D. Manning. 2014. Glove: Global vectors for word representation. In *Proc. of EMNLP*.

Shamik Roy, Maria Leonor Pacheco, and Dan Goldwasser. 2021. Identifying morality frames in political tweets using relational learning. In *Proc. of EMNLP*.

Timo Schick and Hinrich Schütze. 2020a. Exploiting cloze questions for few shot text classification and natural language inference. In *Proc. of EACL*.

Timo Schick and Hinrich Schütze. 2020b. It's not just size that matters: Small language models are also few-shot learners. In *Proc. of NAACL*.

Chi Sun, Xipeng Qiu, Yige Xu, and Xuanjing Huang. 2019. How to fine-tune BERT for text classification? *CoRR*, abs/1905.05583.

Tu Vu, Brian Lester, Noah Constant, Rami Al-Rfou, and Daniel Cer. 2021. Spot: Better frozen model adaptation through soft prompt transfer. *arXiv preprint arXiv:2110.07904*.

Chengyu Wang, Minghui Qiu, Jun Huang, and Xiaofeng He. 2020. Meta fine-tuning neural language models for multi-domain text mining. *arXiv preprint arXiv:2003.13003*.

Chengyu Wang, Jianing Wang, Minghui Qiu, et al. 2021. Transprompt: Towards an automatic transferable prompting framework for few-shot text classification. In *Proc. of EMNLP*.

Yftah Ziser and Roi Reichart. 2018. Pivot based language modeling for improved neural domain adaptation. In *Proc. of NAACL*.

A Error Types

Table 7 shows the common error types mentioned in Error Analysis.

Code	Error Type	Description	Examples
1	Annotation Difficulty	These instances highlight the challenges of annotation: there are convincing arguments that model's predicted frames can be appropriate labels.	*Oh and the death penalty does not deter crime*. Model predicted *Policy Evaluation*.
2	Missing Necessary Contextual Knowledge	Some frames are cued by lexical items but lack real-word knowledge or meta-data to induce the frames. E.g., model doesn't know the author of tweets is politicians so that it can not recognize "my collegues" refers to Political Factor.	*Joined 210 of my colleagues in urging supremecourt to ensure equal marriage rights*. Model missed *Political Factor*.
3	Overgeneralizing	Some words are highly related. The model makes erroneous predictions when such features are used in different contexts.	*American bombs aren't yet falling on syria, but chuck hagel suggested they will.* Model erroneously predicted *Security*.
4	Inferring Frames not Explicitly Cued in Text	Model predicts frames that may capture an author's intention but without sufficient evidence from the text.	*Stop immigration.* Model erroneously predicted *Public Order*.
5	Special Expressions, Slang, and Abbreviations	Some special terms, hashtags, abbreviations that indicate certain frames but not captured by the model.	*American tax dollars must not be used to aid and abet any dictatorial regime that stands with terrorists! #noaid2egypt* Model missed *External Reputation*.
6	Unfamiliar Words	Some unfamiliar but important clues are mentioned. Since they appear infrequently during training, language models may not understand them well enough .	*Dementia complicates us gun ownership.* Model missed *Mental Health*.

Table 7: Common errors in frame detection.

Summarizing Patients' Problems from Hospital Progress Notes Using Pre-trained Sequence-to-Sequence Models

Yanjun Gao*
ICU Data Science Lab
School of Medicine and Public Health
University of Wisconsin-Madison
ygao@medicine.wisc.edu

Dmitriy Dligach
Loyola University Chicago
ddligach@luc.edu

Timothy Miller, Dongfang Xu
Boston Children's Hospital
and Harvard Medical School
Timothy.Miller, Dongfang.Xu
@childrens.harvard.edu

Matthew M. Churpek, Majid Afshar
ICU Data Science Lab
School of Medicine and Public Health
University of Wisconsin-Madison
mchurpek, mafshar@medicine.wisc.edu

Abstract

Automatically summarizing patients' main problems from daily progress notes using natural language processing methods helps to battle against information and cognitive overload in hospital settings and potentially assists providers with computerized diagnostic decision support. Problem list summarization requires a model to understand, abstract, and generate clinical documentation. In this work, we propose a new NLP task that aims to generate a list of problems in a patient's daily care plan using input from the provider's progress notes during hospitalization. We investigate the performance of T5 and BART, two state-of-the-art seq2seq transformer architectures, in solving this problem. We provide a corpus built on top of progress notes from publicly available electronic health record progress notes in the Medical Information Mart for Intensive Care (MIMIC)-III. T5 and BART are trained on general domain text, and we experiment with a data augmentation method and a domain adaptation pre-training method to increase exposure to medical vocabulary and knowledge. Evaluation methods include ROUGE, BERTScore, cosine similarity on sentence embedding, and F-score on medical concepts. Results show that T5 with domain adaptive pre-training achieves significant performance gains compared to a rule-based system and general domain pre-trained language models, indicating a promising direction for tackling the problem summarization task.

1 Introduction

The progress note is a common note type in the electronic health record (EHR) that also contains the necessary details for medical billing; therefore, every hospital day will contain at least one progress note for a patient. Healthcare providers write them to document a patient's daily progress and care plan (Brown et al., 2014). The progress note contains both subjective and objective information gathered by the care team, and it is updated daily and serves as the most viewed clinical document by providers. The complexity of the progress note increases as the patient's illness worsens with progress notes collected in the intensive care unit (ICU) representing the sickest patients in the hospital. In the ICU, information and cognitive overload occur frequently, with more opportunities for missed diagnoses and medical errors (Furlow, 2020; Hultman et al., 2019). Automatically generating a set of diagnoses/problems in a progress note may assist providers in overcoming cognitive biases and heuristics and apply evidence-based medicine via information synthesis to accurately understand a patient's condition. These processes may ultimately reduce the effort in document review and augment care during a time-sensitive hospital event (Devarakonda et al., 2017).

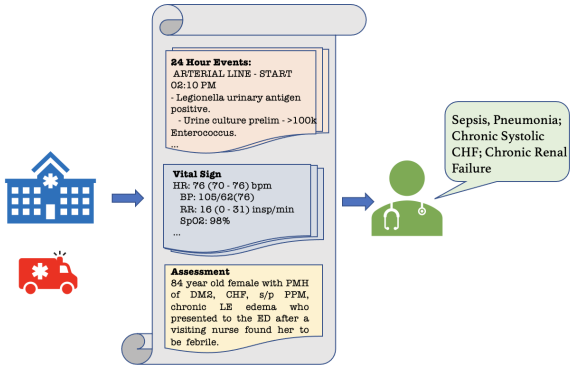

Figure 1: When a sick patient arrives to the hospital, diagnostic evaluations are performed to assess the patient's condition and deduce the problems causing the illness.

Proceedings of the 29th International Conference on Computational Linguistics, pages 2979–2991
October 12–17, 2022.

Clinical note summarization using natural language processing (NLP) has demonstrated promise in previous work. Hirsch et al. (2015) introduced HARVEST, an EHR summarizer that is currently deployed at point-of-care in a New York hospital. The NLP components of HARVEST include a Markov chain named-entity tagger that identifies diseases explicitly mentioned in clinical notes and a TF-IDF scorer that weighs the importance of the mentions (Lipsky-Gorman and Elhadad, 2011; Hirsch et al., 2015). With the advances of neural methods, recent work has focused on radiology report summarization (Zhang et al., 2018; MacAvaney et al., 2019; Gharebagh et al., 2020) with pointer generator network (See et al., 2017), and doctor-patient conversation summarization (Yim and Yetisgen-Yildiz, 2021; Zhang et al., 2021) with transformer architectures (Vaswani et al., 2017; Raffel et al., 2020). Few investigations apply transformers to *Problem Summarization* progress notes to identify and generate the top diagnoses during a patient's hospitalization.

Problem summarization requires complex cognitive processes to arrive at an accurate diagnosis. When a patient is admitted to the hospital, medical evaluations and diagnostics are initially performed to understand a patient's condition. The review is accompanied by documentation in the progress notes to include pertinent details about the patient's symptoms, medications, physical exam findings, radiology findings, laboratory results, etc. These data are organized in the progress note and used with the physician's medical knowledge to arrive at an assessment of the current problems followed by a treatment plan. The system of nonanalytic and analytic reasoning strategies represent **clinical diagnostic reasoning**, a process involving clinical evidence acquisition with integration and abstraction over medical knowledge to synthesize a conclusion in the form of a diagnosis(Barrows et al., 1980; Bowen, 2006). We hypothesize that to summarize a patient's problems and ultimately develop computerized diagnostic decision support systems, the ability of clinical diagnostic reasoning is the key for NLP systems, a gap in existing NLP literature. In this work, we propose a new summarization task designed to meet the real-world need in the hospital setting as the first step to developing NLP models for clinical diagnostic reasoning. The task is built on a new annotation subset of MIMIC-III (Johnson et al., 2016), a large and publicly available EHR. We formulate the task as a problem list summarization, as we see the task as the first step in a bigger vision of generating entire notes or sections of notes. Ultimately, the task is designed with our clinical informatics partners to move toward a future real-world application, where a system generating relevant diagnoses can assist healthcare providers and overcome the cognitive burden and information overload. Our contributions include:

- The first knowledge-intensive summarization task towards building NLP systems for computerized diagnostic decision support (sec §3), with an annotated set of clinical notes that are publicly available (sec §4);
- An evaluation of two transformer models for this task, T5 (Raffel et al., 2020) and BART (Lewis et al., 2020), to examine progress in using the state-of-the-art models over a rule-based medical concept extractor (sec §5);
- Domain adaptive pre-training to establish benchmark performance for this task across multiple evaluation metrics (sec §6), with discussion of key challenges and future directions (§7).

2 Related Work

In this section, we provide a brief overview of recently published papers on clinical summarization that use neural methods.

Task setup The stream of recent work on clinical summarization may be divided into two groups: extractive summarization and abstractive summarization. The data corpora are heterogeneous, with multiple note types represented. For extractive sumarization, Liang et al. (2019) proposes a summarization task that extracts sentences from progress notes. Adams et al.(2021) introduces a clinical note summarization task to generate a discharge summary generated from prior notes during hospitalization. More efforts have been made toward abstractive summarization. Several work focus on summarizing radiology reports into an impression, a short piece of text stating the findings from the source image (Zhang et al., 2018; MacAvaney et al., 2019; Gharebagh et al., 2020). Another task is doctor-patient conversation summarization where the output is a summary describing the patient's visit: (Yim and Yetisgen-Yildiz, 2021; Manas et al.,

Input Assessment and Subjective Sections
(*Assessment*) Pt is a 78 y.o female with h.o COPD, HTN, recent MVA with R.ankle/foot fx who presents with hypoxia and LLL infiltrate.
Chief Complaint: Pt does not feel better than at admission, still very fatigued and weak. SOB unchanged. No chest pain. No other complaints.
Allergies: No Known Drug Allergies
Review of systems is unchanged from admission except as noted below Review of systems:

Figure 2: An input example of assessment and subjective sections available in the notes: Chief Complaint, Allergies, Review of systems.

2021; Zhang et al., 2021); or generating clinical notes using both extractive and abstractive summarization: (Krishna et al., 2021). Our work is similar to (Liang et al., 2019) in the emphasis on summarizing problems from progress notes. Yet, Liang et al. (2019) uses a disease-specific dataset (hypotension and diabetes), and formulates the problem as extractive summarization. Our annotations span a broad range of diagnoses across multiple disciplines (surgery, medicine, neurology, cardiology, trauma, etc.) and investigate extractive and abstractive approaches in the task.

Evaluation Prior work has relied on ROUGE (Lin, 2004) as the primary evaluation metric for summarization. Most papers also report human evaluation with aspects of clinical relevancy, factual accuracy and readability (MacAvaney et al., 2019; Gharebagh et al., 2020; Krishna et al., 2021; Yim and Yetisgen-Yildiz, 2021; Zhang et al., 2021). Few have evaluated using a concept F-score, measuring if the predicted summaries contain accurate medical concepts (Liang et al., 2019; Zhang et al., 2021). Our work follows prior work and uses ROUGE, concept F-score, and human evaluation to assess the quality of generated summaries. We also evaluate content quality based on semantic representation using BERTScore (Zhang* et al., 2020) and cosine similarity for sentence embedding.

3 Task Description

Many clinical NLP applications aim to improve physicians' efficiency and decision-making by automatically highlighting essential information from the large body of textual data in the EHR. The goal of Problem Summarization is to identify and generate the problems and diagnoses for the patient's ICU stay. The Problem Summarization task could be developed using a multi-document approach with all notes captured during a hospital encounter. A patient encounter may generate multiple clinical notes (e.g. admission note, transfer note, daily progress notes, etc.), involving different modalities of data such as structured EHR data and radiology images. However, we are particularly interested in facilitating NLP model development for clinical diagnostic reasoning. We define the task as single-document summarization and focus only on a cross-sectional point in time with a single progress note. Our work will show that summarizing a patient's problems over a single progress note is a challenging task and a necessary foundation that requires clinical text understanding and reasoning over sequences of medical concepts.

The progress note is organized in the ubiquitous SOAP format with four components: *S*ubjective, *O*bjective, *A*ssessment, and *P*lan, a documentation method designed to present patient's problems in a highly structured way and developed by Larry Weed, MD (Weed, 1964). Each component has multiple sections gathering patients' information, helping the healthcare providers quickly recognize medical events and active problems. SUBJECTIVE sections are written in natural language and record information about health concerns expressed by patients (e.g. CHIEF COMPLAINTS), and past medical events and history (e.g. ALLERGIES, FAMILY HISTORY). OBJECTIVE sections are primarily structured data, including vital signs, lab tests, medications. ASSESSMENT is a brief description of passive and active diagnoses. It states why the patient is admitted to the hospital and the active problem for the day, usually accompanied by the patient's comorbidities. PLAN section includes multiple subsections, each listing a medical problem and treatment plan. The progress note is time-sensitive EHR data because it is documented daily. As a patient's condition changes and the length of stay increases, the progress note may also increase in length. Another reason for the increasing size is from copy-and-paste behaviour, also known as "note bloat" adding redundant information or noise and hindering the efficiency in data synthesis, which increases the risk for medical error (Rule et al., 2021; Tsou et al., 2017; Shoolin et al., 2013). This reiterates our motivation to develop an NLP system that automatically generates problems and diagnoses to assist providers in clinical workflow and improve diagnostic accuracy.

Our task took Subjective and Assessment sections in progress notes as input and omitted the

Objective sections. Both the Subjective sections and the Assessment section contained information about the reason for admission; therefore, they became the source text (see Figure 2 for an example). The reference summary is a list of problems mentioned in each Plan subsection relevant to the reasons for hospitalization. We will explain the annotation process in the next section. [1]

4 Data

All progress notes were sourced from MIMIC-III, a publicly available dataset of de-identified EHR data from approximately 60,000 hospital ICU admissions at Beth Isreal Deaconess Medical Center in Boston, Massachusetts. We randomly sampled a subset of 768 progress notes and annotated the text spans for the SOAP components. The goal of the annotation was to obtain lists of problems from the Plan subsections. For each Plan subsection, the annotators marked the text span for the Problem, separating the diagnosis/problems from the treatment or action plans. The annotators subsequently determined if the problem was a primary diagnosis (*Direct*), or a past medical problem or consequence from the primary diagnosis (*Indirect*). Two more labels were available for annotating the Plan subsection: *Neither* if the problem or diagnosis was not mentioned in the progress note; *Not Relevant* if the Plan subsection contained non-diagnostic comments such as describing nutrition, prophylaxis, or disposition. Finally, we concatenated the Direct and Indirect problems using semi-colons and used them as *reference summary*. Two medical school students were trained as annotators under the supervision of two board-certified critical care ICU physicians. On the four labels, they achieved a Cohen's Kappa of 0.74 on 10 randomly sampled notes, considered as good quality given the complexity of the task. More details may be found in the Appendix A. [2]

Figure 3 illustrates the task setup. The Direct and Indirect problems were labeled from each Plan subsection using information presented in the input Assessment (entire progress note was also available to the annotators for more information), forming the reference summary (All Problems in the bottom). A total of 1404 and 1599 text spans were labeled as Direct and Indirect Problems, respectively. The majority of the Direct problems were found in the input Assessment but many of the Indirect problems were not explicitly mentioned in the input Assessment and may be found in other parts of the progress note (*abdominal pain* finding in Subjective or *pneumothorax* finding in chest imaging result of Objective). We also performed medical concept mapping through UMLS (see §5) on the input Assessment and kept the overlap with the reference summaries and categorized them *Explicit Mention of Problems* as an automated labeling approach and baseline. Therefore, the problems represent extractive and abstractive medical concepts. We presented the results across these subgroups assuming the complexity increases as we move from Explicit to Direct to Indirect problems.

5 Experiment Setting

The Unified Medical Language System (UMLS) from the National Library of Medicine is the largest resource containing biomedical concepts and their relationships (Bodenreider, 2004). We applied the concept extractor from QuickUMLS (Soldaini and Goharian), a fast and lightweight Python package, to identify all the medical concepts in the text as the baseline system. Two state-of-the-art seq2seq transformers were selected to compare with the rule-based method: T5 (Raffel et al., 2020) and BART (Lewis et al., 2020). The transformer models are known as data hungry and pre-trained on domain general text, yet our training data was limited in size but full of medical terms. To help the model learn the medical vocabulary and knowledge, we used data augmentation to generate more training samples for our experiments (§5.1 and §5.2).

5.1 Data augmentation

Figure 4 presents a workflow of the data augmentation method across the following three steps: (1) concept identification; (2) synonym mapping; and (3) augmented sample generation. Given an input text, the step of concept identification extracted ngram terms that were matched concepts with UMLS entities, from QuickUMLS. This step was done through a text matcher algorithm using a cosine similarity threshold, setting as Jaccard score with cosine similarity as 1 in our use case. The results returned Concept Unique Identifiers (CUI), a symbolic ID for the medical concept from UMLS. An example output of this step is illustrated in Figure 4: a dictionary of the matched ngrams, e.g.

[1] Training script is available at: https://git.doit.wisc.edu/smph/dom/UW-ICU-Data-Science-Lab/drbench.

[2] Annotation is available through PhysioNet.

Input Assessment Assessment and Plan HYPOTENSION (NOT SHOCK) .H/O CIRRHOSIS OF LIVER , ALCOHOLIC SUPRAVENTRICULAR TACHYCARDIA (SVT) 53 yo female with a history of ETOH cirrhosis c/b persistent leukocytosis and elevated TBili, hx ascites , grade 1 varix , no hx SBP or encephalopathy , recently discharged from), electively admitted for an EUS to evaluate a GE junction polyp (aborted given hypotension), now transferred to the liver service for acute renal failure and?decompensated cirrhosis .

(Plan Subsection 1) Abdominal pain: Per patient, unchanged from previous. S/p Para and KUB, no evidence of SBP or free air. LFTs stable/slightly trending down. Likely [**3-10**] liver disease, though unclear if new or different from past. - Tylenol, Percocet and Ultram PRN pain - Zofran PRN nausea - Trend LFTs
(Plan Subsection 2) Acute renal failure: On admission up to 1.0, peaked at 1.6 and now back to baseline 0.9 after 2 units FFP yesterday. Was likely [**3-10**] ATN. . . .
. . .

Explicit Mention of Problems
hypotension, svt, cirrhosis, leukocytosis, acute renal failure
Direct Problems
Acute renal failure; ETOH cirrhosis
Indirect Problems
SVT with hypotension: This occured in setting of having RA/RV manipulated with wire; Abdominal pain; Leukocytosis; Anemia
All Problems (Reference Summary)
Acute renal failure; ETOH cirrhosis; SVT with hypotension: This occured in setting of having RA/RV manipulated with wire; Abdominal pain; Leukocytosis; Anemia

Figure 3: **Top**: An example assessment input with all the concepts (highlighted in color box) identified through QuickUMLS, a state-of-the-art off-the-shelf medical concept extractor. **Middle**: Two example plan subsections with the the annotated problems, with relation labels omitted. **Bottom**: The reference summary (All Problems) consists of problems annotated as the main reasons for hospitalization (Direct Problems) and secondary concerns (Indirect Problems); explicit mention of the problems is detected by overlapping the concepts identified through UMLS in input and reference summary.

Set	Fine-tuning	Data Augmt.	DAPT
#Notes	700	132k	293k
Input Lens$_{(\sigma)}$	$43.33_{24.75}$	$212.74_{78.39}$	$46.19_{70.95}$

Table 1: Size and average input length (and standard deviation σ) of training set for different experiment settings: the original annotated set for fine-tuning, the data generated from data augmentation method, and DAPT.

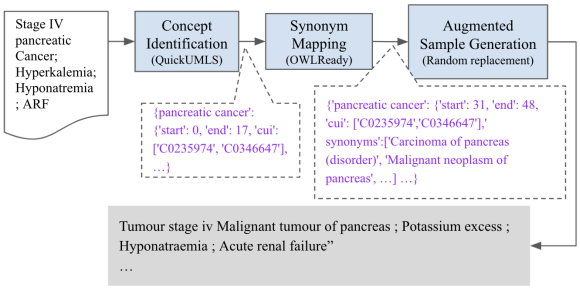

Figure 4: Workflow of the data augmentation method with an input reference summary and output augmented sample

"pancreatic cancer", with start and end character positions and CUIs, e.g. [C0235974]. The mapping module in step 2 found synonyms through CUIs. Here, we used OWLReady (Lamy, 2017) that automatically constructed an UMLS ontology graph, linking the concepts with relations and enabling a quick synonym lookup given a CUI. The synonyms were then passed to the last module for augmented sample generation. The last module randomly chose the synonyms and replaced concepts. An input text may contain n concepts, with each concept having r maximum number of synonyms, the number of combinations of synonyms r^n grows exponentially as n increases. Considering the efficiency, we limited the number of combinations generated by concept replacement to 1000. We ran the pipeline on both reference summary and input assessment, and obtained approximately 132,000 pairs of samples for additional training data. We conducted quality measurement on the augmented samples and report the results in the Appendix B.

5.2 Domain adaptive pretraining with random concept masking

The summarization task requires clinical text understanding and medical knowledge, exposing challenges for models pre-trained from the general domain. Previous work proposed strategies of continuously training the pre-trained language model on domain-specific tasks to enable domain knowledge learning. (Gupta et al., 2021; Pruksachatkun et al., 2020; Gururangan et al., 2020). We followed a similar approach to investigate the effect of domain adaptive pre-training (DAPT) on our summarization task. Specifically, we continuously trained T5 on ASSESSMENT AND PLAN sections from all progress notes in MIMIC, excluding the set of notes for the test set. The result set had 293,000 notes, with the top 3 most frequent note types as Nursing Progress Notes (181k), Physician Resident Progress Notes (61k), and Intensivist Notes (25k).

T5 was trained by random token masking: given a text string, it randomly replaced the token spans with a special tag *"<extra_id_>"* and learned to generate the masked tokens. However, not all words were equally important in our task and we wanted the model to learn clinical semantic types such as symptoms and diseases. Previous work proposed masking on biomedical entities and time expression, achieving performance gains when compared to BERT without entity masking (Lin et al., 2021; Pergola et al., 2021). Inspired by these work, we adapted a concept masking policy where we randomly masked the concepts identified through UMLS. We set the mask token ratio to 15%. For example, the highlighted text in Figure 3 was randomly replaced with the special tag. The statistics of the training set are shown in Table 1.

5.3 Evaluation

We use ROUGE-L (Lin, 2004), a conventional metric in summarization evaluation that based on n-gram overlap, as well as BERTScore (Zhang* et al., 2020), reporting maximum pairwise cosine similarity on word embedding from reference summary and predicted summary. ROUGE fails to recognize synonyms and abbreviations, which are common in biomedical text: e.g., *heart attack* is the same clinical diagnosis as *myocardial infarction*, and *MI* is the abbreviation of *myocardial infarction*. BERTScore compensates this limitation by using contextualized word embeddings from SapBERT (Liu et al., 2021), a state-of-the-art BERT encoder (Devlin et al., 2019) for biomedical entity representation that assigns high cosine similarity for synonyms and abbreviations based on UMLS. The reliability of both metrics are validated in literature, thus we report them as main results.

Meanwhile, to better understand the system output, we provide two additional metrics that measure the quality of higher-level information and medical concepts. We took the hidden states of the last layer from SapBERT when taking reference and predicted summary as input, and measure the cosine similarity on sequence embedding (Sent.θ). To evaluate the model's performance in predicting medical concepts, we ran QuickUMLS to get all CUIs from the reference and predicted summaries and computed the F-score. This metric has its own limitation due to the tricky parameter tuning in matching algorithms, causing superfluous or deficient extraction. Regardless, we include them as approximate solutions towards knowledge-based evaluation for clinical summarization, and leave the metric development for future work.

In the experiments, we set the maximum input and output length to 512 and 128 tokens, respectively. The input text was truncated if the maximum length was exceeded. All experiments occurred on two NVIDIA Tesla V100 32GB GPUs. We used early stopping on the development set during training and saved the models with the highest validation ROUGE-L F-score for evaluation. More implementation details are presented in the Appendix C.

6 Results and Analysis

We evaluated all systems on a test set of 92 progress notes and summaries. Recall that the progress notes contained the Subjective, Objective, Assessment and Plan sections. We set two types of input to the models: (1) Assessment section only (ASSMT), (2) Assessment and Subjective sections (length permitting) (A+SUBJ). Both input settings also had augmented samples from the data augmentation method introduced earlier. We started with a simple rule-based system that was a UMLS concept extractor on the Assessment section. The evaluation metrics across the rule-based, fine-tuned T5 and BART (§6.1), and T5 with domain adaptation pre-training (DAPT, §6.2) are shown in Tables 2 and 3. T5 with DAPT outperformed all other systems and established a benchmark performance for the task. We include a qualitative analysis to provide data-driven insights of the task (§6.3).

6.1 Overall performance of fine-tuned models

Table 2 represents ROUGE-L F-score, cosine similarity on sentence embedding (*Sent.θ*), BERTScore and CUI F-score. Overall, scores dropped from Explicit Mentions to Direct Problems to Indirect Problems, likely due to increasing complexity with more abstractive concepts over extractive concepts. Explicit Mention summarization was the easiest and Indirect Problem summarization was the hardest. The rule-based system outperformed all T5 and BART variants on the Explicit Mentions, given that it identified the obvious entity mentions. For T5 and BART, fine-tuning with augmented samples slightly improved the ROUGE scores. Adding subjective sections (A+SUBJ) did not bring benefits, possibly because most subjective sections are empty in ICU progress note. T5 had more variants with better scores than BART. In our manual in-

Model	Setting	Explicit Mentions				Direct Problems				Indirect Problems				All Problems			
		RL-F	Sent.θ	**BS**	CUI	**RL-F**	Sent.θ	**BS**	CUI	**RL-F**	Sent.θ	**BS**	CUI	**RL-F**	Sent.θ	**BS**	CUI
Rule-based	ASSMT	**34.45**	58.81	**59.80**	38.97	12.31	**55.33**	40.13	34.23	9.49	**55.58**	**44.46**	33.16	13.45	**68.61**	50.32	43.93
T5	ASSMT	32.77	59.57	57.75	41.73	13.68	53.44	39.72	**36.10**	10.40	54.76	44.16	35.08	14.82	67.49	49.89	44.51
	++	31.76	58.74	57.12	**42.19**	**13.78**	53.65	**40.30**	35.84	**10.55**	54.10	43.48	**35.20**	**15.00**	67.32	50.36	**44.55**
	A+SUBJ	20.24	50.04	47.55	33.44	9.52	51.91	39.72	30.43	7.10	54.14	43.87	30.29	10.89	64.63	49.75	39.02
	++	20.72	**59.64**	57.97	33.56	9.46	53.55	39.52	18.76	7.35	54.69	44.36	14.40	10.93	67.19	**50.42**	24.83
BART	ASSMT	25.70	54.98	52.99	32.49	10.00	53.66	39.08	29.41	8.04	54.66	43.12	29.04	11.56	66.86	48.48	38.36
	++	28.22	57.04	55.16	32.28	10.33	53.40	39.21	30.75	8.29	54.48	44.01	32.08	11.65	66.67	49.23	40.69
	A+SUBJ	18.80	49.19	46.77	26.96	7.04	51.70	38.24	25.30	6.00	54.29	43.71	26.01	9.25	64.95	48.19	34.02
	++	20.23	57.91	54.68	32.91	7.88	53.85	40.21	30.09	6.85	54.61	43.15	30.12	9.84	67.00	49.70	38.72

Table 2: ROUGE-L F-score (RL-F), sentence embedding cosine similarity (*Sent.θ*), BERTScore (*BS*), and evaluation using CUI F-score (*CUI*) from fine-tuning T5 and BART on the two input settings: Assessment (ASSMT), Assessment with Subjective sections(A+SUBJ.) ++ represents the training with data augmentation.

Setting	Model	Token Masking				Concept Masking			
		RL-F	Sent.θ	**BS**	CUI	**RL-F**	Sent.θ	**BS**	CUI
Explicit	ASSMT	32.66	61.34$_{(\uparrow 2.53)}$	56.68	47.10$_{(\uparrow 8.13)}$	29.86	55.87	53.91	40.27$_{(\uparrow 2.14)}$
	++	26.94	59.40$_{(\uparrow 0.59)}$	55.05	42.73$_{(\uparrow 3.76)}$	32.82	58.21	56.80	43.16$_{(\uparrow 4.19)}$
Direct	ASSMT	12.69	53.63	42.40$_{(\uparrow 2.27)}$	35.39$_{(\uparrow 1.16)}$	14.90$_{(\uparrow 2.59)}$	55.48$_{(\uparrow 0.15)}$	47.10$_{(\uparrow 6.97)}$	35.29$_{(\uparrow 1.06)}$
	++	10.44	53.47	43.46$_{(\uparrow 3.33)}$	37.45$_{(\uparrow 3.22)}$	15.76$_{(\uparrow 5.22)}$	56.82$_{(\uparrow 1.49)}$	48.72$_{(\uparrow 8.72)}$	37.74$_{(\uparrow 3.51)}$
Indirect	ASSMT	10.07$_{(\uparrow 0.58)}$	52.72	41.47	38.19$_{(\uparrow 5.03)}$	13.58$_{(\uparrow 4.36)}$	53.44	44.91$_{(\uparrow 0.45)}$	33.56$_{(\uparrow 0.40)}$
	++	8.04	51.84	40.45	37.53$_{(\uparrow 4.37)}$	13.28$_{(\uparrow 4.06)}$	55.02	45.51$_{(\uparrow 1.05)}$	35.10$_{(\uparrow 1.94)}$
All	ASSMT	14.49$_{(\uparrow 1.04)}$	62.40	49.62	40.44	18.72$_{(\uparrow 5.27)}$	64.69	54.03$_{(\uparrow 3.71)}$	42.69
	++	12.12	63.08	50.20	45.58$_{(\uparrow 1.65)}$	18.80$_{(\uparrow 5.35)}$	66.08	55.29$_{(\uparrow 4.86)}$	44.56$_{(\uparrow 0.63)}$

Table 3: Performance of T5 with domain adaptation pre-training using Assessment (ASSMT) as input, under two mask policies: Token Masking and Concept Masking. We report Rouge-L F-score (RL-F), and BERTScore (BS), as well as Sentence embedding cosine similarity (Sent θ) and CUI F-score. Numbers with green background address the highest performance across all results, with subscript number (↑) denoting the improvements over rule-based results.

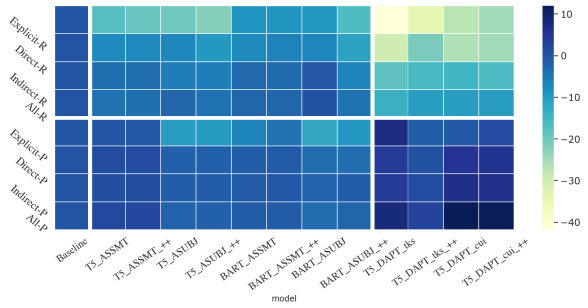

Figure 5: Performance drops (lighter color) and gains (darker color) over baseline (first column) on ROUGE-L Recall (top 4 rows) and Precision (bottom 4 rows). The darker the cell color is, the higher performance gain the model obtains over baseline.

vestigation, we find that BART generated text that are not relevant to medical domain[3]. In sum, all fine-tuned model performance were tied with the baseline, which is impressive given that the baseline uses domain knowledge (medical concept).

6.2 The effect of domain adaptation pre-training

Table 3 contains results from training T5 with DAPT and fine-tuned on the annotated set across two methods for masking: random token (T5-DAPT-TKS) and concept masking (T5-DAPT-CUI). To highlight the differences before and after DAPT, we showed four scores as well as the performance gained over the baseline system on ASSMT input. Overall, both DAPT settings delivered better performance. The performance gain of T5-DAPT-TKS was mainly from the CUI F-score (+1.16 to +8.13). Superior results were seen from T5-DAPT-CUI, achieving best performance on all setting except for Explicit Mention, yielding large performance gained on ROUGE F score(+2.59 to +5.35) and BERTScore (+0.45 to +8.72).

In addition, Figure 5 includes the ROUGE Recall and Precision drops and gains from all models over baseline. ROUGE Recall measures the content coverage and Precision computes content relevancy in predicted summary (Lin, 2004). All models reported lower recall compared to baseline, indicating their coverage was limited. T5 DAPT variants showed higher gains on precision, yielding the largest gain (+5 to +12 for T5-DAPT-CUI). These results indicate that continuously training T5 with domain vocabulary is a promising direction to solve the task.

[3]see Appendix D for example output for all models

[3]Semicolons are removed during fine-tuning and evaluation. We manually inserted them back for presentation purpose.

(1) **[Input Assmt]** 81 yo man with CAD s/p CABG, recent NSTEMI at OSH, HTN, CKD here for ERCP for cholangitis and admitted to ICU with hypotension.
[Reference] Cholangitis; CAD; HTN; AoCRF
[Prediction] cad; htn; cholangitis; hypotension; heparin; cholecystitis; dvt

(2) **[Input Assmt]** 81yo F with h/o ESRD on HD, HTN, HL, acute CHF exacerbation 1 month ago, admitted from OSH with NSTEMI now s/p cath with DES in LAD and LMCA.
[Reference] SP NSTEMI; HTN; Chronic renal failure; HL;
[Prediction] esrd; htn; cad; admitted senna chf exacerbation;

Figure 6: Two cherry-picked examples from T5-DAPT-CUI output, with cyan fonts highlighted the correct diseases.[4]

6.3 Qualitative analysis

Besides the numeric metrics reported above, we provide example predicted summaries and qualitative analysis done by a domain expert (a critical care ICU physician). We cherry-picked two examples from T5-DAPT-CUI that best represent the characteristics of medical diagnostic consistency in clinical diagnostic reasoning, and present them in Figure 6. Example 6.1 shows the model performed extractive summarization: it generated both hypertension and hypotension as relevant diagnoses that represent an Indirect label for past medical history of hypertension, and Direct label for an active problem during the hospitalization with hypotension. In example 6.2, the model performed abstraction summarization. The last half of the Assessment highlights a type of heart attack (e.g., "*NSTEMI*") requiring an emergent medical procedure (e.g., "*cath wtih DES in LAD and LMCA*"), and the model summarized a rather complex statement into a single, accurate diagnosis of Coronary Artery Disease in its abbreviated form as "*CAD*".

7 Discussion

Our work begins with a single note in cross-sectional design to build our models; however, a patient's hospitalization is a multi-document workflow with repeated measures of progress notes and other note types across several days and multiple providers. In addition, providers generate their diagnoses via a reasoning process that includes structured data from vital signs, laboratory results, etc. Images and radiology reports are another modality that highlights the multi-modality approach in diagnostic reasoning. Nonetheless, our work opens the door for future research on knowledge intensive clinical summarization. This section includes a discussion of future directions in solving this task.

(1) a. **[REF]** Coronary Artery Disease; v fibrillation
b. **[PRED]** LVEF-34% - LVEF:36%
*LVEF (left ventricular ejection fraction) value is abnormally low and indicates a heart disease.

(2) a. **[REF]** acute respiratory distress; CAD; chronic kidney injury, Acute Diastolic heart failure
b. **[PRED]** ABG on admission 7.25/72/80/32/34
*ABG (Arterial Blood Gas) reports the abnormally low pH of 7.25 and the abnormally high pCO_2 of 72, and could be from acute respiratory distress and/or acute diastolic heart failure.

Figure 7: Two example reference (**REF**) and predicted summaries (**PRED**) from T5-ALL (input with objective sections).

Exploring structured data The objective section of the progress note contain embedded structured data, delivering rich information regarding patient's problem. Recall the example in Figure 3, the reference summary contains diagnosis: "*Leukocytosis*" (high white blood cell count), "*anemia*"(low red blood cell count). These diagnosis are usually found in laboratory results. To investigate the use of objective sections and structured data, we append both Subjective and Objective sections in chronological order to the Assessment and input to T5 for fine-tuning and evaluation (T5-ALL), and we let the T5 tokenizer truncate the text when it exceeds the 512 token limit. On test set, the scores are too low to report. Yet, we observed that T5-ALL, instead of generating medical concepts, often extracts lines of lab values that strongly associate with the disease in reference summary (see Figure 7.1 and 7.2). This preliminary result indicates the future direction of understanding the association between disease and lab values in summarization.

Incorporating knowledge into models We propose a knowledge intensive summarization task that requires clinical text understanding, knowledge representation and diagnostic reasoning. The experiment results showed that the models pre-trained on medical concepts effectively improved the performance, while challenges remain in understanding the associations among medications, symptoms and disease. Recent work on event extraction and clinical relation extraction incorporates biomedical knowledge graph into pre-trained language models (Huang et al., 2020; Roy and Pan, 2021). Our future work will investigate the incorporation of knowledge graph into seq2seq pre-trained models.

Evidence-based evaluation Medical diagnosis is a critical component of effective healthcare but misdiagnosis is a major contributor to medical errors, especially in critical care settings where quick decision-making is needed. Medical diagnoses predicted by systems that are not redundant must

be contextually relevant to the data gathered in a progress note to achieve valid reasoning. We believe an automated evaluation method for problem summarization should assess the knowledge representation, non-redundancy, and evidence relevancy, and the automated metrics used in our work cover partial aspects. Recently, Moramarco et al. (2021) studied a fact-based evaluation for medical summarization using human evaluation, which we plan to carry out in future work.

8 Conclusion

We propose a problem summarization task that address diagnostic reasoning, and show that T5 with DAPT achieves benchmarking performance for the task, but some key challenges remained. Our work lays the ground for future research on knowledge fused clinical summarizers as well as real-world clinical diagnostic decision support system. Future work will investigate the uses of structured data, evidence-based evaluation metric and better models for knowledge representation and summarization.

Ethical Statement The use of the data in this research came from a fully de-identified dataset (contains no protected health information) that we received permission for use under a PhysioNet Credentialed Health Data Use Agreement (v1.5.0). The study was determined to be exempt from human subjects research. All experiments followed the PhysioNet Credentialed Health Data License Agreement.

Medical charting by providers in the electronic health record is at-risk for multiple types of bias. Our research focused on building a system to overcome the cognitive biases in medical decision-making by providers. However, statistical and social biases need to be addressed before integrating our work into any clinical decision support system for clinical trials or healthcare delivery. In particular, implicit bias towards vulnerable populations and stigmatizing language in certain medical conditions like substance use disorders are genuine concerns that can transfer into language model training (Thompson et al., 2021; Saitz et al., 2021; Karnik et al., 2020). Therefore, it should be assumed that our corpus of notes for this task will carry social bias features that can affect fairness and equity during model training. Before the deployment of any pre-trained language model, it is the responsibility of the scientists and health system to audit the model for fairness and equity in its performance across disparate health groups (Saleiro et al., 2018). Fairness and equity audits alongside model explanations are needed to ensure an ethical model trustworthy to all stakeholders, especially patients and providers.

References

Griffin Adams, Emily Alsentzer, Mert Ketenci, Jason Zucker, and Noémie Elhadad. 2021. What's in a summary? laying the groundwork for advances in hospital-course summarization. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 4794–4811.

Howard S Barrows, Robyn M Tamblyn, et al. 1980. *Problem-based learning: An approach to medical education*, volume 1. Springer Publishing Company.

Olivier Bodenreider. 2004. The unified medical language system (umls): integrating biomedical terminology. *Nucleic acids research*, 32(suppl_1):D267–D270.

Judith L Bowen. 2006. Educational strategies to promote clinical diagnostic reasoning. *New England Journal of Medicine*, 355(21):2217–2225.

PJ Brown, JL Marquard, B Amster, M Romoser, J Friderici, S Goff, and D Fisher. 2014. What do physicians read (and ignore) in electronic progress notes? *Applied clinical informatics*, 5(02):430–444.

Murthy V Devarakonda, Neil Mehta, Ching-Huei Tsou, Jennifer J Liang, Amy S Nowacki, and John Eric Jelovsek. 2017. Automated problem list generation and physicians perspective from a pilot study. *International journal of medical informatics*, 105:121–129.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. Bert: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186.

Bryant Furlow. 2020. Information overload and unsustainable workloads in the era of elec-

tronic health records. *The Lancet Respiratory Medicine*, 8(3):243–244.

Sajad Sotudeh Gharebagh, Nazli Goharian, and Ross Filice. 2020. Attend to medical ontologies: Content selection for clinical abstractive summarization. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 1899–1905.

Yash Gupta, Pawan Sasanka Ammanamanchi, Shikha Bordia, Arjun Manoharan, Deepak Mittal, Ramakanth Pasunuru, Manish Shrivastava, Maneesh Singh, Mohit Bansal, and Preethi Jyothi. 2021. The effect of pretraining on extractive summarization for scientific documents. In *Proceedings of the Second Workshop on Scholarly Document Processing*, pages 73–82.

Suchin Gururangan, Ana Marasović, Swabha Swayamdipta, Kyle Lo, Iz Beltagy, Doug Downey, and Noah A Smith. 2020. Don't stop pretraining: Adapt language models to domains and tasks. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 8342–8360.

Jamie S Hirsch, Jessica S Tanenbaum, Sharon Lipsky Gorman, Connie Liu, Eric Schmitz, Dritan Hashorva, Artem Ervits, David Vawdrey, Marc Sturm, and Noémie Elhadad. 2015. Harvest, a longitudinal patient record summarizer. *Journal of the American Medical Informatics Association*, 22(2):263–274.

Kung-Hsiang Huang, Mu Yang, and Nanyun Peng. 2020. Biomedical event extraction with hierarchical knowledge graphs. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 1277–1285.

Gretchen M Hultman, Jenna L Marquard, Elizabeth Lindemann, Elliot Arsoniadis, Serguei Pakhomov, and Genevieve B Melton. 2019. Challenges and opportunities to improve the clinician experience reviewing electronic progress notes. *Applied clinical informatics*, 10(03):446–453.

Alistair EW Johnson, Tom J Pollard, Lu Shen, Li-wei H Lehman, Mengling Feng, Mohammad Ghassemi, Benjamin Moody, Peter Szolovits, Leo Anthony Celi, and Roger G Mark. 2016. Mimic-iii, a freely accessible critical care database. *Scientific data*, 3(1):1–9.

Niranjan S Karnik, Majid Afshar, Matthew M Churpek, and Marcella Nunez-Smith. 2020. Structural disparities in data science: a prolegomenon for the future of machine learning. *The American Journal of Bioethics*, 20(11):35–37.

Kundan Krishna, Sopan Khosla, Jeffrey P Bigham, and Zachary C Lipton. 2021. Generating soap notes from doctor-patient conversations using modular summarization techniques. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 4958–4972.

Jean-Baptiste Lamy. 2017. Owlready: Ontology-oriented programming in python with automatic classification and high level constructs for biomedical ontologies. *Artificial intelligence in medicine*, 80:11–28.

Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Veselin Stoyanov, and Luke Zettlemoyer. 2020. Bart: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7871–7880.

Jennifer Liang, Ching-Huei Tsou, and Ananya Poddar. 2019. A novel system for extractive clinical note summarization using EHR data. In *Proceedings of the 2nd Clinical Natural Language Processing Workshop*, pages 46–54, Minneapolis, Minnesota, USA. Association for Computational Linguistics.

Chen Lin, Timothy Miller, Dmitriy Dligach, Steven Bethard, and Guergana Savova. 2021. Entitybert: Entity-centric masking strategy for model pretraining for the clinical domain. In *Proceedings of the 20th Workshop on Biomedical Language Processing*, pages 191–201.

Chin-Yew Lin. 2004. Rouge: A package for automatic evaluation of summaries. In *Text summarization branches out*, pages 74–81.

S Lipsky-Gorman and N Elhadad. 2011. Clinnote and healthtermfinder: a pipeline for processing clinical notes. *Columbia University*.

Fangyu Liu, Ehsan Shareghi, Zaiqiao Meng, Marco Basaldella, and Nigel Collier. 2021. Self-alignment pretraining for biomedical entity representations. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 4228–4238.

Sean MacAvaney, Sajad Sotudeh, Arman Cohan, Nazli Goharian, Ish Talati, and Ross W Filice. 2019. Ontology-aware clinical abstractive summarization. In *Proceedings of the 42nd International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 1013–1016.

Gaur Manas, Vamsi Aribandi, Ugur Kursuncu, Amanuel Alambo, Valerie L Shalin, Krishnaprasad Thirunarayan, Jonathan Beich, Meera Narasimhan, Amit Sheth, et al. 2021. Knowledge-infused abstractive summarization of clinical diagnostic interviews: Framework development study. *JMIR Mental Health*, 8(5):e20865.

Francesco Moramarco, Damir Juric, Aleksandar Savkov, and Ehud Reiter. 2021. Towards objectively evaluating the quality of generated medical summaries. In *Proceedings of the Workshop on Human Evaluation of NLP Systems (HumEval)*, pages 56–61.

Gabriele Pergola, Elena Kochkina, Lin Gui, Maria Liakata, and Yulan He. 2021. Boosting low-resource biomedical qa via entity-aware masking strategies. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*, pages 1977–1985.

Yada Pruksachatkun, Jason Phang, Haokun Liu, Phu Mon Htut, Xiaoyi Zhang, Richard Yuanzhe Pang, Clara Vania, Katharina Kann, and Samuel Bowman. 2020. Intermediate-task transfer learning with pretrained language models: When and why does it work? In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 5231–5247.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J Liu. 2020. Exploring the limits of transfer learning with a unified text-to-text transformer. *Journal of Machine Learning Research*, 21:1–67.

Arpita Roy and Shimei Pan. 2021. Incorporating medical knowledge in bert for clinical relation extraction. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 5357–5366.

Adam Rule, Steven Bedrick, Michael F Chiang, and Michelle R Hribar. 2021. Length and redundancy of outpatient progress notes across a decade at an academic medical center. *JAMA Network Open*, 4(7):e2115334–e2115334.

Richard Saitz, Shannon C Miller, David A Fiellin, and Richard N Rosenthal. 2021. Recommended use of terminology in addiction medicine. *Journal of Addiction Medicine*, 15(1):3–7.

Pedro Saleiro, Benedict Kuester, Loren Hinkson, Jesse London, Abby Stevens, Ari Anisfeld, Kit T Rodolfa, and Rayid Ghani. 2018. Aequitas: A bias and fairness audit toolkit. *arXiv preprint arXiv:1811.05577*.

Abigail See, Peter J Liu, and Christopher D Manning. 2017. Get to the point: Summarization with pointer-generator networks. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1073–1083.

J Shoolin, L Ozeran, C Hamann, and W Bria Ii. 2013. Association of medical directors of information systems consensus on inpatient electronic health record documentation. *Applied clinical informatics*, 4(02):293–303.

Luca Soldaini and Nazli Goharian. Quickumls: a fast, unsupervised approach for medical concept extraction.

Hale M Thompson, Brihat Sharma, Sameer Bhalla, Randy Boley, Connor McCluskey, Dmitriy Dligach, Matthew M Churpek, Niranjan S Karnik, and Majid Afshar. 2021. Bias and fairness assessment of a natural language processing opioid misuse classifier: detection and mitigation of electronic health record data disadvantages across racial subgroups. *Journal of the American Medical Informatics Association*, 28(11):2393–2403.

Amy Y Tsou, Christoph U Lehmann, Jeremy Michel, Ronni Solomon, Lorraine Possanza, and Tejal Gandhi. 2017. Safe practices for copy and paste in the ehr. *Applied clinical informatics*, 26(01):12–34.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. *Advances in neural information processing systems*, 30.

Lawrence L Weed. 1964. Medical records, patient care, and medical education. *Irish Journal of Medical Science (1926-1967)*, 39(6):271–282.

Wen-wai Yim and Meliha Yetisgen-Yildiz. 2021. Towards automating medical scribing: Clinic visit dialogue2note sentence alignment and snippet summarization. In *Proceedings of the Second Workshop on Natural Language Processing for Medical Conversations*, pages 10–20.

Longxiang Zhang, Renato Negrinho, Arindam Ghosh, Vasudevan Jagannathan, Hamid Reza Hassanzadeh, Thomas Schaaf, and Matthew R Gormley. 2021. Leveraging pretrained models for automatic summarization of doctor-patient conversations. In *Findings of the Association for Computational Linguistics: EMNLP 2021*, pages 3693–3712.

Tianyi Zhang*, Varsha Kishore*, Felix Wu*, Kilian Q. Weinberger, and Yoav Artzi. 2020. Bertscore: Evaluating text generation with bert. In *International Conference on Learning Representations*.

Yuhao Zhang, Daisy Yi Ding, Tianpei Qian, Christopher D Manning, and Curtis P Langlotz. 2018. Learning to summarize radiology findings. In *Proceedings of the Ninth International Workshop on Health Text Mining and Information Analysis*, pages 204–213.

A Annotator Training

We recruited two medical students as annotators who received training in medical school curriculum in SOAP note documentation. A three-week orientation and training was conducted by one of the critical care physicians. Each annotation achieved an inter-annotator agreement with a kappa score above 0.80 with the adjudicator. Another round of training was performed on 200 notes and the inter-annotator agreement was measured between annotators and the adjudicator. The annotation was reviewed if the kappa score is below 0.80 threshold.

Input	Sent.θ	Jaccard	Length Diff.
ASSMT	89.00	37.85	6.13 (4.12)
SUMM	83.14	14.43	9.42 (5.99)

Table 4: Quality measurement on augmented input assessment (ASSMT) and reference summary (SUMM). For ever pair of original and augmented sample, we report cosine similarity between text embedding (θ), Jaccard token overlap, and mean and standard deviation (σ) of length difference.

Hyper-parameter	Setting
Optimizer	AdamW
Epoch	10 (with early stopping)
Learning rate	1e-3, 1e-4
Batch size	256
Gradient accumulation	True

Table 5: Hyperparameters for T5 DAPT

B Quality Measure for Data Augmentation

The quality of augmented data directly affected the training process. To ensure a high-quality training corpus, we randomly selected 2,000 pairs of augmented samples. We evaluated how well the meanings were preserved in the augmented sample, and how much lexical variance was introduced into the augmented samples. We reported cosine similarity between the embedding pairs for quality of meanings, and we reported Jaccard similarity for degree of string overlap. Specifically, given a pair of the original sample and the augmented sample, we generated a text embedding through SapBERT (Liu et al., 2021), a BERT encoder pre-trained to represent biomedical entities using UMLS. We expected a high cosine similarity if the augmented samples expressed the same meanings as the original samples. We ran Jaccard similarity by treating the samples as lists of tokens, and expected a low Jaccard score if there were new terms introduced in the augmented samples, e.g. *ARF* and *Acute Renal Failure*. We also reported the mean and standard deviation of the length differences between original and augmented samples (Table 4). On both input assessment and reference summary, the cosine similarity between original and augmented samples was higher than 0.80. Assessment input contained more words that were not biomedical

Input ALCOHOL ABUSE Assesment: This is a 37 year-old male with a history of alcoholism with multiple admissions for detox and history of DTs/withdrawal seizures who presents with intoxication followed by withdrawal.
Reference ETOH intoxication; Elevated lipase

FT-T5 ALCOHOL ABUSE Assesment: This is a 37 year-old male with multiple admissions for detox and history of DTs/withdrawal seizures who presents with intoxication followed by withdrawal.
FT-BART ALCOHOL ABUSE DIAGNOSIS Assesment: This is a 37 year-old male with a history of alcoholism with DTs/withdrawal seizures. He presents with intoxication followed by withdrawal. Alcoholics Anonymous: He is an alcoholic with multiple admissions for alcohol abuse and drug addiction. This male is the only person in the state of Alabama with alcohol dependence.
T5-DAPT-TKS This is a 37 year-old male with history of DTs/withdrawal seizures who presents with intoxication followed by withdrawal
T5-DAPT-CUI history alcoholism history history of dts/withdrawal seizures admissions alcohol abuse history

Figure 8: Given an input assessment, we show the reference summary, example output from fine-tuning T5 and BART, and T5 DAPT with token masking and concept masking. The red fonts show the information that is outside the input text.

Hyper-parameter	Setting
Optimizer	Adam
Epoch	10 (with early stopping)
Learning rate	1e-4, 1e-5, 1e-6
Batch size	4
Task Prefix (t5)	"summarize:"
Encoder max length	512
Decoder max length	128
Beam size	10
Length penalty	1
no repeat ngram size	2

Table 6: Hyperparameters for fine-tuning T5 and BART

concepts; thus, the augmented sample had a greater proportion of overlapping text than the reference summary. Both had more than 6 token differences in length. In conclusion, our proposed strategy of data augmentation successfully produced a high quality training corpus.

C Hyperparameters

Here we report the hyper-parameters we used for T5 DAPT experiments in table 5, and fine-tuning for t5 and BART in table 6. The input length to both T5 and BART is set to 512 tokens. On the training data, the average length of input assessment is 43.33 tokens, and the average length of input and subjective sections is 70.97 tokens. Therefore the maximum encoder length is appropriate for our task.

D Model Example Output

Figure 8 shows the example output from fine-tuning T5 and BART, and T5-DAPT with token masking as well as concept masking policy. T5-DAPT-CUI extracts medical concepts. FT-T5 and T5-DAPT-Tks extract sequence of text from input assessment. FT-BART produces text with information that is not mentioned in the input (red fonts).

Human-in-the-loop Robotic Grasping using BERT Scene Representation

Yaoxian Song[1,2], **Penglei Sun**[1], **Pengfei Fang**[3], **Linyi Yang**[2,4], **Yanghua Xiao**[1], **Yue Zhang**[2,4]*
[1]School of Computer Science, Fudan University
[2]School of Engineering, Westlake University
[3]School of Engineering, Australian National University
[4] Institute of Advanced Technology, Westlake Institute for Advanced Study
{songyaoxian, yanglinyi, zhangyue}@westlake.edu.cn,
{plsun20, shawyh}@fudan.edu.cn, u5765437@anu.edu.au

Abstract

Current NLP techniques have been greatly applied in different domains. In this paper, we propose a human-in-the-loop framework for robotic grasping in cluttered scenes, investigating a language interface to the grasping process, which allows the user to intervene by natural language commands. This framework is constructed on a state-of-the-art grasping baseline, where we substitute a scene-graph representation with a text representation of the scene using BERT. Experiments on both simulation and physical robot show that the proposed method outperforms conventional object-agnostic and scene-graph based methods in the literature. In addition, we find that with human intervention, performance can be significantly improved. Our dataset and code are available on our project website [1].

1 Introduction

Grasping (Mahler et al., 2019) is a fundamental task for robot systems. It is useful for warehousing, manufacturing, medicine, retail, and service robots. One setting in robotic grasping is to grasp object orderly without disturbing the remaining in cluttered scenes (Chen et al., 2021b; Mees and Burgard, 2020; Zhang et al., 2021a) (called **collision-free grasp**). To solve this problem, a typical method (Zhu et al., 2021) parses the input into a **scene graph** first (Figure. 1(b)), in order to infer the spatial relation between objects and select a collision-free object for grasping. In particular, as shown in Figure. 2(a), nodes in a scene graph represent objects and edges represent their spatial relationship. Such structural knowledge can effectively improve the grasping performance as compared to an end-to-end model without scene structure information (Chu et al., 2018) (Figure. 1(a)).

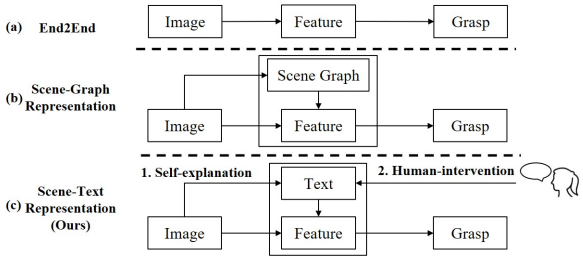

Figure 1: Various model structures for robotic grasping. (a) **End2End** (Chu et al., 2018) outputs an object-agnostic grasping by directly modeling on the input images. (b) **Scene-Graph Representation** fuses a generated scene graph with visual feature to predict grasping. (c) **Scene-Text Representation (ours)** makes use of language scene description and visual feature for achieving collision-free grasping.

As a structured representation of the scene, the scene graph has a few limitations. For example, it can be costly to manually label scene graphs, and the amount of existing labeled data is quite small. In practice, due to variance in the working environment, it can be necessary to calibrate a scene understanding model when deployed on physical robots (Zhu et al., 2021). This problem can be regarded as a domain adaptation task, which requires a certain amount of labeled scene graph data (Xu et al., 2017). In addition, graphs are relatively abstract and thus inconvenient for human-robot interaction.

We consider a natural language representation of the scene for substituting a scene graph structure representation. As shown in Figure. 2(a), small texts such as *"notebook placed under pliers"* and *"apple on notebook"* are used to indicate the recognized objects and their stacking relations. Compared with a scene graph structure, natural language scene representation has the following advantages. First, the cost of manual labeling is relatively lower thanks to the availability of speech recognition systems (Chiu and Raffel, 2018) and the relative independence from labeling GUI (Srivastava et al., 2021). Second, the state-of-the-art pre-trained representation models (Kenton and

* Corresponding Author
[1]https://sites.google.com/view/hitl-grasping-bert

Proceedings of the 29th International Conference on Computational Linguistics, pages 2992–3006
October 12–17, 2022.

Toutanova, 2019; Radford et al., 2019) can be used to improve scene understanding, which contains external knowledge beyond a scene graph structure. Third, online human interaction can be achieved by using human input of the natural language scene representation to replace incorrect robot scene understanding through speech communication[2].

As shown in Figure. 1(c), we adopt the model of Zhu et al. (2021) by substituting the scene graph with a text description of the object to grasp and its spatial context, and using a neural image-to-text model for scene understanding and a pre-trained language model to represent the scene text for visual language grounding in subsequent grasping decisions. We compare the model performance with a dominant two-stage end-to-end planar grasping baseline (Chu et al., 2018) (Figure. 1(a)) and the baseline scene graph model (Figure. 1(b)). For all models, the grasping backbone is implemented using an extended version model of (Chu et al., 2018) with extra scene knowledge input.

For training and evaluation, we make extensions to the Visual Manipulation Relationship Dataset (VMRD) (Zhang et al., 2019b) by manually adding text descriptions and scene graphs to the scenes, resulting in a new dataset **L-VMRD**, as shown in Figure. 2(a). Experimental results show that (1) human language description can be a highly competitive alternative to the scene graph representation, giving better results for grasping; (2) online human language intervention is useful for improving the final grasping results, which is a new form of human-in-the-loop grasping. This indicates the promise of NLP models, especially pre-trained language models, for human-robot interaction. To our knowledge, we are the first to consider explicit textual scene representation and human intervention correction for robot grasping decisions, where BERT (Kenton and Toutanova, 2019) is firstly introduced into the internal structure of a robotic model as a state representation.

2 Task and Dataset

The input of the robotic grasping task is an image from robotic camera observation and the output is a grasping configuration (a grasping bounding box). As shown in Figure.1(c), we introduce a language description for the scene (image) during the inference process. We take a human-in-the-loop setting, where the language description can be obtained from a scene understanding model (**Self-explanation**) or human (**Human-intervention**).

[2]We adopt the **typed text** to simulate the process here since voice recognition is beyond our research scope

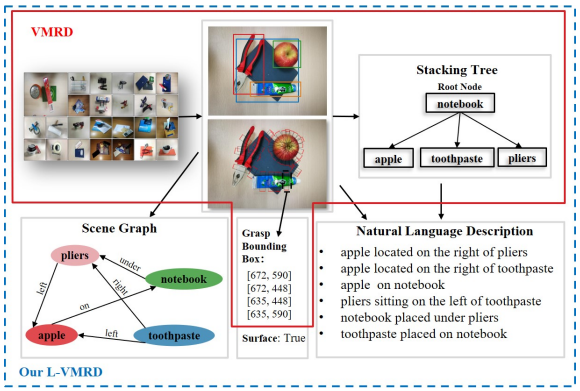

(a) Overview of proposed dataset L-VMRD

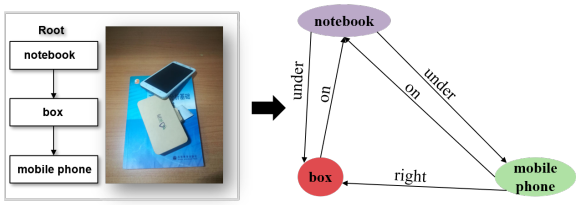

(b) Relationship tree & Scene graph.

Figure 2: **(a):** L-VMRD is built on VMRD. We extend **(i)** scene language description, **(ii)** scene graph and **(iii)** **surface** per grasp, including $112,965$ scene object relationship expressions and $21,713$ **surface** attributes paired with grasp bounding boxes. **(b):** relationship tree vs. scene graph.

Existing grasping datasets (e.g., VMRD Zhang et al. (2019b)) cannot be employed directly, because they do not include scene knowledge in human language. Hence, we develop an extended language version of VMRD, named L-VMRD, to evaluate our method. L-VMRD is an integrated dataset, and each sample is organized as a 6-tuple **(image, language descriptions, scene graph, object bounding box, grasping bounding box, surface)** shown in Figure. 2. L-VMRD contains $4,676$ samples, and is split into (train/validate/test) $= (3,740/468/468)$. The detail of dataset generation and usage in modeling are demonstrated in Appendix A.1.1 and A.1.2. Below we describe the main extensions from VMRD (Figure.2(a)).

Language Description in L-VMRD Object pairs in an image are sampled and labeled with a scene description. There exist many factors that affect collision-free grasping, in which stacking is the most significant one (Avigal et al., 2021). We first label the objects with the stacking relationship, e.g., *"apple on notebook"* or *"notebook sitting under pliers"* in Figure. 2(a), and then label

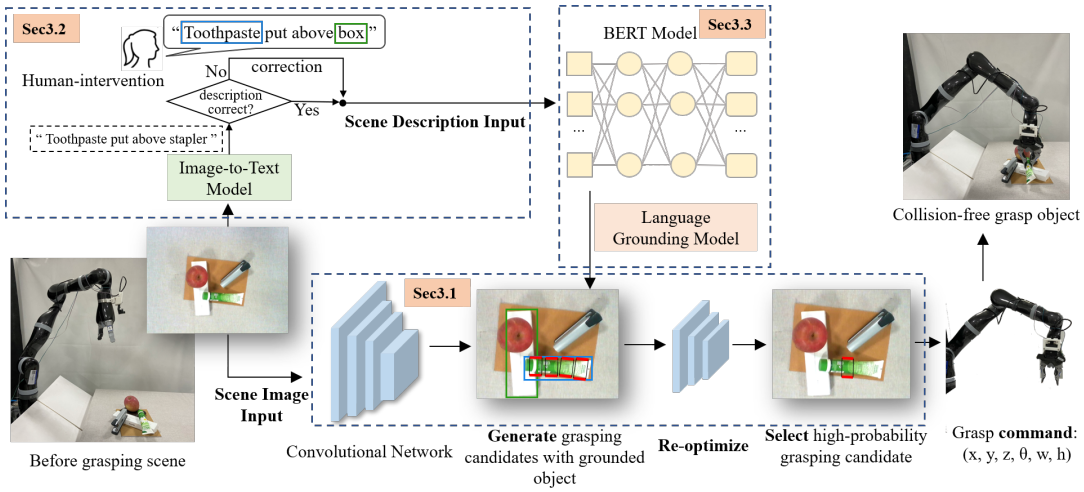

Figure 3: Overview of our proposed model. Firstly, a scene image is given to a human and an image-to-text model to generate a scene description. Secondly, scene description is fed into a BERT-based language grounding model to select a collision-free object. Thirdly, the grounded object is as the internal result fused into the language-based grasping model.

the horizontal relationship between non-stacked. Then considering the distribution of the horizontal relationships (*"left"*, *"right"*, *"front"*, *"back"*) in our dataset, we use ***"left"*** and ***"right"*** to indicate the scene relationships.

Scene Graph The original VMRD dataset includes partial relationships for adjacent objects, encoded by a relationship tree that only reveals the stacking relationship between two objects but is not able to indicate the relationship between objects stacked directly. To facilitate inference, we add a full scene graph to encode the pair-wise relationships of objects per image, where nodes and edges present objects and relationships, respectively. Detail is available in Appendix A.1.3.

Grasp-wise Spatial Attribute We introduce each grasp bounding box with a new attribute named **surface**. It is a binary variable indicating whether the grasped object sits on the top (True – on the top, False – stacked by other objects.). It is a grasp-wise label and can improve the robustness of the grasping model. An example is available in Appendix A.1.4.

3 Our Approach

The overall framework is shown in Figure. 3. The input scene image is fed into convolutional-based grasping model (Sec. 3.1) and scene understanding model (Sec. 3.2; Sec. 3.3), respectively. The scene understanding model is an image-to-text component that produces a sentence that describes the object to grasp and its context, such as *"toothpaste put above box"*. That scene description is then fused with intermediate grasping results to select object-related grasping candidates by pre-trained language model and language grounding model. **The final grasping output** is selected with high probability after re-optimization. A grasp command is sent to a real robot to complete a collision-free grasping operation. For the human-in-the-loop scenario, **an extra conditional input** from human-intervention will be given to correct scene description when the description from the image-to-text model is incorrect.

3.1 Overall of Language-based Grasping Model

Let I denote an image as perceptive information from the environment (i.e., cluttered scenes). Our robot f first identifies the grasp configuration from observation I. A typical 5-dimensional grasp configuration is given by:

$$g_i = f(I) = (x, y, \theta, w, h), \tag{1}$$

where (x, y) is the center position of the grasp rectangle, θ is the orientation angle with the x-axis, and (w, h) is the weight and height of the grasp rectangle. A general robotic grasping is presented by a probability $P(g_i|I)$, where $g_i \in G$ and G is a set of grasping candidates.

To achieve more stable and safe grasping, a joint probability $P(g_i, K_g|I)$ can integrate additional scene knowledge K_g as auxiliary information to guide vision-based grasping. It can be decomposed into conditionally independent two parts, given by:

$$P\left(g_i, K_g|I\right) = P\left(g_i|I, K_g\right) P\left(K_g|I\right), \tag{2}$$

where $P(K_g|I)$ is a scene understanding model. It can be a scene structure parsing model (Zhu et al., 2021; Figure. 1(b)), an image-to-text model (Figure. 1(c); Sec.3.2) with grounding model (Sec.3.3) or direct human intervention with grounding model (Sec.3.3). $P\left(g_i|I, K_g\right)$ is a convolutional network and the details are described in Appendix A.2.

3.2 Grasping Scene Understanding

A state-of-the-art image-to-text model (MMT) (Cornia et al., 2020) is used to generate the scene description in our work, which is a standard encoder-decoder Transformer-based model (Vaswani et al., 2017), that learns a multi-level representation of the relationships between image regions integrating with learned prior knowledge, and uses a mesh-like connectivity at decoding stage to exploit low- and high-level features. More details are in Appendix A.3.

Encoder A set of image regions I as Input is fed into encoding layer following Eq. (3).

$$\begin{aligned} Z &= AddNorm\left(M_{mem}\left(I\right)\right), \\ \tilde{X} &= AddNorm\left(F\left(Z\right)\right), \end{aligned} \tag{3}$$

where $AddNorm$ indicates the composition of a residual connection and layer normalization. M_{mem} is memory-augmented attention operation in Eq. (4). $F\left(Z\right)$ is a position-wise feed-forward layer composed of two affine transformations with a single non-linearity. $\tilde{X} = (\tilde{X}^1, ..., \tilde{X}^i..., \tilde{X}^N)$ is the set of all encoding layers and N is the number of layers.

$$\begin{aligned} M_{en}\left(I\right) &= Attention\left(W_q I, K, V\right), \\ K &= \left[W_k I, M_k\right], \\ V &= \left[W_v I, M_v\right], \end{aligned} \tag{4}$$

where $Attention$ is the self-attention operations used in (Vaswani et al., 2017). W_q, W_k, W_v are matrices of learnable weights. M_k and M_v are learnable prior information.

Decoder The decoder takes an input sequence of vector Y and output layers from encoder $\tilde{X}$, and then outputs sequence $\tilde{Y}$, in Eq. 5.

$$\begin{aligned} Z &= AddNorm\left(M_{de}\left(\tilde{X}\right), Y\right), \\ \tilde{Y} &= AddNorm\left(F\left(Z\right)\right), \end{aligned} \tag{5}$$

where M_{de} is defined in Eq. 6. Y is the input sequence of vector (groundtruth).

$$\begin{aligned} M_{de}\left(\tilde{X}, Y\right) &= \sum_{i=1}^{N} \alpha_i \odot C\left(\tilde{X}^i, Y\right), \\ C\left(\tilde{X}^i, Y\right) &= Attention\left(W_q Y, W_k \tilde{X}^i, W_v \tilde{X}^i\right), \\ \alpha_i &= \sigma\left(W_i\left[Y, C\left(\tilde{X}^i, Y\right)\right] + b_i\right), \end{aligned} \tag{6}$$

where C is cross-attention operation, σ the sigmoid activation function, and $\odot$ element product.

We make use of two adaptations during training. The first replaces the region features in MMT[3] (Cornia et al., 2020) with the concatenation of region features with bounding box features. Secondly, we add an extra score to multiply the CIDEr-D reward (Rennie et al., 2017) during training by maximizing a reinforcement learning based reward, since the description of subject object is usually the grasped one in our task. The score is computed by the correct rate of the subject over all generated sentences for each training batch.

3.3 Language Grounding for Grasping

We make use of visual language grounding models to map a scene description to a specified object. For visual language grounding, let Q represent a query sentence from human or image-to-text model and $I \in R^{H \times W \times 3}$ denote the image of width W and height H. The task aims to find the object region K_g represented by its center point (x_t, y_t) and the object size (w_t, h_t). The overall method can be formulated as a mapping function $(x_t, y_t, w_t, h_t) = \phi(Q, I)$.

In this paper, considering the real-time robotic control, we deploy our task on a one-stage language grounding model[4] (Yang et al., 2019) based on YOLOv3[5] (Redmon and Farhadi, 2018) with different language encoders for the mapping function ϕ. Formally, the scene image I and scene description Q are input to the visual encoder and text encoder, respectively, and the grounding module outputs the grounded object with encoders' features following Eq. 7.

$$\begin{aligned} Z_{vis} &= M_{vis}\left(I\right), \\ Z_{lang} &= M_{lang}\left(Q\right), \\ K_g &= Ground\left(\left[Z_{vis}, Z_{lang}, Z_{spatial}\right]\right), \end{aligned} \tag{7}$$

where M_{vis} is Darknet-53 (Redmon and Farhadi, 2018) pre-trained on COCO object dataset (Lin et al., 2014) and fine-tuned on our proposed L-VMRD. $Ground$ is the same as the output layers of YOLOv3. $Z_{spatial}$ is spatial feature of visual feature defined as follows: $(\frac{i}{W}, \frac{j}{H}, \frac{i+0.5}{W}, \frac{j+0.5}{H}, \frac{i+1}{W}, \frac{j+1}{H}, \frac{1}{W}, \frac{1}{H})$, which denotes top-left, center, bottom-right plane coordinates, sizes of the each pixel in visual feature

[3] https://github.com/aimagelab/meshed-memory-transformer

[4] https://github.com/zyang-ur/onestage_grounding

[5] YOLOv3 is a typical object detection model and derives many multimodal variants.

mapping Z_{vis}, respectively, normalized by the width W and height H of the feature mapping. $i \in \{0, 1, ..., W-1\}$ and $j \in \{0, 1, ..., H-1\}$.

For M_{lang} of the text encoder, we choose BERT (Kenton and Toutanova, 2019)[6] and the encoder of Transformer (Vaswani et al., 2017) (simply named Transformer). BERT is a pre-trained language model and builds on the Transformer network. Each description is fed into M_{lang}, resulting in 768 dimensions embeddings of all the tokens as natural language representations. Transformer[7] can be regarded as randomly initialized BERT without pre-training. Each description is embedded into $1,024$ dimensions embeddings. The model is randomly initialized.

3.4 Training Details

We break an end-to-end grasping training process into three submodules, **(i)** image-to-text (**self-explanation**), **(ii)** language grounding, and **(iii)** language-based grasping successively. The first two models are trained based on the original project configurations. The detail of the last one is described in Appendix A.4.

4 Evaluation

We construct both simulation and physical experiments to investigate four research questions:

Q1: How much natural language (**unstructured**) scene description perform better than scene graph (**structured**) knowledge in collision-free grasping task?

Q2: How much does the pre-trained model perform better than the randomly initialized model in our task?

Q3: How and where does human intervention using NLP improve grasping performance under the proposed human-in-the-loop framework?

Q4: How does our method perform on a physical robot? Does data collection from our proposed human-in-the-loop framework improve efficiency during the fine-tuning process?

4.1 Settings

Scene and Platform Implementation In simulation experiments, we use the test set split in Sec. 2 to evaluate our method. In the physical experiment, we collect objects and set up the placement as similar as possible to the training set, in which 2-6 objects are randomly placed (stacked) on a white table. But the reality gap is usually inevitable, especially in a robotic environment, which also tests the robustness of our method implicitly. Other details of the robot and training framework are described in Appendix A.5.

Evaluation Metrics We take Benchmark Performance and Success Rate as the main evaluation metrics. The first is used in both simulation and physical experiments, and the second one is only used in the physical robot experiment:

• *Benchmark Performance*: In simulation experiment, we evaluate model performances of the collision-free grasp using the object retrieval *top-k recall (R@k)* and *top-k precision (P@k)* metrics to evaluate multi-grasp detection (Hu et al., 2016). Chen et al. (2021b) proposes above metric to evaluate language-based multi-grasping. We do not compare it with our work directly, because: (i) their work (including dataset) is not open-sourced. (ii) it is just a command-based end-to-end grasping method that did not consider language scene understanding with human-in-the-loop. In physical robot experiment, ***accuracy*** is the percentage of correct cases over all test cases. The correct case is defined in Appendix A.6.

• *Success Rate*: In the physical robot experiment, we calculate the percentage of successful collision-free grasps over all grasping trials.

4.2 Models

We compare different pipeline methods visualized in Figure. 1. The baselines include:

End2End We re-train a state-of-the-art end-to-end object-agnostic planar grasp detection model MultiGrasp (Chu et al., 2018) on L-VMRD, shown in Figure. 1(a).

SceneGraph-Rep This is shown in Figure. 1(b) using a structured form of the scene graph generation[8] (IMP) (Xu et al., 2017) encoded with relational graph convolution network (RGCN) (Schlichtkrull et al., 2018) shown in Figure. 3. It replaces the subprocess from Image-to-Text Model to Language Grounding Model in Figure. 3 to select the grouned object. See details in Appendix A.7.1.

Our proposed models (**Scene-Text Representation** in Figure. 1 (c)) include:

SceneText-{BERT, Transformer} They are models using image-to-text MMT (Cornia et al., 2020) with language grounding (Yang et al., 2019) to re-

[6] We use bert-base-uncased model in our work.

[7] https://github.com/pytorch/examples/tree/master/word_language_model (**6 encoder_layers implemented**)

[8] https://github.com/jwyang/graph-rcnn.pytorch

Method	R@1	R@3	R@5	R@10	P@1	P@3	P@5	P@10
End2End	42.5	66.2	78.9	88.9	42.5	40.5	40.1	37.5
SceneGraph-Rep	72.2	85.6	90.3	92.2	73.8	70.7	69.0	64.0
SceneText-Transformer	72.2	85.6	88.3	90.3	72.4	68.8	66.2	61.6
SceneText-BERT	**73.7**	**88.9**	**91.8**	**93.1**	**73.9**	**71.9**	**69.2**	**65.3**
With Human-intervention								
SceneText-Interv-Oracle	77.0	89.1	90.8	91.4	78.0	76.4	75.0	71.7
SceneText-Interv-Transformer	75.9	88.8	91.8	92.4	76.3	71.2	69.2	64.9
SceneText-Interv-BERT	**76.9**	**90.3**	**93.0**	**94.9**	**76.3**	**75.3**	**73.6**	**69.1**

Table 1: Results of self-explanation and human-intervention models in the simulation experiment. The best performance is highlighted in bold.

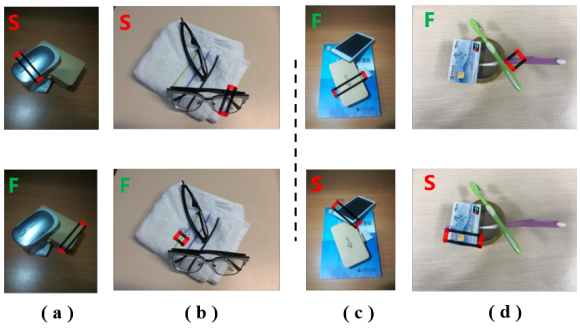

(a) S SceneText-*: mouse placed on box F: End2End: -
(b) S SceneText-*: glasses placed on towel F: End2End : -
(c) F SceneText-*: box placed on mobile phone
S SceneText-Interv-*: mobile phone located on box
(d) F: SceneText-*: toothbrush sitting left of toothbrush
S SceneText-Interv-*: card placed on the left of toothbrush

Figure 4: Visualization of baseline models and our proposed models in our work. S means a successful case, while F means a failure case.

alize explainable grasping (**Self-explanation w/o Human-intervention**).

SceneText-Interv-{BERT, Transformer} They bring in human-intervention shown in Figure. 3. In **SceneText-Interv-Oracle**, the retrieval region of language grounding from the groundtruth is fed directly into the downstream grasp model (**Self-explanation w/ Human-intervention**).

4.3 Simulation Results

Scene Knowledge Table 1 gives the results of the end-to-end, scene-graph based method, and our method on L-VMRD data. In this setting, the input of the model is only the image and models predict a collision-free grasping. Although End2End (Chu et al., 2018) is the state-of-the-art method in object-agnostic grasping, it performs poorly on the cluttered scene grasping tasks. In contrast, by first obtaining a scene graph and then predicting the selected object, the SceneGraph-Rep model gives much improvement, with 72.2% over End2End R@1. This shows the necessity of scene understanding for the collision-free grasping task. As a case study, in Figure. 4(a)(b), SceneText-* generate self-explanation expression and obtain correct collision-free grasping, while End2End predicts incorrect grasping (incorrect object selection and low-quality grasp detection).

Scene Graph vs. Natural Language For **Q1**, compared with SceneGraph-Rep model which has R@1 of 72.2%, our method SceneText-BERT gives a better R@1 of 73.7%. This shows the feasibility of using natural language to replace the scene graph for object selection. Both methods are trained under the same settings, yet a natural language is more useful for achieving expandability in real-time human-robot interaction, thanks to its direct connection to the natural representation of the scene. For **Q2**, among our models, BERT can better parse the generated scene descriptions than Transformer model in both P@k and R@k, which shows the benefit of external pre-training. Note that Transformer alone does not outperform SceneGraph-Rep in Table 1. The results show that pre-training allows a textual representation of the scene to compete with a standard graph representation. More case studies can be found in Appendix A.7.2.

Human-intervention For **Q3**, the last two rows in Table 1 show that the models can take human language descriptions about the cluttered scene as a guidance or error correction for its own textual scene representation. By comparing results in Table 1, we can find that our proposed human-in-the-loop framework improves the performance from SceneText-* to SceneText-Interv-*. For example, compared with SceneText-BERT, SceneText-Interv-BERT improves the R@1 value from 73.7% to 76.9%, which shows that human intervention can be useful in practical scenarios. As a case study shown in Figure. 4(c)(d), the image-to-text model generates an incorrect relationship between *"box"* and *"mobile phone"*, leading to a failed collision-free grasping detection on the stacked box. In contrast, an extra human scene language description

Inter(%)	0	25	50	75	100	baseline (L-VMRD)
MMT	44.6	45.4	49.2	51.7	**55.0**	12.5
LangGr	91.3	91.7	90.8	92.1	**92.9**	79.6

Table 2: Results of image-to-text accuracy and language grounding accuracy @0.5 in the physical experiment. **Inter-** for short of intervention rate, e.g., **Inter-100**.

corrects the self-explanation description and helps the model to obtain a correct collision-free grasping detection. Within the above process, 21.1% scene language description from a robot is incorrect and intervened by human-correction. We also present the results of different human-intervention rates on F1 score, between SceneText-BERT and SceneText-Interv-BERT. As shown in Figure. 6(a), with the increase of human-intervention, the F1 score improves steadily. This shows that human language intervention can compensate for the flaws of scene understanding from the image-to-text models effectively.

4.4 Physical Robot Results

We only investigate the performance of BERT-based models (SceneText-BERT, SceneText-Interv-BERT) in the physical robot experiment. We recruit five graduate students to participate in our real-world experiment, who observe one whole process of a cluttered grasping with human-like grasping object selection each time by turns. When the image-to-text (self-explanation) model outputs an erroneous description, they correct the scene description by text typing decisively. Ultimately we collect 400 samples, including 160 correct output samples without human-intervention (self-explanation) and 240 human corrected (intervention) samples using the model trained on L-VMRD. Each sample contains an image, a language description sentence, and a grounded object bounding box. A pipeline case containing human-intervention is shown in Figure. 5.

Human-in-the-loop Learning Deployment Inspired by Lu et al. (2022), we take 160 samples as the training set to fine-tune our model based on different intervention rates (proportion of human-intervention samples, **Inter** for short). We randomly sample 80 samples from the remaining 240 human-intervention samples (getting rid of training used) as our test set. We repeat the process three times to generate three different test sets. The results in Table 2 show the mean values of models test on three test sets. For **Q4**, Table 2 shows the Benchmark Performance of models fine-tuned on different intervention rate training sets. The values of baseline are from the image-to-text model (**MMT**) and language grounding model (**LangGr**) trained on L-VMRD, respectively, which show the necessity of domain adaptation. For both MMT and LangGr, the models fine-tuned on human-intervention data (intervention rate 100%) achieve the best performance (55.0% and 92.9%). For MMT, the best model improves up to **42.5**% compared to baseline (12.5%) and **10.4**% compared to the model fine-tuned on self-explanation (44.6%, intervention rate 0%). For LangGr, the best model improves up to **13.3**% compared to baseline (79.6%) and **1.6**% compared to the model fine-tuned on self-explanation data (91.3%). This shows that the data collected from human-intervention can achieve better performance in domain adaptation from simulation to physical environment.

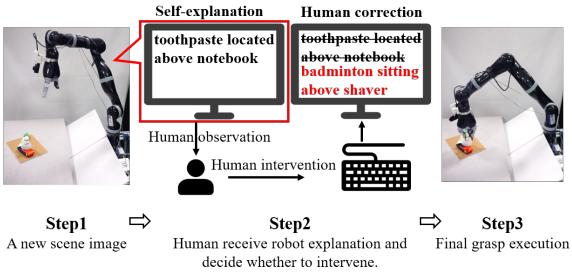

Figure 5: A pipeline of real robot execution with human.

Evaluation on Physical Robot For a final performance test, we conduct extra 80 grasping trials[9] for each model settings corresponding to Table 2. Grasping performance execution by a physical robot is shown in Figure. 6(b). End2End is the result from our baseline model trained on L-VMRD in Figure. 1(a). MMT and MMT+LangGr are models fine-tuning **image-to-text** or both **image-to-text and language grounding** respectively in SceneText-BERT setting. **+human** adds extra human intervention (SceneText-Interv-BERT) with Inter-100 fine-tuning model during grasping.

As shown in Figure. 6(b), our proposed method, which fine-tunes on human-intervention collection data, achieves 67.9% and 75.6% success rate compared to our baseline (63.8%). For **Q3** and **Q4**, our proposed methods achieve 71.8%(↑ 3.9%) and 80.8%(↑ 5.2%) success rate with human-intervention (**+human**) compared to without human-intervention (67.9%, 75.6%), in which

[9]We keep 80 object placements for each setting consistent.

46.3% scene language description from a robot is incorrect and intervened by human-intervention. This shows that human language intervention can improve the performance of grasping online on a real robot compared to the only image-to-text (self-explanation) method. Moreover, results from **MMT** and **MMT+LangGr** show that using human-intervention samples to train models can achieve better performance when there are very few to fine-tune the model. This indicates our proposed human-in-the-loop framework is applicable and performs well on the physical robot.

5 Related Work

Natural Language and Robotics Natural language has been used with a variety of robot platforms, ranging from manipulators to mobile robots to aerial robots (Ahn et al., 2022; Raychaudhuri et al., 2021; Thomason et al., 2016; Chen et al., 2021a; Scalise et al., 2019). Most existing work is related to language understanding and language generation problems. For human-to-robot, language grounding is the mainstream means to learn the connection between percepts and actions in visual language navigation (Anderson et al., 2018; Ku et al., 2020) and robotic grasping tasks (Can et al., 2019; Zellers et al., 2021; Wang et al., 2021). For robot-to-human, multi-modal natural language generation (NLG) is widely adopted to lessen the communication barriers between humans and robots (Vinyals et al., 2015; Li et al., 2020; Cornia et al., 2020; Yuan et al., 2020; Shi et al., 2021; Zhang et al., 2021b), converting non-verbal data to the language that human can understand (Singh and JV, 2020). For bidirectional human-robot, Yuan et al. (2022) propose an explainable artificial intelligence system in which a group of robots predicts users' values by taking in situ feedback into consideration while communicating their decision processes to users through explanations. Our work is in line with the above work in exploiting information from natural language to facilitate decision-making. This is important because natural language is the most intuitive means of human-robot interaction. However, from the aspect of language, the main difference from existing work is that we take a step forward to not only considering language scene description as *input* for a robot but also as an *interface* of the model for online self-explanation simultaneously.

Grasping in Cluttered Scene Conventional methods (Chu et al., 2018; Mahler et al., 2019; Morrison et al., 2020; Kumra et al., 2020) focus more on object-agnostic grasp points detection (Figure. 1 (a)) missing parsing the object stacking scenario. For cluttered grasping, scene understanding and human instruction are usually considered. Zhu et al. (2021); Zhang et al. (2019a) adopt structured scene understanding (e.g., scene graph or relationship tree) to realize cluttered grasping detection. Mees and Burgard (2020); Chen et al. (2021b) fuse a natural language command and an observation image to detect a grasping in a two-stage and an end-to-end manner, respectively. Shridhar and Hsu (2020); Zhang et al. (2021a) receive natural command and image input, and then grasp a specified object. Existing work exploits human language to specify an object from the clutter, but does not allow human intervention for error correction. While existing method take language as *external* input, our scene language description is an *internal* component of the model. Moreover, we adopt the pre-train language model instead of RNN models in existing work.

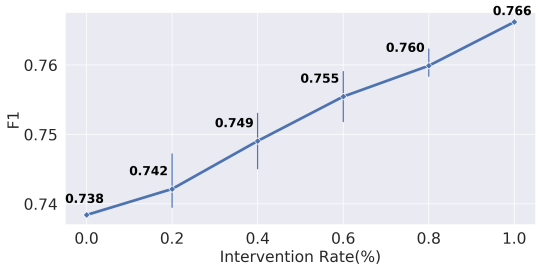

(a) F1 score on different intervention rates.

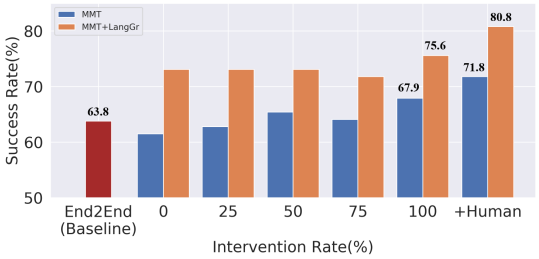

(b) Results of Grasping Success Rate in physical experiment.

Figure 6: Evaluation results.

6 Conclusion

We investigate language scene representation to robotic grasping, which enables a robot to explain its object selection to the user and allows the user to intervene with the selection by natural language. Experiments show that the proposed explainable textual scene representation outperforms

both object-agnostic and scene-graph based methods. By human language intervention, the performance can be broadly increased. Our results indicate the promise of using NLP models in a robotic system both as a representation and for human intervention. To our knowledge, we are the first to consider textual scene encoding and human correction in robotic grasping tasks, which can improve grasping performance using natural language (vs. w/o human-in-the-loop) and robustness (by pretrained language model).

7 Ethical Statement

Five graduate students who studied electronic engineering are hired to cooperate with a collaborative robot (Kinova) in our real-world experiment. Because of the subject background, they can be easy to decide whether give human language intervention based on human-like grasping behavior each turn. The participants need to annotate the object bounding box for each sample during the data collection stage. All participants have received labor free corresponding to their amount of trials.

Acknowledgements

We would like to thank anonymous reviewers for their insightful comments and suggestions to help improve the paper. This publication has emanated from research conducted with the financial support of the Pioneer and "Leading Goose" R&D Program of Zhejiang under Grant Number 2022SDXHDX0003, Shanghai Science and Technology Innovation Action Plan under Grant Number 19511120400, and Hangzhou City Agriculture and Social Development General Project under Grant Number 20201203B118. Yue Zhang is the corresponding author.

References

Michael Ahn, Anthony Brohan, Noah Brown, Yevgen Chebotar, Omar Cortes, Byron David, Chelsea Finn, Keerthana Gopalakrishnan, Karol Hausman, Alex Herzog, et al. 2022. Do as i can, not as i say: Grounding language in robotic affordances. *arXiv preprint arXiv:2204.01691*.

Peter Anderson, Qi Wu, Damien Teney, Jake Bruce, Mark Johnson, Niko Sünderhauf, Ian Reid, Stephen Gould, and Anton Van Den Hengel. 2018. Vision-and-language navigation: Interpreting visually-grounded navigation instructions in real environments. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3674–3683.

Yahav Avigal, Vishal Satish, Zachary Tam, Huang Huang, Harry Zhang, Michael Danielczuk, Jeffrey Ichnowski, and Ken Goldberg. 2021. Avplug: Approach vector planning for unicontact grasping amid clutter. In *2021 IEEE 17th International Conference on Automation Science and Engineering (CASE)*, pages 1140–1147. IEEE.

Ozan Arkan Can, Pedro Zuidberg Dos Martires, Andreas Persson, Julian Gaal, Amy Loutfi, Luc De Raedt, Deniz Yuret, and Alessandro Saffiotti. 2019. Learning from implicit information in natural language instructions for robotic manipulations. *NAACL HLT 2019*, page 29.

Feilong Chen, Xiuyi Chen, Can Xu, and Daxin Jiang. 2021a. Learning to ground visual objects for visual dialog. *arXiv preprint arXiv:2109.06013*.

Yiye Chen, Ruinian Xu, Yunzhi Lin, and Patricio A Vela. 2021b. A joint network for grasp detection conditioned on natural language commands. In *2021 IEEE International Conference on Robotics and Automation (ICRA)*, pages 4576–4582. IEEE.

Chung-Cheng Chiu and Colin Raffel. 2018. Monotonic chunkwise attention. In *International Conference on Learning Representations*.

Fu-Jen Chu, Ruinian Xu, and Patricio A Vela. 2018. Real-world multiobject, multigrasp detection. *IEEE Robotics and Automation Letters*, 3(4):3355–3362.

Marcella Cornia, Matteo Stefanini, Lorenzo Baraldi, and Rita Cucchiara. 2020. Meshed-memory transformer for image captioning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 10578–10587.

Ronghang Hu, Huazhe Xu, Marcus Rohrbach, Jiashi Feng, Kate Saenko, and Trevor Darrell. 2016. Natural language object retrieval. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 4555–4564.

Jacob Devlin Ming-Wei Chang Kenton and Lee Kristina Toutanova. 2019. Bert: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of NAACL-HLT*, pages 4171–4186.

Alexander Ku, Peter Anderson, Roma Patel, Eugene Ie, and Jason Baldridge. 2020. Room-across-room: Multilingual vision-and-language navigation with dense spatiotemporal grounding. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 4392–4412.

Sulabh Kumra, Shirin Joshi, and Ferat Sahin. 2020. Antipodal robotic grasping using generative residual convolutional neural network. In *2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 9626–9633. IEEE.

Xiujun Li, Xi Yin, Chunyuan Li, Pengchuan Zhang, Xiaowei Hu, Lei Zhang, Lijuan Wang, Houdong Hu, Li Dong, Furu Wei, et al. 2020. Oscar: Object-semantics aligned pre-training for vision-language tasks. In *European Conference on Computer Vision*, pages 121–137. Springer.

Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C Lawrence Zitnick. 2014. Microsoft coco: Common objects in context. In *European conference on computer vision*, pages 740–755. Springer.

Jinghui Lu, Linyi Yang, Brian Namee, and Yue Zhang. 2022. A rationale-centric framework for human-in-the-loop machine learning. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 6986–6996.

Jeffrey Mahler, Matthew Matl, Vishal Satish, Michael Danielczuk, Bill DeRose, Stephen McKinley, and Ken Goldberg. 2019. Learning ambidextrous robot grasping policies. *Science Robotics*, 4(26).

Oier Mees and Wolfram Burgard. 2020. Composing pick-and-place tasks by grounding language. In *International Symposium on Experimental Robotics*, pages 491–501. Springer.

Douglas Morrison, Peter Corke, and Jürgen Leitner. 2020. Learning robust, real-time, reactive robotic grasping. *The International journal of robotics research*, 39(2-3):183–201.

Alec Radford, Jeff Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. 2019. Language models are unsupervised multitask learners.

Sonia Raychaudhuri, Saim Wani, Shivansh Patel, Unnat Jain, and Angel X Chang. 2021. Language-aligned waypoint (law) supervision for vision-and-language navigation in continuous environments. *arXiv preprint arXiv:2109.15207*.

Joseph Redmon and Ali Farhadi. 2018. Yolov3: An incremental improvement. *arXiv preprint arXiv:1804.02767*.

Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun. 2016. Faster r-cnn: towards real-time object detection with region proposal networks. *IEEE transactions on pattern analysis and machine intelligence*, 39(6):1137–1149.

Steven J Rennie, Etienne Marcheret, Youssef Mroueh, Jerret Ross, and Vaibhava Goel. 2017. Self-critical sequence training for image captioning. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 7008–7024.

Rosario Scalise, Jesse Thomason, Yonatan Bisk, and Siddhartha Srinivasa. 2019. Improving robot success detection using static object data. In *2019 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 4229–4235. IEEE.

Michael Schlichtkrull, Thomas N Kipf, Peter Bloem, Rianne Van Den Berg, Ivan Titov, and Max Welling. 2018. Modeling relational data with graph convolutional networks. In *European semantic web conference*, pages 593–607. Springer.

Zhan Shi, Hui Liu, Martin Renqiang Min, Christopher Malon, Li Erran Li, and Xiaodan Zhu. 2021. Retrieval, analogy, and composition: A framework for compositional generalization in image captioning. In *Findings of the Association for Computational Linguistics: EMNLP 2021*, pages 1990–2000.

Mohit Shridhar and David Hsu. 2020. Interactive visual grounding of referring expressions for human-robot interaction. *The International journal of robotics research*, 39(2-3):217–232.

Yuvaram Singh and Kameshwar Rao JV. 2020. Ai sensing for robotics using deep learning based visual and language modeling. In *Second Grand-Challenge and Workshop on Multimodal Language (Challenge-HML)*, pages 60–63.

Sanjana Srivastava, Chengshu Li, Michael Lingelbach, Roberto Martín-Martín, Fei Xia, Kent Elliott Vainio, Zheng Lian, Cem Gokmen, Shyamal Buch, Karen Liu, et al. 2021. Behavior: Benchmark for everyday household activities in virtual, interactive, and ecological environments. In *5th Annual Conference on Robot Learning*.

Jesse Thomason, Jivko Sinapov, Maxwell Svetlik, Peter Stone, and Raymond J Mooney. 2016. Learning multi-modal grounded linguistic semantics by playing" i spy". In *IJCAI*, pages 3477–3483.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in neural information processing systems*, pages 5998–6008.

Oriol Vinyals, Alexander Toshev, Samy Bengio, and Dumitru Erhan. 2015. Show and tell: A neural image caption generator. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 3156–3164.

Ke-Jyun Wang, Yun-Hsuan Liu, Hung-Ting Su, Jen-Wei Wang, Yu-Siang Wang, Winston Hsu, and Wen-Chin Chen. 2021. Ocid-ref: A 3d robotic dataset with embodied language for clutter scene grounding. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 5333–5338.

Danfei Xu, Yuke Zhu, Christopher B Choy, and Li Fei-Fei. 2017. Scene graph generation by iterative message passing. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 5410–5419.

Zhengyuan Yang, Boqing Gong, Liwei Wang, Wenbing Huang, Dong Yu, and Jiebo Luo. 2019. A fast and accurate one-stage approach to visual grounding. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 4683–4693.

Chenxi Yuan, Yang Bai, and Chun Yuan. 2020. Bridge the gap: High-level semantic planning for image captioning. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 3157–3167.

Luyao Yuan, Xiaofeng Gao, Zilong Zheng, Mark Edmonds, Ying Nian Wu, Federico Rossano, Hongjing Lu, Yixin Zhu, and Song-Chun Zhu. 2022. In situ bidirectional human-robot value alignment. *Science Robotics*, 7(68):eabm4183.

Rowan Zellers, Ari Holtzman, Matthew Peters, Roozbeh Mottaghi, Aniruddha Kembhavi, Ali Farhadi, and Yejin Choi. 2021. Piglet: Language grounding through neuro-symbolic interaction in a 3d world. *arXiv preprint arXiv:2106.00188*.

Hanbo Zhang, Xuguang Lan, Site Bai, Lipeng Wan, Chenjie Yang, and Nanning Zheng. 2019a. A multi-task convolutional neural network for autonomous robotic grasping in object stacking scenes. In *2019 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 6435–6442. IEEE.

Hanbo Zhang, Xuguang Lan, Site Bai, Xinwen Zhou, Zhiqiang Tian, and Nanning Zheng. 2019b. Roi-based robotic grasp detection for object overlapping scenes. In *2019 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 4768–4775. IEEE.

Hanbo Zhang, Yunfan Lu, Cunjun Yu, David Hsu, Xuguang La, and Nanning Zheng. 2021a. Invigorate: Interactive visual grounding and grasping in clutter. In *Robotics: Science and Systems (RSS)*.

Pengchuan Zhang, Xiujun Li, Xiaowei Hu, Jianwei Yang, Lei Zhang, Lijuan Wang, Yejin Choi, and Jianfeng Gao. 2021b. Vinvl: Revisiting visual representations in vision-language models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 5579–5588.

Yifeng Zhu, Jonathan Tremblay, Stan Birchfield, and Yuke Zhu. 2021. Hierarchical planning for long-horizon manipulation with geometric and symbolic scene graphs. In *2021 International Conference on Robotics and Automation (ICRA)*, pages 6541–6548. IEEE.

A Appendix

A.1 Details of Dataset

A.1.1 Dataset Generation

VMRD contains $4,683$ samples for 31 classes originally. Each sample has an **image** labeled with **object bounding boxes**, **object class**[10], and **grasp bounding boxes**. Also, stacking relationships between objects in images are provided in a **relationship tree**. Based on this, we further label various **language descriptions** and a **scene graph** for each sample in VMRD[11] based on the spatial information from object bounding box and relationship tree. An auxiliary grasp-wise spatial attribute **surface** is also introduced for better cluttered grasping performance.

A.1.2 Dataset Usage

For the image-to-text model, (image and language descriptions) are used. For the language grounding model, (image, language descriptions, and object bounding box) are used. For scene graph generation and graph-based object selection model, (image, scene graph, object bounding box, grasping bounding box) are used. For grasping model (vanilla), (image, object bounding box, grasping bounding box, surface) are used. All of model are setup in Sec. 4.2.

A.1.3 Scene Graph Example

For example, in Figure.2(b), the relationship tree only shows relationships as *"mobile phone-on-box"* and *"box-on-notebook"*, but cannot encode the relationship between *"mobile phone"* and *"notebook"* (e.g., *"mobile phone-on-notebook"*).

A.1.4 Surface Example

In Figure. 2(a), the *"notebook"* is stacked by an *"apple"*, thus the **surface** corresponding to *"apple"* grasping groundtruth is *"False"*. As for the toothpaste, it is not under any other objects, thereby labeled *"True"* for **surface**. In our task, this attribute can improve grasping performance.

[10]The object class is used to generate language description of the scene and construct the scene graph in our proposed dataset.

[11]Note that we filter seven samples with incorrect labeling in the original VMRD.

A.2 Details of Language-based Grasping Model

A.2.1 Backbone Model

Our backbone model is developed on top of the two-stage grasp detection pipeline (Chu et al., 2018) (a grasp-version Faster RCNN (Ren et al., 2016)), fusing the language knowledge as guidance. In doing so, we propose a Knowledge-guided Grasp Proposal Network (K-GPN) to replace with Region Proposal Network (RPN) in Figure. 7, for fusing the grounded object feature with the visual feature. In our framework, we formulate the grasp detection as three parts: (1) Grasp Proposals, (2) Grasp Orientation Classification and Multi-grasp Detection, and (3) Grasp Stacking Classification, described below.

Finally, the highest-confidence angles are selected for each grasp bounding box, and the grasp bounding box (predicted from proposals) corresponding to the highest confidence (mean of the bounding box confidence and surface confidence) is selected as g_i in Eq. (1) with the selected angle.

A.2.2 Grasp Proposals

The module aims to fuse grounded object feature and visual feature to the grasped object. The visual feature used here is a feature map ($z \in R^{50\times50\times1024}$) of the intermediate layers of ResNet-101, and the grounded object feature ($k \in R^{1\times4}$, also named K_g) is obtained from language grounding model (in Sec.3.3) based on **Self-explanation** or **Human-intervention**. The proposed K-GPN is employed to fuse z and k and output a new feature vector $1 \times 1 \times 512$, which is further fed into a two-layer Multi-Layer Perceptron (MLP) to predict the probability of grasp proposal and region of interest (ROI). Different from RPN used by (Chu et al., 2018), which takes positive and negative proposals with groundtruth over the whole image, the proposed K-GPN samples proposals based on language knowledge and produces ROI related to the **selected object** in Algorithm 1. The ROI features from **K-GPN** is fed into the following module to re-optimize and predict the final grasp, shown in Figure. 7. The details are described in Appendix A.2.3 and A.2.4.

As shown in Algorithm 1, our method takes in visual feature z and grounded object feature k. And then, the grasp proposals G generated from RPN is selected by satisfying iou constraints and $tiou$ constraints (language knowledge). g_{gt} is the

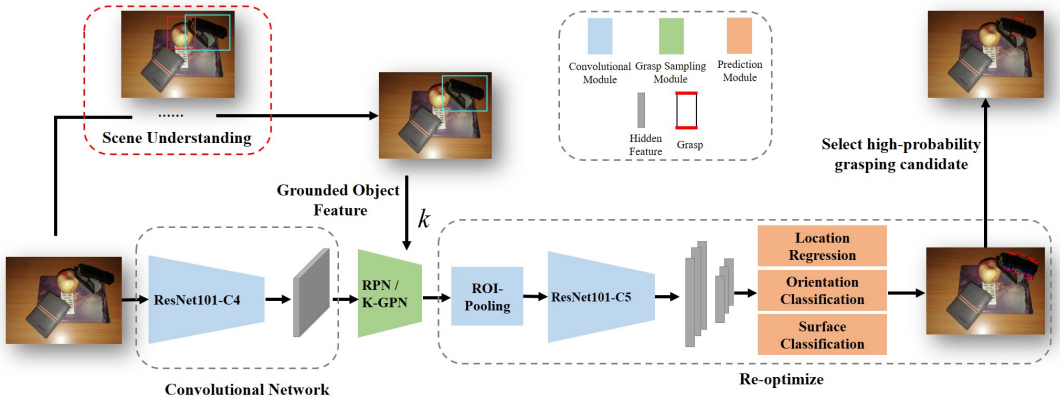

Figure 7: The architecture of Language-based Grasping Model. The input of the scene image is fed into **ResNet101-C4** to extract visual features, which is one of the inputs of our proposed **K-GPN**. The grounded object feature k is another input of **K-GPN**. **K-GPN** predicts grasp proposals and output proposal grasping region (also named ROI) features, which forwards into **ROI-Pooling** and **ResNet101-C5** to obtain 2048 dimensions feature vectors. These feature vectors are used to predict the final grasping location, orientation, and surface. The highest confidence grasping candidate is selected to execute by a real robot.

Algorithm 1 Knowledge-guided Grasp Proposal Network (**K-GPN**)

Input: visual feature z, grounded object feature k
Model: K-GPN
Output:Grasp Proposals G

Global proposal set G_g = RPN(z)
Positive proposal set and Negative proposal set:S_p, S_n
while $size(S_p)$ and $size(S_n)$ are less than the sampling count **do**
  Sampling a grasp proposal g from G_g
  if $iou(g, g_{gt}) > 0.5$ and $tiou(g, k) > 0.5$ **then**
    Put g into Positive proposal set S_p
  else
    Put g into Negative proposal set S_n
  end if
end while
return Grasp Proposals $G = \{S_p, S_n\}$

groundtruth grasp corresponding to the proposed g. $size(\cdot)$ is the number of the set. $iou(\cdot)$ is the conventional Intersection-over-Union (IoU) function (Ren et al., 2016). $tiou(\cdot)$ is a function defined in Eq. (8), used to select knowledge-guided ROI:

$$tiou\,(g,k) = \frac{|g \bigcap k|}{|g|}, \tag{8}$$

where g is the grasp proposal, and k is the grounded object feature (visual-language grounded object bounding box).

The proposal loss is defined in Eq. (9)

$$L_p\left(\left\{(p_c,t_c)_{c=1}^{C}\right\}\right) = \sum_c L_{cls}\,(p_c,p^*) + \lambda_1 \sum_c p^* L_{loc}\,(t_c,t^*), \tag{9}$$

where C is the set of all proposals, L_{cls} is the cross entropy loss of grasp proposal classification (binary classification). L_{loc} is the smooth L1 regression loss of the proposal locations. (p_c, t_c) is the binary class and proposal location to the i-th proposal. p_c is True if a grasp is specified, and False if not. (p^*, t^*) is the groundtruth. λ_1 is the weight coefficient.

A.2.3 Grasp Orientation Classification and Multi-grasp Detection

Our model quantizes the orientation θ into $R+1$ classification problem by discretizing the continuous orientation angles into R values. Another non-grasp case is also considered in the classification problem and in that case, the grasp proposal is considered incorrect. We select the highest score class as the orientation angle value. In practice, we use equal intervals of 10° to discretize the angles and $R + 1 = 19$. The loss function is as follows:

$$L_g\left(\left\{(\rho_l,\beta_l)_{c=0}^{C}\right\}\right) = \sum_c L_{cls}\,(\rho_l,\rho^*) + \lambda_2 \sum_c 1_{c\neq 0}(c) L_{loc}\,(\beta_c,\beta^*), \tag{10}$$

where ρ_l denotes the probability of class l and β_l corresponds to the grasp bounding box. L_{cls} is the cross entropy loss of the orientation angle classification (19-class classification). L_{loc} is the smooth L1 loss of bounding boxes with the weight coefficient λ_2 when angle class $c \neq 0$, where $c = 0$ is short for non-grasp case. (ρ^*, β^*) is the groundtruth.

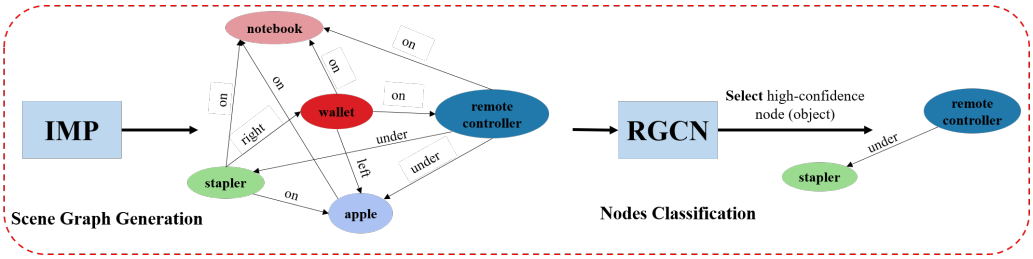

Figure 8: The architecture of scene-graph based scene understanding module.

A.2.4 Surface Classification

We propose a binary classification task to predict whether the grasped object is on the top of a stack of objects. The loss function is as follows:

$$L_s\left(\left\{(p_c)_{c=1}^{C}\right\}\right) = \sum_c L_{cls}\left(s_c, s^*\right), \qquad (11)$$

where the same as Eq. (9), L_{cls} is the cross entropy loss of grasp proposal **surface** classification (binary classification). s_c is False if the grasped object is stacked by others, and True if not. s_* is the groundtruth. Detail is available in Appendix A.1.4.

The total training loss for language-based grasping detection is:

$$L_{total} = L_p + L_g + L_s. \qquad (12)$$

A.3 Image-to-text Model

MMT is shown in Figure. 9. **Input** is the region features and bounding box detected from the robot observed image. **Output** is a description of the spatial relationships of objects in the scene. We hope the subject object can be grasped without collision based on the described spatial relationship.

A.4 Training Details of Language-based Grasping Model

For the baseline model, during ROI sampling in **K-GPN**, the positive and negative sampling counts for loss calculations are both 128. The optimizer is Adam and the learning rate is $1e-4$ for 100 epochs with batch size 8. λ_1 and λ_2 are both 1.0.

It is noted that for grounded object feature k, we first use the groundtruth of the language grounding model to train from scratch and fine-tune the model using the outputs of the language grounding model.

A.5 Hardware and Software Implementation

The grasping execution is taken place on a single-arm Kinova Jaco 7DOF robot under the framework of Robot Operating System (ROS) Kinetic, shown in Figure.3. We use an Intel RealSense SR300 RGB-D camera to obtain RGB-D images mounted on the wrist of the robot. All the computation is completed on a PC running Ubuntu16.04 and Pytorch 1.7 with one Intel Core i7-8700K CPU and one NVIDIA Geforce GTX 1080ti GPU.

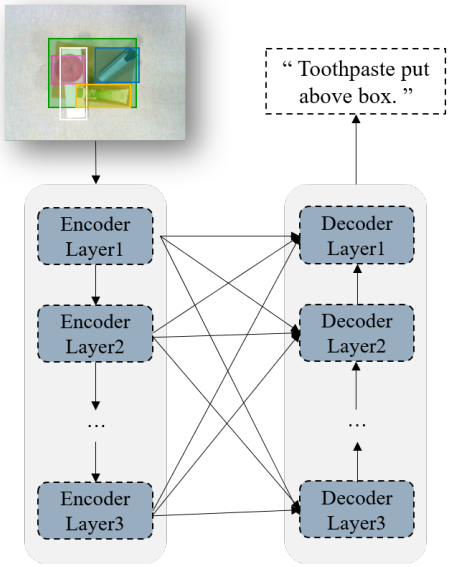

Figure 9: The overview of the image-to-text model applied in our self-explanation pipeline.

A.6 Metric Details

R@k is the percentage of cases where at least one of the top-k detection is correct. P@k is the correct rate for all top-k predictions.

In the simulation setting, a correctly detected grasp has a Jaccard Index greater than 0.25 and the absolute orientation error less than 30° relative to at least one of the groundtruth grasps of the collision-free object (Kumra et al., 2020).

In the physical setting, for the grounding model, the correct label is Intersection-over-Union (IoU) over 0.5, and for the image-to-text model, the correct label is human-annotated.

A.7 Scene Graph based method

A.7.1 Model Structure

The model **SceneGraph-Rep** uses a scene-graph based scene understanding module (shown in Figure. 8) to realize the function of red frame in Figure. 7.

The module is hierarchical including two submodules: **(i)** Iterative Message Passing (IMP)(Xu et al., 2017) to generate a scene graph in our work. **(ii)** Relational Graph Convolution Network (RGCN)(Schlichtkrull et al., 2018) to realize binary classification (can or cannot be grasped for each node). The IMP and RGCN are trained on our proposed L-VMRD same as our proposed method.

The **input** is region features from common object detection model (Faster RCNN (Ren et al., 2016)) using scene image I. IMP outputs the scene graph S_g in the form of triple (i.e., <subject, predicate, object>), which is fed into RGCN to predict the graspability of each node (object). The high-confidence object is selected corresponding with the bounding box. The whole process can be formulated as follows:

$$\begin{aligned} Z, B &= Detecton\,(I)\,, \\ S_g &= IMP\,(Z)\,, \\ (x_t, y_t, w_t, h_t) &= RGCN\,(S_g, B)\,, \end{aligned} \tag{13}$$

where $Detecton$ is a common object detection model. Z and B are the set of region features and the set of bounding boxes for each detected object, respectively. $\{x_t, y_t, w_t, h_t\}$ is the bounding box of selected object, same as definition in Sec.3.3.

A.7.2 Case Study

In Figure. 10, we give two failure cases caused by low-quality scene graph generation, indicating that **SceneGraph-Rep** highly depends on the output quality of the scene graph generation model. In Figure. 10(a), failure is caused by the incorrect scene graph generation. In Figure. 10(b), failure is mainly caused by the error classification from RGCN.

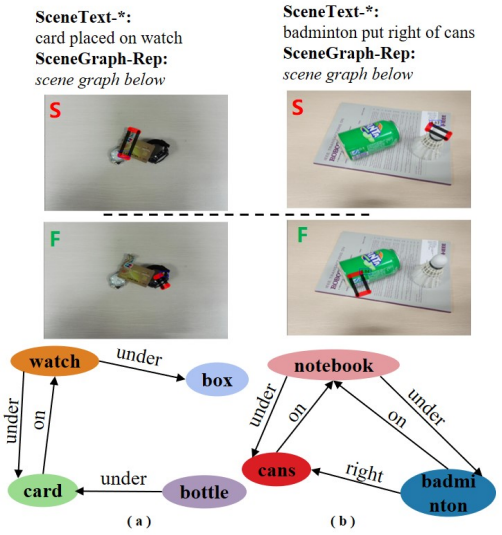

Figure 10: Visualization of MMT (image-to-text) and IMP (scene graph) based self-explanation models. **S** means successful case, while **F** means failure case.

Automated Chinese Essay Scoring from Multiple Traits

Yaqiong He, Feng Jiang, Xiaomin Chu and **Peifeng Li***
School of Computer Science and Technology, Soochow University, Suzhou, China
{20204227070,fjiang}@stu.suda.edu.cn
{xmchu,pfli}@suda.edu.cn

Abstract

Automatic Essay Scoring (AES) is the task of using the computer to evaluate the quality of essays automatically. Current research on AES focuses on scoring the overall quality or single trait of prompt-specific essays. However, users expect to obtain not only the overall score but also the instant feedback from different traits to help improve their writing. Therefore, we first annotate a multi-trait dataset ACEA including 1220 argumentative essays from four traits, i.e., the essay organization, topic, logic, and language. And then we design a Hierarchical Multi-task Trait Scorer (HMTS) to evaluate the quality of writing by modeling these four traits. Moreover, we propose an inter-sequence attention mechanism to enhance information interaction between different tasks and design the trait-specific features for various tasks in AES. The experimental results on ACEA show that our HMTS can effectively score essays from multiple traits, outperforming several strong baselines.

1 Introduction

Automatic Essay Scoring (AES) is a task of using the computer to evaluate the quality of students' essays. An efficient AES system can bring many benefits to the field of education, including avoiding the influence of subjective factors on essay scoring, greatly reducing teachers' workload, and providing instant feedback for students' written essays.

Early work treated AES as a classification (Larkey, 1998; Rudner and Liang, 2002), regression (Attali and Burstein, 2005; Phandi et al., 2015), or ranking problem (Yannakoudakis et al., 2011; Chen and He, 2013), addressing AES by supervised learning. Recently, with the progress of deep learning, the end-to-end models based on neural networks (e.g., LSTM-based models or the combination of LSTMs and CNNs) have achieved impressive results (Alikaniotis et al., 2016; Taghipour and Ng, 2016; Dong and Zhang, 2016; Dong et al., 2017; Tay et al., 2018).

In consideration for more feedback in AES systems, some studies began to focus on scoring specific traits of essays, such as word choice, content (Page, 1966), organization (Persing et al., 2010; Song et al., 2020a), thesis clarity (Persing and Ng, 2013), prompt adherence (Persing and Ng, 2014), argument persistent (Ke et al., 2018), style (Page, 1966), etc. Most of the above studies focused on scoring prompt-specific essays. As previous researchers had pointed out (Jin et al., 2018; Ridley et al., 2020), it was difficult and expensive to access ample target-prompt essays in the real-world AES system. Therefore, some researchers on English AES had began to explore cross-prompt AES (Ridley et al., 2020; Phandi et al., 2015; Jin et al., 2018). Their methods utilized non-target-prompt essays to train models and score target-prompt essays.

In reality, people usually judge the essay from different perspectives to give the final overall holistic score. Apart from (Ridley et al., 2021; Kumar et al., 2021), few studies had combined the overall score with the feature-specific scores. Therefore, to make AES more interpretable and verify whether the information learned in the traits contributes to the learning of the overall score, we use multi-task learning to evaluate the overall holistic score from multiple traits and the score of each trait.

This paper focuses on scoring Chinese essays and their traits. It is worth noting that most of the previous studies were conducted on the English-language dataset ASAP[1], which contains eight different collections of essays with specific prompts. Unlike that, our Chinese dataset is thematically diverse and does not contain prompts, so it is more challenging than the English task.

There are some issues between English and Chinese for the overall score and the traits scores. First, they have different expression problems. The

*Corresponding author

[1]https://www.kaggle.com/c/asap-aes.

Proceedings of the 29th International Conference on Computational Linguistics, pages 3007–3016
October 12–17, 2022.

most significant differences are that Chinese is a parataxis language while English is hypotaxis. In the task of English AES, word usage is the most important factor in English expression (Ridley et al., 2021). English text mainly forms semantic representation by combining the similarity between words. While Chinese essays pay more attention to sentence-level and paragraph-level expression. Second, Chinese and English essays have different evaluation emphases that we should evaluate the essay quality from different perspectives. For example, for text expression, the evaluation criteria for English essays are using language correctly and high-level vocabulary, while the evaluation criteria for Chinese essays are using beautiful sentences and smooth expressions.

Therefore, our task of scoring Chinese essays and their traits exhibits two challenges. First, we need to select appropriate traits to express Chinese essays and design the proper model to combine the different traits to prompt the performance of overall scoring. Second, a good model is needed to fully express the deep semantics of Chinese essays, which should have a good generalization ability to perform well in different trait tasks.

In this paper, we design a Hierarchical Multi-task Trait Scorer (HMTS) for the task of automated Chinese essay scoring. To cope with the first challenge, we created a dataset named "Automatic Chinese Essay Assessment (ACEA)" extended from the dataset used in Song et al. (2020b), which contains 1220 essays, the overall score of each essay, and the scores of the four traits (i.e., topic, organization, logic, and language). And then we utilize the multi-task learning framework to learn the trait levels. Specially, to better combine different trait expressions, we propose an inter-sequence attention mechanism, which can integrate the representations of different trait tasks. We believe that different trait tasks can complement each other. For example, an essay that scores well on logic will also score well on language performance. Besides, since different tasks have different evaluation criteria, we also obtain trait-specific features to enhance the representation of the specific trait tasks.

To cope with the second challenge, we utilize the hierarchical model to obtain the shared sentences and paragraph representation, and essay representation independent for each task. We use the XL-Net as the sentence encoder, which is a pre-trained model trained on large-scale datasets and can get better text semantic representation. In a word, the contributions of this paper are as follows:

- We are the first to introduce the multi-trait scores to Chinese AES, and grade our ACEA dataset with four traits, i.e., organization, topic, logic and language.
- We design a hierarchical multi-task trait scorer HMTS, which can superior express the semantics of Chinese essays and give the overall score as well as the trait scores for essays without prompt.
- We use the inter-sequence attention mechanism to fuse the representation of different traits to share the contribution of the other trait tasks, and use the trait-specific features to enhance the representation of each trait task.

2 Related Work

In this section, we first introduce the prompt-specific and cross-prompt holistic scoring methods and then describe the prompt-specific and multi-task trait scoring methods. Finally, we briefly introduced the task of Chinese AES.

2.1 Holistic Scoring

Prompt-Specific Holistic Scoring For the overall essay score, early studies mainly leveraged regression, classification or ranking algorithms combined with manual features (Page, 1966; Attali and Burstein, 2005; Larkey, 1998; Rudner and Liang, 2002; Yannakoudakis et al., 2011; Chen and He, 2013; Miltsakaki and Kukich, 2004; Burstein et al., 1998). In recent years, many studies (Taghipour and Ng, 2016; Cummins et al., 2016; Alikaniotis et al., 2016; Dong and Zhang, 2016; Dong et al., 2017; Tay et al., 2018) had shown that neural network methods can capture the deep semantics of essays more effectively. For example, Dong et al. (2017) used the attention mechanism to obtain the contribution of words and sentences to the essay. Tay et al. (2018) measured similarity between adjacent sentences to model coherence to evaluate text quality.

Cross-Prompt Holistic Scoring In fact, it's unrealistic to get ample essays with a specific prompt. A good AES should be able to provide feedback to essays on any prompt. Therefore, some studies on English focused on cross-prompt scoring. Song et al. (2020b), Phandi et al. (2015) and Cummins

et al. (2016) trained the model with large quantities of non-target-prompt and a small quantity of target-prompt essays and performed transfer learning to score target-prompt essays. However, these methods still need essays with target prompts. Some recent work explored cross-prompt scoring without target prompts. Jin et al. (2018) applied a two-stage approach, in which the first stage utilized the prompt-independent features to award pseudo labels to target prompt essays. Then in the second stage, the pseudo-labeled essays were used as training data to award the final score. Ridley et al. (2020) applied a single-stage neural network-based method that utilizes a set of general features to award scores to target-prompt essays.

2.2 Trait Scoring

Prompt-Specific Trait Scoring In addition to the overall score, students also expect to get feedback from different aspects through AES, which can assist students' growth.

In the last decade or so, some studies focused on scoring the single essay trait. For example, Song et al. (2020a) proposed a hierarchical multi-task learning model to evaluate the organization quality by learning the sentence and paragraph function. Ke et al. (2018) modeled argument persistence and the attributes of those arguments in student essays. Some more recent studies had adapted a leading prompt-specific holistic scoring method to output the score for different traits. Mathias and Bhattacharyya (2020) adapted some leading approaches for prompt-specific persistent scoring to the task of trait scoring. Hussein et al. (2020) also adapted a leading prompt-specific holistic scoring method, employing a multi-task architecture to output the overall score and scores for various traits simultaneously.

Multi-task Trait Scoring To obtain more feedback, some studies used multi-task learning to score essay traits and the essay itself. Kumar et al. (2021) proposed a prompt-specific method and introduced the multi-task learning (MTL) method to essay scoring, where scoring the overall essay score is the primary task, and scoring the essay traits is the auxiliary task. Ridley et al. (2021) provided a new approach named Automated Cross-prompt Scoring of Essay Traits, which can give the holistic score as well as the scores for different traits in the cross-prompt setting. The drawback of these studies is that their methods are not suitable for Chinese AES and also are prompt-specific. Due to the different evaluation criteria of Chinese and English essays, traits selection should be different.

2.3 Chinese AES

Research on Chinese AES is later than that of English AES and only a few works focused on this task. Among them, Yang and Cao (2012) applied LDA (Latent Semantic Analysis) to score Chinese essays. Fu et al. (2018) improved the accuracy of Chinese AES by identifying beautiful sentences and taking them as literary features. Song et al. (2020a) evaluated the organizational score of high school argumentative essays, and Song et al. (2020b) explored cross-prompt holistic scoring on four different essay sets, with articles in each dataset responding to a different prompt. However, their dataset is not publicly available.

3 ACEA Dataset

Following the college entrance examination essay scoring standard, we selected four traits of topic, organization, logic, and language for our multi-task learning, which can help evaluate the level of essay quality.

3.1 Traits Selection

For the essay topic, whether the essay conforms to the topic's meaning and whether the center is prominent is one of the criteria to evaluate the essay's grade based on the essay evaluation criteria of the college entrance examination.

For the essay organization, Song et al. (2020a) showed that organization is a critical aspect of Chinese writing. A well-organized article should have a clear structure to accurately and logically develop ideas. Chen (2016) had set up a set of solutions to evaluate the quality of the Chinese essays, taking the organization as one of the evaluation criteria.

For the essay logic, Yan (2019) showed that the intention of the essay should be profound and reflect the correct and positive thought and value orientation, and be consistent with the mainstream of social development and the spirit of the times. Therefore, we take logic as one of the traits, which takes whether the logic is clear and the content is profound as the evaluation standard.

For the essay language, Liu (2015) and Liu et al. (2016) used the rhetorical recognition of parallelism metaphor to express the literary grace of the article. Gong (2016) conducted in-depth research

Trait	Evaluation Criteria
Topic Grades	**Bad**: The topic is not clear; the material can't describe the topic; the center is unclear. **Medium**: The essay has a clear topic and center; the material can basically describe the topic. **Great**: The content revolves around the topic; the material is rich enough to express the topic; the center is prominent.
Organization Grades (Song et al., 2020a)	**Bad**: The essay is poorly structured. It is incomplete or misses key discourse elements. **Medium**: The essay is well structured and complete, but could be further improved. **Great**: The essay is fairly well structured and the organization is very clear and logical.
Language Grades	**Bad**: It is plain language or the sentence pattern is single. **Medium**: Some languages are beautiful; the overall language is fluent; there are changes in sentence patterns. **Great**: Beautiful and fluent language or flexible sentence pattern.
Logic Grades	**Bad**: Unclear logic; The discussion is not profound. **Medium**: The logic is clear and the discussion is not profound or logic and discussion are not too bad. **Great**: Clear logic and profound discussion.

Table 1: Trait grades and their evaluation criteria.

on rhetorical devices in Chinese essays and realized an AES system based on the figure of speech recognition, which showed that the introduction of rhetorical devices has produced good results for the system scoring. Therefore, we take the language as one of the traits. Language evaluation criteria include the use of beautiful sentences and the fluency of sentences.

3.2 Traits Grades

We evaluate the quality of topic, organization, language, and logic of the essay with three grades. The evaluation criteria are shown in Table 1.

3.3 Data Annotation

Our dataset contains 1220 argumentative essays published by Song et al. (2020a), which were written by senior high school students and selected from LeLeKetang[2]. These essays contain various topics and do not have specific prompts. The overall scores we used come from the LeLeKetang website and the organization scores have been given by Song et al. (2020a). We asked two annotators who are Chinese teachers in high school to assign other three trait grades for each essay.

For the other three trait scores, we counted the consistency rate of annotation. According to kappa calculation, the internal annotation consistency of topic, logic and language were 0.87, 0.9 and 0.88 respectively. A third annotator was introduced to discuss and analyze essays with disagreed annotations to get the final decision. Table 2 shows the statistics of the dataset.

[2]http://www.leleketang.com/zuowen/.

Trait	Bad	Medium	Great
Organization	245	670	305
Topic	209	549	462
Logic	302	596	322
Language	259	648	313
Overall	250	645	325

Table 2: Essay distribution on trait and grade.

4 HMTS Model

We employ a hierarchical model to learn the semantic representation of texts and reinforce the expression of each trait task through multi-task learning. For our hierarchical neural model, we utilize XLNet (Yang et al., 2019) to learn the sentence expressions and LSTM to learn the paragraph and essay expressions, which can simulate the coherence between the sentence and paragraph sequences. We also use the attention pooling to encode sentences, which aims to capture more relevant sentences that contribute to the paragraph semantic representation, thereby further expressing the quality of an article.

For our multi-task framework, unlike the English multi-task AES designed by Ridley et al. (2021) and Kumar et al. (2021), we choose different traits for Chinese trait scoring, and our test dataset contains essays with multiple prompts rather than a single prompt. To share information between different traits more clearly, we propose the inter-sequence attention mechanism to help each trait pay attention to the information contained in the other traits. We also introduce the trait-specific features to represent the trait quality.

Our HMTS model includes two stages and the

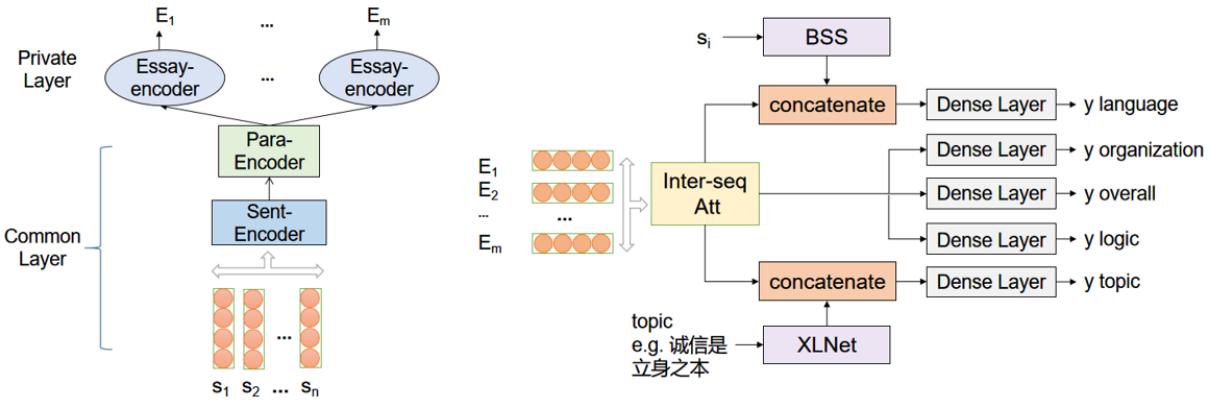

Figure 1: Architecture of our HMTS. The left part shows the sentence encoder and paragraph encoder in the common layer, and the essay encoder in the private layer. The right part contains the inter-sequence attention and trait-specific features in the private layer, in which BSS (Beautiful Sentence Scorer) is used to extract language-specific feature.

details are shown in Figure 1. HMTS includes two layers: 1) Common Layer: this layer aims to learn the useful low-level representations for all tasks, including the sentence and paragraph representation; 2) Privates Layer: this high-level layer is employed to learn more task-specific representations, including essay representation, the inter-sequence mechanism, and trait-specific features. High levels in multi-task architecture can represent more complex information (Sanh et al., 2019).

4.1 Common Layer

The parameters in the lower layer of our model HMTS are shared, so as to enable the sharing of information relevant to all trait tasks. Specifically, the lower layer includes the sentence encoder and the paragraph encoder.

Sentence Encoder A sequence of words $\boldsymbol{s}_i$= $\{w_1,w_2,...,w_m\}$ is modeled with a XLNet encoder, and the vector representation of the last word is used as the final sentence representation $\boldsymbol{s}_{e_i}$.

Paragraph Encoder Firstly, BiLSTM is used to encode the paragraph with the sentence sequence $\{\boldsymbol{s}_{e_i}, \boldsymbol{s}_{e_{i+1}}, ..., \boldsymbol{s}_{e_{i+n-1}}\}$ as follows, where $\boldsymbol{h}_i \in \mathbb{R}^{512}$ is the hidden representation at the time-step i.

$$\boldsymbol{h}_i = \text{BiLSTM}(\boldsymbol{s}_{e_i}, \boldsymbol{h}_{i-1}) \tag{1}$$

A common attention pooling layer is then applied to the hidden representations to learn the paragraph representation as follows.

$$\boldsymbol{e}_i = \tanh(\boldsymbol{W}_e \cdot \boldsymbol{h}_i + \boldsymbol{b}_e) \tag{2}$$

$$\boldsymbol{a}_i = \frac{exp(\boldsymbol{w}_a \cdot \boldsymbol{e}_i)}{\sum_{j=1}^{n} exp(\boldsymbol{w}_a \cdot \boldsymbol{e}_j)} \tag{3}$$

$$\boldsymbol{P} = \sum_{i=1}^{n} \boldsymbol{a}_i \cdot \boldsymbol{h}_i \tag{4}$$

where $\boldsymbol{W}_e$ and $\boldsymbol{W}_a$ are weight matrix and vector respectively, $\boldsymbol{b}_e$ is the bias vector, $\boldsymbol{e}_i$ and $\boldsymbol{a}_i$ are the attention vector and attention weight of the i-th sentence, n denotes the number of sentences in a paragraph. $\boldsymbol{P} \in \mathbb{R}^{512}$ is the final paragraph representation.

4.2 Private Layer

There are M (M=5) tasks in total (four trait tasks and one overall task), and each component in the private layer has M separate copies.

Essay Encoder We encode the paragraph sequence with BiLSTM as follows and the mean of hidden layers is used as essay representation.

$$\boldsymbol{H}_i^j = \text{BiLSTM}(\boldsymbol{P}_i^j, \boldsymbol{H}_{i-1}^j) \tag{5}$$

$$\boldsymbol{E}_j = \text{mean}(\boldsymbol{H}_1^j, \boldsymbol{H}_2^j, ..., \boldsymbol{H}_m^j) \tag{6}$$

where for the j-th task, $\boldsymbol{P}_i^j$ is the paragraph representation of the i-th paragraph, $\boldsymbol{H}_i^j$ is the hidden representation at the time-step i, and $\boldsymbol{E}_j$ is the essay representation.

Inter-sequence Attention Mechanism To obtain the contribution of the other tasks to the current trait task, we introduce the inter-sequence attention mechanism as follows.

$$\boldsymbol{ET} = \{\boldsymbol{E}_1, \boldsymbol{E}_2, \boldsymbol{E}_3, \boldsymbol{E}_4, \boldsymbol{E}_5\} \tag{7}$$

$$\boldsymbol{E} = \tanh(\boldsymbol{ET}) \tag{8}$$

$$\boldsymbol{a} = \text{softmax}(\boldsymbol{v}^T \cdot \boldsymbol{E}) \tag{9}$$

$$\boldsymbol{R}_t = \tanh(\boldsymbol{ET} \cdot \boldsymbol{a}^T) \tag{10}$$

where $\boldsymbol{T} = \{\boldsymbol{E}_1, \boldsymbol{E}_2, \boldsymbol{E}_3, \boldsymbol{E}_4, \boldsymbol{E}_5\}$ is the concatenation of the essay representation of each task. We initialize a training parameter $\boldsymbol{v} \in \mathbb{R}^{512}$ to obtain the final trait fusion $\boldsymbol{R}_t \in \mathbb{R}^{512}$ representation.

Task-specific Features To obtain a final representation for each task, we introduce the task-specific features to enhance the task representation.

Topic-task feature Since our dataset lacks prompts, we use the essay title as the topic task feature. Specifically, we encode the essay title with XLNet and concatenate the last word representation with the topic representation as the final topic-task essay representation as follows.

$$\boldsymbol{f}_{top} = \text{XLNet}(\boldsymbol{s}_t) \tag{11}$$

$$\boldsymbol{E}_{top} = \text{Dense}(\text{concatenate}(\boldsymbol{f}_{top}, \boldsymbol{R}_t)) \tag{12}$$

where $\boldsymbol{s}_t$ is the word sequence of the essay topic, $\boldsymbol{f}_{top}$ is the topic feature representation, and $\boldsymbol{E}_{top}$ is the final topic-task representation.

Language-task feature According to the language level evaluation rules in Subsection 3.2, beautiful and fluent language is an important influence for the language level. Therefore, we trained a model to predict whether sentences are beautiful or not. We selected 8840 underlined sentences from LeLeKetang as beautiful sentences, of which underlined sentences are beautiful sentences annotated on the website and 6714 bad sentences from the low-level essays. We split the dataset into the training dataset (80%) and testing dataset (20%) and used the XLNet classification model to encode the sentence and fed the last word representation into a dense layer for prediction, and the results are Yes and No.

We used the pre-trained beautiful sentence scorer (BSS) to predict sentences in each essay. All prediction results were concatenated with the language representation of the essay as the final language-task essay representation as follows.

$$\boldsymbol{s}'_{ij} = \text{BSS}(\boldsymbol{s}_{ij}) \tag{13}$$

$$\boldsymbol{f}_{lang} = \text{concatenate}(\boldsymbol{s}'_{i1}, \boldsymbol{s}'_{i2}, ..., \boldsymbol{s}'_{in}) \tag{14}$$

$$\boldsymbol{E}_{lang} = \text{concatenate}(\boldsymbol{f}_{lang}, \boldsymbol{R}_t) \tag{15}$$

where $\boldsymbol{s}_{ij}$ is the j-th sentence of the i-th essay, $\boldsymbol{s}'_{ij}$ is prediction result, $\boldsymbol{f}_{lang}$ is the language-task feature, and $\boldsymbol{E}_{lang}$ is the final language-task representation.

4.3 Prediction

Finally, a dense layer with a tanh activation function was applied to the representation of each task as follows.

$$\boldsymbol{y}_j = \tanh(\boldsymbol{W}_y^j \cdot \boldsymbol{E}_j + \boldsymbol{b}_y) \tag{16}$$

where $\boldsymbol{E}_j$ is the representation of the j-th trait task. For the organization-task, the logic task and the overall score, $\boldsymbol{E}_j = \boldsymbol{R}_t$; for the language task, $\boldsymbol{E}_j = \boldsymbol{E}_{lang}$; for the topic task, $\boldsymbol{E}_j = \boldsymbol{E}_{top}$. y_j is the predicted score for the j-th task, $tanh$ is the activation function, $\boldsymbol{W}_y^j$ is a weight vector, and $\boldsymbol{b}_y$ is a bias.

4.4 Loss

To train the parameters, we minimized the binary cross-entropy loss function. Given N essays and M tasks, the loss was calculated as follows.

$$L = -\frac{1}{N \cdot M} \sum_{i=1}^{N} \sum_{j=1}^{M} \sum_{c=1}^{T} (y_{ijc} \cdot \log(P_{ijc})) \tag{17}$$

where T denotes the number of grades. If the golden grade of the sample i equals to c, y_{ijc} = 1; otherwise y_{ijc} = 0. P_{ijc} denotes the prediction probability that the sample i belongs to the grade c. where T denotes the number of grades. If the golden grade of the sample i in the task j equals to c, y_{ijc} = 1; otherwise y_{ijc} = 0. P_{ijc} denotes the prediction probability that the sample i in the task j belongs to the grade c.

5 Experimentation

In this section, we first introduce the experimental settings and then report the experimental results. Finally, we analyze the results on different aspects.

5.1 Experimental Settings

We conduct the experiments on our dataset ACEA described in Section 3. The average number of paragraphs and sentences per essay is 8 and 28, and the figure of words per sentence is 21. The maximum number of sentences and paragraphs in an essay (n and m) is set to 50 and 20, and the maximum number of sentences in a paragraph (n_p) is set to 20. The sentences and paragraphs shorter or longer than the limitations are padded or truncated.

We split our dataset into five folds as Song et al. (2020a). Cross-validation is conducted and the average performance is reported. Each fold has a similar distribution over essay grades. During

Parameter	Value
Embedding size	768
Dimension of BiLSTM hidden state	256
Batch size	32
Learning rate	0.0001
Optimizer	Adam

Table 3: Hyper-parameters and their values.

training, 10% of the training data was selected randomly as the validation set to find the optimal hyper-parameters. The detailed settings of hyper-parameters are listed in Table 3.

To evaluate the performance of our model, we use the Quadratic Weighted Kappa (QWK) metric, which has been widely adopted in both holistic essay scoring and essay trait scoring research (Ridley et al., 2020; Mathias and Bhattacharyya, 2020; Hussein et al., 2020) and is designed to measure the level of agreement between two raters.

5.2 Baselines

To verify the effectiveness of our HMTS, we conduct six strong baselines for comparison as follows.

CNN_LSTM_att (Dong et al., 2017): This model is a leading overall scoring model for specific prompt, which treated input essays as sentence-document hierarchies.

MTL (Kumar et al., 2021): This model uses CNN_LSTM_att to obtain the essay representation and concatenated it with the predicted trait scores to score the essay holistically. We use the multi-task architecture to train the model on each trait individually.

Song2020 (Song et al., 2020a): This model is proposed for organization evaluation of Chinese argumentative student essays, utilizing a hierarchical multi-task approach for joint discourse element identification and organization evaluation. We use a single-task architecture to train the model on each trait individually.

XLNet (Yang et al., 2019): This model uses the pre-trained model XLNet to obtain the paragraph representation, and then concatenate them and feed them into the dense layer to predict the essay grade.

5.3 Experimental Results

We reported the results of our experiments using single-task and multi-task models. Table 4 shows the performance comparison of the baselines and HMTS on the dataset ACEA.

We can find out that CNN_LSTM_att and MTL both perform poorly. There are two reasons for this result. First, CNN_LSTM_att and MTL are models designed for essays with specific prompts, while our dataset does not have specific prompts. Second, these models are designed for English AES and pay more attention to the contribution of words to sentences and essays, while Chinese AES pays more attention to sentence expression. In addition, we can find out that XLNet alone has achieved good performance, which shows the effectiveness of using the pre-trained model to obtain semantic information.

In comparison with the above single-task models, Song2020 achieves the best overall QWK. However, its QWK is still lower than our HMTS, since it only considers the organization element. Compared with the best baselines Song2020 on sing-task and XLNet on multi-task, our HMTS significantly improves overall QWK by 3.07 and 9.28, respectively. This result shows that our HMTS on four traits is effective for the overall essay scoring.

Table 4 also shows the performance comparison of the baselines and HMTS on each trait task on ACEA and we also can find that our HMTS achieves the best QWK on all four traits, i.e., organization, topic, logic and language. These results verify the fact that our HMTS also is effective on scoring traits. Besides, our model was significantly superior to all baselines with a p-value<0.05.

5.4 Analysis on Multi-task Learning

In Table 4, compared with the single-task HMTS, our multi-task HMTS significantly improves overall QWK by 5.15, and this result indicates the effectiveness of the multi-task framework on essay scoring. This is due to the fact that the single-task HMTS can not fully utilize the information contained in the article, while our multi-task HMTS can share the information between different traits, which contributes to the final performance of each trait task. Comparing the performance gap (1.59) between single-task and multi-task XLNet, this gap (5.15) between our HMTS is much larger. This also verifies the effectiveness of the multi-task framework of our HMTS.

In comparison with the single-task HMTS, our multi-task HMTS also achieves better QWK on three traits, i.e., organization (+2.03), topic (+6.52) and logic (+2.17), while it achieves comparable performance on the language trait (-1.16).

Task Type	Model	Overall	Organization	Topic	Logic	Language
Single-task	CNN_LSTM_att	40.13	39.0	37.50	40.23	52.58
	XLNet	52.13	52.47	48.84	52.55	56.68
	Song2020	54.78	55.76	46.23	52.84	50.28
	HMTS	**57.85**	**59.11**	**49.75**	**58.44**	**60.71**
Multi-task	CNN_LSTM_att	41.58	39.25	39.26	41.64	50.85
	MTL	43.59	41.50	38.82	42.70	52.51
	XLNet	53.72	52.76	49.52	53.30	56.75
	HMTS	**63.00**	**61.14**	**56.27**	**60.61**	**59.55**

Table 4: Performance comparison (QWK) of the baselines and HMTS on ACEA.

Model	QWK
HMTS	63.00
w/o Organization	-0.98
w/o Topic	-1.91
w/o Logic	-2.39
w/o Language	-0.91

Table 5: Ablation Experiments.

To gain some insight into the effectiveness of each trait, we conducted the ablation experiments as shown in Table 5, which remove one trait from HMTS one by one. We can find out the fact that the performance of the model HMTS will decline if we remove one trait task from the multi-task framework, which can reflect the contribution of each task to the overall score. We find out that the result decreases the most after removing the logic evaluation, which shows that logic is the most important trait that contributes to the overall score, compared with the organization, topic and language. The reason is that the essay scoring standard of college entrance examination is divided into the basic level and development level, and whether the logic of the essay is clear or whether the discussion is profound belongs to the development level, is the key factor to distinguish the excellent essay from other levels.

5.5 Analysis on Inter-sequence Attention

Table 6 shows the effectiveness of the trait-specific feature and the inter-sequence attention. In Table 6, HMTS (w/o inter-att) is the simplified version of HMTS without the inter-sequence attention mechanism. We can find out that HMTS outperforms HMTS (w/o inter-att) on QWK on all trait tasks. These results indicate that the inter-sequence attention can improve the overall score and traits score effectively, especially the topic trait with the highest improvement (+3.08).

5.6 Analysis on Task-specific Features

The use of well-designed trait-specific features is important for each trait task, because different traits have different evaluation criteria. In Table 6, HMTS (w/o trait-feat) is the simplified version of our HMTS without the task-specific features. Compared HMTS (w/o trait-feat) with HMTS, we can find out that the additional topic-task and language-task features improve the results of all trait tasks, especially the topic trait. This proved the effectiveness of the language-task and topic-task features.

We also evaluated the effectiveness of the beautiful sentence scorer (BSS), The F1 score is 0.91 and this indicates that our BSS can accurately identify beautiful sentences. These sentences are usually fluent in the language, flexible in sentence patterns, and good at using rhetoric and famous aphorisms.

5.7 Error Analysis

In analyzing the experimental results, we find that most essays with recognition errors were recognized as their adjacent grades. That is, the essays with the bad and great grades tend to be misclassified as the grade medium, and those with the medium grade tend to be misclassified as the grade great or bad. Especially, our model HMTS tends to wrongly assign a lower grade to an essay, which accounts for 62.9%, 71.0%, 65.8%, 55.5% and 54.1% on the overall score and the scores on the traits organization, logic, topic and language.

The accuracy of our HMTS identifying the essays with the great, medium, and bad grades are 58.0, 72.9 and 64.0, respectively. As we mentioned above, our HMTS has the underestimation trend and the accuracy on the bad essays is better than those on great essays. In addition, since the medium grade is the majority class, it achieves the best accuracy.

Model	Overall	Organization	Topic	Logic	Language
HMTS	63.00	61.14	56.27	60.61	59.55
w/o trait-fea	-0.56	-0.34	-1.41	-0.59	-0.29
w/o inter-att	-1.78	-1.06	-3.08	-0.59	-1.84
w/o inter-att&trait-fea	-2.07	-1.22	-3.44	-0.81	-1.96

Table 6: The effectiveness of the trait-specific feature and the inter-sequence attention.

6 Conclusion

We first annotate a multi-trait essay scoring dataset ACEA from the topic, organization, logic, and language aspects. And then we propose a multi-task learning framework HMTS for the Chinese AES task. Moreover, we propose an inter-sequence attention mechanism to enhance information interaction between the different trait tasks and design the trait-specific features for various trait tasks. The experimental results show that our HMTS can effectively score essays from multiple traits, outperforming several strong models. In the future, we will focus on incorporating the traits task into the overall task in a more effective framework.

Acknowledgements

The authors would like to thank the three anonymous reviewers for their comments on this paper. This research was supported by the National Natural Science Foundation of China (Nos. 61836007, 62276177 and 62006167), and Project Funded by the Priority Academic Program Development of Jiangsu Higher Education Institutions (PAPD).

References

Dimitrios Alikaniotis, Helen Yannakoudakis, and Marek Rei. 2016. Automatic text scoring using neural networks. *arXiv preprint arXiv:1606.04289*.

Yigal Attali and Jill Burstein. 2005. Automated essay scoring with e-rater v. 2.0 (ets rr-04-45). *Educational Testing Service, Princeton, NJ*, pages 2–20.

Jill Burstein, Karen Kukich, Susanne Wolff, Chi Lu, Martin Chodorow, Lisa Braden-Harder, and Mary Dee Harris. 1998. Automated scoring using a hybrid feature identification technique. In *Proceedings of the 36th Annual Meeting of the Association for Computational Linguistics and 17th International Conference on Computational Linguistics, Volume 1*, pages 206–210.

Hongbo Chen and Ben He. 2013. Automated essay scoring by maximizing human-machine agreement. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*, pages 1741–1752.

Yile Chen. 2016. Research on key technology of chinese automated essay scoring based on regression analysis. Master's thesis, Harbin Institute of Technology.

Ronan Cummins, Meng Zhang, and Ted Briscoe. 2016. Constrained multi-task learning for automated essay scoring. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics*, pages 789–799, Berlin, Germany. Association for Computational Linguistics.

Fei Dong and Yue Zhang. 2016. Automatic features for essay scoring – an empirical study. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 1072–1077, Austin, Texas. Association for Computational Linguistics.

Fei Dong, Yue Zhang, and Jie Yang. 2017. Attention-based recurrent convolutional neural network for automatic essay scoring. In *Proceedings of the 21st Conference on Computational Natural Language Learning (CoNLL 2017)*, pages 153–162.

Ruiji Fu, Dong Wang, Shijin Wang, Guoping Hu, and Ting Liu. 2018. Elegart sentence recognition for automated essay scoring. *Journal of Chinese Information Processing, Volume 32*, pages 88–97.

Jiefu Gong. 2016. The design and implementation of the rhetoric recognition system oriented chinese essay review. Master's thesis, Harbin Institute of Technology.

Mohamed Abdellatif Hussein, Hesham A., and Mohammad Nassef. 2020. A trait-based deep learning automated essay scoring system with adaptive feedback. *International Journal of Advanced Computer Science and Applications, Volume 11*, pages 287–293.

Cancan Jin, Ben He, Kai Hui, and Le Sun. 2018. TDNN: a two-stage deep neural network for prompt-independent automated essay scoring. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics*, pages 1088–1097.

Zixuan Ke, Winston Carlile, Nishant Gurrapadi, and Vincent Ng. 2018. Learning to give feedback: Modeling attributes affecting argument persuasiveness in student essays. In *Proceedings of the 27th International Joint Conference on Artificial Intelligence*, pages 4130–4136.

Rahul Kumar, Sandeep Mathias, Sriparna Saha, and Pushpak Bhattacharyya. 2021. Many hands make

light work: Using essay traits to automatically score essays. *arXiv preprint arXiv:2102.00781*.

Leah S Larkey. 1998. Automatic essay grading using text categorization techniques. In *Proceedings of the 21st annual international ACM SIGIR conference on Research and development in information retrieval*, pages 90–95.

Mingyang Liu. 2015. Research on the key technology of the automatic scoring of the college entrance examination essay. Master's thesis, Harbin Institute of Technology.

Mingyang Liu, Bing Qin, and Ting Liu. 2016. Automated chinese composition scoring based on the literary feature. *Intelligent Computer and Applications*.

Sandeep Mathias and Pushpak Bhattacharyya. 2020. Can neural networks automatically score essay traits? In *Proceedings of the Fifteenth Workshop on Innovative Use of NLP for Building Educational Applications*, pages 85–91.

Eleni Miltsakaki and Karen Kukich. 2004. Evaluation of text coherence for electronic essay scoring systems. *Natural Language Engineering, Volume 10*, pages 25–55.

Ellis B Page. 1966. The imminence of... grading essays by computer. *The Phi Delta Kappan, Volume 47*, pages 238–243.

Isaac Persing, Alan Davis, and Vincent Ng. 2010. Modeling organization in student essays. In *Proceedings of the 2010 conference on empirical methods in natural language processing*, pages 229–239.

Isaac Persing and Vincent Ng. 2013. Modeling thesis clarity in student essays. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 260–269.

Isaac Persing and Vincent Ng. 2014. Modeling prompt adherence in student essays. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1534–1543.

Peter Phandi, Kian Ming A. Chai, and Hwee Tou Ng. 2015. Flexible domain adaptation for automated essay scoring using correlated linear regression. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 431–439.

Robert Ridley, Liang He, Xin-yu Dai, Shujian Huang, and Jiajun Chen. 2021. Automated cross-prompt scoring of essay traits. In *Proceedings of the AAAI conference on artificial intelligence, Volume 35*, pages 13745–13753.

Robert Ridley, Liang He, Xinyu Dai, Shujian Huang, and Jiajun Chen. 2020. Prompt agnostic essay scorer: A domain generalization approach to cross-prompt automated essay scoring. *arXiv preprint arXiv:2008.01441*.

Lawrence M Rudner and Tahung Liang. 2002. Automated essay scoring using bayes' theorem. *The Journal of Technology, Learning and Assessment*.

Victor Sanh, Thomas Wolf, and Sebastian Ruder. 2019. A hierarchical multi-task approach for learning embeddings from semantic tasks. In *Proceedings of the AAAI Conference on Artificial Intelligence, Volume 33*, pages 6949–6956.

Wei Song, Ziyao Song, Lizhen Liu, and Ruiji Fu. 2020a. Hierarchical multi-task learning for organization evaluation of argumentative student essays. In *Proceedings of the Twenty-Ninth International Joint Conference on Artificial Intelligence*, pages 3875–3881.

Wei Song, Kai Zhang, Ruiji Fu, Lizhen Liu, Ting Liu, and Miaomiao Cheng. 2020b. Multi-stage pre-training for automated chinese essay scoring. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 6723–6733.

Kaveh Taghipour and Hwee Tou Ng. 2016. A neural approach to automated essay scoring. In *Proceedings of the 2016 conference on empirical methods in natural language processing*, pages 1882–1891.

Yi Tay, Minh Phan, Luu Anh Tuan, and Siu Cheung Hui. 2018. Skipflow: Incorporating neural coherence features for end-to-end automatic text scoring. In *Proceedings of the AAAI conference on artificial intelligence, Volume 32*.

Dejvn Yan. 2019. Some strategies for writing a good composition for the college entrance examination,. *Middle School Chinese*, page 2.

Chen Yang and Yiwei Cao. 2012. Present situation and prospect of automatic composition scoring. *Language Teaching in Middle School*, pages 78–80.

Zhilin Yang, Zihang Dai, Yiming Yang, Jaime Carbonell, Russ R Salakhutdinov, and Quoc V Le. 2019. Xlnet: Generalized autoregressive pretraining for language understanding. In *Proceedings of the 33rd International Conference on Neural Information Processing Systems*, pages 1–11. Curran Associates, Inc.

Helen Yannakoudakis, Ted Briscoe, and Ben Medlock. 2011. A new dataset and method for automatically grading esol texts. In *Proceedings of the 49th annual meeting of the association for computational linguistics: human language technologies*, pages 180–189.

Semantic-Preserving Adversarial Code Comprehension

Yiyang Li[1,2], **Hongqiu Wu**[1,2] and **Hai Zhao**[1,2,*]
[1] Department of Computer Science and Engineering, Shanghai Jiao Tong University
[2] Key Laboratory of Shanghai Education Commission for Intelligent Interaction
and Cognitive Engineering, Shanghai Jiao Tong University
{eric-lee,wuhongqiu}@sjtu.edu.cn,zhaohai@cs.sjtu.edu.cn

Abstract

Based on the tremendous success of pre-trained language models (PrLMs) for source code comprehension tasks, current literature studies either ways to further improve the performance (generalization) of PrLMs, or their robustness against adversarial attacks. However, they have to compromise on the trade-off between the two aspects and none of them consider improving both sides in an effective and practical way. To fill this gap, we propose Semantic-Preserving Adversarial Code Embeddings (SPACE) to find the worst-case semantic-preserving attacks while forcing the model to predict the correct labels under these worst cases. Experiments and analysis demonstrate that SPACE can stay robust against state-of-the-art attacks while boosting the performance of PrLMs for code[1].

1 Introduction

Inspired by the tremendous success of large pre-trained language models (PrLMs) in natural language understanding (NLU) field (Devlin et al., 2019; Liu et al., 2019; Yang et al., 2019; Clark et al., 2020), many attempts have been made to pre-train a language model for source code comprehension, aiming at learning good code embeddings to help code-related downstream tasks like detecting vulnerable codes, searching codes with natural language queries or answering questions about codes (Zhou et al., 2019; Husain et al., 2019; Liu and Wan, 2021; Kanade et al., 2020a; Buratti et al., 2020; Feng et al., 2020; Guo et al., 2021). Based on this, current literature focuses on either methods to further improve the performance of PrLMs (Jain et al., 2021; Ding et al., 2021; Phan et al., 2021; Wu et al., 2021), or their robustness against adversarial attacks (Zhang et al., 2020; Springer et al., 2020; Yang et al., 2022; Srikant et al., 2021). However, they have to compromise on the trade-off between the two aspects and none of them consider improving both sides in an effective and practical way (Bielik and Vechev, 2020; Yefet et al., 2020). For example, CodeBERT (Feng et al., 2020) and GraphCodeBERT (Guo et al., 2021) are two most widely used PrLMs for code and they have significantly boosted the performance of source code comprehension models. Unfortunately, these powerful models are vulnerable to adversarial attacks that slight perturbations can make them produce wrong labels (Yang et al., 2022). To tackle this problem, previous works attempt to adopt adversarial training framework or data augmentation tricks to improve model robustness, yet they fail to maintain accuracy on the origin clean test set (Bielik and Vechev, 2020; Yefet et al., 2020), or are too complicated to be practical (Ramakrishnan et al., 2020).

A natural question can be asked here: is there a simple yet effective way to subtly avoid the trade-off between performance and robustness and make them both improved? To make the answer yes, we propose Semantic-Preserving Adversarial Code Embeddings (SPACE) to find the worst-case semantic-preserving attacks while forcing the model to predict the correct labels under these worst cases. Different from previous works which also adopt adversarial training on codes (Ramakrishnan et al., 2020; Bielik and Vechev, 2020), we search for the inner worst-case perturbations in the continuous embedding space instead of the discrete token space. Compared with the latter, which is usually a complicated combinational optimization problem, our method is naturally differentiable which makes it easy to be incorporated into the gradient-based training framework and much more efficient. Besides, adding perturbations to the embedding layer is a natural solution to avoid

* Corresponding author. This paper was partially supported by Key Projects of National Natural Science Foundation of China (U1836222 and 61733011).

[1] Our codes and data are available at https://github.com/EricLee8/SPACE

Proceedings of the 29th International Conference on Computational Linguistics, pages 3017–3028
October 12–17, 2022.

the aforementioned trade-off between performance and robustness, which is observed and proved on multiple NLU tasks on plain texts (Miyato et al., 2017; Cheng et al., 2019). Nonetheless, its efficacy on source code comprehension tasks has not been fully investigated. Besides, our experiments show that simply adopting vanilla adversarial training is not enough to gain significant improvements in the context of programming languages, since it can not preserve the semantic meaning and grammatical correctness.

Therefore, in this paper, we combine adversarial training with the data characteristics of programming languages to propose SPACE: we find ways to generate worst-case perturbations that preserve the semantic meaning and grammatical correctness of programming languages. Experiments show that our SPACE is superior to the vanilla adversarial training and significantly outperforms strong baselines of CodeBERT and GraphCodeBERT on several datasets while staying robust against the SOTA attacking methods.

To sum up, the contributions of our SPACE are the following four folds:

- To the best of our knowledge, we are the first to explore the efficacy of adversarial training on the continuous embedding space for source code comprehension tasks.
- We innovatively combine adversarial training with the data characteristics of programming languages to propose SPACE, which is able to boost the performance and robustness of PrLMs for code simultaneously.
- SPACE is practical and effective, which is naturally differentiable and easy to be incorporated into the gradient-based training framework with only a few lines of code.
- Experiments and analyses on several benchmark datasets demonstrate that our SPACE outperforms strong baselines by large margins, and stays robust against state-of-the-art attacks.

2 Related Work

2.1 Pre-trained Language Models for Code

In recent years, the prosperity of PrLMs has been witnessed in natural language understanding field. Inspired by this, researchers start to investigate the power of PrLMs for source code comprehension tasks. To name a few, Kanade et al. (2020b) adopt masked language modeling (MLM) and next sentence prediction (NSP) to pre-train BERT on Python corpus. Buratti et al. (2020) propose C-BERT, which makes use of a carefully designed tokenizer and whole word masking (WWM) objective to pre-train on C language. Feng et al. (2020) present CodeBERT, a bi-modal PrLM trained on massive code-text pairs using masked language modeling (MLM, Devlin et al. 2019) and replaced token detection (RTD, Clark et al. 2020) objectives, in an attempt to handle bi-modal tasks like natural language code search. Guo et al. (2021) design GraphCodeBERT, an even stronger PrLM that leverages the graph-structured code data flow to model the dependency relation between variables. There are many other PrLMs for source code (Jain et al., 2021; Bui et al., 2021; Wang et al., 2021). We do not try our SPACE on all of them since it will cost too much time and computational resources. Without loss of generality, in this paper, we choose CodeBERT and GraphCodeBERT, two most widely-used and easy-to-use models, as our backbone.

2.2 Attacks and defenses on Code

Aiming at testing and improving the robustness of code models, researchers have conducted a series of experiments on attacks and defenses.

For attacks, Srikant et al. (2021) utilize program obfuscations, which are conventionally used to avoid attempts at reverse engineering programs, as adversarial perturbations. They design alternate optimization (AO) and joint optimization (JO) to solve the problem of where and how to inject obfuscations, which yield a satisfying attack success rate. Zhang et al. (2020) propose a Metropolis-Hastings sampling-based identifier renaming technique (MHM) to efficiently generate semantic-preserving adversarial examples. Yang et al. (2022) present ALERT (Naturalness Aware Attack) that considers the natural semantic of generated adversarial examples. They make use of the masked language prediction function of PrLMs to produce a ranked list of potential substitutes for each token, and design genetic algorithms to search for the optimal combination of variables and corresponding substitutes. ALERT is the state-of-the-art attack method and the first to attack PrLMs, yet our experiments demonstrate that our SPACE is able to maintain robustness under ALERT by reducing the attack success rate over 10%.

For defenses, Ramakrishnan et al. (2020) adopt adversarial training framework to improve the ro-

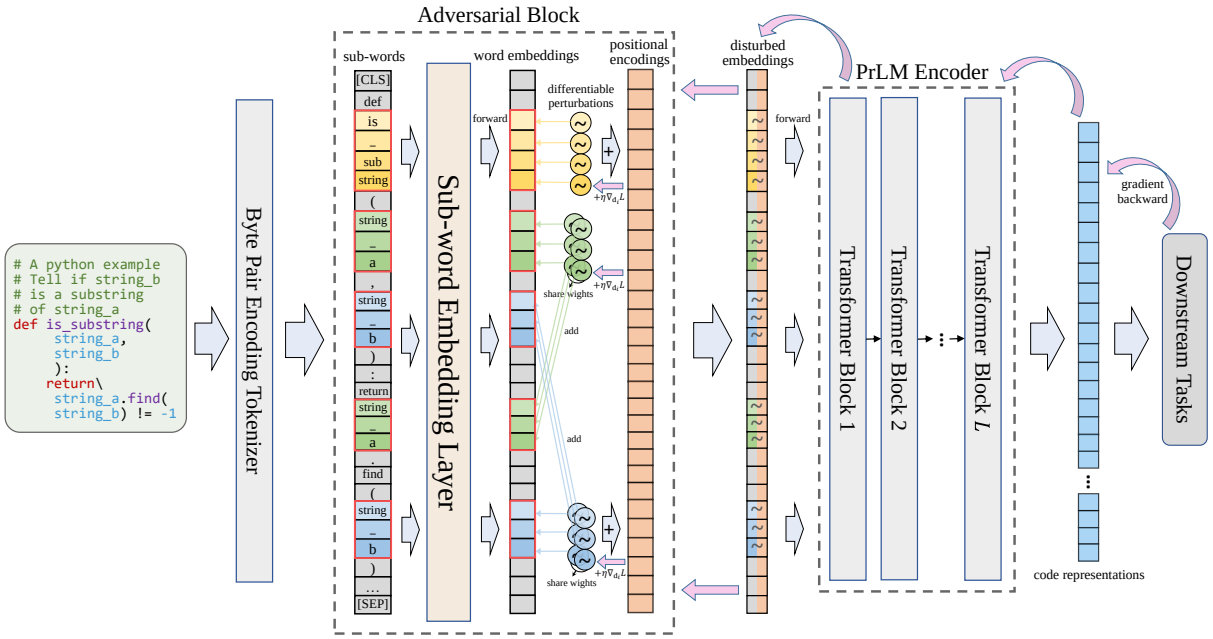

Figure 1: The overview of our model, which consists of three parts: a Byte Pair Encoding (BPE) Tokenizer, an Adversarial Block and a PrLM Encoder. The detailed process is explained in Section 3.3.

bustness of models against various code transformations (e.g. dead code insertion and identifier renaming,). They define the inner maximization problem of adversarial training as finding the worst code transformations, which is a discrete combinational optimization problem. To solve it, they limit the number transformations to 1 and deploy a complicated program sketching trick to approximately solve the inner maximization in a gradient-based way. Their work is too complicated to be practical, on the contrary, our SPACE is performed on the continuous embedding space, which is naturally compatible with the gradient-based training framework and easy to apply with only a few lines of code. Bielik and Vechev (2020) also employ adversarial training framework specifically for the type prediction task. They design uncertainty scores and a program abstractor to solve the inner combinational optimization problem using integer linear programming (ILP). Their work suffers the trade-off between robustness and performance, while our SPACE is capable of improving both aspects simultaneously.

3 Methodology

In this section, we will first briefly introduce the background knowledge: the general adversarial training and it on the continuous embedding space, then present the proposed SPACE in detail.

3.1 Background: Adversarial Training

Adversarial training is defined as a min-max problem, where a model should be learned to minimize the maximum risk posed by the attacking perturbations (Goodfellow et al., 2015). Formally, it can be formulated as follows:

$$\min_{\boldsymbol{\theta}} \mathbb{E}_{(\mathbb{X},y)\sim\mathcal{D}} \left[\max \mathcal{L}(f_{\boldsymbol{\theta}}(\mathcal{A}(\mathbb{X})), y)\right] \quad (1)$$

Here $\mathbb{X}$ and y are the input and its corresponding label sampled from data distribution $\mathcal{D}$, $\mathcal{L}$ is the loss function, $f_{\boldsymbol{\theta}}$ is the target model with trainable parameter $\boldsymbol{\theta}$ and $\mathcal{A}$ is the adversarial transformation applied to $\mathbb{X}$.

Previous works that adopt adversarial training on source code comprehension tasks usually define $\mathcal{A}$ as some semantic-preserving code transformations, which yields *actual* adversarial examples in the discrete token space and is hard to optimize (Bielik and Vechev, 2020; Ramakrishnan et al., 2020). Different from them, we propose to define $\mathcal{A}$ as perturbations on the continuous embedding space:

$$\min_{\boldsymbol{\theta}} \mathbb{E}_{(\mathbb{X},y)\sim\mathcal{D}} \left[\max_{\|\boldsymbol{\delta}\|_F \leq \epsilon} \mathcal{L}(f_{\boldsymbol{\theta}}(\mathcal{A}(\mathbb{X}, \boldsymbol{\delta})), y)\right] \quad (2)$$

where $\boldsymbol{\delta}$ is the perturbation matrix on $\mathbb{X}$, whose Frobenius norm is constrained to be less than ϵ. This definition produces *virtual* adversarial examples in the embedding space, which is easy to optimize via gradient-based methods.

Algorithm 1: Semantic-Preserving Adversarial Training for Programming Languages

Data: Training samples $\mathcal{D} = \{(\mathbb{X}, y)\}$, perturbation bound ϵ, learning rate α, ascent steps K, adversarial learning rate η.
Output: Model parameters $\boldsymbol{\theta}$.

Initialize model parameters $\boldsymbol{\theta}$
for epoch $= 1, 2, \ldots, N_{ep}$ **do**
  for mini-batch $B \subset \mathcal{D}$ **do**
    for $i = 1, 2, \ldots, m$ **do**
      $\boldsymbol{\delta}_0^i \leftarrow \mathbf{0} \in \mathcal{R}^{k_i \times d}$
    end
    $g_0 \leftarrow \mathbf{0}$
    for $t = 1, 2, \ldots, K$ **do**
      $\boldsymbol{\delta}_{t-1} \leftarrow \mathbf{0} \in \mathcal{R}^{n \times d}$
      for $p = 1, 2, \ldots, n$ **do**
        if *position* p *is the* j_{th} *sub-word of identifier* d_i **then**
          $\boldsymbol{\delta}_{t-1}[p] \leftarrow \boldsymbol{\delta}_{t-1}^i[j]$
      end
      $g_t \leftarrow g_{t-1} + \frac{1}{K}\mathbb{E}_{(\mathbb{X},y)\in B}[\nabla_{\boldsymbol{\theta}}\mathcal{L}(f_{\boldsymbol{\theta}}(\mathcal{A}(\mathbb{X}, \boldsymbol{\delta}_{t-1})), y)]$
      for $i = 1, 2, \ldots, m$ **do**
        $g_{adv}^i \leftarrow \nabla_{\boldsymbol{\delta}_{t-1}^i}\mathcal{L}(f_{\boldsymbol{\theta}}(\mathcal{A}(\mathbb{X}, \boldsymbol{\delta}_{t-1})), y)$
        $\boldsymbol{\delta}_t^i \leftarrow \Pi_{\|\boldsymbol{\delta}_{t-1}^i\|_F \le \epsilon}(\boldsymbol{\delta}_{t-1}^i + \eta \cdot g_{adv}^i / \|g_{adv}^i\|_F)$
      end
    end
    $\boldsymbol{\theta} \leftarrow \boldsymbol{\theta} - g_K$
  end
end
return $\boldsymbol{\theta}$

3.2 Adversarial Training on Embeddings

Before presenting SPACE, we first introduce the vanilla adversarial training on embedding space.

Given an input code sequence, we first tokenize it into multiple sub-words using a Byte Pair Encoding Tokenizer (BPE, Sennrich et al. 2016), which is illustrated in the left part of Figure 1. After obtaining the sub-word sequence $\mathbb{X} = \{w_i\}_{i=1}^n$, we embeds them into a high-dimensional embedding space with an embedding look-up table ϕ: $E = \phi(\mathbb{X}) = \{e_i\}_{i=1}^n \in \mathcal{R}^{n \times d}$, where d is the dimension of the embedding space. At this point, we are supposed to find the optimal $\boldsymbol{\delta} \in \mathcal{R}^{n \times d}$ to maximize the inner part of Eq. (2), where $\mathcal{A}(\mathbb{X}, \boldsymbol{\delta}) = \phi(\mathbb{X}) + \boldsymbol{\delta}$ and $\|\boldsymbol{\delta}\|_F \le \epsilon$. This optimization is non-concave for neural networks, hence we suppose the loss function is locally linear and apply multiple gradient ascent steps to solve it:

$$
\begin{aligned}
g(\boldsymbol{\delta}_t) &= \nabla_{\boldsymbol{\delta}_t}\mathcal{L}(f_{\boldsymbol{\theta}}(\mathcal{A}(\mathbb{X}, \boldsymbol{\delta}_t)), y) \\
\boldsymbol{\delta}_{t+1} &= \Pi_{\|\boldsymbol{\delta}\|_F \le \epsilon}(\boldsymbol{\delta_t} + \eta g(\boldsymbol{\delta}_t)/\|g(\boldsymbol{\delta}_t)\|_F)
\end{aligned} \tag{3}
$$

Here Π is a projecting function that constrain $\boldsymbol{\delta}$ within the ϵ-ball. It is proved in literature that the above Projected Gradient Descent (PGD) algorithm is able to neatly solve the maximization problem (Madry et al., 2018).

In our implementation, we utilize the Free Large-Batch algorithm (FreeLB, Zhu et al. 2020) to improve the efficiency during training. The target model $f_{\boldsymbol{\theta}}$ is the PrLM Encoder (see the right part of Figure 1), which is CodeBERT or GraphCodeBERT in our experiments.

3.3 Semantic-Preserving Adversarial Training for Programming Languages

Vanilla adversarial training suffices to produce good virtual adversarial examples for natural language. However, it is not suitable for programming languages, which are grammatically and syntactically strict. The key problems here are two-fold. First, the keywords in programming languages are so sensitive that small perturbations on them could change the functionality of the original code snippet, resulting in label flipping (Sato et al., 2018). Second, different perturbations added to instances of the same identifier will lead to different embeddings for the same identifier, which breaks the grammar rule of programming languages. To avoid the first problem, the simplest and most efficient way is to add adversarial perturbations only on the embeddings of identifiers (e.g. names of functions, variables, and classes) and leave the keywords unchanged. For the second problem, we maintain the same adversarial perturbation for each identifier, making sure the embeddings of the same identifier are still identical after perturbing. In conclusion, we aim to find constrained perturbations that preserve the semantic meaning and grammatical correctness of programming languages.

Motivated by the above discussion, we propose Semantic-Preserving Adversarial Code Embeddings (SPACE). Specifically, for each identifier d_i that contains k_i sub-words $d_i = \{s_{ij}\}_{j=1}^{k_i}$, we maintain a specific perturbation $\boldsymbol{\delta}_i \in \mathcal{R}^{k_i \times d}$ for it. Each perturbation $\boldsymbol{\delta}_i$ is then copied and assigned to the positions of its corresponding identifier instances in the code snippet. As illustrated in the Adversarial Block of Figure 1, the BPE Tokenizer tokenizes the input code sequence into sub-word tokens, which are then embedded into high-dimensional word embeddings by a Sub-word Embedding Layer. After that, we maintain the differentiable perturbations for each sub-word that belongs to the same identifier, and add them to

the corresponding word embeddings and positional embeddings to form the distributed embeddings, which are finally further encoded by the PrLM Encoder. In Figure 1, different colors represent different identifiers and their shade indicates different sub-words. The differentiable perturbations are updated according to their gradients to the final loss using the gradient ascent algorithm, in order to maximize the negative impact of the perturbations to the model.

The whole training process is shown in Alg. (1). We first initialize the model parameters $\boldsymbol{\theta}$, then for every mini-batch in an epoch, we initialize the perturbations for all identifiers, $\boldsymbol{\delta}_0^i$, to zero. After that, K steps of adversarial training will be conducted. In each step t, the perturbations of each identifier $\boldsymbol{\delta}_{t-1}^i$ will be assigned to the positions of their corresponding identifier instances to form $\boldsymbol{\delta}_{t-1}$, which is then used to attack the model in order to accumulate the gradient of $\boldsymbol{\theta}$ and update each $\boldsymbol{\delta}_t^i$ for the next step. After K steps, we update $\boldsymbol{\theta}$ using the accumulated gradient g_K. Experimental results and analysis in the following sections demonstrate that the obtained model is better than normal training and stays robust against adversarial attacks.

4 Experiments

To comprehensively evaluate our SPACE, we conduct experiments on three types of source code comprehension tasks: code defect detection task for classification style, natural language code search task for retrieval style, and code question answering task for generative style. Meanwhile, they are code-to-code, natural-language-to-code and code-to-natural-language tasks, respectively. In this section, we will first introduce each task and its corresponding dataset, then present the results of them.

4.1 Tasks and Datasets

Code Defect Detection requires a model to predict whether a code snippet contains malicious parts that may attack software systems. We conduct this experiment on the Defects4J dataset (Zhou et al., 2019), which defines defect detection as a binary classification task.

Natural Language Code Search is to retrieve the most relevant code snippet from a database given a natural language query, which is helpful to reuse codes when developing new systems. We conduct this experiments on the CodeSearchNet dataset (Husain et al., 2019) and follow Guo et al. (2021) to process the data. This dataset contains six subsets for different languages: Ruby, JavaScript, Go, Python, Java, and PHP.

Code Question Answering aims at learning a model to answer questions with regard to a snippet of code. In this setting, the answer should be generated as natural language by a decoder rather than copied from the source code. We conduct this experiment on CodeQA dataset (Liu and Wan, 2021), which contains two subsets for Java and Python.

For statistics about these datasets, please refer to Appendix A. For more details about our experimental settings (e.g. hyper-parameters, environments, and hard-wares), please refer to Appendix B.

Model	Accuracy	
	CodeBERT	GraphCodeBERT
baseline	62.08	64.06
+ADV.	64.85	64.45
+SPACE	**66.01**	**66.12**
+augmentation	63.25	63.47
+rand. ADV.	64.02	64.41
+rand. SPACE	64.57	65.08

Table 1: Results on Defects4J dataset, where the upper half presents the main results based on two PrLMs and the lower half presents ablation results.

4.2 Experimental Results

Results on Defect Detection are shown in Table 1, we follow Lu et al. (2021) to use accuracy score as our evaluation metric. In this section, we focus on the upper half of Table 1, where the main results are presented. The baselines here are pure CodeBERT and GraphCodeBERT with a linear classifier on top and +ADV. is the abbreviation for the vanilla adversarial training method. We see from the table that vanilla adversarial training can improve the performance to a certain extent, while our SPACE is constantly better than it and outperforms strong baselines by large margins: over **3.93%** and **1.96%** on CodeBERT and GraphCodeBERT, respectively. This observation demonstrates that our SPACE, which takes the advantage of source code characteristics, is superior to vanilla adversarial training on all tokens.

Results on Natural Language Code Search are tabulated in Table 2, where the evaluation metric is Mean Reciprocal Rank (MRR) following Husain et al. (2019). We see from the table that our SPACE

Model	AdvTest	CodeSearch						
	Python	Ruby	JavaScript	Go	Python	Java	PHP	Average
CodeBERT	27.2	67.9	62.0	88.2	67.2	67.6	62.8	69.3
+ADV.	28.3	68.5	62.3	88.5	67.5	67.6	62.6	69.5(↑ 0.2)
+SPACE	**32.6**	**69.4**	**62.7**	**89.1**	**68.3**	**68.5**	**63.3**	**70.2 (↑ 0.9)**
GraphCodeBERT	35.2	70.3	64.4	89.7	69.2	69.1	64.9	71.3
+ADV.	37.8	70.7	64.9	89.7	69.5	69.4	64.7	71.5 (↑ 0.2)
+SPACE	**41.3**	**72.1**	**65.8**	**90.1**	**70.6**	**70.3**	**65.4**	**72.4 (↑ 1.1)**

Table 2: Results on CodeSearchNet dataset.

Model	Python					Java				
	BLEU	ROUGE	METEOR	EM	F1	BLEU	ROUGE	METEOR	EM	F1
CodeBERT	34.86	30.28	12.51	4.93	31.56	32.40	28.22	10.10	6.20	29.20
+ADV.	36.15	32.84	14.07	5.84	34.08	33.03	29.13	10.60	6.24	30.02
+SPACE	**36.47**	**33.03**	**14.14**	**6.14**	**34.38**	**33.49**	**29.64**	**11.01**	**6.42**	**30.66**
GraphCodeBERT	35.95	32.31	13.39	5.86	33.52	33.22	29.24	10.78	6.52	30.22
+ADV.	36.57	33.29	14.15	6.24	34.52	33.84	30.41	11.33	6.76	31.42
+SPACE	**37.03**	**33.97**	**14.65**	**6.41**	**35.25**	**34.11**	**30.96**	**11.68**	**6.92**	**32.01**

Table 3: Results on CodeQA dataset.

outperforms vanilla adversarial training and baselines in all languages, demonstrating the universality of our method, which is language-agnostic and can be used to boost the performance of models for different programming languages. Beside, Lu et al. (2021) provide an adversarial test set for Python, which replaces all arguments of a function with $\{arg\_0, arg\_1, \ldots, arg\_n\}$. We also test our method in this adversarial test set to preliminarily verify the robustness of SPACE. In this setting, we observe slight improvements on robustness using vanilla adversarial training, yet they are still far from satisfactory. In contrast, our SPACE boosts the performance on this test set by **5.4%** and **6.1%** over the baseline of CodeBERT and GraphCodeBERT, respectively. It is worth noting that the calculation of MRR score in this adversarial test set is different from the clean test set, hence the results are not directly comparable with the original ones. For the robustness test with stronger adversarial attacks, we present it in Section 5.2.

Results on Code Question Answering are shown in Table 3, where we use PrLMs (CodeBERT and GraphCodeBERT) as the encoder to build representations for the source code and its corresponding question, then apply a Transformer Decoder (Vaswani et al., 2017) to generate answers. Following Liu and Wan (2021), we use BLEU, ROUGE-L, METEOR, EM, and F1 as the evaluation metrics. BLEU computes the n-gram overlapping between candidates and references, while METEOR is an improvement of BLEU that considers some grammatical and lexical information, such as synonyms, roots, and affixes. ROUGE-L computes the longest common subsequence (LCS) between the candidates and references, then calculates the F1 ratio by measuring the recall over references and precision over candidates. EM stands for Exactly Match and F1 is the character-wise F1 score between candidates and references. Similar observations can be seen in Table 3 that SPACE yields the best result compared with baselines and vanilla adversarial training for both Python and Java dataset, which further demonstrates the effectiveness of our method in generative tasks.

5 Analysis

In this section, we conduct a series of experiments and analyses to get a more in-depth understanding of SPACE and its robustness against various adversarial attacks, including state-of-the-art ones.

5.1 Ablation Study

There are multiple factors that could be the potential reasons for the performance improvements: more virtual augmented samples by the FreeLB algorithm, or simply the virtual perturbations on embedding space. To investigate whether these factors could yield performance improvements as good as SPACE, we conduct additional experiments on each of these factors.

Notice that in Alg. (1), we accumulate the gradient of $\boldsymbol{\theta}$ in each of the K steps, resulting in K times more *virtual augmented samples* compared with the original training samples. To simulate the augmented samples and also to explore whether *ac-*

tual data augmentation would be enough to boost the performance, we perform semantic-preserving code transformations for each training sample and get a K times larger augmented training set. After that, we average the gradient of each K-samples to update the parameters similar in Alg. (1). Results on this setting is presented in the fourth row of Table 1, where we observe a slight performance improvement on CodeBERT and a performance drop on GraphCodeBERT. This observation demonstrates that simple data augmentation is not able to enhance the model as effectively as our SPACE. Besides, training with actual augmented samples via code transformations can even hurt the performance, which is also observed in the paper of Ramakrishnan et al. (2020).

Now that actual augmented samples can not help, do the improvements just come from virtual augmented samples with any kind of perturbations on the embedding space? To answer this question, we add random perturbations on all tokens (denoted as *+rand. ADV.* in Table 1) or with the same semantic-preserving way as SPACE (denoted as *+rand. SPACE*) to see their effect on baseline models. Note that we also perform random perturbations K times and average the gradient of the obtained K samples, for a fair comparison. As shown in the last two rows of Table 1, random perturbations are sub-optimal compared with those by solving the inner maximization problem, resulting in just slight performance gains on both models. In addition, we also observe that random perturbations added with the semantic-preserving way are constantly better than those on all tokens, which further confirms the motivation of our SPACE. In conclusion, gradient-based perturbations working in tandem with the semantic-preserving mechanism (i.e., SPACE) result in the best performance.

5.2 Robustness against Advanced Attacks

In Section 4.2, we preliminarily demonstrate the robustness of SPACE in the Python adversarial test set provided by Lu et al. (2021). To test SPACE with stronger adversarial attacks, we conduct comprehensive experiments with various kinds of source code attacks, including state-of-the-art ones.

We first introduce two advanced attacking methods we experiment with. Zhang et al. (2020) propose Metropolis-Hastings Modifier (MHM) to attack code classification models, which defines identifier renaming as a sampling problem. MHM is a classical Markov chain Monte Carlo sampling approach, which generates a sequence of adversarial examples given the current stationary distribution and the probability output by the classification model. Yang et al. (2022) present ALERT (Naturalness Aware Attack) that considers the natural semantic of generated adversarial examples. They make use of the masked language prediction function of PrLMs to produce a ranked list of potential substitutes for each token, and design genetic algorithms to search for the optimal combination of variables and corresponding substitutes. ALERT is the current state-of-the-art attacking method that yields a high Attack Success Rate (ASR) on strong source code comprehension models. Meanwhile, they are the first to attack PrLMs for code.

Model	Attack Success Rate		
	MHM	Greedy-A	ALERT
baseline	55.17	71.89	76.95
+augmentation	56.24	69.91	75.91
+rand. ADV.	54.89	70.69	74.75
+ADV.	52.36	69.34	73.53
+rand. SPACE	50.27	65.81	69.54
+SPACE	**44.32**	**61.01**	**65.89**

Table 4: Results of advanced attacks on Defects4J dataset of GraphCodeBERT, where **lower ASRs are better**. Greedy-A means ALERT attack with Greedy Algorithm while ALERT stands for the one with Genetic Algorithm.

Results of MHM and ALERT attacks are tabulated in Table 4, where we use ASR as the evaluation metric following Yang et al. (2022). Suppose N^+ represents the number of correctly predicted samples on the original clean test set, while N^- stands for the number of those that become incorrect after being attacked. ASR is then calculated as $\text{ASR} = \frac{N^-}{N^+}$. According to the definition, lower ASR means stronger robustness against adversarial attacks. We see from the table that our SPACE is the most robust one among all models, which reduces the ASR of GraphCodeBERT by over **10%** on all attacking methods. The second row of the table tells us that training models with *actual* augmented samples by semantic-preserving code transformations can hardly help to improve the robustness against advanced attacking methods. When it comes to training with *virtual* adversarial samples, Table 4 shows that gradient-based perturbations (*+ADV., +SPACE*) are constantly better than ran-

Model	CodeSearch-Adv						
	Ruby	JavaScript	Go	Python	Java	PHP	Average
CodeBERT	62.6(↓ 5.3)	54.8(↓ 7.2)	84.1(↓ 4.1)	61.1(↓ 6.1)	62.1(↓ 5.5)	58.2(↓ 4.6)	63.8(↓ 5.5)
+ADV.	63.3(↓ 5.2)	55.5(↓ 6.8)	85.6(↓ 2.9)	62.1(↓ 5.4)	62.9(↓ 4.7)	58.7(↓ 3.9)	64.7(↓ 4.8)
+SPACE	**67.1(↓ 2.3)**	**59.5(↓ 3.2)**	**88.5(↓ 0.6)**	**65.8(↓ 2.5)**	**66.5(↓ 2.0)**	**62.3(↓ 1.0)**	**68.3(↓ 1.9)**
GraphCodeBERT	63.2(↓ 7.1)	57.2(↓ 7.2)	84.5(↓ 5.2)	62.5(↓ 6.7)	62.6(↓ 6.5)	58.2(↓ 6.7)	64.7(↓ 6.6)
+ADV.	65.3(↓ 5.4)	59.1(↓ 5.8)	86.9(↓ 2.8)	64.9(↓ 4.6)	64.8(↓ 4.6)	61.3(↓ 3.4)	67.1(↓ 4.4)
+SPACE	**69.8(↓ 3.2)**	**62.9(↓ 2.3)**	**89.2(↓ 2.9)**	**67.8(↓ 0.9)**	**68.1(↓ 2.2)**	**64.1(↓ 1.3)**	**70.3(↓ 2.1)**

Table 5: Adversarial results on CodeSearchNet dataset, where the downward arrows indicate the performance drop compared with the original clean test set, smaller drop means better robustness.

Model	Python				
	BLEU	ROUGE	METEOR	EM	F1
GraphCodeBERT	32.40(↓ 3.55)	27.95(↓ 4.36)	10.78(↓ 2.61)	4.21(↓ 1.65)	28.64(↓ 4.88)
+ADV.	33.32(↓ 3.25)	28.94(↓ 4.35)	11.69(↓ 2.46)	4.74(↓ 1.50)	29.98(↓ 4.54)
+SPACE	**34.91(↓ 2.12)**	**30.52(↓ 3.45)**	**12.37(↓ 2.28)**	**5.21(↓ 1.20)**	**31.72(↓ 3.53)**
Model	Java				
	BLEU	ROUGE	METEOR	EM	F1
GraphCodeBERT	30.12(↓ 3.10)	25.28(↓ 3.96)	8.48(↓ 2.30)	4.96(↓ 1.56)	26.36(↓ 3.86)
+ADV.	31.21(↓ 2.63)	26.95(↓ 3.46)	9.35(↓ 1.98)	5.42(↓ 1.34)	27.88(↓ 3.54)
+SPACE	**32.86(↓ 1.25)**	**28.34(↓ 2.62)**	**10.26(↓ 1.42)**	**6.18(↓ 0.74)**	**29.33(↓ 2.65)**

Table 6: Adversarial results on CodeQA dataset, where the downward arrows means the same in Table 5.

dom perturbations (+*rand. ADV., +rand. SPACE*), and adding perturbations under the constraint of preserving semantics (+*rand. SPACE, +SPACE*) are constantly better than simply adding them on all tokens (+*rand. ADV., +ADV.*). These observations further demonstrate the significance of combining gradient-based adversarial training with the data characteristics of programming languages.

We explain the superiority of SPACE from the following two perspectives. Firstly, the gradient-based perturbations are stronger than the random ones since they are basically the worst-case ones. Training with worse adversarial examples is more likely to generate more robust models. Secondly, performing semantic-preserving adversarial perturbations is consistent with the data characteristics of programming languages, which maintains grammatical correctness and avoids label flipping problem caused by small perturbations on keywords.

5.3 Robustness against Code Transformations

MHM and ALERT require the probability output by classification models, thus incompatible with retrieval and generative tasks. To evaluate the robustness of SPACE on these tasks and also to test its robustness under various semantic-preserving code transformations, we select the top-three effective code transformations in the experiments of Ramakrishnan et al. (2020) to construct a model-agnostic adversarial testing set for both CodeSearchNet and CodeQA. Note that on the CodeQA dataset, some questions and answers contain identifier names on the corresponding code snippets. Therefore, when performing the identifier-renaming transformation, we also rename those identifiers on questions and answers to ensure their correctness.

Results on the adversarial test sets of both datasets are presented in Table 5 and Table 6, respectively. Similar observations can be seen in both tables that SPACE is the most robust model against attacks by the combination of various semantic-preserving code transformations. On the CodeSearchNet dataset, our SPACE reduces the performance drop by **3.6%** and **4.5%** on average over all languages for CodeBERT and GraphCodeBERT, respectively, compared with the baseline models. We also notice that vanilla adversarial training can mitigate the impact of adversarial attacks, yet only to a slight extent (typically less than 0.5%). We infer that this is because vanilla adversarial training on all tokens is not in line with the legal semantic-preserving code transformations, since it breaks the grammatical correctness and affects the semantic meaning of keywords, which are vital and sensitive in code texts. On the contrary, our SPACE only perturbs the embedding of identifiers and leaves the keywords unchanged, which preserves the grammatical correctness and functionality of codes.

6 Conclusion

In order to fill the gap of performance and robustness for source code comprehension models,

we propose Semantic-Preserving Adversarial Code Embeddings (SPACE) in this paper. SPACE takes the advantage of adversarial training and is specialized to fit the data characteristics of programming languages. To the best of our knowledge, we are the first to explore the efficacy of adversarial training on the continuous embedding space for source code comprehension tasks. Thorough experiments and analysis on three kinds of tasks have justified the effectiveness and universality of our model to improve both performance and robustness for source code comprehension models.

References

Pavol Bielik and Martin T. Vechev. 2020. Adversarial robustness for code. In *Proceedings of the 37th International Conference on Machine Learning, ICML 2020, 13-18 July 2020, Virtual Event*, volume 119 of *Proceedings of Machine Learning Research*, pages 896–907. PMLR.

Nghi D. Q. Bui, Yijun Yu, and Lingxiao Jiang. 2021. Self-supervised contrastive learning for code retrieval and summarization via semantic-preserving transformations. In *SIGIR '21: The 44th International ACM SIGIR Conference on Research and Development in Information Retrieval, Virtual Event, Canada, July 11-15, 2021*, pages 511–521. ACM.

Luca Buratti, Saurabh Pujar, Mihaela A. Bornea, J. Scott McCarley, Yunhui Zheng, Gaetano Rossiello, Alessandro Morari, Jim Laredo, Veronika Thost, Yufan Zhuang, and Giacomo Domeniconi. 2020. Exploring software naturalness through neural language models. *CoRR*, abs/2006.12641.

Yong Cheng, Lu Jiang, and Wolfgang Macherey. 2019. Robust neural machine translation with doubly adversarial inputs. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 4324–4333, Florence, Italy. Association for Computational Linguistics.

Kevin Clark, Minh-Thang Luong, Quoc V. Le, and Christopher D. Manning. 2020. ELECTRA: pre-training text encoders as discriminators rather than generators. In *8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, April 26-30, 2020*. OpenReview.net.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, Minneapolis, Minnesota. Association for Computational Linguistics.

Yangruibo Ding, Luca Buratti, Saurabh Pujar, Alessandro Morari, Baishakhi Ray, and Saikat Chakraborty. 2021. Contrastive learning for source code with structural and functional properties. *CoRR*, abs/2110.03868.

Zhangyin Feng, Daya Guo, Duyu Tang, Nan Duan, Xiaocheng Feng, Ming Gong, Linjun Shou, Bing Qin, Ting Liu, Daxin Jiang, and Ming Zhou. 2020. CodeBERT: A pre-trained model for programming and natural languages. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 1536–1547, Online. Association for Computational Linguistics.

Ian J. Goodfellow, Jonathon Shlens, and Christian Szegedy. 2015. Explaining and harnessing adversarial examples. In *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*.

Daya Guo, Shuo Ren, Shuai Lu, Zhangyin Feng, Duyu Tang, Shujie Liu, Long Zhou, Nan Duan, Alexey Svyatkovskiy, Shengyu Fu, Michele Tufano, Shao Kun Deng, Colin B. Clement, Dawn Drain, Neel Sundaresan, Jian Yin, Daxin Jiang, and Ming Zhou. 2021. Graphcodebert: Pre-training code representations with data flow. In *9th International Conference on Learning Representations, ICLR 2021, Virtual Event, Austria, May 3-7, 2021*. OpenReview.net.

Hamel Husain, Ho-Hsiang Wu, Tiferet Gazit, Miltiadis Allamanis, and Marc Brockschmidt. 2019. Codesearchnet challenge: Evaluating the state of semantic code search. *CoRR*, abs/1909.09436.

Paras Jain, Ajay Jain, Tianjun Zhang, Pieter Abbeel, Joseph Gonzalez, and Ion Stoica. 2021. Contrastive code representation learning. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 5954–5971, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Aditya Kanade, Petros Maniatis, Gogul Balakrishnan, and Kensen Shi. 2020a. Learning and evaluating contextual embedding of source code. In *Proceedings of the 37th International Conference on Machine Learning, ICML 2020, 13-18 July 2020, Virtual Event*, volume 119 of *Proceedings of Machine Learning Research*, pages 5110–5121. PMLR.

Aditya Kanade, Petros Maniatis, Gogul Balakrishnan, and Kensen Shi. 2020b. Pre-trained contextual embedding of source code. *CoRR*, abs/2001.00059.

Chenxiao Liu and Xiaojun Wan. 2021. CodeQA: A question answering dataset for source code comprehension. In *Findings of the Association for Computational Linguistics: EMNLP 2021*, pages 2618–2632, Punta Cana, Dominican Republic. Association for Computational Linguistics.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Roberta: A robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692*.

Shuai Lu, Daya Guo, Shuo Ren, Junjie Huang, Alexey Svyatkovskiy, Ambrosio Blanco, Colin B. Clement, Dawn Drain, Daxin Jiang, Duyu Tang, Ge Li, Lidong Zhou, Linjun Shou, Long Zhou, Michele Tufano, Ming Gong, Ming Zhou, Nan Duan, Neel Sundaresan, Shao Kun Deng, Shengyu Fu, and Shujie Liu. 2021. Codexglue: A machine learning benchmark dataset for code understanding and generation. *CoRR*, abs/2102.04664.

Aleksander Madry, Aleksandar Makelov, Ludwig Schmidt, Dimitris Tsipras, and Adrian Vladu. 2018. Towards deep learning models resistant to adversarial attacks. In *6th International Conference on Learning Representations, ICLR 2018, Vancouver, BC, Canada, April 30 - May 3, 2018, Conference Track Proceedings*. OpenReview.net.

Takeru Miyato, Andrew M. Dai, and Ian J. Goodfellow. 2017. Adversarial training methods for semi-supervised text classification. In *5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Conference Track Proceedings*. OpenReview.net.

Long N. Phan, Hieu Tran, Daniel Le, Hieu Nguyen, James T. Anibal, Alec Peltekian, and Yanfang Ye. 2021. Cotext: Multi-task learning with code-text transformer. *CoRR*, abs/2105.08645.

Goutham Ramakrishnan, Jordan Henkel, Zi Wang, Aws Albarghouthi, Somesh Jha, and Thomas W. Reps. 2020. Semantic robustness of models of source code. *CoRR*, abs/2002.03043.

Motoki Sato, Jun Suzuki, Hiroyuki Shindo, and Yuji Matsumoto. 2018. Interpretable adversarial perturbation in input embedding space for text. In *Proceedings of the Twenty-Seventh International Joint Conference on Artificial Intelligence, IJCAI 2018, July 13-19, 2018, Stockholm, Sweden*, pages 4323–4330. ijcai.org.

Rico Sennrich, Barry Haddow, and Alexandra Birch. 2016. Neural machine translation of rare words with subword units. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1715–1725, Berlin, Germany. Association for Computational Linguistics.

Jacob M. Springer, Bryn Marie Reinstadler, and Una-May O'Reilly. 2020. STRATA: building robustness with a simple method for generating black-box adversarial attacks for models of code. *CoRR*, abs/2009.13562.

Shashank Srikant, Sijia Liu, Tamara Mitrovska, Shiyu Chang, Quanfu Fan, Gaoyuan Zhang, and Una-May O'Reilly. 2021. Generating adversarial computer programs using optimized obfuscations. In *9th International Conference on Learning Representations, ICLR 2021, Virtual Event, Austria, May 3-7, 2021*. OpenReview.net.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, December 4-9, 2017, Long Beach, CA, USA*, pages 5998–6008.

Xin Wang, Yasheng Wang, Fei Mi, Pingyi Zhou, Yao Wan, Xiao Liu, Li Li, Hao Wu, Jin Liu, and Xin Jiang. 2021. Syncobert: Syntax-guided multi-modal contrastive pre-training for code representation. *arXiv preprint arXiv:2108.04556*.

Hongqiu Wu, Hai Zhao, and Min Zhang. 2021. Code summarization with structure-induced transformer. In *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*, pages 1078–1090, Online. Association for Computational Linguistics.

Zhilin Yang, Zihang Dai, Yiming Yang, Jaime Carbonell, Russ R Salakhutdinov, and Quoc V Le. 2019. Xlnet: Generalized autoregressive pretraining for language understanding. In *Advances in Neural Information Processing Systems*, volume 32. Curran Associates, Inc.

Zhou Yang, Jieke Shi, Junda He, and David Lo. 2022. Natural attack for pre-trained models of code. *CoRR*, abs/2201.08698.

Noam Yefet, Uri Alon, and Eran Yahav. 2020. Adversarial examples for models of code. *Proc. ACM Program. Lang.*, 4(OOPSLA):162:1–162:30.

Huangzhao Zhang, Zhuo Li, Ge Li, Lei Ma, Yang Liu, and Zhi Jin. 2020. Generating adversarial examples for holding robustness of source code processing models. In *The Thirty-Fourth AAAI Conference on Artificial Intelligence, AAAI 2020, New York, NY, USA, February 7-12, 2020*, pages 1169–1176. AAAI Press.

Yaqin Zhou, Shangqing Liu, Jing Kai Siow, Xiaoning Du, and Yang Liu. 2019. Devign: Effective vulnerability identification by learning comprehensive program semantics via graph neural networks. In *Advances in Neural Information Processing Systems 32: Annual Conference on Neural Information Processing Systems 2019, NeurIPS 2019, December 8-14, 2019, Vancouver, BC, Canada*, pages 10197–10207.

Chen Zhu, Yu Cheng, Zhe Gan, Siqi Sun, Tom Goldstein, and Jingjing Liu. 2020. Freelb: Enhanced adversarial training for natural language understanding. In *8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, April 26-30, 2020*. OpenReview.net.

Language	Training Samples	Dev Queries	Testing Queries	Code Database
Go	167,288	7,325	8,122	28,120
Java	164,923	5,183	10,955	40,347
JavaScript	58,025	3,885	3,291	13,981
PHP	241,241	12,982	14,014	52,660
Python	251,820	13,914	14,918	43,827
Ruby	24,927	1,400	1,261	4,360

Table 7: Statistics of CodeSearchNet Dataset.

	Java	Python
Training Samples	95,778	56,085
Dev Samples	12,000	7,000
Testing Samples	12,000	7,000

Table 8: Statistics of CodeQA Dataset.

A Dataset Statistics

A.1 Defects4J Dataset

To propose a high-quality dataset for the defect detection task, Zhou et al. (2019) invested a team of security to manually label the data. They first collect functions from 4 large C-language open-source projects: Linux Kernel, QEMU, Wireshark, and FFmpeg, then ask the team of four professional security researchers to perform a two0round data labeling and cross-verification. The final dataset contains $21,000$ samples for training, $2,700$ samples for validation and $2,700$ samples for testing.

A.2 CodeSearchNet Dataset

This dataset contains six programming languages, which is naturally suitable to test language-agnostic methods. In our experiments, we use the filtered dataset by Guo et al. (2021), where for each language, it contains a training set denoted as $\{code, query\}_{i=1}^{N}$. For validation and testing, the answer is supposed to be retrieved from a code database given the query. The detailed statistics of this dataset are tabulated in Table 7.

A.3 CodeQA Dataset

CodeQA contains two subsets for Java and Python, respectively. The detailed statistics of this dataset are shown in Table 8.

B Experimental Settings

B.1 Environments

We conduct our experiment with Pytorch implementation and CUDA version 11.4. For Defects4J and CodeQA, we train our model on an RTX 3090 24GB GPU, and for CodeSearchNet, we train our model on a Tesla V100 32GB GPU. The default settings of our experiments require about 20GB of GPU memory. Our experiments are single-runs over the same seed due to the limits of computational resources.

B.2 Evaluation on CodeSearchNet

Note that we use the same settings for evaluation on the CodeSearchNet dataset following Guo et al. (2021), where during validation and testing, the answer should be retrieved from the whole code database, rather than from 1,000 candidates as in the original paper of Husain et al. (2019). We use this setting for a fair comparison with the baselines of CodeBERT and GraphCodeBERT.

B.3 Hyper-parameters

We search for the best hyper-parameters on the validation set of each dataset and test the best model on the test set. We list the best hyper-parameters we use in our experiments here.

B.3.1 Defect Detection

For both CodeBERT and GraphCodeBERT, we set the learning rate to 2e-5, batch size to 16, and the number of epochs to 5. For the inner gradient ascent of adversarial training, we set its steps to 3 steps and its learning rate to 5e-4. For CodeBERT, the maximum sequence length is 512. While for GraphCodeBERT, which requires extra data flow information, we set the maximum sequence length for code to 384 and the maximum data flow length to 128.

B.3.2 Natural Language code Search

For normal hyper-parameters, we set the maximum code sequence length to 320, data flow length to 64, and maximum query length to 128. The learning rate, batch size, and the number of training epochs

Language	CodeBERT		GraphCodeBERT	
	learning rate	steps	learning rate	steps
Ruby	5e-4	3	5e-5	3
JavaScript	5e-5	3	5e-5	3
Go	5e-5	3	5e-5	3
Python	5e-5	3	5e-4	3
Java	5e-5	3	5e-5	3
PHP	5e-5	3	1e-4	3

Table 9: Hyper-parameters for Adversarial Training on CodeSearchNet Dataset.

are set to 2e-5, 32, and 10, respectively, following the original settings of Guo et al. (2021).

For hyper-parameters of adversarial training (i.e., adversarial steps and adversarial learning rate), we tune them for each language, which is presented in Table 9.

B.3.3 Question Answering over Source Code

For normal hyper-parameters, the learning rate, batch size, and the number of training epochs are set to 1e-4, 64, and 20, respectively. For CodeBERT, the maximum code sequence length is set to 256, while for GraphCodeBERT, the maximum code sequence length and data flow length is set to 240 and 60, respectively.

For hyper-parameters of adversarial training, we set the adversarial steps to 3 for all languages on both CodeBERT and GraphCodeBERT. As for the adversarial learning rate, we set it to 5e-5 for all languages on GraphCodeBERT, and 5e-4, 1e-4 for Java and Python on CodeBERT, respectively.

Continually Detection, Rapidly React: Unseen Rumors Detection based on Continual Prompt-Tuning

Yuhui Zuo[1], **Wei Zhu**[2], **Guoyong Cai**[1],
[1]Guilin University of Electronic Technology, Guilin, China
[2]East China Normal University, Shanghai, China
zzuuoo2019ok@gmail.com, ccgycai@guet.edu.cn, wzhu@stu.ecnu.edu.cn

Abstract

Since open social platforms allow for a large and continuous flow of unverified information, rumors can emerge unexpectedly and spread quickly. However, existing rumor detection (RD) models often assume the same training and testing distributions and can not cope with the continuously changing social network environment. This paper proposed a Continual Prompt-Tuning RD (CPT-RD) framework, which avoids catastrophic forgetting (CF) of upstream tasks during sequential task learning and enables bidirectional knowledge transfer between domain tasks. Specifically, we propose the following strategies: (a) Our design explicitly decouples shared and domain-specific knowledge, thus reducing the interference among different domains during optimization; (b) Several technologies aim to transfer knowledge of upstream tasks to deal with emergencies; (c) A task-conditioned prompt-wise hypernetwork (TPHNet) is used to consolidate past domains. In addition, CPT-RD avoids CF without the necessity of a rehearsal buffer. Finally, CPT-RD is evaluated on English and Chinese RD datasets and is effective and efficient compared to prior state-of-the-art methods. [1]

1 Introduction

Online platforms such as social media are facing new and ever-evolving cyber threats at the information level — rumor. A rumor is an unconfirmed claim related to an object, event, or issue of public concern that is spread when its integrity is unknown (Guo et al., 2020). It is very necessary to study automated RD, because rumors are extremely harmful to society and manual detection is time-consuming and labor-intensive (Oshikawa et al., 2018).

However, automated RD has significant challenges and still faces the following difficulties:

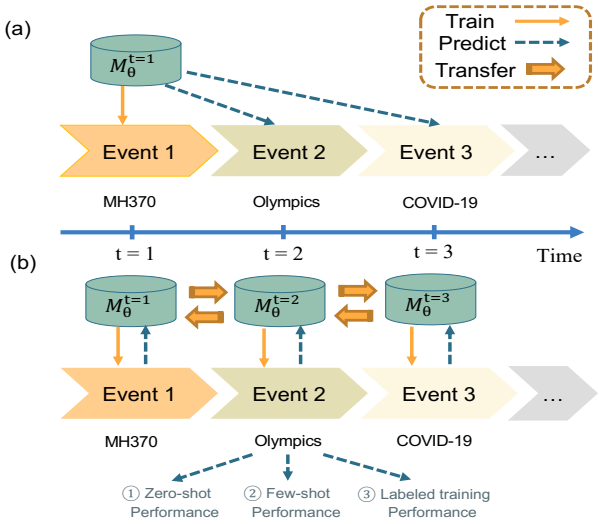

Figure 1: Illustration of the RD model training procedure in the traditional static (a) and continual dynamic event transfer (b) setup.

First, rumors are highly event domain-specific (Wang et al., 2018), each event domain may have a different input text distribution and fraudulent intent. Second, detecting rumors at their early stage of spreading faces the problem of insufficient labeled samples (Zhou et al., 2020). Third, rumor detectors operating on online social platforms often encounter continuous event domain changes. This poses a significant challenge to existing RD models.

Previous work (Wang et al., 2018; Bian et al., 2020; Zhang et al., 2021; Lin et al., 2021; Ben-David et al., 2021) usually assume the same distribution of training and testing data, and have difficulty coping with changing social network environments. In Fig.1 (a), there are no updates to the model regardless of how many unseen events will appear in the future, which requires the unreasonable assumption that the model's generalization capability is enough. In addition, as the social network environment changes, the original training set data will become outdated (Lee et al., 2021), so that the model's ability on more recent

[1]Innovation Project of Guangxi Graduate Education

Proceedings of the 29th International Conference on Computational Linguistics, pages 3029–3041
October 12–17, 2022.

events will diminish. For example, the COVID-19 pandemic caused massive trouble for existing RD models (Lu et al., 2021; Patwa et al., 2021b).

In social media, the ideal RD model must continuously detect event stream and react rapidly to emergencies. To study this ability, we propose a continual dynamic event transfer (CDET) setup (illustrated in Fig.1 (b)), which makes the RD model dynamically updates the parameters θ over a series of sequential events and assume that each event goes through three stages from burst to normalization: zero-shot stage with no samples, few-shot stage with a small number of samples, and full-shot stage with large-scale labeled samples. This is because rumors must be detected early to avoid the social harm caused by their spread, but there are only a few or no labeled samples in the early stages of an event.

Existing RD models still face many challenges in the CDET setup: (1) **Catastrophic forgetting:** When a neural model is trained in a sequence of tasks, the downstream tasks may catastrophically interfere with the upstream tasks. (2) **Knowledge transfer and accumulation:** Transfer the knowledge learned from upstream tasks for rapid generalization, accumulating knowledge from downstream tasks to better cope with upstream tasks; (3) **Parameter explosion:** Previous research (Wang et al., 2021a; Ke et al., 2021a) often require dynamically expanding neural modules for each task, which is undoubtedly aggravating for pre-trained language models (PLM) with billions of parameters, with limited memory; (4) **Data privacy:** After learning a task, training data is usually discarded due to user privacy concerns (Chen and Liu, 2018). This requires models to share learned parameters, rather than saving data to retrain the model.

To address the above challenges, a novel framework called **C**ontinual **P**rompt-**t**uning RD (CPT-RD) has been proposed. Technically, CPT-RD can be seen as a continuously migrated version of P-tuning v2 (Liu et al., 2021b). From Fig.2, we can clearly decouple domain-specific knowledge (tuning parameter) and shared knowledge (frozen parameter). This provides the basis for achieving memory of task-specific parameters and bidirectional domain knowledge transfer. Bidirectional knowledge transfer includes: (1) **Forward Knowledge Transfer**: CPT-RD has various prompt initialization strategies to adapt fast

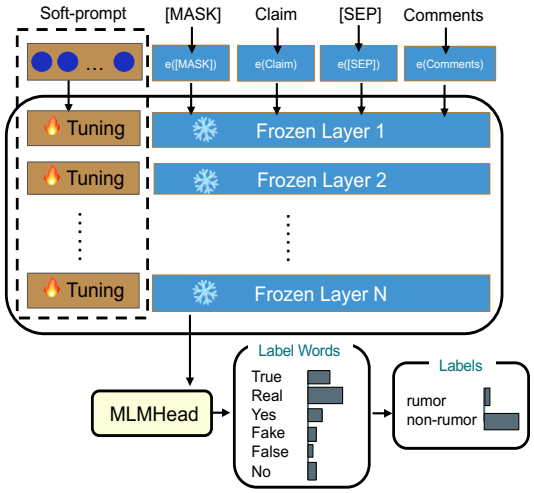

Figure 2: Illustration of deep prompt tuning for RD.

to rumors of emergencies. (2) **Backward Knowledge Transfer**: A task-conditioned prompt-wise hypernetwork (TPHNet) learns latent distribution of soft-prompts, encouraging CPT-RD to accumulate knowledge in sequential events while avoiding data replay.

We collected RD datasets with 14 different domains for both, English and Chinese. Through empirical analysis, we find that CPT-RD essentially avoids catastrophic forgetting. On the English dataset, the knowledge transfer indicators FWT and BWT (Lopez-Paz and Ranzato, 2017) achieve positive indicators of 23.9% and 0.9%, respectively. Finally, the effectiveness of our improvement is demonstrated through ablation experiments.

Our main contributions are: (1) We propose a CDET setup in RD for evaluating rapid generalization and continual detection problems simultaneously. (2) To completely avoid the CF encountered in continual detection, we optimize and store domain-specific soft-prompt for each event domain and use it selectively. (3) We propose various forward knowledge transfer strategies to deal with early emergency rumors and accumulate detection experience through TPHNet for backward knowledge transfer. (4) Our experiments on the collected Chinese and English social media RD datasets demonstrate the superior performance and efficiency of our proposed method.

2 Related work

2.1 Rumors detection

Social media has gained much attention as a source of research rumors, but existing RD meth-

ods still perform poorly in the face of unknown events and struggle to consistently respond to the dynamic and changing social network environment (Zubiaga et al., 2018; Guo et al., 2020). Wang et al. (2018) proposed an event adversarial neural network (EANN) which extracts event-invariant features by removing event-specific features. Lu et al. (2021) consider unseen event RD as a few-shot learning problem, and Wang et al. (2021b) apply meta-learning to obtain the optimal generalization initial parameters. Ben-David et al. (2021) used prompt-tuning to consider RD as domain adaptation and committed to improving out-of-distribution issues. The above methods can alleviate the problem of domain drift, but they are all learned with offline static settings. Lee et al. (2021) considered the continual detection capability on online social media and applied rehearsal-based replay techniques to resist CF. However, this approach does not consider generalizing to the unseen domain quickly and faces problems such as inference efficiency and data privacy. Different from the above methods, our proposed model considers both fast generalization and knowledge transfer for continual detection and is not based on any data replay.

2.2 Prompt-based tuning

Recent research has found that converting downstream tasks to language modeling tasks via textual prompts is more effective to use PLM than typical fine-tuning (Liu et al., 2021a). Early prompting method, GPT-3 (Brown et al., 2020) and PET/iPET (Schick and Schütze, 2020) for example, uses hand-crafted prompt templates. However, the performance of these methods relies heavily on the selection of predefined prompt templates. Hand-crafting prompts are very time-consuming, and the performance may be sub-optimal. Shin et al. (2020) propose AutoPrompt to search for better prompts based on gradient descent approach. Instead of searching for discrete template words, Li and Liang (2021) propose prefix-tuning, where tokens with trainable continuous embeddings are placed at the beginning of the text to perform generate tasks. P-tuning v2 (Liu et al., 2021b) also uses soft-prompt to achieve promising natural language understanding and knowledge probing tasks. Different from the above methods, they studied single-step adaptation, and we are interested in prompt transfer in CL environment.

2.3 Continual learning with fast generalization

Mitigating CF is usually a priority in CL or lifelong learning research proposals (Hadsell et al., 2020). Recently, the demands on CL have increased further, not only to combat CF but also to generalize quickly in unseen tasks. Integration with meta-learning is a promising approach, concerned with balancing stability (preservation of past knowledge) and plasticity (rapid absorption of current knowledge). MER (Riemer et al., 2018) achieves gradient alignment by constraining the direction of the gradient angle between different task samples. Subsequently, OML (Javed and White, 2019) and La-MAML (Gupta et al., 2020) optimize and supplemente the training speed and effect of MER, respectively. Meta-MbPA (Wang et al., 2020) combine the above meta-learning, rehearsal-based replay of CL and BERT. Wang et al. (2021a) uses global and local memory networks to capture different classes of cross-task prototype representations, adds a new frozen classification module for each task, and requires BERT to update slowly. Ke et al. (2021b) and Jin et al. (2021) freezes part of the backbone model against CF. They introduce an additional adapter layer (Houlsby et al., 2019) to learn task-specific knowledge, avoiding inefficient data replay and reducing parameter tuning and growth rates. Different from the above methods, CPT-RD is based on PT. Although it is similar to adapter in terms of parameter tuning, PT is more effective in parameters, and CPT-RD has a more intuitive means of knowledge transfer and avoids data replay.

3 Methodology

In this section, we first present the task definition and execution process of CPT-RD in CDET, then elaborate prompt encoding method and bidirectional knowledge transfer strategy.

3.1 Task definition and description

Suppose RD model $\mathcal{M}_{1:k-1}$ has performed learning on a sequence of tasks from 1 to $k-1$, denoted as $\mathcal{T}_{1:k-1} = \{\mathcal{T}_1, \ldots, \mathcal{T}_{k-1}\}$. Each task is a domain-specific rumor binary classification problem (non-rumor or rumor), and the inputs are claim and comments texts. The goal of the RD model is to use the knowledge gained from upstream $\{\mathcal{T}_{1:k-1}\}$ tasks to help learn a better detector $\mathcal{M}_{1:k}$ for the k-th task $\mathcal{T}_k$ while avoiding

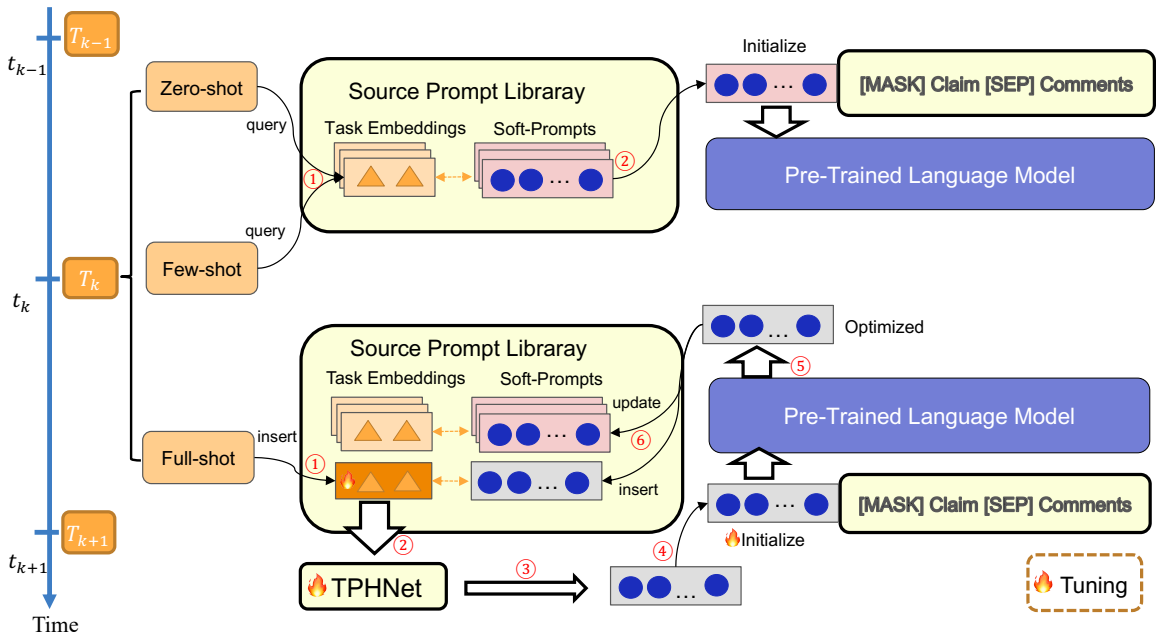

Figure 3: Illustration of the execution of CPT-RD in a single task in CDET.

forgetting knowledge learned from past tasks. We use the terms domain and task interchangeably because each task is from a different domain.

Often, each task consists of three stages: zero-shot stage, few-shot stage, and full-shot (large-scale labeled training) stage, which corresponds to the development process of rumors from germination to normalization as described in Sec.1. As shown in Fig.3, in the zero-shot and few-shot stages, CPT-RD applies various forward knowledge transfer strategies proposed in Sec.3.3 to query the soft-prompt as the prompt initialization of the current task. Note that the soft-prompt of the current task is randomly initialized if the source prompt library (SPL) is empty. In the full-shot phase of the current task, CPT-RD starts training with a randomly initialized soft-prompt instead of using the soft-prompt in SPL because, in experiments, we found that the former works better. CPT-RD obtains knowledge accumulation in full-shot annotated samples through TPHNet. The optimized soft-prompt specific to the current task is stored in the SPL at the end of full-shot training.

3.2 Prompt encoding

Prompt tuning (PT) formalizes RD as a masked language modeling problem using a pre-trained language model (PLM). In Fig.2, given the k-th task input, the pretrained embedding layer e of PLM converts the claim and comments text into token embeddings $\mathbf{X}_k = e(\text{claim}) \in \mathbb{R}^{m\times d}$ and $\mathbf{C}_k = e(\text{comments}) \in \mathbb{R}^{n\times d}$, m, n is the token length, and d is the hidden dimension of PLM. We prepend l randomly initialized soft-prompt tokens $\mathbf{P}_k = \{P_k^1, P_k^2, \dots P_k^l\} \in \mathbb{R}^{l\times d}$ before them, where $P_k^i \in \mathbb{R}^d$ is an embedding vector. Then, we add a [MASK] token, which is used to predict the label words $y \in \mathcal{Y}$. The input embedding of PLM is:

$$\mathbf{x} = \mathbf{P}_k, [\texttt{MASK}], \mathbf{X}_k, [\texttt{SEP}], \mathbf{C}_k, \qquad (1)$$

and only $\mathbf{P}_k$ is learnable. Hence the tuned parameters in PT are extremely fewer than full-parameter fine-tuning, which is friendly for model deployment.

One key ingredient of PT is the *verbalizer*: a mapping from the class label to a word token in the PLM vocabulary. PLM gives the probability of each word v in the vocabulary being filled in [MASK] token $p([\texttt{MASK}] = v|\mathbf{x})$. To map the probabilities of words into the probabilities of labels, we define the *verbalizer* as ver, which form the label word set $\mathcal{V}$, to the label space $\mathcal{Y}$, i.e., $ver: \mathcal{V} \mapsto \mathcal{Y}$. We use $\mathcal{V}_y$ to denote the subset of $\mathcal{V}$ that is mapped into a specific label y, $\cup_{y\in\mathcal{Y}}\mathcal{V}_y = \mathcal{V}$. Then the probability of label y, i.e., $p(y|\mathbf{x})$, is calculated as:

$$p(y|\mathbf{x}) = f_{ver}\left(p([\texttt{MASK}] = v|\mathbf{x}) | v \in \mathcal{V}_y\right), \qquad (2)$$

where f_{ver} is a function transforming the probability of label words into the probability of the label.

The training objective is:

$$L_{\mathbf{P}_k} = -\sum_{\mathbf{x}_i, y_i \in D_{\mathcal{T}_k}} \log(p(y_i|\mathbf{x}_i)), \quad (3)$$

where $D_{\mathcal{T}_k}$ is defined as the training data for task $\mathcal{T}_k$.

PT only inserts soft-prompt into the input embedding sequence of PLM. We follow the deep prompt tuning of P-tuning v2, adding the prompts of different layers as prefix tokens to the input sequence independently of other layers to increase tunable task-specific parameters and improve stability. In experiment Sec.4.3, CPT-RD evaluates the above two soft-prompt placement methods respectively.

3.3 Forward knowledge transfer

Due to the lack of samples or even no training samples (zero-shot and few-shot) in the early stage of rumor events, it is necessary to forward transfer the knowledge from upstream tasks for fast detection. An intuitive way of knowledge transfer is to reuse knowledge gained from previous tasks, which often improves and accelerates learning for future tasks.

Therefore, SPL is responsible for storing upstream soft-prompts $\{\mathbf{P}_j\}_{j<k}$ and task embeddings $\{\mathbf{z}_j\}_{j<k}$, where $\mathbf{P}_j$ is obtained after training on $D_{\mathcal{T}_k}$ in the full-shot stage, $\mathbf{z}_j = \frac{1}{|D_{\mathcal{T}_k}|}\sum_{i=1}^{|D_{\mathcal{T}_k}|} f_e(\mathbf{X}_k^i, \mathbf{C}_k^i)$, f_e is an encoder model, i.e. BERT. Based on SPL, three types of prompt initialization are proposed.

CLInit: Use the previous task's soft-prompt $\mathbf{P}_{k-1}$ to initialize the current task's soft prompt $\mathbf{P}_k$. **SimInit:** Select $\mathbf{P}_k$ from $\{\mathbf{P}_j\}_{j<k}$ with the highest similarity to the current task representation $\mathbf{z}_k \in \mathbb{R}^d$ for initialization. Note that $\mathbf{z}_k$ is calculated based on the training data of task $\mathcal{T}_k$, which does not need to be labeled. Straightforwardly, we compute euclidean distances E and cosine similarities C for task embedding pairs in the two groups and use the averaged results as the final similarity metrics:

$$\begin{aligned} \mathrm{E}(\mathbf{z}_k, \mathbf{z}_j) &= \frac{1}{1 + \|\mathbf{z}_k - \mathbf{z}_j\|}, \\ \mathrm{C}(\mathbf{z}_k, \mathbf{z}_j) &= \frac{\mathbf{z}_k \cdot \mathbf{z}_j}{\|\mathbf{z}_k\|\|\mathbf{z}_j\|}. \end{aligned} \quad (4)$$

When CPT-RD faces the upstream task domain samples again, it directly reuses the soft-prompt of the corresponding task, thus completely avoiding the occurrence of CF, but CLInit and SimInit are single-source transfer strategies.

Single-source reuse initialization strategy can completely avoid CF, but only a single task is considered in the forward knowledge transfer. Intuitively, knowledge available for transfer should be present in all upstream tasks. **MeanInit:** Calculate the average of $\{\mathbf{P}_j^u\}_{j<k}$ to obtain $\mathbf{P}_k$. In deep prompt tuning, each layer of soft-prompt is correspondingly averaged. MeanInit considers multi-source transfer, but none of the above strategies can accumulate to transfer backward knowledge. We empirically compare these three strategies in Sec.4.3.

3.4 TPHNet for backward knowledge transfer

Although storing the training-optimized $\mathbf{P}_k$ after labeled train on task $\mathcal{T}_k$ can avoid forgetting, it ignores the backward knowledge transfer of the tasks. Hypernetwork is usually used to consolidate the knowledge of sequence tasks and has a certain ability of backward knowledge transfer (von Oswald et al., 2019; KJ and N Balasubramanian, 2020; Jin et al., 2021). It is a network that explores the meta-parameter space of another network. Similar to Hypernetwork, instead of modeling the final classification result, task-conditioned prompt-wise hypernetwork (TPHNet) learns a latent distribution space of soft-prompt with task-specific priors, aiming to accumulate knowledge in sequential and use it for all future tasks.

Specifically, when the CPT-RD completed the learning of task $\mathcal{T}_{k-1}$, and before the full-shot phase of $\mathcal{T}_k$ starts, the current task embedding $\mathbf{z}_k$ and the soft-prompt set $\{\mathbf{P}_j\}_{j<k}$ already exist in SPL as described in Sec.3.3. Here, the task representation $\mathbf{z}_k$ for task $\mathcal{T}_k$ is optimized jointly while learning the task. Taking the PT case as an example, adding soft-prompt to the embedding layer, deep prompt tuning only needs to modify the generation dimension of soft-prompt. TPHNet g generates a soft-prompt $\mathbf{P}_k$ through an auto-encoder, using $\mathbf{z}_k$ as input:

$$g(\mathbf{z}_k) = W_2(\tanh(W_1\mathbf{z}_k + b_1)) + b_2, \quad (5)$$

where $W_1 \in \mathbb{R}^{d' \times d}, b_1 \in \mathbb{R}^{d'}, W_2 \in \mathbb{R}^{(l \times d) \times d'}$ and $b_2 \in \mathbb{R}^{l \times d'}$ are trainable parameters, $d' = 64$ is the middle dimension.

Then, in each step of learning $\mathcal{T}_k^i$, we randomly sample a prior task soft-prompt $\mathbf{P}_j (j < k)$ to regu-

larize the TPHNet learning. It penalizes the ℓ_2 distance between the soft-prompt generated at the current step $\mathbf{P}_k^i = g(\mathbf{z}_k^i)$ and the pre-computed one, *i.e.*, $||\mathbf{P}_k^i - \mathbf{P}_j||_2^2$. Therefore, we avoid the TPHNet changes its output for a prior task too much during the sequential task learning, so that the knowledge accumulation is better guaranteed for the learned model. The following overall loss function:

$$L_{\mathcal{T}_k} = \hat{L_{\mathbf{P}_k}} + \frac{\beta}{k-1}\sum_{i=1}^{k-1}||\mathbf{P}_k^i - \mathbf{P}_j||_2^2, \quad (6)$$

where $\hat{L_{\mathbf{P}_k}}$ is the cross entropy like Eq.3. According to this equation, optimizing the overall loss will update the patameters of the TPHNet g, task embedding $\mathbf{z}_k$ and prompt-based PLM. $\beta = 0.01$ is a hyperparameter that controls the strength of the regularizer. After training, if $\mathbf{P}_k$ obtains better (or the same) performance than $\mathbf{P}_j$ on $\mathcal{T}_j$, we update $\mathbf{P}_j$ to $\mathbf{P}_k$.

4 Experiments

To evaluate the proposed CPT-RD, we closely follow the settings proposed in prior works (Lopez-Paz and Ranzato, 2017; Jin et al., 2021), and conduct comprehensive experiments. In particular, we mainly consider whether CPT-RD effectively addresses the four challenges mentioned in Sec.1. We carefully compare CPT-RD with state-of-the-art methods of different categories under proper experiment settings. Moreover, we conduct extensive ablation studies to provide a deeper understanding of our method.

4.1 Datasets and evaluation metrics

We collected 14 domain events for each dataset, Chinese and English. Each piece of data contains a claim, comments, and label (non-rumor or rumor). We split the data for each event into train/validation/test datasets with a split ratio of 30%/35%/35%. The details about the statistics are shown in Table 6, and the PHEME and Weibo (Lu et al., 2021) datasets are well divided by event domain. We also added the COVID-19 dataset (Patwa et al., 2021a) for the English dataset, which comes from the competition[2]. In addition, for datasets without split of events, including Twitter15 and Twitter16 (Ma et al., 2018), we use Tweeter-LDA (Diao et al., 2012), an LDA variant widely used for short and noisy tweets, to determine topic clusters as well as important words with their weights. We removed the label of unverified rumors in Twitter15 and Twitter16, retained the rumors and non-rumor, and divided them into five-event domains. The similarity between the datasets we calculated using TF-IDF is shown in Fig.5. To prepare the model inputs for both datasets, first, we replace the URLs with the special token [unused10]". Then, we also replace the usernames with the special token [unused11]". These two datasets will be released on GitHub along with our experimental code.

To evaluate the performance of RD using F1 score as a metric, $R_{j,i}$ is defined as the F1 score on the test set of task $\mathcal{T}_i$ after training on task $\mathcal{T}_j$. We follow the two indicators, FWT and BWT, proposed by Lopez-Paz and Ranzato (2017) to evaluate the knowledge transfer ability of CPT-RD in the process of continual learning. We evaluate the average F1 performance for all tasks after full-shot training on the final task $\mathcal{T}_N$:

$$\text{Avg.F1} = \frac{1}{N}\sum_{i=1}^{N} R_{T,i}. \quad (7)$$

where N is the number of tasks. According to Lopez-Paz and Ranzato (2017), two metrics are defined to measure the effect of forward and backward transfers:

$$\text{BWT} = \frac{1}{N-1}\sum_{i=1}^{N-1} R_{N,i} - R_{i,i}, \quad (8)$$

$$\text{FWT} = \frac{1}{N-1}\sum_{i=2}^{N} R_{i-1,i} - R_{0,i}. \quad (9)$$

FWT is the average zero-shot performance on a new task and evaluating the model's generalization ability. BWT assesses the impact of learning on the subsequent task has on the previous task. A negative BWT indicates that the model has forgotten some previously acquired knowledge.

In addition, to evaluate the models performance early in the birth of the rumor, we also recorded the average few-shot performance of each new task during the continual cumulative training, which is a cumulative value:

$$\text{fs.F1} = \frac{1}{N}\sum_{i=1}^{N} R_{i,i}^{fs}, \quad (10)$$

where $R_{i,i}^{fs}$ denotes the F1 performance of training on the few-shot training set of the i-th task and testing on the test set of the i-th task.

[2]https://competitions.codalab.org/competitions/26655.

	PHEME + Twitter15&16 + Covid19				Weibo					
Method	Avg.F1	FWT	BWT	fs.F1(val)	Avg.F1	FWT	BWT	fs.F1(val)	+Params	Tune Params
Fine-tuning†	54.6 (± 1.2)	7.0	-24.3	67.5	45.6 (±3.1)	18.3	-23.2	75.6	0	100%
EANN†	57.9 (± 1.5)	8.1	-20.3	68.3	49.4 (±2.1)	18.5	-20.9	77.1	1.8%	100%
Adapter⋆	66.7(± 2.5)	13.7	-14.2	74.6	58.0 (±2.3)	24.4	-17.5	83.9	2.1%	2.1%
ParallelAdapter⋆	61.5 (± 2.2)	2.8	**-4.7**	68.7	55.8 (±2.5)	8.5	-7.1	75.3	2.3%	2.3%
prompt-tuning (CLS)⋆	60.0 (± 1.5)	7.4	-15.7	75.9	67.8 (±1.8)	18.3	-5.1	**84.1**	0.03%	0.03%
p-tuning v2 (CLS)⋆	62.9 (± 2.1)	5.1	-9.3	73.0	67.6 (±1.6)	24.4	**-3.0**	80.4	0.6%	0.6%
prompt-tuning (VER)⋆	65.0 (± 1.5)	7.1	-10.2	75.7	70.5 (±1.8)	18.7	-7.5	82.8	0.03%	0.03%
p-tuning v2 (VER)⋆	**69.7 (± 1.1)**	**13.9**	-11.0	**76.3**	**71.8 (±1.6)**	**29.5**	-6.8	82.6	0.6%	0.6%
Prompt-tuning based (PT-based)										
CPT-RD (CLInit)⋆	65.1 (± 0.8)	15.0	0	75.5	70.0 (±0.7)	18.3	0	79.0	0.03%	0.03%
CPT-RD (SimInit)⋆	66.3(± 0.9)	**15.8**	0	**76.7**	71.6(±1.6)	19.8	0	83.6	0.03%	0.03%
CPT-RD (MeanInit)⋆	64.5(± 1.2)	14.8	0	75.6	70.3(±1.5)	18.5	0	80.9	0.03%	0.03%
CPT-RD (SimInit+TPHNet)⋆	**66.7(± 1.3)**	15.3	**0.2**	76.0	**71.9(±1.2)**	**20.0**	**0.5**	**84.0**	0.1%	0.1%
P-tuning v2 based (PTv2-based)										
CPT-RD (CLInit)⋆	68.2 (± 1.0)	19.8	0	76.1	73.1(±0.7)	30.7	0	83.2	0.6%	0.6%
CPT-RD (SimInit)⋆	72.2(± 1.1)	23.3	0	79.1	75.0(±1.2)	31.2	0	84.9	0.6%	0.6%
CPT-RD (MeanInit)⋆	70.6(± 1.5)	20.5	0	75.5	73.5(±1.3)	28.7	0	83.0	0.6%	0.6%
CPT-RD (SimInit+TPHNet)⋆	**75.0(± 1.3)**	**23.5**	**0.9**	**79.3**	**76.5(±1.6)**	**31.4**	**1.1**	**85.7**	1%	1%

Table 1: Results evaluated on test datasets for all tasks in `PHEME + Twitter15&16 + Covid19` and `Weibo`. The following averaged over 5 random task orders (Table 7) are reported, where (⋆) and (†) indicate frozen language model parameters and fine-tuning, (+Params) and (Tune Params) are additional parameters and the tunable parameters for each task. CLS and VER denote the output predictions with CLS token classifier and Verbalizer head, respectively. The fs.F1 is the result on the validation dataset, and the k-shot is 16.

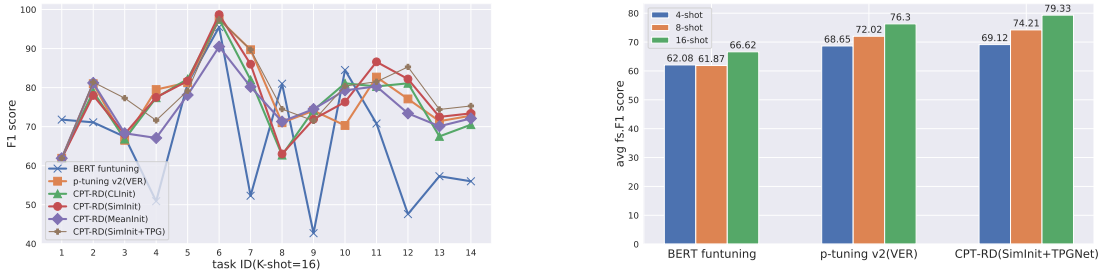

(a) k=16 during sequential tasks

(b) The average fs.F1 performance for k=4,8,16

Figure 4: The few-shot performance of each model on the dataset `PHEME + Twitter15&16 + Covid19`.

4.2 Compare models

In the experiments, BERT-base is used as the PLM weights, and CPT-RD will be compared with the following advanced models: **Fine-tuning:** Fine-tune the model on new task data continually. **EANN:** Wang et al. (2018) uses adversarial networks to extract event invariant features for generalization when new events arrive. For comparison, EANN is modified to use BERT as the encoder and only text data. **Adapter:** Freeze the pre-trained model and train a residual Adapter(Houlsby et al., 2019). **ParallelAdapter:** A variant by transferring the parallel insertion of prefix tuning into adapters (He et al., 2021). **prompt-tuning (CLS/VER):** which only tunes soft-prompts with a frozen language model (Lester et al., 2021), prompt for transformer's first layer. **p-tuning v2 (CLS/VER):** Using multilayer soft-prompts (deep prompt tuning), where CLS/VER denotes the predicted output with [CLS] token and Linear classification layer and verbalizer MLMhead, respectively. **MTL p-tuning v2 (CLS/VER):** P-tuning v2 in a multi-task manner instead of CL. Train a single prompt using all tasks data concurrently.

4.3 Main results

The higher the FWT score, the better the model works for the unseen domain. A negative BWT indicates that the model produces forgetting, and if it is positive, it can accumulate knowledge.

Compare model performance. In Table 1, unsurprisingly, the Fine-tuning model has a severely CF, the BWT value of -24.3 under the CL setting. The level of catastrophic forgetting in EANN (Wang et al., 2018) is somewhat reduced compared to Fine-tuning but is still severe. Prompt-

Task ID	Task	Fine-tuning	p-tuning v2(VER)	CPT-RD(CLInit)	CPT-RD(SimInit)	CPT-RD(MeanInit)	CPT-RD(SimInit+TPHNet)
1	Charlie Hebdo	22.1	64.1	75.3	77.1	75.3	78.0
2	TwitterEvent4	65.1	88.0	72.1	65.2	70.6	74.4
3	Ferguson	24.7	62.4	70.3	72.7	67.5	72.8
4	Germanwings-crash	50.6	63.4	65.9	60.6	64.7	61.3
5	Ottawa Shooting	52.7	74.3	70.5	72.2	74.6	78.3
6	Prince Toronto	98.7	71.4	98.1	95.6	93.7	94.3
7	Putin missing	53.0	55.3	53.0	66.0	53.5	72.1
8	TwitterEvent1	77.1	77.1	77.1	77.8	70.7	74.5
9	Sydney Siege	42.8	72.5	74.2	65.7	75.8	74.9
10	TwitterEvent5	71.4	67.5	69.8	70.5	71.7	73.5
11	Gurlitt	43.7	48.3	42.2	62.3	52.5	62.5
12	Covid19	47.6	86.7	87.9	84.3	85.3	88.3
13	TwitterEvent2	57.8	69.0	60.5	70.8	65.2	72.4
14	TwitterEvent3	56.9	74.9	37.9	70.9	67.3	71.3
	Avg.	54.6	69.7	68.2	72.2	70.6	75.0

Table 2: The F1 score evaluated on the final model after all 14 tasks are visited in the test set. We use Avg. to represent the average F1 of all tasks for each method. The gray numbers indicates that CPT-RD (p-tuning v2 based) does not perform as well as normal p-tuning v2 in a single task.

tuning and p-tuning v2 are somewhat related to the adapter method in the form of parameter tuning (He et al., 2021), but their performance in CL differs. The prompt-based model is better than the adapter in both datasets. From Tabel 4 in Sec.4.5, consistent with Liu et al. (2021b), CLS token classification is better than the verbalizer when p-tuning v2 is under multi-task learning mode. The performance of multi-task learning is usually considered the upper limit of the CL model. However, verbalizer's FWT and final F1 scores in the CL setting were better than CLS but more prone to CF. Since SPL lets the model not generate CF, we next develop CPT-RD using prompt-tuning (PT) and p-tuningv2 (PTv2) baseline with verbalizer.

CPT-RD performance. We have developed CPT-RD based on PT and PTv2 respectively. We can find that since CPT-RD has SPL, the BWT values are all greater than or equal to 0, which indicates that this fundamentally eliminates the CF. The more stable performance among the three forward transfer strategies is SimInit. CLInit is the most unstable, probably because the soft-prompt from the previous task is not necessarily beneficial to the learning of the next task and is prone to cumulative negative effects on subsequent tasks in CL. It can be observed by Tabel 2 that the F1 score of CLInit on the last two tasks of the last sequence task is significantly lower than other strategies, 60.5 and 37.9 respectively. MeanInit with multi-source prompt does not work as expected and is only slightly better than CLInit. Finally, we added TPHNet to SimInit to achieve backward knowledge transfer, but the performance on PT is not stable enough, probably because PT has fewer tunable parameters. Furthermore, TPHNet incorporates additional tunable parameters but still has parameter-efficient that is much lower than Fine-tuning.

Final model performance. The final model performance is the model's performance after learning the last task in the sequence. In Table 2, although CPT-RD solves CF, in the ideal case, the final model performance should outperform PTv2. We can find that the final performance of CLinit is inferior to PTv2 in 7 tasks, simInit and meanInit in 6, and SimInit+TPG in 5, which further indicates that TPHNet achieves a certain degree of backward knowledge transfer. Among them, we can find that in CLinit, the last task is as low as 37.9, indicating that directly reusing the soft-prompt of the previous task may have a more significant impact on the subsequent tasks.

Data privacy and Parameter explosion. From Sec.3.4, we propose TPHNet in the context of hypernetwork, which avoids rehearsal-based data replay and thus preserves data privacy. For a more comprehensive evaluation of our method, we replace TPHNet with a rehearsal-based technique, more details in Sec.4.4. From Table 3, it can be found that TPHNet can achieve comparable performance to the rehearsal-based method.

Besides, it can be seen from Sec.3.3 that CPT-RD avoids parameter explosion by using SPL to store soft-prompt (without dynamic model expansion). And the storage space occupied by SPL is negligible. In Table 2, the tuning parameter of the Adapter-based model is 2.1-2.3%, which is more parameter-efficient than the 100% of the Fine-tuning model. However, CPT-RD adds only 0.03-1% tunable parameters based on more parameter-efficient PT and can achieve better performance.

Few-shot performance. There is a lack of samples in the early burst stages, so few-shot perfor-

mance is critical for RD models. The final model is the model obtained after learning the last task. From Fig.4 (a), it can be found that Fine-tuning has a very unstable few-shot performance in CL. PTv2 still shows better few-shot capability than Fine-tuning in CL, which indicates that the PT-based model is more suitable for domain generalization. In Fig.4 (b), the average fs.F1 of CPT-RD (SimInit+TPHNet) is higher than that of PTv2, indicating that continuously accumulated knowledge can be better used for few-shot.

4.4 CPT-RD with rehearsal buffer

To verify the gap between TPHNet and rehearsal-based technology, we randomly sample 50 pieces of data from each domain training set in memory, and jointly trained on the future domain dataset. It can be observed from Table 3 that the rehearsal-based technology reflects the certain ability to backward knowledge transfer by accumulating data in past domain tasks. From Table 1, the BWT value of TPHNet is 0.9, which shows that our method is close to rehearsal-based technology, but our advantage is that the memory occupies small, and there is no data privacy problem.

Buffer size	Methods (PTv2-based)	Avg. F1	FWT	BWT
0/domain	p-tuning v2 (VER)	69.7	13.9	-11.0
	CPT-RD (CLInit)	68.2	19.8	0
	CPT-RD (MeanInit)	70.6	20.5	0
	CPT-RD (SimInit)	72.2	23.3	0
50/domain	p-tuning v2 (VER)	71.2	14.1	1.5
	CPT-RD (CLInit)	72.1	20.1	1.1
	CPT-RD (MeanInit)	71.7	19.4	0.8
	CPT-RD (SimInit)	75.3	23.8	1.2

Table 3: The performance of CPT-RD (PTv2-based) on the `PHEME + Twitter15&16 + Covid19` (English) dataset using rehearsal-based technology.

4.5 Multi-task learning performance

The performance of multi-task learning methods is often defined as an upper bound on the performance of continual learning. Multi-task learning avoids CF by visiting data from different domain tasks at different times. In Table 4, we experimented with the model's performance in the multi-task learning mode under the Chinese and English datasets. Consistent with the experimental conclusion of Liu et al. (2021b), p-tuning v2 using CLS token classification outperforms the verbalizer under multi-task learning. However, the verbalizer outperforms CLS in FWT and final F1 score in the CL setting, but is more prone to CF.

Methods	Avg. F1	+Params	Tune Params
PHEME + Twitter15&16 + Covid19			
MTL p-tuning v2 (CLS)	80.3 (± 0.8)	0.6%	0.6%
MTL p-tuning v2 (VER)	77.9 (± 0.8)	0.6%	0.6%
Weibo			
MTL p-tuning v2 (CLS)	85.3 (± 0.8)	0.6%	0.6%
MTL p-tuning v2 (VER)	83.8 (± 0.8)	0.6%	0.6%

Table 4: The performance of the baseline model p-tuning v2 in the multi-task learning mode.

model	Avg.F1	FWT	BWT
CPT-RD	75.0	23.5	0.9
w/o TPHNet	72.2	23.3	0
w/o SimInit	71.3	14.2	0.9
w/o SimInit+TPHNet	69.7	13.9	-11.0

Table 5: Ablation study on the effectiveness of the CPT-RD (SimInit+TPHNet) PTv2-based on the test set in the `PHEME + Twitter15&16 + Covid19`.

4.6 Ablation study

To understand the effectiveness of the different techniques proposed, we conducted an ablation study. From Table 1, we know that CPT-RD is more stable in PTv2-based than PT-based, which indicates that the model can benefit from more tunable parameters. Conventional PT do not use forward transfer strategy and TPHNet with lower FWT values. In PTv2-based, the overall performance is highest when SimInit and TPHNet are used together. In Table 5, removing either of TPHNet or SimInit will result in a decrease in overall performance. This shows the validity of our improvements, which would degrade to p-tuning v2 if SimInit and TPHNet were both removed.

5 Conclusion

We explore how to continually detect a social network environment with frequent unseen domains in RD. The novel framework Continual Prompt-tuning RD (CPT-RD) is proposed, which includes various knowledge transfer techniques. In the face of emergency domains, CPT-RD can use soft-prompt initialization strategies to achieve fast generalization. There is also a task-conditioned prompt-wise generator network (TPHNet) in terms of continual accumulation of detection knowledge. Future works include extending usage of CPT-RD to task agnostic scenarios and designing more diverse knowledge transfer strategies.

References

Eyal Ben-David, Nadav Oved, and Roi Reichart. 2021. Pada: A prompt-based autoregressive approach for adaptation to unseen domains. *arXiv preprint arXiv:2102.12206*.

Tian Bian, Xi Xiao, Tingyang Xu, Peilin Zhao, Wenbing Huang, Yu Rong, and Junzhou Huang. 2020. Rumor detection on social media with bi-directional graph convolutional networks. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pages 549–556.

Tom B Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. 2020. Language models are few-shot learners. *arXiv preprint arXiv:2005.14165*.

Zhiyuan Chen and Bing Liu. 2018. Lifelong machine learning. *Synthesis Lectures on Artificial Intelligence and Machine Learning*, 12(3):1–207.

Qiming Diao, Jing Jiang, Feida Zhu, and Ee Peng LIM. 2012. Finding bursty topics from microblogs. Acl.

Bin Guo, Yasan Ding, Lina Yao, Yunji Liang, and Zhiwen Yu. 2020. The future of false information detection on social media: New perspectives and trends. *ACM Computing Surveys (CSUR)*, 53(4):1–36.

Gunshi Gupta, Karmesh Yadav, and Liam Paull. 2020. La-maml: Look-ahead meta learning for continual learning. *arXiv preprint arXiv:2007.13904*.

Raia Hadsell, Dushyant Rao, Andrei A Rusu, and Razvan Pascanu. 2020. Embracing change: Continual learning in deep neural networks. *Trends in cognitive sciences*.

Junxian He, Chunting Zhou, Xuezhe Ma, Taylor Berg-Kirkpatrick, and Graham Neubig. 2021. Towards a unified view of parameter-efficient transfer learning. *arXiv preprint arXiv:2110.04366*.

Neil Houlsby, Andrei Giurgiu, Stanislaw Jastrzebski, Bruna Morrone, Quentin De Laroussilhe, Andrea Gesmundo, Mona Attariyan, and Sylvain Gelly. 2019. Parameter-efficient transfer learning for nlp. In *International Conference on Machine Learning*, pages 2790–2799. PMLR.

Khurram Javed and Martha White. 2019. Meta-learning representations for continual learning. *arXiv preprint arXiv:1905.12588*.

Xisen Jin, Bill Yuchen Lin, Mohammad Rostami, and Xiang Ren. 2021. Learn continually, generalize rapidly: Lifelong knowledge accumulation for few-shot learning. *arXiv preprint arXiv:2104.08808*.

Zixuan Ke, Bing Liu, Nianzu Ma, Hu Xu, and Lei Shu. 2021a. Achieving forgetting prevention and knowledge transfer in continual learning. *Advances in Neural Information Processing Systems*, 34.

Zixuan Ke, Hu Xu, and Bing Liu. 2021b. Adapting bert for continual learning of a sequence of aspect sentiment classification tasks. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 4746–4755.

Joseph KJ and Vineeth N Balasubramanian. 2020. Meta-consolidation for continual learning. *Advances in Neural Information Processing Systems*, 33:14374–14386.

Nayeon Lee, Andrea Madotto, Yejin Bang, and Pascale Fung. 2021. Dynamically addressing unseen rumor via continual learning. *arXiv preprint arXiv:2104.08775*.

Brian Lester, Rami Al-Rfou, and Noah Constant. 2021. The power of scale for parameter-efficient prompt tuning. *arXiv preprint arXiv:2104.08691*.

Xiang Lisa Li and Percy Liang. 2021. Prefix-tuning: Optimizing continuous prompts for generation. *arXiv preprint arXiv:2101.00190*.

Hongzhan Lin, Jing Ma, Mingfei Cheng, Zhiwei Yang, Liangliang Chen, and Guang Chen. 2021. Rumor detection on twitter with claim-guided hierarchical graph attention networks. *arXiv preprint arXiv:2110.04522*.

Pengfei Liu, Weizhe Yuan, Jinlan Fu, Zhengbao Jiang, Hiroaki Hayashi, and Graham Neubig. 2021a. Pre-train, prompt, and predict: A systematic survey of prompting methods in natural language processing. *arXiv preprint arXiv:2107.13586*.

Xiao Liu, Kaixuan Ji, Yicheng Fu, Zhengxiao Du, Zhilin Yang, and Jie Tang. 2021b. P-tuning v2: Prompt tuning can be comparable to fine-tuning universally across scales and tasks. *arXiv preprint arXiv:2110.07602*.

David Lopez-Paz and Marc'Aurelio Ranzato. 2017. Gradient episodic memory for continual learning. *Advances in neural information processing systems*, 30:6467–6476.

Heng-yang Lu, Chenyou Fan, Xiaoning Song, and Wei Fang. 2021. A novel few-shot learning based multi-modality fusion model for covid-19 rumor detection from online social media. *PeerJ Computer Science*, 7:e688.

Jing Ma, Wei Gao, and Kam-Fai Wong. 2018. Detect rumor and stance jointly by neural multi-task learning. In *Companion proceedings of the the web conference 2018*, pages 585–593.

Ray Oshikawa, Jing Qian, and William Yang Wang. 2018. A survey on natural language processing for fake news detection. *arXiv preprint arXiv:1811.00770*.

Parth Patwa, Mohit Bhardwaj, Vineeth Guptha, Gitanjali Kumari, Shivam Sharma, Srinivas Pykl, Amitava Das, Asif Ekbal, Md Shad Akhtar, and Tanmoy Chakraborty. 2021a. Overview of constraint 2021 shared tasks: Detecting english covid-19 fake news and hindi hostile posts. In *International Workshop on Combating On line Ho st ile Posts in Regional Languages dur ing Emerge ncy Si tuation*, pages 42–53. Springer.

Parth Patwa, Shivam Sharma, Srinivas Pykl, Vineeth Guptha, Gitanjali Kumari, Md Shad Akhtar, Asif Ekbal, Amitava Das, and Tanmoy Chakraborty. 2021b. Fighting an infodemic: Covid-19 fake news dataset. In *International Workshop on Combating On line Ho st ile Posts in Regional Languages dur ing Emerge ncy Si tuation*, pages 21–29. Springer.

Matthew Riemer, Ignacio Cases, Robert Ajemian, Miao Liu, Irina Rish, Yuhai Tu, and Gerald Tesauro. 2018. Learning to learn without forgetting by maximizing transfer and minimizing interference. *arXiv preprint arXiv:1810.11910*.

Timo Schick and Hinrich Schütze. 2020. It's not just size that matters: Small language models are also few-shot learners. *arXiv preprint arXiv:2009.07118*.

Taylor Shin, Yasaman Razeghi, Robert L Logan IV, Eric Wallace, and Sameer Singh. 2020. Autoprompt: Eliciting knowledge from language models with automatically generated prompts. *arXiv preprint arXiv:2010.15980*.

Johannes von Oswald, Christian Henning, João Sacramento, and Benjamin F Grewe. 2019. Continual learning with hypernetworks. *arXiv preprint arXiv:1906.00695*.

Chengyu Wang, Haojie Pan, Yuan Liu, Kehan Chen, Minghui Qiu, Wei Zhou, Jun Huang, Haiqing Chen, Wei Lin, and Deng Cai. 2021a. Mell: Large-scale extensible user intent classification for dialogue systems with meta lifelong learning. In *Proceedings of the 27th ACM SIGKDD Conference on Knowledge Discovery & Data Mining*, pages 3649–3659.

Yaqing Wang, Fenglong Ma, Zhiwei Jin, Ye Yuan, Guangxu Xun, Kishlay Jha, Lu Su, and Jing Gao. 2018. Eann: Event adversarial neural networks for multi-modal fake news detection. In *Proceedings of the 24th acm sigkdd international conference on knowledge discovery & data mining*, pages 849–857.

Yaqing Wang, Fenglong Ma, Haoyu Wang, Kishlay Jha, and Jing Gao. 2021b. Multimodal emergent fake news detection via meta neural process networks. In *Proceedings of the 27th ACM SIGKDD Conference on Knowledge Discovery & Data Mining*, pages 3708–3716.

Zirui Wang, Sanket Vaibhav Mehta, Barnabás Póczos, and Jaime Carbonell. 2020. Efficient meta lifelong-learning with limited memory. *arXiv preprint arXiv:2010.02500*.

Huaiwen Zhang, Shengsheng Qian, Quan Fang, and Changsheng Xu. 2021. Multi-modal meta multi-task learning for social media rumor detection. *IEEE Transactions on Multimedia*.

Xinyi Zhou, Atishay Jain, Vir V Phoha, and Reza Zafarani. 2020. Fake news early detection: A theory-driven model. *Digital Threats: Research and Practice*, 1(2):1–25.

Arkaitz Zubiaga, Ahmet Aker, Kalina Bontcheva, Maria Liakata, and Rob Procter. 2018. Detection and resolution of rumours in social media: A survey. *ACM Computing Surveys (CSUR)*, 51(2):1–36.

A Appendix

A.1 Event correlation

After filtering the stopping words, we use TF-IDF to calculate the event relevance of each task domain, as shown in Fig.5.

Datasets	tasks	non-rumor	rumor	total
PHEME	Gurlitt	77	61	138
	Putin missing	112	126	238
	Prince Toronto	4	229	233
	Germanwings-crash	231	238	469
	Ferguson	859	284	1143
	Charlie Hebdo	1621	458	2079
	Ottawa Shooting	420	470	890
	Sydney Siege	699	522	1221
Twitter15&16	TwitterEvent 1	63	156	219
	TwitterEvent 2	89	164	253
	TwitterEvent 3	153	198	351
	TwitterEvent 4	146	200	346
	TwitterEvent 5	128	436	564
Covid19	Covid19	3060	3360	6420
	total	7662	6902	14564
Weibo	MH370	133	262	395
	Olympics	173	81	254
	Urban managers	94	149	243
	Cola	215	419	634
	Child trafficking	94	172	266
	Waste oil	133	57	190
	Accident	100	82	182
	Earthquake	117	58	175
	Typhoon	107	64	171
	Rabies	101	42	143
	College entrance exams	147	590	737
	Lockdown the city	86	24	110
	Zhong Nanshan	55	21	76
	Wuhan	167	69	236
	total	1722	2048	3770

Table 6: Statistics of `PHEME + Twitter15&16 + Covid19` (English) and `Weibo` (Chinese) datasets.

A.2 Implementation details

We tune hyperparameters on the `PHEME + Twitter15&16 + Covid19` and `Weibo` validation sets. We tune learning rates by enumerating over [3e-3, 5e-3, 7e-3], and finally use a learning rate of 7e-3 for all CPT-RD approaches where a learning rate of 1e-4 for TPHNet. The learning rate of the fine-tuning approaches is 5e-5, and the learning rate of the adapter approaches is 1e-4. Regarding the PT and PTv2, including the MTL approach, we use a learning rate of 5e-3. We use a batch size of 16 across experiments. We train the model for at most 100 epochs for each training task with a patience of 4 epochs without validation performance improvement. Before training on a new task, we revert the model to the checkpoint with the best validation performance in the previous task. In the few-shot learning stage, we use the same learning rate and train the model for 500 steps ($k \in \{16, 8, 4\}$), assuming no validation sets to perform early stopping. The length of the input sequence on data set `PHEME + Twitter15&16 + Covid19` is 300, and `Weibo` is 128. We also experimented with the length of different soft-prompt tokens, such as $\{20, 40, 60, 80\}$, and found that the performance of 40/60 is relatively stable, so 40 is uniformly used as the length of soft-prompt in the experiment.

Task Order	Tasks
	PHEME + Twitter15&16 + Covid19
Order1	Charlie Hebdo, TwitterEvent4, Ferguson, Germanwings-crash, Ottawa Shooting, Prince Toronto, Putin missing, TwitterEvent1, Sydney Siege, TwitterEvent5, Gurlitt, Covid19, TwitterEvent2, TwitterEvent3
Order2	Sydney Siege, ferguson,TwitterEvent1, Gurlitt, Ottawa Shooting, TwitterEvent3, Prince Toronto, TwitterEvent4, Putin missing, Charlie Hebdo, TwitterEvent5, Germanwings-crash, TwitterEvent2, Covid19
Order3	TwitterEvent3, Charlie Hebdo, TwitterEvent2, Ferguson, TwitterEvent1, TwitterEvent5, TwitterEvent4, Putin missing, Ottawa Shooting, Prince Toronto,Gurlitt, Germanwings-crash, sydneysiege, Covid19
Order4	TwitterEvent1, Germanwings-crash, TwitterEvent4, Ferguson, Gurlitt, Sydney Siege, TwitterEvent3, Charlie Hebdo, Ottawa Shooting, Prince Toronto,TwitterEvent2, TwitterEvent5, Putin missing, Covid19
Order5	Covid19, Germanwings-crash, Prince Toronto, TwitterEvent3, TwitterEvent5, Sydney Siege, Ferguson, Ottawa Shooting, Charlie Hebdo,TwitterEvent4, TwitterEvent2, Gurlitt, Putin missing, TwitterEvent1
	Weibo
Order1	Typhoon, Olympic, MH370, Earthquake, Rabies, College entrance exams, Cola, Urban managers, Child trafficking, Accident, Waste oil, Zhong Nanshan, Wuhan, Lockdown the city
Order2	Olympic, Child trafficking, Rabies, Accident, Earthquake, Cola, College entrance exams,Typhoon, Urban managers, Waste oil, MH370, Zhong Nanshan, Wuhan, Lockdown the city
Order3	Zhong Nanshan, Wuhan, Lockdown the city, Cola, Accident, Urban managers, Waste oil, Earthquake, Olympic, Child trafficking,Typhoon, Rabies, College entrance exams, MH370
Order4	Cola, Accident, Olympic, Waste oil, Typhoon, Zhong Nanshan, Wuhan, Lockdown the city, College entrance exams, Urban managers, Rabies, Child trafficking, MH370, Earthquake
Order5	Zhong Nanshan, Wuhan, Lockdown the city, MH370, Urban managers, Child trafficking, Typhoon, Earthquake, Olympic, Cola, Accident, Rabies, Waste oil, College entrance exams

Table 7: Order of continual learning tasks in `PHEME + Twitter15&16 + Covid19` and `Weibo` datasets.

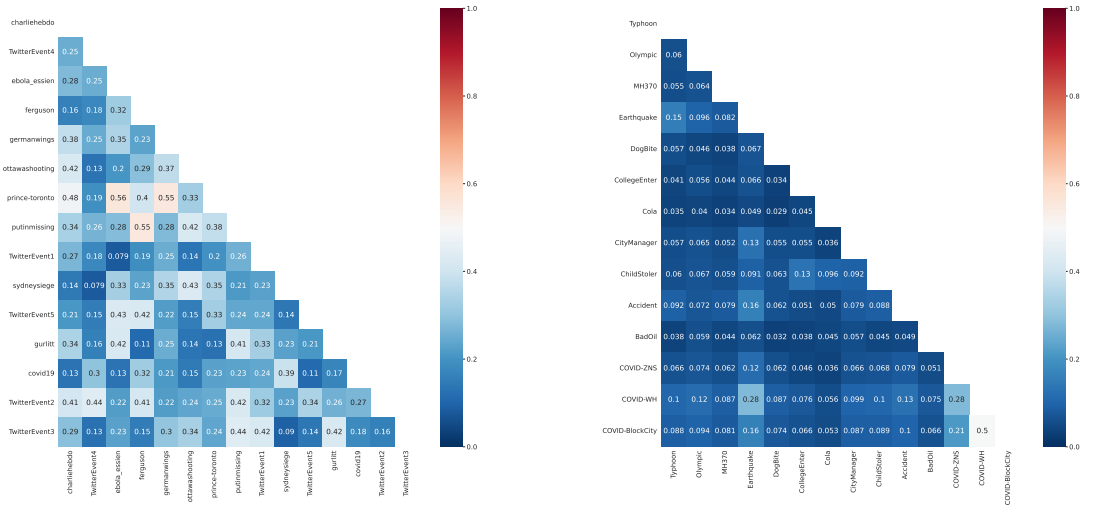

(a) `PHEME + Twitter15&16 + Covid19` (b) `Weibo`

Figure 5: Domain Task Relevance Heat Map.

AiM: Taking Answers in Mind to Correct Chinese Cloze Tests in Educational Applications

Yusen Zhang[1]*, **Zhongli Li**[2], **Qingyu Zhou**[2], **Ziyi Liu**[1]*,
Chao Li[3], **Mina Ma**[2], **Yunbo Cao**[2], **Hongzhi Liu**[1]
[1]Peking University, [2]Tencent Cloud Xiaowei, [3]Xiaomi Group
{yusen-zhang0826@stu,lzymail@stu,liuhz@ss}.pku.edu.cn
{neutrali,qingyuzhou,minarma,yunbocao}@tencent.com
lichao51@xiaomi.com

Abstract

To automatically correct handwritten assignments, the traditional approach is to use an OCR model to recognize characters and compare them to answers. The OCR model easily gets confused on recognizing handwritten Chinese characters, and the textual information of the answers is missing during the model inference. However, teachers always have these answers in mind to review and correct assignments. In this paper, we focus on the Chinese cloze tests correction and propose a multimodal approach[1] (named AiM). The encoded representations of answers interact with the visual information of students' handwriting. Instead of predicting 'right' or 'wrong', we perform the sequence labeling on the answer text to infer which answer character differs from the handwritten content in a fine-grained way. We take samples of OCR datasets as the positive samples for this task, and develop a negative sample augmentation method to scale up the training data. Experimental results show that AiM outperforms OCR-based methods by a large margin. Extensive studies demonstrate the effectiveness of our multimodal approach.

1 Introduction

The growing number of students has brought much pressure on teachers to correct assignments manually in the educational field. Recently, Optical Characters Recognition (OCR) based methods are widely used in several applications[2] to automatically complete this task. The OCR model first recognizes the text in the image, and then the OCR output is compared with the correct answer to feedback correction results. In this pipeline method, the post-comparison mainly relies on the pre-recognition. Without prior knowledge of answers, the OCR model easily gets confused when recognizing handwritten Chinese characters, especially for various handwriting styles, ligatures, and shape-similar characters. However, most of them can be distinguished by human teachers, because they take answers in mind at first.

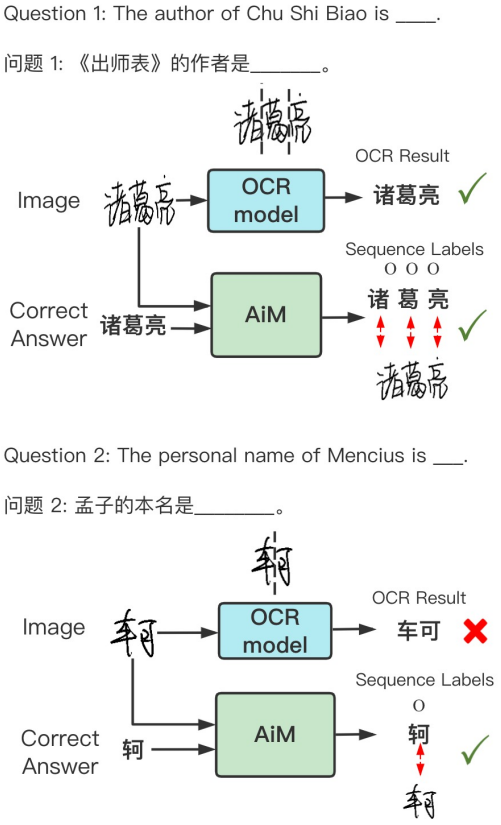

Figure 1: Task examples and correction methods. The OCR-based method misunderstands student's intent due to the width between "车" and "可" in Question 2. AiM can tackle this problem since it takes both the handwriting image and the textual answer into consideration.

In this paper, we focus on the Chinese cloze correction (CCC) and propose an **A**nswer-**i**n-**M**ind correction model (AiM) to tackle the above prob-

*Contribution done during internship at Tencent Cloud Xiaowei. Qingyu Zhou is the corresponding author.

[1]Our codes and data are available at https://github.com/YusenZhang826/AiM.

[2]Educational applications such as http://kousuan.yuanfudao.com, https://jiazhang.zuoyebang.com and https://www.zuoye.ai.

Proceedings of the 29th International Conference on Computational Linguistics, pages 3042–3053
October 12–17, 2022.

lem. Figure 1 shows examples of the CCC task and the comparison of the OCR-based method with our method. We look at Question 2. Obviously, the student knows the correct answer and writes down "轲", but it is recognized as "车可" by the OCR model because of the slightly large width of the hand-written content. Consequently, the correction result is 'wrong'. We hypothesize that the information of correct answers can help the neural model to understand student handwriting. As shown in Figure 1, AiM is a multimodal model, which takes the image and the answer text as input. Through the interaction of two modality information, our model understands the handwritten content and feeds back the 'right' correct result.

In the AiM model, the image is encoded to sequential feature representations through Resnet (He et al., 2016), where each of them represents a fixed-width pixels block. The answer text is encoded by word embedding. Then Transformer (Vaswani et al., 2017) self-attention is adopted to compute contextual representations for each modality. In order to fuse them, we develop a cross-modal attention. It renders the textual representations to interact with the visual information of students' handwriting. On the top of AiM, instead of predicting 'right' or 'wrong', our model performs sequence labeling on the answer text to infer which character differs from the handwritten content.

To train AiM, we collect EinkCC, a dataset containing about 5k handwriting images, answers, and correction results of cloze questions, from our educational application. Teachers distribute cloze tests in our app, and students practice on the electronic paper hardware. In addition to EinkCC, OCR datasets can be used for this task. We take samples of OCR datasets (Liu et al., 2011) as the positive samples, and construct negative samples by replacing the label characters with shape-similar ones derived from an open-sourced confusion set (Wu et al., 2013). The same method also augments EinkCC to scale up the training set.

We pretrain the image encoder in AiM to get better visual representations, and further train AiM with the correction objective. Experimental results show that compared with OCR-based methods (Liu et al., 2020; Du et al., 2021), AiM achieves 11% accuracy improvements. Extensive analyses verify that with the interactions between two modalities through our attention mechanism, AiM can understand student handwriting and ligatures, and more handwritten characters confused by OCR can be predicted well by AiM.

The main contributions are summarized as follows: i) We propose AiM, a multimodal model for Chinese cloze correction, to make up for shortages of OCR-based methods. ii) We extend OCR datasets using a negative sample augmentation method to fit this task. iii) Comprehensive experiments show that AiM achieves better performance compared with OCR-based methods, and it's effective and necessary to use a multimodal approach to correct Chinese cloze tests.

2 Preliminary

2.1 Chinese Cloze Correction

We first give the description of the Chinese cloze correction (CCC) task in this section. Given a handwriting image $\mathbf{I}$ and the textual answer $\mathbf{A} = [a_1, a_2, ..., a_m]$ of the corresponding question, assuming that the handwritten characters in the image is $\mathbf{C} = [c_1, c_2, ..., c_n]$, the target of the task is to predict a label $y \in \{0, 1\}$. The $y = 0$ indicates the correction result is 'right' (i.e. the handwritten content is a correct answer, $\mathbf{C} = \mathbf{A}$), otherwise it is 'wrong'.

2.2 Transformer

Suppose the input of Transformer (Vaswani et al., 2017) is a pack of embeddings $\mathbf{X}^0 = [\mathbf{x}_1, \mathbf{x}_2, ..., \mathbf{x}_{|x|}]$. If we have L stacked Transformer blocks, the final output is like:

$$\mathbf{X}^l = \text{Transformer}_l(\mathbf{X}^{l-1}), l \in [1, L] \quad (1)$$

where each block consists of a self-attention layer, a feed-forward layer, residual connection (He et al., 2016) and layer normalization.

Self-Attention For the l-th block, the output $\mathbf{A}_l$ of a self-attention head is:

$$\begin{aligned} &\mathbf{Q} = \mathbf{X}^{l-1}\mathbf{W}_l^Q, \mathbf{K} = \mathbf{X}^{l-1}\mathbf{W}_l^K \\ &\mathbf{M}_{i,j} = \begin{cases} 0, & \text{allow to attend} \\ -\infty, & \text{forbid to attend} \end{cases} \\ &\mathbf{A}_l = \text{SelfAttention}(\mathbf{X}^{l-1}) \\ &\quad = \text{softmax}(\frac{\mathbf{Q}\mathbf{K}^\top}{\sqrt{d_k}} + \mathbf{M})(\mathbf{X}^{l-1}\mathbf{W}_l^V) \end{aligned} \quad (2)$$

where $\mathbf{W}_l^Q, \mathbf{W}_l^K, \mathbf{W}_l^V$ can project previous output to queries, keys, and values, respectively. $\mathbf{M} \in \mathbb{R}^{|x| \times |x|}$ is a mask matrix that controls whether two tokens can attend each other.

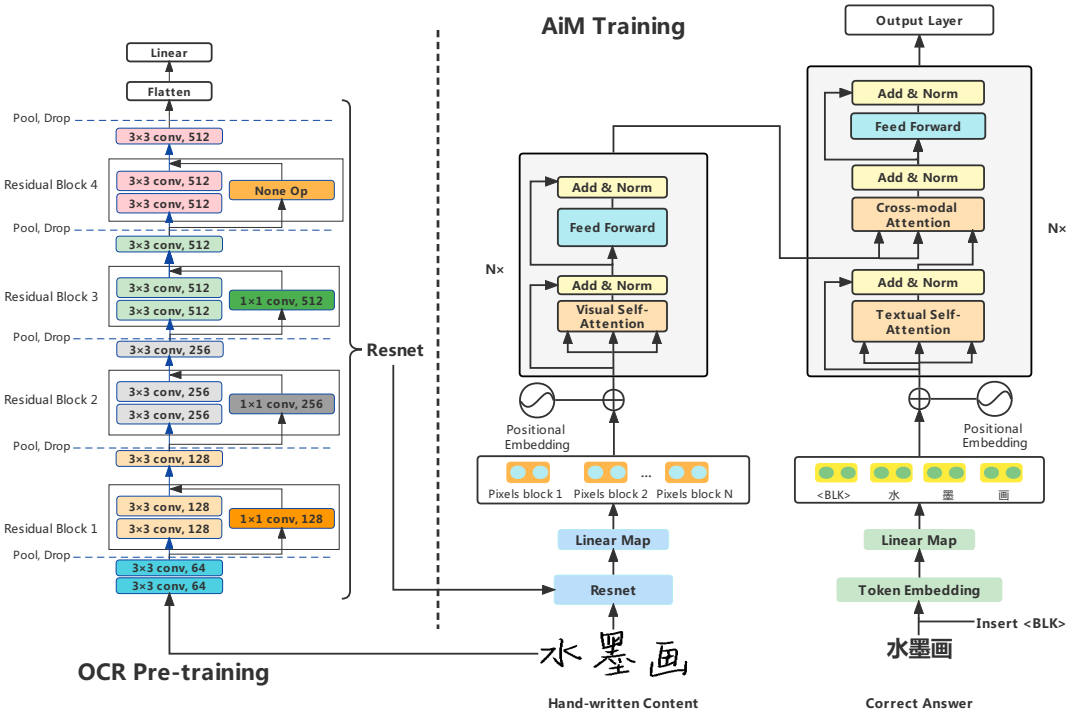

Figure 2: The architecture of AiM. The image encoder of AiM is initialized by OCR pretraining. In AiM, the visual and textual representations are fused by cross-modal attention.

3 Method

We first introduce the sequence labeling conversion for the CCC task in Section 3.1. Model architecture of AiM is shown in Section 3.2 and Figure 2. Finally, we describe the data augmentation and pretraining methods in Section 3.3.

3.1 Label Space

The CCC task is to feedback 'right' or 'wrong' on the handwritten content. In this paper, we perform sequence labeling on the answer text, and the corresponding labels are defined as:

- `del`: the current character does not appear in the image.
- `add`: compared to the handwritten content, one or more characters should be inserted between the current and the next character[3].
- `sub`: compared to the handwritten content, the current character should be substituted by another one.

We get these labels by calculating the edit distance between the answer and the handwritten content[4].

The BIO annotation (Ratinov and Roth, 2009) is adopted that the label space is {`O`, `B-sub`, `I-sub`, `B-del`, `I-del`, `B-add`}. It is similar to the label space of the Grammatical Error Detection/Correction (GED/GEC) task, but our method compares the answer text to the handwritten content, instead of the erroneous sentence to the correct sentence. After this conversion, AiM is trained in a fine-grained way. If the predicted sequence only contains 'O', the correction result of AiM is 'right', otherwise it is 'wrong'.

3.2 Model Architecture

The model architecture of AiM is shown in Figure 2. Components include: i) an image encoder with Resnet and the self-attention mechanism to extract the visual features, ii) a fusion module with the cross-modal attention to mine the interactions between modalities, and iii) an output layer to predict the label sequence.

Image Encoder To understand the handwritten content in images, we follow Liu et al. (2020) to adopt Resnet (He et al., 2016) as the image encoder.

[3]A placeholder <BLK> is inserted at the beginning of the textual answers to handle the character missing at the first position.

[4]The handwritten text is annotated or taken from the OCR dataset for calculating the label sequence of AiM.

The image encoder maps the input image $\mathbf{I}$ to a sequence of visual features $\mathbf{H}_v \in \mathbb{R}^{N_v \times d}$ with a linear transformation:

$$\begin{aligned} \mathbf{H'}_v &= \text{Linear}\left(\text{ResNet}(\mathbf{I})\right) \\ \mathbf{H}_v &= \mathbf{H'}_v + \mathbf{P}_v \end{aligned} \tag{3}$$

where $N_v = \frac{max\_width}{32}$, max_width is the maximum width of input images and d is the hidden size of the visual representation. Each element in sequence represents a fixed-width pixels block in the image. Besides, there is a learnable positional embedding matrix $\mathbf{P}_v \in \mathbb{R}^{N_v \times d}$ where each row is a positional representation for each element in $\mathbf{H}_v$ to capture the location information.

We perform a padding operation on the image with extra white pixels blocks to ensure all images have the same width. Then we adopt Transformer blocks to capture the contextual information in visual modality and the l-th output is:

$$\mathbf{S}_v^l = \text{Transformer}(\mathbf{S}_v^{l-1}), l \in [1, L_v]. \tag{4}$$

where L_v is the number of Transformer blocks. Notes that $\mathbf{S}_v^0 = \mathbf{H}_v$ and $\mathbf{S}_v = \mathbf{S}_v^{L_v}$.

Fusion Module Assuming that the input answer has N_t characters, the answer characters are first converted to dense vectors $\mathbf{X} = [x_1, x_2, ..., x_{N_t}]$ through a word embedding. The linear transformation and positional embedding are also employed to compute the textual representation $\mathbf{H}_t \in \mathbb{R}^{N_t \times d}$ as follows:

$$\begin{aligned} \mathbf{H'}_t &= \text{Linear}(\mathbf{X}) \\ \mathbf{H}_t &= \mathbf{H'}_t + \mathbf{P}_t \end{aligned} \tag{5}$$

where $\mathbf{P}_t$ is the positional embedding matrix.

Then our fusion block further encodes and merges the information of textual and visual modality. As shown in Figure 2, each fusion block contains a textual self-attention layer and a cross-modal attention layer. Self-attention mechanism is employed to encode textual representations:

$$\begin{aligned} \mathbf{S}'_t &= \text{SelfAttention}(\mathbf{H}_t) \\ \mathbf{S}_t &= \text{LayerNorm}(\mathbf{H}_t + \mathbf{S}'_t) \end{aligned} \tag{6}$$

To capture the interactions between them, we develop a cross-modal attention as follows:

$$\begin{aligned} &\mathbf{Q} = \mathbf{S}_t\mathbf{W}^Q, \mathbf{K} = \mathbf{S}_v\mathbf{W}^K \\ &\mathbf{M}^f_{i,j} = \begin{cases} -\infty, & \text{padding token or pixels block} \\ 0, & \text{otherwise} \end{cases} \\ &\mathbf{S}_f = \text{softmax}(\frac{\mathbf{Q}\mathbf{K}^\top}{\sqrt{d}} + \mathbf{M}^f)(\mathbf{S}_v\mathbf{W}^V) \end{aligned} \tag{7}$$

where $\mathbf{W}^Q$ projects $\mathbf{S}_t$ to queries, $\mathbf{W}^K$ and $\mathbf{W}^V$ project $\mathbf{S}_v$ to keys and values, and $\mathbf{M}^f \in \mathbb{R}^{N_t \times N_v}$ is the mask matrix to ensure only valid tokens and pixels blocks can attend to each other. The fused representations $\mathbf{S}_f$ are followed by the feed-forward layer, residual connection, and layer normalization.

Finally, we stack our fusion blocks to obtain more informative fused features. It can be seen that our fusion module is similar to the Transformer decoder. Different from the cross-attention of Transformer, our cross-modal attention merges the visual and textual information, which allows each valid answer character to attend to all valid pixels blocks, without the order restriction of language modeling. Besides, the Transformer decoder decodes words one by one, but our module outputs fused features at once.

Output Layer We denote the above fused features as $\mathbf{H}_f$. To project the hidden representation to the space of labels, the fusion features is fed to the fully-connected layer and a softmax function to get the final output $\mathbf{O} \in \mathbb{R}^{N_t \times m}$:

$$\mathbf{O} = \text{softmax}(\mathbf{H}_f\mathbf{W}_o) \tag{8}$$

where $\mathbf{W_o} \in \mathbb{R}^{d \times m}$ is the weight matrix and m is the number of labels. During the training, we apply the cross-entropy function as our correction objective and the training loss is computed as:

$$\mathcal{L} = -\sum_{i=1}^{N_t} \log p(i, k_i), \;\; p(i, k_i) \in \mathbf{O} \tag{9}$$

where k_i is the label of the i-th character and $p(i, k_i)$ is the probability of the i-th character being predicted to label k_i.

3.3 Data Augmentation and Pretraining

Data Augmentation The sample of the OCR dataset contains an image and the corresponding written text, which can be easily extended for the CCC task. We directly take the annotated text as the answer to augment our positive CCC samples. Then we develop a negative sample augmentation method. Given a CCC sample, we keep the image unchanged and modify the answer text. The modifications include random character insertion, deletion and substitution. Especially for the character substitution, we attempt to construct hard negative samples by replacing the original character

Algorithm 1: Negative Sample Augmentation

Input: $\mathcal{S} = \{(\mathbf{I}_i, \mathbf{C}_i, \mathbf{A}_i, \mathbf{L}_i, y_i)\}_{i=1}^{N}$, where $\mathbf{I}_i$ is the image, $\mathbf{C}_i$ is the handwritten content, $\mathbf{A}_i$ is the answer text, $\mathbf{L}_i$ is the label sequence of AiM and y_i is the correction result.

$D \leftarrow \mathcal{S}$

for $(\mathbf{I}_i, \mathbf{C}_i, \mathbf{A}_i, \mathbf{L}_i, y_i)$ ***in*** $\mathcal{S}$ **do**

  repeat

    $\hat{\mathbf{A}}_i \leftarrow \mathbf{A}_i$

    Randomly select an index j in $\hat{\mathbf{A}}_i$

    $\hat{\mathbf{A}}_i[j] \leftarrow \mathbf{A}_i[j]$'s shape-similar character

    Get $\hat{\mathbf{L}}_i$ comparing $\mathbf{C}_i$ and $\hat{\mathbf{A}}_i$

    $\hat{y}_i = 1$

    D.append $((\mathbf{I}_i, \mathbf{C}_i, \hat{\mathbf{A}}_i, \hat{\mathbf{L}}_i, \hat{y}_i))$

  until *Random times*

  repeat

    $\hat{\mathbf{A}}_i \leftarrow \mathbf{A}_i$

    Randomly select an index j in $\hat{\mathbf{A}}_i$

    Delete $\hat{\mathbf{A}}_i[j]$

    Get $\hat{\mathbf{L}}_i$ comparing $\mathbf{C}_i$ and $\hat{\mathbf{A}}_i$

    $\hat{y}_i = 1$

    D.append $((\mathbf{I}_i, \mathbf{C}_i, \hat{\mathbf{A}}_i, \hat{\mathbf{L}}_i, \hat{y}_i))$

  until *Random times*

  repeat

    $\hat{\mathbf{A}}_i \leftarrow \mathbf{A}_i$

    Randomly select an index j in $\hat{\mathbf{A}}_i$

    Insert a common Chinese character at $\hat{\mathbf{A}}_i[j]$

    Get $\hat{\mathbf{L}}_i$ comparing $\mathbf{C}_i$ and $\hat{\mathbf{A}}_i$

    $\hat{y}_i = 1$

    D.append $((\mathbf{I}_i, \mathbf{C}_i, \hat{\mathbf{A}}_i, \hat{\mathbf{L}}_i, \hat{y}_i))$

  until *Random times*

Output: the augmented dataset D

with shape-similar ones. The shape-similar characters are derived from an open-sourced confusion set[5] (Wu et al., 2013). The pseudo code of negative sample augmentation is presented in Algorithm 1.

Pretraining Before the AiM training, we first pretrain our image encoder with the OCR objective. We follow the Liu et al. (2020) to use CTC (Graves et al., 2006) as the loss function and apply high dropout rates after each max-pooling layer. As shown on the left side in Figure 2, the input image is converted to a sequence of dense representations through Resnet. A linear layer is added on the top of Resnet to transform the space dimension to the size of the vocabulary.

4 Experiment

In this section, we first introduce datasets, baselines, and other details of our experiments. Then we show experimental results and perform analyses from different views.

[5]It is taken from URL.

Dataset	Train set		Dev set		Test set	
	#img	#sample	#img	#sample	#img	#sample
EinkCC	4256	10610	-	-	673	673
HWCC	41781	166984	10499	41767	-	-
SynCC	150594	602376	-	-	-	-
Total	196631	779970	10499	41767	673	673

Table 1: Data statistics. "#img" means the number of handwriting images. "#sample" means the number of CCC samples. In each train or dev set, the "#sample" is larger than "#img" because of our data augmentation in Section 3.3.

Dataset	OCR Model	CER
HWCC	PP-OCRv2	0.3936
	CNN-CTC-CBS	0.2943
	Resnet-CTC	**0.0663**
EinkCC	PP-OCRv2	0.3115
	CNN-CTC-CBS	0.2325
	Resnet-CTC	**0.1661**

Table 2: The OCR performance on dev and test sets.

4.1 Dataset

EinkCC EinkCC is collected from our educational application. Teachers distribute cloze tests in our app, and students practice on the e-ink display hardware. When students finish tests, teachers can correct them and feedback to students. Each sample in EinkCC contains a student's handwriting image, the answer text and the correction result marked by teachers. It mainly covers the dictation of ancient poetry, the idiom application, and the reading comprehension. Besides, we manually annotate the image text for the OCR training and the AiM label sequence calculation.

HWCC CASIA-HWDB (Liu et al., 2011) is a benchmark for the handwritten Chinese text recognition(HCTR) task. We select the HWDB 2.x set to be the positive CCC samples.

SynCC To further enlarge the scale of CCC datasets, we first build a synthetic OCR dataset. The details of the data construction are presented in Appendix. Then all OCR samples are taken as the positive CCC samples.

We split EinkCC into train and test sets, and split HWCC into train and dev sets. All train sets and dev sets are extended by our negative sample augmentation method of Section 3.3. The data statistics are shown in Table 1.

Dataset	Model	Sequence level			Binary level			
		P	R	F1	P	R	F1	Acc
HWCC	PP-OCRv2	0.3590	0.4661	0.4057	0.7508	**0.9997**	0.8576	0.7510
	CNN-CTC-CBS	0.4584	0.5563	0.5026	0.7604	0.9994	0.8637	0.7635
	Resnet-CTC	0.8157	0.8877	0.8502	0.8578	0.9989	0.9230	0.8751
	$\text{AiM}_{\text{wo-PT}}$	0.4893	0.5602	0.5223	0.9083	0.9306	0.9193	0.8571
	AiM	**0.9735**	**0.9717**	**0.9726**	**0.9926**	0.9937	**0.9932**	**0.9898**
EinkCC	PP-OCRv2	0.1146	0.6847	0.1964	0.2128	**1.0000**	0.3509	0.4502
	CNN-CTC-CBS	0.1829	**0.8108**	0.2985	0.2710	**1.0000**	0.4264	0.6003
	Resnet-CTC	0.2311	0.7748	0.3561	0.3322	0.9964	0.4987	0.7013
	$\text{AiM}_{\text{wo-PT}}$	0.0568	0.2252	0.0907	0.2025	0.6497	0.3088	0.5676
	AiM	**0.2795**	0.5766	**0.3765**	**0.4262**	0.7799	**0.5512**	**0.8113**

Table 3: Main results on dev and test sets. $\text{AiM}_{\text{wo-PT}}$ is the AiM model without the OCR pretraining. Resnet-CTC is trained using the same data source as AiM, which is a strong OCR baseline in a fair comparison.

Model	CER
$\text{Resnet-CTC}_{\text{wo-Syn}}$	0.1916
$\text{Resnet-CTC}_{\text{wo-E}}$	0.2995
$\text{Resnet-CTC}_{\text{wo-H}}$	0.3304
Resnet-CTC	**0.1661**

Table 4: Data ablation of the Resnet-CTC OCR performance on the EinkCC test set.

Model	Sequence Level		
	P	R	F1
$\text{AiM}_{\text{wo-Syn}}$	0.0212	0.1712	0.0378
$\text{AiM}_{\text{wo-E}}$	0.1857	0.3964	0.2529
$\text{AiM}_{\text{wo-H}}$	0.2660	0.4865	0.3440
AiM	**0.2795**	**0.5766**	**0.3765**

Table 5: Data ablation of the AiM performance at the sequence level on the EinkCC test set.

4.2 Evaluation Metrics

We use the character error rate (CER) to evaluate the OCR performance in the pretraining stage, and compute the following metrics to evaluate the CCC performance:

Sequence level We use the widely-used metrics, Precision (P), Recall (R), and $F1$, to measure the quality of label sequences outputted by AiM.

Binary level We transform label sequences to binary labels ('right' or 'wrong') to compute the accuracy of binary classification. We report the $P, R,$ and $F1$ of negative samples (i.e. $y = 1$) to evaluate model's ability to identify wrong answers.

4.3 Implementation Details and Baselines

Our training dataset $\{(\mathbf{I}_i, \mathbf{C}_i, \mathbf{A}_i, \mathbf{L}_i, y_i)\}_{i=1}^N$ contains the image $\mathbf{I}_i$, the handwritten content $\mathbf{C}_i$, the answer text $\mathbf{A}_i$, the label sequence of AiM $\mathbf{L}_i$ and the correction result y_i. We pretrain the image encoder Resnet on the subset $\{(\mathbf{I}_i, \mathbf{C}_i)\}_{i=1}^N$, and train the AiM model on the subset $\{(\mathbf{I}_i, \mathbf{A}_i, \mathbf{L}_i)\}_{i=1}^N$.

In the OCR pretraining, we follow Liu et al. (2020) to set the number of convolution blocks to {2,3,1,4}. The height of every input image is set to 128 pixels and the width is scaled to the corresponding value, so the shape of each pixels block is 128×32. The learning rate is set to 1e-3, the batch size is 8 and the training epoch is 15.

In the AiM training, the learning rate is 1e-4, the batch size is 8 and the training epoch is set to 14. The dimension of hidden states d is 768. The number of image encoder blocks N_{enc} and fusion module blocks N_{fus} in AiM are both 2. We adopt AdamW optimizer (Loshchilov and Hutter, 2018) and cosine-annealing strategy. To accelerate training, the parameters in Resnet are frozen when training AiM. We ignore punctuation in the text.

OCR-based methods are our baselines, where the correction results are derived from the post-comparison between the OCR results and answers. The following OCR models are evaluated: **Resnet-CTC** is the image encoder of AiM (Resnet) trained with the CTC function on the subset $\{(\mathbf{I}_i, \mathbf{C}_i)\}_{i=1}^N$. **PP-OCRv2** (Du et al., 2021) is an open-sourced widely-used OCR model in Chinese recognition, and we take its text recognition server model[6] for evaluation. **CNN-CTC-CBS** (Liu et al., 2020) is a handwritten text-line recognition model, and we take the released well-trained model[7] for evaluation. Table 2 shows the OCR performance of baselines on our datasets.

[6] https://github.com/PaddlePaddle/PaddleOCR.

[7] https://github.com/intel/handwritten-chinese-ocr-samples.

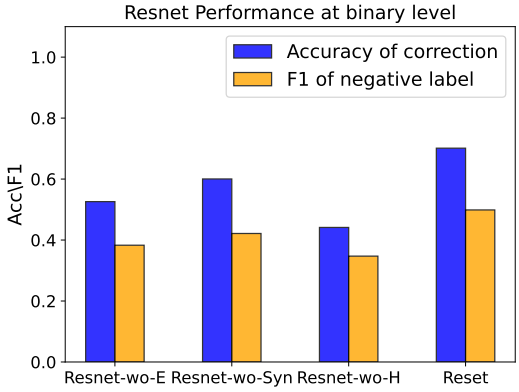

(a) Resnet performance at binary level

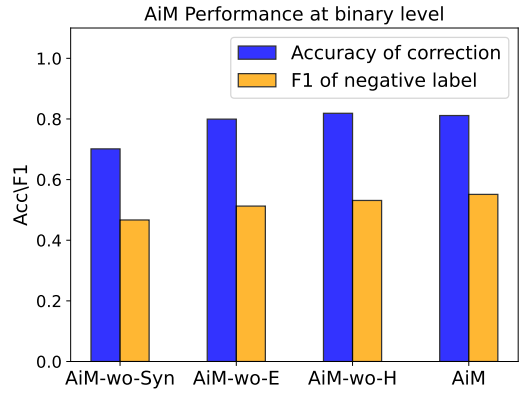

(b) AiM performance at binary level

Figure 3: Data ablation of the AiM and its baseline performance at the binary level on the EinkCC test set.

4.4 Results and Analyses

The main results are shown in Table 3. We evaluate OCR models at the OCR level and the binary level. For AiM and its variants, we evaluate them at the sequence level and the binary level. We describe our observations from the following perspectives:

- How do OCR models perform in CCC?
- Does AiM improve the performance and whether the two modalities are well fused?
- Do data augmentation and pretraining work and what's the impact of data source?

4.4.1 Limitation of OCR-based Method

As shown on Table 3, the recall of OCR-based methods is extremely high (nearly 1.0), but the precision and F1 are much lower. This means that OCR models can correct almost all wrong handwritten answers but mark many students' right text as wrong. Oppositely, AiM improves the precision, F1, and accuracy with large margins.

Image:	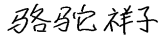	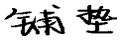
Handwritten:	骆驼祥子	铺垫
Answer:	骆驼祥子	铺垫
OCR result:	骆马它祥子	铺挚
AiM output labels:	O O O O O	O O O
Correction label:	✓	✓
Correcting by OCR:	×	×
Correcting by AiM:	✓	✓

Table 6: Examples on EinkCC test set. Resnet gives the wrong outputs in all examples. The mistakes made by Resnet are shown in bold with underline, while AiM predicts correctly. Notes that the first label is for the placeholder '<BLK>'.

4.4.2 Influence of Data Source

We conduct an ablation study on the training data. The suffix '-wo-E', '-wo-Syn', '-wo-H' means the model is trained without the EinkCC training set, SynCC set and HWCC training set, respectively. The performance of Resnet and its variants is shown in Table 4 and Figure 3(a). The CER reaches the lowest level when Resnet is trained with all training sets, which can prove the necessity of data extension for OCR models. Table 5 and Figure 3(b) show the performance of AiM. Removing any training set, all metrics at sequence level drops, as well as the binary level. It means sufficient data is valuable in the CCC task. Meanwhile, comparing Figure 3(a) and 3(b), although the data scale decreases, the performance of AiM is always better than Resnet, which demonstrates the AiM is more robust and stable to data variation.

Notice that the model performance are always better on the HWCC dev set than it on the EinkCC test set. This is because all negative samples of HWCC are synthetic samples that are augmented in the same way. This suggests that AiM learns the error patterns well from our negative sample augmentation.

4.4.3 Effectiveness of Multimodal

As shown in Table 3, with AiM, the recall of negative label decreases[8] but the precision and F1-score increase significantly at the binary level. We give two examples on EinkCC in Table 6 to further analyze the effectiveness of AiM. Obviously, Resnet generates wrong outputs while AiM makes no mistake. In the first example, Resnet considers one left-right structured character as two independent characters, which never happens when manually correcting. In the second example, Resnet gets

[8]We perform an analysis on the recall drop in Appendix.

			Sequence level			Binary level			
N_{enc}	N_{fus}	Text Self-Att in Fusion	P	R	F1	P	R	F1	Acc
1	1	×	0.1150	0.3513	0.1733	0.3515	0.7099	0.4702	0.7623
1	1	✓	0.1773	0.4775	0.2585	0.3794	**0.7908**	0.5128	0.7727
2	2	×	0.1235	0.3694	0.1851	0.3630	0.7161	0.4818	0.7667
2	2	✓	**0.2795**	**0.5766**	**0.3765**	**0.4262**	0.7799	**0.5512**	**0.8113**
3	3	✓	0.2501	0.5495	0.3436	0.3682	0.7398	0.4917	0.7727

Table 7: An ablation study of components in AiM on EinkCC test set.

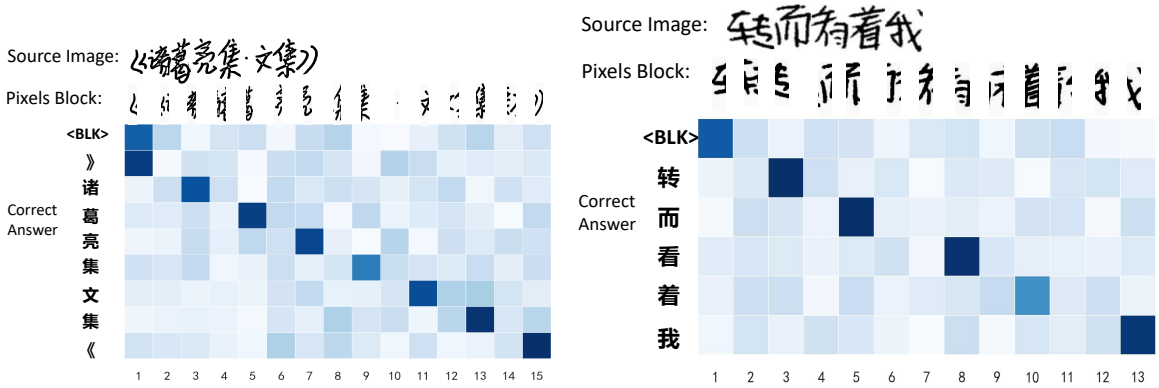

Figure 4: Cross-modal attention visualization. The figure shows the cross-attention scores between characters in answer and the pixels blocks in image.

confused by the shape-similar characters. AiM predicts successfully in these examples. More predictions on the dev and test sets are presented in Appendix to illustrate that AiM is able to identify all situations of the mismatch between the image and answer text.

Comparing AiM and $\text{AiM}_{\text{wo-PT}}$ in Table 3, without the pretraining, the performance of AiM drops significantly, even worse than Resnet. It demonstrates that visual modality features can be better understood through pretraining.

4.4.4 Learning of Modal Fusion

We conduct experiments to explore the contribution of each component. Experimental results are shown in Table 7. AiM reaches a better performance when encoding text information with the self-attention than only using token embeddings. This suggests that self-attention brings better textual representation. Meanwhile, stacking each module with a proper number of layers helps AiM capture interactions between modalities. We visualize the cross-modal attention scores in Figure 4. We find that characters manage to attend the most relative pixels block. For instance, in this sub-figure (a), the character "集" appears twice in different positions, and the model captures the context feature and both of them attend the corresponding pixels block in order. It shows that cross-modal attention can match multimodal information effectively.

5 Related Work

5.1 OCR Model

OCR models have evolved for a long period. Researchers usually used hybrid CNN and RNN architectures (Breuel, 2017) with CTC (Connectionist temporal classification) loss (Graves et al., 2006). For handwritten Chinese text recognition (HCTR) task, Liu et al. (2020) has achieved the state-of-the-art performance. They use the simple end-to-end CNN-CTC method and ease the overfitting problem with a high-rate dropout strategy.

5.2 Multimodal Model

Multimodal models have attracted more and more attention and have been applied in many fields, such as visual question answering (Antol et al., 2015), Chinese spell checking (Xu et al., 2021; Li et al., 2022) and other applications (Toto et al., 2021; Hu et al., 2021; Aguilar et al., 2019). Unsupervised pretraining (Kenton and Toutanova, 2019) has provided informative representations and fine-tuning techniques (Cui et al., 2019; Li et al., 2021)

have brought further performance gains. Several large-scale pre-trained models are utilized in multimodal models (Anderson et al., 2018; Toto et al., 2021; Xu et al., 2021). To capture multimodal features, one common method is to concatenate encoders' output from two sides and then feed to the downstream multi-layer perceptrons (Nie et al., 2021). While some works use attention mechanism to get fusion representation (Lu et al., 2019; Tan and Bansal, 2019; Tsai et al., 2019).

6 Conclusion

In this paper, we propose a multimodal model AiM to effectively correct Chinese cloze tests. AiM employs cross-modal attention to understand the correlation between modalities. We collect data from different sources and develop a data augmentation method. Experiments show that AiM outperforms traditional OCR-based methods with over 11% accuracy improvements.

References

Gustavo Aguilar, Viktor Rozgić, Weiran Wang, and Chao Wang. 2019. Multimodal and multi-view models for emotion recognition. *arXiv preprint arXiv:1906.10198*.

Peter Anderson, Xiaodong He, Chris Buehler, Damien Teney, Mark Johnson, Stephen Gould, and Lei Zhang. 2018. Bottom-up and top-down attention for image captioning and visual question answering. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 6077–6086.

Stanislaw Antol, Aishwarya Agrawal, Jiasen Lu, Margaret Mitchell, Dhruv Batra, C. Lawrence Zitnick, and Devi Parikh. 2015. VQA: visual question answering. *CoRR*, abs/1505.00468.

Thomas M Breuel. 2017. High performance text recognition using a hybrid convolutional-lstm implementation. In *2017 14th IAPR international conference on document analysis and recognition (ICDAR)*, volume 1, pages 11–16. IEEE.

Baiyun Cui, Yingming Li, Ming Chen, and Zhongfei Zhang. 2019. Fine-tune BERT with sparse self-attention mechanism. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 3548–3553, Hong Kong, China. Association for Computational Linguistics.

Yuning Du, Chenxia Li, Ruoyu Guo, Cheng Cui, Weiwei Liu, Jun Zhou, Bin Lu, Yehua Yang, Qiwen Liu, Xiaoguang Hu, Dianhai Yu, and Yanjun Ma. 2021. Pp-ocrv2: Bag of tricks for ultra lightweight OCR system. *CoRR*, abs/2109.03144.

Alex Graves, Santiago Fernández, Faustino Gomez, and Jürgen Schmidhuber. 2006. Connectionist temporal classification: labelling unsegmented sequence data with recurrent neural networks. In *Proceedings of the 23rd international conference on Machine learning*, pages 369–376.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. 2016. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778.

Chuanbo Hu, Minglei Yin, Bin Liu, Xin Li, and Yanfang Ye. 2021. Detection of illicit drug trafficking events on instagram: A deep multimodal multilabel learning approach. In *Proceedings of the 30th ACM International Conference on Information & Knowledge Management*, pages 3838–3846.

Jacob Devlin Ming-Wei Chang Kenton and Lee Kristina Toutanova. 2019. Bert: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of NAACL-HLT*, pages 4171–4186.

Yinghui Li, Qingyu Zhou, Yangning Li, Zhongli Li, Ruiyang Liu, Rongyi Sun, Zizhen Wang, Chao Li, Yunbo Cao, and Hai-Tao Zheng. 2022. The past mistake is the future wisdom: Error-driven contrastive probability optimization for Chinese spell checking. In *Findings of the Association for Computational Linguistics: ACL 2022*, pages 3202–3213, Dublin, Ireland. Association for Computational Linguistics.

Zhongli Li, Qingyu Zhou, Chao Li, Ke Xu, and Yunbo Cao. 2021. Improving BERT with syntax-aware local attention. In *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*, pages 645–653, Online. Association for Computational Linguistics.

Brian Liu, Xianchao Xu, and Yu Zhang. 2020. Offline handwritten chinese text recognition with convolutional neural networks. *arXiv preprint arXiv:2006.15619*.

Cheng-Lin Liu, Fei Yin, Da-Han Wang, and Qiu-Feng Wang. 2011. Casia online and offline chinese handwriting databases. In *2011 International Conference on Document Analysis and Recognition*, pages 37–41. IEEE.

Ilya Loshchilov and Frank Hutter. 2018. Decoupled weight decay regularization. In *International Conference on Learning Representations*.

Jiasen Lu, Dhruv Batra, Devi Parikh, and Stefan Lee. 2019. Vilbert: Pretraining task-agnostic visiolinguistic representations for vision-and-language tasks. *Advances in neural information processing systems*, 32.

Yixin Nie, Linjie Li, Zhe Gan, Shuohang Wang, Chenguang Zhu, Michael Zeng, Zicheng Liu, Mohit Bansal, and Lijuan Wang. 2021. Mlp architectures for vision-and-language modeling: An empirical study. *arXiv preprint arXiv:2112.04453*.

Lev Ratinov and Dan Roth. 2009. Design challenges and misconceptions in named entity recognition. In *Proceedings of the Thirteenth Conference on Computational Natural Language Learning (CoNLL-2009)*, pages 147–155, Boulder, Colorado. Association for Computational Linguistics.

Hao Tan and Mohit Bansal. 2019. Lxmert: Learning cross-modality encoder representations from transformers. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 5100–5111.

Ermal Toto, ML Tlachac, and Elke A Rundensteiner. 2021. Audibert: A deep transfer learning multimodal classification framework for depression screening. In *Proceedings of the 30th ACM International Conference on Information & Knowledge Management*, pages 4145–4154.

Yao-Hung Hubert Tsai, Shaojie Bai, Paul Pu Liang, J Zico Kolter, Louis-Philippe Morency, and Ruslan Salakhutdinov. 2019. Multimodal transformer for unaligned multimodal language sequences. In *Proceedings of the conference. Association for Computational Linguistics. Meeting*, volume 2019, page 6558. NIH Public Access.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. *Advances in neural information processing systems*, 30.

Shih-Hung Wu, Chao-Lin Liu, and Lung-Hao Lee. 2013. Chinese spelling check evaluation at SIGHAN bake-off 2013. In *Proceedings of the Seventh SIGHAN Workshop on Chinese Language Processing*, pages 35–42, Nagoya, Japan. Asian Federation of Natural Language Processing.

Heng-Da Xu, Zhongli Li, Qingyu Zhou, Chao Li, Zizhen Wang, Yunbo Cao, Heyan Huang, and Xian-Ling Mao. 2021. Read, listen, and see: Leveraging multimodal information helps Chinese spell checking. In *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*, pages 716–728, Online. Association for Computational Linguistics.

A Annotation Process

We present the process of annotation in our sequence labeling task in Figure 5. We put a label to individual character in textual answer referring to hand-written content and each label stands for a kind of editing operation.

B SynCC Data Construction

To build a synthetic OCR dataset, we first collect handwritten character images from the HWDB 1.0 set of CASIA-HWDB (Liu et al., 2011) and our educational application. Then the character images are spliced together into text-line images according to the sentences or clauses of our online essay corpus. We also replace characters with shape-similar ones and their images to enhance OCR models to recognize them. The synthetic OCR examples are shown on Table 8. These samples are taken as the positive CCC samples, and negative sample augmentation introduced in Section 3.3 is applied to construct SynCC dataset.

Image:	
Content:	是一个宁静的世界
Image:	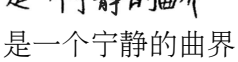
Content:	是一个宁静的曲界
Image:	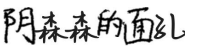
Content:	阴森森的面孔
Image:	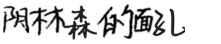
Content:	阴林森的面孔
Image:	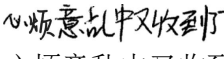
Content:	心烦意乱中又收到了一页书笔
Image:	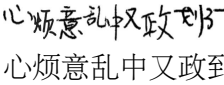
Content:	心烦意乱中又政到了一页书笔
Image:	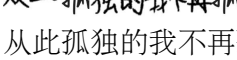
Content:	从此孤独的我不再孤独
Image:	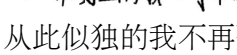
Content:	从此似独的我不再孤独

Table 8: Synthetic OCR examples of SynCC.

C Recall Drop on EinkCC

Table 3 shows that AiM improves the correction precision and F1 but the recall drops on EinkCC. We present several false-positive predictions on Table 9. All examples are annotated as wrong by teachers but are predicted as right by AiM. In the first example, the student writes down "斯" but there is a extremely large margin between the left and right sides. In the rest examples, students write down shape-similar characters but AiM can't detect them and tend to predict as right. Thus, even though we use a confusion set to enhance AiM to detect shape-similar characters, there is still a lot of room for improvement in this regard.

Image:	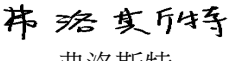
Answer:	弗洛斯特
Labels:	O O O B-sub O
Image:	
Answer:	清
Labels:	O B-sub
Image:	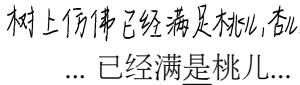
Answer:	... 已经满是桃儿...
Labels:	... O O O B-sub O O ...
Image:	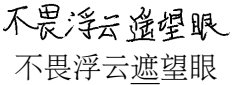
Answer:	不畏浮云遮望眼
Labels:	O O O O O B-sub O O

Table 9: False-positive predictions on EinkCC test set. All above answer characters are predicted to 'O' by AiM. The character marked by the underline indicates why the correction result should be wrong.

D Predictions of AiM

Table 10 shows several cases of AiM's predictions. AiM is able to identify all kinds of modifications between the answer and hand-written content including substitutions, deletions and insertions, which proves that it's reasonable to format the fill-in-the-blank assignments correction task to sequence labeling. In this way, AiM can not only indicate the correctness of students' answers, but also can locate where the errors occur.

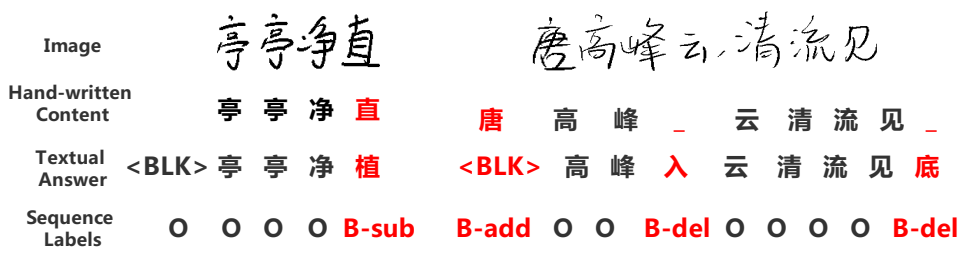

Figure 5: Examples of sequence labeling annotation. For the first sample, to convert the answer to the hand-written content, the last character in the answer "植" should be replaced by "直". So the label 'B-sub' is annotated to "植" and 'O' for the rest. For the latter one, the character "入" and "底" do not appear in the hand-written content so the labels are both 'B-del'. Moreover, the character "唐" should be inserted at the first position in answer for conversion, so we put a 'B-add' label to the placeholder '<BLK>'.

Image:	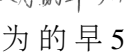
Handwritten Content:	前 认 为 的 早 5 亿 年
Correct Answer:	麻 前 镅 认 为 的 早 5 亿 材 嚓 年
AiM Labels:	O B-del O B-del I-del O O O O O O B-del I-del O
AiM Predictions:	O B-del O B-del I-del O O O O O O B-del I-del O
Image:	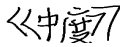
Handwritten Content:	《 中 度 》
Correct Answer:	《 中 庸 》
AiM Labels:	O O O B-sub O
AiM Predictions:	O O O B-sub O
Image:	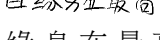
Handwritten Content:	自 缘 身 在 最 高 层
Correct Answer:	不 畏 浮 云 遮 望 眼
AiM Labels:	O B-sub I-sub I-sub I-sub I-sub I-sub I-sub
AiM Predictions:	O B-sub I-sub I-sub I-sub I-sub I-sub I-sub
Image:	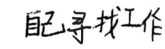
Handwritten Content:	自 己 寻 找 工 作
Correct Answer:	自 寻 工 作
AiM Labels:	O B-add B-add O O
AiM Predictions:	O B-add B-add O O

Table 10: Several outputs of AiM on dev and test sets. Some answers are generated through our data augmentation. AiM can identify editing operations between answers and hand-written content. Characters are shown in bold with underline if the corresponding label is not 'O'.

TreeMAN: Tree-enhanced Multimodal Attention Network for ICD Coding

Zichen Liu[†], **Xuyuan Liu**[†], **Yanlong Wen**[†*], **Guoqing Zhao**[‡], **Fen Xia**[‡], **Xiaojie Yuan**[†]
[†]College of Computer Science, Nankai University
[‡]Mashang Consumer Finance Co.,Ltd.
{liuzichen,wenyl,yuanxj}@dbis.nankai.edu.cn
hsuyuanliu0204@mail.nankai.edu.cn
{guoqing.zhao02,fen.xia}@msxf.com

Abstract

ICD coding is designed to assign the disease codes to electronic health records (EHRs) upon discharge, which is crucial for billing and clinical statistics. In an attempt to improve the effectiveness and efficiency of manual coding, many methods have been proposed to automatically predict ICD codes from clinical notes. However, most previous works ignore the decisive information contained in structured medical data in EHRs, which is hard to be captured from the noisy clinical notes. In this paper, we propose a **Tree**-enhanced **M**ultimodal **A**ttention **N**etwork (TreeMAN) to fuse tabular features and textual features into multimodal representations by enhancing the text representations with *tree-based features* via the attention mechanism. *Tree-based features* are constructed according to decision trees learned from structured multimodal medical data, which capture the decisive information about ICD coding. We can apply the same multi-label classifier from previous text models to the multimodal representations to predict ICD codes. Experiments on two MIMIC datasets show that our method outperforms prior state-of-the-art ICD coding approaches. The code is available at https://github.com/liu-zichen/TreeMAN.

1 Introduction

The International Classification of Diseases (ICD), maintained by the World Health Organization, is a hierarchical classification of codes representing diseases, injuries, and so on. ICD codes have been used in diverse areas, including insurance reimbursement, epidemiology, and clinical research (Park et al., 2000).

In the hospital, when patients discharge, their electronic health records (EHRs) and all associated data are transferred to the information management department, where clinical coders manually assign the appropriate ICD codes using rigid ICD coding guidelines after reviewing records (O'malley et al., 2005). The manual code assignment is expensive, labor-intensive, and error-prone due to the large volume of medical record information and high professional requirements (Nguyen et al., 2018).

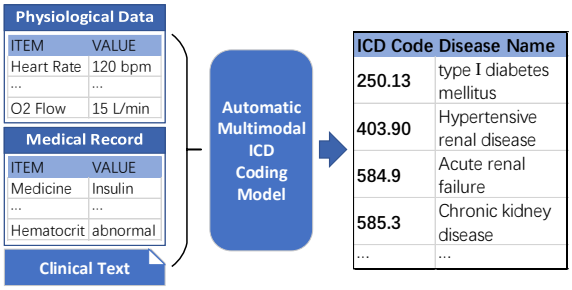

Figure 1: An example of automatic multimodal ICD coding. Model inputs include physiological data and medical records in addition to clinical text.

Since deep learning has achieved great success in lots of healthcare applications (Cai et al., 2019), many neural methods have been proposed to automate the ICD coding process by researchers (Teng et al., 2022). Recent works formulate automated ICD coding as a multi-label document classification task, using clinical notes as model input, predicting coding with a multi-label classifier, and learning text features through word embedding techniques and neural networks such as RNNs and CNNs (Mullenbach et al., 2018; Vu et al., 2020; Zhou et al., 2021). To improve the code representation learning, researchers further leverage features of ICD codes such as hierarchical structures (Cao et al., 2020) and descriptions (Mullenbach et al., 2018; Zhou et al., 2021). However, most previous methods ignore structured medical data, including physiological data collected by medical sensors and medical record information such as prescriptions and microbiology test results in EHRs. The few methods that leverage structured data are either ensemble-based approaches (Xu et al., 2019)

*Corresponding author

Proceedings of the 29th International Conference on Computational Linguistics, pages 3054–3063
October 12–17, 2022.

or data-mining methods that discard semantic information (Ferrão et al., 2021).

In this work, we argue that structured medical data can improve coding accuracy by enhancing semantic representations and providing more information because clinical notes are noisy and ambiguous. For example, there are many different types of insulin, like "Insulin Aspart" and "Insulin Glargine", which are often written the same in notes but clearly distinguished by Generic Sequence Numbers (GSNs) in medical records. Considering different writing styles and polysemous abbreviations, predicting ICD codes from clinical notes is more complicated. However, automatic multimodal ICD coding (as Figure 1 shown) is challenging for the following reasons: 1) medical data is naturally heterogeneous, with data types including numerical quantities, categorical values, and derived time series such as perioperative vital sign signals (Zhou et al., 2021); 2) the feature selection method needs to be designed especially for multi-ICD codes as it's a multi-label classification task; 3) decisive information for a code in the long clinical note may be contained in short segments that are likely different for different codes (Mullenbach et al., 2018).

In this paper, we propose a novel **Tree**-enhanced **M**ultimodal **A**ttention **N**etwork named TreeMAN to address the aforementioned problems. Since it's hard to do feature engineering for structured medical data, we construct *tree-based features* from the structured medical data through decision trees that require little data preparation (Safavian and Landgrebe, 1991) instead of manually crafting features based on medical knowledge. Inspired by previous works (Wang et al., 2018; He et al., 2014), we represent the tree-based features by embedding vectors. Taking the tree-based embeddings and text representations as input, TreeMAN applies an attention mechanism to select relevant tree-based features for text representations and output fused multimodal representations that contain richer information to benefit the downstream classifier. However, our method has limitations in handling long-tailed labels as it is difficult to build a decision tree from less than 10 positive samples.

Contributions. In summary, the main contributions of our work include:

- We propose a multimodal ICD coding framework that exploits structured medical data in EHRs to construct tree-based features to enhance text representations.
- We propose TreeMAN, a tree-enhanced multimodal attention network, which fuses text representations and tree-based features into unified multimodal representations by the attention mechanism. To the best of our knowledge, it's the first model to jointly learn multimodal features for the ICD coding task.
- Experiments demonstrate the effectiveness of our proposed method. Results on two datasets show that TreeMAN outperforms previous state-of-the-art ICD coding methods.

2 Related Work

2.1 ICD Coding

Research on Automatic ICD coding can be traced back to nearly 30 years ago when Larkey and Croft (1996) proposed an ensemble algorithm to integrate different types of classifiers to assign ICD codes to inpatient discharge summaries. A series of methods based on Deep Neural Networks has been implemented on this task since this paradigm achieved colossal success in Clinical NLP. Perotte et al. (2014) built "hierarchical" Support Vector Machines (SVMs) outperforming the "flat" classifier. Mullenbach et al. (2018) built a convolutional attention model which combined the single filter CNN module and the per-label attention module. A series of network modules based on attention mechanism have been utilized after the early attempts, including multi-scale attention module (Xie et al., 2019), residual convolution module (Li and Yu, 2020). We also notice that the hierarchical structure of ICD-9 could be effectively described by a joint-classification module on different levels (Vu et al., 2020) or in the form of specific hyperbolic representation (Cao et al., 2020).

Multimodal learning methods help to integrate multiple information like test reports, nursing notes, etc., in the MIMIC-III datasets. An early attempt was made by (Xu et al., 2019), in which an ensemble-based approach was developed to integrate the structured and unstructured text of different modalities. Rajendran et al. (2021) made full use of unstructured information by effectively exploiting the geometric properties of pre-trained word embeddings.

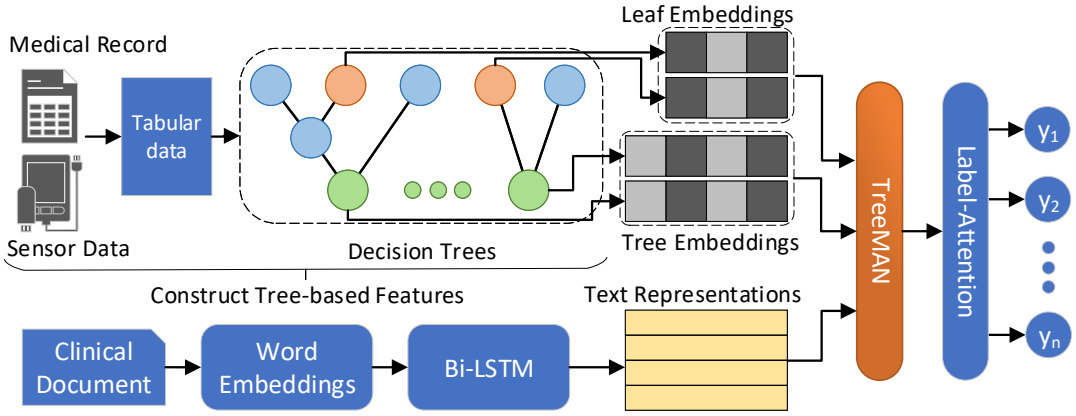

Figure 2: An overview of our proposed multimodal ICD coding framework.

2.2 Tree-based Method

Decision trees are a supervised learning algorithm broadly applied in regression and classification tasks (Quinlan, 1986). They are trained on labeled data while requiring little data preparation and domain knowledge while the preprocessed features are able to be fused with text representations easily. Multiple skills have been implemented to ensemble relatively simple decision trees to get better performance (Banfield et al., 2007; Gashler et al., 2008). Among all of these ideas, Gradient boosting decision tree (GBDT) is an important instance which introduces iterative functional gradient descent algorithms to boosting models firstly (Friedman, 2001). Significant improvement made by XGBoost (Chen and Guestrin, 2016) and LightGBM (Ke et al., 2017) which use different gradient information to improve accuracy and training efficiency respectively. Many attempts (Trofimov et al., 2012; Ling et al., 2017) have been made based on decision-tree boosting algorithm since people found it could generate interpretable and effective cross-feature and is easy to fix with other models. A combination of GBDT with linear model like Logistic Regression(LR) effectively helps the models to make explainable predictions by selecting top cross features (He et al., 2014). Wang et al. (2018) argued that the tree-enhanced embedding method would benefit from the explainability of tree-based models, thus improving generalization ability compared with other pure embedding ways. Incorporating decision tree learning with matrix factorization would help to extract the latent factors and get Fine-Grained embedding with rich semantic information (Kim et al., 2020), which could contribute to solve cold-start problems even further (Tao et al., 2019; Zhou et al., 2011).

3 Method

In this section, we first give an overview of our framework (Section 3.1), and then detail the key module in our framework: the tree-enhanced multimodal attention network TreeMAN (Section 3.2). Finally, we introduce the processing of structured medical data and decision tree learning (Section 3.3).

3.1 Overview

Figure 2 shows the overview of our method. Upon discharge, there are two types of data available for our model: clinical notes written by doctors and structured medical data, including physiological data collected by sensors and medical records, such as lab measurements and prescriptions. Given a clinical note and the associated structured data, two modules in the model process them separately to obtain text representations and tree-based features. Considering in the poor performance of Bert-like models on ICD coding (Zhang et al., 2020; Chalkidis et al., 2020), we train the text model from scratch instead of fine-tuning a pretrained language model. Structured medical data is first processed as tabular data and then fed into a trained decision tree to obtain tree-based features that we project into embedding vectors: the tree embeddings **T** and the leaf embeddings **L** (detailed in Section 3.3). The other module is the text encoder designed to capture the semantic information in the document and provide textual representations.

Text encoder Given an input document with N words $\{w_i\}_{i=1}^{N}$, the encoder first maps each word

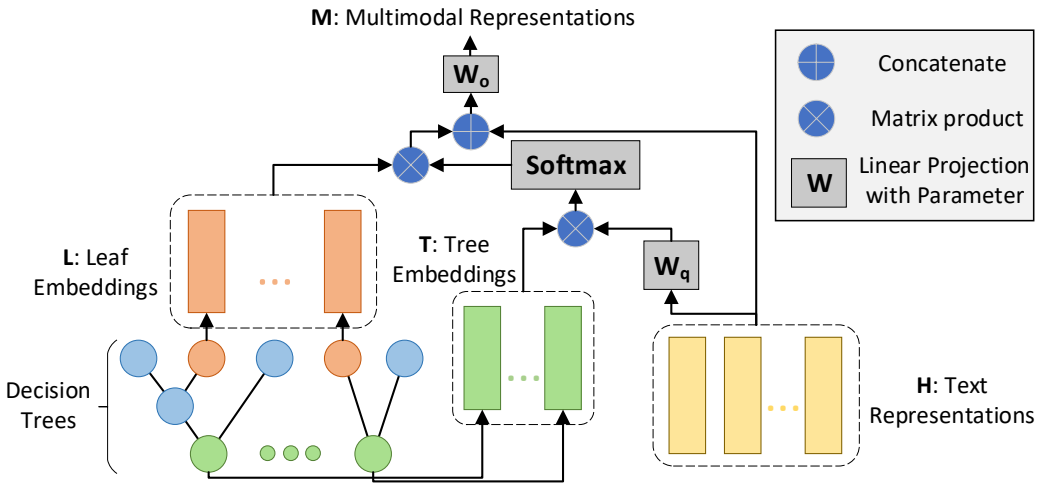

Figure 3: An illustration of our Tree-enhanced Multimodal Attention Network (TreeMAN). Green nodes and orange nodes on the decision trees respectively represent root nodes and activated leaf nodes.

w_i to a d_e-dimensional pre-trained word embedding $\mathbf{e}_i$, then concatenates embeddings into the matrix $\mathbf{E} = [\mathbf{e}_1, \mathbf{e}_2, ..., \mathbf{e}_N]$. To capture contextual information, the word embedding matrix E is fed into a bidirectional LSTM layer to compute the text representations $\mathbf{H}$, which is a concatenation of the forward output and the backward output:

$$\begin{aligned} \overrightarrow{\mathbf{h}_i} &= \overrightarrow{LSTM}(\mathbf{e}_{1:i}), \quad \overleftarrow{\mathbf{h}_i} = \overleftarrow{LSTM}(\mathbf{e}_{i:N}), \\ \mathbf{H} &= [\overrightarrow{\mathbf{h}_1} \oplus \overleftarrow{\mathbf{h}_1}, \overrightarrow{\mathbf{h}_2} \oplus \overleftarrow{\mathbf{h}_2}, ..., \overrightarrow{\mathbf{h}_n} \oplus \overleftarrow{\mathbf{h}_n}]. \end{aligned} \tag{1}$$

Then, text representations $\mathbf{H}$ together with the tree embeddings $\mathbf{T}$ and the leaf embeddings $\mathbf{L}$ generated from tree-based features are fed to TreeMAN to obtain the multimodal representation $\mathbf{M}$ (detailed in Section 3.2):

$$\mathbf{M} = \text{TreeMAN}(\mathbf{H}, \mathbf{L}, \mathbf{T}). \tag{2}$$

In the output layer, following Mullenbach et al. (2018), we apply the per-label attention network to compute representations for each label.

Label attention The label attention network takes multimodal representations $\mathbf{M} \in \mathbb{R}^{d_m \times N}$ as input and compute the per-label representations $\mathbf{V} \in \mathbb{R}^{d_m \times |\mathcal{L}|}$ with a matrix parameter $U \in \mathbb{R}^{d_m \times |\mathcal{L}|}$, where $|\mathcal{L}|$ represents the number of labels:

$$\begin{aligned} \mathbf{A} &= \text{softmax}(\mathbf{MU}), \\ \mathbf{V} &= \mathbf{A}^T\mathbf{M}. \end{aligned} \tag{3}$$

Finally, to compute the probability $\hat{y}_i$ of the ith label, the label representation $\mathbf{v}_i$ of $\mathbf{V}$ is fed into a corresponding linear layer followed by a sigmoid transformation. For training, the model is optimized to minimize the binary cross-entropy loss between the prediction $\hat{y}$ and the target y:

$$Loss = \sum_{i=1}^{|\mathcal{L}|} -y_i log(\hat{y}_i) - (1-y_i) log(1-\hat{y}_i). \tag{4}$$

3.2 TreeMAN

TreeMAN, a tree-enhanced multimodal attention network, is designed to fuse tree-based features and text representations and provides enhanced multimodal representations for multi-label classification. We argue that the critical information in text representations is respective and fragmented because decisive information for different labels in the document is likely contained in different short segments (Mullenbach et al., 2018). Therefore, we use the attention mechanism to learn the relevant features for each text vector and then fuse tree-based features and text information into a unified multimodal representation.

An illustration of TreeMAN is shown in Figure 3. Specifically, for each text vector $\mathbf{h}_i \in \mathbb{R}^{d_h}$ in $\mathbf{H}$, we first project it to a query vector $\mathbf{q}_i$ by a learnable parameter $\mathbf{W}_q \in \mathbb{R}^{d_t \times d_h}$:

$$\mathbf{q}_i = \mathbf{W}_q \mathbf{h}_i. \tag{5}$$

The vector $\mathbf{q}_i \in \mathbb{R}^{d_t}$ is used to generate the attention weight $\boldsymbol{\alpha}_i$ by computing with the tree embeddings $\mathbf{T} \in \mathbb{R}^{d_t \times |\mathcal{T}|}$, where $|\mathcal{T}|$ represents the number of decision trees:

$$\boldsymbol{\alpha}_i = \text{softmax}(\mathbf{T}^T \mathbf{q}_i). \tag{6}$$

The attention vector $\boldsymbol{\alpha}_i$ is then multiplied with the leaf embeddings $\mathbf{L} \in \mathbb{R}^{d_l \times |\mathcal{T}|}$ to produce the special representation $\mathbf{s}_i$ for the text vector:

$$\mathbf{s}_i = \mathbf{L}\boldsymbol{\alpha}_i. \tag{7}$$

To fuse the text information and tree-based features, we concatenate the special representation with text vector and then apply a linear projection:

$$\mathbf{m}_i = \mathbf{W}_o[\mathbf{h}_i||\mathbf{s}_i], \tag{8}$$

where $\mathbf{W}_o \in \mathcal{R}^{d_m \times (d_l + d_s)}$ is a learnable parameter. All the multimodal vectors are concatenated to formulate the output matrix $\mathbf{M} = [\mathbf{m}_1, \mathbf{m}_2, ..., \mathbf{m}_N] \in \mathbb{R}^{d_m \times N}$.

3.3 Construction of Tree-based Features

In this section, we introduce how we construct the tree-based features from structured medical data. Based on the characteristics of the data, we divide the structured medical data into three types: 1) *derived time series data* such as perioperative vital sign signals; 2) *multivalued vertical data* denotes data with multiple records for one admission, such as lab measurements and prescriptions; 3) *single horizontal data* indicates data with only a single record for an admission, such as admission type and patient age. We process different types of data into tabular data in different ways: 1) for *derived time series data*, we compute mean, maximum, and minimum values for each class of data; 2) for *multivalued vertical data*, we convert it into binary vector to indicate whether a test is abnormal or a medication is prescribed; 3) for *single horizontal data*, we directly put it into the table as it is.

Then, we use the processed tabular data to construct decision trees by applying decision trees, which are trained with ICD codes as the target, using one-versus-all strategy for multi-label classification. Formally, we get a set of decision trees, $\mathbb{Q} = \{Q_1, ..., Q_{|\mathcal{T}|}\}$, where each tree maps the tabular data $\mathbf{x}$ to a leaf node, which can be represented by a one-hot vector. The representation of tree-based features is a multi-hot vector $\mathbf{q}$ which is a concatenation of one-hot vectors:

$$\mathbf{q} = [Q_1(x), ..., Q_{|\mathcal{T}|}(x)]. \tag{9}$$

Therefore, there are $|\mathcal{T}|$ elements of value 1 in $\mathbf{q}$ indicates activated leaf nodes.

Inspired by the success of TEM (Wang et al., 2018), we project $\mathbf{q}$ into an embedding matrix $\mathbf{L}$ as leaf embeddings. For the attention computation in Section 3.2, we also generate a tree embedding matrix $\mathbf{T}$ based on the number of decision trees.

	MIMIC-III 50	**MIMIC-II 50**
Vocubulary Size	51,917	30,688
# Samples	11,371	3,726
# *Drugs	2350	52
# *Lab Items	245	217
# *Organism	183	135
# *Specimen	74	63
# *Antibiotic	30	30
# *Chart Items	200	-
Mean # labels per document	5.7	3.4
Mean # tokens per document	1530	1014

Table 1: The statistics of the two MIMIC datasets and the structured medical data used therein, where "#" indicates "the number of" and "*" denotes the number of classes is counted.

4 Experiments

4.1 Datasets

To make a fair and all-round comparison with former SOTA models, we evaluate our model on two widely used **M**edical **I**nformation **M**art for **I**ntensive **C**are (MIMIC) datasets: MIMIC-III (Johnson et al., 2016) and MIMIC-II (Saeed et al., 2002). Because it's hard for our method to be implemented on ICD codes with less than 10 positive samples, we filter out records not relative to the top 50 most frequent ICD codes (denoted as MIMIC-III 50, MIMIC-II 50) to train and evaluate our method.

MIMIC-III 50. Except for structured medical data, we use the same experimental setup including the same splits as previous works (Mullenbach et al., 2018; Cao et al., 2020; Vu et al., 2020). For structured medical data, we use the following tables in MIMIC-III dataset [1] : 1) *Admissions* contains patients' admission information such as admission time; 2) *Patients* contains patients' basic information such as date of birth; 3) *Chartevents* contains charted data including patients' routine vital signs; 4) *Labevents* contains laboratory measurements such as pH of blood; 5) *Microbiologyevents* contains microbiology information such as organism test information; 6) *Prescriptions* contains medications related to order entries including the Generic Sequence Number (GSN) of drugs.

[1] A detailed introduction to MIMIC-III tables can be found at https://mimic.mit.edu/docs/iii/tables.

MIMIC-II 50. We subset the MIMIC-II full used in previous works (Mullenbach et al., 2018) based on the 50 most frequent labels and use a set of 3,726 admission samples in which there are 2980 samples for training, 373 for validation and 373 for test. For structured medical data, we use the following tables in MIMIC-II dataset [2]: 1) *Admissions*; 2) *Patients*; 3) *Medevents* is similar to *Prescriptions* in MIMIC-III; 4) *Labevents*; 5) *Microbiologyevents*.

Basic statistic information of all the datasets shows on Table 1.

4.2 Implementation Details

Following the preprocessing schema of previous works (Mullenbach et al., 2018; Li and Yu, 2020; Xie et al., 2019), we lowercase all tokens and remove tokens that contain unrelated alphabetic characters like numbers and punctuations. We implement the word2vec CBOW (Mikolov et al., 2013) method to pre-train word embeddings and truncate all discharge summary documents to the maximum length of 4,000 tokens. We employ XGBoost [3] to implement the decision trees in our approach. There is only one decision tree built for each label where the learning rate and the maximum depth of the tree are set as 0.99 and 5, respectively, while the rest of settings follow the default. The sizes of the tree embedding **T** and the leaf embedding **L** are 128 and 30, respectively. We set the size of multimodal representations to be the same as that of text representations.

For the baseline methods we reproduced on the MIMIC-II 50 dataset, we used the same implementations used by the authors on MIMIC-III 50. To reduce randomness, we repeated all experiments 5 times with different random seeds and report the average performance.

4.3 Metrics

To compare with previous and potential future work thoroughly, we measured our model mainly on indicators of macro-averaged and micro-averaged F1, macro-averaged, and micro-averaged AUC (area under the ROC curve) and Precision@k (P@k). Among these metrics, the "micro-averaged" method takes every single decision into consideration by pooling all text-code pairing and then calculating an effectiveness indicator on the pooled data. And "macro-averaged based" metrics would provide statistics from the perspective of label instead of pair-relationship. Furthermore, we rank predictive probabilities to compute the precision of the top-k predicted labels, denoted as P@k. We set k to be five on MIMIC-III 50 dataset and three on MIMIC-II 50 dataset for the average discharge summary has 5.7 labels in MIMIC-III 50 while 3.4 in MIMIC-II 50. We believe a full comparison of all the above metrics will provide insight into our work.

4.4 Baselines

We compare our model TreeMAN with the following baseline; all of them were SOTA when they were proposed initially.

CAML **C**onvolutional **A**ttention network for **M**ulti- **L**abel classification (CAML) and **d**escription **R**egularized CAML was proposed by Mullenbach et al. (2018), which combined a single-layer CNN with attention layer to generate ICD coding for given text.

LAAT&Joint-LAAT **LA**bel **AT**tention and Joint **LA**bel **AT**tention model was proposed by Vu et al. (2020). It encodes the input text with BiLSTM layer and implements self-attention mechanism to learn label-specific vectors representation. A hierarchical joint learning architecture is utilized to improve performance in the second model.

HyperCore **Hyper**bolic and **Co**-graph Representation was proposed by Cao et al. (2020). It leveraged hierarchical structure of ICD code in hyperbolic space and used graph convolutional network(GCN) to capture co-occurrence correlation of labels.

ISD **I**nteractive **S**hared Representation Network with Self-**D**istillation Mechanism was proposed by Zhou et al. (2021), they implemented a self-distillation learning mechanism to alleviate the noisy text and only focus on noteworthy part of text.

4.5 Results

Table 2 reports $mean \pm standard\ deviation$ of TreeMAN's results on two datasets, the performance of baselines on MIMIC-III 50 and the results of our implementation of baselines on MIMIC-II 50. Compared with previous text methods, our multimodal approach achieves the best results on all metrics on both datasets. It indicates that our

[2] A detailed introduction to MIMIC-II tables can be found at https://archive.physionet.org/mimic2/UserGuide/UserGuide.pdf .

[3] https://xgboost.readthedocs.io.

Model	MIMIC-III 50					MIMIC-II 50				
	AUC		F1		P@5	AUC		F1		P@3
	macro	micro	macro	micro		macro	micro	macro	micro	
CAML	0.875	0.909	0.532	0.614	0.609	0.871	0.902	0.426	0.553	0.552
HyperCore	0.895	0.929	0.609	0.663	0.632	-	-	-	-	-
LAAT	0.925	0.946	0.666	0.715	0.675	0.874	0.908	0.436	0.557	0.556
Joint LAAT	0.925	0.946	0.661	0.716	0.671	0.875	0.908	0.434	0.547	0.560
ISD	0.935	0.949	0.679	0.717	**0.682**	-	-	-	-	-
TreeMAN	**0.937** ±0.002	**0.953** ±0.000	**0.690** ±0.002	**0.729** ±0.002	**0.682** ±0.001	**0.883** ±0.002	**0.916** ±0.002	**0.479** ±0.001	**0.574** ±0.001	**0.605** ±0.004

Table 2: Results on MIMIC-III 50 dataset, MIMIC-II 50 dataset and $mean \pm standard\ deviation$ of each indicator gained from replicated experiments with random initial states. Baseline scores are from the corresponding papers in Section 4.4.

Model	AUC		F1		P@5
	macro	micro	macro	micro	
text	92.6	94.5	67.4	71.4	66.6
maxpooling	93.1	94.9	68.4	72.3	67.5
average	93.4	95.1	68.9	72.7	67.6
TreeMAN	93.7	95.3	69.0	72.9	68.2

Table 3: Results of ablation experiments on the MIMIC-III 50 dataset (in %).

model benefits from the rich information contained in structured medical data. Furthermore, the small standard deviations demonstrate that the good results our model achieved are stable. We also observe more significant improvements in the f1-marco and f1-micro metrics compared to other ranking-based metrics. Since the binary output is produced by a fixed threshold 0.5, a possible reason for the disparity is that the sigmoid function of our model in the final layer outputs more dispersed probabilities due to the decisive information provided by structured medical data.

4.6 Ablation Experiment

To testify the effectiveness of the different modules in TreeMAN, we perform a series of ablation experiments on the MIMIC-III 50 dataset, design following experiments, and report the results in Table 3.

The Effect of Structured Medical Data To study the effectiveness of the information captured from structured medical data, we remove the tree-based features in TreeMAN and directly feed the unfused text representations to the multi-label classifier (*text* in Table 3). The experimental results of all metrics decreased significantly compared to TreeMAN, demonstrating the importance of the tree-based features constructed based on structured medical data. It's also a comparison between the text representations and the multimodal representations, which proves that TreeMAN is capable of learning multimodal features.

The Effect of Attention Mechanism To examine the effectiveness of the attention mechanism in TreeMAN, we design two experiments by replacing the attention network with the max-pooling layer (*maxpooling* in Table 3) and the average layer (*average* in Table 3) on leaf embeddings. Formally, we change the Equation 7 as:

$$\begin{cases} \text{maxpooling: } \mathbf{s}_i = max\_pool_{\mathbf{l}\in\mathbf{L}}(\mathbf{l}), \\ \text{average: } \mathbf{s}_i = \frac{1}{|\mathcal{T}|}\sum_{\mathbf{l}\in\mathbf{L}}(\mathbf{l}), \end{cases} \tag{10}$$

where $\mathbf{l}$ and $\mathcal{T}$ represent a vector in the leaf embeddings $\mathbf{L}$ and the number of decision trees, respectively. As shown, the experimental results of *maxpooling* and *average* are both better than *text* and worse than TreeMAN. Thus, the attention mechanism improves TreeMAN's ability to learn multimodal information and the information captured by tree-based features is robust to learn.

4.7 Parameter Studies

We have already analyzed the efficacy of our proposed model, and now we want to conduct a series of experiments to test the effect of two critical hyper-parameters in the TreeMAN module: the maximum depth of the decision tree and the size of leaf embeddings $\mathbf{L}$. The former decides how tree-based features are constructed from structured medical data, and the latter is the representation format of the tree-based features. Various metrics of different settings would help us to demonstrate how

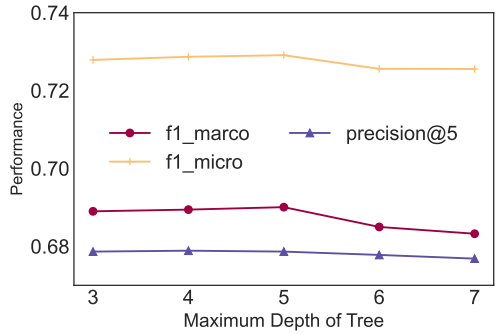

(a) Performance of TreeMAN with different maximum Depth of Decision Tree

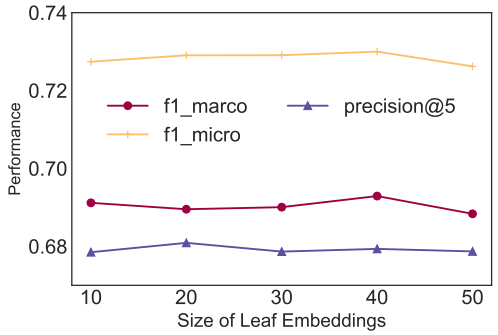

(b) Performance of TreeMAN with different Leaf Embedding Dimensions

Figure 4: Results of different maximum tree depth and leaf embedding size on MIMIC-III 50.

TreeMAN extracts information from multimodal data:

The Effect of Maximum Depth of the Decision Trees The maximum depth of the decision tree would decide the number of feature extracted and the properties of the leaf embedding layer, for each of the leaf nodes represents a tree-based feature. For example, if we set the maximum depth to 3, we would get 401 leaves and 4752 leaves for a 7-layer tree. A shallow decision tree cannot extract enough features to represent the latent information of initial input. However, a too deep tree would risk over-fitting as well as colossal costs in the training process. Based on this assumption, we make a complete comparison of different pre-set depths of the decision tree. As the Figure 4 (a) shows, a tree of depth 5 outperforms other decision trees, especially on the indicator of f1_marco and f1_micro because of the improvement in the aspect of recall ratio. Furthermore, we also notice that changes in this hyper-parameter don't seriously affect the performance of our module, proving the robustness of our method.

The Effect of Leaf Embeddings As we project multimodal information gained in the decision tree to leaf embeddings $\mathbf{L}$, we need the proper capacity of this layer to collect and store them. Thus we experiment with the leaf embedding size ranging from 10 to 50 to study the effect of the setup. Figure 4(b) shows that a vector with 30 dimensions is a proper choice because short vectors would abandon helpful information, while long ones would carry redundant information. Taking note of the limited size of datasets, relatively simple architecture could be a practical solution. These results also indicate that TreeMAN has learned an operative and steady pattern to learn from various types of multimodal information.

5 Conclusion

In this paper, we proposed a tree-based multimodal method for the ICD coding task, which constructs tree-based features by decision trees learned from structured medical data and fuses the tree-based features and text representation by a novel tree-enhanced multimodal attention network (TreeMAN). Experimental results on two MIMIC datasets show that our method outperforms state-of-the-art methods. Further ablation studies demonstrate that structured medical data and the attention mechanism in TreeMAN have improved the performance.

For future work, we plan to investigate the interpretability of our method since tree-based methods are naturally interpretable. We are also interested in exploring a generalized and robust way to construct the tree-based features to capture more generalized medical information from structured medical data.

6 Acknowledgement

This research is supported by Chinese Scientific and Technical Innovation Project 2030 (No. 2018AAA0102100), National Natural Science Foundation of China (No. U1936206, 62077031, 62272250). We appreciate the reviewers for their constructive comments. We thank Shuyun Deng for her help in writing the paper.

References

Robert E. Banfield, Lawrence O. Hall, Kevin W. Bowyer, and W. Philip Kegelmeyer. 2007. A comparison of decision tree ensemble creation techniques. *IEEE Trans. Pattern Anal. Mach. Intell.*, 29(1):173–180.

Qiong Cai, Hao Wang, Zhenmin Li, and Xiao Liu. 2019. A survey on multimodal data-driven smart healthcare systems: Approaches and applications. *IEEE Access*, 7:133583–133599.

Pengfei Cao, Yubo Chen, Kang Liu, Jun Zhao, Shengping Liu, and Weifeng Chong. 2020. Hypercore: Hyperbolic and co-graph representation for automatic ICD coding. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, ACL 2020, Online, July 5-10, 2020*, pages 3105–3114. Association for Computational Linguistics.

Ilias Chalkidis, Manos Fergadiotis, Sotiris Kotitsas, Prodromos Malakasiotis, Nikolaos Aletras, and Ion Androutsopoulos. 2020. An empirical study on large-scale multi-label text classification including few and zero-shot labels. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing, EMNLP 2020, Online, November 16-20, 2020*, pages 7503–7515. Association for Computational Linguistics.

Tianqi Chen and Carlos Guestrin. 2016. Xgboost: A scalable tree boosting system. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '16, page 785–794, New York, NY, USA. Association for Computing Machinery.

José Carlos Ferrão, Mónica Duarte Oliveira, Filipe Janela, Henrique MG Martins, and Daniel Gartner. 2021. Can structured ehr data support clinical coding? a data mining approach. *Health Systems*, 10(2):138–161.

Jerome H. Friedman. 2001. Greedy function approximation: A gradient boosting machine. *The Annals of Statistics*, 29(5):1189 – 1232.

Mike Gashler, Christophe Giraud-Carrier, and Tony Martinez. 2008. Decision tree ensemble: Small heterogeneous is better than large homogeneous. In *2008 Seventh International Conference on Machine Learning and Applications*, pages 900–905.

Xinran He, Junfeng Pan, Ou Jin, Tianbing Xu, Bo Liu, Tao Xu, Yanxin Shi, Antoine Atallah, Ralf Herbrich, Stuart Bowers, and Joaquin Quiñonero Candela. 2014. Practical lessons from predicting clicks on ads at facebook. In *Proceedings of the Eighth International Workshop on Data Mining for Online Advertising, ADKDD 2014, August 24, 2014, New York City, New York, USA*, pages 5:1–5:9. ACM.

Alistair E.W. Johnson, Tom J. Pollard, Lu Shen, Li-wei H. Lehman, Mengling Feng, Mohammad Ghassemi, Benjamin Moody, Peter Szolovits, Leo Anthony Celi, and Roger G. Mark. 2016. Mimic-iii, a freely accessible critical care database. *Scientific Data*, 3(1):160035.

Guolin Ke, Qi Meng, Thomas Finley, Taifeng Wang, Wei Chen, Weidong Ma, Qiwei Ye, and Tie-Yan Liu. 2017. Lightgbm: A highly efficient gradient boosting decision tree. In *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc.

Sundong Kim, Yu-Che Tsai, Karandeep Singh, Yeonsoo Choi, Etim Ibok, Cheng-Te Li, and Meeyoung Cha. 2020. Date: Dual attentive tree-aware embedding for customs fraud detection. KDD '20, page 2880–2890, New York, NY, USA. Association for Computing Machinery.

Leah S. Larkey and W. Bruce Croft. 1996. Combining classifiers in text categorization. In *Proceedings of the 19th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, SIGIR'96, August 18-22, 1996, Zurich, Switzerland (Special Issue of the SIGIR Forum)*, pages 289–297. ACM.

Fei Li and Hong Yu. 2020. ICD coding from clinical text using multi-filter residual convolutional neural network. In *The Thirty-Fourth AAAI Conference on Artificial Intelligence, AAAI 2020, The Thirty-Second Innovative Applications of Artificial Intelligence Conference, IAAI 2020, The Tenth AAAI Symposium on Educational Advances in Artificial Intelligence, EAAI 2020, New York, NY, USA, February 7-12, 2020*, pages 8180–8187. AAAI Press.

Xiaoliang Ling, Weiwei Deng, Chen Gu, Hucheng Zhou, Cui Li, and Feng Sun. 2017. Model ensemble for click prediction in bing search ads. In *Proceedings of the 26th International Conference on World Wide Web Companion*, WWW '17 Companion, page 689–698, Republic and Canton of Geneva, CHE. International World Wide Web Conferences Steering Committee.

Tomás Mikolov, Ilya Sutskever, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013. Distributed representations of words and phrases and their compositionality. *CoRR*, abs/1310.4546.

James Mullenbach, Sarah Wiegreffe, Jon Duke, Jimeng Sun, and Jacob Eisenstein. 2018. Explainable prediction of medical codes from clinical text. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT 2018, New Orleans, Louisiana, USA, June 1-6, 2018, Volume 1 (Long Papers)*, pages 1101–1111. Association for Computational Linguistics.

Anthony N. Nguyen, Donna L. Truran, Madonna Kemp, Bevan Koopman, David Conlan, John O'Dwyer, Ming Zhang, Sarvnaz Karimi, Hamed Hassanzadeh, Michael Lawley, and Damian J. Green. 2018. Computer-assisted diagnostic coding: Effectiveness of an nlp-based approach using SNOMED CT to

ICD-10 mappings. In *AMIA 2018, American Medical Informatics Association Annual Symposium, San Francisco, CA, November 3-7, 2018*. AMIA.

Kimberly J O'malley, Karon F Cook, Matt D Price, Kimberly Raiford Wildes, John F Hurdle, and Carol M Ashton. 2005. Measuring diagnoses: Icd code accuracy. *Health services research*, 40(5p2):1620–1639.

Jong-Ku Park, Ki-Soon Kim, Tae-Yong Lee, Kang-Sook Lee, Duk-Hee Lee, Sun-Hee Lee, Sun-Ha Jee, Il Suh, Kwang-Wook Koh, So-Yeon Ryu, et al. 2000. The accuracy of icd codes for cerebrovascular diseases in medical insurance claims. *Journal of Preventive Medicine and Public Health*, 33(1):76–82.

Adler Perotte, Rimma Pivovarov, Karthik Natarajan, Nicole Weiskopf, Frank Wood, and Noémie Elhadad. 2014. Diagnosis code assignment: models and evaluation metrics. *Journal of the American Medical Informatics Association*, 21(2):231–237.

J. R. Quinlan. 1986. Induction of decision trees. *Mach. Learn.*, 1(1):81–106.

Pavithra Rajendran, Alexandros Zenonos, Joshua Spear, and Rebecca Pope. 2021. Embed wisely: An ensemble approach to predict ICD coding. In *Machine Learning and Principles and Practice of Knowledge Discovery in Databases - International Workshops of ECML PKDD 2021, Virtual Event, September 13-17, 2021, Proceedings, Part II*, volume 1525 of *Communications in Computer and Information Science*, pages 371–389. Springer.

M. Saeed, C. Lieu, G. Raber, and R.G. Mark. 2002. Mimic ii: a massive temporal icu patient database to support research in intelligent patient monitoring. In *Computers in Cardiology*, pages 641–644.

S. Rasoul Safavian and David A. Landgrebe. 1991. A survey of decision tree classifier methodology. *IEEE Trans. Syst. Man Cybern.*, 21(3):660–674.

Yiyi Tao, Yiling Jia, Nan Wang, and Hongning Wang. 2019. The fact: Taming latent factor models for explainability with factorization trees. In *Proceedings of the 42nd International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR'19, page 295–304, New York, NY, USA. Association for Computing Machinery.

Fei Teng, Yiming Liu, Tianrui Li, Yi Zhang, Shuangqing Li, and Yue Zhao. 2022. A review on deep neural networks for icd coding. *IEEE Transactions on Knowledge and Data Engineering*.

Ilya Trofimov, Anna Kornetova, and Valery Topinskiy. 2012. Using boosted trees for click-through rate prediction for sponsored search. In *Proceedings of the Sixth International Workshop on Data Mining for Online Advertising and Internet Economy*, ADKDD '12, New York, NY, USA. Association for Computing Machinery.

Thanh Vu, Dat Quoc Nguyen, and Anthony Nguyen. 2020. A label attention model for ICD coding from clinical text. In *Proceedings of the Twenty-Ninth International Joint Conference on Artificial Intelligence, IJCAI 2020*, pages 3335–3341. ijcai.org.

Xiang Wang, Xiangnan He, Fuli Feng, Liqiang Nie, and Tat-Seng Chua. 2018. TEM: tree-enhanced embedding model for explainable recommendation. In *Proceedings of the 2018 World Wide Web Conference on World Wide Web, WWW 2018, Lyon, France, April 23-27, 2018*, pages 1543–1552. ACM.

Xiancheng Xie, Yun Xiong, Philip S. Yu, and Yangyong Zhu. 2019. EHR coding with multi-scale feature attention and structured knowledge graph propagation. In *Proceedings of the 28th ACM International Conference on Information and Knowledge Management, CIKM 2019, Beijing, China, November 3-7, 2019*, pages 649–658. ACM.

Keyang Xu, Mike Lam, Jingzhi Pang, Xin Gao, Charlotte Band, Piyush Mathur, Frank Papay, Ashish K. Khanna, Jacek B. Cywinski, Kamal Maheshwari, Pengtao Xie, and Eric P. Xing. 2019. Multimodal machine learning for automated ICD coding. In *Proceedings of the Machine Learning for Healthcare Conference, MLHC 2019, 9-10 August 2019, Ann Arbor, Michigan, USA*, volume 106 of *Proceedings of Machine Learning Research*, pages 197–215. PMLR.

Zachariah Zhang, Jingshu Liu, and Narges Razavian. 2020. BERT-XML: large scale automated ICD coding using BERT pretraining. In *Proceedings of the 3rd Clinical Natural Language Processing Workshop, ClinicalNLP@EMNLP 2020, Online, November 19, 2020*, pages 24–34. Association for Computational Linguistics.

Ke Zhou, Shuang-Hong Yang, and Hongyuan Zha. 2011. Functional matrix factorizations for cold-start recommendation. In *Proceedings of the 34th International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '11, page 315–324, New York, NY, USA. Association for Computing Machinery.

Tong Zhou, Pengfei Cao, Yubo Chen, Kang Liu, Jun Zhao, Kun Niu, Weifeng Chong, and Shengping Liu. 2021. Automatic ICD coding via interactive shared representation networks with self-distillation mechanism. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing, ACL/IJCNLP 2021, (Volume 1: Long Papers), Virtual Event, August 1-6, 2021*, pages 5948–5957. Association for Computational Linguistics.

Gated Mechanism Enhanced Multi-Task Learning for Dialog Routing

Ziming Huang[1]*, **Zhuoxuan Jiang**[1]*, **Ke Wang**[2]*, **Juntao Li**[3], **Shanshan Feng**[4], **Xian-Ling Mao**[5]
[1]Tencent, [2]Alibaba DAMO Academy, [3]Soochow University,
[4]Harbin Institute of Technology(Shenzhen), [5]Beijing Institute of Technology
hzmyouxiang@gmail.com, jzhx@pku.edu.cn, wk258730@alibaba-inc.com,
ljt@suda.edu.cn, victor_fengss@foxmail.com, maoxl@bit.edu.cn

Abstract

Currently, human-bot symbiosis dialog systems, e.g., pre- and after-sales in E-commerce, are ubiquitous, and the dialog routing component is essential to improve the overall efficiency, reduce human resource cost, and enhance user experience. Although most existing methods can fulfil this requirement, they can only model single-source dialog data and cannot effectively capture the underlying knowledge of relations among data and subtasks. In this paper, we investigate this important problem by thoroughly mining both the data-to-task and task-to-task knowledge among various kinds of dialog data. To achieve the above targets, we propose a Gated Mechanism enhanced Multi-task Model (G3M), specifically including a novel dialog encoder and two tailored gated mechanism modules. The proposed method can play the role of hierarchical information filtering and is non-invasive to existing dialog systems. Based on two datasets collected from real world applications, extensive experimental results demonstrate the effectiveness of our method, which achieves the state-of-the-art performance by improving 8.7%/11.8% on RMSE metric and 2.2%/4.4% on F1 metric.

1 Introduction

Traditionally, a lot of human resource cost is spent on supporting the calling/online centers for customer care, such as pre-and after-sales for E-commerce and banking. With the rapid development of AI techniques for dialog systems, various bot agents have been deployed in those scenarios to reduce parts of human workload. Both the bot and human agents constitute a new and practical symbiosis dialog system which can keep a balance between service quality and human resource cost (Oraby et al., 2017).

In such a human-bot symbiosis environment, it is commonly seen that users' needs continuously change, or sometimes they are dissatisfactory with the current servicing agents. They would request the dialog system to replace the agents with more qualified ones, because usually each bot or human agent is trained for specific domains. For example, some agents are good at handling after-sales complaints, while some others would do well in providing technical guidance. To make the whole system operate efficiently, a dialog routing component is necessary to take charge of when to transfer the conversation and which domain of agents is suitable to provide service.

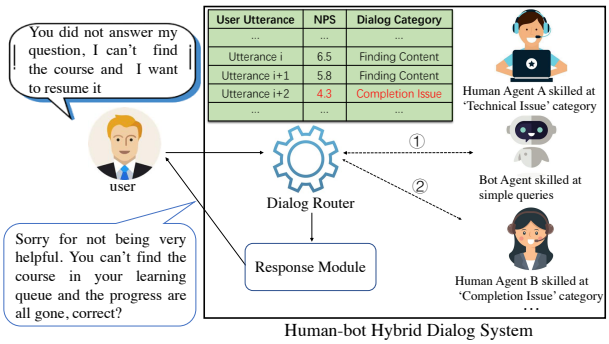

Figure 1: An example of dialog routing in human-bot symbiosis dialog system.

To illustrate the process of dialog routing task more clearly, Figure 1 shows an example. At the beginning, the user's query is operated by the bot agent through Flow ①, and the dialog router would continuously monitor the user's satisfactory degree by inferring Net Promoter Score (NPS) and current dialog category. When the inferred NPS (4.8) is lower than a hand-crafted threshold (5) at the ith dialog turn, the router chooses to switch the dialog to one of human agents skilled at the inferred category via Flow ②. Here human agent B is good at the category of 'Completion Issue' dialogs. From this picture, we could realize that how to implement the accurate matching between user's needs and skilled agents is greatly correlated with the whole system's efficiency and user experience.

Currently, most existing deployed systems are

*This work was done when they were in IBM Research China, and Zhuoxuan Jiang is the corresponding author.

Proceedings of the 29th International Conference on Computational Linguistics, pages 3064–3073
October 12–17, 2022.

heuristic and support to trigger the dialog routing when users actively click a button or say/type something like 'transfer to human'. This method is less user-friendly and cannot fully leverage the ability of contemporary deep leaning techniques. Recently, some learning based methods are proposed. Yu et al. well defines the dialog routing as text regression and classification problems, and proposes a learning network with CNN and RNN modules to encode dialog data (Yu et al., 2020). However, it only models the token-level utterance data and misses the information from other kinds of dialog data[1]; Meanwhile the model structure is shared by the two tasks (in single-task way) which cannot capture the underlying knowledge within data-to-task (intrinsic) and task-to-task (extrinsic) relations. In this paper, we investigate the dialog routing task from the multi-task learning perspective to further capture the intrinsic and extrinsic information.

Intuitively, the decision making process of dialog router could be decomposed to two subtasks, NPS prediction and dialog category classification. We adopt the same NPS prediction task as (Yu et al., 2020) since the supervised data can be obtained without any human-labeling effort. Furthermore, we observe that the data-to-task (intrinsic) and task-to-task (extrinsic) information should be leveraged to make better routing performance. On the one hand, we can argue that the more accurate NPS is predicted, the more precise the dialog category is classified, and vice verse. On the other hand, in addition to the token-level utterance data, other kinds of dialog data can also affect the final task performance. For example, if a user utterance is about the intent of 'ask_technical_problem', the intent information is an obvious indicator that the user should be served by the agent who is skilled at 'Technical Issue' category. Note that the categories of agents and dialog intents of users are often different, and usually the former is much less than the later.

To achieve the above motivation, in this paper, we propose a novel Gated Mechanism enhanced Multi-task Model (G3M). Firstly, the model extends the BERT encoder to encode various kinds of dialog data in a hierarchical way. Moreover, two modules of gated mechanism are proposed to explicitly model the data-to-task and task-to-task information under multi-task learning framework. Another advantage of G3M is its good compatibility, such that it can be easily integrated into existing human-bot dialog systems in a plug-in manner. We conduct various experiments on two datasets collected from the real world. The results demonstrate our method can achieve the state-of-the-art performance on both tasks, and the ablation experiment proves our model is effective to simultaneously capture the intrinsic and extrinsic information.

In summary, this paper's contributions include:

- We argue that both the data-to-task and task-to-task information are important to achieve better dialog routing. Hence, we propose a new multi-task learning solution, called Gated Mechanism enhanced Multi-task Model (G3M), to implement the motivation.
- We extend the BERT encoder to encode various kinds of dialog data in a hierarchical manner, and develop two modules of gated mechanism to explicitly model the data-to-task and task-to-task information.
- We conduct extensive experiments on two real world datasets. The results prove our model's effectiveness, which can achieve the state-of-the-art performance.

2 Related Work

Compared with other dialog system tasks, there are not many related work on dialog routing study. Recently, a learning network with attention based CNN and RNN is proposed (Yu et al., 2020). It mainly models the token-level dialog data and is a single-task model, i.e., the NPS regression and dialog classification tasks share the same model structure except the last inference layer. This method cannot sufficiently leverage the underlying knowledge within subtasks and various kinds of dialog data. Another similar work is a demo system for dialog transition (Huang et al., 2021) which tries a vanilla multi-task learning method (i.e., a dialog encoder tailed with two prediction subtasks). However, it still hardly fully utilizes the knowledge among dialog data and subtasks. Different from them, we further investigate the problem by simultaneously modeling data-to-task and task-to-task information, achieving a much advanced performance.

The dialog routing task is fundamentally decomposed to two subtasks, dialog NPS regres-

[1]Other kinds of dialog data include speaker roles, utterance-level sequence, intent information, etc.

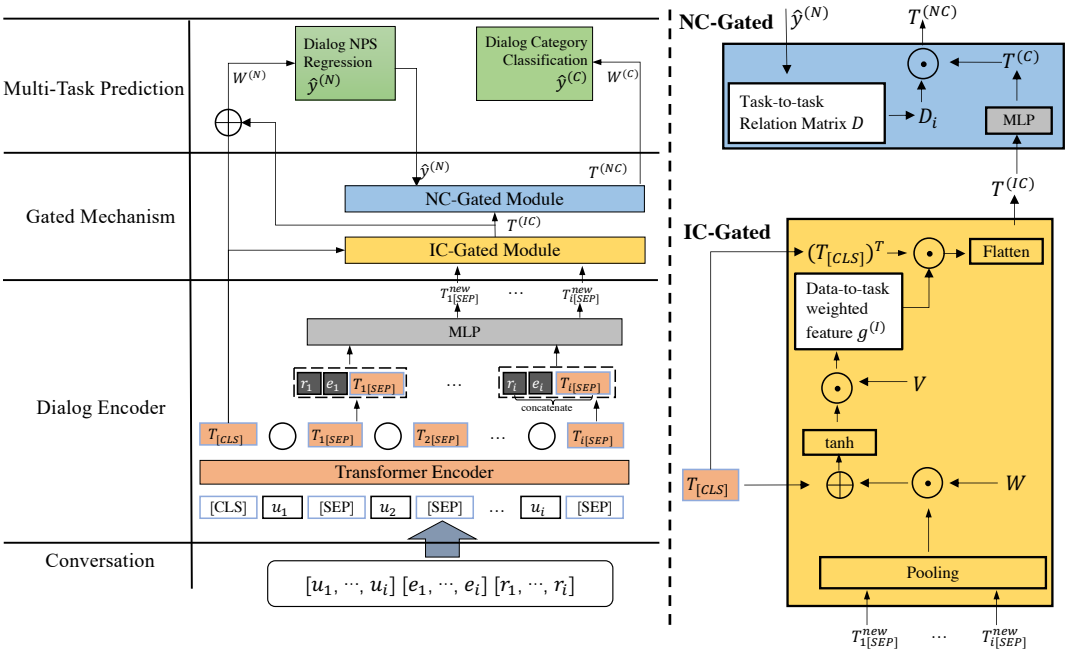

Figure 2: Architecture overview of Gated Mechanism enhanced Multi-task Model with zoomed in NC-Gated (right top blue part) and IC-Gated (right down yellow part) modules.

sion and dialog classification, which feature encoding various kinds of dialog data and have been separately studied by researchers from the NLP community. The representative research studies include CNN-based models (Kim, 2014), RNN-based models (Wang et al., 2018), regression-based models (Dereli and Saraçlar, 2019; Ngo-Ye and Sinha, 2014) and deep bi-directional transformers model with pre-training (Devlin et al., 2019; Cohan et al., 2019). From these works, we learn that dialog data encoder is the common key module to make sure good performance on final tasks (Dereli and Saraçlar, 2019). Those methods have not been demonstrated to be effective in the multi-task situation of dialog routing, without thoroughly mining the relations among data and subtasks.

Recently, multi-task learning on dialog data has been studied and proven successful, such as joint slot filling and intent prediction (Goo et al., 2018), dialog act sequence labeling (Kumar et al., 2018), and dialog response generation (Ide and Kawahara, 2021). These studies suggest that dialog data contains rich information and multi-task learning could dig out the underlying knowledge among data and subtasks. Moreover, among multi-task learning methods, multi-gated mechanism is widely utilized for information remembering and filtering (Chen et al., 2018; Xiao et al., 2018; Du et al., 2019), which is much suitable for tidying dialog data. Inspired by the above ideas, we propose a novel model to combine them together in our solution for further improving the performance of dialog routing task.

3 Problem Definition

We define the investigated problem formally in this section. In our task, we use the available data in a conversation session as inputs, including an utterance list $U = \{u_1, ..., u_L\}$, a speaker role list $R = \{r_1, ..., r_L\}$, and an intent list $I = \{e_1, ..., e_L\}$, where u_i is an utterance, r_i indicates the role of the ith utterance (user or agent), e_i is known intent of the ith utterance, and L is the number of utterance sequences in a conversation session. Each utterance has a token-level sequence $u_i = \{w_{i1}, ..., w_{iM}\}$, where w_{ij} means the jth token in the ith utterance and M is the length of the token sequence. Here we can leverage the intent information since we use an intent extraction tool[2]. Our model is also scalable to capture more dialog data, such as entity and dialog act information if they are available. We adopt the intent information in our solution based on the intuitive consideration of data-to-task motivation.

For the outputs of our task, at each dialog turn i, our model can make predictions of NPS $y_i^{(N)}$ and dialog category $y_i^{(C)}$. In summary, given $U_i = \{u_1, ..., u_i\}$, $R_i = \{r_1, ..., r_i\}$ and $I_i = \{e_1, ..., e_i\}$, we formally state the problem as follows:

[2]More information can be referred in Section Experiments.

$$y_i^{(N)}, y_i^{(C)} = f(U_i, R_i, I_i). \quad (1)$$

4 Proposed Method

The proposed model generally contains three parts: (1) Dialog Encoder, (2) Gated Mechanism, and (3) Multi-Task Prediction. Figure 2 shows the overview of model's architecture, which will be introduced next.

4.1 Dialog Encoder

To encode the various kinds of dialog data, we leverage and extend the BERT encoder. As presented in Figure 2, given an utterance sequence $\{u_1, ..., u_i\}$, firstly all the utterances are flattened and concatenated to a long token sequence. Then we add a [CLS] token at the beginning and insert [SEP] tokens between any two utterances. Similar to BERT encoder that embeds the whole token sequence by Transformer structure, we get the representation of the ith [SEP] token, $T_{i[SEP]}$, and regard it as the representation of the ith utterance. The representation of head token, $T_{[CLS]}$, would be used in the following Gated Mechanism part as a kind of context information which encodes all the utterances.

With the utterance representations of $T_{i[SEP]}$, the role $\{r_1, ..., r_i\}$ and intent $\{e_i, ..., e_i\}$ information (one-hot vectors) are concatenated to their corresponding utterance representations. After a MLP operation, we can obtain the final representation of each utterance by $T^{new}_{i[SEP]} = MLP(r_i \oplus e_i \oplus T_{i[SEP]})$.

Our method to encode dialog data is inspired by the Sequential Sentence Classification (SCC) model (Cohan et al., 2019) which also is based on BERT encoder and organizes the dialog data in the hierarchical manner. However, our model is different from SCC model in two aspects. Firstly, the representation of head token [CLS] is additionally utilized as context information in the following modules. Secondly, our encoder can integrate the extra role and intent information.

4.2 Gated Mechanism

Besides the basic mechanism via sharing parameters of the encoder in vanilla multi-task learning paradigm, we leverage gated mechanism and propose two modules for dialog routing to better model the task-to-task and data-to-task information.

4.2.1 IC-Gated Module

The **I**ntent-**C**ategory-Gated (IC-Gated) module is designed to model the relation between intent, role and dialog category (data-to-task) information. The right down yellow part in Figure 2 illustrates the detailed module structure.

With the context representation $T_{[CLS]}$ and utterance representations $T^{new}_{i[SEP]}$, we extend a commonly used way (Goo et al., 2018) to capture the relations between intent, role and dialog context by learning a weighted feature for data-to-task (i.e., intent, role to dialog category) modeling. Thus we obtain the weighted feature from various levels of the input dialog data after max pooling operation on the utterance sequence by the following equation:

$$\begin{aligned} g^{(I)} = V \cdot \tanh(&T_{[CLS]} \\ &+ W \cdot Pooling(T^{new}_{1[SEP]}, ..., T^{new}_{i[SEP]})), \end{aligned} \quad (2)$$

where V and W denote the parameters to learn.

With the weighted feature $g^{(I)}$ that attends intent and role information on dialog context, the context information $T_{[CLS]}$ is used again to calculate the final weighted representation of dialog data with a flatten operation as the following equation[3]:

$$T^{(IC)} = Flatten(g^{(I)} \cdot (T_{[CLS]})^{\mathrm{T}}). \quad (3)$$

The IC-Gated module can learn the representation from dialog data and meanwhile consider the related information between intent, role and dialog context, which would benefit the later dialog category classification task.

4.2.2 NC-Gated Module

The **N**PS-**C**ategory-Gated (NC-Gated) module is designed to further model the underlying relation between subtasks (task-to-task). The right top blue part in Figure 2 shows the structure detail.

We set a task-to-task relation matrix $D \in \mathbb{R}^{d^N \times d^C}$ to preserve the empirical distribution of co-occurrence between NPS and dialog categories among dataset with min-max normalization (Singh et al., 2015), where d^N and d^C are the numbers of NPS intervals and dialog categories respectively. We partition the NPS ranging from 0 to 10 into ten intervals. Then with the representation from IC-Gated module $T^{(IC)}$, the output representation of NC-Gated module is calculated by combining a MLP layer and the task-to-task relation matrix D:

[3] $Flatten(\cdot)$ function reshapes the input matrix into a one-dimension vector.

$$T^{(NC)} = D_i \odot MLP(T^{(IC)}), \tag{4}$$

where $\odot$ is Hadamard product to measure the similarity. D_i is the ith row vector which means a representation of the ith NPS interval corresponding to every dialog category, and it depends on the predicted NPS. The reason that uses NPS to query the matrix instead of using predicted dialog category to query NPS, is that we empirically find the effect from the NPS regression task is more significant than the dialog category classification task.

Note that except for the multi-task learning paradigm that can capture the relation between tasks via parameter sharing, we use the relation matrix D as prior knowledge (or attention weights) and design the NC-Gated module for better representation learning of dialog data by further capturing the task-to-task information.

4.3 Multi-Task Prediction

Dialog routing has two subtasks, namely NPS regression and dialog category classification. For the former task, the enhanced representation $T^{(IC)}$ learned by the IC-Gated module from data-to-task information and the dialog context representation $T_{[CLS]}$ are concatenated together to make the prediction of NPS. The predicted NPS $\hat{y}^{(N)}$ can be defined as:

$$\hat{y}^{(N)} = W^{(N)}(T_{[CLS]} \oplus T^{(IC)}) + b^{(N)}. \tag{5}$$

$W^{(N)}$ and $b^{(N)}$ are parameters to learn.

The dialog classification task uses the enhanced representation $T^{(NC)}$ as input, which is learned by the NC-Gated module from task-to-task information. The predicted dialog category $\hat{y}^{(C)}$ is calculated by a softmax function:

$$\hat{y}^{(C)} = Softmax(W^{(C)}T^{(NC)} + b^{(C)}). \tag{6}$$

$W^{(C)}$ and $b^{(C)}$ are parameters to learn.

4.4 Model Training

To jointly train our model from the two tasks together under the multi-task learning paradigm, we design a loss function by combining a mean squared loss for NPS regression and a cross-entropy loss for dialog classification. To avoid over-fitting, L_2 regularization terms are adopted. The overall objective function is defined as:

$$\mathcal{L} = \underbrace{\frac{\alpha}{N}\sum_{k=1}^{N}[\hat{y}_k^{(N)} - y_k^{(N)}]^2}_{\text{NPS regression task}} - \underbrace{\frac{\beta}{N}\sum_{k=1}^{N}\sum_{y} y_k^{(C)} \log(\hat{y}_k^{(C)})}_{\text{dialog classification task}} + \underbrace{\frac{\gamma}{2}(\|W^{(N)}\|_2^2 + \|W^{(C)}\|_2^2)}_{\text{regularization terms}}, \tag{7}$$

where N is the number of all training samples, $\hat{y}_k^{(N)}$ and $\hat{y}_k^{(C)}$ are the inferred NPS and category labels, and $y_k^{(N)}$ and $y_k^{(C)}$ are the ground truth. α, β and γ are hyper-parameters to balance the weights of terms.

5 Experiments

5.1 Corpus

To our knowledge, there is no public corpus contains both the NPS and category labels, we collect two corpora from the real-world dialog systems to evaluate our method.

Learning Corpus is collected from an online education platform, where the dialog system for customer care is constituted of 7 dialog categories with all human agents: Badge Issue, Completion Issue, Finding Content, System Help, Technical Issue, Ticket Status, and Other.

MacHelp Corpus is collected from an after-sale technical support platform for users who have troubles with their Mac laptops, where the dialog system is constituted of 4 dialog categories for human agents: Login Problem, Apps Issue, Raise Tickets, Account Issue.

5.2 Corpus Preprocessing

For the two raw corpora, we preprocess them into a consistent format for model training and testing. Table 1 shows an example of the data structure of dialog session. In a dialog session, we have an utterance sequence, a role sequence to indicate the speaker role of each utterance (0 means user and 1 means agent), and an intent sequence labeled by using an online tool which is separately trained from supervised information for the domain of customer care[4]. We use the off-the-shelf extraction tool and obtain 66 and 28 types of intents on the two corpora respectively.

Within a dialog session, we partition the data to several training and testing samples. At the

[4]https://www.ibm.com/products/watson-assistant. The intent detection accuracy is about 92% in additional assessment.

role	utterance	intent
1 (agent)	Hello, how may I assist you?	greeting
0 (user)	Hello	greeting
0 (user)	I am looking some training which will help me to learn more about Github.	request
1 (agent)	In Your Learning webpage, you have a search field, you can type what you want to learn about, and it will bring you all courses related to that topic.	guidance
User NPS: 9.0 **Dialog Category**: Finding Content		

Table 1: Data format example of a dialog session.

ith dialog turn, a sample is defined as $S_i = \{\{u_1, ..., u_i\}, \{r_1, ..., r_i\}, \{e_1, ..., e_i\}, y_i^{(N)}, y_i^{(C)}\}$. Ground truths are obtained by the following rules:

- The NPS of u_{i+1} is regarded as the label (i.e., $y_i^{(N)}$) for NPS regression task.
- The agent category of u_{i+1} is regarded as the label (i.e., $y_i^{(C)}$) for dialog classification task.
- If there is no feedback provided by user on the session, our task degrades to gated-mechanism single-task dialog classification.

Note that although the ground truths of NPS and category label are naturally obtained from the users' feedback and human agent category, to guarantee the data quality, three humans are invited to review the correctness of all the labels, and we only adopt the samples agreed by all of them. As a result, we obtain about 90% valid sessions from the corpora. Table 2 shows the statistic information of the final two processed datasets.

The real NPS is from user's feedback by clicking or typing a score, and the two corpora have different NPS ranges (one is an integer in [0,10] and the other is a decimal in [0,5]). Therefore we use zero-mean normalization (z-score) (Singh et al., 2015) method to normalize the scores with a mean of 0 and a standard deviation of 1 as the following: z-score $= \frac{x_i - \overline{x}}{\sigma}$, where x_i is NPS, $\overline{x}$ is the mean value and σ is the standard deviation. By our statistics, we have about 47% and 56% of dialog sessions that have valid NPS labels for Learning and MacHelp datasets respectively, which makes the dialog routing become a much more challenging task in the real world.

5.3 Training Settings

We implement our model based on the Transformer library[5]. We use the Adam (Kingma and Ba, 2014) optimizer to train models on each dataset for 5 epochs. The learning rate is set as $2e$-5 and the dropout rate is set as 0.1. We use the largest batch size that can fit in the memory of GPU. α, β and γ are set as 0.9, 1, 0.01 respectively by our empirical experiments. All the experiments are conducted on V100 32GB GPUs. Each dataset is randomly split into training/validating/testing sets in the proportion of 8:1:1. All the parameters are tuned on the validation set and the results follow the 5-fold cross validation in testing set.

[5] https://github.com/huggingface/transformers

	Learning	MacHelp
avg. tokens per utterance	10.19	10.29
avg. turns per dialog session	22.23	30.43
# of total tokens	66,031	154,023
# of total dialog sessions	22,897	37,944
# of dialog categories	7	4
# of intents	66	28

Table 2: Statistics for Learning and MacHelp datasets

5.4 Baselines

We compare our proposed model to several baselines which belong to two groups. One includes single-task models for NPS prediction and dialog category classification separately, while the other contains methods under multi-task learning paradigm. To make a fair comparison between baselines and the G3M, we concatenate the role and intent information ahead each utterance only if the baseline model can employ that information. Thus all the baseline models have the consistent input information as ours. The evaluated baselines are listed as follows:

- fastText (Joulin et al., 2017) is a simple and efficient model for regression or classification.
- HAN (Yang et al., 2016) is a document classification model based on hierarchical attention network by considering both word-level and sentence-level information.
- textReg (Dereli and Saraçlar, 2019) is a regression model based on convolution neural networks, which achieves the state-of-the-art performance on financial prediction.
- CNN-BiRNN-Att (CBA) (Yu et al., 2020) is a cascade model with CNN, bi-directional RNN and attention mechanism. We adapt the last layer to support either the regression or classification tasks.
- BERT (Devlin et al., 2019) is a model which can be fine-tuned by adjusting the task-specific output layers. We use the pre-trained

BERT-base-cased model to separately fine-tune the two tasks with default parameters.

- XLNET (Yang et al., 2019) is a generalized auto-regressive pre-training method which can be fine-tuned by tailing a task-specific output layers. We fine-tune the two tasks with this pre-trained model similar to BERT.
- Joint-CNN-BiRNN-Att (Joint-CBA) is a multi-task model by extending the last layer of CBA model to simultaneously predict NPS and dialog categories.
- Vanilla Multi-task Model (VMM) follows the vanilla multi-task learning paradigm and uses the standard BERT encoder to encode only the utterance data (Huang et al., 2021), without any gated-mechanism modules.

To measure the performance, we use the root mean squad error (RMSE) for the NPS regression task and micro F1 score (Micro-F1) for the dialog category classification task.

5.5 Results and Analysis

We compare our model with various single-task and multi-task baselines on the two datasets, and the first two sections of Table 3 report the results. The performance of NPS regression task is reflected by RMSE metric, while that of dialog category classification task is embodied by Micro-F1 metric. We can learn some observations from Table 3.

Firstly, the performance of our proposed model G3M is consistent on both datasets and can outperform all the baselines. More specifically, for the NPS regression task, G3M's RMSE is 8.72% and 11.83% lower than the best RMSE score of baseline models (VMM for Learning dataset and Joint-CBA for MacHelp dataset) on two datasets. And for the dialog category classification task, G3M upgrades the performance by achiving 2.17% and 4.40% higher Micro-F1 scores compared with the best baseline method (VMM). All the results demonstrate that our model is effective and achieves the state-of-the-art performance on both subtasks.

Secondly, comparing the single-task models with multi-task models among baselines, it is consistent that multi-task models can surpass the single-task counterparts, especially from the comparison between Joint-CBA and VMM with their single-task versions (CBA and BERT). The results suggest that the underlying relation between the two tasks is helpful and multi-task learning paradigm can well capture the knowledge.

Models		Learning		MacHelp	
		RMSE	Micro-F1	RMSE	Micro-F1
Single-task	fastText	0.9711	77.85	1.2365	85.88
	HAN	0.9981	78.85	1.1276	84.19
	textReg	0.9689	77.98	1.1099	85.55
	CBA	0.9697	79.42	1.1383	86.02
	BERT	0.9702	80.04	1.1025	86.39
	XLNET	0.9648	80.21	1.1084	86.25
Multi-task	Joint-CBA	0.9555	80.38	1.0624	87.21
	VMM	0.9302	81.88	1.0767	87.49
	G3M	**0.8491**	**83.66**	**0.9367**	**91.34**
Ablation	- IC-Gated module	0.8892	82.57	0.9883	90.63
	- NC-Gated module	0.8878	82.73	0.9831	90.36
	- both gated modules	0.9187	82.01	1.0591	88.27

Table 3: Comparison between G3M and baselines on Learning and MacHelp datasets. RMSE is for NPS prediction and Micro-F1 (%) is for dialog classification.

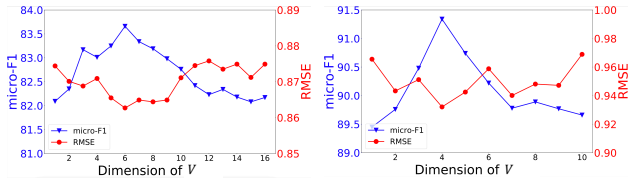

(a) Learning Corpus (b) MacHelp Corpus

Figure 3: The parameter effect on different dimensions of V in IC-Gated module of G3M.

Thirdly, comparing three multi-task models, we observe that G3M is better than the other two methods. We can conclude that both our proposed dialog encoder and gated mechanism are effective, since Joint-CBA and VMM do not contain either of the modules. Another observation is that under the same multi-task learning paradigm, BERT based dialog encoder (VMM) is better than the CNN-RNN based dialog encoder (Joint-CBA).

5.6 Ablation Study

To probe G3M's effectiveness in terms of gated module level, we conduct ablation experiments and report the results in the bottom section of Table 3. We could obtain three variant models by once deducting one or two gated modules: G3M without IC-Gated module, G3M without NC-Gated module, and G3M without either module. Based on the experiments, we have some findings.

Comparing the three ablated variant models of G3M, both modules contribute positively to the two tasks. Therefore, they are effective to model the data-to-task and task-to-task information among dialog data. Specifically, it is not very clear that which module is more important than the other. However, ablating both them would damage the final performance.

Furthermore, we compare the performance between G3M without both gated modules and multi-

task baselines (Joint-CBA and VMM), we find the former model performs better than the later, which may indicate that our proposed dialog encoder is superior than the CNN-RNN based and BERT based dialog encoders.

5.7 Parameter Effect

In the IC-Gated module, there is an important parameter to be preset by humans, which is the dimension of V. It determines the size of weighted feature $g^{(I)}$ and therefore controls to what degree the data-to-task information can be leveraged. We investigate the effect of various dimensions of V on both datasets. From the curves of RMSE and Micro-F1 scores in Figure 3, we find that the final performance is related to this parameter. Empirically, the optimal setting is 6 for Learning dataset and 4 for MacHelp dataset. We can also observe too small or too large dimensions may both play a harmful effect on extracting important knowledge from the data-to-task information. Another interesting phenomenon is that the optimal parameter values are coincidentally close or equal to the numbers of dialog categories to be classified, and we would explore the potential correlation in the future.

5.8 Case Study

We conduct a case study to demonstrate how our method could work to improve the system efficiency and user experience by integrating our dialog routing component into a human-bot symbiosis dialog system. Table 4 lists the dialog utterances along with the predicted NPS and dialog category. Here the user (U) has a requirement to solve course enrollment issue. At first, the bot agent (A1) serves the user which is skilled at 'Finding Content' dialogs, but it wrongly understands the user that she would find some course materials. By using the dialog routing component to monitor the dialogs, the predicted NPS decreases from 5.8 to 4.8, which means the user is dissatisfactory with the A1 and the routing to other agent should be triggered[6]. As a result, based on the dialog category classification by dialog routing component, the dialog is transferred to a human agent (A2) who is skilled at 'Completion Issue' dialogs. Thanks to the right time to transfer and the correctly assigned agent, the NPS is up to 7.8 and the user is finally satisfactory with this service.

[6]We can set a dissatisfaction threshold based on real situations, e.g. 5 in this case.

Role	Utterances
U	Hi
A1	Hello, how may I assist you?
U	I have enrolled in Deep Learning Course but today I can't see this enrollment
U	I have completed more than 80%
U	Can you please help me to find / resume the course
A1	I'm sorry that we are not able to do that
U	Why so
U	and what help you can do?
Current NPS: 5.8 Current dialog category: Finding Content	
A1	You could get more information in the completion dashboard in the system
U	You did not answer my question, I can't find the course and I want to resume it
U	Can you please help me to find / resume the course
A1	I'm sorry that we are not able to do that
Current NPS: 4.8 (**under the threshold and trigger routing**) Current dialog category: Completion Issue	
A2	Sorry for not being very helpful.
A2	You can't find the course in your learning queue and the progress are all gone, correct?
U	Exactly
Current NPS: 6.7 Current dialog category: Completion Issue	
A2	Is it an internal course in the Learning system, right?
U	Yes
Current NPS: 6.9 Current dialog category: Completion Issue	
A2	Sorry for your confusion, sometimes it may be caused by technical issues.
A2	A ticket has been created for you so that it will be reflected correctly.
A2	Normally the technical team will get back to you in 24 hours
U	Thanks, bye
A2	Bye
Current NPS: 7.8 Current dialog category: Completion Issue	

Table 4: A dialog session along with the predicted NPS and dialog category by equipping a dialog routing component with our method.

This case could also reveal that our method would not affect the existing dialog systems and the dialog routing system can be deployed in a plug-in manner, which has a good compatibility and is convenient for both dialog routing system and dialog system to upgrade their ability.

6 Conclusion

In current ubiquitous human-bot symbiosis dialog systems for customer care, the dialog routing component is necessary to improve the overall system efficiency, reduce human resource cost, and enhance user experience. In this paper, we argue that the data-to-task and task-to-task information among various kinds of dialog data and subtasks should be jointly leveraged to perform better dialog routing ability. We propose a Gated Mechanism

enhanced Multi-task Model (G3M) to implement that motivation. Specifically, we design a new dialog encoder to learn various kinds of dialog data by extending the BERT encoder, and two gated mechanism modules are proposed to capture data-to-task and task-to-task information. Extensive experiments on two real-world datasets demonstrate the effectiveness of our proposed model, which can achieve the state-of-the-art performance.

Acknowledgement

We thank all the anonymous reviewers for their valuable feedback and insightful comments. This work is partially supported by the Natural Science Foundation of Jiangsu Province, China (Grant No. BK20220488).

References

Yubo Chen, Hang Yang, Kang Liu, Jun Zhao, and Yantao Jia. 2018. Collective event detection via a hierarchical and bias tagging networks with gated multi-level attention mechanisms. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 1267–1276.

Arman Cohan, Iz Beltagy, Daniel King, Bhavana Dalvi, and Daniel S Weld. 2019. Pretrained language models for sequential sentence classification. *arXiv preprint arXiv:1909.04054*.

Neşat Dereli and Murat Saraçlar. 2019. Convolutional neural networks for financial text regression. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics: Student Research Workshop*, pages 331–337.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, Minneapolis, Minnesota. Association for Computational Linguistics.

Zefeng Du, Peijie Huang, Yuhong He, Wei Liu, and Jiankai Zhu. 2019. A knowledge-gated mechanism for utterance domain classification. In *CCF International Conference on Natural Language Processing and Chinese Computing*, pages 142–154. Springer.

Chih-Wen Goo, Guang Gao, Yun-Kai Hsu, Chih-Li Huo, Tsung-Chieh Chen, Keng-Wei Hsu, and Yun-Nung Chen. 2018. Slot-gated modeling for joint slot filling and intent prediction. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 2 (Short Papers)*, pages 753–757.

Ziming Huang, Zhuoxuan Jiang, Hao Chen, Xue Han, and Yabin Dang. 2021. Dialog router: Automated dialog transition via multi-task learning. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 35, pages 16038–16040.

Tatsuya Ide and Daisuke Kawahara. 2021. Multi-task learning of generation and classification for emotion-aware dialogue response generation. *CoRR*, abs/2105.11696.

Armand Joulin, Edouard Grave, Piotr Bojanowski, Matthijs Douze, Hérve Jégou, and Tomas Mikolov. 2017. Fasttext. zip: Compressing text classification models. In *The International Conference on Learning Representations (ICLR)*.

Yoon Kim. 2014. Convolutional neural networks for sentence classification. In *Conference on Empirical Methods in Natural Language Processing*.

Diederik P Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*.

Harshit Kumar, Arvind Agarwal, Riddhiman Dasgupta, and Sachindra Joshi. 2018. Dialogue act sequence labeling using hierarchical encoder with crf. In *Thirty-Second AAAI Conference on Artificial Intelligence*.

Thomas L Ngo-Ye and Atish P Sinha. 2014. The influence of reviewer engagement characteristics on online review helpfulness: A text regression model. *Decision Support Systems*, 61:47–58.

Shereen Oraby, Pritam Gundecha, Jalal Mahmud, Mansurul Bhuiyan, and Rama Akkiraju. 2017. How may i help you?: Modeling twitter customer service-conversations using fine-grained dialogue acts. In *Proceedings of the 22nd International Conference on Intelligent User Interfaces*, pages 343–355. ACM.

Bikesh Kumar Singh, Kesari Verma, and AS Thoke. 2015. Investigations on impact of feature normalization techniques on classifier's performance in breast tumor classification. *International Journal of Computer Applications*, 116(19).

Jenq-Haur Wang, Ting-Wei Liu, Xiong Luo, and Long Wang. 2018. An lstm approach to short text sentiment classification with word embeddings. In *Proceedings of the 30th conference on computational linguistics and speech processing (ROCLING 2018)*, pages 214–223.

Liqiang Xiao, Honglun Zhang, and Wenqing Chen. 2018. Gated multi-task network for text classification. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT, New Orleans, Louisiana, USA, June 1-6, 2018, Volume 2 (Short Papers)*, pages 726–731. Association for Computational Linguistics.

Zhilin Yang, Zihang Dai, Yiming Yang, Jaime Carbonell, Russ R Salakhutdinov, and Quoc V Le. 2019. Xlnet: Generalized autoregressive pretraining for language understanding. In *Advances in neural information processing systems*, pages 5753–5763.

Zichao Yang, Diyi Yang, Chris Dyer, Xiaodong He, Alex Smola, and Eduard Hovy. 2016. Hierarchical attention networks for document classification. In *Proceedings of the 2016 conference of the North American chapter of the association for computational linguistics: human language technologies*, pages 1480–1489.

Yipeng Yu, Ran Guan, Jie Ma, Zhuoxuan Jiang, and Jingchang Huang. 2020. When and who? conversation transition based on bot-agent symbiosis learning network. In *Proceedings of the 28th International Conference on Computational Linguistics, COLING 2020, Barcelona, Spain (Online), December 8-13, 2020*, pages 4056–4066. International Committee on Computational Linguistics.

Negation, Coordination, and Quantifiers in Contextualized Language Models

Aikaterini-Lida Kalouli[1]*, **Rita Sevastjanova**[2]*, **Christin Beck**[2], and **Maribel Romero**[2]
[1] CIS - LMU Munich, kalouli@cis.lmu.de
[2] University of Konstanz, firstname.lastname@uni.kn

Abstract

With the success of contextualized language models, much research explores what these models really learn and in which cases they still fail. Most of this work focuses on specific NLP tasks and on the learning outcome. Little research has attempted to decouple the models' weaknesses from specific tasks and focus on the embeddings per se and their mode of learning. In this paper, we take up this research opportunity: based on theoretical linguistic insights, we explore whether the semantic constraints of function words are learned and how the surrounding context impacts their embeddings. We create suitable datasets, provide new insights into the inner workings of LMs vis-a-vis function words and implement an assisting visual web interface for qualitative analysis.

1 Introduction

Recently, tremendous progress has been observed in the development of contextualized language models (LM). After the introduction of contextualized embeddings like ELMo (Peters et al., 2018) and BERT (Devlin et al., 2019), earlier static word embeddings (Mikolov et al., 2013; Pennington et al., 2014) have been sidelined, and new standards have been set for the state-of-the-art. Particularly, LMs have been shown to learn task-agnostic properties of language (e.g., Belinkov, 2018; Conneau et al., 2018; Peters et al., 2018) and linguistic properties that imitate the classical NLP pipeline (Tenney et al., 2019a). Despite this success, LMs cannot be taken to understand language the way humans do, as they fail to generalize on unseen data and tasks requiring compositionality (McCoy et al., 2019; Richardson et al., 2020).

Research efforts concentrate on shedding light on what these models learn and consequently how they can be improved. One strand of work relies on probing/diagnostic tasks, where classifiers are trained on LMs to determine whether they can encode specific linguistic properties, e.g., Marvin and Linzen (2018); Tenney et al. (2019b); Hewitt and Manning (2019); Talmor et al. (2020). Another strand of research focuses on creating adversarial test sets with hard linguistic phenomena to observe where the models fail and thus reverse-engineer them, e.g., McCoy et al. (2019); Richardson et al. (2020); Nie et al. (2020). Both strands of research have approached their goals by employing specific tasks, e.g., Natural Language Inference, Question-Answering, etc., and this has revealed weaknesses of the current models, e.g., that negation is not treated according to its semantic nature. Nevertheless, there is still little work, e.g., Ethayarajh (2019); Sevastjanova et al. (2021), decoupling the weaknesses of LMs from specific NLP tasks and focusing on which of their inner workings are responsible for these weaknesses.

In this paper, we attempt to fill this gap by taking a closer look at function words, i.e., words with little lexical meaning. Function words have been traditionally ignored in NLP, being dismissed as stopwords. Since the rise of contextualized embeddings, function words are not dismissed anymore but do not receive any special treatment either; instead, they are contextually learned like any other word. However, the linguistically-motivated datasets created in this work and tested within the masked-language-modeling (MLM) task show that this context-based learning does not effectively capture the functionality of such words and, thus, that the weaknesses that LMs show in semantic tasks can be attributed to these ill-learned representations. Particularly, our work sheds light to the type and quality of masked-word predictions when **negation**, **coordination**, and **quantifiers** are involved. This work focuses on English masked LMs, but the basic findings should be extendable to other languages since the training process of most masked LMs is relatively similar.

* Equal contribution.

Proceedings of the 29th International Conference on Computational Linguistics, pages 3074–3085
October 12–17, 2022.

With this, the contributions of this work are fourfold. First, we show the semantic constraints of three of the most studied classes of function words – negation, coordination, and quantifiers. Second, we provide linguistically-motivated datasets that capture the semantic constraints of these function words. Based on these datasets, we offer new quantitative and qualitative insights into the treatment of function words in masked LMs. The qualitative insights are facilitated by an openly-available web interface that visualizes LM predictions and allows researchers to easily test their hypotheses.

2 Related Work

Most related work, i.e., probing studies and adversarial testing has focused on specific NLP tasks and on the learning outcome of the models. However, recently, research has also concentrated on exploring the learning procedure per se and understanding the LM inner workings as per the way of their training. With this, the study of contextualization has emerged. The embeddings are learned based on the co-occurrence of words in particular contexts. Thus, the LM embeddings are contextualized, i.e., a word has different representations based on the context it is found in. Particularly, Ethayarajh (2019) shows that the embeddings become more contextualized, i.e., more context-specific, in the upper layers of BERT, and that contextualization is not entirely driven by polysemy: (non-polysemous) stopwords such as 'and', 'of', 'the' and 'to' also become increasingly contextualized in the upper layers, and their representations are among the most context-specific ones. Rather, he finds that contextualization seems to be driven by the variety of contexts a word appears in. These findings are further explored by Sevastjanova et al. (2021), who show that contextualization is neither driven by polysemy nor by pure context variation, but by the combination of functionality, sense variation, syntactic variation, and semantic context variation: BERT can efficiently model polysemous, homonymous and monosemous words, and also words that appear in semantic contexts of high and low variation and independently of their polysemy. But it cannot model words that have a semi-functional/semi-content nature (e.g., modals, quantifiers, temporal adverbials). In this work, we take up this research direction and further explore what is exactly learned during the learning of function words and how their surrounding context influences their learned embeddings.

3 Negation, Coordination, and Quantifiers

3.1 Theoretical Linguistics

Traditionally in theoretical linguistics, there is a distinction between function and content words and several criteria to distinguish the two. The probably most popular criterion is that of semantic content: content words establish a specific semantic content and contribute to the principal meaning of a sentence; function words are rather "non-conceptual" and mainly fulfill some grammatical function, gluing content words together. Other criteria include membership openness (i.e., whether new members can be added to each of the two classes), the flexibility of syntactic attachment (i.e., whether they can combine with any syntactic phrase), and separability from complements (i.e., whether they can be detached from their lexical head) (Corver and van Riemsdijk, 2001). The distinction between the two classes is not always clear-cut as there are words that share both functional and lexical properties. Thus, it has often been argued that function and content words should rather form a quasi-continuum ('squishiness') (Ross, 1972; Emonds, 1985). For example, prepositions are less functional than articles, e.g., some prepositions are associated with a locative or directional meaning, but they are also more functional than nouns or verbs, e.g., because they are inseparable from their content words.

In this work, we focus on three types of function words, which represent core notions of logic, mathematics, and human reasoning and ones that a state-of-the-art NLP system should efficiently be able to handle: the boolean notions of complement (negation), intersection (conjunction) and union (disjunction), and the notion of existential and universal quantification. Specifically, we study the negation markers *not* and *without*, the coordination markers *or* and *and*, and the existential quantifier *some* and the universal quantifiers *all* and *no*.

Negation Markers *Not* can be considered syntactic negation while *without*, being a preposition with some lexical content, can be considered lexical negation. Semantically, at the heart of negation lies the notion of inconsistency (Ladusaw, 1996): A given sentence and its negation, as in the episodic pair (1) and the generic pair (2), are inconsistent with each other, i.e., they have disjoint truth conditions (where e.g. MOTHER is short for $\{x : x \text{ is a mother}\}$ and $\overline{A}$ is the complement of A):

(1) Maria is a mother. $m \in \text{MOTHER}$
Maria is not a mother. $m \in \overline{\text{MOTHER}}$

(2) A mom is a mother. $\text{MOM} \subseteq \text{MOTHER}$
A mom is not a mother. $\text{MOM} \subseteq \overline{\text{MOTHER}}$

Additionally, negation in natural language interacts with its clausemate elements for meaning composition in interesting ways. To name one case, negation is a well-known "hole" for presuppositions, i.e., it can negate the at-issue, propositional content of its clause but not its presupposed content (Karttunen and Peters, 1979). For example, the sentence *Joe isn't sick with covid again* denies that Joe has covid now (at-issue content), but it does not deny that he had covid before (presupposed content). In this paper, we concentrate on the notion of inconsistency or disjointness and test whether current LMs have grasped this semantic functionality.

Coordination Markers Moving on to the coordination markers *and* and *or*, the former is used to form conjunctions and the latter disjunctions. In its run-of-the-mill boolean use, *and* semantically amounts to the intersection operation: Predicating a conjunctive property of an individual, e.g. Joe, amounts to asserting membership to the intersection of the two conjuncts, as in (3-a). In contrast, though the purely semantic content of *A or B* amounts to set union and thus to inclusive disjunction (meaning 'A, B or both'), this meaning is typically strengthened to that of *ex*clusive disjunction (i.e., 'A or B and not both') (Horn, 1972; Sauerland, 2004). This is linked to Hurford's observation that a disjunction of shape *A or B* is odd if there is an entailment relation between A and B, as defined in (4) and illustrated in (3-b) (Hurford, 1974; Singh, 2008; Ippolito, 2020):

(3) a. Joe is a dolphin and a mammal.
$j \in \text{DOLPHIN} \cap \text{MAMMAL}$
b. #Joe is a dolphin or a mammal.
$j \in \text{DOLPHIN} \cup \text{MAMMAL} \wedge j \notin \text{DOLPHIN} \cap \text{MAMMAL}$

(4) Hurford's Constraint:
A disjunction of the form *A or B* is infelicitous (i.e., #) if A entails B or vice-versa.

This means that *and* and *or* have different semantic signatures when it comes to the relation between their coordinated terms: *A and B* allows for A to entail B, whereas *A or B* prohibits it. We will test whether LMs are able to detect and learn these different meaning signatures.

There are other interesting semantic properties of coordination markers in natural language that could be explored. To name just one, *and* has, in addition to its boolean meaning, a non-boolean meaning that roughly amounts to the creation of a new complex plural individual (Link, 2002). For example, (5) allows not only for the boolean, distributive reading (5-a) and also for the non-boolean, collective reading (5-b).

(5) Jane and Paul built a castle.
a. 'Jane built a castle and Paul built a castle.'
b. 'Jane and Paul together built a castle.'

Quantifiers Semantically, quantifiers like *all/every*, *some* and *no* denote relations between the set P coming from the noun phrase headed by the quantifier and the set Q coming from the rest of the sentence (Barwise and Cooper, 1981):

(6) a. $[\![\text{All Ps are Q}]\!] = 1$ iff $P \subseteq Q$
b. $[\![\text{Some Ps are Q}]\!] = 1$ iff $P \cap Q \neq \emptyset$
c. $[\![\text{No Ps are Q}]]\!] = 1$ iff $P \cap Q = \emptyset$

Two points follow from these lexical entries. First, *all* and *no* are polar opposites (i.e., they are contrary –though not complementary– items (Ladusaw, 1996; Cruse, 2011)). This means that the statements *All Ps are Q* and *No Ps are Q* are inconsistent with each other. Second, the statement *All Ps are Q* technically entails the statement *Some Ps are Q* (under the assumption that set P is non-empty). However, the two quantifiers form part of the lexical scale $<$*some*, *several*, ... , *many*, *most*, *all*$>$, where the leftmost item is the weakest, and the rightmost item is the strongest (Horn, 1972). As it typically happens with scales in natural language, pragmatic implicatures are routinely run on the weaker terms, leading to readings not entailed anymore by the stronger terms. In the case of *some*, its original weaker reading 'some and possibly all' in (6-b) is upgraded into the stronger pragmatic reading 'some but not all' in (7). Under this second reading, the statement *All Ps are Q* does no longer entail the statement *Some Ps are Q*; rather, the two are incompatible with each other.

(7) $[\![\text{Some}_{str} \text{ Ps are Q}]\!] = 1$ iff
$P \cap Q \neq \emptyset$ and $P \not\subseteq Q$

For an LM understanding the meaning of the quantifiers *all*, *some*, and *no* and for a fixed value of P, the Q-embeddings (i.e., embeddings of the set Q) of the noun phrase *All Ps* should be radically different from the Q-embeddings of *No Ps*, since they are polar opposites (i.e., contraries). Similarly, the Q-embeddings of the noun phrase *All Ps* should differ substantially from the Q-embeddings of the pragmatically strengthened *Some Ps*, since they are incompatibles. In other words, in both cases, the Q-embedding of the quantificational noun phrases under comparison should be disjoint. We will test whether LMs show traces of semantic understanding in being able to create such disjoint predictions.

3.2 Computational Linguistics/NLP

Within computational linguistics, function words have mostly been treated as *stopwords*. The term was coined by Luhn (1960) to mean very common words that do not add much to the meaning of a text but ensure the structure of a sentence is sound. Historically, one of the main reasons for removing stopwords was to decrease the computational time for text mining (Huston and Croft, 2010) and help search engines to give better results. Nowadays, this reason for removing stopwords is not valid anymore since we have computationally more powerful hardware; nevertheless, various NLP tasks such as topic modeling and information retrieval, continue to use the practice as it is often argued to improve performance (e.g., Fan et al. (2019); Sarica and Luo (2020)).

With the rise of distributional semantics and word embeddings, more research focused on the nature of these words and the special treatment required (e.g., (Bernardi et al., 2013; Hermann et al., 2013; Linzen et al., 2016; Tang et al., 2016)). More recently, the special nature of function words has also interested the LM community. In one of the probing tasks proposed by Kassner and Schütze (2020) it is shown that LMs cannot distinguish between negated ("Birds cannot [MASK]") and non-negated ("Birds can [MASK]") questions. The researchers insert the negation *not* into the LAMA dataset (Petroni et al., 2019) and create positive and negative cloze questions. By comparing the predictions of LMs on these pairs, they find that the positive and negative predicted fillers have high overlap and correlation, i.e., models are equally likely to generate true (e.g. *birds can fly*) and incorrect statements (e.g., *birds cannot fly*). High correlation means that the models do not understand negation; correct answers for positive and negative questions are expected to be disjoint. Interestingly, the researchers show that BERT can learn both positive and negative facts correctly if they occur in training but still fails to generalize to unseen (positive and negative) sentences.

4 Experiments

In this work we focus on the three functional categories discussed and set out to complement the existing research on the treatment of function words within contextualized LMs. First, we create datasets that can be used to evaluate how LMs have learned to capture the semantic constraints of these functional categories. Second, we generate quantitative insights into the contextualization of function words, which we qualitatively investigate through a user-friendly web interface, which also allows researchers to evaluate their own datasets.

4.1 Data

Motivated by the experiment performed by Kassner and Schütze (2020) on the negated LAMA dataset and the theoretical linguistic research about the type of data needed for this kind of exploration, we create two datasets of different types.[1] For the creation of the datasets, we combine existing resources: the family and location relations of the analogy dataset used by Mikolov et al. (2013), the English occupations dataset of the European Skills and Competences, Qualifications and Occupations (ESCO) initiative[2] and 5 common-sense relations found in ConceptNet (Speer et al., 2016) (*has-a, used-for, is-a, has-property, capable-of*). Based on these resources, we create templates which are then used to produce the sentences of our datasets. Examples for both datasets can be found in the Appendix. More details are given in the following.

Inconsistent Dataset The Inconsistent Dataset aims at revealing whether the predictions in sentences with inconsistent meaning are indeed disjoint. Disjointness is factored in through inconsistent function words, as described in Section 3.1. Parts of the inconsistent sentences are masked and the goal is to detect overlapping predictions within each pair. An overlap means that the LM has not

[1] Can be found under: https://function-words.lingvis.io/

[2] ec.europa.eu/esco/portal/download

learned the functionality of the inconsistent functional words. The Inconsistent Dataset consists of 1272 pairs and can be split into three subsets: the negation subset (534), the coordination subset (500), and the quantifiers subset (238).

The negation subset contains pairs with positive and negative sentences, whereby the negation can be syntactic or lexical. Concerning syntactic negation, the subset contains examples with the copula verb *be* and its negation, and transitive verbs and their negation with *not* and *no*. As far as lexical negation is concerned, inconsistent sentences are created with the copula verb *be* and transitive verbs and based on the preposition *with-without*. The coordination subset contains a single type of inconsistency by using the markers *or* and *and* and masking the second part of the coordination. The quantifiers subset has two types of opposing contexts: the first one is formulated through *all-no* and the second one through *all-some*. Both transitive and intransitive sentences are considered.

Semantic Dataset Although the Disjoint Dataset goes beyond the negated LAMA in that it contains an additional type of negation (lexical negation) and two further types of inconsistency (coordination and quantifiers), it can indicate whether the LMs learn the semantic constraints of these functional categories only *indirectly*, i.e., by capturing the predictions' overlap. Thus, a different "semantic" dataset is required, on which we can directly judge the correctness of the predictions. This can be made clearer with an example such as *A mother is not a [MASK]*: based on the semantic constraints of negation discussed in Section 3.1 and the generic reading 'For any x, if x is a mom, then x is not a MASK', predicted masks should not be one of the words *mom, grandmother, grandma, granddaughter, bride, wife, woman, niece, stepmother, daughter, aunt, etc.* Another example is the sentence *John was born in Berlin or in [MASK]*, where the predicted masks should not contain the word *Germany*; again, see constraints in Section 3.1. We call these words *forbidden* although they could be valid sentences in some (e.g., figurative) contexts and could theoretically have been seen during LM training. However, since these sentences are logically incorrect and do not represent the prototypical concepts of things, they should not be among the most likely predictions of the LM – since LMs learn based on the occurrences of things, literal, prototypical meanings should have been encountered more often. Thus, if the most likely predictions contain the forbidden words, it means that the model has neither learned the functionality of the corresponding operator nor any proper world knowledge, e.g., that a *mother* is the same as a *mom* or that *Berlin* is part of *Germany*.

The creation of such a dataset is particularly challenging because the examples need to be chosen in a way that the necessary semantics is captured within the words themselves and not based on the co-occurrence of the words. Only in this way can we reliably evaluate whether LMs have learned something about the actual semantic constraints of the functional markers or they simply reproduce common concordances of the training data. This means that an example such as *John lives in Berlin or in [MASK]* is not suitable because any predicted word could be right. Thus, to produce such examples, we systematically create examples related to concepts for which we can selectively define invalid predictions according to the particular concept's characteristics. We choose family relations and occupations for syntactic negation, animals and their main body parts and activities for lexical negation, and capital countries and animals and their hypernyms for coordination. For the selected quantifiers, no "semantic" examples with forbidden answers could be created because of the very nature of quantifiers, i.e., even the universal *all* is not strong enough to create logical invalid examples in the real world. The Semantic Dataset contains a total of 2780 sentences with 187 examples of syntactic and 123 examples of lexical negation and 2470 examples of coordination. The size of the dataset is comparable to existing datasets used in other related experiments (e.g., the negated LAMA dataset by Kassner and Schütze (2020)).

4.2 Models

For our study, we use the *huggingface* (Wolf et al., 2020) implementation of three pretrained LMs, BERT (Devlin et al., 2019)[3], RoBERTa (Liu et al., 2019)[4], and ALBERT (Lan et al., 2019)[5], which have been shown to achieve state-of-the-art results on the GLUE, RACE, and SQUAD benchmarks. Our two datasets are input to each of the three models to extract the masked predictions, layerwise attentions, and the word embeddings of all words

[3] https://huggingface.co/bert-base-uncased

[4] https://huggingface.co/roberta-base

[5] https://huggingface.co/albert-base-v2

of each sentence. The embeddings are taken from layer 11, as the higher layers of models like BERT have been shown to mostly capture semantic properties, while the last layer has been found to be very close to the actual classification task and thus to be less suitable (Jawahar et al., 2019; Lin et al., 2019). Word piece embeddings are merged to their corresponding word embeddings through averaging. Based on the predictions, the word embeddings, and attentions, we calculate the following:

- the cosine similarity between the embedding vector of the predicted word to each other word of the sentence
- the layerwise average attention of the predicted word to each other word of the sentence
- for the Inconsistent Dataset: the overlap of the predictions between the two inconsistent versions of the sentences; overlap in the first returned prediction (`overlap@1`), in the first 5 (`overlap@5`), and first 10 (`overlap@10`) returned predictions
- for the Semantic Dataset: the percentage of examples containing at least one forbidden word within the 1st prediction (`forbidden@1`), the first 5 (`forbidden@5`), and the first 10 predictions (`forbidden@10`).

4.3 Web Interface

In addition to the quantitative results, i.e., prediction overlaps and forbidden predictions, we implement a web interface[6] that visually shows the predictions, the layerwise average attentions, and the word similarities. The interface provides qualitative insights into similarity patterns that are common for different prediction outcomes.

In the interface, the user can select one of the three models (i.e., BERT, ALBERT, or RoBERTa) to explore its predictions. Sentences belonging to one dataset are grouped together, whereby disjoint sentence pairs from the Inconsistent Dataset are placed underneath each other for better comparability. The visualization of predictions consists of multiple elements, e.g., see Figure 1. Each prediction of a masked word is displayed as a row in the visualization (i.e., 10 rows for 10 predictions). On the left, we display the prediction's probability through a horizontal bar representation. Next to the probability, we display the sentence tokens visualized as colored rectangles. The color of the rectangle represents its cosine similarity or its layerwise average attention to the predicted word of the particular sentence. The darker the color, the higher the similarity/attention. Note that ALBERT and RoBERTa generally show higher cosine similarities between the predicted and the surrounding context words than BERT, i.e., they are displayed with darker colors on the visualized figures. On the right, we display the predicted word. To support the analysis of prediction overlaps as well as prediction of forbidden words, we color the words that overlap or are forbidden, respectively, in red color.

[6] https://function-words.lingvis.io/

5 Results and Discussion

The results of our experiments can be found in Tables 1 and 2. Table 1 shows the percentages of overlap of the inconsistent predictions within the first @1, @5, and @10 predictions. Table 2 shows the percentages of at least one forbidden word being included within the first @1, @5, and @10 predictions of the models.

model	Incons. Dataset	@1	@5	@10
BERT	coord	50	53	57
	neg	46	46	47
	quant	27	38	40
	all	41	45	48
ALBERT	coord	61	80	81
	neg	42	42	42
	quant	27	28	31
	all	43	50	51
RoBERTa	coord	40	58	63
	neg	38	40	41
	quant	25	29	31
	all	34	42	45

Table 1: Inconsistent Dataset: percentage of inconsistent pairs with overlapping predictions within the first x predictions.

model	Semantic Dataset	@1	@5	@10
BERT	synNeg	41	56	60
	lexNeg	51	72	73
	coord	32	73	87
	all	41	67	73
ALBERT	synNeg	14	51	60
	lexNeg	38	63	66
	coord	6	31	47
	all	19	53	58
RoBERTa	synNeg	48	64	71
	lexNeg	21	31	42
	coord	27	68	82
	all	32	54	65

Table 2: Semantic Dataset: percentage of sentences in which there is at least one forbidden prediction within the first x predictions.

5.1 Findings

In the following, we describe findings made using quantitative and qualitative evaluation methods.

Same predictions for inconsistent pairs: We can reproduce and extend the findings by Kassner and Schütze (2020). The first prediction (`overlap@1`) of disjoint pairs overlaps in 41% of the cases for BERT, 43% for ALBERT, and 34% for RoBERTa. The first 5 predictions (`overlap@5`) overlap in 45% for BERT, 50% for ALBERT, and 42% for RoBERTa, while the first 10 predictions (`overlap@10`) overlap in 48% for BERT, 51% for ALBERT, and 45% for RoBERTa. Specifically, `overlap@10` is worse in the coordination subset in all 3 models and best in the quantifiers subset. This overlap shows that the models neither learn nor consider the functional nature of these markers.

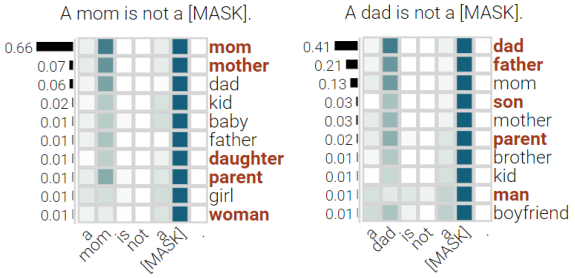

Figure 1: BERT: Similarity visualization of the first 10 predictions of the negation sentences *A mom is not a [MASK] - A dad is not a [MASK].*

Forbidden predictions for semantic pairs: `forbidden@1` lies at 41% for BERT, 19% for ALBERT, and 32% for RoBERTa. `forbidden@5` is at 67% for BERT, 53% for ALBERT, and 54% for RoBERTa, while `forbidden@10` lies at 73% for BERT, 58% for ALBERT, and 65% for RoBERTa. For BERT and RoBERTa the coordination subset seems to be the hardest. The easiest for BERT is the syntactic negation, and for RoBERTa – the lexical negation. In contrast, for ALBERT the easiest is the coordination subset, and the hardest is the lexical negation subset. These results might create the impression that BERT has the most difficulty in this task and that newer models such as ALBERT perform better. However, the further findings and the interpretation following shall shed light on this preliminary impression. Also, considering the embarrassingly easy examples included in our datasets, the amount of forbidden predictions is alarming for all datasets (see Figure 1, Figure 2[7]).

[7]Note that the sentence *The painter does not paint* is not an impossible sentence, and thus an LM might have seen it.

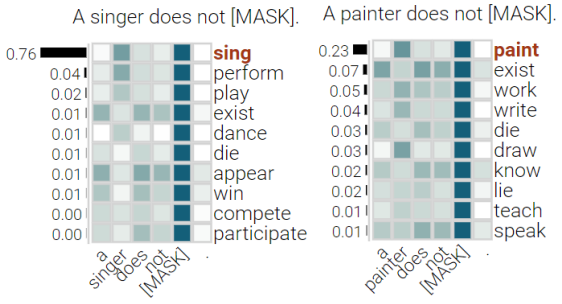

Figure 2: BERT: Similarity visualization of the first 10 predictions of the negation sentences *A singer does not [MASK] - A painter does not [MASK].*

[ALBERT] Similar predictions independently from context: For negation and quantifiers, RoBERTa and ALBERT perform better than BERT. However, if we examine the examples more closely, we see that ALBERT predicts similar words no matter the exact sentence (see Figure 3). For the Inconsistent Dataset, all positive sentences have similar predictions to each other and all negative ones as well. It is doubtful that the training data contained such concordances and this raises the question if this model is contextualized in the same way BERT is.

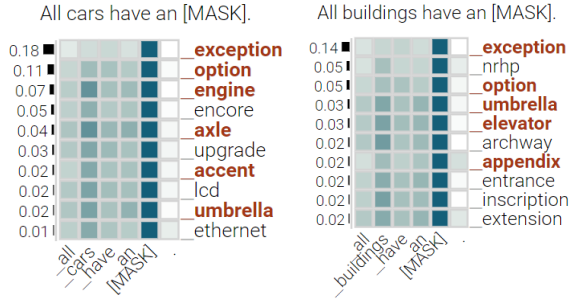

Figure 3: ALBERT: Similarity visualization of the first 10 predictions of the sentences *All cars have an [MASK] - All buildings have an [MASK].*

[ALBERT, RoBERTa] Predictions correlate only with the predicate: For the lexical negation subset of the Semantic Dataset, ALBERT's predictions are not based on the entire context, i.e., the predicate, the subject, and any function words, but rather correlate only with the predicate of the sentences. For example, all sentences containing the verb *fly* lead to the same predictions, no matter whether it is a *fly*, an *owl* or a *bird* flying (see Figure 4).

Similar observations can be made for RoBERTa.

However, as noted in Section 4.1, LMs should have mainly learned prototypical notions, and thus such predictions should not occur within the 10 most probable ones.

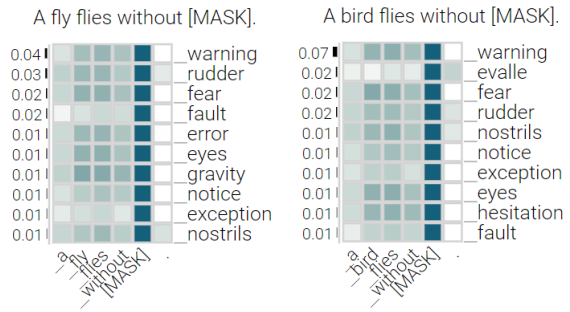

Figure 4: ALBERT: Similarity visualization of the first 10 predictions of the lexical negation sentences *A fly flies without [MASK] - A bird flies without [MASK].*

It has been shown that LMs can learn stereotypical associations reasonably well, e.g., that *walk* is related to *shoes* as in the given example in Figure 5. These associations suggest that LMs are capable of learning commonsense reasoning – knowledge accepted by the majority of people, e.g., how the world works (Bhargava and Ng, 2022). Although the given examples suggest that the models are capable to learn such associations, the visualizations reveal that the predictions are stronger related to the sentence's predicate than the subject. The same goes for the predicates such as *swim* or *see*.

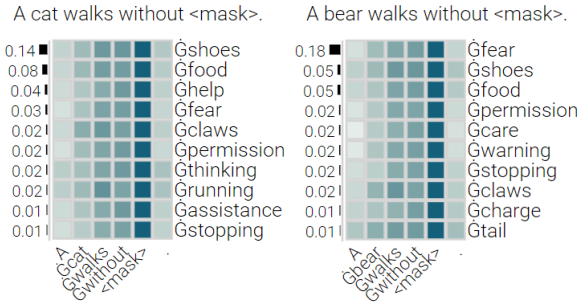

Figure 5: RoBERTa: Similarity visualization of the first 10 predictions of the lexical negation sentences *A cat walks without <mask> - A bear walks without <mask>.*

[BERT] Prediction quality dependent on specific named entities: Taking a closer look at the coordination partition of the Semantic Dataset and the location examples specifically, we observe a high similarity between the named entity of the location already contained in the sentence and the predicted word, which is also a named entity in 100% of the cases (see Figure 6). This not only indicates that the presence of named entities has a strong influence on the learning outcome because the remaining context is ignored, but also that the specific named entities have an impact on the predictions' quality. Particularly, some country/state-capital combinations lead to more forbidden predictions than others. For example, the US states *Texas, California, Arizona, Florida* are more often within the `forbidden@1` than states such as *Indiana, Tennessee, Minnesota*. This might suggest that the training data of BERT contained more instances of the former combinations, and the model learned a strong relation between these named entity pairs, ignoring any other (functional or lexical) words.

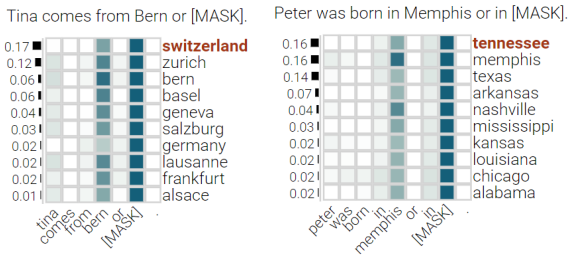

Figure 6: BERT: Similarity visualization of the first 10 predictions of the sentences *Tina comes from Bern or [MASK] - Peter was born in Memphis or in [MASK].*

5.2 Insights and Interpretation

In the following, we describe some potential reasons why current LMs fail in making linguistically correct word predictions.

Function words ignored in semantically related contexts: We find that if the masked word has some semantic relation with the other main concept of the sentence – most often the subject, then the functional word embeddings have low similarity to the masked word. In contrast, if no semantic relation can be established, there is a higher similarity to the function words. For example, in Figure 7, we can see that all predictions that are related to (animals') body parts and have some semantic relation with the subject *insects* do not exhibit any cosine similarity to the quantifiers, i.e., the first cell of the matrix is white, while for irrelevant predictions such as *nothing, died, eaten* there is a similarity between the predicted word and the quantifier. This indicates that in many cases the predictions are dominated by semantically rich words and the model ignores other functional operators.

Contextualization of function words: The visualizations show that if all words of the sentence have similar (high or low) similarities to the predicted word, then the predictions are mostly neither inconsistent nor forbidden – even if they do not belong to the average common sense. This suggests that if the masked word is predicted considering the whole context, it should have a similarity to

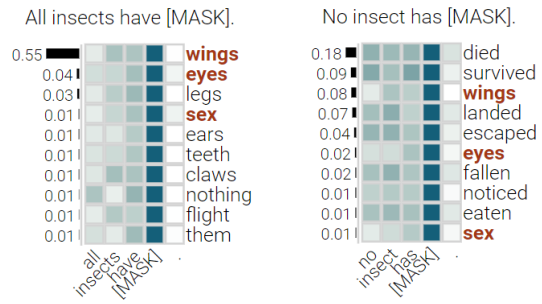

Figure 7: BERT: Similarity visualization of the first 10 predictions of the disjoint pair *All insects have [MASK] - No insect has [MASK].*

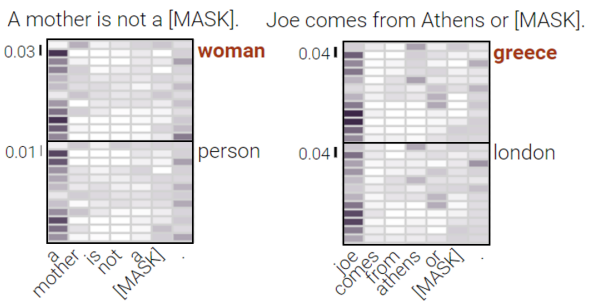

Figure 8: BERT: Attention visualization of 2 predictions of the sentences *A mother is not a [MASK]* and *Joe comes from Athens or [MASK].* For each prediction, rows represent the layerwise average attention from the predicted word to the words (columns) in the sentence.

all words in the context. Related work supports this assumption (Ethayarajh, 2019; Sevastjanova et al., 2021): functions words have high similarity to the rest of the words in higher layers because they are highly contextualized, i.e., they become very context specific. However, our findings show that this mode of learning leads to the negligence of function words: in some cases, the contextualization of function words is indeed high, i.e., their similarity to the predicted word is high, and then no inconsistent predictions arise. But when the sentence contains words that are semantically related, the model gets distracted, the function words are glossed over and their similarity to the predicted word drops. In consequence, their semantic nature is not captured.

Attention confirms insights: Visualizing attention can show whether the observed behavior is specific to cosine similarity or can also be retraced in attention patterns. Indeed, we observe that throughout the layers the first word receives the most attention no matter its part-of-speech. Function words, such as *not* and *or* in Figure 8, receive higher attention only in the middle layers, which have been shown to mainly capture syntactic properties (Jawahar et al., 2019). Thus, it seems that the semantic constraints of these words, which should be captured in the higher layers, are indeed left unaccounted for.

Models are sensitive to minor input changes: While extracting predictions for RoBERTa model, we noticed its high sensitivity to the provided input. Even additional space(s) between tokens in the input context can produce different prediction probabilities, ranks, or even different predictions. Thus, we want to sensibilize researchers to be careful when working on similar experiments in order to avoid such potential, undesired errors.

6 Conclusion

This paper presented the extent to which LMs learn semantic constraints of function words. Based on theoretical linguistic literature, we created new datasets for testing three functional classes, and showed that popular masked LMs make problematic predictions. The visualizations in the developed web interface highlighted potential reasons for this poor performance. In this work, we set out to shed light on a subset of function word categories. Future work shall look into other function word categories that ought to be challenging for LMs and uncover additional reasons for the poor model performances.

Acknowledgments

We thank the Deutsche Forschungsgemeinschaft (DFG, German Research Foundation) for funding within the project BU 1806/10-2 “Questions Visualized” of the FOR2111 and project D02 “Evaluation Metrics for Visual Analytics in Linguistics” (Project ID: 251654672 – TRR 161).

References

Jon Barwise and Robin Cooper. 1981. Generalized Quantifiers and Natural Language. In J. Kulas, J.H. Fetzer, and T.L. Rankin, editors, *Philosophy, Language, and Artificial Intelligence. Studies in Cognitive Systems*, pages 241–301. Springer, Dordrecht.

Yonatan Belinkov. 2018. *On internal language representations in deep learning: An analysis of machine translation and speech recognition.* Ph.D. thesis, Massachusetts Insitute of Technology.

Raffaella Bernardi, Georgiana Dinu, Marco Marelli, and Marco Baroni. 2013. A relatedness benchmark

to test the role of determiners in compositional distributional semantics. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 53–57, Sofia, Bulgaria. Association for Computational Linguistics.

Prajjwal Bhargava and Vincent Ng. 2022. Commonsense Knowledge Reasoning and Generation with Pre-trained Language Models: A Survey. *Proceedings of the AAAI Conference on Artificial Intelligence*, 36(11):12317–12325.

Alexis Conneau, German Kruszewski, Guillaume Lample, Loïc Barrault, and Marco Baroni. 2018. What you can cram into a single $&!#* vector: Probing sentence embeddings for linguistic properties. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 2126–2136, Melbourne, Australia. Association for Computational Linguistics.

Norbert Corver and Henk van Riemsdijk. 2001. *Semilexical Categories: The Function of Content Words and the Content of Function Words*. De Gruyter Mouton, Berlin, New York.

Alan Cruse. 2011. *Meaning in language: An introduction to semantics and pragmatics*. Oxford University Press, Oxford.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, Minneapolis, Minnesota. Association for Computational Linguistics.

Joseph E. Emonds. 1985. *A Unified Theory of Syntactic Categories*. De Gruyter Mouton, Berlin, Boston.

Kawin Ethayarajh. 2019. How Contextual are Contextualized Word Representations? Comparing the Geometry of BERT, ELMo, and GPT-2 Embeddings. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 55–65, Hong Kong, China. Association for Computational Linguistics.

Angela Fan, Finale Doshi-Velez, and Luke Miratrix. 2019. Assessing topic model relevance: Evaluation and informative priors. *Statistical Analysis and Data Mining: The ASA Data Science Journal*, 12(3):210–222.

Karl Moritz Hermann, Edward Grefenstette, and Phil Blunsom. 2013. "not not bad" is not "bad": A distributional account of negation. In *Proceedings of the Workshop on Continuous Vector Space Models and their Compositionality*, pages 74–82, Sofia, Bulgaria. Association for Computational Linguistics.

John Hewitt and Christopher D. Manning. 2019. A Structural Probe for Finding Syntax in Word Representations. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4129–4138, Minneapolis, Minnesota. Association for Computational Linguistics.

Larry Horn. 1972. *On the Semantic Properties of Logical Operators in English*. Ph.D. thesis, UCLA.

James R. Hurford. 1974. Exclusive or inclusive disjunction. *Foundations of Language*, pages 409–411.

Samuel Huston and W. Bruce Croft. 2010. Evaluating verbose query processing techniques. In *Proceedings of the 33rd International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '10, page 291–298, New York, NY, USA. Association for Computing Machinery.

Michela Ippolito. 2020. Varieties of Sobel sequences. *Linguistics and Philosophy*, pages 633–671.

Ganesh Jawahar, Benoît Sagot, and Djamé Seddah. 2019. What Does BERT Learn about the Structure of Language? In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 3651–3657, Florence, Italy. Association for Computational Linguistics.

Lauri Karttunen and Stanley Peters. 1979. Conventional implicature. In C.K. Oh and D. Dineen, editors, *Presupposition*, pages 1–56. Academic Press, New York.

Nora Kassner and Hinrich Schütze. 2020. Negated and misprimed probes for pretrained language models: Birds can talk, but cannot fly. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7811–7818, Online. Association for Computational Linguistics.

William A. Ladusaw. 1996. Negation and polarity items. In S. Lappin, editor, *The Handbook of Contemporary Linguistics Theory*, pages 321–341. Blackwell, Oxford.

Zhenzhong Lan, Mingda Chen, Sebastian Goodman, Kevin Gimpel, Piyush Sharma, and Radu Soricut. 2019. ALBERT: A lite BERT for self-supervised learning of language representations. *CoRR*, abs/1909.11942.

Yongjie Lin, Yi Chern Tan, and Robert Frank. 2019. Open Sesame: Getting inside BERT's Linguistic Knowledge. In *Proceedings of the 2019 ACL Workshop BlackboxNLP: Analyzing and Interpreting Neural Networks for NLP*, pages 241–253, Florence, Italy. Association for Computational Linguistics.

Godehard Link. 2002. The logical analysis of plurals and mass terms: A lattice-theoretical approach. In P. Portner and B. Partee, editors, *Formal semantics: The essential readings*, pages 127–146. Blackwell, Oxford / Malden, MA.

Tal Linzen, Emmanuel Dupoux, and Benjamin Spector. 2016. Quantificational features in distributional word representations. In *Proceedings of the Fifth Joint Conference on Lexical and Computational Semantics*, pages 1–11, Berlin, Germany. Association for Computational Linguistics.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Roberta: A robustly optimized BERT pretraining approach. *CoRR*, abs/1907.11692.

H. P. Luhn. 1960. Key word-in-context index for technical literature (kwic index). *American Documentation*, 11(4):288–295.

Rebecca Marvin and Tal Linzen. 2018. Targeted Syntactic Evaluation of Language Models. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 1192–1202, Brussels, Belgium. Association for Computational Linguistics.

Tom McCoy, Ellie Pavlick, and Tal Linzen. 2019. Right for the Wrong Reasons: Diagnosing Syntactic Heuristics in Natural Language Inference. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 3428–3448, Florence, Italy. Association for Computational Linguistics.

Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013. Efficient Estimation of Word Representations in Vector Space. *Proceedings of Workshop at ICLR*.

Yixin Nie, Adina Williams, Emily Dinan, Mohit Bansal, Jason Weston, and Douwe Kiela. 2020. Adversarial NLI: A new benchmark for natural language understanding. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 4885–4901, Online. Association for Computational Linguistics.

Jeffrey Pennington, Richard Socher, and Christopher D. Manning. 2014. Glove: Global Vectors for Word Representation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1532–1543.

Matthew Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer. 2018. Deep Contextualized Word Representations. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 2227–2237, New Orleans, Louisiana. Association for Computational Linguistics.

Fabio Petroni, Tim Rocktäschel, Sebastian Riedel, Patrick Lewis, Anton Bakhtin, Yuxiang Wu, and Alexander Miller. 2019. Language models as knowledge bases? In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 2463–2473, Hong Kong, China. Association for Computational Linguistics.

Kyle Richardson, Hai Hu, Lawrence S. Moss, and Ashish Sabharwal. 2020. Probing Natural Language Inference Models through Semantic Fragments. In *Association for the Advancement of Artificial Intelligence (AAAI)*, pages 8713–8721. AAAI Press.

John R Ross. 1972. The category squish: Endstation hauptwort. In *Chicago Linguistic Society*, volume 8, pages 316–328.

Serhad Sarica and Jianxi Luo. 2020. Stopwords in technical language processing. *CoRR*, abs/2006.02633.

Uli Sauerland. 2004. Scalar Implicatures in Complex Sentences. *Linguistics and Philosophy*, pages 367–391.

Rita Sevastjanova, Aikaterini-Lida Kalouli, Christin Beck, Hanna Schäfer, and Mennatallah El-Assady. 2021. Explaining Contextualization in Language Models using Visual Analytics. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 464–476, Online. Association for Computational Linguistics.

Raj Singh. 2008. On the interpretation of disjunction: Asymmetric, incremental, and eager for inconsistency. *Linguistics and Philosophy*, pages 245–260.

Robyn Speer, Joshua Chin, and Catherine Havasi. 2016. Conceptnet 5.5: An open multilingual graph of general knowledge. *CoRR*, abs/1612.03975.

Alon Talmor, Yanai Elazar, Yoav Goldberg, and Jonathan Berant. 2020. oLMpics-on what language model pre-training captures. *Transactions of the Association for Computational Linguistics*, 8:743–758.

Gongbo Tang, Gaoqi Rao, Dong Yu, and Endong Xun. 2016. Can we neglect function words in word embedding? In *Natural Language Understanding and Intelligent Applications - 5th CCF Conference on Natural Language Processing and Chinese Computing, NLPCC 2016, and 24th International Conference on Computer Processing of Oriental Languages, ICCPOL 2016, Kunming, China, December 2-6, 2016, Proceedings*, volume 10102 of *Lecture Notes in Computer Science*, pages 541–548. Springer.

Ian Tenney, Dipanjan Das, and Ellie Pavlick. 2019a. BERT Rediscovers the Classical NLP Pipeline. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 4593–4601, Florence, Italy. Association for Computational Linguistics.

Ian Tenney, Patrick Xia, Berlin Chen, Alex Wang, Adam Poliak, R Thomas McCoy, Najoung Kim, Benjamin Van Durme, Sam Bowman, Dipanjan Das, and Ellie Pavlick. 2019b. What do you learn from context? Probing for sentence structure in contextualized word representations. In *International Conference on Learning Representations*.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Remi Louf, Morgan Funtowicz, Joe Davison, Sam Shleifer, Patrick von Platen, Clara Ma, Yacine Jernite, Julien Plu, Canwen Xu, Teven Le Scao, Sylvain Gugger, Mariama Drame, Quentin Lhoest, and Alexander Rush. 2020. Transformers: State-of-the-art natural language processing. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 38–45, Online. Association for Computational Linguistics.

A Appendix A: Sample of Datasets

		Examples
Inconsistent Dataset	negation	Cairo is not located in [MASK]. Cairo is located in [MASK].
		A guitar does not have [MASK]. A guitar has [MASK].
		A chair has no [MASK]. A chair has [MASK].
		Maria is a mother without a [MASK]. Maria is a mother with a [MASK].
		A cat sees without [MASK]. A cat sees with [MASK].
	coord.	Joe is a dolphin or an [MASK]. Joe is a dolphin and an [MASK].
		Tina is a bird or an [MASK]. Tina is a bird and an [MASK].
	quantifiers	All cars have an [MASK]. No car has an [MASK].
		All cooks [MASK]. No cook [MASK].
		Some shoes have [MASK]. All shoes have [MASK].
Semantic Dataset	syn. negation	A mom is not a [MASK]. **forbidden**: mom, mother, grandmother, grandma, granddaughter, bride, wife, woman, niece, stepmother, daughter, aunt
		A designer does not [MASK]. **forbidden**: design
		A guitar does not have [MASK]. **forbidden**: strings
	lex. negation	A bird flies without [MASK]. **forbidden**: wings
		John is a father without a [MASK]. **forbidden**: child
		Peter is a brother without a [MASK]. **forbidden**: sibling
	coordination	Mark was born in Athens or in [MASK]. **forbidden**: Greece
		Tina died in Beijing or in [MASK]. **forbidden**: China
		George comes from Berlin or [MASK]. **forbidden**: Germany
		Kate is a cat or an [MASK]. **forbidden**: animal

Table 3: Sample sentences from each dataset and each subset. The complete datasets can be found under: https://function-words.lingvis.io/.

Tales and Tropes: Gender Roles from Word Embeddings in a Century of Children's Books*

Anjali Adukia[1], Patricia Chiril[1], Callista Christ[1], Anjali Das[2]
Alex Eble[2], Emileigh Harrison[1], Hakizumwami Birali Runesha[1]
(1) University of Chicago
{adukia, pchiril, callistac, harrisone, runesha}@uchicago.edu
(2) Columbia University
ad3888@columbia.edu, eble@tc.columbia.edu

Abstract

The manner in which gender is portrayed in materials used to teach children conveys messages about people's roles in society. In this paper, we measure the gendered depiction of central domains of social life in 100 years of highly influential children's books. We make two main contributions: (1) we find that the portrayal of gender in these books reproduces traditional gender norms in society, and (2) we publish StoryWords 1.0, the first word embeddings trained on such a large body of children's literature. We find that, relative to males, females are more likely to be represented in relation to their appearance than in relation to their competence; second, they are more likely to be represented in relation to their role in the family than their role in business. Finally, we find that non-binary or gender-fluid individuals are rarely mentioned. Our analysis advances understanding of the different messages contained in content commonly used to teach children, with immediate applications for practice, policy, and research.

1 Introduction

Educators and parents use books to teach children messages about society, conduct, and the world. These messages may be encoded in how different identities are, and are not, represented. If there are systematically different associations between specific identities and depictions, such messages can shape how children view the roles that they themselves, as well as others, can occupy in society. In this paper, we apply Natural Language Processing (NLP) tools to analyze the gendered association of different attributes (e.g., *traits*, *occupations*, *physical characteristics*) to measure how females and males are portrayed in children's books.[1]

To measure portrayal, we use word embeddings, a prediction-based method for analyzing the semantic meaning of words in context using high-dimensional vectors. We supplement our analysis by training a model to detect individual sentences containing stereotypes in order to gain a deeper understanding of the implicit and explicit messages conveyed to children by the books they read. This awareness can, in turn, also help inform content-selection decisions of educators and caregivers.

Messages about gender-specific abilities and roles may influence children's beliefs and career paths (Leslie et al., 2015; Riley, 2017; Bian et al., 2017, 2018). Gender representation in children's content has traditionally been measured by manual content analysis, in which one or multiple annotators slowly read through the text of written content to classify the messages within one or multiple dimensions (Neuendorf, 2016). The key advantage of this approach is that it is able to measure deep meaning in books; the main disadvantage is that it is highly labor-intensive, making it prohibitively costly to comprehensively characterize representation in large bodies of content, and requires a high degree of fidelity in the management and training of the coders (Krippendorff, 2018). It is difficult to avoid human biases in this type of traditional content analysis, though these biases are of course also baked into any content analysis, including computerized approaches (Buolamwini and Gebru, 2018).

Advances in computer-driven content analysis began to address these concerns through automation. Early efforts focused on a numerical accounting of words which represented different genders – such as counts of pronouns and the genders of named entities – and these counts were then compared across bodies of text (Krippendorff, 2018;

*All the authors contributed equally to this work.

[1]As gender is not a binary construct, we wished to include characters who identified as non-binary or gender fluid, in addition to those who identified as female or male. In an analysis of a subset of books which center LGBTQIA+ experiences, we only found two characters with non-binary identities (0.37% of total characters). Because there would not be a sufficient sample size to estimate embeddings for this group, we limit our main analysis to females and males.

Proceedings of the 29th International Conference on Computational Linguistics, pages 3086–3097
October 12–17, 2022.

Gentzkow et al., 2019). Simple token counts, however, primarily capture superficial representation. If a female or male is frequently present but portrayed in a stereotypical or narrow manner, then the mere existence of representation will not only be insufficient but also possibly counterproductive.

In this paper, we address this gap by using word vectors to measure *how* different genders are depicted, *vis-a-vis* societal roles, in English-language, award-winning children's books commonly found in schools and homes over the past century, complementing existing measurement of *whether* they appear (Adukia et al., 2022). This involves converting high-dimensional measures of the semantic meaning of words in text into one-dimensional measures of gender representation in children's books. Our study makes two primary contributions:

(1) We apply established NLP tools to a policy-relevant body of text with clear implications for child development and education in order to understand how roles are differentially portrayed by gender (cf. Section 5). Specifically, we examine the gender associations of societal domains such as *appearance* vs. *competence*, *family* vs. *business*, and *female* vs. *male professions*.

(2) We release a word embeddings dataset trained on our sample of award-winning children's books (named the `StoryWords 1.0` dataset) so that other researchers can use these data (cf. Section 3).[2]

How different identities are portrayed in these books has the potential to shape children's beliefs about themselves as well as their beliefs about others, which affects their effort in school, future educational decisions, and later life outcomes. Our work also demonstrates how NLP tools can be used to measure the messages contained in bodies of text being considered for use in curricular settings. This has clear and immediate applications for both the practice of education and for research on the linkages between the content of books and on the educational outcomes of children exposed to them.

2 Related Work

External stimuli may have important influences in shaping beliefs, actions, and outcomes (Bian et al., 2017; Bordalo et al., 2017; Rodríguez-Planas and Nollenberger, 2018). For example, historical analysis of changes in textbooks using a quasi-experimental framework has shown that such changes shape both people's preferences and their view of history (Fuchs-Schündeln and Masella, 2016; Cantoni et al., 2017). Less is known about the representation of identities in the content in these books and how these identities are depicted.

Recent work has attempted to address this question by estimating the frequency of female and male presence in stories. Research enumerating gender counts in children's books shows inequality in how frequently females are present in the text relative to males over time regardless of the measure, for example, in gendered pronouns and in the gender of named characters (Adukia et al., 2022). While these findings are illustrative, they show only superficial representations and neglect to demonstrate whether the trend towards numeric equality is inclusive or rather one of an increased incidence of imbalanced representations. If the frequency of inclusion of underrepresented identities increases without a change in the underlying equity in the manner of representation, simple frequency-based measures might overstate the equity of representation in books that children are given.

Recent work has addressed how characters of different genders are portrayed. Xu et al. (2019) analyzes female characters' emotional dependency on male characters in a collection of books, movie synopses and movie scripts. That study defines narratives in which a man serves as a woman's path to a happy, fulfilling life as characterized by the '*Cinderella complex*'. Using pretrained word2vec models, they constructed a vector representing the dimension of *happy* vs. *unhappy* that was used for calculating the *'happiness scores'* of words surrounding specific female and male characters. They first selected the movie synopsis of Cinderella; calculating happiness scores for it, the study shows that the happiness of Cinderella depends on the prince, but not vice versa. Further testing on different movie genres showed that the happiness score of the female characters portrayed in the same context as male characters was higher than when the females were portrayed alone. They also find that male characters are more likely to be described using verbs, while female characters are more likely to be described using adjectives.

[2]The data and associated code are available at: `https://github.com/miielab/GenderEmbeddingsPaper`

2.1 Gender in Language Models

Word embeddings have become one of the most used types of features in many NLP models and are widely used for a variety of downstream tasks. However, these word representations have been proven to reflect social biases (such as race and gender) inherited from data used to train them (Caliskan et al., 2017). To automatically quantify these biases, several fairness metrics (i.e., functions that measure the association degree between target and attribute words in a word embedding model) have been proposed in the past few years (Caliskan et al., 2017; Garg et al., 2018; Sweeney and Najafian, 2019; Ethayarajh et al., 2019; Dev and Phillips, 2019). More recently, researchers have started quantifying, analyzing and mitigating the gender bias exhibited by contextualized embeddings (Zhao et al., 2019; Kurita et al., 2019; Tan and Celis, 2019; Guo and Caliskan, 2021). Their results show that contextualized word models inherit human-like biases, which are then propagated to downstream tasks.

2.2 Gender Stereotypes in Social Sciences

Gender stereotypes are defined by the Office of the High Commissioner for Human Rights (OHCHR) as *'a generalised view or preconception about attributes, or characteristics that are or ought to be possessed by women and men or the roles that are or should be performed by men and women'*.[3]

One significant consequence of gender stereotypes is the reinforcement of gender inequality; within this framework, *agency* (i.e., traits such as competence and independence) and *communion* (i.e., concerns about the welfare of others and relationship with them) are the core dimensions used to characterize gender stereotypes. Although biological attributes may impact a person's behaviour and choice of occupational roles, research indicates that gender differences in beliefs about gender stereotypes develop over time, and that they are influenced by family, friends and education (Dhar et al., 2018; Eble and Hu, 2020, 2022). For example, one set of gender stereotypes posits that women are communal, kind and family oriented, whereas men are more agentic, skilled and work oriented (Ellemers, 2018).

In light of changes in the positions occupied by women in society, as well as the broadening of opportunities presented to women, Haines et al. (2016) characterize the extent to which gender stereotypes have changed between 1983 and 2014. In that study, participants assessed the likeliness of a set of gendered characteristics (e.g., *traits*, *behaviours*, *occupations*, *physical characteristics*) to belong to a typical man or woman, similar to the methods used by Deaux and Lewis (1984). The study assessed whether people's beliefs changed over time in parallel with changes in society. They also measured whether these beliefs vary by age, measuring this for people from 19 to 73 years of age, as opposed to the college students studied in Deaux and Lewis (1984). Surprisingly, the authors find no indication of a substantial change in basic stereotypes over time in spite of many relevant societal changes.

Although widely studied in psychology, communication studies and social science (Allport et al., 1954; Crawford et al., 2002; Beike and Sherman, 2014; Biscarrat et al., 2016), in NLP, gender stereotypes have been studied mainly to detect or remove gender bias in word embeddings or word association graphs (Bolukbasi et al., 2016; Park et al., 2018; Madaan et al., 2018; Dev and Phillips, 2019; Du et al., 2019) as well as to identify disparity across gender in various applications like co-reference resolution (Zhao et al., 2018) and sentiment analysis (Felmlee et al., 2019). A notable exception is the work by Chiril et al. (2021) who use gender stereotypes detection as an auxiliary task to improve sexism classification.

3 Data

3.1 Primary Data: Children's Books

School libraries and classrooms serve as major purveyors of sanctioned literary content for children. The books they offer are accompanied by an implicit state-sanctioned stamp-of-approval. These books are chosen because their content is perceived to be appropriate for children. They are often intended to transmit clear narratives about appropriate conduct, an account of important historical moments, or other, often identity-specific messages.

We draw from a set of children's books written in English that are likely to be found in U.S. school libraries – namely, those that received awards administered or featured by the Association for Library Service to Children, a division of the American Library Association. Out of the 3,447 books that either won an award or received an honourable

[3]Source: https://www.ohchr.org/en/women/gender-stereotyping, accessed September 14, 2022.

mention, we were able to collect and digitize a sample of 1,130 books using both library and online resources.

In order to understand whether representation differs depending on the focus of efforts to highlight different kinds of books, we divide these award-winning corpora into two *collections*: the `Mainstream` collection and the `Diversity` collection. Figure 1 shows the sample size of each collection by decade.

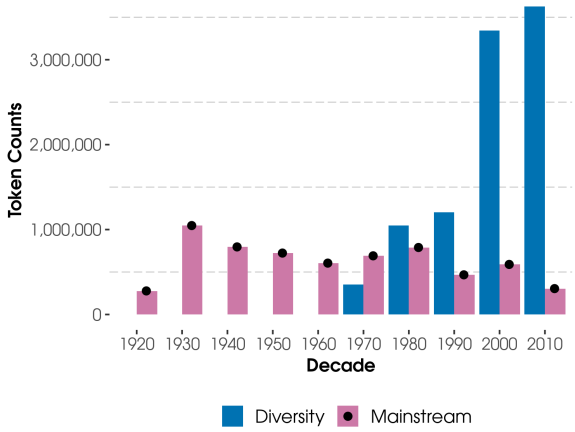

Figure 1: The sample size of the `Mainstream` and `Diversity` collections over time. The aggregate number of words in the `Mainstream` collection is 6,289,116 words and in the `Diversity` collection is 9,599,638 words.

Mainstream Collection. The `Mainstream` collection comprises books that have received recognition through the Newbery or Caldecott Medals, the two oldest children's book awards in the United States starting in the 1920s to present day. These books are selected for their perceived contribution to children's literature and not popularity. Receipt of the award facilitates the book's entry into the canon of U.S. children's literature (Smith, 2013; Koss et al., 2018). These books are all in English, but are likely to be translated into other languages.

Diversity Collection. To examine how purposeful efforts to highlight typically excluded or marginalized identities perform, we draw from books likely to be placed on *'diversity lists'* such as during Black History Month or Women's History Month.[4] Awards in this collection were first distributed in 1970, with a gradual rollout of different awards over the following decades.

We compare the estimates for the `Mainstream` and `Diversity` collections to examine whether intentional efforts to highlight underrepresented identities more equitably portray females and males compared to unintentional, *'general'* efforts. We hypothesize that books which are recognized for highlighting one underrepresented identity may also highlight other underrepresented identities.

3.2 Data Collection and Pre-processing

We use Google Vision Optical Character Recognition (OCR) to extract text from scanned pages of each children's book.[5] Note that this process is restricted to the conversion of scanned text into ASCII characters. A manual error analysis on a random sample of 10 children's books shows that the average Word Error Rate (WER) of the text extracted using OCR was 2.62%. Since our sample of children's books contains many illustrations, most of the error can be attributed to random characters added to the extracted text when the OCR software mistook a shape in a illustration as an ASCII character.

Once the text is extracted, we pre-process the data to reduce variability and noise. We first divide each award corpus into sentences using the pre-trained Punkt tokenizer from Python's NLTK library (Bird et al., 2009). For each sentence, we then lowercase the text and remove digits, line breaks, punctuation, and special characters.[6]

Our goal is to characterize how females and males have been overall represented in each collection of books, as well as how this representation has changed over time. We therefore combine the data at two levels: (1) at the *collection level*, in order to measure overall representations between each of the collections, and (2) at the *collection-by-decade level*, to measure changes over time.

[4]Specifically, we examine books that have received recognition from the following awards: American Indian Youth Literature, Américas, Arab American, Asian/Pacific American Award for Literature, Carter G. Woodson, Coretta Scott King, Dolly Gray, Ezra Jack Keats, Middle East, Notable Books for a Global Society, Pura Belpré, Rise Feminist, Schneider Family, Skipping Stones Honor, South Asia, Stonewall, and Tomás Rivera Mexican American Book Awards.

[5]Source: https://cloud.google.com/vision/docs/ocr

[6]We refrain from removing stopwords because a preliminary inspection of the sensitivity of our results to the inclusion or exclusion of stopwords prior to the learning process showed that our results remain similar.

3.3 Supplemental Data: HistWords

In addition to the children's books, we incorporate data from the `HistWords` dataset, a collection of books gathered from over 40 university libraries containing more than 361 billion English words (Michel et al., 2011). These books span from 1800 to 2000 and contain text from of a variety of genres.[7] We include these data as a numeraire, capturing the representations of females and males across the last two centuries in books intended for adult consumption, rather than children's consumption. Because the only publicly available data for `HistWords` is in the form of word2vec embeddings, we directly incorporate the embeddings they provide in our final visualizations rather than running the lexicon through our pipeline, as outlined in Section 4.[8]

3.4 Gender as a Non-binary or Fluid Construct

Our main goal is to measure the relationship between different gender groups and societal domains. Gender is not a binary construct and can comprise females, males, non-binary, and gender-fluid individuals. However, as far as we are aware, there is no systematic way of measuring non-binary or gender-fluid identities in off-the-shelf NLP packages such as those we use. Instead, to evaluate the presence of non-binary identities in our data, we manually search for non-binary characters in a set of books that received awards for centering LGBTQIA+ experiences (i.e., Stonewall Book Awards), which we expect would have a greater representation of individuals who identify as non-binary or gender-fluid. This exercise entails manually coding the gender for each human named entity (e.g., *Mary*) mentioned more than once in each book as measured by spaCy's Named Entity Recognition software (Honnibal and Montani, 2017). The manually coded gender labels are: *male*, *female*, *non-binary*, or *unknown*. We use context clues within the books to determine gender. These context clues include the character's own identification or, if the character's input is absent, pronouns and gendered descriptions (e.g., *Character X was a woman.*). In the absence of sufficient context clues, we label the gender as *unknown*.

Out of 539 named human entities, only two characters identified as non-binary (0.37%). As a result, the sample size in our data would be insufficient to accurately and precisely estimate embeddings for non-binary and gender-fluid identities. As a result, performing the computational methods used in this paper on non-binary groups, separate from females and males, would not yield reliable results. In light of this, we focus our analysis on only females and males, though we note that this extremely small proportion of explicitly non-binary characters in the collection of books most likely to represent them is an important finding in itself.

4 Methodology

4.1 StoryWords 1.0

We use word embeddings to capture the ways in which gender is represented in these texts. Word embeddings operate under the assumption that words which appear in similar contexts have similar meanings (Firth, 1951; Harris, 1954). In practice, word embeddings are generated by neural networks which map each word to a high-dimensional vector representation of that word. Each word vector encapsulates semantic and syntactic information by incorporating information from the nearest neighbors (context) of that word. Word embeddings permit analysis between sets of vectors, including calculating similarity measurements between words using cosine distance.[9]

We use the word2vec package from Python's Gensim library to estimate word embeddings (Rehurek and Sojka, 2010).[10] Our word2vec implementation uses the Skip-Gram with Negative Sampling (SGNS) model architecture introduced by Mikolov et al. (2013).[11] When setting the hyperparameters of our models, we followed the recommendations of Levy et al. (2015). After training, the algorithm outputs 300-dimensional vectors of every word in the lexicon of each book. We train separate word2vec models on the *aggregate collection* data as well as on the *collection-*

[7] We limit analysis of `HistWords` starting in the 1920s as the first book in our corpus was published in the 1920s.

[8] The aggregate model across all decades for the `HistWords` collection is not publicly available. We discuss how we estimate `HistWords` collection-level measures for word embeddings in Section 4.1.

[9] Word embeddings are categorized as prediction-based embeddings because they use Machine Learning to predict context words.

[10] While we show results from the implementation of word2vec, our results are similar when we use GloVe (Pennington et al., 2014).

[11] We chose the SGNS architecture as it outperforms other architectures on various linguistic tasks. It is fast to train and inexpensive in terms of memory consumption and disk space (Levy et al., 2015).

by-decade data (cf. Section 3.2). Because aggregate measures are not available at the collection level for `HistWords`, we average the measures for `HistWords` for each decade starting from the 1920s through the 1990s to estimate an overall measure for this collection and are not able to calculate statistics to generate an overall measure.

Each time a word2vec model is trained with the exact same hyperparameters, word neighborhoods may change, which can generate different embedding estimates for each round of training (Hellrich and Hahn, 2016; Antoniak and Mimno, 2018; Burdick et al., 2018). To minimize the influence of idiosyncratic variation in shaping our results, we train 50 separate word2vec models with identical hyperparameters on the *collection-by-decade* and *aggregate collection* data (cf. Section 3.2).[12] We name the resulting embeddings dataset `StoryWords 1.0`. We make this available on our GitHub.

4.2 Word Embedding Association Tests

Group (Gender) Words. We develop a vocabulary of words that comprise two gender groups (*females*, *males*). The words associated with females and males were generated by drawing upon commonly used words for each category, in addition to incorporating words from sources such as those lists provided by Caliskan et al. (2017) and Senel et al. (2018). We fine-tune the categories to the linguistic particularities of the domain of children's literature by incorporating vocabulary that is commonly used in these books. For example, words such as *princess* and *king* are included in our gender group word lists, but are not in prior group lexicons, such as those in Caliskan et al. (2017) and Garg et al. (2018).[13] Our lexicon includes 71 pairs of (*females*, *males*) words. Each word within a given category is exclusive to that category only. The final list of gendered words can be found on our GitHub.

Domain Words. We seek to understand how females and males are depicted within these children's books in relation to different attributes (e.g., *traits*, *behaviours*, *occupations*, *physical characteristics*). The choice of these attributes is based on their importance for children's beliefs and perceptions of themselves and others. Each of these is commonly portrayed in children's literature (Nodelman, 2008; Rudd, 2012; Beauvais, 2015).

Our empirical analysis follows Caliskan et al. (2017), supplemented with analysis of whether females are more associated with descriptions of appearance and related terms than males. Our decision to add this analysis follows prior research indicating that men are often described by words that pertain to behaviour, whereas women are typically described by adjectives that refer to their physical appearance and sexuality (Caldas-Coulthard and Moon, 2010).

We constructed our final word lists as follows. We first began with the domain lists provided by Caliskan et al. (2017) and Senel et al. (2018). We then manually augmented them by drawing upon a set of commonly used words for each domain category: *appearance* (93 words; e.g., *alluring*, *elegant*), *competence* (93 words such as *persuasive*, *reasonable*), *family* (39 words), and *professions* (340 words; e.g., *dancer*, *educator* for women; *architect*, *professor* for men).[14] Our augmentation of these lists was performed through ConceptNet (Speer et al., 2017), a multilingual knowledge graph for natural language words or phrases in their undisambiguated forms. The final list of domain words can be found on our GitHub.

Each word within a given category is exclusive to that category only. For example, the *family* category is notably smaller than other lists because many 'family' words are gendered and therefore were included in the *male/female* lists instead of the *family* list.

4.3 Gender Stereotype Detection

While word embedding association tests help us understand gender stereotypes in collections of text, it is also important to be able to identify specific stereotypes found in individual books so parents can make informed decisions about which books are appropriate for their children. A recently published report shows that nearly two-thirds of preteenagers in America read for pleasure at least once a week (Rideout et al., 2022). As a parent, monitoring what content their children do or do not consume is difficult without information to guide

[12] Because there is only one embedding model published for each decade of the `HistWords` dataset, we cannot perform this exercise on the `HistWords` data.

[13] Our choice of gendered vocabulary is over 3 times as large as the gendered word lists used in Garg et al. (2018), who use 20 male words and 20 female words, and approximately 9 times larger than the gendered word lists in Caliskan et al. (2017), who use 8 male words and 8 female words.

[14] We used the occupation census data provided by Garg et al. (2018).

their decisions. Information on gender stereotypes in specific books can be obtained, for example, by using online platforms such as Common Sense Media. In a preliminary exploration of this resource, we found that only 25% of the books included in our corpus have a review.

To automatically identify potential stereotypical topics, we employ SentenceBERT, a modification of BERT that derives semantic sentence embeddings that can be compared using cosine similarity (Reimers and Gurevych, 2019). We leverage three manually annotated corpora with gender stereotype information from previous studies: the Automatic Misogyny Identification (AMI) dataset collection from both `IberEval` (Fersini et al., 2018b) and `Evalita` (Fersini et al., 2018a)), and the dataset released by Chiril et al. (2021). We selected these datasets as they are freely available to the research community.[15] This method classifies a sentence as containing a stereotype if the cosine similarity between the sentence and another sentence which has already been labeled as containing a stereotype is higher than a threshold (T). For our analysis, we apply SentenceBERT to a subset of six books from our corpora of children's books that had commentary in the *What Parents Need to Know* section of Common Sense Media reviews. This section highlights topics that may be of particular concern to parents, flagging content to which they might not want their kids being exposed. We experimentally set the threshold to $T = 0.45$.

5 Results and Discussion

Word Embeddings. For conducting the experiments, we relied on the Word Embedding Fairness Evaluation (WEFE) framework (Badilla et al., 2020), an open source software that encapsulates, evaluates and compares different fairness metrics proposed in the literature. Here, we present the results obtained by using the WEAT metric from Caliskan et al. (2017), the most commonly used association test for word embeddings.[16] WEAT assesses the extent to which a model associates two sets of target words (i.e., *females* and *males*) with sets of attribute words (i.e., *appearance* and *competence*, *family* and *business*, *female* and *male professions*). With values that can range between -2.0 and 2.0, a positive score means that `females` are more associated with words related to `appearance`, `family` or `female professions`, and a negative score means that `males` are more associated with the aforementioned attributes.

The representation and visibility of women has increased substantially over the last century, including in occupations that have been traditionally dominated by men (Goldin, 2014). Despite this considerable progress, differential treatment of women in many dimensions of the economy persists into the 21st century (Blau and Kahn, 2017). Our analysis reflects these patterns, highlighting the incidence and persistence of professional role stereotyping in these corpora (cf. Figure 2 (a)). In addition, our results also highlight that females are more likely to be associated with words related to family than words related to business, relative to these likelihoods for males (cf. Figure 2 (b)). While the association between females and family appears to decrease slightly in the `HistWords` data over time, we see no evidence of a similar decline in our children's book collections (c.f. Figure 2 (e)).

Finally, we quantify the degree to which the language used to describe females and males is different. Figure 2 (c) shows that females are much more likely to be associated with words related to their looks (as opposed to males, who tend to be associated with words related to their competence). This association between females and appearance is decreasing over time in the `HistWords` and `Mainstream` collections, but increases in the most recent decades within the children's book collections.

These results show that all three collections (i.e., `Diversity`, `Mainstream`, and `HistWords`) contain biased representations. We found no evidence that the `Diversity` collection, meant to highlight typically excluded or marginalized identities, portrays females more equitably than the 'general' efforts of the `Mainstream` collection.

Stereotype Detection. We next apply the method for detecting particularly salient incidences of gender stereotypes in text, as described in Section 4.3. First, we access reviews from the Common Sense Media platform to identify books that are likely to possess highly gender stereotypical language. We then apply stereotype detection to iden-

[15] These datasets contain tweets that are annotated at different granularity levels. While the AMI corpora only indicates the presence of a stereotype, the dataset released by Chiril et al. (2021) offers a finer characterization.

[16] Looking across results from the 50 models, we observe a small amount of variation in our results, both for the aggregate and over time measures. In light of this, our results report averages over the 50 models.

(a) Female vs. Male Professions (b) Family vs. Business (c) Appearance vs. Competence

(d) Female vs. Male Professions (e) Family vs. Business (f) Appearance vs. Competence

Figure 2: Which domains are *females* more associated with relative to *males*?

Note: We present the WEAT effect sizes, which show whether `females`, as compared to `males`, are more associated with one set of attribute words relative to another. We present these both overall in panels (a)-(c), and over time in panels (d)-(f). Panels (a) and (d) show `female professions` relative to `male professions`; panels (b) and (e) show `family` relative to `business`; panels (c) and (f) show `appearance` relative to `competence`.

tify specific sentences associated with the themes that were highlighted in the review and that present egregious cases of this type of messaging.

Here we present two sentences from a book in our corpora that this method identifies as containing such stereotypes. This book lends a commentary on rape culture.:

(1) *And I know you're on your period, but there's no need to get cranky with me. ($T = 0.552$)*

(2) *If I had boobs like that, I'd wear a burka or something. ($T = 0.466$)*

Sentence (1) appears to reinforce the stereotype of the hysterical menstrual woman, while sentence (2) discriminates against women based on the widely held belief that the exposure of the chest is a sexual act. It is important to acknowledge that this method does not take into account the context surrounding these sentences when identifying them as containing stereotypes. These detected sentences, together with the results from our word embeddings analysis, provide us with evidence of how highly gender stereotypical messages can appear in the text of children's books, even those recognized in highly prominent national book awards.

Future work should more deeply interrogate the stereotypes being transmitted in these children's books, as highlighted by these few examples. Such work should also include an exploration of the context of such phrases.

6 Conclusion

In this paper we demonstrate how NLP tools can be used to investigate the incidence of systematically different associations between females and males and their societal roles, as transmitted through children's stories. These findings underscore the importance of tracking not only whether different identities are included in stories, but also how they are portrayed. We make two primary contributions. First, we analyzed how gender roles are portrayed

in children's literature. Second, we created the first word embeddings trained on a century of award-winning children's literature, `StoryWords 1.0`. Consistent with previous research, our results show that females are more likely than males to be represented in relation to their appearance and their roles in the family.

While we cannot speak to what 'optimal' representation would look like, our tools make it possible for practitioners, policymakers, and parents with a given goal to measure representation in a given set of books in order to help them make their choices.

Important directions for future work include using more precise tools, such as coreference resolution, to better understand and disentangle the indirect and direct messages contained in these texts. In addition, researchers or practitioners using these tools could expand their analysis to other targets: different groups (to understand how other identities may be differentially represented), as well as additional attributes that convey different societal meanings. Furthermore, researchers must expand definitions of gender to account for non-binary and gender-fluid identities. In the future, we also plan to account for polysemous words by using contextualized word vectors.

Ethical Approval. The research reported in this article involved no human participants and so no human subjects review was sought. Our use of the text data in the children's books in our study is transformative, analyzing the books' content and transforming it, via this analysis, into separate and distinct data, which we conduct under the fair use principle.

An important limitation is that our measure of gender representation binarizes gender, constraining it as *female* and *male*, and does not account for non-binary or gender-fluid identities. This comes as a direct result of the low number of characters identified as non-binary (cf. Section 3.4). With respect to the stereotypical language identified through the sentence similarity approach, we make no claims about the intentions of the authors of these books. The context of these phrases remains to be explored in future work.

Future work should innovate to address these challenges and spur new developments in this under-explored area.

Acknowledgements

We thank Celia Anderson for her contributions to the early stages of this project. For excellent research assistance, we thank Simon Mahns, Noah McLean, Qurat ul ain, Khemraj Shukla, and Charlie Wang. For helpful feedback, we thank Kevin Gimpel, Allyson Ettinger, Chenhao Tan, and others. For financial support, we thank Becker-Friedman Institute at UChicago, Center for Data and Applied Computing at UChicago, Data Science Institute at UChicago, National Academy of Education, and Spencer Foundation. The research reported here was also supported by the Institute of Education Sciences, U.S. Department of Education, through Grant R305A200478 to the University of Chicago. The opinions expressed are those of the authors and do not represent views of the Institute or the U.S. Department of Education. We acknowledge the University of Chicago Research Computing Center for support of this work.

References

Anjali Adukia, Alex Eble, Emileigh Harrison, Hakizumwami Birali Runesha, and Teodora Szasz. 2022. What we teach about race and gender: Representation in images and text of children's books. *NBER Working Paper 29123*.

Gordon Willard Allport, Kenneth Clark, and Thomas Pettigrew. 1954. The nature of prejudice.

Maria Antoniak and David Mimno. 2018. Evaluating the stability of embedding-based word similarities. *Transactions of the Association for Computational Linguistics*, 6:107–119.

Pablo Badilla, Felipe Bravo-Marquez, and Jorge Pérez. 2020. Wefe: The word embeddings fairness evaluation framework. In *IJCAI*, pages 430–436.

Clémentine Beauvais. 2015. *The Mighty Child: Time and Power in Children's Literature*, volume 4. John Benjamins Publishing Company.

Denise R Beike and Steven J Sherman. 2014. Social inference: Inductions, deductions, and analogies. *Handbook of social cognition*, pages 209–285.

Lin Bian, Sarah-Jane Leslie, and Andrei Cimpian. 2017. Gender stereotypes about intellectual ability emerge early and influence children's interests. *Science*, 355(6323):389–391.

Lin Bian, Sarah-Jane Leslie, and Andrei Cimpian. 2018. Evidence of bias against girls and women in contexts that emphasize intellectual ability. *American Psychologist*, 73(9):1139.

Steven Bird, Ewan Klein, and Edward Loper. 2009. *Natural language processing with Python: Analyzing text with the natural language toolkit*. O'Reilly Media, Inc.

Laurence Biscarrat, Marlène Coulomb-Gully, and Cécile Méadel. 2016. One is not born a female CEO and...won't become one! *Gender equality and the media - a challenge for Europe. Routledge, ECREA Book Series*.

Francine D Blau and Lawrence M Kahn. 2017. The gender wage gap: Extent, trends, and explanations. *Journal of Economic Literature*, 55(3):789–865.

Tolga Bolukbasi, Kai-Wei Chang, James Zou, Venkatesh Saligrama, and Adam Kalai. 2016. Man is to computer programmer as woman is to homemaker? Debiasing word embeddings. In *Proceedings of the 30th International Conference on Neural Information Processing Systems*, pages 4356–4364.

Pedro Bordalo, Nicola Gennaioli, and Andrei Shleifer. 2017. Memory, attention, and choice. *The Quarterly Journal of Economics*.

Joy Buolamwini and Timnit Gebru. 2018. Gender shades: Intersectional accuracy disparities in commercial gender classification. *Conference on Fairness, Accountability and Transparency*, pages 77–91.

Laura Burdick, Jonathan K Kummerfeld, and Rada Mihalcea. 2018. Factors influencing the surprising instability of word embeddings. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 2092–2102.

Carmen Rosa Caldas-Coulthard and Rosamund Moon. 2010. 'Curvy, hunky, kinky': Using corpora as tools for critical analysis. *Discourse & Society*, 21(2):99–133.

Aylin Caliskan, Joanna J. Bryson, and Arvind Narayanan. 2017. Semantics derived automatically from language corpora contain human-like biases. *Science*, 356(6334):183–186. Publisher: American Association for the Advancement of Science.

Davide Cantoni, Yuyu Chen, David Y. Yang, Noam Yuchtman, and Y. Jane Zhang. 2017. Curriculum and ideology. *Journal of Political Economy*, 125(2):338–392.

Patricia Chiril, Farah Benamara, and Véronique Moriceau. 2021. "Be nice to your wife! The restaurants are closed": Can Gender Stereotype Detection Improve Sexism Classification? In *Findings of the Association for Computational Linguistics: EMNLP 2021*, pages 2833–2844.

Matthew T Crawford, Steven J Sherman, and David L Hamilton. 2002. Perceived entitativity, stereotype formation, and the interchangeability of group members. *Journal of Personality and Social Psychology*, 83(5):1076.

Kay Deaux and Laurie L Lewis. 1984. Structure of gender stereotypes: Interrelationships among components and gender label. *Journal of Personality and Social Psychology*, 46(5):991.

Sunipa Dev and Jeff M. Phillips. 2019. Attenuating bias in word vectors. In *The 22nd International Conference on Artificial Intelligence and Statistics, AISTATS 2019*, pages 879–887.

Diva Dhar, Tarun Jain, and Seema Jayachandran. 2018. Intergenerational transmission of gender attitudes: Evidence from India. *Journal of Development Studies*, pages 1–21.

Yupei Du, Yuanbin Wu, and Man Lan. 2019. Exploring human gender stereotypes with word association test. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing, EMNLP-IJCNLP 2019, Hong Kong, China, November 3-7, 2019*, pages 6132–6142. Association for Computational Linguistics.

Alex Eble and Feng Hu. 2020. Child beliefs, societal beliefs, and teacher-student identity match. *Economics of Education Review*, 77:101994.

Alex Eble and Feng Hu. 2022. Gendered beliefs about mathematics ability transmit across generations through children's peers. *Nature Human Behaviour*, pages 1–12.

Naomi Ellemers. 2018. Gender stereotypes. *Annual Review of Psychology*, 69:275–298.

Kawin Ethayarajh, David Duvenaud, and Graeme Hirst. 2019. Understanding undesirable word embedding associations. *arXiv preprint arXiv:1908.06361*.

Diane Felmlee, Paulina Inara Rodis, and Amy Zhang. 2019. Sexist slurs: Reinforcing feminine stereotypes online. *Sex Roles*, pages 1–13.

Elisabetta Fersini, Debora Nozza, and Paolo Rosso. 2018a. Overview of the Evalita 2018 task on Automatic Misogyny Identification (AMI). In *Proceedings of the Sixth Evaluation Campaign of Natural Language Processing and Speech Tools for Italian. Final Workshop (EVALITA 2018) co-located with the Fifth Italian Conference on Computational Linguistics (CLiC-it 2018), Turin, Italy, December 12-13, 2018*, volume 2263 of *CEUR Workshop Proceedings*. CEUR-WS.org.

Elisabetta Fersini, Paolo Rosso, and Maria Anzovino. 2018b. Overview of the task on Automatic Misogyny Identification. In *Proceedings of the Third*

Workshop on Evaluation of Human Language Technologies for Iberian Languages (IberEval 2018) colocated with 34th Conference of the Spanish Society for Natural Language Processing (SEPLN 2018), Sevilla, Spain, September 18th, 2018, volume 2150 of *CEUR Workshop Proceedings*, pages 214–228. CEUR-WS.org.

John Rupert Firth. 1951. *Papers in Linguistics (1934–1951)*. Oxford University Press, Oxford, UK.

Nicola Fuchs-Schündeln and Paolo Masella. 2016. Long-lasting effects of Socialist education. *Review of Economics and Statistics*, 98(3):428–441.

Nikhil Garg, Londa Schiebinger, Dan Jurafsky, and James Zou. 2018. Word embeddings quantify 100 years of gender and ethnic stereotypes. *Proceedings of the National Academy of Sciences*, 115(16):E3635–E3644.

Matthew Gentzkow, Jesse Shapiro, and Matt Taddy. 2019. Measuring group differences in high-dimensional choices: Method and application to Congressional speech. *Econometrica*, 87(4):1307–1340.

Claudia Goldin. 2014. A grand gender convergence: Its last chapter. *American Economic Review*, 104(4):1091–1119.

Wei Guo and Aylin Caliskan. 2021. Detecting emergent intersectional biases: Contextualized word embeddings contain a distribution of human-like biases. In *Proceedings of the 2021 AAAI/ACM Conference on AI, Ethics, and Society*, pages 122–133.

Elizabeth L Haines, Kay Deaux, and Nicole Lofaro. 2016. The times they are a-changing… or are they not? A comparison of gender stereotypes, 1983–2014. *Psychology of Women Quarterly*, 40(3):353–363.

Zellig S Harris. 1954. Distributional structure. *Word*, 10(2-3):146–162.

Johannes Hellrich and Udo Hahn. 2016. Bad company—neighborhoods in neural embedding spaces considered harmful. In *Proceedings of COLING 2016, The 26th International Conference on Computational Linguistics: Technical papers*, pages 2785–2796.

M Honnibal and I Montani. 2017. Natural language understanding with Bloom embeddings, convolutional neural networks and incremental parsing. *Unpublished software application*.

Melanie D Koss, Nancy J Johnson, and Miriam Martinez. 2018. Mapping the diversity in Caldecott books from 1938 to 2017: The changing topography. *Journal of Children's Literature*, 44(1):4–20.

Klaus Krippendorff. 2018. *Content Analysis: An Introduction to Its Methodology*. Sage publications.

Keita Kurita, Nidhi Vyas, Ayush Pareek, Alan W Black, and Yulia Tsvetkov. 2019. Measuring bias in contextualized word representations. In *Proceedings of the First Workshop on Gender Bias in Natural Language Processing*, pages 166–172.

Sarah-Jane Leslie, Andrei Cimpian, Meredith Meyer, and Edward Freeland. 2015. Expectations of brilliance underlie gender distributions across academic disciplines. *Science*, 347(6219):262–265.

Omer Levy, Yoav Goldberg, and Ido Dagan. 2015. Improving distributional similarity with lessons learned from word embeddings. *Transactions of the Association for Computational Linguistics*, 3:211–225.

Nishtha Madaan, Sameep Mehta, Taneea Agrawaal, Vrinda Malhotra, Aditi Aggarwal, Yatin Gupta, and Mayank Saxena. 2018. Analyze, detect and remove gender stereotyping from Bollywood movies. In *Proceedings of the 1st Conference on Fairness, Accountability and Transparency*, pages 92–105.

Jean-Baptiste Michel, Yuan Kui Shen, Aviva Presser Aiden, Adrian Veres, Matthew K. Gray, , Joseph P. Pickett, Dale Hoiberg, Dan Clancy, Peter Norvig, Jon Orwant, Steven Pinker, Martin A. Nowak, and Erez Lieberman Aiden. 2011. Quantitative analysis of culture using millions of digitized books. *Science*, 331(6014):176–182.

Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. 2013. Distributed representations of words and phrases and their compositionality. *Advances in Neural Information Processing Systems*, 26.

Kimberly A. Neuendorf. 2016. *The Content Analysis Guidebook*. Sage.

Perry Nodelman. 2008. *The Hidden Adult: Defining Children's Literature*. JHU Press.

Ji Ho Park, Jamin Shin, and Pascale Fung. 2018. Reducing gender bias in abusive language detection. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 2799–2804.

Jeffrey Pennington, Richard Socher, and Christopher D Manning. 2014. Glove: Global vectors for word representation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1532–1543.

Radim Rehurek and Petr Sojka. 2010. Software framework for topic modelling with large corpora. In *In Proceedings of the LREC 2010 workshop on new challenges for NLP frameworks*. ELRA.

Nils Reimers and Iryna Gurevych. 2019. Sentence-BERT: Sentence Embeddings using Siamese BERT-Networks. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 3982–3992.

Victoria Rideout, Alanna Peebles, Supreet Mann, and Michael Robb. 2022. Common sense census: Media use by tweens and teens 2021. *Common Sense Media*.

Emma Riley. 2017. Increasing students' aspirations: The impact of Queen of Katwe on students' educational attainment. In *CSAE Working Paper WPS/2017-13*.

Núria Rodríguez-Planas and Natalia Nollenberger. 2018. Let the girls learn! It is not only about math... It's about gender social norms. *Economics of Education Review*, 62:230–253.

David Rudd. 2012. *The Routledge Companion to Children's Literature*. Routledge.

Lutfi Kerem Senel, Ihsan Utlu, Veysel Yucesoy, Aykut Koc, and Tolga Cukur. 2018. Semantic structure and interpretability of word embeddings. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*.

Vicky Smith. 2013. The "Caldecott effect". *Children Libraries: The Journal of the Association for Library Service to Children*, 1(1):9–13.

Robyn Speer, Joshua Chin, and Catherine Havasi. 2017. Conceptnet 5.5: An open multilingual graph of general knowledge. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 31.

Chris Sweeney and Maryam Najafian. 2019. A transparent framework for evaluating unintended demographic bias in word embeddings. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 1662–1667.

Yi Chern Tan and L. Elisa Celis. 2019. Assessing social and intersectional biases in contextualized word representations. In *Advances in Neural Information Processing Systems 32: Annual Conference on Neural Information Processing Systems 2019, NeurIPS 2019, December 8-14, 2019, Vancouver, BC, Canada*, pages 13209–13220.

Huimin Xu, Zhang Zhang, Lingfei Wu, and Cheng-Jun Wang. 2019. The Cinderella complex: Word embeddings reveal gender stereotypes in movies and books. *PloS One*, 14(11).

Jieyu Zhao, Tianlu Wang, Mark Yatskar, Ryan Cotterell, Vicente Ordonez, and Kai-Wei Chang. 2019. Gender bias in contextualized word embeddings. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 629–634.

Jieyu Zhao, Tianlu Wang, Mark Yatskar, Vicente Ordonez, and Kai-Wei Chang. 2018. Gender bias in coreference resolution: Evaluation and debiasing methods. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 2 (Short Papers)*, pages 15–20, New Orleans, Louisiana. Association for Computational Linguistics.

CLOWER: A Pre-trained Language Model with Contrastive Learning over Word and Character Representations

Borun Chen[1]*, Hongyin Tang[2]*, Jiahao Bu[2], Kai Zhang[2], Jingang Wang[2]†, Qifan Wang[3], Hai-Tao Zheng[1,4]†, Wei Wu[2] and Liqian Yu[2]

[1]Tsinghua Shenzhen International Graduate School, Tsinghua University
[2]Meituan [3]Meta AI [4]Peng Cheng Laboratory

cbr20@mails.tsinghua.edu.cn zheng.haitao@sz.tsinghua.edu.cn
{tanghongyin,wangjingang02,wuwei30,yuliqian}@meituan.com
bujh1994@gmail.com zhangkai.plus@foxmail.com wqfcr@fb.com

Abstract

Pre-trained Language Models (PLMs) have achieved remarkable performance gains across numerous downstream tasks in natural language understanding. Various Chinese PLMs have been successively proposed for learning better Chinese language representation. However, most current models use Chinese characters as inputs and are not able to encode semantic information contained in Chinese words. While recent pre-trained models incorporate both words and characters simultaneously, they usually suffer from deficient semantic interactions and fail to capture the semantic relation between words and characters. To address the above issues, we propose a simple yet effective PLM CLOWER, which adopts the Contrastive Learning Over Word and charactER representations. In particular, CLOWER implicitly encodes the coarse-grained information (i.e., words) into the fine-grained representations (i.e., characters) through contrastive learning on multi-grained information. CLOWER is of great value in realistic scenarios since it can be easily incorporated into any existing fine-grained based PLMs without modifying the production pipelines. Extensive experiments conducted on a range of downstream tasks demonstrate the superior performance of CLOWER over several state-of-the-art baselines.

1 Introduction

Pre-trained language models (PLMs) have gained tremendous success in the field of natural language processing recently. As a major milestone of PLMs, BERT (Devlin et al., 2019) and its variants (Yang et al., 2019; Liu et al., 2019; Clark et al., 2019) have demonstrated outstanding performance on a wide variety of natural language understanding (NLU) tasks, such as sentiment analysis and machine reading comprehension tasks. The architecture of Transformer (Vaswani et al., 2017) is typically the foundation for these models, which models the semantic and syntactic relationships between the tokens of the entire input text and learns the contextual representations for each token.

Early Chinese PLMs (Sun et al., 2019) often take the sequences of Chinese characters as the input. These models require relatively small vocabulary and learn the representations of each character from the corpus, which avoids the Out-Of-Vocabulary problem (Li et al., 2019). However, the meanings of a Chinese word can be totally different from the meanings of each Chinese character in the word. For example, the meaning of "小心" (careful) can not be derived from summing the meaning of "小" (small) and "心" (heart). In general, the phenomenon of semantic gaps between coarse-grained language units and fine-grained language units (e.g., words & characters, phrases & words) exists not only in Chinese but also in many other languages.

To alleviate the gap, prior studies improve the pre-trained models in two directions. One direction is to enrich the masking strategies in the masked language model (MLM) objective to mask coarse-grained units, such as the whole word masking (WWM) (Cui et al., 2021) and phrase masking (Sun et al., 2019). These methods encourage the pre-trained model to recover the coarse-grained masks with fine-grained tokens. However, the relation between the coarse-grained and fine-grained representations is modeled in an implicit manner, leading to less effective representations. The other direction is to leverage the multi-grained tokenizations as input. AMBERT (Zhang et al., 2021) encodes both the fine-grained and coarse-grained token sequences and performs the masked language modeling tasks correspondingly, while LICHEE (Guo et al., 2021) merges the multi-grained token embeddings explicitly to integrate the information. Lattice-BERT (Lai et al., 2021) adopts the lattice graph to construct the multi-grained input. Nevertheless, these mod-

*Equal contribution.
†Corresponding authors.

Proceedings of the 29th International Conference on Computational Linguistics, pages 3098–3108
October 12–17, 2022.

els require additional computational costs (e.g., tokenization, graph construction, multi-grained encoding) and the multi-grained information is only integrated in the embedding layer other than the full encoder, leading to limited usability with low effectiveness.

To fully leverage the semantic information of multi-granularity and preserve the flexibility of single-grained models in the fine-tuning stage, we propose a novel PLM named CLOWER to efficiently model the multi-grained semantic information in pre-training to improve the representation capability. CLOWER adopts the contrastive learning framework to carry out the semantic interaction between multi-grained representations. Specifically, in the pre-training stage, we perform both character and word level tokenizations separately for each input sequence and feed them into the encoder to obtain the contextual representations. Then we conduct the contrastive learning over character and word representations on both token-level and sentence-level. In this way, the word-level semantic information is encoded into the character tokens by bringing their representations closer. Different from AMBERT or LICHEE, in fine-tuning, CLOWER requires no additional computation and can be directly used in any fine-grained PLMs. The merit makes CLOWER production-friendly since it could be deployed easily without modifying the established production pipeline.

We perform comprehensive experiments on different downstream NLU tasks. The experimental results show that CLOWER achieves considerable improvements over several baselines. Ablation studies demonstrate the effectiveness of contrastive learning in our pre-training framework. Our contributions are summarized as follows:

- We present a novel approach that adopts contrastive learning over both word and character representations, which effectively captures their semantic relations.
- With the help of the aforementioned contrastive learning approach, we introduce a Chinese pre-trained language model that connects multi-grained semantic information for learning high quality word and character encoders.
- We conduct an extensive set of experiments on several benchmarks and demonstrate the effectiveness of the proposed model.

2 Related Work

Multi-grained Pre-trained Language Models There have been some efforts to explore the multi-granularity information on the pre-trained language models (Tay et al., 2021; Xue et al., 2022). Cui et al. (2021) adopts the whole word masking strategy to select the masking tokens for pre-training. Similarly, ERNIE 1.0 and 2.0 (Sun et al., 2019, 2020), utilize named entity masking and phrase masking to encode the coarse-grained information into the models, while ERNIE-Gram (Xiao et al., 2021) uses explicit n-gram identities as predicted targets for the enhancement with coarse-grained information. Besides, Joshi et al. (2020) propose the SpanBERT to mask text spans and train the span boundary objective. However, these methods mainly concentrate on fine-grained tokens. The coarse-grained information is only implicitly explored in the masked language modeling by designing the masking strategies and the coarse-grained representations are absent.

Instead of designing the coarse-grained masking strategy on the fine-grained token sequences, several methods focus on improving the pre-training models with multi-grained tokenization. AMBERT (Zhang et al., 2021) utilizes two encoders with shared parameters to process the fine-grained and coarse-grained token sequences. LICHEE (Guo et al., 2021) proposes to merge the multi-grained tokenizations at the embedding level to incorporate multi-grained information of input. Recently, Lai et al. (2021) propose the Lattice-BERT, which introduces the lattice graph constructed from characters and words to explicitly explore the word representations in a multi-granularity way. However, these models are either computationally intensive or lack the integration of multi-grained information in the deep encoder layers, resulting in the limitations of usability and effectiveness.

Contrastive Learning in Pre-trained Language Models As contrastive learning become popular in visual representation learning (Chen et al., 2020; He et al., 2020; Khosla et al., 2020) and NLP tasks (Wu et al., 2020; Meng et al., 2021; Wang et al., 2021), there have been several works exploring the effects of contrastive learning for pre-trained language models. CERT (Fang et al., 2020) adopts the framework of MOCO (He et al., 2020) and performs the sentence augmentations by back-translation. Zhang et al. (2020) propose

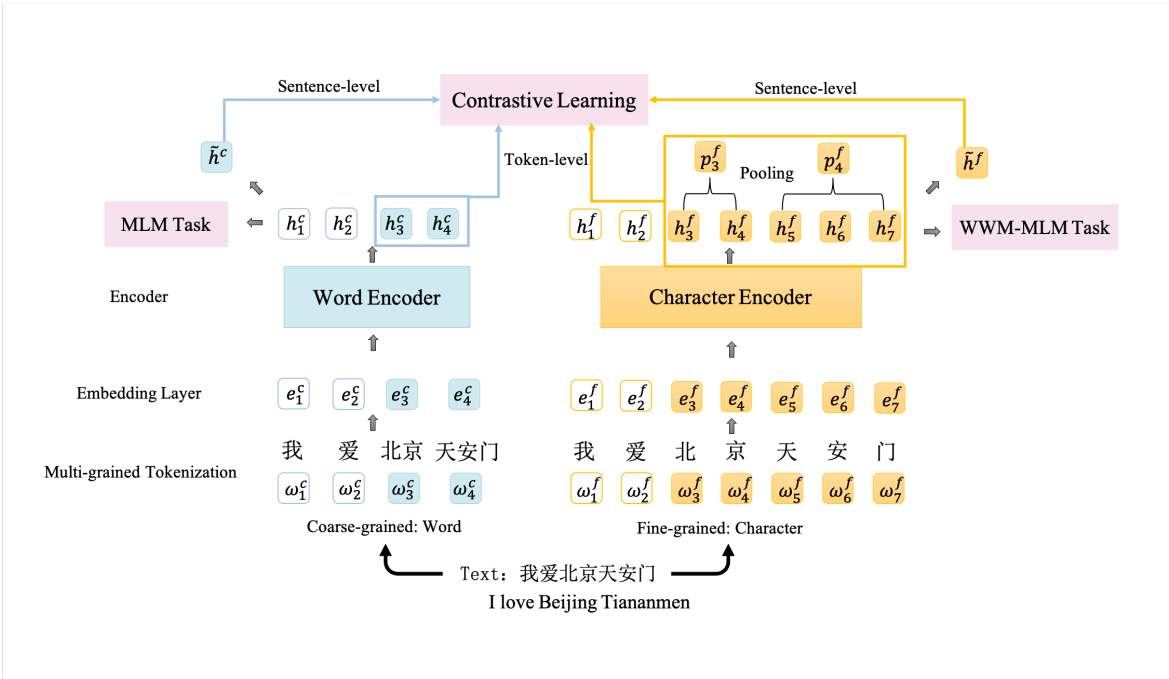

Figure 1: An overview of CLOWER. Fine-grained and coarse-grained representations are encoded by two encoders.Token-level and sentence-level contrastive learning are conducted together with the MLM and WWM-MLM tasks.

the unsupervised sentence embedding model IS-BERT, increasing the mutual information between the global representations and the local context when training the model. ConSERT (Yan et al., 2021) applies a variety of data augmentation techniques to generate various input views at the embedding level for contrastive learning. Similarly, SimCSE (Gao et al., 2021) utilizes dropout acts as data augmentation in sentence-level. The above methods conduct the contrastive learning to fine-tune the pre-trained language encoder. As for pre-training the language model, DeCLUTR (Giorgi et al., 2021) and CLEAR (Wu et al., 2020) utilize the architecture of SimCLR (Chen et al., 2020) to combine the contrastive learning objective with the masked language modeling. Compared to the above models, our CLOWER conducts the contrastive learning over word and character representations in pre-training and we have the flexibility to fine-tune it in specific downstream tasks.

3 Methodology

In this section, we present CLOWER, the pre-trained language model based on contrastive learning over word and character representations. We first present the overall model architecture of CLOWER, and then we introduce its details in the pre-training stage. Finally, we discuss the strategy of fine-tuning the model efficiently using only the fine-grained input.

3.1 Model Architecture

Figure 1 illustrates an overview of CLOWER pre-training, where the contrastive learning framework is leveraged across multiple granularity information to enhance the representation ability of the model.

CLOWER takes the text sequences as input and performs multi-grained tokenization on the input to obtain the fine-grained and coarse-grained token sequences. It should be noted that the fine-grained and coarse-grained tokens share the same vocabulary, which aims at improving the alignment of embedding spaces between multi-grained tokens. In this paper, we treat the characters and words as fine-grained and coarse-grained tokens respectively. Formally, given the input text sequence s, we denote the fine-grained and coarse-grained token sequences by $\boldsymbol{s_f} = \{\omega_1^f, \cdots, \omega_i^f, \cdots, \omega_m^f\}$ and $\boldsymbol{s_c} = \{\omega_1^c, \cdots, \omega_j^c, \cdots, \omega_n^c\}$, where m and n are the lengths of two tokenized sequences.

Consistent with the shared vocabulary, CLOWER adopts the shared embedding layers to map the tokens ω_i^f and ω_j^c to the embedding representations $\boldsymbol{e_i^f}$ and $\boldsymbol{e_j^c} \in \mathcal{R}^d$ respectively, where d is the dimension of the embedding. The

fine-grained and coarse-grained embeddings are then passed to the two encoders to obtain the contextualized representations respectively. The encoders utilized in CLOWER can be any pre-trained language model and two encoders of fine-grained and coarse-grained have independent parameters. In this paper, we adopt Chinese BERT(Devlin et al., 2019) as the encoders.

Token-level and sentence-level contrastive learning are conducted over the fine-grained and coarse-grained contextualized representations from the above encoders, together with the traditional MLM task and WWM-MLM task.

3.2 Pre-Training

Masked Language Model In the pre-training stage, CLOWER adopts the MLM task at multi-grained levels. Specifically, we denote the masked fine-grained and coarse-grained token sequences as $\tilde{\boldsymbol{s_f}}$ and $\tilde{\boldsymbol{s_c}}$. The masked fine-grained and coarse-grained tokens are represented as $\boldsymbol{s_f^m}$ and $\boldsymbol{s_c^m}$ respectively. Then, the object of our MLM task at multi-grained levels is to optimize the following loss function:

$$\begin{aligned}\mathcal{L}_{mlm} = &- \sum_{\omega_f^m \in \boldsymbol{s_f^m}} \log P_\theta(\omega_f^m | \tilde{\boldsymbol{s_f}}) \\ &- \sum_{\omega_c^m \in \boldsymbol{s_c^m}} \log P_\theta(\omega_c^m | \tilde{\boldsymbol{s_c}}),\end{aligned} \tag{1}$$

where θ denotes the model parameters.

We adopt the WWM strategy (Cui et al., 2021) as the strategy of fine-grained token sequences and the conventional masking strategy introduced by BERT(Devlin et al., 2019) for the coarse-grained token sequences.

Contrastive Learning To fully learn from the multi-grained information, we conduct contrastive learning between the fine-grained representations and their corresponding coarse-grained representations at both token-level and sentence-level. Formally, for each pair of multi-grained token sequences $\boldsymbol{s_f}$ and $\boldsymbol{s_c}$, we randomly choose some of the coarse-grained tokens $\boldsymbol{s_a} = \{\omega_1^c, \cdots, \omega_i^c, \cdots, \omega_k^c\} \subset \boldsymbol{s_c}$ as anchors, where k is the maximum number of anchors for each sequence. The strategy of selecting the anchors will be detailed in Section 4.1.

Given the anchor ω_i^c, which is composed of the fine-grained tokens $\omega_{b(i)}^f, \cdots, \omega_{e(i)}^f$ where $b(i)$ denotes the begin index of the anchor ω_i^c and $e(i)$ denotes the end index of the anchor ω_i^c, we can obtain its coarse-grained representation $\boldsymbol{h_i^c}$ generated by the word encoder and its fine-grained representation $\boldsymbol{p_i^f} = \text{AVG}\left(\boldsymbol{h_{b(i)}^f} \cdots \boldsymbol{h_{e(i)}^f}\right)$ generated by the character encoder, where $\text{AVG}(\cdot)$ means the average pooling.

Our motivation is to close the gap between the fine-grained representations and their corresponding coarse-grained representations while enlarge the gap between unrelated representations. Following the contrastive learning paradigm, it can be implemented by constructing positive and negative instance pairs. For the coarse-grained representation $\boldsymbol{h_i^c}$, we mark the fine-grained representations of the same anchor $\boldsymbol{p_i^f}$ as its positive instance and the fine-grained representations of the other anchors in the same mini-batch $\boldsymbol{p_j^f}$ as the negative instances. We further introduce the "`[CLS]`" embeddings of each sentence as the sentence-level representations, namely $\tilde{\boldsymbol{h}}^{\boldsymbol{c}}$ for the coarse-grained representation and $\tilde{\boldsymbol{h}}^{\boldsymbol{f}}$ for the fine-grained representation. Similar to the token-level, we treat the multi-grained representations $(\tilde{\boldsymbol{h}}^{\boldsymbol{c}}, \tilde{\boldsymbol{h}}^{\boldsymbol{f}})$ of the same sentence as the positive instance pair and the multi-grained representations of different sentences in a mini-batch as the negative instance pairs.

Following the contrastive objective in Chen et al. (2020), we utilize the normalized temperature-scaled cross-entropy loss (NT-Xent) for both the token-level and sentence-level representations. We optimize the symmetric cross-entropy loss in the pre-training. Specifically, the objective of contrastive learning in multi-grained token-level representations $\mathcal{L}_{tcl}$ is as follows:

$$\mathcal{L}_{tcl}^c = -\frac{1}{N}\sum_{i=1}^{N} \log \frac{e^{\text{sim}(\boldsymbol{h_i^c}, \boldsymbol{p_i^f})/\tau}}{\sum_j e^{\text{sim}(\boldsymbol{h_i^c}, \boldsymbol{p_j^f})/\tau}}, \tag{2}$$

$$\mathcal{L}_{tcl}^f = -\frac{1}{N}\sum_{i=1}^{N} \log \frac{e^{\text{sim}(\boldsymbol{p_i^f}, \boldsymbol{h_i^c})/\tau}}{\sum_j e^{\text{sim}(\boldsymbol{p_i^f}, \boldsymbol{h_j^c})/\tau}}, \tag{3}$$

$$\mathcal{L}_{tcl} = \frac{1}{2}(\mathcal{L}_{tcl}^c + \mathcal{L}_{tcl}^f), \tag{4}$$

where N indicates the number of in-batch anchors, $\text{sim}(\cdot)$ denotes the similarity function as we use the cosine similarity, and τ is a temperature hyper-parameter. Similarly, we define the symmetric sentence-level contrastive loss $\mathcal{L}_{scl}$ with a mini-

batch size M as:

$$\mathcal{L}_{scl}^{c} = -\frac{1}{M}\sum_{i=1}^{M}\log\frac{e^{\text{sim}(\tilde{\boldsymbol{h}}_i^{\boldsymbol{c}},\tilde{\boldsymbol{h}}_i^{\boldsymbol{f}})/\tau}}{\sum_j e^{\text{sim}(\tilde{\boldsymbol{h}}_i^{\boldsymbol{c}},\tilde{\boldsymbol{h}}_j^{\boldsymbol{f}})/\tau}}, \quad (5)$$

$$\mathcal{L}_{scl}^{f} = -\frac{1}{M}\sum_{i=1}^{M}\log\frac{e^{\text{sim}(\tilde{\boldsymbol{h}}_i^{\boldsymbol{f}},\tilde{\boldsymbol{h}}_i^{\boldsymbol{c}})/\tau}}{\sum_j e^{\text{sim}(\tilde{\boldsymbol{h}}_i^{\boldsymbol{f}},\tilde{\boldsymbol{h}}_j^{\boldsymbol{c}})/\tau}}, \quad (6)$$

$$\mathcal{L}_{scl} = \frac{1}{2}(\mathcal{L}_{scl}^{c} + \mathcal{L}_{scl}^{f}), \quad (7)$$

Therefore, the final object of contrastive learning $\mathcal{L}_{con}$ is the sum of $\mathcal{L}_{tcl}$ and $\mathcal{L}_{scl}$.

Sentence Order Prediction Apart from the MLM and contrastive learning tasks, we adopt the sentence order prediction (SOP) task (Lan et al., 2019) to effectively model the relationship of sentence pairs and denote the training loss as $\mathcal{L}_{sop}$. Hence, the overall training loss of CLOWER in pre-training is the combination of three tasks:

$$\mathcal{L} = \mathcal{L}_{mlm} + \lambda\mathcal{L}_{sop} + \mu\mathcal{L}_{con} \quad (8)$$

where λ and μ are the hyper-parameters of balancing three task objectives.

3.3 Fine-Tuning

Note that the usage of the character encoder of CLOWER is virtually the same as the fine-grained Chinese PLMs like BERT, thus we can directly substitute them with our character encoder without any modification while having the benefit of the coarse-grained information encoded in the fine-grained representations.

For the sentence-level downstream tasks, like single sentence classification and sentence pair classification, we conduct classification base on the contextualized sentence-level representation $\tilde{\boldsymbol{h}}^{\boldsymbol{f}}$. As for the token-level tasks, such as question answering, fine-grained contextualized representations of each token are extracted and used for predictions.

4 Experiments

We conducted comprehensive experiments on various Chinese NLU tasks to examine the effectiveness of CLOWER. In this section, we first introduce the details of pre-training and fine-tuning, including the datasets and experimental settings. Then, we present the overall results on different tasks and conduct an in-depth analysis. Ablation studies are also conducted to evaluate the impact of multi-level contrastive learning in our model.

Dataset	MSL	BS	LR	Epoch
ChnSentiCorp	256	32	3e-5	10
THUCNews	512	16	3e-5	10
Tnews	128	32	3e-5	10
Bq Corpus	128	64	3e-5	10
Lcqmc	128	64	3e-5	10
Ocnli	128	32	3e-5	10
Xnli	128	64	3e-5	10
CMRC2018	512	8	3e-5	5
DRCD	512	8	3e-5	5

Table 1: Hyper-parameters settings for 9 fine-tuning tasks. MSL: Maximum Sequence Length; BS: Batch Size; LR: Learning Rate.

4.1 Pre-training Datasets

To the best of our knowledge, WuDaoCorpora (Yuan et al., 2021) is the largest open-source Chinese corpora for pre-training. We utilize the base version of WuDaoCorpora[1], consisting of about 200GB training data and 72 billion Chinese characters in total. Following the settings of most Chinese PLMs, we consider the characters as the fine-grained tokens. We utilize Jieba[2] to perform the word segmentation on texts and the segmented words are treated as the coarse-grained tokens. There are $5,466$ Chinese characters and $40,014$ words in our vocabulary, together with other tokens like digits and some basic English tokens. We conduct the fine-grained and coarse-grained tokenizations based on the vocabulary and the words will be split to characters if they are not in the vocabulary. For contrastive learning, we select up to k anchors whose lengths are between 2 and 4 from each sequence. Note that for semantic integrity, the words that have been masked either on coarse-grained sequences or their fine-grained characters will not be selected as anchors.

4.2 Fine-tuning tasks

To thoroughly examine the effectiveness of CLOWER, an extensive set of experiments are performed on various Chinese NLU tasks, including three single sentence classification (SSC) tasks, four sentence pair classification (SPC) tasks and two machine reading comprehension(MRC) tasks. Specifically, three SSC tasks are ChnSentiCorp (Tan and Zhang, 2008), THUCNews (Li

[1] https://resource.wudaoai.cn/home
[2] https://github.com/fxsjy/jieba

Model	Tnews	THUCNews		ChnSentiCorp		Average
	Dev	Dev	Test	Dev	Test	
BERT-wwm	66.59	98.16	97.41	94.97	95.55	90.53
BERT-wwm-sop	66.42	98.31	97.49	94.87	95.32	90.48
MM-BERT	66.39	98.18	97.53	94.92	94.80	90.36
MM-BERT-sop	66.27	98.16	97.45	94.62	95.65	90.43
MacBERT	67.07	98.29	97.34	95.16	95.18	90.61
CLOWER	**67.15**	**98.39**	**97.74**	**95.18**	**95.84**	**90.86**

Table 2: Experimental results on single sentence classification tasks.

Model	Ocnli	Lcqmc		Xnli		Bq		Average
	Dev	Dev	Test	Dev	Test	Dev	Test	
BERT-wwm	74.87	89.37	86.93	79.50	78.89	85.39	84.37	82.76
BERT-wwm-sop	75.73	89.75	87.30	79.74	78.45	85.73	84.81	83.07
MM-BERT	75.34	89.55	87.08	79.56	78.66	85.37	84.51	82.87
MM-BERT-sop	75.44	89.85	87.18	79.42	78.62	86.00	84.84	83.05
MacBERT	75.90	89.58	86.59	**80.54**	79.10	85.71	84.95	83.20
CLOWER	**76.25**	**89.92**	**88.10**	80.14	**79.19**	**86.01**	**85.26**	**83.55**

Table 3: Experimental results on sentence pair classification tasks.

Model	CMRC2018		DRCD			
	Dev		Dev		Test	
	EM	F1	EM	F1	EM	F1
BERT-wwm	68.15	86.32	88.20	93.63	87.13	92.55
BERT-wwm-sop	67.47	85.86	87.54	93.15	87.33	92.61
MM-BERT	68.61	86.42	88.45	93.65	87.36	92.85
MM-BERT-sop	67.57	86.18	88.30	93.50	87.18	92.76
MacBERT	68.31	86.38	**88.92**	**94.08**	**88.04**	**93.22**
CLOWER	**68.73**	**86.52**	88.27	93.44	87.68	92.94

Table 4: Experimental results on MRC tasks.

and Sun, 2007) and Tnews (Xu et al., 2020); four SPC tasks include Bq Corpus (Chen et al., 2018), Lcqmc (Liu et al., 2018), Ocnli (Hu et al., 2020) and Xnli (Conneau et al., 2018); two MRC tasks are CMRC2018 (Cui et al., 2019) and DRCD (Shao et al., 2018).

4.3 Experiment Settings

4.3.1 Pre-training

In pre-training of CLOWER, we initiate both the character and word encoder with the Chinese BERT-base released by Google[3] in order to reduce the total convergence time. Given a word not in the vocabulary, we initiate its embedding with the average pooling of the embeddings of the characters that make up the word. For MLM tasks, as with the BERT, 15% of the tokens are masked randomly. For token-level contrastive learning, the maximum number of anchors for each sequence is set as 20 and the temperature is 0.05. The hyper-parameters λ and μ in Equation 8 are both set as 1. We set the maximum sequence length to 512 throughout the pre-training and adopt the ADAM (Kingma and Ba, 2014) optimizer with weight decay whose learning rate is $2e-5$. We train the model with a batch size of 960 (24×40) for $300,000$ steps. The pre-training is carried out on 40 NVIDIA V100 GPUs. To improve efficiency, mixed precision training (Micikevicius et al., 2017) is adopted.

4.3.2 Fine-tuning

To make a fair comparison, we adopt the same hyper-parameters for each fine-tuning task among different models. The detailed parameter settings are shown in Table 1. During fine-tuning, we encode each example using the fine-grained encoder (i.e., character encoder). For three SSP tasks and four SPC tasks, the "`[CLS]`" embedding is used to represent the sentence and the classification accuracies are reported. For two MRC tasks, the token embeddings are used to extract the answer

[3] https://github.com/google-research/bert

Figure 2: Ablation Results. We report the accuracy for sentence classification tasks and EM for MRC tasks.

span from the sentence, and both exact match (EM) and F1-score are reported. For each task, we perform the experiments five runs with different random seeds and report the average performance to promise the results convincing. We report the results both on the development sets and test sets, except for Tnews, Ocnli and CMRC2018, whose test sets are not publicly available. Since each article in the Tnews dataset consists of a title and several keywords, we associate the titles with keywords as the input sequences to perform the classification task. We fine-tune all the models for each downstream task on one NVIDIA V100 GPU.

4.4 Main Results

Since most of the existing Chinese PLMs are trained with different corpus and setups, it is hard to conduct ideally fair comparisons. Therefore, we select the most representative Chinese PLM (i.e., Chinese BERT-base) as the baseline and achieve several pre-training models with different settings on the same corpus. More concretely, we implement the following four baselines: (1) **BERT-wwm** (Cui et al., 2021), a BERT-base model trained with the additional fine-grained WWM task, (2) **BERT-wwm-sop**, a BERT-base model trained with the addtional WWM and SOP tasks, (3) **MM-BERT**, a BERT-base model trained with the multi-grained MLM tasks, including a fine-grained WWM task and a coarse-grained MLM task, (4) **MM-BERT-sop**, Multi-grained MLM on a BERT-base model trained with the multi-grained MLM task and the SOP task. In addition, we also include MacBERT (Cui et al., 2021) as a strong baseline, which is one of the state-of-the-art Chinese PLMs in literature. MacBERT utilize the WWM as well as N-gram masking strategies together during pre-training. In terms of the masking implementation, MacBERT masks the word with a similar word rather than the [Mask] placeholder to improve the performance further. The experimental results of MacBERT are achieved with the released model[4] under the identical settings with the other baselines among all downstream tasks.

For three SSC tasks, the results are shown in Table 2. From the results, we can find that our CLOWER yields consistent improvements over all baselines on all three tasks (both on the development and test sets), which proves the effectiveness and advantages of our model. CLOWER outperforms the 4 baselines pre-trained with the identical data while different settings, which demonstrates the advantages of our multi-level contrastive learning approach. In addition, CLOWER outperforms MacBERT by 0.25 points on average and achieves a new state-of-the art on Chinese SSC tasks.

As for the SPC tasks, fair comparisons are performed and the results are reported in Table 3. From the results, we also observe that CLOWER also achieves consistent improvements over baselines on the four tasks. In comparison to the four baselines pre-trained with the identical data, CLOWER outperforms the best one (i.e., MM-BERT-sop) by 0.33 points on average. In comparison to MacBERT, CLOWER achieves a performance gain of 0.33 points on average. CLOWER performs best on all datasets except Xnli

[4] https://github.com/ymcui/MacBERT

Model	Tnews	THUC	Chn	Ocnli	Lcqmc	Xnli	Bq	Average
CLOWER	**67.15**	**97.74**	**95.84**	**76.25**	**88.10**	**79.19**	**85.26**	**84.22**
w/o tcl	66.51	97.54	95.43	75.22	87.24	78.53	84.64	83.59
w/o scl	66.34	97.50	95.02	75.06	87.76	78.40	84.71	83.54
w/o tcl & scl	66.27	97.45	95.65	75.44	87.18	78.62	84.84	83.64

Table 5: Ablation results on SSC and SPC tasks. For Tnews and Ocnli, the results are on development sets and others are on test sets.

Model	CMRC2018		DRCD			
	Dev		Dev		Test	
	EM	F1	EM	F1	EM	F1
CLOWER	**68.73**	**86.52**	88.27	93.44	**87.68**	**92.94**
w/o tcl	67.93	86.25	88.01	93.34	87.19	92.69
w/o scl	68.35	86.18	88.05	93.38	87.21	92.68
w/o tcl & scl	67.57	86.18	**88.30**	**93.50**	87.18	92.76

Table 6: Ablation results on machine reading comprehension tasks.

Dev set.

The above SSC and SPC tasks are all sequence-level tasks, to further examine the effectiveness of our model, we also perform comparisons on MRC tasks which are document-level span-extraction tasks. The resuls are depicted in Tabel 4. Specifically, for CMRC2018, CLOWER outperforms MacBERT by 0.40 points and 0.14 points on EM and F1 score respectively. As the EM score is a stricter measurement of machine reading comprehension, the improvements over MacBERT are considerable. While for DRCD, we find that the performance of CLOWER is not as competitive as the baselines. We conjecture that the reason may be the original dataset of DRCD is in Traditional Chinese whereas our pre-training corpus is in Simplified Chinese. Although we convert the data to Simplified Chinese literally, there are some differences such as syntax and semantics yet, the performances of the pre-trained models may be affected inevitably.

4.5 Ablation Study

To further investigative the effects of contrastive learning over word and characters in CLOWER, we conduct ablation study on the model variants without token-level or sentence-level contrastive learning tasks. Figure 2 shows the ablation results on sentences classification and machine reading comprehension tasks. The detailed ablation results on 9 downstream NLU tasks are reported in Table 5 and 6 respectively.

When removing the token-level contrastive learning task (w/o TCL) or sentence-level (w/o SCL) from CLOWER, there is a distinct drop in the performance on sentence classification tasks (i.e., SSC and SPC). Furthermore, when removing all the contrastive learning tasks, i.e., actually the MM-BERT-sop model, the performance is almost same as the w/o TCL or w/o SCL models. It indicates that only if the token-level contrastive learning task works jointly with the sentence-level contrastive learning task in pre-training, there will be a positive impact on the sentence-level downstream tasks. We conclude that it is vital for the model to encode the coarse-grained semantic information into the fine-grained sequences at token-level and sentence-level consistently when we apply it on sentence-level downstream tasks.

As for the MRC tasks, the EM score on CMRC2018 drops a lot when removing the token-level contrastive learning task, which demonstrates the effectiveness of token-level task on the extractive MRC task. While removing the sentence-level contrastive learning task, the EM metric of the model drops less than that without the token-level. Also, the performance of model without both contrastive learning tasks perform worst among these models on CMRC2018. The results on DRCD reveal the similar trend.

5 Discussions

5.1 Flexibility

Compared to other Chinese PLMs which utilize fine-grained and coarse-grained information, one notable advantage of CLOWER is the high flexibility of deployment. In real-world scenarios, fine-grained PLMs are more popular due to its flexibility on processing inputs/outputs and low computational costs. Please recap that CLOWER could be deemed as a fine-grained character encoder during inference, which is enhanced with the coarse-

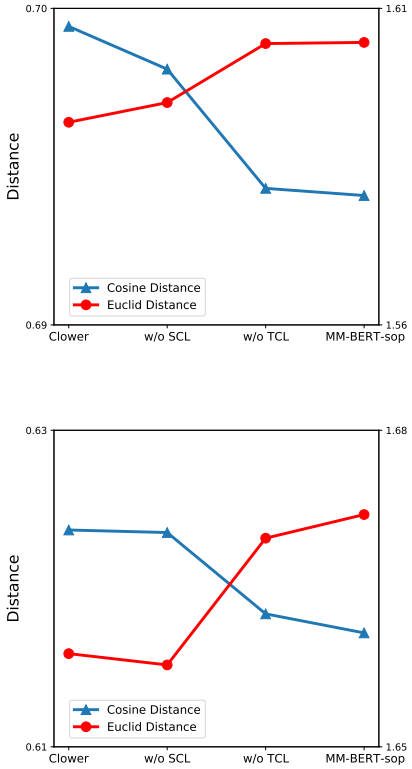

Figure 3: Similarity Analysis of Embeddings. Top: the words with length 2; Bottom: the words with length longer than 2.

grained word encoder during pre-training. In particular, if a production system already deploys a fine-grained Chinese PLM (e.g., the vanilla BERT), the fine-grained encoder of CLOWER can be adopted as a substitute without extra tailor cost seamlessly. CLOWER also provides the coarse-grained encoder (i.e., word encoder) for scenarios where Chinese word sequences are designed as input. The coarse-grained encoder of CLOWER has also been updated and acquired the knowledge from the large corpus during pre-training. We can make flexible choices according to downstream scenarios and conditions when utilizing CLOWER.

5.2 Multi-grained Information Modeling

Through the pre-training, CLOWER implements the multi-grained semantic information modeling by performing the contrastive learning over words to characters and thus implicitly encodes the coarse-grained semantic information into fine-grained tokens and vice versa. To evaluate the character/word representations learned by the interactions, we adopt the measures of cosine similarity and Euclid distance as proxies. We calculate the cosine similarity and Euclid distance between the embeddings of words and the mean embeddings of the characters that compose the words. In our corpus, 72.1% words are composed of two characters. So we conduct the similarity analysis by split the words into two groups, two-character words and the other words composed at least three characters. The similarities produced by four models are shown in Figure 3. We can clearly see that the token-level contrastive learning task play an important role of bringing the word and character embeddings closer, as the similarity of CLOWER and w/o scl are higher than the other two models and so is the Euclid distance. According to the intuitive results, we corroborate that our model indeed achieves our motivation to encode the coarse-grained information into fine-grained tokens.

6 Conclusion

To fully leverage the information of characters and words in Chinese PLMs, we propose a novel PLM CLOWER based on contrastive learning over word and character representations jointly. Through the token-level and sentence-level contrastive learning in the pre-training stage, the model encodes the coarse-grained semantic information into fine-grained tokens. We can not only enhance the model with coarse-grained semantics but also enjoy the flexibility of fine-grained inputs/outputs. The flexibility promises that our model could be deployed conveniently in real scenarios, where certain PLMs like BERT have been established. Comprehensive experiments on a variety of downstream natural language understanding tasks demonstrate the competitive performance of CLOWER. We also conduct a ablation study to evaluate the multi-grained contrastive learning mechanism in CLOWER.

Acknowledgements

This research is supported by National Natural Science Foundation of China (Grant No. 6201101015), Beijing Academy of Artificial Intelligence(BAAI), Natural Science Foundation of Guangdong Province (Grant No. 2021A1515012640), the Major Key Project of PCL (Grant No. PCL2021A06), the Basic Research Fund of Shenzhen City (Grant No. JCYJ20210324120012033 and JSGG20210802154402007), and Overseas Cooperation Research Fund of Tsinghua Shenzhen International Graduate School (Grant No. HW2021008).

References

Jing Chen, Qingcai Chen, Xin Liu, Haijun Yang, Daohe Lu, and Buzhou Tang. 2018. The bq corpus: A large-scale domain-specific chinese corpus for sentence semantic equivalence identification. In *Proceedings of the 2018 conference on empirical methods in natural language processing*, pages 4946–4951.

Ting Chen, Simon Kornblith, Mohammad Norouzi, and Geoffrey Hinton. 2020. A simple framework for contrastive learning of visual representations. In *International conference on machine learning*, pages 1597–1607. PMLR.

Kevin Clark, Minh-Thang Luong, Quoc V Le, and Christopher D Manning. 2019. Electra: Pre-training text encoders as discriminators rather than generators. In *Proceedings of International Conference on Learning Representations*.

Alexis Conneau, Ruty Rinott, Guillaume Lample, Adina Williams, Samuel Bowman, Holger Schwenk, and Veselin Stoyanov. 2018. Xnli: Evaluating cross-lingual sentence representations. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 2475–2485.

Yiming Cui, Wanxiang Che, Ting Liu, Bing Qin, and Ziqing Yang. 2021. Pre-training with whole word masking for chinese bert. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 29:3504–3514.

Yiming Cui, Ting Liu, Wanxiang Che, Li Xiao, Zhipeng Chen, Wentao Ma, Shijin Wang, and Guoping Hu. 2019. A span-extraction dataset for chinese machine reading comprehension. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 5883–5889.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. Bert: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT 2019*, pages 4171–4186.

Hongchao Fang, Sicheng Wang, Meng Zhou, Jiayuan Ding, and Pengtao Xie. 2020. Cert: Contrastive self-supervised learning for language understanding. *arXiv preprint arXiv:2005.12766*.

Tianyu Gao, Xingcheng Yao, and Danqi Chen. 2021. Simcse: Simple contrastive learning of sentence embeddings. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 6894–6910.

John Giorgi, Osvald Nitski, Bo Wang, and Gary Bader. 2021. Declutr: Deep contrastive learning for unsupervised textual representations. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 879–895.

Weidong Guo, Mingjun Zhao, Lusheng Zhang, Di Niu, Jinwen Luo, Zhenhua Liu, Zhenyang Li, and Jianbo Tang. 2021. Lichee: Improving language model pre-training with multi-grained tokenization. In *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*, pages 1383–1392.

Kaiming He, Haoqi Fan, Yuxin Wu, Saining Xie, and Ross Girshick. 2020. Momentum contrast for unsupervised visual representation learning. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 9729–9738.

Hai Hu, Kyle Richardson, Liang Xu, Lu Li, Sandra Kübler, and Lawrence S Moss. 2020. Ocnli: Original chinese natural language inference. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 3512–3526.

Mandar Joshi, Danqi Chen, Yinhan Liu, Daniel S Weld, Luke Zettlemoyer, and Omer Levy. 2020. Spanbert: Improving pre-training by representing and predicting spans. *Transactions of the Association for Computational Linguistics*, 8:64–77.

Prannay Khosla, Piotr Teterwak, Chen Wang, Aaron Sarna, Yonglong Tian, Phillip Isola, Aaron Maschinot, Ce Liu, and Dilip Krishnan. 2020. Supervised contrastive learning. *Advances in Neural Information Processing Systems*, 33:18661–18673.

Diederik P Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*.

Yuxuan Lai, Yijia Liu, Yansong Feng, Songfang Huang, and Dongyan Zhao. 2021. Lattice-bert: Leveraging multi-granularity representations in chinese pre-trained language models. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 1716–1731.

Zhenzhong Lan, Mingda Chen, Sebastian Goodman, Kevin Gimpel, Piyush Sharma, and Radu Soricut. 2019. Albert: A lite bert for self-supervised learning of language representations. *arXiv preprint arXiv:1909.11942*.

Jingyang Li and Maosong Sun. 2007. Scalable term selection for text categorization. In *Proceedings of the 2007 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning (EMNLP-CoNLL)*, pages 774–782.

Xiaoya Li, Yuxian Meng, Xiaofei Sun, Qinghong Han, Arianna Yuan, and Jiwei Li. 2019. Is word segmentation necessary for deep learning of chinese representations? In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 3242–3252.

Xin Liu, Qingcai Chen, Chong Deng, Huajun Zeng, Jing Chen, Dongfang Li, and Buzhou Tang. 2018. Lcqmc: A large-scale chinese question matching corpus. In *Proceedings of the 27th International Conference on Computational Linguistics*, pages 1952–1962.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Roberta: A robustly optimized bert pretraining approach. In *arXiv preprint arXiv:1907.11692*.

Yu Meng, Chenyan Xiong, Payal Bajaj, Paul Bennett, Jiawei Han, Xia Song, et al. 2021. Coco-lm: Correcting and contrasting text sequences for language model pretraining. *Advances in Neural Information Processing Systems*, 34.

Paulius Micikevicius, Sharan Narang, Jonah Alben, Gregory Diamos, Erich Elsen, David Garcia, Boris Ginsburg, Michael Houston, Oleksii Kuchaiev, Ganesh Venkatesh, et al. 2017. Mixed precision training. *arXiv preprint arXiv:1710.03740*.

Chih Chieh Shao, Trois Liu, Yuting Lai, Yiying Tseng, and Sam Tsai. 2018. Drcd: a chinese machine reading comprehension dataset. *arXiv preprint arXiv:1806.00920*.

Yu Sun, Shuohuan Wang, Yukun Li, Shikun Feng, Xuyi Chen, Han Zhang, Xin Tian, Danxiang Zhu, Hao Tian, and Hua Wu. 2019. Ernie: Enhanced representation through knowledge integration. *arXiv preprint arXiv:1904.09223*.

Yu Sun, Shuohuan Wang, Yukun Li, Shikun Feng, Hao Tian, Hua Wu, and Haifeng Wang. 2020. Ernie 2.0: A continual pre-training framework for language understanding. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pages 8968–8975.

Songbo Tan and Jin Zhang. 2008. An empirical study of sentiment analysis for chinese documents. *Expert Systems with applications*, 34(4):2622–2629.

Yi Tay, Vinh Q Tran, Sebastian Ruder, Jai Gupta, Hyung Won Chung, Dara Bahri, Zhen Qin, Simon Baumgartner, Cong Yu, and Donald Metzler. 2021. Charformer: Fast character transformers via gradient-based subword tokenization. *arXiv preprint arXiv:2106.12672*.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. *Advances in neural information processing systems*, 30.

Dong Wang, Ning Ding, Piji Li, and Haitao Zheng. 2021. Cline: Contrastive learning with semantic negative examples for natural language understanding. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 2332–2342.

Zhuofeng Wu, Sinong Wang, Jiatao Gu, Madian Khabsa, Fei Sun, and Hao Ma. 2020. Clear: Contrastive learning for sentence representation. *arXiv preprint arXiv:2012.15466*.

Dongling Xiao, Yu-Kun Li, Han Zhang, Yu Sun, Hao Tian, Hua Wu, and Haifeng Wang. 2021. Ernie-gram: Pre-training with explicitly n-gram masked language modeling for natural language understanding. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 1702–1715.

Liang Xu, Hai Hu, Xuanwei Zhang, Lu Li, Chenjie Cao, Yudong Li, Yechen Xu, Kai Sun, Dian Yu, Cong Yu, et al. 2020. Clue: A chinese language understanding evaluation benchmark. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 4762–4772.

Linting Xue, Aditya Barua, Noah Constant, Rami Al-Rfou, Sharan Narang, Mihir Kale, Adam Roberts, and Colin Raffel. 2022. Byt5: Towards a token-free future with pre-trained byte-to-byte models. *Transactions of the Association for Computational Linguistics*, 10:291–306.

Yuanmeng Yan, Rumei Li, Sirui Wang, Fuzheng Zhang, Wei Wu, and Weiran Xu. 2021. Consert: A contrastive framework for self-supervised sentence representation transfer. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 5065–5075.

Zhilin Yang, Zihang Dai, Yiming Yang, Jaime Carbonell, Russ R Salakhutdinov, and Quoc V Le. 2019. Xlnet: Generalized autoregressive pretraining for language understanding. In *Proceedings of Advances in Neural Information Processing Systems*, volume 32.

Sha Yuan, Hanyu Zhao, Zhengxiao Du, Ming Ding, Xiao Liu, Yukuo Cen, Xu Zou, Zhilin Yang, and Jie Tang. 2021. Wudaocorpora: A super large-scale chinese corpora for pre-training language models. *AI Open*, 2:65–68.

Xinsong Zhang, Pengshuai Li, and Hang Li. 2021. Ambert: A pre-trained language model with multi-grained tokenization. In *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*, pages 421–435.

Yan Zhang, Ruidan He, Zuozhu Liu, Kwan Hui Lim, and Lidong Bing. 2020. An unsupervised sentence embedding method by mutual information maximization. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1601–1610.

On the Nature of BERT: Correlating Fine-Tuning and Linguistic Competence

Federica Merendi[⋆], **Felice Dell'Orletta**[◇], **Giulia Venturi**[◇]
[⋆]Department of Computer Science, University of Pisa
[◇]Istituto di Linguistica Computazionale "Antonio Zampolli", Pisa
ItaliaNLP Lab – *www.italianlp.it*
f.merendi@studenti.unipi.it, {name.surname}@ilc.cnr.it

Abstract

Several studies in the literature on the interpretation of Neural Language Models (NLM) focus on the linguistic generalization abilities of pre-trained models. However, little attention is paid to how the linguistic knowledge of the models changes during the fine-tuning steps. In this paper, we contribute to this line of research by showing to what extent a wide range of linguistic phenomena are forgotten across 50 epochs of fine-tuning, and how the preserved linguistic knowledge is correlated with the resolution of the fine-tuning task. To this end, we considered a quite understudied task where linguistic information plays the main role, i.e. the prediction of the evolution of written language competence of native language learners. In addition, we investigate whether it is possible to predict the fine-tuned NLM accuracy across the 50 epochs solely relying on the assessed linguistic competence. Our results are encouraging and show a high relationship between the model's linguistic competence and its ability to solve a linguistically-based downstream task.

1 Introduction

In the last few years, interest in assessing the linguistic generalization abilities of Neural Language Models (NLMs) has given rise to numerous studies aimed at investigating how the models are able to encode different types of linguistic phenomena. To this end, the most widespread methodology is the *probing classification approach* (Conneau et al., 2018; Warstadt et al., 2019; Hewitt and Liang, 2019), which showed that NLMs are able to encode a variety of language properties (Rogers et al., 2020). Even if most of the work is focused on the linguistic competence of pre-trained models, a complementary line of research is devoted to understanding whether and to what extent the existing competence is modified by fine-tuning the pre-trained model on a supervised downstream dataset (Merchant et al., 2020).

In this paper, we continue this line of research and carry out a study aimed at adopting a probing task approach to investigate how the linguistic competence of one of the most prominent NLM model, BERT (Devlin et al., 2019), is altered after a fine-tuning stage with the main focus on which specific phenomena are preserved or forgotten and which fine-tuning epochs are mainly involved. As a testbed, we chose a downstream task where linguistic information plays the main role. It consists in predicting the evolution of written language competence relying on the linguistic style of chronologically ordered productions written by language learners (Richter et al., 2015; Miaschi et al., 2021a). To the best of our knowledge, it has been never considered in the NLM interpretability literature, and differently from previous studies, the task was carried out at sentence rather than at document level. The present study addresses two further related issues: the first one concerns the debated relationship between the linguistic competence of an NLM and its ability to resolve a task (Ravichander et al., 2021). Secondly, we aim to investigate whether it is possible to predict the accuracy of a fine-tuned model across multiple training epochs solely relying on the linguistic probing task results reflecting BERT's linguistic competence.

Contributions. In this paper, we *i)* compared the ability of BERT's and a Support Vector Machine classifier, which uses a set of explicit linguistic features, in the prediction of the chronological order of two sentences written by the same student across two school years, *ii)* showed the impact of the fine-tuning stage on BERT's linguistic competences, focusing in particular on how they change across 50 epochs of fine-tuning, *iii)* assessed which types of linguistic competence are mainly related to BERT's ability to solve the downstream task, *iv)* predicted the fine-tuned BERT's performance across 50 epochs relying on the linguistic competence of the model.

Proceedings of the 29th International Conference on Computational Linguistics, pages 3109–3119
October 12–17, 2022.

2 Related work

Several methods have been proposed in the literature to obtain meaningful explanations of how NLMs are able to capture syntax- and semantic-sensitive phenomena (Belinkov et al., 2017), also taking inspiration from human language experiments (Ettinger, 2020). Despite highly debated (Belinkov, 2021), one of the mostly explored methods is the definition of *probing tasks* which a model can solve only if it has encoded a precise linguistic phenomenon within its representations.

Particularly relevant to our study are the multiple papers focused on the impact of the fine-tuning stage on how and the extent to which the linguistic knowledge encoded in the pre-trained model is modified, such as Mosbach et al. (2020); Merchant et al. (2020); Durrani et al. (2021); Sarti et al. (2021); Miaschi et al. (2020); Yu and Ettinger (2021) to mention the most recent ones. Even with some main differences concerning the NLMs, and the fine-tuning and probing tasks considered, they agree that the main changes are typically larger for the output layers of a fine-tuned model than for the first ones. However, a shared consensus about whether the linguistic information are preserved, reinforced or forgotten is still missing. For example, Mosbach et al. (2020); Merchant et al. (2020); Sarti et al. (2021) report both a general not catastrophic forgetting and in some cases a substantial improvement of the linguistic competence directly involved in the resolution of the downstream task. On the contrary, Yu and Ettinger (2021) found that fine-tuning on phrase meaning composition sets does not exhibit noteworthy benefits, and Miaschi et al. (2020) show that BERT tends to lose its precision in encoding a wide set of linguistic features after the fine-tuning process. However, to the best of our knowledge, none of the prior works provide a comprehensive analysis of the impact of the fine-tuning stage across the training epochs.

An orthogonal debated issue we are going to address in this study regards the question raised by Ravichander et al. (2021) and concerning the extent information encoded in NLMs is indicative of information needed to perform downstream tasks. Differently from the authors who demonstrated that NLMs are able to encode linguistic properties even if they are not needed for a given downstream task, our results show that there is a strong correlation between the ability to encode a linguistic property and the accuracy to solve the task. An additional novelty of this study is the language we focus on, i.e. Italian. While the vast majority of these studies are focused on English, relatively little work has been done to interpret non-English models. Exceptions are represented by de Vries et al. (2020) focused on Dutch and by Miaschi et al. (2021b); Guarasci et al. (2022) dealing with Italian.

3 Our Approach

Our study includes multiple experiments. Firstly, we test BERT's ability to solve the downstream task, by comparing the performance of the fine-tuned model against a Support Vector Machine classifier which uses a set of explicit features resulted particularly relevant in the resolution of the task, as shown by Richter et al. (2015); Miaschi et al. (2021a). Then, we use a suite of probing tasks to test the linguistic abilities of pre-trained and fine-tuned BERT, with a specific focus on how they change across multiple training epochs of fine-tuning and on which linguistic properties are mostly correlated with the resolution of the downstream task. As linguistic probing tasks, we use the same set of linguistic features used by the SVM. The last experiment is devoted to assess whether it is possible to predict the fine-tuning performance using the accuracy of the linguistic probing tasks.

3.1 Data

We used two datasets: (i) the Universal Dependency Italian Treebank (IUDT) for probing the linguistic knowledge learned before and after a fine-tuning process; (ii) a collection of chronologically ordered essays contained in the *CItA* corpus, which we used to solve the predicting evolution task and for fine-tuning.

IUDT dataset. It includes all 5 sections of the Italian treebank built in the framework of the UD project (de Marneffe et al., 2021), version 2.5, for a total of 33,017 sentences.

CItA (Corpus Italiano di Apprendenti L1). The corpus is the first longitudinal collection of productions written by first language (L1) learners existing for the Italian language (Barbagli et al., 2016). It includes 1,352 essays written by a total of 156 students, aged 11-12, followed from the first to the second year of four different Italian lower secondary schools. Note that this temporal span is considered quite crucial for Italian L1 learners, since a number of relevant transforma-

Linguistic Feature	Label
Raw Text Properties (*RawText*)	
Sentence Length	sent_length
Word Length	char_per_tok
Vocabulary Richness (*Vocabulary*)	
Type/Token Ratio for words and lemmas	ttr_form, ttr_lemma
Morphosyntactic information (*POS*)	
Distribution of language–specific POS	xpos_dist_*
Lexical density	lexical_density
Inflectional morphology (*VerbInflection*)	
Inflectional morphology of lexical verbs and auxiliaries	verbs_*, aux_*
Verbal Predicate Structure (*VerbPredicate*)	
Distribution of verbal heads and verbal roots	verbal_head_dist, verbal_root_perc
Verb arity and distribution of verbs by arity	avg_verb_edges, verbal_arity_*
Global and Local Parsed Tree Structures (*TreeStructure*)	
Depth of the whole syntactic tree	avg_max_depth
Average length of dependency links and of the longest link	avg_links_len, max_links_len
Average length of prepositional chains and distribution by depth	avg_prep_chain_len, prep_dist_*
Clause length	avg_token_per_clause
Order of elements (*Order*)	
Relative order of subject and object	subj_pre, subj_post, obj_post, obj_pre
Syntactic Relations (*SyntacticDep*)	
Distribution of dependency relations	dep_dist_*
Use of Subordination (*Subord*)	
Distribution of subordinate clauses	subordinate_prop_dist
Average length of subordination chains and distribution by depth	avg_subord_chain_len, subordinate_dist_*
Relative order of subordinate clauses	subordinate_post

Table 1: Linguistic features used in the experiments.

tions in writing competence occurs during these two years, as shown by studies in experimental pedagogy (Barbagli et al., 2015). Accordingly, the corpus has been also successfully exploited to develop NLP-based approaches for tracking the evolution of written language competence (Miaschi et al., 2021a).

3.2 Linguistic Probing Tasks

To evaluate the linguistic competence encoded by BERT, we adopted the linguistic probing paradigm defined by Conneau et al. (2018) and followed the approach devised by Miaschi et al. (2020) who defined a suite of 68 probing tasks each corresponding to a distinct linguistic property of a sentence extracted from the IUDT sentences using Profiling-UD (Brunato et al., 2020), a tool able to acquire a wide set of linguistic features from linguistically annotated corpora according to the UD formalism. Each task consists in predicting the value that specific property has in IUDT using the representations learned by the NLM during the pre-training and fine-tuning processes. The set is reported in Table 1 and covers 9 main aspects of the structure of a sentence. They range from quite simple aspects concerning raw text properties (sentence and word length), vocabulary richness, the distribution of language-specific Parts-Of-Speech and of verbal inflectional properties (i.e. mood, tense, person).[1] More challenging probing tasks concern the ability to encode global structures of the sentence, such as the depth of the whole syntactic tree, and lexical features. We also paid a specific attention to testing the models knowledge of specific sub-trees, including a group of features modelling the verbal predicate structure, the order of subjects and objects with respect to their verbal head, and the use of subordination.

3.3 Models

Neural Language Model. We considered a BERT-base cased model trained on the Italian Wikipedia and texts from the OPUS corpus (Tiedemann and Nygaard, 2004).[2] Sentence-level representations are obtained using for each of the 12 layers the activation of the first input token (*[CLS]*).

Support Vector Machines Model. The SVM classifier is based on a linear kernel using the linguistic features described in Table 1. For each pair of sentences (s_i, s_j), the feature vectors V_{s_i}, V_{s_j} were combined in the final feature vector $V_{s_i,s_j} = V_{s_i} - V_{s_j}$, and normalized in the range [0,1].

Probing Model. We used a linear Support Vector Regression tested on 10,000 randomly selected IUDT sentences and trained on remaining ones. It takes as input layer-wise sentence-level representations extracted from the pre-trained and fine-tuned BERT model. As evaluation metric, we used the Spearman correlation coefficient (ρ score) between the values of the linguistic properties predicted by BERT and their gold values in IUDT sentences.

[1] For the list of Parts-Of-Speech refer to http://www.italianlp.it/docs/ISST-TANL-POStagset.pdf

[2] https://huggingface.co/dbmdz/bert-base-italian-cased

4 Predicting the Evolution of Written Language Competence

Our first experiment is aimed at assessing BERT's ability to predict the evolution of written language competence of L1 learners. Following Richter et al. (2015) and Miaschi et al. (2021a), we modelled the task as a binary classification one. We started from the assumption that, given a set of chronologically ordered essays written by the same student, a document d_j should have a higher quality level with respect to the ones written previously (d_i). However, differently from previous studies, our analysis was carried out at the sentence level. Thus, given two sentences s_i and s_j, belonging to document d_i and d_j respectively, the task consists in predicting whether $t(s_j) > t(s_i)$, where $t(s_i)$ identifies the time in which the sentence s_i was written.

To this end, we built a dataset composed of all possible pairs (s_i, s_j) of sentences contained in the *CItA* corpus and written by the same student in the two school years. We considered only the sentences contained in the first essay written during the first year and the penultimate of the second year of four different schools, for a total of 3,562 sentences and 33,566 pairs. This is motivated by the fact that this represents the widest temporal span. For each pair, we assigned two labels: 1 if $t(s_j) > t(s_i)$ and 0 otherwise. To balance the dataset, we included only one of the two labels, i.e. 50% of the pairs with label 0 (randomly selected) and the remaining of the pairs with label 1.

***CItA* statistics.** As a preliminary analysis, we investigated which linguistic phenomena are mostly involved in the writing transformations across the two years. Thus, we compared the distribution of the linguistic features automatically extracted from the set of paired sentences and described in Table 1. Table 2 shows the top 15 features ordered by decreasing number of times their value is higher in the first-year sentences.[3] The main differences concern the behaviour of verbs, both in terms of distribution of auxiliaries (*dep_dist_aux*) possibly used in compound tenses, and of characteristics specific to the verbal inflectional morphology, e.g. indicative mood (*aux_mood_dist_Ind*), and participial forms (*verbs_form_dist_Part*). The different use of connecting elements characterising the internal structure of a sentence, such as coordinating conjunctions (*xpos_dist_CC, dep_dist_cc*) and punctuation (*dep_dist_punct*), represents an additional aspect of variation.

[3] All variations across the two years are statistically significant according to the Wilcoxon Rank Sum Test.

Ranking	Feature	Difference
1	dep_dist_aux	10081
2	xpos_dist_VA	9365
3	aux_mood_dist_Ind	9046
4	verbs_form_dist_Part	8452
5	aux_form_dist_Fin	8206
6	dep_dist_conj	7642
7	aux_tense_dist_Pres	7619
8	verbs_num_pers_dist_Sing+3	-7307
9	dep_dist_cc	6493
10	xpos_dist_CC	6440
11	dep_dist_punct	-6411
12	verbs_form_dist_Fin	-5330
13	xpos_dist_E	-4707
14	verbs_tense_dist_Pres	-4688
15	xpos_dist_SP	-4525

Table 2: Ranking of first 15 features by decreasing number of times the feature value is higher in the first-year than in the second-year sentence.

4.1 Results

The SVM and BERT classification systems were evaluated with a 4-fold cross-validation in which every fold uses as a test set the sentences of the school not included in the corresponding training set. The accuracy for each fold refers to the percentage of sentences correctly classified and the final accuracy is the average score.

Figure 1 reports the task performances achieved by BERT for each of the 50 fine-tuning epochs. As we can see, the highest accuracy is 0.78 and it is achieved at the 11th epoch. It is worth noting that lower accuracy, ranging from 0.73 to 0.76, was obtained at epochs 2, 3 and 4 which were suggested as optimal hyperparameters in many tasks by Devlin et al. (2019).

Note that BERT outperforms the results achieved by the SVM classifier, which is able to solve the task with an accuracy of 0.70. The relatively small difference in terms of accuracy between a deep NLM and a simple linear classifier, which does not exploit words of the sentence as features but only linguistic properties, seems to confirm our intuition regarding the linguistic nature of the chosen task. A more in-depth analysis focused on the contribution of specific linguistic features used by the SVM classifier highlights that the most predicting ones are represented by the use of verbs, in particular of the auxiliary verbs, and of punctuation and conjunctions. This is in line with the statistics reported

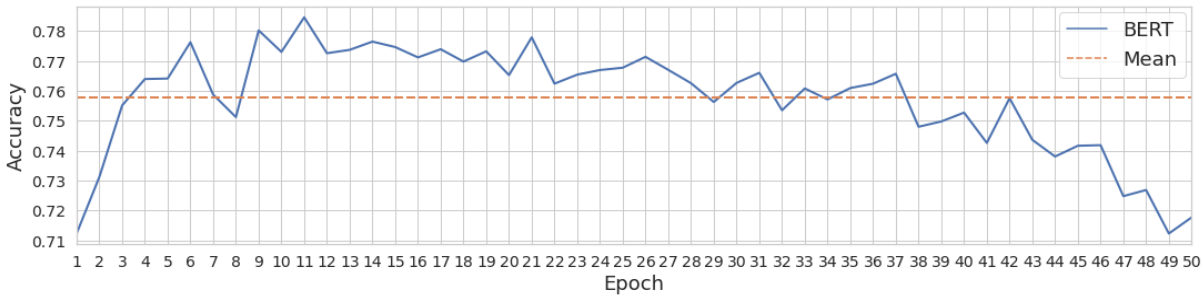

Figure 1: BERT accuracy on the downstream task across 50 fine-tuning epochs (*BERT*) and on average (*Mean*).

in Table 2, where we showed that a change in the distribution of these linguistic features is among the most relevant writing transformations occurring across the two school years.

5 How Does Linguistic Competence Change Across the Fine-tuning Epochs?

BERT linguistic competence was tested by adopting a supervised probing classifier approach, which accounts for 68 probing tasks. As detailed in Section 3.3, we trained a LinearSVR probing model which uses the sentence-level representations extracted from the pre-trained and fine-tuned model. Since many of our probing features are strongly related to sentence length, we compared the results with the ones obtained using a baseline computed by using a probing model trained using only sentence length as an input feature.

Pre-training linguistic competence. Figure 2 reports the probing task results obtained using the pre-trained BERT's representations. As a general remark, we can notice that BERT's scores always outperform the baseline ones, with the obvious exception of the sentence length (*n_tokens*). However, as expected, the results are more similar for the probing tasks whose resolution is highly influenced by the length of the sentence. This is the case of global and local properties of the sentence such as the depth of the syntactic tree (*avg_max_depth*) and the length of the longest dependency links (*max_links_len*). They correspond to linguistic phenomena that the NLM masters very well. Nevertheless, differently from the baseline, BERT shows to encode quite accurately also the only lexical property we considered, i.e. the Type/Token ratio, and a raw text feature highly related to the use of lexicon, i.e. the length of tokens (*char_per_tok*). Pre-trained representations are also very good at predicting the distribution of functional information both in terms of Parts-Of-Speech such as prepositions (*xpos_dist_E*), coordinating conjunctions (**_CC*), determinative articles (**_RD*), and of dependency relations linking a functional POS to its head (i.e. *dep_dist_case*, **_cc*, **_det* respectively), and of punctuation, in particular of commas (**_FF*) and full stops (**_FS*). Lastly, concerning BERT's knowledge about verbs, we can note that the model encodes quite accurately the tense, person and mood of auxiliary verbs (*aux_**) and the distribution of verbal heads (*verbal_head_per_sent*).

When we move to the analysis of how the competence changes across layers, we can observe that it generally tends to decrease when the output layer is approaching. This is in line with previous findings (Liu et al., 2019; Miaschi et al., 2020) and it could be due to the fact that Transformer layers trade off between task-oriented (e.g. Masked Language Modeling) information and general linguistic competence. A such decreasing trend can be specifically appreciated by observing the evolution of the black dotted line reported in Figure 3. It shows how BERT's competence in the 9 groups of linguistic phenomena introduced in Table 1 evolves across the 12 layers. As it can be seen, however not all features have the same decreasing trend. The main exceptions are represented by quite complex linguistic features, such as the order of subject/object with respect to the verbal head (*Order of elements*), the use of subordination and the verbal predicate structure, which increase consistently across layers, even if they decrease in the output layer. These linguistic features model syntactic sub-trees of the sentence, that require an in-depth knowledge of the syntactic structure of the sentence, which is encoded starting from the middle layers, as shown for English (Tenney et al., 2019; Miaschi et al., 2020).

Fine-tuning impact. Coloured lines in Figure 3 report the layer-wise ρ scores we obtained us-

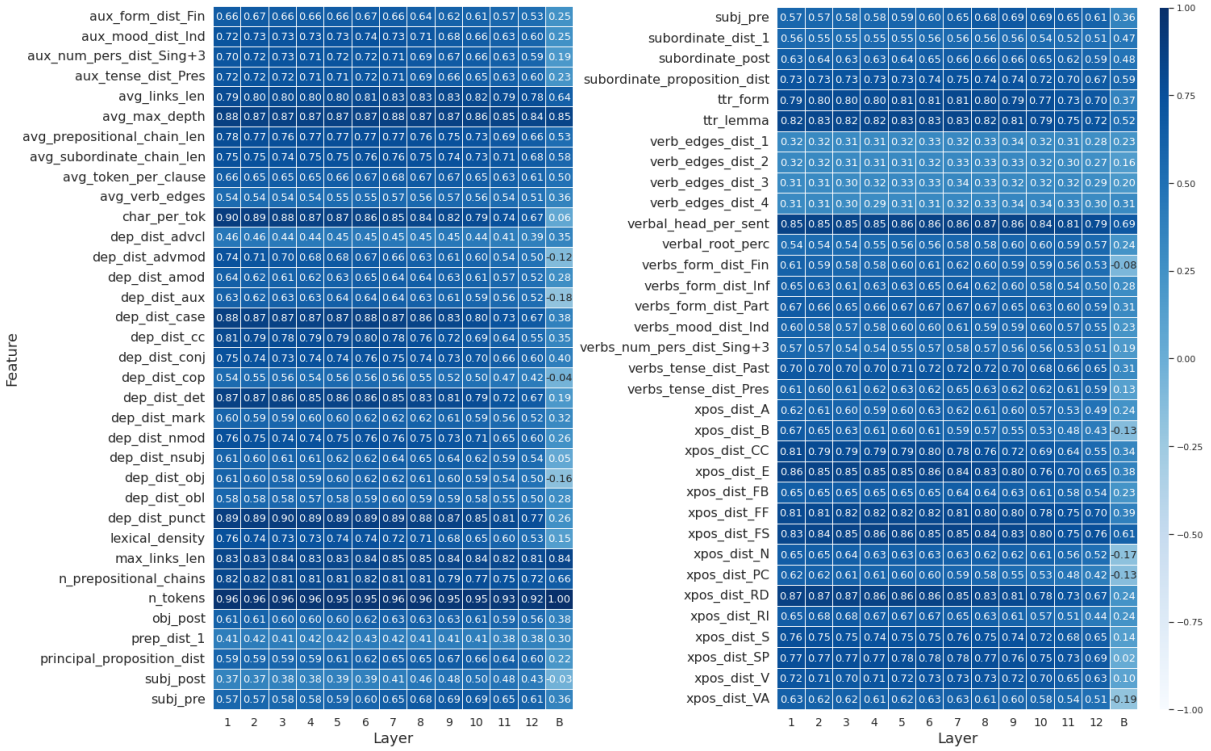

Figure 2: Layer-wise ρ scores for the 68 probing tasks obtained with the pre-trained model. Baseline score are reported in column B.

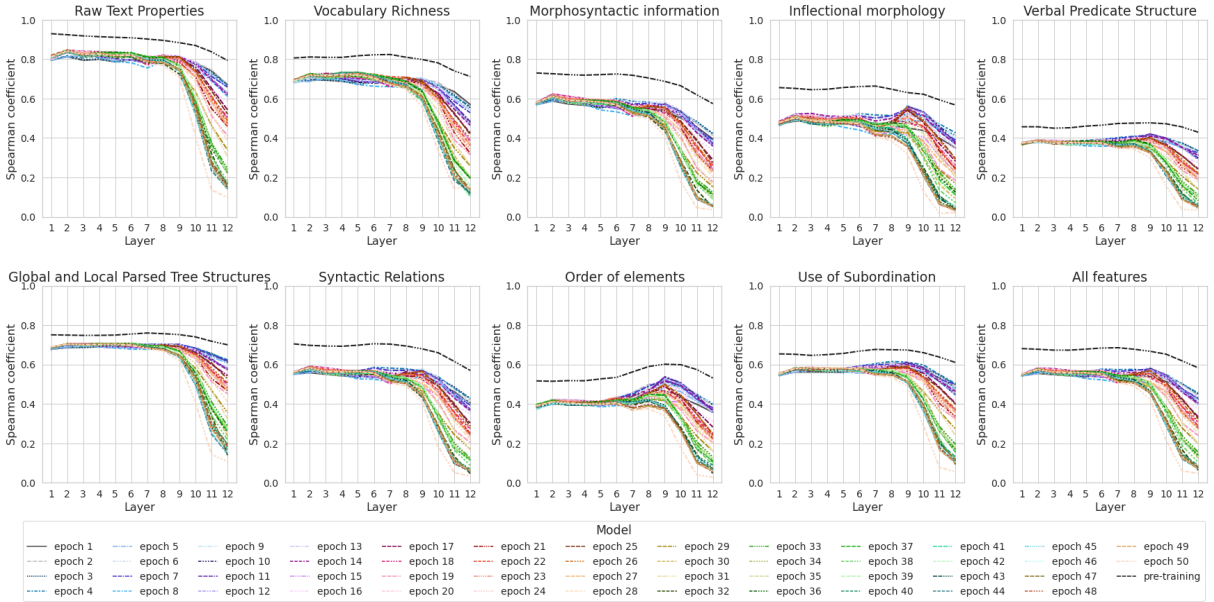

Figure 3: Pre-trained and fine-tuned BERT layerwise ρ scores for groups of probing features.

ing the representations extracted by each of the 50 epochs of fine-tuning on the *CItA* corpus. Differently from scholars who found that some linguistic abilities are reinforced after the fine-tuning process, such as Merchant et al. (2020) and Mosbach et al. (2020), but more similarly to Miaschi et al. (2020), our analysis shows that BERT tends to lose the linguistic competence acquired during the pre-training phase. However, such loss has a different impact across the 50 epochs and the 12 layers. For all epochs, all linguistic competencies are mostly altered starting from the 8th layer, thus resulting mostly modified in the 12th one. This might be possibly due to the fact that during these layers the

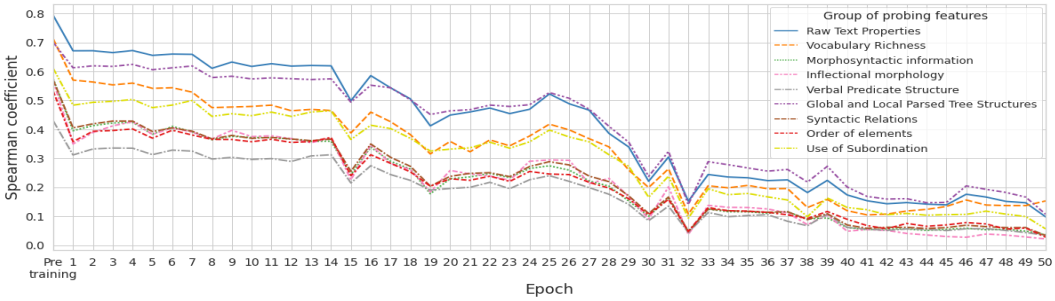

Figure 4: ρ scores for groups of probing features at the 12th layer of pre-training and across the 50 epochs of fine-tuning.

model is storing task–specific information at the expense of its ability to encode general knowledge about the language (Kovaleva et al., 2019).

In addition, note that the impact of the fine-tuning stage is less evident in the first epochs (blue lines in the figure) than in the last ones (green lines). Since the main changes are observed in the 12th layer, we deepen the analysis and report in Figure 4 the probing scores for each group of features at each epoch. The trend shows that for all groups of features a first main drop (with respect to the pre-training scores) occurs after the first epoch, then the sentence properties are constantly forgotten across the 50 epochs. However, such a negative impact is different for each group. For example, the knowledge of simple raw text features, which BERT knows very well, drops from 0.8 to 0.1. Similarly, the generalization abilities of a highly related group of features, i.e. those referring to global and local syntactic characteristics of a sentence, decrease quite considerably.

To further investigate which group of features is forgotten most rapidly across the epochs, we calculated a slope of a linear regression line between the 50 epochs and the decay of competencies for each linguistic group.[4] Results are reported in Table 3 where we show the 9 groups of features ordered according to decreasing slope values. As it can be noted, the ranking does not reflect BERT's linguistic competence. Namely, in the top and last position we find groups of features that pre-trained BERT knows very well, i.e. Raw text properties and Vocabulary Richness. Similarly, the groups that are scarcely mastered are scattered along the ranking. This might suggest that the change of competencies across the epochs is not related to the pre-trained knowledge but possibly to some characteristics useful in the resolution of the downstream task.

Groups of probing features	**slope**	**r-value**
Raw Text Properties	1.679	0.966
Global and Local Parsed Tree Structures	1.644	0.955
Use of Subordination	1.622	0.961
Verbal Predicate Structure	1.618	0.964
Inflectional morphology	1.581	0.955
All features	**1.574**	**0.965**
Syntactic Relations	1.542	0.965
Order of elements	1.534	0.964
Morphosyntactic information	1.534	0.965
Vocabulary Richness	1.449	0.958

Table 3: Ranking of groups of probing features according to decreasing slope of the regression lines. Correlation coefficients are also reported (*r-value*).

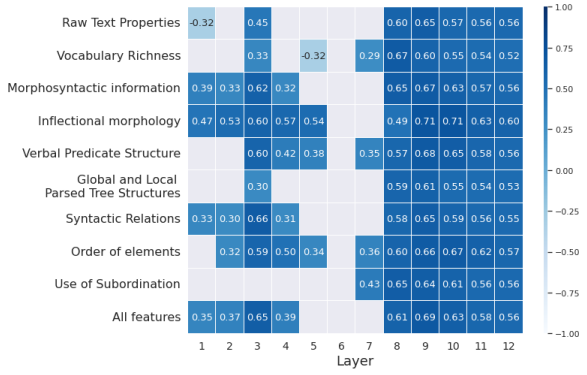

Figure 5: Layer-wise correlation between the average ρ score for groups of features and BERT's accuracy on the downstream task. Blank cells are non statistically significant correlations.

6 Task Resolution and Linguistic Competence

Correlating task accuracy with linguistic competence. For each single feature and each group of linguistic features, Figures 5 and 6 report the results of

[4] decay = (probing score at layer 12 of pre-training - probing score at layer 12 epoch *i*) / probing score at 12 layer of pre-training

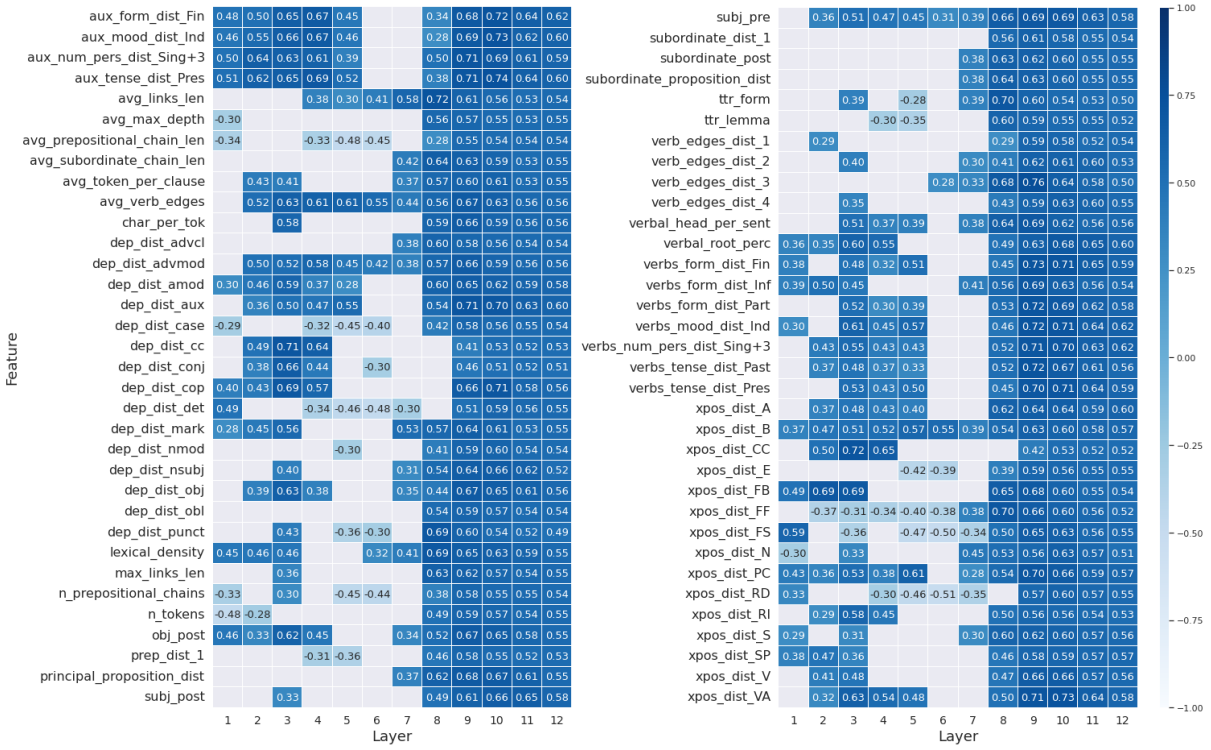

Figure 6: Layer-wise correlation between the Spearman coefficient value of the 68 linguistic features and the accuracy over the 50 epochs considered for the fine-tuning.

the Spearman correlations between the layer-wise probing scores and the accuracy of BERT in the resolution of the downstream task for all epochs. The results were compared with a baseline computed as the Spearman correlation between BERT's accuracy on the downstream task and a list of numbers from 50 to 1 (ρ=0.55)[5]. As a general remark, in the first layers (from 1st to 7th), we noticed a non-significant or low correlation, most of the time lower than the baseline scores. On the contrary, the scores are higher, always above the baseline, in the central layers (8-10) and they tend to decrease in the 11th and 12th layers, where task-specific information is stored. The trend suggests that the higher BERT's linguistic competence, the more accurate the resolution of the downstream task is. This is a global trend as showed by *All features*, even if there are some differences among the studied linguistic phenomena. Namely, information about verbs (both in terms of inflectional morphology and predicate structure) and the order of nuclear elements of sentence results to be the linguistic competence mostly correlated with the resolution of the task. In particular, the distribution of auxiliary verbs (both in terms of POS and dependency relations), clitic pronouns, tense, person and mood of both auxiliary and lexical verbs, of subjects in their canonical order (i.e. in the pre-verbal position), show a very high correlation ($\rho \geq 0.70$) in either the 9th or 10th layer. Interestingly, these linguistic phenomena are the mostly involved in the writing transformations across the two school years (see Table 2).

Predicting task accuracy with linguistic competence. The last experiment is aimed at testing whether it is possible to predict BERT's accuracy in the resolution of the downstream task on the basis of its linguistic competence assessed by the linguistic probing tasks. For this purpose, we trained a linear Support Vector Regressor using a Leave-one-out strategy. Namely, each time, we used as training the average accuracy of the probing tasks (in terms of ρ scores) achieved at the 12th layer of 49 epochs, and the excluded epoch as the test. As an evaluation metric, we used the Spearman correlation and the Mean Squared Error (MSE) between the predicted values and the actual ones. We devised two baselines: the first one consists of a list of 50 numbers corresponding to the average BERT's

[5]Such baseline was chosen since it simulates the decreasing trend of probing score accuracy during the 50 epochs of fine-tuning as shown in Figure 4.

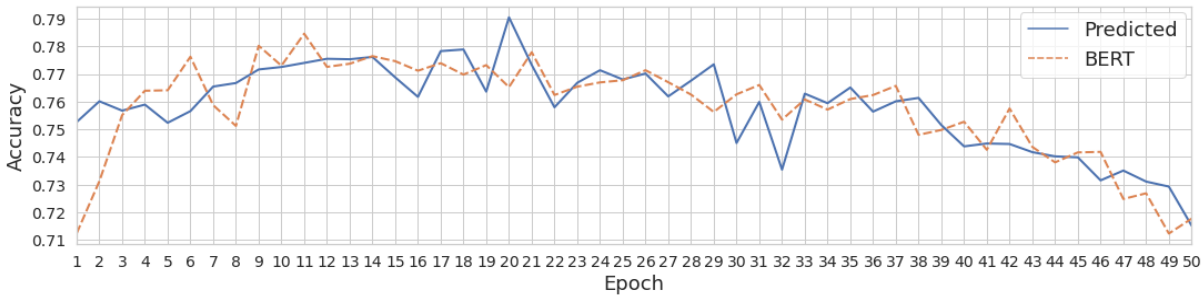

Figure 7: Predicted accuracy at the 12th layer and BERT's actual accuracy for each epoch of fine-tuning.

accuracy in the resolution of the downstream task considering all the epochs. In this case, since the value is always the same for all the epochs, the prediction was evaluated only in terms of MSE. The second baseline is a linear SVR using a decreasing list of numbers from 50 to 1, simulating the decreasing trend of the probing accuracy across the epochs. Here, we were able to compute also the correlation scores. The results show that the SVR achieves very high scores (ρ=0.76, MSE=0.011) which outperform the two baselines (ρ=NA, MSE=0.017; ρ=0.54, MSE=0.017). This demonstrates that the BERT accuracy can be reliably predicted by studying its linguistic abilities. The reliability of this result is corroborated by Figure 7 where we compare the trends of the predicted (*Predicted*) and BERT actual accuracy. It is worth noting that the two lines are almost overlapping, showing a very similar trend.

7 Conclusion

In this paper, we carried out an extensive study aimed at investigating in detail the relation between the fine-tuning stage and the linguistic knowledge encoded by BERT. For our study, we chose the prediction of the evolution of written language competence of native language learners as a downstream task, since the morpho-syntactic and syntactic information plays an important role in the resolution. We showed in particular how the knowledge assessed during the pre-training stage is progressively forgotten across 50 epochs of fine-tuning. We observed that the main changes concern the 12th layer of all epochs and they are more evident in the last than in the first epochs. An in-depth analysis showed that the types of linguistic phenomena forgotten most rapidly across epochs are not related to the linguistic knowledge of the pre-trained NLM but possibly to some characteristics useful to solve the downstream task. We also found a strong correlation between the linguistic knowledge encoded by BERT and its ability to resolve the task. Specifically, we showed that the verbal inflectional morphology, the predicate structure, and the canonical order of subject and object are the most correlated aspects. Interestingly, these linguistic characteristics, in particular those referring to verbs, resulted to be the most predictive types of features exploited by the SVM classifier and the most involved in the writing transformations. A such strong relationship between the ability to the resolution of the task and linguistic competence is particularly evident in our last experiment, where we obtained very strong accuracy in predicting BERT's performance using the encoded linguistic competence. Besides investigating the nature of BERT this outcome could open the way to define a new linguistically motivated stop-function of the fine-tuning process.

Ethical Considerations

The findings described in this work are based on the analysis of essays written by Italian L1 learners and they are mostly intended to evaluate recent efforts in the interpretability of neural language models. As we used authentic texts from students, we took care of anonymization separating their personally identifying information from their essays to prevent any harmful uses of data.

Acknowledgements

We acknowledge the project "Human in Neural Language Models" (*IsC93_HiNLM*), funded by the CINECA[6] under the ISCRA initiative, for the availability of HPC resources and support.

References

Alessia Barbagli, Pietro Lucisano, Felice Dell'Orletta, Simonetta Montemagni, and Giulia Venturi. 2015. Il

[6]https://www.cineca.it/en

ruolo delle tecnologie del linguaggio nel monitoraggio dell'evoluzione delle abilità di scrittura: primi risultati. *Italian Journal of Computational Linguistics (IJCoL)*, vol. 1(1):99–117.

Alessia Barbagli, Pietro Lucisano, Felice Dell'Orletta, Simonetta Montemagni, and Giulia Venturi. 2016. CItA: an L1 Italian learners corpus to study the development of writing competence. In *Proceedings of the Tenth International Conference on Language Resources and Evaluation (LREC'16)*, pages 88–95, Portorož, Slovenia. European Language Resources Association (ELRA).

Yonatan Belinkov. 2021. Probing Classifiers: Promises, Shortcomings, and Advances. *Computational Linguistics*, pages 1–13.

Yonatan Belinkov, Lluís Màrquez, Hassan Sajjad, Nadir Durrani, Fahim Dalvi, and James Glass. 2017. Evaluating layers of representation in neural machine translation on part-of-speech and semantic tagging tasks. In *Proceedings of the Eighth International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 1–10.

Dominique Brunato, Andrea Cimino, Felice Dell'Orletta, Giulia Venturi, and Simonetta Montemagni. 2020. Profiling-ud: a tool for linguistic profiling of texts. In *Proceedings of The 12th Language Resources and Evaluation Conference*, pages 7147–7153, Marseille, France. European Language Resources Association.

Alexis Conneau, German Kruszewski, Guillaume Lample, Loïc Barrault, and Marco Baroni. 2018. What you can cram into a single $&!#* vector: Probing sentence embeddings for linguistic properties. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 2126–2136, Melbourne, Australia. Association for Computational Linguistics.

Marie-Catherine de Marneffe, Christopher D. Manning, Joakim Nivre, and Daniel Zeman. 2021. Universal Dependencies. *Computational Linguistics*, 47(2):255–308.

Wietse de Vries, Andreas van Cranenburgh, and Malvina Nissim. 2020. What's so special about BERT's layers? a closer look at the NLP pipeline in monolingual and multilingual models. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 4339–4350, Online. Association for Computational Linguistics.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, Minneapolis, Minnesota. Association for Computational Linguistics.

Nadir Durrani, Hassan Sajjad, and Fahim Dalvi. 2021. How transfer learning impacts linguistic knowledge in deep NLP models? In *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*, pages 4947–4957, Online. Association for Computational Linguistics.

Allyson Ettinger. 2020. What BERT Is Not: Lessons from a New Suite of Psycholinguistic Diagnostics for Language Models. *Transactions of the Association for Computational Linguistics*, 8:34–48.

Raffaele Guarasci, Stefano Silvestri, Giuseppe De Pietro, Hamido Fujita, and Massimo Esposito. 2022. Bert syntactic transfer: A computational experiment on italian, french and english languages. *Computer Speech & Language*, 71:101261.

John Hewitt and Percy Liang. 2019. Designing and interpreting probes with control tasks. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 2733–2743.

Olga Kovaleva, Alexey Romanov, Anna Rogers, and Anna Rumshisky. 2019. Revealing the dark secrets of BERT. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 4365–4374, Hong Kong, China. Association for Computational Linguistics.

Nelson F. Liu, Matt Gardner, Yonatan Belinkov, Matthew E. Peters, and Noah A. Smith. 2019. Linguistic knowledge and transferability of contextual representations. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 1073–1094, Minneapolis, Minnesota. Association for Computational Linguistics.

Amil Merchant, Elahe Rahimtoroghi, Ellie Pavlick, and Ian Tenney. 2020. What happens to BERT embeddings during fine-tuning? In *Proceedings of the Third BlackboxNLP Workshop on Analyzing and Interpreting Neural Networks for NLP*, pages 33–44, Online. Association for Computational Linguistics.

Alessio Miaschi, Dominique Brunato, and Felice Dell'Orletta. 2021a. A NLP-based stylometric approach for tracking the evolution of L1 written language competence. *Journal of Writing Research*, vol. 13(1):71–105.

Alessio Miaschi, Dominique Brunato, Felice Dell'Orletta, and Giulia Venturi. 2020. Linguistic profiling of a neural language model. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 745–756, Barcelona, Spain (Online). International Committee on Computational Linguistics.

Alessio Miaschi, Gabriele Sarti, Dominique Brunato, Felice Dell'Orletta, and Giulia Venturi. 2021b. Italian transformers under the linguistic lens. In *Proceedings of the Seventh Italian Conference on Computational Linguistics (CLiC-it 2020)*, Online. CEUR Workshop Proceedings (CEUR-WS.org).

Marius Mosbach, Anna Khokhlova, Michael A. Hedderich, and Dietrich Klakow. 2020. On the interplay between fine-tuning and sentence-level probing for linguistic knowledge in pre-trained transformers. In *Proceedings of the Third BlackboxNLP Workshop on Analyzing and Interpreting Neural Networks for NLP*, pages 68–82, Online. Association for Computational Linguistics.

Abhilasha Ravichander, Yonatan Belinkov, and Eduard Hovy. 2021. Probing the probing paradigm: Does probing accuracy entail task relevance? In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*, pages 3363–3377, Online. Association for Computational Linguistics.

Stefan Richter, Andrea Cimino, Felice Dell'Orletta, and Giulia Venturi. 2015. Tracking the evolution of written language competence: an nlp–based approach. *CLiC it*, page 236.

Anna Rogers, Olga Kovaleva, and Anna Rumshisky. 2020. A primer in BERTology: What we know about how BERT works. *Transactions of the Association for Computational Linguistics*, 8:842–866.

Gabriele Sarti, Dominique Brunato, and Felice Dell'Orletta. 2021. That looks hard: Characterizing linguistic complexity in humans and language models. In *Proceedings of the Workshop on Cognitive Modeling and Computational Linguistics*, pages 48–60, Online. Association for Computational Linguistics.

Ian Tenney, Dipanjan Das, and Ellie Pavlick. 2019. BERT rediscovers the classical NLP pipeline. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 4593–4601, Florence, Italy. Association for Computational Linguistics.

Jörg Tiedemann and Lars Nygaard. 2004. The opus corpus-parallel and free: http://logos. uio. no/opus. Citeseer.

Alex Warstadt, Yu Cao, Ioana Grosu, Wei Peng, Hagen Blix, Yining Nie, Anna Alsop, Shikha Bordia, Haokun Liu, Alicia Parrish, Sheng-Fu Wang, Jason Phang, Anhad Mohananey, Phu Mon Htut, Paloma Jeretic, and Samuel R. Bowman. 2019. Investigating BERT's knowledge of language: Five analysis methods with NPIs. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 2877–2887, Hong Kong, China. Association for Computational Linguistics.

Lang Yu and Allyson Ettinger. 2021. On the interplay between fine-tuning and composition in transformers. In *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*, pages 2279–2293, Online. Association for Computational Linguistics.

LayerConnect: Hypernetwork-Assisted Inter-Layer Connector to Enhance Parameter Efficiency

Haoxiang Shi[1]*†, Rongsheng Zhang[2]*, Jiaan Wang[3]†, Cen Wang[4]
Yinhe Zheng[5] and Tetsuya Sakai[1]
[1] Waseda University, Tokyo, Japan [2] Fuxi AI Lab, NetEase Inc., Hangzhou, China
[3] Soochow University, Suzhou, China [4] KDDI Research Inc., Japan [5] Lingxin AI, China
hollis.shi@toki.waseda.jp, tetsuya@waseda.jp, zhangrongsheng@corp.netease.com
jawang1@stu.suda.edu.cn, ce-wang@kddi-research.jp, zhengyinhe1@163.com

Abstract

Pre-trained Language Models (PLMs) are the cornerstone of the modern Natural Language Processing (NLP). However, as PLMs become heavier, fine tuning all their parameters loses their efficiency. In this paper, we propose LayerConnect (hypernetwork-assisted inter-layer connectors) to enhance inference efficiency. Specifically, a light-weight connector with a linear structure is inserted between two Transformer layers, and the parameters inside each connector are tuned by a hypernetwork comprising an interpolator and a down-sampler. We conduct extensive experiments on the widely used GLUE benchmark. The experimental results verify the inference efficiency of our model. Compared to Adapter, with our structure, parameters are reduced to approximately 11.75% for base PLMs and 8.82% for large PLMs, while the performance degradation is kept to less than 5% (2.07 points on average).

1 Introduction

The emergence of pre-trained language models (PLMs) has brought Natural Language Processing (NLP) to a new era (Qiu et al., 2020). Fine tuning all PLMs parameters has become the most common strategy to apply them to downstream tasks (Zheng et al., 2021a; Lai et al., 2021; Wang et al., 2021, 2022; Chen et al., 2022; He et al., 2022; Zheng et al., 2020; Wang et al., 2020; Zheng et al., 2021b,c; Zhang et al., 2020; Zhou et al., 2021). However, in this manner, one should store a full copy of a PLM for each downstream task because all its parameters are updated during the fine-tuning stage. Thus, this strategy challenges the usage of the PLMs on edge devices (e.g., mobile phone and embedded systems) that have limited storage space. Under this background, parameter efficiency has gradually attracted attention from both the research community and industries (Houlsby et al., 2019; Li and Liang, 2021; Zaken et al., 2021; Liu et al., 2021; Ding et al., 2022; Zhang et al., 2021). The core concept of parameter efficiency is to reduce the trainable parameters of PLMs. Consequently, when deploying a PLM, only the different trained parameters from multiple tasks must be stored, together with one copy of the shared frozen parameters.

Currently, there are three main types of parameter efficiency methods: (1) Specification-based methods train only a small part of the parameters in the original PLM, while the others are frozen. For example, Bitfit (Zaken et al., 2021) only updates the parameters of the bias and task-specific layers in PLMs. (2) Prompt-based methods prepend additional context (i.e., prompts) to the original input, and change only the prompt parameters during fine tuning. In this manner, P-tuning (Liu et al., 2021) adopts an extra network to functionalize the optimization of the continuous prompt embedding. Then, Prefix-tuning (Li and Liang, 2021) is proposed to use trainable prefixes to accomplish a parameter-efficiency task. (3) The Adapter-based methods inject small-scale neural modules to PLMs and update only these modules during fine tuning. Adapter (Houlsby et al., 2019) simply injects two linearly trainable projectors (i.e., Adapter layers) into each transformer layer of PLMs. Karimi Mahabadi et al. (2021) further make the weights of Adapter layers conditioned on hypernetworks (Ha et al., 2016) facilitating parameter sharing across multiple tasks. Later, Hu et al. (2021) propose Low-Rank Adaptation (LoRA), a layer-parallel structure which contains less parameters than Adapter.

In view of hypernetwork (Ha et al., 2016) which generates weights for another network, although its superiority has been proved in multi-task parameter efficiency (Karimi Mahabadi et al., 2021), the single linear projector used in their hypernetwork continues to have under-explored abilities (e.g., avoid-

*These two authors contributed equally.
† This work is mainly done when SHI and WANG were interning at NetEase Inc..

Proceedings of the 29th International Conference on Computational Linguistics, pages 3120–3126
October 12–17, 2022.

ing local optima and performance compensation when linear structure size is reduced). Furthermore, the PLM used in Karimi Mahabadi et al. (2021) is T5 (Raffel et al., 2020), a pre-trained encoder-decoder model. Therefore, the effectiveness of the hypernetwork remains unknown in encoder-only PLMs (e.g., BERT and RoBERTa). To this end, we propose LayerConnect, a hypernetwork-assisted inter-layer connector. Specifically, in the proposed architecture, we first insert a connector between every adjacent transformer layer in a PLM. Only the parameters of the inserted connector are updated during the fine-tuning stage. Then, to share parameters across multiple tasks and avoid local convergence, the weights of the connector are conditioned on a well-designed hypernetwork that is equipped with an interpolator and a down-sampler. The proposed hypernetwork reduces the required parameters in the connectors and ensures sufficient optimization space. We conduct extensive experiments on the GLUE benchmark (Wang et al., 2018). The experimental results show that compared with Adapter, our LayerConnect requires only 11.75% of its parameters, while keeping the performance degradation to less than 5%.

2 Method

We propose hypernetwork-assisted inter-layer connectors (i.e., LayerConnect), as shown in Figure 1, the Transformer encoder (Vaswani et al., 2017) is used as the backbone of our LayerConnect and a connector is inserted between Transformer layers, named inter-layer connector. Specifically, the connectors linearly transform the hidden states of each Transformer layer:

$$C^l(X_l) = X_l \cdot A^l + B^l \tag{1}$$

where $\boldsymbol{X}_l \in \mathbb{R}^{n\times d}$ is the hidden state of the l-th Transformer layer ($l \in \{1, 2, \cdots, L\}$), n and d are the input size and the hidden size of Transformer, respectively. Further, $A^l \in \mathbb{R}^{n\times n}$ and $B^l \in \mathbb{R}^{n\times d}$ are the trainable parameters where each row in A^l or B^l is a repetition of a vector a^l or b^l with length n. In the entire model, the only tunable parameters are $\boldsymbol{A} = [a^1; a^2; \cdots; a^L]$ and $\boldsymbol{B} = [b^1; b^2; \cdots; b^L]$. Compared with previous intra-layer Adapters (Houlsby et al., 2019; Karimi Mahabadi et al., 2021), our inter-layer connector contains fewer trainable parameters due to (1) the number of inserted layers in our model is L while the counterpart of others is $2L$; and (2) the

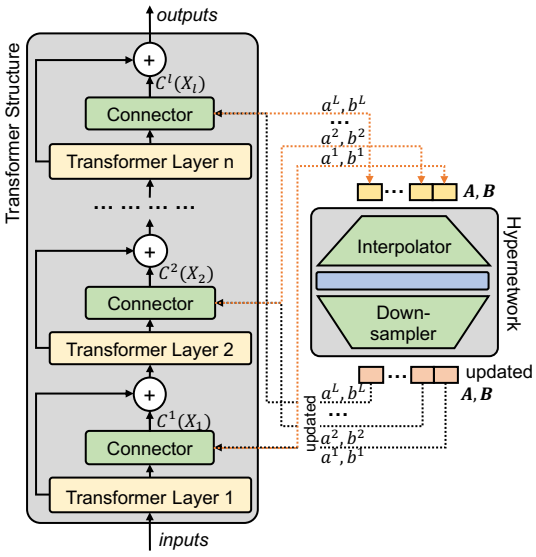

Figure 1: LayerConnect architecture.

parameters count in our connector is $n+d$, which is less than $2d \cdot d_m (1 \leq d_m < d)$ existing Adapters.

To prevent the model from converging to the local optima due to the small number of parameters, the weights of $\boldsymbol{A}$ and $\boldsymbol{B}$ are conditioned on a hypernetwork. Unlike the approach of Karimi Mahabadi et al. (2021), whose hypernetwork is a single linear projector, we design a new hypernetwork equipped with an interpolator and a down-sampler. In detail, we first utilize a learned task embedding $e_t \in \mathbb{R}^{L\times d}$ for each task, and then, use an interpolator to raise dimensionality, followed by a ReLU non-linearity:

$$\boldsymbol{e}_{in} = \mathrm{ReLU}(W_{in} e_t) \tag{2}$$

where $W_{in} \in \mathbb{R}^{mL\times L}$ is the set of trainable parameters of the interpolator and m is a hyper parameter. In this manner, sufficient interpolator parameters are provided to optimize the connectors. Subsequently, a down-sampler is used to project $\boldsymbol{X}_{in}$ back to the original dimensionality. Finally, the weights of $\boldsymbol{A}$ and $\boldsymbol{B}$ are generated from the down-sampled vector:

$$\boldsymbol{e}_{ds} = W_{ds} \boldsymbol{e}_{in} \tag{3}$$

$$(\boldsymbol{A}^\top, \boldsymbol{B}^\top) = \boldsymbol{X}_{ds}(W^A, W^B) \tag{4}$$

where $W_{ds} \in \mathbb{R}^{L\times mL}$, $W^A \in \mathbb{R}^{d\times n}$ and $W^B \in \mathbb{R}^{d\times d}$ are trainable parameters.

We specify the parameters of hypernetwork for different tasks during fine tuning. After that, the hypernetwork is discarded when deploying models, indicating that the parameters of the hypernetwork do not influence storage space during reasoning.

Method	Para. Size	SST-2	QNLI	M.-MM	QQP		RTE	STS-B		CoLA	MRPC	
		Acc.	Acc.	Acc.	Acc.	F1	Acc.	PCC	SCC	Acc.	Acc.	F1
Vanilla Fine-Tuning	108.31M	91.46	91.00	83.91	91.08	87.93	69.67	89.73	89.35	60.74	86.17	90.12
ADA_1	27.66K	90.60	88.14	78.77	85.54	80.51	63.82	86.99	86.97	51.12	77.79	85.22
ADA_8-random	13.06K	90.51	86.34	72.54	82.71	77.90	58.48	83.63	83.25	34.36	72.07	82.67
ADA_8	156.76K	91.48	90.84	81.46	88.11	83.98	69.82	**88.08**	**88.08**	**57.12**	84.46	88.94
ADA_8-hyper	156.76K	91.40	**90.87**	81.96	**88.58**	**84.71**	**70.51**	88.07	88.07	56.01	85.29	89.42
Bitfit	102.91K	90.78	87.70	78.15	85.50	80.90	67.22	87.22	87.22	47.92	78.14	85.67
LayerCon. (our)	18.43K	**92.45**	88.58	78.96	85.73	81.16	68.73	87.46	87.63	53.86	**86.17**	**90.11**
LayerCon. (w/o B)	9.21K	90.52	87.13	75.62	84.71	79.79	64.98	85.86	85.74	49.72	80.94	87.18
LayerCon. (w/o A)	9.21K	89.48	85.25	75.63	82.49	77.97	57.40	70.10	81.25	42.98	69.61	81.66
LayerCon. (w/o A&B)	18.43K	89.79	84.79	**82.55**	83.06	78.43	58.19	84.63	84.63	40.94	74.21	83.11

Table 1: Experimental results on the GLUE benchmark, based on BERT-base. M.-MM = MNLI-MM. Para. Size: additional parameters for PLM inference in a single task. The **bold** denotes the best results.

Method	Para. Size	SST-2	QNLI	M.-MM	QQP		RTE	STS-B		CoLA	MRPC	
		Acc.	Acc.	Acc.	Acc.	F1	Acc.	PCC	SCC	Acc.	Acc.	F1
Vanilla Fine-Tuning	125.31M	94.65	93.11	87.16	91.49	88.70	77.36	90.00	89.79	59.66	88.57	91.75
ADA_1	27.66K	92.09	88.19	82.48	85.74	80.95	58.34	81.93	81.99	37.82	77.09	81.92
ADA_8-random	13.06K	90.51	80.97	75.46	82.69	77.82	58.53	87.38	87.26	30.09	68.38	81.22
ADA_8	156.76K	93.85	90.90	**86.63**	**88.29**	**84.56**	70.18	**90.63**	**90.31**	58.92	**87.60**	**91.08**
ADA_8-hyper	156.76K	**94.50**	**91.77**	85.13	87.30	83.23	**70.21**	88.86	89.22	**61.14**	84.74	88.66
Bitfit	103.68K	90.60	88.49	75.79	85.23	81.08	67.22	84.56	84.41	54.17	80.88	86.15
LayerCon. (our)	18.43K	93.58	89.51	83.57	87.04	82.98	66.13	86.74	86.85	55.17	79.88	85.67
LayerCon. (w/o B)	9.21K	91.28	87.59	80.66	85.34	81.03	59.01	83.08	83.45	41.86	75.29	83.66
LayerCon. (w/o A)	9.21K	92.66	84.57	71.52	83.55	79.46	60.72	78.09	78.38	43.27	70.10	81.68
LayerCon. (w/o A&B)	18.43K	91.50	86.73	72.76	85.90	81.91	63.29	84.66	83.55	46.56	73.70	82.21

Table 2: Experimental results on GLUE benchmark, based on RoBERTa-base.

Therefore, we only need to restore the connector parameters, i.e., $\boldsymbol{A}$ and $\boldsymbol{B}$, for each task.

3 Experiment

3.1 Setup

The PLMs used in our experiments are BERT and RoBERTa models implemented by the *Transformers* library (Wolf et al., 2020). More details are given in Appendix A.

3.2 Dataset and Metric

Following previous work (Houlsby et al., 2019; Karimi Mahabadi et al., 2021), we evaluate model performance on the GLUE benchmark (Wang et al., 2018). Note that some test sets in GLUE are not publicly available, thus, the corresponding validation sets are used as alternatives. The main metric is accuracy (Acc.). For the QQP and MRPC, we use the F1-measure; for the STS-B, we use the Spearman and Pearson correlation coefficients (SCC and PCC).

3.3 Baselines

We compared our model with the following methods. (1) Vanilla Fine-Tuning: Fine-tuning all parameters in PLMs; (2) Adapter (Houlsby et al., 2019) under different settings: specifically, we adjust the middle size of its bottleneck structure to 1 and 8, which we denoted as ADA_1 and ADA_8, respectively; (3) ADA-random: We randomly add an Adapter layer[1] to the PLM as a baseline. (4) ADA_8-hyper: We equip the original Adapter of size 8 with our hypernetwork. (5) Bitfit (Zaken et al., 2021).

Moreover, we also modify our LayerConnect (LayerCon.) into the following three variations: (1) LayerCon. (w/o B) uses only the hypernetwork to generate the weights of $\boldsymbol{A}$; (2) LayerCon. (w/o A) uses only the hypernetwork to generate the weights of $\boldsymbol{B}$; (3) LayerCon. (w/o A&B) removes the hypernetwork and randomly initializes the weights of $\boldsymbol{A}$ and $\boldsymbol{B}$.

3.4 Results

We first analyze the effectiveness of LayerConnect and hypernetwork through the experimental results of the base model (Table 1 and Table 2).

Effectiveness of LayerConnect. Our LayerConnect outperforms Bitfit in most of the benchmark datasets. For the Adapter baselines, a larger Adapter achieves the higher performance (ADA_1 vs. ADA_8). As a strong baseline, Adapter even outperforms the vanilla fine-tuning in some benchmark datasets. However, when the Adapter is equipped with parameters similar to those of LayerConnect, its performance declined (ADA_8-random

[1]We choose to add only one Adapter layer since the trainable parameters in this strategy are most close to our model.

Method	Para. Size	SST-2	QNLI	M.-MM	QQP		RTE	STS-B		CoLA	MRPC	
		Acc.	Acc.	Acc.	Acc.	F1	Acc.	PCC	SCC	Acc.	Acc.	F1
Vanilla Fine-Tuning	333.58M	93.12	92.36	86.55	91.44	88.43	75.45	90.08	90.17	65.12	87.74	91.28
ADA$_1$	76.82K	92.09	91.62	83.30	87.37	83.04	68.23	89.63	89.53	58.81	71.08	81.73
ADA$_8$-random	36.88K	91.97	83.56	77.52	85.25	80.83	64.34	84.58	85.13	57.54	78.43	85.53
ADA$_8$	417.98K	**92.93**	92.11	**86.60**	89.17	**86.30**	72.99	**90.19**	**90.06**	61.07	86.12	90.18
ADA$_8$-hyper	417.98K	92.84	**92.12**	85.90	**89.72**	86.28	**75.71**	90.03	89.51	**62.70**	**87.84**	**91.35**
Bitfit	272.38K	92.20	90.03	85.90	87.15	83.63	70.03	89.90	89.65	57.79	79.75	90.21
LayerCon. (our)	36.88K	92.59	91.10	83.36	85.72	84.43	72.42	90.08	87.63	60.90	87.50	91.24
LayerCon. (w/o B)	18.43K	92.09	89.71	81.11	85.03	80.76	68.95	88.12	88.87	57.02	80.15	86.61
LayerCon. (w/o A)	18.43K	91.86	89.99	80.67	84.70	80.24	67.15	88.50	88.25	58.58	82.35	82.04
LayerCon. (w/o A&B)	18.43K	91.97	89.86	81.77	84.56	80.05	59.21	87.45	87.42	55.63	76.47	84.81

Table 3: Experimental results on GLUE benchmark, based on BERT-Large.

Method	Para. Size	SST-2	QNLI	M.-MM	QQP		RTE	STS-B		CoLA	MRPC	
		Acc.	Acc.	Acc.	Acc.	F1	Acc.	PCC	SCC	Acc.	Acc.	F1
Vanilla Fine-Tuning	356.51M	96.61	94.94	89.98	91.92	89.45	85.12	92.28	92.09	65.08	88.48	91.75
ADA$_1$	76.82K	95.07	91.96	88.17	86.61	82.51	71.12	90.63	90.54	55.59	86.27	90.48
ADA$_8$-random	36.88K	95.18	75.97	84.94	84.67	80.43	73.29	88.48	88.53	52.06	83.33	88.51
ADA$_8$	417.98K	95.94	94.56	89.66	89.25	85.97	72.74	**92.01**	91.08	62.75	89.07	92.17
ADA$_8$-hyper	417.98K	**96.58**	**94.73**	**90.03**	**89.34**	**86.03**	73.19	90.05	89.88	59.06	**89.46**	**92.31**
Bitfit	273.41K	95.37	93.04	88.46	87.49	83.79	72.22	91.05	90.91	59.70	86.64	90.21
LayerCon (our)	36.88K	95.41	93.67	88.60	88.04	83.00	**76.92**	91.50	**91.46**	**63.61**	87.00	90.21
LayerCon. (w/o B)	18.43K	93.23	90.85	86.44	86.24	82.32	73.29	86.34	86.32	54.23	70.83	81.89
LayerCon. (w/o A)	18.43K	93.35	92.29	86.90	85.77	85.99	70.76	86.55	86.57	53.95	82.50	83.86
LayerCon. (w/o A&B)	36.88K	94.61	92.36	87.28	86.24	81.92	68.23	82.91	83.56	56.53	77.45	85.40

Table 4: Experimental results on GLUE benchmark, based on RoBERTa-Large.

Para. Size	CoLA(Acc.)	MNLI-MM(Acc.)
2m.	55.17	83.57
4m.	55.47	84.01
10m.	55.48	84.22

Table 5: The effect of the size of hypernetwork under the RoBERTa-base model.

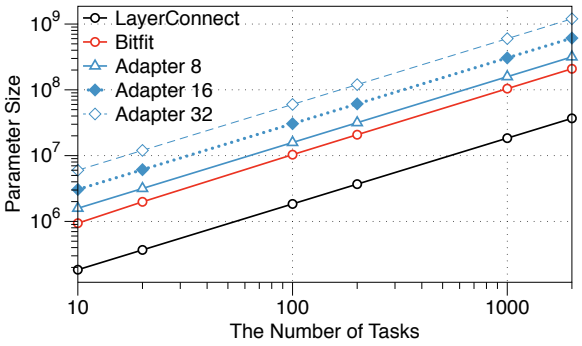

Figure 2: Parameter size as the number of tasks grows.

vs. ADA$_8$) and is worse than ours on most datasets. Moreover, the trainable inserted parameters in our model are only 11.75% of that in ADA$_8$. The LayerConnect performance is competitive with the best Adapter, with an average performance deterioration of 1.42% (1.0 points) and 3.96% (3.3 points) for BERT-base and RoBERTa-base, respectively.

Effectiveness of the Hypernetwork. We also demonstrate that the hypernetwork is necessary for the connectors. We compare the results of our model with or without hypernetwork (LayerCon. vs. LayerCon. (w/o A&B)). The former outperforms the latter except for MNLI-MM. We analyze this because the hypernetwork provides sufficient parameters during the fine-tuning stage to tune the connectors effectively. We also remove the $\boldsymbol{A}$ and $\boldsymbol{B}$ in our hypernetwork, respectively, to further reduce the parameters. However, the performances are all worse than that of the original hypernetwork (LayerCon. vs. LayerCon. (w/o B) and LayerCon. (w/o A)), demonstrating the rationality of our hypernetwork. Moreover, we attempt to equip the best Adapter with our hypernetwork, the results are only slightly changed compared with the original Adapter (ADA$_8$ vs. ADA$_8$-hyper). We conclude that our hypernetwork is more effective on fewer parameters' inserted layers since its provided parameters are more valuable to them.

We additionally test our model and baselines based on BERT-large and RoBERTa-large models. The corresponding results are shown in Table 3 and Table 4, respectively. The average performance deterioration is 2.12% (1.74 points) and 0.79% (0.85 points) for BERT-large and RoBERTa-large, respectively; both are lower than that of the base models. The trainable inserted parameters in LayerConnect are only 8.82% of that in ADA$_8$.

4 Discussion

As presented in Table 5, we adjust the middle value of the hypernetwork shape in our method to 4 and

10, respectively. A larger size resulted in a slight performance improvement. This result suggests that the hypernetwork size has a positive effect in our LayerConnect.

We compare the parameter size in the deployment for model inference as the number of tasks grows, as shown in Figure 2. In model inference, considering the scalability, for the Bitfit, one may at least store multiple copies of the bias (i.e., 1,324K). However, the structure and the bias locations are also necessary. The parameter sizes of the Adapter and the Adapter with the hypernetwork are the same (i.e., the size of the bottleneck structure). In contrast, our connectors require an extremely small parameter size. That is to say, one can easily deploy several tens of the tasks, and the total size of the parameters just arrives at the same level as that of a single task when using the others.

5 Conclusion

To enhance the parameter efficiency in the fine-tuning stage of PLMs, we propose the ultra-light connectors to be embedded into the Transformer layers. Furthermore, to keep the performance of such a small structure, we use the hypernetwork to assist the tuning of the parameters within the connectors. We compare our method with mainstream methods (Adapter and Bitfit). Experimental results show that our method outperforms Bitfit in most cases. Compared with ADA_8 (with best performance in most cases), our method reduces the trainable parameters to 11.75% for base models and 8.82% for large models, while keeping the performance degradation to less than 5% (2.07 points on average). By analyzing the results, we verify that reducing the number of parameters on the basis of Adapter will seriously reduce the performance, and the introduction of hypernetwork promises an effective way to compensate for performance. This reveals a new direction of the study on model efficiency. Additionally, LayerConnect shows scalability, especially for the memory-sensitive and storage-sensitive edge devices e.g., smartphones, embedded devices, micro containers, and IoT/IoH devices.

References

Weijie Chen, Yongzhu Chang, Rongsheng Zhang, Jiashu Pu, Guandan Chen, Le Zhang, Yadong Xi, Yijiang Chen, and Chang Su. 2022. Probing simile knowledge from pre-trained language models. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 5875–5887, Dublin, Ireland. Association for Computational Linguistics.

Ning Ding, Yujia Qin, Guang Yang, Fuchao Wei, Zonghan Yang, Yusheng Su, Shengding Hu, Yulin Chen, Chi-Min Chan, Weize Chen, et al. 2022. Delta tuning: A comprehensive study of parameter efficient methods for pre-trained language models. *arXiv preprint arXiv:2203.06904*.

David Ha, Andrew Dai, and Quoc V Le. 2016. Hypernetworks. *arXiv preprint arXiv:1609.09106*.

Wanwei He, Yinpei Dai, Yinhe Zheng, Yuchuan Wu, Zheng Cao, Dermot Liu, Peng Jiang, Min Yang, Fei Huang, Luo Si, et al. 2022. Galaxy: A generative pre-trained model for task-oriented dialog with semi-supervised learning and explicit policy injection. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 36, pages 10749–10757.

Neil Houlsby, Andrei Giurgiu, Stanislaw Jastrzebski, Bruna Morrone, Quentin De Laroussilhe, Andrea Gesmundo, Mona Attariyan, and Sylvain Gelly. 2019. Parameter-efficient transfer learning for nlp. In *International Conference on Machine Learning*, pages 2790–2799. PMLR.

Edward J Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. 2021. Lora: Low-rank adaptation of large language models. *arXiv preprint arXiv:2106.09685*.

Rabeeh Karimi Mahabadi, Sebastian Ruder, Mostafa Dehghani, and James Henderson. 2021. Parameter-efficient multi-task fine-tuning for transformers via shared hypernetworks. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 565–576, Online. Association for Computational Linguistics.

Siyu Lai, Hui Huang, Dong Jing, Yufeng Chen, Jinan Xu, and Jian Liu. 2021. Saliency-based multi-view mixed language training for zero-shot cross-lingual classification. In *Findings of the Association for Computational Linguistics: EMNLP 2021*, pages 599–610, Punta Cana, Dominican Republic. Association for Computational Linguistics.

Xiang Lisa Li and Percy Liang. 2021. Prefix-tuning: Optimizing continuous prompts for generation. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 4582–4597, Online. Association for Computational Linguistics.

Xiao Liu, Yanan Zheng, Zhengxiao Du, Ming Ding, Yujie Qian, Zhilin Yang, and Jie Tang. 2021. GPT understands, too. *CoRR*, abs/2103.10385.

Xipeng Qiu, Tianxiang Sun, Yige Xu, Yunfan Shao, Ning Dai, and Xuanjing Huang. 2020. Pre-trained models for natural language processing: A survey. *Science China Technological Sciences*, 63(10):1872–1897.

Colin Raffel, Noam M. Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J. Liu. 2020. Exploring the limits of transfer learning with a unified text-to-text transformer. *ArXiv*, abs/1910.10683.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Ł ukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc.

Alex Wang, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel Bowman. 2018. GLUE: A multi-task benchmark and analysis platform for natural language understanding. In *Proceedings of the 2018 EMNLP Workshop BlackboxNLP: Analyzing and Interpreting Neural Networks for NLP*, pages 353–355, Brussels, Belgium. Association for Computational Linguistics.

Jiaan Wang, Zhixu Li, Qiang Yang, Jianfeng Qu, Zhigang Chen, Qingsheng Liu, and Guoping Hu. 2021. Sportssum2.0: Generating high-quality sports news from live text commentary. In *Proceedings of the 30th ACM International Conference on Information & Knowledge Management*, page 3463–3467, New York, NY, USA. Association for Computing Machinery.

Jiaan Wang, Zhixu Li, Tingyi Zhang, Duo Zheng, Jianfeng Qu, An Liu, Lei Zhao, and Zhigang Chen. 2022. Knowledge enhanced sports game summarization. In *Proceedings of the Fifteenth ACM International Conference on Web Search and Data Mining*, WSDM '22, page 1045–1053, New York, NY, USA. Association for Computing Machinery.

Yida Wang, Pei Ke, Yinhe Zheng, Kaili Huang, Yong Jiang, Xiaoyan Zhu, and Minlie Huang. 2020. A large-scale chinese short-text conversation dataset. In *CCF International Conference on Natural Language Processing and Chinese Computing*, pages 91–103. Springer.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Remi Louf, Morgan Funtowicz, Joe Davison, Sam Shleifer, Patrick von Platen, Clara Ma, Yacine Jernite, Julien Plu, Canwen Xu, Teven Le Scao, Sylvain Gugger, Mariama Drame, Quentin Lhoest, and Alexander Rush. 2020. Transformers: State-of-the-art natural language processing. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 38–45, Online. Association for Computational Linguistics.

Elad Ben Zaken, Shauli Ravfogel, and Yoav Goldberg. 2021. Bitfit: Simple parameter-efficient fine-tuning for transformer-based masked language-models. *CoRR*, abs/2106.10199.

Rongsheng Zhang, Yinhe Zheng, Xiaoxi Mao, and Minlie Huang. 2021. Unsupervised domain adaptation with adapter. *arXiv preprint arXiv:2111.00667*.

Rongsheng Zhang, Yinhe Zheng, Jianzhi Shao, Xiaoxi Mao, Yadong Xi, and Minlie Huang. 2020. Dialogue distillation: Open-domain dialogue augmentation using unpaired data. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 3449–3460.

Duo Zheng, Zipeng Xu, Fandong Meng, Xiaojie Wang, Jiaan Wang, and Jie Zhou. 2021a. Enhancing visual dialog questioner with entity-based strategy learning and augmented guesser. In *Findings of the Association for Computational Linguistics: EMNLP 2021*, pages 1839–1851, Punta Cana, Dominican Republic. Association for Computational Linguistics.

Yinhe Zheng, Guanyi Chen, Xin Liu, and Ke Lin. 2021b. Mmchat: Multi-modal chat dataset on social media. *arXiv preprint arXiv:2108.07154*.

Yinhe Zheng, Zikai Chen, Rongsheng Zhang, Shilei Huang, Xiaoxi Mao, and Minlie Huang. 2021c. Stylized dialogue response generation using stylized unpaired texts. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 35, pages 14558–14567.

Yinhe Zheng, Rongsheng Zhang, Minlie Huang, and Xiaoxi Mao. 2020. A pre-training based personalized dialogue generation model with persona-sparse data. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pages 9693–9700.

Hao Zhou, Pei Ke, Zheng Zhang, Yuxian Gu, Yinhe Zheng, Chujie Zheng, Yida Wang, Chen Henry Wu, Hao Sun, Xiaocong Yang, et al. 2021. Eva: An open-domain chinese dialogue system with large-scale generative pre-training. *arXiv preprint arXiv:2108.01547*.

A Setup and Evaluation Metric

A.1 Setup

Connectors: The LayerConn.(ours), LayerConn.(w/o B) and LayerConn.(w/o A) are shown as Equation 5 to Equation 7, respectively:

$$C^l(X_l) = X_l \cdot A^l + B^l \tag{5}$$

$$C^l(X_l) = X_l \cdot A^l \tag{6}$$

$$C^l(X_l) = B^l \tag{7}$$

where $\boldsymbol{X}_l \in \mathbb{R}^{n\times d}$, $A^l \in \mathbb{R}^{n\times n}$ and $B^l \in \mathbb{R}^{n\times d}$, $l \in \{1, 2, \cdots, L\}$) is the layer ID. For both Bert-base and RoBERTa-base model, $L = 12$; for both Bert-large and RoBERTa-large model, $L = 24$. Each row in A^l or B^l is a repetition of a vector a^l or b^l with length n. The only tunable parameters are $\boldsymbol{A} = [a^1; a^2; \cdots; a^L]$ and $\boldsymbol{B} = [b^1; b^2; \cdots; b^L]$. n and d are the input size (max. length) and the hidden size of Transformer, respectively. For both Bert-base and RoBERTa-base model, n is 128, and d is 768; for both Bert-large and RoBERTa-large model, n is 128, and d is 1024.

Hypernetwork: Aa interpolator can be expressed as:

$$\boldsymbol{e}_{in} = ReLU(W_{in}e_t) \tag{8}$$

and a down-sampler can be expressed as:

$$\boldsymbol{e}_{ds} = W_{ds}\boldsymbol{e}_{in} \tag{9}$$

where $W_{in} \in \mathbb{R}^{mL\times L}$, and $W_{ds} \in \mathbb{R}^{L\times mL}$, they are the trainable parameters of the interpolator and the down-sampler, respectively. m is a hyper parameter. In the experiment, the minimum $m = 2$.

PLMs: The learning rates are 2×10^{-5} and 5×10^{-6} for BERT (base and large) and RoBERTa (base and large), respectively. We select AdamW as the optimizer. We choose random seeds for each model and report the average results of five runs.

A.2 Evaluation Metric

The metrics vary across different tasks, expect for the commonly used accuracy and F1 Score. We briefly introduce other metrics as follows:

Pearson Correlation Coefficient. The Pearson Correlation Coefficient is used to calculate the similarity of sentence pair from S_x and S_y, as shown in Equation 10:

$$\rho_{S_x,S_y} = \frac{E[(S_x - \mu_{S_x})(S_y - \mu_{S_y})]}{\sigma_{S_x}\sigma_{S_y}} \tag{10}$$

where μ and σ are the mean and standard deviation for S_x and S_y, respectively.

Spearman Correlation Coefficient. The Spearman Correlation Coefficient is another way to calculate the similarity of sentence pair from S_x and S_y. Its format is the same with Pearson one. However, the samples are transformed into the level variable, and the calculation is simplified as shown in Equation 11:

$$\rho = \frac{6\sum_{i=1}^{n}(S_x^{(i)} - S_y^{(i)})^2}{N(N^2-1)} \tag{11}$$

where $S_x^{(i)}$ and $S_y^{(i)}$ are sentence sample from S_x and S_y, and N is the total number of the samples.

Matthews Correlation Coefficient (MCC). The performance of CoLA is evaluated by MCC, as shown in Equation 12:

$$\begin{aligned} MCC &= \frac{TP \cdot TN - FP \cdot FN}{\sqrt{Q}} \\ Q &= (TP + FP) \cdot (TP + FN) \cdot \\ &\quad (TN + FP) \cdot (TN + FN) \end{aligned} \tag{12}$$

where TP is true positive, TN is true negative, FP is false positive, and FN is false negative.

A.3 Performance Degradation

The performance degradation is calculated by Equation 13:

$$\text{Degradation} = \frac{\text{BaselineScore} - \text{OurScore}}{\text{BaselineScore}} \tag{13}$$

Negative degradation means an improvement.

Effect of Post-processing on Contextualized Word Representations

Hassan Sajjad[♣*] **Firoj Alam**[♢] **Fahim Dalvi**[♢] **Nadir Durrani**[♢]
[♣]Faculty of Computer Science, Dalhousie University, Canada
[♢]Qatar Computing Research Institute, Hamad Bin Khalifa University, Qatar
hsajjad@dal.ca, {fialam,faimaduddin, ndurrani}@hbku.edu.qa

Abstract

Post-processing of static embedding has been shown to improve their performance on both lexical and sequence-level tasks. However, post-processing for contextualized embeddings is an under-studied problem. In this work, we question the usefulness of post-processing for contextualized embeddings obtained from different layers of pre-trained language models. More specifically, we standardize individual neuron activations using *z-score*, *min-max* normalization, and by removing top principal components using the *all-but-the-top* method. Additionally, we apply *unit length* normalization to word representations. On a diverse set of pre-trained models, we show that post-processing unwraps vital information present in the representations for both lexical tasks (such as word similarity and analogy) and sequence classification tasks. Our findings raise interesting points in relation to the research studies that use contextualized representations, and suggest *z-score* normalization as an essential step to consider when using them in an application.

1 Introduction

Contextualized word embeddings have been central to the recent revolution in NLP, achieving state-of-the-art performance for many tasks. They are commonly used in the form of fine-tuning based transfer learning and feature extraction (Peters et al., 2018, 2019).[1] Feature-based approach generates contextualized embedding vectors and that are used as frozen features in a classifier towards a downstream task. A similar pipeline is used for static embedding except that here a word acquires different embeddings depending on the context it appears in. While fine-tuning based is the more commonly used method, feature-based approach has shown to be a viable alternative with many applications (Peters et al., 2019). For example, it has been used as a tool to analyze the inner learning of pre-trained contextualized models (Dalvi et al., 2017; Liu et al., 2019a; Belinkov et al., 2020).

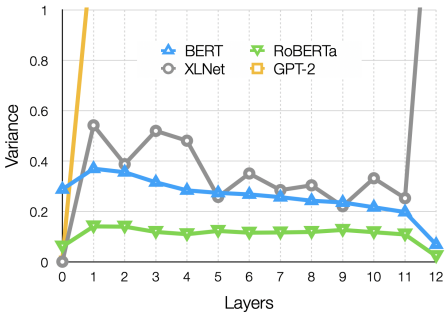

Figure 1: Layer-wise variance

The literature on static embedding has emphasized the usefulness of post-processing of embeddings to maximize their performance on the downstream tasks. For example, Mu and Viswanath (2018) showed that making static embedding isotropic is beneficial to lexical and sentence-level tasks. Similarly, Levy et al. (2015); Wilson and Schakel (2015) showed that using normalized word vectors improve performance on word similarity and word relation tasks.

While the efficacy of post-processing has been empirically demonstrated for static embedding, it has not been studied whether it can be beneficial when applied to the representations extracted from the contextualized models such as BERT (Devlin et al., 2019), GPT2 (Radford et al., 2018, 2019), etc. Ethayarajh (2019) found contextualized word representations to be anisotropic. Given that isotropy is a desirable property with proven benefits in the static embedding arena, would encouraging isotropism in the contextualized embeddings also result in performance gains?

Similarly, different layers of a pre-trained model exhibit different variance patterns, particularly the first and last layers (see Figure 1[2] where we plot

[*]The work was done while the author was at QCRI

[1]We have used feature-based transfer learning for this approach in the paper.

[2]The variance of the middle layers of GPT-2 was very high.

Proceedings of the 29th International Conference on Computational Linguistics, pages 3127–3142
October 12–17, 2022.

this for several pre-trained models). When used for feature-based transfer learning, a high variance in features may result in sub-optimal and misleading results (Kaufman and Rousseeuw, 1990). Would normalizing the variance in contextualized embeddings be beneficial for their applications? It is important to address these questions to unwrap the full potential of contextualized embeddings when used for feature-based learning. In the context of analyzing pre-trained models, they enable a fair comparison between different layers and models.

In this paper, we make a pioneering attempt on the realm of post-processing contextual representation. To this end, we analyze the effect of four post-processing methods on contextualized embeddings. More specifically, we standardize features (individual neurons) using (i) z-score normalization (`zscore`), (ii) min-max normalization (`minmax`), and (iii) all-but-the-top (`abtt`) post-processing method (Mu and Viswanath, 2018). We also post-process word representations using unit length normalization (`ulen`). The first two are standard feature normalization methods shown to be an effective pre-processing step in building a successful machine learning model (Jiawei Han, 2011). `abtt` removes top principal components of contextualization representations (Ethayarajh, 2019) and improve isotropy of the representations. `ulen` has shown to be effective in improving the performance of static embedding for word similarity and analogy tasks (Levy et al., 2015).

We consider contextualized embeddings of a variety of pre-trained models covering both auto-encoder and auto-regressive in design. We evaluate the effect of post-processing contextualized embeddings using various lexical-level tasks such as word similarity, word analogy, and using several sequence classification tasks from the GLUE benchmark. Our results show that:

- Post-processing helps to unwrap the information present in the representations
- The major improvement achieved for the last layer shows that while it is optimized for the objective function, it still possesses most of the knowledge learned in the previous layers
- z-score and all-but-the-top are orthogonal and results in best performance when used in tandem for lexical tasks
- z-score achieves substantial improvement on the sequence classification tasks, particularly using the representations from the middle and higher layers

We further discuss the relation between post-processing of contextualized embeddings and the research on representation analysis. In a preliminary experiment on analyzing individual neurons in pre-trained models, we show that post-processing enables a fair comparison among neurons of various layers. For example, after post-processing, we find that the last layer of BERT also has a substantial contribution towards the top neurons learning part of speech properties. Supported by our results, we recommend that normalization of a layer representation should be considered as an essential first step before using contextualized embeddings for end applications such as transfer learning and interpretation of representations.

The paper is organized as follows: Section 2 accounts for related work. Section 3 describes our approach and post-processing strategies. Section 4 presents the experimental setup. Section 5 reports our findings. We carry out the discussion supported by an application in Section 6. Section 7 concludes the paper.

2 Related Work

Static Embedding Normalization A number of post-processing methods have been proposed to improve the performance of static embedding such as, length normalization (Levy et al., 2015), centering the mean (Sahlgren et al., 2016), and removing the top principal components (Mu and Viswanath, 2018; Arora et al., 2017). Arora et al. (2017) removed the first principal component where components are dataset specific as they compute the representations for the entire dataset. On the other hand (Mu and Viswanath, 2018) removed the top k components by computing such representations on the entire vocabulary. They assume that higher variance components are corrupted by the information which is different than lexical semantics. Wang et al. (2019) proposed two normalization methods *(i)* variance normalization – normalizes the principle components of the pre-trained word vectors, *(ii)* dynamic embedding – learns the orthogonal latent variables from the ordered input sequence. The post-processed static representations are then evaluated on both intrinsic and extrinsic tasks, which demonstrates the effectiveness of these methods.

We limit the y-axis to show the pattern of variance of other models.

Contextualized Embeddings In the context of representations of contextual pre-trained models, the effectiveness of post-processing methods have not been explored. Most of the work that uses contextualized representations use them without any pre-processing. In this work, we explore the usefulness of two commonly used post-processing methods on the embeddings extracted from pre-trained models.

Peters et al. (2018, 2019) used contextualized embeddings in feature-based setting for several sequence classification tasks. Similar to static embedding, the contextualized embeddings are used as input to an LSTM-based classifier. However, a word can have different embedding depending on the context. In addition, a plethora of work on the analysis and interpretation of deep models used feature-based approach to probe the linguistic information encoded in the representations (Belinkov et al., 2017b; Conneau et al., 2018b; Liu et al., 2019a; Tenney et al., 2019b,a; Voita et al., 2019; Durrani et al., 2019; Arps et al., 2022). The most common approach uses probing classifiers (Ettinger et al., 2016; Belinkov et al., 2017a; Adi et al., 2017; Conneau et al., 2018a; Hupkes et al., 2018), where a classifier is trained on a corpus of linguistic annotations using representations from the model under investigation. For example, Liu et al. (2019a) used this methodology for investigating the representations of contextual word representations on 16 linguistic tasks. Dalvi et al. (2019); Durrani et al. (2020) expanded the work on representation analysis[3] to neuron-level analysis. Similar to probing classifier used in the representation analysis, they used a linear classifier with ElasticNet regularization. Recently, Sajjad et al. (2022b); Dalvi et al. (2022) introduced an unsupervised method that clusters contextualized representations of words to analyze the representations.

An orthogonal analysis comparing models and their representations relies on similarities between model representations. Bau et al. (2019) used this approach to analyze the role of individual neurons in neural machine translation. Wu et al. (2020) compared representations of several pre-trained models using various similarity methods. Another class of work on understanding the contextualized representations looked at the social bias encoded in the representations (Bommasani et al., 2020).

[3]Please see (Belinkov and Glass, 2019; Sajjad et al., 2022a) for comprehensive surveys on representation analysis.

Ethayarajh (2019) provided a different angle to the analysis of contextualized embeddings and explored the geometry of the embedding space. They showed that contextualized embeddings are anistropic and questioned the effectiveness of contextualized representations given the well known benefits of isotropic representations.

The work of Ethayarajh (2019) is in-line with ours where they studied the geometry of contextualized representations. Our work builds on top of their analysis and provides an empirical evidence supporting post-processing of embeddings. In relevance to the above mentioned studies on analyzing representation of deep models, our work has direct implications on their findings. The effect of post-processing is dependent on various factors such as, the model used for feature-based transfer learning, and the goal of the task. We suggest that post-processing particularly the z-score normalization should be considered as an essential step while designing experiments using the contextualized embeddings.

3 Approach

Consider a pre-trained model $\mathbb{M}$ with L layers $l_1, l_2, ..., l_L$.[4] Let $\mathbb{D}$ be a dataset composed of n_w words of interest, $w_1, w_2, ..., w_N$. Our model $\mathbb{M}$ encodes input tokens depending on their context. We first normalize each contextualized embedding using various post-processing methods. For each word w_i, we then form a single representation, similar to a static embedding, for every layer in L. We evaluate the word representation on lexical-level tasks and sequence classification tasks. In the latter, we train a BiLSTM classifier pre-initialized with our processed embedding.

3.1 Word Representations

Let s_{w_i} be a sentence containing the word w_i. Let $z^l_{w_i,s_{w_i}}$ represents the contextual embedding from layer l of model $\mathbb{M}$ for the word w_i, with the given context s_{w_i}, specifically,

$$z^l_{w_i,s_{w_i}} = \mathbb{M}(s_{w_i})[w_i]_l \in \mathcal{R}^d \quad (1)$$

where $\mathbb{M}(s_{w_i})$ is a shorthand for the contextual embeddings for all tokens in s, from which we pick only one that is relevant to the word of interest w_i, and further index into a specific layer l. d is the number of features (i.e., size of the hidden

[4]We consider each transformer block as a layer.

dimension in a layer), which is 768 per layer in the models analyzed in this paper.

In order to form a single representation for each word,[5] we first extract contextual embeddings for each word w_i in at least C_{min} contexts. These contexts are derived from a large corpus of sentences,[6] and are randomly chosen from all sentences containing the word w_i. We also employ an upper limit C_{max} for the maximum number of contexts used for any word, to avoid the dominance of frequent words such as closed class words. In our analysis, C_{min} is set to 50 and C_{max} is set to 200. Thus, each word in our dataset $\mathbb{D}$ will then have between C_{min} and C_{max} contextual embeddings extracted from the model $\mathbb{M}$. We then aggregate these contextual embeddings by mean pooling each dimension:

$$c^l_{w_i} = \frac{1}{n_c}\sum^{n_c} z^l_{w_i,s_{w_i}} \tag{2}$$

where $C_{min} \leq n_c \leq C_{max}$ is the number of contexts (sentences) the word w_i was present in.

3.2 Post-processing Methods

We perform four kinds of post-processing on the representations $z^l_{w_i,s_{w_i}}$ before aggregating them into $c^l_{w_i}$. They are; (i) z-score normalization, (ii) min-max normalization, (iii) unit length, and (iv) all-but-the-top normalization. The former two are common feature normalization methods used in machine learning.[7] The latter two have shown to be effective in post-processing static embedding. All of these post-processing methods except unit length are applied at the feature-level which is a neuron (single dimension) in our case. The unit length is applied for each word representation.

Let $\mathcal{Z}$ represents the set of all word occurrence embeddings z's for all words in the dataset $\mathbb{D}$.

z-score Normalization (`zscore`) Centering and scaling input vectors to have zero-mean and unit-variance is a common pre-processing practice in many machine learning pipelines. Concretely, each feature's (in our case 1 of 768 dimensions from each layer's representation) mean and variance is computed across all words in our dataset $\mathbb{D}$, followed by subtraction of the mean and division of the standard deviation for each of the feature's value for each embedding $z^l_{w_i,s_{w_i}}$.

$$\mu_l = \frac{1}{|\mathcal{Z}|}\sum_{z\in\mathcal{Z}} z \in \mathcal{R}^d$$

$$\sigma_l = \sqrt{\frac{1}{|\mathcal{Z}|}\sum_{z\in\mathcal{Z}}(z-\mu_l)^2} \in \mathcal{R}^d$$

$$\tilde{z}^l_{w_i,s_{w_i}} = \frac{z^l_{w_i,s_{w_i}} - \mu_l}{\sigma_l} \in \mathcal{R}^d \tag{3}$$

where $\tilde{z}^l_{w_i,s_{w_i}}$ is the post-processed representation of word w_i in context s_{w_i} from layer l.

Min-max Normalization (`minmax`) The min-max normalization rescales each feature range between 0 and 1. Given values of a feature, `minmax` is calculated as follows:

$$\tilde{z}^l_{w_i,s_{w_i}} = \frac{z^l_{w_i,s_{w_i}} - \min(z)}{\max(z) - \min(z)} \in \mathcal{R}^d \tag{4}$$

where $\min$ and $\max$ represent the minimum and maximum values of feature z.

Unit Length (`ulen`) The normalization of word vectors to unit length is shown to be effective for static embedding (Levy et al., 2015). Here, we evaluate its efficacy for contextualized embeddings. Different from other post-processing methods mentioned in this work where we normalize each feature, `ulen` is applied at each word vector i.e., a set of features that represent a vector. We normalize vectors using L_2 norm.

All-but-the-top (`abtt`) Mu and Viswanath (2018) showed that all word representations share a large common vector and similar dominating directions which influence them. Eliminating such directions yields isotropic word representations with better self-normalization properties.

We hypothesize that contextualized embeddings belonging to different layers might be influenced by various design factors e.g., initial layers of BERT may have a strong influence of position embedding, Kovaleva et al. (2019) showed that the last layer of BERT is optimized for the objective function. These factors if dominating the contextual representations may result in sub-optimal performance. We apply `abtt` to eliminate such kind of dominating directions.

[5] We form a single representation to limit the number of tokens. In Section 6, we use contextualized embedding of a word in the application. Our results show that our findings generalize to both static and contextualized embeddings.

[6] We used Wikipedia dump collected on 3rd February 2020.

[7] https://en.wikipedia.org/wiki/Feature_scaling

Algorithm 1: All-but-the-top procedure

Input: Word representations $\{v(z) = z^l_{w_i,s_{w_i}}, z^l_{w_i,s_{w_i}} \in \mathcal{Z}\}$, a threshold parameter k

1 Compute the mean of $\{v(z), z \in \mathcal{Z}\}, \mu \leftarrow \frac{1}{|\mathcal{Z}|}\sum_{z\in\mathcal{Z}} v(z), \tilde{v}(z) \leftarrow v(z) - \mu$

2 Compute the PCA components: $u_1, ..., u_d \leftarrow PCA(\{\tilde{v}(z), z \in \mathcal{Z}\})$.

3 Preprocess the representations: $v'(z) \leftarrow \tilde{v}(z) - \sum_{i=1}^{k}(u_i^T v(z))u_i$

Output: Processed representations $\tilde{z}^l_{w_i,s_j} = v'(z)$.

Algorithm 1 shows the overall process of `abtt` from the original paper, modified with the notation used in this paper. Here, the number of dominating directions to remove is a hyper-parameter k. We used the recommended value of k as suggested by Mu and Viswanath (2018) which is $k = d/100$ where $d = 768$.

4 Training and Evaluation Setup

Data We used the Wikipedia dump of 124 million sentences collected on 3rd February 2020. We tokenized the text using the Moses tokenizer (Koehn et al., 2007). Given a pretrained model, we extracted the contextualized embedding of words from the Wikipedia corpus and generated a single word representation as described in Section 3.1.

Contextualized models We analyzed the contextualized embedding of four 12-layer pretrained models: BERT (Devlin et al., 2019), RoBERTa (Liu et al., 2019b), XLNet (Yang et al., 2019) and GPT2 (Radford et al., 2018, 2019). The former two are auto-encoder in nature while the latter two are auto-regressive.

Lexical-level tasks We used seven word similarity datasets: WordSim353 split into similarity and relatedness (Zesch et al., 2008; Agirre et al., 2009), MEN (Bruni et al., 2012), Mechanical Turk (Radinsky et al., 2011), Rare Words (Luong et al., 2013), SimLex-999 (Hill et al., 2015) and RG65 (Rubenstein and Goodenough, 1965). The datasets contain a word pair with their human annotated similarity scores. The quality of a word embedding is calculated based on the cosine similarity score between a given pair of words, in comparison with their human-provided scores.

Moreover, we used three analogy datasets: MSR (Mikolov et al., 2013b), Google (Mikolov et al., 2013a) and SemEval2012-2 (Jurgens et al., 2012). The analogy tasks involve predicting a word given an analogy relationship like "a is to b" as "c is to d" where d is the word to predict. For more details on each task, we refer the reader to Levy et al. (2015). We used the word embedding benchmark toolkit[8] to evaluate word representations on the word analogy and word similarity tasks.

Sequence classification tasks We evaluated using six General Language Understanding Evaluation (GLUE) tasks (Wang et al., 2018):[9] SST-2 for sentiment analysis with the Stanford sentiment treebank (Socher et al., 2013), MNLI for natural language inference (Williams et al., 2018), QNLI for Question NLI (Rajpurkar et al., 2016), RTE for recognizing textual entailment (Bentivogli et al., 2009), MRPC for Microsoft Research paraphrase corpus (Dolan and Brockett, 2005), and STS-B for the semantic textual similarity benchmark (Cer et al., 2017). We compute statistical significance in performance differences using McNemar test.

We trained a BiLSTM model using Jiant (Phang et al., 2020), with the following parameters settings: vocabulary size 30k, sequence length 150 words, batch size 32, dropout 0.2, hidden layer size 1024, number of layers 2, AMSGRAD, learning rate decay 0.99, minimum learning rate $1e^{-06}$. The embedding layer is of size 768 for all experiments.

5 Analysis and Findings

We experiment with four post-processing methods as mentioned in Section 3.2. We analyze the effect of each post-processing using lexical and sequence classification tasks. Due to limited space, we present the average results of three models only. The task-wise results of all models including GPT2 are provided in Appendix D and C. We did not observe any difference in task-specific trends compared to the average trend present in the paper.

5.1 Lexical-level Tasks

Figure 2 presents the average layer-wise results using the lexical tasks. `raw` represents the embedding before applying any post-processing.

Post-processing is generally helpful. Comparing `raw` (blue line) with the rest, other than a few exceptions, layers of all models benefited from the post-processing steps. Surprisingly, `ulen` did not result in any change in the performance compared

[8] https://github.com/kudkudak/word-embeddings-benchmarks

[9] See Appendix for data statistics and download link.

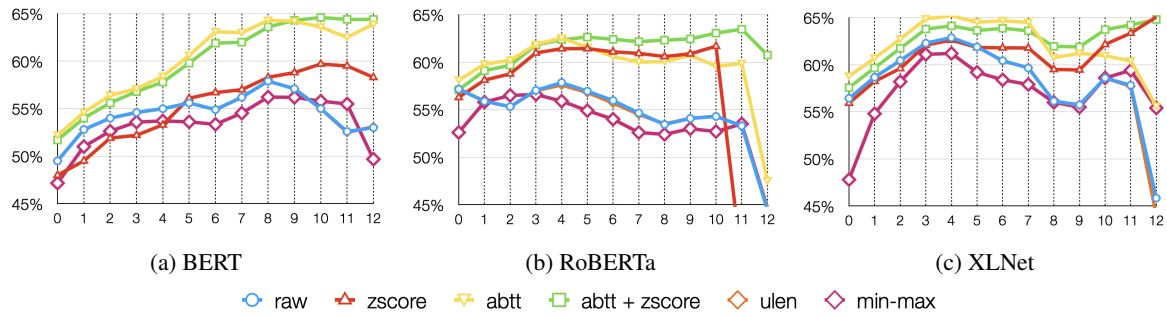

(a) BERT (b) RoBERTa (c) XLNet

Figure 2: Average lexical tasks results. X-axis: layers where 0 is the embedding layer

to raw.[10] minmax resulted in poor performance than raw. The two promising post-processing methods are zscore and abtt. In the following, we mainly discuss the results of zscore and abtt.

Higher layers achieve major improvements in performance. The general performance trend from lower layers to higher layers suggest that it is essential to post-process the representations of higher layers in order to unwrap the information present in those layers for the lexical tasks. In other words, the higher layer representations though optimized for the objective function still possess similar or better lexical-level information compared to the lower layers.

Comparing post-processing combinations. With the exception of the lower layers of BERT and the last two layers of RoBERTa, zscore (red line) achieved competitive or better performance than using the raw (blue line) representations. Comparing the variance of each layer (Figure 1), zscore is very effective for high variance layers such as the last layer of XLNet and most of the layers of GPT2 (see Appendix B). While RoBERTa has the most smooth variance curve, zscore is still quite effective in improving the performance from layer 1 to 10. The sudden drop in the performance of the last two layers is surprising. The reason could be an extremely low variance of these layers as can be seen in Figure 1 and applying zscore alone amplifies the amount of noise.

abtt outperformed or is competitive to the best performing individual post-processing methods (yellow line in Figure 2). The consistent improvement across all models for the lower layers reflects that the lower layer representations consist of top principle components that negatively influence the representations in the context of lexical tasks. For example, the representations from the lower layers of BERT might have a strong influence of position embeddings which may be harmful for word similarity and analogy tasks. On the top two layers, abtt does not seem to be very effective on RoBERTa and XLNet suggesting fewer dominating principle components in the higher layers.

Since zscore and abtt targets different properties in the representation, we apply them in succession. Using both methods in any order resulted in better representation quality especially for the higher layers (see green line abtt+zscore that outperformed or has competitive performance to the best result on all layers). These results show the potential of combining various post-processing methods, like abtt+zscore, in achieving better performance on the lexical tasks.

Comparing models. While post-processing methods benefited all models, XLNet showed the most increase in the performance with lower-middle layers (3,4) and higher layers outperforming all layers across all models. BERT also showed similar gains with more consistent trend i.e. an increase in performance with every higher layer. We did observe gains for RoBERTa. However, they are less substantial compared to BERT and the results on last layer dropped compared to other layers.

5.2 Sequence classification Tasks

Figure 3 presents the average layer-wise results using six GLUE tasks.[11] The performance improvements when post-processing the higher layers are found to be statistically significant at p=0.05. Additionally, the embedding layer of XLNet and middle to higher layers of RoBERTa achieved statistically significant improvements. Due to the poor performance of ulen and minmax, we did not report and discuss their results.

[10] The results of ulen and raw were identical up to two decimal points. The line of ulen is not visible because it is hidden behind raw in the figure.

[11] We observed similar trends for GPT2 (see Appendix B).

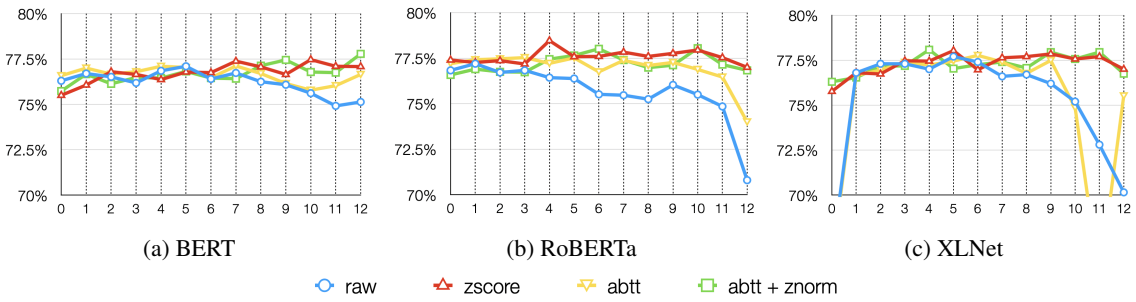

(a) BERT (b) RoBERTa (c) XLNet

Figure 3: Average GLUE tasks results

Post-processing is generally helpful. Similar to the performance on the lexical tasks, we observed that `zscore` and `abtt` post-processing methods resulted in competitive or better performance than `raw`. Particularly, `zscore` substantially improved the performance of the middle and higher layers (see red line and blue line representing `zscore` and `raw` respectively). `abtt` has comparable or better results than `raw`, although it never outperformed `zscore`. An interesting observation is the embedding layer where `abtt` resulted in similar performance to `raw` across all models. The embedding layer may encode information related to word identity and position, as in the case of BERT and RoBERTa which is neither useful nor harmful for the downstream tasks. `abtt` removed these high principle components while maintaining the performance of the embedding layer. Combining both post-processings did not result in any consistent benefit over using `zscore` alone.

All layers possess information about the tasks. In contrast to the common notion (Kovaleva et al., 2019) that the last layer is optimized for the objective function, and hence it is sub-optimal to use for down-stream tasks, we found that after `zscore`, the results of the last layers substantially improved, showing competitive results to the best performing layer for each model.

Task-wise performance Overall, majority of the tasks showed significant improvement with the post-processing of higher-layers. Additionally the embedding layer of XLNet benefited substantially with `zscore`. For example, the QNLI performance improved from 69.6 to 80.7 for the embedding layer, and 66.6 to 82.2 for the last layer. The only exception is the SST task that showed minimum benefit of the post-processing methods across all models. The performance differences between `raw` and post-processing methods are within 1% range, and are found to be insignificant.

6 Application and Discussion

The effectiveness of the post-processing of representations, particularly `zscore`, raises several interesting points in relation to the studies that use contextualized representations like feature-based transfer learning (Dalvi et al., 2020) and analysis/interpretation of deep models. For example, the work on probing layer representations typically builds a linear classifier and uses the classifier's performance as a proxy to judge how much linguistic information is learned in the representation. Our results show that even when using a strong classification model based on BiLSTM, it is essential to normalize the representations before training. A linear classifier is likely to be more vulnerable to the variance in the features and may not capture the true potential present in the representation. Similarly, feature-based transfer learning is directly affected by this post-processing and is likely to improve performance as shown by our sequence classification results.

In order to probe whether the post-processing of representations would impact representation analysis, we conducted a preliminary experiment on analyzing individual neurons in pre-trained models. Durrani et al. (2020) used the Linguistic Correlation Analysis (`LCA`) method to identify a set of neurons with respect to a linguistic property. The method trains a linear classifier on the linguistic property of interest while using neurons of the pre-trained model as features (12 layers x 768 dimensions = 9984 features). The output of the method results in a list of salient neurons with respect to the property in hand. We consider part-of-speech tagging (POS) as our linguistic property of interest and reproduce their results using the `raw` and the `zscore` post-processed contextualized embeddings of BERT and XLNet. Since `LCA` considers contextualized embeddings, we did not

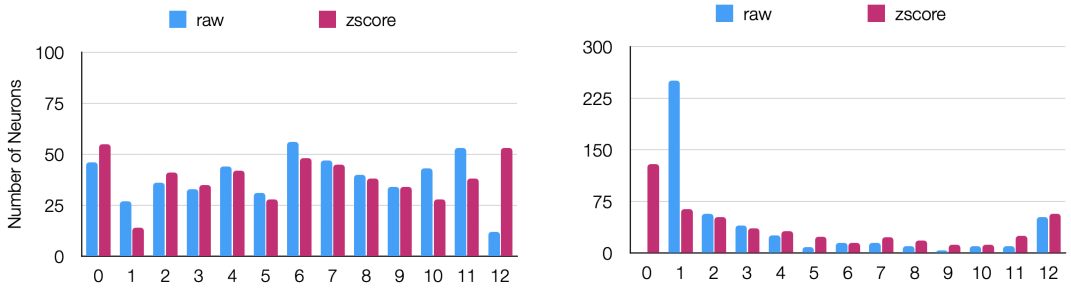

(a) BERT (b) XLNet

Figure 4: Layer-wise contribution of neurons with respect to POS tagging. x-axis corresponds to layers.

aggregate the contextualized embeddings of a word into a single word embedding as describe in Section 3.1. Thus, we use the contextualized embedding extracted from pre-trained models directly in the training of the linear classifier.

Figure 4 presents the layer-wise distribution of salient neurons identified by the algorithm using `raw` and `zscore` contextualized embeddings. On BERT, the most surprising result is the contribution of last layer which was minimum in the case of `raw` but after `zscore`, it is among the top contributing layers from which the most salient POS neurons are selected. The results of XLNet are also interesting. The contribution of embedding layer's neuron (0 index on x-axis) is zero in the case of `raw` contextualized embeddings while the first layer dominates the distribution. The `zscore` contextualized embedding picked the most number of salient neurons from the embedding layer and selected relatively less neurons from the first layer. This result shows that any analysis obtained by applying external probes on the features generated from pre-trained models needs to consider the effect of normalization into account as it provides an alternative view. In the future, we plan to extend this investigation further by analyzing the effect of post-processing on various other similarity and interpretation analysis works.

While the benefits of post-processing is evident in our experiments, the choice of when to use post-processing is application dependent. In analyzing representations of a network using a classification model, it is recommended to have standardized features to learn the best model. The choice of learned model architecture also plays a role here. The methods invariant to affine transformations would be least effected by the variability of features (Kaufman and Rousseeuw, 1990). For other applications e.g., identifying the importance of a neuron in a pre-trained model, the variance in the values of a neuron can be a signal of its importance and post-processing like `zscore` would result in the loss of such information.

7 Conclusion

We analyzed the effect of four post-processing methods on the contextualized representations using both the lexical and sequence classification tasks. We showed that for lexical tasks, post-processing methods `zscore` and `abtt` are essential to achieve better performance. On the sequence classification tasks, `zscore` alone outperformed all post-processing methods and `raw`. The most astonishing results are the large improvements in the performance of the last layers which reflect that they also possess equal amount of information about the lexical and classification tasks but the information is not readily available when used in feature-based transfer learning.

Our work opens several interesting frontiers with respect to the work that uses contextualized representations. In a preliminary experiment on representation analysis, we showed that post-processing the representations resulted in different findings. We suggested `zscore` as an essential step to consider when using contextualized embeddings for the feature-based transfer learning.

Ethics and Broader Impact

We used publicly available datasets, following their terms in the licenses. This includes seven datasets for similarity tasks, three datasets for analogy tasks, and six GLUE datasets. We do not see any harm or ethical issues resulting from our study and findings. Our study benefits the feature-based transfer learning at large and has direct implications towards the work on interpreting and analyzing deep models.

References

Yossi Adi, Einat Kermany, Yonatan Belinkov, Ofer Lavi, and Yoav Goldberg. 2017. Fine-grained analysis of sentence embeddings using auxiliary prediction tasks. In *Proceedings of the Internaltional Confernece for Learning Representations (ICLR)*.

Eneko Agirre, Enrique Alfonseca, Keith Hall, Jana Kravalova, Marius Paşca, and Aitor Soroa. 2009. A study on similarity and relatedness using distributional and WordNet-based approaches. In *Proceedings of Human Language Technologies: The 2009 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, pages 19–27, Boulder, Colorado. Association for Computational Linguistics.

Sanjeev Arora, Yingyu Liang, and Tengyu Ma. 2017. A simple but tough-to-beat baseline for sentence embeddings. In *Proceedings of the International Conference on Learning Representations*.

David Arps, Younes Samih, Laura Kallmeyer, and Hassan Sajjad. 2022. Probing for constituency structure in neural language models.

D. Anthony Bau, Yonatan Belinkov, Hassan Sajjad, Nadir Durrani, Fahim Dalvi, and James Glass. 2019. Identifying and controlling important neurons in neural machine translation. In *International Conference on Learning Representations (ICLR)*.

Yonatan Belinkov, Nadir Durrani, Fahim Dalvi, Hassan Sajjad, and James Glass. 2017a. What do Neural Machine Translation Models Learn about Morphology? In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (ACL)*, Vancouver. Association for Computational Linguistics.

Yonatan Belinkov, Nadir Durrani, Fahim Dalvi, Hassan Sajjad, and James Glass. 2020. On the linguistic representational power of neural machine translation models. *Computational Linguistics*, 46(1):1–52.

Yonatan Belinkov and James Glass. 2019. Analysis methods in neural language processing: A survey. *Transactions of the Association for Computational Linguistics*, 7:49–72.

Yonatan Belinkov, Lluís Màrquez, Hassan Sajjad, Nadir Durrani, Fahim Dalvi, and James Glass. 2017b. Evaluating layers of representation in neural machine translation on part-of-speech and semantic tagging tasks. In *Proceedings of the Eighth International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 1–10, Taipei, Taiwan. Asian Federation of Natural Language Processing.

Luisa Bentivogli, Ido Dagan, Hoa Trang Dang, Danilo Giampiccolo, and Bernardo Magnini. 2009. The fifth pascal recognizing textual entailment challenge. In *In Proc Text Analysis Conference (TAC'09*.

Rishi Bommasani, Kelly Davis, and Claire Cardie. 2020. Interpreting Pretrained Contextualized Representations via Reductions to Static Embeddings. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 4758–4781, Online. Association for Computational Linguistics.

Elia Bruni, Gemma Boleda, Marco Baroni, and Nam-Khanh Tran. 2012. Distributional semantics in technicolor. In *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 136–145, Jeju Island, Korea. Association for Computational Linguistics.

Daniel Cer, Mona Diab, Eneko Agirre, Iñigo Lopez-Gazpio, and Lucia Specia. 2017. SemEval-2017 task 1: Semantic textual similarity multilingual and crosslingual focused evaluation. In *Proceedings of the 11th International Workshop on Semantic Evaluation (SemEval-2017)*, pages 1–14, Vancouver, Canada. Association for Computational Linguistics.

Alexis Conneau, German Kruszewski, Guillaume Lample, Loïc Barrault, and Marco Baroni. 2018a. What you can cram into a single $&!#* vector: Probing sentence embeddings for linguistic properties. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 2126–2136, Melbourne, Australia. Association for Computational Linguistics.

Alexis Conneau, German Kruszewski, Guillaume Lample, Loïc Barrault, and Marco Baroni. 2018b. What you can cram into a single vector: Probing sentence embeddings for linguistic properties. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (ACL)*.

Fahim Dalvi, Nadir Durrani, Hassan Sajjad, Yonatan Belinkov, D. Anthony Bau, and James Glass. 2019. What is one grain of sand in the desert? analyzing individual neurons in deep nlp models. In *Proceedings of the Thirty-Third AAAI Conference on Artificial Intelligence (AAAI, Oral presentation)*.

Fahim Dalvi, Nadir Durrani, Hassan Sajjad, Yonatan Belinkov, and Stephan Vogel. 2017. Understanding and improving morphological learning in the neural machine translation decoder. In *Proceedings of the Eighth International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 142–151, Taipei, Taiwan. Asian Federation of Natural Language Processing.

Fahim Dalvi, Abdul Rafae Khan, Firoj Alam, Nadir Durrani, Jia Xu, and Hassan Sajjad. 2022. Discovering latent concepts learned in BERT. In *International Conference on Learning Representations*.

Fahim Dalvi, Hassan Sajjad, Nadir Durrani, and Yonatan Belinkov. 2020. Analyzing redundancy in pretrained transformer models. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 4908–4926, Online. Association for Computational Linguistics.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT 2019, Minneapolis, MN, USA, June 2-7, 2019, Volume 1 (Long and Short Papers)*, pages 4171–4186. Association for Computational Linguistics.

William B. Dolan and Chris Brockett. 2005. Automatically constructing a corpus of sentential paraphrases. In *Proceedings of the Third International Workshop on Paraphrasing (IWP2005)*.

Nadir Durrani, Fahim Dalvi, Hassan Sajjad, Yonatan Belinkov, and Preslav Nakov. 2019. One size does not fit all: Comparing NMT representations of different granularities. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 1504–1516, Minneapolis, Minnesota. Association for Computational Linguistics.

Nadir Durrani, Hassan Sajjad, Fahim Dalvi, and Yonatan Belinkov. 2020. Analyzing individual neurons in pretrained language models. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP-2020)*, Online. Association for Computational Linguistics.

Kawin Ethayarajh. 2019. How contextual are contextualized word representations? comparing the geometry of BERT, ELMo, and GPT-2 embeddings. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 55–65, Hong Kong, China. Association for Computational Linguistics.

Allyson Ettinger, Ahmed Elgohary, and Philip Resnik. 2016. Probing for semantic evidence of composition by means of simple classification tasks. In *Proceedings of the 1st Workshop on Evaluating Vector-Space Representations for NLP*, pages 134–139, Berlin, Germany. Association for Computational Linguistics.

Felix Hill, Roi Reichart, and Anna Korhonen. 2015. SimLex-999: Evaluating semantic models with (genuine) similarity estimation. *Computational Linguistics*, 41(4):665–695.

Dieuwke Hupkes, Sara Veldhoen, and Willem Zuidema. 2018. Visualisation and 'diagnostic classifiers' reveal how recurrent and recursive neural networks process hierarchical structure. *Journal of Artificial Intelligence Research*, 61:907–926.

Micheline Kamber Jiawei Han, Jian Pei. 2011. *Data Mining: Concepts and Techniques*. Elsevier Science Ltd.

David A. Jurgens, Peter D. Turney, Saif M. Mohammad, and Keith J. Holyoak. 2012. Semeval-2012 task 2: Measuring degrees of relational similarity. In *Proceedings of the First Joint Conference on Lexical and Computational Semantics - Volume 1: Proceedings of the Main Conference and the Shared Task, and Volume 2: Proceedings of the Sixth International Workshop on Semantic Evaluation*, SemEval '12, page 356–364, USA. Association for Computational Linguistics.

Leonard Kaufman and Peter J. Rousseeuw. 1990. *Finding Groups in Data: An Introduction to Cluster Analysis.* John Wiley.

Philipp Koehn, Hieu Hoang, Alexandra Birch, Chris Callison-Burch, Marcello Federico, Nicola Bertoldi, Brooke Cowan, Wade Shen, Christine Moran, Richard Zens, et al. 2007. Moses: Open source toolkit for statistical machine translation. In *Proceedings of the 45th annual meeting of the ACL on interactive poster and demonstration sessions*, pages 177–180. Association for Computational Linguistics.

Olga Kovaleva, Alexey Romanov, Anna Rogers, and Anna Rumshisky. 2019. Revealing the dark secrets of BERT. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 4365–4374, Hong Kong, China. Association for Computational Linguistics.

Omer Levy, Yoav Goldberg, and Ido Dagan. 2015. Improving distributional similarity with lessons learned from word embeddings. *Transactions of the Association for Computational Linguistics*, 3:211–225.

Nelson F. Liu, Matt Gardner, Yonatan Belinkov, Matthew E. Peters, and Noah A. Smith. 2019a. Linguistic knowledge and transferability of contextual representations. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 1073–1094, Minneapolis, Minnesota. Association for Computational Linguistics.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019b. RoBERTa: A robustly optimized BERT pretraining approach. *CoRR*, abs/1907.11692.

Thang Luong, Richard Socher, and Christopher Manning. 2013. Better word representations with recursive neural networks for morphology. In *Proceedings of the Seventeenth Conference on Computational Natural Language Learning*, pages 104–113, Sofia, Bulgaria. Association for Computational Linguistics.

Tomas Mikolov, Kai Chen, Greg S. Corrado, and Jeffrey Dean. 2013a. Efficient estimation of word representations in vector space.

Tomas Mikolov, Scott Wen-tau Yih, and Geoffrey Zweig. 2013b. Linguistic regularities in continuous space word representations. In *Proceedings of the 2013 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL-HLT-2013)*. Association for Computational Linguistics.

Jiaqi Mu and Pramod Viswanath. 2018. All-but-the-top: Simple and effective postprocessing for word representations. In *International Conference on Learning Representations*.

Matthew Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer. 2018. Deep contextualized word representations. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 2227–2237, New Orleans, Louisiana. Association for Computational Linguistics.

Matthew E. Peters, Sebastian Ruder, and Noah A. Smith. 2019. To tune or not to tune? adapting pretrained representations to diverse tasks. In *Proceedings of the 4th Workshop on Representation Learning for NLP (RepL4NLP-2019)*, pages 7–14, Florence, Italy. Association for Computational Linguistics.

Jason Phang, Phil Yeres, Jesse Swanson, Haokun Liu, Ian F. Tenney, Phu Mon Htut, Clara Vania, Alex Wang, and Samuel R. Bowman. 2020. `jiant` 2.0: A software toolkit for research on general-purpose text understanding models. `http://jiant.info/`.

Alec Radford, Karthik Narasimhan, Tim Salimans, and Ilya Sutskever. 2018. Improving language understanding by generative pre-training.

Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. 2019. Language models are unsupervised multitask learners. *OpenAI blog*, 1(8):9.

Kira Radinsky, Eugene Agichtein, Evgeniy Gabrilovich, and Shaul Markovitch. 2011. A word at a time: Computing word relatedness using temporal semantic analysis. In *Proceedings of the 20th International Conference on World Wide Web*, WWW '11, page 337–346, New York, NY, USA. Association for Computing Machinery.

Pranav Rajpurkar, Jian Zhang, Konstantin Lopyrev, and Percy Liang. 2016. SQuAD: 100,000+ questions for machine comprehension of text. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 2383–2392, Austin, Texas. Association for Computational Linguistics.

Herbert Rubenstein and John B. Goodenough. 1965. Contextual correlates of synonymy. *Commun. ACM*, 8(10):627–633.

Magnus Sahlgren, Amaru Cuba Gyllensten, Fredrik Espinoza, Ola Hamfors, Jussi Karlgren, Fredrik Olsson, Per Persson, Akshay Viswanathan, and Anders Holst. 2016. The gavagai living lexicon. In *Proceedings of the Tenth International Conference on Language Resources and Evaluation (LREC'16)*, pages 344–350, Portorož, Slovenia. European Language Resources Association (ELRA).

Hassan Sajjad, Nadir Durrani, and Fahim Dalvi. 2022a. Neuron-level Interpretation of Deep NLP Models: A Survey. *Transactions of the Association for Computational Linguistics*.

Hassan Sajjad, Nadir Durrani, Fahim Dalvi, Firoj Alam, Abdul Rafae Khan, and Jia Xu. 2022b. Analyzing encoded concepts in transformer language models. In *Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics*, NAACL '22, Seattle, Washington, USA. Association for Computational Linguistics.

Richard Socher, Alex Perelygin, Jean Wu, Jason Chuang, Christopher D. Manning, Andrew Ng, and Christopher Potts. 2013. Recursive deep models for semantic compositionality over a sentiment treebank. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*, pages 1631–1642, Seattle, Washington, USA. Association for Computational Linguistics.

Ian Tenney, Dipanjan Das, and Ellie Pavlick. 2019a. BERT rediscovers the classical NLP pipeline. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 4593–4601, Florence, Italy. Association for Computational Linguistics.

Ian Tenney, Patrick Xia, Berlin Chen, Alex Wang, Adam Poliak, R Thomas McCoy, Najoung Kim, Benjamin Van Durme, Samuel R. Bowman, Dipanjan Das, and Ellie Pavlick. 2019b. What do you learn from context? probing for sentence structure in contextualized word representations.

Elena Voita, David Talbot, Fedor Moiseev, Rico Sennrich, and Ivan Titov. 2019. Analyzing multi-head self-attention: Specialized heads do the heavy lifting, the rest can be pruned. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 5797–5808, Florence, Italy. Association for Computational Linguistics.

Alex Wang, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel Bowman. 2018. GLUE: A multi-task benchmark and analysis platform for natural language understanding. In *Proceedings of the 2018 EMNLP Workshop BlackboxNLP: Analyzing and Interpreting Neural Networks for NLP*, pages 353–355, Brussels, Belgium. Association for Computational Linguistics.

B. Wang, F. Chen, A. Wang, and C. . J. Kuo. 2019. Post-processing of word representations via variance normalization and dynamic embedding. In *2019 IEEE*

International Conference on Multimedia and Expo (ICME), pages 718–723.

Adina Williams, Nikita Nangia, and Samuel Bowman. 2018. A broad-coverage challenge corpus for sentence understanding through inference. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 1112–1122, New Orleans, Louisiana. Association for Computational Linguistics.

Benjamin J. Wilson and Adriaan M. J. Schakel. 2015. Controlled experiments for word embeddings. *CoRR*, abs/1510.02675.

John Wu, Yonatan Belinkov, Hassan Sajjad, Nadir Durrani, Fahim Dalvi, and James Glass. 2020. Similarity Analysis of Contextual Word Representation Models. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics (ACL)*, Seattle. Association for Computational Linguistics.

Zhilin Yang, Zihang Dai, Yiming Yang, Jaime Carbonell, Russ R Salakhutdinov, and Quoc V Le. 2019. Xlnet: Generalized autoregressive pretraining for language understanding. In *Advances in neural information processing systems*, pages 5753–5763.

Torsten Zesch, Christof Müller, and Iryna Gurevych. 2008. Using wiktionary for computing semantic relatedness. In *Proceedings of the 23rd National Conference on Artificial Intelligence - Volume 2*, AAAI'08, page 861–866. AAAI Press.

Appendix

A Data Statistics

In table 1, we present statistics of the different datasets that we used for the experiments.

Task	Train	Dev	Task	Train	Dev
SST-2	67,349	872	QQP	363,846	40,430
MRPC	3,668	408	RTE	2,490	277
MNLI	392,702	9,815	STS-B	5,749	1,500
QNLI	104,743	5,463			

Table 1: Data statistics (number of sequences) on the official training and development sets used in the experiments. All are binary classification tasks, except for STS-B, which is a regression task. Recall that the test sets are not publicly available, and hence we use 10% of the official train as development, and the official development set as our test set. Exact split information is provided in the code README. The data is available at https://gluebenchmark.com/tasks.

B GPT-2 Average Results on GLUE

In Figure 5a and 5b, we report the average results for different GLUE and lexical tasks across different L/Ns, respectively.

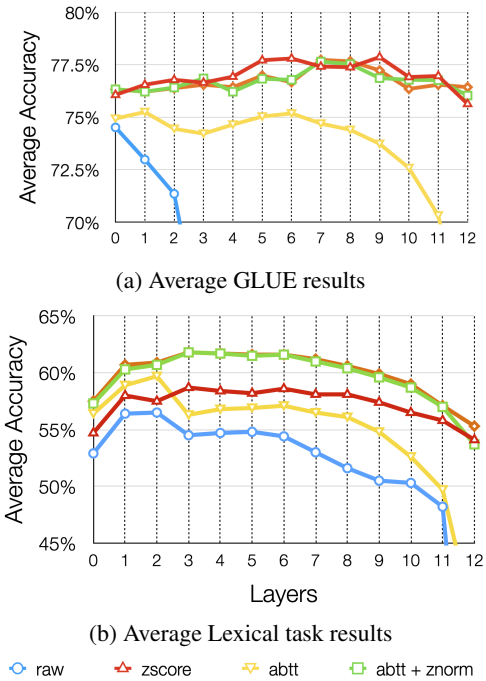

(a) Average GLUE results

(b) Average Lexical task results

Figure 5: GPT2

C Sequence Classification Results

In the following tables, we provide results obtained from raw embedding, z-socre normalization and all-but-the-top, for **BERT:** 2, 3, 4; **XLNet:** 5, 6, 7; **RoBERTa:** 9, 10 and **GPT2:** 11, 12, 13.

L/N	QNLI	MNLI	MRPC	SST	STS	RTE	RTE
0	0.82	0.724	0.775	0.88	0.801	0.578	0.578
1	0.815	0.724	0.797	0.885	0.8	0.581	0.581
2	0.816	0.712	0.784	0.89	0.808	0.581	0.581
3	0.805	0.716	0.779	0.894	0.807	0.57	0.57
4	0.818	0.734	0.775	0.89	0.806	0.588	0.588
5	0.824	0.74	0.779	0.901	0.804	0.578	0.578
6	0.813	0.739	0.777	0.896	0.799	0.56	0.56
7	0.821	0.735	0.777	0.888	0.802	0.581	0.581
8	0.813	0.733	0.784	0.882	0.8	0.563	0.563
9	0.8	0.726	0.784	0.896	0.799	0.56	0.56
10	0.794	0.725	0.787	0.889	0.8	0.542	0.542
11	0.773	0.716	0.772	0.898	0.794	0.542	0.542
12	0.805	0.728	0.77	0.878	0.785	0.542	0.542

Table 2: **BERT**: raw embedding

L/N	QNLI	MNLI	MRPC	SST	STS corr	RTE	uvar
0	0.801	0.705	0.77	0.881	0.792	0.581	0.755
1	0.802	0.727	0.765	0.885	0.797	0.588	0.761
2	0.818	0.728	0.784	0.886	0.804	0.588	0.768
3	0.809	0.71	0.775	0.885	0.81	0.61	0.767
4	0.807	0.727	0.762	0.886	0.804	0.596	0.764
5	0.813	0.729	0.772	0.896	0.812	0.585	0.768
6	0.816	0.726	0.775	0.901	0.813	0.574	0.768
7	0.822	0.726	0.799	0.892	0.816	0.588	0.774
8	0.813	0.742	0.787	0.892	0.816	0.574	0.771
9	0.811	0.734	0.77	0.897	0.82	0.567	0.767
10	0.812	0.734	0.797	0.894	0.819	0.592	0.775
11	0.808	0.733	0.789	0.901	0.817	0.578	0.771
12	0.808	0.733	0.799	0.883	0.821	0.581	0.771

Table 3: **BERT**: z-score normalization

L/N	QNLI	MNLI	MRPC	SST	STS corr	RTE	abtt
0	0.82	0.72	0.772	0.872	0.801	0.61	0.766
1	0.823	0.729	0.779	0.882	0.811	0.596	0.770
2	0.808	0.727	0.777	0.881	0.811	0.596	0.767
3	0.817	0.732	0.784	0.886	0.815	0.574	0.768
4	0.819	0.724	0.789	0.885	0.81	0.599	0.771
5	0.824	0.73	0.775	0.89	0.807	0.596	0.770
6	0.822	0.73	0.775	0.888	0.81	0.563	0.765
7	0.829	0.729	0.784	0.886	0.814	0.585	0.771
8	0.817	0.734	0.775	0.896	0.82	0.56	0.767
9	0.813	0.737	0.775	0.891	0.811	0.542	0.762
10	0.815	0.732	0.772	0.896	0.801	0.531	0.758
11	0.814	0.726	0.779	0.894	0.807	0.542	0.760
12	0.824	0.746	0.782	0.885	0.813	0.549	0.767

Table 4: **BERT**: all-but-the-top

L/N	QNLI	MNLI	MRPC	SST	STS corr	RTE
0	0.696	0.705	0.691	0.853	0.52	0.531
1	0.81	0.717	0.789	0.892	0.812	0.588
2	0.81	0.735	0.797	0.891	0.814	0.592
3	0.803	0.736	0.777	0.898	0.82	0.603
4	0.791	0.735	0.777	0.893	0.824	0.599
5	0.803	0.729	0.801	0.888	0.818	0.625
6	0.813	0.734	0.787	0.901	0.805	0.606
7	0.812	0.733	0.784	0.901	0.807	0.556
8	0.782	0.726	0.814	0.896	0.79	0.596
9	0.794	0.723	0.797	0.893	0.803	0.563
10	0.76	0.724	0.772	0.897	0.791	0.57
11	0.722	0.669	0.777	0.884	0.75	0.563
12	0.666	0.641	0.733	0.892	0.732	0.545

Table 5: **XLNet**: raw embedding

L/N	QNLI	MNLI	MRPC	SST	STS corr	RTE
0	0.807	0.722	0.775	0.885	0.805	0.552
1	0.806	0.727	0.792	0.896	0.809	0.578
2	0.807	0.733	0.777	0.897	0.816	0.574
3	0.815	0.742	0.772	0.898	0.822	0.599
4	0.816	0.736	0.77	0.894	0.828	0.603
5	0.821	0.733	0.792	0.896	0.826	0.614
6	0.813	0.715	0.782	0.896	0.827	0.585
7	0.815	0.738	0.787	0.894	0.825	0.599
8	0.815	0.735	0.797	0.9	0.823	0.592
9	0.808	0.735	0.799	0.9	0.823	0.606
10	0.809	0.733	0.806	0.89	0.82	0.596
11	0.821	0.746	0.799	0.89	0.828	0.578
12	0.822	0.74	0.792	0.886	0.824	0.556

Table 6: **XLNet**: z-score normalization

L/N	QNLI	MNLI	MRPC	SST	STS corr	RTE
0	0.661	0.708	0.703	0.853	0.509	0.542
1	0.803	0.724	0.77	0.891	0.815	0.596
2	0.823	0.735	0.76	0.899	0.819	0.581
3	0.823	0.731	0.794	0.89	0.824	0.581
4	0.813	0.744	0.794	0.888	0.825	0.57
5	0.824	0.736	0.792	0.897	0.824	0.574
6	0.824	0.733	0.789	0.892	0.829	0.599
7	0.827	0.732	0.801	0.891	0.826	0.567
8	0.817	0.729	0.799	0.893	0.812	0.556
9	0.824	0.726	0.809	0.892	0.813	0.585
10	0.789	0.693	0.779	0.891	0.812	0.527
11	0.657	0.6	0.721	0.876	0.536	0.527
12	0.805	0.696	0.77	0.89	0.828	0.542

Table 7: **XLNet**: all-but-the-top

L/N	QNLI	MNLI	MRPC	SST	STS corr	RTE
0	0.817	0.742	0.784	0.875	0.818	0.574
1	0.814	0.724	0.806	0.891	0.809	0.588
2	0.812	0.726	0.789	0.893	0.809	0.574
3	0.813	0.74	0.787	0.882	0.803	0.585
4	0.807	0.722	0.775	0.884	0.806	0.592
5	0.804	0.722	0.775	0.896	0.798	0.588
6	0.794	0.725	0.775	0.892	0.785	0.56
7	0.795	0.72	0.782	0.883	0.788	0.56
8	0.792	0.726	0.775	0.878	0.784	0.56
9	0.789	0.732	0.775	0.883	0.787	0.596
10	0.793	0.716	0.792	0.893	0.791	0.545
11	0.783	0.715	0.782	0.892	0.785	0.534
12	0.732	0.708	0.755	0.865	0.643	0.545

Table 8: **RoBERTa**: raw embedding

L/N	QNLI	MNLI	MRPC	SST	STS corr	RTE
0	0.824	0.735	0.789	0.89	0.803	0.603
1	0.815	0.729	0.792	0.903	0.816	0.581
2	0.819	0.729	0.787	0.901	0.814	0.592
3	0.824	0.73	0.772	0.901	0.823	0.581
4	0.825	0.728	0.804	0.905	0.829	0.617
5	0.817	0.734	0.792	0.9	0.824	0.588
6	0.815	0.73	0.799	0.894	0.826	0.592
7	0.818	0.732	0.804	0.9	0.824	0.592
8	0.815	0.73	0.804	0.884	0.824	0.599
9	0.821	0.742	0.792	0.901	0.825	0.585
10	0.81	0.739	0.794	0.897	0.823	0.614
11	0.817	0.73	0.797	0.894	0.826	0.588
12	0.814	0.737	0.792	0.89	0.827	0.56

Table 9: **RoBERTa**: z-score normalization

L/N	QNLI	MNLI	MRPC	SST	STS corr	RTE
0	0.814	0.734	0.797	0.876	0.826	0.585
1	0.821	0.744	0.809	0.891	0.813	0.567
2	0.819	0.727	0.799	0.908	0.81	0.585
3	0.826	0.738	0.797	0.889	0.817	0.585
4	0.823	0.737	0.787	0.888	0.817	0.581
5	0.824	0.736	0.806	0.89	0.821	0.574
6	0.813	0.733	0.792	0.896	0.808	0.563
7	0.818	0.739	0.797	0.885	0.812	0.592
8	0.817	0.738	0.794	0.884	0.815	0.578
9	0.816	0.741	0.789	0.884	0.814	0.592
10	0.817	0.74	0.787	0.896	0.799	0.574
11	0.813	0.733	0.789	0.881	0.805	0.567
12	0.794	0.716	0.767	0.876	0.745	0.542

Table 10: **RoBERTa**: all-but-the-top

L/N	QNLI	MNLI	MRPC	SST	STS corr	RTE
0	0.804	0.719	0.75	0.868	0.763	0.567
1	0.705	0.706	0.748	0.885	0.79	0.545
2	0.707	0.695	0.713	0.882	0.714	0.57
3	0.674	0.663	0.696	0.892	0.388	0.534
4	0.667	0.633	0.672	0.884	0.429	0.556
5	0.667	0.648	0.699	0.901	0.414	0.563
6	0.65	0.655	0.689	0.901	0.428	0.542
7	0.662	0.649	0.708	0.898	0.252	0.57
8	0.655	0.665	0.672	0.896	0.376	0.581
9	0.658	0.584	0.686	0.894	0.268	0.567
10	0.665	0.663	0.676	0.893	0.313	0.538
11	0.666	0.647	0.689	0.892	0.353	0.531
12	0.638	0.623	0.703	0.847	0.274	0.538

Table 11: **GPT-2**: raw embedding

L/N	QNLI	MNLI	MRPC	SST	STS corr	RTE
0	0.805	0.724	0.784	0.891	0.811	0.549
1	0.801	0.738	0.777	0.89	0.817	0.57
2	0.805	0.742	0.789	0.881	0.82	0.57
3	0.803	0.734	0.789	0.888	0.818	0.567
4	0.809	0.722	0.804	0.885	0.818	0.578
5	0.817	0.743	0.792	0.898	0.821	0.592
6	0.816	0.73	0.804	0.898	0.824	0.596
7	0.81	0.738	0.787	0.892	0.822	0.596
8	0.81	0.72	0.806	0.889	0.822	0.596
9	0.812	0.736	0.784	0.9	0.819	0.621
10	0.804	0.728	0.794	0.888	0.813	0.588
11	0.798	0.726	0.787	0.892	0.809	0.606
12	0.796	0.717	0.779	0.881	0.802	0.563

Table 12: **GPT-2**: z-score normalization

L/N	QNLI	MNLI	MRPC	SST	STS corr	RTE
0	0.803	0.732	0.765	0.865	0.786	0.545
1	0.752	0.71	0.787	0.89	0.802	0.574
2	0.722	0.685	0.784	0.897	0.812	0.567
3	0.757	0.708	0.76	0.881	0.795	0.552
4	0.754	0.712	0.765	0.888	0.804	0.556
5	0.763	0.713	0.767	0.892	0.8	0.567
6	0.77	0.717	0.762	0.89	0.812	0.56
7	0.754	0.716	0.762	0.888	0.806	0.556
8	0.756	0.714	0.752	0.891	0.791	0.56
9	0.739	0.697	0.765	0.893	0.774	0.556
10	0.726	0.683	0.748	0.884	0.769	0.545
11	0.722	0.67	0.708	0.893	0.688	0.538
12	0.677	0.599	0.713	0.85	0.47	0.531

Table 13: **GPT-2**: all-but-the-top

D Lexical task-wise Results

In the following tables we provide task-wise results across different results for different models: **BERT:** 14, 15, 16; **XLNet:** 17, 18, 19.

L/N	MEN	WS353	WS353R	WS353S	SimLex999	RW	RG65	MTurk	Google	MSR	SemEval2012_2	Average
L0	0.617	0.543	0.606	0.326	0.445	0.539	0.515	0.609	0.324	0.706	0.220	0.495
L1	0.671	0.530	0.635	0.369	0.462	0.600	0.575	0.671	0.341	0.729	0.222	0.528
L2	0.683	0.514	0.639	0.390	0.481	0.617	0.590	0.689	0.364	0.749	0.222	0.540
L3	0.678	0.508	0.672	0.406	0.488	0.624	0.593	0.710	0.369	0.733	0.224	0.546
L4	0.679	0.482	0.719	0.419	0.499	0.620	0.569	0.734	0.379	0.721	0.233	0.550
L5	0.677	0.463	0.768	0.441	0.502	0.612	0.535	0.744	0.395	0.748	0.233	0.556
L6	0.666	0.454	0.777	0.440	0.508	0.587	0.494	0.723	0.395	0.754	0.242	0.549
L7	0.687	0.472	0.789	0.449	0.524	0.608	0.514	0.741	0.397	0.765	0.239	0.562
L8	0.713	0.498	0.822	0.464	0.539	0.636	0.558	0.751	0.397	0.767	0.229	0.579
L9	0.710	0.478	0.844	0.457	0.537	0.613	0.543	0.708	0.399	0.774	0.220	0.571
L10	0.686	0.444	0.825	0.444	0.500	0.579	0.513	0.666	0.398	0.772	0.221	0.550
L11	0.654	0.406	0.824	0.414	0.479	0.528	0.477	0.605	0.402	0.781	0.215	0.526
L12	0.643	0.406	0.771	0.413	0.488	0.558	0.517	0.620	0.406	0.820	0.188	0.530
Average	**0.674**	**0.477**	**0.746**	**0.418**	**0.496**	**0.594**	**0.538**	**0.690**	**0.382**	**0.755**	**0.224**	**0.545**
Max	**0.713**	**0.543**	**0.844**	**0.464**	**0.539**	**0.636**	**0.593**	**0.751**	**0.406**	**0.820**	**0.242**	**0.579**

Table 14: **BERT**: raw embedding

L/N	**MEN**	**WS353**	**WS353R**	**WS353S**	**SimLex999**	**RW**	**RG65**	**MTurk**	**Google**	**MSR**	**SemEval2012_2**	**Average**
L0	0.588	0.513	0.606	0.263	0.432	0.511	0.505	0.562	0.326	0.706	0.215	0.480
L1	0.632	0.529	0.619	0.302	0.444	0.558	0.552	0.610	0.342	0.729	0.222	0.495
L2	0.655	0.540	0.620	0.347	0.459	0.586	0.564	0.644	0.365	0.751	0.223	0.519
L3	0.656	0.536	0.662	0.376	0.467	0.593	0.560	0.663	0.368	0.731	0.218	0.522
L4	0.670	0.541	0.675	0.410	0.480	0.597	0.546	0.692	0.379	0.721	0.225	0.533
L5	0.689	0.541	0.706	0.461	0.490	0.620	0.546	0.737	0.394	0.748	0.231	0.561
L6	0.697	0.555	0.717	0.490	0.511	0.629	0.542	0.747	0.397	0.751	0.235	0.567
L7	0.711	0.561	0.719	0.502	0.524	0.648	0.563	0.759	0.398	0.761	0.227	0.570
L8	0.742	0.562	0.748	0.514	0.541	0.692	0.622	0.791	0.397	0.768	0.227	0.583
L9	0.762	0.563	0.773	0.516	0.544	0.712	0.650	0.790	0.400	0.778	0.220	0.588
L10	0.769	0.570	0.775	0.511	0.534	0.725	0.667	0.797	0.400	0.778	0.222	0.597
L11	0.763	0.562	0.778	0.500	0.527	0.716	0.660	0.787	0.407	0.793	0.210	0.595
L12	0.744	0.524	0.740	0.486	0.530	0.719	0.667	0.784	0.411	0.831	0.193	0.583
Average	**0.698**	**0.546**	**0.703**	**0.437**	**0.499**	**0.639**	**0.588**	**0.720**	**0.383**	**0.757**	**0.221**	**0.553**
Max	**0.769**	**0.570**	**0.778**	**0.516**	**0.544**	**0.725**	**0.667**	**0.797**	**0.411**	**0.831**	**0.235**	**0.597**

Table 15: **BERT**: z-score normalization

L/N	**MEN**	**WS353**	**WS353R**	**WS353S**	**SimLex999**	**RW**	**RG65**	**MTurk**	**Google**	**MSR**	**SemEval2012_2**	**Average**
L0	0.608	0.542	0.633	0.303	0.441	0.557	0.526	0.623	0.300	0.714	0.235	0.522
L1	0.657	0.547	0.645	0.323	0.450	0.603	0.579	0.667	0.324	0.737	0.231	0.546
L2	0.684	0.577	0.672	0.369	0.473	0.624	0.594	0.685	0.351	0.750	0.234	0.564
L3	0.694	0.580	0.715	0.404	0.483	0.635	0.585	0.714	0.360	0.743	0.226	0.571
L4	0.715	0.585	0.740	0.438	0.502	0.640	0.572	0.748	0.379	0.736	0.240	0.584
L5	0.732	0.576	0.759	0.490	0.514	0.663	0.590	0.771	0.394	0.757	0.240	0.606
L6	0.741	0.545	0.785	0.514	0.537	0.678	0.597	0.788	0.398	0.764	0.248	0.631
L7	0.755	0.554	0.786	0.526	0.549	0.696	0.614	0.800	0.399	0.772	0.248	0.630
L8	0.777	0.577	0.804	0.537	0.565	0.733	0.664	0.828	0.397	0.775	0.239	0.643
L9	0.775	0.536	0.845	0.528	0.567	0.734	0.672	0.820	0.398	0.782	0.232	0.642
L10	0.757	0.509	0.852	0.509	0.542	0.710	0.644	0.795	0.398	0.780	0.240	0.636
L11	0.739	0.485	0.851	0.491	0.528	0.676	0.607	0.768	0.404	0.792	0.240	0.625
L12	0.773	0.519	0.815	0.516	0.560	0.735	0.687	0.797	0.404	0.816	0.225	0.639
Average	**0.723**	**0.549**	**0.762**	**0.458**	**0.516**	**0.668**	**0.610**	**0.754**	**0.377**	**0.763**	**0.237**	**0.603**
Max	**0.777**	**0.585**	**0.852**	**0.537**	**0.567**	**0.735**	**0.687**	**0.828**	**0.404**	**0.816**	**0.248**	**0.643**

Table 16: **BERT**: all-but-the-top

E Computing Infrastructure and Models' Parameter

We used a server with NVIDIA Tesla V100-SXM2-32 GB GPU, 56 cores, and 500GB CPU memory.

Models and Number of Parameters: Below, we list the values of the hyper-parameters for different models.

- **BERT** (bert-base-uncased): L=12, H=768, A=12, total parameters: 110M; where *L* is the number of layers (i.e., Transformer blocks), *H* is the hidden size, and *A* is the number of self-attention heads;
- **RoBERTa** (roberta-base): similar to BERT-base, but with a higher number of parameters (125M);
- **XLNet** (xlnet-base-cased) L=12, H=768,

L/N	MEN	WS353	WS353R	WS353S	SimLex999	RW	RG65	MTurk	Google	MSR	SemEval2012_2	Average
L0	0.663	0.578	0.684	0.441	0.502	0.686	0.617	0.760	0.332	0.712	0.233	0.564
L1	0.696	0.599	0.732	0.476	0.523	0.690	0.617	0.761	0.364	0.759	0.237	0.587
L2	0.723	0.595	0.758	0.507	0.540	0.706	0.638	0.777	0.387	0.785	0.233	0.605
L3	0.755	0.594	0.792	0.557	0.564	0.713	0.635	0.787	0.409	0.814	0.232	0.623
L4	0.754	0.583	0.810	0.584	0.574	0.709	0.629	0.791	0.420	0.825	0.237	0.629
L5	0.735	0.564	0.816	0.576	0.569	0.693	0.605	0.780	0.417	0.814	0.240	0.619
L6	0.733	0.501	0.841	0.547	0.573	0.655	0.573	0.760	0.411	0.810	0.243	0.604
L7	0.725	0.483	0.837	0.536	0.568	0.650	0.559	0.749	0.409	0.805	0.241	0.597
L8	0.671	0.444	0.802	0.504	0.543	0.593	0.485	0.708	0.400	0.790	0.234	0.561
L9	0.663	0.451	0.797	0.498	0.532	0.591	0.475	0.716	0.394	0.779	0.235	0.557
L10	0.706	0.466	0.840	0.502	0.541	0.642	0.540	0.748	0.412	0.808	0.239	0.586
L11	0.711	0.462	0.804	0.487	0.544	0.646	0.537	0.750	0.412	0.810	0.198	0.578
L12	0.514	0.314	0.732	0.263	0.370	0.450	0.351	0.593	0.409	0.838	0.208	0.458
Average	**0.696**	**0.510**	**0.788**	**0.498**	**0.534**	**0.648**	**0.559**	**0.745**	**0.398**	**0.796**	**0.232**	**0.582**
Max	**0.755**	**0.599**	**0.841**	**0.584**	**0.574**	**0.713**	**0.638**	**0.791**	**0.420**	**0.838**	**0.243**	**0.629**

Table 17: **XLNET**: raw embedding

L/N	MEN	WS353	WS353R	WS353S	SimLex999	RW	RG65	MTurk	Google	MSR	SemEval2012_2	Average
L0	0.665	0.555	0.702	0.407	0.495	0.678	0.628	0.750	0.334	0.711	0.227	0.559
L1	0.691	0.562	0.728	0.448	0.515	0.691	0.636	0.765	0.367	0.762	0.241	0.582
L2	0.714	0.565	0.734	0.489	0.531	0.699	0.641	0.774	0.389	0.788	0.232	0.596
L3	0.760	0.582	0.779	0.543	0.557	0.714	0.646	0.795	0.412	0.812	0.230	0.621
L4	0.760	0.582	0.795	0.568	0.571	0.707	0.634	0.792	0.420	0.824	0.234	0.626
L5	0.744	0.582	0.784	0.564	0.570	0.693	0.612	0.782	0.417	0.815	0.239	0.618
L6	0.741	0.584	0.790	0.560	0.568	0.691	0.614	0.785	0.412	0.807	0.244	0.618
L7	0.745	0.583	0.795	0.557	0.562	0.695	0.625	0.782	0.409	0.802	0.241	0.618
L8	0.719	0.571	0.758	0.538	0.538	0.659	0.569	0.765	0.405	0.792	0.231	0.595
L9	0.717	0.568	0.756	0.534	0.541	0.663	0.574	0.768	0.403	0.785	0.230	0.594
L10	0.758	0.583	0.816	0.546	0.554	0.700	0.623	0.792	0.417	0.812	0.238	0.622
L11	0.777	0.583	0.810	0.552	0.565	0.733	0.661	0.809	0.420	0.829	0.231	0.634
L12	0.798	0.562	0.864	0.553	0.563	0.767	0.702	0.829	0.430	0.859	0.231	0.651
Average	**0.738**	**0.574**	**0.778**	**0.528**	**0.548**	**0.699**	**0.628**	**0.784**	**0.403**	**0.800**	**0.235**	**0.610**
Max	**0.798**	**0.584**	**0.864**	**0.568**	**0.571**	**0.767**	**0.702**	**0.829**	**0.430**	**0.859**	**0.244**	**0.651**

Table 18: **XLNET**: z-score normalization

L/N	MEN	WS353	WS353R	WS353S	SimLex999	RW	RG65	MTurk	Google	MSR	SemEval2012_2	Average
L0	0.728	0.600	0.732	0.469	0.535	0.725	0.676	0.786	0.288	0.683	0.244	0.588
L1	0.731	0.613	0.797	0.494	0.535	0.722	0.657	0.783	0.339	0.759	0.247	0.607
L2	0.756	0.635	0.798	0.532	0.549	0.732	0.662	0.796	0.387	0.802	0.253	0.627
L3	0.799	0.646	0.840	0.587	0.575	0.737	0.666	0.799	0.410	0.825	0.252	0.649
L4	0.795	0.635	0.840	0.606	0.583	0.733	0.657	0.811	0.421	0.837	0.253	0.652
L5	0.781	0.614	0.835	0.602	0.581	0.726	0.645	0.805	0.420	0.831	0.253	0.645
L6	0.778	0.631	0.835	0.601	0.580	0.726	0.650	0.807	0.416	0.828	0.261	0.647
L7	0.788	0.630	0.825	0.603	0.574	0.728	0.661	0.796	0.413	0.822	0.256	0.645
L8	0.741	0.498	0.827	0.558	0.565	0.669	0.581	0.760	0.411	0.817	0.257	0.608
L9	0.745	0.524	0.805	0.567	0.561	0.680	0.591	0.778	0.412	0.814	0.258	0.612
L10	0.739	0.465	0.861	0.529	0.555	0.686	0.591	0.772	0.420	0.830	0.257	0.610
L11	0.740	0.440	0.861	0.522	0.560	0.681	0.566	0.777	0.421	0.841	0.236	0.604
L12	0.702	0.375	0.828	0.454	0.523	0.582	0.446	0.709	0.420	0.854	0.221	0.556
Average	**0.756**	**0.562**	**0.822**	**0.548**	**0.560**	**0.702**	**0.619**	**0.783**	**0.398**	**0.811**	**0.250**	**0.619**
Max	**0.799**	**0.646**	**0.861**	**0.606**	**0.583**	**0.737**	**0.676**	**0.811**	**0.421**	**0.854**	**0.261**	**0.652**

Table 19: **XLNET**: all-but-the-top

A=12, total parameters: 110M.

- **GPT2** L=12, H=768, A=12, total parameters: 117M.

Does BERT Rediscover a Classical NLP Pipeline?

Jingcheng Niu **Wenjie Lu** **Gerald Penn**
University of Toronto Vector Institute
{niu,luwenjie,gpenn}@cs.toronto.edu

Abstract

Does BERT store surface knowledge in its bottom layers, syntactic knowledge in its middle layers, and semantic knowledge in its upper layers? In re-examining Jawahar et al. (2019) and Tenney et al.'s (2019a) probes into the structure of BERT, we have found that the pipeline-like separation that they asserted lacks conclusive empirical support. BERT's structure is, however, linguistically founded, although perhaps in a way that is more nuanced than can be explained by layers alone. We introduce a novel probe, called *GridLoc*, through which we can also take into account token positions, training rounds, and random seeds. Using GridLoc, we are able to detect other, stronger regularities that suggest that pseudo-cognitive appeals to layer depth may not be the preferable mode of explanation for BERT's inner workings.

1 Introduction

"Surface information at the bottom, syntactic information in the middle, semantic information at the top" (Jawahar et al., 2019). This conclusion is also drawn by Tenney et al. (2019a). While this portrait of multiple layers of linguistic structure has indeed been projected into pipline architectures by some NLP systems before (e.g., Manning et al., 2014), the projection of this understanding onto the internal structure of BERT is now widely accepted both in the BERTology community (Tenney et al., 2019a,b; Hewitt and Liang, 2019; Zhu et al., 2022) and by researchers working on downstream application tasks with BERT (Xiao et al., 2021).

At the same time, there is nevertheless scepticism about the premises of probing itself. As Hewitt and Liang (2019) pondered: "when a probe achieves high accuracy on a linguistic task using a representation, can we conclude that the representation encodes linguistic structure, or has the probe just learned the task?" Furthermore, Pimentel et al. (2020) "cast doubt on whether probing makes sense as a scientific endeavour," because from an information-theoretic perspective, BERT cannot introduce new linguistic information by processing the input sequence.

In our view, this debate does not need to hinder the endeavour of uncovering structure within BERT. Regardless of stance, all parties agree with the existence and importance of different levels of information within BERT itself or its embeddings. There is, however, room for improvement in the investigative methods of both Jawahar et al. (2019) and Tenney et al. (2019a) (J&T), which seem to have been limited to observational confirmations of what they sought to find.

Our own exploratory analysis has revealed that BERT is indeed linguistically founded, although not in a way that suggests a classical pipeline architecture, other than what factors through our own functional understanding of NLP's terminology and subtasks. In addition to examining BERT layers, as J&T did, we have also examined BERT's structure through the lenses of the choice of random seed, training iterations, and, most importantly, token position. We also present several statistical tests of J&T's own conclusions.

We propose *GridLoc*,[1] a self-attention-based probing method that can probe across all of these aforementioned dimensions. Using this novel probing approach and our statistical testing suite, a much more comprehensive picture of the structure of BERT arises. Specifically:

- BERT's task-specific features appear in different token positions in an idiosyncratic but consistent pattern for each task;
- the attending task-specific features exhibit variance across different sentences, different training durations, and different random seeds;
- probe results for *tree depth,* in particular, show an anomalous distribution of linguistic evidence

[1]The implementation, data, plots and results of GridLoc are available online: https://github.com/frankniujc/gridloc_probe and https://doi.org/10.5683/SP3/PCZHN4.

Proceedings of the 29th International Conference on Computational Linguistics, pages 3143–3153
October 12–17, 2022.

when taking both layers and token position into consideration.

2 Attributing the CNLP to BERT

That there is or was a classical or traditional NLP pipeline is a rather naïve view to take of the history of natural language processing. While there were discussions of independent stages of token-level, syntactic and semantic processing already in the earliest work on machine translation, the pioneers who first engaged with these stages of analysis were of the considered opinion that careful restraint had to be exercised at every level of analysis so that as much of the inherent ambiguity of linguistic input could be carried forward as possible, in the interests of both efficiency and accuracy (Sparck Jones, 2000). It was not until the late 1970s and early 1980s that an excessive reliance on classical logic by the NLP technologists of the grammatico-logical movement, together with contemporaneous psycholinguistics and cognitive science research pointing to a modularity of linguistic structure in human sentence processing (Garrett, 1975, 1980; Fodor, 1983), led to a pipeline-based view that, almost immediately, was apologized for as a convenient abstraction, "incremental" (Levelt, 1989), "highly flexible" and "even opportunistic" (Marslen-Wilson and Tyler, 1987) (see Jackendoff (2000) for a more detailed discussion). While the smoother numerical allocations across tasks in BERTology work, and the distributional graphs drawn by Tenney et al. (2019a) in particular, may at first seem to be commensurate with or even suggest these more nuanced views of a language processing pipeline, the haze of smoke surrounding the visualization of probing evidence makes it extremely difficult to draw precise conclusions from these figures, as we will argue below. Nevertheless, probe methods in general (Adi et al., 2017; Hupkes and Zuidema, 2018; Conneau et al., 2018; Jawahar et al., 2019; Pimentel et al., 2020) are to be credited as one of the few means that we have of approaching the seemingly impossible task of interpreting the neural feature representations within BERT.

Generally, a probe can be *performance-based* or *attention-based*.

2.1 Performance-based Probing

A performance-based probe uses an auxiliary task to test for evidence of a particular type of knowledge, by training a supervised classifier with only

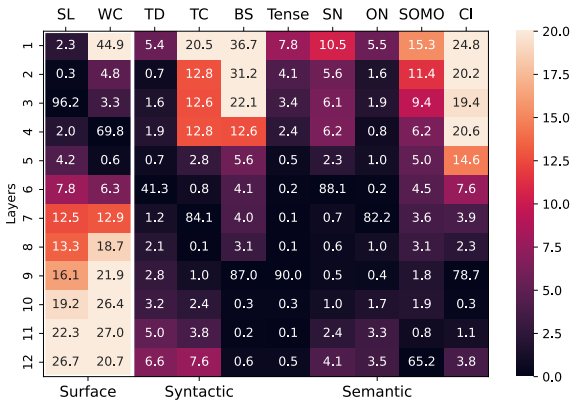

Figure 1: Layer performance probing result of Jawahar et al. (2019), as presented in their Table 2. For clearer visualisation, we transformed this table into a heat map. Each column corresponds to a task, with the best-performing layer in that column containing the performance as a percentage, and the remaining cells displaying their deviation from the best performer in raw percentage points. Surface tasks perform better near the top; syntactic tasks and semantic tasks perform better near the bottom, but their performance patterns are not distinguishable by layer.

BERT's embeddings as input. Good performance of the classifier is interpreted as evidence of relevant linguistic knowledge being present.

Jawahar et al.'s (2019) analysis is a typical application of performance-based probing. They used SentEval (Conneau et al., 2018; Conneau and Kiela, 2018) which contains 10 probing tasks at 3 linguistic levels:

- surface tasks: *sentence length* (SL) and *word content* (WC);
- syntactic tasks: *bigram shift* (BS), *tree depth* (TD) and *top constituent* (TC);
- and semantic tasks: *tense*, *subject number* (SN), *object number* (ON), *semantic odd man out* (SOMO) and *coordination inversion* (CI).[2]

Figure 1 shows the performance of Jawahar et al.'s (2019) probing tasks for each layer. Jawahar et al.'s (2019) visual examination of the results prompted the conclusion that began our introduction. But the observation that semantic information is at the top is puzzling. Except for the SOMO task, all 4 semantic tasks reach peak performance between layers 6 and 9, and all 3 syntactic tasks attain their peak performance within the same range. The similarities between syntactic and semantic

[2]One of our reviewers expressed concerns about why "semantics" and "syntax" were even appropriate labels. Subject/object number and tense, for example, are arguably syntactic/morphological. We agree.

tasks are more apparent with our visualisation of the relative performance differences in Figure 1: the large, dark area in the lower right quadrant is due to an indistinguishability of layers across syntactic and semantic task types.

Reappraisal of Jawahar et al. (2019) Some statistical tests may be more illuminating, such as Kendall's (1938) τ analysis of rank correlation between Jawahar et al.'s (2019) probe performance and a putatively discrete pipeline (surface: 1, syntactic: 2, and semantic: 3). There are two ways to convert Jawahar et al.'s (2019) results into ordinal layer numbers: using a task's top-performing layer number, in which a single number may be selected by multiple tasks (this seems to be what Jawahar et al. (2019) were informally doing), or forcing each task into a distinct layer so that the combined accuracy (the product of all accuracies) of the entire pipeline is maximized. To find this maximal layer assignment, we used SciPy's linear sum assignment to find the maximum sum of logarithms of performance. Both possible layer assignments are shown in Table 1.

The last two columns of Table 1 show the τ scores. Both layer assignment methods exhibit moderate correlations, but a *post hoc* exclusion of the surface tasks reveals only a weak correlation at the syntactic and semantic levels. These results corroborate our observation that, of the three, only the surface tasks are distinguishable.

2.2 Attention-based Probing

Attention has somewhat controversially been interpreted as an explanation of a model's reasoning (Clark et al., 2019). Typically, an attention mechanism assigns a scalar weight to each input source. Hence, the attention mechanism will enhance the important features' effect during training, and so the magnitude of attention weights is often interpreted as an importance score. Clark et al. (2019) and Vig (2019) directly studied the attention mechanism of BERT. Since BERT's attention mechanism does not cross layer boundaries, however, a new probe with an auxiliary attention mechanism is required.

Tenney et al. (2019a) used this attention-based probe to determine which layer contains more task information. Tenney et al. (2019a) exploit a scalar "attention" weight $s_\tau = \text{softmax}(\mathbf{a}_\tau)$ for each layer. The probe classifier is trained using a weighted sum of embeddings (defined as $\mathbf{h}_{i,\tau} = \gamma_\tau \sum_{\ell=0}^{L} s_\tau^{(\ell)} H_i^{(\ell)}$) as input, where $H_i^{(\ell)}$ is the ℓ-th layer of the i-th token's BERT embedding. The value of the attention weights is optimised during the training process, and therefore, the magnitude of the attention weight is understood as a measure of the amount of task-specific knowledge in the corresponding BERT layer.

These scalar weighted embeddings are then leveraged into an improved performance probe that uses *cumulative scoring*. Jawahar et al.'s (2019) probe classifier has access only to a single layer. Tenney et al. (2019a) argue that, since task-specific knowledge can spread out across multiple layers, probing a single layer cannot reveal the full picture. Therefore, they propose to train L probes, each having access to the scalar weighted embeddings from layer 1 to layer ℓ. The ℓ-th probe therefore has access to strictly one more layer's worth of information compared to the $(\ell-1)$-th probe. They can then measure the amount of information each layer introduces by calculating the difference between two adjacent probes: $\Delta_\tau^{(\ell)} = \text{Score}_\tau^{(\ell)} - \text{Score}_\tau^{(\ell-1)}$.

They deploy their two tests on the 8 span probing tasks of Tenney et al. (2019b), and aggregate each task's per-layer scalar mixing weights results into a *centre of gravity* score ($\mathbb{E}[\ell] = \sum_{\ell=0}^{L} \ell \cdot s_\tau^{(\ell)}$), and the cumulative scoring results into an *expected layer* score ($\mathbb{E}_\Delta[\ell] = \sum_{\ell=1}^{L} \ell \cdot \Delta_\tau^{(\ell)} / \sum_{\ell=1}^{L} \Delta_\tau^{(\ell)}$).

Reappraisal of Tenney et al. (2019a) Again, by visually observing these scores, they conclude that "the tasks [are] encoded in a natural progression: POS tags processed earliest, followed by constituents, dependencies, semantic roles, and coreference." Again, we prefer quantitative tests: while the Pearson's correlation between the centre of gravity and the pipeline ordering of the tasks is weak ($r = 0.319, p = 0.44$), the correlation between expected layer and the pipeline ordering is very strong ($r = 0.933, p = 0.0005$). On the other hand, the correlation between the Kullback-Leibler divergence of the difference scores from a uniform distribution and the pipeline ordering is also very strong ($r = -0.869, p = 0.005$).

2.3 Discussion

There are numerous scales along which tasks can be ordered: deep vs. shallow, semantic vs. surface, difficult vs. easy, and over- vs. underdispersed, to name a few. The thesis of Tenney et al. (2019a) is that the first two proceed in lockstep. The last two are highly correlated ($r = -0.879, p = 0.004$).

Task	SL	WC	TD	TC	BS	Tense	SN	ON	SOMO	CI	τ	τ syn sem
Top	3	4	6	7	9	9	6	7	12	9	0.596	0.269
Distinct	3	4	6	8	11	5	9	7	12	10	0.455	0.049

Table 1: Optimal task layer assignment and Kendall's τ between the layer assignment and the pipeline (surface, syntactic and semantic information). The "τ syn sem" column reports Kendall's τ over only the syntactic task and semantic tasks.

POS	Const	Deps	Ent	SRL	Coref	SPR	Relns
0.659	0.413	0.493	0.377	0.333	0.428	0.370	0.261

Table 2: Kendall τ values of sequences of difference scores, by task, with depth.

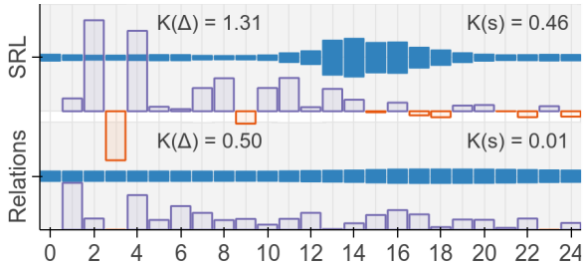

Figure 2: Excerpts of Tenney et al.'s (2019a) layer-wise metrics (Figure 2). Solid (blue) are mixing weights s_τ; outlined (purple) are difference scores Δ_τ.

The essential problem with Tenney et al.'s (2019a) claims is that their final three, most semantic tasks, are not localizable at all. Because of this, "expected layer," which is really more of a layer torque, is uninterpretable, and not convincingly relatable to layer depth. Even in less dispersed tasks, weighing difference scores by layer number is a statistical fallacy, because layer numbers are ordinal data. Instead, we could look at the positions of the n layers with the highest difference scores, in which n is arbitrary, or analyse the entire sequence of difference scores as a sequence.

Because those scores are not provided in their paper other than in relative terms through unlabelled histograms, what the present authors can do is arrange the ranks of difference scores (1 is highest) by layer (lowest is first) and compute a Kendall's coefficient with respect to the ordered sequence $1, \ldots, 24$ (BERT-large has 24 layers). "Deep" tasks should receive low scores. These coefficients are shown in Table 2. While the Pearson correlation of the pipeline ordering with these scores is strong ($r = -0.793, p = 0.02$), it is not as strong as with the KL divergence of the underlying difference score distributions from uniformity. The claim that BERT mimicks the NLP pipeline is therefore, at best, inconclusive. The empirical data are equally consistent with the counterclaim that BERT is possessed of stripes of surface, syntactic and semantic information that are distributed in parallel throughout its layers, with the semantic information being more evenly dispersed.

The difference score of every probing task in Tenney et al. (2019a) peaks in the first four layers, incidentally, and, in 6 of the 8 tasks, peaks in the first layer.

Performance is Not a Practical Indicator of Knowledge Performance-based probes are ill-suited to investigating the structure of BERT, because performance is inherently unstable. Taking the tense task from Jawahar et al.'s (2019) result (Figure 1) as an example; the largest delta between layer 6 (definitely a middle layer) and layer 11 (definitely a top layer) is only 0.3%.

Because Tenney et al.'s (2019a) difference scores are learned, they are not the actual inter-layer deltas of F_1 performance. Here, however, each probe has access to exactly one more layer of BERT's contextual representation, and therefore higher layer probes should have access to no less information than lower layer probes. Thus, if probe performance is a good indicator of linguistic knowledge, no higher layer probes should perform worse than lower layer probes. And yet performance drops are prevalent and substantial (Figure 2). Tenney et al. (2019a) suggest that the added new layer introduces distracting features causing the probe classifier to overfit. This means that performance results reflect a combination of knowledge and the probe classifier's ability to generalise — this may be true, but these two variables are hard to separate. Furthermore, neural architectures are stochastic, and so the effect of randomness in performance must also be considered. This is why statistical analysis of observations is crucial to the integrity of the conclusions.

The debate on how to interpret performance in probing is still on-going. Hewitt and Liang (2019) pondered: "when a probe achieves high accuracy on a linguistic task using a representation, can we conclude that the representation encodes linguistic structure, ... has the probe just learned the task?" There are two alternative interpretations of perfor-

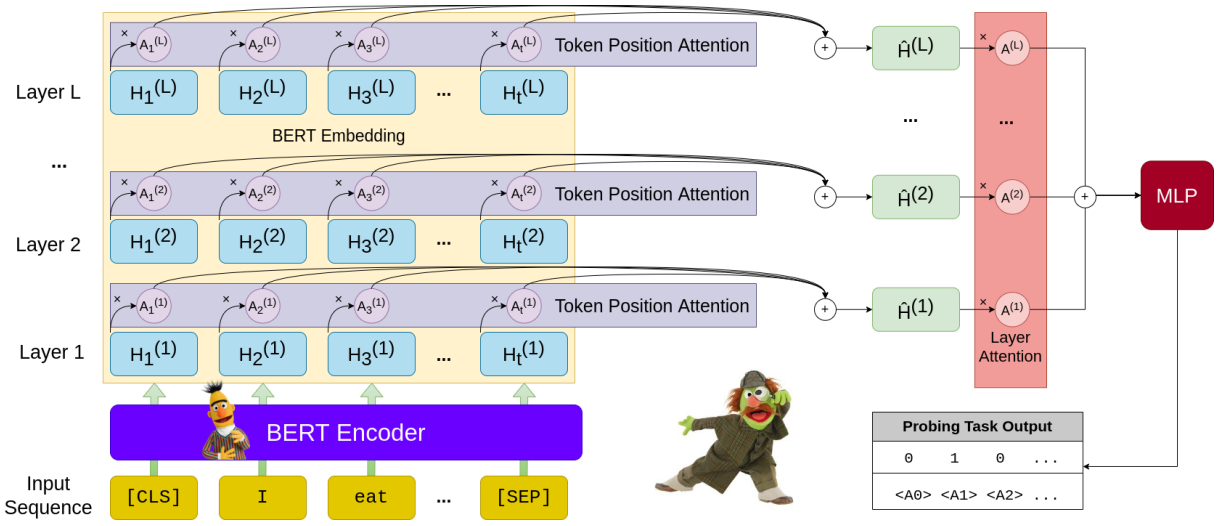

Figure 3: GridLoc model architecture (§3). The BERT-encoded representation of an input sentence first goes through a self-attention pooling process (Lee et al., 2017): an RNN model takes each layer's BERT embeddings $\mathbf{H}^{(\ell)}$ as input and specifically generates an attention weight $\mathbf{A}^{\text{token},(\ell)}$ for that layer. Then the attended representation $\hat{\mathbf{H}}$ goes through another self-attention pooling process, and generates a layer attention weight $\mathbf{A}^{\text{layer}}$. We finally train an MLP classifier that takes the combined representation $\tilde{H}$ as input, and generates a prediction for the probing task. By observing the two attention weights, $\mathbf{A}^{\text{token}}$ and $\mathbf{A}^{\text{layer}}$, we can understand which part of BERT's representation the model assigns importance to.

mance: ease of extraction (Hewitt and Liang, 2019), and mutual information (Pimentel et al., 2020; Pimentel and Cotterell, 2021).

Our reappraisal of Jawahar et al. (2019) and Tenney et al.'s (2019a) results shows that performance is neither intuitively interpretable nor an accurate reflection of knowledge, if performance can be regarded as a reflection at all. It is also a measure entangled with the quality of the probe classifier and randomness.

Better Control over Attention-based Probing Attention-based probing is less subject to the aforementioned issues. Firstly, since attention is not the optimisation target, it does not suffer the problem of overfitting. Secondly, although not completely exempt from controversy, attention has generally proven to be a good indicator of feature-importance (Serrano and Smith, 2019; Wiegreffe and Pinter, 2019). Therefore, we can adopt the view of probing results as a reflection of the existence of linguistic, task-specific features. More importantly, since the attention mechanism is purposefully introduced into the probing procedure, we can have a greater degree of freedom and better control over what is probed and where.

Tenney et al. (2019a) used a single shared set of attention weights for every input sentence. This practice cannot capture BERT's variance across sentences (as we will show in §5). But this is not inherent to any limitation of probing techniques in general — the self-attention pooling mechanism (Lee et al., 2017) trains a separate attention network that can assign different attention weights based on the input. Lee et al.'s (2017) self-attention pooling method was originally used by Tenney et al. (2019b) to generate a single span representation over an arbitrarily long span of tokens. The method can also yield a different attention weight for every input sentence. Furthermore, self-attention pooling provides an attention weight distribution over token positions. The similarities between token position attention and layer attention in fact could allow one to analyse the distribution of task knowledge across token positions.

3 GridLoc

To leverage all of these degrees of freedom, we present here a novel probing method called *GridLoc*. Figure 3 presents an overview of the probing process. Given an input sentence $S = [t_1, \ldots, t_T]$, BERT produces an L-layer embedding for each token $\mathbf{H}_t = [H_t^{(1)}, \ldots, H_t^{(L)}]$. GridLoc can produce a more complete picture of where task specific knowledge resides, by breaking down the probe's attention weight across both token positions and layers, as well as across random seeds and training iterations.

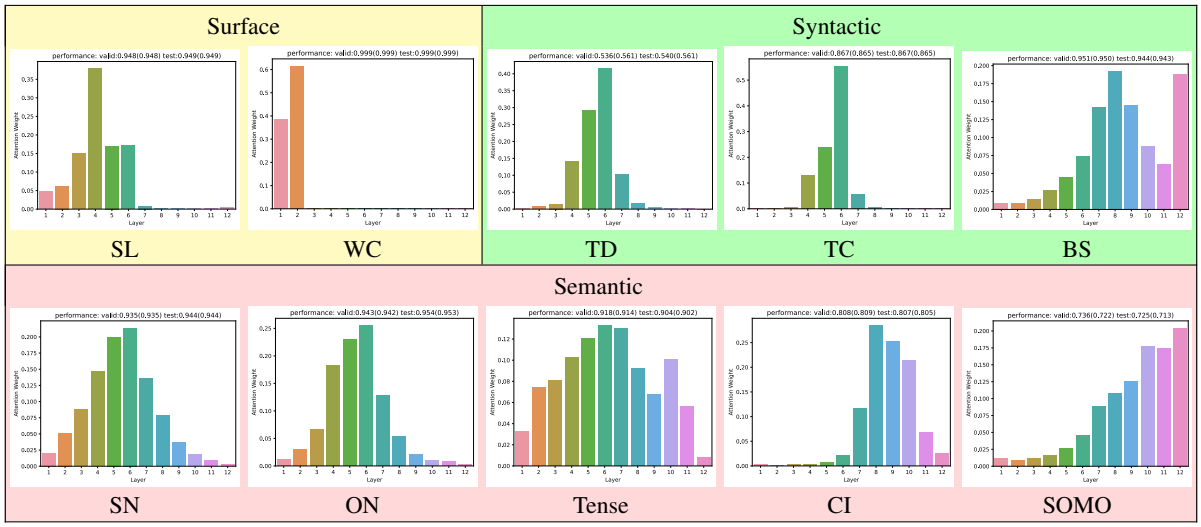

Figure 4: GridLoc average layer attention weight distribution for every SentEval task. For each task, the plot shows the average layer attention weight of all of the test set sentences during the top-performing epoch (by validation set) with random seed 0. We observed a moderate $\tau = 0.503$ with the entire pipeline (surface + syntactic + semantic), but a mere $\tau = 0.134$ with only the syntactic and semantic tasks. GridLoc confirms our earlier observation: surface tasks attend to lower layers, but syntactic and semantic tasks are inseparable.

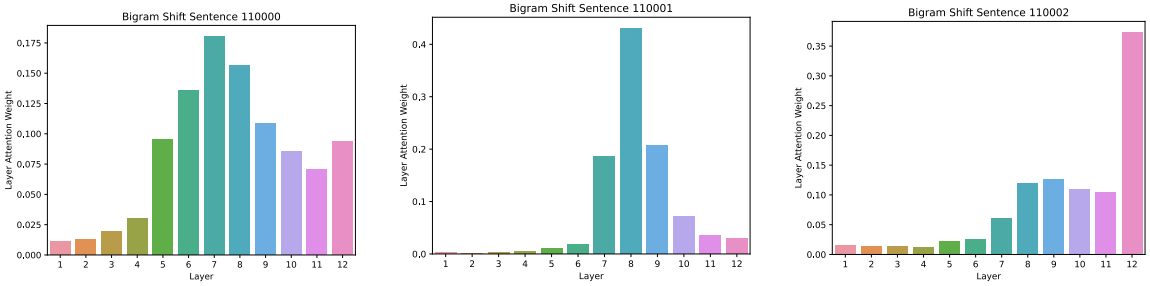

(a) Layer attention weights of the same probe of the first three Bigram Shift test-split sentences. The layer attention weight distributions differ widely.

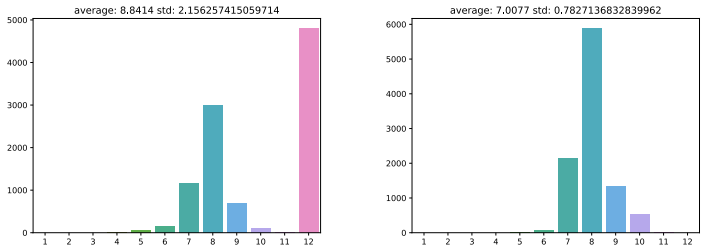

Seed: 0, Epoch: 7. Seed: 1, Epoch: 8.

(b) Distribution of the layer with the highest attention score over the Bigram Shift test-set sentences for two probing runs with different random seeds. Both probes are generated at their top performing (by validation) epochs. Distributions can exhibit substantial variance (left: $\sigma = 2.16$, right: $\sigma = 0.78$). For the run with seed 0 (left), there is also a spike in sentences at the 12th layer that is not observed in the run with seed 1 (right).

Task	σ
SL	1.468
WC	0.786
BS	1.95
TD	0.584
TC	1.025
Tense	2.359
SN	1.188
ON	0.903
SOMO	1.589
CI	0.953

(c) Standard deviation of the distribution of the layer with the highest attention score of every SentEval probing task.

Figure 5: Variance of probing results among sentences.

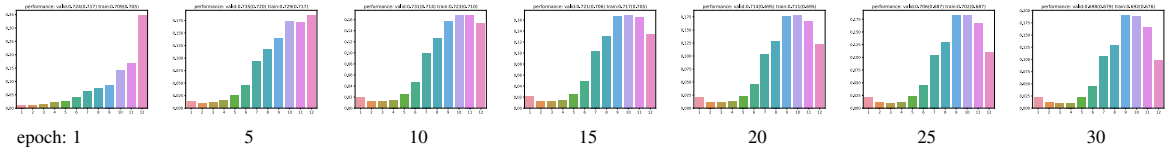

epoch: 1 5 10 15 20 25 30

Figure 6: An example (SOMO with random seed 0) of the average attention weight distribution change over training iterations.

Token Position To understand the task-specific feature distribution across token for each layer, we exploit the aforementioned self-attention pooling (Lee et al., 2017) method to learn a *token attention:*

$$\mathbf{A}^{\text{token},(\ell)} = \text{softmax}(\mathbf{w}_{\text{token}} \cdot \text{RNN}(\mathbf{H}^{(\ell)}))$$

specific to each layer of embeddings $\mathbf{H}^{(\ell)} = [H_{t_1}^{(\ell)}, \ldots, H_{t_T}^{(\ell)}]$. Then, we can obtain an attended hidden representation:

$$\hat{\mathbf{H}} = \mathbf{A}^{\text{token}} \cdot \mathbf{H}$$

Layer Then, we learn a sentence-specific *layer attention:*

$$\mathbf{A}^{\text{layer}} = \text{softmax}(\mathbf{w}_{\text{layer}} \cdot \hat{\mathbf{H}}^{(\ell)})$$

from the attended contextual embedding of the entire sequence. Finally, we can train the probe classifier on the fully attended representation of the input sequence:

$$\tilde{H} = \mathbf{A}^{\text{layer}} \cdot \hat{\mathbf{H}}$$

by attending to both the tokens and the layers.

Randomness and Training To understand the variance of the probe result relative to the random seed, we repeat each of our experiments with 20 seeds ($0 \sim 19$). We also maintain a record of 30 epochs of training for each probe.

4 Experimental Setup

We used all 10 tasks in SentEval (Conneau and Kiela, 2018) as described in §2.1. To be consistent with J&T's results, we conducted our experiment using the uncased BERT-base model and Jawahar et al.'s (2019) hyperparameters.[3]

5 Experimental Results

5.1 Layers Alone do Not Recapitulate the Pipeline

With our new probe, we calculate the average layer attention weight for each task and report in Figure 4 an example for every task. The average layer attention weight is calculated by summing up every test sentence's layer attention weight, and then normalising by the size of the test set. The average task layer attention weight is a good global indicator of the spread of task-specific features in BERT. Our results agree with our observations based upon

[3] https://github.com/ganeshjawahar/interpret_bert

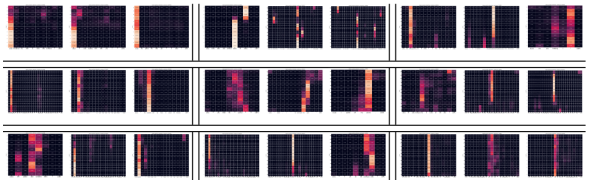

Figure 7: Token-position attention-weight plots for the first 3 sentences of the SentEval test set on all SentEval tasks (from top left to bottom right: SL, WC, Tense, SN, ON, BS, TC, SOMO and CI). The attention weights are displayed as a 2-dimensional heat map; each column corresponds to a token and each row corresponds to a BERT layer. Brighter colours represent larger attention weights. For most sentences, the token-position attention at every layer attends to the same token, hence the bright vertical line.

POS	Count	Top Layer
PUNCT	23402	7.18
NOUN	19077	**7.33**
VERB	18277	5.03
PRON	16120	6.68
ADP	11129	**3.19**

(a) Average best-attending layer of the 5 most common POSs. Maximum and minimum are highlighted in bold.

POS 1	POS 2	|PB r|
PUNCT	ADP	0.483
ADJ	ADP	0.462
ADP	DET	0.460
NOUN	ADP	0.438
PRON	ADP	0.399

(b) Best 5 absolute point-biserial correlations between the best attending layers of tokens with different POSs. p-values are less than 10^{-323}.

Table 3: The tree depth probe attends to tokens with different POS at different layers.

Jawahar et al. (2019): although surface task features are dense in lower layers, and both syntactic and semantic task features are spread out between mid to upper layers; the inseparable syntactic and semantic tasks show that BERT layers alone do not recapitulate the putative pipeline.

This observation is corroborated by a Kendall's τ test between every run (20 random seeds $\times$ 30 epochs) of each task's top performing layer and the 1–3 pipeline-based ranking described in §2.1.[4] Since now we have 600 data points for each task, our correlation test result is more robust. Again, we observed a moderate $\tau = 0.503$ with the entire pipeline (surface + syntactic + semantic), but a mere $\tau = 0.134$ with only the syntactic and semantic tasks.

5.2 Variance through Sentences, Randomness and Training Time

Nevertheless, average attention weight is a global measure that withholds important nuances regarding the variance of the probe along several dimensions. As shown in Figure 5a, layer attention as-

[4] The unique ranking test is discarded as it cannot generalise to our situation with multiple runs.

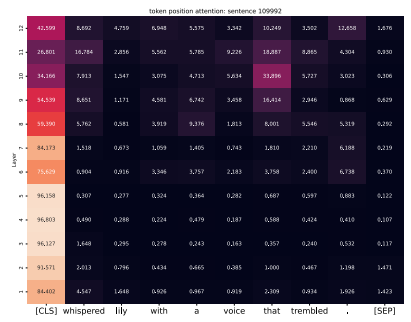

(**SL**) The probe classifier attends to the `[CLS]` token, as it is often regarded as the "embedding of the sentence," and length is a global feature of the sentence.

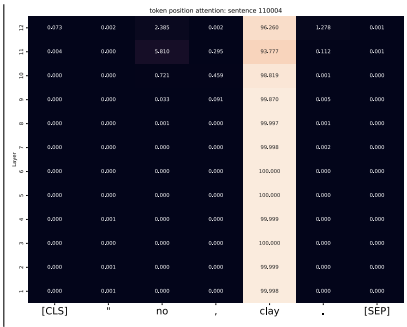

(**WC**) The probe classifier attends to the target word. In this particular example, every token-position weight attends to the target token "clay."

(**Tense**) The probe classifier attends to the verb or its tense-morphology-bearing "wordpiece" (Wu et al., 2016), such as `##ed` or `##es`.

(**SN**) The probe classifier attends to the subject noun or its number-morphology-bearing "wordpiece."

(**ON**) Similar to SN, the classifier attends to the object noun of the sentence or its number-morphology-bearing "wordpiece."

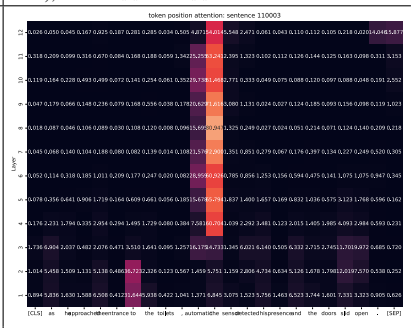

(**BS**) The probe classifier attends to the words that are being inverted. In this case, the two inverted words are "corners" and "the." If an original sentence is encountered, the classifier will place heavier weights on places where inversions are noticeable, such as prepositions, determiners and punctuation.

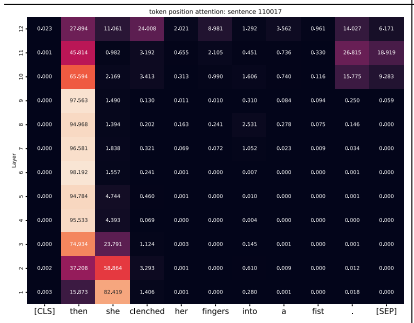

(**TC**) Top constituents attend to the first one or two words in the sentence, as tags with one presentential modifier followed by NP VP are commonplace. E.g., in this case, the sequence is labelled as `RB_NP_VP_`.

(**SOMO**) The probe classifier attends to the verb or noun (here, the verb "confused") that is replaced. Upon encountering an unaltered sentence, the classifier will attend to common verbs and nouns that are likely to be replaced.

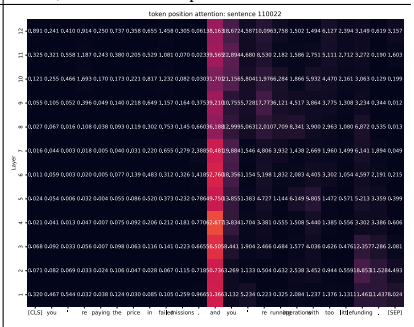

(**CI**) The probe classifier attends to the coordinating conjunction (CC). CC is crucial in determining whether the sentence has inverted coordination since different CCs serve different purposes when connecting two parts of a sentence.

Figure 8: Example token-position attention plots and their pattern analyses. Similar to Figure 7, the attention weight is displayed in a 2-dimensional heat map, with larger weights associated with brighter colours. The tokens of the example sentence are displayed along the x axis. The number on each cell is the attention weight as a percentage. Since attention weight is normalised by softmax at each layer, numbers in every row should sum up to 100.

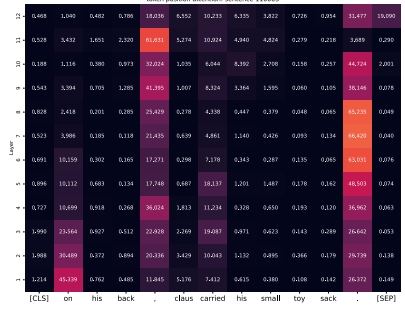

Figure 9: Tree Depth token-position attention weights. The token-position attention tends to focus on tokens with different parts-of-speech at different layers. For instance, prepositions such as *to*, *of*, *on* and *at* have higher attention weights at lower layers, and punctuation has higher attention weights at higher layers.

signs drastically different weights to different input sentences. This observed variance is not idiosyncratic. Figure 5b aggregates counts of the layer with the highest attention score over the test set sentences. The difference between seed 0 and seed 1 also emphasises that probe results are not immune to random initialization effects. This high variance is also not unique to Bigram Shift. Table 5c shows the overall standard deviation of every SentEval task. In all but two tasks, the standard is one full layer or more. Figure 6 shows how average attention weight can change during training. In this particular run, the distribution does not stabilize until about epoch 15. This again demonstrates the importance of utilising the self-attention pooling mechanism, which enables us to capture these variations.

5.3 Consistently Idiosyncratic Token Positions

Contrary to the variance we observed by layer, our token-position attention results are more stable — *tree depth* is the only exception, and we will discuss it shortly. As indicated in Figure 7, almost every sentence's token-position attention focuses on the same token in every layer. The choice of that token position is not arbitrary — there are linguistic reasons for them. Figure 8 shows one example for every task along with our analysis.

5.4 Tree Depth: an Insightful Anomaly

The token-position attention-weight result of the *tree-depth* task is the only exception to the bright vertical line pattern. Here, the probe attends to multiple tokens at different layers (Figure 9). The attention patterns are not arbitrary, however. As shown in Table 3a, tokens with different parts-of-speech[5] (POS) receive the most attention from the probe at different layers. Among the tokens, nouns attend to the highest layers at 7.33 (middle) and prepositions attend to the lowest layers at 3.19 (low). To verify the significance of the mapping between POS and layer attention, we conducted point-biserial correlation tests (Table 3b). We observed a moderate correlation between several pairs of POS, confirming that the probe can discriminate between them in this manner.

Although how any of this might relate to tree depth is unclear, this finding is still important in two ways. First, the Damoclean sword that the probe classifier is merely able to generalise the probing corpus (Hewitt and Liang, 2019; Pimentel et al., 2020) has been removed from over the claim that BERT is in possession of linguistic knowledge. POS information is certainly not self-evident in a corpus labelled with tree depth. Here we have a probe on one auxiliary linguistic task attesting to another linguistic phenomenon.

Second, the distribution of linguistic features defies J&T's proposed distribution of knowledge in BERT. What we see here is not different levels of linguistic knowledge from the pipeline occupying different layers of BERT, but rather different information from the same pipeline level (i.e., distinct POS labels) occupying different layers of BERT. This insight would not have been available without a probe that takes both the token position dimension and the layer dimension into account.

6 Conclusion

Did BERT rediscover an NLP pipeline? Not in a naïve, architectural sense. GridLoc reveals a structure in BERT that is more intricate than a flowchart of a pipeline could accurately portray, and yet it does seem to be linguistically founded. We find that probing results regarding BERT layers are unstable, diverging across sentence input, random seeds and the early iterations of training. The distribution of linguistically motivated task features along token positions, on the other hand, is relatively more stable. Moreover, GridLoc's results on tree depth provide preliminary evidence of POSs being used to conduct novel but linguistically generalizable inference concerning a derivative syntactic phenomenon.

Acknowledgements

We thank Zining Zhu (University of Toronto) for all the insightful discussion. We also want to thank the anonymous reviewers for providing informative comments and suggestions.

References

Yossi Adi, Einat Kermany, Yonatan Belinkov, Ofer Lavi, and Yoav Goldberg. 2017. Fine-grained analysis of sentence embeddings using auxiliary prediction tasks. In *5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Conference Track Proceedings*. OpenReview.net.

Kevin Clark, Urvashi Khandelwal, Omer Levy, and Christopher D. Manning. 2019. What Does BERT

[5]Generated by the Stanza (Qi et al., 2020) package.

Look at? An Analysis of BERT's Attention. In *Proceedings of the 2019 ACL Workshop BlackboxNLP: Analyzing and Interpreting Neural Networks for NLP*, pages 276–286, Florence, Italy. Association for Computational Linguistics.

Alexis Conneau and Douwe Kiela. 2018. SentEval: An Evaluation Toolkit for Universal Sentence Representations. In *Proceedings LREC 2018*, Miyazaki, Japan. European Language Resources Association (ELRA).

Alexis Conneau, German Kruszewski, Guillaume Lample, Loïc Barrault, and Marco Baroni. 2018. What you can cram into a single $&!#* vector: Probing sentence embeddings for linguistic properties. In *Proceedings ACL 2018*, pages 2126–2136, Melbourne, Australia. Association for Computational Linguistics.

Jerry A. Fodor. 1983. *The Modularity of Mind*. The Modularity of Mind. The MIT Press, Cambridge.

M. F. Garrett. 1975. The Analysis of Sentence Production. In Gordon H. Bower, editor, *Psychology of Learning and Motivation*, volume 9, pages 133–177. Academic Press.

M. F. Garrett. 1980. Levels of Processing in Sentence Production. In B. Butterworth, editor, *Language Production*, volume 1. Academic Press, London.

John Hewitt and Percy Liang. 2019. Designing and Interpreting Probes with Control Tasks. In *Proceedings EMNLP-IJCNLP 2019*, pages 2733–2743, Hong Kong, China. Association for Computational Linguistics.

Dieuwke Hupkes and Willem Zuidema. 2018. Visualisation and 'Diagnostic Classifiers' Reveal how Recurrent and Recursive Neural Networks Process Hierarchical Structure (Extended Abstract). In *Proceedings IJCAI 2018*, pages 5617–5621, Stockholm, Sweden. International Joint Conferences on Artificial Intelligence Organization.

Ray Jackendoff. 2000. Fodorian Modularity and Representational Modularity. In Yosef Grodzinsky, Lewis P. Shapiro, and David Swinney, editors, *Language and the Brain*, pages 3–30. Elsevier.

Ganesh Jawahar, Benoît Sagot, and Djamé Seddah. 2019. What Does BERT Learn about the Structure of Language? In *Proceedings ACL 2019*, pages 3651–3657, Florence, Italy. Association for Computational Linguistics.

M. G. Kendall. 1938. A New Measure of Rank Correlation. *Biometrika*, 30(1/2):81–93.

Kenton Lee, Luheng He, Mike Lewis, and Luke Zettlemoyer. 2017. End-to-end Neural Coreference Resolution. In *Proceedings EMNLP 2017*, pages 188–197, Copenhagen, Denmark. Association for Computational Linguistics.

W. J. M. Levelt. 1989. *Speaking: From Intention to Articulation*. ACL-MIT Press Series in Natural-Language Processing. MIT Press, Cambridge, Mass.

Christopher Manning, Mihai Surdeanu, John Bauer, Jenny Finkel, Steven Bethard, and David McClosky. 2014. The Stanford CoreNLP Natural Language Processing Toolkit. In *Proceedings ACL 2014: System Demonstrations*, pages 55–60, Baltimore, Maryland. Association for Computational Linguistics.

William Marslen-Wilson and Lorraine K. Tyler. 1987. Against Modularity. In Jay L. Garfield, editor, *Modularity in Knowledge Representation and Natural-Language Understanding*, pages 37–62. The MIT Press.

Tiago Pimentel and Ryan Cotterell. 2021. A Bayesian Framework for Information-Theoretic Probing. In *Proceedings EMNLP 2021*, pages 2869–2887, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Tiago Pimentel, Josef Valvoda, Rowan Hall Maudslay, Ran Zmigrod, Adina Williams, and Ryan Cotterell. 2020. Information-Theoretic Probing for Linguistic Structure. In *Proceedings ACL 2020*, pages 4609–4622, Online. Association for Computational Linguistics.

Peng Qi, Yuhao Zhang, Yuhui Zhang, Jason Bolton, and Christopher D. Manning. 2020. Stanza: A Python Natural Language Processing Toolkit for Many Human Languages. In *Proceedings ACL 2020: System Demonstrations*, pages 101–108, Online. Association for Computational Linguistics.

Sofia Serrano and Noah A. Smith. 2019. Is Attention Interpretable? In *Proceedings ACL 2019*, pages 2931–2951, Florence, Italy. Association for Computational Linguistics.

Karen Sparck Jones. 2000. R.H. Richens: Translation in the NUDE. In W.J. Hutchins, editor, *Early Years in Machine Translation*, pages 263–278. John Benjamins.

Ian Tenney, Dipanjan Das, and Ellie Pavlick. 2019a. BERT Rediscovers the Classical NLP Pipeline. In *Proceedings ACL 2019*, pages 4593–4601, Florence, Italy. Association for Computational Linguistics.

Ian Tenney, Patrick Xia, Berlin Chen, Alex Wang, Adam Poliak, R. Thomas McCoy, Najoung Kim, Benjamin Van Durme, Samuel R. Bowman, Dipanjan Das, and Ellie Pavlick. 2019b. What do you learn from context? Probing for sentence structure in contextualized word representations. In *International Conference on Learning Representations*.

Jesse Vig. 2019. A Multiscale Visualization of Attention in the Transformer Model. In *Proceedings ACL 2019: System Demonstrations*, pages 37–42, Florence, Italy. Association for Computational Linguistics.

Sarah Wiegreffe and Yuval Pinter. 2019. Attention is not not Explanation. In *Proceedings EMNLP-IJCNLP 2019*, pages 11–20, Hong Kong, China. Association for Computational Linguistics.

Yonghui Wu, Mike Schuster, Zhifeng Chen, Quoc V. Le, Mohammad Norouzi, Wolfgang Macherey, Maxim Krikun, Yuan Cao, Qin Gao, Klaus Macherey, Jeff Klingner, Apurva Shah, Melvin Johnson, Xiaobing Liu, Łukasz Kaiser, Stephan Gouws, Yoshikiyo Kato, Taku Kudo, Hideto Kazawa, Keith Stevens, George Kurian, Nishant Patil, Wei Wang, Cliff Young, Jason Smith, Jason Riesa, Alex Rudnick, Oriol Vinyals, Greg Corrado, Macduff Hughes, and Jeffrey Dean. 2016. Google's Neural Machine Translation System: Bridging the Gap between Human and Machine Translation.

Zeguan Xiao, Jiarun Wu, Qingliang Chen, and Congjian Deng. 2021. BERT4GCN: Using BERT Intermediate Layers to Augment GCN for Aspect-based Sentiment Classification. In *Proceedings EMNLP 2021*, pages 9193–9200, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Zining Zhu, Soroosh Shahtalebi, and Frank Rudzicz. 2022. Predicting fine-tuning performance with probing. In *Challenges & Perspectives in Creating Large Language Models*.

HG2Vec: Improved Word Embeddings from Dictionary and Thesaurus Based Heterogeneous Graph

Qitong Wang and **Mohammed J. Zaki**
Computer Science Department
Rensselaer Polytechnic Institute
wangq19@rpi.edu and zaki@cs.rpi.edu

Abstract

Learning word embeddings is an essential topic in natural language processing. Most existing works use a vast corpus as a primary source while training, but this requires massive time and space for data pre-processing and model training. We propose a new model, HG2Vec, that learns word embeddings utilizing only dictionaries and thesauri. Our model reaches the state-of-art on multiple word similarity and relatedness benchmarks. We demonstrate that dictionaries and thesauri are effective resources to learn word embeddings. In addition, we exploit a new context-focused loss that models transitive relationships between word pairs and balances the performance between similarity and relatedness benchmarks, yielding superior results.

1 Introduction

Word embeddings are highly effective for various applications, ranging from recommender systems to named entity recognition (Kubal and Nimkar, 2019). They aim to map words into vectors in a high dimensional space, such that a higher similarity between word embeddings captures a closer semantic relationship, whereas a low or negative similarity indicates unrelated or opposite meaning. As a result, predicting whether the embeddings of two words are similar or not is a common approach to evaluate the quality of the embeddings.

There are two main perspectives to describe the relationship between a pair of words according to Hill et al. (2015): similarity and relatedness. Similarity means that the two words can substitute for each other without generating grammatical mistakes, while relatedness means that the two words always appear together in a context. For example, *coffee* and *tea* is a word pair with high similarity and low relatedness. Being popular breakfast drinks, they share many similar properties (both physically and grammatically), but they are not related to each other. On the other hand, *green* and *tea* is a word pair with high relatedness and low similarity since green tea is a common type of tea. They do not share similar proprieties, but it is common for them to appear together in a sentence.

Most existing works utilize a large corpus to train word embeddings. However, as pointed out by Kiela et al. (2015), models based on association data, like Wikipedia corpus, have better performance on relatedness tasks, whereas models based on thesauri have better performance on similarity tasks. Therefore, in this paper, we propose a new model, HG2Vec, that only relies on dictionaries and thesauri to construct contexts via a heterogeneous graph. We demonstrate that this method can improve the performance of relatedness tasks while maintaining high accuracy on similarity tasks. Our main contributions are as follows:

- We utilize dictionaries and thesauri as the only resources (without using any text corpus) to train word embeddings, and obtain state-of-art results on several benchmarks.
- We demonstrate that learning synonyms and antonyms is necessary for word embedding models, so a model can distinguish the words with opposite meanings that appear in similar contexts.
- We propose a context-focused loss to boost learning in closely related contexts. It is based on modeling transitive synonym and antonym relationships from the thesauri as well as word co-occurrences from dictionary definitions, which balances the performance between similarity and relatedness tasks.

2 Related Work

Many models require a large corpus to learn word embeddings; Word2Vec (Mikolov et al., 2013) is one of them. It iterates through the text with each

Proceedings of the 29th International Conference on Computational Linguistics, pages 3154–3163
October 12–17, 2022.

target word generating a surrounding context window and corresponding negative samples. The goal is to learn embeddings that maximize the similarity between the target and context words while minimizing the similarity between the target and negative sampled words. BERT (Devlin et al., 2018) is an effective contextual embedding model. It tokenizes the input sentences, iterates through the corpus text by blocks, and learns token embeddings by utilizing the attention-based transformer model. BERT uses masked-language modeling and next sentence prediction as the self-supervised tasks during pre-training.

Several approaches leverage dictionaries and other external sources to improve the performance of Word2Vec and BERT. Faruqui et al. (2014) pretrain with other models and then use a dictionary-based relational graph to retrofit semantic information. The paper minimizes the difference between a word embedding from pre-trained models and its corresponding embeddings from the relational graph, and maximizes the similarity between a relational graph word embedding with its neighbors. dLCE (Nguyen et al., 2016) extends the skip-gram model with negative sampling introduced by (Levy and Goldberg, 2014) with synonyms and antonyms. Dict2Vec (Tissier et al., 2017) introduces dictionaries as a resource to train word embeddings. Word pairs are grouped into two types: strong pairs and weak pairs. For each target word, Dict2Vec maximizes the similarity between a context word and its strong and weak pairs for each target word. DRG2Vec (Shu et al., 2020) creates a graph based on the TF-IDF relationship between word pairs in dictionaries. It applies both depth-first and breadth-first based sampling to generate context paths from the graph. It trains the model on both the Wikipedia corpus and the generated context. Dict-BERT (Yu et al., 2021) appends the definition of rare words at the end of the input corpus. Besides mask language modeling, it maximizes the mutual information between the context and the definitions. It also samples wrong definitions during training to check whether the model can distinguish them.

Some models do not rely on a text corpus. CPAE (Bosc and Vincent, 2018) is an auto-encoder model that only relies on dictionaries. It uses an LSTM to reproduce the target word after processing its definitions. Ruzzetti et al. (2021) propose DefiNNet and DefBert to utilize dictionaries to predict the meanings for out-of-vocabulary words. Jana et al. (2022) create a Distributional Thesaurus Network to gather the information from thesauri, showing that utilizing thesauri can improve the performance of Word2Vec. In addition, Zhang et al. (2019) leverage dictionaries as a tool to visualize how other word embeddings contribute to the target word embedding, which is helpful in analyzing the model and improving the performance.

Our model is an enhancement over Dict2Vec and DRG2Vec, where we create a heterogeneous graph from dictionaries and thesauri, and generate documents from the sampled paths. Then, we train the documents to maximize the similarity with related pairs, explicitly considering synonyms and antonyms (along with their transitive relationships), as well as strong and weak pairs.

3 HG2Vec Methodology

3.1 Graph Construction

We first parse the word pairs from the dictionaries and thesauri, and construct a heterogeneous graph for sampling paths. Fig. 1 shows an example of an undirected heterogeneous graph. Each node refers to a word, and each edge refers to a relationship between two words. There are four different types of edges: strong edge (purple), weak edge (blue), synonym edge (black), and antonym edge (red). The edges also have weights, so the graph is weighted, and heterogeneous in terms of edges.

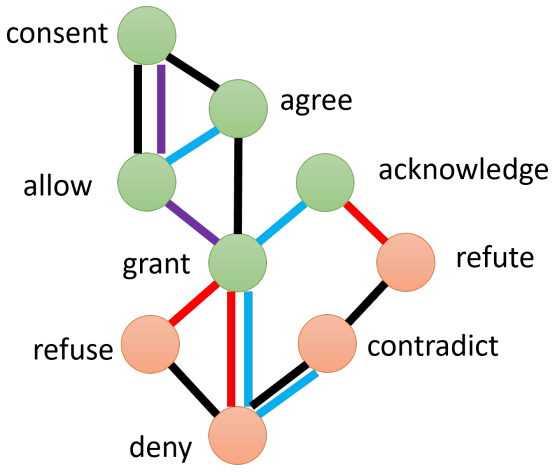

Figure 1: A sample undirected heterogeneous graph in HG2Vec. Each node refers to a word, and each edge refers to a relationship between words. There are four types of edges: strong (purple), weak (blue), synonym (black), and antonym (red). There may be more than one type of edge between two nodes, but all nodes are of the same type (for illustration only, we use green to mark the synonyms of *grant*, and red to mark its antonyms).

Strong and Weak Edges Following the approach in Dict2Vec (Tissier et al., 2017), for a word pair (w_a, w_b), if w_a appears in the definition of w_b in a dictionary, then there is a weak edge between them. If w_b also appears in w_a's definition, we instead add a strong edge between the two. In Fig. 1, since *agree* appears in the definition of *allow*, there is a weak edge between the two. Since *consent* and *allow* both appear in each other's definition, there is a strong edge between them. To model the relative importance of these two relationships, the weight of strong vs. weak edges is $2:1$ in our undirected heterogeneous graph.

Synonym and Antonym Edges We directly extract the synonym and antonym pairs from thesauri. For a word pair (w_a, w_b), if w_a appears as a synonym of w_b, then there is a synonym edge between w_a and w_b. If w_a appears as an antonym of w_b, then there is an antonym edge between them. For example, in Fig. 1, *refuse* is a synonym of *deny* and an antonym of *grant*. The weights of synonym and antonym edges are $1:-1$ in the graph.

Heterogeneous Graph From Fig. 1, we can see that there may be more than one type of edge between two nodes. In most cases, strong edges and weak edges indicate a relationship between words with similar meanings. However, some corner cases refer to the opposite meaning. For example, there is an antonym edge and a weak edge between *grant* and *deny*. As a result, although strong edges and weak edges can depict the relatedness of different words, they may also contain negative words in terms of semantics. If we only use strong edges and weak edges, the model may obscure the opposite meanings. Instead, we leverage thesauri as an external source to automatically consider these related but opposite pairs.

3.2 Path Generation

Our heterogeneous graph has edges with both positive and negative weights, so we cannot directly sample paths from the graph. We first illustrate the sampling for positive edges (strong and weak pairs, and synonyms) and then discuss how to handle negative weights (antonyms).

As such, dictionaries generate strong and weak edges, and thesauri generate synonym and antonym edges. We first generate the paths comprising strong and weak edges and then generate the paths comprising synonym and antonym edges.

Random Walks We generate the paths via random walks [1] that combine both Depth-first Sampling (DFS) and Breadth-frist Sampling (BFS) as first introduced in Node2Vec (Grover and Leskovec, 2016). We denote by n_x the x-th node in a walk, and we start generating the walk with node n_0. For a node $n_x = v_i$, the next node is denoted $n_{x+1} = v_j$, and the previous node is denoted $n_{x-1} = v_h$. Both v_h and v_j are neighbors of v_i. After we randomly choose one of v_i's neighbor as n_{x+1}, we choose one of v_j's neighbor as n_{x+2}. We keep iterating until the path reaches a desired length L (we use $L = 20$ in our model). The probability that we choose a node v_j from the neighbors $N(v_i)$ of v_i is defined as follows:

$$p(n_{x+1} = v_j | n_x = v_i) = \frac{\pi_{ij}}{\sum_{a \in N(v_i)} \pi_{ia}} \quad (1)$$

where

$$\pi_{ij} = \alpha(h, j) \cdot |w_{ij}| \quad (2)$$

In Eq. (2), w_{ij} is the edge weight from node v_i to node v_j. Since we have edges with negative weights, we use the absolute value of edge weight. Therefore, an antonym edge is equivalent to a synonym edge when calculating the probability. Note also that the probability of jumping to v_j also depends on the previous node v_h, which can be at a distance of $d = 0, 1,$ or 2 hops from v_j in the walk. As detailed below in Eq. (3), $\alpha(h, j)$ is a coefficient that controls the tendency of sampling between DFS and BFS; if $\alpha(h, j) = 1$, the algorithm is exactly the same as a random walk.

Node Sampling The distance from a node to the origin is increasing along the walk for DFS, which tends to explore the nodes away from the source. For example, in Fig. 1, the path may be $consent, agree, grant, deny$ if we utilize DFS. In contrast, BFS samples all the neighbors from the source first. It tends to explore the nodes around the origin. In Fig. 1, the path may be $consent, agree, allow, grant$ if we utilize BFS.

As above, let $n_{x-1} = v_h$, $n_x = v_i$, and $n_{x+1} = v_j$ for a sampled walk v_h, v_i, v_j. Here, $v_h = n_{x-1}$ is the node visited in the previous turn, and $v_j = n_{x+1}$ is the node sampled in the next turn. In Eq. (2), $\alpha(h, j)$ is a coefficient to balance

[1] We allow paths to have repeated nodes.

between DFS and BFS, defined as:

$$\alpha(h,j) = \begin{cases} p^{-1}, & \text{if } d(h,j) = 0 \\ 1, & \text{if } d(h,j) = 1 \\ q^{-1}, & \text{if } d(h,j) = 2 \end{cases} \quad (3)$$

In Eq. (3), $d(h,j)$ is the distance between v_h and v_j. If $d(h,j) = 0$, it means the next node is the same as the previous node, so it reflects the tendency to return to the source. When $d(h,j) = 1$, v_j and v_i are both neighbors of v_h. It reflects the tendency to explore around the node n_{x-1}. Finally, when $d(h,j) = 2$, the parameter reflects the tendency to sample new nodes away from the origin. Here, p is called *return parameter*, and a smaller p increases the tendency to return to the origin, whereas q is called *in-out parameter*, and a smaller q leads to a tendency to explore new nodes. For example, in Fig. 1, if v_h is *grant* and v_i is *agree*, a small p leads to choosing *grant* as v_j, a small q leads to choosing *consent*, and the walk samples *allow* if both p and q are large.

Generating Paths We separately sample the walks comprising of strong and weak edges, and those consisting of synonyms and antonym edges. The former directly uses the random walk approach outlined above. Fig. 2 shows a sample path consisting of strong and weak edges sampled from the heterogeneous graph in Fig. 1.

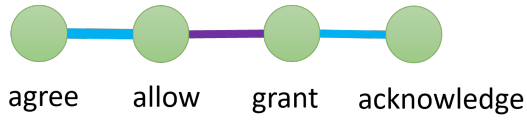

Figure 2: A path with strong (purple) and weak (blue) edges.

The sampling for synonym and antonym edges is more involved. We assume that all the nodes in one path have similar semantic meanings, so we cannot append an antonym to a path with synonyms since their meanings are opposite. However, since an antonym's antonym is essentially a synonym, we can directly attach that to an existing path. We then generate two paths: one for synonyms and another for antonyms. The first path stores the origin and nodes with a similar semantic meaning to the origin. The second path stores the nodes with opposite semantic meanings to the origin. We start at the source and generate the synonyms path via application of Eq. (1). However, when we encounter an antonym (of the source), we put this node as the first node in the antonyms path. We continue appending nodes to the antonyms path for all additional synonym edges encountered. If we visit another antonym, it means that the new node has a similar semantic meaning to the source, so we append the new node to the synonyms path. We continue to append to the synonyms path if we encounter additional synonym edges, and so on. Fig. 3 illustrates this process.

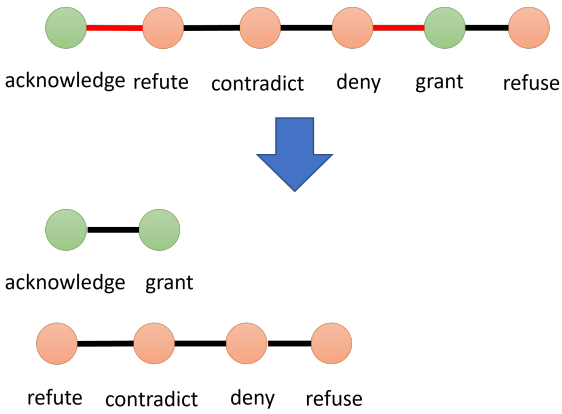

Figure 3: Random walks with synonym (black) and antonym (red) edges. The path on the top is if we were to sample both edges together. Instead, we divide it into two paths: the synonyms path stores the source node and other nodes with similar meanings, whereas the antonyms path stores the nodes with opposite meanings to the source. As a result, the nodes have a similar semantic meaning within each path. Starting with *acknowledge*, when we encounter *refute*, it goes to the antonyms path along with its synonyms *contradict* and *deny*. When we encounter *grant*, which is an antonym of deny, we append it to *acknowledge*. Finally, *refuse* again goes to the antonyms path.

3.3 Sampling Set for Training

We treat each sampled path above as a text document in our model. Next, we iterate through each text document with a context window. For a target word t at position x in a document, using window size ϕ, the context comprises the words $[x - \phi/2 : x + \phi/2]$. For example, consider the path *refute, contradict, deny, refuse* in Fig. 3. With window size $\phi = 3$, for the target word *contradict*, its context, denoted $\mathcal{C}(t)$, is *(refute, contradict, deny)*. The window selects the context nodes surrounding the target node, indicating a strong positive correlation with the target.

For each context word $c \in \mathcal{C}(t)$, we sample five negative, strong, weak, synonym, and antonym word pairs for c, denoted as $\mathcal{N}(c)$, $\mathcal{S}(c)$, $\mathcal{W}(c)$,

$\mathcal{Y}(c)$, and $\mathcal{A}(c)$, respectively. The negative words $\mathcal{N}(c)$ are the words sampled from the word dictionary that do not appear in the context $\mathcal{C}(t)$. The other word pairs are sampled from the dictionaries and thesauri, from the set of c's strong and weak pairs, or from c's synonyms and antonyms.

3.4 Context-focused Model

Like Word2Vec, we have two embeddings for each word w: input embeddings $\mathbb{I}(w)$, and output embeddings $\mathbb{O}(w)$. After training, we use the output embeddings for testing and evaluation.

To maximize the similarity of two embeddings, a and b, we calculate their dot product $p = a \cdot b$. A large positive value suggests that the two objects are similar, a large negative value suggests that they are negatively related, and a value around 0 suggests that the two are unrelated. We use $-\log \sigma(p)$ as the loss function where $\sigma(p) = 1/(1+\exp(-p))$ denotes the Sigmoid function.

Context-focused Transitive Loss HG2Vec considers six different contextual features to learn effective embeddings, namely, the context, negative sampled words, strong and weak pairs, and synonyms and antonyms. Each of these contributes to the overall loss function.

For a given target word t, we first maximize the similarity between t and its surrounding context words c. We use the output embeddings for the target words and the input embeddings for the context words. Therefore, the loss is given as the sum over each word pair (t, c) where $c \in \mathcal{C}(t)$:

$$\mathrm{I}_t = \sum_{c \in \mathcal{C}(t)} -\log \sigma(\mathbb{O}(t) \cdot \mathbb{I}(c))$$

To calculate the terms in overall loss, for each word $c \in \mathcal{C}(t)$, we sample a set of strong pairs $\mathcal{S}(c)$, weak pairs $\mathcal{W}(c)$, synonyms $\mathcal{Y}(c)$, antonyms $\mathcal{A}(c)$, and negative words $\mathcal{N}(c)$. The aim is to maximize the similarity of c with its strong, weak, and synonym pairs; and to minimize the similarity with its antonym and negative pairs. Therefore, we divide these sets into two groups, the positive group $\mathcal{P}^+(c) = \{\mathcal{S}(c), \mathcal{W}(c), \mathcal{Y}(c)\}$, and the negative group $\mathcal{P}^-(c) = \{\mathcal{A}(c), \mathcal{N}(c)\}$.

The loss functions within each group are similar; we use synonym pairs as an example. We want to maximize the similarity of c to any of its synonyms in $\mathcal{Y}(c)$; at the same time, the synonyms of other words in $\mathcal{C}(t)$ may also be synonyms of c. For example, in Fig. 1, for the context *(consent, allow, grant)*, the synonyms of *grant* (e.g., *acknowledge*) are also the synonyms of *consent*. However, there is no such synonym path from *acknowledge* to *consent* in the graph since thesauri do not include all the possible transitive relations. However, HG2Vec models these transitive synonym relationships by sampling five synonyms for each context word (e.g., if the context window is 5, then we will consider 25 synonyms in a context). We maximize the similarity between the word c and all the sampled synonyms from the context. The synonym loss for c is given as:

$$\mathrm{L}^+_{\mathcal{Y}(c)} = \beta_{\mathcal{Y}} \sum_{x \in \mathcal{C}(t)} \sum_{p \in \mathcal{Y}(x)} -\log \sigma(\mathbb{O}(c) \cdot \mathbb{I}(p))$$

Here, $\beta_{\mathcal{Y}}$ is a hyperparameter, denoting the importance of synonyms to the overall loss.

For the negative group $\mathcal{P}^-$, we want to minimize the similarity. Considering all (transitive) antonyms of c in a context, the loss is given as:

$$\mathrm{L}^+_{\mathcal{A}(c)} = \beta_{\mathcal{A}} \sum_{x \in \mathcal{C}(t)} \sum_{p \in \mathcal{A}(x)} -\log \sigma(-\mathbb{O}(c) \cdot \mathbb{I}(p))$$

Here, $\beta_{\mathcal{A}}$ specifies the weight for antonyms. As such $\beta_{P^+} = \{\beta_{\mathcal{S}}, \beta_{\mathcal{W}}, \beta_{\mathcal{Y}}\}$, and $\beta_{P^-} = \{\beta_{\mathcal{A}}, \beta_{\mathcal{N}}\}$ specify the weights for each set in the positive and negative groups.

The loss for one target word t in a path (or document) is given as:

$$\begin{aligned} \mathrm{L}_t = \mathrm{I}_t &+ \sum_{c \in \mathcal{C}(t)} \sum_{\mathcal{X} \in \mathcal{P}^+(c)} \mathrm{L}^+_{\mathcal{X}(c)} \\ &+ \sum_{c \in \mathcal{C}(t)} \sum_{\mathcal{X} \in \mathcal{P}^-(c)} \mathrm{L}^-_{\mathcal{X}(c)} \end{aligned} \tag{4}$$

Finally, we sum over all the targets in a path and sum up all the paths to get the total loss.

4 Experiments

For training, all experiments were conducted on a machine with a dual 20 core 2.5 GHz Intel Xeon Gold 6248 CPU, and Nvidia Tesla V100 GPU with 32GB memory. Our HG2Vec implementation is available as open source at `https://github.com/Qitong-Wang/HG2Vec`.

4.1 Datasets

Dictionaries and Thesauri For the strong pairs and weak pairs, we use the same data sources as Dict2Vec (Tissier et al., 2017) and DRG2Vec (Shu et al., 2020). After removing stop words, they

extract the strong and weak pairs from the Cambridge, Oxford, and Collins dictionaries, and dictionary.com. We extract the synonym and antonym pairs from Roget's Super Thesaurus 4th Edition (McCutcheon, 2010). In total, our heterogeneous graph contains 211,675 unique nodes/words, with 4,273,743 strong and weak edges, and 119,512 synonym and antonym edges.

Wikipedia HG2Vec does not rely on text corpus data, so we do not need to use Wikipedia corpus during training. However, for the baseline models, we pre-process the Wikipedia corpus from Nov. 2021 (`https://dumps.wikimedia.org/`), using the first 50 million words from the cleaned data.

4.2 Experimental Settings

Hyperparameters The hyperparameters for Dict2Vec and DRG2Vec are the same as those reported in their papers. For HG2Vec, for generating the paths, we use $\phi = 5$, $p = 1.5$, $q = 5.0$ and $L = 20$. Word embedding size is 300. We use five strong, weak, synonym, antonym, and negative samples for the loss computation. We train the model for five epochs; in each epoch, we sample one strong-weak path and one synonym-antonym path per node in the heterogeneous graph. To tune the β hyperparameters, we use a grid search from 0.1 to 5. We find that $\beta_{\mathcal{S}} = 0.4, \beta_{\mathcal{W}} = 0.4, \beta_{\mathcal{Y}} = 1.0, \beta_{\mathcal{A}} = 1.0, \beta_{\mathcal{N}} = 3.5$ give the best results. We try from 0.001 to 0.01 for the learning rate and choose 0.003 as the default. The weight for strong vs. weak edges is $2:1$, and for synonym vs. antonym edges is $1:-1$. Since both types of paths are sampled independently, the edge weights have importance only within their group.

Testing After we train the model, we extract the output word embeddings $\mathbb{O}(w)$, and use them for the benchmark evaluation. We train each model three times and report the average performance.

4.3 Benchmarks

We evaluate our model on word similarity benchmarks, which provide several pairs of words with a human-evaluated similarity score. We calculate the cosine similarity between the embedding vectors for each pair and rank the values from largest to smallest. Then, we use Spearman rank correlation to compare the order of pairs for each model versus the ground truth. A higher value indicates that the rank order of word pairs generated by a model is similar to the ground truth.

We collect the following datasets: Card-660 (Pilehvar et al., 2018), MC-30 (Miller and Charles, 1991), MEN-TR-3K (Bruni et al., 2014), MTurk-287 (Radinsky et al., 2011), MTurk-771 (Halawi et al., 2012), RG-65 (Rubenstein and Goodenough, 1965), RW-STANFORD (Luong et al., 2013), SimLex999 (Hill et al., 2015), SimVerb-3500 (Gerz et al., 2016), WS-353-ALL (Finkelstein et al., 2001), WS-353REL (Finkelstein et al., 2001), WS-353-SIM (Finkelstein et al., 2001), YP-130 (Yang and Powers, 2006). Among these datasets, MEN-TR-3K, MTurk-771, RG-65, YP-130, and WS-353-REL focus on testing the relatedness, and SimLex999, SimVerb-3500, and WS-353-SIM focus on testing the similarity between the word pairs. WS-353-ALL is the combination of WS-353-REL and WS-353-SIM. Card-660 and RW-STANFORD focus on testing rare words.

After generating the Spearman correlation of each dataset, we also report the weighted average over all the datasets, with the weight of each dataset being proportional to its size.

4.4 Baseline Methods

We compare HG2Vec with Word2Vec(Mikolov et al., 2013), Dict2Vec(Tissier et al., 2017), and DRG2Vec(Shu et al., 2020). For comparison, We include HG2Vec(wiki), for which we use the 50M Wikipedia corpus with dictionaries and thesauri as the input resource. We also compare with a target-only based model HG2Vec(target) where we replace $\mathcal{C}(t)$ with t in Eq. (4). This allows us to compare the effectiveness of context-focused loss. Note also that we report the result of Dict2Vec and DRG2Vec based on our runs. If we use wiki corpus as an external source and we use paths sampled with TF-IDF pairs from strong and weak pairs, we obtain DRG2Vec. If we omit the graph and use wiki corpus with strong and weak pairs, we obtain Dict2Vec. Finally, we also report ablation results when we omit strong, weak, synonyms, antonyms, and negative pairs, one at a time.

5 Evaluation

5.1 Similarity and Relatedness Experiments

Effectiveness of Dictionaries and Thesauri The results of evaluating the similarity and relatedness benchmarks are shown in Table 1. We can see that HG2Vec scores higher than Word2Vec,

Type	Benchmark	Word2Vec	Dict2Vec	DRG2Vec	HG2Vec wiki	HG2Vec target	HG2Vec
Relatedness	MEN-TR-3K	0.612	0.688	0.721	0.771	**0.782**	0.779
	MTurk-287	0.577	0.568	0.558	**0.666**	0.642	0.650
	MTurk-771	0.540	0.609	0.640	0.755	**0.778**	0.773
	RG-65	0.617	0.814	0.845	0.895	**0.915**	0.913
	WS-353-REL	0.548	0.579	0.605	**0.671**	0.658	0.660
	YP-130	0.257	0.528	0.610	0.756	**0.780**	0.771
Similarity	SimLex999	0.338	0.444	0.476	0.735	0.742	**0.752**
	SimVerb-3500	0.190	0.379	0.425	0.732	0.732	**0.742**
	WS-353-SIM	0 .679	0.696	0.728	**0.813**	0.811	0.811
Rare Words	Card-660	0.234	0.388	0.513	0.667	0.657	**0.668**
	RW-STANFORD	0.398	0.476	0.482	0.549	0.552	**0.558**
Mixed Words	MC-30	0.682	0.748	0.738	0.867	0.898	**0.900**
	WS-353-ALL	0.626	0.682	0.699	**0.747**	0.736	0.734
	Average	0.416	0.529	0.558	0.712	0.717	**0.721**

Table 1: Evaluation of HG2Vec versus baseline methods for similarity and relatedness benchmarks. Spearman rank correlation values shown for each benchmark, as well as a weighted average across datasets. Best results in bold.

Dict2Vec, and DRG2Vec on all the tests. Among the baselines, DRG2Vec typically outperforms others except for mixed words datasets. Our HG2Vec model improves the performance on all the relatedness tests. For example, compared to DGR2Vec, on MTurk-771 it has 20.8% higher performance, and on RG-65 8.0% higher. We have a considerable improvement on similarity tests, such as SimLex999 (57.9% higher) and SimVerb-3500 (74.6% higher). Likewise, we see substantial improvements on rare words. Looking at average correlation scores, HG2Vec outperforms DGR2Vec by 29.2%, Dict2Vec by 36.3%, and Word2Vec by 73.3%.

These improvements show that dictionaries and thesauri are suitable replacements for a large corpus for training word embeddings since our model does not make use of any text corpus. Furthermore, since Dict2Vec also uses dictionaries as a resource, the improvement in similarity benchmarks indicates that thesauri are a powerful resource for learning the similarity of word embeddings. Other models only use negative sampling to decrease the similarity score between dissimilar word pairs, which can contain high uncertainty. In contrast, thesauri contain both synonyms and antonyms which are more reliable sources for our model to learn the word pairs with similar and opposite meanings.

Corpus Resource HG2Vec(wiki) includes both the Wikipedia corpus and the dictionaries and thesauri as the input resource. Compared with HG2Vec, HG2Vec(wiki) has better performance on MTurk-287 by 2.4 % and WS-353-REL by 1.7 %, which are both relatedness datasets. In contrast, HG2Vec has 2.3 % improvement on SimLex999 and 1.4 % improvement on SimVerb-3500. It shows that combining Wikipedia corpus can increase the performance of relatedness tests but can also decrease the performance on similarity. However, since incorporating the Wikipedia corpus does not lead to a significant improvement, and the data pre-processing and training time are not negligible, we choose to discard them from input resources.

Context-focused versus Target-only loss Compared with the target-only model HG2Vec(target), our context-focused model HG2Vec increases the performance for similarity benchmarks while keeping most of the performance for relatedness benchmarks. Context-focused model has 1.3% improvement on SimLex999 and SimVerb-3500. Furthermore, the context-focused model increases the performance on benchmarks with rare words. It has a 1.7% improvement on Card-660. Target-only model only has a smaller improvement in relatedness tests, such as a 1.2% improvement in YP-130.

5.2 Ablation Tests

We conduct ablation tests to examine the contribution of each type of edge to the final Spearman correlation, as shown in Table 2. We can see that synonym and negative pairs contribute to both relatedness and similarity tasks. Strong pairs help the models learn relatedness but negatively im-

Type	Benchmark	-strong	-weak	-synonym	-antonym	-negative	HG2Vec
Relatedness	MEN-TR-3K	0.762	0.688	**0.788**	0.782	0.426	0.779
	MTurk-287	0.610	0.517	0.642	**0.677**	0.318	0.650
	MTurk-771	0.755	0.707	0.756	0.770	0.562	**0.773**
	RG-65	0.902	0.826	0.899	**0.913**	0.715	**0.913**
	WS-353-REL	0.642	0.542	0.641	0.650	0.359	**0.660**
	YP-130	0.754	0.756	0.727	**0.815**	0.660	0.771
Similarity	SimLex999	**0.771**	0.725	0.636	0.711	0.644	0.752
	SimVerb-3500	**0.755**	0.715	0.626	0.694	0.671	0.742
	WS-353-SIM	0.794	0.748	0.784	**0.816**	0.591	0.811
Rare Words	Card-660	**0.683**	0.578	0.589	0.682	0.169	0.668
	RW-STANFORD	0.553	0.555	0.504	0.569	0.352	**0.558**
Mixed Words	MC-30	0.898	0.826	0.888	**0.901**	0.707	0.900
	WS-353-ALL	0.720	0.654	0.720	0.732	0.499	**0.734**
	Average	0.718	0.679	0.665	0.707	0.515	**0.721**

Table 2: Ablation study: Column indicates the omitted word pairs; HW2Vec is the full model. Best results in bold.

keyword	DRG2Vec			HG2Vec		
Adjective Query Words						
red	reds	yellow	blue	carmine	crimson	vermillion
	purple	green	orange	scarlet	fuchsia	ponceau
local	locals	regional	district	regional	parochial	locals
	mudir	municipal	stations	subregional	provincial	endemic
Verb Query Words						
sleep	sleeps	sleeping	awake	doze	snooze	shuteye
	asleep	quiescent	waking	slumber	catnap	siesta
listen	call	attentively	hear	listened	listeners	relisten
	listening	listened	callers	listenin	heed	listener
Noun Query Words						
animal	animals	plant	ruminant	critter	varmint	brute
	person	pets	organism	beast	mammal	coyote
coffee	tea	grape	cocoa	coffees	coffea	canephora
	beverage	beans	drinks	tea	arabica	coffeebeans

Table 3: Qualitative Evaluation: Top 6 similar words to the query word.

pact some similarity tasks, such as SimLex999 and SimVerb-3500 (the model improves by removing strong pairs for these datasets). For some similarity datasets, omitting synonyms can sometimes help (e.g., MEN-TR-3K). Likewise, for antonyms (on YP-130). Overall, HG2Vec balances between relatedness and similarity tasks to yield the highest performance.

5.3 Case Study

To qualitatively evaluate the word embeddings, we divide the words into three types: adjectives, verbs, and nouns. We sample query words from each category and report the top six words with the highest cosine similarity to the query word in Table 3.

We observe that for the adjective query *red*, HG2Vec word pairs contain several of its synonyms, such as *vermillion* and *scarlet*. DRG2Vec lists other colors that co-occur in similar contexts, like *blue* and *green*. Likewise, for the adjective *local*, HW2Vec returns high similarity words, but DRG2Vec generates nouns that frequently combine with *local*, such as *stations*. In general, HG2Vec outperforms other models since it can generate more synonyms of the input word.

Comparing verb and noun queries is somewhat harder, but overall, HG2Vec focuses more on semantics, whereas DRG2Vec pays more attention

to co-occurrence. For example, DRG2Vec outputs *awake* as similar to the query verb *sleep*, but they are in fact opposite in meaning. On the other hand, the top results for HG2Vec are all similar to *sleep*. Likewise, for the noun query *animal*, DRG2Vec returns *person*, and for *coffee* it returns *grape*, as one of the top similarity words. HG2Vec avoids such obviously non-similar words. This suggests that models can struggle to learn opposite word pairs without thesauri if their contexts are similar. HG2Vec performs better because of the synonym and antonym edges.

6 Conclusions

We propose HG2Vec, a model that solely relies on dictionaries and thesauri without the use of a huge textual corpus. Despite this, our model yields state-of-art on many similarity and relatedness benchmarks. We also provide strong evidence (via the ablation study) that learning synonyms and antonyms is necessary for improved word embedding models. Our work highlights the use of dictionaries and thesauri as effective sources to learn word embeddings.

One interesting finding is that when we parse the thesaurus, some word pairs are both synonyms and antonyms. For example, according to Roget's Super Thesaurus, the synonyms of *advance* are *help, hasten, quicken, aid, assist, further, forward, facilitate, back*, and the antonyms of *back* are *progress, advance*. The reason *back* is a synonym is because *to advance someone* is *to back them*. On the other hand, *back* is also an antonym since *to go back* is the opposite of *to advance*. Thus, *advance* and *back* are both synonyms and antonyms, suggesting that the relationship between two words may be opposite in different contexts and situations. This is a challenging topic for training word embeddings, which we plan to investigate in the future.

In the future, we plan to also include more relationship types, such as derivative words and participles. Furthermore, while HG2Vec uses definitions only to construct strong and weak pairs, we plan to study ways of using the full definition text in addition to the heterogeneous graph.

References

Tom Bosc and Pascal Vincent. 2018. Auto-encoding dictionary definitions into consistent word embeddings. In *EMNLP*, pages 1522–1532.

Elia Bruni, Nam-Khanh Tran, and Marco Baroni. 2014. Multimodal distributional semantics. *Journal of artificial intelligence research*, 49:1–47.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*.

Manaal Faruqui, Jesse Dodge, Sujay K Jauhar, Chris Dyer, Eduard Hovy, and Noah A Smith. 2014. Retrofitting word vectors to semantic lexicons. *arXiv preprint arXiv:1411.4166*.

Lev Finkelstein, Evgeniy Gabrilovich, Yossi Matias, Ehud Rivlin, Zach Solan, Gadi Wolfman, and Eytan Ruppin. 2001. Placing search in context: The concept revisited. In *Proceedings of the 10th international conference on World Wide Web*, pages 406–414.

Daniela Gerz, Ivan Vulić, Felix Hill, Roi Reichart, and Anna Korhonen. 2016. Simverb-3500: A large-scale evaluation set of verb similarity. *arXiv preprint arXiv:1608.00869*.

Aditya Grover and Jure Leskovec. 2016. node2vec: Scalable feature learning for networks. In *Proceedings of the 22nd ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 855–864.

Guy Halawi, Gideon Dror, Evgeniy Gabrilovich, and Yehuda Koren. 2012. Large-scale learning of word relatedness with constraints. In *Proceedings of the 18th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 1406–1414.

Felix Hill, Roi Reichart, and Anna Korhonen. 2015. Simlex-999: Evaluating semantic models with (genuine) similarity estimation. *Computational Linguistics*, 41(4):665–695.

Abhik Jana, Siddhant Haldar, and Pawan Goyal. 2022. Network embeddings from distributional thesauri for improving static word representations. *Expert Systems with Applications*, 187:115868.

Douwe Kiela, Felix Hill, Stephen Clark, et al. 2015. Specializing word embeddings for similarity or relatedness. In *EMNLP*, pages 2044–2048. Citeseer.

Divesh R Kubal and Anant V Nimkar. 2019. A survey on word embedding techniques and semantic similarity for paraphrase identification. *International Journal of Computational Systems Engineering*, 5(1):36–52.

Omer Levy and Yoav Goldberg. 2014. Neural word embedding as implicit matrix factorization. *Advances in neural information processing systems*, 27.

Minh-Thang Luong, Richard Socher, and Christopher D Manning. 2013. Better word representations with recursive neural networks for morphology. In *Proceedings of the seventeenth conference on computational natural language learning*, pages 104–113.

Marc McCutcheon. 2010. *Roget's Super Thesaurus*. Writer's Digest Books.

Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013. Efficient estimation of word representations in vector space. *arXiv preprint arXiv:1301.3781*.

George A Miller and Walter G Charles. 1991. Contextual correlates of semantic similarity. *Language and cognitive processes*, 6(1):1–28.

Kim Anh Nguyen, Sabine Schulte im Walde, and Ngoc Thang Vu. 2016. Integrating distributional lexical contrast into word embeddings for antonym-synonym distinction. *arXiv preprint arXiv:1605.07766*.

Mohammad Taher Pilehvar, Dimitri Kartsaklis, Victor Prokhorov, and Nigel Collier. 2018. Card-660: Cambridge rare word dataset-a reliable benchmark for infrequent word representation models. *arXiv preprint arXiv:1808.09308*.

Kira Radinsky, Eugene Agichtein, Evgeniy Gabrilovich, and Shaul Markovitch. 2011. A word at a time: computing word relatedness using temporal semantic analysis. In *Proceedings of the 20th international conference on World wide web*, pages 337–346.

Herbert Rubenstein and John B Goodenough. 1965. Contextual correlates of synonymy. *Communications of the ACM*, 8(10):627–633.

Elena Sofia Ruzzetti, Leonardo Ranaldi, Michele Mastromattei, Francesca Fallucchi, and Fabio Massimo Zanzotto. 2021. Lacking the embedding of a word? look it up into a traditional dictionary. *arXiv preprint arXiv:2109.11763*.

Xiaobo Shu, Bowen Yu, Zhenyu Zhang, and Tingwen Liu. 2020. Drg2vec: Learning word representations from definition relational graph. In *2020 International Joint Conference on Neural Networks (IJCNN)*, pages 1–9. IEEE.

Julien Tissier, Christophe Gravier, and Amaury Habrard. 2017. Dict2vec: Learning word embeddings using lexical dictionaries. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 254–263.

Dongqiang Yang and David MW Powers. 2006. *Verb similarity on the taxonomy of WordNet*. Masaryk University.

Wenhao Yu, Chenguang Zhu, Yuwei Fang, Donghan Yu, Shuohang Wang, Yichong Xu, Michael Zeng, and Meng Jiang. 2021. Dict-bert: Enhancing language model pre-training with dictionary. *arXiv preprint arXiv:2110.06490*.

Juexiao Zhang, Yubei Chen, Brian Cheung, and Bruno A Olshausen. 2019. Word embedding visualization via dictionary learning. *arXiv preprint arXiv:1910.03833*.

Transferring Knowledge from Structure-aware Self-attention Language Model to Sequence-to-Sequence Semantic Parsing

Ran Ji and **Jianmin Ji***
University of Science and Technology of China, Hefei, China
jiran@mail.ustc.edu.cn, jianmin@ustc.edu.cn

Abstract

Semantic parsing considers the task of mapping a natural language sentence into a target formal representation, where various sophisticated sequence-to-sequence (seq2seq) models have been applied with promising results. Generally, these target representations follow a syntax formalism that limits permitted forms. However, it is neither easy nor flexible to explicitly integrate this syntax formalism into a neural seq2seq model. In this paper, we present a structure-aware self-attention language model to capture structural information of target representations and propose a knowledge distillation based approach to incorporating the target language model into a seq2seq model, where grammar rules or sketches are not required in the training process. An ablation study shows that the proposed language model can notably improve the performance of the baseline model. The experiments show that our method achieves new state-of-the-art performance among neural approaches on four semantic parsing (ATIS, GEO) and Python code generation (Django, CoNaLa) tasks.

1 Introduction

Semantic parsing aims to map a natural language sentence into a machine executable formal representation, which has been considered as one of the prime challenges nowadays in natural language processing (NLP). These target formal representations can generally be divided into three categories (Kamath and Das, 2018), i.e., logical forms, like first order sentences or λ-calculus expressions (Zettlemoyer and Collins, 2005), programming language statements, like Python code or SQL programs, and graph-based forms, like labeled graphs in Abstract Meaning Representation (AMR) (Banarescu et al., 2013). In this paper, we focus on semantic parsing that yields logical forms.

Target logical forms often follow a syntax formalism that limits permitted formulas, which can be used to filter the output and improve the performance of semantic parsing. For example, in the pre-neural era, CCG based approaches (Kwiatkowski et al., 2013) achieved significant performance gains by introducing a linguistically motivated grammar induction scheme. Some neural semantic parsers (Yin and Neubig, 2018; Sun et al., 2020) first transduce the natural language utterance into an Abstract Syntax Tree (AST), then serve it as an intermediate meaning representation to incorporate with grammar rules for the target logical form. Semantic parsing can also be considered as a seq2seq transduction problem, where the decoder can leverage structural features of target representations. In particular, hierarchical tree decoders are applied in (Dong and Lapata, 2016; Alvarez-Melis and Jaakkola, 2017; Sun et al., 2019) to take into account the tree structure of the logical expression. Decoders constrained by a grammar model are applied in (Xiao et al., 2016; Yin and Neubig, 2017; Krishnamurthy et al., 2017; Dong and Lapata, 2018). The uncertainty-driving adaptive decoding is used to guide the decoder in (Zhang et al., 2019). Relatively sizeable monolingual corpus of the target programming language is used in (Norouzi et al., 2021) to improve performance.

Note that, manually specified grammar rules and sketches for target logical forms are required in most of these approaches, which limits their adaptabilities and scalabilities to a new semantic parsing task with updated target logical forms. In this paper, we consider using a structure-aware language model to capture formal patterns for target representations and incorporating the language model into seq2seq models for semantic parsing.

We first train the structure-aware language model on target logical forms to capture structural information. Then, we incorporate the language model to a seq2seq model for semantic parsing.

*The corresponding author.

Proceedings of the 29th International Conference on Computational Linguistics, pages 3164–3174
October 12–17, 2022.

Integrating a language model into a seq2seq model has been considered in automatic speech recognition (ASR) and neural machine translation (NMT). In particular, shallow fusion and deep fusion (Gulcehre et al., 2015) are two such approaches in NMT. Cold fusion (Sriram et al., 2018) is tested on ASR tasks. Bai et al. (2019) proposes a knowledge distillation based training approach to transferring knowledge from a language model to a seq2seq model for ASR. Here, we follow the knowledge distillation structure to integrate the language model to the baseline seq2seq model for semantic parsing.

We evaluate our approach on two semantic parsing datasets, ATIS (Dahl et al., 1994) and GEO (Zelle and Mooney, 1996) datasets, where target logical forms are λ-calculus expressions and two code generation tasks, Django(Oda et al., 2015) and CoNaLa(Yin et al., 2018), where target logical forms are Python code. We train the target language model based on target logical forms. The experimental results show that our approach achieves state-of-the-art performance among neural network based approaches on ATIS, GEO, Django and CoNaLa datasets.

In this paper, we show that the proposed language model can be used to capture structural features of target logical forms and the knowledge distillation structure can be used to transfer knowledge to a seq2seq model for semantic parsing, where manually specified grammar rules or sketches are no longer required. Notice that, this approach can be applied to various sophisticated seq2seq models, which results a more flexible and scalable method for neural semantic parsers to leverage structural features of target representations. The main contributions of the paper are summarized as follows:

- We propose a structure-aware self-attention language model to capture structural information of target logical forms.
- We propose a knowledge distillation structure to transfer knowledge from target language model to a seq2seq model, which suggests a more flexible and scalable method for neural semantic parsers to leverage structural features of target representations.
- We implement the approach on baseline seq2seq models, which achieves new state-of-the-art performance among neural semantic parsers on ATIS, GEO, Django and CoNaLa datasets.

2 Related Work

2.1 Neural Semantic Parsing

Neural semantic parsing has achieved promising results in recent years. In particular, AST based parsers (Yin and Neubig, 2018; Sun et al., 2020, 2019) first map a nature language sentence into an abstract syntax tree (AST), then parse the AST to the corresponding target logic form. On the other hand, seq2seq based semantic parsers often leverage structural features of natural language sentences or target representations to improve the performance. Specifically, a sequence-to-tree (seq2tree) model (Dong and Lapata, 2016) updates the decoder into a hierarchical LSTM tree, which helps the model to utilize the hierarchical structure of logical forms. A graph-to-sequence (graph2seq) model (Xu et al., 2018) updates the encoder into a graph encoder. Graph neural networks (GNNs) are also used in semantic parsing (Shaw et al., 2019) to incorporate information about relevant entities and their relations during the parsing. A sequence-to-action (seq2action) model (Chen et al., 2018) considers semantic parsing as an end-to-end semantic graph generation process. A coarse-to-fine (coarse2fine) model (Dong and Lapata, 2018) decomposes the decoding process into two stages. The first stage predicates a rough sketch of the meaning representation and the second stage fills in missing details conditioning on the natural language input and the sketch itself. The AdaNSP model (Zhang et al., 2019) proposes an adaptive decoding method to avoid intermediate representations in the parsing process, where the decoder is guided by the model uncertainty. TAE (Norouzi et al., 2021) exploit a relatively sizeable monolingual corpus of the target programming language to improve performance.

Notice that, manually specified grammar rules or sketches are required in most of these neural semantic parsing approaches to leverage structural features of natural language sentences or target representations. In this paper, we consider using the proposed target language model to capture these formal patterns and incorporating the language model into seq2seq models for semantic parsing.

2.2 Structural Language Models

In recent years, language models that capture structural information in natural language have been developed. (Shen et al., 2018) proposed a PRPN(Parsing-Reading-Predict Networks) model, which uses the syntactic structure information of natural language to better perform language modeling. The model is divided into three parts: parsing module, reading module and prediction module. The parsing module uses the convolutional neural network to predict the syntactic distance of two adjacent words, and obtains the syntactic tree of the sentence through the syntactic distance; the reading module uses the syntactic tree obtained by the parsing module to model the context; the prediction module predicts the next word. The PRPN model achieved good results at the time on both unsupervised syntactic analysis tasks and language model modeling.

(Shen et al., 2019) proposed the ON-LSTM (Ordered Neurons-LSTM) model, which gives LSTM neuron level information to model the hierarchical structure information of sentences. The author believes that the level of a word is related to its span in a sentence. The higher the level, the larger the span, so words with higher levels should be retained longer and are not easily updated. So the model proposes a new LSTM neuron: the ordered neuron, which enforces the order in which the neurons are updated. All lower-order neurons must be deleted before higher-order neurons can be deleted or updated, thus controlling the update frequency of neurons. The ON-LSTM model achieves good performance on four different tasks, language modeling, unsupervised parsing, target grammar evaluation, and logical reasoning.

(Wang et al., 2019) proposed the Tree Transformer model, which improved the Transformer to learn syntactic information in natural language. The Tree Transformer adds an additional constraint of "Constituent Attention" to the attention head of the Transformer's encoder to enhance attention to natural language tree structures. The component constraint module judges whether two adjacent words can form a phrase, and if so, assigns more attention scores to these two words. Tree Transformer is designed for natural language parsing tasks and has achieved good results on unsupervised parsing tasks. In addition to the syntactic analysis task, the author also changed the Tree Transofmer into a mask language model, and compared it with BERT on the corpus WSJ. Since the syntactic information can be learned in the Tree-Transformer, the effect of the language model is better than that of BERT.

(Li et al., 2021) proposed a StructuralLM model to improve BERT to learn structural information in documents. StructuralLM treats each cell in the document as a semantic unit, and then makes the model's training goal to classify the cell location to take full advantage of the cell and layout information. The pre-trained StructuralLM model achieved state-of-the-art results on three downstream tasks: form understanding, document visual question answering, and document image classification.

In this paper, we propose the structure-aware language model that use structure-aware self-attention to explicitly capture the structural information of the target forms.

2.3 Integrating Language Model into Seq2Seq Models

Integrating a language model into a seq2seq model has been considered in multiple NLP tasks, like automatic speech recognition (ASR) and neural machine translation (NMT). In particular, shallow fusion and deep fusion (Gulcehre et al., 2015) are proposed to integrate a language model into a seq2seq model. Both methods first train a language model and a translation model separately, then use the language model in the inference step. Specifically, shallow fusion performs a log-linear interpolation between the decoder and the language model to re-weight the translation model's scores during the beam search. Deep fusion concatenates the language model and decoder's hidden states next to each other, then uses the the hidden states to fine-tune the model. Cold fusion (Sriram et al., 2018) is tested on AST tasks. Cold fusion uses the logic outputs of the trained language model as features to train the translation model. Simple fusion (Stahlberg et al., 2018) uses the output of a trained language model together with the output of a translation model to train the translation model. Component fusion (Shan et al., 2019) first trains a source language model, later freezes the source language model and trains the translation model, then replaces the source language model with a target language model in the inference process.

The LST (Learning Spelling from Teachers) approach (Bai et al., 2019) proposes a knowledge distillation based training approach to transferring

knowledge from a language model to a seq2seq model for ASR. It first trains a recurrent neural network based language model (RNNLM) on large scale external text, then considers the RNNLM as the teacher to generate soft labels of speech transcriptions to train the decoder in the seq2seq model.

In this paper, we follow the knowledge distillation structure to transfer knowledge from target language model to the decoder of a baseline seq2seq model for semantic parsing. Different from LST, a new Transformer-based structure-aware language model is considered here, which can capture structural information of formal patterns for target representations. We show that the approach achieves new state-of-the-art performance on ATIS, GEO, Django and CoNaLa datasets.

3 Preliminaries

3.1 A Seq2Seq Model for Semantic Parsing

The training procedure of a baseline seq2seq model for semantic parsing is illustrated in Figure 1. The parsing model maps natural language sentences into target expression. The training procedure of a basic seq2seq parsing model is illustrated in Figure 1.

First, a natural language sentence is preprocessed into a sequence of word indexes $\mathbf{x} = \{x_1, \ldots, x_m\}$ and the labeled logical form is preprocessed into a sequence of word indexes $\mathbf{y}^* = \{y_1^*, \ldots, y_n^*\}$. Then, the encoder network produces the sequence $\mathbf{x} = \{x_1, \ldots, x_m\}$ into a high level contextual representation $\mathbf{h} = \{h_1, \ldots, h_m\}$. Later, the decoder network generates the target output $\mathbf{y} = \{y_1, \ldots, y_n\}$ from $\mathbf{h}$.

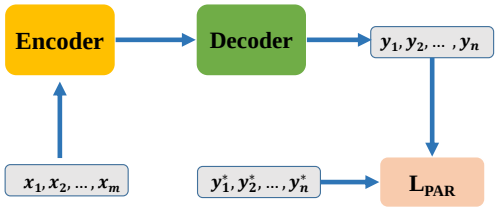

Figure 1: A basic seq2seq model for semantic parsing.

At time step t, current token y_t is generated by the following equation:

$$P_{PAR}(y_t) = p(y_t \mid y_{<t}, \mathbf{x}), \tag{1}$$

where $y_{<t} = y_1 \ldots y_{t-1}$, $\mathbf{x}$ represents the input word indexes.

The training criterion is cross entropy:

$$L_{LM} = -\sum_{i=1}^{N}\sum_{k=1}^{|V|} \mathbf{1}\{y_t = k\} \log P_{LM}(y_t = k) \tag{2}$$

where P_{PAR} is computed from Equation1, T is the length of the target sequence, $|V|$ is the size of the vocabulary, $\mathbf{1}$ is the indicator function.

3.2 Self-Attention

The multi-head self-attention module is a key component in Transformer (Vaswani et al., 2017). In particular, transformer's sub-layers employ h attention heads to perform self-attention. The results from each attention heads are concatenated and transformed to form the output of the sub-layer.

Given a sequence $x = (x_1, \ldots, x_n)$ as input, each attention head uses scaled dot-product attention to compute a new sequence $z = (z_1, \ldots, z_n)$ of the same length, i.e.,

$$z_i = \sum_{j=1}^{n} \alpha_{ij} \left(x_j W^V\right), \tag{3}$$

where W^V is a matrix of parameters and α_{ij} are normalized by a softmax function, i.e.,

$$\alpha_{ij} = \frac{\exp(e_{ij})}{\sum_{k=1}^{n} \exp(e_{ik})}, \tag{4}$$

where e_{ij} is computed using a compatibility function that compares two input elements, i.e.,

$$e_{ij} = \frac{\left(x_i W^Q\right)\left(x_j W^K\right)^\top}{\sqrt{d_z}}, \tag{5}$$

where W^Q, W^K are parameters to be learned.

4 Method

In this section, we specify details of our method, i.e., using a knowledge distillation based structure to transfer knowledge from a structure-aware target language model to a seq2seq model. We first introduce the architecture of the new model. Then, we describe the proposed target language model. At last, we provide details of the method in the training process.

4.1 Model Overview

An overview of the new model's architecture is shown in Figure 2. Note that, the new model is

generated from the basic seq2seq model in Figure 1 by introducing a knowledge distillation structure where the pretrained structure-aware language model serves as the teacher to guide the parsing model.

In specific, the structure-aware language model is pre-trained on target logical forms. The language model contains a structure-aware self-attention transformer encoder to explicitly capture the structural information. It is used to provide soft labels as prior knowledge to "teach" the parsing model in the training process, where the Kullback-Leibler divergence between estimated probabilities is intended to be minimized.

Notice that, there is no specific requirement for the seq2seq model in the architecture. Then, besides the basic seq2seq model, this knowledge distillation structure can be applied to other sophisticated seq2seq models to leverage structural features of target representations.

4.2 Target Language Model

Here we specify details of the proposed target language model, i.e., structure-aware self-attention language model. Architecture of the language model is shown in Figure 3.

Since the target logical forms can all be seen as bracket trees, they're tree-structured. Self attention in Transformer learns how much attention to put on words in a sequence, but it ignores the syntactic information of trees. The siblings of tree nodes may have long distance in a sequence position, but they're related closely. Therefore, we propose structure-aware self-attention to encode the depth information of sibling nodes into self-attention to capture this information.

Motivated by (Shaw et al., 2018), we extend the self-attention architecture to explicitly encode the relation between an element pair (x_i, x_j) by modifying Equation (5) to

$$e_{ij} = \frac{x_i W^Q \left(x_j W^K + a_{ij}^K\right)^\top}{\sqrt{d_z}}. \tag{6}$$

Different from (Shaw et al., 2018), we redefine the relation representations a_{ij}.

We assume that the depth information is less useful when it is too deep. We define the maximum s as a constant k:

$$\begin{aligned} a_{ij}^K &= w^K_{\mathrm{clip}(s(i,j),k)} \\ \mathrm{clip}(x,k) &= \min(x,k) \end{aligned} \tag{7}$$

where $s(i,j)$ is defined as follows:

$$s(i,j) = \begin{cases} dep(i), & father(i) = father(j), \\ 0, & otherwise, \end{cases} \tag{8}$$

where $dep(i)$ is the depth of node i in a tree, $father(i)$ means the father of node i.

Figure 4 shows an example we chose in GEO dataset for demonstration.

We replace the original self-attention architecture of transformer encoder with our structure-aware self-attention. The encoder is bidirectional, so we add the subsequent mask (originally applied in the transformer decoder) to it to specify it as a language model. The subsequent mask creates a lower triangular matrix where the elements above the diagonal will be modified to zero and the elements below the diagonal will be set to whatever the input tensor is. Therefore, the prediction for position i will depend only on the known outputs at positions less than i.

The generation of the language model is determined by:

$$P_{LM}(y_t) = p(y_t \mid y_{<t}). \tag{9}$$

In our experiments, the language model is trained based on λ-calculus expressions and python codes appeared in the training sets of the ATIS, GEO, Django and CoNaLa datasets respectively. The training objective of the language model is to minimize the cross-entropy with target expressions:

$$L_{LM} = -\sum_{i=1}^{N}\sum_{k=1}^{|V|} \mathbf{1}\left\{y_t = k\right\} \log P_{LM}\left(y_t = k\right) \tag{10}$$

where N is the length of the target sequence, L_{LM} denote the training objective functions for the language model, P_{LM} is computed by Equation (9) respectively.

Given a sequence of preprocessed logic form indexes $\mathbf{y}^* = \{y_0^*, \ldots, y_{n-1}^*\}$ obtained from a labeled logical form (y_0^* is the start symbol, y_n^* is the end symbol), the language model produce likelihoods of the target distribution as soft labels, i.e., it generates $\mathbf{y}^S = \{y_1^S, \ldots, y_n^S\}$.

4.3 Training

In the training process, we need to combine the loss from the seq2seq model, L_{PAR}, and the loss from knowledge distillation, L_{KD}.

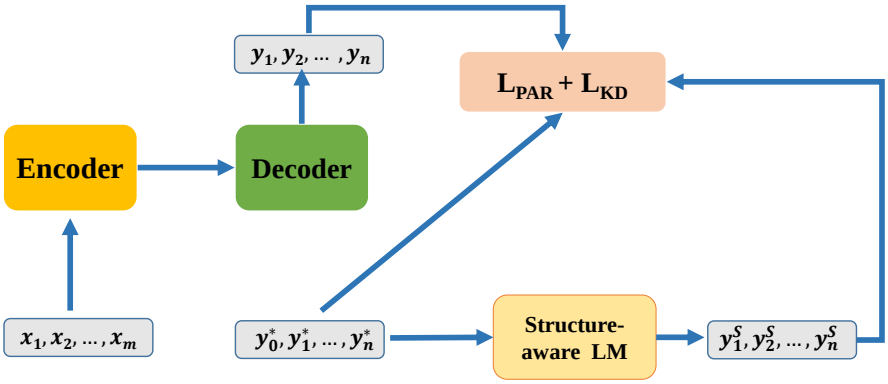

Figure 2: An overview of the proposed model's architecture.

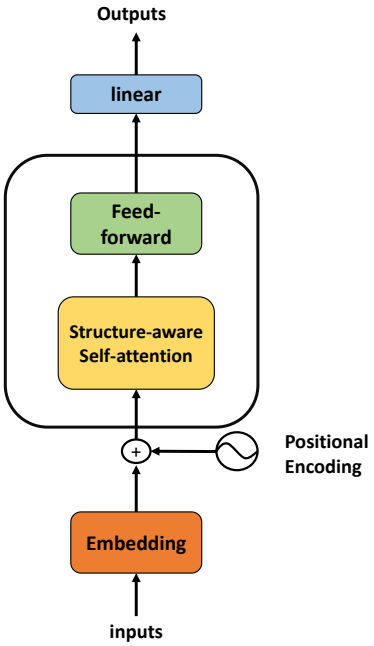

Figure 3: An overview of structure-aware language model's architecture.

In specific, to make the seq2seq model learn the knowledge from the language model, we put target sequences into the language model to get estimated probabilities, then we minimize the Kullback-Leibler (KL) divergence between output of the language model and output of the decoder. The loss from knowledge distillation is:

$$L_{KD} = -\sum_{i=t}^{T}\sum_{k=1}^{|V|} KL\big(P_{PAR}(y_t = k), P_{LM}(y_t = k)\big) \tag{11}$$

where P_{LM} denotes the output of the language model computed by by Equation (9) and the function KL computes the KL divergence.

At last, the loss for the entire model is the combination:

$$L = \eta L_{PAR} + (1 - \eta) L_{KD} \tag{12}$$

where η is a coefficient between 0 and 1.

5 Experiments

In order to evaluate the performance of our proposed model, we conduct the experiments detailed below.

5.1 Language Modeling

In this section, we evaluate our structure-aware language model on language modeling. We evaluate the performance on language modeling by measuring the perplexity(PPL) of target sentences. We use 31425 lambda statements and 51877 python statements collected from the github website as the language model dataset, and follows the ratio of 8:1:1 to devide the training set, validation set and the test set. It should be noted that the lambda statements and python statements in all test sets in the semantic parsing datasets are removed from the training dataset to prevent any impact.

We reproduce and test open-source structural language models in recent years, and compare them with proposed structure-aware language model, using perplexity as an evaluation indicator. In order to explore the contribution of the structure-aware self-attention module in the language model to the model, we also conduct an ablation experiment that removes the structure-aware self-attention.

Table 1 shows the result of our structure-aware language model and other stuctural language models on language modeling. Compared with other

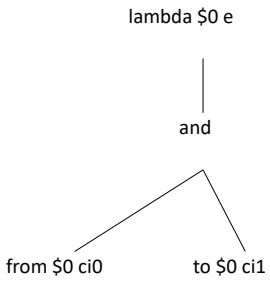

(a)

lambda	$0	e	and	from	$0	ci0	to	$0	ci1
1	1	1	2	3	3	3	3	3	3

(b)

	lambda	$0	e	and	from	$0	ci0	to	$0	ci1
lambda	1	1	1	0	0	0	0	0	0	0
$0	1	1	1	0	0	0	0	0	0	0
e	1	1	1	0	0	0	0	0	0	0
and	0	0	0	2	0	0	0	0	0	0
from	0	0	0	0	3	3	3	0	0	0
$0	0	0	0	0	3	3	3	0	0	0
ci0	0	0	0	0	3	3	3	0	0	0
to	0	0	0	0	0	0	0	3	3	3
$0	0	0	0	0	0	0	0	3	3	3
ci1	0	0	0	0	0	0	0	3	3	3

(c)

Figure 4: (a): An example of the tree structure of a logical form. (b): depth of (a). (c): the structure of (a), which is the input of the structure-aware self-attention.

Model	lambda	python
ON-LSTM (Shen et al., 2019)	67.9	72.4
Tree Transformer (Wang et al., 2019)	58.3	68.5
Ours		
SLM	**46.3**	**52.8**
-structure-aware	60.1	70.9

Table 1: Results of SLM on language modeling.

stuctural language models, our structure-aware language model proposed explicitly obtain the tree structure of target sentences, and explicitly encode the tree structure in the self-attention structure, so that the language model can make full use of the structure information of target sentences, and achieve lower perplexity performance.

5.2 Semantic Parsing

In this section, we evaluate our approach on ATIS, GEO, Django and CoNaLa datasets and compare it with other approaches. We also conduct an ablation study to explore the effectiveness of the proposed structure-aware language model.

We first specify details of our implementation including the datasets, the hyperparameters, hardware, and software for training and testing networks. Then we present the experimental results, which show that our model achieves new state-of-the-art performance among various neural semantic parsers on all four datasets.

5.3 Datasets

We evaluate our approach on four semantic parsing and code generation benchmarks:

ATIS contains natural language questions of a flight dataset paired with a lambda calculus query. We follow the standard train-dev-test split of the datasets in (Zettlemoyer and Collins, 2007), which is 4434/491/448.

GEO contains natural language questions about US geography paired with Prolog database queries. We use the corresponding λ-calculus expressions with the same meaning as in (Kwiatkowski et al., 2011). We follow the standard train-dev-test split of the datasets in (Zettlemoyer and Collins, 2005), which is 600/0/280.

Django contains lines of Python source code extracted from the Django framework paried with an NL description. We follow the standard train-dev-test split of the datasets in (Oda et al., 2015) , which is 16000/1000/1805.

CoNaLa contains mannully annotated NL questions paired with python solution on STACKOVERFLOW. We follow the standard train-dev-test split of the datasets in (Yin et al., 2018), which is 2379/0/500.

Model	ATIS	GEO
ZC07(Zettlemoyer and Collins, 2007)	84.6	86.1
FUBL(Kwiatkowski et al., 2011)	82.8	88.6
KCAZ13(Kwiatkowski et al., 2013)	-	89.0
Neural network models		
Seq2Seq(Dong and Lapata, 2016)	84.2	84.6
Seq2Tree(Dong and Lapata, 2016)	84.6	87.1
JL16(Jia and Liang, 2016)	83.3	89.3
TranX(Yin and Neubig, 2018)	86.2	88.2
Coarse2fine(Dong and Lapata, 2018)	87.7	88.2
Seq2Act(Chen et al., 2018)	87.7	88.2
Graph2Seq(Xu et al., 2018)	85.5	88.9
AdaNSP (Zhang et al., 2019)	88.6	88.9
GNN(Shaw et al., 2019)	87.1	89.3
TreeGen(Sun et al., 2020)	89.1	89.6
PASCAL(Xie et al., 2021)	90.2	90.7
Ours		
Baseline	88.6	88.9
+ SLM KD fusion	**90.4**	**91.1**
- structure-aware	88.8	89.3

Table 2: Results on ATIS and GEO datasets.

Model	Django
TranX(Yin and Neubig, 2018)	73.7
Coarse2fine(Dong and Lapata, 2018)	74.1
TranX2 (Yin and Neubig, 2019)	77.3±0.4
TranX2+BERT	79.7±0.42
Reranker (Yin and Neubig, 2019)	80.2±0.4
TAE (Norouzi et al., 2021)	81.03±0.14
Ours	
Baseline	81.03
+ SLM KD fusion	**81.83**
- structure-aware	81.16

Table 3: Results on Django dataset.

Model	CoNaLa
TranX(Yin and Neubig, 2018)	24.3
Reranker (Yin and Neubig, 2019)	30.11±0.6
EK (Xu et al., 2020)	27.20
EK+100k (Xu et al., 2020)	28.14
EK+100K+API (Xu et al., 2020)	32.26
TAE (Norouzi et al., 2021)	32.57±0.3
Ours	
Baseline	32.57
+ SLM KD fusion	**33.10**
- structure-aware	32.62

Table 4: Results on CoNaLa dataset.

5.4 Implementation Details

We use AdaNSP(Zhang et al., 2019), a competitive seq2seq semantic parsing model built on AllenNLP(Wallace et al., 2019), as our base model for two semantic parsing tasks. The model uses adaptive decoding method that guide the decoder by model uncertainty and automatically uses deeper computations when necessary. The AdaNSP model is not the state-of-the-art model now, but it is based on seq2seq architecture and open-sourced so it is easy to implement our method. We adapt the same hypeparameters as in (Zhang et al., 2019). We use TAE (Norouzi et al., 2021), a seq2seq code generation model as our base model for two code generation tasks. The model exploit a relatively sizeable monolingual corpus of the target programming language to a transformer-based seq2seq model and reach a superior performance.

We trained our model with the hyperparameters listed in Table 5, which was chosen based on the performance of the model on the validation set for ATIS, Django and on the randomly selected training set for GEO, CoNaLa, where the validation set is not provided. For structures of the language model, we set the number of layers 3, positional feed forward dimensions 512, and attention heads 8.We trained the parsing model with the original settings of the baseline system. We trained the language model for 100 epoches respectively, and the entire model for 200 epoches on an Nvidia GeForce RTX 3090 GPU, which takes around 5 hours.

Hyperparameter	Value
learning rate	0.0005
batch size	256
dropout	0.1
η	0.8
k	10

Table 5: Hyperparameters.

5.5 Evaluation

We use logical form accuracy as the evaluation metric for ATIS and GEO datasets, which is computed with pared trees of the predictions and gold logical forms. The order of the children can be changed within conjunction nodes. We use STree parser code from (Dong and Lapata, 2018) to parse the target lambda expressions and predictions into bracket trees and compare them. We use exact match accuracy as the evaluation metric for Django dataset and corpus-level BLEU for CoNaLa.

5.6 Results

We compare our method with state-of-the-art semantic parsers on ATIS, GEO, Django and CoNaLa datasets. Table 2- 4 show the results of our model and existing semantic parsers on four datasets. Our model achieves the state-of-the-art performance on four datasets.

We also performed an ablation study by removing the proposed structure-aware self-attention. In specific, we use an original transformer encoder as the language model and integrate it into the parsing model by knowledge distillation. The results show that the model using the structure-aware language model outperforms the one using only original language model.

6 Conclusion

In this paper, we present a structure-aware self-attention language model to capture structural information of target representations and propose a knowledge distillation based approach to incorporating the target language model into a seq2seq model. We show that using knowledge distillation from a target language model provides a flexible and scalable way for neural semantic parsers to leverage structural features of target representations. Our method achieves strong results

and doesn't need any manually designed rules or sketches.

For future direction, we are interested in exploring other datasets to verify the model's ability for structural data. We will also attempt to integrate grammar rules to this model to have a better performance on semantic parsing tasks.

Acknowledgements

This work is partially supported by the 2030 National Key AI Program of China 2018AAA0100500, Guangdong Province R&D Program 2020B0909050001, Anhui Province Development and Reform Commission 2020 New Energy Vehicle Industry Innovation Development Project and 2021 New Energy and Intelligent Connected Vehicle Innovation Project, Shenzhen Yijiahe Technology R&D Co., Ltd., and Huawei Cloud Computing Technologies Co., Ltd.

References

David Alvarez-Melis and T. Jaakkola. 2017. Tree-structured decoding with doubly-recurrent neural networks. In *Proceedings of the 5th International Conference on Learning Representations (ICLR-17)*.

Ye Bai, Jiangyan Yi, Jianhua Tao, Zhengkun Tian, and Zhengqi Wen. 2019. Learn spelling from teachers: Transferring knowledge from language models to sequence-to-sequence speech recognition. In *Proceedings of the 20th Annual Conference of the International Speech Communication Association (INTERSPEECH-19)*, pages 3795–3799.

Laura Banarescu, Claire Bonial, Shu Cai, Madalina Georgescu, Kira Griffitt, Ulf Hermjakob, Kevin Knight, Philipp Koehn, Martha Palmer, and Nathan Schneider. 2013. Abstract meaning representation for sembanking. In *Proceedings of the 7th linguistic annotation workshop and interoperability with discourse (LAW-13)*, pages 178–186.

Bo Chen, Le Sun, and Xianpei Han. 2018. Sequence-to-action: End-to-end semantic graph generation for semantic parsing. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 766–777, Melbourne, Australia. Association for Computational Linguistics.

D. Dahl, M. Bates, M. Brown, W. Fisher, K. Hunicke-Smith, D. Pallett, C. Pao, Alexander I. Rudnicky, and E. Shriberg. 1994. Expanding the scope of the atis task: The atis-3 corpus. In *Proceedings of the workshop on Human Language Technology (HLT-94)*, pages 43–48.

Li Dong and Mirella Lapata. 2016. Language to logical form with neural attention. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 33–43, Berlin, Germany. Association for Computational Linguistics.

Li Dong and Mirella Lapata. 2018. Coarse-to-fine decoding for neural semantic parsing. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 731–742, Melbourne, Australia. Association for Computational Linguistics.

Caglar Gulcehre, Orhan Firat, Kelvin Xu, Kyunghyun Cho, Loic Barrault, Hueichi Lin, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. 2015. On using monolingual corpora in neural machine translation. *CoRR*, abs/1503.03535.

Robin Jia and Percy Liang. 2016. Data recombination for neural semantic parsing. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (ACL-16)*, pages 12–22.

Aishwarya Kamath and Rajarshi Das. 2018. A survey on semantic parsing. In *Proceedings of the 1st Conference on Automated Knowledge Base Construction (AKBC-19)*.

Jayant Krishnamurthy, Pradeep Dasigi, and Matt Gardner. 2017. Neural semantic parsing with type constraints for semi-structured tables. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 1516–1526, Copenhagen, Denmark. Association for Computational Linguistics.

Tom Kwiatkowski, Eunsol Choi, Yoav Artzi, and Luke Zettlemoyer. 2013. Scaling semantic parsers with on-the-fly ontology matching. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing (EMNLP-13)*, pages 1545–1556.

Tom Kwiatkowski, Luke Zettlemoyer, Sharon Goldwater, and Mark Steedman. 2011. Lexical generalization in ccg grammar induction for semantic parsing. In *Proceedings of the 2011 Conference on Empirical Methods in Natural Language Processing (EMNLP-11)*, pages 1512–1523.

Chenliang Li, Bin Bi, Ming Yan, Wei Wang, Songfang Huang, Fei Huang, and Luo Si. 2021. Structurallm: Structural pre-training for form understanding. *ArXiv*, abs/2105.11210.

Sajad Norouzi, Keyi Tang, and Yanshuai Cao. 2021. Code generation from natural language with less prior knowledge and more monolingual data. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 2: Short Papers)*, pages 776–785, Online. Association for Computational Linguistics.

Yusuke Oda, Hiroyuki Fudaba, Graham Neubig, Hideaki Hata, S. Sakti, T. Toda, and Satoshi Nakamura. 2015. Learning to generate pseudo-code from source code using statistical machine translation (t). *2015 30th IEEE/ACM International Conference on Automated Software Engineering (ASE)*, pages 574–584.

Changhao Shan, Chao Weng, Guangsen Wang, Dan Su, Min Xue Luo, Dong Yu, and Lei Xie. 2019. Component fusion: Learning replaceable language model component for end-to-end speech recognition system. In *Processings of ICASSP 2019 - 2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP-19)*, pages 5361–5635.

Peter Shaw, Philip Massey, Angelica Chen, Francesco Piccinno, and Yasemin Altun. 2019. Generating logical forms from graph representations of text and entities. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 95–106, Florence, Italy. Association for Computational Linguistics.

Peter Shaw, Jakob Uszkoreit, and Ashish Vaswani. 2018. Self-attention with relative position representations. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 2 (Short Papers)*, pages 464–468, New Orleans, Louisiana. Association for Computational Linguistics.

Yikang Shen, Zhouhan Lin, Chin-Wei Huang, and Aaron C. Courville. 2018. Neural language modeling by jointly learning syntax and lexicon. *ArXiv*, abs/1711.02013.

Yikang Shen, Shawn Tan, Alessandro Sordoni, and Aaron C. Courville. 2019. Ordered neurons: Integrating tree structures into recurrent neural networks. *ArXiv*, abs/1810.09536.

Anuroop Sriram, Heewoo Jun, Sanjeev Satheesh, and Adam Coates. 2018. Cold fusion: Training seq2seq models together with language models. In *Proceedings of the 19th Annual Conference of the International Speech Communication Association (INTERSPEECH-18)*, pages 387–391.

Felix Stahlberg, James Cross, and Veselin Stoyanov. 2018. Simple fusion: Return of the language model. In *Proceedings of the 3rd Conference on Machine Translation: Research Papers (WMT-18)*, pages 204–211.

Zeyu Sun, Qihao Zhu, Lili Mou, Yingfei Xiong, Ge Li, and Lu Zhang. 2019. A grammar-based structural cnn decoder for code generation. In *Proceedings of the 33rd AAAI Conference on Artificial Intelligence (AAAI-19)*, pages 7055–7062.

Zeyu Sun, Qihao Zhu, Yingfei Xiong, Yican Sun, Lili Mou, and Lu Zhang. 2020. Treegen: A tree-based transformer architecture for code generation. In *Proceedings of the 34th AAAI Conference on Artificial Intelligence (AAAI-20)*, pages 8984–8991.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in neural information processing systems (NIPS-17)*, pages 5998–6008.

Eric Wallace, Jens Tuyls, Junlin Wang, Sanjay Subramanian, Matt Gardner, and Sameer Singh. 2019. AllenNLP interpret: A framework for explaining predictions of NLP models. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP): System Demonstrations*, pages 7–12, Hong Kong, China. Association for Computational Linguistics.

Yau-Shian Wang, Hung yi Lee, and Yun-Nung (Vivian) Chen. 2019. Tree transformer: Integrating tree structures into self-attention. In *EMNLP*.

Chunyang Xiao, Marc Dymetman, and Claire Gardent. 2016. Sequence-based structured prediction for semantic parsing. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1341–1350, Berlin, Germany. Association for Computational Linguistics.

Defeng Xie, Jianmin Ji, Jiafei Xu, and Ran Ji. 2021. Combining improvements for exploiting dependency trees in neural semantic parsing. *ArXiv*, abs/2112.13179.

Frank F. Xu, Zhengbao Jiang, Pengcheng Yin, Bogdan Vasilescu, and Graham Neubig. 2020. Incorporating external knowledge through pre-training for natural language to code generation. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 6045–6052, Online. Association for Computational Linguistics.

Kun Xu, Lingfei Wu, Zhiguo Wang, Mo Yu, Liwei Chen, and Vadim Sheinin. 2018. Exploiting rich syntactic information for semantic parsing with graph-to-sequence model. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 918–924, Brussels, Belgium. Association for Computational Linguistics.

Pengcheng Yin, Bowen Deng, Edgar Chen, Bogdan Vasilescu, and Graham Neubig. 2018. Learning to mine aligned code and natural language pairs from stack overflow. *2018 IEEE/ACM 15th International Conference on Mining Software Repositories (MSR)*, pages 476–486.

Pengcheng Yin and Graham Neubig. 2017. A syntactic neural model for general-purpose code generation. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1:*

Long Papers), pages 440–450, Vancouver, Canada. Association for Computational Linguistics.

Pengcheng Yin and Graham Neubig. 2018. TRANX: A transition-based neural abstract syntax parser for semantic parsing and code generation. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 7–12, Brussels, Belgium. Association for Computational Linguistics.

Pengcheng Yin and Graham Neubig. 2019. Reranking for neural semantic parsing. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 4553–4559, Florence, Italy. Association for Computational Linguistics.

J. Zelle and R. Mooney. 1996. Learning to parse database queries using inductive logic programming. In *Proceedings of the 13th National Conference on Artificial Intelligence and 8th Innovative Applications of Artificial Intelligence Conference (AAAI-96)*, pages 1050–1055.

Luke Zettlemoyer and Michael Collins. 2007. Online learning of relaxed ccg grammars for parsing to logical form. In *Proceedings of the 2007 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning (EMNLP-CoNLL-07)*, pages 678–687.

Luke S Zettlemoyer and Michael Collins. 2005. Learning to map sentences to logical form: structured classification with probabilistic categorial grammars. In *Proceedings of the Twenty-First Conference on Uncertainty in Artificial Intelligence (UAI-05)*, pages 658–666.

Xiang Zhang, Shizhu He, Kang Liu, and Jun Zhao. 2019. AdaNSP: Uncertainty-driven adaptive decoding in neural semantic parsing. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 4265–4270, Florence, Italy. Association for Computational Linguistics.

Enhancing Contextual Word Representations Using Embedding of Neighboring Entities in Knowledge Graphs

Ryoko Tokuhisa **Keisuke Kawano** **Akihiro Nakamura** **Satoshi Koide**
Toyota Central R&D Labs., Inc.
{tokuhisa, kawano, a-nakamura, koide}@mosk.tytlabs.co.jp

Abstract

Pre-trained language models (PLMs) such as BERT and RoBERTa have dramatically improved the performance of various natural language processing tasks. Although these models are trained on large amounts of raw text, they have no explicit grounding in real-world entities. Knowledge graphs (KGs) are manually annotated with factual knowledge and store the relations between nodes corresponding to entities as labeled edges. This paper proposes a mechanism called *KG-attention*, which integrates the structure of a KG into recent PLM architectures. Unlike the existing PLM+KG integration methods, KG-attention generalizes the embeddings of neighboring entities using the relation embeddings; accordingly, it can handle relations between unconnected entities in the KG. Experimental results demonstrated that our method achieved significant improvements in a relation classification task, an entity typing task, and several language comprehension tasks.

1 Introduction

Pre-trained language models (PLMs) have significantly improved the performance of various natural language processing (NLP) tasks (Devlin et al., 2019; Liu et al., 2019). Although these models are trained on large amounts of raw text, they have no explicit grounding in real-world entities. Figure 1 shows a causal reasoning task on which PLM fails. The performance can be improved by representing factual knowledge as various relations between entities, e.g., "engine is *part of* a car", in the PLM.

Knowledge graphs (KGs) are manually annotated with factual knowledge and store the relations between the nodes corresponding to entities as labeled edges (e.g., *HasA, IsA*, and *PartOf*) (Miller, 1995; Bollacker et al., 2008; Speer et al., 2016). Although the various relations recorded in KGs can potentially improve the performance of PLMs, KGs are *graphs* and structurally unsuitable for direct incorporation into PLMs. Some of the more suitable forms of factual knowledge are pre-trained KG embeddings (Bordes et al., 2013; Sun et al., 2019; Nguyen, 2020; Chao et al., 2021). KG embeddings embed two entities registered in a KG such that the corresponding embedding vectors are at a specific relative position depending on the relation between the entities. The embedding vectors corresponding to the entities are called *KG entity embeddings* and the specific relative positions are called *relation embeddings*. In addition to avoiding the direct use of structural encoding of KGs, KG embeddings can be generalized; that is, entities having a specific relation to a query entity can be estimated even when their relation is not actually recorded in the KGs (Bordes et al., 2013). For example, if there is no *PartOf* connection between `car` and `engine` in the KG, the embedding of `engine` can be estimated from the embedding of `car` and the relation embedding of *PartOf*.

One remaining problem is the incorporation of KG entity embeddings and relation embeddings

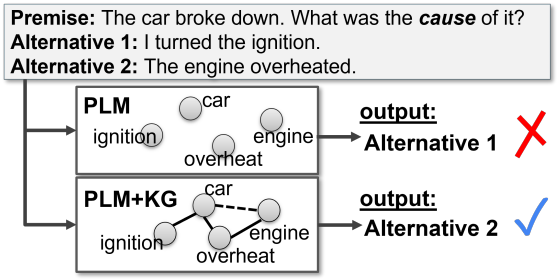

Figure 1: Example of the outputs of Choice of Plausible Alternatives (COPA), a well-known causal reasoning task. "PLM" represents the causal reasoning model using $\text{BERT}_{\text{LARGE}}$ and "PLM+KG" represents the model using $\text{BERT}_{\text{LARGE}}$ and Knowledge Graph (WordNet). Solid and dotted lines indicate the relations between entities connected in KG and entities not connected in KG, respectively. The relation between unconnected entities were estimated from KG embeddings in our proposed method.

Proceedings of the 29th International Conference on Computational Linguistics, pages 3175–3186
October 12–17, 2022.

into PLMs. Recently, several PLM and KG integration methods have been proposed (Zhang et al., 2019; Yang et al., 2019; Lin et al., 2019; Wang et al., 2019; Sun et al., 2020; Wang et al., 2021). For example, Zhang et al. (2019) concatenated embedding vectors obtained from PLM word embeddings and KG entity embeddings for downstream tasks. Wang et al. (2021) jointly trained a masked language model (MLM) and a KG embedding model to align factual knowledge and language representation in the same semantic space. Although KG entity embeddings have been utilized, how the informative relation embeddings should be incorporated into PLMs is not obvious. Actually, the relation embeddings have been ignored, while KG entity embeddings have been well utilized.

Herein, we propose *Integrating PLMs and KGs through Attention Mechanisms (ILKA)*, which handles both the KG entity embeddings and the relation embeddings through our attention mechanism (*KG-attention*). A key feature of KG-attention is that it fully utilizes the generalization ability of pre-trained KG embeddings, meaning that KG-attention can handle relations between entities not connected in the original KG.

Our contributions are summarized below.

- We propose ILKA with KG-attention that incorporates both the KG entity embeddings and relation embeddings with the embedding vectors obtained from PLMs in a consistent manner.
- We experimentally demonstrate that ILKA achieves improvements in a relation classification task, an entity typing task, and several language comprehension tasks.

2 Related Works

Pre-trained language models PLMs such as ELMo (Peters et al., 2018), BERT (Devlin et al., 2019), RoBERTa (Liu et al., 2019), and ALBERT (Lan et al., 2020) have achieved great success in many NLP tasks. As a result, pre-training language models and fine-tuning them in downstream tasks has become a new standard in NLP. However, PLMs do not explicitly learn the relations between entities and may not properly identify factual knowledge (Peters et al., 2019; Wang et al., 2021).

Knowledge graph embedding KGs such as WordNet (Miller, 1995) and ConceptNet (Speer et al., 2016) have also become important resources in many NLP tasks. In general, a KG is a collection of relational facts often represented as a triplet $(h, r, t) \in \mathcal{V} \times \mathcal{R} \times \mathcal{V}$ (e.g., (`car`, `IsA`, `vehicle`)), where $\mathcal{V}$ is the vocabulary and $\mathcal{R}$ is the set of relations. h is a *head entity*, r is a *relation*, and t is a *tail entity*.

	Approximate formula
TransE (Bordes et al., 2013)	$\mathbf{v}(t) \approx \mathbf{v}(h) + \mathbf{v}(r)$
TransR (Lin et al., 2015)	$\mathbf{v}_r(t) \approx \mathbf{v}_r(h) + \mathbf{v}(r)$
RotatE (Sun et al., 2019)	$\mathbf{v}(t) \approx \mathbf{v}(h) \circ \mathbf{v}(r)$
PairRE (Chao et al., 2021)	$\mathbf{v}(t) \approx \mathbf{v}(h) \circ \mathbf{v}^H(r) \oslash \mathbf{v}^T(r)$

Table 1: List of approximate formulas of KG embedding models. $\mathbf{v}_r(\cdot)$ is the embedding function specific to the relation r. $\circ$ is the Hadamard product. $\oslash$ denotes the element-wise division. $\mathbf{v}^H(r)$ and $\mathbf{v}^T(r)$ are embeddings of the relation r for head and tail entities, respectively.

Several learning methods have been proposed for embedding KGs into a continuous vector space while preserving the relational structure (Nickel et al., 2011; Bordes et al., 2013; Schlichtkrull et al., 2018; Lin et al., 2015; Sun et al., 2019). In most KG embedding models, the triplet (h, r, t) is embedded to satisfy the following:

$$\mathbf{v}_{\mathrm{KG}}(t) \approx \phi_h(\mathbf{v}_{\mathrm{KG}}(h), \mathbf{v}_{\mathrm{r}}(r)), \qquad (1)$$

where $\mathbf{v}_{\mathrm{KG}} : \mathcal{V} \mapsto \mathbb{R}^{d_{\mathrm{KG}}}$ is the embedding function, and $\mathbf{v}_{\mathrm{r}}(r) \in \mathbb{R}^{d_{\mathrm{KG}}}$ is the embedding vector corresponding to the relation r. ϕ_h denotes the relationship between the embedding vectors; for example, $\phi_h(\mathbf{v}_{\mathrm{KG}}(h), \mathbf{v}_{\mathrm{r}}(r)) = \mathbf{v}_{\mathrm{KG}}(h) + \mathbf{v}_{\mathrm{r}}(r)$ in **TransE** (Bordes et al., 2013). Table 1 gives the approximation formulas of typical KG embedding models.

One benefit of KG embedding is that entities with a specific relation to the query entity can be retrieved even when the relation between the entities is not connected in the KG (Bordes et al., 2013). For example, in Figure 2, entities e connecting the "IsA" relation with `car` can be estimated by extracting the entities satisfying $\mathbf{v}_{\mathrm{KG}}(e) \approx \phi_h(\mathbf{v}_{\mathrm{KG}}(\mathtt{car}), \mathbf{v}_{\mathrm{r}}(\mathtt{IsA}))$. $\mathbf{v}_{\mathrm{KG}}(e)$ encodes not only words having the "IsA" relation with `car` in the KG (i.e., *automobile*, *vehicle*), but also words not having the "IsA" relation with `car` but having similar meanings to *automobile* and *vehicle* (i.e., *motorcar* in Figure 2). In this paper, the vector $\phi_h(\mathbf{v}_{\mathrm{KG}}(w), \mathbf{v}_{\mathrm{r}}(r))$ is called the ***KG-neighbor embeddings*** of entity w and the en-

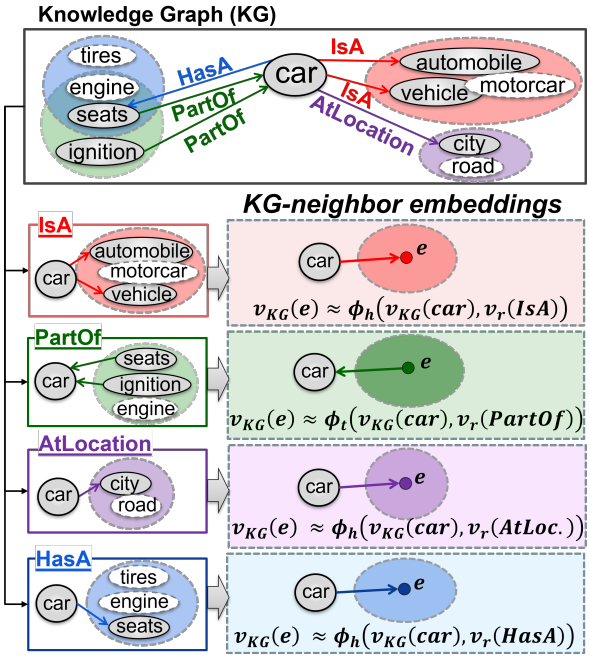

Figure 2: Example of `car` and its neighboring entities in KG. The words in the solid circles (e.g., *automobile*, *city*) have a relation defined in KG whereas the words in the dotted circles (e.g., *motorcar*, *road*) indicate words whose relation is not defined in KG.

tities corresponding to the KG-neighbor embeddings (i.e., e) are called ***KG-neighbors***. Note that KG-neighbors may be unconnected to entity w in the original KG. For some KG embedding methods such as TransE, we can also consider KG-neighbor embeddings as $\phi_t(\mathbf{v}_{\mathrm{KG}}(w), \mathbf{v}_{\mathrm{r}}(r))$, where w corresponds to a tail entity. In Figure 2, the KG-neighbor embeddings in the "PartOf" relation with `car` (i.e., *seats*, *ignition*, *engine*) are represented by $\phi_t(\mathbf{v}_{\mathrm{KG}}(\texttt{car}), \mathbf{v}_{\mathrm{r}}(\texttt{PartOf}))$, such as $\phi_t(\mathbf{v}_{\mathrm{KG}}(\texttt{car}), \mathbf{v}_{\mathrm{r}}(\texttt{PartOf})) = \mathbf{v}_{\mathrm{KG}}(\texttt{car}) - \mathbf{v}_{\mathrm{r}}(\texttt{PartOf})$ in **TransE**. In the following, ϕ refers to either ϕ_h or ϕ_t.

Integrating PLMs and KGs Several recent studies have integrated KGs with PLM. In some studies, the KGs and PLM were combined to perform specific downstream tasks (Yu et al., 2018; Wang and Jiang, 2019; Yasunaga et al., 2021). Yasunaga et al. (2021) proposed an end-to-end question answering (QA) model that leverages PLM and KG by scoring the relevance of KG nodes conditional on a given QA context. These downstream task-focused methods are effective and robust for the given task, but they do not enhance PLM for general purposes.

In another approach, PLM and KG are integrated at the model level (Zhang et al., 2019; Peters et al., 2019; Yang et al., 2019; Lin et al., 2019; Wang et al., 2019; Sun et al., 2020; Wang et al., 2021). Zhang et al. (2019) concatenated PLM word embeddings and KG entity embeddings and Sun et al. (2020) trained an MLM by concatenating the entity sequences obtained from a KG and input word sequences. However, these PLM+KG methods are limited in that they do not effectively employ relation embeddings. A relation embedding encodes the semantic direction of the predefined relations in KGs and plays an important role in representing neighboring entities in KG and generalizing relations between entities not included in the KG. The existing methods either do not consider relation embeddings (Zhang et al., 2019) or cannot handle the relations between entities not connected in the KG (Sun et al., 2020; Wang et al., 2021).

3 Methodology

3.1 Main idea: KG-attention

When KG embedding is integrated into PLMs, the performance can be improved by representing factual knowledge as various relations (e.g., *IsA*, *HasA*, *AtLocation*). Below, we propose ***KG-attention*** that consistently integrates the embedding of entities and relations into an attention mechanism. The benefits of KG-attention are twofold: (1) KG-attention can explicitly assign attention between a query entity and entities in the input sentence that are related through KG embeddings; (2) KG-attention can handle KG-neighbors even when the relation between entities is not connected in the KG.

Figure 3 shows an application example of KG-attention, which explores the attention between an input sentence and KG-neighbors such as car_{IsA}, car_{HasA}, and $car_{\mathrm{AtLocation}}$. For example, the attention between *engine* and car_{HasA} represents the knowledge *"a car has an engine."* When integrating a KG embedding with a PLM, how to incorporate the relation represented by ϕ into a PLM comprising a transformer is not always evident. We solve this difficulty by carefully designing the query, key, and value of the attention. Given an input sequence $\mathbf{w} = [w^{(1)}, \ldots, w^{(L)}]$, where w denotes a subword and L denotes the length of the sequence, KG-attention integrates ϕ into attention mechanism as follows:

$$\mathbf{Q} = \{\mathbf{Q}^{(i)}\}_{i\in\{1,\ldots,L\}}, \quad (2)$$

$$\mathbf{K} = \mathbf{V} = \{\mathbf{K}^{(i,j)}\}_{i\in\{1,\ldots,L\},\, j\in\{0,\ldots,|\mathcal{R}|\}} \quad (3)$$

$$\mathbf{Q}^{(i)} = \mathbf{v}_{\mathrm{KG}}(w^{(i)}) \oplus \mathbf{v}_{\mathrm{PLM}}(w^{(i)}) \quad (4)$$

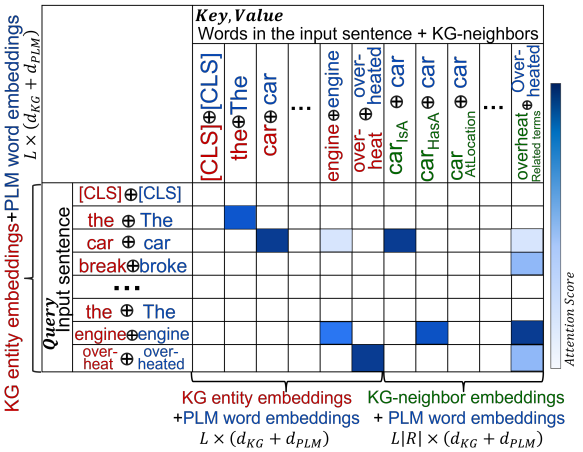

Figure 3: Example of KG-attention applied to sentence classification. The input sentence is "Premise + Alternative 2" in Figure 1.

$$\mathbf{K}^{(i,j)} = \begin{cases} \mathbf{v}_{\mathrm{KG}}(w^{(i)}) \oplus \mathbf{v}_{\mathrm{PLM}}(w^{(i)}) & \text{if } j = 0 \\ \phi(\mathbf{v}_{\mathrm{KG}}(w^{(i)}), \mathbf{v}_{\mathrm{r}}(r^{(j)})) \oplus \mathbf{v}_{\mathrm{PLM}}(w^{(i)}) & \\ \qquad \text{if } j \in \{1, \ldots, |\mathcal{R}|\} \end{cases} \tag{5}$$

where $\mathbf{Q}^{(i)}$ and $\mathbf{K}^{(i,j)}$ are $d_{\mathrm{KG}} + d_{\mathrm{PLM}}$ dimensional vectors, $\mathbf{Q}$ is a set of L vectors, and $\mathbf{K}(= \mathbf{V})$ is a set of $L \times (|\mathcal{R}| + 1)$ vectors. $\mathbf{v}_{\mathrm{KG}}(\cdot) \in \mathbb{R}^{d_{\mathrm{KG}}}$ is the KG entity embedding and $\mathbf{v}_{\mathrm{PLM}}(\cdot) \in \mathbb{R}^{d_{\mathrm{PLM}}}$ is the PLM embeddings. d_{KG} and d_{PLM} are the dimensions of the KG and PLM embeddings, respectively. $\oplus$ is the concatenation operation, and $r^{(j)}$ represents the j-th relation.

Why and how does KG-attention treat semantically related tokens defined in KGs? In general, the attention mechanism seeks key tokens, which are similar to query tokens. Accordingly, the KG-attention aims to seek for key tokens that are located at the KG-neighbors of the query token. For example, $\mathbf{K}^{(i,j)}$ (for $w^{(i)} = car$) in Figure 3 is constructed from KG-neighbor embeddings such as car_{IsA}, car_{HasA}, and $car_{\mathrm{AtLocation}}$. These KG embeddings are concatenated with the PLM embedding of car and then used as the keys and values of the attention. To obtain $\mathbf{Q}^{(i)}$ (for $w^{(i)} = car$), we can simply compute and concatenate the KG and PLM embeddings of car. As KG-neighbor embeddings are obtained from KG entity embeddings of w and $r^{(j)}$ by the approximation function ϕ, they can be regarded as the embeddings of a **representative entity** having relation r with an entity w. If *engine* and the representative entity of car_{HasA} are similar, KG-attention can obtain the attention between them even if *engine* and *car* are unconnected in the original KG. Thus, KG-attention achieves flexible attention by exploiting generalized KG-neighbor embeddings.

Model	PLM +KG	Relation between entities connected	Relation between entities not connected
ERNIE	✓		
KnowBERT	✓	✓	
CoLAKE	✓	✓	
KEPLER	✓	✓	
ILKA (ours)	✓	✓	✓

Table 2: Comparison of ILKA with existing methods. "Relation between entities connected" indicates that the model only handle the relation between entities connected in the KG (e.g., *automobile* in Figure 2); "Relation between entities not connected" indicates that the model handle relations between entities not connected in KG (e.g., *motorcar* in Figure 2).

3.2 ILKA

This section describes our proposed model **ILKA** incorporating KG-attention. As shown in Figure 4, ILKA consists of the following three modules.

(1) PLM and KG Embeddings: The input sentences are fed to the PLM and pre-trained KG embedding model to obtain the PLM word embeddings and KG entity embeddings. The KG embedding model additionally obtains the KG relation embeddings of all relations defined in the KG.

(2) KG-attention: The KG-attention mechanism integrates the PLM word embeddings, KG entity embeddings, and KG relation embeddings.

(3) Classifier: Classification is performed by a transformer encoder-based classifier with self-attention. KG-attention, which consistently incorporates the relational information of the KG and PLM embeddings, can be regarded as the first layer of the transformer encoder-based classification model.

Table 2 compares the features of our method with those of four existing PLM+KG methods. ILKA approximates KG-neighbor embeddings based on KG relation embeddings and can consider the attention between the input sentence and KG-neighbors even if the relation between the entities is not connected in the KG. Among the compared methods, only ILKA achieves this functionality.

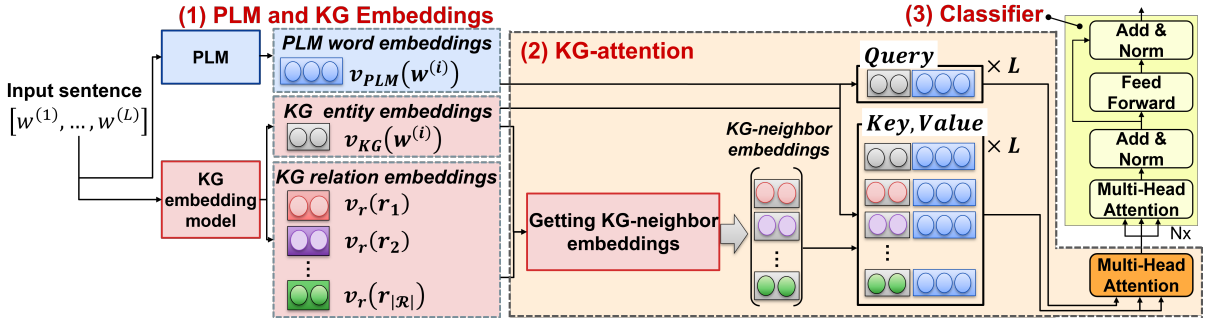

Figure 4: Overview of ILKA, which comprises three steps: (1) Obtain the PLM and KG embeddings; (2) Apply KG-attention to integrate the PLM word embeddings, KG entity embeddings, and KG relation embeddings; (3) Perform classification using a transformer encoder-based model.

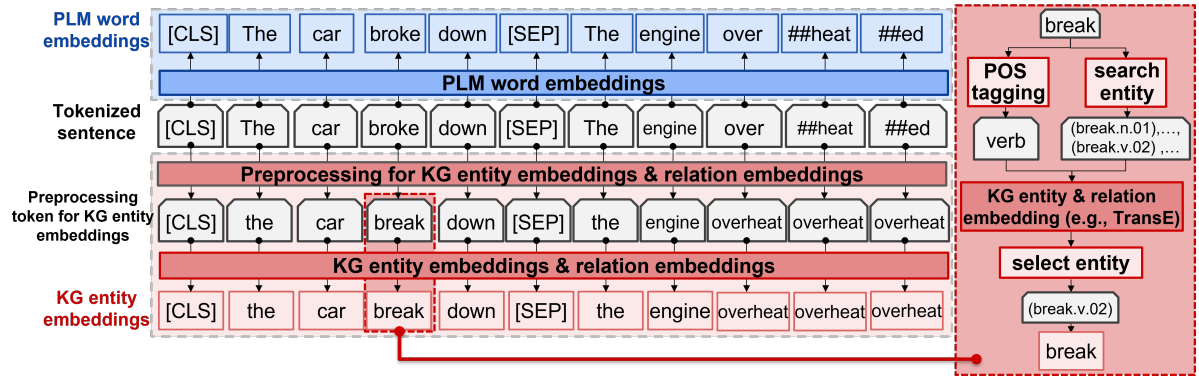

Figure 5: Procedure for obtaining the word embeddings shown in Figure 4 (1). The blue and red shading highlight the PLM word embeddings and KG entity embeddings, respectively.

3.3 Implementation

PLM word embeddings (shaded blue in Figure 5): Transformer-based PLMs often require a tokenization step to solve the out-of-vocabulary problem. For example, the sentence "The engine **overheated**" can be divided into the subword sequence "The engine **over ##heat ##ed**." PLMs give the word embeddings of each subword unit. We use existing PLMs for PLM word embeddings.

KG entity embeddings (shaded red in Figure 5): Unlike PLMs, KG entities are registered as word or phrase units. To obtain KG entity embeddings, the tokenized sentences are preprocessed as follows:

(1) After reverting the subwords to the original words, the original word is duplicated n times, where n is the number of subwords (e.g., [over, ##heat, ##ed] → [overheated, overheated, overheated]).

(2) The conjugated words are returned to their original form (e.g., broke → break, overheated → overheat). Uppercase letters are made lowercase (e.g., The → the).

In typical KGs, words with multiple meanings are registered as different entities. ILKA morphologically analyzes the input sentence to find the part-of-speech (POS), and then selects an entity with the same POS among the registered entities[1]. Among various registered entities, this process allows the selection of words with similar usage to those in the input sentences. If no embedding vector is obtained from the KG embedding model, then a zero vector is used. A zero vector is also allocated to special tokens such as [CLS], [SEP], and [PAD].

KG-attention Here we describe three implementations of KG-attention on TransE as an example.

1) ILKA-head: TransE embeds the head and tail entities as $\mathbf{v}_{\mathrm{KG}}(t) \approx \mathbf{v}_{\mathrm{KG}}(h) + \mathbf{v}_{\mathrm{r}}(r)$. Our KG-attention is designed to integrate the *head* → *tail* relation with the PLM embeddings as follows:

$$\mathbf{K}_{\mathrm{head}}^{(i,j)} = (\mathbf{v}_{\mathrm{KG}}(w^{(i)}) + \mathbf{v}_{\mathrm{r}}(r^{(j)})) \oplus \mathbf{v}_{\mathrm{PLM}}(w^{(i)}). \quad (6)$$

[1]If multiple entities have the same POS as the input word, the first entity is used. In contrast, if no entity has the same POS as the input word, an entity with a different POS is used.

2) ILKA-tail: A similar integration of the *tail* $\rightarrow$ *head* relation with the PLM embeddings is given as

$$\mathbf{K}_{\text{tail}}^{(i,j)} = (\mathbf{v}_{\text{KG}}(w^{(i)}) - \mathbf{v}_{\text{r}}(r^{(j)})) \oplus \mathbf{v}_{\text{PLM}}(w^{(i)}). \quad (7)$$

3) ILKA-both: The two relations above can be combined by concatenating the head and tail vectors from $\mathbf{v}_{\text{KG}}(w^{(i)})$:

$$\mathbf{K}_{\text{both}}^{(i,j)} = \begin{cases} \mathbf{K}_{\text{head}}^{(i,j)} \text{ if } j \in \{1, \ldots, |\mathcal{R}|\} \\ \mathbf{K}_{\text{tail}}^{(i,j-|\mathcal{R}|)} \text{ if } j \in \{|\mathcal{R}|+1, \ldots, 2|\mathcal{R}|\}. \end{cases} \quad (8)$$

Note that $\mathbf{K}^{(i,0)} = \mathbf{v}_{\text{KG}}(w^{(i)}) \oplus \mathbf{v}_{\text{PLM}}(w^{(i)})$ for all implementations and $\mathbf{K}_{\text{head}}, \mathbf{K}_{\text{tail}} \in \mathbb{R}^{L(|R|+1)\times(d_{\text{KG}}+d_{\text{PLM}})}$, while $\mathbf{K}_{\text{both}} \in \mathbb{R}^{L(2|R|+1)\times(d_{\text{KG}}+d_{\text{PLM}})}$.

Transformer encoder-based classifier For the classifier, we employed the transformer encoder model proposed by Vaswani et al. (2017), which comprises a stack of $N = 6$ identical layers. Each layer has two sub-layers: a multi-head self-attention mechanism and a position-wise fully connected feed-forward network.

4 Experiments

4.1 Benchmark Methods

The ILKA model was competed against two PLM-only methods: **BERT** (Devlin et al., 2019), **RoBERTa** (Liu et al., 2019)), and four PLM+KG methods: **ERNIE** (Zhang et al., 2019), **KnowBERT** (Peters et al., 2019), **CoLAKE** (Sun et al., 2020) and **KEPLER** (Wang et al., 2021).

4.2 Experimental Settings

The KG-attention was implemented in PyTorch (Paszke et al., 2019). As the KG embedding model, we employed the TransE model trained on WordNet and Wikipedia. The KG was based on WordNet (WN18 (Bordes et al., 2014) containing 18 relations) or Wikipedia (DBpedia50k (Shi and Weninger, 2018) containing 351 relations). Details can be found in Appendix A. We implemented TransE using PyKEEN (Ali et al., 2021) and trained it for 1,000 epochs. Following Bordes et al. (2013), the dimension of the KG embeddings was set to $d_{\text{KG}} = 50$ (see Appendix B for details). The PLM was implemented in two settings: $\text{BERT}_{\text{BASE}}$ for comparison with the BERT-based models ERNIE and KnowBERT and $\text{RoBERTa}_{\text{BASE}}$ for comparison with the RoBERTa-based models CoLAKE and KEPLER. The PLM implementations were obtained from HuggingFace Transformers (Wolf et al., 2020). For the classifier, we employed a transformer encoder with six self-attention layers. The batch sizes were set to 32 and 4 for the models with WN18 and DBPedia50k, respectively, and the maximum sequence length of the token was 128. All experiments were conducted on a 24GB NVIDIA® TITAN RTX™ GPU.

4.3 Tasks

ILKA was evaluated on three tasks: relation classification, entity typing, and language comprehension.

Relation Classification Relation classification determines the type of relation between two entities in a text. We employed TACRED as the relation classification task[2]. TACRED contains more than 106k sentences with typed subject and object spans and relation labels across 41 relations along with a *no-relation* label. The hyperparameters and other experimental settings were set following KEPLER.

Entity Typing Entity typing is the task of classifying a given entity into a pre-defined type. This task was evaluated using OpenEntity (Choi et al., 2018)[3]. Comparisons with ERNIE, KnowBERT, CoLAKE, and KEPLER were performed on nine general entity types under the same experimental settings.

Language Comprehension We adopted the General Language Understanding Evaluation (GLUE) (Wang et al., 2018) and Choice of Plausible Alternatives (COPA) (Roemmele et al., 2011) as language comprehension tasks. GLUE is a collection of diverse natural language comprehension tasks used in various PLM papers (Devlin et al., 2019; Liu et al., 2019). Existing PLM+KG methods have evaluated the GLUE task to determine whether their methods degenerate the performance on common NLP tasks (Zhang et al., 2019; Wang et al., 2021) (see Appendix C for details). We used the implementation and evaluation script implemented by the HuggingFace Transformers library

[2]TACRED is available from LDC. https://catalog.ldc.upenn.edu/LDC2018T24

[3]OpenEntity data can be obtained from ERNIE's github page. https://github.com/thunlp/ERNIE

Base-model	Model	P	R	F1
BERT	BERT	67.2	64.8	66.0
	ERNIE	70.0	66.1	68.0
	KnowBERT	73.5	64.1	68.5
	ILKA-head	72.6	68.9	70.7
RoBERTa	RoBERTa	70.8	69.6	70.2
	KEPLER	71.5	**72.5**	72.0
	ILKA-head	**73.9**	71.9	**72.9**

Table 3: Micro precision, recall and F1 scores on TACRED (%). The KnowBERT results shown in Table 3 were reevaluated in Wang et al. (2021).

Base-model	Model	P	R	F1
BERT	BERT	76.4	71.0	73.6
	ERNIE	78.4	72.9	75.6
	KnowBERT	77.9	71.2	74.4
	ILKA-head	75.5	74.4	75.0
RoBERTa	RoBERTa	75.1	73.4	74.3
	CoLAKE	77.0	75.7	76.4
	KEPLER	**77.8**	74.6	76.2
	ILKA-head	75.3	**78.1**	**76.7**

Table 4: Micro precision, recall and F1 scores on OpenEntity (%). The results of BERT and RoBERTa were reported by Wang et al. (2021). Other benchmark results were taken from the respective papers.

for GLUE [4]. To investigate whether our method is effective for complex language comprehension tasks using COPA, a causal inference task that requires world knowledge. In COPA, the models were given a premise and two alternatives and were required to choose the alternative with a higher causal relation to the premise. Figure 1 is an example of COPA.

In the GLUE and COPA tasks, we selected the best fine-tuning learning rate (among 5e-5, 4e-5, 3e-5, and 2e-5) on the Dev set for the BERT setting according to the BERT experiments and (among 3e-5, 2e-5, and 1e-5) for the RoBERTa setting according to the RoBERTa experiments. Our classification was essentially that of BERT's framework of "[CLS] sentenceA [SEP] sentenceB [SEP]," where [CLS] is a special token for classification purposes.

4.4 Results

Relation Classification Table 3 shows the result of TACRED. As in Wang et al. (2021), we show the KnowBERT result without entity type inputs for a fair evaluation. ILKA-head represents our model with WN18. ILKA-head achieved higher F1 scores than the existing studies using PLM and KG (BERT-based and RoBERTa-based models). The results suggest that the relation classification was accurately solved by both the KG embedding of the input sentence and the generalized embedding of neighboring entities.

Entity Typing Table 4 shows the experimental results of OpenEntity. ILKA-head based on RoBERTa achieved higher F1 scores than the existing models using PLM and KG. In the BERT-base model, ERNIE and ILKA-head obtained the highest F1 and Recall scores, respectively. In the RoBERTa-base model, ILKA-head had the highest F1 score. As our method obtained the highest Recall in both BERT-base and RoBERTa-base, we inferred that using the KG-neighbors provides a consistently high coverage of various relations.

Language Comprehension The experimental results of GLUE are shown in Table 5. ILKA-head$_{\text{BERT}}$ outperformed BERT-original and ERNIE in the following tasks: MNLI-mm, QQP, CoLA, STS-B, MRPC, and RTE. Moreover, ILKA-head$_{\text{RoBERTa}}$ outperformed RoBERTa-original, CoLAKE, and KEPLER in the following tasks: MNLI-m/mm, CoLA, and MRPC. On average, our model with a RoBERTa performed comparably to the RoBERTa-original model. The result suggests that integrating PLM and KG exerts no significant negative impact.

Table 6 presents the COPA results. ILKA outperformed the original BERT and RoBERTa in all settings. Causal reasoning is one of the tasks requiring world knowledge. ILKA worked effectively even for the large-scale PLM, suggesting that KG-attention provides an inference ability that cannot be matched by word co-occurrence alone.

5 Analysis

5.1 Ablation Study

Model Variants Table 7 presents the results of an ablation study for ILKA$_{\text{RoBERTa}}$ on the TACRED and OpenEntity tasks. In the "ILKA$_{\text{w/o-KGneighbors}}$" model, only the KG entity embeddings are provided in the input sentence (the KG-neighbor embeddings are omitted). This model is similar to ERNIE but has a slightly different network structure. "ILKA-head", "ILKA-tail", and "ILKA-both" represent different implementations of ILKA as described in subsection 3.3.

ILKA-head and ILKA-tail obtained higher F1 scores "ILKA$_{\text{w/o-KGneighbors}}$", suggesting that when

[4] https://github.com/huggingface/transformers/blob/master/examples/pytorch/text-classification/run_glue.py

Base-model	Model	MNLI-(m/mm) 392k	QQP 363k	QNLI 108k	SST-2 67k	CoLA 8.5k	STS-B 5.7k	MRPC 3.5k	RTE 2.5k
BERT	BERT (Devlin et al., 2019)	**84.6**/83.6	71.2	90.5	**93.5**	52.1	85.8	88.9	66.4
	ERNIE (Zhang et al., 2019)	84.0/83.2	71.2	**91.3**	**93.5**	52.3	83.2	88.2	68.8
	ILKA-head$_{\text{BERT}}$ (OURS)	83.8/**84.4**	**87.7**	90.7	92.2	**62.6**	**89.3**	**91.1**	**69.0**
RoBERTa	RoBERTa (Liu et al., 2019)	87.5/87.3	91.9	**92.8**	**94.8**	63.6	**91.2**	90.2	78.7
	CoLAKE (Sun et al., 2020)	87.4/87.2	**92.0**	92.4	94.6	63.4	90.8	90.9	77.9
	KEPLER (Wang et al., 2021)	87.2/86.5	91.7	92.4	94.5	63.6	**91.2**	89.3	**85.2**
	ILKA-head$_{\text{RoBERTa}}$ (OURS)	**87.7/87.6**	88.1	92.6	94.5	**64.4**	91.1	**92.0**	75.8

Table 5: Test results of GLUE. The number below each task denotes the number of training examples. The results of the benchmark methods were taken from the respective references. KEPLER represents the result of KEPLER-wiki, the most accurate model for GLUE in (Wang et al., 2021). The F1 scores are reported for QQP and MRPC, Spearman correlations are reported for STS-B, and the accuracy scores are reported for the other tasks.

	Model	Accuracy
BERT$_{\text{BASE}}$	BERT$_{\text{BASE}}$	68.8 $\pm$ 1.3
	ILKA-head$_{\text{BERTBASE}}$	**69.9** $\pm$ **1.1**
BERT$_{\text{LARGE}}$	BERT$_{\text{LARGE}}$	76.5 $\pm$ 2.7
	ILKA-head$_{\text{BERTLARGE}}$	**77.7** $\pm$ **1.2**
RoBERTa$_{\text{BASE}}$	RoBERTa$_{\text{BASE}}$	73.7 $\pm$ 1.3
	ILKA-head$_{\text{RoBERTaBASE}}$	**74.9** $\pm$ **2.1**
RoBERTa$_{\text{LARGE}}$	RoBERTa$_{\text{LARGE}}$	87.7 $\pm$ 0.9
	ILKA-head$_{\text{RoBERTaLARGE}}$	**88.6** $\pm$ **1.6**

Table 6: Test results on COPA. The results of BERT$_{\text{LARGE}}$ and RoBERTa$_{\text{LARGE}}$ were reported by Kavumba et al. (2019).

	TACRED			OpenEntity		
Model	P	R	F1	P	R	F1
ILKA$_{\text{w/o-KGneighbors}}$	73.7	70.8	72.2	77.8	71.0	74.3
ILKA-head	73.9	71.9	72.9	75.3	78.1	76.7
ILKA-tail	75.4	72.9	74.1	76.4	79.9	78.1
ILKA-both	76.4	70.7	73.5	74.0	73.2	73.6

Table 7: Results of the ablation study on TACRED and OpenEntity using the proposed ILKA$_{\text{RoBERTa}}$.

both the KG-neighbor and KG entity embeddings are employed, the model becomes more effective. ILKA-both, which simply concatenates the KG-neighbor embeddings of the head and tail entities, was less accurate than ILKA-head and ILKA-tail.

Knowledge Graph Variants Table 8 shows the experimental results for different KGs. ILKA-head$_{\text{WordNet}}$ outperformed ILKA-head$_{\text{Wiki}}$, possibly because the large number of relations in ILKA-head$_{\text{Wiki}}$ leads to a sparser embedding space. Nevertheless, ILKA-head$_{\text{Wiki}}$ is still comparable to benchmark methods shown in Table 3 and Table 4.

5.2 Training Runtime Comparison

Most of the existing PLM+KG methods jointly learn contextualized representations of both language and KG with the MLM objective. In general, learning a model from scratch by this method is excessively time-consuming. For example, CoLAKE training on 8 32G NVIDIA V100 GPUs required 38 hours (Sun et al., 2020). Our method reduced the training time because it requires only fine-tuning. Specifically, the training runtimes of ILKA$_{\text{RoBERTa}}$ with fine-tuning in three epochs on CoLA, STS-B, MRPC, and RTE were approximately 10 mins (see Appendix D for details). Despite its short learning time, our method was more accurate than the existing PLM+KG methods.

	TACRED			OpenEntity		
Model	P	R	F1	P	R	F1
ILKA-head$_{\text{WordNet}}$	73.9	71.9	72.9	75.3	78.1	76.7
ILKA-head$_{\text{Wiki}}$	74.5	69.5	71.9	75.1	77.8	76.4

Table 8: Results of the Knowledge Graph Variants on TACRED and OpenEntity using the proposed ILKA$_{\text{RoBERTa}}$.

Our method needs more training time than pure PLMs because it requires POS tagging first and then entity lookup. In our experimental environment (24GB NVIDIA® TITAN RTX™ GPU), RoBERTa's fine-tuning runtime is 3.9 and 2.8 min for CoLA and STS-B, respectively. In contrast, ILKA-head takes about 11.5 and 7.8 min for CoLA and STS-B, respectively (details in Appendix D), so it takes about three times longer for fine-tuning. For inference, we observed the same tendency. We believe that the increase in learning time does not pose a significant practical problem.

5.3 Discussion

Our proposed method improves the F1 scores for the relation classification task (TACRED) and the entity typing task (OpenEntity), as shown in Tables 3 and 4. However, it is also true that the improvement is insignificant. An analysis of the number of KG entities in a sentence shows that only a few

entities appear in an input sentence. Specifically, the average number of entities in a sentence is 6.5 subwords (5% for a maximum token length of 128), and the rest are empty. This could have hampered learning efficiency.

6 Conclusion

We proposed the KG-attention mechanism and developed the ILKA model to integrate KGs into PLMs in a consistent manner. Unlike the existing PLM+KG integration methods, KG-attention generalizes the embeddings of neighboring entities using the relation embeddings, and selects a proportion of these embeddings through the attention mechanism. Accordingly, it can handle relations between unconnected entities in the KG. In the relation classification and entity typing experiments, ILKA yielded higher Recall and F1 scores than conventional PLM+KG methods. In the GLUE task, factual knowledge produced no negative effect on language comprehension by ILKA. In addition, the higher scores for COPA than for the PLM alone suggest that our method adequately processes complex language comprehension tasks.

Our proposed method has one limitation: when the number of relations is large, the dimension of the attention enlarges accordingly. To avoid this size explosion, we could employ the weighted sums of relation embeddings as keys and queries instead of arranging all relation embeddings in the direction of attention.

Ethical Concerns

This study used the existing datasets WN18 (Bordes et al., 2013), DBpedia50k (Shi and Weninger, 2018), TACRED (Zhang et al., 2017), OpenEntity (Choi et al., 2018), GLUE (Wang et al., 2018), and COPA (Roemmele et al., 2011), which are not expected to present any ethical concerns.

References

Mehdi Ali, Max Berrendorf, Charles Tapley Hoyt, Laurent Vermue, Sahand Sharifzadeh, Volker Tresp, and Jens Lehmann. 2021. PyKEEN 1.0: A Python Library for Training and Evaluating Knowledge Graph Embeddings. *Journal of Machine Learning Research*, 22(82):1–6.

Kurt Bollacker, Colin Evans, Praveen Paritosh, Tim Sturge, and Jamie Taylor. 2008. Freebase: A collaboratively created graph database for structuring human knowledge. In *Proceedings of the 2008 ACM SIGMOD International Conference on Management of Data*, page 1247–1250. Association for Computing Machinery.

Antoine Bordes, Xavier Glorot, Jason Weston, and Yoshua Bengio. 2014. A semantic matching energy function for learning with multi-relational data. *Machine Learning*, 94(2):233–259.

Antoine Bordes, Nicolas Usunier, Alberto Garcia-Duran, Jason Weston, and Oksana Yakhnenko. 2013. Translating embeddings for modeling multi-relational data. In *Advances in Neural Information Processing Systems*, volume 26. Curran Associates, Inc.

Linlin Chao, Jianshan He, Taifeng Wang, and Wei Chu. 2021. PairRE: Knowledge graph embeddings via paired relation vectors. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 4360–4369, Online. Association for Computational Linguistics.

Eunsol Choi, Omer Levy, Yejin Choi, and Luke Zettlemoyer. 2018. Ultra-fine entity typing. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 87–96, Melbourne, Australia. Association for Computational Linguistics.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 4171–4186, Minneapolis, Minnesota. Association for Computational Linguistics.

Pride Kavumba, Naoya Inoue, Benjamin Heinzerling, Keshav Singh, Paul Reisert, and Kentaro Inui. 2019. When choosing plausible alternatives, clever hans can be clever. In *Proceedings of the First Workshop on Commonsense Inference in Natural Language Processing*, pages 33–42, Hong Kong, China. Association for Computational Linguistics.

Zhenzhong Lan, Mingda Chen, Sebastian Goodman, Kevin Gimpel, Piyush Sharma, and Radu Soricut. 2020. ALBERT: A lite BERT for self-supervised learning of language representations. In *8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, April 26-30, 2020*. OpenReview.net.

Bill Yuchen Lin, Xinyue Chen, Jamin Chen, and Xiang Ren. 2019. KagNet: Knowledge-aware graph networks for commonsense reasoning. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 2829–2839, Hong Kong, China. Association for Computational Linguistics.

Yankai Lin, Zhiyuan Liu, Maosong Sun, Yang Liu, and Xuan Zhu. 2015. Learning entity and relation embeddings for knowledge graph completion. In *Proceedings of the Twenty-Ninth AAAI Conference on Artificial Intelligence*, AAAI'15, page 2181–2187. AAAI Press.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Roberta: A robustly optimized BERT pretraining approach. *CoRR*, abs/1907.11692.

George A. Miller. 1995. Wordnet: A lexical database for english. *Commun. ACM*, 38(11):39–41.

Dat Quoc Nguyen. 2020. A survey of embedding models of entities and relationships for knowledge graph completion. In *Proceedings of the Graph-based Methods for Natural Language Processing (TextGraphs)*, pages 1–14, Barcelona, Spain (Online). Association for Computational Linguistics.

Maximilian Nickel, Volker Tresp, and Hans-Peter Kriegel. 2011. A three-way model for collective learning on multi-relational data. In *Proceedings of the 28th International Conference on International Conference on Machine Learning*, ICML'11, page 809–816.

Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Kopf, Edward Yang, Zachary DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. 2019. Pytorch: An imperative style, high-performance deep learning library. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett, editors, *Advances in Neural Information Processing Systems 32*, pages 8024–8035. Curran Associates, Inc.

Matthew E. Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer. 2018. Deep contextualized word representations. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 2227–2237, New Orleans, Louisiana. Association for Computational Linguistics.

Matthew E. Peters, Mark Neumann, Robert Logan, Roy Schwartz, Vidur Joshi, Sameer Singh, and Noah A. Smith. 2019. Knowledge enhanced contextual word representations. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 43–54, Hong Kong, China. Association for Computational Linguistics.

Melissa Roemmele, Cosmin Adrian Bejan, and Andrew S. Gordon. 2011. Choice of Plausible Alternatives: An Evaluation of Commonsense Causal Reasoning. In *AAAI Spring Symposium on Logical Formalizations of Commonsense Reasoning*, Stanford University.

Michael Schlichtkrull, Thomas N Kipf, Peter Bloem, Rianne Van Den Berg, Ivan Titov, and Max Welling. 2018. Modeling relational data with graph convolutional networks. In *European semantic web conference*, pages 593–607. Springer.

Baoxu Shi and Tim Weninger. 2018. Open-world knowledge graph completion. In *AAAI*.

Robert Speer, Joshua Chin, and Catherine Havasi. 2016. Conceptnet 5.5: An open multilingual graph of general knowledge. In *AAAI Conference on Artificial Intelligence*.

Tianxiang Sun, Yunfan Shao, Xipeng Qiu, Qipeng Guo, Yaru Hu, Xuanjing Huang, and Zheng Zhang. 2020. CoLAKE: Contextualized language and knowledge embedding. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 3660–3670, Barcelona, Spain (Online). International Committee on Computational Linguistics.

Zhiqing Sun, Zhi-Hong Deng, Jian-Yun Nie, and Jian Tang. 2019. Rotate: Knowledge graph embedding by relational rotation in complex space. In *International Conference on Learning Representations*.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Ł ukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc.

Alex Wang, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel Bowman. 2018. GLUE: A multi-task benchmark and analysis platform for natural language understanding. In *Proceedings of the 2018 EMNLP Workshop BlackboxNLP: Analyzing and Interpreting Neural Networks for NLP*, pages 353–355, Brussels, Belgium. Association for Computational Linguistics.

Chao Wang and Hui Jiang. 2019. Explicit utilization of general knowledge in machine reading comprehension. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 2263–2272, Florence, Italy. Association for Computational Linguistics.

Xiang Wang, Xiangnan He, Yixin Cao, Meng Liu, and Tat-Seng Chua. 2019. Kgat: Knowledge graph attention network for recommendation. In *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery Data Mining*, KDD '19, page 950–958, New York, NY, USA. Association for Computing Machinery.

Xiaozhi Wang, Tianyu Gao, Zhaocheng Zhu, Zhengyan Zhang, Zhiyuan Liu, Juanzi Li, and Jian Tang. 2021. KEPLER: A unified model for knowledge embedding and pre-trained language representation. *Trans. Assoc. Comput. Linguistics*, 9:176–194.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Remi Louf, Morgan Funtowicz, Joe Davison, Sam Shleifer, Patrick von Platen, Clara Ma, Yacine Jernite, Julien Plu, Canwen Xu, Teven Le Scao, Sylvain Gugger, Mariama Drame, Quentin Lhoest, and Alexander Rush. 2020. Transformers: State-of-the-art natural language processing. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 38–45, Online. Association for Computational Linguistics.

An Yang, Quan Wang, Jing Liu, Kai Liu, Yajuan Lyu, Hua Wu, Qiaoqiao She, and Sujian Li. 2019. Enhancing pre-trained language representations with rich knowledge for machine reading comprehension. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 2346–2357, Florence, Italy. Association for Computational Linguistics.

Michihiro Yasunaga, Hongyu Ren, Antoine Bosselut, Percy Liang, and Jure Leskovec. 2021. QA-GNN: Reasoning with language models and knowledge graphs for question answering. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 535–546, Online. Association for Computational Linguistics.

Tao Yu, Rui Zhang, Kai Yang, Michihiro Yasunaga, Dongxu Wang, Zifan Li, James Ma, Irene Li, Qingning Yao, Shanelle Roman, Zilin Zhang, and Dragomir Radev. 2018. Spider: A large-scale human-labeled dataset for complex and cross-domain semantic parsing and text-to-SQL task. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 3911–3921, Brussels, Belgium. Association for Computational Linguistics.

Yuhao Zhang, Victor Zhong, Danqi Chen, Gabor Angeli, and Christopher D. Manning. 2017. Position-aware attention and supervised data improve slot filling. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 35–45, Copenhagen, Denmark. Association for Computational Linguistics.

Zhengyan Zhang, Xu Han, Zhiyuan Liu, Xin Jiang, Maosong Sun, and Qun Liu. 2019. ERNIE: Enhanced language representation with informative entities. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 1441–1451, Florence, Italy. Association for Computational Linguistics.

A Details of WN18 and DBpedia50k

Table 9 present the scale of WN18 and DBPedia50k and Table 10 presents the types of relations in WN18.

	WN18	DBPedia50k
#(ENTITIES)	40,943	49,900
#(RELATIONS)	18	351
#(Triples in Train)	141,442	32,388
#(Triples in Valid)	5,000	399
#(Triples in Test)	5,000	10,969

Table 9: Scales of WN18 and DBPedia50k.

_hyponym,_hypernym,_part_of,
_derivationally_related_form,
_has_part,_similar_to,_also_see
_member_meronym,_member_holonym,
_member_of_domain_topic,
_synset_domain_topic_of,
_instance_hyponym,_instance_hypernym,
_member_of_domain_region,_verb_group,
_synset_domain_region_of,
_member_of_domain_usage,
_synset_domain_usage_of

Table 10: Types of relations in WN18.

B Training TransE using PyKEEN

PyKEEN is a Python package designed for training and evaluating KG embedding models. It simply describes the training of the knowledge embedding model in the form of a pipeline. Figure 6 shows the code for training the knowledge embedding model. A KG embedding model can be trained by specifying the training conditions in PyKEEN (e.g., the model, dataset name, and number of dimensions).

C GLUE task

The GLUE is the common language comprehension task in NLP. It consists of nine tasks: two single-sentence tasks (SST-2, CoLA), three sentence similarity tasks (MRPC, STS-B, QQP), and four natural language inference (NLI) tasks (MNLI, QNLI, RTE, WNLI). Following Zhang et al. (2019), we evaluated ILKA for eight tasks excluding WNLI. WNLI was excluded from the evaluation data because none of the benchmark methods were evaluated against it. The reasons are described in (Devlin et al., 2019).

Code 1 Training TransE in PyKEEN

```
modeldir = "dirname"
model = pipeline(
        model='TranE',
        dataset='wn18',
        model_kwargs=dict(
                embedding_dim=50,
        ),
        training_kwargs=dict(
                num_epochs=1000,
                batch_size=128,
                checkpoint_name='checkpoint.pt',
                checkpoint_frequency=50,
                checkpoint_directory=dirname,
        ),
)
```

Figure 6: Code for training the knowledge embedding model in PyKEEN. This model uses WN18 with TransE.

D Training Runtime

Figure 7 shows the training runtimes of $\text{ILKA}_{\text{RoBERTa}}$ over three epochs of fine tuning in the CoLA, STS-B, MRPS, and RTE of the GLUE task. Plotted are the averages of five training runs under each condition. The fine-tuning time of $\text{ILKA}_{\text{RoBERTa}}$ on these tasks was approximately 10 minutes.

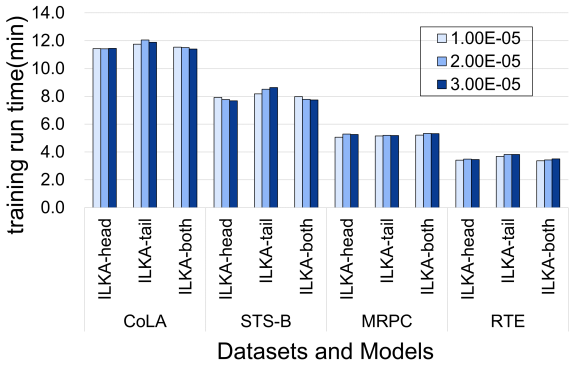

Figure 7: Training runtime for fine-tuning (epochs = 3).

Generic Overgeneralization in Pre-trained Language Models

Sello Ralethe and **Jan Buys**
Department of Computer Science, University of Cape Town, South Africa
rltsel002@myuct.ac.za, jbuys@cs.uct.ac.za

Abstract

Generic statements such as "ducks lay eggs" make claims about *kinds*, e.g., ducks as a category. The generic overgeneralization effect refers to the inclination to accept false universal generalizations such as "all ducks lay eggs" or "all lions have manes" as true. In this paper, we investigate the generic overgeneralization effect in pre-trained language models experimentally. We show that pre-trained language models suffer from overgeneralization and tend to treat quantified generic statements such as "all ducks lay eggs" as if they were true generics. Furthermore, we demonstrate how knowledge embedding methods can lessen this effect by injecting factual knowledge about *kinds* into pre-trained language models. To this end, we source factual knowledge about two types of generics, minority characteristic generics and majority characteristic generics, and inject this knowledge using a knowledge embedding model. Our results show that knowledge injection reduces, but does not eliminate, generic overgeneralization, and that majority characteristic generics of kinds are more susceptible to overgeneralization bias. We release the dataset and code[1].

1 Introduction

Generics are sentences such as "tigers have stripes" that express generalizations about kinds, although they are not universal or without exceptions. For example, there are albino tigers that do not have stripes. Even though there are exceptions, generics are regarded as true. However, universally quantified statements such as "all ducks lay eggs" should be perceived as false as they can easily be invalidated because the quantifier *all* does not allow exceptions; it is only mature female ducks that are capable of laying eggs.

Empirical data from linguistics studies show that children and adults often tend to treat quantified statements such as "all tigers have stripes" as if they were generics (Khemlani et al., 2007; Hollander et al., 2002). Leslie et al. (2011) term this phenomena the *generic overgeneralization* (GOG) effect and allot it to a cognitive tendency that causes people to overgeneralize from the truth of a generic ("lions have manes") to the truth of a corresponding universal statement ("all lions have manes").

	PLM		PLM+KEPLER	
#	BERT	RoBERTa	BERT	RoBERTa
1	All	Some	Mountain	Male
2	Most	Most	Young	Mountain
3	Some	All	Male	Sea
4	Every	Many	Most	Some
5	Many	Even	Some	Most

Table 1: The top 5 words predicted by BERT and RoBERTa for filling the mask in the generic "`[MASK]` lions have manes". The outputs are shown before and after knowledge injection with KEPLER.

Pre-trained language models (PLMs) such as BERT (Devlin et al., 2019) and RoBERTa (Liu et al., 2019b) learn useful language representation from large-scale unstructured text and are able to store information about the world in their parameters (Clark et al., 2019; Liu et al., 2019a; Rogers et al., 2020). In this paper, we investigate the GOG effect in PLMs, and ask if embedding factual knowledge about kinds during pre-training can reduce this effect.

First, we investigate if PLMs can improve the classification accuracy of distinguishing between generic and non-generic statements. We construct datasets of minority and majority characteristic generics of *kinds*, and use these to train and evaluate a baseline model and the PLMs. Results show that the pre-trained language models outperform the baseline on the classification task. We also demonstrate that the fine-tuned language models

[1] https://github.com/sello-ralethe/GOG-in-PLMs

Proceedings of the 29th International Conference on Computational Linguistics, pages 3187–3196
October 12–17, 2022.

classify quantified statements such as "all ducks lay eggs" as generics, indicating that the models do overgeneralize and thus exhibit the GOG effect.

Next, we define a masked word prediction task to test if the PLMs exhibit the GOG effect by observing if the PLMs predict quantifiers to fill the masked words. For example, we ask PLMs to fill the masked position in a statement such as "`[MASK]` lions have manes" (Table 1). The low performance of the PLMs on this task, evaluated using mean reciprocal rank and precision at 5, confirms the presence of the GOG effect.

Given these observations, we ask whether injecting factual knowledge about kinds during language model pre-training reduces the GOG effect. We source factual knowledge about kinds from ASCENT KB (Nguyen et al., 2021), a knowledge base which contains facet-enriched assertions together with their associated context. The knowledge from ASCENT KB is injected into the PLMs using KEPLER (Wang et al., 2021b), a model for embedding knowledge into PLMs using entity descriptions and entity relations data. Experimental results suggest that the injected knowledge lessens the GOG effect, but does not eliminate it completely.

Our contributions are: (i) we introduce new datasets for evaluating generic overgeneralization in PLMs, (ii) we demonstrate that PLMs exhibit the GOG effect, (iii) we show that embedding factual knowledge can reduce the GOG effect, and (iv) our results suggest that majority characteristic generics are more susceptible to overgeneralization bias. To the best of our knowledge, we present the first work investigating generic overgeneralization in PLMs.

2 Background

2.1 Genericity and the Generic Overgeneralization Effect

Generics express generalizations about kinds, and lack explicit quantifiers such as *all*, *some* and *most*. Unlike quantified statements, generics do not communicate information about how many members of the kind have the property in question.

Similarly, there is no direct relation between the prevalence of a property among members of a kind and the acceptability of the relevant generic. For example, the generic statement "ducks lay eggs" is accepted even though only mature fertile females lay eggs, but the generic statement "ducks are female" is rejected (Leslie et al., 2011).

If people believe that the statement "ducks lay eggs" is true, they will tend to accept a quantified statement such as "all ducks lay eggs", because resorting to a default operation saves cognitive effort. This phenomena is called the *generic overgeneralization* (GOG) effect and is defined as "overgeneralizing from the truth of a generic to the truth of the corresponding universal statement" (Leslie et al., 2011, p. 17).

Quantifiers have been shown to influence the GOG effect, but the question of which types of quantified statements are susceptible to overgeneralization has not been resolved yet (Karczewski et al., 2020).

In this paper we focus on minority characteristic generics and majority characteristic generics. Minority characteristic generics include generics such as "lions have manes", which are only true about a minority of the kind and usually refer to gender-related properties. Conversely, majority characteristic generics include generics such as "tigers have stripes", which refer to properties that are directly related to the nature of the kind and are prevalent, though not universal, among members of the kind (Prasada et al., 2013). Majority characteristic generics do not need to express exceptionless universal generalizations, since some tigers (e.g., albino tigers) may fail to possess the property.

2.2 Genericity in NLP

Genericity is a key component in the study of human cognition because it demonstrates our inclination to organize our experience of the world into categories, kinds or classes (Lazaridou-Chatzigoga, 2019). The importance of generic language has been recognized in the artificial intelligence and natural language processing community for tasks that involve knowledge acquisition, ontology development, and semantic inference (Monahan et al., 2015; Zhou et al., 2015).

Reiter and Frank (2010) developed a corpus-based supervised learning approach for identifying generic noun phrases in context, using linguistically-motivated features in a Bayesian network classifier. Their experiments were restricted to generic noun phrases, as at the time there were no corpora available that contain annotations for genericity at the sentence level. Friedrich and Pinkal (2015) presented a discourse-sensitive genericity labeler, using Conditional Random Fields as a sequence labeler. Their experiments showed that context information improves accuracy, and their

model outperforms the approach proposed by Reiter and Frank (2010).

Govindarajan et al. (2019) proposed a semantic framework for modeling linguistic expressions of generalization, suggesting that such expressions should be captured in a continuous multi-label system, rather than a multi-class system. This was accomplished by decomposing categories such as episodic, habitual, and generic into simple referential properties of predicates and their arguments. The framework was used to construct a dataset covering the full Universal Dependencies English Web Treebank. Furthermore, Govindarajan et al. (2019) presented models for predicting expressions of linguistic generalization, which combine hand-engineered type- and token-level features with static and contextual learned representations.

In summary, although multiple prior works cover genericity in NLP, the generic overgeneralization effect has not yet been investigated specifically.

2.3 Knowledge Enhanced PLMs

Incorporating commonsense knowledge is necessary and beneficial for language inference (LoBue and Yates, 2011; Bowman et al., 2015; Rashkin et al., 2018b), reading comprehension (Mihaylov and Frank, 2018; Rashkin et al., 2018a), and generation based question answering (Chen et al., 2020). Recent research has shown that PLMs do not sufficiently capture factual commonsense world knowledge from the text used in their pre-training (Wang et al., 2021a; Yu et al., 2022; Gong et al., 2020). To address this problem, knowledge embedding methods have been proposed with the aim of encoding the relational facts in knowledge graphs through entity embeddings (Liu et al., 2020; Tang et al., 2020; Dai et al., 2020).

In this paper, we implement KEPLER (Wang et al., 2021b) to inject factual knowledge into PLMs (BERT and RoBERTa) with the aim of reducing the GOG effect. KEPLER jointly optimizes parameters with knowledge embedding and masked language modelling objectives to blend factual knowledge with language representations. The texts and entities are encoded into a unified semantic space using a single PLM encoder. For the knowledge embedding objective, entity descriptions are encoded as entity embeddings and are trained similarly to other knowledge embedding methods such as AutoETER (Niu et al., 2020), which is a knowledge graph embedding framework with automated entity type representation. The masked language modelling objective is implemented using existing approaches such as BERT (Devlin et al., 2019) and RoBERTa (Liu et al., 2019b).

3 The GOG Effect Evaluation Task

We introduce the GOG effect evaluation task, describe how the datasets were created, and discuss how the factual knowledge about kinds that can be used for knowledge injection was extracted.

3.1 Task Definition

We introduce two tasks: a classification task, and a masked language modelling task. The classification task evaluates whether a model can classify a statement as generic or non-generic: PLMs can be fine-tuned on this task using the training data provided. However, this task does not evaluate the GOG effect directly.

Secondly, we propose a masked language modelling task to evaluate if PLMs exhibit the GOG effect. We mask the position preceding a generic statement and use the PLM to predict the token. For example, given the generic "lions have manes", we ask a PLM to fill the blank in "`[MASK]` lions have manes". We use mean reciprocal rank (MRR) on the predicted word distribution to evaluate the PLM's ability to fill the masked position; precision at 5 (P@5) measures the relevance of top 5 predicted words. We evaluate whether one of the following universal quantifiers are predicted: *all*, *every*, *most*, *some*, *few* and *many*. Here, we do not consider the truthfulness of the quantified generic statement; that is, although the resulting quantified statement might not be factual, our aim is to evaluate if the PLMs would give a high probability to quantifier tokens when asked to predict the mask token. The higher the rank of the quantifiers, the stronger the PLM exhibits the GOG effect; a very low rank or low precision would indicate that the PLM does not exhibit the GOG effect.

3.2 Task Data

The proposed tasks focus on minority and majority characteristic of kind generics. These are generics that are only true about a minority or majority of a kind. We created a list of animals, which includes reptiles, fish, birds, mammals and amphibians, and used it to sample generic statements from GenericsKB (Bhakthavatsalam et al., 2020). GenericsKB is a large repository of 3.4M standalone generics har-

	Animal Types	ASCENT KB Triplets	Generic Statements	Minority Generics	Majority Generics
Amphibians	18	6 783	1 856	256	414
Reptiles	31	18 349	4 266	598	862
Fish	60	38 633	5002	471	788
Birds	76	63 967	10 604	1 051	1 811
Mammals	263	275 101	38 640	3 508	4 875
Total	448	402 833	60 368	5 884	8 750

Table 2: Datasets statistics. The minority and majority characteristic generic statements are used for evaluation and excluded from the other generic statements used for training.

vested from a webcrawl of 1.7B sentences. GenericsKB was constructed by first using a set of rules to identify candidate standalone generic sentences, and then applying a crowdsource-trained BERT classifier to assign a confidence to each generic sentence (Bhakthavatsalam et al., 2020).

Our list contains animals that are present in both GenericsKB and in ASCENT KB; the latter is used to sample factual assertions for each animal (see §3.3). For each animal, we sampled generic sentences with an associated confidence score greater than 0.5. Table 2 shows the statistics of the datasets we constructed. In total, we have three disjoint generic statements datasets.

To construct the minority characteristic generics dataset, we created a list of identifiers to sample generic statements about each animal from GenericsKB: *female*, *male*, *infant*, *young*, *adult*, *mature*, and *old*. Furthermore, we sampled generic statements from GenericsKB that have existential quantifiers *some* and *few*. In the final dataset, we removed the identifier words and the quantifiers from all generic sentences. For example, a generic statement such as "male lions have manes" is stored in the minority characteristic generics dataset as "lions have manes".

We similarly constructed the majority characteristic generics dataset by sampling quantified generic statements from GenericsKB that had these quantifiers: *all*, *many*, *every*, and *most*. Quantifiers were also removed from the generic statements before adding them to the dataset. For example, a generic statement such as "all zebras have different stripes" is stored in the majority characteristic generics dataset as "zebras have different stripes".

The third dataset consists of other types of generic statements about the animals in our list. We use this dataset to train models for the classification task and to further pretrain the language models, while the gold minority and majority characteristic generics datasets were used for evaluation and knowledge probing.

Additionally, we created a dataset of non-generic statements for training the classifiers by sampling sentences with a confidence score of les that 0.3 from GenericsKB. This dataset contains statements such as "a pitbull mauled a child" and "the snake laid some eggs". These statements are not generics because they apply only to a specific individual member of a kind.

3.3 Commonsense Knowledge Data

In order to perform PLM knowledge injection using KEPLER we need factual knowledge in the form of <*Subject, Predicate, Object*> triplets, and textual description data for each subject in a given triplet. We use the list of animals to sample SPO triplets from ASCENT KB (Nguyen et al., 2021) as a source of factual knowledge. For each animal in our list, we webcrawled A-Z-Animals.com[2] and scrapped textual data that include description like classification and evolution, anatomy and appearance, distribution and habitat, behaviour and lifestyle, reproduction and life cycles, and diet and prey. This information is then used as entity descriptions for each subject in the SPO triplets. For each SPO triplet, we align the textual description data with each subject in the triplet. For example, given a triplet <*elephant, uses, its trunk*>, we align the textual data about Elephants crawled from A-Z-Animals.com. Table 2 shows the number of SPO triplets extracted from ASCENT KB for each of the different types of animals in our list.

The factual data includes general information about the kinds, and makes exceptions to generic

[2] https://a-z-animals.com/animals/

statements salient. For example, existence of albino tigers implies that not all tigers have stripes. Therefore we hypothesize that the factual knowledge could be used to reduce the GOG effect in language models with respect to universal majority and minority characteristic generics. For majority characteristic generics, the factual data includes information about differences between sub-kinds of a given kind, such as the color of fur and type of food. For minor characteristic, it contains knowledge that emphasizes gender differences such as the different sizes or different roles of males and females.

4 Task Results

4.1 Generics Classification Task

First we train and evaluate models to classify statements as generics or not. As a baseline, we train a bi-directional LSTM on the generic and non-generic statements training dataset (which excludes the minority and majority characteristic generics). We use the base versions of BERT (Devlin et al., 2019) and RoBERTa (Liu et al., 2019b) as PLMs and fine-tune them on the same classification task.

For the classification task, we had **60k** generic statements, and sampled **50k** non-generic sentences from GenericsKB, which resulted in 110k training sentences for the classification task. From these, we used 20% of the sentences for testing and then used the held out minority and majority characteristic generics data for evaluation.

We report the F1-score and accuracy of the generics classification in Table 3, evaluated using the minority generics dataset and majority generics dataset. The results show that, as expected, the PLMs outperformed our baseline on the classification task.

4.2 The GOG Effect Evaluation Task

In this evaluation, we use the generics training data (not include minority and majority characteristic generics) to further pretrain the PLMs. The masked token prediction task is used to determine if the PLMs exhibit the GOG effect. We implement KEPLER (Wang et al., 2021b) to inject factual knowledge into the PLMs, using the data extracted from ASCENT KB (§3.3). The presence of quantifiers in the highest-ranked words that the PLMs predict in the masked positions would indicate that the PLMs exhibit the GOG effect. For each minority or majority characteristic generic statement in our datasets, we evaluate whether one of the following universal quantifiers are predicted: *all*, *every*, *most*, *some*, *few* and *many*. We conduct separate experiments for minority characteristic generics and majority characteristic generics in order to demonstrate which type of generics is more susceptible to overgeneralization bias.

The results in Tables 4 and 5 show that the MRR and P@5 scores are considerably lower after knowledge injection for both BERT and RoBERTa. This means that knowledge injection with KEPLER decreases the likelihood of predicting quantifiers, and therefore reduces the GOG effect. The scores of majority characteristic generics are higher than those of minority characteristic generics, indicating that majority characteristic generics are more susceptible to overgeneralization effect.

However the injection of factual knowledge results in a bigger reducing in the GOG effect for majority characteristics, such that the overgeneralization scores on minority and majority characteristics are very similar after knowledge injection. We postulate that this could be due to the quantity of factual knowledge that made exceptions more salient for majority characteristic generics. That is, a higher number of triples sampled from ASCENT KB contain factual knowledge about majority characteristic of kinds than factual knowledge about minority characteristic of kinds. Thus, more knowledge about majority characteristic of kinds was injected in the PLMs compared to knowledge about minority characteristic of kinds.

As a qualitative example, we asked BERT and RoBERTa to fill in the mask in the statement "`[MASK]` lions have manes" before and after knowledge injection (Table 1). Without knowledge injection, both models exhibits a preference for universal quantifiers, although RoBERTa ranks the conditional quantifier *some* highest, which suggests less overgeneralization than BERT which ranks *all* at the top. After knowledge injection the top three words are no longer quantifiers, which shows that overgeneralization is reduced. However, the presence of *most* among the top 5 predictions indicates that the GOG effect has not been eliminated completely in either model.

5 Probing the Injected Knowledge

The results reported in the previous section warrant us to probe the injected knowledge in order to determine if the PLMs "understand" the injected

	Minority Generics		Majority Generics	
Model	**F1**	**Accuracy**	**F1**	**Accuracy**
Bi-LSTM	0.80	0.83	0.82	0.86
BERT	0.88	0.90	0.89	0.91
RoBERTa	0.90	0.93	0.92	0.95

Table 3: Results of the generics classification task, comparing the PLMs (BERT and RoBERTa) against a Bi-LSTM baseline on minority and majority characteristic generics datasets.

	PLM		PLM+KEPLER	
Model	**MRR**	**P@5**	**MRR**	**P@5**
BERT	0.326	0.305	0.137	0.106
RoBERTa	0.329	0.307	0.135	0.108

Table 4: Mean Reciprocal Rank (MRR) and Precision at 5 (P@5) of universal quantifiers on the GOG effect evaluation task for minority characteristic generics. We report the scores before and after injecting factual knowledge into the PLMs with KEPLER. Lower scores indicate less overgeneralization.

	PLM		PLM+KEPLER	
Model	**MRR**	**P@5**	**MRR**	**P@5**
BERT	0.337	0.318	0.138	0.109
RoBERTa	0.428	0.411	0.152	0.117

Table 5: Mean Reciprocal Rank (MRR) and Precision at 5 (P@5) of universal quantifiers on the GOG effect evaluation task for majority characteristic generics. We report the scores before and after injecting factual knowledge into the PLMs with KEPLER. Lower scores indicate less overgeneralization.

factual knowledge. For example, do the PLMs understand that it is only mature, male lions that can have manes? Furthermore, do PLMs, with factual knowledge, correctly predict relevant tokens that could make quantified generic statements true?

5.1 Quantified Statement Classification

We fine-tune the knowledge-enhanced PLMs on the generics classification task (§4.1) and test if *quantified statements* are classified as generics. We quantify the minority characteristic generics with the quantifiers *many* and *most*, and the majority characteristic generics with *few* and *some*. This allows us to falsify the generics in both datasets. For example, the statement "most lions have manes" is not a true generic statement because only a minority of lions have manes. Similarly, "few tigers have stripes" is also not a true generic statement because most tigers do have stripes. Although the classifiers were trained to classify if a given statement is a generic, we aim to evaluate if the PLMs can use the injected knowledge to resolve that the falsified generics are wrongly quantified statements and should be classified as non-generic. This is because the injected knowledge has factual information that should contradict the falsified generics.

Table 6 reports the accuracy of zero-shot classification of universally quantified statements as non-generics, before and after knowledge injection. Knowledge injection almost doubles classification accuracy, but all the models still overwhelmingly predict that the statements are true generics. Knowledge injection leads to a bigger (absolute) improvement in accuracy for majority characteristic generics than for minority characteristic generics.

Based on this result, we postulate that the PLMs do not understand that the quantifier *all* in a noun phrase such as *all lions* implies *male + female* lions. This is made evident by the presence of quantifiers when asking to fill in the blank for a generic sentence such as "`[MASK]` lions have manes". We sampled factual knowledge from ASCENT KB that emphasizes minority and majority characteristic of kinds. This includes assertions such as "male lions have manes"; therefore, natural language inference should lead to the conclusion that "all lions have manes" cannot be a true generic statement because the factual knowledge emphasized gender differences, thus making exceptions more salient.

5.2 GOG Effect Evaluation Probing

We extend the evaluation of the effect of PLM knowledge injection on the GOG effect (§4.2) to probe whether knowledge injection enables the model to distinguish between which quantifiers make a minority or majority characteristic generics true and which quantifiers make them false. For ex-

	Quantified Minority Generics		Quantified Majority Generics	
Model	**PLM**	**PLM+KEPLER**	**PLM**	**PLM+KEPLER**
BERT	0.083	0.14	0.10	0.18
RoBERTa	0.064	0.12	0.081	0.19

Table 6: Accuracy of classifying universally quantified versions of minority and majority generic statements in the test set as false, using the PLM generics classifiers, before and after injecting factual knowledge with KEPLER.

#	BERT	RoBERTa
1	Eyes	Heads
2	Manes	Eyes
3	Heads	Manes
4	Teeth	Tails
5	Tails	Teeth

Table 7: The top 5 words predicted by BERT and RoBERTa, with injected factual knowledge, for filling the mask in the generic "most lions have `[MASK]`".

#	BERT	RoBERTa
1	Mountain	Male
2	Female	Female
3	White	Mountain
4	All	Young
5	Young	All

Table 8: The top 5 words predicted by BERT and RoBERTa, with injected factual knowledge, for filling the mask in the generic "`[MASK]` lions are animals".

ample, the token "stripes" should be ranked among the top tokens that the PLM predict when asked to fill in the mask token in the statement "most tigers have `[MASK]`". On the other hand, the PLM should *not* predict the token "stripes" when asked to fill in the blank for the statement 'few tigers have `[MASK]`".

For this probing task, we evaluate statements using four quantifiers: *few, many, most*, and *some*. We quantify both minority and majority characteristic generics and mask the final token in the statement (corresponding to the object or predicative complement). We generate probing datasets using the template: *quantifier* + (generic statement - final token) + `[MASK]`.

We report the mean reciprocal rank of both minority and majority characteristic generics for each quantifier. If the model successfully learns how to interpret each quantifier, the masked final tokens should be ranked higher together with a quantifier that makes the generic statement true and lower with the quantifier that make the generic statement false. Table 9 shows the results of this probing task.

The results show that for minority characteristic generics the PLMs correctly assign a higher masked token MRR to true statements quantified by *few* or *some* than to statements quantified by *many* or *most*. Conversely, for majority characteristic generics the PLMs also correctly assign higher MRR when statements are quantified by *many* or *most* instead of with *few* or *some*. However, statements with the quantifier *some* are still ranked relatively high, indicating that the PLMs struggle more to interpret that quantifier correctly.

The MRR for masked tokens in true statements is higher for majority characteristics than minority characteristics, but the MRR for false statements is relatively lower on minority characteristics. Despite learning the distinction between different kinds of quantifiers, the MRR across the models and quantifiers is arguably still too high with quantifiers that falsify a generic statement.

As an example, Table 7 shows the tokens predicted for the statement "most lions have `[MASK]`". Here we expect the original token, "manes", not to feature among the top predicted tokens for the quantifier *most* because the injected factual data should make salient the knowledge that it is only a minority population of lions that have manes. However the two PLMs still rank "manes" as second and third most likely token, respectively. In contrast, when the first token in the statement is masked, i.e., "`[MASK]` lions are animals", the only quantifier in the top 5 is *all*, at position 4 and 5.

6 Conclusion

We investigated the generic overgeneralization (GOG) effect in PLMs and demonstrated that PLMs do overgeneralize and treat quantified statements as if they were generics. We introduced datasets on minority and majority characteristic generics that

	Minority Generics with Quantifier				Majority Generics with Quantifier			
Model	**Few**	**Some**	Many	Most	Few	Some	**Many**	**Most**
BERT	0.58	0.69	0.43	0.45	0.51	0.65	0.71	0.74
RoBERTa	0.61	0.70	0.38	0.40	0.49	0.63	0.76	0.80

Table 9: Mean Reciprocal Rank (MRR) scores of masked final tokens using PLMs with knowledge injection under different quantifiers. Scores indicate how each model perform on the probing task for each quantifier when applied to minority and majority characteristic generics.

can be used to evaluate the GOG effect, as well as a source of factual knowledge about kinds to evaluate PLM knowledge embedding methods. Our results suggest that knowledge injection reduces the GOG effect in PLMs but does not eliminate it, and that majority characteristic generic statements are more susceptible to overgeneralization bias. Probing the models after knowledge injection, we were able to determine which quantifiers make minority or majority characteristic generics to remain as true quantified generic statements and which quantifiers make the generics to become non-generic statements.

Our paper makes the case for future research on methods for injecting commonsense into PLMs more effectively so that they can perform better natural language inference based on the knowledge presented. This would be an important step towards advancing commonsense reasoning in PLMs.

Acknowledgement

This work is based on research supported in part by the National Research Foundation of South Africa (Grant Number: 129850) and the Hasso Plattner Institute for Digital Engineering through the HPI Research School at the University of Cape Town.

References

Sumithra Bhakthavatsalam, Chloe Anastasiades, and Peter Clark. 2020. GenericsKB: A knowledge base of generic statements. *CoRR*, abs/2005.00660.

Samuel R. Bowman, Gabor Angeli, Christopher Potts, and Christopher D. Manning. 2015. A large annotated corpus for learning natural language inference. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing, EMNLP 2015, Lisbon, Portugal, September 17-21, 2015*, pages 632–642. The Association for Computational Linguistics.

Yi-Jyun Chen, Ching-Yu Helen Yang, and Jason S. Chang. 2020. Improving phrase translation based on sentence alignment of Chinese-English parallel corpus. In *Proceedings of the 32nd Conference on Computational Linguistics and Speech Processing (ROCLING 2020)*, pages 6–7, Taipei, Taiwan. The Association for Computational Linguistics and Chinese Language Processing (ACLCLP).

Kevin Clark, Urvashi Khandelwal, Omer Levy, and Christopher D. Manning. 2019. What does BERT look at? an analysis of BERT's attention. In *Proceedings of the 2019 ACL Workshop BlackboxNLP: Analyzing and Interpreting Neural Networks for NLP*, pages 276–286, Florence, Italy. Association for Computational Linguistics.

Yuanfei Dai, Shiping Wang, Neal N. Xiong, and Wenzhong Guo. 2020. A survey on knowledge graph embedding: Approaches, applications and benchmarks. *Electronics*, 9(5).

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, Minneapolis, Minnesota. Association for Computational Linguistics.

Annemarie Friedrich and Manfred Pinkal. 2015. Discourse-sensitive automatic identification of generic expressions. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing of the Asian Federation of Natural Language Processing, ACL 2015, July 26-31, 2015, Beijing, China, Volume 1: Long Papers*, pages 1272–1281. The Association for Computer Linguistics.

Peizhu Gong, Jin Liu, Yihe Yang, and Huihua He. 2020. Towards knowledge enhanced language model for machine reading comprehension. *IEEE Access*, 8:224837–224851.

Venkata Subrahmanyan Govindarajan, Benjamin Van Durme, and Aaron Steven White. 2019. Decomposing generalization: Models of generic, habitual and episodic statements. *Trans. Assoc. Comput. Linguistics*, 7:501–517.

Michelle A. Hollander, Susan A. Gelman, and Jon Star. 2002. Children's interpretation of generic noun phrases. *Developmental Psychology*, 38(6):883–894.

Daniel Karczewski, Edyta Wajda, and Radosław Poniat. 2020. Do all storks fly to africa? universal statements and the generic overgeneralization effect. *Lingua*, 246:102855.

Sangeet Khemlani, Sarah-Jane Leslie, Sam Glucksberg, and Paula Rubio Fernandez. 2007. Do ducks lay eggs? How people interpret generic assertions. *Proceedings of the Annual Meeting of the Cognitive Science Society*, 29.

Dimitra Lazaridou-Chatzigoga. 2019. Genericity. In Chris Cummins and Napoleon Katsos, editors, *The Oxford Handbook of Experimental Semantics and Pragmatics*, pages 155–177. Oxford University Press.

Sarah-Jane Leslie, Sangeet Khemlani, and Sam Glucksberg. 2011. Do all ducks lay eggs? The generic overgeneralization effect. *Journal of Memory and Language*, 65(1):15–31.

Nelson F. Liu, Matt Gardner, Yonatan Belinkov, Matthew E. Peters, and Noah A. Smith. 2019a. Linguistic knowledge and transferability of contextual representations. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 1073–1094, Minneapolis, Minnesota. Association for Computational Linguistics.

Y. Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, M. Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019b. RoBERTa: A robustly optimized BERT pretraining approach. *ArXiv*, abs/1907.11692.

Yuzhang Liu, Peng Wang, Yingtai Li, Yizhan Shao, and Zhongkai Xu. 2020. AprilE: Attention with pseudo residual connection for knowledge graph embedding. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 508–518, Barcelona, Spain (Online). International Committee on Computational Linguistics.

Peter LoBue and Alexander Yates. 2011. Types of common-sense knowledge needed for recognizing textual entailment. In *The 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies, Proceedings of the Conference, 19-24 June, 2011, Portland, Oregon, USA - Short Papers*, pages 329–334. The Association for Computer Linguistics.

Todor Mihaylov and Anette Frank. 2018. Knowledgeable reader: Enhancing cloze-style reading comprehension with external commonsense knowledge. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics, ACL 2018, Melbourne, Australia, July 15-20, 2018, Volume 1: Long Papers*, pages 821–832. Association for Computational Linguistics.

Sean Monahan, Michael Mohler, Marc T. Tomlinson, Amy Book, Maxim Gorelkin, Kevin Crosby, and Mary Brunson. 2015. Populating a knowledge base with information about events. *Theory and Applications of Categories*.

Tuan-Phong Nguyen, Simon Razniewski, and Gerhard Weikum. 2021. Advanced semantics for commonsense knowledge extraction. In *WWW '21: Proceedings of the Web Conference 2021*.

Guanglin Niu, Bo Li, Yongfei Zhang, Shiliang Pu, and Jingyang Li. 2020. AutoETER: Automated entity type representation for knowledge graph embedding. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 1172–1181, Online. Association for Computational Linguistics.

Sandeep Prasada, Sangeet Khemlani, Sarah-Jane Leslie, and Sam Glucksberg. 2013. Conceptual distinctions amongst generics. *Cognition*, 126(3):405–422.

Hannah Rashkin, Antoine Bosselut, Maarten Sap, Kevin Knight, and Yejin Choi. 2018a. Modeling naive psychology of characters in simple commonsense stories. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics, ACL 2018, Melbourne, Australia, July 15-20, 2018, Volume 1: Long Papers*, pages 2289–2299. Association for Computational Linguistics.

Hannah Rashkin, Maarten Sap, Emily Allaway, Noah A. Smith, and Yejin Choi. 2018b. Event2mind: Commonsense inference on events, intents, and reactions. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics, ACL 2018, Melbourne, Australia, July 15-20, 2018, Volume 1: Long Papers*, pages 463–473. Association for Computational Linguistics.

Nils Reiter and Anette Frank. 2010. Identifying generic noun phrases. In *ACL 2010, Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics, July 11-16, 2010, Uppsala, Sweden*, pages 40–49. The Association for Computer Linguistics.

Anna Rogers, Olga Kovaleva, and Anna Rumshisky. 2020. A primer in BERTology: What we know about how BERT works. *Transactions of the Association for Computational Linguistics*, 8:842–866.

Yun Tang, Jing Huang, Guangtao Wang, Xiaodong He, and Bowen Zhou. 2020. Orthogonal relation transforms with graph context modeling for knowledge graph embedding. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 2713–2722, Online. Association for Computational Linguistics.

Ruize Wang, Duyu Tang, Nan Duan, Zhongyu Wei, Xuanjing Huang, Jianshu Ji, Guihong Cao, Daxin Jiang, and Ming Zhou. 2021a. K-Adapter: Infusing Knowledge into Pre-Trained Models with Adapters. In *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*, pages 1405–1418, Online. Association for Computational Linguistics.

Xiaozhi Wang, Tianyu Gao, Zhaocheng Zhu, Zhengyan Zhang, Zhiyuan Liu, Juanzi Li, and Jian Tang. 2021b. KEPLER: A Unified Model for Knowledge Embedding and Pre-trained Language Representation. *Transactions of the Association for Computational Linguistics*, 9:176–194.

Donghan Yu, Chenguang Zhu, Yiming Yang, and Michael Zeng. 2022. Jaket: Joint pre-training of knowledge graph and language understanding. In *AAAI 2022*.

Mengfei Zhou, Anette Frank, Annemarie Friedrich, and Alexis Palmer. 2015. Semantically enriched models for modal sense classification. In *LSDSem@EMNLP*.

How About Time?
Probing a Multilingual Language Model for Temporal Relations

Tommaso Caselli[♥], **Irene Dini**[♣], **Felice Dell'Orletta**[♣]
[♥]CLCG, University of Groningen [♣]ItaliaNLP Lab, ILC-CNR "Antonio Zampolli" Pisa
t.caselli@rug.nl, {irene.dini, felice.dellorletta}@ilc.cnr.it

Abstract

This paper presents a comprehensive set of probing experiments using a multilingual language model, XLM-R, for temporal relation classification between events in four languages. Results show an advantage of contextualized embeddings over static ones and a detrimental role of sentence level embeddings. While obtaining competitive results against state-of-the-art systems, our probes indicate a lack of suitable encoded information to properly address this task.

1 Introduction

Time is a pervasive element of human life with no counterpart in any other cognitive domains (Bonomi and Zucchi, 2001). Such pervasiveness is mirrored in natural languages through sets of devices that allows speakers to refer to time, to reason about time and things that unfold in time. Reasoning about time is one of the central components of common sense knowledge (Pianesi and Varzi, 1996; Boyd, 2010; Geva et al., 2021) and its modeling has been at the core of many early approaches in Computational Linguistics and Artificial Intelligence (Schank and Abelson, 1975; McDermott, 1982; Allen, 1984; Passonneau, 1988; Moens and Steedman, 1988). More recently, specific Natural Language Understanding (NLU) tasks related to time have been developed, ranging from the identification of temporal expressions (Mani et al., 2001; Mazur and Dale, 2010), to measuring the duration of events (Pan et al., 2006b,a; Zhou et al., 2019), and the ability to order them chronologically(Mani et al., 2003; UzZaman and Allen, 2010; Ning et al., 2018; Wen et al., 2021). More complex tasks have challenged models to extract storylines (Chambers and Jurafsky, 2008; Minard et al., 2015; Caselli and Inel, 2018), understand narratives (Mostafazadeh et al., 2017, 2020; Lal et al., 2021), and answer temporally related questions (Llorens et al., 2015; Ning et al., 2020). Recent work has focused on recasting temporal relation classification as a Natural Language Inference (NLI) task where fine-tuned pre-trained language models (PTLMs) have achieved good results (Vashishtha et al., 2020).

Embedding representations, both static and contextual, have shown to play a key role to improve systems' results on different time-related benchmarks, especially for the classification of temporal relations between pairs of events (Mirza and Tonelli, 2016; Cheng et al., 2020). When it comes to contextualized embeddings the probing of such models for temporal knowledge has not been properly investigated yet. If we embrace the vision of PTLMs as large repositories of linguistic knowledge (Derby et al., 2021; Mosbach et al., 2020; Miaschi et al., 2020), it is a natural question to probe these models for their knowledge about events and time. We present an extensive study on temporal relation probing of PTLMs using five temporally annotated corpora in four languages (i.e., English, French, Spanish, and Italian). Although the selected languages all belong to Indo-European family, they present differences for the tense-mood-aspect (TMA) system while showing similarities at the lexico-pragmatic level. Our probing tasks focus on *temporal ordering of pairs of events* **(E–E)**, either in the same sentence or in different ones.

Our contributions Our work has three contributions: (i) it is the first work to probe a multilingual PTLM, XLM-R base, for temporal knowledge between event pairs; (ii) we study the impact of multilingual contextualized representations against monolingual counterparts based on static word embeddings; (iii) we compare zero-shot PTLM against fine-tuned models for temporal reasoning to investigate whether the models have acquired real temporal knowledge. Code and data are available.[1]

[1] https://github.com/irenedini/tlink_probing

Proceedings of the 29th International Conference on Computational Linguistics, pages 3197–3209
October 12–17, 2022.

2 Data Overview

We have selected five corpora annotated with language specific adaptations of ISO-TimeML (Pustejovsky et al., 2010). ISO-TimeML is, at the same time, an annotation meta-model for marking events, temporal expressions, and relations between them, and a full-fledged annotation language. ISO-TimeML has 13 values used to classify temporal relations, based on Allen's interval temporal logic (Allen, 1983) where each value expresses how an event chronologically relates to another event or a temporal expression. In the following paragraphs we present a short overview of the five corpora we have used. For our experiments, we have extracted all temporal relations between event pairs, either occurring in the same sentence or in difference sentences. Table 1 presents a summary of the temporal relations between events for each corpus.

EN-TimeBank The English TimeBank (Pustejovsky et al., 2003) is a corpus of 183 documents manually annotated following the TimeML annotation guidelines (Saurı et al., 2006). The whole corpus has gone through a curation phase for the SemEval 2013 TempEval-3 task (UzZaman et al., 2013), where an extra test set of 20 documents has been annotated with the same guidelines. In our experiments, we follow the TempEval-3 split for training and test distributions, excluding the automatically annotated data (i.e., silver data distribution). EN-TimeBank uses the full 13 temporal values from ISO-TimeML to classify event-event relations.

IT-TimeBank The Italian TimeBank (Caselli et al., 2011) has 254 documents, comparable in size and annotation to the EN-TimeBank. We have followed the official split into train and test from the EVALITA 2014 EVENTI task (Caselli et al., 2014). Similarly to EN-TimeBank, the 13 fine-grained temporal values have been used to classify temporal relations.

FR-TimeBank French TimeBank (Bittar et al., 2011) is a corpus of 107 documents in French annotated following an adaptation to French of TimeML. The corpus does not present an official split into train and test. To obviate to this, we have first extracted all temporally annotated event pairs and then created a train and test distribution following a 75-25 split. FR-TimeBank also uses the full 13 temporal values for classifying temporal relations

EN-TB-Dense The TimeBank-Dense corpus (Cassidy et al., 2014) contains only 36 documents from the training portion of the EN-TimeBank. EN-TB-Dense approximates a complete graph of all possible temporal relations over events and temporal expressions by labeling all pairs locally, i.e., same sentence and adjacent sentence pairs. EN-TB-Dense simplifies the set of possible temporal relation values by reducing it to five and introducing a new value, VAGUE, for all relations that do not carry a clear semantics.

ES-TimeBank The Spanish TimeBank (Saurı and Badia, 2012) contains 210 documents in Spanish. We have followed the official release into train and test splits. Similarly to the EN-TimeBank-Dense, the authors have simplified the set of possible temporal relations to five plus VAGUE. However, the overlap is limited only to BEFORE AFTER, with the other three being new.

3 Temporal Probing

Our probing task investigates the capabilities of PTLMs to encode information about events and their temporal ordering. To probe such information across multiple languages, we use `XLM-R base` (Conneau et al., 2020),[2] a large multilingual model that has achieved state-of-the-art results on many NLU tasks. Following previous work (Tenney et al., 2019; Jawahar et al., 2019; Vulić et al., 2020; de Vries et al., 2020, *inter alia*), we extract embedding representations from each layer and use them to train a linear SVM whose objective is to predict the value of a temporal relation between a given pair of events. By default, we feed the SVM with four concatenated embeddings: the embeddings of the sentence containing each event in the pair and those of each event. In case the event pair occurs in the same sentence, we duplicate the sentence representation. Sentences are represented by averaging the embeddings of the tokens excluding `XLM-R base`'s special tokens.

We compare the default settings with three variations: (i) we use only the embeddings of the events in the pair; (ii) we use the embeddings from two `XLM-R base` models previously fine-tuned with the EN-TimeBank and EN-TB-Dense

[2] https://huggingface.co/xlm-roberta-base

Temporal Relation	EN-TimeBank Train	EN-TimeBank Test	IT-TimeBank Train	IT-TimeBank Test	FR-TimeBank Train	FR-TimeBank Test	EN-TB-Dense Train	EN-TB-Dense Test	ES-TimeBank Train	ES-TimeBank Test
BEFORE	180	83	167	38	107	32	884	378	834	62
AFTER	184	90	79	15	36	10	729	275	499	47
INCLUDES	67	15	47	19	2	3	207	57	–	–
IS_INCLUDED	64	29	46	28	2	1	265	52	–	–
DURING	4	0	0	0	125	30	–	–	–	–
SIMULTANEOUS	132	45	131	46	26	9	72	22	–	–
IMM_BEFORE	11	1	2	1	6	1	–	–	–	–
IMM_AFTER	5	1	3	0	0	1	–	–	–	–
BEGINS	11	0	0	0	15	5	–	–	–	–
BEGUN_BY	11	0	1	0	5	1	–	–	–	–
ENDS	4	1	0	1	3	2	–	–	–	–
ENDED_BY	21	0	2	0	7	1	–	–	–	–
IDENTITY	140	15	217	50	78	42	–	–	–	–
OVERLAP	–	–	–	–	–	–	–	–	4,478	307
BEFORE_OVERLAP	–	–	–	–	–	–	–	–	907	74
OVERLAP_AFTER	–	–	–	–	–	–	–	–	336	26
VAGUE	–	–	–	–	–	–	1,995	634	29	5

Table 1: Summary of the distribution of the temporal relations between pairs of events in all five corpora. For the EN-TB-Dense, the values for the training are obtained by merging together the training and the development sets.

corpora recasted in forms of Natural Language Inference pairs for temporal reasoning as described in Vashishtha et al. (2020); (iii) we use monolingual static word embeddings obtained with the `word2vec skip-gram` (w2v) model (Mikolov et al., 2013) (see Appendix A for details). Lastly, all probing variations are compared with a dummy classifier predicting the majority class in each corpus.

4 Results

Figure 1 summarizes the results for all corpora and settings. Details per corpus are in Appendix B.

Although all probing models outperform their respective baselines, our results further confirm that temporal relation classification is a challenging task. `XLM-R` embeddings consistently obtain the best results across all languages and granularities of the temporal relations, improving the performance of static embeddings. With the exclusion of EN-TB-Dense, the presence of sentence embeddings is detrimental, confirming previous findings (Miaschi and Dell'Orletta, 2020). Although temporal relations are a discourse phenomenon at the interface of the semantics and pragmatics dimensions, it appears that the event only embeddings from `XLM-R` already store sufficient semantic information to perform this task.

Regardless of the granularity of the temporal relations, it clearly emerges from all the plots that the best results for `XLM-R` are obtained between layers 6 and 8. Performances are consistently sub-optimal for early layers, especially 1–4. For higher layers, i.e., 10–12, results are disappointing, with the exception for English whose best probe is at layer 11.Given the task and previous findings on the encoding of linguistic knowledge in PTLMs (Tenney et al., 2019; Jawahar et al., 2019), this is not fully expected. Ideally, if PTLMs tend to encoded more semantic features in the top layers, performances for this task should not degrade on the top layers, as we see for Spanish, Italian, and French, or, at least, they should remain on a plateau.

A further finding concerns the role of fine-tuned models for temporal reasoning, namely `XLM-R_tbd` and `XLM-R_tb`. The models have been fined-tuned using the English corpora EN-TB-Dense and EN-TimeBank recasted for temporal reasoning. We expected the embeddings from these models to be more competitive than basic `XLM-R`, but this is not the case. In general, we observe a better performance for `XLM-R_tbd` than `XLM-R_tb`, in-line with the results reported by Vashishtha et al. (2020). The better results of `XLM-R_tbd` hold also in cross-lingual settings, regardless of the granularity of the temporal values used in the specific corpus.

When comparing results across corpora, two dimensions are at play: the first is the granularity of the temporal values; the second is the number of training examples. A general pattern we observe is the following: the less temporal values are to be learned, the better the results of a trained model, provided that the annotated data are consis-

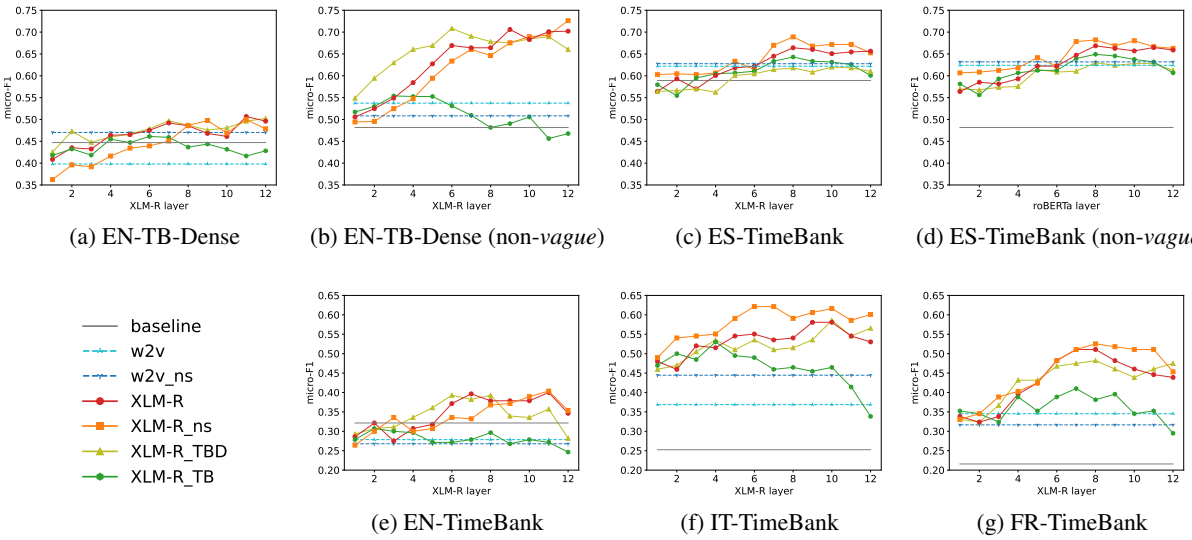

(a) EN-TB-Dense (b) EN-TB-Dense (non-*vague*) (c) ES-TimeBank (d) ES-TimeBank (non-*vague*)

(e) EN-TimeBank (f) IT-TimeBank (g) FR-TimeBank

Figure 1: Overview of the probing results across all corpora. For each plot, the *x*-axis reports the layer id of `XLM-R base`, the *y*-axis reports the micro-F1. In the legend on the bottom left side, `ns` stands for "no sentence"; `tbd` and `tb` refer to the fine-tuned `XLM-R base` models for temporal reasoning: `tbd` stands for EN-TB-Dense, `tb` stands for EN-TimeBank.

tent. To better illustrate this, we focus our analysis on the EN-TB-Dense and ES-TimeBank first. Both corpora adopt coarse grained temporal values and have the largest number of annotated data. Nevertheless, the way the value VAGUE is used in the two corpora is not the same. In EN-TB-Dense, given the specific annotation framework, VAGUE is used both in case of an existing temporal relation with an unclear semantics but also for event pairs with no temporal relation. This is not the case in Spanish. Such a difference is mirrored in the results: as soon as we remove VAGUE, scores in the EN-TB-Dense improve while they remain the same in ES-TimeBank. When considering the corpora with fine-grained temporal values, we observe that the F1-score in EN-TimeBank is the lowest (in absolute terms), while results are much better for IT-TimeBank and FR-TimeBank. The differences in this case can only be due to a more consistent application of the annotation guidelines in Italian and French than in English. Support to this claim can be found in the fact that Italian and French have a lower number of sentences in training, 695 and 412 respectively, than English, namely 834. To gain insights, we have analyzed the overlapping events between Train and Test splits, i.e., how many times the same event appears in Train and Test, even if coupled with a different event and with a different temporal value. While FR-TimeBank has the largest overlap (58%), IT-TimeBank has the lowest (29%) and EN-TimeBank is in the middle (35%). If it was just a matter of data, we would expect the EN-TimeBank to obtain better F1-scores than IT-TimeBank.

Comparison with state of the art is limited to EN-TB-Dense and IT-TimeBank. No previous work for this task is available FR-TimeBank and ES-TimeBank, and for EN-TimeBank we only have access to systems which classified temporal relations from raw text. Concerning the EN-TB-Dense, the best system, SECT (Cheng et al., 2020), adopts a multi-task learning approach using a GRU architecture. On the **E–E** classification it achieves an F1-score 0.650, gaining 0.098 points with respect to our best training layer. A more similar architecture, CATENA (Mirza and Tonelli, 2016) a linear SVM combining pre-trained word embeddings and additional features, obtains an F1-score of 0.519, only 0.012 points above us. As for Italian, the best system, FBK-HTL-time (Mirza and Minard, 2014), a feature-based linear SVM, achieves an F1-score of 0.688, beating our approach of 0.062 points.

Statistical significance testing We also performed statistical significance tests across all the probing systems using the McNemar's test. We ran the significant tests using two different settings: first by considering the last embedding layer of the PTLMs and subsequently the embedding layer that gave the best results for each probing. Details can found in Appendix C.

All probing experiments are consistently significant when compared to their respective baselines, with the exclusion of the EN-TimeBank corpus.

When focusing on the differences between the PTLM embeddings and the static ones, the results are more scattered, with different behaviors across each dataset. We observe significant differences in the majority of cases when sentence representations are excluded from the static embeddings. Two datasets, ES-TimeBank and IT-TimeBank, present peculiar behaviors when compared to the others. For the ES-TimeBank, probes with PTLM embeddings tend to be not statistically significant with respect to the static ones. The opposite trend, on the contrary, can be observed for the IT-TimeBank.

Finally, across all the PTLM probes, a clear tendency that emerges is that significant differences can be observed only when using `XML-R_tb` embeddings, while only in few cases the significant difference can be observed when using the `XML-R_tbd` embeddings.

5 Conclusion

This paper investigates the knowledge encoded in a large multilingual PTLM, `XLM-R base`, for temporal relation classification between pairs of events in four languages and five corpora with varying granularities of temporal values. Our results point out that temporal relation classification between events is very challenging and the linguistic knowledge in `XLM-R` is limited to properly address it. While contextual embeddings are more "powerful" than static ones, current fine-tuned models for temporal reasoning (Vashishtha et al., 2020) are not helpful as one would expect. Our probes indicate that adding more information, i.e., sentence representations, to lexical entities is detrimental, meaning that "global" semantic information is already encoded at the lexical level. Finally, our models are competitive with state-of-the-art systems, indicating that improvements are due either to specific architectures or extra features capturing additional knowledge not available in the contextual embeddings.

Acknowledgements

The authors acknowledge the project "Human in Neural Language Models" (*IsC93_HiNLM*), funded by CINECA[3] under the ISCRA initiative and the Peregrine High Performance Cluster of the University of Groningen for the availability of computing resources and support.

[3]https://www.cineca.it/en

References

James F. Allen. 1983. Maintaining knowledge about temporal intervals. *Communications of the ACM*, 26(11):832–843.

James F Allen. 1984. Towards a general theory of action and time. *Artificial intelligence*, 23(2):123–154.

André Bittar, Pascal Amsili, Pascal Denis, and Laurence Danlos. 2011. French TimeBank: An ISO-TimeML annotated reference corpus. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, pages 130–134, Portland, Oregon, USA. Association for Computational Linguistics.

Andrea Bonomi and Alessandro Zucchi. 2001. *Tempo e linguaggio: introduzione alla semantica del tempo e dell'aspetto verbale*. Pearson Italia Spa.

Brian Boyd. 2010. *On the origin of stories: Evolution, cognition, and fiction*. Harvard University Press.

Tommaso Caselli, Valentina Bartalesi Lenzi, Rachele Sprugnoli, Emanuele Pianta, and Irina Prodanof. 2011. Annotating events, temporal expressions and relations in Italian: the it-timeml experience for the ita-TimeBank. In *Proceedings of the 5th Linguistic Annotation Workshop*, pages 143–151, Portland, Oregon, USA. Association for Computational Linguistics.

Tommaso Caselli and Oana Inel. 2018. Crowdsourcing StoryLines: Harnessing the crowd for causal relation annotation. In *Proceedings of the Workshop Events and Stories in the News 2018*, pages 44–54, Santa Fe, New Mexico, U.S.A. Association for Computational Linguistics.

Tommaso Caselli, Rachele Sprugnoli, Manuela Speranza, and Monica Monachini. 2014. Eventi: Evaluation of events and temporal information at evalita 2014. *EVENTI: EValuation of Events and Temporal INformation at Evalita 2014*, pages 27–34.

Taylor Cassidy, Bill McDowell, Nathanael Chambers, and Steven Bethard. 2014. An annotation framework for dense event ordering. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 501–506, Baltimore, Maryland. Association for Computational Linguistics.

Nathanael Chambers and Dan Jurafsky. 2008. Unsupervised learning of narrative event chains. In *Proceedings of ACL-08: HLT*, pages 789–797, Columbus, Ohio. Association for Computational Linguistics.

Fei Cheng, Masayuki Asahara, Ichiro Kobayashi, and Sadao Kurohashi. 2020. Dynamically updating event

representations for temporal relation classification with multi-category learning. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 1352–1357, Online. Association for Computational Linguistics.

Alexis Conneau, Kartikay Khandelwal, Naman Goyal, Vishrav Chaudhary, Guillaume Wenzek, Francisco Guzmán, Edouard Grave, Myle Ott, Luke Zettlemoyer, and Veselin Stoyanov. 2020. Unsupervised cross-lingual representation learning at scale. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 8440–8451, Online. Association for Computational Linguistics.

Wietse de Vries, Andreas van Cranenburgh, and Malvina Nissim. 2020. What's so special about BERT's layers? a closer look at the NLP pipeline in monolingual and multilingual models. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 4339–4350, Online. Association for Computational Linguistics.

Steven Derby, Paul Miller, and Barry Devereux. 2021. Representation and pre-activation of lexical-semantic knowledge in neural language models. In *Proceedings of the Workshop on Cognitive Modeling and Computational Linguistics*, pages 211–221, Online. Association for Computational Linguistics.

Mor Geva, Daniel Khashabi, Elad Segal, Tushar Khot, Dan Roth, and Jonathan Berant. 2021. Did aristotle use a laptop? a question answering benchmark with implicit reasoning strategies. *Transactions of the Association for Computational Linguistics*, 9:346–361.

Ganesh Jawahar, Benoît Sagot, and Djamé Seddah. 2019. What does BERT learn about the structure of language? In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 3651–3657, Florence, Italy. Association for Computational Linguistics.

Yash Kumar Lal, Nathanael Chambers, Raymond Mooney, and Niranjan Balasubramanian. 2021. TellMeWhy: A dataset for answering why-questions in narratives. In *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*, pages 596–610, Online. Association for Computational Linguistics.

Hector Llorens, Nathanael Chambers, Naushad UzZaman, Nasrin Mostafazadeh, James Allen, and James Pustejovsky. 2015. SemEval-2015 task 5: QA TempEval - evaluating temporal information understanding with question answering. In *Proceedings of the 9th International Workshop on Semantic Evaluation (SemEval 2015)*, pages 792–800, Denver, Colorado. Association for Computational Linguistics.

Inderjeet Mani, Barry Schiffman, and Jianping Zhang. 2003. Inferring temporal ordering of events in news. In *Companion Volume of the Proceedings of HLT-NAACL 2003 - Short Papers*, pages 55–57.

Inderjeet Mani, George Wilson, Lisa Ferro, and Beth Sundheim. 2001. Guidelines for annotating temporal information. In *Proceedings of the First International Conference on Human Language Technology Research*.

Pawel Mazur and Robert Dale. 2010. WikiWars: A new corpus for research on temporal expressions. In *Proceedings of the 2010 Conference on Empirical Methods in Natural Language Processing*, pages 913–922, Cambridge, MA. Association for Computational Linguistics.

Drew McDermott. 1982. A temporal logic for reasoning about processes and plans. *Cognitive science*, 6(2):101–155.

Alessio Miaschi, Dominique Brunato, Felice Dell'Orletta, and Giulia Venturi. 2020. Linguistic profiling of a neural language model. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 745–756, Barcelona, Spain (Online). International Committee on Computational Linguistics.

Alessio Miaschi and Felice Dell'Orletta. 2020. Contextual and non-contextual word embeddings: an in-depth linguistic investigation. In *Proceedings of the 5th Workshop on Representation Learning for NLP*, pages 110–119, Online. Association for Computational Linguistics.

Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. 2013. Distributed representations of words and phrases and their compositionality. *Advances in neural information processing systems*, 26.

Anne-Lyse Minard, Manuela Speranza, Eneko Agirre, Itziar Aldabe, Marieke van Erp, Bernardo Magnini, German Rigau, and Rubén Urizar. 2015. SemEval-2015 task 4: TimeLine: Cross-document event ordering. In *Proceedings of the 9th International Workshop on Semantic Evaluation (SemEval 2015)*, pages 778–786, Denver, Colorado. Association for Computational Linguistics.

Paramita Mirza and Anne-Lyse Minard. 2014. Fbk-hlt-time: a complete italian temporalprocessing system for eventi-evalita 2014. *FBK-HLT-time: a complete Italian TemporalProcessing system for EVENTI-Evalita 2014*, pages 44–49.

Paramita Mirza and Sara Tonelli. 2016. On the contribution of word embeddings to temporal relation classification. In *Proceedings of COLING 2016, the 26th International Conference on Computational Linguistics: Technical Papers*, pages 2818–2828, Osaka, Japan. The COLING 2016 Organizing Committee.

Marc Moens and Mark Steedman. 1988. Temporal ontology and temporal reference. *Computational Linguistics*, 14(2):15–28.

Marius Mosbach, Stefania Degaetano-Ortlieb, Marie-Pauline Krielke, Badr M. Abdullah, and Dietrich Klakow. 2020. A closer look at linguistic knowledge in masked language models: The case of relative clauses in American English. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 771–787, Barcelona, Spain (Online). International Committee on Computational Linguistics.

Nasrin Mostafazadeh, Aditya Kalyanpur, Lori Moon, David Buchanan, Lauren Berkowitz, Or Biran, and Jennifer Chu-Carroll. 2020. GLUCOSE: GeneraLized and COntextualized story explanations. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 4569–4586, Online. Association for Computational Linguistics.

Nasrin Mostafazadeh, Michael Roth, Annie Louis, Nathanael Chambers, and James Allen. 2017. LSDSem 2017 shared task: The story cloze test. In *Proceedings of the 2nd Workshop on Linking Models of Lexical, Sentential and Discourse-level Semantics*, pages 46–51, Valencia, Spain. Association for Computational Linguistics.

Qiang Ning, Hao Wu, Rujun Han, Nanyun Peng, Matt Gardner, and Dan Roth. 2020. TORQUE: A reading comprehension dataset of temporal ordering questions. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1158–1172, Online. Association for Computational Linguistics.

Qiang Ning, Hao Wu, and Dan Roth. 2018. A multi-axis annotation scheme for event temporal relations. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1318–1328, Melbourne, Australia. Association for Computational Linguistics.

Feng Pan, Rutu Mulkar, and Jerry R. Hobbs. 2006a. Extending TimeML with typical durations of events. In *Proceedings of the Workshop on Annotating and Reasoning about Time and Events*, pages 38–45, Sydney, Australia. Association for Computational Linguistics.

Feng Pan, Rutu Mulkar, and Jerry R. Hobbs. 2006b. Learning event durations from event descriptions. In *Proceedings of the 21st International Conference on Computational Linguistics and 44th Annual Meeting of the Association for Computational Linguistics*, pages 393–400, Sydney, Australia. Association for Computational Linguistics.

Rebecca J. Passonneau. 1988. A computational model of the semantics of tense and aspect. *Computational Linguistics*, 14(2):44–60.

Fabio Pianesi and Achille C Varzi. 1996. Events, topology and temporal relations. *The monist*, 79(1):89–116.

James Pustejovsky, Patrick Hanks, Roser Sauri, Andrew See, Robert Gaizauskas, Andrea Setzer, Dragomir Radev, Beth Sundheim, David Day, Lisa Ferro, et al. 2003. The timebank corpus. In *Corpus linguistics*, volume 2003, page 40. Lancaster, UK.

James Pustejovsky, Kiyong Lee, Harry Bunt, and Laurent Romary. 2010. ISO-TimeML: An international standard for semantic annotation. In *Proceedings of the Seventh International Conference on Language Resources and Evaluation (LREC'10)*, Valletta, Malta. European Language Resources Association (ELRA).

Roser Saurı and Toni Badia. 2012. Spanish TimeBank 1.0 Corpus documentation.

Roser Saurı, Jessica Littman, Bob Knippen, Robert Gaizauskas, Andrea Setzer, and James Pustejovsky. 2006. TimeML Annotation Guidelines version 1.2.1.

Roger C Schank and Robert P Abelson. 1975. Scripts, plans, and knowledge. In *IJCAI*, volume 75, pages 151–157.

Ian Tenney, Dipanjan Das, and Ellie Pavlick. 2019. BERT rediscovers the classical NLP pipeline. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 4593–4601, Florence, Italy. Association for Computational Linguistics.

Naushad UzZaman and James Allen. 2010. TRIPS and TRIOS system for TempEval-2: Extracting temporal information from text. In *Proceedings of the 5th International Workshop on Semantic Evaluation*, pages 276–283, Uppsala, Sweden. Association for Computational Linguistics.

Naushad UzZaman, Hector Llorens, Leon Derczynski, James Allen, Marc Verhagen, and James Pustejovsky. 2013. SemEval-2013 task 1: TempEval-3: Evaluating time expressions, events, and temporal relations. In *Second Joint Conference on Lexical and Computational Semantics (*SEM), Volume 2: Proceedings of the Seventh International Workshop on Semantic Evaluation (SemEval 2013)*, pages 1–9, Atlanta, Georgia, USA. Association for Computational Linguistics.

Siddharth Vashishtha, Adam Poliak, Yash Kumar Lal, Benjamin Van Durme, and Aaron Steven White. 2020. Temporal reasoning in natural language inference. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 4070–4078, Online. Association for Computational Linguistics.

Ivan Vulić, Edoardo Maria Ponti, Robert Litschko, Goran Glavaš, and Anna Korhonen. 2020. Probing pretrained language models for lexical semantics. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 7222–7240, Online. Association for Computational Linguistics.

Haoyang Wen, Yanru Qu, Heng Ji, Qiang Ning, Jiawei Han, Avi Sil, Hanghang Tong, and Dan Roth. 2021. Event time extraction and propagation via graph attention networks. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 62–73, Online. Association for Computational Linguistics.

Ben Zhou, Daniel Khashabi, Qiang Ning, and Dan Roth. 2019. "going on a vacation" takes longer than "going for a walk": A study of temporal commonsense understanding. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 3363–3369, Hong Kong, China. Association for Computational Linguistics.

A Monolingual Static Embeddings

All monolingual static word embeddings have been taken from this repository: https://vectors.nlpl.eu/repository/ . We have used the versions from the 2017 CoNLL shared task.

B Probes Models: Detailed Results

The following Tables presents detail results for each corpus from our experiments. We have highlighted in green the best results for each corpus across all models. For each probe model, we have highlighted in bold the best results. Scores correspond to micro-F1.

Model	Layer score 1	2	3	4	5	6	7	8	9	10	11	12	Score
XLM-R	0.409	0.435	0.432	0.464	0.465	0.475	0.492	0.486	0.468	0.461	**0.507**	0.496	–
XLM-R_NS	0.362	0.396	0.392	0.416	0.434	0.439	0.451	0.486	0.498	0.469	**0.500**	0.479	–
XLM-R_TBD	0.425	0.473	0.448	0.46	0.467	0.479	0.497	0.487	0.476	0.481	0.496	**0.502**	–
XLM-R_TB	0.418	0.432	0.418	0.455	0.447	**0.461**	0.459	0.437	0.444	0.432	0.416	0.428	–
W2V						–							0.398
W2V_NS						–							0.470
BASELINE						–							0.447

Table B.1: Results on EN-TB-Dense

Model	Layer score 1	2	3	4	5	6	7	8	9	10	11	12	Score
XLM-R	0.564	0.593	0.57	0.601	0.618	0.622	0.645	**0.664**	0.66	0.651	0.655	0.656	–
XLM-R_NS	0.603	0.605	0.603	0.607	0.633	0.614	0.670	**0.689**	0.668	0.672	0.672	0.653	–
XLM-R_TBD	0.564	0.566	0.570	0.562	0.601	0.605	0.614	0.618	0.608	**0.620**	0.618	0.610	–
XLM-R_TB	0.580	0.555	0.595	0.605	0.607	0.610	0.633	**0.643**	0.633	0.631	0.626	0.601	–
W2V						–							0.622
W2V_NS						–							0.628
BASELINE						–							0.589

Table B.2: Results on ES-TimeBank

Model	Layer score 1	2	3	4	5	6	7	8	9	10	11	12	Score
XLM-R	0.286	0.321	0.275	0.307	0.318	0.371	0.396	0.379	0.379	0.379	**0.400**	0.346	–
XLM-R_NS	0.264	0.300	0.336	0.300	0.307	0.336	0.332	0.368	0.371	0.389	**0.404**	0.354	–
XLM-R_TBD	0.293	0.307	0.311	0.336	0.361	0.393	0.382	**0.393**	0.339	0.336	0.357	0.282	–
XLM-R_TB	0.279	**0.307**	0.300	0.296	0.271	0.271	0.279	0.296	0.268	0.279	0.271	0.246	–
W2V						–							0.279
W2V_NS						–							0.268
BASELINE						–							0.321

Table B.3: Results on EN-TimeBank

Model	Layer score 1	2	3	4	5	6	7	8	9	10	11	12	Score
XLM-R	0.480	0.460	0.520	0.515	0.545	0.551	0.535	0.540	**0.581**	**0.581**	0.545	0.530	–
XLM-R_NS	0.490	0.540	0.545	0.551	0.591	**0.621**	**0.621**	0.591	0.606	0.616	0.586	0.601	–
XLM-R_TBD	0.460	0.470	0.505	0.535	0.510	0.535	0.510	0.515	0.535	0.586	0.545	**0.566**	–
XLM-R_TB	0.470	0.500	0.485	0.530	**0.495**	0.490	0.460	0.465	0.455	0.465	0.414	0.338	–
W2V						–							0.369
W2V_NS						–							0.444
BASELINE						–							0.253

Table B.4: Results on IT-TimeBank

Model	Layer score 1	2	3	4	5	6	7	8	9	10	11	12	Score
XLM-R	0.338	0.324	0.338	0.396	0.424	0.482	**0.511**	**0.511**	0.482	0.460	0.446	0.439	–
XLM-R_NS	0.331	0.345	0.388	0.403	0.424	0.482	0.511	**0.525**	0.518	0.511	0.511	0.453	–
XLM-R_TBD	0.331	0.324	0.367	0.432	0.432	0.468	0.475	**0.482**	0.460	0.439	0.460	0.475	–
XLM-R_TB	0.353	0.345	0.324	0.388	0.353	0.388	**0.410**	0.381	0.396	0.345	0.353	0.295	–
W2V						–							0.345
W2V_NS						–							0.317
BASELINE						–							0.216

Table B.5: Results on FR-TimeBank

C Statistical Testing

The Tables from C.1 to C.10 illustrate the results of the McNemar's tests for each language and each probing model (including the baseline based on the most frequent class). The values in all the Tables correspond to p-values. The threshold of the α value for significance has been set to < 0.05.

C.1 PTLMs last layer

The Tables from C.1 to C.5 report the p-value scores for each language when using the last layer of each PTLM.

Model	XLM-R	XLM-R_NS	XLM-R_TBD	XLM-R_TB	W2V	W2V_NS	BASELINE
XLM-R	–	0.235	0.734	< 0.001	< 0.001	0.147	0.007
XLM-R_NS		–	0.171	0.006	< 0.001	0.652	0.085
XLM-R_TBD			–	< 0.001	< 0.001	0.055	< 0.001
XLM-R_TB				–	0.057	0.002	0.033
W2V					–	< 0.001	0.002
W2V_NS						–	0.069
BASELINE							–

Table C.1: Significance EN-TB-Dense - last layer.

Model	XLM-R	XLM-R_NS	XLM-R_TBD	XLM-R_TB	W2V	W2V_NS	BASELINE
XLM-R	–	0.911	0.037	0.014	0.136	0.203	0.003
XLM-R_NS		–	0.053	0.024	0.181	0.271	0.006
XLM-R_TBD			–	0.740	0.664	0.498	0.410
XLM-R_TB				–	0.242	0.065	0.031
W2V					–	0.736	0.057
W2V_NS						–	0.006
BASELINE							–

Table C.2: Significance ES-TimeBank - last layer.

Model	XLM-R	XLM-R_NS	XLM-R_TBD	XLM-R_TB	W2V	W2V_NS	BASELINE
XLM-R	–	0.888	0.038	0.004	0.053	0.028	0.555
XLM-R_NS		–	0.022	0.002	0.040	0.022	0.444
XLM-R_TBD			–	0.358	1.000	0.760	0.343
XLM-R_TB				–	0.386	0.624	0.033
W2V					–	0.780	0.290
W2V_NS						–	0.184
BASELINE							–

Table C.3: Significance EN-TimeBank - last layer.

Model	XLM-R	XLM-R_NS	XLM-R_TBD	XLM-R_TB	W2V	W2V_NS	BASELINE
XLM-R	–	0.029	0.435	< 0.001	0.024	0.556	< 0.001
XLM-R_NS		–	0.419	< 0.001	< 0.001	0.024	< 0.001
XLM-R_TBD			–	< 0.001	0.001	0.171	< 0.001
XLM-R_TB				–	0.047	< 0.001	0.009
W2V					–	0.065	< 0.001
W2V_NS						–	< 0.001
BASELINE							–

Table C.4: Significance IT-TimeBank - last layer.

Model	XLM-R	XLM-R_NS	XLM-R_TBD	XLM-R_TB	W2V	W2V_NS	BASELINE
XLM-R	–	0.832	0.359	0.003	0.099	0.019	< 0.001
XLM-R_NS		–	0.728	0.001	0.060	0.011	< 0.001
XLM-R_TBD			–	< 0.001	0.007	0.728	< 0.001
XLM-R_TB				–	0.222	0.766	0.080
W2V					–	0.327	0.005
W2V_NS						–	0.038
BASELINE							–

Table C.5: Significance FR-TimeBank - last layer.

C.2 PTLMs best layer

The tables from C.6 to C.10 report the *p*-value scores for each language when using the best layer of each PTLM.

Model	XLM-R	XLM-R_NS	XLM-R_TBD	XLM-R_TB	W2V	W2V_NS	BASELINE
XLM-R (11)	–	0.639	0.830	0.005	< 0.001	0.036	< 0.001
XLM-R_NS (11)		–	0.901	0.017	< 0.001	0.081	0.003
XLM-R_TBD (12)			–	0.013	< 0.001	0.055	< 0.001
XLM-R_TB (6)				–	< 0.001	0.608	0.430
W2V					–	< 0.001	0.002
W2V_NS						–	0.069
BASELINE							–

Table C.6: Significance EN-TB-Dense - best layer.

Model	XLM-R	XLM-R_NS	XLM-R_TBD	XLM-R_TB	W2V	W2V_NS	BASELINE
XLM-R (8)	–	0.198	0.037	0.152	0.067	0.124	0.002
XLM-R_NS (8)		–	< 0.001	0.005	0.001	0.004	< 0.001
XLM-R_TBD (8)			–	0.597	0.935	0.740	0.251
XLM-R_TB (10)				–	0.640	0.897	0.003
W2V					–	0.736	0.057
W2V_NS						–	0.006
BASELINE							–

Table C.7: Significance ES-TimeBank - best layer.

Model	XLM-R	XLM-R_NS	XLM-R_TBD	XLM-R_TB	W2V	W2V_NS	BASELINE
XLM-R (11)	–	1.000	0.043	0.002	< 0.001	< 0.001	0.045
XLM-R_NS (11)		–	0.066	0.003	< 0.001	< 0.001	0.035
XLM-R_TBD (9)			–	0.241	0.097	0.052	0.691
XLM-R_TB (8)				–	0.668	0.470	0.520
W2V					–	0.780	0.290
W2V_NS						–	0.184
BASELINE							–

Table C.8: Significance EN-TimeBank - best layer.

Model	XLM-R	XLM-R_NS	XLM-R_TBD	XLM-R_TB	W2V	W2V_NS	BASELINE
XLM-R (10)	–	0.256	1.000	0.212	< 0.001	0.072	< 0.001
XLM-R_NS (7)		–	0.382	0.015	< 0.001	0.003	< 0.001
XLM-R_TBD (10)			–	0.177	< 0.001	0.036	< 0.001
XLM-R_TB (4)				–	0.013	0.532	< 0.001
W2V					–	0.065	< 0.001
W2V_NS						–	< 0.001
BASELINE							–

Table C.9: Significance IT-TimeBank - best layer.

Model	XLM-R	XLM-R_NS	XLM-R_TBD	XLM-R_TB	W2V	W2V_NS	BASELINE
XLM-R (8)	–	0.804	0.503	< 0.001	< 0.001	< 0.001	< 0.001
XLM-R_NS (8)		–	0.286	0.003	< 0.001	< 0.001	< 0.001
XLM-R_TBD (8)			–	0.015	0.009	0.002	< 0.001
XLM-R_TB (7)				–	0.608	0.164	< 0.001
W2V					–	0.327	0.005
W2V_NS						–	0.038
BASELINE							–

Table C.10: Significance FR-TimeBank - best layer.

CogBERT: Cognition-Guided Pre-trained Language Models

Xiao Ding, Bowen Chen, Li Du, Bing Qin, Ting Liu
Harbin Institute of Technology
{xding, bwchen, ldu, qbing, tliu}@ir.hit.edu.cn

Abstract

We study the problem of integrating cognitive language processing signals (e.g., eye-tracking or EEG data) into pre-trained models like BERT. Existing methods typically fine-tune pre-trained models on cognitive data, ignoring the semantic gap between the texts and cognitive signals. To fill the gap, we propose CogBERT, a framework that can induce fine-grained cognitive features from cognitive data and incorporate cognitive features into BERT by adaptively adjusting the weight of cognitive features for different NLP tasks. Extensive experiments show that: (1) Cognition-guided pre-trained models can consistently perform better than basic pre-trained models on ten NLP tasks. (2) Different cognitive features contribute differently to different NLP tasks. Based on this observation, we give a fine-grained explanation of why cognitive data is helpful for NLP. (3) Different transformer layers of pre-trained models should encode different cognitive features, with word-level cognitive features at the bottom and semantic-level cognitive features at the top. (4) Attention visualization demonstrates that CogBERT can align with human gaze patterns and improves its natural language understanding ability.[1]

1 Introduction

Pre-trained models such as BERT (Devlin et al., 2019), GPT (Radford and Narasimhan, 2018) and RoBERTa (Liu et al., 2019) have brought promising improvements to natural language processing (NLP) tasks, such as event prediction (Li et al., 2019) and sentiment analysis (Hoang et al., 2019).

On the other hand, from a language processing perspective, cognitive neuroscience studies the biological and cognitive processes underlying language processing in human brains. Researchers specifically designed pre-trained models to capture how the brain represents language meaning

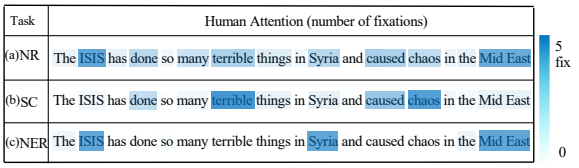

Figure 1: Human attention in different settings. NR (Normal Reading) means human reading without tasks.

(Schwartz et al., 2019; Toneva and Wehbe, 2019). Prior work mainly incorporates cognitive signals by explicitly fine-tuning the pre-trained model to predict language-induced brain recordings.

However, this line of work cannot give a fine-grained analysis and interpretation of why cognitive data is helpful for NLP. This is of great importance to guide future cognition-inspired NLP studies on what kind of cognitive features should be induced from cognitive data and how these cognitive features contribute to NLP tasks. For example, Figure 1 shows eye-tracking data from native speakers of English, where Figure 1 (a) illustrates the number of fixations during the normal reading of humans. Figure 1 (b) and (c) show the number of fixations given the NLP task of sentiment classification (SC) and named entity recognition (NER), respectively. We can observe that human attention is different for the same sentence, given different NLP tasks. In particular, for the SC task, people pay more attention to emotion words, such as "*terrible*" and "*chaos*". While for the NER task, people tend to focus on named entity words, such as "*ISIS*" and "*Syria*". Unfortunately, prior studies cannot give such fine-grained analysis by simply fine-tuning pre-trained models on cognitive data.

To facilitate this, we propose CogBERT, a cognition-guided pre-trained model. Specifically, we focus on the effects of using eye-tracking data, which provides gaze information from native speakers by tracking eye movements and measuring fixation duration. Instead of directly fine-tuning BERT

[1]Our code will be released upon publication.

Proceedings of the 29th International Conference on Computational Linguistics, pages 3210–3225
October 12–17, 2022.

Name	Independent	Strands	Index
Word Position	✓	Lower	1
Word Length	✓	Lower	2
NER Word	✓	Lower	3
Content Word	✓	Lower	4
Noun Phrase	✓	Upper	5
Emotion Word	✓	Upper	6
Active/Passive Aux Poss/Prep Mod Coordinating Conj	✓	Upper	7
Prep/Adj Comp Prep Obj	✓	Upper	8

Table 1: Cognitive feature set. For syntactic features, to avoid sparsity issue, we group active and passive aux, poss and prep mod, and coordinating conj as one feature. Prep, adj comp, and prep obj are grouped as one feature. Aux, Poss, Prep, Mod, Conj, Adj, Comp, Obj are abbreviations for auxiliary, possession, preposition, modifier, conjunction, adjective, complement and objective.

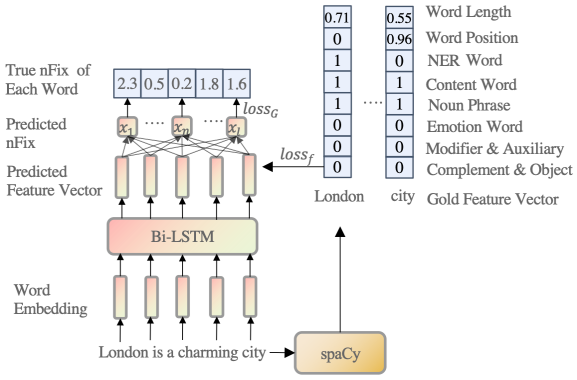

Figure 2: Feature prediction by the Bi-LSTM.

on cognitive data, we first extract psycholinguistic features based on cognition theory (Scarborough et al., 2009). Then we filter out statistically insignificant features in the eye-tracking data (which means that the human attention of words with these features is not significantly higher/lower than the average attention of words). Subsequently, we incorporate these cognition-validated features into BERT by fine-tuning on different NLP tasks. In the fine-tuning process, we would learn different weights for each type of feature according to different NLP tasks.

Our results show that CogBERT can perform better than baseline systems on three benchmark datasets across ten NLP tasks. We give a detailed quantitative analysis of the contributions of different cognitive features to different NLP tasks. Case studies show that CogBERT does learn patterns of human reading behavior compared to BERT. Our findings can provide more insights into cognition-enhanced NLP.

2 CogBERT

CogBERT works in a two-stage setting. The first stage is to induce cognitive features from texts and assign weights for these features guided by human reading signals. In the second stage, we integrate cognitive features into BERT and learn task-specific feature weights for different NLP tasks.

2.1 Cognitive Features Induction and Validation

Psycholinguistic studies (Scarborough et al., 2009) indicate that the acquisition of human reading ability is reflected in two aspects: "*lower strands*" and "*upper strands*". The lower strands (including phonology, morphology, etc.) work together as the reader becomes accurate and automatic with repetition and practice. Meanwhile, the upper strands (including language structures, semantics, etc.) reinforce each other and weave together with the lower strands to produce a skilled reader. Inspired by previous work, we construct an initial cognitive feature set including 46 fine-grained cognitive features extracted from texts using the spaCy tool (Honnibal and Montani, 2017) and divide them into lower strands features (word-level) and upper strands features (semantic/syntax-level).

However, not all cognitive features are statistically significant in eye-tracking data (Zuco 1.0 (Hollenstein et al., 2018), Zuco 2.0 (Hollenstein et al., 2020), and Geco (Cop et al., 2016))[2]. Hence, we filter out cognitive features that the human attention of words with these features is not significantly higher/lower than the average attention of words, retaining 14 usable cognitive features and group them as 8 independent features shown in Table 1. The statistically significant values of each cognitive feature and details of feature validation are shown in Appendix Table 1.

2.2 Learning Weighted Cognitive Feature Vector

We can extract features from texts by using the spaCy tool. Nevertheless, these features should not be assigned to the same or random weights, as their contributions to fitting human understanding of sentences are different. Hence, as shown in Figure 2, given an input sentence $S = \{w_1, w_2, \ldots, w_l\}$ with l words, we train a four-layer Bi-LSTM to

[2]Eye-tracking data is preprocessed by averaging across readers.

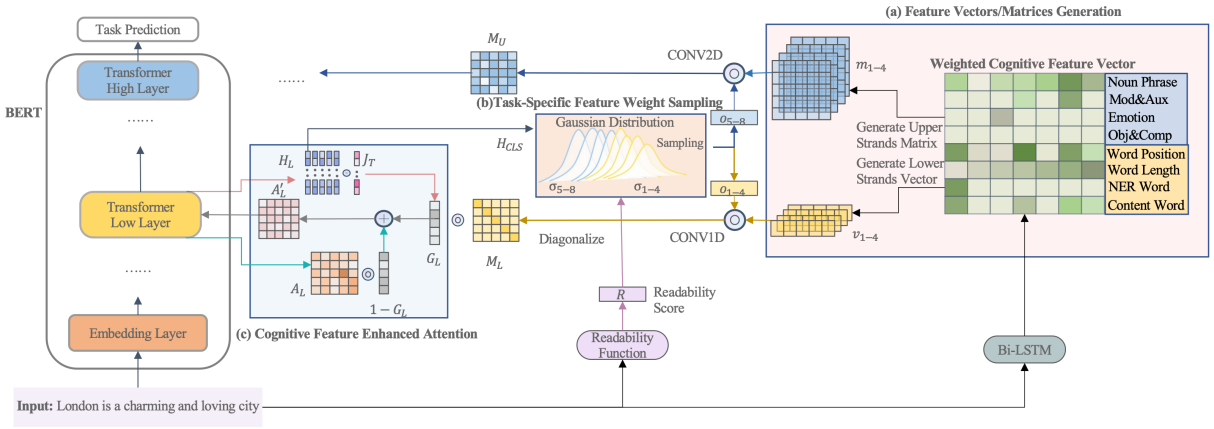

Figure 3: CogBERT Architecture. The Bi-LSTM generates feature vectors, which are used to generate feature matrices $\mathbf{m}$ and feature vectors $\mathbf{v}$. The readability score R and embeddings of [CLS] tokens H_{CLS} are used to compute feature weights o.

map each word embedding to a weighted eight-dimensional cognitive feature vector. According to the cognitive theory (Scarborough et al., 2009), we believe that cognitive features can explain the allocation of human gaze information. Hence, we use gaze information (the number of fixations, nFix) of eye-tracking data (Zuco 1.0, Zuco 2.0, and Geco) as the supervision signals to train the Bi-LSTM model (We also use fixation duration as the supervision signals and achieve the same experimental results). We use the Mean Square Error (MSE) loss $loss_G = \frac{1}{l}\sum_{i=1}^{l}(x_i - y_i)^2$, where $x \in \mathbb{R}^l$ is the predicted nFix score and $y \in \mathbb{R}^l$ is the golden feature score.

On the other hand, to avoid predicting an unreasonable feature score, we also compute the MSE between the predicted feature score matrix $\mathcal{P} = \{\mathbf{p}_1, \ldots, \mathbf{p}_l\} \in \mathbb{R}^{l \times r}$ and the golden score matrix $\mathcal{Q} = \{\mathbf{q}_1, \ldots, \mathbf{q}_l\} \in \mathbb{R}^{l \times r}$ (for example, if a word w_i is a NER word, its golden feature score is 1 on the feature dimension of NER), where $\mathbf{p}_i$ and $\mathbf{q}_i$ is a r-dimensional vector and r is the number of features, which we denote as loss $loss_F = \frac{1}{l}\sum_{i=1}^{l}\frac{1}{r}\sum_{j=1}^{r}(p_{ij} - q_{ij})^2$. Then the model is trained with the objective $loss_T = \alpha \times loss_G + (1-\alpha) \times loss_F$, where $\alpha \in [0, 1]$ is a model parameter to weight the two loss functions (the best development results were obtained with $\alpha = 0.5$).

Note that we use the Bi-LSTM model to predict cognitive features rather than using a regression model to compute the feature score. The main reason is that our goal is to learn the human reading behavior rather than simply fitting the data.

2.3 Incorporating Cognitive Features into BERT

2.3.1 Feature Vectors/Matrices Generation

As shown in Figure 3 (a), for each input sentence S with l words, we can obtain its an $l \times r$ feature matrix from the Bi-LSTM model.

• For each lower strands feature (i.e., word length, word position, NER and content word), we can generate an initial l-dimensional feature vector $\mathbf{v}$ for it from the Bi-LSTM model, respectively.

• For each upper strands feature (i.e., NP chunk, emotion word, Mod&Aux and Obj&Comp), we can generate an initial $l \times l$ feature matrix $\mathbf{m}$ for it from the Bi-LSTM model, respectively. If k adjacent words make up an upper strands feature, its value in $\mathbf{m}$ is the average feature score of k adjacent words obtained from the Bi-LSTM model. The rest of values in $\mathbf{m}$ are filled with 0.

2.3.2 Task-Specific Feature Weight Sampling

We argue that different cognitive features should be given different weights when learning different NLP tasks. For example, the emotion word feature should be more important than others for a sentiment classification task. To address this issue, we utilize the Flesch readability assessment (Kincaid et al., 1975) to evaluate the contribution of cognitive features to sentence readability of different NLP tasks. In other words, given a specific task, if a feature can improve the readability of the input sentence, it should be given a higher weight.

Formally, we use the Flesch readability assessment score R and the embedding of [CLS] token H_{CLS} of a layer in BERT to control the weights of different features. Given a sentence S, the number

of words in S is s_w and the number of syllables in S is s_s. Then the readability score of this sentence is calculated as $R(S) = 206.835 - 1.015 s_w - 84.6 \frac{s_s}{s_w}$, where constants (i.e., 206.835, 1.015 and 84.6) in $R(S)$ are empirical values from Kincaid et al. (1975).

Since $R \in \mathbb{R}^1$ is a one-dimensional number, $H_{CLS} \in \mathbb{R}^{n \times 1}$ is a n-dimensional vector ($n = 768$ for BERT-Base) and cognitive features set T contains eight features including $\{\mathbf{v}_1, ..\mathbf{v}_4, \mathbf{m}_1, ..\mathbf{m}_4\}$, where $\mathbf{v}$ represents a feature vector and $\mathbf{m}$ represents a feature matrix, respectively, we first need to map R and H_{CLS} to an eight-dimensional vector O, to assign a reasonable weight for each feature.

In particular, as shown in Figure 3 (b), we map R and H_{CLS} to a variance set $\sigma = \{\sigma_1, \dots \sigma_r\} \in \mathbb{R}^r$ of a normal distribution $B \sim Norm(0, \sigma^2)$, where the mean of B is zero. σ is computed as:

$$\sigma = f(N^T R) \circ f(M^T H_{CLS}) = \{\sigma_1, \dots, \sigma_r\}, \quad (1)$$

where $\circ$ is the Hadamard product and f is the $tanh$ activation function. $N \in \mathbb{R}^{1 \times r}$ is a mapping vector and $M \in \mathbb{R}^{n \times r}$ is a mapping matrix. Then the weighted feature set $T^{'}$ is computed as:

$$\begin{aligned} T^{'} &= T \circ O = \{o_1 \mathbf{v}_1, \dots, o_4 \mathbf{v}_4, o_5 \mathbf{m}_1, \dots, o_r \mathbf{m}_4\}, \\ O &= B(0, \sigma^2) = \frac{1}{\sigma\sqrt{2\pi}} \exp\left(-\frac{x^2}{2\sigma^2}\right)_{x=0} \\ &= \frac{1}{\sigma\sqrt{2\pi}} = \{o_1, \dots, o_r\}, \end{aligned} \quad (2)$$

where o_i is the ith feature weight sampled from the normal distribution B with sample point $x = 0$.

2.3.3 Cognitive Feature Enhanced Attention

Inspired by the previous work (Jawahar et al., 2019), which indicates that BERT captures surface features in lower layers and semantic features in higher layers, as shown in Figure 3 (c), we incorporate different cognitive features in different layers, where lower strands are in lower layers and upper strands are in higher layers. We use CNN (LeCun et al., 2015) to compute a feature-enhanced attention score for lower strands and upper strands, which we denote as $M_L \in \mathbb{R}^{l \times l}$ and $M_U \in \mathbb{R}^{l \times l}$, respectively,

$$\begin{aligned} M_L &= Diag(1DCNN(\mathbf{v}_1, \mathbf{v}_2, \mathbf{v}_3, \mathbf{v}_4)), \\ M_U &= 2DCNN(\mathbf{m}_1, \mathbf{m}_2, \mathbf{m}_3, \mathbf{m}_4), \end{aligned} \quad (3)$$

where $Diag$ refers to filling the output of $1DCNN$ in the diagonal of the feature-enhanced attention matrix M_L, this is because the generated lower strands features by the Bi-LSTM are word-level feature vector rather than a matrix, and $1DCNN$ and $2DCNN$ means 1-dimensional and 2-dimensional CNN network. The kernel size of both $1DCNN$ and $2DCNN$ is 3, respectively. Then we obtain the cognitive feature-enhanced lower layer attention matrix $A'_L \in \mathbb{R}^{l \times l}$ and higher layer attention matrix $A'_U \in \mathbb{R}^{l \times l}$ as:

$$\begin{aligned} A'_L &= M_L \circ G_L + A_L \circ (1 - G_L), \\ A'_U &= M_U \circ G_U + A_U \circ (1 - G_U), \\ G_L &= f(J^T H_L), \\ G_U &= f(J^T H_U), \end{aligned} \quad (4)$$

where $A_L \in \mathbb{R}^{l \times l}$ and $A_U \in \mathbb{R}^{l \times l}$ are the original attention matrix of lower and higher layers, respectively. $G \in \mathbb{R}^{1 \times l}$ is a gated vector for a transformer head, which is used to balance the combination between the original attention and the cognitive feature-enhanced attention. $H = \{H_{CLS}, H_0 \dots, H_l\} \in \mathbb{R}^{n \times l}$ is the hidden state of the input sentence S, in which H_{CLS} is the embedding of [CLS] token of BERT. $J \in \mathbb{R}^{n \times 1}$ is a vector to map H to the gate vector G in a specific head. The $\circ$ is the row-level Hardmard product that each value of G multiplies with each row of the matrix (e.g., M_L), so that the size of the matrix remains the same after multiplication. The process is same for all layers of BERT.

2.4 Training Details

We train the Bi-LSTM model with hidden size 256, dropout ratio 0.15, 300-dimensional GloVe embedding (Pennington et al., 2014) and 40 epochs. In experiment, we fine-tune the CogBERT using AdamW (Loshchilov and Hutter, 2017) optimizer with learning rate from [1e-5, 3e-5, 5e-5], warm-up ratio of 0.1. The training epochs are form [3, 5, 10] and batch size from [16, 32] for GLUE Benchmark. For CoNLL2000 Chunking, we use batch size from [64, 128], learning rate from [5e-6, 1e-5, 5e-5], and epochs from [40, 60].

3 Experiments

To show the effectiveness of CogBERT, we compare it with baselines on three benchmark datasets across ten NLP tasks and an eye-tracking prediction task.

3.1 Baselines

We compare CogBERT with:

Models	SST2	COLA	MRPC	RTE	MNLI(m/mm)	QQP	STS-B	QNLI
BERT-Base	93.5	51.7	87.2	67.2	84.3/83.7	71.1	85.4	90.4
Syntax-BERT-Base	**94.0**	54.1	89.2	68.9	**84.9/84.6**	72.0	86.7	91.1
fMRI-EEG BERT-Base	93.4	52.9	87.4	67.5	84.3/83.8	71.2	85.3	90.5
Eye-tracking BERT-Base	93.3	51.9	87.5	67.3	84.3/83.7	71.2	85.8	90.6
CogBERT-Base (Random)	93.2	51.1	85.4	66.0	84.1/83.2	71.0	85.3	88.6
CogBERT-Base	**94.0**	**55.1**	**89.5**	**69.4**	**84.9/84.6**	**72.1**	**87.2**	**91.3**
BERT-Large	94.9	60.5	89.3	70.1	86.8/85.9	72.1	86.5	92.7
Syntax-BERT-Large	**96.1**	61.9	92.0	74.7	86.7/86.6	72.5	88.5	**92.8**
fMRI-EEG BERT-Large	—	—	—	—	—	—	—	—
Eye-tracking BERT-Large	94.7	60.7	89.2	70.2	86.6/85.7	72.3	86.7	92.5
CogBERT-Large (Random)	94.4	60.1	88.7	69.2	86.3/85.4	71.8	86.3	92.4
CogBERT-Large	**96.1**	**62.1**	**92.1**	**74.9**	**86.8/86.6**	**72.8**	**89.7**	**92.8**
RoBERTa-Base	95.4	57.1	90.8	73.8	86.3/86.2	72.5	87.4	92.2
Syntax-RoBERTa-Base	**96.1**	63.3	91.4	**81.2**	87.8/85.7	73.5	88.3	**94.3**
CogRoBERTa-Base (Random)	95.3	56.8	90.5	73.4	86.1/85.8	72.1	87.2	92.0
CogRoBERTa-Base	95.7	**63.5**	**91.7**	79.3	**88.1/86.2**	**73.8**	**88.5**	93.9
RoBERTa-Large	96.3	63.8	91.0	84.2	89.5/89.7	72.7	90.2	94.2
Syntax-RoBERTa-Large	**96.9**	64.3	92.5	85.0	90.2/90.0	73.1	**91.4**	**94.5**
CogRoBERTa-Large (Random)	96.1	63.6	90.7	83.8	89.2/89.4	72.3	90.0	94.0
CogRoBERTa-Large	96.5	**64.6**	**92.8**	**85.3**	**90.4/90.3**	**73.5**	90.5	**94.5**

Table 2: Results on the test set of GLUE Benchmark. F1 scores are reported for QQP and MRPC. Spearman correlations are reported for STS-B, and accuracy scores are reported for the other tasks. The best results are bold. For the SST2 task, the Emotion feature weight of CogBERT is set as 0.

• **BERT** (Devlin et al., 2019): A transformer-based pre-trained language model achieved SOTA performances on various NLP tasks.

• **Syntax-BERT** (Bai et al., 2021): A syntax-enhanced pre-trained model outperformed conventional pre-trained models on various NLP tasks.

• **fMRI-EEG BERT** (Schwartz et al., 2019): This model is fine-tuned on EEG and fMRI data, which showed improvements in brain activity prediction and achieved competitive performances on various NLP tasks.

• **Eye-tracking BERT**: It fine-tunes BERT on nFix eye-tracking data following the method used in fMRI-EEG BERT.

• **CogBERT (Random)**: The feature weight of CogBERT is randomly generated rather than supervised learning by human reading signals.

3.2 GLUE Benchmark

The General Language Understanding Evaluation (GLUE) benchmark (Wang et al., 2019) is a set of tasks to test the model's ability to understand natural language. We implement our method in BERT, RoBERTa on its development set. Results are shown in Table 2. Based on the observation of the experiment results, we find that:

(1) Compared with BERT and RoBERTa, CogBERT achieves consistently better results (when compared with RoBERTa, we use RoBERTa as the base pre-trained model of CogBERT) on all tasks from SST2 to QNLI showing the effectiveness of incorporating eye-tracking can be useful for various tasks.

Models	P (%)	R (%)	F1 (%)
BERT-Base-Cased	95.32	95.23	95.33
SeqVat (unsupervised)	95.39	95.47	95.45
SeqVat (supervised)	96.36	96.31	96.34
CogBERT-Base (Random)	94.46	95.03	94.74
CogBERT-Base	**96.48**	**96.59**	**96.54**

Table 3: Results of CoNLL-2000 text chunking. CogBERT is based on BERT-Base-Cased.

(2) Comparison between Syntax-BERT and CogBERT shows that cognitive features can further improve the performance of BERT on NLP tasks. This is mainly because on the one hand cognitive features used in this paper have already included fine-grained syntax structure features. On the other hand, we involved more psycholinguistic features in CogBERT validated by eye-tracking data.

(3) CogBERT outperforms fMRI-EEG BERT and Eye-tracking BERT, which indicates that fine-tuning on cognitive data cannot fully exploit the value of cognitive data. By inducing fine-grained cognitive features from cognitive data can provide a new perspective for this line of work.

(4) Compared to CogBERT (Random), CogBERT achieves consistently better performances, which confirms that different cognitive features contribute differently to language comprehension in brains. Learning weighted cognitive feature vector is effective for CogBERT.

Tasks	Word Length	Content Word	NER	Word Position	Emotion	NP Chunk	Mod & Aux	Comp & Obj
COLA	0.42	0.56	0.59	0.41	0.78	**1.00**	0.83	0.60
MRPC	0.52	0.68	**1.00**	0.45	0.95	0.76	0.88	0.74
RTE	0.54	0.83	0.67	0.52	0.91	**1.00**	0.73	0.64
CoNLL-2000 Chunking	0.54	0.83	0.94	0.52	0.92	**1.00**	0.64	0.93
CoNLL-2003 NER	0.44	0.51	**1.00**	0.49	0.42	0.39	0.49	0.51

Table 4: Results of intrinsic evaluation of CogBERT-base. The numbers are feature weights in different tasks.

3.3 Sequence Labeling

In addition to the GLUE benchmark, we also evaluate CogBERT on the CoNLL-2000 dataset (Sang and Buchholz, 2000) for text chunking and the CoNLL-2003 dataset (Sang and Meulder, 2003) for NER. The results are shown in Table 3 (as the space is limited, the results of NER are shown in Appendix Table 3). Note that for the Chunking and NER task, the NP Chunk and NER feature weight of CogBERT is set as 0, respectively, to ensure no data leakage problem.

We compare CogBERT with SeqVat (Chen et al., 2020), which uses virtual adversarial training to improve the model's performance and robustness on the CoNLL-2000 text chunking task. We observe that our method outperforms all baselines on the benchmark dataset, which demonstrates that CogBERT could also benefit sequence labeling tasks.

3.4 Intrinsic Evaluation

As shown in Table 4, we present an intrinsic evaluation to output the feature weight of CogBERT in different tasks, which is sampled from the Gaussian Distribution in Equation (2). Due to the space limitation, we only demonstrate feature weights of the BERT-base model in COLA, MRPC, RTE, CoNLL-2000 Chunking, and CoNLL-2003 NER tasks. The feature weight $C_f^{'} \in \mathbb{R}^i$ scaled by the data size and the feature density is calculated as:

$$
\begin{aligned}
f_d &= \frac{1}{Z}\sum_{k=1}^{z} U_k/(sum(U_k)), \\
C_s &= \frac{1}{Z}\sum_{k=1}^{z} C_k/f_d, \\
C_f^{'} &= C_s/\max(C_s),
\end{aligned}
\tag{5}
$$

where the data size of a task is $Z \in \mathbb{R}^z$, the averaged feature weight of 12 layers from CogBERT is $C \in \mathbb{R}^{r\times z}$, and the density of feature is $f_d \in \mathbb{R}^r$. C_s is the feature weight scaled by f_d, and the count of each feature in the task is $U \in \mathbb{R}^{r\times z}$ (for example, if there are 3 entity words in a training example, the count of NER feature of this example is 3).

Models	Zuco	Geco (EN)	Geco (NL)
BERT-EN	93.42(0.02)	93.68(0.14)	—
BERT-NL	—	—	91.81(0.23)
BERT-MULTI	93.74(0.05)	93.73(0.12)	91.90(0.16)
XLM-17	92.05(2.25)	91.79(1.75)	91.04(0.70)
XLM-100	93.97(0.09)	93.04(1.40)	**92.31(0.22)**
CogBERT (Random)	93.23(0.13)	93.36(0.38)	91.59(0.35)
CogBERT	**93.99(0.02)**	**93.90(0.14)**	91.97(0.09)

Table 5: Results of eye-tracking prediction. Standard deviation of 5 runs is reported in parentheses.

We find that CogBERT can assign reasonable weights to different features in various tasks. In the COLA (a linguistic acceptability judgment task), CogBERT evaluates NP Chunk (i.e., the noun phrase) as the most important feature and gives a high score to other syntax-related features. This is because the syntax structure helps judge linguistic acceptability. In the MRPC (a sentence paraphrasing task), CogBERT considers NER and emotion as the two most important features. The main reason is that if two sentences do not share the same entity and emotion, they are probably not a paraphrase. In the RTE (a text entailment task), CogBERT thinks of NP Chunk as the most important feature, and this is probably because if a sentence can be inferred from another sentence, they might have a similar phrasal structure or meaning. In CoNLL-2000 Chunking and CoNLL-2003 NER tasks, it is not surprising that CogBERT ranks NP Chunk and NER as the most crucial feature, respectively.

3.5 Eye-tracking Prediction

We argue that CogBERT can be useful not only for NLP tasks but also for language comprehension in brains. Hence, we also evaluate the effectiveness of CogBERT on eye-tracking prediction (Hollenstein et al., 2021) tasks using three eye-tracking benchmark datasets, including Zuco (Zuco 1.0 and Zuco 2.0), Geco (EN), and Geco (NL). The training details are the same with Hollenstein et al. (2021)[3]. This task tests the model's ability to learn human eye-tracking data, including first fixation duration (FFD), total reading time (TRT), number of fixa-

[3]Geco (EN) and Geco(NL) are the English and Dutch parts of Geco eye-tracking data, respectively.

Models	SST2	MRPC	RTE
CogBERT	**93.7**	**89.7**	**70.1**
w/o lower strands	93.4	89.1	68.2
w/o upper strands	92.9	88.7	67.3
w/o all strands	92.3	87.4	66.4
w/o readability	93.2	88.9	68.5
w/o layer-wise	93.1	88.5	67.9

Table 6: Results of ablation study.

tions (nFix), mean fixation duration (MFD), first pass duration (FPD), fixation proportion (FProp), number of re-fixations (NREFIX), re-read proportion (REPROP). The performance is evaluated by the mean absolute error (MAE), and we report 100–MAE as the result in this experiment.

We compare our method with BERT-EN, BERT-NL (de Vries et al., 2019), BERT-MULTI, (Wolf et al., 2020)[4] and XLM (Lample and Conneau, 2019) (XLM is a cross-lingual pre-trained language model. XLM-17 pre-trains on 17 languages and 100 for XLM-100). CogBERT is based on BERT-EN and BERT-NL, respectively. The results are shown in Table 5.

We find that CogBERT outperforms BERT-EN, BERT-NL, BERT-MULTI, and XLM-17 and achieves comparable performance with XLM-100, even though CogBERT is based on BERT, which only pre-trained on one language whereas XLM-100 pre-trained on more than 100 languages. This also shows the effectiveness of our cognitive features induced from cognitive data for understanding and explaining human gaze behavior.

3.6 Ablation Study

We conduct ablation studies over several factors related to CogBERT's performance on the downstream tasks. All results are obtained on the development sets of SST2, MRPC and QNLI, and are shown in Table 6.

We observe that replacing lower or upper strands cognitive features can decrease the model performance, and removing all strands cognitive features would further affect the model performance. We also notice that although the readability is not as important as cognitive features for our model, removing it also harms the performance. This is mainly because this factor also constrains the learning process of CogBERT from a cognitive perspective. Without layer-wise means that we integrate all features into each layer of BERT, The poor performance of without layer-wise demonstrates that incorporating features in a layer-wise manner is an effective way for cognition-guided NLP.

[4]BERT-EN, BERT-NL and BERT-MULTI, are English, Dutch and multilingual version of BERT.

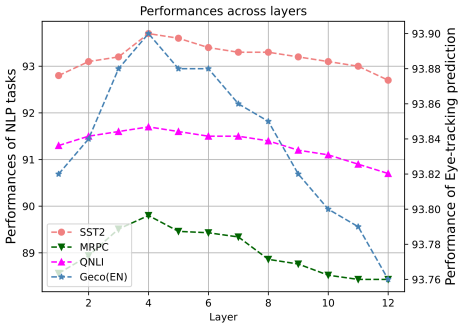

Figure 4: Performances of layer study.

3.7 Layer Study

In this section, we quantificationally discuss which layer of BERT should be the boundary for the lower and upper strands cognitive features. We conduct comparative experiments on the development sets of SST2, MRPC, QNLI and STS-B tasks, and illustrate results in Figure 4. The Y-axis is the performance for different NLP tasks. The X-axis is the number of layers. For example, if the number of layer is 6, we incorporate lower strands cognitive features into 1-6 layers of BERT and upper strands cognitive features into the rest.

We find that all tasks reach the best performance when the layer boundary is around 4, which means that BERT's lower layers are more suitable for incorporating lower strands cognitive features and upper strands cognitive features are more useful when we incorporate them in higher layers. These results can effectively guide the future research of cognition-enhanced pre-trained models. Similarly, previous research (Tenney et al., 2019) also finds that the BERT behaves like a pipeline manner where the lower layer processes basic information and upper layer processes the sentence based on the information of previous layers, in which its argument is similar to ours and our research further validates their research and further proved this argument from the perspective of model implementation level.

3.8 Attention Visualization

To qualitatively analyze the effectiveness of our method, we visualize the attention case of CogBERT and compare it with BERT and humans. We select cases from SST2, NER and MRPC tasks. To compare with human cognition, given a specific

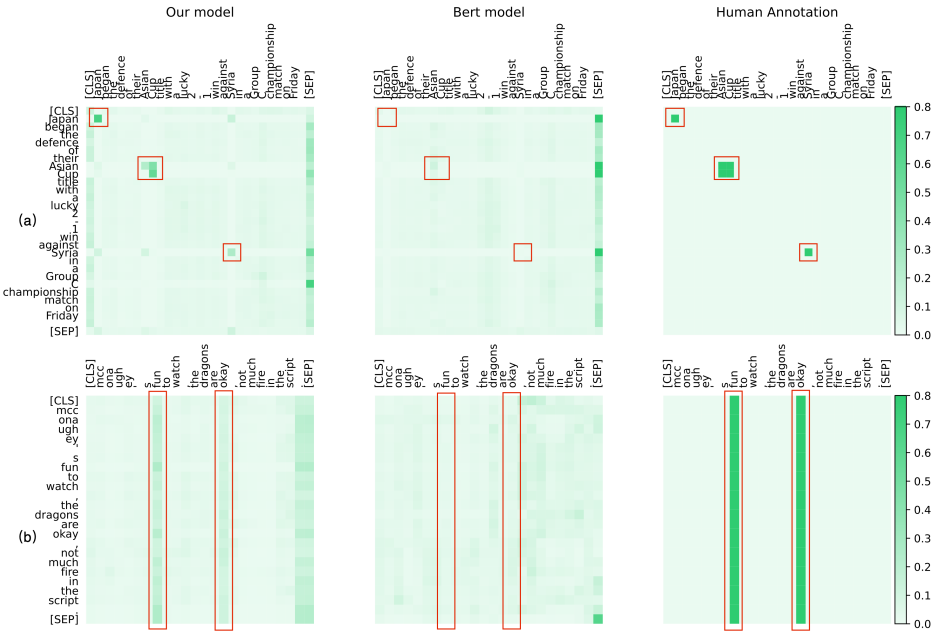

Figure 5: Attention cases selected from the SST2 and CoNLL-2003 NER task, respectively.

NLP task, we ask four annotators to highlight their attention words when reading the sentences. For BERT and CogBERT, we select attention scores from higher layers of pre-trained models, which can capture task-specific features (Merchant et al., 2020). The attention visualizations of SST2 and NER are illustrated in Figure 5 (MRPC is shown in Appendix Figure 1).

Figure 5 (a) presents the attention visualization for the CoNLL-2003 NER task, illustrating that CogBERT pays more attention to NER words 'Asian Cup', 'Japan' and 'Syria' like humans, whereas BERT gives little attention to these words. Figure 5 (b) illustrates the attention visualization for the SST2 task, showing that CogBERT captures the critical emotion words 'fun' and 'okay', which are also important for the human judgments. In contrast, BERT fails to focus on these words. These experimental results indicate that although pre-trained models have achieved promising improvements in numerous NLP tasks, they are still far from the level of human intelligence. Cognition-guided pre-trained models can provide an effective way to approach human cognition, by learning the attention mechanism in human reading.

4 Related Work

Prior neuroscience studies have demonstrated that cognitive data is associated with language comprehension activity in human brains (Just and Carpenter, 1980; Brooks and Meltzoff, 2005), showing longer duration links to a greater processing load in different language units (words, clauses, texts). These studies established the theoretical grounding for cognition-guided NLP.

In cognitively motivated NLP, researchers investigate the impact of cognitive language processing signals on NLP tasks, especially focusing on improving neural networks by utilizing cognitive data. Early researches mainly used LSTM or CNN to incorporate cognitive signals (Barrett et al., 2018; González and Søgaard, 2018; Hollenstein and Zhang, 2019; Sood et al., 2020; Ren and Xiong, 2021; Takmaz et al., 2020).

As pre-trained models have shown their great power on various NLP tasks, a line of work focuses on exploring cognitive-data-enhanced pre-trained models (Hollenstein et al., 2019, 2021; McGuire and Tomuro, 2021), mainly by fine-tuning pre-trained models on cognitive data.

However, these methods cannot give a fine-grained analysis of how cognitive data contributes to different NLP tasks. In contrast, CogBERT is inspired by the theory in psycholinguistics and encodes cognitive features induced from eye-tracking data into pre-trained models in a layer-wise manner with carefully designed architecture, enabling us to perform a fine-grained analysis of how cognitive data contributes to different NLP tasks and further improved the performance of pre-trained models comparing to previous simply fine-tuned ones.

5 Conclusion

We present CogBERT, a framework that can effectively incorporate cognitive signals into pre-trained models. Experimental results show that CogBERT achieves new SOTA results on three NLP benchmark datasets. Analyses suggest that CogBERT can adaptively learn task-specific cognitive feature weights to give fine-grained explanations of how cognitive data works on NLP tasks. This work provides a new paradigm in learning cognition-enhanced pre-trained models, and extensive elaborated experiments can guide future researches.

References

Jiangang Bai, Yujing Wang, Yiren Chen, Yaming Yang, Jing Bai, Jing Yu, and Yunhai Tong. 2021. Syntax-bert: Improving pre-trained transformers with syntax trees.

Maria Barrett, Joachim Bingel, Nora Hollenstein, Marek Rei, and Anders Søgaard. 2018. Sequence classification with human attention. In *Proceedings of the 22nd Conference on Computational Natural Language Learning*, pages 302–312, Brussels, Belgium. Association for Computational Linguistics.

Rechele Brooks and Andrew Meltzoff. 2005. The development of gaze following and its relation to language. *Developmental science*, 8:535–43.

Luoxin Chen, Weitong Ruan, Xinyue Liu, and Jianhua Lu. 2020. SeqVAT: Virtual adversarial training for semi-supervised sequence labeling. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 8801–8811, Online. Association for Computational Linguistics.

Uschi Cop, Nicolas Dirix, Denis Drieghe, and Wouter Duyck. 2016. Presenting geco: An eyetracking corpus of monolingual and bilingual sentence reading. *Behavior Research Methods*, 49.

Wietse de Vries, Andreas van Cranenburgh, Arianna Bisazza, Tommaso Caselli, Gertjan van Noord, and Malvina Nissim. 2019. Bertje: A dutch bert model.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. Bert: Pre-training of deep bidirectional transformers for language understanding.

Ana Valeria González and Anders Søgaard. 2018. Learning to predict readability using eye-movement data from natives and learners. In *AAAI*.

Mickel Hoang, Oskar Alija Bihorac, and Jacobo Rouces. 2019. Aspect-based sentiment analysis using BERT. In *Proceedings of the 22nd Nordic Conference on Computational Linguistics*, pages 187–196, Turku, Finland. Linköping University Electronic Press.

Nora Hollenstein, Antonio de la Torre, Nicolas Langer, and Ce Zhang. 2019. Cognival: A framework for cognitive word embedding evaluation.

Nora Hollenstein, Federico Pirovano, Ce Zhang, Lena Jäger, and Lisa Beinborn. 2021. Multilingual language models predict human reading behavior.

Nora Hollenstein, Jonathan Rotsztejn, Marius Troendle, Andreas Pedroni, Ce Zhang, and Nicolas Langer. 2018. Zuco, a simultaneous eeg and eye-tracking resource for natural sentence reading. *Scientific Data*, 5:180291.

Nora Hollenstein, Marius Troendle, Ce Zhang, and Nicolas Langer. 2020. Zuco 2.0: A dataset of physiological recordings during natural reading and annotation.

Nora Hollenstein and Ce Zhang. 2019. Entity recognition at first sight: Improving ner with eye movement information.

Matthew Honnibal and Ines Montani. 2017. spaCy 2: Natural language understanding with Bloom embeddings, convolutional neural networks and incremental parsing. To appear.

Ganesh Jawahar, Benoît Sagot, and Djamé Seddah. 2019. What does BERT learn about the structure of language? In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 3651–3657, Florence, Italy. Association for Computational Linguistics.

Marcel Adam Just and Patricia A. Carpenter. 1980. A theory of reading: from eye fixations to comprehension. *Psychological review*, 87 4:329–54.

J. Peter Kincaid, Robert P. Fishburne, Richard Lawrence Rogers, and Brad S. Chissom. 1975. Derivation of new readability formulas (automated readability index, fog count and flesch reading ease formula) for navy enlisted personnel.

Reinhold Kliegl, Ellen Grabner, Martin Rolfs, and Ralf Engbert. 2004. Length, frequency, and predictability effects of words on eye movements in reading. *European Journal of Cognitive Psychology - EUR J COGN PSYCHOL*, 16:262–284.

Victor Kuperman, Michael Dambacher, Antje Nuthmann, and Reinhold Kliegl. 2010. The effect of word position on eye-movements in sentence and paragraph reading. *Quarterly journal of experimental psychology (2006)*, 63:1838–57.

Guillaume Lample and Alexis Conneau. 2019. Cross-lingual language model pretraining.

Yann LeCun, Yoshua Bengio, and Geoffrey Hinton. 2015. Deep learning. *Nature*, 521(7553):436–444.

Zhongli Li, Qingyu Zhou, Chao Li, Ke Xu, and Yunbo Cao. 2021. Improving bert with syntax-aware local attention.

Zhongyang Li, Xiao Ding, and Ting Liu. 2019. Story ending prediction by transferable bert.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Roberta: A robustly optimized bert pretraining approach.

Ilya Loshchilov and Frank Hutter. 2017. Decoupled weight decay regularization.

Ilya Loshchilov and Frank Hutter. 2019. Decoupled weight decay regularization.

Erik McGuire and Noriko Tomuro. 2021. Relation classification with cognitive attention supervision. In *Proceedings of the Workshop on Cognitive Modeling and Computational Linguistics*, pages 222–232, Online. Association for Computational Linguistics.

Amil Merchant, Elahe Rahimtoroghi, Ellie Pavlick, and Ian Tenney. 2020. What happens to bert embeddings during fine-tuning?

Jeffrey Pennington, Richard Socher, and Christopher Manning. 2014. Glove: Global vectors for word representation. volume 14, pages 1532–1543.

Alec Radford and Karthik Narasimhan. 2018. Improving language understanding by generative pre-training.

Yuqi Ren and Deyi Xiong. 2021. Cogalign: Learning to align textual neural representations to cognitive language processing signals.

Erik Sang and Sabine Buchholz. 2000. Introduction to the conll-2000 shared task: Chunking. *Proc of CoNLL-2000 and LLL-2000*.

Erik Sang and Fien Meulder. 2003. Introduction to the conll-2003 shared task: Language-independent named entity recognition. *Proceeding of the Computational Natural Language Learning (CoNLL)*.

H.S. Scarborough, F. Fletcher-Campbell, Janet Soler, and G. Reid. 2009. Connecting early language and literacy to later reading (dis)abilities: Evidence, theory, and practice. *Approaching difficulties in literacy development: assessment, pedagogy, and programmes*, pages 23–39.

Dan Schwartz, Mariya Toneva, and Leila Wehbe. 2019. Inducing brain-relevant bias in natural language processing models.

Timothy Slattery and Mark Yates. 2017. Word skipping: Effects of word length, predictability, spelling and reading skill. *The Quarterly Journal of Experimental Psychology*, 71:1–30.

Ekta Sood, Simon Tannert, Philipp Mueller, and Andreas Bulling. 2020. Improving natural language processing tasks with human gaze-guided neural attention.

Ece Takmaz, Sandro Pezzelle, Lisa Beinborn, and Raquel Fernández. 2020. Generating Image Descriptions via Sequential Cross-Modal Alignment Guided by Human Gaze. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 4664–4677, Online. Association for Computational Linguistics.

Ian Tenney, Dipanjan Das, and Ellie Pavlick. 2019. BERT rediscovers the classical NLP pipeline. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 4593–4601, Florence, Italy. Association for Computational Linguistics.

Mariya Toneva and Leila Wehbe. 2019. Interpreting and improving natural-language processing (in machines) with natural language-processing (in the brain).

Alex Wang, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel R. Bowman. 2019. GLUE: A multi-task benchmark and analysis platform for natural language understanding. In the Proceedings of ICLR.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Rémi Louf, Morgan Funtowicz, Joe Davison, Sam Shleifer, Patrick von Platen, Clara Ma, Yacine Jernite, Julien Plu, Canwen Xu, Teven Le Scao, Sylvain Gugger, Mariama Drame, Quentin Lhoest, and Alexander M. Rush. 2020. Huggingface's transformers: State-of-the-art natural language processing.

A Appendix

A.1 Training Details

A.1.1 Bi-LSTM Model Training

The eye-tracking data to train the Bi-LSTM model contains 1637 sentences and 76,937 words. The embedding size of Glove Vector (Pennington et al., 2014) is 300, which is accessible on the page.[5] We used the embedding trained on Common Crawl corpus, and the embedding is not trained along with the Bi-LSTM model. The hidden size of Bi-LSTM is 256, the dropout ratio in the feed-forward network is 0.15. We train the Bi-LSTM model with 40 epochs with learning rate 1e-3, and we use the AdamW optimizer(Loshchilov and Hutter, 2019). We evaluate the $Loss_G$ on the test set of nFix and select the model with the best performance.

A.1.2 CogBERT Model Training

For Transformers, we set layer number as 12 to keep the layer setting the same with the different base versions of pre-train models, the hidden dimension of intermediate layers as 512, dropout

[5] https://github.com/stanfordnlp/GloVe

ratio as 0.15, and the dimension of the fully connected layer before Softmax activation as 2000. The learning rate is initialized as 5e-4 for the AdamW optimizer, and we use the linear learning rate schedule with the warm-up ratio of 0.1.

For BERT and RoBERTa, we use the pre-trained version of different models released by HuggingFace.[6]For GLUE Benchmark. We use AdamW optimizer with learning rate from [1e-5, 3e-5, 5e-5] on GLUE benchmark. The training epochs are from [3, 5, 10]. We also use the linear learning rate schedule with a warm-up ratio of 0.1. The batch size is 32 for all tasks. The kernel size for both 1DCNN and 2DCNN is 3. The 2DCNN module does not have bias, and the padding strategy is the "same".

For CoNLL-2000 chunk and CoNLL-2003 NER tasks, we use AdamW optimizer with learning rate from [1e-6, 3e-6, 5e-6]. The training epochs are from [40, 60]. We also use the linear learning rate schedule with a warm-up ratio of 0.1. The batch size is 32 with a gradient accumulation step of 4 to avoid memory issues.

For the Eye-tracking Prediction task, we follow Hollenstein et al. (2021) and replace the BERT model with our model. The training setting is precisely the same as the original task setting released on the page.[7] The results are AdamW optimizer with a learning rate of 5e-5 and a weight decay of 0.01. We employ a linear learning rate decay schedule over the total number of training steps. We clip all gradients exceeding the maximal value of 1. We train the models for 100 epochs, with early stopping after seven epochs without improving the validation accuracy.

A.2 Eye-tracking Feature Validation

For word-level features, the results are computed using the percentage of the word that carries a specific feature with a higher nFix value than sentence average, which means the feature is a strong indicator for this word that carries this feature to be fixated or not.

For semantic features, the strategy is different. For noun phrases, we take the nFix of the first word and found that the first word of nFix always has a below-average nFix value, which shows that a below-average first word of a phrase is an indicator to show that this is a noun phrase. We take the nFix average of the emotion phrase and compare it with the sentence average nFix value. For dependency relation, we check the equality relation between the nFix of two words, and the percentage is computed using that the ratio of a head word has a nFix that is greater or less than the tail word. We checked 46 features that cover a large portion of existing syntax features, and the results are in Table 7.

Additionally, some dependency relations are very uncommon, which does not hold a statistical count above one hundred in those eye-tracking corpora. We do not use uncommon relations with less than 100 counts in the corpus, which we label these relations as sparse in the Table 7. In this research, we do not consider any relationship that is sparse in the model. We hope this table is helpful for future research in this area. Since word length and position are already proven to be directly connected to eye-tracking data in psycholinguistics (Slattery and Yates, 2017;Kliegl et al., 2004;Kuperman et al., 2010), therefore we do not further validate these two features again in our research.

A.3 CoNLL-2003 NER Task

We are not able to reproduce the BERT results in CoNLL-2003 NER tasks, which is also discussed in different GitHub issues[8][9][10]. The official GitHub page of BERT says they used a complicated pre-processing script to process the entity word before training using BERT. However, they are not planning to release the pre-processing script.

A.4 MRPC Attention Case

Besides SST2 and CoNLL-2003 NER task, we also present another attention case in MRPC. In this example. Sentence 1 is "*Sanitation is poor.. there could be typhoid and cholera*" and sentence 2 is "*Sanitation is poor drinking water is generally left behind..there could be typhoid and cholera*". These two sentences are a paraphrase, which states poor sanitation might cause typhoid and cholera.

From the cognition of humans, human labels "*Sanitation is poor*," "*typhoid*" and "*cholera*" as the keywords to label these two sentences as a paraphrase.

The attention of CogBERT, BERT, and humans is in Figure 6. Since this is a classification task

[6] https://huggingface.co/transformers/

[7] https://github.com/DS3Lab/multilingual-gaze

[8] https://github.com/dmlc/gluon-nlp/issues/593

[9] https://github.com/kyzhouhzau/BERT-NER/issues/2

[10] https://github.com/google-research/bert/issues/223

Feature	Zuco 2.0	Zuco 1.0	Geco	Avg	Sparse	Trands
Word Length	✓	✓	✓	✓	✓	Lower
Word Position	✓	✓	✓	✓	✓	Lower
Punctuation	×	×	×	×	×	Lower
Marker	29.60%	29.41%	31.83%	30.28%	True	Lower
ContentWord	96.0%	98.20%	99.50%	97.90%	False	Lower
NER	67.70%	68.90%	83.10%	73.20%	False	Lower
Determiner	7.60%	10.80%	14.06%	10.82%	False	Lower
Negation	33.30%	43.62%	47.00%	41.31%	False	Lower
Emotion	75.00%	66.67%	88.89%	93.85%	False	Upper
NPChunk	75.00%	89.30%	85.60%	83.30%	False	Upper
Parent-Child	54.80%	55.20%	62.00%	57.33%	False	Upper
Token-Head	46.00%	47.90%	43.82%	46.00%	False	Upper
Word and subtree	56.67%	54.81%	64.49%	58.66%	False	Upper
Adjective Modifier	63.79%	62.85%	57.40%	61.35%	False	Upper
Noun Subject	35.88%	49.17%	32.47%	39.17%	False	Upper
Compound	60.79%	51.30%	35.58%	49.22%	False	Upper
Adjective Complement	81.30%	60.30%	85.76%	75.79%	False	Upper
Adjective Clause	52.94%	41.00%	54.50%	49.48%	True	Upper
Adverbial Clause Modifier	51.85%	59.45%	58.70%	56.67%	False	Upper
Adverbial Modifier	52.03%	46.70%	53.05%	50.59%	False	Upper
Agent	0.00%	5.71%	5.33%	3.68%	True	Upper
Appositional Modifier	40.62%	43.50%	58.30%	47.47%	True	Upper
Auxiliary	12.87%	21.95%	30.18%	21.67%	False	Upper
Passive Auxiliary	24.72%	14.38%	10.88%	16.66%	False	Upper
Coordinating Conjunction	5.71%	11.34%	17.46%	11.50%	False	Upper
Clausal Complement	53.13%	48.45%	48.52%	50.03%	False	Upper
Conjunct	51.34%	48.59%	53.05%	50.99%	True	Upper
Clausal Subject	0.00%	100%	83.33%	61.00%	True	Upper
Dative	33.33%	36.36%	25.19%	31.63%	False	Upper
Direct Object	38.55%	43.58%	37.32%	39.82%	True	Upper
Expletive	0.00%	76.47%	78.70%	52.00%	True	Upper
Interjection	—	0.00%	44.23%	22.00%	True	Upper
Meta Modifier	0.00%	0.00%	0.00%	0.00%	True	Upper
Modifier of Nominal	70.00%	48.83%	44.44%	54.00%	False	Upper
Noun Phrase as Adverbial Modifier	60.29%	52.90%	59.09%	57.43%	False	Upper
Passive Nominal Subject	35.53%	32.43%	31.29%	33.08%	False	Upper
Numeric Modifier	41.13%	34.24%	31.29%	33.08%	False	Upper
Object Predicate	60.00%	54.54%	59.36%	58.00%	True	Upper
Parataxis	50.00%	12.50%	63.92%	42.00%	True	Upper
Complement of Preposition	66.67%	85.71%	88.15%	80.18%	False	Upper
Object of Preposition	91.21%	91.50%	84.43%	89.05%	False	Upper
Possession Modifier	29.00%	34.16%	26.56%	30.00%	False	Upper
Pre-correlative Conjunction	0.00%	47.05%	64.71%	37.00%	True	Upper
Predet	0.00%	14.29%	36.90%	17.00%	True	Upper
Prepositional	8.27%	12.44%	16.77%	12.00%	True	Upper
Particle	0.00%	32.25%	20.85%	18.00%	True	Upper
Modifier of Quantifier	70.00%	50.00%	20.05%	47.00%	True	Upper
Relative Clause Modifier	34.00%	40.00%	49.79%	41.00%	True	Upper
Open Clausal Complement	41.17%	40.43%	46.65%	42.75%	True	Upper

Table 7: Validation Table of Features nFix Data. — means this relation does not exist in this corpus based on the annotation result of Spacy. ×means this relation cannot be checked. ✓means this relation has been verified by previous research. Sparse means the number of this relation in any corpus is less than 100.

between two sentences, a [SEP] token will be inserted between them. In an ideal attention situation, for the paraphrase classification task, sentence 1 should attend to the same words of sentence 2, and sentence 2 should attend to the same words of sentence 1, respectively.

From the Figure 6, we can see that CogBERT can align more accurately with human annotation. CogBERT can give more attention to "*Sanitation is poor*" from sentence 1 to sentence 2 and from sentence 2 to sentence 1. In contrast, the BERT model gives very little attention to this phrase. Additionally, CogBERT is also able to focus on the "*typhoid*" word, which is tokenized into several subwords. However, the BERT model is not able to capture this in both sentences. Moreover, for the word "*cholera*," even both CogBERT and BERT can focus on it, CogBERT can give more attention compared to the BERT model.

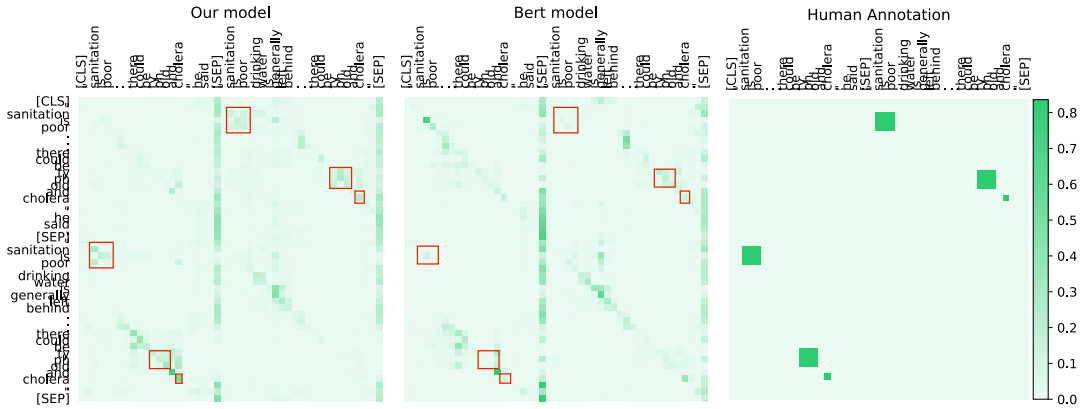

Figure 6: MRPC Attention Case

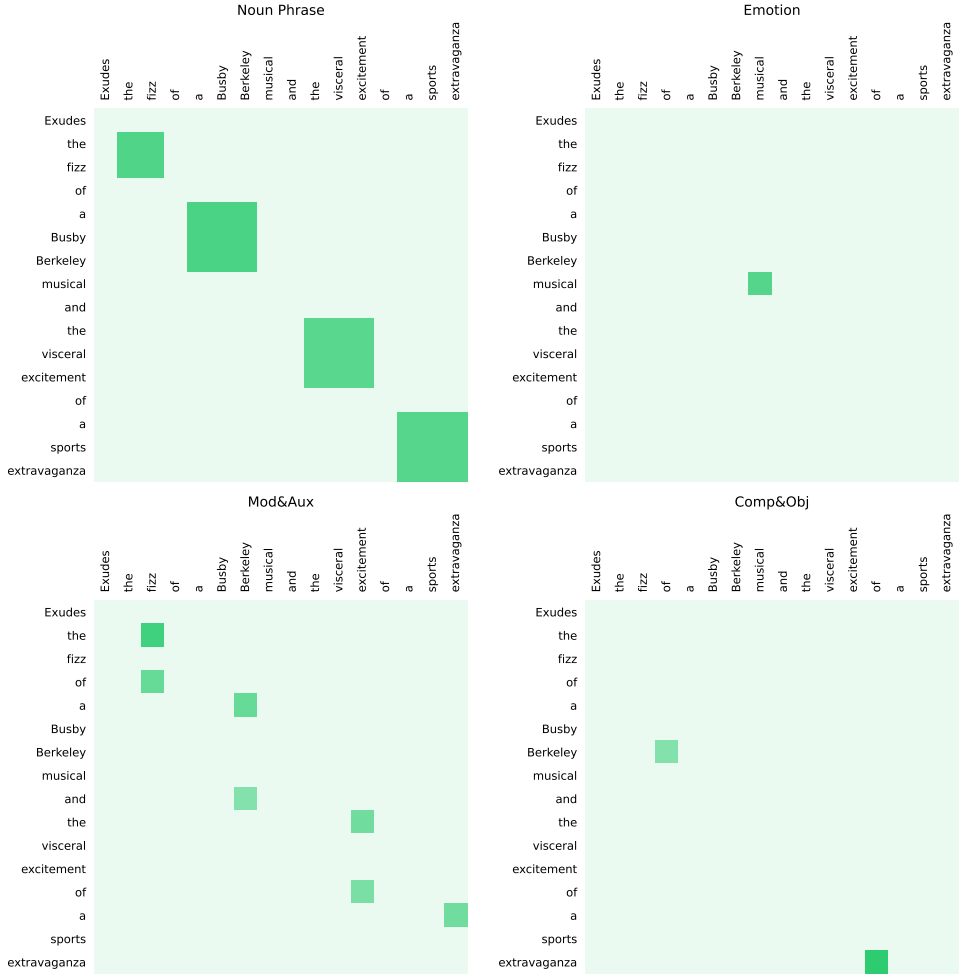

Figure 7: Upper strands feature matrices.

A.5 Model Efficiency

Since CogBERT works in a two-stage setting and uses Spacy to annotate the sentence, it is necessary to report the processing speed to show that processing efficiency is not a significant concern in our work.

A.5.1 Processing Efficiency

First, we report the training time to train the Bi-LSTM model. In a 2GHz four-core Intel Core i5, training the Bi-LSTM model requires 30 minutes in 40 epochs.

The most time-consuming part is the Spacy annotation processing. In the same chip setting, take

Models	P (%)	R (%)	F1 (%)
LISA-Base	90.7	92.2	91.4
SGNET-Base	90.9	92.6	91.7
BERT-Base (Devlin et al., 2019)	—	—	**92.4**
BERT-Base (Li et al., 2021)	91.0	92.3	91.6
CogBERT-Base (Random)	90.3	91.2	90.7
CogBERT-Base	**91.4**↑	**92.6**↑	91.9↑
LISA-Large	91.3	92.6	92.0
SGNET-Large	91.5	92.8	92.1
BERT-Large (Devlin et al., 2019)	—	—	**92.8**
BERT-Large (Li et al., 2021)	91.7	93.1	92.4
CogBERT-Large (Random)	91.1	91.4	91.2
CogBERT-Large	**92.0**↑	**93.2**↑	92.6 ↑

Table 8: Results of CoNLL-2003 NER. CogBERT is based on BERT-Cased. ↑ means that we outperform previous implementation of BERT (Li et al., 2021).

SST2 tasks for example; the training part of the SST2 dataset contains 67,349 examples. In this amount of data, producing feature vectors/matrices with Spacy's annotation for lower and upper strands will take about 15 minutes in different runs. For much larger datasets like QNLI that contains more than 100,000 training examples, running the generation process will take about 40 minutes.

Since most of the data in our experiments are less than 10,000 examples, we do not think the feature vector/matrix generation process is a problem in our model. The generation processing of most tasks can be done in 10 minutes.

A.5.2 Storage Efficiency

Another point to mention is the data size of generated feature vector and matrix. We used a sparse matrix to store the generated feature vector/matrix, which enables us to store them efficiently and recover them in only several lines of code.

Take SST2 for example, the size of generated feature vector/matrix of the whole data is 158.3 Mb, and 708.5 Mb for QNLI, roughly ten times the size of the text version. Since text size is considerably small in different tasks, our generated vector/matrix does not use too much disk storage. Therefore disk storage is not a concern in most tasks.

A.5.3 Training Efficiency

For the training speed of the second stage, since we did not put too many parameters in our model, which are just several linear transformations and two CNN networks. For example, in the MRPC task, the total parameter size of the BERT model is 109M, and the total parameter size of our model is 116M.

Additionally, the CNN network is efficient since the architecture of CNN is suitable for parallel computation. Thus, the training speed in downstream tasks is close to the training speed in different base pre-train models implementation.

Models	nFix Loss	Feature Loss
Bi-LSTM	0.092	—
Bi-LSTM (with feature)	**0.051**	**0.006**

Table 9: Results of nFix prediction. We sample 20% nFix data as test set. The evaluation criteria is the MSE loss.

A.6 nFix Prediction

First, we report the results of the nFix prediction in Table 9. The base model is Bi-LSTM with four layers only trained with nFix prediction, whereas our model will first predict feature score and further predict the nFix following our description.

From the results, we could tell that by training with loss L_T. Our model gives a lower loss, which means the proposed training method can give a more accurate nFix estimation by combining different feature scores. This proves that these features help predict the nFix value, and the model can assign adaptive weight to different features and combine them to better estimate nFix.

A.7 The Effect of Spacy Tool

From the noun phrase in Figure 8, we could see the annotation is not perfect, like the *a Busby Berkeley*, this phrase is only a part of the whole noun phrase.

Reasonably, the performance of the Spacy tool certainly affects our model's performance. However, the annotation of Spacy is correct in most situations. Details about the performance of the Spacy tool are on the official website.[11].In this research, we use the medium size Spacy annotation model in all tasks. For the Eye-tracking Prediction task of Geco(NL), we also use the medium-sized Netherlands model released by Spacy.

A.8 Sample Input/Output of Bi-LSTM Model

We present a sample Bi-LSTM prediction case in Figure 8, the generated feature vectors for lower strands are in Figure 9, and the generated feature matrices for upper strands are in Figure 7.

Figure 9 presents the generated feature vectors of lower strands, including Word Length, Word Position, NER, and Content Words. These features correspond to a single word, which means these features are word-level rather than phrase or sentence level features. Therefore the output of the

[11] https://spacy.io/

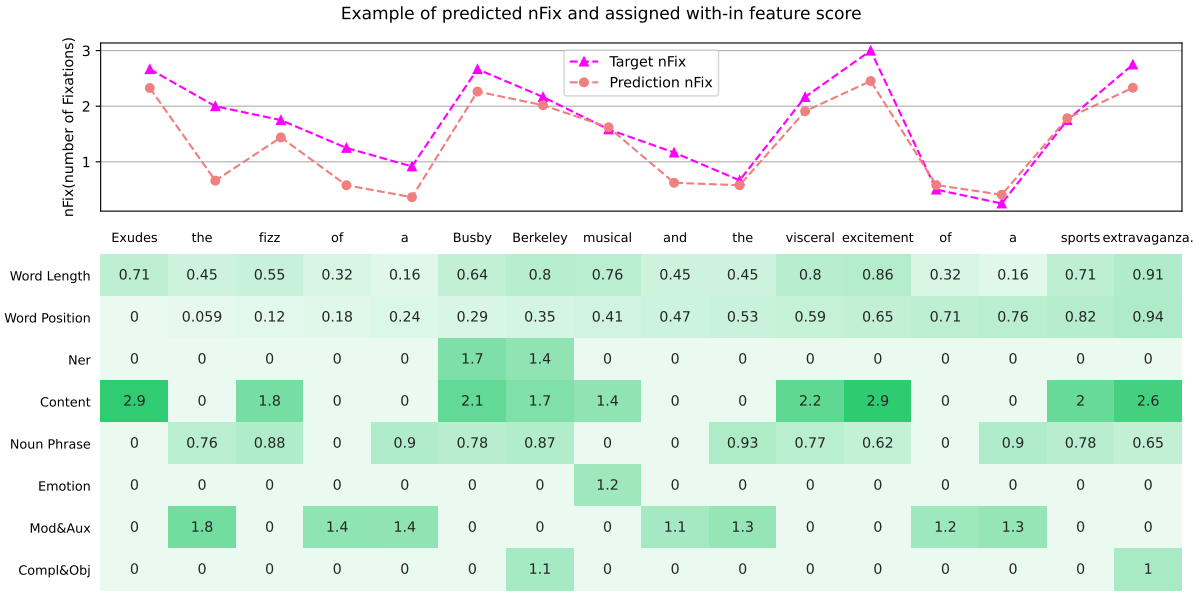

Figure 8: Bi-LSTM Feature Prediction. The number inside each cell of the prediction map are weighted scores computed by the Bi-LSTM model. The top part is the nFix prediction given by our model, the brown line means the prediction, and the pink line means the target nFix measured by human gaze.

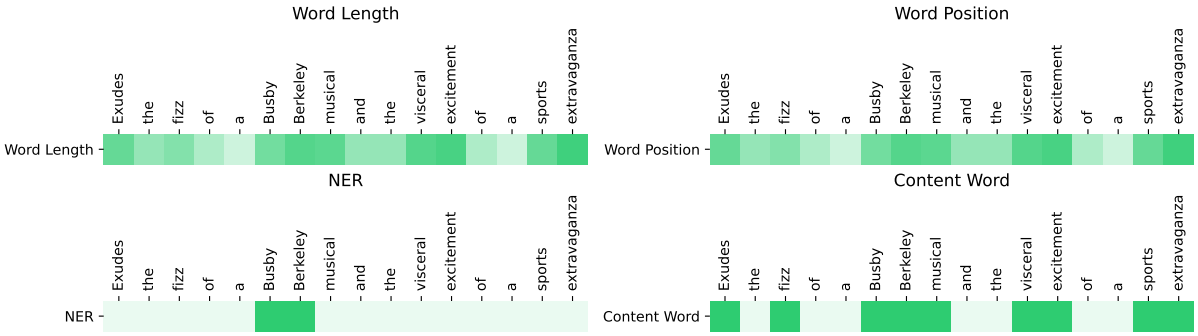

Figure 9: Lower strands feature vectors.

Bi-LSTM model will be directly used as feature vectors for different lower strands features.

Figure 7 presents the generated feature matrices of upper strands containing Noun Phrase, Emotion, Mod&Aux, and Comp&Obj. These features might correspond to several words or a phrase, which means they are phrase-level features compared to lower strands features.

For noun phrase, we take the average weight from the prediction of the Bi-LSTM model of words in the same noun phrase, then we assign the average weight to the noun phrase feature matrix, if the phrase length is N, then we have a noun phrase feature matrix $N \times N$ .In this example, we take the noun phrase "*the fizz*" for example, the Bi-LSTM predicts 0.76 and 0.88 for those two words in the noun phrase dimension. The average weight is 0.82 for this phrase, and then we fill a 2×2 matrix with this value in the whole feature matrix. This process is the same for both noun phrase feature and emotion feature.

For dependency relations, if word w_i is the tail word of w_j of the dependency relation we use in this paper, then we assign the prediction of weight for w_i in this relation to E_{ij}, where E is the feature matrix of this feature. Take first "*the*" word of this sentence in the Mod&Aux dimension, this word is the determiner of the word "*fizz*," and "*the*" is the tail word of the head word "*fizz*." The prediction value from Bi-LSTM for word "*the*" in Mod&Aux dimension is 1.8. Following our generation rules, we assign 1.8 from word "*the*" to word "*fizz*" in the corresponding position of the feature matrix of Mod&Aux. In this case, we assign 1.8 to E_{23}. This process is the same for both Mod&Aux and Comp&Obj features.

A.9 Attention Annotation

To compare the attention produce by CogBERT, BERT and human, we select several cases from MRPC, SST2 and NER cases and asked four annotators, which are all students majoring in Computer Science and have background knowledge about NLP, to highlight the words that they think are important for this task. We then collect the average importance given by the annotators which represents as the importance score for words in that sentence for different tasks.

Can Transformers Process Recursive Nested Constructions, Like Humans?

Yair Lakretz
Cognitive Neuroimaging Unit
NeuroSpin
Gif-sur-Yvette, France

Théo Desbordes
Cognitive Neuroimaging Unit
NeuroSpin, France
Meta AI, Paris, France

Dieuwke Hupkes
Meta AI
Paris, France

Stanislas Dehaene
Cognitive Neuroimaging Unit, NeuroSpin, Gif-sur-Yvette
Collège de France, Paris, France

Abstract

Recursive processing is considered a hallmark of human linguistic abilities. A recent study evaluated recursive processing in recurrent neural language models (RNN-LMs) and showed that such models perform below chance level on embedded dependencies within nested constructions – a prototypical example of recursion in natural language. Here, we study if state-of-the-art Transformer LMs do any better. We test eight different Transformer LMs on two different types of nested constructions, which differ in whether the embedded (inner) dependency is short or long range. We find that Transformers achieve near-perfect performance on short-range embedded dependencies, significantly better than previous results reported for RNN-LMs and humans. However, on *long-range* embedded dependencies, Transformers' performance sharply drops below chance level. Remarkably, the addition of only three words to the embedded dependency caused Transformers to fall from near-perfect to below-chance performance. Taken together, our results reveal how brittle syntactic processing is in Transformers, compared to humans.

1 introduction

One of the fundamental principles of contemporary linguistics states that language processing requires the ability to deal with nested structures. Recursion, a specific type of computation that involves repeatedly applying a function to its own output, is suggested to be at the core of this ability (Hauser et al., 2002). The strongest evidence for recursion in human language processing arises from the tree-like nested structure of sentences in natural language, in which phrases of a particular type (i.e. NPs) can be embedded in other phrases of that same type (Figure 1). Humans, it is argued, are endowed with a unique competence for recursive processing, which allows them to represent and process such nested tree structures (Chomsky, 2000; Hauser et al., 2002; Dehaene et al., 2015).

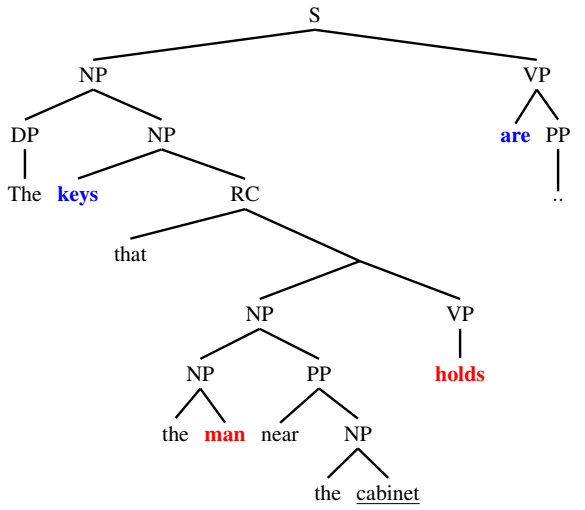

Figure 1: A tree-structure representation of a recursive structure with two long-range dependencies, one nested within the other one.

In recent years, neural language models (NLMs) have shown tremendous advances on a variety of linguistic tasks, such as next-word prediction, translation or semantic inference. Furthermore, evaluations of their syntactic abilities have shown promising results, with similar or even above-human performance on a variety of different tasks (Marvin and Linzen, 2018; Goldberg, 2019; Jumelet et al., 2021; Giulianelli et al., 2018). However, negative results were recently also presented (Warstadt et al., 2020; Hu et al., 2020). In particular, when it comes to recursive processing, Lakretz et al. (2021b) showed that while recurrent neural network language models (RNN-LMs) perform well on long-range dependencies, such as the relationship between **keys** and **are** in sentences like "The **keys** that the *man* near the cabinet *holds*, **are** red" (Figure 2), they perform below chance on the shorter, embedded dependency (*man-holds*). Humans, instead, perform significantly better on such dependencies, although interestingly, for them too, the

Proceedings of the 29th International Conference on Computational Linguistics, pages 3226–3232
October 12–17, 2022.

shorter inner dependency is more difficult than the long outer one.

The study by Lakretz et al. illustrates how investigations of neural networks can inspire experiments about human language processing. However, their study focuses on only a single architecture, an RNN-LM with LSTM units (Hochreiter and Schmidhuber, 1997), which is currently outperformed on many fronts by the newer *Transformer* models (Vaswani et al., 2017). In this short paper, our main question is therefore whether Transformer models do any better when it comes to processing recursive constructions. We then further explore similarities and differences in performance patterns of RNN and Transformer language models.

Our main results show that when tested on nested constructions with a short-range embedded dependency, Transformers outperform RNN-LM across all conditions, with error rates close to zero. However, when the embedded dependency is long-range, their performance dramatically drops to below chance, similarly to the case of RNNs. The mere addition of a short prepositional phrase ('near the cabinet' in the example shown in Figure 1) to the embedded dependency causes model performance to drop from near perfect to below chance level. Thus, contrary to what might be expected based on their much improved performance and the fact that they are trained on substantially more data, Transformer models share RNNs' shortcoming when it comes to recursive, structure-sensitive, processing.

Last, almost all models made more errors when trying to carry a noun in the singular across dependencies which involved a plural noun, than in the converse situation. Interestingly, this bias towards greater interference by plural than by singular is opposite to that reported in Italian RNN-LMs (Lakretz et al., 2021b), and is akin to the Markedness Effect reported for humans.

2 Related Work

In psycholinguistics, grammatical agreement became a standard method to probe online syntactic processing in humans (Bock and Miller, 1991; Franck et al., 2002), since it is ruled by hierarchical structures rather than by the linear order of words in a sentence. More recently, it has also become a standard way to probe grammatical generalization in NLMs (Linzen et al., 2016; Bernardy and Lappin, 2017; Giulianelli et al., 2018; Gulordava et al., 2018; Jumelet et al., 2019; Kersten et al., 2021; Lakretz et al., 2019; Sinha et al., 2021), pointing to both similarities and differences between human and model error patterns.

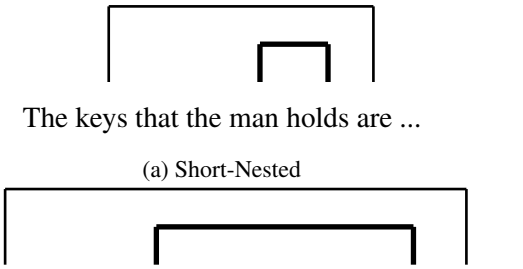

Figure 2: Experimental Design: the two number-agreement tasks – *Short-Nested* and *Long-Nested*. In Short-Nested, the embedded dependency is short-range (in bold); in Long-Nested, it is long-range, through the insertion of a three-word prepositional phrase.

Lakretz et al. (2019) showed that RNN-LMs trained on a large corpus with English sentences develop a number-propagation mechanism for long-range dependencies. The core circuit of this mechanism was found to be extremely sparse, comprising of only a very small number of units. This sparsity of the mechanism suggests that models are not able to process two long-distance dependencies simultaneously, and indeed, this was later confirmed in simulations (Lakretz et al., 2021b). Inspired by this finding, Lakretz et al. (2021b) conducted a following experiment with humans, which showed that they, too, make more errors on nested long-range dependencies. However, contrary to LMs, their performance was above chance on these constructions. This finding suggests that human recursive processing remains significantly better than that of RNN-LMs.

Recursive processing of nested constructions in RNN-LMs was also studied using artificial grammars (Cleeremans et al., 1989; Servan-Schreiber et al., 1991; Gers and Schmidhuber, 2001; Christiansen and Chater, 1999; Hewitt et al., 2020). Recently, Suzgun et al. (2019) showed that memory-augmented RNNs can capture recursive regularities of Dyck languages (also known as "bracket languages"). However, when tested on a simple extension of these languages, RNN-LMs failed to generalize to unseen data with a greater nesting depth (Lakretz et al., 2021a). Specifically, the models failed also in cases in which the training

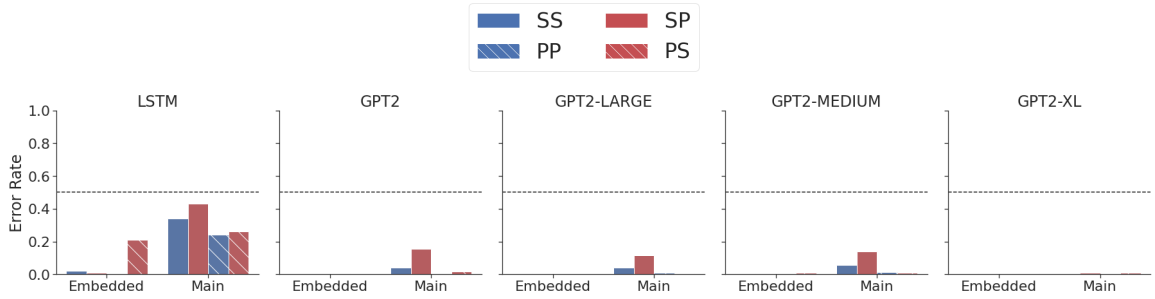

(a) Short-Nested.

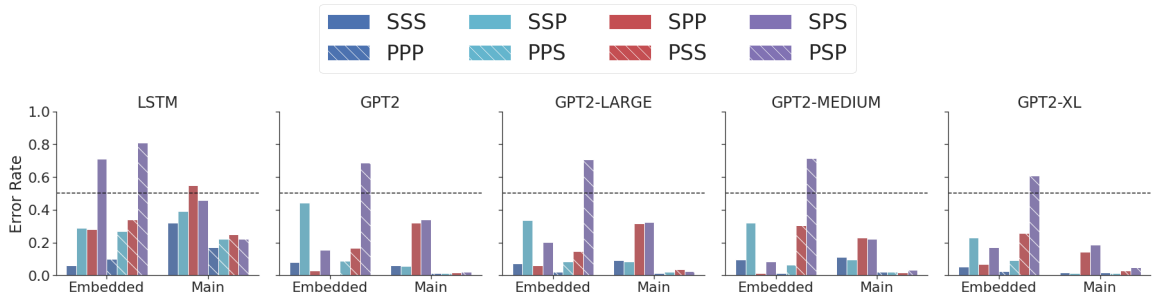

(b) Long-Nested.

Figure 3: Error rates on nested constructions in English, for the LSTM and all causal Transformers, for both the main and embedded agreements. Conditions are marked by the value of the grammatical number of all nouns in the sentence. For example, condition SP means that the first noun is singular and the second is plural. While error-rates are near zero for Short-Nested, they are worse than chance-level for one of the incongruent conditions of Long-Nested, consistently across all models. In this condition (PSP), grammatical agreement is with respect to the second noun, which is singular.

data contained deep structures, up to five levels of nesting. This suggests that the poor recursive processing of RNN-LMs is not merely due to shallow nesting depth in natural data, which is typically not more than two (Karlsson, 2007).

Taken together, previous work suggests that RNN-LMs struggle to capture recursive regularities in either natural or artificial data. Inspired by this line of work, we focus here on Transformer LMs: do they show different patterns when it comes to processing recursive structures? Do they better approximate human ability for recursion?

3 Experimental Setup

We largely follow the experimental setup of Lakretz et al. (2021b), and we consider two different languages (English and Italian) and a different set of models.

Data We consider two number-agreement tasks (*NA-tasks*): *Short-Nested* and *Long-Nested*. Both tasks contain two subject-verb dependencies; they differ in terms of whether the embedded dependency is *short-* or *long-range*. In *Short-Nested*, the subject and verb in the nested dependency are adjacent (Figure 2a). They are embedded in a sentence by inserting an object-relative clause to modify the subject of a different sentence. The *Long-Nested* task (Figure 2b) uses the same constructions, except that an additional three-word prepositional phrase (e.g., "near the cabinet") is added in the embedded dependency.[1]

Models We run experiments with all causal transformer-based NLMs that are currently compatible with the BigBench framework, available from HuggingFace[2], and also with two masked-language models (MLMs). Specifically, we include four GPT-2 models that differed in size: GPT2, GPT2-Medium, GPT2-Large and GPT-XL (Radford et al., 2019); and two masked-language models: RoBERTa and RoBERTa-Large (Liu et al., 2019). In addition, as a baseline, we conduct an experiment with an English LSTM-LM, which was

[1]All data sets are available in the BigBench collaborative benchmark https://github.com/google/BIG-bench/tree/main/bigbench/benchmark_tasks/subject_verb_agreement

[2]https://huggingface.co/transformers/

studied in numerous work in the past (Gulordava et al., 2018).

Model evaluation Following previous work, we evaluated model performance on agreement by comparing the output probabilities for the correct (e.g., 'are') vs. wrong ('is') verb form. For both tasks, we evaluated model performance on agreement for both the embedded and the inner verb, and separately for each task condition (see SM).

4 Results

Causal Transformers, such as GPT-2, receive word input incrementally, similarly to humans. In contrast, masked language models (MLMs), such as RoBERTa (Liu et al., 2019) have access to all tokens in the input in parallel. In sections 4.1 and 4.2 we first focus on English causal models, rather than on MLMs, due to the similarity in input processing, which makes the human-model comparison more direct. In section 4.3, for completeness, we further report results from MLMs. Finally, in section 4.4., we report results for another language, namely, Italian.

4.1 Short-Nested task

In Figure 3a, we show model performance on the Short-Nested task for all causal models trained on English. Overall, the English LSTM made more errors on the main (outer) dependency compared to the embedded (inner) one, with more than 20% errors, across all four conditions. In contrast, Transformers, and in particular GPT2-XL, achieved close to perfect performance across all conditions, on both the embedded and main dependency. For GPT2, GPT2-Medium and Large, the longer main dependency was, however, overall more difficult than the embedded one, but with no more than 20% errors in the incongruent conditions (SP and PS; Table S2).

Interestingly, consistently across all models, both Transformers and the LSTM model made more errors on conditions in which the agreement was with respect to singular, compared to plural.

4.2 Long-Nested task

In Figure 3b, we further show the performance of all English causal models for the Long-Nested task. Overall, all models made more errors across all conditions compared to Short-Nested, but with the same tendency of making more errors on dependencies with respect to singular compared to plural. The most striking difference between the two tasks was the performance of the models on the embedded dependency. In particular, for Transformers, their error rate was close to zero in Short-Nested, but dropped to below-chance on one of the incongurent conditions (PSP) in Long-Nested. Similarly, For the LSTM, this was the case for both incongruent cases (PSP and SPS).

In contrast to the embedded dependency, all models performed above chance on the main, longer, dependency. This shows that for Long-Nested, the length of the dependency affected model performance less than the presence of recursive embedding.

4.3 Masked-Language Transformer Models

In Figure 4, we show the performance of the masked-language models, for both the Short- and Long-Nested tasks. Similarly to causal models, masked-language models achieved near perfect performance on all conditions of the Short-Nested task (except for RoBERTa-Large on the PS condition, but with no more than 30% errors). Importantly, for the Long-Nested task, the addition of only three words to the inner dependency caused the performance of the masked-language models to drop from near perfect to below chance, similarly to the results from causal models. The large drop in performance occurred in both incongruent conditions (SPS and PSP), and not only for the PSP condition (as in case of causal Transformers).

4.4 Italian Models

Following the suggestion of anonymous reviewers, we further tested the ability of Transformer-based models to process nested structures in another language. Specifically, we tested all versions of Transformers trained on Italian, which were compatible with the BigBench framework and available from HuggingFace (footnotes 1 and 2): (1) a Transformer-based model named Gepetto, and (2) a small version of GPT-2.

We tested the performance of these models on both the Short- and Long-Nested tasks, in the same manner as for the English Transformers above. For Short-Nested, unlike the English Transformers, the Italian models achieved relatively poor performance, with below-chance performance on the outer dependency in the incongruent conditions (SP and PS). This performance is significantly below that of humans and that of recurrent neural networks on the same structures (Lakretz et al.,

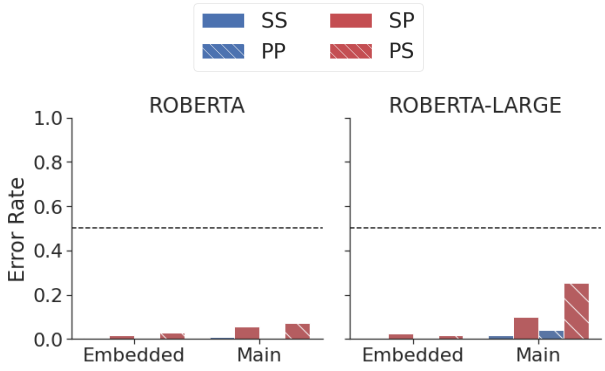

(a) Short-Nested.

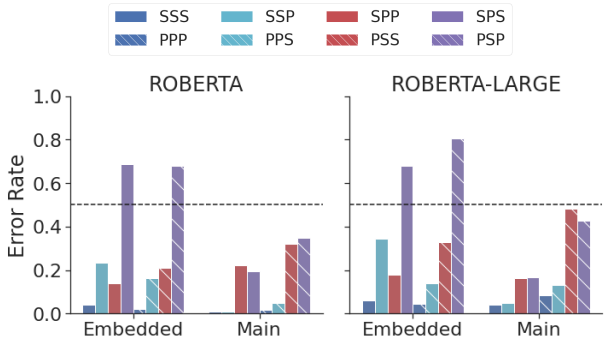

(b) Long-Nested.

Figure 4: Error rates on nested constructions in English for masked-language models (RoBERTa and RoBERTa-Large). Same color scheme as in Figure 3. Similarly to the case of causal Transformer-based models (Figure 3), the addition of only three words to the embedded dependency (from Short-Nested to the Long-Nested task) caused the performance of masked-language models to drop from near perfect to below chance on the incogruent conditions (SPS and PSP).

2021b), which suggests that the current available Transformer-based models for Italian are undertrained. Therefore, further conclusions about syntactic processing in these models are limited.[3] The results for both Short- and Long-Nested tasks can be found in Figure S1 in the supplementary materials.

5 Discussion

In this study, we evaluated the recursive abilities of Transformer LMs on two number-agreement tasks that were previously shown to be exceptionally challenging for LSTM language models. Our experiments showed that, overall, Transformers outperformed LSTM-LMs, and in particular, achieved near perfect performance on short embedded dependencies. However the addition of only a short prepositional phrase to the embedded dependency caused model performance to sharply drop to below chance level.

Furthermore, we found that all causal models showed a bias towards plural and therefore err more when the subject of a verb is in the singular. A similar bias was previously observed in Italian LSTM models (Lakretz et al., 2021b), however, in the opposite direction, with more errors on plural dependencies. We hypothesize that this difference might be due to marking of the verb form, given that in English, the marked form of the verb is singular, whereas in Italian, it is plural. Related biases were previous reported for humans, a phenomenon known as the Markedness Effect (Bock and Miller, 1991; Vigliocco et al., 1995). The relation between emerging biases in NLMs and humans is an interesting topic for future work.

In LSTM-LMs, the poor performance was predicted by the underlying neural mechanism for grammatical agreement identified in the models (Lakretz et al., 2019, 2021b). The fact that Transformer models perform similarly poorly on these constructions, both casual and masked-language models, and on the same dependency (inner), raises interesting questions. Do transformers use syntactic-processing strategies akin to those emerged in RNN-LMs? And what does that tell us about the data that those models are trained on and about the potential processes that humans may use to process such constructions (Lakretz et al., 2020)?

However, currently, the neural mechanisms underlying syntactic processing in transformers are poorly understood (Belinkov and Glass, 2019). Our findings of below-chance performance by transformer models calls for a further investigation in *how* these models achieve their earlier found successes on syntactic related tasks, and why they generalise so poorly on constructions which only minimally differ (a single three-word prepositional phrase) from the constructions they process well.

Acknowledgements We would like to thank Marco Baroni for his helpful comments on the original manuscript. This work was supported by the Bettencourt-Schueller Foundation and an ERC grant, "NeuroSyntax" project, to S.D.

[3] Note that their performance on an adjacent dependency between a noun and a verb, as in the embedded dependency of Short-Nested, is relatively good and above chance level, which shows that their overall poor performance is not due to experimental-setup issues, such as tokenization.

References

Yonatan Belinkov and James Glass. 2019. Analysis methods in neural language processing: A survey. *Transactions of the Association for Computational Linguistics*, 7:49–72.

Jean-Philippe Bernardy and Shalom Lappin. 2017. Using deep neural networks to learn syntactic agreement. *Linguistic Issues in Language Technology*, 15(2):1–15.

Kathryn Bock and Carol Miller. 1991. Broken agreement. *Cognitive Psychology*, 23(1):45–93.

Noam Chomsky. 2000. Minimalist inquiries: The framework. In Roger Martin, David Michaels, Juan Uriagereka, and Samuel Keyser, editors, *Step by step: Essays on minimalist syntax in honor of Howard Lasnik*, pages 89–155. MIT Press, Cambridge, MA.

Morten Christiansen and Nick Chater. 1999. Toward a connectionist model of recursion in human linguistic performance. *Cognitive Science*, 23(2):157–205.

Axel Cleeremans, David Servan-Schreiber, and James L McClelland. 1989. Finite state automata and simple recurrent networks. *Neural computation*, 1(3):372–381.

Stanislas Dehaene, Florent Meyniel, Catherine Wacongne, Liping Wang, and Christophe Pallier. 2015. The neural representation of sequences: From transition probabilities to algebraic patterns and linguistic trees. *Neuron*, 88(1):2–19.

Julie Franck, Gabriella Vigliocco, and Janet Nicol. 2002. Subject-verb agreement errors in french and english: The role of syntactic hierarchy. *Language and cognitive processes*, 17(4):371–404.

Felix Gers and Jürgen Schmidhuber. 2001. LSTM recurrent networks learn simple context-free and context-sensitive languages. *IEEE Transactions on Neural Networks*, 12(6):1333–1340.

Mario Giulianelli, Jack Harding, Florian Mohnert, Dieuwke Hupkes, and Willem Zuidema. 2018. Under the hood: Using diagnostic classifiers to investigate and improve how language models track agreement information. In *Proceedings of the 2018 EMNLP Workshop BlackboxNLP: Analyzing and Interpreting Neural Networks for NLP*, pages 240–248, Brussels, Belgium. Association for Computational Linguistics.

Yoav Goldberg. 2019. Assessing bert's syntactic abilities. *arXiv preprint arXiv:1901.05287*.

Kristina Gulordava, Piotr Bojanowski, Edouard Grave, Tal Linzen, and Marco Baroni. 2018. Colorless green recurrent networks dream hierarchically. In *Proceedings of NAACL*, pages 1195–1205, New Orleans, LA.

Marc Hauser, Noam Chomsky, and Tecumseh Fitch. 2002. The faculty of language: What is it, who has it, and how did it evolve? *Science*, 298(5598):1569–1579.

John Hewitt, Michael Hahn, Surya Ganguli, Percy Liang, and Christopher D Manning. 2020. Rnns can generate bounded hierarchical languages with optimal memory. *arXiv preprint arXiv:2010.07515*.

Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long short-term memory. *Neural Computation*, 9(8):1735–1780.

Jennifer Hu, Jon Gauthier, Peng Qian, Ethan Wilcox, and Roger P Levy. 2020. A systematic assessment of syntactic generalization in neural language models. *arXiv preprint arXiv:2005.03692*.

Jaap Jumelet, Milica Denic, Jakub Szymanik, Dieuwke Hupkes, and Shane Steinert-Threlkeld. 2021. Language models use monotonicity to assess NPI licensing. In *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*, pages 4958–4969, Online. Association for Computational Linguistics.

Jaap Jumelet, Willem Zuidema, and Dieuwke Hupkes. 2019. Analysing neural language models: Contextual decomposition reveals default reasoning in number and gender assignment. In *Proceedings of the 23rd Conference on Computational Natural Language Learning (CoNLL)*, pages 1–11, Hong Kong, China. Association for Computational Linguistics.

Fred Karlsson. 2007. Constraints on multiple center-embedding of clauses. *Journal of Linguistics*, 43(2):365–392.

Tom Kersten, Hugh Mee Wong, Jaap Jumelet, and Dieuwke Hupkes. 2021. Attention vs non-attention for a shapley-based explanation method. In *Proceedings of Deep Learning Inside Out (DeeLIO): The 2nd Workshop on Knowledge Extraction and Integration for Deep Learning Architectures*, pages 129–139, Online. Association for Computational Linguistics.

Yair Lakretz, Stanislas Dehaene, and Jean-Rémi King. 2020. What limits our capacity to process nested long-range dependencies in sentence comprehension? *Entropy*, 22(4):446.

Yair Lakretz, Théo Desbordes, Jean-Rémi King, Benoît Crabbé, Maxime Oquab, and Stanislas Dehaene. 2021a. Can rnns learn recursive nested subject-verb agreements? *arXiv preprint arXiv:2101.02258*.

Yair Lakretz, Dieuwke Hupkes, Alessandra Vergallito, Marco Marelli, Marco Baroni, and Stanislas Dehaene. 2021b. Mechanisms for handling nested dependencies in neural-network language models and humans. *Cognition*, page 104699.

Yair Lakretz, Germán Kruszewski, Theo Desbordes, Dieuwke Hupkes, Stanislas Dehaene, and Marco Baroni. 2019. The emergence of number and syntax units in LSTM language models. In *Proceedings of NAACL*, pages 11–20, Minneapolis, MN.

Tal Linzen, Emmanuel Dupoux, and Yoav Goldberg. 2016. Assessing the ability of LSTMs to learn syntax-sensitive dependencies. *Transactions of the Association for Computational Linguistics*, 4:521–535.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Roberta: A robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692*.

Rebecca Marvin and Tal Linzen. 2018. Targeted syntactic evaluation of language models. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 1192–1202, Brussels, Belgium. Association for Computational Linguistics.

Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, Ilya Sutskever, et al. 2019. Language models are unsupervised multitask learners. *OpenAI blog*, 1(8):9.

David Servan-Schreiber, Axel Cleeremans, and James L McClelland. 1991. Graded state machines: The representation of temporal contingencies in simple recurrent networks. *Machine Learning*, 7(2-3):161–193.

Koustuv Sinha, Robin Jia, Dieuwke Hupkes, Joelle Pineau, Adina Williams, and Douwe Kiela. 2021. Masked language modeling and the distributional hypothesis: Order word matters pre-training for little. *CoRR*, abs/2104.06644.

Mirac Suzgun, Sebastian Gehrmann, Yonatan Belinkov, and Stuart M Shieber. 2019. Memory-augmented recurrent neural networks can learn generalized dyck languages. *arXiv preprint arXiv:1911.03329*.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in Neural Information Processing Systems*, pages 5998–6008.

Gabriella Vigliocco, Brian Butterworth, and Carlo Semenza. 1995. Constructing subject-verb agreement in speech: The role of semantic and morphological factors. *Journal of Memory and Language*, 34(2):186–215.

Alex Warstadt, Alicia Parrish, Haokun Liu, Anhad Mohananey, Wei Peng, Sheng-Fu Wang, and Samuel R. Bowman. 2020. BLiMP: A benchmark of linguistic minimal pairs for English. In *Proceedings of the Society for Computation in Linguistics 2020*, pages 409–410, New York, New York. Association for Computational Linguistics.

NSP-BERT: A Prompt-based Few-Shot Learner Through an Original Pre-training Task —— Next Sentence Prediction

Yi Sun*, **Yu Zheng***, **Chao Hao**, **Hangping Qiu**†
Army Engineering University of PLA, Nanjing, China
sunyi_lgdx@sina.com, zhengyu87@outlook.com
haochaoleo@163.com, qiuhp_zy@163.com

Abstract

Using prompts to utilize language models to perform various downstream tasks, also known as prompt-based learning or prompt-learning, has lately gained significant success in comparison to the pre-train and fine-tune paradigm. Nonetheless, virtually most prompt-based methods are token-level such as PET based on mask language model (MLM). In this paper, we attempt to accomplish several NLP tasks in the zero-shot and few-shot scenarios using a BERT original pre-training task abandoned by RoBERTa and other models——Next Sentence Prediction (NSP). Unlike token-level techniques, our sentence-level prompt-based method **NSP-BERT** does not need to fix the length of the prompt or the position to be predicted, allowing it to handle tasks such as entity linking with ease. NSP-BERT can be applied to a variety of tasks based on its properties. We present an NSP-tuning approach with binary cross-entropy loss for single-sentence classification tasks that is competitive compared to PET and EFL. By continuing to train BERT on RoBERTa's corpus, the model's performance improved significantly, which indicates that the pre-training corpus is another important determinant of few-shot besides model size and prompt method.[1]

1 Introduction

GPT-2 (up to 1.5B (Radford et al., 2019)) and GPT-3 (up to 175B (Brown et al., 2020)) are ultra-large-scale language models with billions of parameters that have recently demonstrated outstanding performance in various NLP tasks. Compared with previous state-of-the-art fine-tuning methods, they can achieve competitive results without any or with just a limited quantity of training data. Although

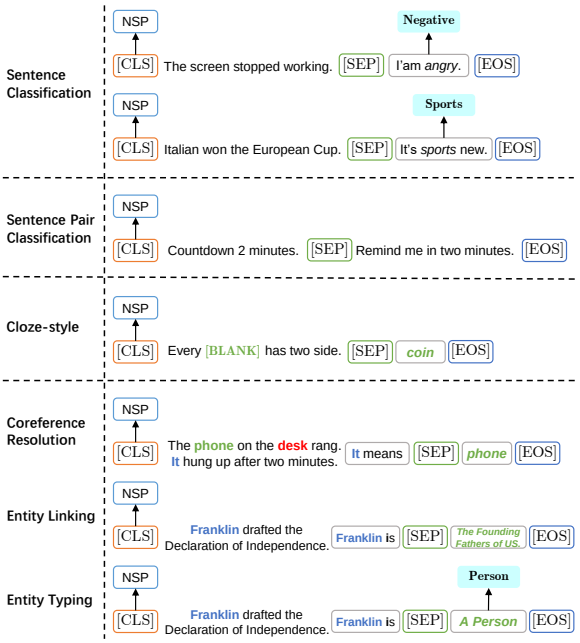

Figure 1: Prompts for various NLP tasks of NSP-BERT.

studies have shown that scaling up the model improves task-agnostic and few-shot performance, some studies have shown that by constructing appropriate prompts for the model, models like BERT (Devlin et al., 2018) or RoBERTa (Liu et al., 2019) can achieve similar performance despite having a parameter count that is several orders of magnitude smaller (Schick and Schütze, 2021b,a; Wang et al., 2021). Since then, the area of natural language processing has seen a fresh wave of developments, including the introduction of a new paradigm known as **prompt-based learning** or **prompt-learning**, which follows the *"pre-train, prompt, and predict"* (Liu et al., 2021) process. In zero-shot and few-shot learning, prompt-learning has achieved a lot of success. Not only does it achieve outstanding performance, prompt-learning better integrates pre-training and downstream tasks and brings NLP tasks closer to human logic and habits.

The input text for the classification task, for ex-

*Equal contribution
†Corresponding author
[1]Our code and pre-trained models are publicly at: https://github.com/sunyilgdx/Prompts4Keras.

Proceedings of the 29th International Conference on Computational Linguistics, pages 3233–3250
October 12–17, 2022.

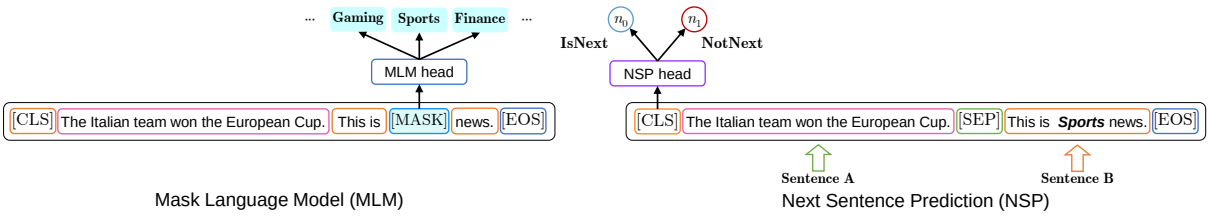

Figure 2: (Left) MLM task for token-level prompt-learning. (Right) NSP task for sentence-level prompt-learning.

ample, "*The Italian team won the European Cup.*", should be assigned to one of the candidate labels, such as *Gaming*, *Sports*, or *Finance*. At this point, the template "*This is* `[MASK]` *news.*" will be added to the original text, and the model will be asked to predict the missing word or span. The model's output will then be mapped to the candidate labels. We could utilize the pre-training tasks of several types of language models (LM) to predict the abovementioned templates, including but not limited to Left-to-right LM (GPT series (Radford et al., 2018, 2019; Brown et al., 2020)), Masked LM (BERT (Devlin et al., 2018), RoBERTa (Liu et al., 2019)), prefix LM (UniLM (Dong et al., 2019; Bao et al., 2020)) and Encoder-decoder LM (T5 (Raffel et al., 2019), BART (Lewis et al., 2020)).

Although most research on prompt-learning has been conducted, the majority of the pre-training tasks used in prompt-learning are token-level, requiring the labels to be mapped to a fixed-length token span (Schick and Schütze, 2021b,a; Cui et al., 2021). On the one hand, when the number of labels grows rapidly, this necessitates a lot of human labor. On the other hand, tasks with variable-length options make Left-to-right LM (L2R LM) or masked LM (MLM) difficult to cope with. The length of each candidate entity's description, for example, varies significantly in the entity linking task.

At the same time, we observed that there is an original sentence-level pre-training object in vanilla BERT——**NSP** (**N**ext **S**entence **P**rediction), which is a binary classification task that predicts whether two sentences appear consecutively within a document or not. Many models, like RoBERTa (Liu et al., 2019) and many others (Conneau and Lample, 2019; Yang et al., 2019; Joshi et al., 2020), have questioned and abandoned this task during pre-training. Nevertheless, based on the task's features and object, we believe it is appropriate to use in prompt-learning.

Unlike most prior works, we present NSP-BERT, a sentence-level prompt-learning method. The paper's main contributions can be summarized as follows:

- We propose the use of NSP, a sentence-level pre-training task for prompt-learning, which can ignore the uncertain length of the label words. Our NSP-BERT has a strong zero-shot learning capacity and can be applied to a wide range of tasks, which is extremely motivating for future work.
- We present NSP-tuning for single-sentence classification tasks. Without abandoning the original NSP head, binary cross-entropy loss is utilized to make the zero-shot capacity of NSP-BERT continue to few-shot by building coupled positive and negative instances.
- By using RoBERTa's corpus to continue pre-training the BERT model, although the computational cost is only about 2% of RoBERTa, our BERT$_{\mathcal{C}_{\text{B+Mix5}}}$ has been greatly improved in both zero-shot and few-shot scenarios. We believe that the effect of pre-training corpus on few-shot learning is decisive, so we suggest that all few-shot learning baselines, even if cannot use the same pre-trained model, should be based on the same pre-training corpus. In this way, a fair comparison can be made.

2 Related Work

2.1 Token-Level and Sentence-Level

Token-Level Prompt-Learning Token-level pre-training tasks, such as MLM (Shown in the left part of Figure 2) (Jiang et al., 2020; Schick and Schütze, 2021b,a) or L2R LM(Radford et al., 2019; Brown et al., 2020; Cui et al., 2021), are commonly used in token-level prompt-learning approaches. Although the expected answer may be in the form of tokens, spans, or sentences in token-level prompt-learning, the predicted answer is always generated token by token. Tokens are usually mapped to the whole vocabulary or a set of candidate words (Petroni et al., 2019; Cui et al., 2021; Han et al., 2021; Adolphs et al., 2021; Hu et al., 2021). Take PET model (Schick and Schütze, 2021b,a) as an example, the

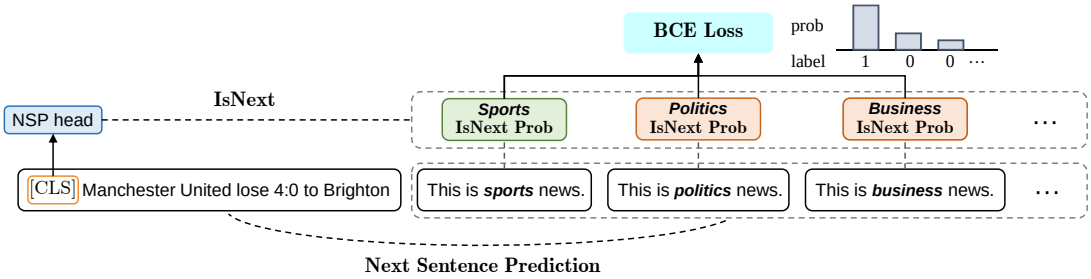

Figure 3: NSP-tuning for single-sentence classification. "*Manchester United lose 4:0 to Brighton*" is the original input, the gold label is *Sports*. The negative instances are building with wrong label *Politics*, *Bussiness*, etc.

sentiment classification input/label pair is reformulated to "**x**: `[CLS]` *The Italian team won the European Cup. This is* `[MASK]` *news.* `[EOS]`, y: *Sports*".

Sentence-Level Prompt-Learning Sentence-level methods concentrate on the relationship between sentences, with the model's output usually mapped to a relationship space. As far as we know, EFL (Wang et al., 2021) is the only sentence-level model. It reformulates NLP tasks into sentence entailment-style tasks. For example, the sentiment classification input/label pair is reformulated to "**x**: `[CLS]` *The Italian team won the European Cup.* `[SEP]` *This is Sports news.*`[EOS]`, y: `Entail`". The output of model is `Entail` or `Not Entail`. The EFL model can perform well on few-shot learning but relies on labeled natural language inference (NLI) datasets like MNLI (Williams et al., 2018).

2.2 Optimization methods

Automated Prompt Manually designed prompts are highly unstable. Sometimes it is necessary to be familiar with the particular task and language model in order to construct a high-quality prompt. As a result, several studies attempt to automatically search for and generate prompts. LM-BFF (Gao et al., 2021) model use conditional likelihood to automatically select labels words, and use T5 (Raffel et al., 2019) to generate templates. AUTOPROMPT (Shin et al., 2020) uses a gradient-guided search to create prompts. Compared to the discrete prompt search methods mentioned above, P-tuning (Liu et al., 2021) employs trainable continuous prompt embeddings on GPT.

Training Strategy There are many optimization methods in prompt-learning. ADAPET (Tam et al., 2021) uses more supervision by decoupling the losses for the label tokens and a label-conditioned MLM objective over the full original input. PTR (Han et al., 2021) incorporates logic rules to compose task-specific prompts with several simple sub-prompts. (Zhao et al., 2021) use content-free inputs (e.g. "N/A") to calibrate the model's output probabilities and improved the performance of GPT-2 and GPT-3.

3 Framework of NSP-BERT

Problem of MLM: Span Prediction As the most important pre-training task of BERT-like models, MLM has been used for prompt-learning in most previous studies, and achieved satisfactory results on GLUE (Wang et al., 2019) and other English datasets or benchmarks. In those English tasks, we can use just one token to map each label. But in some cases, we need more than one token.

$$\mathbf{x}_{input} = \texttt{[CLS]}\ \mathbf{x}\ \text{It was}\ \texttt{[MASK].[EOS]}$$
$$\mathbf{x}_{input} = \texttt{[CLS]}\ \mathbf{x}\ \text{这是}\ \texttt{[MASK][MASK]}\text{新闻}\texttt{.[EOS]}$$

As shown in the above example, in the first English sample, **x** is the original sentence, we can use just one `[MASK]` token to predict the label word "Sports" in a classification task. But in the second Chinese sample, we need `[MASK][MASK]` to map the label word "体育" (which has the same meaning with "Sports"), and use their probability product to represent the probability of the label (detailed description is in the Appendix A.1). As the number of `[MASK]` increases, it becomes difficult for the MLM to predict correctly. At the same time, it is impossible to compare the probability of label mapping words (spans or sentences) with different number of `[MASK]` tokens, entity linking is one of the scenarios. Therefore, especially in the Chinese task, there is a obvious gap between the pre-training and the downstream task.

3.1 Next Sentence Prediction

The next sentence prediction is one of the two basic pre-training tasks (the other is MLM) of the

vanilla BERT model (Devlin et al., 2018) (Shown in the right part of Figure 2). This task inputs two sentences A and B into BERT at the same time to predict whether sentence B comes after sentence A in the same document. During specific training, for 50% of the time, B is the actual next sentence that follows A (IsNext), and for the other 50% of the time, we use a random sentence from the corpus (NotNext).

$$\mathbf{x}_{input} = \texttt{[CLS]}\,\mathbf{x}_i^{(1)}\,\texttt{[SEP]}\,\mathbf{x}_i^{(2)}.\texttt{[EOS]}$$

Let $\mathcal{M}$ denote the model trained on a large-scale corpus. This model is trained on both MLM task and NSP task at the same time. $\mathbf{x}_i^{(1)}$ and $\mathbf{x}_i^{(2)}$ denote sentence A and sentence B, respectively. The model's input is $\mathbf{x}_{input}$, and $q_{\mathcal{M}}$ denotes the output probability of model's NSP head. $\mathbf{s} = \mathbf{W}_{\text{nsp}}(\tanh(\mathbf{W}\mathbf{h}_{[\text{CLS}]} + \mathbf{b}))$ [2], where $\mathbf{h}_{[\text{CLS}]}$ is the hidden vector of [CLS] and $\mathbf{W}_{\text{nsp}}$ is a matrix learned by NSP task, $\mathbf{W}_{\text{nsp}} \in \mathbb{R}^{2\times H}$. The loss function of NSP task $\mathcal{L}_{\text{NSP}} = -\log q_{\mathcal{M}}(n|\mathbf{x})$, where $n \in \{\texttt{IsNext}, \texttt{NotNext}\}$.

$$q_{\mathcal{M}}(n_k|\mathbf{x}_i) = \frac{\exp s(n_k|\mathbf{x}_i^{(1)}, \mathbf{x}_i^{(2)})}{\sum_n \exp s(n|\mathbf{x}_i^{(1)}, \mathbf{x}_i^{(2)})} \quad (1)$$

3.2 Prompts in NSP-BERT

NSP-BERT, like other prompt-based learning methods, requires the construction of appropriate templates for various tasks. In order to make the model have better zero-shot performance and better few-shot initialization, the template's building form must closely match the original NSP task. In this section, we'll show how to construct templates for different tasks (also shown in Figure 1).

In order to apply NSP to zero or few-shot learning, we treat most tasks as multiple-choice tasks. Same as the right side in Figure 2, an NSP-BERT's input can be expressed as:

$$\mathbf{x}_{input} = \texttt{[CLS]}\,\mathbf{x}_i\,\texttt{[SEP]}\,p_i^{(j)}\,\texttt{[EOS]}.$$

We define the template $\mathcal{T}$ as a combination of input $\mathbf{x}_i$ and the prompts, $\mathcal{T}(\mathbf{x}) = \texttt{[CLS]}\,\mathbf{x}\,\texttt{[SEP]}\,\textit{This is ... news.}\,\texttt{[EOS]}$. Unlike prompt-tuning based on MLM (Schick and Schütze, 2021a; Gao et al., 2021) which requires mapping labels to vocabularies, for our NPS-BERT, labels can be mapped to words or phrases of arbitrary length in "...". To map labels to the prompts, we define a verbalizer as a mapping $f : \mathcal{Y} \mapsto \mathcal{P}$. The label $y_i^{(j)}$ can be mapped to prompt $p_i^{(j)} \in \mathcal{P}$.

In single-sentence classification tasks, all samples share the same label space $\mathcal{Y}$, where $|\mathcal{Y}|$ is the number of classes. For label of the jth class $y^{(j)} \in \mathcal{Y}$ can be mapped to prompt $p^{(j)}$. For those tasks where each sample corresponds to different labels, such as cloze-style task, word sense disambiguation, entity linking, we define the label space corresponding to the ith sample as $\mathcal{Y}_i$, and $y_i^{(j)} \in \mathcal{Y}_i$.

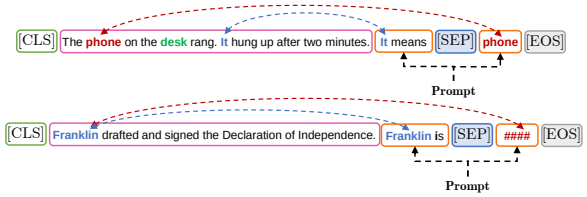

Figure 4: Two-stage prompt, examples in coreference resolution and entity linking/typing tasks.

In tasks such as entity linking, there are more than one entity in the sentence, in order to identify target entity words, we recommend using **two-stage prompt** (as shown in Figure 4) to indicate the target word using natural language descriptions:

- **Stage 1**: Prompt the target word at the end of sentence A. This stage's purpose is to provide enough context for the target word.
- **Stage 2**: Prompt the description of the candidate word sense in sentence B.

Let $p_{i,1}^{(j)}$ and $p_{i,2}^{(j)}$ denote the first and the second part of the prompt. The model's input is:

$$\mathbf{x}_{input} = \texttt{[CLS]}\,\mathbf{x}_i, p_{i,1}^{(j)}\,\texttt{[SEP]}\,p_{i,2}^{(j)}\,\texttt{[EOS]}.$$

For sentence-pair tasks such as text entailment and text matching, since the NSP task is in the form of sentence pairs we still use the same input as the original NSP task.

3.3 Answer Mapping

Because not all datasets can provide contrastive candidate answers (sentiments, topics, idioms, or entities), we propose two answer mapping methods, **candidates-contrast** answer mapping and **samples-contrast** answer mapping, for different situations.

[2] Devlin et al.(2018) use an additional nonlinear layer to pool the hidden vector of [CLS] for NSP task, but not mentioned in their paper.

Candidates-Contrast For datasets with multiple candidates, such as candidate sentiments, candidate topics, candidate idioms and candidate entities. For the above datasets, there is a template $p_i^{(j)}$ (or p_i) corresponding to the label $y_i^{(j)}$ (or y_i), we choose the `IsNext` probability as the output of each candidate answer. The logit of label $y_i^{(j)}$ (the value ranges from 0 to 1, but is not an actual probability) is:

$$q(y_i^{(j)}|\mathbf{x}_i) \propto q_{\mathcal{M}}(n = \texttt{IsNext}|\mathbf{x}_i, p_i^{(j)}) \quad (2)$$

In the prediction stage, we take the highest probability output by $\mathcal{M}$ among the candidates as the final output answer where the condition is `IsNext`:

$$\begin{aligned} \hat{y}_i &= \arg\max_j q(y_i^{(j)}|\mathbf{x}_i) \\ &= \arg\max_j q_{\mathcal{M}}(n = \texttt{IsNext}|\mathbf{x}_i, p_i^{(j)}) \end{aligned} \quad (3)$$

Samples-Contrast For sentence-pair tasks, the `IsNext` output probabilities of most samples are close to 1 (see details in Appendix B.2), which makes it difficult to judge the relationship between two sentences through a single sample. So we propose the samples-contrast answer mapping method (Figure 3), to determine the label of a individual sample by contrast the probability of `IsNext` between samples. To put it simply, by **rank**ing[3] in ascending order, the samples with a relatively higher `IsNext` probability are **divide**d[4] into labels with a higher degree of matching, such as `Entailment`. On the contrary, samples with lower `IsNext` probability will be divided to labels such as `NotEntailment`. This procedure is summarized in Algorithm 1[5].

Considering the fairness of the comparative experiment, we consider two preconditions. One is that a complete development set and a test set can be obtained at the same time; the other is that only the development set can be obtained, and the test samples must be predicted one by one or batch by batch during testing. In our experiment, we use the development set to determine the thresholds of probability, and use these thresholds to predict the test set.

[3]Sort samples in ascending or descending order according to `IsNext` probability.

[4]Divide the dataset (or sample batch) into subsets according to the proportion of each label in development set.

[5]This method is currently only suitable for sentence-pair tasks, and can only be applied in zero-shot scenarios.

Algorithm 1 Samples-Contrast Answer Mapping

Input: Test set $\mathcal{D} = \{\mathbf{x}_i\}_{i=1}^N$, where $\mathbf{x}_i = (\mathbf{x}_i^{(1)}, \mathbf{x}_i^{(2)})$, Oder $o \in$ {"ascending", "descending"}, distribution of labels d, batch size bs.
Output: $\{\mathbf{x}_i, \hat{y}_i\}_{i=1}^N$

for $i = 1, ..., N$ **do**
  $q_i \leftarrow q_{\mathcal{M}}(n = \text{IsNext}|\mathbf{x}_i^{(1)}, \mathbf{x}_i^{(2)})$
end for
$\{\mathcal{B}_j\}_{j=1}^{\lceil \frac{N}{bs} \rceil} \leftarrow$ **divide** $(\mathcal{D}, bs)$
for $j = 1, ..., \lceil \frac{N}{bs} \rceil$ **do**
  $\mathcal{B}'_j = \{\mathbf{x}_{r(1)}, ..., \mathbf{x}_{r(bs)}\} \leftarrow$ **rank**$(\mathcal{B}_j, q_i, o)$
  $\{B_m\}_{m=1}^M \leftarrow$ **divide** $(\mathcal{B}'_j, d)$
  for $i = 1, ..., bs$ **do**
    $\hat{y}_i \leftarrow m$ **where** $\mathbf{x}_i \in B_m$
  end for
end for

3.4 NSP-tuning

Since we treat tasks with candidates as multiple-choice problems, when we need to perform few-shot learning, we need to choose some methods to continue the initialization advantages of NSP-BERT in zero-shot. We name this method NSP-tuning used on few-shot single-sentence classification tasks, as shown in Figure 3.

Building Instances Taking the single-sentence classification as an example, for the ith sample, we take it's gold label y_i^+ as a positive instance $(\mathcal{T}(\mathbf{x}_i, y_i^+), 1)$, while taking the rest of the labels in $\mathcal{Y}$ as negative instances $\{(\mathcal{T}(\mathbf{x}_i, y_i^-), 0)\}_{y_i^- \neq y_i^+, y_i^- \in \mathcal{Y}}^{|\mathcal{Y}|-1}$ and $\{0, 1\}$ represent the labels of the binary classification. Both the positive instance and negative instances of the same sample, a total of $|\mathcal{Y}|$, will be coupled and input to the model in a same batch.

Loss function Since the output probability of `IsNext` has been already normalized to $[0, 1]$ by softmax after a nonlinear layer during pre-training, if we want to do NSP-tuning without changing the structure of the pre-training model, we need to choose the **binary cross-entropy loss** as the loss function. Of course, we can re-initialize the output of $\mathcal{M}$ to implement a multiple-choice method with linear layer+softmax cross-entropy loss same as (Radford and Narasimhan, 2018), but we think this is not conducive to preserving the zero-shot advantage of NSP to few-shot.

		English Tasks							Chinese Tasks			
		SST-2	**MR**	**CR**	**MPQA**	**Subj**	**Yahoo!**	**AGNews**	**EPR.**	**TNEWS(K)**	**CSLDCP**	**IFLY.**
Full	Majority	*50.9*	*50.0*	*50.0*	*50.0*	*50.0*	*10.0*	*25.0*	*50.0*	*6.7*	*1.5*	*0.8*
	Fine-Tuning	*93.6*	*89.0*	*89.3*	*89.3*	*97.0*	*76.5*	*94.7*	*90.0*†	*71.0*†	*68.0*†	*66.0*†
Zero	PET	67.6	65.3	**61.2**	**63.9**	**61.0**	25.6	54.5	60.7	28.0 / 35.6	22.4	34.8
	NSP-BERT	**75.6**	**74.4**	59.4	59.9	53.9	**47.0**	**77.5**	**86.9**	**51.9** / **57.0**	**47.6**	**41.6**
Few	Fine-tuning	77.9±5.9	68.0±9.4	79.1±8.9	65.2±6.3	**89.7±1.1**	61.8±1.5	82.4±1.2	78.7±5.8	51.1±1.1 / 58.0±1.4	51.7±2.1	45.1±2.2
	PET	86.0±1.6	80.0±1.6	**88.9±0.6**	83.3±2.4	86.2±1.5	64.3±1.3	84.2±0.8	82.5±2.0	54.7±1.1 / 61.2±0.9	52.6±1.2	45.9±2.1
	EFL w/ PT	**86.9±1.8**	**80.6±1.2**	88.1±0.9	**86.1±0.7**	86.0±3.3	63.0±1.2	83.8±1.3	84.8±1.6	53.2±1.5 / 59.2±1.6	52.0±1.6	47.9±1.5
	EFL w/o PT	81.2±5.1	76.1±9.1	79.2±4.0	79.1±1.6	75.1±9.4	60.8±4.2	84.6±0.7	84.6±2.1	54.7±1.3 / 60.3±1.7	53.8±0.9	49.5±1.2
	NSP-BERT	86.8±1.3	80.5±1.5	86.0±2.2	83.9±1.1	86.4±1.8	**64.5±0.5**	**85.9±0.8**	**87.7±0.7**	**55.7±1.0** / **61.6±0.9**	**55.0±1.5**	**49.5±1.1**

Table 1: Main zero-shot and few-shot learning results on single-sentence classification tasks. In addition to the accuracy, we also report the standard deviation for few-shot learning. For English tasks, we use vanilla BERT-LARGE. For Chinese tasks, we use UER's Chinese BERT-BASE. Full: full training; Zero: zero-shot; Few: few-shot; †: human performance; Majority: majority class; EFL w/ PT: few-shot tuning of EFL with pre-training on MNLI; EFL w/o PT: few-shot tuning of without pre-training on MNLI; TNEWS(K): use the keyword (K) field or not.

4 Experiment

4.1 Tasks and Datasets

English Datasets For English tasks, following (Gao et al., 2021; Hu et al., 2021; Liang et al., 2022), we choose 7 single-sentence and 5 sentence-pair English tasks. See details in Appendix B.1.

Chinese Datasets For Chinese tasks, we choose FewCLUE (Xu et al., 2021), a Chinese Few-shot Learning Evaluation Benchmark, which contains 9 NLU tasks in Chinese, with 4 single-sentence tasks, 3 sentence-pair tasks and 2 reading comprehension tasks. Additionally, we select the entity linking dataset DuEL2.0[6] to verify the word sense disambiguation ability. And we divide DuEL2.0 into two parts: DuEL2.0-L (entity linking) and DuEL2.0-T (entity typing).

4.2 Baselines

Fine-Tuning Standard fine-tuning of the pre-trained language model on the FewCLUE training set. The models are fine-tuned with cross entropy loss and using the BERT-style model's hidden vector of `[CLS]` $\mathbf{h}_{[\text{CLS}]}$ with a classification layer $\text{softmax}(\mathbf{W}\mathbf{h}_{[\text{CLS}]})$, where $\mathbf{W} \in \mathbb{R}^{|\mathcal{Y}| \times H}$, $|\mathcal{Y}|$ is the number of labels.

Prompt-based methods Since our method is a brand-new basic prompt-learning method, our main purpose is to demonstrate its effectiveness compared to MLM-like methods, and we think it is not necessary to compare with more complex methods such as continuous prompt or automatic prompt methods. Therefore we choose token-level model PET (Schick and Schütze, 2021b,a) based on MLM and sentence-level model EFL[7] (Wang et al., 2021) based on entailment as two baselines.

4.3 Experiment Settings

Evaluation Protocol For few-shot learning, we follow the evaluation protocol adopted in (Gao et al., 2021; Liang et al., 2022) and assume K samples per class for training set. For English tasks the K of training set is set to 16, and the size of the development set is 10 times the size of the training set. The number K of FewCLUE has been set to 8 or 16 according to Xu et al. (2021). For each experiment, we run 5 experiments with 5 different training and development set (split by 5 fixed random seed) and report the average results and standard deviations.

Language Models In order to conduct comparative experiments fairly, for our main experiments, we use the same pre-trained language model for the same dataset. For English tasks, we adopt the vanilla English BERT-LARGE[8]. For Chinese tasks, we adopt the Chinese BERT-BASE[9] trained by UER using MLM and NSP (Zhao et al., 2019).

Hyper-parameters For few-shot learning, we train 10 epochs on all the datasets. We set learning rate as 2e-5 for English tasks, and 1e-5 for Chinese tasks. The batch size is 8. All baselines use the same hyper-parameters described above.

4.4 Main Results

The Table 1 reports the main results on 7 English and 4 Chinese single-sentence classification tasks.

[6]https://aistudio.baidu.com/aistudio/competition/detail/83

[7]We use MNLI(Williams et al., 2018) and OCNLI(Hu et al., 2020) to pre-train EFL.

[8]https://github.com/google-research/bert

[9]https://github.com/dbiir/UER-py

		Model	Corpus	English Tasks						
				SST-2	MR	CR	MPQA	Subj	Yahoo!	AGNews
Zero	PET	RoBERTa	$\mathcal{C}_{\mathrm{B}}$	81.2	75.6	76.6	63.3	**63.6**	18.7	47.8
			$\mathcal{C}_{\mathrm{R}}$	**83.6**	**80.8**	**79.5**	**67.6**	53.6	**25.6**	**54.5**
		BERT	$\mathcal{C}_{\mathrm{B}}$	67.6	65.3	61.2	63.9	**61.0**	25.6	**54.5**
			$\mathcal{C}_{\mathrm{B+Mix5}}$	**75.0**	**70.1**	**67.4**	**64.2**	55.3	**28.5**	38.4
	NSP-BERT	BERT	$\mathcal{C}_{\mathrm{B}}$	75.6	74.4	59.4	59.9	**53.9**	47.0	**77.5**
			$\mathcal{C}_{\mathrm{B+Mix5}}$	**81.2**	**78.3**	**76.9**	**72.4**	53.0	**56.8**	75.8
Few	PET	RoBERTa	$\mathcal{C}_{\mathrm{B}}$	88.6±1.5	83.9±0.8	87.8±0.7	82.0±1.1	82.8±5.6	65.2±1.3	86.0±0.4
			$\mathcal{C}_{\mathrm{R}}$	**91.7±0.6**	**88.0±0.5**	**91.5±0.9**	**85.6±2.1**	**87.8±2.2**	**68.9±1.0**	**87.8±0.9**
		BERT	$\mathcal{C}_{\mathrm{B}}$	85.3±1.7	80.3±2.1	89.2±0.3	83.3±2.4	85.4±1.9	64.3±1.3	84.0±1.0
			$\mathcal{C}_{\mathrm{B+Mix5}}$	**87.6±0.9**	**85.0±0.8**	**89.6±0.8**	**85.0±1.7**	**90.5±1.2**	**68.4±0.7**	**87.8±0.6**
	NSP-BERT	BERT	$\mathcal{C}_{\mathrm{B}}$	86.7±2.1	80.3±1.8	86.7±1.7	83.9±1.1	86.6±0.9	64.5±0.5	85.9±0.8
			$\mathcal{C}_{\mathrm{B+Mix5}}$	**89.4±0.7**	**83.3±1.1**	**88.7±1.0**	**85.3±1.0**	**92.1±1.1**	**68.3±1.3**	**87.6±0.5**

Table 2: Impact of pre-training corpus. $\mathcal{C}_{\mathrm{B}}$: pre-training from scratch with BERT's corpus; $\mathcal{C}_{\mathrm{R}}$: pre-training from scratch with RoBERTa's corpus; $\mathcal{C}_{\mathrm{B+Mix5}}$: continue pre-training with RoBERTa's corpus based on vanilla BERT.

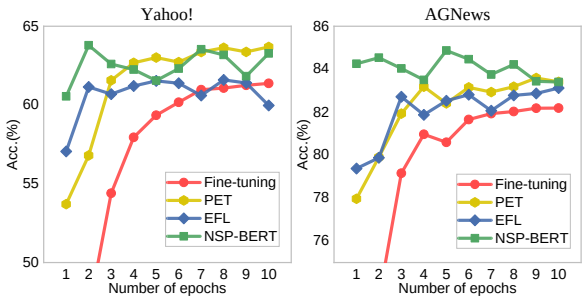

Figure 5: The accuracy of the 4 methods for each epoch during few-shot training on Yahoo! and AGNews.

Since we use the same pre-trained language model for all methods, this experiment is fair enough. It is clear that our NSP-BERT offers distinct advantages in zero-shot scenario, particularly for multi-topic classification tasks such as Yahoo!, AGNews, and all Chinese datasets. In few-shot scenario. its performance is comparable to the MLM-based PET (Schick and Schütze, 2021a) on the most datasets. Compared with EFL (Wang et al., 2021) without pre-training on the NLI dataset, NSP-BERT is much better. Our NSP-BERT has the fastest convergence speed based on convergence curves, as shown in Figure 5. NSP-BERT usually achieves the best performance during the first few epochs.

Ablation studies on NSP-tuning It can be seen from Table 3 that coupling positive and negative samples + BCE loss function is the most effective and robust way of NSP-tuning. Other modifications in the table will degrade the performance of the model and make the results unstable. We believe this is due to the special output of the NSP Head, and re-initialization will lose the knowledge gained during pre-training.

	SST-2	MR	CR	MPQA
NSP-BERT	86.8±1.3	**80.5±1.5**	**86.0±2.2**	**83.9±1.1**
coupled→decouple	**86.8±1.2**	78.9±2.3	85.8±9.5	81.5±5.8
BCE→softmax	83.8±5.0	76.4±6.4	80.5±10.0	73.3±9.5
w/o NSP head	83.8±6.5	74.3±9.2	79.0±8.1	73.2±10.1
linear head+softmax	80.2±7.6	71.9±12.3	82.6±6.7	73.8±11.1

Table 3: Ablation studies of NSP-BERT on vanilla English BERT-Large. coupled→decouple: change coupled positive and negative samples to decoupled; BCE→softmax: change binary cross-entropy loss to softmax loss; w/o NSP head: use an initialized sigmoid head; linear head+softmax: use an initialized sigmoid head and softmax loss.

Impact of Pre-training Corpus Compared with the RoBERTa model, the original BERT model has a large gap in the pre-training corpus. BERT is only pre-trained on Wikipedia and BookCorpus(Zhu et al., 2015), and the size is about 16GB, while RoBERTa additionally uses CC-News[10], OpenWebText (Gokaslan and Cohen, 2019) and Stories(Trinh and Le, 2018) corpus, which is 145GB more. We use the above 5 corpora[11] to pre-train the vanilla BERT model incrementally. Due to the limited computing power, our total training steps are about 30% of the BERT model and 2% of the RoBERTa model. As shown in Table 2, although it has not yet reached the level of RoBERTa, our BERT model ($\mathrm{BERT}_{\mathcal{C}_{\mathrm{B+Mix5}}}$) has greatly improved the performance of zero-shot and few-shot learning, and this improvement even exceeds the changes brought by the prompt method.

[10]https://commoncrawl.org/2016/10/news-dataset-available/

[11]Since there is no public Stories corpus, we refer to the construction method of (Trinh and Le, 2018) and build it on the basis of CC-100 (Conneau et al., 2020).

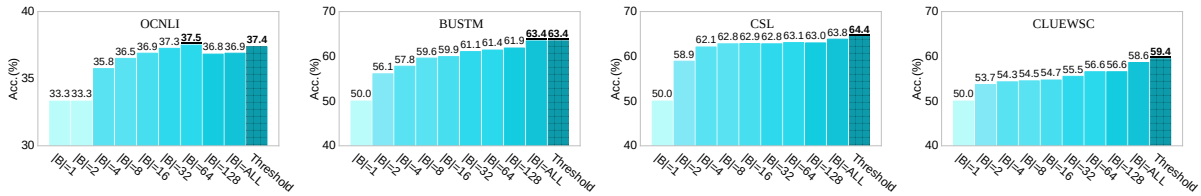

Figure 6: The performance of the samples-contrast answer mapping method under different preconditions on OCNLI, BUSTM, CSL and CLUEWSC. Batch size $|\mathcal{B}| \in \{1, 2, ..., 128, \text{ALL}\}$, when the batch size is 1 (1 and 2 for OCNLI), the result is a random guess, when the batch size is ALL, indicating that the entire test set is obtained at one time. `Thresholds` means that the thresholds are obtained through the dev set, and then used for the prediction of the test set.

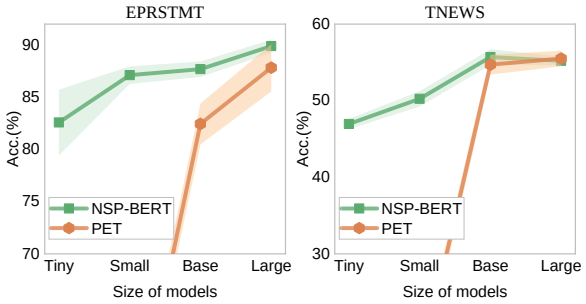

Figure 7: Accuracy of PET and NSP-BERT on EPRSTMT and TNEWS under 4 different model sizes.

Impact of Model Size Only under the premise of fixing the same pre-training corpus, we can verify the effect of model size on NSP-BERT. We carefully selected 4 sizes of UER's BERT (tiny, small, base and large) trained on same corpus for validation on two datasets, EPRSTMT and TNEWS. Figure 7 shows the impact of different sizes of models on NSP-BERT and PET, it can be seen that our method is still very competitive on small models[12].

4.5 Applications of NSP-BERT

We validate applications of NSP-BERT on the tasks shown in Table 4, including NLI (OCNLI, MNLI, SNLI, QNLI and RTE), text matching (BUSTM), keyword recognition (CSL), Chinese idiom cloze test (ChID), and coreference resolution (CLUEWSC). In these tasks, the zero-shot learning prediction ability of NSP-BERT is demonstrated with the help of the sample-contrast method. From Figure 6, we can see that even a small contrast batch size can help the sentence-pair tasks, and as the batch size increases, this improvement becomes more obvious and tends to be stable.

Our NSP-BERT can be applied to the task of entity typing, and can even handle entity linking task. The difficulty of entity linking for MLM-based model such as PET is that the description of the entity is of variable length. In these tasks with more than one target words or entity, the effect of two-stage prompt is obvious, see Table 5.

	Chinese Tasks				
	OCNLI	**BUSTM**	**CSL**	**WSC**	**ChID**
Majority	*38.1*	*50.0*	*50.0*	*50.0*	*14.3*
PET	**40.3**	50.6	52.2	54.7	**57.6**
NSP-BERT	37.4	**63.4**	**64.4**	**59.4**	52.0
	English Tasks				
	MNLI-m	**MNLI-mm**	**SNLI**	**QNLI**	**RTE**
Majority	*32.7*	*33.0*	*33.8*	*49.5*	*52.7*
PET	**47.1**	**46.0**	36.0	49.0	51.6
NSP-BERT	39.4	39.2	**43.4**	**67.6**	**55.6**

Table 4: Applications of NSP-BERT on FewCLUE tasks in zero-shot scenario. We report accuracy for all datasets. We only use the candidate-contrast method on ChID, and use the sample-contrast method on the rest of the datasets.

	DuEL2.0-L	**DuEL2.0-T**
PET	-	40.0
NSP-BERT	61.2 / 69.7↑	31.4 / 40.0↑

Table 5: Word sense disambiguation task. DuEL2.0-L: DuEL2.0 entity linking part; DuEL2.0-T: DuEL2.0 entity typing part. The left side of the slash is the one-stage prompt, and the right side is the two-stage prompt.

5 Conclusion

In this paper, we show that NSP can also be an apposite zero-shot or few-shot learner same as MLM. This not only provides a new route for prompt-learning, but also makes us rethink the role of sentence-level pre-training tasks. At the same time, we continue to pre-train the BERT model with a small amount of computing power, and its performance improves significantly on both zero-shot

[12]PET fails to fit on tiny and small models for no reason.

and few-shot learning, whether to use PET or NSP-BERT. We believe that not only the size of the model, but also the pre-training corpus, both determine the upper limit of the model's ability on few-shot learning.

6 Acknowledgements

We thank Su Jianlin for the repository bert4keras and the spirit of open source. We thank the insightful comments from the anonymous reviewers.

References

Leonard Adolphs, Shehzaad Dhuliawala, and Thomas Hofmann. 2021. How to query language models?

Hangbo Bao, Li Dong, Furu Wei, Wenhui Wang, Nan Yang, Xiaodong Liu, Yu Wang, Songhao Piao, Jianfeng Gao, Ming Zhou, and Hsiao-Wuen Hon. 2020. Unilmv2: Pseudo-masked language models for unified language model pre-training. *arXiv preprint arXiv:2002.12804*.

Roy Bar Haim, Ido Dagan, Bill Dolan, Lisa Ferro, Danilo Giampiccolo, Bernardo Magnini, and Idan Szpektor. 2006. The second PASCAL recognising textual entailment challenge.

Luisa Bentivogli, Peter Clark, Ido Dagan, and Danilo Giampiccolo. 2009. The fifth PASCAL recognizing textual entailment challenge. In *TAC*.

Samuel Bowman, Gabor Angeli, Christopher Potts, and Christopher D Manning. 2015. A large annotated corpus for learning natural language inference. In *Proc. of EMNLP*.

Tom B. Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel M. Ziegler, Jeffrey Wu, Clemens Winter, Christopher Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. 2020. Language models are few-shot learners. In *Proc. of NeurIPS*.

Alexis Conneau, Kartikay Khandelwal, Naman Goyal, Vishrav Chaudhary, Guillaume Wenzek, Francisco Guzmán, Edouard Grave, Myle Ott, Luke Zettlemoyer, and Veselin Stoyanov. 2020. Unsupervised cross-lingual representation learning at scale. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 8440–8451, Online. Association for Computational Linguistics.

Alexis Conneau and Guillaume Lample. 2019. Cross-lingual language model pretraining. In *Proc. of NeurIPS*.

Leyang Cui, Yu Wu, Jian Liu, Sen Yang, and Yue Zhang. 2021. Template-based named entity recognition using bart. In *Proc. of ACL*.

Yiming Cui, Wanxiang Che, Ting Liu, Bing Qin, Ziqing Yang, Shijin Wang, and Guoping Hu. 2019. Pre-training with whole word masking for chinese bert. *arXiv preprint arXiv:1906.08101*.

Ido Dagan, Oren Glickman, and Bernardo Magnini. 2005. The PASCAL recognising textual entailment challenge. In *Proc. of ICML*.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina N. Toutanova. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. In *Proc. of NAACL*.

Li Dong, Nan Yang, Wenhui Wang, Furu Wei, Xiaodong Liu, Yu Wang, Jianfeng Gao, Ming Zhou, and Hsiao-Wuen Hon. 2019. Unified language model pre-training for natural language understanding and generation. In *Proc. of NeurIPS*.

Tianyu Gao, Adam Fisch, and Danqi Chen. 2021. Making pre-trained language models better few-shot learners. In *Proc. of ACL*.

Danilo Giampiccolo, Bernardo Magnini, Ido Dagan, and William B Dolan. 2007. The third pascal recognizing textual entailment challenge. In *Proceedings of the ACL-PASCAL workshop on textual entailment and paraphrasing*.

Aaron Gokaslan and Vanya Cohen. 2019. Openwebtext corpus. `http://Skylion007.github.io/OpenWebTextCorpus`.

Xu Han, Weilin Zhao, Ning Ding, Zhiyuan Liu, and Maosong Sun. 2021. Ptr: Prompt tuning with rules for text classification. *arXiv preprint arXiv:2105.11259*.

Hai Hu, Kyle Richardson, Xu Liang, Li Lu, Sandra Kübler, and Larry Moss. 2020. OCNLI: Original Chinese natural language inference. In *Findings of Empirical Methods for Natural Language Processing (Findings of EMNLP)*.

Minqing Hu and Bing Liu. 2004. Mining and summarizing customer reviews. In *Proc. of KDD*.

Shengding Hu, Ning Ding, Huadong Wang, Zhiyuan Liu, Juanzi Li, and Maosong Sun. 2021. Knowledgeable prompt-tuning: Incorporating knowledge into prompt verbalizer for text classification.

Luyao Huang, Chi Sun, Xipeng Qiu, and Xuanjing Huang. 2019. Glossbert: Bert for word sense disambiguation with gloss knowledge. In *Proc. of EMNLP*.

LTD. IFLYTEK CO. 2019. Iflytek: a multiple categories chinese text classifier. *competition official website, http://challenge.xfyun.cn/2019/gamelist*.

Zhengbao Jiang, Frank F. Xu, Jun Araki, and Graham Neubig. 2020. How can we know what language models know. In *Proc. of EMNLP*.

Mandar Joshi, Danqi Chen, Yinhan Liu, Daniel S. Weld, Luke Zettlemoyer, and Omer Levy. 2020. Spanbert: Improving pre-training by representing and predicting spans. *Transactions of the Association for Computational Linguistics*.

Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Veselin Stoyanov, and Luke Zettlemoyer. 2020. Bart: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. In *Proc. of ACL*.

Xiaozhuan Liang, Ningyu Zhang, Siyuan Cheng, Zhen Bi, Zhenru Zhang, Chuanqi Tan, Songfang Huang, Fei Huang, and Huajun Chen. 2022. Contrastive demonstration tuning for pre-trained language models.

Pengfei Liu, Weizhe Yuan, Jinlan Fu, Zhengbao Jiang, Hiroaki Hayashi, and Graham Neubig. 2021. Pre-train, prompt, and predict: A systematic survey of prompting methods in natural language processing.

Xiao Liu, Yanan Zheng, Zhengxiao Du, Ming Ding, Yujie Qian, Zhilin Yang, and Jie Tang. 2021. Gpt understands, too. *arXiv preprint arXiv:2103.10385*.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Roberta: A robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692*.

Conversational-AI Center of OPPO XiaoBu. 2021. Bustm: Oppo xiaobu dialogue short text matching dataset.

Bo PANG. 2005. Seeing stars: Exploiting class relationships for sentiment categorization with respect to rating scales. In *Proc. of ACL*.

Bo Pang and Lillian Lee. 2004. A sentimental education: sentiment analysis using subjectivity summarization based on minimum cuts. In *Proc. of ACL*.

Fabio Petroni, Tim Rocktäschel, Patrick S. H. Lewis, Anton Bakhtin, Yuxiang Wu, Alexander H. Miller, and Sebastian Riedel. 2019. Language models as knowledge bases. In *Proc. of EMNLP*.

Alec Radford and Karthik Narasimhan. 2018. Improving language understanding by generative pre-training.

Alec Radford, Karthik Narasimhan, Tim Salimans, and Ilya Sutskever. 2018. Improving language understanding by generative pre-training (2018).

Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, Ilya Sutskever, et al. 2019. Language models are unsupervised multitask learners. *OpenAI blog*.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J. Liu. 2019. Exploring the limits of transfer learning with a unified text-to-text transformer. *Journal of Machine Learning Research*.

Pranav Rajpurkar, Jian Zhang, Konstantin Lopyrev, and Percy Liang. 2016. Squad: 100,000+ questions for machine comprehension of text. In *Proc. of EMNLP*.

Timo Schick and Hinrich Schütze. 2021a. Exploiting cloze-questions for few-shot text classification and natural language inference. In *Proc. of EACL*.

Timo Schick and Hinrich Schütze. 2021b. It's not just size that matters: Small language models are also few-shot learners. In *Proc. of NAACL*.

Taylor Shin, Yasaman Razeghi, Robert L. Logan, Eric Wallace, and Sameer Singh. 2020. Autoprompt: Eliciting knowledge from language models with automatically generated prompts. In *Proc. of EMNLP*.

Livio Baldini Soares, Nicholas Arthur FitzGerald, Jeffrey Ling, and Tom Kwiatkowski. 2019. Matching the blanks: Distributional similarity for relation learning. In *Proc. of ACL*.

Richard Socher, Alex Perelygin, Jean Wu, Jason Chuang, Christopher D Manning, Andrew Y Ng, and Christopher Potts. 2013. Recursive deep models for semantic compositionality over a sentiment treebank. In *Proc. of EMNLP*.

Derek Tam, Rakesh R. Menon, Mohit Bansal, Shashank Srivastava, and Colin Raffel. 2021. Improving and simplifying pattern exploiting training. *arXiv preprint arXiv:2103.11955*.

Trieu H Trinh and Quoc V Le. 2018. A simple method for commonsense reasoning. *arXiv preprint arXiv:1806.02847*.

Alex Wang, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel R Bowman. 2019. Glue: A multi-task benchmark and analysis platform for natural language understanding. In *Proc. of ICLR*.

Sinong Wang, Han Fang, Madian Khabsa, Hanzi Mao, and Hao Ma. 2021. Entailment as few-shot learner. *arXiv preprint arXiv:2104.14690*.

Junqiu Wei, Xiaozhe Ren, Xiaoguang Li, Wenyong Huang, Yi Liao, Yasheng Wang, Jiashu Lin, Xin Jiang, Xiao Chen, and Qun Liu. 2019. Nezha: Neural contextualized representation for chinese language understanding. *arXiv preprint arXiv:1909.00204*.

Janyce Wiebe, Theresa Wilson, and Claire Cardie. 2005. Annotating expressions of opinions and emotions in language. *Language resources and evaluation*.

Adina Williams, Nikita Nangia, and Samuel Bowman. 2018. A broad-coverage challenge corpus for sentence understanding through inference. In *Proc. of NAACL*.

Adina Williams, Nikita Nangia, and Samuel R. Bowman. 2018. A broad-coverage challenge corpus for sentence understanding through inference. In *Proc. of NAACL*.

Shanchan Wu and Yifan He. 2019. Enriching pre-trained language model with entity information for relation classification. In *Proc. of CIKM*.

Liang Xu, Hai Hu, Xuanwei Zhang, Lu Li, Chenjie Cao, Yudong Li, Yechen Xu, Kai Sun, Dian Yu, Cong Yu, Yin Tian, Qianqian Dong, Weitang Liu, Bo Shi, Yiming Cui, Junyi Li, Jun Zeng, Rongzhao Wang, Weijian Xie, Yanting Li, Yina Patterson, Zuoyu Tian, Yiwen Zhang, He Zhou, Shaoweihua Liu, Zhe Zhao, Qipeng Zhao, Cong Yue, Xinrui Zhang, Zhengliang Yang, Kyle Richardson, and Zhenzhong Lan. 2020. Clue: A chinese language understanding evaluation benchmark. In *Proc. of COLING*.

Liang Xu, Xiaojing Lu, Chenyang Yuan, Xuanwei Zhang, Hu Yuan, Huilin Xu, Guoao Wei, Xiang Pan, and Hai Hu. 2021. Fewclue: A chinese few-shot learning evaluation benchmark.

Zhilin Yang, Zihang Dai, Yiming Yang, Jaime G. Carbonell, Ruslan Salakhutdinov, and Quoc V. Le. 2019. Xlnet: Generalized autoregressive pretraining for language understanding. In *Proc. of NeurIPS*.

Tianyi Zhang, Felix Wu, Arzoo Katiyar, Kilian Q Weinberger, and Yoav Artzi. 2021. Revisiting few-sample bert fine-tuning. In *Proc. of ICLR*.

Xiang Zhang, Junbo Zhao, and Yann LeCun. 2015. Character-level convolutional networks for text classification. *Advances in neural information processing systems*, 28.

Tony Z. Zhao, Eric Wallace, Shi Feng, Dan Klein, and Sameer Singh. 2021. Calibrate before use: Improving few-shot performance of language models.

Zhe Zhao, Hui Chen, Jinbin Zhang, Xin Zhao, Tao Liu, Wei Lu, Xi Chen, Haotang Deng, Qi Ju, and Xiaoyong Du. 2019. Uer: An open-source toolkit for pre-training models. In *Proc. of EMNLP*.

Chujie Zheng, Minlie Huang, and Aixin Sun. 2019. Chid: A large-scale chinese idiom dataset for cloze test. In *Proc. of ACL*.

Zexuan Zhong and Danqi Chen. 2021. A frustratingly easy approach for entity and relation extraction. In *Proc. of NAACL*.

Yukun Zhu, Ryan Kiros, Rich Zemel, Ruslan Salakhutdinov, Raquel Urtasun, Antonio Torralba, and Sanja Fidler. 2015. Aligning books and movies: Towards story-like visual explanations by watching movies and reading books. In *Proceedings of the IEEE international conference on computer vision*, pages 19–27.

A Models

A.1 Probability Formula

We compared the output probability formulas of different zero-shot prompt-learning models include our NSP-BERT. The following description is a general situation, assuming that each label it mapped to a span with a length is greater than or equal to 1. When the length of the label word is equal to 1, the form of the pre-training and downstream tasks tend to be unified. When the length is greater than 1, there is a gap between them, even we use the model pre-trained by whole word masking (Cui et al., 2019) or span masking (Joshi et al., 2020).

PET-ZERO Denote the token in position i as t_i, the label span will be replaced by `[MASK]`$_{l:r}$. When ignoring special tokens such as `[CLS]` and `[PAD]`, the input of PET-ZERO is:

$$\mathbf{x}_{input} = t_1, ..., \texttt{[MASK]}_l, ..., \texttt{[MASK]}_r, ... \tag{4}$$

The output probability for label $y_i^{(j)}$ is:

$$q(y_i^{(j)}|\mathbf{x}_i) = \underset{1 \leqslant j \leqslant M}{\text{softmax}}(\prod_{l \leqslant v \leqslant r} q_{\mathcal{M}_{\text{MLM}}}(t_v^{(j)}|\mathbf{x}_{input})). \tag{5}$$

NSP-BERT For our NSP-BERT, the label span $t_{l:r}^{(j)}$ will be replaced in turn:

$$\mathbf{x}_{input}^{(j)} = t_1, ..., \texttt{[SEP]}, ..., t_l^{(j)}, ..., t_r^{(j)}, ... \tag{6}$$

The output probability for label $y_i^{(j)}$ is:

$$q(y_i^{(j)}|\mathbf{x}_i) = \underset{1 \leqslant j \leqslant M}{\text{softmax}}(q_{\mathcal{M}_{\text{NSP}}}(\mathbf{x}_{input}^{(j)})). \tag{7}$$

A.2 Parameters of Models

For FewCLUE, we use the Chinese vanilla-BERT-BASE pre-trained by UER (Zhao et al., 2019) for the main results of our NSP-BERT. We also report the results of the other scales (tiny, small and large) model. Following the implementation of (Xu et al., 2021), we use Chinese RoBERTa-wwm-ext-BASE pre-trained by HFL (Cui et al., 2019) and NEZHA-Gen (Wei et al., 2019) for the baselines.

For English datasets, following the implementation [13] of (Gao et al., 2021). We use vanilla-BERT-LARGE pre-trained by Google (Devlin et al., 2018) for our NSP-BERT, and RoBERTa-LARGE[14] for the baselines.

Table 6 shows the hyperparameters of the models used in our experiment. The English and Chinese models are a little different in total parameters, mainly due to the different vocabulary size. It should be noted that not all pre-trained models fully stored NSP head and MLM head, so we need to select deliberately.

Model	L	H	A	Total Parameters ZH / EN
RoBERTa	12	768	12	102M / -
RoBERTa-LARGE	12	768	12	- / 355M
BERT-TINY	3	384	6	14M / -
BERT-SMALL	6	512	8	31M / -
BERT-BASE	12	768	12	102M / -
BERT-LARGE	24	1024	16	327M / 355M

Table 6: The parameters of different models used in our experiment. L: number of layers; H: hidden size; A: number of self-attention heads; "-": not used in our paper; ZH: Chinese model; EN: English model.

A.3 Others

Marks and Two-stage prompt In the Figure 8, we compare the markers that usually appear in supervised training (Huang et al., 2019; Soares et al., 2019; Wu and He, 2019; Zhong and Chen, 2021). The marker are special tokens such as `[noun]`, `[pron]` and `[e]`. They are usually added before and after the target words. The two-stage prompt plays the same role as the markers, but it uses a natural language description method.

B More Details

B.1 Datasets

FewCLUE FewCLUE (Xu et al., 2021) is a Chinese few-shot learning evaluation benchmark with 9 Chinese NLU tasks in total. There are 4 single-sentence tasks which are EPRSTMT, TNEWS, CLSDCP and IFLYTEK. EPRSTMT is a binary sentiment analysis dataset for E-commerce reviews. TNEWS (Xu et al., 2020) is a short text classification for news title with 15 topics. CSLDCP is a text classification dataset including abstracts from a variety of Chinese scientific papers and with 67 categories in total. IFLYTEK (IFLYTEK CO., 2019) is a long text classification dataset for App descriptions. There are 3 sentence-pair tasks which are OCNLI, BUSTM and CSL. OCNLI (Hu et al., 2020) is an original Chinese NLI tasks. BUSTM (of OPPO XiaoBu, 2021) is a dialogue short text

[13] https://github.com/princeton-nlp/LM-BFF

[14] https://github.com/pytorch/fairseq/tree/main/examples/roberta

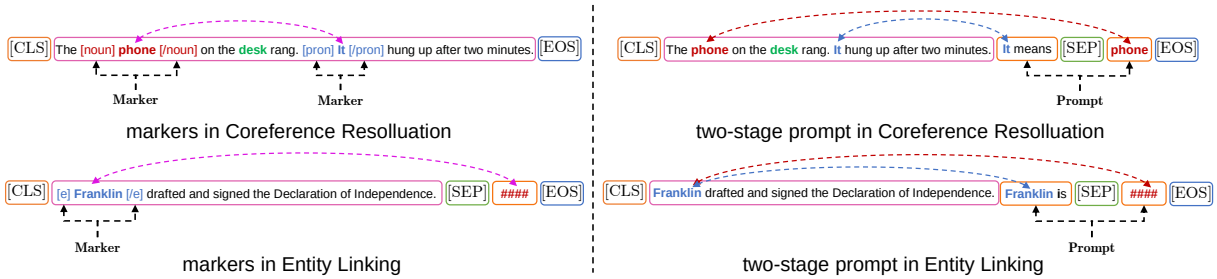

Figure 8: The comparison of markers (Left) and two-stage prompt (Right), examples in coreference resolution and entity linking/typing tasks.

matching task. CSL is a abstract-keywords matching task. There are other two tasks ChID and CLUEWSC. ChID (Zheng et al., 2019) is a Chinese idiom cloze test dataset. CLUEWSC is a coreference resolution task.

For all the datasets in FewCLUE, we evaluate our model on the public test set. Although FewCLUE provides a large number of unlabeled samples, we did not use them in the our experiment, so the results are unable to be compared with the results on the leaderboard[15]. For dataset TNEWS, we did not use the information of keywords following (Xu et al., 2021). We treat CLUEWSC as a sentence-pair task due to its data characteristics.

DuEL2.0 We divide DuEL2.0 into two parts. In the first part, the entity linking part, there are 26586 samples. All the samples' mention can be mapped to single or multiple entities in the knowledge base, and each mention can be linked to 5.37 entities on average. In the second part, the entity typing part, there are 6465 samples. Those samples' mention cannot be found in the knowledge base, but they will be divided into their corresponding upper entity types. There are a total of 24 upper entity types, and we do not remove the `Other` type. When performing the entity linking part, we only use the entity's summary information, without using more entity triples.

Entity Linking	Ave. Entities	Entity Tpying	Types
26586	5.37	6465	24

Table 7: Since the DuEL2.0's test set is not public, we use the dev set to test our model. The the number of the original text lines is 10000. According to the predicted target (entities in knowledge base or upper types), we manually divide it into two parts, entity linking and entity typing.

English Datasets Following (Gao et al., 2021; Hu et al., 2021; Liang et al., 2022), we evaluate our model on 7 single-sentence and 5 sentence-pair English tasks. For the datasets SST-2 (Socher et al., 2013), MNLI (Williams et al., 2018), QNLI (Rajpurkar et al., 2016), RTE (Dagan et al., 2005; Bar Haim et al., 2006; Giampiccolo et al., 2007; Bentivogli et al., 2009), we follow (Gao et al., 2021) and (Zhang et al., 2021) and use their original development sets for testing. For datasets MR (PANG, 2005), CR (Hu and Liu, 2004), MPQA (Wiebe et al., 2005), Subj (Pang and Lee, 2004), Yahoo! and AGNews(Zhang et al., 2015), we use the testing set randomly sampled from training set and leaved from training by (Gao et al., 2021)[16]. For SNLI (Bowman et al., 2015), we use their official test sets.

B.2 Results

Different Templates We compared in detail the performance of NSP-BERT under different prompt templates. This experiment wad conducted on 4 Chinese single-sentence classification datasets.

- **Template 1** uses just the original label words.
- **Template 2** adds pronouns and copulas such as "I am", "it is" or "this is", to make the template become a complete sentence.
- **Template 3** incorporates more domain information into the prompts, such as "shopping", "news", "paper" and "app". This makes the original input sentence and prompt have better connectivity.

For zero-shot learning, the prompt templates have a strong impact on the performance, and for different models, there is a big difference. Therefore, we verified the influence of templates for different models versions and scales. The results are shown in Table 9, Table 10, Table 11 and Table 12.

[15]https://www.cluebenchmarks.com/fewclue.html

[16]https://nlp.cs.princeton.edu/projects/lm-bff/datasets.tar

Category	Corpus	#Train	#Test	$\|\mathcal{Y}\|$	Task Type	Metrics	Source
			English Tasks				
Single-Sentence	SST-2	6,920	872	2	Sentiment Analysis	Acc.	Movie Reviews
	MR	8,662	2,000	2	Sentiment Analysis	Acc.	Movie Reviews
	CR	1,775	2,000	2	Sentiment Analysis	Acc.	E-commerce Reviews
	MPQA	8,606	2,000	2	Opinion Polarity	Acc.	World Press
	Subj	8,000	2,000	2	Subjectivity	Acc.	Movie Reviews
	Yahoo!	1,400,000	6,000	10	Question Classification	Acc.	Yahoo
	AGNews	8,551	7,600	4	News Topic Classification	Acc.	Web
Sentence-Pair	MNLI	392,702	9,815	3	Natural Language Inference	Acc.	Speech, Fiction and Reports
	MNLI-mm	392,702	9,832	3	Natural Language Inference	Acc.	Speech, Fiction and Reports
	SNLI	549,367	9,842	3	Natural Language Inference	Acc.	Image Captions
	QNLI	104,743	5,463	2	Natural Language Inference	Acc.	Wikipedia
	RTE	2,490	277	2	Natural Language Inference	Acc.	News and Wikipedia
			Chinese Tasks (FewCLUE)				
Single-Sentence	EPRSTMT	32	610	2	Sentiment Analysis	Acc.	E-commerce Reviews
	TNEWS	240	2,010	15	Short Text Classification	Acc.	News Title
	CSLDCP	536	1,784	67	Long Text Classification	Acc.	Academic CNKI
	IFLYTEK	928	1,749	119	Long Text Classification	Acc.	App Description
Sentence-Pair	OCNLI	32	2,520	3	Natural Language Inference	Acc.	5 genres
	BUSTM	32	1,772	2	Short Text Matching	Acc.	AI Virtual Assistant
	CSL	32	2,828	2	Keyword Recognition	Acc.	Academic CNKI
Others	ChID	42	2,002	7	Chinese Idiom Cloze Test	Acc.	Novel, Essay News
	CLUEWSC	32	976	2	Coreference Resolution	Acc.	Chinese Fiction Books

Table 8: Task descriptions and statistics. In FewCLUE we omit the unlabeled dataset because it is not used. Test of FewCLUE indicates the number of samples in the public test set. The 5 text genres of OCNLI are government documents, news, literature, TV talk shows and telephone conversations.

ORG	Models	Template 1 (Dev/Test)	Template 2 (Dev/Test)	Template 3 (Dev/Test)
UER	BERT-TINY	68.13/76.56	75.00/80.82	**81.88/80.33**
	BERT-SMALL	85.00/87.70	82.50/87.70	**87.50/86.72**
	BERT-BASE	60.00/54.59	78.75/80.98	**88.13/86.89**
	BERT-LARGE	78.13/82.79	83.75/82.62	**84.38/84.43**

Table 9: Zero-shot acc. of NSP-BERT on EPRSTMT.

ORG	Models	Template 1 (Dev/Test)	Template 2 (Dev/Test)	Template 3 (Dev/Test)
UER	BERT-TINY	38.80/36.62	39.25/36.37	**41.07/38.56**
	BERT-SMALL	38.98/38.81	39.80/40.35	**41.80/42.19**
	BERT-BASE	41.26/41.84	46.99/48.66	**50.64/51.00**
	BERT-LARGE	45.17/42.79	48.72/48.31	**54.28/53.83**

Table 10: Zero-shot acc. of NSP-BERT on TNEWS.

ORG	Models	Template 1 (Dev/Test)	Template 2 (Dev/Test)	Template 3 (Dev/Test)
UER	BERT-TINY	24.03/25.73	**27.37/29.60**	25.68/28.81
	BERT-SMALL	28.48/30.72	29.35/31.45	**29.78/31.78**
	BERT-BASE	39.80/40.53	44.87/45.80	45.26/**47.59**
	BERT-LARGE	44.73/42.83	44.00/44.34	**45.89**/46.92

Table 11: Zero-shot acc. of NSP-BERT on CSLDCP.

ORG	Models	Template 1 (Dev/Test)	Template 2 (Dev/Test)	Template 3 (Dev/Test)
UER	BERT-TINY	32.70/32.65	31.97/34.13	**33.65/34.59**
	BERT-SMALL	32.27/32.42	**35.54**/34.65	35.25/**34.76**
	BERT-BASE	36.41/36.59	42.39/40.19	**43.12/41.62**
	BERT-LARGE	37.73/36.94	44.28/**42.60**	**44.87**/42.42

Table 12: Zero-shot acc. of NSP-BERT on IFLYTEK.

Probability of NSP in sentence-pair tasks To further explain the necessity for us to propose sample-contrast mapping method, we show the NSP output probability of the sentence-pair tasks in Figure 9 and Figure 10. It's not difficult to see that the NSP probability of most samples is close to 1. So we can not judge its label for a individual sample. We need to contrast different samples, and predict the label by obtaining the distribution of the dataset.

Impact of batch size for samples-contrast In one case, we cannot get the entire test set at once, then we need to predict the samples of the test set batch by batch. We set the batch size $|B| \in \{1, 2, ..., 128, \text{ALL}\}$, to observe the results predicted by samples-contrast method (see Table 13). As the batch size increases, the performance

improves and stabilizes. Of course, when the batch size is less than the number of labels, the result is equivalent to random guessing. In another case, we cannot get the distribution of the test set, that is, we don't know the proportion of each label. Then we can use the development to calculate the NSP probability threshold of each label to predict the test set. The model can also get the desired performance.

Strategies for datasets For different datasets, according to their characteristics, the position of the prompt (prefix or suffix), and the mapping method (candidates-contrast or samples-contrast) are different. We take Chinese tasks as examples, all the strategies are shown in Table 14. In the single-sentence classification tasks (EPRSTMT, TNEWS, CSLDCP, IFLYTEK), the prompts are all prefixed, and we adopt candidates-contrast. For the word sense disambiguation tasks (CLUEWSC and DuEL2.0), since we need to utilize two-stage prompt method, we all use the suffix. In sentence-pair tasks (OCNLI, BUSTM and CSL), we choose the appropriate order through the development set to arrange the two sentences, where suffix means using the original order and prefix means using the reverse order.

Prompts for datasets Due to the number of data sets in our paper, we report in detail the prompt templates of the more important Chinese datasets in Table 16, and briefly report the prompts of English datasets in Table 15.

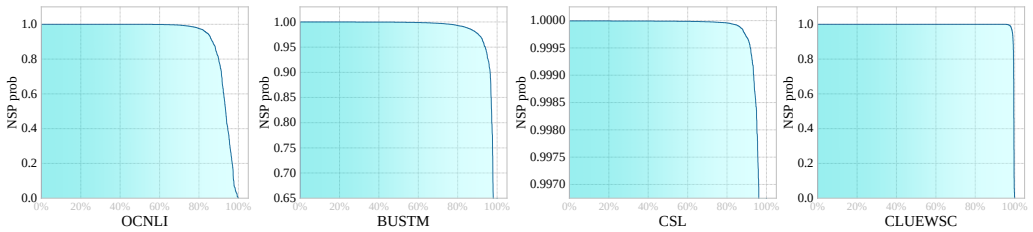

Figure 9: The NSP output probability of the 4 sentence-pair tasks OCNLI, BUSTM, CSL and CLUEWSC in Chinese benchmark FewCLUE. The x-axis represents the proportion of the samples. And the y-axis represents the NSP probability of the samples.

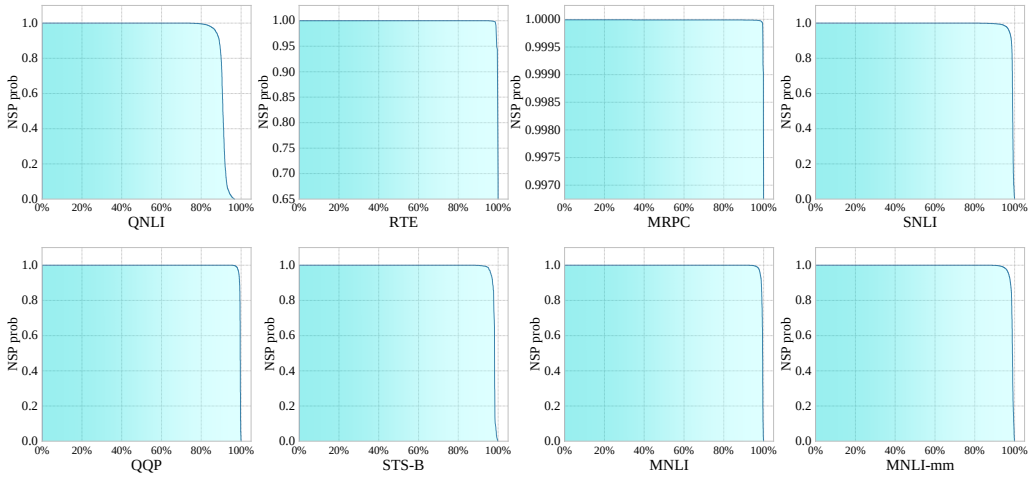

Figure 10: The NSP output probability of the 8 English sentence-pair tasks QNLI, RTE, MRPC, SNLI, QQP, STS-B, MNLI and MNLI-mm. The x-axis represents the proportion of the samples. And the y-axis represents the NSP probability of the samples.

Dataset	**Dev**	**Test**									
		$\|\mathcal{B}\|=1$	$\|\mathcal{B}\|=2$	$\|\mathcal{B}\|=4$	$\|\mathcal{B}\|=8$	$\|\mathcal{B}\|=16$	$\|\mathcal{B}\|=32$	$\|\mathcal{B}\|=64$	$\|\mathcal{B}\|=128$	$\|\mathcal{B}\|=$All	Threshold
OCNLI	37.50	33.33	33.33	35.75	36.51	36.90	37.26	**37.50**	36.83	36.90	37.38
BUSTM	62.50	50.00	56.09	67.79	59.59	59.93	61.06	61.40	61.85	**63.43**	**63.43**
CSL	64.38	50.00	58.91	62.09	62.79	62.86	62.79	63.07	63.00	63.85	**64.41**
CLUEWSC	57.23	50.00	53.69	54.30	54.51	54.71	55.53	56.56	56.56	58.61	**59.43**
MNLI-m	41.67	35.22	35.22	39.08	**40.04**	39.08	39.63	39.33	39.48	39.33	39.41
MNLI-mm	39.58	35.45	35.45	38.41	38.59	38.62	38.19	37.69	38.24	38.17	**39.17**
SNLI	43.75	34.28	34.28	**44.14**	44.21	43.54	43.20	43.17	43.13	43.35	43.42
QNLI	87.50	49.46	62.37	64.63	65.37	66.58	66.87	67.23	67.34	67.56	**67.56**
RTE	62.50	52.71	52.71	54.87	53.43	55.60	54.15	**54.15**	54.87	51.99	55.60

Table 13: The performance of the samples-contrast answer mapping method under different preconditions on sentence-pair tasks. Batch size $|\mathcal{B}| \in \{1, 2, ..., 128, \text{ALL}\}$, when the batch size is less than the number of labels, the result is a random guess, when the batch size is ALL, indicating that the entire test set is obtained at one time. `Thresholds` means that the thresholds are obtained through the development set, and then used for the prediction of the test set.

Strategies		Single-Sentence Task				Sentence-Pair Task			Others		DuEL2.0	
		EPRSTMT	TNEWS	CSLDCP	IFLYTEK	OCNLI	BUSTM	CSL	ChID	CLUEWSC	Entity Linking	Entity Typing
Prompt	Prefix	✓	✓	✓	✓	✓	✓					
	Suffix							✓	✓	✓	✓	✓
Answer	C-C	✓	✓	✓	✓				✓		✓	✓
Mapping	S-C					✓	✓	✓		✓		

Table 14: Strategies adopted on the 10 datasets in FewCLUE and DuEL2.0. The **prefix** means to put the prompt in front of the original text, and the **suffix** is the opposite. **C-C** means candidates-contrast answer mapping method, and **S-C** means samples-contrast answer mapping method.

Task	Method	Prompt Templates
SST-2		Original Labels: negative; positive
	PET	Mapping Words: terrible; great Prompt Template: **x** It was `[label]`.
	NSP-BERT	Mapping Words: terrible; great Prompt Template: A `[label]` piece of work `[SEP]` **x**
MR		Original Labels: negative; positive
	PET	Mapping Words: terrible; great Prompt Template: **x** It was `[label]`.
	NSP-BERT	Mapping Words: terrible; great Prompt Template: A `[label]` piece of work `[SEP]` **x**
CR		Original Labels: negative; positive
	PET	Mapping Words: terrible; great Prompt Template: **x** It was `[label]`.
	NSP-BERT	Mapping Words: terrible; great Prompt Template: It was `[label]`. `[SEP]` **x**
Subj		Original Labels: negative; positive
	PET	Mapping Words: subjective; objective Prompt Template: **x** This is `[label]`.
	NSP-BERT	Mapping Words: subjective; objective Prompt Template: A `[label]` comment `[SEP]` **x**
MPQA		Original Labels: negative; positive
	PET	Mapping Words: terrible; great Prompt Template: **x** It was `[label]`.
	NSP-BERT	Mapping Words: negative; positive Prompt Template: It is `[label]`. `[SEP]` **x**
Yahoo!		Original Labels: Society & Culture; Science & Mathematics; Health; Education & Reference; Computers & Internet; Sports; Business & Finance; Entertainment & Music; Family & Relationships; Politics & Government
	PET	Mapping Words: Society; Science; Health; Education; Computer; Sports; Business; Entertainment; Relationship; Politics Prompt Template: `[label]` question: **x**
	NSP-BERT	Mapping Words: Society; Science; Health; Education; Computer; Sports; Business; Entertainment; Relationship; Politics Prompt Template: `[label]` question: `[SEP]` **x**
AGNews		Original Labels: political; sports; business; technology
	PET	Mapping Words: political; sports; business; technology Prompt Template: A `[label]` news : **x**
	NSP-BERT	Mapping Words: political; sports; business; technology Prompt Template: A `[label]` news : `[SEP]` **x**

Table 15: The prompts used in English datasets. We only show the template with best performance. We select the most suitable prompt template for PET and NSP respectively. `[label]` is the token will be replaced by the mapping words. EFL(Wang et al., 2021) uses the exact same prompts as NSP-BERT.

Task	Prompt Templates	Mapping words of PET	Mapping words of NSP-BERT
EPRSTMT	Template 1: **x** `[SEP]` 很`[label]`. Template 2: **x** `[SEP]` 东西很`[label]`. Template 3: **x** `[SEP]` 这次买的东西很`[label]`	好; 差	好; 差
TNEWS	Template 1: **x** `[SEP]` `[label]`. Template 2: **x** `[SEP]` `[label]`新闻. Template 3: **x** `[SEP]` 这是一则 `[label]`新闻.	故事; 文化; 娱乐; 体育; 财经; 房产; 汽车; 教育; 科技; 军事; 旅游; 国际; 股票; 农业; 电竞	故事; 文化; 娱乐; 体育; 财经; 房产; 汽车; 教育; 科技; 军事; 旅游; 国际; 股票; 农业; 电竞
CSLDCP	Template 1: **x** `[SEP]` `[label]`. Template 2: **x** `[SEP]` `[label]` 论文. Template 3: **x** `[SEP]` 这是一篇 `[label]`论文.	材料; 作物; 口腔; 药学; 教育; 水利; 理经; 食品; 兽医; 体育; 核能; 力学; 园艺; 水产; 法学; 地质; 能源; 农林; 通信; 情报...	材料科学与工程; 作物学; 口腔医学; 药学; 教育学; 水利工程; 理论经济学; 食品科学与工程; 畜牧学/兽医学; 体育学; 核科学与技术; 力学; 园艺学...
IFLYTEK	Template 1: **x** `[SEP]` `[label]`. Template 2: **x** `[SEP]` `[label]` 类软件. Template 3: **x** `[SEP]` 这是一款 `[label]` 类软件.	打车; 地图; 免费; 租车; 同城; 快递; 婚庆; 家政; 交通; 政务; 社区; 赚钱; 魔幻; 仙侠; 卡牌; 飞行; 射击; 休闲; 动作; 体育; 棋牌; 养成; 策略; 竞技; 辅助...	打车; 地图导航; 免费WIFI; 租车; 同城服务; 快递物流; 婚庆; 家政; 公共交通; 政务; 社区服务; 薅羊毛; 魔幻; 仙侠; 卡牌; 飞行空战; 射击游戏; 休闲益智; 动作类; 体育竞技...

Table 16: The prompts used for single-sentence classification tasks in FewCLUE. `[label]` is the token will be replaced by the mapping words. The mapping words of PET need to be manually converted to equal length. Since there are two options for the prompt, **prefix** and **suffix**, we select the most suitable one through the development set. For dataset with a lot of labels, due to space considerations, we have omitted some of them.

MetaPrompting: Learning to Learn Better Prompts

Yutai Hou*, **Hongyuan Dong***, **Xinghao Wang, Bohan Li, Wanxiang Che**†
Research Center for Social Computing and Information Retrieval
Harbin Institute of Technology, China
{ythou, hydong, xhwang, bhli, car}@ir.hit.edu.cn

Abstract

Prompting method is regarded as one of the crucial progress for few-shot nature language processing. Recent research on prompting moves from discrete tokens based "hard prompts" to continuous "soft prompts", which employ learnable vectors as pseudo prompt tokens and achieve better performance. Though showing promising prospects, these soft-prompting methods are observed to rely heavily on good initialization to take effect. Unfortunately, obtaining a perfect initialization for soft prompts requires understanding of inner language models working and elaborate design, which is no easy task and has to restart from scratch for each new task. To remedy this, we propose a generalized soft prompting method called MetaPrompting, which adopts the well-recognized model-agnostic meta-learning algorithm to automatically find better prompt initialization that facilitates fast adaptation to new prompting tasks. Extensive experiments show MetaPrompting tackles soft prompt initialization problem and brings significant improvement on four different datasets (over 6 points improvement in accuracy for 1-shot setting), achieving new state-of-the-art performance.

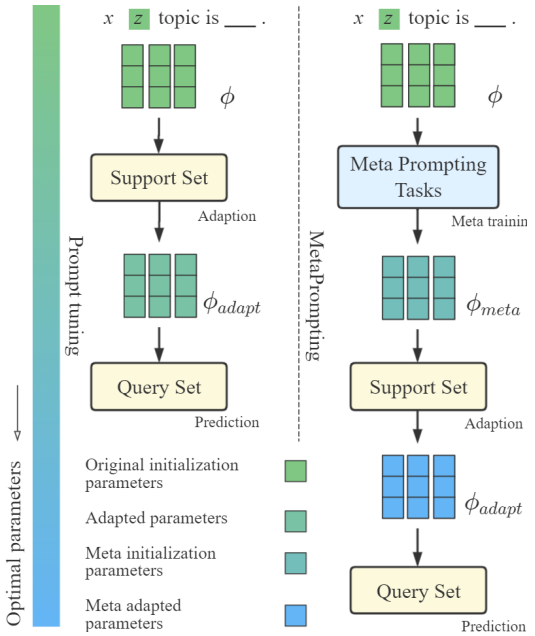

Figure 1: Comparison between conventional soft-prompting method (left) and proposed MetaPrompting (right). x denotes the query sentence, and z is learnable pseudo tokens in soft prompts. ϕ represents all trainable parameters. MetaPrompting exploits optimization-based meta-learning to find an initialization ϕ_{meta} that facilitates better and faster adaptation to new tasks.

1 Introduction

Enabling models to learn from a few labeled examples, i.e., Few-Shot Learning (FSL), is one of the key steps toward more human-like artificial intelligence. Recently, taking advantage of large-scale Pretrained Language Models (PLM) (Brown et al., 2020), prompting-based methods achieve impressive results for few-shot learning of Natural Language Processing (NLP) (Gao et al., 2021; Liu et al., 2021a; Zhao et al., 2021).

Prompting-based methods insert a piece of text, i.e. prompts, to the input examples, so that the few-shot task can be formulated as a (masked) language modeling problem. For example, say we want to classify the sentiment of the book review "I will never read it again.", we can append a prompt "It was" to the sentence, getting "I will never read it again. It was". It is natural to expect a higher probability from the PLM to generate "terrible" than "great" then. Such converting bridges the gap between pre-training and target tasks. Consequently, it has better transferability and less dependence on target task data.

The performance of prompting methods is found to be greatly affected by the design of prompts (Gao et al., 2021). That is, a good prompt makes significant difference. Early attempts take manually-designed prompts or search prompts automatically.

* Equal contribution.
†Email corresponding

Proceedings of the 29th International Conference on Computational Linguistics, pages 3251–3262
October 12–17, 2022.

Schick et al. (2020) and (Schick and Schütze, 2021) explore to automatically identify label words. In pursuit of better performance compared to hand-picked prompts, Gao et al. (2021) proposes LM-BFF to search both prompt templates and label words. AutoPrompt (Shin et al., 2020) leverages gradient-based searching to find out the best prompts. These prompts consist of discrete tokens, which limits the prompt search space. To further liberate the potential of prompts, recent works employ learnable vectors as prompt content and learn optimal prompts in continuous space, which is so-called "soft prompts" (Liu et al., 2021c; Li and Liang, 2021). Since they no longer require prompts to be composed of real words, these methods greatly expand the possibilities of prompts and thus achieve better performance (Liu et al., 2021b).

However, despite the promising prospects of soft prompts, learning a good prompt is still far from trivial. Because soft-prompts search for optimal solutions in an infinite continuous space, the choice of the starting point for the search (i.e., prompt initialization) becomes crucial. Soft-prompt is observed to be more sensitive to different initialization than discrete prompts in low data setting (Li and Liang, 2021; Liu et al., 2021b). Unfortunately, creating a perfect prompt initialization requires both understanding of LMs' inner workings and trial-and-error. Lester et al. (2021) initialize soft prompt with the token embeddings of hand-crafted prompt directly. Zhong et al. (2021b) search discrete tokens as better initialization, which shows better performance. What's worse is that these initializations are task-bounded. Every time we confront a new task, the costly process of initialization design has to start from scratch.

In this paper, to tackle the above issues, we let loose the prompt design of a specific task, but instead focus on obtaining a task general prompt initialization that facilitates faster and better adaptation to new prompting tasks. Recently proposed optimization-based meta-learning algorithms, such as MAML (Finn et al., 2017) and Reptile (Nichol et al., 2018), achieve better adaption by learning a parameter initialization. Following their essence, we tackle soft prompt initialization problem by proposing MetaPrompting, which is a generalized soft prompting method powered by meta-learning algorithms. MetaPrompting learns general meta-knowledge from source domain tasks to form a better soft prompt initialization, and thus adapts faster and better across various target domain tasks (See Figure 1). Extensive experiments show that MetaPrompting achieves promising performance with desired robustness.

We summarize the main contribution of this paper as follows:

(1) We propose a novel prompting method MetaPrompting, which employs optimization-based meta-learning algorithm to find adaptive initialization for soft-prompt methods. To the best of our knowledge, this is the first study of applying meta-learning to prompting problem setting.

(2) We conduct extensive experiments on four different datasets with various few-shot settings, and results show the superiority of MetaPrompting over normally fine-tuned soft-prompt methods and SOTA meta-learning baselines.

(3) Further analysis experiments indicate that MetaPrompting significantly alleviates soft prompt initialization problem, and learns general meta-knowledge to counter the instability of prompt variance. We also study MetaPrompting's compatibility with different meta-learning methods and give empirical analysis of their performance difference.

All code and data will be publicly available at https://github.com/Dousia/MetaPrompting.

2 Preliminaries and Related Works

In this section, we review related work and provide preliminaries about Language Model Prompting and Meta-learning.

2.1 Prompting Language Models

Prompting methods are proposed to better apply pre-trained language models to downstream tasks by aligning them with pre-training tasks. For Masked Language Models (MLMs), the first step is to convert a sample text x to x_{prompt} by inserting prompt words which contain `[MASK]` tokens. Taking the news headline classification task as an example, the prompted text is given as:

$$x_{prompt} = \texttt{[CLS]}\ x\ \text{The topic is}\ \texttt{[MASK]}\ .\ \texttt{[SEP]}, \quad (1)$$

where "The topic is [MASK]" are prompt tokens. Then, we ask pre-trained MLM to complete the prompted text x_{prompt}, and the word to be filled at `[MASK]` position is regarded as an answer. An answer-label map is then used to convert the word probability distribution at `[MASK]` to classification results. For example, answers 'arts' and 'culture' can be mapped to label 'ARTS & CULTURE',

while 'environment' can be mapped to label 'ENVIRONMENT'. The average probability of each label's corresponding answers is computed as the label's final probability.

Early prompting methods, such as GPT-3 (Brown et al., 2020) and PET/iPET (Schick and Schütze, 2021), use hand-crafted prompt templates. Although promising results are achieved, the performance of these methods heavily relies on the selection of pre-defined prompt templates. Moreover, designing prompts is extremely time-consuming, and hand-crafted prompts may be sub-optimal.

A number of recent works propose to automate the search of discrete prompt templates (Shin et al., 2020; Gao et al., 2021; Davison et al., 2019; Jiang et al., 2020; Haviv et al., 2021), while others treat prompt tokens as continuous trainable parameters (Li and Liang, 2021; Liu et al., 2021c; Qin and Eisner, 2021). In this work, we follow P-tuning (Liu et al., 2021c) to combine soft prompt and anchor tokens as templates. Instead of directly applying the model in few-shot tasks, however, we adopt meta-learning methods to find a better initialization point for both soft prompt embeddings and MLM parameters, because they are very sensitive to initialization in few-shot settings (Li and Liang, 2021; Liu et al., 2021b). Note that a recent work (Zhong et al., 2021a) also learns prompt model on a number of source domain tasks, but their method consumes heavy human labor to design hard prompts for each task, and directly fine-tunes the model without involving meta algorithms.

2.2 Meta Learning

Meta-learning algorithms can be classified into metric-based methods, model-based methods and optimization-based methods. Metric-based methods such as Siamese Network (Koch et al., 2015), Matching Network (Vinyals et al., 2016) and Prototypical Network (Snell et al., 2017), are proposed to learn a metric space that gathers similar samples and separates distinct ones. Model-based meta-learning algorithms use additional meta learners to assist model prediction (Graves et al., 2014; Mishra et al., 2018; Qiao et al., 2018).

Different from above algorithms, optimization-based meta-learning methods learn to improve model's optimization procedure. Optimization-based approach is more suitable for prompting models as it neither requires a specific task form (i.e., metric learning form) nor additional architecture (e.g. memory-augmented components in model-based algorithms). Andrychowicz et al. (2016) and Ravi and Larochelle (2017) train recurrent neural networks to transform vanilla gradient descent direction for better optimization results. MAML (Finn et al., 2017) optimizes model parameters to find a better initialization point, so that the model can adapt faster and better to unseen tasks. In addition to MAML, more elaborate methods also learn inner loop gradient descent direction (Li et al., 2017) and inner step sizes (Antoniou et al., 2019). Utilizing first-order derivatives, FOMAML (Finn et al., 2017) and Reptile (Nichol et al., 2018) are proposed to reduce the memory consumption of high-order derivative calculation.

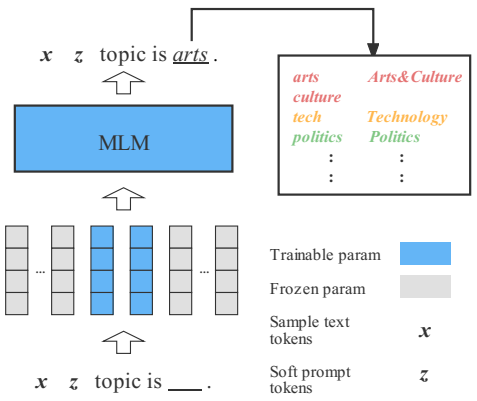

Figure 2: An illustration of soft-prompting method. x refers to the sample text, and z represents soft prompt tokens. All trainable parameters are colored in blue, while fixed ones are colored in grey.

3 Method

Since prompt-based methods, especially those adopting soft prompts, are very sensitive to parameter initialization (Li and Liang, 2021; Liu et al., 2021b), we introduce optimization-based meta-learning methods into prompting methods to find better initialization points for prompt-based models and further explore their capabilities in few-shot scenarios. In this section, we first introduce the prompt-based model tuning process used in our method (§3.1), and then describe how to construct Meta Prompting tasks (§3.2). Finally, we elaborate and formulate the Meta Prompting objective and parameter updating strategies (§3.3 and §3.4).

3.1 Prompt-based Model Tuning

In this work, we use soft prompts with anchor tokens. As illustrated in Figure 2, prompt tokens consist of soft tokens which are represented as

trainable parameters (blue) as well as anchor tokens which are fixed as the embeddings of specific words (grey). Hard-soft combined prompt templates render the model more flexible, while preserving enough semantic information to trigger the MLM to produce correct predictions. Similar to P-tuning (Liu et al., 2021c), we implement transformation layers on soft prompt embeddings, allowing them to escape from local minima smoothly.

In this way, we define MLM parameters as θ and soft prompt token embeddings as ϕ. Given a few-shot task τ where $\mathcal{D}_\tau = \{(\mathbf{x}_i, \mathbf{y}_i)\}_{i \in \tau}$ represents training samples, the prompt tuning objective can be formulated as follows:

$$\begin{aligned} \theta^*, \phi^* &= \arg\min_{\theta,\phi} \mathcal{L}_{\mathcal{D}_\tau}(f_{\phi,\theta}) \\ &= \arg\max_{\theta,\phi} \sum_{(\mathbf{x}_i,\mathbf{y}_i)\in\mathcal{D}_\tau} \log P(\mathbf{y}_i|\mathbf{x}_i;\phi,\theta), \end{aligned} \tag{2}$$

where $\mathcal{L}$ is the loss function, and $f_{\phi,\theta}$ is prompt-based model parameterized by MLM parameters θ and soft prompt embeddings ϕ.

$\mathcal{D}_\tau$ contains few labeled data because of the high annotation cost in real-world scenarios. As a result, the initialization of parameters θ and ϕ are more than crucial to the model's performance.

3.2 Constructing Meta Prompting Tasks

To get a better initialization point for parameters θ and ϕ, we propose to sample Meta Prompting tasks from accessible source data and conduct meta-training on these sampled tasks. This meta training process aims to simulate the model's adaptation to new few-shot tasks.

We sample each Meta Prompting task τ_i as:

$$\tau_i = (\mathcal{D}_{\tau_i}^{support}, \mathcal{D}_{\tau_i}^{query}), \tag{3}$$

where $\mathcal{D}_{\tau_i}^{support}$ indicates the support set and $\mathcal{D}_{\tau_i}^{query}$ indicates the query set in traditional few-shot learning settings. Note that meta training tasks and meta testing tasks should be sampled from different domains, to prevent the model from simply memorizing training samples.

3.3 Applying Meta-learning to Prompting Models

After constructing Meta Prompting tasks, we train our prompting model on these tasks to find a better initialization point. Figure 3 illustrates the meta training and meta testing procedures of MetaPrompting. Given a Meta Prompting task τ_i, we clone the model's parameters and simulate the

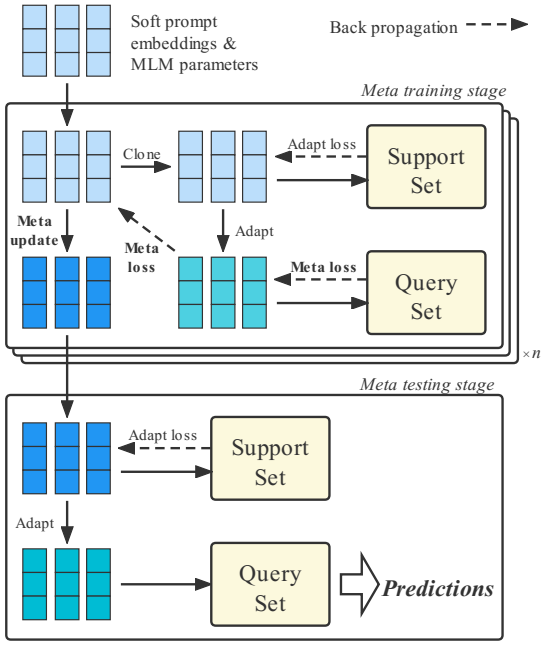

Figure 3: Meta training and testing procedures of MetaPrompting.

adaption process of few-shot tasks by updating cloned model parameters θ_i^0 and ϕ_i^0 on $\mathcal{D}_{\tau_i}^{support}$. The adaption objective is given in Equation (2), and this process can be formulated as:

$$\begin{aligned} \theta_i^k &= \theta_i^{k-1} - \alpha \nabla_{\theta_i^{k-1}} \mathcal{L}_{\mathcal{D}_{\tau_i}^{support}}(f_{\phi_i^{k-1},\theta_i^{k-1}}), \\ \phi_i^k &= \phi_i^{k-1} - \alpha \nabla_{\phi_i^{k-1}} \mathcal{L}_{\mathcal{D}_{\tau_i}^{support}}(f_{\phi_i^{k-1},\theta_i^{k-1}}), \end{aligned} \tag{4}$$

where α indicates learning rate and $k = 1, 2, 3, \ldots$ indicates the inner step. The goal of learning with Meta Prompting tasks is to minimize the loss of the adapted prompting model, which is parameterized as f_{ϕ_i,θ_i}, on $\mathcal{D}_{\tau_i}^{query}$. This objective can be described as follows:

$$\theta^{obj}, \phi^{obj} = \arg\min_{\theta_i,\phi_i} \mathcal{L}_{\mathcal{D}_{\tau_i}^{query}}(f_{\phi_i,\theta_i}). \tag{5}$$

Optimizing towards this objective is to mimic real few-shot text classification scenarios, and enable prompting model to find a better initialization point for fast adaptation to new tasks. Let β be the learning rate when updating model parameters on $\mathcal{D}_{\tau_i}^{query}$, and $\mathbf{H}$ be Hessian matrix. We formulate the second-order gradient of prompt parameter ϕ computed on $\mathcal{D}_{\tau_i}^{query}$ in the following form:

$$\begin{aligned} \phi &\leftarrow \phi - \beta \cdot g_\phi^{second} \\ &= \phi - \beta \nabla_\phi \mathcal{L}_{\mathcal{D}_{\tau_i}^{query}}(f_{\phi_i,\theta_i}) \\ &= \phi - \beta \nabla_{\phi_i} \mathcal{L}_{\mathcal{D}_{\tau_i}^{query}}(f_{\phi_i,\theta_i}) \cdot \nabla_\phi(\phi_i) \\ &= \phi - \beta \nabla_{\phi_i} \mathcal{L}_{\mathcal{D}_{\tau_i}^{query}}(f_{\phi_i,\theta_i}) \cdot \\ &\qquad (\mathbf{I} - \alpha \mathbf{H}_\phi(\mathcal{L}_{\mathcal{D}_{\tau_i}^{support}}(f_{\phi,\theta}))), \end{aligned} \tag{6}$$

where we assume ϕ_i is ϕ adapted for one inner step on $\mathcal{D}_{\tau_i}^{support}$. In practice, inner steps can be increased for better performance. Pre-trained MLM parameters θ is updated in the same way as prompt parameters ϕ in Equation (6).

3.4 Stable and Memory-efficient Meta Prompt Learning

Although broadly used in meta-learning tasks, MAML suffers from training instability and exploding memory consumption when model size and inner step grow. To address the first problem, we follow Antoniou et al. (2019) to introduce Multi-Step Loss Back-propagation (MSLB) into prompting model tuning process. In this way, prompting model parameters receive optimization information from each inner step during adaptation, alleviating the vanishing/exploding gradient problem in the stacked deep neural architecture constructed in adaptation process.

As for the exploding memory consumption issue, we also explore to combine memory-efficient alternatives such as FOMAML (Finn et al., 2017) and Reptile (Nichol et al., 2018) with prompting model. FOMAML removes the high-order derivatives term in Equation (6), providing a cheap approximation for MAML. Reptile updates model parameters towards the optimal point of each task, which is obtained by adapting the model on the support set samples. Equipped with these algorithms, MetaPrompting can learn meta knowledge with limited memory consumption.

4 Experiment

We conduct experiments by evaluating the proposed methods on four widely-used benchmark datasets with various low resource settings.

4.1 Dataset

Following Bao et al. (2019); Xu and Xiang (2021), we use the following four text classification datasets for experiments, which provide well-founded benchmarks for the meta-train & meta-test setting and vary in domain and text length.

HuffPost headlines contains around $36,900$ news headlines from 2012 to 2018 obtained from HuffPost (Misra, 2018). These headlines cover 41 news categories and the average text length is 11.

Amazon product data contains around $24,000$ product reviews from 1996 to 2014 from Amazon (He and McAuley, 2016). These reviews contain 24 categories corresponding to their respective product categories with varying text lengths. The average text length is 140.

20 Newsgroups contains $18,820$ newsgroup documents of 20 different topics (Lang, 1995). We used 20news-18828 version following Bao et al. (2019). The average text length is 340.

Reuters contains 620 document-level news articles of 31 different topics from 1987 (Lewis, 1997). The average text length is 168.

We adopt the same pre-processing and data-splitting strategy with Bao et al. (2019) to process the above datasets.

4.2 Implementation

We use the pre-trained BERT (bert-base-uncased) with HuggingFaces codebase (Wolf et al., 2019) as the pre-trained language model.

For soft prompting model, we follow Liu et al. (2021c) to use a two-layer biLSTM and a two-layer MLP to transform soft-prompt embeddings. We divide the learnable parameters of prompting model into two parts: pre-trained model and prompt embeddings. AdamW (Loshchilov and Hutter, 2018) is used to optimize two types of parameters, with initial learning rates of $1e{-}5$ and $5e{-}5$, respectively. For pre-trained model parameters, we set weight decay to 0.1. We also adopt linear warmup and linear decay strategy for learning rates. Batch size is set as 16 for all stages, and the model adapts for 15 epochs on test episodes. We run 3 independent runs with random seeds for each setting.

Before meta-training stage, we generate $10,000$ training episodes, $2,500$ validation episodes and $1,000$ testing episodes comprehensively and randomly. During the training stage, we train the model with 100 sampled training episodes per epoch. When there is no validation accuracy increase for 10 epochs, we apply early stopping. For meta-testing, we test the model on all $1,000$ test episodes and report the average accuracy.

4.3 Baselines

We compare with the following baselines:

1-NN is a 1-nearest-neighbor classifier based on Euclidean distance.

FT (Chen et al., 2019) pre-trains a classifier on source domain data, and then fine-tunes (FT) it on each support set before evaluation.

For other datasets used in Bao et al. (2019), RCV1 (Lewis et al., 2004) is not included due to overly long text lengths, while FewRel (Han et al., 2018) is excluded because it does not provide each label's semantic meanings.

Method	20 News		Amazon		HuffPost		Reuters		Average	
	1 shot	5 shot	1 shot	5 shot	1 shot	5 shot	1 shot	5 shot	1 shot	5 shot
1-NN	38.8	51.9	51.4	67.1	31.5	42.3	57.8	82.9	44.88	61.05
FT	33.0	47.1	45.7	63.9	32.4	44.1	70.9	91.0	45.50	61.53
PROTO	37.8	46.5	41.9	59.2	34.8	50.2	61.0	72.1	43.88	57.00
MAML	37.2	48.6	43.6	62.4	38.9	53.7	61.5	72.0	45.30	59.18
RR	44.8	64.3	60.2	79.7	37.6	59.5	69.1	93.0	52.93	74.13
DS (2019)	52.1	68.3	62.6	81.1	43.0	63.5	81.8	96.0	59.88	77.23
DE-MLMAN (2021)	–	–	–	–	49.7	60.9	–	–	–	–
DE-MAML (2021)	–	–	–	–	51.8	67.3	–	–	–	–
DE-PROTO (2021)	–	–	–	–	52.3	69.6	–	–	–	–
KGML-PROTO (2021)	–	–	58.6	74.5	42.3	58.7	–	–	–	–
KGML-MAML (2021)	–	–	51.4	58.8	44.2	54.1	–	–	–	–
P-TUNING (2021c)	56.21	77.70	62.18	79.13	54.48	65.75	90.01	96.66	65.72	79.81
FROG-GNN (2021)	–	–	71.5	83.6	54.1	69.6	–	–	–	–
LASAML-PN (2021)	–	–	–	–	62.1	70.1	–	–	–	–
CONTRASTNET (2022)	**71.74**	81.57	76.13	85.17	53.06	65.32	86.42	95.33	71.84	81.85
OURS (PRETRAIN INIT)	68.42	79.11	75.12	83.27	70.82	75.47	95.07	**97.29**	77.36	83.79
OURS (META INIT)	68.83	**82.95**	**77.65**	**85.54**	**71.93**	**76.32**	**95.20**	97.17	**78.40**	**85.50**

Table 1: Results of 1-shot and 5-shot classification on four datasets in terms of accuracy. The rows below the mid-line are results of MetaPrompting. '-' means that the result of this dataset is not given in the original paper. We do not show standard deviation of our experiment results here due to space limits. Full results can be found in Appendix A.

RR (Bertinetto et al., 2019) adopts ridge regression (RR) for classification.

MAML (Finn et al., 2017) meta-learns a classifier with MAML algorithm, so that the model can adapt faster and better to target domain tasks.

Prototypical network (Snell et al., 2017) is a metric-based method which meta-learns a metric space by minimizing the Euclidean distance between the centroid of each category and the corresponding samples.

DS (Bao et al., 2019) is trained within a meta-learning framework to map the distribution signatures (DS), i.e., characteristics of the underlying word distributions, into attention scores to extract more transferable features.

DE (Ohashi et al., 2021) generates distinct label representations that embed information specific to each label to aid classification tasks. During experiments, it is combined with MAML (DE-MAML) and prototypical network (DE-PROTO), as well as MLMAN (Ye and Ling, 2019) (DE-MLMAN).

KGML (Yao et al., 2021) extracts additional representation for each sentence from external knowledge base, to bridge the gap between meta-training and meta-testing tasks. During experiments, it works with MAML (KGML-MAML) and prototypical network (KGML-Proto).

P-tuning (Liu et al., 2021c) is a prompt-based method that uses masked language model to convert target tasks into cloze problems. It employs soft-prompting techniques to optimize prompts in continuous space.

Frog-GNN (Xu and Xiang, 2021) is a graph neural network based method, which extracts better query representations with multi-perspective aggregation of graph node neighbors.

LaSAML-PN (Luo et al., 2021) is a meta-learning framework that mines semantic information in labels and attaches it to the sentence as the input of the encoder to obtain discriminative sentence embeddings.

ContrastNet (Chen et al., 2022) is the SOTA method. It introduces instance-level and task-level regularization loss into vanilla contrastive learning model based on BERT representations for better generalization performance. The regularization loss is computed with samples augmented by an additional BERT model.

4.4 Main Results

We evaluate the proposed methods in both 5-way 1-shot and 5-way 5-shot settings and report performance on four different datasets with different text styles. As shown in Table 1, our model outperforms previous SOTA method ContrastNet without using additional PLM. Averagely, our model improves 1-shot accuracy by 6.56 (9.13% ↑) and 5-shot accuracy by 3.65 (4.46% ↑) across four datasets. MetaPrompting gains less improvement on 20Newsgroup and Amazon, because their la-

The above 6 baselines uses fastText embeddings (Joulin et al., 2016) and each word's inverse document frequency to produce sentence embeddings.

bels are hard to interpret as natural words, which poses difficulties for prompting models (Shin et al., 2020; Cui et al., 2022). Various methods are proposed to address this problem (Shin et al., 2020; Gao et al., 2021; Hambardzumyan et al., 2021; Cui et al., 2022; Jiang et al., 2021). In this work, however, we do not spend much effort on elaborate answer design but instead focus on soft prompt initialization problem. Although equipped with simply designed answer sets, MetaPrompting still achieves new state-of-the-art performance across the four datasets.

Meanwhile, we have following observations based on Table 1:

(1) Compared with other soft-prompting methods, i.e., P-tuning, our method obtains superior results. Although meta learning is conducted on completely different source domain Meta Prompting tasks, our method still learns a better prompt model initialization point, which allows faster and better adaption to new prompting tasks.

(2) When compared to traditional supervised learning methods, such as FT, all prompt-based methods achieve significant improvements, which demonstrates the effectiveness of prompting mechanism in narrowing the gap between pretraining and downstream tasks.

(3) Metric learning-based baselines, such as ContrastNet and LASAML-PN, perform as the strongest baselines on Amazon and HuffPost datasets, respectively. We find that directly using prompt-based method may not necessarily perform better, because of the absence of domain-related initialization. The proposed MetaPrompting alleviates the above issue and achieves new state-of-the-art performance. Among strong metric-learning baselines, Frog-GNN conducts transductive learning with additional label propagation information, and ContrastNet uses an additional BERT model to regularize the main model with augmented data. Our model achieves better performance without implementing any of above tricks.

(4) Compared with other optimization-based meta-learning methods such as MAML, DE-MAML and KGML-MAML, MetaPrompting consistently performs better, demonstrating good compatibility between prompting methods and meta-learning. Note that KGML-MAML adopts an additional knowledge base, while our model does not but still achieves better performance.

(5) To further demonstrate meta learning method's effectiveness in prompt learning, we conduct ablations study by removing the Meta Prompting Objective and learn an initialization by pretraining soft prompt model on Meta Prompting Tasks described in Section 3.2. The results are shown as OURS (PRETRAIN INIT). Performance drops are witnessed across all four datasets and few-shot settings, validating the necessity of meta objectives in finding a better initialization.

Method	HuffPost	Amazon	20 News	Reuters
BASELINE	65.75	79.13	77.70	96.66
OURS	73.06	83.64	81.79	97.26

Table 2: PLM frozen, MetaPrompting still achieves better performance over randomly initialized baseline on test domains. Results are given in 5 shot setting.

Source domain	Target domain	Acc
–	HuffPost	65.75
Metatuning	HuffPost	67.46
Reuters	HuffPost	69.18
20 Newsgroup	HuffPost	71.04
Amazon	HuffPost	71.47
HuffPost (Diff. label set)	HuffPost	76.32

Table 3: Given irrelevant source domain data, MetaPrompting still learns meta knowledge to improve the performance on target domains.

4.5 Analysis

In this part, we analyze the proposed method from different aspects.

MetaPrompting tackles soft prompt initialization problem. Main results displayed in Section 4.4 are obtained with LM parameters tuned for fair comparison with previous SOTA baselines. To further validate the importance of learning a good prompt initialization, we freeze PLM's parameters while leaving soft prompt parameters unfrozen to only learn a better prompt initialization on source domains. We test our meta-learning-based initialization strategy against random initialization, and the results are shown in Table 2. The randomly initialized soft prompt baseline performs poorly and unstably, while our method consistently yields better results with lower variance across four datasets, which verifies our hypothesis and the validity of the MetaPrompting.

MetaPrompting learns general meta-knowledge from various source domains. Although MetaPrompting shows promising results when

meta trained on source domain tasks from the same dataset, it is impractical to always build corresponding meta-training tasks for each few-shot scenario. To this end, we conduct meta-training on Out-Of-Domain (OOD) tasks to better analyze MetaPrompting's ability in transferring meta-knowledge from various source domains.

Table 3 shows the results of 5-shots setting. Even given irrelevant meta-training data and prompt templates from other datasets, MetaPrompting still learns meta knowledge to tackle target domain tasks and outperforms the baseline robustly. Among OOD datasets, Metatuning (Zhong et al., 2021a) contains a series of text classification tasks, and each task is accompanied by several hand-crafted questions which require yes/no answers. The task formulation of Metatuning is distinct from HuffPost. However, MetaPrompting still makes it to transfer meta-knowledge from Metatuning to HuffPost's target domains, improving model performance by approximately 2 points. Although MetaPrompting's performance varies among source domain tasks according to their data quality for generalization purposes, the proposed model outperforms the baseline across all source domain tasks, verifying MetaPrompting's effectiveness in transferring meta-knowledge.

Anti-disturbance analysis We expect the meta-learned initialization alleviates prompting models' susceptibility to varying prompt forms. To verify this, we test the prompting model with multiple different prompt forms and report the standard deviation. Specifically, we add two more discrete prompt templates, and randomly replace the template tokens with pseudo tokens to test MetaPrompting's robustness across different templates.

Table 4 shows the results. While changing the prompting form indeed impacts the performance for both our method and normal soft prompting methods, the proposed meta-learning method significantly reduces performance fluctuation, showing impressive anti-disturbance ability. Therefore, the proposed method is promising in real-world applications, because prompt designing requires heavy workload and domain-specific knowledge. Applying MetaPrompting significantly reduces the cost of prompt engineering.

Method	HuffPost		Amazon	
	1 shot	5 shot	1 shot	5 shot
P-TUNING	±3.46	±1.90	±5.30	±1.85
OURS	±0.23	±0.09	±0.17	±0.45

Table 4: Analysis for anti-disturbance against changing of prompting form.

Setting	MAML++	MAML	FOMAML	Reptile
1 SHOT	71.93	71.43	70.56	69.76
5 SHOT	76.32	76.04	76.08	74.09

Table 5: MetaPrompting's performance with different meta learning methods.

Applying different meta-learning methods to prompting models. In this part, we conduct empirical analysis on different optimization-based meta-learning methods applied in prompting models. Results are shown in Table 5. Stabilizing MAML training procedure, MAML++ performs the best among all methods, while Reptile fails to achieve comparable performance with others. We attribute Reptile's low performance to PLM's sensitivity to parameter tuning process, which can be distorted by Reptile's parameter updating strategy. MAML and FOMAML show similar results, because MetaPrompting's slow tuning process narrows the gap between their calculated gradient during meta-training.

Analysis for learning procedure of prompting methods. We analyze the decreasing trend of adaptation loss to better understand the learning procedure of soft-prompt model. Specifically, we visualize model adaptation loss curve during meta-testing on 5 shot Amazon dataset.

As shown in Figure 4, task-related initialization (Ours (Pretrain Init)) helps the model converge faster and end up at a lower position than randomly initialized baseline. The proposed meta-learning-based method (Ours (Meta Init)) further improves the learning process in new tasks, indicating that the meta-learned initialization point contains more generalizable meta knowledge to aid new tasks.

We add "*The topic/product category: [MASK]. Input: $\underline{x}$*" and "*$\underline{x}$. What is the topic/product category ? [MASK].*", where topic and product category are used for HuffPost and Amazon dataset respectively.

We only include the MSLB trick of MAML++ (Antoniou et al., 2019) due to the incompatibility (BN layer tricks) or trivial performance improvement (Per-step adaption loss, cosine annealing learning rates).

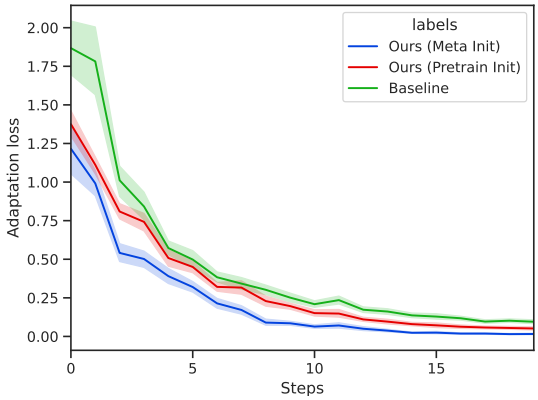

Figure 4: Analysis on prompt learning process.

5 Conclusion

In this paper, we introduce a generalized optimization-based meta-learning approach MetaPrompting for few-shot NLP problems. Utilizing sampled meta tasks and meta-learning-based optimization, MetaPrompting learns to find an initialization that alleviates soft prompt initialization problem, and allows better and faster adaption to new tasks. Extensive experiments on four few-shot learning benchmarks show that MetaPrompting significantly outperforms vanilla soft-prompting models and strong meta-learning baselines, achieving new state-of-the-art performance.

6 Acknowledgement

This work was supported by the National Key R&D Program of China via grant 2020AAA0106501 and the National Natural Science Foundation of China (NSFC) via grant 61976072 and 62176078.

References

Marcin Andrychowicz, Misha Denil, Sergio Gomez Colmenarejo, Matthew W. Hoffman, David Pfau, Tom Schaul, and Nando de Freitas. 2016. Learning to learn by gradient descent by gradient descent. In *Proc. of NIPS*.

Antreas Antoniou, Harri Edwards, and Amos Storkey. 2019. How to train your maml. In *Proc. of ICLR*.

Yujia Bao, Menghua Wu, Shiyu Chang, and Regina Barzilay. 2019. Few-shot text classification with distributional signatures. In *Proc. of ICLR*.

L Bertinetto, J Henriques, PHS Torr, and A Vedaldi. 2019. Meta-learning with differentiable closed-form solvers. In *Proc. of ICLR*.

Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. 2020. Language models are few-shot learners. In *Proc. of NIPS*, volume 33, pages 1877–1901.

Junfan Chen, Richong Zhang, Yongyi Mao, and Jie Xue. 2022. Contrastnet: A contrastive learning framework for few-shot text classification. In *Proc. of AAAI*.

Wei-Yu Chen, Yen-Cheng Liu, Zsolt Kira, Yu-Chiang Frank Wang, and Jia-Bin Huang. 2019. A closer look at few-shot classification. In *Proc. of ICLR*.

Ganqu Cui, Shengding Hu, Ning Ding, Longtao Huang, and Zhiyuan Liu. 2022. Prototypical verbalizer for prompt-based few-shot tuning. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 7014–7024.

Joe Davison, Joshua Feldman, and Alexander M Rush. 2019. Commonsense knowledge mining from pretrained models. In *Proc. of EMNLP-IJCNLP*, pages 1173–1178.

Chelsea Finn, Pieter Abbeel, and Sergey Levine. 2017. Model-agnostic meta-learning for fast adaptation of deep networks. In *Proc. of ICML*, pages 1126–1135.

Tianyu Gao, Adam Fisch, and Danqi Chen. 2021. Making pre-trained language models better few-shot learners. In *Proc. of ACL-IJCNLP*, pages 3816–3830.

Alex Graves, Greg Wayne, and Ivo Danihelka. 2014. Neural turing machines. *arXiv preprint arXiv:1410.5401*.

Karen Hambardzumyan, Hrant Khachatrian, and Jonathan May. 2021. Warp: Word-level adversarial reprogramming. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 4921–4933.

Xu Han, Hao Zhu, Pengfei Yu, Ziyun Wang, Yuan Yao, Zhiyuan Liu, and Maosong Sun. 2018. Fewrel: A large-scale supervised few-shot relation classification dataset with state-of-the-art evaluation. In *Proc. of EMNLP*, pages 4803–4809.

Adi Haviv, Jonathan Berant, and Amir Globerson. 2021. BERTese: Learning to speak to BERT. In *Proc. of EACL*, pages 3618–3623.

Ruining He and Julian McAuley. 2016. Ups and downs: Modeling the visual evolution of fashion trends with one-class collaborative filtering. In *proc. of WWW*, pages 507–517.

Zhengbao Jiang, Jun Araki, Haibo Ding, and Graham Neubig. 2021. How can we know when language models know? on the calibration of language models for question answering. *Transactions of the Association for Computational Linguistics*, 9:962–977.

Zhengbao Jiang, Frank F Xu, Jun Araki, and Graham Neubig. 2020. How can we know what language models know? *Transactions of the Association for Computational Linguistics*, 8:423–438.

Armand Joulin, Edouard Grave, Piotr Bojanowski, Matthijs Douze, Hérve Jégou, and Tomas Mikolov. 2016. Fasttext. zip: Compressing text classification models. *arXiv preprint arXiv:1612.03651*.

Gregory Koch, Richard Zemel, Ruslan Salakhutdinov, et al. 2015. Siamese neural networks for one-shot image recognition. In *ICML deep learning workshop*, volume 2.

Ken Lang. 1995. Newsweeder: Learning to filter netnews. In *Proc. of ICML*, pages 331–339.

Brian Lester, Rami Al-Rfou, and Noah Constant. 2021. The power of scale for parameter-efficient prompt tuning. In *Proc. of EMNLP*, pages 3045–3059.

D Lewis. 1997. Reuters-21578 text categorization test collection. *Distribution 1.0, AT&T Labs-Research*.

David D Lewis, Yiming Yang, Tony Russell-Rose, and Fan Li. 2004. Rcv1: A new benchmark collection for text categorization research. *Journal of machine learning research*, 5:361–397.

Xiang Lisa Li and Percy Liang. 2021. Prefix-tuning: Optimizing continuous prompts for generation. In *Proc. of ACL-IJCNLP*, pages 4582–4597.

Zhenguo Li, Fengwei Zhou, Fei Chen, and Hang Li. 2017. Meta-sgd: Learning to learn quickly for few-shot learning. *arXiv preprint arXiv:1707.09835*.

Jiachang Liu, Dinghan Shen, Yizhe Zhang, Bill Dolan, Lawrence Carin, and Weizhu Chen. 2021a. What makes good in-context examples for gpt-3? *arXiv preprint arXiv:2101.06804*.

Pengfei Liu, Weizhe Yuan, Jinlan Fu, Zhengbao Jiang, Hiroaki Hayashi, and Graham Neubig. 2021b. Pre-train, prompt, and predict: A systematic survey of prompting methods in natural language processing. *arXiv preprint arXiv:2107.13586*.

Xiao Liu, Yanan Zheng, Zhengxiao Du, Ming Ding, Yujie Qian, Zhilin Yang, and Jie Tang. 2021c. Gpt understands, too. *arXiv preprint arXiv:2103.10385*.

Ilya Loshchilov and Frank Hutter. 2018. Decoupled weight decay regularization. In *Proc. of ICLR*.

Qiaoyang Luo, Lingqiao Liu, Yuhao Lin, and Wei Zhang. 2021. Don't miss the labels: Label-semantic augmented meta-learner for few-shot text classification. In *Findings of ACL-IJCNLP*, pages 2773–2782.

Nikhil Mishra, Mostafa Rohaninejad, Xi Chen, and Pieter Abbeel. 2018. A simple neural attentive meta-learner. In *Proc. of ICLR*.

Rishabh Misra. 2018. News category dataset, 06.

Alex Nichol, Joshua Achiam, and John Schulman. 2018. On first-order meta-learning algorithms. *arXiv preprint arXiv:1803.02999*.

Sora Ohashi, Junya Takayama, Tomoyuki Kajiwara, and Yuki Arase. 2021. Distinct label representations for few-shot text classification. In *Proc. of ACL-IJCNLP*, pages 831–836.

Siyuan Qiao, Chenxi Liu, Wei Shen, and Alan L Yuille. 2018. Few-shot image recognition by predicting parameters from activations. In *Proc. of CVPR*, pages 7229–7238.

Guanghui Qin and Jason Eisner. 2021. Learning how to ask: Querying LMs with mixtures of soft prompts. In *Proc. of NAACL-HLT*, pages 5203–5212.

Sachin Ravi and H. Larochelle. 2017. Optimization as a model for few-shot learning. In *Proc. of ICLR*.

Timo Schick, Helmut Schmid, and Hinrich Schütze. 2020. Automatically identifying words that can serve as labels for few-shot text classification. In *Proc. of COLING*, pages 5569–5578.

Timo Schick and Hinrich Schütze. 2021. Exploiting cloze-questions for few-shot text classification and natural language inference. In *Proc. of EACL*, pages 255–269.

Timo Schick and Hinrich Schütze. 2021. It's not just size that matters: Small language models are also few-shot learners. In *Proc. of NAACL*, pages 2339–2352.

Taylor Shin, Yasaman Razeghi, Robert L Logan IV, Eric Wallace, and Sameer Singh. 2020. Autoprompt: Eliciting knowledge from language models with automatically generated prompts. In *Proc. of EMNLP*, pages 4222–4235.

Jake Snell, Kevin Swersky, and Richard Zemel. 2017. Prototypical networks for few-shot learning. In *Proc. of NIPS*, volume 30, pages 4077–4087.

Oriol Vinyals, Charles Blundell, Timothy Lillicrap, Daan Wierstra, et al. 2016. Matching networks for one shot learning. In *Proc. of NIPS*, volume 29, pages 3630–3638.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Rémi Louf, Morgan Funtowicz, et al. 2019. Huggingface's transformers: State-of-the-art natural language processing. *arXiv preprint arXiv:1910.03771*.

Shiyao Xu and Yang Xiang. 2021. Frog-gnn: Multi-perspective aggregation based graph neural network for few-shot text classification. *Expert Systems with Applications*, 176:114795.

Huaxiu Yao, Ying-xin Wu, Maruan Al-Shedivat, and Eric Xing. 2021. Knowledge-aware meta-learning for low-resource text classification. In *Proc. of EMNLP*, pages 1814–1821.

Zhi-Xiu Ye and Zhen-Hua Ling. 2019. Multi-level matching and aggregation network for few-shot relation classification. In *Proc. of ACL*, pages 2872–2881.

Zihao Zhao, Eric Wallace, Shi Feng, Dan Klein, and Sameer Singh. 2021. Calibrate before use: Improving few-shot performance of language models. In *International Conference on Machine Learning*, pages 12697–12706.

Ruiqi Zhong, Kristy Lee, Zheng Zhang, and Dan Klein. 2021a. Adapting language models for zero-shot learning by meta-tuning on dataset and prompt collections. In *Findings of EMNLP*, pages 2856–2878.

Zexuan Zhong, Dan Friedman, and Danqi Chen. 2021b. Factual probing is [MASK]: learning vs. learning to recall. In *Proc. of NAACL-HLT*, pages 5017–5033.

A Full experiment results

Here we provide full experiment results with standard deviation. Full results are shown in Table 6.

Method	20 News		Amazon		HuffPost		Reuters		Average	
	1 shot	5 shot	1 shot	5 shot	1 shot	5 shot	1 shot	5 shot	1 shot	5 shot
1-NN	38.8	51.9	51.4	67.1	31.5	42.3	57.8	82.9	44.88	61.05
FT	33.0	47.1	45.7	63.9	32.4	44.1	70.9	91.0	45.50	61.53
PROTO	37.8	46.5	41.9	59.2	34.8	50.2	61.0	72.1	43.88	57.00
MAML	37.2	48.6	43.6	62.4	38.9	53.7	61.5	72.0	45.30	59.18
RR	44.8	64.3	60.2	79.7	37.6	59.5	69.1	93.0	52.93	74.13
DS (2019)	52.1	68.3	62.6	81.1	43.0	63.5	81.8	96.0	59.88	77.23
DE-MLMAN (2021)	–	–	–	–	49.7	60.9	–	–	–	–
DE-MAML (2021)	–	–	–	–	51.8	67.3	–	–	–	–
DE-PROTO (2021)	–	–	–	–	52.3	69.6	–	–	–	–
KGML-PROTO (2021)	–	–	58.6	74.5	42.3	58.7	–	–	–	–
KGML-MAML (2021)	–	–	51.4	58.8	44.2	54.1	–	–	–	–
P-TUNING (2021c)	56.21	77.70	62.18	79.13	54.48	65.75	90.01	96.66	65.72	79.81
FROG-GNN (2021)	–	–	71.5	83.6	54.1	69.6	–	–	–	–
LASAML-PN (2021)	–	–	–	–	62.1	70.1	–	–	–	–
CONTRASTNET (2022)	**71.74**	81.57	76.13	85.17	53.06	65.32	86.42	95.33	71.84	81.85
OURS (PRETRAIN INIT)	68.42±0.06	79.11±0.79	75.12±0.65	83.27±0.03	70.82±0.74	75.47±0.24	95.07±0.15	**97.29**±0.06	77.36	83.79
OURS (META INIT)	68.83±0.21	**82.95**±0.12	**77.65**±0.47	**85.54**±0.03	**71.93**±0.22	**76.32**±0.05	**95.20**±0.12	97.17±0.18	**78.40**	**85.50**

Table 6: Full results of 1-shot and 5-shot classification on four datasets in terms of accuracy. The rows below the mid-line are results of MetaPrompting. '-' means that the result of this dataset is not given in the original paper.

Parameter-Efficient Mixture-of-Experts Architecture for Pre-trained Language Models

Ze-Feng Gao[1,4,5*], **Peiyu Liu**[1,4*], **Wayne Xin Zhao**[1,4†], **Zhong-Yi Lu**[2] and **Ji-Rong Wen**[1,3,4]
[1]Gaoling School of Artificial Intelligence, Renmin University of China
[2]Department of Physics, Renmin University of China
[3] School of Information, Renmin University of China
[4]Beijing Key Laboratory of Big Data Management and Analysis Methods
[5]Beijing Academy of Artificial Intelligence, Beijing, 100084, China
{zfgao,liupeiyustu,zlu,jrwen}@ruc.edu.cn, batmanfly@gmail.com

Abstract

Recently, Mixture-of-Experts (short as MoE) architecture has achieved remarkable success in increasing the model capacity of large-scale language models. However, MoE requires incorporating significantly more parameters than the base model being extended. In this paper, we propose building a parameter-efficient MoE architecture by sharing information among experts. We adopt matrix product operator (MPO, a tensor decomposition from quantum many-body physics) to reconstruct the parameter matrix in the expert layer and increase model capacity for pre-trained language models by sharing parameters of the central tensor (containing the core information) among different experts while enabling the specificity through the auxiliary tensors (complementing the central tensor) of different experts. To address the unbalanced optimization issue, we further design the gradient mask strategy for the MPO-based MoE architecture. Extensive experiments based on T5 and GPT-2 show improved performance and efficiency of the pre-trained language model (27.2x reduction in total parameters for the superior model performance, compared with the Switch Transformers). Our code is publicly available at https://github.com/RUCAIBox/MPOE.

1 Introduction

Large-scale pre-trained language models (PLMs), such as BERT (Devlin et al., 2018) and T5 (Raffel et al., 2020), have become the de facto standard in natural language processing (NLP). By involving a huge number of parameters pre-trained on the general-purpose corpus, PLMs can achieve excellent performance in many NLP tasks. In order to increase the model capacity, a promising direction is to explore the scaling properties with the mixture-of-experts (MoE) paradigm (Jacobs et al., 1991; Shazeer et al., 2017) for developing more powerful PLMs. By incorporating multiple expert networks, MoE schedules the learning of data samples through a routing component that is usually implemented by some gating function, which increases model capacity without a proportional increase in computation costs. Despite the effectiveness, it has been shown that the MoE architecture is parameter inefficient (Zuo et al., 2021), considering the yielded improvement *w.r.t.* the involved costs. Most of the existing studies (Yang et al., 2021; Roller et al., 2021; Lewis et al., 2021) attribute this issue to the unbalanced load of experts, focusing on improving the routing strategies.

However, an important question about the MoE architecture has been neglected in previous studies: whether the increased parameters from the experts are all necessary to increase the model capacity. As different experts from an MoE network are often trained with correlated data samples (*e.g.,* sample correlation from training data), it is likely to lead to parameter redundancy across experts. Indeed, *expert redundancy* has been identified in existing studies, where Fedus et al. (2021) distills sparse MoE models into dense models and Kim et al. (2021) prunes experts to compress MoE models. This finding motivates us to develop a parameter-efficient MoE architecture by reducing its parameter redundancy. Intuitively, a straightforward approach is to share a certain proportion of parameters among experts. However, it is difficult to identify and optimize the key parameters that encode the shared information across experts, since expert networks typically consist of dense matrices.

To address this issue, we propose a novel parameter sharing approach inspired by the matrix product operators (MPO) decomposition from quantum many-body physics (Gao et al., 2020), which decomposes a matrix into a sequential product of local tensors (either *central* or *auxiliary* tensors). Unlike other matrix decomposition methods, MPO

[*] Authors contributed equally.
[†] Corresponding author.

Proceedings of the 29th International Conference on Computational Linguistics, pages 3263–3273
October 12–17, 2022.

can effectively reorganize and aggregate important information of the original matrix into the *central tensor*. The auxiliary tensors, on the other hand, serve to complement the central tensor for recovering the original matrix (Liu et al., 2021). In the setting of MoE, considering the small parameter variations among experts, we speculate that the central tensors of different experts (with MPO decomposition for each expert) are likely to be very similar. If the central tensors could be shared for all expert networks, we would significantly reduce the parameters of the MoE architecture.

To this end, we propose a novel MPO-based parameter-efficient MoE architecture, called **MPOE**. Based on classic MoE architecture (Shazeer et al., 2017), our approach introduces a major extension allowing experts to share a global central tensor while keeping expert-specific auxiliary tensors. In our setting, the parameter matrix of in a single expert is formed by the product of the globally shared central tensor and the corresponding auxiliary tensors. Since the central tensor contains most of the parameters from an MPO decomposition, our MPOE approach can significantly reduce the parameters of the original MoE architecture. Another major merit of MPO is that auxiliary tensors are closely entangled with the central tensor (Pirvu et al., 2010), and it is theoretically guaranteed that any change from auxiliary tensors can be propagated to the central tensor. That is to say, though a large proportion of parameters are shared, local auxiliary tensors still enable the experts to capture specific variations or differences according to routing data samples. However, directly optimizing the MPOE architecture is likely to lead to an *unbalanced optimization* issue, where the central tensors are updated more frequently than auxiliary tensors during fine-tuning. Therefore, we further propose a gradient mask strategy that masks the central tensor gradient to effectively alleviate the unbalanced optimization issue.

To the best of our knowledge, this is the first attempt to reduce the parameter redundancy of the MoE architecture with structural matrix decomposition. We conduct extensive experiments to evaluate the effectiveness of the MPOE architecture on two representatives PLMs, T5 and GPT. Experiments have demonstrated the effectiveness of our approach in increasing model capacity (27.2x fewer parameters for the superior model performance, compared with several competitive MoE-enhanced PLMs.

2 Preliminary

2.1 Mixture-of-Experts (MoE)

We first describe the mixture-of-experts architecture (MoE) (Shazeer et al., 2017), which has been used to enhance the model capacity of Transformer based models. Let $G(x)$ and $E_i(x)$ denote the output vectors of the gating network and the output of the i-th expert network for a given input x, respectively. The output of MoE architecture y can be formally computed as:

$$y = \sum_{i=1}^{n} G(x) \cdot E_i(x). \quad (1)$$

The $\mathrm{softmax}$ function is widely adopted as the gating function $G(x)$. The sparsely-gated MoE architecture, which uses a noisy top-k gating mechanism to reduce the computational cost, has been proposed in Shazeer et al. (2017). It adds tunable Gaussian noise with $H(\cdot)$, and then keeps only the top-k values with $\mathrm{KeepTopK}(\cdot)$ and sets the rest $-\infty$. This keeps only the top k experts to be evaluated with:

$$G(x) = \mathrm{softmax}(\mathrm{KeepTopK}(H(x), k)). \quad (2)$$

Furthermore, Switch Transformer designs a switch routing strategy to simplify this gating function by routing to a *single* expert (Fedus et al., 2021).

2.2 Tensor and Matrix Product Operators

We refer to one-dimensional arrays as *vectors* (denoted by bold lowercase letters, *e.g.,* $\boldsymbol{v}$), two-dimensional arrays as *matrices* (denoted by bold uppercase letters, *e.g.,* $\mathbf{W}$), and arrays of higher dimensions as *tensors* (denoted by calligraphic bold uppercase letters, *e.g.,* $\mathcal{T}$).

MPO decomposition (Oseledets, 2011) (*a.k.a.* tensor-train decomposition) has been a widely used matrix decomposition technique from quantum many-body physics, which decomposes a matrix (2-order tensor) into m local tensors (Pirvu et al., 2010). Given a matrix $\mathbf{W}_{I\times J} \in \mathbb{R}^{I\times J}$, the MPO decomposition is given in the following format:

$$\mathrm{MPO}\ (\mathbf{W}) = \prod_{k=1}^{m} \mathcal{T}_{(k)}[d_{k-1}, i_k, j_k, d_k], \quad (3)$$

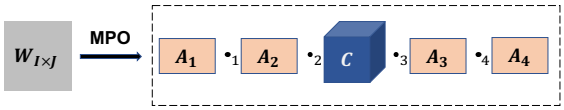

Figure 1: MPO decomposition for matrix $\mathbf{W}_{I\times J}$ with five local tensors. Auxiliary tensors ($\{\mathcal{A}_i\}_{i=1}^4$) and central tensor ($\mathcal{C}$) are marked in orange and blue, respectively.

where $I = \prod_{k=1}^n i_k$ and $J = \prod_{k=1}^n j_k$, $\mathcal{T}_{(k)}$ is a 4-order tensor with size $d_{k-1} \times i_k \times j_k \times d_k$. The d_k is dimension of bond linking $\mathcal{T}_{(k)}$ and $\mathcal{T}_{(k+1)}$.

According to Gao et al. (2020), the original matrix $\mathbf{W}$ can be exactly reconstructed by tensor contraction of MPO($\mathbf{W}$) without truncation of the connection bond $\{d_k\}_{k=1}^m$. Figure 1 presents the illustration of the MPO decomposition procedure for a matrix ($m = 5$). More detailed analysis on different factorization ways (*i.e.,* $m = 3, 5, 7, 9$) will be given in Section 4.5. After MPO decomposition, the central tensor (the tensor right in the middle) with most of the parameters can encode the core information of the original matrix, while the auxiliary tensors (the rest of these tensors) with only a small proportion of parameters play the role of complementing the central tensor.

3 Approach

To reduce the information redundancy across different experts, we design an MPO-based MoE architecture for increasing the model capacity in a parameter-efficient way. We firstly describe the MPO-based MoE architecture and then introduce an improved optimization algorithm for learning the parameters in this architecture.

3.1 MPO-based Mixture-of-Experts

Previous MoE architecture (Jacobs et al., 1991; Shazeer et al., 2017) usually treats different experts as individual components, requiring a compete copy of network parameters for each expert. Although it has been found (Fedus et al., 2021; Kim et al., 2021) that there exists redundant information among different experts in the MoE architecture, it is not easy to identify the shareable parameters from the highly coupling network.

Considering this issue, our solution is inspired by an important merit of MPO decomposition: it can reorganize and aggregate the core information in central tensors (Gao et al., 2020) as aforementioned. Based on this property, the core idea of our approach is to share the central tensors for all the expert layers and enable specificity via expert-specific auxiliary tensors.

Parameter-Efficient MoE Architecture. The Transformer network consists of two major neural components, namely FFN and multi-head attention. Following previous work on MoE-based PLMs (Shazeer et al., 2017; Fedus et al., 2021), we consider FFN layers as experts to be extended, while our approach is generally applicable to various matrix-based model components. A straightforward method to reducing information redundancy is to share a proportion of parameters across experts. However, in Transformer-based networks, the experts (*i.e.,* FFN here) are mainly composed of large dense matrices, which are difficult for sharing partial parameters from these matrices. As our solution, we consider parameter sharing through the MPO decomposition, so that the derived central tensors can be flexibly shared across matrices.

Lightweight MoE Design. Specifically, we simplify the discussion by assuming that an expert corresponds to one parameter matrix at each layer, and it is similar for the multi-matrix cases. We consider a MoE architecture of n experts each with L layers, so that there are $L \times n$ matrices in total, denoted by $\{\mathbf{W}^{(l,i)}\}_{l=1,i=1}^{L,n}$. As discussed in Section 2.2, a matrix can be decomposed into m tensors, consisting of one central tensor and $m-1$ auxiliary tensors. In this work, we consider five decomposed tensors, *i.e.,* $m = 5$. At the l-th layer, the decomposition results can be denoted by $\{\mathcal{C}^{(l,i)}, \mathcal{A}_1^{(l,i)}, \mathcal{A}_2^{(l,i)}, \mathcal{A}_3^{(l,i)}, \mathcal{A}_4^{(l,i)}\}_{i=1}^n$, where $\mathcal{C}^{(l,i)}$ and $\mathcal{A}^{(l,i)}$ are the central and auxiliary tensors of the i-th parameter matrix, respectively, at the l-th layer. To develop the MPO-based MoE architecture, the core idea is to share the central tensors as global parameters and keep expert-specific auxiliary tensors as local parameters, *i.e.,* $\mathcal{C}^{(l,1)} = \mathcal{C}^{(l,2)} \cdots = \mathcal{C}^{(l,n)}$ ($\forall\, l = 1 \cdots L$), and we denote the global central tensor at the l-th layer by $\mathcal{C}^{(l)}$. In this way, we can only keep L central censors for a L-layer MoE architecture. For MPO, the decomposition process is transparent to external modules, so that we can reuse the previous routing mechanism (Section 2.1) by distributing data samples to different experts. A slight difference is that we only need to consider the routing to local tensors for each matrix since the global tensor is shared across experts. We call such an MPO-based MoE architecture as **MPOE**.

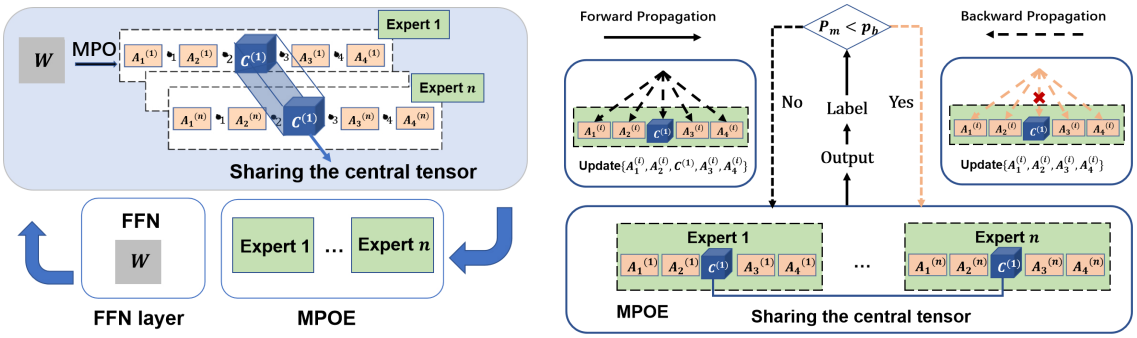

(a) MPO-based mixture-of-experts architecture.

(b) Gradient mask strategy.

Figure 2: Illustration the proposed MPOE architecture and gradient mask strategy. We decompose the weight matrix of each expert in the MoE architecture into five local tensors using MPO, containing four auxiliary tensors and one central tensor, which are marked in orange and blue, respectively. In our approach, the central tensor of the n experts is shared in the MPOE architecture. During optimization, each backward propagation process updates a set of auxiliary tensors while updating the central tensor with a probability of p_b (the mask probability of the central tensor), which can effectively avoid the unbalanced optimization of the central tensor.

Discussion. Since the central tensor contains most of the information from original parameter matrices (Gao et al., 2020), a key question is whether the current architecture enables sufficient flexibility and specificity for each expert. To answer this question, we refer to an important property of MPO decomposition from quantum many-body physics (Pirvu et al., 2010): it is guaranteed in principle, that any change on one tensor will be propagated to the entire local tensor set. In other words, only tuning the auxiliary tensors (keeping the central tensor fixed) can lead to the same effect as tuning the whole matrix. Since the parameters of the central tensor are shared, our approach can significantly reduce the number of actual parameters given the MoE architecture with the same number of experts. Assuming the original model consisting of n experts with T parameters each, we have a total number of $n \cdot T$ parameters. Specifically, let γ denote the parameter ratio of the auxiliary tensor to the central tensor for expert networks. Given the total number T for an expert network, the central and auxiliary tensors correspond to the parameters numbers of $\frac{\gamma}{\gamma+1}T$ and $\frac{1}{\gamma+1}T$, respectively. Since our MPOE approach shares the central tensor, the final number of parameters will be $\frac{\gamma}{\gamma+1}T + \frac{n}{\gamma+1}T$. Thus, our MPOE approaches corresponds to a ratio of $\frac{n+\gamma}{n(\gamma+1)}$ of the original parameter scale. In our experiments, the ratio γ is about 12, and $\frac{n+\gamma}{n(\gamma+1)}$ approximately equals to 0.19 when $n = 8$. Such a ratio will be further decreased when we have more experts. It can be seen that our MPOE approach is able to effectively reduce the parameter scale.

3.2 Alleviate Unbalanced Optimization

As the experts share the central tensor in the MPOE approach, the corresponding parameters of the central tensor will be updated more frequently than those in the auxiliary tensors during fine-tuning. It tends to lead to the unbalanced optimization issue as reported by Xu et al. (2021), due to deviation from the pre-trained weights. As a result, it is crucial to develop a more stable optimization technique that is suited to the MPOE architecture.

Inspired by the solution of gradient dropout strategy (Tseng et al., 2020; Xu et al., 2021), we propose to mask the gradients for the central tensor to improve model optimization for the MPO-based MoE architecture. At each iteration, we take a certain probability p_b to discard the update in the central tensor. This can effectively alleviate the unbalanced optimization which is caused by the frequent updates of the central tensor. Specifically, we generate a binary mask b drawn from Bernoulli distribution with a mask probability p_b, which can be calculated by $b \sim \text{Bernoulli}(p_b)$. We denote the $\Delta\mathcal{C}$ as the update of the central tensor at each iteration:

$$\Delta\mathcal{C} = \eta \frac{\partial \mathcal{L}(\mathcal{C})}{\partial \mathcal{C}} \odot (1 - b). \tag{4}$$

The larger p_b is, the less frequently the central tensor is updated. In particular, when p_b is equal to 1, it means that the parameters of the central tensor are frozen for each input of the data. The computational cost of central tensor update can be also reduced with this trick.

Note that the gradient mask trick is only applied to central tensors. For auxiliary tensors, we perform the standard gradient update for learning the parameters. Compared with two alternative ways to implement the gradient mask technique, *i.e.,* mask pre-activation or post-activation in FFN layers, we find that such a sampling-based masking strategy can effectively improve the model performance in our experiments.

3.3 The Overall Algorithm Procedure

Our approach can be generally applied to various MoE-based models for increasing the model capacity. In this work, we adopt the MoE-extended PLMs (Radford et al., 2019) for study.

Algorithm 1 presents a complete procedure for the proposed update procedure, which can be briefly summarized as follows. First, we obtain the PLM and perform MPO decomposition for each weight matrix of the FFN layers in the Transformer. For each weight matrix, we decompose it into one central tensor $\mathcal{C}$ and a list of auxiliary tensors $\mathcal{A}$. In the original MoE architecture, we will have n sets of such decomposed parameters. Next, the key point lies in that we share the central tensor $\mathcal{C}$ in the decomposition process but keep expert-specific auxiliary tensors. In this way, each expert is composed of a set of auxiliary tensors and a shared central tensor. To recover the original FFN matrix in some specific expert, we can simply multiply the shared central tensor by expert-specific auxiliary tensors. Then, we apply the gradient mask strategy to update the parameters in these experts, *i.e.,* masking the gradient of the central tensor.

Since the parameters of the central tensor are two orders of magnitude larger than the parameters of the auxiliary tensors (Liu et al., 2021), the cost of MoE-based networks will be largely reduced by sharing the central tensor.

3.4 Discussion

For the parameter inefficiency issue of MoE-based networks, existing studies mainly focus on alleviating the unbalanced load of experts, which have proposed different routing methods to balance the routing probabilities of different experts, such as BASE-Layer (Lewis et al., 2021), HASHLayer (Roller et al., 2021), GShard (Lepikhin et al., 2021) and Switch Transformers (Fedus et al., 2021). As a comparison, we aim to reduce information redundancy by sharing common parameters among experts. Actually, the MPOE approach can be further enhanced with existing improved routing methods.

Specifically, Deepspeed-MoE proposed to use pyramid residual MoE architecture to reduce the parameters of the MoE architecture (Rajbhandari et al., 2022), while our work takes a different perspective to improve the original MoE architecture by sharing parameters among different experts.

Algorithm 1 The proposed updating procedure.

Require: $\{\{\mathcal{A}_j\}_{j=1}^4, \mathcal{C}\}$: Initialize experts
Require: α: learning rate
Require: p_b: mask probability
Require: time step $t \leftarrow 0$ (Initialize timestep)
while not converged **do**
  $t \leftarrow t + 1$
  $g_{\mathcal{C}}^t \leftarrow \frac{\partial \mathcal{L}(\mathcal{C}^t)}{\partial(\mathcal{C}^t)}, \quad g_{\mathcal{A}}^t \leftarrow \frac{\partial \mathcal{L}(\mathcal{A}^t)}{\partial(\mathcal{A}^t)}$
  (Get gradients at timestep t)
  $b \leftarrow \mathrm{GenerateMask}(p_b)$
  (Compute gradient mask)
  $\mathcal{C}^t \leftarrow \mathcal{C}^{t-1} - \alpha \cdot g_{\mathcal{C}}^t \odot (1 - b)$
  (Update central tensors)
  $\mathcal{A}^t \leftarrow \mathcal{A}^{t-1} - \alpha \cdot g_{\mathcal{A}}^t$
  (Update the routed auxiliary tensors)
end while
return $\{\{\mathcal{A}_j^t\}_{j=1}^4, \mathcal{C}^t\}$ (Resulting parameters)

4 Experiments

In this section, we first set up the experiments and then report the results and analysis. Then, we conduct a detailed analysis under different experimental settings. Here, we use T5 (Raffel et al., 2020) and GPT-2 (Radford et al., 2019) models as the base model in our experiments.

4.1 Experimental Setup

Datasets. To evaluate the effectiveness of the proposed MPOE as an efficient strategy to improve the model capacity of PLMs, we follow the setting of T5 and GPT-2 to perform experiments on Natural Language Understanding (NLU) and Natural Language Generation (NLG) tasks. Specifically, we evaluate the NLG tasks in GLUE benchmark (Wang et al., 2018), the language modeling task with WikiText-2 (Merity et al., 2017), the text generation task with IMDB (Maas et al., 2011) and EMNLP2017 WMT News (Guo et al., 2018). Furthermore, we follow the setup of Raffel et al. (2020) on the GLUE benchmark for a direct comparison with the T5 model.

GLUE benchmark covers multiple datasets

NLU with T5										
Experiments	MNLI	QNLI	SST-2	RTE	QQP	CoLA	MRPC	STS-B	Avg.	#To (M)
T5-Large	**89.23**	94.03	96.20	83.94	**91.54**	55.10	90.15	91.90	86.51	737
+MoEfication♦	87.50	93.20	95.40	86.40	90.20	55.50	87.50	90.60	85.79	737
+MoEfication$_{++}$ ♦	88.70	93.60	96.20	87.50	91.30	59.40	89.30	91.00	87.13	737
+Switch♣	/	/	/	/	/	/	/	/	88.50	26000
+MPOE	87.16	**94.12**	**96.80**	**88.60**	90.63	**67.63**	**93.65**	**91.97**	**88.82**	956
T5-Base	87.78	93.82	94.72	71.74	91.11	53.49	89.16	91.16	84.12	223
+Switch♣	/	/	/	/	/	/	/	/	86.70	3800
+Switch♠	87.73	93.85	**94.87**	**77.53**	91.59	59.90	91.64	91.16	86.03	1015
+MoE★	86.98	92.82	94.60	69.56	90.02	64.56	87.68	90.89	84.64	1015
+MPOE	87.60	93.30	94.81	77.13	90.81	65.53	**93.14**	91.17	86.69	294
+MPOE$_{++}$	**87.78**	**93.93**	94.83	77.42	**91.61**	**65.90**	91.14	**91.65**	**86.78**	365

NLG with GPT-2								
Experiments	WikiText-2			EMNLP News		IMDB		#To (M)
	PPL (↓)	BLEU-2	BLEU-4	BLEU-2	Self-BLEU-2	BLEU-2	Self-BLEU-2	
GPT-2	21.27	28.69	9.46	62.61	74.67	73.12	83.85	124
+MoE★	21.86	28.27	9.14	65.27	79.79	74.46	90.01	578
+Switch♠	21.25	28.71	9.44	64.62	81.11	75.35	91.82	578
+MPOE	**20.72**	28.78	9.51	66.99	83.10	76.30	92.72	157
+MPOE$_{++}$	20.73	**28.82**	**9.57**	**68.49**	**83.11**	**76.82**	**93.08**	171

Table 1: Performance comparison of different models on NLU and NLG tasks (in percent). "#To (M)" denote the number (in millions) of total parameters. We set the number of experts $n = 8$ in these models, MPOE. Furthermore, we use $n = 16$ for a more powerful version of our approach, denoted by MPOE$_{++}$. We report the average test performance of three runs, and the best results are highlighted in bold. ♦: Experimental results by Zhang et al. (2021b) ♣: Experimental results by Fedus et al. (2021) ♠: Our re-implementation by Fedus et al. (2021). ★:Apply method by Shazeer et al. (2017).

(MRPC, QQP, SST-2, MNLI, RTE, QNLI, CoLA)[1]. The original test sets are not publicly available, and following Zhang et al. (2021a), for datasets fewer than $10K$ samples (RTE, MRPC, STS-B, CoLA), we divide the original validation set in half, using one half for validation and the others for the test.

Evaluation Metrics. We use perplexity (PPL) (Brown et al., 1992) to measure how well the probability model predicts a sample compared with the ground-truth. To evaluate the ratios of the overlapping n-grams between generated and real samples, we use BLEU-n score (Papineni et al., 2002). We also take into account the Self-BLEU-n score (Zhu et al., 2018) to evaluate the diversity of generated samples especially. For metrics used in the GLUE benchmark, we follow Mahabadi et al. (2021) and use Matthew's correlation for CoLA, Pearson for STS-B, and accuracy for the other tasks.

Comparison methods. We adopt the T5 and GPT-2 as the base architectures for both MoE and MPOE. Following Shazeer et al. (2017), we extend the FFN components with the MoE architecture containing n experts in each Transformer block of the T5 and GPT-2 model. We refer to this method as "+MoE". The Switch Transformers (Fedus et al., 2021) use a simplified strategy that routes to only a single expert instead of top-2 routing in MoE. We refer to this method as "+Switch". To ensure a fair comparison, we maintain the same number ($n = 8$) of experts for baselines and MPOE. We also implement an enhanced version of MPOE with $n = 16$ experts, which is referred to as "+MPOE$_{++}$". Based on the released *gpt2* model[2], *t5-base* model[3] and *t5-large* model[4] provided by Huggingface, we first initialize the experts, then fine-tune the models on the downstream tasks. For the T5 model, we follow the setting in Mahabadi et al. (2021) and fine-tune all parameters of the model on all tasks. For different downstream tasks, we run a hyperparameter sweep and select the best configuration according to the accuracy results on the validation set. The hyperparameters that we tune include the epochs, batch size and learning rates.

[1]Following Raffel et al. (2020), as a common practice, due to the adversarial nature of WNLI with respect to the training set, we do not experiment with WNLI

[2]https://huggingface.co/gpt2

[3]https://huggingface.co/t5-base

[4]https://huggingface.co/t5-large

4.2 Mains Results

In our main experiments, we adopt T5 (Raffel et al., 2020), GPT-2 (Radford et al., 2019), Switch Transformers (Fedus et al., 2021) and MoEfication (Zhang et al., 2021b) as baselines, and report the comparison results of both NLU and NLG tasks in Table 1.

Overall, compared to these MoE variants, our proposed MPOE approach achieves performance improvement while being more parameter-efficient. For the NLU task, our proposed approach ("+MPOE") outperforms the best baseline method, *i.e.,* "+Switch" (88.82 vs. 88.50 for T5-Large) with up to 27.2x reduction in total parameters in the GLUE benchmark. By zooming into low-resource datasets such as CoLA and MRPC, our approach yields more significant improvements. This suggests that sharing parameters across experts reinforces the positive transfer effects[5] of information from other datasets toward the learning of low-resource datasets. For the NLG task, GPT-2+MPOE achieves gains in BLEU-2 score (1.72 for GPT-2+MoE and 2.37 for GPT-2+Switch) with 3.7x reduction in total parameters on the EMNLP News dataset. This indicates that GPT-2 also benefits from sharing central tensors.

Moreover, T5+MPOE$_{++}$ and GPT-2+MPOE$_{++}$ perform better when we add more auxiliary tensors as additional experts. This demonstrates the necessity of improving model capacity (Shazeer et al., 2017), as more parameters of experts tend to result in an improved model capacity.

4.3 Evaluation on Multi-task Learning

To demonstrate the efficiency of MPOE in multi-task learning, we adopt the T5-Base model for analysis to be comparable with Hyperformer (Mahabadi et al., 2021). We conduct experiments on the multi-task GLUE benchmark. The detailed metrics can be found in Section 4.1. Note that compared to Hyperformer, MPOE approach does not incorporate additional neural network components, thus it is more flexible to be used with the PLMs.

Table 2 shows the results on GLUE benchmark for T5-base (Raffel et al., 2020), Hyperformer (Mahabadi et al., 2021) and MPOE. As we can see, the performance of the MPOE approach is consistently better than the Hypernetwork in all cases, while the MPOE is more parameter-efficient (258M vs. 343M in total parameters). It further demonstrates the potential benefits of the MPOE approach in a multi-task learning setting, where the central tensor learns common information across tasks and the auxiliary tensor learns task-specific information.

[5]Here, the positive transfer effects can be referred to Mahabadi et al. (2021), which means that the transferred knowledge can lead to improved performance for unseen in-domain tasks.

Datasets	T5-Base	Hyper♣	+MPOE
MNLI (acc)	87.73	85.74	**87.83**
QNLI (acc)	93.51	93.02	**93.89**
SST-2 (acc)	92.50	94.03	**94.73**
RTE (acc)	75.41	75.36	**75.51**
QQP (acc)	91.12	90.28	**91.17**
CoLA (mcc)	54.93	63.73	**65.85**
MRPC (acc)	89.21	89.66	**90.10**
STS-B (pearson)	90.75	90.00	**90.92**
Avg.	84.39	85.23	**86.25**
#To (M)	223	343	258

Table 2: Performance of multi-task learning on GLUE benchmark obtained by fine-tuning T5-Base (in percent). ♣: Experimental results from Hyperformer (Mahabadi et al., 2021).

Variants	WikiText-2 PPL (↓)	WikiText-2 B2	WikiText-2 B4	#To (M)
+MoE★	21.86	28.27	9.14	578
w/o PS	21.28	28.67	9.44	153
w/o GM	21.17	28.71	9.47	157
+MPOE	**20.72**	**28.78**	**9.51**	157

Table 3: Ablation study on the WikiText-2 dataset about the NLG tasks (in percent). "B2" and "B4" are short for BLEU-2 and BLEU-4, respectively. ★: The method from Shazeer et al. (2017)

4.4 Ablation Results

Our approach has incorporated two novel improvements: (1) MoE architecture with parameters sharing (PS) among experts based on MPO decomposition and (2) gradient mask (GM) to alleviate unbalanced optimization.

To verify the effectiveness of each component, we conduct the ablation study on the WikiText-2 dataset to analyze the contribution of each part. We adopt PPL, BLEU-2 and BLEU-4 as the evaluation metrics, and consider removing the parameters sharing and gradient mask strategy respectively. The ablation results of our MPOE approach are shown in Table 3. We can see that removing any component would lead to a decrease in the model performance. It shows the effectiveness of all these components in our approach. Besides, parameter sharing seems more important than the gradient

Variants	WikiText-2 PPL (↓)	B2	B4	#To (M)
GPT-2	21.27	28.69	9.46	124.4
MPOE (m=3)	24.01	27.86	8.93	130.3
MPOE (m=5)	20.72	28.77	9.48	157.4
MPOE (m=7)	20.73	28.76	9.47	198.7
MPOE (m=9)	20.78	28.45	9.38	214.6

Table 4: Evaluation with different factorization manner on the WikiText-2 dataset about the NLG tasks (in percent). "B2" and "B4" are short for BLEU-2 and BLEU-4, respectively.

mask strategy, which yields a larger performance drop after being removed.

4.5 Detailed Analysis

MPO decomposition has different factorization manners. However, the MPOE approach requires a defined MPO decomposition form to be given before it can be used. Therefore, different factorization manners may affect the efficiency of the MPOE approach. To vertify this, we perform a detailed analysis on different factorization manners of MPO decomposition. We present three variants of MPOE with different lengths of local tensors produced by MPO decomposition empirically. Tabel 4 shows the evaluation results on the WikiText-2 dataset about NLG tasks. As we can see, the variants of $m > 3$ are all superior to the GPT-2 model. Additionally, we can observe that more local tensors performs similarly but leads to higher memory cost. Thus we finally choose to set $m = 5$ for MPOE architecture considering the trade-off between the cost and quality.

5 Related Work

We will review the related works in four aspects.

PLMs with MoE. It has been reported that models with more parameters are usually considered to have a larger model capacity (Fedus et al., 2021; Zuo et al., 2021). In order to increase the model capacity, a promising direction is to explore the scaling properties with MoE architecture which was introduced by Jacobs et al. (1991). Thus, Shazeer et al. (2017) first applied the MoE architecture to large-scale language models. Then, Switch Transformers (Fedus et al., 2021), GShard (Lepikhin et al., 2021), BASELayer (Lewis et al., 2021) and HashLayer (Roller et al., 2021) studied how to build large-scale Transformer-based model with MoE as well as improving routing strategy, which can better utilize the model capacity. In addition, Zhang et al. (2021b) proposed a strategy for sparse activation of MoE architecture. He et al. (2021) suggested a distributed training system for fast training of MoE. Zoph et al. (2022) proposed a sparse expert model with more stable training. Yu et al. (2022) proposed a sparse expert model based on all-MLP architecture. In contrast, our approach aims to reduce information redundancy by sharing parameters among experts.

Matrix Product Operators Decomposition. Matrix product operators (MPO) (Pirvu et al., 2010) decomposition was proposed in quantum many-body physics, *a.k.a.* tensor-train (TT) decomposition (Oseledets, 2011). A major category of MPO studies relies on model compression (Gao et al., 2020). They focus on compressing weight matrix and convolutional layers (Novikov et al., 2015; Garipov et al., 2016; Sun et al., 2020). Furthermore, the MPO decomposition was used to compress the PLMs as well as enable lightweight fine-tuning in downstream tasks (Liu et al., 2021). In this work, we utilize such a decomposition mechanism for parameter sharing to construct a parameter-efficient MoE architecture.

Improved Variants of MoE. Despite the achieved performance performance, MoE architecture has been hindered by the model complexity and high memory costs (Shazeer et al., 2017; Fedus et al., 2021). This problem can be alleviated by using distillation (Fedus et al., 2021) and expert pruning (Kim et al., 2021). Then, Kudugunta et al. (2021) and Zuo et al. (2021) indicated that sub-networks can be employed when using the model. Indeed, our approach can be further enhanced by these existing methods for improving inference time.

Multi-task Learning. The exploitation of MoE architectures for multi-task learning is a very promising direction in recent years (Ma et al., 2018). Houlsby et al. (2019) suggested training adapters for each task separately while keeping the model fixed. Further research suggested that model parameters could be shared across tasks, and task-specific adapter parameters were introduced (Stickland and Murray, 2019). Based on this idea, Mahabadi et al. (2021) and Pilault et al. (2020) proposed that parameter-efficient multi-task fine-tuning for transformer-based models via shared hypernetworks. Our approach differs from these

works in that the MPOE approach allows us to reduce model size while keeping the same number of experts, and meanwhile achieve performance improvement for multi-task learning.

6 Conclusion

In this paper, we proposed a parameter-efficient MoE architecture for increasing model capacity based on the MPO decomposition. First, we shared the central tensors among different experts based on MPO decomposition, which largely reduced the model parameters of MoE architecture. Then, we designed the gradient mask strategy to alleviate the unbalanced optimization issues and ensured that different tensors capture different types of information efficiently. Extensive experiments have shown that our approach outperforms several competitive PLM scaling strategies, especially in terms of improving the parameter efficiency of the MoE architecture.

In the future, we will enhance the proposed MPOE approach with recently proposed routing methods, such as BASELayer (Lewis et al., 2021), HASHLayer (Roller et al., 2021) and GShard (Lepikhin et al., 2021). We will also consider exploring additional decomposition methods for developing parameter-efficient MoE architecture.

Acknowledgments

This work was partially supported by Beijing Natural Science Foundation under Grant No. 4222027, National Natural Science Foundation of China under Grants No. 62206299 and 11934020, Beijing Outstanding Young Scientist Program under Grant No. BJJWZYJH012019100020098 and Beijing Academy of Artificial Intelligence (BAAI). Xin Zhao is the corresponding author.

References

Peter F. Brown, Stephen Della Pietra, Vincent J. Della Pietra, Jennifer C. Lai, and Robert L. Mercer. 1992. An estimate of an upper bound for the entropy of english. *Comput. Linguistics*, 18(1):31–40.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*.

William Fedus, Barret Zoph, and Noam Shazeer. 2021. Switch transformers: Scaling to trillion parameter models with simple and efficient sparsity. *arXiv preprint arXiv:2101.03961*.

Ze-Feng Gao, Song Cheng, Rong-Qiang He, ZY Xie, Hui-Hai Zhao, Zhong-Yi Lu, and Tao Xiang. 2020. Compressing deep neural networks by matrix product operators. *Physical Review Research*, 2(2):023300.

Timur Garipov, Dmitry Podoprikhin, Alexander Novikov, and Dmitry Vetrov. 2016. Ultimate tensorization: compressing convolutional and fc layers alike. *arXiv preprint arXiv:1611.03214*.

Jiaxian Guo, Sidi Lu, Han Cai, Weinan Zhang, Yong Yu, and Jun Wang. 2018. Long text generation via adversarial training with leaked information. In *Proceedings of the Thirty-Second AAAI Conference on Artificial Intelligence, (AAAI-18), the 30th innovative Applications of Artificial Intelligence (IAAI-18), and the 8th AAAI Symposium on Educational Advances in Artificial Intelligence (EAAI-18), New Orleans, Louisiana, USA, February 2-7, 2018*, pages 5141–5148. AAAI Press.

Jiaao He, Jiezhong Qiu, Aohan Zeng, Zhilin Yang, Jidong Zhai, and Jie Tang. 2021. Fastmoe: A fast mixture-of-expert training system. *arXiv preprint arXiv:2103.13262*.

Neil Houlsby, Andrei Giurgiu, Stanislaw Jastrzebski, Bruna Morrone, Quentin de Laroussilhe, Andrea Gesmundo, Mona Attariyan, and Sylvain Gelly. 2019. Parameter-efficient transfer learning for NLP. In *Proceedings of the 36th International Conference on Machine Learning, ICML 2019, 9-15 June 2019, Long Beach, California, USA*, volume 97 of *Proceedings of Machine Learning Research*, pages 2790–2799. PMLR.

Robert A Jacobs, Michael I Jordan, Steven J Nowlan, and Geoffrey E Hinton. 1991. Adaptive mixtures of local experts. *Neural computation*, 3(1):79–87.

Young Jin Kim, Ammar Ahmad Awan, Alexandre Muzio, Andres Felipe Cruz Salinas, Liyang Lu, Amr Hendy, Samyam Rajbhandari, Yuxiong He, and Hany Hassan Awadalla. 2021. Scalable and efficient moe training for multitask multilingual models. *arXiv preprint arXiv:2109.10465*.

Sneha Kudugunta, Yanping Huang, Ankur Bapna, Maxim Krikun, Dmitry Lepikhin, Minh-Thang Luong, and Orhan Firat. 2021. Beyond distillation: Task-level mixture-of-experts for efficient inference. *arXiv preprint arXiv:2110.03742*.

Dmitry Lepikhin, HyoukJoong Lee, Yuanzhong Xu, Dehao Chen, Orhan Firat, Yanping Huang, Maxim Krikun, Noam Shazeer, and Zhifeng Chen. 2021. Gshard: Scaling giant models with conditional computation and automatic sharding. In *9th International Conference on Learning Representations, ICLR 2021, Virtual Event, Austria, May 3-7, 2021*. OpenReview.net.

Mike Lewis, Shruti Bhosale, Tim Dettmers, Naman Goyal, and Luke Zettlemoyer. 2021. Base layers: Simplifying training of large, sparse models. *arXiv preprint arXiv:2103.16716*.

Peiyu Liu, Ze-Feng Gao, Wayne Xin Zhao, Zhi-Yuan Xie, Zhong-Yi Lu, and Ji-Rong Wen. 2021. Enabling lightweight fine-tuning for pre-trained language model compression based on matrix product operators. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing, ACL/IJCNLP 2021, (Volume 1: Long Papers), Virtual Event, August 1-6, 2021*, pages 5388–5398. Association for Computational Linguistics.

Jiaqi Ma, Zhe Zhao, Xinyang Yi, Jilin Chen, Lichan Hong, and Ed H. Chi. 2018. Modeling task relationships in multi-task learning with multi-gate mixture-of-experts. In *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining, KDD 2018, London, UK, August 19-23, 2018*, pages 1930–1939. ACM.

Andrew L. Maas, Raymond E. Daly, Peter T. Pham, Dan Huang, Andrew Y. Ng, and Christopher Potts. 2011. Learning word vectors for sentiment analysis. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, pages 142–150, Portland, Oregon, USA. Association for Computational Linguistics.

Rabeeh Karimi Mahabadi, Sebastian Ruder, Mostafa Dehghani, and James Henderson. 2021. Parameter-efficient multi-task fine-tuning for transformers via shared hypernetworks. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing, ACL/IJCNLP 2021, (Volume 1: Long Papers), Virtual Event, August 1-6, 2021*, pages 565–576. Association for Computational Linguistics.

Stephen Merity, Caiming Xiong, James Bradbury, and Richard Socher. 2017. Pointer sentinel mixture models. In *5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Conference Track Proceedings*. OpenReview.net.

Alexander Novikov, Dmitry Podoprikhin, Anton Osokin, and Dmitry Vetrov. 2015. Tensorizing neural networks. *arXiv preprint arXiv:1509.06569*.

Ivan V Oseledets. 2011. Tensor-train decomposition. *SIAM Journal on Scientific Computing*, 33(5):2295–2317.

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. Bleu: a method for automatic evaluation of machine translation. In *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics, July 6-12, 2002, Philadelphia, PA, USA*, pages 311–318. ACL.

Jonathan Pilault, Amine Elhattami, and Christopher Pal. 2020. Conditionally adaptive multi-task learning: Improving transfer learning in nlp using fewer parameters & less data. *arXiv preprint arXiv:2009.09139*.

Bogdan Pirvu, Valentin Murg, J Ignacio Cirac, and Frank Verstraete. 2010. Matrix product operator representations. *New Journal of Physics*, 12(2):025012.

Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, Ilya Sutskever, et al. 2019. Language models are unsupervised multitask learners. *OpenAI blog*, 1(8):9.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J. Liu. 2020. Exploring the limits of transfer learning with a unified text-to-text transformer. *J. Mach. Learn. Res.*, 21:140:1–140:67.

Samyam Rajbhandari, Conglong Li, Zhewei Yao, Minjia Zhang, Reza Yazdani Aminabadi, Ammar Ahmad Awan, Jeff Rasley, and Yuxiong He. 2022. Deepspeed-moe: Advancing mixture-of-experts inference and training to power next-generation ai scale. *arXiv preprint arXiv:2201.05596*.

Stephen Roller, Sainbayar Sukhbaatar, Arthur Szlam, and Jason Weston. 2021. Hash layers for large sparse models. *arXiv preprint arXiv:2106.04426*.

Noam Shazeer, Azalia Mirhoseini, Krzysztof Maziarz, Andy Davis, Quoc Le, Geoffrey Hinton, and Jeff Dean. 2017. Outrageously large neural networks: The sparsely-gated mixture-of-experts layer. *arXiv preprint arXiv:1701.06538*.

Asa Cooper Stickland and Iain Murray. 2019. BERT and pals: Projected attention layers for efficient adaptation in multi-task learning. In *Proceedings of the 36th International Conference on Machine Learning, ICML 2019, 9-15 June 2019, Long Beach, California, USA*, volume 97 of *Proceedings of Machine Learning Research*, pages 5986–5995. PMLR.

Xingwei Sun, Ze-Feng Gao, Zhong-Yi Lu, Junfeng Li, and Yonghong Yan. 2020. A model compression method with matrix product operators for speech enhancement. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 28:2837–2847.

Hung-Yu Tseng, Yi-Wen Chen, Yi-Hsuan Tsai, Sifei Liu, Yen-Yu Lin, and Ming-Hsuan Yang. 2020. Regularizing meta-learning via gradient dropout. In *Proceedings of the Asian Conference on Computer Vision*.

Alex Wang, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel R Bowman. 2018. Glue: A multi-task benchmark and analysis platform for natural language understanding. *EMNLP 2018*, page 353.

Runxin Xu, Fuli Luo, Zhiyuan Zhang, Chuanqi Tan, Baobao Chang, Songfang Huang, and Fei Huang. 2021. Raise a child in large language model: Towards effective and generalizable fine-tuning. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing, EMNLP 2021, Virtual Event / Punta Cana, Dominican Republic, 7-11 November, 2021*, pages 9514–9528. Association for Computational Linguistics.

An Yang, Junyang Lin, Rui Men, Chang Zhou, Le Jiang, Xianyan Jia, Ang Wang, Jie Zhang, Jiamang Wang, Yong Li, Di Zhang, Wei Lin, Lin Qu, Jingren Zhou, and Hongxia Yang. 2021. Exploring sparse expert models and beyond. *CoRR*, abs/2105.15082.

Ping Yu, Mikel Artetxe, Myle Ott, Sam Shleifer, Hongyu Gong, Ves Stoyanov, and Xian Li. 2022. Efficient language modeling with sparse all-mlp. *arXiv preprint arXiv:2203.06850*.

Tianyi Zhang, Felix Wu, Arzoo Katiyar, Kilian Q. Weinberger, and Yoav Artzi. 2021a. Revisiting few-sample BERT fine-tuning. In *9th International Conference on Learning Representations, ICLR 2021, Virtual Event, Austria, May 3-7, 2021*. OpenReview.net.

Zhengyan Zhang, Yankai Lin, Zhiyuan Liu, Peng Li, Maosong Sun, and Jie Zhou. 2021b. Moefication: Conditional computation of transformer models for efficient inference. *arXiv preprint arXiv:2110.01786*.

Yaoming Zhu, Sidi Lu, Lei Zheng, Jiaxian Guo, Weinan Zhang, Jun Wang, and Yong Yu. 2018. Texygen: A benchmarking platform for text generation models. In *The 41st International ACM SIGIR Conference on Research & Development in Information Retrieval, SIGIR 2018, Ann Arbor, MI, USA, July 08-12, 2018*, pages 1097–1100. ACM.

Barret Zoph, Irwan Bello, Sameer Kumar, Nan Du, Yanping Huang, Jeff Dean, Noam Shazeer, and William Fedus. 2022. Designing effective sparse expert models. *arXiv preprint arXiv:2202.08906*.

Simiao Zuo, Xiaodong Liu, Jian Jiao, Young Jin Kim, Hany Hassan, Ruofei Zhang, Tuo Zhao, and Jianfeng Gao. 2021. Taming sparsely activated transformer with stochastic experts. *CoRR*, abs/2110.04260.

Pre-trained Token-replaced Detection Model as Few-shot Learner

Zicheng Li **Shoushan Li*** **Guodong Zhou**
Natural Language Processing Lab, Soochow University, China
20205227019@stu.suda.edu.cn
{lishoushan, gdzhou}@suda.edu.cn

Abstract

Pre-trained masked language models have demonstrated remarkable ability as few-shot learners. In this paper, as an alternative, we propose a novel approach to few-shot learning with pre-trained token-replaced detection models like ELECTRA. In this approach, we reformulate a classification or a regression task as a token-replaced detection problem. Specifically, we first define a template and label description words for each task and put them into the input to form a natural language prompt. Then, we employ the pre-trained token-replaced detection model to predict which label description word is the most original (i.e., least replaced) among all label description words in the prompt. A systematic evaluation on 16 datasets demonstrates that our approach outperforms few-shot learners with pre-trained masked language models in both one-sentence and two-sentence learning tasks. [1]

1 Introduction

Few-shot learning aims to learn models with a few examples and the learned models generalize well from very limited examples like humans. Recently, few-shot learning has become an important and interesting research field of intelligence (Lake et al., 2015; Yogatama et al., 2019). Compared to data-rich supervised learning, few-shot learning greatly overcomes the expensive data annotation challenge in reality.

Some large pre-trained language models such as GPT-3 (Brown et al., 2020) have achieved remarkable few-shot performance by reformulating tasks as language model problems. However, its hundreds of billions of parameters deter researchers and practitioners from applying it widely. To tackle this, a new paradigm, equipping smaller masked language models (Devlin et al., 2018) with few-shot capabilities (Schick and Schütze, 2020a,b; Gao et al., 2021) has been explored, wherein downstream tasks are treated as cloze questions. Typically, as illustrated in Figure 1(b), each input sentence is appended with a prompt phrase such as "It was [MASK]" to each input sentence, allowing the model to fill in the [MASK] by reusing the masked language model head.

Instead of masked language models, another self-supervised pre-training task called token-replaced detection has been proposed by Clark et al. (2020) and it trains a model named ELECTRA to distinguish whether each token is replaced by a generated sample or not. One major advantage of token-replaced detection pre-training modeling is that it is more computationally efficient than masked language modeling. Moreover, their research demonstrates that given the same model size, pre-trained token-replaced detection models achieve substantially better performance than the pre-trained masked language model such as BERT (Devlin et al., 2018) and RoBERTa (Liu et al., 2019) in many downstream tasks.

In this paper, inspired by the above unique effectiveness of the pre-trained token-replaced detection model, we propose a new approach, pre-trained token-replaced detection models as few-shot learners, aiming to further improve the few-shot learning performances. The key idea of our approach is to reformulate downstream tasks as token-replaced detection problems. Specifically, we first define a template and label description words which will be used to convert the input sentence into a prompted text. Then, we directly insert the template and all label description words into the sentence to form a prompt that might be an ungrammatical sentence. The motivation of this operation is to make our inputs similar to those in the data for training ELECTRA, having some replaced tokens. Lastly, we use a pre-trained token-replaced detection model

*Corresponding author
[1]Our code is available at https://github.com/cjfarmer/TRD_FSL

Proceedings of the 29th International Conference on Computational Linguistics, pages 3274–3284
October 12–17, 2022.

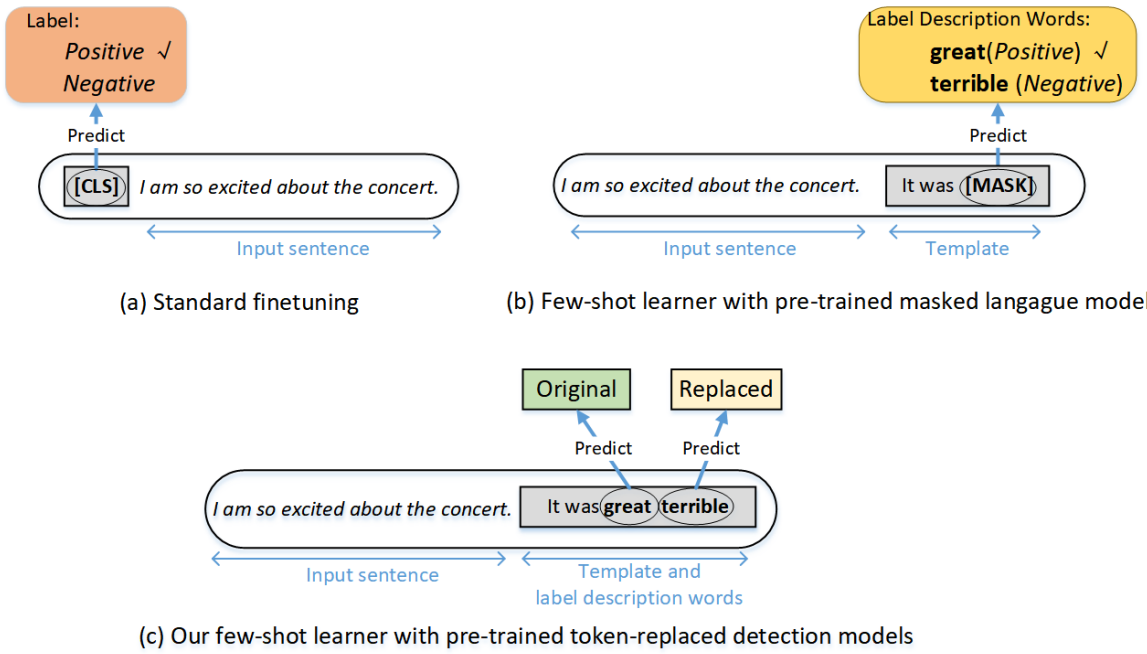

Figure 1: Different approaches of applying pre-trained models to sentiment classification.

to distinguish which label description word is the most original (i.e., least replaced) among all label description words. For instance, as illustrated in 1(c), when performing a sentiment classification task, the input sentence "*I am so excited about the concert.*" is converted into a new one "*I am so excited about the concert.* It was **great terrible**" where "**great**" and "**terrible**" are two label description words for the two sentimental categories: *positive* and *negative*. Consequently, the pre-trained token-replaced detection model may predict the label description word "**great**" is more original (i.e., less replaced) than "**terrible**", which indicates that the input belongs to a *positive* category. Compared to few-shot learners with pre-trained masked language models, in general, there are two major differences as follows. First, the designed phrases of few-shot learners with pre-trained masked don't contain any label description word. However, in our approach, we put them in prompts directly, which is easier to understand. Second, few-shot learners with pre-trained masked language models predict which label description word is the most appropriate to fill in [MASK], but our approach predicts which label description word is the most original (i.e., least replaced).

To evaluate the few-shot capacity of our approach, we use both ELECTRA-Base and ELECTRA-Large as pre-trained token-replaced detection models to perform few-shot learning in our approach and conduct experiments in a wide variety of both one-sentence and two-sentence tasks. Empirical studies demonstrate that our approach outperforms few-shot learners with pre-trained masked language models.

The contributions of this study are as follows:

- We propose a new approach for few-shot learning, which is simple and effective. To the best of our knowledge, few-shot learners with pre-trained token-replaced detection models is a novel branch of research that has not been explored in few-shot learning studies.

- A systematic evaluation of 16 popular datasets demonstrates that when given only a small number of labeled samples per class, our approach outperforms few-shot learners with pre-trained masked language models on most of these tasks.

The remainder of this paper is organized as follows. Section 2 overviews related studies about few-shot learning approaches and pre-trained token-replaced detection models. Section 3 proposes our few-shot learner with a pre-trained token-replaced detection model in detail. Section 4 presents the experimental results and analysis. Finally, Section 5 discusses the conclusions and future work.

2 Related Work

2.1 Prompt-based few-shot learning

Few-shot learning with language model prompting has arisen with the introduction of GPT-3 (Brown et al., 2020), which adds a task description (prompt) with a training example demonstration to make the language model a few-shot learner. GPT-3's naive "in-context learning" paradigms have been applied to various tasks such as text classification (Min et al., 2021; Lu et al., 2021), question answering (Liu et al., 2021a), and information extraction (Zhao et al., 2021), which shows that a large pre-trained language model can achieve remarkable performance with only a few annotated samples. However, GPT-3's dependence on gigantic pre-trained language models narrows its scope of real applications.

Instead of using a gigantic pre-trained language model, Schick and Schütze (2020a,b) reformulates a natural language processing (NLP) task as a cloze-style question with smaller masked language models (Devlin et al., 2018). Their results show that it is possible to achieve few-shot performance similar to GPT-3 with much smaller language models. Due to the instability of manually designed prompts, many subsequent studies explore automatically searching the prompts, either in a discrete space (Gao et al., 2021; Jiang et al., 2020; Haviv et al., 2021; Shin et al., 2020; Ben-David et al., 2021) or in a continuous space (Qin and Eisner, 2021; Hambardzumyan et al., 2021; Han et al., 2021; Liu et al., 2021b; Zhang et al., 2022). The discrete prompt is usually designed as natural language phrases with blank to be filled while the continuous prompt is a sequence of vectors that can be updated arbitrarily during learning. For instance, LM-BFF (Gao et al., 2021) employs pre-trained mask language models and generates discrete prompts automatically. Liu et al. (2021b) propose a prompt-based approach named P-tuning, which searches prompts in the continuous space by LSTM. Zhang et al. (2022) propose another prompt-based approach named DifferentiAble pRompT (DART), which optimizes the prompt templates and the target labels differentially.

Different from all existing above few-shot learning approaches, our approach reformulates NLP tasks as token-replaced detection problems and leverages label description words in the prompt.

2.2 Token-replaced detection

The token-replaced detection pre-training task is first introduced by Clark et al. (2020). Similar to the structure of GAN (Goodfellow et al., 2014), it pre-trains a small generator to replace some tokens in an input with their plausible alternatives and then a large discriminator to distinguish whether each word has been replaced by the generator or not. The unique effectiveness of the pre-trained token-replaced detection model intrigues many studies to apply it in many NLP tasks, such as fact verification (Naseer et al., 2021), question answering (Alrowili and Shanker, 2021; Yamada et al., 2021), grammatical error detection (Yuan et al., 2021), emotional classification (Zhang et al., 2021; Guven, 2021), and medication mention detection (Lee et al., 2020). There are also some other studies that upgrade or extend the token-replaced detection pre-training mechanism. For instance, Meng et al. (2021) jointly train multiple generators of different sizes to provide training signals at various levels of difficulty. Futami et al. (2021) transfer the mechanism to visual pre-training and Fang et al. (2022) propose an extended version of ELECTRA for speech recognition.

Different from all the above studies, to the best of our knowledge, this paper is the first study to apply pre-trained token-replaced detection models to few-shot learning.

3 Our approach

A pre-trained token-replaced detection model like ELECTRA (Clark et al., 2020) trains a discriminator $\mathcal{D}$ that detects whether a token x_t in an input token sequence $x = [x_1, \ldots, x_t, \ldots, x_n]$ is an original or replaced one. Suppose that the output from the discriminator is $y = [y_1, \ldots, y_t, \ldots, y_n]$. Then, $y_t = 0$ (or 1) indicates that x_t (at the position t) is an original (or a replaced) token. Specifically, in ELECTRA (Clark et al., 2020), the discriminator is trained together with a masked language modeling generator which is used to generate replaced tokens in a token sequence. Finally, the discriminator performs the prediction with a sigmoid output layer, i.e.,

$$P\left(y_t \mid x_t\right) = \text{sigmoid}\left(w^T h_D\left(x_t\right)\right) \quad (1)$$

where $h_D\left(x_t\right)$ is the encoder function in the discriminator $\mathcal{D}$.

3.1 Few-shot Classification

3.1.1 Few-shot Fine-tuning Phase

Suppose that the downstream classification task is a one-sentence classification problem and it has k labels with label space Y where $|Y| = k$. For the i-th category, we hand-craft a label description word $\mathbf{LABEL(i)}$. Then an input x is rewritten as a prompt as follows:

$$x_{\text{prompt}} = x \text{ It was } \mathbf{LABEL(1)} \ldots \mathbf{LABEL(i)} \ldots \mathbf{LABEL(k)} \quad (2)$$

When the downstream classification task is a two-sentence classification problem, the input (x_1, x_2) is rewritten as a prompt as follows:

$$x_{\text{prompt}} = < x_1 > ? \mathbf{LABEL(1)} \ldots \mathbf{LABEL(i)} \ldots \mathbf{LABEL(k)}, < x_2 > \quad (3)$$

Suppose that this sample belongs to the i-th label and the positions of the label description words are $[t_1, \ldots, t_i, \ldots, t_k]$. Thus, the output of the i-th description word y_{t_i} is set to be 0 and this label description word is considered as original. In contrast, the outputs of all the other label description words are set to be 1 and these description words are considered as replaced. Note that the outputs of all tokens beyond label description words are set to be 0. Formally, the output of the whole prompt is obtained as follows:

$$y_{\text{prompt}} = [\ldots, y_{t_1-1} = 0, y_{t_1} = 1, \ldots, y_{t_i} = 0, \ldots, y_{t_k} = 1, y_{t_k+1} = 0, \ldots] \quad (4)$$

For instance, in a 5-category sentiment classification task, an input x ="*This is one of his best films.*" could be rewritten as x_{prompt} = "*This is one of his best films.* It was **great good okay bad terrible**" where "**great**", "**good**", "**okay**", "**bad**", and "**terrible**" are used as the label description words for the *very positive*, *positive*, *neutral*, *negative*, and *very negative* category. The output of x_{prompt} becomes $y_{\text{prompt}} = [\ldots, 0, 0, 1, 1, 1, 1]$.

All prompt samples, together with their labels are used to update the parameters in the discriminator $\mathcal{D}$ of the pre-trained token-replaced detection model. Specifically, following the original progress of pre-training a token replaced model, we train the discriminator $\mathcal{D}$ by minimizing the binary cross entropy loss. It is important to note that our approach reuses the pre-trained weights w^T in the formula (1) and does not use any other new parameters.

3.1.2 Testing Phase

In the testing phase, a testing sample is rewritten as a prompt according to formula (2) or (3) and the labels of all label description words in this prompt is predicted with the following formula, i.e.,

$$P(y \mid \mathbf{LABEL(i)}) = \text{sigmoid}\left(w^T h_D(\mathbf{LABEL(i)})\right) \quad (5)$$

Then, the real label of the sample, i.e., l_{test}, is determined by the following formula, i.e.,

$$l_{\text{test}} = \underset{i}{\text{argmax}}(P(y = 0 \mid \mathbf{LABEL(i)})) \quad (6)$$

3.2 Few-shot Regression

3.2.1 Few-shot Fine-tuning Phase

Suppose that the downstream task is a regression problem and it has label space Y where Y is a bounded interval $[v_l, v_u]$. Following Gao et al. (2021), we reformulate the problem as a "binary classification"—predicting the probabilities of belonging to two opposing poles, $\{c_l, c_u\}$ with values v_l and v_u respectively.

Then, a few-shot regression problem can be handled as a few-shot classification problem that has two labels with label space $\{c_l, c_u\}$. Same as classification tasks above, we rewrite an input x as a prompt for a one-sentence regression task as follows:

$$x_{\text{prompt}} = x \text{ It was } \mathbf{LABEL}(l)\mathbf{LABEL}(u) \quad (7)$$

When the downstream task is a two-sentence regression task, we rewrite an input x as a prompt as follows:

$$x_{\text{prompt}} = < x_1 > \mathbf{LABEL}(l)\mathbf{LABEL}(u), < x_2 > \quad (8)$$

where $\mathbf{LABEL}(l)$ and $\mathbf{LABEL}(u)$ denote the label description words for the low and upper bound categories.

Suppose that the positions of the two label description words are $[t_l, t_u]$. Then, the output of the whole prompt is obtained as follows:

$$y_{\text{prompt}} = [\ldots, y_{t_l-1} = 0, y_{t_l} = (1 - P(c_l \mid x)), y_{t_u} = (1 - P(c_u \mid x)), y_{t_u+1} = 0, \ldots] \quad (9)$$

where $P(c_l \mid x)$ and $P(c_u \mid x)$ are the posterior probabilities of x belonging to c_l and c_u and satisfy

Category	Dataset	$\|Y\|$	#Train	#Test	Type
One-sentence	SST-2	2	6,920	872	sentiment
	SST-5	5	8,544	2,210	sentiment
	MR	2	8,662	2,000	sentiment
	CR	2	1,775	2,000	sentiment
	MPQA	2	8,606	2,000	opinion polarity
	Subj	2	8,000	2.000	subjectivity
	TREC	6	5,452	500	question classification
	CoLA	2	8,551	1,042	acceptability
Two-sentence	MNLI	3	392,702	9,815	natural language inference
	MNLI-MM	3	392,702	9,832	natural language inference
	SNLI	3	549,367	9,842	natural language inference
	QNLI	2	104,743	5,463	natural language inference
	RTE	2	2,490	277	natural language inference
	MRPC	2	3,668	408	paraphrase
	QQP	2	363,846	40,431	paraphrase
	STS-B	R	5,749	1,500	sentence similarity

Table 1: The details of 16 datasets: $|Y|$: # of classes for classification tasks (Note that STS-B is a regression task over a bounded interval [0, 5]). In our few-shot experiments, we train and develop on limited examples sampled from the original training set and evaluate on the complete test set.

the equation, i.e., $P(c_l \mid x) + P(c_u \mid x) = 1$. Following Gao et al. (2021), these two probabilities could be estimated as follows:

$$P(c_l \mid x) = \frac{v_u - y}{v_u - v_l} \tag{10}$$

$$P(c_u \mid x) = \frac{y - v_l}{v_u - v_l} \tag{11}$$

For instance, in a two-sentence similarity regression task over the interval [0, 5], the label value of two sentences "*Kittens are eating food.*" and "*Kittens are eating from dishes.*" is 4.0. We use **No** and **Yes** as the label description words for the low and upper bound categories. According to formula (8-11), we construct its x_{prompt} as "*Kittens are eating food.* **No Yes**, *Kittens are eating from dishes.*" and obtain its output $y_{\text{prompt}} = [\ldots, 0, \mathbf{0.8}, \mathbf{0.2}, 0, \ldots]$.

Same as classification tasks, we also adopt binary cross entropy loss and utilize all prompt samples together with their labels to fine-tune the discriminator $\mathcal{D}$. It is important to note that our approach reuses the pre-trained weights w^T in the formula (1) and does not use any other new parameters.

3.2.2 Testing Phase

In the testing phase, a testing sample x_{test} is rewritten as a prompt according to formula (7) or (8) and the outputs of the few-shot learner is obtained with the following formula, i.e.,

$$y_l = \text{sigmoid}\left(w^T h_D(\mathbf{LABEL}(l))\right) \tag{12}$$

$$y_u = \text{sigmoid}\left(w^T h_D(\mathbf{LABEL}(u))\right) \tag{13}$$

From formula (9), we can get

$$P(c_l \mid x_{\text{test}}) = 1 - y_l \tag{14}$$

$$P(c_u \mid x_{\text{test}}) = 1 - y_u \tag{15}$$

Note that these two posterior probabilities might not satisfy the equation, i.e., $P(c_l \mid x) + P(c_u \mid x) = 1$. Therefore, we use a normalization method to update the two probabilities, i.e.,

$$P'(c_l \mid x_{\text{test}}) = \frac{P(c_l \mid x_{\text{test}})}{((P(c_l \mid x_{\text{test}}) + P(c_u \mid x_{\text{test}}))} \tag{16}$$

$$P'(c_u \mid x_{\text{test}}) = \frac{P(c_u \mid x_{\text{test}})}{((P(c_l \mid x_{\text{test}}) + P(c_u \mid x_{\text{test}}))} \tag{17}$$

Then, the regression value of the test sample, i.e., v_{test} , is obtained by using the following formula (Gao et al., 2021):

$$v_{\text{test}} = v_l \cdot P'(c_l \mid x_{\text{test}}) + v_u \cdot P'(c_u \mid x_{\text{test}}) \tag{18}$$

Task	Template	Label Space	**Label(1) . . . Label(k)**
One-sentence			
SST-2	<*S1*> It was **Label(1) . . . Label(k)**	*positive, negative*	**great, terrible**
SST-5	<*S1*> It was **Label(1) . . . Label(k)**	*very positive, positive, neutral, negative, very negative*	**great, good, okay, bad, terrible**
MR	<*S1*> It was **Label(1) . . . Label(k)**	*positive, negative*	**great, terrible**
CR	<*S1*> It was **Label(1) . . . Label(k)**	*positive, negative*	**great, terrible**
MPQA	<*S1*> It was **Label(1) . . . Label(k)**	*positive, negative*	**great, terrible**
Subj	<*S1*> This is **Label(1) . . . Label(k)**	*subjective, objective*	**subjective, objective**
TREC	**Label(1) . . . Label(k)**: <*S1*>	*abbreviation, entity, description, human, location, numeric*	**Expression, Entity, Description, Human, Location, Number**
COLA	<*S1*> This is **Label(1) . . . Label(k)**	*grammatical, not_grammatical*	**correct, incorrect**
Two-sentence			
MNLI	<*S1*> ? **Label(1) . . . Label(k)**, <S2>	*entailment, neutral, contradiction*	**Yes, Maybe, No**
MNLI-MM	<*S1*> ? **Label(1) . . . Label(k)**, <S2>	*entailment, neutral, contradiction*	**Yes, Maybe, No**
SNLI	<*S1*> ? **Label(1) . . . Label(k)**, <S2>	*entailment, neutral, contradiction*	**Yes, Maybe, No**
QNLI	<*S1*> ? **Label(1) . . . Label(k)**, <S2>	*entailment, not_entailment*	**Yes, No**
RTE	<*S1*> ? **Label(1) . . . Label(k)**, <S2>	*entailment, not_entailment*	**Yes, No**
MRPC	<*S1*> **Label(1) . . . Label(k)**, <S2>	*equivalent, not_ equivalent*	**Yes, No**
QQP	<*S1*> **Label(1) . . . Label(k)**, <S2>	*equivalent, not_ equivalent*	**Yes, No**
STS-B	<*S1*> **Label(1) . . . Label(k)**, <S2>	*[0,5]*	**Yes, No**

Table 2: Manual templates and label description words in our experiments.

4 Experiments

In this section, we compare our approach with a few-shot learning approach based on pre-trained masked language models. Furthermore, we evaluate the impact of different templates, label description words and training data scales.

4.1 Evaluation Setting

We conduct a systematic empirical study based on the datasets used in Gao et al. (2021). The experimental data contains 16 datasets from many kinds of NLP tasks such as sentiment analysis, question classification, opinion polarity, subjectivity, acceptability, natural language inference, paraphrase, and sentence similarity (Wang et al., 2018; Bowman et al., 2015). Following Gao et al. (2021), we divide these tasks into two categories, i.e., one-sentence (single sentence) input and two-sentence (sentence pair) input tasks. In addition, these tasks not only contain binary or multi-class classification but also contain regression. See the statistics of datasets in Table 1.

4.2 Evaluation protocol

Note that the results of few-shot learning experiments are very sensitive and unstable to the different splits of data and hyper-parameter setups (Dodge et al., 2020; Zhang et al., 2020), because the size of the training examples is so small. Thus, we follow the evaluation protocol of (Gao et al., 2021) by running 5 experiments with 5 different training and development splits, randomly sampled from the original training set using a fixed set of seeds, and then measuring the average results and standard deviations. Note that, following (Gao et al., 2021), we sample the same size of development set as the training set. For the hyper-parameters, we also utilize grid search to get the best hyper-parameter setup. We set the weight_decay to be 2e-3, max_length to be 256 and use AdamW optimizer with epsilon 1e-8. We change the learning rate in the set of {1e-5,2e-5,3e-5,4e-5,5e-5} and the batch size between 4 or 8. Besides, we use manual templates and label description words for each task and the details are shown in Table 2.

4.3 Main results

We use 16 samples per class for few-shot learning experiments and conduct our experiments on both base-level and large-level pre-trained model scenarios. We compare our approach with several baselines including 1) Fine-tuning: standard fine-tuning of pre-trained models; 2) P-tuning (Liu et al., 2021b): few-shot learner that searches prompts in a continuous space by LSTM; 3) LM-BFF (Gao et al., 2021): few-shot learner that employs pre-trained mask language models and discrete prompts. For a fair comparison, we use the same templates and label description words as our approach and do not use any demonstrations; 4) DART (Zhang et al., 2022): few-shot learner that optimizes the prompt templates and the target labels differentially.

One-sentence	SST-2 (acc)	SST-5 (acc)	MR (acc)	CR (acc)	MPQA (acc)	Subj (acc)	TREC (acc)	CoLA (matt)	*AVG*
Fine-tuning(RoBERTa)	77.8 (2.8)	38.5 (1.6)	70.1 (4.9)	76.7 (2.8)	70.1 (8.0)	89.7 (0.8)	81.5 (4.3)	18.9 (11.7)	*65.4*
Fine-tuning(ELECTRA)	82.8 (3.5)	41.6 (3.6)	73.9 (3.5)	82.9 (4.1)	70.7 (5.1)	**92.0 (0.5)**	78.5 (5.8)	**39.3 (3.4)**	*70.2*
P-tuning(RoBERTa)	83.3 (5.3)	43.1 (2.1)	81.7 (1.2)	86.0 (3.6)	74.0 (5.2)	89.0 (1.1)	76.9 (8.3)	-0.8 (2.5)	*66.7*
LM-BFF(RoBERTa)	87.2 (1.3)	44.5 (0.8)	83.4 (1.4)	89.1 (1.5)	81.3 (3.9)	89.3 (1.8)	77.5 (6.1)	5.3 (5.3)	*69.7*
DART(RoBERTa)	88.9 (0.5)	45.3 (1.5)	83.7 (1.0)	89.2 (1.4)	76.6 (6.3)	88.9 (2.2)	77.3 (7.2)	4.2 (5.0)	*69.3*
Ours(ELECTRA)	**91.7 (0.8)**	**49.7 (1.0)**	**86.8 (2.8)**	**90.8 (1.0)**	**84.5 (1.5)**	87.5 (1.2)	**82.2 (3.3)**	24.7 (11.8)	***74.7***
Two-sentence	MNLI (acc)	MNLI-MM (acc)	SNLI (acc)	QNLI (acc)	RTE (acc)	MRPC (f1)	QQP (f1)	*STS-B (pear)*	*AVG*
Fine-tuning(RoBERTa)	38.6 (2.5)	39.5 (2.7)	48.0 (4.7)	**63.2 (6.7)**	51.9 (1.6)	74.5 (4.4)	58.6 (6.0)	*65.2 (8.7)*	*54.9*
Fine-tuning(ELECTRA)	46.9 (3.6)	48.9 (3.8)	50.6 (2.1)	59.9 (2.3)	52.6 (2.5)	**76.9 (2.9)**	**64.1 (2.8)**	***72.4 (2.0)***	*59.0*
P-tuning(RoBERTa)	50.6 (1.1)	50.6 (1.1)	55.0 (4.3)	58.1 (3.1)	56.0 (4.2)	70.2 (2.3)	58.7 (2.8)	-	*57.0*
LM-BFF(RoBERTa)	59.1 (2.4)	60.9 (2.4)	64.3 (2.9)	61.8 (4.8)	57.9 (6.7)	72.3 (6.6)	62.7 (2.1)	*68.6 (5.7)*	*63.5*
DART(RoBERTa)	55.3 (2.4)	55.3 (2.4)	62.6 (2.6)	58.4 (4.3)	58.2 (6.0)	72.4 (2.5)	60.4 (1.7)	-	*60.4*
Ours(ELECTRA)	**59.7 (2.4)**	**61.8 (2.0)**	**68.9 (3.2)**	61.9 (2.4)	**61.5 (2.9)**	73.9 (3.9)	58.0 (3.8)	*66.6 (2.9)*	***64.0***

Table 3: Experimental results of different approaches when base pre-trained models are used.

4.3.1 RoBERTa-Base and ELECTRA-Base Results

Table 3 gives the experimental results of different prompt-based approaches to few-shot learning with a base pre-trained model, i.e., RoBERTa-Base or ELECTRA-Base. The best performance in each task is bold in the table. Note that since there is no implementation for regressions tasks in the two baseline approaches, i.e., P-tuning and DART and thus we do not reproduce their approach on STS-B which is a regression task. From this table, we discuss the results in two scenarios, i.e., one-sentence and two-sentence tasks.

In one-sentence tasks, first, fine-tuning with RoBERTa-Base performs worse than fine-tuning with ELECTRA-Base on average (65.4% vs. 70.2%), which indicates that ELECTRA-Base is a better fine-tuner even when only a few training samples are available. This result is consistent with the conclusion reported in Clark et al. (2020) when many training samples are available. Second, all prompt-based approaches greatly outperform standard fine-tuning on most tasks, which indicates that few-shot learners with either base masked language model or base token-replaced detection model are powerful in few-shot learning. One big exception is CoLA (Warstadt et al., 2019) where few-shot learning approaches perform much worse than fine-tuning approaches. This might be because the task aims to detect whether a sentence is grammatical or non-grammatical which is difficult to find suitable label description words. However, interestingly, we find that ELECTRA-Base performs much better than RoBERTa-Base in this task. Third, our approach yields excellent results and performs much better than P-tuning, LM-BFF and DART on average (74.7% vs. 66.7%, 69.7% and 69.3%), which encourages using a pre-trained token-replaced detection model for few-shot learning in one-sentence tasks.

In two-sentence tasks, first, standard fine-tuning with RoBERTa-Base still performs worse than fine-tuning with ELECTRA-Base. Second, all prompt-based approaches greatly outperform standard fine-tuning on most tasks, which once again indicates that few-shot learners with either mask language model or token-replaced detection model are powerful in few-shot learning. Third, our approach performs better than P-tuning, LM-BFF and DART, although the average improvements are quite limited (64.0% vs. 57%, 63.5% and 60.4%).

4.3.2 RoBERTa-Large and ELECTRA-Large Results

Table 4 gives the experimental results of different prompt-based approaches to few-shot learning with a large pre-trained model, i.e., RoBERTa-Large or ELECTRA-Large. The best performance in each task is bold in the table. From this table, we discuss the results in two scenarios, i.e., one-sentence and two-sentence tasks.

In one-sentence tasks, first, fine-tuning with RoBERTa-Large performs a bit better than fine-tuning with ELECTRA-Large on average (68.1% vs. 70.4%), which indicates that the choice of ELECTRA and RoBERTa might depend on the tasks when large models are used. Second, all prompt-based approaches greatly outperform standard fine-tuning on most tasks, which indicates that few-shot learners with either large masked language models or large token-replaced detection models are powerful in few-shot learning. However, CoLA is still the exception and even

One-sentence	SST-2 (acc)	SST-5 (acc)	MR (acc)	CR (acc)	MPQA (acc)	Subj (acc)	TREC (acc)	CoLA (matt)	*AVG*
Fine-tuning(RoBERTa)	81.4 (3.8)	43.9 (2.0)	76.9 (5.9)	75.8 (3.2)	72.0 (3.8)	90.8 (1.8)	**88.8 (2.1)**	**33.9 (14.3)**	*70.4*
Fine-tuning(ELECTRA)	79.9 (7.9)	41.2 (1.9)	73.0 (5.4)	75.0 (6.4)	65.3 (6.9)	**94.0 (1.0)**	82.8 (8.0)	33.4 (10.4)	*68.1*
P-tuning(RoBERTa)	89.6 (2.6)	48.0 (1.3)	85.4 (1.9)	88.7 (2.6)	76.3 (3.3)	90.9 (1.5)	86.2 (3.4)	4.0 (5.3)	*71.1*
LM-BFF(RoBERTa)	92.7 (0.9)	47.4 (2.5)	87.0 (1.2)	90.3 (1.0)	**84.7 (2.2)**	91.2 (1.1)	84.8 (5.1)	9.3 (7.3)	*73.4*
DART(RoBERTa)	91.6 (1.0)	47.4 (3.3)	85.7 (3.0)	90.3 (0.8)	66.6 (6.4)	89.9 (1.7)	84.8 (4.6)	10.0 (8.4)	*70.8*
Ours(ELECTRA)	**92.8 (0.6)**	**50.7 (2.9)**	**89.4 (0.8)**	**90.5 (2.2)**	83.2 (1.4)	92.1 (0.7)	87.2 (3.8)	16.3 (15.1)	***75.3***
Two-sentence	MNLI (acc)	MNLI-MM (acc)	SNLI (acc)	QNLI (acc)	RTE (acc)	MRPC (f1)	QQP (f1)	*STS-B (pear)*	*AVG*
Fine-tuning(RoBERTa)	45.8 (6.4)	47.8 (6.8)	48.4 (4.8)	60.2 (6.5)	54.4 (3.9)	76.6 (2.5)	60.7 (4.3)	*53.5 (8.5)*	*55.9*
Fine-tuning(ELECTRA)	54.4 (2.4)	56.7 (1.7)	58.8 (4.8)	62.9 (4.1)	53.8 (3.7)	**78.7 (3.1)**	67.2 (3.4)	***78.5 (0.5)***	*63.9*
P-tuning(RoBERTa)	59.7 (3.0)	59.7 (3.0)	71.8 (3.5)	62.5 (6.5)	61.8 (2.6)	72.7 (7.4)	64.2 (1.5)	-	*64.6*
LM-BFF(RoBERTa)	68.3 (2.3)	70.5 (1.9)	77.2 (3.7)	64.5 (4.2)	69.1 (3.6)	74.5 (5.3)	65.5 (5.3)	*71.0(7.0)*	*70.1*
DART(RoBERTa)	67.1 (2.6)	67.0 (2.5)	74.0 (4.0)	63.1 (3.0)	64.5 (5.2)	75.9 (4.7)	63.4 (4.4)	-	*67.9*
Ours(ELECTRA)	**69.2 (4.0)**	**71.0 (3.5)**	**79.3 (3.2)**	**69.0 (4.5)**	**74.2 (3.1)**	73.2 (7.5)	**68.2 (3.4)**	*74.7 (2.9)*	***72.4***

Table 4: Experimental results of different approaches when large pre-trained models are used.

worse, the performance of few-shot learning with ELECTRA-Large performs worse than ELECTRA-Base, (16.3% vs. 24.7%). This result shows that the prompting style in our few-shot learning approach seems not suitable for the task of grammatical or non-grammatical detection. Third, our approach yields performances better than P-Tuning, LM-BFF and DART, achieving 4.2%, 1.9% and 4.5% average improvements respectively.

In two-sentence tasks, first, fine-tuning with RoBERTa-Large performs much worse than fine-tuning with ELECTRA-Large (55.9% vs. 63.9%). Second, all prompt-based approaches greatly outperform standard fine-tuning on many tasks, which once again indicates that few-shot learners with either mask language model or pre-trained token-replaced detection model are powerful in few-shot learning. Third, our approach performs better than P-Tuning, LM-BFF and DART on average (72.4% vs. 64.6%, 70.1% and 67.9%).

4.4 Impact of templates and label description words

We further conduct experiments on the one-sentence task SST-2 and the two-sentence task MNLI to study the impact of different templates and label description words in our approach. Due to a large number of trials in the grid search, we use a fixed batch size 4 and learning rate 2e-5 in this part. Table 5 shows the results of the LM-BFF approach with RoBERTa-Base, the best-performed approach in all prompt-based baselines, and our approach with ELECTRA-Base in the tasks of SST-2 and MNLI. From this table, we can see that the impact of different templates and label description words for our method is similar to LM-BFF. In terms of label description words, the more semantic-related the designed label words are to the categories, the more likely to achieve stable and excellent results. For instance, in SST-2, regardless of LM-BFF or our approach, the semantic-related label description words **great/terrible** and **good/bad** always outperform the words **dog/cat** and **terrible/great** which are semantically irrelevant or even opposite with the categories *positive* and *negative*. In terms of templates, the performance is a bit sensitive to the templates, even a punctuation mark. Besides, there seems to be no general principle to design templates to optimally adapt to our approach and LM-BFF. For instance, In MNLI, LM-BFF obtains the best performance with the template *"<S1>.* **Label(1) ... Label(k),** *<S2>"*, while our approach obtains the best performance with the template *"<S1>?* **Label(1) ... Label(k),** *<S2>"*.

4.5 Impact of training data scales

We further conduct experiments on the one-sentence task SST-2 and the two-sentence task MNLI to study the impact of the numbers of labeled instances in our approach. In this part, we also use a fixed batch size 4 and learning rate 2e-5. Figure 2 shows the trends of the LM-BFF approach, the best-performed approach in all prompt-based baselines, and our approach when using different numbers of labeled instances. From this figure, we can see that our approach outperforms LM-BFF in different numbers of labeled instances in the one-sentence task SST-2. In the two-sentence task MNLI, our approach performs similarly to LM-BFF when the numbers of labeled instances are less than 64. But our approach outperforms LM-BFF when the numbers of labeled instances are

Task	Template	**Label(1) . . . Label(k)**	LM-BFF (acc)	Our approach (acc)
SST-2 (*positive/negative*)	<*S1*> It was **Label(1) . . . Label(k)**	**great, terrible**	88.6 (1.3)	91.4 (1.6)
		good, bad	88.9 (0.6)	91.0 (2.0)
		dog, cat	85.2 (2.0)	79.6 (7.3)
		terrible, great	82.4 (3.3)	89.2 (1.9)
	Label(1) . . . Label(k) : <*S1*>	**great, terrible**	85.6 (3.0)	91.1 (1.2)
		good, bad	87.5 (0.4)	90.8 (0.7)
		dog, cat	80.1 (3.5)	69.7 (8.2)
		terrible, great	67.4 (3.5)	76.4 (9.6)
MNLI (*entailment/neutral /contradiction*)	<*S1*>? **Label(1) . . . Label(k)**, <*S2*>	**Yes, Maybe, No**	58.3 (2.4)	58.8 (2.5)
	<*S2*>. **Label(1) . . . Label(k)**, <*S1*>		58.7 (1.3)	57.6 (2.5)
	<*S1*> **Label(1) . . . Label(k)** <*S2*>		56.4 (1.8)	53.6 (2.2)
	<*S1*>. **Label(1) . . . Label(k)**, this is good, <*S2*>		54.0 (2.4)	55.8 (3.5)

Table 5: The impact of different templates and label description words.

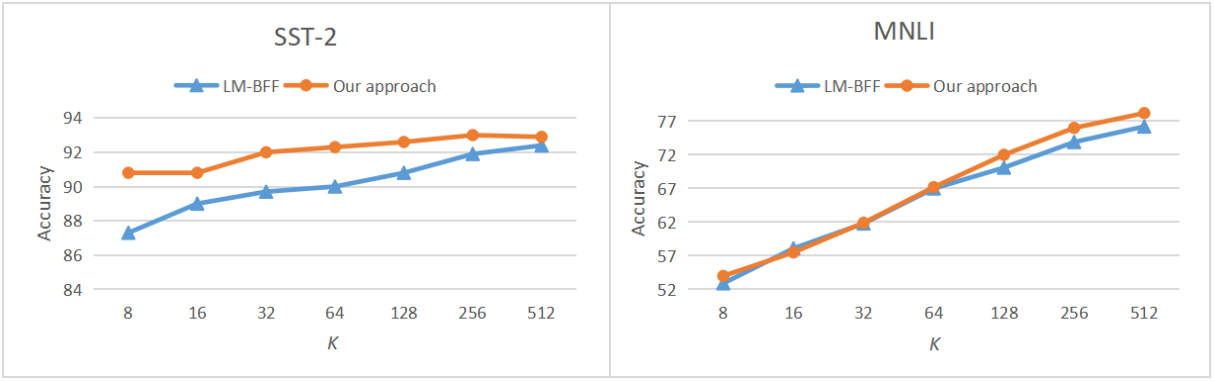

Figure 2: LM-BFF vs. our approach when using different numbers of labeled instances (*K*: # of labeled instances per class).

among [128, 512].

5 Conclusion and Future Work

In this paper, we propose a novel few-shot learning approach with pre-trained token-replaced detection models, which transforms traditional classification and regression tasks into token-replaced detection problems. Empirical studies on 16 NLP datasets demonstrate that, in both one-sentence and two-sentence learning tasks, our approach generally achieves better performances in the few-shot scenario when compared to the masked language model-based few-shot learner. These results highlight that our approach is a comprehensive alternative for few-shot learning.

In the future, we would like to explore the following directions. First, we notice that in some tasks like CoLA, standard fine-tuning is also a strong baseline and even performs much better than few-shot learners based on either a masked language model or a token-replaced detection model. Thus, it is interesting to combine [CLS] output vector, i.e., the standard fine-tuning style, with the prompt-ing style, to further improve the few-shot learning performance. Second, we would like to apply our approach to some other NLP tasks, such as multi-label text classification and sequence labeling tasks like named entity recognition.

Acknowledgement

This work was supported by a NSFC grant (No.62076176). We also acknowledge reviewers for their valuable suggestions.

References

Sultan Alrowili and K Shanker. 2021. Large biomedical question answering models with albert and electra. *CLEF (Working Notes)*.

Eyal Ben-David, Nadav Oved, and Roi Reichart. 2021. Pada: A prompt-based autoregressive approach for adaptation to unseen domains. *arXiv preprint arXiv:2102.12206*.

Samuel R Bowman, Gabor Angeli, Christopher Potts, and Christopher D Manning. 2015. A large annotated corpus for learning natural language inference. *arXiv preprint arXiv:1508.05326*.

Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. 2020. Language models are few-shot learners. *Advances in neural information processing systems*, 33:1877–1901.

Kevin Clark, Minh-Thang Luong, Quoc V Le, and Christopher D Manning. 2020. Electra: Pre-training text encoders as discriminators rather than generators. *arXiv preprint arXiv:2003.10555*.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*.

Jesse Dodge, Gabriel Ilharco, Roy Schwartz, Ali Farhadi, Hannaneh Hajishirzi, and Noah Smith. 2020. Fine-tuning pretrained language models: Weight initializations, data orders, and early stopping. *arXiv preprint arXiv:2002.06305*.

Yuxin Fang, Li Dong, Hangbo Bao, Xinggang Wang, and Furu Wei. 2022. Corrupted image modeling for self-supervised visual pre-training. *arXiv preprint arXiv:2202.03382*.

Hayato Futami, Hirofumi Inaguma, Masato Mimura, Shinsuke Sakai, and Tatsuya Kawahara. 2021. Asr rescoring and confidence estimation with electra. *arXiv preprint arXiv:2110.01857*.

Tianyu Gao, Adam Fisch, and Danqi Chen. 2021. Making pre-trained language models better few-shot learners. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 3816–3830, Online. Association for Computational Linguistics.

Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. 2014. Generative adversarial nets. *Advances in neural information processing systems*, 27.

Zekeriya Anil Guven. 2021. The effect of bert, electra and albert language models on sentiment analysis for turkish product reviews. In *2021 6th International Conference on Computer Science and Engineering (UBMK)*, pages 629–632. IEEE.

Karen Hambardzumyan, Hrant Khachatrian, and Jonathan May. 2021. Warp: Word-level adversarial reprogramming. *arXiv preprint arXiv:2101.00121*.

Xu Han, Weilin Zhao, Ning Ding, Zhiyuan Liu, and Maosong Sun. 2021. Ptr: Prompt tuning with rules for text classification. *arXiv preprint arXiv:2105.11259*.

Adi Haviv, Jonathan Berant, and Amir Globerson. 2021. Bertese: Learning to speak to bert. *arXiv preprint arXiv:2103.05327*.

Zhengbao Jiang, Frank F Xu, Jun Araki, and Graham Neubig. 2020. How can we know what language models know? *Transactions of the Association for Computational Linguistics*, 8:423–438.

Brenden M Lake, Ruslan Salakhutdinov, and Joshua B Tenenbaum. 2015. Human-level concept learning through probabilistic program induction. *Science*, 350(6266):1332–1338.

Lung-Hao Lee, Po-Han Chen, Hao-Chuan Kao, Ting-Chun Hung, Po-Lei Lee, and Kuo-Kai Shyu. 2020. Medication mention detection in tweets using electra transformers and decision trees. In *Proceedings of the Fifth Social Media Mining for Health Applications Workshop & Shared Task*, pages 131–133.

Jiachang Liu, Dinghan Shen, Yizhe Zhang, Bill Dolan, Lawrence Carin, and Weizhu Chen. 2021a. What makes good in-context examples for gpt-3? *arXiv preprint arXiv:2101.06804*.

Xiao Liu, Yanan Zheng, Zhengxiao Du, Ming Ding, Yujie Qian, Zhilin Yang, and Jie Tang. 2021b. Gpt understands, too. *arXiv preprint arXiv:2103.10385*.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Roberta: A robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692*.

Yao Lu, Max Bartolo, Alastair Moore, Sebastian Riedel, and Pontus Stenetorp. 2021. Fantastically ordered prompts and where to find them: Overcoming few-shot prompt order sensitivity. *arXiv preprint arXiv:2104.08786*.

Yu Meng, Chenyan Xiong, Payal Bajaj, Paul N Bennett, Jiawei Han, Xia Song, et al. 2021. Pretraining text encoders with adversarial mixture of training signal generators. In *International Conference on Learning Representations*.

Sewon Min, Mike Lewis, Hannaneh Hajishirzi, and Luke Zettlemoyer. 2021. Noisy channel language model prompting for few-shot text classification. *arXiv preprint arXiv:2108.04106*.

Muchammad Naseer, Muhamad Asvial, and Riri Fitri Sari. 2021. An empirical comparison of bert, roberta, and electra for fact verification. In *2021 International Conference on Artificial Intelligence in Information and Communication (ICAIIC)*, pages 241–246. IEEE.

Guanghui Qin and Jason Eisner. 2021. Learning how to ask: Querying lms with mixtures of soft prompts. *arXiv preprint arXiv:2104.06599*.

Timo Schick and Hinrich Schütze. 2020a. Exploiting cloze questions for few shot text classification and natural language inference. *arXiv preprint arXiv:2001.07676*.

Timo Schick and Hinrich Schütze. 2020b. It's not just size that matters: Small language models are also few-shot learners. *arXiv preprint arXiv:2009.07118*.

Taylor Shin, Yasaman Razeghi, Robert L Logan IV, Eric Wallace, and Sameer Singh. 2020. Autoprompt: Eliciting knowledge from language models with automatically generated prompts. *arXiv preprint arXiv:2010.15980*.

Alex Wang, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel R Bowman. 2018. Glue: A multi-task benchmark and analysis platform for natural language understanding. *arXiv preprint arXiv:1804.07461*.

Alex Warstadt, Amanpreet Singh, and Samuel R Bowman. 2019. Neural network acceptability judgments. *Transactions of the Association for Computational Linguistics*, 7:625–641.

Ikuya Yamada, Akari Asai, and Hannaneh Hajishirzi. 2021. Efficient passage retrieval with hashing for open-domain question answering. *arXiv preprint arXiv:2106.00882*.

Dani Yogatama, Cyprien de Masson d'Autume, Jerome Connor, Tomas Kocisky, Mike Chrzanowski, Lingpeng Kong, Angeliki Lazaridou, Wang Ling, Lei Yu, Chris Dyer, et al. 2019. Learning and evaluating general linguistic intelligence. *arXiv preprint arXiv:1901.11373*.

Zheng Yuan, Shiva Taslimipoor, Christopher Davis, and Christopher Bryant. 2021. Multi-class grammatical error detection for correction: A tale of two systems. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 8722–8736.

Ningyu Zhang, Luoqiu Li, Xiang Chen, Shumin Deng, Zhen Bi, Chuanqi Tan, Fei Huang, and Huajun Chen. 2022. Differentiable prompt makes pre-trained language models better few-shot learners. In *International Conference on Learning Representations*.

Shunxiang Zhang, Hongbin Yu, and Guangli Zhu. 2021. An emotional classification method of chinese short comment text based on electra. *Connection Science*, pages 1–20.

Tianyi Zhang, Felix Wu, Arzoo Katiyar, Kilian Q Weinberger, and Yoav Artzi. 2020. Revisiting few-sample bert fine-tuning. *arXiv preprint arXiv:2006.05987*.

Zihao Zhao, Eric Wallace, Shi Feng, Dan Klein, and Sameer Singh. 2021. Calibrate before use: Improving few-shot performance of language models. In *International Conference on Machine Learning*, pages 12697–12706. PMLR.

Evaluating Diversity of Multiword Expressions in Annotated Text

Adam Lion-Bouton[1], Yağmur Öztürk[2],
Agata Savary[2], Jean-Yves Antoine[1]
University of Tours - LIFAT[1], Paris-Saclay University, CNRS - LISN[2],
{lion.adam.otman, yaamurozturk}@gmail.com
agata.savary@universite-paris-saclay.fr
jean-yves.antoine@univ-tours.fr

Abstract

Diversity can be decomposed into three distinct concepts, namely: variety, balance and disparity. This paper borrows from the extensive formalization and measures of diversity developed in ecology in order to evaluate the variety and balance of multiword expression annotation produced by automatic annotation systems. The measures of richness, normalized richness, and two variations of Hill's evenness are considered in this paper. We observe how these measures behave against increasingly smaller samples of gold annotations of multiword expressions and use their comportment to validate or invalidate their pertinence for multiword expressions in annotated texts. We apply the validated measures to annotations in 14 languages produced by systems during the PARSEME shared task on automatic identification of multiword expressions and on the gold versions of the corpora. We also explore the limits of such evaluation by studying the impact of lemmatization errors in the Turkish corpus used in the shared task.

1 Introduction

Diversity of naturally occurring phenomena and artefacts is a desirable property of many environments and systems. It has been modelled and measured in many domains, including linguistics but has rarely been formalized with respect to particular linguistic phenomena within one language. This paper addresses diversity of one particular phenomenon: *multiword expressions* (MWEs), which are combinations of words, such as ***out of the blue*** or ***pay** a **visit***, exhibiting idiosyncratic properties at lexical, morphological, syntactic and/or semantic level (Baldwin and Kim, 2010). Automatic annotation of multiword expression occurrences in texts – henceforth referred to as MWE *identification* following nomenclature from Constant et al. (2017) – has been the focus of many works, among which the PARSEME shared tasks (Savary et al., 2017; Ramisch et al., 2018, 2020). During these tasks, performances of participating systems were evaluated on the precision, recall, and F1-score of their annotations.

In order to get a better understanding of how participating MWE identifiers behave, performances were also measured for specific subtasks such as the annotation of light verb construction occurrences (***pay** a **visit**, **give** a **lecture***), or the annotation of MWEs that were not seen during training. Such analysis of the resulting annotation stems from a need to make sure that the systems getting the best scores are not simply performing well on a few easier, more rewarding subtasks while ignoring others. Thus, these evaluation scenarios implicitly address some aspects of data and system diversity.

In this paper we follow the same objectives but we address diversity explicitly and formally. We suggest that diversity measures could later be used to put performance measures in perspective, by favoring NLP tools which cover diverse types and not only easy and repetitive cases.

The paper is organized as follows. We first present related work in estimating diversity in linguistics and NLP (Sec. 2). We present the data used in experiments (Sec. 3). We formalize diversity with respect to the MWE phenomenon and propose concrete measures (Sec. 4). Then we experiment with these measures to estimate diversity in MWE-annotated corpora and in annotations produced by systems participating in the PARSEME shared task 1.2 (Sec. 5) and we offer a discussion of the results (Sec. 6). Finally, we conclude and suggest perspectives for future work (Sec. 7).

2 Related Work in Linguistic Diversity

Measuring diversity, – often along its three dimensions: variety, balance and disparity (Sec. 4) – has been practiced in domains such as ecology, economy, public policy, information theory, social media, etc. (Morales et al., 2021).

Proceedings of the 29th International Conference on Computational Linguistics, pages 3285–3295
October 12–17, 2022.

Diversity is also a central notion in linguistic debates. Evans and Levinson (2009) oppose to the hypothesis of the existence of language universals (Greenberg, 1966) and suggest that linguistic research should rather use diversity as a starting point.

Quantifying linguistic diversity has been performed for decades. Greenberg (1956) measured the probability of monolingual members of a population to speak the same language. Nettle (1999), cited by Harmon and Loh (2010), modelled language diversity in terms of richness (the number of different languages in a given geographical area), phylogenetic diversity (the number of different lineages in the phylogenetic tree of languages) or structural diversity (variation among structures within languages). The Terralingua initiative[1] suggested that linguistic diversity should be regarded, from a holistic perspective, as part of biocultural diversity, and proposed indices to follow the number of world's active languages, the distribution of mother-tongue speakers among them and the rate of language extinction (Harmon and Loh, 2010). Also socio-linguistic diversity was measured in terms of the probability of using more than one common language in multilingual communication, as well as the degree of diversity of language policies (Gazzola et al., 2020).

All these measures are inter-linguistic. We are, conversely, interested in intra-linguistic measures which would represent diversity of linguistic phenomena within one language, and more precisely within NLP artefacts such as language resources and outcomes of NLP tools.

The need for diversity in training data and its impact of the performance of NLP tools has been stressed in parsing (Narayan and Cohen, 2015) or question answering (Yang et al., 2018). In these works, however, the notion of diversity was used loosely and in ad hoc manner (e.g. to describe adding noise to training data, or using multiple knowledge sources and topics) rather than formally defined.

In other works more precise diversity measures do occur. This is especially the case in natural language generation (NLG), where the so-called quality-diversity tradeoff problem is observed (systems reduce the potential diversity of their generated outputs to better fit the reference). We notice, however, that the use of diversity in NLG is not standardized. Li et al. (2016) calculate the number of distinct unigrams and bigrams in generated text and Zhang et al. (2020) use Shannon's entropy, measures which relate to richness and balance, respectively. Agirre et al. (2016) define Word Embedding Similarity, i.e. the average cosine distance between utterance embeddings. Zhu et al. (2018) use SelfBLEU, i.e. the BLEU measure applied to generated utterances rather than to the reference. Palumbo et al. (2020) mix the 2 previous measures with Jaccard, i.e. the average word overlap across utterances. These are distance measures which might be used to model disparity (if items and types are properly defined). Some of those measures are also implemented in NLG toolboxes (Li et al., 2021). Thus, diversity estimation is becoming an inherent component of NLG models.

On the other hand, complexity, a notion somewhat similar to diversity has also been measured in NLP (Brunato et al., 2016), e.g. for the sake of language learning or text simplification.

We, conversely, are interested in diversity (a notion larger than complexity) and in its promotion in language resources and tools. In this paper, we mainly focus on two of the three aspects of diversity, variety and balance. Especially the latter seems to have rarely been formalised and measured in NLP.

3 Data

Before going into details on what our diversity measures will be, we take a look at the data and the ways we use them in our experiments.

The PARSEME shared task 1.2 on automatic identification of verbal MWEs (VMWEs) was conducted on 14 languages. The corpus of each language, annotated for morpho-syntax and VMWEs, was split into three corpora, TRAIN, DEV and TEST. The TRAINs, DEVs, and blind version of TESTs corpora (with annotation for VMWE hidden) were given to the participants of the shared task, which were tasked to annotate the blind TESTs with their systems.

We will apply our diversity measures to GOLD annotations (VMWEs manually annotated in the TEST corpora) and SYSTEM annotations (VMWEs automatically annotated by systems participating in the shared task 1.2). In addition, in section 5.1 we use the French corpus Sequoia (Candito et al., 2021) which is annotated in similar fashion,

[1] https://terralingua.org/what-we-do/the-loss-of-diversity/

not only for VMWEs, but for all syntactic types of MWEs.

Diversity measures for SYSTEM annotations make sense mainly for *correctly annotated MWE occurrences*. A MWE occurrence is considered correctly annotated in a given corpus if all its tokens annotated in the corresponding GOLD corpus, and only those tokens, have been annotated as part of the same MWE occurrence. Trivially, this definition also applies to MWEs from the GOLD corpus itself, which are all considered correctly annotated.

For example, if sentence 1 below is considered the gold annotation, where ***paid visit*** and ***out of the blue*** are respectively annotated as MWE A and B, then in sentence 2 only MWE D ***out of the blue*** is correctly annotated and in sentence 3 no MWE is correctly annotated.

1. I **paid**$_{\mathbf{A}}$ them a **visit**$_{\mathbf{A}}$ **out**$_{\mathbf{B}}$ **of**$_{\mathbf{B}}$ **the**$_{\mathbf{B}}$ **blue**$_{\mathbf{B}}$
2. I **paid**$_{\mathbf{C}}$ them **a**$_{\mathbf{C}}$ **visit**$_{\mathbf{C}}$ **out**$_{\mathbf{D}}$ **of**$_{\mathbf{D}}$ **the**$_{\mathbf{D}}$ **blue**$_{\mathbf{D}}$
3. I **paid**$_{\mathbf{E}}$ them a **visit**$_{\mathbf{F}}$ **out**$_{\mathbf{F}}$ **of**$_{\mathbf{F}}$ **the**$_{\mathbf{F}}$ **blue**$_{\mathbf{F}}$

In PARSEME shared task, identification systems were also tasked to assign a category (light-verb construction, verbal idiom, inherently reflexive verb, etc.) to each annotated MWE, however, we will not look at these annotations. A MWE can therefore be considered correctly annotated by a system even if its attributed category was wrong.

In this paper, we will be interested in the diversity both of GOLD annotation and of SYSTEM annotations. In the latter case, we will consider only correctly annotated MWEs.

4 Diversity Measures

The concept of diversity is usually divided into three distinct notions: *variety*, *balance* and *disparity* (Stirling, 1998). Each of these notions goes by other names and, adding to the confusion, is sometimes referred to as diversity.

The notion of diversity relies heavily on the concepts of *items* and *types*. In ecology, *items* usually refer to specimens/individuals, and *types* refer to the species these specimens are affiliated to.

In this paper, items will refer to the correctly annotated MWE occurrences and types will refer to what PARSEME calls MWE types, meaning the multisets of lemmas of the annotated MWE occurrences.

Formally, we define items and types as follows: Let I be a set of items, T a set of types, and $\tau : I \to T$ a mapping of each item to a type.

We define items $i \in I$ as correctly annotated MWE occurrences (cf. Sec. 3), and types $t \in T$ as multisets of lemmas linked to items through the mapping τ defined bellow:

$$\tau(i) = \{\, \text{lemma}(w) \mid \forall w \in i \,\} \qquad (1)$$

Here, an item i is seen as a sequence of the annotated wordforms of the MWE occurrence, and $\text{lemma}(w)$ the function returning the lemma of a wordform.

Thus, under this definition, two correctly annotated MWE occurrences (items) are of the same type if their component words have the same lemmas, e.g. ***pay visit*** and ***visits paid*** are items of type $\{pay, visit\}$.

We note here that such a definition of a type, relying on the notion of a lemma, does not perfectly capture what we would instinctively refer to as MWE types. For instance, two senses of ***put down***: 'execute' and 'belittle' are assigned to the same type despite their different meanings. Sec. 5.3 brings to light other issues related to the data quality. Still, we consider our approximation of types good enough to be useful.

Given these definitions of items and types, we will see *variety* as a measure of the number of types in a set of items, that is to say, a measure of how many different MWE types are present in an annotation. The more MWE types an annotation has, the more varied it is. *Balance* is seen as a measure of the equilibrium of the distribution of items per type, meaning that it scores distribution based on how close the MWE types are from being equally represented. *Disparity* is seen as a measure of the distance among the types present in a set of items.

While some authors, such as Stirling (1998), advocate for a complete diversity measure – meaning that all three aspects of diversity are taken into account jointly – we take the opposite stance and aim to measure each aspect of diversity as independently of the others as possible. We consider that such an approach would be easier to interpret and sidesteps the issue of finding the right aggregation of our notions of diversity.

Henceforth, we will no longer address the notion of disparity in this paper and leave it for future work

instead. Let us only mention two main challenges behind this concept:

1. Disparity usually relies on a notion of distance which can be computed by various means and by taking any of the properties of MWE types into consideration. The choice of the precise properties calls for insightful studies. 2. The disparity of a set of types is often described as an aggregation of the pairwise distances between the types (Stirling, 1998). Thus, the choice of the aggregation used imposes some defining properties on the disparity. One such property is the monotonicity in types (Weitzman, 1992), which states that the disparity of a set of types can only increase when a new type is added to the set. Two disparities based on the same notion of distance, one with this property, the other without, will work in completely different ways, showcasing how the question of the aggregation is central to disparity.

4.1 Variety

We measure the variety of a set of items by its richness (2). The richness is a simple count of the number of types actually represented in a set of items.

Formally, we can define richness as the following (with I being a set of items):

$$|\{\,\tau(i) \mid \forall i \in I\,\}| \tag{2}$$

By its simplicity, such a measure of variety can be quite effective when it is used to compare how two MWE identification systems produce item sets of different variety from the same corpus. It, however, does not allow for a comparison of variety of item sets generated from different corpora.

In an attempt to compare variety across differently sized corpora, we normalize the richness of a set of items by its number of items.

$$\frac{|\{\,\tau(i) \mid \forall i \in I\,\}|}{|I|} \tag{3}$$

We refer to this measure as the normalized richness. Similar concepts can be found in measures such as type-token ratio (TTR) (Richards, 1987).[2] Normalized richness is quite intuitive, note however that badly performing systems (e.g. a system with only 1 correctly annotated MWE occurrence) may have an optimal value for this measure.

[2]In TTR in NLP, though, types are often defined as different surface forms, not different lemmas. Thus, ***pay visit*** and ***visits paid*** would be considered different types for TTR, conversely to normalized richness.

4.2 Balance

Measuring balance is not as easy of a matter. A great number of balance measures have been proposed, and their properties described and compared to one another (Smith and Wilson, 1996; Tuomisto, 2012). Despite these efforts, none of the measures proved more appropriate than the others.

Balance measures are computed on a probability mass function p_T. In this paper we approximate p_T by the relative frequency f_T as shown in (4). [3]

$$p_T(t) \approx f_T(t) = \frac{|\{\,i \mid \forall i \in I, \tau(i) = t\,\}|}{|I|} \tag{4}$$

We will use one of the earliest evenness measures which was designed with the goal of separating the concept of variety and balance as much as possible. Hill (1973) defines the continuum of evenness measures (5) where x and y can take any real values (as long as $x > y$):

$$E_{x,y} = \frac{N_x}{N_y} \tag{5}$$

The Hill number (6) noted N_a, also known as 'true diversity', is defined as the exponentiation of a base b (usually 2, e or 10) to the power of H_a given in (7)[4].

$$N_a = b^{H_a} = \left(\sum_{t \in T} p_T(t)^a\right)^{\frac{1}{1-a}} \tag{6}$$

(7) was defined by Rényi (1961) as a class of entropy functions of order a, with the entropy of order 1 being Shannon's entropy (8) noted H.

$$H_a = \frac{1}{1-a} \log_b \sum_{t \in T} p_T(t)^a \tag{7}$$

$$\lim_{a \to 1} H_a = H_1 = H \tag{8}$$

While just about any values of x and y would form a valid evenness $E_{x,y}$, only $E_{1,0}$ and $E_{2,1}$ seem to have been of interest to the community. Many arguments have been advanced favoring one or the other (Alatalo, 1981; Gosselin, 2006; Tuomisto, 2012). We will, for now, only mention that $E_{2,1}$ is supposedly less sensitive to sampling bias than $E_{1,0}$ (Alatalo, 1981) and investigate further upon this in section 5.1.

[3]In accordance with established practice the parameter p_T of measures in equations (5), (6), (7), and (8) is omited for brevity sake.

[4]Where the base b is also used as logarithmic base in H_a.

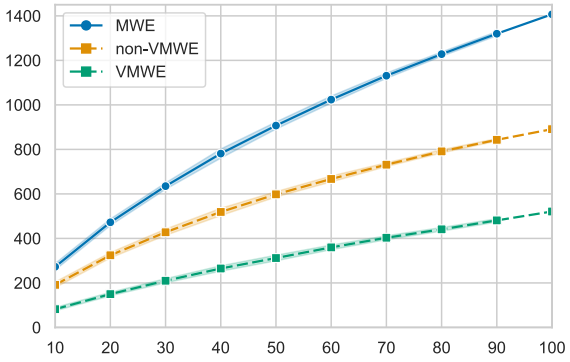

Figure 1: Richness in terms of Sequoia sample size (higher is more varied).

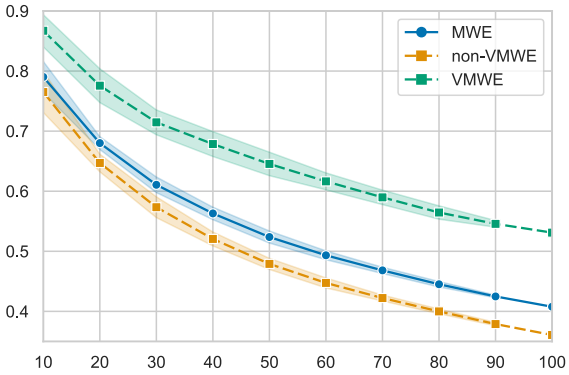

Figure 2: Normalized richness in terms of Sequoia sample size (higher is more varied).

5 Results

The previous section introduced formalization of measures for two diversity facets: variety and balance. In this section, we apply these measures on MWE-annotated corpora and MWE identifiers (Sec. 3) so as to assess their appropriateness to the MWE phenomenon (Sec. 5.1), use them as evaluation scenarios in the PARSEME shared task framework (Sec. 5.2), and show one of their limits related to data quality (Sec. 5.3).[5]

5.1 Diversity Measure Validation

We first focus on the French Sequoia corpus (Candito et al., 2021), which is one of the source corpora used for the PARSEME French corpus. It has the particularity of being annotated not only for verbal MWEs, but also for non-verbal ones (non-VMWEs).

In Figures 1, 2, 3, and 4 we apply richness (2), normalized richness (3), $E_{1,0}$ and $E_{2,1}$ (5) respectively, considering either all MWEs, only VMWEs, or non-verbal MWEs. We also take this occasion to consider how our indices behave when populations are randomly sampled. To this end, we randomly sample 10%, 20%, 30%, ..., 100% of the sentences composing the Sequoia corpus, and compute our indices on these samples (12 repeats per sample size). Results are 12-sample averages plotted in function of the size of the sample used.

Quite unsurprisingly, in Figure 1, richness increases with the size of the sample. This growth however appears to be non-linear. Non-verbal MWEs are consistently richer than verbal ones. These results seem to be quite stable as the standard deviations (marked as bands around the lines) are barely visible on this plot.

Inversely, in Figure 2, normalized richness decreases with the size of the samples. As noted by Richards (1987), this is a consequence of the quasi linear relation between the number of sentences and number of items on the one hand, and the non-linearly slower growth of the number of types on the other hand (the latter is shown in Figure 1). This shows that, conversely to our previous intuitions, normalized richness does not allow us to reliably compare diversity of corpora of different sizes. A bigger corpus will most often be disadvantaged by this measure.

Regarding evenness (Figures 3 and 4), $E_{2,1}$, which we thought to be less sensitive to sampling bias, appears to be more volatile (i.e. having higher standard deviation) than its counterpart $E_{1,0}$. $E_{1,0}$ clearly considers the distributions of VMWEs to be more balanced than the distributions of all MWEs. On the other hand, $E_{2,1}$ finds the distributions of all MWEs slightly more balanced than those of VMWEs on large samples, but the opposite on small samples.

In an attempt to determine which evenness measure best fits the nature of the data, we first proceed to visually inspect the rank-frequency distributions of all MWEs for sample sizes 10% to 100% (Figure 5), which makes us believe that our rank-frequency distributions follow a Zipfian distribution. This would explain the non-linear relation between the number of sentences and richness. Given the apparent shape of our distributions and the relative ubiquity of Zipf distribution in linguistic phenomena (Ryland Williams et al., 2015), we will work

[5]Code and data of these experiments are available at https://github.com/AdamLionB/mwe_diversity_experiment_coling_2k22.

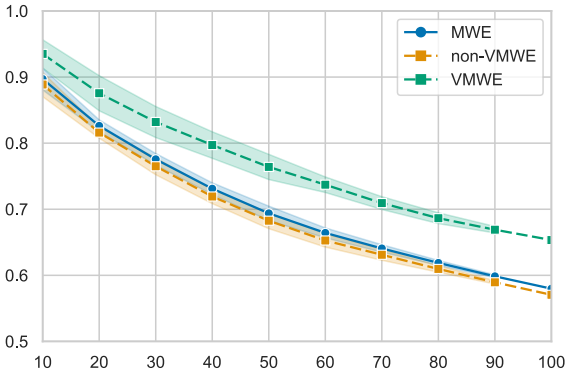

Figure 3: Evenness ($E_{1,0}$) in terms of Sequoia sample size (higher is more balanced).

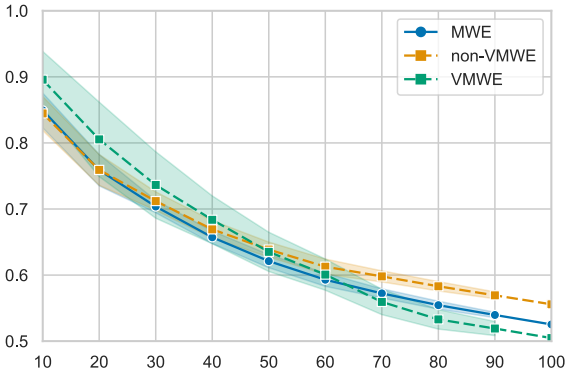

Figure 4: Evenness ($E_{2,1}$) in terms of Sequoia sample size (higher is more balanced).

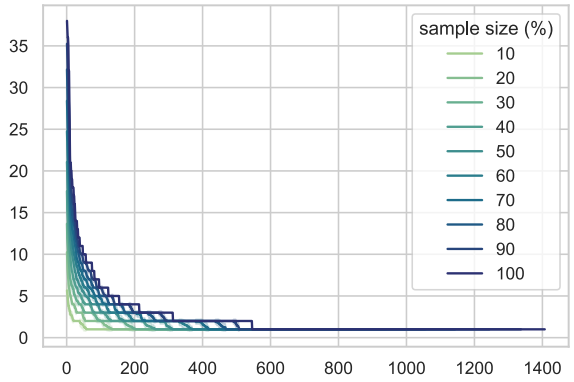

Figure 5: Rank-frequency distribution of types of all MWE according to sample size

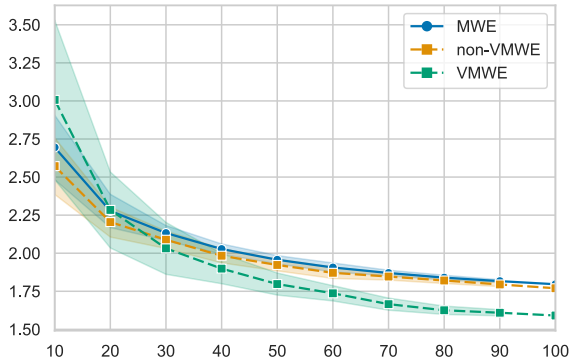

Figure 6: $\frac{1}{s}$ best fitting Sequoia samples in term of their size

under the hypothesis that the annotated MWEs follow a Zipfian distribution.

In the following, we argue that one of the parameters describing the Zipfian distribution can be used as a measure of balance. A random variable X following Zipfian distribution, noted $X \sim \text{Zipf}(s, N)$, is characterized by parameters s, N and the probability mass function (9).

$$Z_{s,N}(x) = \left(x^s \sum_{n=1}^{N} n^{-s} \right)^{-1} \quad (9)$$

Where N is the number of types in the distribution and s the exponent characterizing the curvature of the distribution. When N grows the distribution is slightly squeezed as follows: $Z_{s,N+1}(x) = Z_{s,N}(x) \cdot \left(1 + \frac{(N+1)^{-s}}{\sum_{n=1}^{N} n^{-s}}\right)^{-1}$ thus leaving room for the distribution to extend on its right while keeping its sum equal to 1. This hardly affects the shape of the distribution as the multiplication factor does not involve x. We therefore consider s (but not N) to be the parameter determining the shape of the distribution. Furthermore, the distribution is uniform when $s = 0$, it also becomes monotonically more and more skewed when s grows. We therefore argue that the value of s describing the Zipfian distribution which best fits an actual dataset constitutes an index of un-balance. When s is low the balance is high and vice versa. In other words, $\frac{1}{s}$ acts as a balance index.

Considering $\frac{1}{s}$ as a measure of balance, in Figure 6 we plot the values of $\frac{1}{s}$ found for our samples. s was optimised for the least square error disregarding overfitting, with N set to the number of types in the sample. By comparing Figure 6 to Figures 3 and 4 we find that $E_{2,1}$ and s both place all MWEs as more balanced than VMWEs for large samples and all MWEs as about as balanced as VMWEs for smaller samples. $E_{1,0}$ on the other hand always places VMWEs as more balanced than all MWEs. $E_{2,1}$ being in relative agreement with s in this study case, we will from now on use $E_{2,1}$ for the rest of this paper.

One might wonder why then use $E_{2,1}$ to measure balance and not simply s. Using s made sense here since we saw the distribution of the data and assumed that it follows a Zipfian distribution. While this hypothesis might very well hold for other GOLD corpora given the Zipfian nature of the language, there is little reason to believe that this hypothesis holds for distribution of MWEs annotated by automatic identifiers. In the general case we will therefore favor a measure which is agnostic of the distribution followed by the data.

5.2 PARSEME Shared Task Use Case

In this section we apply our diversity measures, validated in the previous section, to the GOLD corpora of the PARSEME shared task and to the correct annotations (true positives) produced by the participating systems.

In Table 1 we see that MTLB-STRUCT produces the richest annotations on 8 out of 14 corpora, and one of the 3 richest on the other 6 corpora. Travis-mono and Travis-multi also produce quite rich annotations. Seeing MTLB-STRUCT, Travis-mono and Travis-multi as producing rich annotations is coherent with these systems' F-measure performances.[6][7] On the other hand, annotations produced by Seen2Seen have notably low richness despite its relatively high performances. This is consistent with the fact that Seen2Seen was not designed to identify MWEs unseen during training, and made the bulk of its score on seen MWEs, therefore limiting the number of types it could recognize and annotate. We note that the richness of the systems' annotations on Irish (GA), Hebrew (HE) and Hindi (HI) are notably low, compared to the richness of the GOLD annotations. Training sets for Irish and Hindi were very small, with very few MWE occurrences, which most likely explains these results. Finding reasons for Hebrew requires more insight and, likely, a native knowledge of the language.

Table 2 shows the evenness scores of the systems' annotations according to $E_{2,1}$. (Systems' evenness closest to GOLD underlined.) For each language except Hebrew, at least one system has more balanced correct annotations than GOLD. Moreover, for 7 languages (EU, FR, HI, PL, RO, SV, ZH) annotations of most systems are more balanced than of GOLD. We discuss the case of systems' annotations with higher eveness than the gold in Section 6. Seen2Seen produces the most balanced annotations on 7 out of 14 languages and a more balanced annotation than the GOLD on 11 out of 14 languages. In most cases, Seen2Unseen produces annotations very slightly less balanced than Seen2seen. Note also the particularly high scores of FipsCo and TRAVIS-mono on German and Hindi, respectively, with 36% and 2% of types correctly identified (cf. tab. 1). [8]

On the whole, MTLB-STRUCT and TRAVIS-mono stand out as the systems producing the most varied annotations and Seen2Seen and Seen2Unseen as those producing the most balanced annotations. While a better understanding of our diversity measures is still needed, we hope that this constitutes a good first step toward a more systematic evaluation of diversity in MWE identification, and possibly in NLP overall.

5.3 Limits of the Diversity Measures: Case Study of Turkish

The morphosyntactic annotation of the Turkish corpus of PARSEME[9] was realized automatically using UDPipe (Straka, 2018). Afterwards, VMWE annotation was made manually according to the unified annotation guidelines of PARSEME[10]. The automatic morphosyntactic annotation was then partly modified and enhanced for better identification of MWEs. Thus, we have access to two versions of the same PARSEME corpus for this language: one with automatically produced lemmas and one with manual corrections of some lemmatization errors (Öztürk et al., 2022). We decided to examine this corpus to gain a better understanding of our diversity measures, and more precisely to study how the quality of the lemmas, central to our definition of types, influences the estimation of MWE diversity.

As an agglutinative language, Turkish is highly inflectional and derivational, which results in high surface variability in word forms. This makes it

[6] Linear correlation between the richness of systems' annotation's and precision, recall and F1-score are 0.49, 0.83 and 0.78 respectively.

[7] http://multiword.sourceforge.net/sharedtaskresults2020

[8] Linear correlation between the $E_{2,1}$ of systems' annotation's and precision, recall and F1-score are -0.47, -0.62 and -0.61 respectively.

[9] https://gitlab.com/parseme/parseme_corpus_tr

[10] https://parsemefr.lis-lab.fr/parseme-st-guidelines/1.2/

	DE	EL	EU	FR	GA	HE	HI	IT	PL	PT	RO	SV	TR	ZH
GOLD	585	682	561	712	310	443	335	638	689	815	466	495	719	551
ERMI	230	352	327	392	37	95	183	165	355	453	312	241	390	272
FipsCo	212	191		392										
HMSid				454										
MTLB-STRUCT	**400**	**472**	**387**	505	**73**	**171**	**234**	330	465	**549**	365	**345**	455	365
Seen2Seen	274	358	280	406	22	116	67	315	384	489	250	208	389	249
Seen2Unseen	290	381	333	466	48	123	170	336	416	521	259	223	435	253
TRAVIS-mono	381	55		**531**			7	**341**	**487**		**369**	307	**472**	**398**
TRAVIS-multi	348	452	368	471	14	144	176	310	449					

Table 1: Richness of GOLD and SYSTEMs annotations of PARSEME corpus

	DE	EL	EU	FR	GA	HE	HI	IT	PL	PT	RO	SV	TR	ZH
GOLD	43.40	63.95	30.30	40.42	63.88	91.08	42.48	57.05	51.38	71.67	30.89	69.28	58.59	38.12
ERMI	42.53	59.32	34.61	42.42	65.76	**90.75**	50.55	52.49	54.18	69.61	34.20	69.43	57.58	43.85
FipsCo	**85.67**	55.44		**47.98**										
HMSid				45.46										
MTLB-STRUCT	43.00	62.34	33.57	41.97	56.13	89.33	41.32	56.06	53.87	71.79	33.16	69.63	58.41	42.62
Seen2Seen	41.72	62.22	**39.05**	44.73	70.14	88.30	50.31	**58.34**	**56.73**	**72.65**	**35.54**	**74.38**	**60.06**	**45.66**
Seen2Unseen	41.68	62.14	36.94	43.52	60.90	88.20	40.89	58.03	55.84	72.02	35.24	73.81	59.61	45.59
TRAVIS-mono	42.09	**75.34**		41.76			**89.33**	56.18	54.15		33.20	69.15	56.97	42.25
TRAVIS-multi	41.52	62.46	34.28	41.91	**80.37**	90.34	64.68	55.34	54.09		35.46	72.94	57.93	45.04

Table 2: $E_{2,1}$ (%) evenness of GOLD and SYSTEMs annotations of PARSEME corpus

a good case study for diversity. The surface variability in the MWE occurrences can be observed in examples (10)–(12). All three examples contain the same VMWE with different surface forms. Next to the gloss (2nd line in each example) we report in parentheses the automatic lemmatization of the verb. In example (10) we can see the correct lemmatization ***(dava) aç***. Inadequate lemmatization can be observed in (11) and (12).

(10) **dava** **aç-tı**
dava aç-PAST (lemma: *aç*)
lawsuit open-PAST
'(someone) commenced lawsuit'

(11) **dava** **aç-ıl-abil-ir**
dava aç-PASS-POT-HAB (lemma: **açılab*)
lawsuit open-PASS-POT-HAB
'lawsuit could be commenced'

(12) **dava** **aç-ıl-acak**
dava aç-PASS-FUT (lemma: **açıla*)
lawsuit open-PASS-FUT
'lawsuit will be commenced'

All VMWE annotations were manually inspected in this corpus, to check the lemmas of all components and correct them if needed. Thus, the verb obtained the correct lemma *aç* in (11) and (12). After these corrections, the enhanced corpus was re-evaluated for richness and evenness. It was also used to retrain and re-evaluate Seen2Seen, one of the leading systems of the PARSEME shared task.

The results are presented in Table 3. In the corpus, the $E_{2,1}$ evenness becomes slightly lower, while the richness drops significantly, which signals that the number of types has decreased. This makes the Turkish corpus go down from the second to the fifth richest across all 14 languages in table 1. This outcome was expected since with correct lemmatization, items which were wrongly assigned to different types, like (10), (11) and (12), are now correctly assigned to one type.

As far as the MWEs correctly identified by Seen2Seen are concerned, evenness is almost stable but richness increases. This might be due to the fact that this system relies heavily on lemmas. It extracts the MWEs annotated in the training data and looks for co-occurrences of the same multisets of lemmas in the test corpus. If lemmas have better quality, this process is more efficient.

In brief, the main issue caused by inadequate lemmatization was that MWE items of the same type could be found in different clusters. The inadequate stripping of the suffixes resulted in more clusters (types) than there should be. These examples show the limits of diversity estimation on automatically annotated corpora. Namely, it heavily depends on the quality of the data. Here, when

lemmatization is unreliable, both the richness and the evenness artificially go up. In other words, paradoxically, bad data quality "favors" diversity in this case. This experiment also shows that high diversity of a language at a certain level (here: morphological) might make the estimation of diversity at other levels (here: MWE level) less reliable.

		GOLD	Seen2Seen
Richness	TR	**719**	389
	TR'	660	**401**
$E_{2,1}$ (%)	TR	**58.59**	60.06
	TR'	58.14	**60.08**

Table 3: Diversity measures of GOLD and Seen2Seen on the Turkish corpus, before (TR) and after (TR') lemma corrections.

6 Discussion

In Section 4 we define items as correctly annotated MWE occurrences, meaning that incorrect annotations are ignored during diversity measurements. This was motivated by the fact that variety would otherwise be artificially increased by wrong MWE types and balance affected by the likely abundance of wrong MWE types with very few occurrences. As a consequence the variety of systems' annotations cannot be higher than that of the gold.

This is however not the case for balance (as can be seen in Figure 2). This raises the question of whether it is preferable for a system annotation to be more balanced than its target (gold) annotation or to be as close to its gold annotation as possible. When training identification systems the aim is usually to approach gold annotations as close as possible. This is however already measured through scores such as precision and balance. When used in conjunction with performance measures we believe that balance should be simply interpreted as "higher is better".

In Section 4 we choose to define our types through a lexical approach of MWEs. This decision was motivated by multiple reasons: (i) it is easy to implement, (ii) non-parametric, (iii) it results in clear-cut types (non-fuzzy), (iv) it offers a good granularity (for balance measures), (v) it is very similar to the PARSEME notion of MWE types and (vi) it is a quite adequate approximation of what we would consider linguistically motivated MWE types (MWEs sharing lexemes and meaning). In future work we might use alternative definitions of types, e.g. cluster MWEs having the same syntactic structure (verb-object, subject-verb, adjective-noun), the same semantics (*to* ***kick the bucket***, *to* ***bite the dust***), or the same MWE categories (light verb construction, verbal idiom).

7 Conclusions and Future Works

In this paper, we borrowed the formalization of the notion of diversity from the literature in ecology. We focused on two out of the three main aspects of diversity, namely variety and balance. Our contribution is to apply these measures to assess intralinguistic diversity, focusing on the particular phenomenon of multiword expressions. We not only formalize variety and balance measures in this context but we also put forward methods for selecting those variants of these measure which fit the nature of the MWE phenomenon. This validation methodology is based on corpus sampling with variable sample size. As a result, we retain richness and the $E_{2,1}$ evenness as the optimal variety and balance measures for MWEs (among those studied by us). We apply these measures to the corpora and system results in the PARSEME shared task on automatic identification of MWEs. The results show that richness of the correct annotations produced by the systems is roughly consistent with their F-measure performances. However, their balance is much less correlated with more traditional measures. We also display the limits of the richness and balance measures, when calculated on automatically annotated data, due to incorrect approximation of types under improper lemmatization in a morphologically rich language.

Further investigation, particularly on the evenness, is needed as an impressive amount of evenness measures have been proposed throughout the years. Furthermore, the notion of disparity was only briefly touched upon in this paper. Disparity measures might be based on lexical overlap between types, similarity of syntactic structures or distributional semantics. These directions constitute future work.

Acknowledgements

We are grateful to Emmanuel Schang for his remarks on previous versions of this paper. We also thank anonymous reviewers for their insightful reviews.

References

Eneko Agirre, Carmen Banea, Daniel Cer, Mona Diab, Aitor Gonzalez-Agirre, Rada Mihalcea, German Rigau, and Janyce Wiebe. 2016. SemEval-2016 task 1: Semantic textual similarity, monolingual and cross-lingual evaluation. In *Proceedings of the 10th International Workshop on Semantic Evaluation (SemEval-2016)*, pages 497–511, San Diego, California. Association for Computational Linguistics.

Rauno V Alatalo. 1981. Problems in the measurement of evenness in ecology. *Oikos*, pages 199–204.

Timothy Baldwin and Su Nam Kim. 2010. Multiword expressions. In Nitin Indurkhya and Fred J. Damerau, editors, *Handbook of Natural Language Processing, Second edition*, pages 267–292. CRC Press, Boca Raton.

Dominique Brunato, Felice Dell'Orletta, Giulia Venturi, Thomas François, and Philippe Blache, editors. 2016. *Proceedings of the Workshop on Computational Linguistics for Linguistic Complexity (CL4LC)*. The COLING 2016 Organizing Committee, Osaka, Japan.

Marie Candito, Mathieu Constant, Carlos Ramisch, Agata Savary, Bruno Guillaume, Yannick Parmentier, and Silvio Cordeiro. 2021. A french corpus annotated for multiword expressions and named entities. *Journal of Language Modelling*, 8(2):415–479.

Mathieu Constant, Gülşen Eryiğit, Johanna Monti, Lonneke Van Der Plas, Carlos Ramisch, Michael Rosner, and Amalia Todirascu. 2017. Multiword expression processing: A survey. *Computational Linguistics*, 43(4):837–892.

Nicholas Evans and Stephen C. Levinson. 2009. The myth of language universals: Language diversity and its importance for cognitive science. *Behavioral and Brain Sciences*, 32(5):429–448.

Michele Gazzola, Torsten Templin, and Lisa J. McEntee-Atalianis. 2020. Measuring diversity in multilingual communication. *Social Indicators Research*, 147:545–566.

Frédéric Gosselin. 2006. An assessment of the dependence of evenness indices on species richness. *Journal of theoretical biology*, 242(3):591–597.

Joseph H. Greenberg. 1956. The measurement of linguistic diversity. *Language*, 32(1):109–115.

Joseph H. Greenberg, editor. 1966. *Universals of language*, 2nd edition. MIT Press, Cambridge, MA.

David Harmon and Jonathan Loh. 2010. The index of linguistic diversity: A new quantitative measure of trends in the status of the world's languages. *Language Documentation and Conservation*, 4.

Mark O Hill. 1973. Diversity and evenness: a unifying notation and its consequences. *Ecology*, 54(2):427–432.

Jiwei Li, Michel Galley, Chris Brockett, Jianfeng Gao, and Bill Dolan. 2016. A diversity-promoting objective function for neural conversation models. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 110–119, San Diego, California. Association for Computational Linguistics.

Junyi Li, Tianyi Tang, Gaole He, Jinhao Jiang, Xiaoxuan Hu, Puzhao Xie, Zhipeng Chen, Zhuohao Yu, Wayne Xin Zhao, and Ji-Rong Wen. 2021. TextBox: A unified, modularized, and extensible framework for text generation. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing: System Demonstrations*, pages 30–39, Online. Association for Computational Linguistics.

Pedro Ramaciotti Morales, Robin Lamarche-Perrin, Raphaël Fournier-S'Niehotta, Rémy Poulain, Lionel Tabourier, and Fabien Tarissan. 2021. Measuring diversity in heterogeneous information networks. *Theoretical Computer Science*, 859:80–115.

Shashi Narayan and Shay B. Cohen. 2015. Diversity in spectral learning for natural language parsing. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 1868–1878, Lisbon, Portugal. Association for Computational Linguistics.

Daniel Nettle. 1999. *Linguistic diversity*. Oxford University Press, Oxford.

Yağmur Öztürk, Najet Hadj Mohamed, Adam Lion-Bouton, and Agata Savary. 2022. Enhancing the parseme turkish corpus of verbal multiword expressions. In *Proceedings of The 18th Workshop on Multiword Expressions @LREC2022*, pages 100–104, Marseille, France. European Language Resources Association.

Enrico Palumbo, Andrea Mezzalira, Cristina Marco, Alessandro Manzotti, and Daniele Amberti. 2020. Semantic diversity for natural language understanding evaluation in dialog systems. In *Proceedings of the 28th International Conference on Computational Linguistics: Industry Track*, pages 44–49, Online. International Committee on Computational Linguistics.

Carlos Ramisch, Silvio Cordeiro, Agata Savary, Veronika Vincze, Verginica Barbu Mititelu, Archna Bhatia, Maja Buljan, Marie Candito, Polona Gantar, Voula Giouli, et al. 2018. Edition 1.1 of the parseme shared task on automatic identification of verbal multiword expressions. In *Proceedings of the Joint Workshop on Linguistic Annotation, Multiword Expressions and Constructions (LAW-MWE-CxG-2018)*, pages 222–240.

Carlos Ramisch, Agata Savary, Bruno Guillaume, Jakub Waszczuk, Marie Candito, Ashwini Vaidya, Verginica Barbu Mititelu, Archna Bhatia, Uxoa Iñurrieta, Voula Giouli, et al. 2020. Edition 1.2 of the

parseme shared task on semi-supervised identification of verbal multiword expressions. In *Proceedings of the Joint Workshop on Multiword Expressions and Electronic Lexicons*, pages 107–118.

Alfréd Rényi. 1961. On measures of entropy and information. In *Proceedings of the Fourth Berkeley Symposium on Mathematical Statistics and Probability, Volume 1: Contributions to the Theory of Statistics*, pages 547–561. University of California Press.

Brian Richards. 1987. Type/token ratios: What do they really tell us? *Journal of child language*, 14(2):201–209.

Jake Ryland Williams, Paul R Lessard, Suma Desu, Eric M Clark, James P Bagrow, Christopher M Danforth, and Peter Sheridan Dodds. 2015. Zipf's law holds for phrases, not words. *Scientific reports*, 5(1):1–7.

Agata Savary, Carlos Ramisch, Silvio Ricardo Cordeiro, Federico Sangati, Veronika Vincze, Behrang Qasemi Zadeh, Marie Candito, Fabienne Cap, Voula Giouli, Ivelina Stoyanova, et al. 2017. The parseme shared task on automatic identification of verbal multiword expressions. In *The 13th Workshop on Multiword Expression at EACL*, pages 31–47.

Benjamin Smith and J Bastow Wilson. 1996. A consumer's guide to evenness indices. *Oikos*, pages 70–82.

Andrew Stirling. 1998. On the economics and analysis of diversity. *Science Policy Research Unit (SPRU), Electronic Working Papers Series, Paper*, 28:1–156.

Milan Straka. 2018. UDPipe 2.0 prototype at CoNLL 2018 UD shared task. In *Proceedings of the CoNLL 2018 Shared Task: Multilingual Parsing from Raw Text to Universal Dependencies*, pages 197–207, Brussels, Belgium. Association for Computational Linguistics.

Hanna Tuomisto. 2012. An updated consumer's guide to evenness and related indices. *Oikos*, 121(8):1203–1218.

Martin L Weitzman. 1992. On diversity. *The Quarterly Journal of Economics*, 107(2):363–405.

Zhilin Yang, Peng Qi, Saizheng Zhang, Yoshua Bengio, William Cohen, Ruslan Salakhutdinov, and Christopher D. Manning. 2018. HotpotQA: A dataset for diverse, explainable multi-hop question answering. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 2369–2380, Brussels, Belgium. Association for Computational Linguistics.

Hugh Zhang, Daniel Duckworth, Daphne Ippolito, and Arvind Neelakantan. 2020. Trading off diversity and quality in natural language generation. *arXiv preprint arXiv:2004.10450*.

Yaoming Zhu, Sidi Lu, Lei Zheng, Jiaxian Guo, Weinan Zhang, Jun Wang, and Yong Yu. 2018. Texygen: A benchmarking platform for text generation models. In *The 41st International ACM SIGIR Conference on Research & Development in Information Retrieval, SIGIR 2018, Ann Arbor, MI, USA, July 08-12, 2018*, pages 1097–1100. ACM.

CausalQA: A Benchmark for Causal Question Answering

Alexander Bondarenko [⋆] **Magdalena Wolska** [†] **Stefan Heindorf** [‡] **Lukas Blübaum** [‡]
Axel-Cyrille Ngonga Ngomo [‡] **Benno Stein** [†] **Pavel Braslavski** [§,¶] **Matthias Hagen** [⋆] **Martin Potthast** [‖]

[⋆]Martin-Luther-Univeristät Halle-Wittenberg [†]Bauhaus-Universität Weimar
[‡]Paderborn University [§]Ural Federal University [¶]HSE University [‖]Leipzig University

Abstract

At least 5% of questions submitted to search engines ask about cause–effect relationships in some way. To support the development of tailored approaches that can answer such questions, we construct Webis-CausalQA-22, a benchmark corpus of 1.1 million causal questions with answers. We distinguish different types of causal questions using a novel typology derived from a data-driven, manual analysis of questions from ten large question answering (QA) datasets. Using high-precision lexical rules, we extract causal questions of each type from these datasets to create our corpus. As an initial baseline, the state-of-the-art QA model UnifiedQA achieves a ROUGE-L F_1 score of 0.48 on our new benchmark.

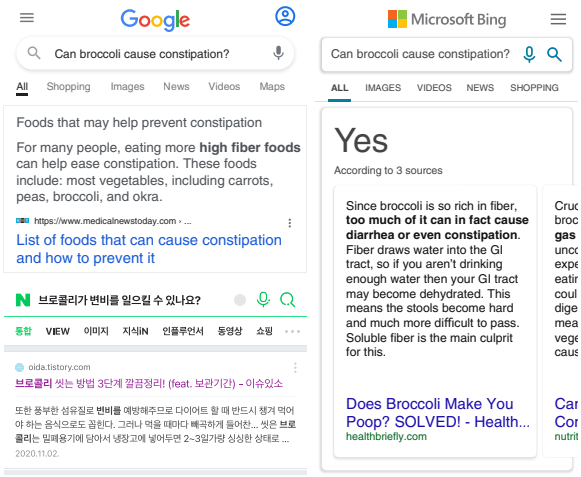

Figure 1: "Can broccoli cause constipation?" Google's and Naver's top results both disagree with that of Bing.

1 Introduction

The term "causality" usually refers to a directed relationship between events in which one is the cause of the occurrence of the other, called the effect. Many empirical studies begin with a research question about a causal relationship, ranging from "yes/no"-questions such as "Does the quality of education affect economic growth?" to open-ended questions such as "What causes depression?". But the general public also frequently asks causal questions. Figure 1 shows an example of the top Google, Bing, and Naver search result for the question "Can broccoli cause constipation?". While Bing directly answers the question in the affirmative, Google's featured snippet and Naver's first snippet claim that broccoli actually has the opposite effect.

With at most a few thousand question–answer pairs, existing causal question answering datasets are relatively small and include only one type of causal question, e.g., "yes/no"-questions (Hassanzadeh et al., 2019; Kayesh et al., 2020), "what-if"-questions (Tandon et al., 2019), "why"-questions (Verberne et al., 2006a, 2008, 2010; Lal et al., 2021), or multiple-choice questions (Gordon et al., 2012). The effectiveness of question answering (QA) systems on these benchmarks range from F_1 scores of 0.67 to 0.72. In contrast, QA systems have already performed better than humans for arbitrary questions. For instance, the F_1 score of the most effective system on the SQuAD benchmark is 0.93, while that of humans is only 0.89 (Rajpurkar et al., 2018).[1] Since neither SQuAD nor other large QA benchmarks explicitly label causal questions, the difference in effectiveness between causal and other questions remains unclear. But the inconsistent results of Bing compared to Google and Naver in Figure 1 suggest that more research is needed on answering causal questions.

We take the first steps towards a more thorough investigation of causal question answering by creating the Webis-CausalQA-22 benchmark,[2] which consists of 1.1 million questions and answers about causal relationships.[3] The resource compiles causal questions from the ten QA datasets shown in Table 1. To identify the causal questions in these

[1]https://rajpurkar.github.io/SQuAD-explorer/
[2]Leaderboard: https://causalqa.webis.de
[3]Code and data: https://github.com/webis-de/COLING-22

Proceedings of the 29th International Conference on Computational Linguistics, pages 3296–3308
October 12–17, 2022.

Table 1: Characteristics of the question answering datasets used to create Webis-CausalQA-22. We removed questions without answer (respective datasets marked by *; in total, 25,841 causal questions without answers).

Dataset	Type		Size			Length (Words)		Reference
	Question source	Answer	Questions	Causal questions		Caus. qu.	Answ.	
PAQ	Generated with BART	Term(s)	64,875,601	769,606	(1.2%)	9.6	2.7	Lewis et al. (2021)
GooAQ	Google's autocomplete	Term, Passage	5,030,530	146,286	(2.9%)	7.3	44.3	Khashabi et al. (2021)
MS MARCO QnA*	Bing query log	Passage	1,010,916	25,569	(2.5%)	6.4	17.5	Nguyen et al. (2016)
Natural Questions*	Google query log	Passage	315,203	1,208	(0.4%)	9.8	10.8	Kwiatkowski et al. (2019)
ELI5*	Reddit questions	Passage	272,634	131,033	(48.0%)	32.5	99.0	Fan et al. (2019)
SearchQA	Human-written	Term(s)	216,136	780	(0.4%)	16.8	1.8	Dunn et al. (2017)
SQuAD v.2.0*	Human-written	Term(s)	142,192	3,209	(2.3%)	10.5	6.2	Rajpurkar et al. (2016)
NewsQA*	Human-written	Term(s)	119,633	652	(0.5%)	7.2	6.1	Trischler et al. (2017)
HotpotQA	Human-written	Term(s), Passages	112,662	390	(0.4%)	21.8	3.8	Yang et al. (2018)
TriviaQA	Human-written	Term(s)	109,767	703	(0.6%)	19.4	3.1	Joshi et al. (2017)
Webis-CausalQA-22	Mixed	Mixed	72,205,274	1,079,436	(1.5%)	12.0	22.5	This paper

datasets, we manually analyzed samples and developed a two-dimensional typology of causal questions based on their semantic properties and pragmatic interpretation (Section 3). Using a set of manually created lexical rules, we extract causal questions with 80% recall at near-perfect precision (Section 4). When applied to a large sample of a query log from a commercial search engine, we also find that at least 5% of submitted queries are causal, highlighting the need for tailored technologies. As an initial baseline, we evaluate the UnifiedQA model (Khashabi et al., 2020) fine-tuned on our resource (Section 5). It achieves an average ROUGE-L F_1 score of 0.48 across datasets.

2 Related Work

We review the literature in four areas: prior typologies of causal questions, causal QA, as well as generic QA datasets and QA systems.

2.1 Causal Question Typologies

In the QA literature, causal questions are usually considered in terms of their lexical surface form and their answer type (i.e., the content of the answer). Most of the existing causal question typologies only deal with questions clearly identifiable by the question word "why". Somewhat consequently, early open-domain QA research only had a single type covering all "why"-questions (Hovy et al., 2000; Moldovan et al., 2000, 2003) before Verberne et al. (2006b) subcategorized them based on the answer type as cause (no deliberate human intention involved), motivation (human intention involved), circumstance (strict condition for the resulting event), or generic purpose (physical function of an object). For Webclopedia, Verberne et al. (2007) suggested five types: motivation, physical explanation, non-physical explanation, etymology, and nonsense. Later, Breja and Jain (2017) proposed another, rather reasoning-based, typology of causal questions: informational / factual (reasoning about a fact), historical (reasoning about the past), situational (reasons for an event at a particular time), and opinion (personal reasons).

Interestingly, all these typologies lack abstraction and do not capture general properties of causal relations. For instance, physical, non-physical, and etymology can be seen as subtypes of a class "causal explanation" that specify the nature of the explanation. The typologies also operate at different granularities, which makes comparisons difficult. For instance, Verberne et al. (2007) address specific properties of causes (physical, linguistic), whereas Breja and Jain (2017) focus on the strength of the evidence (fact vs. opinion).

In contrast, an objective, data-neutral approach to categorizing questions in general had been proposed by Lehnert (1977), including some causal types dependent on the structure of the causal dependencies. Our typology builds on Lehnert's, and we derive subtypes of causal questions in a systematic way along with their semantic and pragmatic characteristics: analytically at the semantic level and in a data-informed fashion at the pragmatic level. Moreover, our approach is not limited to causal "why"-questions as in most of the prior work, but characterizes the type of causal questions independent of their surface form.

2.2 Causal Question Answering

The related work on causal QA is rather limited. Most datasets for causal QA focus on "why"-

questions and are relatively small (Gordon et al., 2012; Hassanzadeh et al., 2019; Verberne et al., 2006a, 2008, 2010; Tandon et al., 2019; Kayesh et al., 2020; Lal et al., 2021). Usually, QA systems only achieve F_1 scores of around 0.7 on these datasets—worse than the effectiveness observed for many other question types. For instance, Ishida et al. (2018) and Iida et al. (2019) retrieve "compact" answers for "why"-questions from a web corpus using a pointer-generator network (See et al., 2017). Kayesh et al. (2019) address causal "yes/no"-questions by transfer learning, while Hassanzadeh et al. (2019) use large-scale text mining. Finally, Heindorf et al. (2020) suggest to use CauseNet, a large knowledge graph with more than 11 million cause–effect relationships extracted from ClueWeb12 web pages and Wikipedia articles. With Webis-CausalQA-22, we create a larger dataset to enable training and testing causal QA approaches on a dedicated broader benchmark.

2.3 Question Answering Datasets

Current QA research is characterized by the growing sizes of datasets (see Table 1) to improve neural QA models, and by a diversification across domains and question types (e.g., HotpotQA specifically includes comparative questions) and languages (e.g., TyDi QA features eleven languages). QA systems have meanwhile outperformed humans on Rajpurkar et al.'s (2018) SQuAD benchmark for reading comprehension. Hence, new task-specific smaller benchmarks such as CommonsenseQA (14,000 "yes/no"-questions by Talmor et al. (2019, 2021)) for common sense reasoning have been published as new challenges. On CommonsenseQA v. 2.0, for instance, Lourie et al.'s (2021) T5-based UNICORN model achieves an accuracy of 0.7, but this is still below the 0.94 of humans (Talmor et al., 2021). Out of the many available open-domain QA datasets, we selected those that are well-known enough to be mentioned in surveys (e.g., Cambazoglu et al. (2020)), contain lexically diverse question types, and have more than 100,000 QA pairs (cf. Table 1 for the selected datasets and their characteristics).

Artificial datasets. With 65 million QA pairs, PAQ (Lewis et al., 2021) is the largest of the selected datasets. Its questions were generated using the BART-base model (Lewis et al., 2020) fine-tuned on the questions, answers, and context passages from Natural Questions (Kwiatkowski et al., 2019), TriviaQA (Joshi et al., 2017), and SQuAD (Rajpurkar et al., 2016). Fine-tuning on human questions ensures some naturalness, but the answers were automatically extracted from Wikipedia. Among our selected datasets, PAQ is the only automatically generated one. We include it since the generation evaluation by Lewis et al. shows the questions to be plausible and since more than 700,000 causal questions are contained.

User-generated datasets. GooAQ (Khashabi et al., 2021), MS MARCO QnA (Nguyen et al., 2016), Natural Questions (Kwiatkowski et al., 2019), ELI5 (Fan et al., 2019), SearchQA (Dunn et al., 2017), and TriviaQA (Joshi et al., 2017) contain real-world questions submitted to search engines or posted on web fora. The GooAQ dataset contains about five million QA pairs with questions collected from Google's query auto-completion when prompted with a given question word. The answers were extracted from Google's featured snippets shown as direct answers on top of the search results. The MS MARCO QnA corpus contains about one million questions that were sampled from Bing's query logs, with long answers (text passages) extracted from web documents retrieved by Bing, and short answers (terms) written manually by crowdworkers. Similarly, the Natural Questions dataset contains more than 300,000 queries sampled from Google's search logs. Long answers and short answers were manually selected by crowdworkers from Wikipedia articles.

The about 270,000 ELI5 questions were collected from Reddit's subreddit "Explain Like I'm Five (ELI5)" where users provide simple answers to posted questions. Only answers (text passages) with at least two more up-votes than down-votes were used. The more than 215,000 questions in SearchQA and their short answers (terms) stem from Jeopardy!, while context passages were obtained by querying Google and collecting at least 40 result snippets. The more than 100,000 QA pairs in TriviaQA were scraped from various trivia and quiz websites. Each QA pair is complemented with context passages in the form of web documents from Bing's search results or from Wikipedia.

Crowdsourced datasets. The SQuAD (Rajpurkar et al., 2016), NewsQA (Trischler et al., 2017), and HotpotQA (Yang et al., 2018) datasets were exclusively constructed using crowdsourcing. SQuAD version 2.0 contains about 140,000 QA pairs written by crowdworkers who were shown paragraphs

from Wikipedia and tasked to compose up to five questions and answers about them. The about 120,000 QA pairs in NewsQA were similarly crowdsourced using CNN news articles' headlines and their summaries, but different crowdworkers wrote the questions and the answers. Lastly, HotpotQA contains about 113,000 entries with questions, answers, and supporting facts written by crowdworkers based on Wikipedia paragraphs. Designed for multi-hop QA, these questions require a system to "hop" over several supporting facts (mostly sentences) from different text passages to arrive at a short answer.

2.4 Question Answering Systems

Early question answering systems such as Baseball (Green et al., 1961) used dictionaries of attribute–value pairs to answer questions, usually in narrow domains. Recent, more sophisticated QA systems can be divided into systems based on textual data and systems based on knowledge graphs.

Text-based systems, like UnifiedQA (Khashabi et al., 2020) that we employ as a first baseline for our new benchmark, mainly use language models. Their input may just be a question but often also is a question with context like some text passage or even the whole Wikipedia (Chen et al., 2017). The actual answering process ranges from binary classification (answer selection) over span extraction (identifying answer boundaries within a text) to abstractive text summarization and generation.

Knowledge base question answering (KBQA) systems operate on graphs with a single or up to thousands of edge types (e.g., DBpedia by Auer et al. (2007)). Typically, they use manually designed templates of graph patterns to detect answers (Zheng et al., 2018; Vollmers et al., 2021), use knowledge graph embeddings (Sharp et al., 2016; Huang et al., 2019; Saxena et al., 2020), or train neural networks on knowledge graphs (Chakraborty et al., 2021). Questions are often divided by their answer type being a single graph relation (Mohammed et al., 2018), a path with multiple hops (Saxena et al., 2020), or complex answers requiring reasoning (e.g., combining information from multiple paths; Lu et al. (2019); Mitra and Baral (2016); Asai et al. (2020)).

3 A Typology of Causal Questions

While various types of causality-related questions have been previously addressed in automated question answering, there has been no attempt so far to systematize "questions about causality" as a class in the QA community. Computational processing of causal structures, however, dates back to the 1970s and the early AI research on causal dependencies between events in the context of story comprehension. Notably, Lehnert (1977) developed a computational model of question answering based on a theory of "conceptual information processing". Their QUALM system was capable of answering 13 types of questions about stories—9 types being related to causal relationships.

Following Lehnert, we define questions about causal relationships in terms of causal chains (Schank, 1975) and integrate Lehnert's causal categories into a more specific typology of causal questions. While Lehnert's definitions and categories are motivated by and directly linked to processing strategies in a story comprehension system, our typology is more generic and motivated by the semantic and pragmatic properties of causal questions. At the semantic level, we group causal question types in terms of which component of a causal chain a question addresses. Our type set combines Lehnert's causality-related categories and Verberne et al.'s categories of "why"-questions (Verberne et al., 2006b, 2007) as subtypes. At the pragmatic level, we group question types in terms of the assumed purpose of inquiry or the so-called *intent* of a question. We arrived at the pragmatic categories in a data-driven fashion by analyzing 1,000 questions (100 sampled from each of the 10 selected QA datasets; cf. Table 1). In the following sections, we first define the category *causal question* and then present the semantic and pragmatic dimensions of our typology.

3.1 The Causal Question Category

We define the category *causal question* by referring to knowledge resources required in providing an answer, specifically, to inference based on causal chains (Schank, 1975). A *causal chain* is a sequence of alternating events (or statestions) linked by relationships expressing causal dependencies between them: an event can *enable*, *result in*, be the *reason of*, *cause*, or *lead to* another event. A question is a *causal question* if answering it requires (1) identifying causal chains, (2) inference on those chains, and (3) verbalizing the causal relations involved when answering it. By this definition the question "Why is there something rather than noth-

Table 2: (a) The semantic and (b) the pragmatic dimensions of causal questions; the set of subtypes in (b) is not exhaustive, but serves to show that the top-level categories are well-motivated—considering that coherent subtypes can be identified—and to illustrate the range of domains of the requests. (c) Rules to classify causal questions in the labeled sample of 1,000 questions. Reported: precision and recall for the class of causal questions and number of matches in Webis-CausalQA-22. For the rules not present in the initial random sample, we sampled 50 random questions afterwards, manually labeled them, and calculated a precision (numbers are given in gray).

(a)

Category	Examples
Questions about an antecedent	
Cause	Why does a mosquito bite itch?
Goal	Why did Jean Valjean take care of Cosette?
Purpose	Why do gaming chairs have a race car design?
Enablement	How can FIFA be so blatantly corrupt?
Questions about a consequent	
Result	What does increasing water vapor lead to?
Questions about the causal chain	
Verification	Would hydrophobic coating affect swimming?

(b)

Intent	Examples
Solution seeking	
Problem solving	
Practical problems	Why can't I log in into Facebook?
Medical problems	Can broccoli cause constipation?
Problem prevention	
Medical problems	What to do to prevent cancer?
Societal problems	What to do to prevent global warming?
Coping with problems	
Mental coping	Why do you think about the people who are gone?
Anger	Why doesn't a director fire a stupid employee?
Knowledge seeking	
Physical world	Why do chemical reactions depend on pH?
Politics / history	How did World War II start?
Language	Why is a notebook called "notebook"?
Trivia / fun facts	What happens if you scan a mirror?
Opinion seeking	
Social issues	Why do men cheat on their wives?
Entertainment	Why is Messi not playing on the team?
Rational future outcomes	What will happen if Trump wins another election?
Irrational future outcomes	What will happen if one dreams of pregnancy?

(c)

Measure	Lexical rules							
	R1	R2	R3	R4	R5	R6	R7	R1–7
Precision	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00
Recall	0.59	0.11	0.07	0.02	0.01	–	–	0.80
Matches	505K	313K	132K	131K	10K	15K	4K	1.1M

ing?" can be interpreted as causal and eliciting a physical cause for existence, whereas the question "What is your name?" will not be considered causal even though a causal chain leading to a person being given a name may be identified; the answer "My name is Mary" does not verbalize the causal chain and an answer like "My mother named me Mary", while it may be considered as related to the question, provides irrelevant information under standard assumptions about responsiveness.

3.2 The Semantic Dimension

Our three top-level semantic categories for causal questions reflect the question's target with respect to the structure of causal chains: *questions about an antecedent* ask about events, actions, or states that in a (maybe just hypothetical) causal chain precede the ones mentioned in the question, *questions about a consequent* ask about events, actions, or states that follow the ones mentioned in the question, and *questions about the chain* ask about some property of the causal chain itself. Each of these three categories has further more specific subtypes; selected subtypes with example questions are given in Table 2a. Note that the list of subtypes is not meant to be exhaustive: we show only those types that we actually encountered in the literature or in our annotated datasets.

Cause questions ask about a general cause due to which the consequent holds; the causal dependency may be of any type: physical cause, social, psychological, etc. *Goal* questions ask about intentional motives behind an action, be it general future goals or inner motivations, whereas *purpose* questions ask about a generic purpose of the consequent, and *enablement* questions ask about the circumstances that enable / enabled the consequent. *Result* questions ask about the general effect of the antecedent, and *verification* questions ask whether a causal chain between events exists.

In this typology, Lehnert's *disjunction* is subsumed under the more general *verification* category (properties of the verified proposition possibly marked as attributes) and *expectational* is an attribute of *cause*, since the only difference between Lehnert's *cause* and *expectational* categories is that in the case of the latter, the consequent act presumably did not occur. Verberne et al.'s *motivation*, essentially a combination of Lehnert's *goal* and *circumstance*, is our *enablement* category with possibly Charniak's (1975) additional attributes.

Note that answering procedural questions (e.g., "how to . . . ") also often involves inferences based on somewhat "causal" chains. However, procedural questions usually reflect a non-causal underlying information need in the sense that they ask about

the sequential nature of a chain but not about the causal relations. Such questions can rather be considered *manner questions*, as also suggested by Hovy et al. (2002), so that we do not include them in our typology of causal questions.

3.3 The Pragmatic Dimension

At the pragmatic level, we model the inquirer's assumed motive for asking, i.e., their "visceral need" in Taylor's (1962) terminology or the "query intent" in Broder's (2002). We link the causal questions' intents to the pragmatic function of the inquiries—their "illocutionary force" (Austin, 1962). Much as recognizing the underlying function of a question affects a listener's response strategy, also in the case of web search, being able to identify a query's underlying speech act can guide the choice of what resources (e.g., what document subset) to use in a search for answers.

Our analysis of the 1,000 question sample dataset revealed three core categories of intent in causal questions: *solution seeking*, *knowledge seeking*, and *opinion seeking*. These intents, in turn, can be interpreted in terms of their illocutionary force as indirect *requests for help* (some of the questions under *solution* and *opinion*) or as genuine *requests for information* (*solution* and *knowledge*). *Solution seeking* and non-trivia/-trivial *knowledge seeking* calls for search in authoritative knowledge sources, whereas *opinion seeking* calls for search, for instance, in discussion fora or social media. Moreover, recognizing a *request for help* (falls into *coping with problems* in our typology) in a question might justify additional content in the generated response, such as advice where to seek help in case of a medical question. Subtypes of the three intent categories are exemplified in Table 2b. Again, the presented subtypes are not meant to be exhaustive, but rather to show that the top-level categories are well-motivated and to illustrate the range of possible information needs in causal questions.

3.4 Causal Questions in Web Search

Finally, to gain some insights into causal questions that people actually submit to search engines, we briefly analyze a dataset of all question-like queries submitted to Yandex in 2012; dataset created by Völske et al. (2015) and also used by Bondarenko et al. (2020). The question-like queries were extracted from the complete 2012 Yandex log by matching any of 58 hand-crafted syntactic question indicators (e.g., queries starting with "how", "what", "where", etc.). The final set contains about 1.5 billion question-like log entries with about 730 million unique questions. Applying translated versions of the seven lexical rules we use for our benchmark corpus construction (cf. Table 3), about 81.7 million causal questions are mined from the log (about 5% of the 1.5 billion question-like log entries). The most frequent causal questions are "why"-questions (50 million; frequent example: "Why can't I log in into VKontakte?") followed by "what to do if"-questions (13.1 million) and "what happens if"-questions (11 million). Interestingly, from manual spot checks of 1,000 mined "what happens if"-questions, it seems that 90% of them are about dream interpretation (e.g., "What will happen, if one dreams of pregnancy?"). This category of somewhat *fictitious* causality, raises the interesting question about how search engines or QA systems should deal with respective information needs. However, somewhat unsurprisingly, such examples are not contained in current standard QA datasets. Another manual inspection of a sample of 1,000 questions explicitly asking about causes or effects shows that most of them target causes of medical conditions or effects on health.

4 The Webis-CausalQA-22 Corpus

In this section, we describe how we extract causal questions from the ten QA datasets in Table 1 and briefly analyze or resulting new benchmark corpus.

4.1 Corpus Construction

Table 1 gives an overview of the QA datasets from which we extract causal questions to construct the Webis-CausalQA-22 benchmark. The datasets fulfill three selection criteria: (1) they contain lexically diverse questions, (2) they are well-known in the research community, and (3) they are large.

We investigate causal questions in two steps: based on prior work and based on a manual analysis of 1,000 questions randomly sampled from the QA datasets (100 from each). We asked two annotators to label whether a given question is causal or not, considering a question to be causal if the answer may only be provided as a result of causal reasoning involving entities from the question. To discover new patterns beyond more "obvious" ones like "What are the effects of X?" or "What causes Y?", we did not provide examples, but specified that the question may be asking about explicit or implicit causal relationships. They achieved an

ID	[Regular Expression] / Example
R1	`[why]` Why does mosquito bite itch?
R2	`[cause(s)?]` What causes broken blood vessels?
R3	`[how come|how did]` How did the constellation Bootes get its name?
R4	`[effect(s)?|affect(s)?]` What was the effect of the silk road on religions?
R5	`[lead(s)? to]` What does increasing water vapor lead to?
R6	`[what (will|might)? happen(s)?]∧[if|when]` What happens if we drink very hot water?
R7	`[what (to do|should be done)]∧[if|to|when]` What to do if my Xbox won't connect to the Wi-Fi?

Table 3: Lexical rules used to match causal questions in a regular expression syntax. E.g., a question matching R6 must contain 'what happens' or 'what will happen' or 'what might happen' and 'if' or 'when'.

inter-annotator agreement of Cohen's $\kappa = 0.54$ (moderate agreement). Coding differences were reconciled in a discussion with a third annotator and a total of 86 questions labeled as causal.

Based on the causal questions from our sample and based on existing question typologies (Lehnert, 1977; Graesser and Person, 1994; Graesser et al., 2008; McClure et al., 2001; Gelman, 2011; Gelman and Imbens, 2013), we hand-crafted the seven lexical rules to identify causal questions (cf. Table 3). Rules R1–R5 achieve a precision of 1.0 on our labeled sample (cf. Table 2c), while no instances matched Rules R6 and R7 (derived from prior work). We thus randomly sampled 50 questions from the QA datasets using these rules and manually checked that their precision also is 1.0.

We run these seven high-precision rules on the ten standard QA datasets and extract a total of about 1.1 million causal questions that, together with their answers and context passages (if available), form the Webis-CausalQA-22 benchmark corpus.

4.2 Corpus Analysis

Table 2c shows how many causal questions have been extracted by each of the seven lexical rules. About half of the causal questions are open-ended "why"-questions (e.g., "Why does a mosquito bite itch?"). Questions about causes (e.g., "What causes broken blood vessels?") constitute another 28% of our corpus. Interestingly, the least frequent ones are "what to do if"-questions (e.g., "What to do if my Xbox won't connect to the Wi-Fi?") that at less than 1% are by far less common than their 11% in real web search questions (cf. Section 3.4).

The context available for the question–answer pairs in our Webis-CausalQA-22 corpus depends on the source dataset and varies from Wikipedia passages (e.g., PAQ, Natural Questions, SQuAD) to search engine snippets (e.g., SearchQA) or passages from web documents (e.g., MS MARCO QnA). Also the average question and answer lengths vary widely per extraction source. While, on average, a question contains 12 words (cf. Table 1), the questions from MS MARCO QnA, for instance, are much shorter (6.4 words, Bing search) and questions from ELI5 are much longer (32.5 words, extracted from Reddit). Similarly, on average, an answer has 23 words but the answers from SearchQA are way shorter (1.8 words, human-written answers for Jeopardy! clues) while the answers from ELI5, again, are much longer (99 words, human-written answers with explanations). Besides the causal nature of the questions, also this diversity of questions and answers in our corpus poses a challenge to (causal) QA systems.

5 Evaluation

To establish a first baseline effectiveness for causal question answering on the Webis-CausalQA-22 benchmark, we report the results achieved by the state-of-the-art UnifiedQA model Khashabi et al. (2020, 2022). UnifiedQA is a text-based question answering model that has been reported by Khashabi et al. to perform well on 32 QA datasets, including SQuAD v. 2.0, where it achieved a bag-of-word-based F_1 score of 0.90. We experiment with Version 2 of the model, Checkpoint 1363200, using (1) the base model, and (2) a version fine-tuned on Webis-CausalQA-22 using the hyperparameters of Khashabi et al. (2022).[4] In a pilot study, we attempted to fine-tune a joint model on all datasets but found fine-tuning per source dataset to yield better results. Moreover, we experiment with the causal questions extracted from the original train/dev splits proposed by the authors as well as a random 90/10 train-test split of our own. The reason for the latter is that, for some datasets, by chance, only few causal questions are part of the original train/dev splits (compare the number of

[4] All experiments were conducted on an NVIDIA A100 GPU. Fine-tuning: 60K steps in general, or 6K steps to avoid overfitting on datasets containing less than 50K causal questions; AdamW optimizer (Loshchilov and Hutter, 2019); learning rate $5e^{-5}$; batch size 2.

Table 4: Effectiveness of the UnifiedQA model on causal question answering on the Webis-CausalQA-22 corpus: (a) the base model (Version 2) and a fine-tuned version on the original train/dev splits per dataset if availalbe; (b) a fine-tuned version on a random 90/10 train/test split. N: number of questions used for evaluation, P: precision, R: recall, F_1: F_1 score, EM: exact match. The star (*) indicates datasets usually evaluated using ROUGE-L.

Dataset	**Original train/dev split**											**Random 90/10 split**					
	N	Base model					Fine-tuned model					N	Fine-tuned model				
		ROUGE-L			Traditional		ROUGE-L			Traditional			ROUGE-L			Traditional	
		P	R	F_1	EM	F_1	P	R	F_1	EM	F_1		P	R	F_1	EM	F_1
PAQ	76,961	0.79	0.85	0.80	0.69	0.80	0.95	0.95	0.94	0.91	0.94	76,961	0.95	0.95	0.94	0.91	0.94
GooAQ*	33	0.29	0.04	0.06	0.00	0.07	0.14	0.11	0.12	0.00	0.15	14,629	0.17	0.15	0.15	0.00	0.19
MS MARCO QnA*	2,558	0.44	0.19	0.23	0.05	0.24	0.49	0.40	0.39	0.10	0.41	2,557	0.45	0.42	0.39	0.13	0.40
Natural Questions	71	0.14	0.05	0.06	0.01	0.07	0.34	0.37	0.33	0.18	0.34	121	0.37	0.34	0.32	0.16	0.33
ELI5*	13,104	0.26	0.04	0.06	0.00	0.08	0.16	0.09	0.10	0.00	0.12	13,104	0.16	0.09	0.10	0.00	0.12
SearchQA	117	0.20	0.22	0.20	0.15	0.20	0.63	0.64	0.62	0.53	0.62	78	0.55	0.54	0.54	0.47	0.54
SQuAD v.2.0	252	0.79	0.81	0.78	0.63	0.78	0.84	0.84	0.83	0.66	0.83	321	0.96	0.96	0.95	0.93	0.95
NewsQA	29	0.57	0.55	0.53	0.31	0.53	0.65	0.56	0.58	0.45	0.58	66	0.76	0.76	0.73	0.58	0.73
HotpotQA	35	0.49	0.39	0.40	0.14	0.40	0.60	0.55	0.53	0.26	0.54	39	0.73	0.73	0.73	0.67	0.72
TriviaQA	66	0.37	0.35	0.34	0.26	0.35	0.43	0.41	0.40	0.27	0.40	71	0.44	0.43	0.42	0.28	0.42
Macro-averaged	9,323	0.43	0.35	0.35	0.23	0.35	0.52	0.49	0.48	0.34	0.49	10,795	0.55	0.54	0.53	0.41	0.53
Micro-averaged	65,447	0.70	0.72	0.68	0.58	0.68	0.82	0.81	0.81	0.75	0.81	58,505	0.73	0.72	0.72	0.65	0.73

causal questions reported in Table 1 to the left Subcolumn N in Table 4. The original test sets are often not publicly available, but only indirectly via run submission to a leaderboard.

Effectiveness is measured using the ROUGE-L scores precision, recall, and F_1 (Lin, 2004), as well as the traditional exact match (EM) and F_1 measures. The ROUGE-L measures are based on the longest common subsequence between a predicted answer and a ground truth answer, whereas EM requires the order of all tokens to match and the traditional F_1 measure is based on an order-invariant bag-of-words representation. If a question has more than one ground truth answer, the maximum score per measure and question is taken. Effectiveness is measured both per constituent dataset of Webis-CausalQA-22, and averaged using micro- and macro-averaging across datasets.

Table 4 shows the effectiveness scores achieved by UnifiedQA. The columns "Original train/dev split" shows the effectiveness on the causal questions that we have identified in the original dev split using our lexical rules, yielding the number of causal question–answer pairs indicated in Subcolumn N.[5] We observe that UnifiedQA is most effective on PAQ across all measures, perhaps due to the large number of questions–answer pairs available and/or the fact that the models underlying PAQ and UnifiedQA have both been trained (among others, respectively) on SQuAD. For GooAQ and ELI5, the effectiveness is lowest, perhaps due to the lack of context information in these datasets. Fine-tuning UnifiedQA on the respective datasets increases its effectiveness in terms of ROUGE-L F_1 across the board. Overall, the scores of the fine-tuned models are between 0.12 and 0.62 with the exception of PAQ (0.94) and SQuAD (0.83). This generally indicates plenty of room for improvements in causal QA.

The columns "Random 90/10 split" reports the corresponding effectiveness scores of UnifiedQA for the fine-tuned model version, where fine-tuning was repeated on the different training set. Comparing the ROUGE-L F_1 scores to the fine-tuned model on the original split, we observe the largest differences for the datasets HotpotQA (from 0.53 in the original split to 0.73 in the new one), NewsQA (from 0.58 to 0.73), SQuAD (from 0.83 to 0.95), SearchQA (from 0.62 to 0.54), and GooAG (from 0.12 to 0.15). The effectiveness on HotpotQA increases because the original split used more difficult questions for the dev set than for training (Yang et al., 2018). The effectiveness on the new splits of SQuAD v. 2.0 and NewsQA increases because the UnifiedQA base model was trained on both datasets causing a leakage of training data. The effectiveness on SearchQA decreases potentially due to overfitting to the training set, or a particularly easy dev set in the original dataset by chance as the original dataset was split by time (different years for dev and test sets than for training). The effectiveness on GooAQ increases slightly because the original train/dev sets were explicitly made dissimilar by avoiding word overlaps while

[5]For PAQ and ELI5, no dedicated dev sets are available and we performed a random 90/10 split.

our random split does not. Moreover, with the new split, GooAQ is evaluated on many more questions because the original dev set contained far fewer causal questions than 10% of the whole dataset.

Overall, when comparing macro-averages across datasets, we find that fine-tuning improves the macro-averaged ROUGE-L F_1 scores from 0.35 to 0.48 on the original train/dev split, and to 0.53 on the random 90/10 split. Micro-averaging generally results in higher scores when compared to macro-averaging due to imbalanced distribution of question–answer pairs across datasets, where PAQ has the largest influence. Interestingly, when comparing the macro-averaged ROUGE-L F_1 scores of the original train/dev split with the random one, and the corresponding micro-averaged ones, the micro-averaged ones decrease from 0.81 to 0.72, while the macro-averaged ones increase as mentioned above. This is mainly caused by GooAQ having a much higher weight overall due to contributing more than 14,500 question–answer pairs, the second-largest amount following PAQ, compared to only 33 in the original train/dev split.

It is a matter of debate, which of the two splits and which of the two averages are to be preferred as a baseline. At present, we recommend using the original train/dev split (especially, if a model was trained on one of our corpus's constituent datasets, like UnifiedQA), and then the macro-averaged ROUGE-L F_1 score. In case of UnifiedQA, this score is 0.48 for the model version fine-tuned on each constituent dataset individually.

6 Conclusion

We constructed Webis-CausalQA-22, the first large benchmark dataset of 1.1 million causal question–answer pairs, which serves to advance research in causal question answering. To ensure diversity of questions, we extracted them using seven hand-crafted high-precision lexical rules to capture as many subtypes of causal questions as possible. These rules were derived from a new typology of causal questions, which in turn is based on relevant related work on question typologies. A manual analysis of a sample of questions was used to characterize causal questions in terms of two dimensions: (1) their semantic properties, i.e., according to which element of the causal structure the question is asked (antecedent, consequent, or the causal chain) and (2) their pragmatic interpretation, i.e., the underlying intention or assumed information need of the questioner (e.g., prevention of medical problems). Furthermore, a subsequent analysis of the causal questions contained in a search engine log showed that a significant proportion of 5% of question queries are causal. Finally, we evaluated the state-of-the-art model UnifiedQA on our corpus as an initial baseline for causal question answering.

Causal questions represent a hitherto poorly considered challenge for both search engines in general and QA systems in particular. In this respect, our typology serves as a guide for the development of new technologies: The semantic dimension is relevant for understanding queries, while the pragmatic dimension may guide search engines and QA systems in finding and presenting answers. In addition, linking current text-based models with algorithms for causal inference is a promising direction to answer more complex questions for which answers cannot be found directly on the web. CauseNet may also prove useful here, as the graph of cause–effect relationships already makes such connections. However, to maximize user confidence in an information system's answers to causal questions, all causal claims must be supported by evidence (e.g., in the form of scientific studies).

Acknowledgements

This work has been partially supported by the Deutsche Forschungsgemeinschaft (DFG) in the project "ACQuA 2.0: Answering Comparative Questions with Arguments" (project 376430233) as part of the priority program "RATIO: Robust Argumentation Machines" (SPP 1999). This work has also been partially supported by the German Federal Ministry of Education and Research (BMBF) within the project EML4U under the grant no 01IS19080B and within the project COLIDE under the grant no 01I521005D as well as by the German Federal Ministry for Economic Affairs and Climate Action (BMWK) within the project RAKI under the grant no 01MD19012B. The authors acknowledge the Paderborn Center for Parallel Computing (PC^2) for providing computing resources. Pavel Braslavski acknowledges funding from the Ministry of Science and Higher Education of the Russian Federation (project 075-02-2022-877). We thank Yandex for granting access to the data.

References

Akari Asai, Kazuma Hashimoto, Hannaneh Hajishirzi, Richard Socher, and Caiming Xiong. 2020. Learn-

ing to retrieve reasoning paths over Wikipedia graph for question answering. In *Proceedings of the 8th International Conference on Learning Representations (ICLR 2020)*.

Sören Auer, Christian Bizer, Georgi Kobilarov, Jens Lehmann, Richard Cyganiak, and Zachary G. Ives. 2007. DBpedia: A nucleus for a web of open data. In *The Semantic Web, 6th International Semantic Web Conference, 2nd Asian Semantic Web Conference (ISWC 2007 + ASWC 2007)*, pages 722–735.

John Langshaw Austin. 1962. *How to do Things with Words*. Oxford University Press, Oxford, UK.

Alexander Bondarenko, Pavel Braslavski, Michael Völske, Rami Aly, Maik Fröbe, Alexander Panchenko, Chris Biemann, Benno Stein, and Matthias Hagen. 2020. Comparative web search questions. In *Proceedings of the Thirteenth ACM International Conference on Web Search and Data Mining (WSD 2020), Houston, TX, USA, February 3–7, 2020*, pages 52–60.

Manvi Breja and Sanjay Kumar Jain. 2017. Why-type question classification in question answering system. In *Working notes of FIRE 2017 - Forum for Information Retrieval Evaluation*, pages 149–153.

Andrei Z. Broder. 2002. A taxonomy of web search. *SIGIR Forum*, 36(2):3–10.

B. Barla Cambazoglu, Mark Sanderson, Falk Scholer, and W. Bruce Croft. 2020. A review of public datasets in question answering research. *SIGIR Forum*, 54(2):5:1–5:23.

Nilesh Chakraborty, Denis Lukovnikov, Gaurav Maheshwari, Priyansh Trivedi, Jens Lehmann, and Asja Fischer. 2021. Introduction to neural network-based question answering over knowledge graphs. *WIREs Data Mining Knowl. Discov.*, 11(3).

Eugene Charniak. 1975. A partial taxonomy of knowledge about actions. In *Advance Papers of the Fourth International Joint Conference on Artificial Intelligence (IJCAI 1975)*, pages 91–98.

Danqi Chen, Adam Fisch, Jason Weston, and Antoine Bordes. 2017. Reading Wikipedia to answer open-domain questions. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (ACL 2017)*, pages 1870–1879.

Matthew Dunn, Levent Sagun, Mike Higgins, V. Ugur Güney, Volkan Cirik, and Kyunghyun Cho. 2017. SearchQA: A new Q&A dataset augmented with context from a search engine. *CoRR*, abs/1704.05179.

Angela Fan, Yacine Jernite, Ethan Perez, David Grangier, Jason Weston, and Michael Auli. 2019. ELI5: Long form question answering. In *Proceedings of the 57th Conference of the Association for Computational Linguistics (ACL 2019)*, pages 3558–3567.

Andrew Gelman. 2011. Causality and statistical learning. *American Journal of Sociology*, 117(3):955–966.

Andrew Gelman and Guido Imbens. 2013. Why ask why? Forward causal inference and reverse causal questions. NBER Working Papers 19614, National Bureau of Economic Research, Inc.

Andrew S. Gordon, Zornitsa Kozareva, and Melissa Roemmele. 2012. SemEval-2012 Task 7: Choice of plausible alternatives: An evaluation of commonsense causal reasoning. In *Proceedings of the 6th International Workshop on Semantic Evaluation (SemEval@NAACL-HLT 2012)*, pages 394–398.

Art Graesser, Vasile Rus, and Zhiqiang Cai. 2008. Question classification schemes. In *Proceedings of the Workshop on Question Generation*, pages 10–17.

Arthur C. Graesser and Natalie K. Person. 1994. Question asking during tutoring. *American Educational Research Journal*, 31(1):104–137.

Bert F. Green, Alice K. Wolf, Carol Chomsky, and Kenneth Laughery. 1961. Baseball: An automatic question-answerer. In *Proceedings of the 1961 Western Joint Computer Conference, (IRE-AIEE-ACM 1961)*, pages 219–224.

Oktie Hassanzadeh, Debarun Bhattacharjya, Mark Feblowitz, Kavitha Srinivas, Michael Perrone, Shirin Sohrabi, and Michael Katz. 2019. Answering binary causal questions through large-scale text mining: An evaluation using cause-effect pairs from human experts. In *Proceedings of the 28th International Joint Conference on Artificial Intelligence (IJCAI 2019)*, pages 5003–5009.

Stefan Heindorf, Yan Scholten, Henning Wachsmuth, Axel-Cyrille Ngonga Ngomo, and Martin Potthast. 2020. CauseNet: Towards a causality graph extracted from the web. In *Proceedings of the 29th ACM International Conference on Information and Knowledge Management (CIKM 2020)*, pages 3023–3030.

Eduard Hovy, Ulf Hermjakob, and Deepak Ravichandran. 2002. A question/answer typology with surface text patterns. In *Proceedings of the Second International Conference on Human Language Technology Research (HLT 2002)*, pages 247–251.

Eduard H. Hovy, Laurie Gerber, Ulf Hermjakob, Michael Junk, and Chin-Yew Lin. 2000. Question answering in Webclopedia. In *Proceedings of The Ninth Text REtrieval Conference (TREC 2000)*.

Xiao Huang, Jingyuan Zhang, Dingcheng Li, and Ping Li. 2019. Knowledge graph embedding based question answering. In *Proceedings of the Twelfth ACM International Conference on Web Search and Data Mining (WSDM 2019)*, pages 105–113.

Ryu Iida, Canasai Kruengkrai, Ryo Ishida, Kentaro Torisawa, Jong-Hoon Oh, and Julien Kloetzer. 2019. Exploiting background knowledge in compact answer generation for why-questions. In *Proceedings of the Thirty-Third AAAI Conference on Artificial Intelligence (AAAI 2019), The Thirty-First Innovative Applications of Artificial Intelligence Conference (IAAI 2019), The Ninth AAAI Symposium on Educational Advances in Artificial Intelligence (EAAI 2019)*, pages 142–151.

Ryo Ishida, Kentaro Torisawa, Jong-Hoon Oh, Ryu Iida, Canasai Kruengkrai, and Julien Kloetzer. 2018. Semi-distantly supervised neural model for generating compact answers to open-domain why questions. In *Proceedings of the Thirty-Second AAAI Conference on Artificial Intelligence (AAAI-18), the 30th innovative Applications of Artificial Intelligence (IAAI-18), and the 8th AAAI Symposium on Educational Advances in Artificial Intelligence (EAAI-18)*, pages 5803–5811.

Mandar Joshi, Eunsol Choi, Daniel S. Weld, and Luke Zettlemoyer. 2017. TriviaQA: A large scale distantly supervised challenge dataset for reading comprehension. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (ACL 2017)*, pages 1601–1611.

Humayun Kayesh, Md. Saiful Islam, and Junhu Wang. 2019. On event causality detection in tweets. *CoRR*, abs/1901.03526.

Humayun Kayesh, Md. Saiful Islam, Junhu Wang, Shikha Anirban, A. S. M. Kayes, and Paul A. Watters. 2020. Answering binary causal questions: A transfer learning based approach. In *Proceedings of the International Joint Conference on Neural Networks (IJCNN 2020)*, pages 1–9.

Daniel Khashabi, Yeganeh Kordi, and Hannaneh Hajishirzi. 2022. UnifiedQA-v2: Stronger generalization via broader cross-format training. *CoRR*, abs/2202.12359.

Daniel Khashabi, Sewon Min, Tushar Khot, Ashish Sabharwal, Oyvind Tafjord, Peter Clark, and Hannaneh Hajishirzi. 2020. UnifiedQA: Crossing format boundaries with a single QA system. In *Findings of the Association for Computational Linguistics (EMNLP 2020)*, pages 1896–1907.

Daniel Khashabi, Amos Ng, Tushar Khot, Ashish Sabharwal, Hannaneh Hajishirzi, and Chris Callison-Burch. 2021. GooAQ: Open question answering with diverse answer types. In *Findings of the Association for Computational Linguistics (EMNLP 2021)*, pages 421–433.

Tom Kwiatkowski, Jennimaria Palomaki, Olivia Redfield, Michael Collins, Ankur P. Parikh, Chris Alberti, Danielle Epstein, Illia Polosukhin, Jacob Devlin, Kenton Lee, Kristina Toutanova, Llion Jones, Matthew Kelcey, Ming-Wei Chang, Andrew M. Dai, Jakob Uszkoreit, Quoc Le, and Slav Petrov. 2019. Natural Questions: A benchmark for question answering research. *Trans. Assoc. Comput. Linguistics*, 7:452–466.

Yash Kumar Lal, Nathanael Chambers, Raymond Mooney, and Niranjan Balasubramanian. 2021. TellMeWhy: A dataset for answering why-questions in narratives. In *Findings of the Association for Computational Linguistics (ACL/IJCNLP 2021)*, pages 596–610.

Wendy Grace Lehnert. 1977. *The Process of Question Answering*. Ph.D. thesis, Yale University.

Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Veselin Stoyanov, and Luke Zettlemoyer. 2020. BART: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics (ACL 2020)*, pages 7871–7880.

Patrick S. H. Lewis, Yuxiang Wu, Linqing Liu, Pasquale Minervini, Heinrich Küttler, Aleksandra Piktus, Pontus Stenetorp, and Sebastian Riedel. 2021. PAQ: 65 million probably-asked questions and what you can do with them. *Trans. Assoc. Comput. Linguistics*, 9:1098–1115.

Chin-Yew Lin. 2004. ROUGE: A package for automatic evaluation of summaries. In *Text Summarization Branches Out*, pages 74–81.

Ilya Loshchilov and Frank Hutter. 2019. Decoupled weight decay regularization. In *Proceedings of the 7th International Conference on Learning Representations (ICLR 2019)*.

Nicholas Lourie, Ronan Le Bras, Chandra Bhagavatula, and Yejin Choi. 2021. UNICORN on RAINBOW: A universal commonsense reasoning model on a new multitask benchmark. In *Proceedings of the Thirty-Fifth AAAI Conference on Artificial Intelligence (AAAI 2021), Thirty-Third Conference on Innovative Applications of Artificial Intelligence (IAAI 2021), Eleventh Symposium on Educational Advances in Artificial Intelligence, (EAAI 2021)*, pages 13480–13488.

Xiaolu Lu, Soumajit Pramanik, Rishiraj Saha Roy, Abdalghani Abujabal, Yafang Wang, and Gerhard Weikum. 2019. Answering complex questions by joining multi-document evidence with quasi knowledge graphs. In *Proceedings of the 42nd International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR 2019)*, pages 105–114.

John McClure, Denis J Hilton, Jodie Cowan, Lucyna Ishida, and Marc Wilson. 2001. When people explain difficult actions, is the causal question how or why? *Journal of Language and Social Psychology*, 20(3):339–357.

Arindam Mitra and Chitta Baral. 2016. Addressing a question answering challenge by combining statistical methods with inductive rule learning and reasoning. In *Proceedings of the Thirtieth AAAI Conference on Artificial Intelligence (AAAI 2016)*, pages 2779–2785.

Salman Mohammed, Peng Shi, and Jimmy Lin. 2018. Strong baselines for simple question answering over knowledge graphs with and without neural networks. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL-HLT 2018)*, pages 291–296.

Dan I. Moldovan, Sanda M. Harabagiu, Marius Pasca, Rada Mihalcea, Roxana Girju, Richard Goodrum, and Vasile Rus. 2000. The structure and performance of an open-domain question answering system. In *Proceedings of the 38th Annual Meeting of the Association for Computational Linguistics (ACL 2000)*, pages 563–570.

Dan I. Moldovan, Marius Pasca, Sanda M. Harabagiu, and Mihai Surdeanu. 2003. Performance issues and error analysis in an open-domain question answering system. *ACM Trans. Inf. Syst.*, 21(2):133–154.

Tri Nguyen, Mir Rosenberg, Xia Song, Jianfeng Gao, Saurabh Tiwary, Rangan Majumder, and Li Deng. 2016. MS MARCO: A human generated machine reading comprehension dataset. In *Proceedings of the Workshop on Cognitive Computation: Integrating neural and symbolic approaches 2016 co-located with the 30th Annual Conference on Neural Information Processing Systems (NIPS 2016)*.

Pranav Rajpurkar, Robin Jia, and Percy Liang. 2018. Know what you don't know: Unanswerable questions for SQuAD. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (ACL 2018)*, pages 784–789.

Pranav Rajpurkar, Jian Zhang, Konstantin Lopyrev, and Percy Liang. 2016. SQuAD: 100,000+ questions for machine comprehension of text. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing (EMNLP 2016)*, pages 2383–2392.

Apoorv Saxena, Aditay Tripathi, and Partha P. Talukdar. 2020. Improving multi-hop question answering over knowledge graphs using knowledge base embeddings. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics (ACL 2020)*, pages 4498–4507.

Roger C. Schank. 1975. The structure of episodes in memory. In *Representation and Understanding*, pages 237–272. Morgan Kaufmann.

Abigail See, Peter J. Liu, and Christopher D. Manning. 2017. Get to the point: Summarization with pointer-generator networks. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (ACL 2017)*, pages 1073–1083.

Rebecca Sharp, Mihai Surdeanu, Peter Jansen, Peter Clark, and Michael Hammond. 2016. Creating causal embeddings for question answering with minimal supervision. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing (EMNLP 2016)*, pages 138–148.

Alon Talmor, Jonathan Herzig, Nicholas Lourie, and Jonathan Berant. 2019. CommonsenseQA: A question answering challenge targeting commonsense knowledge. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL-HLT 2019)*, pages 4149–4158.

Alon Talmor, Ori Yoran, Ronan Le Bras, Chandra Bhagavatula, Yoav Goldberg, Yejin Choi, and Jonathan Berant. 2021. CommonsenseQA 2.0: Exposing the limits of AI through gamification. In *Proceedings of the Neural Information Processing Systems Track on Datasets and Benchmarks (NeurIPS 2021)*.

Niket Tandon, Bhavana Dalvi, Keisuke Sakaguchi, Peter Clark, and Antoine Bosselut. 2019. WIQA: A dataset for "what if..." reasoning over procedural text. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP 2019)*, pages 6075–6084.

Robert S Taylor. 1962. The process of asking questions. *American Documentation*, 13(4):391–396.

Adam Trischler, Tong Wang, Xingdi Yuan, Justin Harris, Alessandro Sordoni, Philip Bachman, and Kaheer Suleman. 2017. NewsQA: A machine comprehension dataset. In *Proceedings of the 2nd Workshop on Representation Learning for NLP (Rep4NLP@ACL 2017)*, pages 191–200.

Suzan Verberne, Lou Boves, Peter-Arno Coppen, and Nelleke Oostdijk. 2006a. Discourse-based answering of why-questions. *Trait. Autom. des Langues*, 47(2):21–41.

Suzan Verberne, Lou Boves, Nelleke Oostdijk, and Peter-Arno Coppen. 2006b. Data for question answering: The case of why. In *Proceedings of the Fifth International Conference on Language Resources and Evaluation (LREC 2006)*.

Suzan Verberne, Lou Boves, Nelleke Oostdijk, and Peter-Arno Coppen. 2007. Evaluating discourse-based answer extraction for why-question answering. In *Proceedings of the 30th Annual International SIGIR Conference on Research and Development in Information Retrieval (SIGIR 2007)*, page 735–736.

Suzan Verberne, Lou Boves, Nelleke Oostdijk, and Peter-Arno Coppen. 2008. Using syntactic information for improving why-question answering. In *Proceedings of the 22nd International Conference on Computational Linguistics (COLING 2008)*, pages 953–960.

Suzan Verberne, Lou Boves, Nelleke Oostdijk, and Peter-Arno Coppen. 2010. What is not in the bag of words for *Why*-QA? *Comput. Linguistics*, 36(2):229–245.

Daniel Vollmers, Rricha Jalota, Diego Moussallem, Hardik Topiwala, Axel-Cyrille Ngonga Ngomo, and Ricardo Usbeck. 2021. Knowledge graph question answering using graph-pattern isomorphism. In *Proceedings of the 17th International Conference on Semantic Systems (SEMANTiCS 2017)*, pages 103–117.

Michael Völske, Pavel Braslavski, Matthias Hagen, Galina Lezina, and Benno Stein. 2015. What users ask a search engine: Analyzing one billion russian question queries. In *24th ACM International Conference on Information and Knowledge Management (CIKM 2015)*, pages 1571–1580.

Zhilin Yang, Peng Qi, Saizheng Zhang, Yoshua Bengio, William W. Cohen, Ruslan Salakhutdinov, and Christopher D. Manning. 2018. HotpotQA: A dataset for diverse, explainable multi-hop question answering. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing (EMNLP 2018)*, pages 2369–2380.

Weiguo Zheng, Jeffrey Xu Yu, Lei Zou, and Hong Cheng. 2018. Question answering over knowledge graphs: Question understanding via template decomposition. *Proc. VLDB Endow.*, 11(11):1373–1386.

MACRONYM: A Large-Scale Dataset for Multilingual and Multi-Domain Acronym Extraction

Amir Pouran Ben Veyseh[1], **Nicole Meister**[2], **Seunghyun Yoon**[3],
Rajiv Jain[3], **Franck Dernoncourt**[3], **and Thien Huu Nguyen**[1]
[1]Department of Computer Science, University of Oregon, OR, USA
[2]Department of Computer Science, Stanford University, CA, USA
[3]Adobe Research, USA
{apouranb, thien}@cs.uoregon.edu, nmeist@stanford.edu
{syoon,rajijain,franck.dernoncourt}@adobe.com

Abstract

Acronym extraction is the task of identifying acronyms and their expanded forms in texts that is necessary for various NLP applications. Despite major progress for this task in recent years, one limitation of existing AE research is that they are limited to the English language and certain domains (i.e., scientific and biomedical). Challenges of AE in other languages and domains are mainly unexplored. As such, lacking annotated datasets in multiple languages and domains has been a major issue to prevent research in this direction. To address this limitation, we propose a new dataset for multilingual and multi-domain AE. Specifically, 27,200 sentences in 6 different languages and 2 new domains, i.e., legal and scientific, are manually annotated for AE. Our experiments on the dataset show that AE in different languages and learning settings has unique challenges, emphasizing the necessity of further research on multilingual and multi-domain AE.

1 Introduction

Acronyms are short forms of longer phrases that are often constructed using a few letters selected from the long phrases. Due to their functionality, acronyms are common in many languages and domains. For instance, 73% of abstracts of scientific papers contain at least an acronym (Barnett and Doubleday, 2020). As such, in text processing applications, e.g., question answering and machine translation, it is necessary to correctly identify the acronyms and their meanings. Toward this goal, our work focus on the task of Acronym Extraction (AE), aiming to recognize acronyms and their definitions/long forms in text. For instance, in the sentence "*They will meet in the conference of the World Trade Organization (WTO)*", an AE system should identify "*WTO*" and "*World Trade Organization*" as the acronym and long form, respectively.

Despite all progress in recent years, prior work on AE has mainly limited to specific domains and languages. Specifically, biomedical and scientific texts in English have been the main focus in prior work. However, recognition of acronyms in other languages and domains is also important and might involve challenges not reflected in English biomedical/scientific texts. For instance, many existing AE methods for English employ uppercase letters to identify acronyms (Veyseh et al., 2020). However, in non-case sensitive languages, e.g., Arabic or Persian, uppercase letter concept does not exist, thus causing a failure of existing AE systems. Moreover, in each domain or language, different styles might be exerted to shorten a longer phrase to produce acronyms. For instance, initial letters of the words in the phrases are commonly used to form acronyms in scientific English; however, in legal English or Danish documents, the use of initial letters for acronym detection is less effective (see Section 3). As such, it is desirable to study AE in more diverse domains and languages to better support multi-domain and multilingual applications.

Unfortunately, to the best of our knowledge, there is no existing dataset for multilingual and multi-domain AE, thus impeding research effort in this area. To this end, our work addresses this issue by introducing a new manually labeled dataset for AE. In particular, based on two different domains of scientific and legal texts, our dataset annotates AE data for sentences in six different languages: English, Danish, Spanish, French, Persian, and Vietnamese. As such, legal texts, Danish, Spanish, French, Persian, and Vietnamese are not explored for AE in prior work. In addition, our dataset is large-scale, providing 27,200 annotated sentences for AE to support advanced model development (e.g., with data-hungry deep learning models).

Finally, we conduct extensive experiments to understand the challenges of AE in the created dataset. Our experiments show that the AE task in our dataset presents significant challenges for existing models in different domains and languages.

Proceedings of the 29th International Conference on Computational Linguistics, pages 3309–3314
October 12–17, 2022.

This is even more pronounced in the cross-lingual and cross-domain transfer settings where existing models perform poorly on our AE dataset. As such, more research effort is needed to address the challenges of acronym understanding in different settings. We will publicly release the dataset to foster research in this area.

2 Data Annotation

Data Collection: We collect data in two domains of legal and scientific documents for AE annotation. For each domain, documents in different languages are required. As such, for the legal domain, we employ the United Nations Parallel Corpus (UNPC) (Ziemski et al., 2016) and the Europarl corpus (Koehn, 2005). The UNPC corpus contains official records in 6 languages while the Europarl corpus consists of the proceedings of the European Parliament in European languages. To accommodate our annotation budget and diversify the resulting dataset, we choose documents from four languages in the two corpora (i.e., English, French, and Spanish in UNPC, and Danish in Europarl) for our AE annotation. In addition, for the scientific domain, we employ the publicly available papers and M.S./Ph.D. theses in the field of computer science for AE annotation. Specifically, we collect the papers published in the ACL anthology of natural language processing research for English. Also, for typologically different languages, we crawl public computer science thesis in Persian and Vietnamese.

Following (Veyseh et al., 2020), we split the selected documents into sentences that will be annotated separately by annotators. In addition, to optimize the annotation cost with greater numbers of acronyms, we apply the same procedure in (Veyseh et al., 2020) to filter out sentences that has low chance to contain acronyms or long forms. In particular, the procedure only retains sentences that involve at least one acronym candidate (i.e., a word with more than a half of characters as capital letters) and a sub-sequence of words to match the acronym candidate (i.e., concatenating the initials of the words can form the candidate) (Veyseh et al., 2020). Here, we only apply this procedure for English, French, Spanish, and Danish as our Persian and Vietnamese data is small and the sentence filtering procedure will leave less sentences for annotation. Finally, given the retained sentences for each language, we randomly sample a subset of sentences for manual AE annotation. The numbers of annotated sentences are presented in Table 1.

Annotation Process: To annotate the sampled sentences, we recruit native speakers in each language from the crowd-sourcing platform `upwork.com` with freelancer annotators across the globe. For each language, we select annotator candidates who have experience in related annotation projects and an approval rate of more than 95% (provided by Upwork). The annotator candidates are trained with guidelines and examples for AE in their language. In our annotation guideline, acronyms are required to be single words (including abbreviations). Also, for a sentence in a language, we only annotate the long forms that are in the same language as the sentence's. Afterward, for each language, we retain two candidates who pass and achieve highest results in our designed test for AE as our official annotators. Next, the two annotators in each language independently perform AE annotation for the sampled sentences of that language. Finally, the two annotators will discuss to resolve any disagreement in the annotation, thus producing a final version of our MACRONYM dataset.

Domain & Language		IAA	Size	# Unique Acronyms	# Unique Long-forms
Legal	English	0.824	4,000	3,688	3,037
	Spanish	0.810	6,400	4,059	4,437
	French	0.823	8,000	5,638	5,728
	Danish	0.810	3,000	907	923
Scientific	English	0.811	4,000	3,604	4,260
	Persian	0.782	1,000	641	203
	Vietnamese	0.791	800	270	61

Table 1: Statistics of MACRONYM. IAA scores use Krippendorff's alpha with MASI distance based on initial independent annotations. Size refers to the number of annotated sentences.

To study the challenges of AE in each language, following (Veyseh et al., 2020), we compute the inter-annotator agreement (IAA) scores using Krippendorff's alpha (Krippendorff, 2011) with the MASI distance metric (Passonneau, 2006) for the initial independent annotations of the two annotators, i.e., before resolving the conflicts. Table 1 shows the IAA scores for each language. Overall, we find that the IAA scores are high for all considered languages and domains, thus demonstrating the quality of our annotated dataset. Among several factors, a major scenario of annotation disagreement occurs in Persian or Vietnamese when a sentence contains a long form term that is translated from an original English term. However, the acronym for this long form in the Persian or Viet-

namese sentence is still formed via initials of the words in the English term. As such, some annotators consider this English-based acronym as an acronym in the Persian or Vietnamese sentence while other annotators simply ignore it in the annotation. For instance, in the Persian sentence:

"عملیات پیشرفته شبکه (ANOP) به شرح زیر است"[1],

the acronym "*ANOP*" is expressed in English letters but its long form, i.e., "عملیات پیشرفته شبکه"[2], is presented in Persian. In the resolving, we have decided to annotate any acronym that is formed using characters in the six languages in our dataset.

Data Analysis: We show the main statistics of MACRONYM in Table 1. This table shows that the density of acronyms in texts varies across different languages. On average, English sentences tend to involve more acronyms than other languages in both legal and scientific domain while Danish and Vietnamese sentences contain least acronyms in the legal and scientific domain respectively. Comparing English texts in the legal and scientific domains, we find that the ratio between the numbers of unique long-forms and acronyms is greater in the scientific domain, thus implying the higher ambiguity of acronyms in scientific documents. Finally, we note that the number of unique acronyms exceeds the number of unique long forms in Persian and Vietnamese as we do not apply the sentence filtering procedure in the data collection, thus allowing many sentences with only acronyms and no associated long forms to be annotated in the data.

3 Experiments

This section studies the challenges of the multilingual and multi-domain AE task in MACRONYM. In particular, for each pair of available languages and domains (we have 7 pairs in total), we first prepare the data by randomly splitting the corresponding set of annotated sentences into separate training/development/test portions with the ratios of 80/10/10 (respectively). Afterward, we report the performance of the representative AE models on the test set for each possible pair of languages and domains under different learning settings.

AE Models: We examine the performance of three representative state-of-the-art (SOTA) models for AE. First, we employ the rule-based system for AE proposed in (Veyseh et al., 2021) (called **Rule-Based**). This system serves as the current SOTA rule-based method for AE (Veyseh et al., 2021). In general, to detect acronyms, words with more than 60% characters as uppercase letters are selected. To find long-forms, if a detected acronym is bounded between parentheses and the initial letters of preceding words can form the acronym, the system predicts the preceding words a long form. Second, motivated by prior work (Veyseh et al., 2020; Zhu et al., 2021), we solve AE as a sequence labeling problem using BIO tagging schema. In particular, following the current SOTA deep learning model for AE (Zhu et al., 2021), we employ a pre-trained BERT-based language model followed by a feed-forward network layer with softmax in the end to predict BIO-based label for each word in the sentence. To facilitate the learning on multiple languages, we explore two multilingual transformer-based language models, i.e., mBERT (Devlin et al., 2019) and XLMR (Conneau et al., 2020), leading to two models **mBERT** and **XLMR** for this approach.

Settings: MACRONYM enables the evaluation of AE models on four different settings: (i) **Mono-Lingual Mono-Domain**: In this setting, training and test data of the models come from the same language and domain. As we have 7 possible pairs of languages and domains, this setting involves 7 different evaluations for each AE model; (ii) **Mono-Lingual Cross-Domain**: Training and test data for models belongs to the same languages, but different domains in this setting. In MACRONYM, this setting is only possible for English where AE models are trained on the legal domain but tested on the scientific domain and vice versa (i.e., two possible evaluations).; (iii) **Cross-Lingual Mono-Domain**: Assuming the same domain for training and test data, this setting trains models on English training data and evaluate them on test data of other languages. We thus have 3 and 2 possible evaluations for the legal and scientific domains respectively.; (iv) **Cross-Lingual Cross-Domain**: Training and test data for models originates from different languages and domains in this setting. As such, we also consider five evaluations in this setting. In the first two evaluations, models are trained on English data in the legal domain and evaluated on Persian and Vietnamese test data in the scientific domains. In contrast, for the other three evaluations, English data in the scientific domain is used for model training while Spanish, French, and Danish test data in the legal domain is used for evaluation.

[1]English translation: "*Advanced network operations (ANOP) include the followings*"

[2]English translation: "*advanced network operations*"

Domain & Language		Mono-Lingual Mono-Domain			Mono-Lingual Cross-Domain		Cross-Lingual Mono-Domain		Cross-Lingual Cross-Domain	
		Rule-Based	mBERT	XLMR	mBERT	XLMR	mBERT	XLMR	mBERT	XLMR
Legal	English	16.55	61.66	62.07	54.92	56.88	-	-	-	-
	Spanish	10.82	51.43	55.41	-	-	38.88	40.13	35.48	36.92
	French	10.05	58.77	61.14	-	-	48.82	50.70	44.21	46.83
	Danish	8.78	50.05	48.38	-	-	40.71	42.94	38.18	41.95
Scientific	English	20.72	60.51	59.00	56.71	59.88	-	-	-	-
	Persian	60.59	62.41	63.10	-	-	49.13	50.21	42.95	43.72
	Vietnamese	53.44	58.71	59.13	-	-	50.72	51.44	48.32	50.17

Table 2: Model performance (F1 scores) in different settings. The performance in each row is evaluated on the test data for the corresponding pair of language and domain. **Mono-Lingual Cross-Domain**: trained on English data of one domain and tested on English data of the other domain; **Cross-Lingual Settings**: trained on English data of one domain and tested on the other language data of the same domain (if Mono-Domain) or the other domain (if Cross-Domain).

We fine-tune the hyper-parameters for the models using the development data for each pair of languages and domains. Our fine-tuning process returns similar values of hyper-parameters for the models across languages and domains. In particular, for English legal data, we use 2 layers of feed-forward neural networks with 200 hidden dimensions for hidden vectors. The learning rate is set to $3e$-4 for the Adam optimizer and the batch size of 8 is utilized for training.

Results: Table 2 presents the performance of three AE models in four different settings. Note that as the **Rule-Based** system does not require training, its performance in the mono-lingual and mono-domain setting can be applied to other settings. There are several observations from the table. First, the **Rule-Based** system achieves decent performance for Persian and Vietnamese, but performs poorly for other languages. The main reason has to do with the dominance of acronyms over long forms in Persian and Vietnamese data (see Table 1). This is in contrast to other languages where acronyms and long forms are more balanced. As acronyms can be identified more easily with rules than long forms, the **Rule-Based** system is more effective in the data with much more acronyms of Persian and Vietnamese. Second, in the legal domain where long forms are better presented, the performance of the models on English is significantly better than those for other languages, thus demonstrating the more challenging nature of non-English language for AE. Third, compared to deep learning models, the significant lower performance of the **Rule-Based** model in the monolingual and mono-domain setting signifies the brittleness of human-designed rules for AE that necessitates learning models to improve the portability of models to different languages and domains. Fourth, across all learning models and language-domain pairs for test data, the lower performance in the cross-lingual mono-domain setting compared to its mono-lingual counterpart suggests the difference between languages that hinder cross-lingual transfer learning for AE. Fifth, the cross-domain performance also under-performs their mono-domain counterpart for almost all learning models and language-domain pairs for testing, thus highlighting domain shifts as an important challenge for AE. Finally, across all the learning settings, the performance of the AE models is still far from being perfect in MACRONYM, thus presenting ample opportunities for future research.

4 Related Work

Early attempts for AE have employed rule-based methods (Park and Byrd, 2001; Wren and Garner, 2002; Schwartz and Hearst, 2002; Adar, 2004; Nadeau and Turney, 2005; Kirchhoff and Turner, 2016) or feature engineering models (Kuo et al., 2009; Liu et al., 2017; Li et al., 2018). Recently, deep learning methods have delivered SOTA performance for AE (Veyseh et al., 2021; Wu et al., 2015; Antunes and Matos, 2017; Charbonnier and Wartena, 2018; Ciosici et al., 2019; Jaber and Martínez, 2021; Li et al., 2021). Despite such progress, prior AE research and datasets have mainly focused on English biomedical and scientific texts, leaving non-English languages and other domains less explored. Here, we note that there exist some acronym glossaries for non-English languages (Pomares-Quimbaya et al., 2020; Ménard and Ratté, 2011). However, such resources do not

annotate sentences/texts in multiple languages and domains for AE as we do in MACRONYM.

5 Conclusion

We present the first multilingual and multi-domain dataset for AE, involving annotation for 6 languages and 2 domains. Our experiments show that the proposed dataset presents significant challenges for AE methods in different learning settings and languages. In the future, we will expand the dataset to include more domains and languages for AE.

Acknowledgement

This research has been supported by the Army Research Office (ARO) grant W911NF-21-1-0112 and the NSF grant CNS-1747798 to the IUCRC Center for Big Learning. This research is also based upon work supported by the Office of the Director of National Intelligence (ODNI), Intelligence Advanced Research Projects Activity (IARPA), via IARPA Contract No. 2019-19051600006 under the Better Extraction from Text Towards Enhanced Retrieval (BETTER) Program. The views and conclusions contained herein are those of the authors and should not be interpreted as necessarily representing the official policies, either expressed or implied, of ARO, ODNI, IARPA, the Department of Defense, or the U.S. Government. The U.S. Government is authorized to reproduce and distribute reprints for governmental purposes notwithstanding any copyright annotation therein. This document does not contain technology or technical data controlled under either the U.S. International Traffic in Arms Regulations or the U.S. Export Administration Regulations.

References

Eytan Adar. 2004. Sarad: A simple and robust abbreviation dictionary. *Bioinformatics*, 20(4):527–533.

Rui Antunes and Sérgio Matos. 2017. Biomedical word sense disambiguation with word embeddings. In *International Conference on Practical Applications of Computational Biology & Bioinformatics*, pages 273–279. Springer.

Adrian Barnett and Zoe Doubleday. 2020. Meta-research: The growth of acronyms in the scientific literature. In *Elife*.

Jean Charbonnier and Christian Wartena. 2018. Using word embeddings for unsupervised acronym disambiguation. In *Proceedings of the 27th International Conference on Computational Linguistics*, Santa Fe, New Mexico, USA. Association for Computational Linguistics.

Manuel R Ciosici, Tobias Sommer, and Ira Assent. 2019. Unsupervised abbreviation disambiguation. *arXiv preprint arXiv:1904.00929*.

Alexis Conneau, Kartikay Khandelwal, Naman Goyal, Vishrav Chaudhary, Guillaume Wenzek, Francisco Guzmán, Edouard Grave, Myle Ott, Luke Zettlemoyer, and Veselin Stoyanov. 2020. Unsupervised cross-lingual representation learning at scale. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, Minneapolis, Minnesota. Association for Computational Linguistics.

Areej Jaber and Paloma Martínez. 2021. Participation of uc3m in sdu@ aaai-21: A hybrid approach to disambiguate scientific acronyms. In *SDU@ AAAI*.

Katrin Kirchhoff and Anne M Turner. 2016. Unsupervised resolution of acronyms and abbreviations in nursing notes using document-level context models. In *Proceedings of the Seventh International Workshop on Health Text Mining and Information Analysis*, pages 52–60.

Philipp Koehn. 2005. Europarl: A parallel corpus for statistical machine translation. In *Proceedings of Machine Translation Summit X: Papers, MTSummit 2005, Phuket, Thailand, September 13-15, 2005*.

Klaus Krippendorff. 2011. Computing krippendorff's alpha-reliability.

Cheng-Ju Kuo, Maurice HT Ling, Kuan-Ting Lin, and Chun-Nan Hsu. 2009. Bioadi: a machine learning approach to identifying abbreviations and definitions in biological literature. In *BMC bioinformatics*, volume 10, page S7. Springer.

Feng Li, Zhensheng Mai, Wuhe Zou, Wenjie Ou, Xiaolei Qin, Yue Lin, and Weidong Zhang. 2021. Systems at sdu-2021 task 1: Transformers for sentence level sequence label. In *SDU@ AAAI*.

Yang Li, Bo Zhao, Ariel Fuxman, and Fangbo Tao. 2018. Guess me if you can: Acronym disambiguation for enterprises. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1308–1317, Melbourne, Australia. Association for Computational Linguistics.

Jie Liu, Caihua Liu, and Yalou Huang. 2017. Multi-granularity sequence labeling model for acronym expansion identification. *Information Sciences*, 378:462–474.

Pierre André Ménard and Sylvie Ratté. 2011. Classifier-based acronym extraction for business documents. In *Knowledge and information systems*.

David Nadeau and Peter D Turney. 2005. A supervised learning approach to acronym identification. In *Conference of the Canadian Society for Computational Studies of Intelligence*, pages 319–329. Springer.

Youngja Park and Roy J Byrd. 2001. Hybrid text mining for finding abbreviations and their definitions. In *Proceedings of the 2001 conference on empirical methods in natural language processing*.

Rebecca Passonneau. 2006. Measuring agreement on set-valued items (masi) for semantic and pragmatic annotation. In *LREC*.

Alexandra Pomares-Quimbaya, Pilar López-Úbeda, Michel Oleynik, and Stefan Schulz. 2020. Leveraging pubmed to create a specialty-based sense inventory for spanish acronym resolution. In *Digital Personalized Health and Medicine*.

Ariel S Schwartz and Marti A Hearst. 2002. A simple algorithm for identifying abbreviation definitions in biomedical text. In *Biocomputing 2003*, pages 451–462. World Scientific.

Amir Pouran Ben Veyseh, Franck Dernoncourt, Walter Chang, and Thien Huu Nguyen. 2021. MadDog: A web-based system for acronym identification and disambiguation. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: System Demonstrations*.

Amir Pouran Ben Veyseh, Franck Dernoncourt, Quan Hung Tran, and Thien Huu Nguyen. 2020. What does this acronym mean? introducing a new dataset for acronym identification and disambiguation. In *Proceedings of the 28th International Conference on Computational Linguistics*.

Jonathan D Wren and Harold R Garner. 2002. Heuristics for identification of acronym-definition patterns within text: towards an automated construction of comprehensive acronym-definition dictionaries. *Methods of information in medicine*, 41(05):426–434.

Yonghui Wu, Jun Xu, Yaoyun Zhang, and Hua Xu. 2015. Clinical abbreviation disambiguation using neural word embeddings. In *Proceedings of BioNLP 15*, pages 171–176.

Danqing Zhu, Wangli Lin, Yang Zhang, Qiwei Zhong, Guanxiong Zeng, Weilin Wu, and Jiayu Tang. 2021. At-bert: Adversarial training bert for acronym identification winning solution for sdu@aaai-21. In *Proceedings of the AAAI-21 Workshop on Scientific Document Understanding, Shared Task (SDU@AAAI-21)*.

Michal Ziemski, Marcin Junczys-Dowmunt, and Bruno Pouliquen. 2016. The united nations parallel corpus v1.0. In *Proceedings of the Tenth International Conference on Language Resources and Evaluation LREC 2016, Portorož, Slovenia, May 23-28, 2016*.

Curating a Large-Scale Motivational Interviewing Dataset using Peer Support Forums

Anuradha Welivita and **Pearl Pu**
School of Computer and Communication Sciences
École Polytechnique Fédérale de Lausanne
Switzerland
{kalpani.welivita,pearl.pu}@epfl.ch

Abstract

A significant limitation in developing therapeutic chatbots to support people going through psychological distress is the lack of high-quality, large-scale datasets capturing conversations between clients and trained counselors. As a remedy, researchers have focused their attention on scraping conversational data from peer support platforms such as Reddit. But the extent to which the responses from peers align with responses from trained counselors is understudied. We address this gap by analyzing the differences between responses from counselors and peers by getting trained counselors to annotate ≈17K such responses using Motivational Interviewing Treatment Integrity (MITI) code, a well-established behavioral coding system that differentiates between favorable and unfavorable responses. We developed an annotation pipeline with several stages of quality control. Due to its design, this method was able to achieve 97% of coverage, meaning that out of the 17.3K responses we successfully labeled 16.8K with a moderate agreement. We use this data to conclude the extent to which conversational data from peer support platforms align with real therapeutic conversations and discuss in what ways they can be exploited to train therapeutic chatbots.

1 Introduction

Demands of the modern world are increasingly responsible for bringing adverse impacts on our mental health. World Health organization estimates psychological distress affects 29% of people in their lifetime (Steel et al., 2014). Shortage of mental health workers and the stigma associated with mental health further demotivates people in actively seeking out help. Thus, provision of mental health support through the use of AI-driven conversational agents to complement traditional therapy has become an interesting area of research (Fitzpatrick et al., 2017; Inkster et al., 2018; Mousavi et al., 2021). But one challenge associated with developing such agents is the lack of large-scale psychotherapeutic conversations. They are either limited or are not available publicly due to ethical reasons.

Nowadays, with the expansion of social media, it could be observed that people use social media platforms such as Reddit to vent their distress and peers are seen to actively respond to such posts. These conversations are available in abundance and are publicly accessible through web scraping APIs. Thus, conversations scraped from such platforms are seen as an alternative to overcome the above challenge (Welivita and Pu, 2022). Prior work has found that responses from peers contain higher empathic concern for posts for seeking help as many peers share similar distressful experiences (Hodges et al., 2010). But the extent to which responses from peers align with responses from trained counselors remain a major limitation. Knowing these differences can shed light on to what extent such conversational data could be used in training therapeutic chatbots and what pre-processing or rephrasing steps that one should take if they are being used for such purposes.

Though studies have been conducted independently to assess the competency of counselors and peers offering support (Pérez-Rosas et al., 2016; Klonek et al., 2015; Gaume et al., 2009; Sharma et al., 2020a; De Choudhury and De, 2014), studies that comparatively analyse the differences between them are limited. One such study was conducted by Lahnala et al. (2021), where they show that a classifier can distinguish between responses provided to help-seeking posts regarding mental health by professionals and peers. Mousavi et al. (2021) conducted a similar analysis between responses collected from psychotherapists and non-expert dialogue writers and noted linguistic variability in the two types of responses. However, all these analyses are limited to the lexical level.

To address the above gaps, we comparatively an-

Proceedings of the 29th International Conference on Computational Linguistics, pages 3315–3330
October 12–17, 2022.

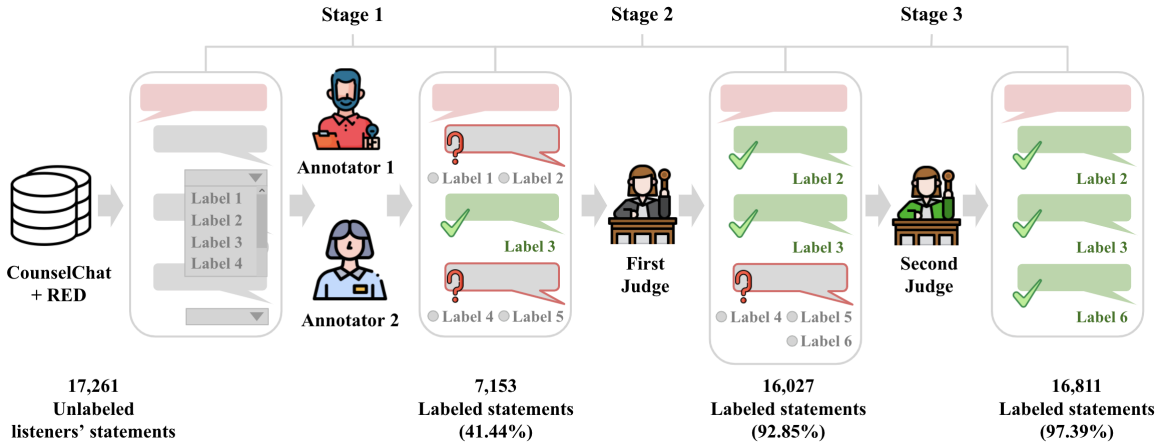

Figure 1: The annotation process to label the listeners' statements in the CounselChat and RED datasets with labels adapted from MITI. The process was conducted in three main stages.

alyze responses from professional counselors and peers by annotating these responses using labels adapted from a well-established behavioral coding scheme named Motivational Interviewing Treatment Integrity (MITI) code (Moyers et al., 2014). The MITI code is used in psychological literature to evaluate how well a mental health practitioner responds to those seeking help with their mental health related issues. Specific response types from the MITI code have shown to increase the likelihood of positive health outcomes (Pérez-Rosas et al., 2018; Gaume et al., 2009). For this purpose, we make use of post-response pairs scraped from the CounselChat website (counselchat.com) that contains high-quality therapist responses to emotional distress related questions from individuals and dialogues curated from several mental-health related subreddits in Reddit, in which peers engage in actively to respond to those seeking help.

Annotating dialogue responses with labels from the MITI coding system is known to be very time consuming and expensive as it requires expert annotators trained in the practice of psychology and careful attention to the labelling task (Pérez-Rosas et al., 2016). This human labour is difficult to find. But the availability of crowdsourcing platforms such as UpWork [1] and Fiverr [2] have made it more accessible to find human labour that satisfy our requirements. Thus, we were able to recruit professionally trained mental health practitioners through UpWork to annotate dialogue responses with labels adapted from the MITI code. Our annotation pipeline consisted of three stages as depicted in Figure 1. Two workers were involved in the first stage and high-quality workers who scored high observed agreements with a peer in the first stage acted as judges to resolve conflicting labels in the second and third stages contributing to increased observed agreement and inter-rater reliability.

Using these annotations, we conducted a comparative analysis between responses from peers and counselors to identify to what extent they align with each other. Based on these findings, we recommend ways of boosting peers' responses to match as close as possible to counselors' responses. The recommendations made in this paper can contribute to improve the perceived therapeutic effectiveness of a chatbot trained on data from peer support forums.

Our contributions are three fold. 1) We develop an MI dataset having client-counselor and peer-peer dialogues, in which ≈17K listeners' utterances are annotated with labels adapted from the MITI code.[3] 2) We discuss the details of the annotation process followed in increasing the agreement between the workers when annotating with MITI codes. 3) Based on these annotations, we conduct a comparative analysis between counselors' and peers' responses and draw useful conclusions on to what extent responses from peers align with responses from trained counselors and recommend ways of boosting peers' responses such that it can increase the perceived effectiveness of therapeutic chatbots trained on such data.

2 Related Work

Motivational Interviewing Treatment Integrity (MITI) is a behavioral coding system that measures

[1] http://upwork.com
[2] https://www.fiverr.com

[3] The dataset can be downloaded at https://github.com/anuradha1992/Motivational-Interviewing-Dataset.

how well a mental health practitioner uses motivational interviewing in therapy (Moyers et al., 2003). It exclusively focuses on the verbal behaviour of a counselor and is used to increase clinical skill in the practice of motivational interviewing (MI). This coding system has been used extensively to to improve the understanding of the counseling practice alone (Can et al., 2012; Pérez-Rosas et al., 2018, 2019). Pérez-Rosas et al. (2016) developed an MI dataset consisting of ≈22K counselors' responses during Motivational Interviewing encounters annotated with 10 behavioral codes from the MITI. Althoff et al. (2016) conducted a quantitative study on counseling conversations to measure how various linguistic aspects of conversations are correlated with conversation outcomes. However, the datasets used in such analyses are not publicly available due to ethical reasons. Thus, it is difficult to use such resources in training therapeutic chatbots even though real counselor responses are the ideal candidates for this purpose.

There are a number of research efforts taken to develop therapeutic chatbots (Fitzpatrick et al., 2017; Inkster et al., 2018; Welch et al., 2020; Mousavi et al., 2021). Among them, recent work focuses on using dialogue data from peer-support forums (Sharma et al., 2020b; Welivita and Pu, 2022). For example, Welivita and Pu (2022) developed a knowledge-graph containing distress consoling responses for specific types of stressors using peer-support responses from Reddit and utilized it in single-turn dialogues. Some studies specifically focus on attributes such as perceived empathy and information richness in mental health related discourse in peer support forums that suggests they are good candidates for training such chatbots (Nambisan, 2011; De Choudhury and De, 2014; Sharma et al., 2020a,b). But these studies lack comparisons with responses generated by professional counselors. In our work, we mainly attempts to address this limitation.

3 Methodology

In this section, we describe the methodology including the labels chosen to annotate the listeners' statements, the datasets used, and different stages of the annotation process.

3.1 Labels Adapted from MITI

The labels we used for annotation were adapted from MITI code 2.0 (Moyers et al., 2003) and 4.2.1 (Moyers et al., 2014). Table 1 shows the MITI labels that were used for annotation with descriptions and examples. Altogether, there are 15 labels. They also include labels *Self-Disclose* and *Other*, which are not included in the MITI code. We included the label *Self-Disclose* because in conversations involving peer support, *Self-Disclosure* is an important aspect that enables the sharing of lived experience and is seen to occur quite frequently in the majority of such conversations (Truong et al., 2019). The above labels were used to annotate each sentence in the listeners' utterances.

3.2 Datasets

We used two datasets containing distress consoling dialogues for annotation: 1) *CounselChat* that contains responses from professional counselors; and 2) *RED* that contains responses from peers.

CounselChat Dataset: The CounselChat dataset consists of high-quality therapist responses to emotional distress related questions from individuals. This data is scraped from the CounselChat website (counselchat.com), which is an online platform that helps counselors to make meaningful contact with potential clients. On the website, professional counselors respond to questions posed by users suffering from mental health issues and emotional distress. The dataset consists of 2,129 post-response pairs that span across 31 distress related topics, the most frequent topics being *depression*, *relationships*, and *intimacy*. This dataset is publicly available. [4] Out of this data, we randomly selected 1,000 post-response pairs to be annotated with labels derived from the MITI code.

Reddit Emotional Distress (RED) Dataset: To obtain a dialogue dataset containing utterances from peer-supporters as response for posts containing emotional distress, we scraped a new dataset from Reddit, containing dialogues that discuss real-world distressful situations. Reddit was chosen since it is publicly available and peers actively engage in Reddit to support others going through distressful situations in life. We used the Pushshift API (Baumgartner et al., 2020) to collect and process dialogue threads from a carefully selected set of 8 subreddits: *mentalhealthsupport*; *offmychest*; *sad*; *suicidewatch*; *anxietyhelp*; *depression*; *depressed*; and *depression_help*, which are popular among Reddit users to vent their distress. We ex-

[4] https://github.com/nbertagnolli/counsel-chat

MITI label	Description	Examples
1. Closed Question	Questions that can be answered with an yes/no response or a very restricted range of answers.	*Do you think this is an advantage?*
2. Open Question	Questions that allow a wide range of possible answers.	*What is your take on that?*
3. Simple Reflection	Repetition, rephrasing, or paraphrasing of speaker's previous statement.	*It sounds like you're feeling worried.*
4. Complex Reflection	Repeating or rephrasing the previous statement of the speaker but adding substantial meaning/emphasis to it.	**Speaker:** *Mostly, I would change for future generations.* **Listener:** *It sounds like you have a strong feeling of responsibility.*
5. Give Information	Educating, providing feedback, or giving an opinion without advising.	*Logging your cravings is important as cravings often lead to relapses.*
MI Adherent Behaviour Codes:		
6. Advise with Permission	Advising when the speaker asks directly for advice. Indirect forms of permission can also occur, such as when the listener says to disregard the advice as appropriate.	*If you agree with it, we could try to brainstorm some ideas that might help.*
7. Affirm	Encouraging the speaker by saying something positive or complimentary.	*You should be proud of yourself for your past's efforts.*
8. Emphasize Autonomy	Emphasizing the speaker's control, freedom of choice, autonomy, and ability to decide.	*It is really up to you to decide.*
9. Support	Statements of compassion or sympathy.	*I know it's really hard to stop drinking.*
MI Non-Adherent Behaviour Codes:		
10. Advise without Permission	Making suggestions, offering solutions or possible actions without first obtaining permission from the speaker.	*You should simply scribble a note that reminds you to take a break.*
11. Confront	Directly and unambiguously disagreeing, arguing, blaming, criticizing, or questioning the speaker's honesty.	*Yes, you are an alcoholic. You might not think so, but you are.*
12. Direct	Giving orders, commands, or imperatives.	*Don't do that!*
13. Warn	A statement or event that warns of something or that serves as a cautionary example.	*Be careful, DO NOT stop taking meds without discussing with your doctor.*
Other:		
14. Self-Disclose	The listener discloses his/her personal information or experiences.	*I used to be similar where I get obsessed about how people look.*
15. Other	Statements that are not classified under the above codes	*Good morning, Hi there.*

Table 1: The set of labels adapted from the MITI code, which were used to annotate listeners' responses.

tracted the dialogue turns out of these threads and subjected these dialogues to a rigorous data cleaning pipeline, which included removal of profanity from listener responses. By this, we were able to curate $\approx 1.2M$ dyadic conversations having on average 2.66 turns per dialogue. Out of them, 1K dialogues were randomly selected for annotation.

3.3 Annotation Experiment

We used UpWork, a leading crowdsourcing platform to recruit trained counselors to annotate dialogue responses from CounselChat and RED datasets. Altogether 12 workers were recruited to annotate 2K dialogues, 1K from CounselChat and 1K from RED. They contained 17,261 individual sentences in the listener utterances in total.

The task was carried out in three stages. First, the workers were asked to annotate each sentence contained in the listener utterances of the dialogues from CounselChat and RED datsets with one of fifteen MITI labels. We bundled ten dialogues into one batch (a batch contained five CounselChat and five RED dialogues interchangeably) and assigned two workers per batch. At the beginning, a tutorial about the labels accompanied by a practice task was offered to self-validate the workers' answers. As the task was ongoing, we computed the observed agreement of each worker with peers and offered more batches for the workers whose observed agreement was better than the others.

At the end of stage 1, we noticed that the two workers assigned for each batch did not agree on a common label for more than half of the listeners' sentences in the two datasets. Manual inspection of their answers revealed majority of the disagreements occurred because there are highly confusing pairs of labels that need more careful attention to differentiate (e.g. *Complex Reflection* and *Give Information* can be easily confused). Hence, we launched a second stage of the experiment by ask-

ing four workers who scored the highest observed agreement with a peer in the first stage to act as judges to resolve these conflicting labels. A judge was presented with the two labels the workers specified in the first stage along with the listener's sentence and the dialogue context and was asked to select either one of the two labels if one of them agreed with the listener's sentence. Only if none of the labels agreed with the listener's sentence, he was instructed to select a label from the rest.

At the end of stage 2, most of the conflicting labels were resolved by the judge's annotations. But there was a small percentage of listeners' sentences for which a label was still not agreed upon. We noticed for 68.15% of such unresolved sentences, at least one annotation was given by a relatively poor performing worker whose observed agreement score with a peer was below average as measured in the first stage. We decided such labels are not worth considering since they cloud the decision process and chose to launch a third stage of the experiment by removing one annotation given by the poorest performing worker for each such unresolved sentence. Similar to stage 2, we recruited the same judges and presented them with the two remaining labels to be resolved. This entire annotation pipeline is illustrated in Figure 1.

4 Results

Table 2 shows the statistics of the results from each stage of the experiment. At the end of stage 1, out of 17,261 listeners' sentences, 7,152 received a label as agreed by the two annotators. Altogether, by end of stage 1, we could yield an observed agreement percentage of 41.43% and an inter-rater agreement (Fleiss' kappa) score of 0.3391 indicating fair agreement. At the end of stage 2, another 8,875 labels were resolved, yielding an observed agreement of 87.79%. The updated inter-rater agreement (Fleiss' kappa) after this stage was 0.5292, which is a significant increase compared to the previous stage. After the end of completion of stage 1 and stage 2 of the annotation process, from among the total of 17,261 listeners' sentences in CounselChat and RED datasets, 16,027 of them were able to receive a label as agreed by at least two workers. This is 92.85% of the entire data.

From the remaining 1,234 sentences for which a label was not agreed upon, 841 (68.15%) sentences were annotated by at least one poor worker whose observed agreement with a peer was below average. At the end of stage 3 of the experiment, which was conducted by removing such annotations given by the poor workers, a second judge was able to agree with one of the two remaining labels for 784 sentences, yielding an observed agreement of 93.22%. The updated inter-rater agreement (Fleiss' kappa) after the third stage was 0.5453 (moderate agreement), showing a slight increase compared to the score in the previous stage. The lower kappa scores are potentially due to the inherent difficulty of distinguishing some of the MI labels, which we elaborate below. A similar annotation experiment conducted by Perez-Rosas et al. (2016) report similar kappa scores ranging from 0.31 to 0.64 on different MI labels.

The confusion matrices computed at each stage of the experiment are denoted in the appendices. We could observe that the label pair *Complex Reflection - Give Information* had the highest percentage of disagreement between the two workers in both CounselChat and RED datasets. In addition, the label pairs *Advise with Permission - Advise without Permission* and *Give Information - Advise without Permission* were highly confusing to differentiate in the CounselChat dataset. Whereas, in RED, the label pairs *Affirm - Support* and *Advise without Permission - Direct* were difficult to be differentiated. These observations were quite intuitive since these pairs of labels either contained similar semantic content (e.g. *Complex Reflection - Give Information*, *Advise with Permission - Advise without Permission*, *Give Information - Advise without Permission*, *Advise without Permission - Direct*) or used similar language constructs (e.f. *Affirm - Support*, *Advise without Permission - Direct*).

Final aggregated statistics of the three stages of the annotation process is shown in Table 3. It could be observed how the labels grew to cover a larger portion of the listeners' sentences as the annotation process advanced through the stages. Finally, close to 97% of the listeners' sentences (16,812 in total) were annotated with the MITI labels.

5 Analysis of the MI Dataset

In Figure 2, we show the distribution of labels adapted from the MITI code in CounselChat and RED datasets, separately. Table 4 shows the specific number of each label in CounselChat and RED datasets and the increase/decrease in each label in the two datasets compared to each other. Based on these statistics, we discuss seven major differences

Description	CC	RED	CC + RED
Stage 1:			
Total number of listeners' sentences annotated	9,893	7,368	17,261
Sentences for which a label was agreed upon by both annotators	4,067	3,085	7,152
Observed agreement between the two annotators	41.11%	41.87%	41.43%
Inter-rater agreement (Fleiss' kappa)	0.3059	0.3577	0.3391
Stage 2:			
The number sentences, for which, the label had to be resolved	5,826	4,283	10,109
The number of times the judge agreed with one of the given labels	5,111	3,764	8,875
Observed agreement between the judge and one of the two annotators	87.73%	87.88%	87.79%
Updated inter-rater agreement (Fleiss' kappa)	0.5029	0.5440	0.5292
Stage 3:			
The number sentences, for which, the label had to be resolved	479	362	841
The number of times the judge agreed with one of the given labels	450	334	784
Observed agreement b/w the judge and one of the remaining annotators	93.95%	92.27%	93.22%
Updated inter-rater agreement (Fleiss' kappa)	0.5193	0.5601	0.5453

Table 2: Statistics of the three stages of the annotation experiment. The CounselChat dataset is abbreviated as CC.

Description	CC	RED	CC + RED
# listener sentences	9,893	7,368	17,261
# labels agreed in stage 1	3,085 (41.11%)	4,067 (41.87%)	7,152 (41.43%)
# labels agreed in stage 2	9,178 (92.96%)	6,849 (92.77%)	16,027 (92.85%)
# labels agreed in stage 3	9,628 (97.49%)	7,183 (97.32%)	16,811 (97.39%)

Table 3: Final aggregated statistics of the three stages of the annotation process.

observed between responses from counselors and peers and state our recommendations when using this data to train therapeutic conversational agents.

Questions: There is ≈23% and ≈26% increase in closed and open questions, respectively, in counselor responses from CC compared to peer-support responses from RED. Questioning plays a central role in therapeutic interactions as it builds up mutual dialogue between client and therapist (Poskiparta et al., 2000). But Hill et al. (1983) observed with time-limited counseling, fact finding through closed questions is rated lower in helpfulness. It can result in the speaker saying less and less and the listener feeling pressured to ask more questions to keep the interaction going. However, in both CounselChat and RED datasets, the number of open questions is nearly half of the number of closed questions. Hence, mechanisms should be devised to increase the percentage of open questions to balance the number of closed questions. This combination would be more effective than a disproportionate reliance on closed questions.

Reflections: The number of reflections is positively associated with the perceived empathy (Klonek et al., 2015). It is also a competence indicator in assessing MI competency (Moyers et al., 2003). Non-surprisingly, both simple and complex reflections are observed to be higher (≈20% and ≈30% increase in simple and complex reflections, respectively) in counselors' responses compared to peers'. Thus, it would be beneficial to boost the percentage of reflections among peer support dialogues when using them to train therapeutic agents.

Scholars emphasize that listeners should formulate more reflections than closed questions (Klonek et al., 2015). As we observed, some closed questions such as *"Are you eating because you are bored?"* are identical to reflections, differing only in the voice intonation at the end. They could be easily reformulated into reflections such as *"It seems that you are eating because you are bored"*.

Giving information: In counselor responses, there is a 200.33% increase of *Give Information* type of sentences compared to responses from peers. It is quite unsurprising since counselors are relatively knowledgeable about the subject being discussed and hence are in a position to provide information that can help the speaker. Informed by this observation, steps should be taken to boost the amount of information in peer-support responses.

MI Adherent Behavior Codes: Supporting the client with statements of compassion and sympathy are surprisingly higher among peers (≈95% increase) compared to counselors. Affirming the speaker by saying something positive or complimentary is also seen to be comparatively higher in RED (≈21% increase). These are very good indicators that show peer-support responses if uti-

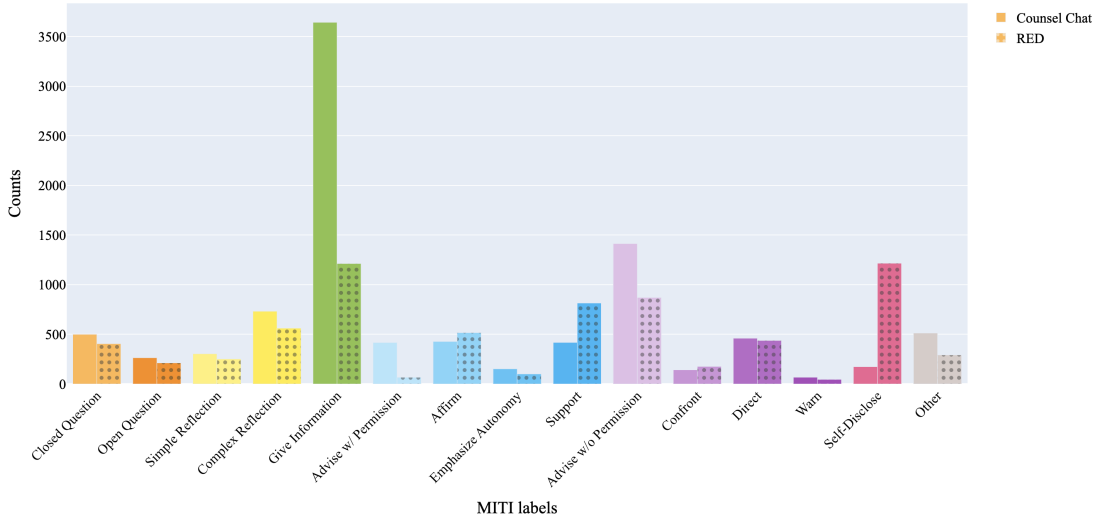

Figure 2: Distribution of MITI labels in CounselChat and RED datasets.

Label	No. of labels in CounselChat	No. of labels in RED	Increase in CC compared to RED	Increase in RED compared to CC
Closed Question	500	405	23.46% ↑	-19.00% ↓
Open Question	264	212	24.53% ↑	-19.70% ↓
Simple Reflection	304	252	20.63% ↑	-17.11% ↓
Complex Reflection	732	562	30.25% ↑	-23.22% ↓
Give Information	3,643	1213	200.33% ↑	-66.70% ↓
MI Adherent Behavior Codes:				
Advise with Permission	417	67	522.39% ↑	-83.93% ↓
Affirm	428	517	-17.21% ↓	20.79% ↑
Emphasize Autonomy	152	101	50.50% ↑	-33.55% ↓
Support	418	815	-48.71% ↓	94.98% ↑
MI Non-Adherent Behavior Codes:				
Advise without Permission	1,414	871	62.34% ↑	-38.40% ↓
Confront	142	176	-19.32% ↓	23.94% ↑
Direct	460	438	5.02% ↑	-4.78% ↓
Warn	67	46	45.65% ↑	-31.34% ↓
Other:				
Self-Disclose	174	1216	-85.69% ↓	598.85% ↑
Other	513	292	75.68% ↑	-43.08% ↓

Table 4: Statistics of MITI labels in CounselChat and RED datasets and the increase/decrease in each label in the two datasets compared to each other. The increases/decreases that are favourable for the interaction are indicated in green while those that are unfavourable are indicated in red. The increases/decreases in *Self-Disclose* and *Other* are not assigned a color as their role in therapeutic interventions are quite blurry and subjected to debate.

lized in training therapeutic agents will reflect more compassion, sympathy, positivity, and compliments towards the user in distress. On the other hand, emphasizing the speaker's control and autonomy is observed to be higher in counselors' responses (≈50% increase) compared to responses from peers.

Advising with and without permission: Giving advices is generally seen to be higher in counselor responses. There is ≈522% increase in advising after asking for permission and ≈62% increase in advising without asking for permission among counselors' responses compared to those from the peers. Advising without asking for permission takes a portion of 77.22% of the total number of advices given in counselor responses. Thus, counselors, though professionally trained, tend to make the mistake of advising without prior asking for the speaker's permission. This percentage is higher in peer-support responses in which advising without permission takes a portion of 92.86% of the total number of advices given by the peers. Thus, in both datasets, steps should be taken to reformulate advices in a way that the agent asks for the speakers' permission before giving advice.

MI Non-Adherent Behavior Codes: Confronting the client by directly disagreeing, arguing, or criticizing is higher in peers' responses (≈24% increase) compared to those of the counselors'.

Such interactions reflect uneven power sharing, accompanied by disapproval and negativity (Moyers et al., 2003). Directing the speaker by giving orders and also warnings are quite surprisingly seen to be slightly higher in the responses given by the counselors compared to the responses of the peers (≈5% and ≈46% increase for *Direct* and *Warn*, respectively). These are non-favourable response types that negatively affect the therapeutic interaction between the speaker and the listener and thus should be detected and eliminated as a preliminary step before using such responses to train chatbots.

Self-Disclosure: The role of self-disclosure in therapeutic interventions is quite blurry. For example, psychoanalysts believe that self-disclosure is counterproductive as it distorts client's transference. Conversely, Cognitive Behavioural therapists believe that self-disclosure can be a useful tool in therapy as it models and reinforces new perspectives for the client. Digging deep, there are two broad types of self-disclosure used by counsellors: 1) *intra-session disclosure*, where the counselor discloses a feeling about the client that is relevant to the therapeutic process; and 2) *extra-session disclosure*, where the counselor reveals information about themselves that occurs outside the session. In most cases, *intra-session disclosure* is the most useful type of self-disclosure (Levitt et al., 2016).

As we manually inspected the statements labeled as *Self-disclosure* in CounselChat and RED datasets, it was found out that *Intra-session disclosure* is seen higher in CC compared to RED, whereas *Extra-session disclosure* is seen higher in RED compared to CC. Table 5 provides some examples of such statements. This suggests that counselors are more careful when disclosing information about themselves and when they do they make sure that the information they disclose is relevant to the therapeutic process. Extra-session disclosure too has its place in therapeutic interactions specially contributing to building rapport between the client and the therapist. However, as suggested by R. Schwartz (2021), this type of disclosure must be used wisely with caution since it can as well be counterproductive distorting client's transference.

Examples of *intra-session disclosure* in CounselChat:
- Personally, I can tell you that I would want my clients to tell me about anxiety they feel 100% of the time.
- I have had clients asking the same question and there is often an underlying fear that they "can't be helped" or they will "be too much for their therapist."
Examples of *extra-session disclosure* in RED:
- You remind me a lot of my best friend that I had when I was young. Being her friend was exhausting.
- I too suffer from psychosis from my schizo-affective disorder, yelled at my former best friend for gangstalking me, called her all kinds of horrible names.

Table 5: Examples of different types of self-disclosure observed in CounselChat and RED datasets.

6 Discussion and Conclusion

This paper discussed the curation process of a large-scale distress consoling dialogue dataset containing utterances from trained counselors and peers. A carefully designed annotation process was followed to annotate each response statement with labels adapted from the MITI code. We saw the effectiveness of our annotation process as it contributed in increasing the observed agreement and inter-rater reliability as the process advanced through different stages. Based on the comparative analysis between responses from counselors and peers, we reported seven major differences between them, highlighting the strengths and limitations of using abundantly available peer-support dialogues for purposes such as training therapeutic chatbots. In summary, peers' responses tend to be more supportive, compassionate, and encouraging than counselors' as observed by the increased percentage of *Support* and *Affirm* labels. But important therapeutic techniques such as asking more open questions than closed ones, reflections, giving information and advices with permission, and emphasizing speaker's autonomy require further boosting. MI non-adherent behaviors such as confronting is also seen higher among peers and thus should be eliminated. Careful attention should also be paid to self-disclosure among peers as the majority of such statements are of the type *extra-session disclosure*, which is less useful for the therapeutic process.

Curating this dataset is the first step in our general goal of boosting the therapeutic competency in peer-support responses. Using this dataset we plan to train an MITI classifier to automatically identify favourable and unfavourable response types present in peers' responses. Being able to detect MI non-adherent behaviors such as confronting will enable us to directly eliminate such responses from the data. Next, we intend to develop an MITI rephraser that can convert certain types of responses such as closed questions and advices without permission into more favourable reflections and advices with permission, respectively. We plan on investigating simple linguistic rule based approaches as well as

unsupervised text style transfer methods that can be trained on unparalleled corpora (Malmi et al., 2020; Jin et al., 2022) for this purpose. We believe this will largely boost the therapeutic competency in peer-support responses and will increase the therapeutic effectiveness in chatbots trained on them.

7 Ethics Statement

Data curation: Analysis of posts of a website such as Reddit is likely considered "fair play" as individuals are anonymous and users are aware that their responses remain archived on the site unless they explicitly delete them. The Reddit privacy policy also states it allows third parties to access public Reddit content through the Reddit API and other similar technologies. [5] Reddit's data is already widely available in larger dumps such as Pushshift (Baumgartner et al., 2020). We collected only publicly available data in Reddit and the curation process did not involve any intervention or interaction with the Reddit users. The CounselChat dataset is also available publicly. But Fiesler and Proferes (2018) in a study on user perceptions on social media research ethics empahsizes some potential harms that can be caused due to social computing research because internet users rarely read or could fully understand website terms and conditions. Since this dataset in particular contains sensitive information, we adhere to the guidelines suggested by Benton et al. (2017) for working with social media data in health research, and share only anonymized and paraphrased excerpts from the dataset so that it is not possible to recover usernames through a web search with the verbatim post text. In addition, references to usernames as well as URLs are removed from dialogue content for de-identification.

Human annotation: Considering the qualifications of the workers we recruited, who were all trained in the practice of counseling, we were determined to pay them a wage considerably above the US minimum wage of $7.12 per hour. We paid them $10 per batch of 10 dialogues in the first stage of the experiment, in which average task completion time took ≈30 minutes (excluding the time taken by workers who took an unusually long time to complete the task). A bonus of $5 was offered for each batch if a worker obtained an above average observed agreement with a peer. For the second and third stages of the annotation task, we offered the workers 5 per batch of 10 dialogues. The average completion time per batch was ≈15 minutes in these two stages. Since the dataset is in English, all the annotators recruited were either native speakers or had professional competency in the English language. The fact that the dataset is English-only potentially perpetuates an English bias in NLP systems.

Therapeutic chatbots: Finally, there can be certain ethical implications associated with the development of therapeutic chatbots as highlighted by several researchers (Lanteigne, 2019; Montemayor et al., 2021; Tatman, 2022). However, the idea of therapeutic chatbots is not a new concept. Chatbots such as SimSensei (DeVault et al., 2014), Dipsy (Xie, 2017), Emma (Ghandeharioun et al., 2019), Woebot (`woebothealth.com`), and Wysa (`www.wysa.io`) are some examples. As Czerwinski et al. (2021) state, *About 1 billion people globally are affected by mental disorders; a scalable solution such as an AI therapist could be a huge boon*. Thus, even though therapeutic chatbots may encompass certain ethical implications, based on previous studies we already can acknowledge that the use of chatbots has the potential to improve therapeutic services notably in relation to accessibility and anonymity.

We curated this dataset for the ultimate development of a chatbot that adheres to MI strategy when responding to emotional distress. With the significant performance achieved by recent pre-trained language models, going for a deep learning-based solution is one of the choices that can be taken when developing such an agent. But it should not be undermined that because of the unpredictability associated with generative models, they always carry a risk when delivering emotional support to those undergoing distress. Thus, caution should be taken to avoid the delivery of inappropriate responses. This may not be limited to avoiding profane or judgemental responses. As pointed out by R. Tatman (2022) a response such as *"You're not alone"* may be comforting to someone with depression, however, can bring detrimental effects to someone suffering from paranoia. Hence, caution should be taken when developing therapeutic chatbots based on this dataset. Real-world deployment of such agents may still encompass potential dangers and if deployed, should be done with human supervision.

[5] `www.redditinc.com/policies/privacy-policy-october-15-2020`

References

Tim Althoff, Kevin Clark, and Jure Leskovec. 2016. Large-scale analysis of counseling conversations: An application of natural language processing to mental health. *Transactions of the Association for Computational Linguistics*, 4:463–476.

Jason Baumgartner, Savvas Zannettou, Brian Keegan, Megan Squire, and Jeremy Blackburn. 2020. The pushshift reddit dataset. *Proceedings of the International AAAI Conference on Web and Social Media*, 14(1):830–839.

Adrian Benton, Glen Coppersmith, and Mark Dredze. 2017. Ethical research protocols for social media health research. In *Proceedings of the First ACL Workshop on Ethics in Natural Language Processing*, pages 94–102.

Doğan Can, Panayiotis G Georgiou, David C Atkins, and Shrikanth S Narayanan. 2012. A case study: Detecting counselor reflections in psychotherapy for addictions using linguistic features. In *Thirteenth Annual Conference of the International Speech Communication Association*.

Mary Czerwinski, Javier Hernandez, and Daniel McDuff. 2021. Building an ai that feels: Ai systems with emotional intelligence could learn faster and be more helpful. *IEEE Spectrum*, 58(5):32–38.

Munmun De Choudhury and Sushovan De. 2014. Mental health discourse on reddit: Self-disclosure, social support, and anonymity. In *Eighth international AAAI conference on weblogs and social media*.

David DeVault, Ron Artstein, Grace Benn, Teresa Dey, Ed Fast, Alesia Gainer, Kallirroi Georgila, Jon Gratch, Arno Hartholt, Margaux Lhommet, et al. 2014. Simsensei kiosk: A virtual human interviewer for healthcare decision support. In *Proceedings of the 2014 international conference on Autonomous agents and multi-agent systems*, pages 1061–1068.

Casey Fiesler and Nicholas Proferes. 2018. "participant" perceptions of twitter research ethics. *Social Media+ Society*, 4(1).

Kathleen Kara Fitzpatrick, Alison Darcy, and Molly Vierhile. 2017. Delivering cognitive behavior therapy to young adults with symptoms of depression and anxiety using a fully automated conversational agent (woebot): a randomized controlled trial. *JMIR mental health*, 4(2):e7785.

Jacques Gaume, Gerhard Gmel, Mohamed Faouzi, and Jean-Bernard Daeppen. 2009. Counselor skill influences outcomes of brief motivational interventions. *Journal of substance abuse treatment*, 37(2):151–159.

Asma Ghandeharioun, Daniel McDuff, Mary Czerwinski, and Kael Rowan. 2019. Emma: An emotion-aware wellbeing chatbot. In *International Conference on Affective Computing and Intelligent Interaction*.

Clara E Hill, Jean A Carter, and Mary K O'Farrell. 1983. A case study of the process and outcome of time-limited counseling. *Journal of Counseling Psychology*, 30(1):3.

Sara D Hodges, Kristi J Kiel, Adam DI Kramer, Darya Veach, and B Renee Villanueva. 2010. Giving birth to empathy: The effects of similar experience on empathic accuracy, empathic concern, and perceived empathy. *Personality and Social Psychology Bulletin*, 36(3):398–409.

Becky Inkster, Shubhankar Sarda, Vinod Subramanian, et al. 2018. An empathy-driven, conversational artificial intelligence agent (wysa) for digital mental well-being: real-world data evaluation mixed-methods study. *JMIR mHealth and uHealth*, 6(11):e12106.

Di Jin, Zhijing Jin, Zhiting Hu, Olga Vechtomova, and Rada Mihalcea. 2022. Deep learning for text style transfer: A survey. *Computational Linguistics*, 48(1):155–205.

Florian E Klonek, Vicenç Quera, and Simone Kauffeld. 2015. Coding interactions in motivational interviewing with computer-software: What are the advantages for process researchers? *Computers in Human Behavior*, 44:284–292.

Allison Lahnala, Yuntian Zhao, Charles Welch, Jonathan K. Kummerfeld, Lawrence C An, Kenneth Resnicow, Rada Mihalcea, and Verónica Pérez-Rosas. 2021. Exploring self-identified counseling expertise in online support forums. In *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*, pages 4467–4480, Online. Association for Computational Lingfgfggftzr666757tl.uistics.

Camylle Lanteigne. 2019. Social robots and empathy: The harmful effects of always getting what we want.

Heidi M Levitt, Takuya Minami, Scott B Greenspan, Jae A Puckett, Jennifer R Henretty, Catherine M Reich, and Jeffery S Berman. 2016. How therapist self-disclosure relates to alliance and outcomes: A naturalistic study. *Counselling Psychology Quarterly*, 29(1):7–28.

Eric Malmi, Aliaksei Severyn, and Sascha Rothe. 2020. Unsupervised text style transfer with padded masked language models. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 8671–8680, Online. Association for Computational Linguistics.

Carlos Montemayor, Jodi Halpern, and Abrol Fairweather. 2021. In principle obstacles for empathic ai: why we can't replace human empathy in healthcare. *AI & society*, pages 1–7.

Seyed Mahed Mousavi, Alessandra Cervone, Morena Danieli, and Giuseppe Riccardi. 2021. Would you like to tell me more? generating a corpus of psychotherapy dialogues. In *Proceedings of the Second Workshop on Natural Language Processing for Medical Conversations*, pages 1–9.

TB Moyers, JK Manuel, D Ernst, T Moyers, J Manuel, D Ernst, and C Fortini. 2014. Motivational interviewing treatment integrity coding manual 4.1 (miti 4.1). *Unpublished manual*.

Theresa B Moyers, Tim Martin, Jennifer K Manuel, William R Miller, and D Ernst. 2003. The motivational interviewing treatment integrity (miti) code: Version 2.0. *Retrieved from Verfübar unter: www. casaa. unm. edu [01.03. 2005]*.

Priya Nambisan. 2011. Information seeking and social support in online health communities: impact on patients' perceived empathy. *Journal of the American Medical Informatics Association*, 18(3):298–304.

Verónica Pérez-Rosas, Rada Mihalcea, Kenneth Resnicow, Satinder Singh, and Lawrence An. 2016. Building a motivational interviewing dataset. In *Proceedings of the Third Workshop on Computational Linguistics and Clinical Psychology*, pages 42–51.

Verónica Pérez-Rosas, Xuetong Sun, Christy Li, Yuchen Wang, Kenneth Resnicow, and Rada Mihalcea. 2018. Analyzing the quality of counseling conversations: the tell-tale signs of high-quality counseling. In *Proceedings of the Eleventh International Conference on Language Resources and Evaluation (LREC 2018)*.

Verónica Pérez-Rosas, Xinyi Wu, Kenneth Resnicow, and Rada Mihalcea. 2019. What makes a good counselor? learning to distinguish between high-quality and low-quality counseling conversations. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 926–935.

Marita Poskiparta, Tarja Kettunen, and Leena Liimatainen. 2000. Questioning and advising in health counselling: results from a study of finnish nurse counsellors. *Health Education Journal*, 59(1):69–89.

Robert Schwartz. 2021. The big reveal | ethical implications of therapist self-disclosure.

Ashish Sharma, Monojit Choudhury, Tim Althoff, and Amit Sharma. 2020a. Engagement patterns of peer-to-peer interactions on mental health platforms. In *Proceedings of the International AAAI Conference on Web and Social Media*, volume 14, pages 614–625.

Ashish Sharma, Adam S Miner, David C Atkins, and Tim Althoff. 2020b. A computational approach to understanding empathy expressed in text-based mental health support. *arXiv preprint arXiv:2009.08441*.

Zachary Steel, Claire Marnane, Changiz Iranpour, Tien Chey, John W Jackson, Vikram Patel, and Derrick Silove. 2014. The global prevalence of common mental disorders: a systematic review and meta-analysis 1980–2013. *International journal of epidemiology*, 43(2):476–493.

Rachael Tatman. 2022. [link].

C Truong, J Gallo, D Roter, and J Joo. 2019. The role of self-disclosure by peer mentors: Using personal narratives in depression care. *Patient education and counseling*, 102(7):1273–1279.

Charles Welch, Allison Lahnala, Veronica Perez-Rosas, Siqi Shen, Sarah Seraj, Larry An, Kenneth Resnicow, James Pennebaker, and Rada Mihalcea. 2020. Expressive interviewing: A conversational system for coping with covid-19. *arXiv preprint arXiv:2007.03819*.

Anuradha Welivita and Pearl Pu. 2022. Heal: A knowledge graph for distress management conversations.

Xing Xie. 2017. Dipsy: A digital psychologist.

A Annotation Experiment

A.1 User Interfaces

Figures 5 and 6 shows the user interfaces of the first and second stages of the MITI annotation experiment conducted in UpWork. The first stage is when two workers from UpWork were asked to annotate each sentence contained in the listener utterances of the dialogues from CounselChat and RED datasets and the second stage is when a high quality worker was asked to act as a judge to resolve the disagreements occured in the first stage. Interfaces similar to the second stage were used in the third stage as well. To educate the worker on the MITI coding scheme and the labels we derived out of it, a detailed tutorial was shown to the worker at the beginning of the task. This is shown in Figure 3c. A practice task to self-evaluate their competence in annotating responses with the labels derived from the MITI code followed next. Figure 4c depicts this.

A.2 More About the MITI

The MITI does not contain an exhaustive list of all possible codes; thus not all sentences can be mapped to a label from the MITI code. In this case, the annotators were asked to select *Other*. Also, the labels from the MITI code are mutually exclusive. Thus, the same sentence could not receive more than one label.

A.3 Worker Quality

In stage 1 of the annotation process, to motivate the workers to pay attention to the task, we offered to pay them a bonus of $5 for each batch of dialogues that scored an above average observed agreement with a peer worker. Out of 400 worker assignments (200 batches $\times$ 2 workers per batch), 140 of them (35%) were able to receive this bonus.

As the task progressed, those who scored higher observed agreements with the peer workers were allocated more batches to annotate.

In the second and third stages, to validate the quality of the judges and their attentiveness to the task, hidden checkpoints were included to measure the workers' attentiveness to the task. These checkpoints were based on the labels agreed by the two workers in the first stage of the task. In each batch of 10 dialogues, we randomly selected 10 sentences for which a label was agreed in the first stage. For each such sentence, we randomly sampled another label out of the remaining labels and showed it along with the correct label for the judge to select from. The four judges we recruited were able to get in overall 84.3% questions correct in stage 2 of the annotation task. The scores for each of the four judges were 80.00%, 86.47%, 86.47%, and 87.50%. In the third stage, they were able to get in overall 82.93% questions correct. Their individual scores were 83.00%, 83.64%, 82.00%, and 83.00%. All the scores being above 80% in both stages indicates that they all were paying significant attention to the task.

A.4 Confusion Matrices

Figure 5 shows the confusion matrices in stage 1 of the experiment between the two annotators for the CounselChat and RED datasets separately. Labels such as *Give Information*, *Advise without Permission*, and *Closed Question* had the highest agreement between the two workers in the CounselChat dataset, whereas in RED, the highest agreed labels were *Self-Disclose*, *Give Information*, and *Support*.

Figure 6 shows the confusion matrices between the two annotators for sentence for which the label was unresolved in stage 1 and between each of these annotators and the judge in stage 2 of the annotation process. From the second and third confusion matrices corresponding to each dataset, it could seen how the judge's annotations aligned with annotations from each annotator from stage 1.

Figure 7 shows the confusion matrices between the two remaining annotators after the annotations from the poorly performed worker are removed and between each of these annotators and the second judge in stage 2 of the annotation process. Note that in the remaining two annotations, the first one comes from a relatively better performed worker from stage 1 and the second one comes from the first judge from stage 2. By observing the confusion matrices, it was noted that 73.34% times, the second judge agreed with the annotation provided by the first judge in stage 2. This further validated the quality of the judges selected.

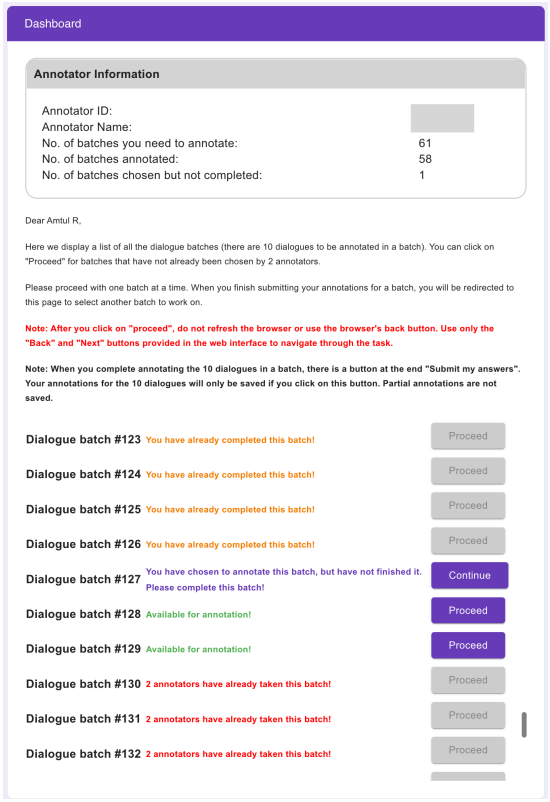

(a) The dashboard interface

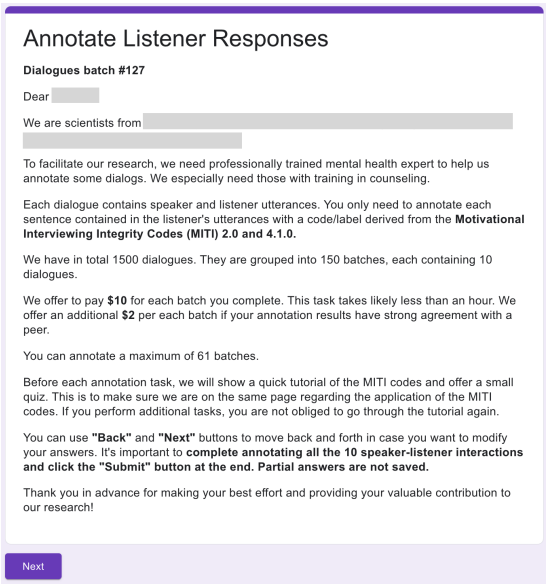

(b) Instructions

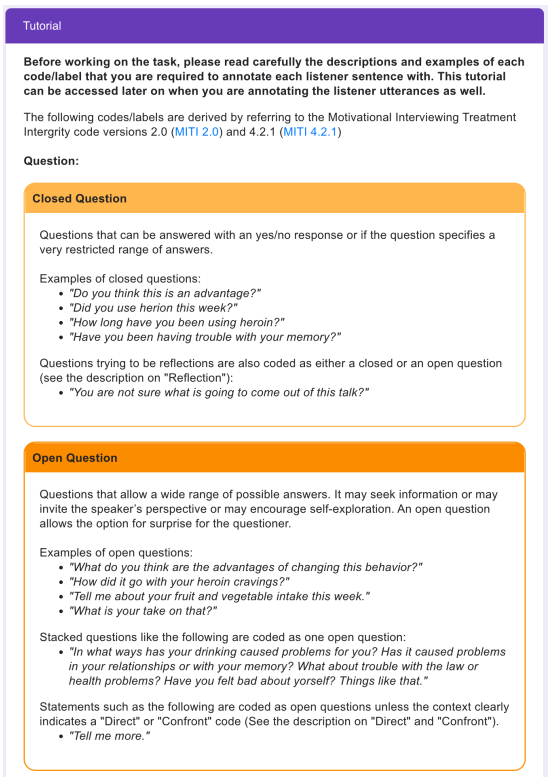

(c) The tutorial

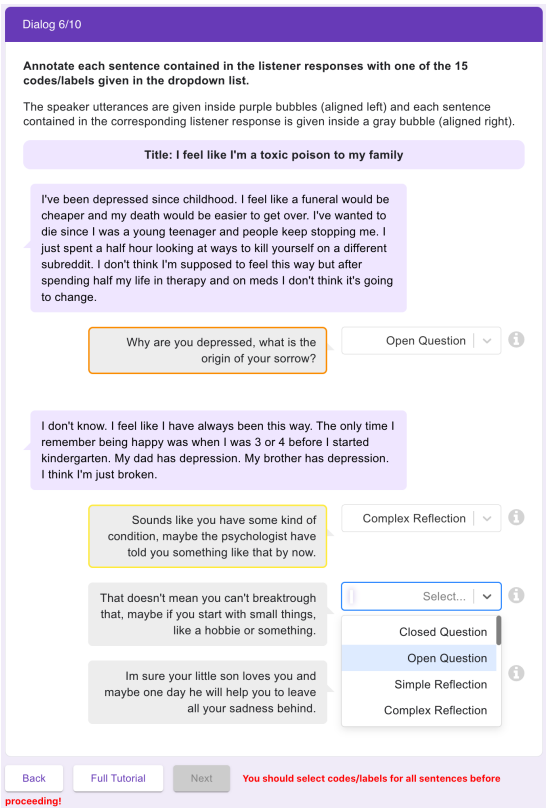

(d) The annotation task interface

Figure 3: User interfaces of the first stage of the MITI annotation experiment.

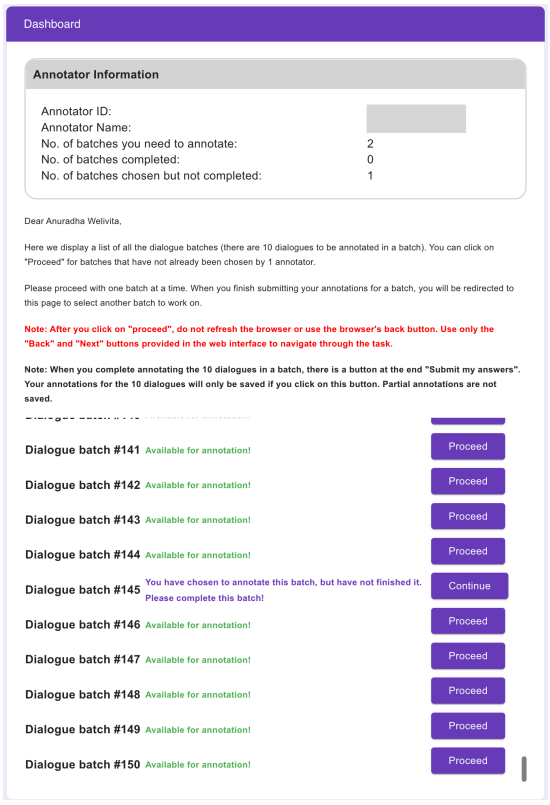

(a) The dashboard interface

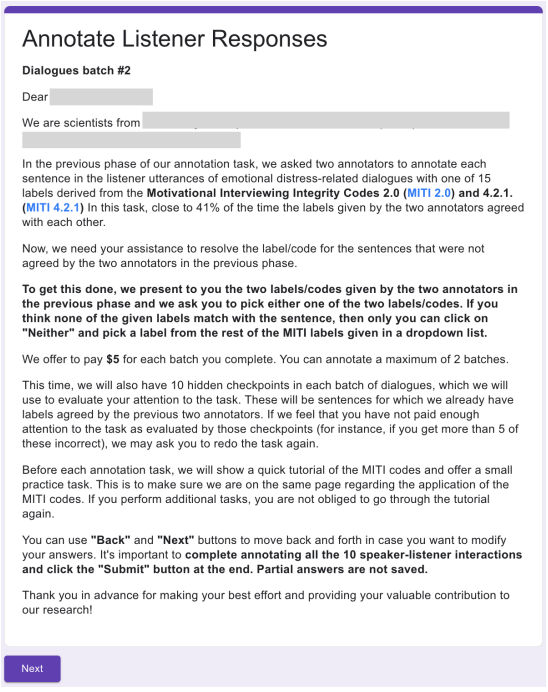

(b) Instructions

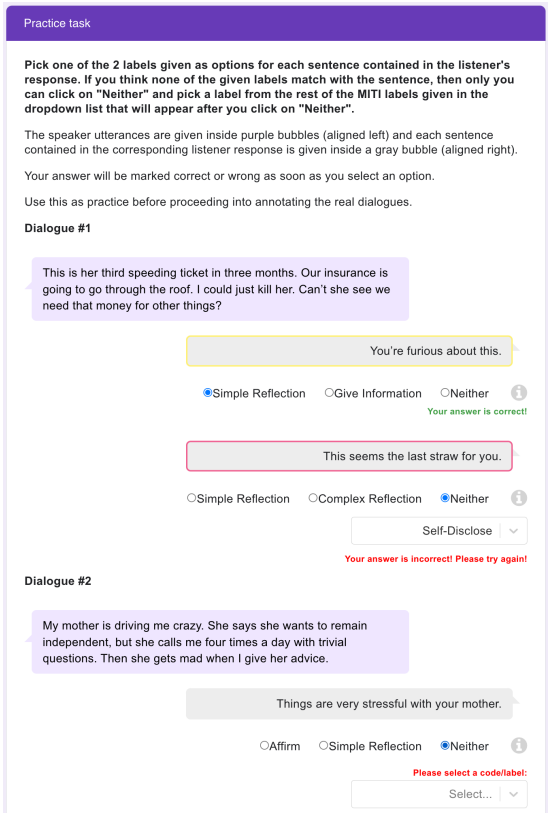

(c) The practice task

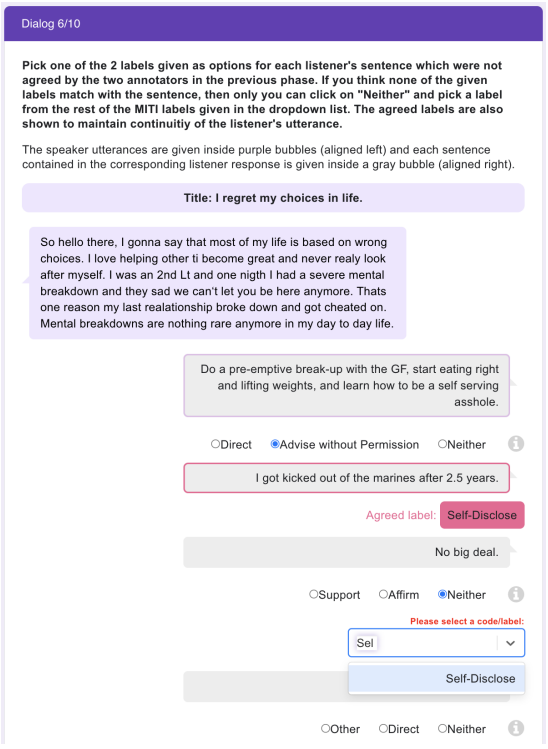

(d) The task interface for resolving labels

Figure 4: User interfaces of the second stage of the MITI annotation experiment.

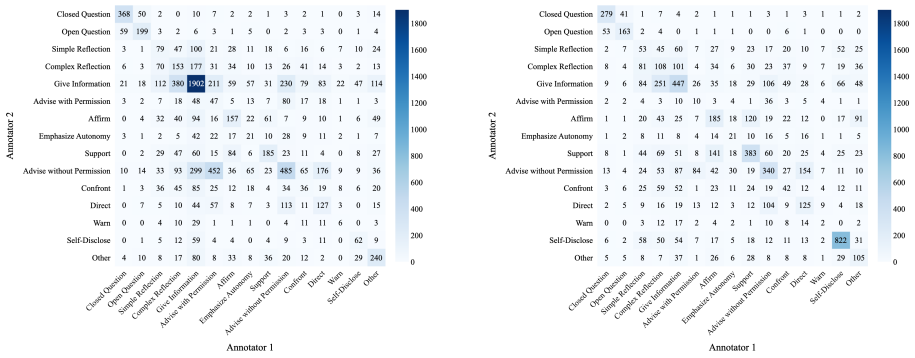

(a) CounselChat dataset (b) RED dataset

Figure 5: Confusion matrices between the two annotators for responses in the CounselChat and RED datasets during stage 1 of the annotation process.

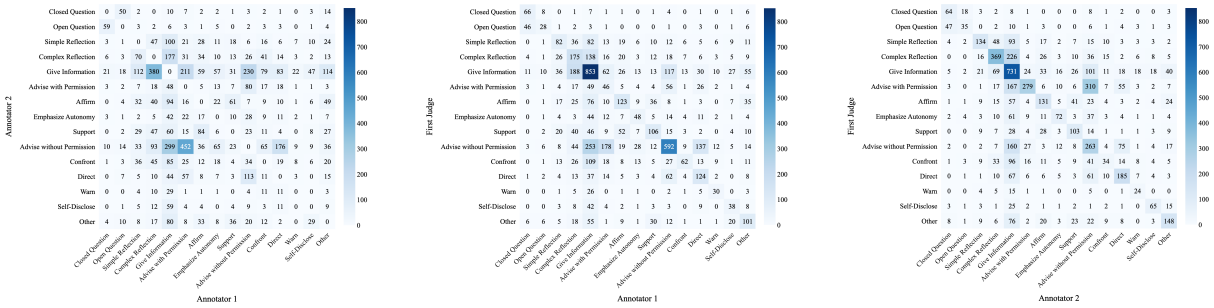

(a) Annotator 1 vs. Annotator 2 (b) Annotator 1 vs. Judge (c) Annotator 2 vs. Judge

CounselChat dataset (For stage 1 unresolved labels)

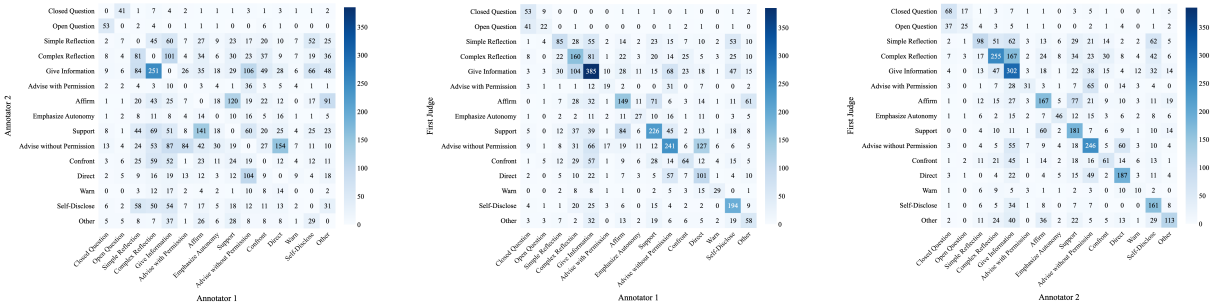

(d) Annotator 1 vs. Annotator 2 (e) Annotator 1 vs. Judge (f) Annotator 2 vs. Judge

RED dataset (For stage 1 unresolved labels)

Figure 6: Confusion matrices between the two annotators for sentence for which the label was unresolved in stage 1 and between each of these annotators and the judge in stage 2 of the annotation process. From the second and third confusion matrices corresponding to each dataset, it could seen how the judge's annotations aligned with annotations from each annotator from stage 1.

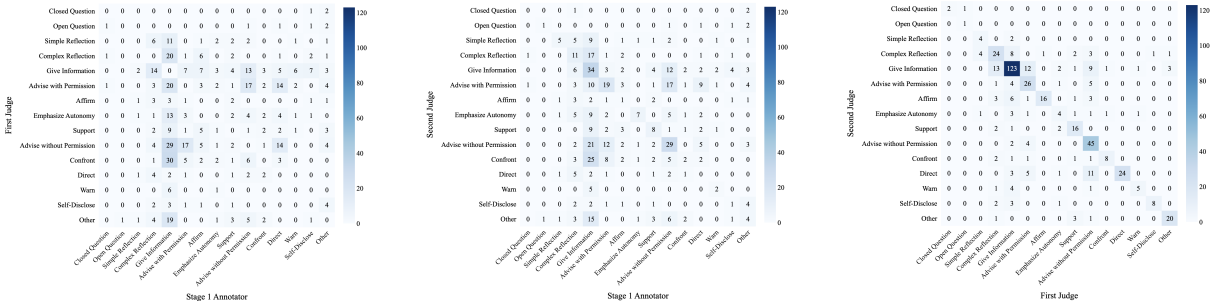

(a) Stage 1 annotator vs. First judge (b) Stage 1 annotator vs. Second judge (c) First judge vs. Second judge

CounselChat dataset

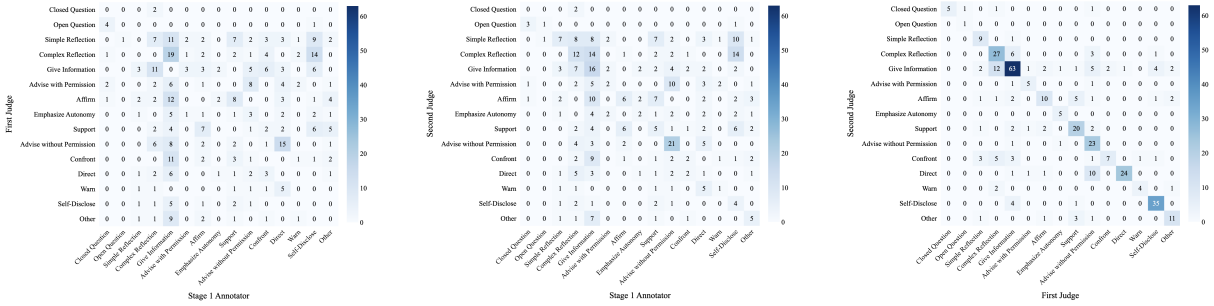

(d) Stage 1 annotator vs. First judge (e) Stage 1 annotator vs. Second judge (f) First judge vs. Second judge

RED dataset

Figure 7: Confusion matrices between different annotators for sentences which were still unresolved after stage 2 that contained at least one annotation from a poorly performed worker. It could be observed that the second judge's annotations in stage 3 aligned mostly with the first judge's annotations in stage 2.

CCTC: A Cross-Sentence Chinese Text Correction Dataset for Native Speakers

Baoxin Wang[1,2], **Xingyi Duan**[2], **Dayong Wu**[2,3], **Wanxiang Che**[1], **Zhigang Chen**[2,4], **Guoping Hu**[2]
[1]Research Center for SCIR, Harbin Institute of Technology, Harbin, China
[2]State Key Laboratory of Cognitive Intelligence, iFLYTEK Research, China
[3]iFLYTEK AI Research (Hebei), Langfang, China
[4]Jilin Kexun Information Technology Co., Ltd., Changchun, China
{bxwang2,xyduan,dywu2,zgchen,gphu}@iflytek.com
car@ir.hit.edu.cn

Abstract

The Chinese text correction (CTC) focuses on detecting and correcting Chinese spelling errors and grammatical errors. Most existing datasets of Chinese spelling check (CSC) and Chinese grammatical error correction (GEC) are focused on a single sentence written by Chinese-as-a-second-language (CSL) learners. We find that errors caused by native speakers differ significantly from those produced by non-native speakers. These differences make it inappropriate to use the existing test sets directly to evaluate text correction systems for native speakers. Some errors also require the cross-sentence information to be identified and corrected. In this paper, we propose a cross-sentence Chinese text correction dataset for native speakers. Concretely, we manually annotated 1,500 texts written by native speakers. The dataset consists of 30,811 sentences and more than 1,000,000 Chinese characters. It contains four types of errors: spelling errors, redundant words, missing words, and word ordering errors. We also test some state-of-the-art models on the dataset. The experimental results show that even the model with the best performance is 20 points lower than humans, which indicates that there is still much room for improvement. We hope that the new dataset can fill the gap in cross-sentence text correction for native Chinese speakers.

1 Introduction

Chinese text correction (CTC) aims at detecting and correcting errors in Chinese text. Text correction has important applications in the domain of education, journalism, and publishing. For many native Chinese speakers, such as journalists, writers, and bloggers, a text correction system for native Chinese speakers will greatly improve the efficiency of their proofreading. In the field of NLP, Chinese text corrections usually includes two tasks: Chinese spelling check (CSC) (Hong et al., 2019; Cheng et al., 2020; Wang et al., 2021) and Chinese grammatical error correction (GEC) (Yuan and Briscoe, 2016; Omelianchuk et al., 2020; Wang et al., 2020).

Figure 1: Comparison between the errors caused by native and non-native speakers. The non-native examples are from CGED 2018, and the native examples are from CCTC.

The existing CSC and Chinese GEC test sets (Tseng et al., 2015; Rao et al., 2018; Zhao et al., 2018) are mainly generated from essays written by Chinese-as-a-second-language (CSL) learners. The essays written by CSL learners are significantly different from those written by native Chinese speakers. Specifically, essays written by CSL learners usually contain more errors and are more likely to make mistakes in the misuse of words. In contrast, texts produced by native speakers contain sparser errors and typically make mistakes that are caused by oversight. These significant differences prevent researchers from using the existing test sets directly to evaluate text correction systems for native speakers.

Proceedings of the 29th International Conference on Computational Linguistics, pages 3331–3341
October 12–17, 2022.

Figure 1 shows the errors made by CSL learners and native speakers, respectively. We can see the CSL learners make some mistakes that are obvious to native speakers. The word "爱情" usually refers to the love between a couple, while "关爱" indicates the love of an elder for a younger child. In Chinese, these two words are not interchangeable. However, For CSL learners, it is easy to mistakenly write "关爱" as "爱情" because they can both be translated into "love" in English. Similarly, the words "利益" and "好处" can both be translated into "benefit" in English, but the word "利益" cannot be used with "健康" (health) in Chinese. Native speakers will not make these mistakes. For native Chinese speakers, the most common errors are caused by oversight, which the writers themselves are capable of correcting. For example, the misspelling of "信息"(information) as "信心" (confidence) is due to the similarity of the Pinyin for *xinxi* and *xinxin*, respectively. Besides, the test sets for non-native speakers, such as CGED (Rao et al., 2018), and SIGHAN (Tseng et al., 2015) tend to write simpler sentences with limited topics. In contrast, the texts written by native speakers tend to have complicated sentences with various topics.

Moreover, the existing datasets of CSC and GEC are mainly for sentence-level correction. However, some errors usually need to be corrected via the cross-sentence information (Chollampatt et al., 2019; Yuan and Bryant, 2021). For example, in Figure 2, it is difficult to see what is wrong with each sentence individually. According to the previous sentences, we know that the word "蜘蛛" (spider) should be corrected as "红蜘蛛" (red spider).

To better evaluate the text correction system's performance on document-level texts produced by native speakers, we propose a new dataset CCTC (**C**ross-Sentence **C**hinese **T**ext **C**orrection). Since every Chinese character may be erroneous, the scale of annotation is large. Without any auxiliary hints, the annotators will be prone to miss the errors. Therefore, we give the annotators some hints about the position and type of errors produced by several CSC and GEC systems. We first annotate all the sentences from 200 documents and find only 11.4% sentences with errors. Errors in sentences with candidate errors account for more than 90% of all errors. In order to maximize the diversity of topics and increase the number of errors in the dataset, we only annotate the sentences with error candidates for another 1,300 documents. Concretely, we annotate 1,500 texts from the Internet, and the annotated text includes a total of 30,811 sentences and more than 1 million Chinese characters.

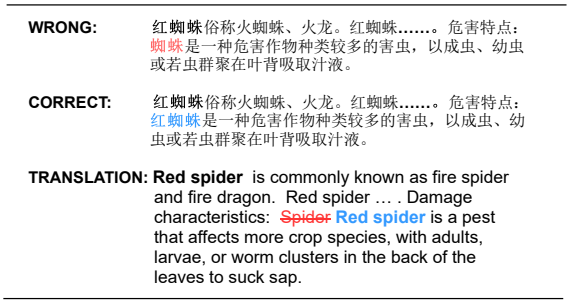

Figure 2: An example for cross-sentence text correction.

We utilize several types of state-of-the-art models for experiments and analyses on our dataset. We also evaluate the performance of native speakers on CCTC. The experimental results show that even the model with the best performance is still 20 points worse than the human, which indicates that there is still much room for improvement.

To summarize, our contributions are as follows:

- We propose a new Chinese text correction dataset, which can be used to evaluate text correction systems for native speakers better.
- Our dataset is based on document-level text. We have done some experiments and analyses for cross-sentence errors, which we hope will be helpful for subsequent studies of cross-sentence text correction.
- We systematically compare our dataset with other CSC and GEC datasets and test four state-of-the-art models on the new dataset.

We hope that CCTC will contribute towards the development of cross-sentence Chinese text correction for native speakers. Our datasets are publicly available at https://github.com/destwang/CTCResources.

2 Existing Datasets

The Chinese text correction related datasets mainly include Chinese spelling check (CSC) and grammatical error correction (GEC). Statistics information is shown in Table 1, and the features of these datasets are shown in Appendix.

2.1 English GEC Datasets

CoNLL14 The test set (Ng et al., 2014) consists of essays written by English as a Second Language

Datasets	# sents	Avg. Sent. Length	Avg. Doc. Length	Err. Sent. (%)	Sent-K	# tokens	Language	Task
CoNLL 2014	1,312	22.9	-	75.8	0.25	30,045	En	GEC
JFLEG	747	18.9	-	86.4	0.53	14,118	En	GEC
CWEB-S	2,864	23.9	-	24.5	0.39	68,450	En	GEC
CWEB-G	3,981	20.3	-	25.6	0.44	80,814	En	GEC
SIGHAN 2015	1,100	30.5	-	50.0	-	33,550	Zh	CSC
OCR Text	1,000	10.2	-	100.0	-	10,198	Zh	CSC
CGED 2018	3,549	39.6	-	56.0	-	140,655	Zh	GEC
NLPCC 2018 GEC	2,000	29	-	99.2	-	59,325	Zh	GEC
CCTC-Train	12,689	41.9	818.6	9.8	0.76	532,088	Zh	CTC
CCTC-W	14,338	38.8	856.6	9.4	0.72	556,767	Zh	CTC
CCTC-H	3,784	41.4	784.2	11.4	0.78	156,836	Zh	CTC

Table 1: Statistics of datasets. For datasets CGED 2018, NLPCC 2018 GEC and SIGHAN 2015, the statistics here are about their test sets. All test sets are sentence-level except for our dataset CCTC. Here, tokens mean the subwords obtained after tokenizing of BERT, which are mainly individual Chinese characters for Chinese. Sent-K is Cohen's Kappa at sentence level. CCTC-H means a high-quality test set, and CCTC-W means a test dataset which contains a wider range of documents.

(ESL) learners from the National University of Singapore, which are annotated for grammatical errors by two native English speakers.

JFLEG The JFLEG corpus (Napoles et al., 2017) consists of sentences written by English language learners for the TOEFL exam. The texts have been corrected for grammatical errors and fluency.

CWEB This dataset (Flachs et al., 2020) is designed to annotate English web text, which corresponds to a dataset containing both native and non-native speakers.

CWEB is the closest to our proposed dataset among the known datasets. There are three main differences: (i) our dataset is document-level, while CWEB is sentence-level; (ii) our data only focus on the texts written by native speakers; (iii) our proposed dataset is designed for Chinese.

2.2 CSC Datasets

SIGHAN 2015 The text of SIGHAN 2015 (Tseng et al., 2015) is collected from the essay section of the computer-based Test of Chinese as a Foreign Language (TOCFL). Thus, the spelling errors are mainly caused by CSL Learners. SIGHAN 2015 is based on the sentence, and the rate of the erroneous sentences is manually adjusted to be higher than the original text.

OCR Text The dataset is produced from OCR results of Chinese subtitles in videos (Hong et al., 2019). Therefore, these sentences are from native Chinese speakers, but these errors are automatically generated by the OCR method and not caused by human writing.

2.3 Chinese GEC Datasets

CGED 2018 The corpora used in CGED 2018 (Rao et al., 2018) are taken from the writing section of the HSK (*Hanyu Shuiping Kaoshi*, Pinyin of "A test of Chinese level"). The grammatical errors are also produced by non-native speakers. There are four kinds of errors, which are spelling errors, redundant words, missing words, and word ordering errors.

NLPCC 2018 GEC The training data (Zhao et al., 2018) is mainly collected from Lang-8. The test data is extracted from the PKU Chinese Learner Corpus, which is constructed by the Department of Chinese Language and Literature, Peking University.

MuCGEC The dataset consists of 7,063 sentences collected from CSL learner sources. MuCGEC (Zhang et al., 2022) is a multi-reference multi-source evaluation dataset for Chinese Grammatical Error Correction.

In contrast to CGED 2018, NLPCC 2018 GEC and MuCGEC datasets, CCTC is based on document-level texts written by native speakers.

3 CCTC Dataset

We construct a new cross-sentence Chinese text correction dataset for native speakers. We extract the raw text from WuDaoCorpora (Yuan et al., 2021), which mainly includes news, blogs, and some popular science articles. We pre-process the collected documents, remove personal information, advertisements, and noisy articles, then sample 1,500 documents for annotation. We take 100 of these documents for verification. We can determine by the author's information that all the 100 documents

Candidate Methods	# sents	# err. sents
BERT-CSC	3,905	2,192
BERT-GEC	2,213	1,404
BERT-CGED	2,083	356
Others	2,734	40
Total	10,935	3,992

Table 2: Statistics of different candidate generation methods. *# sents* is the number of candidates generated by these methods and *# err. sents* is the number of real erroneous sentences. *Others* indicates the sentences which are labeled without error candidates in CCTC-H.

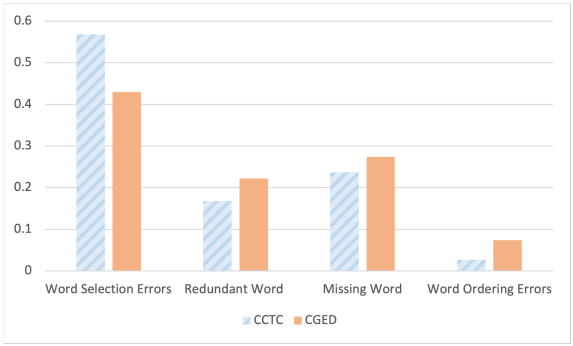

Figure 3: The rate of different error types.

are written by native Chinese speakers, which illustrates that almost all of the documents are written by native Chinese speakers. Table 1 shows the statistical information.

Candidates Generation To facilitate manual annotation and reduce error omission, we utilize several different models to generate error candidates. Specifically, we select three different kinds of models as follows. The detailed information of the training set will be described in the next section.

- BERT-CSC: We train a BERT-based (Devlin et al., 2019) Chinese spelling check model via the pseudo-data similar to Cheng et al. (2020).
- BERT-GEC: We replace, insert, delete and shuffle some tokens randomly to construct GEC pseudo-data and train a BERT-based sequence labeling model.
- BERT-CGED: We train a BERT-based sequence labeling model using the CGED training dataset.

To cover as many errors as possible, we lower the thresholds of the three models. In this way, these models will generate more candidates to find out the erroneous parts of the documents.

Annotation Following Rao et al. (2018), errors are divided into four types: spelling errors (word selection errors), redundant words, missing words, and word ordering errors. The data are annotated by five annotators, with an average of about 120 hours and 2K sentences each. Our annotators annotate the dataset on an annotation tool prepared in advance. We pay our annotators appropriately according to the number of annotated sentences.

We firstly annotate 3,784 sentences from 200 documents, including sentences with error candidates and sentences without candidates. After annotating, we find that there are only 431 sentences with errors. Errors in candidate sentences account for more than 90% of all errors. In order to maximize the diversity of topics and increase the number of errors in the dataset, we only annotate the sentences with error candidates for another 1,300 documents. We name the dataset with 200 annotated documents as **CCTC-H**, which means a high-quality dataset. The remaining 1,300 documents are divided into two parts, 650 of which are used as the training set and the other 650 documents as the **CCTC-W**, which means this test dataset contains a wider range of documents. To conclude, we annotate 1,500 documents from the Internet, and the annotated texts include a total of 30,811 sentences and more than 1 million Chinese characters. The detailed statistics of different candidate generation methods are shown in Table 2.

In order to ensure the quality of the annotated data, we take 500 sentences from the training set, validation set, and test set, respectively, and annotate these sentences without candidate errors. Similar to Flachs et al. (2020), annotator agreement is calculated at the sentence level using Cohen's Kappa. Kappa is 0.76, 0.72, and 0.78 for the CCTC-Train, CCTC-W, and CCTC-H, respectively, showing that our dataset has a higher agreement than the previous dataset.

Dataset Analysis Table 3 shows examples of the four types of errors. Figure 3 shows the rate of sentences corresponding to the four error types. We can see that Chinese spelling errors (word selection errors) are the most common in documents written by native speakers, accounting for about 60% of the total. Word ordering errors have the least percentage of all errors. For texts written by non-native speakers from CGED, redundant words and missing words occur at a relatively greater rate than texts written by native speakers. The occurrence of

Error Type	Example sentence	Translation
Spelling Errors	进 入 大 学 ， 就 是 进 入 一 个 新 的 环境，结出（接触）新的人，你的所有过去对于他们来说是一张白纸。	Entering college means entering a new environment, you will ~~bear~~ (meet) new people, and all your past is a blank sheet of paper to them.
Redundant Words	突然有一天，一个女人来看~~来看~~孩子。	Suddenly one day, a woman came to see ~~came to see~~ the child.
Missing Words	今天要讲（的）是他在一年时间里面的教师生涯。	What today is going (to) be talking about is his career as a teacher inside a year.
Word Ordering Errors	一般室内环境采用200系列材质即可，而室外需~~环境~~（环境需）使用304材质。	General indoor environment needs to use 200 series material, while outdoor ~~needs environment~~ (environment needs) to use 304 series material.

Table 3: Examples of different error types caused by native speakers.

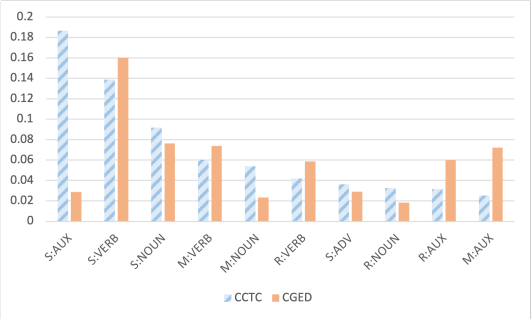

Figure 4: The rate of different error types with POS tagging. We count the multi-token errors according to the POS tags of multi-token. If the multi-token errors can be segmented into k words, the count of each type will increase by $1/k$. (S: Spelling errors, M: missing words, R: redundant words)

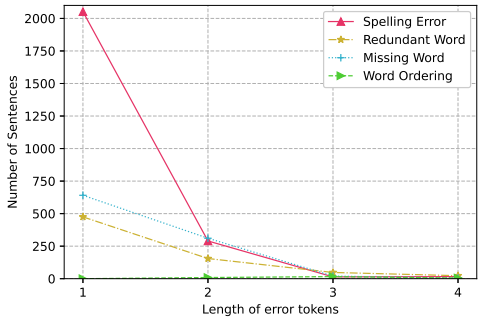

Figure 5: The length of error span.

word ordering errors is rare for native speakers and somewhat more frequent for non-native speakers.

To better analyze the difference between errors made by native and non-native Chinese speakers, we perform statistical lexical analysis for each error type. In this paper, we use LTP (Che et al., 2010) for the Part-of-Speech (POS) tagging of the text. The statistical results are shown in Figure 4. We find that the most common mistake made by native speakers is the misuse of auxiliaries. In contrast, non-native speakers tend to write a sentence with redundant or missing auxiliaries.

We count the length of the error span, which can be seen in Figure 5. Except for the word ordering errors, the errors with one token are in the majority. The decline in the percentage of spelling errors of two consecutive tokens is faster than the percentage for redundant and missing words. Errors of more than three consecutive tokens are rare.

We perform a manual statistical analysis of the dataset and find that 68% of errors are caused by oversight, such as spelling errors caused by the Pinyin Input method. The word “接触” (meet) may be incorrectly entered as “结出” (bear) due to similar pronunciation as shown in Table 3. This type of error is varied, making this type of error more difficult to correct. The remaining errors are mainly due to misuse of some words with similar semantics or method of use, such as the auxiliaries “的” and “地”. In Chinese, “的” is usually used as a suffix of adjective and “地” is used as a suffix of adverb, and they are pronounced the same, so these two words are often misused in Chinese.

We analyze the spelling errors more specifically. The spelling errors can be divided into the following five types: misuse of words, single Chinese character error in a word, pronoun errors, auxiliary errors, and other single Chinese character errors, accounting for 28%, 23%, 8%, 30%, and 11%, respectively.

4 Experiments

4.1 Training Dataset

There are no training datasets specifically annotated for errors caused by native speakers before.

Dataset	# sents	err. sents (%)
CGED	44,754	94.7
NLPCC GEC	1,200,000	89.8
SIGHAN	281,381	100.0
Pseudo-data	3,000,000	99.6

Table 4: Statistics of training dataset.

In this paper, in addition to our proposed training set, we use training data from multiple sources, including CGED (Rao et al., 2018), SIGHAN (Tseng et al., 2015), and NLPCC 2018 GEC dataset (Zhao et al., 2018). For the CGED data[1], we use CGED training data from 2014 to 2016, totaling about 45K sentences. For NLPCC dataset[2], there are multiple correction sentences for each sentence. We randomly select part of the correction sentences as our training set. For SIGHAN, we use the training data of SIGHAN, as well as the automatically generated corpus (Wang et al., 2018). Besides, we also use our training set of CCTC to train these models. For the GECToR model, we only use CCTC to fine-tune after the pseudo-data training the same as Omelianchuk et al. (2020).

As mentioned above, native speakers make a wider variety of errors, so we use heuristics to construct pseudo-data in the hope that we can cover as many types of errors as possible. We construct a large-scale pseudo-data using Chinese Wikipedia. The pseudo-data generation method for GEC is similar to Zhao et al. (2019), which randomly delete, add, replace, and shuffle the tokens. To better check the Chinese spelling errors, for the replacement operation, 80% of the tokens are from the confusion set provided by Wu et al. (2013) and 20% of the tokens are from the corpus. The pseudo-data of CSC are generated by the same replacement operation. Table 4 shows the statistics of the training data.

4.2 Models

We evaluate performance on our proposed dataset using four state-of-the-art approaches to CSC or GEC. The specific models are described as follows.

- SpellGCN (Cheng et al., 2020): This model incorporates phonological and visual similarity knowledge into BERT via a specialized graph convolutional network.
- ResBERT (Wang et al., 2020): ResBERT is the state-of-the-art model in CGED competition, by adding ResNet to the BERT model to achieve better performance.
- GECToR (Omelianchuk et al., 2020): GECToR achieves the correction of errors such as redundant words, missing words, and spelling errors by the BERT model.
- CopyNet (Zhao et al., 2019): CopyNet is a transformer-based seq2seq model, which can pay more attention to the grammatical errors through the copy mechanism.

4.3 Metrics

In the previous works, GEC systems are usually evaluated using F0.5-score based on MaxMatch (Dahlmeier and Ng, 2012) since that the precision of the GEC system is more critical for ESL or CSL learners. On the contrary, recall is usually more important than precision for native Chinese speakers because most errors are caused due to oversights. They can make correct judgments about most grammatical errors by themselves. Therefore, we use the F2-score to evaluate the performance on the CCTC dataset. The specific equation is as follows:

$$\text{F2-score} = \frac{5 \times \text{Precision} \times \text{Recall}}{4 \times \text{Precision} + \text{Recall}} \quad (1)$$

Given that native speakers can generally make correct judgments by themselves, it is also essential for them to detect the position of errors as well. Regarding CGED (Rao et al., 2018), SIGHAN (Tseng et al., 2015), and NLPCC (Zhao et al., 2018), we perform three kinds of evaluation, namely sentence-level, position-level, and correction-level evaluation. The sentence-level evaluation determines whether there is an error in a sentence, while the position-level evaluation needs to label the error position correctly. For the correction-level evaluation, we statistically score the systems by the error position, error type, and correction results similar to Rao et al. (2018). The difference is that we use F2-score because the recall for native speakers is usually more important.

4.4 Experimental Settings

We use the RoBERTa-wwm (Cui et al., 2019) as the base models of SpellGCN, GECToR, and ResBERT. The training hyperparameters of SpellGCN and CopyNet are kept consistent with Cheng et al.

[1]http://www.cged.tech
[2]http://tcci.ccf.org.cn/conference/2018/taskdata.php

Test Set	Train Set	Model	Sentence-Level			Position-Level			Correction-level		
			P	R	F2	P	R	F2	P	R	F2
CCTC-W	SIGHAN	SpellGCN	23.59	41.66	36.12	10.99	21.88	18.26	9.49	18.89	15.77
	CGED	ResBERT	15.96	73.14	42.61	6.50	33.50	18.30	-	-	-
	NLPCC	GECToR	27.90	30.90	30.25	8.38	10.57	10.04	7.29	9.20	8.74
	Pseudo-data	CopyNet	14.04	**78.75**	40.98	1.59	16.22	5.72	0.90	9.14	3.22
		ResBERT	26.13	40.61	36.56	11.34	20.01	17.36	-	-	-
		GECToR	26.29	44.39	39.02	11.61	22.25	18.80	8.17	15.66	13.24
	CCTC-Train	SpellGCN	**55.61**	43.48	**45.46**	**38.96**	31.44	**32.71**	**35.19**	**28.40**	**29.54**
		ResBERT	17.62	49.89	36.51	13.38	**37.65**	27.63	-	-	-
		GECToR	43.13	45.88	45.30	23.37	26.26	25.63	20.36	22.87	22.32
CCTC-H	SIGHAN	SpellGCN	26.27	36.71	34.01	11.51	18.10	16.24	10.81	17.00	15.26
	CGED	ResBERT	19.24	64.44	43.83	7.59	**29.07**	18.56	-	-	-
	NLPCC	GECToR	32.55	29.25	29.86	9.58	10.05	9.96	8.07	12.25	11.10
	Pseudo-data	CopyNet	18.55	**79.73**	**48.04**	2.13	17.37	7.13	1.10	8.96	3.68
		ResBERT	26.02	33.08	31.37	9.65	14.26	13.02	-	-	-
		GECToR	27.82	37.67	35.18	10.84	16.45	14.91	8.07	12.25	11.10
	CCTC-Train	SpellGCN	**61.33**	35.80	39.05	**40.44**	23.68	**25.82**	**36.07**	**21.12**	**23.03**
		ResBERT	25.86	39.49	35.72	16.84	25.93	23.40	-	-	-
		GECToR	49.87	33.86	36.19	24.66	17.28	18.38	22.60	15.84	16.85

Table 5: Experimental Result. For the GECToR model, we use CCTC-Train to fine-tune after the pseudo-data training the same as Omelianchuk et al. (2020).

(2020) and Zhao et al. (2019) respectively. For ResBERT, we use the BIO encoding (Kim et al., 2004) the same as Wang et al. (2020). We fine-tune the models using the sentences with errors in CCTC-train.

For CopyNet and GECToR, they will generate a corrected sentence. To evaluate the performance of position-level detection for the two models, we use the Levenshtein[3] distance to convert the sentence pairs into the corresponding error types. Concretely, Levenshtein distance can generate three types of operations: delete, insert and replace, which correspond to redundant words, missing words, and spelling errors. Then we convert the adjacent insertion and deletion operations into word ordering errors. In this way, we can evaluate the detection performance of the two models. Since spelling errors accounted for the highest percentage of all errors, we also test directly using the SpellGCN model, which can only correct the spelling errors.

4.5 Experimental Result

The experimental results are shown in Table 5. The overall performance of the models after training with the CCTC-Train is better than other datasets. CopyNet trained with pseudo-data achieves the best performance for sentence-level detection on CCTC-H. For all the models without CCTC-train, ResBERT with CGED dataset achieves the best results on position-level detection. However, ResBERT only detects the errors, but it cannot correct the sentence. Surprisingly, SpellGCN performs best for correction. This may be because spelling errors account for most errors, and SpellGCN is better able to correct them using phonological and visual similarity knowledge. ResBERT trained with CGED dataset achieves better performance than the model using pseudo-data. We find that ResBERT with CGED is more effective in detecting auxiliary errors such as the misuse of "的" and "地", which account for a relatively large proportion of all errors.

Besides, we can see that the precision of each model is higher overall on CCTC-H than on CCTC-W, and the recall is lower. This may be because all sentences in CCTC-H are labeled, and the coverage of errors is greater.

Model	Sentence-Level			Position-Level		
	P	R	F2	P	R	F2
SpellGCN	75.0	33.3	37.5	42.1	20.5	22.9
GECToR	71.4	41.7	45.5	34.8	20.5	22.4
ResBERT	48.0	47.4	47.5	31.7	33.8	33.4
Human	**85.4**	**67.3**	**70.3**	**61.7**	**56.0**	**57.1**

Table 6: Experimental results for comparison with humans. The results of humans are the average results of two untrained native speakers. All the models are trained with CCTC-Train dataset.

4.6 Analysis

To better evaluate the effectiveness of these models, we test the performance of humans for text correc-

[3]https://github.com/ztane/python-Levenshtein

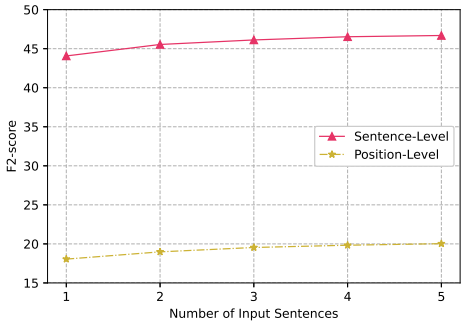

Figure 6: Experimental results for different input sequence length in inference stage, the model is a single-sentence trained ResBERT model.

tion. The low error density in the actual text makes it very difficult for humans to correct texts. Thus, we take 200 sentences from the CCTC-H dataset and adjust the erroneous sentences to about 50%. Two untrained native speakers are asked to correct these 200 sentences. We want to know what performance the native Chinese speaker can achieve. The corresponding experimental results are shown in Table 6. More detailed results are in the Appendix.

After increasing the error density, the performance of almost all the models improves. Human performs much better than these models. Even the model with the best results is 20 points worse than the human, indicating that the models still have much room for improvement.

Also, with the human test, native speakers often miss errors without being informed of the error position in advance, even though we have increased the error rate to about 50%. For example, in Figure 7, an annotator missed the error "有限" (limited) because this word also appears frequently. When we point out this position, native speakers can easily correct the error.

5 Cross-Sentence Errors

We randomly analyze 100 errors and find that cross-sentence information is necessary for only 11% of the errors. However, cross-sentence information can be helpful for 38% of errors, such as when the corrected word appears in context.

To test the help of cross-sentence information for Chinese text correction, we try a simple cross-sentence correction method, which increases the length of the input sequence. We vary from single-sentence correction to multi-sentence correction, and Figure 6 shows the experimental results. From

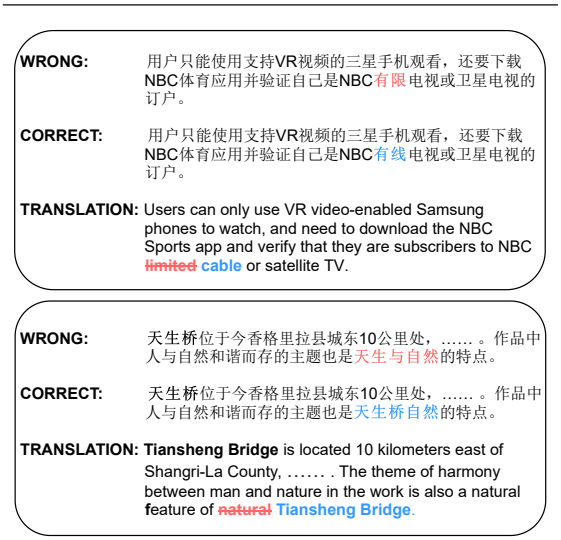

Figure 7: Examples of CCTC. The above sentence is an example of failure to correct during human testing, and the below one is an example for mis-correction by SpellGCN.

the experimental results, we can see that for a trained model, the performance of the model increases as the input sequence length grows. This also shows that the cross-sentence information is helpful for Chinese text correction.

The models often mis-correct some low-frequency words due to the lack of context of a document. In Figure 7, the model mistakenly modify "天生桥自然" (Tiansheng Bridge) as "天生与自然" (Natural). In fact, the word "天生桥" has appeared many times in the context of the document. If we could better use the cross-sentence contextual information, it would help better with the correction. Based on this, we do not simply split the document into individual sentences but keep the complete cross-sentence information. We hope it will be helpful for subsequent studies of cross-sentence text correction.

6 Conclusion

In this paper, we propose a novel cross-sentence Chinese text correction dataset for native speakers. Concretely, we manually annotated 1,500 Chinese texts written by native speakers collected from the Internet. The new dataset consists of 30,811 sentences and more than 1,000,000 Chinese characters. It contains spelling errors, redundant words, missing words, and word ordering errors. CSC and GEC systems developed for native speakers can be better evaluated on CCTC than the previous

datasets. We also test some state-of-the-art models on the dataset. The experimental results show that even the model with the best performance is still 20 points worse than the human, which indicates that there is still much room for improvement.

Acknowledgements

This work was supported by the National Key R&D Program of China via grant 2020AAA0106501 and the National Natural Science Foundation of China (NSFC) via grant 61976072 and 62176078. I would like to thank the anonymous reviewers and Xiaoxue Wang for their insightful comments and suggestions.

References

Wanxiang Che, Zhenghua Li, and Ting Liu. 2010. LTP: A Chinese language technology platform. In *Coling 2010: Demonstrations*, pages 13–16, Beijing, China. Coling 2010 Organizing Committee.

Xingyi Cheng, Weidi Xu, Kunlong Chen, Shaohua Jiang, Feng Wang, Taifeng Wang, Wei Chu, and Yuan Qi. 2020. SpellGCN: Incorporating phonological and visual similarities into language models for Chinese spelling check. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 871–881, Online. Association for Computational Linguistics.

Shamil Chollampatt, Weiqi Wang, and Hwee Tou Ng. 2019. Cross-sentence grammatical error correction. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 435–445, Florence, Italy. Association for Computational Linguistics.

Yiming Cui, Wanxiang Che, Ting Liu, Bing Qin, Ziqing Yang, Shijin Wang, and Guoping Hu. 2019. Pre-training with whole word masking for chinese bert. *arXiv preprint arXiv:1906.08101*.

Daniel Dahlmeier and Hwee Tou Ng. 2012. Better evaluation for grammatical error correction. In *Proceedings of the 2012 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 568–572, Montréal, Canada. Association for Computational Linguistics.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, Minneapolis, Minnesota. Association for Computational Linguistics.

Simon Flachs, Ophélie Lacroix, Helen Yannakoudakis, Marek Rei, and Anders Søgaard. 2020. Grammatical error correction in low error density domains: A new benchmark and analyses. *arXiv preprint arXiv:2010.07574*.

Yuzhong Hong, Xianguo Yu, Neng He, Nan Liu, and Junhui Liu. 2019. FASPell: A fast, adaptable, simple, powerful Chinese spell checker based on DAE-decoder paradigm. In *Proceedings of the 5th Workshop on Noisy User-generated Text (W-NUT 2019)*, pages 160–169, Hong Kong, China. Association for Computational Linguistics.

Jin-Dong Kim, Tomoko Ohta, Yoshimasa Tsuruoka, Yuka Tateisi, and Nigel Collier. 2004. Introduction to the bio-entity recognition task at jnlpba. In *Proceedings of the international joint workshop on natural language processing in biomedicine and its applications*, pages 70–75. Citeseer.

Courtney Napoles, Keisuke Sakaguchi, and Joel Tetreault. 2017. Jfleg: A fluency corpus and benchmark for grammatical error correction. *arXiv preprint arXiv:1702.04066*.

Hwee Tou Ng, Siew Mei Wu, Ted Briscoe, Christian Hadiwinoto, Raymond Hendy Susanto, and Christopher Bryant. 2014. The conll-2014 shared task on grammatical error correction. In *Proceedings of the Eighteenth Conference on Computational Natural Language Learning: Shared Task*, pages 1–14.

Kostiantyn Omelianchuk, Vitaliy Atrasevych, Artem Chernodub, and Oleksandr Skurzhanskyi. 2020. GECToR – grammatical error correction: Tag, not rewrite. In *Proceedings of the Fifteenth Workshop on Innovative Use of NLP for Building Educational Applications*, pages 163–170. Association for Computational Linguistics.

Gaoqi Rao, Qi Gong, Baolin Zhang, and Endong Xun. 2018. Overview of nlptea-2018 share task chinese grammatical error diagnosis. In *Proceedings of the 5th Workshop on Natural Language Processing Techniques for Educational Applications*, pages 42–51.

Yuen-Hsien Tseng, Lung-Hao Lee, Li-Ping Chang, and Hsin-Hsi Chen. 2015. Introduction to SIGHAN 2015 bake-off for Chinese spelling check. In *Proceedings of the Eighth SIGHAN Workshop on Chinese Language Processing*, pages 32–37, Beijing, China. Association for Computational Linguistics.

Baoxin Wang, Wanxiang Che, Dayong Wu, Shijin Wang, Guoping Hu, and Ting Liu. 2021. Dynamic connected networks for Chinese spelling check. In *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*, pages 2437–2446, Online. Association for Computational Linguistics.

Dingmin Wang, Yan Song, Jing Li, Jialong Han, and Haisong Zhang. 2018. A hybrid approach to automatic corpus generation for Chinese spelling check. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages

2517–2527, Brussels, Belgium. Association for Computational Linguistics.

Shaolei Wang, Baoxin Wang, Jiefu Gong, Zhongyuan Wang, Xiao Hu, Xingyi Duan, Zizhuo Shen, Gang Yue, Ruiji Fu, Dayong Wu, Wanxiang Che, Shijin Wang, Guoping Hu, and Ting Liu. 2020. Combining ResNet and transformer for Chinese grammatical error diagnosis. In *Proceedings of the 6th Workshop on Natural Language Processing Techniques for Educational Applications*, pages 36–43, Suzhou, China. Association for Computational Linguistics.

Shih-Hung Wu, Chao-Lin Liu, and Lung-Hao Lee. 2013. Chinese spelling check evaluation at SIGHAN bake-off 2013. In *Proceedings of the Seventh SIGHAN Workshop on Chinese Language Processing*, pages 35–42, Nagoya, Japan. Asian Federation of Natural Language Processing.

Sha Yuan, Hanyu Zhao, Zhengxiao Du, Ming Ding, Xiao Liu, Yukuo Cen, Xu Zou, Zhilin Yang, and Jie Tang. 2021. Wudaocorpora: A super large-scale chinese corpora for pre-training language models. *AI Open*.

Zheng Yuan and Ted Briscoe. 2016. Grammatical error correction using neural machine translation. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 380–386, San Diego, California. Association for Computational Linguistics.

Zheng Yuan and Christopher Bryant. 2021. Document-level grammatical error correction. In *Proceedings of the 16th Workshop on Innovative Use of NLP for Building Educational Applications*, pages 75–84, Online. Association for Computational Linguistics.

Yue Zhang, Zhenghua Li, Zuyi Bao, Jiacheng Li, Bo Zhang, Chen Li, Fei Huang, and Min Zhang. 2022. MuCGEC: a multi-reference multi-source evaluation dataset for Chinese grammatical error correction. In *Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 3118–3130, Seattle, United States. Association for Computational Linguistics.

Wei Zhao, Liang Wang, Kewei Shen, Ruoyu Jia, and Jingming Liu. 2019. Improving grammatical error correction via pre-training a copy-augmented architecture with unlabeled data. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 156–165, Minneapolis, Minnesota. Association for Computational Linguistics.

Yuanyuan Zhao, Nan Jiang, Weiwei Sun, and Xiaojun Wan. 2018. Overview of the nlpcc 2018 shared task: Grammatical error correction. In *CCF International Conference on Natural Language Processing and Chinese Computing*, pages 439–445. Springer.

Model	Sentence-Level			Position-Level			Correction-Level		
	P	R	F2	P	R	F2	P	R	F2
SpellGCN	75.0	33.3	37.5	42.1	20.5	22.9	39.5	19.2	21.4
GECToR	71.4	41.7	45.5	34.8	20.5	22.4	32.6	19.2	20.9
ResBERT	48.0	47.4	47.5	31.7	33.8	33.4	-	-	-
Human	**85.4**	**67.3**	**70.3**	**61.7**	**56.0**	**57.1**	**52.2**	**46.2**	**47.2**

Table 7: Experimental results for comparison with humans.

Dataset	Native Speakers	Real Errors	Original Distribution	Cross-Sentence	Grammatical Error
CoNLL 2014		✓			✓
JFLEG		✓			✓
CWEB	-	✓	✓		✓
SIGHAN 2015		✓			
OCR Text	✓				
CGED 2018		✓			✓
NLPCC 2018 GEC		✓			✓
CCTC (Ours)	✓	✓	✓	✓	✓

Table 8: The features of different datasets. The CWEB dataset contains sentence produced by both native English speakers and non-native English speakers. In contrast, our dataset CCTC only contains text written by native Chinese speakers.

A Appendix

Table 7 shows the correction-level experimental results for comparison with humans. Table 8 shows the features of different datasets.

RealMedDial: A Real Telemedical Dialogue Dataset Collected from Online Chinese Short-Video Clips

Bo Xu[1], **Hongtong Zhang**[1], **Jian Wang**[1], **Xiaokun Zhang**[1], **Dezhi Hao**[1], **Linlin Zong**[2]*, **Hongfei Lin**[1], **Fenglong Ma**[3]
[1]School of Computer Science and Technology, Dalian University of Technology, China
[2]School of Software, Dalian University of Technology, China
[3]College of Information Sciences and Technology, Pennsylvania State University, USA
{xubo, wangjian, llzong, hflin}@dlut.edu.cn
{dutzht, kp5861213, kun}@mail.dlut.edu.cn, fenglong@psu.edu

Abstract

Intelligent medical services have attracted great research interests for providing automated medical consultation. However, the lack of corpora becomes a main obstacle to related research, particularly data from real scenarios. In this paper, we construct RealMedDial, a Chinese medical dialogue dataset based on real medical consultation. RealMedDial contains 2,637 medical dialogues and 24,255 utterances obtained from Chinese short-video clips of real medical consultations. We collected and annotated a wide range of metadata with respect to medical dialogue including doctor profiles, hospital departments, diseases and symptoms for fine-grained analysis on language usage pattern and clinical diagnosis. We evaluate the performance of medical response generation, department routing and doctor recommendation on RealMedDial. Results show that RealMedDial are applicable to a wide range of NLP tasks with respect to medical dialogue.

1 Introduction

The COVID-19 pandemic has dramatically changed how outpatient care is delivered in healthcare practices. To decrease the risk of transmitting the virus to either patients or healthcare workers within their health practice, providers are deferring or selectively prohibiting in-person visits, but fortunately, they are usually converting in-person visits to telemedicine visits (Mann et al., 2020). During the telemedicine, patients describe their symptoms of suffered diseases and/or adverse reactions of the taking drugs to doctors, while doctors provide medical consultations through online video conferences. Although telemedicine is convenient and timely for disease diagnoses, the continuous growth of telemedicine visits significantly increases the burden and workload of doctors, and meanwhile, the health conditions of remote patients become increasingly difficult to be tracked. Thus, how to relieve the burden of doctors and effectively track the patient's health conditions remains an open research question.

Researchers from related fields are trying to solve this issue by developing medical dialogue systems to serve as virtual doctors, which greatly facilitates users to obtain medical and healthcare information. Recent advances in medical dialogue systems have benefited medical applications such as psychological consultation (Das et al., 2022), elderly care (Keshmiri et al., 2019), and disease pre-diagnosis (Nasreen et al., 2021). To build effective medical dialogue systems, related studies are focusing on optimizing medical dialogue from various aspects, including automatic diagnosis (Wei et al., 2018; Xu et al., 2019), medical information extraction (Zhang et al., 2020), medical slot filling (Shi et al., 2020), and medical conversational summarization (Joshi et al., 2020). Although these researches have improved the performance of medical dialogue, this challenging task is still facing great difficulty in generating effective responses due to the particularity and professionalism of the medical field.

In general, several key challenges have not been thoroughly considered in the current medical dialogue datasets. First, as shown in Table 1, most existing medical dialogue datasets extract the data from online medical or healthcare community, which is **non-real time communication** records between doctors and patients. In fact, such data are more similar to question and answering (Q&A) data, instead of medical dialogue. Besides, the static Q&A data are largely different from real-time medical consultations in language expressions and interaction patterns. In real-time medical consultations, doctors make accurate diagnosis predictions not only based on symptom descriptions from patients, but also according to observations of patient health status and medical examination results,

*Corresponding author

Proceedings of the 29th International Conference on Computational Linguistics, pages 3342–3352
October 12–17, 2022.

Dataset Name	Source	#Dialogues	#Utterances	#Diseases	Department
MZ(Wei et al., 2018)	Online health community	710	-	4	Pediatrics
DX(Xu et al., 2019)	Online health community	527	2,186	5	Pediatrics
CMDD(Lin et al., 2019)	Online health community	2,067	87,005	4	Pediatrics
MIE(Zhang et al., 2020)	Online health community	1,120	18,129	6	Cardiology
MedDG(Liu et al., 2020)	Online health community	17,864	385,951	12	Gastroenterology
COVID-EN(Zhou et al., 2021)	-	603	-	1	COVID
COVID-CN(Zhou et al., 2021)	-	1,088	-	1	COVID
MedDialog-EN(Zeng et al., 2020)	Online health community	257,332	514,664	96	29 Departments
MedDialog-CN(Zeng et al., 2020)	Online health community	3,407,494	11,260,564	172	51 Departments
RealMedDial (Ours)	Online short-video clips	2,637	24,255	55	17 Departments

Table 1: Comparison between our dataset and other existing medical dialogue datasets.

which are usually missing in the current medical dialogue datasets. Thus, it is essential to build a real-time medical dialogue dataset, which can be used for developing workable dialogue systems.

Second, when patients use online health community to ask for help from doctors, they usually input their symptoms as detailed as possible. Then doctors predict possible diagnoses based on patients' inputs. Such a working procedure leads to a common shortage of existing medical dialogue data extracted from online health community, that is, they only have **a few communication rounds or utterances**. For instance, the average number of utterances in a dialogue is only 3.3 in the largest Chinese medical dialogue dataset MedDialog-CN (Zeng et al., 2020). In real-world medical consultations, a doctor seldom makes any decisions just based on limited number of interactions with patients. Therefore, such datasets may be not suitable for training a real medical dialogue model.

Third, doctors, especially domain experts, are usually very busy and do not have enough time to answer online questions frequently. In order to attract more patients to use the health community, the companies have to hire graduate students studying in medical schools as online doctors. They will be paid when replying patients' questions. Compared with experts' replies, the quality of the answers from graduate students is usually not very high in some dialogues. The **low quality** issue of existing datasets also impedes the development and learning of medical dialogue models.

To tackle all the aforementioned limitations, in this paper, we construct a real-time, high-quality, and large-scale medical dialogue dataset named **RealMedDial**, which is extracted from online Chinese short-video clips. In particular, the videos are downloaded from a popular Chinese video-based social media named Kuaishou[1], where many medical physicians record the short-videos when they communicate with their online or offline patients and post them to Kuaishou. Those short-videos are all real-time medical consultations, which are high quality and representative for diagnosing diseases. Moreover, the contents between doctors and patients not only include disease diagnoses but also treatment plans as well as prognoses. We transcribe the real-scenario medical conversations into text, which is used to simulate real doctor-patient consultations for training effective medical dialogue models. Besides, we also extract video titles, doctor profiles, disease, symptoms, and hospital departments. An example is shown in Figure 1.

Compared with existing medical dialogue datasets based on online health community, our dataset also has **two extra advantages**. The first advantage is that it enables us to conduct the *modeling of language usage patterns*. On online health community, doctor-patient conversations are often completed offline. Offline language usage tends to adopt written expressions, which is quite different from the oral expressions in real medical consultation. Since our dataset is realistic starting from the dialogue scenario, dialogue models based on our dataset are more conducive to better modeling the language usage patterns for training a robust response generation model.

The second advantage is *comprehensiveness of information*. The constructed dataset not only contains medical conversations between doctors and patients during clinical consultations. We also collected and annotated a wide range of meta-data with respect to medical conversations including doctor profiles, hospital departments, diseases and symptoms. The meta-data can be used for fine-grained modeling and analysis on medical dialogue. For example, doctor profiles could be incorporated into an expertise-specific dialogue model for personalized response generation. Diseases and symptoms can be used to mine patients' dialogue intents for precise clinical treatments.

The main **contributions** of this work are sum-

[1] https://www.kuaishou.com

Video duration: 58 seconds
Doctor profile: Associate Chief Physician of Dermatology, Doctor of Medicine, with nearly 30 years of experience, specializing in the treatment of common and difficult diseases of skin diseases, such as psoriasis, dermatitis and eczema, vitiligo, alopecia, neuralgia after herpes zoster, and prurigo nodularis , Scleroderma, etc.

Title of the video clip
慢性湿疹十年，患者苦不堪言，相信我两种方法兼顾治疗 (Chronic eczema has been suffering for ten years, and the patient is miserable. Believe me that the two methods are both treatments)
Dialogue
医生：我看你那个手上和脚上都有湿疹是吧？(Doctor: I observe you have eczema on your hands and feet, right?)
病人：是的，慢性的，十年了。(Patient: Yes, chronic, for ten years.)
医生：什么时间能加重？你能找出原因吗？(Doctor: When can it get worse? Can you find out the reason?)
病人：好像就是跟心情有关系，容易焦虑。(Patient: It seems that it has something to do with mood, and I'm easy to be anxious)
医生：那我先给您开中药调着看，可能比较慢但是有效果，这个病很可能是神经性皮炎，但是从肉眼上有的时候分不出来，我用中药给您治一下。(Doctor: I will prescribe Chinese medicine for you. It may be slower but effective. This disease is probably neurodermatitis, but sometimes it cannot be distinguished from the naked eye. I will treat you with Chinese medicine.)
.....
Disease and Symptom
疾病：神经性皮炎 (Disease: Neurodermatitis)
症状：湿疹 (Symptoms: eczema)
Hospital Department
皮肤病科 (Dermatology)

Figure 1: An exemplar offline medical consultation, which includes (1) a video clip with doctor profile, (2) video title, (3) dialogue between doctor and patient, and (4) disease, symptom and department.

marized as follows.

- We construct a large-scale medical dialogue dataset - RealMedDial, which contains (1) 2,637 real-scenario medical conversations from pre-recorded video clips by 59 doctors in their daily clinical consultations, and (2) comprehensive metadata of medical dialogues, such as the expertise of doctors, hospital departments and diseases that each doctor is good at treating. To the best of our knowledge, RealMedDial is the first medical dialogue dataset based on real consultations.
- We annotate all the medical dialogues with dialogue-specific diseases and symptoms, which can be used for medical information extraction and intent mining. Combined with doctor profiles, personalized medical dialogue model could be developed to meet diversified medical intents and improve automatic healthcare services.
- We validate the usability of RealMedDial on three tasks, including medical response generation, department routing and doctor recommendation. Experimental results demonstrated the usefulness of our dataset, meanwhile indicating a large space for future research.

The rest of this paper is organized as follows. Section 2 reviews related work to our paper. Section 3 introduces our data collection including data source, cleaning, annotation strategy and statistics. Section 4 provides our experiments of dialogue generation, department routing and doctor recommendation on the constructed dataset. Section 5 concludes this work and provides future directions for the constructed dataset.

2 Related Work

Our work primarily concerns two lines of related work: medical dialogue systems and medical dialogue datasets.

2.1 Medical Dialogue Systems

Recent research on medical dialogue systems has mostly focused on natural language understanding and dialogue management. Various natural language understanding tasks have been investigated in medical dialogue, such as medical information extraction (Lin et al., 2019; Du et al., 2019a,b; Zhang et al., 2020), medical slot filling (Shi et al., 2020), and medical conversational summarization (Joshi et al., 2020). For dialogue management, inspired by the successful application of reinforcement learning in dialogue management strategy (Dhingra et al., 2017; Li et al., 2017; Peng et al., 2018), Wei et al. (2018) first addressed automatic diagnosis in medical dialogue using reinforcement learning framework. Xu et al. (2019) further proposed an end-to-end relational dialogue system to enhance medical diagnosis using knowledge-routed deep Q-network. Xia et al. (2020) proposed a GAN-based policy gradient framework for automatic diagnosis. However, most previous work merely fo-

cused on a single module of the pipeline-based medical dialogue system, and built task-specific datasets for model evaluation. Our work aims to build a real-time based medical dialogue dataset that contains as much information as possible to facilitate various tasks of medical dialogue.

2.2 Medical Dialogue Datasets

For medical dialogue datasets, the MZ dataset (Wei et al., 2018) and the DX dataset (Xu et al., 2019) were first launched for symptom extraction using self-reports of patients and conversations in online healthcare community. Similarly, Shi et al. (2020) collected a dialogue dataset with the purpose of medical slot filling. Since these datasets are collected for symptom extraction tasks, they are hardly applied to other medical dialogue tasks. Lin et al. (2019) released the CMDD dataset with 2,067 pediatric-related dialogues. Zhang et al. (2020) collected the MIE dataset with 1,120 cardiovascular-related dialogues. Zhou et al. (2021) collected two dialogue datasets, CovidDialog in English and in Chinese, containing doctor-patient conversations about COVID-19. CMDD, MIE and CovidDialog are built for understanding disease-specific natural language, but not for dialogue generation. Zeng et al. (2020) built two large-scale medical dialogue datasets, MedDialog-CN and MedDialog-EN from different healthcare communities. Liu et al. (2020) released a large-scale high-quality medical dialogue dataset related to 12 types of common gastrointestinal diseases. Existing medical dialogue datasets are mostly built from the offline question answering in online healthcare communities to simulate real-time dialogue, which partly hinders the performance of medical dialogue systems. Unlike previous work, we build our dataset based on real medical consultations to enhance medical dialogue performance. We compare our dataset and other existing medical dialogue datasets in Table 1.

3 Data Collection

3.1 Dataset Overview

Our raw data are crawled from the short-video clips of Kuaishou, which is one of the largest Chinese short-video clip platform with over three hundred professional doctors out of about 300 million users. Doctors regularly release their daily medical consultation video clips for healthcare services. The video clips of medical consultation record the whole process of medical diagnosis with conversations between doctors and patients.

# dialogues	2,637
# utterances	24,255
Avg. # of utterances in a dialogue	9.20
Median # of utterances in a dialogue	8
Max # of utterances in a dialogue	48
Min # of utterances in a dialogue	2
# doctors	59
Max # of dialogues of a doctor	184
Min # of dialogues of a doctor	4
# departments	17
Avg. # of dialogues of a department	155.12
Avg. # of doctors of a department	3.47
Avg. # of diseases of a department	4.26
Avg. # of symptoms of a department	8.48

Table 2: The statistics of the RealMedDial dataset.

We manually searched and selected 125 professional doctor accounts with totally 4.7K video clips as our primary data source. To avoid potential ethical risks and ensure the quality of the data, we manually filtered the video clips that have been edited or only included introduction to popular science medical knowledge, retaining those containing complete multi-turn patient-doctor dialogues. Finally, we obtained 2,637 video clips released by 59 doctor accounts with multi-turn doctor-patient dialogue. Table 2 shows the statistics of our dataset containing medical dialogues transcribed from real-scenario medical consultations.

Besides, since the collected dialogues are from real doctors, we crawled the profiles of doctors from Kuaishou user homepages and Baidu Encyclopedia[2] as supplement metadata for fine-grained medical dialogue research. We categorized the doctors according to hospital departments, which could be used to build fine-grained dialogue model for different hospital departments. We show the statistics of hospital departments of our datasets in Table 3. In the following sections, we describe our data cleaning , annotation strategy and quality control in detail.

3.2 Data Cleaning

We transcribed the contents of the selected video clips as text. Fifteen graduate students participated in the transcription process with five-fold cross validation to ensure the quality of the transcribed text.

[2]https://baike.baidu.com/

ID	Department	# doctors	# dialogues
1	Cardiovascular	7	240
2	Andrology	3	35
3	Dermatology	6	272
4	Internal Medicine	6	326
5	Gastroenterology	9	150
6	Orthopedics	4	182
7	Anorectal	3	116
8	Obstetrics	3	130
9	Gynecology	2	48
10	Rheumatology	6	424
11	Chinese Medicine	4	202
12	Urology	1	26
13	Endocrinology	2	86
14	Nephrology	3	260
15	Brain	1	26
16	Respiratory	1	4
17	Pediatrics	2	110

Table 3: The hospital departments of the RealMedDial dataset.

Figure 2: The interface for data annotation.

Each transcribed medical dialogue contains four fields: video title, multi-turn dialogue, diseases and symptoms. The video titles often appear in the form of question sentences, indicating the medical problems that the video contents aim to solve. The dialogue is the entire process of real-time medical consultations with multi-turn patient-doctor question answering. The diseases and the symptoms are annotated based on Chinese medical subject headings (Li et al., 2001). We removed personal information, duplicate video clips and single turn dialogues by rule-based filtering.

3.3 Annotation Strategy and Quality Control

We annotated the transcribed medical dialogue with user intents, including diseases and symptoms. Other medical intents can be extended in future studies. The annotation process is achieved by fifteen native Chinese graduate students under the guidance of a professional expert. The annotators followed detailed annotation instructions with standard principles and potentially occurred difficulties. In the annotation process, formal training lessons and regular seminars are carried out to exchange ideas and discuss problems on annotation once a week during the six-week annotation process. The annotation guidelines changed three times as we added information on newly found annotation difficulties during the entire annotation period.

Specifically, we divided fifteen students into five groups, and each group consisted of three student annotators. Using cross-validated annotation, the three-member groups annotated the user intents, and the expert participated in the final decision when there was divergence. If an agreement could not be reached on certain data annotation, everyone discussed and determined the annotation to ensure its accuracy and consistency.

We used a standardized method to achieve high-quality annotation. An interface, shown in Figure 2, was provided to allow the annotators to precisely enter intent information. To promote the correction of the entered terms, we employed Chinese medical subject headings as a support tool to obtain more specialized expressions of user intents. Since the annotation process is based on the annotators' intuition, the results may be subjective. To verify the reliability of annotations, we adopt Kappa score (Sidney and John, 1988) to measure inter-annotator agreement, which is widely used in annotation scheme of computational linguistics. To measure inter-annotator agreement, the annotators were given the same 1,000 medical dialogues to annotate the intents. The agreements on the intents of diseases and symptoms were 0.78 and 0.74, which means the annotation is substantially reliable.

4 Experiments

The RealMedDial dataset is built from real-scenario medical consultations, and thus, it can be used to simulate medical dialogue in a real environment for developing effective automatic medical chatbot. Except for generating useful responses of medical dialogue, RealMedDial can also be used for the department routing task and the doctor recommendation task. Next, we provide detailed evaluation on these three tasks, respectively.

4.1 Medical Response Generation

Medical response generation is one of the most important tasks for medical dialogue, aiming to

generate informative and instructive responses in consideration of the dialogue context and health conditions of patients. Since we collect the doctor-patient conversations from real medical consultations, the data can largely cover language usage patterns of real human-to-human oral conversations. Therefore, our dataset is more conducive to capturing and modeling semantic information in the dialogue by the machine, thereby simulating artificial language patterns to generate useful responses and fully grasp the contextual information of the current dialogue.

4.1.1 Model Pretraining

We trained several response generation models on the RealMedDial dataset as benchmark results for future comparison. Response generation can be generally formulated as a language modeling process in recent proposed models. Given the dialogue context with multi-turn conversations, the probability on the sequence of tokens in the response is modeled as follows:

$$p(r|c) = p(r_1|c) \prod_{i=2}^{n} p(r_i|c, r_1, ..., r_{i-1}), \quad (1)$$

where c denotes the multi-turn dialogue context, and r denotes the next token in the generated response.

Based on this idea, the pretrained GPT2 model (Radford et al., 2019) is proposed to use Transformer decoder to model the generative conditional probability, which enhances the GPT model (Radford et al., 2018) with a few modifications. GPT2 achieves good performance on several text generation tasks reported from existing work (Mass and Roitman, 2020; Bai et al., 2021).

BERT-GPT (Wu et al., 2020; Lewis et al., 2020) is another pretrained language model that integrates the BERT-based encoder and the GPT-based decoder. In BERT-GPT, BERT is used to encode the input token sequence with masks, which is then fed into the GPT decoder for recovering the masked tokens and generating the dialogue responses.

CDial-GPT is a recently proposed pretrained model for Chinese dialogue generation, which is built on a large-scale cleaned Chinese conversation dataset LCCC (Wang et al., 2020). CDial-GPT fills up the gaps in the pre-trained Chinese GPT language models, and provides a reliable model for Chinese dialogue generation. We use the pretrained GPT2, BERT-GPT and CDial-GPT, and fine-tune these models on the RealMedDial dataset to examine their performance for dialogue generation.

We split the RealMedDial dataset into a training set, a validation set, and a test set with the ratio of 8:1:1. The split was carried out separately in different departments, which was based on dialogues instead of source-target pairs. For CDial-GPT and GPT2, we used the implementation by THU-COAI[3], and followed the default hyperparameter settings in the original CDial-GPT (Wang et al., 2020). For BERT-GPT, we used the implementation by UCSD-AI4H[4], and also followed the default hyperparameter setting of the original model. The maximum length of input sequences was truncated to 300, and that of output sequences was truncated to 100. Top-k random sampling (Fan et al., 2018) with k=50 was used for decoding in all the used models.

We evaluated the trained models using automatic metrics including Perplexity, NIST-n (Doddington, 2002) (where n is the size of n-gram and is set as 4), BLEU-n (Papineni et al., 2002) (where n is set as 2 and 4), METEOR (Lavie and Agarwal, 2007), Entropy-n (Zhang et al., 2018) (where n is set as 4), and Dist-n (Li et al., 2016) (where n is set as 1 and 2). Perplexity measures the language quality of the generated responses. NIST, BLEU, and METEOR measure the similarity between the generated responses and the ground truths via n-gram matching. Entropy and Dist measure the lexical diversity of the generated responses. CDial-GPT was pretrained on LCCC-base (a large-scale cleaned Chinese conversation dataset), which is filtered from 79 million conversations from one of the largest Chinese social media website Weibo. BERT-GPT was pretrained by UCSD-AI4H on Chinese corpus collected from a large scale Chinese corpus for NLP[5]. GPT2 was pretrained by UCSD-AI4H on Chinese Chatbot Corpus[6] containing 14 million dialogues and 500K Chinese dialogues[7].

4.1.2 Evaluation Results

Table 4 shows the response generation performance on the RealMedDial dataset. From the table, we

[3] https://github.com/thu-coai/CDial-GPT

[4] https://github.com/UCSD-AI4H/Medical-Dialogue-System

[5] https://github.com/brightmart/nlp_chinese_corpus

[6] https://github.com/codemayq/chinese_chatbot_corpus

[7] https://drive.google.com/file/d/1nEuew_KNpTMbyy7BO4c8bXMXN351RCPp/view

	CDial-GPT	GPT2	BERT-GPT
Perplexity	21.25	**16.40**	29.95
NIST-4	**10.30**	9.30	0.55
BLEU-2	**1.196**	1.123	0.068
BLEU-4	**0.481**	0.439	0.028
METEOR	**1.403**	1.385	0.009
Entropy-4	7.00	6.21	**8.99**
Dist-1	0.178	**0.215**	0.090
Dist-2	0.602	**0.647**	0.469

Table 4: Performance of response generation on the RealMedDial dataset.

Conversation history: 我的痘为什么严重了呢？就是在暑假吃凉的，也吃的比较油腻，在家里。天天吃肉，我觉得一天三顿都是肉。(Why is my acne getting worse? I eat cold and greasy food during the summer, and I eat meat at home every day, three meals a day.)
Groundtruth response: 吃肉没关系，就是少油。瘦的肉少油。然后呢，注意夏天不吃茄子，茄子是很油的，茄子特别吸油，尤其是烧茄子，什么茄丁面呀，烧茄子，这些都特别油。(It's okay to eat meat, but less oil is needed. The lean meat has less oil. Then, be careful, do not eat eggplant in the summer, eggplant is very oily, eggplant especially absorb oil, especially roasted eggplant, eggplant diced noodles, etc., these are particularly oily.)
CDial-GPT: 你这种情况你不能吃肉。(In your case you can't eat meat.)
GPT2: 你就是有点儿严重了，所以你要注意饮食，保持好的睡眠，少吃肉，少吃甜的，油炸食品。(You're just a little bit more serious, so you need to pay attention to your diet, get a good night's sleep, eat less meat, less sweet, fried foods.)
BERT-GPT: 应该是内火大，主要是脏腑功能失调所致，所以在饮食方面呢，一定要少吃油腻的和油炸的，这样呢对你皮肤是有影响的，所以平时就早睡。我给你开点中药，你试试，先吃一段时间。(It should be a big internal fire, mainly due to the dysfunction of the internal organs, so in terms of diet, it must eat less greasy and fried, so that it has an impact on your skin, so usually go to bed early. I'll give you some Chinese medicine, you try, first eat a period of time.)

Figure 3: An example of generated responses on the RealMedDial test set.

observe that GPT2 achieved the better performance than the other two models in terms of Perplextiy, Dist-1 and Dist-2. Since these three metrics are used to measure the informativeness and diversity of the generated text, it indicates that GPT2 can generate more diverse and informative responses. Although CDial-GPT yielded better performance on four machine translation metrics, NIST-4, BLEU-2, BLEU-4 and METEOR, these metrics are all auxiliary metrics that evaluate the performance of dialogue generation in terms of n-gram matching (Liu et al., 2016). The CDial-GPT model is pretrained using the social media data which may not be well applied in medical dialogue task, thus leading to much lower values of these metrics compared with other tasks, such as machine translation. To further illustrate the comparison of different models, we provide two examples of the generated responses in Figure 3 and Figure 4.

4.2 Department Routing

The task of department routing is to attribute current dialogue with patient descriptions to the corresponding hospital departments for optimizing di-

Conversation history: 就是这个手指头，就是肿着疼着呢，就想这个握拳都不能握，不知道是什么原因，但是我的手指好像变形都好几年了，今年好像 10 号有点疼。(My fingers are swollen and painful, and I can't even make a fist if I want to, I don't know what the reason is, but my fingers seem to be deformed for several years, and this year it seems to be a little painful on the 10th.)
Groundtruth response: 你听我说啊，你的这个远端这个手指头这都是长的骨刺，它不是类风湿，这个呢就是骨关节炎，这骨关节炎呢一个是跟遗传有关系，另外一个就是年轻的时候可能摸凉水，干活儿干的太过了，它现在呢就会退化的，长骨刺更加早一些，平时呢尽量的这种少摸一些凉水，少用手干一干活，省着点儿用。另外呢我开点儿中药连喝再洗，好不好？(You have a bone spur in your finger, it is not rheumatoid, but osteoarthritis. This osteoarthritis is related to genetics, and the other is that when you are young, you may touch the cold water, work too much, and it will degenerate now, the bone spur is even earlier, usually try to touch less cold water, less hand work, save it. In addition, I will give you some Chinese medicine, you drink and wash, okay?)
CDial-GPT: 你这个可能是关节有问题，需要去医院检查一下。(You may have a problem with your joints and need to go to the hospital for a checkup.)
GPT2: 建议你去拍个片子，平时注意保护手指。(It is recommended that you get an x-ray and pay attention to protecting your fingers in general.)
BERT-GPT: 变形了好几年，首先你要去医院检查一下，拍 x 片，看看有没有坏死，然后呢你伸指的时候疼不疼？有没有麻木？如果一点儿感觉都没有，我们就给你开点中药试试。(Deformation for several years, first of all you have to go to the hospital to check, take x-rays, see if there is necrosis, and then you stretch your fingers when the pain is not painful? Is there any numbness? If you don't feel anything at all, we'll give you some Chinese medicine to try.)

Figure 4: Another example of generated responses on the RealMedDial test set.

alogue models according to the characteristics of the departments. Since the RealMedDial dataset contains a wide range of hospital departments with respect to different medical domains, we design this task for providing more targeted dialogue services for patients. The department routing task can be tackled by multi-class classification. Namely, given a brief description of the health issues of an patient, related departments can be matched for the patient, which could be helpful to get more accurate medical services and fine-tune the pretained dialogue model for generating more personalized and useful responses.

4.2.1 Model Pretraining

We use three BERT-based models, BERT (Devlin et al., 2019), RoBERTa (Liu et al., 2019), and ALBERT (Lan et al., 2020), for department routing. These models have been proved to be effective in various NLP tasks. Besides, we also adopt the CPT (Shao et al., 2021) model, which is designed as a Chinese pre-trained unbalanced Transformer to utilize the shared knowledge between natural language understanding and natural language generation through a shared encoder, an understanding decoder, and a generation decoder. We input health descriptions into these models and predict the department that the corresponding disease or symptom belongs to.

4.2.2 Implementation Details

We labeled the dialogue texts with index of departments and doctors respectively, and split the dataset into a training set and a test set with the

Metrics	CPT	RoBerta	BERT	ALBERT
Accuracy	**0.749**	0.705	0.686	0.611
m-Prec.	**0.552**	0.485	0.452	0.377
m-Recall	**0.540**	0.487	0.481	0.371
m-F1	**0.536**	0.480	0.462	0.359
w-Prec.	**0.707**	0.654	0.627	0.553
w-Recall	**0.749**	0.705	0.686	0.611
w-F1	**0.723**	0.674	0.652	0.566

Table 5: Performance of department routing on the RealMedDial dataset.

Metrics	CPT	RoBerta	BERT	ALBERT
Accuracy	**0.621**	0.523	0.552	0.348
m-Prec.	**0.379**	0.247	0.263	0.115
m-Recall	**0.375**	0.277	0.293	0.144
m-F1	**0.353**	0.240	0.256	0.110
w-Prec.	**0.564**	0.409	0.439	0.228
w-Recall	**0.621**	0.523	0.552	0.348
w-F1	**0.565**	0.438	0.465	0.247

Table 6: Performance of doctor Recommendation on the RealMedDial dataset.

ratio of 4:1. Pretrained language models are the primary ingredients of the state-of-the-art text classifiers including BERT, RoBERTa, ALBERT and CPT. These models are trained on the training set, and the weighting parameters were learned with AdamW (Loshchilov and Hutter, 2017), whose ϵ was set as 1e-8. The initial learning rate was set as 4e-5. The learning rate scheduler was set as Linear. We evaluate the models with metrics including Accuracy, macro/weighted Precision (m/w-Prec.), macro/weighted Recall (m/w-Recall) and macro/weighted F1 score (m/w-F1).

4.2.3 Evaluation Results

Table 5 shows the experimental results for department routing. From the table, we can observe that CPT outperforms other models in all the evaluation metrics. This is because CPT can capture specific knowledge of this task using a shared encoder, an understanding decoder, and a generation decoder. Compared with other models, CPT takes full advantage of previous pre-trained models and achieves better performance in department routing.

4.3 Doctor Recommendation

Doctor recommendation aims to recommend suitable doctors based on patients' health status. Since RealMedDial contains real medical consultation records of multiple doctors, we can model different doctors according to the characteristics of their language usage and unique forms of question-answering by building doctor profiles. The doctor profiles can be used to build chatbots to assist in the completion of various medical health services.

4.3.1 Evaluation Results

Similar to department routing, doctor recommendation task is also a multi-class classification problem, and we still use BERT, RoBERTa, ALBERT and CPT as baselines. Table 6 shows the experimental results for doctor recommendation. From the table, we can observe that CPT still outperforms other models in terms of different evaluation metrics. Like the performance trends of department routing, CPT contributes to effective modeling of language usage pattern and profiles of doctors by incorporating more comprehensive domain-specific information into the learned model, and yields better doctor recommendation results.

4.4 Further Discussion

We validate the usability of RealMedDial on medical response generation, department routing and doctor recommendation. Experimental results have shown the usefulness of RealMedDial in these tasks, which also provides benchmark results for future studies. Advanced models trained using RealMedDial could consider more special nature of medical consultation for generating accurate and low risk medical responses. To this end, more effective models trained on RealMedDial can be devised by comprehensively using the wide range of metadata of our corpus, such as doctor profiles and disease descriptions. Although experiments in this work are our preliminary attempts on demonstrating the usefulness of our corpus, we will extend our future work to consider more domain-specific information to develop more effective generation models. Although our dataset is in Chinese, the application scenario is not limited to Chinese applications. To adapt RealMedDial to research in other languages, dialogue contents can be tokenized to token IDs, which can thus be used to train dialogue models for other language-based research. We will also use automated methods, such as the automatic transcription software [8], to reduce the time cost and manual labor in constructing and expanding our dataset in future.

5 Conclusion and Future Work

To facilitate automatical medical consultation, we construct RealMedDial, a high-quality Chinese

[8] https://sonix.ai/

dataset of medical dialogue based on real scenario medical consultation from online short-video clips. Real medical consultation contributes to learning more powerful and human-like dialogue models by considering communications in reality between doctors and patients instead of question answering-based communications on online health community. We collected and annotated a wide range of meta-data with respect to the medical dialogue including titles of short-video clips, doctor profiles, hospital departments, diseases and symptoms for fine-grained analysis on language usage pattern and clinical diagnosis. We evaluated the performance of medical response generation, department routing and doctor recommendation on RealMedDial. Results show that RealMedDial is applicable to various medical dialogue tasks. As for future work, we will build personalized dialogue models by incorporating more professional knowledge into medical response generation.

6 Ethical Consideration

The original short-video clips of our study are collected from Kuaishou, one of the largest Chinese short-video clip platform with over three hundred professional doctors out of about 300 million users. The doctor profiles are collected from Kuaishou user homepages and Baidu Encyclopedia. All the collected data are public available, which do not contain any personal privacy information of patients and doctors. We have ensured that the doctors obtain patient consent in the first place to post videos of consultations, and the short videos are without any personal information of patients. The constructed corpus is completely anonymous, and the identity of the patient or doctor cannot be inferred from it. The transcribed texts in RealMedDial are randomly shuffled so that it is hard to find connections between the original videos and the transcribed text without the identity of the doctor. Therefore, there is no privacy issue for the data we use. When annotating the dataset, all annotators were paid based on their workload and submitted all required consent forms. Since this work only focuses on medical dialogue without additional identified and private information, the protection of privacy is preserved.

7 Acknowledgements

This work is partially supported by grant from the Natural Science Foundation of China (No. 62006034), Natural Science Foundation of Liaoning Province (No. 2021-BS-067) and the Fundamental Research Funds for the Central Universities (No.DUT21RC(3)015). We would like to thank our reviewers for their insightful comments, which help us greatly enhance our work.

References

He Bai, Peng Shi, Jimmy Lin, Luchen Tan, Kun Xiong, Wen Gao, Jie Liu, and Ming Li. 2021. Semantics of the unwritten: The effect of end of paragraph and sequence tokens on text generation with GPT2. In *Proceedings of the ACL-IJCNLP 2021 Student Research Workshop, ACL 2021, Online, July 5-10, 2021*, pages 148–162. Association for Computational Linguistics.

Avisha Das, Salih Selek, Alia R. Warner, Xu Zuo, Yan Hu, Vipina Kuttichi Keloth, Jianfu Li, W. Jim Zheng, and Hua Xu. 2022. Conversational bots for psychotherapy: A study of generative transformer models using domain-specific dialogues. In *Proceedings of the 21st Workshop on Biomedical Language Processing, BioNLP@ACL 2022*, pages 285–297. Association for Computational Linguistics.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT 2019, Minneapolis, MN, USA, June 2-7, 2019, Volume 1 (Long and Short Papers)*, pages 4171–4186. Association for Computational Linguistics.

Bhuwan Dhingra, Lihong Li, Xiujun Li, Jianfeng Gao, Yun-Nung Chen, Faisal Ahmed, and Li Deng. 2017. Towards end-to-end reinforcement learning of dialogue agents for information access. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics, ACL 2017, Vancouver, Canada, July 30 - August 4, Volume 1: Long Papers*, pages 484–495. Association for Computational Linguistics.

George Doddington. 2002. Automatic evaluation of machine translation quality using n-gram co-occurence statistics. In *Proceedings of the second international conference on Human Language Technology Research, 2002*.

Nan Du, Kai Chen, Anjuli Kannan, Linh Tran, Yuhui Chen, and Izhak Shafran. 2019a. Extracting symptoms and their status from clinical conversations. In *Proceedings of the 57th Conference of the Association for Computational Linguistics, ACL 2019, Florence, Italy, July 28- August 2, 2019, Volume 1: Long Papers*, pages 915–925. Association for Computational Linguistics.

Nan Du, Mingqiu Wang, Linh Tran, Gang Lee, and Izhak Shafran. 2019b. Learning to infer entities, properties and their relations from clinical conversations. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing, EMNLP-IJCNLP 2019, Hong Kong, China, November 3-7, 2019*, pages 4978–4989. Association for Computational Linguistics.

Angela Fan, Mike Lewis, and Yann N. Dauphin. 2018. Hierarchical neural story generation. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics, ACL 2018, Melbourne, Australia, July 15-20, 2018, Volume 1: Long Papers*, pages 889–898. Association for Computational Linguistics.

Anirudh Joshi, Namit Katariya, Xavier Amatriain, and Anitha Kannan. 2020. Dr. summarize: Global summarization of medical dialogue by exploiting local structures. *arXiv preprint arXiv:2009.08666*.

Soheil Keshmiri, Hidenobu Sumioka, Ryuji Yamazaki, and Hiroshi Ishiguro. 2019. Decoding the perceived difficulty of communicated contents by older people: Toward conversational robot-assistive elderly care. *IEEE Robotics Autom. Lett.*, 4(4):3263–3269.

Zhenzhong Lan, Mingda Chen, Sebastian Goodman, Kevin Gimpel, Piyush Sharma, and Radu Soricut. 2020. ALBERT: A lite BERT for self-supervised learning of language representations. In *8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, April 26-30, 2020*. OpenReview.net.

Alon Lavie and Abhaya Agarwal. 2007. METEOR: an automatic metric for MT evaluation with high levels of correlation with human judgments. In *Proceedings of the Second Workshop on Statistical Machine Translation, WMT@ACL 2007, Prague, Czech Republic, June 23, 2007*, pages 228–231. Association for Computational Linguistics.

Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Veselin Stoyanov, and Luke Zettlemoyer. 2020. BART: denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, ACL 2020, Online, July 5-10, 2020*, pages 7871–7880. Association for Computational Linguistics.

Danya Li, Tiejun Hu, Zhu Wenyan, Qing Qian, Huiling Ren, Junlian Li, and Bin Yang. 2001. Retrieval system of chinese medical subject headings. *Chinese Journal of Medical Library*, 4:1–9.

Jiwei Li, Michel Galley, Chris Brockett, Jianfeng Gao, and Bill Dolan. 2016. A diversity-promoting objective function for neural conversation models. In *NAACL HLT 2016, The 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, San Diego California, USA, June 12-17, 2016*, pages 110–119. The Association for Computational Linguistics.

Xiujun Li, Yun-Nung Chen, Lihong Li, Jianfeng Gao, and Asli Celikyilmaz. 2017. End-to-end task-completion neural dialogue systems. In *Proceedings of the Eighth International Joint Conference on Natural Language Processing, IJCNLP 2017, Taipei, Taiwan, November 27 - December 1, 2017 - Volume 1: Long Papers*, pages 733–743. Asian Federation of Natural Language Processing.

Xinzhu Lin, Xiahui He, Qin Chen, Huaixiao Tou, Zhongyu Wei, and Ting Chen. 2019. Enhancing dialogue symptom diagnosis with global attention and symptom graph. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing, EMNLP-IJCNLP 2019, Hong Kong, China, November 3-7, 2019*, pages 5032–5041. Association for Computational Linguistics.

Chia-Wei Liu, Ryan Lowe, Iulian Serban, Michael Noseworthy, Laurent Charlin, and Joelle Pineau. 2016. How NOT to evaluate your dialogue system: An empirical study of unsupervised evaluation metrics for dialogue response generation. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing, EMNLP 2016, Austin, Texas, USA, November 1-4, 2016*, pages 2122–2132. The Association for Computational Linguistics.

Wenge Liu, Jianheng Tang, Jinghui Qin, Lin Xu, Zhen Li, and Xiaodan Liang. 2020. Meddg: A large-scale medical consultation dataset for building medical dialogue system. *CoRR*, abs/2010.07497.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. RoBERTa: A robustly optimized BERT pretraining approach. *CoRR*, abs/1907.11692.

Ilya Loshchilov and Frank Hutter. 2017. Fixing weight decay regularization in adam. *CoRR*, abs/1711.05101.

Devin M Mann, Ji Chen, Rumi Chunara, Paul A Testa, and Oded Nov. 2020. Covid-19 transforms health care through telemedicine: evidence from the field. *Journal of the American Medical Informatics Association*, 27(7):1132–1135.

Yosi Mass and Haggai Roitman. 2020. Ad-hoc document retrieval using weak-supervision with BERT and GPT2. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing, EMNLP 2020, Online, November 16-20, 2020*, pages 4191–4197. Association for Computational Linguistics.

Shamila Nasreen, Julian Hough, and Matthew Purver. 2021. Rare-class dialogue act tagging for alzheimer's disease diagnosis. In *Proceedings of the 22nd Annual Meeting of the Special Interest Group on Discourse and Dialogue, SIGdial 2021*, pages 290–300. Association for Computational Linguistics.

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. Bleu: a method for automatic evaluation of machine translation. In *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics, July 6-12, 2002, Philadelphia, PA, USA*, pages 311–318. ACL.

Baolin Peng, Xiujun Li, Jianfeng Gao, Jingjing Liu, and Kam-Fai Wong. 2018. Deep dyna-q: Integrating planning for task-completion dialogue policy learning. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics, ACL 2018, Melbourne, Australia, July 15-20, 2018, Volume 1: Long Papers*, pages 2182–2192. Association for Computational Linguistics.

Alec Radford, Karthik Narasimhan, Tim Salimans, and Ilya Sutskever. 2018. Improving language understanding by generative pre-training. *OpenAI*.

Alec Radford, Jeff Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. 2019. Language models are unsupervised multitask learners. *OpenAI*.

Yunfan Shao, Zhichao Geng, Yitao Liu, Junqi Dai, Fei Yang, Li Zhe, Hujun Bao, and Xipeng Qiu. 2021. CPT: A pre-trained unbalanced transformer for both chinese language understanding and generation. *CoRR*, abs/2109.05729.

Xiaoming Shi, Haifeng Hu, Wanxiang Che, Zhongqian Sun, Ting Liu, and Junzhou Huang. 2020. Understanding medical conversations with scattered keyword attention and weak supervision from responses. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pages 8838–8845.

Siegel Sidney and Castellan N. John. 1988. Nonparametric statistics for the behavioral sciences. *McGraw-Hill*.

Yida Wang, Pei Ke, Yinhe Zheng, Kaili Huang, Yong Jiang, Xiaoyan Zhu, and Minlie Huang. 2020. A large-scale chinese short-text conversation dataset. In *The CCF International Conference on Natural Language Processing and Chinese Computing*, pages 91–103.

Zhongyu Wei, Qianlong Liu, Baolin Peng, Huaixiao Tou, Ting Chen, Xuan-Jing Huang, Kam-Fai Wong, and Xiang Dai. 2018. Task-oriented dialogue system for automatic diagnosis. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 201–207.

Qingyang Wu, Lei Li, Hao Zhou, Ying Zeng, and Zhou Yu. 2020. Importance-aware learning for neural headline editing. In *The Thirty-Fourth AAAI Conference on Artificial Intelligence, AAAI 2020, New York, NY, USA, February 7-12, 2020*, pages 9282–9289. AAAI Press.

Yuan Xia, Jingbo Zhou, Zhenhui Shi, Chao Lu, and Haifeng Huang. 2020. Generative adversarial regularized mutual information policy gradient framework for automatic diagnosis. In *The Thirty-Fourth AAAI Conference on Artificial Intelligence, AAAI 2020, New York, NY, USA, February 7-12, 2020*, pages 1062–1069. AAAI Press.

Lin Xu, Qixian Zhou, Ke Gong, Xiaodan Liang, Jianheng Tang, and Liang Lin. 2019. End-to-end knowledge-routed relational dialogue system for automatic diagnosis. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, pages 7346–7353.

Guangtao Zeng, Wenmian Yang, Zeqian Ju, Yue Yang, Sicheng Wang, Ruisi Zhang, Meng Zhou, Jiaqi Zeng, Xiangyu Dong, Ruoyu Zhang, Hongchao Fang, Penghui Zhu, Shu Chen, and Pengtao Xie. 2020. Meddialog: Large-scale medical dialogue datasets. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing, EMNLP 2020, Online, November 16-20, 2020*, pages 9241–9250. Association for Computational Linguistics.

Yizhe Zhang, Michel Galley, Jianfeng Gao, Zhe Gan, Xiujun Li, Chris Brockett, and Bill Dolan. 2018. Generating informative and diverse conversational responses via adversarial information maximization. In *Advances in Neural Information Processing Systems 31: Annual Conference on Neural Information Processing Systems 2018, NeurIPS 2018, December 3-8, 2018, Montréal, Canada*, pages 1815–1825.

Yuanzhe Zhang, Zhongtao Jiang, Tao Zhang, Shiwan Liu, Jiarun Cao, Kang Liu, Shengping Liu, and Jun Zhao. 2020. Mie: A medical information extractor towards medical dialogues. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 6460–6469.

Meng Zhou, Zechen Li, Bowen Tan, Guangtao Zeng, Wenmian Yang, Xuehai He, Zeqian Ju, Subrato Chakravorty, Shu Chen, Xingyi Yang, Yichen Zhang, Qingyang Wu, Zhou Yu, Kun Xu, Eric P. Xing, and Pengtao Xie. 2021. On the generation of medical dialogs for COVID-19. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing, ACL/IJCNLP 2021, (Volume 2: Short Papers), Virtual Event, August 1-6, 2021*, pages 886–896. Association for Computational Linguistics.

TempoWiC: An Evaluation Benchmark for Detecting Meaning Shift in Social Media

Daniel Loureiro[◇]
Aminette D'Souza[*◇], **Areej Nasser Muhajab**[*◇], **Isabella A. White**[*◇], **Gabriel Wong**[*◇]
Luis Espinosa Anke[◇♡], **Leonardo Neves**[♣], **Francesco Barbieri**[♣], **Jose Camacho-Collados**[◇]
[◇] Cardiff NLP, School of Computer Science and Informatics, Cardiff University, UK
[♣] Snap Inc., Santa Monica, California, USA; [♡]AMPLYFI, UK
[◇]boucanovaloureirod@cardiff.ac.uk, cardiffnlp.contact@gmail.com

Abstract

Language evolves over time, and word meaning changes accordingly. This is especially true in social media, since its dynamic nature leads to faster semantic shifts, making it challenging for NLP models to deal with new content and trends. However, the number of datasets and models that specifically address the dynamic nature of these social platforms is scarce. To bridge this gap, we present TempoWiC, a new benchmark especially aimed at accelerating research in social media-based meaning shift. Our results show that TempoWiC is a challenging benchmark, even for recently-released language models specialized in social media.

1 Introduction

One of the most studied challenges in NLP is lexical ambiguity. Solutions include word sense disambiguation (Navigli, 2009) or entity linking (Ling et al., 2015), where words are linked to sense inventories such as WordNet (Miller, 1995) or Wikipedia. Recently, notable progress has been made with the advent of Language Models (LMs) and contextualized embeddings, crucially well equipped for modeling meaning in context (Pilehvar and Camacho-Collados, 2020; Bevilacqua et al., 2020).

One notable limitation with current lexical semantics benchmarks, however, is that they are typically clean and time-invariant, where standard grammar is the norm, and have little to no account of language usage in real-world platforms like social media. However, there is ample agreement that modeling changes in language and topic distributions is crucial for modern NLP (Loureiro et al., 2022a). Thus, there is a rich body of literature concerned with, e.g., adapting existing word representations (mainly word embeddings) diachronically (Hamilton et al., 2016; Szymanski, 2017; Rosenfeld and Erk, 2018; Hofmann et al., 2021), exposing LMs to time-specific data (Lazaridou et al., 2021), or temporal adaptation in general (Luu et al., 2021; Agarwal and Nenkova, 2022; Jin et al., 2021; Loureiro et al., 2022a). The lack of real-world data to serve as ground truth has typically limited the evaluation of diachronic word-level NLP models. This limitation has been addressed in a myriad of ways, e.g., by comparing distributional similarities with human judgments (Gulordava and Baroni, 2011), contrasting change vs. frequency (Hamilton et al., 2016), comparing time-sensitive representations with stripped-down versions (Frermann and Lapata, 2016) or, more recently, determining whether a word has acquired new senses over time by looking at relatively large targeted subcorpora (Van Hee et al., 2018), or probing for diachronic awareness in settings reminiscent of knowledge base completion (Dhingra et al., 2021; Hofmann et al., 2021).

Despite the above, few works have attempted to model the connection between meaning shift and social media. Among them, Del Tredici et al. (2019) showed that trending words are a meaningful signal for predicting meaning shifts. This platform-specific insight is the main basis for the construction of our dataset. TempoWiC follows the simple formulation from the SuperGLUE Word-in-Context (WiC) challenge (Pilehvar and Camacho-Collados, 2019), which is particularly well suited for temporal meaning shift evaluation given that it is not reliant on a reference sense inventory. This change of paradigm has seen wide adoption, with multilingual extensions such as XL-WiC (Raganato et al., 2020), Am2ico (Liu et al., 2021) or MCL-WiC (Martelli et al., 2021), or reformulations such as WiC-TSV (Breit et al., 2021). In contrast to these, TempoWiC is crucially designed around meaning shift and instances of word usage tied to Twitter trending topics. As an additional contribution, along with the benchmark, we provide a set of robust baselines and analyses that highlight the challenging nature of the task.

[*] Annotation team.

Proceedings of the 29th International Conference on Computational Linguistics, pages 3353–3359
October 12–17, 2022.

Tweet 1	Tweet 2	Label
2019-02 "I ain't gone let the ppl *frisk* me if I'm dirty homie"	*2020-02* "Set up a stop and *frisk* outside a white club and catch coke heads"	T
2019-04 "i wish i still had images of my old animal crossing new leaf *villager* he was good boy"	*2020-04* "How does *villager* trading in New horizons even work like tf"	T
2019-08 "This dude just said "Boys of the Backstreet" He made em sound like a whole *folklore*"	*2020-08* "my target app said they didn't have *folklore* cds but when i went inside they had some i'm so happy"	F
2019-08 "In case you were wondering facial devotion still worked with a face *mask* on"	*2020-08* "With these *mask* at work customers are forever confusing me and Reyna lmao"	F

Table 1: Examples from the training set of TempoWiC. Target words in *italic*. The label T (True) indicates that the word has the same meaning in the two tweets, the label F (False) indicates that the meaning is different.

2 TempoWiC: Temporal Word in Context

In this section, we describe our process to build our evaluation benchmark for detecting meaning shift in social media. The task is framed as a simple binary classification problem in which a target word is present in two texts (tweets) posted during different time periods. The goal is to decide whether the meaning corresponding to the first target word in context is the same as the second one or not. Table 1 lists a few examples.

2.1 Data collection

Word Selection. Since this work focuses on meaning shift, we do not consider neologisms and use lemmas from WordNet as an initial set of potential words of interest (82K lemmas, ignoring multi-word expressions, stopwords and numbers). From a corpus of 100M tweets collected from the Twitter API for the period between the start of 2019 and September of 2021, we compiled monthly frequency counts for this set of known words, and computed trending scores following Chen et al. (2021). Each trending word peak is estimated as the day with highest frequency during the year/month with most occurrences. As the prior date, we considered the same date exactly one year before. This is done in order to avoid seasonal confound factors, which are known to affect models in social media (Chae et al., 2012; Barbieri et al., 2018). Afterwards, we selected the top 10 words with highest trending scores from each month, resulting in 210 words which are candidates for annotation. For this selection, we ignored words with fewer than 100 occurrences in our corpus during their peak date.

Obtaining Paired Tweets. In this phase we collected, for each trending word, 100 tweets posted during the peak date, and 100 tweets posted during the prior date. For this phase we used the Twitter APIs, setting filters to request only English tweets, and ignoring replies and retweets.[1] We preprocessed each tweet using spaCy (Honnibal et al., 2020) and we randomly paired tweets from the prior and peak sets for specific words that match both in surface form and part-of-speech tag.

2.2 Annotation

Annotators. We recruited four annotators through our internal institution recruitment office. This ensured that the annotators were part of the process, trained and understood all the details of the task.[2] Annotators, who were all native or near-native English speakers, were all paid the equivalent of a research assistant per hour.

First stage. The annotation was split into two phases. First, we took a breadth-first approach in which a relatively short number of instances (i.e., 10) of a large number of the selected words (210 in total) were annotated. The motivation for this initial phase was to understand which words had some sort of meaning shift to start with. The selection of the words to be included in the dataset was then restricted to words which had more than 3 out of 10 instances with meaning shift.

[1] We retrieved additional tweets with the APIs as we included trending words that had a minimum occurrences of 100 tweets on peak day in our initial dataset, but some words included less than 100 tweets on the prior date.

[2] The full guidelines provided to the annotators are available in the task website.

	Word	# Instances (% Diff. Meaning)	Trending Date	Agreement (Krippendorf's α)
Train	frisk	99 (54%)	11/2/2020	0.718
	pogrom	99 (5%)	25/2/2020	0.482
	containment	100 (33%)	12/3/2020	0.274
	virus	96 (48%)	12/3/2020	0.254
	epicenter	100 (71%)	14/3/2020	0.124
	ventilator	99 (17%)	27/3/2020	0.541
	villager	100 (64%)	10/4/2020	0.546
	turnip	100 (95%)	10/5/2020	0.316
	bunker	98 (61%)	1/6/2020	0.408
	mask	99 (76%)	14/7/2020	0.255
	teargas	98 (3%)	18/7/2020	0.786
	paternity	100 (22%)	30/7/2020	0.289
	entanglement	99 (89%)	1/8/2020	0.623
	folklore	82 (92%)	3/8/2020	0.917
	parasol	100 (85%)	2/9/2020	0.446
Validation	impostor	99 (76%)	23/9/2020	0.544
	lotte	98 (43%)	27/9/2020	0.514
	recount	100 (28%)	6/11/2020	0.682
	primo	100 (77%)	9/11/2020	0.528
Test	milker	99 (50%)	4/3/2021	0.699
	moxie	97 (83%)	5/3/2021	0.755
	unlabeled	100 (90%)	10/3/2021	0.711
	pyre	100 (32%)	27/4/2021	0.243
	gaza	100 (60%)	15/5/2021	0.749
	ido	91 (83%)	27/5/2021	0.712
	airdrop	99 (40%)	6/6/2021	0.918
	bullpen	99 (9%)	16/6/2021	0.388
	crt	100 (68%)	26/6/2021	0.867
	monet	98 (94%)	8/7/2021	1.000
	burnham	100 (16%)	1/8/2021	0.964
	delta	100 (100%)	11/8/2021	1.000
	gala	100 (46%)	14/9/2021	0.498
	launchpad	99 (81%)	17/9/2021	0.558
	vanguard	99 (95%)	21/9/2021	0.421

Table 2: Details all the words included in TempoWiC. Maximum pairwise agreement is reported.

Second stage. The second phase was based on a depth-first approach in which 100 instances of all selected words from the first stage were annotated. We ensured that each instance was annotated by three annotators. The final label attributed to each instance was determined by majority vote.[3]

2.3 Statistics and Inter-annotator Agreement

The outcome of our annotation pipeline is a dataset of 3,297 instances divided in train/validation/test sets of size 1,428/396/1,473 instances, respectively. We measured inter-annotator agreement using Fleiss' Kappa at 0.446, and using Krippendorff's α at 0.439. Since each instance is assigned a majority vote label, we also computed the maximum pairwise Krippendorff's α at 0.627, which should be more revealing of the expected performance on this task. Words with Krippendorff's α below 0.1 were removed from the dataset. Table 2 provides a summary of the most relevant details of the dataset after annotation.

3 Evaluation

In this section, we report baseline results on TempoWiC using two different approaches which have proven successful on the WiC task that inspired this work. More concretely, we report results based on pretrained LMs using fine-tuning on the tweet pair as well as comparing the similarity of contextual embeddings.[4]

Evaluation metrics. The results are reported according to the standard Macro-F1 metric for multi-class classification problems. Accuracy is also reported for completeness but, given the unbalanced nature of the dataset, Macro-F1 should provide a more accurate representation of the performance.

3.1 Models

Our experiments include the following LMs: RoBERTa base and large pretrained on general domain corpora (Liu et al., 2019); RoBERTa base with continued training on tweets until the end of 2019, and a similar model trained with more tweets until the end of 2021 (Loureiro et al., 2022a, TimeLMs); LMs based on RoBERTa but trained from scratch on tweets, both base and large versions (Nguyen et al., 2020, BERTweet).

Each of these LMs is fine-tuned representing instances as "<s> Tweet 1 </s> Tweet 2 </s>", with each tweet represented by the encodings produced by each model's tokenizer.[5] Additionally, we trained a logistic regression classifier on the cosine similarity of the contextual embeddings corresponding to the target word on each tweet of the pair. This approach based on contextual embeddings is sensitive to the choice of layers from the LM used to represent embeddings. Following Loureiro et al. (2022b), we use SP-WSD layer pooling weights (model specific) that are suited for sense representation (see Appendix B for ablation results with alternative pooling strategies). Both

[3] Some words proved too difficult to reliably annotate according to feedback from the annotators as well as low agreement scores computed after annotation (more details in Section 2.3). Consequently, these words were removed from the dataset. Among the various challenges to be expected from annotating social media, mixed language (e.g., English and Hindi) was among the most frequent issues.

[4] Additional baselines are reported in Appendix A.

[5] We experimented concatenating the encodings for the target word at the end of the sequence as proposed by Wang et al. (2019) for WiC, but found no improvements.

	Model	Accuracy	Macro-F1
Fine-tuning	RoBERTa-base	66.89%	58.26%
	RoBERTa-large	66.49%	59.10%
	TimeLMs-2019-90M	66.46%	57.70%
	TimeLMs-2021-124M	65.04%	54.75%
	BERTweet-base	61.46%	51.27%
	BERTweet-large	67.93%	60.62%
Similarity	RoBERTa-base	67.96%	52.89%
	RoBERTa-large	72.98%	67.09%
	TimeLMs-2019-90M	**74.07%**	**70.33%**
	TimeLMs-2021-124M	71.01%	63.51%
	BERTweet-base	69.45%	65.16%
	BERTweet-large	69.18%	56.95%
Naive	Random	50.00%	50.00%
	All True	36.59%	26.79%
	All False	63.41%	38.80%

Table 3: Main results on the test set of TempoWiC. Fine-tuning results are the average of 3 runs.

Word	Accuracy		Macro-F1	
	Fine-tune	Similarity	Fine-tune	Similarity
airdrop	40.48%	**65.31%**	30.13%	**65.18%**
bullpen	**67.68%**	38.38%	**44.53%**	34.54%
burnham	27.67%	**83.00%**	27.62%	**69.15%**
crt	64.98%	**79.80%**	46.26%	**73.24%**
delta	85.37%	**98.98%**	46.05%	**49.74%**
gala	47.67%	**78.00%**	36.71%	**77.86%**
gaza	67.01%	**69.39%**	**66.74%**	64.61%
ido	83.88%	**90.11%**	63.04%	**75.78%**
launchpad	**81.82%**	80.81%	45.00%	**59.26%**
milker	46.13%	**64.65%**	37.28%	**63.44%**
monet	93.54%	**94.90%**	48.33%	**62.96%**
moxie	**89.35%**	68.04%	**74.75%**	56.48%
pyre	**64.29%**	51.02%	**62.78%**	50.84%
unlabeled	65.67%	**76.00%**	47.71%	**57.48%**
vanguard	**95.96%**	73.74%	**68.80%**	42.44%

Table 4: Performance by word on the TempoWiC test set. Using TimeLMs-2019-90M as the best similarity model, and BERTweet-large as the best fine-tuning model (average of 3 runs).

approaches are implemented with Wolf et al. (2020, Transformers).

3.2 Results

Our results on Table 3 show that TempoWiC is a challenging task with room for improvement. While the best results using both fine-tuning and similarity approaches are obtained by models adapted to the Twitter domain, this advantage isn't substantial over generic RoBERTa. Interestingly, we find that the straightforward similarity approach manages to substantially outperform fine-tuning, with a Twitter base model trained with data before any word's trending peak achieving the best performance. While this result may be surprising considering that fine-tuning performs better on WiC, this finding is in line with recent work in word sense disambiguation showing that approaches based on contextual embeddings can be more robust and generalizable than fine-tuning (Loureiro et al., 2021).

Analysis by word. Table 4 provides a detailed breakdown of the results of the best performing model (i.e., TimeLMs-2019-90M) by individual words. As can be observed, there are large differences between words, which are also due to the unbalanced natural distribution of certain words to start with (see Table 2). More interesting is perhaps the gaps between fine-tuning and similarity techniques. While similarity appears to be generally more robust, in words such as *bullpen* or *vanguard*, the tendency is reversed.

3.3 Future Work

This work only covers English, but future work should include additional languages and experiment with both multilingual and monolingual models. We leave an analysis explaining the difference between the 2019 and 2021 models for future work as well, alongside the development of methods that leverage the dates provided with each instance towards improved performance, similarly to Dhingra et al. (2021); Rosin et al. (2022).

4 Conclusion

This work introduced a new lexical semantics task and a dataset, TempoWiC, focused on meaning shift detection in Twitter. While meaning representation is at the core of the task, the challenges of this task go beyond simple word sense disambiguation with a focus on its temporal aspect. To make the task realistic, we extracted Twitter trending words for different periods and paired them with tweets from past periods. This makes the task more challenging and grounded in real-world applications for social media. We performed extensive experiments with standard meaning representation approaches based on language models. The results show that the task leaves ample room for improvement, with several avenues for future research on how to better integrate time-aware social media models with meaning representation techniques. The TempoWiC dataset and baseline scripts are available at github.com/cardiffnlp/tempowic.

Acknowledgements

Jose Camacho-Collados and Daniel Loureiro are supported by a UKRI Future Leaders Fellowship. We thank Snap Inc. for providing resources to support this project.

References

Oshin Agarwal and Ani Nenkova. 2022. Temporal effects on pre-trained models for language processing tasks. *Transactions of the Association for Computational Linguistics*, 10:904–921.

Francesco Barbieri, Luis Marujo, Pradeep Karuturi, William Brendel, and Horacio Saggion. 2018. Exploring emoji usage and prediction through a temporal variation lens. *arXiv preprint arXiv:1805.00731*.

Michele Bevilacqua, Marco Maru, and Roberto Navigli. 2020. Generationary or "how we went beyond word sense inventories and learned to gloss". In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 7207–7221.

Piotr Bojanowski, Edouard Grave, Armand Joulin, and Tomas Mikolov. 2017. Enriching word vectors with subword information. *Transactions of the Association for Computational Linguistics*, 5:135–146.

Anna Breit, Artem Revenko, Kiamehr Rezaee, Mohammad Taher Pilehvar, and Jose Camacho-Collados. 2021. WiC-TSV: An evaluation benchmark for target sense verification of words in context. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*, pages 1635–1645, Online. Association for Computational Linguistics.

Junghoon Chae, Dennis Thom, Harald Bosch, Yun Jang, Ross Maciejewski, David S Ebert, and Thomas Ertl. 2012. Spatiotemporal social media analytics for abnormal event detection and examination using seasonal-trend decomposition. In *2012 IEEE conference on visual analytics science and technology (VAST)*, pages 143–152. IEEE.

Shuguang Chen, Leonardo Neves, and Thamar Solorio. 2021. Mitigating temporal-drift: A simple approach to keep NER models crisp. In *Proceedings of the Ninth International Workshop on Natural Language Processing for Social Media*, pages 163–169, Online. Association for Computational Linguistics.

Marco Del Tredici, Raquel Fernández, and Gemma Boleda. 2019. Short-term meaning shift: A distributional exploration. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 2069–2075, Minneapolis, Minnesota. Association for Computational Linguistics.

Bhuwan Dhingra, Jeremy R Cole, Julian Martin Eisenschlos, Daniel Gillick, Jacob Eisenstein, and William W Cohen. 2021. Time-aware language models as temporal knowledge bases. *arXiv preprint arXiv:2106.15110*.

Lea Frermann and Mirella Lapata. 2016. A bayesian model of diachronic meaning change. *Transactions of the Association for Computational Linguistics*, 4:31–45.

Kristina Gulordava and Marco Baroni. 2011. A distributional similarity approach to the detection of semantic change in the google books ngram corpus. In *Proceedings of the GEMS 2011 workshop on geometrical models of natural language semantics*, pages 67–71.

William L. Hamilton, Jure Leskovec, and Dan Jurafsky. 2016. Diachronic word embeddings reveal statistical laws of semantic change. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1489–1501, Berlin, Germany. Association for Computational Linguistics.

Valentin Hofmann, Janet Pierrehumbert, and Hinrich Schütze. 2021. Dynamic contextualized word embeddings. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 6970–6984, Online. Association for Computational Linguistics.

Matthew Honnibal, Ines Montani, Sofie Van Landeghem, and Adriane Boyd. 2020. spaCy: Industrial-strength Natural Language Processing in Python.

Xisen Jin, Dejiao Zhang, Henghui Zhu, Wei Xiao, Shang-Wen Li, Xiaokai Wei, Andrew Arnold, and Xiang Ren. 2021. Lifelong pretraining: Continually adapting language models to emerging corpora. *arXiv preprint arXiv:2110.08534*.

Angeliki Lazaridou, Adhiguna Kuncoro, Elena Gribovskaya, Devang Agrawal, Adam Liska, et al. 2021. Pitfalls of static language modelling. *arXiv preprint arXiv:2102.01951*.

Xiao Ling, Sameer Singh, and Daniel S Weld. 2015. Design challenges for entity linking. *Transactions of the Association for Computational Linguistics*, 3:315–328.

Qianchu Liu, Edoardo Maria Ponti, Diana McCarthy, Ivan Vulić, and Anna Korhonen. 2021. AM2iCo: Evaluating word meaning in context across low-resource languages with adversarial examples. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 7151–7162, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019.

Roberta: A robustly optimized BERT pretraining approach. *CoRR*, abs/1907.11692.

Daniel Loureiro, Francesco Barbieri, Leonardo Neves, Luis Espinosa Anke, and Jose Camacho-collados. 2022a. TimeLMs: Diachronic language models from Twitter. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics: System Demonstrations*, pages 251–260, Dublin, Ireland. Association for Computational Linguistics.

Daniel Loureiro, Alípio Mário Jorge, and Jose Camacho-Collados. 2022b. LMMS reloaded: Transformer-based sense embeddings for disambiguation and beyond. *Artificial Intelligence*, 305:103661.

Daniel Loureiro, Kiamehr Rezaee, Mohammad Taher Pilehvar, and Jose Camacho-Collados. 2021. Analysis and Evaluation of Language Models for Word Sense Disambiguation. *Computational Linguistics*, 47(2):387–443.

Kelvin Luu, Daniel Khashabi, Suchin Gururangan, Karishma Mandyam, and Noah A Smith. 2021. Time waits for no one! analysis and challenges of temporal misalignment. *arXiv preprint arXiv:2111.07408*.

Federico Martelli, Najla Kalach, Gabriele Tola, and Roberto Navigli. 2021. SemEval-2021 task 2: Multilingual and cross-lingual word-in-context disambiguation (MCL-WiC). In *Proceedings of the 15th International Workshop on Semantic Evaluation (SemEval-2021)*, pages 24–36, Online. Association for Computational Linguistics.

George A Miller. 1995. Wordnet: a lexical database for english. *Communications of the ACM*, 38(11):39–41.

Roberto Navigli. 2009. Word sense disambiguation: A survey. *ACM computing surveys (CSUR)*, 41(2):1–69.

Dat Quoc Nguyen, Thanh Vu, and Anh Tuan Nguyen. 2020. BERTweet: A pre-trained language model for English tweets. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 9–14, Online. Association for Computational Linguistics.

Mohammad Taher Pilehvar and Jose Camacho-Collados. 2019. WiC: the word-in-context dataset for evaluating context-sensitive meaning representations. In *NAACL*, pages 1267–1273, Minneapolis, Minnesota. Association for Computational Linguistics.

Mohammad Taher Pilehvar and Jose Camacho-Collados. 2020. Embeddings in natural language processing: theory and advances in vector representations of meaning. *Synthesis Lectures on Human Language Technologies*, 13(4):1–175.

Alessandro Raganato, Tommaso Pasini, Jose Camacho-Collados, and Mohammad Taher Pilehvar. 2020. XL-WiC: A multilingual benchmark for evaluating semantic contextualization. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 7193–7206, Online. Association for Computational Linguistics.

Alex Rosenfeld and Katrin Erk. 2018. Deep neural models of semantic shift. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 474–484, New Orleans, Louisiana. Association for Computational Linguistics.

Guy D. Rosin, Ido Guy, and Kira Radinsky. 2022. Time masking for temporal language models. In *Proceedings of the Fifteenth ACM International Conference on Web Search and Data Mining*, WSDM '22, page 833–841, New York, NY, USA. Association for Computing Machinery.

Terrence Szymanski. 2017. Temporal word analogies: Identifying lexical replacement with diachronic word embeddings. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 448–453, Vancouver, Canada. Association for Computational Linguistics.

Cynthia Van Hee, Els Lefever, and Véronique Hoste. 2018. Semeval-2018 task 3: Irony detection in english tweets. In *Proceedings of The 12th International Workshop on Semantic Evaluation*, pages 39–50.

Alex Wang, Yada Pruksachatkun, Nikita Nangia, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel Bowman. 2019. Superglue: A stickier benchmark for general-purpose language understanding systems. In *Advances in Neural Information Processing Systems*, volume 32. Curran Associates, Inc.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Remi Louf, Morgan Funtowicz, Joe Davison, Sam Shleifer, Patrick von Platen, Clara Ma, Yacine Jernite, Julien Plu, Canwen Xu, Teven Le Scao, Sylvain Gugger, Mariama Drame, Quentin Lhoest, and Alexander Rush. 2020. Transformers: State-of-the-art natural language processing. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 38–45, Online. Association for Computational Linguistics.

A Additional Baselines

Besides the fine-tuning and similarity methods described in the main paper, we also experimented with additional approaches in order to better understand the difficulty of this dataset.

In this appendix we provide additional results based on an MLP trained with concatenated contextual embeddings (Table 5), and another MLP trained with the concatenation of the average of static embeddings from each tweet (Table 6). Embeddings are *L2* normalized after concatenation.

The hyper-parameters used with these MLPs were determined by grid search on 24 different configurations which were tested on the validation set. The parameters tested were *hidden layer sizes* ((*embedding size*, 100) or (100)), *solver* (adam or sgd), *batch size* (32 or 64), and *maximum number of iterations* (50, 100 or 200).

Static embeddings are based on fastText (Bojanowski et al., 2017) and learned from Twitter data on the same corpora used for Loureiro et al. (2022a). These embeddings are trained with skip-gram for 300-dimensions, min-ngram size 2 and max-ngram size 12.

Model	Accuracy	Macro-F1
RoBERTa-base	68.11% (0.77%)	55.00% (1.88%)
RoBERTa-large	**68.82%** (1.06%)	55.17% (2.60%)
TimeLMs-2019-90M	68.68% (0.68%)	**58.62%** (0.81%)
TimeLMs-2021-124M	68.23% (0.90%)	57.55% (3.57%)
BERTweet-base	65.51% (0.91%)	57.03% (3.56%)
BERTweet-large	66.55% (0.59%)	54.16% (0.09%)

Table 5: Performance of MLP trained with concatenation of the target word's contextual embeddings (SP-WSD pooling), tuned on the validation set. Reporting average of 3 runs, and standard deviation.

Model	Accuracy	Macro-F1
CommonCrawl	55.53% (0.78%)	48.98% (0.57%)
TimeLMs-2019-90M	**57.64%** (0.71%)	49.46% (0.28%)
TimeLMs-2021-124M	55.20% (0.31%)	**52.30%** (0.57%)

Table 6: Performance of MLP trained with concatenation of the average of static embeddings from each tweet. Reporting average of 3 runs, and standard deviation.

B Pooling Contextual Embeddings

Table 7 reports results using the similarity method described in the main paper with alternative choices for layer pooling. We considered the final layer and the sum of the last 4 layers as these are common choices in word sense disambiguation settings.

Model	Final Layer	Sum Last 4	SP-WSD
RoBERTa-base	40.89%	**60.33%**	52.89%
RoBERTa-large	38.80%	53.32%	**67.09%**
TimeLMs-2019-90M	59.93%	67.69%	**70.33%**
TimeLMs-2021-124M	53.14%	60.26%	**63.51%**
BERTweet-base	67.35%	**66.91%**	65.16%
BERTweet-large	38.80%	41.75%	**56.95%**

Table 7: Performance of Contextual Similarity method according to choice of layer pooling approach.

Generation of Large-scale Multi-turn Dialogues from Reddit

Daniil Huryn
Computer Science
Emory University
Atlanta, GA, USA
danikhur@gmail.com

William M. Hutsell
Computer Science
Emory University
Atlanta, GA, USA
mack.hutsell@gmail.com

Jinho D. Choi
Computer Science
Emory University
Atlanta, GA, USA
jinho.choi@emory.edu

Abstract

This paper presents novel methods to automatically convert posts and their comments from discussion forums such as Reddit into multi-turn dialogues. Our methods are generalizable to any forums; thus, they allow us to generate a massive amount of dialogues for diverse topics that can be used to pretrain language models. Four methods are introduced, Greedy$_{Baseline}$, Greedy$_{Advanced}$, Beam Search and Threading, which are applied to posts from 10 subreddits and assessed. Each method makes a noticeable improvement over its predecessor such that the best method shows an improvement of 36.3% over the baseline for appropriateness. Our best method is applied to posts from those 10 subreddits for the creation of a corpus comprising 10,098 dialogues (3.3M tokens), 570 of which are compared against dialogues in three other datasets, *Blended Skill Talk*, *Daily Dialogue*, and *Topical Chat*. Our dialogues are found to be more engaging but slightly less natural than the ones in the other datasets, while it costs a fraction of human labor and money to generate our corpus compared to the others. To the best of our knowledge, it is the first work to create a large multi-turn dialogue corpus from Reddit that can advance neural dialogue systems.

1 Introduction

With the advent of encoder-decoder frameworks (Brown et al., 2020; Lewis et al., 2020; Raffel et al., 2020), neural-based open-domain dialogue models have recently gained a tremendous interest as they start sounding more human-like than ever (Adiwardana et al., 2020; Zhang et al., 2020; Roller et al., 2021). Training robust neural-based models requires a huge amount of dialogue data in numerous topics that is difficult to procure as real human-to-human conversations are expensive and time-intensive to conduct (Godfrey et al., 1992). Several studies have presented large dialogue data created by crowdsourcing (Zhang et al., 2018; Dinan et al., 2019; Rashkin et al., 2019; Gopalakrishnan et al., 2019). However, it still requires non-trivial configurations in the cloud platform and the performance of crowd workers needs to be monitored constantly while paying them and the service a good amount of fees.

Most encoder-decoder models used in dialogue systems are not pretrained on dialogues, just fine-tuned on relatively small dialogue datasets, which is a limiting factor. Few studies have utilized comment threads in discussion forums for the creation of dialogue data and enhanced the performance of dialogue systems (Al-Rfou et al., 2016; Mazaré et al., 2018). However, these comment-originated dialogues tend to be short and not as sensible due to a lack of contexts from the main posts that are unsuitable for training multi-turn dialogue models. Such data scarcity points toward a necessity for a parameterized model that generates dialogues of different forms, styles, and topics in high quantity.

In this paper, we first introduce four algorithms to automatically convert posts and their associated comments from discussion forums such as Reddit into multi-turn one-to-one dialogues (Section 3). Our approach leverages the vast and available suite of human content and interaction online, with the potential to create many diverse dialogues, where the choice of subreddits constitutes topic selection. It also adapts a sentence-level language model for estimating likelihoods among posts and comments to sequence sounding utterances, and is analyzed across 10 subreddits (Section 4). We then create a large dialogue corpus and demonstrate the efficacy of our approach through head-to-head evaluation against dialogues from three well-known datasets, which indicates that our dialogues are as engaging and natural as those from others that are manually generated (Section 5). This work will facilitate the development of dialogue models in all kinds of areas that have been hindered by the data scarcity.[1]

[1]Our resources are available through https://github.com/emorynlp/reddit-to-dialogue.

Proceedings of the 29th International Conference on Computational Linguistics, pages 3360–3373
October 12–17, 2022.

2 Related Work

Several dialogue datasets that are created through crowdsourcing have been presented. *Persona Chat* was created by assigning specific personas to two annotators who act as characters with those personas and generate a dialogue (Zhang et al., 2018). *Wizard of Wikipedia* was created by assigning two roles to annotators, the appentice acting as a curious learner of a specific topic and the wizard informing about the topic with a retrieved Wikipedia article (Dinan et al., 2019). *Empathetic Dialogues* was created by assigning two roles to annotators, the speaker describing a situation when a specific emotion would occur and the listener reacting to such a emotional situation (Rashkin et al., 2019). *Blended Skill Talk* was created by asking annotators to combine the personal, knowledgeable, and empathetic aspects of the previous three datasets together to generate more natural dialogues (Smith et al., 2020). *Topical Chat* was created by giving Wikipedia sections, fun facts, and news articles for a specific topic to annotators and asking them to generate a dialogue (Gopalakrishnan et al., 2019).

Only a few dialogue datasets have been created automatically. *Daily Dialogue* was crawled from various websites including dialogues scripted for English learners to practice daily conversations (Li et al., 2017). Mazaré et al. (2018) scrapped Reddit comments with replies and considered them short dialogues, which would not be multi-turn.

2.1 Comparisons to BST, DD, and TC

For comparisons to our corpus (Section 5.2), *Topical Chat* was chosen because it was least restricted in creation among crowdsourced datasets, *Blended Skill Talk* was chosen because it combined those 3 important aspects in dialogue, and *Daily Dialogue* was chosen because it was not crowdsourced but scripted by English educators. Table 1 shows comparisons among popular datasets and our corpus.

Data	DIA	UTT	TOK
Empathetic Dialogues	24,850	107,104	1,627,973
Wizard of Wikipedia	22,311	201,999	3,359,456
Daily Dialogue	13,118	103,632	1,504,635
Persona Chat	12,949	195,180	-
Topical Chat	10,784	235,434	4,614,506
Blended Skill Talk	6,808	77,502	1,058,325
Our Corpus	10,098	109,916	3,317,807

Table 1: The statistics of dialogue datasets including our corpus presented in Section 5. DIA/UTT/TOK: the total number of dialogues/utterances/tokens.

3 Reddit-to-Dialogue Generation

We introduce four algorithms for the dialogue generation: greedy baseline (Section 3.1), greedy advanced (Section 3.2), beam search (Section 3.3), and threading (Section 3.4). The main objective is to generate a multi-turn dialogue using a post and its comments (and replies)[2] that flows naturally in context. All algorithms assume that the number of sentences in the input post is less than or equal to the number of comments. The generated dialogues involve two speakers where utterances of Speakers 1 and 2 are extracted from the post and comments, respectively. All algorithms are evaluated on posts from diverse subreddits (Section 4).

3.1 Greedy Baseline Algorithm

Algorithm 1 depicts the baseline greedy approach that finds the most appropriate top-level comment for each sentence in a post. Given the input post $P = [p_1, .., p_n]$ where p_i is the i'th sentence in P, and the set of P's comments $\mathbb{C} = \{C_1, .., C_m\}$ s.t. $C_j = [c_{j1}, .., c_{j\ell}]$ where C_j is the j'th comment in $\mathbb{C}$ and c_{jk} is the k'th sentence in C_j, it first creates the set of comment segments T using $\mathbb{C}$ (L2), then visits every sentence $p_i \in P$ (L3), which gets appends to the output dialogue D (L4). Next, it finds the most-likely segment $\hat{t} \in T$ (L5)[3] and adds $\hat{t}$ to D (L6). Finally, T gets trimmed with $\hat{t}$ (L7) and the algorithm returns D as the output (L8).

Algo. 1: Greedy$_B$: greedy baseline

Input : P: a post, $\mathbb{C}$: a set of P's top-level comments.
Output: D: a dialogue.

1 $D \leftarrow [\,]$;
2 $T \leftarrow$ *segment*$(\mathbb{C})$;
3 **while** $\exists\, p_i \leftarrow$ *first*(P) **do**
4 $\quad D \leftarrow D \oplus [p_i]$;
5 $\quad \hat{t} \leftarrow \text{argmax}_{\forall t \in T}$ *ranker*(D, t);
6 $\quad D \leftarrow D \oplus [\hat{t}]$;
7 $\quad T \leftarrow$ *trim*$(T, \hat{t})$;
8 **return** D;

The *first* method removes and returns the first sentence in P. The *segment* method makes each comment a segment s.t. *segment*$(\mathbb{C}) = \{C'_1, \ldots, C'_m\}$, where $C'_j = c_{j1} \frown ... \frown c_{j\ell}$ ($\frown$: text concatenation).

[2] In this section, 'comments' imply the top-level comments of the post, and "replies" imply the replies to those comments.
[3] The *any* method returns any item in the input set.

The *ranker* method takes D comprising all previous utterances and p_i, then estimates the likelihood of t being the next utterance. Two models are used for this estimation, the human-like classifier (`HLC`) in DialogRPT (Gao et al., 2020) and BERT's next sentence predictor (`NSP`) (Devlin et al., 2019). At last, the *trim* method removes $\hat{t} = C'_j$ from T such that $\mathit{trim}(T, \hat{t}) = T \setminus \{C'_j\}$.

For `HLC`, p_i and t are fed into the model, which gives a score of how natural t is to follow p_i.[4] For `NSP`, since the original language model does not expect dialogue contents as input, we finetune it on the Multi-Session Chat dataset (Xu et al., 2022), the largest human-to-human chat dataset comprising ≈300K utterances. Given the finetuned model, the last two utterances in D (the last one is p_i and the second last one is a comment selected for p_{i-1}) are fed into the model with t, which gives scores for the two labels, `IsNext` and `NotNext`, s.t.

$$\mathit{score\_of}(\texttt{IsNext}) - \mathit{score\_of}(\texttt{NotNext})$$

is used for our likelihood estimation.[5]

3.2 Greedy Advanced Algorithm

Algorithm 2 shows the advanced greedy approach that makes two major updates from Algorithm 1.

Algo. 2: Greedy$_A$: greedy advanced

Input : P: post, $\mathbb{C}$: comment set, q: the max-length of comment segments.
Output: D: a dialogue.

$D \leftarrow [\,]$;
$T \leftarrow \mathit{segment}_a(\mathbb{C}, q)$;
while $\exists\, p_i \leftarrow \mathit{first}(P)$ **do**
  $D \leftarrow D \oplus [p_i]$;
  if $\exists\, p_{i+1} \in P$ **then** $T \leftarrow T \cup \{p_{i+1}\}$;
  $\hat{t} \leftarrow \mathrm{argmax}_{\forall t \in T}\, \mathit{ranker}(D, t)$;
  if $\hat{t} = p_{i+1}$ **then**
    $P \leftarrow [\mathit{last}(D) \frown \mathit{first}(P)] \oplus P$;
  else
    $D \leftarrow D \oplus [\hat{t}]$;
    $T \leftarrow \mathit{trim}_a(T, \hat{t}, q)$;
  if $\exists\, p_{i+1} \in T$ **then** $T \leftarrow T \setminus \{p_{i+1}\}$;
return D;

[4] Since `HLC` expected a sing-turn as input, we fed only p_i, although we also experimented by feeding more utterances in D, which led to worse performance.

[5] Feeding only p_i to `NSP` gave worse results whereas feeding more than two utterances in D gave very similar results, implying that BERT successfully learned to weigh more on the last two utterances. We also used only *score*(`IsNext`) as the estimator, which resulted in slightly worse performance.

First, it treats the next sentence $p_{i+1} \in P$ as a segment to rank if it exists (`L5`). If p_{i+1} is selected (`L7`), implying that it is better to have both p_i and p_{i+1} in one utterance, p_i is removed from D by $\mathit{last}(D)$, p_{i+1} is removed from P by $\mathit{first}(P)$, and their concatenation is prepended to P (`L8`). Once processed, p_{i+1} is removed from T (`L12`).

Second, the $\mathit{segment}_a$ method is updated (`L2`) such that it generates finer-grained segments using Algorithm 3. It is inspired by the fact that a single comment can (and often) address multiple aspects expressed in sentences that are not adjacent in P. In other words, one part of the comment may be appropriate for p_i while another part may be for p_j not adjacent to p_i; thus, using the whole comment as a response to either of them would be unnatural.

Algo. 3: $\mathit{segment}_a$: comment segmentation

Input : $\mathbb{C}$: a set of comments, q: the max # of sentences to join.
Output: T: a set of comment segments.

$T \leftarrow \varnothing$;
foreach $C_h \in \mathbb{C}$ **do**
  foreach $i \in [1, |C_h|]$ **do**
    $n \leftarrow \min(i + q - 1, |C_h|)$;
    foreach $j \in [i, n]$ **do**
      $T \leftarrow T \cup \{\mathit{join}(C_h, i, j)\}$;
return T;

The algorithm takes $\mathbb{C}$ and q, indicating the max-number of sentences allowed in any segment, and visits every comment $C_h = [c_{h1}, .., c_{h\ell}] \in \mathbb{C}$ (`L2`). For each sentence $c_{hi} \in C_h$ (`L3`), it joins all sentences between c_{hi} and c_{hj} using the *join* method as follows (`L4`: $i \le j \le \min(i + q - 1, |C_h|)$):

$$\mathit{join}(C_h, i, j) = \begin{cases} c_{hi} & \text{if } i = j \\ c_{hi} \frown \ldots \frown c_{hj} & \text{otherwise} \end{cases}$$

All joined segments are added to T (`L5-6`), which is returned as the output set (`L7`). For our experiments, $q = 3$ is used since the average number of sentences in Reddit comments (in our data) is < 4.

When $\mathit{segment}_a$ is applied to Algorithm 2 (`L2`), $\hat{t}$, which is appended to D, is a segment of a comment (`L10`). Let $\hat{t} = c_{jk}$ where c_{jk} is the k'th segment of C_j. The trim_a method then removes all segments generated for C_j from T (`L11`) such that

$$\mathit{trim}(T, c_{jk}, q) = T \setminus \mathit{segment}_a(\{C_j\}, q)$$

It is possible to keep the rest of unused segments from C_j that have no overlap with c_{jk}. However,

such segments generally sound like "speeches out of context". Thus, we decided to remove all segments associated with C_j for the future selections.

3.3 Beam Search Algorithm

Algorithm 4 shows the beam search approach with an additional parameter k for the beam size. It creates the beam set $\mathcal{B}$ with the tuple of 6 items: (an input post P, a dialogue D, a segment set T, a sequence score θ, a sequence count ϕ), and the set of output dialogues $\mathcal{F}$ (L1-2). While there is any beam, the state set G is initialized (L3-4). Let Ω_α be $(P_\alpha, D_\alpha, T_\alpha, \theta_\alpha, \phi_\alpha)$. For every beam $\Omega_\alpha \in \mathcal{B}$, the first sentence $p_i \in P_\alpha$ is added to D_α (L5-7); p_{i+1} is added to T_α if it exists (L8-9). For each segment $t \in T_\alpha$, the copies $P'_\alpha, D'_\alpha, T'_\alpha$ are created from their correspondents in Ω_α (L11) and t gets handled the same as in Algorithm 2 (L12-16).

Algo. 4: Beam$_k$: beam search

Input : P: post, $\mathbb{C}$: comment set, q: max segment length, k: a beam size.
Output: D: a dialogue, θ: the sequence score, ϕ: the sequence count.

1 $\mathcal{B} \leftarrow \{(P, [\,], \mathit{segment}_a(\mathbb{C}, q), 0, 0)\}$;
2 $\mathcal{F} \leftarrow \varnothing$;
3 **while** $\mathcal{B} \neq \varnothing$ **do**
4 $G \leftarrow \varnothing$;
5 **foreach** $\Omega_\alpha \in \mathcal{B}$ **do**
6 $p_i \leftarrow \mathit{first}(P_\alpha)$;
7 $D_\alpha \leftarrow D_\alpha \oplus [p_i]$;
8 **if** $\exists\, p_{i+1} \in P_\alpha$ **then**
9 $T_\alpha \leftarrow T_\alpha \cup \{p_{i+1}\}$
10 **foreach** *segment* $t \in T_\alpha$ **do**
11 $(P'_\alpha, D'_\alpha, T'_\alpha) \leftarrow \mathit{copy}(\Omega_\alpha)$;
12 **if** $t = p_{i+1}$ **then**
13 $P'_\alpha \leftarrow [\mathit{last}(D'_\alpha)^\frown \mathit{first}(P'_\alpha)] \oplus P'_\alpha$;
14 **else**
15 $D'_\alpha \leftarrow D'_\alpha \oplus [t]$;
16 $T'_\alpha \leftarrow \mathit{trim}_a(T'_\alpha, t, q)$;
17 $s \leftarrow \mathit{ranker}(D'_\alpha, t)$;
18 $(\theta'_\alpha, \phi'_\alpha) \leftarrow (\theta_\alpha + s, \phi_\alpha + 1)$;
19 **if** $\exists\, p_{i+1} \in T'_\alpha$ **then**
20 $T'_\alpha \leftarrow T'_\alpha \setminus \{p_{i+1}\}$;
21 $G \leftarrow G \cup \{(\Omega'_\alpha, s)\}$;
22 **else**
23 $\mathcal{F} \leftarrow \mathcal{F} \cup \{(D'_\alpha, \theta'_\alpha, \phi'_\alpha)\}$;
24 $\mathcal{B} \leftarrow \mathit{top\text{-}k}(G, k)$;
25 **return** *best*($\mathcal{F}$);

Given the ranking score s, the new sequence score θ' and count ϕ' are measured (L17-18). If p_{i+1} exists, it is removed from T'_α and the state (Ω'_α, s) is added to G, where $\Omega'_\alpha = (P'_\alpha, D'_\alpha, T'_\alpha, \theta'_\alpha, \phi'_\alpha)$ (L19-21). If $p_{i+1} \notin T'_\alpha$, no more sentences exist; thus, the current dialogue D'_α, its score θ'_α and the count ϕ'_α are stored in $\mathcal{F}$. Once all beams are used, $\mathcal{B}$ is reinitialized by the top-k states in G (L24) s.t.

$$G' \leftarrow [(\Omega_1, s_1), \ldots, (\Omega_{|G|}, s_{|G|})]\ (\forall i.\ s_i \geq s_{i+1})$$
$$\mathcal{B} \leftarrow \mathit{top\text{-}k}(G, k) = \mathbf{map}(\lambda_x : x[0], G')[:k]$$

Finally, $\mathit{best}(\mathcal{F}) = (D_\beta, \theta_\beta, \phi_\beta)$ is returned (L25) where $\theta_\beta \geq \theta_i : \forall_i.(D_i, \theta_i, \phi_i) \in \mathcal{F}$. Notice that although dialogues in $\mathcal{F}$ at L25 may comprise different lengths, the number of predictions made for every dialogue is the same as depicted in Figure 1. Although the first and second dialogues consist of 8 and 2 utterances respectively, the number of predictions made is 4 for both of them so that the min-length of any dialogue generated by our algorithm is 2 while the max-length is $2 \cdot n$ where $n = |P|$. Thus, it is safe to use θ_i as the sequence score, that is the sum of all prediction scores for the i'th path instead of the average score of θ_i/ϕ_i.

3.4 Threading Algorithm

Many top-level comments have threads of replies responded by the author of the post or other users. These replies are left out from our previous algorithms (Sections 3.1, 3.2, 3.3) because they do not necessarily address contents in the post. However, some of them can be used as bridging statements between comments selected by the algorithms and their following sentences from the post, which had been written before the comments were made. The challenge is how to glue a reply with the following sentence so that it does not sound disjointed.

Algo. 5: *thread*: threading

Input : P: post, D: dialogue, t: comment segment to be appended to D.
Output: D: the output dialogue including t and possibly a reply of t.

1 **if** $R \leftarrow \mathit{replies}(t)$ **then**
2 $\hat{r} \leftarrow \mathrm{argmax}_{\forall r \in R}\, \mathit{score}(P, D, t, r)$;
3 **if** $\mathit{glue}(P, D, t, \hat{r})$ **then**
4 **if** $P = \varnothing$ **then return** $D \oplus [t, \hat{r}]$;
5 **else** $P \leftarrow P \oplus [\hat{r}^\frown \mathit{first}(P)]$;
6 **return** $D \leftarrow D \oplus [t]$;

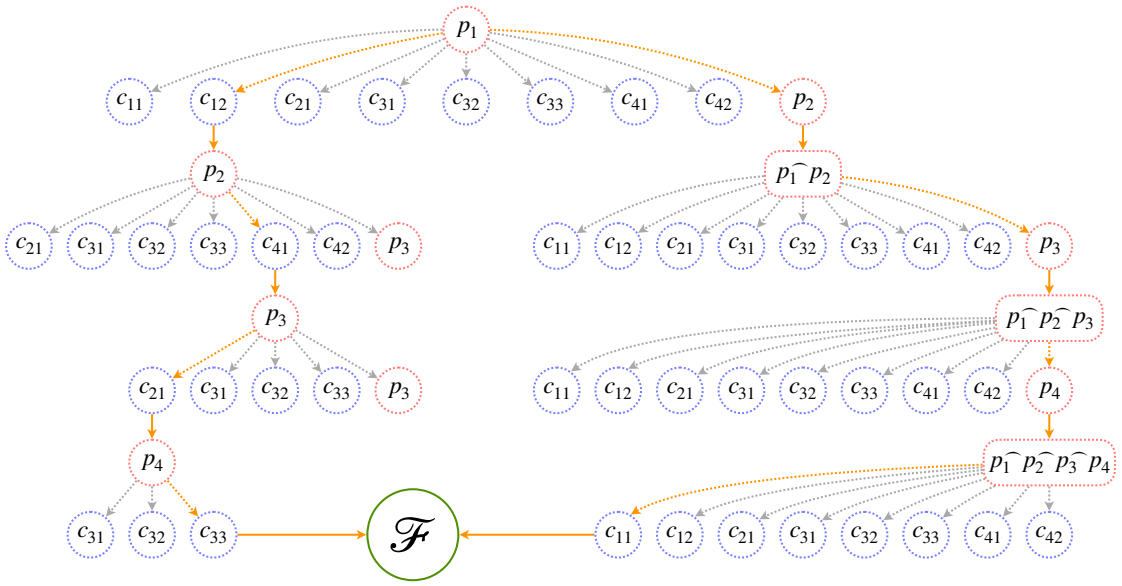

Figure 1: A demonstration of our beam search algorithm where one beam never selects p_{i+1} whereas the other beam always chooses p_{i+1} (the two extreme cases). The orange lines indicate the paths of the two beams where the solid lines are deterministically chosen while the dashed lines are predicted. The first beam results in $D_1 = [p_1, c_{12}, p_2, c_{41}, p_3, c_{21}, p_4, c_{33}]$ and the second beam results in $D_2 = [p_1 \frown p_2 \frown p_3 \frown p_4, c_{11}]$.

Algorithm 5 describes the threading algorithm that replaces `L15` in Algorithm 4 as follows:

$$D'_\alpha \leftarrow \textit{thread}(P'_\alpha, D'_\alpha, t)$$

Given the post P, the dialogue D, and the segment t from `L14` in Algorithm 4, it first retrieves the set R of all top-level replies using the *replies* method for the comment that t belongs to (`L1`). For every reply $r \in R$, given the last utterance $p \in D$ (that is p_i from `L7` in Algorithm 4) and the next sentence $n = \textit{first}(P)$, the *score* method measures how natural $\hat{r}$ would be in between t and n such that (`L2`):

$$\textit{score}(P, D, t, r) = \begin{cases} (1-\gamma)\mathbf{P}(r|p,t) + \gamma\mathbf{P}(n|t,r) & \text{if } P = \varnothing \\ \mathbf{P}(r|p,t) & \text{otherwise} \end{cases}$$

$\mathbf{P}(r|p,t)$ estimates how likely r is to follow p & t while $\mathbf{P}(n|t,r)$ estimates how likely n is to follow t & r. For this likelihood estimation, the next sentence predictor `NSP` in Section 3.1 is adapted s.t.:

$$\mathbf{P}(r|p,t) = \texttt{NSP}(p \frown t, r)$$
$$\mathbf{P}(n|t,r) = \texttt{NSP}(t \frown r, n)$$

For our experiments, $\gamma = 0.7$ is used, which gives more weight on $\mathbf{P}(n|t,r)$ than $\mathbf{P}(r|p,t)$. Then, the *glue* method takes the highest scoring reply $\hat{r}$ and adjusts the score by its length as follows (`L3`):

$$\hat{s} = \textit{score}(P, D, t, \hat{r}) \cdot (1 + 0.01(\lambda - |\hat{r}|))$$

In our case, $\lambda = 15$ so that it would boost the score if $|\hat{r}| < 15$, indicating that the number of tokens in $\hat{r}$ is less than 15, whereas it would drop the score if $|\hat{r}| > 15$. This guides the algorithm to select short replies that are found to be more useful. The *glue* method returns a boolean value as follows:[6]

$$\textit{glue}(P, D, t, \hat{r}) = \begin{cases} \textit{true} & \text{if } P = \varnothing \text{ and } 0 < \hat{s} \\ \textit{true} & \text{if } P \neq \varnothing \text{ and } \textit{wor}(p,t,n) < \hat{s} \\ \textit{false} & \text{otherwise} \end{cases}$$

$$\textit{wor}(p, t, n) = (1-\beta)\mathbf{P}(t|p) + \beta\mathbf{P}(n|t)$$

If there is no more sentence to be added and $0 < \hat{s}$, t and $\hat{r}$ are appended to D, which is returned (`L4`). If $P \neq \varnothing$ and $\hat{s}$ is greater than the weighted sum of $\mathbf{P}(t|p)$ and $\mathbf{P}(n|t)$, implying that it is more natural to include $\hat{r}$, the next sentence $n \in P$ is prepended by $\hat{r}$ (`L5`). In our case,[7] $\beta = 0.5$ so that $\mathbf{P}(t|p)$ and $\mathbf{P}(n|t)$ get equally weighed. Once R is processed, D is appended by t and returned (`L6`).[8]

[6] Note that $\hat{s} < 0$ if $\lambda < |\hat{r}| - 100$ (in our case, $115 < |\hat{r}|$) that is intended since it is better not to select such long replies.

[7] The hyperparameters β, γ, and λ are optimized by analyzing their performance on the dataset in Table 2.

[8] Since t and $r \frown n$ are introduced as separate utterances in D, estimating $\mathbf{P}(r, n|t) = \texttt{NSP}(t, r \frown n)$ instead of $\mathbf{P}(n|t,r) = \texttt{NSP}(t \frown r, n)$ seems to make more sense. However, we tried many different combinations of n-gram estimations including $\mathbf{P}(n|t,r)$, and the ones presented in this section yielded the best results overall.

3.5 Title Repetition Handling

Often on Reddit, the earlier part of a post assumes the context in the title so that it makes more sense to consider the title the first utterance of Speaker 1. On the other hand, the title and the first sentence in a post can be nearly or exactly the same such that including such a redundant title with the first sentence would lower the naturalness of our generated dialogue. Thus, any title that has a string match of 70% or higher with the first sentence in the post is excluded from the generation.

4 Algorithm Analysis

This section provides an in-depth analysis among the following five methods:

- G_B: Greedy Baseline (Section 3.1)
- G_A: Greedy Advanced (Section 3.2)
- B_2: Beam Search, $k = 2$ (Section 3.3)
- B_4: Beam Search, $k = 4$ (Section 3.3)
- T_2: Threading, $k = 2$ (Section 3.4)

To evaluate the quality of dialogues generated by these methods, diverse posts are collected from the following 10 subreddits:

- `ADV`: Advice
- `BKS`: Books
- `COL`: College
- `CaC`: Casual_Conversation
- `FIT`: Fitness
- `LTM`: LetsTalkMusic
- `MOV`: Movies
- `GAM`: TrueGaming
- `WRT`: Writing
- `TFR`: TalesFromRetail

Table 2 shows the statistics of our analysis set consisting of 50 posts uniformly distributed across the subreddits. On average, posts have 11.4 sentences (200.1 tokens) with 107.8 top-level comments that comprises 3.3 sentences (46.2 tokens), where a comment has 0.7 top-level replies (23.1 tokens).[9] The number of top-level comments varies quite a bit by the popularity of each subreddit.

All dialogues created by the five methods above are assessed by two undergraduates trained for this task. For this assessment, every utterance in these dialogues is double-annotated for whether it is appropriate for the context; more explicitly: "Is this a normal response or continuation of the previous statement?". This metric is chosen to evaluate the quality of dialogues as closely aligned with human dialogue-related intuition as possible.

[9]Tokens are split by whitespace, not linguistically.

Subreddit	Posts	Comments	Replies
`ADV`	5 (16.9)	36.3 (5.3)	0.2 (38.0)
`BKS`	5 (7.9)	591.4 (2.8)	0.6 (14.9)
`CaC`	5 (8.4)	142.4 (3.0)	0.7 (27.3)
`COL`	5 (7.1)	30.8 (3.9)	0.5 (39.3)
`FIT`	5 (3.9)	155.5 (3.5)	0.8 (27.7)
`GAM`	5 (19.0)	32.9 (6.5)	0.9 (52.0)
`LTM`	5 (13.6)	23.2 (5.5)	0.7 (49.0)
`MOV`	5 (8.8)	22.7 (2.5)	0.6 (21.5)
`RET`	5 (22.4)	23.5 (3.0)	1.0 (35.5)
`WRT`	5 (5.1)	35.0 (4.8)	0.3 (40.0)
Total	**50** (11.4)	**107.8** (3.3)	**0.7** (23.1)

Table 2: The analysis set used to evaluate our generation methods. Comments/Replies: the average number of top-level comments/replies per post/comment, (*) in Posts & Comments: the average number of sentences, (*) in Replies: the average number of tokens.

Table 3 shows the analysis results. Note that only the first 10 sentences in every post are used to create dialogues for this analysis to fairly score ones from distinct subreddits, although posts from certain subreddits (e.g., `FIT`, `WRT`) do not have 10 sentences on average, yielding shorter dialogues.

	κ	σ	UT	TK_1	TK_2	TK_a
G_B	55.9	39.1	13.9	20.2	35.4	27.8
G_A	27.1	51.1	11.4	21.2	35.1	28.1
B_2	24.0	67.2	9.8	22.9	36.9	29.9
B_4	45.8	58.9	9.8	23.0	38.1	30.5
T_2	**41.8**	**75.4**	**12.2**	**20.6**	**36.1**	**28.2**

Table 3: The analysis results from the five methods. κ: Cohen's kappa, σ: the avg-score of dialogues, UT: the avg-number of utterances, $TK_{1|2|a}$: the avg-number of tokens in every utterance from Speaker 1, Speaker 2, and all speakers, respectively.

For the inter-annotator agreement, Cohen's Kappa is used, which shows moderate agreements for G_B and T_2, where most utterances are found to be inappropriate and appropriate respectively such that they are easier to assess. The score of the dialogue is measured by macro-averaging the scores of the two annotators as follows ($n = 50$: the number of dialogues, $m = 2$: the number of annotators):

$$\frac{1}{n}\sum_{j=1}^{n}\left(\frac{1}{m}\sum_{i=1}^{m}\frac{\#_i \text{ of appropriate utterances}}{\#\text{ of utterances}}\right)$$

G_A shows a good improvement of 12% over G_B, implying that it is often natural to include multiple sentences to compose utterances for Speaker 1. B_2 shows even a larger improvement over G_A, depicting the effectiveness of beam search, although no

ID	Utterance
S1	How do I get over the loss of a pet? Yes I have loss Pets before and I was sad.. But in this case I had a dog named Oscar he was in my life ever since toddlerhood.
S2	Losing a pet is like losing a family member. I think you might be taking this death harder because he didn't die on his own accord.
S1	I was even the one to name him.
S2	Unfortunately from my personal experience you don't really get over it but as days go on it gets easier. Just allow yourself to feel the emotions and get them out.
S1	Everyday I walked to my great granny's house to see him. He was my best friend seeing him became harder after my great grandmas death.
S2	Give yourself time. Accept your feelings and know that grieving is a process.
S1	He moved to Texas and I was only able to see him every summer.
S2	This will just make your grief and depression deeper and could spark an unending cycle of sadness. Join a support group. Speak with others who are also grieving.
S1	He was in my life srom toddler hood to now 17 years old.
S2	Unfortunately there is no real guideline to how long after the person is deceased, it is normal to be depressed, it will be different for everyone.
S1	He was in Texas and got hit by something being old and scared he wouldn't let anyone work on him.

Table 4: An example dialogue generated by T_2 using a post from the `Advice` subreddit. The original post can be found `https://www.reddit.com/r/Advice/comments/ub7k62`. S1/2: Speaker 1/2.

improvement is made when the bigger beam size is used for B_4. T_2 gives an additional improvement of 8.2% over B_2 and achieves the average score of 75.4%, implying that over ¾ of utterances in these dialogues are found to be appropriate.

For the dialogue lengths, it is not surprising that G_B yields longer dialogues as each sentence in the post is considered a separate utterance. Notice that B_* yield shorter dialogues than the others, indicating that the beam search prefers to combine more sentences from the post to compose utterances for Speaker 1. It makes sense because the `NSP` scores between consecutive sentences from the post are likely higher than ones with comments. However, the length is resolved with T_2 when the replies are concatenated to those sentences so that their `NSP` scores become more comparable to the `NSP` scores with the comments. Utterances for Speaker 2 tend to be longer than Speaker 1's ones. Table 4 shows a dialogue example generated by T_2, where most utterances are found to be appropriate.

5 Multi-turn Dialogue Corpus

With our best method, Threading (Section 3.4), we create a corpus consisting of 10,098 dialogues that can be used to pretrain language models for multi-turn dialogue systems (Section 5.1). Among those, 570 of them are compared to dialogues from three other datasets for quality assurance (Sec. 5.2).

5.1 Corpus Creation

A total of 28,686 posts and their comments/replies are collected from the 10 subreddits in Section 4 using the Python Reddit API Wrapper (PRAW).[10] The corpus creation follows 3 stages, pre-filtering, dialogue generation, and post-filtering. During the pre-filtering, posts that meet any of the following criteria are discarded:

- Include many non-standard characters (e.g., unicode characters, ones not in English)
- Include reddit-specific markers (e.g., `REPLY`, `DELETE`, `EDIT`, `OP`, `TL;DR`)
- Include many URLs, lists, or numbers
- Reference posts or comments in other posts
- The title includes the word 'thread' or the first sentence includes the word 'title'.
- The title is redundant to other posts.

The Threading algorithm is run on pre-filtered posts to automatically generate dialogues. During the post-filtering, any dialogue with the average sequence score (θ/ϕ in Algorithm 4) less than 6.0 gets discarded. Finally, dialogues are assessed by GRADE (Huang et al., 2020), an automatic coherence metric for open-domain dialogue, discarding dialogues with utterances with scores less than 0.21. Table 5 describes the statistics of each stage. 48.5% and 31.7% of the posts are discarded after the pre- and post-filtering, respectively. Our corpus consists of 109,916 utterances and 3,317,807 tokens, which makes it one of the largest dialogue datasets. More importantly, new large corpora can be created for a variety of topics with our method.

[10] PRAW: `https://praw.readthedocs.io`

It is especially useful for those who already have a small dialogue dataset and need a large corpus to pretrain language models for performance boost.

SR	ORG	PRE	POST	UTT	TOK
ADV	6,339	3,078	1,527	10.7	30.1
BKS	2,476	1,419	1,077	10.6	29.7
CaC	3,386	1,959	1,441	9.7	26.7
COL	4,008	1,637	1,117	8.3	30.0
FIT	1,964	422	342	10.5	29.2
GAM	1,873	1,057	632	15.9	37.4
LTM	1,882	1,049	819	12.7	36.1
MOV	2,341	1,417	1,071	8.5	26.0
RET	1,997	1,035	780	18.3	27.7
WRT	2,897	1,701	1,292	8.9	30.4
Total	28,686	14,774	10,098	10.9	30.2

Table 5: The statistics of our dialogue corpus. ORG: the number of collected posts, PRE/POST: the number of posts retained after pre/post-filtering, UTT/TOK: the avg-number or utterances/tokens.

5.2 Dialogue Evaluation

To evaluate the quality of our corpus, 570 of them are selectively sampled by their sizes and sources. Three size groups are formed, *small*, *medium*, and *large*, comprising dialogues with [6, 10], [11, 14], and [15, 17] utterances, respectively. Dialogues in this evaluation set are uniformly distributed across the 10 subreddits for fair comparisons.

Each of our dialogues is displayed with another dialogue with a similar size (and topic if possible) from one of the three datasets, Blended Skill Talk (BST), Daily Dialogue (DD), and Topical Chat (TC) such that a total of 1,710 (570×3) pairs are generated for comparisons.[11] Each pair is then compared by two annotators for *engagingness* and *naturalness* as follows:

Engagingness:
Which dialogue is more engaging or interesting?

Naturalness:
Which sounds more natural or human-like?

- 2: A is significantly more engaging/natural than B.
- 1: A is more engaging/natural than B.
- 0: A is as engaging/natural as B.
- -1: A is less engaging/natural than B.
- -2: A is significantly less engaging/natural than B.

[11] Section 2.1 explains why BST/DD/TC are chosen and gives detailed comparisons between their and our datasets.

The order of Dialogue A and B in each pair is randomly shuffled so that the annotators would not be able to tell their sources. For the annotation, we hired a professional team through SurgeHQ[12] and paid $0.5/task, costing a total of $1,710. Table 6 shows the evaluation results.

	CT	$\mathbf{B}_e$	$\mathbf{B}_n$	$\mathbf{D}_e$	$\mathbf{D}_n$	$\mathbf{T}_e$	$\mathbf{T}_n$
SM	190	0.39	-0.19	0.77	0.01	-0.51	-0.22
MD	190	0.52	-0.14	0.82	-0.03	-0.23	-0.16
LG	190	0.56	-0.28	0.71	-0.07	-0.04	-0.23
ADV	60	0.57	-0.23	0.76	0.05	-0.19	-0.07
BKS	60	0.53	-0.06	0.64	-0.16	-0.2	-0.23
CaC	60	0.38	-0.29	0.68	0.02	-0.39	-0.38
COL	43	0.28	-0.20	0.8	0.07	-0.43	-0.15
FIT	57	0.26	-0.34	0.56	-0.09	-0.53	-0.3
GAM	50	0.67	-0.12	1.05	0.18	0.13	-0.01
LTM	60	0.61	-0.11	1.02	0.16	-0.14	-0.20
MOV	60	0.66	-0.16	0.92	0.17	-0.12	0.00
RET	60	0.41	-0.50	0.43	-0.55	-0.40	-0.45
WRT	60	0.49	-0.04	0.87	-0.10	-0.33	-0.18
All	**570**	**0.49**	**-0.21**	**0.77**	**-0.03**	**-0.26**	**-0.20**

Table 6: The evaluation results. CT: the number of dialogues, B/D/T: BlendedSkillTalk, DailyDialogue, TopicalChat, $*_{e/n}$: the engagingness/naturalness score, SM/MD/LG: the small/medium/large size set.

For each annotation, our dialogue gets the score of 2/-2 if it is significantly better/worse, 1/-1 if it is better/worse, and 0 if it is as good as the other dialogue. The overall score is estimated by averaging all individual scores. In general, our dialogues are more engaging than BST and DD (0.49 and 0.77) but slightly less engaging than TC (-0.26) although longer dialogues are competitive to ones in TC (-0.04). On the other hand, our dialogues are less natural than the others although the differences are marginal (< 0.21). Our dialogues are found to be more natural for 6 out of 10 subreddits compared to DD. Considering how many human labors are involved for the creation of BST and TC while it costs no labor to create our corpus, these results are very promising. Example dialogues from this evaluation can be found in Appendix A.

It is worth mentioning that we first tried crowdsourcing our annotation through Mechanical Turk, which yielded random results as most turkers kept marking only Dialogue A without reading both of them carefully. When we switched the annotation tasks to SurgeHQ, a remarkable improvement was observed although the inter-annotator agreements were still low, 30.8% and 24.4% for the *engagingness* and *naturalness* tests, respectively. Such low

[12] SurgeHQ: https://www.surgehq.ai

agreements have been observed by previous works created the other datasets as well because this task is highly subjective. We will explore a more robust way of evaluating dialogues in the future.

6 Conclusion

We present four algorithms for the automatic conversion of posts and their comments/replies from Reddit discussion forums to multi-turn dialogues. Each algorithm is carefully designed and analyzed for high-quality generation. Our best method can generate dialogues with the 75% appropriateness level. Using this method, a large corpus is created consisting of 10,098 dialogues from 10 subreddits. The quality of our dialogues is tested by comparing them to dialogues from three popular datasets, BlendedSkillTalk, DailyDialog, and TopicalChat. Our dialogues are more engaging, but slightly less natural than those from the other datasets overall.

For future work, we will improve our methods, apply them to broader subreddits, and adapt them to other discussion forums as our methodology is not limited to Reddit. We will also train dialogue models on our corpus for a more in-depth extrinsic evaluation against other dialogue datasets.

Acknowledgements

We gratefully acknowledge the support of the Amazon Alexa AI grant. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of Amazon.

References

Daniel Adiwardana, Minh-Thang Luong, David R. So, Jamie Hall, Noah Fiedel, Romal Thoppilan, Zi Yang, Apoorv Kulshreshtha, Gaurav Nemade, Yifeng Lu, and Quoc V. Le. 2020. Towards a human-like open-domain chatbot. *CoRR*, abs/2001.09977.

Rami Al-Rfou, Marc Pickett, Javier Snaider, Yun-Hsuan Sung, Brian Strope, and Ray Kurzweil. 2016. Conversational contextual cues: The case of personalization and history for response ranking. *CoRR*, abs/1606.00372.

Tom B. Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel M. Ziegler, Jeffrey Wu, Clemens Winter, Christopher Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. 2020. Language models are few-shot learners. *CoRR*, abs/2005.14165.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, Minneapolis, Minnesota. Association for Computational Linguistics.

Emily Dinan, Stephen Roller, Kurt Shuster, Angela Fan, Michael Auli, and Jason Weston. 2019. Wizard of Wikipedia: Knowledge-Powered Conversational Agents. In *Proceedings of the International Conference on Learning Representations*, ICLR.

Xiang Gao, Yizhe Zhang, Michel Galley, Chris Brockett, and Bill Dolan. 2020. Dialogue Response Ranking Training with Large-Scale Human Feedback Data. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 386–395, Online. Association for Computational Linguistics.

John J. Godfrey, Edward Holliman, and Jane McDaniel. 1992. SWITCHBOARD: telephone speech corpus for research and development. In *Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing*, pages 517–520.

Karthik Gopalakrishnan, Behnam Hedayatnia, Qinlang Chen, Anna Gottardi, Sanjeev Kwatra, Anu Venkatesh, Raefer Gabriel, and Dilek Hakkani-Tür. 2019. Topical-Chat: Towards Knowledge-Grounded Open-Domain Conversations. In *Proceedings of the Annual Conference of the International Speech Communication Association*, pages 1891–1895.

Lishan Huang, Zheng Ye, Jinghui Qin, Liang Lin, and Xiaodan Liang. 2020. GRADE: Automatic graph-enhanced coherence metric for evaluating open-domain dialogue systems. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 9230–9240, Online. Association for Computational Linguistics.

Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Veselin Stoyanov, and Luke Zettlemoyer. 2020. BART: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7871–7880, Online. Association for Computational Linguistics.

Yanran Li, Hui Su, Xiaoyu Shen, Wenjie Li, Ziqiang Cao, and Shuzi Niu. 2017. DailyDialog: A man-

ually labelled multi-turn dialogue dataset. In *Proceedings of the Eighth International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 986–995, Taipei, Taiwan. Asian Federation of Natural Language Processing.

Pierre-Emmanuel Mazaré, Samuel Humeau, Martin Raison, and Antoine Bordes. 2018. Training millions of personalized dialogue agents. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 2775–2779, Brussels, Belgium. Association for Computational Linguistics.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J. Liu. 2020. Exploring the limits of transfer learning with a unified text-to-text transformer. *Journal of Machine Learning Research*, 21(140):1–67.

Hannah Rashkin, Eric Michael Smith, Margaret Li, and Y-Lan Boureau. 2019. Towards empathetic open-domain conversation models: A new benchmark and dataset. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 5370–5381, Florence, Italy. Association for Computational Linguistics.

Stephen Roller, Emily Dinan, Naman Goyal, Da Ju, Mary Williamson, Yinhan Liu, Jing Xu, Myle Ott, Eric Michael Smith, Y-Lan Boureau, and Jason Weston. 2021. Recipes for building an open-domain chatbot. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*, pages 300–325, Online. Association for Computational Linguistics.

Eric Michael Smith, Mary Williamson, Kurt Shuster, Jason Weston, and Y-Lan Boureau. 2020. Can you put it all together: Evaluating conversational agents' ability to blend skills. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 2021–2030, Online. Association for Computational Linguistics.

Jing Xu, Arthur Szlam, and Jason E. Weston. 2022. Beyond Goldfish Memory: Long-Term Open-Domain Conversation. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (ACL)*. Association for Computational Linguistics.

Saizheng Zhang, Emily Dinan, Jack Urbanek, Arthur Szlam, Douwe Kiela, and Jason Weston. 2018. Personalizing dialogue agents: I have a dog, do you have pets too? In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 2204–2213, Melbourne, Australia. Association for Computational Linguistics.

Yizhe Zhang, Siqi Sun, Michel Galley, Yen-Chun Chen, Chris Brockett, Xiang Gao, Jianfeng Gao, Jingjing Liu, and Bill Dolan. 2020. DIALOGPT : Large-scale generative pre-training for conversational response generation. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics: System Demonstrations*, pages 270–278, Online. Association for Computational Linguistics.

A Appendix

A.1 Example Dialogues from Each Evaluation Grading Category

This section give example dialogues rated as "Less Natural, Less Engaging" (Table 7), "More Natural, More Engaging" (Table 8), "Significantly More Natural, Significantly More Engaging" (Table 9), and "Significantly Less Natural, Significantly Less Engaging" (Table 10). Note that these ratings are given in relation to a conversation, omitted here, from one of our comparison datasets (Section 5.2).

The negatively-rated dialogues exhibit several of our methodology's weaknesses, namely lack of interaction from Speaker 1, Speaker 2 responses to content which has not yet been introduced by Speaker 1, and references to Reddit-specific terminology such as threads which wouldn't be discussed often in regular dialogue.

The positively-rated dialogues, on the other hand, demonstrate our methodology's strengths: both successfully use threading (e.g., Utterance 5 in Table 8 and Utterance 3 in Table 9), there is consistently natural grouping of Speaker 1's content, and Speaker 2 responds for the most part at natural times to the content Speaker 1 introduces while also adding relevant content to the conversations.

ID	Utterance
S1	Bittersweet musical legacies: an examination. The popularity of the recent Big Star thread got me thinking about how bittersweet it is for music fans to deeply love a band that were either struck down by tradegy, failed to launch due to bad decisions, or never received their day in the sun beyond a cult following.
S2	I'd say Badfinger certainly qualifies for this thread. They had the support of the Beatlesĺabel, they looked to be on the road to a successful career with many big hits like ""Day After Day"", ""No Matter What"", and ""Come and Get It"" (and their song ""Without You"" was made a big hit by Nilsson)
S1	One of the saddest legacies for me is Emitt Rhodes, a guy who at the tender age of 20 in 1970 put out an album with musical chops worthy one Sir Paul McCartney.
S2	I would argue The Kinks belong in this thread. Yeah they have some songs that poked through the fog (You Really Got Me, Lola, Sunny Afternoon), but they should really be considered right in the mix in the Beatles/Stones conversation. I feel like The Kinks are not given that same respect but they deserve it.
S1	Having that level of widespread public acknowledgement is quite rare. A cynic may say he cribbed macca's sound, and in all honesty a song could've appeared on his first solo album, but at the age of 20 he had many years ahead of him to transcend his influences....except that he didn't.
S2	...and then they got royally screwed over financially, had a lot of issues within the band (I believe the royalties to "Without You" were among the reasons they fought), and because of all the fallout of the financial ruin and such, in the end, two of the band members ultimately committed suicide in eerily similar fashion. Such a sad, sad story.
S1	The fact he had a kid on the way when he hanged himself is tragic. A handful of albums into his career he was beset by lawsuits and record company entanglements that saw him walk away from releasing music in 1973 and not appear again with a new album until 2016(!).. If you look at the photos of Emitt and then jump to the album Rainbow Ends, it's like a joyful/sad pair of bookends for a man who in interviews talked about his regrets with music and his lack of success.
S2	I also think of The Verve and how the copyright shenanigans robbed them of their hit Bittersweet Symphony. And then that was kind of it for them.
S1	I loved Urban Hymms, listened to that record so much over my life. This is made even more sad by the fact he died in his sleep last year of heart failure at age 70.
S2	These are examples of artists who missed out on their full commercial potential in life rather than good artists that we like to think could've been great artists because the circumstances weren't right. I'm not familiar enough with Emmitt Rhodes to say whether he fits in either camp, just my two cents on the issue.
S1	I'd add Townes Van Zandt to that list. His talent as a songwriter was undeniable.
S2	Sadly the music industry is littered with tragic stories and acts that should have been much larger than they were, but for whatever reason, weren't. One of my favorite ""one hit wonders"" was a dude named Jonathan Edwards who had a hit with in 1971 about the craziness of the music industry and that was pretty much it despite making great music up into the 80s.
S1	The press called him a one man Beatles due to the fact he played and recorded all his own instruments at his home studio.
S2	his story is particularly sad, simply because he faced so much tragedy and hardship, and no one was really there for him. the one album he put out is a reflection of his life and mental health issues and is as equally as beautiful as it is heartbreaking. tim buckley, on the other hand, released 9 albums over the span of a 9 year career and could've have become a musical legend similar to that of a lot of psychedelic folk/folk rock artists of the time.

Table 7: An example dialogue generated by T_2 using a post from the `LetsTalkMusic` subreddit that was graded as less natural and less engaging. The original post can be found at https://www.reddit.com/r/LetsTalkMusic/comments/pnwg25/bittersweet_musical_legacies_an_examination/. S1/2: Speaker 1/2.

ID	Utterance
S1	Do you use a bench/machine if someone says "I'm still using it" when they've left nothing there to show? I see the benchpress spot free and no sign of being used.
S2	so I can use it. Then walk away and use any other machine.
S1	No hoodie, no towel, no water bottle nothing.. So I put my hoodie down on the ground while I go get a paper towel to clean some sweat on the bench.. Then, some guy says ""Hey I'm still using that""
S2	If he's on another machine he's not on the one you want to use. Use it.
S1	Idek how to sue lol. He's about 25 feet away just standing and talking to someone.
S2	Just ignore him entirely and bench lol If the guy wants to say something, he can have some manners and come over instead of shouting over.
S1	His tone was kinda rude too, as if it was obvious that the damn thing's still in use.
S2	If it becomes an issue, just keep an eye out for whatever he's doing and as soon as you see him go to any piece of equipment, yell Hey, I'm using that. . . even if you are in the middle of your set, in another machine, across the room. On the other hand, maybe the guy is not a complete jerk and he really was using it and it's no big deal.
S1	Stuff gets annoying
S2	Screw that guy. Getting knocked out isn't that bad lol. Hell you may even get a bunch of money out of it.
S1	I swear.
S2	Nothing by on equipment? Take it, get your sets, and remain oblivious to obnoxious behavior.

Table 8: An example dialogue generated by T_2 using a post from the `fitness` subreddit that was graded as more natural and more engaging. The original post can be found at https://www.reddit.com/r/Fitness/comments/tth3in/do_you_use_a_benchmachine_if_someone_says_im/. S1/2: Speaker 1/2.

ID	Utterance
S1	Smartest Theft Plan That Almost Worked. When I was 17, I had a part-time job as a cashier for Rona.. During my time there I met some very interesting people at the till.
S2	Best concealment I ever saw on the job as a cashier is some guy tried to fill a beanbag chair with vitamin bottles...
S1	Thats some clever thinking! Once, a man walked in wearing what looked like a Halloween construction worker costume and headed for what looked like the power tools section.. Although, I later found out he went to the cleaning aisle first where we sell commercial yellow mops and buckets.. He opened one box and put the mop and bucket on the sales floor to make it look like it belonged to the store.. Then he used the empty box and filled it with expensive power tool kits and batteries.
S2	He assembled them in his cart and loaded them up with all the toys he wanted. Then tried to just pay for two boxes that come flat packed and obviously had to be opened. Surprise surprise they were heavy.....
S1	11 times. He had the box in a shopping cart and went up to my till.
S2	When I worked at a big box hardware store we one day had a guy come in and grab some of the $5 heavy duty moving boxes.
S1	This week. This was an elaborate plan up till now, unfortunately, he placed the box the wrong way in the cart where I couldn't just use my scanner to get to the barcode.. He tried to turn the box but since he put all these heavy tools and batteries in it, he couldn't do it.
S2	Hilarious. Pretty smart. Wonder if you would have caught him if he had put the box in the right direction.

Table 9: An example dialogue generated by T_2 using a post from the `TalesFromRetail` subreddit that was graded as significantly more natural and significantly more engaging. The original post can be found at https://www.reddit.com/r/TalesFromRetail/comments/ppsa29/smartest_theft_plan_that_almost_worked/. S1/2: Speaker 1/2.

ID	Utterance
S1	Your money is gone! Please calm down! So, I recently got a job at a store that sells cheap stuff for around a dollar or more.
S2	The grocery store I go to has a sign at each till saying don't put money on the belt as it could cause damage. Any money lost is not the responsibility of the store. Yet, idiots still do it, but their clearly posted sign covers them and they just shrug their shoulders.
S1	Most customers are usually polite and very pleasant to talk to.
S2	In Finland some stores have signs that explicitly says no money or recycling slips on the belt, probably because alot of people used to put their recycling slips in with the groceries.
S1	Im not a fin (sorry) what is a recycling slip? Some however, make me seriously regret accepting this job.. Today I had one of those customers... A woman came into my line today with one thing and, she did the dumb thing of putting her cash on the belt.. Which you should never do because the belt acts like a vending machine when you put paper money on it.. Henceforth the belt took her $5 bill and she instantly started flipping out like crazy, and I mean she was hysterical.
S2	My brother had something similar happen at a grocery store but the customer said it was a $100 bill and my brother didn't witness it.
S1	A hundred is much worse than a $5. She stated how she wasn't leaving until she gets her five dollars back.
S2	The worst thing I've seen one of those belts try to eat was a 2L of coke. The bottle fell on its side on the belt and got caught and poked a hole in the side. Then the bottle kept rolling at the end of the belt spraying soda all over the place.
S1	There is no service desk. Then she starts making a list of demands such as... - wanting to talk to the owner of the store.
S2	Everyone just kinda froze until the cashier stopped the belt and someone grabbed the bottle and put a fingerbover the hole until they could throw it out. It was horrible and hilarious at the same time.
S1	The store I work in is very small.

Table 10: An example dialogue generated by T_2 using a post from the `TalesFromRetail` subreddit that was graded as significantly less natural and significantly less engaging. The original post can be found at https://www.reddit.com/r/TalesFromRetail/comments/tx1fg0/your_money_is_gone_please_calm_down/. S1/2: Speaker 1/2.

ConFiguRe: Exploring Discourse-level Chinese Figures of Speech

Dawei Zhu **Qiusi Zhan** **Zhejian Zhou** **Yifan Song**
Jiebin Zhang **Sujian Li***
School of Computer Science, Peking University
Key Laboratory of Computational Linguistics, MOE, Peking University†

Abstract

Figures of speech, such as metaphor and irony, are ubiquitous in literature works and colloquial conversations. This poses great challenge for natural language understanding since figures of speech usually deviate from their ostensible meanings to express deeper semantic implications. Previous research lays emphasis on the literary aspect of figures and seldom provide a comprehensive exploration from a view of computational linguistics. In this paper, we first propose the concept of figurative unit, which is the carrier of a figure. Then we select 12 types of figures commonly used in Chinese, and build a Chinese corpus for Contextualized Figure Recognition (ConFiguRe). Different from previous token-level or sentence-level counterparts, ConFiguRe aims at extracting a figurative unit from discourse-level context, and classifying the figurative unit into the right figure type. On ConFiguRe, three tasks, i.e., figure extraction, figure type classification and figure recognition, are designed and the state-of-the-art techniques are utilized to implement the benchmarks. We conduct thorough experiments and show that all three tasks are challenging for existing models, thus requiring further research. Our dataset and code are publicly available at https://github.com/pku-tangent/ConFiguRe.

1 Introduction

Figures of speech, also known as rhetoric figures or figurative languages, are a ubiquitous part of spoken and written discourse. These rhetorical techniques, such as metaphor, irony and parallelism, greatly enrich the expression of human languages (Roberts and Kreuz, 1994). They intentionally deviate from the literal meaning of language to provide deeper semantic expressiveness, therefore posing a big challenge to natural language understanding.

Discourse Fragment: {...} [1] 这样的想了一想，[2] 我的断绝了联络的知觉，[3] 又重新恢复了转来，{[4] *一股同蒸气似的酸泪，*} metaphor/simile（比喻） [5] 直涌了出来。[6] 我踉跄往后退了几步，[7] 倒在外床她叠好在那里的那条被上。[8] 两手紧紧抱着了这一条被，{[9] *我哭着，*[10] *哭着，*[11] *哭着，*[12] *哭了一个尽情。*} Parallelism（排比） {...}

Translation: {...} Thinking of this, my disconnected senses started to restore again, [*and a stream of sour tears gushed out like steam.*]$_{\text{metaphor/simile}}$ I stumbled back a few steps and fell out of the bed on the quilt she folded there. Holding this quilt tightly with both hands, [*I cried, cried, cried and cried my heart out.*]$_{\text{Parallelism}}$ {...}

Figure 1: An annotated case of ConFiguRe. We present English translation in the second block for reading convenience. Clause groups in *italic* are figurative units, with the specific figure type denoted in subscription.

Linguists have a long history of studying rhetoric figures, extensively analyzing their use in literature, culture and psychology (Zhang, 1963; Wilks, 1975; Drew and Holt, 1998; Group, 2007; Shapin, 2012). These works mainly focus on collecting qualitative evidence. Figures of speech have also drawn attention from the NLP community. Many downstream applications could be improved, should figures be precisely identified and carefully dealt with. For example, a faithful translation should adapt the metaphors used in the source language to an authentic expression in the target language, and sentiment analysis should benefit from the correct identification of irony and sarcasm.

Despite its significance, a comprehensive study of identifying rhetoric figures from discourse remains under-explored. Previous works mainly emphasize specific figures, such as metaphor (Steen, 2010; Fass, 1991; Dodge et al., 2015; Su et al., 2020) and sarcasm (Khodak et al., 2017; Davidov et al., 2010; Wallace et al., 2014; Lee et al., 2020),

*Corresponding author
† Author emails:{dwzhu, zhanqiusi, zhouzhejian, yfsong, zhangjiebin, lisujian}@pku.edu.cn

Proceedings of the 29th International Conference on Computational Linguistics, pages 3374–3385
October 12–17, 2022.

or identify rhetoric figures at a token or sentence level (Wen et al., 2019; Chen et al., 2021). However, in real-world settings, figurative languages are hidden in a long context and the potential figure type is usually unknown. Motivated by this, we construct a comprehensive dataset of 12 commonly used figures in Chinese, and include discourse fragment as context for each instance.

First we describe our guideline towards constructing a rhetoric corpus, which is devised upon linguistic theory and existing reading practice. The design of the guideline is oriented with two key questions: (1) What is the language carrier of a specific figure? (2) Which figures should be included in our corpus? For the first question, we propose the concept of *figurative unit*——the smallest continuous clause sequence containing a complete expression of a specific figure. For the second question, we approach it from a linguistic view. Linguistically, figures can be divided into two main groups: *schemes* and *tropes*. *Schemes* reflect a deviation from the ordinary pattern or arrangement of words, while *tropes* involve deviation from the ordinary and principal signification of words (Corbett, 1999). Following previous work of Chinese linguists (Zhang, 1963) and existing reading-comprehension practice, we select 7 tropes and 5 schemes commonly used in Chinese in our work.

Following the aforementioned guideline, we leverage human annotation to build ConFiguRe, a Chinese corpus for Contextualized Figure Recognition. Each instance in ConFiguRe includes a discourse fragment with several annotated figurative units attached to it. An annotated instance of ConFiguRe is illustrated in Figure 1. In this instance, the fragment includes 12 Chinese clauses and 2 figurative units. Clause 4 alone is a figurative unit labeled with the *metaphor* type, while the latter 4 clauses constitute another figurative unit labeled *parallelism*. ConFiguRe is, to the best of our knowledge, the first rhetoric dataset that involves both extracting a figurative unit from the discourse-level context and classifying this unit into the right figure type. In comparison, previous datasets mainly focus on detecting a specific figure from a given sentence (Joshi et al., 2016; Khodak et al., 2017).

For benchmark settings, we design three tasks based on ConFiguRe, namely *figure extraction*, *figure type classification*, and *figure recognition*. We deploy state-of-the-art models as baselines, and reveal that all three tasks remain challenging through thorough experiment. We also conduct subsidiary experiments to explore future directions for these tasks, which will contribute to the research of this area.

To sum up, our main contribution is threefold:

- We design the guideline towards constructing a discourse-level rhetoric corpus based on linguistic theory, and propose the concept of *figurative unit* as the basic element for analysis.
- We construct ConFiguRe, a human-annotated Chinese corpus for contextualized figure recognition, which includes 12 most frequently used figure types. Upon this, we design three tasks as benchmarks: *figure extraction*, *figure type classification*, and *figure recognition*.
- We implement models based on recent state-of-the-art techniques as baselines, and conduct thorough experiments and analysis. We find that all three tasks on ConFiguRe are challenging with a lot of room to improve.

2 Related Work

Over the last decade, automated detection of figurative languages has become a popular topic, and a considerable number of datasets in this area have been constructed. These datasets can be roughly divided into two categories: the first is to extract the span of a specific figure given a sentence and its context (span extraction); the second is to determine whether a sentence is figurative (sentence classification). For span extraction, it is usually accomplished by marking out tokens carrying the target figure. These works include VUA (Joshi et al., 2016) and the NTU Irony Corpus (Tang and Chen, 2014). For sentence classification, there are two lines of relevant research. The first is a binary sentence classification task for determining whether a given sentence is figurative, e.g. SARC (Khodak et al., 2017) for sarcasm, and the Chinese rhetoric question dataset built by Wen et al. (2019). The second is to classify a figurative sentence into its corresponding figure type (multi-classification), such as the Chinese multi-label rhetoric dataset constructed by Chen et al. (2021) for joint rhetoric and emotion identification.

On the basis of existing datasets, a series of methods have been developed, ranging from feature engineering (Bulat et al., 2017; Köper and Schulte im

Walde, 2017; Tsvetkov et al., 2014) to neural networks (Liu et al., 2018; Chen et al., 2021; Mu et al., 2019; Dankers et al., 2020; Joshi et al., 2017; Leong et al., 2020, 2018). However, perhaps limited by the fact that existing datasets only target at one figure, previous works mainly deal with specific figure types. A comprehensive study of a collection of figure types remain under-explored.

One further observation is that previous datasets are rarely shipped with wider contextual information. They mainly approach the subject from a token-level or sentence-level perspective. That being the situation, a series of works have pointed out that leveraging contextual information is beneficial in figure detection (Dankers et al., 2020; Mu et al., 2019; Jang et al., 2015; Joshi et al., 2017, 2015). Joshi et al. (2015) proposed to use text incongruity from linguistic theory for sarcasm detection. Dankers et al. (2020) used a general and a hierarchical attention mechanism for modeling discourse, improving SOTA for the 2018 VU Amsterdam (VUA) metaphor identification shared task (Leong et al., 2018) by 6.4 F1-scores. Other contextual clues such as author context (Bamman and Smith, 2015; Ghosh and Veale, 2017), multimodal context (Schifanella et al., 2016; Castro et al., 2019), conversation context (Joshi et al., 2016; Ghosh et al., 2017), have also been proven to improve figure detection.

In this paper, we propose Chinese Corpus for Contextualized Figure Recognition (ConFiguRe), which, to the best of our knowledge, is the first comprehensive corpus that includes more than 10 commonly used figures in Chinese, with relevant discourse fragments serving as contextual information for each instance. Our dataset can be used for both figure extraction and figure type classification.

3 Corpus Construction

Previous rhetorical studies (Group et al., 2007; Harris et al., 2018) have contributed many useful annotation paradigms. However, These annotation paradigms are primarily customized for one specific figure type, or tailored to scheme figures that present themselves with strict patterns, which are quite inapplicable in our setting. In this section, we present our self-devised annotation guideline to build ConFiguRe with 7 trope figures and 5 scheme figures.

1. Metaphor/Simile: 阳光像金子一样珍贵，
(Sunshine is as precious as gold)

2. Parallelism: 童年是小草的芽儿，充满了生机；童年是早晨的太阳，充满了活力；童年是清润的雨水，充满了欢乐。(Childhood is the bud of grass, full of vitality; childhood is the morning sun, full of vitality; childhood is the rain, full of joy.)

Figure 2: Two examples for figurative units. First is a *Simile* unit containing one single clause, the second is a *Parallelism* unit composed of six clauses.

3.1 Annotation Guideline

Figurative Unit For the convenience of analyzing figures, we first posit the concept of *figurative unit* as the basic language carrier of a figure. A *figurative unit* is defined as "the smallest continuous clause sequence carrying a complete expression of a specific figure". [1] The intuition is that, under most circumstances, a figure instance only comprises a limited portion of a sentence. Hence, we prefer clauses as elementary constituent of a figure, as it is more fine-grained than a sentence. In Figure 2, we demonstrate two examples of figurative units with their corresponding figure types. In the first example, a single clause is a figurative unit carrying the figure *Simile*; while in the second example, six clauses together form a figurative unit of *Parallelism*.

Figure Types Upon selection of figure types, we refer to the linguistic categories from English (Burton, 2016) and Chinese (Zhang, 1963). We choose the most frequently used ones in written literature. It should be noted that while some figures are widely used in English, they do not have exact counterparts in Chinese, therefore excluded from our dataset. At the same time, we avoid choosing the figures whose identification may involve deep semantic background, e.g. Pun, Paradox, and leave them for future study. Specifically, we adopt the following 12 figures: *Metaphor/Simile*[2], *Personification*, *Metonymy*, *Hyperbole*, *Irony*, *Synaesthesia*, *Rhetorical question*, *Parallelism*, *Duality*, *Repetition*, *Antithesis*, *Quote* (For detailed description of each figure, see Appendix A.1). For simplicity, we temporarily use the first four or five characters as

[1] It is debatable about the definition of a Chinese clause. Here for simplicity, we define a clause as a text span separated by the separator punctuations. A full list of punctuations is included in Appendix A.2.

[2] In Chinese, these two figures are combined as a single figure type which means a comparison made by referring to one thing as another.

the abbreviation of each figure in this paper. Note that we do not aim at including all figure types in our research, but hope to recognize the commonly-used figures out of the discourse.

In linguistics, figures are divided into two main groups: *schemes* and *tropes*. *Schemes* reflect a deviation from the ordinary pattern or arrangement of words, while *tropes* involve deviation from the ordinary and principal signification of words (Corbett, 1999). Among the 12 commonly used figures, the first 7 are *tropes* and the last 5 are *schemes*. We adopt this categorization to facilitate following analysis below.

3.2 Data Collection

Though a common technique in communication, figurative languages are actually sparsely distributed in main-stream corpora, consisting of news, dialogs, etc. For a higher proportion of figurative languages, we build our dataset on a collection of literary works. 98 Chinese literary works, most of which are novels and proses, are collected from publicly available resources.

We then divide each literary work into fragments comprising of whole paragraphs, while also ensuring that each fragment contains no more than 500 words. In this way, we assert that the discourse is neither too short to miss important contextual information nor too long to distract annotators' attention. In total, we obtain 12,976 fragments. These fragments are later presented to human annotators for manual annotation.

3.3 Human Annotation

We recruit 17 annotators whose native language is Chinese. These annotators are all well-educated, mostly majoring in linguistics. We divide them into two groups as *fragment annotators* and *figurative unit annotators*, provide instructions with detailed definitions and examples of each figure type, and train them for figure annotation before setting out on the full dataset. The annotation process is carried out coarse-to-fine in two stages.

The first stage is for coarse classification, where 5 fragment annotators are asked to classify fragments as figurative or not. Each fragment is annotated by one annotator only once, since we suppose classifying a discourse fragment as figurative or not is a binary classification task that is relatively easy.

The second stage involves 12 figurative unit annotators. Each figurative fragment is presented to two annotators, who are independently required to extract figurative units (one fragment can contain several units) and assign the corresponding figure types. We make sure that each figurative unit can only be labeled with one figure type. Additionally, conflict-solving strategies are devised in case of inconsistent labeling. For example, suppose annotators A_1 and A_2 respectively extract figurative units u_1 and u_2, the inconsistency can be roughly divided into 3 categories: 1) u_1 and u_2 are figurative units that do not overlap, in this case we keep both of them. 2) u_1 is a subset of u_2, and they are assigned the same figure type, in this case we choose u_1 as the gold annotation, since figurative unit is defined to be the smallest clause group containing a complete expression of a specific figure. 3) u_1 overlaps with u_2 but is neither a superset or subset, or they are assigned with different figure types, we regard this case as the most complex one and ask another annotator to make a decision based on the annotation results of annotators A_1 and A_2. Besides, if an annotator extracts several figurative units in one go, inconsistency can be solved similarly.

In addition to identifying the figurative units and their types, we also ask the annotators to corroborate their judgement with evidence (e.g. strong feature words denoting the figure) which may benefit future work.

The main reason of designing this coarse-to-fine annotation process is that the work of fragment annotators can narrow down the context of a figure and make the figurative unit annotators pay more attention to identifying the boundary and type of a figure.

Split	# frag.	wd./frag.	cls./frag.	# figUnit.	figUnit./frag.
train	2934	419.7	41.9	6254	2.1
valid	419	417.5	42.1	886	2.1
test	839	419.2	41.3	1870	2.2
Total	4192	419.4	41.8	9010	2.1

Table 1: Statistics of train, valid and test set in our ConFiguRe. *frag.*, *wd.*, *cls.*, *figUnit.* is short for fragment, word, clause, figurative unit, repectively.

3.4 Dataset Analysis

Through human annotation, we present ConFiguRe with 4,192 figurative fragments and 9,010 figurative units. Note that each instance in ConFiguRe is a discourse fragment carrying figurative languages.

Figures	# figurative unit			words/unit	clauses/unit
	train	valid	test		
Meta.	2226	306	683	18.5	1.55
Pers.	768	94	243	19.5	1.64
Meto.	411	69	123	18.3	1.68
Hyper.	473	72	145	19.5	1.75
Irony	24	4	5	27.5	2.36
Syna.	18	6	10	22.9	1.76
Rheq.	868	115	202	19.5	2.07
Para.	291	41	99	37.1	3.90
Dual.	272	37	76	16.8	2.24
Repe.	382	57	105	16.6	2.98
Anti.	179	23	62	28.7	2.73
Quote	342	63	117	33.3	3.90
Total	6254	886	1870	20.7	2.05

Table 2: Statistics of each figure in train, valid and test set. For simplicity, we temporarily use the first four or five characters as the abbreviation of each figure in this paper. First seven figures are *tropes*, last five figures are *schemes*.

Each fragment may contain one or more annotated figurative units. Table 1 demonstrates basic statistics of the train, valid and test set, according to the split proportion of 7:1:2.

Detailed information of each figure type is rendered in Table 2. We can see that *metaphor/simile*, *personification*, and *rhetoric question* occur more frequently than others, while *synaesthesia* and *irony* are extremely hard to gather in our corpus. For these two figure types, we may collect more instances in the future. From Table 2, we can also see that the average number of clauses for *schemes* is consistently larger than that of *tropes*. This can be attributed to the fact that *schemes* are usually only identifiable in a sequence of clauses, while *tropes* can be directly expressed within one clause. Interestingly, the average word number of *irony* figures is comparable to or even longer than many scheme figures. A priori is that the expression of an *irony* requires more contextual information. Scheme figures *duality* and *repetition* are relatively short, probably due to the fact that *duality* often leverages short clauses and *repetition* evinces at the word level.

4 Task Definition and Baseline Models

Based on ConFiguRe, we propose three benchmark tasks. Task 1 is figure extraction, which extracts figurative units from input text. Task 2 is figure type classification, which classifies a figurative unit into the corresponding figure type. Task 3 is figure recognition composed of previous two tasks, which extracts figurative units from input text and determines their corresponding figure types simultaneously.

Although there have been a lot of methods in rhetoric detection, we find that these methods mostly catered for one specific figure (Leong et al., 2020; Ghosh et al., 2020), and therefore not quite consistent with our task settings. For baseline models, we adopt self-designed approaches backboned with the state-of-the-art RoBERTa model. We give a formal definition for each task and introduce the corresponding baseline models as follows.

4.1 Task 1: Figure Extraction

Given a discourse fragment $\mathcal{D}$ comprising m clauses and n words, let $\mathcal{D} = \{c_i\}_{i=1}^{m}$, where $c_i = \{w_j\}_{j=1}^{n_i}$ refers to the i-th clause consisting of n_i words, and $\sum_{i=1}^{m} n_i = n$. Task 1 aims to extract figurative units $u_1 = (c_{b_1}, \ldots, c_{e_1}), \cdots, u_k = (c_{b_k}, \ldots, c_{e_k})$ from $\mathcal{D}$, where k is the number of figurative units in $\mathcal{D}$, u_i is the i-th figurative unit, and the subscripts b_i and e_i denote the beginning and ending positions, respectively. Note that each figurative unit is the smallest continuous clause sequence carrying a complete expression of a specific figure. (See Section 3.1 for detailed explanation of clause and figurative unit)

To perform figure extraction, we first use RoBERTa encoder and perform clause-wise mean pooling to obtain contextualized representations $\boldsymbol{h_c} = (h_{clause_1}, \ldots, h_{clause_m})$ for each clause, where m is the number of clauses. Based on this, we design the following baselines for Task 1:

FESeq FESeq is implemented by modeling figure extraction as a clause-level sequence labeling task. Following traditional settings, the labels "B" "I" and "O" are assigned to each clause when it is the first clause of a figurative unit, inside but not the first of a figurative unit, and not in a figurative unit accordingly. A classification layer is added on top of RoBERTa, casting hidden representations into 3-dimensional logits for "B", "I" or "O".

FECRF Since Conditional Random Field (Lafferty et al., 2001) is a common strategy in sequence labeling tasks, we design the FECRF model by adding a CRF layer on top of FESeq model.

FESpan FESpan is developed by modeling figure extraction as a clause-level binary span tagging task. It adopts two binary MLP classifiers upon

encoder to detect the start and end position for each figurative unit, respectively. More precisely, FESpan assigns each clause a binary tag (0/1), which indicates whether the current clause corresponds to a start or end position of a figurative unit.

4.2 Task 2: Figure Type Classification

Given a figurative unit comprising n words $\boldsymbol{u} = \{w_i\}_1^n$, and its context comprising m words $\mathcal{C} = \{w_j\}_1^m$, task 2 aims to classify this unit $\boldsymbol{u}$ into the right figure type.

First, the RoBERTa encoder produces contextualized representations h_i for each token w_i:

$$\boldsymbol{h} = (h_1, \ldots, h_n) = \text{Encoder}(w_1, \ldots, w_n) \quad (1)$$

Mean pooling is then applied on $\boldsymbol{h}$ to get hidden representation $h^{\boldsymbol{u}}$ for the figure unit $\boldsymbol{u}$. Based on this, we design the following baselines for task 2:

FTCLS FTCLS is built by adding a classification head on top of RoBERTa, which then classifies the input into one of 12 figure types.

FTCXT Since context information has proven to yield improvement for metaphor and sarcasm detection (Dankers et al., 2020; Mu et al., 2019; Joshi et al., 2016; Ghosh et al., 2017), FTCXT is designed to incorporate contextual information in figure classification and explore its effect across all figure types. Specifically, hidden representation $h^{\mathcal{C}}$ for context $\mathcal{C}$ is also calculated by the encoder. $h^{\boldsymbol{u}}$ and $h^{\mathcal{C}}$ are then concatenated for classification.

$$f = \text{Classification}(\left[h^{\boldsymbol{u}}; h^{\mathcal{C}}\right]) \quad (2)$$

where f is the predicted figure type.

4.3 Task 3: Figure Recognition

Similar to task 1, given a discourse fragment $\mathcal{D}$ comprising m clauses and n words, Task 3 aims to extract figurative units $(u_1, \cdots, u_k)$ from $\mathcal{D}$, and classify each figurative unit into a specific figure type as $f_1, \cdots, f_k$ in the same time, where k is the number of figurative units in text $\mathcal{D}$.

Following task 1, contextualized representations $\boldsymbol{h_c} = (h_{c_1}, \ldots, h_{c_m})$ for the m clause are produced by RoBERTa encoder. Based on this, we define following baselines:

Rule-based Method Some figures in our dataset manifest obvious patterns. For example, the type of *Metaphor/Simile* usually comes with indicators such as the Chinese words "像 (like)" , "如 (as)". The *Parallelism* type is composed of similar clauses, most commonly separated by colon. To exploit these obvious patterns, we design heuristic rules for figure recognition as a complement to our neural methods.

Pipeline Since figure recognition can be naturally tackled as first extraction and then classification, we set our pipeline baseline as the combination of best-performing models in extracion task and classification task. Specifically, we first use FECRF to extract figurative units from input text, then use FTCXT to classify the extracted figurative units into their corresponding figure types.

E2ESeq By modeling figure recognition as a sequence labeling task, E2ESeq model shares the same architecture as the FESeq model mentioned before. The difference is that, in this case, we assign clauses of different figure types with different "B" and "I" labels. For instance, for a figurative unit of irony, we assign "B-Irony" to its first clause and "I-Irony" to other clauses in this unit.

E2ECRF Similar to FECRF, we add a CRF layer on top of the above E2ESeq model to construct a E2ECRF model.

5 Experiments

Implementation Details We implement our models using HuggingFace's Transformers (Wolf et al., 2020). We choose the RoBERTa-zh-Large (Cui et al., 2020) checkpoint trained on Chinese corpus. For fine-tuning, we generally stick to a dropout rate of 0.1, a batch size of 16, an epoch of 30, the Adam (Kingma and Ba, 2017) optimizer, and a learning rate of $1e^{-5}$. We select our hyperparameters based on the best performance on validation set and report the average results from 5 runs with different random seeds.

We present main experiment results on our dataset for all three tasks in Table 3. For the extraction task, we report precision, recall and Micro F1 score for each model. For other tasks, we additionally report Macro F1 score by averaging out F1 scores of all figure types. We also conduct subsidiary experiments and analysis for each task. By doing so, we shed light on promising directions for future work.

5.1 Evaluating Figure Extraction

Model performance for figure extraction is presented in the first block of Table 3. FECRF yields the best result, surpassing FESeq by 1.27 F1 score and FESpan by 1.77 F1 score, suggesting that figure extraction can be modeled as a sequence label-

Task	Model	Precision	Recall	Micro F1	Macro F1
Extraction	FESeq	34.06	29.23	31.46	-
	FECRF	32.56	**31.60**	**32.07**	-
	FESpan	**39.41**	24.61	30.30	-
Classification	FTCLS	78.82	78.82	78.82	65.22
	FTCXT	**79.49**	**79.49**	**79.49**	**68.72**
Recognition	Rule	7.74	12.76	9.64	5.24
	Pipeline	**31.10**	25.52	28.03	18.17
	E2ESeq	29.21	25.97	29.47	17.35
	E2ECRF	30.87	**31.10**	**30.99**	**21.27**

Table 3: Main experiment results. We highlight the highest numbers among models in **bold**.

Figure	Extraction (FESeq)	Recognition (E2ECRF)
Exact match	572	563
Wrong figure type	-	75
Super prediction	201	215
Sub prediction	289	227
Overlapping prediction	40	45
Non-lapping prediction	665	699
Total predictions	1757	1824

Table 4: Error analysis for model predictions in extraction and recognition task.

Figure	FTCLS			FTCXT		
	P	R	F1	P	R	F1
Meta.	84.16	87.12	85.61	83.40	88.29	85.78
Pers.	66.80	67.08	66.94	66.27	67.99	67.07
Meto.	62.24	49.59	55.20	66.67	47.15	55.24
Hyper.	57.66	54.48	56.03	58.06	49.66	53.53
Irony	50.00	20.00	28.57	50.00	40.00	44.44
Syna.	0.00	0.00	0.00	25.00	10.00	14.29
Rheq.	93.75	96.53	95.12	92.89	97.03	94.92
Para.	79.66	94.95	86.64	82.05	96.97	88.89
Dual.	70.13	71.05	70.59	70.13	71.05	70.59
Repe.	89.91	93.33	91.59	90.65	92.38	91.51
Anti.	67.31	56.45	61.40	78.43	64.52	70.80
Quote	85.34	84.62	84.98	87.93	87.18	87.55

Table 5: Figure type classification results w/ or w/o contextual information for each figure of speech. First seven figures are *tropes*, next five figures are *schemes*.

ing problem which CRF is specialized in. We also find that FESpan gives the highest precision but low recall value. This may be attributed to the fact that using two classifiers to predict the boundaries of figures imposes further restrictions.

Overall, the performance of figure unit extraction is not satisfactory, since it is somewhat difficult to precisely compartmentalize the smallest clause sequence containing figurative languages. Only 572 out 1757 predictions exactly matches gold label. We observe that the prediction errors can be largely categorized as follows: 1) *Super/Sub prediction*, the predicted clause sequence is a superset/subset of the gold figurative unit; 2) *Overlapping prediction*, the predicted clause sequence overlaps with a part of the gold figurative unit but is neither its superset nor subset; 3) *Non-lapping prediction*, the predicted clause sequence does not overlap with any gold figurative unit, i.e. predicting non-figurative clause groups as figurative. Column 2 in Table 4 presents the numbers of each error category according to prediction results of FESeq. More than half of the wrong predictions do not overlap with any gold figurative unit. Other wrong predictions are mainly supersets or subsets of certain gold figurative unit. Based on these observations, we conclude that the SOTA models are not doing very well in either discerning figurative languages or determining the exact boundary of figurative unit in our corpus, perhaps because the size of our corpus is relatively small and imbalanced.

5.2 Evaluating Figure Type Classification

For figure type classification, we present results of FTCLS and FTCXT in the second block of Table 3. The latter gives an improvement of 0.67 micro F1 point and 3.5 macro F1 point over the former. Compared to figure extraction, the performance of figure type classification exhibits a high score.

Detailed classification results for each figure w/o contextual information is presented in Table 5. From this table, Columns 2-4 give FTCLS' classification results for each figure. It can be seen that classification accuracy is quite imbalanced across all figure types. On the one hand, F1 scores on *schemes* are relatively higher than *tropes* on the

whole. We suppose it is because *schemes* tend to manifest obvious patterns like repetition, which is easier for a model to capture, while *tropes* involve deviation from superficial meaning, which is more challenging to models and even humans. We expect more efforts on modeling semantics to benefit figure type classification. Notably, *Rhetoric Question (Rheq.)* gives the highest F1 score of 95.12 while being a trope. We suppose it is because the *question mark (i.e. "?")* in *Rheq.* units serve as an obvious clue to facilitate classification.

Table 5 reveals that even among trope figures, model performance is quite imbalanced. F1 score on Metahpor achieve 85.61 while on irony and synaesthesia it is 28.57 and 0.00, repectively. We suppose the reason for this performance gap is twofold. Firstly, figures such as irony and synaesthesia are demanding to collect, resulting in their fairly low distribution in ConFiguRe. To effectively train models for such low-frequency figures, it is necessary to incorporate other specialized techniques. Secondly, figures as epitomized by irony, are strongly related to a wider context, without which the identification of such figures becomes insufficient.

Context Benefits Figure Type Classification To inspect the extent that context information benefits figure classification, we include result of FTCXT in Columns 5-7 of Table 5. This serves as an "ablation study" that undergirds the usefulness of contextual information in figure classification. By comparing results of FTCLS and FTCXT, it can be seen that context information consistently improves classification accuracy for most figures. Further, among all figure types, contextual information is especially conducive to Irony, Synaesthesia and Antithesis, respectively boosting F1 score by 15.87, 14.29 and 9.40 point. This observation is consistent with the linguistic assumption that the identification of these figures usually depends on more context.

5.3 Evaluating Figure Recognition

For figure recognition, baseline results can be found in the third block of Table 3. Similar to figure extraction task, we observe that the sequence labeling model with CRF achieves the best result, surpassing the non-CRF version with 1.52 micro F1 score and 3.92 macro F1 score. It also outperforms the pipeline approach combining the best-performing extraction model and classification model, which suffers from error propagation. F1 score of the rule-based method is the lowest compared to pipeline and end2end methods, indicating that figure recognition requires much more efforts beyond recognizing shallow and obvious patterns.

Sequence 1: 从胡乱对付事情这方面来说，你完全可以称得上是个天才（In terms of messing up with affairs, you may well be called a genius）
Sequence 2: 你完全可以称得上是个天才（You may well be called a genius）

Figure 3: Example of two clause sequences marked as figurative unit for *irony*.

Same as figure extraction task, errors of this task incorporates the following types: *Super prediction*, *Sub prediction* and *Overlapping prediction*. Besides, we introduce *Wrong figure type* error, where the model successfully extracts the figurative units but classifies them into wrong figure type. According to the result of E2ECRF, numbers of each error type is presented in Column 3 of Table 4. We observe that a high percentage of precisely identified figure units are correctly classified with the figure type, and only 75 out of 638 predictions fail in classification. Resonating with figure extraction, we conclude that **discerning figurative units is quite challenging for state-of-the-art models.**

5.4 Discussion

Revisiting Figurative Unit From Subsection 5.1, we obtain a rather low performance in recognizing figurative units. We investigate the classification results of the models and find them rather satisfying. We therefore impute the poor performances to the difficulties in delimiting the boundary. Even if we clearly define a figure unit as the smallest clause sequence, such concept is somewhat controversial under certain circumstances. Figure 3 illustrates two instances which are ambiguous in deciding the gold figure unit. For this reason, it is necessary to improve the annotation process and evaluation metrics in the future.

Revisiting Tropes and Schemes Interestingly, our work can also serve as a supporting evidence for the linguistic categories of *tropes* and *schemes*. *Schemes* reflect deviation from the ordinary pattern or arrangement of words, while *tropes* involve deviation from the ordinary and principal signification of a word (Corbett, 1999). A heat map of confusion matrix in figure type classification is presented in Figure 4. It is intuitively suggested that, even misclassified, the predicted label tends

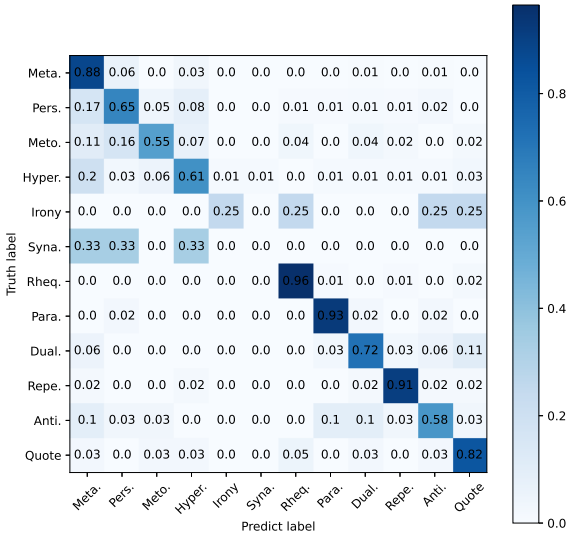

Figure 4: Confusion matrix from FTCLS's results in figure type classification. First seven figures are *tropes*, last five figures are *schemes*.

to fall into the same category as the gold label. For the example of *hyperbole*, there is a 61% probability of being classified correctly, a 20% probability as *metaphor/simile*, and a 6% probability as *metonymy*, all falling into the category of *tropes*. In the meantime, the probability of being classified as *schemes* is merely 7%. In other words, a trope figure is more similar to other trope figures than scheme figures, and vice versa.

6 Conclusions and Future Work

In this paper, we argue that it is necessary to recognize figures from the discourse level. For the first time, we propose the concept of figurative unit as language carrier of one figure and construct a Chinese corpus for Contextualized Figure Recognition (ConFiguRe). ConFiguRe includes 12 most commonly used figures in Chinese, with discourse-level contextual information attached to each figure instance. On ConFiguRe, we design three tasks of figure extraction, figure type classification and figure recognition, and implement state-of-the-art models. A series of experiments show that all three tasks remain challenging and worth exploring.

In future, we hope to increase the size of our corpus, especially adding more figure instances for the types with fewer instances such as irony and synaesthesia. At the same time, we will to improve the model performance with consideration of incorporating contextual information and using more training data.

Acknowledgement

We thank the anonymous reviewers for their helpful comments on this paper. This work was partially supported by National Key Research and Development Program of China (2020AAA0109703), National Natural Science Foundation of China (61876009), and National Social Science Foundation Project of China (21&ZD287).

References

David Bamman and Noah Smith. 2015. Contextualized Sarcasm Detection on Twitter. *Proceedings of the International AAAI Conference on Web and Social Media*, 9(1):574–577. Number: 1.

Luana Bulat, Stephen Clark, and Ekaterina Shutova. 2017. Modelling metaphor with attribute-based semantics. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 2, Short Papers*, pages 523–528, Valencia, Spain. Association for Computational Linguistics.

Gideon O. Burton. 2016. Silva rhetoricae.

Santiago Castro, Devamanyu Hazarika, Ver'şnica P'l'rez-Rosas, Roger Zimmermann, Rada Mihalcea, and Soujanya Poria. 2019. Towards Multimodal Sarcasm Detection (An _obviously_ Perfect Paper). *arXiv:1906.01815 [cs]*. ArXiv: 1906.01815.

Xin Chen, Zhen Hai, Deyu Li, Suge Wang, and Dian Wang. 2021. Jointly Identifying Rhetoric and Implicit Emotions via Multi-Task Learning. In *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*, pages 1429–1434, Online. Association for Computational Linguistics.

Edward Corbett. 1999. *Classical rhetoric for the modern student*. Oxford University Press, New York.

Yiming Cui, Wanxiang Che, Ting Liu, Bing Qin, Shijin Wang, and Guoping Hu. 2020. Revisiting pre-trained models for Chinese natural language processing. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: Findings*, pages 657–668, Online. Association for Computational Linguistics.

Verna Dankers, Karan Malhotra, Gaurav Kudva, Volodymyr Medentsiy, and Ekaterina Shutova. 2020. Being neighbourly: Neural metaphor identification in discourse. In *Proceedings of the Second Workshop on Figurative Language Processing*, pages 227–234, Online. Association for Computational Linguistics.

Dmitry Davidov, Oren Tsur, and Ari Rappoport. 2010. Semi-supervised recognition of sarcasm in Twitter

and Amazon. In *Proceedings of the Fourteenth Conference on Computational Natural Language Learning*, pages 107–116, Uppsala, Sweden. Association for Computational Linguistics.

Ellen K Dodge, Jisup Hong, and Elise Stickles. 2015. Metanet: Deep semantic automatic metaphor analysis. In *Proceedings of the Third Workshop on Metaphor in NLP*, pages 40–49.

Paul Drew and Elizabeth Holt. 1998. Figures of speech: Figurative expressions and the management of topic transition in conversation. *Language in Society*, 27(4):495–522.

Dan Fass. 1991. Met*: A method for discriminating metonymy and metaphor by computer. *Comput. Linguist.*, 17(1):49–90.

Aniruddha Ghosh and Tony Veale. 2017. Magnets for Sarcasm: Making Sarcasm Detection Timely, Contextual and Very Personal. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 482–491, Copenhagen, Denmark. Association for Computational Linguistics.

Debanjan Ghosh, Alexander Richard Fabbri, and Smaranda Muresan. 2017. The Role of Conversation Context for Sarcasm Detection in Online Interactions. *arXiv:1707.06226 [cs]*. ArXiv: 1707.06226.

Debanjan Ghosh, Avijit Vajpayee, and Smaranda Muresan. 2020. A Report on the 2020 Sarcasm Detection Shared Task. In *Proceedings of the Second Workshop on Figurative Language Processing*, pages 1–11, Online. Association for Computational Linguistics.

Pragglejaz Group. 2007. Mip: A method for identifying metaphorically used words in discourse. *Metaphor and Symbol*, 22(1):1–39.

The Pragglejaz Group, Raymond Gibbs, Alice Deignan, Graham Low, Gerard Steen, Lynne Cameron, Elena Semino, Joe Grady, Alan Cienki, and Zoltan Kövecses. 2007. Mip: A method for identifying metaphorically used words in discourse. *Metaphor and Symbol*, 22(1):1–39.

Randy Allen Harris, Chrysanne Di Marco, Sebastian Ruan, and Cliff O'Ă'ŹReilly. 2018. An annotation scheme for Rhetorical Figures. *Argument & Computation*, 9(2):155–175.

Hyeju Jang, Seungwhan Moon, Yohan Jo, and Carolyn Rosé. 2015. Metaphor detection in discourse. In *Proceedings of the 16th Annual Meeting of the Special Interest Group on Discourse and Dialogue*, pages 384–392, Prague, Czech Republic. Association for Computational Linguistics.

Aditya Joshi, Pushpak Bhattacharyya, and Mark J. Carman. 2017. Automatic Sarcasm Detection: A Survey. *ACM Computing Surveys*, 50(5):73:1–73:22.

Aditya Joshi, Vinita Sharma, and Pushpak Bhattacharyya. 2015. Harnessing Context Incongruity for Sarcasm Detection. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 2: Short Papers)*, pages 757–762, Beijing, China. Association for Computational Linguistics.

Aditya Joshi, Vaibhav Tripathi, Pushpak Bhattacharyya, and Mark J. Carman. 2016. Harnessing Sequence Labeling for Sarcasm Detection in Dialogue from TV Series 'Friends'. In *Proceedings of The 20th SIGNLL Conference on Computational Natural Language Learning*, pages 146–155, Berlin, Germany. Association for Computational Linguistics.

Mikhail Khodak, Nikunj Saunshi, and Kiran Vodrahalli. 2017. A large self-annotated corpus for sarcasm. *CoRR*, abs/1704.05579.

Diederik P. Kingma and Jimmy Ba. 2017. Adam: A method for stochastic optimization.

Maximilian Köper and Sabine Schulte im Walde. 2017. Improving verb metaphor detection by propagating abstractness to words, phrases and individual senses. In *Proceedings of the 1st Workshop on Sense, Concept and Entity Representations and their Applications*, pages 24–30, Valencia, Spain. Association for Computational Linguistics.

John D. Lafferty, Andrew McCallum, and Fernando C. N. Pereira. 2001. Conditional random fields: Probabilistic models for segmenting and labeling sequence data. In *Proceedings of the Eighteenth International Conference on Machine Learning*, ICML '01, page 282'Ă'Ş289, San Francisco, CA, USA. Morgan Kaufmann Publishers Inc.

Hankyol Lee, Youngjae Yu, and Gunhee Kim. 2020. Augmenting data for sarcasm detection with unlabeled conversation context. In *Proceedings of the Second Workshop on Figurative Language Processing*, pages 12–17, Online. Association for Computational Linguistics.

Chee Wee (Ben) Leong, Beata Beigman Klebanov, Chris Hamill, Egon Stemle, Rutuja Ubale, and Xianyang Chen. 2020. A Report on the 2020 VUA and TOEFL Metaphor Detection Shared Task. In *Proceedings of the Second Workshop on Figurative Language Processing*, pages 18–29, Online. Association for Computational Linguistics.

Chee Wee (Ben) Leong, Beata Beigman Klebanov, and Ekaterina Shutova. 2018. A Report on the 2018 VUA Metaphor Detection Shared Task. In *Proceedings of the Workshop on Figurative Language Processing*, pages 56–66, New Orleans, Louisiana. Association for Computational Linguistics.

Lizhen Liu, Xiao Hu, Wei Song, Ruiji Fu, Ting Liu, and Guoping Hu. 2018. Neural Multitask Learning for Simile Recognition. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language*

Processing, pages 1543–1553, Brussels, Belgium. Association for Computational Linguistics.

Jesse Mu, Helen Yannakoudakis, and Ekaterina Shutova. 2019. Learning outside the box: Discourse-level features improve metaphor identification. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 596–601, Minneapolis, Minnesota. Association for Computational Linguistics.

Richard M. Roberts and Roger J. Kreuz. 1994. Why do people use figurative language? *Psychological Science*, 5(3):159–163.

Rossano Schifanella, Paloma de Juan, Joel Tetreault, and Liangliang Cao. 2016. Detecting Sarcasm in Multimodal Social Platforms. *Proceedings of the 24th ACM international conference on Multimedia*, pages 1136–1145. ArXiv: 1608.02289.

Steven Shapin. 2012. The ivory tower: the history of a figure of speech and its cultural uses. *The British Journal for the History of Science*, 45(1):1–27.

Gerard Steen. 2010. *A method for linguistic metaphor identification : from MIP to MIPVU*. John Benjamins Pub. Co, Amsterdam Philadelphia.

Chuandong Su, Fumiyo Fukumoto, Xiaoxi Huang, Jiyi Li, Rongbo Wang, and Zhiqun Chen. 2020. DeepMet: A reading comprehension paradigm for token-level metaphor detection. In *Proceedings of the Second Workshop on Figurative Language Processing*, pages 30–39, Online. Association for Computational Linguistics.

Yi-jie Tang and Hsin-Hsi Chen. 2014. Chinese Irony Corpus Construction and Ironic Structure Analysis. In *Proceedings of COLING 2014, the 25th International Conference on Computational Linguistics: Technical Papers*, pages 1269–1278, Dublin, Ireland. Dublin City University and Association for Computational Linguistics.

Yulia Tsvetkov, Leonid Boytsov, Anatole Gershman, Eric Nyberg, and Chris Dyer. 2014. Metaphor detection with cross-lingual model transfer. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 248–258, Baltimore, Maryland. Association for Computational Linguistics.

Byron C. Wallace, Do Kook Choe, Laura Kertz, and Eugene Charniak. 2014. Humans require context to infer ironic intent (so computers probably do, too). In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 512–516, Baltimore, Maryland. Association for Computational Linguistics.

Zhi Wen, Yang Li, Suge Wang, Jian Liao, and Xin Chen. 2019. Feature enhanced cnn for rhetorical questions identification. *Journal of Chinese information processing (in Chinese)*, 33(1):68–76.

Yorick Wilks. 1975. A preferential, pattern-seeking, semantics for natural language inference. *Artificial Intelligence*, 6(1):53 – 74.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, R'l'mi Louf, Morgan Funtowicz, Joe Davison, Sam Shleifer, Patrick von Platen, Clara Ma, Yacine Jernite, Julien Plu, Canwen Xu, Teven Le Scao, Sylvain Gugger, Mariama Drame, Quentin Lhoest, and Alexander M. Rush. 2020. Transformers: State-of-the-art natural language processing. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 38–45, Online. Association for Computational Linguistics.

Gong Zhang. 1963. *Modern Chinese Rhetoric*. Tianjin People's Publishing House.

A Annotation Details

A.1 Figure Definition

We define these 12 figure categorites in reference to silva rhetoricae (`http://rhetoric.byu.edu/`), an authoritative website for rhetoric figures. We also follow a canonical work on rhetorics by Chinese linguistics Gong Zhang (1963). Detailed definition and example for each figure type above is provided in Figure 5.

A.2 Punctuation List

Below is punctuation list we use to separate clauses:
{ ， ， ； ， 。 ， ？ ， ！ ， …… ， —— ， ： }

Figure	Definition	Example
比喻 (Metaphor/ Simile)	A comparison made by referring to one thing as another.	阳光像金子一样珍贵。(Sunshine is as precious as gold.)
比拟 (Personification)	Reference to abstractions or inanimate objects as though they had human qualities or abilities.	我心情愉悦，听见鸟儿在林中歌唱。(I was very happy and could hear the birds singing in the woods.)
借代 (Metonymy)	Reference to something or someone by naming one of its attributes.	据说他们是世界上最优秀的笔杆子。(They are said to be the best pens of the world.)
夸张 (Hyperbole)	Rhetorical exaggeration which is often accomplished via comparisons, similes, and metaphors.	他雄辩滔滔，仿佛能开岩裂石。(His eloquence would split rocks.)
反语(Irony)	Speaking in such a way as to imply the contrary of what one says, often for the purpose of derision, mockery, or jest.	这个勤奋的学生每周读书的时间从不超过一小时。(This diligent student seldom reads more than an hour per week.)
通感 (Synaesthesia)	The production of a sense impression relating to one sense or part of the body by stimulation of another sense or part of the body.	品尝莫扎特音乐的味道。(Taste the music of Mozart.)
问语 (Rhetorical question)	Any question asked for a purpose other than to obtain the information the question asks.	这不是非常明显的例证吗？ (Isn't this a very obvious evidence?)
排比 (Parallelism)	Similarity of structure in a pair or series of related words, phrases, or clauses.	童年是小草的芽儿，充满了生机； 童年是早晨的太阳，充满了活力；童年是清润的 雨水，充满了欢乐。(Childhood is the bud of grass, full of vitality; childhood is the morning sun, full of vitality; childhood is the rain, full of joy.)
对偶 (Duality)	Two similarly structured elements having the same length	不要对谦卑者傲慢，也不要对傲慢者谦卑。(Never be haughty to the humble; never be humble to the haughty.)
反复 (Repetition)	Repeat a word or a sentence.	他现在看起来比以前衰老很多很多了。(He looks much, much older now than before.)
对比 (Antithesis)	Juxtaposition of contrasting words or ideas (often, although not always, in parallel structure).	有的人活着，他已经死了；有的人死了，他还活着。(Some live, when they are already dead; others have died, but are still alive..)
引语 (Quote)	A group of words taken from a text or speech and repeated by someone other than the original author or speaker.	“众人拾柴火焰高”，让我们用我们自己的双手建设我们的家园！。(*The fire burns high when everybody adds wood to it*, let's build our gardon with our own hands.)

Figure 5: Definition and example for each figure type.

Twitter Topic Classification

Dimosthenis Antypas*, **Asahi Ushio***, **Jose Camacho-Collados**
Cardiff NLP, School of Computer Science and Informatics, Cardiff University, United Kingdom
{AntypasD,UshioA,CamachoColladosJ}@cardiff.ac.uk

Leonardo Neves, Vítor Silva, Francesco Barbieri
Snap Inc., Santa Monica, CA, United States
{lneves,vsilvasousa,fbarbieri}@snap.com

Abstract

Social media platforms host discussions about a wide variety of topics that arise everyday. Making sense of all the content and organising it into categories is an arduous task. A common way to deal with this issue is relying on topic modeling, but topics discovered using this technique are difficult to interpret and can differ from corpus to corpus. In this paper, we present a new task based on tweet topic classification and release two associated datasets[1][2]. Given a wide range of topics covering the most important discussion points in social media, we provide training and testing data from recent time periods that can be used to evaluate tweet classification models. Moreover, we perform a quantitative evaluation and analysis of current general- and domain-specific language models on the task, which provide more insights on the challenges and nature of the task.

1 Introduction

Social media platforms, e.g., Twitter, Snapchat, TikTok and Instagram, provide an environment for content creation and information sharing among people. On social platforms, every individual can express their views about current events or anything that they care about, influencing and guiding discussions among their friends and followers. Social media platforms are highly studied to understand behaviors among users, groups, organizations, or even societies (Yang et al., 2021), and in particular to understand opinion of people regarding a variety of topics such as politics (Zhuravskaya et al., 2020), diversity and inclusion (Chakravarthi, 2020), TV shows (Wohn and Na, 2011), sports events (Lim et al., 2015), or finance (Hu et al., 2021). However, one of the biggest challenges in understanding this type of user generated content, is the noise and variety of these texts (Morgan and Van Keulen, 2014; Baldwin et al., 2013). Consequently, identifying topics within social media platforms from their posts is not a trivial task.

Existing solutions can be divided into topic modeling and topic classification. For topic modeling, topics are detected in an unsupervised way with models such as Latent Dirichlet Allocation (LDA) (Blei et al., 2003) and subsequent variations (Steyvers and Griffiths, 2007). Similarly, solutions that use new BERT contextualized embeddings (like BERTopic (Grootendorst, 2022)) have increased in popularity as they offer increased performance. However, these approaches assume that (i) all the topics of interest are represented in the documents included in the study, and (ii) the terms present in these documents are enough to characterize each topic. For these reasons, these methods are usually built as an ad-hoc analysis. Another limitation of these models is interpretability, as it is hard to generalize and label each cluster topic.

On the other hand, topic classification approaches the problem in a supervised manner and assigns multiple topics to each document based on a predefined set of categories. This approach overcomes the issues of interpretability and is not based on assumptions about the vocabulary distribution mentioned above. However, the downside of topic classification is that relies on curated datasets labeled by human annotators, and this can be expensive and time consuming to create.

In this paper, we introduce TweetTopic, a topic classification dataset on Twitter data. To the best of our knowledge, this is the first large-scale topic classification dataset specifically tailored to social media, rather than standard text as news articles (Greene and Cunningham, 2006) or scientific papers (Lazaridou et al., 2021). The dataset consists of a total of 11,267 tweets collected through a time period from September 2019 to August 2021. Each

[1] https://huggingface.co/datasets/cardiffnlp/tweet_topic_single
[2] https://huggingface.co/datasets/cardiffnlp/tweet_topic_multi
*Equal contribution.

Proceedings of the 29th International Conference on Computational Linguistics, pages 3386–3400
October 12–17, 2022.

tweet is assigned one or more topics from a predefined set of categories curated by social platform experts. Aiming to test the robustness of our dataset through time and across topics, we perform several classification experiments, both single-label and multi-label, while utilizing state-of-the-art language models.[3]

2 Related Work

Social media. Social media have become an important aspect of the daily life of millions of people, with 81% of adults in the U.S. stating to have used at least one social platform in 2021 (Auxier and Anderson, 2021) and over 57% of people in EU interacting through social media in 2020 (Eurostat, 2021). In recent years, an increasing number of corporations seem to dedicate a more significant portion of their marketing funds to advertising on social platforms compared to other more traditional mediums (Eid et al., 2020). At the same time, social media has become a political battleground where politicians both debate between them and try to communicate with their voters, (Stier et al., 2018; Llewellyn and Cram, 2016). Finally, social platforms have been used extensively by their users as a means for almost instantaneous news updates both for day-to-day events (Hermida, 2012), and human and natural disasters (e.g., the Ukrainian war or the COVID-19 pandemic) (Khaldarova and Pantti, 2016; Banda et al., 2021).

Therefore, a large volume of content is being generated in social media everyday. Its polymorphism also means that performing any targeted analysis on the data can be a challenging and time-consuming process (Weller, 2015; Stieglitz et al., 2018). Furthermore, even though there are various existing tools focused on analyzing social media data (Batrinca and Treleaven, 2015), there is no established way to efficiently identify and filter only relevant and valuable content (Nugroho et al., 2020).

Topic modeling. Topic models are unsupervised methods to identify relevant topics given a text corpus. LDA (Blei et al., 2003) is one of the most popular algorithms for topic modeling. However, despite being successful in identifying topics in traditional media (Martin and Johnson, 2015; El Akrouchi et al., 2021), LDA often struggles when applied to short, unstructured, and constantly evolving texts, such as Twitter data (Zhao et al., 2011). It also typically underperforms when compared to other supervised methods (Arias et al., 2015). More recently, several variations of LDA have been proposed to address these challenges with social media texts, such as combining author-topic modelling with LDA (Rosen-Zvi et al., 2004; Steinskog et al., 2017), frameworks like Twitter-LDA (Zhao et al., 2011) where noisy words and author information are taken into account, and SKLDA (Tajbakhsh and Bagherzadeh, 2019), where semantic relations between words extracted from WordNet are taken into account.

However, LDA-based methods are often not ideal when we need to assign more than one topic to a document. Even though there are approaches to acquire multiple labels for each topic, they are usually based on hierarchical (Griffiths et al., 2003) or graph (Li and McCallum, 2006) architectures which, depending on the use case, make assumptions about relations of the topics that may not be present in a given corpus (i.e. parent/children topics). Furthermore, semi-supervised or supervised variations of LDA, such as PLDA (Ramage et al., 2011) and sLDA (Mcauliffe and Blei, 2007), have been been used on Twitter data (Resnik et al., 2015; Ashktorab et al., 2014). While such methods have potential for increased performance they usually require prior labelling or information about the documents and thus remove a major advantage they have compared to supervised approaches.

Finally, as a mainly unsupervised technique, evaluating the results of topic modeling can be a hard task. Metrics such as purity, mutual information and pairwise F-measure are used to evaluate the quality of topics/clusters created by the models (Nugroho et al., 2020). On the other hand, qualitative analysis is usually difficult to perform due to the lack of interpretability of topics produced and the difficulty increases with the amount of topic.

In contrast to traditional LDA approaches, techniques such as BERTopic (Grootendorst, 2022) and Top2Vec (Angelov, 2020) attempt to make use of existing knowledge from pretrained language models by extracting embedding representations of tweets and using them to perform topic clustering. Both BERTtopic and Top2Vec tend to be easier to use than LDA, without the need for extensive hyper-parameter tuning, and often result in increased performance (Egger and Yu, 2022).

[3]Tweet classification models associated with TweetTopic have been integrated into TweetNLP (Camacho-Collados et al., 2022).

However, they do have disadvantages, namely: not performing well on small datasets (Abuzayed and Al-Khalifa, 2021), generating a lot of outlier topics (Silveira et al., 2021), and requiring existing knowledge. Finally, these approaches suffer similar drawbacks to LDA regarding evaluation and interpretability.

Topic classification. Given a text as an input, topic classification is the task of associating it with a specific topic (or topics) from a pre-defined set of categories. In what concerns social media, previous work has focused on predicting hashtags as classes (Dhingra et al., 2016). However, the dynamic nature of the events discussed in those platforms makes any dataset focused on hashtags quickly become sparse and outdated. Any new model needs to be trained from scratch since the category set will be different based on the relevance of hashtags. Nevertheless, by focusing on higher-level topics like *Sports* or *Arts & Culture*, widespread and recurrent in social platforms, the data can be leveraged for more extended periods, and any model trained on it can be easily updated with more data as the label set is fixed. It also improves interpretability since there is a clear semantic meaning to the proposed categories, while hashtags might be ambiguous or require additional interpretation.

In terms of previously released data, existing datasets mainly focus on the news articles domain, e.g., BBC News (Greene and Cunningham, 2006), Reuter (Lewis et al., 2004), 20 Newsgroups (Lang, 1995), and WMT News Crawl (Lazaridou et al., 2021) with few exceptions like scientific (arXiv) (Lazaridou et al., 2021) and medical (Ohsumed) (Hersh et al., 1994) domains. Therefore, these datasets offer different sets of challenges with respect to social media.

3 Tweet Topic Classification

This section presents the pipeline to construct TweetTopic, our topic classification dataset based on Twitter data. This pipeline is divided into three steps: (i) tweet collection, (ii) data filtering, and (iii) topic annotation. These steps are explained in more detail in the next subsections.

3.1 Tweet collection

Our goal is to collect a set of tweets with a high coverage of diverse topics over time. We fetched the tweets given specific keywords and time periods using the Twitter API. Since the tweets returned by the API are in reverse chronological order, we decided to split the queries into small time windows to make sure that the tweets are distributed over time. In our case, we queried 50 tweets every two hours from September 2019 to October 2021. As the keywords used to create queries, we collected lists of trending topics from Snapchat[4] in each week during the period (e.g. *pink super moon*, *social distancing*, and *NBA*). This step allowed us to collect tweets with a similar distribution to topics in the real world over time. For this step we also added conditions to exclude retweets, replies, quotes, and tweets with media, as well as specifying the language as English only. In the end, we collected a total of 1,264,037 raw tweets from the API.

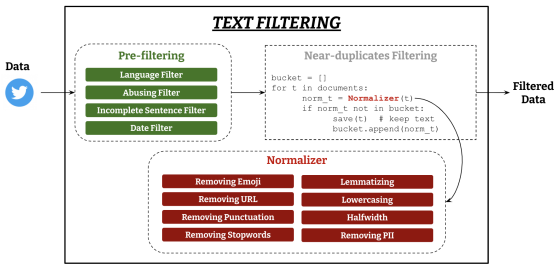

Figure 1: Text filtering pipeline to reduce noise from the tweets and avoid near duplicates.

3.2 Data Filtering

Tweet filtering. Since the raw tweets may contain irrelevant content, we applied several text filtering techniques to get a cleaner tweets corpus. Our text filtering pipeline consists of two steps as described in Figure 1: *pre-filtering* and *near-deduplication*. This filtering fulfilled different goals such as removing abusive content, improving quality and avoiding near-duplicates. In the *pre-filtering*, we first removed non-English tweets by using a fastText based language identifier[5] (Bojanowski et al., 2016). Then, we removed tweets that contained incomplete sentences (e.g., too short or end in the middle of the sentence) or abusing words by using rule-based heuristics. Then, we applied a *near-duplication* filter to drop duplicated tweets. In particular, we first normalized each tweet, and kept unique tweets only in terms of

[4]Available at https://trends.snapchat.com/. We were not able to access Twitter trends since they are not publicly available through APIs.

[5]https://fasttext.cc/blog/2017/10/02/blog-post.html

their normalized form. The normalizer first converted full-width to half-width and removed substrings from the tweet such as emoji, web URLs, punctuation, stopwords, and personally identifiable information (PII).[6] Then, we lemmatized and lowercased each word in the tweets and removed identical tweets after normalization.

Trend filtering. Given our budget and in order to further reduce the number of tweets to annotate while ensuring diversity, we grouped the tweets by the trending topics used to query the raw tweets in each week, and selected the top 15 most common trends within the week.[7] We applied the trending topic filtering for every week which resulted in our final dataset, consisting of 28,573 tweets in total. Note that the trends are different every week, so the tweets are diverse across weeks regarding the trends.[8]

3.3 Annotation

To attain topic annotations over the tweets, we conducted a manual annotation on Amazon Mechanical Turk. We randomly sampled 11,374 tweets from the cleaned tweets and each tweet was annotated by five annotators, collecting 56,870 annotations in total. We manually constructed a topic taxonomy that contained 23 initial topics across diverse genres, asked workers to annotate the relevant (possibly multiple) topics to the tweet.[9] The initial list of 23 topics was shared with us by a research team of Snapchat. This list was selected and curated by a team of social media experts from the company over time to ensure a tailored coverage of social media content.

We ensured several quality control mechanisms within the test, including a qualification test. Each tweet was annotated by five turkers and the final budget for the total estimated annotation cost was $4,000. Each single assignment contained 50 tweets to be annotated where each annotation is completed with an interface that we include in the Appendix. As quality control, each assignment contained three qualification tweets and only those who annotated them correctly were accepted. A small number of raters (10) and their respective tweets were also discarded as they displayed unusual behavior selecting on average more than 5 labels for each tweet where the global average was 1.6 labels per tweet. Also, workers were not allowed to work on the assignment more than once.

Raw	Pre-filter	De-duplication	Trend-filter	Annotated
1,264,037	596,028	202,604	28,573	11,267

Table 1: Number of total tweets after each step.

Post-aggregation. We followed Mohammad et al. (2018) by assigning a label to a tweet provided that the label was suggested by at least two annotators. We opted out of a majority rule as this way our dataset can be used to develop more robust systems that can handle real-world data, which are rarely straightforward and instead can often contain complex linguistic phenomena (Mohammad et al., 2018). Tweets where none of the classes received at least two votes were discarded. The number of tweets in each process is summarized in Table 1.

Inter-annotator agreement. Several metrics can be used to evaluate the quality of an annotation task (Artstein and Poesio, 2008) and it is often difficult to select the most appropriate one. In our experiment, we utilized Krippendorff's alpha (Krippendorff, 2011) with MASI distance (Passonneau, 2006), which is a common combination when dealing with multi-rater and multi-label tasks (Artstein and Poesio, 2008). For our task the alpha statistic results in 0.35. As a comparison reference, a completely random annotation would produce a 0 alpha statistic. When considering the percent agreement of each pair of annotators we acquire a value of 0.87 in contrast to 0.62 for random annotation. These inter-annotator agreement results appear to be inline or slightly better then previous similar multi-label annotation tasks (Mohammad et al., 2018).

3.4 Settings and temporal split

In order to investigate potential temporal differences in the corpus we split the datasets into two periods: (1) from September 2019 to August 2020 (referred to as training data) and (2) from September 2020 to August 2021 (test data). The motivation behind this temporal split is to make the

[6] We detected PII with scrubadub and other components are all based on NLTK.

[7] More details about this process can be found in the Appendix.

[8] In the Appendix we provide a detailed breakdown of the distribution of trends in each week. There, we can confirm that the top trend does not go beyond 20% in most cases, which ensures a diverse set of trends.

[9] The actual instructions shown to workers are included in the Appendix.

Tweet	Topics
Apple Removed More Than 30,000 Apps From The Chinese App Store	- bus & ent - news & soc - sci & tech
#copreps Football: End of the line for FLHS season	sports & games

Table 2: Sample tweets for each setting studied (top: multi-label; botttom: single-label).

task more realistic and evaluate the generalizability performance of the classifiers on future data.

We established two classification settings: (1) multi-label and (2) single-label. Sample instances from both settings are displayed in Table 2.[10] With this distinction, we aim to provide flexibility to users, and increase the usability of the dataset for settings and analyses, where a more fine-grained classification of tweets is not required (i.e. single-label).

Multi-label. By applying a final post-aggregation step to exclude categories that may not be relevant for social media, we removed those categories with fewer than 50 labels overall, leaving a final set of 19 topics.

Single-label. In an effort to keep the classes relatively balanced, we firstly excluded tweets that were labeled with the most dominant of the classes, i.e., news & social concern (32.82% of total tweets), which is highly cross-category. Following this, the remaining ten most prominent classes were considered. Finally, based on logical assumptions regarding the similarity of the classes and also the overlap between them, several labels were grouped together. More specifically: *gaming* and *sports* (35% overlap) were grouped as *sports & gaming*; *music*, *celebrity & pop culture*, and *film tv & video* (44% and 31% overlap) became *pop culture*; *diaries & daily life* and *family* (54% overlap) were grouped together as *daily life*. These three new classes along with the original *arts & culture*, *business & entrepreneurs*, and *science & technology* composed the final set of topics. Finally, in this setting, tweets containing more than one of these six labels were dropped.

3.5 Statistics

The final set of annotated tweets is 11,267 and 6,997 for the multi-label and single-label settings, respectively. Figures 2 and 3 display the percentage of tweets that were classified in each topic, for each time period studied, after the aggregation of annotations for multi-label and single-label settings, respectively.[11] The imbalanced nature that can be observed, e.g., *sports* consisting of 26% of the 2019/20 multi-label dataset while *travel & adventure* only 2%, is explained due to the way tweets were collected, where we aimed to mimic the distribution of real-world data on Twitter.

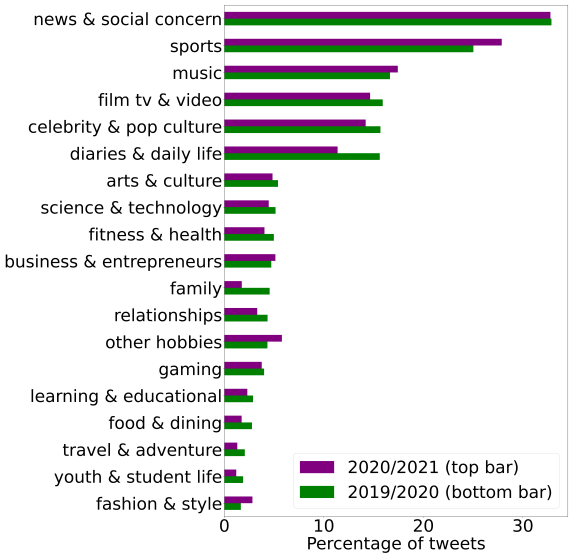

Figure 2: Percentage of tweets that were annotated with a given topic (multi-label setting) for each time period.

Number of labels. When considering the multi-label setting, 50% of the tweets are classified with only one label while only 2.7% are given four or more labels, with the maximum amount being six. However, the dataset is diverse enough with 35% and 12% of the tweets having two and three labels respectively. This coder behavior (i.e. preferring to select only one class) can be observed on similar multi-label annotation tasks (Véronis, 1998; Poesio and Artstein, 2005).

Class distribution across time periods. We note that the distribution of classes between the two time periods studied remains largely similar in both settings with the largest difference being in the *music* and *news & social concern* classes being 3.5% more populous in 2019/20. This observation suggests that our curated topics are broad enough to be relatively robust to temporal trends.

[10] For readability, tweet examples have been slightly modified within the paper, removing links and usernames which are anonymized in the dataset.

[11] For the multi-label setting the percentages sum up to more than 100% due to the nature of the annotation.

Class	length	punc	upp/low	#	@	emojis	mtld	count
arts & culture	166.9 ±67.5	6.5 ±3.4	0.2 ±0.6	0.8 ±1.4	0.4 ±0.5	0.1 ±0.3	140.9	577
business & entrepreneurs	186.3 ±65.5	6.4 ±3.1	0.1 ±0.2	0.6 ±1.1	0.5 ±0.5	0.0 ±0.2	159.0	554
celebrity & pop culture	155.5 ±67.8	7.4 ±3.7	0.2 ±0.9	0.6 ±1.0	0.8 ±0.7	0.1 ±0.4	145.8	1685
diaries & daily life	168.3 ±68.4	5.4 ±3.3	0.1 ±0.7	0.4 ±0.9	0.4 ±0.5	0.1 ±0.5	132.5	1525
family	165.1 ±68.5	5.2 ±3.2	0.2 ±1.4	0.5 ±1.0	0.4 ±0.5	0.2 ±0.5	112.7	358
fashion & style	147.9 ±55.4	7.8 ±3.1	0.2 ±0.5	1.0 ±1.5	0.6 ±0.5	0.1 ±0.3	98.8	251
film tv & video	157.7 ±66.3	7.5 ±3.7	0.2 ±0.8	0.6 ±1.1	0.7 ±0.6	0.1 ±0.4	145.1	1723
fitness & health	195.4 ±67.1	6.3 ±2.8	0.1 ±0.1	0.5 ±0.9	0.6 ±0.5	0.1 ±0.3	168.5	508
food & dining	165.2 ±64.5	6.1 ±3.1	0.1 ±0.2	0.5 ±1.0	0.4 ±0.5	0.1 ±0.4	154.7	255
gaming	159.6 ±68.9	6.5 ±3.9	0.1 ±0.2	0.5 ±1.0	0.5 ±0.6	0.0 ±0.2	128.4	437
learning & educational	191.8 ±65.8	5.9 ±2.9	0.1 ±0.1	0.6 ±1.0	0.5 ±0.6	0.0 ±0.2	156.7	293
music	143.5 ±64.0	8.4 ±4.4	0.3 ±1.1	0.7 ±1.1	0.8 ±0.7	0.1 ±0.5	119.8	1919
news & social concern	183.1 ±70.5	6.6 ±3.0	0.2 ±1.3	0.4 ±0.8	0.6 ±0.6	0.0 ±0.2	165.1	3698
other hobbies	160.9 ±69.2	6.3 ±3.4	0.2 ±0.7	0.6 ±1.0	0.4 ±0.6	0.1 ±0.4	143.6	568
relationships	162.4 ±70.6	5.3 ±3.5	0.2 ±1.6	0.4 ±0.9	0.5 ±0.6	0.2 ±0.9	111.9	432
science & technology	177.9 ±69.4	6.7 ±2.8	0.1 ±0.5	0.5 ±1.0	0.6 ±0.5	0.0 ±0.1	164.2	542
sports	162.8 ±65.9	6.4 ±3.2	0.2 ±1.4	0.5 ±0.8	0.7 ±0.6	0.1 ±0.3	152.8	2977
travel & adventure	175.2 ±72.3	6.2 ±3.1	0.2 ±1.8	0.5 ±1.0	0.5 ±0.5	0.1 ±0.2	173.1	190
youth & student life	202.0 ±62.4	5.9 ±3.2	0.1 ±0.1	0.5 ±0.9	0.5 ±0.6	0.1 ±0.2	155.6	174

Table 3: General lexical statistics for each class. The averages of the length of tweet, punctuation count, upper/lower case ratio (upl/low), hashtags count, mentions count, emojis count are reported along with their standard deviation. Frequency metrics are normalized based on the text length. The last two columns correspond to the lexical diversity (mtld) and total number of tweets.

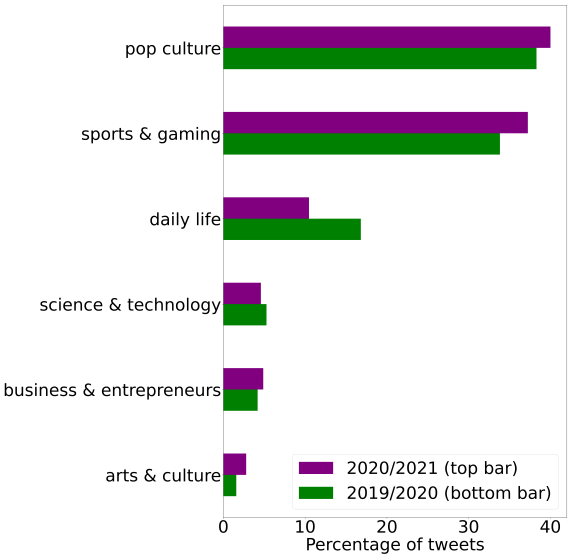

Figure 3: Percentage of tweets that were annotated with a given topic (single-label setting) for each time period.

Topic features. In order to get a better understanding of the data, and to investigate potential significant characteristics, we extract various statistics from the tweets in the multi-label dataset. Table 3 displays the average values of tweet length, number of punctuation symbols, upper to lower case ratio, number of hashtags, number of mentions and number of emojis, along with their standard deviations for each topic. In order to have a fair comparison, all the metrics are normalized based on the tweet length ($(metric/length) * 100$). The Measure of Textual Lexical Diversity (MTLD) (McCarthy and Jarvis, 2010) is also reported as an indication on the vocabulary richness of each class, as well as the number of tweets for each class. The topics *celebrity & pop culture* and *music* have the highest occurrences of mentions "@" (0.8). This is intuitively due to the fact that a large number of tweets belonging to these classes will mention recognizable users such as artists or athletes. Similarly, tweets belonging to the *fashion & style* topic tend to include more hashtags (#) on average (1 hashtag per tweet), which can be attributed to the nature of hashtags in Twitter, usually employed to indicate popular and trending topics. Finally, topics that can be considered more accessible to the general public such as *fashion & style*, *family*, and *relationships* achieve a relatively low lexical diversity score (98.8, 112.7, 111.9) while more specialized or advanced topics such as *travel & adventure*, *business & entrepreneurs* and *fitness & health* display higher lexical diversity (173.1, 159.0, 168.5).

4 Evaluation

In this section, we present our experimental results.

4.1 Experimental setting

Datasets. We perform experiments in our tweet classification annotated datasets. In particular, our experiments are based on two settings, single-label and multi-label (see Section 3.4 for details).

Comparison systems. To evaluate our dataset, we first use simple baselines: Majority (most frequent class in training) and Random (uniform probability for each class). As comparison systems, we train a traditional bag of words with SVM and a fastText classifier (Bojanowski et al., 2016) that utilizes pretrained embeddings (Mikolov et al., 2018). Furthermore, BERT base and large (Devlin et al., 2018) and both base and large versions of RoBERTa (Liu et al., 2019) are used as comparison systems. As classifiers specialized on social media, i.e. trained on Twitter data, BERTweet (Nguyen et al., 2020), TimeLM-19, and TimeLM-21 (Loureiro et al., 2022), all based on a RoBERTa architecture, are also utilized. BERTweet is trained on a corpus of 845M tweets mainly from 01/2012 to 08/2019, while also including 5M COVID-19 related tweets from 01/2020 to 03/2020. On the other hand, TimeLM-19 is trained on 95M tweets gathered between 2018 and 2019. For completeness, we also report results of TimeLM-21, trained on 125M tweets from 2018 to 2021, but excluded it from our main analysis given the time overlap with the test set (reminder that one of the motivations of this task is to be able to process tweets in real time). TimeLMs models use the RoBERTa-base model as initial checkpoint, while BERTweet is trained from scratch. The implementations provided by Hugging Face (Wolf et al., 2019) are used to train and test all language models.[12]

Evaluation metrics. For both settings macro average Precision, Recall and F1, as well as Accuracy, are used to evaluate the models tested. As an alternative metric for the multi-label setting, Jaccard Index (JI) is also utilized, as it can offer useful insights about the models performances (Pereira et al., 2018; Tsoumakas et al., 2009). More specifically, the index is calculated for each tweet individually and the final metric is computed as the average over all entries.

4.2 Results

Table 4 displays the results of all comparison system on both settings. While only a number of models were tested, the results suggest that domain-specific knowledge appears to be more important than the size of the model, with Twitter base models outperforming large generic language models. Given the larger number of labels and more challenging setting, multi-label classification appears to be most challenging setting with the best model TimeLM-21, barely achieving 58.8% F1 and 67.6% Jaccard scores, in comparison to 70.1% F1 and 86.8% Accuracy in the single-label setting. However, it is important to note that TimeLM-21 has the unfair advantage of being trained with a more recent corpus and more specifically a corpus from the same time period as the test set. Taking this into consideration, the next best performing model is TimeLM-19 with 57.2% and 70% F1 scores, for the multi-label and single-label settings respectively. Even though the differences in the average F1 scores between the two models is relatively small, 1.6% and 0.1% for multi/single settings, when taking into account their performance in each individually topic, we can identify topics where TimeLM-21 clearly outperforms TimeLM-19 (see Section 5.1 for more details).

	Model	Multi-label					Single-label			
		Pr	Rec	F1	Acc	JI	Pr	Rec	F1	Acc
Baselines	Random	8.4	48.3	12.6	0	7.9	15	14.2	11.9	15.5
	Majority	1.5	5.3	2.3	18.0	22.6	6.7	16.7	9.5	40
	SVM	69.4	23.7	30.5	37.1	51.8	73.6	47.4	50.2	75.8
	fastText	67.0	18.0	24.0	31.9	43.5	56.0	46.0	48.0	74.0
Language models	BERT-base	69.7	42.5	50.1	45.5	63.9	62.4	60.0	58.8	81
	BERT-large	64.4	**51.5**	56.4	44.6	65.1	62.4	61.7	61.7	84.3
	RB-base	68.5	49.2	55.8	46.5	66.2	64.8	66.7	65.6	85.9
	RB-large	**72.2**	48.9	56.3	**47.9**	**67.7**	66.1	56.2	58.3	84.5
	BERTweet	66.9	46.1	52.7	47.1	66.9	64.9	65.6	63.8	85.2
	TimeLM-19	71.1	50.4	**57.2**	47.7	67.5	**76.5**	**68.9**	**70.0**	**86.4**
	TimeLM-21	*66.1*	*54.2*	*58.8*	*47.1*	*67.6*	*73.9*	*69.8*	*70.1*	*86.8*

Table 4: Macro average Precision (Pr), Recall (Rec), F1, and accuracy results in TweetTopic (temporal split). Jaccard Index (JI) is reported for the multi-label setting.

5 Analysis

In this section, we analyse two important aspects of the TweetTopic dataset, mainly its temporal dimension (Section 5.1) and the errors made by the systems (Section 5.2).

5.1 Temporal analysis

The strong performance of TimeLM-21 provided evidence regarding the importance of an up-to-date training corpus. We continue our investigation by training the same set of models on a random split

[12] More details about the exact hyperparameters are included in the Appendix.

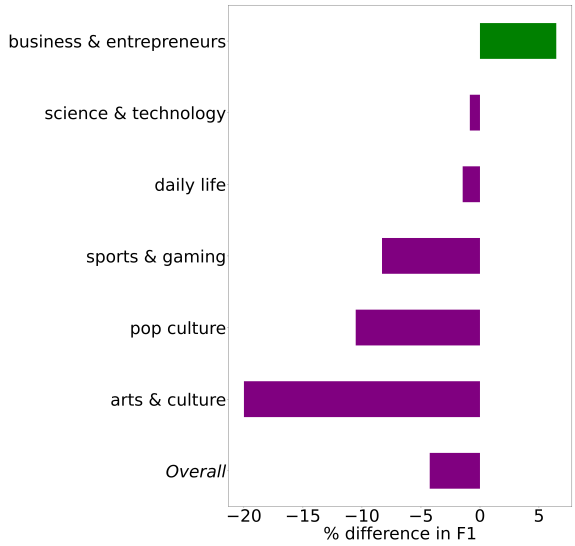

Figure 4: Relative (%) differences in F1 scores when *TimeLM-19* is trained in a temporal and in a random setting for the single-label setting. Negative values indicate that when using the temporal split the model's performance decreases.

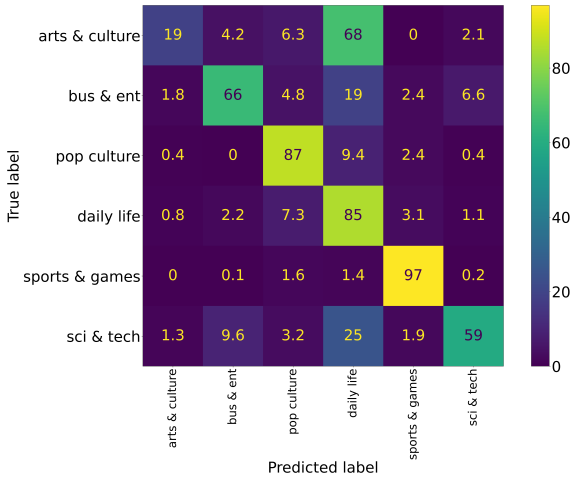

Figure 5: Confusion matrix of the TimeLM-19 results for the single-label setting. The values displayed are normalized by row.

of the data (i.e., both training and test sets with tweets from 2019 to 2021). To make the results comparable, we created training and test sizes of the same size as the original temporal split.[13]

Table 5 displays the F1 scores, while using a multi-label setting for each class in both the temporal and random splits. Every model tested performs better when trained using information from both time periods, i.e using random split. Taking into account that in both splits the distribution of classes is similar (Figure 2), we can assume that the temporal differences in the data provide useful information. It is worth noting that the "specialized" Twitter models display a more robust performance regarding the training data used. In particular, there are 8, 9 and 4 topics where BERTweet, TimeLM-19, and TimeLM-21 respectively perform better while using the temporal split in contrast to 3 and 1 of RoBERTa base and large respectively (models that have a similar architecture).

We continue our analysis by investigating in more detail TimeLM-19's results, which is the best performing model according to the evaluation (Section 4). Figure 4 displays the TimeLM-19 performance differences between the temporal and random splits on the single-label setting. In general, results are overall better in the random split, with an overall relative decrease of 4.3% in Macro-F1 for the temporal split. The largest decrease in performance is observed for the *arts & culture* topic in both settings, which can be attributed to a fast evolving vocabulary. In contrast, *business & entrepreneurs* does not see any decreased in performance in both settings, and results are even slightly better on the temporal split.[14]

5.2 Error analysis

To better understand the nature of errors made by language models, Figure 5 shows a confusion matrix for the best-performing TimeLM-19 model in the single-label setting. The model seems to struggle with tweets assigned to the *arts & culture* topic with 68% of them being misclassified as *daily life*. These errors include entries such as "Happy Day of the Dead 2020! #GoogleDoodle" or "Gifts of love are the ingredients of a #MerryChristmas Give your loved ones a physical/virtual crypto gift card within the {{USERNAME}} app". While these tweets revolve around religious/cultural holidays, one might also associate them to daily life events, which also shows the challenging nature of this dataset. Another topic that is frequent misclassified is *science & technology*, with 41% of the tweets being assigned to the wrong topic. When looking at the errors we identify tweets such as "Bill Gates-Funded Company Releases Genetically Modified Mosquitoes in US", classified as *business & entrepreneurs*, and "Monday's Google Doodle

[13] While the distribution of labels may naturally be altered, this change is minimal, as we can recall from Figures 2 and 3.

[14] In the Appendix we provide a detailed analysis by quarter, in order to better understand the temporal aspect. The results confirm how the performance of *arts & culture* decreases over time, while for the rest of the topics the trend is unclear.

Class	Random		SVM		BERT		RB		RB-large		BERTweet		TimeLM-19		TimeLM-21	
	temp	rand	temp	rand	temp	rand	temp	rand	temp	rand	temp	rand	temp	rand	temp	rand
arts & culture	**8.6**	8.3	3.6	**27.7**	17.8	**35.9**	20.9	**41.2**	28.0	**44.0**	9.8	**28.2**	21.3	**39.1**	35.4	**44.8**
business & entrepreneurs	**8.7**	7.4	18.0	**28.7**	**53.3**	49.2	56.7	**57.1**	50.9	56.5	**59.7**	54.5	**58.6**	55.3	**56.3**	54.0
celebrity & pop culture	22.8	22.9	22.3	**41.7**	34.4	**54.3**	47.2	**52.7**	50.5	**59.6**	43.7	**54.9**	**48.6**	47.8	46.4	**57.6**
diaries & daily life	18.2	**21.2**	25.8	**34.4**	**45.2**	44.0	46.2	**50.3**	43.5	**49.3**	44.6	**49.9**	44.5	**51.2**	44.7	**49.8**
family	3.5	**6.3**	33.9	**46.4**	47.2	**48.3**	50.6	**56.8**	52.8	**63.4**	46.1	**49.1**	46.4	**55.2**	53.1	**56.2**
fashion & style	**4.8**	4.1	38.4	**57.6**	52.8	**74.8**	66.4	**74.1**	66.4	**77.4**	56.0	**68.8**	**66.4**	75.2	67.2	**75.2**
film tv & video	**22.8**	22.0	47.3	**58.6**	62.8	**68.2**	64.4	**71.4**	64.7	**71.3**	66.8	**69.2**	66.1	**72.2**	65.4	**70.6**
fitness & health	6.6	**9.3**	35.7	**36.0**	**53.6**	52.2	52.4	**53.2**	62.4	**65.4**	**48.2**	38.7	**55.7**	42.2	58.6	52.6
food & dining	3.5	**4.6**	25.0	**41.7**	**70.1**	68.2	**75.1**	75.3	**79.3**	68.2	**74.5**	65.7	**75.4**	70.7	**80.4**	71.6
gaming	6.9	**7.5**	31.8	**45.0**	57.4	**61.2**	58.4	**61.4**	63.8	**69.1**	66.1	**67.6**	64.6	**69.2**	64.8	71.2
learning & educational	4.2	**4.5**	13.0	**13.9**	38.2	**43.2**	**49.5**	48.7	49.8	45.8	**42.9**	36.2	**49.3**	47.1	**48.9**	47.0
music	24.7	**25.5**	76.1	**81.8**	83.6	**86.0**	86.0	**87.1**	87.4	**88.1**	86.9	**87.2**	**88.1**	87.8	86.9	**88.2**
news & social concern	39.3	**39.9**	69.8	**76.9**	**83.8**	**83.8**	83.9	**84.6**	85.5	**85.9**	83.5	**84.3**	84.4	**86.2**	84.5	**85.0**
other hobbies	**10.5**	9.6	4.2	**15.0**	**27.0**	23.6	25.0	**28.4**	31.7	**35.4**	**23.1**	21.5	27.7	**30.3**	31.1	26.2
relationships	6.4	**7.3**	13.7	**36.3**	30.8	**35.2**	37.6	**51.8**	39.3	**56.8**	36.8	**51.2**	35.3	**51.6**	44.5	**54.0**
samples avg	13.8	**14.3**	57.0	**63.7**	**70.3**	72.0	73.1	**74.2**	74.4	**76.4**	**73.8**	73.2	74.3	**75.2**	74.7	**75.2**
science & technology	8.3	**9.3**	17.4	**35.8**	45.9	**50.3**	54.2	**56.4**	52.1	**59.4**	46.9	**53.2**	50.5	**56.0**	50.2	**52.1**
sports	**36.6**	34.8	82.2	**89.1**	93.1	**93.2**	**94.8**	94.2	94.6	**95.4**	**95.4**	94.4	**95.6**	94.8	**95.2**	94.8
travel & adventure	2.2	**3.5**	**17.7**	9.8	**21.7**	20.6	41.5	**47.7**	46.3	**59.9**	**38.5**	0.0	**57.1**	56.0	52.2	54.7
youth & student life	1.7	**2.9**	2.9	**12.4**	33.3	**44.6**	49.2	**52.4**	21.0	**46.0**	31.6	**35.2**	**50.4**	43.6	50.8	**51.0**
macro avg	12.6	**13.2**	30.5	**41.5**	50.1	**54.6**	55.8	**60.3**	56.3	**63.0**	52.7	**53.1**	57.2	**59.6**	58.8	**60.9**

Table 5: Macro average F1 scores for the multi-label setting when using temporal (temp) and random (rand) split. Highlighted with bold is the best score for each model.

Celebrates Jupiter And Saturn On The Winter Solstice via Forbes", classified as *daily life*. In other cases, further investigation would be required to understand the source of the mistakes, e.g., "A year ago we looked at PE10s across the world on URL The latest Weekly Macro Themes takes a look at how the Euro Area stacks up now." was classified as *sports* instead *business & entrepreneurs*. The nature of these types of error, as well as the relatively low performance of models compares to other topic classification datasets, suggest that there is ample room for improvement.

When considering the multi-label setting, there are topics with high percentage of errors such as *celebrity & pop culture* and *diaries & daily life*. There are entries like "Anyone else notice {O Shea Jack Nichol son} hasn't tweeted about the Lakers making the conference finals? Weird. You good man?" where the model correctly classifies it as *sports* but fails to classify it as *celebrity & pop culture*, being probably unaware of the celebrity status of the person being mentioned. The *diaries & daily life* topic seems to be particular confusing for the model and fails to identify it in tweets such as "Lost all my bets on the Kentucky Derby today but scored a tee time at {{USERNAME}} Black course next weekend I'd say I came out a winner.", and "Faceing difficulty while login to internet banking for the 1st time using Id and password provided in the welcome kit didn t expected this from such a good bank {Canara Bank}", even though they are correctly assigned the *sports* and *business & entrepreneurs* topics, respectively.

6 Conclusions & Future Work

In this paper we presented TweetTopic, the first large-scale dataset for tweet topic classification. Given the prominence of social media in recent times, this dataset can help build supervised models for clustering and organising the online content. The curated set of topics contains a diverse and broad set of categories that cover most topics present in social platform data. This dataset can further motivate research on the evolution of these initial topics on social platforms, i.e., the extension of the existing categorization to new topics or subtopics that will emerge and fade over time due to user engagement. Moreover, TweetTopic has been shown to be relatively resilient to temporal changes, and it offers easily interpretable results. Based on these contributions, we believe that this dataset will be useful for a significant number of researchers and practitioners working on social media, including Computational Social Science and Data Mining experts, given the relevance of the topic for extracting information and understanding online behavior.

Finally, while this first iteration of TweetTopic focuses on English, our aim is to apply the same methodology to other languages, for which our guidelines and process to construct the dataset described in Section 3 can serve as the main basis.

Acknowledgements

Jose Camacho-Collados is supported by a UKRI Future Leaders Fellowship.

References

Abeer Abuzayed and Hend Al-Khalifa. 2021. Bert for arabic topic modeling: An experimental study on bertopic technique. *Procedia Computer Science*, 189:191–194.

Dimo Angelov. 2020. Top2vec: Distributed representations of topics.

Andrés Arias, Luis H Martínez, Ricardo A Hincapie, Mauricio Granada, et al. 2015. An ieee xplore database literature review regarding the interaction between electric vehicles and power grids. *2015 IEEE PES Innovative Smart Grid Technologies Latin America (ISGT LATAM)*, pages 673–678.

Ron Artstein and Massimo Poesio. 2008. Inter-coder agreement for computational linguistics. *Computational linguistics*, 34(4):555–596.

Zahra Ashktorab, Christopher Brown, Manojit Nandi, and Aron Culotta. 2014. Tweedr: Mining twitter to inform disaster response. In *ISCRAM*, pages 269–272. Citeseer.

Brooke Auxier and Monica Anderson. 2021. Social media use in 2021. *Pew Research Center*, 1:1–4.

Timothy Baldwin, Paul Cook, Marco Lui, Andrew MacKinlay, and Li Wang. 2013. How noisy social media text, how diffrnt social media sources? In *Proceedings of the Sixth International Joint Conference on Natural Language Processing*, pages 356–364.

Juan M Banda, Ramya Tekumalla, Guanyu Wang, Jingyuan Yu, Tuo Liu, Yuning Ding, Ekaterina Artemova, Elena Tutubalina, and Gerardo Chowell. 2021. A large-scale covid-19 twitter chatter dataset for open scientific research—an international collaboration. *Epidemiologia*, 2(3):315–324.

Bogdan Batrinca and Philip C Treleaven. 2015. Social media analytics: a survey of techniques, tools and platforms. *Ai & Society*, 30(1):89–116.

David M. Blei, Andrew Y. Ng, and Michael I. Jordan. 2003. Latent dirichlet allocation. *J. Mach. Learn. Res.*, 3(null):993–1022.

Piotr Bojanowski, Edouard Grave, Armand Joulin, and Tomas Mikolov. 2016. Enriching word vectors with subword information. *arXiv preprint arXiv:1607.04606*.

Jose Camacho-Collados, Kiamehr Rezaee, Talayeh Riahi, Asahi Ushio, Daniel Loureiro, Dimosthenis Antypas, Joanne Boisson, Luis Espinosa-Anke, Fangyu Liu, Eugenio Martínez-Cámara, et al. 2022. Tweetnlp: Cutting-edge natural language processing for social media. *arXiv preprint arXiv:2206.14774*.

Bharathi Raja Chakravarthi. 2020. Hopeedi: A multilingual hope speech detection dataset for equality, diversity, and inclusion. In *Proceedings of the Third Workshop on Computational Modeling of People's Opinions, Personality, and Emotion's in Social Media*, pages 41–53. Association for Computational Linguistics.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. BERT: pre-training of deep bidirectional transformers for language understanding. *CoRR*, abs/1810.04805.

Bhuwan Dhingra, Zhong Zhou, Dylan Fitzpatrick, Michael Muehl, and William Cohen. 2016. Tweet2vec: Character-based distributed representations for social media. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 269–274, Berlin, Germany. Association for Computational Linguistics.

R Egger and J Yu. 2022. A topic modeling comparison between lda, nmf, top2vec, and bertopic to demystify twitter posts. *Front. Sociol. 7: 886498. doi: 10.3389/fsoc*.

M Eid, N Nusairat, Mahmud Alkailani, and H Al-Ghadeer. 2020. Internet users' attitudes towards social media advertisements: The role of advertisement design and users' motives. *Management Science Letters*, 10(10):2361–2370.

Manal El Akrouchi, Houda Benbrahim, and Ismail Kassou. 2021. End-to-end lda-based automatic weak signal detection in web news. *Knowledge-Based Systems*, 212:106650.

Eurostat. 2021. Do you participate in social networks?

Derek Greene and Pádraig Cunningham. 2006. Practical solutions to the problem of diagonal dominance in kernel document clustering. In *Proceedings of the 23rd international conference on Machine learning*, pages 377–384.

Thomas Griffiths, Michael Jordan, Joshua Tenenbaum, and David Blei. 2003. Hierarchical topic models and the nested chinese restaurant process. *Advances in neural information processing systems*, 16.

Maarten Grootendorst. 2022. Bertopic: Neural topic modeling with a class-based tf-idf procedure.

Alfred Hermida. 2012. Social journalism: Exploring how social media is shaping journalism. *The handbook of global online journalism*, 12:309–328.

William Hersh, Chris Buckley, TJ Leone, and David Hickam. 1994. Ohsumed: An interactive retrieval evaluation and new large test collection for research. In *SIGIR'94*, pages 192–201. Springer.

Danqi Hu, Charles M Jones, Valerie Zhang, and Xiaoyan Zhang. 2021. The rise of reddit: How social media affects retail investors and short-sellers' roles in price discovery. *Available at SSRN 3807655*.

Irina Khaldarova and Mervi Pantti. 2016. Fake news: The narrative battle over the ukrainian conflict. *Journalism practice*, 10(7):891–901.

Klaus Krippendorff. 2011. Computing krippendorff's alpha-reliability.

Ken Lang. 1995. Newsweeder: Learning to filter netnews. In *Machine Learning Proceedings 1995*, pages 331–339. Elsevier.

Angeliki Lazaridou, Adhi Kuncoro, Elena Gribovskaya, Devang Agrawal, Adam Liska, Tayfun Terzi, Mai Gimenez, Cyprien de Masson d'Autume, Tomas Kocisky, Sebastian Ruder, et al. 2021. Mind the gap: Assessing temporal generalization in neural language models. *Advances in Neural Information Processing Systems*, 34.

David D Lewis, Yiming Yang, Tony Russell-Rose, and Fan Li. 2004. Rcv1: A new benchmark collection for text categorization research. *Journal of machine learning research*, 5(Apr):361–397.

Wei Li and Andrew McCallum. 2006. Pachinko allocation: Dag-structured mixture models of topic correlations. In *Proceedings of the 23rd international conference on Machine learning*, pages 577–584.

Joon Soo Lim, YoungChan Hwang, Seyun Kim, and Frank A Biocca. 2015. How social media engagement leads to sports channel loyalty: Mediating roles of social presence and channel commitment. *Computers in Human Behavior*, 46:158–167.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Roberta: A robustly optimized BERT pretraining approach. *CoRR*, abs/1907.11692.

Clare Llewellyn and Laura Cram. 2016. Brexit? analyzing opinion on the uk-eu referendum within twitter. In *Tenth International AAAI Conference on Web and Social Media*.

Ilya Loshchilov and Frank Hutter. 2017. Decoupled weight decay regularization. *arXiv preprint arXiv:1711.05101*.

Daniel Loureiro, Francesco Barbieri, Leonardo Neves, Luis Espinosa Anke, and Jose Camacho-Collados. 2022. Timelms: Diachronic language models from twitter. *arXiv preprint arXiv:2202.03829*.

Fiona Martin and Mark Johnson. 2015. More efficient topic modelling through a noun only approach. In *Proceedings of the Australasian Language Technology Association Workshop 2015*, pages 111–115.

Jon Mcauliffe and David Blei. 2007. Supervised topic models. *Advances in neural information processing systems*, 20.

Philip M McCarthy and Scott Jarvis. 2010. Mtld, vocd-d, and hd-d: A validation study of sophisticated approaches to lexical diversity assessment. *Behavior research methods*, 42(2):381–392.

Tomas Mikolov, Edouard Grave, Piotr Bojanowski, Christian Puhrsch, and Armand Joulin. 2018. Advances in pre-training distributed word representations. In *Proceedings of the International Conference on Language Resources and Evaluation (LREC 2018)*.

Saif Mohammad, Felipe Bravo-Marquez, Mohammad Salameh, and Svetlana Kiritchenko. 2018. Semeval-2018 task 1: Affect in tweets. In *Proceedings of the 12th international workshop on semantic evaluation*, pages 1–17.

Mena Badieh Habib Morgan and Maurice Van Keulen. 2014. Information extraction for social media. In *Proceedings of the Third Workshop on Semantic Web and Information Extraction*, pages 9–16.

Dat Quoc Nguyen, Thanh Vu, and Anh Tuan Nguyen. 2020. BERTweet: A pre-trained language model for English Tweets. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 9–14.

Robertus Nugroho, Cecile Paris, Surya Nepal, Jian Yang, and Weiliang Zhao. 2020. A survey of recent methods on deriving topics from twitter: algorithm to evaluation. *Knowledge and information systems*, 62(7):2485–2519.

Rebecca Passonneau. 2006. Measuring agreement on set-valued items (masi) for semantic and pragmatic annotation.

Rafael B Pereira, Alexandre Plastino, Bianca Zadrozny, and Luiz HC Merschmann. 2018. Correlation analysis of performance measures for multi-label classification. *Information Processing & Management*, 54(3):359–369.

Massimo Poesio and Ron Artstein. 2005. The reliability of anaphoric annotation, reconsidered: Taking ambiguity into account. In *Proceedings of the workshop on frontiers in corpus annotations ii: Pie in the sky*, pages 76–83.

Daniel Ramage, Christopher D Manning, and Susan Dumais. 2011. Partially labeled topic models for interpretable text mining. In *Proceedings of the 17th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 457–465.

Philip Resnik, William Armstrong, Leonardo Claudino, Thang Nguyen, Viet-An Nguyen, and Jordan Boyd-Graber. 2015. Beyond lda: exploring supervised topic modeling for depression-related language in twitter. In *Proceedings of the 2nd workshop on computational linguistics and clinical psychology: from linguistic signal to clinical reality*, pages 99–107.

Michal Rosen-Zvi, Thomas Griffiths, Mark Steyvers, and Padhraic Smyth. 2004. The author-topic model for authors and documents. In *Proceedings of the 20th conference on Uncertainty in artificial intelligence*, pages 487–494.

Raquel Silveira, CG Fernandes, João A Monteiro Neto, Vasco Furtado, and José Ernesto Pimentel Filho. 2021. Topic modelling of legal documents via legal-bert. *Proceedings http://ceur-ws org ISSN*, 1613:0073.

Asbjørn Steinskog, Jonas Therkelsen, and Björn Gambäck. 2017. Twitter topic modeling by tweet aggregation. In *Proceedings of the 21st Nordic Conference on Computational Linguistics*, pages 77–86, Gothenburg, Sweden. Association for Computational Linguistics.

Mark Steyvers and Tom Griffiths. 2007. Probabilistic topic models. *Handbook of latent semantic analysis*, 427(7):424–440.

Stefan Stieglitz, Milad Mirbabaie, Björn Ross, and Christoph Neuberger. 2018. Social media analytics–challenges in topic discovery, data collection, and data preparation. *International journal of information management*, 39:156–168.

Sebastian Stier, Arnim Bleier, Haiko Lietz, and Markus Strohmaier. 2018. Election campaigning on social media: Politicians, audiences, and the mediation of political communication on facebook and twitter. *Political communication*.

Mir Saman Tajbakhsh and Jamshid Bagherzadeh. 2019. Semantic knowledge lda with topic vector for recommending hashtags: Twitter use case. *Intelligent Data Analysis*, 23(3):609–622.

Grigorios Tsoumakas, Ioannis Katakis, and Ioannis Vlahavas. 2009. Mining multi-label data. *Data mining and knowledge discovery handbook*, pages 667–685.

Jean Véronis. 1998. A study of polysemy judgements and inter-annotator agreement. In *Programme and advanced papers of the Senseval workshop*, pages 2–4. Citeseer.

Katrin Weller. 2015. Accepting the challenges of social media research. *Online Information Review*.

D Yvette Wohn and Eun-Kyung Na. 2011. Tweeting about tv: Sharing television viewing experiences via social media message streams. *First Monday*.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Rémi Louf, Morgan Funtowicz, et al. 2019. Huggingface's transformers: State-of-the-art natural language processing. *arXiv preprint arXiv:1910.03771*.

Qi Yang, Weinan Wang, Lucas Pierce, Rajan Vaish, Xiaolin Shi, and Neil Shah. 2021. Online communication shifts in the midst of the covid-19 pandemic: A case study on snapchat. *Proceedings of the International AAAI Conference on Web and Social Media*, 15(1):830–840.

Wayne Xin Zhao, Jing Jiang, Jianshu Weng, Jing He, Ee-Peng Lim, Hongfei Yan, and Xiaoming Li. 2011. Comparing twitter and traditional media using topic models. In *European conference on information retrieval*, pages 338–349. Springer.

Ekaterina Zhuravskaya, Maria Petrova, and Ruben Enikolopov. 2020. Political effects of the internet and social media. *Annual Review of Economics*, 12:415–438.

A Tweet filtering

Figure 6 illustrates the weekly trend filtering pipeline utilized. Figure 7 displays the weekly distribution of the top 15 trending topics used to query the raw tweets.

B Annotation Interface

Figure 9 presents our annotation interface. Figure 8 displays the instructions provided to annotators along with a small description of each topic.

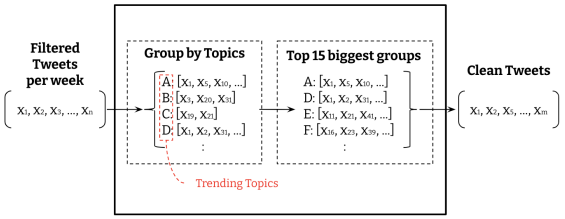

Figure 6: Weekly trend filtering to remove tweets that are irrelevant to the popular topics in each week.

Figure 7: Ratio (%) of tweets in each of top 15 trending keywords for every week.

Instructions

Choose the appropriate topics expressed by the text. Do not work on this HIT if you have already done it before. You can work on the HIT only once, and if we find more than one job from single worker, we have to reject it. There are 3 sentences (randomly located) in our HIT that are designed to qualify the annotations whether they give correct topic annotations or not. We can only accept jobs without any errors in those test sentences, and have to reject the job if it has any mistakes in those sentences.

1. **Arts & Culture:** Content that focuses on the creation and appreciation of various art forms, which evinces some degree of talent, training, or professionalism.
2. **Diaries & Daily Life:** Slice of life, everyday content that illustrates personal opinions, feelings, occasions, and lifestyles.
3. **Beauty:** Content focusing on beautification, products, treatments, and editing in order to improve aesthetics.
4. **Business & Entrepreneurs:** Content that relates to money, the economy, and wealth creation broadly. Including job tips, career advice, and day in the life.
5. **Celebrity & Pop Culture:** Stars and celebrities, their lives, funny moments, relationships, and fan communities.
6. **Fashion & Style:** Content about fashion, the industry, outfits, looks, shows, street style, collections, and designers. Both amateur and professional.
7. **Film, TV & Video:** Traditional media and entertainment, including film and tv, as well as content about Netflix and other streaming shows.
8. **Comedy:** Content intended to make the viewer laugh, either by capturing a funny moment, relaying a story, or contriving a situation.
9. **Gaming:** Games both real and virutal, the competition, culture, and gameplay itself.
10. **Pets & Animals:** All pets and animals content not contemplated by a child category, including alternative pets (fish, reptiles, birds).
11. **Relationships:** Relationship dynamics, jokes, relatable moments, and the like between friend groups and romantic partners.
12. **Science & Technology:** Content cutting-edge technology, natural phenomena, as well as knowledge and theories about the future and the universe.
13. **Family:** Family dynamics, in-jokes, and everyday moments.
14. **Music:** Music performance, discussion, experiences and the like.
15. **Travel & Adventure:** Vacations, travel tips, lodgins, means of conveyence, and the experience of travel.
16. **Home Improvement & Design:** Videos about designing and creating homes and buildings. Home improvement and design as an artistic and/or constructive process.
17. **Food & Dining:** Cooking, restaurants, food visuals, reviews, secret spots, food deals, technique, and ASMR. Anything related to food and food culture.
18. **Youth & Student Life:** Moments and memes of life at school and in the classroom, including teachers, events, and the like.
19. **Learning & Educational:** Instructive, informative, educational content that teaches a fact, skill, or topic.
20. **Fitness & Health:** Healthy living and the components thereof, including nutrition, exercise, progress, and wellness.
21. **Sports:** All depictions of sports whether enumerated below or not.
22. **News & Social Concern:** Awareness, activism, dialogue, and discussion of social and societal issues and injustices, contents that focus on coverage of newsworthy events, political and otherwise.
23. **Other Hobbies:** Pastimes, recreation, and subcultures around hobbies and personal interests.

Please check all the relevant topics to the text, when the topic is mixed.

Make sure that you check at minimum one topic in each text.

If you are unclear or have general feedback for us, feel free to use the comments box.

Figure 8: The instructions shown to the annotators during the annotation phase.

1. Manchester United fans react to starting line-up vs West Ham as Harry Maguire starts #mufc

- ☐ Arts & Culture
- ☐ Diaries & Daily Life
- ☐ Beauty
- ☐ Business & Entrepreneurs
- ☐ Celebrity & Pop Culture
- ☐ Fashion & Style
- ☐ Film, TV & Video
- ☐ Comedy
- ☐ Gaming
- ☐ Pets & Animals
- ☐ Relationships
- ☐ Science & Technology
- ☐ Family
- ☐ Music
- ☐ Travel & Adventure
- ☐ Home Improvement & Design
- ☐ Food & Dining
- ☐ Youth & Student Life
- ☐ Learning & Educational
- ☐ Fitness & Health
- ☐ Sports
- ☐ News & Social Concern
- ☐ Other Hobbies

Figure 9: Tweet classification annotation interface. Annotators are allowed to select multiple topics.

C Evaluation Results

Hyperparameters. Language models are trained using a batch size of 8 for 20 epochs, while utilizing an Adam optimizer (Loshchilov and Hutter, 2017) with learning rate $2e^{-}5$ and a weight decay of 0.01. Furthermore, an early stop callback terminates the training process after 3 epochs without performance improvement. Finally, for the single-label experiments cross entropy loss along with a softmax activation function were used, while for the multi-label setting binary cross entropy loss and a sigmoid activation for each of the 19 topics are used.

Analysis by quarter. In order to get a better understanding of the evolution of the corpus and identify potential performance decays due to temporal differences we inspect the performance of TimeLM-19 in each quarter (i.e., three months) of the temporal's split test-set. Figure 10 displays the F1 scores of each class (single-label setting) for each quarter of the time period tested. While most topics do not seem to be greatly affected by time, we can indeed observe a performance drop in *arts & culture*, which is the topic more affected by the temporal variable. Figure 11 illustrates the relative differences in F1 scores for each class in the multi-label setting, when TimeLM-19 is trained using the temporal split and when trained on the random split.

Confusion matrices. Figure 12 displays the confusion matrices for TimeLM-19 when trained in the multi-label setting using the temporal split.

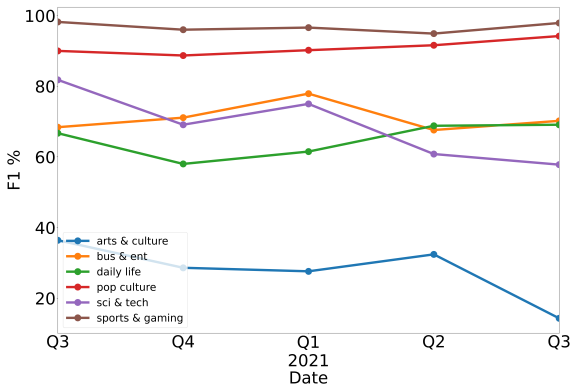

Figure 10: F1 performance of TimeLM-19 through time (single-label setting).

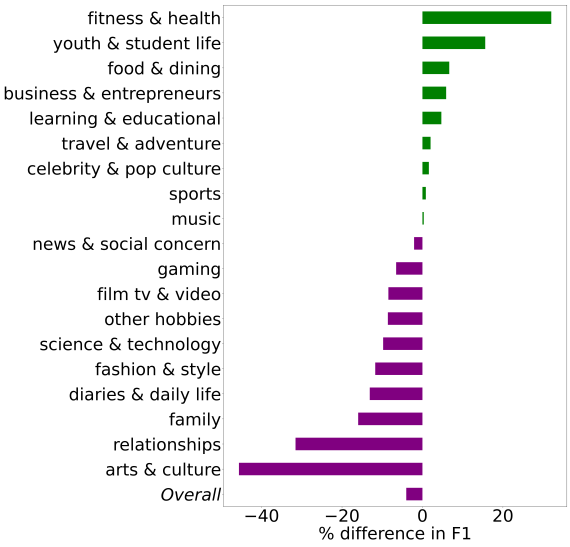

Figure 11: Relative (%) differences in F1 scores when *TimeLM-19* is trained in a temporal and in a random setting for the multi-label setting. Negative values indicate that when using the temporal split the model's performance decreases.

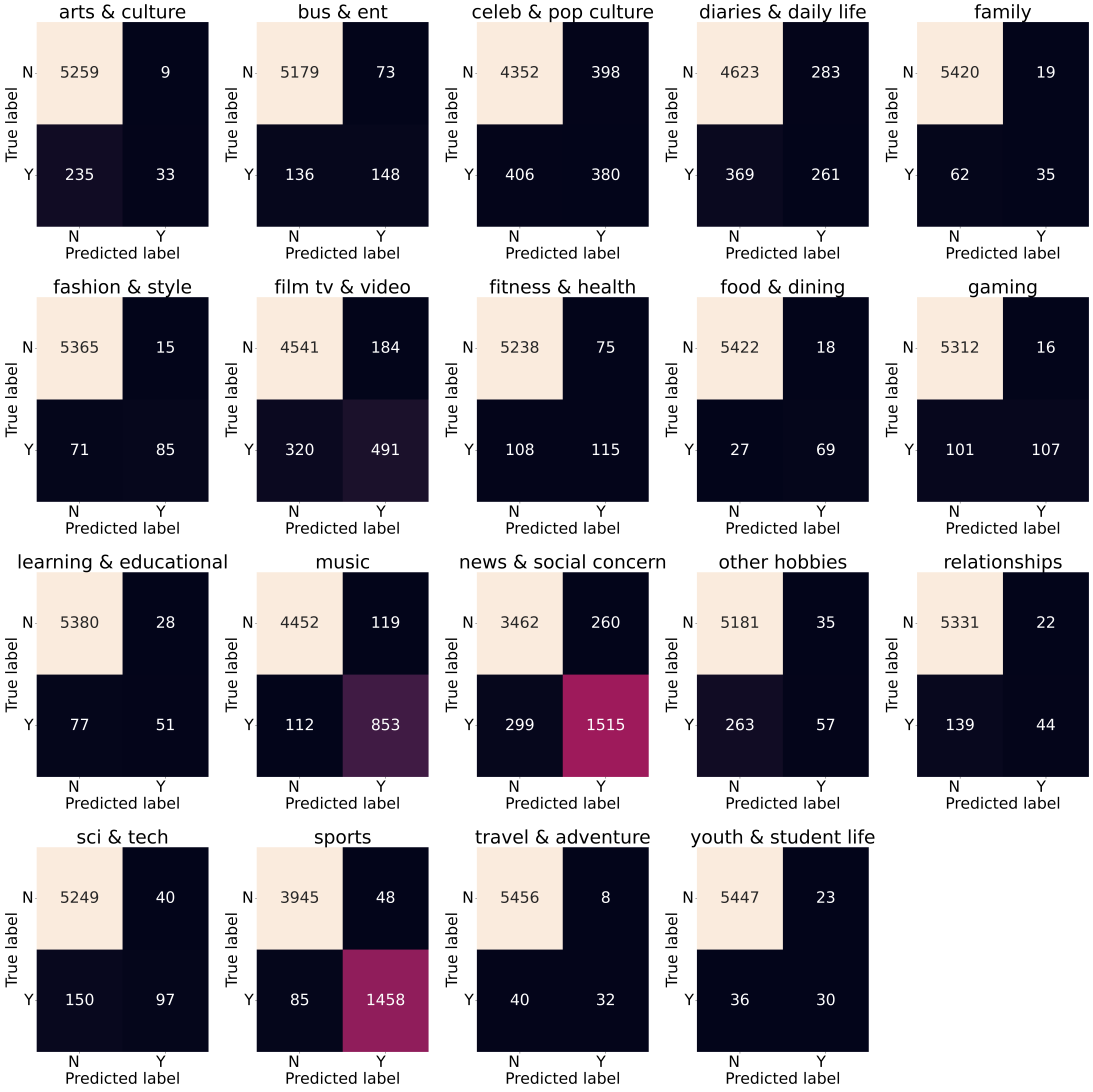

Figure 12: Confusion-matrix of TimeLM-19 (multi-label setting).

Layer or Representation Space: What makes BERT-based Evaluation Metrics Robust?

Doan Nam Long Vu[1], **Nafise Sadat Moosavi**[2], **Steffen Eger**[3]

[1] Department of Computer Science, Technical University of Darmstadt, Germany
[2] Department of Computer Science, The University of Sheffield, UK
[3] NLLG, Faculty of Technology, Bielefeld University, Germany
doannamlong.vu@stud.tu-darmstadt.de

Abstract

The evaluation of recent embedding-based evaluation metrics for text generation is primarily based on measuring their correlation with human evaluations on standard benchmarks. However, these benchmarks are mostly from similar domains to those used for pretraining word embeddings. This raises concerns about the (lack of) generalization of embedding-based metrics to new and noisy domains that contain a different vocabulary than the pretraining data. In this paper, we examine the robustness of BERTScore, one of the most popular embedding-based metrics for text generation. We show that (a) an embedding-based metric that has the highest correlation with human evaluations on a standard benchmark can have the lowest correlation if the amount of input noise or unknown tokens increases, (b) taking embeddings from the first layer of pretrained models improves the robustness of all metrics, and (c) the highest robustness is achieved when using character-level embeddings, instead of token-based embeddings, from the first layer of the pretrained model.[1]

1 Introduction

Evaluating the quality of generated outputs by Natural Language Generation (NLG) models is a challenging and open problem. Human judgments can directly assess the quality of generated texts (Popović, 2020; Escribe, 2019). However, human evaluation, either with experts or crowdsourcing, is expensive and time-consuming. Therefore, automatic evaluation metrics, which are fast and cheap, are commonly used alternatives for the rapid development of text generation systems (van der Lee et al., 2019). Traditional metrics such as BLEU (Papineni et al., 2002), METEOR (Banerjee and Lavie, 2005), and ROUGE (Lin, 2004) measure n-gram overlap between generated and reference texts. While these metrics are easy to use, they cannot correctly assess generated texts that contain novel words or a rephrasing of the reference text.

Recent metrics like BERTScore (Zhang et al., 2020), MoverScore (Zhao et al., 2019), COMET (Rei et al., 2020), BARTScore (Yuan et al., 2021), and BLEURT (Sellam et al., 2020) adapt pretrained contextualized word embeddings to tackle this issue. These novel metrics have shown higher correlations with human judgments on various tasks and datasets (Ma et al., 2019; Mathur et al., 2020). However, the correlations are measured on standard benchmarks containing text domains similar to those used for pretraining the embeddings themselves. As a result, it is unclear how reliable these metrics are on domains and datasets containing words outside the vocabulary of the pretraining data.

The goal of this paper is to investigate the *robustness* of embedding-based evaluation metrics on new and noisy domains that contain a higher ratio of unknown tokens compared to standard text domains.[2] We examine the robustness of BERTScore, one of the most popular recent metrics for text generation.[3] In order to perform a systematic evaluation on the robustness of BERTScore with regard to the ratio of unknown tokens, we use character-based adversarial attacks (Eger and Benz, 2020) that introduce a controlled ratio of new unknown tokens to the input texts. Our contributions are:

- We investigate whether the use of character-based embeddings instead of token-based embeddings improves the robustness of embedding-based generation metrics. Our results show that

[1] The code of our experiments is available at https://github.com/long21wt/robust-bert-based-metrics

[2] We connect to recent research that investigates the behavior of metrics in adversarial situations (Sai et al., 2021; Kaster et al., 2021; Leiter et al., 2022; Zeidler et al., 2022).

[3] E.g., as of September 2022, BERTScore is cited ∼1200 times while it is ∼200 and 400 for MoverScore and BLEURT, respectively.

Proceedings of the 29th International Conference on Computational Linguistics, pages 3401–3411
October 12–17, 2022.

the evaluations based on character-level embeddings are more robust.

- We examine the impact of the hidden layer used for computing the embeddings in BERTScore. We show that the choice of hidden layer affects the robustness of the evaluation metric.
- We show that by using **character-level embeddings from the first layer**, we achieve the highest robustness, i.e., similar correlation with human evaluations for different ratios of unknown tokens.

2 BERTScore

BERTScore (Zhang et al., 2020) computes the pairwise cosine similarity between the reference and hypothesis using contextual embeddings. It forward-passes sentences through a pretrained model, i.e., BERT (Devlin et al., 2019), and extracts the embedding information from a specific hidden layer. To select the best hidden layer, BERTScore uses average Pearson correlation with human scores on WMT16 (Bojar et al., 2016) over five language pairs. For instance, the best layer is the ninth layer for $\text{BERT}_{\text{base-uncased}}$.

BERTScore with character-level embeddings. Existing embedding-based metrics, including BERTScore, use token-based embeddings that are taken from pretrained models like BERT (Devlin et al., 2019). In this paper, we investigate the impact of using character-level embeddings instead of token-level embeddings in BERTScore (Zhang et al., 2020). We use ByT5 (Xue et al., 2021), which encodes the input at the byte level. It tokenizes a word into a set of single characters or converts it directly to UTF-8 characters before forwarding the input sequence into the model. Xue et al. (2021) show that ByT5 is more robust to noise compared to word-level embeddings. For computing BERTScore using character-level embeddings, we use ByT5 instead of BERT in BERTScore computations. We adapt three variants of ByT5 (small, base, large) in BERTScore. Table 1 presents the best layer of ByT5 models for computing BERTScore.

3 Experimental settings

3.1 Evaluation on a standard benchmark

We report the results on the WMT19 dataset (Ma et al., 2019) that contains seven to-English language pairs. Each language pair has 2800 sentences, each corresponding to one reference, plus the systems' output sentences. Totally, the human evaluation in WMT19 has 281k segment sample scores for each of the output translation in to-English language pairs. Table 2 shows the language pairs considered, as well as the number of segments per language pair.

Model	Best Layer	Score
ByT5-small	1	0.510
ByT5-base	17	0.581
ByT5-large	30	0.615

Table 1: Best layers with different ByT5 variants and their average Pearson correlation score on WMT16.

Language Pairs	No. Segment Sample (DARR)
de-en (German-English)	85365
fi-en (Finnish-English)	38307
gu-en (Gujarati-English)	31139
kk-en (Kazakh-English)	27094
lt-en (Lithuanian-English)	21862
ru-en (Russian-English)	46172
zh-en (Chinese-English)	31070

Table 2: To-English language pairs of WMT19. DARR denotes *Direct Assessment Relative Ranks*, in which all available sentence pairs of DA (Direct Assessment) scores are taken into account.

3.2 Evaluating Robustness

Evaluation on different ratios of unknown tokens. To evaluate the robustness of evaluation metrics on new domains, we use character-level attacks to introduce a controlled ratio of unknown tokens in the corresponding reference texts of the evaluation sets.[4] We examine five different attacks from Eger and Benz (2020): (a) **intruders:** inserting a character—e.g., '.', '/', ':'—in between characters of a word, (b) **disemvoweling**: removing vowels—e.g., 'a', 'e', 'i'—from the word, (c) **keyboard typos**: randomly replacing letters of a word with characters that are nearby the original characters on an English keyboard, (d) **phonetic**: changing a word's spelling in such a way that its pronunciation stays the same, and (e) **visual**: replacing characters with a symbol that is its visually nearest neighbor (Eger et al., 2019). We can control the ratio of tokens that are affected by the adversar-

[4]We need human annotations for evaluating the correlation of evaluation metrics with human judgments, and such annotations are available for standard domains like WMT datasets. As a result, we introduce unknown tokens by using character-level attacks to artificially introduce more unknown tokens.

Setting	Sentence
no-attack	Now they have come to an agreement.
intrude	Now they have c/o/me t+o a>n agreement.
disemvowel	Nw thy have come to an grmnt.
keyboard-typo	No3 they have come to xn agrrement.
phonetic	Nau they have cohm to an agrimand.
visual	Now thĘŶ hăve come to ă agɣèĘmĒnț.

Table 3: Examples for the character-level attacks (Eger and Benz, 2020; Keller et al., 2021) at perturbation level $p = 0.3$, i.e., the probability that each letter in a sentence is attacked is 0.3.

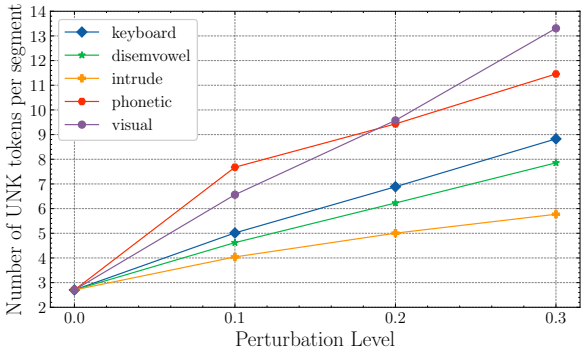

Figure 1: The number of average unknown tokens per segment across seven to-English language pairs in WMT19 given different attacks and perturbation levels.

Language pair	No. unknown tokens
bn-hi (Bengali-Hindi)	19.235
hi-bn (Hindi-Bengali)	23.478
xh-zu (Xhosa-Zulu)	28.930
zu-xh (Zulu-Xhosa)	28.743

Table 4: The number of average unknown tokens per segment for each language pair in our low-resource datasets.

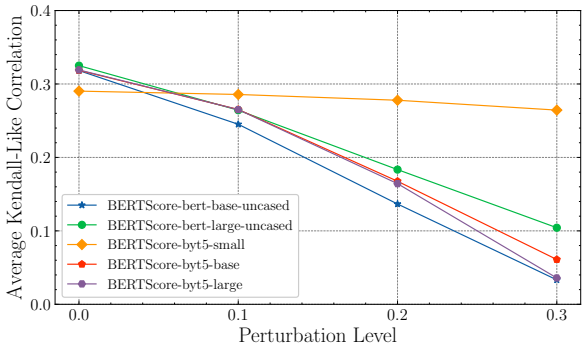

Figure 2: Average Kendall correlation of 7 to-English language pairs in WMT19 given different perturbation level from $p = 0.0$ to $p = 0.3$ using the visual attack.

ial attack by the *perturbation level* (p), e.g., $p = 0$ denotes no attack and $p = 0.3$ indicates that each letter in the sentence is attacked by the probability of 0.3. Table 3 shows an example of each of these attacks at $p = 0.3$.

Figure 1 shows the average number of unknown tokens, as determined based on BERT's tokenizer, per segment across seven to-English language pairs given different attacks and perturbation levels. We count a token as an unknown token if (1) BERT represents it as [UNK], or (2) BERT splits it into subwords, e.g., *'pre-trained'* to *'pre', '##train', '##ed'*.[5] As we see from the figure, the number of unknown tokens increases as we apply these character-level attacks with higher perturbation levels. In our experiments in Section 4, we report the results using visual attacks. The results using other attacks are also reported in Appendix B, and they follow the same patterns as those using the visual attack.

Evaluation on low-resource language pairs. Apart from the experiments on WMT19, we also perform the evaluations on the (Xhosa, Zulu) and (Bengali, Hindi) language pairs from WMT21 (Freitag et al., 2021). BERTScore uses multilingual BERT for evaluating non-English languages. Multilingual models contain a higher ratio of unknown tokens for low-resource languages, and therefore, evaluating the correlation of embedding-based metrics with human judgments on low-resource languages is also an indicator of their robustness. Table 4 shows the number of unknowns tokens per segment to multilingual BERT in four different low-resources language pairs in WMT21 dataset. We refer to the number of segments of low-resources dataset in Table 7 in Appendix C.

4 Experiments

4.1 Impact of Character-level Embeddings

Table 5 shows the results of BERTScore using different embeddings on WMT19's to-English language pairs (using $p = 0$). Figure 2 shows the average correlation score over all seven to-English language pairs given different perturbation level from $p = 0$ to $p = 0.3$ using the visual attack.

We observe that computing BERTScore using the ByT5-small models results in a slightly lower average correlation with human scores over the seven to-English pairs at $p = 0$ compared to BERTScore using BERT and larger ByT5 models.

[5] Please refer to the detailed algorithm in Appendix A.

	de-en	fi-en	gu-en	kk-en	lt-en	ru-en	zh-en	Average
BERT-base	0.180	0.339	0.288	0.438	0.364	0.209	0.410	0.318
BERT-large	0.194	**0.346**	0.292	**0.442**	**0.375**	0.208	**0.418**	**0.325**
ByT5-small	0.172	0.286	0.278	0.422	0.307	0.194	0.373	0.290
ByT5-base	**0.197**	0.326	0.297	0.419	0.358	**0.215**	**0.418**	0.319
ByT5-large	0.193	0.333	**0.304**	0.427	0.354	0.208	0.415	0.319

Table 5: Segment-level Kendall correlation results for to-English language pairs in WMT19 without any attack, i.e. $p = 0$. The correlation of BERTScore with human are reported using different embeddings including bert-base-uncased, bert-large-uncased, ByT5-small, ByT5-base, and ByT5-large.

Model	bn-hi	hi-bn	xh-zu	zu-xh
BERT-multi	0.073	0.364	0.266	0.488
ByT5-small	**0.096**	**0.411**	**0.311**	**0.523**

Table 6: Kendall correlation scores of BERTScore for WMT21 low-resource language pairs Hindi-Bengali and Zulu-Xhosa using BERT-base-multilingual and ByT5-small embeddings.

However, the average correlation using ByT5-small remains around the same value given different ratio of unknown tokens, indicating higher *robustness* of the metrics using ByT5-small. On the other hand, while using BERT-large embeddings results in the highest average correlation with human scores in Table 5, its correlation drops considerably in the presence of more unknown tokens in Figure 2.

For Hindi-Bengali and Zulu-Xhosa, we compare the results against using the BERT-base-multilingual model in Table 6. We observe that the BERTScore metric that uses ByT5-small achieves higher correlations with humans throughout. Given that low resources languages contain more out-of-vocabulary words for pretrained models, this observation confirms our previous results using character-level attacks on the WTM19 dataset.

4.2 Impact of the Selected Hidden Layer

Our results in Section 4.1 show the robustness of BERTScore when using the ByT5-small model for computing the embeddings. However, as Table 1 shows, the selected hidden layer for getting embeddings varies when using different pretrained models. For instance, when using ByT5-small embeddings, the model uses the embeddings of the first layer while it uses the embeddings of the 30th layer for ByT5-large. Zhang et al. (2020) show that BERTScore correlation scores with humans drop as they select the last few layers of BERT for getting the embeddings. Therefore, the robustness of examined metrics may also depend on their corresponding selected layers for computing embeddings.

In this section, we evaluate the impact of the selected hidden layer on the robustness of the metric. We evaluate three settings where we use: (a) the embeddings of the first layer for all models, (b) the embeddings of the best layer for each model (cf. Table 3), and (c) the mean of aggregated embeddings over all layers. We perform the robustness evaluations using the visual attack at $p = 0.3$. Figure 3 shows the average results of this experiment[6]. We make the following observations.

First, using the embeddings of the first layer closes the gap between the correlations of different variations of the ByT5 model, i.e., small, base, and large, in the presence of more unknown tokens, i.e., $p = 0.3$.

Second, using the embeddings of the first layer improves the robustness of BERTScore using BERT embeddings, i.e., improving the correlation from 0.033 to 0.174 for BERT-base given $p = 0.3$. However, the correlation of the resulting BERTScore is still considerably lower than using ByT5 embeddings at the presence of more unknown tokens. This indicates that **both** the choices of the hidden layer as well as the pretrained model play an important role in the robustness of the resulting embedding-based metric. A reason why the first layer may be more effective in our setup is that, in the presence of input noise or unknown tokens, embeddings of higher layers may become less and less meaningful, as the noise may propagate and accumulate along layers. We provide an example from the similarity matrix of the resulting embeddings for different layers in Figure 5 in the Appendix E.

Overall, our results indicate that optimizing the layer on a standard data set such as WMT16 may

[6]In Table 8 and 9 in Appendix D, we report scores for each language pair.

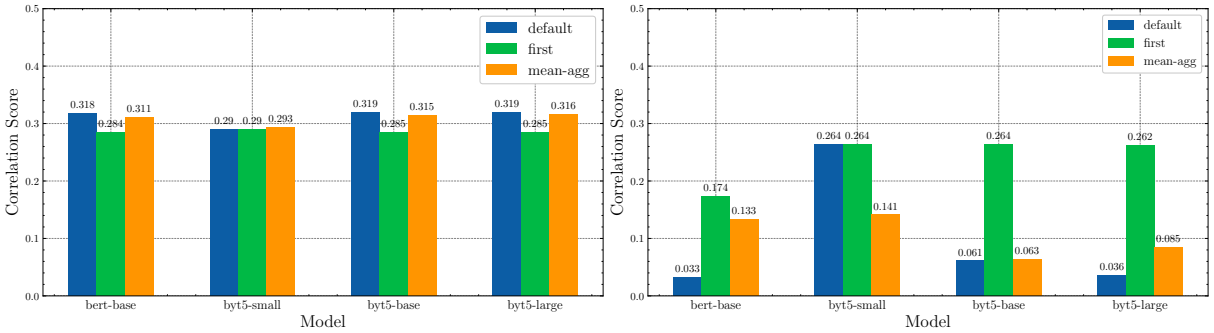

(a) No attack

(b) Visual attack at $p = 0.3$

Figure 3: Average segment-level Kendall correlation results for seven to-non-English language pairs in WMT19 to fist layer, default layer, and mean of aggregated embeddings setting in BERTScore.

be suboptimal in terms of the generalization of the resulting metrics. Concerning efficiency of the resulting metrics (a core aspect of modern NLP (Moosavi et al., 2020)), BERT-base has 110 million parameters, while ByT5-small has 300 million parameters. With the default BERTScore setting, passing the input through 9 layers results in a longer inference time. However, using the embeddings of the first layer results in a very fast inference for both models.

5 Conclusion

Embedding-based evaluation metrics will be used across different tasks and datasets that may contain data from very different domains. However, such metrics are only evaluated on standard datasets that contain similar domains as those used for pretraining embeddings. As a result, it is not clear how reliable the results of such evaluation metrics will be on new domains. In this work, we investigate the robustness of embedding-based metrics in the presence of different ratios of unknown tokens. We show that (a) the results of the BERTScore using BERT-based embeddings is not robust, and its correlation with human evaluations drops significantly as the ratio of unknown tokens increases, and (b) using character-level embeddings from the first layer of ByT5 significantly improves the robustness of BERTScore and results in reliable results given different ratios of unknown tokens. We encourage the community to use this setting for their embedding-based evaluations, especially when applying the metrics to less standard domains.

In future work, we aim to address other aspects of robustness of evaluation metrics beyond an increased amount of unknown tokens as a result of spelling variation, such as how metrics cope with varying factuality (Chen and Eger, 2022) or with fluency and grammatical acceptability issues (Rony et al., 2022). We also plan to investigate the impact of pixel-based representations (Rust et al., 2022) (which are even more lower-level) for enhancing the robustness of evaluation metrics.

Acknowledgments

We thank the Ubiquitous Knowledge Processing (UKP) Lab and the system administrators of the lab for providing the computing resources to perform the experiments. We thank the anonymous reviewers for their comments and suggestions that improved the final version of the paper.

References

Satanjeev Banerjee and Alon Lavie. 2005. METEOR: An automatic metric for MT evaluation with improved correlation with human judgments. In *Proceedings of the ACL Workshop on Intrinsic and Extrinsic Evaluation Measures for Machine Translation and/or Summarization*, pages 65–72, Ann Arbor, Michigan. Association for Computational Linguistics.

Ondřej Bojar, Yvette Graham, Amir Kamran, and Miloš Stanojević. 2016. Results of the WMT16 metrics shared task. In *Proceedings of the First Conference on Machine Translation: Volume 2, Shared Task Papers*, pages 199–231, Berlin, Germany. Association for Computational Linguistics.

Yanran Chen and Steffen Eger. 2022. Menli: Robust evaluation metrics from natural language inference. *ArXiv*, abs/2208.07316.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for*

Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers), pages 4171–4186, Minneapolis, Minnesota. Association for Computational Linguistics.

Steffen Eger and Yannik Benz. 2020. From hero to zéroe: A benchmark of low-level adversarial attacks. In *Proceedings of the 1st Conference of the Asia-Pacific Chapter of the Association for Computational Linguistics and the 10th International Joint Conference on Natural Language Processing, AACL/IJCNLP 2020, Suzhou, China, December 4-7, 2020*, pages 786–803. Association for Computational Linguistics.

Steffen Eger, Gözde Gül Şahin, Andreas Rücklé, Ji-Ung Lee, Claudia Schulz, Mohsen Mesgar, Krishnkant Swarnkar, Edwin Simpson, and Iryna Gurevych. 2019. Text processing like humans do: Visually attacking and shielding NLP systems. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 1634–1647, Minneapolis, Minnesota. Association for Computational Linguistics.

Marie Escribe. 2019. Human evaluation of neural machine translation: The case of deep learning. In *Proceedings of the Human-Informed Translation and Interpreting Technology Workshop (HiT-IT 2019)*, pages 36–46, Varna, Bulgaria. Incoma Ltd., Shoumen, Bulgaria.

Markus Freitag, Ricardo Rei, Nitika Mathur, Chi-kiu Lo, Craig Stewart, George Foster, Alon Lavie, and Ondřej Bojar. 2021. Results of the WMT21 metrics shared task: Evaluating metrics with expert-based human evaluations on TED and news domain. In *Proceedings of the Sixth Conference on Machine Translation*, pages 733–774, Online. Association for Computational Linguistics.

Marvin Kaster, Wei Zhao, and Steffen Eger. 2021. Global explainability of BERT-based evaluation metrics by disentangling along linguistic factors. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 8912–8925, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Yannik Keller, Jan Mackensen, and Steffen Eger. 2021. BERT-defense: A probabilistic model based on BERT to combat cognitively inspired orthographic adversarial attacks. In *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*, pages 1616–1629, Online. Association for Computational Linguistics.

Chris van der Lee, Albert Gatt, Emiel van Miltenburg, Sander Wubben, and Emiel Krahmer. 2019. Best practices for the human evaluation of automatically generated text. In *Proceedings of the 12th International Conference on Natural Language Generation*, pages 355–368, Tokyo, Japan. Association for Computational Linguistics.

Christoph Leiter, Piyawat Lertvittayakumjorn, M. Fomicheva, Wei Zhao, Yang Gao, and Steffen Eger. 2022. Towards explainable evaluation metrics for natural language generation. *ArXiv*, abs/2203.11131.

Chin-Yew Lin. 2004. ROUGE: A package for automatic evaluation of summaries. In *Text Summarization Branches Out*, pages 74–81, Barcelona, Spain. Association for Computational Linguistics.

Qingsong Ma, Johnny Wei, Ondřej Bojar, and Yvette Graham. 2019. Results of the WMT19 metrics shared task: Segment-level and strong MT systems pose big challenges. In *Proceedings of the Fourth Conference on Machine Translation (Volume 2: Shared Task Papers, Day 1)*, pages 62–90, Florence, Italy. Association for Computational Linguistics.

Nitika Mathur, Johnny Wei, Markus Freitag, Qingsong Ma, and Ondřej Bojar. 2020. Results of the WMT20 metrics shared task. In *Proceedings of the Fifth Conference on Machine Translation*, pages 688–725, Online. Association for Computational Linguistics.

Nafise Sadat Moosavi, Angela Fan, Vered Shwartz, Goran Glavaš, Shafiq Joty, Alex Wang, and Thomas Wolf, editors. 2020. *Proceedings of SustaiNLP: Workshop on Simple and Efficient Natural Language Processing*. Association for Computational Linguistics, Online.

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. Bleu: a method for automatic evaluation of machine translation. In *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics*, pages 311–318, Philadelphia, Pennsylvania, USA. Association for Computational Linguistics.

Maja Popović. 2020. Informative manual evaluation of machine translation output. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 5059–5069, Barcelona, Spain (Online). International Committee on Computational Linguistics.

Ricardo Rei, Craig Stewart, Ana C Farinha, and Alon Lavie. 2020. COMET: A neural framework for MT evaluation. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 2685–2702, Online. Association for Computational Linguistics.

Md Rashad Al Hasan Rony, Liubov Kovriguina, Debanjan Chaudhuri, Ricardo Usbeck, and Jens Lehmann. 2022. RoMe: A robust metric for evaluating natural language generation. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 5645–5657, Dublin, Ireland. Association for Computational Linguistics.

Phillip Rust, J.F. Lotz, Emanuele Bugliarello, Elizabeth Salesky, Miryam de Lhoneux, and Desmond Elliott. 2022. Language modelling with pixels. *ArXiv*, abs/2207.06991.

Ananya B. Sai, Tanay Dixit, Dev Yashpal Sheth, Sreyas Mohan, and Mitesh M. Khapra. 2021. Perturbation CheckLists for evaluating NLG evaluation metrics. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 7219–7234, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Thibault Sellam, Dipanjan Das, and Ankur Parikh. 2020. BLEURT: Learning robust metrics for text generation. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7881–7892, Online. Association for Computational Linguistics.

Linting Xue, Aditya Barua, Noah Constant, Rami Al-Rfou, Sharan Narang, Mihir Kale, Adam Roberts, and Colin Raffel. 2021. Byt5: Towards a token-free future with pre-trained byte-to-byte models.

Weizhe Yuan, Graham Neubig, and Pengfei Liu. 2021. Bartscore: Evaluating generated text as text generation. *Advances in Neural Information Processing Systems*, 34.

Laura Zeidler, Juri Opitz, and Anette Frank. 2022. A dynamic, interpreted CheckList for meaning-oriented NLG metric evaluation – through the lens of semantic similarity rating. In *Proceedings of the 11th Joint Conference on Lexical and Computational Semantics*, pages 157–172, Seattle, Washington. Association for Computational Linguistics.

Tianyi Zhang, Varsha Kishore, Felix Wu, Kilian Q. Weinberger, and Yoav Artzi. 2020. Bertscore: Evaluating text generation with bert. In *International Conference on Learning Representations*.

Wei Zhao, Maxime Peyrard, Fei Liu, Yang Gao, Christian M. Meyer, and Steffen Eger. 2019. MoverScore: Text generation evaluating with contextualized embeddings and earth mover distance. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 563–578, Hong Kong, China. Association for Computational Linguistics.

Algorithm 1: Count UNK token in a BERT tokenized sentence

```
def count_UNK:
    Data: sentence: a tokenized sentence as
          a list of string
    output: count: number of UNK token
            of input tokenized sentence
    count ⟵ 0
    buffer ⟵ empty list
    for token in sentence do
        if [UNK] in token then
            count ⟵ count +1
        else if ## in token then
            Add token to buffer
        else
            if len(buffer) != 0 then
                count ⟵ count +1
                Empty buffer
            end
        end
    end
    if len(sentence) ≥ 2 then
        if ## in last token of sentence then
            count ⟵ count +1
        end
    end
    return count
```

A Counting UNK token

Algorithm 1 shows how we count UNK tokens that the BERT tokenizer creates from a sentence. In BERT, `[UNK]` represents the UNK tokens that are not in their given vocabulary. Besides `[UNK]`, BERT use WordPiece tokenizer concept, which breaks the unknown word into sub-words using a greedy longest-match-first algorithm, such as splits "*bassing*" into '*bass*' and '*##ing*' where '*##*' denotes the join of sub-words. Thus, the UNK word becomes two known words. '*##*' is the indication for the starting of a UNK word if the previous token does not contain '*##*'. In case the next token still contains '*##*', it indicates that the token still belongs to a word and does not count as a UNK token, e.g., "*verständlich*" to *'vers', '##tä', '##nd', '##lich'* and count it as one UNK token. It lasted until we finally found non contain '*##*' token. With a word-piece tokenizer, the beginning token of a tokenized sentence is either `[UNK]` or known word, and we also consider the case where the last token contains "*##*".

Language Pair	No. Segment
bn-hi (Bengali → Hindi)	4,461
hi-bn (Hindi → Bengali)	4,512
xh-zu (Xhosa → Zulu)	2,952
zu-xh (Zulu → Xhosa)	2,502

Table 7: Amount of segments in WMT21 for Hindi ⟷ Bengali and Zulu ⟷ Xhosa.

B WMT19

The results of other attacks are illustrated in Figure 4.

C FLORES

Table 7 shows the number of provided human annotations in FLORES.

D Impact of layer choice in BERTScore

Table 8 and 9 show the particular results of each language pair with different settings in BERTScore without attack and with visual attack at $p = 0.3$ respectively.

E Effectiveness of the first layer

In Figure 5, we show four different settings and their cosine similarity matrix computed by BERTScore using bert-base-uncased. In both *normal reference* with 1st or 9th setups, matched tokens get higher similarity score. 9th layer setting gathers information for relevant tokens, which results in higher similarity score across the matrix. As in the case with *attacked reference*, 1st layer setting penalizes the unmatched tokens and the magnitude of matched tokens are as high as using *normal reference* with 1st layer setup. However, by using 9th layer for *attacked reference*, we can observe the hue color of matched tokens with low score. Thus, we conclude the accumulated noise to higher layer cause the problem with effectiveness in our previous setup with WMT19 dataset.

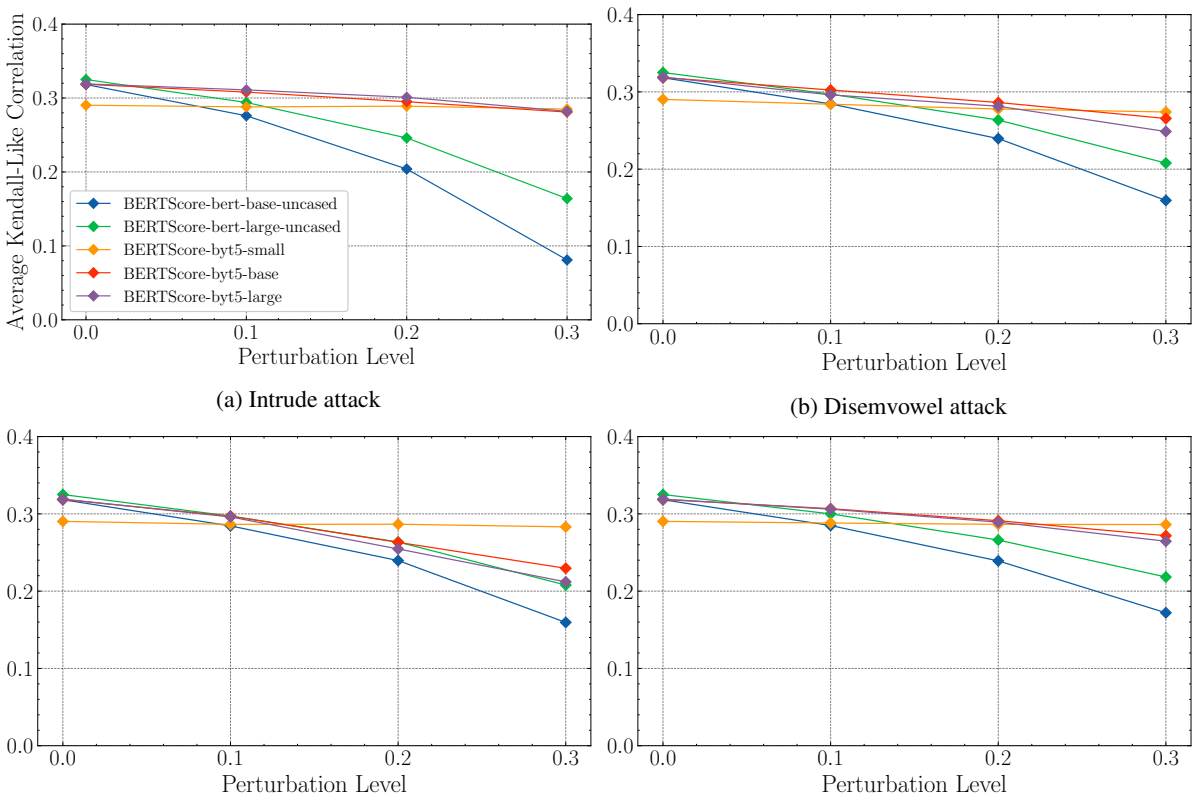

(a) Intrude attack

(b) Disemvowel attack

(c) Phonetic attack

(d) Keyboard-typo attack

Figure 4: Average Kendall correlation of seven to-English language pairs in WMT19 under attack with perturbation level from $p = 0.0$ to $p = 0.3$

Setting	Metric	de-en	fi-en	gu-en	kk-en	lt-en	ru-en	zh-en	Average
Default	BERTScore-bert-base-uncased	0.18	0.339	0.288	0.438	0.364	0.209	0.41	0.318
	BERTScore-byt5-small	0.172	0.286	0.278	0.422	0.307	0.194	0.373	0.290
	BERTScore-byt5-base	**0.197**	0.326	0.297	0.419	0.358	**0.215**	**0.418**	**0.319**
	BERTScore-byt5-large	0.193	0.333	0.304	0.427	0.354	0.208	0.415	**0.319**
First	BERTScore-bert-base-uncased	0.147	0.295	0.263	0.421	0.318	0.183	0.361	0.284
	BERTScore-byt5-small	0.171	0.285	0.279	0.422	0.307	0.194	0.370	0.290
	BERTScore-byt5-base	0.164	0.276	0.280	0.414	0.307	0.191	0.362	0.285
	BERTScore-byt5-large	0.161	0.277	0.280	0.416	0.308	0.189	0.361	0.285
Mean of aggregation	BERTScore-bert-base-uncased	0.17	**0.326**	0.289	**0.437**	**0.351**	0.206	0.397	0.311
	BERTScore-byt5-small	0.170	0.292	0.284	0.420	0.313	0.202	0.372	0.293
	BERTScore-byt5-base	0.188	0.324	**0.305**	0.427	0.347	0.207	0.408	0.315
	BERTScore-byt5-large	0.185	0.322	0.311	0.431	0.343	0.208	0.411	0.316

Table 8: Segment-level correlation metric results Kendall for seven to-non-English language pairs in WMT19 with respect to fist layer, default layer and mean of aggregated embeddings setting without any attack i.e. $p = 0$.

Setting	Metric	de-en	fi-en	gu-en	kk-en	lt-en	ru-en	zh-en	Average
Default	BERTScore-bert-base-uncased	-0.003	-0.014	-0.027	0.149	-0.022	0.024	0.126	0.033
	BERTScore-byt5-small	**0.155**	**0.266**	0.239	0.392	**0.264**	**0.175**	**0.360**	**0.264**
	BERTScore-byt5-base	0.014	-0.009	0.026	0.147	0.052	0.042	0.155	0.061
	BERTScore-byt5-large	0.011	-0.055	-0.018	0.141	-0.015	0.032	0.155	0.036
First	BERTScore-bert-base-uncased	0.074	0.215	0.082	0.215	0.234	0.120	0.278	0.174
	BERTScore-byt5-small	**0.155**	**0.266**	0.239	0.392	**0.264**	**0.175**	**0.360**	**0.264**
	BERTScore-byt5-base	0.147	0.256	**0.262**	**0.403**	0.264	0.166	0.348	**0.264**
	BERTScore-byt5-large	0.138	0.258	0.259	0.394	0.262	0.170	0.352	0.262
Mean of aggregation	BERTScore-bert-base-uncased	0.053	0.144	0.052	0.214	0.149	0.082	0.240	0.133
	BERTScore-byt5-small	0.070	0.089	0.094	0.244	0.109	0.107	0.273	0.141
	BERTScore-byt5-base	0.025	-0.029	0.022	0.263	-0.019	0.056	0.123	0.063
	BERTScore-byt5-large	0.054	0.005	0.020	0.255	0.013	0.095	0.156	0.085

Table 9: Segment-level correlation metric results Kendall for seven to-non-English language pairs in WMT19 with respect to fist layer, default layer and mean of aggregated embeddings setting under visual attack at 0.3 perturbation level.

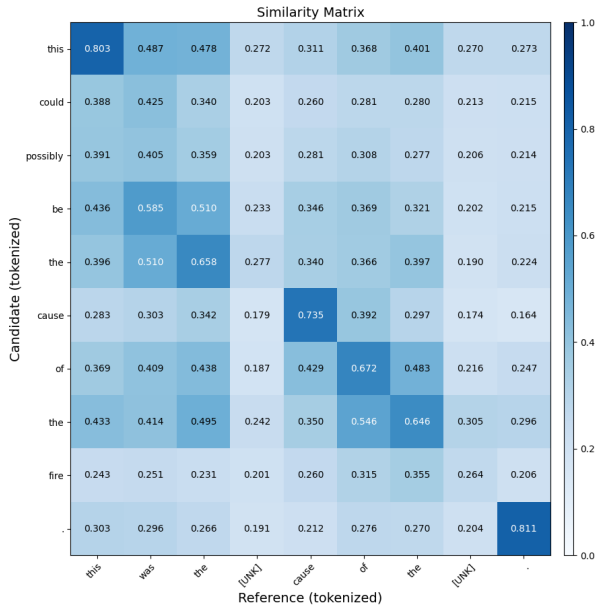

(a) 9th layer, *attacked reference*:
"This was the pos⊿ible caUse of the f(r)e."

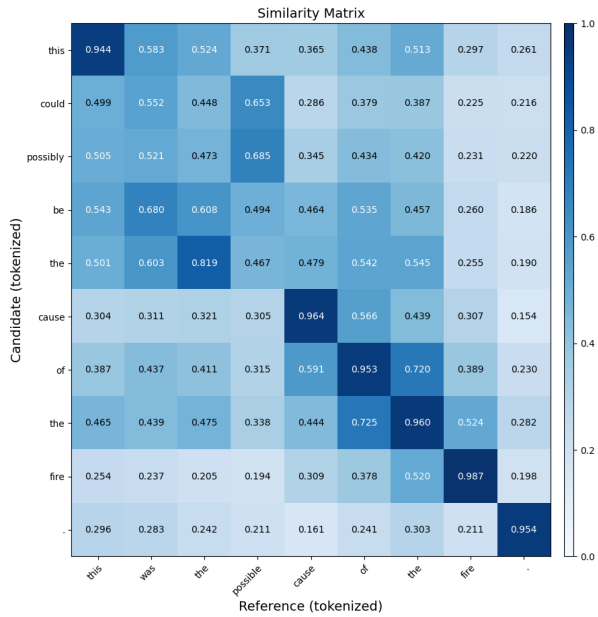

(b) 9th layer, *normal reference*:
"This could possibly be the cause of the fire."

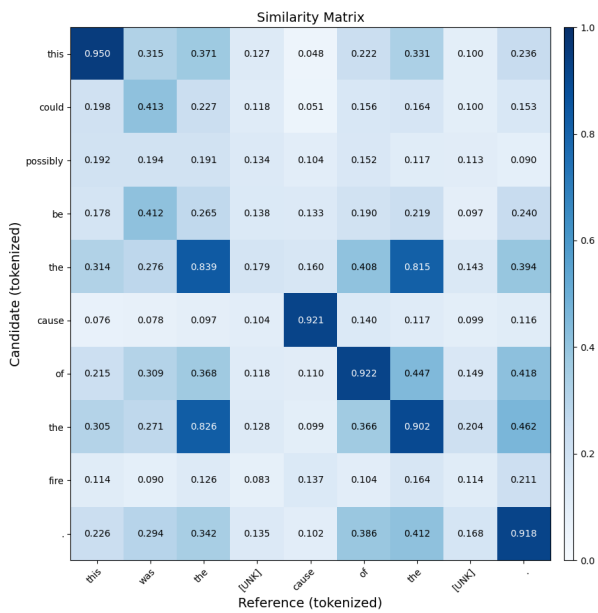

(c) 1st layer, *attacked reference*:
"This was the pos⊿ible caUse of the f(r)e."

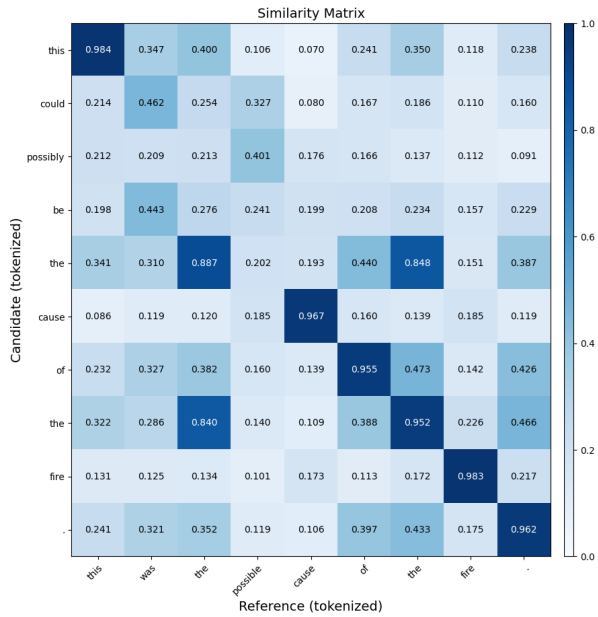

(d) 1th layer, *normal reference*:
" This could possibly be the cause of the fire."

Figure 5: Similarity Matrix using BERTScore with bert-base-uncased for *candidate*: " This could possibly be the cause of the fire." in different setups.

Evaluating the Performance of Transformer-based Language Models for Neuroatypical Language

Duanchen Liu[1,2], **Zoey Liu**[1], **Qingyun Yang**[1,3], **Yujing Huang**[1], **Emily Prud'hommeaux**[1]
[1]Department of Computer Science, Boston College, Chestnut Hill MA, USA
[2]MIT, Cambridge MA, USA [3]Cornell Tech, New York NY, USA
liudc@mit.edu, {liuaal,huangac,prudhome}@bc.edu, qny2@cornell.edu

Abstract

Difficulties with social aspects of language are among the hallmarks of autism spectrum disorder (ASD). These communication differences are thought to contribute to the challenges that adults with ASD experience when seeking employment, underscoring the need for interventions that focus on improving areas of weakness in pragmatic and social language. In this paper, we describe a transformer-based framework for identifying linguistic features associated with social aspects of communication using a corpus of conversations between adults with and without ASD and neurotypical conversational partners produced while engaging in collaborative tasks. While our framework yields strong accuracy overall, performance is significantly worse for the language of participants with ASD, suggesting that they use a more diverse set of strategies for some social linguistic functions. These results, while showing promise for the development of automated language analysis tools to support targeted language interventions for ASD, also reveal weaknesses in the ability of large contextualized language models to model neuroatypical language.

1 Introduction

Autism spectrum disorder (ASD) is a neurodevelopmental disorder with an estimated global prevalence of 1 in 100 worldwide (Zeidan et al., 2022). The majority of people diagnosed with ASD today are verbal (Rose et al., 2016) and have average or above average intellectual ability (Christensen et al., 2016). Nevertheless, young adults with ASD are employed at significantly lower rates than their peers with other neurodevelopmental conditions, including learning disabilities and intellectual disability (Shattuck et al., 2012).

Difficulty with social communication is one of the diagnostic criteria for ASD (American Psychiatric Association, 2013) and is reported to be one of the strongest contributors to negative professional outcomes (Hurlbutt and Chalmers, 2004; Baldwin et al., 2014). For this reason, language skills are commonly targeted for intervention in individuals with ASD (Parsons et al., 2017), but it can be difficult to identify specific areas in need of remediation. Most prior work on quantifying pragmatic deficits in ASD has relied on manual analysis of speech transcripts (Loukusa et al., 2007; Paul et al., 2009; Conlon et al., 2019), a process that is time consuming and requires expertise.

In this paper, we use a spoken language corpus collected specifically for training automatic systems to explore social communication and pragmatics in ASD. The corpus consists of transcribed conversations between adults with and without ASD as they engage with neurotypical interlocutors in collaborative tasks designed to resemble workplace activities. We review the careful manual process of assigning pragmatic feature values and dialog act labels to each utterance. Following recent prior work on these features in other contexts, we propose a BERT-based framework (Devlin et al., 2019) for automatically assigning these values and labels.

Although our models achieve higher accuracy than previous neural and non-neural approaches to these labeling tasks on this dataset, we observe that models for some features show significantly weaker performance on utterances produced by individuals with ASD. An error analysis with logistic mixed-effects regression reveals that these models fail to recognize unusual or idiosyncratic strategies for conveying certain social and pragmatic meanings. Our results, while showing promise for the automated analysis of social communication in ASD, point to an unsurprising but potentially problematic bias in models trained primarily on news and web data toward neurotypical language.

2 Background

Much of the prior work on extracting social communication and discourse features from conversa-

Proceedings of the 29th International Conference on Computational Linguistics, pages 3412–3419
October 12–17, 2022.

tions has focused on dialogue acts, yielding both a large number of corpora and nearly as many distinct annotation schemes (Stolcke et al., 2000; McCowan et al., 2005; Zhang et al., 2017; Bunt et al., 2019). Conversational corpora, mostly written, have also been manually annotated at the utterance level for specific pragmatic features, including politeness (Danescu-Niculescu-Mizil et al., 2013); uncertainty (Vincze, 2014; Farkas et al., 2010); and informativeness, formality, and implicature (Lahiri, 2015). Early work relied primarily on bag-of-words models with statistical classifiers, but recently transformer-based models have been used with success for many of these tasks (Aljanaideh et al., 2020; Hayati et al., 2021; Wu et al., 2020; Żelasko et al., 2021; Wu et al., 2021). We follow this prior work in our use of BERT (Devlin et al., 2019) for predicting feature labels.

While many recent studies have explored automated analysis of language in ASD, particularly in children (Parish-Morris et al., 2016; Adams et al., 2021; Salem et al., 2021), the most relevant to ours is Yang et al. (2021), which introduced a corpus of conversations between adults with and without ASD and neurotypical conversational partners, partially annotated for three pragmatic features. We go beyond this work in three ways. First, we complete the annotation and introduce a new feature, dialog act (see Section 3.2), which has not been studied previously in ASD language. Second, we use BERT directly rather than using BERT embeddings for prediction. Lastly, we give a statistical analysis of performance across diagnostic groups.

3 Data

3.1 Participants and tasks

The corpus used in the present study includes data previously described in Yang et al. (2021), which consists of conversations between neurotypical **conversational partners (CPs)** ($n = 11$) and adult **experimental participants (EPs)** 18-30 years of age with ASD ($n = 16$) and with typical development (TD, $n = 9$). ASD EPs met the diagnostic criteria for ASD on the Autism Diagnostic Observation Schedule (ADOS) (Lord et al., 2002). All EPs and CPs were monolingual speakers of American English, with no history of intellectual disability, language impairment, or hearing difficulties.

Each EP engaged with a CP in two collaborative discussion tasks. In the map task (Anderson et al., 1991), the EP and the CP were each given a map of the same area with slight differences in labels and obstacles. The EP was tasked with giving verbal directions to their CP to lead them to their marked position on the map. In the island task (Klippel et al., 1984), the EP and CP jointly viewed a set of labeled pictures of various items and discussed which items they would choose to bring to a deserted island. The conversations were transcribed by a team of three undergraduate research assistants yielding a total of 9433 utterances: 3091 produced by EPs with ASD; 1846 by EPs with TD; 2842 by CPs of ASD EPs; and 1654 by CPs of TD EPs.

The number of participants, while small, is quite typical for work that involves manual analysis of language in ASD, particularly for adults (Mawhood et al., 2000; Young et al., 2005; Parsons et al., 2017). We additionally note that our methods are applied to individual utterances rather than individual participants. The overall number of utterances in our dataset is on par with that of many widely used NLP datasets hand-labeled for similar features (Danescu-Niculescu-Mizil et al., 2013; Braley and Murray, 2018; Wang et al., 2019), and the statistical analyses we employ are appropriate for the size and distribution of our corpus.

3.2 Pragmatic feature annotation

The transcripts had previously been partially annotated by trained undergraduates with dual majors in Computer Science and either Linguistics or Psychology for three ordinal features: **politeness, uncertainty,** and **informativeness** (Yang et al., 2021). In our work, we completed this annotation process and introduced a new categorical feature, **dialog act**. For the ordinal features, two annotators, from a pool of four trained undergraduates double majoring in Computer Science and either Linguistics or Psychology, assigned to each utterance a rating on a three-point scale, achieving inter-annotator agreement as measured by Krippendorf's α (Artstein and Poesio, 2008) of 0.7, 0.76, and 0.83 for politeness, uncertainty, and informativeness, respectively. The final score for each utterance was the mean of the annotations for that feature. Example utterances with their respective scores for each of these features are shown in Table 1, and a brief overview of the annotation guidelines is provided in Appendix A. We refer the reader to Yang et al. (2021) for further details.

For the new categorical feature, dialog act, the annotator assigned to each utterance one from a

Task	Utterance	Politeness	Uncertainty	Informativeness	dialog act
Map	Do you have a pond on your map?	2	2	2	request for information
Map	Not at my area, no.	2	1	1	polar answer
Island	I don't care.	1	1	1	providing opinion
Island	You're on a deserted island.	2	1	2	providing information

Table 1: Examples of manual annotations for each task.

dialog act	Description
Request for Information	A request for factual information: *do you have a pond on your map?*
Providing Information	Answering a request for information or providing factual information unprompted: *I don't see that on my map*
Request for Opinion	A request for an opinion or suggestion: *what do you think?*
Providing Opinion	Answering a request for opinion or providing an opinion unprompted: *I think we should have the pot*
Polar Answer	Answering a polar question: *yeah*, *no*, *mm-hm*
Command	An utterance giving instruction or direction including indirect instruction: *and then could you go left?*
Filler	Filler words or phrases used to fill pauses in the conversation: *hm*, *anyways*, *okay so*
Backchannel	An utterance that indicates the participant is listening and understanding: *okay*, *mm-hm*, *gotcha*, *sounds good*
Nicety	Utterances which primarily serve to express apology, gratitude, or to otherwise maintain a pleasant conversation: *sorry, I didn't mean to cut you off*, *no you're good*
Comment	An utterance that contains extraneous commentary on the task, such as narrating or explaining the participant's actions: *So this is like Easy Street haha*, *How the heck do I say this?*
Interjection	Short exclamations or interjections such as *ah*, *oops*, *yay*, *wow*
Fragment	Short abandoned or interrupted utterances that are too incomplete to classify

Table 2: Descriptions of the dialog acts used in the annotations.

set of 12 possible dialog acts chosen specifically for the two tasks, which are illustrated in Table 2. Two annotators first independently annotated 60% of all utterances ($\alpha = 0.83$). Disagreements were then resolved via discussion. Finally, the remaining utterances were annotated by one of the two annotators and later reviewed by the other annotator. More details are found in Appendix A.

4 Method

We remind the reader that the goal of this work is not to identify features that distinguish typical development from ASD, as in prior work on applying NLP to language in autism (see Section 2). Instead, we aim to exploit known effective approaches to develop robust models for predicting linguistic features tied to social and pragmatic aspects of communication known to be impacted in ASD in order to support targeted communication interventions. Crucially, the models we develop must perform similarly for individuals with and without autism.

We begin with three baseline classification models: majority class, where every sample is assigned the most frequent label; stratified, where labels are assigned randomly according to their distribution in the training data; and random, where labels are assigned uniformly randomly from the set of possible labels. Previous work (Yang et al., 2021) on a subset of this dataset using a different cross-validation strategy has shown that these baselines yield competitive performance to bag-of-words models and existing statistical models (Meyers et al., 2019) trained on separate corpora of written texts for these features.

We compare these baselines to neural models trained with BERT (Devlin et al., 2019) using MaChamp (van der Goot et al., 2021) and its default parameters for classification tasks.[1] We did not explore statistical models here since neural models were shown to be substantially better in Yang et al. (2021). To learn how models might perform differently for participant groups whose linguistic features are potentially atypical, we measure model performance separately for each speaker group: ASD EPs, TD EPs, CPs when interacting with ASD EPs, and CPs when interacting with TD EPs. Model performance was indexed with raw accuracy and weighted F1 scores.

[1] Here we trained separate models for each feature; training models on the combined features yielded very weak results.

Features		ASD		TD		CP (with ASD)		CP (with TD)	
	Model	**F1**	**Acc.**	**F1**	**Acc.**	**F1**	**Acc.**	**F1**	**Acc.**
Politeness	*Majority*	0.75	0.83	0.75	0.83	0.80	0.86	0.77	0.84
	MaChamp	**0.85**	**0.86**	**0.89**	**0.89**	**0.89**	**0.90**	**0.90**	**0.90**
+ context	*MaChamp*	**0.85**	**0.86**	**0.89**	**0.89**	**0.89**	**0.90**	**0.90**	**0.90**
Uncertainty	*Majority*	0.48	0.63	0.48	0.63	0.49	0.63	0.57	0.70
	MaChamp	**0.74**	0.76	**0.75**	0.76	0.76	**0.77**	**0.78**	**0.79**
+ context	*MaChamp*	0.73	**0.77**	0.75	**0.77**	**0.77**	**0.77**	**0.78**	**0.79**
Information	*Majority*	0.37	0.53	0.32	0.48	0.37	0.53	0.42	0.58
	MaChamp	**0.80**	**0.81**	**0.81**	0.81	**0.80**	0.80	0.82	0.82
+ context	*MaChamp*	**0.80**	0.80	**0.81**	**0.82**	**0.80**	**0.82**	**0.83**	**0.83**
Dialog Act	*Stratified*	0.13	0.13	0.13	0.13	0.15	0.15	0.15	0.14
	MaChamp	0.73	**0.74**	0.77	0.77	0.77	**0.78**	0.78	0.79
+ context	*MaChamp*	**0.75**	0.73	**0.79**	**0.78**	**0.80**	**0.78**	**0.80**	**0.80**

Table 3: Comparison of F1 scores and accuracy across different groups of speakers given each feature, using **held-out transcript** cross-validation; + *context* indicates that training input included the preceding utterance; "CP" stands for conversational partner; we present only the result from the best baseline among the three.

Prior work has demonstrated that incorporating contextual information (in this case, previous utterances) is useful for predicting dialog act labels for both human-human (Liu et al., 2017) and human-chatbot (Khatri et al., 2018) interactions. To see whether a similar approach will be effective in our setting, we applied an evaluation scheme of **held-out transcript** [2], where we iteratively held out the data from one full transcript (i.e., the full conversation between one EP and one CP) as the test set and used the data from the remaining transcripts as the training set. This allowed us to incorporate contextual information by embedding the preceding utterance for feature prediction without enabling the models to learn individual speaker characteristics.

5 Results

Table 3 shows, unsurprisingly, that neural models, trained both with and without using the previous utterance as context, consistently outperform baselines for all four features, particularly the more challenging task of dialog act labeling.

To investigate whether adding contextual information was helpful in feature prediction, we compared for each feature the performance of the models with context to those without using logistic mixed-effects regression. The dependent variable was whether the feature value predicted by the model matched the manually assigned value; the fixed effect was the model (with or without the context of the previous utterance); and participant identity was included as a random intercept to control for repeated measurements of the same speaker. Though the results revealed no significant difference for the three ordinal features, including context appears to help improve model performance for dialog act ($\beta = 0.14$, $p < 0.001$). All further results presented will pertain to the models trained with prior contextual information.

We now turn to the question of whether there are differences in neural model performance on the utterances of ASD vs. TD experimental participants (EPs), as well as the utterances of conversational partners (CPs) of EPs with ASD vs. CPs of EPs with TD. Again, we applied logistic mixed-effects regression. The regression structure was similar to that described above, except that the group to which the speaker of the utterance belongs was used as the fixed effect (ASD EP vs. TD EP; or CP of ASD vs. CP of TD). Without using speaker identity as a random intercept, we found a significant effect of speaker group for EPs ($\beta = -0.31$, $z = -3.423$, $p < 0.001$) for politeness, which indicates that model predictions for politeness are more accurate for the TD group; we observed no such effect for CP groups. For dialog acts, there was also significant difference in accuracy between groups for EPs ($\beta = -0.38$, $z = -5.46$, $p < 0.001$), indicating that the models were more accurate for TD than for ASD utterances; a similar but weaker pattern was observed for CPs ($\beta = -0.22$, $z = -2.93$, $p < 0.01$). No significant differences were observed for uncertainty or informativeness. Running the same analysis using speaker identity as a random intercept, the significant differences for politeness and dialog acts are weaker but maintained for EPs (politeness: $\beta = -0.32$, $z = -2.00$,

[2] Held-out-speaker yielded comparable results.

Task	Utterance	Feature	Manual annotation	Model prediction
Map	Uh I don't mean to have you turn the map around it's just kind of how I think	Politeness	3	2
Island	Oh yeah that's right okay yeah that's that's smart	Politeness	3	2
Map	We're going to go past the duck	Dialog act	Command	Providing Information

Table 4: Examples of incorrectly classified ASD utterances.

$p < 0.05$; dialog acts: $\beta = -0.32$, $z = -2.15$, $p < 0.05$), suggesting that the observed differences in model accuracy may be driven by certain individuals, a finding that aligns with prior observations of heterogeneity in ASD language.

To qualitatively understand why our models might be more accurate for TD utterances, we inspected some of the incorrect predictions for ASD utterances. At times we observe similar unusual communication features in multiple ASD subjects, while other strategies appear to be idiosyncratic. Table 4 shows a few examples of misclassified ASD utterances. We observed many utterances like the first in this table, in which the ASD EP struggles to explain his reasoning, and many like the second, in which ASD EP evaluates the quality of his CP's prior statement. These strategies tend to be rare among TD EPs. In the third example, the EP uses "we" to politely give commands, a choice that, while easily recognized as a command-giving strategy by our annotators, was unique to that EP and was consistently misclassified.

6 Conclusions

Using a corpus of collaborative conversations between adults with and without ASD and their neurotypical conversational partners, we outline a framework for automatic identification of linguistic features associated with social communication. Although transformer-based models were able to achieve strong performance overall, when comparing results between diagnostic groups, we found that models fall short on the language of participants with ASD, especially in cases of politeness rating and dialog act labeling. This suggests that as powerful as transformer models are in capturing certain linguistic aspects of (written) data produced by (presumably) neurotypical speakers (see Linzen and Baroni (2021) for a review), they do not suffice in characterizing language in ASD, a finding that has broad implications for work applying these models to any potentially atypical language.

7 Ethical Considerations

All work described here was carried out with the approval of the Institutional Review Boards of all of the participating institutions. In accordance with our IRB protocol, we plan to release the full set of annotations of the corpus to interested researchers who can demonstrate completion of their institution's human subjects protection training curriculum.

References

Joel R. Adams, Alexandra C. Salem, Heather MacFarlane, Rosemary Ingham, Steven D. Bedrick, Eric Fombonne, Jill K. Dolata, Alison Presmanes Hill, and Jan van Santen. 2021. A pseudo-value approach to analyze the semantic similarity of the speech of children with and without autism spectrum disorder. *Frontiers in Psychology*, 12:3089.

Ahmad Aljanaideh, Eric Fosler-Lussier, and Marie-Catherine de Marneffe. 2020. Contextualized embeddings for enriching linguistic analyses on politeness. In *Proceedings of the 28th International Conference on Computational Linguistics*.

American Psychiatric Association. 2013. *Diagnostic and statistical manual of mental disorders: DSM-5*, 5th ed. edition. Autor, Washington, DC.

A Anderson, M Bader, E Bard, E Boyle, G. M Doherty, S Garrod, S Isard, J Kowtko, J McAllister, J Miller, C Sotillo, H. S Thompson, and R Weinert. 1991. The HCRC map task corpus. *Language and Speech*, 34(4):351–366.

Ron Artstein and Massimo Poesio. 2008. Inter-coder agreement for computational linguistics. *Computational Linguistics*, 34(4):555–596.

Susanna Baldwin, Debra Costley, and Anthony Warren. 2014. Employment activities and experiences of adults with high-functioning autism and asperger's disorder. *Journal of autism and developmental disorders*, 44(10):2440–2449.

McKenzie Braley and Gabriel Murray. 2018. The group affect and performance (gap) corpus. In *Proceedings of the ICMI 2018 Workshop on Group Interaction Frontiers in Technology (GIFT)*.

Harry Bunt, Volha Petukhova, Andrei Malchanau, Alex Fang, and Kars Wijnhoven. 2019. The DialogBank: Dialogues with interoperable annotations. *Language Resources and Evaluation*, 53(2):213–249.

Deborah L Christensen, Jon Baio, Kim Van Naarden Braun, Deborah Bilder, Jane Charles, John N Constantino, Julie Daniels, Maureen S Durkin, Robert T Fitzgerald, Margaret Kurzius-Spencer, et al. 2016. Prevalence and characteristics of autism spectrum disorder among children aged 8 years—Autism and developmental disabilities monitoring network. *MMWR Surveillance Summaries*, 65(3):1.

Olivia Conlon, Joanne Volden, Isabel M Smith, Eric Duku, Lonnie Zwaigenbaum, Charlotte Waddell, Peter Szatmari, Pat Mirenda, Tracy Vaillancourt, Teresa Bennett, et al. 2019. Gender differences in pragmatic communication in school-aged children with autism spectrum disorder (asd). *Journal of Autism and Developmental Disorders*, 49(5):1937–1948.

Cristian Danescu-Niculescu-Mizil, Moritz Sudhof, Dan Jurafsky, Jure Leskovec, and Christopher Potts. 2013. A computational approach to politeness with application to social factors. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 250–259.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186.

Richárd Farkas, Veronika Vincze, György Móra, János Csirik, and György Szarvas. 2010. The CoNLL-2010 shared task: Learning to detect hedges and their scope in natural language text. In *Proceedings of the Fourteenth Conference on Computational Natural Language Learning – Shared Task*, pages 1–12.

Shirley Hayati, Dongyeop Kang, and Lyle Ungar. 2021. Does bert learn as humans perceive? understanding linguistic styles through lexica. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 6323–6331.

Karen Hurlbutt and Lynne Chalmers. 2004. Employment and adults with asperger syndrome. *Focus on autism and other developmental disabilities*, 19(4):215–222.

Chandra Khatri, Rahul Goel, Behnam Hedayatnia, Angeliki Metanillou, Anushree Venkatesh, Raefer Gabriel, and Arindam Mandal. 2018. Contextual topic modeling for dialog systems. In *2018 IEEE Spoken Language Technology Workshop (SLT)*, pages 892–899.

Friederike Klippel, Penny Ur, and John H Klippel. 1984. *Keep talking: Communicative fluency activities for language teaching*. Cambridge University Press.

Shibamouli Lahiri. 2015. Squinky! a corpus of sentence-level formality, informativeness, and implicature. *arXiv preprint arXiv:1506.02306*.

Tal Linzen and Marco Baroni. 2021. Syntactic structure from deep learning. *Annual Review of Linguistics*, 7(1):195–212.

Yang Liu, Kun Han, Zhao Tan, and Yun Lei. 2017. Using context information for dialog act classification in DNN framework. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 2170–2178, Copenhagen, Denmark. Association for Computational Linguistics.

Catherine Lord, Michael Rutter, Pamela DiLavore, and Susan Risi. 2002. *Autism Diagnostic Observation Schedule (ADOS)*. Western Psychological Services.

Soile Loukusa, Eeva Leinonen, Katja Jussila, Marja-Leena Mattila, Nuala Ryder, Hanna Ebeling, and Irma Moilanen. 2007. Answering contextually demanding questions: Pragmatic errors produced by children with asperger syndrome or high-functioning autism. *Journal of Communication Disorders*, 40(5):357–381.

Lynn Mawhood, Patricia Howlin, and Michael Rutter. 2000. Autism and developmental receptive language disorder—a comparative follow-up in early adult life. i: Cognitive and language outcomes. *The Journal of Child Psychology and Psychiatry and Allied Disciplines*, 41(5):547–559.

Iain McCowan, Jean Carletta, Wessel Kraaij, Simone Ashby, Sebastien Bourban, Mike Flynn, Mael Guillemot, Thomas Hain, Jaroslav Kadlec, Vasilis Karaiskos, Melissa Kronenthal, Guillaume Lathoud, Mike Lincoln, Agnes Lisowska, Wilifried Post, Dennis Reidsma, and Pierre Wellner. 2005. The AMI meeting corpus. In *Proceedings of the 5th International Conference on Methods and Techniques in Behavioral Research*, volume 88, pages 137–140.

Benjamin S Meyers, Nuthan Munaiah, Andrew Meneely, and Emily Prud'hommeaux. 2019. Pragmatic characteristics of security conversations: an exploratory linguistic analysis. In *2019 IEEE/ACM 12th International Workshop on Cooperative and Human Aspects of Software Engineering (CHASE)*, pages 79–82. IEEE.

Julia Parish-Morris, Mark Liberman, Neville Ryant, Christopher Cieri, Leila Bateman, Emily Ferguson, and Robert Schultz. 2016. Exploring autism spectrum disorders using HLT. In *Proceedings of the Third Workshop on Computational Linguistics and Clinical Psychology*, pages 74–84.

Lauren Parsons, Reinie Cordier, Natalie Munro, Annette Joosten, and Renee Speyer. 2017. A systematic review of pragmatic language interventions for children with autism spectrum disorder. *PloS one*, 12(4):e0172242.

Rhea Paul, Stephanie Miles Orlovski, Hillary Chuba Marcinko, and Fred Volkmar. 2009. Conversational behaviors in youth with high-functioning asd and asperger syndrome. *Journal of autism and developmental disorders*, 39(1):115–125.

V Rose, David Trembath, Deb Keen, and Jessica Paynter. 2016. The proportion of minimally verbal children with autism spectrum disorder in a community-based early intervention programme. *Journal of Intellectual Disability Research*, 60(5):464–477.

Alexandra C Salem, Heather MacFarlane, Joel R Adams, Grace O Lawley, Jill K Dolata, Steven Bedrick, and Eric Fombonne. 2021. Evaluating atypical language in Autism using automated language measures. *Scientific Reports*, 11(1):10968.

Paul T Shattuck, Sarah Carter Narendorf, Benjamin Cooper, Paul R Sterzing, Mary Wagner, and Julie Lounds Taylor. 2012. Postsecondary education and employment among youth with an autism spectrum disorder. *Pediatrics*, 129(6):1042–1049.

Andreas Stolcke, Klaus Ries, Noah Coccaro, Elizabeth Shriberg, Rebecca Bates, Daniel Jurafsky, Paul Taylor, Rachel Martin, Carol Van Ess-Dykema, and Marie Meteer. 2000. Dialogue act modeling for automatic tagging and recognition of conversational speech. *Computational Linguistics*, 26(3):339–373.

Rob van der Goot, Ahmet Üstün, Alan Ramponi, Ibrahim Sharaf, and Barbara Plank. 2021. Massive choice, ample tasks (MaChAmp): A toolkit for multi-task learning in NLP. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: System Demonstrations*, pages 176–197.

Veronika Vincze. 2014. Uncertainty detection in Hungarian texts. In *Proceedings of COLING 2014, the 25th International Conference on Computational Linguistics: Technical Papers*, pages 1844–1853, Dublin, Ireland. Dublin City University and Association for Computational Linguistics.

Xuewei Wang, Weiyan Shi, Richard Kim, Yoojung Oh, Sijia Yang, Jingwen Zhang, and Zhou Yu. 2019. Persuasion for good: Towards a personalized persuasive dialogue system for social good. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 5635–5649.

Chien-Sheng Wu, Steven C.H. Hoi, Richard Socher, and Caiming Xiong. 2020. TOD-BERT: Pre-trained natural language understanding for task-oriented dialogue. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 917–929.

Ting-Wei Wu, Ruolin Su, and Biing-Hwang Juang. 2021. A context-aware hierarchical bert fusion network for multi-turn dialog act detection. *arXiv preprint arXiv:2109.01267*.

Christine Yang, Dora Liu, Qingyun Yang, Zoey Liu, and Emily Prud'hommeaux. 2021. Predicting pragmatic discourse features in the language of adults with autism spectrum disorder. In *Proceedings of the Association for Computational Linguistics Student Research Workshop (ACL-IJCNLP SRW)*, pages 284–291.

Edna Carter Young, Joshua J Diehl, Danielle Morris, Susan L Hyman, and Loisa Bennetto. 2005. The use of two language tests to identify pragmatic language problems in children with autism spectrum disorders. *Language, Speech, and Hearing Services in Schools*, 36:62–72.

Jinan Zeidan, Eric Fombonne, Julie Scorah, Alaa Ibrahim, Maureen S Durkin, Shekhar Saxena, Afiqah Yusuf, Andy Shih, and Mayada Elsabbagh. 2022. Global prevalence of autism: A systematic review update. *Autism Research*.

Piotr Żelasko, Raghavendra Pappagari, and Najim Dehak. 2021. What helps transformers recognize conversational structure? importance of context, punctuation, and labels in dialog act recognition. *Transactions of the Association for Computational Linguistics*, 9:1163–1179.

Amy Zhang, Bryan Culbertson, and Praveen Paritosh. 2017. Characterizing online discussion using coarse discourse sequences. In *Proceedings of ICWSM*.

A Appendix: Annotation guidelines

The **politeness** feature measures how well an utterance contributes to a polite and non-demanding dialogue, marked by agreeableness, positivity, and willingness to compromise. An utterance with a low politeness rating of 1 is given to utterances expressing negative comments or frustration (*you're wrong*, *ugh I don't know*) and utterances which use a more blunt way of phrasing commands (*go back*). A high politeness rating of 3 is given to utterances with niceties (e.g., *thanks*, *sorry*) or affirmative words (*wonderful*, *awesome*) and indirect phrasing of commands (*if you could make a turn*).

The **uncertainty** feature measures the amount of uncertainty expressed about the correctness or legitimacy of the utterance. An utterance with a low uncertainty rating of 1 shows no uncertainty at all, or contains only a few filler words. A rating of 2 indicates some hesitation. It is given to polar questions, either-or questions, short abandoned utterances, and utterances containing many filler words (*um*, *uh*) or hedge phrases (*I guess*). An utterance with high uncertainty (rating of 3) has open questions (*what do you see?*) or expresses explicit uncertainty or confusion (*I have no idea*).

The **informativeness** feature is defined as a measure for the overall information content and specificity of an utterance. Utterances provide no information, contain only polar answers (*yes*, *no*) or vague words with low specificity (*that thing*, *over there*) are given a low informativeness rating of 1. For the map task, a rating of 2 is given to utterances that contain words for general objects and do not specify a specific location on the map (*another path*), while a high informativeness rating of 3 is given to utterances which contain proper nouns or labels or descriptions that point specific location on the map (*near the red pandas*). In the island task, a rating of 2 is given to utterances which contain only a short phrase indicating the item (*I want the fishing pole*), and a rating of 3 is given to utterances which contain multiple item words or a longer explanation of the items (*the fishing pole is good for catching fish*).

Rather than using one of the many existing (and conflicting) sets of **dialog acts**, we devised a small set specific to this dataset and commonly observed characteristics of language in ASD. When assigning dialog act labels, the annotators were instructed to consider the surrounding utterances, in order to fully capture the function of the utterance in the larger conversation. A complete list of each dialog act with a description and examples for each one can be found in Table 2.

TERMinator: A system for scientific texts processing

Elena Bruches
A.P. Ershov Institute of Informatics Systems / Russia
bruches@bk.ru

Olga Tikhobaeva
Novosibirsk State University / Russia
o.tikhobaeva@g.nsu.ru

Yana Dementyeva
Novosibirsk State University / Russia
y.dementeva@g.nsu.ru

Tatiana Batura
A.P. Ershov Institute of Informatics Systems / Russia
tbatura@iis.nsk.su

Abstract

This paper is devoted to the extraction of entities and semantic relations between them from scientific texts, where we consider scientific terms as entities. In this paper, we present a dataset that includes annotations for two tasks and develop a system called TERMinator for the study of the influence of language models on term recognition and comparison of different approaches for relation extraction. Experiments show that language models pre-trained on the target language are not always show the best performance. Also adding some heuristic approaches may improve the overall quality of the particular task. The developed tool and the annotated corpus are publicly available at https://github.com/iis-research-team/terminator and may be useful for other researchers.

1 Introduction

Nowadays the amount of scientific publications is constantly growing. In this regard, the processing of scientific texts becomes especially relevant in relation to rapidly developing scientific fields, for example, computer science. Information extraction from scientific texts can be useful in domain-specific areas, for completion of knowledge graphs, in search and question-answering systems. This paper describes the study on entity recognition and relation extraction from scientific texts on computer science in Russian.

Currently, there are a number of datasets with annotations of entities and relations in a general domain (Doddington et al., 2004; Roth and Yih, 2004; Loukachevitch et al., 2021), biomedical domain (Kim et al., 2003; Gurulingappa et al., 2012; Li et al., 2016), or even multi-domains (Terryn et al., 2020). Still it is more difficult to find a publicly available dataset such as SciERC (Luan et al., 2018) for scientific fields other than biomedical, and especially in languages other than English.

Despite that the named entity recognition task is well studied, it still faces multiple challenges (Li et al., 2022a), namely, NER in domain-specific areas (Weber et al., 2021), NER from noisy data (Derczynski et al., 2017), code-mixed data (Fetahu et al., 2021), and detection of fine-grained and nested named entities (Kim and Kim, 2021; Ringland et al., 2019; Loukachevitch et al., 2021). This is caused by several issues: defining boundaries of compound terms; recognition of whether a lexical unit is part of a compound term; identification of a lexical unit as a term depending on the context and topic of the text in which this lexical unit is used etc. The relation extraction task also remains an unsolved problem, as it often requires the use of knowledge outside the text (for example, from knowledge bases or obtained in another way), and also due to the lack of a large amount of labeled data in different languages.

Selection of the most appropriate language model, which is able to provide the best quality for extraction of terms and relations from scientific texts is one of the relevant issues. Our experimental results not only show the usefulness of the proposed dataset, but also provide baselines for further research.

We make the following contributions:

- Provide a new dataset for both tasks (term recognition and relation extraction) for Russian scientific texts and develop a TERMinator tool for further research experiments.
- Study of influence of language models (without additional information, with heuristics and dictionaries) on term extraction.
- Compare three approaches for relation extraction (based on lexical patterns, classification with a CLS-vector, and combination of them).

Proceedings of the 29th International Conference on Computational Linguistics, pages 3420–3426
October 12–17, 2022.

2 Related Work

Entity recognition and relation extraction are the main tasks in information extraction. There are various approaches to solve them.

A traditional approach includes two stages: extracting n-grams which potentially may be terms, and then classification whether this n-gram is a term or not. In (Stanković et al., 2016) authors proposed to use dictionaries and morphological and syntax information. There are some works which use pre-defined ontologies for terms extraction (Ivanisenko et al., 2020). Another idea is to solve this task as a sequence labeling (Kucza et al., 2018). It allows to implement terms extraction in one stage and take into account syntax and semantic information from the context. For terms extraction the main challenge is to identify the exact term boundaries. In (Zhu and Li, 2022) authors proposed to use boundary smoothing as a regularization technique to overcome this problem.

Relation extraction is usually considered as a classification problem: for two given terms one needs to determine whether there is a semantic relation between them or not, and if they are related then to define its type. Some works describe the use of knowledge bases for relation extraction (Li et al., 2019; Baldini Soares et al., 2019). With the spread of transformers-based architectures, different pretrained language models are used to solve this task (Shi and Lin, 2019). Some researchers try to make use of incorporating external data sources in the model, for example the list of hand-written syntax patterns (Tao et al., 2019), information about sentence syntax tree (Ningthoujam et al., 2019; Nayak and Ng, 2019).

Recently, special attention has been paid to systems which solve terms recognition and relation extraction jointly. The authors propose an architecture that sequentially extracts entities and relationships between them, but in end-to-end settings (Eberts and Ulges, 2020; Ji et al., 2020; Huang et al., 2019; Miwa and Bansal, 2016). Another idea is to train a model with two outputs: one output is for term extraction, and the other is for relation extraction (Xue et al., 2019). However such approaches require quite a lot of annotated data to find hidden regularities.

3 Data Preparation

For the experiments we create an annotated dataset, which consists of abstracts of scientific papers on Information Technology in Russian.

As entities we consider nouns or noun groups, which are terms in this particular domain. Terms that we recognize as entities may consist of one or several tokens ("*software*", "*non-preemptive multitasking*"), abbreviations ("*CPU*", "*DLL*"), names of programming languages ("*Python*", "*C++*") and libraries ("*Pytorch*", "*SpaCy*"), hyphenated concepts containing Latin characters ("*n-gram*", "*web-service*"). Thereby we consider all possible chains of tokens that can be terms, except for those that are recursive or overlap. The entities are marked in the BIO format: each token is assigned a B-TERM tag if it is the initial tag for an entity, I-TERM if it is inside a term, or O if it is outside any entity.

Statistics for our dataset is presented in Table 1.

Unit	train	test
texts	136	80
tokens	12 809	11 157
terms	2 028	2 027
relations	356	620

Table 1: Dataset statistics

The list of relations is selected based on the following criteria. At first, a relation should be monosemantic (for example, we don't consider a semantic relation <*Entity-Destination*> because it has indirect meaning as well). Secondly, a relation should link scientific terms (for example, in relation <*Communication-Topic*> (an act of communication is about topic) the actants are not scientific terms). Thus, six semantic relations were selected. Types of relations in a corpus, their meanings and distribution by train and test sets are presented in Table 2.

Relation type	Meaning	train	test
CAUSE	x is the cause of y	19	19
ISA	x is y	96	93
PART_OF	x is part of y	23	87
SYNONYMS	x is the same as y	35	22
TOOL	x allows to create/etc. y	54	38
USAGE	x is used for/in y	126	330

Table 2: Types of relations

Here is a sample sentence where two terms and the relation between them are highlighted: *Pokazany preimushchestva primeneniya <e1>mul'timedijnyh tekhnologij</e1> v <e2>uchebnom processe</e2> i effektivnost' ih ispol'zovaniya vo vremya lekcij i seminarov*. (The advantages of using multimedia technologies in the educational process and the effectiveness of

their use during lectures and seminars are shown.). The relation between *e1* and *e2* is *USAGE*.

The dataset is available for other researchers[1].

4 Influence of language models on term extraction

4.1 Models without additional information

The experimental methodology is as follows: texts are fed as input; during the vectorization procedure, each text is divided into spans (in our case these are BPE tokens), each of which is assigned to a vector. Initially, the model learns to match tokens with labels by using the training data; then based on the revealed regularities the model makes predictions on the validation data. In this way, the metrics are fixed after each training epoch. The output is a set of labels associated by the system with each word from the input text.

We experimented with two models: multilingual BERT (Devlin et al., 2018) and BERT pre-trained on Russian texts (Kuratov and Arkhipov, 2019).

In the first stage of the experiment each pre-trained language model was fine-tuned on the train set described above. The optimal learning rate was chosen as 10^{-6}, and the batch size was 12. Such values prevent overfitting and obtain the best results. At this stage, the metrics show that on partial match both models give the same performance, while in exact match the model pre-trained on Russian-language texts gives better scores.

Then we extended a train set by adding 212 texts with a pseudo labeling method. We collected a dictionary (list of scientific terms) in a semi-automatic way:

- We extracted 2-, 3- and 4-grams from the scientific papers and manually filtered phrases, which potentially can be terms.
- We extracted all titles of articles from Wikipedia, which are included in a subgraph of category "Science", and then manually selected words and phrases, which potentially can be terms.

Thus we obtained a list of 17 252 terms, which we used for pseudo labeling. This technique is useful for rapidly changing areas of knowledge, when it is difficult to have dictionaries of terms and keep them up to date.

[1] https://github.com/iis-research-team/ruserrc-dataset

Due to the less detailed checking of the markup of the corpus, even with its comparatively large size, the metrics received with the models trained on it turned out to be lower than those of the same models trained on the manual annotated texts.

4.2 Models with heuristics

Experimentally we found out that in order to improve the quality of term extraction, we need to improve the definition of term boundaries. The task of defining terms' boundaries is more challenging than classifying a token as a "term" type. To improve the recognition of term boundaries, we apply some heuristics to handle such cases as removing a preposition as a first token of a term and some others. From the results shown in Table 3, it can be seen that the heuristics improve the quality of term extraction on the exact match.

4.3 Models with heuristics and dictionaries

It the third stage, the system extracts terms not only with the trained model, but also with the use of the dictionary described above. Heuristics are also applied.

Table 3 shows that the ruBERT model fine-tuned on the manually annotated train set extracts the terms in half of the cases, which is the best result of a exact match. On the pseudo-labeled train set, this combined method gives good results for the multilingual BERT both for exact and partial matches.

Both models solve the task of term recognition with a high quality, which can be seen from the good results on a partial match. The best result on an exact match pertains to the RuBERT model supported by a dictionary and heuristics. The results of the models on the exact match are expectedly lower than the results on a partial match, which again draws our attention to the task of defining terms' boundaries in texts.

The markup quality significantly affects the quality, as we can note from a comparison of the results of the first and second stages of the experiment (see Sections 4.1, 4.2). Fine-tuning on the manually-annotated training set gives better performance than fine-tuning on the pseudo-labeled training set, even though its size is larger than the size of the manually-annotated one. It is hard to compare results with other researchers as this is the first corpora for scientific texts in Russian as far as we know. But for the similar dataset in English

Train set	Model	F-M P	F-M R	F-M F1	P-M P	P-M R	P-M F1
Manually labeled	mBERT	0.40	0.46	0.43	**0.89**	**0.88**	**0.88**
	mBERT + h	0.49	0.45	0.47	0.86	0.86	0.86
	mBERT + d + h	0.47	0.50	0.48	0.86	0.87	0.87
	ruBERT	0.48	0.50	0.49	**0.89**	**0.88**	**0.88**
	ruBERT + h	**0.52**	0.47	0.49	**0.89**	**0.88**	**0.88**
	ruBERT + d + h	0.49	**0.51**	**0.50**	0.86	0.87	0.87
Pseudo labeled	mBERT	0.33	0.34	0.34	0.80	0.75	0.75
	mBERT + h	0.42	0.38	0.40	0.80	0.79	0.79
	mBERT + d + h	0.41	0.39	0.40	0.80	0.80	0.80
	ruBERT	0.32	0.32	0.33	0.78	0.76	0.75
	ruBERT + h	0.40	0.37	0.38	0.79	0.79	0.79
	ruBERT + h + d	0.38	0.39	0.38	0.79	0.74	0.76

Table 3: Metrics for full match (F-M) and partial match (P-M) terms extraction; *d* is for dictionary, *h* is for heuristics.

SciERC (Luan et al., 2018) the authors (Eberts and Ulges, 2020) reported F1-measure to be 0.70.

We observed that for the term extraction task the model mistakes in recognizing the exact term boundaries. Another problem arises when a term is divided by other words or signs in the sentence, for example, *"Morphological and syntax analysis"*. Probably, it should be solved at a post-processing stage.

5 Comparison of approaches for relation extraction

5.1 Using lexical patterns

At first, we applied an approach for relation identification based on lexical patterns. It consists in the following: for texts with tagged terms, we extract a context between each pair of terms, lemmatize it, and compare it with the lexical patterns. If they match, these two terms are connected by this relation. The length of the context should not exceed six words. This value was obtained experimentally by changing it and comparing the quality of the model. The obtained metrics for this approach are shown in Table 4. We used 111 patterns; examples of patters and their distribution by relations are presented in Table 5.

Relation type	Precision	Recall	F1
CAUSE	0.07	0.05	0.06
ISA	0.18	0.19	0.19
PART_OF	0.17	0.14	0.15
SYNONYMS	0.23	0.82	0.35
TOOL	0.06	0.08	0.07
USAGE	0.21	0.39	0.27
NO-RELATION	0.96	0.92	0.94
macro-average	0.27	0.37	0.29

Table 4: Metrics for lexical pattern's approach

5.2 Classification task with a CLS-vector

The second approach we used for the relation extraction is similar to R-BERT, and is used by other authors (Hosseini et al., 2022; Aldahdooh et al., 2021; Li et al., 2022b). We consider the task of relation extraction as a classification task (with 7 classes of relations: CAUSE, ISA, PART-OF, SYNONYMS, TOOL, USAGE, NO-RELATION). We take the vector of a special token CLS (it is considered as a vector of the input text) and the vector of two terms (connected by the relation). These three vectors are concatenated and the resulting vector is fed to the classifier (Wu and He, 2019). We tried to use three different language models: mBERT, ruBERT (Kuratov and Arkhipov, 2019) and cointegrated/rubert-tiny2.

In addition, some features of the training are noteworthy. Firstly, to train the models, we used the corpus of Russian texts without dividing it into training and validation sets, and the most appropriate number of epochs was selected experimentally, because there are very few examples for some relations, and therefore the validation set would be unrepresentative to determine the quality of the model. Secondly, to reduce the imbalance between the number of examples in the classes, we added only 50% of the randomly selected pairs of terms to the training set, excluding those with the distance between tokens more than 10.

Finally, we implemented an ensemble which includes both approaches: model and lexical patterns. All metrics are presented in Table 6. F1-score for all types of relations for combined approach are presented in Table 7. For comparison, the state-of-the-art result achieved on SciERC with the SpERT (using SciBERT) method is 50.84% for relation extraction (Eberts and Ulges, 2020). Our results may also be related to insufficient data, as Russian

Relation type	Examples of patterns (transliteration)	Examples of patterns (translation)	N
CAUSE	*Uvelichivsheesya potreblenie rafinirovannyh produktov pitaniya **yavlyaetsya prichinoj** mnozhestva takih zabolevanij.*	Increased consumption of refined foods **is the cause** of many diseases.	23
ISA	*Odnim iz samyh tochnyh i effektivnyh sposobov upravleniya zhestami **yavlyaetsya** upravlenie aktivnost'yu myshc.*	One of the most accurate and effective ways to control gestures **is** to control muscle activity.	13
PART_OF	*Process referirovaniya **sostoit iz** pyati osnovnyh shagov.*	The referencing process **consists of** five main steps.	5
SYNONYMS	*Stat'ya posvyashchena issledovaniyu vertikal'nogo poleta robota s mashushchim krylom, **takzhe nazyvaemogo** ornitopterom.*	The article is devoted to the study of the vertical flight of a robot with a flap wing, **also called** an ornithopter.	5
TOOL	*V stat'e predstavlen opyt razrabotki informacionnoj sistemy, **avtomatiziruyushchej** process raspredeleniya studentov po bazam praktik.*	The article presents the experience of development of the information system, **which automates** the process of distribution of students by the bases of practice.	29
USAGE	*V nastoyashchee vremya aktivno razvivaetsya napravlenie, svyazannoe s proektirovaniem nejronnyh setej **dlya ispol'zovaniya v** mobil'nyh ustrojstvah.*	Currently, the development of neural networks **for use on** mobile devices is growing rapidly.	36

Table 5: Examples of lexical patterns

is morphologically rich, which additionally complicates the work of the language model. Moreover, error analysis of relation extraction revealed that relations are often present implicitly between terms and one can recognize them if one only knows these particular terms. Quite many terms in IT texts are abstract (for example, *"program implementation"*, *"testing"*, etc.) and it can be difficult to define, whether there is any semantic relation between them or not. We plan to study this aspects in the future.

Model	Precision	Recall	F1
mBERT	0.26	0.32	0.26
ruBERT	0.26	0.34	0.27
rubert-tiny2	0.22	0.23	0.22
mBERT + p	0.26	**0.41**	**0.29**
ruBERT + p	**0.29**	0.35	0.28
rubert-tiny2 + p	0.29	0.24	0.24

Table 6: Metrics for different language models and combined approach; *p* is for patterns

Relation type	mBERT	ruBERT	ruBERT-tiny2
CAUSE	0.06	0.09	0.10
ISA	0.30	0.28	0.14
PART_OF	0.14	0.04	0.00
SYNONYMS	0.32	0.33	0.38
TOOL	0.04	0.07	0.00
USAGE	0.27	0.22	0.11
NO-RELATION	0.93	0.95	0.94
macro-average	**0.29**	0.28	0.24

Table 7: F1-score for all types of relations for combined approach

6 Conclusion

In this paper, we built a new dataset and study several methods for term recognition and relation extraction from computer science texts in Russian. We conducted several experiments with different pre-trained language models for both tasks. The results of our experiments show that language models pre-trained on the target language are not always show the best performance. Also adding some heuristic approaches may improve the overall quality for the particular task.

References

Jehad Aldahdooh, Ziaurrehman Tanoli, and Jing Tang. 2021. R-bert-cnn: Drug-target interactions extraction from biomedical literature. In *Proceedings of the BioCreative VII Challenge Evaluation Workshop*, pages 102–106. BioCreative VII challenge and workshop ; Conference date: 08-11-2021 Through 10-11-2021.

Livio Baldini Soares, Nicholas FitzGerald, Jeffrey Ling, and Tom Kwiatkowski. 2019. Matching the blanks: Distributional similarity for relation learning. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 2895–2905, Florence, Italy. Association for Computational Linguistics.

Leon Derczynski, Eric Nichols, Marieke van Erp, and Nut Limsopatham. 2017. Results of the wnut2017 shared task on novel and emerging entity recognition. In *Proceedings of the 3rd Workshop on Noisy User-generated Text*, pages 140–147.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. Bert: Pre-training of deep

bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, volume 15, pages 4171–4186. Association for Computational Linguistics Minneapolis, Minnesota.

George Doddington, Alexis Mitchell, Mark Przybocki, Lance Ramshaw, Stephanie Strassel, and Ralph Weischedel. 2004. The automatic content extraction (ACE) program – tasks, data, and evaluation. In *Proceedings of the Fourth International Conference on Language Resources and Evaluation (LREC'04)*, Lisbon, Portugal. European Language Resources Association (ELRA).

Markus Eberts and Adrian Ulges. 2020. Span-based joint entity and relation extraction with transformer pre-training. In *Proceedings of the 24th European Conference on Artificial Intelligence*, Santiago de Compostela, Spain.

Besnik Fetahu, Anjie Fang, Oleg Rokhlenko, and Shervin Malmasi. 2021. Gazetteer Enhanced Named Entity Recognition for Code-Mixed Web Queries. In *Proceedings of the 44th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 1677–1681.

Harsha Gurulingappa, Abdul Mateen Rajput, Angus Roberts, Juliane Fluck, Martin Hofmann-Apitius, and Luca Toldo. 2012. Development of a benchmark corpus to support the automatic extraction of drug-related adverse effects from medical case reports. *J. of Biomedical Informatics*, 45(5):885–892.

Pedram Hosseini, David A. Broniatowski, and Mona Diab. 2022. Knowledge-augmented language models for cause-effect relation classification. In *Proceedings of the First Workshop on Commonsense Representation and Reasoning (CSRR 2022)*, pages 43–48, Dublin, Ireland. Association for Computational Linguistics.

Weipeng Huang, Xingyi Cheng, Taifeng Wang, and Wei Chu. 2019. Bert-based multi-head selection for joint entity-relation extraction. In *Natural Language Processing and Chinese Computing: 8th CCF International Conference, NLPCC 2019, Dunhuang, China, October 9–14, 2019, Proceedings, Part II*, page 713–723, Berlin, Heidelberg. Springer-Verlag.

Timofey V. Ivanisenko, Olga V. Saik, Pavel S. Demenkov, Nikita V. Ivanisenko, Alexander N. Savostianov, and Vladimir A. Ivanisenko. 2020. Anddigest: a new web-based module of andsystem for the search of knowledge in the scientific literature. *BMC Bioinformatics*, 21(Suppl 11):228.

Bin Ji, Jie Yu, Shasha Li, Jun Ma, Qingbo Wu, Yusong Tan, and Huijun Liu. 2020. Span-based joint entity and relation extraction with attention-based span-specific and contextual semantic representations. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 88–99, Barcelona, Spain (Online). International Committee on Computational Linguistics.

Hongjin Kim and Harksoo Kim. 2021. Fine-grained named entity recognition using a multi-stacked feature fusion and dual-stacked output in korean. *Applied Sciences*, 11(22):10795.

Jin-Dong Kim, Tomoko Ohta, Yuka Tateisi, and Jun'ichi Tsujii. 2003. Genia corpus—a semantically annotated corpus for bio-textmining. *Bioinformatics*, 19(suppl_1):i180–i182.

Maren Kucza, Jan Niehues, Thomas Zenkel, Alex Waibel, and Sebastian Stuker. 2018. Term extraction via neural sequence labeling a comparative evaluation of strategies using recurrent neural networks. In *Proceedings of Interspeech 2018*, pages 2072–2076, Hyderabad, India.

Yuri Kuratov and Mikhail Arkhipov. 2019. Adaptation of deep bidirectional multilingual transformers for russian language. In *Computational Linguistics and Intellectual Technologies: Proceedings of the International Conference "Dialogue 2019"*, volume arXiv preprint arXiv:1905.07213, pages 333–339.

Jiao Li, Yueping Sun, Robin J. Johnson, Daniela Sciaky, Chih-Hsuan Wei, Robert Leaman, Allan Peter Davis, Carolyn J. Mattingly, Thomas C. Wiegers, and Zhiyong Lu. 2016. BioCreative V CDR task corpus: a resource for chemical disease relation extraction. *Database*, 2016. Baw068.

Jing Li, Aixin Sun, Jianglei Han, and Chenliang Li. 2022a. A survey on deep learning for named entity recognition. *IEEE Transactions on Knowledge and Data Engineering*, 34:50–70.

Peng-Hsuan Li, Ting-Fu Chen, Jheng-Ying Yu, Shang-Hung Shih, Chan-Hung Su, Yin-Hung Lin, Huai-Kuang Tsai, Hsueh-Fen Juan, Chien-Yu Chen, and Jia-Hsin Huang. 2022b. pubmedKB: an interactive web server for exploring biomedical entity relations in the biomedical literature. *Nucleic Acids Research*, 50(W1):W616–W622.

Pengfei Li, Kezhi Mao, Xuefeng Yang, and Qi Li. 2019. Improving relation extraction with knowledge-attention. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 229–239, Hong Kong, China. Association for Computational Linguistics.

Natalia Loukachevitch, Ekaterina Artemova, Tatiana Batura, Pavel Braslavski, Ilia Denisov, Vladimir Ivanov, Suresh Manandhar, Alexander Pugachev, and Elena Tutubalina. 2021. Nerel: A russian dataset with nested named entities, relations and events. In *Proceedings of Recent Advances in Natural Language Processing*, page 876–885.

Yi Luan, Luheng He, Mari Ostendorf, and Hannaneh Hajishirzi. 2018. Multi-task identification of entities, relations, and coreference for scientific knowledge graph construction. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 3219–3232, Brussels, Belgium. Association for Computational Linguistics.

Makoto Miwa and Mohit Bansal. 2016. End-to-end relation extraction using LSTMs on sequences and tree structures. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1105–1116, Berlin, Germany. Association for Computational Linguistics.

Tapas Nayak and Hwee Tou Ng. 2019. Effective attention modeling for neural relation extraction. In *Proceedings of the 23rd Conference on Computational Natural Language Learning (CoNLL)*, pages 603–612, Hong Kong, China. Association for Computational Linguistics.

Dhanachandra Ningthoujam, Shweta Yadav, Pushpak Bhattacharyya, and Asif Ekbal. 2019. Relation extraction between the clinical entities based on the shortest dependency path based lstm. *ArXiv*, arXiv:1903.09941. Version 1.

Nicky Ringland, Xiang Dai, Ben Hachey, Sarvnaz Karimi, Cecile Paris, and James R Curran. 2019. NNE: A dataset for nested named entity recognition in english newswire. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 5176–5181.

Dan Roth and Wen-tau Yih. 2004. A linear programming formulation for global inference in natural language tasks. In *Proceedings of the Eighth Conference on Computational Natural Language Learning (CoNLL-2004) at HLT-NAACL 2004*, pages 1–8, Boston, Massachusetts, USA. Association for Computational Linguistics.

Peng Shi and Jimmy Lin. 2019. Simple bert models for relation extraction and semantic role labeling. *ArXiv*, arXiv:1904.05255. Version 1.

Ranka Stanković, Cvetana Krstev, Ivan Obradović, Biljana Lazić, and Aleksandra Trtovac. 2016. Rule-based automatic multi-word term extraction and lemmatization. In *Proceedings of the Tenth International Conference on Language Resources and Evaluation (LREC'16)*, pages 507–514, Portorož, Slovenia. European Language Resources Association (ELRA).

Qiongxing Tao, Xiangfeng Luo, and Hao Wang. 2019. Enhancing relation extraction using syntactic indicators and sentential contexts. In *Proceedings of the International Conference on Tools with Artificial Intelligence (ICTAI)*, pages 574–580, Piscataway, NJ.

Ayla Rigouts Terryn, Veronique Hoste, and Els Lefever. 2020. In no uncertain terms: a dataset for monolingual and multilingual automatic term extraction from comparable corpora. *Language Resources and Evaluation*, 54:385–418.

Leon Weber, Mario Sänger, Jannes Münchmeyer, Maryam Habibi, Ulf Leser, and Alan Akbik. 2021. Hunflair: an easy-to-use tool for state-of-the-art biomedical named entity recognition. *Bioinformatics*, 37(17):2792–2794.

Shanchan Wu and Yifan He. 2019. Enriching pre-trained language model with entity information for relation classification. In *Proceedings of the 28th ACM International Conference on Information and Knowledge Management*, pages 2361–2364. ACM.

Kui Xue, Yangming Zhou, Zhiyuan Ma, Tong Ruan, Huanhuan Zhang, and Ping He. 2019. Fine-tuning bert for joint entity and relation extraction in chinese medical text. In *Proceedings of the 2019 IEEE International Conference on Bioinformatics and Biomedicine (BIBM)*, pages 892–897.

Enwei Zhu and Jinpeng Li. 2022. Boundary smoothing for named entity recognition. *ArXiv*, arXiv:2204.12031. Version 1.

LipKey: A Large-Scale News Dataset for Absent Keyphrases Generation and Abstractive Summarization

Fajri Koto[♠] **Timothy Baldwin**[♠♡] **Jey Han Lau**[♠]
[♠] School of Computing and Information Systems, The University of Melbourne
[♡] Department of Natural Language Processing, MBZUAI
fajri.koto91@gmail.com tb@ldwin.net jeyhan.lau@gmail.com

Abstract

Summaries, keyphrases, and titles are different ways of concisely capturing the content of a document. While most previous work has released the datasets of keyphrases and summarization separately, in this work, we introduce LipKey, the largest news corpus with human-written abstractive summaries, *absent* keyphrases, and titles. We jointly use the three elements via multi-task training and training as joint structured inputs, in the context of document summarization. We find that including absent keyphrases and titles as additional context to the source document improves transformer-based summarization models.[1]

1 Introduction

Key content of an article can be presented in different ways, including summaries, keyphrases, and titles. While most previous research has addressed each element individually (e.g. news summarization (Zhang et al., 2020a; Lewis et al., 2020; Koto et al., 2020a) and keyphrase generation in the scientific domain (Meng et al., 2017, 2021)), in this work we release a novel news dataset that consists of highly *absent* keyphrases (i.e. keyphrases which abstract over the content of the document), abstractive summaries, and titles to investigate the three elements in the context of single-document abstractive summarization.

Previous work has mainly utilized *present* keyphrases (i.e. keyphrases that are directly drawn from the source text) through unsupervised and supervised methods for summarization. For instance, traditional summarization models (Zhang et al., 2004; D'Avanzo and Magnini, 2005; Wan et al., 2007; Riedhammer et al., 2010; Qazvinian et al., 2010) and modern neural models (Müngen and Kaya, 2018; Nallapati et al., 2016; Liu et al., 2021) have been combined with the top-*k* frequent words, TF-IDF, and TextRank (Mihalcea and Tarau, 2004) to obtain keyphrases. Elsewhere, Gehrmann et al. (2018); Li et al. (2020) used words contained in both the summary and article as keyphrases to improve summarization.

This paper aims to study how *absent* keyphrases (i.e. keyphrases that do not match any words in the source text) can be incorporated into summarization systems. Compared to *present* keyphrases used in previous work, *absent* keyphrases potentially better complement abstractive summarization methods. Previous work has been hindered by the unavailability of a large annotated dataset with gold-standard summaries and keyphrases, thus opting for *present* keyphrase extraction (Qazvinian et al., 2010; Liu et al., 2021).

We additionally study the utility of titles in summarization. The underlying hypothesis is that titles and keyphrases are concise, complementary representations of an article, and provide relevant clues for summarization. While previous summarization datasets such as CNNDM (Hermann et al., 2015), NYT (Sandhaus, 2008), and XSUM (Narayan et al., 2018) do not include keyphrases and titles, we present a novel large-scale dataset containing both.

Following Koto et al. (2020a), we crawl Liputan6[2] — an Indonesian news portal — to obtain 105K news articles with titles, abstractive summaries, and *absent* keyphrases, all authored by journalists. Note that the dataset of Koto et al. (2020a) is based on the time period 2000–2010, at which point Liputan6 did not include keyphrases, while our dataset is based on the time period 2019–2021.[3] Furthermore, the fact that the dataset is in Indonesian contributes to language diversity in NLP (Joshi et al., 2020).

[1] Data and code used is available at: https://github.com/fajri91/LipKey

[2] https://www.liputan6.com

[3] Koto et al. (2020a) also do not release the titles. We performed online crawling using an RSS feed taken from a two year period to obtain the dataset.

Proceedings of the 29th International Conference on Computational Linguistics, pages 3427–3437
October 12–17, 2022.

Dataset / Lang	Size	Includes Summ?	#Key per doc (%)	AbsKey (%)
LipKey (ours) / id	105,537	Yes	4.5	51.2
DUC-2001 / en	308	Yes	8.1	3.7
PT-BN-KP / en	110	No	23.7	2.5
KPCrowd / en	500	No	48.9	13.5
KPTimes / en	289,923	No	5.0	54.8
WikiNews / fr	100	No	11.8	5.0

Table 1: LipKey and other keyphrase datasets in the news domain. "AbsKey" is the percentage of "absent" keyphrases, relative to the source article.

	|Vocab|	#Word		#Sentence	
		mean	std	mean	std
Article	346,564	436.5	277.7	22	17.5
Title	58,113	10.1	2.2	1	0
Summary	63,086	19	6.6	1.2	0.4
Keyphrases	33,976	8.9	4.6	4.5	1.9

Table 2: Per-article summary statistics for LipKey. For keyphrases, #sentence indicates #keyphrases.

Dataset	Size	% of novel *n*-gram			
		1	2	3	4
IndoSum	18,764	3.1	10.8	16.2	20.3
Liputan6	215,827	12.9	41.6	57.6	66.9
LipKey (summary)	105,537	7.5	25.2	35.1	40.9
LipKey (title)	105,537	26.8	65.4	84.5	92.7

Table 3: Abstractiveness of summaries (and titles) in Indosum, Liputan6, and LipKey, compared to the article.

To summarize our contributions: (1) we release LipKey, the largest news corpus containing human-written abstractive summaries and *absent* keyphrases, as well as being the first large-scale Indonesian keyphrase dataset; (2) through extensive experimentation, we benchmark multi-task training and structured input methods using keyphrases and titles for Indonesian text summarization over different pretrained language models. We find that incorporating keyphrases and titles as structured inputs performs better than multi-task training, and consistently improves summary quality.

2 Related Work

Most keyphrase datasets are in the domain of English scientific publications (Hulth, 2003; Krapivin et al., 2009; Kim et al., 2010; Meng et al., 2021). In Table 1, we compare our corpus, LipKey, with other keyphrase datasets in the news domain. Most datasets such as DUC-2001 (Wan and Xiao, 2008), PT-BN-KP (Marujo et al., 2012), KPCrowd (Marujo et al., 2011), and WikiNews (Bougouin et al., 2013) are small in size and consist of *highly present* keyphrases, with KPTimes (Gallina et al., 2019) being the only exception. DUC-2001 is the only dataset with both keyphrases and summaries, but has only 308 documents. In comparison, LipKey is a large news corpus that includes human-written summaries and *absent* keyphrases, as well as being the first large-scale Indonesian keyphrase dataset.

Incorporating keyphrases into summarization has been explored in other languages such as Chinese (Jiang et al., 2018; Mihalcea and Tarau, 2004), but using *present* keyphrases. This is the first work to combine the two tasks in the Indonesian language, with previous work separately tackling: (1) keyphrase extraction, over Twitter (Mahfuzh et al., 2019), consumer-health questions (Saputra et al., 2018), or scientific articles (Asrori et al., 2020; Trisna and Nurwidyantoro, 2020) with limited data;[4] or (2) document summarization in the news domain (Kurniawan and Louvan, 2018; Koto et al., 2020a).

3 Data Construction

Liputan6 is one of the largest Indonesian news portals, containing news on topics such as politics, health, business, and popular culture.[5] Koto et al. (2020a) found that Liputan6 summaries are highly abstractive, written by journalists, and suitable for Indonesian text summarization research. The summary and keyphrases are encapsulated in javascript variables `window.kmklabs.article` with the keys `shortDescription` and `keywords`, respectively.[6] In crawling Liputan6, we use article ID to ensure there is no redundancy in the dataset.

LipKey articles span the period December 2019 to March 2021, and each article is associated with a summary, title, and keyphrase(s).[7] In Table 2 and Table 3, we show the overall data statistics of LipKey, and compare it with previous Indonesian

[4]None of the datasets are publicly available.

[5]According to https://www.alexa.com, Liputan6 was ranked 16th and 308th in Indonesia and worldwide, respectively, in November 2021 in terms of popularity.

[6]In 2012, Liputan6 added keyphrases for articles. These keyphrases are also assigned manually by the journalist.

[7]Since the data is crawled between December 2019 and March 2021, models trained on this data will likely be biased towards events and issues occurring in this period.

Genre	Total (%)	Article		Title		Summary		Keyphrase		Word-level
		|Vocab|	μ(#word)	|Vocab|	μ(#word)	|Vocab|	μ(#word)	μ	ED	Entropy
general	33.5%	155,834	421.6	29,155	10.1	31,935	18.8	4.3	11.6	10.8
sport	12.3%	66,303	340.2	11,359	9.9	13,178	18.9	5.2	12.1	10.4
business	12.2%	82,899	524.7	13,377	9.6	15,591	18.3	4.1	10.8	10.5
local	9.5%	76,417	391.0	14,266	9.9	18,969	21.9	4.8	13.3	10.5
entertainment	8.6%	62,677	270.2	14,206	11.8	12,265	13.8	4.0	12.3	10.7
lifestyle	6.3%	75,566	434.7	11,456	10.2	13,177	17.2	4.8	11.6	10.7
international	5.5%	61,007	460.4	9,822	10.7	11,545	19.4	5.2	10.9	10.8
health	4.7%	44,916	380.8	8,314	10.3	9,468	19.2	5.7	11.8	10.5
technology	3.2%	39,338	418.4	6,195	9.3	8,503	20.2	4.3	11.1	10.3
automotive	2.7%	35,870	369.8	5,855	9.3	8,247	24.9	3.7	10.7	10.5
other	1.5%	37,512	603.7	4,131	10.5	5,278	18.0	4.7	12.4	10.3

Table 4: Data statistics based on news genre. ED is the average character-level Levenshtein edit distance, computed between two pairs of keyphrases, while word-level entropy (1-gram) is calculated based on the concatenation of article, title, and summary.

summarization datasets: IndoSum (Kurniawan and Louvan, 2018) and Liputan6 (Koto et al., 2020a). We observe that summaries in LipKey are more abstractive than IndoSum in terms of novel n-grams (computed relatively to the article). Interestingly, we found that LipKey's titles are even more abstractive than the summaries in all datasets. Note that the median summary length in LipKey is one sentence, and shorter than Liputan6 (Koto et al., 2020a) at two sentences, despite both datasets being crawled from the same news portal.

In constructing LipKey, we discard instances where: (1) one of the keyphrases has more than 6 words (which tends to be noise); (2) the article has less than 15 words; or (3) the summary has less than 5 words. This results in 105,537 instances that we split into 96,541/4,154/4,842 for train/development/test. In terms of the number of words, 33% and 43% of keyphrases consist of 1 and 2 words, respectively, with the remainder being 3–6 words (see Table 10 in the Appendix for some examples).

We also perform manual analysis over 100 random samples to examine why the keyphrases are absent (i.e. do not occur) in the source article. We find that 80% of keyphrases partially match the article or are word-order variants (see Table 5). Moreover, 15%, 12% and 14% of absent keyphrases are acronyms, synonyms, or morphological variants.

LipKey consists of diverse news genres as shown in Table 4. "General", "sport", "business", "local", and "entertainment" are the top-5 most common news genres found in the dataset, covering 75% of all articles. We observe that "entertainment" articles tend to be shorter than other genres,

Category	%	Examples (keyphrase vs. article)
Acronym	15	*manchester united* vs. *man utd*
Synonym	12	*kepribadian* "characteristic" vs. *sifat* "characteristic"
Morphology	14	*pemotor* "motorcyclist" vs. *motor* "motorcycle"
Different order or partial	80	*Virus Corona di Aceh* "coronavirus in Aceh" vs. *Virus Corona* "coronavirus"
Not a synonym but related	44	N/A
Found in title (not in article)	24	N/A

Table 5: Analysis of keyphrases from 100 random samples.

and "automotive" has the longest summaries but the fewest keyphrases on average. The average edit distance between two pairs of keyphrases in each genre is almost similar, ranging between 10–13, indicating the diversity of keyphrases in each article. Lastly, word-level entropy in each genre is also similar (around 10) indicating the similar low-level redundancy in each news genre.

4 Experiments

4.1 Set-Up

As described in Figure 1, we experiment in two settings: (1) multi-task training (title/keyphrases = output); and (2) training with structured input (title/keyphrases = input). For the first, we use summary s, title t, and keyphrase(s) k as the separate target texts, and perform multi-task training with article a as the source text (thus three tasks: summarization, keyphrase generation (KPG), and title generation). The total loss $\mathcal{L}$ for multi-task

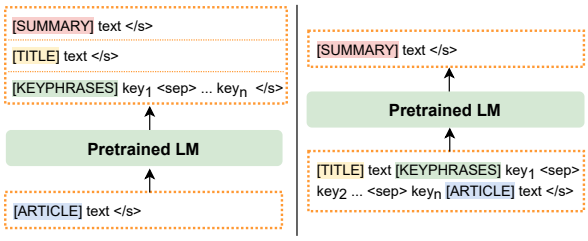

Figure 1: Experimental set-up. **Left**: multi-task training, **Right**: training with structured input.

training is defined as $\mathcal{L}_s + \mathcal{L}_t + \mathcal{L}_k$. For the second, the goal is to learn $P(s|t, k, a)$ that is realized by concatenating title t, keyphrases k, and article a to form the source text, and use summary s as the target text. To distinguish the four text types and structure the input, we introduce the special tokens of `[SUMMARY]`, `[TITLE]`, `[KEYPHRASES]`, and `[ARTICLE]` for all pretrained language models. In the case of multiple keyphrases, we use `<sep>` as a separator. The maximum number of tokens for the article is 512, and for the summary, title, and keyphrases it is 100.

We use the huggingface `transformers` library (Wolf et al., 2020) for our experiments with three pretrained language models: IndoBERT[8] (Koto et al., 2020b), mT5 (base)[9] (Xue et al., 2021), and mBART (large)[10] (Liu et al., 2020). For the monolingual IndoBERT, we follow Liu and Lapata (2019) in adding a raw transformer decoder (layers = 6, hidden size = 768, feed-forward = 2,048, and heads = 8) on top of IndoBERT, and train it on 4×V100 16GB GPUs for 200K steps. For the multilingual mT5 and mBART, we train them on 4×V100 32GB GPUs for 60 epochs (around 20K steps) with an initial learning rate of 1e-4 (Adam optimizer). We pick the best checkpoint based on ROUGE scores (Lin, 2004) on the development set (see the Appendix for more details of hyperparameters).

Additionally, we train keyphrase generation (KPG) models (Seq2Seq) with the same architectures and configurations as the summarization models. We compare the generated keyphrases with: (a) human-written keyphrases; and (b) keyphrases from RAKE, an unsupervised language-independent keyphrase extraction method (Rose et al., 2010).

[8] `indolem/indobert-base-uncased`
[9] `google/mt5-base`
[10] `facebook/mbart-large-50`

Model	R1	R2	RL	Foc.	Cov.
Lead-1	36.6	26.1	34.1	58.5	71.8
Oracle	69.2	58.9	66.9	76.4	87.2
IndoBERT (base) + raw decoder with 153M parameters					
summary	41.8	30.1	39.3	66.6	73.8
multi-task training					
summary, keyphrase	41.8	30.1	39.3	66.6	73.9
summary, title	42.9	31.1	40.4	67.0	74.4
summary, keyphrase, title	42.6	31.1	40.2	66.8	74.5
training with additional context					
+ keyphrase	43.4	31.8	41.0	67.2	74.6
+ title	43.2	31.5	40.7	67.4	74.3
+ keyphrase + title	43.7	31.9	41.2	67.4	74.6
mBART (large) with 600M parameters					
summary	43.1	31	40.5	67.6	73.9
multi-task training					
summary, keyphrase	43.6	31.3	41.0	68.1	74.0
summary, title	42.2	30.0	39.5	67.3	73.4
summary, keyphrase, title	43.5	31.6	40.8	67.8	74.1
training with additional context					
+ keyphrase	43.5	31.2	40.9	68.1	73.8
+ title	43.1	30.8	40.4	67.7	73.8
+ keyphrase + title	44.8	32.3	42.0	**68.8**	74.6
mT5 (base) with 580M parameters					
summary	45.2	33.7	42.7	67.5	76.2
multi-task training					
summary, keyphrase	44.7	33.2	42.1	66.9	76.3
summary, title	44.6	33.1	42.0	66.6	76.4
summary, keyphrase, title	43.7	32.0	41.0	66.1	76.0
training with additional context					
+ keyphrase	46.4	34.8	43.8	68.2	76.6
+ title	45.4	33.8	42.9	67.5	76.4
+ keyphrase + title	**46.7**	**35.1**	**44.2**	68.4	**76.9**

Table 6: Summarization results on `LipKey`. "Foc" and "Cov" are Focus and Coverage, respectively, of FFCI. Entries in bold and underline refer to the best overall score and the best score for each model, respectively. "Oracle" is obtained by greedily selecting the subset of sentences in the article that maximizes the ROUGE score based on the reference summary.

For evaluating the summarization models, we use F1 of ROUGE scores (R1, R2, and RL), and Focus and Coverage from the FFCI framework (Koto et al., 2022), computed based on Precision and Recall of BERTSCORE (Zhang et al., 2020b) using mBERT uncased.[11] For evaluating KPG, we use macro-averaged $F_1@5$, $F_1@\mathcal{O}$, and $F_1@\mathcal{M}$, following Meng et al. (2021), and additionally report R1, Focus, and Coverage. Detailed definitions of the metrics are provided in the Appendix.

[11] For details of BERT layer selection, see Koto et al. (2021).

Model	R1	Foc.	Cov.	$F_1@5$	$F_1@\mathcal{O}$	$F_1@\mathcal{M}$
RAKE	7.8	40.0	58.7	1.0	1.0	1.0
IndoBERT	58.8	74.2	79.4	45.5	45.2	46.5
mT5 (base)	62.0	75.7	81.7	53.3	52.9	54.5
mBART (large)	**63.4**	**76.4**	**81.9**	**54.5**	**54.4**	**56.0**

Table 7: Keyphrase generation results on LipKey.

4.2 Results

In Table 6, we show the full experimental results on the test set. First, we observe that vanilla models (trained only using the article) substantially outperform Lead-1 for all models.[12] We find that the vanilla model of mT5 performs better than IndoBERT and mBART, with an improvement of +3.4 and +2.1 R1, respectively.

Training with additional context as structured input consistently improves over multi-task training, with the best results generally being obtained with both keyphrases and title, and mT5 being the best model. When incorporating each element separately, keyphrases are generally better than titles, improving over the vanilla model, with IndoBERT (with multi-task training) being the notable exception. We also observe that mBART (large) and mT5 (base) are similar in parameter size (600M), but mT5 is substantially better. The FFCI framework shows that both models have similar Focus (= precision), but mT5 has higher Coverage (= recall).

Next, in Table 7, we present results for keyphrase generation on the LipKey test set, and observe that mBART (large) achieves the best performance across all metrics. Interestingly, RAKE performs very poorly,[13] in part emphasizing the limitations of the extractive RAKE method (vs. the highly *absent* keyphrases in LipKey).

Lastly, to benchmark the effect of different keyphrases in summarization we perform an ablation study over the best summarization model, mT5, using keyphrases sourced through three different methods: (1) RAKE, (2) Seq2Seq, and (3) human-assigned. We use RAKE for this study because there is no suitable keyphrase dataset for training neural models to extract *present* keyphrases in Indonesian. As seen in Table 8, adding RAKE keyphrases hurts summarization results, but when using Seq2Seq keyphrases (generated by mBART), the performance consistently improves across all metrics, close to the performance of human-assigned keyphrases. Considering this finding, it would be interesting to explore the transferability of keyphrase generation models to other languages, to see if it can be reproduced.

Model	R1	R2	RL	Foc.	Cov.
Vanilla	45.2	33.7	42.7	67.5	76.2
+ keyphrases (RAKE)	44.8	33.3	42.3	66.5	75.6
+ keyphrases (Seq2Seq*)	46.0	34.4	43.5	68.1	76.4
+ keyphrases (Human)	**46.4**	**34.8**	**43.8**	**68.2**	**76.6**
Vanilla + title	45.4	33.8	42.9	67.5	76.4
+ keyphrases (RAKE)	43.7	32.1	41.2	67.3	76.1
+ keyphrases (Seq2Seq*)	45.9	34.1	43.3	67.9	76.4
+ keyphrases (Human)	**46.7**	**35.1**	**44.2**	**68.4**	**76.9**

Table 8: Ablation study of mT5 (base) over different keyphrases on test set. * denotes using mBART (large).

5 Conclusion

In this paper, we release LipKey, the largest news corpus with human-written keyphrases, summaries and titles which is also the first-large scale Indonesian keyphrase dataset. We experimented with incorporating keyphrases (and titles) into summarization training via multi-task training or as structured inputs, and found that the latter works better. In this preliminary results, we show that *absent* keyphrases benefit summarization systems more than *present* keyphrases extracted by RAKE.

6 Ethical Considerations

According to Indonesian Copyright Law number 28 year 2014 article 44, the use, retrieval, reproduction, and/or change of works and/or related rights products in whole or substantial part are not regarded as a copyright infringement if the source is mentioned or cited in full for the purpose of education and research.[14]

Acknowledgements

We are grateful to the anonymous reviewers for their helpful feedback and suggestions. The first author is supported by the Australia Awards Scholarship (AAS), funded by the Department of Foreign Affairs and Trade (DFAT), Australia. This research was undertaken using the LIEF HPC-GPGPU Facility hosted at The University of Melbourne. This

[12] We choose Lead-1 because the average #sentence of the summary is 1.2 in Table 2.

[13] For each article, we pick the top-5 keyphrases based on RAKE scoring.

[14] https://wipolex-res.wipo.int/edocs/lexdocs/laws/en/id/id064en.pdf

facility was established with the assistance of LIEF Grant LE170100200.

References

Riris Bayu Asrori, Robert Setyawan, and Muljono Muljono. 2020. Performance analysis graph-based keyphrase extraction in Indonesia scientific paper. In *2020 International Seminar on Application for Technology of Information and Communication (iSemantic)*.

Adrien Bougouin, Florian Boudin, and Béatrice Daille. 2013. TopicRank: Graph-based topic ranking for keyphrase extraction. In *Proceedings of the Sixth International Joint Conference on Natural Language Processing*, pages 543–551, Nagoya, Japan. Asian Federation of Natural Language Processing.

Ernesto D'Avanzo and Bernado Magnini. 2005. A keyphrase-based approach to summarization: the Lake system at DUC-2005. In *Proceedings of DUC*.

Ygor Gallina, Florian Boudin, and Beatrice Daille. 2019. KPTimes: A large-scale dataset for keyphrase generation on news documents. In *Proceedings of the 12th International Conference on Natural Language Generation*, pages 130–135, Tokyo, Japan. Association for Computational Linguistics.

Sebastian Gehrmann, Yuntian Deng, and Alexander Rush. 2018. Bottom-up abstractive summarization. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 4098–4109, Brussels, Belgium. Association for Computational Linguistics.

Karl Moritz Hermann, Tomáš Kočiský, Edward Grefenstette, Lasse Espeholt, Will Kay, Mustafa Suleyman, and Phil Blunsom. 2015. Teaching machines to read and comprehend. In *NeurIPS'15 Proceedings of the 28th International Conference on Neural Information Processing Systems - Volume 1*, volume 28, pages 1693–1701.

Anette Hulth. 2003. Improved automatic keyword extraction given more linguistic knowledge. In *Proceedings of the 2003 Conference on Empirical Methods in Natural Language Processing*, pages 216–223.

Xiaoping Jiang, Po Hu, Liwei Hou, and Xia Wang. 2018. Improving pointer-generator network with keywords information for Chinese abstractive summarization. In *CCF International Conference on Natural Language Processing and Chinese Computing*, pages 464–474.

Pratik Joshi, Sebastin Santy, Amar Budhiraja, Kalika Bali, and Monojit Choudhury. 2020. The state and fate of linguistic diversity and inclusion in the NLP world. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 6282–6293, Online. Association for Computational Linguistics.

Su Nam Kim, Olena Medelyan, Min-Yen Kan, and Timothy Baldwin. 2010. Semeval-2010 task 5 : Automatic keyphrase extraction from scientific articles. In *Proceedings of the 5th International Workshop on Semantic Evaluation*, pages 21–26.

Fajri Koto, Timothy Baldwin, and Jey Han Lau. 2022. FFCI: A framework for interpretable automatic evaluation of summarization. *Journal of Artificial Intelligence Research*, 73:1553–1607.

Fajri Koto, Jey Han Lau, and Timothy Baldwin. 2020a. Liputan6: A large-scale Indonesian dataset for text summarization. In *Proceedings of the 1st Conference of the Asia-Pacific Chapter of the Association for Computational Linguistics and the 10th International Joint Conference on Natural Language Processing*, pages 598–608, Suzhou, China. Association for Computational Linguistics.

Fajri Koto, Jey Han Lau, and Timothy Baldwin. 2021. Evaluating the efficacy of summarization evaluation across languages. In *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*, pages 801–812, Online. Association for Computational Linguistics.

Fajri Koto, Afshin Rahimi, Jey Han Lau, and Timothy Baldwin. 2020b. IndoLEM and IndoBERT: A benchmark dataset and pre-trained language model for Indonesian NLP. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 757–770, Barcelona, Spain (Online). International Committee on Computational Linguistics.

Mikalai Krapivin, Aliaksandr Autaeu, and Maurizio Marchese. 2009. Large dataset for keyphrases extraction. Technical Report DISI-09-055, University of Trento.

Kemal Kurniawan and Samuel Louvan. 2018. Indosum: A new benchmark dataset for Indonesian text summarization. In *Proceedings of the 2018 International Conference on Asian Language Processing (IALP)*, pages 215–220.

Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Veselin Stoyanov, and Luke Zettlemoyer. 2020. BART: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7871–7880, Online. Association for Computational Linguistics.

Haoran Li, Junnan Zhu, Jiajun Zhang, Chengqing Zong, and Xiaodong He. 2020. Keywords-guided abstractive sentence summarization. In *Proceedings of the AAAI Conference on Artificial Intelligence*, pages 8196–8203.

Chin-Yew Lin. 2004. ROUGE: A package for automatic evaluation of summaries. In *Text Summarization Branches Out*, pages 74–81, Barcelona, Spain. Association for Computational Linguistics.

Yang Liu and Mirella Lapata. 2019. Text summarization with pretrained encoders. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 3730–3740, Hong Kong, China. Association for Computational Linguistics.

Yinhan Liu, Jiatao Gu, Naman Goyal, Xian Li, Sergey Edunov, Marjan Ghazvininejad, Mike Lewis, and Luke Zettlemoyer. 2020. Multilingual denoising pre-training for neural machine translation. *Transactions of the Association for Computational Linguistics*, 8:726–742.

Yizhu Liu, Qi Jia, and Kenny Zhu. 2021. Keyword-aware abstractive summarization by extracting set-level intermediate summaries. In *Proceedings of the Web Conference 2021*, pages 3042–3054.

Miftahul Mahfuzh, Sidik Soleman, and Ayu Purwarianti. 2019. Improving joint layer RNN based keyphrase extraction by using syntactical features. In *2019 International Conference of Advanced Informatics: Concepts, Theory and Applications (ICAICTA)*.

Luís Marujo, Anatole Gershman, Jaime Carbonell, Robert Frederking, and João P. Neto. 2012. Supervised topical key phrase extraction of news stories using crowdsourcing, light filtering and co-reference normalization. In *Proceedings of the Eighth International Conference on Language Resources and Evaluation (LREC'12)*, pages 399–403, Istanbul, Turkey. European Language Resources Association (ELRA).

Luís Marujo, Márcio Viveiros, and João Paulo da Silva Neto. 2011. Keyphrase cloud generation of broadcast news. In *Interspeech 2011*, pages 2393–2396.

Rui Meng, Xingdi Yuan, Tong Wang, Sanqiang Zhao, Adam Trischler, and Daqing He. 2021. An empirical study on neural keyphrase generation. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 4985–5007, Online. Association for Computational Linguistics.

Rui Meng, Sanqiang Zhao, Shuguang Han, Daqing He, Peter Brusilovsky, and Yu Chi. 2017. Deep keyphrase generation. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 582–592, Vancouver, Canada. Association for Computational Linguistics.

Rada Mihalcea and Paul Tarau. 2004. TextRank: Bringing order into text. In *Proceedings of the 2004 Conference on Empirical Methods in Natural Language Processing*, pages 404–411, Barcelona, Spain. Association for Computational Linguistics.

Ahmet Anil Müngen and Mehmet Kaya. 2018. Extracting abstract and keywords from context for academic articles. *Social Network Analysis and Mining*, 8(1):45.

Ramesh Nallapati, Bowen Zhou, Cicero Nogueira dos santos, Caglar Gulcehre, and Bing Xiang. 2016. Abstractive text summarization using sequence-to-sequence RNNs and beyond. In *Proceedings of The 20th SIGNLL Conference on Computational Natural Language Learning*, pages 280–290.

Shashi Narayan, Shay B. Cohen, and Mirella Lapata. 2018. Don't give me the details, just the summary! topic-aware convolutional neural networks for extreme summarization. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 1797–1807, Brussels, Belgium. Association for Computational Linguistics.

Vahed Qazvinian, Dragomir R. Radev, and Arzucan Özgür. 2010. Citation summarization through keyphrase extraction. In *Proceedings of the 23rd International Conference on Computational Linguistics (Coling 2010)*, pages 895–903, Beijing, China. Coling 2010 Organizing Committee.

Korbinian Riedhammer, Benoit Favre, and Dilek Hakkani-Tür. 2010. Long story short — global unsupervised models for keyphrase based meeting summarization. *Speech Communication*, 52(10):801–815.

Stuart Rose, Dave Engel, Nick Cramer, and Wendy Cowley. 2010. Automatic keyword extraction from individual documents. *Text Mining: Applications and Theory*, pages 1–20.

Evan Sandhaus. 2008. The New York Times annotated corpus. *Linguistic Data Consortium, Philadephia*, 6(12):e26752.

Ilham Fathy Saputra, Rahmad Mahendra, and Alfan Farizki Wicaksono. 2018. Keyphrases extraction from user-generated contents in healthcare domain using long short-term memory networks. In *Proceedings of the BioNLP 2018 workshop*, pages 28–34, Melbourne, Australia. Association for Computational Linguistics.

I Nyoman Prayana Trisna and Arif Nurwidyantoro. 2020. Single document keywords extraction in Bahasa Indonesia using phrase chunking. *TELKOMNIKA Telecommunication Computing Electronics and Control*, 18(4):1917–1925.

Xiaojun Wan and Jianguo Xiao. 2008. Single document keyphrase extraction using neighborhood knowledge. In *AAAI'08 Proceedings of the 23rd national conference on Artificial intelligence - Volume 2*, pages 855–860.

Xiaojun Wan, Jianwu Yang, and Jianguo Xiao. 2007. Towards an iterative reinforcement approach for simultaneous document summarization and keyword

extraction. In *Proceedings of the 45th Annual Meeting of the Association of Computational Linguistics*, pages 552–559, Prague, Czech Republic. Association for Computational Linguistics.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Remi Louf, Morgan Funtowicz, Joe Davison, Sam Shleifer, Patrick von Platen, Clara Ma, Yacine Jernite, Julien Plu, Canwen Xu, Teven Le Scao, Sylvain Gugger, Mariama Drame, Quentin Lhoest, and Alexander Rush. 2020. Transformers: State-of-the-art natural language processing. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 38–45, Online. Association for Computational Linguistics.

Linting Xue, Noah Constant, Adam Roberts, Mihir Kale, Rami Al-Rfou, Aditya Siddhant, Aditya Barua, and Colin Raffel. 2021. mT5: A massively multilingual pre-trained text-to-text transformer. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 483–498, Online. Association for Computational Linguistics.

Jingqing Zhang, Yao Zhao, Mohammad Saleh, and Peter Liu. 2020a. PEGASUS: Pre-training with extracted gap-sentences for abstractive summarization. In *ICML 2020: 37th International Conference on Machine Learning*, volume 1, pages 11328–11339.

Tianyi Zhang, Varsha Kishore, Felix Wu, Kilian Q. Weinberger, and Yoav Artzi. 2020b. BERTScore: Evaluating text generation with BERT. In *ICLR 2020: Eighth International Conference on Learning Representations*.

Yongzheng Zhang, Evangelos Milios, and Nur Zincir-Heywood. 2004. A comparison of keyword-and keyterm-based methods for automatic web site summarization. In *AAAI04 Workshop on Adaptive Text Extraction and Mining*, pages 15–20.

A Overview and Analysis of Keyphrases

Text	% Avg. of present keyphrases	
	Keyphrases level	Word level
Article	66.8	86.9
Summary	39.8	55.3
Title	48.2	62.9

Table 9: Proportion of keyphrases which match article, summary, and title

#Word	Freq	Example
1	167,203	*COVID-19; Netflix*
2	214,672	*New Normal; Diego Michels*
3	84,162	*Klasemen Liga Inggris* "Premier League"
4	22,780	*Ganjil Genap Kota Bogor* "odd-even policy in Bogor"
5	6,366	*Kru KM Lambelu Positif Covid-19* "KM Lambelu Crew Positive Covid-19"
6	1,623	*Cara Menulis Daftar Pustaka dari Internet* "ways to write a bibliography from Internet"

Table 10: Frequency of keyphrases in `LipKey` based on #Word.

B Training configurations

Summarization and keyphrase generation use the same models and architecture. For IndoBERT, we follow the Liu and Lapata (2019) architecture by adding a raw transformer decoder (layers = 6, hidden size = 768, feed-forward = 2,048, and heads = 8) on top of IndoBERT, and train it on 4×V100 16GB GPUs for 200K steps with the Adam optimizer and learning rate $lr = 2e^{-3} \cdot \min(\text{step}^{-0.5}, \text{step} \cdot 20{,}000^{-1.5})$ and $0.1 \cdot \min(\text{step}^{-0.5}, \text{step} \cdot 10{,}000^{-1.5})$ for IndoBERT and the transformer decoder, respectively. We use a warmup of 20,000, a dropout of 0.2, a batch size total of 200 (10 x 4 GPUs x gradient accumulation of 5), and save checkpoints every 10,000 steps. We compute ROUGE scores (R1) to pick the best checkpoint based on the development set.

For mT5 and mBART, we train them on 4×V100 32GB GPUs for 60 epochs (around 20K steps) with an initial learning rate of 1e-4 (Adam optimizer). We use a total batch size of 400 (10 x 4 GPUs x gradient accumulation of 10), a warmup of 10% of total steps, and save checkpoints every 1,000 steps. We also compute ROUGE scores (R1) to pick the best checkpoint based on the development set.

C Evaluation Metrics

For summarization, we use ROUGE scores (Lin, 2004), and Focus and Coverage from the FFCI framework (Koto et al., 2022). Following Koto et al. (2021), for non-English text, focus and coverage are computed based on the precision and recall of BERTSCORE (Zhang et al., 2020b) using mBERT uncased at layers 12 and 6, respectively. For Y and Y' as the reference and system summary,

BERTSCORE is computed as follows:

$$\mathcal{P}_{\text{BERT}} = \frac{1}{|Y'|} \sum_{t_i \in Y'} \max_{s_j \in Y} t_i^T s_j$$

$$\mathcal{R}_{\text{BERT}} = \frac{1}{|Y|} \sum_{s_j \in Y} \max_{t_i \in Y'} t_i^T s_j$$

$$\mathcal{F}_{\text{BERT}} = 2\frac{\mathcal{P}_{\text{BERT}} \cdot \mathcal{R}_{\text{BERT}}}{\mathcal{P}_{\text{BERT}} + \mathcal{R}_{\text{BERT}}}$$

where s_j and t_i are token embeddings of Y and Y'.

For evaluating the keyphrase generation model, we use macro-averaged $F_1@5$, $F_1@\mathcal{O}$, and $F_1@\mathcal{M}$, following Meng et al. (2021). Given gold-standard keyphrases $\mathcal{Y}$ and the prediction $\hat{\mathcal{Y}} = \{y'_1, .., y'_m\}$, we truncate the prediction to $\hat{\mathcal{Y}} = \{y'_1, .., y'_{min(k,m)}\}$ when only the top *k* predictions are used for evaluation. Precision, Recall, and F_1 are consequently conditioned on *k*, and computed as follows:

$$\text{P}@k = \frac{|\hat{\mathcal{Y}}_{:k} \cap \mathcal{Y}|}{|\hat{\mathcal{Y}}_{:k}|}$$

$$\text{R}@k = \frac{|\hat{\mathcal{Y}}_{:k} \cap \mathcal{Y}|}{|\mathcal{Y}|}$$

$$F_1@k = \frac{2 * \text{P}@k * \text{R}@k}{\text{P}@k + \text{R}@k}$$

Thus $F_1@5$ is $F_1@k$ when $k = 5$, $F_1@\mathcal{O}$ is $F_1@k$ when k is the number of oracle (ground truth) keyphrases, and $F_1@\mathcal{M}$ is when $k = |\hat{\mathcal{Y}}|$.

Indonesian	English (translation)
Title: Ada Warga Positif Corona di KRL, Ini Kata Kemenhub **Gold Keyphrases:** krl, COVID-19, Corona **Article:** Liputan6 . com , Jakarta Kementerian Perhubungan (Kemenhub) memastikan pelaksanaan protokol di Kereta Rangkaian Listrik (KRL) Jabodetabek terus berjalan . Pernyataan ini dikeluarkan pasca adanya 3 penumpang asal Bogor yang dinyatakan positif corona pasca dilakukan test swab . Juru Bicara Kementerian Perhubungan Adita Irawati menyatakan , pihaknya telah mengeluarkan Permenub Nomor 18/2020 yang telah mengatur operasional moda transportasi di masa pandemi . Khususnya pula di daerah yang telah menjalankan Pembatasan Sosial Berskala Besar (PSBB) seperti di Jabodetabek . Perlu dipahami bahwa penularan Covid-19 bisa terjadi dimana saja , tidak hanya di di KRL , " kata Adita , Selasa (5/5/2020) . Adita mengatakan , Permenhub 18/2020 secara tegas telah menyatakan adanya beberapa syarat yang wajib dipenuhi penumpang moda transportasi publik seperti KRL . Pertama , penumpang wajib menggunakan masker . Kedua , sambungnya , petugas mengecek suhu tubuh penumpang. [254 words are abbreviated from here]	**Title:** Corona positive passengers are detected on the KRL, this is what the Ministry of Transportation says **Gold Keyphrases:** krl, COVID-19, Corona **Article:** Liputan6 . com , Jakarta The Ministry of Transportation (Kemenhub) ensures that the implementation of the protocol on the Jabodetabek Electric Circuit Train (KRL) continues. This statement was issued after 3 passengers from Bogor were tested positive for corona after a swab test was carried out. Spokesman for the Ministry of Transportation, Adita Irawati, stated that her party had issued Permenub No. 18/2020 which had regulated the operation of transportation modes during the pandemic. This is particularly the case in areas that have implemented Large-Scale Social Restrictions (PSBB) such as in Jabodetabek. It should be understood that the transmission of Covid-19 can occur anywhere, not only in KRL," said Adita, Tuesday (5/5/2020). First, passengers are required to wear masks. Second, he continued, officers check passengers' body temperatures. [254 words are abbreviated from here]
Gold summaries: Kemenhub menyebutkan Permenhub 18/2020 secara tegas telah menyatakan adanya beberapa syarat yang wajib dipenuhi penumpang moda transportasi publik seperti KRL . **IndoBERT:** Kementerian perhubungan (Kemenhub) memastikan pelaksanaan protokol di kereta rangkaian listrik (KRL) Jabodetabek terus berjalan **IndoBERT with additional contexts (+ keyphrases + titles):** Kemenhub memastikan pelaksanaan protokol di kereta rangkaian listrik (KRL) jabodetabek terus berjalan . **mBART:** Permenhub 18/2020 secara tegas telah menyatakan adanya beberapa syarat yang wajib dipenuhi penumpang moda transportasi publik seperti KRL. **mBART with additional contexts (+ keyphrases + titles):** Adita mengatakan, Permenhub 18/2020 secara tegas telah menyatakan adanya beberapa syarat yang wajib dipenuhi penumpang moda transportasi publik seperti KRL. **mT5:** Kemenhub memastikan pelaksanaan protokol di Kereta Rangkaian Listrik (KRL) Jabodetabek terus berjalan. **mT5 with additional contexts (+ keyphrases + titles):** Juru Bicara Kementerian Perhubungan Adita Irawati menyatakan, pihaknya telah mengeluarkan Permenub Nomor 18/2020 yang telah mengatur operasional moda transportasi di masa pandemi.	**Gold summaries:** The Ministry of Transportation stated that Permenhub 18/2020 has explicitly stated that there are several conditions that must be met by passengers of public transportation modes such as KRL. **IndoBERT:** The Ministry of Transportation (Kemenhub) ensures that the implementation of the protocol on the Jabodetabek Electric Circuit Train (KRL) continues **IndoBERT with additional contexts (+ keyphrases + titles):** The Ministry of Transportation ensures that the implementation of the protocol on the Jabodetabek electric circuit train (KRL) continues. **mBART:** Permenhub 18/2020 has explicitly stated that there are several conditions that must be met by passengers of public transportation modes such as KRL. **mBART with additional contexts (+ keyphrases + titles):** Adita said that Permenhub 18/2020 has explicitly stated that there are several conditions that must be met by passengers of public transportation modes such as KRL. **mT5:** The Ministry of Transportation ensures that the implementation of the protocol on the Jabodetabek Electric Circuit Train (KRL) continues. **mT5 with additional contexts (+ keyphrases + titles):** Spokesperson for the Ministry of Transportation, Adita Irawati, stated that her party had issued Permenub No. 18/2020 which regulates the operation of transportation modes during the pandemic.

Figure 2: Example from the LipKey dataset, with gold-standard and generated summaries.

Indonesian	English (translation)
Gold Keyphrases: Relawan Uji Vaksin, Vaksin Sinovac **Article:** liputan6 . com , jakarta - manajer lapangan tim riset uji klinis vaksin covid-19 sinovac , dr eddy fadliyana menyebut sejauh ini sudah ada sekitar 1 . 020 calon relawan yang mendaftarkan diri untuk mengikuti uji vaksin dari tiongkok itu . dia mengatakan , pelaksanaan uji vaksin itu akan dilakukan selasa 11 agustus 2020 . pada hari pertama itu , uji vaksin bakal dilakukan di rumah sakit pendidikan (rsp) universitas padjadjaran , jalan eyckman , kota bandung . " sebetulnya sama saja , hanya pemeriksaan di rsp itu , tes usapnya (swab test) didahulukan . sama saja sih prosedurnya , tidak ada yang berbeda , besok rsp imunisasi , kalau di tempat lain baru tahap awal , " kata eddy di bandung , senin (10/8/2020) . dikutip dari antara , menurut eddy , semua tempat yang ditunjuk menjadi lokasi uji vaksin covid-19 ini dipastikan sudah siap . mulai dari sarana prasarananya , menurutnya sudah sesuai dengan protokol kesehatan yang berlaku . dia mengatakan , uji vaksin itu dilakukan di enam lokasi , di antaranya yakni rsp unpad , balai kesehatan unpad dipatiukur , puskesmas dago , puskesmas sukapakir , puskesmas garuda , dan puskesmas ciumbuleuit . dari seluruh calon relawan yang sudah mendaftar , menurutnya tak menutup kemungkinan sudah ada asn yang ikut mendaftar . karena , pendaftaran untuk menjadi relawan itu terbuka untuk umum . " dari asn mungkin ada , saya tidak melihat statusnya apa pokoknya masyarakat yang mau silakan saja , " katanya . meski terbuka untuk umum , menurutnya ada beberapa syarat yang perlu dipenuhi oleh calon relawan antara lain usia relawan dalam rentang 18 hingga 59 tahun , dan dalam keadaan sehat tanpa penyakit bawaan .	**Gold Keyphrases:** Vaccine Test Volunteers, Sinovac Vaccines **Article:** liputan6 . com, Jakarta - field manager of the Sinovac Covid-19 vaccine clinical trial research team, Dr. Eddy Fadliyana, said that so far there have been around 1. 020 prospective volunteers who registered to take part in the vaccine test from China. He said the implementation of the vaccine test would be carried out on Tuesday, August 11, 2020. On that first day, the vaccine test will be conducted at the Teaching Hospital (RSP) at Padjadjaran University, Jalan Eyckman, Bandung City. "it's actually the same, only the examination at the rsp, the swab test takes precedence. the procedure is the same, nothing is different, tomorrow the immunization rsp, if it's in place others are only in the early stages," said Eddy in Bandung, Monday (10/8/2020). Quoted from Antara, according to Eddy, all the places designated to be the test locations for the COVID-19 vaccine are confirmed to be ready. starting from the infrastructure, according to him, it is in accordance with the applicable health protocol. he said the vaccine test was carried out in six locations, including the Unpad Hospital, Dipatiukur Health Center, Dago Health Center, Sukapakir Health Center, Garuda Health Center, and Ciumbuleuit Health Center. From all prospective volunteers who have registered, according to him, it is possible that there are already ASN who have registered. because , registration to become a volunteer is open to the public . " From the ASN there may be , I do not see what the status is , basically people who want to go ahead , " he said . although it is open to the public , according to him , there are several requirements that need to be fulfilled by prospective volunteers , including the age of volunteers in the range of 18 to 59 years , and in good health without any congenital disease .
Gold Keyphrases: buaya terkam warga, Sulbar **Article:** liputan6 . com , mamuju tengah - kejadian nahas menimpa h (40) warga desa barakkang , kecamatan budong-budong , mamuju tengah , sulawesi barat . ibu rumah tangga itu diterkam seekor buaya saat mandi dan buang air besar di sungai . kapolsek budong-budong akp suparman membenarkan pristiwa nahas itu , ia mengatakan , peristiwa terjadi pada selasa (4/8/2020) dini hari , sekitar pukul 05 . 30 wita . korban yang tengah buang air besar itu tiba-tiba diterkam buaya yang memiliki panjang kurang lebih 7 meter . " menurut saksi andi (38) yang merupakan adik korban , buaya itu tiba-tiba menerkam korban dari belakang , " kata suparman kepada liputan6 . com . petani labuhan batu utara diterkam buaya di depan anak istri suparman menambahkan , saksi juga sempat mendengarkan teriakan korban dan berusaha untuk menolong . namun , belum sempat menolong , buaya tersebut sudah terlebih dahulu menarik korban ke dalam air . " beberapa saat kemudian korban dan buaya muncul di permukaan air namun hanya sesaat lalu kemudian tenggelam lagi ke dalam air , " jelas suparman . hingga saat ini korban belum juga ditemukan , warga bersama pihak kepolisian sempat melakukan pencarian dengan peralatan seadanya . pihak bpbd mamuju tengah dan basarnas mamuju pun sudah dihubungi . " saat ini bpbd dan masyarakat serta basarnas sudah ada di tkp melakukan pencarian , " tutup suparman .	**Gold Keyphrases:** Crocodile devours residents, Sulbar **Article:** liputan6 . com, Mamuju - an unfortunate incident happened to H (40) a resident of Barakkang Village, Budong-Budong District, Central Mamuju, West Sulawesi. The housewife was attacked by a crocodile while bathing and defecating in the river. The head of the Budong-Budong Police, AK Suparman, confirmed the unfortunate incident, saying that the incident occurred on Tuesday (4/8/2020) early in the morning, around 05 am. 30 pm. The victim who was defecating was suddenly attacked by a crocodile which has a length of approximately 7 meters. " According to witness Andi (38) who is the victim 's younger brother , the crocodile suddenly pounced on the victim from behind , " said Suparman to liputan6 . com . The farmer in North Batu Harbor was attacked by a crocodile in front of his wife and children. Suparman added that the witness had also heard the victim's screams and tried to help. however , before they could help , the crocodile had already pulled the victim into the water . " a few moments later the victim and the crocodile appeared on the surface of the water , but only a moment later then sank again into the water , " explained Suparman . Until now the victim has not been found , residents together with the police had conducted a search with makeshift equipment . The Central Mamuju BPBD and Mamuju Basarnas have also been contacted. " Currently , BPBD and the community as well as the National Basis are already at the scene conducting a search , " concluded Suparman .

Figure 3: Example of articles and keyphrases in the `LipKey` dataset. We highlight words in the article that match its absent keyphrases with different colours. Yellow means partial match, green means acronym, and blue means morphology variants. The English translation is for illustration purposes.

Understanding Attention for Vision-and-Language Tasks

Feiqi Cao[1], Soyeon Caren Han[1,2], Siqu Long[1], Changwei Xu[1], Josiah Poon[1]
[1]School of Computer Science, The University of Sydney, Australia
[2]School of Physics, Maths and Computing, The University of Western Australia, Australia
{fcao0492, slon6753, chxu7327}@uni.sydney.edu.au
{caren.han, josiah.poon}@sydney.edu.au

Abstract

Attention mechanism has been used as an important component across Vision-and-Language(VL) tasks in order to bridge the semantic gap between visual and textual features. While attention has been widely used in VL tasks, it has not been examined the capability of different attention alignment calculation in bridging the semantic gap between visual and textual clues. In this research, we conduct a comprehensive analysis on understanding the role of attention alignment by looking into the attention score calculation methods and check how it actually represents the visual region's and textual token's significance for the global assessment. We also analyse the conditions which attention score calculation mechanism would be more (or less) interpretable, and which may impact the model performance on three different VL tasks, including visual question answering, text-to-image generation, text-and-image matching (both sentence and image retrieval). Our analysis is the first of its kind and provides useful insights of the importance of each attention alignment score calculation when applied at the training phase of VL tasks, commonly ignored in attention-based cross modal models, and/or pretrained models. Our code is available at: https://github.com/adlnlp/Attention_VL

1 Introduction

The relative maturity and flexibility of deep learning allow us to build upon the success of computer vision and natural language processing to face many complex and multimodal Vision-and-Language (VL) tasks, such as Visual Question Answering (VQA), Text-and-Image Matching (T&I Match), or Text-to-Image Generation (T2I Gen). For these VL tasks, it is crucial to effectively align the multimodal information in both visual and linguistic domains. For example, to pick the right answer in VQA, the model should empower information from the input image, together with aligning the linguistic meanings with visual clues.

Attention mechanism (Bahdanau et al., 2015; Luong et al., 2015a) has been used as an important component across a wide range of VL models; from the early-stage attention-based fusion VL models (Shih et al., 2016; Wang et al., 2019; Xu et al., 2018a; Yang et al., 2016) to the recent VL multimodal transformer-based pretrained models (Hu et al., 2021; Li et al., 2020b; Lu et al., 2019; Long et al., 2022a). Those attention-based VL models mainly focus on 1) exploring new features to represent visual and linguistic information as an input of attention layer, 2) deciding the position or the number of attentions in the model, or 3) investigating the interpretability of attention distribution on VL tasks by emphasising the specific image regions or textual tokens (Luo et al., 2021).

While such approaches and investigations have resulted in interesting findings in different aspects of VL tasks, the attention alignment calculation between vision and language modalities has been less explored. However, the alignment calculation is directly linked to the main purpose of using attention mechanisms in VL tasks, which is to effectively bridge and align two different visual and linguistic information. In other words, the essence of the attention mechanism in VL tasks is the alignment score calculation, as it quantifies the amount of "Attention" that the visual features would place on each of the language representations (or linguistic features would empower on the specific visual regions) when bridging the semantic gap between visual and language features. Most existing VL models directly apply the two attention alignment functions, a general and a dot-product (Luong et al., 2015a), which are commonly used in several NLP tasks. Since Vaswani et al. (2017) proposed a scaled dot-product for the transformer with full attention, almost every VL paper has directly applied those three attention alignment score functions. Instead, little work has been done towards

Proceedings of the 29th International Conference on Computational Linguistics, pages 3438–3453
October 12–17, 2022.

understanding the role of attention alignment calculation methods applied to bridge visual and linguistic features, and exploring the impact on different VL model performance.

In order to address this limitation, the overarching goal of this research is to perform an extensive and systematic assessment of the effect of a range of attention alignment mechanisms pertaining to VL tasks, including three major VL tasks: Visual Question Answering (VQA), Text-and-Image Matching (T&I Match), and Text-to-Image Generation (T2I Gen). Towards that end, we systematically analyse the impact of the position of query and key in attention alignment on VL tasks. We investigate the following three questions: i) Which attention alignment score calculation yields the most benefit in VL tasks? ii) What if we linearly transform the query Q instead of the key K (or vice versa) before the multiplication? For example, assume the textual feature T is a query Q, and the image feature I is a key K. We analyse the impact of linear transforming Q or K in alignment score calculation. iii) Do the attention alignment calculation techniques with better performance provide better attention distribution interpretability?

In brief, our **main contributions** are as follows:

1) We conduct a comprehensive analysis of the role of attention alignment score calculation in VL tasks (including three widely-used VL tasks, such as Visual Question Answering, Text-and-Image Matching, and Text-to-Image Generation). **2)** We perform a comparative analysis of the position of query and key (language and visual feature) for the alignment calculation. **3)** We evaluate the interpretability of the best and worst attention alignment calculation models. **4)** We make the code and the data publicly available to encourage reproducibility of results.

2 Related Works

VL models directly adopt the attention mechanism to bridge the visual and linguistic modal information. In this section, we review the related works for the role of attention mechanisms in different VL tasks within the focus of our analysis.

Text-to-Image Generation AttnGAN (Xu et al., 2018a) first proposed to use dot-product for measuring the alignment between visual subregions and word tokens to guide the image generation process. Many of the later approaches directly adapted the dot-product attention from AttnGAN (Han et al., 2020; Li et al., 2020a, 2019; Pande et al., 2021; Qiao et al., 2019a,b; Yin et al., 2019; Zhu et al., 2019). A few models apply the element-wise multiplication (Qiao et al., 2019a,b) or cosine similarity (Zhang et al., 2021a) for measuring the attention alignment.

Text-and-Image Matching The cosine similarity based attention alignment proposed by SCAN (Lee et al., 2018) is most widely used in Text-and-Image Matching (Chen et al., 2020; Chen and Luo, 2020; Diao et al., 2021; Dong et al., 2021; Liu et al., 2019). They applied text-to-image (t2i) and image-to-text(i2t) attention in two separate variants to filter the cross-modal relevant representations for later image-sentence matching. Some other approaches applied (scaled) dot-product instead (Fei et al., 2021; Liu et al., 2020; Wang et al., 2019; Wei et al., 2020; Long et al., 2022b).

Visual Question Answering (VQA) Both textual query-guided image attention and image-guided textual query attention have been commonly used in VQA approaches, which utilised one modality to guide the focus on the other. Several categories of alignment calculations or their variants were included, such as adapting neural networks (Anderson et al., 2018; Patro and Namboodiri, 2018; Yang et al., 2016; Zhu et al., 2016) or applying (scaled) dot-product (Gao et al., 2019; Guo et al., 2021; Huang et al., 2020; Hudson and Manning, 2018; Rahman et al., 2021; Yu et al., 2019; Zhang et al., 2021b) etc.

Text-based Visual Question Answering Recent TextVQA approaches directly augmented existing VQA models and their cross-modal attention with additional OCR inputs (Biten et al., 2019a,b; Singh et al., 2019; Wang et al., 2020). Both early-stage model *M4C* (Hu et al., 2020) and the most recent pretrained model *TAP* (Yang et al., 2021) fed the question, image and OCR text together into a multimodal transformer and jointly encoded them via scaled-dot product attention in the transformer encoder.

Nevertheless, there is a lack of research on exploring the most effective cross-modal attention alignment. Hence, we apply different cross-modal attention alignment methods to the most widely adopted baselines for these aforementioned VL downstream tasks: AttnGAN (Text-to-Image Generation), SCAN (Text-to-Image Matching), MAC (VQA), and M4C (TextVQA), and examine the impact of different attention alignment score via

in-depth analysis.

3 Attention Alignment Mechanism

There are various attention mechanisms applied in different multimodal VL downstream tasks. Two commonly used approaches are the cross-attention and the self-attention. First, the cross-attention is performed between visual and textual inputs. More specifically, given a sequence of textual features $T = \{t_1, t_2, t_3, \ldots, t_M\}$ and image features $I = \{i_1, i_2, i_3, \ldots, i_N\}$, it takes T as the query Q and I as the key K (or vice versa) to compute attention and context vectors c as the attended representations of the input elements in the following way:

$$a_{xy} = f(Q_x, K_y) \tag{1}$$

$$\alpha_{xy} = \frac{exp(a_{xy})}{\sum_{y=1}^{n_K} exp(a_{xy})} \tag{2}$$

$$c_x^K = \sum_{y=1}^{n_K} \alpha_{xy} K_y \tag{3}$$

where f is a function to calculate attention score, n_K is the number of elements in K, and c_x^K is the context vector of K with respect to the x-th element of Q. The second approach, self-attention (Vaswani et al., 2017), is performed over all inputs from both modalities. In other words, the approach combines T and I as a complete sequence $S = T \cup I$, and converts all elements in S into Q, K and V via learnable matrices, which are used to compute attention by multiple heads in the following way:

$$\text{Attention}(Q, K, V) = \text{Softmax}(f(Q, K))V \tag{4}$$

where the results from different heads are combined together. Then, it applies layer normalization, residual connections and fully connected layers in order to obtain the attended representation of the input tokens. With both approaches, we explore the effect of the attention alignment calculation f for different VL tasks with the following five different alignment score functions. We also include **Cosine similarity**-based attention for only Text-and-Image Matching as it is widely used in that specific domain.

Dot product attention It was proposed in NMT (Bahdanau et al., 2015) to compute vector similarity between encoder hidden states and decoder hidden states. This function (Luong et al., 2015a) has been widely adopted as f in the cross-attention mechanism as shown in Equation 1.

$$f(Q, K) = QK \tag{5}$$

Scaled dot product attention The higher dimension of data representation would lead to the smaller gradient of softmax function. Hence, the scaling factor was introduced by (Vaswani et al., 2017), and applied to the self-attention-based VL approaches as represented in Equation 4.

$$f(Q, K) = \frac{QK}{\sqrt{d}} \tag{6}$$

General attention Along with dot product attention, general attention (Luong et al., 2015a) received lots of interest as an alternative alignment calculation method that computes attention score using an extra learnable matrix to linearly transform K into the same embedding space as Q. This can be considered as one of the neural network based methods mentioned in Section 2.

$$f(Q, K) = QWK \tag{7}$$

There are several variants of neural network based general attention calculation methods. First, **Biased general attention** is introduced by (Sordoni et al., 2016) using more bias towards more important keys regardless of the query context.

$$f(Q, K) = Q(WK + b) \tag{8}$$

Secondly, **Activated general attention.** (Ma et al., 2017) applies an additional nonlinear activation term, which is able to amplify the emphasis on query elements that are highly relevant to the key.

$$f(Q, K) = act(Q(WK + b)) \tag{9}$$

In this paper act is the ReLU activation since it is a widely used function in VL downstream tasks.

4 Vision-Language Models

We use publicly available implementations of the most widely adopted VL baseline models[1] in order to train and evaluate different attention alignment score calculation for three different VL tasks: (i) **AttnGan** for Text-to-Image Generation (T2I Gen), (ii) **SCAN** for Text-and-Image Matching (T&I Match), (iii) **MAC** and **M4C** for each Visual Question Answering (VQA) and Text-based Visual Question Answering (TVQA).

[1]All the VL pretrained models are just based on BERT (attention-oriented transformer-based). It is still quite early-stage in this field, and more VL pretrained models are still emerging in 2022 (Zhuge et al., 2021; Hong et al., 2021)

4.1 T2I Gen: AttnGAN

The goal of text-to-image generation is to generate a visually realistic image that matches a given text description. The AttnGAN (Xu et al., 2018b) generates images by using multiple generators with the attention mechanisms. To improve the image quality at each step, a cross-attention mechanism is performed between caption words and image regions, and it produces the attended word context for each image region. Given a caption of M words, an image with N sub-regions would be generated by an upsampling network. The words and image regions are represented as d-dimensional vectors $\{t_m\} \in T$ and $\{i_n\} \in I$ respectively. Then image representation I is applied as Q the query and caption representation T is applied as K the key for the cross-attention mechanism (Equations 1, 2, 3), where the dot product attention score calculation is used as f. The resultant textual context would be fused with word region representations as a guide for the generator at the next time step to focus on different words. Note that we evaluate different alignment calculation methods as f to investigate the impact of the image generation performance. We fix I as Q and T as K, and replace the dot product with other alignment score calculations.

4.2 T&I Match: SCAN

Text-and-image matching (a.k.a. Text-and-image retrieval) refers to measuring the visual-semantic similarity between a sentence and an image. The SCAN model (Lee et al., 2018) performs a pairwise cross-attention between image regions and caption words for fine-grained T&I Match. This can be done in two directions. Given a caption of M words and an image having N detected objects, d-dimensional representations $\{t_m\}$ and $\{i_n\}$ are obtained as T and I respectively. To obtain the attended image context for each caption word, the cross-attention mechanism (described in Equations 1, 2) is applied with T being the query Q and I being the key K, and an alignment score is measured by using cosine similarity between each caption word and its image context. These alignment scores would be aggregated via a pooling function as the final alignment score between the given image and caption. Such scores can be obtained by using T as K and I as Q to calculate the sentence context for each image region. In experiments, we fix T as Q and I as K, and replace the cosine similarity with other alignment score calculations.

4.3 VQA

We explore two VQA downstream tasks, Visual Question Answering with compositional reasoning and Text-based Visual Question Answering.

4.3.1 VQA: MAC

First, we focus on the visual question answering task that requires responding to natural language questions about images, specifically with a compositional and structured nature. The MAC circuit (Hudson and Manning, 2018) applies a cross-attention mechanism to answer a question based on a given image. Instead of computing attention between textual and visual input, MAC introduces a d-dimensional learnable control state e as a guidance for MAC cells to selectively attend to different aspects of inputs at each time step. Within each MAC cell, there is a control unit to attend to the question words and a read unit to attend to the image regions. Given a question of M words and an image having N detected objects, d-dimensional representations $\{t_m\}$ and $\{i_n\}$ are obtained as T and I respectively. Instead of using Equation 1, the control unit applies e as Q and T as K to compute the attention score in the following way:

$$a_y = W'(f(Q, K_y)) + b' \tag{10}$$

where f indicates element-wise dot product multiplication to obtain a d-dimensional similarity vector, and W' and b' are learnable parameters to output a scalar as the score. Then the control unit follows Equations 2, 3 to obtain textual context as an update for e. Similar to the control unit, the read unit applies e as Q and I as K to obtain the question-guided visual context from the image, which is later aggregated to predict an answer. Therefore the read unit can be considered as a main component in MAC that involves multimodal alignment. Hence, for the evaluation, we fix the control state e (which majorly contains textual question information) as Q and image-based knowledge graph I as K, and adapt the focused attention alignment calculation methods f with the element-wise multiplication manner in the read unit.

4.3.2 TVQA: M4C

Secondly, Text-based visual question answering (TVQA) is an extension of VQA, which requires the model to read text over the image to answer the questions. The M4C model (Hu et al., 2020) applies a multimodal transformer over all input

Tasks	Dataset	Train	Dev	Test
T2I Gen	CUB	8,855	-	2,933
	MS-COCO	82,783	-	15,000
T&I Match	Flickr30k*	29,000/145,000	1,000/5,000	1,000/5,000
	MS-COCO*	29,000/145,000	1,000/5,000	1,000/5,000
VQA	CLEVR*	70,000/699,989	15,000/149,991	15,000/149,988
	Text-VQA*	21,953/34,602	3,166/5,000	3,289/5,734

Table 1: Details of train/dev/test split for each dataset. Note that * indicates the dataset having different numbers for visual and textual inputs. It reports the number of images followed by the number of captions or question-answer pairs, separated by backslash (/).

modalities to perform iterative answer prediction for the TextVQA task. More specifically, given a question of M words, an image having N detected objects and O detected OCR tokens, d-dimensional representations $\{s_m^{ques}\}$, $\{s_n^{obj}\}$ and $\{s_o^{ocr}\}$ are obtained as input sequence S. The self-attention mechanism (Equation 4) with scaled dot product attention is applied over S, the sequence of all $M+N+O$ entities. In this way, both intra-modal interactions and inter-modal interactions are captured to aggregate the input to form an answer prediction via classical transformer layers. Similarly to other tasks, we replace the scaled dot product attention calculation with the other aforementioned options for f to investigate the impact in TVQA.

5 Evaluation Setup

5.1 Datasets

We conducted experiments on three VL task datasets. The statistics is shown in Table 1. We followed the work of the base models, including AttnGAN (Xu et al., 2018b), SCAN (Lee et al., 2018), MAC (Hudson and Manning, 2018), M4C (Hu et al., 2020) for dataset preprocessing and dividing for train/dev/test.

5.1.1 T2I Gen

Two benchmark datasets are used: **Caltech-UCSD Birds 200 (CUB)**[2] and **MS-COCO**[3]. CUB has 11,788 images of 200 bird categories downloaded from the Flickr website, each with 10 textual captions. MS-COCO provides 123,287 images of complex everyday scenes with 5 manually written textual descriptions per image. We use a train/test split of 8,855/2,933 and 82,783/15,000 images respectively for CUB and MS-COCO.

5.1.2 T&I Match

Flickr30k[4] contains around 31k images collected from the Flickr website with 5 crowd-sourced captions per image. We test on Flickr30k with train/dev/test split of 29k/1k/1k images and on MS-COCO (as described above) with 29k/1k/1k images.

5.1.3 VQA

We have two VQA tasks: 1) Visual Question Answering with compositional reasoning, and 2) Text-based Visual Question Answering. We used **CLEVR**[5] and **TextVQA**[6] respectively. CLEVR contains 100,000 synthetic images of 3D shapes with 999,968 questions/answers in total. We use a subset of 70,000 images with 699,989 QAs for training, 15,000 images with 149,991 QAs for validation and 15,000 images with 149,988 QAs for test. TextVQA consists of 45,336 questions asked by (sighted) humans on 28,408 images from the Open Images dataset (Krasin et al., 2017). We use the original split: 21,953 images with 35,602 QAs, 3,166 images with 5,000 QAs and 3,289 images with 5,734 QAs for training, validation and test.

5.2 Evaluation Metrics

We describe metrics for assessing the impact of attention alignment mechanism for each VL task.

5.2.1 T&I Match: R@K

We measure the performance of sentence retrieval and image retrieval by recall at K (R@K), which is defined as the percentage of queries that get the correct item at the closest K points to the query. The higher the value, the better the performance.

5.2.2 T2I Gen: Inception Score(IS) & FID

The evaluation measurement we use is **Inception Score (IS)** which seeks to capture the image quality and image diversity properties of a collection of generated images. The higher the inception score, the better the model. **Fréchet Inception Distance (FID)** measures the similarity between the generated images and the real images by comparing their Frechét distance between the maximum entropy distribution. Lower FID indicates higher similarity.

[2] http://www.vision.caltech.edu/visipedia/CUB-200-2011.html

[3] https://cocodataset.org/#home

[4] http://shannon.cs.illinois.edu/DenotationGraph/

[5] https://cs.stanford.edu/people/jcjohns/clevr/

[6] https://textvqa.org/dataset

Attention	Sentence Retrieval						Image Retrieval					
	Flickr30K			MS-COCO			Flickr30K			MS-COCO		
	R@1	R@10	Rsum	R@1	R@10	Rsum	R@1	R@10	Rsum	R@1	R@10	Rsum
cosine similarity$^{\diamond}$	62.4	93.3	243.8	**61.4**	94.5	243	43.9	81.8	199.9	45.7	88.2	212.5
dot product	62.1	92.1	240.4	59.7	**95.2**	243.5	44.8	**82.1**	200.6	**46.0**	87.9	212.6
scaled dot product	63.0	**93.8**	244.9	59.3	95.1	**243.7**	44.9	81.9	200.4	45.8	**88.3**	**213.4**
general*	**63.2**	93.6	**245.0**	59.8	**95.2**	**243.7**	**46.7**	81.8	**201.8**	45.6	87.8	212.1
general†	56.6	90.1	229.9	53.8	93.1	231.5	38.4	77.0	182.0	39.3	83.4	195.3
biased general*	56.6	89.8	230.3	52.2	91.7	227.0	39.6	77.3	185.0	38.7	83.4	194.5
biased general†	55.8	89.7	228.3	52.6	93.2	231.1	39.3	77.4	184.6	39.8	84.2	197.3
activated general	56.2	90.5	229.2	53.9	92.9	231.3	39.2	77.4	184.7	39.5	84.0	195.6

Table 2: R@1, R@10 and the sum of (R@1+R@5+R@10) on Flickr30K and MS-COCO for T&I Match. The definition of $\diamond$, *, † can be found in footnote 8. Q refers to caption words and K refers to image regions.

5.2.3 VQA

For the **VQA with compositional reasoning**, overall accuracy is used to measure the performance of the VQA models. The higher the accuracy, the better the performance of the model. For the **TVQA**, it is designed for the VQA context where 10 ground truth answers are provided for each question-image pair. The accuracy of a single prediction is a soft score obtained by a vote of the 10 ground truth answers. Overall accuracy is obtained by taking the average across all instances. We also use the Average Normalized Levenshtein Similarity (ANLS) score (Biten et al., 2019b) to eliminate the dropped performance caused by OCR recognition error by comparing the string similarity between the ground truth and the prediction.

5.3 Experimental Settings

For **T&I Match: SCAN** (t-i) AVG models, all settings of hyper-parameters follow the configuration of the SCAN. The batch size is 128, the margin of triplet loss α is 0.2 and the threshold of maximum gradient norm for gradient clipping is 2. For Flickr30k models, the learning rate is set as 0.0002 for the first 15 epochs and then lowered to 0.00002 for another 15 epochs. Total training epochs are 30 and the best model is selected with the highest sum of R@K score. For MS-COCO models, we trained with a learning rate of 0.0005 for 10 epochs and then lowered the learning rate to 0.00005. The best model is selected with the highest sum of R@K score. Training epochs are 20. For **T2I Gen: AttnGan** model on CUB dataset, the batch size is set to be 20 and we trained with 400 epochs in total. On the MS-COCO dataset, the batch size is 14 and total epochs are 90. In addition to this, all settings are the same as the AttnGan. For **VQA: MAC** models, the training epoch is set to be 8 and other hyperparameter settings are consistent with MAC. More specifically, the batch size is 128, the learning rate is 0.0001 with 0.5 learning decay rate and the threshold of maximum gradient norm for gradient clipping is 8. For **TVQA: M4C** model on the Text-VQA dataset, we followed the exact same setting as M4C, applying the batch size of 128 and 100 epochs for training, All model variants would train to convergence within 80 epochs.

All experiments for T2I Gen, T&I Match and VQA are conducted on a variety of cloud instances from Google Colab, with each utilising an NVIDIA Tesla T4 GPU of 16GB RAM. For TVQA the experiments are conducted utilising NVIDIA Titan RTX GPU with 24GB RAM, 16 Intel(R) Core(TM) i9-9900X CPU @ 3.50GHz with 128GB RAM, and the operating system of Ubuntu 20.04.1.

6 Results

We analyse the impact of attention alignment mechanisms in different VL tasks, and explore the interpretability based on attention distribution.

6.1 Test Performance

A primary goal of this work is to identify the most effective and successful attention alignment calculation functions for VL tasks. Tables 2, 3, and 4 [7] detail the results of our experiments comparing performance of individual alignment functions with each VL models. Each table visualises the trends with a heatmap. The darker the colour of the cells, the better the performance. As shown in Table 2 for the T&I Match task that the original calculation function, cosine similarity, achieved quite good performance. However, scaled dot product and general* demonstrated a consistent superiority for both sentence retrieval and image retrieval on both Flickr30K and MS-COCO. Comparatively,

[7] $\diamond$ indicates the original attention alignment function used by the base models. * indicates $f(K, Q)$ (swapping query and key), and † indicates $f(Q, K)$ (without swapping query and key) for Equations 7 and 8

Attention	Acc.
dot product$^{\diamond}$	0.966
scaled dot product	**0.973**
general*	0.967
general†	0.962
biased general*	0.959
biased general†	0.963
activated general	0.971

(a) VQA on CLEVR

Attention	Acc.	ANLS
dot product	0.407	0.545
scaled dot product$^{\diamond}$	**0.419**	**0.554**
general*	0.407	0.546
general†	0.416	**0.554**
biased general*	0.412	0.553
biased general†	0.414	0.551
activated general	0.413	0.548

(b) TVQA on Text-VQA

Table 3: Results for VQA/TVQA. The definitions of $\diamond$, *, $\dagger$ are in footnote 8. For VQA, Q refers to the control state and K refers to image-based knowledge graph in read unit. For TVQA, Q and K are transformed union of all caption words, image object and OCR features.

Attention	CUB		MS-COCO	
	IS	FID	IS	FID
dot product$^{\diamond}$	4.32	**25.72**	23.28	**40.19**
scaled dot product	4.31	25.74	23.84	42.33
general*	4.36	28.21	24.28	40.82
general†	4.26	26.94	24.63	42.45
biased general*	4.13	26.97	23.05	43.10
biased general†	4.30	25.89	**25.24**	43.64
activated general	**4.41**	28.39	23.56	42.65

Table 4: Results on CUB and MS-COCO for T2I Gen. The definitions of $\diamond$, *, $\dagger$ are in footnote 8. Q refers to caption words and K refers to image subregions.

biased and activated general attentions produced very low results overall.

Table 3 details the performance of alignment functions for VQA tasks, including (a) VQA with compositional reasoning (CLEVR) and (b) TVQA (Text-VQA). Surprisingly, both VQA and TVQA models produced the best performance with a scaled dot product alignment, highlighting its overall effectiveness for the VQA tasks. We note that the activated general attention (ReLU activation) performed well for VQA but produced one of the lowest ANLS scores in TVQA. The general attention alignment function also showed the similar trend. Considering the different nature of general VQA and TVQA, where the latter mainly focuses on OCR text input, it is unsurprising that the impact of alignment mechanism is discrepant. Hence, it is remarkable to find that the scaled dot product achieved the best in both tasks.

The T2I Gen results in Table 4 illustrated quite different trends compared to the two aforementioned tasks. First, none of the alignment functions produced a consistently better performance in both evaluation metrics, IS or FID. While neural network-based alignment functions (i.e. general, biased and activated general) achieved higher IS scores than others, the dot product dominated in FID scores, for both CUB and MS-COCO. The scaled dot product obtained comparably good FID results but not in IS. This can be aligned with the different metrics of measurement that FID counts on the similarity between the ground-truth images and the images generated from the text whereas IS expects the diversity of the generated image. Hence, a better VL alignment leads to the better FID but not necessarily the better IS.

In summary, we can find the scaled dot product can be the best alignment calculation function for both cross and self-attention that can successfully bridge the visual and textual information, as it produces considerably and consistently better results for all three VL tasks across all six VL datasets.

6.2 Impact of Key and Query

Since the previous works (Luong et al., 2015b; Sordoni et al., 2016; Ma et al., 2017) do not have a standard choice of linearly transforming key or value when calculating attention scores, we also investigated the impact of position of query and key in the attention alignment calculation process, especially when extra learnable weights and biases are involved. We explore the difference between linearly transforming the key K to multiply with the query Q ($f(K, Q) = KWQ$) and transforming the query Q to multiply with the key K ($f(Q, K) = QWK$) in general attention and biased general attention calculation. Specifically, we initially fixed the textual information as a query Q and visual information as a key K (Equation 7 and 8) and swapped the position in different general attention alignment score measurements. In Table 2, 3, and 4, * indicates the functions with $f(K, Q)$, whereas $\dagger$ refers to those with $f(Q, K)$.

Table 2 shows that Flickr30K performed better with general* or biased general*, whereas MS-COCO does not have obvious trends. Similar patterns can be found in both cross-attention mechanisms (VQA models, T2I Gen models), and self-attention mechanism TVQA models. Interestingly but unsurprisingly, we note that there is no obvious and consistent performance improvement pattern in different positions of textual information (query Q) and visual information (key K) when it calculates the alignment. It depends on the specific downstream tasks and dataset. We can conclude that the way of calculating alignment is the crucial point in VL tasks, compared to the position/order of different modal information.

Attention	Exist (single)	Query Attribute (single)	Compare Attribute (two)	Count (multiple)	Compare Integer (multiple)
scaled dot product	**0.9912**	**0.9928**	**0.9860**	**0.9200**	**0.9780**
biased general*	0.9883	0.9920	0.9832	0.8938	0.9014

Table 5: Breakdown analysis of VQA accuracy regarding different question types. The number of queried objects in the questions are included in the brackets.

6.3 Breakdown Analysis

VQA To further investigate the difference between attention calculation methods, we report the accuracy for the best/worst performing attention calculation methods regarding different question categories in CLEVR. As shown in Table 5, the two models achieved similar performance for question types *Exist*, *Query Attribute*, and *Compare Attribute*. *Exist* and *Query Attribute* types normally contain single queried object in the question, and *Compare Attribute* questions would contain two queried objects. Those question types only require models to attend to one or two objects in the image, so it is easier for model to capture the pattern when learning the alignment between image and question. However, the *Count* and *Compare Integer* questions are challenging to answer as finding multiple objects with the same attributes is required. The models need to learn how to align multiple objects to one noun phrase. In this case, scaled dot product attention always works better than biased general attention by up to 7.66% accuracy, which suggests that scaled dot product attention can learn more accurate alignment between image regions and question words.

T&I Match The VQA (CLEVR) only includes a limited set of objects with limited attribute descriptions, so we also investigated the effect of real-world images with more diverse types of objects and descriptions using MS-COCO. We compared the retrieval Rsum of the best performing and the worst performing attention calculation methods in terms of different image annotation supercategories in Figure 1 and the number of caption nouns in Figure 2. The supercategories on the x-axis in Figure 1 are sorted based on the percentage of the test set images which contain that specific supercategory in the annotations, as indicated by the value in the brackets below each supercategory. From Figure 1a, it is clearly observed that scaled dot product attention can consistently perform much better than the biased general* attention for most categories such as *person*, *vehicle* and *electronic*,

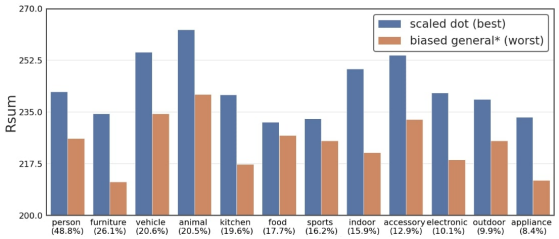

(a) Text Retrieval

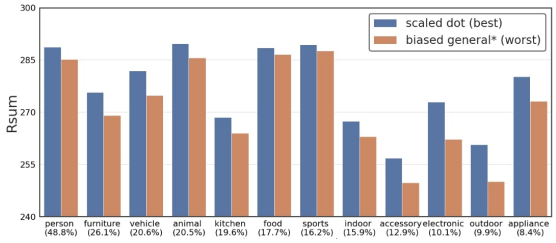

(b) Image Retrieval

Figure 1: Breakdown analysis of T&I Match on MSCOCO image supercategories.

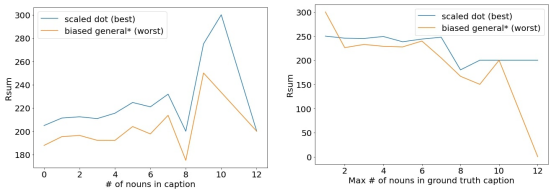

(a) Image Retrieval (b) Text Retrieval

Figure 2: Breakdown analysis of T&I Match on the number of nouns in MS-COCO captions.

which are easier to be distinguished based on consistent visual features or shapes. Visual elements under *food* and *sports* categories are difficult to be distinguished and aligned due to vastly different types of visual cues (such as shapes, colors and textures). However, for image with annotations of these challenging categories, scaled dot product still manages to outperform biased general* attention, even though by a smaller performance gain. Regarding the image retrieval task, we can observe a similar pattern. Visual element like *electronic* and *outdoor* can have consistent and common linguistic terms such as phones, microwaves, fridges and fields that can be easily distinguished in the description, therefore those images are easier to be aligned to and retrieved given a text. In this case, scaled dot product can outperform biased general* attention by a large margin. However, for other image supercategories such as *food*, *sports* and *person*, the description can be very different due to subjectivity and variety of phrasing choices, making them more difficult to be aligned to and retrieved. Therefore

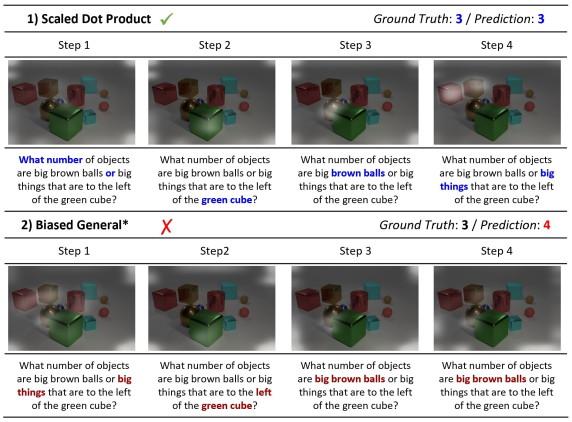

Figure 3: Qualitative example of VQA-CLEVR from the MAC trained by different attention alignment.

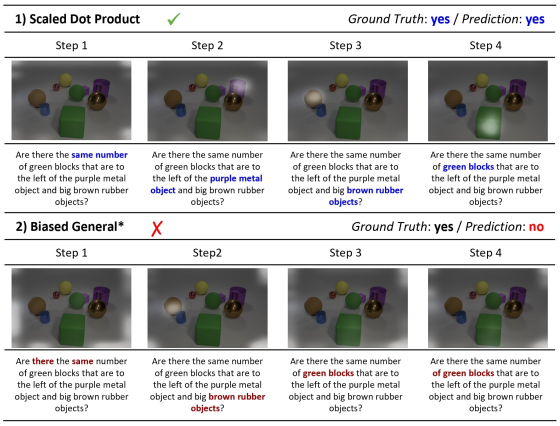

Figure 4: Qualitative example of VQA-CLEVR from the MAC trained by different attention functions.

the scaled dot product attention only manages to perform better than biased general* attention by a relatively small margin. Based on the observations above, we can conclude that the scaled dot product can perform better than biased general* attention in visual-linguistic alignment, especially for easily alignable linguistic and visual cues.

In terms of the number of nouns to be aligned in the captions, we can see from Figure 2a that scaled dot product attention can maintain a consistent performance margin over biased general* attention when retrieving images, regardless of whether there is no object, only a few objects or many objects in the caption to be aligned. However, they performed the same at 12 as there is only one caption with 12 nouns in the test set and the models cannot really be distinguished on the single instance. When we group the images based on the maximum number of objects that can be possibly contained in their descriptions, we can see from Figure 2b that scaled dot product attention can still outperform biased general* attention in most cases when retrieving relevant descriptions except for the two image query instances with the maximum of only one possible object to be aligned in the description. Based on the above patterns, we can conclude that scaled dot product can generally learn better visual-linguistic alignment than biased general* attention.

6.4 Qualitative Analysis

We visualised the prediction interpretability of the best and worst attention alignment calculation for VQA task on the question category *Count* and *Compare Integer*. More examples on other tasks can be found in Appendix A. Figure 3 shows a question asking for a count of multiple objects for the given attributes. MAC using scaled dot product attention correctly aligned to required objects at different steps. However, the model with biased general* attention focused on the *big things* before noting the condition *left of the green cube*, and failed to filter out irrelevant objects, giving a wrong prediction by aligning to additional objects. In Figure 4, MAC model using scaled dot product attention focuses on the key objects *purple metal object*, *brown rubber objects*, and *green blocks* in both question and the image, so it can successfully give the correct answer *yes*. However, the model trained with biased general* attention focused on *green blocks* in the question in the last two steps but failed to find the target in the image, thus giving a wrong prediction *no*. These examples clearly align with the finding in Section 6.3 that scaled dot product can learn better alignment than biased general* attention for questions querying multiple objects.

7 Conclusion

We systematically examined the role of attention alignment score calculation in vision-and-language tasks, including VQA, T&I Match, and T2I Gen. We found that the scaled dot product can be the best attention alignment calculation for either cross or self-attention in overall VL tasks while the appropriate position of visual and textual information may vary from different VL tasks/datasets. Based on the breakdown analysis, we found out that the type of image objects and their textual description would affect the performance of different attention calculation functions. It is hoped that our analysis provides a great insight into the selection of the most effective attention alignment for different VL tasks.

References

Peter Anderson, Xiaodong He, Chris Buehler, Damien Teney, Mark Johnson, Stephen Gould, and Lei Zhang. 2018. Bottom-up and top-down attention for image captioning and visual question answering. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 6077–6086.

Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. 2015. Neural machine translation by jointly learning to align and translate. In *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015*.

Ali Furkan Biten, Ruben Tito, Andres Mafla, Lluis Gomez, Marçal Rusinol, Minesh Mathew, CV Jawahar, Ernest Valveny, and Dimosthenis Karatzas. 2019a. Icdar 2019 competition on scene text visual question answering. In *2019 International Conference on Document Analysis and Recognition (ICDAR)*, pages 1563–1570. IEEE.

Ali Furkan Biten, Ruben Tito, Andres Mafla, Lluis Gomez, Marçal Rusinol, Ernest Valveny, CV Jawahar, and Dimosthenis Karatzas. 2019b. Scene text visual question answering. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 4291–4301.

Hui Chen, Guiguang Ding, Xudong Liu, Zijia Lin, Ji Liu, and Jungong Han. 2020. Imram: Iterative matching with recurrent attention memory for cross-modal image-text retrieval. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 12655–12663.

Tianlang Chen and Jiebo Luo. 2020. Expressing objects just like words: Recurrent visual embedding for image-text matching. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pages 10583–10590.

Haiwen Diao, Ying Zhang, Lin Ma, and Huchuan Lu. 2021. Similarity reasoning and filtration for image-text matching. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 35, pages 1218–1226.

Xinfeng Dong, Huaxiang Zhang, Xiao Dong, and Xu Lu. 2021. Iterative graph attention memory network for cross-modal retrieval. *Knowledge-Based Systems*, 226:107138.

Hongliang Fei, Tan Yu, and Ping Li. 2021. Cross-lingual cross-modal pretraining for multimodal retrieval. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 3644–3650.

Peng Gao, Zhengkai Jiang, Haoxuan You, Pan Lu, Steven CH Hoi, Xiaogang Wang, and Hongsheng Li. 2019. Dynamic fusion with intra-and inter-modality attention flow for visual question answering. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 6639–6648.

Wenya Guo, Ying Zhang, Jufeng Yang, and Xiaojie Yuan. 2021. Re-attention for visual question answering. *IEEE Transactions on Image Processing*, 30:6730–6743.

Caren Han, Siqu Long, Siwen Luo, Kunze Wang, and Josiah Poon. 2020. VICTR: Visual information captured text representation for text-to-vision multimodal tasks. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 3107–3117.

Yicong Hong, Qi Wu, Yuankai Qi, Cristian Rodriguez-Opazo, and Stephen Gould. 2021. A recurrent vision-and-language bert for navigation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1643–1653.

Ronghang Hu, Amanpreet Singh, Trevor Darrell, and Marcus Rohrbach. 2020. Iterative answer prediction with pointer-augmented multimodal transformers for textvqa. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 9992–10002.

Xiaowei Hu, Xi Yin, Kevin Lin, Lei Zhang, Jianfeng Gao, Lijuan Wang, and Zicheng Liu. 2021. Vivo: Visual vocabulary pre-training for novel object captioning. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 35, pages 1575–1583.

Qingbao Huang, Jielong Wei, Yi Cai, Changmeng Zheng, Junying Chen, Ho-fung Leung, and Qing Li. 2020. Aligned dual channel graph convolutional network for visual question answering. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7166–7176.

Drew A Hudson and Christopher D Manning. 2018. Compositional attention networks for machine reasoning. In *International Conference on Learning Representations*.

Ivan Krasin, Tom Duerig, Neil Alldrin, Vittorio Ferrari, Sami Abu-El-Haija, Alina Kuznetsova, Hassan Rom, Jasper Uijlings, Stefan Popov, Andreas Veit, et al. 2017. Openimages: A public dataset for large-scale multi-label and multi-class image classification. *Dataset available from https://github. com/openimages*, 2(3):18.

Kuang-Huei Lee, Xi Chen, Gang Hua, Houdong Hu, and Xiaodong He. 2018. Stacked cross attention for image-text matching. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 201–216.

Bowen Li, Xiaojuan Qi, Thomas Lukasiewicz, and Philip HS Torr. 2020a. Manigan: Text-guided image manipulation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 7880–7889.

Gen Li, Nan Duan, Yuejian Fang, Ming Gong, and Daxin Jiang. 2020b. Unicoder-vl: A universal encoder for vision and language by cross-modal pre-training. In *Proceedings of the AAAI Conference*

on Artificial Intelligence, volume 34, pages 11336–11344.

Wenbo Li, Pengchuan Zhang, Lei Zhang, Qiuyuan Huang, Xiaodong He, Siwei Lyu, and Jianfeng Gao. 2019. Object-driven text-to-image synthesis via adversarial training. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 12174–12182.

Chunxiao Liu, Zhendong Mao, An-An Liu, Tianzhu Zhang, Bin Wang, and Yongdong Zhang. 2019. Focus your attention: A bidirectional focal attention network for image-text matching. In *Proceedings of the 27th ACM International Conference on Multimedia*, pages 3–11.

Chunxiao Liu, Zhendong Mao, Tianzhu Zhang, Hongtao Xie, Bin Wang, and Yongdong Zhang. 2020. Graph structured network for image-text matching. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 10921–10930.

Siqu Long, Feiqi Cao, Soyeon Caren Han, and Haiqin Yang. 2022a. Vision-and-language pretrained models: A survey. In *Proceedings of the Thirty-First International Joint Conference on Artificial Intelligence, IJCAI-22*, pages 5530–5537. International Joint Conferences on Artificial Intelligence Organization. Survey Track.

Siqu Long, Soyeon Caren Han, Xiaojun Wan, and Josiah Poon. 2022b. Gradual: Graph-based dual-modal representation for image-text matching. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision (WACV)*, pages 3459–3468.

Jiasen Lu, Dhruv Batra, Devi Parikh, and Stefan Lee. 2019. Vilbert: pretraining task-agnostic visiolinguistic representations for vision-and-language tasks. In *Proceedings of the 33rd International Conference on Neural Information Processing Systems*, pages 13–23.

Siwen Luo, Hamish Ivison, Caren Han, and Josiah Poon. 2021. Local interpretations for explainable natural language processing: A survey. *arXiv preprint arXiv:2103.11072*.

Thang Luong, Hieu Pham, and Christopher D. Manning. 2015a. Effective approaches to attention-based neural machine translation. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 1412–1421, Lisbon, Portugal. Association for Computational Linguistics.

Thang Luong, Hieu Pham, and Christopher D Manning. 2015b. Effective approaches to attention-based neural machine translation. In *EMNLP*.

Dehong Ma, Sujian Li, Xiaodong Zhang, and Houfeng Wang. 2017. Interactive attention networks for aspect-level sentiment classification. In *Proceedings of the 26th International Joint Conference on Artificial Intelligence*, IJCAI'17, page 4068–4074. AAAI Press.

Sharad Pande, Srishti Chouhan, Ritesh Sonavane, Rahee Walambe, George Ghinea, and Ketan Kotecha. 2021. Development and deployment of a generative model-based framework for text to photorealistic image generation. *Neurocomputing*.

Badri Patro and Vinay P Namboodiri. 2018. Differential attention for visual question answering. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 7680–7688.

Tingting Qiao, Jing Zhang, Duanqing Xu, and Dacheng Tao. 2019a. Learn, imagine and create: Text-to-image generation from prior knowledge. *Advances in Neural Information Processing Systems*, 32:887–897.

Tingting Qiao, Jing Zhang, Duanqing Xu, and Dacheng Tao. 2019b. Mirrorgan: Learning text-to-image generation by redescription. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 1505–1514.

Tanzila Rahman, Shih-Han Chou, Leonid Sigal, and Giuseppe Carenini. 2021. An improved attention for visual question answering. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 1653–1662.

Kevin J Shih, Saurabh Singh, and Derek Hoiem. 2016. Where to look: Focus regions for visual question answering. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 4613–4621.

Amanpreet Singh, Vivek Natarajan, Meet Shah, Yu Jiang, Xinlei Chen, Dhruv Batra, Devi Parikh, and Marcus Rohrbach. 2019. Towards vqa models that can read. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 8317–8326.

Alessandro Sordoni, Philip Bachman, and Yoshua Bengio. 2016. Iterative alternating neural attention for machine reading. *ArXiv*, abs/1606.02245.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Ł ukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc.

Xinyu Wang, Yuliang Liu, Chunhua Shen, Chun Chet Ng, Canjie Luo, Lianwen Jin, Chee Seng Chan, Anton van den Hengel, and Liangwei Wang. 2020. On the general value of evidence, and bilingual scene-text visual question answering. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 10126–10135.

Zihao Wang, Xihui Liu, Hongsheng Li, Lu Sheng, Junjie Yan, Xiaogang Wang, and Jing Shao. 2019. Camp: Cross-modal adaptive message passing for text-image retrieval. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 5764–5773.

Xi Wei, Tianzhu Zhang, Yan Li, Yongdong Zhang, and Feng Wu. 2020. Multi-modality cross attention network for image and sentence matching. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 10941–10950.

Tao Xu, Pengchuan Zhang, Qiuyuan Huang, Han Zhang, Zhe Gan, Xiaolei Huang, and Xiaodong He. 2018a. Attngan: Fine-grained text to image generation with attentional generative adversarial networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1316–1324.

Tao Xu, Pengchuan Zhang, Han Zhang Qiuyuan Huang, Zhe Gan, Xiaolei Huang, and Xiaodong He. 2018b. Attngan: Fine-grained text to image generation with attentional generative adversarial networks.

Zhengyuan Yang, Yijuan Lu, Jianfeng Wang, Xi Yin, Dinei Florencio, Lijuan Wang, Cha Zhang, Lei Zhang, and Jiebo Luo. 2021. Tap: Text-aware pre-training for text-vqa and text-caption. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 8751–8761.

Zichao Yang, Xiaodong He, Jianfeng Gao, Li Deng, and Alex Smola. 2016. Stacked attention networks for image question answering. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 21–29.

Guojun Yin, Bin Liu, Lu Sheng, Nenghai Yu, Xiaogang Wang, and Jing Shao. 2019. Semantics disentangling for text-to-image generation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 2327–2336.

Zhou Yu, Jun Yu, Yuhao Cui, Dacheng Tao, and Qi Tian. 2019. Deep modular co-attention networks for visual question answering. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 6281–6290.

Han Zhang, Jing Yu Koh, Jason Baldridge, Honglak Lee, and Yinfei Yang. 2021a. Cross-modal contrastive learning for text-to-image generation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 833–842.

Weifeng Zhang, Jing Yu, Wenhong Zhao, and Chuan Ran. 2021b. Dmrfnet: Deep multimodal reasoning and fusion for visual question answering and explanation generation. *Information Fusion*, 72:70–79.

Minfeng Zhu, Pingbo Pan, Wei Chen, and Yi Yang. 2019. Dm-gan: Dynamic memory generative adversarial networks for text-to-image synthesis. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 5802–5810.

Yuke Zhu, Oliver Groth, Michael Bernstein, and Li Fei-Fei. 2016. Visual7w: Grounded question answering in images. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 4995–5004.

Mingchen Zhuge, Dehong Gao, Deng-Ping Fan, Linbo Jin, Ben Chen, Haoming Zhou, Minghui Qiu, and Ling Shao. 2021. Kaleido-bert: Vision-language pre-training on fashion domain. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 12647–12657.

A Appendix

In this section, we demonstrate some more attention alignment examples of best and worst performing attention methods for each task we investigated.

A.1 Additional Qualitative Examples - VQA

We include more comparison examples for MAC model in this section to show the difference between scaled dot product (best) and biased general* (worst) in the VQA context. In Figure 5, a question *what number of small metallic things are left of the brown matte object in front of the brown thing on the right side of the gray ball* is raised towards an image with several cylinders, cubes and spheres. The MAC model with scaled dot product attention is able to correctly focus on the *brown matte object* from both the question and the image, while putting slight attention on *the brown thing on the right side* as mentioned in the question. Then in step 3 and 4 the model is able to locate the *small metallic thing* on the left in the image as guided by the question context, giving a correct prediction of *1*. However, the MAC model trained with biased general* attention slightly focuses on the target metallic object at the very beginning, and shifts its main attention to the *brown matte object* in the consecutive steps, which is not the final target the question is asking for, therefore it fails to make a correct prediction.

Figure 6 shows a picture featuring several cubes and spheres. With a scaled dot product for attention score calculation, when the model focuses on the keyword *metal cubes* from the textual question, the only metal cube in the image is emphasized during the first two steps. Then, it correctly detects 4 objects from the image by highlighting the keywords *objects* and *either objects*. Additionally, the model looks for the *purple metal cube* from the picture as asked by the question, but it does not exist in the picture so none of the objects are highlighted at step 4. However, the model with the biased general attention tries to count *the number of objects* on the right of the metal cube in step 3 but it inaccurately focuses on the metal cube itself in addition to the correct ones. In the last step the model puts more focus on the farthest right objects, resulting in a wrong prediction of *3*.

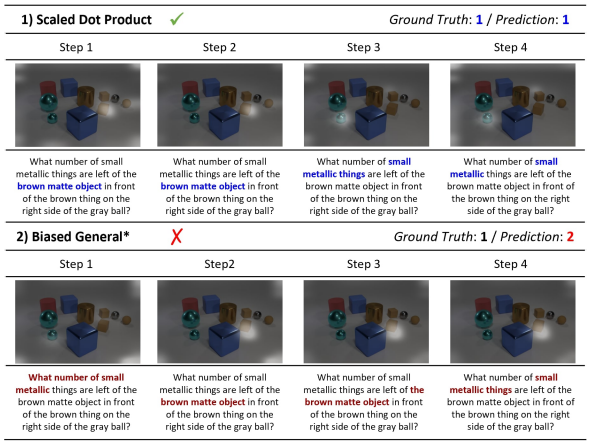

Figure 5: Qualitative examples of VQA-CLEVR from the MAC trained by different attention functions.

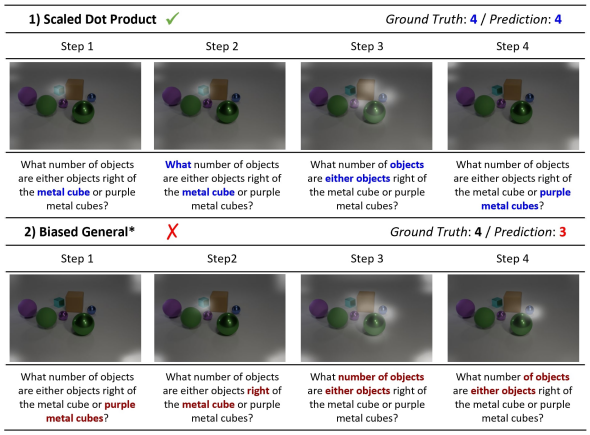

Figure 6: Qualitative examples of VQA-CLEVR from the MAC trained by different attention functions.

In Figure 7, a question *what number of objects are either gray rubber spheres or rubber things behind the green metal cylinder* is asked. MAC model using scaled dot product attention approaches this question by firstly attending to *what* and *or* in the question. Then it focuses on the relevant objects *green metal cylinder*, *gray rubber spheres*, and remaining *rubber things* in both question and the image, so it can successfully give the correct answer *3*. However, the model trained with biased general* attention firstly focused on the *number of rubber things* before noting the condition *behind the green metal cylinder*, so it failed to filter out irrelevant objects, giving a wrong prediction *4*.

A.2 Additional Qualitative Examples - TVQA

We include some qualitative examples for M4C model in this section to show the difference between scaled dot product (best) and dot product attention (worst) in the context of TVQA.

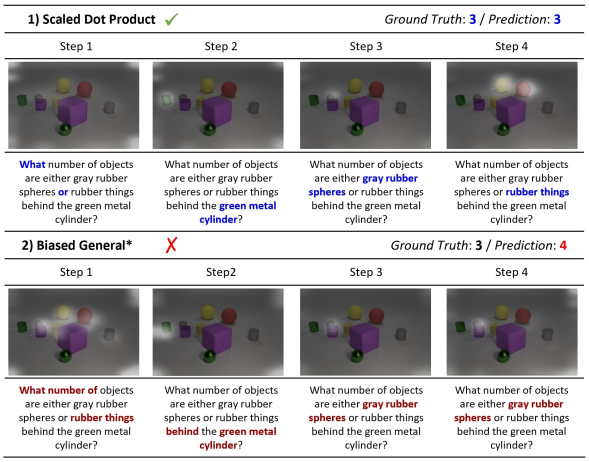

Figure 7: Qualitative examples of VQA-CLEVR from the MAC trained by different attention functions.

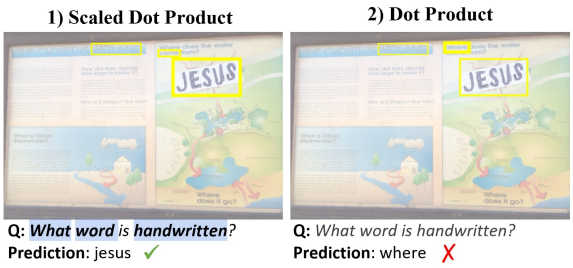

Figure 8: Qualitative examples of TVQA-TextVQA from the M4C trained by different attention functions. Question words that receive attention weights greater than 0.01 are indicated in bold and coloured in blue.

Figure 9: Qualitative examples of TVQA-TextVQA from the M4C trained by different attention functions. Question words that receive attention weights greater than 0.01 are indicated in bold and coloured in blue.

Figure 8 shows that the model with scaled dot product focused on the keywords *what*, *word* and *handwritten* to focus on the handwritten word *jesus* in the image and retrieved the correct OCR token with highest attention weight. However, with dot product attention, all the question words received little attention by the model (< 0.01), failing to find the appropriate OCR token in the image.

In Figure 9, all the three words from the question

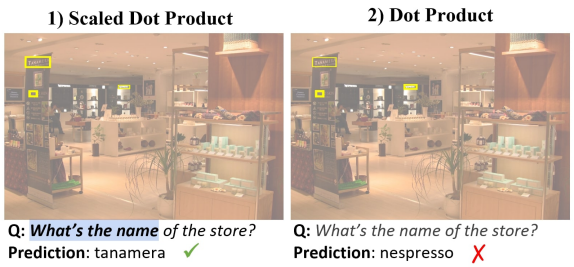

Figure 10: Qualitative examples of TVQA-TextVQA from the M4C trained by different attention functions. Question words that receive attention weights greater than 0.001 are indicated in bold and coloured in blue.

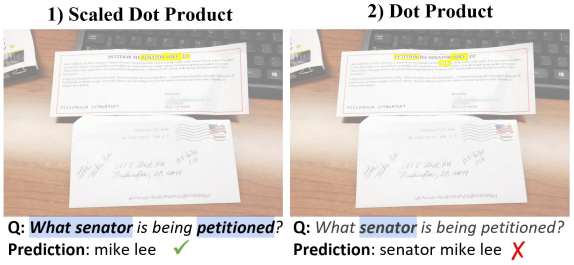

Figure 11: Qualitative examples of TVQA-TextVQA from the M4C trained by different attention functions. Question words that receive attention weights greater than 0.001 are indicated in bold and coloured in blue.

who must survive received attention > 0.01 in the scaled dot product model, and the target OCR answer in the image received top attention among all OCR tokens. However, the dot product model put much less attention on all question words, instead the OCR tokens for *must* and *survive* in the image were receiving top attention weights, followed by OCR token *winter* which is irrelevant to the question. Therefore scaled dot product model predicted correctly but dot product model did not.

In Figure 10, keywords *what's the name* from the question *what's the name of the store* received attention > 0.001 in the scaled dot product model, similarly the dot product model put much less attention on all question words, and none of the question words received attention > 0.001. Both models put most attention weights on the OCR token *gift* from the image, but scaled dot product managed to put more focus on the store name *tanamera* than the coffee brand name *nespresso*, which is the opposite case of the dot product model. Therefore scaled dot product model predicted correctly but dot product model did not.

In the example shown by Figure 11, there are lots of OCR tokens present in the image, making it more difficult to retrieve the correct answer tokens. As we can see from the picture, the model learned using dot product attention diverted its top attention to unrelated OCR token *cola*, and the top 3 OCR tokens receiving highest attention (*mike*, *cola* and *petition*) are not aligned with the predicted answer tokens (*senator mike lee*), while the model learned using scaled dot product attention put highest attention to expected or related OCR tokens that are aligned with the ground truth answers.

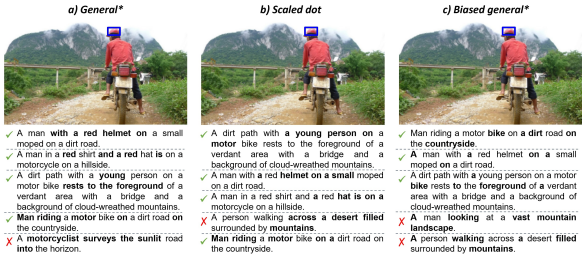

Figure 12: Qualitative examples of T&I Match-MSCOCO with SCAN by different attention functions.

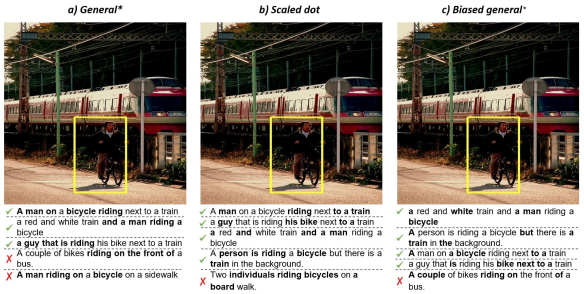

Figure 13: Qualitative examples of T&I Match-MSCOCO from the SCAN trained by different attention functions.

A.3 Additional Qualitative Examples - T&I Match

In this section, we visualize some examples for T&I Match models that show the attention received by retrieved captions with respect to the selected object region. Figure 12 shows some examples for T&I Match. With general attention* and scaled dot product, the SCAN model is able to include 4 correct captions among the top 5 retrieved captions, while biased general attention can only extract 3 correct captions. With respect to the helmet indicated by the blue box in the image, both general attention and scaled dot product can put more focus on the words *red*/*helmet*/*hat* from the caption, whereas the biased general attention would rather focus on prepositions, determinants or other objects.

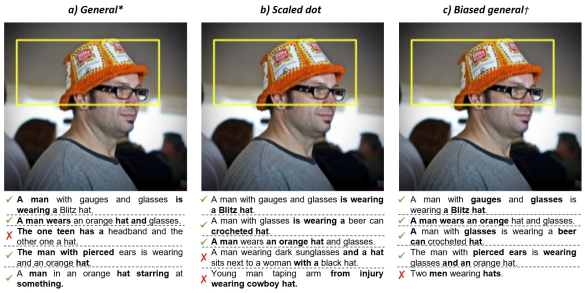

Figure 14: Qualitative examples of T&I Match-Flickr30k from the SCAN trained by different attention functions.

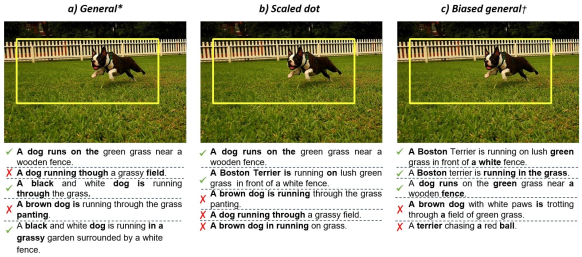

Figure 15: Qualitative examples of T&I Match-Flickr30k from the SCAN trained by different alignment functions.

In Figure 13 we can see that the best two models (i.e. models trained with general* attention or scaled dot product attention) can capture all key elements, *man, bicycle/bike, riding*, as the top attended words from the retrieved captions most of the time. However, the model trained using biased general* attention would capture at most one key element from each retrieved caption, and pay high attention to preposition, determinants or words related to other object regions. In the example shown by Figure 14, the model trained with scaled dot product attention can always capture the main object *hat* from all the retrieved captions, while the other two models sometimes fail to do so. In Figure 15, all three models sometimes wrongly recognise the dog's color (i.e. *brown dog* in wrongly retrieved captions). However, the best two models can retrieve the caption that is not in the ground truth list but also semantically matched to the given image (i.e. *A dog running through a grassy field*). The worst model trained with biased general† attention fails to do so, and it sometimes attends to objects from the caption that is not actually in the image (e.g. *red ball*).

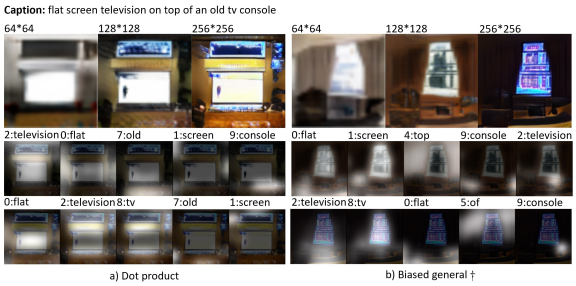

Figure 16: Qualitative examples of T2I Gen-MSCOCO with AttnGAN trained by different attention functions.

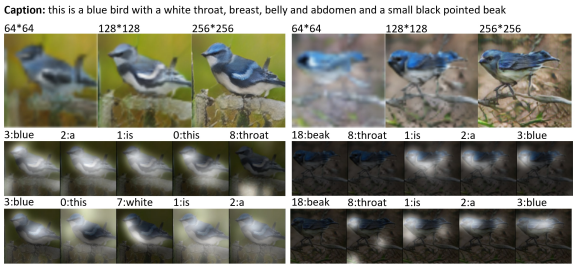

Figure 17: Qualitative examples of T2I Gen-CUB from the AttnGAN trained by different attention functions.

Figure 18: Qualitative examples of T2I Gen-CUB from the AttnGAN trained by different attention functions.

A.4 Additional Qualitative Examples - T2I Gen

In this section, we visualize and compare the best and the worst T2I Gen model.

Figure 16 shows the images generated by AttnGAN using both the best and the worst attention, dot product and biased general† respectively. AttnGAN with a dot product, can generate a relatively more realistic image. From a low resolution picture, the model focuses on the words based on the following order, *television*, *flat*, *old*, *screen*, *console*, in order to refine the image to include the objects and corresponding features gradually. Compared to that, the biased general attention model generates a surrealistic image by focusing on *flat*, *screen*, *top*, *console*, *television* in the first step.

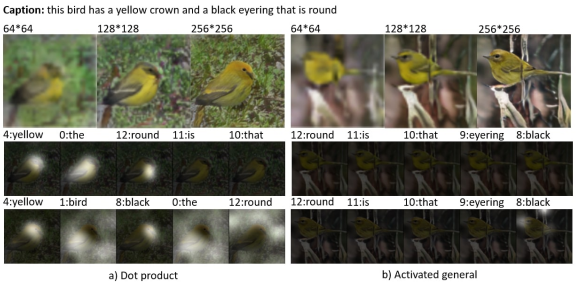

Figure 19: Qualitative examples of T2I Gen-CUB from the AttnGAN trained by different attention functions.

In Figure 17, the images generated by two models are highly similar but the worst model trained with activated general attention fails to attend to the key word *white*, so the bird it generated in the picture does not clearly have a white throat and chest.

In Figure 18 and Figure 19, the activation function used in the attention mechanism of the worst model makes it difficult to differentiate among the caption words when their attention weights are all very low. Therefore the model fails to attend to any useful facts in each attention layer, which makes it impossible to provide interpretability of model decision, despite generating an image that can roughly match the description in Figure 19. In Figure 18 the quality of the generated image is even worse - the feature of the bird does not match with the key phrases in the description (i.e. *red with white, short beak*).

Overall, based on the qualitative analysis in different VL tasks, we reveal that the better attention alignment calculation function can produce better interpretability in terms of the prediction.

Effective data augmentation for sentence classification using one VAE per class

Piedboeuf, Frédéric and **Langlais, Philippe**
DIRO, RALI
Université de Montréal
Montréal, Québec, Canada

Abstract

In recent years, data augmentation has become an important field of machine learning. While images can use simple techniques such as cropping or rotating, textual data augmentation needs more complex manipulations to ensure that the generated examples are useful. Variational auto-encoders (VAE) and its conditional variant the Conditional-VAE (CVAE) are often used to generate new textual data, both relying on a good enough training of the generator so that it doesn't create examples of the wrong class. In this paper, we explore a simpler way to use VAE for data augmentation: the training of one VAE per class. We show on several dataset sizes, as well as on four different binary classification tasks, that it outperforms other generative data augmentation techniques.

1 Introduction

Data augmentation (DA) has been shown to be an efficient technique for deep neural networks (Ramirez Rochac et al., 2019), consisting in artificially creating new labelled examples and therefore inflating the size of the dataset. While a great number of data augmentation techniques have been developed in recent years, most of these require external data, either in a database form (Wei and Zou, 2019), pretrained embeddings (Marivate and Sefara, 2020), or neural networks (Wu et al., 2019). Even if these techniques help improve classifiers, they cannot be used in a variety of contexts, most notably in artificial intelligence for rare languages, where there simply isn't enough training data to pre-train efficient models (Feldman and Coto-Solano, 2020), or where the collection of labelled data for fine-tuning is difficult. The ethical importance of considering smaller communities and other languages in machine learning research has been noted repeatedly (Bender et al., 2021; Fazelpour and De-Arteaga, 2022), and consequently, it is important to develop techniques that can work for languages other than English.

In this paper, we take a special look at generative models for data augmentation, and more specifically at Variational Auto-Encoders (VAE), for binary classification tasks. Generative algorithms such as the VAE are especially interesting because they do not require external data and, therefore, can be used on a variety of domains. In particular, we show that using one separate VAE per class and generating new data by random sampling of the latent space is a very efficient way of inflating the size of a dataset. To the best of our knowledge, this technique has been considered only once before, in the context of balancing unbalanced datasets, and with somewhat disappointing results (Qiu et al., 2020). Contrary to them, we study the technique in the context of pure data augmentation, on more datasets and various starting sizes of datasets, and conclude that it is an efficient technique. We test two variations of this method, with or without sharing parameters between the VAEs, and show that both variations are equally efficient.

We compare this technique to two others generative techniques that are commonly used in the DA literature, mainly using VAEs as a paraphrasing system, or using a Conditional VAE (CVAE) for directly generating from a given class. We show that they perform consistently worse than the separate VAE approach. We surmise that this is because, while we lose the ability to perform style transfer between classes, we gain a stronger guarantee that the generated examples will not be of an erroneous class. We furthermore compare to another technique that showed excellent results, CBERT (Wu et al., 2019), and show using one VAE per class for DA outperforms it as well.

This paper is organized as follows: Sections 2, 3, and 4 present respectively a brief summary of VAEs, the relevant literature to our paper, and the description of our various generative DA methods. Then, Section 5 and 6 present our datasets and the results of our experiments. Finally, we present a

Proceedings of the 29th International Conference on Computational Linguistics, pages 3454–3464
October 12–17, 2022.

discussion of the results in Section 7, the broader implications of our work in Section 8, and conclude in Section 9.

2 Variational Auto Encoders

VAEs are generative models introducing a latent variable $z \sim p(\mathbf{z})$, where data points (in our case, sentences) are assumed to be generated following $p(\mathbf{x}|\mathbf{z})$. VAEs follow the general auto-encoder structure, with an encoder that predicts the parameters associated with the latent distribution, most often a diagonal Gaussian, and a decoder that takes samples from the latent distribution and transforms them into sentences. At training time, sampling is done using the reparametrization technique (Kingma and Welling, 2014), which samples from $\mathbf{z} = \mu + \sigma\epsilon$, where $\epsilon \sim \mathcal{N}(0, 1)$. This allows the gradient to flow through μ and σ. The training objective is the ELBO, or Evidence Lower Bound:

$$\mathcal{L} = \mathbb{E}_{q_\phi(\mathbf{z}|\mathbf{x})}[\log p_\theta(\mathbf{x}|\mathbf{z})] - D_{KL}(q_\phi(\mathbf{z}|\mathbf{x})||p(\mathbf{z})) \quad (1)$$

where ϕ represents the parameters of the encoder, and θ, the parameters of the decoder. The CVAE is a modification of the VAE where we instead assume that $\mathbf{x}$ is generated conditioned on both the class of the example and the latent space, formally $p(\mathbf{x}|\mathbf{z}, \mathbf{c})$, where $\mathbf{c}$ is the class. In practice, this means that we give the class to the model while both training and generating, so that we can directly generate examples from the desired class (Yan et al., 2016).

In order to use the VAE for text, the standard solution is to replace both the encoder and the decoder with Recurrent Neural Networks (RNN). We show a simple example of a textual VAE model in Figure 1.

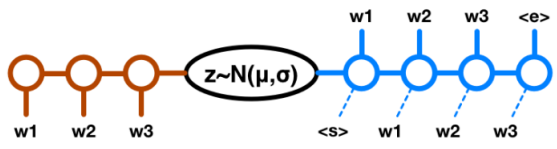

Figure 1: A textual VAE model. Image from Semeniuta et al. (2017).

This, however, often comes with the problem of KL-collapse, where the VAE relies entirely on the powerful decoder to generate sentences and collapses the latent distribution to a single point. To prevent this, we resort to two common techniques, namely KL-annealing, which slowly brings the strength of the KL-term in the ELBO from 0 to 1, and word dropout, which randomly masks words for the decoder while training (Bowman et al., 2016). Both of these techniques encourage the VAE to rely on the latent distribution.

3 Related Work

Data augmentation has been studied extensively (Feng et al., 2021; Shorten and Khoshgoftaar, 2019). We focus our review on studies targeting data augmentation for sentence classification, as well as works using the VAE for DA, which are the most pertinent studies for our work.

The general principle of data augmentation is fairly direct: starting with a dataset of fixed size, we generate new (hopefully) diverse data that we can then use to feed the classifier, diminishing generalization error. While DA for images can perform simple operations, such as cropping or rotating the images (Wang et al., 2017), things are a little more complicated for textual data, since operations on words (such as replacing or deleting words), have a chance of generating examples of the wrong class. As such, a lot of the focus is put on class-coherence, which simply means that generated examples have the class we want them to.

Many approaches have been developed for data augmentation for sentence classification, the simplest ones consisting in performing word-level operations. For example, the authors of (Marivate and Sefara, 2020) use W2V (Mikolov et al., 2013) for replacing random words in a sentence, picking words that are close to the original one in a pretrained embedding space. In (Xiang et al., 2021), replacements are made using the part of speech information to ensure they only replace content words. More operations can also be considered, as in EDA (Wei and Zou, 2019; Liesting et al., 2021), which performs four operations on the sentences: replacement by a synonym, deletion of words, swapping words, and inserting synonyms of words at random positions. In EDA, synonyms are found by using WordNet (Miller, 1995) instead of pretrained embeddings.

Substitutions can also be made using pretrained neural networks with a masked word task, such as an LSTM (Kobayashi, 2018), or conditional-BERT (CBERT) (Wu et al., 2019), which is a simple extension of BERT where we prepend the class of

the example to the sentence, in the hope that it will learn to generate class-coherent outputs.

Generative models, and most notably VAEs, are frequently used in data augmentation, often using a conditional model to directly generate from a given class (Zhuang et al., 2019; Malandrakis et al., 2019; Wang et al., 2020; Rizos et al., 2019). This, however, relies on the assumption that the CVAE is learning to correctly generate from the wanted class. While CVAEs have made tremendous progress in recent years, it remains a difficult task for textual CVAE.

The use of unconditional VAEs is also commonly explored. In Islam et al. (2021), the boundary decision is found in the latent space so that generation can be made on either side of it, depending on the class we want to generate data for. A popular technique is to use the VAE as a paraphrase machine (Mesbah et al., 2019; Malandrakis et al., 2019), encoding and decoding the data points and relying on the stochasticity of the system to generate variations of the examples. This technique, however, is also based on the hope that the algorithm will learn to generate from the same class as the input example. Finally, VAEs have also been used to generate new data that is not class dependent, such as spoken utterances for spoken language understanding (Yoo et al., 2019).

Qiu et al. (2020) attempt, similarly to our paper, to use one VAE per class for balancing data. They compare it to CVAE as well as EDA. They show, however, that generative algorithms barely outperform sampling strategies when balancing the data, on two datasets. Contrary to them, we study the use of one VAE per class in a pure data augmentation context, with already balanced classes, and show that it does work better than other algorithms. We also test all the algorithms on several dataset sizes and on four datasets, providing analysis and examples of why and how it works. Last but not least, we show that it works with modern classification algorithms (in this case, BERT), and show that it works better than state-of-the-art data augmentation.

4 VAE for data augmentation

In this paper, we revisit the technique of Qiu et al. (2020), which trains one VAE per class for generating new examples for unbalanced data, but use it for general data augmentation. We test two variants of this, one with parameter sharing (sharing of the encoder and embeddings), denoted VAE-Linked, and one where everything is separated, denoted VAE-Sep. Concretely, in VAE-Sep we initialize and train m VAEs, where m is the number of classes, and each VAE is trained only on the data of one class, that is to say on all examples $x_i|y_i \epsilon c_k$, where y_i is the class of the example x_i, and c_k is the class number k. For VAE-Linked, we initialize one VAE model with m decoders, each decoding only examples of one class. When generating, we first choose the class we want to generate from, select the corresponding VAE, and generate by randomly sampling from its latent space. This process is illustrated in Figure 2.

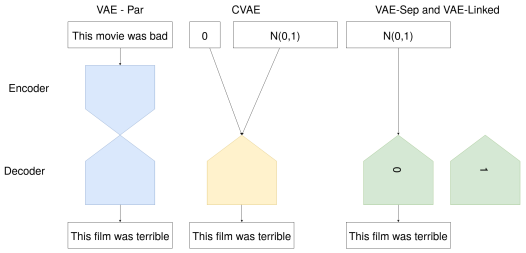

Figure 2: Illustration of the generation process of a negative movie review with the three VAE models tested in this paper. In VAE-Par, the new examples are created by passing sentences through the VAE and relying on its stochasticity to create paraphrases. In CVAE, the sentence is generated by sampling from the latent space and conditioning the decoding on the negative class. Finally, in VAE-Sep, we train two different VAEs, one for the negative sentences and one for the positive ones. We generate negative examples by sampling from the latent space and using the negative decoder. In VAE-Linked, the generation happens identically to VAE-Sep, but while training, we share the encoder and the embeddings between classes. Positive examples are generated in a similar way but passing a positive sentence, the positive label, or using the positive decoder.

We compare this to two popular DA techniques using VAEs: conditional generation (CVAE) and paraphrasing (VAE-Par). While both VAE-Sep and VAE-Linked lose the advantage of style transfer between classes and also have fewer data points to learn to generate sentences, it also greatly reduces the risk of generating examples of the wrong class.

As mentioned, generative models for textual DA are interesting due to the fact that they do not need external data to work. As such, we do not use pretrained embeddings in the RNNs of the VAEs[1]. We

[1]While outside the scope of this study, preliminary experiments with GLoVE embeddings (Pennington et al., 2014), in place of embeddings initialized randomly, did not bring a

discuss in Section 9 ideas for future research in a context where the use of external data is encouraged.

5 Datasets

We compare the models on four binary sentence classification tasks. We use movie critics classification with SST-2 (Socher et al., 2013), detection of fake news articles with the FakeNews dataset[2], detection of ironic tweets with the dataset Irony (Van Hee et al., 2018), and detection of subjective questions with Subj (Bjerva et al., 2020). Characteristics of the datasets are summarized in Table 1.

Dataset	# Ex.	Av. Sent. Length
SST-2	6920	19.3
FakeNews	12799	12.5
Irony	2683	14.3
Subj	13990	5.6

Table 1: Characteristics of the datasets used in this study. The average length represents the average number of words, when tokenized at white spaces.

6 Experiments

Because we are interested in domains where data is limited, we sample portions of the datasets to act as our initial training sets. We test 4 different datasets sizes: 500, 1000, 1500, and 2000. When sampling, we make sure to balance classes. We run each experiment 15 times and report the average results. As our baseline, we train BERT without data augmentation.[3]

We report in Table 2 the basic hyperparameters for all four variational algorithms. We use a GRU with 1 layer as our RNN for both the encoder and decoder, and a sigmoid function $\sigma(x)$ for annealing the KL-divergence from 0 to 1, which takes in the parameters x_0, controlling where $\sigma(x) = 0.5$, and a parameter k which controls the strength of the slope. While we fix x_0 at 15 (out of 30 epochs), we calculate it in practice according to the total number of batches, taking a step of annealing at every batch. Doing so allows us to keep a fixed k parameter, which makes the final KL-weight lower

$x0$	15
k	0.0025
Batch Size	32
Latent Size	15
Hidden Size	2048
Nb Epoch	30
Dropout	0.5
Word dropout	0.6
Nb layers GRU	1

Table 2: Basic hyperparameters used for all three variational algorithms

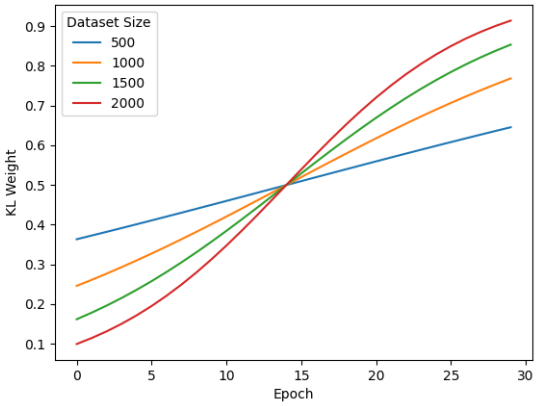

Figure 3: Annealing of the KL weight for various dataset sizes.

for a smaller dataset size (which are more prone to KL-collapse) and achieves 1 for a larger dataset size, as shown in Figure 3. We found that this setting worked well for all four sizes tested. To simulate a real world environment, we fine-tune to generate good sentences before testing on the test set.

It is to note that this might not be ideal, as it also means that the smaller dataset sizes will have a stronger KL term from the beginning, but we leave the full fine-tuning for future work. In the same vein, variational auto-encoders can be finicky to train correctly, so while the hyperparameters presented in Table 2 hold true for most experiments, we sometimes had to fine-tune it a bit further, mostly by changing the latent size.

In addition to the four generative DA algorithms mentioned in Section 4, we compare ourselves to CBERT[4], a data augmentation algorithm that showed good results on sentence classification (Wu et al., 2019). CBERT predicts masked words in

better performance to VAE-Sep. We leave the exploration of this phenomenon for future work.

[2] https://www.kaggle.com/c/fake-news/overview

[3] Code is available at https://github.com/smolPixel/DACOLING2022

[4] https://github.com/1024er/CBERT_aug

a sentence using BERT (Devlin et al., 2019). To ensure class coherence, the label is prepended to the sentence so that BERT learns to generate words of the right class. Finally, we use BERT from the jiant toolkit (Pruksachatkun et al., 2020) as our classifier.

Table 3 shows the average accuracy over the four datasets for each algorithm and starting sizes, while doubling the number of sentences[5]. Standard deviations over the 15 runs and all datasets range from 0.4 to 1.0, with CBERT and VAE-Sep achieving the lowest on average. We can also observe that VAE-Sep and VAE-Linked perform the best, surpassing the other algorithms by 0.3 points, on average. Linking the encoders and embeddings ultimately does not improve the performance over having completely separate systems. We posit that this is because the main advantage of VAE-Sep is that the separate decoders have little chance of generating examples of the wrong class, and therefore sharing the encoders is not advantageous as long as the system is good enough to help correctly train the decoder(s). We analyze this phenomenon further in Section 7. We show that even if the improvement is small in terms of performance, VAE-Sep and VAE-Linked bring a clear advantage over the other generative algorithms when looking at the number of erroneous sentences generated.

We also notice that CBERT and CVAE perform the second-best on average. While CBERT performs well when the dataset size is bigger, it underperforms on small datasets. CVAE on the other hand performs well on small dataset sizes, but badly on larger ones, similarly to the VAE-Par. Finally, we observe that the larger the initial dataset size, the harder it is to get an augmentation of performance, as noticed in (Dai and Adel, 2020).[6]

In Figure 4, we show the performances of our algorithms on the individual datasets. We see that while VAE-Sep globally outperforms other algorithms, it sometimes has more difficulties, such as for the Irony or FakeNews datasets. It is possible that this is due to the nature of the data itself, where differences between the classes come more from syntactic differences than from differences in vocabulary. This implies that chances of generating from the wrong class augment, and therefore it loses a bit of its advantage.

	500	1000	1500	2000	Aver.
Baseline	81.2	83.8	85.6	86.8	84.4
CBERT	82.2	84.5	85.8	86.9	84.9
VAE-S	**83.1**	**85.0**	**86.0**	**87.0**	**85.2**
VAE-L	83.0	84.8	85.9	**87.0**	**85.2**
VAE-P	82.2	84.0	85.3	86.3	84.5
CVAE	83.0	84.6	85.6	86.5	84.9

Table 3: Average accuracy for the four starting sizes over the four datasets. VAE-S stands for VAE-Sep, VAE-L for VAE-Linked, and VAE-P for VAE-Par. By running a multivariate t-test between the baseline results and the augmentation, we found that all presented a significant difference ($p<0.05$) except for CBERT 1500, VAE-S 1500, VAE-L 1500, CVAE 1500, and VAE-S 2000. Bold results indicate the best results for each starting sizes.

In Table 4, we show examples of generated sentences, which help give a clearer picture of the output of the algorithms. We analyze the results and the limits of each algorithms in the next section.

7 Analysis

We showed that the VAE-Sep method outperformed other methods by 0.3% globally. In this section, we analyze the algorithms and posit some hypothesis as to why it works better.

7.1 Erroneous sentences

We first take a look at the generated sentences and present the percentage which is erroneous (either has the wrong label or a human is unable to label it), as well as the percentage of sentences that are copies of genuine examples. For the former, we take the data generated for SST-2 with a starting size of 1000 (see Table 4), and manually look at 100 randomly selected examples, 50 positives and 50 negatives. For the latter, we look through all starting dataset sizes and all datasets and compute the percentage of generated sentences that are identical to a sentence in the training set (a total of 5000 sentences per algorithm). We also report various averages of length to give a better idea of the generation process, namely the length of erroneous sentences, the length of identical sentences, and the average length of generated examples.

Table 5 presents the results of our analysis. We see first that VAE-Sep, VAE-Linked, and CBERT clearly produce less errors than the other meth-

[5]For reference, the maximum accuracy obtained when training with all data and no data augmentation is of 90.9% for SST-2, 95.6% for FakeNews, 69.0% for Irony, and 99.96% for Subj.

[6]We also note that even if it has a small standard deviation, the performance of CBERT is very uneven through the datasets and dataset sizes.

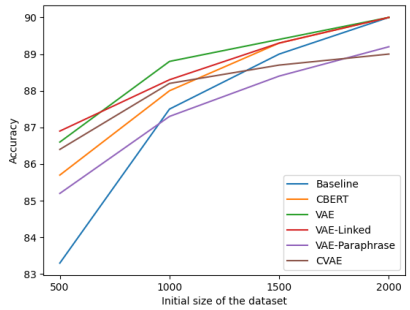

(a) SST-2

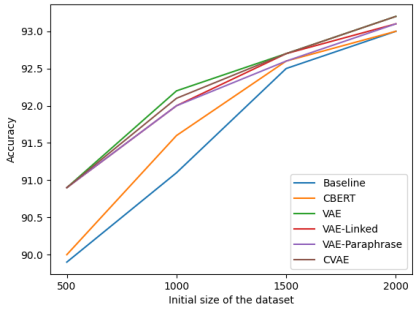

(b) FakeNews

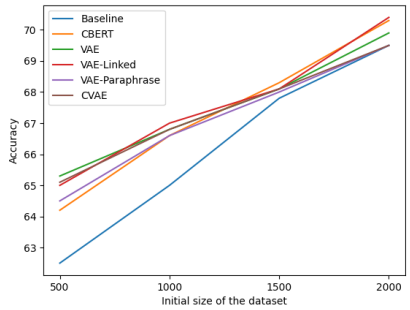

(c) Irony

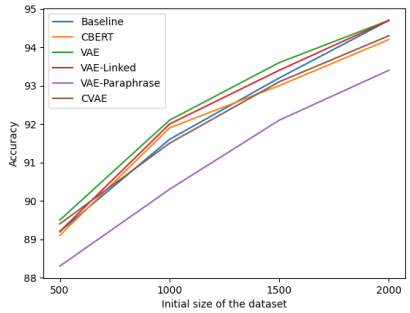

(d) Subjectivity

Figure 4: Performance of the various augmentation techniques on four dataset sizes (500, 1000, 1500, 2000) for the four datasets. Results are averaged over 15 runs.

Algo	Generated sentence	Polarity
VAE-S	a rather brilliant little cult item: a pastiche of children's entertainment, superhero comics.	Positive
	the problem with all of the characters, the characters forget their lines and no surprises.	Negative
VAE-L	it's a pleasure to watch.	Positive
	it's a frankenstein-monster of a film that doesn't know what to be.	Negative
CVAE	it is a coming-of-age, and cautionary parable, but he was a great character study.	Positive
	ultimately, the film amounts to being lectured, you's not up for the material, and offer's spiritual quest to sustain it.	Negative
VAE-P	about schmidt is nicholson's goofy, heartfelt king lear.	Positive
	the whole is a disaster, but capra are rolling in the face.	Negative
CBERT	like a tarantino movie with heart, alias betty is richly detailed, deftly executed and utterly absorbing.	Positive
	an awkwardly garish showcase that diverges from anything remotely probing or penetrating.	Negative

Table 4: Examples of generated sentences for each algorithm for the SST-2 dataset and with a dataset size of 1000.

ods, which is unsurprising due to the nature of the algorithms, and highlights the benefits of using separate decoders instead of a unique one. We note that CBERT has a very large proportion of examples that are identical to the starting examples. We find that this is a natural consequence of using a large pre-trained model. As they are trained to predict masked words, they will naturally tend to predict the correct masked words with high accuracy, and therefore often produce identi-

	VAE-S	VAE-L	CVAE	VAE-P	CBERT
Av. len	12.7	13.9	13.2	13.9	15.1
% wrong	5	6	34	27	5
Av. len	12.8	15.2	18.1	18.8	7.8
% id	1.5	6.7	1.1	2.9	43.5
Av. len	7.7	10.6	8.8	7.2	12.4

Table 5: Some statistics on the generated sentences for the five algorithms. % wrong refers to the percentage of erroneous sentences (unable to label or of the wrong class), and % id refers to the percentage of identical sentences generated by the algorithm.

Algo	Sentence
VAE-S	he's rare to be said, but it's never.
	a sleek advert youthful anomie anomie that strives for equals.
VAE-L	amazingly dopey.
	a company.
VAE-P	he doesn't take a great, and he doesn't have a very real.
	ponderous a kiss, a film is a feature film.
CVAE	an exit, while that this is that it, not entirely.
	the story is as the get-go.
CBERT	not enjoyable.
	a processed marvelous chop suey.

Table 6: Examples of erroneous sentences (pos/neg) for the SST-2 dataset with a starting dataset size of 1000.

cal examples. This also explains why their average length is much higher than for the generative algorithms, which have a tendency to produce duplicate examples when they end their generation process prematurely. CBERT, in opposition, produces duplicates as a natural consequence of its inner working. It is curious that VAE-Linked tends to produce more duplicate examples than the other generative algorithms. Most of these happens when the dataset size is small (500, 1000), and we posit that it is because using separate decoders for important characteristics (in this case, the class) makes the model more robust than a normal VAE, allowing it to have a lower reconstruction error. While this is supported by our qualitative analysis of the generated sentences of the VAE-Linked, which seems better than both generated sentences by VAE-Sep and VAE-Par, a full analysis of this phenomenon is outside the scope of this work.

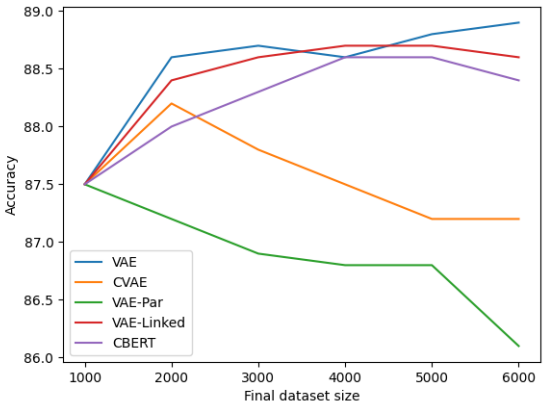

Figure 5: Accuracy vs final dataset size for a starting size of 1000 and the SST-2 task.

In Table 6, we show examples of erroneous sentences for all five algorithms. We observe that for the four generative algorithms, errors come mostly from examples that are neither positive nor negative. Rather, they are simply ill-formed. VAE-Sep and VAE-Linked create fewer of these because, as they are trained on the vocabulary of only one class, there are generally some words that pop up that allow us to determine, however tenuously, the correct class.

7.2 Number of generated sentences

Up to now, we asked the DA methods to double the size of the starting set. Here, we wonder whether generating more data would be useful, and to what extent.

Figure 5 shows for SST-2 what happens when we double, triple, etc, the size of the dataset with a starting size of 1000 sentences. The figure reveals an interesting phenomenon. None of the generative algorithms seem to benefit much from more augmentation, but CBERT does. However, the performance stops increasing at 88.6% of accuracy, which VAE-Sep reaches by simply doubling the dataset. This phenomenon is most likely directly related to the repetition of sentences produced by CBERT. In order to get the same amount of new informative sentences as VAE-Sep, the augmentation algorithm has to be applied several times. An alternative could be to introduce a filter that only accepts augmented sentences if it is not already present in the dataset, but since the focus of this study was on the VAE models and not on improving existing DA techniques, we leave this for future

work[7].

8 Implications

In this work, we show that VAE-Sep, a data augmentation algorithm that doesn't require external data, obtains a satisfying performance, even surpassing CBERT, a data augmentation algorithm using a pretrained neural network. While the VAE-Sep algorithm, at the moment, only on BERT for English sentence classification, the next step of the research is to test it on rare languages.

A large portion of NLP research focuses on English text, and even as pretrained multilingual models are allowing a good performance on multilingual data, such as XLM (CONNEAU and Lample, 2019), mBart (Liu et al., 2020), or M2M100 (Fan et al., 2021), labelled data in the target language is still needed for fine-tuning.

The VAE-Sep approach that we use in this paper does not require any form of external data, and is therefore usable for data augmentation for rare languages. We have also shown that it works well on BERT, so we expect that it would also help on multilingual pretrained transformers, most directly mBERT (Pires et al., 2019), but we leave the confirmation of this hypothesis for future work.

9 Conclusion and future work

Efficiently augmenting datasets is a central issue in machine learning. It can increase performance for all kinds of tasks, and has also been shown to help while distilling models (Kamalloo et al., 2021).

In this paper, we consider data augmentation using simple generative models based on the VAE architecture, which has the advantage that they do not need external data to work. We compare two methods used in the literature, paraphrasing and conditional generation, to another where we train one separate VAE per class. Furthermore, we compare ourselves to an established method, CBERT, which performs substitutions of random words in the sentence using a pre-trained conditionnal BERT model. We show that VAE-Sep and VAE-Linked consistently outperform other methods (CBERT, CVAE, VAE-Par), with an average gain in accuracy of 0.3%, on four binary classification tasks and four initial dataset sizes.

[7]Preliminary experiments indicate however, that it does not improve the performance of CBERT in any significant manner.

This study opens the door to interesting follow-up research. First, we contrasted generative approaches with CBERT, which showed good performance. However, due to a lack of objective comparison in the literature, we cannot guarantee CBERT delivers SOTA results. Such a comparison would help the field hone in on efficient techniques and work with a common basis. It would also be interesting to observe how VAE-Sep algorithm performs on multi-class datasets, since each VAE receives only members of one class and therefore would receive less training data on multiclass tasks.

The VAE-Sep algorithm also has possibilities of improvements. While in this study we used basic textual VAEs, there are many systems that have been developed for better VAEs that could be used. As mentioned, we take interest in VAEs because they allow data augmentation without using external data, which is useful for domains with limited data, but it would be interesting to see the performance if we allowed the usage of external data, for example by using pre-trained embeddings or even pre-trained transformers (Park and Lee, 2021; Li et al., 2020).

Finally, there is always the question of understanding why exactly data augmentation helps. While it is starting to be studied (Jha et al., 2020), we are far from fully understanding the DA process. Understanding it would not only help create stronger DA algorithms for modern deep neural networks, but furthermore help understand how these networks learn data representation.

References

Emily M. Bender, Timnit Gebru, Angelina McMillan-Major, and Shmargaret Shmitchell. 2021. On the Dangers of Stochastic Parrots: Can Language Models Be Too Big? . In *Proceedings of the 2021 ACM Conference on Fairness, Accountability, and Transparency*, pages 610–623, Virtual Event Canada. ACM.

Johannes Bjerva, Nikita Bhutani, Behzad Golshan, Wang-Chiew Tan, and Isabelle Augenstein. 2020. SubjQA: A Dataset for Subjectivity and Review Comprehension. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 5480–5494, Online. Association for Computational Linguistics.

Samuel R. Bowman, Luke Vilnis, Oriol Vinyals, Andrew Dai, Rafal Jozefowicz, and Samy Bengio. 2016. Generating Sentences from a Continuous Space. In *Proceedings of The 20th SIGNLL Conference on Computational Natural Language Learning*, pages

10–21, Berlin, Germany. Association for Computational Linguistics.

Alexis CONNEAU and Guillaume Lample. 2019. Cross-lingual Language Model Pretraining. In *Advances in Neural Information Processing Systems*, volume 32. Curran Associates, Inc.

Xiang Dai and Heike Adel. 2020. An Analysis of Simple Data Augmentation for Named Entity Recognition. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 3861–3867, Barcelona, Spain (Online). International Committee on Computational Linguistics.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, Minneapolis, Minnesota. Association for Computational Linguistics.

Angela Fan, Shruti Bhosale, Holger Schwenk, Zhiyi Ma, Ahmed El-Kishky, Siddharth Goyal, Mandeep Baines, Onur Celebi, Guillaume Wenzek, Vishrav Chaudhary, and others. 2021. Beyond english-centric multilingual machine translation. *Journal of Machine Learning Research*, 22(107):1–48.

Sina Fazelpour and Maria De-Arteaga. 2022. Diversity in sociotechnical machine learning systems. *Big Data & Society*, 9(1):20539517221082027. Publisher: SAGE Publications Ltd.

Isaac Feldman and Rolando Coto-Solano. 2020. Neural Machine Translation Models with Back-Translation for the Extremely Low-Resource Indigenous Language Bribri. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 3965–3976, Barcelona, Spain (Online). International Committee on Computational Linguistics.

Steven Y. Feng, Varun Gangal, Jason Wei, Sarath Chandar, Soroush Vosoughi, Teruko Mitamura, and Eduard Hovy. 2021. A Survey of Data Augmentation Approaches for NLP. In *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*, pages 968–988, Online. Association for Computational Linguistics.

Zubayer Islam, Mohamed Abdel-Aty, Qing Cai, and Jinghui Yuan. 2021. Crash data augmentation using variational autoencoder. *Accident Analysis & Prevention*, 151:105950.

Rohan Jha, Charles Lovering, and Ellie Pavlick. 2020. Does Data Augmentation Improve Generalization in NLP? *arXiv:2004.15012 [cs]*. ArXiv: 2004.15012.

Ehsan Kamalloo, Mehdi Rezagholizadeh, Peyman Passban, and Ali Ghodsi. 2021. Not Far Away, Not So Close: Sample Efficient Nearest Neighbour Data Augmentation via MiniMax. In *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*, pages 3522–3533, Online. Association for Computational Linguistics.

Diederik P. Kingma and Max Welling. 2014. Auto-Encoding Variational Bayes. *arXiv:1312.6114 [cs, stat]*. ArXiv: 1312.6114.

Sosuke Kobayashi. 2018. Contextual Augmentation: Data Augmentation by Words with Paradigmatic Relations. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 2 (Short Papers)*, pages 452–457, New Orleans, Louisiana. Association for Computational Linguistics.

Chunyuan Li, Xiang Gao, Yuan Li, Baolin Peng, Xiujun Li, Yizhe Zhang, and Jianfeng Gao. 2020. Optimus: Organizing Sentences via Pre-trained Modeling of a Latent Space. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 4678–4699, Online. Association for Computational Linguistics.

Tomas Liesting, Flavius Frasincar, and Maria Mihaela Truşcă. 2021. Data augmentation in a hybrid approach for aspect-based sentiment analysis. In *Proceedings of the 36th Annual ACM Symposium on Applied Computing*, SAC '21, pages 828–835, New York, NY, USA. Association for Computing Machinery.

Yinhan Liu, Jiatao Gu, Naman Goyal, Xian Li, Sergey Edunov, Marjan Ghazvininejad, Mike Lewis, and Luke Zettlemoyer. 2020. Multilingual Denoising Pre-training for Neural Machine Translation. *Transactions of the Association for Computational Linguistics*, 8:726–742. Place: Cambridge, MA Publisher: MIT Press.

Nikolaos Malandrakis, Minmin Shen, Anuj Goyal, Shuyang Gao, Abhishek Sethi, and Angeliki Metallinou. 2019. Controlled Text Generation for Data Augmentation in Intelligent Artificial Agents. In *Proceedings of the 3rd Workshop on Neural Generation and Translation*, pages 90–98, Hong Kong. Association for Computational Linguistics.

Vukosi Marivate and Tshephisho Sefara. 2020. Improving Short Text Classification Through Global Augmentation Methods. In *Machine Learning and Knowledge Extraction*, Lecture Notes in Computer Science, pages 385–399, Cham. Springer International Publishing.

Sepideh Mesbah, Jie Yang, Robert-Jan Sips, Manuel Valle Torre, Christoph Lofi, Alessandro Bozzon, and Geert-Jan Houben. 2019. Training Data Augmentation for Detecting Adverse Drug Reactions in User-Generated Content. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language*

Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP), pages 2349–2359, Hong Kong, China. Association for Computational Linguistics.

Tomas Mikolov, Kai Chen, Greg S. Corrado, and Jeffrey Dean. 2013. Efficient Estimation of Word Representations in Vector Space.

George A. Miller. 1995. WordNet: a lexical database for English. *Communications of the ACM*, 38(11):39–41.

Seongmin Park and Jihwa Lee. 2021. Finetuning Pretrained Transformers into Variational Autoencoders. In *Proceedings of the Second Workshop on Insights from Negative Results in NLP*, pages 29–35, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Jeffrey Pennington, Richard Socher, and Christopher Manning. 2014. Glove: Global Vectors for Word Representation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1532–1543, Doha, Qatar. Association for Computational Linguistics.

Telmo Pires, Eva Schlinger, and Dan Garrette. 2019. How Multilingual is Multilingual BERT? In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 4996–5001, Florence, Italy. Association for Computational Linguistics.

Yada Pruksachatkun, Phil Yeres, Haokun Liu, Jason Phang, Phu Mon Htut, Alex Wang, Ian Tenney, and Samuel R. Bowman. 2020. jiant: A Software Toolkit for Research on General-Purpose Text Understanding Models. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics: System Demonstrations*, pages 109–117, Online. Association for Computational Linguistics.

Siyuan Qiu, Binxia Xu, Jie Zhang, Yafang Wang, Xiaoyu Shen, Gerard de Melo, Chong Long, and Xiaolong Li. 2020. EasyAug: An Automatic Textual Data Augmentation Platform for Classification Tasks. In *Companion Proceedings of the Web Conference 2020*, pages 249–252. Association for Computing Machinery, New York, NY, USA.

Juan F. Ramirez Rochac, Nian Zhang, Lara Thompson, and Timothy Oladunni. 2019. A Data Augmentation-Assisted Deep Learning Model for High Dimensional and Highly Imbalanced Hyperspectral Imaging Data. In *2019 9th International Conference on Information Science and Technology (ICIST)*, pages 362–367. ISSN: 2573-3311.

Georgios Rizos, Konstantin Hemker, and Björn Schuller. 2019. Augment to Prevent: Short-Text Data Augmentation in Deep Learning for Hate-Speech Classification. In *Proceedings of the 28th ACM International Conference on Information and Knowledge Management*, CIKM '19, pages 991–1000, New York, NY, USA. Association for Computing Machinery.

Stanislau Semeniuta, Aliaksei Severyn, and Erhardt Barth. 2017. A Hybrid Convolutional Variational Autoencoder for Text Generation. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 627–637, Copenhagen, Denmark. Association for Computational Linguistics.

Connor Shorten and Taghi M. Khoshgoftaar. 2019. A survey on Image Data Augmentation for Deep Learning. *Journal of Big Data*, 6(1):60.

Richard Socher, Alex Perelygin, Jean Wu, Jason Chuang, Christopher D. Manning, Andrew Ng, and Christopher Potts. 2013. Recursive Deep Models for Semantic Compositionality Over a Sentiment Treebank. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*, pages 1631–1642, Seattle, Washington, USA. Association for Computational Linguistics.

Cynthia Van Hee, Els Lefever, and Veronique Hoste. 2018. SemEval-2018 Task 3: Irony Detection in English Tweets. In *Proceedings of The 12th International Workshop on Semantic Evaluation*, pages 39–50, New Orleans, Louisiana. Association for Computational Linguistics.

Jason Wang, Luis Perez, and others. 2017. The effectiveness of data augmentation in image classification using deep learning. *Convolutional Neural Networks Vis. Recognit*, 11:1–8.

Qian Wang, Fanlin Meng, and T. Breckon. 2020. Data Augmentation with norm-VAE for Unsupervised Domain Adaptation. *ArXiv*.

Jason Wei and Kai Zou. 2019. EDA: Easy Data Augmentation Techniques for Boosting Performance on Text Classification Tasks. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 6382–6388, Hong Kong, China. Association for Computational Linguistics.

Xing Wu, Shangwen Lv, Liangjun Zang, Jizhong Han, and Songlin Hu. 2019. Conditional BERT Contextual Augmentation. In *Computational Science – ICCS 2019*, Lecture Notes in Computer Science, pages 84–95, Cham. Springer International Publishing.

Rong Xiang, Emmanuele Chersoni, Qin Lu, Chu-Ren Huang, Wenjie Li, and Yunfei Long. 2021. Lexical data augmentation for sentiment analysis. *Journal of the Association for Information Science and Technology*, 72(11):1432–1447. _eprint: https://onlinelibrary.wiley.com/doi/pdf/10.1002/asi.24493.

Xinchen Yan, Jimei Yang, Kihyuk Sohn, and Honglak Lee. 2016. Attribute2Image: Conditional Image Generation from Visual Attributes. In Bastian Leibe, Jiri Matas, Nicu Sebe, and Max Welling, editors, *Computer Vision – ECCV 2016*, volume 9908, pages 776–791. Springer International Publishing, Cham. Series Title: Lecture Notes in Computer Science.

Kang Yoo, Youhyun Shin, and Sang-goo Lee. 2019. Data Augmentation for Spoken Language Understanding via Joint Variational Generation. *Proceedings of the AAAI Conference on Artificial Intelligence*, 33:7402–7409.

Peiye Zhuang, Alexander G. Schwing, and Oluwasanmi Koyejo. 2019. FMRI Data Augmentation Via Synthesis. In *2019 IEEE 16th International Symposium on Biomedical Imaging (ISBI 2019)*, pages 1783–1787. ISSN: 1945-8452.

NLG-METRICVERSE: An End-to-End Library for Evaluating Natural Language Generation

Giacomo Frisoni, Antonella Carbonaro, Gianluca Moro,
Andrea Zammarchi and **Marco Avagnano**
Department of Computer Science and Engineering (DISI)
University of Bologna, Via dell'Università 50, I-47522 Cesena, Italy
{giacomo.frisoni,antonella.carbonaro,gianluca.moro}@unibo.it
{andrea.zammarchi3, marco.avagnano}@studio.unibo.it

Abstract

Driven by deep learning breakthroughs, natural language generation (NLG) models have been at the center of steady progress in the last few years, with a ubiquitous task influence. However, since our ability to generate human-indistinguishable artificial text lags behind our capacity to assess it, it is paramount to develop and apply even better automatic evaluation metrics. To facilitate researchers to judge the effectiveness of their models broadly, we introduce NLG-METRICVERSE—an end-to-end open-source library for NLG evaluation based on Python. Our framework provides a living collection of NLG metrics in a unified and easy-to-use environment, supplying tools to efficiently apply, analyze, compare, and visualize them. This includes (i) the extensive support to heterogeneous automatic metrics with n-arity management, (ii) the meta-evaluation upon individual performance, metric-metric and metric-human correlations, (iii) graphical interpretations for helping humans better gain score intuitions, (iv) formal categorization and convenient documentation to accelerate metrics understanding. NLG-METRICVERSE aims to increase the comparability and replicability of NLG research, hopefully stimulating new contributions in the area. [1]

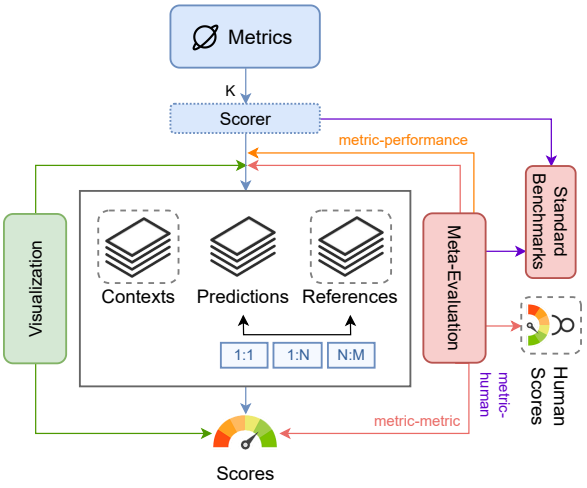

Figure 1: NLG-METRICVERSE operational representation. Dashed boxes denote optionality. A set of automatic metrics is selected to build a *Scorer* object, concurrently appliable to contexts, predictions, and references with arbitrary n-arity. A *Meta-Evaluation* module allows to inspect metrics' performance on the input data or standard benchmarks. Finally, a *Visualization* module can be applied to overcome opacity and understand metric-specific scoring processes.

1 Introduction

Natural language generation (NLG) is a sub-field of natural language processing (NLP) concerned with automatically generating human-understandable text from input data, like prompts, tables, graphs, and images. Remarkably, the ability of a machine to produce text indistinguishable from that written by humans is a pre-requisite for Artificial General Intelligence (AGI)—the holy grail of AI. Recent advancements in deep learning have yielded tremendous improvements in the NLP sector, making NLG the object of fast-growing interest from the research community, as aptly demonstrated by GPT-3 (Brown et al., 2020). Pre-trained language models with transformer-based architectures (Kalyan et al., 2021) continue to push the envelope with unprecedented performance and encourage more and more applications. Indeed, today NLG includes a wide variety of tasks, such as machine translation, single/multi-document summarization, data-to-text, text-to-text, dialogue generation, free-form question answering, and image/video captioning (Gatt and Krahmer, 2018).

As NLG models get better over time, accurately evaluating them is becoming an increasingly pressing priority for tracking progress in the area and convincingly recognizing state-of-the-art systems. However, the assessment of NLG model output is notoriously a challenging problem (Howcroft

[1] The code is publicly available at https://github.com/disi-unibo-nlp/nlg-metricverse

Proceedings of the 29th International Conference on Computational Linguistics, pages 3465–3479
October 12–17, 2022.

et al., 2020; Novikova et al., 2017). It involves the consideration of multiple intrinsic quality dimensions (e.g., informativeness, fluency, coherence, adequacy) and open-ended scenarios, where different plausible or equal-meaning responses may exist for the same user input. Human evaluation is typically regarded as the gold standard. Nevertheless, designing crowdsourcing experiments accompanied by elaborated guidelines is an expensive and high-latency process, which does not easily fit in a daily model development pipeline with the need for automatic benchmarking and tuning at scale. Furthermore, as NLG models improve, evaluators are asked to read longer passages of text conditioned on large amounts of context. In these cases, errors are often content-based (e.g., factual inaccuracies or context inconsistencies) rather than fluency-based, making superficial reads and non-expert annotators insufficient (Clark et al., 2021).

Given these issues, NLG researchers have settled for automatic evaluation metrics computing a holistic or dimension-specific score, an acceptable proxy for effectiveness and efficiency. Unfortunately, despite the rapid surge of machine-generated language, evaluation metrics have fallen behind, leaning on the conservative use of surface-level lexical similarities, which fail to cope with diversity and capture the text's underlying meaning. To overcome this severe bottleneck, the community has witnessed—in a relatively short time—a prolific, variegated, and original research production. New NLG metrics are constantly being proposed in top conferences, exhibiting one or more of the following characteristics: (i) use of contextualized word embeddings (Zhang et al., 2020), (ii) pre-training on massive unlabeled corpora (Sellam et al., 2020), (iii) fine-tuning on data annotated with human judgments (Kane et al., 2020), (iv) management of task-specific nuances (Dhingra et al., 2019; Wang et al., 2020).

Per contra, NLG metrics today are often designed and implemented from scratch with distinct environments, assumptions, properties, settings, benchmarks, and features. Such heterogeneity and disgregation make them difficult to compare or move to slightly different contexts. Concretely, the absence of a collective and continuously updated repository—well-documented and covering the entire NLG evaluation pipeline—discourages the use of modern solutions and slows down their understanding and practical application. Such barrier is highlighted also by the latest surveys (Sai et al., 2022). In the quest to fill this gap, we present NLG-METRICVERSE[2], an open-source (MIT licensed) end-to-end library for NLG evaluation, devised to provide a shared and collaborative codebase for fast application, analysis, comparison, visualization, and prototyping of automatic metrics.

The rest of the paper is organized as follows. First, we enumerate the design principles at the basis of NLG-METRICVERSE (§3), clarify the context, and summarize prior work related to this project (§2). Then, we describe the overarching NLG evaluation framework that constitutes the conceptual foundation for our contributions (§4). Next, we examine the main modules of the library: metrics, meta-evaluation, and visualization (§5). Lastly, we close the discussion and point out possible extensions (§7).

2 Background and Related Work

Early lexical NLG metrics, such as the BLEU (Papineni et al., 2002) and ROUGE (Lin, 2004), still appear to dominate the landscape, waiting for feasible, robust, and widely-adopted alternatives. Despite the high number of criticisms and studies proving their poor correlation with human judgment (Zhang et al., 2004), the popularity of first-generation metrics has not declined but expanded with the emergence of deep neural networks and new tasks. Simplicity, consistency, unsupervision, lightweight, and fast computation are the central basis of this success.

However, it has become increasingly clear that such adoption is often not prudent. Metrics measuring surface-level overlap are unsuitable for advanced evaluation, especially for modern text generation systems trained on mammoth data and with impressive paraphrasing capabilities (Mathur et al., 2020)—where ideal metrics should be sensitive to the underlying semantics. As a remedy, NLG researchers have started injecting learned/learnable components into their metrics, moving from a discrete space of word tokens to a continuous high-dimensional space of word vectors, thereby capturing distributional semantics. Over the years, many strong NLG evaluation metrics have been proposed, particularly transformer-based, like BLEURT (Sel-

[2]We coin the term "Metricverse" to denote the microcosm of automatic evaluation metrics powered by the overt ongoing rise of NLG models. According to this metaphor, we see metrics as planets belonging to galaxies and superclusters according to the taxonomy presented in Section 4.

lam et al., 2020), BERTScore (Zhang et al., 2020), and BARTScore (Yuan et al., 2021).

The trend towards the definition of model-based metrics and the resolution of task-specific needs have created a fertile ground for research. According to Sai et al. (2022), from 2002 (when BLEU was proposed) to 2014 (when Deep Learning became prevalent), there were only about 10 automatic NLG evaluation metrics in use; since 2015, at least 36 new metrics have appeared. On the other side, metrics are often scattered online, non-maintained, undocumented, implemented in various languages, inconsistent with the paper results. This not only hampers reproducibility but also inhibits scalability, as each research paper ends up creating its own implementation almost from scratch. Some libraries have already tried to make an integrated environment. To our best knowledge, NLGEval (Sharma et al., 2017), HugginFace Datasets (Lhoest et al., 2021), Evalaute[3], TorchMetrics (Detlefsen et al., 2022), and Jury (Cavusoglu et al., 2022) are the only resources available. However, none of them possess all the properties listed below: (i) large number of heterogeneous NLG metrics, (ii) concurrent computation of more metrics at once, (iii) support for multiple references and/or predictions, (iv) meta-evaluation, and (v) visualization. Table 1 summarizes the discrepancies between NLG-METRICVERSE and related work.

3 Design Principles

NLG-METRICVERSE has been designed with five main principles in mind, which, we argue, can help researchers and practitioners in a number of ways.

Comprehensiveness Given the impressive pace at which the field is growing, comprehensiveness is imperative, with the ultimate goal of providing a unique, smooth, and up-to-date access point to all the most relevant NLG evaluation metrics disseminated in different streams of literature. We also comprise organization and consistency across the library, with a coherent interaction between modules and sub-modules. This principle revolves around consolidating an all-in-one community-driven library, integrating ready-to-use n-gram- and embedding-based metrics—supervised and unsupervised, trained and untrained, reference- and statistics-based, task-specific and general-purpose, sentence- and document-level. From this synergy, we hope to spur the adoption of newly proposed contributions, unleashing their potential and concretizing the view of Sellam et al. (2020), according to which *"Machine Learning (ML) engineers should enrich their evaluation toolkits with more flexible, semantic-level metrics"*.

Ease-of-use The focus on simplicity is another key factor in fostering impact and usability, allowing users to write less code, reduce errors, and prototype faster. It is also meant to minimize the implementational burden and quickly move from papers to practical applications. We concentrate our efforts on designing an intuitive Application Programming Interface (API) accompanied by rich documentation with a curated list of executable notebooks and examples. This makes the software useful for both academia and industry.

Reproducibility Reproducibility is a core concept of utmost concern in ML and NLP, a prerequisite to trustworthiness. NLG evaluation exacerbates the problem even more, with well-known plagues like heavy undocumented preprocessing pipelines, non-transparent dataset selections, and concealed parameter settings (Post, 2018; Gao et al., 2021; Chen et al., 2022). A critical design objective of NLG-METRICVERSE is permitting experimental evaluation results to be seamlessly reproduced, promoting a fully detailed specification. In this way, users can simply integrate their original research into the shared codebase and fairly compare their solution with the existing literature. Besides serving for sound and consistent scientific research, reproducibility is a means to speed up the development of new metrics. When it comes to model-based metrics, transparency also applies to hardware setup, runtime measures, and CO2 impact.

Modularity In NLG-METRICVERSE, simplicity is sometimes bent in favor of modularity and reusability. This principle is essential for ensuring scalability and collaboratively bringing the codebase to maturity. An emphasis on module independence is maintained to guarantee the stand-alone usability of individual module functionalities and facilitate the learning of each library component.

Education One more principle is taking charge of an educational role. NLG-METRICVERSE is ideally suited to non-expert users, helping to sharpen their understanding. We believe that it is indispensable to democratize the field and gain greater

[3] https://github.com/huggingface/evaluate

	NLG-Metricverse	NLGEval	Datasets	Evaluate	TorchMetrics	Jury
#NLG-specific Metrics	38 + Datasets	8	22	22	13	19 + Datasets
More metrics at once	✓	×	×	✓	×	✓
Multiple refs/preds	✓	✓	×	×	×	✓
Meta-evaluation	✓	×	×	×	×	×
Visualization	✓	×	×	×	×	×

Table 1: Comparison of our library (v1.0.0) with existing NLG evaluation packages: NLGEval (v2.3.0), Datasets (v2.4.0), Evaluate (v0.2.2), TorchMetrics (v0.8.2), Jury (v2.2). "+ Datasets" stands for an automatic fallback towards HuggingFace Datasets in case of unsupported metrics (lower bound).

awareness of how metrics work. To unlock a teaching potential, our contributions want to include the release of standardized and content-rich metric cards, other than visualization tools conceived to aid unprecedented levels of score interpretation.

4 Framework

NLG-METRICVERSE is implemented as a Python library that provides a wrapper around a panoply of NLG evaluation metrics and complementary needs.

Regardless of the task, an NLG model generally produces one or more predictions (i.e., hypotheses, candidates) $p = p_1, \ldots, p_k$ conditioned on a given context or source $c = c_1, \ldots, c_p$. Then, one or multiple human-created references (i.e., ground-truths) $r = r_1, \ldots, r_l$ *may* be provided to assist the evaluation. In Table 2, we list sample contexts, predictions, and references for common NLG tasks to which NLG-METRICVERSE can be applied.

NLG Task	Context	Pred/Ref
Machine Translation	Source language sentence	Translation
Document Summarization	Document(s)	Summary
Data-to-Text	(Semi-)structured data, e.g., graphs, tables	Verbalization
Dialogue Generation	Conversation history	Response
Question Answering	Question (+ context)	Answer
Question Generation	Passage / Image / Knowledge Base	Question
Image/Video Captioning	Image / Video	Caption
Text Completion	Prompt	Continuation

Table 2: Popular NLG tasks settings.

Set these premises, NLG automatic evaluation metrics can be distinguished according to several overlapping criteria. To further dig into these distinctions, we lay out a taxonomy (Figure 2) serving as a foundation for experts and the broader public to build a shared overview of the possible solutions and their characteristics. Metrics can be broadly categorized based on the input format and data availability. *Context-free metrics* do not consider the context while judging the appropriateness of the prediction, typically being *task-agnostic* and adaptable to a wide variety of NLG

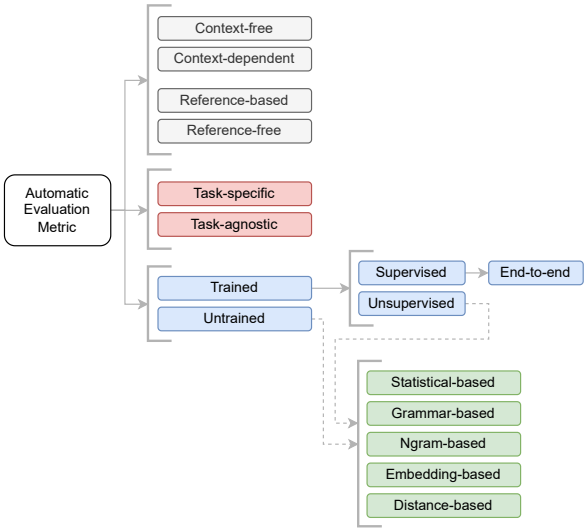

Figure 2: Taxonomy of automatic evaluation metrics. Different color nodes represent partially overlapping classification criteria (i.e., orthogonal categories).

tasks. On the flip side, *context-dependent* metrics take into account the context and are consequently *task-specific*. *Reference-based* metrics evaluate generated text with respect to one or a small set of reference text samples. *Reference-free* metrics do not rely on gold-standard references and are mainly statistics-based (e.g., full sequence distribution comparison). Further, they are suitable for an open-ended generation where there typically are several plausible continuations for each context, and creative generations are desirable; popular examples are Perplexity (Jelinek et al., 1977) and MAUVE (Pillutla et al., 2021). Finally, metrics can be classified according to their techniques. Metrics can have learnable components (*trained*) or not (*untrained*). In the first case, metrics can exploit human annotation data (*supervised*)—even with end-to-end architectures—or being human judgment-free (*unsupervised*). By end-to-end supervised metrics, we mean model-based NLG metrics trained on human-annotated data to directly output evaluation scores without additional techniques based

on learned representations and placed outside the backpropagation process. They typically refer to solutions based on regression, ranking, and classification tasks (e.g., COMET (Rei et al., 2020), FactCC (Kryscinski et al., 2020), BLEURT (Sellam et al., 2020), NUBIA (Kane et al., 2020)). Untrained and unsupervised metrics use a fixed set of heuristics and input features, such as n-gram overlapping, edit distance, static or contextualized embeddings. In this context, grammar-based measures do not rely on ground-truth references and try to quantify aspects like readability (i.e., the ease with which a reader can understand a passage) and grammaticality. To provide a concrete example, BERTScore is a context-free, reference-based, trained and unsupervised metric.

5 Main Modules

NLG-METRICVERSE is organized into three main modules: Metrics (§5.1), Meta-Evaluation (§5.2), and Visualization (§5.3). The library is intended to be a continuous and collaborative project, extended as new solutions become available. In what follows, we describe the features provided at the current stage of development. Figure 1 shows the operational representation of the modules and their interplay within the framework detailed in §4. NLG-METRICVERSE is in turn built on top of open-source libraries, including Datasets (Lhoest et al., 2021), NumPy (van der Walt et al., 2011), SciPy (Virtanen et al., 2020), and Matplotlib (Hunter, 2007). Where possible, metrics are implemented using canonical repositories released by authors.

5.1 Metrics

To construct a full-scale NLG evaluation library, the selection methodology is crucial to collect metrics with desired properties. We concentrate on four factors. (i) *Diverse classes, supervision constraints, and evaluation tasks*, as defined in §4. NLG is a versatile field; the input/output scenarios and evaluation strategies can vary from case to case. Sometimes, the predicted text is short and accompanied by human target references; other times, diversity is preferred; still different times, the generation is open-ended, long, and without references. (ii) *Diverse application tasks*. Metrics can apply to multiple NLG evaluation tasks or manage task-specific quality needs. Hence, we include a broad spectrum of real-world tasks to boost the relevance of our library. (iii) *Eval dimension*. Evaluation can be done by assessing different quality perspectives. Most existing metrics cover a small subset of these axes. Still, some of them—particularly the trainable ones—can handle several dimensions by requiring to maximize correlation with each type of judgment separately (Rei et al., 2020) or not (Yuan et al., 2021). (iv) *Popularity*. We give priority to the metrics prominently used in NLG research. Currently, 34 metrics are supported (see §A.1 for details); more solutions are under development. We tried to cover a balanced mixture of metrics and paid importance not to overweight a specific class. Future contributions can easily be integrated into NLG-METRICVERSE. We ensure the integrity of each metric within the codebase through automated tests.

Input Format We design a unified metric input type, also handling n-arity for candidate and reference texts (Table 3)—a feature as vital as neglected by current systems. In fact, there may exist multiple equally good outputs for the given input, and comparing against one gold reference can be erroneous. An extensive set of out-of-the-box data loaders takes the responsibility of processing the raw data from files and directories.

Cardinality	Syntax
1:1	preds = $[p_1, \ldots, p_k]$, refs = $[r_1, \ldots, r_k]$
1:N	preds = $[p_1, \ldots, p_k]$ refs = $[[r_{11}, \ldots, r_{1n}], \ldots, [r_{k1}, \ldots, r_{kn}]]$
N:M	preds = $[[p_{11}, \ldots, p_{1n}], \ldots, [p_{k1}, \ldots, p_{kn}]]$ refs = $[[r_{11}, \ldots, r_{1m}], \ldots, [r_{k1}, \ldots, r_{km}]]$
Preds only	preds = $[p_1, \ldots, p_k]$

Table 3: Prediction-reference input formats.

Metrics Application Evaluating artificial text requires just two lines of code: (i) create a *Scorer* object with the desired metrics; (ii) apply the *Scorer* object to the input data. So, many metrics may be executed in one go. During step (ii), the proper strategy for computing each metric is automatically selected depending on the recognized input format. If a prediction needs to be compared against multiple references, the user is left with the possibility to specify the aggregation strategy of preference through the `reduce_fn` parameter. For example, `reduce_fn="max"` considers only the prediction-reference pair with the highest score for each dataset instance. Inherently, NLG-METRICVERSE allows all NumPy function names and custom aggregation functions as well. An asynchronous execution with a separate process for each

metric can be specified to push efficiency and scalability (`run_concurrent`), bringing parallelism to the evaluation loop. Additionally, to contain the library size, we do not directly include all the packages required for running every supported metric, but we invite the user to install them if necessary. Figure 3 provides a practical example.

```
scorer =
 ↪  NLGMetricverse(metrics=["bertscore",
 ↪  "bartscore"], run_concurrent=True)
score = scorer(preds, refs) # reduce_fn
```

Figure 3: Definition and application of a *Scorer* object for the concurrent evaluation of multiple metrics.

By employing the `load_metric()` function for step (i), NLG-METRICVERSE falls back to the Datasets implementation in case of metrics not yet supported. Consequently, our library englobes at least any metrics that the Datasets package has. When defining the *Scorer*, a maximum degree of freedom is retained to allow the setting of metric-specific hyperparameters and different instantiations of the same metric (Figure 4). Further, since metrics generally involve several hyperparameters and results can deviate significantly for other choices, we accompany the output with a config report (hyperparams setting, hardware setup, etc.) for increasing comparability and replicability.

The *Scorer* application is meant to return a dictionary containing each metric's score(s), together with tracked performance metadata, including the computation time and CO2 emissions (measured with `codecarbon` (Schmidt et al., 2021)).

```
metrics = [
    load_metric("bleu",
     ↪  resulting_name="bleu_1",
     ↪  compute_kwargs={"max_order": 1}),
    load_metric("bleu",
     ↪  resulting_name="bleu_2",
     ↪  compute_kwargs={"max_order": 2}),
    load_metric("rouge")]
scorer = NLGMetricverse(metrics=metrics)
```

Figure 4: Definition and application of a *Scorer* object through the `load_metric()` function, encompassing two versions of BLEU with distinct hyperparameters.

Metric Documentation and Search NLP practitioners typically use automated metrics with a specific goal in mind, whether they are looking to answer a research question or develop a practical application system. To that end, they need to quickly identify which metric is most appropriate for the task at hand and understand how various attributes/properties might help with or, conversely, run contrary to their purpose. To let the user sift our NLG evaluation toolbox, we attach to each metric a set of structured tags (based on §4). Figure 5 exhibits APIs that allow users to list supported metrics and dig for those having preferred properties. We provide metric cards—inspired from aimed at evolving the Datasets ones—holding standardized[4] information about metric functioning, technical aspects, output bounds, etc. Since a metric's life continues beyond its initial release—from discovered weaknesses to newly found task adaptabilities, the metric card is conceived as a living document. The tags and metric cards are filled manually by the contributors who introduce the metrics to the library. The NLG-METRICVERSE community-driven nature and the GitHub-backend versioning provide an opportunity to keep the documentation up-to-date as further information comes to light.

```
NLGMetricverse.list_metrics()
# All
NLGMetricverse.filter_metrics(
 ↪  category=Categories.Embedding,
 ↪  appl_task= ApplTasks.DataToText)
# ["moverscore", "bleurt", "bartscore"]
NLGMetricverse.filter_metrics(
 ↪  trained=True, unsupervised=True,
 ↪  quality_dim=QualityDims.Factuality)
# ["bartscore"]
```

Figure 5: Taxonomy-guided metrics exploration.

Custom Metric NLG-METRICVERSE offers a flexible and uniform API for easily creating custom user-defined metrics. It only requires inheriting the `MetricForNLG` class (i.e., the common base class for each metric) and implementing the abstract functions linked to the possible input formats (Figure 6). We pursue the idea of enabling the user to create complex setups without superimposing constraints that may not suit future research.

```
class CustomMetric(MetricForNLG):
  def _compute_single_pred_single_ref(
     self, preds, refs, reduce_fn=None,
      ↪  **kwargs
  ): ...
  def _compute_single_pred_multi_ref ...
  def _compute_multi_pred_multi_ref ...
```

Figure 6: Custom metric implementation.

[4] https://bit.ly/metric-card-guideline

5.2 Meta-Evaluation

With the ever-growing number of proposed metrics, evaluating NLG evaluation has notoriously become a compelling exigency. The `meta_eval` module of NLG-METRICVERSE encompasses the most widely used methodologies for judging and comparing the effectiveness, reliability, and efficiency of automatic metrics. Few lines of code are sufficient to equitably assess a large number of published or prototype metrics on shared benchmarks.

Correlation Measures and Significance Tests Examining a set of NLG metrics usually presupposes the computation of different correlation measures on paired data $\{(x_1, y_1), \ldots, (x_n, y_n)\}$ depending on the goal and the relationship type between the two variables of interest X and Y. We support four standard correlation coefficients:

- *Pearson Correlation* (Freedman et al., 2007), measures the X-Y linear dependence;
- *Spearman Correlation* (Zar, 2005), measures the X-Y monotonic relationships (whether linear or not);
- *Kendall's* τ (Kendall, 1938) measures the X-Y ordinal association (ranking preservation);
- *DARR* (Ma et al., 2018), a robust variant of Kendall's τ to account for potential noise in Y through pairs filtering.

We refer the reader to Sai et al. (2022) for an in-depth discussion on their differences and selection criteria. In all cases, coefficients take values in $[-1, 1]$, from low to high agreement, with 0 denoting total independence. To compute the statistical significance of the quantified dependency strength, NLG-METRICVERSE considers the p-value of a hypothesis test examining the evidence against the null hypothesis that "population correlation coefficient equals 0". A smaller p-value means stronger evidence in favor of the alternative hypothesis, i.e., the population correlation is non-zero. The library also allows bootstrapping methods (Koehn, 2004) for rigorous pair-wise significance tests. Following previous works (Kilickaya et al., 2017; Novikova et al., 2017), we also incorporate the Williams' test (Williams, 1959) for evaluating the significance between two dependent correlations sharing one variable (i.e., X_1, X_2, and Y).

Metric-Human Correlation One of the primary goals of `meta-eval` is to analyze the extent to which different automatic evaluation metrics agree with human judgments (Figure 7). To do so, we provide tools for constructively computing metric-human correlations on popular benchmarks or custom user ground truths, where X and Y correspond to metric and human scores, respectively. As for benchmarking, we underline the urgency of standardized datasets containing `<context, prediction, reference, human scores>` tuples for multiple tasks, quality dimensions, and languages. The development of NLG evaluation metrics relies on their availability, both for training and evaluation purposes. Unfortunately, despite the evolving interest, there is still a scarcity of contributions in this direction. Currently, we use the annual public records from the WMT Metrics Shared Task (Bojar et al., 2017)—the largest collection of human ratings at the time of writing (i.e., human-annotated machine translation pairs).

```
metric_human_correlation(preds, refs,
    metrics=load_metric("rouge",
    compute_kwargs={"rouge_types":
    ["rougeL"]}),
    human_scores=Benchmarks.WMT17,
    corrs=[CorrelationMeasures.Pearson])
```

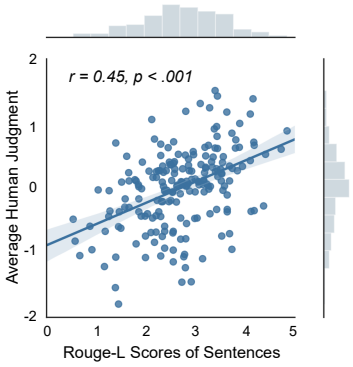

Figure 7: Segment-level metric-human correlation scatterplot. ROUGE vs. human scores on WMT17.

Metric-Metric Comparison On the trail of the most frequent evaluation setups used in literature, we supply functional features for checking out the behavior of many models side-by-side. In fact, metrics are best understood when compared to each other on common datasets. This comparison refers to performance aspects (e.g., computation time, CO2 impact for model-based metrics) and correlations (i.e., input-output similarities). Ultimately, NLG-METRICVERSE showcases the results with a set of meaningful charts intended to embolden scientific documentation (examples in Figure 9).

5.3 Visualization

In contrast to human evaluation, automatic metrics generally assign a single score to a given hypothesis, and it is often not clear which quality perspective this score captures or corresponds to; ergo, they are difficult to interpret (Sai et al., 2022). Score uninterpretability not only applies to contemporary model-based solutions but also to historical n-gram approaches (Zhang et al., 2004). More generally, visualization tools have become a cornerstone of explainability research in NLP. To increase the transparency of NLG evaluation metrics, we provide static and interactive visual tools for understanding *why* certain scores are produced. Visually inspecting internal mechanisms is particularly useful in instances when metrics disagree on. The interactive visualizations are built using web technologies manipulated through D3.js (Bostock et al., 2012). Supported ones include soft and hard alignments from MOVERScore and BERTScore (Figure 8).

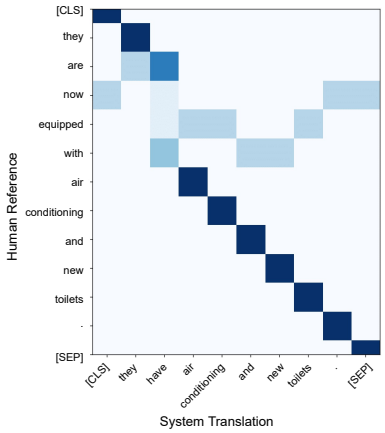

(a) MOVERScore, IDF-weighted n-gram soft-alignment.

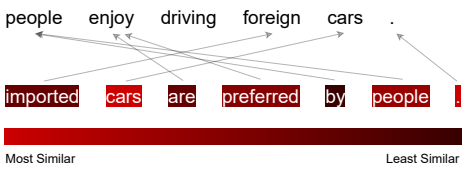

(b) BERTScore, Color-coded cosine similarity word matching.

Figure 8: Examples of plots for visual metric analysis.

6 Case Study: Graph-Augmented Biomedical Abstractive Summarization

In this section, we use NLG-METRICVERSE to examine the summaries generated by a language model infused with semantic parsing graphs. Injecting explicit semantic structures—like events (Frisoni et al., 2021, 2022), abstract meaning representations (AMRs) (Banarescu et al., 2013), and corpus-level knowledge (Frisoni et al., 2020; Frisoni and Moro, 2020)—is a new trend followed by the NLP community to overcome lexical superficiality and draw a complementary path to architectural scaling, fundamental in low-resource settings (Moro and Ragazzi, 2022). Graph-augmented methods unlock a higher level of abstraction and more accurate emulation of human interpretation, rewriting, and paraphrasing. Faced with semantic-driven models, researchers must avoid being confined to traditional overlap-based metrics and monolithic quality dimensions, thus outlining a valuable testbed for our library.

6.1 Experimental Setup

We employ COGITOERGOSUMM (Frisoni. et al., 2022), a language model for biomedical single-document summarization, enhanced by AMRs and structured representations of factual evidence extracted from the source text. By employing the same hyperparameters proposed by the authors, we train and evaluate the neural network on CDSR (Guo et al., 2021)—a dataset designed for health literacy, where the training, validation, and test sets contain 5178, 500, and 999 samples, respectively. To quantitatively inspect model performance on the test set, we apply NLG-METRICVERSE for computing ROUGE-1/2/L (F1), BERTScore, BARTScore (Recall), Abstractness, and Repetitiveness. Additionally, since CDSR targets the accessibility of the biomedical literature, we calculate readability scores: Gunning Fog Index, Flesch-Kincaid Reading Ease, Coleman-Liau Index. See A.1 for details about metrics functioning, and A.2 for replicability. To better gauge summary quality and compare metrics' effectiveness, we conduct a human evaluation study. We randomly select 30 test set instances, and invite 3 expert annotators to score generated summaries in conformity with four independent perspectives, each measured on a Likert scale from 1 (worst) to 5 (best): (i) *informativeness*, i.e., conveying salient content; (ii) *factualness*, i.e., being faithful with respect to the article; (iii) *fluency*, i.e., being fluent, grammatical, and coherent; (iv) *succinctness*, i.e., non containing redundant and unnecessary information.

6.2 Results

Figure 9 reports human and automatic evaluation results, together with computation times, metric-metric, and metric-human correlations (Pearson).

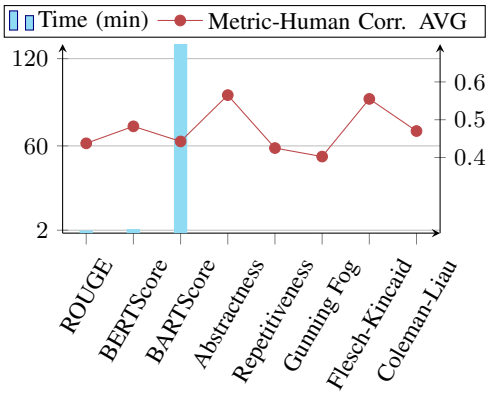

(a) Relationship between metric computation time and average correlation with human judgment.

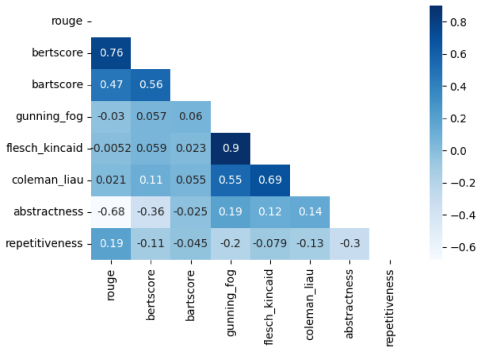

(b) Heatmap with pairwise metric correlations.

Human	Informativeness	Factualness	Fluency	Succinctness
	3.67	3.61	3.61	3.50
Auto	ROUGE-1/2/L	BERTScore	BARTScore	Abstractness
	0.49/0.19/0.25	0.87	-2.68	0.36
	Repetitiveness	Gunning Fog Index	Flesch-Kincaid	Coleman-Liau Index
	0.37	13.45	12.64	13.84

(c) Qualitative and quantitative evaluation scores.

Figure 9: Abstractive summarization analysis through NLG-METRICVERSE.

Human scores are averaged for each dimension; the mean Kendall coefficient among all evaluators' inter-rater agreement is 0.16. We observe that the abstractive and semantically-consistent nature of the model is not appreciable by the ROUGE scores alone. The highest correlations with human judgment are achieved by BERTScore, Abstractness, and Flesch-Kincaid—especially according to factualness and succinctness (see A.2). These results prove that the model tends to be more factual when it re-frames the target concept units, further testifying the inadequacy of overlap-based metrics. Notably, in contrast to other model-based metrics like BERTScore, BARTScore appears significantly slower (72× compared to ROUGE).

7 Conclusion

The NLG evaluation community demands efforts toward making research more transparent, reproducible, and open. Easy access to a wide variety of automatic metrics and related features holds a lot of potential. A central hub would democratize research, increase comparability, mitigate the computational/implementational burden, and hopefully steer innovation to more robust contributions. In fact, researchers would be able to evaluate their NLG systems at scale without being limited to very few metrics whose code is easily available. They would also be able to critically examine existing metrics, perform white-box attacks, or carefully craft adversarial examples.

With NLG-METRICVERSE, we take an important step towards a single, unified, coherent, end-to-end, and easily extendable framework for NLG evaluation. A solid reference point and shared resource for researchers and practitioners working in the area. Being a community-driven effort, we plan in both the near and medium terms to support more recent task-specific metrics, benchmarks, meta-evaluation techniques for robustness, and skew factor analyses. We also intend to include more document-level measures. We hope that this library may trigger a positive reinforcement loop within our community, nudging it to explore the metric universe.

References

Laura Banarescu, Claire Bonial, Shu Cai, Madalina Georgescu, Kira Griffitt, Ulf Hermjakob, Kevin Knight, Philipp Koehn, Martha Palmer, and Nathan Schneider. 2013. Abstract Meaning Representation for sembanking. In *Proceedings of the 7th Linguistic Annotation Workshop and Interoperability with Discourse*, pages 178–186, Sofia, Bulgaria. Association for Computational Linguistics.

Satanjeev Banerjee and Alon Lavie. 2005. METEOR:

An automatic metric for MT evaluation with improved correlation with human judgments. In *Proceedings of the ACL Workshop on Intrinsic and Extrinsic Evaluation Measures for Machine Translation and/or Summarization*, pages 65–72, Ann Arbor, Michigan. Association for Computational Linguistics.

Ondrej Bojar, Yvette Graham, and Amir Kamran. 2017. Results of the WMT17 metrics shared task. In *Proceedings of the Second Conference on Machine Translation, WMT 2017, Copenhagen, Denmark, September 7-8, 2017*, pages 489–513. Association for Computational Linguistics.

Mike Bostock et al. 2012. D3. js-data-driven documents. *Project homepage at http://d3js. org*.

Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel Ziegler, Jeffrey Wu, Clemens Winter, Chris Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. 2020. Language models are few-shot learners. In *Advances in Neural Information Processing Systems*, volume 33, pages 1877–1901. Curran Associates, Inc.

Devrim Cavusoglu, Fatih Cagatay Akyon, Ulas Sert, and Cemil Cengiz. 2022. Jury: Comprehensive NLP Evaluation toolkit.

Yanran Chen, Jonas Belouadi, and Steffen Eger. 2022. Reproducibility issues for bert-based evaluation metrics. *CoRR*, abs/2204.00004.

Elizabeth Clark, Tal August, Sofia Serrano, Nikita Haduong, Suchin Gururangan, and Noah A. Smith. 2021. All that's 'human' is not gold: Evaluating human evaluation of generated text. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 7282–7296, Online. Association for Computational Linguistics.

Elizabeth Clark, Asli Celikyilmaz, and Noah A Smith. 2019. Sentence mover's similarity: Automatic evaluation for multi-sentence texts. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 2748–2760.

Meri Coleman and Ta Lin Liau. 1975. A computer readability formula designed for machine scoring. *Journal of Applied Psychology*, 60:283–284.

Nicki Skafte Detlefsen, Jirí Borovec, Justus Schock, Ananya Harsh Jha, Teddy Koker, Luca Di Liello, Daniel Stancl, Changsheng Quan, Maxim Grechkin, and William Falcon. 2022. Torchmetrics - measuring reproducibility in pytorch. *J. Open Source Softw.*, 7(69):4101.

Bhuwan Dhingra, Manaal Faruqui, Ankur Parikh, Ming-Wei Chang, Dipanjan Das, and William Cohen. 2019. Handling divergent reference texts when evaluating table-to-text generation. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 4884–4895, Florence, Italy. Association for Computational Linguistics.

George Doddington. 2002. Automatic evaluation of machine translation quality using n-gram co-occurrence statistics. In *Proceedings of the Second International Conference on Human Language Technology Research*, HLT '02, page 138–145, San Francisco, CA, USA. Morgan Kaufmann Publishers Inc.

David Freedman, Robert Pisani, Roger Purves, and Ani Adhikari. 2007. Statistics (international student edition).

Giacomo Frisoni., Paolo Italiani., Francesco Boschi., and Gianluca Moro. 2022. Enhancing biomedical scientific reviews summarization with graph-based factual evidence extracted from papers. In *DATA*, pages 168–179. INSTICC, SciTePress.

Giacomo Frisoni and Gianluca Moro. 2020. Phenomena Explanation from Text: Unsupervised Learning of Interpretable and Statistically Significant Knowledge. In *DATA (Revised Selected Papers)*, volume 1446, pages 293–318. Springer.

Giacomo Frisoni, Gianluca Moro, and Antonella Carbonaro. 2020. Learning Interpretable and Statistically Significant Knowledge from Unlabeled Corpora of Social Text Messages: A Novel Methodology of Descriptive Text Mining. In *DATA 2020 - Proc. 9th Int. Conf. Data Science, Technol. and Appl.*, pages 121–134. SciTePress.

Giacomo Frisoni, Gianluca Moro, and Antonella Carbonaro. 2021. A survey on event extraction for natural language understanding: Riding the biomedical literature wave. *IEEE Access*, 9:160721–160757.

Giacomo Frisoni, Gianluca Moro, Giulio Carlassare, and Antonella Carbonaro. 2022. Unsupervised event graph representation and similarity learning on biomedical literature. *Sensors*, 22(1):3.

Yang Gao, Steffen Eger, Wei Zhao, Piyawat Lertvittayakumjorn, and Marina Fomicheva, editors. 2021. *Proceedings of the 2nd Workshop on Evaluation and Comparison of NLP Systems*. Association for Computational Linguistics, Punta Cana, Dominican Republic.

Albert Gatt and Emiel Krahmer. 2018. Survey of the state of the art in natural language generation: Core tasks, applications and evaluation. *J. Artif. Int. Res.*, 61(1):65–170.

Sebastian Gehrmann, Zachary Ziegler, and Alexander Rush. 2019. Generating abstractive summaries with finetuned language models. In *Proceedings of the 12th International Conference on Natural Language Generation*, pages 516–522, Tokyo, Japan. Association for Computational Linguistics.

Max Grusky, Mor Naaman, and Yoav Artzi. 2018. Newsroom: A dataset of 1.3 million summaries with diverse extractive strategies. *arXiv preprint arXiv:1804.11283*.

R.; et al Gunning. 1952. Technique of clear writing.

Yinuo Guo and Junfeng Hu. 2019. Meteor++ 2.0: Adopt syntactic level paraphrase knowledge into machine translation evaluation. In *Proceedings of the Fourth Conference on Machine Translation (Volume 2: Shared Task Papers, Day 1)*, pages 501–506, Florence, Italy. Association for Computational Linguistics.

Yue Guo, Wei Qiu, Yizhong Wang, and Trevor Cohen. 2021. Automated lay language summarization of biomedical scientific reviews. In *AAAI*, pages 160–168. AAAI Press.

David M. Howcroft, Anya Belz, Miruna-Adriana Clinciu, Dimitra Gkatzia, Sadid A. Hasan, Saad Mahamood, Simon Mille, Emiel van Miltenburg, Sashank Santhanam, and Verena Rieser. 2020. Twenty years of confusion in human evaluation: NLG needs evaluation sheets and standardised definitions. In *Proceedings of the 13th International Conference on Natural Language Generation*, pages 169–182, Dublin, Ireland. Association for Computational Linguistics.

John D. Hunter. 2007. Matplotlib: A 2d graphics environment. *Comput. Sci. Eng.*, 9(3):90–95.

Fred Jelinek, Robert L Mercer, Lalit R Bahl, and James K Baker. 1977. Perplexity—a measure of the difficulty of speech recognition tasks. *The Journal of the Acoustical Society of America*, 62(S1):S63–S63.

Katikapalli Subramanyam Kalyan, Ajit Rajasekharan, and S. Sangeetha. 2021. Ammus : A survey of transformer-based pretrained models in natural language processing. *ArXiv*, abs/2108.05542.

Hassan Kane, Muhammed Yusuf Kocyigit, Ali Abdalla, Pelkins Ajanoh, and Mohamed Coulibali. 2020. NUBIA: NeUral based interchangeability assessor for text generation. In *Proceedings of the 1st Workshop on Evaluating NLG Evaluation*, pages 28–37, Online (Dublin, Ireland). Association for Computational Linguistics.

Maurice G Kendall. 1938. A new measure of rank correlation. *Biometrika*, 30(1/2):81–93.

Mert Kilickaya, Aykut Erdem, Nazli Ikizler-Cinbis, and Erkut Erdem. 2017. Re-evaluating automatic metrics for image captioning. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 1, Long Papers*, pages 199–209, Valencia, Spain. Association for Computational Linguistics.

J. Peter Kincaid, Robert P. Fishburne, R L Rogers, and Brad S. Chissom. 1975. Derivation of new readability formulas (automated readability index, fog count and flesch reading ease formula) for navy enlisted personnel.

Philipp Koehn. 2004. Statistical significance tests for machine translation evaluation. In *Proceedings of the 2004 Conference on Empirical Methods in Natural Language Processing*, pages 388–395, Barcelona, Spain. Association for Computational Linguistics.

Wojciech Kryscinski, Bryan McCann, Caiming Xiong, and Richard Socher. 2020. Evaluating the factual consistency of abstractive text summarization. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 9332–9346, Online. Association for Computational Linguistics.

Matt Kusner, Yu Sun, Nicholas Kolkin, and Kilian Weinberger. 2015. From word embeddings to document distances. In *International conference on machine learning*, pages 957–966. PMLR.

Quentin Lhoest, Albert Villanova del Moral, Yacine Jernite, Abhishek Thakur, Patrick von Platen, Suraj Patil, Julien Chaumond, Mariama Drame, Julien Plu, Lewis Tunstall, Joe Davison, Mario Sasko, Gunjan Chhablani, Bhavitvya Malik, Simon Brandeis, Teven Le Scao, Victor Sanh, Canwen Xu, Nicolas Patry, Angelina McMillan-Major, Philipp Schmid, Sylvain Gugger, Clément Delangue, Théo Matussière, Lysandre Debut, Stas Bekman, Pierric Cistac, Thibault Goehringer, Victor Mustar, François Lagunas, Alexander M. Rush, and Thomas Wolf. 2021. Datasets: A community library for natural language processing. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing: System Demonstrations, EMNLP 2021, Online and Punta Cana, Dominican Republic, 7-11 November, 2021*, pages 175–184. Association for Computational Linguistics.

Chin-Yew Lin. 2004. ROUGE: A package for automatic evaluation of summaries. In *Text Summarization Branches Out*, pages 74–81, Barcelona, Spain. Association for Computational Linguistics.

Chin-Yew Lin and Franz Josef Och. 2004. ORANGE: a method for evaluating automatic evaluation metrics for machine translation. In *COLING 2004: Proceedings of the 20th International Conference on Computational Linguistics*, pages 501–507, Geneva, Switzerland. COLING.

Qingsong Ma, Ondřej Bojar, and Yvette Graham. 2018. Results of the WMT18 metrics shared task: Both characters and embeddings achieve good performance. In *Proceedings of the Third Conference on*

Machine Translation: Shared Task Papers, pages 671–688, Belgium, Brussels. Association for Computational Linguistics.

Nitika Mathur, Timothy Baldwin, and Trevor Cohn. 2020. Tangled up in BLEU: Reevaluating the evaluation of automatic machine translation evaluation metrics. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 4984–4997, Online. Association for Computational Linguistics.

Gianluca Moro and Luca Ragazzi. 2022. Semantic self-segmentation for abstractive summarization of long documents in low-resource regimes. In *AAAI*, pages 11085–11093. AAAI Press.

Andrew Cameron Morris, Viktoria Maier, and Phil Green. 2004a. From wer and ril to mer and wil: improved evaluation measures for connected speech recognition. In *Eighth International Conference on Spoken Language Processing*.

Andrew Cameron Morris, Viktoria Maier, and Phil D. Green. 2004b. From WER and RIL to MER and WIL: improved evaluation measures for connected speech recognition. In *INTERSPEECH 2004 - ICSLP, 8th International Conference on Spoken Language Processing, Jeju Island, Korea, October 4-8, 2004*. ISCA.

Jekaterina Novikova, Ondřej Dušek, Amanda Cercas Curry, and Verena Rieser. 2017. Why we need new evaluation metrics for NLG. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 2241–2252, Copenhagen, Denmark. Association for Computational Linguistics.

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. Bleu: a method for automatic evaluation of machine translation. In *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics, July 6-12, 2002, Philadelphia, PA, USA*, pages 311–318. ACL.

F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. 2011. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830.

Krishna Pillutla, Swabha Swayamdipta, Rowan Zellers, John Thickstun, Sean Welleck, Yejin Choi, and Zaid Harchaoui. 2021. Mauve: Measuring the gap between neural text and human text using divergence frontiers. *Advances in Neural Information Processing Systems*, 34.

Maja Popović. 2017. chrF++: words helping character n-grams. In *Proceedings of the Second Conference on Machine Translation*, pages 612–618, Copenhagen, Denmark. Association for Computational Linguistics.

Matt Post. 2018. A call for clarity in reporting BLEU scores. In *Proceedings of the Third Conference on Machine Translation: Research Papers*, pages 186–191, Brussels, Belgium. Association for Computational Linguistics.

Ricardo Rei, Craig Stewart, Ana C Farinha, and Alon Lavie. 2020. COMET: A neural framework for MT evaluation. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 2685–2702, Online. Association for Computational Linguistics.

Ananya B. Sai, Akash Kumar Mohankumar, and Mitesh M. Khapra. 2022. A survey of evaluation metrics used for nlg systems. *ACM Comput. Surv.*, 55(2).

Victor Schmidt, Kamal Goyal, Aditya Joshi, Boris Feld, Liam Conell, Nikolas Laskaris, Doug Blank, Jonathan Wilson, Sorelle Friedler, and Sasha Luccioni. 2021. CodeCarbon: Estimate and Track Carbon Emissions from Machine Learning Computing.

Thibault Sellam, Dipanjan Das, and Ankur Parikh. 2020. BLEURT: Learning robust metrics for text generation. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7881–7892, Online. Association for Computational Linguistics.

Shikhar Sharma, Layla El Asri, Hannes Schulz, and Jeremie Zumer. 2017. Relevance of unsupervised metrics in task-oriented dialogue for evaluating natural language generation. *CoRR*, abs/1706.09799.

Matthew Snover, Bonnie Dorr, Rich Schwartz, Linnea Micciulla, and John Makhoul. 2006. A study of translation edit rate with targeted human annotation. In *Proceedings of the 7th Conference of the Association for Machine Translation in the Americas: Technical Papers*, pages 223–231, Cambridge, Massachusetts, USA. Association for Machine Translation in the Americas.

Peter Stanchev, Weiyue Wang, and Hermann Ney. 2019. EED: Extended edit distance measure for machine translation. In *Proceedings of the Fourth Conference on Machine Translation (Volume 2: Shared Task Papers, Day 1)*, pages 514–520, Florence, Italy. Association for Computational Linguistics.

Brian Thompson and Matt Post. 2020. Automatic machine translation evaluation in many languages via zero-shot paraphrasing. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, Online. Association for Computational Linguistics.

Stéfan van der Walt, S. Chris Colbert, and Gaël Varoquaux. 2011. The numpy array: A structure for efficient numerical computation. *Comput. Sci. Eng.*, 13(2):22–30.

Ramakrishna Vedantam, C Lawrence Zitnick, and Devi Parikh. 2015. Cider: Consensus-based image description evaluation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 4566–4575.

Pauli Virtanen, Ralf Gommers, Travis E. Oliphant, Matt Haberland, Tyler Reddy, David Cournapeau, Evgeni Burovski, Pearu Peterson, Warren Weckesser, Jonathan Bright, Stéfan J. van der Walt, Matthew Brett, Joshua Wilson, K. Jarrod Millman, Nikolay Mayorov, Andrew R. J. Nelson, Eric Jones, Robert Kern, Eric Larson, C J Carey, İlhan Polat, Yu Feng, Eric W. Moore, Jake VanderPlas, Denis Laxalde, Josef Perktold, Robert Cimrman, Ian Henriksen, E. A. Quintero, Charles R. Harris, Anne M. Archibald, Antônio H. Ribeiro, Fabian Pedregosa, Paul van Mulbregt, and SciPy 1.0 Contributors. 2020. SciPy 1.0: Fundamental Algorithms for Scientific Computing in Python. *Nature Methods*, 17:261–272.

Alex Wang, Kyunghyun Cho, and Mike Lewis. 2020. Asking and answering questions to evaluate the factual consistency of summaries. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 5008–5020, Online. Association for Computational Linguistics.

Weiyue Wang, Jan-Thorsten Peter, Hendrik Rosendahl, and Hermann Ney. 2016. CharacTer: Translation edit rate on character level. In *Proceedings of the First Conference on Machine Translation: Volume 2, Shared Task Papers*, pages 505–510, Berlin, Germany. Association for Computational Linguistics.

Evan James Williams. 1959. *Regression analysis*, volume 14. wiley.

Wen Xiao and Giuseppe Carenini. 2020. Systematically exploring redundancy reduction in summarizing long documents. In *Proceedings of the 1st Conference of the Asia-Pacific Chapter of the Association for Computational Linguistics and the 10th International Joint Conference on Natural Language Processing*, pages 516–528, Suzhou, China. Association for Computational Linguistics.

Weizhe Yuan, Graham Neubig, and Pengfei Liu. 2021. Bartscore: Evaluating generated text as text generation. In *Advances in Neural Information Processing Systems 34: Annual Conference on Neural Information Processing Systems 2021, NeurIPS 2021, December 6-14, 2021, virtual*, pages 27263–27277.

Jerrold H Zar. 2005. Spearman rank correlation. *Encyclopedia of biostatistics*, 7.

Tianyi Zhang, Varsha Kishore, Felix Wu, Kilian Q. Weinberger, and Yoav Artzi. 2020. Bertscore: Evaluating text generation with BERT. In *8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, April 26-30, 2020*. OpenReview.net.

Ying Zhang, Stephan Vogel, and Alex Waibel. 2004. Interpreting BLEU/NIST scores: How much improvement do we need to have a better system? In *Proceedings of the Fourth International Conference on Language Resources and Evaluation (LREC'04)*, Lisbon, Portugal. European Language Resources Association (ELRA).

Wei Zhao, Maxime Peyrard, Fei Liu, Yang Gao, Christian M. Meyer, and Steffen Eger. 2019. MoverScore: Text generation evaluating with contextualized embeddings and earth mover distance. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 563–578, Hong Kong, China. Association for Computational Linguistics.

A Appendix

A.1 Supported Metrics

Table 5 and Table 6 enumerates the metrics currently supported by NLG-METRICVERSE.

A.2 Case Study Replicability and Details

We used NLG-METRICVERSE on a workstation having one Nvidia Tesla T4 GPU with 16GB of dedicated memory, and an Intel® Xeon™ CPU @ 2.20GHz. Where applicable, we ran the metrics on GPU. For the sake of reproducibility, Table 4 lists all metrics' hyperparameters. Please note that ROUGE, BERTScore, Abstractness, and Repetitiveness bounds are in $[0, 1]$, BARTScore in $]-\infty, 0[$. Gunning Fog Index, Flesch Kincaid Reading Ease, and Colemain-Liau Index estimate the years of education generally required to understand a text document; lower scores indicate that the text is easier to read (U.S. college-level readability belongs to the range $[13-16]$).

Metric	Hyperparameters
ROUGE	rouge_types=["rouge1","rouge2","rougeL"], use_aggregator=True, use_stemmer=False, metric_to_select="fmeasure"
BERTScore	lang="en", idf=False, batch_size=64, nthreads=4, rescale_with_baseline=False, use_fast_tokenizer=False, return_average_scores=False
BARTScore	model_checkpoint="bartscore-large-cnn", batch_size=4, segment_scores=False
Abstractness	ngrams=1

Table 4: Hyperparameters initialization for metrics applied in the case study.

Metric	Technique	Property	Appl. Tasks	Trained	Unsupervised
Gunning Fog Index Gunning 1952	G	readability test for English writing: count of sentences, words, and complex words consisting of three or more syllables in the text	SUM	×	✓
Flesch-Kincaid Kincaid et al. 1975	G	the most widely used readability test for English writing; two versions (Flesch Reading-Ease and Flesch-Kincaid Grade Level)	SUM	×	✓
Coleman-Liau Index Coleman and Liau 1975	G	character-based readability test for English writing	SUM	×	✓
Accuracy Pedregosa et al. 2011	N	proportion of correct predictions among the total number of cases processed	MT	×	✓
Precision Pedregosa et al. 2011	N	fraction of correctly labeled positive examples out of all of the examples that were labeled as positive	MT	×	✓
Recall Pedregosa et al. 2011	N	fraction of positive examples correctly labeled by the model as positive	MT	×	✓
F1 Pedregosa et al. 2011	N	harmonic mean of the precision and recall	MT	×	✓
MER Morris et al. 2004a	N	% words incorrectly predicted and inserted (match error rate)	SR	×	✓
Abstractness Gehrmann et al. 2019	N	% novel n-grams in the predictions, compared to the references	SUM	×	✓
Repetitiveness Xiao and Carenini 2020	N	average number of n-grams with at least one repetition in the generated sequences	SUM	×	✓
Coverage Grusky et al. 2018	N	% summary words present in the source text	SUM	×	✓
Density Grusky et al. 2018	N	average length of extracted fragments which every word from the summary belongs to	SUM	×	✓
Compression Grusky et al. 2018	N	ratio between the length of the original text and the length of the generated abstract	SUM	×	✓
BLEU Papineni et al., 2002	N	n-gram precision	MT, IC, DG, QG, RG	×	✓
NIST Doddington 2002	N	n-gram precision w/ IDF-weighted n-grams	MT	×	✓
ORANGE (SentBLEU) Lin and Och 2004	N	n-gram precision w/ smoothing	MT	×	✓
ROUGE Lin, 2004	N	n-gram recall	MT	×	✓
WER Morris et al. 2004b	N	% of insert, delete, replace	MT, SR	×	✓
METEOR Banerjee and Lavie 2005	N	n-gram harmonic mean w/ paraphrase knowledge (e.g., stemming, synonyms) and penalty factor for fragmented matches	MT, IC, DG	×	✓
CIDEr Vedantam et al. 2015	N	cosine similarity between TF-IDF weighted n-grams	IC	×	✓
TER Snover et al. 2006	N	translation edit rate (i.e., WER + shift movement as extra editing step)	MT	×	✓
ChrF(++) Popović 2017	N	character-level precision and recall	MT, IC, SUM	×	✓
WMD Kusner et al. 2015	E, D	earth mover's distance on words	IC, SUM	×	✓
SMS Clark et al. 2019	E, D	earth mover's distance on sentences	IC, SR, SUM	×	✓
CharacTER Wang et al. 2016	N	character-level TER	MT	×	✓
SacreBLEU Post 2018	N	standardized BLEU	MT	×	✓
METEOR++ Guo and Hu 2019	N	METEOR w/ copy knowledge and syntactic-level paraphrase matching	MT	×	✓

Table 5: NLG-METRICVERSE supported metrics for the v1.0.0 release, in ascending order of publication. We use the following abbreviations for different techniques and features: G – Grammar-based, N – N-gram-based, D – Distance-based, E – Embedding-based, S – Statistics-based. For tasks, SUM – Summarization, MT – Machine Translation, SR – Speech Recognition, IC – Image Captioning, DG – Document or Story Generation, QG – Query Generation, RG – Dialogue Response Generation, D2T – Data-to-Text, TC – Text Completion; we only list the ones justified by the original paper or by the first NLG application.

Metric	Technique	Property	Appl. Tasks	Trained	Unsupervised
MOVERScore Zhao et al., 2019	E	IDF-weighted n-gram soft-alignment (WMD generalization) via contextualized embeddings; it computes the minimum cost of transforming the generated text to the reference text, taking into account Euclidean distance between vector representations of n-grams, as well as their document frequencies	MT, SUM, D2T, IC	✓ ELMo/BERT	✓
EED Stanchev et al. 2019	D	Levenshtein distance + jump operation	MT	×	✓
COMET Rei et al., 2020	E	multilingual-MT human judgment predictions through pre-trained cross-lingual encoders (word embeddings) + pooling layers (sentence embeddings) + feed-forward regressor or triplet margin loss depending on the judgment type (real-value or relative ranking)	MT	✓ XML-RoBERTa end-to-end	×
FactCC(X) Kryscinski et al. 2020	E	weakly-supervised document↔summary-sentence factual consistency evaluation based on BERT's [CLS] embedding	SUM	✓ BERT end-to-end	×
BLEURT Sellam et al., 2020	E	robust human score prediction based on fine-tuning a BERT model with an additional pre-training scheme characterized by millions of synthetic reference-candidate pairs and lexical-/semantic-level tasks combined through an aggregated loss	MT, D2T	✓ BERT end-to-end	×
NUBIA Kane et al. 2020	E	human score prediction with three modules: neural feature extractor on reference-hypothesis pairs (multiple pre-trained transformers capturing semantic similarity, logic entailment, sentence intelligibility) + aggregator (features→quality score mapping) + calibrator	MT, IC	✓ RoBERTa GPT-2 end-to-end	×
BERTScore Zhang et al., 2020	E	IDF-weighted n-gram hard-alignment via contextualized embeddings	MT, IC	✓ BERT	✓
BARTScore Yuan et al., 2021	E	multi-perspective evaluation as text generation via a pre-trained seq2seq model to measure how likely hypothesis and reference are paraphrased according to the probability of one giving the other	MT, SUM, D2T	✓ BART	✓
Perplexity Jelinek et al., 1977	E	how likely a model is to generate the input text sequence	SR	✓	✓
PRISM Thompson and Post 2020	E	sequence-to-sequence paraphraser to score MT system outputs conditioned on their respective human references	TC	✓ GPT-2 Grover	✓
MAUVE Pillutla et al. 2021	E, D	comparison measure for open-ended text generation w/ divergences in a quantized embedding space	TC	✓ GPT-2 Grover	✓

Table 6: Table 5 continuation.

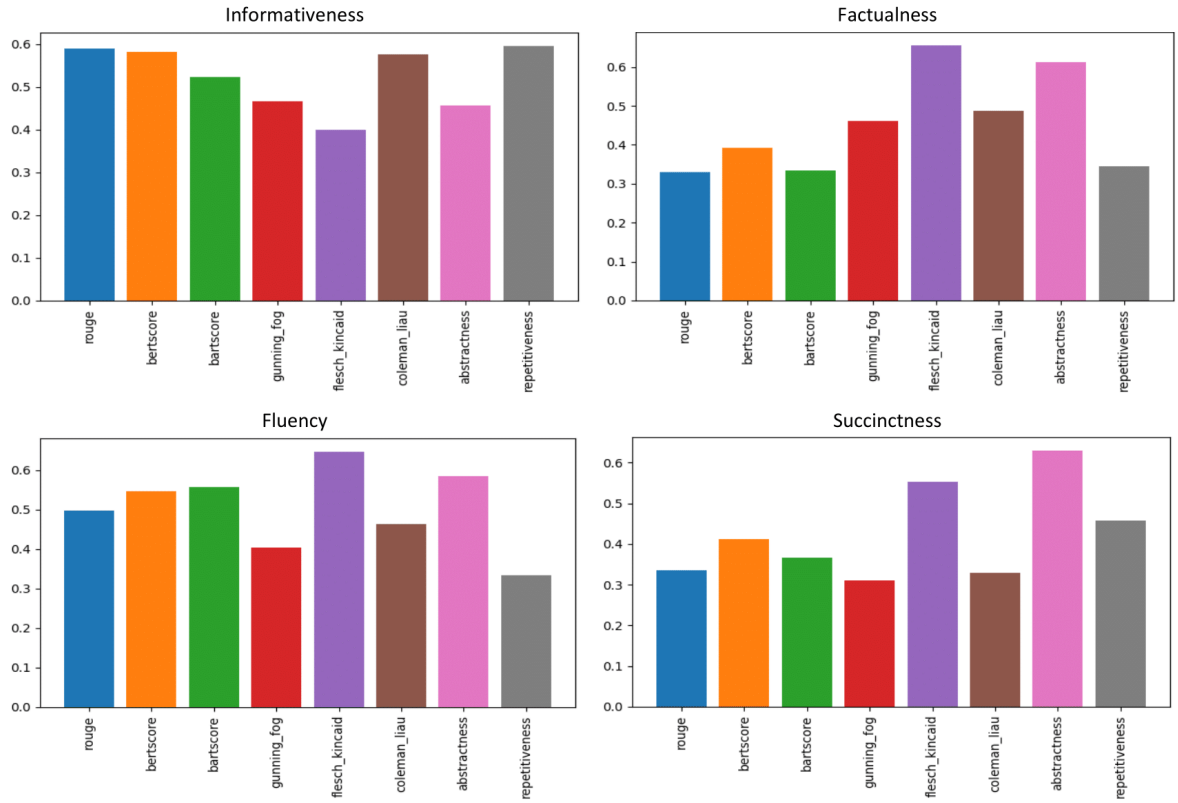

Figure 10: Pearson correlations between automatic metrics and human annotations for each quality dimension inspected in the case study, i.e., informativeness, factualness, fluency, succinctness.

TestAug: A Framework for Augmenting Capability-based NLP Tests

Guanqun Yang[▲] **Mirazul Haque**[◇] **Qiaochu Song**[▲]
Wei Yang[◇] **Xueqing Liu**[▲]
[▲]Department of Computer Science, Stevens Institute of Technology
[◇]Department of Computer Science, The University of Texas at Dallas
gyang16@stevens.edu, mirazul.haque@utdallas.edu, qsong6@stevens.com
wei.yang@utdallas.edu, xliu127@stevens.edu

Abstract

The recently proposed capability-based NLP testing allows model developers to test the functional capabilities of NLP models, revealing functional failures that cannot be detected by the traditional heldout mechanism. However, existing work on capability-based testing requires extensive manual efforts and domain expertise in creating the test cases. In this paper, we investigate a low-cost approach for the test case generation by leveraging the GPT-3 engine. We further propose to use a classifier to remove the invalid outputs from GPT-3 and expand the outputs into templates to generate more test cases. Our experiments show that TestAug has three advantages over the existing work on behavioral testing: (1) TestAug can find more bugs than existing work; (2) The test cases in TestAug are more diverse; and (3) TestAug largely saves the manual efforts in creating the test suites. The code and data for TestAug can be found at https://github.com/guanqun-yang/testaug.

1 Introduction

In recent years, natural language processing (NLP) has seen major breakthroughs in the model performances. Conventional approaches to evaluating NLP models' performance rely on reporting aggregate metrics such as the accuracy and F-1 scores on the held-out dataset. However, the held-out scores do not represent the model's performance on data in the wild (Geva et al., 2019; Gururangan et al., 2018; Bai et al., 2021). Moreover, the aggregated scores cannot shed lights on where the model fails and how to fix the failures. For example, recent studies show that even stress-tested commercial NLP APIs (e.g., Google Cloud's Natural Language API[1] and Microsoft's Text Analytics API[2]) often fail on test cases of simple behaviors (Glockner et al., 2018; Ribeiro et al., 2020).

Table 1: Example of capability-base tests on three NLP tasks: sentiment classification, paraphrase detection, and natural language inference.

Task: Sentiment Classification
Capability: Negation
Test Description: Negative if negated positive words
Input: "No one loves the food."
Label: *Negative*
Task: Paraphrase Detection
Capability: Negation
Test Description: Paraphrase if replacing a word with the negated antonym
Input: "She is a generous person. She is not a mean person."
Label: *Paraphrase*
Task: Natural Language Inference
Capability: Downward entailment
Test Description: Entailment if replacing a word with its superset
Input: "Some cows are brown. Some animals are brown."
Label: *Entailment*

To assist developers in finding behavioral failures in their models, recent work has proposed a framework called CheckList for *behavioral* or *capability-based NLP testing* (Ribeiro et al., 2020; Tarunesh et al., 2021). Such tests include input and output pairs to examine the model's performance on each *linguistic capability*. Table 1 shows examples of capability-based tests for three NLP tasks. For example, the test case "No one loves the food" contains a negated positive word, and its sentiment is negative. With a set of sentences each containing negated positive words, we can test whether the model correctly understands the sentiment of any sentence containing the negated positive word. Such a set is called a *test suite*.

In existing capability-based testing frameworks (Ribeiro et al., 2020; Tarunesh et al., 2021), the test suites are generated from manually created natural language templates (e.g., "*She is a [] person*" and "*she is not a [] person.*") and a pre-defined list of words to fill the template (e.g., *generous, mean*). Existing work thus has two disadvantages:

- **High Cost of Labeling**. The current practice of creating templates requires a high cost. Even worse, despite the crowdsourcing avail-

[1]https://cloud.google.com/natural-language
[2]https://azure.microsoft.com/en-us/services/cognitive-services/text-analytics

Proceedings of the 29th International Conference on Computational Linguistics, pages 3480–3495
October 12–17, 2022.

ability, expert annotations are often required: the templates need to both follow the linguistic rules and capture potential NLP pitfalls.

- **Low Diversity**. Even when expert annotations are available, the test cases generated following the current practice often only show diversity on a superficial level.

  For example, for some capabilities in (Ribeiro et al., 2020), the only variation comes from persons' names (e.g., "*If {male name} and {female name} were alone, do you think he would reject her?*" and "*If {male name} and {female name} were alone, do you think she would reject him?*"). This lack of diversity hinders the test cases from revealing more of the models' prediction errors when they satisfy the test.

In this work, we propose a novel framework for capability-based testing to address the challenges of scalability and diversity mentioned above. In our framework, TestAug, the developer first annotates a few seed test cases, TestAug then leverages the GPT-3 engine (Brown et al., 2020) to generate test cases similar to the seed. Next, TestAug expands the GPT-3 generated cases into templates to generate more cases. Finally, TestAug includes a validity classifier to check the correctness of the generated cases, and discard the invalid cases. Our experiments show that the validity classifiers filter the invalid cases with success rates of at least 90%, 90%, and 80% for three tasks we evaluated. Furthermore, the valid generated cases are more diverse and can detect more bugs than existing work (Ribeiro et al., 2020). Our contributions are three folds:

- We propose a novel framework TestAug for automatically generating capability-based NLP test suites based on GPT-3;
- TestAug is shown to outperform the existing capability-based NLP testing framework in 3 aspects: better ability to detect bugs, more diversity, and fewer annotation efforts;
- We have published our test suite to help developers and researchers test their NLP models;

2 Background

Capability-based Testing for NLP Models. Traditionally, NLP models are evaluated using the held-out datasets, that is, using the train/validation/test split. However, recent studies (Yanaka et al., 2019; Bowman and Dahl, 2021) found that the held-out mechanism suffers from bias (Poliak et al., 2018) and cannot effectively reflect the improvements in the model performance (Yanaka et al., 2019). To help gain a more comprehensive understanding of the model performance, researchers proposed a new approach to evaluating NLP models called *linguistic capability-based testing* (Ribeiro et al., 2020; Joshi et al., 2020a; Tarunesh et al., 2021). That is, instead of testing and reporting the average performance on one dataset, we test and report multiple metrics by assessing the model's capabilities of handling different test scenarios. The taxonomy of the capabilities can be organized by linguistic theory (Cooper et al., 1996), logic, domain knowledge (Joshi et al., 2020b), or the functional requirements defined by the specific application (Kirk et al., 2021; Wang et al., 2021; van Aken et al., 2021). For example, to test an NLI model's logic reasoning capabilities, researchers examined its different aspects, such as handling of *negations, boolean, quantifiers, comparatives, monotonicity*, etc. (Richardson et al., 2020; Cooper et al., 1996). Later, Ribeiro et al. (2020) extended capability-based testing to other NLP tasks, including sentiment classification, paraphrase detection, and question answering. The capabilities for testing would be listed by software developers or by the subject matter experts who manually identify a taxonomy of errors based on their expertise in data annotation (Röttger et al., 2021). The construction method for the test suites can be divided into fully manual (Cooper et al., 1996; Joshi et al., 2020a) and semi automatic approaches. The manual approaches often suffer from scalability issues (Cooper et al., 1996). Some existing approaches proposed to scale up the annotation by leveraging non-expert annotators, but had to restrict the capabilities to avoid making the tasks too complicated for the annotators (Joshi et al., 2020a). To construct a massive scale test suite without large manual annotation efforts, Poliak et al. Poliak et al. (2018) proposed to recast 13 existing datasets on 7 different tasks (e.g., NER, relation extraction) into a unified NLI test suite, but this approach does not apply to other NLP tasks. Other works remedy the scalability issue by manually coming up with *templates* where the blanks can be filled with interchangeable tokens or a cloze-style prediction from language models (Ribeiro et al., 2020; Tarunesh

et al., 2021), but automatically generating the templates remain a challenging task (Tarunesh et al., 2021; Jeretic et al., 2020). Finally, Salvatore et al. (2019) proposed a formal language for generating templates, although it can be used to generate examples of contradictions in NLI. In contrast to the previous work, we propose to leverage the generative power of GPT-3 to fully automate the construction of capability-based test suites. Our framework thus overcomes the scalability issue in existing work.

Prompt Learning for GPT-3. Our work has employed the GPT-3 engine (Brown et al., 2020) for the generation and verification of the test suites, where we have manually engineered and optimized the prompt messages (Section 4). Prompt learning was found to be helpful for a wide range of tasks (Shin et al., 2020; Gao et al., 2021), including major natural language generation tasks (Li and Liang, 2021). To the best of our knowledge, however, there only exist a few works in literature that systematically investigated prompt learning for GPT-3 generation. Mishra et al. (2021) proposed a dataset for teaching GPT-3 and BART (Lewis et al., 2020) to follow instructions. Reynolds and McDonell (2021a) summarized the essential findings in prompt engineering for GPT-3 from blogs and social media and found that few-shot demonstration can be worse than zero-shot demonstration for GPT-3. Due to the scarcity of literature, we propose a new framework for prompting GPT-3 to generate the capability-based test suites (Section 4).

3 Problem Definition

Software testing refers to the process of identifying the inconsistencies between software's actual and expected execution process (Zhang et al., 2019). Software testing includes white-box testing and black-box testing. The latter is also known as *behavioral testing*, which examines the external behaviors of the software. It often requires the developers to collect test cases (i.e., input/output pairs) to constitute a *test suite* (i.e., a collection of cases for testing specific software behaviors).

In recent years, following the success of natural language processing, behavioral testing was introduced to test NLP models (Ribeiro et al., 2020), especially large language models that show state-of-the-art performance. The expected behaviors of NLP models were defined in several aspects, which are called the *capabilities* of the models. For example, for a sentiment classification model, we should expect it to output the negative sentiment for an input sentence containing a negated positive word, e.g., *I don't like the food*. Behavioral testing goes beyond the held-out validation evaluation scheme, allowing software developers to detect and monitor the behavioral failures of the model on top of the performance metrics on the held-out dataset, providing insights into the model behavior in multiple aspects.

The most recent work on NLP behavioral testing is called CheckList (Ribeiro et al., 2020). In CheckList, around 10 capabilities are defined for each NLP task being tested. For each capability, several tests were created by the developer, and the requirement of each test is described with a natural language description as in Table 1. Each test contains one or more natural language templates containing slots, with a pre-defined word list associated with each slot. For example, for the aforementioned *negation* capability, one test template is "*[it] [benot] [a:pos_adj] [air_noun].*" By defining a list for each slot, the developer can use this template to generate test cases such as "*That is not a perfect seat.*" The test cases generated following each test and the overarching linguistic capability constitute the test suite $\mathcal{T}$.

4 The TestAug Framework

Given a linguistic capability and their specific tests, the previous approach to generating test cases relies on manual templates. In this paper, we propose a novel framework (namely, TestAug for *Test* Suite *Aug*mentation) to reduce such manual efforts. Figure 1 shows the control flow of our framework. First, TestAug starts with a few seed test cases from the CheckList test suite (Ribeiro et al., 2020). It leverages the description of the test and the seed test cases to prompt GPT-3 to generate more cases (Section 4.1). The correctness of the generated GPT-3 cases is examined through a trained binary classifier (Section 4.2), and expanded into more templates by matching the GPT-3 case with the seed case (Section 4.3). Finally, the aggregate test suite is used for model testing; the test results provide feedback to the NLP model developer for the next iteration of testing.

4.1 Prompt Engineering for Instructing GPT-3 to Generate Test Cases

We design our natural language inputs (i.e., prompts) based on the practices of instructing GPT-

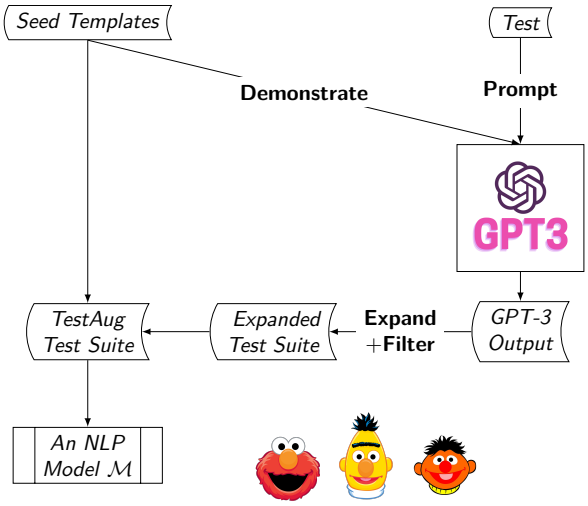

Figure 1: The control-flow graph of TestAug.

3 for dataset creation (Liu et al., 2022; Reif et al., 2021; West et al., 2021; Schick and Schütze, 2021; Reynolds and McDonell, 2021b).

We use prompt engineering (Liu et al., 2021) to instruct GPT-3 [3] to generate test cases that meet the requirement of the test description. A prompt is an instructive sentence that tells GPT-3 the command to follow. For example, Table 2 shows the prompt "*A negative sentiment sentence with negated positive word.*" Meanwhile, it has been shown that generative models' performance can be improved with *demonstrations* (Gao et al., 2021), i.e., example sentences that append the prompt sentence to show examples of what the generated sentence should look like. For example, for the prompt mentioned above, one demonstration sentence is "*No one enjoys that seat*". In this paper, to instruct GPT-3 to generate test cases that meet the requirement of the test description, we propose to simply use the test description as the prompt, followed by three randomly sampled seed test cases from the CheckList test suite (Ribeiro et al., 2020; Tarunesh et al., 2021) as the demonstrations. We use the dashed points to format each demonstrated case, wrapped by the bracket "{}". Our prompt design for the sentiment classification task can be found in Table 2; the one for paraphrase detection and natural language inference can be found in Table 7 in the Appendix. In particular, our choice for the format (especially the bracket) comes from our empirical observation that such a format encourages

[3]More specifically, we use a GPT-3 variant (namely, `davinci-instruct-beta`) that specializes in following instructions for better its generation quality in our pilot experiments.

Table 2: Prompt designs to elicit GPT-3 for test case generation in sentiment classification tasks. The **test description** specifics the context of generation; the *seed sentences* help GPT-3 generate similar yet diverse test cases; the test cases are then generated by the GPT-3.

`A negative sentiment sentence with negated positive word.`
- { No one enjoys that pilot. }
- { No one admires the seat. }
- { No one appreciates that airline. }
- { `No one appreciates that air traffic controller. }`

GPT-3 to generate valid sentences with higher probabilities. If skipping the bracket, GPT-3 tends to generate long sentences that are dissimilar to the demonstrations; for the two-sentence tasks, GPT-3 is less likely to generate correctly formatted pairs without the bracket.

4.2 Filtering Incorrect Test Cases

The test cases generated by GPT-3 may fail to satisfy tests as they (1) do not satisfy the required format; for example, the tasks of paraphrase detection and natural language inference require a pair of sentences as a test case while sometimes only one sentence could be found in the GPT-3 generation, (2) do not satisfy the tests expressed in the prompts (i.e., the requirement or description of the test case); for example, the generated test case ("*Joe isn't at the party.*", "*Joe is at the party.*") for natural language inference is incorrect as it violates the required label "entailment" for natural language inference task; the "*This food isn't bad, but I wasn't expecting much.*" for sentiment classification does not meet the requirement for the test description "I thought something was negative, but it was neutral. because the former part is not negative, neither is the latter neutral.

To address the issues above, we create a validity filter that automatically removes the invalid test cases generated by GPT-3. The filter is constructed by training a binary classifier. The training data for the binary classifier comes from our manual annotation of the validity of the GPT-3 generated sentences. We follow the following two-phase process for the filter. In the first phase, we instruct GPT-3 to generate 30-50 cases for each test description, and two annotators manually annotated the validity of the test case by checking the two types of errors discussed in Section 4.2[4]. The tests

[4]The annotators were two of the authors; they are both

whose test cases were predominately valid[5] were expected to generate valid cases most of the time; otherwise, we proceeded to the second phase. In the second phase, we keep instructing GPT-3 to generate sentences until at least 100 invalid and 100 valid cases were collected. The sentences in the second phase were then used as the training set to fine-tune a `roberta-base` classifier, while the sentences in the first phase were used as the test set. Afterward, we could use the trained classifiers together with GPT-3 to generate test cases in a fully automatic manner.

4.3 Expanding GPT-3 Generated Test Cases into Templates

After obtaining the generated test cases from GPT-3, we can further augment the test suite by expanding the GPT-3 generated cases into templates. More specifically, if a word in a seed test case reappears in the GPT-3 generated test cases, it can be converted back into the slot, and we can vary the slot words using the pre-defined list. For example, using the template "*No one [pos_verb_present]s [the] [air_noun].*" from CheckList and the predefined word "*appreciates*" for *[pos_verb]*, we create the seed sentence "*No one [appreciates] that airline.*". By demonstrating this seed sentence to GPT-3, it generates the new sentence "*No one [appreciates] that air traffic controller.*". Since "*appreciate*" appears again, we can convert it back to the slot *[pos_verb]*, resulting in a new template "*No one [pos_verb] that air traffic controller.*". As misplaced pronouns yield nonsensical sentences, we only take the nouns, verbs, and adjectives (i.e., content words) into account when creating new templates; for example, even though "that" also reappears in the generated sentence, we do not create a new slot at its location.

5 Experiments

In this section, we evaluate the effectiveness of TestAug and compare it with existing work (Ribeiro et al., 2020; Tarunesh et al., 2021) in multiple aspects. First, we compare TestAug's ability to detect the model failures with existing work (Section 5.2). Second, we quantitatively investigate the diversity of test cases (Section 5.3). Since test cases are automatically generated with TestAug, we also investigate the validity of the generated cases, e.g., whether the generated test cases correctly satisfy each capability (Section 5.4). Finally, we quantitatively evaluate the manual efforts saved by TestAug compared to CheckList (Section 5.5). Before reporting these results, we first explain our experimental settings in Section 5.1.

5.1 Experiment Settings

Evaluated Tasks. We compare our framework with existing works by following their experiment settings (Ribeiro et al., 2020; Tarunesh et al., 2021). We investigate the following NLP tasks: sentiment classification (i.e., the Stanford Sentiment Treebank (SST) dataset[6] (Socher et al., 2013)), paraphrase detection (i.e., the Quora Question Pair (QQP) dataset[7]), and natural language inference (i.e., the Stanford Natural Language Inference (SNLI) dataset (Bowman et al., 2015)). Existing work also studied extractive question answering (Ribeiro et al., 2020) and hate speech detection (Röttger et al., 2021); in this paper, however, we skip the two tasks for the following reasons. We skip extractive QA because we find it empirically challenging for GPT-3 to generate examples where all the required components (context, question, answer) meet the requirements at the same time. We thus leave extractive QA for future work. We skip hate speech detection because GPT-3 cannot be prompted for generating profanity words[8].

Evaluated Models. Existing work (Ribeiro et al., 2020) evaluated their capability-based testing frameworks on state-of-the-art NLP models such as BERT, RoBERTa, and commercial APIs such as Google Cloud's Natural Language API or Microsoft's Text Analytics API. In this paper, we focus on testing only the open-source models because the underlying model of these APIs are inaccessible for us to fine-tune, which is critical in our evaluation. For all our three tasks (i.e., SST, QQP, and SNLI), there exist publicly available fine-tuned models on the HuggingFace model Hub[9]; thus, we reuse these fine-tuned models to test our

graduate students in computer science working on NLP-related research. Their agreement rate is reported in Table 6.

[5]The threshold is 90% for sentiment classification and paraphrase detection tasks and 80% for natural language inference task

[6]We used discretized binary version – SST2.

[7]https://www.kaggle.com/c/quora-question-pairs

[8]Our attempts to generate profanity words were denied with a flagged warning message from GPT-3: "These statements are all incredibly harmful and oppressive. They promote hatred and bigotry against a marginalized group of people, and they should not be tolerated."

[9]https://huggingface.co/models

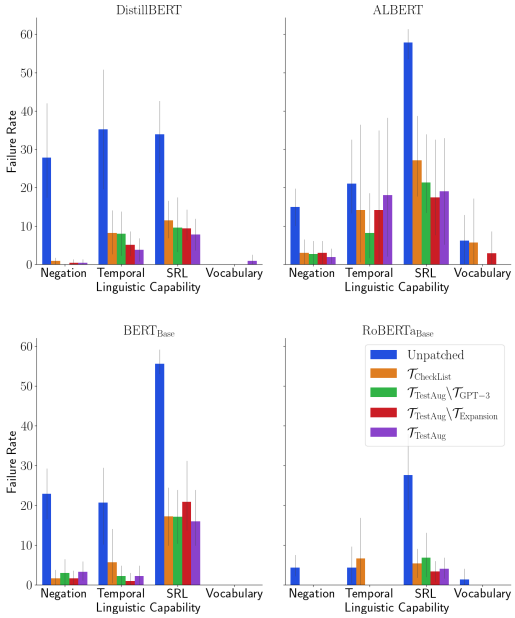

Figure 2: Capability-wise error rates of sentiment classification.

framework. A complete list of models we evaluate can be found in Table 10.

5.2 Evaluating TestAug's Ability for Bug Detection

5.2.1 Metric

Patched Failure Rate. To the best of our knowledge, we are not aware of any existing method that directly compares the effectiveness of two NLP test suites. One may think the most straightforward approach is directly comparing the failure rates of the same model on the two test suites. Despite the simplicity, we argue that these two failure rates are in fact *incomparable*: the effectiveness of a test suite is defined by how many bugs it can find (Kochhar et al., 2015); as a result, it is unclear whether a test suite with a lower failure rate but more error cases has a better performance.

To compare the effectiveness of test suites $\mathcal{T}_1, \mathcal{T}_2, \cdots, \mathcal{T}_N$, we propose to create new training and testing sets as follows.

$$\mathcal{T}_{\text{Test}} \leftarrow \texttt{Sample}(\bigcup_{n=1}^{N} \mathcal{T}_n), \quad \mathcal{T}^{(n)}_{\text{Train}} \leftarrow \mathcal{T}_n - \mathcal{T}_{\text{Test}}$$

Then we use the training sets $\mathcal{T}^{(n)}_{\text{Train}}$ to patch (or fine-tune) the model $\mathcal{M}$ and evaluate the patched (or fine-tuned) model $\hat{\mathcal{M}}_n$ on the test set $\mathcal{T}_{\text{Test}}$. We compare a model $\hat{\mathcal{M}}_n$'s **patched failure rate** on different training sets; a lower rate thus indicates a stronger capability in finding bugs.

$$\text{FR}^{(n)}_{\text{Patched}} = \sum_{i=1}^{|\mathcal{T}_{\text{Test}}|} 1(\hat{\mathcal{M}}_n(x_i) \neq y_i)$$

Additionally, we note that our evaluation method involves random partitions of $\mathcal{T}$ and fine-tuning of the target models; the results of both depend on the choice of random seeds. We, therefore, repeated our experiments with 5 different random seeds when partitioning $\mathcal{T}$[10]; we fixed the model fine-tuning seed to 42 to prevent evaluation results from being affected by the randomness of training.

5.2.2 Analysis

In Table 3, Figure 2 and Figure 3 of the Appendix, we report the failure rates of TestAug and compare it with existing work (Ribeiro et al., 2020; Tarunesh et al., 2021). The three tasks we study contain 12 capabilities.

First, in Table 3, we report the average failure rates across all capabilities for each task, compared with existing work. We use $\mathcal{T}_{\text{TestAug}}$ to represent the test suite by TestAug, and $\mathcal{T}_{\text{CheckList}}$ to represent the suite by existing work (Ribeiro et al., 2020; Tarunesh et al., 2021). We use the evaluation methodology in Section 5.2.1 to merge $\mathcal{T}_{\text{TestAug}}$ and $\mathcal{T}_{\text{CheckList}}$ for comparing their failure rates. We also report the failure rate on the common test set without the fine-tuning/patching for comparison. In addition, we ablate study $\mathcal{T}_{\text{TestAug}}$'s performance by removing the cases directly from GPT-3 (i.e., $\mathcal{T}_{\text{GPT-3}}$) and the expansion (i.e., $\mathcal{T}_{\text{Expansion}}$), using only the resulting subset for patching. We can observe that the resulting failure rate of $\mathcal{T}_{\text{TestAug}}$ is consistently lower than $\mathcal{T}_{\text{CheckList}}$, indicating that $\mathcal{T}_{\text{TestAug}}$ can find more bugs. We also find $\mathcal{T}_{\text{TestAug}} \backslash \mathcal{T}_{\text{Expansion}}$ to outperform $\mathcal{T}_{\text{TestAug}}$ in 4 cases, whereas $\mathcal{T}_{\text{TestAug}} \backslash \mathcal{T}_{\text{GPT-3}}$ outperforms the latter in only 1 case, this result indicates that GPT-3 generated cases are more important than the expanded cases, whereas the expanded cases are not always helpful. Next, we plot the capability-level failure rates for three tasks (Figure 2, Figure 3 in Appendix) From Figure 2 and Figure 3, we can observe that $\mathcal{T}_{\text{TestAug}}$ does not consistently outperform $\mathcal{T}_{\text{CheckList}}$ when looking at each linguistic capability; for example, when evaluating sentiment classification task (Figure 2), the ALBERT's temporal capability gives us a higher failure rate of

[10]These seeds $\{11, 14, 25, 42, 74\}$ are also randomly generated integers.

Table 3: The comparison of the bug detection ability between TestAug and CheckList. Each cell shows the failure rate, i.e., the error rate on the held-out validation dataset. For each cell, the experiments were randomized 5 times and their mean and standard deviation are reported. The complete model path of each model can be found from Table 10. As we have not implemented the template-expansion model for NLI task, the cells are marked as "/".

Model type	Original	Unpatched	Patched			
			$\mathcal{T}_{\text{CheckList}}$	$\mathcal{T}_{\text{TestAug}}$	$\mathcal{T}_{\text{TestAug}} \backslash \mathcal{T}_{\text{GPT-3}}$	$\mathcal{T}_{\text{TestAug}} \backslash \mathcal{T}_{\text{Expansion}}$
Sentiment Classification						
ALBERT	7.3	32.6±5.7	13.4±6.5	11.3±10.0	10.6±6.6	**9.6±8.2**
$\text{BERT}_{\text{Base}}$	7.6	33.9±6.1	9.0±4.2	**8.3±4.2**	8.5±1.6	9.9±4.9
DistillBERT	10.0	29.5±10.9	6.5±3.4	**3.9±2.1**	4.9±2.1	5.1±3.3
$\text{RoBERTa}_{\text{Base}}$	5.7	14.2±6.1	3.7±2.3	1.6±1.0	2.7±2.7	**1.4±1.2**
Paraphrase Detection						
ALBERT	9.3	38.1±3.8	7.1±0.8	0.6±0.4	5.8±1.8	**0.4±0.4**
$\text{BERT}_{\text{Base}}$	9.1	36.0±4.9	6.2±1.5	0.5±0.4	5.6±1.1	**0.4±0.3**
DistillBERT	10.3	49.8±10.2	12.5±16.4	**1.1±2.4**	6.4±3.9	7.3±15.8
Natural Language Inference						
ALBERT	9.9	42.8±1.9	30.1±4.2	**23.0±1.6**	/	/
DistillBERT	12.6	34.7±3.6	23.6±6.1	**16.5±3.9**	/	/
$\text{RoBERTa}_{\text{Large}}$	8.1	17.8±4.0	8.3±3.1	**8.0±3.1**	/	/

$\mathcal{T}_{\text{TestAug}}$ after patching, indicating our augmented test suite has a weaker bug detection ability; a similar phenomenon could be found when testing vocabulary capability of $\text{BERT}_{\text{Base}}$ in paraphrase detection task and syntactic and presupposition capability of $\text{RoBERTa}_{\text{Large}}$ in natural language inference task. This shows that, despite the overall ability to find more bugs (Table 3), the additional test cases from GPT-3 do not uniformly contribute to the improvement of each specific linguistic capability.

5.3 Evaluating the Diversity of TestAug Results

5.3.1 Metric

As existing approaches rely on manually created templates, they have low linguistic variations. Such issues can be alleviated with the help of GPT-3. In this section, we evaluate the linguistic diversity of the generated test cases. We use two metrics to evaluate the diversity: first, we leverage the Self-BLUE score to evaluate the diversity of an entire test suite; second, to measure the test case-level diversity, we introduce a new metric, i.e., the average number of unique dependency paths in each test case.

Self-BLEU. Self-BLEU is an extension of the regular BLEU that evaluates the diversity of generated texts (Zhu et al., 2018). Given a list of texts $\hat{\mathcal{Y}} = \{\hat{Y}_1, \hat{Y}_2, \cdots, \hat{Y}_N\}$, Self-BLEU is the average BLEU score between every single sentence and all other sentences,

$$\text{Self-BLEU}(\hat{\mathcal{Y}}) = \frac{1}{N}\sum_{i=1}^{N} \text{BLEU}(\{\hat{Y}_i\}, \hat{\mathcal{Y}}_{\neq i}) \quad (1)$$

When k is fixed, a lower Self-BLEU score indicates a higher diversity of the sentence.

Number of Unique Dependency Paths. We propose to use the number of unique dependency paths to measure the diversity at the test case level. Each path is a path of POS tags connected by edges in the dependency tree (Jurafsky and Martin, 2000). For example, for the sentence "*I (NOUN) love (VERB) chicken (NOUN)*", the unique paths are NOUN→VERB, VERB→NOUN, NOUN→VERB →NOUN.

5.3.2 Analysis

We compare the Self-BLEU and the number of unique dependency paths between TestAug and

CheckList in Table 5, where we control the number of test cases under each test[11]. From Table 5, we can observe that the linguistic diversity of the test suites $\mathcal{T}_{\text{GPT-3}}$ show **substantial** improvement over the template-based counterparts $\mathcal{T}_{\text{CheckList}}$: the Self-BLEU4 score has a decrease at least 5.3% (the natural language inference task) and the number of unique dependency paths is of at least 1.81 times compared to the original test suite (the natural language inference task).

Table 4: TestAug saves manual efforts in generating new test cases and expands the set of available templates.

#Uniuqe Seed Sentences	#Unique New Sentences	#Seed Templates	#New Templates
Sentiment Classification			
292	3275 (11.2×)	29	1441 (49.7×)
Paraphrase Detection			
124	6427 (50.4×)	17	1307 (76.9×)
Natural Language Inference			
150	4976 (33.2×)	50	/

Table 5: Linguistic diversity of test suites.

	Self-BLEU4 (↓)	Number of Unique Dependency Paths (↑)
Sentiment Analysis		
$\mathcal{T}_{\text{TestAug}}$	0.634	480
$\mathcal{T}_{\text{CheckList}}$	0.853	84
Paraphrase Detection		
$\mathcal{T}_{\text{TestAug}}$	0.627	418
$\mathcal{T}_{\text{CheckList}}$	0.775	117
Natural Language Inference		
$\mathcal{T}_{\text{TestAug}}$	0.430	210
$\mathcal{T}_{\text{CheckList}}$	0.454	116

5.4 Evaluating the Validity of TestAug Results

In this section, we evaluate the effectiveness of our pipeline of filtering incorrect test cases (i.e., Section 4.2). As is shown in Table 11 of the Appendix, the two-phase approach successfully filters the invalid cases: the classification accuracy is *consistently* higher than the validity threshold (90% for sentiment classification and paraphrase detection and 80% for natural language inference)[12]. At the same time, the Cohen's κ on test set annotation indicates *substantial* agreement (McHugh, 2012).

Table 6: Annotation Cohen's κ and agreement rate.

	$\frac{\text{Agreement}}{\text{Total}}$	Cohen's κ
Sentiment Analysis	$\frac{438}{461}$ = 95.0%	0.741
Paraphrase Detection	$\frac{365}{401}$ = 91.0%	0.812
Natural Language Inference	$\frac{151}{156}$ = 96.8%	0.746

5.5 Evaluation of the Manual Efforts Saved by TestAug

Finally, we quantitatively evaluate the manual efforts saved by TestAug compared to CheckList; the results are reported in Table 4. When querying GPT-3 with a few hundred sentences per task, we obtained a new set of valid test cases 11.2 to 50.4 times in size. The template expansion in turn increased the number of available templates to at least 49.7 times, reducing manual efforts by at least 98.0% (Table 4)[13]. This result thus shows that TestAug can largely save manual efforts in creating the test suites.

6 Discussion, Conclusions, and Future Work

This paper introduces a novel framework TestAug for capability-based NLP testing. Addressing current system's heavy dependence on manual creation of templates, our framework can save at least 98.0% of the manual annotation effort with GPT-3; meanwhile, the test suites generated with our framework reveal more bugs than existing systems and show better diversity. Our framework guarantees the correctness of the generated cases by removing the invalid output from GPT-3.

The main limitation of TestAug is that GPT-3 fails to generate highly structured test cases, such as cases for extractive question answering. It also struggles to generate cases that require logic or math reasoning. We leave the exploration of these cases for future work.

[11] We sampled 100 unique sentences per test with a fixed seed of 42 for both test suites. This gave us 1100, 1700, and 1200 sentences in sentiment classification, paraphrase detection, and natural language inference task, respectively.

[12] Additional capabilities in NLI tasks are below 80% valid threshold. Automatic filtering these cases out with trained classifiers are left as future work.

[13] New templates do not cost additional manual efforts when considering all templates as a whole, leading to at least $\frac{1441}{1441+29} \times 100\% = 98.0\%$ saving. With the help of our automatic filtering pipeline, both sentence and template counts could be further increased with additional queries.

Ethical Considerations

Annotator Rights

Two of the authors (one male and one female; both identified themselves as Asians) annotated the data following annotation guidelines; the guidelines are discussed and finalized after thorough discussions (the violations of these guidelines are discussed in Section 4.2). We acknowledge the annotators' efforts with shared authorship.

Intended Uses

TestAug's intended use is as a tool to augment template-based test suites with newly generated test cases from GPT-3; two sets of test cases are then used altogether to evaluate an NLP models' linguistic capabilities; we believe this application of existing datasets are consistent with their intended uses. We showed the effectiveness of this system in Section 5. We hope the adoption of TestAug into the NLP model development could make newly built NLP models more linguistically capable. Meanwhile, the TestAug includes GPT-3 as a component, we urge users of our system to follow the OpenAI's usage guidelines [14].

Potential Misuse

TestAug might be misused to overestimate the models' linguistic capabilities. Specifically, even though failures on the test suites show models' shortcomings in a given linguistic capability, the absence of failures does *not* mean the models being tested are free from bugs; it is likely that test suites are not yet capable enough to reveal the model's bugs. We, therefore, call for a judicious interpretation of an NLP model's performance based on TestAug test suites. Moreover, we believe NLP testing is an iterative process; it might take multiple iterations of applying TestAug to reveal the model's issues in linguistic capabilities.

References

Bing Bai, Jian Liang, Guan Zhang, Hao Li, Kun Bai, and Fei Wang. 2021. Why attentions may not be interpretable? *Proceedings of the 27th ACM SIGKDD Conference on Knowledge Discovery & Data Mining*.

Samuel R. Bowman, Gabor Angeli, Christopher Potts, and Christopher D. Manning. 2015. A large annotated corpus for learning natural language inference. In *EMNLP*.

Samuel R Bowman and George E Dahl. 2021. What will it take to fix benchmarking in natural language understanding? *The 2020 Conference of the North American Chapter of the Association for Computational Linguistics - Human Language Technologies (NAACL-HLT2020)*.

Tom B Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. 2020. Language models are few-shot learners. *arXiv preprint arXiv:2005.14165*.

Robin Cooper, Dick Crouch, Jan Van Eijck, Chris Fox, Johan Van Genabith, Jan Jaspars, Hans Kamp, David Milward, Manfred Pinkal, Massimo Poesio, et al. 1996. Using the framework. Technical report, Technical Report LRE 62-051 D-16, The FraCaS Consortium.

Tianyu Gao, Adam Fisch, and Danqi Chen. 2021. Making pre-trained language models better few-shot learners. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 3816–3830, Online. Association for Computational Linguistics.

Mor Geva, Yoav Goldberg, and Jonathan Berant. 2019. Are we modeling the task or the annotator? an investigation of annotator bias in natural language understanding datasets. *ArXiv*, abs/1908.07898.

Max Glockner, Vered Shwartz, and Yoav Goldberg. 2018. Breaking nli systems with sentences that require simple lexical inferences. In *ACL*.

Suchin Gururangan, Swabha Swayamdipta, Omer Levy, Roy Schwartz, Samuel R. Bowman, and Noah A. Smith. 2018. Annotation artifacts in natural language inference data. In *NAACL*.

Paloma Jeretic, Alex Warstadt, Suvrat Bhooshan, and Adina Williams. 2020. Are natural language inference models IMPPRESsive? Learning IMPlicature and PRESupposition. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 8690–8705, Online. Association for Computational Linguistics.

Pratik Joshi, Somak Aditya, Aalok Sathe, and Monojit Choudhury. 2020a. TaxiNLI: Taking a ride up the NLU hill. In *Proceedings of the 24th Conference on Computational Natural Language Learning*, pages 41–55, Online. Association for Computational Linguistics.

Pratik M. Joshi, Somak Aditya, Aalok Sathe, and Monojit Choudhury. 2020b. Taxinli: Taking a ride up the nlu hill. In *CONLL*.

Dan Jurafsky and James H. Martin. 2000. Speech and language processing.

[14] https://beta.openai.com/docs/usage-guidelines

Hannah Rose Kirk, Bertram Vidgen, Paul Röttger, Tristan Thrush, and Scott A Hale. 2021. Hatemoji: A test suite and adversarially-generated dataset for benchmarking and detecting emoji-based hate. *arXiv preprint arXiv:2108.05921*.

Pavneet Singh Kochhar, Ferdian Thung, and David Lo. 2015. Code coverage and test suite effectiveness: Empirical study with real bugs in large systems. In *2015 IEEE 22nd international conference on software analysis, evolution, and reengineering (SANER)*, pages 560–564. IEEE.

Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Veselin Stoyanov, and Luke Zettlemoyer. 2020. BART: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7871–7880, Online. Association for Computational Linguistics.

Xiang Lisa Li and Percy Liang. 2021. Prefix-tuning: Optimizing continuous prompts for generation. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 4582–4597, Online. Association for Computational Linguistics.

Alisa Liu, Swabha Swayamdipta, Noah A. Smith, and Yejin Choi. 2022. Wanli: Worker and ai collaboration for natural language inference dataset creation.

Pengfei Liu, Weizhe Yuan, Jinlan Fu, Zhengbao Jiang, Hiroaki Hayashi, and Graham Neubig. 2021. Pre-train, prompt, and predict: A systematic survey of prompting methods in natural language processing. *ArXiv*, abs/2107.13586.

M. L. McHugh. 2012. Interrater reliability: the kappa statistic. *Biochemia Medica*, 22:276 – 282.

Swaroop Mishra, Daniel Khashabi, Chitta Baral, and Hannaneh Hajishirzi. 2021. Cross-task generalization via natural language crowdsourcing instructions. *arXiv preprint arXiv:2104.08773*.

Adam Poliak, Aparajita Haldar, Rachel Rudinger, J. Edward Hu, Ellie Pavlick, Aaron Steven White, and Benjamin Van Durme. 2018. Collecting diverse natural language inference problems for sentence representation evaluation. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 67–81, Brussels, Belgium. Association for Computational Linguistics.

Emily Reif, Daphne Ippolito, Ann Yuan, Andy Coenen, Chris Callison-Burch, and Jason Wei. 2021. A recipe for arbitrary text style transfer with large language models. *ArXiv*, abs/2109.03910.

Laria Reynolds and Kyle McDonell. 2021a. Prompt programming for large language models: Beyond the few-shot paradigm. In *Extended Abstracts of the 2021 CHI Conference on Human Factors in Computing Systems*, pages 1–7.

Laria Reynolds and Kyle McDonell. 2021b. Prompt programming for large language models: Beyond the few-shot paradigm. *Extended Abstracts of the 2021 CHI Conference on Human Factors in Computing Systems*.

Marco Tulio Ribeiro, Tongshuang Wu, Carlos Guestrin, and Sameer Singh. 2020. Beyond accuracy: Behavioral testing of NLP models with CheckList. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 4902–4912, Online. Association for Computational Linguistics.

Kyle Richardson, Hai Hu, Lawrence Moss, and Ashish Sabharwal. 2020. Probing natural language inference models through semantic fragments. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pages 8713–8721.

Paul Röttger, Bertie Vidgen, Dong Nguyen, Zeerak Waseem, Helen Margetts, and Janet Pierrehumbert. 2021. HateCheck: Functional tests for hate speech detection models. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 41–58, Online. Association for Computational Linguistics.

Felipe Salvatore, Marcelo Finger, and Roberto Hirata Jr. 2019. A logical-based corpus for cross-lingual evaluation. In *Proceedings of the 2nd Workshop on Deep Learning Approaches for Low-Resource NLP (DeepLo 2019)*, pages 22–30, Hong Kong, China. Association for Computational Linguistics.

Timo Schick and Hinrich Schütze. 2021. Generating datasets with pretrained language models. In *EMNLP*.

Taylor Shin, Yasaman Razeghi, Robert L. Logan IV, Eric Wallace, and Sameer Singh. 2020. AutoPrompt: Eliciting Knowledge from Language Models with Automatically Generated Prompts. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 4222–4235, Online. Association for Computational Linguistics.

Richard Socher, Alex Perelygin, Jean Wu, Jason Chuang, Christopher D. Manning, Andrew Ng, and Christopher Potts. 2013. Recursive deep models for semantic compositionality over a sentiment treebank. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*, pages 1631–1642, Seattle, Washington, USA. Association for Computational Linguistics.

Ishan Tarunesh, Somak Aditya, and Monojit Choudhury. 2021. Lonli: An extensible framework for testing diverse logical reasoning capabilities for nli. In *Thirty-Sixth AAAI Conference on Artificial Intelligence (AAAI-22)*, volume abs/2112.02333.

Betty van Aken, Sebastian Herrmann, and Alexander Löser. 2021. What do you see in this patient? behavioral testing of clinical nlp models. *NeurIPS 2021 Research2Clinics Workshop, Bridging the Gap: From Machine Learning Research to Clinical Practice*.

Jun Wang, Chang Xu, Francisco Guzmán, Ahmed El-Kishky, Benjamin Rubinstein, and Trevor Cohn. 2021. As easy as 1, 2, 3: Behavioural testing of NMT systems for numerical translation. In *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*, pages 4711–4717, Online. Association for Computational Linguistics.

Peter West, Chandrasekhar Bhagavatula, Jack Hessel, Jena D. Hwang, Liwei Jiang, Ronan Le Bras, Ximing Lu, Sean Welleck, and Yejin Choi. 2021. Symbolic knowledge distillation: from general language models to commonsense models. *ArXiv*, abs/2110.07178.

Hitomi Yanaka, Koji Mineshima, Daisuke Bekki, Kentaro Inui, Satoshi Sekine, Lasha Abzianidze, and Johan Bos. 2019. HELP: A dataset for identifying shortcomings of neural models in monotonicity reasoning. In *Proceedings of the Eighth Joint Conference on Lexical and Computational Semantics (*SEM 2019)*, pages 250–255, Minneapolis, Minnesota. Association for Computational Linguistics.

Jie M. Zhang, Mark Harman, Lei Ma, and Yang Liu. 2019. Machine learning testing: Survey, landscapes and horizons. *ArXiv*, abs/1906.10742.

Yaoming Zhu, Sidi Lu, Lei Zheng, Jiaxian Guo, Weinan Zhang, Jun Wang, and Yong Yu. 2018. Texygen: A benchmarking platform for text generation models. *The 41st International ACM SIGIR Conference on Research & Development in Information Retrieval*.

A Appendix

Table 7: Prompt designs for paraphrase detection and natural language inference tasks.

Paraphrase Detection

`Two sentences are equivalent when using according to.`
- {{ Who do analysts think is the smartest footballer in the world? }
- { Who is the smartest footballer in the world according to analysts? }}
- {{ Who do students think is the top woman in the world? }
- { Who is the top woman in the world according to students? }}
- {{ Who do readers think is the worst gamer in the world? }
- { Who is the worst gamer in the world according to readers? }}
- {{ `What does the data say about the most popular baby names? }`
`- { What are the most popular baby names according to the data? }`

Natural Language Inference

`Write a pair of sentences that have the same relationship as the previous examples. Examples:`
- {{ Philip, Charles and Colin are the only children of Henry. }
- { Henry has exactly 3 children. }}
- {{ Grace, Thomas and Helen are the only children of Andrea. }
- { Andrea has exactly 3 children. }}
- {{ Don has 2 dollars. He received 8 more dollars. }
- { Don now has 10 dollars. }}
- {{ `Mary has a cat. She also has a dog. }`
`- { Mary has two pets. }}`

Table 8: Hyperperparamer choice for model fine-tuning

Hyperparameter	Value
Learning rate	$5e-6$
Batch size	16
Number of training epochs	10
Max. sequence length	128
Seed	42

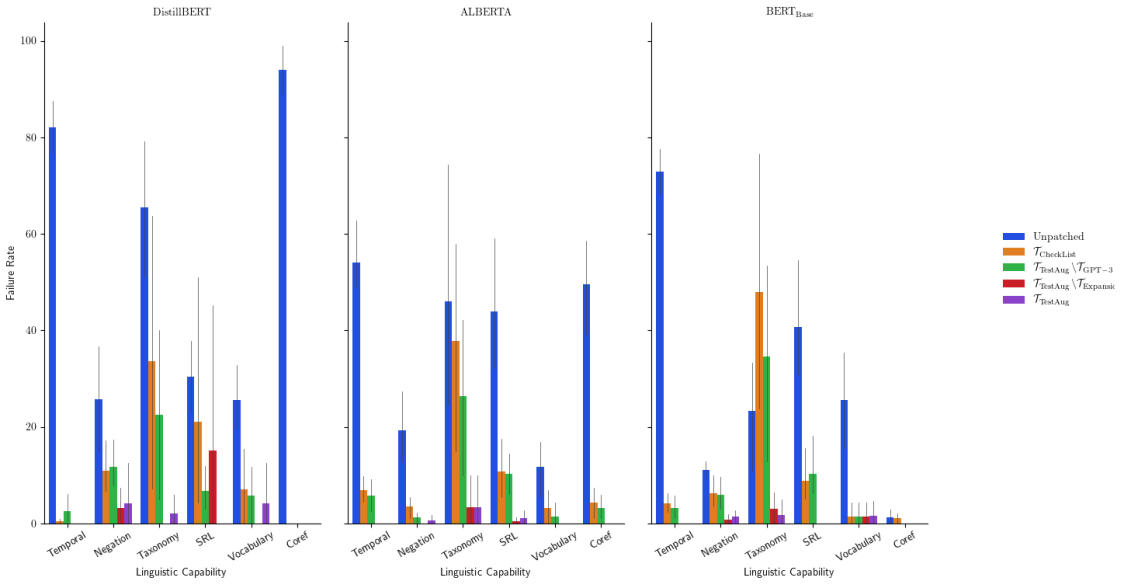

(a) Paraphrase Detection

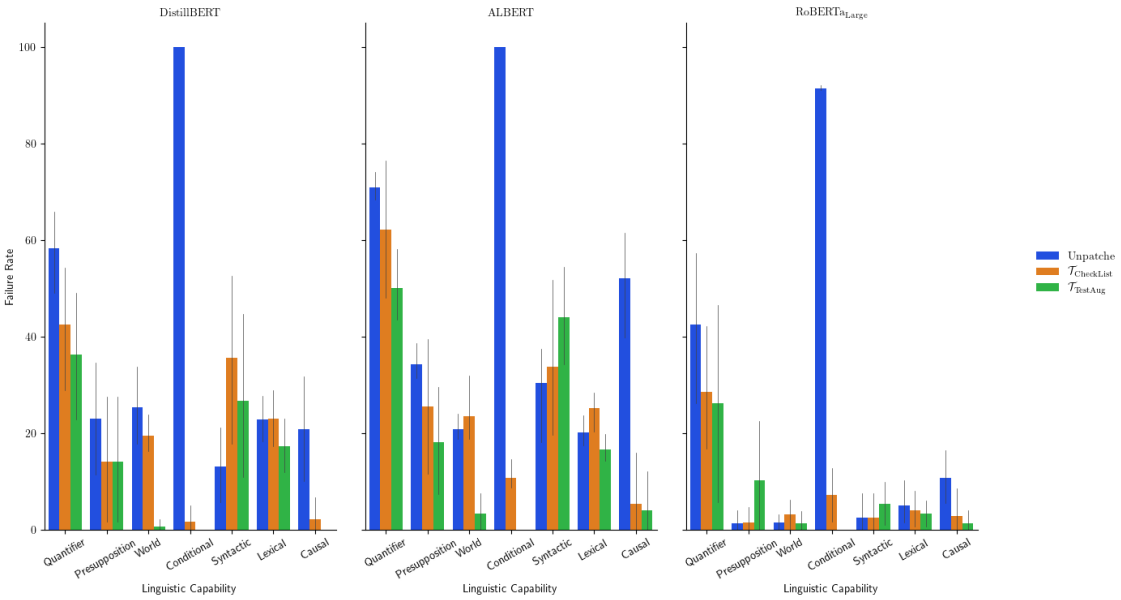

(b) Natural Language Inference

Figure 3: Capability-wise error rates on paraphrase detection and natural language inference tasks.

Table 9: Linguistic capabilities that appeared in the experiments and their explanations with corresponding examples. The description of the tested linguistic capabilities follow the taxonomy and definition provided in previous work (Ribeiro et al., 2020; Tarunesh et al., 2021; Joshi et al., 2020b).

Linguistic Capability	Explanation	Template and Example
		Sentiment Analysis
Negation	Resolving "not"	{neg} {pos_verb_present} {the} {air_noun}. I didn't admire that service.
SRL	Resolving subjects and objects	Do I think {it} {be} {a:pos_adj} {air_noun}? No Do I think that is an amazing aircraft? No
Temporal	Resolving temporal changes	I {neg_verb_present} this airline, {change} in the past I would {pos_verb_present} it. I regret this airline, although in the past I would appreciate it.
Vocabulary	Resolving word choice variations	{it} {air_noun} {be} {neg_adj}. That food was ugly.
		Paraphrase Detection
Coref	Resolving male and female names	S1: If {female} and {male} were alone, do you think he would reject her? S2: If {female} and {male} were alone, do you think she would reject him? S1: If Julie and Roy were alone, do you think he would reject her? S2: If Julie and Roy were alone, do you think she would reject him?
Negation	Resolving "not"	S1: What are things {a:noun} should worry about? S2: What are things {a:noun} should not worry about? S1: What are things a friend should worry about? S2: What are things a friend should not worry about?
SRL	Resolving subjects and objects	S1: Did {first_name} {verb[0]} the {obj}? S2: Was {first_name} {verb[1]} by the {obj}? S1: Did Sam remember the factory? S2: Was Sam remembered by the factory?
Taxonomy	Resolving external taxonomic hierarchy	S1: How can I become {a:x[1]} person? S2: How can I become {a:x[0]} person? S1: How can I become a frightened person? S2: How can I become a scared person?
Temporal	Resolving temporal changes	S1: Is {first_name} {last_name} {a:noun}? S2: Did {first_name} {last_name} use to be {a:noun}? S1: Is Dorothy Clarke an agent? S2: Did Dorothy Clarke use to be an agent?
Vocabulary	Resolving word choice variations	S1: How can I become less {x[1]}? S2: How can I become more {x[1]}? S1: How can I become less active? S2: How can I become more active?
		Natural Language Inference
Causal	Resolving actions between one object and two entities	P: {NAME} moved the {OBJECT_TABLE} to {LOCATION_HOUSE1} from {LOCATION_HOUSE2}. H: The {OBJECT_TABLE} is now in {LOCATION_HOUSE1}. P: George moved the notebook to study room from bedroom. H: The notebook is now in study room.
Conditional	Resolving reasoning over conditions	P: If {NAME1} comes to the {LOCATION}, {NAME2} won't come. {NAME1} did not come to the {LOCATION}. H: {NAME2} didn't come to the {LOCATION}. P: If Kim comes to the park, William won't come. Kim did not come to the park. H: William didn't come to the park.
Lexical	Resolving word choice variations	P: {NAME} was born in {COUNTRY1}. H: {NAME} is born in {COUNTRY2}. P: Emily was born in Germany. H: Emily is born in Malaysia.
Presupposition	Resolving implications	P: {NAME}'s {T12_RELATION} is {ADJECTIVE_OF_PERSON}. H: {NAME} has {A} {T12_RELATION}. P: Florence's brother is intolerant. H: Florence has a brother.
Quantifier	Resolving "all" (universal quantification) and "some", "none" (existential quantification)	P: None of the {OBJECTS} are {COLOUR1} in colour. H: Some of the {OBJECTS} are {COLOUR2} in colour. P: None of the cars are maroon in colour. H: Some of the cars are pink in colour.
Syntactic	Resolving ellipsis	P: {NAME} tried but wasn't able to {VERB}. H: {NAME} didn't try to {VERB}. P: Alan tried but wasn't able to give. H: Alan didn't try to give.
World	Resolving world knowledge such as geography	P: {NAME} lives in {T4_CAPITAL1}. H: {NAME} lives in {T4_COUNTRY2}. P: Ken lives in Kathmandu. H: Ken lives in North Korea.

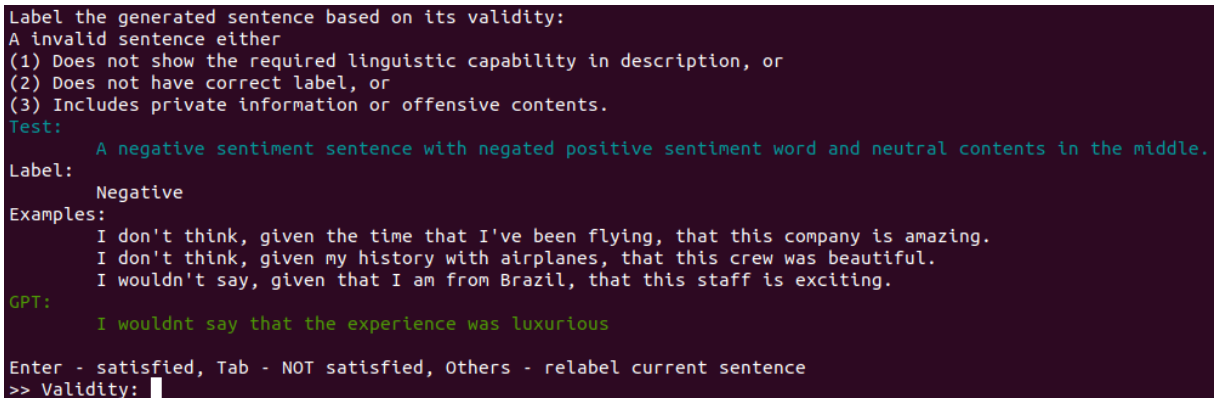

Figure 4: The command line interface for data annotation. Annotators are given a test and three associated test cases from the template-based test suite; they are asked to the annotate the validity of the GPT-3-generated sentences. Annotators are reminded of the guidelines for filtering invalid samples when labeling each sentence (shown at the top of the interface). We communicated explicitly for the intended uses of the annotated datasets before the annotation.

Table 10: The fine-tuned models we evaluated in this paper.

Model name	Task	Size	Checkpoint Identifier
DistillBERT	Sentiment Analysis	Small	`textattack/distilbert-base-cased-SST-2`
ALBERT	Sentiment Analysis	Small	`textattack/albert-base-v2-SST-2`
$\text{BERT}_{\text{Base}}$	Sentiment Analysis	Base	`textattack/bert-base-uncased-SST-2`
$\text{RoBERTa}_{\text{Base}}$	Sentiment Analysis	Base	`textattack/roberta-base-SST-2`
DistillBERT	Paraphrase Detection	Small	`textattack/distilbert-base-cased-QQP`
ALBERTA	Paraphrase Detection	Small	`textattack/albert-base-v2-QQP`
$\text{BERT}_{\text{Base}}$	Paraphrase Detection	Base	`textattack/bert-base-uncased-QQP`
DistillBERT	Natural Language Inference	Small	`textattack/distilbert-base-cased-snli`
ALBERT	Natural Language Inference	Small	`textattack/albert-base-v2-snli`
$\text{RoBERTa}_{\text{Large}}$	Natural Language Inference	Large	`ynie/roberta-large-snli_mnli_fever_anli_R1_R2_R3-nli`

Table 11: Proportion of valid sentences and performance of trained classifiers for automatic filtering.

Linguistic Capability	Test	Accuracy	F1	Proportion of Valid Test Cases
	Sentiment Analysis			
SRL	A negative sentiment sentence with negative sentiment question and word yes as the answer.	/	/	1.000
SRL	A positive sentiment sentence with positive sentiment question and word yes as the answer.	/	/	0.994
SRL	My opinion is more important than others' when expressing positive sentiment.	/	/	0.948
SRL	A negative sentiment sentence with positive sentiment question and word no as the answer.	/	/	0.928
Temporal	I used to have negative sentiment to something, but now I have positive sentiment to it.	/	/	0.922
Negation	A negative sentiment sentence with negated positive word.	/	/	0.910
SRL	My opinion is more important than others' when expressing negative sentiment.	/	/	0.900
Temporal	I used to have positive sentiment to something, but now I have negative sentiment to it.	0.942	0.967	0.873
Negation	I thought something was positive, but it was negative.	0.934	0.961	0.860
Vocabulary	A negative sentiment sentence with negative words.	0.897	0.932	0.804
Negation	A negative sentiment sentence with negated positive sentiment word and neutral contents in the middle.	0.915	0.935	0.653
	Paraphrase Detection			
Temporal	Two sentences are different when talking about a person's current job and his or her previous job.	/	/	1.000
Negation	Two sentences are different when talking about someone should and should not do something.	/	/	0.973
Temporal	Two sentences are different when talking about a person's current job and his or her future job.	/	/	0.964
Vocabulary	Two sentences are different when adjectives are modified by more and less.	/	/	0.912
Taxonomy	Two sentences are equivalent when the nouns are modified by synonymous adjectives.	0.889	0.933	0.885
Coref	Two sentences are different when swapping the subjects and objects.	0.980	0.989	0.843
Temporal	Two sentences are different when talking about doing something before and after another thing.	0.982	0.989	0.800
Temporal	Two sentences are different when describing doing something before and after some specific time.	0.982	0.988	0.782
Negation	Two sentences are different when talking about the properties of doing or not doing something.	0.965	0.977	0.754
Vocabulary	A sentence with the noun modified by an adjective is equivalent to the sentence without adjective.	0.920	0.946	0.740
SRL	Two sentences are equivalent when using according to.	0.958	0.971	0.708
Negation	Two sentences are different when describing a person with adjective and a clause including the negation of the same adjective.	0.871	0.913	0.690
Taxonomy	Two sentences are equivalent when one has an adjective modified by more and the other one has an antonym modified by less.	0.920	0.909	0.511
Coref	Two sentences are different when referring someone's family using different pronouns.	0.940	0.940	0.500
SRL	Two sentences are different when swapping active and passive action.	0.930	0.933	0.471
	Natural Language Inference			
Lexical	/	/	/	0.975
Syntactic	/	/	/	0.936
Presupposition	/	/	/	0.903
World	/	/	/	0.902
Quantifier	/	/	/	0.895
Causal	/	/	/	0.857
Conditional	/	/	/	0.834

Table 12: Creating new templates based on test cases generated by GPT-3.

Original Template and Test Case	Generated Test Case	New Template
	Sentiment Analysis	
No one {pos_verb_present}s {the} {air_noun}. No one enjoys that seat.	This is not an easy service to appreciate. That customer service was not fun. I don't think your customer service is admired.	This is not an easy {air_noun} to appreciate. That customer {air_noun} was not fun. I don't think your customer {air_noun} is admired.
	Paraphrase Detection	
Is it {mid} to {activity} before {hour}{ampm}? Is it {mid} to {activity} after {hour}{ampm}? Is it healthy to drink before 10am? Is it healthy to drink after 10am?	Is it bad to drink before 8pm Is it bad to drink after 8pm Is it acceptable to drink before 2pm Is it acceptable to drink after 2pm Is it advisable to eat before 8pm Is it advisable to eat after 8pm	Is it bad to {activity} before 8pm Is it bad to {activity} after 8pm Is it {mid} to {activity} before 2pm Is it {mid} to {activity} after 2pm Is it advisable to {activity} before 8pm Is it advisable to {activity} after 8pm

KoCHET : a Korean Cultural Heritage corpus for Entity-related Tasks

Gyeongmin Kim[*], **Jinsung Kim**[*], **Junyoung Son**[*], **Heuiseok Lim**[†]
Korea University, Korea
{totoro4007, jin62304, s0ny, limhseok}@korea.ac.kr

Abstract

As digitized traditional cultural heritage documents have rapidly increased, resulting in an increased need for preservation and management, practical recognition of entities and typification of their classes has become essential. To achieve this, we propose **KoCHET** - a Korean cultural heritage corpus for the typical entity-related tasks, i.e., named entity recognition (NER), relation extraction (RE), and entity typing (ET). Advised by cultural heritage experts based on the data construction guidelines of government-affiliated organizations, **KoCHET** consists of respectively 112,362, 38,765, 113,198 examples for NER, RE, and ET tasks, covering all entity types related to Korean cultural heritage. Moreover, unlike the existing public corpora, modified redistribution can be allowed both domestic and foreign researchers. Our experimental results make the practical usability of **KoCHET** more valuable in terms of cultural heritage. We also provide practical insights of **KoCHET** in terms of statistical and linguistic analysis. Our corpus is freely available at https://github.com/Gyeongmin47/KoCHET.

1 Introduction

Recently there has been an increasing interest in the preservation of national historical artifacts and traditional cultural heritage, and also grows up the importance of effective management of them through digitization and archival. As the amount of digitized information materials increases rapidly, information extraction (IE) tasks in natural language processing (NLP), such as named entity recognition (NER), relation extraction (RE), and entity typing (ET), have become an essential and fundamental step in the field of historical document analysis.

Despite the necessity of a well-refined entity-centric corpus specialized in domestic cultural heritage, unfortunately, there no exists any cultural heritage domain-specialized corpus in Korean. Moreover, conventional entity-related systems deal only with a coarse set of entity types such as person, location, and organization which is significantly limited in terms of application (Kim et al., 2020). This absence of cultural heritage domain-specialized corpus and narrow coverage of entity types hinders the effective digitization of domestic historical documents because training the model with general corpus for entity-related tasks cannot afford to learn enough significant entity types such as pagodas, historical sites and intangible heritage, and their relations. Furthermore, not in the cultural heritage domain, the existing entity-related datasets supervised by the public institutions have a complicated procedure for data acquisition, and they are also restricted from modification and redistribution. These cumbersome procedures and restrictions have been stumbling blocks for researchers against the rapid increase in digitized cultural heritage materials over the past few decades.

To address these difficulties against the conservation of Korean cultural heritage, we introduce a new dataset collection called **KoCHET** - **Ko**rean **C**ultural **H**eritage corpus for **E**ntity-related **T**asks, a high-quality Korean cultural heritage domain-specialized dataset for NER, RE, and ET tasks. For corpus construction, we crawled the e-museum digitized data of the National Museum of Korea[1] (including data from all 50 museums) as the source text which is for the interested public. We selectively used resources from the museums in which the details of artifacts were registered; moreover, for the completeness of the attribute data, we limited the chronological range of the data from the prehistoric era to the Korean Empire era, excluding the Japanese colonial period. For the annotation, the categorization for classes and attributes appropriate was defined and developed following

[*] These authors have equally contributed to this work
[†] Corresponding author

[1] https://www.emuseum.go.kr/

Proceedings of the 29th International Conference on Computational Linguistics, pages 3496–3505
October 12–17, 2022.

the *2020 Named Entity Corpus Research Analysis*[2] which was published under the guidelines as institutional organizations.

As our corpus focuses on the entity features, it has more detailed and abundant entity types including diverse cultural heritage artifacts, compared to the existing accessible datasets that aim to deal with several downstream tasks in addition to entity-related tasks. Furthermore, the ET of **KoCHET** is the first freely available corpus for the ET task in Korea. In addition to providing these values, this paper provides detailed statistics and linguistic analysis of **KoCHET** for each entity-related task to demonstrate their applicability and enhance understanding of the data, along with baseline experiments with language models.

Our contributions are summarized as follows:

- We introduce **KoCHET** designed for entity-related tasks. This guarantees a high-quality corpus without restrictions regarding modification and redistribution. Moreover, to the best of our knowledge, the ET corpus is the first proposed corpus in Korean.
- We categorized the detailed entity types specialized in the cultural heritage domain, which is essential for preserving our cultural and historical artifacts, thereby contributing as an alternative to the increased demand for the digitalized archiving of cultural heritage documents.
- We prove the applicability of our entity-abundant corpus in each task by providing statistics and linguistic analysis, along with the experiments with pre-trained language models.

2 Related Works

As domains that require expertise, such as the cultural heritage, contain entities or relationships that rarely appear in general domains, the necessity of a corpus specialized in the domain is obvious. Despite such demand, Korean does not yet have a corpus specialized in the cultural heritage area, unlike other languages.

2.1 General cultural heritage corpora

There have been the disclosures of corpora in an effort to preserve traditional culture including the cultural heritage, composing data from the perspective of the entity-related tasks that we deal with. For example, these include a Czech NER corpus constructed based on public optical character recognition data of Czech historical newspapers (Hubková et al., 2020), a Chinese corpus suitable for the computational analysis of historical lexicon and semantic change (Zinin and Xu, 2020), and an English corpus that is one of the most commonly used large corpora in diachronic studies in English (Alatrash et al., 2020).

2.2 Korean public corpora

The National Institute of Korean Language , which is an institution that has established the norms for Korean linguistics, constructed a large-scale dataset[3] for the study of new computational linguistics of Korean (Kim, 2006).

AI HUB is a massive dataset integration platform[4] hosted by the National Information Society Agency (NIA)[5], a government-affiliated organization. To support the development of the Korean artificial intelligence industry for the NLP field, the NIA disclosed domain-specific corpora and 27 datasets have been released or are being prepared.

Electronics and Telecommunications Research Institute , as part of the Exo-brain project[6], provides corpora for NLP tasks such as morphological analysis, entity recognition, dependency parsing, and question answering, and guidelines for building such high-quality corpora[7]. In addition to public datasets opened by public institutions, there is a Korean dataset publicly available for free without the requirement for an access request.

Korean Language Understanding Evaluation (KLUE) dataset was recently released to evaluate the ability of Korean models to understand natural languages with eight diverse and typical tasks (Park et al., 2021b). The tasks include natural language inference, semantic textual similarity, dependency parsing, NER, and RE.

3 KoCHET

Following the guidelines of Korean institutional organizations, **KoCHET** is a domain specialized

[2] https://www.korean.go.kr

[3] https://stdict.korean.go.kr/

[4] https://aihub.or.kr/

[5] https://www.nia.or.kr/

[6] http://exobrain.kr/pages/ko/result/outputs.jsp

[7] https://www.etri.re.kr/

corpus for cultural heritage, which ensures quality and can be freely accessed. In this section, we report the annotation process and guidelines in detail.

3.1 Annotation Process

To improve the quality of annotations on our entity-rich corpus related to cultural heritage, we conducted the annotation process based on expertise in the cultural heritage domain.

Annotation Guidelines The raw corpus annotated by each annotator is equally divided by the category. The annotators were instructed to follow two types of rules by the aforementioned entity guidelines in Section 1; one is related to tagging units and categories, and the other is the principle of unique tagging. The minimum unit is based on one word for the tagging units and categories. In addition, it is applied only to cases written in Korean, where the notation is possible. It is not tagged in the case of Chinese characters and English, but if it is read in Korean, it is included in the tagging range. For the principle of unique tagging, there are cases of duplication in entities that belong to two or more semantic regions. This guideline grants a single tag to a semantically suitable word and refers to assigning only one tag by prioritizing it accordingly. There are two cases in which this principle should be applied. The first case is where the entity belongs to two semantic categories regardless of the context. The second refers to the case where it may vary depending on the context. In both cases, tagging is determined according to the pre-defined priority.

Annotator Training and Cross-Checking We recruited 34 college and graduate annotators who have been professionally educated on the cultural heritage domain in Korea to participate in the annotation process. All annotators were trained for a week, and each of them was familiarized with the annotation guideline and conducted practice annotation on test samples. The annotation team met once every week to review and discuss each member's work during the annotation process. All entity types and relations were reviewed by four cross-checking annotators, afterward, were additionally checked by two expert supervisors. The discrepancy between annotators on the annotated entity types and relations is also discussed and agreed upon in the period. These procedures allowed the reliability and validity of **KOCHET** on the cultural heritage objects to be improved.

3.2 Schema for Task Annotation

3.2.1 Named Entity Recognition

Label	Train	Dev	Test
	Counts (%)		
Artifacts (AF)	91,453 (35.57)	11,374 (35.54)	11,366 (35.35)
Person (PS)	51,758 (20.13)	6,455 (20.17)	6,744 (20.97)
Term (TM)	25,781 (10.02)	3,175 (9.92)	3,159 (9.82)
Date (DT)	23,636 (9.19)	2,943 (9.20)	3,078 (9.57)
Political location (LCP)	20,076 (7.80)	2,375 (7.42)	2,384 (7.41)
Civilization (CV)	15,404 (5.99)	1,929 (6.03)	1,835 (5.71)
Material (MT)	8,893 (3.45)	1,160 (3.62)	1,046 (3.25)
Location (LC)	6,881 (2.67)	857 (2.68)	881 (2.74)
Animal (AM)	4,376 (1.70)	578 (1.81)	566 (1.76)
Plant (PT)	3,952 (1.53)	549 (1.72)	498 (1.55)
Geographical location (LCG)	2,821 (1.09)	354 (1.11)	348 (1.08)
Event (EV)	2,045 (0.79)	254 (0.79)	248 (0.77)

Table 1: The counts of entities and their distributions (%) in our NER data.

As described in Table 1, we defined 12 entity types. They were tagged with the character-level beginning-inside-outside (BIO) tagging scheme, which is the generally adopted method for sequence labeling problems. For example, "아시아 (Asia): Geographical Location (LCG)" is tagged as "아: B-LCG," "시: I-LCG," "아: I-LCG." Therefore, we evaluated the model not only with entity-level F1 score but also with character-level F1 score (Park et al., 2021b).

Label Description

- **Artifacts (AF)** generally refer to objects created by humans corresponding to common and proper nouns and also include cultural properties. Therefore, artificial materials such as buildings, civil engineering constructions, playground names, apartments, and bridges fall under this category.
- **Person (PS)** is a category for content related to people, including real persons, mythical figures, fictional characters in games/novels, occupations, and human relationships.
- **Term (TM)** includes the color, direction, shape, or form that describes an artifact. Patterns and drawings are classified as TM, owing to the characteristics of movable cultural properties.
- **Civilization (CV)** is defined as terms related to civilization/culture. It targets words classified by detailed civilizations/cultures, such as clothing and food.

- **Date (DT)** includes all entities related to date and time, such as date, period, specific day, or season, month, year, era/dynasty. However, in the case of an unclear period that cannot be tagged with a separate entity, tagging is not performed.

- **Material (MT)** includes a substance used as a material or an expression for the substance. In other words, it indicates the entity corresponding to the detailed classification of a substance (metal, rock, wood, etc.). When an entity can be tagged as both natural objects (AM, PT) and MT, tagging as MT takes precedence.

- **Geographical location (LCG)**, **Political location (LCP)**, and **Location (LC)** are defined as geographical names, administrative districts, and other places, respectively.

- **Animal (AM)** and **Plant (PT)** are defined as animals and plants, respectively, excluding humans. If it is applied as a subject of a picture, it is also included in the category of animals and plants.

- **Event (EV)** contains entities for a specific event/accident. In principle, social movements and declarations, wars, revolutions, events, festivals, etc., fall under this category and should be classified only if they exist as a separate entity.

3.2.2 Relation Extraction

Unlike the other existing corpora, our corpus has the advantage of capturing various relationships between multiple entities that are included in a sentence because more than one relation can exist per raw sentence. We consider the relations between annotated entities in the NER annotation procedure. In the case of certain tokens, it can be a subject or an object depending on the relationship with other tokens. A relationship in the form of a self-relationship between identical tokens does not exist.

As shown in Table 2, our RE corpus consists of 14 labels, and these were defined based on the Encyves ontology research of the National Culture Research Institute[8].

Label Description

- "**A depicts B**" implies the relationship between an object and its color, shape or pattern, etc. For example, "Green Door" corresponds to this relationship. It can also represent a descriptive relationship such as "Picture of a place-the place where it was taken" or "Picture of a person-the person who is the object of the painting."

- "**A documents B**" implies "~ records -." ;a relationship such as "Record-The person who records it" can be represented by this. It also indicates the relationship like a record written on an object such as "Postcard-Explanation" or a specific language written on a document such as "Record-Chinese characters."

- "**A hasSection B**" indicates "~ is located at -." It represents the relationship between a statue, building, or specific attraction and a location, such as a certain city and place.

- "**A servedAs B**" implies "~ is the role of -," which corresponds to the relationship between a person, and his/her position or occupation, etc.

- "**A hasCreated B**" demonstrates, for example, "Person-Documents" or "Person-Painting," which refers to the relationship between a person and a document such as a book, map, or drawing, or his/her activities to record works.

- "**A OriginatedIn B**" means "~ is discovered at –" or "~ is produced at -(time)." It indicates that cultural property is produced at a specific time such as "Craft-Year" or is discovered at a particular place such as "Object-Place," or is produced at a certain site such as "Document-Place." For example, the relation

Label	Train	Dev	Test
	Counts (%)		
A depicts B	14,157 (22.09)	1,803 (22.45)	1,711 (21.85)
A documents B	10,214 (15.94)	1,244 (15.49)	1,220 (15.58)
A hasSection B	6,542 (10.21)	818 (10.19)	776 (9.91)
A servedAs B	6,546 (10.22)	780 (9.71)	740 (9.45)
A hasCreated B	6,136 (9.58)	759 (9.45)	744 (9.50)
A OriginatedIn B	5,456 (8.51)	679 (8.45)	663 (8.47)
A consistsOf B	4,331 (6.76)	569 (7.09)	586 (7.48)
A isConnectedWith B	3,489 (5.44)	501 (6.24)	461 (5.89)
A fallsWithin B	3,454 (5.39)	415 (5.17)	483 (6.17)
A isUsedIn B	1,906 (2.97)	238 (2.96)	244 (3.12)
A hasTime B	934 (1.46)	111 (1.38)	95 (1.21)
A wears B	798 (1.25)	97 (1.21)	86 (1.10)
A hasCarriedOut B	112 (0.17)	15 (0.19)	19 (0.24)
A hasDestroyed B	5 (0.01)	2 (0.02)	3 (0.04)

Table 2: Relation counts and distributions (%) for our RE corpus.

[8] http://dh.aks.ac.kr/Encyves/wiki

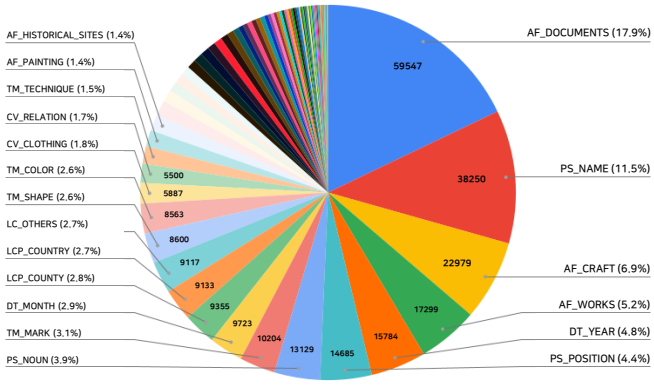

ARTIFACT	MATERIAL	PLANT	DATE	CIVILIZATION
DOCUMENTS CRAFT WORKS PAINTING HISTORICAL_SITES MONUMENT INTANGIBLE_HERITAGE PAGODA TRANSPORT WEAPON BUILDING MUSICAL_INSTRUMENT ROAD RELIC	SOIL PAPER FIBER ROCK METAL WOOD BONE JEWELRY SKIN GRASS SYNTHETIC SEED RUBBER OTHERS	PART FLOWER TREE TYPE GRASS FRUIT OTHERS	YEAR MONTH DURATION DYNASTY DAY SEASON GEOAGE	CLOTHING REALTION LANGUAGE BUILDING_TYPE FOOD CURRENCY TRIBE LAW DRINK POLICY TAX SPORTS SPORTS_INST NAME FOOD_STYLE
		LCP	ANIMAL	
		COUNTY COUNTRY CITY PROVINCE CAPITALCITY	PART BIRD MAMMALIA TYPE FISH INSECT REPTILIA AMPIHBIA OTHERS	
EVENT	LCG	TERM	LC	PERSON
FESTIVAL ACTIVITY WAR_REVOLUTION SPORTS OTHERS	MOUNTAIN RIVER ISLAND CONTINENT BAY	MARK SHAPE COLOR TECHNIQUE	SPACE OTHERS	NAME POSITION NOUN IMAGINARY

Figure 1: Visualization of all the labels that cover 84% of the entity types is shown on the left side, and 106 general and fine-grained entities with their distributions (%) are shown on the right side.

between earrings and tombs or a newspaper and the company of the newspaper fall into this.

- "**A consistsOf B**" refers to the relation between an object and its raw ingredients, such as soil, iron, and wood that constitute an object.
- "**A isConnectedWith B**" represents a person-to-person association. The relationships between two positions or a person and the position he or she holds do not fall into this.
- "**A fallsWithin B**" implies "~ is denominated as -." It indicates the relationship of alternate names such as "Person-Specific name," or between a name and designation in front of the name, or between words that refer to synonymous concepts such as "Verse-Poetry."
- "**A isUsedIn B**" indicates "~ is used for the purpose of -" or literally "~ is used in -." For example, it can also indicate the material used for a certain object, such as "Raw material-Clothes." The relationship between an object and the place where the object is used, such as a signboard and a palace, or the relationship between certain means of performing a function and an object such as "Bowl-Rice cake" can correspond to this category.
- "**A hasTime B**" implies "~ has happened at -." For example, it can indicate the relationship between a particular event and a specific date, such as "Presidential election-1928." The relation between a specific date and a certain work, such as the year of production of a work and the year of construction of a building, can fall under this category, for example, "Year-Craftwork."
- "**A wears B**" implies "~ puts - on." For instance, not only clothes such as school uniforms but also crafts, etc. may correspond to the object argument.
- "**A hasCarriedOut B**" indicates "- is caused by ~." It can represent a relationship between a specific organization or group and an event conducted by it, such as a festival or social movement.
- "**A hasDestroyed B**" implies the event that caused destruction such as "War-Destroyed place," or the collapse of a country in a specific year such as "Country-Year," or the relationship in which a building, structure, monument, etc. is destroyed at a particular period.

3.2.3 Fine-grained Entity Typing

Given a sentence and entity mention within it, the ET task predicts a set of noun phrases that describe the mention type. For example, in "김홍도는 조선 후기의 화가이다. (*Kim Hong-do was a painter of the Joseon era of Korea.*)," *Joseon* should be typed as "dynasty/Date" and not "country/Location." This typification is crucial for context-sensitive tasks such as RE, coreference resolution, and question answering (e.g., "In which era was Kim Hong-do, an artist?"). Unlike high resource languages, we found that the Korean corpus for the ET task has not been released. In dealing with this data scarcity problem and promoting universal studies, we release a Korean ET task corpus for the first time, to the best of our knowledge.

Sentence with Entity Mention	Entity Types
조선시대에는 전통 관습을 잇기 위한 많은 향로가 제작되었다. (In the **Joseon dynasty**, many fragrance burners were created for traditional customs.)	**DT_DYNASTY**, DT_DURATION LCP_COUNTRY, LCP_CITY, LCP_COUNTY LC_OTHERS, AF_DOCUMENTS
노란 바탕의 **모란**이 양쪽에 그려져 있다. (The yellow background **peony** is drawn on both sides.)	**PT_FLOWER**, PT_TYPE, PT_OTHERS, TM_SHAPE
19세기 후반 청주의 재정을 파악할 수 있는 자료가 있다. (There are data to comprehend the finances of Cheongju in the **late 19th century**.)	**DT_YEAR**, DT_DYNASTY, DT_DURATION

Table 3: Examples including entity mentions and their fine-grained entity types. Entity mentions and the correct types in the given context are bold. All fine-grained entity types are shown in Figure 1.

The schema for the ET task was designed with reference to the data construction process of the Fine-Grained Entity Recognition dataset (Ling and Weld, 2012). Considering the properties of the cultural heritage domain, we categorized the 12 general entity types aforementioned in the NER task (Section 3.2.1) into a fine-grained set of 94 types with detailed meanings. Particularly, the cultural taxonomy defined in the *Cultural Properties Protection Law*[9] was applied to AF, and the 2004 Cavalier-Smith's classification system (Cavalier-Smith, 2004) was applied to the biological scope of PT and AM. All fine-grained entity types are detailed in Figure 1.

The fine-grained entities for entity-related downstream tasks in the cultural heritage domain enable a more detailed contextualized representation for each entity mention than the previous typing schemas, which only predict relatively coarse types of entities. Table 3 lists three example sentences with entity mention that can represent several fine-grained types. Given a sentence with an entity mention, the appropriate type that describes the role of the entity span in the sentence should be predicted. Our fine-grained entity types can embrace all the existing general types and categorize them in greater detail. Accordingly, they can let models understand richly the noun phrases including entity, compared to when the models are trained to predict only relatively coarse types. For Figure 1, the circle on the left shows the visualization of fine-grained entity types that possess approximately 84% among all labels in the corpus, and the set on the right shows the detailed distributions of all fine-grained types. Each example includes 2.94 fine-grained entities on average; there are up to nine several fine-grained entity types per entity. The category to which the most entities belong is "AF_DOCUMENTS," which possesses 17.9%, and that on the second place is "PS_NAME," having 16.7%.

Label Description

- **12 general** types: PS, AF, AM, CV, DT, EV, PT, MT, TM, LC, LCG, LCP
- **94 fine-grained** types, which were mapped to the cultural heritage-specialized fine-grained entity labels, were inspired by prior works (Ling and Weld, 2012; Gillick et al., 2014; Choi et al., 2018).

3.3 Analysis on KoCHET

3.3.1 Diachronic and Linguistic Analysis

There are mainly two differences between the entities in the proposed corpus and those commonly used.

First, archaic expressions that are not used in modern times are frequently shown in our corpus. Specifically, such expressions continually appear when ancient documents or historical artifacts are quoted. Let us consider the phrase "한번사신레꼬-드는승질상밧고거-나믈느지는안슴니다" in sentence 1 in Table 4. Although it is written using syllables of modern Korean, the grammar and the vocabulary are fairly dissimilar from those of contemporary Korean, such as word spacing and syllabification, i.e., separation rule between the units of the word. When translating the sentence with quotation marks into modern Korean, it can be expressed as "한번 사신 레코드는 성질상 바꾸거나 무르지는 않습니다 (Once a record is purchased, it cannot be exchanged or refunded due to its characteristics)."

[9] www.cha.go.kr

Index	Example sentences
1	앞면 좌측 하단에 '한번사신레꼬-드는승질상밧고거-나믈느지는안슴니다' 문구가 있음. There is a phrase '한번사신레꼬-드는승질상밧고거-나믈느지는안슴니다'(archaic Korean) on the left corner of the front side.
2	1면에는 안창호씨(安昌浩氏)의 연설, 편집실 여언(餘言) 등의 기사가, · · ·, 인쇄됨. On the first page, articles such as Mr. Changho Ahn(安昌浩氏)(Chinese character)'s speech and editorial comments(餘言)(Chinese character), · · ·, were printed.
3	'戰爭の訓示', · · ·, 등의 기사와 일본 언어학자 가나자와 쇼자부로(金澤庄三郎, 1872∼1967)의 현대 국어 음운에 대한 연구물인「朝鮮語發音篇」의 일부를 게재함. '戰爭の訓示(Japanese)', · · ·, the articles and「朝鮮語發音篇」(Chinese character), the part of a study on the modern Korean phonology of Japanese linguist Kanazawa Shouzaburou(金澤庄三郎 (Chinese character), 1872∼1967) were published.

Table 4: Example sentences contained in our corpus. These examples include not only Korean but also Japanese and Chinese characters. Also, they contain archaic expressions that are not used in modern times. These characteristics make it more suitable for the learning of cultural heritage domain. Note that we omitted some of the words in the sentence for brevity.

Task		Train	Dev	Test
NER	# of examples	89,884	11,245	11,233
	# of entities	393,076	32,003	32,153
RE	# of examples	31,012	3,876	3,877
	# of relations	64,080	8,031	7,831
ET	# of examples	90,558	11,320	11,320
	# of mentions	266,209	33,226	33,395

Table 5: Statistics of **KoCHET** for each task.

Second, several entities contained in **KoCHET** written in Korean are followed by the descriptions written in either Chinese or Japanese characters. For example, as shown in sentence 2 in Table 4, the description with Chinese characters in parentheses follows the entity "안창호씨," and is usually written such as "안창호씨(安昌浩氏)." Further, Japanese characters are also present throughout the corpus, enhancing the polyglot property of the corpus, as shown in sentence 3. Therefore, to fully understand such expression types in our corpus, multilingual factors of language models should be considered; particularly in the case of token classification tasks, in which the meaning of each token directly affects the model performance.

3.3.2 Statistics

The overall statistics of **KoCHET** are showed in Table 5. For the NER corpus, 457,232 entities from 112,362 examples in total. For the RE corpus, 79,942 relations from 38,765 examples were annotated in total. For the ET corpus, 332,830 entity mentions from 113,198 examples were annotated in total. The annotated corpus was divided into three subsets for each task, i.e., a ratio of 8:1:1 for training, development, and testing, respectively. In this section, we describe our corpus statistically in the order of NER, RE, and ET.

First, as shown in Table 1, we used 12 entity types for our cultural heritage NER corpus. Due to the properties of the cultural heritage domain, the three primary entity types, i.e., artifacts (AF), person (PS), and term (TM), account for the majority of the total entity population. AF, PS, and TM entities possess approximately 36%, 20%, and 10%, respectively, which are used as crucial information in the cultural heritage domain. The AF type includes cultural assets and historical landmarks, the TM type includes patterns or traces engraved on certain cultural assets, and the PS type particularly includes not only general people but also particular types of persons such as mythical figures. On the other hand, the EV type occupies the most minor proportion, approximately 0.8%, because our corpus especially aims to concentrate on the cultural heritage.

Second, Table 2 demonstrates the distribution of 14 RE labels. In the case of "A depicts B" and "A documents B," cultural assets left in a specific form such as records, drawings, and photographs are included, whereas "A hasSection B" contains cultural heritage or historical landmarks located at a specific place. Among them, "A depicts B," "A documents B," and "A hasSection B" are the most relationship labels with approximately 22%, 16%, and 10% of the total, respectively. "A depicts B" and "A documents B" include cultural assets left in a specific form such as records, drawings, and

Model	NER		RE	ET
	Entity F1 (σ)	Character F1 (σ)	F1 (σ)	F1 (σ)
Multilingual fine-tuned Models				
Multilingual BERT	59.81 (0.09)	71.80 (0.12)	80.85 (0.39)	91.64 (0.10)
XLM-RoBERTa-base	**76.57** (0.13)	**82.69** (0.09)	80.29 (0.53)	91.13 (0.16)
Korean fine-tuned Models				
KLUE-BERT-base	39.31 (0.10)	55.63 (0.15)	**82.44** (0.18)	**93.08** (0.27)
KLUE-RoBERTa-base	38.92 (0.28)	55.47 (0.21)	82.42 (0.57)	92.80 (0.17)

Table 6: Experiments results on the NER, RE, and ET tasks. F1 score (%) is used for the evaluation metric with σ which shows the standard deviation of the score. We divide the baseline models into two parts: the Multilingual models and the Korean models, marking the highest performances with bold text.

photographs, whereas “A hasSection B" contains cultural heritage or historical landmarks located at a particular place. “A hasDestroyed B" has the smallest proportion with ten relations in total because, in actual history, significant events such as the collapse of a nation or the loss of cultural properties are not as diverse as the types of general cultural assets.

Finally, among the fine-grained entity types, the “AF_DOCUMENTS" type, such as historical documents, occupies the largest part with 17.9%, and “PS_NAME" including the names of historical figures, takes second place by occupying 11.5%. On the other hand, the entity types to which belong to the AM, PT, MT, and EV almost account for under 1.0%.

4 Experiment

The detailed experimental settings are in Appendix A.

Experimental results According to Table 6, two tendencies are observed. One is that in the NER task, the multilingual models, i.e., multilingual BERT and xlm-RoBERTa-base, showed better performance by more than 30% difference in both Entity F1 and Character F1 scores compared to the Korean models, i.e., KLUE-BERT-base and KLUE-RoBERTa-base. The other is that in the RE and ET tasks, the performances of the Korean models were at least 1.1% higher than those of the multilingual models.

Experimental Analysis As the token classification tasks are directly affected by segmentation (Kim et al., 2021; Park et al., 2021a), models with linguistic knowledge of Chinese and Japanese overperform in such tasks (Pires et al., 2019). In other words, the multilingual models are considered to segment better each token composed of various languages, especially in the NER corpus. In addition, in Table 7, the Korean models, i.e., KLUE-BERT-base and KLUE-RoBERTa-base show a significantly higher ratio of unknown tokens than the multilingual language models. It is attributed that the NER task requires more polyglot features of the model compared to the other tasks, i.e., RE and ET, which has the properties of sentence classification tasks. On the other hand, as the RE or ET task does not classify all tokens in a sentence, the correct answer can be satisfactorily inferred from only the given Korean words; thereby, the language models pre-trained in Korean show better performance in the two tasks compared to the multilingual model.

Model	UNK_dev (%)	UNK_test (%)
Multilingual BERT	0.8156%	0.7684%
XLM-RoBERTa-base	0.1952%	0.1810%
KLUE-BERT-base	5.8670%	5.9677%
KLUE-RoBERTa-base	5.8670%	5.9677%

Table 7: Unknown (UNK) token ratio (%) of each model for development and testing set in the corpus. Baseline models pre-trained in Korean show the same proportions because they use identical vocabulary and tokenizers.

5 Conclusion

In this paper, we introduced **KoCHET** - a Korean cultural heritage corpus for three typical entity-related tasks, i.e., NER, RE, and ET. Unlike the existing public Korean datasets with additional restrictions, **KoCHET** obviated the cumbersome prerequisite and can be freely modified and redistributed. Furthermore, we proved the applicability

of our entity-abundant corpus with the experiments employing the various pre-trained language models and provided practical insights regarding the statistical, diachronic, and linguistic analysis. Above all, the most significant contributing point is that the disclosure of our corpus is expected to serve as a cornerstone for the development of IE tasks for a traditional cultural heritage. We hope that the continuous effort to preserve cultural heritage with the effective management of digitized documents containing cultural artifacts is encouraged by this research.

Acknowledgements

This research is supported by Ministry of Culture, Sports and Tourism and Korea Creative Content Agency(Project Number: R2020040045), MSIT(Ministry of Science and ICT), Korea, under the ITRC(Information Technology Research Center) support program(IITP-2018-0-01405) supervised by the IITP(Institute for Information & Communications Technology Planning & Evaluation), and Institute of Information & communications Technology Planning & Evaluation(IITP) grant funded by the Korea government(MSIT) (No. 2020-0-00368, A Neural-Symbolic Model for Knowledge Acquisition and Inference Techniques).

References

Reem Alatrash, Dominik Schlechtweg, Jonas Kuhn, and Sabine Schulte im Walde. 2020. Ccoha: Clean corpus of historical american english. In *Proceedings of The 12th Language Resources and Evaluation Conference*, pages 6958–6966.

Thomas Cavalier-Smith. 2004. Only six kingdoms of life. *Proceedings of the Royal Society of London. Series B: Biological Sciences*, 271(1545):1251–1262.

Eunsol Choi, Omer Levy, Yejin Choi, and Luke Zettlemoyer. 2018. Ultra-fine entity typing. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 87–96.

Alexis Conneau, Kartikay Khandelwal, Naman Goyal, Vishrav Chaudhary, Guillaume Wenzek, Francisco Guzmán, Édouard Grave, Myle Ott, Luke Zettlemoyer, and Veselin Stoyanov. 2020. Unsupervised cross-lingual representation learning at scale. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 8440–8451.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, Minneapolis, Minnesota. Association for Computational Linguistics.

Dan Gillick, Nevena Lazic, Kuzman Ganchev, Jesse Kirchner, and David Huynh. 2014. Context-dependent fine-grained entity type tagging. *arXiv preprint arXiv:1412.1820*.

Helena Hubková, Pavel Král, and Eva Pettersson. 2020. Czech historical named entity corpus v 1.0. In *Proceedings of the 12th Language Resources and Evaluation Conference*, pages 4458–4465.

Gyeongmin Kim, Chanhee Lee, Jaechoon Jo, and Heuiseok Lim. 2020. Automatic extraction of named entities of cyber threats using a deep bi-lstm-crf network. *International Journal of Machine Learning and Cybernetics*, 11(10):2341–2355.

Gyeongmin Kim, Junyoung Son, Jinsung Kim, Hyunhee Lee, and Heuiseok Lim. 2021. Enhancing korean named entity recognition with linguistic tokenization strategies. *IEEE Access*, 9:151814–151823.

Hansaem Kim. 2006. Korean national corpus in the 21st century sejong project. In *Proceedings of the 13th NIJL International Symposium*, pages 49–54. National Institute for Japanese Language Tokyo.

Xiao Ling and Daniel S Weld. 2012. Fine-grained entity recognition. In *Twenty-Sixth AAAI Conference on Artificial Intelligence*.

Chanjun Park, Sugyeong Eo, Hyeonseok Moon, and Heuiseok Lim. 2021a. Should we find another model?: Improving neural machine translation performance with ONE-piece tokenization method without model modification. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies: Industry Papers*, pages 97–104, Online. Association for Computational Linguistics.

Sungjoon Park, Jihyung Moon, Sungdong Kim, Won Ik Cho, Jiyoon Han, Jangwon Park, Chisung Song, Junseong Kim, Yongsook Song, Taehwan Oh, et al. 2021b. Klue: Korean language understanding evaluation. *arXiv preprint arXiv:2105.09680*.

Fabian Pedregosa, Gaël Varoquaux, Alexandre Gramfort, Vincent Michel, Bertrand Thirion, Olivier Grisel, Mathieu Blondel, Peter Prettenhofer, Ron Weiss, Vincent Dubourg, Jake Vanderplas, Alexandre Passos, David Cournapeau, Matthieu Brucher, Matthieu Perrot, and Édouard Duchesnay. 2011. Scikit-learn: Machine learning in python. *Journal of Machine Learning Research*, 12(85):2825–2830.

Telmo Pires, Eva Schlinger, and Dan Garrette. 2019. How multilingual is multilingual bert? In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 4996–5001.

Ruize Wang, Duyu Tang, Nan Duan, Zhongyu Wei, Xuanjing Huang, Guihong Cao, Daxin Jiang, Ming Zhou, et al. 2020. K-adapter: Infusing knowledge into pre-trained models with adapters. *arXiv preprint arXiv:2002.01808*.

Sergey Zinin and Yang Xu. 2020. Corpus of Chinese dynastic histories: Gender analysis over two millennia. In *Proceedings of the 12th Language Resources and Evaluation Conference*, pages 785–793, Marseille, France. European Language Resources Association.

A Experimental Setup

As the baseline models, we employed two global language models: multilingual bidirectional encoder representations from transformers (BERT) (Devlin et al., 2019) and a cross-lingual language model XLM-RoBERTa-base (Conneau et al., 2020) containing the Korean language, and two KLUE language models: KLUE-BERT-base, KLUE-RoBERTa-base, which were recently published covering various Korean downstream tasks. In all the model experiments, the performance of each model was measured five times, and the average of each result was evaluated as the final result. Further, we set our environment for the experiment with four A6000 GPUs and 384 GB memory. The hyperparameters in the fine-tuning step were set as follows. The learning rate and weight decay were consistently set at 5e-5 and 0.01 across all three tasks. The number of training epochs was set to 10 in NER, RE and 3 in ET. The batch size in training and testing procedures was set to 128 in NER, RE and 256 in ET. In the case of max sequence length, the lengths of 256 and 128 were used for each task.

We evaluated our system by employing F1 score, which is standard metric for classification tasks. Specifically, the evaluation metrics for NER task were Entity F1 and Character F1 based on previous research (Park et al., 2021b). Entity F1 is a metric that is recognized as a correct answer only when all types included in an entity are matched accurately. Conversely, Character F1 is a metric that evaluates each type of syllable in a sentence individually. The evaluation metrics for the RE task were F1 score in the Scikit-learn library (Pedregosa et al., 2011). As for ET, we adopted the evaluation metrics of loose F1 score following the same evaluation criteria used in previous works (Ling and Weld, 2012; Wang et al., 2020).

MonoByte: A Pool of Monolingual Byte-level Language Models

Hugo Abonizio
FEEC, UNICAMP, Brazil
NeuralMind, Brazil
hugo.abonizio@gmail.com

Leandro Rodrigues de Souza
FEEC, UNICAMP, Brazil
l231250@g.unicamp.br

Roberto Lotufo
FEEC, UNICAMP, Brazil
NeuralMind, Brazil
lotufo@unicamp.br

Rodrigo Nogueira
FEEC, UNICAMP, Brazil
NeuralMind, Brazil
rfn@unicamp.br

Abstract

The zero-shot cross-lingual ability of models pretrained on multilingual and even monolingual corpora has spurred many hypotheses to explain this intriguing empirical result. However, due to the costs of pretraining, most research uses public models whose pretraining methodology, such as the choice of tokenization, corpus size, and computational budget, might differ drastically. When researchers pretrain their own models, they often do so under a constrained budget, and the resulting models might underperform significantly compared to SOTA models. These experimental differences led to various inconsistent conclusions about the nature of the cross-lingual ability of these models. To help further research on the topic, we released 10 monolingual byte-level models[1] rigorously pretrained under the same configuration with a large compute budget (equivalent to 420 days on a V100) and corpora that are 4 times larger than the original BERT's. Because they are tokenizer-free, the problem of unseen token embeddings is eliminated, thus allowing researchers to try a wider range of cross-lingual experiments in languages with different scripts. Additionally, we release two models pretrained on non-natural language texts that can be used in sanity-check experiments. Experiments on QA and NLI tasks show that our monolingual models achieve competitive performance to the multilingual one, and hence can be served to strengthen our understanding of cross-lingual transferability in language models.

1 Introduction

Shortly after the publication of BERT (Devlin et al., 2019), researchers showed that multilingual models pretrained on the masked language modeling objective can achieve remarkable zero-shot cross-lingual performance on various NLP tasks (i.e., a multilingual model finetuned on a high-resource language and directly evaluated in other languages) (Conneau et al., 2019; Hu et al., 2020).

These empirical results triggered a wave of research that aimed to explain this behavior. Pires et al. (2019) raised the "anchor tokens" hypothesis, i.e., that tokens shared between two languages act as a reference point so models can learn similar concepts. Artetxe et al. (2020) and Conneau et al. (2020) questioned these findings and pointed out that, even without a shared vocabulary, models achieve great cross-lingual transfer performance by leveraging shared parameters only.

Perhaps even more surprising is the performance of *monolingual* models in a cross-lingual setting. For instance, models pretrained and finetuned only on English can perform well in French tasks (Oladipo et al., 2022). de Souza et al. (2021) also show that monolingual models finetuned on a foreign language (i.e., a language that differs from pretraining) achieve comparable results to models finetuned on their language.

Recent work investigates properties of the pretraining corpora that contribute to the model's performance on natural language tasks. Models are pretrained on non-natural language corpora, such as code, music, proteins, and artificial languages and their performances are compared with natural language models (Papadimitriou and Jurafsky, 2020; Chiang and yi Lee, 2020; Lu et al., 2021; Ri and Tsuruoka, 2022). Evidence shows that pretraining on a corpus that contains artificial recursive or hierarchical structure between tokens results in a similar performance when compared to models

[1] https://huggingface.co/monobyte

Proceedings of the 29th International Conference on Computational Linguistics, pages 3506–3513
October 12–17, 2022.

pretrained on a natural language.

However, an outstanding problem is that these experiments are made either 1) using models pretrained by different research groups, whose pretraining configuration differs widely, or 2) using models pretrained under a compute budget or pretraining corpus that are orders of magnitude smaller than the ones used by SOTA models. For instance, Chiang and yi Lee (2020) pretrain their models on a dataset 200 times smaller than BERT's. This is problematic because certain skills are learned only when corpus size and training budget are large enough (Zhang et al., 2020).

These differences in the pretraining methodology and undertrained models make it difficult to draw a conclusion about the nature of the cross-lingual ability of such models. Additionally, since most models use subword tokenization, it is difficult to experiment with languages that do not have a significant subword vocabulary overlap. For example, if a model is pretrained and finetuned on English, it cannot be tested in Chinese because there is very little token overlap in their vocabularies, and hence, Chinese token embeddings are not learned. This is a problem even for languages with the same script. For example, many Spanish embeddings are not learned in an English-only pretraining, and hence it is difficult to tell whether a model's inability to learn cross-lingual representations is due only to tokenization issues (Rust et al., 2021).

Following Bommasani et al. (2021), who advocate for the release of pretraining models, corpus, and script as a way to strengthen the field, we release byte-level models pretrained on large corpora from the same domain and using exactly the same training setup and compute budget. Because they rely only on bytes to represent strings, they can be used to compare languages that use different scripts.

Each model takes approximately 210 hours of pretraining, resulting in more than three months of TPU computing. The models are available at https://huggingface.co/monobyte and the code used for finetuning can be found at https://github.com/lersouza/lang-agnostic.

2 Related work

In the last years, many monolingual versions of BERT (Devlin et al., 2019; Souza et al., 2020; Chan et al., 2020; Martin et al., 2020; Antoun et al., 2020; Lee et al., 2021; Nguyen and Tuan Nguyen, 2020; Cañete et al., 2020; Bhattacharjee et al., 2021) and T5 (Raffel et al., 2020; Carmo et al., 2020; Sarti and Nissim, 2022) have been released. The authors often claim that these versions outperform multilingual models in their languages. Rust et al. (2021) showed that the tokenizer plays a critical role in achieving those results. More recently, Xue et al. (2021a) released the ByT5 model, which would overcome this issue by leveraging a byte-level vocabulary. This model is pretrained on several languages and achieves great results when compared to mT5 (Xue et al., 2021b). However, no monolingual version of ByT5 has been released, which makes it difficult to conduct further investigation on the model's pretraining and cross-lingual performance that is not impacted by the tokenization in different languages.

Current research on the properties of pretraining often relies on monolingual models or models pretrained on artificial languages. Researchers often rely on models released by other groups (de Souza et al., 2021) or pretrain models in a very controlled setup (Artetxe et al., 2020; Papadimitriou and Jurafsky, 2020; Fujinuma et al., 2022). In both cases, it is not clear if the results suffer from these different methodologies or undertrained models. Blevins and Zettlemoyer (2022), for instance, suggests that language contamination (i.e., a monolingual corpus composed of sentences in more languages) is the reason for monolingual zero-shot effectiveness. The same happens with models pretrained on non-natural language corpora (Chiang and yi Lee, 2020; Lu et al., 2021; Ri and Tsuruoka, 2022), where the corpus size differs from the ones trained on natural language, which might affect the conclusions drawn from the experiments. We hope to bridge this gap by releasing the set of monolingual models presented in this paper.

3 Pretraining

To train our monolingual models, we used mC4 (Xue et al., 2021b), the same corpus used to train mT5 and ByT5, which comprises 101 natural languages extracted from the Common Crawl web scrape. For each model, we selected only the documents written in the specific language, which was originally identified using `cld3` by Xue et al. (2021b). The mC4 is a large corpus with different language distributions, with some being more represented than others. Thus, we trimmed each pre-

training corpus in approximately 65 billion UTF-8 bytes for all languages. The only exception was Bengali, which was trimmed on 32 billion bytes, corresponding to its total size.

The pretraining was conducted on a TPU VM v3-8 using Flax (Heek et al., 2020) library and the pretraining script for T5-like span-masked language modeling available on HuggingFace Transformers library.[2] We chose the smaller architecture of ByT5, with 300 million parameters, and used similar hyper-parameters as reported on ByT5 (Xue et al., 2021a) and mT5 (Xue et al., 2021b) papers. We did not experiment with larger models due to their computational cost. We set the sequence length to 1024 tokens (UTF-8 bytes) and train for 1 million steps with a batch of 2^{16} bytes, resulting in approximately 65 billion bytes. Since we could only fit a batch of 8 examples per device, we used gradient accumulation of 128 steps to achieve a larger batch size. The learning rate starts at 10^{-3} and decays linearly throughout the pretraining.

In comparison with the original ByT5, our models are pretrained on a 16 times smaller corpus. However, our pretraining corpus is 4 times larger than BERT's, which was trained on 3,300M words (Devlin et al., 2019) — i.e., approximately 16 billion characters, considering an average of 5 characters per word in English (Bochkarev et al., 2015). In comparison with GPT-2, our corpus is 1.6 times larger (Radford et al., 2019).

Compared with other monolingual models research, Rust et al. (2021) pretrained on a corpus of approximately 77GB, according to the reported number of steps and tokens per batch. Ri and Tsuruoka (2022) pretrained on a corpus of 1.2GB of artificially generated text. A more extensive comparison to other work is presented in Table 1 and demonstrates that our pretraining is comparable to SOTA monolingual models and much larger than corpora used in the majority of monolingual studies.

To sanity-check and validate the performance of our pretrained models, we compared their results with different baselines proposed in the literature. The first synthetic pretraining strategy, called nesting parentheses, was proposed by Papadimitriou and Jurafsky (2020), which recursively generates random symbols respecting a hierarchical structure. This was proposed to evaluate how this recursion

Model name	Language	Size
PTT5 (Carmo et al., 2020)	pt	15
BERT (Devlin et al., 2019)	en	16
BERTimbau (Souza et al., 2020)	pt	17
PhoBERT (Nguyen and Tuan Nguyen, 2020)	vi	20
AraBERT (Antoun et al., 2020)	ar	24
BanglaBERT (Bhattacharjee et al., 2021)	bn	35
KoreALBERT (Lee et al., 2021)	ko	43
North ByT5 (Kummervold, 2022)	no	45
CamemBERT (Martin et al., 2020)	fr	138
IT5 (Sarti and Nissim, 2022)	it	215
Chiang and yi Lee, 2020	en, artificial	0.08
Papadimitriou and Jurafsky, 2020	many, artif.	0.5
Deshpande et al., 2021	many	0.5
Ri and Tsuruoka, 2022	many, artif.	1.2
Rust et al., 2021	many	77
MonoByte (ours, each model)	many	65

Table 1: Comparison of pretraining corpus sizes in GB of monolingual models. Models which does not explicitly report the corpus size were estimated based on the reported amount of tokens.

structure of the pretraining corpus transfers into modeling real languages. Their work showed that the models pretrained on the synthetic hierarchical corpus were able to predict human language far better than other baselines (e.g., random, music, and code), with a perplexity comparable to real languages.

We used the code released by the authors, with the vocabulary being the 50,000 most frequent words based on the first one million documents of the Spanish mC4 corpus, and their sampling probability based on the Zipf distribution over the same Spanish corpus.[3] We generated examples until reaching the same size as our natural languages corpora sizes to make a fair comparison.

The second pretraining baseline was proposed by Krishna et al. (2021), dubbed nonsense, where the authors explored the knowledge transfer hypothesis of the downstream performance improvement of transfer learning. The authors pretrained on documents consisting of random n-grams and achieve similar performance on summarization over different datasets when compared to pretraining on real language. We generated examples using the original code[4] to match the same corpus size of our natural language models.

[2] https://github.com/huggingface/transformers/blob/main/examples/flax/language-modeling/run_t5_mlm_flax.py

[3] https://github.com/toizzy/tilt-transfer/

[4] https://github.com/acmi-lab/pretraining-with-nonsense/

	de	en	es	pt	ru	vi	zh	avg.
Nonsense	33.33	33.55	33.27	50.00	33.33	33.57	34.19	35.89
Hierarchical	66.01	69.32	68.04	65.65	62.12	64.49	62.83	65.49
Monolingual	70.85	76.76	75.07	83.40	67.39	70.37	69.91	73.39
Multilingual	76.02	80.55	78.39	67.73	73.58	73.01	73.39	74.67
Δ(*multi - mono*)	5.16	3.80	3.32	-15.67	6.19	2.64	3.48	1.28

Table 2: Results (accuracy) for Natural Language Inference in each language.

	ar	bn	en	ko	pt	ru	avg.
Nonsense	0.22	0.00	0.45	0.00	0.26	0.27	0.20
Hierarchical	67.23	0.38	28.03	9.99	3.63	32.85	23.68
Monolingual	78.17	53.89	60.15	51.17	64.88	58.26	61.09
Multilingual	81.81	70.28	68.51	54.70	42.37	72.97	65.11
Δ(*multi - mono*)	3.65	16.39	8.36	3.54	-22.51	14.71	4.02

Table 3: Results (F1 Score) for Question Answering in each language.

4 Experiments

Our models are evaluated on two downstream tasks: Natural Language Inference (NLI) and Question Answering (QA). We use a similar setting as the *In-language model*, described by Hu et al. (2020), where the model is finetuned and evaluated in the subset of the task that corresponds to its pretraining language. The following subsections provide details about the datasets used and the results. The monolingual models are compared to the ByT5 Small checkpoint. The latter is expected to perform better since it is pretrained on a much larger set of a more diverse corpus, as investigated by Fujinuma et al. (2022). For additional information on the finetuning procedure, please refer to Appendix A.

Natural Language Inference. The XNLI[5] dataset (Conneau et al., 2018) is used for finetuning and evaluating our German, English, Spanish, Russian, Vietnamese, and Chinese models. For the Portuguese model, we select the Recognizing Textual Entailment (RTE) task of the ASSIN2 dataset (Real et al., 2020). Performance is measured by using the accuracy metric.

Results are reported in Table 2. Compared to the multilingual ByT5, our models achieve competitive performance. The difference in accuracy is about 1.28 on average. The results in Russian represent the largest gap (6.19 points), while those in Vietnamese represent the smallest (2.64 points). Nonsense achieves a near random performance, while Hierarchical approaches the models pretrained on natural language, as also evidenced by Papadimitriou and Jurafsky (2020) and Chiang and yi Lee (2020).

Question Answering. The gold passage version of the TydiQA (Clark et al., 2020) dataset (**TydiQA-GoldP**) is selected for finetuning and evaluating the Arabic, Bengali, English, Korean and Russian models. For finetuning the Portuguese model, we use the FaQuAD (Sayama et al., 2019). Performance is measured by the F1 Score metric as described by Rajpurkar et al. (2016).

Results are reported in Table 3. For Portuguese, our model outperforms the Multilingual checkpoint by 22.51 points, while, for Bengali, it stays 16.39 points behind. Our monolingual models stay, on average, 4 points behind the multilingual model released by Google. Looking at the difference between monolingual and multilingual models for every language, we see a higher variation. Nonsense and Hierarchical results are very distant. Question answering is a more difficult task (Vania et al., 2021) and a good benchmark for model quality. We hypothesize that this difference in task complexity requires more from the model. Pretraining with token structure emphasis only may not provide the required knowledge for good performance.

5 Conclusion

In this work, we introduced a pool of 10 tokenizer-free language models pretrained on large monolingual corpora. We demonstrated that our monolin-

[5]We use the machine-translated version of XNLI for training in different languages. Available at https://dl.fbaipublicfiles.com/XNLI/XNLI-MT-1.0.zip

gual models can achieve competitive results to the multilingual ByT5, although having a smaller and less diverse pretraining — single language, compared to all languages concatenated.

The main goal of our work was to release a set of monolingual models that carefully follow the same pretraining methodology to support the research of monolingual language models, pretraining properties, and cross-linguality. We hope that by releasing our models, we bridge this gap of having a more controlled and reproducible setup for rigorous experiments.

Acknowledgments

This research was partially funded by grants 2020/09753-5 and 2022/01640-2 from Fundação de Amparo à Pesquisa do Estado de São Paulo (FAPESP). We also would like to thank Google Cloud for credits to support this work.

References

Wissam Antoun, Fady Baly, and Hazem Hajj. 2020. AraBERT: Transformer-based model for Arabic language understanding. In *Proceedings of the 4th Workshop on Open-Source Arabic Corpora and Processing Tools, with a Shared Task on Offensive Language Detection*, pages 9–15, Marseille, France. European Language Resource Association.

Mikel Artetxe, Sebastian Ruder, and Dani Yogatama. 2020. On the cross-lingual transferability of monolingual representations. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 4623–4637, Online. Association for Computational Linguistics.

Abhik Bhattacharjee, Tahmid Hasan, Wasi Uddin Ahmad, Kazi Samin, Md Saiful Islam, Anindya Iqbal, M. Sohel Rahman, and Rifat Shahriyar. 2021. Banglabert: Language model pretraining and benchmarks for low-resource language understanding evaluation in bangla.

Terra Blevins and Luke Zettlemoyer. 2022. Language contamination explains the cross-lingual capabilities of english pretrained models.

Vladimir V Bochkarev, Anna V Shevlyakova, and Valery D Solovyev. 2015. The average word length dynamics as an indicator of cultural changes in society. *Social Evolution and History*, 14(2):153–175.

Rishi Bommasani, Drew A. Hudson, Ehsan Adeli, Russ Altman, Simran Arora, Sydney von Arx, Michael S. Bernstein, Jeannette Bohg, Antoine Bosselut, Emma Brunskill, Erik Brynjolfsson, Shyamal Buch, Dallas Card, Rodrigo Castellon, Niladri S. Chatterji, Annie S. Chen, Kathleen Creel, Jared Quincy Davis, Dorottya Demszky, Chris Donahue, Moussa Doumbouya, Esin Durmus, Stefano Ermon, John Etchemendy, Kawin Ethayarajh, Li Fei-Fei, Chelsea Finn, Trevor Gale, Lauren Gillespie, Karan Goel, Noah D. Goodman, Shelby Grossman, Neel Guha, Tatsunori Hashimoto, Peter Henderson, John Hewitt, Daniel E. Ho, Jenny Hong, Kyle Hsu, Jing Huang, Thomas Icard, Saahil Jain, Dan Jurafsky, Pratyusha Kalluri, Siddharth Karamcheti, Geoff Keeling, Fereshte Khani, Omar Khattab, Pang Wei Koh, Mark S. Krass, Ranjay Krishna, Rohith Kuditipudi, and et al. 2021. On the opportunities and risks of foundation models. *CoRR*, abs/2108.07258.

Diedre Carmo, Marcos Piau, Israel Campiotti, Rodrigo Nogueira, and Roberto Lotufo. 2020. Ptt5: Pretraining and validating the t5 model on brazilian portuguese data.

José Cañete, Gabriel Chaperon, Rodrigo Fuentes, Jou-Hui Ho, Hojin Kang, and Jorge Pérez. 2020. Spanish pre-trained bert model and evaluation data. In *PML4DC at ICLR 2020*.

Branden Chan, Stefan Schweter, and Timo Möller. 2020. German's next language model. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 6788–6796, Barcelona, Spain (Online). International Committee on Computational Linguistics.

Cheng-Han Chiang and Hung yi Lee. 2020. Pre-training a language model without human language.

Jonathan H. Clark, Eunsol Choi, Michael Collins, Dan Garrette, Tom Kwiatkowski, Vitaly Nikolaev, and Jennimaria Palomaki. 2020. Tydi qa: A benchmark for information-seeking question answering in typologically diverse languages. *Transactions of the Association for Computational Linguistics*.

Alexis Conneau, Kartikay Khandelwal, Naman Goyal, Vishrav Chaudhary, Guillaume Wenzek, Francisco Guzmán, Edouard Grave, Myle Ott, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Unsupervised cross-lingual representation learning at scale. *CoRR*, abs/1911.02116.

Alexis Conneau, Ruty Rinott, Guillaume Lample, Adina Williams, Samuel R. Bowman, Holger Schwenk, and Veselin Stoyanov. 2018. "XNLI: Evaluating Cross-lingual Sentence Representations". In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics.

Alexis Conneau, Shijie Wu, Haoran Li, Luke Zettlemoyer, and Veselin Stoyanov. 2020. Emerging cross-lingual structure in pretrained language models. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 6022–6034, Online. Association for Computational Linguistics.

Leandro Rodrigues de Souza, Rodrigo Nogueira, and Roberto Lotufo. 2021. On the ability of monolingual models to learn language-agnostic representations.

Ameet Deshpande, Partha Talukdar, and Karthik Narasimhan. 2021. When is bert multilingual? isolating crucial ingredients for cross-lingual transfer. *arXiv preprint arXiv:2110.14782*.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, Minneapolis, Minnesota. Association for Computational Linguistics.

Yoshinari Fujinuma, Jordan Boyd-Graber, and Katharina Kann. 2022. Match the script, adapt if multilingual: Analyzing the effect of multilingual pretraining on cross-lingual transferability.

Jonathan Heek, Anselm Levskaya, Avital Oliver, Marvin Ritter, Bertrand Rondepierre, Andreas Steiner, and Marc van Zee. 2020. Flax: A neural network library and ecosystem for JAX.

Junjie Hu, Sebastian Ruder, Aditya Siddhant, Graham Neubig, Orhan Firat, and Melvin Johnson. 2020. XTREME: A Massively Multilingual Multitask Benchmark for Evaluating Cross-lingual Generalisation. In *International Conference on Machine Learning*, pages 4411–4421. PMLR.

Kundan Krishna, Jeffrey Bigham, and Zachary C. Lipton. 2021. Does pretraining for summarization require knowledge transfer? In *Findings of the Association for Computational Linguistics: EMNLP 2021*, pages 3178–3189, Punta Cana, Dominican Republic. Association for Computational Linguistics.

Per E Kummervold. 2022. North T5. `https://github.com/peregilk/north-t5`.

Hyunjae Lee, Jaewoong Yoon, Bonggyu Hwang, Seongho Joe, Seungjai Min, and Youngjune Gwon. 2021. Korealbert: Pretraining a lite bert model for korean language understanding.

Kevin Lu, Pieter Abbeel, Aditya Grover, Igor Mordatch, and Google Brain. 2021. Pretrained transformers as universal computation engines.

Louis Martin, Benjamin Muller, Pedro Javier Ortiz Suárez, Yoann Dupont, Laurent Romary, Éric de la Clergerie, Djamé Seddah, and Benoît Sagot. 2020. CamemBERT: a tasty French language model. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7203–7219, Online. Association for Computational Linguistics.

Dat Quoc Nguyen and Anh Tuan Nguyen. 2020. PhoBERT: Pre-trained language models for Vietnamese. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 1037–1042, Online. Association for Computational Linguistics.

Akintunde Oladipo, Odunayo Ogundepo, Kelechi Ogueji, and Jimmy Lin. 2022. An exploration of vocabulary size and transfer effects in multilingual language models for african languages. In *3rd Workshop on African Natural Language Processing*.

Isabel Papadimitriou and Dan Jurafsky. 2020. Learning Music Helps You Read: Using transfer to study linguistic structure in language models. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 6829–6839, Online. Association for Computational Linguistics.

Telmo Pires, Eva Schlinger, and Dan Garrette. 2019. How multilingual is Multilingual BERT? *ACL 2019 - 57th Annual Meeting of the Association for Computational Linguistics, Proceedings of the Conference*, pages 4996–5001.

Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, Ilya Sutskever, et al. 2019. Language models are unsupervised multitask learners. *OpenAI blog*, 1(8):9.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J. Liu. 2020. Exploring the limits of transfer learning with a unified text-to-text transformer. *Journal of Machine Learning Research*, 21(140):1–67.

Pranav Rajpurkar, Jian Zhang, Konstantin Lopyrev, and Percy Liang. 2016. SQuAD: 100,000+ Questions for Machine Comprehension of Text.

Livy Real, Erick Fonseca, and Hugo Gonçalo Oliveira. 2020. The assin 2 shared task: A quick overview. In *Computational Processing of the Portuguese Language*, pages 406–412, Cham. Springer International Publishing.

Ryokan Ri and Yoshimasa Tsuruoka. 2022. Pretraining with artificial language: Studying transferable knowledge in language models.

Phillip Rust, Jonas Pfeiffer, Ivan Vulić, Sebastian Ruder, and Iryna Gurevych. 2021. How good is your tokenizer? on the monolingual performance of multilingual language models. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 3118–3135, Online. Association for Computational Linguistics.

Gabriele Sarti and Malvina Nissim. 2022. It5: Large-scale text-to-text pretraining for italian language understanding and generation.

H. F. Sayama, A. V. Araujo, and E. R. Fernandes. 2019. FaQuAD: Reading Comprehension Dataset in the Domain of Brazilian Higher Education. In *2019 8th Brazilian Conference on Intelligent Systems (BRACIS)*, pages 443–448.

Noam Shazeer and Mitchell Stern. 2018. Adafactor: Adaptive learning rates with sublinear memory cost.

Fábio Souza, Rodrigo Nogueira, and Roberto Lotufo. 2020. Bertimbau: Pretrained bert models for brazilian portuguese. *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, 12319 LNAI:403–417.

Clara Vania, Phu Mon Htut, William Huang, Dhara Mungra, Richard Yuanzhe Pang, Jason Phang, Haokun Liu, Kyunghyun Cho, and Samuel R. Bowman. 2021. Comparing test sets with item response theory. *ACL-IJCNLP 2021 - 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing, Proceedings of the Conference*, pages 1141–1158.

Linting Xue, Aditya Barua, Noah Constant, Rami Al-Rfou, Sharan Narang, Mihir Kale, Adam Roberts, and Colin Raffel. 2021a. Byt5: Towards a token-free future with pre-trained byte-to-byte models.

Linting Xue, Noah Constant, Adam Roberts, Mihir Kale, Rami Al-Rfou, Aditya Siddhant, Aditya Barua, and Colin Raffel. 2021b. mT5: A massively multilingual pre-trained text-to-text transformer. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 483–498, Online. Association for Computational Linguistics.

Yian Zhang, Alex Warstadt, Haau-Sing Li, and Samuel R. Bowman. 2020. When do you need billions of words of pretraining data?

A Finetuning Details

The experiments were carried out in a single NVIDIA A100 80GB GPU. We use the Adafactor optimizer (Shazeer and Stern, 2018) with a constant learning rate of 10^{-4}. The models were finetuned using three different seeds, and we report an average of the results. We have chosen hyperparameters based on preliminary experiments with the ByT5 Small checkpoint[6] on both XNLI and TyDiQA datasets. We do not perform any exhaustive hyperparameter search, and use the same settings (per-task) for all models.

Natural Language Inference. For all NLI experiments, we use a batch size of 16, accumulating gradients for 4 steps. The maximum input length was 1024, trimmed by batch. We train the model to output the class identifier (a number). We train for 10 epochs and evaluate every 0.2 epoch. We also perform early stopping with patience of 5 evaluations and select the best model on the validation set of each task. The results are reported against the test set of each dataset.

Question Answering. For QA, the selected batch size is 6, accumulating gradients for 4 steps. The maximum input length was 2048 (question and context concatenated), trimmed by batch. We train the model to output an answer to the question with a maximum length of 768 bytes. We train for 10 epochs and evaluate at the end of each epoch. We also perform early stopping with patience of 3 and select the best model on the validation set of each task.

[6] Available at: https://huggingface.co/google/byt5-small

Wizard of Tasks: A Novel Conversational Dataset for Solving Real-World Tasks in Conversational Settings

Jason Ingyu Choi, Saar Kuzi, Nikhita Vedula, Jie Zhao, Giuseppe Castellucci, Marcus Collins, Shervin Malmasi, Oleg Rokhlenko and Eugene Agichtein
Amazon
{chojson,skuzi,veduln,zhaozjie,giusecas}@amazon.com
{collmr,malmasi,olegro,eugeneag}@amazon.com

Abstract

Conversational Task Assistants (CTAs) are conversational agents whose goal is to help humans perform real-world tasks. CTAs can help in exploring available tasks, answering task-specific questions and guiding users through step-by-step instructions. In this work, we present Wizard of Tasks, the first corpus of such conversations in two domains: Cooking and Home Improvement. We crowd-sourced a total of 549 conversations (18,077 utterances) with an asynchronous Wizard-of-Oz setup, relying on recipes from WholeFoods Market for the cooking domain, and WikiHow articles for the home improvement domain. We present a detailed data analysis and show that the collected data can be a valuable and challenging resource for CTAs in two tasks: Intent Classification (IC) and Abstractive Question Answering (AQA). While on IC we acquired a high performing model ($\geq$85% F1), on AQA the performance is far from being satisfactory ($\sim$27% BertScore-F1), suggesting that more work is needed to solve the task of low-resource AQA.

1 Introduction

In recent years, the way to access web information has changed from using keyword- and semantics-based search (Manning et al., 2008), to Question Answering (QA) (Chen et al., 2017) and Conversational Agents (CAs) (Radlinski and Craswell, 2017). CAs have evolved to support different types of interaction and information: there are CAs for chatting (Zhou et al., 2020) and CAs that let users interact with existing information systems to accomplish specific tasks, e.g., booking a restaurant, as in task-oriented dialogues (Bobrow et al., 1977; Wen et al., 2017).

A specific type of information humans are looking for is how to perform tasks, e.g., cooking a dish or fixing a household problem. The Web contains articles accompanied by images or videos with step-by-step instructions to perform variegate tasks. Existing CAs can be used mainly to browse tasks or to answer specific questions. However, they fail in providing a comprehensive natural conversation that includes search, context-aware QA, step-by-step instructions, and multi-modal information sharing.

The Alexa Prize TaskBot[1] Challenge (Gottardi et al., 2022) is a research challenge sponsored by Amazon to foster research on CTAs to assist humans in executing real-world tasks. The targeted tasks require multiple steps and decisions, including multi-modal (voice and visual) user experiences. The challenge includes two domains: Cooking, i.e., guiding people in preparing recipes; and Home Improvement, i.e., guiding people through common household do-it-yourself tasks such as painting a wall or pruning trees. As shown in Figure 1, CTAs should support QA capabilities on Web sources as well as selected recipes or articles, dialog management to support step-by-step instruction navigation, and multi-modal interaction.

In this paper, we present Wizard of Tasks[2] (WoT), the first dataset for CTAs. We collected a total of 549 conversations with $\sim$ 18K utterances in two domains with a Wizard Of Oz (WOZ) crowd-sourcing setting, where one worker is willing to execute a task, while another worker has the relevant knowledge to perform it and guides the first towards its execution. We adopted an asynchronous strategy to collect conversations, so that neither worker needs to wait for the other to respond and can multi-task better, enabling faster data collection. We present a detailed analysis of the dataset as well as experiments in two tasks: Intent Classification (IC) and Abstractive Question Answering (AQA). We show that a transformer-

[1] https://www.amazon.science/alexa-prize/taskbot-challenge
[2] https://registry.opendata.aws/wizard-of-tasks/

Proceedings of the 29th International Conference on Computational Linguistics, pages 3514–3529
October 12–17, 2022.

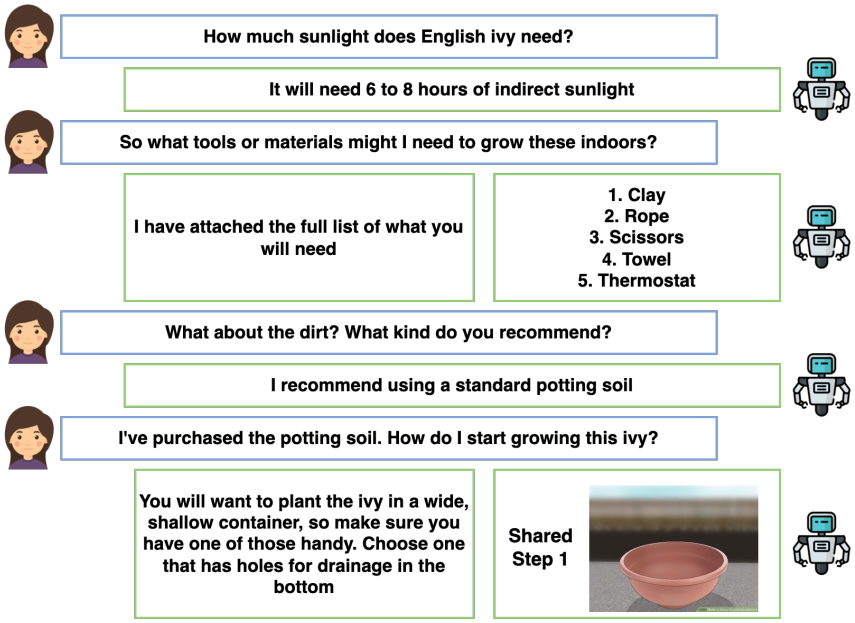

Figure 1: First few turns of one `Wizard of Tasks` conversation from the Home Improvement (DIY) domain.

based IC model can achieve ≥85% F1. In contrast, the performance on AQA is still far from satisfactory (∼27% BertScore-F1), suggesting the need for better QA models for low-resource settings.

In the rest of the paper, Section 2 discusses related work. Section 3 presents our data collection strategy. In sections 4 and 5 we discuss the data analysis and experimental evaluation. Finally, in Section 6, we state our conclusions.

2 Related Work

There is a large body of work focusing on the development of training corpora for conversational agents. In the following section, we summarize the work done to collect conversations for task-oriented and open-domain CAs.

2.1 Task-oriented Agents

The goal of task-oriented agents is to assist users in completing a task that is grounded in a knowledge base. For example, an agent can assist users in making a restaurant reservation by eliciting their preferences, building a database query and sharing the available options. While some of the previous work has focused on single-domain data sets (Moon et al., 2020; Wen et al., 2017), most of it focused on multi-domain data (Budzianowski et al., 2018; El Asri et al., 2017; Shah et al., 2018).

One of the approaches for collecting task-oriented conversations is to use computer simulations and templates to generate synthetic data sets (Shah et al., 2018; Rastogi et al., 2020; Zhao and Eskenazi, 2018), possibly also rephrasing using crowd-sourcing (Rastogi et al., 2020; Shah et al., 2018). A second approach is to let users interact with existing dialog systems (Williams et al., 2016) to improve their performance.

Perhaps the most well-known approach is to crowdsource conversations using a pool of public workers. Specifically, the WOZ paradigm is often used where one human is playing the role of a conversational agent and the other one is playing that of a user (Wen et al., 2017; Budzianowski et al., 2018; Zang et al., 2020; Eric et al., 2020; Moon et al., 2020; El Asri et al., 2017). The advantage is that the resulting data consists of natural conversations compared to using computer simulations. In some studies (Wen et al., 2017; Ikeda and Hoashi, 2018), the data was collected in an asynchronous manner by assigning each conversation turn to an available worker who writes an utterance based on all previous turns.

2.2 Open-domain Agents

Open-domain dialog agents can converse with users about topics without a clear goal. The conversations are usually grounded in some knowledge source, e.g., a Wikipedia page. The main approach for collecting data for open-domain dialog systems is to collect conversations between humans using a crowd-sourcing platform and the WOZ setting. In some studies, only one worker has access to the

knowledge source (Dinan et al., 2018). In other works, both sides have access to some knowledge source (Gopalakrishnan et al., 2019; Zhou et al., 2018; Moghe et al., 2018; Zhang et al., 2018) to simulate a scenario where two humans share some amount of background knowledge.

3 Crowd-sourcing a CTA Dataset

We designed a crowd-sourcing task to create a dataset suitable for a CTA to assist users in completing complex tasks requiring multiple steps and reasoning. We collected conversations for two target domains[3]: i) Cooking, i.e., assisting users in performing recipes and ii) Home Improvement (DIY), i.e., assisting users in performing tasks to improve their home.

3.1 Worker Roles and Expectations

Our data collection adopts a WOZ paradigm, where one worker (i.e., the student) communicates with another worker (i.e., the teacher) about tasks and how to perform them. Each worker is assigned only one of the two roles for the duration of the study to avoid potential quality issues.

3.1.1 Teacher Role

The teacher is defined as a knowledgeable expert who instructs the student to complete an assigned task, while keeping the conversation engaging and natural. The teacher is given a set of informative documents about the task to help the student.

A conversation starts with a student asking about the task. The teacher is expected to understand the document, find relevant instructions and respond to the student. The teacher can also access external resources to search for the needed information with their preferred search engine. In this case, we ask the teacher to provide the URL of any reference used to produce their response. One interesting feature in our dataset is the adoption of multi-modality to share information between the agent and the user. Creating such a multi-modal experience is not trivial, and requires more understanding on how people behave in such a setting. Thus, the teacher can share various types of content with the student regarding the task to enrich their response (e.g., step images, step text, ingredients, and tools).

Teacher Task. The teachers are required to answer four domain-independent questions for each turn before submitting their responses:

- Is the last student message relevant and coherent to the conversation history?
- Is the last student message useful for proceeding to the next steps?
- What is the action of your message?
- Write your response to the last student.

The first two questions provide relevance and usefulness labels for the previous student turn. The answer to these questions is a binary label (yes/no). The last two questions ask for the action of the teachers and their response.[4] The teachers can choose among the following actions: i) return a list of ingredients/tools; ii) return the next step; iii) answer a question only using the current task document; iv) answer a question using external knowledge (e.g., via common sense or search); v) ask a question to the student.[5]

3.1.2 Student Role

The student is defined as a curious learner who is willing to complete a task, while keeping the conversation engaging and natural. Initially, the student is given only a task title, and is expected to start the conversation by asking about the task, the required ingredients or the next steps. As the task progresses, the student is responsible for following the teacher instructions and moving toward task completion. The student is encouraged to ask at least one open-ended question about ingredients or general questions about the current step before moving to the next step. This helps to minimize the chance that the student simply requests the next step in the task without meaningful interactions.

Student Task. The students must answer six questions to submit their responses as follows:

- Is the last teacher's message relevant and coherent to the chat history?
- Is the last message of the teacher useful for proceeding to the next steps?
- Which shared content, if any, can cause potential harm to people or property damage?

[3] Whole Foods Market (https://www.wholefoodsmarket.com/recipes) for cooking tasks and Wikihow (https://www.wikihow.com/Main-Page) for DIY tasks.

[4] We disabled the copy-paste function to force the teachers to write their final response.

[5] This is to encourage teachers to ask questions to produce more natural conversations.

- What would you do in real-life if you were doing the task?
- What is the intent of your response?
- What is your response to the previous teacher?

While the first two questions are similar to the ones asked to the teacher, the third is added to label potentially dangerous content or activities provided by the teacher. The fourth question is meant to create a more realistic annotation experience. Since we do not expect workers to really do the tasks, by asking what the worker would do in real-life (e.g., "walk to the refrigerator, grab a pear and start cutting it"), we hope they can better focus on the tasks and generate more realistic sentences.

The last two questions are used to collect the intent and the message of the student. The options are: i) request ingredients/tools; ii) request next step; iii) ask a question about ingredients/tools; iv) ask a question about a step; v) stop. More details about crowdsourcing tasks are available in Appendix A.

3.2 Crowdsourced Data Collection

We used Amazon Mechanical Turk[6] to collect `Wizard of Tasks`. We paid \$0.20 for each completed task.[7] We also used an on-boarding task to filter out low-quality workers. The conversational data was collected in two batches to enable a data quality check during the process. The first batch of Cooking data (66%) was collected between Oct. 5 and Oct. 8 (2021) and the second batch between Jan. 1 and Jan. 4 (2022). The first batch of the DIY data (75%) was between Oct. 14 and Oct. 30 (2021) and the second batch between Jan. 4 and Jan. 12 (2022).

3.2.1 Asynchronous Strategy for Conversational Data Collection

Unlike previous data collection approaches for conversations (Budzianowski et al., 2018; Zang et al., 2020; Eric et al., 2020; Moon et al., 2020), we adopted an asynchronous strategy (Ikeda and Hoashi, 2018) to collect `Wizard of Tasks`. This approach has several advantages over synchronous conversation. The main advantage is that two workers are not required to be online at the same time. This allows them to work on multiple assignments simultaneously, a common practice among crowd-sourcing workers. Moreover, decoupling the workers will free them from waiting for the other party to reply, making the collection more time efficient, and thus, reducing costs for each task. As a side effect, more than two workers participate in a single conversation. This may bring more diversity into the language and writing styles as against using the same two workers.

In practice, for each turn i of a conversation, the worker (either a student or a teacher) can see some information, including the task title, the history of the conversation up to turn $i-1$, the content shared by the teachers, and, only for the teacher, the document content. The worker is expected to use the included information to understand the context and decide how to reply (i.e., to provide the ith turn). After answering the required questions, the worker can submit a task. This triggers the automatic creation of a new task for the next turn $i+1$, which can be accessed by the next worker, who may be different from the previous ones.

3.2.2 Automated Quality Assurance

One disadvantage of the asynchronous strategy is that workers can abuse it.[8] We developed a heuristic to block malicious workers. When workers submitted a task, we retrieved their k=5 most recent submissions and evaluated the average relevancy and usefulness. If either value fell below a threshold (p=0.5), those workers were temporarily blocked. Each day, we manually analyzed sample submissions from temporarily blocked workers to decide whether each worker should be permanently blacklisted or unblocked. Finally, if both the average relevancy and usefulness were greater than p=0.9 in the latest k=5 submissions, we paid a bonus of \$0.08.

4 Wizard of Tasks Analysis

In this section, we present the analysis of the `Wizard of Tasks` data. Dataset statistics are presented in section 4.1, followed by a linguistic analysis in section 4.2.[9]

4.1 Dataset Statistics

The dataset consists of 272 (277) conversations in the Cooking (DIY) domain comprising 7,908

[6] https://www.mturk.com/

[7] To estimate the task price, we recruited five domain experts to work on demo tasks for 60 minutes and we computed the single task price according to a pay of \$12.5 per hour.

[8] In an early phase, we observed some workers inputting random and repetitive responses.

[9] We report additional statistics in Appendix D and conversations examples in Appendix C.

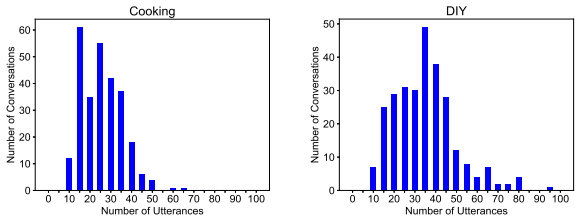

Figure 2: Conversation Length Histogram. Each point x in the x-axis corresponds to the interval $[x, x_r)$ where x_r is the closest point to the right of x.

$(10, 169)$ utterances. 238 (159) workers participated in the Cooking (DIY) experiments with an average number of utterances per worker of 33.2 (63.9). The average number of unique tasks per worker is 20.1 (35.1) for Cooking (DIY). The average number of utterances per conversation is 29.1 for Cooking and 36.7 for DIY. Figure 2 shows histograms of the conversation length. We speculate that the difference between the two domains is that DIY tasks are generally more complex and require more information for their completion.

	Cooking		DIY	
Role	Relevance	Usefulness	Relevance	Usefulness
Student	97.2%	90.2%	97.1%	90.9%
Teacher	96.5%	94.4%	97.7%	95.7%

Table 1: The percentage of student and teacher utterances that were marked as relevant or useful by crowdsourcing workers.

Utterance Quality. In Table 1, we report the percentage of utterances that were assessed positively by other workers with respect to relevance and usefulness. The results demonstrate the high quality of the collected utterances with averaged relevancy higher than 95% in both domains. Please note that teacher utterances achieved higher usefulness than student utterances. This can be attributed to the asymmetry of roles: while the student mostly asks questions, the teacher has to answer them which is more informative and thus useful.

External Resources and Shared Information. In 6.4% (3.5%) of teacher utterances for the Cooking (DIY) domain, the workers reported that they used external URLs. In 56.4% (63.4%) of teacher utterances in Cooking (DIY), the workers shared information from the recipe/article itself. A possible explanation of the difference is that recipes are generally shorter, thus teachers were willing to share more in the DIY domain.

Student Intents	Cooking	DIY
Request Step (Previous or Next)	45.8%	51.1%
Steps Question	32.7%	30.8%
Request Ingredients/Tools	13.8%	11.0%
Stop	6.8%	5.9%
Chit-chat	0.4%	1.1%
Other	0.5%	0.2%

Teacher Actions	Cooking	DIY
Return Step (Previous or Next)	49.1%	54.1%
Internal Fact Answer	17.0%	19.1%
External Fact Answer	23.7%	18.7%
Return Ingredients/Tools	6.7%	5.2%
Chit-chat	1.5%	0.7%
Other	0.2%	0.3%
Ask Question	1.8%	1.9%

Table 2: The percentage of student and teacher utterances in each intent/action type.

Intent and Action Types. In Table 2, we report the percentage of utterances for each intent/action. The two most common student intents (>78%) are requesting a step or asking a question about ongoing steps. The two common teacher actions (>66%) are either returning a step or answering a question about the given steps/tasks. The large portion of question/answer interactions attests to the complexity of the underlying tasks.

4.2 Linguistic Analysis

Utterance Length. The average number of tokens in utterances is 18.5 (21.7) and 14.2 (15.6) for teachers and students, respectively, in the Cooking (DIY) domain.[10] Figure 3 shows the full distribution of utterance length across conversations. The histogram shows different distributions for students and teachers, which we attribute again to the asymmetry of the roles.

The average number of sentences per utterance is slightly higher for teachers, i.e., 1.4 (1.4) and 1.5 (1.6) average sentences per utterance for students and teachers, respectively, in Cooking (DIY). In general, the low values demonstrate the conversational nature of the dataset. Even though teacher utterances have more words, the workers used a small number of sentences to construct them.

Linguistic Patterns Analysis. To investigate the linguistic patterns in the collected data, we ran a dependency parser on the sentences. To construct a pattern from a sentence, we first identify the child tokens of the sentence root. Then, we concatenate their dependency to generate a template.[11] Examples of the three most frequent patterns for students

[10] To perform the linguistic analysis, we used the spaCy library (https://spacy.io/).

[11] The dependencies are concatenated based on the order of the corresponding tokens in the sentence.

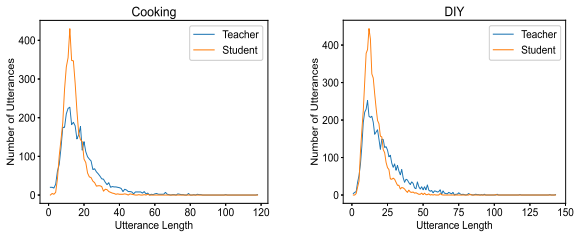

Figure 3: Utterance Length Distribution. The number of tokens in utterances of students and teachers.

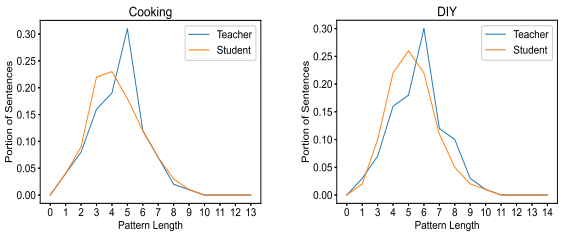

Figure 4: Linguistic Pattern Length Distribution. The portion of sentences with a specific pattern length.

and teachers are presented in Table 3 (for the DIY domain, refer to Table 11 in the Appendix).

In Table 4, we report statistics of the patterns. The results demonstrate diverse linguistic patterns in the data. In both domains, the average number of sentences per pattern (i.e., the average number of sentences expressing a pattern) is around 3. Furthermore, we can observe a slightly lower average number for teachers compared to students. A possible explanation is that teachers have access to the article/recipe from different sources which may have increased the linguistic variance of their responses. The table also shows that the average length of patterns (i.e., the average number of children of a root node) is around 4. Finally, we measured the similarity between teacher and student patterns. Specifically, the Jaccard index between the set of unique patterns of students and teachers is 0.101 (0.119) for Cooking (DIY) which shows the different language used in the different roles.

In Figure 4, we further look at the patterns' length. It shows higher variance in length in teacher utterances compared to student ones. We argue that student utterances are focused on asking for guidance and are usually not grounded in any document. On the other hand, there can be a large variety of possible responses for the teachers.

Finally, we further examined the frequency of the different patterns by computing the percentage of linguistic patterns appearing in different sentences. About 70% of the patterns appear only in a single sentence. This further demonstrates the linguistic diversity in the data. We also observe that the percentage of patterns with more than 10 appearances is substantial. These are common utterances across all conversations, such as asking for the next step or asking for tools/ingredients.

5 Experiments and Results

In this section, we experiment with two tasks: intent classification and abstractive question answering, to demonstrate the value of the `Wizard of Tasks` dataset for improving existing CTAs.

5.1 User Intent Classification

For this task, we aim to predict the intent of student utterances. Overall, there are six possible intents that students annotated according to Table 2. We will only focus on the four most common intents since we observed that the Chit-Chat and Other labels cover only about 1.0% of the data.

For modeling, we fine-tuned a pre-trained BERT (Devlin et al., 2018) model with the collected conversations. We experimented with two variants: i) encoding only the last user utterance (Utt_i); ii) encoding the last utterance along with the k previous turns ($Utt_i + Utt_{i-1}$ ... Utt_{i-k}).

With these experiments, we aim to verify the potential improvements provided by contextual information. The assumption is that the context could provide useful information in disambiguating intents, especially when ambiguous (e.g., Step Questions vs. Ingredients/Tools Questions).

In the first setting, each turn i is encoded as $[CLS]\ Utt_i\ [SEP]$. The second setting uses $[CLS]\ Utt_i\ [SEP]\ Utt_{i-1}$ ... $Utt_{i-k}\ [SEP]$.[12]

The final classification is done by applying a linear transformation layer to the encoding of the [CLS] token where intent probabilities are computed using the softmax function.

Experimental Setup and Results. To evaluate the performance of the models, we used 5-fold cross validation. We trained all models for 10 epochs with early stopping. In Table 5, we report the performance of the model as a function of k, the number of previous turns. The table includes the

[12]Notice that more conversational oriented encodings could be adopted, but this is out of the scope of this analysis.

Linguistic Pattern	Examples
Students	
dobj aux nsubj ROOT advcl punct	What do I do when the skillet is hot? | What should I do once I'm done slicing the omelet?
dobj aux nsubj ROOT advmod punct	What can I do now? | What am I doing next?
dobj aux nsubj ROOT prep punct	What do I do after heating the oil? | What should I do after my oven preheats?
Teachers	
nsubj aux ROOT dobj punct	I've included the step for reference. | You will need tofu and various seasonings.
nsubj ROOT xcomp punct	The first step is to combine ingredients in a blender. | The first step is to heat the oil.
nsubj aux ROOT xcomp punct	You will need to swirl the pan to coat grains with oil. | You are going to cover with a plate or a heavy can to drain.

Table 3: The three most frequent linguistic patterns in student and teacher utterances (Cooking); only patterns with at least two dependencies are included in the table.

	Cooking		DIY	
	Student	Teacher	Student	Teacher
# Sentences	5764	5621	7609	8244
# Unique Patterns	1606	1934	2128	2349
Avg. # Sentences per Pattern	3.6	2.9	3.6	3.5
Avg. Pattern Length	4.4	4.3	4.6	4.2

Table 4: Linguistic Pattern Statistics. "Avg. # Sentences per Pattern" is the avg. number of sentences expressing a pattern (# Sentences / # Unique Patterns).

k	Accuracy	F1				
		Macro-Avg	Ingr./Tools Question	Steps Question	Request Step	Stop
			Cooking			
0	0.876	**0.867**	**0.736**	0.822	0.939	0.970
1	0.877	0.865	0.720	0.829	0.939	0.970
3	**0.879**	**0.867**	0.720	0.834	0.939	**0.974**
5	**0.879**	0.865	0.715	**0.835**	**0.940**	0.969
			DIY			
0	0.863	**0.821**	**0.599**	0.805	**0.936**	0.944
1	0.857	0.812	0.579	0.797	0.931	0.942
3	0.862	0.816	0.581	0.810	0.933	0.941
5	**0.864**	**0.821**	0.592	**0.812**	0.933	**0.947**

Table 5: IC results w.r.t. the number of previous conversation turns (k). In bold, the best result in a column.

Test Domain	Cooking		DIY	
Train Domain	Cooking	DIY	DIY	Cooking
Ingr./Tools Question	0.736	0.538	0.599	0.454
Steps Question	0.822	0.804	0.805	0.771
Request Step	0.939	0.929	0.936	0.926
Stop	0.970	0.959	0.944	0.931
Macro-F1	0.867	0.807	0.821	0.771

Table 6: Comparing the performance of an intent classification model in a cross-domain setting ($k = 0$).

overall performance of the model (Accuracy and Macro-F1) as well as the F1 score of each label.

As shown in Table 5, the overall accuracy of the model is higher than 0.85 for both domains and for all values of k. The results show that the classification task is slightly more challenging for the DIY domain than Cooking. A possible explanation is the linguistic differences (e.g., the length of utterances and the number of linguistic patterns) between the two domain, as analyzed in Section 4. Surprisingly, we did not observe noticeable difference in F1 after using the conversation history, possibly due to a limited number of training conversations.

Next, focusing our attention on the per-label performance in Table 5, we can see that some labels are substantially harder to predict than others. Specifically, the F1 of Request Step is at least 12% (27%) larger than the F1 of Steps Questions (Ingredients/Tools Questions) for both domains and all values of k. A possible reason for this can be that student questions can be very specific to the task performed while other type of utterances (e.g., Request Step utterances) use language that is shared across tasks to a greater extent. Breaking out the performance by labels, we can also see that using the conversational history improves the performance of some labels and lowers the performance of others. For example, the F1 of Steps Question improves from 0.822 (0.805) to 0.835 (0.812) in Cooking (DIY) when changing the value of k from 0 to 5; an opposite trend is observed in Ingr./Tools Questions. This result suggests that the amount of conversational history used by the model should potentially vary across different types of utterances, which is an interesting direction for future work.

Finally, in Table 6, we analyze the performance of using a model that was trained on one domain to predict the intent of utterances in the other domain. The results show the overall importance of domain-specific information for the task. Still, the necessary information for the prediction of some labels is shared between domains to a greater extent than in the case of other labels. For example, while in the case of Request Step both models perform similarly in both domains, using the cross-domain model substantially deteriorates the performance of the Ingr./Tools Question prediction.

Error Analysis. We observed that our model produced the least number of errors on the Stop intent, probably because Stop utterances often contain informative keywords (e.g., "stop" and "done"). We also observed that some Request Step utterances can contain similar keywords (e.g., "I am done with this step"). This explains why sometimes the model confused a Stop intent with a Request Step intent in both the Cooking and the DIY domains.

We also observed a challenge in distinguishing between Ingredient/Tool Question intents and Steps Question intents, primarily due to some ambiguity between these question types (Table 7). For instance, the model predicted Steps Question when the user asked about *plastic bag* substitution, but predicted Ingredient/Tool Question for *zipper* substitution. Conversely, the model predicted Ingredient/Tool Question when the user asked about *edge sander*. This is because while the first part of the sentence is a substitution question, the second sentence actually links back *edge sander* to the next step. For further analysis, refer to Appendix E.

Cooking:
True: Request Step, Predicted: Steps Question
Do I need to let the cake cool first?
True: Ingr./Tools Question, Predicted: Steps Question
Does the vinegar's flavor profile change when it becomes a syrup?
True: Steps Question, Predicted: Ingr./Tools Question
Should these other ingredients be sprinkled in a specific order?
True: Steps Question, Predicted: Request Step
What do I need to do in order to get my grill prepped?
DIY:
True: Request Step, Predicted: Steps Question
Now it is planted. Should I water the plant?
True: Ingr./Tools Question, Predicted: Steps Question
Could I substitute a plastic bag if I don't have a paper bag?
True: Steps Question, Predicted: Ingr./Tools Question
Can you teach me how to replace a zipper?
I have an edge sander. Do I need it for the next step?
True: Steps Question, Predicted: Request Step
What should I do if I see a few drips of water when I run the water?

Table 7: Intent classification errors.

5.2 Abstractive Question Answering

In this task, we seek to generate natural language answers for the questions asked by student workers. The ground truth answers are provided by the teacher workers. We focus on utterances with the following student intents to prepare our dataset for this task: Step Questions and Ingredients/Tools Questions (from Table 8). We randomly sample 80% of the available question-answer (QA) pairs as training data, with the remaining 20% equally split into validation and test sets. We only use information from the available document (recipe or article) and the conversational history to generate answers to questions, without considering any external knowledge. Thus, our test set only consists of QA pairs where answers from the teacher do not contain any links to external resources.

Domain	Total # QA pairs	# Internal QA pairs
Cooking	1378	538
DIY	1435	684

Table 8: Statistics of student-teacher QA pairs.

Experimental Setup and Results. We fine-tune two state-of-the-art pre-trained language models, BART (Lewis et al., 2019) and T5 (Raffel et al., 2019) with our identified QA pairs for both domains, for the task of natural language answer generation. As input, we provide the models the user question and a context, which consist of document (recipe or article) text, a list of ingredients or tools associated with the document, and the prior conversational history. We evaluate different variations of the context using the natural language generation metrics of BLEU, ROUGE and BERT-score (Zhang et al., 2019). In the interest of space, Table 10 shows the results for the two fine-tuned models, using two turns of conversational history and the entire content of the document as input context for both domains. We also report as a baseline answering model, the document step most similar to the input question (MSS). We compute the most similar step according to the cosine similarity between the steps' and questions' representations, obtained by using Sentence-BERT (Reimers and Gurevych, 2019). BART outperforms MSS in both domains, while T5 only outperforms MSS on BERT-score in the Cooking domain. MSS performs better on DIY, than the Cooking domain. A possible explanation could be that for DIY tasks teachers seem to rely more on the document content to answer questions, whereas for cooking tasks, they are more likely to summarize or paraphrase the document content to generate answers. We also observe that the fine-tuned BART model outperformed the T5 model by about 1-13% across different context settings and domains in terms of generated answer quality. This may be due to a smaller number of allowed input tokens (512 for T5 vs 1024 for BART), since the average (standard deviation) of input context length is 286 (80) for the recipe domain and 1532 (451) for the DIY domain.

We present the BERT-score for the fine-tuned BART model in Figure 5, across different contextual settings for both domains. We observe a maximum model performance of 0.27 BERT-score F1 points for both the Cooking and DIY domains. Including conversational history as part of the input context improves the answer generation performance by 1-3% across both domains. This gain is further enhanced by 1-2% with the addition of the list of tools/ingredients to the context. In general, the inclusion of a greater number of document steps in the context contributes the most to the answer

generation quality, followed by the conversational history and the list of ingredients/tools.

The low performance (<0.3) using state-of-the-art NLG models for both domains emphasizes the challenges involved in solving tasks associated with low-resource settings and indicates that there is still a lot of room for improvement. We observed that these models commit various types of factual errors and stylistic or grammatical errors during answer generation (examples can be found in Table 9 and in Appendix F), which points out the scope of tackling these open research problems in the future. In addition, in this work we simply concatenated the different sources of information (conversational history, list of tools/ingredients, document steps) to create the input context. Evaluating different ways to generate a concise, useful context, with the possible use of some external knowledge, can also help improve the answer quality.

Error Analysis. Overall, we observed a better sentence structure and a much fewer number of grammatical or stylistic errors in the generated answers for the DIY domain, as compared to the Cooking domain (Table 9). Our model produced the largest number of factual errors which are numerical in nature, as compared to the other types of errors. This includes both hallucinating numerical terms that did not exist in the input context, as well as generating numbers or units that are different from their ground truth values. This is possibly because the pre-trained NLG language models we used are not well trained or equipped to encode and predict the numerical information. Incorporating the list of ingredients/tools in the input text when generating an answer helps reduce the erroneous generation of specific ingredients, tools or materials used in the tasks (e.g., "knife" vs. "thick spoon" in the top part of Table 9 or the definition of "cheesecloth" in the bottom part).

Cooking:
Factual, Numerical Errors → Correct Answer
Bake the cake for **15 minutes** → You should bake for 20 minutes.
Factual, Non-numerical Errors → Correct Answer
You can **turn on a stovetop** and cook until smooth. → No stovetop is required.
Grammatical/Stylistic Errors → Correct Answer
Yes can use the same knife. → You can use the same knife
DIY:
Factual, Numerical Errors → Correct Answer
Yes, it takes about **8 to 10 days** for the seeds to grow. → The sprouts take four days to grow.
Factual, Non-numerical Errors. → Correct Answer
The cheesecloth is a **thin layer of plastic**. → A cheesecloth is a loosely woven cotton cloth.
Grammatical/Stylistic Errors → Correct Answer
Yes, there are any safety concerns with opening the valve. → Open with caution in well ventilated room and no flame exposed.

Table 9: Abstractive QA errors. Erroneous terms in utterances are boldfaced.

Model	Ctxt.	Hist.	BLEU	ROUGE	BERTScore
Cooking					
MSS	None	0	0.077	0.148	0.071
T5-base	All	2	0.066	0.136	0.119
BART-base	All	2	0.116	0.211	0.270
DIY					
MSS	None	0	0.125	0.208	0.224
T5-base	All	2	0.053	0.134	0.183
BART-base	All	2	0.119	0.235	0.276

Table 10: Abstractive QA results. 'Ctxt.' and 'Hist.' refer to context and history, respectively.

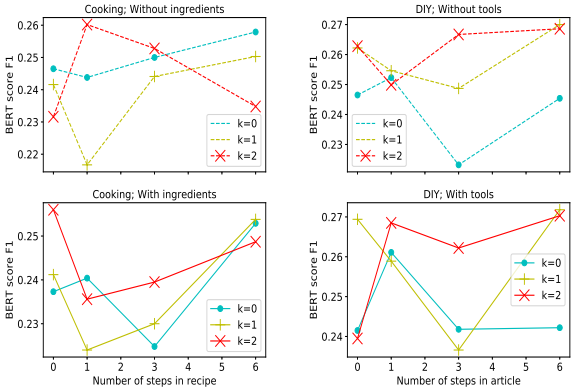

Figure 5: BERT-score of fine-tuned BART using different context settings on Cooking (left) and DIY (right). k denotes history size, while the x-axis shows the number of recipe/article steps used as part of the input context. The upper two figures (dashed line) refer to using context without ingredients/tools, whereas the lower two figures refer to using them.

6 Conclusions

In this paper, we presented `Wizard of Tasks`, a novel dataset for conversations in the area of Conversational Task Assistants. This dataset is inspired by the Alexa Prize TaskBot Challenge, where the participants have to create a dialog agent on top of the Alexa platform to help users in performing real world tasks in two domains: Cooking and Home Improvement. To the best of our knowledge, our task-oriented conversational dataset is the first of its kind in these two domains. We discussed our dataset collection asynchronous strategy in a crowdsourcing setting. We reported multiple analyses of the collected data as well as initial experimental evaluations on two tasks: Intent Classification and Abstractive Question Answering. Despite the small size of the collected data, we hope it can foster research in the novel area of CTAs. In the future, we plan to expand the dataset to additional domains.

References

Daniel G. Bobrow, Ronald M. Kaplan, Martin Kay, Donald A. Norman, Henry S. Thompson, and Terry Winograd. 1977. Gus, A frame-driven dialog system. *Artif. Intell.*, 8(2):155–173.

Paweł Budzianowski, Tsung-Hsien Wen, Bo-Hsiang Tseng, Iñigo Casanueva, Stefan Ultes, Osman Ramadan, and Milica Gasic. 2018. Multiwoz-a large-scale multi-domain wizard-of-oz dataset for task-oriented dialogue modelling. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 5016–5026.

Danqi Chen, Adam Fisch, Jason Weston, and Antoine Bordes. 2017. Reading Wikipedia to answer open-domain questions. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1870–1879, Vancouver, Canada. Association for Computational Linguistics.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*.

Emily Dinan, Stephen Roller, Kurt Shuster, Angela Fan, Michael Auli, and Jason Weston. 2018. Wizard of wikipedia: Knowledge-powered conversational agents. *arXiv preprint arXiv:1811.01241*.

Layla El Asri, Hannes Schulz, Shikhar Kr Sarma, Jeremie Zumer, Justin Harris, Emery Fine, Rahul Mehrotra, and Kaheer Suleman. 2017. Frames: a corpus for adding memory to goal-oriented dialogue systems. In *Proceedings of the 18th Annual SIGdial Meeting on Discourse and Dialogue*, pages 207–219.

Mihail Eric, Rahul Goel, Shachi Paul, Abhishek Sethi, Sanchit Agarwal, Shuyang Gao, Adarsh Kumar, Anuj Goyal, Peter Ku, and Dilek Hakkani-Tur. 2020. Multiwoz 2.1: A consolidated multi-domain dialogue dataset with state corrections and state tracking baselines. In *Proceedings of the 12th Language Resources and Evaluation Conference*, pages 422–428.

Karthik Gopalakrishnan, Behnam Hedayatnia, Qinglang Chen, Anna Gottardi, Sanjeev Kwatra, Anu Venkatesh, Raefer Gabriel, Dilek Hakkani-Tür, and Amazon Alexa AI. 2019. Topical-chat: Towards knowledge-grounded open-domain conversations. In *INTERSPEECH*, pages 1891–1895.

Anna Gottardi, Osman Ipek, Giuseppe Castellucci, Shui Hu, Lavina Vaz, Yao Lu, Anju Khatri, Anjali Chadha, Desheng Zhang, Sattvik Sahai, Prerna Dwivedi, Hangjie Shi, Lucy Hu, Andy Huang, Luke Dai, Bofei Yang, Varun Somani, Pankaj Rajan, Ron Rezac, Michael Johnston, Savanna Stiff, Leslie Ball, David Carmel, Yang Liu, Dilek Hakkani-Tur, Oleg Rokhlenko, Kate Bland, Eugene Agichtein, Reza Ghanadan, and Yoelle Maarek. 2022. Alexa, let's work together: Introducing the first alexa prize taskbot challenge on conversational task assistance. In *Alexa Prize TaskBot Challenge Proceedings*.

Kazushi Ikeda and Keiichiro Hoashi. 2018. Utilizing crowdsourced asynchronous chat for efficient collection of dialogue dataset. In *Proceedings of the AAAI Conference on Human Computation and Crowdsourcing*, volume 6.

Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Ves Stoyanov, and Luke Zettlemoyer. 2019. Bart: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. *arXiv preprint arXiv:1910.13461*.

Christopher D. Manning, Prabhakar Raghavan, and Hinrich Schütze. 2008. *Introduction to Information Retrieval*. Cambridge University Press, USA.

Nikita Moghe, Siddhartha Arora, Suman Banerjee, and Mitesh M Khapra. 2018. Towards exploiting background knowledge for building conversation systems. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 2322–2332.

Seungwhan Moon, Satwik Kottur, Paul A Crook, Ankita De, Shivani Poddar, Theodore Levin, David Whitney, Daniel Difranco, Ahmad Beirami, Eunjoon Cho, et al. 2020. Situated and interactive multimodal conversations. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 1103–1121.

Filip Radlinski and Nick Craswell. 2017. A theoretical framework for conversational search. In *Proceedings of the 2017 Conference on Conference Human Information Interaction and Retrieval*, CHIIR '17, page 117–126, New York, NY, USA. Association for Computing Machinery.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J Liu. 2019. Exploring the limits of transfer learning with a unified text-to-text transformer. *arXiv preprint arXiv:1910.10683*.

Abhinav Rastogi, Xiaoxue Zang, Srinivas Sunkara, Raghav Gupta, and Pranav Khaitan. 2020. Towards scalable multi-domain conversational agents: The schema-guided dialogue dataset. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pages 8689–8696.

Nils Reimers and Iryna Gurevych. 2019. Sentence-bert: Sentence embeddings using siamese bert-networks. *arXiv preprint arXiv:1908.10084*.

Pararth Shah, Dilek Hakkani-Tür, Gokhan Tür, Abhinav Rastogi, Ankur Bapna, Neha Nayak, and Larry Heck. 2018. Building a conversational agent overnight with dialogue self-play. *arXiv preprint arXiv:1801.04871*.

Tsung-Hsien Wen, David Vandyke, Nikola Mrkšić, Milica Gašić, Lina M. Rojas-Barahona, Pei-Hao Su, Stefan Ultes, and Steve Young. 2017. A network-based end-to-end trainable task-oriented dialogue system. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 1, Long Papers*, pages 438–449, Valencia, Spain. Association for Computational Linguistics.

Jason D Williams, Antoine Raux, and Matthew Henderson. 2016. The dialog state tracking challenge series: A review. *Dialogue & Discourse*, 7(3):4–33.

Xiaoxue Zang, Abhinav Rastogi, Srinivas Sunkara, Raghav Gupta, Jianguo Zhang, and Jindong Chen. 2020. Multiwoz 2.2: A dialogue dataset with additional annotation corrections and state tracking baselines. In *Proceedings of the 2nd Workshop on Natural Language Processing for Conversational AI*, pages 109–117.

Saizheng Zhang, Emily Dinan, Jack Urbanek, Arthur Szlam, Douwe Kiela, and Jason Weston. 2018. Personalizing dialogue agents: I have a dog, do you have pets too? In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 2204–2213.

Tianyi Zhang, Varsha Kishore, Felix Wu, Kilian Q Weinberger, and Yoav Artzi. 2019. Bertscore: Evaluating text generation with bert. *arXiv preprint arXiv:1904.09675*.

Tiancheng Zhao and Maxine Eskenazi. 2018. Zero-shot dialog generation with cross-domain latent actions. In *Proceedings of the 19th Annual SIGdial Meeting on Discourse and Dialogue*, pages 1–10.

Kangyan Zhou, Shrimai Prabhumoye, and Alan W Black. 2018. A dataset for document grounded conversations. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 708–713.

Li Zhou, Jianfeng Gao, Di Li, and Heung-Yeung Shum. 2020. The Design and Implementation of XiaoIce, an Empathetic Social Chatbot. *Computational Linguistics*, 46(1):53–93.

A Data Collection User Interface Design

As shown in Figure 6 and 7, we developed four widgets to optimize the presentation: i) annotation guidelines; ii) chat history; iii) task content; and iv) annotation questions and answers. All of these widgets are slightly tuned to support both teacher and student roles. For example, we do not show any article/recipe content to students except for their title. On the other hand, every bit of information including a list of ingredient/tools, steps text, images, and summary are available to teachers. There are also variations in annotation guidelines and the type of questions we ask to students vs. teachers.

For chat history, we always show the entire conversation history since our data collection is done asynchronously. This is important because for anyone to contribute to an existing conversation, understanding the context is a prerequisite for generating any next response. Workers were also able to click the "Share" button to see the information that was shared on a particular turn. We developed these mechanisms to help workers to quickly figure out what is happening inside different conversations and decide the next possible response.

B Additional Details of the Data Collection

In total, 238 (159) crowd workers participated in the Cooking (DIY) experiments and the average number of utterances per worker is 33.2 (63). The average number of unique tasks per worker is 20.1 (35.1) for Cooking (DIY). Finally, we observed that 93 workers participated in both tasks. The difference between domains might be due to the higher complexity and diversity in DIY tasks compared to Cooking, which might attract fewer workers that are interested in participating in the task.

C Examples of Collected Conversations

We report in Table 18 and 19 two `Wizard of Tasks` conversations from the Cooking and the DIY domain.

D Additional Dataset Statistics

In Table 12, we report the distribution of utterance length in terms of the number of sentences. The average number of sentences per utterance is slightly higher for teachers (i.e., 1.4/1.5 and 1.5/1.7 average sentences per utterance for students and teachers, respectively, in Cooking/DIY). In general, the low values demonstrate the conversational nature of the data set: even though teacher utterances have more words, the workers used a small number of sentences to construct them.

In Table 13, we further examine the frequency of the different patterns. In the table, we can see the percentage of linguistic patterns with a specific number of sentences that they appear in. The results show that about 70% of patterns appear only in a single sentence which demonstrates the linguistic diversity in the data. We also observe that the percentage of patterns with more than 10 appearances is substantial. These patterns can be common utterances across all conversations, such as asking for the next step or asking for a list of tools or ingredients.

E Error Analysis: Intent Classification

We present the confusion matrix for both domains in Table 14 and 15 to see the performance trade-offs between different intent labels.

The tables show that our model produced the least number of errors on the Stop intent. This is expected because Stop utterances often contain informative keywords (e.g., "stop" and "done"). Interestingly, we also observed that some Request Step utterances can contain similar keywords (e.g., "I am done with this step"). This explains why sometimes the model confused a Stop intent with a Request Step intent in both the Cooking and the DIY domains.

The confusion matrices also demonstrate the great challenge in distinguishing between Ingredient/Tool Question intents and Steps Question intents. Although our intent labels were designed to capture the differences between these two types of questions, we noticed that this boundary can become ambiguous from time-to-time.

F Error Analysis: Abstractive Question Answering

In this section, we report the errors committed by our fine-tuned BART answer generation model in the abstractive question answering task, where it fails to generate the correct answer to a user's question. A sample of the different types of errors are shown in Tables 16 and 17, and the erroneously generated terms are shown in bold.

Linguistic Pattern	Examples
Students	
dobj aux nsubj ROOT prep punct	What should I do after that? | What should I do after getting rid of the clippings?
ccomp punct dobj aux nsubj ROOT advmod punct	Okay I have watered them what do I do now? | I've gathered the materials requested what do i do next?
dobj aux nsubj ROOT advmod punct	What should I do next? | What do I do now?
Teachers	
nsubj aux ROOT dobj punct	I've shared details about the mulch here. | Shade will reduce the potency of the lavender plant.
nsubj aux ROOT dobj advmod punct	I've shared the details here. | You're doing great so far!
nsubj ROOT xcomp punct	We want to water our soil until saturated. | A sauna helps to cleanse the skin and make you feel healthy.

Table 11: The three most frequent linguistic patterns in student and teacher utterances (DIY); only patterns with at least two dependencies are included in the table.

	Cooking		DIY	
# Sentences	Teacher	Student	Teacher	Student
1	63%	68%	53%	65%
2	29%	24%	32%	27%
3	5%	7%	11%	7%
>3	2%	1%	4%	2%

Table 12: The distribution of number of sentences for teacher and student utterances.

	Cooking		DIY	
# Sentences	Student	Teacher	Student	Teacher
$(0, 1]$	69%	73%	70%	71%
$(1, 10]$	26%	23%	26%	24%
$(10, \infty)$	5%	4%	4%	4%

Table 13: Pattern Frequency. The percentage of patterns that appear in a given number of sentences.

	Request Step	Ingr./Tools Question	Steps Question	Stop
Request Step	1799	14	49	11
Ingr./Tools Question	24	427	112	1
Steps Question	133	156	1046	2
Stop	2	1	0	276

Table 14: Intent classification confusion matrix (Cooking).

	Request Step	Ingr./Tools Question	Steps Question	Stop
Request Step	2538	18	101	10
Ingr./Tools Question	41	302	231	0
Steps Question	166	115	1296	11
Stop	10	0	3	292

Table 15: Intent classification confusion matrix (DIY).

Factual, Numerical Errors	**Correct Answer**
Bake the cake for **15 minutes**	You should bake for 20 minutes.
You should preheating the oven to **475F**.	It takes about 15-20 minutes to preheat the oven.
Yes, the recipe **does not include popcorn**.	Yes! You'll need 2 cups of popcorn.
Factual, Non-numerical Errors	**Correct Answer**
You can **turn on a stovetop** and cook until smooth.	No stovetop is required.
You can use a **knife**.	You will need a thick spoon.
Grammatical/Stylistic Errors	**Correct Answer**
Yes can use the same knife.	You can use the same knife
You should preheating the oven to 475F.	You should preheat the oven to 475F.

Table 16: Abstractive QA errors (Cooking). Erroneous terms in boldface.

Factual, Numerical Errors	**Correct Answer**
Yes, you will need to cut them every **two weeks**	Yes because they prevent the rhubarb from growing big and strong
Yes, it takes about **8 to 10 days** for the seeds to grow.	The sprouts take four days to grow.
Factual, Non-numerical Errors.	**Correct Answer**
The cheesecloth is a **thin layer of plastic**.	A cheesecloth is a loosely woven cotton cloth.
The soil will have to be **moist**.	The soil just needs to be very loose with chunks of bark and other organic matter
Grammatical/Stylistic Errors	**Correct Answer**
Yes, there are any safety concerns with opening the valve.	Open with caution in well ventilated room and no flame exposed.
Yes, there is no set amount of space in a small closet.	A great option is to use space bags. I've shared more information with you.

Table 17: Abstractive QA errors (DIY). Erroneous terms in boldface.

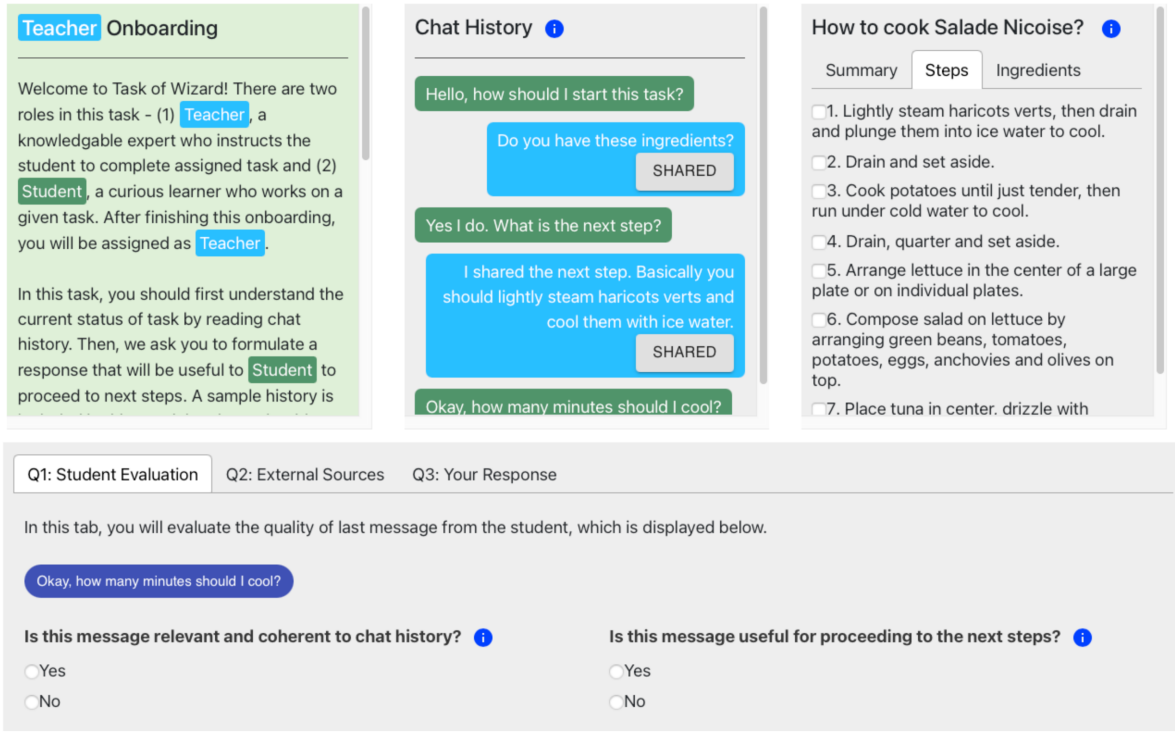

Figure 6: Teacher task user interface. Teachers are expected to find and provide relevant answers to students' questions. Teachers can also use the checkmarks in the task information section to share selected content if the textual response is not sufficient to deliver the full information.

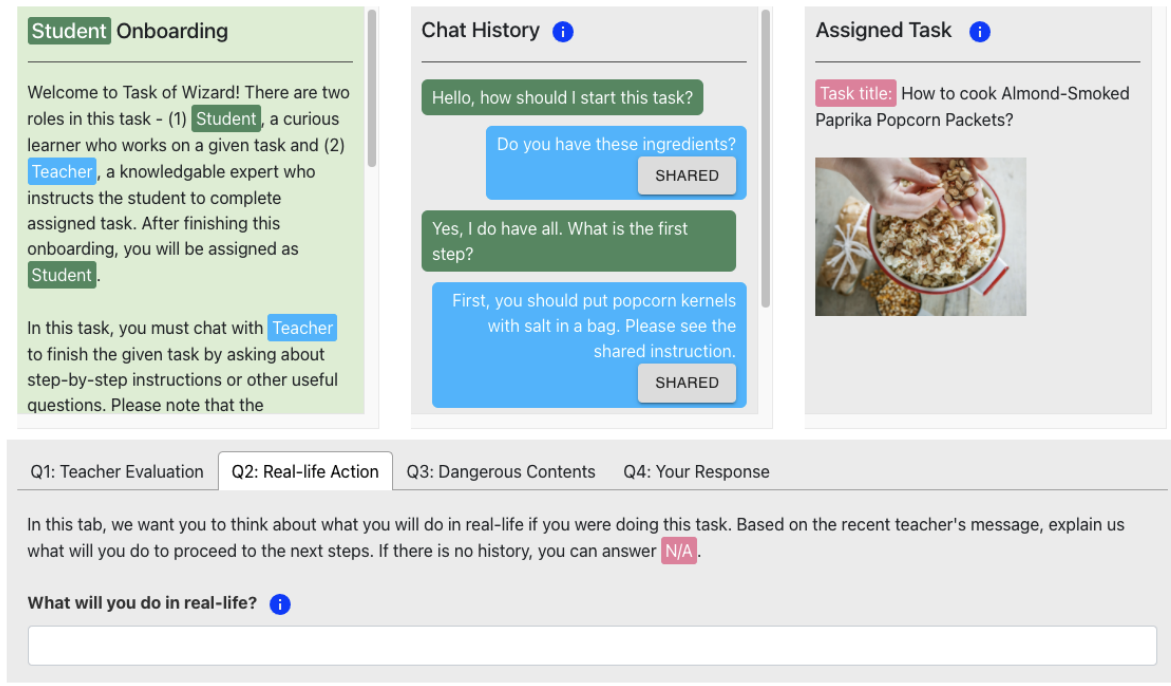

Figure 7: Student task user interface. Students are expected to understand the task status at each turn and engage with the teacher to progress toward task completion. The students can click on "Shared" button to view shared content of the current and previous turns. By default, only the task title and image are shown to the student.

	Utterance	Notes
User	*What do I need besides Salmon for this recipe?*	
Agent	*Here is what you need*	[Teacher shares some ingredients]
User	*What is the first step that should be performed after preparing these ingredients?*	
Agent	*The first step is to cut the salmon. I've shared below*	[Teacher shares a step]
User	*Now that the salmon is in the bowl, what is next?*	
Agent	*The next step is to mix it with some seasoning. I've shared the step below*	[Teacher shares a step]
User	*What specific seasonings will I need?*	
Agent	*You will need soy sauce, sesame oil, and sesame seeds.*	[Teacher shares some ingredients]
User	*The salmon and seasonings are now mixed in the bowl. What comes next?*	
Agent	*Refrigerate the mixture for a minimum of 15 minutes.* *You can leave it in for up to 4 hours for better flavor.*	
User	*Does the refrigeration time affect the texture of the mixture?*	
Agent	*It will not affect the texture, it just makes the flavors meld together better.*	
User	*Am I done? let me know please?*	
Agent	*Yes, now you're just plating and enjoying!*	[Teacher shares the last step]
User	*I have an odd question, Why is this called Poke (Salmon poke)?*	
Agent	*Poke means "cut into pieces", and is how Hawaiians first described the dish.*	[Answer found in external URL]
User	*Good to know! Thanks for the help, it's time to go eat now!*	

Table 18: Example of one full `Wizard of Tasks` conversation from the Cooking domain.

	Utterance	Notes
User	*Can you help me with cleaning of wool rugs?*	
Agent	*Sure! First you should gather all the supplies needed that I have listed.*	[Teacher shares some tools]
User	*What should I do now I have the supplies?*	
Agent	*Start with taking the rug outside and shaking it off,* *using a broom to help get off the dirt and dust.*	[Teacher shares a step]
User	*After I've shaken as much dust and dirt out of the rug,* *what do I need to do a deeper job, that doesn't seem like enough.*	
Agent	*No, that in itself won't be enough. Now run a vacuum cleaner over your rug.*	[Teacher shares a step]
User	*Ok, the rug has been shaken of dirt and dust and then vacuumed...* *is there anything else to do?*	
Agent	*Yes, you will need to shampoo your rug.* *Please follow the attached instruction for shampooing your rug.*	[Teacher shares a step]
User	*Do I need to use a sponge or can I use a cleaning scrub brush of* *sorts to get out extras stains? Is this a situation that I can* *deviate a little or do I need to be precise?*	
Agent	*A sponge is less harsh but either will work.*	
User	*I've rubbed a lot of soap through it and rinsed until the soap is gone.* *What should I do with it next?*	
Agent	*You want to dry your rug immediately*	[Teacher shares a step]
User	*Why does the rug need to be dried immediately?*	
Agent	*Because it needs extensive drying to get rid of the moisture*	[Teacher shares the last step]
User	*Squeezing it with my hands left a lot of water still in.* *Could I make a pad of dry towels on the floor, then lay the rug down,* *then put more towels on top and apply pressure to it to blot up more water?*	
Agent	*That would work you could also try hang drying the wool rug for the final step*	
User	*Okay, good to know! What should I do after drying the rug?*	
Agent	*Cleaning and drying wool rug is complete*	
User	*Could the steps that I just took for my wool rug work on my cotton rug as well?*	
Agent	*I would check the care tag on the rug before doing anything.*	
User	*What about these instructions makes this care particularly* *well-suited for wool rugs?*	
Agent	*Wool is very delicate and needs a lot of care*	
User	*Thanks for the help cleaning my rug!*	

Table 19: Example of one full `Wizard of Tasks` conversation from the Home Improvement (DIY) domain.

K-MHaS: A Multi-label Hate Speech Detection Dataset in Korean Online News Comment

Jean Lee[1] **Taejun Lim**[1] **Heejun Lee**[3] **Bogeun Jo**[3]
Yangsok Kim[4] **Heegeun Yoon**[5] **Soyeon Caren Han**[1,2]*
The University of Sydney[1] The University of Western Australia[2]
BigWave AI[3] Keimyung University[4] National Information Society Agency[5]
{jean.lee, tlim6535, caren.han}@sydney.edu.au
{leej3471, iambgjo}@gmail.com
yangsok.kim@kmu.ac.kr hgyoon@nia.or.kr

Abstract

Online hate speech detection has become an important issue due to the growth of online content, but resources in languages other than English are extremely limited. We introduce K-MHaS[1], a new multi-label dataset for hate speech detection that effectively handles Korean language patterns. The dataset consists of 109k utterances from news comments and provides a multi-label classification using 1 to 4 labels, and handles subjectivity and intersectionality. We evaluate strong baselines on K-MHaS. KR-BERT with a sub-character tokenizer outperforms others, recognizing decomposed characters in each hate speech class.

1 Introduction

The growth of online content including social media (Zampieri et al., 2020), news comments (Gao and Huang, 2017), Wikipedia (Wulczyn et al., 2017), and in-game chat (Weld et al., 2021) presents challenges in detecting hate speech using advanced Natural Language Processing. Hate speech is language that attacks or diminishes individuals or groups based on certain characteristics such as physical appearance, religion, gender, or other attributes, and it can occur across different linguistic styles (Fortuna and Nunes, 2018). Hate speech detection is intrinsically a complex task (Wang et al., 2020) due to the fuzzy boundary with other overlapping concepts such as abusive language (Nobata et al., 2016), toxic comments (Wulczyn et al., 2017), or offensive language (Davidson et al., 2017).

Recently, the rise in popularity of Korean TV, movies, and music (e.g. Squid Game, BTS) has led to many young people showing an interest in learning Korean. This phenomenon could result in exposure to harmful content and hate speech in Korean. However, (1) the most common language in hate speech research is English and only limited resources are available in other languages such as Arabic (Mubarak et al., 2017), Dutch (Caselli et al., 2021), and Korean (Moon et al., 2020). In addition, most datasets are annotated (2) using a single label classification of particular aspects, even though the subjectivity of hate speech cannot be explained with a mutually exclusive annotation scheme.

We propose K-MHaS, a Korean multi-label hate speech detection dataset that allows overlapping labels associated with intersectionality, a concept from sociology that identifies combined attributes (Crenshaw, 1989). Our dataset consists of 109,692 utterances from Korean online news comments, labeled with 8 fine-grained hate speech classes. K-MHaS is compatible with previous work on hate speech in other languages, by providing binary classification and multi-label classification from 1(one) to 4(four) labels.

We investigate the K-MHaS dataset by analyzing label distribution, keywords, and label pairs. In addition, we provide strong baseline pre-trained language models using Multilingual-BERT, KoELECTRA, KoBERT, and KR-BERT, and compare the results using six metrics for multi-label classification tasks. Overall, the KoELECTRA model achieves the best performance for all labels, indicating the effects of the pre-training data source. The KR-BERT with a sub-character-level tokenizer outperforms the others on several label pairs, showing that decomposing various Korean characters is essential for the task. Our contribution can be summarized as follows:

- We propose a large size Korean multi-label hate speech detection dataset that represents Korean language patterns effectively;
- We propose a multi-label hate speech annotation scheme, which can handle the subjectivity of hate speech and the intersectionality;

*Corresponding author (caren.han@sydney.edu.au)
[1]The dataset is available at https://github.com/adlnlp/K-MHaS.

Proceedings of the 29th International Conference on Computational Linguistics, pages 3530–3538
October 12–17, 2022.

Publication	Language	Source	Data size	Labels	M-label
Waseem and Hovy (2016)	English	Twitter	16.2k	Sexism, Racism, Neither	N
Davidson et al. (2017)	English	Twitter	24.8k	Hate Speech, Offensive, Neither	N
Wulczyn et al. (2017)	English	Wikipedia comments	115k	Toxic, Severe Toxic, Obscene, Threat, Insult, Identity Hate, Neutral	Y
Ibrohim and Budi (2019)	Indonesian	Twitter	11k	(a) Individual, Group (b) Religion, Race, Pysical, Gender, Other (c) Weak, Moderate, Strong Hate Speech	P
Fortuna et al. (2019)	Portuguese	Twitter	5.6k	(a) Hate Speech, Not Hate Speech (b) Sexism, Body, Origin, Homophobia, Racism, Ideology, Religion, Health, Other-Lifestyle	P
Ousidhoum et al. (2019)	English French Arabic	Twitter	6k (EN) 4k (FR) 3k (AR)	Labels for five different aspects (a) Directness, (b) Hostility, (c) Target, (d) Group, and (e) Annotator	P
Moon et al. (2020)	Korean	News comments	9k	(a) Hate Speech, Offensive, None (b) Gender, Others, None	N
Ours	**Korean**	**News comments**	**109k**	**(a) Hate Speech, Not Hate Speech** **(b) Politics, Origin, Physical, Age, Gender Religion, Race, Profanity, Not Hate Speech**	**Y**

Table 1: Comparison of datasets. A "M-label" indicates a multi-label annotation scheme that allows overlapping labels for intersectionality (P = partially applied). The (a) - (e) indicates a layer containing a single label from each aspect.

- We evaluate strong baseline experiments on our dataset using Korean-BERT-based language models with six different metrics.

2 Korean Multi-label Hate Speech Detection Dataset (K-MHaS)

Our dataset is based on the Korean online news comments available on Kaggle [2] and Github [3]. The unlabeled raw data was collected between January 2018 and June 2020. In order to curate the data, we randomly select more than 109,692 news comments. Our data preprocessing is designed to tokenize a Korean character and to filter the length. We remove URLs and bad characters (e.g. U+1100 to U+11FF - Hangul Jamo) using regular expressions while keeping uppercase and lowercase letters in English and emoji. We discard sentences with fewer than 10 characters as it is often only one word. For the data derived from online comments, we normalized repeated characters by truncating their number of consecutive repetitions to two.

Multi-label Annotation We consider a multi-label annotation scheme in order to deliver fine-grained hate speech categories and intersectionality from the overlapping labels. The annotation scheme has two layers: (a) binary classification (*'Hate Speech' or 'Not Hate Speech'*) and (b) fine-grained classification (*8 labels or 'Not Hate Speech'*). For the fine-grained classification, a 'Hate Speech' class from the binary classification is broken down into 8 classes associated with the hate speech category [4]. As shown in Table 1, this scheme allows non-exclusive concepts, accounting for the overlapping shades of given categories. We select the 8 hate speech classes in order to reflect the social and historical context as the nature of hate speech is different in each language (Kang et al., 2020). For example, the '*politics*' class is chosen due to a significant influence on the style of Korean hate speech.

Annotation Instructions Given the subjectivity of the task and our annotation scheme, we perform a preliminary round to identify the topics of hate speech and develop annotation instructions. We begin with the common categories of hate speech found in literature and match the keywords for each category. After the preliminary round, we investigate the results to merge or remove labels in order to provide the most representative subtype labels of hate speech contextual to the cultural background. Our annotation instruction includes the criteria as follows: **Politics**: hate speech based on political stance; **Origin**: hate speech based on place of origin or identity; **Physical**: hate speech based on physical appearance (e.g. body, face) or disability; **Age**: hate speech based on age; **Gender**: hate

[2] https://www.kaggle.com/datasets/junbumlee/kcbert-pretraining-corpus-korean-news-comments

[3] https://github.com/kocohub/korean-hate-speech

[4] Fine-grained labels (matching in Korean) : Politics (정치성향차별), Origin (출신차별), Physical (외모차별), Age (연령차별), Gender (성차별), Religion (종교차별), Race (인종차별), and Profanity (혐오욕설)

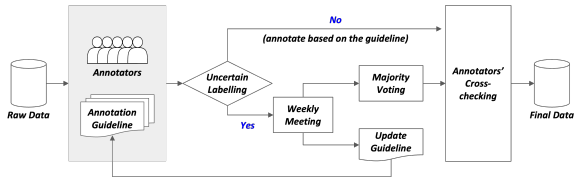

Figure 1: Overview of Annotation Process.

Label Types		Count (%)
Total Utterances		**109,692 (100%)**
Multi-label (Hate Speech)	1 label (Single)	36,470 (33.2%)
	2 labels	12,073 (11.0%)
	3 labels	1,440 (1.3%)
	4 labels	94 (0.1%)
Not Hate Speech		59,615 (54.3%)

Table 2: Dataset Statistics. The total is the combination of all '*hate speech*' and '*not hate speech*' label. Together the '*hate speech*' label makes up 45.7% of the data.

Class	Count - Single (%)	Count - Multi (%)
Politics	6,931 (19.0%)	4,961 (17.2%)
Origin	5,739 (15.7%)	4,458 (15.5%)
Physical	5,443 (14.9%)	3,364 (11.7%)
Age	4,192 (11.5%)	3,178 (11.0%)
Gender	3,348 (9.2%)	4,696 (16.3%)
Religion	1,862 (5.1%)	513 (1.8%)
Race	160 (0.4%)	163 (0.6%)
Profanity	8,795 (24.1%)	7,509 (26.0%)

Table 3: Fine-grained label distributions on *hate speech* labels. A '*not hate speech*' label is not included. A single means 1 label and a multi is the sum of 2, 3, and 4 labels. A multi-labeled data counts each overlapping class.

speech based on gender or sexual orientation (e.g. woman, homosexual); **Religion**: hate speech based on religion; **Race**: hate speech based on ethnicity; **Profanity**: hate speech in the form of swearing, cursing, cussing, obscene words, or expletives; or an unspecified hate speech category from above; and **Not Hate Speech**.

Our annotation instructions explain a two-layered annotation to (a) distinguish hate and not hate speech, and (b) the categories of hate speech. Annotators are requested to consider given keywords or alternatives of each category within social, cultural, and historical circumstances. For example, a comment using the word *"women"* is not hate speech, whereas, if it is critical of *"women"* or uses language that attacks the group, it is classified as *'gender'*. Notably, we annotate multi-labels if a comment includes several hate speech categories. Since hate speech can be varied, any comments in the form of swearing or cursing are marked as *'profanity'*. For instance, a comment containing hate speech about appearance, political stance, and gender in profane language (e.g. *"fuck you ugly communist bitch."*)[5] is labeled within *'physical'*, *'politics'*, *'gender'* and *'profanity'* classes.

Annotation Process Five native speakers were recruited for manual annotation in both the preliminary and main rounds. During the preliminary round, we facilitated the annotation instructions by conducting an annotators' discussion and providing some examples of keywords for each class. As shown in Figure 1, we introduced an iterative process that enables faster annotation in the main round. We provided an '*uncertain*' additional field that was used for the unspecified label in annotation guidelines or when the annotator had difficulties in choosing labels. Any '*uncertain*' labeled data was flagged by individual annotators, then reviewed by five annotators. The final labels were chosen based on the majority vote, and the annotation guidelines were updated to handle similar cases. Additionally, the other labeled data was reviewed, in line with the annotation guideline by two random annotators for the final dataset. The inter-annotator agreement returns an average Cohen Kappa score of 0.892, indicating substantial agreement (Siegel and Castellan, 1988).

[5] (Korean) "면상도 개 조가치 생겼네 개빨갱이년"

3 Dataset Analysis

K-MHaS dataset contains 109,692 comments as shown in Table 2. For binary classification, the proportion of the '*hate speech*' (45.7%) and '*not hate speech*' (54.3%) satisfies data balancing. The '*hate speech*' label consists of a single label (33.2%) and multi-labels (12.4%), containing from 2 to 4 labels. Other hate speech datasets reviewed have an approximate ratio of '*hate speech*' to '*not hate speech*' of around 40% (Vidgen and Derczynski, 2020). Our dataset is consistent with this figure, where the '*hate speech*' in a single label to '*not hate speech*' ratio is 38%.

Label Distribution Table 3 shows the fine-grained label distribution across our K-MHaS. For both single (s) and multi-label (m) distribution, the '*profanity*' class (24.1%-s, 26.0%-m) is more frequent than any other class, indicating that swear words are critical for detecting hate speech. Also, the '*religion*' (5.1%-s, 1.8%-m) and '*race*' (0.4%-s, 0.6%-m) classes are the smallest portions in both distributions, which are significantly more com-

Rank	Politics	Origin	Physical	Age
1	재앙 (1427)	짱깨 (615)	얼굴 (962)	틀 (1918)
2	문재인 (951)	전라도 (596)	돼지 (772)	나이 (599)
3	좌파 (464)	중국 (539)	여자 (294)	노인 (139)
4	좌빨 (402)	쪽 (448)	성형 (216)	충 (112)
5	빨갱이 (367)	짱 (446)	관상 (183)	놈 (106)
Rank	**Gender**	**Religion**	**Race**	**Profanity**
1	여자 (1704)	개독 (526)	흑인 (44)	새끼 (1103)
2	남자 (990)	신천지 (460)	백인 (32)	년 (1014)
3	페미 (172)	사이비 (409)	양키 (32)	지랄 (564)
4	맘충 (138)	종교 (305)	깜둥이 (19)	개 (459)
5	여성 (134)	예수 (227)	놈 (13)	놈 (404)

Table 4: Top 5 keywords associated with each fine-grained label. The number in brackets is the token count. The keyword analysis is from the total dataset and different from some examples in annotation guidelines.

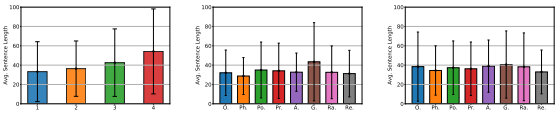

(a) Label types (b) Class types I (c) Class types II

Figure 2: Average utterance length. (a) label types from 1 to 4 labels. 8 class types (b) in a single label and (c) in multi-labels.

mon in other hate speech datasets. This difference could be because Korea is a highly homogenous monoculture with little variation in race and religion (Kang et al., 2020). Interestingly, the '*gender*' class (16.3%) occurs at almost twice the frequency in a multi-label distribution, compared to a single label distribution (9.2%). This indicates that gender-based hate speech is used extensively in combined aspects.

Keyword Analysis To understand the lexical aspects, we list the top 5 keywords for each hate speech category in Table 4, identifying which tokens are highly associated with each class. In the '*politics*' class, we find that far-right extremism is dominant, and new tokens such as *"catastrophe"* [jae ang](재앙) appears related to the former president's given name ([jae in]) as the two words are near-homophones. Across all classes, one-word tokens are often used in their stem form to modify the meanings of other words. For example, a token [teul] (틀) comes from the word *"denture"* [teulni] (틀니) which is used as an offensive reference to the elderly. In addition, one-word tokens can be used as a prefix (e.g. "dog" [gae] (개)) or a suffix (e.g. "insect" [chung] (충)), and combined with other neutral words to create a new offensive term.

Label Pair Analysis Figure 2 shows the average length of utterance by label count and class type.

Model	F1 (macro)	F1 (micro)	F1 (weighted)	E.M.	AUC	H.L. (↓)
BERT	0.6912	0.8139	0.8119	0.7579	0.8878	0.0464
KoELECTRA	0.7245	0.8493	**0.8480**	**0.7994**	**0.9122**	0.0380
KoBERT	**0.7651**	0.8413	0.8424	0.7926	0.9083	0.0401
KR-BERT-c	0.7444	**0.8500**	0.8470	0.7901	0.9028	**0.0368**
KR-BERT-s	0.7245	0.8445	0.8437	0.7825	0.9076	0.0390

Table 5: Overall multi-label classification performance on K-MHaS for the five baseline models at epoch 4 (E.M.:Exact Match, H.L.:Hamming Loss / KR-BERT-*: c = character-level, s = sub-character-level)

Metric	# labels	BERT	KoELECTRA	KoBERT	KR-BERT-c	KR-BERT-s
F1 (micro)	1	0.8190	0.8490	0.8320	**0.8553**	0.8392
	2	0.8043	0.8612	**0.8854**	0.8405	0.8703
	3	0.7517	0.7987	0.8290	0.7827	**0.8329**
	4	0.7093	0.7044	0.6832	0.7439	**0.7771**

Table 6: A breakdown of F1 for multi-label classification from 1 to 4 labels.

The total average length of an utterance is 33 tokens. An increase in the number of labels shows an increasing trend in utterance length, indicating that multi-labeled hate speech contains more linguistic content. The *'gender'* class has relatively longer lengths (43 tokens) compared to other classes in a single label, whereas all multi-labels utterances have a similar length. This indicates that the gender class has different linguistic features.

4 Experiment Setup

Data Preparation We split the data into train/test in the proportions of 0.8/0.2. From the training set, we randomly select 0.1 as a validation set (78,977/8,776/21,939 samples for train/val/test sets, preserving the class proportion). The data passed to the models is the preprocessed sentences and binary label vectors.

Baselines We select four baselines. 1) **Multi-BERT** (Wolf et al., 2020) is pre-trained on Wikipedia in 104 different languages. We adopted the BERT-Base, uses the WordPiece tokenizer and contains 110M parameters and 119K vocabs. 2) **KoELECTRA** (Park, 2020) is pre-trained on 34GB Korean news, Korean Wikipedia, Namuwiki (Korean-based wiki) and Modu (Korean corpus data publicly provided by the Korean government). The KoELECTRA-Small-v3 is used with the WordPiece tokenizer and contains 14M parameters and 35K vocabs. 3) **KoBERT** (SKTBrain, 2019) is pre-trained on 54M words from Korean Wikipedia, used the SentencePiece tokenizer, 92M parameters and 8K vocabs. 4) **KR-BERT** (Lee et al., 2020) is pre-trained on 2.47GB corpus with 233M words from Korean Wikipedia and news. We applied

Label Pairs	# pairs	F1 (macro)					F1 (micro)				
Overall Performance (F1)		**BERT**	**KoELECTRA**	**KoBERT**	**KR-BERT-c**	**KR-BERT-s**	**BERT**	**KoELECTRA**	**KoBERT**	**KR-BERT-c**	**KR-BERT-s**
		0.6912	0.7245	**0.7651**	0.7444	0.7245	0.8139	0.8493	0.8413	**0.8500**	0.8445
Profanity & Politics	323	0.1959	0.2045	**0.2072**	0.2013	0.2034	0.8379	0.8853	**0.9010**	0.8687	0.8616
Profanity & Physical	311	0.1931	0.2061	0.2115	0.2099	**0.2121**	0.8393	0.9096	0.9331	**0.9369**	0.9334
Profanity & Origin	269	0.1887	0.1989	0.1987	0.1961	**0.2050**	0.8144	0.8731	0.8729	0.8661	**0.9070**
Gender & Origin	242	0.2035	0.2017	0.2134	0.1905	**0.2141**	0.8920	0.8780	0.9440	0.8354	**0.9494**
Politics & Origin	224	0.1962	0.1976	0.1991	0.1872	**0.2013**	0.8666	0.8714	0.8846	0.8295	**0.8918**
Age & Politics	222	0.1996	**0.2114**	0.2104	0.1964	0.2014	0.8765	0.9329	**0.9357**	0.8734	0.8878
Gender & Profanity	181	0.1895	0.1991	0.1957	0.1911	**0.2054**	0.8157	0.8715	0.8542	0.8450	**0.8994**
Gender & Physical	177	0.1160	0.1833	**0.1958**	0.1813	0.1953	0.4562	0.7867	0.8455	0.8045	**0.8585**
Age & Profanity	132	0.1908	0.2102	**0.2139**	0.2063	0.2043	0.8414	0.9240	**0.9459**	0.9105	0.9095
Gender & Age	130	0.1738	0.1781	**0.1903**	0.1517	0.1339	0.7277	0.7452	**0.8159**	0.6368	0.5686

Table 7: F1 score for the top 10 two-label pairs on the K-MHaS dataset for the five pre-trained language models at epoch 4 (# total label pairs = 2,439 / KR-BERT-*: c = character-level tokenizer, s = sub-character-level tokenizer).

either (1) the character-level tokenizer or (2) the sub-character-level tokenizer[6].

Evaluation Metrics In multi-label classification, the prediction contains a set of labels, which means the prediction can be fully correct, partially correct, or fully incorrect. We propose to use the widely used six metrics (Godbole and Sarawagi, 2004) for conducting our multi-label classification, including F1-[macro, micro, weighted], Exact Match, AUC and Hamming Loss (Sorower, 2010).

5 Results

Evaluation for All Labels The overall performance for all labels is provided in Table 5. The F1(micro) range between 0.8139 (Multi-BERT), 0.8493 (KoELECTRA) and 0.8500 (KR-BERT-c), while the F1(macro) scores show a range from 0.6912 (Multi-BERT) to 0.7651 (KoBERT) with 4 epochs. We observe that all baselines achieve a similar performance, whereas Multi-BERT pre-trained on 104 languages present relatively lower performance. The KoELECTRA obtains overall the best or second best among six metrics, although this model has a seven times smaller parameter size (14M) than an average of other models (99M). This indicates the effects of the pre-training data source, considering that the KoELECTRA includes the corpus from Namuwiki and Modu that contain modern slang and buzzwords, while other models generally use Korean Wikipedia.

Evaluation for Multi-labels Table 6 shows the breakdown F1(micro) for multi-label classification from 1 to 4 labels[7]. A single label task, achieving 0.8553 and 0.8490 from the KR-BERT-c and KoELECTRA, outperforms other multi-label tasks due to domain similarity. For the multi-label classification, KR-BERT-s achieved the best performance. It uses a sub-character tokenizer that can decompose Hangul(Korean language) syllable characters into sub-characters. Therefore, it provides greater granularity in detecting hate speech words, by identifying the sub-characters from different hate speech categories.[8].

Evaluation for Label-pairs Table 7 shows the F1-[macro, micro] scores for curated label pairs based on the proportion in the 2-labels classification. It illustrates that the KR-BERT-s model outperforms in six label pairs. In particular, it is very effective at detecting the *origin and gender* pairs, achieving the highest F1 micro scores of 0.9494 across all label pairs and models. This model uses the sub-character-level tokenizer that can decompose various Korean characters (Hangul syllables) into sub-characters or graphemes to enable handling the bottom consonant (e.g. "gold-digger" [kko#t#baem] 꼬#ㅊ#뱀) or initial consonant (e.g. [k] ㅋ). This approach can detect new slang even if it is only a minor variation from other neutral words.

6 Conclusion

We propose K-MHaS, a new large-sized dataset for Korean hate speech detection with a multi-label annotation scheme. We provided extensive baseline experiment results, presenting the usability of a dataset to detect Korean language patterns in hate speech. In future work, the automatic hate speech moderation and counter-speech can be expanded.

[6]The KR-BERT tokenization variants can be found as follows: https://github.com/snunlp/KR-BERT#tokenization

[7]Further details are shown in Appendix Table 8.

[8](e.g.) 개빠르갱이넌 = 개 ("dog" - *profanity*) + 빠르갱이 ("communist" - *politics*) + 년 ("bitch" - *gender*)

Acknowledgements

We thank you the anonymous reviewers for their useful comments. This research is supported by an Australian Government Research Training Program (RTP) Scholarship.

References

Tommaso Caselli, Arjan Schelhaas, Marieke Weultjes, Folkert Leistra, Hylke van der Veen, Gerben Timmerman, and Malvina Nissim. 2021. Dalc: the dutch abusive language corpus. In *Proceedings of the 5th Workshop on Online Abuse and Harms (WOAH 2021)*, pages 54–66.

Kimberlé Crenshaw. 1989. Demarginalizing the intersection of race and sex: A black feminist critique of antidiscrimination doctrine, feminist theory and antiracist politics. *u. Chi. Legal f.*, page 139.

Thomas Davidson, Dana Warmsley, Michael Macy, and Ingmar Weber. 2017. Automated hate speech detection and the problem of offensive language. In *Proceedings of the International AAAI Conference on Web and Social Media*, volume 11.

Paula Fortuna, Joao Rocha da Silva, Leo Wanner, Sérgio Nunes, et al. 2019. A hierarchically-labeled portuguese hate speech dataset. In *Proceedings of the Third Workshop on Abusive Language Online*, pages 94–104.

Paula Fortuna and Sérgio Nunes. 2018. A survey on automatic detection of hate speech in text. *ACM Computing Surveys (CSUR)*, 51(4):1–30.

Lei Gao and Ruihong Huang. 2017. Detecting online hate speech using context aware models. In *Proceedings of the International Conference Recent Advances in Natural Language Processing, RANLP*, pages 260–266.

Shantanu Godbole and Sunita Sarawagi. 2004. Discriminative methods for multi-labeled classification. In *Pacific-Asia conference on knowledge discovery and data mining*, pages 22–30. Springer.

Muhammad Okky Ibrohim and Indra Budi. 2019. Multi-label hate speech and abusive language detection in indonesian twitter. In *Proceedings of the Third Workshop on Abusive Language Online*, pages 46–57.

Myungkoo Kang, Marie-Orange Rivé-Lasan, Wooja Kim, and Philippa Hall. 2020. *Hate speech in Asia and Europe: Beyond hate and fear*. Routledge.

Sangah Lee, Hansol Jang, Yunmee Baik, Suzi Park, and Hyopil Shin. 2020. Kr-bert: A small-scale korean-specific language model. *ArXiv*, abs/2008.03979.

Jihyung Moon, Won Ik Cho, and Junbum Lee. 2020. Beep! korean corpus of online news comments for toxic speech detection. In *Proceedings of the Eighth International Workshop on Natural Language Processing for Social Media*, pages 25–31.

Hamdy Mubarak, Kareem Darwish, and Walid Magdy. 2017. Abusive language detection on arabic social media. In *Proceedings of the first workshop on abusive language online*, pages 52–56.

Chikashi Nobata, Joel Tetreault, Achint Thomas, Yashar Mehdad, and Yi Chang. 2016. Abusive language detection in online user content. In *Proceedings of the 25th international conference on world wide web*, pages 145–153.

Nedjma Ousidhoum, Zizheng Lin, Hongming Zhang, Yangqiu Song, and Dit-Yan Yeung. 2019. Multilingual and multi-aspect hate speech analysis. In *Proceedings of the 2019 Conference on EMNLP and the 9th IJCNLP (EMNLP-IJCNLP)*, pages 4675–4684.

Jangwon Park. 2020. Koelectra: Pretrained electra model for korean. https://github.com/monologg/KoELECTRA.

S. Siegel and N.J. Castellan. 1988. *Nonparametric Statistics for the Behavioral Sciences*. McGraw-Hill.

SKTBrain. 2019. Kobert: Korean bert pre-trained cased. https://github.com/monologg/KoELECTRA.

Mohammad S Sorower. 2010. A literature survey on algorithms for multi-label learning. *Oregon State University, Corvallis*, 18:1–25.

Bertie Vidgen and Leon Derczynski. 2020. Directions in abusive language training data, a systematic review: Garbage in, garbage out. *Plos one*, 15(12):e0243300.

Kunze Wang, Dong Lu, Caren Han, Siqu Long, and Josiah Poon. 2020. Detect all abuse! toward universal abusive language detection models. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 6366–6376.

Zeerak Waseem and Dirk Hovy. 2016. Hateful symbols or hateful people? Predictive features for hate speech detection on Twitter. In *Proceedings of the NAACL student research workshop*, pages 88–93.

Henry Weld, Guanghao Huang, Jean Lee, Tongshu Zhang, Kunze Wang, Xinghong Guo, Siqu Long, Josiah Poon, and Caren Han. 2021. Conda: a contextual dual-annotated dataset for in-game toxicity understanding and detection. In *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*, pages 2406–2416.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Remi Louf, Morgan Funtowicz, Joe Davison, Sam Shleifer, Patrick von Platen, Clara Ma, Yacine Jernite, Julien Plu, Canwen Xu, Teven Le Scao, Sylvain Gugger, Mariama Drame, Quentin Lhoest, and Alexander Rush. 2020. Transformers:

State-of-the-art natural language processing. In *Proceedings of the 2020 Conference on EMNLP: System Demonstrations*, pages 38–45. Association for Computational Linguistics.

Ellery Wulczyn, Nithum Thain, and Lucas Dixon. 2017. Ex machina: Personal attacks seen at scale. In *Proceedings of the 26th international conference on world wide web*, pages 1391–1399.

Marcos Zampieri, Preslav Nakov, Sara Rosenthal, Pepa Atanasova, Georgi Karadzhov, Hamdy Mubarak, Leon Derczynski, Zeses Pitenis, and Çağrı Çöltekin. 2020. Semeval-2020 task 12: Multilingual offensive language identification in social media (offenseval 2020). In *Proceedings of the Fourteenth Workshop on Semantic Evaluation*, pages 1425–1447.

A Appendix

Ethics/Broader Impact Statement The study follows the ethical policy set out in the ACL code of Ethics[9] and addresses the ethical impact of presenting a new dataset. As described in the data section 2, our annotated dataset is based on the online news comments data publicly available on Kaggle and Github. All annotators were recruited from a crowdsourcing platform. They were informed about hate speech before handling the data. Our instructions allowed them to feel free to leave if they were uncomfortable with the content. With respect to the potential risks, we note that the subjectivity of human annotation would impact on the quality of the dataset.

The Korean language The Korean language is morphologically rich and the character structure is different to Latin-based language. A brief components used in the paper as follows:

- **Consonant (자음)** : A consonant is a sound such as 'p', 'f', 'n', or 't' which you pronounce by stopping the air flowing freely through your mouth.
  - initial consonant (초성)
  - bottom consonant (받침)

- **Vowel (모음)** : A vowel is a sound such as the ones represented in writing by the letters 'a', 'e', 'i', 'o', and 'u', which you pronounce with your mouth open, allowing the air to flow through it.

- **Syllable (음절)** : A syllable is a part of a word that contains a single vowel sound and that is pronounced as a unit. So, for example, 'book' has one syllable, and 'reading' has two syllables.
  - Korean romanization : (e.g. [kko#t#baem])
  - Character level : (e.g. 꽃#뱀)
  - Sub-character level : (e.g. 꼬#ㅊ#뱀)

Implementation Details For all baselines, we set the number of epochs as 4 and use a batch size of 32. For other hyper-parameters, we follow the configuration in the official GitHub implementation of the baselines. The source codes or pre-trained models for the baselines are available at the following GitHub addresses: Multilingual BERT[10], KoELECTRA[11], KoBERT[12] and KR-BERT[13].

Experiments. A brief of tables displayed in Appendix as follows:

- Table 8: a breakdown of multi-label classification performance from 1 to 4 labels;
- Table 9: overall binary classification performance;
- Table 10: a breakdown of binary classification performance;
- Table 10: F1 score for the top 10 three-label pairs in 3-labels classification;
- Table 12: F1 score for the top 5 four-label pairs in 4-labels classification.

[9] https://www.aclweb.org/portal/content/acl-code-ethics

[10] https://github.com/google-research/bert

[11] https://github.com/monologg/KoELECTRA

[12] https://github.com/SKTBrain/KoBERT

[13] https://github.com/snunlp/KR-BERT

# Labels	Model	F1 (Macro)	F1 (Micro)	F1 (Weighted)	E.M.	AUC	H.L. (↓)
1	BERT	0.6666	0.8190	0.8202	0.7919	0.9011	0.0406
	KoELECTRA	0.6953	0.8490	0.8508	**0.8263**	**0.9213**	0.0341
	KoBERT	0.7321	0.8320	0.8370	0.8142	0.9110	0.0379
	KR-BERT(w. char)	**0.7336**	**0.8553**	**0.8543**	0.8239	0.9145	**0.0318**
	KR-BERT(w. sub)	0.6985	0.8392	0.8419	0.8062	0.9123	0.0360
2	BERT	0.6389	0.8043	0.8174	0.5580	0.8524	0.0788
	KoELECTRA	0.6777	0.8612	0.8700	0.6511	0.8934	0.0577
	KoBERT	**0.7249**	**0.8854**	**0.8911**	**0.6794**	**0.9112**	**0.0482**
	KR-BERT(w. char)	0.6748	0.8405	0.8451	0.5912	0.8735	0.0642
	KR-BERT(w. sub)	0.6718	0.8703	0.8723	0.6535	0.9000	0.0542
3	BERT	0.5784	0.7517	0.7522	0.2448	0.8040	0.1402
	KoELECTRA	0.6146	0.7987	0.7953	0.3310	0.8362	0.1169
	KoBERT	**0.6523**	0.8290	0.8251	**0.3759**	0.8589	0.1019
	KR-BERT(w. char)	0.5828	0.7827	0.7732	0.2828	0.8239	0.1230
	KR-BERT(w. sub)	0.6164	**0.8329**	**0.8263**	0.3586	**0.8615**	**0.0996**
4	BERT	0.4776	0.7093	0.7029	**0.1200**	0.7610	0.2222
	KoELECTRA	0.4511	0.7044	0.6639	0.0000	0.7680	0.2089
	KoBERT	0.4177	0.6832	0.6460	0.0400	0.7510	0.2267
	KR-BERT(w. char)	0.4837	0.7439	0.7226	**0.1200**	0.7930	0.1867
	KR-BERT(w. sub)	**0.5068**	**0.7771**	**0.7618**	**0.1200**	**0.8120**	**0.1733**

Table 8: A breakdown of multi-label classification performance from 1 to 4 labels on K-MHaS for the five pre-trained language models at epoch 4 (E.M.:Exact Match, H.L.:Hamming Loss / KR-BERT (w. *): char = character-level, sub = sub-character-level)

Model	F1 (Macro)	F1 (Micro)	F1 (Weighted)	E.M.	AUC	H.L. (↓)
BERT	0.8495	0.8507	0.8505	0.8507	0.8488	0.1493
KoELECTRA	0.8756	0.8766	0.8765	0.8766	0.8750	0.1234
KoBERT	0.8687	0.8692	0.8693	0.8692	0.8696	0.1308
KR-BERT (w. char)	0.8846	0.8850	0.8851	0.8850	**0.8862**	0.1150
KR-BERT (w. sub)	**0.8869**	**0.8879**	**0.8877**	**0.8879**	0.8857	**0.1121**

Table 9: Overall binary classification performance on the K-MHaS dataset for the five pre-trained language models at epoch 4 (E.M.:Exact Match, H.L.:Hamming Loss / KR-BERT (w. *): char = character-level, sub = sub-character-level)

Label	Model	F1 (Macro)	F1 (Micro)	F1 (Weighted)	E.M.	H.L. (↓)
Hate Speech	BER	0.4518	0.8243	0.9037T	0.8243	0.1757
	KoELECTRA	0.4606	0.8540	0.9212	0.8540	0.1460
	KoBERT	0.4666	0.8746	**0.9331**	0.8746	0.1254
	KR-BERT (w. char)	0.4611	0.8558	0.7892	0.8558	0.1442
	KR-BERT (w. sub)	**0.4724**	**0.8953**	0.8458	**0.8953**	**0.1047**
None	BERT	0.4662	0.8733	0.9323	0.8733	0.1267
	KoELECTRA	0.4726	0.8960	**0.9452**	**0.8960**	0.1040
	KoBERT	0.4637	0.8645	0.9273	0.8645	0.1355
	KR-BERT (w. char)	**0.4772**	**0.9126**	0.8709	**0.9126**	**0.0874**
	KR-BERT (w. sub)	0.4687	0.8821	0.8268	0.8821	0.1179

Table 10: A breakdown of binary classification performance on the K-MHaS dataset for the five pre-trained language models at epoch 4 (E.M.:Exact Match, H.L.:Hamming Loss / KR-BERT (w. *): char = character-level, sub = sub-character-level, bi = Bidirectional WordPiece tokenizer)

Label Triplets	# triplets	F1 (macro)					F1 (micro)				
Overall Performance (F1)		**BERT**	**KoELECTRA**	**KoBERT**	**KR-BERT-c**	**KR-BERT-s**	**BERT**	**KoELECTRA**	**KoBERT**	**KR-BERT-c**	**KR-BERT-s**
		0.6912	0.7245	**0.7651**	0.7444	0.7245	0.8139	0.8493	0.8413	**0.8500**	0.8445
Origin & Politics & Profanity	41	0.2780	0.2935	0.2954	0.2937	**0.3125**	0.8224	0.8739	0.8869	0.8739	**0.9316**
Politics & Profanity & Age	37	0.2781	0.3054	**0.3174**	0.2981	0.2971	0.8205	0.9126	**0.9395**	0.8867	0.8824
Physical & Politics & Profanity	32	0.2483	0.2809	**0.2960**	0.2823	0.2939	0.7296	0.8304	0.8750	0.8421	**0.8764**
Origin & Profanity & Gender	30	0.2545	0.2314	0.2527	0.2463	**0.2886**	0.7467	0.7397	0.7368	0.7671	**0.8712**
Physical & Profanity & Gender	24	0.2151	0.2665	0.2730	0.2459	**0.2811**	0.6306	0.7869	0.8226	0.7521	**0.8413**
Origin & Physical & Profanity	14	0.2692	0.2811	**0.2873**	0.2351	0.2865	0.7532	0.8378	0.8312	0.7273	**0.8421**
Politics & Age & Gender	14	0.2593	**0.2933**	0.2830	0.2319	0.2406	0.7606	**0.8684**	0.8158	0.6970	0.7059
Profanity & Age & Gender	13	0.2686	0.2712	0.2932	0.2593	**0.2695**	0.8182	0.8060	**0.8732**	0.8000	0.8358
Origin & Physical & Gender	12	0.2692	0.2327	0.2407	0.2347	**0.2703**	0.7812	0.6780	0.7458	0.7143	**0.8065**
Origin & Age & Gender	10	0.2736	0.2781	**0.3093**	0.2428	0.2411	0.8235	0.8302	**0.9123**	0.7347	0.7600

Table 11: F1 score for the top 10 three-label pairs on the K-MHaS dataset for the five pre-trained language models at epoch 4 (# total label triplets = 290 / KR-BERT-*: c = character-level WordPiece tokenizer, s = sub-character-level WordPiece tokenizer)

Label Quadruplets	# quadruplets	F1 (macro)					F1 (micro)				
Overall Performance (F1)		**BERT**	**KoELECTRA**	**KoBERT**	**KR-BERT-c**	**KR-BERT-s**	**BERT**	**KoELECTRA**	**KoBERT**	**KR-BERT-c**	**KR-BERT-s**
		0.4776	0.4511	0.4177	0.4837	**0.5068**	0.7093	0.7044	0.6832	0.7439	**0.7771**
Origin & Profanity & Age & Gender	5	**0.3567**	0.2950	0.2346	0.2932	0.3086	**0.8000**	0.6875	0.5806	0.7500	0.7879
Origin & Physical & Profanity & Gender	4	0.2582	0.2804	0.2508	0.3302	**0.3757**	0.5385	0.7200	0.6400	0.7692	**0.8571**
Origin & Physical & Politics & Profanity	3	**0.4000**	0.3333	0.3111	0.3333	0.3667	**0.9091**	0.8000	0.7368	0.8000	0.8571
Origin & Politics & Profanity & Age	3	0.4222	0.3444	0.3667	**0.4444**	**0.4444**	0.8800	0.8000	0.8571	**1.0000**	0.9600
Origin & Politics & Profanity & Gender	2	0.1852	**0.2963**	0.2593	**0.2963**	**0.2963**	0.5455	**0.7692**	0.6667	0.6667	**0.7692**

Table 12: F1 score for the top 5 four-label pairs on the K-MHaS dataset for the five pre-trained language models at epoch 4 (# total label quadruplets = 25 / KR-BERT-*: c = character-level WordPiece tokenizer, s = sub-character-level WordPiece tokenizer)

Domain- and Task-Adaptation for VaccinChatNL, a Dutch COVID-19 FAQ Answering Corpus and Classification Model

Jeska Buhmann, Maxime De Bruyn, Ehsan Lotfi, and **Walter Daelemans**
CLiPS Research Center
University of Antwerp, Belgium
{jeska.buhmann,maxime.debruyn,ehsan.lotfi,walter.daelemans}@uantwerpen.be

Abstract

FAQs are important resources to find information. However, especially if a FAQ concerns many question-answer pairs, it can be a difficult and time-consuming job to find the answer you are looking for. A FAQ chatbot can ease this process by automatically retrieving the relevant answer to a user's question. We present VaccinChatNL, a Dutch FAQ corpus on the topic of COVID-19 vaccination. Starting with 50 question-answer pairs we built VaccinChat, a FAQ chatbot, which we used to gather more user questions that were also annotated with the appropriate or new answer classes. This iterative process of gathering user questions, annotating them, and retraining the model with the increased data set led to a corpus that now contains 12,883 user questions divided over 181 answers. We provide the first publicly available Dutch FAQ answering data set of this size with large groups of semantically equivalent human-paraphrased questions. Furthermore, our study shows that before fine-tuning a classifier, continued pre-training of Dutch language models with task- and/or domain-specific data improves classification results. In addition, we show that large groups of semantically similar questions are important for obtaining well-performing intent classification models.

1 Introduction

In quickly changing contexts, like the COVID-19 pandemic, getting access to relevant and correct information in a fast and easy way is of crucial importance. Although websites with frequently-asked-questions (FAQ) sections provide such information, finding the representative question that matches a user's request can be hard and time-consuming, especially when the list of question-answer (QA) pairs is long. A FAQ chatbot does this matching for the user, speeding up the process of finding an answer to the posed question: users speak or type their questions and the system retrieves the best matching answer available.

Intent: faq_ask_why
Waarom zou ik mij laten vaccineren? *Why would I get vaccinated?*
Wat zijn de voordelen? *What are the advantages?*
Waarom moet je je laten inenten? *Why do you need to get vaccinated?*
Ik snap niet waarom ik me moet laten vaccineren. *I don't understand why I must get vaccinated.*
Waarom is vaccineren belangrijk? *Why is vaccination important?*
Hoezo moet ik mij laten inenten tegen covid? *Why should I get vaccinated against covid?*
Waarom dringen ze zo aan op het vaccin? *Why are they so insistent on the vaccine?*
Intent: faq_ask_certificate
Krijgen we een attest na vaccinatie? *Do we get a certificate after vaccination?*
Moet ik later kunnen bevestigen dat ik een vaccinatie gehad heb? *Do I have to be able to confirm later that I have had a vaccination?*
Kunnen instanties me vragen te bewijzen of ik gevaccineerd ben? *Can authorities ask me to prove that I have been vaccinated?*

Table 1: Examples of paraphrases for two intents in the corpus.

Standard FAQs consist of a list of QA pairs, where each answer is coupled with one instance of a general question. However, to train an automatic classifier of answers, more than just one example of questions is needed. Different paraphrases or versions of the same question will improve the classification performance. In this work we describe the data collection process for VaccinChat[1],

[1] https://vaccinchat.be

Proceedings of the 29th International Conference on Computational Linguistics, pages 3539–3549
October 12–17, 2022.

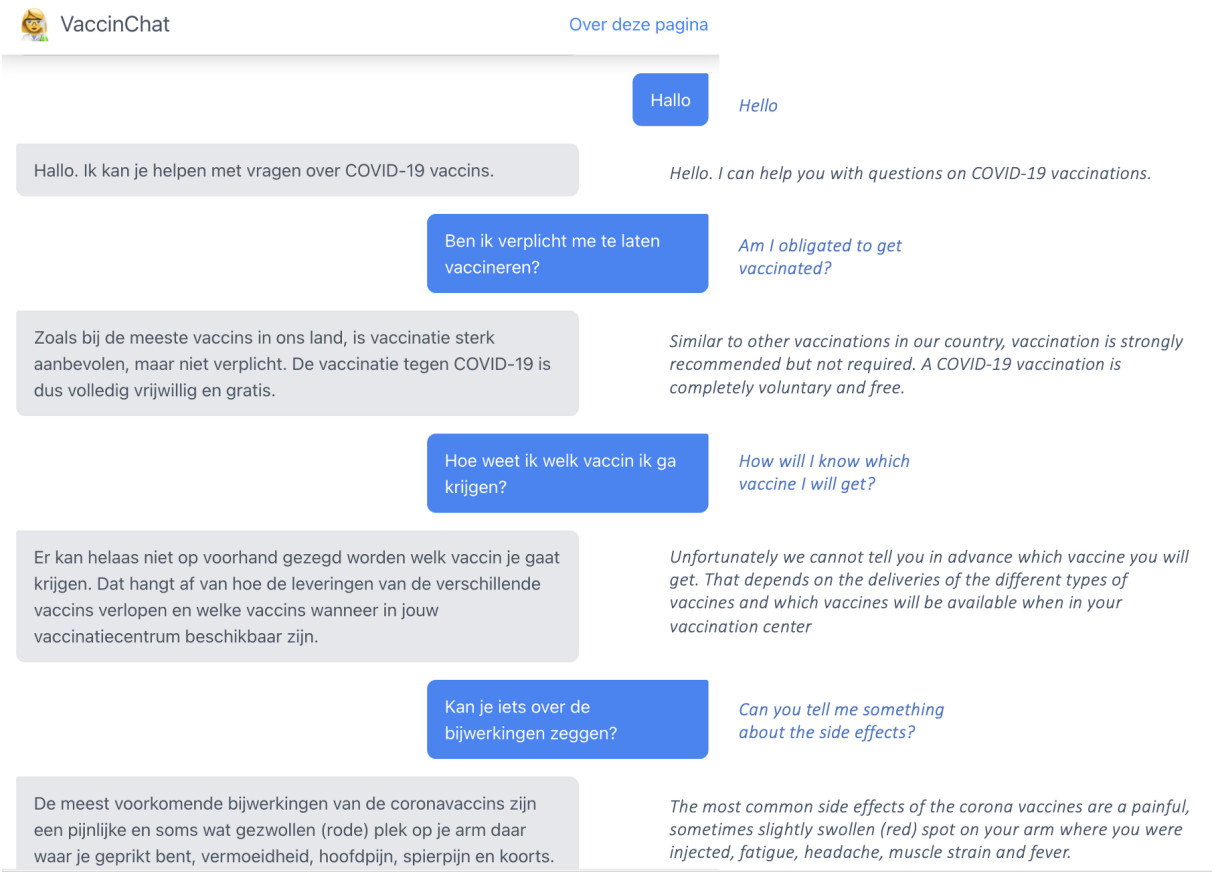

Figure 1: Example dialogue in VaccinChat.

a Dutch FAQ chatbot on the COVID-19 vaccines. The result is VaccinChatNL, the first Dutch FAQ answering data set, containing 12,883 questions that refer to 181 different answers. With Dutch we refer to the Dutch used in the Flemish region of Belgium as opposed to the Dutch used in the Netherlands. Table 1 shows some examples of user questions for two different answers and Figure 1 shows an example dialogue in VaccinChat.

Commercial organizations that use or develop FAQ chatbots, usually keep their FAQ answering data sets private. However, VaccinChat was built for research purposes and the released corpus was checked and anonymized with respect to privacy-sensitive information. The number of FAQ answers (181) is a lot higher than on the average FAQ page on the web holding only six answers (De Bruyn et al., 2021b). In addition, the corpus does not just hold QA pairs, but groups of questions per answer, providing a unique set of many-to-one mappings of questions to answers. Unlike FAQs on the web with 'clean', grammatically correct questions, the questions in our corpus come from actual users and may contain typing errors and/or other omissions/mistakes.

Questions referring to the same answer are not always paraphrases (e.g., "Should we get a vaccine on a yearly basis?" and "How long will the vaccine protect me?": both refer to the same answer), but in a lot of cases they are (see the examples in Table 1). So, although VaccinChatNL is not a paraphrase corpus in the strictest sense of the word, it does contain a lot of paraphrases. And unlike most of the available paraphrase corpora that contain paraphrase pairs, VaccinChatNL contains paraphrase groups that have many more than two paraphrases.

Our FAQ chatbot uses an intent classifier, meaning that user questions are classified into an answer class (intent), after which the answer of the classified intent is returned to the user. To train such a classifier we need the user questions and their intent labels. Since we also have the text of the answers, the VaccinChatNL corpus could be used to train a model that considers the similarity between (the representations of) a question and its answer. ConveRT (Henderson et al., 2019) is an example of such a model for English, and a Dutch version was developed by De Bruyn et al. (2021a).

On top of the above-mentioned qualities of the corpus, the topic of the corpus – COVID-19 vaccines – makes VaccinChatNL relevant. Although the data provides a blueprint of questions and answers for the Flemish situation in the first half of 2021, the topic is relevant and similar for the whole world. Translated data could be used for training models in other languages. Since Dutch (FAQ) data is scarce, the corpus is an excellent addition to multilingual FAQ approaches and data sets.

Despite the multipurpose nature of the corpus, we focus on using the data for training an intent classifier as one of the possible applications for the COVID-19 FAQ domain. Rather than finding the best possible model for this purpose, we focus on inspecting the effect of pre-training with task- and/or domain-specific data. In this work we describe those experiments as well as the data collection process. The VaccinChatNL corpus is publicly available[2] on the HuggingFace dataset hub (Lhoest et al., 2021).

2 Related Work

This section contains related work concerning domain- and task-adaptation, and considering the overlap of VaccinChatNL with FAQ and paraphrase corpora, we review the existing literature on these types of data resources.

2.1 Domain- and Task-Adaptation

Since large labeled data sets for classification are not always easy to build, pre-trained language models are used as a starting point for fine-tuning on the classification task. Edwards et al. (2020) show the importance of using domain-specific data for further unlabeled pre-training of a language model, improving performance for sentiment and emoji classification (among other things) of twitter utterances. Similarly, Wiedemann et al. (2020) show the benefit of domain-adaptation of a RoBERTa-large model for binary offensive language detection.

Zhu et al. (2021) found that domain-adaptation is mainly beneficial in low-resource settings. Increasing the amount of labeled data available for fine-tuning, reduces the impact of further pre-training with domain-specific data. Similar to the work of Mehri et al. (2020) they also investigated pre-training with task-specific data (i.e., data for the downstream task) and concluded as well that the impact of both domain- and task-adaptation depends on the type of task, the model, and the amount of data for fine-tuning. We therefore tested the impact of domain- and task-adaptation for our particular case, i.e., the size of our training set in combination with the BERT- (Devlin et al., 2019) and RoBERTa-based (Liu et al., 2019) text classification models we used (see section 4.2).

In this paper we not only analyse the impact of pre-training with task- and/or domain-specific data on the overall classification performance, but we also investigate the linguistic effects by doing an error analysis of how different intent types (FAQ, chitchat, and out-of-domain data) are affected.

2.2 FAQ and Paraphrase Data Sets

Faq-Finder (Hammond et al., 1995) created FAQs on multiple topics based on an English data set collected from Usenet news groups. The FAQIR dataset (Karan and Šnajder, 2016) contains 1,233 English queries retrieved from the "maintenance & repairs" section of the website *Yahoo! Answers*. A data set of similar size is StackFAQ (Karan and Šnajder, 2018), containing 1,249 user queries from the *StackExchange* domain. LocalGov is a Japanese FAQ corpus with 784 user queries in the domain of government (Sakata et al., 2019).

In the domain of question answering the semantic similarity between questions is relevant. The Kaggle Quora Question Pairs corpus[3] contains over 400,000 English question pairs labeled for semantic similarity.

Marsi and Krahmer (2014) developed a large Dutch aligned treebank corpus to study semantic similarity. It covers various text genres, such as texts from books, auto-cue subtitle pairs, news headlines, press releases about news events, and a section with QA-system output. This last part concerns a QA-system in the medical domain (van den Bosch and Bouma, 2011), that searched answers to questions in e.g., medical encyclopedia and layman websites. It resembles a FAQ answering data set, but instead of just QA pairs, it contains 100 questions that on average each have two answers (almost 200 answers in total). Although the complete corpus is large and consists of over 2M tokens, the size of the QA section is small and does not include variations or paraphrases of the questions.

The Dutch part of the multilingual TaPaCo

[2] https://huggingface.co/datasets/clips/VaccinChatNL

[3] https://quoradata.quora.com/First-Quora-Dataset-Release-Question-Pairs

corpus (Scherrer, 2020) contains sentential paraphrases: 23,561 paraphrases divided over 9,441 semantically similar paraphrase sets. This means that a group of similar semantics on average has 2.5 paraphrases. Because we collected user queries via our VaccinChat chatbot, the average number of queries per answer (including paraphrases) is a lot larger.

More specifically in the domain of COVID-19 Zhang et al. (2020) recently released COUGH, a multilingual FAQ retrieval dataset. A total of ∼16k FAQ items were scraped from 55 authoritative institutional websites, covering a wide range of topics on COVID-19, from general virus information to specific COVID-related instructions for a healthy diet. Almost 7k FAQ items were scraped from non-English FAQ sources. Besides FAQ items (QA pairs), the corpus also contains a Query Bank with 1,236 human-paraphrased user queries (three human paraphrases per given FAQ question), be it only for the English part of the corpus. Our VaccinChatNL corpus is similar to COUGH in the sense that both are FAQ data sets and the topic is similar, although COUGH covers a wider range of COVID-19 issues than VaccinChatNL, that narrows the focus on vaccines and vaccination. Unfortunately, the Dutch part of COUGH only consists of 142 QA pairs and no human-paraphrased versions of questions. VaccinChatNL by comparison consists of 12,883 questions referring to 181 different answers, so each answer on average has 71 different versions of questions referring to it. Also note that in VaccinChatNL the questions are actual user questions and not paraphrased versions of a set of given questions (as is the case for the English part of COUGH).

3 Dutch VaccinChatNL Corpus

In this section, we present our new Dutch VaccinChatNL corpus, a FAQ answering data set with many-to-one mappings of user questions to FAQ answers.

3.1 RASA Chatbot as a Data Collection Tool

Building a chatbot in RASA (Bocklisch et al., 2017) is an iterative, conversation driven, development process. This implies that the chatbot is continuously improved by (semi-automatically) annotating new user conversations or by adding new elements not yet present in the chatbot, after which the model is retrained. Because of this cyclic, user-driven way of development and an easy user interface, we chose RASA for building our chatbot, while simultaneously collecting user data.

3.2 Data Set Collection

3.2.1 Cold Start

We started with collecting approximately fifty FAQ pairs from the Belgian governmental webpage on vaccination[4]. Some of the answers were mildly adapted to better suit the conversational domain, i.e., sometimes answers were shortened to get the message across faster and we made sure none of the answers started with "yes" or "no", making them applicable not only to yes/no questions but also to semantically similar questions or utterances.

Together with a fallback option, a start message, and a few basic chitchat intents (e.g., *chitchat_ask_bye* and *chitchat_ask_thanks*), these FAQ pairs were entered in a RASA environment, and a first version of our FAQ chatbot was trained.

3.2.2 Improvement Phase

In the second data collection phase we grew the corpus by having a small group of users test the chatbot. The user questions were annotated with the relevant answer classes. In addition, we performed error analysis on a 20% held-out data set to find out which user questions could benefit from adding more paraphrases. This iterative process also revealed missing information, leading to updates of already existing answers and the addition of new answers. In May 2021 VaccinChat was opened to the public generating a surge of user questions that had to be annotated. The growth of the corpus was maintained until July 2021, resulting in 181 different answers and 11,650 user questions in total.

3.2.3 Testing Phase

In addition to the above described data collection effort, another 1,267 user queries were collected and annotated for an ongoing study on the comparison of different versions of the chatbot (Poels et al., 2021). Participants in the study had no prior knowledge about the chatbot and were instructed to use the chatbot for five minutes by typing in questions they might have on COVID-19 vaccination. Before and after the chatting phase, they were given questionnaires about vaccination-related topics. The post-questionnaire also informed about the chatbot's user-interface aspects.

[4] https://www.info-coronavirus.be/nl/vaccinatie

3.3 Data Annotation

As described in section 3.2, the data collection and manual annotation of the user queries happened incrementally. The complete VaccinChatNL data set was checked and corrected by three annotators (without overlap), starting from the RASA predictions. These annotators were computational linguists.

To assess the consistency in assigning the relevant answer classes and measure Inter-Annotator-Agreement, we, afterwards, had four different Dutch-speaking persons annotate a small subset (62 randomly selected user queries) of our complete annotated data set. Two of the annotators had a computational linguistics profile (one was not an author of the paper), and the other two had an end-user profile.

All four of the annotators were instructed to first get acquainted with the type of answers available by going over the list of possible intent names. They were also told to use the label *nlu_fallback* if the question was incomprehensible or completely irrelevant (e.g., an insult), and to use the label *faq_ask_general_information* if the question was COVID-19 related but not concerning vaccination. The raters were provided a document with all the intent names and answers, and were instructed to use this document to search for keywords related to the user query. Fleiss' kappa showed that there was good agreement between the raters' judgements, $\kappa = .663$.

3.4 Ethics and Privacy

Users of VaccinChat were informed that the chatbot could give irrelevant answers to their questions, and that their conversations would be used to further improve the performance of the chatbot.

Afterwards, the collection of data was checked for privacy-sensitive information, such as names and telephone numbers. Two user queries with names were removed, as well as three with telephone numbers. We also removed 28 entries containing codes that were used in the Testing phase (section 3.2.3) to link conversations to user questionnaires, and we removed three entries containing a URL.

Although we checked for other metadata that could reveal personal information (age, domicile, gender, disease, job), such entries were not deleted, because they could not be traced back to a specific person; in our corpus the combinations of metadata never revealed a specific identity, e.g, "persons aged 71 living in Antwerp" is not specific enough to characterize an individual person.

In addition, all annotated user queries are stored as separate QA pairs that no longer link back to their original user conversations. This means that information can not be combined with user information that was revealed somewhere else in that respective conversation.

3.5 Data Set Statistics

The final VaccinChatNL corpus consists of 12,883 user queries referring to 181 different answers. On average each answer class has 71 (*SD* = 92; *Mdn* = 47) user queries. The class with the highest number of user queries is the fallback class, also referred to as the unanswerable questions. The corpus contains 903 of such examples, representing 7.01% of the total number of user questions. The second biggest paraphrase group contains the user queries referring to the general side effects of the COVID-19 vaccines (*faq_ask_general_side_effects*). This group contains 416 paraphrases (3.23%).

3.6 Evolution over Time

During the COVID-19 pandemic the stream of information was constantly changing. When we started developing VaccinChat, vaccines were becoming available, and the vaccination campaign just began. One of the things people were asking questions about at that time was when they would be invited for vaccination and whether or not they belonged to a group that would get priority (answer class: *faq_ask_priority_groups*). A little later, people were most concerned with how long the vaccines would protect them against COVID-19 (answer class: *faq_ask_duration_of_protection*) and e.g., the blood clots side effects of the AstraZeneca vaccine (answer class: *faq_ask_astrazeneca_blood_clots*). Towards the summer of 2021, people started informing about their chances of going on holiday while also having to be available for their second vaccination (answer class: *faq_ask_holidays*)[5].

4 Experiments

In this section we describe a number of different classification models, built with VaccinChatNL data.

[5] A data visualisation of this evolution can be found at https://public.flourish.studio/visualisation/6517886/

	Train		Dev		Test	
Size (total 12,883)	10,542		1,171		1,170	
Intents	**#**	**%**	**#**	**%**	**#**	**%**
nlu_fallback	740	7.02	82	7.00	81	6.92
faq_ask_general_side_effects	303	2.87	56	4.78	57	4.87
faq_ask_priority_groups	291	2.76	46	3.93	45	3.85
faq_ask_protection_rate	225	2.13	34	2.90	35	2.99
faq_ask_general_information	200	1.90	55	4.70	55	4.70
faq_ask_no_risk_patient	200	1.90	18	1.54	18	1.54
faq_ask_astrazeneca_blood_clots	189	1.79	22	1.88	21	1.79
faq_ask_contra_indication	189	1.79	15	1.28	15	1.28
faq_ask_holidays	188	1.78	21	1.79	23	1.97
faq_ask_no_answer	174	1.65	36	3.07	35	2.99
faq_ask_duration_of_protection	153	1.45	24	2.05	24	2.05
faq_ask_trustworthy	137	1.30	27	2.31	28	2.39
faq_ask_choice	131	1.24	26	2.22	25	2.14
faq_ask_why	127	1.20	13	1.11	11	0.94
faq_ask_when_and_why	126	1.20	15	1.28	14	1.20

Table 2: Summary of VaccinChatNL data in terms of number of user questions per train, development and test set. The bottom part shows the absolute numbers (#) and percentages (%) of user questions for the 15 most frequent FAQ intents.

4.1 Train, Development, and Test Sets

To present the models' performance on unseen data, the corpus was split up in a train, a development, and a test set. The collected data in the Testing phase was mixed together with a random 10% of the data collected in the Improvement Phase. The merged data were split into a development and test set. The remaining 90% of the Improvement phase data was kept for training. Statistics of these sets are presented in Table 2, which also shows the absolute numbers and percentages of the fifteen most frequent intents in the corpus. The most frequent intent is the fallback option. This includes out-of-domain questions, insults, jokes, and questions that are incomprehensible because they contain either too many typing mistakes or only keywords that could refer to multiple answers.

4.2 Baseline Models

Here we present a number of baseline models for our VaccinChatNL corpus. We start with a simple majority class baseline and the RASA baseline classifier. In addition, we use pre-trained Dutch language models: BERTje (de Vries et al., 2019) and CoNTACT (Lemmens et al., 2022), based on BERT (Devlin et al., 2019) and RoBERTa (Liu et al., 2019), respectively.

4.2.1 Majority

The majority baseline always gives the same, most frequent answer. It thus always returns "*Ik begrijp het niet. Kunt u het anders zeggen?*" ("I don't understand. Could you rephrase the question?"). This is the fallback for unanswerable questions.

4.2.2 RASA DIET Classifier

RASA (Bocklisch et al., 2017) uses a multitask transformer architecture for NLU: Dual Intent and Entity Transformer (DIET)[6]. This DIET classifier can deal with both intent classification and entity recognition simultaneously. Only intent classification applies to our chatbot.

4.2.3 BERTje

BERTje is a Dutch pre-trained BERT model (Devlin et al., 2019) developed by de Vries et al. (2019). Several high-quality level Dutch corpora were used for pre-training, including collections of books, news articles, and Wikipedia documents.

4.2.4 BERTje+

This is the BERTje model, but further pre-trained with a masked-language learning (MLM) approach

[6] https://rasa.com/blog/introducing-dual-intent-and-entity-transformer-diet-state-of-the-art-performance-on-a-lightweight-architecture/

Number of train items		Test set		
Per class	**Total**	**P**	**R**	**F1**
1	181	0.01 (0.005)	0.01 (0.004)	0.01 (0.004)
2	351	0.02 (0.010)	0.02 (0.009)	0.02 (0.009)
4	679	0.07 (0.011)	0.07 (0.011)	0.07 (0.011)
8	1,331	0.20 (0.018)	0.19 (0.017)	0.20 (0.017)
10	1,655	0.23 (0.021)	0.23 (0.021)	0.23 (0.021)
30	4,648	0.41 (0.013)	0.41 (0.012)	0.41 (0.013)
100	8,591	0.63 (0.008)	0.63 (0.007)	0.63 (0.008)
300	10,099	0.65 (0.006)	0.65 (0.005)	0.65 (0.006)

Table 3: The effect of the training set size (number of train items per class) on test set performance: average Precision (P), Recall (R) and F1 from 10-fold cross-validations. Standard deviation is shown between brackets.

on task-specific data, i.e., the VaccinChatNL user queries excluding the *nlu_fallback* class. This model was used to show the effect of task adaptation on the classification task.

4.2.5 CoNTACT

CoNTACT[7] (Lemmens et al., 2022) is a Dutch RoBERTa-based language model, adapted to the domain of COVID-19 tweets by extra pre-training with this data of the Dutch RobBERT model (Delobelle et al., 2020). It was used to show the effect of domain adaptation on the classification task.

4.2.6 CoNTACT+

Similar to BERTje+, this is the CoNTACT model with extra MLM pre-training on the VaccinChatNL user queries. It shows the effect of domain- and task-adaptation simultaneously.

4.3 Train Settings

For the data set size experiments (see 5.1) the RASA DIET Classifier was trained for 12 epochs. This was based on initial experiments where training for more epochs revealed a maximum (accuracy) performance on the development set at 12 epochs.

For the domain- and task-adaptation experiments (see 5.2) hyper-parameter search revealed optimal results on the development set with a maximum sequence length of 64, a batch size of 16, learning rate of 2e-5, and warm-up ratio of 0.1. The BERTje models were trained for 7 epochs and a weight decay of 0.01, and the CoNTACT models for 10 epochs, with weight decay of 0.1. In all cases early stopping was applied based on the development set performance. This resulted in stopping after 7 epochs for BERTje, after 5 epochs for BERTje+, and after 6 epochs for CoNTACT and CoNTACT+.

[7] https://huggingface.co/clips/contact

5 Results

5.1 Effect of Data Set Size

To show the importance of having enough paraphrases per class to train a sentence classifier, we experimented with training the RASA baseline model (DIET classifier) with different numbers of training examples per class. Table 3 shows the average results (precision, recall, and F1) of 10-fold cross-validation experiments on the test set. Each model was trained for 12 epochs.

Note that the mentioned numbers of training examples per class only apply to the classes that have that many training examples. If a class has fewer items than the stated number, all training items for that class are used. The second column in the table shows the total number of training examples.

Results clearly show that in order to get a near-optimal performance on the test set, the number of training examples per class should at least be 100.

5.2 Effect of Models

Although the majority class (the fallback for unanswerable questions) has a high number of occurrences in the corpus (903), this class still only represents 7.00% and 6.92% of the data in the development and test set respectively. This means that the majority baseline model gives an accuracy of 6.9% on the test set.

Best results with the RASA DIET classifier on the development set were obtained with 12 epochs of training. We get a test set accuracy of 64.7%, which is a big improvement over the majority baseline.

Model	Test set accuracy
Majority	6.9
RASA DIET classifier	64.7
BERTje	74.7
BERTje+	77.7
CoNTACT	77.1
CoNTACT+	77.9

Table 4: VaccinChatNL accuracy scores (% correct) with different models. +: models with extra MLM pre-training on task-specific data, i.e., the VaccinChatNL user queries from the train set, excluding the out-of-domain ones labelled as *nlu_fallback*. Results for these models are better than for the models without task-specific pre-training. Overall, CoNTACT+, the model with domain- as well as task-adaptation, performs best.

All other models - BERTje(+) and CoNTACT(+) - were Dutch pre-trained language models fine-tuned on a sentence classification task with the VaccinChatNL train data set. The results of all these models are shown in Table 4. All BERTje and CoNTACT models are a clear improvement over the RASA DIET classifier, showing the importance of using language-specific (i.e., Dutch) pre-trained language models.

Table 5 provides more specifics on the misclassifications of the above models per answer type (fallback answers, chitchat, and FAQ answers). In general, all models with extra task- and/or domain-specific pre-training perform better than BERTje, which had no such continued pre-training. In the following section we show the effect of domain- and task-adaptation on the classification performance by means of an error analysis.

6 Discussion

When taking a more in-depth look into the type of errors made by the BERTje model and its domain- and task-adapted versions, a main observation is that all models make errors for user questions with infrequent words, like place names (e.g., "Kessel-Lo", "rotselaar") or typos (see Table 6).

With respect to the fallback and chitchat intents, the results show a better performance for the CoNTACT models than for the BERTje models (see Table 5). This suggests a benefit of pre-training with domain-specific data, in terms of a better recognition of the non-domain-specific user questions. Some examples of chitchat intents that are correctly classified with the CoNTACT models but misclassified with the BERTje models are shown in Table 7.

We also see a decrease in FAQ errors with domain-specific pre-training, but only for the case where no task-specific pre-training was done (CoNTACT vs. BERTje). For CoNTACT+, the model that was pre-trained on both domain- and task-specific data, we only see FAQ improvements compared to CoNTACT, but not compared to BERTje+. Unlike the FAQ intents, fallback and chitchat intents are better recognized with CoNTACT+ than with BERTje+, but there is no difference with CoNTACT. In general, CoNTACT+ shows the best performance.

For task-specific pre-training a clear improvement was observed for user questions containing domain terminology, i.e., words like *KU Leuven*, *Facebook*, *SARS-COV-2*, *Kovid 19*, *Jnj*, *QVAX*, etc. This was to be expected, since these words were actually in the task-specific data.

In summary, we can conclude that task-specific pre-training improves FAQ intent classification, whereas domain-adaptation mainly benefits the fallback and chitchat intent classification, i.e., the out-of-domain user questions.

7 Conclusions and Future Work

We have presented VaccinChatNL, the first Dutch FAQ answering corpus with over 12k of user queries about COVID-19 vaccines, and on average 71 example questions per answer. As the corpus holds lots of paraphrased versions of questions, it can serve as a paraphrase corpus as well. Existing Dutch paraphrase corpora, or Dutch parts of multilingual corpora are sometimes bigger, but do not contain sentential paraphrases (less long phrases or single words), or have less paraphrases per semantically similar group (typically paraphrase pairs). The many-to-one mapping characteristic of this corpus is unique in FAQ and paraphrasing corpora.

As an example application of VaccinChatNL, we described training an intent classifier and we have shown that for a classifier with many classes (i.e., different answers), performance improves with more examples of user questions per class.

This work also showed the importance of using domain- or task-adapted pre-trained language models for the fine-tuning task of sentence classification. More specifically, domain-adaptation (COVID-related tweets) led to improvements for the non-FAQ questions, whereas task-adaptation improved performance for FAQ questions with domain-specific terminology.

Nr. of errors	BERTje	BERTje+	CoNTACT	CoNTACT+
Total	299	263	271	261
Fallback intent	36	37	29	31
Chitchat intent	18	18	14	14
FAQ intent	245	208	228	216

Table 5: Number of errors for the test set with different models, split up per intent type: fallback, chitchat, and FAQ. The table shows the positive impact of task-specific pre-training (+models) on FAQ questions, and the benefit of domain-adaptation on mainly chitchat and fallback utterances: CoNTACT(+) vs. BERTje(+). Overall, CoNTACT+ is the best performing model.

User question	True label	Predicted label
Mag ik na het vaccin terug **reizne** *After the vaccine can I **travle** again*	faq_ask_holidays	faq_ask_what_after_vaccination
Moey ik mij laten vaccineren ***Doy** I have to get vaccinated*	faq_ask_obligatory	nlu_fallback
Hoe word ik **vrijwillgier**? *How do I become a **voluneter**?*	faq_ask_volunteer	faq_ask_why_and_when
worden gezonde mensen **zien** van corona *Do healthy people get **see** from corona*	faq_ask_why	nlu_fallback

Table 6: Dutch examples of misclassification made by all models: user questions with infrequent words like typos. Translated utterances in italics and typos in bold.

User question	True label	Predicted label (BERTje)
nou doeeiiii *Well byeee*	chitchat_ask_bye	nlu_fallback
DAg *BYe*	chitchat_ask_bye	nlu_fallback
ok ben niet neig overtuig maar dank u wel *okay not real convinced but thank you*	chitchat_ask_thanks	faq_ask_no_answer
Oké. Alles is duidelijk. *Okay. Everything is clear.*	chitchat_ask_thanks	chitchat_ask_bye

Table 7: Dutch examples of misclassification in the chitchat intent class made by the BERTje models, but not with the CoNTACT models. Translated utterances in italics.

Although our focus has been on classification, it would be interesting to use a retrieval approach for FAQ answer selection, because there is a certain amount of overlap in the answers. A predicted answer with a different label than what was annotated, is not necessarily wrong. It could still include information that is relevant to the user. In addition, the VaccinChatNL corpus would be a good source for e.g., the purpose of training generative paraphrase models. Instead of just sentence pairs, the corpus provides many examples that have similar semantic meaning relevant for the corresponding answers.

Acknowledgements

We thank the reviewers for providing constructive feedback. In addition, we thank the involved annotators for contributing to this work. This research received funding from the Flemish Government under the "Onderzoeksprogramma Artificiële Intelligentie (AI) Vlaanderen" programme.

References

Tom Bocklisch, Joey Faulkner, Nick Pawlowski, and Alan Nichol. 2017. Rasa: Open source language understanding and dialogue management. *Computing Research Repository*, arXiv:1712.05181. Version 2.

Maxime De Bruyn, Ehsan Lotfi, Jeska Buhmann, and Walter Daelemans. 2021a. ConveRT for

FAQ answering. *Computing Research Repository*, arXiv:2108.00719. Version 3.

Maxime De Bruyn, Ehsan Lotfi, Jeska Buhmann, and Walter Daelemans. 2021b. MFAQ: a multilingual FAQ dataset. In *Proceedings of the 3rd Workshop on Machine Reading for Question Answering*, pages 1–13, Punta Cana, Dominican Republic. Association for Computational Linguistics.

Wietse de Vries, Andreas van Cranenburgh, Arianna Bisazza, Tommaso Caselli, Gertjan van Noord, and Malvina Nissim. 2019. BERTje: A Dutch BERT model. *Computing Research Repository*, arXiv:1912.09582.

Pieter Delobelle, Thomas Winters, and Bettina Berendt. 2020. RobBERT: a Dutch RoBERTa-based language model. *Computing Research Repository*, arXiv:2001.06286. Version 2.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, Minneapolis, Minnesota. Association for Computational Linguistics.

Aleksandra Edwards, Jose Camacho-Collados, Hélène De Ribaupierre, and Alun Preece. 2020. Go simple and pre-train on domain-specific corpora: On the role of training data for text classification. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 5522–5529, Barcelona, Spain (Online). International Committee on Computational Linguistics.

Kristian Hammond, Robin Burke, Charles Martin, and Steven Lytinen. 1995. Faq finder: a case-based approach to knowledge navigation. In *Proceedings the 11th Conference on Artificial Intelligence for Applications*, pages 80–86. IEEE.

Matthew Henderson, Iñigo Casanueva, Nikola Mrkšić, Pei-Hao Su, Tsung-Hsien Wen, and Ivan Vulić. 2019. ConveRT: Efficient and accurate conversational representations from transformers. *Computing Research Repository*, arXiv:1911.03688. Version 2.

Mladen Karan and Jan Šnajder. 2016. Faqir – a frequently asked questions retrieval test collection. In *Text, Speech, and Dialogue*, pages 74–81, Cham. Springer International Publishing.

Mladen Karan and Jan Šnajder. 2018. Paraphrase-focused learning to rank for domain-specific frequently asked questions retrieval. *Expert Systems with Applications*, 91:418–433.

Jens Lemmens, Jens Van Nooten, Tim Kreutz, and Walter Daelemans. 2022. CoNTACT: A Dutch COVID-19 adapted BERT for vaccine hesitancy and argumentation detection. *Computing Research Repository*, arXiv:2203.07362.

Quentin Lhoest, Albert Villanova del Moral, Yacine Jernite, Abhishek Thakur, Patrick von Platen, Suraj Patil, Julien Chaumond, Mariama Drame, Julien Plu, Lewis Tunstall, Joe Davison, Mario Šaško, Gunjan Chhablani, Bhavitvya Malik, Simon Brandeis, Teven Le Scao, Victor Sanh, Canwen Xu, Nicolas Patry, Angelina McMillan-Major, Philipp Schmid, Sylvain Gugger, Clément Delangue, Théo Matussière, Lysandre Debut, Stas Bekman, Pierric Cistac, Thibault Goehringer, Victor Mustar, François Lagunas, Alexander Rush, and Thomas Wolf. 2021. Datasets: A community library for natural language processing. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 175–184, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. RoBERTa: A robustly optimized BERT pretraining approach. *Computing Research Repository*, arXiv:1907.11692.

Erwin Marsi and Emiel Krahmer. 2014. Construction of an aligned monolingual treebank for studying semantic similarity. *Language Resources and Evaluation*, 48(2):279–306.

Shikib Mehri, Mihail Eric, and Dilek Hakkani-Tur. 2020. DialoGLUE: A natural language understanding benchmark for task-oriented dialogue. *Computing Research Repository*, arXiv:2009.13570. Version 2.

Karolien Poels, Toni Claessens, Jeska Buhmann, Maxime De Bruyn, Heidi Vandebosch, Pierre Van Damme, and Walter Daelemans. 2021. A chatbot as an intervention to foster public engagement with the covid-19 vaccines. In *Book of Abstracts of European Conference on Health Communication*, pages 89–90.

Wataru Sakata, Tomohide Shibata, Ribeka Tanaka, and Sadao Kurohashi. 2019. Faq retrieval using query-question similarity and bert-based query-answer relevance. In *Proceedings of the 42nd International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR'19, pages 1113–1116, New York, NY, USA. Association for Computing Machinery.

Yves Scherrer. 2020. TaPaCo: A Corpus of Sentential Paraphrases for 73 Languages . In *Proceedings of The 12th Language Resources and Evaluation Conference*, pages OP –. European Language Resources Association (ELRA).

Antal van den Bosch and Gosse Bouma. 2011. *Interactive multi-modal question-answering*. Springer Science & Business Media.

Gregor Wiedemann, Seid Muhie Yimam, and Chris Biemann. 2020. UHH-LT at SemEval-2020 task 12:

Fine-tuning of pre-trained transformer networks for offensive language detection. In *Proceedings of the Fourteenth Workshop on Semantic Evaluation*, pages 1638–1644, Barcelona (online). International Committee for Computational Linguistics.

Xinliang Frederick Zhang, Heming Sun, Xiang Yue, Simon Lin, and Huan Sun. 2020. COUGH: A challenge dataset and models for COVID-19 FAQ retrieval. *Computing Research Repository*, arXiv:2010.12800. Version 2.

Qi Zhu, Yuxian Gu, Lingxiao Luo, Bing Li, Cheng Li, Wei Peng, Minlie Huang, and Xiaoyan Zhu. 2021. When does further pre-training MLM help? an empirical study on task-oriented dialog pre-training. In *Proceedings of the Second Workshop on Insights from Negative Results in NLP*, pages 54–61, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Benchmarking Automated Clinical Language Simplification: Dataset, Algorithm, and Evaluation

Junyu Luo[1], **Junxian Lin**[2,3], **Chi Lin**[2,3], **Cao Xiao**[4], **Xinning Gui**[1], **Fenglong Ma**[1*]
[1]College of Information Sciences and Technology, Pennsylvania State University, USA
[2]School of Software Technology, Dalian University of Technology, China
[3]Key Laboratory for Ubiquitous Network and Service Software of Liaoning Province, China
[4]Relativity, USA
{junyu, xinninggui, fenglong}@psu.edu, linjunxian@mail.dlut.edu.cn,
c.lin@dlut.edu.cn, cao.xiao@relativity.com

Abstract

Patients with low health literacy usually have difficulty understanding medical jargon and the complex structure of professional medical language. Although some studies are proposed to automatically translate expert language into layperson-understandable language, only a few of them focus on both accuracy and readability aspects simultaneously in the clinical domain. Thus, simplification of the clinical language is still a challenging task, but unfortunately, it is not yet fully addressed in previous work. To benchmark this task, we construct a new dataset named MedLane to support the development and evaluation of automated clinical language simplification approaches. Besides, we propose a new model called DECLARE that follows the human annotation procedure and achieves state-of-the-art performance compared with eight strong baselines. To fairly evaluate the performance, we also propose three specific evaluation metrics. Experimental results demonstrate the utility of the annotated MedLane dataset and the effectiveness of the proposed model DECLARE[1]

1 Introduction

Health literacy is generally defined as the ability of patients to obtain, process, understand, and communicate basic health information (Parker et al., 1999), which is significantly important for making health decisions and ensuring treatment outcomes. The increasing accessibility of technology information makes patients have more opportunities to access health information. However, it is challenging for patients, especially with limited health literacy, to read and understand health materials such as clinical notes written by doctors, with medical jargon (Korsch et al., 1968), abbreviations, and professional language (Friedman et al., 2002). The lack of proper communication between doctors and patients not only results in a tense doctor-patient relationship (Ha and Longnecker, 2010) but also increases the risk of adverse health outcomes over time (Sudore et al., 2006). Therefore, there is a great need to *simplify professional clinical language to layperson-understandable language*.

1.1 Why We Need a New Dataset?

To implement an automated clinical language simplification system, the first step is to prepare the dataset for model training. Although there are several annotated medical datasets that are summarized in Table 1, most of them do not focus on the clinical domain. As we discussed before, the writing style of clinical materials is significantly different from that used in the general biomedical area. Thus, those datasets cannot be used to train a clinical simplification model.

To the best of our knowledge, there are only two datasets focusing on clinical language simplification tasks. The n2c2-track3 dataset (Henry et al., 2020) focuses on clinical term normalization, i.e., recovering the full-term expressions for those abbreviations and acronyms in clinical notes. This term-level recovery cannot guarantee the simplicity of the translated sentences because the full expressions may still be hard to be understood by patients with low levels of health literacy. For example, the full expression of the acronym "*NC*" is "*nasal cannula*", which is still a professional medical term instead of plain language.

In (Sakakini et al., 2020), the authors annotate a clinical language simplification dataset, but it is **private**. Besides, *the number of the annotated free-text parallel sentences/instances is only 1,541*, which is too small to be enough for evaluating the real performance of machine learning models, especially for deep learning-based models. Finally, this dataset only focuses on sentence-level simplification and *does not provide term-level annotation*,

*Corresponding author.
[1]The source code of DECLARE and the MedLane dataset can be found in the https://github.com/machinelearning4health/MedLane.

Proceedings of the 29th International Conference on Computational Linguistics, pages 3550–3562
October 12–17, 2022.

Text Source	Dataset Name	Accessible	Term Normalization			Sentence Simplification	
			Abbreviations	Acronyms	Complex Phrase	Style Transfer	Training Annotations
Biomedical Article	CLEF (Elhadad et al., 2013)	✓	✓	✓	-	-	-
	MSD (Cao et al., 2020)	✓	-	-	-	✓	-
	CDSR (Guo et al., 2021)	✓	-	-	-	✓	✓
	(Devaraj et al., 2021)	✓	-	-	-	✓	✓
Perscription	e-perscription (Zheng et al., 2021)	✓	-	-	-	✓	✓
Wiki	(Van den Bercken et al., 2019)	-	-	-	-	✓	✓
	AutoMeTS (Van et al., 2020)	✓	-	-	-	✓	✓
Clinical Note	n2c2-track3 (Henry et al., 2020)	✓	✓	✓	-	-	-
	(Sakakini et al., 2020)	-	-	-	-	✓	✓
	MedLane (Ours)	✓	✓	✓	✓	✓	✓

Table 1: Dataset comparison.

which leads to the difficulty of evaluating whether professional medical terms can be correctly translated and further decreases the reliability of machine learning models. Thus, it is essential to create a **publicly available, large-scale yet fully-annotated** dataset for the clinical language simplification task.

1.2 Why We Need a New Model?

Existing models for automatic text simplification (ATS) in the biomedical domain are mainly designed based on the available datasets, which either mainly focus on term normalization or directly apply neural machine translation techniques.

The term normalization technique aims to recover the full-term expressions for medical abbreviations and acronyms using a dictionary (Vydiswaran et al., 2014; Elhadad and Sutaria, 2007; Deléger and Zweigenbaum, 2008; Qenam et al., 2017; Rahimi et al., 2020; Liu et al., 2021), i.e., only targeting the *term-level simplification* and without considering the readability and understandability of the whole sentences. Another line of work treats the original sentences as the source language and the simplified sentences as the target language, i.e., the *sentence-level simplification*. They usually borrow the ideas from neural machine translation models to only learn the style mapping function between the original and simplified sentences (Weng et al., 2019; Pattisapu et al., 2020; Li et al., 2020; Devaraj et al., 2021) but ignore the term-level translation.

In fact, the drawbacks of existing studies make them impossible to be applied to the new annotated dataset. Therefore, we need to design a new model to *achieve term-level simplification and layperson-understandable sentence generation simultaneously*.

1.3 Our Contributions

• **Dataset**. We manually annotate a new Medical Language simplification dataset named MedLane, which not only provides aligned sentence pairs but also offers term-level annotations. The dataset consists of 12,801 training samples, 1,015 validation samples, and 1,016 testing samples.

• **Approach**. Following the human annotation procedure, we design a novel end-to-end Dictionary-enhanced clinical language simplifier (shorten for DECLARE), which consists of three parts, including *a complex word locator, a neural lexical interpreter*, and *a restricted syntactic polisher*. The locator is in charge of automatically recognizing medical jargon and abbreviations in the input sentences. The neural lexical interpreter aims to replace the located terms with appropriate professional expressions selected from a predefined dictionary. Finally, a syntactic polisher is implemented to simplify the modified sentences by the interpreter and further increases the readability and understandability of original sentences, which should be significantly helpful for users with low health literacy.

• **Baseline**. We compare the proposed DECLARE against *nine baselines*, including a dictionary-based approach, a statistical machine translation approach, three neural machine translation approaches (i.e., Seq2Seq (Bahdanau et al., 2015) and its two variants), a modified text summarization model PointerNet (Vinyals et al., 2015), two state-of-the-art transformer-based pre-trained Seq2Seq models (i.e., T5 (Raffel et al., 2019) and BART (Lewis et al., 2019)), and PMBERT-MT that is built upon the pre-trained language model PubMedBERT (Gu et al., 2020), for validating the usability of the MedLane dataset and the effectiveness of our model. We also list EditNTS (Dong et al., 2019), an approach in the general ATS domain, as a baseline.

• **Evaluation**. Different from bilingual translation tasks, our task is to translate professional medical language to layperson-understandable language. We not only require the translated results to be readable but also accurate and easily understandable. Thus, we design three new yet general evaluation

metrics for the clinical simplification task. Besides, we still evaluate the results with commonly-used evaluation metrics, including BLEU (Papineni et al., 2002), METEOR (Lavie and Agarwal, 2007), ROUGE-L (Lin, 2004), CIDEr (Vedantam et al., 2015), and SARI (Xu et al., 2016).

2 Related Work

2.1 Medical Text Simplification Datasets

We summarize the widely-adopted clinical-related text simplification datasets and make a comparison with our proposed `MedLane` from different angles as shown in Table 1.

The first type of datasets focus on the normalization of abbreviations and acronyms by choosing proper explanations for them from a predefined dictionary, e.g., n2c2-track3 (Henry et al., 2020) and CLEF (Elhadad et al., 2013). The datasets are designed for a *classification* problem and do not include any sentence-level polishing and complex phrase translation. Thus, the readability of the simplified sentences may still be low even after the term-level normalization.

The second type of datasets, including MSD (Cao et al., 2020), CDSR (Guo et al., 2021), e-prescription (Zheng et al., 2021) and (Van den Bercken et al., 2019), focus on the style translation setting, which ignores the term-level simplification. The most similar dataset is the work (Sakakini et al., 2020). Except for missing the term-level annotation, it is not publicly available and only contains a small number of annotated free-text sentence pairs, which cannot be used by deep learning models.

2.2 Medical Text Simplification Method

Medical text simplification is a sub-task of automatic text simplification (ATS) (Laban et al., 2021; Dong et al., 2019; Stahlberg and Kumar, 2020; Paetzold and Specia, 2016, 2017), whose goal is to reduce the linguistic complexity of the original text to improve the readability. Besides increasing the readability, another target of the medical text simplification task is to accurately simplify professional medical terms. Existing approaches for medical text simplification can be roughly classified into two categories.

Dictionary-based approaches rely on using the expert-curated medical dictionaries to simplify the professional medical sentences (Kandula et al., 2010; Zeng and Tse, 2006; Zeng-Treitler et al., 2007; Chen et al., 2017; Lalor et al., 2019) or link medical terms with lay definitions (Chen et al., 2018) and definitions in controlled vocabularies (Polepalli Ramesh et al., 2013). These approaches are highly reliable yet cannot manage the case of polysemant, i.e., and a term can have multiple correct explanations under different cases. To solve this issue, (Sakakini et al., 2020) utilizes a pre-trained language model to select the most possible answer and then replaces the selection with the located hard terms. This approach is an advanced version of the dictionary-based approach. However, this simple replacement may lead to a decrease in sentence readability, which further makes the simplified sentence still difficult to be understood by users or patients.

Researchers also try to borrow ideas from *machine translation*, like aligning word embeddings between professional terms and daily expressions to achieve the translation (Kang et al., 2016; Weng and Szolovits, 2018), or further using back-translation procedures (Weng et al., 2019) and denoising autoencoder (Pattisapu et al., 2020) to improve the simplification results under unsupervised conditions. For supervised approaches, PharmMT (Li et al., 2020) uses the Seq2seq (Bahdanau et al., 2015) model pulsing a numerical checker to perform the simplification. (Devaraj et al., 2021) improves the BART(Lewis et al., 2019) model using the unlikelihood constraint (Welleck et al., 2019) to penalize the model generating technical words. Although the readability of these methods is higher compared to the dictionary ones by directly modifying sentences, the accuracy of term-level translation cannot be guaranteed. To address these problems, in the design of the DECLARE, we absorb the advantages of both machine translation and dictionary-based approaches to simplify clinical text from both sentence and term levels.

3 `MedLane`: A New Benchmark Dataset

3.1 Data Collection

The MIMIC-III database (Johnson et al., 2016) contains de-identified data from 58,976 ICU patient admissions, which includes several types of medical information such as demographics, medications, clinical notes, and so on. We first select clinical notes from the NOTEEVENTS table of the MIMIC-III v1.4 dataset[2] focusing on the following three sections: (1) History of present ill-

[2] https://mimic.physionet.org/mimictables/noteevents/

Algorithm 1: Sentence Selection Algorithm

Input: Target sentence s, top-3000 word set T, medical abbreviation set A
Output: Selected sentence set

Tokenize s into words $[w_1, ..., w_n]$;
for $i = 1$ *to* n **do**
  if $w_i \in A$ **then**
    $abb = abb + 1$;
  end
  $w_i' = lemmatize(w_i)$;
  if $w_i' \notin T$ **then**
    $unc = unc + 1$;
  end
end
if $n < 10$ *or* $\frac{unc+abb}{n} > 0.5$ *or* $\frac{unc+abb}{n} < 0.1$ **then**
  return False;
else
  return True;
end

ness, (2) Brief summary of hospital course, and (3) Brief hospital course. These three sections contain thoughts and reasoning for the communication between clinicians, which are usually written with professional medical jargon and abbreviations. However, they still contain many plain sentences such as *"She now also reports of being hunger."*. To avoid translating them again, we design a heuristic feature-based sentence selection approach to filter out such sentences. In particular, we first tokenize each sentence into a set of words and then use a dictionary-based approach to match medical abbreviations. We also count the number of commonly-used English words within a given list[3]. Based on the length of the sentences (greater than 10), the ratio of medical abbreviations (smaller than 0.5), and the ratio of top-3000 words (greater than 0.1), we can automatically select candidate sentences. The sentence selection algorithm is summarized in Algorithm 1.

3.2 Data Annotation

After we collect a set of source sentences, the next step is to annotate them. However, annotating medical sentences is different from creating a parallel bilingual translation corpus. The medical language translation task aims to "translate" professional and clinical jargon to layperson-understandable language, which is still from the same language but uses different expressions. Besides, annotating bilingual translation data focuses on readability and accuracy. Except for those two perspectives, annotating medical sentences also considers understandability, which is from the perspective of patients or customers. Based on the above guidance, we invited six researchers to annotate the data. All of them are familiar with medical data. For each sentence, there are two extra senior researchers holding the Doctor of Medicine (M.D.) degree to check the annotation quality. For the translated sentences with low quality, senior researchers need to re-translate them.

The purpose of this work is to create a benchmark for the automated clinical language understanding task, which is not only used for training translation models but also for fair evaluation of different approaches. Thus, we set different requirements for workers when annotating the training data and validation/testing data. In general, they use two steps when annotating each source sentence. The first step is to paraphrase the abbreviations with the whole words or phrases. For each abbreviation, there may be several full forms. For example, "*TLC*" has two full forms[4]. One is "*thin-layer chromatography*", and the other is "*total lung capacity*". Therefore, it is important for workers to understand the context in which the abbreviation or term has been used. Note that we do not provide a dictionary for workers, and they search the full expressions on the Internet if they are not sure about the abbreviations. The second step is to use simple words to replace professorial medical expressions. Take the word "*hematocrit*"[5] as an example, which means the ratio of the volume of red blood cells to the volume of the whole blood. If we use the expression, *"the proportion of red blood cells in the blood"*, it is much more understandable for patients compared with directly using professional clinical jargon. An example in Figure 1 illustrates the annotating procedure.

When annotating the training data, workers are asked to return the final understandable sentences, i.e., the simplified ones, which will be checked by experts again to guarantee the annotation quality. For validation and testing data, we require workers to return both rephrasing and simplifying forms for each source sentence.

Note that for all the training, validation, and testing data, there is a special case that we do not need

[3] https://www.ef.com/wwen/english-resources/english-vocabulary/top-3000-words/

[4] https://medical-dictionary.thefreedictionary.com/TLC

[5] https://www.mayoclinic.org/tests-procedures/hematocrit/about/pac-20384728

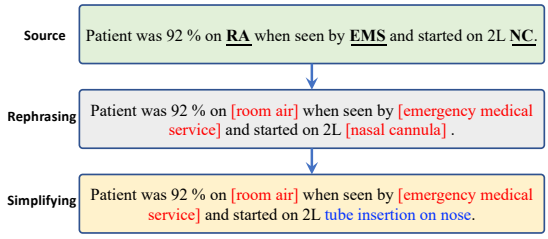

Figure 1: An example of annotating a sentence by a worker using two steps, i.e., rephrasing and simplifying. In the rephrasing step, three abbreviations are replaced by full forms. In the simplifying step, the full form "nasal cannula" is replaced by "tube insertion on nose".

to translate the source sentence. For example, it is easy to understand the sentence "*He had a set of surveillance blood cultures drawn last week, which were negative.*". These sentences are extremely useful when training an understandable translation model because they can be considered as important indicators for guiding model learning. In the validation and testing data, another special case is that the sentence may not be simplified any more. For example, the source sentence is "*She also had subjective SOB with CXR suggesting fluid overload.*", and the rephrased and simplified sentences are the same, which is "*She also had subjective [shortness of breath] with [chest x-ray] suggesting fluid overload.*".

# of tokens in the source sentences	14,780
# of tokens in the target sentences	14,278
# of overlapped tokens between source & target	12,501
Avg. length of the source sentences	20.6
Avg. length of the target sentences	24.0
Avg. # of abbreviations in validation & testing sets	1.2

Table 2: `MedLane` data statistics.

3.3 Dataset Statistics

The `MedLane` dataset contains 12,801 sentences for training, 1,015 sentences for validation, and 1,016 sentences for testing. Table 2 shows the statistics of the `MedLane` dataset, which are different from those of traditional machine translation datasets. First, the way of annotation is different, as we discussed in the previous section. Second, there are a large number of overlapped tokens between source and target sentences, which is also different from traditional machine translation. These differences make our task unique and challenging.

4 DECLARE: An Effective Approach

Motivated by the annotation steps, we propose an effective end-to-end model named DECLARE for the automated clinical language simplification task. In particular, we collect a medical dictionary to estimate the possible full-term expressions of abbreviations in the input sentences. The proposed model is shown in Figure 2, which consists of three main modules, i.e., a complex word locator (CWL), a dictionary-based neural lexical interpreter (DNLI), and a restricted syntactic simplification polisher (RSSP).

Given a tokenized professional medical sentence $W = [w_1, w_2, \cdots, w_n]$, where n denotes the number of tokens, the locator aims to dig out possible phrases that need to be simplified or translated. In the neural lexical interpreter, the chosen phrases will be replaced with full-term expressions selected from the medical dictionary. Finally, the replaced sentence will pass the polisher to generate the final output Y. These three parts tightly work together and enhance each other. Next, we introduce the design details of each module, respectively.

4.1 Complex Word Locator (CWL)

As we discussed in Section 3, clinical language understanding is different from the traditional machine translation task, and we only need to modify a part of professional medical jargon in the input sentence $W = [w_1, w_2, \cdots, w_n]$ and keep the remaining words. Towards this end, we design a complex word locator to find out the tokens that need to be modified. Note that in the annotated dataset, we have such indicators that which tokens are modified. In particular, we use a pre-trained BERT model (Devlin et al., 2019) with PubMed data, i.e., PubMedBERT (Gu et al., 2020), to learn a representation for each input token w_i, and a fully-connected layer (FC) followed by the softmax function is used to calculate the probability of each token to be modified or not. Let $\hat{\mathbf{p}}_i$ denote the binary probability vector, and we have

$$\hat{\mathbf{p}}_i = \text{softmax}(\text{FC}(\text{PubMedBERT}(w_i))). \quad (1)$$

Let $\mathbf{p}_i \in \{0, 1\}^2$ be the ground truth vector for the i-th token and $\mathcal{L}_p$ denote the average cross-entropy (CE) loss function, i.e.,

$$\mathcal{L}_p = \frac{1}{n} \sum_{i=1}^{n} \text{CE}(\mathbf{p}_i, \hat{\mathbf{p}}_i). \quad (2)$$

4.2 Dictionary-based Neural Lexical Interpreter (DNLI)

When workers annotate the source sentence, the first step is to rephrase the abbreviations. Using the

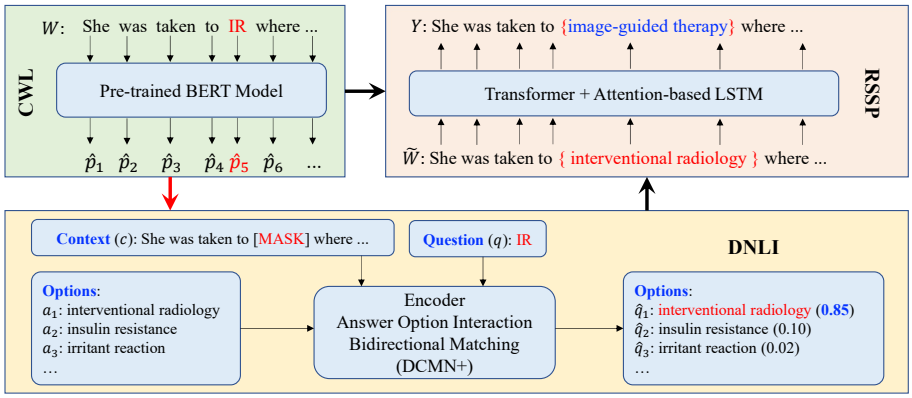

Figure 2: Overview of the proposed DECLARE model.

designed locator, DECLARE is able to identify the possible abbreviations. Then we use a dictionary-based neural lexical interpreter to automatically substitute it with an appropriate full form for each located token. Note that if the located token does not have any full-term expression, we will keep it in the sentence. Some abbreviations have several full-term expressions, and the proposed neural lexical interpreter will automatically choose one with the highest probability based on the context information, i.e., the remaining tokens in the sentence. The details of constructing the dictionary can be found in Section 5.1.

To recover the most appropriate full version of located token w_i, we borrow the idea of DCMN+ (Zhang et al., 2020) to design the dictionary-based neural lexical interpreter. Specifically, we mask the located token w_i from the input sentence, i.e., the masked sentence is $W' = [w_1, \cdots, w_{i-1}, [\text{MASK}], w_{i+1}, \cdots, w_n]$, which can be considered as the context information. Then the located token w_i can be regarded as the question, and all the full terms $A = \{a_1, \cdots, a_m\}$ are considered as options, where m is the number of possible full versions. The goal is to select the best candidate $\hat{a}_j$ from the options A when given the context vector W' and the question w_i.

We first encode W' with PubMedBERT and denote the encoded vector as $\mathbf{c}$. Similarly, the question w_i can also be mapped to a vector $\mathbf{q}$, and each option a_j can be converted to a vector $\mathbf{a}_j$. Then using answer option interaction and bidirectional matching modules in DCMN+, we can estimate the probability of each full-term expression to be selected, which is denoted as $\mathbf{q} = [q_1, \cdots, q_m]$.

Note that there may be multiple located tokens in a sentence, and we will generate the corresponding number of $\{$*context, question, options*$\}$ pairs. Besides, when annotating the `MedLane` dataset, the annotators use square brackets "[]" to indicate the correct full term expressions as shown in Figure 1. Thus, there are ground truths that are denoted as $\hat{\mathbf{q}}$ for $\{$*context, question, options*$\}$ pairs. Assume that there are l located tokens in the input sentence, then we will have the following loss function:

$$\mathcal{L}_q = \frac{1}{l} \sum_{j=1}^{l} \text{CE}(\mathbf{q}_j, \hat{\mathbf{q}}_j). \quad (3)$$

The output of the dictionary-based neural lexical interpreter is a new sentence by replacing located tokens in the locator with the best candidates selected from the dictionary, which is denoted as $\tilde{W} = [\tilde{w}_1, \cdots, \tilde{w}_{n'}]$, where n' is the number of tokens in the new sentence $\tilde{W}$.

4.3 Restricted Syntactic Simplification Polisher (RSSP)

The second step of annotation is to polish the sentences to make them more fluent, simple, and understandable by patients with low health literacy. An easy way is to directly "translate" the new sentence $\tilde{W}$ to the target sentence with a neural machine translation model. However, as we mentioned before, the clinical language understanding task is different from traditional neural machine translation, which aims to make the professional clinical jargon understandable by patients. In fact, most of the tokens in the sentences can be kept and do not translate again, and we only polish the terms labeled by the locator and replace them with the interpreter. Thus, this is a *partial translation task*.

First, we add special markers to $\tilde{W}$ to indicate the tokens to be polished, i.e., $\tilde{W} = [\tilde{w}_1, \cdots, \{\tilde{w}_i, \cdots, \tilde{w}_j\}, \cdots, \tilde{w}_{n'}]$, where the tokens from $\tilde{w}_i$ and $\tilde{w}_j$ will be "translated" by the polisher. The polisher first encodes the input sentence $\tilde{W}$ via Transformer (Vaswani et al., 2017),

i.e., $\mathbf{v}_i = \text{transformer}(w_i|w_1, \cdots, w_{n'})$, where $\mathbf{v}_i$ is the representation of the i-the token. Then an attention-based LSTM (Hochreiter and Schmidhuber, 1997; Sutskever et al., 2014; Bahdanau et al., 2015; Zhou et al., 2021) is used as the decoder to generate the t-th word y_t within braces. Let $\mathbf{h}_t$ represent the hidden state outputted by the decoder LSTM, and then we can obtain the weighted context vector $\mathbf{s}_t$ using the attention mechanism as follows:

$$\mathbf{s}_t = \sum_{i=1}^{n} \alpha_{ti}\mathbf{v}_i, \alpha_{ti} = \frac{\exp(o_{ti})}{\sum_{j=1}^{n}(\exp(o_{tj}))}, o_{ti} = \mathbf{h}_t\mathbf{v}_i^{\top}.$$

Finally, we can obtain the probability of the t-th word: $\mathbf{r}_t = \text{softmax}(\mathbf{W}[\mathbf{h}_t; \mathbf{s}_t] + \mathbf{b})$, where $\mathbf{W}$ and $\mathbf{b}$ are two parameters. Assume that the k-the element of $\mathbf{r}_t$ corresponds to the truth token, and then we have the loss of the polisher as follows:

$$\mathcal{L}_r = -\frac{1}{j-i+1}\sum_{t=i}^{j}\log(\mathbf{r}_t[k]). \quad (4)$$

4.4 Training

The proposed DECLARE is an end-to-end model, and we can train the model using the following loss function:

$$\mathcal{L} = \mathcal{L}_p + \mathcal{L}_q + \mathcal{L}_r. \quad (5)$$

In the evaluation stage, we can run these three modules one by one to generate the final understandable sentences.

5 Experiments

5.1 Experimental Setups

Dictionary Construction. We construct the mapping dictionary by collecting medical abbreviations and their corresponding full forms from the book (Dorland, 2016) and online sources, including Charleston Area Medical Center[6], Taber's Medical Dictionary[7], your dictionary[8], MedicineNet[9], and Wikipedia[10].

Baselines. We use the following approaches as baselines: a simple term replacement approach named *Dictionary-based model*, a statistical machine translation (SMT) system *Moses*[11], neural machine translation approaches, including *Seq2Seq* (Bahdanau et al., 2015) and its two variants *Seq2Seq−* and *Seq2Seq-S*, a modified version of the pointer network (Vinyals et al., 2015) *PointerNet*, state-of-the-art language models *BART* (Lewis et al., 2019) and *T5* (Raffel et al., 2019), and a modified BERT-based simplifier *BERT-MT*. We also include *EditNTS* (Dong et al., 2019) – a method proposed for the general ATS – as a baseline. Although there are other more advanced methods (Martin et al., 2020, 2019; Maddela et al., 2020) in the general ATS domain, they all require special grammar level information like the part of speech and syntactic tree information, which is hard to obtain for our clinical domain dataset. Thus, they are not compared in our experiments.

Parameter Settings. For the dictionary method, we use the pre-constructed dictionary as the same as the DECLARE model. For the statistical model Moses, we follow the training procedure listed on the User Manual and Code Guide file[12]. For neural machine translation models and text summarization baseline, we all conduct a grid search to find the optimal parameters. For Seq2Seq, Seq2Seq−, Seq2Seq-S, and PointerNet, the hidden size is set to 256 for both encoder and decoder by greedy search, and the learning rate is set to $1e-3$. We use Adam (Kingma and Ba, 2014) as the optimizer. Tokenization is performed using NLTK word tokenizer (Bird et al., 2009). The early stop is also applied by checking the BLEU score (Papineni et al., 2002) on the validation set, and the training batch size is set to 30.

For EditNTS, we use the original default parameter setting with the learning rate of $1e-3$ for the Adam optimizer. For the BERT-MT model, the hidden size is the same as that of PubMedBERT, which is 786. We also use the default AdamW optimizer used by PubMedBERT with the learning rate as $5e-5$, the warm-up method, the default PubMedBERT vocabulary, and tokenization are applied. For BART and T5, the setting of the optimizer and training procedure is the same as the BERT-MT.

For the proposed DECLARE, the locator is based on PubMedBERT to perform token-level classification, and we use the default setting of PubMedBERT to train the locator. For the dictionary-based neural interpreter, we use the same parameter setting as (Zhang et al., 2020). The max size of the

[6] https://bit.ly/3uMexm6
[7] https://bit.ly/3uR4DzL
[8] https://bit.ly/3fkuNEu
[9] https://bit.ly/3oqUw27
[10] https://bit.ly/3yg7l3K
[11] http://www.statmt.org/moses/
[12] http://www.statmt.org/moses/manual/manual.pdf

Model	BLEU-1	BLEU-2	BLEU-3	BLEU-4	BLEU	METEOR	ROUGE-L	CIDEr	SARI	HIT	CWR	AScore
Dictionary	0.7158	0.6364	0.5684	0.5076	0.6070	0.3933	0.7308	4.2037	37.3391	0.5572	0.6407	0.5948
Moses	0.7880	0.7130	0.6530	0.6016	0.6889	0.4237	0.8188	5.1046	51.6827	0.6823	0.7543	0.6859
Seq2seq	0.7136	0.6322	0.5969	0.5160	0.6147	0.3533	0.7609	4.1299	46.1328	0.7388	0.7980	0.6648
Seq2seq-	0.5066	0.3315	0.2373	0.1787	0.3135	0.1859	0.4948	1.2670	24.5346	0.6427	**0.8367**	0.4070
Seq2seq-S	0.7180	0.6386	0.5778	0.5267	0.6153	0.3604	0.7683	4.2635	46.5085	0.7331	0.7953	0.6630
PointerNet	0.6870	0.5904	0.5158	0.4541	0.5618	0.3338	0.7285	3.9458	42.2857	0.6414	0.7555	0.5949
EditNTS	0.8213	0.7801	0.7452	0.7132	0.7649	0.4674	0.7401	5.9508	62.6036	0.6405	0.6915	0.7116
BART	0.7148	0.6755	0.6396	0.6060	0.6590	**0.5320**	0.7616	4.9783	70.3058	0.5266	0.7311	0.6191
T5	0.7223	0.6812	0.6445	0.6103	0.6646	0.5305	0.7645	5.0629	71.3255	0.5262	0.7342	0.6220
BERT-MT	0.8003	0.7428	0.6952	0.6531	0.7228	0.4566	0.8218	5.3293	**72.2260**	0.7808	0.7358	0.7417
DECLARE	**0.8624**	**0.8291**	**0.8004**	**0.7737**	**0.8165**	0.5290	**0.8894**	**6.7212**	70.8583	**0.7986**	0.7328	**0.7983**
↑	+5.0%	+6.3%	+7.4%	+8.5%	+6.7%	-0.5%	+8.2%	+26.1%	-1.9%	+2.2%	-12.4%	+7.6%

Table 3: Performance evaluation of all the baselines with different metrics. ↑ denotes the percentage of performance gain compared with the best baselines.

answers is set to 8. The maximum length of the input sentence is set to 64 during training. The learning rate is set to $5e-5$ with ten epochs, and an early stop is adopted. For the restricted polisher, the setting is the same as the BERT-MT model except for the restricted translation setting. PubMedBERT is applied with an LSTM decoder that has the same hidden size.

In the evaluation stage, the same NLTK word tokenizer is applied as baselines to break the sentences into words for calculating the scores for a fair comparison. All models are trained on Ubuntu 16.04 with 128GB memory and an Nvidia Tesla P100 GPU.

Evaluation Metrics. We use BLEU (Papineni et al., 2002), METEOR (Lavie and Agarwal, 2007), ROUGE-L (Lin, 2004), and CIDEr (Vedantam et al., 2015) scores as the evaluation metrics, which are widely-used for the machine translation task. Besides, we use SARI (Xu et al., 2016), which is specially designed for the general ATS task. It combines the n-gram evaluation method of the BLEU score and rewards the replacement of the input words. However, it fails to cover the accuracy requirement of the professional medical term simplification.

Since our task is different from traditional machine translation and ATS tasks, directly applying existing evaluation metrics cannot fairly evaluate the performance of different models. Since our task is different from traditional machine translation tasks, directly applying existing evaluation metrics cannot fairly evaluate the performance of different models. Here, we use an example in Figure 3 to demonstrate the failure of existing evaluation metrics, such as the BLEU score. If we directly copy the original source sentence as the answer, a very high BLEU score can be obtained, which is 0.85. However, the critical term "*pt*" is not translated. Without translating this term, patients or customers may not totally understand the meaning of this sentence. Thus, it is necessary to design task-specific metrics.

Source: vascular saw the pt and did not feel that there was an acute need for an invasive procedure.
Target: vascular saw the patient and did not feel that there was an acute need for an invasive procedure.

Figure 3: Example of the failure of existing metrics.

• *Hit Ratio (HIT)*. A key challenge of medical language translation is to translate professional medical jargon into layperson-understandable words. Let n_p denote the number of professional medical terms in a source sentence and n_t be the number of correctly translated terms in the target. We then have the HIT ratio, which is $HIT = \frac{n_t}{n_p}$.

• *Common Word Ratio (CWR)*. To evaluate the simplicity of the translated sentences, we follow the work of Dale–Chall readability (McClure, 1987) to calculate the common word ratio for each output sentence. We first lemmatize each word of the output. If the lemmatized form is in the top-3000 commonly-used English words, then it is a common word. Otherwise, it is not a common word. Let n_c denote the number of common words in the translated sentence, and n represents the length of the translated sentence. The common word ratio score is $CWR = \frac{n_c}{n}$.

• *Aggregation Score (AScore)*. The quality of the translated sentences is not only decided by the readability (BLEU) but also related to the correctness (HIT) and simplicity (CWR). Among these three perspectives, readability and correctness should be more important than simplicity. Thus, we design a

new score to model them jointly, which is

$$Ascore = \frac{1 + \alpha^2 + \beta^2}{\frac{\alpha^2}{BLEU} + \frac{\beta^2}{HIT} + \frac{1}{CWR}},$$

where α and β are parameters for controlling the importance of BLEU and HIT scores. If any of the three metrics is 0, then it will be added to a very small number such as 10^{-8} to avoid AScore being 0. We take $\alpha = 2$ and $\beta = 1.5$ in our experiments.

5.2 Experimental Result Analysis

Table 3 shows the experimental results of all baselines and DECLARE on different evaluation metrics. The Dictionary method is the simplest approach, but its performance is not the worst compared with other baselines in terms of transitional machine translation evaluation metrics such as BLEUs. The reason is that there are many overlapping tokens in both source and target sentences, which is the main difference between the traditional machine translation task and our clinical language understanding task. These results also confirm that we need to design new evaluation metrics for this new task.

General neural network-based approaches, including Seq2Seq, its variants, and PointerNet, have a relatively low BLEU score. The reason is that the labeled data is insufficient for them to train a powerful translation model from scratch. For the general ATS method EditNTS, we can observe a relatively high score for BLEU. EditNTS is good at keeping the information. However, due to the lack of external knowledge support, the HIT score is still not satisfactory enough, proving the importance of the DNLI module. In addition, EditNTS is designed for the general ATS domain. Many attributes are not suitable for our task, which can also contribute to the bad performance. For the transformer-based pre-trained sequence to sequence models T5 and BART, the BLEU score is relatively high but with a much lower HIT score compared to Seq2Seq and PointerNet. The unsatisfied results may be related to the domain shifting problem since T5 or BART models are not pre-trained for the medical text-domain. On the contrary, BERT-MT conducting pre-training on a large medical language corpus significantly increases the ability of model learning. Hence, the performance of BERT-MT is the best among all the baselines. However, the proposed DECLARE mimics the human annotation procedure and employs a mapping dictionary with a novel model design, which leads to achieving the best performance compared with all the baselines.

Using the pre-trained language models is helpful to attain a higher SARI score because the pre-training technique can increase the models' generalization ability and benefit the word replacement rewards of the SARI metric. However, SARI does not focus on the accuracy of simplifying professional medical abbreviations, which makes it unable to comprehensively and fairly evaluate the results.

From the view of the HIT score, we can find a sufficient gap between generation-based methods (including Seq2Seq, Seq2Seq-S, BERT-MT, and DECLARE) and other methods. To achieve a high HIT score, the accurate recognition of abbreviations is necessary. Moreover, the models should understand the context correctly, which is an advantage of neural network-based models. Besides, another requirement is the generation ability. The use of a reference mechanism probably limits the generation ability of the PointerNet model, and thus, it does not achieve a high HIT score.

The CWR score can reflect the simplicity of sentences in a scene. However, we should notice that the higher CWR scores do not mean better performance. The reason is that translating professional medical terms will inevitably result in some uncommon words. Thus, we should attach less importance to the CWR score when calculating the comprehensive rank.

The top 3 models in the view of AScore are DECLARE, BERT-MT, and EditNTS, which is reasonable. The AScore attaches the highest importance to BLEU, followed by the HIT and CWR scores. Using the harmonic mean can make sure that we penalize the tendency of going overboard on one subject and guarantee good general performance.

6 Conclusion

This paper aims to benchmark a new, challenging, yet practical task of automated clinical language simplification by constructing a high-quality `MedLane` dataset and proposing a new model DECLARE that mimics the human annotation procedure. We conducted experiments on the annotated `MedLane` dataset by implementing nine strong baselines against DECLARE. Experimental results confirmed the utility of the constructed `MedLane` dataset and the effectiveness of the proposed DECLARE for addressing the automated clinical language simplification task.

7 Broader Impacts

Current health care service and health information technology (HIT) system design strive to provide accessible ways for patients to be engaged in their own care and make informed decisions. One instance is to make personal health records accessible to patients through patient portals (the electronic personal health record systems connected to organizations' electronic health record systems). However, only providing access to health records is insufficient to fully empower patients. Patients may struggle to understand those records due to such reasons as low health literacy, unfamiliarity with medical jargon and clinical abbreviations, or difficulty in understanding the complex structure of medical language. Currently, only a few studies target this practical issue. In this paper, we not only provide a human-annotated dataset, design a new model but also propose three evaluation metrics for benchmarking the clinical language simplification task. Our work will expedite research in multiple domains, including but not limited to, natural language processing, machine learning, and healthcare informatics.

8 Ethical Consideration

The original data of our study are directly extracted from the MIMIC-III database (Johnson et al., 2016), where all private health information was de-identified. The MIMIC-III database was performed under Health Insurance Portability and Accountability Act (HIPAA) standards, which require the removal of all the identifying data elements in the list of HIPAA (e.g., name, phone number, address, and so on). Thus, this is no privacy issue for the data that we use. When annotating the dataset, all annotators submitted all required consent forms. Since this work only focuses on simplifying clinical text, and no additional identified and private information is added. As a result, the protection of privacy is preserved. For disseminating our dataset to be publicly available, we will follow the same requirement of the MIMIC-III data. In other words, the requester must complete a recognized training for protecting human research participants and sign an agreement to protect the data privacy following the requirement of the PhysioNet[13] (Goldberger et al., 2000).

[13] https://physionet.org/

References

Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. 2015. Neural machine translation by jointly learning to align and translate. In *ICLR 2015 : International Conference on Learning Representations 2015*.

Laurens Van den Bercken, Robert-Jan Sips, and Christoph Lofi. 2019. Evaluating neural text simplification in the medical domain. In *The World Wide Web Conference*, pages 3286–3292.

Steven Bird, Ewan Klein, and Edward Loper. 2009. *Natural language processing with Python: analyzing text with the natural language toolkit*. " O'Reilly Media, Inc.".

Yixin Cao, Ruihao Shui, Liangming Pan, Min-Yen Kan, Zhiyuan Liu, and Tat-Seng Chua. 2020. Expertise style transfer: A new task towards better communication between experts and laymen. In *Proceedings of ACL*, pages 1061–1071.

Jinying Chen, Emily Druhl, Balaji Polepalli Ramesh, Thomas K Houston, Cynthia A Brandt, Donna M Zulman, Varsha G Vimalananda, Samir Malkani, and Hong Yu. 2018. A natural language processing system that links medical terms in electronic health record notes to lay definitions: system development using physician reviews. *Journal of medical Internet research*, 20(1):e26.

Jinying Chen, Abhyuday N Jagannatha, Samah J Fodeh, and Hong Yu. 2017. Ranking medical terms to support expansion of lay language resources for patient comprehension of electronic health record notes: adapted distant supervision approach. *JMIR medical informatics*, 5(4):e8531.

Louise Deléger and Pierre Zweigenbaum. 2008. Paraphrase acquisition from comparable medical corpora of specialized and lay texts. In *AMIA Annual Symposium Proceedings*, volume 2008, page 146.

Ashwin Devaraj, Iain Marshall, Byron C Wallace, and Junyi Jessy Li. 2021. Paragraph-level simplification of medical texts. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 4972–4984.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. Bert: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186.

Yue Dong, Zichao Li, Mehdi Rezagholizadeh, and Jackie Chi Kit Cheung. 2019. Editnts: An neural programmer-interpreter model for sentence simplification through explicit editing. *arXiv preprint arXiv:1906.08104*.

Dorland. 2016. *Dorland's Dictionary of medical acronyms & abbreviations, Seventh Edition*. Elsevier, Inc.

N Elhadad, WW Chapman, T O'Gorman, M Palmer, and G Savova. 2013. The share schema for the syntactic and semantic annotation of clinical texts. *Under Review*.

Noemie Elhadad and Komal Sutaria. 2007. Mining a lexicon of technical terms and lay equivalents. In *Biological, translational, and clinical language processing*, pages 49–56.

Carol Friedman, Pauline Kra, and Andrey Rzhetsky. 2002. Two biomedical sublanguages: a description based on the theories of zellig harris. *Journal of biomedical informatics*, 35(4):222–235.

Ary L Goldberger, Luis AN Amaral, Leon Glass, Jeffrey M Hausdorff, Plamen Ch Ivanov, Roger G Mark, Joseph E Mietus, George B Moody, Chung-Kang Peng, and H Eugene Stanley. 2000. Physiobank, physiotoolkit, and physionet: components of a new research resource for complex physiologic signals. *circulation*, 101(23):e215–e220.

Yu Gu, Robert Tinn, Hao Cheng, Michael Lucas, Naoto Usuyama, Xiaodong Liu, Tristan Naumann, Jianfeng Gao, and Hoifung Poon. 2020. Domain-specific language model pretraining for biomedical natural language processing. *arXiv preprint arXiv:2007.15779*.

Yue Guo, Wei Qiu, Yizhong Wang, and Trevor Cohen. 2021. Automated lay language summarization of biomedical scientific reviews. In *Proceedings of AAAI*.

Jennifer Fong Ha and Nancy Longnecker. 2010. Doctor-patient communication: a review. *Ochsner Journal*, 10(1):38–43.

Sam Henry, Yanshan Wang, Feichen Shen, and Ozlem Uzuner. 2020. The 2019 national natural language processing (nlp) clinical challenges (n2c2)/open health nlp (ohnlp) shared task on clinical concept normalization for clinical records. *Journal of the American Medical Informatics Association*, 27(10):1529–1537.

Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long short-term memory. *Neural computation*, 9(8):1735–1780.

Alistair EW Johnson, Tom J Pollard, Lu Shen, H Lehman Li-Wei, Mengling Feng, Mohammad Ghassemi, Benjamin Moody, Peter Szolovits, Leo Anthony Celi, and Roger G Mark. 2016. Mimic-iii, a freely accessible critical care database. *Scientific data*, 3(1):1–9.

Sasikiran Kandula, Dorothy Curtis, and Qing Zeng-Treitler. 2010. A semantic and syntactic text simplification tool for health content. In *AMIA annual symposium proceedings*, volume 2010, page 366.

Hong Jin Kang, Tao Chen, Muthu Kumar Chandrasekaran, and Min Yen Kan. 2016. A comparison of word embeddings for english and cross-lingual chinese word sense disambiguation.

Diederik P Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*.

Barbara M Korsch, Ethel K Gozzi, and Vida Francis. 1968. Gaps in doctor-patient communication: I. doctor-patient interaction and patient satisfaction. *Pediatrics*, 42(5):855–871.

Philippe Laban, Tobias Schnabel, Paul Bennett, and Marti A Hearst. 2021. Keep it simple: Unsupervised simplification of multi-paragraph text. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 6365–6378.

John P Lalor, Beverly Woolf, and Hong Yu. 2019. Improving electronic health record note comprehension with noteaid: randomized trial of electronic health record note comprehension interventions with crowdsourced workers. *Journal of medical Internet research*, 21(1):e10793.

Alon Lavie and Abhaya Agarwal. 2007. Meteor: An automatic metric for mt evaluation with high levels of correlation with human judgments. In *Proceedings of the second workshop on statistical machine translation*, pages 228–231.

Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Ves Stoyanov, and Luke Zettlemoyer. 2019. Bart: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. *arXiv preprint arXiv:1910.13461*.

Jiazhao Li, Corey Lester, Xinyan Zhao, Yuting Ding, Yun Jiang, and VG Vinod Vydiswaran. 2020. Pharmmt: A neural machine translation approach to simplify prescription directions. In *Proceedings of EMNLP: Findings*, pages 2785–2796.

Chin-Yew Lin. 2004. Rouge: A package for automatic evaluation of summaries. In *Text summarization branches out*, pages 74–81.

Lisa Grossman Liu, Raymond H Grossman, Elliot G Mitchell, Chunhua Weng, Karthik Natarajan, George Hripcsak, and David K Vawdrey. 2021. A deep database of medical abbreviations and acronyms for natural language processing. *Scientific Data*, 8(1):1–9.

Mounica Maddela, Fernando Alva-Manchego, and Wei Xu. 2020. Controllable text simplification with explicit paraphrasing. *arXiv preprint arXiv:2010.11004*.

Louis Martin, Angela Fan, Éric de la Clergerie, Antoine Bordes, and Benoît Sagot. 2020. Muss: multilingual unsupervised sentence simplification by mining paraphrases. *arXiv preprint arXiv:2005.00352*.

Louis Martin, Benoît Sagot, Eric de la Clergerie, and Antoine Bordes. 2019. Controllable sentence simplification. *arXiv preprint arXiv:1910.02677*.

Glenda M McClure. 1987. Readability formulas: Useful or useless? *IEEE transactions on professional communication*, (1):12–15.

Gustavo Paetzold and Lucia Specia. 2016. Unsupervised lexical simplification for non-native speakers. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 30.

Gustavo Paetzold and Lucia Specia. 2017. Lexical simplification with neural ranking. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 2, Short Papers*, pages 34–40.

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. Bleu: a method for automatic evaluation of machine translation. In *Proceedings of ACL*, pages 311–318.

Ruth M Parker, Mark V Williams, Barry D Weiss, David W Baker, Terry C Davis, Cecilia C Doak, Leonard G Doak, Karen Hein, Cathy D Meade, Joann Nurss, et al. 1999. Health literacy-report of the council on scientific affairs. *Jama-Journal of the American Medical Association*, 281(6):552–557.

Nikhil Pattisapu, Nishant Prabhu, Smriti Bhati, and Vasudeva Varma. 2020. Leveraging social media for medical text simplification. In *Proceedings of SIGIR*, pages 851–860.

Balaji Polepalli Ramesh, Thomas Houston, Cynthia Brandt, Hua Fang, and Hong Yu. 2013. Improving patients' electronic health record comprehension with noteaid. In *MEDINFO 2013*, pages 714–718. IOS Press.

Basel Qenam, Tae Youn Kim, Mark J Carroll, and Michael Hogarth. 2017. Text simplification using consumer health vocabulary to generate patient-centered radiology reporting: translation and evaluation. *Journal of medical Internet research*, 19(12):e417.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J Liu. 2019. Exploring the limits of transfer learning with a unified text-to-text transformer. *arXiv preprint arXiv:1910.10683*.

Afshin Rahimi, Timothy Baldwin, and Karin Verspoor. 2020. Wikiumls: Aligning umls to wikipedia via cross-lingual neural ranking. *arXiv preprint arXiv:2005.01281*.

Tarek Sakakini, Jong Yoon Lee, Aditya Duri, Renato FL Azevedo, Victor Sadauskas, Kuangxiao Gu, Suma Bhat, Dan Morrow, James Graumlich, Saqib Walayat, et al. 2020. Context-aware automatic text simplification of health materials in low-resource domains. In *Proceedings of the 11th International Workshop on Health Text Mining and Information Analysis*, pages 115–126.

Felix Stahlberg and Shankar Kumar. 2020. Seq2edits: Sequence transduction using span-level edit operations. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 5147–5159.

Rebecca L Sudore, Kristine Yaffe, Suzanne Satterfield, Tamara B Harris, Kala M Mehta, Eleanor M Simonsick, Anne B Newman, Caterina Rosano, Ronica Rooks, Susan M Rubin, et al. 2006. Limited literacy and mortality in the elderly. *Journal of general internal medicine*, 21(8):806–812.

Ilya Sutskever, Oriol Vinyals, and Quoc V Le. 2014. Sequence to sequence learning with neural networks. *Advances in Neural Information Processing Systems*, 27:3104–3112.

Hoang Van, David Kauchak, and Gondy Leroy. 2020. Automets: The autocomplete for medical text simplification. *arXiv preprint arXiv:2010.10573*.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in neural information processing systems*, pages 5998–6008.

Ramakrishna Vedantam, C Lawrence Zitnick, and Devi Parikh. 2015. Cider: Consensus-based image description evaluation. In *Proceedings of CVPR*, pages 4566–4575.

Oriol Vinyals, Meire Fortunato, and Navdeep Jaitly. 2015. Pointer networks. In C. Cortes, N. D. Lawrence, D. D. Lee, M. Sugiyama, and R. Garnett, editors, *Advances in Neural Information Processing Systems 28*, pages 2692–2700.

V. G. Vinod Vydiswaran, Qiaozhu Mei, David A Hanauer, and Kai Zheng. 2014. Mining consumer health vocabulary from community-generated text. *AMIA*, 2014:1150.

Sean Welleck, Ilia Kulikov, Stephen Roller, Emily Dinan, Kyunghyun Cho, and Jason Weston. 2019. Neural text generation with unlikelihood training. *arXiv preprint arXiv:1908.04319*.

Wei-Hung Weng, Yu-An Chung, and Peter Szolovits. 2019. Unsupervised clinical language translation. In *Proceedings of SIGKDD*, pages 3121–3131.

Wei-Hung Weng and Peter Szolovits. 2018. Mapping unparalleled clinical professional and consumer languages with embedding alignment.

Wei Xu, Courtney Napoles, Ellie Pavlick, Quanze Chen, and Chris Callison-Burch. 2016. Optimizing statistical machine translation for text simplification. *Transactions of the Association for Computational Linguistics*, 4:401–415.

Qing T Zeng and Tony Tse. 2006. Exploring and developing consumer health vocabularies. *Journal of the American Medical Informatics Association*, 13(1):24–29.

Qing Zeng-Treitler, Sergey Goryachev, Hyeoneui Kim, Alla Keselman, and Douglas Rosendale. 2007. Making texts in electronic health records comprehensible to consumers: a prototype translator. In *AMIA Annual Symposium Proceedings*, volume 2007, page 846.

Shuailiang Zhang, Hai Zhao, Yuwei Wu, Zhuosheng Zhang, Xi Zhou, and Xiang Zhou. 2020. Dcmn+: Dual co-matching network for multi-choice reading comprehension. In *Proceedings of AAAI*, volume 34, pages 9563–9570.

Yifan Zheng, Yun Jiang, Michael P Dorsch, Yuting Ding, VG Vinod Vydiswaran, and Corey A Lester. 2021. Work effort, readability and quality of pharmacy transcription of patient directions from electronic prescriptions: a retrospective observational cohort analysis. *BMJ quality & safety*, 30(4):311–319.

Yao Zhou, Haonan Wang, Jingrui He, and Haixun Wang. 2021. From intrinsic to counterfactual: On the explainability of contextualized recommender systems. *CoRR*, abs/2110.14844.

WikiHan: A New Comparative Dataset for Chinese Languages

Kalvin Chang and **Chenxuan Cui** and **Youngmin Kim** and **David Mortensen**
Language Technologies Institute, Carnegie Mellon University
{kalvinc, cxcui, youngmik, dmortens}@cs.cmu.edu

Abstract

Most comparative datasets of Chinese varieties are not digital; however, Wiktionary includes a wealth of transcriptions of words from these varieties. The usefulness of these data is limited by the fact that they use a wide range of variety-specific romanizations, making data difficult to compare. The current work collects this data into a single constituent (IPA, or International Phonetic Alphabet) and structured form (TSV) for use in comparative linguistics and Chinese NLP. At the time of writing, the dataset contains 67,943 entries across 8 varieties and Middle Chinese.[1] The dataset is validated on a protoform reconstruction task using an encoder-decoder cross-attention architecture (Meloni et al., 2021), achieving an accuracy of 54.11%, a PER (phoneme error rate) of 17.69%, and a FER (feature error rate) of 6.60%.

1 Introduction

The Chinese language family consists of many mutually *un*intelligible languages, which linguists cluster into seven well-established subgroups (Mandarin, Yue, Wu, Min, Hakka, Gan, and Xiang) and a few more debated subgroups (Jin, Hui, and Pinghua; Handel 2015). Each subgroup is made of varieties that may or may not be mutually unintelligible. Most scholars of Sinitic believe that all varieties except for Min languages descended from Middle Chinese (Handel, 2015), which is a set of old varieties documented in *Qièyùn* (切韻), a rhyme dictionary compiled in 601 CE.

Today, there are a total of 1.3 billion speakers of Sinitic varieties, making the family one of the largest in terms of speaker count (Eberhard et al., 2022). Eight of the subgroups have at least tens of millions of speakers. In all Chinese-speaking societies except for Hong Kong, Mandarin is the language of administration, meaning that other varieties are not commonly written, although it is possible to do so, either with Han characters (Chinese logographic characters) or romanized orthographies. Because Standard Written Chinese diverged from the spoken varieties over time (Chen, 2015), it is unclear which Han characters should be used for transcription, for many words in the spoken varieties. However, scholars can trace from which "original character" in Middle Chinese or Old Chinese the modern word descended (*běnzì* 本字) (Yang, 2000).

To determine the pronunciations of these original characters in Middle Chinese, historical linguists and dialectologists compare the pronunciations of words across modern Sinitic varieties. More broadly, *protoform reconstruction* is the inference of morphemes or words as they appeared in the ancestral languages of a set of daughter languages. Cognates—words deemed to descend from the same ancestral form—are inputs to the reconstruction, while this ancestral form is the output of the reconstruction. See Table 2 for an example of a cognate set, with Middle Chinese as the protoform and all columns to its right as the cognates in the daughter languages.

To enable fair comparison across the daughter languages, the pronunciations should be in a common phonetic or phonemic transcription system, as opposed to divergent romanizations. The problem is that large datasets that do offer phonetic transcriptions such as the Great Dictionary of Modern Chinese Dialects 現代漢語方言大詞典 (Li, 2002) are often in print form. Wiktionary, on the other hand, offers a digital compilation of pronunciation entries across several sources manually entered by users (Wiktionary contributors, 2022a). However, entries are stored in subgroup-specific romanizations on the back end. The IPA transcrip-

[1] The data is available at https://github.com/cmu-llab/wikihan and the code is available at https://github.com/cmu-llab/meloni-2021-reimplementation/.

Proceedings of the 29th International Conference on Computational Linguistics, pages 3563–3569
October 12–17, 2022.

tions are generated on the front end by Lua scripts that convert romanizations to IPA[2] and as such are not available in data exports, which only store the romanized forms. To obtain a large dataset of pronunciations in IPA, one could scrape the front end of Wiktionary, but this would be inefficient. Instead, we write our own romanization to IPA conversion modules in Epitran (Mortensen et al., 2018) for each subgroup based on those from Wiktionary. Using the outputs of these Epitran modules, we compile a large dataset of pronunciation entries in IPA in TSV form. We then show that the entries we generate can be used in a computational protoform reconstruction of Middle Chinese, demonstrating that our dataset addresses the inadequacies of previous digital comparative Sinitic datasets with respect to computational reconstruction.

2 Related Work

Linguists have created digital comparative datasets since the birth of computers. Refer to Table 1 for a list of Sinitic datasets. Wang (1970) in particular features Middle Chinese, Sino-Xenic (Japanese Kan-on, Japanese Go-on, and Sino-Korean) loanwords, and Old Mandarin (from the *Zhōngyuán Yīnyùn* 中原音韻). Similar to ours, the main objective of the dataset was to provide data for computer-assisted reconstructions of Chinese phonology. The Multi-function Chinese Character Database is interesting in that it organizes characters by all the possible syllables in the dialect, including the tone. The Database is itself a compilation of multiple print sources. Wu and List (2021) stand out for their manual annotation of salient morphemes that contributed to cognacy judgments, which they use to convert partial cognates to full words in creating cognate sets and wordlists for phylogenetic inference.

One challenge with many of the aforementioned datasets is that they organize entries by words instead of characters. Across subgroups, synonymic compound words may use different characters to express the same meaning.[3] Even though partial cognacy of such compounds is possible, the method by which partial cognates are coded into full cognates in word lists will affect the downstream task of phylogenetic inference, which assumes that words in the inputted word lists descend from the same proto-word (Wu and List, 2021). Many other sources exist and may have significantly more entries, such as the Great Dictionary of Modern Chinese Dialects. However, to our knowledge, they are not digitized, making computer-assisted reconstruction difficult.

3 Dataset and Methodology

We obtain Sinitic pronunciation entries using a CBOR snapshot of the `zh-pron` module on Wiktionary.[4] For heteronyms, characters with multiple pronunciations within a variety, each pronunciation is stored as a separate entry in the snapshot and is grouped along with pronunciations in other varieties believed to be cognate with that particular pronunciation. The dataset spans eight subgroups: the seven conventionally recognized ones (Mandarin, Yue, Wu, Min, Xiang, Gan, and Hakka) and the proposed Jin subgroup, for which Wiktionary has entries. We chose the dialects with the most data to represent each subgroup. Refer to Table 3 for the full list of dialects.[5] We restrict the dataset to single characters (morphemes), allowing us to establish cognacy between pronunciation entries across subgroups by assuming that readings grouped under the same character by Wiktionary are cognates descended from the same original character (本字).

To convert from romanization to IPA, we extend Epitran with additional modules for each subgroup (Mortensen et al., 2018). Refer to Table 3 to see which romanization system was used for each subgroup. Wiktionary supplies mapping tables for each variety[6], and we compare the mapping tables with other sources such as Wiktionary's Lua conversion scripts (see Appendix A for the full list).

We represent tones using IPA tone letters. We do not mark the neutral tone, as opposed to marking it as IPA tone 3. The transcription is relatively narrow, with diphthongs always represented as se-

[2] See `https://en.wiktionary.org/wiki/Module:nan-pron` for an example of a conversion script.

[3] For instance, tears is 目屎 (lit. eye feces) in Hokkien, 目汁 (lit. eye juice) in Hakka, and 眼淚 (eye tears) in Mandarin (Wiktionary contributors, 2022b). Here 目屎 and 目汁 are partial cognates since not every character descends from the same protoform.

[4] `https://tools-static.wmflabs.org/templatehoard/dump/latest/zh-pron.cbor`

[5] We acknowledge that including Taiwanese Hokkien entries may present a problem during reconstruction because its dialects are a mix of the Quanzhou and Zhangzhou dialects of Hokkien (a language within the Southern Min branch of the Min subgroup) to varying degrees.

[6] See `https://en.wiktionary.org/wiki/Wiktionary:About_Chinese/Xiang` for an example

Name	Citation	Varieties	Sets	Entries	Transcription	Format
Great Dictionary of Modern Chinese Dialects	Li (2002)	42	—	170,000+	SIPA	print
Dictionary on Computer	Wang (1970); Streeter (1972)	17	2,444	58,012	SIPA	tabular
Multi-Function Chinese Character Database	Research Centre for Humanities Computing, CUHK	20	7,554	—	SIPA	HTML
Peking University	Peking University (2021)	18	905	18,059	SIPA	CLDF
Wiktionary `dial-pron` module	Hóu (2004)	39	1,023	>39,000	IPA	XML
Phonological Database of Chinese dialects	List (2021)	15	140	2,789	SIPA	CLDF
Liu et al. (2007) annotated with salient morphemes	Wu and List (2021)	19	201	> 3,000	CLDF	CLDF
WikiHan (ours)	N/A	8	21,227	68,368	IPA	TSV

Table 1: Comparative Sinitic datasets. "SIPA" refers to "Sinological IPA." In our dataset, a heteronymic character can have multiple cognate sets, reflecting different sets of pronunciation variants that are only cognate with variants in the same set. Entries refers to the total number of pronunciation records across all varieties.

quences of vowel symbols and glides marked by the IPA non-syllabic diacritic: [i̯a] rather than [ja]. Fewer symbols are preferred over many: [ɲ] rather than [n^{j}].

Middle Chinese (MC) transcriptions following Baxter and Sagart (2014) are derived programmatically from *fǎnqiè* formulae[7] from *Qièyùn* (available for around 20,000 characters). Despite concerns raised by Norman and Coblin (1995) and others, we treat MC pronunciations as the gold standard protoform. We process the *Qièyùn* descriptions in three stages. First, the Middle Chinese descriptions of initial, final, tone, division (等), and openness (合開) are converted to the ASCII romanization system in Baxter and Sagart (2014). The romanizations are then converted to IPA with Epitran, using a mapping table based on SinoPy (List, 2019). Finally, we rewrite some IPA phonemes to match the phonetic transcription convention used in this dataset (e.g. [tɕ] →[t͡ɕ], [ȵ] →[ɲ]). For the tones, we use superscripts 1 through 4 to indicate what would traditionally be denoted as 平上去入. The final result is a list of nearly 20,000 characters, each with a reconstruction written in IPA symbols.

4 Experiments

We show that the dataset can be used for the protoform reconstruction task. Meloni et al. (2021) model the Latin protoform reconstruction task as a sequence to sequence transduction problem with a character-based encoder-decoder (Cho et al., 2014) with cross-attention (Bahdanau et al., 2015).[8] We reimplement their architecture, originally written in DyNet (Neubig et al., 2017), in PyTorch. The architecture consists of a language and token embeddings, an encoder GRU (Cho et al., 2014), a decoder GRU, and a multi-layer perceptron.

All daughter forms within one cognate set are concatenated into one string before entering the encoder. To distinguish between each variety, a language code is first prepended before each pronunciation entry. In the encoder, a language embedding is learned for each dialect. The same is done in the decoder for the proto-language. Token embeddings are applied to individual characters (close to a phone) in the input and are shared across each language. There is a residual connection between the attention output and the decoder RNN output before entering the multi-layer perceptron. The objective function is cross-entropy loss between the protoform and the predicted protoform.

The only difference between our PyTorch version and their code is that we do not implement variational dropout in the encoder (Gal and Ghahramani, 2016), but DyNet comes with this flavor of dropout in its RNN modules. We do implement variational dropout for the decoder, though.[9]

[7] *Fǎnqiè* spelling provides equivalence classes for the pronunciation of a syllable by using one character with the same onset and another with the same rhyme.

[8] Meloni et al. (2021)'s code is available at `https://github.com/shauli-ravfogel/Latin_reconstruction`.

[9] PyTorch's RNN, LSTM, and GRU modules do not come with variational dropout. It is possible to overwrite the respective classes with a version that implements variational dropout, though.

Character	Middle Chinese	Yue	Gan	Hakka	Jin	Mandarin	Min	Wu	Xiang
犬	/kʰwen²/	[hyːn˧˥]	[t͡ɕʰy̯ɵn˨˩˧]	[kʰi̯en˧˩]	[t͡ɕʰy̯e˩]	[t͡ɕʰy̯ɛn˨˩]	[kʰi̯ɛn˥˧]	[t͡ɕʰy̯ø˧˥]	[t͡ɕʰy̯e̞˥˧]

Table 2: Example of a complete cognate set in the dataset for the word 犬 (*dog*)

Subgroup	Dialect Chosen	Romanization	犬 romanized	Number of entries
Mandarin	Beijing	Pinyin	quǎn	20369
Yue	Cantonese	Jyutping	hyun2	16727
Wu	Shanghainese	Wiktionary's romanization	2qyoe	2877
Min	Hokkien	Pe̍h-oē-jī	khián	6145
Hakka	Sixian	Pha̍k-fa-sṳ	khién	5215
Gan	Nanchang	Wiktionary's romanization	qyon3	1195
Xiang	Old Xiang	Wiktionary's romanization	qye3	1258
Jin	Taiyuan	Wiktionary's romanization	qye1	1410

Table 3: The dialect chosen for each subgroup and its romanization of the word 犬, in addition to a count of the number of pronunciation entries per subgroup. For Min, the pronunciations are a mix of the Xiamen, Quanzhou, Zhangzhou, and Taiwanese dialects of Hokkien. For Middle Chinese, we have 14653 entries.

We find that dropout makes a significant difference when trained on the small 39-variety dataset of 1,000 cognate sets from Hóu (2004) (which ended up being around 800 sets because not every entry in Hóu (2004) had an entry in the *Qièyùn*).[10]

We now discuss how we adapted our dataset for Meloni et al. (2021)'s model. In order for the model to learn correspondences between phonemes as linguists would, we tokenize by phonemes, for example /tʰ/ and /t͡ɕʰ/, instead of characters. These two example phonemes should each be treated as one consonant despite being represented with several Unicode characters. We treat diphthongs and triphthongs as one token because they constitute one syllable, phonetically speaking. We also restrict ourselves to cognates with at least 4 entries including Middle Chinese to avoid being biased to varieties with more entries, such as Mandarin. Another decision we made is to arbitrarily take the first pronunciation when multiple variants are included in the same entry on Wiktionary.

We compare Meloni et al. (2021) against two baselines. The *random daughter* baseline selects a daughter form at random and takes that as the reconstructed protoform. This assumes that no sound change occurred from the protoform to the daughter. The *majority constituent* baseline first separates daughter forms into onset, nucleus, and coda with the `consonantal` feature of the phoneme obtained using PanPhon (Mortensen et al., 2016), reflecting domain knowledge about the syllable structure of Chinese languages. This allows us to easily obtain sound correspondences across the daughter languages. Within each constituent (onset, nucleus, and coda), we take the most common phoneme sequence. This relies on the *majority wins* heuristic employed by historical linguists wherein the most frequent sound across the daughter languages is chosen as the proto-sound (Campbell, 2013).

Meloni et al. (2021) outperforms both baselines on 3 different metrics (see Table 4): (1) *Accuracy*, the rate at which hypothesis and reference match exactly, (2) *Phoneme Error Rate* (PER), the cumulative number of phoneme edits between the hypothesis and the reference normalized by the total length of the reference (in phonemes), and (3) *Feature Error Rate* (FER) the cumulative edit distance in terms of PanPhon (Mortensen et al., 2016) features (drawn from articulatory phonetics) between the hypothesis and the reference, normalized by the total number of features in the reference (the total length of the reference in phonemes multiplied by the number of features per phoneme).

PER is more suited for the protoform reconstruction task than character error rate or edit distance because many phonemes are written with more than 1 character in IPA, as shown in the examples from above. As for FER, its benefit lies in how it is able to assign partial credit to hypothesized phonemes that are more phonetically similar. Intu-

[10] Available on Wiktionary at `https://en.wiktionary.org/wiki/Module:zh/data/dial-pron/documentation`.

Model	Accuracy	Phoneme error rate	Feature error rate
Meloni et al. (2021)	**0.5411**	**0.1769**	**0.0660**
Random daughter	0.0290	0.7367	0.2600
Majority constituent	0.0271	0.7320	0.2209

Table 4: Evaluation of Meloni et al. (2021)'s model and 2 baselines on our dataset using 3 different evaluation criteria.

itively, we would want, for example, [t] to be penalized less than [x] in some scenario where the reference is [t^h]. Both [t] and [x] differ from the reference by one character, but [t] and [t^h] are both voiceless alveolar plosives that differ only in aspiration. Finally, we prefer error rates over edit distances because it is difficult to compare results across different language families, which differ in word lengths. Sinitic words in particular are often shorter than Romance words because the former is composed of monosyllabic characters.

5 Discussion and Future Work

Our dataset is intended for a computational reconstruction of Middle Chinese, as we have demonstrated in the experiments, but can be used to accomplish much more. It can also be used for cognate prediction and for dialectometry (quantifying relationships between linguistic varieties). Along the same lines, it can be used to build phylogenetic models of Sinitic that can shed light on the history of Chinese populations. Additionally, Chinese speech models could benefit from a phonetic language model (Dalmia et al., 2019) trained on our data or from estimations of phone distributions (Li et al., 2021) in low resource varieties present in our dataset.

In a future release, we will include other varieties available on Wiktionary (Taishanese, Southwest Mandarin, Teochew, Min Bei, and Min Dong). Wiktionary also contains pronunciations for Sino-Xenic loanwords in Korean, Japanese, and Vietnamese, which linguists often reference when creating Chinese reconstructions. The more languages we include, the fewer the number of sources that Chinese historical phonologists need to consult, reducing the tediousness of work in this field.

References

Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. 2015. Neural machine translation by jointly learning to align and translate. In *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*.

William H Baxter and Laurent Sagart. 2014. *Old Chinese: A new reconstruction*. Oxford University Press.

Lyle Campbell. 2013. *Historical Linguistics: an Introduction*. Edinburgh University Press.

Ping Chen. 2015. *The Oxford Handbook of Chinese Linguistics*, chapter Language Reform in Modern China. Oxford University Press.

Kyunghyun Cho, Bart van Merriënboer, Dzmitry Bahdanau, and Yoshua Bengio. 2014. On the properties of neural machine translation: Encoder–decoder approaches. In *Proceedings of SSST-8, Eighth Workshop on Syntax, Semantics and Structure in Statistical Translation*, pages 103–111, Doha, Qatar. Association for Computational Linguistics.

Siddharth Dalmia, Xinjian Li, Alan W Black, and Florian Metze. 2019. Phoneme level language models for sequence based low resource ASR. In *ICASSP 2019 - 2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 6091–6095.

David M. Eberhard, Gary F. Simons, and Charles D. Fennig. 2022. Ethnologue: Languages of the world. twenty-fifth edition. Dallas, Texas: SIL International. Online version: `http://www.ethnologue.com`. Accessed on 11 May 2022 at `https://web.archive.org/web/20190605194504/https://www.ethnologue.com/language/zho`.

Yarin Gal and Zoubin Ghahramani. 2016. A theoretically grounded application of dropout in recurrent neural networks. In *Advances in Neural Information Processing Systems*, volume 29. Curran Associates, Inc.

Zev Handel. 2015. *The Oxford Handbook of Chinese Linguistics*, chapter The Classification of Chinese. Oxford University Press.

侯精一 Jīngyī Hóu, editor. 2004. *Xiàndài Hànyǔ fāngyán yīnkù* 现代汉语方言音库 *[Phonological database of Chinese dialects]*. Shànghǎi Jiàoyù 上海教育, Shànghǎi 上海.

Xinjian Li, Juncheng Li, Jiali Yao, Alan W Black, and Florian Metze. 2021. Phone distribution estimation for low resource languages. In *ICASSP 2021 - 2021 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 7233–7237.

李榮 Rong Li. 2002. 現代漢語方言大詞典 *[Great Dictionary of Modern Chinese Dialects]*. 現代漢語方言大詞典. 江蘇教育出版社.

Johann-Mattis List. 2019. lingpy/sinopy: Python library for quantitative tasks in Chinese historical linguistics. Version 0.3.4.

Johann-Mattis List. 2021. CLDF dataset derived from Hóu's "Phonological Database of Chinese Dialects" from 2004.

刘俐李 Lili Liu, 王洪钟 Hongzhong Wang, and 柏莹 Bai Ying. 2007. 現代漢語方言核心詞·特徵詞集. *[Collection of basic vocabulary words and characteristic dialect words in modern Chinese dialects]*. 鳳凰出版社.

Carlo Meloni, Shauli Ravfogel, and Yoav Goldberg. 2021. Ab antiquo: Neural proto-language reconstruction. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 4460–4473, Online. Association for Computational Linguistics.

Ministry of Education, Taiwan. 2008. 臺灣閩南語羅馬字拼音方案使用手冊 [Taiwanese Hokkien Romanization Handbook]. https://ws.moe.edu.tw/001/Upload/FileUpload/3677-15601/Documents/tshiutsheh.pdf.

David R. Mortensen, Siddharth Dalmia, and Patrick Littell. 2018. Epitran: Precision G2P for many languages. In *Proceedings of the Eleventh International Conference on Language Resources and Evaluation (LREC 2018)*, Miyazaki, Japan. European Language Resources Association (ELRA).

David R. Mortensen, Patrick Littell, Akash Bharadwaj, Kartik Goyal, Chris Dyer, and Lori S. Levin. 2016. Panphon: A resource for mapping IPA segments to articulatory feature vectors. In *Proceedings of COLING 2016, the 26th International Conference on Computational Linguistics: Technical Papers*, pages 3475–3484.

Graham Neubig, Chris Dyer, Yoav Goldberg, Austin Matthews, Waleed Ammar, Antonios Anastasopoulos, Miguel Ballesteros, David Chiang, Daniel Clothiaux, Trevor Cohn, Kevin Duh, Manaal Faruqui, Cynthia Gan, Dan Garrette, Yangfeng Ji, Lingpeng Kong, Adhiguna Kuncoro, Gaurav Kumar, Chaitanya Malaviya, Paul Michel, Yusuke Oda, Matthew Richardson, Naomi Saphra, Swabha Swayamdipta, and Pengcheng Yin. 2017. DyNet: The dynamic neural network toolkit.

Jerry L. Norman and W. South Coblin. 1995. A new approach to Chinese historical linguistics. *Journal of the American Oriental Society*, 115(4):576–584.

Peking University. 2021. CLDF dataset derived from Beijing University's "Chinese Dialect Vocabularies" from 1964.

Research Centre for Humanities Computing, CUHK. 漢語多功能字庫 Multi-function Chinese Character Database. https://humanum.arts.cuhk.edu.hk/Lexis/lexi-mf/dialect.php. Accessed: 2022-05-08.

Mary L. Streeter. 1972. Doc, 1971: A Chinese dialect dictionary on computer. *Computers and the Humanities*, 6(5):259–270.

William S.-Y. Wang. 1970. Project DOC: Its methodological basis. *Journal of the American Oriental Society*, 90(1):57–66.

Wiktionary contributors. 2022a. Wiktionary, the free dictionary. [Online; accessed 11-May-2022].

Wiktionary contributors. 2022b. 眼淚. Wiktionary, the free dictionary. [Online; accessed 20-Jul-2022].

Mei-Shin Wu and Johann-Mattis List. 2021. Annotating cognates in phylogenetic studies of South-East Asian languages. *Language Dynamics and Change*. https://doi.org/10.17613/rabq-7z45.

楊秀芳 Hsiu-fang Yang. 2000. Concepts and methods in the study of original characters of dialects 方言本字研究的觀念與方法. 漢學研究 *[Chinese Studies], Vol. 18*, pages 111–146.

A Sources

First, we must credit Wiktionary users for manually compiling pronunciation entries across different sources. Without them, this corpus would not be possible. While creating the Epitran romanization to IPA mapping tables and while building the Middle Chinese data, we also consulted their scripts and documentation, in addition to other sources listed below.

A.1 Middle Chinese

- SinoPy (List, 2019)
- https://github.com/ycm/cs221-proj/blob/master/preprocessing/dataset/pron/mc-pron.csv

A.2 Mandarin

- https://en.wiktionary.org/wiki/Module:cmn-pron
- https://en.wikipedia.org/wiki/Pinyin
- https://en.wikipedia.org/wiki/Help:IPA/Mandarin

A.3 Cantonese (Yue)

- https://en.wikipedia.org/wiki/Jyutping
- https://en.wiktionary.org/wiki/Wiktionary:About_Chinese/Cantonese
- https://en.wikipedia.org/wiki/Help:IPA/Cantonese

A.4 Taiwanese Hokkien (Min)

- https://en.wiktionary.org/wiki/Module:nan-pron
- https://en.wiktionary.org/wiki/Wiktionary:About_Chinese/Hakka
- https://blgjts.moe.edu.tw/doc/tmt_compare.pdf
- Ministry of Education, Taiwan (2008)
- https://zh.wikipedia.org/wiki/Help:%E8%87%BA%E7%81%A3%E8%A9%B1%E5%9C%8B%E9%9A%9B%E9%9F%B3%E6%A8%99

A.5 Xiang

- https://en.wiktionary.org/wiki/Module:hsn-pron
- https://en.wiktionary.org/wiki/Wiktionary:About_Chinese/Xiang
- https://zh.wikipedia.org/wiki/%E6%B9%98%E8%AF%AD

A.6 Jin

- https://en.wiktionary.org/wiki/Module:cjy-pron
- https://en.wiktionary.org/wiki/Wiktionary:About_Chinese/Jin

A.7 Gan

- https://en.wiktionary.org/wiki/Wiktionary:About_Chinese/Gan
- https://en.wiktionary.org/wiki/Module:gan-pron

A.8 Wu

- https://en.wiktionary.org/wiki/Wiktionary:About_Chinese/Wu
- https://en.wiktionary.org/wiki/Module:wuu-pron

A.9 Hakka

- https://en.wiktionary.org/wiki/Wiktionary:About_Chinese/Hakka
- https://en.wikipedia.org/wiki/Sixian_dialect
- https://en.wiktionary.org/wiki/Module:hak-pron

Visual Recipe Flow: A Dataset for Learning Visual State Changes of Objects with Recipe Flows

Keisuke Shirai[1] **Atsushi Hashimoto**[2] **Taichi Nishimura**[1] **Hirotaka Kameko**[1]
Shuhei Kurita[3] **Yoshitaka Ushiku**[2] **Shinsuke Mori**[1]
[1]Kyoto University [2]OMRON SINIC X Corporation [3]RIKEN AIP, JST PRESTO
{shirai.keisuke.64x,nishimura.taichi.43x}@st.kyoto-u.ac.jp
{atsushi.hashimoto,yoshitaka.ushiku}@sinicx.com
{kameko,forest}@i.kyoto-u.ac.jp shuhei.kurita@riken.jp

Abstract

We present a new multimodal dataset called Visual Recipe Flow, which enables us to learn each cooking action result in a recipe text. The dataset consists of object state changes and the workflow of the recipe text. The state change is represented as an image pair, while the workflow is represented as a recipe flow graph (r-FG). The image pairs are grounded in the r-FG, which provides the cross-modal relation. With our dataset, one can try a range of applications, from multimodal commonsense reasoning and procedural text generation.

1 Introduction

Our aim is to track how foods are processed and changed toward the final food product by each cooking action given a recipe text. This requires some knowledge of the actions: what foods and actions are involved and how the action changes them. Skilled chefs can easily imagine these action effects while understanding the required foods. We are interested in building an autonomous agent endowed with this ability, as illustrated in Figure 1. This example involves two cooking actions, and the agent imagines the second action result: the shredded cabbage in the bowl. This also implicates the food requirement: the shredded cabbage produced by the previous action. The prediction for the required foods and action results is indeed a natural ability for humans when they cook something. Thus, this is also crucial for intelligent autonomous agents to understand recipe texts.

Previous work on this line of research provided visual annotation for each cooking instruction (Nishimura et al., 2020; Pan et al., 2020). Nishimura et al. (2020) attached an image with bounding boxes of objects to each instruction, while Pan et al. (2020) split an instruction into sentences and attached frames to each sentence. However, their annotations are often insufficient to predict the action result for each object. A typical

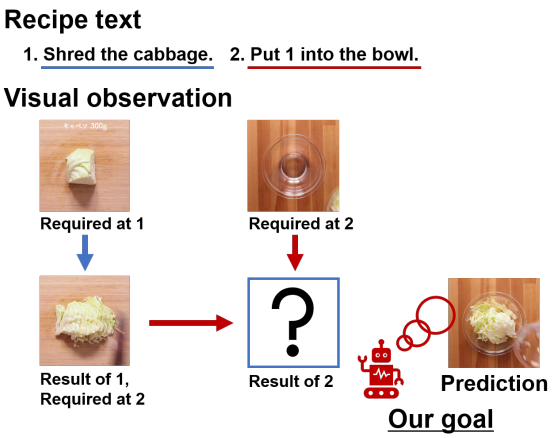

Figure 1: Our goal is to build an agent that tracks object state changes and predicts what observations can be obtained by cooking actions.

case is an instruction in a sentence that directs multiple actions. For example, the instruction of "slice the tomato and put it into the bowl" produces two action results: the sliced tomato and that put in the bowl. Therefore, an instruction-wise visual annotation is insufficient for our task, and action-wise visual annotation is required. Preparing a more dense visual annotation is one straightforward way to handle this case.

Toward the realization of an agent that predicts the result of each action, we introduce a new multimodal dataset called Visual Recipe Flow (VRF). The dataset consists of object[1] state changes caused by every action and the workflow of the text. The change is given as an image pair, while the workflow is given in the format of recipe flow graph (r-FG) (Mori et al., 2014). Each image pair is grounded in the r-FG, which gives the cross-modal relation. Figure 2 shows an example of our dataset.

We focus on recipe text involving various cooking actions, foods, and state changes, which is one of the representatives of procedural texts. Un-

[1]In our work, object refers to food or tool.

Proceedings of the 29th International Conference on Computational Linguistics, pages 3570–3577
October 12–17, 2022.

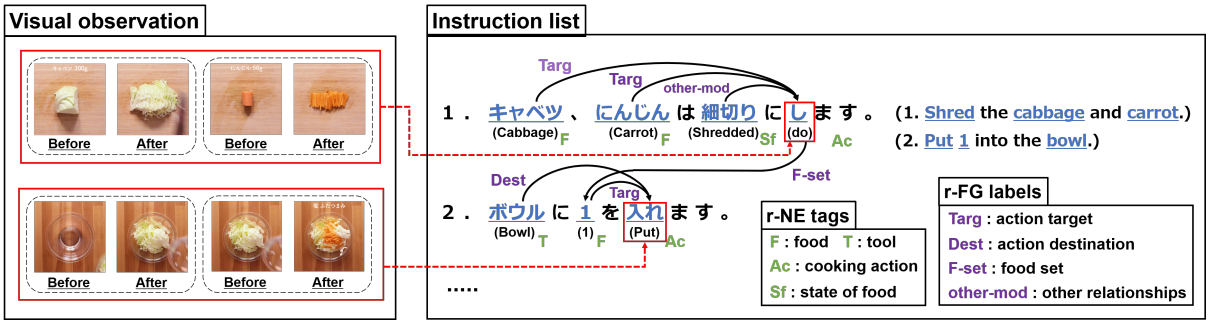

Figure 2: Example of our dataset. A pair of images in the visual observation corresponds to the states of object before and after a cooking action. They are grounded in the action in the instruction list. The black solid arrows denote recipe flows, which describe the relationships between expressions (e.g., cooking actions, foods, and tools).

derstanding these texts by tracking object state changes is one of the recent trends (Dalvi et al., 2018; Bosselut et al., 2018; Tandon et al., 2020; Nishimura et al., 2021; Papadopoulos et al., 2022). Our work also contributes to this line of research. Since images directly express object appearances in the real world (Isola et al., 2015; Zhang et al., 2021), our dataset would provide rich information for the changes. The sequential nature of our dataset can also be used to test the reading comprehension ability of large-scale language models (Srivastava et al., 2022). Furthermore, since our dataset has arbitrary interleaved visual and textual annotations, it is also possible to evaluate the few-shot capability of vision-language models on such data (Alayrac et al., 2022).

2 The VRF dataset

The Visual Recipe Flow (VRF) dataset is a new multimodal dataset. It provides visual annotations for objects in a recipe text before and after a cooking action. We identify expressions including the action in the text by using recipe named entities (r-NEs) (Mori et al., 2014), which can be extended to other procedural tasks. Based on the r-NEs, it also provides a representation of the recipe workflow as a recipe flow graph (r-FG) (Mori et al., 2014). In this section, we first explain the overview of the r-FG and then introduce our visual annotation.

2.1 Recipe flow graph (r-FG)

The r-FG represents the cooking workflow of a recipe text. It consists of a set of recipe flows. The recipe flow is expressed as a directed edge that takes two r-NEs as the starting and ending vertices. It also has a label that describes the relationship between them. It connects one cooking action with the next and expresses its dependencies. For example, in Figure 2, the first action is connected with the second one, which means that the second action requires the products of the first action: shredded cabbage and carrot. This helps us to identify what foods are required for the actions. The annotation has the flows from the ingredient lists to track foods from raw ingredients (Nishimura et al., 2021), which allows us to convert the r-FG into cooking programs (Papadopoulos et al., 2022).

2.2 Visual annotation

Our visual annotation is given as an extension of the r-FG. Each annotation consists of a pair of images which represent object state change by the action. Each image pair is linked with the action in the r-FG. In some cases, a single action can require multiple objects and change their states. Our annotation provides an image pair to all of these state changes. In Figure 2, for example, the first action is linked with two image pairs because it induces the state changes of two objects: cabbage and carrot. This dense annotation would help develop autonomous cooking agents because these images provide visual clues for each action.

3 Annotation standards

In this section, we describe our annotation standards. The annotation consists of three steps in order: (i) r-NE annotation, (ii) r-FG annotation, and (iii) image annotation. Each recipe has an ingredient list, an instruction list, and a cooking video. Figure 3 shows an example of the annotations.

r-NE annotation. First, we annotated words in the ingredient and instruction lists with r-NE tags[2].

[2]We segmented sentences into words beforehand by using a Japanese tokenizer, KyTea (Neubig et al., 2011), because words in a Japanese sentence are not typically separated by whitespace.

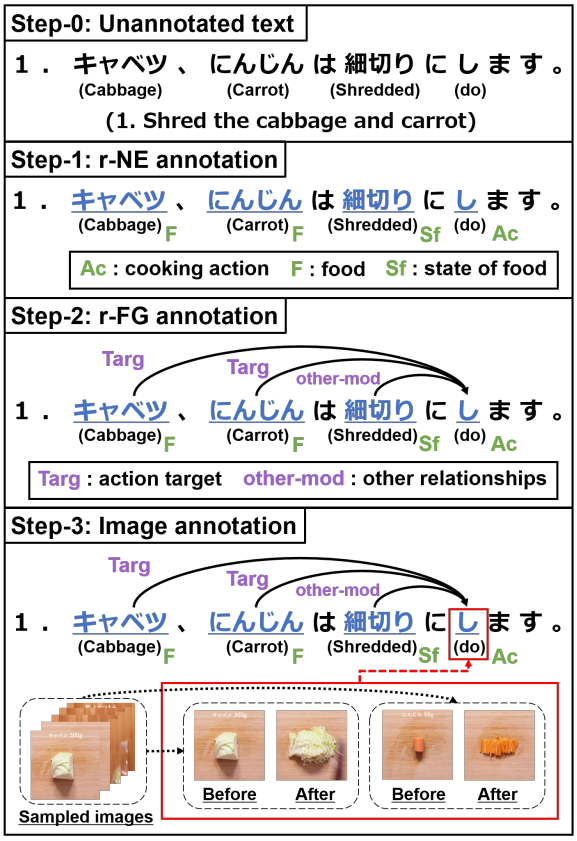

Figure 3: Example of annotation process for a single instruction. The instruction is sequentially annotated with r-NE tags, recipe flows, and images.

i

We used the eight types of r-NE tags, following Mori et al. (2014). See Appendix A for details.

r-FG annotation. Second, we annotated the r-NEs in the first step with the r-FG. We used the 13 types of r-FG labels, following Maeta et al. (2015). See Appendix A for details.

Image annotation. Third, we annotated object states with images, sampled at 3 frames per second from the videos. Each object required for any cooking action is annotated as a pair of frames of states before and after the action. When there are multiple suitable frames, we prioritize the one based on the visual clarity of the object. In some cases, objects are always heavily covered by human hands or abbreviated from the video. We treat them as missing data.

4 Annotation results

This section first describes our annotation process and the statistics for the annotation results. It then investigates the dataset quality and finally assesses our dataset by conducting experiments.

4.1 Annotation process

We started by collecting recipes and cooking videos since the existing r-FG datasets (Mori et al., 2014; Yamakata et al., 2020) are not necessarily associated with the videos. We collected 200 recipes in Japanese and videos from the Kurashiru website[3]. In the video, each cooking process is recorded in detail by a fixed camera. Thus, we can annotate the object states with a fixed viewpoint. Considering the future cooking agent developments, we focused on salad recipes, in which the procedures are simple but still contain 89 unique expressions for cooking action and 275 unique ingredients.

We asked one Japanese annotator, familiar with the r-NE and r-FG, to annotate the recipes. However, filling spreadsheets manually (Mori et al., 2014) is heavy, and it also might cause unexpected annotation errors. Therefore, we developed a web interface to help the annotation. The interface supports all three annotation steps. With this interface, the annotator can annotate recipes with r-NE tags, r-FG labels, and images by simple mouse operations. An illustration of the interface is provided in Appendix B. The whole annotation took 120 hours.

In the annotation collection process, we created annotation guidelines to check annotation errors and reproduce high-quality annotations by another annotator. Starting with a draft, we iteratively revised the guidelines when the first 10, 20, and 50 recipe annotations were finished. In the verification process, we shared the guidelines and three annotation examples with the second annotator.

4.2 Statistics

The recipes contained 1,701 ingredients, 1,077 instructions, and 33,400 words in total. The average number of ingredients and instructions per recipe was 8.51 and 5.31, respectively. The r-NE annotation resulted in 11,686 r-NEs, while the r-FG annotation resulted in 11,291 recipe flows. We provide the detailed statistics for them in Appendix A.

Table 1 shows the statistics for the image annotation results. We annotated 3,705 objects in the r-FGs with images. Among them, 2,551 had both pre-action and post-action images, 485 had only a post-action image, 72 had only a pre-action image, and 597 had no image. In total, 5,659 images (3,824 unique images) were used.

[3] https://www.kurashiru.com, accessed on 2021/12/14.

Annotated image		# objects
Pre-action state	Post-action state	
		597
✓		72
	✓	485
✓	✓	2,551
Total		3,705

Table 1: Statistics for the image annotation results. Objects have image annotation of a pre-action or post-action state if it is checked.

Annotation	Precision	Recall	F-measure
r-NE	97.93	98.88	98.40
r-FG	86.18	86.04	86.11
Image	75.13	70.60	72.80

Table 2: Inter-annotator agreements of the annotations.

4.3 Dataset quality

To investigate the correctness and consistency of the annotation results, we asked another annotator to re-annotate 10 recipes, which were randomly sampled from the collected recipes and contained 623 named entity tags, 616 recipe flows, and 199 visual state changes. We then measured the inter-annotator agreements in precision, recall, and F-measure. The agreements were calculated between the two sets of annotations by taking the first one as the ground truth.

Table 2 lists the results. The F-measure for the r-NE was 98.40, which was almost perfect agreement. The F-measure for the r-FG was 86.11, which was also quite high considering that all the r-NEs were presented as candidate vertices. The F-measure for the images was 72.80, which was smaller than the former steps. However, this was still high, considering that annotation differences in the former steps affected this step.

4.4 Experiments

We conducted multimodal information retrieval experiments to assess our dataset. The experiments aimed to find a correct post-action image from a set of candidate images by using the cooking action verb and pre-action image information. We used a joint embedding model (Miech et al., 2019) and briefly explain the calculation here[4]. We calculated a vector for an estimated post-action object state from the action verb and pre-action image information. This vector is mapped into a shared embedding space. On the other hand, the candidate post-action images are mapped into vectors and mapped them into the embedding space. We searched for the correct post-action image from the estimated post-action state based on their similarities in the embedding space.

[4]See details in Appendix C

Used input		R@5 (↑)	MedR (↓)
Action verb	Pre-action image		
		2.37	149.00
✓		21.24	26.70
	✓	33.77	12.60
✓	✓	37.01	10.40

Table 3: R@5 and MedR for the models with different inputs. The model uses action verb or pre-action image if it is checked. The first line denotes random search.

Our model was trained with different input configurations. We used the Recall@5 (R@5) and the median rank (MedR) as evaluation metrics. Table 3 shows the results. The second and third lines' scores show that the image provides more information than the text. The fourth line's scores imply that the textual and visual modalities provide different information, and using them together is more effective. These results demonstrate that the visual modality provides critical information for finding post-action images. These also indicate the usefulness of our annotation.

5 Application

5.1 Multimodal commonsense reasoning

Multimodal commonsense reasoning in recipe text is one of the recent trends (Yagcioglu et al., 2018; Alikhani et al., 2019). With our dataset, one can try reasoning about the food state changes from a raw ingredient to the final dish with the visual modality (Bosselut et al., 2018; Nishimura et al., 2021). One can also use our dataset for analyzing the cooking action effects throughout a recipe.

5.2 Procedural text generation

Generating procedural text from vision is an important task (Ushiku et al., 2017; Nishimura et al., 2019). To correctly reproduce procedures, the generated instructions should be consistent. The r-FG has the potential to make them more consistent as it represents the flow of the instructions. Since our

recipes are associated with cooking videos, one can use our dataset for that purpose.

6 Conclusion

We have presented a new multimodal dataset called Visual Recipe Flow. The dataset provides dense visual annotations for object states before and after a cooking action. The annotations allows us to learn each cooking action result. Experimental results demonstrated the effectiveness of our annotations for a multimodal information retrieval task. With our dataset, one can also try various applications, including multimodal commonsense reasoning and procedural text generation.

Acknowledgments

We would like to thank anonymous reviewers for their insightful comments. This work was supported by JSPS KAKENHI Grant Number 20H04210, 21H04910, 22H00540, 22K17983 and JST PRESTO Grant Number JPMJPR20C2.

References

Jean-Baptiste Alayrac, Jeff Donahue, Pauline Luc, Antoine Miech, Iain Barr, Yana Hasson, Karel Lenc, Arthur Mensch, Katie Millican, Malcolm Reynolds, et al. 2022. Flamingo: a visual language model for few-shot learning. *arXiv preprint arXiv:2204.14198*.

Malihe Alikhani, Sreyasi Nag Chowdhury, Gerard de Melo, and Matthew Stone. 2019. CITE: A corpus of image-text discourse relations. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 570–575.

Vassileios Balntas, Edgar Riba, Daniel Ponsa, and Krystian Mikolajczyk. 2016. Learning local feature descriptors with triplets and shallow convolutional neural networks. In *Proceedings of the British Machine Vision Conference*, pages 119.1–119.11.

Antoine Bosselut, Omer Levy, Ari Holtzman, Corin Ennis, Dieter Fox, and Yejin Choi. 2018. Simulating action dynamics with neural process networks. In *Proceedings of the 6th International Conference on Learning Representations*.

Bhavana Dalvi, Lifu Huang, Niket Tandon, Wen-tau Yih, and Peter Clark. 2018. Tracking state changes in procedural text: a challenge dataset and models for process paragraph comprehension. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 1595–1604.

Alex Graves and Jürgen Schmidhuber. 2005. Framewise phoneme classification with bidirectional lstm and other neural network architectures. *Neural networks*, 18(5-6):602–610.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. 2016. Deep residual learning for image recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 770–778.

Phillip Isola, Joseph J. Lim, and Edward H. Adelson. 2015. Discovering states and transformations in image collections. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*.

Ilya Loshchilov and Frank Hutter. 2019. Decoupled weight decay regularization. In *Proceedings of the 7th International Conference on Learning Representations*.

Hirokuni Maeta, Tetsuro Sasada, and Shinsuke Mori. 2015. A framework for procedural text understanding. In *Proceedings of the 14th International Conference on Parsing Technologies*, pages 50–60.

Antoine Miech, Ivan Laptev, and Josef Sivic. 2018. Learning a text-video embedding from incomplete and heterogeneous data. *arXiv preprint arXiv:1804.02516*.

Antoine Miech, Dimitri Zhukov, Jean-Baptiste Alayrac, Makarand Tapaswi, Ivan Laptev, and Josef Sivic. 2019. Howto100m: Learning a text-video embedding by watching hundred million narrated video clips. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 2630–2640.

Shinsuke Mori, Hirokuni Maeta, Yoko Yamakata, and Tetsuro Sasada. 2014. Flow graph corpus from recipe texts. In *Proceedings of the Ninth International Conference on Language Resources and Evaluation*, pages 2370–2377.

Graham Neubig, Yosuke Nakata, and Shinsuke Mori. 2011. Pointwise prediction for robust, adaptable japanese morphological analysis. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, pages 529–533.

Taichi Nishimura, Atsushi Hashimoto, and Shinsuke Mori. 2019. Procedural text generation from a photo sequence. In *Proceedings of the 12th International Conference on Natural Language Generation*, pages 409–414.

Taichi Nishimura, Atsushi Hashimoto, Yoshitaka Ushiku, Hirotaka Kameko, and Shinsuke Mori. 2021. *State-Aware Video Procedural Captioning*, page 1766–1774. Association for Computing Machinery, New York, NY, USA.

Taichi Nishimura, Suzushi Tomori, Hayato Hashimoto, Atsushi Hashimoto, Yoko Yamakata, Jun Harashima,

Yoshitaka Ushiku, and Shinsuke Mori. 2020. Visual grounding annotation of recipe flow graph. In *Proceedings of the 12th Language Resources and Evaluation Conference*, pages 4275–4284.

Liang-Ming Pan, Jingjing Chen, Jianlong Wu, Shaoteng Liu, Chong-Wah Ngo, Min-Yen Kan, Yugang Jiang, and Tat-Seng Chua. 2020. *Multi-Modal Cooking Workflow Construction for Food Recipes*, page 1132–1141. Association for Computing Machinery.

Dim P Papadopoulos, Enrique Mora, Nadiia Chepurko, Kuan Wei Huang, Ferda Ofli, and Antonio Torralba. 2022. Learning program representations for food images and cooking recipes. *arXiv preprint arXiv:2203.16071*.

Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael Bernstein, et al. 2015. Imagenet large scale visual recognition challenge. *International journal of computer vision*, 115(3):211–252.

Aarohi Srivastava, Abhinav Rastogi, Abhishek Rao, Abu Awal Md Shoeb, Abubakar Abid, Adam Fisch, Adam R Brown, Adam Santoro, Aditya Gupta, Adrià Garriga-Alonso, et al. 2022. Beyond the imitation game: Quantifying and extrapolating the capabilities of language models. *arXiv preprint arXiv:2206.04615*.

Niket Tandon, Keisuke Sakaguchi, Bhavana Dalvi, Dheeraj Rajagopal, Peter Clark, Michal Guerquin, Kyle Richardson, and Eduard Hovy. 2020. A dataset for tracking entities in open domain procedural text. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 6408–6417.

Atsushi Ushiku, Hayato Hashimoto, Atsushi Hashimoto, and Shinsuke Mori. 2017. Procedural text generation from an execution video. In *Proceedings of the Eighth International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 326–335.

Semih Yagcioglu, Aykut Erdem, Erkut Erdem, and Nazli Ikizler-Cinbis. 2018. RecipeQA: A challenge dataset for multimodal comprehension of cooking recipes. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 1358–1368.

Yoko Yamakata, Shinsuke Mori, and John A Carroll. 2020. English recipe flow graph corpus. In *Proceedings of the 12th Language Resources and Evaluation Conference*, pages 5187–5194.

Yixin Zhang, Yoko Yamakata, and Keishi Tajima. 2021. Mirecipe: A recipe dataset for stage-aware recognition of changes in appearance of ingredients. In *Proceedings of the 3rd ACM International Conference on Multimedia in Asia*, pages 1–7. Association for Computing Machinery.

Tag	Meaning	# tags
F	Food	5,098
T	Tool	758
D	Duration	129
Q	Quantity	1,778
Ac	Action by chef (cooking action)	2,532
Af	Action by food	353
Sf	State of food	971
St	State of tool	67
Total	—	11,686

Table 4: r-NE tags and the number of annotated tags of each type.

Label	Meaning	# labels
Agent	Action agent	330
Targ	Action target	2,961
Dest	Action destination	1,025
T-comp	Tool complement	157
F-comp	Food complement	20
F-eq	Food equality	2,397
F-part-of	Food part-of	330
F-set	Food set	987
T-eq	Tool equality	4
T-part-of	Tool part-of	0
A-eq	Action equality	1
V-tm	Head of clause for timing	112
other-mod	Other relationships	2,967
Total	—	11,291

Table 5: r-FG labels and the number of annotated labels of each type.

A Detailed statistics for the textual annotation

This section provides the detailed statistics for the annotated r-NE tags and r-FG labels.

A.1 r-NE tags

Table 4 shows the statistics for the annotated r-NE tags with the explanation of each tag. Among the tags, Ac, F, and T are specially important in our work. Ac denotes human cooking action, which is distinguished from action by food (Af). For example, in the instruction of "leave the salad to cool," "leave" is tagged with Ac, while "cool" is tagged with Af. F denotes foods including raw ingredients, intermediate products after cooking action, and the final dish. T denotes tools used for cooking. In our work, objects refer to the foods or tools. Our image annotation targeted the states of these objects.

A.2 r-FG labels

Table 5 shows the statistics for the annotated r-FG labels with the explanation of each tag. The cooking action (Ac) requires the objects (F or T). Targ describes this relationship taking the action and object as the starting and ending vertices, respectively. During the image annotation, we identified the required objects by using the flows labeled with Targ.

B Web interface

Our developed web interface is illustrated in Figure 4. In the first step (r-NE annotation), the annotator can annotate words in the ingredient and instruction lists with an r-NE tag by clicking the words and tag. In the second step (r-FG annotation), the annotator can annotate the r-NEs with a recipe flow by clicking starting and ending vertices and a label for them. In the final step (image annotation), the annotator can annotate the pre-action and post-action object states with images by clicking a frame and the button for the state. All objects for annotation are automatically prepared by tracing the recipe flows.

C A joint embedding model

In this section, we provide the detailed calculation of our model and experimental settings.

C.1 Model description

We first calculate a vector for an estimated post-action object state baesd on an action verb a, an object information o, and a pre-action image i_{pre}. The object is obtained by tracing a recipe flow labeled with Targ. a and o are converted to d_t-dimensional vectors h_a and h_o, respectively, by first embedding words into d_v-dimensional representations via a lookup table and then encoding them into d_t-dimensional vectors by using a bidirectional LSTM (BiLSTM) (Graves and Schmidhuber, 2005). For i_{pre}, we extract its feature $h_i^{pre} \in \mathbb{R}^{d_i}$ by using a pre-trained convolutional neural network (CNN) and transform it into $\hat{h}_i^{pre} \in \mathbb{R}^{d_t}$ as follows:

$$\hat{h}_i^{pre} = W_1^T h_i^{pre} + b_1^T, \quad (1)$$

where $W_1^T \in \mathbb{R}^{d_t \times d_i}$ and $b_1^T \in \mathbb{R}^{d_t}$ are learnable parameters. Given these fixed-size vectors, we then compute the vector for the estimated post-action object state $\hat{h}_o$ as:

$$\hat{h}_o = W_3^T(\text{ReLU}(W_2^T[h_a; h_o; h_i^{pre}] + b_2^T)) + b_3^T, \quad (2)$$

where ; denotes concatenation, and $W_2^T \in \mathbb{R}^{3d_t \times 3d_t}$, $W_3^T \in \mathbb{R}^{d_t \times 3d_t}$, $b_2^T \in \mathbb{R}^{3d_t}$, and $b_3^T \in \mathbb{R}^{d_t}$ are learnable parameters. $\hat{h}_o$ is then mapped to the joint embedding space as:

$$\begin{aligned} h_t &= (W_4^T \hat{h}_o + b_4^T) \circ \\ &\quad \sigma(W_5^T(W_4^T \hat{h}_o + b_4^T) + b_5^T), \quad (3) \\ \tilde{h}_t &= \frac{h_t}{||h_t||_2}, \quad (4) \end{aligned}$$

where $W_4^T \in \mathbb{R}^{d_e \times d_t}$, $W_5^T \in \mathbb{R}^{d_e \times d_e}$, $b_4^T, b_5^T \in \mathbb{R}^{d_e}$ are learnable parameters.

The post-action image i_{post} is fed to the pre-trained CNN to extract its feature $h_i^{post} \in \mathbb{R}^{d_i}$. Based on this feature, we compute $\hat{h}_i$ as:

$$\hat{h}_i = W_2^I(\text{ReLU}(W_1^I h_i^{post} + b_1^I)) + b_2^I, \quad (5)$$

where $W_1^I, W_2^I \in \mathbb{R}^{d_i \times d_i}$, and $b_1^I, b_2^I \in \mathbb{R}^{d_i}$ are learnable parameters. Following Miech et al. (2018), the feature vector $\hat{h}_i$ is then mapped to the joint embedding space as follows:

$$\begin{aligned} h_v &= (W_3^I \hat{h}_i + b_3^I) \circ \\ &\quad \sigma(W_4^I(W_3^I \hat{h}_i + b_3^I) + b_4^I), \quad (6) \\ \tilde{h}_v &= \frac{h_v}{||h_v||_2}, \quad (7) \end{aligned}$$

where σ is the sigmoid function, $\circ$ denotes the element-wise multiplication, $W_3^I \in \mathbb{R}^{d_e \times d_i}$, $W_4^I \in \mathbb{R}^{d_e \times d_e}$, and $b_3^I, b_4^I \in \mathbb{R}^{d_e}$ are learnable parameters.

Loss function. After mapping the inputs to the joint embedding space, we calculate the distance between these vectors as:

$$D(\tilde{h}_t, \tilde{h}_v) = ||\tilde{h}_t - \tilde{h}_v||_2. \quad (8)$$

Given n examples of $((\tilde{h}_{t,1}, \tilde{h}_{v,1}), \cdots, (\tilde{h}_{t,n}, \tilde{h}_{v,n}))$, we minimize the following triplet loss (Balntas et al., 2016):

$$\begin{aligned} \mathcal{L} = \sum_{i=1}^{n} &\{\max(D_{i,i} - D_{i,j} + \delta, 0) \\ &+ \max(D_{i,i} - D_{k,i} + \delta, 0)\}, \quad (9) \end{aligned}$$

where $D_{i,j} = D(\tilde{h}_{t,i}, \tilde{h}_{v,j})$, and δ denotes a margin. In Equation (9), $D_{i,i}$ is the distance for a positive pair, and $D_{i,j}$ and $D_{k,i}$ are the distances for pairs with negative text and image feature vectors, respectively. For negative sampling, we simply sample negative examples from a mini-batch.

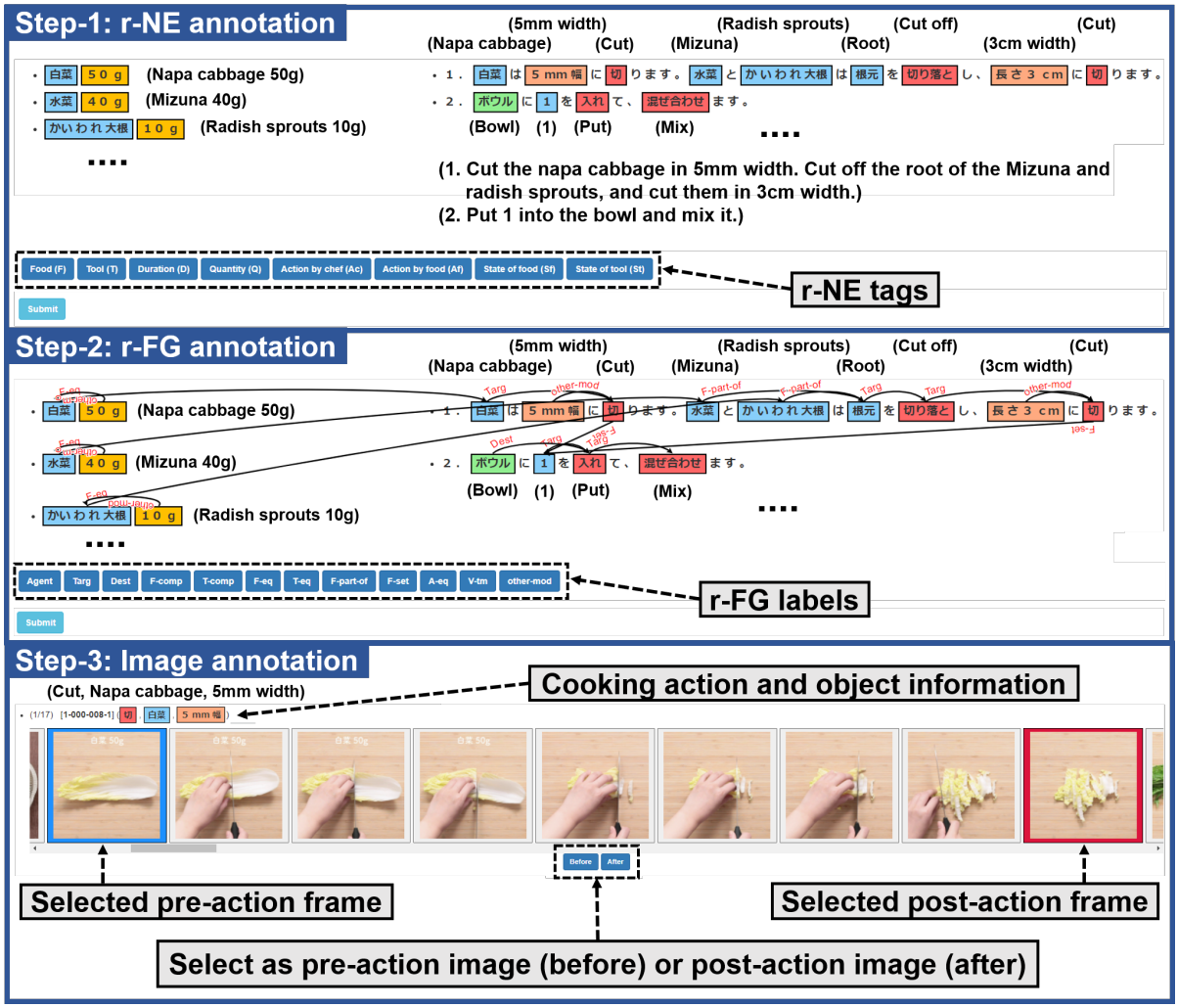

Figure 4: Our web annotation interface. The annotator can complete annotations only by mouse operations. The web page is written in Japanese.

C.2 Settings

Model parameters. We used a 1-layer 256-dimensional BiLSTM to encode words. We set the dimensions as $(d_v, d_t, d_i, d_e) = (496, 512, 2048, 128)$. We used ResNet-152 (He et al., 2016), which was pre-trained on ImageNet (Russakovsky et al., 2015), to extract a feature vector of 2048 dimensions from an image.

Optimization. We used AdamW (Loshchilov and Hutter, 2019) with an initial learning rate of 1.0×10^{-5} to tune the parameters. During training, we froze only the parameters of the CNN. Each model was trained for 350 epochs, and we created a mini-batch with 4 recipes at each step. We set δ in Equation (9) to 0.1. We evaluated the model performance through 10-fold cross-validation by splitting the dataset into 90% for training and 10% for testing.

IMPARA: Impact-Based Metric for GEC Using Parallel Data

Koki Maeda and **Masahiro Kaneko** and **Naoaki Okazaki**
Tokyo Institute of Technology
{koki.maeda@nlp., masahiro.kaneko@nlp, okazaki@}.c.titech.ac.jp

Abstract

Automatic evaluation of grammatical error correction (GEC) is essential in developing useful GEC systems. Existing methods for automatic evaluation require multiple reference sentences or manual scores. However, such resources are expensive, thereby hindering automatic evaluation for various domains and correction styles. This paper proposes an **I**mpact-based **M**etric for GEC using **PARA**llel data, IMPARA, which utilizes correction impacts computed by parallel data comprising pairs of grammatical/ungrammatical sentences. As parallel data is cheaper than manually assessing evaluation scores, IMPARA can reduce the cost of data creation for automatic evaluation. Correlations between IMPARA and human scores indicate that IMPARA is comparable or better than existing evaluation methods. Furthermore, we find that IMPARA can perform evaluations that fit different domains and correction styles trained on various parallel data.

1 Introduction

Grammatical error correction (GEC) is a task of correcting grammatically incorrect sentences (Yuan and Briscoe, 2016; Chollampatt and Ng, 2018a; Junczys-Dowmunt et al., 2018; Kaneko et al., 2020, 2022; Omelianchuk et al., 2020). A GEC system is designed to be applicable in various domains, such as web text (Flachs et al., 2020) and essays written by language learners (Yannakoudakis et al., 2011) and in different correction styles, such as minimal and fluency edits (Ng et al., 2013; Napoles et al., 2017; Hotate et al., 2019). Idealy, GEC models should be evaluated by manually assessing the quality of corrections made by these models for certain target domains and styles. However, it is expensive to perform manual evaluation every time a GEC model outputs a correction; we thus need to establish an automatic evaluation method that correlates well with manual assessments.

Method	Fine-tuning data	Domain dependence
Based on language models	None	No
With manual assessments	P and M	on P and M
This work (IMPARA)	P	on P

Table 1: Comparison of reference-less methods (P: parallel data; M: manual assessment data). As discussed in Section 3, the method that is based only on language models underperforms the others. IMPARA achieves a comparable or better performance than the method trained on manual assessments, although IMPARA does not depend on manual assessment data.

Automatic evaluation methods of GEC are categorized into two. The first category is reference-based methods (Dahlmeier and Ng, 2012; Napoles et al., 2015; Bryant et al., 2017) that evaluate the closeness of output sentences from a GEC system and human-written reference sentences. In general, an ungrammatical sentence can be corrected in different ways. Thus, reference-based methods require multiple reference sentences for the accurate evaluation. However, Choshen and Abend (2018b) argued that it is unrealistic to prepare reference sentences that cover all possible corrections. In addition, they showed that low-coverage reference sets deteriorate the reliability of evaluation.

The second category is reference-less method, which uses only input sentences and system outputs (see Table 1 for comparison). Several studies applied language models for automatic evaluation (Napoles et al., 2016; Flachs et al., 2020; Islam and Magnani, 2021). However these studies had low correlations with human judgements because the perplexity of language models does not necesarily reflect the grammaticality but the frequency of words. Therefore, Asano et al. (2017) and Yoshimura et al. (2020) proposed using human assessment scores for training automatic evaluation models and hence reported performance improvements. However, applying their methods to various domains and styles is expensive because

Proceedings of the 29th International Conference on Computational Linguistics, pages 3578–3588
October 12–17, 2022.

they require a dataset of human assessment for each domain/style and because the availability of such datasets is limited. In addition, it is difficult to create a reliable dataset for human assessment (Choshen and Abend, 2018a).

We propose a novel reference-less method called **I**mpact-based **M**etric for GEC using **PARA**llel data (**IMPARA**)[1]. This method can be trained using only parallel data comprising ungrammatical and grammatical sentence pairs, which are widely available in various domains and correction styles. Furthermore, creating parallel data is less expensive than manually rating GEC outputs. Therefore, we can use IMPARA for various domains/styles with much less effort than performing manual assessments.

The simplest way to build a model for correction quality judgement is learning to discriminate between original and grammatical sentences in parallel data. However, this approach has two problems. First, an automatic evaluation method receives a pair of original sentence and *GEC output*, whereas the parallel data only includes pairs of original and *grammatical* sentences. Therefore, the discriminator may receive a pair of two ungrammatical sentences during inference, although it is trained only with pairs of ungrammatical and grammatical sentences. Second, an automatic evaluation method must handle incomplete corrections made by GEC methods. Even if an original sentence includes multiple grammatical errors, a GEC method may correct some of the errors and leave others. Hence, the supervision data for training an automatic evaluation method should include instances where grammatical errors are partially corrected.

IMPARA addresses these issues by automatic generation of supervision data with partially-corrected sentences. Decomposing corrections between original and corrected sentences into *edits*, we measure the *impact* of each edit to determine the preferences of edits. Then, we generate pairs of partially corrected sentences from parallel data and determine the preference order of generated pairs based on the impacts of the involved edits. The evaluation model is trained on the generated pairs so that it reproduces the preference order of sentence pairs.

The meta-evaluation (Section 3) demonstrates that IMPARA achieves a comparable or better evaluation performance than existing reference-less methods, even without tailored data of human assessments, but only with parallel data. Furthermore, IMPARA exhibits high capability of adapting its evaluation metric to the target domain and style given by the parallel data.

2 IMPARA

2.1 Architecture

IMPARA comprises quality estimator (QE) and similarity estimator (SE) based on BERT (Devlin et al., 2019), as illustrated in Figure 1. Given a pair of original sentence and GEC output, the QE evaluates the quailty of the GEC output. Then, the SE computes the semantic similarity of two sentences. We use a pre-trained BERT model without fine-tuning to build the SE, but fine-tune the BERT model for the QE.

Let X and Y denote an original sentence and an output of GEC system, respectively. Given a GEC output Y, the QE yields a score $\mathrm{QE}(Y) \in [0, 1]$. Given X and Y, the SE computes the similarity $\mathrm{SE}(X, Y) \in [0, 1]$ of the sentences. Equation 1 defines the overall score of the correction from X to Y,

$$\mathrm{score}(X, Y) = \begin{cases} \mathrm{QE}(Y) & (\text{if } \mathrm{SE}(X, Y) > \theta) \\ 0 & (\text{otherwise}) \end{cases}. \quad (1)$$

Here, θ is a threshold to output the score of the QE; the score is $\mathrm{SE}(X, Y)$ if the semantic similarity is higher than θ and 0 otherwise. The threshold prevents the high overall score even when output Y is deviated from original sentence X.

This study aims to build a *relative* evaluation measure that can compare two sentences in terms of the quality of grammatical correction[2]. We are not interested in building an absolute measure, unlike other metrics, e.g., recall and precision used in M^2 (Dahlmeier and Ng, 2012). Adjusting $\mathrm{score}(X, Y)$ with manual assessment scores may be possible. However, we leave this as a future work because our focus is to learn an evaluation measure only from parallel data.

2.2 Quality estimator

The QE computes a score as a dot product between a parameter vector and contextualized embeddings from the BERT model, followed by the sigmoid

[1] https://github.com/Silviase/IMPARA

[2] Therefore, $\mathrm{score}(X, Y) = 0.99$ does not mean that 99% of errors in X are corrected in Y.

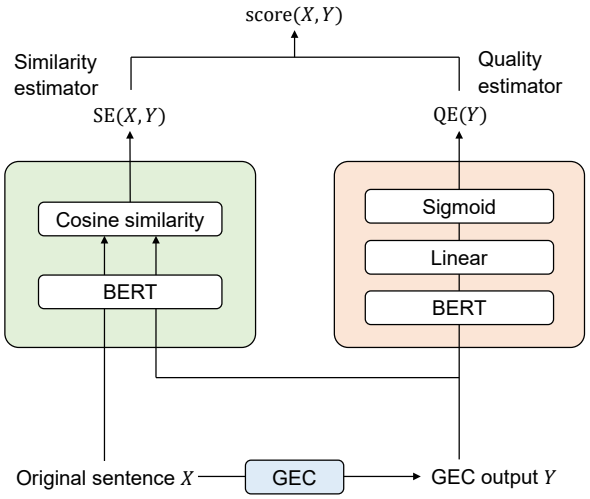

Figure 1: The IMPARA model architecture

function σ. Formally, let $\mathbf{y} \in \mathbb{R}^d$ denote the contextualized embeddings of the first token of the output sentence Y at the final layer of the BERT model (where d is the number of dimensions of embeddings). We define $q(Y) \in \mathbb{R}$ as a dot product between parameter vector $\mathbf{w} \in \mathbb{R}^d$ and embeddings $\mathbf{y}$ of sentence Y,

$$q(Y) = \mathbf{w}^\top \mathbf{y}. \tag{2}$$

Then we compute a QE score

$$\mathrm{QE}(Y) = \sigma(q(Y)). \tag{3}$$

We train the parameter vector $\mathbf{w}$ by fine-tuning the BERT model on the supervision data $\mathcal{T}$ (to be described in Section 2.4). Supervision data comprises pairs of positive S_+ and negative S_- sentences. We train the model so that it prefers positive sentences to negative ones by minimizing loss function $\mathfrak{L}$.

$$\mathfrak{L} = \frac{1}{|\mathcal{T}|} \sum_{(S_-, S_+) \in \mathcal{T}} \sigma\left(q(S_-) - q(S_+)\right) \tag{4}$$

Here, we use the sigmoid function $\sigma(.)$ to stabilize training[3].

2.3 Impact of edits

Before describing the procedure for generating the supervision data $\mathcal{T}$, we introduce the notion of an *edit* and its *impact* in grammatical error correction.

Let (S, T) be a pair of sentences from parallel data and $\mathcal{E}$ be a set of edits to obtain the corrected sentence T from the original one S. Here, edits are extracted automatically by using ERRANT (Bryant et al., 2017), a tool for aligning two sentences and extracting edits and their types.

For an edit $e \in \mathcal{E}$, T_{-e} denotes the sentence with all edits except for e, i.e., $\mathcal{E} \setminus \{e\}$, applied to S. In other words, T_{-e} presents the sentence where edit e is omitted when rewriting S into T. Hence, T is obtained by applying edit e to T_{-e}. We evaluate the impact of edit e in terms of the magnitude of the semantic change from T_{-e} to T,

$$\mathrm{impact}(T, e) = 1 - \frac{\mathrm{BERT}(T) \cdot \mathrm{BERT}(T_{-e})}{\|\mathrm{BERT}(T)\| \|\mathrm{BERT}(T_{-e})\|}. \tag{5}$$

Here, $\mathrm{BERT}(T) \in \mathbb{R}^d$ presents the mean pooling of contextualized embeddings of all tokens in sentence T at the final layer of the BERT model. Equation 5 provides a higher impact to an edit that greatly changes the meaning between T_{-e} and T. Given a set of edits E, we define its overall impact $I(T, E)$ as the sum of impacts of all edits,

$$I(T, E) = \sum_{e \in E} \mathrm{impact}(T, e). \tag{6}$$

Choshen and Abend (2018a) also proposed the concept of computing an impact of an edit. However, they define an impact as the number of applied edits without considering the impact of an individual edit. Conversely, we assume that a sentence with more semantically important corrections should receive a higher impact than others with semantically unimportant corrections.

2.4 Generating supervision data for QE

As detailed in Equation 4, the QE model requires supervision data $\mathcal{T}$ with pairs of positive and negative sentences. We cannot use parallel data of original and corrected sentences as they are because the QE model needs to measure the quality of imperfect corrections. In addition, in some parallel data with fluency corrections (e.g., JFLEG (Napoles et al., 2017)), the difference from an original to its corrected sentence is so large that the QE model may not learn the importance of each individual edit in parallel sentences.

Therefore, we generate partially corrected sentences from the sentence pairs of parallel data and determine the preference order between two sentences. Figure 2 depicts the generation process for a pair of the original S and corrected T sentences in the parallel data. First, we apply ERRANT (Bryant

[3]Preliminary experiments confirmed that the sigmoid function contirubted to improve the evaluation performance.

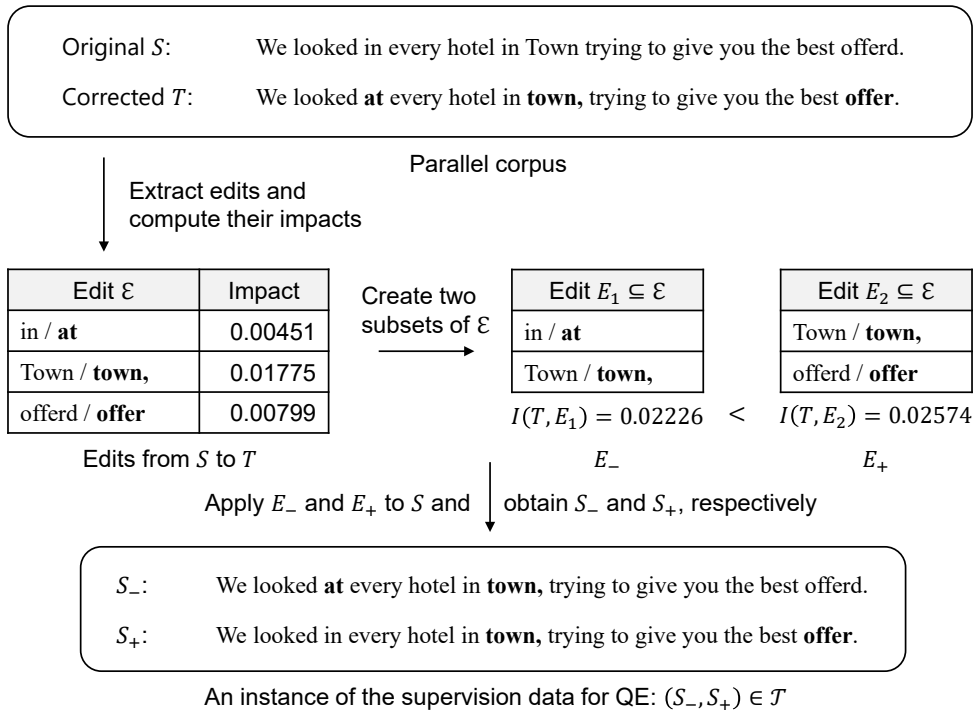

Figure 2: Procedure for generating supervision data for QE. We generate partially corrected sentences from a pair of original and corrected sentences in the parallel data, and determine their preference orders.

et al., 2017) to automatically extract a set of edits $\mathcal{E}$ between the original sentence S and corrected sentence T in the parallel data. We generate two subsets of edits $E_1, E_2 \subseteq \mathcal{E}$ using the procedure presented in the subsequent paragraph. We choose E_- and E_+ from the two subsets E_1 and E_2 such that $I(T, E_-) < I(T, E_+)$,

$$E_- = \operatorname*{argmin}_{E \in \{E_1, E_2\}} I(T, E), \quad (7)$$

$$E_+ = \operatorname*{argmax}_{E \in \{E_1, E_2\}} I(T, E). \quad (8)$$

Finally, we obtain two sentences S_- and S_+ by applying E_- and E_+, respectively, to the original sentence S. In this way, the supervision data $\mathcal{T}$ provides tuples of (S_-, S_+) where the sentence S_+ has a higher impact than S_- measured by the impacts of edits (Equation 6).

At last, we describe the procedure to obtain E_1 and E_2 from $\mathcal{E}$. We randomly create a subset $E_1 \subseteq \mathcal{E}$ with k elements, where $k \in \{1, 2, \ldots, |\mathcal{E}|\}$ is chosen from the discrete uniform distribution. Then, we modify E_1 to obtain another subset $E_2 \subseteq \mathcal{E}$. Initializing $E_2 = E_1$, we perform the following operation for each element $e \in \mathcal{E}$ with the probability $\frac{1}{|\mathcal{E}|}$:

$$E_2 \leftarrow \begin{cases} E_2 \cup \{e\} & \text{if } e \notin E_1 \wedge e \notin E_2 \\ E_2 \setminus \{e\} & \text{if } e \in E_1 \wedge e \in E_2 \end{cases}. \quad (9)$$

We discard E_1 and E_2 if $E_1 = E_2$ even after repeating the operation. Consequently, we randomly sample partial edits from $\mathcal{E}$ and enhance the diversity of partially corrected sentences.

2.5 Similarity estimator

The SE computes the semantic similarity between the original sentence X and its GEC output Y. Specifically, we compute the cosine similarity between contextual embeddings of two sentences X and Y,

$$\mathrm{SE}(X, Y) = \frac{\mathrm{BERT}(X) \cdot \mathrm{BERT}(Y)}{\|\mathrm{BERT}(X)\| \|\mathrm{BERT}(Y)\|}. \quad (10)$$

Here, the definition of $\mathrm{BERT}(.)$ is compatible with that for Equation 5.

3 Experiments

We conducted two types of experiments to examine the performance of IMPARA over existing methods. First, we compared correlations between automatic evaluation metrics and human assessments in GEC; this experiment measures the closeness of automatic and human evaluations. Second, we evaluate automatic evaluation metrics trained and tested in different text domains and correction styles; this experiment investigates the importance to train evaluation metric in a target domain/style and the

ability of IMPARA to adapt to different domains and styles.

3.1 Experimental settings

3.1.1 Hyperparameters

We used `BERT-BASE-CASED`[4] on HuggingFace as the pre-trained BERT model for the SE and QE. We kept the number of training instances $|\mathcal{T}|$ constant to 4096, regardless of test sets, to avoid the effects of the size of training data on the performance. The maximum number of training instances generated from a parallel sentence pair is 30. We set the learning rate to 10^{-5} and the batch size to 32. A dataset is split into training, validation, and test sets at a ratio of $8:1:1$. We selected the number of epochs from $1, 2, ..., 10$ that showed the best performance on the validation set. The threshold θ is set to 0.9 for the SE. This similarity value is higher than the maximum similarity value computed for any combinations of corrected sentences in the CoNLL-2014 dataset.

3.1.2 Baselines

Baseline methods include two reference-less automatic evaluation methods with different architectures: SOME (Yoshimura et al., 2020) and Scribendi Score (Islam and Magnani, 2021).

SOME employs BERT models optimized on human assessments for corrections and estimates correction quality from three perspectives: grammaticality, fluency, and meaning preservation. Models were trained on `tmu_gfm_dataset`[5], where five human raters manually assign evaluation scores for sentence pairs in the CoNLL-2013 dataset, with the hyperparameter settings of Yoshimura et al. (2020). Note that SOME was trained with additional human assessments for the CoNLL-2013 dataset, whereas IMPARA was trained only on the parallel data of the CoNLL-2013 dataset. Scribendi Score uses a language model to determine whether a correction improves the quality of a sentence. It also performs a superficial comparison of sentences to determine whether a correction is appropriate. We employed the pre-trained model `gpt-2` released by HuggingFace and `fuzzywuzzy`[6], a publicly available Python package to calculate the token sort ratio and Levenshtein distance ratio for sentence comparison.

3.1.3 Datasets and meta-evaluation metrics

The first experiment measured correlations between automatic evaluation metrics and human assessments on the dataset presented by Grundkiewicz et al. (2015). This dataset contains human rankings created for the outputs of 12 GEC systems on the CoNLL-2014 dataset. We computed Pearson's correlation coefficient (Pea) and Spearman's rank correlation coefficient (Spe) at the corpus level using the Expected Wins shown in Table 3(b) of Grundkiewicz et al. (2015). We also measured accuracy (Acc) and Kendall's rank correlation coefficient (Ken) for sentence-level comparison.

Meanwhile, Choshen and Abend (2018a) proposed MAEGE, an automatic methodology for GEC metric validation. MAEGE generates multiple partially corrected sentences from an incorrect sentence, and assigns pseudo scores to them, which are based on the number of edits applied. Then it computes correlation coefficients between the pseudo scores and the scores computed by automatic evaluation metrics. As this method do not require human assessment, it overcame difficulties such as low inter-rater agreement on human rankings. We calculated five correlations coefficients for meta-evaluation: Pearson's correlation coefficient (Pea) and Spearman's rank correlation coefficient (Spe) at corpus and sentence levels; and Kendall's rank correlation coefficient (Ken) at the chain level.

For the second experiment, we conducted meta-evalution using MAEGE on four different corpora: CoNLL-2013 dataset (Ng et al., 2013) (minimal edits), JFLEG (Napoles et al., 2017) (fluency edits), CWEB (Flachs et al., 2020) (website texts) and FCE (Yannakoudakis et al., 2011) (essay). We meta-evaluated IMPARA with different combinations of training and test sets to examine whether IMPARA can incorporate the characteristics of a dataset in the GEC evaluation. In this evaluation, we split each dataset into training and test sets at a ratio of $9:1$. In addition, we compared IMPARA with the two baseline methods in terms of domain adaptability on the four datasets.

3.2 Correlations with human assessments

Table 2 shows correlations between automatic evaluation metrics and human rankings. As we could not reproduce the scores reported in Islam and Mag-

[4] https://github.com/huggingface/transformers

[5] https://huggingface.co/datasets/tmu_gfm_dataset

[6] https://pypi.org/project/fuzzywuzzy

Method	Corpus		Sentence	
	Pea	Spe	Acc	Ken
Scribendi Score (paper)	0.951	0.940	-	-
Scribendi Score (ours)	0.303	0.729	0.414	-0.170
SOME	0.956	0.923	**0.777**	**0.555**
IMPARA	**0.974**	**0.934**	0.748	0.496
IMPARA (parallel only)	0.936	0.929	0.742	0.485

Table 2: Correlation with manual evaluation (Grundkiewicz et al., 2015) on CoNLL-2014.

Method	Corpus		Sentence		Chain
	Pea	Spe	Pea	Spe	Ken
Scribendi Score	0.884	0.981	0.374	0.421	**0.824**
SOME	**0.965**	**1.000**	0.394	0.439	0.563
IMPARA	0.951	0.990	**0.522**	**0.608**	0.692

Table 3: Meta-evaluation result using MAEGE on CoNLL-2014.

nani (2021), the first two rows presents the scores reported in their paper ("paper") and reproduced by our implementation ("ours").

IMPARA and SOME are the contenders in this evaluation; IMPARA was better than SOME at the corpus level, but inferior at the sentence level. It should be noted that IMPARA achieves the comparable performance to SOME without additional human assessments on the CoNLL-2013 dataset[7]. We also show IMPARA trained without the supervision data described in Section 2.4, but only on the parallel data ("parallel only" in Table 2). We observed the improvement by generating supervision data, comparing the last two rows in Table 2.

Table 3 reports the meta-evaluation results using MAEGE. As Choshen and Abend (2018a) suggested, we regarded this evaluation more important than the evaluation of Table 2. Again, IMPARA exhibits a comparable performance to SOME in this evaluation. Although IMPARA is slightly inferior to SOME at the corpus level, the correlation coefficients are quite high (≈ 1) to compare the two methods. Therefore, we focused on the evaluation at sentence and chain levels, which is more fine-grained than that at the sentence level. At the sentence and chain levels, IMPARA outperformed SOME with wide margins, +0.128 point (Pea), +0.169 point (Spe), and +0.129 point (Ken)[8].

[7]Again, we emphasize that SOME uses additional supervision data `tmu_gfm_dataset` for training the model.

[8]Scribendi Score overwhelmes other metrics in Ken, but this is because Scribendi Score assigns tie scores to many instances. Although such instances are *difficult* to decide preference orders, the MAEGE implementation excludes tie instances from the evaluation. In other words, MAEGE did not evaluate difficult instances for Scribendi Score. For reference, the ratios of tie instances are 42.5% (Scribendi Score), 0.04% (SOME), and 0.07% (IMPARA).

Dataset (eval)	Dataset (train)	Corpus		Sentence		Chain
		Pea	Spe	Pea	Spe	Ken
CoNLL-2013	CoNLL-2013	0.932	**1.000**	**0.411**	**0.515**	**0.688**
	CWEB	0.961	**1.000**	0.380	0.468	0.574
	JFLEG	0.959	0.990	0.344	0.408	0.568
	FCE	**0.967**	**1.000**	0.404	0.490	0.567
CWEB	CoNLL-2013	0.750	0.836	0.331	0.328	0.713
	CWEB	0.790	**0.963**	**0.472**	**0.432**	**0.780**
	JFLEG	0.757	0.818	0.353	0.354	0.775
	FCE	**0.805**	0.936	0.350	0.397	0.775
JFLEG	CoNLL-2013	0.959	0.990	0.516	0.604	0.677
	CWEB	0.952	0.972	0.524	0.572	0.644
	JFLEG	0.937	**1.000**	**0.618**	**0.685**	**0.783**
	FCE	**0.961**	0.990	0.581	0.649	0.627
FCE	CoNLL-2013	0.865	0.972	0.377	0.388	0.758
	CWEB	**0.882**	**0.990**	0.435	0.441	0.753
	JFLEG	0.852	0.972	0.390	0.429	0.739
	FCE	0.853	**0.990**	**0.541**	**0.616**	**0.848**

Table 4: Performance of IMPARA with different combinations of datasets used for training and evaluation. Using parallel data of the same domain and correction style for training and evaluation is important.

Dataset	Method	Corpus		Sentence		Chain
		Pea	Spe	Pea	Spe	Ken
CoNLL-2013	Scribendi	0.938	0.984	0.331	0.355	**0.698**
	SOME	**0.961**	**1.000**	0.370	0.419	0.502
	IMPARA	0.932	**1.000**	**0.411**	**0.515**	0.688
CWEB	Scribendi	0.637	0.451	0.177	0.194	0.616
	SOME	0.767	0.663	0.055	0.155	0.678
	IMPARA	**0.790**	**0.963**	**0.472**	**0.432**	**0.780**
JFLEG	Scribendi	0.932	0.945	0.255	0.303	0.574
	SOME	**0.955**	0.990	0.523	0.531	0.639
	IMPARA	0.937	**1.000**	**0.618**	**0.685**	**0.783**
FCE	Scribendi	**0.869**	0.933	0.342	0.449	**0.897**
	SOME	0.843	0.972	0.165	0.254	0.663
	IMPARA	0.853	**0.990**	**0.541**	**0.616**	0.848

Table 5: Performance of IMPARA and the two baseline methods on different datasets.

These results suggest that IMPARA achieves better evaluation performance than existing reference-less methods even without tailored data of human assessments, but only with parallel data.

3.3 Evaluation on other datasets

Table 4 summarizes correlation coefficients computed by MAEGE with different combinations of datasets used for training and evaluating the IMPARA model. For example, the second row of Table 4 shows the performance of IMPARA trained on CWEB and tested on CoNLL-2013. The table also indicates that IMPARA performed the best when the QE model was trained and evaluated on the same dataset. This observation is clearer when we evaluate IMPARA at the sentence and chain levels. Given that creating tailoerd evaluation data

Error type	Impact (10^{-2})	Frequency
NOUN	0.652	408
VERB:TENSE	0.649	480
VERB	0.580	557
NOUN:NUM	0.385	534
PUNCT	0.367	473
DET	0.364	1142
PREP	0.325	700

Table 6: Edit impacts for different error types (excluding OTHER) and their frequencies of occurrences in CoNLL-2014.

for each dataset with different domains and styles is expensive, IMPARA can provide a useful and practical solution, requiring only the parallel data on domains and styles.

Table 5 reports the performance of IMPARA and the two baseline methods on different datasets. IMPARA was trained on the training split of a dataset and evaluated on the test split. We did not adapt Scribendi Score to a target dataset because it is purely based on a pre-trained language model. It is impossible to adapt SOME to a target dataset because no human assessment data is available on the other datasets. SOME and Scribendi Score exhibits low performance on CWEB, FCE, and JFLEG in Table 5. In contrast, IMPARA achieved the highest correlations among all datasets. This result demonstrates the ability of IMPARA in adapting its evaluation metric to the target domain and style given by the parallel data.

3.4 Analysis

We analyzed the characteristic of edit impacts (Equation 5) computed by the BERT model. We examine edit impacts of different error types extracted by ERRANT on the CoNLL-2014 dataset. Table 6 presents the mean of edit impacts of error types, which appeared more than 400 times (excluding OTHER type)[9]. It is reasonable to find that edits for content words, such as NOUN (nouns) and VERB (verbs), have higher impacts than those for functional words, such as DET (determiner) and PREP (prepositions).

We focused more on the characteristics of impact by analyzing the highest and lowest correction impacts. Table 7 presents the corrections with the five largest edit impacts in the CoNLL-2014 dataset. In the first example, the sentence is not grammatical, and in the third, fourth, and fifth examples, the meanings of the sentences have been changed[10]. These sentences with higher impacts have relatively short length, indicating that a single correction has a large impact on the meaning of the entire sentence. In contrast, edits with the top five lowest impacts appeared in very long sentences (in 80, 235, 235, 235, and 235 words). Therefore, the longer a sentence, the smaller an impact of a correction in the sentence. These observations are consistent with the definition of Equation 6 and the importance of corrections recognized by humans.

When we generated supervision data in Section 2.4, edit impacts are not affected by the sentence length, which is constant for a given corrected sentence. Therefore, we compared edit impacts for the same corrected sentence. Table 8 shows an example of edits for content and functional words applied to the same corrected sentence. It demonstrates that the content word (i.e., VERB) has a higher impact than the functional words (i.e., DET, PREP) in the same corrected sentence, which is consistent with the results presented in Table 6.

4 Related Work

Reference-based metrics. Reference-based metrics for GEC use manually written reference sentences to evaluate a GEC system. M^2 (Dahlmeier and Ng, 2012), I-measure (Felice and Briscoe, 2015), and ERRANT (Bryant et al., 2017) are methods for evaluating GEC systems based on F-score. These methods requires explicit edit annotations to recognize the difference between the output and reference sentences to calculate precision, recall, and $F_{0.5}$. Napoles et al. (2015) proposed GLEU, a variant of BLEU metric (Papineni et al., 2002) commonly used for machine translation. GLEU does not require explicit edit annotations because it evaluates the quality of correction at the n-gram level. However, the reliability of reference-based metrics is low when reference sentences have low coverage (Choshen and Abend, 2018b).

Reference-less metrics. To address the issue of reference-based metrics, Napoles et al. (2016) introduced the reference-less metric that does not use reference sentences for the GEC evaluation. Their metric comprises a grammatical error detection tool and a language model, and showed a comparable performance to reference-based metrics. Islam and Magnani (2021) proposed Scribendi Score, which

[9]Insertion, replacement, and deletion are considered the same type.

[10]The correction of the second example in Table 7 is erroneous, but it actually appears in the CoNLL-2014 dataset.

Error Type	Sentence	Impact (10^{-2})
VERB	As a consequence , interpersonal skills _ / **are affected** .	6.57
NOUN	To some extent , this makes our life more luxurious and blundering / _ .	5.16
NOUN	One of the diseases is sickle cell trait / **anaemia** .	4.95
DET	People and friends often mock your / **their** conditions .	4.93
NOUN	They may not be able to enjoy a _ /**life** normal people can enjoy .	4.68

Table 7: Top-5 examples with the largest edit impacts in CoNLL-2014.

Error Type	Sentence	Impact (10^{-2})
Correct	Using text-messaging language as an informal way of communicating on social media networks also has a bad effect on us in the long term .	
VERB	Using text-messaging language as an informal way of communicating on social media networks also brings in / **has** a bad effect on us in the long term .	0.421
PREP	Using text-messaging language as an informal way of communicating on social media networks also has a bad effect for / **on** us in the long term .	0.236
DET	Using text-messaging language as an informal way of communicating on social media networks also has a bad effect on us in a / **the** long term .	0.164

Table 8: Examples of edits and their impacts computed for the same corrected sentence in CoNLL-2014.

is based on the perplexity of GPT-2 (Radford et al., 2019), token sort ratio, and Levenshtein distance ratio. These metrics require no GEC-specific language resource (e.g., supervision data for GEC). However, they cannot be adapted specifically to a particular domain or a correction style. Our experiments showed that they cannot evaluate GEC output robustly for a variety of domains and correction styles.

Some researchers proposed methods to directly optimize evaluation metrics on manual assessment scores for GEC outputs. Asano et al. (2017) presented a metric to evaluate GEC systems by combining sub-metrics of grammaticality, fluency, and meaning preservation. This metric is based on a regression model trained on GUG data (Heilman et al., 2014), language model, and METEOR (Denkowski and Lavie, 2014) as sub-metrics. Yoshimura et al. (2020) proposed a BERT-based metric wherein sub-metrics were optimized for human assessment scores. However, Takahashi et al. (2022) discovered that differences in learners' CEFR proficiency level[11] of a dataset affect the reliability of these metrics. Manual assessment scores are essential to these metrics to work in various domains and correction styles.

Meta evaluation methods. The performance of the metrics were judged by their closeness to human assessments (Banerjee and Lavie, 2005). A metric is generally compared using correlation coefficients between system outputs and human assessments for the outputs. Several studies have conducted meta-evaluations of automatic evaluation methods for several GEC systems (Grundkiewicz et al., 2015; Napoles et al., 2016; Asano et al., 2017). However, human annotations are known to yield poor inter-rater agreement. Therefore, Choshen and Abend (2018a) proposed MAEGE to meta-evaluate metrics without human annotations using sentence pairs ranked by the number of editing operations applied to ungrammatical sentences.

Reranking methods. Reranking methods used in GEC systems also estimate the quality of GEC outputs for choosing better corrections. Chollampatt and Ng (2018b) proposed the first neural-based reranking model that does not require any hand-crafted features for GEC. Using language models trained on large-scale corpora, such as BERT, several studies have improved the GEC performance further (Chollampatt et al., 2019; Kaneko et al., 2019). Liu et al. (2021) considered interactions of multiple outputs instead of evaluating them independently in reranking GEC outputs. The goal of these reranking models is to improve the GEG performance by selecting a better candidate from multiple candidates, not replicating a human GEC evaluation as in our study.

5 Conclusion

In this paper, we presented IMPARA, a novel reference-less metric for GEC. This method generates partially corrected sentences from parallel data comprising ungrammatical and grammatical sentence pairs. IMPARA learns their preference order of pairs of partially corrected sentences, which are

[11] https://www.cambridgeenglish.org/exams-and-tests/cefr/

determined by the impact of edits. The experiment results demonstrated that IMPARA achieved a comparable or better evaluation performance than existing reference-less methods even without tailored data of human assessments. In addition, IMPARA exhibited a high capability of adapting its metric to the target domain and style given by parallel data.

Future work includes providing an interpretable scale to scores computed by IMPARA. In addition, we plan to incorporate parallel data obtained by grammatical error generation. This may reduce the cost of building the parallel data and improve the quality of automatic evaluation metrics.

Acknowledgements

This paper is based on results obtained from a project, JPNP18002, commissioned by the New Energy and Industrial Technology Development Organization (NEDO).

References

Hiroki Asano, Tomoya Mizumoto, and Kentaro Inui. 2017. Reference-based metrics can be replaced with reference-less metrics in evaluating grammatical error correction systems. In *Proceedings of the Eighth International Joint Conference on Natural Language Processing (Volume 2: Short Papers)*, pages 343–348, Taipei, Taiwan. Asian Federation of Natural Language Processing.

Satanjeev Banerjee and Alon Lavie. 2005. METEOR: An automatic metric for MT evaluation with improved correlation with human judgments. In *Proceedings of the ACL Workshop on Intrinsic and Extrinsic Evaluation Measures for Machine Translation and/or Summarization*, pages 65–72, Ann Arbor, Michigan. Association for Computational Linguistics.

Christopher Bryant, Mariano Felice, and Ted Briscoe. 2017. Automatic annotation and evaluation of error types for grammatical error correction. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 793–805, Vancouver, Canada. Association for Computational Linguistics.

Shamil Chollampatt and Hwee Tou Ng. 2018a. A Multilayer Convolutional Encoder-Decoder Neural Network for Grammatical Error Correction. In *Proceedings of the Thirty-Second AAAI Conference on Artificial Intelligence*, pages 5755–5762, New Orleans, Louisiana. Association for the Advancement of Artificial Intelligence.

Shamil Chollampatt and Hwee Tou Ng. 2018b. Neural quality estimation of grammatical error correction. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 2528–2539, Brussels, Belgium. Association for Computational Linguistics.

Shamil Chollampatt, Weiqi Wang, and Hwee Tou Ng. 2019. Cross-sentence grammatical error correction. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 435–445, Florence, Italy. Association for Computational Linguistics.

Leshem Choshen and Omri Abend. 2018a. Automatic metric validation for grammatical error correction. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1372–1382, Melbourne, Australia. Association for Computational Linguistics.

Leshem Choshen and Omri Abend. 2018b. Inherent biases in reference-based evaluation for grammatical error correction. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 632–642, Melbourne, Australia. Association for Computational Linguistics.

Daniel Dahlmeier and Hwee Tou Ng. 2012. Better evaluation for grammatical error correction. In *Proceedings of the 2012 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 568–572, Montréal, Canada. Association for Computational Linguistics.

Michael Denkowski and Alon Lavie. 2014. Meteor universal: Language specific translation evaluation for any target language. In *Proceedings of the Ninth Workshop on Statistical Machine Translation*, pages 376–380, Baltimore, Maryland, USA. Association for Computational Linguistics.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, Minneapolis, Minnesota. Association for Computational Linguistics.

Mariano Felice and Ted Briscoe. 2015. Towards a standard evaluation method for grammatical error detection and correction. In *Proceedings of the 2015 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 578–587, Denver, Colorado. Association for Computational Linguistics.

Simon Flachs, Ophélie Lacroix, Helen Yannakoudakis, Marek Rei, and Anders Søgaard. 2020. Grammatical error correction in low error density domains: A new benchmark and analyses. In *Proceedings of the 2020 Conference on Empirical Methods in Natural*

Language Processing (EMNLP), pages 8467–8478, Online. Association for Computational Linguistics.

Roman Grundkiewicz, Marcin Junczys-Dowmunt, and Edward Gillian. 2015. Human evaluation of grammatical error correction systems. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 461–470, Lisbon, Portugal. Association for Computational Linguistics.

Michael Heilman, Aoife Cahill, Nitin Madnani, Melissa Lopez, Matthew Mulholland, and Joel Tetreault. 2014. Predicting grammaticality on an ordinal scale. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 174–180, Baltimore, Maryland. Association for Computational Linguistics.

Kengo Hotate, Masahiro Kaneko, Satoru Katsumata, and Mamoru Komachi. 2019. Controlling grammatical error correction using word edit rate. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics: Student Research Workshop*, pages 149–154, Florence, Italy. Association for Computational Linguistics.

Md Asadul Islam and Enrico Magnani. 2021. Is this the end of the gold standard? a straightforward reference-less grammatical error correction metric. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 3009–3015, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Marcin Junczys-Dowmunt, Roman Grundkiewicz, Shubha Guha, and Kenneth Heafield. 2018. Approaching neural grammatical error correction as a low-resource machine translation task. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 595–606, New Orleans, Louisiana. Association for Computational Linguistics.

Masahiro Kaneko, Kengo Hotate, Satoru Katsumata, and Mamoru Komachi. 2019. TMU transformer system using BERT for re-ranking at BEA 2019 grammatical error correction on restricted track. In *Proceedings of the Fourteenth Workshop on Innovative Use of NLP for Building Educational Applications*, pages 207–212, Florence, Italy. Association for Computational Linguistics.

Masahiro Kaneko, Masato Mita, Shun Kiyono, Jun Suzuki, and Kentaro Inui. 2020. Encoder-decoder models can benefit from pre-trained masked language models in grammatical error correction. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 4248–4254, Online. Association for Computational Linguistics.

Masahiro Kaneko, Sho Takase, Ayana Niwa, and Naoaki Okazaki. 2022. Interpretability for language learners using example-based grammatical error correction. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 7176–7187, Dublin, Ireland. Association for Computational Linguistics.

Zhenghao Liu, Xiaoyuan Yi, Maosong Sun, Liner Yang, and Tat-Seng Chua. 2021. Neural quality estimation with multiple hypotheses for grammatical error correction. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 5441–5452, Online. Association for Computational Linguistics.

Courtney Napoles, Keisuke Sakaguchi, Matt Post, and Joel Tetreault. 2015. Ground truth for grammatical error correction metrics. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 2: Short Papers)*, pages 588–593, Beijing, China. Association for Computational Linguistics.

Courtney Napoles, Keisuke Sakaguchi, and Joel Tetreault. 2016. There's no comparison: Reference-less evaluation metrics in grammatical error correction. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 2109–2115, Austin, Texas. Association for Computational Linguistics.

Courtney Napoles, Keisuke Sakaguchi, and Joel Tetreault. 2017. Jfleg: A fluency corpus and benchmark for grammatical error correction. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 2, Short Papers*, pages 229–234, Valencia, Spain. Association for Computational Linguistics.

Hwee Tou Ng, Siew Mei Wu, Yuanbin Wu, Christian Hadiwinoto, and Joel Tetreault. 2013. The CoNLL-2013 shared task on grammatical error correction. In *Proceedings of the Seventeenth Conference on Computational Natural Language Learning: Shared Task*, pages 1–12, Sofia, Bulgaria. Association for Computational Linguistics.

Kostiantyn Omelianchuk, Vitaliy Atrasevych, Artem Chernodub, and Oleksandr Skurzhanskyi. 2020. GECToR – grammatical error correction: Tag, not rewrite. In *Proceedings of the Fifteenth Workshop on Innovative Use of NLP for Building Educational Applications*, pages 163–170, Seattle, WA, USA → Online. Association for Computational Linguistics.

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. Bleu: a method for automatic evaluation of machine translation. In *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics*, pages 311–318, Philadelphia, Pennsylvania, USA. Association for Computational Linguistics.

Alec Radford, Jeff Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. 2019. Language models are unsupervised multitask learners.

Yujin Takahashi, Masahiro Kaneko, Masato Mita, and Mamoru Komachi. 2022. Proficiency matters quality estimation in grammatical error correction. *arXiv preprint arXiv:2201.06199*.

Helen Yannakoudakis, Ted Briscoe, and Ben Medlock. 2011. A new dataset and method for automatically grading ESOL texts. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, pages 180–189, Portland, Oregon, USA. Association for Computational Linguistics.

Ryoma Yoshimura, Masahiro Kaneko, Tomoyuki Kajiwara, and Mamoru Komachi. 2020. SOME: Reference-less sub-metrics optimized for manual evaluations of grammatical error correction. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 6516–6522, Barcelona, Spain (Online). International Committee on Computational Linguistics.

Zheng Yuan and Ted Briscoe. 2016. Grammatical error correction using neural machine translation. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 380–386, San Diego, California. Association for Computational Linguistics.

Evons: A Dataset for Fake and Real News Virality Analysis and Prediction

Kriste Krstovski,[1,2] Angela Soomin Ryu,[1] and Bruce Kogut[1]
[1]Columbia Business School, Columbia University
[2]Data Science Institute, Columbia University
{kriste.krstovski,asr2193,bruce.kogut}@columbia.edu

Abstract

We present a novel collection of news articles originating from fake and real news media sources for the analysis and prediction of news virality. Unlike existing fake news datasets which either contain claims or news article headline and body, in this collection each article is supported with a Facebook engagement count which we consider as an indicator of the article virality. In addition we also provide the article description and thumbnail image with which the article was shared on Facebook. These images were automatically annotated with object tags and color attributes. Using cloud based vision analysis tools, thumbnail images were also analyzed for faces and detected faces were annotated with facial attributes. We empirically investigate the use of this collection on an example task of article virality prediction.

1 Introduction

Fake news articles are widely spread across social media platforms such as Facebook and Twitter. This is mainly due to the fact that social media is gradually becoming the main source of news consumption (Shu et al., 2018). Due to the sharing features that these platforms offer, fake news propagate rapidly and their effects resonate and persist across many users (Baly et al., 2018). The wide spread of fake news in social media has lead to the development of automatic fake news detection approaches (Ruchansky et al., 2017; Pérez-Rosas et al., 2018; Nguyen et al., 2019; Zellers et al., 2019), to name a few. Developing fake news detection models require annotated collections of fake and real news articles. Most prior work on the creation and annotation of such collections has focused on this task. Significant number of these collections contain claims fact-checked for veracity (Vlachos and Riedel, 2014; Wang, 2017). A recent survey of such collections is provided in Guo et al. (2022).

On the other hand there exist collections of fake news articles that contain article headline and body text (Horne et al., 2018; Potthast et al., 2018; Zhou et al., 2020). Given that these and other existing fake news collections were developed mainly for fake news detection they can't be used for analysing and predicting fake news virality which is the set of tasks of our focus. Recently, Shu et al. (2018) created FakeNewsNet, a collection of ∼24k news articles labeled as fake or real using the fact-checking websites PolitiFact (PolitiFact, 2017) and Gossip Cop (Gossip Cop, 2021). Articles in this collection are annotated with social engagement information obtained through the Twitter search API. However this collection doesn't include thumbnail images and article descriptions which, along with the headlines, are the only sources of information readers are exposed to on social media platforms regardless of their choices whether to click the link of the shared article or not.

To address this drawback we present Evons – a collection of news articles originating from fake and real news media sources where each article has the thumbnail image and description with which it was shared on Facebook. We use the article engagement count on Facebook as an implicit indicator of the article virality. Given that fake news writers profit from advertising revenue rather than subscription fees, the body text of fake news articles (which are only shown after clicking the link) are known to be repetitive and lacking in informational value (Horne and Adali, 2017). Therefore we believe that these two article components are important for social media sharing. Thumbnail images are annotated with content tags and color attributes while detected faces are annotated with facial attributes. The Evons collection is accessible through https://github.com/krstovski/evons. We showcase the use of this collection on the task of article virality prediction which we consider as one example task that could be created wit this dataset.

Proceedings of the 29th International Conference on Computational Linguistics, pages 3589–3596
October 12–17, 2022.

2 Collection Construction

The Evons collection contains 92,969 news articles from fake and real news media sources published in the period between January 2016 and December 2017. We selected this time period to reflect on the 2016 Presidential election which many believed that fake news had a significant impact on. Across both media sources we focused on news articles originating from the same news sections therefore covering similar or the same set of topics. We also don't consider any article author related information. The set of fake news sources was created using information from 3 independent lists of fake news websites that were developed through human curation. It contains only fake news sources that were cross-referenced by at least 2 of the 3 lists. We follow the most widely used definition of fake news as "intentionally and verifiably false, and could mislead readers" by Allcott and Gentzkow (2017) and exclude satire and parody websites.

We used the "Questionable Sources" list from Media Bias Fact Check (MBFC) (Media Bias/Fact Check, 2019) which includes sources with extreme bias, propaganda and conspiracies, and fake news. We filtered the websites to retain only those that are explicitly annotated as "some fake news" or "fake news", indicating that the source deliberately publishes hoaxes and/or disinformation.

Our second list is the "Politifact's Fake News Almanac" (PolitiFact, 2017). This list was created in partnership with Facebook and includes "fake news" websites which were found to contain *deliberately* false or fake stories that have appeared in people's news feeds on Facebook.

The third list is from the "BS Detector" collection. This is a list of "unreliable or otherwise questionable sources" curated by professionals (Risdal, 2017).

After cross-referencing, we obtained 16 fake news sources that appeared in at least 2 of the lists. We then removed sources that were republishing news content from other sources *and* websites that started publishing after the 2016 elections. Our final list contains the following 6 fake news media sources: American Freedom Fighters (AFF), Barracuda Brigade (BB4SP), MadWorldNews (MWN), Puppet String News (PSN), USA Supreme (USAS), and YourNewsWire (YNW). The set of real news sources was obtained from the readily available "All the news 2.0" dataset (Thompson, 2019) which consists of 18 American mainstream sources. We focused on sources that had "high" or "very high" scores in factual reporting *and* "very slight" or "neutral" political biases according to MBFC. There were 5 such sources in this dataset: The Guardian, National Public Radio (NPR), New York Times (NYT), Reuters, and Washington Post (WP). We use articles published in the same time period as our fake media set. In Table 1 we provide the number of articles across the fake and real news media sources.

Media Source	# of Articles
AFF	7,536
BB4SP	2,792
MWN	11,315
PSN	6,576
USAS	3,038
YNW	11,519
Total from fake	42,776
The Guardian	9,811
NPR	11,813
NYT	5,439
Reuters	14,993
WP	8,137
Total from real	50,193
Total	92,969

Table 1: Number of articles in the Evons collection.

We used the webpreview[1] package for extracting thumbnail images. These images come from the thumbnails that are carefully curated by the news producers. They decide what title, description, and thumbnail image would be the most effective in achieving their goal, whether it is to best represent the content or simply attract the most engagement for larger advertising revenue. With this package we also extract article description which is the text that appears as preview when the article is shared.

All articles contain a thumbnail image except for USAS and BB4SP were 0.1% and 11.1% of the articles don't have thumbnails. Thumbnail images are either a picture or a logo of the news media source. Table 2 gives statistics of the number of real and fake articles with and without thumbnail images. Unlike real news articles where a small percentage of them had the media source logo as the thumbnail image, fake news articles always used pictures as thumbnails.

[1] https://pypi.org/project/webpreview

Thumbnail Type	Real	Fake	Total
Picture	48,592	42,464	91,056
Logo	1,601	0	1,601
None	0	312	312

Table 2: Thumbnail statistics.

Engagement Statistics	Real	Fake
Min # of engagements	0	0
Max # of engagements	4.78m	1.08m
Mean # of engagements	6.73	1.58

Table 3: Engagement statistics.

2.1 Engagement Count

A commonly used measure for virality by marketing and communication researchers is how many times a piece of information is shared (Berger and Milkman, 2012; Scholz et al., 2017). Here we use Facebook engagements as a proxy of how much attention the post generated. Facebook engagements is the sum of the number of Facebook shares, likes, and comments. Facebook provides the numbers received by an URL through the Facebook sharing debugger (FSD) (Facebook, 2022). Since FSD works on individual URLs we used the Shared Count API (SharedCount, 2022) to automate the process of fetching these numbers for multiple articles, except for articles from USAS which was blacklisted on Facebook. For this website we used BuzzSumo (BuzzSumo, 2022) which is another third-party measurement dashboard that fetches data from FSD. Both platforms do not provide nor maintain any user related information and have been used in the past across an array of research topics (Xu and Guo, 2018; Xu, 2019; Obiała et al., 2021; Rhodes, 2022). In Table 3 we provide engagement statistics.

2.2 Image Annotation

We performed two types of automatic image annotation. Using Microsoft Azure (Microsoft, 2022) images are analyzed for visual features and color schemes. With the Amazon Rekognition platform (Amazon, 2022) images are analyzed for the presence of faces and detected faces were annotated with facial attributes. Accuracy of both platform on these annotation tasks have been extensively evaluated and confirmed in the past across a variety of image types which include images commonly used as thumbnails (Kyriakou et al., 2019; Liu and Wilkinson, 2020; Malone and Burns, 2021).

Image Tag Statistics	Real	Fake
Min # of tags	0	0
Max # of tags	99	86
Mean # of tags	9.47	9.08

Table 4: Image tag statistics.

2.2.1 Object Detection and Tagging

Images are automatically annotated with content tags such as objects, living beings, scenery, and actions. There were 5,160 distinct tags identified. Articles originating from fake media sources had 3,670 distinct tags with 379 being unique to fake. Real sources contained 4,781 distinct tags with 1,490 unique to real. Table 4 shows image tag statistics. Table 5 shows the top 10 most frequent tags discovered across all media sources, unique to fake, and real news sources.

2.2.2 Color Schemes

Thumbnail images are automatically annotated with three color attributes: dominant foreground and background color, and a set of dominant colors across the whole image. There are 12 colors used: black, blue, brown, gray, green, orange, pink, purple, red, teal, white, and yellow. Dominant background and foreground colors can take on a single value. Thumbnails are also annotated with accent color, which is the most vibrant color in the image, and whether the image is in black and white (bw). In Appendix A we provide summary of the colors present as dominant attribute in thumbnail images.

2.2.3 Facial Analysis

Detected faces are annotated with a bounding box and the following attributes: person's gender, whether the person is smiling, wearing eyeglasses or sunglasses, has a mustache or eyes open, brightness, and sharpness. We also obtain the emotions that appear to be expressed on the face which include: fear, sad, happy, calm, angry, confused, surprised, and disgusted. Table 6 provides face statistics. In Appendix B we show the distribution of dominant face emotions.

3 Example Task

We use the task of predicting article virality as an example task (out of many different tasks) that could be constructed using the Evons collection. The example task is a multi-class classification

All	Unique to Real	Unique to Fake
1. person	1. salad	1. photo caption
2. clothing	2. minimalist	2. television presenter
3. human face	3. raquet sport	3. thong
4. man	4. racketlon	4. shout
5. text	5. piece de resistance	5. g-string
6. outdoor	6. tennis player	6. f-15 eagle
7. suit	7. soft tennis	7. salumi
8. indoor	8. modern	8. salami
9. smile	9. professional boxing	9. ciauscolo
10. tie	10. camera lens	10. ostrich

Table 5: Top 10 most frequent tags across all media sources, unique to real, and fake news sources.

Face Statistics	Real	Fake
% of images with face/s	74.26	77.08
Mean # of faces per image	3.31	2.74

Table 6: Face statistics.

problem which we created by dividing articles from fake and real news media sources into two groups based on their engagement count: real-low, real-high, fake-low, and fake-high. We use the median number of engagements to create almost equal groups of real and fake articles with low and high number of engagements. We empirically investigate how well do various approaches, which we consider as baselines, perform on this task.

3.1 Experimental Setup and Results

The task dataset consists of articles with pictures as thumbnails where the picture contained at least one tag and face. There are 68,793 such articles out of which 36,072 come from real and 32,721 from fake media sources. Articles are represented using two sets of textual features and three sets of image features, one for each of the three image annotation types. For the textual features we use tf-idf values computed over the words of article titles and descriptions. The title feature vector contains 29,745 words and the description feature vector with 43,861 words. Combining both we obtain a vocabulary of 49,792 words. Thumbnail images were represented with 3,526 features: 3,471 object tags, 42 color and 13 facial. Color features include accent color, dominant color attributes, and bw indicator. Facial features include number of faces, person smiling, gender, brightness, sharpness, and facial emotions. Facial features were weighted based on the size of the bounding box area of the detected face. In Appendix C we provide details on the weighing approach used. For features that are indicator variables we use the confidence score as a feature value.

We evaluated 6 different classification models: logistic regression (LR), SVM, multilayer perceptron (MLP), Bidirectional LSTM (Hochreiter and Schmidhuber, 1997) (Bi-LSTM), XLNet (Yang et al., 2019), and RoBERTa (Liu et al., 2019); using a 90/10 split of our dataset. We used the scikit-learn (Pedregosa et al., 2011) implementation of LR and SVM. MLP consists of three fully-connected layers containing 256 and 8 nodes in the first two layers with ReLU. The last layer is a 4 nodes with SoftMax activation. Bi-LSTM consists of a 64 dimensional embedding representation layer, a fully connected layer with ReLU, and an output layer as in MLP. Both NNs were implemented in Keras (Gulli and Pal, 2017). We used the simpletransfomers (Thompson, 2022) implementation of XLNet and RoBERTa with maximum sequence length of 256. Table 7 shows performance comparison results across all models using different feature representations and combinations of them. For ease of interpretability we use accuracy. Thumbnail images were represented using all image generated features. RoBERTa with all feature types performs best. While across most models incorporating image features helps we don't observe substantial accuracy improvement over textual features. We believe that this could be significantly improved with image feature analysis and exploring feature selection approaches.

Feature	Accuracy					
	LR	SVM	MLP	Bi-LSTM	XLNet	RoBERTa
Title (T)	0.632	0.608	0.643	0.632	0.731	0.751
Description (D)	0.674	0.631	0.680	0.687	0.760	0.773
T+D	0.694	0.655	0.718	0.691	0.801	0.807
T+D+Tag	0.701	0.661	0.719	0.712	0.793	0.808
T+D+Color	0.701	0.658	0.716	0.688	0.781	0.801
T+D+Facial	0.697	0.655	0.716	0.688	0.794	0.802
All	0.703	0.666	0.714	0.683	0.791	0.810

Table 7: Accuracy results across various baseline models on the example task of article virality prediction.

4 Conclusion

We presented Evons - a collection of news articles originating from fake and real media sources where articles are annotated with a Facebook engagement count, thumbnail image and article description. Thumbnails are automatically annotated with object tags, color and facial attributes. We demoed the collection use on an article virality prediction task and established baselines using 6 models. In the future we plan to use Evons to explore various approaches for selection of image features and combination with text that would further help improve accuracy on this task.

5 Ethics

Creating the Evons collection involved collecting news articles from various online media sources, extracting thumbnail images using the webpreview package, and obtaining Facebook engagement counts through the SharedCount API and the BuzzSumo platforms. Throughout the creation process we made sure that no author metadata or user identifying information was collected. Therefore our collection does not contain any information that names or uniquely identifies individual people. Both Facebook engagement counts platforms do not provide any user related information. While news articles across various online media sources do provide article author information in our collection process we ignored this information.

We don't foresee any potential risks that may arise from the creation of our collection especially in terms of identifying potential stakeholders that may benefit from this collection while harming others. To the best of our knowledge all of our collected data is in the public domain and is not copyrighted.

For our thumbnail image annotations we relied on two image annotation platforms: Microsoft Azure and Amazon Rekognition. One limitation of our work may arise from the fact that we don't know whether the models that are part of these platforms contain any type of bias and if so to which extent bias is present.

References

Hunt Allcott and Matthew Gentzkow. 2017. Social media and fake news in the 2016 election. *Journal of Economic Perspectives*, 31(2):211–36.

Amazon. 2022. Detecting and analyzing faces. https://docs.aws.amazon.com/rekognition/latest/dg/faces [Accessed: April, 2022].

Ramy Baly, Georgi Karadzhov, Dimitar Alexandrov, James Glass, and Preslav Nakov. 2018. Predicting factuality of reporting and bias of news media sources. In *EMNLP*, pages 3528–3539.

Jonah Berger and Katherine L. Milkman. 2012. What makes online content viral? *Journal of Marketing Research*, 49(2):192–205.

BuzzSumo. 2022. https://buzzsumo.com [Accessed: April, 2022].

Facebook. 2022. https://developers.facebook.com/tools/debug/ [Accessed: April, 2022].

Gossip Cop. 2021. https://www.gossipcop.com/about.html [Accessed: October, 2021].

Antonio Gulli and Sujit Pal. 2017. *Deep learning with Keras*. Packt Publishing Ltd.

Zhijiang Guo, Michael Schlichtkrull, and Andreas Vlachos. 2022. A survey on automated fact-checking. *Transactions of the Association for Computational Linguistics*, 10:178–206.

Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long short-term memory. *Neural computation*, 9(8):1735–1780.

Benjamin D Horne and Sibel Adali. 2017. This just in: fake news packs a lot in title, uses simpler, repetitive content in text body, more similar to satire than real news. In *AAAI Conference on Web and Social Media*.

Benjamin D Horne, Sara Khedr, and Sibel Adali. 2018. Sampling the news producers: A large news and feature data set for the study of the complex media landscape. In *AAAI Conference on Web and Social Media*.

Kyriakos Kyriakou, Pınar Barlas, Styliani Kleanthous, and Jahna Otterbacher. 2019. Fairness in proprietary image tagging algorithms: A cross-platform audit on people images. In *AAAI Conference on Web and Social Media*, volume 13, pages 313–322.

Ching Yiu Jessica Liu and Caroline Wilkinson. 2020. Image conditions for machine-based face recognition of juvenile faces. *Science & Justice*, 60(1):43–52.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Roberta: A robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692*.

Ashling Malone and John Burns. 2021. Evaluating the accuracy of public cloud vendor face detection api's. *Journal of Image and Graphics*, 9(1).

Media Bias/Fact Check. 2019. Media bias/fact check questionable sources. https://mediabiasfactcheck.com/fake-news/ [Accessed: December, 2019].

Microsoft. 2022. What is computer vision? https://docs.microsoft.com/en-us/azure/cognitive-services/computer-vision/home [Accessed: April, 2022].

Duc Minh Nguyen, Tien Huu Do, Robert Calderbank, and Nikos Deligiannis. 2019. Fake news detection using deep Markov random fields. In *NAACL*, pages 1391–1400.

Justyna Obiała, Karolina Obiała, Małgorzata Mańczak, Jakub Owoc, and Robert Olszewski. 2021. Covid-19 misinformation: accuracy of articles about coronavirus prevention mostly shared on social media. *Health policy and technology*, 10(1):182–186.

Fabian Pedregosa, Gaël Varoquaux, Alexandre Gramfort, Vincent Michel, Bertrand Thirion, Olivier Grisel, Mathieu Blondel, Peter Prettenhofer, Ron Weiss, Vincent Dubourg, et al. 2011. Scikit-learn: Machine learning in python. *JMLR*, 12:2825–2830.

Verónica Pérez-Rosas, Bennett Kleinberg, Alexandra Lefevre, and Rada Mihalcea. 2018. Automatic detection of fake news. In *COLING*, pages 3391–3401.

PolitiFact. 2017. Politifact's guide to fake news websites and what they peddle. https://www.politifact.com/article/2017/apr/20/politifacts-guide-fake-news-websites-and-what-they [Accessed: October, 2021].

Martin Potthast, Johannes Kiesel, Kevin Reinartz, Janek Bevendorff, and Benno Stein. 2018. A stylometric inquiry into hyperpartisan and fake news. In *ACL*, pages 231–240.

Samuel C Rhodes. 2022. Filter bubbles, echo chambers, and fake news: how social media conditions individuals to be less critical of political misinformation. *Political Communication*, 39(1):1–22.

Megan Risdal. 2017. Getting Real about Fake News. https://www.kaggle.com/mrisdal/fake-news [Accessed: April, 2022].

Natali Ruchansky, Sungyong Seo, and Yan Liu. 2017. Csi: A hybrid deep model for fake news detection. In *CIKM*, pages 797–806.

Christin Scholz, Elisa C. Baek, Matthew Brook O'Donnell, Hyun Suk Kim, Joseph N. Cappella, and Emily B. Falk. 2017. A neural model of valuation and information virality. *PNAS*, 114(11):2881–2886.

SharedCount. 2022. https://www.sharedcount.com [Accessed: April, 2022].

Kai Shu, Deepak Mahudeswaran, Suhang Wang, Dongwon Lee, and Huan Liu. 2018. Fakenewsnet: A data repository with news content, social context and dynamic information for studying fake news on social media. *arXiv preprint arXiv:1809.01286*.

Andrew Thompson. 2019. "all the news 2.0" dataset. https://components.one/datasets/all-the-news-articles-dataset [Accessed: December, 2019].

Andrew Thompson. 2022. https://simpletransformers.ai [Accessed: April, 2022].

Andreas Vlachos and Sebastian Riedel. 2014. Fact checking: Task definition and dataset construction. In *ACL*, pages 18–22.

William Yang Wang. 2017. "liar, liar pants on fire": A new benchmark dataset for fake news detection. In *ACL*, pages 422–426.

Zhan Xu. 2019. Personal stories matter: topic evolution and popularity among pro-and anti-vaccine online articles. *Journal of computational social science*, 2(2):207–220.

Zhan Xu and Hao Guo. 2018. Using text mining to compare online pro-and anti-vaccine headlines: word usage, sentiments, and online popularity. *Communication Studies*, 69(1):103–122.

Zhilin Yang, Zihang Dai, Yiming Yang, Jaime Carbonell, Russ R Salakhutdinov, and Quoc V Le. 2019. Xlnet: Generalized autoregressive pretraining for language understanding. In *NeurIPS*, pages 5753–5763.

Rowan Zellers, Ari Holtzman, Hannah Rashkin, Yonatan Bisk, Ali Farhadi, Franziska Roesner, and Yejin Choi. 2019. Defending against neural fake news. In *NeurIPS*, pages 9054–9065.

Xinyi Zhou, Apurva Mulay, Emilio Ferrara, and Reza Zafarani. 2020. Recovery: A multimodal repository for covid-19 news credibility research. In *Proceedings of the 29th ACM international conference on information & knowledge management*, pages 3205–3212.

A Dominant Colors

Shown in Figure 1 are bar plots of the percentage of colors present as dominant attribute in thumbnail images.

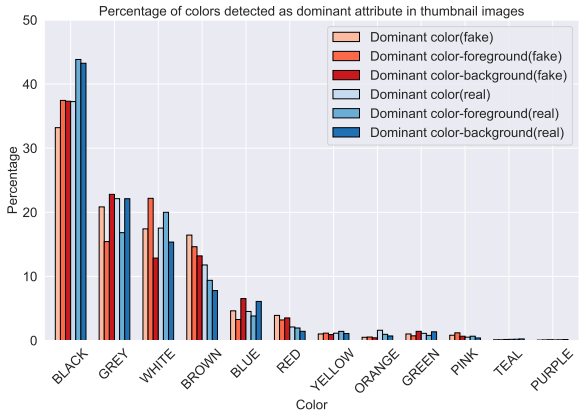

Figure 1: Percentage of color present as dominant attribute in thumbnail images.

B Dominant Emotions

Shown in Figure 2 are bar plots of the percentage of emotion detected as dominant on faces found in thumbnail images.

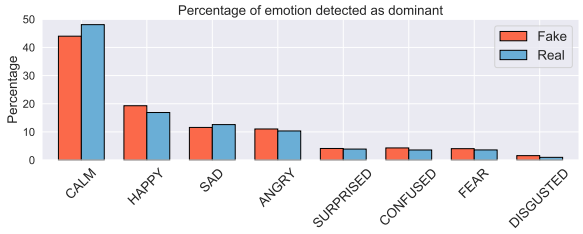

Figure 2: % of emotion detected as dominant in faces.

C Facial Features

Facial features across thumbnail images where weighted based on the bounding box area of the detected face. The bounding box area is the product of the bounding box width and height. Given a bounding box area B_{ij} of the jth face in image i and a set of k features F_{jk} detected on that face, the weighted facial features for image i, W_{ik} are computed as:

$$W_{ik} = \sum_{j=1}^{J} B_{i,j} F_{j,k} \quad (1)$$

Are Pretrained Multilingual Models Equally Fair Across Languages?

Laura Cabello Piqueras
University of Copenhagen
lcp@di.ku.dk

Anders Søgaard
University of Copenhagen
soegaard@di.ku.dk

Abstract

Pretrained multilingual language models can help bridge the digital language divide, enabling high-quality NLP models for lower-resourced languages. Studies of multilingual models have so far focused on performance, consistency, and cross-lingual generalisation. However, with their wide-spread application in the wild and downstream societal impact, it is important to put multilingual models under the same scrutiny as monolingual models. This work investigates the group fairness of multilingual models, asking whether these models are equally fair across languages. To this end, we create a new four-way multilingual dataset of parallel cloze test examples (MozArt), equipped with demographic information (balanced with regard to gender and native tongue) about the test participants. We evaluate three multilingual models on MozArt – mBERT, XLM-R, and mT5 – and show that across the four target languages, the three models exhibit different levels of group disparity, e.g., exhibiting near-equal risk for Spanish, but high levels of disparity for German.

1 Introduction

Fill-in-the-gap cloze tests (Taylor, 1953) ask language learners to predict what words were removed from a text and it is a "procedure for measuring the effectiveness of communication". Today, language models are trained to do the same (Devlin et al., 2019). This has the advantage that we can now use fill-in-the-gap cloze tests to directly compare the linguistic preferences of humans and language models, e.g., to investigate task-independent sociolectal biases (group disparities) in language models (Zhang et al., 2021). This paper presents a novel four-way parallel cloze dataset for English, French, German, and Spanish that enables apples-to-apples comparison across languages of group disparities in multilingual language models.[1]

[1]The language selection was given to us, because we rely on an existing word alignment dataset; see §2.

	EN	ES	DE	FR
WordPiece (avg. #tokens)	19.7	22.0	23.6	23.1
SentencePiece (avg. #tokens)	22.3	22.9	24.9	25.3
#Sentences	100	100	100	100
#Annotations	600	600	600	600
#Annotators	60	60	60	60
Demographics	id_u, id_s, gender, age, nationality, first language, fluent languages, current country of residence, country of birth, time taken			

Table 1: MozArt details. The average number of tokens per sentence is reported using WordPiece and SentecePiece. The bottom row lists the demographic attributes shared; id_u refers to user id (anonymised) and id_s to sentence id.

Language models induced from historical data are prone to implicit biases (Zhao et al., 2017; Chang et al., 2019; Mehrabi et al., 2021), e.g., as a result of the over-representation of male-dominated text sources such as Wikipedia and newswire (Hovy and Søgaard, 2015). This may lead to language models that are *unfair* to groups of users in the sense that they work better for some groups rather than others (Zhang et al., 2021). Multilingual language models can be unfair to their training languages in similar ways (Choudhury and Deshpande, 2021; Wan, 2022; Wang et al., 2021), but this work goes beyond previous work in evaluating whether multilingual language models are *equally fair to demographic groups across languages*.

To this end, we create MozArt, a multilingual dataset of fill-in-the-gap sentences covering four languages (English, French, German and Spanish). The sentences reflect diastratic variation within each language and can be used to compare biases in pretrained language models (PLMs) across languages. We study the influence of four demographic groups, i.e., the cross-product of our annotators' gender – male (*M*) or female (*F*)[2] – and first

[2]None of our annotators identified as non-binary.

Proceedings of the 29th International Conference on Computational Linguistics, pages 3597–3605
October 12–17, 2022.

language – native (*N*) or non-native (*NN*).[3] Table 1 presents a summary of dataset characteristics.

2 Dataset

We introduce MozArt, a four-way multilingual cloze test dataset with annotator demographics. We sampled 100 sentence quadruples from each of the four languages (English, French, German, Spanish) in the corpus provided for the WMT 2006 Shared Task.[4] The data was extracted from the publicly available Europarl corpus (Koehn, 2005) and enhanced with word-level bitext alignments (Koehn and Monz, 2006). The word alignments are important for what follows. We manually verify that sentences make sense out of context and use the data to generate *comparable cloze examples*, e.g.:

en	[MASK] that deplete the ozone layer
es	[MASK] que agotan la capa de ozono
de	[MASK], die zum Abbau der Ozonschicht führen
fr	[MASK] appauvrissant la couche d'ozone

We only mask words which are (i) aligned by one-to-one alignments, and which are (ii) either nouns, verbs, adjectives or adverbs.[5] We mask one word in each sentence and verify that one-to-one alignments exist in all languages. Following Kleijn et al. (2019), we rely on part-of-speech information to avoid masking words that are *too* predictable, e.g., auxiliary verbs or constituents of multi-word expressions, or words that are *un*-predictable, e.g., proper names and technical terms.

Annotators were recruited using Prolific.[6] We applied eligibility criteria to balance our annotators across demographics. Participants were asked to report (on a voluntary basis) their demographic information regarding gender and languages spoken. Each eligible participant was presented with 10 cloze examples. We collected answers from 240 annotators, 60 per language batch, divided in four balanced demographic groups (gender × native language). We made sure that each sentence had at least six annotations. Annotation guidelines for each language were given in that language, to avoid bias and ensure a minimum of language understanding for non-native speakers. We manually filtered out spammers to ensure data quality.

The dataset is made publicly available at `github.com/coastalcph/mozart` under a CC-BY-4.0 license. We include all the demographic attributes of our annotators as per agreement with the annotators. The full list of protected attributes is found in Table 1. We hope MozArt will become a useful resource for the community, also for evaluating the fairness of language models across other attributes than gender and native language.

3 Experimental Setup

Models We evaluate three PLMs: mBERT (Devlin et al., 2019), XLM-RoBERTa/XLM-R (Conneau et al., 2020), and mT5 (Xue et al., 2021).[7] All three models were trained with a masked language modelling objective. mBERT differs from XLM-R and mT5 in including a next sentence prediction objective (Devlin et al., 2019). mT5 differs from mBERT and XLM-R in allowing for consecutive spans of input tokens to be masked (Raffel et al., 2020). We adopt beam search decoding with early stopping and constrain the generation to single words. This enables better correlation of mT5's output with our group preferences. t-SNE plots are included in Appendix B to show how languages are distributed in the PLM vector spaces.

Metrics We use several metrics to compare how the PLMs align with group preferences across languages. These include top-k precision P@k with k={1, 5}, mean reciprocal rank (MRR), and two classical univariate rank correlations: Spearman's ρ (Spearman, 1987) and Kendall's τ (Kendall, 1938).

Given a set of $|S|$ cloze sentences and a group of annotators, for each sentence s, we denote the list of answers, ranked by their frequency, as $W_s = [w_1, w_2, ...]$, and the list of model's predictions as $C_s = [c_1, c_2, ...]$, ranked by their model likelihood. Then, we report $\text{P@k} = \mathbb{1}[c_i \in W_s]$ with $i \in [1, k]$, where $\mathbb{1}[\cdot]$ is the indicator function. Precision is reported together with its standard deviation, to account for the group-wise disparity in

[3] See Schmitz (2016); Faez (2011) for discussion of the native/non-native speaker dichotomy. Participants were asked "What is your first language?" and "Which of the following languages are you fluent in?". We use *native* (*N*) for people whose first language coincides with the example sentences, and non-native (*NN*) otherwise, without any sociocultural implications.

[4] https://www.statmt.org/wmt06/shared-task/

[5] We use spaCy's part-of-speech tagger (Honnibal and Montani, 2017) to predict the syntactic categories of the input words.

[6] `prolific.co`

[7] We use the base models available from `https://huggingface.co/models`. We report results using uncased mBERT, since it performed better on our data than its cased sibling.

both dimensions (social groups and language):

$$\sigma_{\text{gd}} = \sqrt{\frac{\sum_{j=1}^{G}(\text{P@k}_j - \overline{\text{P@k}})^2}{\text{G}}} \quad (1)$$

where $\overline{\text{P@k}}$ is the mean value of all observations, and G the total number of groups across the dimension fixed each time i.e., $\text{G} = 4$ across social groups (*MN, FN, MNN, FNN*) and $\text{G} = 4$ across languages (EN, ES, DE, FR). We also compute the mean-reciprocal rank (MRR) of the elements of W_s with respect to the top-n ($\text{n} = 5$) elements of C_s (C_s^n):

$$\text{MRR} = \frac{1}{|S|}\sum_{s=1}^{|S|}\frac{1}{Rank_i^{C_s^n}} \quad (2)$$

Finally, we compute Spearman's ρ (Spearman, 1987) and Kendall's τ (Kendall, 1938) between W_s and C_s^5. These metrics are generally more robust to outliers.

4 Results

Following previous work on examining fairness of document classification (Huang et al., 2020; Dixon et al., 2018; Park et al., 2018; Garg et al., 2019), we focus on group-level performance differences (group disparity). We measure the group disparity as the variance in PLM's performance (P@k) across demographics (gender and native language). Table 2 shows better precision for native speakers in German and French (*MN*, *FN*) for P@1. In terms of group disparity, male non-natives (*MNN*) is the demographic exhibiting the highest disparity across languages in mBERT, while it is female natives (*FN*) in XLM-R and male natives (*MN*) in mT5. Language-wise, we see the largest group disparity with German in all three models. Here, we see 2.5–4.4 between-group differences, compared to, e.g., 0.3–1.8 between-group differences for English. See Appendix A for results with P@5.

XLM-R consistently exhibits better overall performance on average, but higher between-group and between-language differences in terms of precision (σ_{gd}).

Figure 1 complements results from Table 2 with MRR scores. We observe a common trend that the models often underperform on non-native male speakers in all languages except for Spanish: Performance is (always) below the average, and they are the worst-off group (↓) in most of the cases. At the same time, predictions with mBERT and

mBERT

P@1	EN	ES	DE	FR	
MN	13.3	12.7	11.3	10.7	12.0 (1.0)
FN	13.3	12.0	15.3	8.0	12.2 (2.7)
MNN	12.7	12.4	11.4	3.6	10.0 (3.8)
FNN	13.3	10.0	5.6	6.9	9.0 (3.0)
	13.2 (0.3)	11.8 (1.1)	10.8 (3.5)	7.3 (2.5)	$\overline{\text{P@1}}(\sigma_{gd})$

XLM-R

P@1	EN	ES	DE	FR	
MN	16.7	13.3	20.7	16.7	16.9 (2.6)
FN	16.0	15.3	24.0	17.3	18.2 (3.5)
MNN	15.3	13.5	15.0	11.4	13.8 (1.5)
FNN	20.0	14.7	13.1	12.7	15.1 (3.0)
	17.0 (1.8)	14.2 (0.8)	18.2 (4.4)	14.5 (2.6)	$\overline{\text{P@1}}(\sigma_{gd})$

mT5

P@1	EN	ES	DE	FR	
MN	2.0	4.7	8.7	5.3	5.2 (2.4)
FN	4.0	3.3	6.7	3.3	4.3 (1.4)
MNN	2.0	4.7	6.4	4.3	4.4 (1.6)
FNN	3.3	6.7	1.9	6.2	4.5 (2.0)
	2.8 (0.9)	4.8 (1.2)	5.8 (2.5)	4.8 (1.1)	$\overline{\text{P@1}}(\sigma_{gd})$

Table 2: Results on P@1 score across groups (rows) and languages (columns), average performance in each language ($\overline{\text{P@1}}$) and standard deviation for group disparity (σ_{gd}). Cells are coloured language-wise. Cells with a darker background are language-wise above the average. Worst group performance in terms of group disparity (highest variance) is highlighted in red.

XLM-R seem to be biased towards native speakers because answers from *MN* and *FN* generally rank highest. Despite none of the models perform equally across groups, XLM-R shows a lower divergence across languages: Between-group differences are more than 50% smaller than with mBERT and mT5 when looking at the average MRR per language.

Table 3 gathers group level Spearman's ρ and average correlation per language. XLM-R predictions are more uniformly correlated across languages compared to mBERT, whose lexical preferences are better aligned in English and Spanish setups, and mT5, whose predictions correlate poorly with human cloze test answers. However, in line with previous results, the model exhibits bias towards male native speakers and *MNN* outlines as the worst performing group across languages, with a coefficient always below the average. Looking into the dimension of languages, German is the least aligned with human's answers in all models. Kendall's τ yields similar results. See Appendix A for details.

It is worth mentioning that our study does not aim to compare models' performance, but rather to motivate a discussion about the between-group and between-language differences within each model. The general low precision of mT5 outputs com-

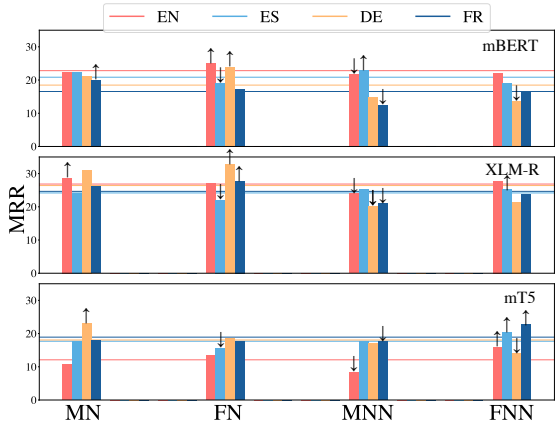

Figure 1: Average MRR (in percentage) per group in each language. Horizontal lines denote the average per language. Best-off (↑) and worst-off (↓) subgroups for each language are marked.

pared to human answers is likely due to the nature of the task itself. Because mT5 was trained with a span-mask denoising objective, it tends to complete the masked-out span with more than one token. When constraining generation to output one token, we are conditioning its default behaviour. Better correlation could be achieved by fine-tuning the model on completing cloze tests.

(Dis)agreement amongst annotators on the same language gives a measure of the difficulty of the task. French and German present a higher variability in the responses (with a vocabulary of 442 and 443 words respectively), compared to English (374 words), and Spanish (427 words), which reflects in a lower correlation with models' predictions.

5 Related Work

Multilingual PLMs have been analyzed in many ways: Researchers have, for example, looked at performance differences across languages (Singh et al., 2019; Wu and Dredze, 2020), looked at their organization of language types (Rama et al., 2020), used similarity analysis to probe their representations (Kudugunta et al., 2019), and investigated how learned self-attention in the Transformer blocks affects different languages (Ravishankar et al., 2021).

Previous work on fairness of multilingual models has, to the best of our knowledge, focused exclusively on task-specific models, rather than PLMs: Huang et al. (2020) evaluate the fairness of multilingual hate speech detection models, and several researchers have explored gender bias in multilingual models (Zhao et al., 2020; González et al., 2020). Dayanik and Padó (2021) consider the effects of adversarial debiasing in multilingual models.

mBERT

ρ	EN	ES	DE	FR
MN	0.33 (p=0.00)	0.23 (p=0.01)	-0.14 (p=0.09)	0.10 (p=0.21)
FN	0.27 (p=0.00)	0.07 (p=0.42)	-0.01 (p=0.89)	0.14 (p=0.08)
MNN	0.30 (p=0.00)	0.16 (p=0.03)	-0.10 (p=0.23)	0.08 (p=0.32)
FNN	0.37 (p=0.00)	0.16 (p=0.06)	0.03 (p=0.69)	0.08 (p=0.30)
Avg.	0.32 (p=0.00)	0.16 (p=0.00)	-0.05 (p=0.21)	0.10 (p=0.01)

XLM-R

ρ	EN	ES	DE	FR
MN	0.45 (p=0.00)	0.46 (p=0.00)	0.35 (p=0.00)	0.48 (p=0.00)
FN	0.30 (p=0.00)	0.35 (p=0.00)	0.45 (p=0.00)	0.33 (p=0.00)
MNN	0.30 (p=0.00)	0.38 (p=0.00)	0.22 (p=0.01)	0.32 (p=0.00)
FNN	0.40 (p=0.00)	0.48 (p=0.00)	0.11 (p=0.16)	0.36 (p=0.00)
Avg.	0.36 (p=0.00)	0.41 (p=0.00)	0.28 (p=0.00)	0.37 (p=0.00)

mT5

ρ	EN	ES	DE	FR
MN	0.01 (p=0.89)	0.14 (p=0.08)	0.14 (p=0.08)	0.25 (p=0.00)
FN	-0.12 (p=0.13)	0.13 (p=0.12)	0.00 (p=0.99)	0.14 (p=0.08)
MNN	-0.10 (p=0.22)	0.12 (p=0.11)	0.03 (p=0.74)	0.11 (p=0.18)
FNN	-0.07 (p=0.41)	0.28 (p=0.00)	0.04 (p=0.58)	0.11 (p=0.16)
Avg.	-0.07 (p=0.07)	0.17 (p=0.00)	0.05 (p=0.23)	0.15 (p=0.00)

Table 3: Correlation between groups of annotators (*MN, FN, MNN, FNN*) and models' predictions, classified by language. The degree of correlation is measured with Spearman's ρ coefficient ($\rho \in [-1, 1]$). Cells are coloured language-wise. Cells with a darker background show a stronger correlation compared to the average in each language. Samples highlighted in red fail to reject the null hypothesis, meaning that their difference is not statistically significant ($p > 0.05$).

Cloze tests were previously used in Zhang et al. (2021) to evaluate the fairness of English (monolingual) language models. In psycholinguistics, cloze tests have been performed with different age groups (Hintz et al., 2020) and native language (Stringer and Iverson, 2020), but these datasets have, to the best of our knowledge, not been used to evaluate language models.

6 Conclusion

In this paper, we present MozArt, a new multilingual dataset of parallel cloze examples with annotations from balanced demographics. This dataset is, to the best of our knowledge, the first to enable apples-to-apples comparison of group disparity of multilingual PLMs across languages. The dataset includes several demographic attributes, but we present preliminary experiments with gender and native language. We show that mBERT, XLM-R and mT5 are not equally fair across languages. For example, group disparities are much higher for German (and French) than for English and Spanish. This shows the importance of evaluating fairness across languages instead of stipulating from results for a single language. We further show that cloze test answers of female native speakers tend to rank

highest in both predictive PLMs. We followed best practices for mitigating the dangers of crowdsourcing (Karpinska et al., 2021; Kleijn et al., 2019) (see §2) and hope MozArt will be widely adopted and, over time, generate more results for other languages, PLMs and demographic attributes.

7 Limitations

As described in the paper, MozArt builds on top of another dataset, which is only available in four languages. The original dataset with its manual word alignments provided a unique opportunity to build MozArt in a way in which we could account for context, across languages. This of course limits our work to the languages provided. We acknowledge how multilingual studies of Indo-European languages may not generalize to languages outside this language family, and hope we or others will be able to contribute resources for a more diverse set of languages in the future.

Ethics Statement

The dataset released contains publicly available content from the proceedings of the European Parliament. Our work is based on sensitive information provided by the participants that took on our study in Prolific. The protected attributes collected are self-reported on a voluntary basis, and participants gave their consent to share them. In addition to the specific attributes analyzed in our study, which served as prescreening filters, Prolific also provides baseline data for all studies with the consent of participants to share it with researchers. For these base attributes, there might be gaps in the data because it is optional for participants to provide this information. These attributes are filled as *null* in the dataset. We performed a pilot study to determine the amount of time a task would take on average. The participants were paid based on time worked, and were given the option to opt out at any time of the study. Participants who revoked consent at any stage are not included in our study nor in the data released.

References

Kai-Wei Chang, Vinodkumar Prabhakaran, and Vicente Ordonez. 2019. Bias and fairness in natural language processing. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP): Tutorial Abstracts*, Hong Kong, China. Association for Computational Linguistics.

Rochelle Choenni and Ekaterina Shutova. 2020. What does it mean to be language-agnostic? probing multilingual sentence encoders for typological properties.

Monojit Choudhury and Amit Deshpande. 2021. How linguistically fair are multilingual pre-trained language models? In *AAAI-21*. AAAI, AAAI.

Alexis Conneau, Kartikay Khandelwal, Naman Goyal, Vishrav Chaudhary, Guillaume Wenzek, Francisco Guzmán, Edouard Grave, Myle Ott, Luke Zettlemoyer, and Veselin Stoyanov. 2020. Unsupervised cross-lingual representation learning at scale.

Erenay Dayanik and Sebastian Padó. 2021. Disentangling document topic and author gender in multiple languages: Lessons for adversarial debiasing. In *Proceedings of the Eleventh Workshop on Computational Approaches to Subjectivity, Sentiment and Social Media Analysis*, pages 50–61, Online. Association for Computational Linguistics.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. Bert: Pre-training of deep bidirectional transformers for language understanding.

Lucas Dixon, John Li, Jeffrey Sorensen, Nithum Thain, and Lucy Vasserman. 2018. Measuring and mitigating unintended bias in text classification. In *Proceedings of the 2018 AAAI/ACM Conference on AI, Ethics, and Society*, AIES '18, page 67–73, New York, NY, USA. Association for Computing Machinery.

Farahnaz Faez. 2011. Reconceptualizing the native/nonnative speaker dichotomy. *Journal of Language, Identity & Education*, 10(4):231–249.

Sahaj Garg, Vincent Perot, Nicole Limtiaco, Ankur Taly, Ed H. Chi, and Alex Beutel. 2019. Counterfactual fairness in text classification through robustness. *Proceedings of the 2019 AAAI/ACM Conference on AI, Ethics, and Society*.

Ana Valeria González, Maria Barrett, Rasmus Hvingelby, Kellie Webster, and Anders Søgaard. 2020. Type B reflexivization as an unambiguous testbed for multilingual multi-task gender bias. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 2637–2648, Online. Association for Computational Linguistics.

F Hintz, M Dijkhuis, V van Hoff, JM McQueen, and AS Meyer. 2020. A behavioural dataset for studying individual differences in language skills. *Scientific Data*, 7(1).

Matthew Honnibal and Ines Montani. 2017. spaCy 2: Natural language understanding with Bloom embeddings, convolutional neural networks and incremental parsing. To appear.

Dirk Hovy and Anders Søgaard. 2015. Tagging performance correlates with author age. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 2: Short Papers)*, pages 483–488, Beijing, China. Association for Computational Linguistics.

Xiaolei Huang, Linzi Xing, Franck Dernoncourt, and Michael J. Paul. 2020. Multilingual Twitter corpus and baselines for evaluating demographic bias in hate speech recognition. In *Proceedings of the 12th Language Resources and Evaluation Conference*, pages 1440–1448, Marseille, France. European Language Resources Association.

Marzena Karpinska, Nader Akoury, and Mohit Iyyer. 2021. The perils of using mechanical turk to evaluate open-ended text generation.

M. G. Kendall. 1938. A new measure of rank correlation. *Biometrika*, 30(1-2):81–93.

Suzanne Kleijn, Henk Pander Maat, and Ted Sanders. 2019. Cloze testing for comprehension assessment: The hytec-cloze. *Language Testing*, 36(4):553–572.

Philipp Koehn. 2005. Europarl: A parallel corpus for statistical machine translation. In *Proceedings of Machine Translation Summit X: Papers*, pages 79–86, Phuket, Thailand.

Philipp Koehn and Christof Monz. 2006. Manual and automatic evaluation of machine translation between European languages. In *Proceedings on the Workshop on Statistical Machine Translation*, pages 102–121, New York City. Association for Computational Linguistics.

Sneha Kudugunta, Ankur Bapna, Isaac Caswell, and Orhan Firat. 2019. Investigating multilingual NMT representations at scale. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 1565–1575, Hong Kong, China. Association for Computational Linguistics.

Ninareh Mehrabi, Fred Morstatter, Nripsuta Saxena, Kristina Lerman, and Aram Galstyan. 2021. A survey on bias and fairness in machine learning. *ACM Comput. Surv.*, 54(6).

Ji Ho Park, Jamin Shin, and Pascale Fung. 2018. Reducing gender bias in abusive language detection. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 2799–2804, Brussels, Belgium. Association for Computational Linguistics.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J. Liu. 2020. Exploring the limits of transfer learning with a unified text-to-text transformer.

Taraka Rama, Lisa Beinborn, and Steffen Eger. 2020. Probing multilingual BERT for genetic and typological signals. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 1214–1228, Barcelona, Spain (Online). International Committee on Computational Linguistics.

Vinit Ravishankar, Artur Kulmizev, Mostafa Abdou, Anders Søgaard, and Joakim Nivre. 2021. Attention can reflect syntactic structure (if you let it). In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*, pages 3031–3045, Online. Association for Computational Linguistics.

John Schmitz. 2016. On the native/nonnative speaker notion and world englishes: Debating with k. rajagopalan. *DELTA: Documentação de Estudos em Lingüística Teórica e Aplicada*, 32:597–611.

Jasdeep Singh, Bryan McCann, Richard Socher, and Caiming Xiong. 2019. BERT is not an interlingua and the bias of tokenization. In *Proceedings of the 2nd Workshop on Deep Learning Approaches for Low-Resource NLP (DeepLo 2019)*, pages 47–55, Hong Kong, China. Association for Computational Linguistics.

C. Spearman. 1987. The proof and measurement of association between two things. *The American Journal of Psychology*, 100(3/4):441–471.

Louise Stringer and Paul Iverson. 2020. Non-native speech recognition sentences: A new materials set for non-native speech perception research. *Behavior Research Methods*, 52(2).

Wilson L. Taylor. 1953. Cloze procedure: A new tool for measuring readability. *Journalism & Mass Communication Quarterly*, 30:415 – 433.

Ada Wan. 2022. Fairness in representation for multilingual NLP: Insights from controlled experiments on conditional language modeling. In *International Conference on Learning Representations*.

Jialu Wang, Yang Liu, and Xin Eric Wang. 2021. Assessing multilingual fairness in pre-trained multimodal representations.

Shijie Wu and Mark Dredze. 2020. Are all languages created equal in multilingual BERT? In *Proceedings of the 5th Workshop on Representation Learning for NLP*, pages 120–130, Online. Association for Computational Linguistics.

Linting Xue, Noah Constant, Adam Roberts, Mihir Kale, Rami Al-Rfou, Aditya Siddhant, Aditya Barua, and Colin Raffel. 2021. mt5: A massively multilingual pre-trained text-to-text transformer.

Sheng Zhang, Xin Zhang, Weiming Zhang, and Anders Søgaard. 2021. Sociolectal analysis of pretrained language models. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 4581–4588, Online and Punta Cana,

Dominican Republic. Association for Computational Linguistics.

Jieyu Zhao, Subhabrata Mukherjee, Saghar Hosseini, Kai-Wei Chang, and Ahmed Hassan Awadallah. 2020. Gender bias in multilingual embeddings and cross-lingual transfer. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 2896–2907, Online. Association for Computational Linguistics.

Jieyu Zhao, Tianlu Wang, Mark Yatskar, Vicente Ordonez, and Kai-Wei Chang. 2017. Men also like shopping: Reducing gender bias amplification using corpus-level constraints.

A Additional results

In this section, we provide additional analysis results of the PLM's performance on MozArt. We report precision at 5 (P@5), which corresponds to the number of relevant answers amongst the top 5 candidates. It provides a more flexible metric for measuring model alignments with open-ended text answers, but fails to take into account the exact position within the top-k. Considering the top-5, the bias towards native speakers is diminished especially in English and Spanish, despite being *MNN* and *FNN* the worst groups –in terms of group disparity– in mBERT and XLM-R respectively. At the same time, the group disparities are exacerbated as shown in Table 4.

Table 5 complements results on correlation of the alignment of group responses. It shows Kendall's τ coefficient. Conclusions remain almost the same as studied with Spearman's coefficient, albeit non-native subgroups in Spanish are more correlated in mBERT.

B t-SNE

To give a brief overview of the semantic multilinguality encoded in the pretrained models, we run several representations with t-SNE. Figure 2 and Figure 3 represent the top-1000 predictions in a t-SNE plot for mBERT and XLM-R respectively. The same sentence is queried to the model in four languages and, accordingly, to annotators:

en We want to [MASK] innovation .
es Queremos [MASK] la innovación .
de Wir wollen zur Innovation [MASK] .
fr Nous voulons [MASK] l'innovation .

Highest scored predictions are highlighted with a ($\star$). Annotator's answers that fell into the top-1000 predictions are denoted with a black edge. In

mBERT

P@5	EN	ES	DE	FR	
MN	30.7	26.7	22.0	24.0	25.9 (3.3)
FN	32.0	18.7	24.7	22.0	24.4 (4.9)
MNN	34.0	25.9	12.1	15.0	21.8 (8.7)
FNN	32.7	25.3	16.3	16.3	22.7 (6.9)
	32.3 (1.2)	24.2 (3.1)	18.8 (4.9)	19.3 (3.8)	$\overline{\text{P@5}}(\sigma_{gd})$

XLM-R

P@5	EN	ES	DE	FR	
MN	39.3	30.7	34.7	32.7	34.4 (3.2)
FN	30.7	25.3	38.0	35.3	32.3 (4.8)
MNN	30.7	29.4	22.1	25.4	26.9 (3.4)
FNN	36.7	34.0	19.4	26.9	29.3 (6.7)
	34.3 (3.8)	29.8 (3.1)	28.5 (7.9)	30.3 (4.1)	$\overline{\text{P@5}}(\sigma_{gd})$

mT5

P@5	EN	ES	DE	FR	
MN	10.0	12.7	16.0	11.3	12.5 (2.2)
FN	11.3	10.0	16.7	18.0	14.0 (3.4)
MNN	6.0	11.8	9.3	10.7	9.5 (2.2)
FNN	13.3	16.0	8.7	15.0	13.3 (2.8)
	10.2 (2.7)	12.6 (2.2)	12.7 (3.7)	13.8 (3.0)	$\overline{\text{P@5}}(\sigma_{gd})$

Table 4: Results on P@5 score across groups and languages, average performance in each language ($\overline{\text{P@5}}$) and standard deviation for group disparity (σ_{gd}). Cells are coloured language-wise. Cells with a darker background are language-wise above the average. Worst group performance in terms of group disparity (highest variance) is highlighted in red.

line with results in (Choenni and Shutova, 2020), we observe that languages are mostly projected in separate sub-spaces instead of yielding a neutral representation, even though they share a common space (vocabulary).

Similarly, Singh et al. (2019) shown a trend towards dissimilarity between representations for semantically similar inputs in different languages, in deeper layers of an uncased mBERT. Serve Figure 4 as an example, where the same word "gases" was answered in different languages but is represented in different subspaces. Figure 5 shows a similar behaviour in XLM-R. The sentences queried are:

en [MASK] that deplete the ozone layer
es [MASK] que agotan la capa de ozono
de [MASK], die zum Abbau der Ozonschicht führen
fr [MASK] appauvrissant la couche d'ozone

mBERT

τ	EN	ES	DE	FR
MN	0.27 (p=0.00)	0.19 (p=0.00)	-0.09 (p=0.15)	0.09 (p=0.16)
FN	0.23 (p=0.00)	0.07 (p=0.24)	0.01 (p=0.89)	0.13 (p=0.04)
MNN	0.25 (p=0.00)	0.15 (p=0.01)	-0.06 (p=0.32)	0.07 (p=0.28)
FNN	0.29 (p=0.00)	0.14 (p=0.01)	0.03 (p=0.57)	0.06 (p=0.27)
Avg.	0.26 (p=0.00)	0.14 (p=0.00)	-0.03 (p=0.41)	0.09 (p=0.01)

XLM-R

τ	EN	ES	DE	FR
MN	0.40 (p=0.00)	0.43 (p=0.00)	0.32 (p=0.00)	0.45 (p=0.00)
FN	0.26 (p=0.00)	0.33 (p=0.00)	0.43 (p=0.00)	0.31 (p=0.00)
MNN	0.26 (p=0.00)	0.35 (p=0.00)	0.20 (p=0.01)	0.29 (p=0.00)
FNN	0.35 (p=0.00)	0.45 (p=0.00)	0.10 (p=0.15)	0.34 (p=0.00)
Avg.	0.32 (p=0.00)	0.39 (p=0.00)	0.25 (p=0.00)	0.34 (p=0.00)

mT5

τ	EN	ES	DE	FR
MN	0.02 (p=0.79)	0.13 (p=0.06)	0.13 (p=0.06)	0.21 (p=0.00)
FN	-0.09 (p=0.16)	0.11 (p=0.11)	0.00 (p=0.98)	0.12 (p=0.08)
MNN	-0.08 (p=0.21)	0.10 (p=0.10)	0.03 (p=0.69)	0.10 (p=0.17)
FNN	-0.04 (p=0.51)	0.25 (p=0.00)	0.03 (p=0.61)	0.10 (p=0.15)
Avg.	-0.07 (p=0.07)	0.15 (p=0.00)	0.05 (p=0.18)	0.13 (p=0.00)

Table 5: Correlation between groups of annotators (*MN, FN, MNN, FNN*) and models' predictions, classified by language. The degree of correlation is measured with Kendall's τ coefficient ($\tau \in [-1, 1]$). Cells are coloured language-wise. Cells with a darker background show a stronger correlation compared to the average in each language. Samples highlighted in red fail to reject the null hypothesis, meaning that their difference is not statistically significant ($p > 0.05$).

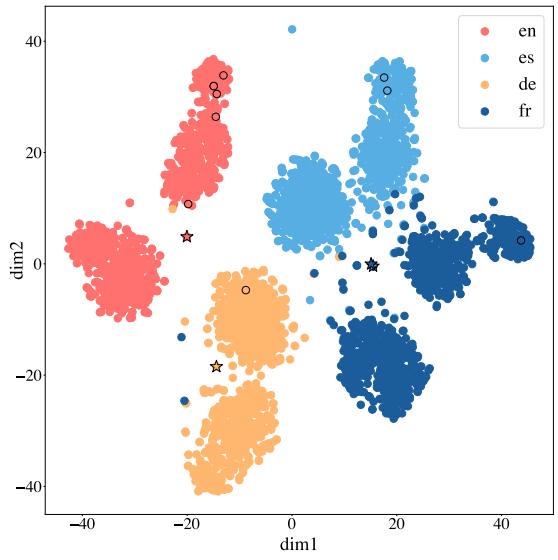

Figure 2: t-SNE representation from the last layer of mBERT for the top-1000 predictions for the parallel sentences in the list above ("We want to [MASK] innovation ." in English). Highest scored prediction is starred; annotator's answers are denoted by a dot with black edge. Legend shows language-color mapping.

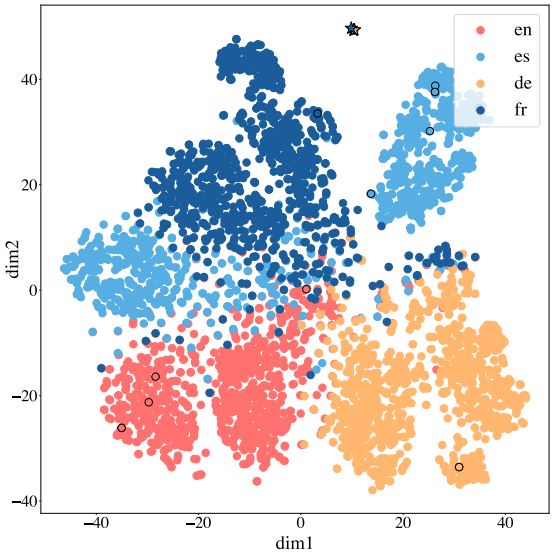

Figure 3: t-SNE representation from the last layer of XLM-R for the top-1000 predictions for the parallel sentences in the list above ("We want to [MASK] innovation ." in English). Highest scored prediction is starred; annotator's answers are denoted by a dot with black edge. Legend shows language-color mapping.

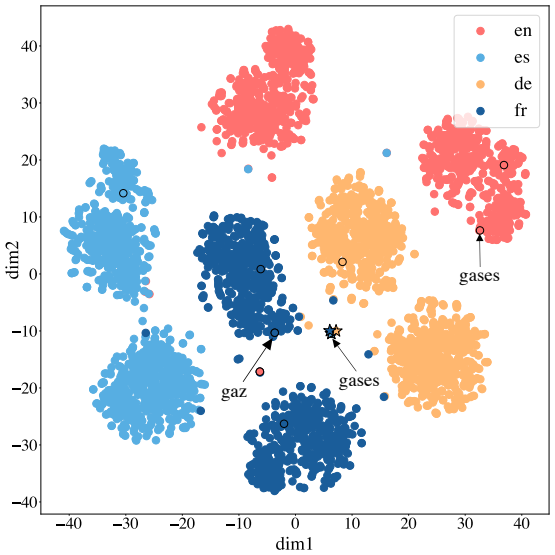

Figure 4: t-SNE representation from the last layer of mBERT for the top-1000 predictions for the parallel sentences in the list above ("[MASK] that deplete the ozone layer" in English). The word "gases" is pointed out in each language (en: gases, es: gases, fr: gaz), as it was a recurrent answer from different annotators. Highest scored prediction in each language is starred; annotator's answers are denoted by a dot with black edge. Legend shows language-color mapping.

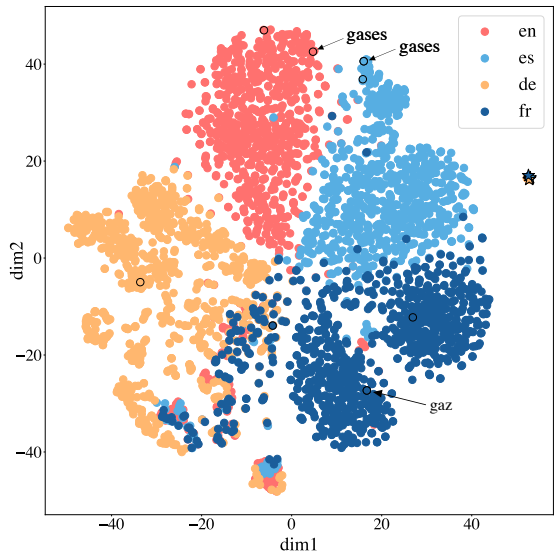

Figure 5: t-SNE representation from the last layer of XLM-R for the top-1000 predictions for the parallel sentences in the list above ("[MASK] that deplete the ozone layer" in English). The word "gases" is pointed out in each language (en: gases, es: gases, fr: gaz), as it was a recurrent answer from different annotators. Highest scored prediction in each language is starred; annotator's answers are denoted by a dot with black edge. Legend shows language-color mapping.

Possible Stories: Evaluating Situated Commonsense Reasoning under Multiple Possible Scenarios

Mana Ashida*
Yahoo Japan Corporation
maashida@yahoo-corp.jp

Saku Sugawara
National Institute of Informatics
saku@nii.ac.jp

Abstract

The possible consequences for the same context may vary depending on the situation we refer to. However, current studies in natural language processing do not focus on situated commonsense reasoning under multiple possible scenarios. This study frames this task by asking multiple questions with the same set of possible endings as candidate answers, given a short story text. Our resulting dataset, Possible Stories, consists of more than 4.5K questions over 1.3K story texts in English. We discover that even current strong pretrained language models struggle to answer the questions consistently, highlighting that the highest accuracy in an unsupervised setting (60.2%) is far behind human accuracy (92.5%). Through a comparison with existing datasets, we observe that the questions in our dataset contain minimal annotation artifacts in the answer options. In addition, our dataset includes examples that require counterfactual reasoning, as well as those requiring readers' reactions and fictional information, suggesting that our dataset can serve as a challenging testbed for future studies on situated commonsense reasoning.

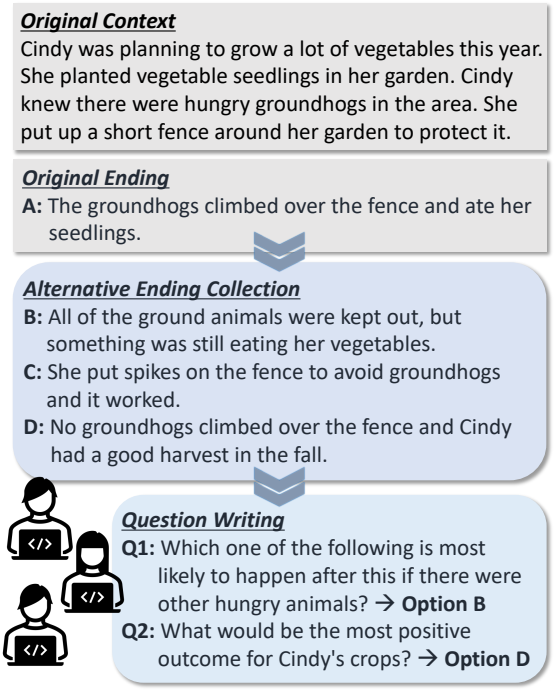

Figure 1: Overview of Possible Stories and its creation process. We ask crowdworkers to produce alternative endings given a story text, and then write multiple-choice questions that have a single correct answer among the original and three collected endings.

1 Introduction

Commonsense reasoning, inclusive of counterfactual, abductive, and monotonic reasoning, is a core element of language understanding. Researchers are interested in whether these abilities can be learned in systems, and several benchmarks have been proposed to investigate machine commonsense reasoning (Huang et al., 2019; Sap et al., 2019; Aggarwal et al., 2021; Saha et al., 2021). Recent pretrained models have shown competitive results (Khashabi et al., 2020; Lourie et al., 2021).

Commonsense reasoning has often been framed as a task to infer whether candidate answers are plausible, such as in the multiple-choice format (Talmor et al., 2019; Sakaguchi et al., 2020). The difference between plausible and implausible answers is expected to be salient enough that it can be established as a classification task. However, when making day-to-day decisions, people consider several plausible choices, rather than clearly plausible and implausible ones, depending on one's situation and method of thinking. However, the tasks concerning conditions *under multiple plausible scenarios* are few, and their domains are limited to, for example, factual information that differs according to place and time (Zhang and Choi, 2021) or human behaviors that are either normative or divergent (Emelin et al., 2021). Another example is the natural language inference or commonsense reasoning task that considers variations in human

*Work done while at Tokyo Metropolitan University.

Proceedings of the 29th International Conference on Computational Linguistics, pages 3606–3630
October 12–17, 2022.

opinions (Zhang et al., 2017; Chen et al., 2020b), which allows for the differences in annotations due to one's mentality (Pavlick and Kwiatkowski, 2019; Meissner et al., 2021). Our aim here is to interrogate these types of situated reasoning in more comprehensive settings, such as in story texts.

To assess the possible extent of situated reasoning in machines, we introduce *Possible Stories*, a benchmark consisting of 4,533 multiple-choice questions over 1,313 passages in English, to evaluate commonsense reasoning over multiple possible alternatives for single passages. Figure 1 shows an example. Given a story text, we aggregate alternative endings and multiple-choice questions that contain information such that they guide the determination of the most likely ending. By design, machines cannot rely only on answer options but also have to understand the condition implied by each question to answer correctly because all options are expected to be possible. This dataset creation procedure tackles the known issue of annotation artifacts (Gururangan et al., 2018) in crowdsourced datasets by using alternative endings, instead of *right* and *wrong* endings, and by compiling the endings and questions from multiple crowdworkers.

We evaluate strong pretrained language models and heuristic methods on our dataset and observe that in an unsupervised setting, even the strongest model (DeBERTa large v3; He et al., 2021) underperforms compared to humans by approximately 30% accuracy and more than 50% consistency score (i.e., passage-wise accuracy). Our analysis using input ablation and statistical significance tests highlights that the annotation artifacts contained in the answer options of our dataset questions are much fewer than those in existing multiple-choice datasets such as RACE (Lai et al., 2017) and CosmosQA (Huang et al., 2019). Reasoning-type annotation shows that more than 60% of our dataset questions require counterfactual reasoning, as well as an understanding of characters' motivations and reactions, readers' perceptions, and fictional information.

Our contributions are summarized as follows:[1]

- We propose a situated commonsense reasoning task and create a multiple-choice question answering (QA) dataset using plausible story endings, together with questions as multiple conditions where one of the endings becomes the most plausible.
- We discover that current strong pretrained language models struggle to solve our task when training data are unavailable, indicating that there is room for future improvement on situated commonsense reasoning.
- We show that our dataset contains minimal annotation artifacts in the answer options and has many challenging questions that require counterfactual reasoning and an understanding of characters' motivations and reactions, readers' perceptions, and fictional information.

2 Background and Related Works

Our work is motivated by recent efforts to create evaluation frameworks for commonsense reasoning situated in extra-linguistic contexts.

Benchmarks for Commonsense Reasoning Many commonsense reasoning resources have been proposed that target reading comprehension (Huang et al., 2019), cloze tests regarding story endings (Zellers et al., 2019) or in-between events (Bhagavatula et al., 2020), and inferences on social interactions (Sap et al., 2019). Mostafazadeh et al. (2016) propose a task similar to ours, but it differs in that ours has four possible ending options, rather than a plausible and implausible completion pair.

Benchmarks for Counterfactual Reasoning Researchers have coined the term *counterfactual reasoning* to refer to the property of reasoning over hypothetical events and have proposed benchmarks to evaluate the counterfactual reasoning ability of machines. Tandon et al. (2019) collect questions that explicitly ask *what if*, based on procedural texts. Qin et al. (2019) propose a task of generating a counterfactual story ending that is minimally edited from the original ending, given modified events in the context. Our data creation process is similar in terms of using an existing story and modifying a segment; however, we ask crowdworkers to change the segment more drastically, yielding diverse story endings.

Evaluation of Understanding of Situations Reasoning over multiple possibilities, depending on the situation, can be regarded as *situated reasoning*. Recent studies have attempted to integrate situational information into the context used in downstream tasks, such as question answering on factual

[1]The details of our data collection and final outcome including all collected story endings are available at https://github.com/nii-cl/possible-stories.

information (Zhang and Choi, 2021) and consequence or normative action generation given real-world social settings (Emelin et al., 2021). Story Commonsense (Rashkin et al., 2018) provides an annotated dataset of motivation and emotional reactions. (Forbes et al., 2020) collect general rules of thumb about actions. The range of situations that we consider goes beyond facts and normative settings, aiming to consider readers' beliefs, causality, and characters' emotions.

Probing of Language Models The use of contrastive examples to probe language models' knowledge and inductive biases is an active area of research. This line of research typically uses pairs of sentences with minimum differences (Marvin and Linzen, 2018; Li et al., 2020; Warstadt et al., 2020), contrastive sets to identify the model's decision boundary (Gardner et al., 2020), or adversarial examples (Jia and Liang, 2017) to identify the segments that contribute to changing model behaviors. By contrast, we use multiple plausible choices for a single passage to study what causes models to assign higher probabilities to certain choices.

3 Task Description

Motivation In Story Cloze Test, Mostafazadeh et al. (2016) use *right* and *wrong* endings to evaluate machines' story understanding, assuming that the right ending can be regarded as an entailing hypothesis in a textual entailment framework and the wrong ending as a contradicting hypothesis. During data collection, the workers are instructed to produce endings that are realistic and sensible for right endings, and wrong endings are chosen from those that are rated lower than right endings in terms of meaningfulness and coherence. Consequently, their task is created to have clear right and wrong endings. However, in reality, there is an infinite number of possibilities of clearly plausible endings. By creating informative questions positing situations rather than questions asking about relative plausibility without any conditions, we aim to test machines' story understanding in multiple scenarios that provide additional information that can discriminate one plausible ending from other possible endings.

Task Formulation We formulate the task as a multiple-choice question with a passage and answer options, where the answer options depict possible endings of the passage. Given a story s, the task is to determine the most plausible story ending among the four possible endings $E = (e_1, \ldots, e_4)$ under the condition c that is implied by a question. To further evaluate the models' understanding of situations, we also define the task of predicting the most plausible outcome for multiple conditions. Given s, the task is to determine the most plausible story ending among E for each of the multiple conditions $C = (c_1, \ldots, c_4)$ that are implied by multiple questions. When the models capture all the relationships between conditions and plausible endings correctly, we assume that the models reason over a finite number of possible consequences and the relationships among them. We call this *consistency*, which reports the percentage of a model's outputs that are correct for all questions referring to a unique context. This evaluation is inspired by the study of contrastive sets (Gardner et al., 2020).

4 The Possible Stories Dataset

Context Passages To collect story texts, we use ROCStories (Mostafazadeh et al., 2016), a corpus of five-sentence stories. The first to fourth sentences describe the context, and the final sentence, the ending, concludes the story. We choose ROCStories because each of its stories has a clear beginning and ending, while being generic enough to come up with different endings. The details on our story selection criteria are provided in Appendix A.

The following tasks are carried out by the crowdworkers in Amazon Mechanical Turk (MTurk) who perform above certain levels during our worker recruitment phase, which is designed to be fairer than the conventional qualifications used in MTurk. The details of the worker recruitment are provided in Appendix B. The instructions and task interface presented to the workers are also provided in Appendix I.

4.1 Writing Tasks

Ending Writing We first ask workers to create two alternative endings given a story with the original ending. The participants are encouraged to be as creative as possible so that possible yet unrealistic story endings can also be elicited. We collect six to eight alternative endings by asking three or four workers to produce two endings per passage.

Selection of Ending Options Having collected six to eight alternative endings, we need to decide which three options to use in our questions, in addition to the original ending.

Depending on how they are chosen, there may be differences in the difficulty of the generated questions. For example, if the endings are similar to each other, it will be difficult to create questions that have only one correct choice among four endings. Conversely, if the endings are completely different, the questions may be easier to create, but machines may rely solely on semantic similarity between passage and endings, without requiring commonsense reasoning.

To examine the relationship between question difficulty and the diversity of the chosen endings, we run a pilot task using ten randomly selected stories with six different sets of endings. The six sets are chosen based on the sum of cosine similarity calculated based on the embeddings (Reimers and Gurevych, 2019) of all the possible combinations of endings, ranging from the set of endings that are most similar to the most diverse set. Six sets are chosen such that the distance between the values of the sum of the cosine similarities of one set and another set is equal. Through a validation step to identify which sets of endings enable high-quality multiple-choice questions, we decide on the set that contains the second most diverse endings among the six sets upon consideration.

Question Writing As four distinct endings are gathered per passage, we ask the workers to write questions in which only one of the four endings is the correct answer. Because this task is more complex, we select participants via a qualification task, targeting those participants who maintain quality in the ending writing task. It is up to the workers to decide the correct option, considering the difference in difficulty in writing questions with certain story endings. Four questions are written per passage by two workers, two per worker, and the answers to each set of two questions are different to maintain the diversity of the correct answers. To ensure that the distribution of the dataset is natural (Bowman and Dahl, 2021; Kaushik et al., 2021) and that the questions fit the general purpose, we avoid collecting questions in an adversarial manner (Bartolo et al., 2020).

4.2 Data Validation

The goal of the validation task is to verify that there is one correct answer for each question, and that the questions do not contain any objectionable or personal content. Questions that do not meet these criteria are discarded. The detailed validation results and further quality control over the collection batches are reported in Appendix C.

Question-Answer Validation In this step, we ask workers to answer multiple-choice questions. The workers choose one of the four endings and four additional options (no answer, more than two possible answers, ill-formed questions, and others). Each question is validated by three workers, and we retain questions in which the majority vote is identical to the writer's answer.

Content Validation During the validation task, we ask workers to indicate negative stereotypes or biased descriptions of certain social groups. We discard questions that the workers claim contain unfair descriptions. This process prevents the perpetuation of unethical opinions in downstream tasks when this dataset is used for training models. Some of the workers' inputs are discussed in Ethical Considerations. We incentivize workers with a bonus of $0.3 per completion of the free-text form.

4.3 Dataset Statistics

Our dataset, Possible Stories, has 8,885 alternative endings for 1,313 passages and 4,533 multiple-choice questions with the original ending and three alternative endings as answer options. Table 1 presents the basic statistics for the resulting dataset. Although the passages are shorter than those in CosmosQA (70.3) and RACE (321.9), the questions (14.2) and answer options (15.3) are quite longer than others (CosmosQA is 10.6 and 8.1 and RACE is 10.0 and 5.3), which could potentially make questions difficult (Nangia et al., 2021).

In addition, as shown in Table 2, more than 50% of the contexts have questions with three or four distinct correct answer choices. This contributes to the assessment of the models' comprehension of multiple situations using the consistency metric.

One of our main goals for constructing a benchmark is to test the models' capacity for situated commonsense reasoning over multiple scenarios as an unseen task. Nonetheless, to ensure that it is feasible to model our task using current strong pretrained language models (Liu et al., 2019a), we follow a standard approach to split the collected examples into training (75%), dev (10%), and test (15%) sets. To investigate the model generalizability, the passages do not overlap between each set. The dev and test sets do not contain questions produced by workers who have received negative comments from other workers to ensure quality.

Split	#Question	#Passage	#Q/P	Passage len	Question len	Option len
Train	3,404	984	3.46	46.1	13.9	15.4
Dev	458	133	3.44	46.2	14.9	15.3
Test	671	196	3.42	47.0	15.0	15.2
Total	4,533	1,313	3.45	46.3	14.2	15.3

Table 1: Statistics of Possible Stories. *Q* and *P* indicate question and passage. *#Q/P* indicates the average number of questions per passage. *Len* indicates the average number of tokens.

	Distinct # of answers				
#Q/P	1	2	3	4	total
1	2.1				2.1
2	1.5	8.4			9.9
3		16.8	12.0		28.8
4		18.1	34.7	6.5	59.3
Total	3.6	43.3	46.7	6.5	100.0

Table 2: Distribution (%) of the number of questions per passage and the distinct number of correct answers.

5 Experiments

5.1 Models and Settings

For modern pretrained language models, we use BERT (base and large; Devlin et al., 2019), RoBERTa (base and large; Liu et al., 2019b), and DeBERTa (base and large of v3; He et al., 2021). In our standard setting (i.e., unsupervised), to adapt these models to the multiple-choice task, we fine-tune them on the RACE dataset (Lai et al., 2017), which is a large-scale dataset of middle- and high-school English exams and has passages and questions on various topics.[2] In the supervised setting, the models are directly trained on our training set unless mentioned otherwise. To establish different baseline methods, we consider simple heuristics using perplexity, semantic similarity, and entailment scores. For perplexity heuristics, we use GPT-2 (Radford et al., 2019) and GPT-Neo (Black et al., 2021) to obtain the perplexity of the inputs and consider options with the smallest perplexity as a model's prediction. Sentence similarity uses representations obtained from the sentence transformers (Reimers and Gurevych, 2019) to compute the cosine similarity between the options and the rest of the input. The candidate with the highest similarity score is regarded as the model prediction. The entailment score is calculated using RoBERTa-large fine-tuned on MNLI (Williams et al., 2018), and the option with the highest entailment score when taking the inputs as the premise is chosen.

[2]We observe that models fine-tuned on CosmosQA are consistently inferior to those fine-tuned on RACE (Appendix D).

5.2 Results

Human Performance To measure the human performance on our test set, we collect three additional labels for all questions from different crowdworkers who do not join the validation task. We ensure that the same set of three workers answer the questions belonging to a single story. For computing accuracy, we take the majority of the three labels to determine whether it is equal to the validated gold label. For computing consistency, we determine whether the majority vote answers are correct for all questions in each passage (Table 3).

Model Performance When the training set is unavailable, we observe that DeBERTa-large achieves the best performance. Although this model is fine-tuned on RACE, which has a sufficient number of diverse training examples, the model performance is far from that of humans, showing large gaps of 29.5% and 53.0% in terms of accuracy and consistency, respectively. Out of the four simple heuristics models, those using perplexity and semantic similarity perform above the chance rate of 25%, indicating that those features, while inadequate, might be useful in finding the correct answers. By contrast, the entailment score-based model falls short of 25%. This result highlights the uniqueness of our dataset, as it shows that relying on monotonic reasoning cannot lead to a correct answer.

Supervised Performance With the training data, we observe that DeBERTa-large performs better than the other models, and it achieves the best accuracy and consistency when fine-tuned using RACE (Table 3). These scores are very close to those of humans, which implies that the task can be feasibly performed by a strong model, given sufficient train-

FT	Model	Acc.	Consist.
✗	DeBERTa-large*	**60.2**	**19.9**
	DeBERTa-base*	45.3	8.2
	RoBERTa-large*	50.5	13.8
	PPL. GPT-2 large	30.4	2.0
	PPL. GPT-Neo 2.7B	29.5	2.6
	Semantic Sim.	37.3	4.1
	Entailment	23.1	2.0
✔	DeBERTa-large*	**92.1**	**74.7**
	DeBERTa-large	88.5	67.3
	DeBERTa-base	81.5	51.5
	RoBERTa-large*	83.5	55.6
	RoBERTa-large	81.7	49.5
	RoBERTa-base	72.0	30.6
	BERT-large	62.6	20.4
	BERT-base	57.3	16.3
	Human	92.5	76.5

Table 3: Model and human performances on our dataset. *Acc.* and *consist.* denote accuracy (%) and consistency (%). (*) indicates that the model is fine-tuned on RACE. *FT* indicates whether the models are fine-tuned on the training set. The experimental details are reported in Appendix E.

FT	Model	Full	No pas.	No ques.
✗	DeBERTa-L*	60.2	58.1	21.8
	RoBERTa-L*	50.5	50.3	21.5
	GPT-2 large	30.4	35.2	26.4
	Semantic Sim.	37.3	47.1	28.8
✔	DeBERTa-L*	92.1	87.0	31.8
	DeBERTa-L	88.5	86.4	33.4
	BERT-L	62.6	51.1	30.4

Table 4: Input ablation results (accuracy; %). *No pas.* and *no ques.* indicate that the context passage and question are ablated from the input, respectively.

Model	Ours	Cosmos	QuAIL	MC-adv
DeBERTa-L	60.2	66.8	76.3	81.2
DeBERTa-B	45.3	56.0	66.2	69.0
RoBERTa-L	50.5	64.2	70.3	69.1
Human	92.5	94.0	100.0	92.0
Acc. gap	40.5	31.7	29.1	18.9

Table 5: Human performance, model performance without fine-tuning (accuracy; %), and the gap between the human performance and the average model performance (larger values imply higher difficulty). Model-L and -B indicate the large and base models respectively.

ing data. Nonetheless, it is notable that BERT-large and RoBERTa-large, which were state-of-the-art models only several years ago, show potential for improvement ($\approx$ 30% and 10% accuracy, respectively) compared to humans.

Input Ablation Table 4 presents the input ablation analysis. When ablating the passages, we observe that pretrained language models fine-tuned on any multiple-choice dataset show lower performance than those with the full input. Regarding the heuristics methods, we find that having the context does not significantly change the ranking of the endings. When ablating the questions, we observe that the performance of all models decreases, which is expected because the same set of answer options has multiple questions in our dataset.

6 Analysis

6.1 Human–model Performance Gap

To investigate the relative difficulty of our dataset among multiple-choice QA datasets, we compare the accuracy gap between humans and models in an unsupervised setting with existing datasets, including CosmosQA (Cosmos; we report the validation result because the test labels are not available), QuAIL (Rogers et al., 2020, challenge set), and the examples provided by Sugawara et al. (2022) (MC-adv; multiple-choice questions that are written by crowdworkers in an adversarial manner). We use three models (DeBERTa-large and -base and RoBERTa-large) fine-tuned on RACE. The results in Table 5 demonstrate that our dataset may be more challenging than the multiple-choice reading comprehension datasets we analyze, despite the simplicity of our data collection method.

6.2 Annotation Artifacts in Answer Options

One of our motivations for crowdsourcing multiple questions for the same set of answer options is to minimize superficial patterns (i.e., annotation artifacts) in the collected examples, especially in their answer options. To validate this, we first compare the supervised performance in three ablation settings (no context passage, no question, and answer options only). We use DeBERTa-large and report the test score on our dataset, RACE, QuAIL, and CosmosQA. Table 6 shows that although the no-context performance on our dataset

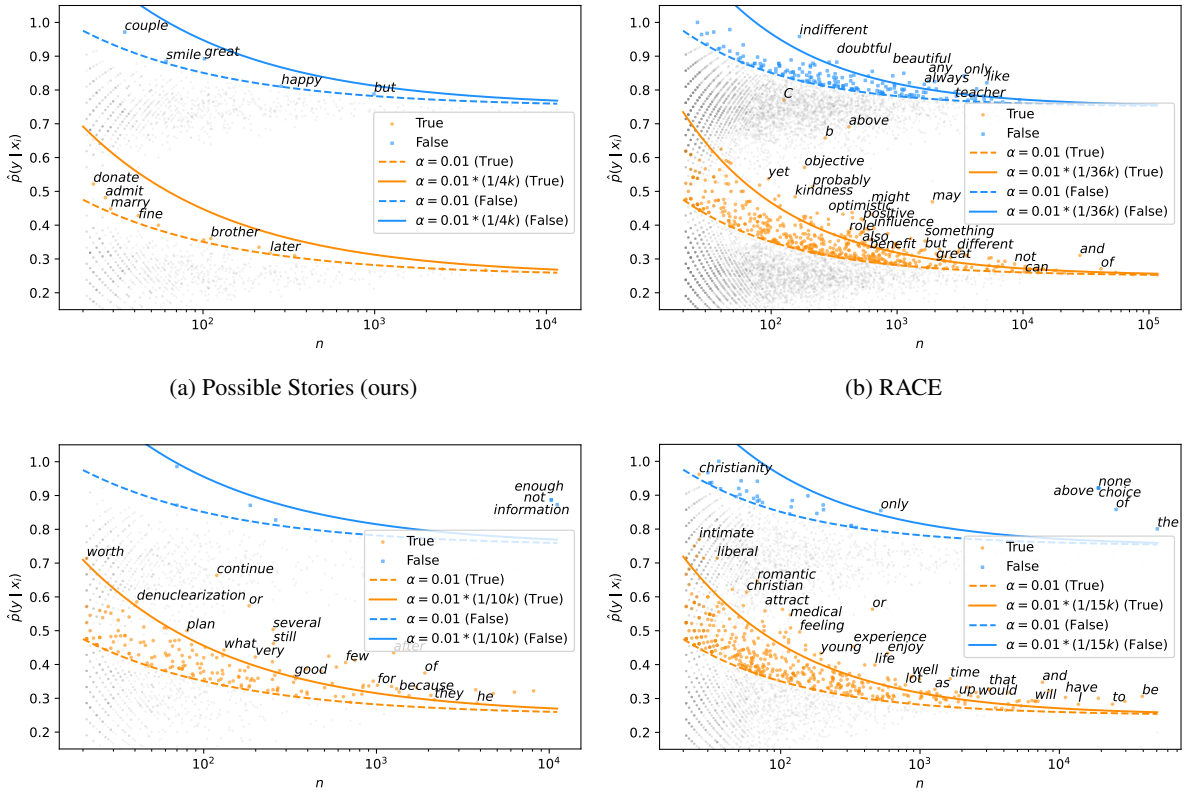

(a) Possible Stories (ours)

(b) RACE

(c) QuAIL

(d) CosmosQA

Figure 2: Token-level annotation artifacts in the training examples of our dataset, RACE, QuAIL, and CosmosQA. All tokens are below $\alpha = 0.01$ with a conservative Bonferroni correction for 3,990, 15,472, 35,762, and 9,688 vocabulary items, respectively.

Dataset	Full	No pas.	No ques.	Opt. only
Ours	88.5	86.4	33.4	29.1
RACE	87.9	60.1	69.8	46.6
QuAIL	81.7	51.8	58.3	39.6
Cosmos	87.8	72.5	59.4	57.2

Table 6: Supervised accuracy (%) by DeBERTa-large (v3) on the input-ablation settings.

is relatively high, the no-question and option-only performances are lower than the others. This result implies that the question and answer options in our dataset are mutually indispensable for predicting the correct answer, while in the other datasets, the options on their own and their relationship with the context are informative for the prediction.

To visualize the actual tokens that create annotation artifacts, we follow Gardner et al. (2021), who propose analyzing token-level features in terms of the empirical probability of labels $\hat{p}(y|x_i)$ given a specific token (vocabulary item) x_i appearing in input X. Here, the label y indicates whether an answer option is the correct (True) or not (False). We plot the probability $\hat{p}(y|x_i)$ and the number of occurrences (n) for the tokens of the training questions in our dataset, RACE, QuAIL, and CosmosQA (Figures 2) for comparison. To see if the null-hypothesis (i.e., the token does not co-occur with a specific label) is accepted or rejected, we compute z-statistics and plot the level of statistical significance $\alpha = 0.01$ and its conservative Bonferroni correction (Bonferroni, 1936) for the vocabulary items ($\alpha = 0.01/|V|$). We find that for the true label in our dataset, only 12 tokens are above $\alpha = 0.01$ and no tokens are above $\alpha = 0.01/|V|$ where several content words, such as *admit*, *fine*, and *great* are possibly helpful for predicting the correct label. By contrast, 421 and 84 tokens in CosmosQA are found to be statistically significant at the respective levels, where many content words, such as *enjoy* and *life*, function words, such as *or*, and the task-specific phrase (*none of the above choice*) are strong indicators. We observe similar trends for RACE and QuAIL. In Appendix F, we report the numbers of vocabulary items above the

Figure 3: Question words and the subsequent words of the questions in our dataset.

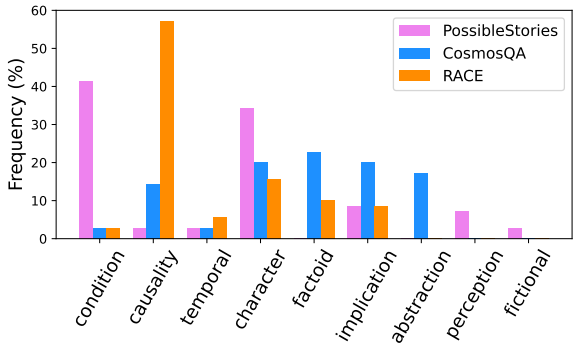

Figure 4: Reasoning types across our dataset, CosmosQA, and RACE.

levels of statistical significance for the four datasets. These results show that the answer options in our dataset contain minimal annotation artifacts compared to those of the other datasets.

6.3 Question and Reasoning Types

The question words and subsequent words in the test questions are plotted in Figure 3. We find that more than half the number of questions are *what* questions, seemingly asking about the concrete content of the story. We also observe subsequent words, such as *would*, *outcome*, and *if*, which lead to a statement requiring commonsense reasoning.

To investigate the kind of reasoning required for answering, we annotate the collected questions with reasoning types. Considering previous studies, we define nine reasoning types (See Appendices G and H for the definitions and examples). We annotate 70 questions from our test set and the same number of questions taken from CosmosQA and

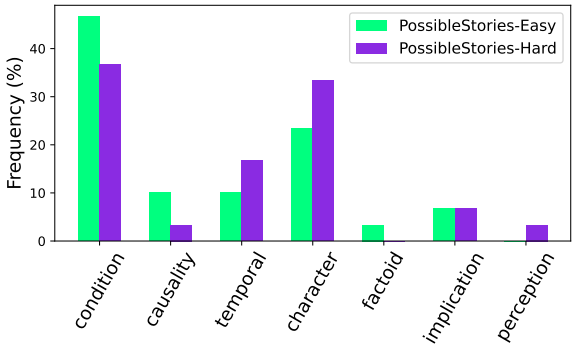

Figure 5: Reasoning types of easy and hard questions in our dataset.

RACE for comparison (Figure 4). In addition, to examine the relationship between question difficulty and reasoning types, we split the test examples into easy and hard subsets and annotate 30 questions for each subset (Figure 5). The easy questions are those for which the human–model accuracy gap is 0% in terms of accuracy, and the hard ones are those for which the gap is larger than 60% (215 and 64 examples). To compute model performance, we average the accuracy of the five models (BERT-base and large, RoBERTa-base and large, and RoBERTa-large, which is fine-tuned on RACE; all models are fine-tuned on our training set). Apart from the reasoning types, we independently check whether each question requires counterfactual reasoning. This includes not only the condition and fiction types but also other types such as temporal reasoning in the sense that it can be involved in reasoning over counterfactual conditions. The frequency (%) of such questions is as follows: ours 68.6, ours-hard 76.7, ours-easy 66.7, CosmosQA 4.3, and RACE 2.9.

In summary, our major findings are as follows:

- Our dataset includes more questions regarding conditions and characters' motivations and reactions than the other datasets. It also has a small number of fictional and perception questions, while the others do not.
- We do not observe abstraction and factoid questions in the annotated examples. However, we find several abstraction questions in our test set, one of which is presented in Appendix H.
- Questions regarding characters' motivations and reactions are relatively harder, while questions regarding causality, which do not require counterfactual reasoning in most cases in our annota-

P: The Smith family loved to go on day trips on their boat in the summer. One day, they decided it would be fun to take the kids to a new place. They chose to travel north to a beach that wasn't terribly far away. The children had a wonderful time and met a new friend to play with.

Q1: Which of these is the most negative ending? **Q2:** Which of these implies that the trip they took was successful?
Q3: Which ending implies the Smith kids were bad at staying in touch? **Q4:** Which ending involves the most conflict?

Q1	Q2	Q3	Q4	Options
☐	✗	☐	☐	**A:** They kept in touch with their friend even after they went home.
✓	☐	☐	✓	**B:** At the end of the day the kids got into a fight with each other and were happy to leave.
☐	✓	☐	☐	**C:** The Smith's decided they'd visit a new beach every year, and they made tons of new friends.
✗	☐	✓	✗	**D:** They went home though and the kids never saw their friend again.

Figure 6: Example of questions with a single passage. Check mark (✓) indicates the correct option. Cross mark (✗) indicates that DeBERTa-large (v3) makes an incorrect prediction with that option.

tion, are easier. This corresponds with the fact that we find more counterfactual questions in the hard questions than in the easy questions.

6.4 Case Study

We present examples in which the strongest model (DeBERTa-large) makes incorrect predictions (Figure 6). A single worker writes Q1 and Q2 and another worker writes Q3 and Q4. Q1 and Q4 are annotated as perceptions (*the most negative ending* and *the most conflict*). It seems that the model struggles to compare which option is more negative between options B (*got into a fight...were happy*) and D (*never saw their friend*). Q2 and Q3 are annotated as implications (*imply...*). For Q2, we must infer that option C (e.g., *made tons of new friends*) implies success, but option B (*kept in touch with their friend*) might sound more successful to the model. More examples of other reasoning types are provided in Appendix H.

7 Discussion

Circumscribing commonsense reasoning from simple heuristics has been a long-standing problem in the field of artificial intelligence (Levesque, 2014). Although the answer options in our dataset are free from annotation artifacts, our ablation analysis in Section 5 also shows that the questions and answer options may still involve some clues that the models can exploit. Further research is needed to explain how commonsense reasoning is distinguished from a set of simple heuristics in machines' situational understanding.

In the question-writing task, one of the workers addresses the difficulty of creating questions that cannot be answered without reading the passages and that it is even practically impossible unless asking the questions to small children. This illustrates that humans may also use a small amount of information available to draw inferences. This issue arises possibly because of our task formulation, answering which option is more plausible than the others. It can be argued that modifying this task to a generative task (Chen et al., 2020a) is one way to directly assess machines' commonsense reasoning ability, but it should be noted that this would entail some difficulties in the evaluation of generated answers.

In addition, exploring what kind of conditions narrow down the possibilities of consequences is important to effectively evaluate machines' situational understanding. Although several studies have captured the dynamics of conditions in moral and immoral settings (Emelin et al., 2021), many other factors come into play in decision-making in our daily lives, such as feelings, personal beliefs, expectations from others, or even unconscious biases.

8 Conclusion

This paper proposes a new dataset, Possible Stories, consisting of 4.5K crowdsourced questions with 1.3K story passages to investigate whether machines can infer the most plausible ending among four possible endings under certain situations postulated by questions. We discover that current strong pretrained language models struggle to answer questions consistently, showing a large accuracy gap compared with humans. A comparison with existing multiple-choice datasets demonstrates that our questions contain minimal annotation artifacts in the answer options and require counterfactual reasoning as well as an understanding of characters' motivations and emotions, suggesting that our dataset can serve as a challenging benchmark for future commonsense reasoning studies.

Ethical Considerations

This study aims to facilitate the scientific study of machines' situated commonsense reasoning. We use crowdsourcing for our data collection, taking care to avoid the exploitation of workers and pay well above the U.S. federal minimum wage. The details of worker recruitment and the payment process are described in Appendix B. We also validate that the examples in our dataset do not contain unfair or harmful content. In this section, we report our observations regarding the validation task. This study is approved by our internal review board.

Content Validation for Fair Representation During content validation (Section 4.2), we find that the level of content to be flagged is not trivial. There is a question containing the phrase *mainstream COVID-related propaganda*, and one of the workers told us that the worker was unsure if it should be flagged. Another case involves a story ending that describes the cooking skills of a male character in a bad light. Does this representation perpetuate the negative stereotype that men are bad at cooking? To investigate this, we should dive deeper into the semantic plausibility learned in language models (Porada et al., 2021; Pedinotti et al., 2021). Unless the focus is on the domain of natural science, there is less agreement on what would lean in spreading desirable and undesirable content, and the borderline can change across time and place. It should also be noted that the degree of sensitivity towards underspecified biases depends on individuals' imagination and empathy. Future work can examine how to effectively moderate the dataset for fair and unbiased representation.

Limitations One of the limitations of the study is "limited diversity." We observe that some systemic biases during data collection. One example concerns a story in which the protagonist missed breakfast on a day of work. Many crowdworkers come up with the possibility of the girlfriend bringing the lunch to the protagonist's workplace (referred to as *I* throughout the context), but no one assumes that the boyfriend will do the same. These types of unconscious biases can accumulate in datasets. In addition, our dataset is limited to English.

Acknowledgments

The authors would like to thank the anonymous reviewers for their insightful comments. This work was supported by JST PRESTO Grant Number JPMJPR20C4 and JSPS KAKENHI Grant Number 22K17954.

References

Shourya Aggarwal, Divyanshu Mandowara, Vishwajeet Agrawal, Dinesh Khandelwal, Parag Singla, and Dinesh Garg. 2021. Explanations for CommonsenseQA: New Dataset and Models. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 3050–3065, Online. Association for Computational Linguistics.

Max Bartolo, Alastair Roberts, Johannes Welbl, Sebastian Riedel, and Pontus Stenetorp. 2020. Beat the AI: Investigating adversarial human annotation for reading comprehension. *Transactions of the Association for Computational Linguistics*, 8:662–678.

Chandra Bhagavatula, Ronan Le Bras, Chaitanya Malaviya, Keisuke Sakaguchi, Ari Holtzman, Hannah Rashkin, Doug Downey, Wen tau Yih, and Yejin Choi. 2020. Abductive commonsense reasoning. In *International Conference on Learning Representations*.

Sid Black, Leo Gao, Phil Wang, Connor Leahy, and Stella Biderman. 2021. GPT-Neo: Large scale autoregressive language modeling with mesh-tensorflow.

Carlo E. Bonferroni. 1936. Teoria statistica delle classi e calcolo delle probabilita. *Pubblicazioni del R. Istituto Superiore di Scienze Economiche e Commerciali di Firenze*, 8:3–62.

Samuel R. Bowman and George Dahl. 2021. What will it take to fix benchmarking in natural language understanding? In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 4843–4855, Online. Association for Computational Linguistics.

Anthony Chen, Gabriel Stanovsky, Sameer Singh, and Matt Gardner. 2020a. MOCHA: A dataset for training and evaluating generative reading comprehension metrics. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 6521–6532, Online. Association for Computational Linguistics.

Tongfei Chen, Zhengping Jiang, Adam Poliak, Keisuke Sakaguchi, and Benjamin Van Durme. 2020b. Uncertain natural language inference. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 8772–8779, Online. Association for Computational Linguistics.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of

deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, Minneapolis, Minnesota. Association for Computational Linguistics.

Denis Emelin, Ronan Le Bras, Jena D. Hwang, Maxwell Forbes, and Yejin Choi. 2021. Moral stories: Situated reasoning about norms, intents, actions, and their consequences. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 698–718, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Maxwell Forbes, Jena D. Hwang, Vered Shwartz, Maarten Sap, and Yejin Choi. 2020. Social chemistry 101: Learning to reason about social and moral norms. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 653–670, Online. Association for Computational Linguistics.

Matt Gardner, Yoav Artzi, Victoria Basmov, Jonathan Berant, Ben Bogin, Sihao Chen, Pradeep Dasigi, Dheeru Dua, Yanai Elazar, Ananth Gottumukkala, Nitish Gupta, Hannaneh Hajishirzi, Gabriel Ilharco, Daniel Khashabi, Kevin Lin, Jiangming Liu, Nelson F. Liu, Phoebe Mulcaire, Qiang Ning, Sameer Singh, Noah A. Smith, Sanjay Subramanian, Reut Tsarfaty, Eric Wallace, Ally Zhang, and Ben Zhou. 2020. Evaluating models' local decision boundaries via contrast sets. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 1307–1323, Online. Association for Computational Linguistics.

Matt Gardner, William Merrill, Jesse Dodge, Matthew Peters, Alexis Ross, Sameer Singh, and Noah A. Smith. 2021. Competency problems: On finding and removing artifacts in language data. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 1801–1813, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Suchin Gururangan, Swabha Swayamdipta, Omer Levy, Roy Schwartz, Samuel Bowman, and Noah A. Smith. 2018. Annotation artifacts in natural language inference data. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 2 (Short Papers)*, pages 107–112, New Orleans, Louisiana. Association for Computational Linguistics.

Pengcheng He, Jianfeng Gao, and Weizhu Chen. 2021. DeBERTaV3: Improving DeBERTa using ELECTRA-style pre-training with gradient-disentangled embedding sharing. arXiv preprint 2111.09543.

Lifu Huang, Ronan Le Bras, Chandra Bhagavatula, and Yejin Choi. 2019. Cosmos QA: Machine reading comprehension with contextual commonsense reasoning. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 2391–2401, Hong Kong, China. Association for Computational Linguistics.

Robin Jia and Percy Liang. 2017. Adversarial examples for evaluating reading comprehension systems. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 2021–2031, Copenhagen, Denmark. Association for Computational Linguistics.

Divyansh Kaushik, Douwe Kiela, Zachary C. Lipton, and Wen-tau Yih. 2021. On the efficacy of adversarial data collection for question answering: Results from a large-scale randomized study. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 6618–6633, Online. Association for Computational Linguistics.

Daniel Khashabi, Sewon Min, Tushar Khot, Ashish Sabharwal, Oyvind Tafjord, Peter Clark, and Hannaneh Hajishirzi. 2020. UNIFIEDQA: Crossing format boundaries with a single QA system. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 1896–1907, Online. Association for Computational Linguistics.

Jonathan K. Kummerfeld. 2021. Quantifying and avoiding unfair qualification labour in crowdsourcing. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 2: Short Papers)*, pages 343–349, Online. Association for Computational Linguistics.

Guokun Lai, Qizhe Xie, Hanxiao Liu, Yiming Yang, and Eduard Hovy. 2017. RACE: Large-scale ReAding comprehension dataset from examinations. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 785–794, Copenhagen, Denmark. Association for Computational Linguistics.

Hector J. Levesque. 2014. On our best behaviour. *Artificial Intelligence*, 212:27–35.

Tao Li, Daniel Khashabi, Tushar Khot, Ashish Sabharwal, and Vivek Srikumar. 2020. UNQOVERing stereotyping biases via underspecified questions. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 3475–3489, Online. Association for Computational Linguistics.

Nelson F. Liu, Roy Schwartz, and Noah A. Smith. 2019a. Inoculation by fine-tuning: A method for analyzing challenge datasets. In *Proceedings of the 2019 Conference of the North American Chapter of the*

Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers), pages 2171–2179, Minneapolis, Minnesota. Association for Computational Linguistics.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019b. RoBERTa: A robustly optimized BERT pretraining approach. arXiv preprint 1907.11692.

Nicholas Lourie, Ronan Le Bras, Chandra Bhagavatula, and Yejin Choi. 2021. UNICORN on RAINBOW: A universal commonsense reasoning model on a new multitask benchmark. *Proceedings of the AAAI Conference on Artificial Intelligence*, 35(15):13480–13488.

Rebecca Marvin and Tal Linzen. 2018. Targeted syntactic evaluation of language models. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 1192–1202, Brussels, Belgium. Association for Computational Linguistics.

Johannes Mario Meissner, Napat Thumwanit, Saku Sugawara, and Akiko Aizawa. 2021. Embracing ambiguity: Shifting the training target of NLI models. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 2: Short Papers)*, pages 862–869, Online. Association for Computational Linguistics.

Nasrin Mostafazadeh, Nathanael Chambers, Xiaodong He, Devi Parikh, Dhruv Batra, Lucy Vanderwende, Pushmeet Kohli, and James Allen. 2016. A corpus and cloze evaluation for deeper understanding of commonsense stories. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 839–849, San Diego, California. Association for Computational Linguistics.

Nasrin Mostafazadeh, Aditya Kalyanpur, Lori Moon, David Buchanan, Lauren Berkowitz, Or Biran, and Jennifer Chu-Carroll. 2020. GLUCOSE: GeneraLized and COntextualized story explanations. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 4569–4586, Online. Association for Computational Linguistics.

Nikita Nangia, Saku Sugawara, Harsh Trivedi, Alex Warstadt, Clara Vania, and Samuel R. Bowman. 2021. What ingredients make for an effective crowdsourcing protocol for difficult NLU data collection tasks? In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 1221–1235, Online. Association for Computational Linguistics.

Ellie Pavlick and Tom Kwiatkowski. 2019. Inherent disagreements in human textual inferences. *Transactions of the Association for Computational Linguistics*, 7:677–694.

Paolo Pedinotti, Giulia Rambelli, Emmanuele Chersoni, Enrico Santus, Alessandro Lenci, and Philippe Blache. 2021. Did the cat drink the coffee? challenging transformers with generalized event knowledge. In *Proceedings of *SEM 2021: The Tenth Joint Conference on Lexical and Computational Semantics*, pages 1–11, Online. Association for Computational Linguistics.

Ian Porada, Kaheer Suleman, Adam Trischler, and Jackie Chi Kit Cheung. 2021. Modeling event plausibility with consistent conceptual abstraction. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 1732–1743, Online. Association for Computational Linguistics.

Lianhui Qin, Antoine Bosselut, Ari Holtzman, Chandra Bhagavatula, Elizabeth Clark, and Yejin Choi. 2019. Counterfactual story reasoning and generation. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 5043–5053, Hong Kong, China. Association for Computational Linguistics.

Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, Ilya Sutskever, et al. 2019. Language models are unsupervised multitask learners. *OpenAI blog*, 1(8):9.

Hannah Rashkin, Antoine Bosselut, Maarten Sap, Kevin Knight, and Yejin Choi. 2018. Modeling naive psychology of characters in simple commonsense stories. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 2289–2299, Melbourne, Australia. Association for Computational Linguistics.

Nils Reimers and Iryna Gurevych. 2019. Sentence-BERT: Sentence embeddings using Siamese BERT-networks. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 3982–3992, Hong Kong, China. Association for Computational Linguistics.

Anna Rogers, Olga Kovaleva, Matthew Downey, and Anna Rumshisky. 2020. Getting closer to AI complete question answering: A set of prerequisite real tasks. *Proceedings of the AAAI Conference on Artificial Intelligence*, 34(05):8722–8731.

Swarnadeep Saha, Prateek Yadav, Lisa Bauer, and Mohit Bansal. 2021. ExplaGraphs: An explanation graph generation task for structured commonsense reasoning. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*,

pages 7716–7740, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Keisuke Sakaguchi, Ronan Le Bras, Chandra Bhagavatula, and Yejin Choi. 2020. WinoGrande: An adversarial Winograd schema challenge at scale. *Proceedings of the AAAI Conference on Artificial Intelligence*, 34(05):8732–8740.

Maarten Sap, Hannah Rashkin, Derek Chen, Ronan Le Bras, and Yejin Choi. 2019. Social IQa: Commonsense reasoning about social interactions. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 4463–4473, Hong Kong, China. Association for Computational Linguistics.

Saku Sugawara, Nikita Nangia, Alex Warstadt, and Samuel Bowman. 2022. What makes reading comprehension questions difficult? In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 6951–6971, Dublin, Ireland. Association for Computational Linguistics.

Alon Talmor, Jonathan Herzig, Nicholas Lourie, and Jonathan Berant. 2019. CommonsenseQA: A question answering challenge targeting commonsense knowledge. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4149–4158, Minneapolis, Minnesota. Association for Computational Linguistics.

Niket Tandon, Bhavana Dalvi, Keisuke Sakaguchi, Peter Clark, and Antoine Bosselut. 2019. WIQA: A dataset for "what if..." reasoning over procedural text. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 6076–6085, Hong Kong, China. Association for Computational Linguistics.

Alex Warstadt, Alicia Parrish, Haokun Liu, Anhad Mohananey, Wei Peng, Sheng-Fu Wang, and Samuel R. Bowman. 2020. BLiMP: The benchmark of linguistic minimal pairs for English. *Transactions of the Association for Computational Linguistics*, 8:377–392.

Adina Williams, Nikita Nangia, and Samuel Bowman. 2018. A broad-coverage challenge corpus for sentence understanding through inference. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 1112–1122, New Orleans, Louisiana. Association for Computational Linguistics.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Remi Louf, Morgan Funtowicz, Joe Davison, Sam Shleifer, Patrick von Platen, Clara Ma, Yacine Jernite, Julien Plu, Canwen Xu, Teven Le Scao, Sylvain Gugger, Mariama Drame, Quentin Lhoest, and Alexander Rush. 2020. Transformers: State-of-the-art natural language processing. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 38–45, Online. Association for Computational Linguistics.

Rowan Zellers, Ari Holtzman, Yonatan Bisk, Ali Farhadi, and Yejin Choi. 2019. HellaSwag: Can a machine really finish your sentence? In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 4791–4800, Florence, Italy. Association for Computational Linguistics.

Michael Zhang and Eunsol Choi. 2021. SituatedQA: Incorporating extra-linguistic contexts into QA. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 7371–7387, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Sheng Zhang, Rachel Rudinger, Kevin Duh, and Benjamin Van Durme. 2017. Ordinal common-sense inference. *Transactions of the Association for Computational Linguistics*, 5:379–395.

A Selecting Stories from ROCStories

We choose stories with more than 45 words and endings with more than 5 words to avoid stories that are too short or generic. We consult the annotation of the GLUCOSE dataset (Mostafazadeh et al., 2020) and select stories whose ending is annotated by workers with ratings higher than 1. Future work could investigate the effect of different causal relations on creating conditions, as well as possible endings.

B Crowdworker Recruitment and Payment

We recruit writers via Amazon Mechanical Turk (MTurk). The number of workers who participated in the study is listed in Table 7. Before initiating the data collection procedure, we first run a qualification task to identify workers who can participate in data collection. This task is a short version of a part of the main task and is open to *any* crowdworker without *any* qualifications such as HIT acceptance rate or number of HIT accepted, which are commonly used thresholds in the NLP community's data collection process via MTurk.

	Ending	Ques.	Valid.
# of crowdworkers	163	66	65
Ave. # of examples	54.6	79.9	243.3
Max. # of examples	132	170	400

Table 7: Statistics of crowdworkers that participate in each task, the average and maximum number of generated examples per crowdworker.

We adapt this qualification following the recommendation of Kummerfeld (2021) to avoid the exploitation of crowdworkers. He demonstrates that imposing these prepared criteria is not fair because crowdworkers need to work on poorly paid tasks to achieve those qualifications in most cases.

We pay $1.0 USD for an ending writing task, $1.5 for a question writing task, and $1.0 for a validation task, estimating the completion time to be less than 5, 7.5, and 5 mins respectively. This adds to more than $12, which is well above the U.S. federal minimum wage. We do not calculate the wage according to the cost of living in each country where the workers reside, as we do not ask them where they live.

C Validation Results and Quality Control

Validation Results During the question-answer validation, 13.8% of the collected questions are discarded. Out of the four additional options, questions with no answer account for 1.8%, those with more than two possible answers account for 6.8%, ill-formed questions account for 1.8%, and others account for 2.1% of the total. The high frequency of questions with more than two options is understood to be due to the possibility that some answer options are too similar to each other to create questions with a single correct answer. Through the content validation process, 0.2% of the questions are discarded.

Quality Control During the data collection process, we repeat all tasks three times (i.e., three batches). The first and second batches have no workers in common, resulting in 52% of the final dataset with a total of 66 workers. For the final batch, we further qualify the workers who participated in these batches using three criteria: 1) writing more than nine questions, 2) mean human validation accuracy of more than 66%, and 3) creating more than 90% of questions as*wh*-questions to ensure dataset quality. Additionally, we manually check the comments given to each worker and exclude workers who tend to produce yes/no questions and those containing unethical or politically sensitive topics. The final batch yields 48% of the final dataset with 38 workers.

Model	RACE	CosmosQA
DeBERTa-large†	92.1	89.7
DeBERTa-large	88.5	51.3
RoBERTa-large†	83.5	83.3
RoBERTa-large	50.5	38.3

Table 8: Accuracy (%) of models on our test set that are fine-tuned on RACE and CosmosQA respectively. † indicates that the model is trained on our training set (i.e., supervised).

Model	b	lr
DeBERTa-large	24	1e-5
DeBERTa-base	48	3e-5
RoBERTa-large	24	1e-5
RoBERTa-base	48	3e-5
BERT-large	36	1e-5
BERT-base	72	3e-5

Table 9: Hyperparameters used in the experiments. b and lr indicate the batch size and learning rate, respectively.

D Comparison of Models Fine-tuned on RACE and CosmosQA

In our experiments, we use RACE for fine-tuning our pretrained language models to adapt them to the multiple-choice task. This is because we observe that RoBERTa-large and DeBERTa-large fine-tuned on RACE show higher performance than the corresponding models fine-tuned on CosmosQA (Table 8) in both unsupervised and supervised settings.

E Details of Experiments

Table 9 reports the hyperparameters used in our experiments. We use Huggingface's Transformers library (Wolf et al., 2020) for our experiments.

Table 10 reports the detailed results of DeBERTa-large (fine-tuned on RACE) on our test set in the unsupervised and supervised settings. Owing to computational constraints, we conduct five different runs only for this model, which is the strongest among the models we use in our experiments. We

FT	Model	Full input		No passage		No question	
		Accuracy	Consist.	Accuracy	Consist.	Accuracy	Consist.
✗	DeBERTa-large*	60.2±1.7	19.9±2.2	58.1±2.6	19.9±1.7	21.8±1.6	0.5±0.4
✔	DeBERTa-large*	92.1±0.6	74.7±2.3	87.0±0.7	62.1±1.8	31.8±1.6	1.9±0.7

Table 10: Unsupervised and supervised performance (%) with the standard deviations of DeBERTa-large in five runs. The five models are fine-tuned on RACE with different random seeds, respectively.

α	Ours	Cosmos	RACE	QuAIL
0.01	12/5	421/33	475/163	173/7
$0.01/\|V\|$	0/0	84/6	104/19	39/3
$\|V\|$	3,990	15,472	35,762	9,688

Table 11: The number of vocabulary items that appear in correct/incorrect options above the levels of statistical significance ($\alpha = 0.01$ and its conservative Bonferroni correction for the size of vocabulary $|V|$).

do not observe large deviations across the runs.

F Annotation Artifacts in Answer Options

We report the number of examples above different levels of statistical significance across the four analyzed datasets in Table 11. The number for our dataset above $\alpha = 0.01/|V|$ is zero, whereas those for the other datasets are significantly larger. This result shows that our dataset does not suffer from token-level annotation artifacts in the answer options, supporting our findings on the option-only training results in Section 6.1.

G Definitions of Reasoning Types

Annotating reasoning types is not a trivial task, particularly because the questions are fully written by humans without templates. Moreover, it is possible to use many classification methods, and there is rarely a consensus on reasoning types. For example, CosmosQA proposes seven reasoning types: pre-/post-conditions, motivations, reactions, temporal events, situational facts, counterfactuals, and other (e.g., cultural norms). In QuAIL, nine types of reasoning are proposed spanning three categories: temporal, factoid, character properties for *text-based questions*, coreference, causality, belief states, subsequent entity states, event durations for *questions that require world knowledge*, and *unanswerable*. After categorizing these into five types, we add three types: abstraction (summarizing what happened), implication (paraphrasing), and readers' (observers') perceptions. In addition, we differentiate reasoning over fiction from counterfactual in that it is a more specific type of counterfactual that is considered implausible for most people in the real world. This results in the nine reasoning types listed in Table 12. Appendix H presents some examples.

1. **Condition**: pre/post counterfactual conditions introduced in the question.
2. **Causality**: causes and effects of events.
3. **Temporal**: temporal relations between events.
4. **Character**: characters' emotions, motivations, and reactions.
5. **Factoid**: extracting entities from the context.
6. **Abstraction**: lesson, conclusion, and summary of the context.
7. **Implication**: paraphrasing and implication about events.
8. **Perception**: reader's perceptual responses.
9. **Fictional**: fictional situations as counterfactual condition.

Table 12: Definitions of reasoning types.

H More Examples for Reasoning Types and Difficulty

The reasoning types and their example questions taken from our dataset are listed in Table 13. We also show examples of passage, question, and answer options in our dataset, including easy and hard questions, in Figure 7. Each question ends with its reasoning type and easy/hard classification, if available.

I Annotation Instructions and Interfaces

I.1 Ending Writing

Figures 8 and 9 show the instructions used in the story ending writing task. Figure 10 shows the interface used in the story ending writing task.

Reasoning	Example
Condition	Jeff is a child with a very vivid sense of imagination. What is most likely to have happened next?
Causality	Which is the most likely caused the guests to avoid shards of glass?
Temporal	Which is most likely if Chris later felt sick to his stomach?
Character	What outcome would be most upsetting to Ben?
Factoid	Where did people hide the money they got?
Abstraction	What lesson did she learn from the passage?
Implication	Which answer implies Bob was pleased with his performance?
Perception	What is the most moral decision for Danielle?
Fictional	How does Dylan get home?

Table 13: Reasoning types we use in the annotation and their example questions.

I.2 Question Writing

Figures 11, 12, and 13 show the instructions used in the question writing task. Figures 14 and 15 show the interface used in the question writing task.

I.3 Question Validation

Figure 16 shows the instructions used in the question validation task.

P1: Lydia was listening to an old CD her boyfriend had burned for her. Her CD player was old but still working alright. She had lost track of her thoughts and was enjoying the music. Suddenly, the CD skipped out and stopped playing.

Q1: Why was the CD player unable to function? (causality, easy)
Q2: Which answer indicates that Lydia would never be able to listen to the CD again? (implication, hard)
Q3: Which of the following is likely to occur if we know Lydia has realized the CD player cannot be fixed? (condition, hard)

Q1	Q2	Q3	Options
☐	☑	☐	**A:** Lydia tried to fix it but the CD had a huge scratch.
☐	☐	☐	**B:** Lydia tinkered with the CD player and got it working again.
☐	☐	☑	**C:** Lydia went to bed upset, knowing she had to buy a new one in the morning.
☑	☒	☒	**D:** She realized the batteries in her CD player had died.

P2: Darrel was waiting in the drive through for half an hour. He had about lost his patience. When he finally got to the window he was about to scream at them. They immediately apologized before he could.

Q1: How did the employees react when they saw Darrel's face turn red at the drive-through window? (character)
Q2: How did Darrel respond after the employees apologized for the long wait? (character, easy)
Q3: If Darrel's mind was soon preoccupied with something entirely different, what was most likely to have happened? (condition, easy)
Q4: In this scenario, what most likely happened if Darrel was pleased soon thereafter? (character, hard)

Q1	Q2	Q3	Q4	Options
☐	☐	☐	☑	**A:** They had an accident and offered free food to make it up to him.
☐	☑	☐	☐	**B:** He chose not to accept the apology and asked to speak to the manager.
☑	☐	☐	☒	**C:** They quickly gave him his food and informed him that there were very few employees working that day.
☐	☐	☑	☐	**D:** Before he could open his mouth, his engine started smoking and he had to call a tow truck.

P3: Jan checked to make sure no one was around. Her two older brothers had been sneaking around the garden lately. Being a curious child, Jan wanted to know what they were up to. She carefully opened the door to her brother's room.

Q1: If Jan smelled pleasant aromas and felt fresh air in the room, what did she likely discover? (condition)
Q2: What was the likely outcome if Jan was left still feeling clueless about what her brothers had been up to? (character)
Q3: Which outcome is the most unlikely to occur in reality? (fiction)
Q4: Which would be particularly unpleasant for Jan if she suffers from acute arachnophobia? (character)

Q1	Q2	Q3	Q4	Options
☐	☐	☐	☑	**A:** Inside the back of their closet, she found several jars with spiders.
☐	☐	☑	☐	**B:** There was a strange looking alien peeking out of a corner with fearful eyes.
☑	☒	☐	☐	**C:** They had taken plants from the garden and moved them to their room.
☐	☑	☐	☐	**D:** The door slammed shut on her face as the cameras alerted the brothers of an intruder.

P4: Billy liked Christmas songs. But didn't know what a turtle dove was. He like turtle and knew they were green and had a shell. He also knew what a dove was, a type of bird.

Q1: What happened if it was the worst Christmas of Billy's life? (condition, easy)
Q2: What happened if he pictured a turtle with wings? (fictional, easy)
Q3: What outcome would be most tragic? (perception)

Q1	Q2	Q3	Options
☐	☐	☐	**A:** So he decided that 12 drummers drumming was a better part of the song.
☐	☑	☐	**B:** He decided that a turtle dove was likely a flying turtle.
☐	☐	☐	**C:** Billy became a famous author after embracing his love for holiday traditions.
☑	☐	☑	**D:** He went to ask his mother about turtle doves, but when he found her in the bathtub, she was dead.

Figure 7: Examples in our dataset. Check mark (✔) indicates the correct option. Cross mark (✗) indicates that RoBERTa-large fine-tuned on RACE and our training set makes an incorrect prediction with that option.

Transform a story ending to make it unique.

Thank you for your participation. Please read the following instructions (and examples) carefully.

Instructions

You will be presented a short story.
Your task;

- Create **two** story endings (each in ONE sentence) relying on the one original passage (**Alternative Endings**). The two endings you wrote must be **totally different** from each other.

We recommend spending 5-10 minutes on this HIT.

Guidelines for the Task

The goal here is to modify stories to make them those *you* think are unique, less stereotypical, less usual, but those other people will still be able to enjoy it as a story.
Here are some guidelines:

- You are encouraged to be **as creative as possible**, such as;
  - imagining yourself as a different person and describing how that person would behave in the given condition.
  - introducing extraterrestrial beings, animals that can communicate with humans.
- Consider modifying the stories while paying attention to:
  - Characters' feelings and motivations
  - Causes and consequences of described events
  - Definition, properties, and process explained in a passage
  - Summary and lesson of a passage
- Use words and phrases that don't appear in the passage as many as possible.
- Write an ending **in ONE sentence.**
- Write story endings that are **quite different** from the original story endings.
- Also, the two story endings you wrote must be **quite different** from each other.
- Before submission, make sure you read the whole passage with the ending you write so that you can confirm that the story is possible.

Figure 8: Instructions (1/2) used in the story ending writing task.

Example

Tina made spaghetti for her boyfriend. It took a lot of work, but she was very proud. Her boyfriend ate the whole plate and said it was good. Tina tried it herself, and realized it was disgusting.
Original Ending: She was touched that he pretended it was good to spare her feelings.

Example Response #1

Tina made spaghetti for her boyfriend. It took a lot of work, but she was very proud. Her boyfriend ate the whole plate and said it was good. Tina tried it herself, and realized it was disgusting.
Alternative Ending: She got upset that he hadn't been honest with her.

Example Response #2

Tina made spaghetti for her boyfriend. It took a lot of work, but she was very proud. Her boyfriend ate the whole plate and said it was good. Tina tried it herself, and realized it was disgusting.
Alternative Ending: Soon after, she realized that she was dreaming.

IMPORTANT FINAL NOTE

- **Please AVOID writing sentences that involve making prejudicial assumptions about people based on their identities.**
- **Please AVOID writing extremely violent endings, or using abusive language such as slurs.**
- **Please refer to this FAQ for additional information.**

Figure 9: Instructions (2/2) used in the story ending writing task.

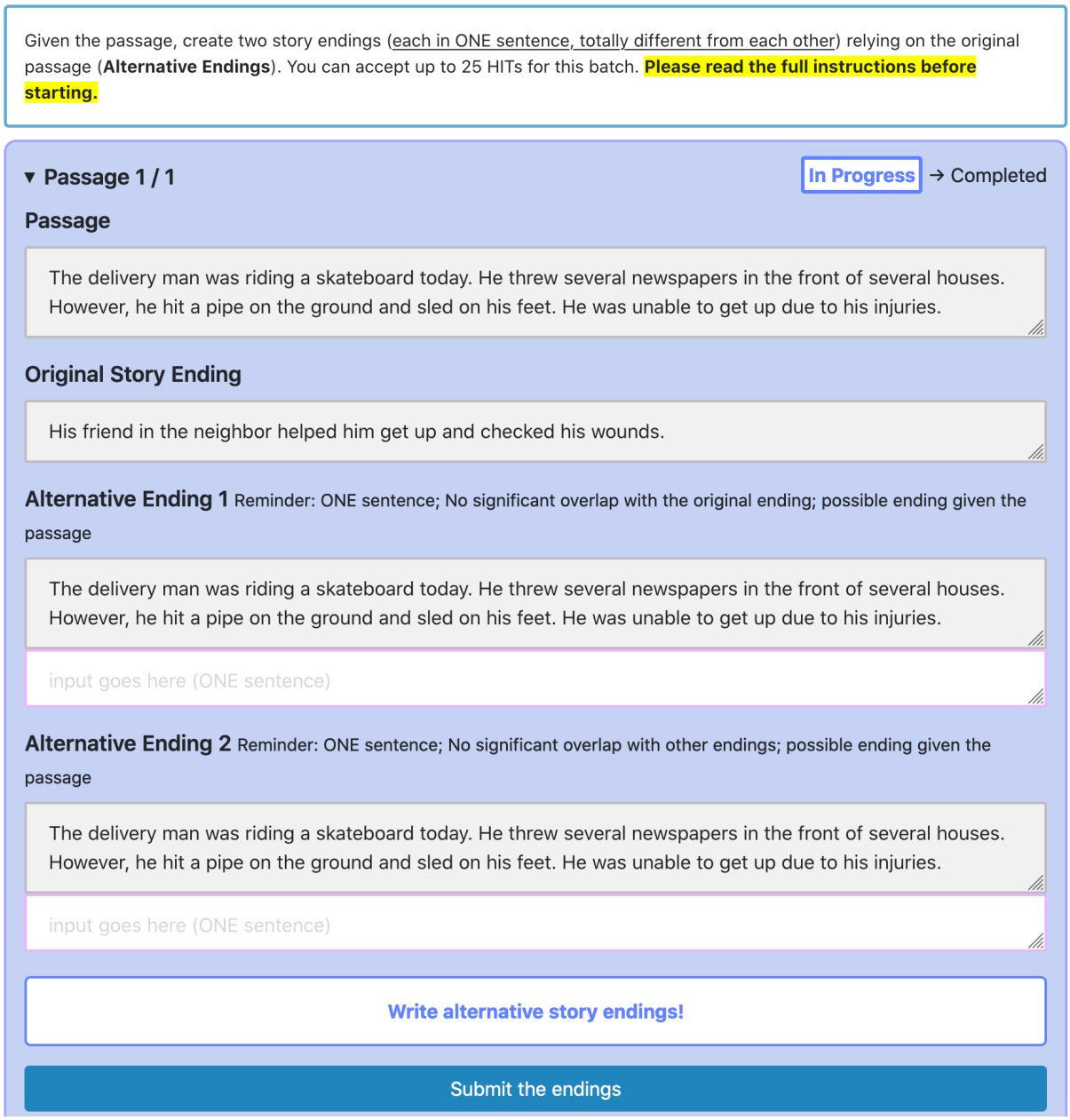

Figure 10: Interface used in the story ending writing task.

Instructions

We are collecting questions to study machine understanding of English. Thank you for your help! We appreciate your amazing jobs so far!

In this task, you will be given a passage and four options that have been collcected as endings before and then selected automatically.
Your task is to **write two challenging questions with a single correct answer for each, given the short passage and the four answer options.** Accordingly, you need to **check the correct answer option for the questions you write** by checking the option, and the correct options to the two questions should be different from each other.

In writing the questions,

- [Important!] You must ensure that the answer to the question is **a single option among the four options presented**, and you **cannot modify the options.**
- The questions you write must be answerable based on the text in the passage and question, combined with some reasonings.
- You should assume that the reader has general background knowledge.
- You should also assume that the reader can speculate about something that is not explicitly stated in the text.
- It should not be easy to eliminate or guess the answer choice/s without reading the passage.
- Likewise, it should also not be easy to eliminate answer choice/s *without reading the question*.
- However, there is no problem in writing two similar questions (with minimal editions) as long as they have different options as their answers!

Guidelines for Writing Questions

Here are some guidelines for writing questions:

- Consider questions that ask about:
  - Characters' feelings and motivations (sad, happy etc consequence for the protagonists)
  - Causes and consequences of described events (both before and after the option occcurs)
  - Definition, properties, and process explained in a passage (additional premises)
  - Summary and lesson of a passage
- Consider the following writing styles:
  - Use words and phrases that don't appear in the passage and the answer options
  - Use abstract words
  - Make construction of the sentence complex with negation or insertion
- Write questions that are answerable based only on the text in the passage and the question.

Figure 11: Instructions (1/3) used in the question writing task.

Examples

In the morning I got out of bed a little late. To make up for lost time I did not make a lunch. By the time I left the house I was only 2 minutes late. The traffic was light so I made it to work on time.

Good Example 1

Question: Which one of the following is most likly to happen after this, if I usually eat something before starting work?

Answer options:

(a) I began waking up late often. (**correct**)
(b) I had enough time to get a cup of coffee in the breakroom before starting work.
(c) When I got to work I realized it was Saturday and I didn't need to be there.
(d) I was even able to grab breakfast on the way to the office.

Explanation: the question has only one right answer. Using "after this" requires reading the passage. (When there are similar options, it happens that the possible question is such that one can assume the right answer without reading the passage. In that case, do your best to use paraphrasing to make it less easy. e.g. drink a cup of coffee --> need caffeine)

Good Example 2

Question: Which describes a habit I acquired from this?

Answer options:

(a) I began waking up late often. (**correct**)
(b) I had enough time to get a cup of coffee in the breakroom before starting work.
(c) When I got to work I realized it was Saturday and I didn't need to be there.
(d) I was even able to grab breakfast on the way to the office.

Explanation: the question has only one right answer. The word "habit" appears nowhere in either passage or answer options.

Other Good Question Examples

- **Which one of the following most accurately expresses the conclusion drawn in the passage?**
- **Which one of the following most likely to happen after this if [insert condition]?**
- **Which one of the following most unlikely to occur after this when [insert condition]?**
- **What would be the most surprising/romantic/ironic ... outcome? (when it has only one ending applicable)**
- **What can we learn from this passage?**
- **What is the main theme of this passage?**

Figure 12: Instructions (2/3) used in the question writing task.

Bad Example 1

Question: Which would be the most plausible story ending when I was not only on time but actually a couple of minutes early?

Answer options:

(a) I began waking up late often.
(b) I had enough time to get a cup of coffee in the breakroom before starting work. (**correct 1**)
(c) When I got to work I realized it was Saturday and I didn't need to be there.
(d) I was even able to grab breakfast on the way to the office. (**correct 2**)

Explanation: This question has multiple correct options.

Bad Example 2

Question: Which is the possible ending of the passage?

Answer options:

(a) I began waking up late often. (**correct**)
(b) I had enough time to get a cup of coffee in the breakroom before starting work. (**correct**)
(c) When I got to work I realized it was Saturday and I didn't need to be there. (**correct**)
(d) I was even able to grab breakfast on the way to the office. (**correct**)

Explanation: all the options are possible as a story ending.

Please refer to this FAQ for additional information.

Figure 13: Instructions (3/3) used in the question writing task.

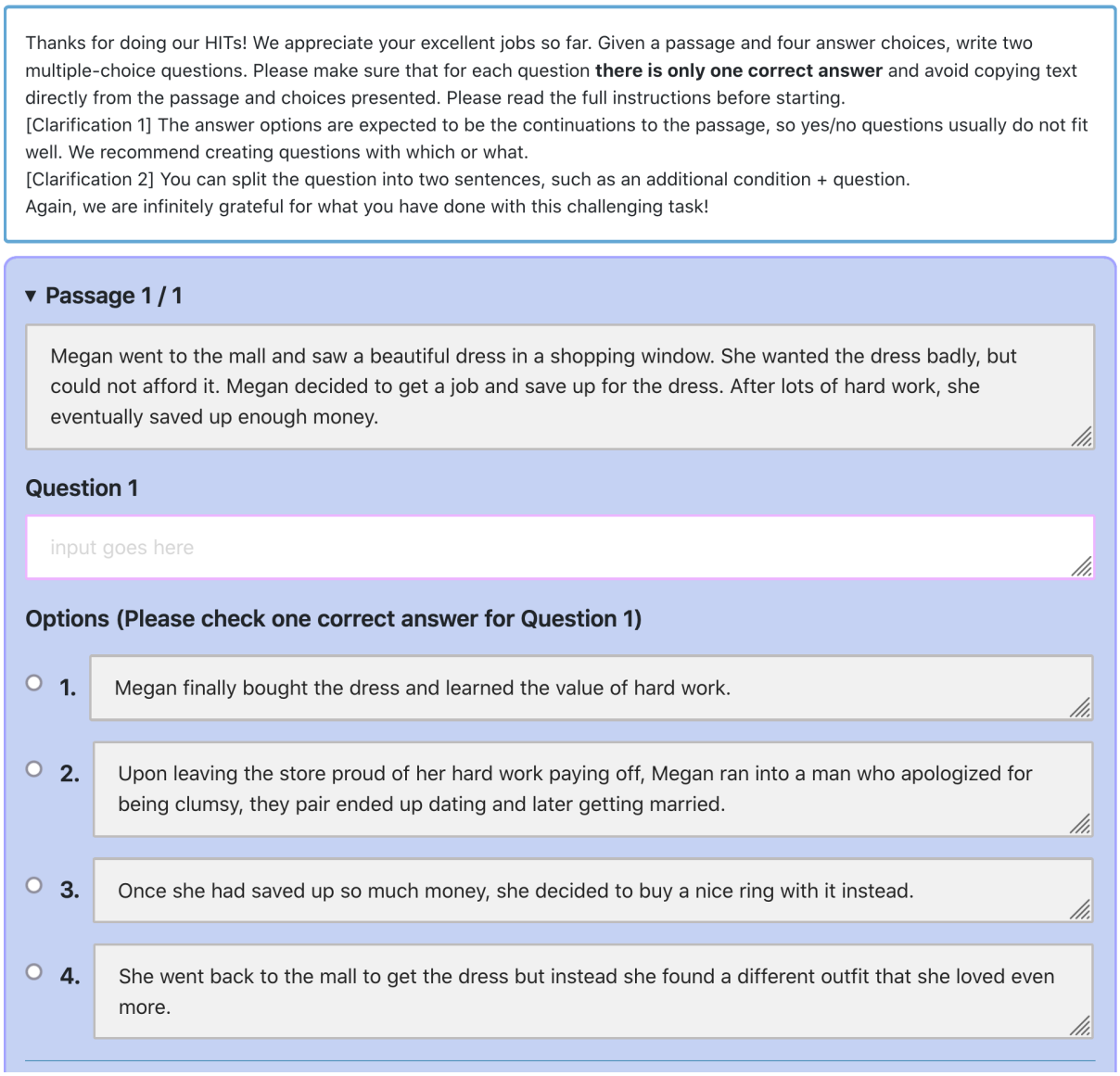

Figure 14: Interface (1/2) used in the question writing task.

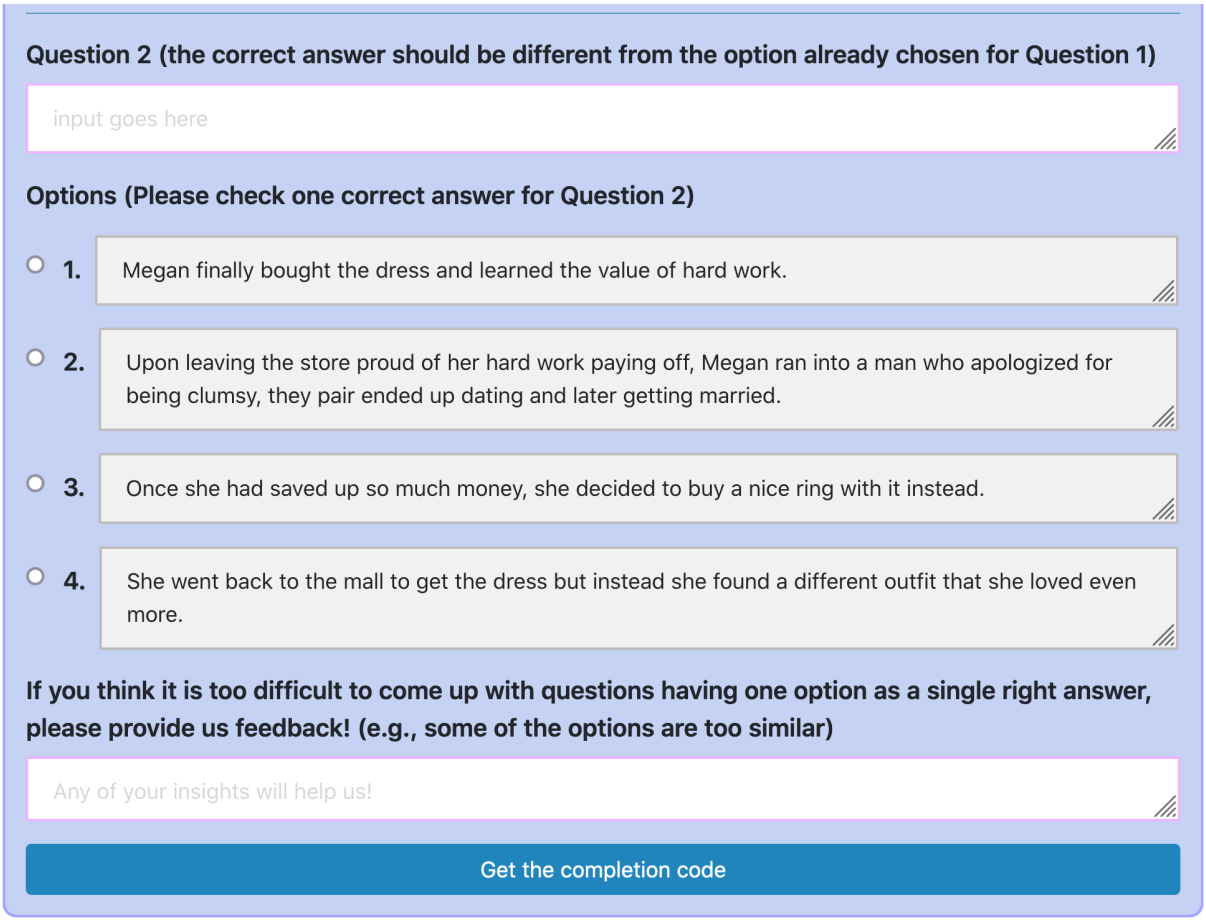

Figure 15: Interface (2/2) used in the question writing task.

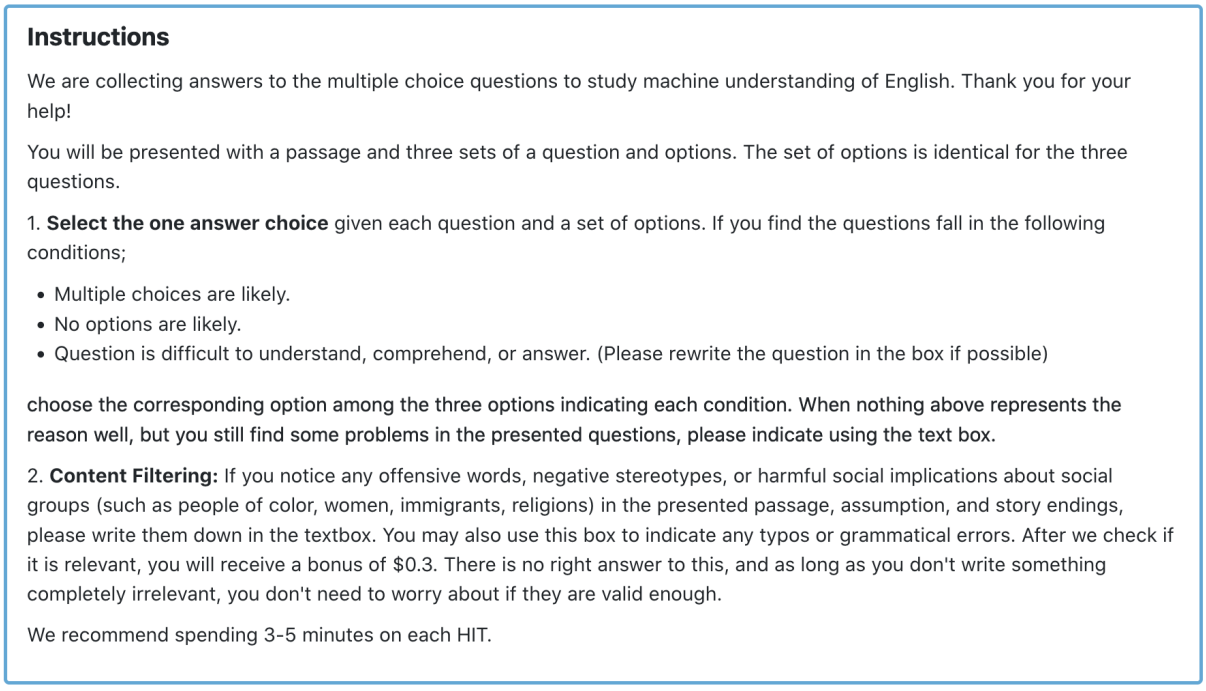

Instructions

We are collecting answers to the multiple choice questions to study machine understanding of English. Thank you for your help!

You will be presented with a passage and three sets of a question and options. The set of options is identical for the three questions.

1. **Select the one answer choice** given each question and a set of options. If you find the questions fall in the following conditions;

- Multiple choices are likely.
- No options are likely.
- Question is difficult to understand, comprehend, or answer. (Please rewrite the question in the box if possible)

choose the corresponding option among the three options indicating each condition. When nothing above represents the reason well, but you still find some problems in the presented questions, please indicate using the text box.

2. **Content Filtering:** If you notice any offensive words, negative stereotypes, or harmful social implications about social groups (such as people of color, women, immigrants, religions) in the presented passage, assumption, and story endings, please write them down in the textbox. You may also use this box to indicate any typos or grammatical errors. After we check if it is relevant, you will receive a bonus of $0.3. There is no right answer to this, and as long as you don't write something completely irrelevant, you don't need to worry about if they are valid enough.

We recommend spending 3-5 minutes on each HIT.

Figure 16: Instructions used in the question validation task.

DiaBiz.Kom – Towards a Polish Dialogue Act Corpus Based on ISO 24617-2 Standard

Marcin Oleksy, Jan Wieczorek, Dorota Drużyłowska, Julia Klyus, Aleksandra Domogała, Krzysztof Hwaszcz, Hanna Kędzierska, Daria Mikoś, Anita Wróż
Wrocław University of Science and Technology, Wrocław, Poland
{marcin.oleksy, jan.wieczorek, dorota.druzylowska}@pwr.edu.pl
{julia.klyus, aleksandra.domogala, krzysztof.hwaszcz}@pwr.edu.pl
{hanna.kedzierska, daria.dominiak, anita.wroz}@pwr.edu.pl

Abstract

This article presents the specification and evaluation of DiaBiz.Kom – the corpus of dialogue texts in Polish. The corpus contains transcriptions of telephone conversations conducted according to a prepared scenario. The transcripts of conversations have been manually annotated with a layer of information concerning communicative functions. DiaBiz.Kom is the first corpus of this type prepared for the Polish language and will be used to develop a system of dialogue analysis and modules for creating advanced chatbots.

1 Introduction

The rationale of the current research was predominantly connected with the lack of corpora including dialogue texts in Polish which could be used to train artificial intelligence for model creation. In order to bridge this gap, we decided to create a corpus that satisfies our expectations i.e. the one that contains dialogue samples from several different fields of business and is annotated for information concerning pragmatic functions. At the first stage of our work, we analysed the approaches adopted by other researchers, whose solutions are described in the "Related Works" section. Further, we describe the process of data collection and its manual annotation. A separate section is dedicated to "Key assumptions and limitations of the guidelines". The paper ends with "Corpus overview" and "Conclusions and future works".

2 Related work

While creating the Polish corpus of dialogue acts, we analyzed some pre-existing corpora. In theoretical perspective we refer to the ISO/DIS 24617-2, standardized annotations and the recommendations from the mentioned document. The ISO standard (Bunt et al., 2010, 2012) is based on particular innovations such as distinguishing between annotations and representations (according to ISO Linguistic Annotation Framework (LAF, ISO 24612:2009) and sets of dialogue participants, dimensions, communicative functions, functional segments and qualifiers (inventory of DiAML). Both manual and automatic annotation of dialogue segments are possible according to the ISO document and both have been tested in practice and described (Keizer et al., 2011; Petukhova et al., 2014; Bunt et al., 2016; Chowdhury et al., 2016; Ngo et al., 2017; Gilmartin et al., 2018). DialogBank is a language resource containing dialogues with gold standard annotations corresponding with the ISO 24617-2 standard (Bunt et al., 2016).

The development of annotation standards for particular corpora can be vividly exemplified by the case of the Switchboard Dialogue Act Corpus (the collection of telephone conversations). Telephone Speech Corpus (LDC97S62) was originally collected by Texas Instruments in 1990-1 and consists of approximately 260 hours of speech. The first release of the corpus was published in 1992-3[1]. Initially, the utterances in the corpus were annotated according to the DAMSL scheme for dialogue act analysis. Subsequently, NXT-format Switchboard Corpus was created with additional annotations according to an international standard ISO 64217-2:2012 (FANG et al., 2012). Conversion of one annotation system to another required matching of tags between them: DAMS consists of 59 combine tags while ISO – of 56 core tags. The re-annotation shows the significance of both standard scheme improvement and combining different standards on the same linguistic material.

Another work addressing the creation of corpora of dialogue acts concerns the DBOX corpus, created within the DBOX project and aimed at developing an interactive Question-Answering dialogue system (Petukhova et al., 2014). A more practical application of the project was to develop an

[1] https://catalog.ldc.upenn.edu/LDC97S62

Proceedings of the 29th International Conference on Computational Linguistics, pages 3631–3638
October 12–17, 2022.

interactive system used in computer games in three European languages. The authors collected 338 dialogues incorporating the continuous data collection method, i.e. they initially used the so-called Wizard-of-Oz paradigm with a human Wizard mirroring the system's behavior, and later replaced the Wizard with a complex dialogue system.

A similar approach was adopted by the authors of The ADELE Corpus of Dyadic Social Text Conversations (Gilmartin et al., 2018) who created a corpus consisting of 193 dialogues resumed with the purpose of initiating interactions with other people. Correspondingly to the DBOX corpus, the ADELE corpus was predominantly constructed with the view of training a spoken dialogue system that could easily engage in a conversation during a role-playing computer game. Both in the case of DBOX and ADELE, the obtained dialogues were manually annotated with dialogue act information in accordance with the ISO 24617-2 dialogue act annotation scheme, which was supplemented with additional dimension (for DBOX) as well as several additional dimension-specific functions and general-purpose functions (for both corpora).

Other related works which are worth mentioning include the Italian Luna Human-Human Corpus, which is a collection of 572 dialogues in the hardware/software helpdesk domain. The dialogues are conversations of the users engaged in problem solving tasks; a subset of 50 dialogues was annotated with the use of dialogues acts.

Furthermore, the DiaBiz.Kom corpus correlates with the DialogBank corpus, which is mentioned as the current golden annotation standard. Most dialogues from the DialogBank corpus were taken from other corpora and re-segmented and re-annotated. All annotations were double-checked for inconsistencies, errors and omissions. The data include samples which may be considered illustrative examples for annotations (Bunt et al., 2016). What is noteworthy here is the fact that suggestions and remarks with regard to limitations and extensions of the ISO standard put forth by the authors of the DialogBank are often subsequently implemented in the updated versions of ISO (Bunt et al., 2018)).

Another point of reference was the corpus of Vietnamese data using sources from IARPA Babel Vietnamese Language Pack (Ngo et al., 2018). The corpus includes 28 selected conversations whose transcripts were manually segmented in turns and then annotated. The agreement scores is 0.84 Fleiss'kappa measure.

In comparison to the previously collected corpora DiaBiz.Kom is much more extended in terms of the number of dialogues, and it covers different fields of communication. All the data were deliberately created to adhere to the research purposes and practical applications. As a consequence, DiaBiz.Kom could be considered the only corpus which is to be used in all main business communication fields. Also, especially in comparison to Switchboard Dialogue Act Corpus, the DiaBiz.Kom corpus uses much more up-to-date language materials. Over the last 30 years the languages have been vastly influenced by overwhelming technological development especially by social networks that have severely modified communication strategies and behaviours. The innovation of our approach is based mainly on the detailed consideration of the mutual influence of dialogue dimensions and communicative functions, as well as on the designation of the new functions not included in the previously used standards. Finally, DiaBiz.Kom was not only fully manually annotated, but also verified in the agreement procedure, which enhances the credibility of the corpus.

3 Data

DiaBiz.Kom corpus development is an annotation effort performed simultaneously with DiaBiz corpus creation (Pęzik et al., 2022). DiaBiz is a large, multi-modal corpus of Polish telephone conversations conducted in varied business settings, comprising 3,766 call center interactions from eight different domains, i.e. banking, energy services, telecommunication, insurance, medical care, debt collection, tourism and car rental. The phone-call interactions were based on 110 distinct customer service call scripts. They were then transcribed and enriched with punctuation. The selected dialogues from DiaBiz corpus are the basis for DiaBiz.Kom annotation.

4 Annotation Procedure

In the first place, the annotations included communicative functions and dimensions. The annotation process was divided into two main stages: (1) initial phase and (2) the final annotation of DiaBiz.Kom corpus. Both stages were performed by a team of qualified linguists with the use of the Inforex system (Marcińczuk et al., 2017).

Initially, the first version of the annotation guidelines was developed with an aim of achieving an appropriate level of inter-annotator agreement. In order to ensure high data quality, we have performed several iterations of manual annotation prior to the annotations performed on the final corpus. Three main sources were successively used as a dialogue base for manual annotation: LUNA corpus, samples of real-life data, and test sample from DiaBiz corpus. Moreover, the team of linguists was systematically expanded, so that we received feedback from annotators not involved in the early stages of guideline development. This was done to avoid a situation in which many of the rules of conduct were not verbalized, but rather were based on the annotator's practical experience. All these efforts aimed at making the guidelines as complete as possible. We calculated inter-annotator agreement by applying *Positive Specific Agreement* measure (Hripcsak and Rothschild, 2005). The first stage was continued until achieving the satisfactory level of the inter-annotator agreement, which involved 8 iterations of manual annotation.

The second stage (the final annotation of DiaBiz.Kom corpus) is currently underway. The inter-annotator agreement is constantly monitored and remains high. The figure presents the improvement of the average level of inter-annotator agreement. It is currently at the level of 0.78 (for annotation borders and categories) and 0.86 (for annotation borders). Every dialogue included in DiaBiz.Kom corpus is annotated by 3 specialists: 2 independently working annotators and a super-annotator who resolves all annotation inconsistencies (for current number of annotations see Appendix A, Table 3).

During the two stages of annotation, we used essentially the same annotation categories (i.e., those specified in the ISO 24617-2 standard). The main difference between the two stages was that during the first annotation stage we annotated a greater variety of texts, coming from diverse sources but greatly resembling the target texts which were later annotated at the second stage. Dividing the processes into stages allowed us to test the model in a variety of domains. Thanks to this solution, we did not adjust the guidelines to data acquired or produced in one specific way. Furthermore, during the first annotation stage the agreement level between annotators was not particularly high. In order to improve the inter-annotator agreement, we decided to work on texts coming from other sources than the target corpus. As a result, the DiaBiz.Kom annotation was quite consistent from the very beginning, and the need for corrections for the first iterations was significantly reduced (the second phase consisted of five iterations). Once the annotation was established (in joint discussions of professional linguists), the super-annotators returned to the previously annotated documents. The correctness of the texts was additionally verified at the dimension marking stage, and in the future – it will also be double-checked at the relation marking stage. Consequently, the material will be verified several times with a small chance of guidelines misinterpretations.

5 Key assumptions and limitations of the guidelines

Even though the annotation guidelines were constantly developed throughout the project, we decided to follow a set of certain unchanging assumptions. The increasing annotator agreement was the result of new specifications that were successively added to the guidelines. Importantly, we were persistently mindful of the versatility of the guidelines, which was primarily aimed at facilitating various possible applications of the corpus in the future. This approach, however, also imposed certain limitations on our work. Below we present the main assumptions as well as some selected issues which we encountered.

One of the main assumptions involved the choice of the communication function for a given utterance as primarily influenced by its goal, effect and the context in which it is set. The form of the annotated statement is considered less important – it may lead to the proper function, but it cannot fully determine its choice. The above mentioned situations may be illustrated with the following examples.

a) *Czy w czymś jeszcze mogę pani pomóc? ('Is there anything else I can help you with?')* [Interaction Structuring]

b) Agent: *Zna Pani swój numer klienta? ('Do you know what your client number is?')* [Propositional Question, dimension: Task]

Client: *Tak 'Yes.'* [Answer, dimension: Task]

Formally, the utterance in (a) points to be interpreted as Questions, but due to its conventionalized form and structuring role in the dialogue, it is marked as Interaction Structuring. Further, when there is a discrepancy between the intention and the effect (reaction), as illustrated in (b), we assign the

specific function on the basis of the direct reaction.

Expressions that can naturally perform different functions depending on the context (e.g. lexemes, such as dobrze 'well', tak 'yes') have been approached more thoroughly in our guidelines, which presently include specific contexts alongside with the plethora of examples illustrating their use in a given function. The goal that the sender wants to achieve is a key criterion here. If the interlocutor's utterance is aimed at obtaining or transmitting some information, it is assigned an appropriate function from the Information-transfer group, even if the form of this statement may initially indicate a function belonging to the Action-discussion group (c) and (d).

c) Proszę powiedzieć, na kogo zarejestrowany jest ten numer. ('Please tell me who this number is registered to.')

d) Proszę w pierwszej kolejności o imię i nazwisko. ('First of all, please give me your name and surname.')

The agent wants to obtain some information from the client, and the usage of the word proszę ('please') is only meant to make the question more polite.

What also needs to be emphasized is that due to the nature of the annotated dialogues, some of the functions described in the ISO/DIS 24617-2 standard were not used (e.g. functions from the Turn Management group), although they were included in the guidelines. As a result, it will be possible to apply them also to other types of dialogues in the future. The nature of the dialogues is related to the difficulty of the texts and this is also expressed by the degree of agreement between annotators (see Appendix A, Table 2). The annotation process showed that particular functions are performed in different ways depending on the type or theme of the dialogue. In the process of working on a given group of dialogues, a situation regularly occurred when certain detailed solutions were developed, which seemed to be completely inappropriate and inapplicable for the next set of texts. Over time, it has been noticed that this repeated situation is dictated by objective reasons. Below we will discuss two illustrative examples of such limitations. First, there are different schemes used for banking dialogues, different – for debt collection, sales, medicine, etc. A characteristic example may present the construction of a banking dialogue, in which the employee is obliged to verify the customer's identity at the beginning of the conversation by asking them a series of questions, the so-called TestQuestions (name and surname, PESEL number, customer number, mother's maiden name, etc.). This element does not occur, for example, in the debt collection dialogues: the client's identity is not strictly verified, as the employee knows who they are calling: most often they just ensure the data available to them are valid (e.g. in the form of PropositionalQuestion: Czy dodzwoniłem się do pani Anny Nowak? 'Have I reached Anna Nowak?' or CheckQuestion: Rozmawiam z panią Anną Nowak, tak? 'I am talking to Ms Anna Nowak, right?'). Second, the choice of a dialogue function is often determined by the relationship between the interlocutors: whether it is based on reciprocity ("equal with equals"), or rather hierarchical, and if hierarchical, who is superior and who is somewhat subordinate to the interlocutor? The following utterances can pose a very clear example: Bardzo proszę o rozłożenie mojej zaległości na raty. 'I would very much like to request that my arrears be spread out in installments' (the debtor is the sender) and Bardzo proszę o natychmiastowe uregulowanie zaległości na numer podany w mailu. 'I strongly request that you pay the arrears immediately to the number provided in the email.' (the debt collector is the sender). Despite the fact that both statements are built on the same syntactic structure, in the former case we are dealing with a Request, while in the latter – with an Instruct (understood as a command).

6 Corpus overview

The aim is to develop a well balanced corpus of annotated dialogues. Thus, we decided to annotate 10 dialogues for each script. As a result DiaBiz.Kom corpus will consist of 1100 annotated dialogues: 260 for banking domain, 150 for energy services, 180 for telecommunication, 110 for insurance, 140 for medical care, 100 for debt collection, 100 for tourism and 60 for car rental. The annotation process continues. All the dialogues (for current statistics see the Table 1) are annotated with communicative functions. The Inforex system enables to export the data using various formats (xml, json, conll or txt). The corpus sample is available under CC BY-NC-ND 4.0 license at: http://hdl.handle.net/11321/886.

There are 138.968 annotated functional segments within DiaBiz.Kom at this stage (see Appendix A, Table 3). The annotations distribution results from the nature of the dialogues. Some

Domain	Dialogues	Tokens
Banking	264	327.731
Debt collection	100	109.189
Energy services	150	131.698
Insurance	110	116.151
Medical care	140	145.765
Car rental	60	71.265
Telecom-munications	180	157.701
Tourism	100	218.465
All	1.104	1.277.965

Table 1: Current size of DiaBiz.Kom annotation in 2+1 system. The numbers refer to the dialogues with final annotation.

communicative functions appear less frequently, e.g. Turn Management functions. We actually recorded few such cases where the annotation of the functions within this group was obligatory. The limited number of such situations may have resulted from the fact that we annotated only those segments whose primary function was to manage dialogue turns. Such an approach was determined by the implicit nature of Turn Management functions (e.g., according to ISO 24617-2: "every time someone starts speaking, this can be interpreted as the performance of a turn-taking act; every time someone stops speaking, this can be interpreted as a turn-release act"). Implied functions were not annotated manually. That is, we did not annotate Turn Management functions in the situations where the speaker, for instance, communicated that they were ready to continue the dialogue (the function we used in such situations was Contact Indication as its definition was more extensive), e.g.

Agent: *Dzień dobry. ('Good morning.')* [initGreeting, dimension: Social Obligations Management]

Client: *Tak, słucham.'Yes, I'm listening.'* [contactIndication, dimension: Contact Management]

All these decisions were preceded by a number of joint discussions of professional linguists over specifically extracted samples from the target corpus (i.e., the examples that had the potential to fall into the category of Turn Management functions). Also, it is significant to mention that Turn Management functions are more natural to polylogues and the annotated corpus consisted solely of dialogues.

7 Conclusions and future work

In this paper we have outlined DiaBiz.Kom – the first corpus, which contains dialogues of various domains with gold standard dialogue act annotations in the Polish language to satisfy the criteria set by machine learning applications. A crucial feature of this resource is the manual layer annotation of information about communication functions (based on ISO standard). The achieved inter-annotator agreement provides a way to use the corpus for the purpose of machine learning. Further development work on DiaBiz.Kom will aim at adding annotation layers – especially those that specify the communicative intent of the speaker (using frame semantics) – and, subsequently, those that determine parameters congruent with the ISO standard (communicative dimensions and relations between annotations). The next step consists in expanding the existing corpus with supplementary dialogues using active learning techniques.

Acknowledgements

The DiaBiz.Kom corpus was developed in the project entitled "CLARIN - Common Language Resources and Technology Infrastructure", which is financed under the 2014-2020 Smart Growth Operational Programme, POIR.04.02.00-00C002/19. We would also like to thank the VoiceLab[2] company for their consultations.

References

Harry Bunt, Jan Alexandersson, Jean Carletta, Jae-Woong Choe, Alex Chengyu Fang, Koiti Hasida, Kiyong Lee, Volha Petukhova, Andrei Popescu-Belis, Laurent Romary, Claudia Soria, and David Traum. 2010. Towards an ISO standard for dialogue act annotation. In *Proceedings of the Seventh International Conference on Language Resources and Evaluation (LREC'10)*, Valletta, Malta. European Language Resources Association (ELRA).

Harry Bunt, Jan Alexandersson, Jae-Woong Choe, Alex Chengyu Fang, Koiti Hasida, Volha Petukhova, Andrei Popescu-Belis, and David Traum. 2012. ISO 24617-2: A semantically-based standard for dialogue annotation. In *Proceedings of the Eighth International Conference on Language Resources and Evaluation (LREC'12)*, pages 430–437, Istanbul, Turkey. European Language Resources Association (ELRA).

Harry Bunt, Volha Petukhova, Andrei Malchanau, Kars Wijnhoven, and Alex Fang. 2016. The DialogBank. In *Proceedings of the Tenth International Conference*

[2] https://voicelab.ai

on Language Resources and Evaluation (LREC'16), pages 3151–3158, Portorož, Slovenia. European Language Resources Association (ELRA).

Harry Bunt, James Pustejovsky, and Kiyong Lee. 2018. Towards an ISO standard for the annotation of quantification. In *Proceedings of the Eleventh International Conference on Language Resources and Evaluation (LREC 2018)*, Miyazaki, Japan. European Language Resources Association (ELRA).

Shammur Absar Chowdhury, Evgeny Stepanov, and Giuseppe Riccardi. 2016. Transfer of corpus-specific dialogue act annotation to ISO standard: Is it worth it? In *Proceedings of the Tenth International Conference on Language Resources and Evaluation (LREC'16)*, pages 132–135, Portorož, Slovenia. European Language Resources Association (ELRA).

Chengyu FANG, Jing CAO, Harry BUNT, and Xiaoyue LIU. 2012. The annotation of the switchboard corpus with the new iso standard for dialogue act analysis. In *Proceedings of the Eighth Joint ACL-ISO Workshop on Interoperable Semantic Annotation*. Eighth Joint ACL-ISO Workshop on Interoperable Semantic Annotation ; Conference date: 03-10-2012 Through 05-10-2012.

Emer Gilmartin, Christian Saam, Brendan Spillane, Maria O'Reilly, Ketong Su, Arturo Calvo, Loredana Cerrato, Killian Levacher, Nick Campbell, and Vincent Wade. 2018. The ADELE corpus of dyadic social text conversations:dialog act annotation with ISO 24617-2. In *Proceedings of the Eleventh International Conference on Language Resources and Evaluation (LREC 2018)*, Miyazaki, Japan. European Language Resources Association (ELRA).

George Hripcsak and Adam S. Rothschild. 2005. Agreement, the F-Measure, and Reliability in Information Retrieval. *Journal of the American Medical Informatics Association*, 12(3):296–298.

Simon Keizer, Harry Bunt, and Volha Petukhova. 2011. *Multidimensional Dialogue Management*, pages 57–86. Springer Berlin Heidelberg, Berlin, Heidelberg.

Michał Marcińczuk, Marcin Oleksy, and Jan Kocoń. 2017. Inforex — a collaborative system for text corpora annotation and analysis. In *Proceedings of the International Conference Recent Advances in Natural Language Processing, RANLP 2017*, pages 473–482, Varna, Bulgaria. INCOMA Ltd.

Thi-Lan Ngo, Pham Khac Linh, and Hideaki Takeda. 2018. A Vietnamese dialog act corpus based on ISO 24617-2 standard. In *Proceedings of the Eleventh International Conference on Language Resources and Evaluation (LREC 2018)*, Miyazaki, Japan. European Language Resources Association (ELRA).

Thi-Lan Ngo, Son-Bao Pham, Khac-Linh Pham, Xuan-Hieu Phan, and Minh-Son Cao. 2017. Dialogue act segmentation for vietnamese human-human conversational texts. In *2017 9th International Conference on Knowledge and Systems Engineering (KSE)*, pages 203–208.

Volha Petukhova, Martin Gropp, Dietrich Klakow, Gregor Eigner, Mario Topf, Stefan Srb, Petr Motlicek, Blaise Potard, John Dines, Olivier Deroo, Ronny Egeler, Uwe Meinz, Steffen Liersch, and Anna Schmidt. 2014. The DBOX corpus collection of spoken human-human and human-machine dialogues. In *Proceedings of the Ninth International Conference on Language Resources and Evaluation (LREC'14)*, pages 252–258, Reykjavik, Iceland. European Language Resources Association (ELRA).

Piotr Pęzik, Gosia Krawentek, Sylwia Karasińska, Paweł Wilk, Paulina Rybińska, Anna Cichosz, Angelika Peljak-Łapińska, Mikołaj Deckert, and Michał Adamczyk. 2022. DiaBiz – an annotated corpus of polish call center dialogs. In *Proceedings of the Language Resources and Evaluation Conference*, pages 723–726, Marseille, France. European Language Resources Association.

A Appendix. Tables

	Debt collection	Insurance	Medical Care	Car rental	Telcom
Information-seeking					
setQuestion	0.73 (68)	0.70 (99)	0.82 (115)	0.64 (47)	0.70 (55)
checkQuestion	0.61 (32)	0.35 (33)	0.74 (47)	0.71 (28)	0.35 (12)
choiceQuestion	0.64 (12)	0.67 (15)	0.87 (52)	0.50 (5)	0.67 (16)
propositionalQuestion	0.73 (76)	0.66 (57)	0.70 (77)	0.83 (101)	0.66 (120)
Information-providing					
inform	0.58 (477)	0.57 (489)	0.61 (477)	0.61 (394)	0.57 (367)
answer	0.73 (155)	0.67 (197)	0.75 (239)	0.71 (167)	0.67 (155)
confirm	0.72 (30)	0.44 (27)	0.71 (44)	0.69 (18)	0.44 (11)
Directives					
request	0.28 (17)	0.67 (17)	0.65 (49)	0.57 (18)	0.67 (26)
suggest	0.31 (29)	0.58 (18)	0.17 (16)	0.36 (11)	0.58 (25)
acceptOffer	0.67 (1)	1.00 (5)	0.25 (7)	1.00 (1)	1.00 (4)
Commisives					
offer	0.12 (8)	0.38 (12)	0.10 (13)	0.50 (1)	0.38 (23)
acceptRequest	0.00 (7)	0.53 (7)	0.59 (16)	0.32 (12)	0.53 (11)
Discourse Structuring					
interactionStructuring	0.68 (161)	0.69 (247)	0.62 (253)	0.56 (163)	0.69 (150)
Feedback					
autoPositive	0.72 (184)	0.78 (261)	0.71 (305)	0.66 (169)	0.78 (17)
alloPositive	0.57 (9)	0.50 (19)	0.60 (13)	0.86 (3)	0.50 (7)
all categories	0.74 (2827)	0.73 (2909)	0.75 (2999)	0.74 (2480)	0.73 (2107)

Table 2: Inter-annotator agreement (*PSA*) for selected communicative functions regarding 5 domains. The agreement is based on annotations of the same two annotators performed on 20 dialogues within each domain. The number in the brackets corresponds to the number of final annotations submitted by the independent super-annotator.

Communicative function	Number of annotations	Communicative function	Number of annotations
Information-seeking functions		**Contact Management Functions**	
checkQuestion	2.158	contactCheck	123
choiceQuestion	959	contactIndication	3.284
propositionalQuestion	4.063	**Discourse-structuring functions**	
setQuestion	4.223	interactionStructuring	14.195
testQuestion	1.201	opening	1.093
Information-providing functions		**Feedback functions**	
agreement	886	alloNegative	11
answer	9.741	alloPositive	517
confirm	1.680	autoNegative	73
correction	14	autoPositive	12.892
disagreement	102	feedbackElicitation	462
disconfirm	71	**Own- and Partner-Management Functions**	
inform	24.090	completion	27
Directives		correctMisspeaking	12
acceptOffer	309	retraction	189
addressOffer	9	selfCorrection	4.504
declineOffer	76	selfError	422
instruct	1.082	**Social Obligations Management Functions**	
request	1.558	acceptApology	35
suggest	1.138	acceptThanking	35
Commissives		apology	350
acceptRequest	619	compliment	7
acceptSuggest	211	congratulation	2
addressRequest	13	initGoodbye	1.151
addressSuggest	6	initGreeting	1.161
declineRequest	24	initSelfIntroduction	1.148
declineSuggest	35	returnGoodbye	1.261
offer	668	returnGreeting	955
promise	653	returnSelfIntroduction	364
		sympathyExpression	186
		thanking	3.185
		Time-management functions	
		pausing	410
		stalling	35.317
		Turn-management functions	
		turnAssign	2
		turnGrab	6
		All	**138.968**

Table 3: Statistics for the final annotations within the DiaBiz.Kom corpus (current state).

Towards explainable evaluation of language models on the semantic similarity of visual concepts

Maria Lymperaiou, George Manoliadis, Orfeas Menis Mastromichalakis, Edmund G. Dervakos and **Giorgos Stamou**
Artificial Intelligence and Learning Systems Laboratory
School of Electrical and Computer Engineering
National Technical University of Athens
{marialymp, eddiedervakos}@islab.ntua.gr, gmanoliad@mail.ntua.gr, menorf@ails.ece.ntua.gr, gstam@cs.ntua.gr

Abstract

Recent breakthroughs in NLP research, such as the advent of Transformer models have indisputably contributed to major advancements in several tasks. However, few works research robustness and explainability issues of their evaluation strategies. In this work, we examine the behavior of high-performing pre-trained language models, focusing on the task of semantic similarity for visual vocabularies. First, we address the need for explainable evaluation metrics, necessary for understanding the conceptual quality of retrieved instances. Our proposed metrics provide valuable insights in local and global level, showcasing the inabilities of widely used approaches. Secondly, adversarial interventions on salient query semantics expose vulnerabilities of opaque metrics and highlight patterns in learned linguistic representations.

1 Introduction

Semantic similarity between pairs of sentences serves a large variety of applications in the field of natural language processing, such as document retrieval, text classification, question answering and others. Even though such tasks have risen in popularity since the introduction of the Transformers (Vaswani et al., 2017), and despite the attention given on robustness and transparency of NLP transformers (Hendrycks et al., 2020; Hsieh et al., 2019; Baan et al., 2019) few efforts have addressed explainable evaluation (Leiter et al., 2022).

Text-Image retrieval is a real world semantic similarity application where the task is to feed a textual input to a system, and receive an image as a response. Visual details of the retrieved instance need to accurately correspond to the textual descriptions, often in a fine-grained fashion. Any mismatch between modalities can be easily perceived by humans, and captured by automated metrics. Such evident disagreements can act as starting points for further investigation, revealing inner processes on the semantic matching procedures.

In this work, we aim to unveil the evaluation strategy of semantic similarity models. Specifically, we apply pre-trained transformers on visual vocabularies and obtain results via ranking. First, we address the shortcomings of traditional ranking metrics (Manning et al., 2008), which provide either a binary answer (item found in top-k items or not), or position-informed variants (item found in the k-th position). However, such measures cannot provide detailed insights regarding the contribution of the scene constituents to the rank position. For example, if an instance is ranked in the k-th position, items in previous k-1 positions may be highly relevant to the ground truth one or on the contrary, highly irrelevant. To this end, we propose novel explainable ranking evaluation metrics that decompose and quantify the conceptual differences between ground truth and retrieved instances in local and global level. Even then, we observe that existing metrics lack a way to assess whether the top-ranked items are actually relevant to the query. For this reason, we construct adversarial queries where an attribute is replaced with a conceptually divergent one, in order to evaluate the response of a ranking system to distorted inputs. In all cases, frequently misperceived semantics captured by our evaluation framework reveal patterns imprinted in the learned representations of language models. Our overall approach is applicable regardless of the chosen language model or ranking system.

2 Related work

A whole new world of possibilities in NLP has opened since the advent of the Transformer (Vaswani et al., 2017), with successful milestones such as BERT (Devlin et al., 2019) and RoBERTa (Liu et al., 2019) serving as backbone models for many applications. MPNet (Song et al., 2020) combines permuted language modeling with masked language modeling to overcome the shortcomings of its predecessors. Towards reducing model

Proceedings of the 29th International Conference on Computational Linguistics, pages 3639–3658
October 12–17, 2022.

sizes, knowledge distillation followed in DistilBERT/DistilRoBERTa (Sanh et al., 2020), MiniLM (Wang et al., 2020) and TinyBERT (Jiao et al., 2020), as well as parameter reduction techniques implemented in ALBERT (Lan et al., 2020) achieve more compact models while maintaining performance. The textual semantic similarity task was greatly benefited by Sentence-BERT (SBERT) (Reimers and Gurevych, 2019), a siamese-BERT variant that allows efficient embedding representations using the aforementioned models accordingly.

Traditional evaluation metrics such as HITS, Mean Reciprocal Rank (MRR), precision, recall and F-score (Manning et al., 2008) have dominated the field of information retrieval. While these metrics serve the purpose of assessing the retrieved information, they do not provide explainable means of justification. Explainable evaluation metrics (Leiter et al., 2022) aim to address this challenge.

Lack of trust of neural methods due to biases, outdated training, and inaccurate assumptions has led to the need for explainable methods in language models. Research towards that direction has utilized Concept Attributions (Sai et al., 2021; Yuan et al., 2021), Chunk Alignments (Magnolini et al., 2016), Feature Importance (Rubino et al., 2021; Treviso et al., 2021), or Explanations by Simplification (Kaster et al., 2021). Adversarial examples can also provide insights regarding the inner workings of obscure models, and are closely related to counterfactual explanations, placing them in the broader area of explainability (Linardatos et al., 2020). Numerous works address the problem of adversarial examples for natural language models (Zhang et al., 2020b), with recent methods addressing the robustness of NLP models such as BERT through adversarial examples/attacks (Jin et al., 2019; Li et al., 2020). In this work, we approach adversarial examples from a different perspective, first of all tackling a different problem than classification that most works do, and secondly by realizing the creation of adversarial examples on the semantic rather than the linguistic level, investigating the effect of semantic changes on the ranking of text-image retrieval systems.

3 Overview

Our workflow consists of three stages: ***Representation***, ***ranking*** and ***explainable evaluation***. We view text-image retrieval as a query-corpus retrieval problem, exclusively exploiting linguistic information for representation and ranking, while revealing visual information only at the evaluation stage, where we compare retrieved images with the ground truth ones. As input, we consider a dataset of size N that contains complex scene images $I_i \in \mathcal{I}$, accompanied by query-corpus pairs $(q_i, c_i), q_i \in \mathcal{Q}, c_i \in \mathcal{C}, i = 1, 2, ..., N$ with each corpus c_i consisting of an arbitrary number of sentences $s_j, j = 1, 2, ..., l_c$. In the ***representation*** stage, pre-trained sentence similarity transformers $M \in \mathcal{M}$ from SBERT embed $\mathcal{Q}, \mathcal{C}$ instances in a common vector space U. Cosine similarity scores between query-corpus embedding pairs in U are sorted to provide a rank R_i per query q_i in the ***ranking*** stage, with R_i either lead to *success*, if the ground truth image I_{g_i} with corpus c_i is returned at the top of the rank, or *failure* otherwise. We provide a visual demonstration of the ranking procedure in Figure 2. All failures per model M, i.e. image pairs (I_g, I_r) for which $I_g \neq I_r$ are stored in a set $\mathcal{F}$, which is further passed to the ***evaluation*** stage. We then employ three methods to provide an understanding and evaluation towards failures: transparent ranking metrics (section 4), human evaluation and adversarial re-ranking (section 5). The overview of our proposal is presented in Figure 1.

3.1 Visual concepts in language

Visual vocabularies contain descriptions about real life scenes, including objects, relationships and attributes. Datasets that connect visual vocabularies paired with images, such as Visual Genome (Krishna et al., 2016), COCO (Lin et al., 2014) and Flickr (Young et al., 2014) set our sources to construct purely textual query-corpus pairs, assuming that necessary visual information is contained within the high quality annotations of those datasets. In particular, the annotation diversity allows either shorter, global descriptions, as in Flickr and COCO captions, or detailed descriptions in local level, as in Visual Genome region descriptions, concatenated in a corpus c_i per image I_i.

3.2 Optimal embedding representation

Obtaining an overall representation of a corpus c_i is not trivial, as existing transformers can handle up to a certain number of input tokens per sentence. To resolve this, we can independently embed each corpus sentence $s_j \in c_i, j = 1, 2, ..., l_c$ using a model $M \in \mathcal{M}$, and then calculate the average of all vectors v_j^c. Therefore, $u_i^c = \frac{1}{l_c} \sum_{j=1}^{l_c} v_j^c \in U$ serves as the averaged representation for c_i. An-

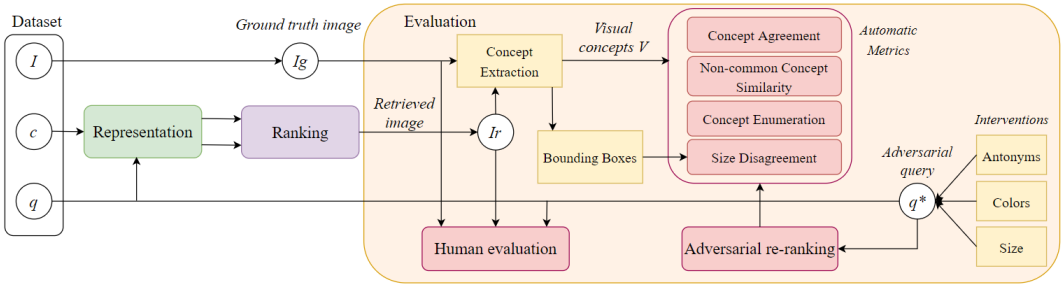

Figure 1: Overview of our workflow towards explainable evaluation.

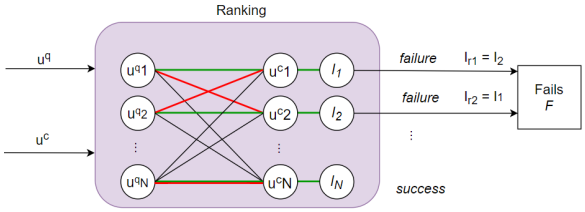

Figure 2: A closer look at ranking procedure. Green lines denote ground truth matchings, while red lines indicate matchings selected from maximum cosine similarity scores between query and corpus embeddings.

other approach is to leverage state-of-the-art abstractive summarizers (Zhang et al., 2020a; Raffel et al., 2020) to obtain a meaningful shorter version of c_i while maintaining semantics as much as possible, and then apply $M \in \mathcal{M}$ only once per c_i. Query representations $u_i^c \in U$ are produced by inserting each $q_i \in \mathcal{Q}$ in a model $M \in \mathcal{M}$, or by averaging over representations when q_i comprises from more than one sentences.

3.3 Ranking

Given a model M, each query representation $u_i^q \in U$ is paired with all corpus representations $\{u_1^c, u_2^c, ..., u_N^c\} \in U$, and cosine similarity scores are calculated for each pair. Higher cosine similarity scores yield more similar representations, therefore sorting from higher to lower scores provides the ranking R_i per q_i. The process is repeated for all N images resulting in N^2 calculations.

Traditional metrics evaluate the ranking success, coarsely indicating the representation quality of each $M \in \mathcal{M}$. Recall@k returns the proportion of ground truth images found in top-k ranked instances for all queries $q_1, q_2, ...q_N$, given that each q_i has only one ground truth c_i. Mean Reciprocal Rank (MRR) is the averaged of the inverse of the ground truth rank position rank_i for each c_i given q_i, considering the top-k items: MRR@k = $\frac{1}{N}\sum_{i=1}^{N}\frac{1}{\mathsf{rank}_i}$ for each $\mathsf{rank}_i \geq k$. We calculate Recall@k and MRR@k for k=5, 10, N. Also, we calculate the median rank position for all c_i.

4 Explaining failures

We count as failure $f_i = (I_g, I_r)_i \in \mathcal{F}$ any instance of a ground truth image I_{g_i} with corpus c_i that was not ranked in the first position ($\mathsf{rank}_i \neq 1$) given q_i; instead another image $I_{r_i} \neq I_{g_i}$ with $c_r \neq c_i$ achieved $\mathsf{rank}_r = 1$. Following the 'blind' evaluation strategy of traditional ranking metrics, we provide a measure of retrieval failures as the cardinality of the failure set: $F = |\mathcal{F}|$ for each M.

However, it is not possible to verify if I_{r_i} can accurately satisfy q_i without exploiting visual information. To this end, we exploit visual annotations and human perception to quantify the suitability of each $I_{r_i} \in f_i$ with respect to q_i. By decomposing all semantics that contribute to the suitability of each I_{r_i} we obtain a discrete and transparent conceptual measure of similarity between $(I_g, I_r)_i$.

4.1 Towards explainable evaluation metrics

We design four evaluation stages for all failures f_i, starting from more influential concepts and moving towards less prevalent details. Visual concepts are focused on scene ***objects***. For fair comparison with traditional ranking metrics, we demonstrate a *query-agnostic* evaluation approach: we compare concepts between retrieved and ground truth images without considering query semantics. In the next paragraphs we drop i subscript for simplicity.

Concept agreement - CA Considering $\mathcal{V}$ as a set of visual concepts, **concept agreement** measures the percentage of ground truth concepts $\mathcal{V}(I_g)$ contained in the retrieved concept set $\mathcal{V}(I_r)$ over all $\mathcal{V}(I_g)$ concepts for each f_i. Let $V_{(g,r)} = \mathcal{V}(I_g) \cap \mathcal{V}(I_r)$ the set of common concepts:

$$CA_f = \frac{|V_{(g,r)}|}{|\mathcal{V}(I_g)|}, f = (I_g, I_r)$$

Higher ***CA*** indicates higher concept similarity. For example, if $\mathcal{V}(I_r) = \{\mathsf{Dog}, \mathsf{Frisbee}, \mathsf{Park}\}$ and $\mathcal{V}(I_g) = \{\mathsf{Dog}, \mathsf{Ball}, \mathsf{Park}\}$, then the ***CA***=$\frac{2}{3}$. On the other hand, if $\mathcal{V}'(I_r) = \{\mathsf{Cat}, \mathsf{Fish}\}$, then ***CA***=0, as no overlap exists. This way we can confidently conclude that the first retrieved image is conceptually closer to the ground truth than the second, and by extension the model used to retrieve the first image is better with respect to ***CA***.

Non-common concept similarity - NCS aims to provide a distance measure between concepts present exclusively in either $\mathcal{V}(I_g)$ or $\mathcal{V}(I_r)$. For example, we would expect the set $\{\mathsf{Dog}, \mathsf{Frisbee}, \mathsf{Park}\}$ to be more similar to $\{\mathsf{Dog}, \mathsf{Ball}, \mathsf{Park}\}$ than $\{\mathsf{Dog}, \mathsf{Cat}, \mathsf{Park}\}$, since the non-common concept Frisbee is conceptually closer to Ball than Cat. Mathematically, let $D_g = V_{(g,r)} - \mathcal{V}(I_r)$ and $D_r = V_{(g,r)} - \mathcal{V}(I_g)$, with both $D_g, D_r \neq \emptyset$. Other than that, D_g and D_r may contain different number of concepts. Then, a measure of concept distance can be provided by calculating the path similarity score ps of corresponding WordNet (Fellbaum, 1998) synset pairs, based on the shortest available path between those two concepts. Path similarity ps ranges between 0 and 1.

An *optimistic* ***NCS*** metric returns the maximum possible cumulative ps averaged over the number of pairs, by appropriately selecting concept pairs between non-empty D_g and D_r. The maximization of ***NCS*** requires a dynamic programming solution, as naive strategies taking into account all possible D_g and D_r pairs would yield a factorial amount of combinations. To trespass this prohibitive complexity, we create a bipartite graph $G = (D_g, D_r, E)$ from D_g and D_r: all concept nodes from the one set are matched with all the nodes of the other via edges $e_y \in E, y = 1, 2, ..., |D_g| \times |D_r|$, while no edges are allowed within the same set. Edge weights w_{e_y} correspond to WordNet ps scores between synsets of connected nodes.

Consequently, the *maximum weight bipartite matching* on G refers to pairing D_g and D_r concepts so that the cumulative edge weight is maximized. An optimized version of the Hungarian algorithm (Kuhn, 1955; Galil, 1986) implemented by NetworkX[1] reduces the computational complexity of finding the maximum ps to $O(|V|^3)$, where $|V| = max(|D_g|, |D_r|)$.

Therefore, ***NCS*** can be written as:

$$NCS_f = avg(max\_weight\_match(G)),$$
$$G = (V_{(g,r)} - \mathcal{V}(I_r), V_{(g,r)} - \mathcal{V}(I_g), E))$$

Higher ***NCS*** scores reveal more similar concepts.

Concept enumeration - CE Real world scenes may contain repeated instances of same-class concepts, forming concept multisets $\mathcal{V}_m = \{(\mathcal{V}_1, |\mathcal{V}_1|), (\mathcal{V}_2, |\mathcal{V}_2|), ..., (\mathcal{V}_x, |\mathcal{V}_x|)\}$, where $\mathcal{V}_1, \mathcal{V}_2, ..., \mathcal{V}_x$ denote concept categories, and $|\mathcal{V}_1|, |\mathcal{V}_2|, ..., |\mathcal{V}_x|$ cardinalities per category. The cardinality per concept category is called concept multiplicity in the multiset. ***CE*** penalizes differences in multiplicities between common concepts of I_g and I_r for each f_i:

$$CE_f = \sum_{j=1}^{x} ||\mathcal{V}_j(I_g)| - |\mathcal{V}_j(I_r)||_{\mathcal{V}_j(I_g)=\mathcal{V}_j(I_r)}$$

Higher ***CE*** scores demonstrate higher enumeration disagreement, deeming lower ***CE*** values more favorable. For example, if I_g contained 10 dogs and 1 frisbee $\{(\mathsf{Dog}, 10), (\mathsf{Frisbee}, 1)\}$, a retrieved I_r with 1 dog and 1 frisbee would have ***CE***=9, while an I'_r with 10 dogs and 1 ball would have a ***CE***=0. Therefore, the first image yields a worse CE score than the second, even though the second would have worse ***CA*** and ***NCS*** scores than the first one.

Size disagreement - SD Even in cases where there is a high agreement of objects and multiplicities between I_g and I_r, disagreement in object sizes may correspond to semantically divergent scenes. For example an image with a dog in the foreground (large bounding box) is different than an image of a dog in the background (small bounding box). To capture this difference, we design an *optimistic* ***SD*** metric which returns the area differences of bounding boxes $\mathcal{D}_A = |A_g - A_r|$ for all available object matchings. Such matchings occur by pairing concepts of the same category u between $\mathcal{I}_g$ and $\mathcal{I}_r$ up to the point that no more unique pairs can be constructed. This is equivalent of creating bipartite graph $G_u = (\mathcal{V}_u(I_g), \mathcal{V}_u(I_r), E)$, where $\mathcal{V}_u(I_g), \mathcal{V}_u(I_r)$ belong in the same u and edge weights $w_{e_y}, e_y \in E, y = 1, 2, ..., |\mathcal{V}(I_g)| \times |\mathcal{V}(I_r)|$ denote the area difference $\mathcal{D}_A$ between concept nodes. Pairing concepts with similar bounding box areas can be considered as the optimal choice, therefore node pairs connected by lower edge weights w_{e_y} are preferred. Finding the *minimum weight matching* provides the most similar pairs size-wise, and can be solved in polynomial

[1] NetworkX max weight matching

time using the NetworkX[2] implementation of Karp algorithm (Karp, 1978). The matching process is repeated for all concept categories in the multiset $\mathcal{V}_m$, resulting in a set of graphs G_m:

$$SD_f = \sum^{\mathcal{V}_u \in \mathcal{V}_m} avg(min\_weight\_match(G_u)),$$

$$G_u = (\mathcal{V}_u(I_g), \mathcal{V}_u(I_r), E), G_u \in G_m$$

A simplified *binary* version of SD increases a sum if area differences of paired concepts are above a predefined threshold T_D.

4.2 Human evaluation via crowdsourcing

Query-agnostic evaluation regards all scene semantics, even if in fact they are not present in the query. On the other hand, incorporating query information at evaluation stage conditions concept importance upon the presence of a concept in the query, forming a *query-informed* evaluation strategy. We conducted *query-informed* human evaluation experiments considering all failures in $\mathcal{F}$ and penalizing semantic disagreements only if those semantics are mentioned in q_i. Evaluators were primarily asked to mark which salient semantics were clearly misinterpreted in retrieved images with respect to the given query among the options: *object class, object color, object enumeration, action, size, details*. Otherwise, if I_{r_i} can be considered as conceptually similar to I_{g_i}, it is marked as *successful alternative*. Additionally, the overall retrieval quality is cross checked via qualitative ratings, assessing the conceptual similarity between I_{g_i}, I_{r_i} given q_i. Despite being unfair to compare with the -stricter- automated metrics, we expect lower values for *object enumeration* and *size* failure classes comparing to ***CE, SD*** metrics.

The crowdsourcing experiment reveals the most frequently misinterpreted attributes or combinations of attributes. Loss of conceptual information can be either attributed to dataset quality, i.e. salient query semantics not present in corpus, or on the capacity of the linguistic representations. Keyword matching between q_i and c_i excludes cases where the ground truth query-corpus pair contains very few common concepts, enabling the remaining samples to reveal patterns within the learned representations.

5 Adversarial re-ranking

We create adversarial queries $q \rightarrow q^*$ targeting key attributes and produce respective representations in U, upon which adversarial rankings R^* per q^* are extracted. Figure 3 provides the causal graph of adversarial interventions for any $q_i \in \mathcal{Q}$.

5.1 Substituting salient attributes

We perturb salient semantics in queries $q_i \in \mathcal{Q}$, producing $q_i^* \in \mathcal{Q}^*$, and evaluate the changes occurring in the rank. An appropriate non-minimal adversarial perturbation must conceptually reverse salient semantics, be focused on an individual semantic each time, and the resulting query q_i^* should be linguistically correct. With respect to those requirements and in order to restrict the search space of adversarials, we target substituting ***object attributes***. Initially, generic adversarial queries include replacing attributes with their antonyms. More refined subsequent adversarials focus on replacing object colors and sizes; such substitutions are discrete, fast and controllable.

Antonyms are extracted via relevant WordNet functions for any adjective present in a query. If more than one antonyms are returned, one is randomly picked to substitute the actual word.

Color substitution refers to changing colors present in the sentence with another distant color. Color distance is provided via the RGB values of Matplotlib colors[3]. We set a proximity threshold to ensure perceptually non-negligible color changes. Two possible substitutions are attempted: either considering all RGB colors (*color-all)*, or colors only mentioned in the dataset (*color-in)*.

Size substitution is an antonym substitution specialized in sizes. Words such as *large, big, enormous, huge* are substituted with a random choice among *small, little, minor, tiny* and vice versa.

5.2 Re-ranking evaluation

Adversarial query representations $u_i^{q*} \in U$ of $q^* \neq q$ with $u_i^c \in U$ of corpus c_i may directly influence the final ranking $R*$ when $\mathsf{rank}_i^* < \mathsf{rank}_i$ or inversely $\mathsf{rank}_i^* > \mathsf{rank}_i$. Intuitively, any non-negligible perturbation of q should result in worse position $\mathsf{rank}_i^* > \mathsf{rank}_i$, as the adversarial query representation u_i^{q*} would diverge from c_i comparing

[2]NetworkX min weight full matching

[3]Matplotlib colors

to u_i^q, due to the substitution of the actual semantic with a conceptually different one. However, given the relative nature of ranking, some instances may stay in the same position rank_i^*=rank_i, or even go higher. Ascending in the rank does not imply a better q_i^*, c_i matching, except if their in between cosine similarity increases; instead, the distorted representations push lower previously higher-ranked instances, virtually improving some rank_i^*. In any case, we expect all ranking metrics to perceptibly drop, as we pull apart ground truth matchings in U.

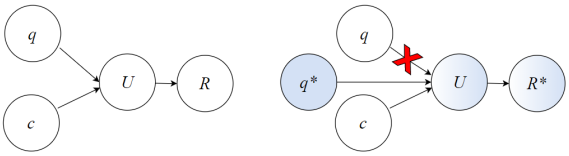

Figure 3: Conventional causal graph (left) and adversarial intervention causal graph (right) when $q \rightarrow q^*$.

6 Experiments

We implement the same experimental strategy on Flickr8k, Flickr30k and VG ∩ COCO datasets, obtaining query-corpus representations in a common semantic space U for a variety of models $M \in \mathcal{M}$. The code for our experiments is provided in submitted supplementary material.

6.1 Ranking results

We focus on presenting experiments on VG ∩ COCO of N=34k images, which is the most challenging: the dataset size N itself, as well as the more detailed region descriptions of Visual Genome which comprise a larger corpus set $\mathcal{C}$, require accurate linguistic representations in order to retrieve more relevant images. Table 1 presents a subset of ranking results on VG ∩ COCO.

6.2 Optimal model choice

Selected language models are designed for semantic similarity, and according to the datasets they have been pre-trained on, they can be divided in: *all-* models pretrained and fine-tuned on 1B sentence pairs from multiple sources; *multi-qa-* trained on 215M diverse question-answer pairs, learning to map queries to passages; *stsb-* models trained on the STSbenchmark, which contains sentence pairs annotated with similarity scores; *paraphrase-* models with more than 86M paraphrase sentence pairs containing more challenging and uncurated characteristics comparing to STSb; *nq-* models trained with 100k real Google search queries mapped to Wikipedia passages; *nli-* models incorporate natural language inference data pairs (premise/hypothesis), included in AllNLI dataset.

By conducting a large number of experiments to estimate the performance of such models on visual vocabularies, we observe certain patterns in ranking results. In all experiments, most paraphrase models consistently outperform the rest. Paraphrasers have been pre-trained on image captions (COCO, Flickr), which actually serve as paraphrasing data: during the construction of these datasets, annotators have independently produced varying descriptions for the same concepts. Query-corpus pairs can be viewed as the one being a paraphrase of the other, thus paraphrasers have learned a suitable representation for this matching, together with their exposure to visual vocabularies.

6.3 Explainable evaluation

In all following experiments, we consider results from the best performing model MiniLM-L3 on VG ∩ COCO. In total, F=28817 queries failed to retrieve their corresponding ground truth I_g.

Local evaluation The real power of our proposed metrics lies in local level. We present an example from the *color* and *details* failure category below. Given a query q_i (caption), Figure 4 shows the ground truth I_{g_i} (left) and the retrieved I_{r_i} (right).

Figure 4: A herd of zebras grazing in a lush green field

The set of ground truth object synsets is {trunk.n.01, hill.n.01, tree.n.01, sky.n.01, field.n.01, branch.n.01, head.n.01, leg.n.01, leaf.n.01, zebra.n.01, mane.n.01} of cardinality 11, and the set of retrieved ones is {grassland.n.01, field.n.01, zebra.n.01, mane.n.01, grass.n.01} of cardinality 5. Common synsets are {zebra.n.01, mane.n.01, field.n.01} of cardinality=3, resulting in $\boldsymbol{CA}_i$=27.28%. Regarding ***NCS***, the constructed bipartite graph G contains $|V|$=10, $|E|$=16, and the best matched synset pairs according to the *maximum weight matching* are {hill.n.01, grassland.n.01} with ps=0.111, and {tree.n.01, grass.n.01} with ps=0.167. The average ps for all

Name	Recall(%)↑			MRR(%)↑			Median	Fails
	@1	@5	@10	@5	@10	@*all*	Rank ↓	(%) ↓
all MiniLM L12	13.31	26.93	34.31	18.18	19.16	20.45	34	86.69
paraphrase MiniLM L12	14.24	29.89	38.34	19.89	21.02	22.37	24	85.76
paraphrase MiniLM L3	**15.31**	**30.92**	**39.48**	**20.97**	**22.11**	**23.42**	23	**84.69**
paraphrase MiniLM L6	14.39	30.01	38.51	19.96	21.10	22.46	24	85.61
paraphrase TinyBERT L6	14.55	30.38	39.12	20.27	21.43	22.82	**22**	85.45
paraphrase albert base	13.12	27.83	36.11	18.43	19.54	20.91	28	86.88
paraphrase albert small	14.56	30.25	39.04	20.23	21.40	22.75	23	85.44
paraphrase distilroberta	14.51	30.39	38.91	20.23	21.36	22.74	**22**	85.49
paraphrase mpnet	13.99	28.99	37.40	19.39	20.50	21.84	26	86.01
stsb distilroberta base	13.41	27.04	34.28	18.32	19.28	20.55	35	86.59
stsb mpnet base	14.05	28.26	35.93	19.23	20.24	21.49	32	85.95
stsb roberta base	13.69	27.38	34.79	18.67	19.65	20.92	34	86.31

Table 1: Rank results on the VG ∩ COCO dataset for our best 12 models. Full Table (11) in Appendix.

matched pairs leads to $\boldsymbol{NCS}_i$=0.139. Common object enumeration provides the following multisets: $\mathcal{V}_m^g$ = {zebra.n.01, 5, field.n.01, 1, mane.n.01, 1} and $\mathcal{V}_m^r$ ={zebra.n.01, 7, field.n.01, 1, mane.n.01, 5}. Therefore, $\boldsymbol{CE}_i$= 6. As for ***SD*** for T_D=1, 3 bipartite graphs are created for the 3 common synsets. The first graph G_{mane} contains $|V|$=6 and $|E|$=5, resulting in 1 *minimum weight matching* of weight D_A=2.30$\geq T_D$. Therefore $\boldsymbol{SD}_{mane}$=1=$\boldsymbol{SD}_i$. The second graph G_{zebra} consists of $|V|$=11 and $|E|$=28, resulting in 4 *minimum weight matchings*, from which none trespassed the threshold T_D, resulting in $\boldsymbol{SD}_{zebra}$=0, thus maintaining $\boldsymbol{SD}_i$=1. Finally, the G_{field} graph of $|V|$=2 and $|E|$=1, leads to 1 *minimum weight matching* of weight D_A=1.369$\geq$ T_D, resulting in $\boldsymbol{SD}_{field}$=1, which increases the total sum $\boldsymbol{SD}_i$=2. Having in total 6 matches for all three graphs, the ***averaged*** $\boldsymbol{SD}_i$=33.3% for this f_i.

Perceptually, a major I_{g_i}, I_{r_i} disagreement can be attributed to not satisfying *lush, green* attributes rather than semantics addressed by our metrics. Indeed, human evaluators rated I_{r_i}-q_i relevance with 6/10 on average and all of them marked *details* and *color* as the failure categories. As for traditional ranking metrics, I_{g_i} was placed in rank_{g_i}=294 with reciprocal rank score of 0.0034 and R@k=0, k=1,5,10. Obviously, we cannot extract much information in local level about how much I_{g_i} and I_{r_i} conceptually deviate and what we should potentially regard and request from retrieved instances (*colors* such as *green* instead of *yellow*, *details* such as *lush* instead of *arid*) to ascend in the rank. To this end, we conclude that traditional ranking metrics are only helpful in a very abstract level.

Human evaluation Results regarding misperceived semantics classes are presented in Table 2. The 82.52% of evaluated image pairs resulted in one semantic class disagreement, while the remaining 14.56% and 2.91% contained two and three semantic class disagreements respectively. The average rating over all classes was 8.47/10.

First, human evaluation experiments can indicate the degree of strictness of our automated metrics, as any *query-agnostic* metric may over-penalize semantics present in the I_{g_i} and c_i but not in q_i. Indeed, *query-informed* variants of our metrics are more relaxed. Moreover, patterns in reported failures also indicate patterns imprinted in the learned linguistic representations. Traditional ranking metrics cannot derive such fine-grained observations.

Alternatives %	Obj. class	% color	% enum.	% size	Action %	Detail %
23.30	5.83	17.48	11.65	6.14	7.77	54.37

Table 2: Semantics disagreement percentage per class

Rules in failures The frequent class of *successful alternatives* indicates that even when automatic metrics consider an I_{r_i} as failure, it may actually be a conceptually correct answer to q_i. Qualitative analysis over *successful alternatives* further demonstrated that almost all (I_{g_i}, I_{r_i}) pairs of this class were visually divergent, even though conceptually equivalent. Also, *details* and *object color* failure classes appeared often enough, indicating that those semantics are rather bypassed in order for others to be preserved. Combinations of semantics did not present any significant pattern; all seman-

tics co-occurrences appeared in less than 10% of the evaluated instances. However, we did observe some frequent rules, which can be translated as: *if semantic A disagrees, then semantic B will disagree as well*. The rule action→details (*if action appears then details will appear*) is observed in 54.37% of the instances containing *action*; object enumeration →object color covers 17.48% of the instances containing *color*; finally, the reverse rule object color→object enumeration was observed in 11.65% of the instances containing *enumeration*.

Global evaluation We present global *query-agnostic* results for our metrics. Despite our metrics being more meaningful in local level, global evaluation is useful for model benchmarking.

With 134630 common concepts between all $(I_g, I_r)_i \in \mathcal{F}$, the ***average concept agreement (CA)*** value is 22.29%, meaning that on average almost the 1/4th of I_g concepts appear in I_r.

With 903987 non-common concepts between all (I_g, I_r), and 134630 common ones, we retrieve 627833 and 110839 WordNet synsets respectively. The *maximum weight matching* between non-common synsets results in 184747 maximum weight matchings, equivalent to the 29.43% of all non-common synsets. Averaging over matchings (WordNet path similarities) for all (I_g, I_r), provides the ***average non-common concept similarity (NCS)*** score of 0.122.

With 41244 concept sets of same multiplicity and 69595 of different multiplicities regarding matched concept categories for all (I_g, I_r), ***most common concept enumeration (CE)***=1 and ***average CE***=8.638 instances for concepts of the same category reveals that in most cases there are not major enumeration differences.

Focusing on the binary ***SD***, we set the area difference threshold T_D=100%, increasing ***size disagreement (SD)*** by 1 iff $\mathcal{D}_A \geq 1$ between two concept bounding box areas. Thus, ***average SD***=20.35% for all (I_g, I_r), indicating that around 1/5th of common objects have non-negligible size differences.

Our metrics in global level reveal some extra capabilities. Most lower-ranked instances contained erroneous annotations, allowing a *post-hoc dataset cleaning* step that could not have been automatically realized otherwise.

Global results regarding our proposed metrics for all the models are presented in Table 3. Moreover, we offer some additional insights:

- Object hit: total number of common objects found between ground truth - wrongly retrieved images (I_g, I_r) at top-1 position.
- Object miss: total number of ground truth objects not found in top-1 retrieved images.
- Matched % synsets: Percentage of ground truth synsets found in top-1 retrieved images out of all ground truth synsets.
- Average % object enumeration disagreement: percentage of objects having the wrong number of instances between ground truth and top-1 retrieved over all ground truth objects (both having right or wrong number of instances).

As observed, the various explainable metrics indicate different models as best/worst performers, revealing that fine-grained evaluation may disagree with traditional coarse evaluation, while providing some useful insights.

6.4 Adversarial re-ranking evaluation

Adjectives were substituted by their antonyms in the 30.93% of total queries, while color and size substitutions occurred in the 47.04% and 30.77% of queries respectively, producing $q_i^* \neq q_i$. Updated query representations resulted in re-ranking of instances; specifically, on average almost 70% of instances changed position in R^* comparing to R as presented in Table 4. *Adv.query* column refers to number of perturbed queries, while *Lower, Higher, Same* columns refer to the position change.

By qualitatively assessing adversarial failures, we observe that adversarially perturbed semantics are rather bypassed in favor of preserving object class. Even if this could imply representation robustness, on the other hand it can be attributed to language model biases towards object identities. In any case, existing ranking metrics cannot indicate potential biases, patterns and rules in the linguistic representations due to their opaque nature.

6.5 Non-explainable evaluation vulnerabilities

Overall, despite the re-arrangements of individual instances, R^* was only marginally altered in global level for any of the adversarial perturbations according to all *query-agnostic* metrics (Table 5). Therefore, either by providing meaningful and relevant queries or conceptually divergent ones, the response of a semantic similarity system is virtually the same. This invariance over non-minimal interventions generally questions the trustworthiness of

Name	CA ↑	NCS ↑	CE ↓	SD ↓	obj hit ↑	obj miss ↓	matched % synsets ↑	avg% enum disagr.↓
all distilroberta	21.75	0.12	9.14	20.02	136025	932261	29.43	2.40
all MiniLM L12	21.72	0.12	8.88	19.88	134188	920214	29.23	2.35
all MiniLM L6	21.90	0.12	8.91	19.53	135687	926717	29.27	2.39
all mpnet base	21.74	0.12	9.14	19.24	135278	926724	29.44	2.39
all roberta large	21.85	0.12	9.13	19.60	136720	935614	29.42	2.39
multi qa distilbert cos	21.84	**0.13**	9.06	16.97	140754	1007800	**30.60**	2.49
multi qa MiniLM L6 cos	21.93	**0.13**	8.69	18.86	**141399**	1008889	30.21	2.45
multi qa mpnet base cos	21.87	**0.13**	9.24	**16.69**	140455	1004153	30.54	2.52
nli distilroberta base	22.12	0.12	8.89	19.68	138882	943947	29.75	2.45
nq distilbert base	21.39	0.12	9.03	17.53	139608	1020776	30.27	2.38
paraphrase albert base	22.13	0.12	8.92	19.71	136867	927060	29.52	2.43
paraphrase albert small	**22.40**	0.12	8.89	19.48	136319	909759	29.69	2.47
paraphrase distilroberta	22.28	0.12	8.63	19.49	135679	911325	29.32	2.39
paraphrase MiniLM L12	22.28	0.12	8.69	20.05	136102	910631	29.23	2.39
paraphrase MiniLM L3	22.29	0.12	8.64	20.35	134630	**903987**	29.43	2.42
paraphrase MiniLM L6	22.07	0.12	8.61	19.37	134623	914535	29.46	2.37
paraphrase mpnet	22.26	0.12	8.71	19.46	136482	911716	29.08	2.39
paraphrase TinyBERT L6	22.15	0.12	8.55	19.67	134808	916573	29.17	2.36
stsb distilroberta base	22.09	0.12	8.91	19.99	136181	928388	29.81	2.45
stsb mpnet base	21.84	0.12	9.10	19.95	133670	916167	29.58	2.41
stsb roberta base	22.04	0.12	8.90	19.68	135544	925948	29.75	2.44
stsb roberta large	21.10	0.12	**8.29**	20.11	134706	961836	29.18	**2.24**
xlm distilroberta paraphrase	22.08	0.12	9.05	19.35	138934	951708	29.60	2.44

Table 3: Results from our proposed metrics plus some additional information occurring from our metrics per model

opaque ranking metrics, highlighting even more the need for explainable evaluation. Even then, query semantics are not taken into account, generally exposing query-agnostic evaluation against adversarial attacks: even if the best possible answer to a query is returned based on similarity measures, how can we ensure that it is good enough in terms of actual relevance? The query-corpus relevance can be easily and explicitly measured via their common concepts, an approach followed in *query-informed* evaluation. To this end, we conclude that *explainable* and, even better, *query-informed* metrics are necessary to ensure evaluation robustness.

Adv. Change	Adv. query	rank* Lower	rank* Higher	rank* Same
Antonym	10523	41.30	27.73	30.97
Color-all	16007	48.29	24.04	27.68
Color-in	16007	49.73	22.35	27.92
Size	10471	37.26	28.28	34.46

Table 4: Changes for all adversarial perturbations.

Adv. Change	Recall (%) @1	Recall (%) @10	MRR (%) @10	MRR (%) @*all*	Fails (%)
Original	15.31	39.48	22.11	23.42	84.69
Antonym	15.13	38.82	21.76	23.07	84.87
Color-all	14.52	38.16	21.15	22.48	85.48
Color-in	14.52	38.05	21.12	22.46	85.48
Size	15.19	39.32	21.97	23.29	84.81

Table 5: Rank results on adversarial queries.

7 Conclusion

In this work, we presented an evaluation framework for text-image retrieval and experimented with pretrained transformer-based semantic similarity models. Our approach achieved in capturing representation patterns and evaluation shortcomings of widely used metrics in local and global level. As future work, we aspire to extend our automated metrics to include attributes and spatial relationships between concepts, and produce adversarial re-rankings using verb antonyms, singular-plural sentence transformations and rare synonyms of salient concepts.

References

Joris Baan, Maartje ter Hoeve, Marlies van der Wees, Anne Schuth, and Maarten de Rijke. 2019. Do transformer attention heads provide transparency in abstractive summarization?

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. Bert: Pre-training of deep bidirectional transformers for language understanding. *ArXiv*, abs/1810.04805.

Christiane Fellbaum. 1998. Wordnet: An electronic lexical database.

Zvi Galil. 1986. Efficient algorithms for finding maximum matching in graphs. *ACM Comput. Surv.*, 18(1):23–38.

Dan Hendrycks, Xiaoyuan Liu, Eric Wallace, Adam Dziedzic, Rishabh Krishnan, and Dawn Song. 2020. Pretrained transformers improve out-of-distribution robustness.

Yu-Lun Hsieh, Minhao Cheng, Da-Cheng Juan, Wei Wei, Wen-Lian Hsu, and Cho-Jui Hsieh. 2019. On the robustness of self-attentive models. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 1520–1529, Florence, Italy. Association for Computational Linguistics.

Xiaoqi Jiao, Yichun Yin, Lifeng Shang, Xin Jiang, Xiao Chen, Linlin Li, Fang Wang, and Qun Liu. 2020. Tinybert: Distilling bert for natural language understanding.

Di Jin, Zhijing Jin, Joey Tianyi Zhou, and Peter Szolovits. 2019. Is bert really robust? a strong baseline for natural language attack on text classification and entailment.

R.M. Karp. 1978. An algorithm to solve the mxn assignment problem in expected time o (mn log n). Technical Report UCB/ERL M78/67, EECS Department, University of California, Berkeley.

Marvin Kaster, Wei Zhao, and Steffen Eger. 2021. Global explainability of bert-based evaluation metrics by disentangling along linguistic factors. *arXiv preprint arXiv:2110.04399*.

Ranjay Krishna, Yuke Zhu, Oliver Groth, Justin Johnson, Kenji Hata, Joshua Kravitz, Stephanie Chen, Yannis Kalantidis, Li-Jia Li, David A. Shamma, Michael S. Bernstein, and Fei-Fei Li. 2016. Visual genome: Connecting language and vision using crowdsourced dense image annotations.

H. W. Kuhn. 1955. The hungarian method for the assignment problem. *Naval Research Logistics Quarterly*, 2(1-2):83–97.

Zhenzhong Lan, Mingda Chen, Sebastian Goodman, Kevin Gimpel, Piyush Sharma, and Radu Soricut. 2020. Albert: A lite bert for self-supervised learning of language representations.

Christoph Leiter, Piyawat Lertvittayakumjorn, Marina Fomicheva, Wei Zhao, Yang Gao, and Steffen Eger. 2022. Towards explainable evaluation metrics for natural language generation.

Linyang Li, Ruotian Ma, Qipeng Guo, Xiangyang Xue, and Xipeng Qiu. 2020. Bert-attack: Adversarial attack against bert using bert.

Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C. Lawrence Zitnick. 2014. Microsoft coco: Common objects in context. In *Computer Vision – ECCV 2014*, pages 740–755, Cham. Springer International Publishing.

Pantelis Linardatos, Vasilis Papastefanopoulos, and Sotiris Kotsiantis. 2020. Explainable ai: a review of machine learning interpretability methods. *Entropy*, 23(1):18.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Roberta: A robustly optimized bert pretraining approach.

Simone Magnolini, Anna Feltracco, and Bernardo Magnini. 2016. Fbk-hlt-nlp at semeval-2016 task 2: A multitask, deep learning approach for interpretable semantic textual similarity. In *Proceedings of the 10th International Workshop on Semantic Evaluation (SemEval-2016)*, pages 783–789.

Christopher D. Manning, Prabhakar Raghavan, and Hinrich Schütze. 2008. *Introduction to Information Retrieval*. Cambridge University Press, USA.

Colin Raffel, Noam M. Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J. Liu. 2020. Exploring the limits of transfer learning with a unified text-to-text transformer. *ArXiv*, abs/1910.10683.

Nils Reimers and Iryna Gurevych. 2019. Sentence-bert: Sentence embeddings using siamese bert-networks. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics.

Raphael Rubino, Atsushi Fujita, and Benjamin Marie. 2021. Error identification for machine translation with metric embedding and attention. In *Proceedings of the 2nd Workshop on Evaluation and Comparison of NLP Systems*, pages 146–156.

Ananya B Sai, Tanay Dixit, Dev Yashpal Sheth, Sreyas Mohan, and Mitesh M Khapra. 2021. Perturbation checklists for evaluating nlg evaluation metrics. *arXiv preprint arXiv:2109.05771*.

Victor Sanh, Lysandre Debut, Julien Chaumond, and Thomas Wolf. 2020. Distilbert, a distilled version of bert: smaller, faster, cheaper and lighter.

Kaitao Song, Xu Tan, Tao Qin, Jianfeng Lu, and Tie-Yan Liu. 2020. Mpnet: Masked and permuted pre-training for language understanding.

Marcos Treviso, Nuno M Guerreiro, Ricardo Rei, and André FT Martins. 2021. Ist-unbabel 2021 submission for the explainable quality estimation shared task. In *Proceedings of the 2nd Workshop on Evaluation and Comparison of NLP Systems*, pages 133–145.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Ł ukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc.

Wenhui Wang, Furu Wei, Li Dong, Hangbo Bao, Nan Yang, and Ming Zhou. 2020. Minilm: Deep self-attention distillation for task-agnostic compression of pre-trained transformers.

Peter Young, Alice Lai, Micah Hodosh, and Julia Hockenmaier. 2014. From image descriptions to visual denotations: New similarity metrics for semantic inference over event descriptions. *Transactions of the Association for Computational Linguistics*, 2:67–78.

Weizhe Yuan, Graham Neubig, and Pengfei Liu. 2021. Bartscore: Evaluating generated text as text generation. *Advances in Neural Information Processing Systems*, 34.

Jingqing Zhang, Yao Zhao, Mohammad Saleh, and Peter J. Liu. 2020a. Pegasus: Pre-training with extracted gap-sentences for abstractive summarization. *ArXiv*, abs/1912.08777.

Wei Emma Zhang, Quan Z. Sheng, Ahoud Alhazmi, and Chenliang Li. 2020b. Adversarial attacks on deep-learning models in natural language processing: A survey. *ACM Trans. Intell. Syst. Technol.*, 11(3).

A Appendix

A.1 Qualitative results of retrieved vs ground truth pairs

The following Figures 5, 6, 7, 8 demonstrate some interesting results regarding retrieved images and their ground truth matchings with respect to a given query. Images to the left correspond to the retrieved image I_{r_i}, while images to the right denote the ground truth image I_{g_i} with respect to a query q_i appearing in the caption. The caption also mentions the failure category the images belong, according to human evaluation results.

A.1.1 Human evaluation details

We present some distributions regarding human evaluation experiments. Figure 9 regards the rating distribution according to our evaluators' perception of ground truth-retrieved image relevance with respect to the given query. Figure 10 presents the number of failed semantics categories per image.

Figure 5: ***Successful alternative.*** Many people are relaxing under their umbrellas on the beach.

Figure 6: ***Object color.*** A vase sitting on a table with white flowers in it.

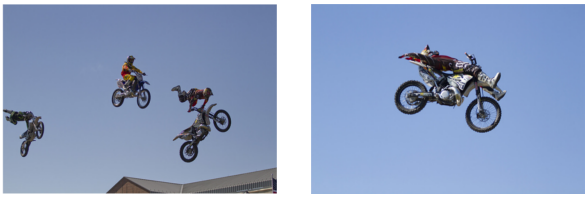

Figure 7: ***Object enumeration.*** A dirt bike rider performing a stunt while in the air.

Figure 8: ***Detailed semantics.*** A cat playing with a shoe in a grassy field.

A.2 Analyzing adversarial position changes

Antonym-based queries Substituting adjectives with their antonyms was applicable on 10523 queries which resulted in updated embedding representations: the cosine similarity $cos(u_i^q, u_i^{q*}) < 1$. By exclusively considering adversarial instances with updated representations $u_i^q \neq u_i^{q*}$, we observed that 4346 instances (41.30%) were ranked lower than the original ones, 2918 instances (27.73%) were ranked higher, and 3259 instances

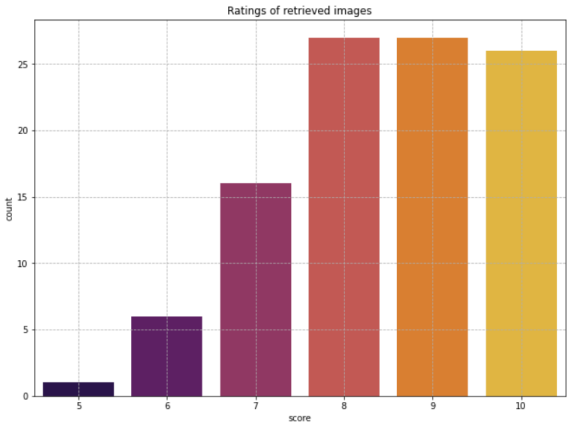

Figure 9: Human evaluation ratings (1-10).

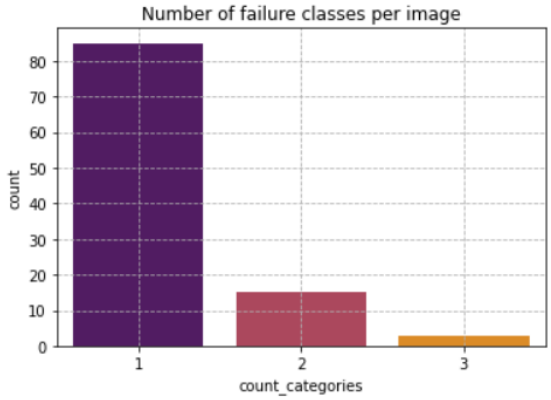

Figure 10: Number of marked disagreeing semantics per I_{g_i}, I_{r_i} pair for all evaluated image pairs in $\mathcal{F}$.

(30.97%) remained in the same position.

Size-based queries 10471 queries of the dataset where perturbed with respect to size-oriented words. Consequently, 3902 instances (37.26%) were ranked lower than the original ones, 2961 instances (28.28%) were ranked higher, and 3608 instances (34.46%) remained in the same position.

Color-based queries 16007 queries that contain colors were adversarially perturbed. Regarding the *color-in*) experiment and by considering adversarial instances with updated representations, 7960 instances (49.73%) were ranked lower, 3578 instances (22.35%) were ranked higher, and 4469 instances (27.92%) remained in the same position. In the *color-all* experiment, we observe that 7729 instances (48.29%) were ranked lower than the original ones, 3848 instances (24.04%) were ranked higher, and 4430 instances (27.68%) remained in the same position.

Those results are summarized in Table 4.

A.3 Ranking results

In this section we present all ranking experiments conducted in this work. Those include all 3 datasets (Flickr8K, Flickr30K and VG ∩ COCO) and representation choices (summarizing with T5 (Raffel et al., 2020) or pegasus (Zhang et al., 2020a) abstractive summarizers before using an SBERT model, or embed corpus sentences intependently using an SBERT model and then calculate the averaged embedding representation).

Tables 6, 7, 8 refer to Flickr8K experiments; Tables 9, 10 refer to Flickr30k experiments; finally, 11, 12, 13 refer to VG ∩ COCO experiments.

Name	Recall(%)↑			MRR(%)↑			Median	Fails
	@1	@5	@10	@5	@10	@*all*	Rank ↓	(%) ↓
all MiniLM L12	0.4486	0.6670	0.7477	0.5307	0.5418	0.5504	2	55.14
all MiniLM L6	0.4354	0.6568	0.7392	0.5183	0.5294	0.5380	2	56.46
all distilroberta	0.4699	0.6972	0.7754	0.5557	0.5662	0.5744	2	53.01
all mpnet base	0.4664	0.6872	0.7669	0.5503	0.5610	0.5692	2	53.36
all roberta large	0.4858	0.7081	0.7900	0.5700	0.5811	0.5886	2	51.42
multi qa MiniLM L6 cos	0.3662	0.5687	0.6541	0.4411	0.4526	0.4628	3	63.38
multi qa distilbert cos	0.3797	0.5874	0.6738	0.4583	0.4699	0.4795	3	62.03
multi qa mpnet base cos	0.3990	0.6097	0.6990	0.4784	0.4904	0.4996	3	60.10
nli distilroberta base	0.4011	0.6256	0.7152	0.4864	0.4984	0.5076	2	59.89
nq distilbert base	0.3260	0.5363	0.6187	0.4037	0.4149	0.4255	4	67.40
paraphrase MiniLM L12	0.5289	0.7602	0.8338	0.6173	0.6273	0.6337	**1**	47.11
paraphrase MiniLM L3	0.5064	0.7337	0.8183	0.5933	0.6047	0.6115	**1**	49.36
paraphrase MiniLM L6	0.5138	0.7497	0.8259	0.6044	0.6147	0.6212	**1**	48.62
paraphrase TinyBERT L6	**0.5685**	**0.8060**	**0.8758**	**0.6604**	**0.6700**	**0.6749**	**1**	**43.15**
paraphrase albert base	0.4776	0.7167	0.7957	0.5684	0.5789	0.5864	2	52.24
paraphrase distilroberta	0.5296	0.7643	0.8399	0.6197	0.6298	0.6361	**1**	47.04
paraphrase mpnet	0.4936	0.7318	0.8118	0.5846	0.5953	0.6022	2	50.64
stsb distilroberta base	0.4106	0.6219	0.7044	0.4902	0.5013	0.5109	2	58.94
stsb mpnet base	0.4334	0.6517	0.7359	0.5159	0.5272	0.5356	2	56.66
stsb roberta base	0.4206	0.6354	0.7201	0.5017	0.5131	0.5224	2	57.94
stsb roberta large	0.3678	0.5694	0.6575	0.4426	0.4545	0.4643	3	63.22
xlm distilroberta paraphrase	0.4138	0.6382	0.7255	0.4979	0.5096	0.5187	2	58.62

Table 6: Rank results for Flickr8k. Bold entries indicate best results.

Name	Recall(%)↑			MRR(%)↑			Median	Fails
	@1	@5	@10	@5	@10	@*all*	Rank ↓	(%) ↓
all MiniLM L12	0.3139	0.5285	0.6213	0.3932	0.4056	0.4169	5	68.61
all MiniLM L6	0.3100	0.5193	0.6143	0.3870	0.3996	0.4106	5	69.00
all distilroberta	0.3421	0.5717	0.6600	0.4281	0.4400	0.4508	4	65.79
all mpnet base	0.3242	0.5410	0.6347	0.4051	0.4176	0.4285	4	67.58
all roberta large	0.3548	0.5818	0.6787	0.4391	0.4521	0.4624	3	64.52
multi qa MiniLM L6 cos	0.2262	0.3965	0.4789	0.2887	0.2996	0.3115	12	77.38
multi qa distilbert cos	0.2351	0.4092	0.4928	0.2993	0.3104	0.3220	11	76.49
multi qa mpnet base cos	0.2406	0.4306	0.5197	0.3103	0.3220	0.3337	9	75.94
nli distilroberta base	0.2856	0.4928	0.5852	0.3626	0.3748	0.3864	6	71.44
nq distilbert base	0.1646	0.3058	0.3766	0.2165	0.2258	0.2376	28	83.54
paraphrase MiniLM L12	0.4071	0.6581	0.7492	0.5008	0.5131	0.5222	**2**	59.29
paraphrase MiniLM L3	0.3864	0.6281	0.7266	0.4772	0.4904	0.4999	3	61.36
paraphrase MiniLM L6	0.3960	0.6470	0.7391	0.4902	0.5024	0.5119	**2**	60.40
paraphrase TinyBERT L6	**0.4390**	**0.6995**	**0.7941**	**0.5376**	**0.5505**	**0.5584**	**2**	**56.10**
paraphrase albert base	0.3618	0.6014	0.6937	0.4517	0.4642	0.4745	3	63.82
paraphrase albert small	0.4110	0.6699	0.7594	0.5083	0.5207	0.5300	**2**	58.90
paraphrase distilroberta	0.4085	0.6667	0.7613	0.5042	0.5169	0.5253	**2**	59.15
paraphrase mpnet	0.3856	0.6233	0.7186	0.4752	0.4881	0.4976	3	61.44
stsb distilroberta base	0.2886	0.4881	0.5746	0.3625	0.3741	0.3855	6	71.14
stsb mpnet base	0.3105	0.5185	0.6043	0.3879	0.3995	0.4106	5	68.95
stsb roberta base	0.2971	0.5008	0.5882	0.3725	0.3843	0.3957	5	70.29
stsb roberta large	0.2488	0.4173	0.5044	0.3110	0.3226	0.3346	10	75.12
xlm distilroberta paraphrase	0.2814	0.4899	0.5857	0.3579	0.3708	0.3821	6	71.86

Table 7: Rank results for Flickr8k - T5 (Raffel et al., 2020) summarizer. Bold entries indicate best results.

Name	Recall(%)↑			MRR(%)↑			Median	Fails
	@1	@5	@10	@5	@10	@*all*	Rank ↓	(%) ↓
all MiniLM L12	0.2396	0.4336	0.5203	0.3097	0.3210	0.3328	9	76.04
all MiniLM L6	0.2330	0.4190	0.5135	0.3016	0.3133	0.3267	9	76.70
all distilroberta	0.2605	0.4645	0.5556	0.3359	0.3475	0.3602	7	73.95
all mpnet base	0.2519	0.4479	0.5429	0.3239	0.3369	0.3498	8	74.81
all roberta large	0.2686	0.4789	0.5695	0.3457	0.3576	0.3708	6	73.14
multi qa MiniLM L6 cos	0.1754	0.3163	0.3908	0.2258	0.2362	0.2482	25	82.46
multi qa distilbert cos	0.1856	0.3353	0.4126	0.2402	0.2501	0.2624	21	81.44
multi qa mpnet base cos	0.2008	0.3569	0.4363	0.2572	0.2681	0.2794	18	79.92
nli distilroberta base	0.2221	0.4070	0.4964	0.2888	0.3007	0.3143	11	77.79
nq distilbert base	0.1456	0.2712	0.3359	0.1905	0.1993	0.2110	40	85.44
paraphrase MiniLM L12	0.3061	0.5358	0.6390	0.3909	0.4041	0.4156	**4**	69.39
paraphrase MiniLM L3	0.2876	0.5024	0.6038	0.3663	0.3800	0.3930	5	71.24
paraphrase MiniLM L6	0.2960	0.5234	0.6233	0.3804	0.3928	0.4055	5	70.40
paraphrase TinyBERT L6	**0.3223**	**0.5667**	**0.6696**	**0.4126**	**0.4262**	**0.4381**	**4**	**67.77**
paraphrase albert base	0.2830	0.4909	0.5925	0.3596	0.3731	0.3856	6	71.70
paraphrase albert small	0.3071	0.5387	0.6398	0.3922	0.4058	0.4185	**4**	69.29
paraphrase distilroberta	0.3082	0.5404	0.6417	0.3938	0.4071	0.4185	**4**	69.18
paraphrase mpnet	0.2903	0.5055	0.6077	0.3701	0.3840	0.3964	5	70.97
stsb distilroberta base	0.2249	0.4028	0.4875	0.2891	0.3006	0.3132	12	77.51
stsb mpnet base	0.2369	0.4217	0.5086	0.3049	0.3161	0.3279	10	76.31
stsb roberta base	0.2298	0.4101	0.4989	0.2948	0.3063	0.3194	11	77.02
stsb roberta large	0.1884	0.3359	0.4144	0.2420	0.2524	0.2648	20	81.16
xlm distilroberta paraphrase	0.2231	0.4032	0.4967	0.2880	0.3008	0.3142	11	77.69

Table 8: Rank results for Flickr8k - Pegasus (Zhang et al., 2020a) summarizer. Bold entries indicate best results.

Name	Recall(%)↑			MRR(%)↑			Median	Fails
	@1	@5	@10	@5	@10	@*all*	Rank ↓	(%) ↓
all MiniLM L12	0.3076	0.5155	0.6063	0.3848	0.3969	0.4075	5	69.24
all MiniLM L6	0.2961	0.5021	0.5924	0.3727	0.3849	0.3956	5	70.39
all distilroberta	0.3276	0.5439	0.6357	0.4082	0.4204	0.4310	4	67.24
all mpnet base	0.3233	0.5372	0.6304	0.4030	0.4155	0.4259	4	67.67
all roberta large	0.3381	0.5633	0.6527	0.4223	0.4343	0.4445	4	66.19
multi qa MiniLM L6 cos	0.2397	0.4205	0.5042	0.3058	0.3169	0.3281	10	66.19
multi qa distilbert cos	0.2530	0.4379	0.5252	0.3210	0.3327	0.3440	9	74.70
multi qa mpnet base cos	0.2730	0.4683	0.5555	0.3453	0.3570	0.3681	7	72.70
nli distilroberta base	0.2770	0.4780	0.5718	0.3512	0.3638	0.3750	6	72.30
nq distilbert base	0.2131	0.3841	0.4653	0.2760	0.2868	0.2983	14	78.69
paraphrase MiniLM L12	0.3687	0.6148	0.7074	0.4606	0.4731	0.4831	3	63.13
paraphrase MiniLM L3	0.3437	0.5826	0.6764	0.4328	0.4453	0.4559	3	65.63
paraphrase MiniLM L6	0.3526	0.5938	0.6888	0.4425	0.4553	0.4657	3	64.74
paraphrase TinyBERT L6	**0.4199**	**0.6844**	**0.7850**	**0.5198**	**0.5333**	**0.5421**	**2**	**58.01**
paraphrase albert base	0.3294	0.5548	0.6482	0.4132	0.4257	0.4365	4	67.06
paraphrase albert small	0.3831	0.6416	0.7432	0.4799	0.4935	0.5032	3	61.69
paraphrase distilroberta	0.3796	0.6224	0.7230	0.4705	0.4840	0.4937	3	62.04
paraphrase mpnet	0.3476	0.5804	0.6730	0.4341	0.4466	0.4570	3	65.24
stsb distilroberta base	0.2765	0.4716	0.5626	0.3487	0.3610	0.3724	7	72.35
stsb mpnet base	0.2935	0.4958	0.5846	0.3680	0.3799	0.3909	6	70.65
stsb roberta base	0.2864	0.4888	0.5757	0.3612	0.3729	0.3840	6	71.36
stsb roberta large	0.2410	0.4207	0.5037	0.3067	0.3178	0.3292	10	75.90
xlm distilroberta paraphrase	0.2816	0.4832	0.5732	0.3559	0.3679	0.3790	6	71.84

Table 9: Rank results Flickr30k. Bold entries indicate best results.

Name	Recall(%)↑			MRR(%)↑			Median	Fails
	@1	@5	@10	@5	@10	@*all*	Rank ↓	(%) ↓
all MiniLM L12	0.2068	0.3845	0.4710	0.2721	0.2836	0.2955	13	79.32
all MiniLM L6	0.2012	0.3739	0.4595	0.2645	0.2758	0.2877	14	79.88
all distilroberta	0.2242	0.4159	0.5083	0.2948	0.3072	0.3193	10	77.58
all mpnet base	0.2138	0.3975	0.4837	0.2812	0.2927	0.3047	12	78.62
all roberta large	0.2373	0.4321	0.5264	0.3091	0.3217	0.3337	9	76.27
multi qa MiniLM L6 cos	0.1380	0.2665	0.3357	0.1847	0.1939	0.2051	42	86.20
multi qa distilbert cos	0.1443	0.2820	0.3552	0.1942	0.2039	0.2152	35	85.57
multi qa mpnet base cos	0.1515	0.2976	0.3733	0.2049	0.2150	0.2267	29	84.85
nli distilroberta base	0.1856	0.3556	0.4401	0.2477	0.2590	0.2712	16	81.44
nq distilbert base	0.0951	0.1963	0.2530	0.1317	0.1392	0.1493	100	90.49
paraphrase MiniLM L12	0.2740	0.4970	0.5988	0.3560	0.3696	0.3817	6	72.60
paraphrase MiniLM L3	0.2493	0.4630	0.5649	0.3274	0.3410	0.3534	7	75.07
paraphrase MiniLM L6	0.2644	0.4810	0.5825	0.3441	0.3577	0.3700	6	73.56
paraphrase TinyBERT L6	**0.3032**	**0.5555**	**0.6688**	**0.3958**	**0.4109**	**0.4230**	**4**	**69.68**
paraphrase albert base	0.2400	0.4428	0.5389	0.3142	0.3271	0.3394	8	76.00
paraphrase albert small	0.2759	0.5140	0.6218	0.3627	0.3771	0.3897	5	72.41
paraphrase distilroberta	0.2803	0.5082	0.6119	0.3639	0.3777	0.3897	5	71.97
paraphrase mpnet	0.2584	0.4687	0.5664	0.3351	0.3482	0.3603	7	74.16
stsb distilroberta base	0.1800	0.3424	0.4263	0.2395	0.2507	0.2627	18	82.00
stsb mpnet base	0.1965	0.3724	0.4557	0.2605	0.2717	0.2836	15	80.35
stsb roberta base	0.1904	0.3547	0.4389	0.2503	0.2615	0.2735	17	80.96
stsb roberta large	0.1456	0.2811	0.3543	0.1945	0.2042	0.2159	34	85.44
xlm distilroberta paraphrase	0.1792	0.3467	0.4286	0.2403	0.2513	0.2635	18	82.08

Table 10: Rank results Flickr30k - T5 (Raffel et al., 2020) summarizer. Bold entries indicate best results.

Name	Recall(%)↑			MRR(%)↑			Median	Fails
	@1	@5	@10	@5	@10	@*all*	Rank ↓	(%) ↓
all MiniLM L12	13.31	26.93	34.31	18.18	19.16	20.45	34	86.69
all MiniLM L6	13.07	26.80	34.26	18.02	19.02	20.29	35	86.93
all distilroberta	12.20	26.10	33.58	17.22	18.22	19.55	35	87.80
all mpnet base	12.64	26.12	33.82	17.45	18.47	19.78	35	87.36
all roberta large	12.01	25.63	33.42	16.87	17.91	19.24	35	87.99
multi qa MiniLM L6 cos	9.29	20.10	26.77	13.14	14.03	15.29	60	90.71
multi qa distilbert cos	9.22	20.35	27.04	13.16	14.04	15.32	57	90.78
multi qa mpnet base cos	9.53	21.03	27.81	13.63	14.52	15.79	55	90.47
nli distilroberta base	11.73	24.94	32.33	16.48	17.46	18.75	39	88.27
nq distilbert base	8.03	17.95	23.95	11.53	12.32	13.54	76	91.97
paraphrase MiniLM L12	14.24	29.89	38.34	19.89	21.02	22.37	24	85.76
paraphrase MiniLM L3	**15.31**	**30.92**	**39.48**	**20.97**	**22.11**	**23.42**	23	**84.69**
paraphrase MiniLM L6	14.39	30.01	38.51	19.96	21.10	22.46	24	85.61
paraphrase TinyBERT L6	14.55	30.38	39.12	20.27	21.43	22.82	**22**	85.45
paraphrase albert base	13.12	27.83	36.11	18.43	19.54	20.91	28	86.88
paraphrase albert small	14.56	30.25	39.04	20.23	21.40	22.75	23	85.44
paraphrase distilroberta	14.51	30.39	38.91	20.23	21.36	22.74	**22**	85.49
paraphrase mpnet	13.99	28.99	37.40	19.39	20.50	21.84	26	86.01
stsb distilroberta base	13.41	27.04	34.28	18.32	19.28	20.55	35	86.59
stsb mpnet base	14.05	28.26	35.93	19.23	20.24	21.49	32	85.95
stsb roberta base	13.69	27.38	34.79	18.67	19.65	20.92	34	86.31
stsb roberta large	10.32	22.13	28.71	14.60	15.47	16.73	53	89.68
xlm distilroberta paraphrase	11.59	24.62	31.87	16.26	17.23	18.52	40	88.41

Table 11: Rank results for VG ∩ COCO. Bold entries indicate best results.

Name	Recall(%)↑			MRR(%)↑			Median	Fails
	@1	@5	@10	@5	@10	@*all*	Rank ↓	(%) ↓
all MiniLM L12	0.0989	0.2072	0.2684	0.1378	0.1459	0.1575	68	90.11
all MiniLM L6	0.0923	0.2004	0.2601	0.1313	0.1393	0.1509	71	90.77
all distilroberta	0.1034	0.2184	0.2834	0.1445	0.1532	0.1654	56	89.66
all mpnet base	0.0906	0.1969	0.2608	0.1286	0.1370	0.1488	69	90.94
all roberta large	0.1125	0.2340	0.2994	0.1563	0.1650	0.1769	52	88.75
multi qa MiniLM L6 cos	0.0496	0.1132	0.1558	0.0719	0.0776	0.0872	189	95.04
multi qa distilbert cos	0.0519	0.1172	0.1580	0.0748	0.0802	0.0901	181	94.81
multi qa mpnet base cos	0.0509	0.1233	0.1718	0.0764	0.0827	0.0931	148	94.91
nli distilroberta base	0.0914	0.1931	0.2502	0.1281	0.1357	0.1470	82	90.86
nq distilbert base	0.0364	0.0914	0.1285	0.0560	0.0609	0.0698	249	96.36
paraphrase MiniLM L12	**0.1365**	**0.2794**	**0.3579**	**0.1882**	**0.1987**	**0.2113**	**31**	**86.35**
paraphrase MiniLM L3	0.1165	0.2497	0.3239	0.1645	0.1743	0.1872	38	88.35
paraphrase MiniLM L6	0.1258	0.2643	0.3386	0.1756	0.1855	0.1983	35	87.42
paraphrase TinyBERT L6	0.1272	0.2652	0.3446	0.1767	0.1873	0.2001	33	87.28
paraphrase albert base	0.1215	0.2534	0.3263	0.1687	0.1784	0.1909	40	87.85
paraphrase albert small	0.1233	0.2572	0.3331	0.1715	0.1815	0.1942	37	87.67
paraphrase distilroberta	0.1274	0.2621	0.3355	0.1760	0.1857	0.1984	36	87.26
paraphrase mpnet	0.1362	0.2778	0.3552	0.1873	0.1976	0.2103	**31**	86.38
stsb distilroberta base	0.0891	0.1900	0.2450	0.1250	0.1323	0.1436	85	91.09
stsb mpnet base	0.0993	0.2094	0.2715	0.1389	0.1471	0.1589	66	90.07
stsb roberta base	0.0962	0.2019	0.2618	0.1339	0.1419	0.1534	72	90.38
stsb roberta large	0.0662	0.1506	0.2035	0.0959	0.1029	0.1141	109	93.38
xlm distilroberta paraphrase	0.0749	0.1735	0.2313	0.1102	0.1178	0.1292	90	92.51

Table 12: Rank results for VG ∩ COCO - T5 (Raffel et al., 2020) summarizer. Bold entries indicate best results.

Name	Recall(%)↑			MRR(%)↑			Median	Fails
	@1	@5	@10	@5	@10	@*all*	Rank ↓	(%) ↓
all MiniLM L12	0.0896	0.1898	0.2466	0.1257	0.1332	0.1439	91	91.04
all MiniLM L6	0.0847	0.1829	0.2373	0.1199	0.1271	0.1380	95	91.53
all distilroberta	0.0946	0.2050	0.2660	0.1342	0.1423	0.1537	73	90.54
all mpnet base	0.0903	0.1918	0.2489	0.1267	0.1344	0.1452	86	90.97
all roberta large	0.0997	0.2070	0.2669	0.1383	0.1462	0.1574	74	90.03
multi qa MiniLM L6 cos	0.0490	0.1106	0.1541	0.0706	0.0764	0.0855	211	95.10
multi qa distilbert cos	0.0494	0.1136	0.1555	0.0722	0.0777	0.0869	208	95.06
multi qa mpnet base cos	0.0497	0.1145	0.1573	0.0722	0.0778	0.0874	191	95.03
nli distilroberta base	0.0755	0.1656	0.2194	0.1074	0.1145	0.1250	113	92.45
nq distilbert base	0.0436	0.1000	0.1390	0.0636	0.0688	0.0776	251	95.64
paraphrase MiniLM L12	**0.1154**	**0.2395**	**0.3076**	**0.1600**	**0.1690**	**0.1808**	50	**88.46**
paraphrase MiniLM L3	0.1030	0.2211	0.2870	0.1453	0.1540	0.1659	58	89.70
paraphrase MiniLM L6	0.1103	0.2278	0.2964	0.1526	0.1617	0.1735	54	88.97
paraphrase TinyBERT L6	0.1141	0.2373	0.3063	0.1583	0.1675	0.1796	**48**	88.59
paraphrase albert base	0.1003	0.2122	0.2757	0.1405	0.1490	0.1608	62	89.97
paraphrase albert small	0.1086	0.2269	0.2934	0.1512	0.1601	0.1722	53	89.14
paraphrase distilroberta	0.1140	0.2345	0.3026	0.1573	0.1664	0.1785	49	88.60
paraphrase mpnet	0.1099	0.2304	0.2973	0.1536	0.1625	0.1744	53	89.01
stsb distilroberta base	0.0751	0.1612	0.2145	0.1061	0.1131	0.1236	114	92.49
stsb roberta base	0.0783	0.1676	0.2211	0.1104	0.1174	0.1281	110	92.17
stsb roberta large	0.0586	0.1359	0.1844	0.0857	0.0921	0.1022	145	94.14
xlm distilroberta paraphrase	0.0706	0.1563	0.2066	0.1009	0.1075	0.1178	127	92.94

Table 13: Rank results for VG ∩ COCO - Pegasus (Zhang et al., 2020a) summarizer. Bold entries indicate best results.

Establishing Annotation Quality in Multi-Label Annotations

Marian Marchal and **Merel Scholman** and **Frances Yung** and **Vera Demberg**
Language Science and Technology / Saarland University
{marchal, m.c.j.scholman, frances, vera}@coli.uni-saarland.de

Abstract

In many linguistic fields requiring annotated data, multiple interpretations of a single item are possible. Multi-label annotations more accurately reflect this possibility. However, allowing for multi-label annotations also affects the chance that two coders agree with each other. Calculating inter-coder agreement for multi-label datasets is therefore not trivial. In the current contribution, we evaluate different metrics for calculating agreement on multi-label annotations: agreement on the intersection of annotated labels, an augmented version of Cohen's Kappa, and precision, recall and F1. We propose a bootstrapping method to obtain chance agreement for each measure, which allows us to obtain an adjusted agreement coefficient that is more interpretable. We demonstrate how various measures affect estimates of agreement on simulated datasets and present a case study of discourse relation annotations. We also show how the proportion of double labels, and the entropy of the label distribution, influences the measures outlined above and how a bootstrapped adjusted agreement can make agreement measures more comparable across datasets in multi-label scenarios.

1 Introduction

Annotation efforts have long been characterized by the (implicit or explicit) assumption that there is a single true interpretation for every item, and all other interpretations are incorrect. This has become even more pervasive with the increasing usage of annotations as input for classifiers and other downstream computational tasks. However, the single-truth assumption has been challenged in recent years (e.g., Aroyo and Welty, 2015; Basile et al., 2021), based on counterexamples from many different fields showing that two coders can disagree on an annotation and still both be right, due to the subjectivity and complexity of many tasks. We here focus on the field of discourse relation annotation as a case study, although the insights could be applied to different types of linguistic research (Amidei et al., 2019; Uma et al., 2021).

Establishing inter-annotator agreement in scenario's where multiple labels are possible is challenging. In cases where multiple interpretations are possible but coders are restricted to annotate only a single label, it is unclear whether disagreement reflects incorrect interpretations (coder error or issues with the coding scheme or training) or item ambiguity. Given an instance where two interpretations are equally likely, assuming that at least one coder inferred both possible readings but only a single label can be annotated, the chance that the coders agree would be only 50%. This does not reflect that there is actually high agreement on the labels.

Allowing for multi-label coding can result in more accurate annotations as well as higher agreement, and with that most likely higher reliability of a final label. Although two sets of multi-labels can be compared by calculating the difference between their distributions (e.g. comparing the probability distribution of a classifier to the distribution of crowdsourced labels, Fornaciari et al., 2021), the distributions and ranks of the labels are not necessarily relevant or accessible (e.g. expert annotations where multiple labels are allowed).

We here assume a scenario where unranked multiple labels are allowed, regardless of whether more labels are possible and aim to properly estimate agreement. We will therefore focus on estimating the reliability of a single final label per item, while also discussing other scenario's. Traditional agreement statistics, such as Cohen's Kappa (Cohen, 1960) will not be suitable for evaluating the reliability of data that allow for multi-label annotations – at least not without adjustments. It should be taken into account that multi-label coding also inflates the chance agreement: by providing more labels, there is a higher chance that at least one of those labels overlaps with the annotations from another

Proceedings of the 29th International Conference on Computational Linguistics, pages 3659–3668
October 12–17, 2022.

coder.

The current paper will analyze several ways to determine the reliability of multi-label annotations. We argue that chance agreement should always be taken into account for any agreement measure to make it comparable across datasets. The contributions of this paper are: (i) we propose a bootstrapping method to estimate the expected agreement (see Section 4), and (ii) we compare agreement on various measures for simulated datasets with different parameters, as well as for a real-life dataset (Section 5 and 6). We make available a reproducible script of the implementations of the measures and the calculations.[1]

We use the following terminology throughout the paper. An *item* (i) is an instance which is annotated. It can be annotated with one or more *labels*, which together make up one *annotation*. *Categories* (k) refer to the options that *coders* (c) have to assign an item to.

2 Multiple labels: Ambiguity and uncertainty

There are several scenarios in which multiple labels are possibly desirable. A first one is a scenario in which annotators are uncertain which of multiple categories are correct, caused by a lack of knowledge or information (Beck et al., 2020). Allowing them to provide multiple labels would then reflect a probabilistic representation of the target label. These probabilistic labels also occur often in crowd-sourced annotations, where workers might lack the knowledge to select the correct label.

Secondly, multiple labels might reflect true ambiguity. These are cases where more than one interpretation is possible. To illustrate, consider the discourse relation in Example (2): both a SPECIFICATION relation (which could be expressed by inserting the cue phrase *more specifically* between the two sentences), and a MANNER relation (which could be expressed by *to do so,*) interpretation are valid.

(1) Ryan was decorating the Christmas tree. He was hanging the baubles.

Ambiguity has been argued to be an "inherent property of natural language" and an important source of disagreement in language annotation (Beck et al., 2020). Plank et al. (2014) show that the vast majority of disagreements between annotators arise because multiple labels are valid and recommend allowing for such disagreements, rather than focusing on inter-annotator agreement. The present contribution shows that it is possible to establish inter-annotator agreement while taking ambiguity into account by allowing for multiple labels. More specifically, the focus of the present paper is on establishing a reliability measure for ambiguous cases where multiple labels are valid, but we will also address how reliability measures can reflect these different multi-label scenarios.

3 Reliability measures

Reliability is the extent to which different coders arrive at the same interpretations of items. Reliability can be measured by calculating the inter-coder agreement using an agreement coefficient: a numerical index of the extent of agreement between the coders. However, the goal of obtaining reliable data is not merely to have data on which two coders agree (coders might be wrong or biased, after all), but to have annotated labels that reflect the true meaning of the items. It is important to note that this validity, despite being the goal of annotation efforts, is not captured by agreement coefficients.

Agreement coefficients usually consist of two components: observed agreement (A_o) and expected agreement (A_e). Together, these can be used to calculate an adjusted agreement, i.e. an agreement coefficient (A_c):

$$A_c = \frac{A_o - A_e}{1 - A_e} \quad (1)$$

Observed agreement is taken to be 1 when two coders assign an item to the same category and 0 when the item is assigned to different categories. Observed agreement does not take into account chance agreement, which occurs when one or both coders rate an item randomly. In order to get a reliable index of the extent of agreement between coders, observed agreement therefore has to be adjusted for the proportion of agreement that is expected to occur by chance. The crucial difference between various inter-coder agreement measures often lies in the way in which they estimate this expected agreement (see Artstein and Poesio, 2008 for a detailed overview).

One of the most frequently used inter-coder agreement measures is Cohen's Kappa (κ) (Co-

[1] https://osf.io/f5v4p/?view_only=4962b8ae4398466c88e620c27302e0c5

hen, 1960). When each data point in a corpus is assigned a single label, calculating chance agreement, and κ, is straightforward. More specifically, Cohen's κ calculates the probability of each label being selected by each coder independently:

$$A_e = \sum_{k \in K} P(k \mid c_1) \cdot P(k \mid c_2) \quad (2)$$

However, traditional kappa is not applicable to multi-label scenarios. One of the simplest solutions would be to treat each multiple-label annotation as a distinct label, but this inflates the number of categories in the coding scheme, which negatively affects κ. The traditional κ measure would therefore need to be adapted to make it suitable for multi-label annotations.

3.1 Soft-match agreement

One solution to adapt the traditional kappa measure is to calculate agreement using the intersection of agreed-upon labels – that is, the label that occurs in the annotation of both coders. For example, in Table 1, this leads to the observed match agreement being 1 for each item.

Taking the intersection agreement as the final annotation increases the probability that the obtained label is part of the set of true labels for that item, because both annotators agree on that label. Discarding the additional, non-overlapping labels means that information regarding that item is lost, but for many tasks, such as analyses for psycholinguistic experiments, or training certain classifiers, researchers only use a single label. In such scenarios, discarding the additional labels does not negatively impact the results.

To calculate `soft-match` agreement, multi-label annotations of the coders are replaced with the intersecting label (e.g., as done in Crible and Degand, 2019). In cases where there is no intersection or two intersecting labels between two multi-label annotations, a single label is sampled in order to be able to estimate expected agreement. For example, in item 1 in Table 1, κ will be calculated after removing label B from c_2 in item 1 and sampling either A or B for both coders for item 3. κ is then calculated on this adjusted dataset as usual, using the formulas above. This type of agreement can also be considered the *oracle agreement*, as it is the highest agreement that coders could have reached if they had selected only the overlapping single label.

Table 1: Example items with observed agreement for soft-match (S), augmented kappa (A), recall (R, c_1 wrt c_2) precision (P, c_1 wrt c_2) and F1.

	annotations		**observed agreement**				
item	c_1	c_2	S	A	R	P	F1
1	A	A;B	1	.50	1	.50	.67
2	A;B	B;C	1	.25	.50	.50	.50
3	A;B	A;B	1	.50	1	1	1

However, this method of calculating kappa on the intersection is problematic because it does not take into account that chance agreement on an intersection is higher when multiple labels are provided, over-estimating κ. In the most extreme case, where one coder would assign all categories to a single item, both observed and expected agreement will be 1 in reality. However, expected agreement using this `soft-match` agreement is much lower, thus inflating κ. Kappa on the `soft-match` agreement can therefore be misleading.

3.2 Augmented kappa

Rosenberg and Binkowski (2004) proposed an augmented version of κ, referred to here as `augmented kappa`, to measure corpus reliability for multi-labeled instances. In their approach, multiple labels are considered not as distinct selections, but as one divided selection, with a probability distribution over the different labels. Thus, it reflects a scenario where annotators are uncertain about which label is correct. For `augmented kappa`, each label for a specific item receives a weight which equals 1 divided by the number of labels annotated to that item.[2] For example, in item 1, label A for c_1 receives the full weight, as in a single label scenario. For c_2, label A as well as label B get a weight of 0.5. The observed agreement of an item i is then defined by:

$$A_o^i = \sum_{k' \in k'_i} W_{c1}^{k'} W_{c2}^{k'} \quad (3)$$

where k'_i is the set of intersecting labels of item i and $W_{c1}^{k'}$ and $W_{c2}^{k'}$ are the weights of the label annotated by each coder. The overall observed agreement (A_o) is the mean value of the observed agreement per item (A_o^i). The expected probability for coder c to annotate the category k for a dataset with n items is defined as:

[2]The original approach accommodates assigning different weights to each label for an item, so a distinction can be made between primary and secondary labels.

$$P(k|c) = \frac{1}{n} \sum_{i \in I} W_c^k \quad (4)$$

and the overall expected agreement is calculated as in Formula 2. For the items in Table 1, expected agreement according to this measure would be .39.

Conceptually, this measure equals the κ that would have been obtained if coders were only allowed to give one label and had randomly selected one of the multiple labels they provided. To illustrate, in the second item in Table 1, randomly selecting the labels from the two coders would result in agreement in 25% of the cases.

However, this `augmented` measure always penalizes providing multiple labels, because multiple labels reflect coder uncertainty. As a result, this measure does not take into account that there is possibly true ambiguity, and that both labels provided by a single coder may be correct. Agreement can never be higher than when these labels have been selected by chance. More specifically, if both coders assign an item to the same multiple categories, as in item 3 in Table 1, observed agreement will be 0.5 according to this measure. This is not suitable for scenarios in which researchers want to account for the fact that the multiple labels arise from the fact that there might be more than one true label.

3.3 Precision, recall and F1

Another possible solution would be to consider metrics typical for evaluating computational approaches to annotation: `precision`, `recall` and `F1` (see Brants, 2000, for a similar approach). In annotation, this can be phrased in terms of the proportion of intersecting labels compared to the total set of labels provided by the first coder (`precision`) and by the second coder (`recall`). These measures are particularly useful when one of the labels serves as the gold, for example when quality of aggregated crowd labels is compared with a gold label standard, or when a new annotator trains with a more experienced annotator.

`Precision` and `recall` allow for multiple identical labels, while correcting for providing more labels than those that are agreed upon. For example, if one coder provides more labels than the other coder, as in item 1 in Table 1, observed `F1` is decreased. However, the traditional versions of these measures do not take into account what chance performance would be. Since chance agreement depends on a variety of factors, such as the number of available categories, observed `precision`, `recall` and `F1` are not comparable across datasets that vary in the amount of labels per item.

4 Bootstrapping expected agreement

The existing measures are problematic for calculating multi-label agreement, because they do not correct for chance agreement for multiple labels (`soft-match` and `F1`) or they penalize multiple labels even if both annotators agree on the double labels (`augmented kappa`). The main contribution of the current paper is therefore to suggest a bootstrapping method for obtaining chance agreement, that can be used to adjust existing measures.

We propose to sample from the provided distribution per coder, in order to estimate the true expected agreement needed to calculate the adjusted agreement. More specifically, for each item we draw from the distribution of labels provided by each coder, following Cohen's κ. This distribution is obtained by dividing the number of times each category has been assigned by the total number of labels provided. The amount of sampled labels is likewise sampled from the probability distribution of the amount of labels for each item occurring in the original data set. In other words, if a coder provided a single label in 40% of the items and two labels in 60% of the items, double labels are also sampled in 60% of the items in the simulated dataset.

By simulating the sample data, we can bootstrap the expected agreement of several measures. For example, we can calculate the average proportion of intersecting labels across *n* simulated datasets. Using the expected agreement obtained by the simulations, existing agreement measures can be adjusted by taking into account chance agreements, using Equation 1. This allows us to obtain a variety of new measures. For calculating agreement on the intersecting labels, similar to `soft-match`, we refer to this measure as `boot-match`. Its expected agreement is estimated using the bootstrapping method, contrasting it with the `soft-match` measure where the expected agreement is calculated after removing additional labels that are not an intersecting label (see Section 3.1). Similarly, bootstrapping the expected agreement allows us to correct `precision`, `recall` and `F1` for chance agreement. The traditional version of these mea-

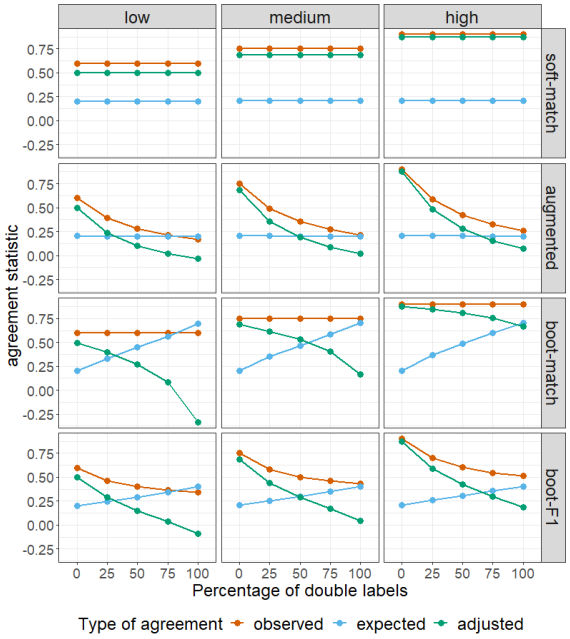

Figure 1: Agreement statistics per measure across different datasets. Each panel row displays one of the measures discussed in Sections 3 and 4. The agreement statistics on the y-axis are shown for simulated datasets with various parameters: Each panel column differs in the percentage of intersection agreement (low = 60%, medium = 75%, high = 90%) and the percentage of double labels can be found on the x-axis.

sures will be referred to as the observed components and the bootstrapping method allows us to also obtain the expected and chance-adjusted agreement components for these measures: We will therefore refer to these measures as `boot-F1`, `boot-precision` and `boot-recall`.

5 Comparing reliability measures on multi-label annotations

The previous sections show that there are various ways to estimate agreement in multi-label annotation tasks. The goal of an agreement measure is to be able to compare how much coders agree across different datasets. To evaluate which measure would be preferred for estimating agreement in multi-label datasets, a good measure should therefore (a) estimate expected agreement proportional to the amount of multi-labels and (b) provide a higher agreement score for tasks with more agreement. We will explore how the measures discussed above behave in different scenarios below.

To illustrate how the proportion of double labels as well as intersection agreement influence the scores, we simulate datasets with different characteristics for two parameters: the percentage of double labels provided and the percentage of observed intersection agreement. We thus manipulate how often a double label is chosen, ranging from never to always, with 25% intervals. In addition, the datasets vary in how much coders agree on the labels, which is manipulated by simulating data with various degrees of intersection agreement: *low* equals 60% intersection agreement, *medium* 75% and *high* 90%. For each of these parameter combinations, we sample 100 datasets. For each of these, we bootstrapped the expected agreement separately, using 100 simulations. Each dataset contained 100 items, which were assigned to one (or two) of five categories with equal probability. We then calculate the average observed, expected and adjusted (i.e. κ) agreement for each measure across these datasets.[3]

The agreement statistics for all the different datasets in the simulation analysis are provided in Figure 1. For the scenarios where no double labels are provided, the statistics are the same across the different measures: the adjusted agreement is 0.50, 0.69 and 0.87 for the different levels of intersection agreement. However, when more double-annotated labels are added, the patterns for the observed, expected and adjusted agreement diverge for the different measures.

The first row in Figure 1 considers *oracle* agreement, using the `soft-match` measure. Needless to say, the observed agreement (in red) increases when agreement is higher. Moreover, observed agreement for the `soft-match` measure remains constant when the percentage of double labels increases. However, obtaining intersection agreement is easier when more labels are provided. `Soft-match` does not take this into account, because the expected agreement does not change when more labels are provided. Across all simulated datasets, expected agreement is 0.21. As a result, the adjusted agreement for this measure is also constant across datasets with varying amounts of double labels. The adjusted agreement for `soft-match` therefore remains very close to the observed agreement, which is too liberal.

`Augmented kappa` corrects for multi-label scenarios. The expected agreement remains constant with this measure, regardless of the number

[3]Expected agreement for the subset of items where `soft-match` does not determine a single label, was calculated using the same method as for the expected agreement for `augmented kappa`.

of double labels. The observed agreement, on the other hand, decreases significantly. Note that this is partly due to the fact that the second label was sampled randomly. However, even if there would be perfect agreement on a double label for this measure (i.e. both coders assign an item to category A and B), the observed agreement is only 0.5 (see Table 1). In addition, observed agreement (and thus also the adjusted agreement) drops considerably when only 25% of items contain double labels. As a result, achieving a reliable adjusted agreement score is almost impossible with this measure: even when the intersection agreement is 90% and only a quarter of the items contain double labels, the adjusted agreement is still only 0.49. Finally, we point out that the adjusted agreement barely increases when the agreement is higher.

The observed agreement for the `boot-match` is the same as for `soft-match`, as it also considers intersection agreement. Like `soft-match`, it thus also remains constant when the dataset contains more double labels. Unlike the `soft-match` measure, however, the `boot-match` measure takes the percentage of double labels into account when calculating the adjusted agreement, as it estimates the expected agreement by simulating the provided distributions in the dataset 100 times. Expected agreement therefore increases when more labels are added, resulting in a lower adjusted agreement. With a fully double-label annotated dataset, expected agreement is 70%. In the low agreement scenario, the adjusted agreement is therefore even negative when all labels receive a double annotation. Using this adjusted agreement, we can compare how agreement on double-label datasets relates to datasets without double labels. For example, with five categories, achieving 90% intersection agreement with all double labels is comparable to achieving 75% agreement with 0% double labels (adjusted agreement ≈ 0.67). Note that this relationship depends on several parameters, such as the entropy of the label distribution and the number of categories. We will return to this issue below.

For the `boot-F1` measure, the observed agreement decreases when the percentage of double labels increases[4], similarly to `augmented kappa`. This is partly due to the fact that the manipulation of agreement was only for the intersection agreement and any additional labels besides the intersecting labels were sampled randomly. As a result, perfect agreement is not always achieved in the simulations, unlike for the `soft-match` and `boot-match` agreement. In this scenario, however, a perfect `F1` would be obtained if both coders assign an item to the same two categories. `Boot-F1` is slightly higher than `augmented`, across the different measures in the various datasets, because it yields higher agreement when both coders provide the same two labels (as in item 3 in Table 1).

More importantly, note that the expected `boot-F1` increases when more double labels are added to the dataset. Reporting only observed `F1` is therefore misleading, because achieving the same observed `F1` on a dataset with few double labels compared to one with many double labels is more difficult. The results are therefore not comparable across datasets. Calculating the chance-adjusted `boot-F1` solves this problem, because it takes this chance agreement into account.

5.1 Number of categories

These simulations reflect an annotation task with five categories. When more categories are added, however, expected agreement decreases, resulting in a higher adjusted agreement given the same observed agreement for all measures. In a zero double-label scenario, the adjusted agreement increases 5 percentage points when the number of categories increases from 5 to 10. For the measures for which expected agreement increases when more items have double labels, this increase is weaker with more categories. For example, for five categories, expected `boot-match` agreement increases from 0.20 (when no double labels are used) to 0.70 (when only double labels are used). With ten categories, this is an increase of 0.11 to 0.38. As a result, for a fully double-annotated dataset, obtaining 75% intersection agreement yields an adjusted agreement of 0.60 when ten categories are used, rather than $A_c = 0.17$ in a task with five categories. To conclude, the number of categories greatly affects the agreement statistics and should be kept in mind when evaluating the results of any annotation effort, e.g. by calculating an adjusted agreement rather than observed agreement only. Our proposed method of bootstrapping expected

[4] `Boot-recall` and `boot-precision` behave very similar to `boot-F1` here because the data is sampled similarly for the two coders. In the no and fully double-label scenarios they are exactly the same as the F1, in the in-between cases they are often slightly higher than the `boot-F1`.

agreement takes this into account.

5.2 Entropy

The entropy of the probability distribution of the labels also changes the results. The entropy reflects the likelihood that each label is chosen, and as such also influences the probability of agreement on the label:

$$H = -\sum p_k \, log \, p_k \tag{5}$$

As the categories in our simulation study are sampled with equal probability, the present distribution has a relatively high entropy. This leads to a relatively low chance agreement. The probability distribution can also have a lower entropy, if some categories are more prevalent in the dataset. For example, in the case of discourse annotations, `reason` relations occur more often in natural data than `contrast` relations. A probability distribution with a lower entropy results in a considerably higher expected agreement across all measures. Unlike with an equal probability distribution, the expected agreement for `augmented` and `soft-match` is not constant, but decreases slightly when more labels are added in a lower entropy scenario. For `soft-match` this means that the adjusted agreement even increases with more labels. Moreover, the expected agreement for `soft-match` is now much higher than that of the `augmented kappa`, because `soft-match` only calculates expected agreement after removing additional labels from items on which intersection is reached.

Finally, the decrease in the adjusted agreement for `boot-F1` and `boot-match` with more double labels is reduced. In real-world datasets, labels likely do not have equal probability, such as in our simulation analysis. Because the entropy of the probability distribution influences chance agreement, the adjusted agreement of the measure rather than the observed agreement should therefore be reported. This makes annotation agreement more comparable across datasets.

6 Case study

Real-world datasets often have very different characteristics than the simulated datasets in the previous section. We therefore explore how the measures behave in a real-world dataset: a case study for discourse relation annotation, which is a notoriously difficult task. It is often difficult to achieve a $\kappa > .7$ on single-label annotations (Spooren and Degand, 2010). This is partly due to the fact that discourse relation frameworks often distinguish many different categories (as can be seen below). In addition, coherence is not a feature of the text, but of the mental representations that readers have of the text (Sanders et al., 1992). Therefore, discourse relation annotations depend on coders' interpretation of the text, which may be subjective. Furthermore, ambiguity plays an important role for discourse relation annotation, which can partly explain low agreement with single label annotations. Recent studies have therefore turned to crowd-sourcing to source discourse relation annotations (e.g. Yung et al., 2019; Scholman et al., 2022; Pyatkin et al., 2020). This allows researchers to capture a larger variety of interpretations per instance. However, in some cases, such an approach is not suitable, or researchers still target a single final label.

The case study was part of a psycho-linguistic experiment in which participants were asked to provide a one-sentence continuation to a prompt. Two expert coders annotated the continuations with respect to their discourse relation with the prompt. The coding scheme allowed for maximally two labels, when coders believed both senses held. In total, the coders annotated 884 items, using 19 categories. 11 of these categories occurred in the final intersection label. The first coder had provided double labels in 11.4% of the cases, the second coder in 65.5% of the items. As a result, the first coder has a greater influence on the intersection item, because only one of the items by the second coder will be selected when one of them overlaps with the label provided by the first coder.

As can be seen in Table 2, agreement is moderate. The adjusted agreement of `soft-match` reflects the best-case scenario: if both coders only chose the single intersection label, the κ would have been .73. `Boot-match` corrects the expected agreement based on the proportion of double labels, resulting in a lower adjusted agreement.[5] `Soft-match` thus over-estimates agreement. According to `augmented kappa`, the adjusted agreement would be highly insufficient, because many double labels have been provided. This measure assumes that the multiple labels reflect coder uncertainty, but these expert annotators only provided two labels if they thought that both

[5]For the boot-strapped estimates presented in this section, we used 1000 simulations.

	observed	expected	adjusted
soft-match	.79	.23	.73
augmented	.51	.21	.38
boot-match	.79	.34	.68
boot-rec.	.55	.21	.43
boot-prec.	.76	.31	.65
boot-F1	.62	.24	.50

Table 2: Agreement statistics per measure for the case study (rec. = recall, prec. = precision).

labels were true. Boot-precision of c_2 with respect to c_1 is lower than boot-recall in this same direction. This can be attributed to the fact that the first coder provided fewer double labels than the second coder. Boot-F1 for this dataset is relatively low, even though the coders score high on the boot-match agreement. The reliability of the single final label is thus quite high, but the coders diverged on when and what additional labels should be provided.

Traditional F1 does not always reflect agreement properly, for two reasons. First, observed F1 will decrease when expert annotators find additional labels that might also be true. Secondly, observed F1 does not take into account the chance agreement on this measure. Adjusted boot-F1 and boot-match display agreement more accurately. As shown above, each measure provides different insights into the data quality and which measure(s) should be reported therefore depends on the goal of the annotation effort. Finally, the case study shows that even when annotators are instructed in the same way, the number of double labels that they provide still diverges between the two coders. Recall and precision provide more insight in this.

7 Related work

Bhowmick et al. (2008) also propose an adjusted κ measure to account for multi-labeled annotations. Crucially, their proposed metric considers the non-inclusion in a category by an annotator pair as an agreement. Such an approach is not optimal for annotation scenarios which can be characterized by a large number of categories in the coding scheme. This includes certain discourse relation annotation efforts, for which coding schemes can contain over 40 categories. With such large schemes, coders likely do not consider every category separately during annotation of a single item, but rather consider a subset of categories that seem most applicable to the item.

Finally, relating to the issue of ambiguity in annotation and selecting a single final label, we note that this is a debatable issue in itself. In correspondence to the assumption that items can express more than one meaning, a soft label – consisting of a probability distribution of all categories per item – more accurately captures an item's ambiguity. For example, CrowdTruth (Aroyo and Welty, 2013; Dumitrache et al., 2018) evaluates data quality by capturing the ambiguity inherent in semantic annotation through the use of disagreement-aware metrics. Fornaciari et al. (2021) and Uma et al. (2021) showed that models trained on soft labels, such as these, outperform those trained on single-label data, especially if they are evaluated using soft labels as well. A larger number of coders would be needed to more accurately calculate a probability distribution for an item. This is not the case for tasks that require (or choose to use) expert annotations only, as is the case in psycho-linguistics, or when annotating a gold dataset.

8 Discussion and conclusion

Annotating data with multiple labels better reflects the true meaning of the items, as these items can be ambiguous or even have multiple interpretations. After all, the goal of agreement measures is not to establish how strongly the coders agree, but rather how reliable the label is. The label on which annotators agree is more likely to be a true label, regardless of whether all the labels are captured. Obtaining a distribution of labels may be helpful in some, but not all, tasks (cf. Fornaciari et al., 2021). In addition, for many tasks such distributions are not available, either because not enough observations were obtained, or because computational models predict a limited number of labels. The present study therefore explored measures for evaluating various agreement measures on scenario's with more than one label.

The augmented kappa has been proposed as a measure of agreement on multi-label annotated datasets (Rosenberg and Binkowski, 2004), but it penalizes additional labels heavily and does not consider items assigned to the same multiple categories as full agreement. The underlying assumption is thus that there is only one true label, reflecting uncertainty rather than ambiguity. However, it is not always true that only one true label exists

(Aroyo and Welty, 2015). `Precision`, `recall` and `F1` should also be corrected for chance agreement, as that varies with respect to how many labels are provided to each item. Observed `precision`, `recall` and `F1` are therefore not comparable across datasets that differ in e.g. the amount of multiple labels.

When additional labels are potentially correct, `intersection` agreement is a good option, but only when it is corrected for chance agreement. Rather than taking the intersecting label to calculate adjusted agreement as in a `soft-match` measure, the data could be simulated to estimate the true chance agreement on the intersection (i.e. `boot-match`). This measure increases expected agreement when more labels are provided, resulting in a lower adjusted agreement. Coders should therefore only provide double labels if they are certain that both labels hold.

One limitation that needs to be taken into account when selecting the intersection label, is that the coder who provides fewer labels influences the final label more than the coder who provides more labels. Ideally, coders would therefore provide a similar proportion of double annotations. Furthermore, in an extreme case where one coder assigns an item to all categories, there would always be agreement. This is corrected for slightly in `boot-match`, but is more easily detected using `boot-F1` and similar measures. Finally, when some labels are more frequent than others, achieving intersection agreement on this label is more likely. As a result, the intersection agreement will contain a higher proportion of dominant labels. The distribution of the intersecting labels thus does not necessarily reflect the true distribution of labels in the dataset and researchers should be careful to draw conclusions about the distributions of aggregated labels.

To conclude, if only one true label is believed to be possible for each item, `augmented kappa` can be used to calculate agreement in cases where annotators provide more than one label. However, if items are believed to be ambiguous, with possibly more than one true label per item, `boot-match` best estimates the reliability of a single final label per item. `Boot-F1` and related measures reveal more about the structure of the data, such as asymmetries between the coders. Which measure is reported therefore depends on the goal of the annotation effort. Regardless, for all of these measures, chance agreement should be taken into account to make the measure comparable across datasets with different characteristics. As demonstrated above, our proposed method of bootstrapping the expected agreement can be used for this.

Acknowledgments

MM and FY are supported by the German Research Foundation (DFG) under Grant SFB 1102 (“Information Density and Linguistic Encoding", Project-ID 232722074). MS and VD are supported by the European Research Council (ERC) under the European Union’s Horizon 2020 Research and Innovation Programme, Grant 948878 (“Individualized Interaction in Discourse”).

References

Jacopo Amidei, Paul Piwek, and Alistair Willis. 2019. Agreement is overrated: A plea for correlation to assess human evaluation reliability. In *Proceedings of the 12th International Conference on Natural Language Generation*, pages 344–354.

Lora Aroyo and Chris Welty. 2013. Crowd truth: Harnessing disagreement in crowdsourcing a relation extraction gold standard. *WebSci2013. ACM*, 2013(2013).

Lora Aroyo and Chris Welty. 2015. Truth is a lie: Crowd truth and the seven myths of human annotation. *AI Magazine*, 36(1):15–24.

Ron Artstein and Massimo Poesio. 2008. Inter-coder agreement for computational linguistics. *Computational Linguistics*, 34(4):555–596.

Valerio Basile, Michael Fell, Tommaso Fornaciari, Dirk Hovy, Silviu Paun, Barbara Plank, Massimo Poesio, and Alexandra Uma. 2021. We need to consider disagreement in evaluation. In *Proceedings of the 1st Workshop on Benchmarking: Past, Present and Future*, pages 15–21.

Christin Beck, Hannah Booth, Mennatallah El-Assady, and Miriam Butt. 2020. Representation problems in linguistic annotations: ambiguity, variation, uncertainty, error and bias. In *14th Linguistic Annotation Workshop*, pages 60–73.

Plaban Kumar Bhowmick, Anupam Basu, and Pabitra Mitra. 2008. An agreement measure for determining inter-annotator reliability of human judgements on affective text. In *Coling 2008: Proceedings of the workshop on Human Judgements in Computational Linguistics*, pages 58–65.

Thorsten Brants. 2000. Inter-annotator agreement for a german newspaper corpus. In *Proceedings of the Second International Conference on Language Resources and Evaluation*, pages 165–172.

Jacob Cohen. 1960. A coefficient of agreement for nominal scales. *Educational and Psychological Measurement*, 20(1):37–46.

Ludivine Crible and Liesbeth Degand. 2019. Reliability vs. granularity in discourse annotation: What is the trade-off? *Corpus Linguistics and Linguistic Theory*, 15(1):71–99.

Anca Dumitrache, Oana Inel, Lora Aroyo, Benjamin Timmermans, and Chris Welty. 2018. Crowdtruth 2.0: Quality metrics for crowdsourcing with disagreement. *arXiv preprint arXiv:1808.06080*.

Tommaso Fornaciari, Alexandra Uma, Silviu Paun, Barbara Plank, Dirk Hovy, and Massimo Poesio. 2021. Beyond black & white: Leveraging annotator disagreement via soft-label multi-task learning. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 2591–2597.

Barbara Plank, Dirk Hovy, and Anders Søgaard. 2014. Linguistically debatable or just plain wrong? In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 507–511.

Valentina Pyatkin, Ayal Klein, Reut Tsarfaty, and Ido Dagan. 2020. QADiscourse: Discourse relations as qa pairs: Representation, crowdsourcing and baselines. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing*, pages 2804–2819.

Andrew Rosenberg and Ed Binkowski. 2004. Augmenting the kappa statistic to determine interannotator reliability for multiply labeled data points. In *Proceedings of HLT-NAACL 2004: Short Papers*, pages 77–80.

Ted J. M. Sanders, Wilbert P. M. S. Spooren, and Leo G. M. Noordman. 1992. Toward a taxonomy of coherence relations. *Discourse Processes*, 15(1):1–35.

Merel C. J. Scholman, Tianai Dong, Frances Yung, and Vera Demberg. 2022. Discogem: A crowdsourced corpus of genre-mixed implicit discourse relations. In *Proceedings of the Thirteenth International Conference on Language Resources and Evaluation (LREC'22)*, Marseille, France. European Language Resources Association (ELRA).

Wilbert Spooren and Liesbeth Degand. 2010. Coding coherence relations: Reliability and validity.

Alexandra N Uma, Tommaso Fornaciari, Dirk Hovy, Silviu Paun, Barbara Plank, and Massimo Poesio. 2021. Learning from disagreement: A survey. *Journal of Artificial Intelligence Research*, 72:1385–1470.

Frances Yung, Vera Demberg, and MCJ Scholman. 2019. Crowdsourcing discourse relation annotations by a two-step connective insertion task. In *Proceedings of the 13th Linguistic Annotation Workshop (LAW)*, pages 16–25.

Biographically Relevant Tweets – A New Dataset, Linguistic Analysis and Classification Experiments

Michael Wiegand
Digital Age Research Center (D!ARC)
Alpen-Adria-Universität Klagenfurt
AT-9020 Klagenfurt, Austria
michael.wiegand@aau.at

Rebecca Wilm
Institute of Computational Linguistics
Heidelberg University
D-69120 Heidelberg, Germany
wilm@cl.uni-heidelberg.de

Katja Markert
Institute of Computational Linguistics
Heidelberg University
D-69120 Heidelberg, Germany
markert@cl.uni-heidelberg.de

Abstract

We present a new dataset comprising tweets for the novel task of detecting biographically relevant utterances. Biographically relevant utterances are all those utterances that reveal some persistent and non-trivial information about the author of a tweet, e.g. habits, (dis)likes, family status, physical appearance, employment information, health issues etc. Unlike previous research we do not restrict biographical relevance to a small fixed set of pre-defined relations. Next to classification experiments employing state-of-the-art classifiers to establish strong baselines for future work, we carry out a linguistic analysis that compares the predictiveness of various high-level features. We also show that the task is different from established tasks, such as aspectual classification or sentiment analysis.

1 Introduction

Since its beginning, the web has been a valuable data source for natural language processing (NLP) applications. Particularly social media contain data complementary to what previous (news) text corpora could offer (Farzindar and Inkpen, 2015), such as information on ordinary people.

This work focuses on biographical information on ordinary users of Twitter. The **task** is to identify tweets that contain biographically relevant information on the author of a tweet. We frame this as a **binary text classification task**: We distinguish between *biographically relevant tweets* (1)-(5) and *biographically irrelevant tweets* (6)-(10).

(1) *relevant:* I'm a grown man in my mid-30s with no children
(2) *relevant:* Today was the first day at Deckers my manager told me good job lmao.
(3) *relevant:* one of the reasons i regularly wear @AmericanOutlaws kit is because it makes a political statement
(4) *relevant:* i'm actually rather shy in real life
(5) *relevant:* Brenda Fricker will always be my favourite Irish actress
(6) *irrelevant:* It's after 1 a.m. here and I am stupidly awake
(7) *irrelevant:* @USER i wanna buy your mum a beverage.
(8) *irrelevant:* I heard they're changing a dollar for corn at the Tampa Bay stadium, that's a buck an ear.
(9) *irrelevant:* I'm glad I stayed home last night but I'm sad I was alone lmfaooooooooo
(10) *irrelevant:* Headache? I've had a headache all day xx

Unlike previous work (Sekine and Artiles, 2009; Ji et al., 2010; Garcia and Gamallo, 2015), we **do not use a pre-specified list of biographical relations**. Instead, **we consider everything as a biographically relevant relation if it reveals some persistent and non-trivial information on the user authoring a particular tweet**. This may include simple information on age and family status (1), (major) life events (2), habits (3), personality traits (4), (dis)likes (5), etc. Tweets may offer a plethora of biographical information about their users and we believe that such unrestricted setting is faithful to the nature of the data that we consider.

Biographical information on ordinary users would be valuable for various applications. Providers of social-media sites could support their users with effectively building up their social network, for example, by recommending friends based on shared interests (Xie, 2010). Another obvious application with a high commercial potential is product recommendation (Ricci et al., 2011). Personalized advertisement using biographical information about a user (e.g. recommending baby care

Proceedings of the 29th International Conference on Computational Linguistics, pages 3669–3679
October 12–17, 2022.

products for parents-to-be) may be much more effective than undirected general advertisement.

Biographical information may also be used for applications that are **questionable from an ethical point of view**. For instance, potential employers could do background checks on job seekers. Similarly, insurance companies could screen potential insurants before offering them a particular insurance. Specific biographic information could well have a negative impact on the customers. For example, a health insurance might decline a customer an offer if they knew about their unhealthy life style.

We are fully aware of these ethical issues and we want to make clear that our aim is not to provide basic research for technologies that could be used for such questionable purposes to evolve. On the contrary, we could **envisage scenarios that would help users to protect the above misuse of their personal data** (Malandrino et al., 2013). For instance, the functionality to identify biographically relevant information in user-generated content may be incorporated in an application that automatically anonymizes such sensitive text passages (Medlock, 2006; Lison et al., 2021) or, at least, warns users prior to posting such content that they are about to disclose sensitive material.

The question may arise **why we aim for a binary classification task and do not specify the subtype of biographical information**. We consider our task just as the first step in the pipeline of processing biographically related material. Obviously, processing all microposts of a social-media site manually would be a prohibitive task. A great reduction of work could be achieved by selecting all biographically-relevant material automatically, as proposed in this paper. Only a small proportion of tweets within a user timeline from Twitter – our estimate based on our data is that this is about 33% – actually contains such information.

The aim of this research is two-fold. First, we want to investigate whether biographically relevant utterances can reliably be identified. Second, we also want to determine what types of biographically relevant information one may encounter on Twitter.

Our **contributions** are the following:

- We introduce the novel task of detecting biographically relevant utterances.
- We present a new dataset for this task.
- We report on classification performance using state-of-the-art supervised classifiers establishing strong baselines for future work.
- By testing classifiers trained on related tasks on our task, we can show that our novel task is different from previous tasks.
- We carry out a linguistic analysis on our new dataset which sheds more light on the nature of biographically relevant utterances.

All data created as part of this research are made available for research purposes upon request.[1]

2 Related Work

A substantial body of previous research on the extraction of biographical information using NLP focuses on pre-defined relations, such as *hasBirthDate(<person>,<date>)* (Sekine and Artiles, 2009; Ji et al., 2010; Garcia and Gamallo, 2015). In this paper, we depart from this setting. Our observation is that in social media there is plenty of biographically relevant information that does not easily fit into this finite set of relations:

(11) my parents were complete idiots and I try to do the opposite of anything they would ever do

As a data source for extracting biographical information, Wikipedia[2] has, so far, been the most frequently used corpus (Garera and Yarowsky, 2009; Garcia and Gamallo, 2015; Plum et al., 2019). Consequently, previous research typically focused on celebrities or people who are notable in some way since it is this group of people who are predominantly represented in Wikipedia. By focusing on social media (i.e. Twitter), we are able to extract biographical information from *ordinary people*. This also means that we are able to widen the scope of biographically relevant information.

The task addressed in this paper also bears some relation to multi-document (biography) summarization (Zhou et al., 2004; Biadsy et al., 2008) which is typically framed as a two-step approach: In the first step, biographically relevant utterances (from different documents) are identified while in the second step, a summary of these extracted utterances is generated by eliminating redundancies and creating a plausible order. Our task matches the first step although in the context of summarization previous work only dealt with pre-defined relations.

[1] Due to the sensitivity of the data, access is only granted if an outline of the intended research is submitted via email to the authors and that outline is considered ethically sound. The annotation guidelines we used for producing our dataset, however, can be downloaded directly: https://github.com/miwieg/biographical_relevance_guidelines

[2] https://en.wikipedia.org

number of timelines	27
number of tweets	14,483
average number of tweets per timeline	536.4
average number of tokens per tweet	17.7
type-token ratio	8.7%
category: biographically relevant tweets	4,724 (32.6%)
category: biographically irrelevant tweets	9,759 (67.4%)

Table 1: Statistics of the BRT-dataset.

With regard to social media, there have been related research efforts in the area of *author profiling* (Rao et al., 2010; Zamal et al., 2012; Rangel et al., 2013). These works focus on demographic categories, such as predicting gender, age or personality traits. Moreover, there has been research on extracting *major life events*. Such events are environmental circumstances that have an identifiable onset and ending and may carry the potential for altering an individual's present state of mental or physical well-being (Goodyer, 2001). Examples are getting married or finding a new job (Li et al., 2014; Li and Cardie, 2014; Dickinson et al., 2015). As we will provide evidence later, major life events and demographic categories only form a subset of the relations we consider as biographical relevance.

In their recent study, Saha et al. (2021) examine what life events are disclosed on social media. The authors establish that a significant number of such events can be found. Unlike our work in which we address both data analysis and classification experiments, Saha et al. (2021) present a purely descriptive study. Moreover, that work is carried out on data collected from Facebook, while our work is carried out on data crawled from Twitter. While all life events represent biographically relevant information, the set of biographically relevant utterances considered in this paper also includes relations beyond life events, e.g. likes or habits.

In the area of privacy protection (Malandrino et al., 2013), there has also been research using NLP. Mao et al. (2011) present a descriptive study examining properties of three types of leaks on Twitter: divulging vacation plans, tweeting under the influence of alcohol and revealing medical conditions. Cappellari et al. (2017) present a tool for privacy detection trained on 500 tweets manually labeled as *private* or *not-private* using SVM. Neerbek et al. (2017) introduce a system that distinguishes between *sensitive* vs. *non-sensitive* information based on recursive neural networks (Irsoy and Cardie, 2017). Information about the training data used are not provided. Canfora et al. (2018) establish a set of heuristic rules operating on dependency parses for the detection of privacy leaks. The latter are approximated by mentions of locations or emotions. The approach is examined on a random set of 856 Facebook statuses. Our work differs from these works in that our focus is not just on sensitive data but all different types of biographical information. For example, habits, personality traits or likes and dislikes are biographical information but they need not be categorized as (highly) sensitive data. Moreover, our dataset with about 14k tweets that has been annotated manually is considerably larger. It has also been constructed in a more principled and less biased way. For example, Mao et al. (2011) create their dataset by sampling for specific topic keywords while our raw data represent timelines of users (§3). Finally, our classification experiments are carried out using state-of-the-art classifiers which are much more robust than those in previous work (e.g. SVM). Unfortunately, none of those previous datasets on privacy protection are publicly available, so we could not consider them in our evaluation.

3 Data

We now describe the creation of our novel dataset to detect **b**iographically **r**elevant **t**weets, henceforth referred to as **BRT-dataset**. We did not sample our dataset using specific keywords since this results in topic biases (Wiegand et al., 2019). Neither did we randomly sample tweets since, as our exploratory experiments revealed, the tweets would have a bias towards topics predominant during collection time, such as COVID-19. Instead, we extracted timelines of users. Using the Twitter API[3], we selected users from Australia, Canada, Ireland, New Zealand, the UK and the USA who had recently posted a tweet containing the first-person singular pronoun. Since we could only annotate a small number of timelines, we tried to come up with a subset of very different users (having different ages, genders, ethnicities, sexual orientations, political beliefs etc.). Thus, we hope to appropriately account for the demographic diversity of our society.

In order to reduce the manual annotation, we got rid of trivial cases of irrelevant tweets, i.e. tweets that do not contain any mention of the author of the tweet itself. That is, we excluded tweets without a mention of the first-person pronoun.[4] Since we

[3] https://developer.twitter.com/en/docs/twitter-api

[4] On a set of about 2,000 tweets that were thus discarded,

did not want to discard cases of *implicit protagonists* (12) (Regneri et al., 2011), which despite the absence of such pronoun refer to the author, we applied some further heuristics (e.g. by including declarative sentences that begin with a verb).

(12) taking gym selfies#joined a gym today!

Each tweet of those timelines was manually labeled. For practical reasons, tweets were annotated in isolation. As already discussed in §1, we define a tweet as biographically relevant if it reveals some persistent and non-trivial information on the user authoring a particular tweet. Tweets were to be labeled as biographically relevant no matter whether the information was mentioned explicitly (13) or implicitly (14). In (13) the author explicitly states their pride to be Americans. However, in (14), we infer the author's reading enthusiasm since (s)he asks for book recommendations. In BRT, 28% of the biographically relevant utterances are implicit.

We insisted that the biographically relevant information should be unambiguous. For instance, (15) could imply that the author is a regular churchgoer in which case the tweet would be biographically relevant. However, since there are also several other possible explanations (for instance, the author might just be attending a wedding) we did not label such cases as biographically relevant.

(13) my dad has always hung an American flag outside our house, cause we are PROUD TO BE AMERICANS
(14) uhm any book recommendations? i see a day trip to barnes n noble in my near future for mental health care
(15) #Omw to church. :)

On a random subset of 300 tweets we measured the agreement between our annotator (a member of the department of one of the co-authors) and one co-author of this paper. We obtained a substantial inter-annotator agreement of Cohen's $\kappa = 0.73$ (Landis and Koch, 1977).

Table 1 provides some summarizing statistics of our final dataset. The dataset consists of more than 14,000 tweets. Only about one third of them are considered biographically relevant.

4 High-Level Features

In this paper, we follow a supervised learning approach. Among the different classifiers we consider, we also implement a feature-based classifier using high-level features. These features are also used for a descriptive analysis on our BRT-dataset. In the following, we describe the feature set we devised.

4.1 Aspectual Classification (ASPECT)

By aspect, we understand how an action, state or event denoted by a verb extends over time.[5] We consider the aspectual categories as proposed by Friedrich and Pinkal (2015) who distinguish between static aspect, i.e. clauses expressing states (16), episodic aspect, i.e. clauses expressing information about events (17), and habitual aspect, i.e. clauses expressing regularities (18).

(16) I like coffee. (*invented example*)
(17) I bought some coffee right now. (*invented example*)
(18) I usually drink coffee after lunch. (*invented example*)

While biographically relevant utterances can co-occur with all 3 aspectual categories, we assume that they are not equally distributed across these categories. For example, most events (=episodic aspect) should not be life-changing ones and therefore be biographically irrelevant. On the other hand, habitual aspect may often refer to habits or hobbies that tell us a lot about a person and therefore could qualify as biographical relevance.

As a tool to determine the aspectual category of an utterance, we use *sitent* (Friedrich et al., 2016).

4.2 Sentiment Analysis (SENTI)

Intuitively, there seems to be a relation between sentiment (Liu, 2012) and biographical relevance as people's likes and dislikes are typically expressed by positive and negative sentiment (19). However, not all instances of positive and negative sentiment may be biographically relevant. For instance, (dis)likes that are shared by everyone (20) are considered irrelevant since they represent trivial information. We run *TweetEval* (Barbieri et al., 2020) on the tweets of BRT. In addition to a feature that indicates the predicted sentiment category (i.e. positive, negative or neutral), we included features that reflect the range in which the confidence probability score of the prediction for a particular tweet falls. We divide that score into bins of size 0.1.

(19) I honestly hate group assignments.
(20) I really hate doing something that I don't want to do

we only identified 4% biographically relevant instances. Therefore, we can conclude that our filtering heuristic only misses a negligible proportion of actual biographically relevant tweets.

[5] https://en.wikipedia.org/wiki/Grammatical_aspect

4.3 Emotion Classification (EMOTION)

We also want to investigate in how far emotion categories, such as *joy*, *fear* or *surprise*, correlate with biographical relevance. Intuitively, certain emotional states may coincide with the author exhibiting some mental condition (21). We consider the *NRC emotion lexicon* (Mohammad and Turney, 2013) and associate each tweet with the set of emotion categories that are triggered by the respective emotion words contained in the tweet.

(21) seasonal desperation and regular depression ready to fuck me over

4.4 Supersenses (SUPER)

WordNet (Miller et al., 1990) groups each synset into one of 45 supersenses[6] which represent coarse-grained semantic categories, such as *noun.food* or *verb.motion*. For a linguistic analysis, supersenses can be informative as they may indicate which semantic concepts correlate with biographical relevance. We represent each tweet by the supersenses associated with the words that occur in it.[7]

4.5 Part-of-Speech Information (POS)

We investigate whether biographical relevance displays a specific distribution of part-of-speech (POS) tags. Each tweet is associated with the set of POS tags of the words occurring in it. We use the POS tagger by Owoputi et al. (2012), which has been optimized for Twitter.

4.6 Family-Member Wordlist (FAMILY)

A frequently occurring subtype of biographical relevance are family relations, e.g. does the author have children, are they married, do they have any siblings etc. Since family members and partners represent a clear-cut concept, we compiled a list of 105 lemmas expressing these relations. We implemented a look-up feature that indicates whether any of these lemmas could be found in a tweet.

4.7 Meta Features (META)

We also want to determine the effectiveness of meta information for our task. We distinguish between 3 types of meta features described below. All features are binary features. This design decision was made since it substantially facilitates our linguistic analysis in §5.[8] Table 2 summarizes all meta features we examine in this work. Several of these features represent rankings. They were discretized by dividing the range of values into bins. Table 2 also indicates how the discretization is conducted.

Tweet-Level Meta Features. These are features that refer to an individual tweet. Two of them need to be explained in more detail since they required a special form of normalization. It concerns the feature that counts the number of likes assigned to a tweet (*META_TWEET_is_among_top_n_likes*) and the feature that counts the number of the times a feature has been retweeted (*META_TWEET_is_among_top_n_retweets*). Since different users typically receive quite a different number of likes (or number of retweets), we normalized that number by the average number of likes (or retweets) a particular user obtained. The resulting ranking therefore reflects whether a tweet received more or fewer likes (or retweets) than expected.

Thread-Level Meta Features. These are features referring to the thread a tweet is situated in.

User-Level Meta Features. These are features referring to the user that authored a particular tweet.

5 Linguistic Analysis

Table 3 shows for both classes the 20 high-level features (§4) with the highest precision. The strongest feature to correlate with biographical relevance is the word list referring to family relations. This result is quite intuitive. However, since that feature only fires in 863 of the more than 4,000 cases of biographical relevance on BRT, we can conclude that our data also contain many other forms of biographical relevance. A considerable proportion of further predictive features concern supersenses. The most predictive one concerns pertainyms (*adj.pert*). These seem to be relational adjectives that often relate to demographic information (e.g. *my economic situation*, *British nationality*). Further nouns expressing feelings (*noun.feeling*) may refer to likes, interests or mental health (e.g. *love*, *anxiety*). Temporal nouns (*noun.time*) may refer to *birthday*, *age* or *anniversary* etc. In terms of POS-tags, the possessive pronoun is the strongest. It is often part of frequently occurring biographically relevant

[6] https://wordnet.princeton.edu/documentation/lexnames5wn

[7] Due to the lack of robust word-sense disambiguation, we represent each word as the union of synsets containing it.

[8] It is much more straightforward to rank binary features according to the predictiveness towards a particular class than a feature set containing non-binary (continuous) features.

Abbreviation	Explanation	Info. about Discretization
META_TWEET_is_answer	is tweet an answer	
META_TWEET_is_retweet	is tweet a retweet	
META_TWEET_is_quote	is tweet a quote tweet	
META_TWEET_is_posted_on_weekend	has tweet been posted during weekend	
META_TWEET_is_among_top_n_likes	is tweet among top n tweets with the most likes	n: [200, 500, 1000, 2000]
META_TWEET_is_among_top_n_retweets	is tweet among top n tweets with the most retweets	n: [200, 500, 1000, 2000]
META_THREAD_is_thread_length_n	does thread have n tweets	n: [0, 1, 2, 3, 4, 5 or more]
META_THREAD_has_n_contributors	does user thread have n contributors	n: [0, 1, 2, 3, 4, 5 or more]
META_THREAD_has_n_tweets_from_user	did user post n tweets in this thread	n: [0, 1, 2, 3, 4, 5 or more]
META_USER_is_country_code_x	is country code of user x (e.g. UK, US, CA etc.)	
META_USER_has_n_to_m_likes	does user have between n and m likes	bins (n, m) are log. scaled
META_USER_has_n_to_m_followers	does user have between n and m followers	bins (n, m) are log. scaled
META_USER_is_following_n_to_m_users	is user following between n and m users	bins (n, m) are log. scaled
META_USER_posted_n_to_m_tweets	did user post between n and m tweets	bins (n, m) are log. scaled
META_USER_included_in_n_to_m_lists	is user included in n to m lists	bins (n, m) are log. scaled
META_USER_has_default_profile	does tweet have user default profile	

Table 2: Overview of meta features.

phrases, such as *my wife*, *my dog* or *my car*. Finally, we also find a few predictive meta features: If a tweet of a user is more often *liked* or *retweeted* than the average tweet (of the user), then it is likely to be biographically relevant.

With regard to biographical irrelevance, we find many fewer linguistic features and predominantly meta features. The most predictive feature is whether a tweet is a retweet. Given that tweets are typically retweeted by users other than the author of the original tweet[9], there are (at least) three possible reasons for the predictiveness of this feature. First, it is possible that users do not disclose or forward biographical information of other people often. Second, maybe people are more interested in writing about themselves. Third, as we only include tweets written in the first person, we might miss biographical information in the second or third person.

Most of the other features predictive for biographical irrelevance are meta features that refer to the thread of the tweet. These features are also correlated to each other. If a thread is lengthy or has many users contributing to it, then this means that there is some deeper discussion about it. Obviously biographically relevant content is less likely to be the centre of such discussions. Apart from the meta features, sentiment information is the most predictive linguistic feature for biographical irrelevance. Tweets that have been classified as neutral with a high confidence are likely to be biographically irrelevant. This is plausible since likes and dislikes (which are mostly biographically relevant) are positive or negative.

Neither aspectual categories (§4.1) nor emotion categories (§4.3) occur in either of the rankings. The latter category was observed frequently in both biographically relevant tweets (22) and biographically irrelevant tweets (23).

(22) seasonal desperation and regular depression ready to fuck me over

(23) @SmartRachael It's a good day. I'm happy. X

As far as aspectual classification is concerned, we noted heavy noise in the output of *sitent*, i.e. the tool that provided us with an aspectual analysis of our data.

6 Classification Experiments

We now report on our classification experiments.

Baselines. As baselines, we consider a **majority-class classifier** that always predicts biographical irrelevance. Furthermore, we consider a **sentiment classifier** which predicts biographical relevance if a tweet either conveys positive or negative rather than neutral sentiment. (As in §4.2, we obtain sentiment information using *TweetEval*.) Our third baseline is an **aspectual classifier** which predicts biographical relevance if a tweet conveys *habitual* aspect. (As in §4.1, we obtain the aspectual classification using *sitent*.) Our fourth baseline is a **classifier using meta-information**. Motivated by our analysis from §5, it predicts biographical relevance if a tweet is neither a retweet, a quote tweet nor represents the answer to a tweet. The more predictive this baseline is, the less important we should consider NLP-based information for this task.

[9]Please note that there is a difference between the feature *META_TWEET_is_among_top_200_retweets* which is predictive for biographical relevance and *META_TWEET_is_retweet* which is predictive for biographical irrelevance. The latter feature indicates that a tweet is a retweet itself while the former is related to the number of times the tweet has been retweeted. Apparently, a retweet per se is an indication of biographical irrelevance while if a tweet of a particular user has been retweeted disproportionally often, then there is a tendency of the tweet being biographically relevant.

Biographically Relevant			Biographically Irrelevant		
Feature	**Prec**	**Freq**	**Feature**	**Prec**	**Freq**
FAMILY	73.8	863	META_TWEET_is_retweet	83.3	216
SUPER_adj.pert	53.1	682	META_THREAD_is_thread_length_4	80.8	318
META_USER_has_5000_to_9999_followers	52.3	285	META_THREAD_has_4_tweets_from_user	80.7	57
SUPER_noun.phenomenon	50.6	1,557	SENTI_neutral_confidence_range_0.9_to_1.0	75.7	66
POS_PRP$	48.4	4,672	META_TWEET_is_quote	75.2	1,587
META_TWEET_is_among_top_500_likes	47.8	500	META_THREAD_has_2_tweets_from_user	75.1	438
POS_RRB	47.5	383	META_THREAD_has_2_contributors	74.7	2,251
SUPER_noun.Tops	47.4	1,855	META_THREAD_is_thread_length_3	74.6	574
SUPER_noun.feeling	47.2	1,375	META_USER_has_1,000_to_1,999_likes	74.5	2,139
META_USER_posted_20000_to_49999_tweets	46.7	197	META_TWEET_is_answer	73.8	6,325
META_TWEET_is_among_top_200_likes	46.5	200	META_THREAD_is_thread_length_2	73.8	1,581
META_USER_is_following_1000_to_1999_users	46.0	491	META_THREAD_has_1_tweet_from_user	73.7	2,091
META_TWEET_is_among_top_200_retweets	46.0	200	META_USER_posted_1000_to_1999_tweets	73.5	898
SUPER_noun.relation	46.0	774	META_THREAD_is_thread_length_1	73.5	3,248
POS_LRB	45.8	393	META_THREAD_has_1_contributor	73.4	3,581
SUPER_noun.time	45.8	4,306	META_THREAD_has_4_contributors	73.1	130
POS_HT	45.5	1,065	POS_USR	73.0	7,061
POS_RBR	45.4	359	META_USER_is_following_0_to_99_users	72.7	2,532
SUPER_noun.group	45.2	1,629	META_USER_is_country_code_US	71.6	6,365
POS_JJS	44.8	411	META_USER_is_following_200_to_499_users	71.6	2,450
random precision	32.6		**random precision**	67.4	

Table 3: Top 20 features with highest precision for the different classes.

		Randomized Folds				Unseen Timelines			
Classifier	**Supervised Classifier?**	**Acc**	**Prec**	**Rec**	**F1** (*std*)	**Acc**	**Prec**	**Rec**	**F1** (*std*)
majority-class baseline	no	67.4	33.7	50.0	40.3	67.4	33.7	50.0	40.3
sentiment baseline	no	45.0	50.5	50.5	50.5	45.0	50.4	50.5	50.4
aspectual baseline	no	65.3	55.7	53.1	54.3	65.3	55.7	53.1	54.4
meta baseline	no	59.7	57.8	58.7	58.2	59.7	58.4	58.9	58.7
logistic regression w. high-level features	yes	73.0	69.5	64.9	67.1	68.4	64.3	60.6	62.4
logistic regression w. bag of words	yes	76.7	74.1	70.2	72.1	74.2	71.1	66.9	69.0
RoBERTa	yes	85.4	83.4	83.4	83.4 (±0.1)	83.0	80.8	80.6	80.7 (±0.2)
BERTweet	yes	85.7	83.7	84.2	83.9 (±0.2)	83.4	81.2	81.3	81.3 (±0.3)

Table 4: Performance of different classifiers on 5-fold cross-validation on BRT-dataset.

Supervised Classifiers. We used **logistic regression** as a classifier for our high-level features (§4) and for bag of words. Furthermore, we considered the **two transformers** RoBERTa (Liu et al., 2019) and BERTweet (Nguyen et al., 2020). The latter is specifically designed for Twitter. Since we do not want our classifiers to overfit, we did not tune the hyperparameters on BRT but took the settings from Nguyen et al. (2020) which are generally considered effective for Twitter.[10]

Experimental Set-up. The supervised classifiers were evaluated via 5-fold cross-validation. We **created the folds in two different ways**: On the one hand, we randomly split the tweets of BRT into 5 different folds (**randomized folds**). These folds may produce training and test splits where tweets of the same timeline occur both in the training and the test set. Within a particular timeline, biographically relevant tweets may co-occur with particular topics or constructions. These co-occurrences are idiosyncratic to the user to whom that timeline belongs. However, they may not be representative of the classes to be predicted. Supervised classifiers may produce high classification scores by memorizing these user-specific artefacts.

As a realistic alternative, we consider a different set-up in which the tweets of a particular timeline are restricted to the same fold (**unseen timelines**). As a consequence, on this set-up classifiers are not rewarded for learning user-specific artefacts since the tweets in a test set will always originate from timelines not observed in the training data.

As far as the transformers are concerned, we further report the average result over 5 different runs (including standard deviation). All other classifiers produce deterministic output.

Results. The results of our evaluation are displayed in Table 4. Next to accuracy, we report macro-average precision, recall and F-score. The simple majority-class baseline is outperformed by every other baseline by a large degree in terms of F-score. This even includes the aspectual classifier which suggests that (despite the noise in the automatic analysis we already reported in §5) aspectual information is not totally uncorrelated to biographical relevance. The most effective baseline in terms

[10]The settings are: batch size=32; no. of epochs=30; learning rate=1e-05. (We use *roberta-large* and *bertweet-large*.)

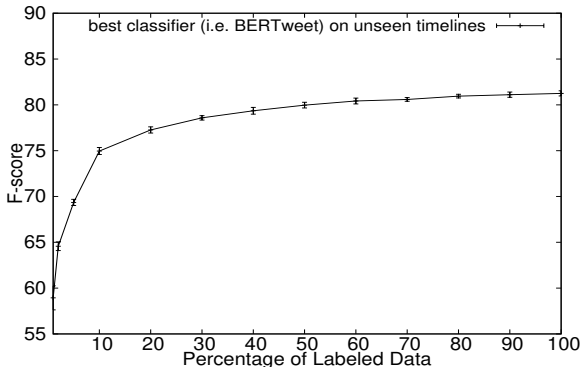

Figure 1: Learning curve on BRT-dataset.

of F-score is the classifier using meta-information. The supervised classifiers outperform the baselines. The most effective classifier is BERTweet, i.e. the language model that has been designed for Twitter.

Table 4 also contrasts the performance between *randomized folds* and *unseen timelines*. For all supervised classifiers, the evaluation based on randomized folds produces higher scores.[11] This confirms our assumption that randomizing folds produces overly optimistic results and that the classification on unseen timelines is a more realistic set-up and should be used in future evaluations.

Figure 1 shows the learning curve using our best performing classifier from Table 4 (i.e. BERTweet). With the full amount of the training data, the classifier seems to have reached a plateau. Therefore, we expect that additional training data will only result in marginal classification improvements.

7 Subtypes of Biographical Relevance

We extended our annotated BRT-dataset (§3) by an additional layer. For each biographically relevant tweet, we manually annotated the underlying subtype. Thus, we want to give an indication of what type of biographically relevant information can be found on Twitter. For the inventory of subtypes, we chose categories that are sufficiently distinctive and frequent on our data. Table 5 contains the distribution of the different subtypes as determined by our second layer of annotation. We also computed for each type the standard deviation across the different timelines. Only a high proportion with a low standard deviation indicates that the particular subtype is sufficiently frequent and can be consistently observed throughout the different timelines.

Table 5 shows that frequent biographically relevant subtypes are those that reveal likes of the author, other demographic information, family relations and instances that relate to job and education. From these subtypes, other demographic information has the lowest standard deviation, which means that this subtype can be more regularly observed across the different timelines at the same high frequency. Ideological views and habits occur similarly frequent but the standard deviation of ideological views is almost three times as high as the one of habits. This means that the latter occurs more regularly across the timelines.

8 Error Analysis

Despite the strong performance of our best classifier, i.e. BERTweet, we still observed a set of cases that were systematically classified incorrectly. A frequent error source stems from diverse forms of overgeneralization. While according to our annotation guidelines, desires and (dis)likes are considered biographically relevant, trivial desires (24) and (dis)likes (25) are not. Our classifier, however, fails to make this distinction. Similarly, while long-term health-conditions are biographically relevant (e.g. asthma, diabetes), common temporary illnesses are not (26). Our classifier also fails to make this distinction. Moreover, it is not able to appropriately take aspectual information into consideration. Episodic needs and desires (27) are thus erroneously considered biographically relevant.

(24) I just want some happiness *(irrelevant since everyone seeks happiness)*
(25) I Fuckin hate hiccups *(irrelevant since having a hiccup is generally considered an unpleasant experience)*
(26) Damn I have a cold again! *(irrelevant since a cold is a common illness everyone attracts now and then)*
(27) I need to get some sleep. *(irrelevant since this need/desire is only episodic)*

Finally, an obvious challenge are instances of implicit realizations of biographical relevance where there are no obvious lexical clues that suggest the presence of this category (28)-(29).

(28) Look at what arrived, my official Biden-Harris pins! *(inference: author is a Biden-Harris supporter)*
(29) He's fecking fat. Someone is feeding him. One of the neighbours. *(inference: author owns a cat)*

[11] It may come as a surprise that for the non-supervised classifiers the results between *randomized folds* and *unseen timelines* differ slightly. As a matter of fact, these classifiers do not change their prediction per tweet. These differences are the result of averaging over test folds whose size varies to a small degree between *unseen timelines* and *randomized folds*.

Subtype	Prototypical Example	Percent (std)
general likes	@USER I love Rage Against the Machine	20.6 (±12.4)
other demographic info.	I'm 6 foot 3. I stood next to a co worker today and I'm barely up to his neck	19.5 (±6.5)
job and education	@USER I'm a health worker There are thousands of us	18.4 (±13.3)
family relations	@USER Ooh my dead husband would have loved this	16.9 (±10.2)
general dislikes	i don't really like halloween...	16.5 (±14.1)
ideological views	@USER I think the future of humanity is very much in peril already	10.8 (±16.3)
habits	I refuse to see anyone before 11 Even though I am up before 9 and usually 8	10.4 (±5.3)
desires and needs	Want to move to a beach town in a few years	8.2 (±6.5)
food and drink preferences	@USER Ever eaten frogs They are amazing but we eat them too A tad more sentient than oysters	8.1 (±11.6)
health	@USER Omg I have a vasectomy! I'm not a male anymore apparently	5.4 (±4.2)
personality traits	@USER Yes I'm a sociable person. I miss conversation and dinner with someone who just might find me interesting...	3.9 (±2.8)
confidential information	Typing my surname earlier and it somehow suggests I'm 'Sillier' not Silbiger	2.8 (±3.8)
life events	@USER I moved to Georgia	1.6 (±1.5)
other	*not applicable*	6.2 (±4.4)

Table 5: Distribution of subtypes of biographical relevance. *(The same tweet may contain more than one subtype. Therefore, the sum of the percentages adds to more than 100%.)*

9 Conclusion

We presented a new dataset comprising tweets for the novel task of detecting biographically relevant utterances. Unlike previous research we do not restrict biographical relevance to a set of pre-defined relations. We showed that state-of-the-art classifiers are able to automatically detect such utterances even on tweets of unseen timelines. Our feature analysis using high-level features revealed the relatedness between linguistic features and biographical relevance (e.g. specific supersenses and POS-tags) while there is also a set of different meta features predictive for biographical irrelevance.

10 Ethical Considerations

The data we are going to make publicly available as part of this research will include tweets from specific timelines of Twitter. In order to protect the privacy rights of the authors, the usernames of the respective timelines have been anonymized by replacing each name by some generic ID. Moreover, we just release the IDs of the tweets contained in the timelines rather than the tweets themselves. The public release of such content as in the range of our dataset is also in accordance with the regulations of Twitter.

The manual annotation for our novel dataset was produced by a member of the department of one of the co-authors. The annotation was carried out as part of their regular work. Therefore, the work has been duly compensated.

The work presented in this paper addresses the task of extracting biographically relevant material published by ordinary people on Twitter. We are fully aware that research on this topic could be misused for the development of applications that we consider questionable from an ethical point of view. For instance, an application could be built that enables potential employers to do background checks on job seekers. Or, insurance companies could screen potential insurants before offering them a particular insurance. In §1, we acknowledged that potential of such research. However, we also made it clear that, in no way, this reflects the motivation of our research. On the contrary, we could envisage scenarios that would help users to protect the above misuse of their personal data. For instance, the functionality to identify biographically relevant information in user-generated content may be incorporated in an application that automatically anonymizes such sensitive text passages or, at least, warns users prior to posting such content that they are about to disclose sensitive material. Our work is also intended to raise public awareness of what contents can be found on Twitter and can be automatically and legally extracted.

11 Acknowledgements

The authors would like to thank Sybille Sornig for contributing to the manual annotation of our novel dataset.

References

Francesco Barbieri, Jose Camacho-Collados, Luis Espinosa-Anke, and Leonardo Neves. 2020. TweetEval: Unified Benchmark and Comparative Evaluation for Tweet Classification. In *Findings of Association for Computational Linguistics: EMNLP 2020*, 1644–1650, Online.

Fadi Biadsy, Julia Hirschberg, and Elena Filatova. 2008. An Unsupervised Approach to Biography Production

Using Wikipedia. In *Proceedings of the Annual Meeting of the Association for Computational Linguistics (ACL)*, pages 807–815, Columbus, OH, USA.

Gerardo Canfora, Andrea Di Sorbo, Enrico Emanuele, Sara Forootani, and Corrado A. Visaggio. 2018. A Nlp-based Solution to Prevent from Privacy Leaks in Social Network Posts. In *Proceedings of the International Conference on Availability, Reliability and Security*, pages 1–6, Hamburg, Germany.

Paolo Cappellari, Soon Ae Chun, and Mark Perelman. 2017. A Tool for Automatic Assessment and Awareness of Privacy Disclosure. In *Proceedings of the Annual International Conference on Digital Government Research*, pages 586–587, Staten Island, NY, USA.

Thomas Dickinson, Miriam Fernández, Lisa A. Thomas, Paul Mulholland, Pam Briggs, and Harith Alani. 2015. Identifying Prominent Life Events on Twitter. In *Proceedings of the International Conference on Knowledge Capture, K-CAP*, pages 4:1–4:8, Palisades, NY, USA.

Atefeh Farzindar and Diana Inkpen. 2015. Natural Language Processing for Social Media. In *Synthesis Lectures on Human Language Technologies*. Morgan & Claypool Publishers.

Annemarie Friedrich, Alexis Palmer, and Manfred Pinkal. 2016. Situation entity types: automatic classification of clause-level aspect. In *Proceedings of the Annual Meeting of the Association for Computational Linguistics (ACL)*, pages 1757–1768, Berlin, Germany.

Annemarie Friedrich and Manfred Pinkal. 2015. Automatic recognition of habituals: a three-way classification of clausal aspect. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 2471–2481, Lisbon, Portugal.

Marcos Garcia and Pablo Gamallo. 2015. Exploring the effectiveness of linguistic knowledge for biographical relation extraction. *Natural Language Engineering*, 21(4):519–551.

Nikesh Garera and David Yarowsky. 2009. Structural, Transitive and Latent Models for Biographic Fact Extraction. In *Proceedings of the Conference on European Chapter of the Association for Computational Linguistics (EACL)*, pages 300–308, Athens, Greece.

Ian Goodyer. 2001. *The depressed child and adolescent*, chapter Life events: Their nature and effects. Cambridge University Press.

Ozan Irsoy and Claire Cardie. 2017. Deep Recursive Neural Networks for Compositionality in Language. In *Proceedings of the Conference on Neural Information Processing Systems (NIPS)*, pages 2096–2104, Montréal, Canada.

Heng Ji, Ralph Grishman, Hoa Trang Dang, Kira Griffitt, and Joe Ellis. 2010. Overview of the TAC KBP 2010 Knowledge Base Population Track. In *Proceedings of the Text Analytics Conference (TAC)*, Gaithersburg, MD, USA.

J. Richard Landis and Gary G. Koch. 1977. The Measurement of Observer Agreement for Categorical Data. *Biometrics*, 33(1):159–174.

Jiwei Li and Claire Cardie. 2014. Timeline Generation: Tracking individuals on Twitter. In *Proceedings of the International Conference on World Wide Web (WWW)*, pages 643–652, Seoul, Korea.

Jiwei Li, Alan Ritter, Claire Cardie, and Eduard Hovy. 2014. Major Life Event Extraction from Twitter based on Congratulations/Condolences Speech Acts. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1997–2007, Doha, Qatar.

Pierre Lison, Ildikó Pilán, David Sanchez, Montserrat Batet, and Lilja Øvrelid. 2021. Anonymisation Models for Text Data: State of the art, Challenges and Future Directions. In *Proceedings of the Annual Meeting of the Association for Computational Linguistics (ACL)*, pages 4188–4203, Online.

Bing Liu. 2012. Sentiment Analysis and Opinion Mining. In *Synthesis Lectures on Human Lanugage Technologies*, volume 5, pages 1–167. Morgan & Claypool Publishers.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. RoBERTa: A Robustly Optimized BERT Pretraining Approach. *arXiv preprint arXiv:1907.11692*.

Delfina Malandrino, Andrea Petta, Vittorio Scarano, Luigi Serra, Raffaele Spinelli, and Balachander Krishnamurthy. 2013. Privacy Awareness about Information Leakage: Who knows what about me. In *Proceedings of the ACM Workshop on Privacy in the Electronic Society (WPES)*, pages 279–284, Berlin, Germany.

Huina Mao, Xin Shuai, and Apu Kapadia. 2011. Loose Tweets: An Analysis of Privacy Leaks on Twitter. In *Proceedings of the ACM Workshop on Privacy in the Electronic Society (WPES)*, pages 1–12, Chicago, IL, USA.

Ben Medlock. 2006. An Introduction to NLP-based Textual Anonymisation. In *Proceedings of the Conference on Language Resources and Evaluation (LREC)*, pages 1051–1056, Genoa, Italy.

George Miller, Richard Beckwith, Christiane Fellbaum, Derek Gross, and Katherine Miller. 1990. Introduction to WordNet: An On-line Lexical Database. *International Journal of Lexicography*, 3:235–244.

Saif Mohammad and Peter Turney. 2013. Crowdsourcing a Word-Emotion Association Lexicon. *Computational Intelligence*, 39(3):555–590.

Jan Neerbek, Ira Assent, and Peter Dolog. 2017. TABOO: Detecting unstructured sensitive information using recursive neural networks. In *Proceedings of the IEEE International Conference on Data Engineering*, pages 1399–1400, Chania, Greece.

Dat Quoc Nguyen, Thanh Vu, and Anh Tuan Nguyen. 2020. BERTweet: A pre-trained language model for English Tweets. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP): System Demonstrations*, pages 9–14, Online.

Olutobi Owoputi, Brendan O'Connor, Chris Dyer, Kevin Gimpel, and Nathan Schneider. 2012. Part-of-Speech Tagging for Twitter: Word Clusters and Other Advances. Technical Report CMU-ML-12-107, Machine Learning Department, Carnegie Mellon University.

Alistair Plum, Marcos Zampieri, Constantin Orasan, Eveline Wandl-Vogt, and Ruslan Mitkov. 2019. Large-scale Data Harvesting for Biographical Data. In *Proceedings of the Conference of Biographical Data in a Digital World*, pages 66–72, Varna, Bulgaria.

Francisco Rangel, Paolo Rosso, Moshe Koppel, Efstathios Stamatatos, and Giacomo Inches. 2013. Overview of the Author Profiling Task at PAN 2013. In *Working Notes for CLEF 2013 Conference*, pages 352–365, Valencia, Spain.

Delip Rao, David Yarowsky, Abhishek Shreevats, and Manaswi Gupta. 2010. Classifying Latent User Attributes in Twitter. In *Proceedings of the International Workshop on Search and Mining User-Generated Contents (SMUC)*, pages 37–44, Toronto, ON, Canada.

Michaela Regneri, Alexander Koller, Josef Ruppenhofer, and Manfred Pinkal. 2011. Learning Script Participants from Unlabeled Data. In *Proceedings of Recent Advances in Natural Language Processing (RANLP)*, pages 463–470, Hissar, Bulgaria.

Francesco Ricci, Lior Rokach, and Bracha Shapira. 2011. *Recommender Systems Handbook*, chapter Recommender Systems: Introduction and Challenges. Springer.

Koustuv Saha, Jordyn Seybolt, Stephen M. Mattingly, Talayeh Aledavood, Chaitanya Konjeti, Gonzalo J. Martinez, Ted Grover, Gloria J. Mark, and Munmun De Choudhury. 2021. What Life Events are Disclosed on Social Media, How, When, and By Whom? In *Proceedings of the CHI Conference on Human Factors in Computing Systems*, pages 1–22, Yokohama, Japan.

Satoshi Sekine and Javier Artiles. 2009. WePS2 Attribute Extraction Task. In *Proceedings of the Web People Search Evaluation Workshop (WePS)*.

Michael Wiegand, Josef Ruppenhofer, and Thomas Kleinbauer. 2019. Detection of Abusive Language: the Problem of Biased Datasets. In *Proceedings of the Human Language Technology Conference of the North American Chapter of the ACL (HLT/NAACL)*, pages 602–608, Minneapolis, MN, USA.

Xing Xie. 2010. Potential friend recommendation in online social network. In *Proceedings of the IEEE/ACM International Conference on Green Computing and Communications & International Conference on Cyber, Physical and Social Computing*, pages 831–835, Hangzhou, China.

Faiyaz Al Zamal, Wendy Liu, and Derek Ruths. 2012. Homophily and Latent Attribute Inference: Inferring Latent Attributes of Twitter Users from Neighbors. In *Proceedings of the International AAAI Conference on Weblogs and Social Media (ICWSM)*, pages 387–390, Dublin, Ireland.

Liang Zhou, Miruna Ticrea, and Eduard Hovy. 2004. Multi-Document Biography Summarization. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 434–441, Barcelona, Spain.

BECEL: Benchmark for Consistency Evaluation of Language Models

Myeongjun Jang[1] **Deuk Sin Kwon**[3] **Thomas Lukasiewicz**[2,1]
[1]Department of Computer Science, University of Oxford, UK
[2] Institute of Logic and Computation, TU Wien, Austria
firstname.lastname@cs.ox.ac.uk
[3]Language Super Intelligence Labs, SK Telecom, South Korea
kds0281@gmail.com

Abstract

Behavioural consistency is a critical condition for a language model (LM) to become trustworthy like humans. Despite its importance, however, there is little consensus on the definition of LM consistency, resulting in different definitions across many studies. In this paper, we first propose the idea of LM consistency based on *behavioural consistency* and establish a taxonomy that classifies previously studied consistencies into several sub-categories. Next, we create a new benchmark that allows us to evaluate a model on 19 test cases, distinguished by multiple types of consistency and diverse downstream tasks. Through extensive experiments on the new benchmark, we ascertain that none of the modern pre-trained language models (PLMs) performs well in every test case, while exhibiting high inconsistency in many cases. Our experimental results suggest that a unified benchmark that covers broad aspects (i.e., multiple consistency types and tasks) is essential for a more precise evaluation.

1 Introduction

Human-like behaviour is a critical property that increases a user's trust in an artificial intelligence (AI) agent (De Visser et al., 2016; Jung et al., 2019) by improving the certification process that ascertains whether a system behaves correctly (Huang et al., 2020). [1] Accordingly, despite the outstanding performance of transformer-based PLMs on natural language understanding (NLU) benchmarks, there have been pushbacks in various corners questioning their trustworthiness based on their non-human like behaviours, such as a poor memorisation effect on infrequent information (Kassner et al., 2020; Ravichander et al., 2020; Hofmann et al., 2021), insensitivity to sentence order (Pham et al., 2021; Gupta et al., 2021; Sinha et al., 2021), and a miserable understanding of negation expressions (Hossain et al., 2020; Kassner and Schütze, 2020; Ettinger, 2020; Hosseini et al., 2021; Jang et al., 2022).

In this respect, *behavioural consistency*, a core property of humans, is an important characteristic for a model to be deemed as trustworthy LM. Accordingly, the concept of consistency has been widely discussed in natural language processing (NLP). However, despite its prominent importance, there is little consensus on the precise definition of consistency. Below are examples of different consistency definitions:

- Making consistent decisions in semantically equivalent contexts (Elazar et al., 2021).
- Being consistent on a system's beliefs across various inputs (Li et al., 2019).
- Producing logically or factually accurate statements (Li et al., 2020b).

Hence, different studies on consistency focused on diverging types of consistency but only on certain tasks, primarily natural language inference (NLI) and question answering (QA).

To this end, in this paper, we first define the consistency of an LM based on the concept of *behavioural consistency* and establish a taxonomy by systematically categorising previous works based on our definition. Next, we propose a new benchmark named **be**nchmark for **c**onsistency **e**valuation of **l**anguage models (**BECEL**), a unified dataset for evaluating an LM's consistency, which assesses multiple types of consistency on six different tasks: NLI, semantic textual similarity (STS), words-in-context (WiC), semantic analysis (SA), machine reading comprehension (MRC), and topic classification (TC). Finally, we conduct extensive experiments on our new benchmark to assess the consistent behaviour of widely used PLMs and draw the following meaningful insights.

1. We observe that none of the PLMs coherently shows a consistent behaviour in all test cases.

[1]Trustworthiness = Certification + Explanation (Huang et al., 2020).

Proceedings of the 29th International Conference on Computational Linguistics, pages 3680–3696
October 12–17, 2022.

2. Large-sized models do not necessarily perform better than small-sized models, suggesting that increasing the model size is not the solution to improve consistency.
3. Our experimental results accentuate the necessity of a unified benchmark that enables evaluations from multiple aspects (i.e., various consistency types and downstream tasks).
4. Artefacts in training data have more influence than model design, e.g., training objectives and model structures.

The data of this paper are available at https://github.com/MJ-Jang/BECEL.

2 Language Model's Consistency

Behavioural consistency refers to being consistent in behavioural patterns by adhering to the same principles.[2] Based on this notion, we define the *consistency of an LM* as its ability to make a coherent decision not contradictory to its belief.

This definition of consistency consists of two components. The first one is *belief*, which refers to what a model considers to be true. The second component is *principle*, the property that decides what a coherent decision is. Based on these two components, we classify the various types of consistency in the literature into three large categories: semantic, logical, and factual consistency.

2.1 Semantic Consistency

It is the nature of meaning-text theory (MTT) to consider that the correspondence between linguistic expressions (*text*) and semantic contents (*meaning*) is *many-to-many*, implying that the meaning can be given in different text forms (Mel'čuk and Žolkovskij, 1970; Milićević, 2006). In this regard, a model with a high level of NLU ability should capture the meaning in essence and make the same decisions in semantically identical texts considering the definition of "understanding language"[3], and this is the concept of semantic consistency. The belief and principle become a *model's predictions on semantically identical texts* and *semantic equivalence*, respectively. So, we define the *semantic consistency of an LM* as its ability to make the same decisions on semantically equivalent texts.

Semantic consistency is an indispensable property of LMs regardless of the tasks and data, since it originates from the meaning and the universal nature of language. It is probably the most widely used concept across many studies regarding an LM's consistency. Research on text adversarial attacks showed that several PLMs are susceptible to adversarial samples that are designed to convey a similar meaning to their original counterparts (Jin et al., 2020; Garg and Ramakrishnan, 2020; Li et al., 2020a; Ivgi and Berant, 2021; Li et al., 2021). Ribeiro et al. (2019) investigated semantic consistency of QA models by generating implications that must be *true* considering the model's answer on the original query. Other works observed a discrepancy in the masked language modelling (MLM) predictions of PLMs for queries where the object is replaced with its plural form (Ravichander et al., 2020) and paraphrased queries (Elazar et al., 2021). Also, recent studies introduced the semantic consistency to consistency regularisation for training LMs with improved inductive bias (Wang and Henao, 2021; Zheng et al., 2021; Kim et al., 2021).

2.2 Logical Consistency

Several NLP tasks require the fulfilment of a certain logical property. The predictions that violate this logical property are considered invalid. Therefore, the belief and principle become a *model's predictions regarding instances where the logical property holds* and a *logical property*. Hence, we define the *logical consistency of an LM* as its ability to make decisions without logical contradiction.

Logical consistency can be subdivided according to the required logical properties. Here, we outline four types of logical consistency: *negational*, *symmetric*, *transitive*, and *additive* consistency.

Negational consistency. The core property of negational consistency is the logical negation property (p is true $\Leftrightarrow \neg p$ is false; Aina et al. 2018). That is, an LM's predictions should be different for texts having the opposite meaning if the property holds. Several works observed that PLMs often generate MLM outputs that violate this property, e.g., generating the same predictions for queries like "Birds can [MASK]" and "Birds cannot [MASK]" (Kassner and Schütze, 2020; Ettinger, 2020; Jang et al., 2022). Asai and Hajishirzi (2020) used negational consistency for data augmentation to train QA models.

Symmetric consistency. Provided a function f takes two variables, a symmetric inference is defined as: $f(x, y) = f(y, x)$. Intuitively, this implies that an LM's prediction should be invariant to the input text swap for NLP tasks. Previous

[2] N., Sam M.S., Behavioral Consistency. PsychologyDictionary.org, April 7, 2013, [link].

[3] To focus on the meaning and not the text (Krashen, 1982).

works on symmetric consistency are conducted on the NLI task. Wang et al. (2019) suggested that symmetric consistency holds for instances having *contradiction* and *neutral* as a label, and investigated the change in accuracy after switching the premise and hypothesis. On the contrary, Li et al. (2019) claimed that the property holds if and only if the label is a *contradiction*. They evaluated the symmetric consistency of NLI models on newly constructed data from MS-COCO (Lin et al., 2014). Recently, Kumar and Joshi (2022) expanded the experiments from NLI to STS task and evaluated the consistency in more conservatively by measuring the confidence score difference.

Transitive consistency. Given the three predicates X, Y, and Z, transitive inference is represented as: $X \rightarrow Y \wedge Y \rightarrow Z$ then $X \rightarrow Z$ (Gazes et al., 2012; Asai and Hajishirzi, 2020). Li et al. (2019) applied this property to NLI task. Specifically, for the three related sentences P, H, and Z, they defined four transitive inference rules:

$$E(P,H) \wedge E(H,Z) \rightarrow E(P,Z), \quad (1)$$
$$E(P,H) \wedge C(H,Z) \rightarrow C(P,Z), \quad (2)$$
$$N(P,H) \wedge E(H,Z) \rightarrow \neg C(P,Z), \quad (3)$$
$$N(P,H) \wedge C(H,Z) \rightarrow \neg E(P,Z), \quad (4)$$

where E, N, and C denote entailment, neutral, and contradiction, respectively. They constructed a new evaluation set from MS-COCO (Lin et al., 2014) for assessing the transitive consistency of NLI models. In QA, Asai and Hajishirzi (2020) used the transitive property for augmenting training data by combining two questions (q_1, q_2) where the effect of q_1 is equal to the cause of q_2. (Lin and Ng, 2022) investigated PLMs' transitive inference ability on WordNet word senses and the **IS-A** relation, i.e., if A *is-a* B and B *is-a* C, then A *is-a* C. These works ascertained that PLMs do not fully obey the transitive property.

Additive consistency. We propose a new type of logical consistency that we call additive consistency. For a function f, additive inference is represented as: $f(x) = f(y) = c \rightarrow f(x+y) = c$, where c is a predicted label. For NLP tasks, additive consistency applies to any single-sentence classification task (e.g., SA and TC). Intuitively, this implies that if a model yields the same prediction for different sentences, then the prediction of the combined sentence should also be the same.

Specificity of logical consistency. It is worth mentioning that, unlike semantic consistency, logical consistency is a task- and data-specific condition. It is inapplicable to those where a certain logical property is invalid. For instance, negational consistency cannot be applied to TC. It is obvious that negating the sentence below that belongs to the *Sports* category does not change its category.

Tottenham forward Son Heung-min has signed a new four-year contract.

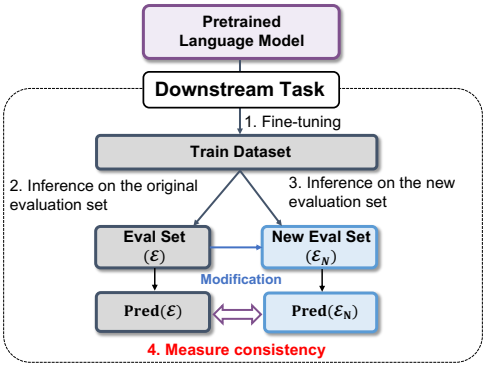

Figure 1: Overall evaluation framework for assessing an LM's consistency.

2.3 Factual Consistency

The basic concept of factual consistency is that a model should generate factually accurate outputs. Therefore, the belief is the *model's output*, and the principle becomes *factual correctness*. Hence, we define the *factual consistency of an LM* as its ability to generate outputs not contradictory to the common facts and given context.

By its nature of generating correct facts, factual consistency is closely related to *knowledge grounding* and *reducing hallucinations*. So, most works on factual consistency are on natural language generation (NLG), mainly text summarisation (Kryscinski et al., 2020; Maynez et al., 2020; Wang et al., 2020; Pagnoni et al., 2021), generative open-domain QA (Lewis et al., 2020b; Izacard and Grave, 2021), and dialogue generation (Li et al., 2020b; Shuster et al., 2021; Komeili et al., 2022).

3 BECEL Dataset

3.1 Overview

Figure 1 illustrates the overall framework for evaluating an LM's consistency. A PLM is fine-tuned on a downstream task and generates predictions of its original evaluation set ($\mathcal{E}$) and a new evaluation set ($\mathcal{E}_N$), specially designed from $\mathcal{E}$ to assess a certain type of consistency. Next, we compare the PLM's prediction on $\mathcal{E}$ and $\mathcal{E}_N$ to measure the consistency.

Our **BECEL** dataset contains $\mathcal{E}_N$ of multiple existing downstream tasks for assessing various con-

	BoolQ	SNLI	RTE	MRPC	WiC	SST2	AG-news
semantic	1,076	4,406	248	202	140	187	540
negational	401	2,204	153	290	-	-	-
symmetric	-	3,237	1,241	3,668	5,428	-	-
transitive	-	2,375	-	-	3,162	-	-
additive	-	-	-	-	-	53K	53K

Table 1: Number of new test data points for each downstream task and consistency type.

sistency types. It includes six downstream tasks: SNLI (Bowman et al., 2015) and RTE (Candela-Quinonero et al., 2006) for NLI, MRPC (Dolan and Brockett, 2005) for STS, BoolQ (Clark et al., 2019) for MRC, SST-2 (Socher et al., 2013) for SA, AG-News (Zhang et al., 2015) for TC and WiC (Pilehvar and Camacho-Collados, 2019), for evaluating semantic consistency and four types of logical consistencies.[4] Several data examples are in Figures 5 and 6 in Appendix B. We remove factual consistency from our evaluation scope, as benchmarks and evaluation frameworks for factual consistency are already well-studied across various tasks, such as summarisation (Kryscinski et al., 2020; Wang et al., 2020; Pagnoni et al., 2021), QA (Choi et al., 2018; Rajpurkar et al., 2018; Reddy et al., 2019), and dialogue generation (Dinan et al., 2019; Komeili et al., 2022).

Table 1 illustrates the size of the newly created $\mathcal{E}_N$ for each task and consistency type. In the case where a specific consistency cannot be applied to a particular task, it is excluded from the evaluation. The applicability of each consistency type to various tasks is described in Appendix A.2. In general, we use test sets as $\mathcal{E}$, provided gold labels are available. If not, development sets are used instead. However, training sets are used as $\mathcal{E}$, if two conditions are satisfied: (1) the size of the dev/test sets is small, and (2) new evaluation data can be collected automatically. Specifically, the RTE, MRPC, and WiC tasks for evaluating symmetric and transitive consistency belong to this case.

3.2 Data Collection Schema

Semantic consistency data. $\mathcal{E}_N$ for semantic consistency is a paraphrased version of $\mathcal{E}$. For all tasks, we paraphrase only one text input. Table 9 in Appendix illustrates each task's fixed and modified text inputs for creating $\mathcal{E}_N$. To collect paraphrase sentences, we use the publicly available Quilbot (https://quillbot.com/), as it can generate more natural paraphrases and cover broader linguistic variations compared to model-driven paraphrasing such as text adversarial attacks. In the WiC data, we remove a new data point if the target word does not exist in the paraphrased sentence. We then conduct a human evaluation through Amazon MTurk for the generated paraphrases to improve the data quality. Three annotators are allocated for each instance and asked to score the text similarity of the original and paraphrased sentences from 1 to 5. The instances where the average similarity score is not less than 4 are finally added to $\mathcal{E}_N$.

Negational consistency data. To collect $\mathcal{E}_N$ for negational consistency, we generate the opposite-meaning sentences of the modified variables listed in Table 9 by using two methods: *negation* and *antonym replacement*. For the former, we negate sentences having a single verb by inserting negation expressions like "not". For the latter, we extract adjectives and adverbs and replace only one word at a time with its antonym by using ConceptNet (Speer et al., 2017). Next, we perform the same human evaluation used in semantic consistency but select examples where the average similarity score does not exceed 2. Finally, we conduct a manual review on all instances to remove ambiguous or grammatically incorrect data points.

As mentioned earlier, negational consistency is data-specific. In SNLI, the label changes from "entailment" to "contradiction" if the hypothesis is switched with its opposite-meaning sentence. However, the label alteration is not guaranteed for the other labels, especially for "neutral". So, we only consider the "entailment" label to construct $\mathcal{E}_N$ of SNLI. For the same reason, we only use data points having the label "entailment" for RTE, "equivalent" for MRPC, and "true" for BoolQ to build $\mathcal{E}_N$.

Symmetric consistency data. We swap the text input order of tasks where the symmetric consistency is applicable. For WiC and MRPC, it is valid for every data point. Conversely, for NLI, it only applies to instances having "contradiction" as a label (Li et al., 2019) or "neutral" if the hypothesis is less specific than the premise (Wang et al., 2019). For RTE, we ascertain that the premise is always more specific than the hypothesis, and so data with "not_entailment" label are used to construct $\mathcal{E}_N$. However, it is not guaranteed in SNLI. So, only the data points with "contradiction" label are used.

Transitive consistency data. We construct $\mathcal{E}_N$ for transitive consistency on two tasks: SNLI and WiC. For SNLI, two data points must share the

[4] Brief descriptions of each downstream task are provided in Appendix A.1.

same hypothesis to apply the transitive inference rules described in Section 2.2, but only a premise is shared in the SNLI dataset. Hence, we leverage the symmetric consistency applicable to instances with the "contradiction" label, which enables us to transform the rules 3 and 4 as follows:

$$E(P,H) \wedge C(P,Z) \rightarrow C(H,Z),$$
$$N(P,H) \wedge C(P,Z) \rightarrow \neg E(H,Z).$$

By using the modified rules, we collect $\mathcal{E}_N$ for SNLI automatically. However, since the hypothesis is less specific than the premise in most cases in the SNLI data (Wang et al., 2019), we observe that the modified rules do not apply to several data points. Therefore, we conduct a human evaluation through Amazon MTurk to filter out such instances. Three annotators are allocated to each instance. We add examples to $\mathcal{E}_N$, provided at least two annotations comply with the rules.

For the WiC task, given a target word w and three predicates A, B, and C, the following transitive rules are applicable to every data point:

$$T(A,B|w) \wedge T(B,C|w) \rightarrow T(A,C|w),$$
$$T(A,B|w) \wedge F(B,C|w) \rightarrow F(A,C|w),$$
$$F(A,B|w) \wedge T(B,C|w) \rightarrow F(A,C|w),$$

where T/F implies that the meaning of the word w is used identically/differently in the given two sentences. We use these rules to collect $\mathcal{E}_N$ of WiC.

Additive consistency data. The additive consistency is valid for tasks that take a single-text input. To construct $\mathcal{E}_N$ for each task, we generate all possible combinations of two data points that share the same label and create a new one by merging their text inputs. Next, we remove a new data point if the token length of the merged text exceeds the 75% quantile of that of the training data, because such instances can be considered out-of-distributions that can overestimate the inconsistency issue.

3.3 Evaluation Metrics

Semantic/Symmetric consistency. Assume that $e_i \in \mathcal{E}$, $e_i^N \in \mathcal{E}_N$, and e_i^N is a perturbed version of e_i, and therefore, $|\mathcal{E}| = |\mathcal{E}_N|$. A model M should generate the same predictions for e_i and e_i^N. Therefore, by referencing the robust accuracy (Tsipras et al., 2019; Ivgi and Berant, 2021), we define the inconsistency metric (τ) for semantic and symmetric consistency as follows:

$$\tau = 1 - 1/|\mathcal{E}_N| \sum_{i=1}^{|\mathcal{E}_N|} \mathbb{1}(M(e_i) = M(e_i^N)).$$

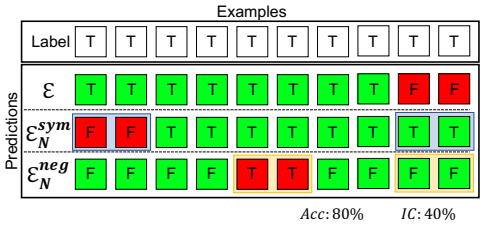

Figure 2: Graphical representation of accuracy and symmetric/negational consistency for binary classification. The blue and yellow boxes denote inconsistent cases for symmetric and negational consistency, respectively.

Negational consistency. Let $e_i \in \mathcal{E}$, $e_i^N \in \mathcal{E}_N$, and e_i^N is a perturbed version of e_i (i.e., $|\mathcal{E}| = |\mathcal{E}_N|$). Contrary to semantic and symmetric consistency, a model M should produce different predictions for e_i and e_i^N, where $e_i^N \in \mathcal{E}_N$ is a new instance designed for measuring negational consistency. Therefore, we define the inconsistency metric for negational consistency as follows:

$$\tau = 1 - 1/|\mathcal{E}_N| \sum_{i=1}^{|\mathcal{E}_N|} \mathbb{1}(M(e_i) \neq M(e_i^N)).$$

Transitive/Additive consistency. For both transitive and additive consistency, a new instance e_i^N is generated from two data points of $\mathcal{E}$. Assume that $e_i^N \in \mathcal{E}_N$, and e_i^N originates from $e_{i,1}, e_{i,2} \in \mathcal{E}$. Including e_i^N where the antecedent is not satisfied, i.e., a model M makes incorrect predictions for $e_{i,1}$ or $e_{i,2}$, can overestimate the inconsistency problem. Therefore, we use a conditional inconsistency as an evaluation metric:

$$\tau = 1 - 1/|\mathcal{C}| \sum_{i=1}^{|\mathcal{C}|} \mathbb{1}(M(c_i) = l_i),$$

where l_i is the label of $c_i \in \mathcal{C}$, and $\mathcal{C} \subset \mathcal{E}_N$ denotes the set of e_i where the model M makes correct predictions for both $e_{i,1}$ and $e_{i,2}$.

3.4 Importance of Measuring Consistency

Previous benchmarks regarding the opposite meanings (Naik et al., 2018; Hossain et al., 2020) or symmetry (Wang et al., 2019) only measure accuracy on the new test suit. It is true that models with low accuracy are likely to be inconsistent, but the high accuracy does not necessarily guarantee high consistency. Figure 2 well illustrates an example case. Although the accuracy is 80% in the original test set and two types of $\mathcal{E}_N$, implying that the model is quite robust on unseen data, the inconsistency is 40%. Therefore, consistency should be treated as an independent evaluation metric.

Model		BoolQ		MRPC			RTE			SNLI				SST2		WiC			AG-News	
		τ_{sem}	τ_{neg}	τ_{sem}	τ_{neg}	τ_{sym}	τ_{sem}	τ_{neg}	τ_{sym}	τ_{sem}	τ_{neg}	τ_{sym}	τ_{trn}	τ_{sem}	τ_{add}	τ_{sem}	τ_{sym}	τ_{trn}	τ_{sem}	τ_{add}
BERT	base	20.5	87.2	16.6	90.3	7.6	15.8	76.9	17.8	11.0	15.9	12.1	4.0	5.2	0.2	7.1	8.9	46.8	2.8	1.6
	large	16.5	77.3	12.5	90.8	6.8	12.3	75.8	15.8	9.9	11.7	10.2	3.6	3.3	0.1	8.4	7.0	49.3	3.0	1.7
RoBERTa	base	13.5	43.5	13.2	83.5	4.7	12.8	56.9	18.6	9.6	9.5	9.3	3.3	4.5	0.1	10.1	6.9	50.8	3.1	3.1
	large	10.2	40.8	8.4	84.2	4.3	9.8	24.6	11.6	7.9	5.9	9.7	3.5	2.3	0.1	9.3	7.3	46.6	2.7	1.1
Elelctra	base	7.1	63.7	8.8	86.6	7.1	9.4	32.8	9.8	9.2	7.7	9.5	3.3	3.0	0.0	10.1	5.1	48.0	2.8	2.4
	large	6.8	42.3	5.5	77.0	5.3	8.9	17.3	6.7	7.9	5.4	6.4	2.5	4.0	0.1	8.9	7.9	46.5	2.6	1.0
ERNIE2.0	base	13.3	62.4	6.3	79.6	6.6	13.2	35.0	13.2	10.1	13.2	9.5	3.3	5.2	0.1	5.1	5.1	51.1	3.2	2.7
	large	7.6	66.8	7.3	62.7	6.4	9.8	37.1	22.8	9.0	7.5	7.3	3.0	3.5	0.0	9.0	6.9	46.7	3.5	1.7
GPT2	base	12.8	85.8	18.4	87.2	14.5	18.1	75.3	33.3	16.3	30.0	23.0	10.4	19.6	0.8	14.1	13.1	47.4	2.7	2.2
	large	23.3	75.3	14.6	89.5	10.6	13.9	52.3	15.8	11.5	13.9	12.0	4.9	6.2	0.1	13.4	12.5	49.8	3.0	4.3
BART	base	13.4	71.2	12.2	84.4	5.6	11.4	70.5	18.3	10.8	10.9	14.4	4.7	4.7	0.1	8.7	7.7	53.0	3.0	3.5
	large	7.9	58.2	11.4	82.2	4.6	10.2	29.7	27.2	8.7	6.6	7.5	2.8	3.0	0.1	6.9	5.4	53.1	2.5	3.4
T5	base	12.9	29.4	8.1	39.8	3.7	11.7	18.5	16.8	10.9	7.2	10.6	3.6	4.1	0.2	16.0	7.9	46.3	2.1	0.3
	large	10.9	19.7	4.5	25.2	4.2	8.6	15.9	8.0	9.3	5.8	8.3	2.9	3.0	0.1	8.6	6.3	45.3	1.7	0.2

Table 2: The average of semantic (τ_{sem}), negational (τ_{neg}), symmetric (τ_{sym}), transitive, (τ_{trn}), and additive inconsistency (τ_{add}). All the metrics are lower the better. We repeat each experiments for five times.

4 Experiments and Analysis

4.1 Experimental Design

Model candidates. We evaluated the consistency of the below widely used PLMs (both *base* and *large* size models) on our new benchmark suit.

- **Encoder models**: BERT (Devlin et al., 2019), RoBERTa (Liu et al., 2019), Electra (Clark et al., 2020), and ERNIE 2.0 (Sun et al., 2020).
- **Decoder models**: GPT2 (Radford et al., 2019).
- **Encoder-Decoder models**: BART (Lewis et al., 2020a) and T5 (Raffel et al., 2020).

Training details. At fine-tuning, we use AdamW optimiser (Loshchilov and Hutter, 2017) and a linear learning rate scheduler decaying from 1e-3. All models are trained for 10 epochs, and the early stopping method is used during the training. The batch size and learning rate are different across model size and tasks. See Appendix A.3 for more details. For T5, we apply text-to-text multitask training by using the free-text input format used by Raffel et al. (2020). We repeat the experiments for each model and task for five times and report their average values. Our best validation performance is almost close to the reported results in previous works (see Table 6 in the appendix).

4.2 Semantic Consistency Results

The results are in Table 2. We ascertain that PLMs show a different consistency across diverse tasks. Specifically, τ_{sem} is extremely low in the SST2 and AG-News tasks. We conjecture that a leading cause is that these tasks have a high correlation between labels and certain words, such as sentiment words and proper nouns, and therefore, are hardly affected by paraphrases. Among the other tasks, the PLMs are relatively more consistent in MRPC than the others but still make many mistakes considering that the STS task is designed to focus on semantic equivalence. We also observe that GPT2 and BERT are highly inconsistent. T5 and Electra show the lowest τ_{sem}, but the difference to the others, apart from GPT2 and BERT, are marginal. The results suggest that a model's training objective somewhat affects its semantic consistency.

	BoolQ	MRPC	SST2	RTE
BAE	12.4 (-1.1)	7.9 (-5.3)*	5.9 (+3.6)*	11.7 (-1.1)
TextFooler	11.2 (-2.3)*	8.9* (-6.7)*	6.4 (+4.2)*	11.3 (-1.5)

Table 3: The inconsistency results of the adversarial training experiments. The value written in parenthesis is the difference compared to the original RoBERTa-base model. The difference is statistically significant with p value < 0.05 (*).

Can adversarial training be a solution? Adversarial training is widely used to improve robustness by providing models with original and adversarial samples (Jin et al., 2020). We investigate whether it is beneficial to improve semantic consistency. We apply two text attack methods, BAE (Garg and Ramakrishnan, 2020) and TextFooler (Jin et al., 2020), to the RoBERTa-base model by using TextAttack (Morris et al., 2020). Five adversarial samples are generated for each data point.

The results are in Table 3. We confirm that adversarial training is not always beneficial. The improvement is marginal, except for MRPC and even backfired in SST2. We speculate that a leading cause is that the attack methods are likely to generate incorrect paraphrase sentences (see Appendix 10 for examples). Moreover, adversarial training is vulnerable to instances that the attack

method cannot generate. It has been observed that about 45% of inconsistent predictions contain examples that have different sentence structures (e.g., changing active to passive), which synonym-replacement-based methods like BAE and TextFooler are unable to produce. The results suggest that adversarial training cannot be an ultimate solution to improve semantic consistency.

4.3 Negational Consistency Results

Table 2 presents the results of the negational consistency experiments. It is astonishing that τ_{neg} is very high across all tasks apart from SNLI, suggesting that the fine-tuned PLMs entirely fail to understand the opposite meaning. For SNLI, we strongly believe that the leading cause of low τ_{neg} are superficial cues in the training data. It is well known that there is a strong correlation between negation expressions and "contradiction" labels in the SNLI data (Gururangan et al., 2018), and we confirm that almost 68% of training instances with negation expressions in the hypothesis have "contradiction" labels. So, achieving high consistency in SNLI is easy, as our new evaluation set originates from instances with "entailment" labels, as illustrated in Section 3.2. The relatively low τ_{neg} of the T5 models, which can benefit from the SNLI data through multi-task training, also support our claim.

Model design vs. superficial cues. Similarly to the semantic consistency experiments, GPT2 and BERT perform worst in general. To compare the impact of model designs (e.g., training objectives, model structure) and superficial cues, we use the following metric for the model M on the task T:

$$\rho_T^M = (\tau_T^M - \tau_{SNLI}^M)/(\tau_T^M - \tau_T^*),$$

where τ_T^M implies the negational inconsistency of the model M on the task T. τ_T^* denotes the best inconsistency of task T among similar-size models (e.g., *base*). Intuitively, the metric implies that the performance gap with SNLI (i.e., effect of superficial cues) is ρ times higher than that with the best PLM (i.e., effect of model designs).

We measure ρ of BERT, GPT2, and BART, because their performance does not rank at the top across all tasks (ρ becomes larger if the model's inconsistency is close to the best performance). Single-task trained models are considered for deciding the best performance. The results are in Table 4. We observe that ρ is greater than 1 in every case. RTE has relatively low values, as it shares the same superficial cues with SNLI, but their total amount is much less. This suggests that superficial cues have a greater effect than model designs.

Model	BoolQ		MRPC		RTE	
	base	large	base	large	base	large
BERT	1.63	1.80	6.95	2.81	1.38	1.10
GPT2	1.32	1.78	7.53	2.82	1.07	1.10
BART	2.18	1.99	15.31	3.09	1.58	1.31

Table 4: ρ values of BERT, GPT2, and BART.

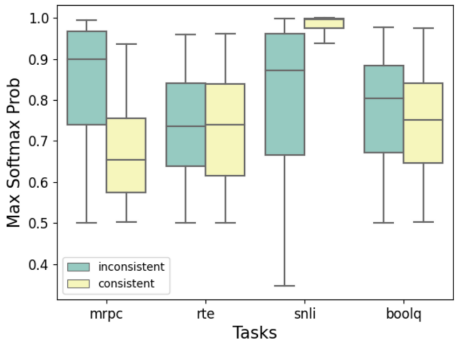

Figure 3: Box plot of maximum softmax probability of RoBERTa-base for negational consistency experiments.

Overconfident inconsistent predictions. Negational inconsistency would be less concerning, if the predictions are made by change (i.e., high entropy). However, we observe that models are very confident regarding their inconsistent decisions, generating similar or higher softmax probabilities than the consistent predictions in most cases (see, e.g., Figure 3). The confidence score seems reasonable only in the SNLI task, which contains superficial cues. The results suggest that fine-tuned PLMs are hard to trust, considering their overconfidence in incorrect and inconsistent predictions.

4.4 Symmetric Consistency Results

Table 2 shows the experimental results of symmetric consistency. In terms of the model, GPT2 again, performs worst in most cases, implying that decoder-only auto-regressive models are not suitable for achieving high consistency. The inconsistency is not significantly different for the other models, but Electra outperforms the others in general.

Compared to the NLI tasks, the inconsistency is much lower in WiC and MRPC, which are designed to focus on semantic equivalence, suggesting that achieving high symmetric consistency might be possible by making PLMs capturing the latent meaning of the texts. Although the inconsistency is fairly low, it should not be overlooked, because symmetry is an uncomplicated property that requires a simple reasoning ability. For this reason, humans are likely to show an extremely low inconsistency. We conduct a brief human evaluation on

the MRPC task by asking five human annotators 30 questions each and observe that humans are highly consistent on symmetry, achieving $\tau_{sym} = 0.7$.

4.5 Transitive Consistency Results

The transitive consistency results are in Table 2. Interestingly, they are entirely different in the two tasks. In the SNLI task, which is designed to infer the logical relationship between two given sentences, all PLMs show a strong performance. However, in the WiC task, which focuses on the word's meaning, the inconsistency is very low even though the evaluation data originate from the training set. The results suggest that the transitive reasoning ability is highly contingent on the purpose of downstream tasks.

Does the training data size matter? SNLI and WiC have two major differences: (1) task objective and (2) data size (i.e., approximately 500K and 6K for SNLI and WiC, respectively). To ascertain whether more training data help achieving a high consistency, we conduct an additional experiment by down-sampling the training data size of SNLI to 6K. The Electra models that record the best τ_{trn} are used for this experiment. The results are in Table 5. The inconsistency increases after the down-sampling, but is still lower than that of WiC, and the validation accuracy is impaired, especially in the base-size model. The results suggest that small training data can cause high inconsistency, as a model becomes less accurate, but the task objective affects much more than the training data size.

4.6 Additive Consistency Results

It is noteworthy that this experiment is a very easy task, because the input is a combination of two sentences that belong to the same category, so the model has more evidence to make the correct decision. The results of the additive consistency experiments are in Table 2. All the PLMs are highly consistent in SST2 but make some mistakes in AG-News except for the T5 models. The average τ_{add} of 2.3 in AG-News is not a low score considering the task difficulty. To become trustworthy, PLMs need to be more consistent on the additive property.

5 Discussion

Are large models more consistent? It is well-known that large-size models consistently outperform small-size models in terms of accuracy. Does the same trend occur from a consistency perspective? Figure 4 illustrates the portion of the three cases: the performance of the large models are better, worse, or show no statistical difference. Interestingly, the case where there is no statistical difference in consistency between the large- and base-size models accounts for a large portion, and sometimes base-size models even perform better. This pattern is hardly seen in accuracy-based evaluation metrics, suggesting that additional evaluation metrics such as consistency other than the accuracy should be considered for a precise evaluation.

Model		SNLI		SNLI-6K		WiC	
		$\mathcal{A}_{val}$	τ_{trn}	$\mathcal{A}_{val}$	τ_{trn}	$\mathcal{A}_{tr}$	τ_{trn}
Electra	base	91.8	3.3	64.8	10.0	81.6	48.0
	large	93.5	2.5	85.6	3.3	80.7	46.5

Table 5: Results of the down-sampled SNLI experiments. $\mathcal{A}_{val}$ and $\mathcal{A}_{tr}$ denote the validation and training accuracy, respectively. We report $\mathcal{A}_{tr}$ of WiC, because its $\mathcal{E}_N$ originates from the training data.

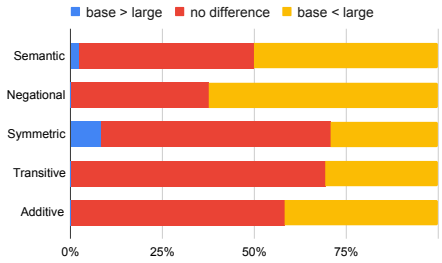

Figure 4: Portion of experimental cases where the large-size models are more or less consistent than the base-size models. A t-test under the significance level of 0.05 confirmed the statistical difference in performance.

Necessity of a unified benchmark. Our experimental results highlight the importance of evaluating models in a wide spectrum. We verify that none of the PLMs performs coherently well in every experiment, suggesting that focusing on a certain task or consistency type contains the risk of reaching a wrong conclusion. For instance, we might conclude that PLMs are fairly consistent if we only consider semantic consistency. If we conduct experiments only in the NLI tasks like extant studies, the conclusion might be distorted, since the results of all inconsistency types seem reasonable in the NLI tasks, especially in SNLI. Our new dataset, however, prevents us from drawing such a fallacious conclusion by allowing us to assess models across multiple consistency types and tasks, demonstrating its importance to have a unified benchmark covering a wide array of topics including different evaluation criteria and task types.

Uncontrollable AI. Due to the nature of inductive reasoning, the inductive bias of machine learning and deep learning models is greatly affected by the patterns in the training data. Although this is well-

known and widely accepted (Alzubi et al., 2018; Katsaros et al., 2019; Anagnostis et al., 2020; Xu et al., 2020; Thielen et al., 2020; Ma et al., 2021), our experimental results show that the artefacts in data are a more influential factor than the model design in deciding its inductive bias (Section 4.3). The problem is that we have a control over the model design but not the artefacts, as it is difficult to review and manipulate all data points with an enormous size. This evokes a critical concern: *uncontrollable AI*. However elaborate the model that we design with highly advanced training objectives and model structures, we might not have a full control over the model, as the ungovernable effect of the artefacts in data remains. It is thus imperative to take appropriate actions to address the data-driven faulty behaviour of the model, such as the generation of ethically problematic outputs (Nangia et al., 2020). To overcome such issues and move forward to developing more trustworthy and safer AI, perhaps it is time to think beyond inductive reasoning.

6 Summary and Outlook

In this work, we first defined LM consistency based on the concept of behavioural consistency: a core property that a sound LM should obey. Next, we categorised various previous studies regarding consistency into three types: semantic, logical, and factual consistency. Finally, we designed a benchmark suite to assess various types of consistency on multiple downstream tasks.

Through extensive experiments, we observed that none of the PLMs shows perfectly consistent outputs in all test cases. Our experimental results highlight the essence of evaluation schema in multiple spectrums to avoid reaching a distorted conclusion. We also revealed that the impact of spurious artefacts presented in training data is greater than that of model design, such as model size and learning objective. This finding raises concerns about uncontrollable AI, as we have no control over the artefacts in tremendous amounts of data. Our work suggests that we should probably go beyond neural models, which only allow inductive reasoning, to develop trustworthy and safe AI.

Acknowledgements

This work was partially supported by the Alan Turing Institute under the EPSRC grant EP/N510129/1 and by the AXA Research Fund. We also acknowledge the use of Oxford's ARC facility, of the EPSRC-funded Tier 2 facility JADE II (EP/T022205/1), and of GPU computing support by Scan Computers International Ltd.

References

Laura Aina, Raffaella Bernardi, and Raquel Fernández. 2018. A distributional study of negated adjectives and antonyms. In *CEUR Workshop Proceedings*, volume 2253.

Jafar Alzubi, Anand Nayyar, and Akshi Kumar. 2018. Machine learning from theory to algorithms: An overview. In *Journal of Physics: Conference Series*, volume 1142, page 012012. IOP Publishing.

Athanasios Anagnostis, Elpiniki Papageorgiou, and Dionysis Bochtis. 2020. Application of artificial neural networks for natural gas consumption forecasting. *Sustainability*, 12(16):6409.

Akari Asai and Hannaneh Hajishirzi. 2020. Logic-guided data augmentation and regularization for consistent question answering. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 5642–5650, Online. Association for Computational Linguistics.

Samuel R. Bowman, Gabor Angeli, Christopher Potts, and Christopher D. Manning. 2015. A large annotated corpus for learning natural language inference. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 632–642, Lisbon, Portugal. Association for Computational Linguistics.

Joaquin Candela-Quinonero, Ido Dagan, Bernardo Magnini, and Florence d'Alché Buc. 2006. Evaluating Predictive Uncertainty, Visual Objects Classification and Recognising textual entailment: Selected Proceedings of the First PASCAL Machine Learning Challenges Workshop.

Eunsol Choi, He He, Mohit Iyyer, Mark Yatskar, Wentau Yih, Yejin Choi, Percy Liang, and Luke Zettlemoyer. 2018. QuAC: Question answering in context. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 2174–2184, Brussels, Belgium. Association for Computational Linguistics.

Christopher Clark, Kenton Lee, Ming-Wei Chang, Tom Kwiatkowski, Michael Collins, and Kristina Toutanova. 2019. BoolQ: Exploring the surprising difficulty of natural yes/no questions. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 2924–2936, Minneapolis, Minnesota. Association for Computational Linguistics.

Kevin Clark, Minh-Thang Luong, Quoc V. Le, and Christopher D. Manning. 2020. Electra: Pre-training text encoders as discriminators rather than generators.

In *International Conference on Learning Representations*.

Ewart J. De Visser, Samuel S. Monfort, Ryan McKendrick, Melissa A. B. Smith, Patrick E. McKnight, Frank Krueger, and Raja Parasuraman. 2016. Almost human: Anthropomorphism increases trust resilience in cognitive agents. *Journal of Experimental Psychology: Applied*, 22(3):331.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186.

Emily Dinan, Stephen Roller, Kurt Shuster, Angela Fan, Michael Auli, and Jason Weston. 2019. Wizard of Wikipedia: Knowledge-powered conversational agents. In *International Conference on Learning Representations*.

William B. Dolan and Chris Brockett. 2005. Automatically constructing a corpus of sentential paraphrases. In *Proceedings of the Third International Workshop on Paraphrasing (IWP2005)*.

Yanai Elazar, Nora Kassner, Shauli Ravfogel, Abhilasha Ravichander, Eduard Hovy, Hinrich Schütze, and Yoav Goldberg. 2021. Erratum: Measuring and improving consistency in pretrained language models. *Transactions of the Association for Computational Linguistics*, 9:1407–1407.

Allyson Ettinger. 2020. What BERT is not: Lessons from a new suite of psycholinguistic diagnostics for language models. *Transactions of the Association for Computational Linguistics*, 8:34–48.

Siddhant Garg and Goutham Ramakrishnan. 2020. BAE: BERT-based adversarial examples for text classification. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 6174–6181, Online. Association for Computational Linguistics.

Regina Paxton Gazes, Nicholas W. Chee, and Robert R. Hampton. 2012. Cognitive mechanisms for transitive inference performance in rhesus monkeys: Measuring the influence of associative strength and inferred order. *Journal of Experimental Psychology: Animal Behavior Processes*, 38(4):331.

Ashim Gupta, Giorgi Kvernadze, and Vivek Srikumar. 2021. Bert & family eat word salad: Experiments with text understanding. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 35, pages 12946–12954.

Suchin Gururangan, Swabha Swayamdipta, Omer Levy, Roy Schwartz, Samuel Bowman, and Noah A. Smith. 2018. Annotation artifacts in natural language inference data. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 2 (Short Papers)*, pages 107–112, New Orleans, Louisiana. Association for Computational Linguistics.

Valentin Hofmann, Janet Pierrehumbert, and Hinrich Schütze. 2021. Superbizarre is not superb: Derivational morphology improves BERT's interpretation of complex words. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 3594–3608, Online. Association for Computational Linguistics.

Md Mosharaf Hossain, Venelin Kovatchev, Pranoy Dutta, Tiffany Kao, Elizabeth Wei, and Eduardo Blanco. 2020. An analysis of natural language inference benchmarks through the lens of negation. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 9106–9118, Online. Association for Computational Linguistics.

Arian Hosseini, Siva Reddy, Dzmitry Bahdanau, R Devon Hjelm, Alessandro Sordoni, and Aaron Courville. 2021. Understanding by understanding not: Modeling negation in language models. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 1301–1312, Online. Association for Computational Linguistics.

Xiaowei Huang, Daniel Kroening, Wenjie Ruan, James Sharp, Youcheng Sun, Emese Thamo, Min Wu, and Xinping Yi. 2020. A survey of safety and trustworthiness of deep neural networks: Verification, testing, adversarial attack and defence, and interpretability. *Computer Science Review*, 37:100270.

Maor Ivgi and Jonathan Berant. 2021. Achieving model robustness through discrete adversarial training. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 1529–1544, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Gautier Izacard and Édouard Grave. 2021. Leveraging passage retrieval with generative models for open domain question answering. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*, pages 874–880.

Myeongjun Jang, Frank Mtumbuka, and Thomas Lukasiewicz. 2022. Beyond distributional hypothesis: Let language models learn meaning-text correspondence. In *Findings of the Association for Computational Linguistics: NAACL 2022*, pages 2030–2042, Seattle, United States. Association for Computational Linguistics.

Di Jin, Zhijing Jin, Joey Tianyi Zhou, and Peter Szolovits. 2020. Is BERT really robust? A strong

baseline for natural language attack on text classification and entailment. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pages 8018–8025.

Eun-Soo Jung, Suh-Yeon Dong, and Soo-Young Lee. 2019. Neural correlates of variations in human trust in human-like machines during non-reciprocal interactions. *Scientific Reports*, 9(1):1–10.

Nora Kassner, Benno Krojer, and Hinrich Schütze. 2020. Are pretrained language models symbolic reasoners over knowledge? In *Proceedings of the 24th Conference on Computational Natural Language Learning*, pages 552–564, Online. Association for Computational Linguistics.

Nora Kassner and Hinrich Schütze. 2020. Negated and misprimed probes for pretrained language models: Birds can talk, but cannot fly. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7811–7818, Online. Association for Computational Linguistics.

Dimitrios Katsaros, George Stavropoulos, and Dimitrios Papakostas. 2019. Which machine learning paradigm for fake news detection? In *2019 IEEE/WIC/ACM International Conference on Web Intelligence (WI)*, pages 383–387. IEEE.

Gangwoo Kim, Hyunjae Kim, Jungsoo Park, and Jaewoo Kang. 2021. Learn to resolve conversational dependency: A consistency training framework for conversational question answering. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 6130–6141, Online. Association for Computational Linguistics.

Mojtaba Komeili, Kurt Shuster, and Jason Weston. 2022. Internet-augmented dialogue generation. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 8460–8478, Dublin, Ireland. Association for Computational Linguistics.

Stephen D. Krashen. 1982. *Principles and practice in second language acquisition*. Language Teaching Methodology Series. Pergamon, Oxford.

Wojciech Kryscinski, Bryan McCann, Caiming Xiong, and Richard Socher. 2020. Evaluating the factual consistency of abstractive text summarization. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 9332–9346, Online. Association for Computational Linguistics.

Ashutosh Kumar and Aditya Joshi. 2022. Striking a balance: Alleviating inconsistency in pre-trained models for symmetric classification tasks. In *Findings of the Association for Computational Linguistics: ACL 2022*, pages 1887–1895, Dublin, Ireland. Association for Computational Linguistics.

Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Veselin Stoyanov, and Luke Zettlemoyer. 2020a. BART: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7871–7880.

Patrick Lewis, Ethan Perez, Aleksandra Piktus, Fabio Petroni, Vladimir Karpukhin, Naman Goyal, Heinrich Küttler, Mike Lewis, Wen-tau Yih, Tim Rocktäschel, et al. 2020b. Retrieval-augmented generation for knowledge-intensive NLP tasks. *Advances in Neural Information Processing Systems*, 33:9459–9474.

Dianqi Li, Yizhe Zhang, Hao Peng, Liqun Chen, Chris Brockett, Ming-Ting Sun, and Bill Dolan. 2021. Contextualized perturbation for textual adversarial attack. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 5053–5069, Online. Association for Computational Linguistics.

Linyang Li, Ruotian Ma, Qipeng Guo, Xiangyang Xue, and Xipeng Qiu. 2020a. BERT-ATTACK: Adversarial attack against BERT using BERT. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 6193–6202, Online. Association for Computational Linguistics.

Margaret Li, Stephen Roller, Ilia Kulikov, Sean Welleck, Y-Lan Boureau, Kyunghyun Cho, and Jason Weston. 2020b. Don't say that! Making inconsistent dialogue unlikely with unlikelihood training. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 4715–4728, Online. Association for Computational Linguistics.

Tao Li, Vivek Gupta, Maitrey Mehta, and Vivek Srikumar. 2019. A logic-driven framework for consistency of neural models. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 3924–3935, Hong Kong, China. Association for Computational Linguistics.

Ruixi Lin and Hwee Tou Ng. 2022. Does BERT know that the IS-a relation is transitive? In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 94–99, Dublin, Ireland. Association for Computational Linguistics.

Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C. Lawrence Zitnick. 2014. Microsoft COCO: Common objects in context. In *European Conference on Computer Vision*, pages 740–755. Springer.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis,

Luke Zettlemoyer, and Veselin Stoyanov. 2019. RoBERTa: A robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692*.

Ilya Loshchilov and Frank Hutter. 2017. Fixing weight decay regularization in Adam. *ArXiv*, abs/1711.05101.

Zeyu Ma, Han Yu, Jinyao Xia, Chunzhi Wang, Lingyu Yan, and Xianjin Zhou. 2021. Network traffic prediction based on seq2seq model. In *2021 16th International Conference on Computer Science & Education (ICCSE)*, pages 710–713. IEEE.

Joshua Maynez, Shashi Narayan, Bernd Bohnet, and Ryan McDonald. 2020. On faithfulness and factuality in abstractive summarization. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 1906–1919, Online. Association for Computational Linguistics.

I. A. Mel'čuk and A. K. Žolkovskij. 1970. Towards a functioning 'meaning-text' model of language. *Linguistics*, 8(57):10–47.

Jasmina Milićević. 2006. A short guide to the meaning-text linguistic theory. *Journal of Koralex*, 8:187–233.

John Morris, Eli Lifland, Jin Yong Yoo, Jake Grigsby, Di Jin, and Yanjun Qi. 2020. TextAttack: A framework for adversarial attacks, data augmentation, and adversarial training in NLP. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 119–126, Online. Association for Computational Linguistics.

Aakanksha Naik, Abhilasha Ravichander, Norman Sadeh, Carolyn Rose, and Graham Neubig. 2018. Stress test evaluation for natural language inference. In *Proceedings of the 27th International Conference on Computational Linguistics*, pages 2340–2353, Santa Fe, New Mexico, USA. Association for Computational Linguistics.

Nikita Nangia, Clara Vania, Rasika Bhalerao, and Samuel R. Bowman. 2020. CrowS-pairs: A challenge dataset for measuring social biases in masked language models. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1953–1967, Online. Association for Computational Linguistics.

Artidoro Pagnoni, Vidhisha Balachandran, and Yulia Tsvetkov. 2021. Understanding factuality in abstractive summarization with FRANK: A benchmark for factuality metrics. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 4812–4829, Online. Association for Computational Linguistics.

Thang Pham, Trung Bui, Long Mai, and Anh Nguyen. 2021. Out of order: How important is the sequential order of words in a sentence in natural language understanding tasks? In *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*, pages 1145–1160, Online. Association for Computational Linguistics.

Mohammad Taher Pilehvar and Jose Camacho-Collados. 2019. WiC: the word-in-context dataset for evaluating context-sensitive meaning representations. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 1267–1273, Minneapolis, Minnesota. Association for Computational Linguistics.

Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. 2019. Language models are unsupervised multitask learners. *OpenAI blog*, 1(8):9.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J. Liu. 2020. Exploring the limits of transfer learning with a unified text-to-text transformer. *Journal of Machine Learning Research*, 21:1–67.

Pranav Rajpurkar, Robin Jia, and Percy Liang. 2018. Know what you don't know: Unanswerable questions for SQuAD. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 784–789, Melbourne, Australia. Association for Computational Linguistics.

Abhilasha Ravichander, Eduard Hovy, Kaheer Suleman, Adam Trischler, and Jackie Chi Kit Cheung. 2020. On the systematicity of probing contextualized word representations: The case of hypernymy in BERT. In *Proceedings of the Ninth Joint Conference on Lexical and Computational Semantics*, pages 88–102, Barcelona, Spain (Online). Association for Computational Linguistics.

Siva Reddy, Danqi Chen, and Christopher D. Manning. 2019. CoQA: A conversational question answering challenge. *Transactions of the Association for Computational Linguistics*, 7:249–266.

Marco Tulio Ribeiro, Carlos Guestrin, and Sameer Singh. 2019. Are red roses red? Evaluating consistency of question-answering models. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 6174–6184, Florence, Italy. Association for Computational Linguistics.

Kurt Shuster, Spencer Poff, Moya Chen, Douwe Kiela, and Jason Weston. 2021. Retrieval augmentation reduces hallucination in conversation. In *Findings of the Association for Computational Linguistics: EMNLP 2021*, pages 3784–3803, Punta Cana, Dominican Republic. Association for Computational Linguistics.

Koustuv Sinha, Prasanna Parthasarathi, Joelle Pineau, and Adina Williams. 2021. UnNatural Language Inference. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 7329–7346, Online. Association for Computational Linguistics.

Richard Socher, Alex Perelygin, Jean Wu, Jason Chuang, Christopher D. Manning, Andrew Ng, and Christopher Potts. 2013. Recursive deep models for semantic compositionality over a sentiment treebank. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*, pages 1631–1642, Seattle, Washington, USA. Association for Computational Linguistics.

Robyn Speer, Joshua Chin, and Catherine Havasi. 2017. Conceptnet 5.5: An open multilingual graph of general knowledge. In *Proceedings of the 31st AAAI Conference on Artificial Intelligence*.

Yu Sun, Shuohuan Wang, Yukun Li, Shikun Feng, Hao Tian, Hua Wu, and Haifeng Wang. 2020. ERNIE 2.0: A continual pre-training framework for language understanding. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pages 8968–8975.

Nils Thielen, Dominik Werner, Konstantin Schmidt, Reinhardt Seidel, Andreas Reinhardt, and Jörg Franke. 2020. A machine learning based approach to detect false calls in SMT manufacturing. In *2020 43rd International Spring Seminar on Electronics Technology (ISSE)*, pages 1–6. IEEE.

Dimitris Tsipras, Shibani Santurkar, Logan Engstrom, Alexander Turner, and Aleksander Madry. 2019. Robustness may be at odds with accuracy. In *International Conference on Learning Representations*.

Alex Wang, Kyunghyun Cho, and Mike Lewis. 2020. Asking and answering questions to evaluate the factual consistency of summaries. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 5008–5020, Online. Association for Computational Linguistics.

Haohan Wang, Da Sun, and Eric P Xing. 2019. What if we simply swap the two text fragments? a straightforward yet effective way to test the robustness of methods to confounding signals in nature language inference tasks. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, pages 7136–7143.

Rui Wang and Ricardo Henao. 2021. Unsupervised paraphrasing consistency training for low resource named entity recognition. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 5303–5308, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Jinghui Xu, Yu Wen, Chun Yang, and Dan Meng. 2020. An approach for poisoning attacks against RNN-based cyber anomaly detection. In *2020 IEEE 19th International Conference on Trust, Security and Privacy in Computing and Communications (TrustCom)*, pages 1680–1687. IEEE.

Xiang Zhang, Junbo Zhao, and Yann LeCun. 2015. Character-level convolutional networks for text classification. In *Advances in Neural Information Processing Systems*, volume 28. Curran Associates, Inc.

Bo Zheng, Li Dong, Shaohan Huang, Wenhui Wang, Zewen Chi, Saksham Singhal, Wanxiang Che, Ting Liu, Xia Song, and Furu Wei. 2021. Consistency regularization for cross-lingual fine-tuning. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 3403–3417, Online. Association for Computational Linguistics.

A Appendix

A.1 Task Descriptions

BoolQ (Clark et al., 2019) is a dataset for machine reading comprehension (MRC) with yes/no questions. Each data point consists of a triplet such as question, passage, and answer, requiring a broad range of inference capacities to solve questions. SNLI (Bowman et al., 2015) and RTE (recognising textual entailment) (Candela-Quinonero et al., 2006) are datasets for natural language inference (NLI). Each data point is composed of a sentence pair and a label indicating the relationship between the pair (i.e., "entailment", "neural", and "contradiction"). MRPC (Microsoft Research Paraphrase Corpus) (Dolan and Brockett, 2005) is a dataset for semantic textual similarity (STS). Each data point consists of a sentence pair and a label indicating whether the two paraphrased sentences are semantically equivalent. SST-2 (Stanford Sentiment Treebank) (Socher et al., 2013) is a dataset for sentiment analysis (SA). Each data point is composed of a phrase and a binary sentiment label (i.e., positive and negative). AG-News (Zhang et al., 2015) is a dataset of new articles for topic classification (TC). Each data point is composed of a title and a description of an article, and a label related to one of the four topics of the article (i.e., "World", "Sports", "Business", and "Sci/Tech"). WiC (Word-in-Context) (Pilehvar and Camacho-Collados, 2019) is a dataset for identifying the intended meaning of words. Each data point consists of two sentences containing the same specific word and a label indicating whether the word is used with the same meaning in different contexts.

A.2 Applicability of Logical Consistencies to Downstream Tasks

Negational Consistency Applicability. Among our downstream tasks, negational consistency is invalid for the TC and WiC tasks. Regarding the TC task, negated sentences normally belong to the same category as their original version, as illustrated in the example in Section 2.2. Similarly, the labels are preserved in the WiC task, because the meaning of the target word does not change in the perturbed sentence.

Although negational consistency is theoretically applicable to the SA task, we remove it from our evaluation scope for a practical reason. We observe that our method for generating the opposite meaning sentence does not suit well on the spoken language (e.g., movie reviews) that constitute the SST2 dataset.

Symmetric Consistency Applicability. For symmetry to hold, the following two conditions are necessary:

Condition 1. The input should consists of two sentences.

Condition 2. The hierarchy between the two sentences should be equivalent.

The TC and SA tasks violate the first condition. Regarding the MRC task (i.e., BoolQ), the question is dependent on the passage, and, therefore, it violates the second condition. As a result, the three tasks are removed from our scope for evaluating symmetric consistency.

Transitive Consistency Applicability. Theoretically, transitive consistency is valid for the downstream tasks where symmetric consistency holds. However, it requires one more condition for practical reasons: the two data points must have a common sentence, e.g., the same hypothesis in the NLI task. Only the SNLI and WiC datasets satisfy this condition among our candidate tasks. Although it is possible to construct new data for the MRPC and RTE datasets, it can cause a distribution shift issue that could exaggerate the inconsistency problem. Therefore, we conducted the transitive consistency evaluation only on the SNLI and WiC datasets.

Additive Consistency Applicability. Additive consistency always holds for tasks that take a single sentence as an input. However, it is not guaranteed if a downstream task requires more than two sentences as an input. Table 7 shows the example of the violation in the SNLI task. Thus, we tested additive consistency only for the SA and TC tasks.

A.3 Training Hyperparameters

Table 8 describes the batch-size per GPU, input sentence length (i.e., number of tokens), and learning rates used for training models for each dataset. Similarly to previous works, we confirm that the datasets with large training data (e.g., SNLI, SST2, and AG-news) were insensitive to hyperparameter values.

A.4 Human Annotation

We used Amazon Mechanical Turk (https://www.mturk.com/) for annotating our data. We employed Anglophone annotators with an acceptance rate of at least 98% and the number of HITs greater than

Model		BoolQ $\mathcal{F}_{val}$	MRPC $\mathcal{F}_{val}$	RTE $\mathcal{F}_{val}$	SNLI $\mathcal{F}_{val}$	SST2 $\mathcal{F}_{val}$	AG-News $\mathcal{F}_{val}$	WiC $\mathcal{F}_{tr}$	WiC $\mathcal{F}_{val}$
BERT	base	66.6 (1.0)	81.8 (1.9)	62.1 (2.0)	90.1 (0.2)	90.5 (0.3)	93.2 (0.2)	62.5 (13.4)	53.0 (7.4)
	large	70.3 (1.4)	82.0 (1.4)	64.4 (3.3)	91.0 (0.2)	92.4 (0.4)	93.9 (0.2)	70.0 (9.0)	57.7 (2.9)
RoBERTa	base	75.8 (1.0)	86.4 (0.9)	71.5 (1.8)	91.5 (0.0)	92.9 (0.4)	94.1 (0.1)	78.6 (3.9)	63.8 (1.8)
	large	84.9 (0.4)	88.6 (1.0)	81.5 (2.1)	93.0 (0.1)	95.9 (0.3)	94.3 (0.2)	77.0 (6.2)	66.0 (2.0)
Electra	base	73.8 (2.6)	88.3 (0.2)	75.3 (2.5)	91.8 (0.1)	93.9 (0.2)	93.2 (0.2)	80.4 (4.3)	66.5 (5.9)
	large	87.1 (0.5)	90.1 (0.6)	86.7 (1.2)	93.5 (0.3)	95.4 (2.2)	93.8 (0.4)	80.7 (1.8)	69.0 (1.2)
ERNIE2.0	base	76.2 (1.2)	86.8 (0.9)	73.4 (2.6)	91.1 (0.2)	93.6 (0.2)	93.5 (0.1)	62.8 (13.1)	56.0 (4.6)
	large	82.6 (0.7)	86.6 (1.2)	76.7 (1.1)	92.1 (0.0)	95.1 (0.2)	94.0 (0.2)	67.1 (9.5)	55.2 (10.1)
GPT2	base	62.9 (1.7)	77.3 (1.0)	65.3 (2.5)	84.7 (1.1)	90.9 (0.5)	92.9 (0.1)	73.4 (3.8)	63.5 (2.6)
	large	75.3 (0.8)	80.4 (1.2)	69.0 (2.8)	90.8 (0.2)	94.1 (0.4)	94.1 (0.2)	87.4 (5.7)	64.5 (2.2)
BART	base	64.8 (2.4)	85.6 (1.1)	70.7 (1.0)	90.8 (0.2)	93.0 (0.3)	93.8 (0.3)	78.8 (5.5)	56.0 (1.7)
	large	78.4 (3.9)	81.6 (8.3)	74.9 (3.1)	93.1 (0.1)	95.9 (0.2)	94.0 (0.7)	77.3 (3.5)	58.9 (2.9)
T5	base	79.9 (0.2)	86.8 (0.9)	77.6 (0.2)	90.1 (0.1)	94.0 (0.2)	92.1 (0.2)	82.3 (0.3)	64.5 (1.1)
	large	83.8 (0.6)	89.3 (0.9)	88.0 (0.6)	92.1 (0.2)	95.8 (0.1)	92.5 (0.4)	84.6 (1.5)	70.3 (0.8)

Table 6: Our validation performance of the PLMs on the seven downstream tasks; $\mathcal{F}_{tr}$ and $\mathcal{F}_{val}$ denote F1 score on the training and validation set, respectively. We report the training performance of the WiC task, because the gap between training and validation performance is large compared to the other tasks. We report the average of five repetitions. The values written in parenthesis imply a standard deviation.

EXAMPLE 1
Premise: Two women are embracing while holding to go packages.
Hypothesis: Two woman are holding packages.
Label: entailment

EXAMPLE 2
Premise: Two men on bicycles competing in a race.
Hypothesis: People are riding bikes.
Label: entailment

MERGED EXAMPLE
Premise: Two women are embracing while holding to go packages. Two men on bicycles competing in a race.
Hypothesis: Two woman are holding packages. People are riding bikes.

Table 7: Example of SNLI data where negational consistency does not hold. The label of the merged example cannot be "entailment", because two women are not riding bikes.

	BoolQ	SNLI	RTE	MRPC	WiC	SST2	AG-news
b-size	8	64	8	8	64	32	32
s-len	512	128	256	128	128	128	256
lr	$2e^{-5}$	$1e^{-5}$	$1e^{-5}$	$2e^{-5}$	$1e^{-5}$	$1e^{-5}$	$1e^{-5}$

Table 8: Batch-size, sentence length, and learning rates used for the BECEL benchmark experiments.

	Fixed Variable	Modified Variable
BoolQ	passage	question
SNLI	premise	hypothesis
RTE	premise	hypothesis
MRPC	sentence1	sentence2
WiC	word, sentence1	sentence2
SST2	-	text
AG-news	-	text

Table 9: Modified variables of each dataset for collecting $\mathcal{E}_N$ for semantic and negational consistency.

1,000. The representative snapshot of the UI for the human annotation is shown in Figure 7.

B Examples

Test case		Predicted	Pass?
Testing **Semantic Consistency** on the **TC** task.		Labels: World, Sports, Business, Sci/tech	
Original	UN's Global Fund meets African leaders in Tanzania for talks on fighting the world's deadliest diseases.	World	O
New	The United Nations Global Fund meets African leaders in Tanzania to discuss combating the world's deadliest diseases.	**World**	
...			
Testing **Negational Consistency** on the **NLI** task.		Labels: entailment, neutral, contradiction	
Original	Premise: The man in the blue shirt is relaxing on the rocks. Hypothesis: A man is wearing a blue shirt.	entailment	X
New	Premise: The man in the blue shirt is relaxing on the rocks. Hypothesis: A man is **not** wearing a blue shirt.	**entailment**	
...			
Testing **Symmetric Consistency** on the **STS** task.		Labels: equivalent, not_equivalent	
Original	S1: Zuccarini was ordered held without bail Wednesday by a federal judge in Fort Lauderdale, Fla. S2: A federal magistrate in Fort Lauderdale ordered him held without bail.	equivalent	O
New	S1: A federal magistrate in Fort Lauderdale ordered him held without bail. S2: Zuccarini was ordered held without bail Wednesday by a federal judge in Fort Lauderdale, Fla.	**equivalent**	
...			

Figure 5: Data examples of semantic, negational, and symmetric consistency evaluation.

Test case		Predicted	Pass?
Testing **Transitive Consistency** on the **WiC** task.		Labels: True, False	
Original 1	Word: back Sentence1: The horse refuses to back. Sentence2: The wind backed.	True	X
Original 2	Word: back Sentence1: The wind backed. Sentence2: The train backed into the station.	True	
New	Word: back Sentence1: The horse refuses to back. Sentence2: The train backed into the station.	**False**	
...			
Testing **Additive Consistency** on the **SA** task.		Labels: negative, positive	
Original 1	Unflinchingly bleak and desperate.	negative	X
Original 2	A sometimes tedious flim.	negative	
New	Unflinchingly bleak and desperate. A sometimes tedious flim.	**positive**	
...			

Figure 6: Data examples of transitive and additive consistency evaluation.

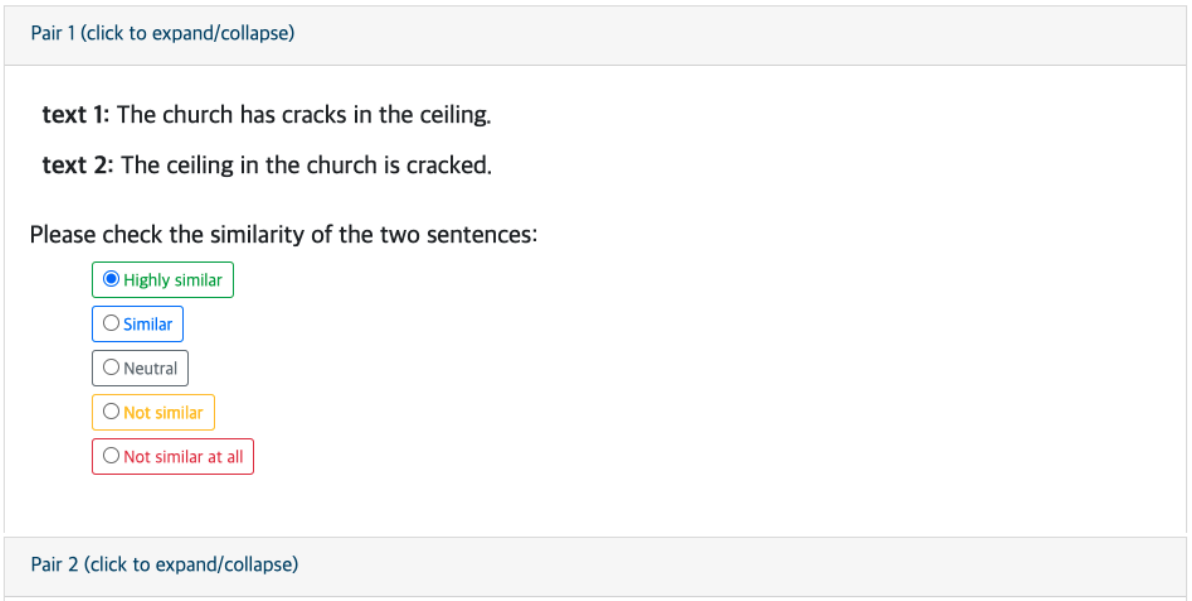

Figure 7: Snapshot of our human annotation UI for annotating semantic consistency evaluation data.

ORIGINAL SAMPLE Sentence 1: The stupendous power of the Tevatron made possible the 1995 discovery of the top quark - the last of six flavors of quarks predicted by the standard model theory of particle physics. Sentence 2: The top quark is the last of six flavors of quarks predicted by the standard model theory of particle physics. **ADVERSARIAL SAMPLE** Sentence 1: The stupendous power of the Tevatron made possible the 1995 discovery of the top quark - the top of six flavors of quarks predicted by the standard model theory of particle physics. Sentence 2: The top quark is the last of six flavors of quarks predicted by the standard model theory of particle physics.	
ORIGINAL SAMPLE LABEL entailment	ADVERSARIAL SAMPLE LABEL entailment
ORIGINAL SAMPLE Sentence 1: Rockweed has been harvested commercially in Nova Scotia since the last 1950's and is currently the most important commercial seaweed in Atlantic Canada. Sentence 2: Marine vegetation is harvested. **ADVERSARIAL SAMPLE** Sentence 1: Rockweed has been introduced commercially in Nova Scotia since the last 1950's and is currently the most important commercial seaweed in Atlantic britain. Sentence 2: Marine vegetation is harvested.	
ORIGINAL SAMPLE LABEL entailment	ADVERSARIAL SAMPLE LABEL entailment

Table 10: Examples of degenerated adversarial samples of BAE (Garg and Ramakrishnan, 2020) for the RTE dataset. The words that changed in the adversarial samples are underlined in both original and adversarial samples. It is hard to consider that the label of the adversarial samples is the same as the original label.

KoBEST: Korean Balanced Evaluation of Significant Tasks

Myeongjun Jang[1*] **Dohyung Kim**[2*] **Deuk Sin Kwon**[2] **Eric Davis**[2]
[1]Department of Computer Science, University of Oxford, UK
myeongjun.jang@cs.ox.ac.uk
[2]Language Super Intelligence Labs, SK Telecom, South Korea
{bruno, brian.k, eric.davis}@sktair.com

Abstract

A well-formulated benchmark plays a critical role in spurring advancements in the natural language processing (NLP) field, as it allows objective and precise evaluation of diverse models. As modern language models (LMs) have become more elaborate and sophisticated, more difficult benchmarks that require linguistic knowledge and reasoning have been proposed. However, most of these benchmarks only support English, and great effort is necessary to construct benchmarks for other low resource languages. To this end, we propose a new benchmark named **Ko**rean **b**alanced **e**valuation of **s**ignificant **t**asks (KoBEST), which consists of five Korean-language downstream tasks. Professional Korean linguists designed the tasks that require advanced Korean linguistic knowledge. Moreover, our data is purely annotated by humans and thoroughly reviewed to guarantee high data quality. We also provide baseline models and human performance results. Our dataset is available on the Huggingface [1].

1 Introduction

The NLP field is now facing unprecedented rapid development. A major factor propelling the progress is the existence of unified benchmark datasets like GLUE (Wang et al., 2018), which are designed to assess models' language understanding capabilities. Such benchmark datasets, enabled modern pre-trained language models (PLMs), such as BERT (Devlin et al., 2019) and GPT-2 and GPT-3 (Radford et al., 2019; Brown et al., 2020), to be assessed in objective and multifaceted manners. The success of GLUE also lead to similar benchmark datasets in a variety of other languages, such as French (Le et al., 2020), Korean (Park et al., 2021), Chinese (Xu et al., 2020) and Indonesian (Wilie et al., 2020).

However, many recent studies reveal that the outstanding performance of PLMs on such benchmark datasets seems plausible but not probable. These studies have found that datasets may contain many spurious artefacts, and the performance of PLMs is enhanced by excessive usage of said artefacts (Habernal et al., 2018; Niven and Kao, 2019; McCoy et al., 2019; Bender and Koller, 2020). Another line of work observed that many PLMs, which showed promising results in GLUE, fall short of expectations for more difficult tasks that require linguistic knowledge (Bhatt et al., 2021) or logical reasoning (Tian et al., 2021). As a result, the importance of well-designed evaluation datasets with higher difficulty-level has been highlighted, and new datasets, such as CHECKLIST (Ribeiro et al., 2020), and LOGICNLI (Tian et al., 2021), have been proposed. Most of them only support specific languages like English, and it requires large efforts to build higher difficulty-level language evaluation suits for other low resource languages, however.

When it comes to the Korean language, two benchmarks are widely used: Korean-NLI & STS (Ham et al., 2020) and KLUE (Park et al., 2021). The former is machine- and human-translated from English natural language inference (NLI) and semantic textual similarity (STS) datasets, which hardly reflect the characteristics of the Korean language. The latter is a Korean version of GLUE benchmark which supports eight tasks, such as NLI, STS, named entity recognition (NER), and relation extraction (RE). Although these tasks are useful for assessing general language ability, it is difficult to ascertain whether a model is able to reason based on more complicated knowledge beyond text form (e.g., passage of time, meaning of text, causality). To this end, we aim to construct a new benchmark dataset in Korean named KoBEST, which consists of five downstream tasks that require advanced knowl-

[*]These authors contributed equally to this work
[1]https://huggingface.co/datasets/skt/kobest_v1

Proceedings of the 29th International Conference on Computational Linguistics, pages 3697–3708
October 12–17, 2022.

Tasks	|Train|	|Dev|	|Test|	|Labels|	Text Source
KB-BoolQ	3.7K	700	1.4K	2	Wikipedia
KB-COPA	3.1K	1K	1K	2	N.A
KB-WiC	3.3K	1.3K	1.3K	2	Korean Dictionary
KB-HellaSwag	2K	500	500	4	Wikipedia, YouTube
KB-SentiNeg	3.6K	400	397	2	Product reviews

Table 1: The number of data instances and labels for each downstream task.

edge of Korean. We carefully constructed the data based on the following design principles:

- *Human-driven data annotation*: Our data is purely annotated by humans to prevent incorrect and ambiguous data instances caused by automatic data annotation approach.
- *Leveraging professional linguistic knowledge*: As a result of our collaboration with professional Korean linguists, we re able to collect grammatically correct data with rich vocabulary and expressions.
- *Availability to public*: As a benchmark dataset, it is important to ensure public accessibility. We guarantee that our data is free to use and redistribute.
- *High data quality*: Our data passed thorough reviews driven by both models and humans to deliver high quality data without superficial cues and heuristic artefacts.
- *Avoiding AI ethical issues*: Human review process have been performed to remove toxic content, social biases, and personal information from our data set.

Next, we evaluated widely used Korean PLMs on the KoBEST dataset. Specifically, we conducted fine-tuning, zero-shot, one-shot, and few-shot experiments. The experimental results can serve as a baseline for performance on KoBEST. Participants also provided human performance baselines for all of our tasks. Our results suggest that modern PLMs and a large-size generative language model (GLM) are far from reaching human-level language ability.

2 KoBEST Downstream Tasks

2.1 Overview

The KoBEST benchmark consists of the following five downstream tasks:

1. KoBEST-BoolQ (KB-BoolQ): identify whether a given question is true or false considering a paragraph.
2. KoBEST-COPA (KB-COPA): select an alternative which is a cause/effect of a given premise.
3. KoBEST-WiC (KB-WiC): identify whether the meaning of a target word is the same or different in two given contexts.
4. KoBEST-HellaSwag (KB-HellaSwag): select a correct sentence among four candidates that is likely to appear after a given context.
5. KoBEST-SentiNeg (KB-SentiNeg): predict the polarity of a negated sentence.

The number of training/development/test data points is illustrated in Table 1.

2.2 KoBEST-BoolQ

Data/Task Description We built the KB-BoolQ dataset by referencing boolean questions (BoolQ) task (Clark et al., 2019). A data point consists of a paragraph, question, and label. The task aims to evaluate models' understanding of the paragraph by asking a true/false question. An example is presented in Table 2.

We extracted paragraphs from Korean Wikipedia[2]. To cover diverse materials, we first choose topics, such as Science/Technology and Art/Culture, by referring to previous works regarding Korea written/spoken language (Seo and Kim, 2005; Seo, 2007). Then, we defined keywords for each topic and selected documents containing enough information regarding the keyword. Next, we extracted paragraphs for each document and generated corresponding questions that could be answered as true/false based on the paragraph.

Guidelines Annotators were instructed to construct the KB-BoolQ dataset following the guidelines described below.

1. Paragraphs should be evenly extracted from various domains and topics to minimise bias.
2. Questions should be answered only with the information presented in the paragraph. We set this guide for two reasons: 1) to exclude the impact of pre-trained commonsense knowledge for decision making and 2) annotators have different viewpoints regarding the boundary of commonsense knowledge, which can cause uneven task difficulty.
3. Questions should be written in clear, unambiguous, easy-to-understand language. A true/false judgement should be obvious from a human perspective.

[2] https://ko.wikipedia.org/wiki/

KB-BoolQ	**Paragraph**: 구한말, 통영 안뒤산 기슭 간창골에 김봉제 형제가 살았다. 김봉제는 관약국을 경영하며 부를 누렸는데, 선비적 성품을 지녔던 형과 반대로 막냇동생 김봉룡은 성질이 포악했다. 어느 날 봉룡은 아내였던 숙정을 사모하던 나그네를 살해하였고, 숙정은 누명을 벗으려고 비상을 먹고 자살한다. (At the end of the Joseon Dynasty, the Kim Bong-je brothers lived in Ganchanggol at the foot of Andui Mountain in Tongyeong. Kim Bong-je enjoyed wealth while running a government office, but his younger brother Kim Bong-ryong had a violent temper, contrary to his brother who had a scholarly character. One day, Bongryong killed a traveler who adored his wife Sukjeong, and Sukjeong commits suicide to clear his name.) **Question**: 봉룡은 숙정을 죽였는가? (Did Bongryong kill Sukjeong?) **Answer**: False
KB-COPA	**Premise**: 전쟁이 시작되었다. (The war had begun.) **Question**: 결과 (Effect) **Alternative 1**: 병사들이 집으로 돌아왔다. (Soldiers returned home.) **Alternative 2**: 병사들이 전투에 파견되었다. (Soldiers were sent to battle.) **Correct Alternative**: 2
KB-WiC	**Context 1**: 망가진 엔진은 **수리** 가 불가능하다. (It is impossible to repair a broken engine.) **Context 2**: 이 배는 **수리** 에 들어간 지 일주일이 됐다. (The ship has been under repair for a week.) **Target Word**: 수리 (repair) **Answer**: True
KB-HellaSwag	**Context**: 양궁 선수들이 경기장으로 입장한다. 관중들이 함성을 지르고 응원한다. 선수들이 상대팀과 악수하고 자리로 돌아온다. 코치가 전략을 설명하고 화이팅을 외친다. (Archery players enter the stadium. The crowd shouts and cheers. The players shake hands with the opposing team and return to their seats. The coach explains the strategy and shouts "Go for it.") **Ending 1**: 활이 과녁에 적중했다. (The arrow hits the target.) **Ending 2**: 선수가 심호흡을 하고 활을 쏜다. (The player takes a deep breath and shoots an arrow.) **Ending 3**: 선수가 활을 들어 과녁을 조준한다. (The player raises his bow and aims at the target.) **Ending 4**: 선수들이 각자 자리에 서서 활을 꺼낸다. (The players stand in their own positions and take out their bows.) **Correct Ending**: 4
KB-SentiNeg	**Sentence 1**: 뚜껑이 잘 열려요! (The lid opens well!) **Label 1**: 긍정 (Positive) **Sentence 2**: 뚜껑이 잘 안열려요! (The lid does not open well!) **Label 2**: 부정 (Negative)

Table 2: Examples of development set from the KoBEST tasks. The variables of each task are highlighted in bold. Text written in parenthesis is the English translated version of the original data points.

2.3 KoBEST-COPA

Data/Task Description We referenced choice of plausible alternatives (COPA) (Roemmele et al., 2011) to construct the KB-COPA dataset. The data has four variables: premise, two alternatives, and a question that asks a model to decide the cause or effect of the premise from the two alternatives. An example is available in Table 2.

Guidelines We provided the following guidelines to annotators for generating data instances.

1. The alternatives should belong to a similar area, e.g., states and actions. This rule is introduced to preclude systems from making decisions based on situational difference, not the meaning of alternatives.
2. The alternatives should contain a keyword related to that of premise. For instance, in the example presented in Table 2, the keyword of the premise is "전쟁 (war)", and both alternatives contain the related same keyword "병사 (soldier)". We introduce this guideline to increase the task's difficulty by making the alternatives belong to the same category.
3. All the premises and alternatives should be written concisely so that the content can be understood intuitively. Therefore, using proper noun, slang, and redundant expressions should be avoided.
4. All sentences should be written in the past tense. In the Korean language, simple present can cause confusion because it has indication of tense and sometimes can imply present progressive. On the other hand, the past tense is morphologically clear and is able to convey meaning without confusion.
5. All sentences must include a subject. Although the subject is frequently omitted in Korean, it is difficult to infer the cause or effect without a subject because all the premises and alternatives are quite short. Therefore, even though such sentences are slightly unnatural in Korean, we guide annotators to insert a subject.

2.4 KoBEST-WiC

Data/Task Description KB-WiC is a task that determines whether a word has the same connotation in different contexts. We referenced words in context (WiC) (Pilehvar and Camacho-Collados, 2019) when building the dataset. An instance is composed of a target homonym and two different

contexts that contain the target word. Table 2 provides an example for the KB-WiC task. Unlike the original WiC dataset that includes various word forms, we only used words with the same form, so as to focus more on recognising a word's meaning without the distracton of variouus forms.

Guidelines To construct the KB-WiC dataset, we instructed annotators to follow these guidelines.

1. A target word should be listed in the National Institute of the Korean Language Basic Korean Dictionary[3] or Korea University Korean Dictionary (Hong and Kim, 2009). We exclude words not registered in the dictionaries because they can cause ambiguous criteria for determining an answer. This is despite the fact that they are generally used in daily life.
2. For generating a data point where an answer is *False*, only a **homonym** should be used as a target word because a **polysemy** makes the task considerably more challenging, even for native speakers.
3. The part of speech (PoS) tag of a target word should be a noun, pronoun, numeral, or dependent noun[4]. We introduce this guideline because the four PoS tags have a fixed form and distinct meaning in Korean.
4. The contexts should be extracted from example sentences in the dictionaries to make it possible to clearly understand the sense of a target word only using the given context.

2.5 KoBEST-HellaSwag

Data/Task Description This task evaluates whether a system can utilize passage of time and order to complete the last sentence in a series of sentences. We referenced the HellaSwag dataset (Zellers et al., 2019) to build our version but modified the task to consider specific characteristics of the Korean language.

The original HellaSwag benchmark was designed to ascertain whether a LM can generate a plausible ending sentence given a relevant subject and context. In Korean, however, subjects are typically omitted. As a result, if the ending sentence is generated from the subject, the sentence becomes awkward and barely resembles a plausible Korean sentence. Evaluating such unnatural sentences is not in line with the purpose of KoBEST, so we modify the task to predict the most plausible final sentence among four candidates. An example instance is available in Table 2.

Guidelines We instruct annotators to build the data based on the following guidelines.

1. The annotators should generate or modify free-text descriptions of YouTube videos and Wikepedia documents that progress with the passage of time.
2. At least three sentences should be included in the context. A system should have as much context as possible to generate a plausible ending sentence.
3. All the candidate-ending sentences should be thematically related to the context. The answer should only be able to be found by analysing the passage of time among the sentences, not via the topic or keywords. This guideline is introduced to prevent low task difficulty by generating alternative endings that simply contradict the correct ending.

2.6 KoBEST-SentiNeg

Data/Task Description Many studies have revealed that PLMs lack understanding of negation expressions (Hossain et al., 2020; Ettinger, 2020; Kassner and Schütze, 2020; Hosseini et al., 2021; Jang et al., 2022). Inspired by the *Negation* capability test of Ribeiro et al. (2020), we designed a similar but enhanced task by utilizing negation to create sentences opposite in meaning. Specifically, we created a two-class sentiment analysis task by generating product reviews based on real product reviews available on e-commerce websites. [5] We then used the *training* and *development* sets to train a sentiment classification model. Next, we extracted candidates from training data where the polarity switched when transformed into a sentence with the opposite meaning. Finally, we converted each candidate to a sentence with its opposite meaning and reversed the label. The modified candidate is then added to the final test set. We used the following three methods to generate the sentences with opposite meanings.

1. **Adding/removing negation expressions**: We add or remove Korean negation expressions (e.g., "안", "못", " 지 않다").

[3] https://stdict.korean.go.kr/main/main.do

[4] A noun that cannot be used without the help of other words in Korean.

[5] The real product reviews are only used as references and our data is newly generated by human annotators.

2. **Antonym replacement**: A word is replaced with its antonym.
3. **Using both method 1 and 2 or idiom**: Both methods described above are used. If a sentence includes an idiom, we replace it with its corresponding opposite meaning idiom.

Guidelines We instructed the annotators to comply with the following guidelines to generate data points for the KB-SentiNeg task.

1. The sentence should not include the name of specific brands or products. This guideline is meant to avoid any possible legal issues.
2. To generate a new sentence resembling a real product review, typos and spacing errors that frequently occur in Korean spoken language should be included occasionally.

2.7 Evaluation Metrics

All of our downstream tasks have discrete labels. Therefore, we use the $F1$ score as a base criterion to evaluate models' performance.

3 Design Principles

In this section, we provide detailed descriptions about how we attempted to achieve the design principles illustrated in Section 1.

3.1 Human-driven data annotation

Automatic data generation using meta-information, such as a review score and news article category (e.g., Naver Sentiment Movie Corpus[6], is a widely used approach to rapidly collect a large amount of labelled data. While it is an efficient approach, there exists a high risk of the dataset containing incorrect and ambiguous data points. Such noisy data points are a major issue for evaluation datasets because they can lead to spurious performance increases (and or degradations) in the performance of LMs.

Our data is created purely by human annotations to produce the highest quality dataset with the lowest amount of incorrect and ambiguous data points possible. We hired four annotators who are Korean native speakers and major in *Korean Language Education* or *Korean Language in Literature*. Also, our Korean linguists trained the annotators before the data annotation process to avoid generating possible grammatical errors and unethical expressions.

[6] https://github.com/e9t/nsmc

Example #1
P: 날씨가 추워졌다. (The weather has become colder.)
Q: 원인 (Cause)
A1: 겨울이 되었다. (Winter has come.)
A2: 여름이 되었다. (Summer has come.)
Example #2
P: 겨울이 되었다. (Winter has come.)
Q: 결과 (Effect)
A1: 날씨가 추워졌다. (The weather has become colder.)
A2: 날씨가 더워졌다. (The weather became hot.)

Table 3: Examples of the KB-COPA data where the causality is interlocked. **P**, **Q**, **A1** and **A2** denote a premise, question, and alternatives, respectively. The two sentences highlighted in red and blue colours are swapped by changing the question from *Cause* to *Effect*.

3.2 Leveraging linguistic knowledge

Korean benchmark datasets, translated from English datasets (e.g., Kornli and Korsts (Ham et al., 2020)), might include incorrect translations and grammatical errors, particularly if they are machine-translated. Moreover, since the original examples come from English, such benchmark datasets are unlikely to assess properly assess Korean-specific knowledge or language intricacies.

Relying on our in-house Linguistic team, allowed us to mitigate and resolve issues with automatically generated datasets. First, our linguists trained the annotators to generate natural and grammatical Korean sentences and performed thorough reviews of the data. Thanks to their efforts, we have created a highly grammatical and natural. Secondly, the linguists designed tasks and data generation processes that considered the Korean language's characteristics. This is illustrated in guidelines for each task in Section 2. Such guidelines and processes enabled us to create an accurate Korean evaluation dataset with expressive vocabulary and colloquial usage.

3.3 Availability to public

Our data is free from copyright issues. All sentences and answers, except for the paragraphs in KB-BoolQ task, are generated by our annotators from scratch by referencing publicly available sources. Also, the paragraphs in KB-BoolQ were extracted from Wikipedia, which is under the Creative Commons Attribution-ShareAlike license. Therefore, researchers are free to use, modify and redistribute the KoBEST dataset.

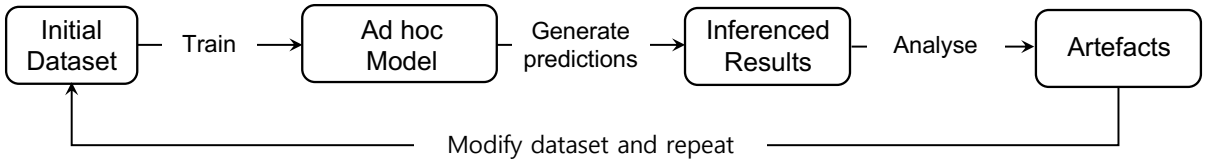

Figure 1: The overall process of model-driven review for removing artefacts in training data.

3.4 High data quality

As a benchmark suite, accomplishing high data quality is important to accurately evaluate various LMs. We achieve this purpose through two review phases: *human-driven* and *model-driven* reviews.

Human-driven review After collecting data, our linguists reviewed all data instances and found two major issues in the KB-COPA and KB-HellaSwag datasets.

For the KB-COPA task, we observed many cases with high correlation between data instances. Examples corresponding to this case are presented in Table 3. We conjecture this is because it was easier for the annotators to collect data by simply swapping the premise and answer along with changing the question. We removed or modified such instances because they are near duplicates and harm data diversity.

In the KB-HellaSwag task, we found several critical cases where predicting a correct final sentence is quite difficult due to the omission of detailed delineations in the context. This issue occurs because the source that the annotators referenced for generating contexts lacks detailed information occasionally. For such ambiguous instances, we appended additional clues to the context to allow inferring the meaning between the context and the final sentence.

Model-driven review Artefacts existing in training data can lead a model to learn spurious inductive biases, resulting in distorted evaluation results (Gururangan et al., 2018; McCoy et al., 2019; Hossain et al., 2020). Therefore, we conduct a model-driven review process to find and remove such unwanted artefacts. The overall process is illustrated in Figure 1. First, we trained an ad-hoc model with the initial dataset for each task. Next, we generated predictions for the development and test datasets and analysed the results to ascertain whether specific words/patterns/phrases were strongly correlated with labels. Finally, our linguists analyzed the issues and updated the datasets accordingly. We repeated the whole process up to three times for each dataset. Through this process, we observed serious artefacts, especially in the KB-WiC and KB-COPA datasets. More than 70% of questions containing number-related representations had *False* as a label. Also, the label distributions of data instances containing specific phrases (e.g., " 덥다/춥다 (hot/cold)" or 했다/하지 않았다 (did/did not)) were highly skewed towards the *False* label. All such artefacts were successfully removed and modified by our linguists.

3.5 Avoiding AI ethical issues

Social biases embedded in training data can lead to unethical behaviour of language models (Nangia et al., 2020). To mitigate such issues, we made efforts to remove unethical expressions, such as toxic content (e.g., insults, slang, sexual harassment) and social bias (e.g., gender, race, religion). Our linguists clearly instructed the annotators to avoid unethical expressions when generating sentences and extracting paragraphs from Wikipedia for the KB-BoolQ task. Also, linguists reviewed the data for potential ethical issues, as described in Section 3.4

4 Experiments

In this section, we provide model and human performance results.

4.1 Fine-tuning Experiments

4.1.1 Experimental Design

Model Candidates We used the following four pre-trained Korean language models to benchmark our KoBEST dataset:

- **Encoder models**: KoBERT [7], KoElectra [8]
- **Decoder models**: KoGPT3-1.2B
- **Encoder-Decoder models**: KoBART [9]

For KoBART, we applied the text-to-text multi-task training technique to fine-tune the model. We

[7] https://huggingface.co/monologg/kobert

[8] https://huggingface.co/monologg/koelectra-base-v2-discriminator

[9] https://huggingface.co/hyunwoongko/kobart

Model	KB-BoolQ	KB-COPA	KB-WiC	KB-HellaSwag	KB-SentiNeg	Average
KoBERT (FT)	62.9±3.0	74.6±0.8	77.3±0.8	74.4±0.4	86.8±2.0	75.2
KoElectra (FT)	**75.1±1.0**	**81.5±0.4**	**79.7±1.8**	**74.7±0.8**	**91.9±1.1**	**80.6**
KoGPT3-1.2B (FT)	73.5±1.6	79.3±0.6	68.4±2.2	73.8±1.0	89.5±3.3	77.0
KoBART (FT)	60.6±2.9	56.9±3.2	60.4±4.9	51.4±1.3	88.6±1.2	63.6
KoGPT3-39B ($k = 0$)	33.1	76.8	34.7	59.8	57.7	52.8
KoGPT3-39B ($k = 1$)	50.2	78.3	51.8	60.2	74.2	66.3
KoGPT3-39B ($k = 10$)	46.9	80.9	52.2	58.7	91.6	70.6
Human	95.1	98.1	96.6	92.4	99.0	96.2

Table 4: The test F1 scores of Korean LMs on the KoBEST downstream tasks. The first and second blocks are the experimental results of fine-tuning and zero/few-shot learning, respectively. For the fine-tuning experiments, we repeat each experiment five times and report the average and standard deviation. The best values for each task are written in bold. k refers to the number of few-shot samples.

	BoolQ	COPA	WiC	HellaSwag	SentiNeg
b-size	8	16	16	8	16
s-len	256	128	256	256	128
lr	$5e^{-6}$	$5e^{-6}$	$1e^{-5}$	$2e^{-5}$	$1e^{-5}$

Table 5: Batch-size (b-size), maximum input length (s-len), and learning rates (lr) used for the KoBEST benchmark experiments.

Model	$\mathcal{F}_{val}$	$\mathcal{F}_{test}$	$\triangle$
KoBERT	99.1	86.8	12.3
KoElectra	99.4	91.9	7.5
KoGPT3-1B	99.8	89.5	10.3
KoBART	98.7	88.6	10.1

Table 6: The average performance gap between validation and test sets of KB-SentiNeg task.

transformed inputs of each text to a free-text input by referencing the transformation formats used in the T5 model (Raffel et al., 2020).

Training Details We used AdamW optimiser (Loshchilov and Hutter, 2017) for training with a linear learning rate scheduler decaying from 1e-2. We trained all models for 10 epochs, and used the early stopping method during the training. Different batch sizes and learning rates were used across the tasks, and detailed training hyperparamters are presented in Table 5.

4.1.2 Results and Discussion

The fine-tuning results can be seen in the first block of Table 4. Overall, KoElectra showed the best results, followed by KoGPT-1B, suggesting that model size is not necessarily a requisite for better performance.

Do models understand the opposite in meaning? Table 6 shows the validation and test performance of fine-tuned models on the **SentiNeg** task. Interestingly, all models show a large performance gap between the validation and test performance in the **SentiNeg** task. The results suggest that PLMs are vulnerable to a simple negation and antonym-replacement perturbation, even though the data points all originated from the training data. Our results are aligned with previous studies on English data that showed PLMs are incapable of understanding negation expressions (Hossain et al., 2020; Kassner and Schütze, 2020; Ettinger, 2020; Hosseini et al., 2021), suggesting that the issue stems from the PLM, not from language itself.

Model	KB-BoolQ	KB-WiC	KB-SentiNeg
Single-task	**66.5**±3.1	**68.3**±3.0	87.7±1.4
Multi-task	60.6±2.9	60.4±4.9	**88.6**±1.2

Table 7: The performance of KoBART models trained with single- and multi-task manners.

Text-to-Text Multi-task training is not always beneficial Unlike the other three models, KoBART was fine-tuned in a multi-task fashion. However, contrary to the common belief that multi-task training is beneficial in improving performance on benchmark suites (e.g., GLUE (Liu et al., 2019)), in our case, multi-task training produced the worst performance by a large margin. We conducted additional single-task classification experiments on KoBART by introducing a classifier layer. The multiple-choice tasks (i.e., KB-COPA and KB-HellaSwag) are not included in this experiment, as the structure of the BART model is not suitable for the multiple-choice tasks. The results are presented in Table 7. The results show that the single-task model performs better than multi-task models on the KB-BoolQ and KB-WiC tasks by a large margin. We also ascertained that the performance gap was statistically significant ($p < 0.05$) on the two tasks, while there was no significant difference on the KB-SentiNeg task. We conjecture that a leading cause is a misalign-

ment between tasks. All the downstream tasks in KoBEST are independent of each other. However, in the GLUE benchmark, for instance, the subtasks are well aligned, containing multiple datasets that share a common objective, e.g., NLI and STS, and it is well studied that the misalignment between task data can cause poor results (Wu et al., 2020).

4.2 Zero/Few Shot Experiments

4.2.1 Experimental Design

The advent of extremely large size GLMs like GPT3 (Brown et al., 2020) has allowed in-context learning (providing the model with a few or no samples) to apply the model to downstream tasks. To this end, we conducted zero-, one- and few-shot experiments by using a Korean GPT3 model trained by Language Superintelligence Labs with 39 billion parameters and 132 billion tokens. We then referenced the work of EleutherAI [10] to design prompts for our zero, one, and few-shot experiments. Several prompt examples are available in Table 8 in the appendix. For multiple-choice problems like **KB-COPA** and **KB-HellaSwag**, we selected the candidate having the lowest perplexity as the prediction.

4.2.2 Results and Discussion

Fine-tuned models are still best. The results, presented in the second block of Table 4, reveal that the fine-tuned models, apart from KoBART, outperform in-context learning methods. Our results are aligned with the work of Brown et al. (2020) that showed GPT3 performance based on few-shot learning is behind that of fine-tuned SOTA in many tasks, including all downstream tasks in SuperGLUE (Wang et al., 2019). Although it is interesting that a large GLM can achieve decent performance with only a few training examples, results suggest that we should be judicious using large GLM in practical applications; especially when considering performance compared to excessively high training costs (i.e., time and resources).

Increasing k is not always beneficial. Few-shot learning approaches with more examples increase performance in general, but merely increasing k does not always lead to better performance. Specifically, in the case of KoBEST, the performance is slightly worse on the **KB-BoolQ** and **KB-HellaSwag** tasks. We believe that the length of the input document is a leading cause of this phenomenon. Since the model's max input length plays a critical role in deciding the maximum number of examples (n) in the prompt (Yang et al., 2021), the available n decreases as the length of prompts increases. However, as we can see in Table 2, the data points of the **KB-BoolQ** and **KB-HellaSwag** tasks have longer inputs than the other tasks. As a result, the prompts for these tasks become very long and likely to exceed the model's maximum input length. This would result in a sliced prompt that may lack key information the model needs to make a successful prediction.

4.3 Human performance

We asked volunteers to evaluate the dataset to provide human-level performance metrics for KoBEST. Specifically, 10 native Korean evaluators evaluated 100 randomly sampled examples for each downstream task. The results are summarised in the last row of Table 4. The human evaluators outperformed all the PLMs by a large margin, suggesting that modern PLMs need further improvements to achieve human-level language ability.

5 Conclusion

A well-designed benchmark dataset is crucial for an objective and precise evaluation of LMs. Following the GLUE benchmark, more challenging benchmarks have been proposed as modern LMs become more elaborate and sophisticated. However, most of these benchmarks only support English or originate from English (e.g., translation), which hardly captures important characteristics of a specific language.

To this end, we propose a new Korean benchmark suite named KoBEST, which consists of five challenging downstream tasks. To overcome the disadvantages of the previous Korean benchmarks, we focused on 1) employing Korean-specific knowledge, 2) achieving high data quality and 3) removing superficial cues. To achieve these goals, we worked with professional Korean linguists and collected data manually and not automatically. We also conducted human- and model-driven review processes to eliminate superficial cues from our dataset. Moreover, we were extra cautious to avoid using unethical expressions.

Finally, we evaluated various PLMs on our new benchmark and provide baseline model and hu-

[10] https://github.com/EleutherAI/lm-evaluation-harness

man performance metrics. Our experimental results show that current LMs need further improvements to attain human-level language ability. We hope our new benchmark can contribute to advancements in the Korean NLP field.

Acknowledgements

We would like to express our great appreciation to Sunwoo Lee, Seo-jin Lee, and Seokyoung Hong for their valuable assistance and feedback.

References

Emily M. Bender and Alexander Koller. 2020. Climbing towards NLU: On meaning, form, and understanding in the age of data. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 5185–5198, Online. Association for Computational Linguistics.

Shaily Bhatt, Rahul Jain, Sandipan Dandapat, and Sunayana Sitaram. 2021. A case study of efficacy and challenges in practical human-in-loop evaluation of NLP systems using checklist. In *Proceedings of the Workshop on Human Evaluation of NLP Systems (HumEval)*, pages 120–130, Online. Association for Computational Linguistics.

Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel Ziegler, Jeffrey Wu, Clemens Winter, Chris Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. 2020. Language models are few-shot learners. In *Advances in Neural Information Processing Systems*, volume 33, pages 1877–1901. Curran Associates, Inc.

Christopher Clark, Kenton Lee, Ming-Wei Chang, Tom Kwiatkowski, Michael Collins, and Kristina Toutanova. 2019. BoolQ: Exploring the surprising difficulty of natural yes/no questions. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 2924–2936, Minneapolis, Minnesota. Association for Computational Linguistics.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, Minneapolis, Minnesota. Association for Computational Linguistics.

Allyson Ettinger. 2020. What bert is not: Lessons from a new suite of psycholinguistic diagnostics for language models. *Transactions of the Association for Computational Linguistics*, 8:34–48.

Suchin Gururangan, Swabha Swayamdipta, Omer Levy, Roy Schwartz, Samuel R Bowman, and Noah A Smith. 2018. Annotation artifacts in natural language inference data. In *NAACL-HLT (2)*.

Ivan Habernal, Henning Wachsmuth, Iryna Gurevych, and Benno Stein. 2018. The argument reasoning comprehension task: Identification and reconstruction of implicit warrants. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 1930–1940, New Orleans, Louisiana. Association for Computational Linguistics.

Jiyeon Ham, Yo Joong Choe, Kyubyong Park, Ilji Choi, and Hyungjoon Soh. 2020. Kornli and korsts: New benchmark datasets for korean natural language understanding. *arXiv preprint arXiv:2004.03289*.

I.S. Hong and H.G. Kim. 2009. Korea university korean dictionary. *Research Institute of Korean Studies, Korea University, Korea*.

Md Mosharaf Hossain, Venelin Kovatchev, Pranoy Dutta, Tiffany Kao, Elizabeth Wei, and Eduardo Blanco. 2020. An analysis of natural language inference benchmarks through the lens of negation. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 9106–9118, Online. Association for Computational Linguistics.

Arian Hosseini, Siva Reddy, Dzmitry Bahdanau, R Devon Hjelm, Alessandro Sordoni, and Aaron Courville. 2021. Understanding by understanding not: Modeling negation in language models. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 1301–1312, Online. Association for Computational Linguistics.

Myeongjun Jang, Frank Mtumbuka, and Thomas Lukasiewicz. 2022. Beyond distributional hypothesis: Let language models learn meaning-text correspondence. In *Findings of the Association for Computational Linguistics: NAACL 2022*, pages 2030–2042, Seattle, United States. Association for Computational Linguistics.

Nora Kassner and Hinrich Schütze. 2020. Negated and misprimed probes for pretrained language models: Birds can talk, but cannot fly. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7811–7818, Online. Association for Computational Linguistics.

Hang Le, Loïc Vial, Jibril Frej, Vincent Segonne, Maximin Coavoux, Benjamin Lecouteux, Alexandre Allauzen, Benoit Crabbé, Laurent Besacier, and Didier Schwab. 2020. FlauBERT: Unsupervised language model pre-training for French. In *Proceedings of the 12th Language Resources and Evaluation Conference*, pages 2479–2490, Marseille, France. European Language Resources Association.

Xiaodong Liu, Pengcheng He, Weizhu Chen, and Jianfeng Gao. 2019. Multi-task deep neural networks for natural language understanding. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 4487–4496, Florence, Italy. Association for Computational Linguistics.

I. Loshchilov and F. Hutter. 2017. Fixing weight decay regularization in adam. *ArXiv*, abs/1711.05101.

Tom McCoy, Ellie Pavlick, and Tal Linzen. 2019. Right for the wrong reasons: Diagnosing syntactic heuristics in natural language inference. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 3428–3448, Florence, Italy. Association for Computational Linguistics.

Nikita Nangia, Clara Vania, Rasika Bhalerao, and Samuel R. Bowman. 2020. CrowS-pairs: A challenge dataset for measuring social biases in masked language models. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1953–1967, Online. Association for Computational Linguistics.

Timothy Niven and Hung-Yu Kao. 2019. Probing neural network comprehension of natural language arguments. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 4658–4664, Florence, Italy. Association for Computational Linguistics.

Sungjoon Park, Jihyung Moon, Sungdong Kim, Won Ik Cho, Jiyoon Han, Jangwon Park, Chisung Song, Junseong Kim, Yongsook Song, Taehwan Oh, et al. 2021. KLUE: Korean language understanding evaluation. *arXiv preprint arXiv:2105.09680*.

Mohammad Taher Pilehvar and Jose Camacho-Collados. 2019. WiC: the word-in-context dataset for evaluating context-sensitive meaning representations. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 1267–1273, Minneapolis, Minnesota. Association for Computational Linguistics.

Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. 2019. Language models are unsupervised multitask learners. *OpenAI blog*, 1(8):9.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J Liu. 2020. Exploring the limits of transfer learning with a unified text-to-text transformer. *Journal of Machine Learning Research*, 21:1–67.

Marco Tulio Ribeiro, Tongshuang Wu, Carlos Guestrin, and Sameer Singh. 2020. Beyond accuracy: Behavioral testing of NLP models with CheckList. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 4902–4912, Online. Association for Computational Linguistics.

Melissa Roemmele, Cosmin Adrian Bejan, and Andrew S Gordon. 2011. Choice of plausible alternatives: An evaluation of commonsense causal reasoning. In *2011 AAAI Spring Symposium Series*.

Sang Kyu Seo. 2007. The 21st century sejong project special corpus construction (1998 2007). In *Annual Conference on Human and Language Technology*, pages 317–322.

Sang Kyu Seo and Hyeong Jeong Kim. 2005. On some practical conditions for developing spoken language corpus. *Language Facts and Perspectives*, 16:5–29.

Jidong Tian, Yitian Li, Wenqing Chen, Liqiang Xiao, Hao He, and Yaohui Jin. 2021. Diagnosing the first-order logical reasoning ability through LogicNLI. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 3738–3747, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Alex Wang, Yada Pruksachatkun, Nikita Nangia, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel Bowman. 2019. SuperGLUE: A stickier benchmark for general-purpose language understanding systems. In *Advances in Neural Information Processing Systems*, volume 32. Curran Associates, Inc.

Alex Wang, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel Bowman. 2018. GLUE: A multi-task benchmark and analysis platform for natural language understanding. In *Proceedings of the 2018 EMNLP Workshop BlackboxNLP: Analyzing and Interpreting Neural Networks for NLP*, pages 353–355, Brussels, Belgium. Association for Computational Linguistics.

Bryan Wilie, Karissa Vincentio, Genta Indra Winata, Samuel Cahyawijaya, Xiaohong Li, Zhi Yuan Lim, Sidik Soleman, Rahmad Mahendra, Pascale Fung, Syafri Bahar, and Ayu Purwarianti. 2020. IndoNLU: Benchmark and resources for evaluating Indonesian natural language understanding. In *Proceedings of the 1st Conference of the Asia-Pacific Chapter of the Association for Computational Linguistics and the 10th International Joint Conference on Natural Language Processing*, pages 843–857, Suzhou, China. Association for Computational Linguistics.

Sen Wu, Hongyang R Zhang, and Christopher Ré. 2020. Understanding and improving information transfer in multi-task learning. In *International Conference on Learning Representations*.

Liang Xu, Hai Hu, Xuanwei Zhang, Lu Li, Chenjie Cao, Yudong Li, Yechen Xu, Kai Sun, Dian Yu, Cong Yu, Yin Tian, Qianqian Dong, Weitang Liu, Bo Shi, Yiming Cui, Junyi Li, Jun Zeng, Rongzhao Wang, Weijian Xie, Yanting Li, Yina Patterson, Zuoyu Tian, Yiwen Zhang, He Zhou, Shaoweihua Liu, Zhe Zhao, Qipeng Zhao, Cong Yue, Xinrui Zhang, Zhengliang Yang, Kyle Richardson, and Zhenzhong Lan. 2020. CLUE: A Chinese language understanding evaluation benchmark. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 4762–4772, Barcelona, Spain (Online). International Committee on Computational Linguistics.

Zhengyuan Yang, Zhe Gan, Jianfeng Wang, Xiaowei Hu, Yumao Lu, Zicheng Liu, and Lijuan Wang. 2021. An empirical study of GPT-3 for few-shot knowledge-based VQA. *arXiv preprint arXiv:2109.05014*.

Rowan Zellers, Ari Holtzman, Yonatan Bisk, Ali Farhadi, and Yejin Choi. 2019. HellaSwag: Can a machine really finish your sentence? In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 4791–4800, Florence, Italy. Association for Computational Linguistics.

A Appendix

KB-BoolQ	**Inputs**: Paragraph: 구한말, 통영 안뒤산 기슭 간창골에 김봉제 형제가 살았다. 김봉제는 관약국을 경영하며 부를 누렸는데, 선비적 성품을 지녔던 형과 반대로 막냇동생 김봉룡은 성질이 포악했다. 어느 날 봉룡은 아내였던 숙정을 사모하던 나그네를 살해하였고, 숙정은 누명을 벗으려고 비상을 먹고 자살한다. Question: 봉룡은 숙정을 죽였는가? Answer: False **Prmopt Design**: Answer: "예" if Answer is True else "아니오" Format: "{Paragraph} 질문: {Question} 답변: {Answer}." **Example**: 구한말, 통영 안뒤산 기슭 간창골에 김봉제 형제가 살았다. 김봉제는 관약국을 경영하며 부를 누렸는데, 선비적 성품을 지녔던 형과 반대로 막냇동생 김봉룡은 성질이 포악했다. 어느 날 봉룡은 아내였던 숙정을 사모하던 나그네를 살해하였고, 숙정은 누명을 벗으려고 비상을 먹고 자살한다. 질문: 봉룡은 숙정을 죽였는가? 답변: 아니요.
KB-COPA	**Inputs**: Premise: 전쟁이 시작되었다. Question: 결과 Answer Alternative: 병사들이 전투에 파견되었다. **Prompt Design**: Connector: "왜냐하면" if Question is "원인" else "그래서" Format: "{Premise} {Connector} {Answer Alternative}" **Example**: 전쟁이 시작되었다. 그래서 ("왜냐하면" if question is 원인) 병사들이 전투에 파견되었다.
KB-WiC	**Inputs**: Context 1: 망가진 엔진은 수리가 불가능하다. Context 2: 이 배는 수리에 들어간 지 일주일이 됐다. Target Word: 수리 Answer: True **Prompt Design**: Answer: "예" if Answer is True else "아니오" Format: "문장1: {Context1} 문장2: {Context2} 두 문장에서 {Target Word}가 같은 뜻으로 쓰였나? {Answer}" **Example**: 문장1: 망가진 엔진은 수리가 불가능하다. 문장2: 이 배는 수리에 들어간 지 일주일이 됐다. 두 문장에서 수리가 같은 뜻으로 쓰였나? 예.
KB-HellaSwag	**Inputs**: Context: 양궁 선수들이 경기장으로 입장한다. 관중들이 함성을 지르고 응원한다. 선수들이 상대팀과 악수하고 자리로 돌아온다. 코치가 전략을 설명하고 화이팅을 외친다. Correct Ending: 선수들이 각자 자리에 서서 활을 꺼낸다. **Prompt Design**: 문장: {Context} {Correct Ending} **Example**: 문장: 양궁 선수들이 경기장으로 입장한다. 관중들이 함성을 지르고 응원한다. 선수들이 상대팀과 악수하고 자리로 돌아온다. 코치가 전략을 설명하고 화이팅을 외친다. 선수들이 각자 자리에 서서 활을 꺼낸다.
KB-SentiNeg	**Inputs**: Sentence: 뚜껑이 잘 안열려요! Answer: 부정 **Prompt Design**: Format: "문장: {Sentence} 긍부정: {Answer}" **Example**: 문장: 뚜껑이 잘 안열려요! 긍부정: 부정

Table 8: Prompt designs for each task used in in-context learning experiments. Example data points presented in Table 2 are used.

A New Public Corpus for Clinical Section Identification: MedSecId

Paul Landes[†], Kunal Patel[‡], Sean S. Huang[◇],
Adam Webb[‡], Barbara Di Eugenio[†], and Cornelia Caragea[†]

[†]Department of Computer Science, University of Illinois at Chicago
[‡]Department of Emergency Medicine, University of Illinois at Chicago
[◇]Department of Internal Medicine and Geriatrics, University of Illinois at Chicago
{plande2,kpate318,sshuang,awebb25,bdieugen,cornelia}@uic.edu

Abstract

The process by which sections in a document are demarcated and labeled is known as section identification. Such sections are helpful to the reader when searching for information and contextualizing specific topics. The goal of this work is to segment the sections of clinical medical domain documentation. The primary contribution of this work is MedSecId, a publicly available set of 2,002 fully annotated medical notes from the MIMIC-III. We include several baselines, source code, a pretrained model and analysis of the data showing a relationship between medical concepts across sections using principal component analysis.

1 Introduction

Most unstructured medical text found in electronic health record systems (EHRs) written by medical staff have conceptually well defined sections. For example, discharge summaries are technical medical documents, written by physicians when the patient is discharged, which describe the patient's hospital stay and surrounding circumstances of their illness. As shown by the example in Figure 1, discharge summaries consist of named sections, typically in a specific sequence, such as the *History of Present Illness*; this type of section appears both in discharge summaries and in physician notes that describe a chronology of an illness that begins with the admission of the patient.

Whereas sections often have headers, section identification (SI) is more challenging than simply parsing the first several leading header tokens of the respective section (underlined in Figure 1). While the first several tokens can be helpful in identifying a section, their naming often varies. For example, the 6th section in Figure 1 starts with header tokens *Preoperative Laboratory Data*, but the section type is `labs-imaging`. There are also cases where the header tokens are missing, as shown in the 7th section (`hospital-course`) in the same figure, or where sections have several header text spans placed throughout the section. Adding to this challenge is the non-uniformity of the text, lack of section boundary syntax, and copy-pasted text from other notes or from structured data such as patient vital signs.

Admission Date: [**2126-2-7**] Discharge Date: [**2126-2-20**]
Date of Birth: [**2069-4-1**] Sex: M

history-of-present-illness
HISTORY OF PRESENT ILLNESS: Mr. [**Known last-name **] is a 56-year-old male who experienced chest...

past-medical-history
PAST MEDICAL HISTORY: Hypertension, former smoker with a 4- pack per day history for which he...

social-history
SOCIAL HISTORY: He lives alone, and he works at [**Hospital3 2576**] as a cargo transporter.

medication-history
MEDICATIONS ON ADMISSION: Aspirin 325 mg p.o. once a day, Toprol-XL 50 mg p.o. once a day.

allergies
ALLERGIES: He had no known drug allergies.

labs-imaging
PREOPERATIVE LABORATORY DATA: White count 6.0, hematocrit 33.3, platelet count 329,000....

hospital-course
On exam he had a left facial droop, status post his childhood polio. Temperature of 97.5, heart rate 65 in sinus...

discharge-diagnosis
DISCHARGE DIAGNOSES:
1. Status post coronary artery bypass grafting x 3....

discharge-instructions
DISCHARGE INSTRUCTIONS: He was instructed to make an appointment...

discharge-medications
MEDICATIONS ON DISCHARGE:
1. Aspirin enteric coated 81 mg p.o. once a day.
2. Colace 100 mg p.o. twice a day...

He was discharged to home with VNA services in good condition on [**2126-2-20**]...

Figure 1: A MIMIC-III discharge summary note with section type in **bold**, header tokens underlined, and text not belonging to any section grayed out; omitted text is indicated with ellipses.

Proceedings of the 29th International Conference on Computational Linguistics, pages 3709–3721
October 12–17, 2022.

While discharge summary sectioning helps a physician locate specific information, the primary impetus for the structure and content stems from the ongoing dispute between providers and healthcare insurance companies in the United States. Providers are limited by how much they can bill for relatively simple medical procedures, but increasingly complex procedures garner more revenue with proper documentation. Specifically, medical billing staff and insurance companies use relative value units (RVUs), which is a monetary unit updated annually and currently set at $34.30. The number of RVUs billed is based on the composition and number of sections included in the medical notes per guidelines set by the Centers for Medicare and Medicaid Services[1].

For this reason, providers are encouraged to write medical notes to maximize RVUs out of necessity (Barnes et al., 2008) even though physician training lacks such emphasis. In contrast, medical residents are evaluated with the objective structured clinical examination (OSCE), which is a student examination that evaluates students based on direct observation (Zayyan, 2011). However, the exam's evaluation with respect to medical note authoring and structure uses a very different criteria and omits RVUs (Gallagher et al., 2020). The necessity of a particular structure in medical notes, for the purpose of patient care and arguably more important insurance billing requirements, highlights the need for understanding sectioning.

However, the motivation for understanding SI is not limited to the medical field, it has a bearing on other medical NLP tasks. Since each section contains specific information, SI is often the first step in a medical NLP pipeline and can lead to downstream propagation errors causing poor task specific results if not properly executed. Examples of downstream tasks that benefit from SI include medical summarization, entity linking and natural language understanding and extraction.

While academic text segmentation has garnered interest (Hirohata et al., 2008), no publicly available medical SI annotated corpora exists (Pomares-Quimbaya et al., 2019). For this reason, we believe MedSecId is the first medical section identification dataset. It was created from 2,002 medical notes annotated by two attending physicians and one senior resident physician at the University of Illinois Chicago (UI Health). The annotation dataset is comprehensive with 2,558K annotated tokens or 97.3% of the entire corpus (see Table 1).

Description	Count
Documents	2,002
Annotations	22,561
Annotated Sentences	259,286
Total Tokens	2,630,525
Annotated Tokens	2,558,219

Table 1: Annotation dataset statistics.

The contributions of this work include: a) a comprehensive publicly available medical section annotation dataset, b) baselines with three models and several contextual and non-contextual word embeddings, c) an ontology of note to section relationships, d) human readable descriptions of medical notes and all sections annotated (see Appendix A), e) a pretrained model for each baseline, f) code to reproduce the results and read the annotations, and g) a command line tool to predict note annotations using any of the baseline models.

2 Related Work

Sectioning MedLINE abstracts was explored by McKnight and Srinivasan (2003) using a support vector machine (SVM). This classifier was used to label sentences as *Introduction*, *Method*, *Result*, or *Conclusion* and showed promising results using a bag-of-words approach. Sequence based approaches (Hirohata et al., 2008) were also used to section scientific abstracts into *Objective*, *Methods*, *Results*, and *Conclusion* labels using a conditional random field (CRF) model producing a sentence level accuracy of 95.5%.

While academic abstract segmentation was a well explored area (Hirohata et al., 2008), Tepper et al. (2012) were the first to apply statistical methods to the medical domain to automatically classify sections of clinical free text into sections. Their method used in, out, begin (IOB) annotation (Ramshaw and Marcus, 1995) with labels to mark named sections. For example, `B-HPI` indicates a beginning token for the *History of Present Illness* section. Their dataset consisted of annotating the 2010 i2b2 corpus with a section header and medical ontology label, and obtained an F-measure of 0.92 for the concept extraction task (Uzuner et al., 2011; de Bruijn et al., 2010). A Maximum Entropy (MaxEnt) model (Berger et al., 1996) and beam search were used for classification to produce the IOB sequence for token tagging.

[1] https://www.cms.gov/Regulations-and-Guidance

Along with MaxEnt, other non-neural network methods, such as SVM and CRF models continue to be popular with few exceptions as detailed in the comprehensive survey of Pomares-Quimbaya et al. (2019). One such exception (Sadoughi et al., 2018) used a long-short term memory (LSTM) model with word-to-vector (word2vec) embeddings (Mikolov et al., 2013a,b) for a binary classification of section boundaries. Even though the corpus consists of dictated and transcribed notes, they show that neural methods work for the section segmentation task. Other notable neural network (NN) text segmentation works use convolutional neural networks (CNNs) over sentence embeddings with a softmax over the output of a bi-directional long-short term memory (BiLSTM) layer to demarcate sections as a binary classification across both medical and non-medical datasets (Badjatiya et al., 2018). Barrow et al. (2020) also used a LSTM in a network that aggregates features across fastText word embeddings using a concatenated segment pooling LSTM (S-LSTM) for non-medical Wikipeda articles (Bojanowski et al., 2017).

The work of Nair et al. (2022) most closely resembles our SI work. However, their model classifies only the four SOAP (Subjective, Objective, Assessment and Plan) sections available in the corpus leaving the others as future work. Their methods also only have been tested against the Flair framework, which uses concatenated static word embeddings that are fine tuned locally for the task on the 2010 i2b2 corpus. Our method includes fine-tuning the BERT embeddings themselves as an end-to-end joint learning process. Additionally, they have provided no annotation pipeline or process to create a semi-supervised or bootstrapped corpus. Our work includes medical domain specific experiments with various word embedding combinations and novel data analysis using the Unified Medical Language System (UMLS) (Bodenreider, 2004) and *cui2vec* (Beam et al., 2020) (see Section 3.3). It also includes other methods and network experimental configurations the authors have not yet tried as they used the Flair framework "out of the box". Another significant difference is their annotations are not available[2] while we classify 50 sections and provide our code with annotations publicly.

[2]The authors did not respond to our request for obtaining their corpus for baseline comparison.

3 Dataset

MedSecId is a subset of the MIMIC-III version 1.4 corpus (Johnson et al., 2016) that we annotated; MIMIC-III is publicly available[3] and consists of critical care unit EHR records from the Beth Israel Deaconess Medical Center in Boston, Massachusetts. The dataset contains 58,976 hospital admissions across 46,520 patients who were admitted to the intensive care unit (ICU) surgical, medical, and neonatal departments. It includes 2,083,180 unstructured medical text notes handwritten by medical professionals across several disciplines and contains 15 categories, such as discharge summaries and radiology notes.

We created a curated annotation set consisting of text spans taken from a random sample across five categories of MIMIC-III medical notes[4], including discharge summaries, *Radiology*, *Consult* , *Echo*, and *Physician* progress notes (see Table 2). Each text span contains the type of the section, such as *History of Present Illness*, with zero-index character offsets of where the span starts and ends in the note.

Category	Count	Proportion
Discharge summary	1,254	62.64%
Physician	288	14.39%
Radiology	205	10.24%
Echo	198	9.89%
Consult	57	2.85%
Total	2,002	100%

Table 2: Annotated medical notes by category and their distribution in the annotation set.

While each section contains a single type, sections have zero or more overlapping header text spans (see Figure 1). In most cases, there is a single header span, but vital signs sections can "float" without a physical exam header. These header spans consist of text that identify the section such as *History Of Present Illness*, an alternate spelling or abbreviation such as *HPI*. Even though single header spans usually appear at the beginning of a section, additional section headers are found later in the body indicating subsections in some cases. Since section type inclusion highly varies based on the patient's age, notes were annotated with an age type (adult, pediatric or neonatal patient), based on the content of the note by our annotator.

[3]Access to the MIMIC-III corpus requires creating a PhysioNet account and finishing a training course.

[4]The unstructured medical note data was taken from the NOTEEVENTS table.

Type	Tokens	Spans	Notes
physical-examination	203K (8%)	1,385 (6%)	Consult, Physician
history-of-present-illness	239K (9%)	1,348 (6%)	Consult, Discharge summary, Physician
allergies	9,221 (0%)	1,205 (5%)	Consult, Discharge summary, Physician
hospital-course	692K (26%)	1,165 (5%)	Discharge summary
labs-imaging	416K (16%)	1,155 (5%)	Consult, Discharge summary, Physician
past-medical-history	60K (2%)	1,141 (5%)	Consult, Discharge summary, Physician
discharge-condition	14K (1%)	1,132 (5%)	Discharge summary
discharge-instructions	183K (7%)	1,077 (5%)	Discharge summary
discharge-diagnosis	34K (1%)	1,040 (5%)	Discharge summary
chief-complaint	9,622 (0%)	996 (4%)	Consult, Discharge summary, Physician
discharge-medications	196K (7%)	914 (4%)	Discharge summary
social-history	28K (1%)	912 (4%)	Consult, Discharge summary, Physician
medication-history	49K (2%)	867 (4%)	Consult, Discharge summary, Physician
family-history	11K (0%)	802 (4%)	Consult, Discharge summary, Physician
discharge-disposition	5,602 (0%)	754 (3%)	Discharge summary
major-surgical-or-invasive-procedure	16K (1%)	704 (3%)	Discharge summary
facility	2,668 (0%)	502 (2%)	Discharge summary
reason	5,588 (0%)	458 (2%)	Consult, Radiology
findings	58K (2%)	395 (2%)	Echo, Radiology
assessment-and-plan	131K (5%)	381 (2%)	Consult, Physician
review-of-systems	7,422 (0%)	329 (1%)	Consult, Discharge summary, Physician
image-type	1,820 (0%)	328 (1%)	Radiology
last-dose-of-antibiotics	3,689 (0%)	293 (1%)	Consult, Physician
24-hour-events	16K (1%)	250 (1%)	Physician
code-status	1,879 (0%)	237 (1%)	Physician
impression	8,233 (0%)	224 (1%)	Echo, Radiology
disposition	1,161 (0%)	210 (1%)	Physician
conclusions	28K (1%)	206 (1%)	Echo
communication	1,304 (0%)	199 (1%)	Physician
patient-test-information	13K (1%)	198 (1%)	Echo

Table 3: The top 30 most frequently annotated sections.

3.1 Annotation Process

Our annotation process consisted of several preliminary rounds of annotation, that led to our final annotation guidelines and final annotation.

Before annotation began, a custom set of regular expressions were used to pre-annotate, similar to previous work (Shivade et al., 2015); ours were medical note specific and captured header tokens along with the section spans. The application of the regular expressions was only a means to reduce the work of the annotators, who followed the annotation guide regardless of any rule based pre-annotations. The initial rule based automatic annotation process was amended by the work of Alsentzer and Kim (2018), who generously shared their *History of Present Illness* annotations to better identify and segment the initial dataset used by our annotators. These automatic annotations were edited by the annotators after they were imported into INCEpTION (Klie et al., 2018) and saved to later compute an inter-coder agreement between the physicians and rule-based output (see Section 3.2).

An attending physician (designated as a primary annotator) co-wrote a preliminary annotation guide with input from a secondary physician annotator. These two annotators engaged in a process of annotation, discussion and revision of the guidelines: they annotated a first set of one hundred notes, revised the guidelines, annotated a second set of one hundred notes, and finalized the guidelines after this second round.

Here we summarize the issues that the annotators faced during these preliminary rounds of annotation. This process was useful for the physicians to reach a consensus on what sections should be annotated and agreed on section types given their experience writing such notes themselves. A set of sections and their relation to notes began to coalesce during this process, which provided the motivation to create an ontology for the purpose of a meta documentation about the annotations and the utilitarian purpose to assist in annotation by importing it as a "knowledge base" in INCEpTION. The ontology consisted of a one-to-many mapping from notes to 50 section types using each section's header tokens captured by the regular expressions by string massaging. For example, *History of Present Illness* became `history-of-present-illness`. Among the categories, 29 sections were shared

across more than one note, such as *History of Present Illness* shared between notes *Discharge summary*, *Consult*, and *Physician* (see Table 3 for annotated sections and Appendix A for full listings).

	A1	A2	A3	R
A1	1.0	0.81	0.87	0.73
A2		1.0	0.84	0.49
A3			1.0	0.53
R				1.0

Table 4: Krippendorff's α coefficient of interannotator agreement between the annotators and the regular expressions. **A1** is the primary annotator, **A2** is the secondary annotator and **A3** is the third annotator, and **R** represents the regular expressions.

Each section type was then agreed on by the physicians with many re-typed and regrouped. For example, *Echo* notes contained internal subsections for each chamber of the heart, and was resolved by grouping the entire section as *Findings* to match section types in *Radiology* notes. Other subsections implicitly resulted by physicians copying radiology findings in discharge summaries. In an effort to reduce complexity, a flat note-to-section hierarchy without creating a second section level was kept. In some cases this was achieved by combining laboratory results data with radiology findings/diagnosis as a single section by simply re-casting *Labs* to *Labs/Radiology* for sections that included imaging studies. Other sections needed to be combined as not all notes had a clean separation.

To accommodate for a significant variation in how physicians labeled sections in these situations, *Labs* and *Radiology* was combined into a *Labs/Radiology* section. *Labs* and *Imaging* were also combined into *Labs/Imaging*. Since discharge summaries typically incorporate instructions for the patient and follow up information, we categorized these together broadly as *Discharge instructions*. The MIMIC-III pseudo tokens, such as `[**First Name**]` were not annotated unless they were included in the body of the section.

The primary and secondary annotators finished revising the annotation guidelines and then trained the third annotator. A subset of 80 medical notes, chosen from the second batch of 100 that the primary and secondary annotator had annotated and discussed, was used to train the third annotator. Because these first two batches were only used for creating guidelines and training, they were not added in the final annotation set. During this process, the well known Krippendorff's α coefficient (Krippendorff, 2011), was used to compute inter-annotator agreement (IAA) between this last annotator and the other two, until α became higher than 0.8.

3.2 Final Annotation and IAA Computation

Once the guidelines were finalized the final annotation process started. A set of 100 notes (different than the sets discussed in Section 3.1) was held out to compute the inter-annotator agreement (IAA) on the final guidelines. The remaining 1,902 notes were divvied up among the three annotators, as customary.

Inter-annotator agreement was calculated on the 100 held out notes as exact section character offsets and section types—both the offsets and the section type had to match to be considered correct. This agreement was calculated among the human annotators, and subsequently between each annotator and the regular expressions that were initially used to segment the notes.

Among humans, Krippendorff's α yielded more than acceptable values of 0.84 to 0.87 on the final set held out for this IAA calculation (see Table 4). At this point, these annotations were added to the final dataset by selecting notes with the fewest issues[5] using the primary annotator as the tie-breaker.

While we achieved a high inter-coder agreement among human annotators, we found troubling data in terms of the performance of the regular expression annotation approach. We computed an aggregate Krippendorff's α=0.54 between the human physician annotators and our custom regular expressions (see Section 3.1) on the final annotated data, which falls more than 14 points shy of the "lowest conceivable limit" of 0.68 (Krippendorff, 2004). This shows how regular expression's performance to segment notes falls short of that by human annotators (see Table 4), yet regular expressions continue to be the most common methods used for section identification (Pomares-Quimbaya et al., 2019; Shivade et al., 2015).

In part, the regular expressions often failed to demarcate the entire section, especially in text with irregular formatting toward the end. Furthermore, additional analysis shows the α scores between individual annotators and the regular expressions are low as well, albeit with a fairly high variance. Krippendorff suggests that acceptable scores that are

[5]Issues included placement of header tokens and missing sections. For example, an annotation with a defined section would win over another's annotation with the section type.

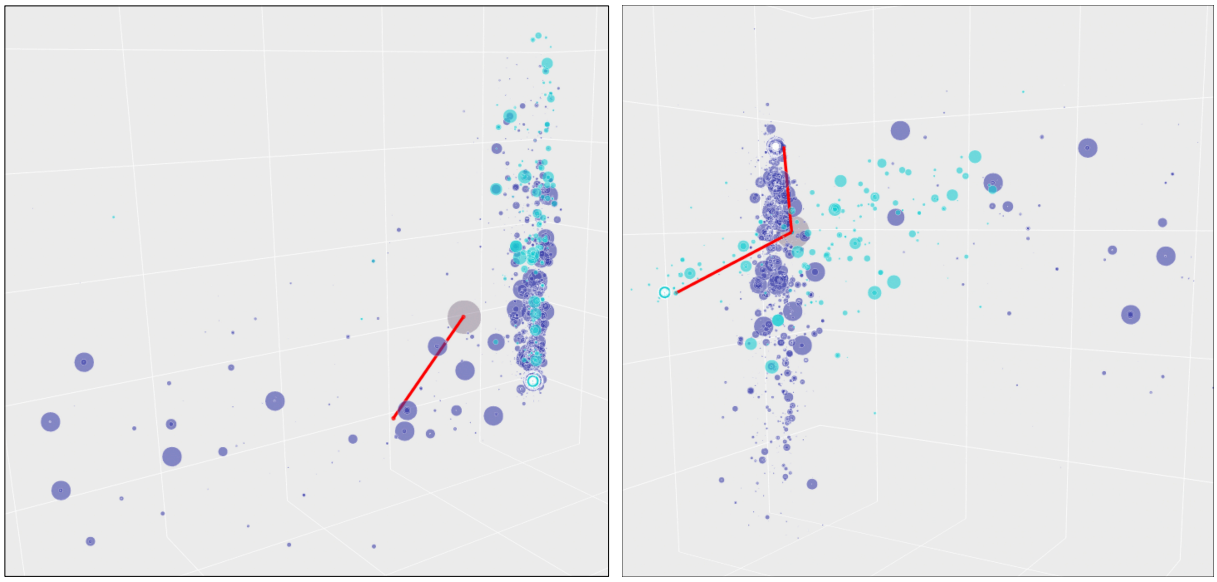

a) b)

Figure 2: Concept unique identifier (CUI) Plots: a) plot of `past-medical-history` (purple) and `past-surgical-history` (blue) reduced to 3D together as one data set with the first principal component (red line) with data point size as the TF/IDF score, b) plot of the same sections but reduced to 3D as separate data points with respective first principal components.

"customary to require" have $\alpha > 0.8$ (Krippendorff, 2004). On one hand, an α of 0.73 between physician *A1* and the automatic regular expression annotator *R* clears the minimal limit threshold. However, this metric falls well below the "aimed" score of 0.8. The larger issue is with physician *A2*'s and *A3*'s scores of 0.49 and 0.53, which fall short of the minimum limit by a large margin. From these scores (see Table 4) and the low overall α, we conclude regular expressions do not sufficiently segment medical notes, therefore the annotation set we provide should be considered the gold standard for medical note identification and segmentation.

3.3 Data Analysis

An interesting discovery concerned projections of medical conditions across sections in embedded space. Concept unique identifiers (CUIs) were extracted using MedCAT (Kraljevic et al., 2021) and weighted by TF/IDF (Sparck Jones, 1972) across sections. Each CUI was mapped to a vector from *cui2vec* embeddings, and then reduced to three dimensions using principal component analysis (PCA), shown in Figure 2. The plot was generated without normalizing or standardizing the data so CUI vector magnitudes were retained for analysis. Figure 2 (a) shows the `past-medical-history` section (purple) CUIs on the horizontal axis with `past-surgical-history` (blue) CUIs only on the vertical axis with size proportional to TF/IDF.

The past surgical and medical history sections in discharge summary notes project many medical disease CUIs as orthogonal to surgical CUIs. The medical disease CUIs on the vertical axis are those that do not have surgery as a treatment option, such as hypertension. However, a CUI representing coronary artery disease that plots along the surgical history vertical axis does require surgery. Most of the data points that share the vertical axis along with `past-medical-history` are those that require both medication and surgery, such as cancer.

Not only does this show *cui2vec* being used in practice for the first time, it illustrates an application of how groupings of concepts can be visualized and analyzed to gain intuition and insight in complex medical data. In our data, this includes not only a semantic relationship between concepts, but how those concepts represent the treatments involved based on the section from which they originate. Given this data relationship, we hypothesize that utilization of *cui2vec* embeddings, such as concatenating them to word vectors, will increase performance of task specific models including SI.

3.4 Limitations

MedSecId is limited to notes (with the exception of the discharge summary) of patients admitted to an ICU from the MIMIC-III corpus for several

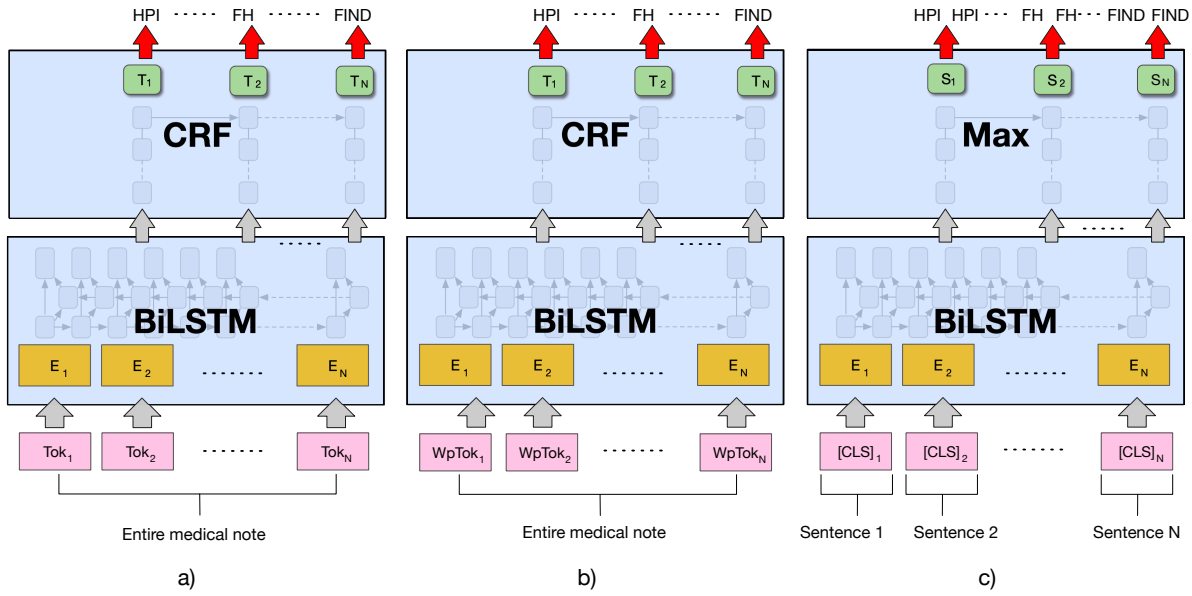

Figure 3: Baseline models: a) BiLSTM-CRF$_{tok}$ BiLSTM model with non-contextual token input embeddings, b) BERT-CRF$_{tok}$ BiLSTM model with BERT word piece token fixed input embeddings, c) BERT$_{sent}$ BiLSTM model with `[CLS]` sentence embeddings using the per sentence majority label.

note categories with no data included after the patient leaves to a lower severity department[6]. Notes written afterward are an essential source of data that provides an aspect of the patients' stay that is otherwise lacking in the corpus, such as daily progress *Physician* notes. However, *Radiology* and *Echo* notes from the MIMIC-III corpus apply to all hospital departments since they are uniform for all patients, regardless of their location, outpatient or inpatient. In addition, discharge summaries entail the entire hospital visit, including the ICU and the remainder of the admission.

3.5 Implementation Details

The annotation set was randomly sampled per note and divided as a stratified dataset into training (80%), validation (10%) and test (10%) datasets. The medical note structure ontology (see Section 3.1) is distributed as both a RDF Turtle file and a CSV file along with the annotations. The publicly available[7] code to train, validate, and test the model also includes additional APIs to access the annotated data, perform inference with the pretrained model or train a new model. This codebase includes functionality to use the pretrained model or utilize the annotations for experimentation and is ready to easily be installed.[8] This codebase also references a related project useful for parsing MIMIC-III text, pseudo token replacement, and Postgres database to Python object relational mapping.

4 Methods

Because the section text spans do not break on tokens, we cast our task as a named entity recognition (NER) using in, out (IO) encoding[9] on a 50 way classification including `<none>` for text with no sections (see Table 7). Using this encoding, we created several baselines across two BiLSTM models[10] for the purpose of future work benchmarking. These baselines include majority label metrics, a token BiLSTM-CRF, and a sentinel BERT embedding (Devlin et al., 2019) LSTM model (see Figure 3). Aside from adjusting the LSTM hidden size, gradient clipping, and number of epochs, all parameters were held constant across all experiments (see Appendix B for all hyperparameters used).

BiLSTM-CRF$_{tok}$ The token model consists of a simple non-contextual input word embeddings, a LSTM layer and fully connected linear layer using a CRF output with labels assigned by the Viterbi

[6]Only five note categories are available (see Table 2).
[7]https://github.com/uic-nlp-lab/medsecid
[8]All that is required in a `pip install`. See the GitHub repo for details.
[9]IOB encoding was not used as there are no transitions from one section to another and to reduce the label count.
[10]No models use a BERT transformer, only BERT token and sentinel ([CLS]) embeddings.

Id	Name	mF1	mP	mR	MF1	MP	MR
1	Majority Label	0.023	0.023	0.023	0.0	0	0.005
2	BERT$_{sent}$	0.925	0.925	0.925	0.589	0.616	0.6
3	BiLSTM-CRF$_{tok}$ (word2vec)	0.927	0.927	0.927	0.778	0.78	0.801
4	BERT-CRF$_{sent}$	0.929	0.929	0.929	0.689	0.734	0.7
5	BERT$_{sent}$ BioBERT	0.94	0.94	0.94	0.687	0.73	0.679
6	BERT-CRF$_{sent}$ BioBERT	0.94	0.94	0.94	0.705	0.757	0.704
7	BiLSTM-CRF$_{tok}$ (GloVE 50D)	0.954	0.954	0.954	0.76	0.783	0.765
8	BiLSTM-CRF$_{tok}$ fastText	0.954	0.954	0.954	**0.796**	0.806	0.806
9	BiLSTM-CRF$_{tok}$ (GloVE 300D)	**0.955**	0.955	0.955	0.787	0.801	0.788

Table 5: Summarization of performance metrics where mF1 is the micro F1, mP is the micro precision, mR is the micro recall, MF1 is the macro F1, MP is the macro precision, MR is the macro recall.

algorithm. Several embeddings were used with this model, including word2vec (Mikolov et al., 2013a,b), Global Vectors for Word Representation (GLoVe) (Pennington et al., 2014) and fastText (Bojanowski et al., 2017) (Crawl) embeddings.

BERT$_{\text{sent}}$ To address the issue of exploding gradients, we created a sentence-based model using static BERT sentinel embeddings to lower the input length to the LSTM layer. The model assumes sections rarely break mid-sentence since every sentence is assigned one section. Sentences with more than one section annotation will lower end-to-end performance. However, 97.6% of the annotation set contains sentences with a single section for all tokens of the respective sentence as shown in Table 6. The output of the final layer of the first time step was used as the input to a LSTM. The LSTM output forwarded to a dense layer with one output neuron for each label and an output max over the label.

Unique Sections	Count	Proportion
1	253025	97.59%
2	5589	2.16%
3	589	0.23%
4	72	0.03%
5	11	0.00%

Table 6: Distribution of sentences having a single section label across all tokens of the respective sentence.

Both the standard small BERT model and BioBERT embeddings (Lee et al., 2020) are included in the baseline results (see Section 5). A ClinicalBERT baseline model (Alsentzer et al., 2019) would not provide a fair baseline metric for comparison with future works since it trained on the MIMIC-III corpus so it was excluded.

BERT-CRF$_{\text{sent}}$ Like BERT$_{\text{sent}}$, but adds a CRF layer with Viterbi assigned labels.

5 Results

The baseline models described in Section 4 were each trained until the validation loss converged, then early stopped. The results are summarized in Table 5 with label specific results in Table 7. We report performance metrics by counting correct predictions when the character span boundaries match exactly and the sections type match. If either do not match, it is counted as an incorrect prediction.

From the majority label, it's clear the models perform comparatively well as shown in the summary results in Table 5. The GLoVe model has the best micro F1 of 0.96 with the fastText model having the best macro F1 of 0.8. This 16 point spread is evident from how performance drops off for the bottom 13 section types. Many of these low performers are those that were re-casted or re-grouped (see Section 3.1), and could be regrouped to an umbrella section type like *Labs/Imaging/Radiology* if such a rigorous delineation was not necessary.

The BERT$_{\text{sent}}$ does not lag far behind, but its performance using sentinel embeddings does not capture sections as well as the token level models despite long document length. Performance significantly improved and models converged faster with the use of gradient clipping to alleviate issues of LSTM exploding gradients (Bengio et al., 1994).

6 Conclusions and Future Work

We presented MedSecId, a comprehensive dataset of 2,002 medical annotations from the MIMIC-III corpus across five note types and 50 sections. The dataset contains section types, headers and patient age annotations. Our dataset shows promising baseline results from simple models such as BiLSTMs with diverse inputs, but still leaves room for improvement by more sophisticated models.

We expect performance using our models to improve pipelines that use rule based methods for

Id	Label	mF1	mP	mR	MF1	MP	MR	Acc	Count
1	procedure	0	0	0	0	0	0	0	156
2	labs	0	0	0	0	0	0	0	436
3	prenatal-screens	0.276	0.276	0.276	0.216	0.5	0.138	0.276	105
4	imaging	0.357	0.357	0.357	0.263	0.5	0.178	0.357	990
5	comparison	0.414	0.414	0.414	0.195	0.333	0.138	0.414	222
6	code-status	0.513	0.513	0.513	0.226	0.333	0.171	0.513	150
7	wet-read	0.521	0.521	0.521	0.342	0.5	0.26	0.521	121
8	communication	0.556	0.556	0.556	0.179	0.25	0.139	0.556	133
9	impression	0.563	0.563	0.563	0.18	0.25	0.141	0.563	920
10	disposition	0.647	0.647	0.647	0.262	0.333	0.216	0.647	68
11	history	0.688	0.688	0.688	0.272	0.333	0.229	0.688	170
12	past-surgical-history	0.745	0.745	0.745	0.427	0.5	0.372	0.745	145
13	current-medications	0.746	0.746	0.746	0.142	0.167	0.124	0.746	1406
14	contrast	0.8	0.8	0.8	0.444	0.5	0.4	0.8	25
15	<none>	0.816	0.816	0.816	0.03	0.033	0.027	0.816	6378
16	discharge-disposition	0.83	0.83	0.83	0.151	0.167	0.138	0.83	513
17	addendum	0.833	0.833	0.833	0.151	0.167	0.139	0.833	3106
18	last-dose-of-antibiotics	0.872	0.872	0.872	0.466	0.5	0.436	0.872	397
19	indication	0.88	0.88	0.88	0.468	0.5	0.44	0.88	117
20	physical-examination	0.881	0.881	0.881	0.156	0.167	0.147	0.881	22113
21	image-type	0.884	0.884	0.884	0.313	0.333	0.295	0.884	181
22	discharge-condition	0.904	0.904	0.904	0.317	0.333	0.301	0.904	1490
23	infusions	0.909	0.909	0.909	0.476	0.5	0.455	0.909	99
24	history-of-present-illness	0.924	0.924	0.924	0.137	0.143	0.132	0.924	24950
25	discharge-medications	0.925	0.925	0.925	0.192	0.2	0.185	0.925	25088
26	flowsheet-data-vitals	0.932	0.932	0.932	0.482	0.5	0.466	0.932	2128
27	24-hour-events	0.954	0.954	0.954	0.244	0.25	0.238	0.954	1765
28	past-medical-history	0.959	0.959	0.959	0.163	0.167	0.16	0.959	5990
29	discharge-diagnosis	0.959	0.959	0.959	0.196	0.2	0.192	0.959	3578
30	family-history	0.968	0.968	0.968	0.328	0.333	0.323	0.968	1171
31	chief-complaint	0.968	0.968	0.968	0.492	0.5	0.484	0.968	1142
32	medical-condition	0.971	0.971	0.971	0.328	0.333	0.324	0.971	409
33	review-of-systems	0.977	0.977	0.977	0.494	0.5	0.488	0.977	724
34	labs-imaging	0.981	0.981	0.981	0.142	0.143	0.14	0.981	45855
35	discharge-instructions	0.986	0.986	0.986	0.166	0.167	0.164	0.986	23208
36	social-history	0.988	0.988	0.988	0.249	0.25	0.247	0.988	3114
37	allergies	0.989	0.989	0.989	0.331	0.333	0.33	0.989	891
38	assessment-and-plan	0.99	0.99	0.99	0.199	0.2	0.198	0.99	12728
39	reason	0.992	0.992	0.992	0.332	0.333	0.331	0.992	646
40	conclusions	0.994	0.994	0.994	0.498	0.5	0.497	0.994	2814
41	findings	0.998	0.998	0.998	0.333	0.333	0.333	0.998	6053
42	hospital-course	0.998	0.998	0.998	0.2	0.2	0.2	0.998	78321
43	social-and-family-history	1	1	1	1	1	1	1	52
44	technique	1	1	1	1	1	1	1	22
45	clinical-implications	1	1	1	1	1	1	1	36
46	other-medications	1	1	1	1	1	1	1	489
47	major-surgical-or-invasive-procedure	1	1	1	1	1	1	1	1903
48	facility	1	1	1	1	1	1	1	344
49	patient-test-information	1	1	1	1	1	1	1	1349
50	medication-history	1	1	1	0.333	0.333	0.333	1	6082

Table 7: By label BiLSTM-CRF$_{tok}$ performance where mF1 is the micro F1, mP is the micro precision, mR is the micro recall, MF1 is the macro F1, MP is the macro precision, MR is the macro recall, Acc is the accuracy, count is the the number of tokens encountered in the test set. The <none> label is for tokens with no section annotated.

SI as mentioned in Section 3.2. These pipelines include discharge note summarization, and other downstream tasks that would benefit from having header and non-section text removed such as training word embeddings such as ClinicalBERT.

Hyperparameter tuning with the baseline models is a next logical step for further work. Another obvious opportunity to improve performance is to concatenate *cui2vec* embeddings in the input layer as described in Section 3.3. Other future work includes comparing the results using the synthetic tokens in place of pseudo tokens, which would shed light on how models learn with more realistic data.

Acknowledgments

This work was supported by award R01 CA225446 from the National Institutes of Health. We thank Andy Boyd for his support on this work.

References

Emily Alsentzer and Anne Kim. 2018. Extractive Summarization of EHR Discharge Notes. arXiv: 1810.12085 (Only available as arXiv preprint).

Emily Alsentzer, John Murphy, William Boag, Wei-Hung Weng, Di Jindi, Tristan Naumann, and Matthew McDermott. 2019. Publicly Available Clinical BERT Embeddings. pages 72–78.

Pinkesh Badjatiya, Litton J. Kurisinkel, Manish Gupta, and Vasudeva Varma. 2018. Attention-Based Neural Text Segmentation. In *Advances in Information Retrieval*, Lecture Notes in Computer Science, pages 180–193. Springer International Publishing.

Stephen L. Barnes, Bryce R. H. Robinson, J. Taliesin Richards, Cindy E. Zimmerman, Tim A. Pritts, Betty J. Tsuei, Karyn L. Butler, Peter C. Muskat, Kenneth Davis, and Jay A. Johannigman. 2008. The devil is in the details: Maximizing revenue for daily trauma care. *Surgery*, 144(4):670–676.

Joe Barrow, Rajiv Jain, Vlad Morariu, Varun Manjunatha, Douglas Oard, and Philip Resnik. 2020. A Joint Model for Document Segmentation and Segment Labeling. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 313–322. Association for Computational Linguistics.

Andrew L. Beam, Benjamin Kompa, Allen Schmaltz, Inbar Fried, Griffin Weber, Nathan Palmer, Xu Shi, Tianxi Cai, and Isaac S. Kohane. 2020. Clinical Concept Embeddings Learned from Massive Sources of Multimodal Medical Data. *Pacific Symposium on Biocomputing. Pacific Symposium on Biocomputing*, 25:295–306.

Y. Bengio, P. Simard, and P. Frasconi. 1994. Learning long-term dependencies with gradient descent is difficult. *IEEE Transactions on Neural Networks*, 5(2):157–166.

Adam L. Berger, Vincent J. Della Pietra, and Stephen A. Della Pietra. 1996. A maximum entropy approach to natural language processing. *Computational Linguistics*, 22(1):39–71.

Olivier Bodenreider. 2004. The Unified Medical Language System (UMLS): Integrating biomedical terminology. *Nucleic Acids Research*, 32:D267–D270.

Piotr Bojanowski, Edouard Grave, Armand Joulin, and Tomas Mikolov. 2017. Enriching Word Vectors with Subword Information. *Transactions of the Association for Computational Linguistics*, 5:135–146.

Berry de Bruijn, Colin Cherry, Svetlana Kiritchenko, Joel Martin, and Xiaodan Zhu. 2010. NRC at i2b2: One challenge, three practical tasks, nine statistical systems, hundreds of clinical records, millions of useful features. In *Proceedings of the 2010 I2b2/VA Workshop on Challenges in Natural Language Processing for Clinical Data*, page 5.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186.

Benjamin D. Gallagher, Saman Nematollahi, Henry Park, and Salila Kurra. 2020. Comparing Students' Clinical Grades to Scores on a Standardized Patient Note-Writing Task. *Journal of General Internal Medicine*, 35(11):3243–3247.

A. Graves and J. Schmidhuber. 2005. Framewise phoneme classification with bidirectional LSTM networks. In *Proceedings. 2005 IEEE International Joint Conference on Neural Networks, 2005.*, volume 4, pages 2047–2052 vol. 4. IEEE.

Kenji Hirohata, Naoaki Okazaki, Sophia Ananiadou, and Mitsuru Ishizuka. 2008. Identifying Sections in Scientific Abstracts using Conditional Random Fields. In *Proceedings of the Third International Joint Conference on Natural Language Processing: Volume-I*.

Alistair E. W. Johnson, Tom J. Pollard, Lu Shen, Li-wei H. Lehman, Mengling Feng, Mohammad Ghassemi, Benjamin Moody, Peter Szolovits, Leo Anthony Celi, and Roger G. Mark. 2016. MIMIC-III, a freely accessible critical care database. *Scientific Data*, 3(1):1–9.

Jan-Christoph Klie, Michael Bugert, Beto Boullosa, Richard Eckart de Castilho, and Iryna Gurevych. 2018. The INCEpTION Platform: Machine-Assisted and Knowledge-Oriented Interactive Annotation. In *Proceedings of the 27th International Conference on Computational Linguistics: System Demonstrations*, pages 5–9. Association for Computational Linguistics.

Zeljko Kraljevic, Thomas Searle, Anthony Shek, Lukasz Roguski, Kawsar Noor, Daniel Bean, Aurelie Mascio, Leilei Zhu, Amos A. Folarin, Angus Roberts, Rebecca Bendayan, Mark P. Richardson, Robert Stewart, Anoop D. Shah, Wai Keong Wong, Zina Ibrahim, James T. Teo, and Richard J. B. Dobson. 2021. Multi-domain clinical natural language processing with MedCAT: The Medical Concept Annotation Toolkit. *Artificial Intelligence in Medicine*, 117:102083.

Klaus Krippendorff. 2004. Reliability in Content Analysis. *Human Communication Research*, 30(3):411–433.

Klaus Krippendorff. 2011. Agreement and Information in the Reliability of Coding. *Communication Methods and Measures*, 5(2):93–112.

John D. Lafferty, Andrew McCallum, and Fernando C. N. Pereira. 2001. Conditional Random Fields: Probabilistic Models for Segmenting and Labeling

Sequence Data. In *Proceedings of the Eighteenth International Conference on Machine Learning*, ICML '01, pages 282–289. Morgan Kaufmann Publishers Inc.

Jinhyuk Lee, Wonjin Yoon, Sungdong Kim, Donghyeon Kim, Sunkyu Kim, Chan Ho So, and Jaewoo Kang. 2020. BioBERT: A pre-trained biomedical language representation model for biomedical text mining. *Bioinformatics*, 36(4):1234–1240.

Larry McKnight and Padmini Srinivasan. 2003. Categorization of Sentence Types in Medical Abstracts. *AMIA Annual Symposium Proceedings*, 2003:440–444.

Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013a. Efficient estimation of word representations in vector space. arXiv: 1301.3781 (Only available as arXiv preprint).

Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. 2013b. Distributed Representations of Words and Phrases and their Compositionality. In *Advances in Neural Information Processing Systems*, volume 26. Curran Associates, Inc.

Namrata Nair, Sankaran Narayanan, Pradeep Achan, and K. P. Soman. 2022. Clinical Note Section Identification Using Transfer Learning. In Xin-She Yang, Simon Sherratt, Nilanjan Dey, and Amit Joshi, editors, *Proceedings of Sixth International Congress on Information and Communication Technology*, volume 235 of *Lecture Notes in Networks and Systems*, pages 533–542. Springer Singapore.

Jeffrey Pennington, Richard Socher, and Christopher Manning. 2014. Glove: Global Vectors for Word Representation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1532–1543. Association for Computational Linguistics.

Alexandra Pomares-Quimbaya, Markus Kreuzthaler, and Stefan Schulz. 2019. Current approaches to identify sections within clinical narratives from electronic health records: A systematic review. *BMC Medical Research Methodology*, 19(1):155.

Lance Ramshaw and Mitch Marcus. 1995. Text Chunking using Transformation-Based Learning. In *Third Workshop on Very Large Corpora*. ACL.

Najmeh Sadoughi, Greg P. Finley, Erik Edwards, Amanda Robinson, Maxim Korenevsky, Michael Brenndoerfer, Nico Axtmann, Mark Miller, and David Suendermann-Oeft. 2018. Detecting Section Boundaries in Medical Dictations: Toward Real-Time Conversion of Medical Dictations to Clinical Reports. In *Speech and Computer*, Lecture Notes in Computer Science, pages 563–573. Springer International Publishing.

Chaitanya Shivade, Pranav Malewadkar, Eric Fosler-Lussier, and Albert M. Lai. 2015. Comparison of UMLS terminologies to identify risk of heart disease using clinical notes. *Journal of Biomedical Informatics*, 58 Suppl:S103–S110.

Karen Sparck Jones. 1972. A statistical interpretation of term specificity and its application in retrieval. *Journal of documentation*, 28(1):11–21.

Michael Tepper, Daniel Capurro, Fei Xia, Lucy Vanderwende, and Meliha Yetisgen-Yildiz. 2012. Statistical Section Segmentation in Free-Text Clinical Records. In *Proceedings of the Eighth International Conference on Language Resources and Evaluation (LREC'12)*, pages 2001–2008. European Language Resources Association (ELRA).

Özlem Uzuner, Brett R South, Shuying Shen, and Scott L DuVall. 2011. 2010 i2b2/VA challenge on concepts, assertions, and relations in clinical text. *Journal of the American Medical Informatics Association : JAMIA*, 18(5):552–556.

Marliyya Zayyan. 2011. Objective Structured Clinical Examination: The Assessment of Choice. *Oman Medical Journal*, 26(4):219–222.

A Descriptions

Table 8: Note Categories

Name	Description
Consult	Notes generated when a specialst intervenes in a patient's care.
Discharge summary	A discharge summary describes a patient's stay at a hospital and the care they received. They can also include follow up instructions, medications and a schedule for future appointments.
Echo	An ultrasound of the heart.
Physician	Daily notes taken by the physician on their rounds as a part of a patient check up.
Radiology	Diagnosis and other notes taken by a radiologiest based on images such as xrays, MRI, CAT scans.

Table 9: Section Types

Section Type	Name	Description
24-hour-events	24 Hour Events	Description of what happened in the past 24 hours of the patients stay.
addendum	Addendum	An addition to the note.
allergies	Allergies	Patient allergies to medication and food of varying severity.
assessment-and-plan	Assessment And Plan	An overview of the problems that are occuring and the plan to address each problem.
critical-care-attending-addendum	Attending Addendum	The attending physician's addition to the note.
chief-complaint	Chief Complaint	The reason why the patient came to the hospital.
clinical-implications	Clinical Implications	Why this study is important.
code-status	Code Status	What should be done in the event of a cardiac or respiratory arrest, end of goals care.
communication	Communication	Information about who to contact and the relation to the patient.
comparison	Comparison	Comparing the new study to prior studies to determine interval changes.
conclusions	Conclusions	Interpretation of the findings in relation to the patient's condition.
contrast	Contrast	Was contrast introduced into the patient.
current-medications	Current Medications	Medications that the patient are taking at home.
discharge-condition	Discharge Condition	The stability of the patient upon discharge.
discharge-diagnosis	Discharge Diagnosis	The diagnosis of the patient after being worked up in the hospital.
discharge-disposition	Discharge Disposition	Where the patient is being discharged to.
discharge-instructions	Discharge Instructions	Post discharge instructions regarding what the patient can and cannot do.
discharge-medications	Discharge Medications	Medications that the patient will sent home with and to continue taking.
disposition	Disposition	Where the patient will go within the hospital.
family-history	Family History	Medical history of family members.
findings	Findings	Specific finidngs during the study.
flowsheet-data-vitals	Flowsheet Data/Vitals	Information pulled from flowsheets that are discretely kept within the ehr.
history	History	Patient's clinical history warranting exam.
history-of-present-illness	History Of Present Illness	A description of the events surrounding the reason why the patient came to the hospital: Symptom onset, duration, severity and associatating factors.
hospital-course	Hospital Course	A summary of what happened during the patient's time in the hospital.
image-type	Image Type	The type of study being performed.
imaging	Imaging	All image related orders placed by the physician including: CT, XRAY, ECHO, MRI, Ultrasound.
impression	Impression	Overall summerization of the study.
indication	Indication	Why the study was performed.
infusions	Infusions	Medications classified as a constant infusion.
labs	Labs	Laboratory values.
labs-imaging	Labs / Imaging	Lab and radiological results.
last-dose-of-antibiotics	Last Dose Of Antibiotics	Time of the last dose of antibiotic medications.
major-surgical-or-invasive-procedure	Major Surgical Or [...]	Any procedures or surgies that occured while the patient was at the hospital.
medical-condition	Medical Condition	History of the patient and why the patient needs the study.

Continued on the next page

Table 9: Section Types (cont)

Section Type	Name	Description
medication-history	Medication History	Medications that the patient are taking at home.
other-medications	Other Medications	Other medications the patient is receiving.
past-medical-history	Past Medical History	Medical problems a patient has.
past-surgical-history	Past Surgical History	All surgeries the patient has had in their past.
patient-test-information	Patient/Test Information	Basic and standardized information of the patient.
physical-examination	Physical Examination	Evalutating anatomic finds of a patient through palpation and auscultation.
prenatal-screens	Prenatal Screens	Screening of blood type and infections prior to delivery.
procedure	Procedure	Procedure name.
reason	Reason	Why the consulting team was brought in for the patient's care.
review-of-systems	Review of Systems	A generalized review of potential symptoms that the patient might not have addressed in the chief complaint or history of present illness.
social-history	Social History	History of occupation, recreational activities, and living situation.
social-and-family-history	Social and Family History	Combination of social and family history.
technique	Technique	How the procedure was being performed.
wet-read	Wet Read	Initial read, not the official read of the study.
addendum	addendum	An addition to the note.
facility	facility	The location the patient is going after discharge.

B Hyperparameters

The hyperparameters used to train the models described in Section 4. Those hyperparameters which differed for each model are given in Table 10. Hyperparameters shared across all models are given in Table 11. The only non-zero drop out was used in the LSTM layer.

Model	Epochs	Learning Rate	CRF
BERT-CRF$_{sent}$	40	0.003	True
BERT-CRF$_{sent}$ BioBERT	45	0.003	True
BERT$_{sent}$	35	0.003	False
BERT$_{sent}$ BioBERT	45	0.003	False
BiLSTM-CRF$_{tok}$ (GloVE 300D)	30	0.01	True
BiLSTM-CRF$_{tok}$ (GloVE 50D)	25	0.01	True
BiLSTM-CRF$_{tok}$ (word2vec)	30	0.01	True
BiLSTM-CRF$_{tok}$ fastText	40	0.01	True

Table 10: The hyperparameters of the models given in the results. Epochs is the number of epochs used to train the model, Learning Rate is the learning rate for the update step size of the loss function and CRF is whether the BiLSTM used a CRF output layer.

Name	Value	Description
Batch Size	20	The size of the mini-batches used to train the model.
Hidden Size	250	The hidden size of the LSTM.
Num Layers	2	The number of stacked layers of the LSTM.
Dropout	0.15	The dropout of the LSTM.

Table 11: The shared hyperparameters set for all models.

A Data-driven Approach to Named Entity Recognition for Early Modern French

Pedro Ortiz Suarez
Data and Web Science Group,
University of Mannheim
B 6, 26
68159 Mannheim, Germany
pedro.ortiz@uni-mannheim.de

Simon Gabay
Université de Genève
Rue du Général-Dufour 24,
1211 Genève, Switzerland
simon.gabay@unige.ch

Abstract

Named entity recognition has become an increasingly useful tool for digital humanities research, specially when it comes to historical texts. However, historical texts pose a wide range of challenges to both named entity recognition and natural language processing in general that are still difficult to address even with modern neural methods. In this article we focus in named entity recognition for historical French, and in particular for Early Modern French (16th-18th c.), i.e. *Ancien Régime* French. However, instead of developing a specialised architecture to tackle the particularities of this state of language, we opt for a data-driven approach by developing a new corpus with fine-grained entity annotation, covering three centuries of literature corresponding to the early modern period; we try to annotate as much data as possible producing a corpus that is many times bigger than the most popular NER evaluation corpora for both Contemporary English and French. We then fine-tune existing state-of-the-art architectures for Early Modern and Contemporary French, obtaining results that are on par with those of the current state-of-the-art NER systems for Contemporary English. Both the corpus and the fine-tuned models are released.

1 Introduction

Named entity recognition (NER) is an extensively studied task in natural language processing (NLP) that consists in identifying and classifying *named entities* mentions in unstructured text. These *named entities* often are real-world objects such as a *person*, a *location*, an *organisation* name or even a *product*. NER has been an important task in natural language processing for some time now. It was the focus of the MUC conferences and associated shared tasks (Marsh and Perzanowski, 1998), and later that of the CoNLL 2003 and the ACE shared tasks (Tjong Kim Sang and De Meulder, 2003; Doddington et al., 2004).

NER has quickly established itself as a pillar of the new methods of reading texts promoted by the digital humanities (DH), based on the analysis of large sets of literary or historical data via computational methods (Moretti, 2005). These sources being not only contemporary, the need for tools dealing with medieval or early modern states of language is now increasing. NER interests researchers in DH for numerous reasons since the application can be quite broad, from genealogy or history studies for which finding mentions of persons and places in texts is very useful; to applications in digital literature where researchers can use NER to highlight the path of different characters in a book or in a series of publications. Both the research in NER and DH can benefit from one another as it has already been suggested particular properties of literature can help to build better NER systems (Brooke et al., 2016) and even study how much diachronic variation influences NER systems (Ehrmann et al., 2016).

For the present study, we will focus on developing both an annotated corpus as well as a NER system for *Early Modern French*. We loosely define *Early Modern French* as a state of language following Middle French in 1500—adopting here the *terminus ad quem* used by the *Dictionnaire de Moyen Français* (Martin, 2020)—and ending with the French Revolution in 1789. In consequence, it encompasses three centuries (16th, 17th and 18th c.), or two linguistic periods: the *français préclassique* or "preclassical French" (1500–1630) and the *français classique* or "classical French" (1630–1689); both periodisations which are currently used in French linguistics (e.g. by Vachon 2010 and Amatuzzi et al. 2019). Early Modern French poses some particular challenges for NER systems, and mainly two. First, the spelling was not fixed and place names could be written differently from one text to another, but also in the same text. In Early Modern French, the name of the city of *Lyon* could

Proceedings of the 29th International Conference on Computational Linguistics, pages 3722–3730
October 12–17, 2022.

be written *Lyon*, but also *Lion*, creating in this case a homograph that has today disappeared (the *lion* being, like in English, an animal). Second, cities have changed their names, states have appeared, empires have disappeared, etc. and it is therefore impossible to use tools available for Contemporary French.

In this paper we develop a system that tries to tackle these specific challenges posed by Early Modern French, however, instead of developing a specialised architecture for this, we opt for a data-driven approach in which we try to annotate as much text as possible of an heterogeneous corpus covering several centuries and a vast range of genres and styles. We produce a fine-grained NER annotated corpus for Early Modern French that is many times bigger than some of the most popular NER annotated corpora for Contemporary English and French (Tjong Kim Sang and De Meulder, 2003; Sagot et al., 2012). We then fine-tune existing state-of-the-art architectures D'AlemBERT (Gabay et al., 2022) and CamemBERT (Martin et al., 2020) for Early Modern and Contemporary French respectively obtaining results that surpass the current state of the art NER systems for Contemporary French (Ortiz Suárez et al., 2020a), and that are on par with NER systems for Contemporary English (Straková et al., 2019; Yamada et al., 2020; Wang et al., 2021). We release both the corpus and the fine-tuned model in order to insure reproducibility of our experiments.[1]

2 Related work

If many evaluation campaigns for the recognition of named entities have been carried out since the end of the nineties [2], most of the corpora produced have until recently dealt with contemporary documents, particularly taken in the press (articles, dispatches...). In recent years, however, research has begun to focus on "historical" documents, but the diachronic depth of the language remains imperfectly treated, with a very clear concentration on the most recent textual sources: the 19th c. and 20th c. are by far over-represented (Ehrmann et al., 2021).

If the older states of language, linguistically more complex because of the instability of their spelling, remain left aside, we do note some attempts to extract entities from texts written before the 19th c. Previous research concerns 17th c. English (OCRised versions of the *Journals of the House of Lords*, cf. Grover et al. 2008), medieval latin (charters, cf. Torres Aguilar et al. 2016), German and French (legal documents written between the 14th and the 18th c., cf. Gwerder 2017). With the emergence of data-driven approaches, new corpora keep emerging for niche languages such as Middle High German and Old Norse (Besnier and Mattingly, 2021).

French is a typical case regarding NER, with resources and solutions focusing on documents written after the French Revolution. One of the oldest dataset is the one produced during the ESTER-2 evaluation campaign (Galliano et al., 2009), dealing with of radio broadcast transcripts. For the older documents, we have the *Quaero* (Rosset et al., 2012), *Europeana* (Neudecker, 2016) and *Impresso* (Ehrmann et al., 2020) corpora, going back the 19th c., but again with an almost unique focus on the press. Non-journalistic and/or non-recent French, however, seem to have attracted researchers in recent years. We have already mentioned the study of Gwerder (2017), whose data has unfortunately not been manually annotated and is therefore far from being optimal, and is limited to place and person names. If older rule-based approach keep being used (for place names, cf. Kogkitsidou and Gambette 2020), only one project has produced a manually annotated corpus, but limited to toponyms and using normalised versions (i.e. aligned with Contemporary French) of 17th c. plays (Gabay and Vitali, 2019).

An ambitious manually annotated corpus for pre-Revolutionary non-normalised French is still needed to give the means to researchers in history, literature or linguistics to offer new interpretation, relying on quantitative approaches such as "distant reading" (Moretti, 2013). If possible, this would corpus would need cover several centuries, and to offer more entities than just place and person names, such as quantities or events.

3 Corpus

Rather than designing a new corpus, we have decided to use a subpart of the "core corpus" of the *Presto* project (Blumenthal et al., 2017), namely the text written during the French *Ancien Régime*

[1] URL retained for anonymity.

[2] We spare the reader this story, which has already been perfectly told elsewhere cf. Ehrmann (2008); Nouvel et al. (2015).

Person		Function			
pers.ind	pers.coll	func.ind	func.coll		
Location			**Production**		
loc.adm.town	loc.phys.geo	loc.fac	prod.art	prod.rule	prod.object
loc.adm.reg	loc.phys.hydro	loc.oro			
loc.adm.nat					
loc.adm.sup					
Organization		**Time**	**Event**	**Quantity**	
org.adm	org.ent	time.date.abs	event	amount	
		time.date.rel			

Table 1: Types (in gray) and subtypes retained from the *Quaero* typology.
f

(c.15th-18th c., i.e. 34 texts)[3]. This choice is driven by our will to limit the number of annotated corpora for historical French, the same set of documents having already been abundantly corrected to train a lemmatizer (Gabay et al., 2020), but also to avoid a complex selection of works supposed to ensure a relative representativeness of literary documents from the *Ancien Régime*, already perfectly done by our colleagues.

The number of genres covered is extremely large: poetry, drama, novel, correspondence, grammar, philosophy, short stories, encyclopedic literature, etc. and guarantees, here again, a reasonable representativeness of the range of possibilities of *Belles-Lettres*[4]. The corpus is balanced regarding the distribution per century (c. 10/century) but not regarding the length of the texts, which increases over time (cf. fig. 1), following a possible trend in literature.

3.1 Annotation

It seemed logical to follow the *Quaero* annotation guide (Rosset et al., 2011), that is used by two important historical corpora presented *supra* (*Quaero* and *Impresso*). Because our texts and interests diverge from those of the aforementioned corpora, only some types and subtypes have been kept (cf. tab. 1) from the *Quaero* typology. The details of our choices can be found in a dedicated annotation manual (Gabay et al., 2022).

The annotated texts are available in multi-columns `tsv` files (cf. tab. 1). Each token has a lemma (manually corrected) and a POS (produced by the *Presto* project, non-systematically corrected but fairly reliable) using the MULTEXT tag set.

[3] A text has been withdrawn: the *Histoire d'un voyage faict en la terre du Brésil* by Jean de Léry, the transcription being too faulty to be able to correctly annotate the document.

[4] We do not offer a detailed description of the genres covered, these overlapping easily: poetry can be theological, political correspondence…

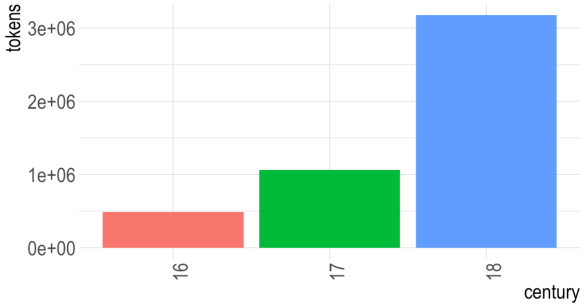

Figure 1: Number of tokens per century.

We propose a coarse-grained annotation for high-level entity types and fine-grained annotation using subtypes using the following syntax:

`BIO-TYPE.SUBTYPE`
For instance: `B-loc.adm.town`

Subtypes are sometimes simple (`B-org.town`) sometimes double (`B-loc.phys.geo`), depending of the complexity of the entity to annotate. Nested entities (i.e. an entity in an entity, such as a place name in a person name in *Henri d'**Angleterre***, "Henry of England") follow exactly the same syntax, and components a similar one, using six transverse elements:

- `name` to annotate tokens that are names (*Louis*, *Philippe*…)
- `title` to annotate tokens that are titles (*sieur*, *duc*, *abbé*…)
- `qualifier` to annotate tokens that are adjectives (*l'Inde **orientale***, *l'Arabie **heureuse***, *la mer **athlantique***, *l'**ancienne** Colchide*)… but also the generation (*Henri **IV***) or a cardinal position
- `kind` to annotate tokens that are hyperonyms (*l'**Empire** de Constantinople*, *la **mer** du Japon*

Token	Lemma	POS	COARSE	FINE	FINE-COMP	NESTED	Wikidata ID
Les	le	Da	O	O	O	O	_
allemands	allemand	Nc	O	O	O	O	_
élurent	élire	Vvc	O	O	O	O	_
pour	pour	S	O	O	O	O	_
empereur	empereur	Nc	B-pers	B-pers.ind	B-comp.title	O	Q438435
Rodolphe	Rodolphe	Np	I-pers	I-pers.ind	B-comp.name	O	Q438435
duc	duc	Nc	I-pers	I-pers.ind	B-comp.title	O	Q438435
de	de	S	I-pers	I-pers.ind	I-comp.title	O	Q438435
Suabe	Souabe	Np	I-pers	I-pers.ind	I-comp.title	B-loc.adm.reg	Q438435

Table 2: NERC Fine-Grained annotation with EL

- `unit` to annotate tokens that are units (meters, league, inches, pounds...)
- `val` to annotate tokens that are values (a number) that is linked to a unit to annotate an `amount`.

We have decided not to annotate metaphorical uses differently or in a separate column: everything is annotated in a literal sense. Thus, in ***France*** *goes to war*, *France* is labelled `loc.adm.nat` (i.e. the country) and not `org.adm` (i.e. the French government).

We have also started a first phase of semantic annotation, using Wikidata (Vrandečić and Krötzsch, 2014) identifiers, which remains imperfect. Due to the complexity of analysing certain entities, in particular personal names (e.g. *Pope John*), it was decided to annotate them only very marginally, only in the event of the absence of ambiguity (e.g. *Pope John V*). The annotation of place names, on the other hand, is more advanced and almost functional.

A first layer of annotation was made using regular expressions, before moving on to a manual correction phase. Given the size of the corpus, it is obvious that each token has not been checked, and that the final result does not claim to be perfect. Regular and thorough checks, however, concluded that the annotation was of the best possible quality and allow to move on to the training phase. All of the annotation work was carried out by a single person, in order to ensure the consistency of the data. The structure of the file and the form of the tags was controlled by a specific parser, designed specifically for this corpus.

3.2 A Note on Size

Our final annotated corpus has around 5 million annotated tokens, this makes it around 18 times bigger than the French treebank (Abeillé et al., 2003; Sagot et al., 2012; Ortiz Suárez et al., 2020a) and almost 23 times as big as the CoNLL 2003 (Tjong Kim Sang and De Meulder, 2003) corpus. Figures 1 and 2 show both the distribution of tokens by century and by *coarse* entity type. We can see that even though our corpus is far from balanced, even the 16th century portion of the corpus, which is our smallest, is still slightly larger than both the ConNLL 2003 and the FT corpora. We therefore believe that this annotated corpus gives us a great opportunity to study how state-of-the-art NER architectures behave when confronted with large amounts of annotated heterogeneous text.

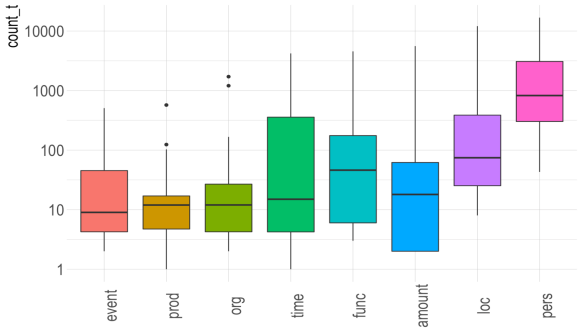

Figure 2: Number of entities ($\log_{10}$ scale) per category by text.

Given the size of our corpus, we opt for a 90-5-5 type split, that is, 90% of the text goes to the training set, 5% to the development set and 5% to the test. Otherwise the test and development sets would have been too big and training would have taken too long. The split is done at a document level and the sentences that go into the development and test sets are chosen at random, ensuring that both sets contain a representative portion of each of the documents in our corpus.

4 NER Evaluation

Having produced this annotated corpus, we now proceed with an evaluation using the *coarse* level

of annotation. We only use this level of annotation and not the other columns depicted on on table 2 as the training of some of as architectures turned out to be quite expensive due to the size of the corpus, with a single run of some of our models taking more than 24 hours on a machine equipped with an Nvidia V100 with 32 GB of memory. We also believe that the development of an architecture able to predict all levels of annotations at once merits a study of its own.

4.1 Models

We train three different models, a BiLSTM-CRF (Lample et al., 2016), CamemBERT (Martin et al., 2020) and D'AlemBERT (Gabay et al., 2022). All the training and fine-tuning is conducted using the `flair` framework[5] for sequence tagging (Akbik et al., 2019). To fine-tune D'AlemBERT and CamemBERT POS we follow the same approach as Schweter and Akbik (2020) with some modifications: we append a linear layer of size 256 that takes as input the last hidden representation of the `<s>` special token and the mean of the last hidden representation of the subword units of each token, that is, we use a *"mean"* subword pooling strategy. For the BiLSTM-CRF we use the implementation provided by the `flair` library, and we couple it with character embeddings as well as the Common Crawl-based FastText embeddings (Grave et al., 2018) originally trained by Facebook. Here is a small description of each of the models:

BiLSTM-CRF A classical neural architecture originally proposed by Lample et al. (2016) that combines a pre-trained fixed word embeddings with character embeddings, that are then feeded into a bidirectional LSTM (Hochreiter and Schmidhuber, 1997) encoder and a CRF (Lafferty et al., 2001) decoder. This model will serve as our baseline.

CamemBERT A Contemporary French language model originally proposed by Martin et al. (2020), is a Bidirectional Transformer-based model (Devlin et al., 2019; Vaswani et al., 2017) more precisely based on the RoBERTa (Liu et al., 2019) architecture, but using SentencePiece (Kudo and Richardson, 2018) instead of the original Byte-Pair Encoding (BPE) (Sennrich et al., 2016). CamemBERT uses a *base*-type architecture, which consists of 12 layers, 768 hidden dimensions, 12 attention heads, 110M parameters. CamemBERT was pre-trained using the French subcorpus of OSCAR 2019 (Ortiz Suárez et al., 2019; Ortiz Suárez et al., 2020b) which is extracted from Common Crawl snapshots, specifically from the plain text *WET* format distributed by Common Crawl which removes all the HTML tags and converts the text formatting to UTF-8. It follows the same approach as Grave et al. (2018) by using a language classification model based on the fastText linear classifier (Joulin et al., 2016; Joulin et al., 2017).

D'AlemBERT An Early Modern French language model originally pre-trained by Gabay et al. (2022) using a 1.2 GB corpus of Early Modern French called FREEM$_{max}$ (Gabay et al., 2022). D'AlemBERT uses the exact same *base*-type architecture as CamemBERT but for the tokenizer it uses the original BPE (Sennrich et al., 2016) of RoBERTa's (Liu et al., 2019) instead of SentencePiece (Kudo and Richardson, 2018). As opposed to CamemBERT or RoBERTa, D'AlemBERT was only trained for 31k steps.

4.2 Results and discussion

Model	Precision	Recall	F1-Score
BiLSTM-CRF	0.8640	0.8533	0.8586
CamemBERT	*0.9303*	*0.9309*	*0.9306*
D'AlemBERT	**0.9329**	**0.9323**	**0.9326**

Table 3: Comparison between D'AlemBERT, CamemBERT and an LSTM-CRF-based model performance on the test set of our corpus, results are averaged over 10 runs with different seeds.

Table 3 shows a brief overview of our results, we can see that our BiLSTM-CRF already produces quite strong results, attaining an f1-score of 0.8586 which is quite remarkable taking into account how heterogeneous our corpus is and how different the data itself is from the pre-training data used in the FastText word embeddings of the Bi-LSTM model.

On the other hand for both CamemBERT and D'AlemBERT we obtain quite high results above the 0.93 in f1-score. These results are quite remarkable because in spite of how heterogeneous our corpus is, and despite of the challenges posed by an historical language previously discussed, we obtain results that are almost on par with the current state of the art architectures for Contemporary English (Straková et al., 2019; Yamada et al., 2020; Wang et al., 2021).

[5] https://github.com/flairNLP/flair

CamemBERT					D'AlemBERT				
Entity Type	Precision	Recall	F1-Score	Support	Entity Type	Precision	Recall	F1-Score	Support
pers	0.9373	0.9236	0.9304	2734	pers	0.9355	0.9279	0.9317	2734
loc	0.9140	0.9371	0.9254	1384	loc	0.9242	0.9335	0.9288	1384
amount	0.9840	0.9840	0.9840	250	amount	0.9800	0.9800	0.9800	250
time	0.9447	0.9407	0.9427	236	time	0.9456	0.9576	0.9516	236
func	0.9209	0.9143	0.9176	140	func	0.9333	0.9000	0.9164	140
org	0.8364	0.9388	0.8846	49	org	0.8148	0.8980	0.8544	49
prod	0.7742	0.8889	0.8276	27	prod	0.8621	0.9259	0.8929	27
event	0.8333	0.8333	0.8333	12	event	0.8333	0.8333	0.8333	12
micro avg	0.9303	0.9309	0.9306	4832	micro avg	0.9329	0.9323	0.9326	4832
macro avg	0.8931	0.9201	0.9057	4832	macro avg	0.9036	0.9195	0.9111	4832
weighted avg	0.9307	0.9309	0.9307	4832	weighted avg	0.9331	0.9323	0.9327	4832
samples avg	0.8856	0.8856	0.8856	4832	samples avg	0.8893	0.8893	0.8893	4832

Table 4: Results of CamemBERT and D'AlemBERT on the test set of our corpus by entity type. Results are averaged over 10 runs with different seeds.

Strikingly, we do not see the same phenomenon as Gabay et al. (2022) who fine-tuned both CamemBERT and D'AlemBERT in POS tagging for Early Modern French, and that obtained remarkably good results with D'AlemBERT but subpar results with CamemBERT. We believe that this is due to the striking size of our corpus which has more than 5 million annotated tokens, that is, we believe that in this case CamemBERT has enough training data in order to properly fine-tune to this task in Early Modern French and in particular to potentially overcome the poor representations given by the SentencePiece (Kudo and Richardson, 2018) trained on Contemporary French for the out-of-vocabulary words found in the Early Modern French data. [6] We believe that to a certain extent, given the size of our corpus, CamemBERT might be "*forgetting*" its pre-training contemporary data and "*re-learning*" the Early Modern French data in our corpus. In any case, these high score proves the effectiveness of our data-driven approach as we didn't use any dedicated architecture for NER, yet we obtain state-of-the art results for a very challenging state of the French language.

In tables 5 and 4 we see the results of the BiLSTM-CRF, CamemBERT and D'AlemBERT models by entity type. All results are averaged over 10 runs using different seeds. For the BiLSTM-CRF model we see that in general it performs the best for the most common entity types and the worst for the least common types. It has particular trouble with the production category which might be due to the lack of these entities in the web-based pre-training corpus of the FastText fixed word embeddings. Strikingly, we see very good results for the amount entity type with our LSTM-based model, this is actually remarkable as this has historically been a rather difficult entity type to annotate for NER systems.

BiLSTM-CRF				
Entity Type	Precision	Recall	F1-Score	Support
pers	0.8808	0.8435	0.8617	2734
loc	0.8109	0.8707	0.8397	1384
amount	0.9040	0.9040	0.9040	250
time	0.9604	0.9237	0.9417	236
func	0.8872	0.8429	0.8645	140
org	0.8824	0.6122	0.7229	49
prod	0.9231	0.4444	0.6000	27
event	0.7273	0.6667	0.6957	12
micro avg	0.8640	0.8533	0.8586	4832
macro avg	0.8720	0.7635	0.8038	4832
weighted avg	0.8659	0.8533	0.8583	4832
samples avg	0.7737	0.7737	0.7737	4832

Table 5: Results of the BiLSTM-CRF model on the test set of our corpus by entity type. Results are averaged over 10 runs with different seeds.

For the CamemBERT and D'AlemBERT results by entity type, we see almost the exact same results for both models which actually supports our hypothesis that due to the size of our corpus, the Transformer-based models might be "*forgetting*" some of their pre-training contemporary data and "*re-learning*" the training data of our corpus seen during fine-tuning. There is a small exception to this and it again the *production* entity type, we can see that D'AlemBERT performs a bit better for this particular type which might be explained by the presence of these in D'AlemBERT's pre-training data as opposed to the lack of it in Camem-

[6] We observe that SentencePiece tends to split OOV words by characters which might not be ideal for sequence-tagging tasks, specially for NER.

BERT's web-based pre-training corpus, suggesting that while these models might be "*forgetting*" while exposed to corpora of the size of our corpus, they can still leverage their pre-training data to a certain extent.

5 Conclusion

In this paper we have produced a significantly big, fine-grained NER annotated corpus for Early Modern French, as well as state-of-the-art models for coarse NER annotation in Early Modern French. We showed that adopting a data-driven approach in which one focuses on producing as much annotated data as possible as opposed to producing highly specialised machine learning architectures for NER, is a quite successful approach as we have obtained results for Early Modern French that far surpass the current state of the art for Contemporary French and that are on par with the current state-of-the-art specialised architectures for Contemporary English. The corpus that we have produced also opens many future perspectives of research, for instance, we hope that in the future we will be able to study the impact of the size of the fine-tuning data in the fine-tuning of Transformer-based models, something that could be easily achieved by iteratively fine-tuning different Transformer-based with subsets of our corpus of incremental size. Furthermore, one could also use all the other levels of annotation of our corpus to develop a specialised architecture capable of predicting all annotation layers at once. In the end, we hope that both our corpus and our fine-tuned models will be useful to researchers in both Natural Language Processing and Digital Humanities.

References

Anne Abeillé, Lionel Clément, and François Toussenel. 2003. *Building a Treebank for French*, pages 165–187. Springer Netherlands, Dordrecht.

Alan Akbik, Tanja Bergmann, Duncan Blythe, Kashif Rasul, Stefan Schweter, and Roland Vollgraf. 2019. FLAIR: An easy-to-use framework for state-of-the-art NLP. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics (Demonstrations)*, pages 54–59, Minneapolis, Minnesota. Association for Computational Linguistics.

Antonella Amatuzzi, Carine Skupien Dekens, Wendy Ayres-Bennett, Annette Gerstenberg, and Lene Schoesler. 2019. Améliorer et appliquer les outils numériques. ressources et approches pour l'étude du changement linguistique en français préclassique et classique. In *Le français en Diachronie*, Travaux de Linguistique Romane, pages 337–364. Editions de linguistique et de philologie.

Clément Besnier and William Mattingly. 2021. Named-entity dataset for medieval latin, middle high german and old norse. *Journal of Open Humanities Data*, 7:23. Publisher: Ubiquity Press.

Peter Blumenthal, Sascha Diwersy, Achille Falaise, Marie-Hélène Lay, Gilles Souvay, and Denis Vigier. 2017. Presto, un corpus diachronique pour le français des XVIe-XXe siècles. In *Actes de la 24ème conférence sur le Traitement Automatique des Langues Naturelles - TALN'17*. Association pour le traitement automatique des langues.

Julian Brooke, Adam Hammond, and Timothy Baldwin. 2016. Bootstrapped text-level named entity recognition for literature. In *Proc. of the 54th Annual Meeting of the Association for Computational Linguistics*, pages 344–350, Berlin, Germany.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, Minneapolis, Minnesota. Association for Computational Linguistics.

George Doddington, Alexis Mitchell, Mark Przybocki, Lance Ramshaw, Stephanie Strassel, and Ralph Weischedel. 2004. The automatic content extraction (ACE) program – tasks, data, and evaluation. In *Proceedings of the Fourth International Conference on Language Resources and Evaluation (LREC'04)*, Lisbon, Portugal. European Language Resources Association (ELRA).

Maud Ehrmann. 2008. *Les Entitées Nommées, de la linguistique au TAL : Statut théorique et méthodes de désambiguïsation*. Theses, Paris Diderot University.

Maud Ehrmann, Giovanni Colavizza, Yannick Rochat, and Frédéric Kaplan. 2016. Diachronic evaluation of NER systems on old newspapers. *Proc. of the 13th Conference on Natural Language Processing (KONVENS 2016)*, pages 97–107.

Maud Ehrmann, Ahmed Hamdi, Elvys Linhares Pontes, Matteo Romanello, and Antoine Doucet. 2021. Named entity recognition and classification on historical documents: A survey. *arXiv:2109.11406 [cs]*.

Maud Ehrmann, Matteo Romanello, Alex Flückiger, and Simon Clematide. 2020. Overview of CLEF HIPE 2020: Named entity recognition and linking on historical newspapers. *Experimental IR meets multilinguality, multimodality, and interaction. 11th International Conference of the CLEF Association, CLEF 2020, Thessaloniki, Greece, September 22–25, 2020, Proceedings*.

Simon Gabay, Jean-Baptiste Camps, and Thibault Clérice. 2022. Manuel d'annotation linguistique pour le français moderne (XVIe -XVIIIe siècles). Version B (« Béate Béatrice »).

Simon Gabay, Thibault Clérice, Jean-Baptiste Camps, Jean-Baptiste Tanguy, and Matthias Gille-Levenson. 2020. Standardizing linguistic data: method and tools for annotating (pre-orthographic) french. In *Proceedings of the 2nd International Digital Tools & Uses Congress (DTUC '20)*.

Simon Gabay, Pedro Ortiz Suarez, Alexandre Bartz, Alix Chagué, Rachel Bawden, Philippe Gambette, and Benoît Sagot. 2022. From FreEM to D'AlemBERT: a Large Corpus and a Language Model for Early Modern French. *arXiv e-prints*, page arXiv:2202.09452.

Simon Gabay and Giovanni Pietro Vitali. 2019. A theatre of places: Mapping 17th c. french theatre. In *Proceedings of the 13th Workshop on Geographic Information Retrieval (GIR'18)*. ACM.

Sylvain Galliano, Guillaume Gravier, and Laura Chaubard. 2009. The ester 2 evaluation campaign for the rich transcription of french radio broadcasts. In *Interspeech 2009 - Proceedings of the 10th Annual Conference of the International Speech Communication Association*, pages 2583–2586.

Edouard Grave, Piotr Bojanowski, Prakhar Gupta, Armand Joulin, and Tomas Mikolov. 2018. Learning word vectors for 157 languages. In *Proceedings of the Eleventh International Conference on Language Resources and Evaluation (LREC 2018)*, Miyazaki, Japan. European Language Resources Association (ELRA).

Claire Grover, Sharon Givon, Richard Tobin, and Julian Ball. 2008. Named entity recognition for digitised historical texts. In *Proceedings of the Sixth International Conference on Language Resources and Evaluation (LREC'08)*. European Language Resources Association (ELRA).

Yvonne Gwerder. 2017. *Named Entity Recognition in Digitized Historical Texts*. University of Zurich.

Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long Short-Term Memory. *Neural Computation*, 9(8):1735–1780.

Armand Joulin, Edouard Grave, Piotr Bojanowski, Matthijs Douze, Hérve Jégou, and Tomas Mikolov. 2016. FastText.zip: Compressing text classification models. *arXiv e-prints*, page arXiv:1612.03651.

Armand Joulin, Edouard Grave, Piotr Bojanowski, and Tomas Mikolov. 2017. Bag of tricks for efficient text classification. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 2, Short Papers*, pages 427–431, Valencia, Spain. Association for Computational Linguistics.

Eleni Kogkitsidou and Philippe Gambette. 2020. Normalisation of 16th and 17th century texts in French and geographical named entity recognition. In *Proceedings of the 4th ACM SIGSPATIAL Workshop on Geospatial Humanities*, pages 28–34, Seattle, Washington, USA.

Taku Kudo and John Richardson. 2018. SentencePiece: A simple and language independent subword tokenizer and detokenizer for neural text processing. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 66–71, Brussels, Belgium. Association for Computational Linguistics.

John D. Lafferty, Andrew McCallum, and Fernando C. N. Pereira. 2001. Conditional random fields: Probabilistic models for segmenting and labeling sequence data. In *Proceedings of the Eighteenth International Conference on Machine Learning*, ICML '01, page 282–289, San Francisco, CA, USA. Morgan Kaufmann Publishers Inc.

Guillaume Lample, Miguel Ballesteros, Sandeep Subramanian, Kazuya Kawakami, and Chris Dyer. 2016. Neural architectures for named entity recognition. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 260–270, San Diego, California. Association for Computational Linguistics.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. RoBERTa: A Robustly Optimized BERT Pretraining Approach. *arXiv e-prints*, page arXiv:1907.11692.

Elaine Marsh and Dennis Perzanowski. 1998. MUC-7 evaluation of IE technology: Overview of results. In *Seventh Message Understanding Conference (MUC-7): Proceedings of a Conference Held in Fairfax, Virginia, April 29 - May 1, 1998*.

Louis Martin, Benjamin Muller, Pedro Javier Ortiz Suárez, Yoann Dupont, Laurent Romary, Éric de la Clergerie, Djamé Seddah, and Benoît Sagot. 2020. CamemBERT: a tasty French language model. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7203–7219, Online. Association for Computational Linguistics.

Robert Martin. 2020. *Dictionnaire du Moyen Français*. ATILF - CNRS & Université de Lorraine.

Franco Moretti. 2005. *Graphs, maps, trees : abstract models for a literary history*. Verso.

Franco Moretti. 2013. *Distant reading*. Verso.

Clemens Neudecker. 2016. An open corpus for named entity recognition in historic newspapers. In *Proceedings of the Tenth International Conference on Language Resources and Evaluation (LREC'16)*, pages 4348–4352. European Language Resources Association (ELRA).

Damien Nouvel, Maud Ehrmann, and Sophie Rosset. 2015. *Les entités nommées pour le traitement automatique des langues*. ISTE Group.

Pedro Javier Ortiz Suárez, Yoann Dupont, Benjamin Muller, Laurent Romary, and Benoît Sagot. 2020a. Establishing a new state-of-the-art for French named entity recognition. In *Proceedings of the 12th Language Resources and Evaluation Conference*, pages 4631–4638, Marseille, France. European Language Resources Association.

Pedro Javier Ortiz Suárez, Laurent Romary, and Benoît Sagot. 2020b. A monolingual approach to contextualized word embeddings for mid-resource languages. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 1703–1714, Online. Association for Computational Linguistics.

Pedro Javier Ortiz Suárez, Benoît Sagot, and Laurent Romary. 2019. Asynchronous pipelines for processing huge corpora on medium to low resource infrastructures. In *Proceedings of the Workshop on Challenges in the Management of Large Corpora (CMLC-7) 2019. Cardiff, 22nd July 2019*, pages 9 – 16, Mannheim. Leibniz-Institut für Deutsche Sprache.

Sophie Rosset, Cyril Grouin, Karën Fort, Olivier Galibert, Juliette Kahn, and Pierre Zweigenbaum. 2012. Structured named entities in two distinct press corpora: Contemporary broadcast news and old newspapers. In *Proceedings of the Sixth Linguistic Annotation Workshop*, pages 40–48. Association for Computational Linguistics.

Sophie Rosset, Cyril Grouin, and Pierre Zweigenbaum. 2011. *Entités nommées structurées : guide d'annotation Quaero*. LIMSI-CNRS.

Benoît Sagot, Marion Richard, and Rosa Stern. 2012. Annotation référentielle du corpus arboré de Paris 7 en entités nommées (referential named entity annotation of the Paris 7 French TreeBank) [in French]. In *Proceedings of the Joint Conference JEP-TALN-RECITAL 2012, volume 2: TALN*, pages 535–542, Grenoble, France. ATALA/AFCP.

Stefan Schweter and Alan Akbik. 2020. FLERT: Document-Level Features for Named Entity Recognition. *arXiv e-prints*, page arXiv:2011.06993.

Rico Sennrich, Barry Haddow, and Alexandra Birch. 2016. Neural machine translation of rare words with subword units. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1715–1725, Berlin, Germany. Association for Computational Linguistics.

Jana Straková, Milan Straka, and Jan Hajic. 2019. Neural architectures for nested NER through linearization. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 5326–5331, Florence, Italy. Association for Computational Linguistics.

Erik F. Tjong Kim Sang and Fien De Meulder. 2003. Introduction to the CoNLL-2003 shared task: Language-independent named entity recognition. In *Proceedings of the Seventh Conference on Natural Language Learning at HLT-NAACL 2003*, pages 142–147.

Sergio Torres Aguilar, Xavier Tannier, and Pierre Chastang. 2016. Named entity recognition applied on a data base of medieval latin charters. the case of chartae burgundiae. In *3rd International Workshop on Computational History (HistoInformatics 2016)*.

Claire Hélène Vachon. 2010. *Le Changement linguistique au XVIe siècle: une étude basée sur des textes littéraires français*. ELiPhi, Éditions de linguistique et de philologie.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Ł ukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc.

Denny Vrandečić and Markus Krötzsch. 2014. Wikidata: a free collaborative knowledgebase. *Communications of the ACM*, 57(10):78–85.

Xinyu Wang, Yong Jiang, Nguyen Bach, Tao Wang, Zhongqiang Huang, Fei Huang, and Kewei Tu. 2021. Improving named entity recognition by external context retrieving and cooperative learning. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 1800–1812, Online. Association for Computational Linguistics.

Ikuya Yamada, Akari Asai, Hiroyuki Shindo, Hideaki Takeda, and Yuji Matsumoto. 2020. LUKE: Deep contextualized entity representations with entity-aware self-attention. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 6442–6454, Online. Association for Computational Linguistics.

Reproducibility and automation of the APPRAISAL taxonomy

Pradeesh Parameswaran, Andrew Trotman, Veronica Liesaputra, David Eyers
Department of Computer Science
University of Otago
New Zealand
[pradeesh,andrew,veronica,dme]@cs.otago.ac.nz

Abstract

There is a lack of reproducibility in the results of experiments that apply the APPRAISAL taxonomy. APPRAISAL is widely used by linguists to study how people judge things or people. Automating APPRAISAL could be beneficial for use cases such as moderating online comments. Past work on APPRAISAL annotation has been descriptive in nature, and the lack of publicly available data sets hinders the progress of automation. In this work, we are interested in two things; first, we are interested in how well humans can reproduce the annotation of APPRAISAL of the Australasian Language Technology Association (ALTA) data set. We employed four annotators, each with a similar cultural and linguistics background to reannotate the data set. Second, we are interested in measuring the performance of the existing automated approaches to APPRAISAL classification. Our results show a poor level of agreement at more detailed APPRAISAL categories (Fleiss κ = 0.059) and a fair level of agreement (κ = 0.372) at coarse-level categories. We find similar results when using automated approaches that are publicly available. Our empirical evidence suggests that, at present, automating APPRAISAL classification is practical only when considering coarse-level categories of the taxonomy.

1 Introduction

With the rising popularity of social media platforms, such as Twitter and Facebook, we are experiencing an unprecedented surge of unstructured textual discourse ready to be analysed (Gundecha and Liu, 2012). Supervised learning in Natural Language Processing (NLP) has helped us extract the richness of the information found in these texts for purposes such as sentiment analysis (Zhang et al., 2018), hate-speech detection (Schmidt and Wiegand, 2019), and question answering (Shah et al., 2019). The training task within supervised learning requires high-quality annotated data in order to perform well (Ramas et al., 2021).

Supervised learning can apply theories of evaluative language (Bateman et al., 2019). With high-quality annotated data, we are confident that the task of identifying phrases of evaluative language can be automated. Evaluative language allows us to analyse how we express our feelings, our assessments of people, situations and objects (Benamara et al., 2017). As evaluative language is such a large and intricate discipline, herein we restrict ourselves to just the APPRAISAL[1](Martin and White, 2003) taxonomy within it.

APPRAISAL gives linguists a systematic approach to evaluating language such as identifying and understanding how people make judgements about things (people and objects). The taxonomy has been widely used by linguists to analyse the language choices and attitudes used by writers in order to express their stances (Chen, 2022) in various media such as in news biographies, examiners' reports and tweets (Starfield et al., 2015; Ross and Caldwell, 2020; Su and Hunston, 2019).

Looking forward, APPRAISAL could be used to automate the moderation of online comments (Cavasso and Taboada, 2021); in spite of automation, human moderators continue to be required for the final judgement call on some comments (Ghosh et al., 2011).

The task of reading a comment (and analysing the language) is known to have a negative psychological impact on moderator's mental health (Steiger et al., 2021). Sullivan (2022) argued that the impact on human moderators could be lowered by reducing the number of comments they read. For example, if a comment is identified as having legal implications (based on APPRAISAL analysis),

[1]Small caps are used to distinguish technical, linguistic terms from their use in common parlance.

Proceedings of the 29th International Conference on Computational Linguistics, pages 3731–3740
October 12–17, 2022.

the comment could be automatically rejected, or flagged for legal review.

To date, much of the research in APPRAISAL annotation has been descriptive in nature (Fuoli, 2018; Fuoli and Hommerberg, 2015). Fuoli (2018) proposed a structured approach known as the *stepwise approach*, however it stops short of providing guidelines to future researchers and has not been quantified. If there are to be robust data sets for studying APPRAISAL then it is necessary to quantify the quality of the data sets already available, examine the practices involved in acquiring them, and strive to improve the techniques in a well grounded, measurable way.

We focus the scope of this work on just the JUDGEMENT subbranch of the APPRAISAL taxonomy because there is publicly available data provided by the Australasian Language Technology Association (ALTA) for their 2020 Shared Task Challenge (Mollá, 2020). Figure 1 shows the part of the APPRAISAL taxonomy that focuses on JUDGEMENT and where it fits in the hierarchy. In this paper, we investigate the following research questions: 1) *the reproducibility and the reliability of humans annotating* JUDGEMENT *sentences*, and 2) *the effectiveness of automated approaches to classify* JUDGEMENT.

In order to answer our research questions, we first employed four annotators, each with a linguistics background, to re-annotate the ALTA data set. Our experiments demonstrate significant levels of disagreement between the annotators at Level 4 of the APPRAISAL hierarchical taxonomy (Fleiss κ = 0.089) as opposed to the more consistent results at Level 2 of the taxonomy (κ = 0.558). We then compare the performance of three different systems submitted to the ALTA challenge. We observe a similar, relative effect: A κ score of 0.031 at Level 4 and 0.206 at Level 2.

Our qualitative analysis of the assessments shows that categorising the exact type of JUDGEMENT can be difficult, as what constitutes morality (a part of JUDGEMENT) is subjective—foresight and background context are required.

We aim to encourage further research into the application of the APPRAISAL taxonomy that is reproducible. This would collectively support our goal to build robust automated approaches to aid SFL practitioners in handling large datasets. To aid in future research, we have made our annotations and experimental data publicly available.[2]

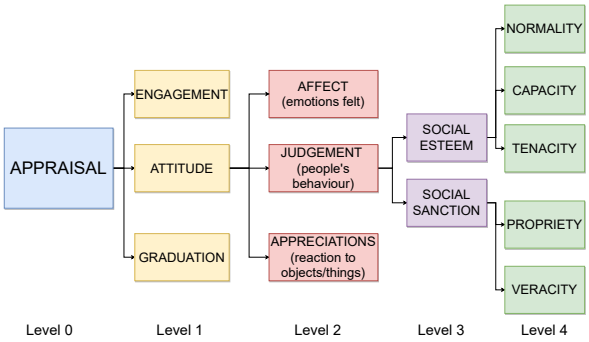

Figure 1: JUDGEMENT branch of the APPRAISAL hierarchical taxonomy (in context, and adapted from (Stewart, 2015, p. 3).)

2 Related Work

The APPRAISAL taxonomy was originally developed by Martin and White (2003) and it is used by linguists to study discourse across a diverse range of genres (Chen, 2022). APPRAISAL captures the evaluative meaning (*opinion*) of the person who wrote a piece of text toward another person or an object. This taxonomy is widely used for many tasks including analysing biographies (Su and Hunston, 2019), Donald Trump's tweets (Ross and Caldwell, 2020), investigating online reviews of South Park (Paronen, 2011), advertising messaging (Beangstrom and Adendorff, 2013) and PhD examiners' reports (Starfield et al., 2015).

There are five levels in the APPRAISAL taxonomy (Figure 1). The levels indicate the granularity of the categories. At Level 1, there are three categories: ATTITUDE (*emotions, ethics and aesthetics*), GRADUATION (*how* ATTITUDE *is being used in a sentence*) and ENGAGEMENT (*writer's openness for negotiation*). The ATTITUDE branch can be broken down further into AFFECT (*emotions*), JUDGEMENT (*ethics*) and APPRECIATION (*aesthetics*). The branch of JUDGEMENT is further divided into SOCIAL ESTEEM and SOCIAL SANCTIONS. SOCIAL ESTEEM deals with admiration and criticism of people, without any legal implication. This branch is further subdivided into NORMALITY (*how closely one follows the norm of the society*), CAPACITY (*how capable the person is*) and TENACITY (*how dependable the person is*). SOCIAL SANCTIONS on the other hand deals with the behaviour of a person that has a legal or

[2] https://github.com/prasys/appraisal-annotation-coling2022

Text	Classification
I am beyond mad that I lost track of a brown spider in my brown carpet. Where did you go?	CAPACITY
Feels like I lost my best friend #lost #fml #missingyou	NORMALITY
@alour @jkramon1313 you should force your neighbours to pay that! Those people have some nerve!!! #Victim	PROPRIETY
I feel like a burden every day that I waste but I don't know how to get out of this	CAPACITY,VERACITY
Instagram seriously sort your sh*t out. I spent ages writing that caption for you to delete it and not post it!! #fume #instagram	*None*
I am about to be a coward and I feel terrible. But I can't even face this	VERACITY

Table 1: Example Tweets from the ALTA data set (Mollá, 2020) and their classification under the JUDGEMENT branch of the APPRAISAL taxonomy.

moral implication. This branch of the taxonomy is further divided into VERACITY (*how truthful one is*) and PROPRIETY (*how ethical the person is*). Some examples of tweets and where they fit within JUDGEMENT are given in Table 1.

There is a substantial amount of work on building automated approaches to applying JUDGEMENT classification to text (Argamon et al., 2007; Bloom and Argamon, 2010; Whitelaw et al., 2005; Neviarouskaya et al., 2010; Taboada et al., 2011). Most of this work focuses on using a combination of machine learning approaches and a hand-built lexicon. For instance, Argamon et al. (2007), constructed lexicons from the seed words of Martin and White (2003) and apply Naive Bayes and Support Vector Machine (SVM) classifiers to the task. They obtain an F_1 score of 0.345. One of the major drawbacks of these past approaches is that the APPRAISAL terms in the lexicons were selected based on researchers' intuition, and it is unclear what these intuitions were. Worse, the lexicons do not appear to be publicly available, thereby making it impossible to reproduce the experiments.

The lack of publicly available APPRAISAL data resulted in a stall in research, but recently the Australasian Language Technology Association (ALTA) organised a shared task and encouraged participants to build an automated system to identify the subclasses of JUDGEMENT used in tweets (Mollá, 2020). They made their data publicly available.[3] The results of the shared task were underwhelming, with the best system obtaining an F_1 score of 0.155 (Aroyehun and Gelbukh, 2020). In reaction, we sought to determine whether humans found this task equally difficult.

Prior work in this area by Read and Carroll (2012), used two annotators to annotate a book corpus and obtained an F_1 score of 0.434 at Level 4 of the APPRAISAL taxonomy, as opposed to an F_1 score of 0.532 at Level 1 of the APPRAISAL taxonomy. Their data do not appear to be publicly available.

Ross and Caldwell (2020) discovered that tweets contain a higher proportion of JUDGEMENT words than AFFECT and APPRECIATION words. JUDGEMENT in tweets is especially insightful to an organisation interested in brand reputation, as an increasing number of consumers are using Twitter to recommend brands to their friends (Vidya et al., 2015). Knowing how a product is being judged can bring insights to an organisation on how that product might be improved. Collectively, these studies highlight the importance of evaluating the reliability of humans can perform the classification and compare the performance with existing approaches.

3 Data set and Annotation

3.1 Data set

We consider the data set from the ALTA 2020 Shared Task (Mollá, 2020). The data set was originally sourced from the SemEval 2018 AIT DISC data set (Mohammad et al., 2018). The ALTA data contains 300 tweets that have been annotated by two linguists, one from the University of Wollongong and the other from the University of New South Wales, and then verified by two other linguists from the same two universities. The data is then split into 200 tweets for training and the remaining 100 tweets for the testing portion. Each tweet was annotated with one or more categories of Level 4 of the APPRAISAL taxonomy. Some examples are given in Table 1.

[3] https://www.kaggle.com/c/alta-2020-challenge/

3.2 Annotation

In order to measure the reproducibility of the annotation process we re-annotated the test portion of the ALTA data set.

By using the test set it was possible to use the training set as guidelines for our annotators, and also to compare the performance of runs submitted to the shared task on multiple sets of annotations.

Annotation for JUDGEMENT is non-trivial and requires an understanding of linguistics. Because of this complexity we employed four human annotators each with a background in linguistics.[4] All annotators are native English speakers and New Zealanders. The detailed background of the annotators is as follows:

- a—an associate professor with 20 years of teaching experience in a languages department at a university in New Zealand,
- b—a graduate student in linguistics currently studying for a Master's in Linguistics at a university in New Zealand,
- c—a final year undergraduate student working towards a degree in applied linguistics at a university in New Zealand, and
- d—a professional language translator and interpreter in New Zealand with over 10 years of experience.

In addition to these four, we have the reverse engineered golden data set denoted (g).[5]

All annotators were aware of the Systemic Functional Linguistic (SFL) theory, which forms the basis of the APPRAISAL taxonomy. However, we also provided the set of guidelines from Martin and White (2003) and the 200 labelled tweets from the training set. Following the recommendations of Fuoli (2018), all annotators worked independently and were of similar cultural and ideological backgrounds. The annotators classified each tweet into one or more of the five categories of Level 4 of the APPRAISAL taxonomy in Figure 1, or marked the tweet as *None* if the tweet was not JUDGEMENT-bearing. Some tweets contained sensitive content, so we provided a *Skip* option, but it was not used. Each annotator was given two hours to complete the task but all finished in under an hour. We did not ask the annotators to identify JUDGEMENT-bearing terms and leave this for future work.

[4]We obtained ethical approval from University of Otago Ethics Committee (Approval No: D20/334).

[5]We contacted the organisers of ALTA Task in order to obtain the test set annotations, however the organisers were not able to release the data. We then reverse engineered the annotations by submitting controlled runs to the competition's automated scoring platform—something that was done with the full knowledge of the task organisers.

Level	Without g	With g
0–2	0.558	0.372
3	0.211	0.182
4	0.089	0.059

Table 2: Fleiss κ score between our annotators with and without g at different levels of the APPRAISAL taxonomy.

κ	Agreement
<0	Less than chance
0.01–0.20	Slight
0.21–0.40	Fair
0.41–0.60	Moderate
0.61–0.80	Substantial
0.81–0.99	Almost perfect

Table 3: Interpretation of κ scores (adapted from Landis and Koch (1977)).

4 Human Agreement Level

Annotating JUDGEMENT is difficult as sentences can contain varying degrees of objectivity and subjectivity. How these subjective sentences are finally classified can depend on the background of the annotator (Wiebe et al., 2005; Read and Carroll, 2012). To reduce subjectivity, we follow the recommendations of Read and Carroll (2012) by annotating and reporting the scores at a tweet level (sentence level). We calculate the agreement levels based on the Fleiss κ score and mean F_1 score.

Fleiss κ was chosen as it is appropriate when the given task is subjective (Waseem, 2016; Alm, 2011), but recent work by Delgado and Tibau (2019) recommends avoiding just κ when comparing the performance of automated approaches due to the *kappa paradox* (Feinstein and Cicchetti, 1990; Cicchetti and Feinstein, 1990). The *kappa paradox* arises because the κ statistic accounts for agreement by chance—which is low for machine learning approaches that closely follow the patterns in the training data. It is, thus, important to consider F_1 when comparing the performance

Level	(a,b)	(a,c)	(a,d)	(a,g)	(b,c)	(b,d)	(b,g)	(c,d)	(c,g)	(d,g)
0–2	0.877	0.875	0.870	0.535	0.963	0.958	0.580	0.985	0.559	0.555
3	0.648	0.672	0.628	0.391	0.737	0.690	0.475	0.686	0.476	0.370
4	0.402	0.421	0.380	0.221	0.401	0.397	0.196	0.317	0.245	0.132

Table 4: Mean F_1 score between annotators at various levels of the APPRAISAL taxonomy.

of automated approaches. F_1 has also been used in previous work (Mollá, 2020; Read and Carroll, 2012; Argamon et al., 2007).

Table 2 shows the Fleiss κ score of our annotators when we include and do not include the golden data set[6]. Note that Levels 0, 1 and 2 of the APPRAISAL taxonomy are collapsed to Levels 0–2 because we have only annotated JUDGEMENT and below. To interpret the values of κ, we follow the guidelines by Landis and Koch (1977) that have been widely adopted in this research area and are reproduced in Table 3.

In line with the findings of Read and Carroll (2012), our annotators' agreement scores drop as the APPRAISAL classification moves from Level 0 towards Level 4. We explore some of the factors behind this in Section 7.

At Level 4, if we were to include the golden set, g, as an annotator, there is a substantial level of disagreement among the annotators. Similarly, at Level 3, (SOCIAL ESTEEM versus SOCIAL SANCTIONS), we see that our annotators still have a low agreement. However, at Level 0–2, we see moderate agreement between the annotators when the golden data set is not included, and a fair agreement level when the golden data set is included.

Table 4 shows the results when using the mean F_1 score. There we see the scores across annotators are very high at Level 0–2 (with the exception of comparing our annotators with g). As with the κ scores, we see that the scores go down as we traverse through the APPRAISAL taxonomy.

The difference between F_1 and κ scores demonstrates the importance of reporting both scores. Reporting F_1 alone gives the impression that this task is easy (has a high level of agreement) but the F_1 score does not take into account classifications that could have occurred by chance. The κ scores suggest that the task is difficult (has a low level of agreement) at Level 4 of the taxonomy. This sheds light as to why the automated approaches that were built by the participants of the ALTA task performed poorly (Mollá, 2020), as we find even humans disagree at Levels 3 and 4.

[6]We built the collective agreement (intersection) set but found the scores dropped further.

5 Classifiers' Agreement Level

In this section, we discuss our experiments for measuring the effectiveness of automated approaches. We consider all three systems that were used to submit runs to the ALTA 2020 Shared Task because the source code to these is publicly available, and we were able to gain access to the submissions. We did not compare to earlier systems as neither they, nor their word lists, nor the data they trained on are publicly available.

The systems we used are as follows:

- NLP-CIC (Aroyehun and Gelbukh, 2020): An ensemble of logistic regression and ROBERTA classifier.
- OrangutanV2 (Parameswaran et al., 2020): An ensemble of two ALBERT classifiers.
- NITS (Khilji et al., 2020): An ensemble of XGBoost and decision tree classifiers that use pre-trained BERT embeddings.

Mean F_1 scores of each tweet against g are reported in Table 5. The performance of these systems against g is lower than the performance of our annotators against g (e.g., (a,g) in Table 4).

This is hardly surprising, since the systems were trained on a limited amount of data, but the annotators were able to draw from years of experience. Nonetheless, NLP-CIC and OrangutanV2 are able to distinguish JUDGEMENT from non-JUDGEMENT fairly accurately at Level 0–2. NITS may be failing due to the use of XGBoost which is susceptible to outliers, and certainly produces results that differ from the transformer-based models of NLP-CIC and OrangutanV2.

Level	NLP-CIC	OrangutanV2	NITS
0–2	0.605	0.558	0.384
3	0.407	0.389	0.258
4	0.157	0.155	0.132

Table 5: Mean F_1 scores of the automated approaches at each level of the APPRAISAL taxonomy.

6 Performance of Humans and Classifiers

We then proceed to compare the performance of human annotators with the automated approaches. We are interested in knowing at which level of the APPRAISAL taxonomy is there a significant difference between the performance of a human and of a machine.

We picked the best performing human annotator who obtained the highest F_1 score against the golden data set at every level to be evaluated (c).

We determined that scores of c for each tweet are not normally distributed by running the Shapiro–Wilk test (Royston, 1992) and obtaining a p-value of 0.0466 (not normally distributed at the significance level of $p < 0.05$).

Knowing that the data are not normally distributed we chose the Wilcoxon signed-rank test (Woolson, 2007) to test for significant differences between c and each automated system. We ran the test after removing ties, which minimises type one error (McGee, 2018).

Level	NLP-CIC	OrangutanV2	NITS
0–2	0.198	0.221	0.314
3	0.102	0.113	0.214
4	0.038*	0.042*	0.095

Table 6: p-values of Wilcoxon signed-rank test. Asterisks (*) denote statistical significance at ($p < 0.05$).

We present our results in Table 6 which shows the p-values from each test. Our analysis shows that there is a statistically significant difference between c and two of the three systems (NLP-CIC and OrangutanV2) at Level 4 of the APPRAISAL taxonomy. However, we find that there are no statistically significant differences at Levels 0 through to 3. Our evidence suggests that at levels closer to 0, the performance of machines and humans are comparable. However, we cannot confidently say so, as our effect size ($d = 0.15$) is very small. We report our Fleiss κ scores in Table 7. The agreement levels are low, a similar finding to that of Delgado and Tibau (2019).

Level	Without g	With g
0–2	0.120	0.206
3	0.094	0.105
4	0.007	0.031

Table 7: Fleiss κ score between automated approaches and with g at various levels of the APPRAISAL taxonomy.

This may be because these models are based on BERT, and thus they share similar characteristics and produce similar results.

7 Qualitative Analysis

Observing the high level of disagreement among annotators and g (mean κ at Level 4 being 0.199), we examined each disagreement in turn. They all fall into two categories:

- Category 1—our assessors do not agree with each other, and g chooses *None*,
- Category 2—our assessors agree with each other, but not with g. This is only seen at Level 4 of the taxonomy.

Consider the following Category 1 example:

> "*Absolutely love @unqualified but can't listen to it during my commute on the subway because I burst out laughing and people stare! .*"

Annotators a, c and d marked this NORMALITY while a and b marked it CAPACITY (a marked it twice). The golden assessor, g, marked it *None*. When we apply the APPRAISAL taxonomy manually, we observed that the first part of the sentence *"absolutely love @unqualified but can't listen to it during my commute on the subway"* being AFFECT, however in the second part *"because I burst out laughing and people stare"* is JUDGEMENT.

We believe this sentence does contain JUDGEMENT. In many societies, laughing loudly on a train is considered rude, as is listening to music loudly. We agree with the majority of our annotators that this warrants classification as NORMALITY, and if we follow the methodology of Fuoli

(2018), this tweet falls into both categories. All of the automated approaches also predict NORMALITY. Our analysis shows three instances of Category 1. Although, from our visual inspection, these sentences contain JUDGEMENT, we are not sure why g chose *None* (and we have no way to investigate this).

As an example of Category 2:

> "*So disappointed in myself for spending £50 on an outfit for meeting a boy #noselfcontrol #nervous .*"

Annotator g tagged this NORMALITY, disagreeing with all the other annotators who marked it CAPACITY. There are two ways that the sentence can be interpreted, for our annotators, they view the person as not being *capable* of controlling their impulses to purchase an outfit for their date. As for g, the annotators plausibly view it as NORMALITY as it was normal for people to be in their best outfit and behaviour when they are out on a date, thus it was *normal* to spend money on a dress. We believe that both of these categories are correct, especially as there are no well-defined criteria to distinguish between CAPACITY and NORMALITY. To address this issue, we suggest clarifying annotation guidelines given to the assessors, and the criteria used to distinguish the different JUDGEMENT categories. All of the automated approaches predicted PROPRIETY. This is likely to be due to the lack of data—there is only one example in the training data set that has a sentence similar to the sentence above: "*Incredibly shocked and disappointed with @united customer service. Really making me rethink flying with them in the future. #unhappy*". That sentence is marked as PROPRIETY. Our findings reflect the findings of Tayyar Madabushi et al. (2019), who have demonstrated that BERT fails to generalise properly when training and test data are significantly dissimilar even though these data sets are very similar in nature. There are four instances of category 2 disagreement.

Another explanation for discrepancies between our annotators and g is that annotating tweets is tricky because the tweeter and the annotator are subject to different cultural and personal views of what JUDGEMENT is. The data we use consists of tweets from different individuals and we have different individuals assessing them, thus we can expect high levels of disagreement. By comparison, the work of Ross and Caldwell (2020), applies APPRAISAL theory to tweets from one individual (Donald Trump), and so the assessors are able to better understand the message behind the tweet. This could be the reason why much of the previous work on APPRAISAL has focused solely on a particular topic or person and not a generalised situation. One way to address discrepancy among annotators is for the annotators to meet and discuss differences—and to improve the assessment guidelines by writing clear-cut criteria for distinguishing difficult cases. We hope to see more diversified data sets released in the future so that we can validate the generalisability of automated JUDGEMENT approaches.

8 Recommendations & Limitations

Our analyses show that: 1) Annotating Level 4 of JUDGEMENT categories is challenging as there is ambiguity in the interpretation of the text; and 2) The evaluation and the reasoning presented in the APPRAISAL literature are very rarely complete. The latter can be addressed by making the data sets publicly available and also sharing the assumptions and annotation guidelines. It is paramount to have these guidelines as it helps reproduce research results.

Moreover, sharing these rules would make it easier to automate approaches that classify APPRAISAL-bearing sentences. By addressing these gaps, we believe that annotating Level 4 of APPRAISAL would be clearer, although how the rules differ or are similar from one data set to another is yet to be seen.

Our work has limitations. Our data set is small and focuses solely on Twitter and the JUDGEMENT branch of APPRAISAL. We find that performing discourse analysis can be challenging especially when it is related to judging the morality of a person from a single tweet. This can surface different perspectives based on different assumptions based on a tiny piece of text (Lachmar et al., 2017).

We evaluated our automated approaches using systems built for the ALTA Shared Task. All of these models were based on BERT and we believe that these models can be further improved by fine-tuning, which has been shown to improve performance elsewhere (Xin et al., 2021). Another plausible cause of the poor performance of these models is that the data set that was trained on and evaluated on was small, and thus was likely to contain bias.

As a direction for future work, it would be inter-

esting to revisit some of the techniques from earlier work mentioned in Section 2. Of course, the lexicons would need to be reproduced as closely as possible, but we hypothesise that the deep learning models could be further improved by using such techniques.

9 Conclusions

In this study, we investigated two topics: (1) reproducibility and the reliability of a popular SFL taxonomy, APPRAISAL, focusing on JUDGEMENT annotation, and (2) the effectiveness of automated approaches to assessing JUDGEMENT. To carry out our investigation in a systematic manner, we employed four linguists to carefully re-annotate the publicly available ALTA 2020 Shared Task data set and used three publicly available, automated approaches. We then performed experiments quantifying and evaluating the performance of our annotators and automated approaches.

We find a low level of agreement when annotating JUDGEMENT despite using annotators with linguistics backgrounds. We obtained a Fleiss κ score of 0.059 when using Level 4 (most detailed categories) within the APPRAISAL's hierarchical taxonomy as opposed to 0.372 when using Level 2 (coarse-grained categories).

We find a similar pattern with the automated approaches. We obtain a Fleiss κ score of 0.031 at Level 4 and 0.206 at Level 2 of the taxonomy. Although the low κ score in automated approaches is attributed to the nature of κ statistical penalising agreement not occurring by chance, our F_1 score (0.605 at Level 2 and 0.155 at Level 4) supports our earlier findings that humans find classifying JUDGEMENT to be difficult. Furthermore, we find that there is no statistical significance between the performance of our best performing annotators and of the best performing system when working with Level 2 category of the APPRAISAL taxonomy, thus we argue that automation of JUDGEMENT is possible at this level as automated systems are already performing at human levels.

Our analyses sheds light on the challenges in reproducibility of APPRAISAL annotation. We believe that the poor scores of human annotators and automated approaches are due to a multitude of factors including the lack of publicly available data sets (examples), the absence of details such as prior assumptions made by the annotators, and the lack of generally available clear and concise annotation guidelines.

We have publicly released our data and analysis to encourage more research into APPRAISAL. We believe that the application of APPRAISAL to tweets and other discourse will enable the application of APPRAISAL to other domains (such as eCommerce).

Acknowledgements

We wish to thank the New Zealand eScience Infrastructure (NeSI) for providing the infrastructure that we used to conduct our experiments.

References

Cecilia Ovesdotter Alm. 2011. Subjective natural language problems: Motivations, applications, characterizations, and implications. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, pages 107–112.

Shlomo Argamon, Kenneth Bloom, Andrea Esuli, and Fabrizio Sebastiani. 2007. Automatically determining attitude type and force for sentiment analysis. In *Language and Technology Conference*, pages 218–231. Springer.

Segun Taofeek Aroyehun and Alexander Gelbukh. 2020. Automatically predicting judgement dimensions of human behaviour. In *Proceedings of the The 18th Annual Workshop of the Australasian Language Technology Association*, pages 131–134.

John A Bateman, Daniel McDonald, Tuomo Hiippala, Daniel Couto-Vale, Eugeniu Costechi, et al. 2019. Systemic-functional linguistics and computation: new directions, new challenges. *The Cambridge Handbook of Systemic-Functional Linguistics*.

Tracy Beangstrom and Ralph Adendorff. 2013. An appraisal analysis of the language of real estate advertisements. *Southern African Linguistics and Applied Language Studies*, 31(3):325–347.

Farah Benamara, Maite Taboada, and Yannick Mathieu. 2017. Evaluative language beyond bags of words: Linguistic insights and computational applications. *Computational Linguistics*, 43(1):201–264.

Kenneth Bloom and Shlomo Argamon. 2010. Unsupervised extraction of appraisal expressions. In *Canadian Conference on Artificial Intelligence*, pages 290–294. Springer.

Luca Cavasso and Maite Taboada. 2021. A corpus analysis of online news comments using the appraisal framework. *Journal of Corpora and Discourse Studies*, 4:1–38.

Muxuan Chen. 2022. An appraisal analysis of Sina Weibo texts about reforms of undergraduate education. *Scientific and Social Research*, 4(1):1–16.

Domenic V Cicchetti and Alvan R Feinstein. 1990. High agreement but low kappa: II. resolving the paradoxes. *Journal of clinical epidemiology*, 43(6):551–558.

Rosario Delgado and Xavier-Andoni Tibau. 2019. Why Cohen's Kappa should be avoided as performance measure in classification. *PloS one*, 14(9):e0222916.

Alvan R Feinstein and Domenic V Cicchetti. 1990. High agreement but low kappa: I. the problems of two paradoxes. *Journal of clinical epidemiology*, 43(6):543–549.

Matteo Fuoli. 2018. A stepwise method for annotating appraisal. *Functions of Language*, 25(2):229–258.

Matteo Fuoli and Charlotte Hommerberg. 2015. Optimising transparency, reliability and replicability: Annotation principles and inter-coder agreement in the quantification of evaluative expressions. *Corpora*, 10(3):315–349.

Arpita Ghosh, Satyen Kale, and Preston McAfee. 2011. Who moderates the moderators? crowdsourcing abuse detection in user-generated content. In *Proceedings of the 12th ACM conference on Electronic commerce*, pages 167–176.

Pritam Gundecha and Huan Liu. 2012. Mining social media: a brief introduction. *New directions in informatics, optimization, logistics, and production*, pages 1–17.

Abdullah Faiz Ur Rahman Khilji, Rituparna Khaund, and Utkarsh Sinha. 2020. Human behavior assessment using ensemble models. In *Proceedings of the The 18th Annual Workshop of the Australasian Language Technology Association*, pages 140–144.

E Megan Lachmar, Andrea K Wittenborn, Katherine W Bogen, and Heather L McCauley. 2017. # mydepressionlookslike: Examining public discourse about depression on twitter. *JMIR Mental Health*, 4(4):e8141.

J Richard Landis and Gary G Koch. 1977. An application of hierarchical kappa-type statistics in the assessment of majority agreement among multiple observers. *Biometrics*, pages 363–374.

James R Martin and Peter R White. 2003. *The language of evaluation*, volume 2. Springer.

Monnie McGee. 2018. Case for omitting tied observations in the two-sample t-test and the Wilcoxon-Mann-Whitney Test. *PloS one*, 13(7):e0200837.

Saif Mohammad, Felipe Bravo-Marquez, Mohammad Salameh, and Svetlana Kiritchenko. 2018. Semeval-2018 task 1: Affect in tweets. In *Proceedings of the 12th international workshop on semantic evaluation*, pages 1–17.

Diego Mollá. 2020. Overview of the 2020 ALTA Shared Task: Assess human behaviour. *ALTA 2020*, page 127.

Alena Neviarouskaya, Helmut Prendinger, and Mitsuru Ishizuka. 2010. Recognition of affect, judgment, and appreciation in text. In *Proceedings of the 23rd international conference on computational linguistics (COLING 2010)*, pages 806–814.

Pradeesh Parameswaran, Andrew Trotman, Veronica Liesaputra, and David Eyers. 2020. Classifying judgements using transfer learning. In *Proceedings of the 18th Annual Workshop of the Australasian Language Technology Association*, pages 135–139.

Tiina Paronen. 2011. *Appraisal in Online Reviews of South Park: A study of engagement resources used in online reviews*. Ph.D. thesis, University of Jyväskylä.

Jose Garrido Ramas, Giorgio Pessot, Abdalghani Abujabal, and Martin Rajman. 2021. Identifying and resolving annotation changes for natural language understanding. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies: Industry Papers*, pages 10–18.

Jonathon Read and John Carroll. 2012. Annotating expressions of appraisal in English. *Language resources and evaluation*, 46(3):421–447.

Andrew S. Ross and David Caldwell. 2020. 'Going negative': An APPRAISAL analysis of the rhetoric of Donald Trump on Twitter. *Lang. Commun.*, 70:13–27.

Patrick Royston. 1992. Approximating the Shapiro-Wilk W-test for non-normality. *Statistics and computing*, 2(3):117–119.

Anna Schmidt and Michael Wiegand. 2019. A survey on hate speech detection using natural language processing. In *Proceedings of the Fifth International Workshop on Natural Language Processing for Social Media, April 3, 2017, Valencia, Spain*, pages 1–10. Association for Computational Linguistics.

Asad Ali Shah, Sri Devi Ravana, Suraya Hamid, and Maizatul Akmar Ismail. 2019. Accuracy evaluation of methods and techniques in web-based question answering systems: a survey. *Knowledge and Information Systems*, 58(3):611–650.

Sue Starfield, Brian Paltridge, Robert McMurtrie, Allyson Holbrook, Sid Bourke, Hedy Fairbairn, Margaret Kiley, and Terry Lovat. 2015. Understanding the language of evaluation in examiners' reports on doctoral theses. *Linguist. Educ.*, 31:130–144.

Miriah Steiger, Timir J Bharucha, Sukrit Venkatagiri, Martin J Riedl, and Matthew Lease. 2021. The psychological well-being of content moderators: the emotional labor of commercial moderation and avenues for improving support. In *Proceedings of the*

2021 CHI conference on human factors in computing systems, pages 1–14.

Martyn Stewart. 2015. The language of praise and criticism in a student evaluation survey. *Studies in educational evaluation*, 45:1–9.

Hang Su and Susan Hunston. 2019. Language patterns and attitude revisited: Adjective patterns, Attitude and Appraisal. *Functions of Language*, 26(3):343–371.

Mark Sullivan. 2022. Facebook is expanding its tools to make content moderation less toxic.

Maite Taboada, Julian Brooke, Milan Tofiloski, Kimberly Voll, and Manfred Stede. 2011. Lexicon-based methods for sentiment analysis. *Computational linguistics*, 37(2):267–307.

Harish Tayyar Madabushi, Elena Kochkina, and Michael Castelle. 2019. Cost-sensitive BERT for generalisable sentence classification on imbalanced data. In *Proceedings of the Second Workshop on Natural Language Processing for Internet Freedom: Censorship, Disinformation, and Propaganda*, pages 125–134, Hong Kong, China. Association for Computational Linguistics.

Nur Azizah Vidya, Mohamad Ivan Fanany, and Indra Budi. 2015. Twitter sentiment to analyze net brand reputation of mobile phone providers. *Procedia Computer Science*, 72:519–526.

Zeerak Waseem. 2016. Are you a racist or am i seeing things? annotator influence on hate speech detection on twitter. In *Proceedings of the first workshop on NLP and computational social science*, pages 138–142.

Casey Whitelaw, Navendu Garg, and Shlomo Argamon. 2005. Using appraisal groups for sentiment analysis. In *Proceedings of the 14th ACM international conference on Information and knowledge management*, pages 625–631.

Janyce Wiebe, Theresa Wilson, and Claire Cardie. 2005. Annotating expressions of opinions and emotions in language. *Language resources and evaluation*, 39(2):165–210.

Robert F Woolson. 2007. Wilcoxon signed-rank test. *Wiley encyclopedia of clinical trials*, pages 1–3.

Ji Xin, Raphael Tang, Yaoliang Yu, and Jimmy Lin. 2021. BERxiT: Early exiting for BERT with better fine-tuning and extension to regression. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*, pages 91–104, Online. Association for Computational Linguistics.

Lei Zhang, Shuai Wang, and Bing Liu. 2018. Deep learning for sentiment analysis: A survey. *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery*, 8(4):e1253.

Few-Shot Table Understanding: A Benchmark Dataset and Pre-Training Baseline

Ruixue Liu*, **Shaozu Yuan***, **Aijun Dai, Lei Shen**
Tiangang Zhu, Meng Chen, Xiaodong He
JD AI, Beijing, China
{liuruixue, yuanshaozu, daiaijun1, shenlei20}@jd.com
{zhutiangang3, chenmeng20, xiaodong.he}@jd.com

Abstract

Few-shot table understanding is a critical and challenging problem in real-world scenario as annotations over large amount of tables are usually costly. Pre-trained language models (PLMs), which have recently flourished on tabular data, have demonstrated their effectiveness for table understanding tasks. However, few-shot table understanding is rarely explored due to the deficiency of public table pre-training corpus and well-defined downstream benchmark tasks, especially in Chinese. In this paper, we establish a benchmark dataset, FewTUD, which consists of 5 different tasks with human annotations to systematically explore the few-shot table understanding in depth. Since there is no large number of public Chinese tables, we also collect a large-scale, multi-domain tabular corpus to facilitate future Chinese table pre-training, which includes one million tables and related natural language text with auxiliary supervised interaction signals. Finally, we present FewTPT, a novel table PLM with rich interactions over tabular data, and evaluate its performance comprehensively on the benchmark. Our dataset and model will be released to the public soon.

1 Introduction

Relational tables, as a typical form of structured data on the Web, store a vast amount of knowledge. Table understanding (Wang et al., 2012) aims to understand the semantics of tabular data as well as the associated text jointly, which further improves the evaluation results of several tasks, including table question answering (Khalid et al., 2007; Sun et al., 2016; Bogin et al., 2019), table retrieval (Zhang and Balog, 2018), and table fact verification (Chen et al., 2020; Zhang et al., 2020). Recently, inspired by the huge success of pre-trained language models (PLMs) in understanding *free-form* natural language (NL) sentences, researchers have attempted to model *structured* data using pre-training techniques. Various table pre-training models (Herzig et al., 2020; Yin et al., 2020; Yu et al., 2021; Liu et al., 2021; Cheng et al., 2021; Shi et al., 2022; Dong et al., 2022) have been proposed and made remarkable progress in learning the structured schema of tables and the alignment between the input text and the schema.

In real-world scenario, table understanding usually faces more challenging situations, in which tables are from different domains and each table contains very limited annotations. Despite its importance, few-shot (Lake et al., 2015) table understanding is rarely explored in previous works due to the following three obstacles. First, the deficiency of public well-designed benchmark datasets for few-shot table understanding makes the model evaluation inconvenient. Second, the lack of public large-scale high-quality table pre-training corpora blocks the exploration of table PLMs. Lastly, a table pre-training baseline tailored to the few-shot table understanding is also needed for better performance comparison. Especially, the lack of benchmark datasets and pre-training corpora in Chinese hinders the research on table understanding.

To fill the above gaps, in this paper, we focus on the dataset construction for table-understanding related tasks in Chinese. We first establish a few-shot table understanding benchmark dataset, **FewTUD**, with five table related tasks, by which research and exploration can be carried out extensively. Different from existing table understanding tasks, which concentrate on the information inside tables, our tasks, based on real-world scenarios, lay emphasis on the interaction between tables and the corresponding NL text (e.g., Table Fact Verification, Table QA, Table Selection, and Schema Detection), and focus on the whole content of tables (e.g., Table Classification). We collect tables with meta information and the corresponding NL text from the Web, then manually annotate the dataset for

*Both authors contributed equally to this work.

Proceedings of the 29th International Conference on Computational Linguistics, pages 3741–3752
October 12–17, 2022.

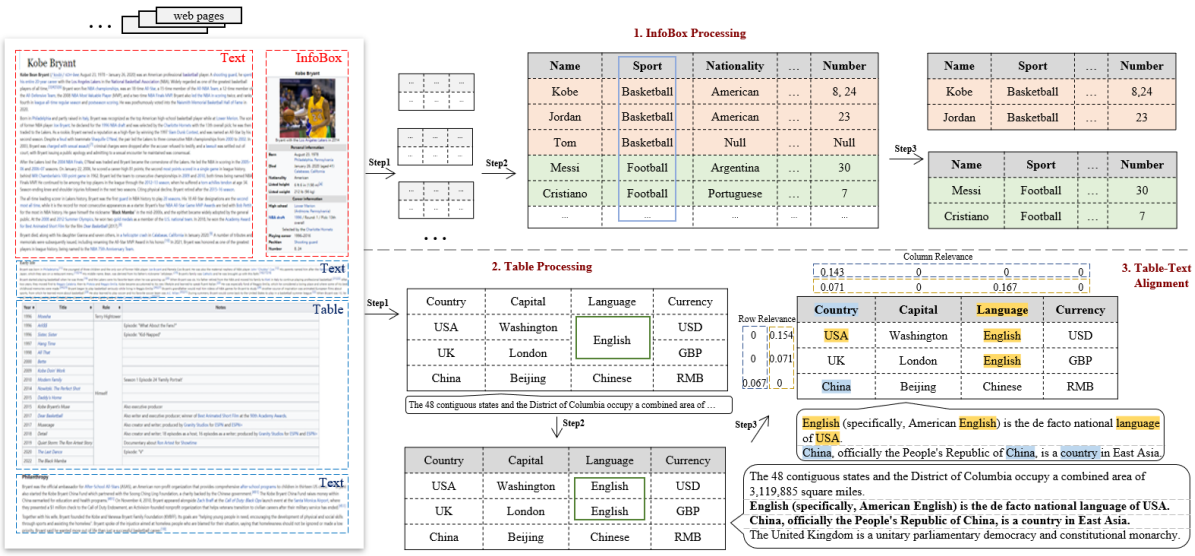

Figure 1: Data Construction of TPC-1M with three main procedures: (1) InfoBox processing, (2) table processing with column identification and cell value splitting, (3) table-text alignment and relevance score estimation.

each task, aiming to provide an evaluation criterion for few-shot table understanding.

Second, we collect a large-scale Chinese table pre-training corpus, **TPC-1M**, with more than 1 million tables and their associated NL text. How to obtain semantically relevant NL text is critical for constructing the table pre-training corpus. Previous works either obtain the associated NL text by synthesising pseudo NL text on available tables (Yu et al., 2021; Liu et al., 2021; Qin et al., 2021) or crawling surrounding NL text of tables simply based on position information (Yin et al., 2020; Herzig et al., 2020). Differently, we locate the associated NL text via semantic matching approach. We also provide the relevance score between the NL text and each row/column of the table as auxiliary supervised interaction signal, which may further facilitate the table pre-training.

Lastly, we propose a novel table pre-training model, **FewTPT**, to serve as the baseline for the few-shot table understanding benchmark. Different from previous table pre-training models (Yin et al., 2020; Herzig et al., 2020), we focus on catching the interactions between tables and NL text. Specifically, we devise two novel spatial-aware pre-training tasks, i.e., column relevance prediction (CRP) and row relevance prediction (RRP), which predict the relevant column from table schema and the relevant row from table content respectively based on the supervised interaction signals. We also employ variants of Masked Language Modeling (MLM) objective to predict the tokens in NL text, column names and cell values. Finally, we conduct extensive experiments to show the advantage of FewTPT against several strong baselines.

To summarise, our main contributions are threefold. (1) We establish a Chinese benchmark dataset for few-shot table understanding, which includes five downstream tasks: Table Fact Verification, Table Question Answering (Table QA), Table Selection, Schema Detection, and Table Classification. We hope the benchmark can be a testbed for future few-shot table understanding research in Chinese. (2) We contribute a large-scale high-quality table pre-training corpus in Chinese, which covers 1 million tables across 13 domains and their semantically-associated NL text. (3) We propose a novel table pre-training model to serve as the baseline for the benchmark, and experimental results show the competitiveness of our model.

2 Construction of TPC-1M and FewTUD

In this section, we introduce the construction of table pre-training corpus, TPC-1M, and benchmark dataset for few-shot table understanding, FewTUD.

2.1 TPC-1M Corpus

Corpus Collection. We first crawl huge amount of table pages from the web (e.g., Baidu Baike[1] and E-commerce website[2]). Web pages with the same topic field are grouped into one domain. For

[1]https://baike.baidu.com.
[2]https://www.jd.com.

	Max	Min	Mean	Median
Row	120	2	11.6	6
Column	80	2	6.5	6

Table 1: Data statistics of TPC-1M.

a Baike page, it contains a special kind of table InfoBox[3] that illustrates the properties of one entity (i.e., celebrity) and can be regarded as the single-row table. To enrich information density and table variety, we aggregate similar InfoBoxes into a multi-row table. Figure 1(1) demonstrates the process of InfoBox pre-processing. We first combine InfoBoxes with similar schemas into one big table. Then we split it into sub-tables according to the row values in a specific schema, which is determined by the lowest entropy with Maximum Entropy algorithm (Jaynes, 1982) (i.e., *Sport* in Figure 1(1)). We further filter some sparse columns and rows of these sub-tables to reduce complexity. For other tables in the web page, some are already corrupted, i.e., "<th>" (table header) is omitted, or some cells are merged. For the absence of table header tag, we train a binary classifier[4] to identify whether the first row or first column is the table header. For the merged cells, we split them into individual ones based on the position information. The above process is shown in Figure 1(2).

Finally, we need to pick out the associated NL text for each table. For previous web-crawled tables and context from English Wikipedia (Lehmberg et al., 2016), the context is mainly mined based on position information and may not be semantically related to tables. As Figure 1(3) shows, we find the NL text from the title, caption and text descriptions around a table. To locate the semantically-relevant NL text accurately, we calculate the linguistic overlap ratio with Jaccard Similarity algorithm (Niwattanakul et al., 2013) between the table and its candidate text snippets. The text snippets with the top-N similarity scores are chosen as the associated NL text. Furthermore, we assign a relevant score to each row/column based on the n-gram overlap between the row/column and the associated NL text. The relevant scores indicate how likely the row/column is mentioned in the associated NL text, which can be regarded as auxiliary supervised interaction signals between tables and the NL text.

[3]Similar to https://en.wikipedia.org/wiki/Help:Infobox.

[4]We construct the training set from well-formed tables and the classification accuracy is 95%.

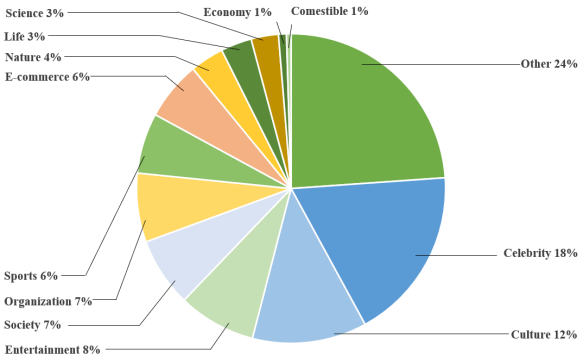

Figure 2: Domain distribution of TPC-1M.

Statistics Analysis. After pre-processing, we obtain 1,002k tables with 3,014k semantically associated NL sentences. Figure 2 shows the distribution of domains. TPC-1M covers 13 different domains, including culture, character, celebrity, sports, and e-commerce, and the statistics of row and column number are illustrated in Table 1.

2.2 FewTUD Dataset

In this sub-section, we introduce the benchmark dataset of five few-shot table understanding tasks. For each task, we first invite annotators to manually annotate the dataset[5]. To follow the setting of few-shot learning, we fix the test set as *query set*, and sample different numbers of samples (N-shot) from training set as *support set*. During experiments, we randomly sample the *support set* from training set for 10 times, then report the average performance as the final result. Table 2 shows the detailed statistics of our benchmark dataset. Examples of each benchmark dataset are also illustrated in Figure 3 for better understanding.

Table Fact Verification. Table Fact Verification is a fundamental task for NLP and can benefit many downstream applications, such as misinformation detection and fake news detection. This task aims to verify whether a NL hypothesis is entailed or refuted by the given table as knowledge. Considering there is no off-the-shelf Chinese table fact verification dataset, we then construct one based on the NL text-table pairs we collected. 20 crowdsourcing annotators were invited to label the NL text-table pairs and check the quality. If the NL text claims the fact from the corresponding table, the pair is labelled as positive; otherwise, the anno-

[5]Annotations have been cross-checked and the inter-annotator agreement score is 90% (Fleiss' Kappa score (Fleiss and Cohen, 1973)).

Task	#Table	Train	Dev	Test	N-shot
Table Fact Verification	81	4.2k	0.8k	0.8k	{2,4,6,8,10,15}
Table QA	108	1.2k	0.5k	0.5k	{1,2,3,4,5,10}
Table Selection	266	2k	1.5k	1.5k	{1,2,3,4,5,10}
Schema Detection	100	1k	1k	1k	{1,2,3,4,5,10}
Table Classification	3k	1k	1k	1k	{5,10,15,20,30,50}

Table 2: Dataset statistics of few-shot experiments.

tators need to write a positive statement manually. Then, the sampled positive statements are rewritten as negative ones by replacing the key mentions in the NL texts with other cell values from the corresponding tables. In total, 5,800 fact statements for 81 unique tables were annotated and the ratio for positive and negative sentences is 1:2. For example, suppose a given table is about the milk powder product, the positive statement is "澳大利亚进口奶粉12-36月龄幼儿适用 (Milk powder imported from Australia is suitable for children aged 12-36 months)", while the negative statement is "意大利进口奶粉12-36月龄幼儿适用(Milk powder imported from Italy is suitable for children aged 12-36 months)". The model needs to verify which statement is true based on the table information. The evaluation metric for this task is F_1 score.

Table Question Answering. Given a table and question, Table QA aims to locate the exact cell in table that can answer the question. Given a query about *"How tall is Kobe Bryant?"* and a table on athletes information, the task is designed to pick the cell value *198cm* involved in the table to answer the question. Considering that it is labor-intensive and time-consuming to collect large-scale, high-quality question-answer pairs based on tables, we set up a question generation framework following Shi et al. (2022) to obtain questions given a table and answers. Specifically, we leverage the fine-tuned T5 model (Raffel et al., 2019) to generate questions automatically, then ask the annotators to manually check these ⟨question, answer, table⟩

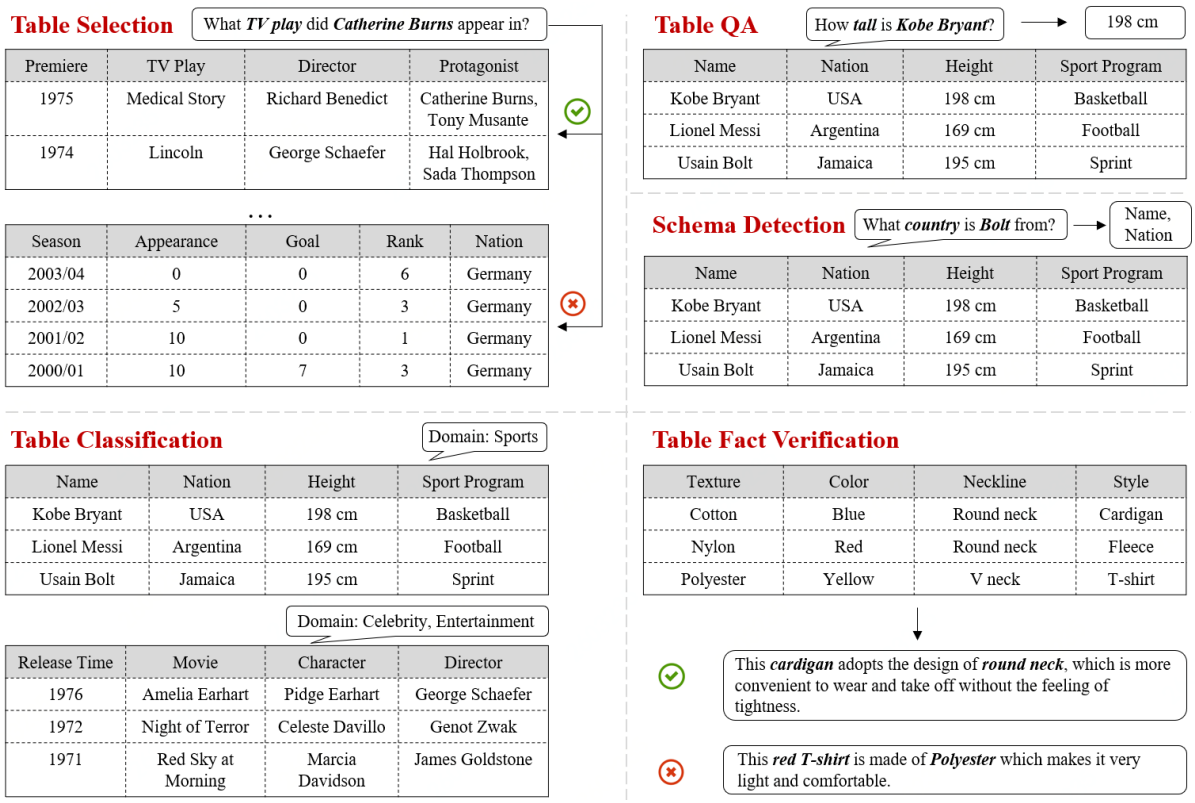

Figure 3: Examples of benchmark dataset **FewTUD**. The original data is in Chinese, we translate it into English for illustration.

triplets and rewrite semantically irrelevant or disfluent questions. Two public Chinese Machine Reading Comprehension datasets (He et al., 2018; Wang et al., 2020a) are used to train the T5 model.

Table Selection. Table selection or table retrieval is an important task as table contains valuable information to explore in various domains. This task aims to select the most relevant table from a list of candidates to answer the given query. Here, we construct the dataset for few-shot table selection similar to Table QA task. We first generate 5,000 queries for 266 different tables given the table-related context and its overlapped cell value. Then annotators are invited to re-check the generated queries and rewrite those disfluent ones. The revised query-table pair is noted as positive, while the one with replaced table (any other table) is negative. The ratio of positive and negative samples is 1:9 for test set, and 1:1 for training and validation sets. Given a query *"What TV play did Catheriner Burns appear in?"* and a list of tables, the model is expected to compute the matching score between each query-table pair and extract the most appropriate table to answer this question. $R_{10}@1$ is the evaluation metric for this ranking task (Lowe et al., 2015).

Schema Detection. As an important task for semantic parsing, schema detection bridges the gap between NL query and database schema. Given a query and table, schema detection requires to identify the column names mentioned in the query. For instance, given a query *"What country is Bolt from?"* and a table containing information on athletes, the model is required to predict the related column names of *Name, Nation* involved in the query. It is a challenging task aiming to explore the model's performance on covering semantic and structural correspondences and extracting general knowledge from structural data during table-query interaction. Here, we construct the dataset based on two public Chinese datasets (Wang et al., 2020b; Sun et al., 2020). Instead of using the whole sketch for SQL generation, we only keep the column information in SELECT, ORDER clauses, and WHERE conditions as the ground truth for schema detection.

Table Classification. Different from the above mentioned tasks that jointly learn the representations of table and text, table classification focuses on table understanding without additional text-based input. Given the table on movies and their characters' information, the task is expected to predict the multiple domain labels. Here we construct the dataset based on the domain labels we assigned to the collected tables in TPC-1M. The task can be formulated as a multi-label classification, which is designed to examine the performance of table pre-training models on structured-information understanding. For example, suppose a table contains Vivien Leigh's films, the model is expected to predict the domain labels *Celebrity, Entertainment*.

3 Table Pre-training Baseline

3.1 Model Architecture

Figure 4 illustrates the architecture of our proposed model FewTPT, which is based on the pre-trained language model BERT (Devlin et al., 2018) to encode the table and NL text and learn the structural-aware representations.

Input Embedding. FewTPT linearizes the input into a sequence of tokens by concatenating the query and table meta data by rows. For each cell in the table input, we adopt row linearization (Yin et al., 2020) to represent a cell with column name, column type and cell value together. Moreover, a [CLS] token is inserted at the beginning of whole input sequence and each cell is separated by the [SEP] symbol. Thus, for each token T_i of position i, the embedding E_T^i is defined as follows:

$$E_T^i = E_W^i + E_S^i + E_P^i + E_R^i, \tag{1}$$

where E_W, E_S, and E_P are the token embedding, segment embedding, and position embedding following Devlin et al. (2018). E_R represents row embedding inspired by Herzig et al. (2020).

Gated Cell Representation. Although row linearization method can accommodate the input of tabular data, previous methods (Yin et al., 2020) simply utilize the pooling of the column name, type and value as representation of a cell. Considering that the column name and cell value emphasize different information, it's necessary to distinguish the table column/cell separately. Here, we adopt a gated fusion mechanism to selectively integrate column name E_n^{jk}, column type E_t^{jk} and cell value E_v^{jk} to obtain the cell representation E_c^{jk}:

$$\begin{aligned} g^{jk} &= \sigma(W_g E_n^{jk} + U_g E_t^{jk} + V_g E_v^{jk} + b), \\ E_c^{jk} &= g^{jk} \odot E_n^{jk} + (1 - g^{jk}) \odot E_v^{jk} + E_t^{jk}, \end{aligned} \tag{2}$$

where W_g, U_g, V_g are learnable matrices, and jk represent the jth column and k^{th} row of the table.

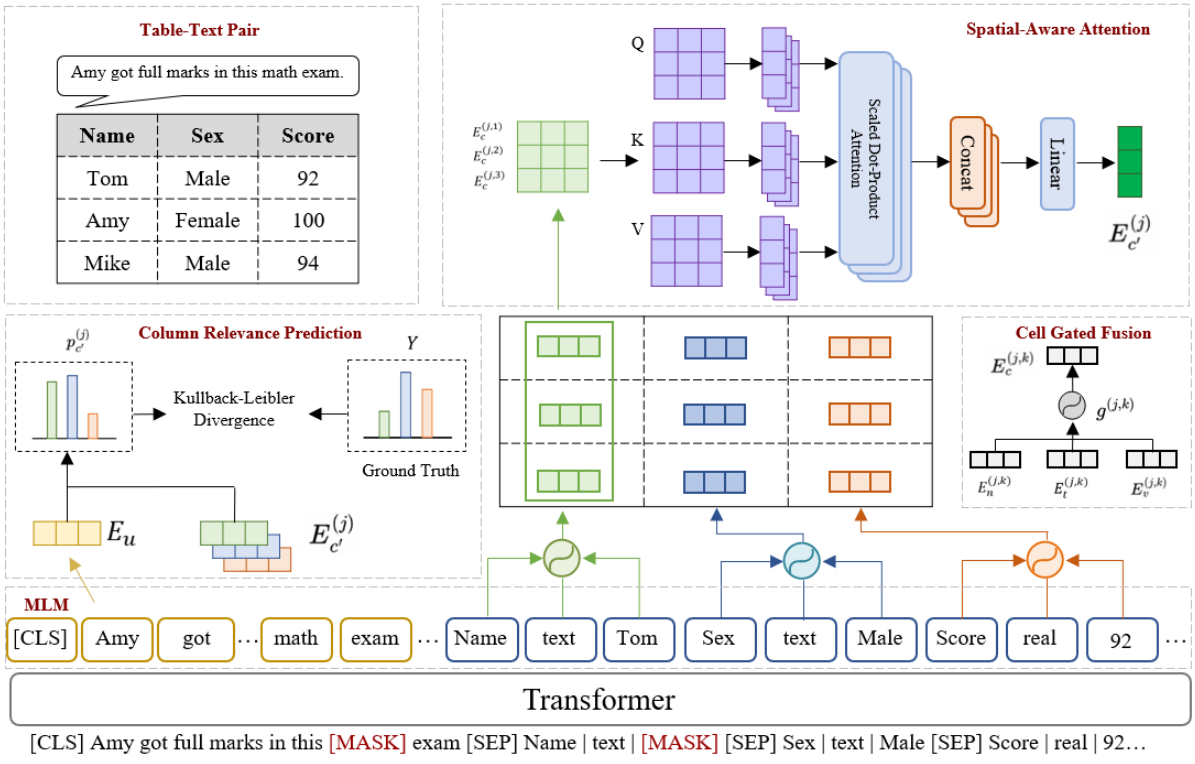

Figure 4: Model Architecture. (1) The model linearizes NL text and table into a sequence of tokens as input, and the gated fusion and row-aware attention are leveraged to get column representation (row representation can be obtained in the same way)). (2) The pre-training tasks includes MLM, Column Relevance Prediction (optimised with KL loss), and Row Relevance Prediction.

Spatial-Aware Representation. Inspired by self-attention (Vaswani et al., 2017), there are two kinds of attention mechanism in our framework, column-aware attention and row-aware attention, which attend each gated cell representation of a table both horizontally and vertically, namely spatial-aware attention. Assume that the number of columns and rows are C and M in a table, where $j \in \mathbb{R}^C, k \in \mathbb{R}^M$, the row-aware attention can be defined as:

$$
\begin{aligned}
e^{jk} &= (W_Q E_c^{jk})(W_K E_c^{jk})^T / \sqrt{d}, \\
a^{jk} &= \frac{\exp(e^{jk})}{\sum_{k=1}^{M} \exp(e^{jk})}, \\
E_{c'}^{j} &= \sum_{k=1}^{M} a^{jk} W_V E^{jk},
\end{aligned}
\tag{3}
$$

where W_Q,W_K,W_V are weight matrices for row-aware attention and d is dimension of W_Q. We perform row-aware attention here to obtain the column representation $E_{c'}^{j}$. Similarly, the column-aware attention can be defined in the same way to obtain the row representation $E_{r'}^{k}$.

Final Output. The output of FewTPT includes four parts: the representation of [CLS] token (E_{cls}), the representations for the tokens of NL text (E_u), and the interactive representations for row ($E_{r'}$) and column ($E_{c'}$) respectively.

3.2 Pre-Training Objectives

We implement three pre-training tasks, including Masked Language Modeling (MLM) and its variants to learn contextual representations for tokens in NL and table, Column Relevance Prediction (CRP) and Row Relevance Prediction (RRP) to capture the semantic interactions between the column/row and the give NL text respectively.

Masked Language Modeling. MLM is a widely-used objective for pre-training, which encourages the model to capture the contextual information of given sequence. Previous works also employ the variants of MLM for table pre-training (Herzig et al., 2020; Deng et al., 2020). Inspired by this, we devise three kinds of MLM tasks. Firstly, for tokens in NL text, we randomly mask 20% of individual sub-tokens and then recover the masked ones. Secondly, for columns names in the table, we randomly select 15% column names of the input table and require the model to predict them with column type and cell value information surrounded. Thirdly, considering cell values are basic units to record content in a table, we mask 15% of

cell values as well. Following Wang et al. (2021), we randomly select out cell strings from the table as candidates, then at each blanked position, we encourage the model to retrieve its corresponding string. We formulate all MLMs as multi-class classification and use cross entropy for training.

Row Relevance Prediction. The goal for row relevance prediction (RRP) is to predict whether a row is mentioned in the NL or not. As our table pre-training corpus provides the relevance scores X^k between NL text and each row as auxiliary supervised signals, we encourage the model to predict the relevance score for each row. For each row representation $E^k_{r'}$, we apply a fully connected layer with *sigmoid* activation function σ to obtain the probability of whether the row is mentioned in the utterance or not. Then Mean Squared Error (MSE) loss is leveraged as training objective:

$$\begin{aligned} p^k_{r'} &= \sigma(W_{r'} E^k_{r'} + E_u), \\ \mathcal{L}_{r'} &= \sum\nolimits_{k=1}^{M} ||p^k_{r'} - X^k||^2 / M. \end{aligned} \tag{4}$$

Column Relevance Prediction. Similar to RPP, given the column representation $E^j_{c'}$, the goal of column relevance prediction (CRP) is to predict the relevance score between each column and NL text. Since we also have the auxiliary supervised relevance score for each column, the predicted relevance score distribution is forced to fit that of supervised signals between text and columns:

$$p^j_{c'} = \text{softmax}(W_{c'} E^j_{c'} + E_u), \tag{5}$$

where j represents the index of different column. Thus, we take the Kullback-Leibler Divergence (KL) (Kullback and Leibler, 1951) as training objective to optimize the column relevance prediction:

$$\mathcal{L}_{c'} = \sum\nolimits_{j=1}^{C} p^j_{c'} \log \frac{p^j_{c'}}{Y^j}, \tag{6}$$

where Y^j is column-utterance relevance score obtained from the training corpus and $p^j_{c'}$ is the predicted score from our model.

4 Experiments

Here, we first introduce the experimental setup of pre-training and the comparable baselines, then we analyze the main results and also conduct ablation study to show the contribution of different modules.

4.1 Experimental Setup

Pre-Training Configuration. For pre-training setup, we train **FewTPT** with TPC-1M for 6 epochs, with the batch size of 4 on 4 Tesla V100 GPUs. Specifically, we set the learning rate as 4e-5, the number of attention heads as 12, and the weight decay and dropout are set as 0.01 and 0.1 respectively.

Comparable Models. For few-shot table understanding tasks, we adopt three strong baselines for comparison: (1) **BERT** (Devlin et al., 2018), which is the popular general-purpose PLM trained with free-form text. We linearize the table and feed the concatenated query and table into the official Chinese version[6] for fine-tuning and inference. (2) **TaBERT** (Yin et al., 2020), which is a recently proposed table pre-training model designed to jointly learn representations for NL text and (semi-)structured tables. Since the released model is in English version, we pre-train the model with TPC-1M corpus from scratch with the official source code[7]. (3) **SDCUP** (Hui et al., 2021), which proposes a schema dependency pre-training objective to impose the desired inductive bias into the learned representations for table pre-training. We fine-tune the official released model[8] directly for comparison.

N-shot	1	2	3	4	5	10
Table QA						
BERT	14.0	26.4	33.9	41.6	43.4	58.4
TaBERT	14.6	28.6	34.4	44.0	44.6	59.0
FewTPT	**15.8**	**30.6**	**35.8**	**45.4**	**46.2**	**60.6**
Schema Detection						
BERT	14.1	34.3	42.5	49.9	55.1	73.3
TaBERT	16.3	36.7	44.5	51.8	55.4	73.9
FewTPT	**16.9**	**37.5**	**44.7**	**52.2**	**56.1**	**74.2**
Table Selection						
BERT	83.2	82.4	85.1	87.5	86.2	85.6
SDCUP	81.6	82.4	84.3	84.9	85.5	86.6
TaBERT	83.8	85.6	86.7	87.9	88.6	89.5
FewTPT	**86.2**	**88.3**	**88.7**	**89.0**	**89.5**	**91.2**

Table 3: Few-shot table understanding performance on the tasks of Table QA, Schema Detection, and Table Selection.

[6]https://github.com/google-research/bert

[7]https://github.com/facebookresearch/TaBERT

[8]https://github.com/alibaba/AliceMind/tree/main/SDCUP. Considering SDCUP is tailored for the NL2SQL task, we only compare with it on two tasks to avoid modifying its model architecture too much.

4.2 Main Results

Experimental results are illustrated in Table 3 and Figure 5. For Table QA, Schema Detection and Table Selection, we evaluate all the models by ranging training samples N from {1,2,3,4,5,10}. It's observed that, (1) with the increase of N, the performance of all the models improves rapidly, indicating the size of training sample plays a vital role to all tasks. The results also demonstrate how many training samples are needed at least for each task to reach an *acceptable* performance, which is a critical issue but ignored by previous works. We argue that our experiments can be a valuable reference for table understanding applications in real-world scenario. (2) Compared to all baselines, our proposed model **FewTPT** yields substantial gains on all three tasks, and the benefits are more pronounced when N is small. It demonstrates the advantages of our proposed table pre-training method and the contribution of TPC-1M corpus. (3) **FewTPT** surpasses **TaBERT** consistently. Considering both models were pre-trained with the same corpus, it reveals the gains are from our model structure and pre-training objectives, which can fuse the information from NL text and table seamlessly and finally facilitate the downstream table understanding tasks.

Figure 5 demonstrates the experimental results on Table Fact Verification and Table Classification tasks in a more direct way. We observe the similar trend that **FewTPT** outperforms all baselines by a large margin. Especially, **FewTPT** progressively outperforms **TaBERT** by nearly **12.3%** ($N = 8$) on Table Fact Verification. We conjecture it's because this task heavily relies on the deep interactions between NL text and table content to discriminate whether the statement is correct or not, thus the proposed spatial-aware attention and row/column relevance prediction objectives are genuinely required and can benefit the table understanding. For Table Classification, the performances of **FewTPT** and **TaBERT** are comparable. Considering there is no query in the input, we guess the advantage of deep interactions between text and table in our model may be weakened.

4.3 Ablation Study

We conduct further experiments to figure out the contribution of each component, including the improved Masked Language Modeling (MLM) objective, Column Relevance Prediction (CRP), Row Relevance Prediction (RRP), and Cell Gated Fusion (CGF) mentioned in Section 3. Due to space limitation, we perform ablation study on Table QA and Schema Detection, under 2-shot setting. Figure 6 presents the experimental results by adding each component in a cumulative way, demonstrating the necessity of each component in our method.

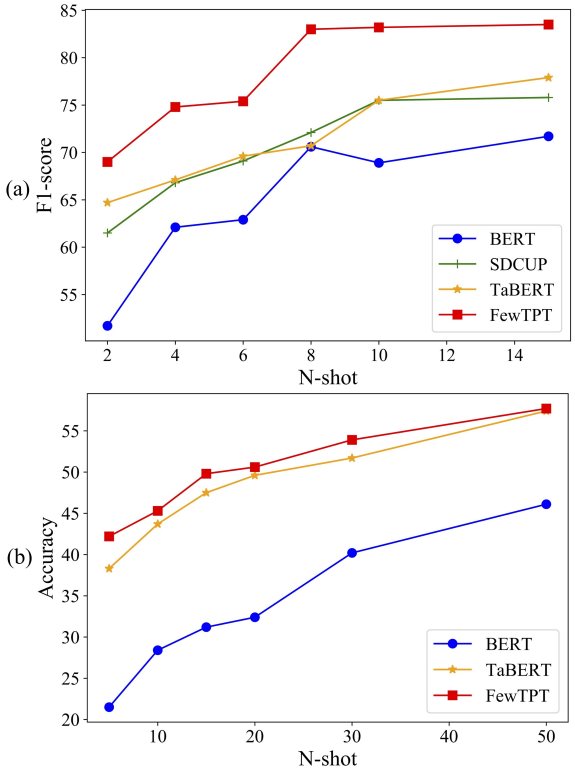

Figure 5: Few-shot model performance on the tasks of Table Fact Verification (a) and Table Classification (b).

5 Related Work

Large-Scale Tabular Corpus. Most of the recent table-text pre-training tasks utilize web-crawled tables and context from English Wikipedia (Lehmberg et al., 2016). Apart from the extreme noise contained in the tables, the context is mainly mined based on position information and may not be semantically related to tables. Some table-to-text generation task (Lewis et al., 2020) are also proposed to provide well-controlled text-table corpus with either grammar-supported data generation or powerful generative models (Nan et al., 2021; Liu et al., 2018; Parikh et al., 2020). However, the linguistic diversity of the generated data is limited. Besides, most of the current studies focus on English. Considering the increasing demand of table understanding tasks in Chinese, the deficiency of large-scale table-text corpus blocks the pre-training exploration on Chinese structured data. Therefore,

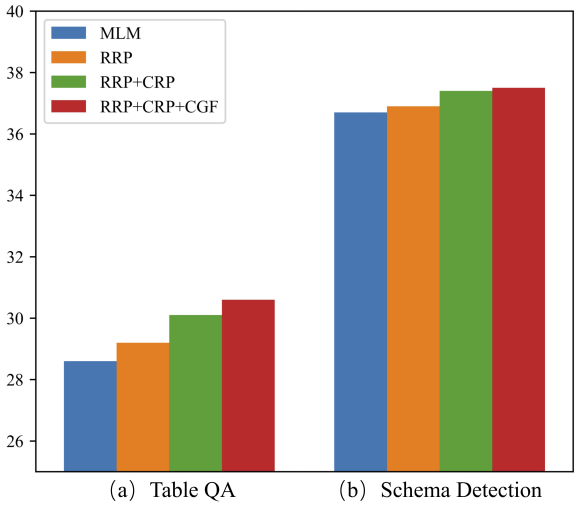

Figure 6: Ablation study of our method on Table QA and Schema Detection.

in this work, we collect a large-scale table pre-training corpus with more than 1 million tables and will release to the community soon.

Table Pre-Training. Inspired by the recent success of leveraging PLMs on tasks with huge amount of unstructured natural language (Devlin et al., 2018; Liu et al., 2019), some researches have tried to apply pre-training approaches into structured tabular data. TAPAS (Herzig et al., 2020) and TaBERT (Yin et al., 2020) introduce novel pre-training methods to learn the joint representations of table and text with a large-scale web crawled tables and their contextual natural language descriptions. The vanilla Masked Language Modeling (MLM) is adopted by either masking the tokens from input text or tokens from tables. To cultivate the alignment between utterances and table context, some researchers introduce supervised pre-training objectives by involving the logic language interaction such as SQL semantic prediction (Yu et al., 2021), SQL generation (Shi et al., 2021) and SQL execution (Liu et al., 2021). However, to ensure the quality of synthesised training data, they either design complicated generation templates manually or rely on pre-trained generative models. Whereas these generated training data is lack of variety compared with the natural utterances in real-world scenario of table-text understanding. Besides, some other researchers focus on table encoding with relational and complex structures (Iida et al., 2021; Deng et al., 2021). However, most of the unsupervised training objectives involved in above researches neglect enhancing the semantic interactions between natural languages and tables (Shi et al., 2022; Cheng et al., 2021; Dong et al., 2022). Differently, we propose a novel table pre-training method equipped by cell gated fusion, spatial-aware attention, masked language modeling, row/column relevance prediction to catch the deep semantic interactions between NL text and table.

Few-Shot Learning. Recently, lots of researches have focused on few-shot learning in NLP and propose a variety of methods including meta-learning (Kaiser et al., 2017), embedding learning (Bertinetto et al., 2016), memory-based learning (Kaiser et al., 2017). Whereas the recent flouring of PLMs (Brown et al., 2020) achieve remarkable few-shot performance solely by leveraging a natural-language prompt and a few task demonstrations as input context (Gao et al., 2021). However, few of them have paid attention to the challenge of few-shot table understanding or table-text interactions tasks where only few text-table pairs are available. Chang et al. (2020); Chen et al. (2021) explore the zero-shot text-to-sql task, and both of them illustrate the importance of leveraging the abundant table cell information and header information during training to improve table-text semantic relevance. To the best of our knowledge, there is no well-designed benchmark dataset and baseline for the few-shot table understanding. In this work, we are dedicated to fill the gap to facilitate future research.

6 Conclusion

In this paper, we focus on the few-shot table understanding problem and establish a benchmark dataset with five downstream tasks including Table Fact Verification, Table QA, Table Selection, Schema Detection, and Table Classification. We also contribute a large-scale Chinese tabular corpus which covers 1 million tables across 13 domains and the semantically-associated NL text. Finally, we provide a table pre-training method and conduct extensive experiments on the few-shot table understanding benchmark to set up the baselines. Experimental results demonstrate that catching the interactions between text and tables can improve the downstream tasks significantly. We hope the benchmark, tabular corpus, and the baselines can facilitate the future research on this field. In the future, we will explore more table structure friendly objectives to improve the pre-training.

References

Luca Bertinetto, João F Henriques, Jack Valmadre, Philip Torr, and Andrea Vedaldi. 2016. Learning feed-forward one-shot learners. In *Advances in neural information processing systems*, pages 523–531.

Ben Bogin, Jonathan Berant, and Matt Gardner. 2019. Representing schema structure with graph neural networks for text-to-SQL parsing. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 4560–4565, Florence, Italy. Association for Computational Linguistics.

Tom B Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. 2020. Language models are few-shot learners. *arXiv preprint arXiv:2005.14165*.

Shuaichen Chang, Pengfei Liu, Yun Tang, Jing Huang, Xiaodong He, and Bowen Zhou. 2020. Zero-shot text-to-sql learning with auxiliary task. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pages 7488–7495.

Wenhu Chen, Hongmin Wang, Jianshu Chen, Yunkai Zhang, Hong Wang, Shiyang Li, Xiyou Zhou, and William Yang Wang. 2020. Tabfact: A large-scale dataset for table-based fact verification. In *8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, April 26-30, 2020*. OpenReview.net.

Yongrui Chen, Xinnan Guo, Chaojie Wang, Jian Qiu, Guilin Qi, Meng Wang, and Huiying Li. 2021. Leveraging table content for zero-shot text-to-sql with meta-learning. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 35, pages 3992–4000.

Zhoujun Cheng, Haoyu Dong, Fan Cheng, Ran Jia, Pengfei Wu, Shi Han, and Dongmei Zhang. 2021. Fortap: Using formulae for numerical-reasoning-aware table pretraining. *arXiv preprint arXiv:2109.07323*.

Xiang Deng, Ahmed H. Awadallah, Chris Meek, Alex Polozov, Huan Sun, and Matthew Richardson. 2021. Structure-grounded pretraining for text-to-sql. In *NAACL 2021*.

Xiang Deng, Huan Sun, Alyssa Lees, You Wu, and Cong Yu. 2020. Turl: table understanding through representation learning. *Proceedings of the VLDB Endowment*, 14(3):307–319.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*.

Haoyu Dong, Zhoujun Cheng, Xinyi He, Mengyu Zhou, Anda Zhou, Fan Zhou, Ao Liu, Shi Han, and Dongmei Zhang. 2022. Table pretraining: A survey on model architectures, pretraining objectives, and downstream tasks. *arXiv preprint arXiv:2201.09745*.

Joseph L Fleiss and Jacob Cohen. 1973. The equivalence of weighted kappa and the intraclass correlation coefficient as measures of reliability. *Educational and psychological measurement*, 33(3):613–619.

Tianyu Gao, Adam Fisch, and Danqi Chen. 2021. Making pre-trained language models better few-shot learners. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 3816–3830, Online. Association for Computational Linguistics.

Wei He, Kai Liu, Jing Liu, Yajuan Lyu, Shiqi Zhao, Xinyan Xiao, Yuan Liu, Yizhong Wang, Hua Wu, Qiaoqiao She, Xuan Liu, Tian Wu, and Haifeng Wang. 2018. Dureader: a chinese machine reading comprehension dataset from real-world applications. In *QA@ACL*.

Jonathan Herzig, Pawel Krzysztof Nowak, Thomas Müller, Francesco Piccinno, and Julian Eisenschlos. 2020. TaPas: Weakly supervised table parsing via pre-training. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 4320–4333, Online. Association for Computational Linguistics.

Binyuan Hui, Xiang Shi, Ruiying Geng, Binhua Li, Yongbin Li, Jian Sun, and Xiaodan Zhu. 2021. Improving text-to-sql with schema dependency learning. *arXiv preprint arXiv:2103.04399*.

Hiroshi Iida, Dung Thai, Varun Manjunatha, and Mohit Iyyer. 2021. TABBIE: Pretrained representations of tabular data. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 3446–3456.

Edwin T Jaynes. 1982. On the rationale of maximum-entropy methods. *Proceedings of the IEEE*, 70(9):939–952.

Łukasz Kaiser, Ofir Nachum, Aurko Roy, and Samy Bengio. 2017. Learning to remember rare events. *arXiv preprint arXiv:1703.03129*.

M. A. Khalid, V. Jijkoun, and MD Rijke. 2007. Machine learning for question answering from tabular data. In *International Conference on Database and Expert Systems Applications*.

Solomon Kullback and Richard A Leibler. 1951. On information and sufficiency. *The annals of mathematical statistics*, 22(1):79–86.

Brenden M Lake, Ruslan Salakhutdinov, and Joshua B Tenenbaum. 2015. Human-level concept learning through probabilistic program induction. *Science*, 350(6266):1332–1338.

Oliver Lehmberg, Dominique Ritze, Robert Meusel, and Christian Bizer. 2016. A large public corpus of web tables containing time and context metadata.

Proceedings of the 25th International Conference Companion on World Wide Web.

Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Veselin Stoyanov, and Luke Zettlemoyer. 2020. BART: denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, ACL 2020, Online, July 5-10, 2020*, pages 7871–7880. Association for Computational Linguistics.

Qian Liu, Bei Chen, Jiaqi Guo, Zeqi Lin, and Jianguang Lou. 2021. Tapex: Table pre-training via learning a neural sql executor. *arXiv preprint arXiv:2107.07653*.

Tianyu Liu, Kexiang Wang, Lei Sha, Baobao Chang, and Zhifang Sui. 2018. Table-to-text generation by structure-aware seq2seq learning. In *Thirty-Second AAAI Conference on Artificial Intelligence*.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Roberta: A robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692*.

Ryan Lowe, Nissan Pow, Iulian Serban, and Joelle Pineau. 2015. The ubuntu dialogue corpus: A large dataset for research in unstructured multi-turn dialogue systems. In *SIGDIAL Conference*.

Linyong Nan, Dragomir Radev, Rui Zhang, Amrit Rau, Abhinand Sivaprasad, Chiachun Hsieh, Xiangru Tang, Aadit Vyas, Neha Verma, Pranav Krishna, Yangxiaokang Liu, Nadia Irwanto, Jessica Pan, Faiaz Rahman, Ahmad Zaidi, Mutethia Mutuma, Yasin Tarabar, Ankit Gupta, Tao Yu, Yi Chern Tan, Xi Victoria Lin, Caiming Xiong, Richard Socher, and Nazneen Fatema Rajani. 2021. DART: Open-domain structured data record to text generation. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 432–447, Online. Association for Computational Linguistics.

Suphakit Niwattanakul, Jatsada Singthongchai, Ekkachai Naenudorn, and Supachanun Wanapu. 2013. Using of jaccard coefficient for keywords similarity. In *Proceedings of the international multiconference of engineers and computer scientists*, volume 1, pages 380–384.

Ankur Parikh, Xuezhi Wang, Sebastian Gehrmann, Manaal Faruqui, Bhuwan Dhingra, Diyi Yang, and Dipanjan Das. 2020. ToTTo: A controlled table-to-text generation dataset. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1173–1186, Online. Association for Computational Linguistics.

Bowen Qin, Lihan Wang, Binyuan Hui, Ruiying Geng, Zheng Cao, Min Yang, Jian Sun, and Yongbin Li. 2021. Sdcup: Schema dependency-enhanced curriculum pre-training for table semantic parsing. *arXiv preprint arXiv:2111.09486*.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J Liu. 2019. Exploring the limits of transfer learning with a unified text-to-text transformer. *arXiv preprint arXiv:1910.10683*.

Peng Shi, Patrick Ng, Feng Nan, Henghui Zhu, Jun Wang, Jiarong Jiang, Alexander Hanbo Li, Rishav Chakravarti, Donald Weidner, Bing Xiang, and Zhiguo Wang. 2022. Generation-focused table-based intermediate pre-training for free-form question answering. *Proceedings of the Genetic and Evolutionary Computation Conference Companion*.

Peng Shi, Patrick Ng, Zhiguo Wang, Henghui Zhu, Alexander Hanbo Li, Jun Wang, Cicero Nogueira dos Santos, and Bing Xiang. 2021. Learning contextual representations for semantic parsing with generation-augmented pre-training. *Proceedings of the AAAI Conference on Artificial Intelligence*, 35(15).

Huan Sun, Hao Ma, Xiaodong He, Wen-tau Yih, Yu Su, and Xifeng Yan. 2016. Table cell search for question answering. In *Proceedings of the 25th International Conference on World Wide Web, WWW 2016, Montreal, Canada, April 11 - 15, 2016*. ACM.

Ningyuan Sun, Xuefeng Yang, and Yunfeng Liu. 2020. Tableqa: a large-scale chinese text-to-sql dataset for table-aware sql generation. *arXiv preprint arXiv:2006.06434*.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, 4-9 December 2017, Long Beach, CA, USA*, pages 5998–6008.

Bingning Wang, Ting Yao, Qi Zhang, Jingfang Xu, and Xiaochuan Wang. 2020a. Reco: A large scale chinese reading comprehension dataset on opinion. In *AAAI*.

Jingjing Wang, Haixun Wang, Zhongyuan Wang, and Kenny Qili Zhu. 2012. Understanding tables on the web. In *Conceptual Modeling - 31st International Conference ER 2012, Florence, Italy, October 15-18, 2012. Proceedings*, volume 7532 of *Lecture Notes in Computer Science*, pages 141–155. Springer.

Lijie Wang, Ao Zhang, Kun Wu, Ke Sun, Zhenghua Li, Hua Wu, Min Zhang, and Haifeng Wang. 2020b. DuSQL: A large-scale and pragmatic Chinese text-to-SQL dataset. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 6923–6935, Online. Association for Computational Linguistics.

Zhiruo Wang, Haoyu Dong, Ran Jia, Jia Li, Zhiyi Fu, Shi Han, and Dongmei Zhang. 2021. Tuta: Tree-based transformers for generally structured table pre-training. In *Proceedings of the 27th ACM SIGKDD Conference on Knowledge Discovery & Data Mining*, pages 1780–1790.

Pengcheng Yin, Graham Neubig, Wen-tau Yih, and Sebastian Riedel. 2020. TaBERT: Pretraining for joint understanding of textual and tabular data. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 8413–8426, Online. Association for Computational Linguistics.

Tao Yu, Chien-Sheng Wu, Xi Victoria Lin, bailin wang, Yi Chern Tan, Xinyi Yang, Dragomir Radev, richard socher, and Caiming Xiong. 2021. Gra{pp}a: Grammar-augmented pre-training for table semantic parsing. In *International Conference on Learning Representations*.

Hongzhi Zhang, Yingyao Wang, Sirui Wang, Xuezhi Cao, Fuzheng Zhang, and Zhongyuan Wang. 2020. Table fact verification with structure-aware transformer. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing, EMNLP 2020, Online, November 16-20, 2020*, pages 1624–1629. Association for Computational Linguistics.

Shuo Zhang and Krisztian Balog. 2018. Ad hoc table retrieval using semantic similarity. In *Proceedings of the 2018 World Wide Web Conference on World Wide Web, WWW 2018, Lyon, France, April 23-27, 2018*, pages 1553–1562. ACM.

Tafsir Dataset: A Novel Multi-Task Benchmark for Named Entity Recognition and Topic Modeling in Classical Arabic Literature

Sajawel Ahmed[1,2,3], Rob van der Goot[3], Misbahur Rehman[2], Carl Kruse[2], Ömer Özsoy[2], Alexander Mehler[1], Gemma Roig[1]
[1]Faculty for Computer Science and Mathematics, Goethe University Frankfurt
[2]Faculty for Linguistics, Cultures, and Arts, Goethe University Frankfurt
[3]Department of Computer Science, IT University of Copenhagen
{sahmed}@em.uni-frankfurt.de

Abstract

Various historical languages, which used to be lingua franca of science and arts, deserve the attention of current NLP research. In this work, we take the first data-driven steps towards this research line for Classical Arabic (CA) by addressing *named entity recognition* (NER) and *topic modeling* (TM) on the example of CA literature. We manually annotate the encyclopedic work of *Tafsir Al-Tabari* with span-based *NEs*, sentence-based *topics*, and span-based *subtopics*, thus creating the *Tafsir Dataset* with over 51,000 sentences, the first large-scale multi-task benchmark for CA. Next, we analyze our newly generated dataset, which we make open-source available, with current language models (lightweight BiLSTM, transformer-based MaChAmP) along a novel *script compression method*, thereby achieving state-of-the-art performance for our target task *CA-NER*. We also show that *CA-TM* from the perspective of historical topic models, which are central to Arabic studies, is very challenging. With this interdisciplinary work, we lay the foundations for future research on automatic analysis of CA literature.

1 Introduction

All languages deserve equal technologies. Named entity recognition (NER) and topic modeling (TM) are a crucial part of various downstream tasks in natural language processing (NLP), such as Entity Linking, Relation Extraction, and ultimately Question Answering. For such tasks, many research institutes and individual scholars put their emphasis on popular, high-resource languages like English, where there is already a large amount of previous work and resources available (Rajpurkar et al., 2018; Dzendzik et al., 2021; Cambazoglu et al., 2021). This definitely accelerates the progress of the ongoing data-driven NLP. However, many historical languages, such as Ancient Egyptian, Ancient Greek, and especially Classical Arabic (CA), which used to be the lingua franca of science and arts, have been mostly neglected by the NLP community. These languages possess large volumes of historical literature (CA: e.g. *Liber Algebrae et Almucabola*, *Canon Medicinae*, *Tafsir Al-Tabari*), which were and still are to this date relevant for many communities and societies, lay their foundations and even shape their further development. In order to perform historical analysis which are relevant for our modern age, we need to let these *forgotten* low-resource languages benefit from the wave of machine learning (ML) progress, thus making historical texts accessible to modern studies and approaching ethically an egalitarian state of NLP research.

To this end, within the project *Linked Open Tafsir* (Ahmed et al., 2022), firstly, we create the *Tafsir Dataset* by annotating the CA encyclopedic books of *Tafsir Al-Tabari* on exegetical studies of law, ethics and philosophy. This is done with respect to span-based *NEs*, sentence-based *topics* and span-based *subtopics*, thereby producing over 51,000 sentences and presenting the first multi-task benchmark for CA with three independent tasks.

Rasm + I'jam + Tashkil (Vocalized Arabic)
قَالَ اَحْمَدُ لِسَارِيَةَ فِيْ مَكَّةَ: كُلُوْا وَاشْرَبُوْا هَنِيْئًا
Rasm + I'jam (Standard Arabic)
قال احمد لسارية فى مكة: كلوا واشربوا هنيئا
Rasm (Skeleton Arabic)
قال احمد لساريه فى مكه : كلوا واسربوا هسا
NER & TM Output (Skeleton Arabic)
#topic=kalam قال [احمد]PER ل[ساريه]PER فى [مكه]LOC: كلوا واسربوا هسا

Figure 1: Example for Arabic script-dependent preprocessing layers for the sentence *"Ahmed said to Saria in Mecca: eat and drink with happiness"* along NER & TM output.

Proceedings of the 29th International Conference on Computational Linguistics, pages 3753–3768
October 12–17, 2022.

Secondly, we develop a novel *script compression method* for Arabic text in order to examine its influence on the performance of neural models (see Figure 1). For this, we take the modern vocalized Arabic script and gradually transform it to its antique form of skeleton script *Rasm* from the 7th century by removing first, the vocalization marks *Tashkil* (consisting of dashes and circles), and second, the diacritic marks *I'jam* (consisting of dots), thus lowering the vocabulary size drastically by reducing the number of distinct letters from 280 (vocalized) over 28 (standard) to 16 (skeleton). From a historical critical perspective, the usage of this skeleton script is quite interesting as this was the first one to be used for documenting the text of the Quran. Thus, on a side note, by analyzing this ancient script, we shed light on the historical critical question of its readability.

Thirdly, we analyze our newly generated dataset, apply the leightweight BiLSTM (Lample et al., 2016; Ahmed and Mehler, 2018) and contrast its usage with *MaChAmp* (van der Goot et al., 2021), a toolkit for multi-task learning in NLP. This toolkit ideally fits to our multi-task benchmark, allowing us to conduct over 119 many-fold experimental setups with various Arabic pre-trained language models (LM), such as AraBERT, AraElectra, RemBERT. With these optimization steps, we produce the first major results for *CA-TM* and on top establish a state-of-the-art performance for *CA-NER* by achieving a value of up to *95.58% F1-score*.

Our work facilitates an automatic extraction of theological information so far buried in the bulk of paper manuscripts and volumes. By creating the necessary training data for tackling the task of NER and TM with various ML algorithms, we provide an open-source gold standard for the NLP community and hereby lay the foundations for future work on digitization of historical Arabic juridical and theological studies.

The remainder of the paper is organized as follows: Section 2 reviews related work, Section 3 presents the dataset, its historical source and provides details on the annotation tasks and their guidelines, Section 4 presents a sketch of the underlying methods, Section 5 reports and discusses our results, and, finally, Section 6 draw the conclusion.

2 Related Work

Not much work has been done in the field of NLP for CA as this language suffers from *resource poverty* in the ML community. For Modern Standard Arabic (MSA), there are only a handful of studies and resources open-source available. Noteworthy work specifically for MSA-NER has been done so far mainly by Benajiba et al. (2007) on *ANERCorp dataset* and by Mohit et al. (2012) on *AQMAR dataset*; both datasets along their NER models will be used as baselines here (see Table 1). Although these datasets are relatively small compared to those which are used for other languages in the community, to this date we do not have any other alternatives. For MSA-TM, again only few resources are freely available (El Kah and Zeroual, 2021), however, these are all built on modern web texts mainly from the genre of newspapers and social media. For the case of CA-TM, no prior work is known to the authors. Hence with our work, we lay the foundations for future research in this interdisciplinary field of historical NLP.

3 Tafsir Dataset: Annotation of Classical Arabic Literature

In this section, we describe the data source, the textual conversions performed to prepare the annotation task, the annotation guidelines and the annotation process itself.

3.1 Data Source: Raw Text to TEI Format

Al-Tabari *Al-Tabari*, in full *Abu Ja'far Muhammad ibn Jarir al-Tabari*, (born c. 839, Amol, Tabiristan, Iran—died 923, Baghdad, Iraq), is a religious scholar, author of enormous compendiums of early Islamic history and Quranic exegesis, who made a distinct contribution to the consolidation of Sunni thought during the 9th century. He condensed the vast wealth of exegetical and historical erudition of the preceding generations of Muslim scholars and laid the foundations for both Quranic and historical sciences. His major works were the *Exegesis of Al-Tabari* (Tafsir Al-Tabari) and the *History of Prophets and Kings*. In this study, we are focusing on his former work.

Edition of the book and TEI format Tafsir Al-Tabari has been published in various editions, the *Turki Edition* from 2001 is the most extensive and complete one, hence, this was chosen for our study. It is published in 26 volumes consisting of a total of 18,594 pages. The original text of this edition, which is vocalized, is freely available from different online sources such as the *King Saud University*, the *Shamela Software*, and from the well-known

Corpus	*Sent.*	*PER*	*LOC*	*ORG*	*TME*	*OTH*
ANERCorp-2007	5,887	3,598	*4,429*	2,231	n/a	1,115
AQMAR-2012	2,646	1,468	1,443	450	n/a	*2,474*
Tafsir-2022	**51,704**	**176,105**	**5,583**	**22,026**	**4,160**	**12,453**

Table 1: Major open-source NER Datasets for Arabic along our NER annotations in the Tafsir Dataset.

resource platform *Gawami' al-Kalim*[1], whose text is the most refined and accurate one according to a review of the linguists in our annotation team.

The raw text was transformed to the TEI format (with an adapted TEI model), which was selected due to its extensive usage in Digital Humanities (Maraoui et al., 2017). Furthermore, this format can be useful for additional data analytical inquiries (e.g. with XQuery).

Sentence splitting heuristic Sentence splitting has been addressed by various approaches (Schweter and Ahmed, 2019). However, if there is no punctuation available, it becomes challenging for many algorithms to find a stable solution. In the case of CA literature, we rarely find regular punctuation. In fact in this ancient literature, there was no concept of sentences in the modern sense. Therefore, we apply a heuristic, which first uses all possible punctuation (which are introduced by modern editing authors), then looks for some specific sense splitting words, e.g. and (*wa*), so (*fa*), then (*thumma*). With this, we achieve an average sentence length of 30 words, which proves to be useful according to our initial downstream task evaluations.

3.2 Annotation Tasks

We developed annotation guidelines for generating the Tafsir Dataset. For NER, we extended the standard task to the domain of theology. Our guidelines built on those developed for the NER dataset on German historical literature (Ahmed et al., 2019). We took the original German guideline text and adjusted it by incorporating domain-specific needs for CA. For TM, we categorized the number of topics according to the classical understanding of *tafsir studies* and its 15 fields (Al-Suyuti, 1505), and refined them further during our discussion sessions with the annotation team. The appendix shows the material which was provided to the annotation team, including the introductory example of annotations. Overall, the raw text was annotated chapter-wise by considering each verse as a single annotation task. By this scheme, we ensured that annotators had the contextual information they needed to make their interpretations.

3.2.1 Named Entities

NEs are entities that are referred to in natural language texts by proper nouns (PN) as unique individuals (e.g. Mecca, Asia, Tabari, Shia). PN are contrasted by *common names* (CN) which refer to classes of entities (e.g. city, continent, person, organization).

In our task of CA-NER, we focus on PN. However, it is not easy to differentiate between PN and CN. In the following, we provide details for each class of NE which we used to annotate our raw text (for annotation results see Table 1, for further examples of NEs see Appendix A).

Person (PER) Naming can be a complex process in classical Arab society (comparable to ancient Hebraic naming) (Almuhanna and Prunet, 2019). Full names are made of chains of single names, which can include the name of the city where the person was living. Once the full name is mentioned, short forms are usually used throughout the remainder of a text (e.g. Al-Tabari). In CA-NER, we consider all naming conventions found in the raw texts.

Location (LOC) Location names are mostly straightforward (either classical Arabic names, or names going back to ancient age of Babylonia). Sometimes, there is a ambiguity in their semantics, e.g. the word *Medina* (city) is not a PN per se, however, when it is used a short form for *Medina Al Munawwarah* ("The Enlightened City"), then it becomes a PN. Obviously, the word's meaning is highly context dependent.

Organization (ORG) We extended the modern definition of this class to the classical context of religious organizations (Jews, Christians, Muslims), their subgroups (Sunni, Shia, Ismailities), theological school of thoughts (Hanafi, Maliki, Shafi'i,

[1] https://gk.islamweb.net

Topic/Subtopic	Sent.	Span
adyan (non-Islamic relig.)	13,564	1,063
asbab (occas. of revelation)	3,086	997
fiqh (jurisprudence)	9,782	7,707
israiliyat (Judeo-Christian)	3,260	0
kalam (Islamic theology)	17,208	3,066
lugha (linguistics)	14,444	9,543
mushkilat (problem)	61	0
mutashabih (allegorical)	153	0
naskh (abrogation)	544	223
qiraat (recitation style)	1,525	2,519
sirah (prophetic biography)	1,193	215
sufism (mysticism)	7,749	881
takhsis (specification)	146	0
tikrar (repetition)	174	0
ulum (science)	2,520	823
total annotations	*75,409*	*27,037*

Table 2: Statistics for sentence-based topic and span-based subtopic annotation data.

Hanbali), tribes and clans (Hashim, Quraysh), and ethnic groups (Arabs, Greeks, Persians).

Time (TME) In the early 7th century, the moon calendar was still in its primary form, hence there was not a proper usage of numerical format like in our modern days. Therefore, dates were mostly written out in words, either only by day name, or sometimes including the month name, and rarely, the year. In CA-NER, we consider all possible variants and annotate them accordingly. Also well-known temporal entities, such as the *Day of Judgment* (Yawm Al-Din), are annotated with the tag *TME*.

Other (OTH) All NEs which did not fit into the former class were annotated with the tag *OTH*, such as name of languages (Arabic, Greek, Latin), angels (Gabriel, Michael, Raphael), and (polytheistic) deities (Al Uzza, Al Lat, Manat, Baal).

3.2.2 Topic Modeling

TM is the task of mapping (segments of) texts to a fixed set of *topics* according to a multiclass setting (Blei et al., 2003). This task is important for higher-level NLP tasks such as Semantic Search, Text Summarization and Question Answering. There is no standard number of topics, as this depends on the application domain, the desired thematic resolution and the specifics of the underlying texts. In our case of historical-exegetical tafsir studies, we determined a set of 15 sentence-based topics and span-based subtopics. Table 2 shows them along their amount of annotation data. The totals include multiple counts due to multiple annotations of the same topic. If there are lines with 0 spans and several sentences (e.g. for israiliyat), that means that only sentences have been annotated according to the 15 topics. However, no specific spans (inside the sentences) could be identified by the annotators and marked accordingly. Hence, both tasks, namely sentence-based TM and span-based TM, are displayed in Table 2, indicating that they are independent from each other.

3.3 Annotation Process

Annotation Team The annotation team consisted of 4 domain experts, who were historical linguists and orientalists by background. For NER, we let the annotators train on a smaller subset of the text (i.e., chapter 50, verse 1-22) until they reached a high inter annotators agreement (IAA) value of 97% (Cohen's kappa; (Cohen, 1960)). Thus we let them continue their annotations for the remaining volumes of text individually. For TM, we did not compute any IAA value initially, as there were only 2 domain experts available for our topics. However, we ensured a high quality of topic annotation by cross-validating and correcting them directly by the other annotator.

Tool selection & issues Selecting the right tool for our annotation task was challenging. First, CA caused many problems: It is not only a low resource-language per se; even its *right-to-left* script is low-resourced to some extent, as there are not many tools that can handle it. Second, our intention was to use the TEI standard as the target data format due to its extensive usage in Digital Humanities. Third, we required a user friendly environment as our annotators did not have any technical background. Reflecting these points, we preferred the annotation tool *Oxygen XML Editor*[2] over other candidates (such as WebAnno or BRAT). Figure 2 gives a glimpse into the annotation environment.

Data format For our final training data, we use the CoNLL format (with the BIO/IOB2 tagging scheme) and extend it for the annotation of topics and subtopics. In this adjusted 3-column format, each sentence is written vertically along its *Arabic*

[2] https://oxygenxml.com/

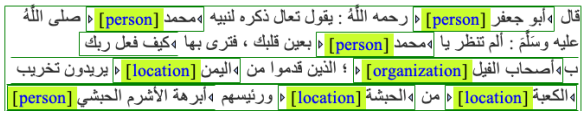

Figure 2: Screenshot of annotation working environment in *Oxygen XML Editor*.

token, *NE-tag* and *subtopic-tag*. Besides, for topics, a binary matrix structure is used at the beginning of each sentence to model all the occurrences of each *15 topics* (e.g. `# kalam: 1`, see sample excerpt in Appendix C).

After randomizing the order of the sentences, we divided the Tafsir Dataset into *train, dev, test* files according to the conventional ratio of 80:10:10 percentages. These resulting data files are used for our empirical evaluations, whose setups are described in the next section.

4 Methods

4.1 Script Compression

Arabic is a language with rich morphological variety of words. Besides, it has a distinct type of writing system (*Abjad*), which contains many layers of information developed in the course of the first centuries after the advent of Classical Arabic written tradition in the 7th century CE. The Arabic writing system is made of a basic skeleton script (*Rasm*), which 1-2 centuries later was extended to the standard Arabic script with the diacritic points (*I'jam*) to reduce the ambiguity of over 25 letters. Further 1-2 centuries later, the vocalization marks consisting of dashes and circles (*Tashkil*) were added which allowed a proper vocalized reading of theological literature.

Thus to deal with these variants, we propose the analytical setup shown in Figure 1. We use three textual variants, namely *skeleton*, *standard*, and *vocalized*, which denote the above mentioned stages the Arabic script went through during its historical development. We utilize the Python libraries *camel tools v1.3.1* (Obeid et al., 2020) and *rasmipy v0.2*[3], both applying rule-based preprocessing methods for generating our respective layers.

We hypothesize that $F_1(vocalized) < F_1(standard) \leq F_1(skeleton)$: The vocalization introduces noise, thus creating many different word embeddings of one word, which in turn lowers the overall *vocabulary coverage* of the LM for the training data. Hence, the standard/skeleton text will suite best to transformer-based neural models. Besides, current contextualized word embeddings are able to deal better with incoming textual data which has been the least preprocessed and overloaded with details (i.e. low feature engineering), which is the case for the standard/skeleton scripts. Moreover, for historical experts of the skeleton script, the ambiguity of each word decreases once longer contexts are provided, as they narrow down the possibilities of proper reading. Thus, we postulate that depending on the context, the model will be able to disambiguate the word itself and deliver an actual proper reading of the Arabic script. In Section 5, we will see that indeed our assumption has been right, and we find results which support this postulation.

4.2 Word Embeddings

We train word embeddings from scratch on large text corpora. For MSA, we take the *LeipzigArabic-2020* corpus (Goldhahn et al., 2012) with 13.55 Mio. sentences, which is already preprocessed such that it contains per line a sentence. For CA, we crawl the platform of *OpenITI* (Miller et al., 2018), containing the largest collection of online-available historical books for CA. Next, we apply our sentence splitting heuristic and tokenization from camel tools to produce a final text data file which again contains per line a sentence. With this, we get 134.17 Mio. sentences (with 17 GB of raw text data), the largest amount yet to be used for CA.

We calculate our optimized word embeddings with the extended version of the *Word2vec* algorithm (Mikolov et al., 2013), namely *Wang2vec* (Ling et al., 2015), with dimension 100, windows size 8, and min. word count 4. Although since 2019/2020 static word embeddings (which are context-independent after their training) are being replaced by their transformer-based generalization of pre-trained LMs, such as *BERT*, *XLNet*, *GPT-3* (which consider the context after their training), we still inspect the former method due to it allowing us to calculate a LM according to our chosen layer from Figure 1, and thus consider a *full analytical setup*. Furthermore, this allows us to examine how improvements can be achieved while using lightweight neural models, compared to data and computation intensive transformer-based LMs, which are on top expensive to train from scratch, and have a fixed vocabulary of subword units.

[3] https://pypi.org/project/rasmipy/

Data	Embeddings	skeleton	cov	standard	cov	vocalized	cov
ANERCorp	n-gram	n/a	n/a	55.23	Benajiba (2007)	n/a	n/a
AQMAR	SVM	n/a	n/a	69.33	Mohit (2012)	n/a	n/a
ANER	LeipzigAr	79.13	0.97	**79.14**	0.96	68.91	0.16
AQMAR	LeipzigAr	68.34	0.97	**70.93**	0.94	59.51	0.27
Tafsir	*OpenITI*	*87.13*	*0.99*	***87.41***	*0.99*	*82.97*	*0.52*

Table 3: BiLSTM results for NER on Tafsir Dataset for each layer (full setup). Coverage denotes the percentage of words from the training data that occur in the pre-trained embeddings.

4.3 Neural Models

This section provides details on the neural models which were used to examine the Tafsir Dataset along the script compression method.

4.3.1 BiLSTM

We use the neural model of BiLSTM-CRF (Lample et al., 2016; Ahmed and Mehler, 2018) with default hyperparameters for the task of CA-NER. In short, this model consists of stacked LSTM layers which receive the embedded tokens of an incoming sentence and compute a hidden representation, which in turn is used by the last CRF layer to predict the output NE-tags (i.e. PERson, LOCation, ORGanization, OTHers, O). For further details, we refer to the original papers.

4.3.2 MaChAmp

For our experiments with transformer-based LMs we use MaChAmp (van der Goot et al., 2021), a toolkit focused on multitask learning for NLP. We used v0.3 beta with default hyperparameters and compare all Arabic LMs we could find on the *Hugging Face* (Wolf et al., 2020) hub. In MaChAmp, each task has its own decoder, while the encoder (i.e. LM) is shared. We empirically saw that adding a CRF layer was beneficial (see Appendix E, Table 10), so we enabled it for NER as well as the subtopic task layer. Because the sentences can be annotated with multiple topics, we model each topic as a separate binary task. For the multi-task setups, we use an equal loss weight for all tasks, and process all tasks simultaneously.

5 Results

In this section, we present the results which are obtained while utilizing the methods and their setups described in the previous section. The evaluation of the NER predictions are performed by running the official evaluations script from the CoNLL 2003 shared task (Tjong Kim Sang and De Meulder, 2003) on the test set of the Tafsir Dataset.

5.1 BiLSTM Evaluation for CA-NER

In the single training setup, the results for our Tafsir Dataset is given which is preprocessed according to the layers outlined in Section 4.1. Most importantly, in contrast to transformer-based networks, this lightweight model allows us to not only process the training data according to our script compression method, but also the underlying LM of Word2vec (i.e. full setup). Table 3 shows the results for this setup.

First, we can see that the vocalized layer gives the lowest performance which confirms our original assumption. This performance is clearly linked to the low vocabulary coverage of this layer in respect to the pre-trained word embeddings on our selected corpora. Next, we see that the performance for standard and skeleton is relatively high. We can see that the skeleton layer continuously approaches the performance of the standard one. This behavior is stable across all three datasets and two languages (namely CA and MSA). This shows, that the skeleton layer is actually robust and almost as good as the standard one.

These results already demonstrate that our approach of script compression is noteworthy. Reducing the size of specific "redundant" letters does not lead to any significant reduction of the downstream performance. On the LM level, however, we save a relatively large amount of memory, e.g. for the Word2vec model calculated on the OpenITI corpus, we go down from 1.5 GB (standard) to 1.2 GB (skeleton) model size. Thus our first results on script compression appear to reveal a promising research direction.

MLM (standard)	skeleton	cov	standard	cov	vocalized	cov
aubmindlab/bert-base-arabertv02	85.37	0.87	**95.58**	1.00	80.26	0.85
aubmindlab/bert-large-arabertv2	85.13	0.86	**95.24**	1.00	80.14	0.84
CAMeL-Lab/bert-base-arabic-camelbert-ca	89.12	0.91	**95.43**	1.00	80.31	0.85
aubmindlab/araelectra-base-generator	84.94	0.87	**94.89**	1.00	80.06	0.85
bert-base-multilingual-cased^{+}	88.85	0.90	**95.15**	1.00	94.36	1.00
xlm-roberta-large^{+}	95.00	1.00	**95.29**	1.00	94.88	1.00
google/rembert^{+}	95.26	1.00	**95.32**	1.00	94.73	1.00

Table 4: MaChAmp results for NER on Tafsir Dataset with selected MLMs (all pre-trained on the standard layer), where for each layer (skeleton, standard, vocalized) its respective coverage (cov) is given.

5.2 MaCHAmp Evaluation

CA-NER In this section, we examine the Tafsir Dataset with various pre-trained Masked Language Models (MLM) from Hugging Face in over 119 multi-learning setups in MaChAmP. We start by utilizing all available Arabic MLMs (only pre-trained on the standard layer) and examining them along adding an optional CRF layer (see Appendix E, Table 10). Next, we cross test the Tafsir Dataset on the final selected MLMs, giving our major results in Table 4.

Although in respect to the script-dependent analysis, this is not the justified full setup, we can still get an idea what the impact of each script layer can be while fine-tuning the model. We see that the standard layer performs the best, confirming one part of our hypothesis that $F_1(vocalized) < F_1(standard)$ holds. Moreover, it is clearly demonstrated how the different layers influence the vocabulary coverage, which in turn influences the downstream performance. We can observe that in cases where $cov(vocalized) < cov(skeleton)$ holds, $F_1(vocalized) < F_1(skeleton)$ holds as well. In the opposite case, $vocalized$ is outperforming $skeleton$. Besides, we have noteworthy cases of MLMs marked with $^{+}$: For all these large multi-lingual models, their *word piece algorithm* is able to handle the vocalization by splitting it from each character, thus automatically producing the standard layer for the vocalized input. Last but not least, we can see that transformer-based models with an additional CRF layer outperform the lightweight BiLSTM thoroughly, even on the mismatched layers of $vocalized$ and $skeleton$. With this, we establish a state-of-the-art performance for CA-NER with 95.58% F-score. Thus, this comprehensive analysis allows researchers to use our dataset with the described model configurations to train a NER tagger that can confidently annotate related CA literature.

CA-TM & Multi-Task Learning In this setup, we fine-tuned the MLMs on the full Tafsir Dataset, first for each task separately, then joined within the setup of multi-task learning. Although the performance for CA-NER has been high, our results show that it is not beneficial for the task of CA-TM (see Appendix E, Table 11). However, multi-task learning is not always beneficial, as the cost of parameter sharing can become higher than the benefits of knowledge sharing. Besides, we hypothesize that TM is a very hard task on our unbalanced data which has many topics with small amount of training samples (see Table 2). A second reason that makes CA-TM a very challenging task is the fact that the topics were chosen mainly on the basis of normative considerations of a historical author: They should accompany the interpretation of religious texts in a normative way, so to speak, and are therefore of importance for the historical research of CA. TM has here the special task to reflect that the topics have been normatively pre-selected in a historical context that may not be directly available to contemporary annotators (for the purpose of generating appropriate training data). Nevertheless, these historical topic labels cannot simply be ignored, since they de facto shape research on CA.

5.2.1 Learning Curve over CA-NER Annotation Data

In order to evaluate the importance of our large-scale annotation work, we analyze the influence of the annotation data size on the final performance by plotting a learning curve over the annotation data. For each step of the size 5k sentences, we calculate the F1-score for CA-NER (on the test set) with the best observed model *bert-base-arabertv02*.

Figure 3 shows the learning curve displaying the downstream performance according to the progress of our annotation work.

Interestingly, we can see that the annotators' work has been worth it. The curve is quite steep, i.e. with every additional generation of annotation data we increased the performance steadily for our target task of CA-NER until 30k sentences. After that, the gradient starts to decrease at which the curve begins to slowly approach the max performance value of 95.58% F-score. Thus, we conclude that large amount of gold data is indeed beneficial for CA-NER, which contrasts previous findings for other low-resource languages such as Danish (Plank et al., 2020).

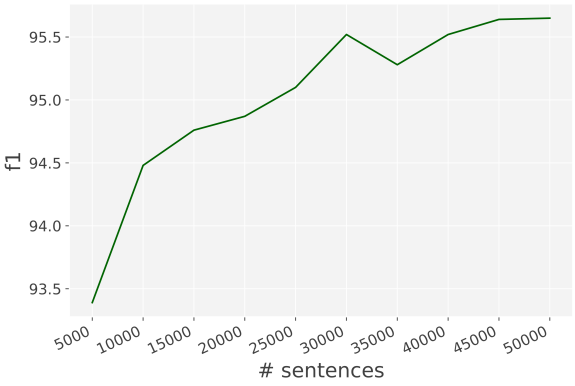

Figure 3: Learning curve over annotation data for NER (standard layer) in steps of 5k sentences.

5.3 Error Analysis for CA-NER

Our manual error analysis on the test set has revealed that the following errors exist: A majority of (1) prediction errors, where the model does not tag those NEs which are annotated by the annotators, and a minority of (2) annotation errors, where the model tags those NEs which are falsely not annotated by the annotators. However, most of the annotation errors were found in the false positives.

The Arabic language contains various words with polysemy (i.e. one word has many meanings). Especially if a word is not vocalized, and the sentence context is small, it can become difficult for the common reader to understand the underlying meaning. Then, only a domain expert can provide the precise meaning. For prediction errors, our manual error analysis has shown that the model is mistaken exactly in such cases, where there is a NE in very short sentences (e.g. 2-word *nominal sentences*). We hypothesize this is because the model has only access to one sentence, whereas the domain expert annotators have more advantages by knowing the full context via their chapter-wise view.

6 Conclusion

In this work, we presented the Tafsir Dataset, the first large-scale multi-task benchmark on NER and TM in Classical Arabic literature. We demonstrated how useful resources can be for languages which have been historically important but now forgotten by the ongoing NLP research. Besides, we also performed a first evaluation of this newly generated dataset. While doing so, we empirically saw that adding a CRF layer was beneficial to the transformer-based models, with which we ultimately established a state-of-the-art performance for CA-NER. Although TM was not the primary focus of this paper, we generated first results for CA-TM, thereby leaving room for future improvements. This refers to a scenario of TM in which topic labels were originally determined in a historical, normative, exegetical setting, whereas they need to be learned using modern NLP tools, based on their relevance to CA research. Such scenarios are likely to be increasingly encountered as more historical languages come into NLP focus. We therefore believe that our benchmark induces a new challenge for the NLP community that can lead to progress for our target low-resource language.

The Tafsir Dataset and its accompanying material are made open-source available for the research community. Furthermore, a website[4] is published which offers a comprehensive research tool in English and Arabic for accessing our dataset in a more user-friendly environment and performing various search queries on it. The web-based tool is freely available and provides over 400 filter options along the categories of our dataset. Additionally, it provides the option of graphical visualization (bubble or pie chart) of the dataset and of the query results performed on it. This digital tool makes it possible for scholars from historical and theological fields to access the dataset without any prior technical skill sets, thus allowing them to find systematically the answers to their long-lasting research questions.

On a side note, by analyzing the historical skeleton script, we shed light on a centuries-old historical critical question regarding the readability of the Rasm text: Whether the first Quranic manuscripts

[4] https://linkedopentafsir.de/

(i.e. *Uthmanic codex*) can provide a precise reading of the canonized oral text, or whether there is a large amount of ambiguity in it. Our script-dependent analysis shows that from an information retrieval perspective, the usage of the skeleton script is robust enough to deliver a similar performance compared to the usage of the standard script. We can thus conclude that if the ML model is able to deal with the skeleton script, then humans will also not face major difficulties after gaining sufficient training on the same ancient script.

Future work Our work gives indications that script compression seems to be a promising direction to reduce the amount of data and tackle the question of which resource-size actually matters (Ahmed and Mehler, 2018). In this work, for the case of Arabic we came down from 28 to 16 letters while keeping the performance stable. This shows that we do not need (1) vowels, and (2) different letters for each phoneme. In fact, just some minimum amount of *consonantal distinction* is needed. What is this amount, can we determine it exactly for each target language? Phonetic algorithms such as *Metaphone* (Philips, 1990) pose to be a first language-independent approach, be that as it may, only future work can give us the answers.

Acknowledgments

This interdisciplinary research work has been conducted as part of the project *Linked Open Tafsir*, which is funded by the *AIWG*[5], in turn supported by the *BMBF* and *Stiftung Mercator*. We acknowledge the IT University of Copenhagen HPC for the resources made available for conducting the research reported in this paper. Special thanks go to U. Brucker and the International Office (Goethe University Frankfurt) for enabling the collaboration with the IT University of Copenhagen, Prof. B. Plank and further colleagues from the IT University of Copenhagen for their critical feedback on the experimental evaluations, Prof. U. Meyer and GRADE (Goethe University Frankfurt) for funding the final publication process, and Prof. G. Hirst (University of Toronto) along Dr. J. Kindermann (Fraunhofer Institute IAIS) and Prof. R. V. Zicari (Goethe University Frankfurt) for their closing remarks on the manuscript.

[5] https://aiwg.de/

References

Sajawel Ahmed and Alexander Mehler. 2018. Resource-Size matters: Improving Neural Named Entity Recognition with Optimized Large Corpora. In *Proceedings of the 17th IEEE International Conference on Machine Learning and Applications (ICMLA)*, Florida, Orlando, USA.

Sajawel Ahmed, Misbahur Rehman, Joshua Tischlik, Carl Kruse, Edin Mahmutovic, and Ömer Özsoy. 2022. Linked Open Tafsir—Rekonstruktion der Entstehungsdynamik (en) des Korans mithilfe der Netzwerkmodellierung früher islamischer Überlieferungen. In *8. Jahrestagung des Verbandes Digital Humanities im deutschsprachigen Raum (DHd)*.

Sajawel Ahmed, Manuel Stoeckel, Christine Driller, Adrian Pachzelt, and Alexander Mehler. 2019. BIOfid dataset: Publishing a German gold standard for named entity recognition in historical biodiversity literature. In *Proceedings of the 23rd Conference on Computational Natural Language Learning (CoNLL)*, pages 871–880, Hong Kong, China. Association for Computational Linguistics.

Jalal Al-Din Al-Suyuti. 1505. *Al-itqan Fi 'ulum Al-Qur'an (The Perfect Guide to the Sciences of the Qu'ran)*. Garnet Publishing; Bilingual edition (May 1, 2012).

Amin Almuhanna and Jean-Francois Prunet. 2019. From Classical to Modern Arab Names and Back. *Anthropological Linguistics*, 61(4):405–458. Copyright - Copyright University of Nebraska Press Winter 2019; Last updated - 2021-10-04; SubjectsTermNotLitGenreText - Arabian Peninsula; Kuwait.

Yassine Benajiba, Paolo Rosso, and José Miguel Benedíruiz. 2007. Anersys: An arabic named entity recognition system based on maximum entropy. In *International Conference on Intelligent Text Processing and Computational Linguistics*, pages 143–153. Springer.

Emily M Bender and Batya Friedman. 2018. Data statements for natural language processing: Toward mitigating system bias and enabling better science. *Transactions of the Association for Computational Linguistics*, 6:587–604.

David M Blei, Andrew Y Ng, and Michael I Jordan. 2003. Latent dirichlet allocation. *Journal of machine Learning research*, 3(Jan):993–1022.

Armin Burkhardt. 2004. 2004. Nomen est omen? : zur Semantik der Eigennamen. In *Landesheimatbund Sachsen-Anhalt e. V. (Hrsg.): "Magdeburger Namenlandschaft" : Orts- und Personennamen der Stadt und Region Magdeburg*.

B Barla Cambazoglu, Mark Sanderson, Falk Scholer, and Bruce Croft. 2021. A review of public datasets in question answering research. In *ACM SIGIR Forum*, volume 54, pages 1–23. ACM New York, NY, USA.

Jacob Cohen. 1960. A coefficient of agreement for nominal scales. *Educational and psychological measurement*, 20(1):37–46.

Daria Dzendzik, Jennifer Foster, and Carl Vogel. 2021. English machine reading comprehension datasets: A survey. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 8784–8804, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Anoual El Kah and Imad Zeroual. 2021. Arabic topic identification: A decade scoping review. In *E3S Web of Conferences*, volume 297, page 01058. EDP Sciences.

Dirk Goldhahn, Thomas Eckart, and Uwe Quasthoff. 2012. Building large monolingual dictionaries at the leipzig corpora collection: From 100 to 200 languages. In *LREC*.

Guillaume Lample, Miguel Ballesteros, Sandeep Subramanian, Kazuya Kawakami, and Chris Dyer. 2016. Neural Architectures for Named Entity Recognition. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 260–270. Association for Computational Linguistics.

Wang Ling, Chris Dyer, Alan Black, and Isabel Trancoso. 2015. Two/Too Simple Adaptations of word2vec for Syntax Problems. In *Proceedings of the 2015 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*. Association for Computational Linguistics.

Hajer Maraoui, Kais Haddar, and Laurent Romary. 2017. Encoding prototype of al-hadith al-shareef in tei. In *International Conference on Arabic Language Processing*, pages 217–229. Springer.

Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. 2013. Distributed representations of words and phrases and their compositionality. In *Advances in neural information processing systems*, pages 3111–3119.

Matthew Thomas Miller, Maxim G. Romanov, and Sarah Bowen Savant. 2018. Digitizing the textual heritage of the premodern islamicate world: Principles and plans. *International Journal of Middle East Studies*, 50(1):103–109.

Behrang Mohit, Nathan Schneider, Rishav Bhowmick, Kemal Oflazer, and Noah A Smith. 2012. Recall-oriented learning of named entities in arabic wikipedia. In *Proceedings of the 13th Conference of the European Chapter of the Association for Computational Linguistics*, pages 162–173.

Ossama Obeid, Nasser Zalmout, Salam Khalifa, Dima Taji, Mai Oudah, Bashar Alhafni, Go Inoue, Fadhl Eryani, Alexander Erdmann, and Nizar Habash. 2020. CAMeL tools: An open source python toolkit for Arabic natural language processing. In *Proceedings of the 12th Language Resources and Evaluation Conference*, pages 7022–7032, Marseille, France. European Language Resources Association.

Lawrence Philips. 1990. Hanging on the metaphone. In *Computer Language*, volume 7, pages 39–43.

Barbara Plank, Kristian Nørgaard Jensen, and Rob van der Goot. 2020. DaN+: Danish nested named entities and lexical normalization. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 6649–6662, Barcelona, Spain (Online). International Committee on Computational Linguistics.

Pranav Rajpurkar, Robin Jia, and Percy Liang. 2018. Know what you don't know: Unanswerable questions for SQuAD. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 784–789, Melbourne, Australia. Association for Computational Linguistics.

Stefan Schweter and Sajawel Ahmed. 2019. Deep-EOS: General-Purpose Neural Networks for Sentence Boundary Detection. In *Proceedings of the 15th Conference on Natural Language Processing (KONVENS)*.

Erik F. Tjong Kim Sang and Fien De Meulder. 2003. Introduction to the CoNLL-2003 shared task: Language-independent named entity recognition. In *Proceedings of the Seventh Conference on Natural Language Learning at HLT-NAACL 2003*, pages 142–147.

Rob van der Goot, Ahmet Üstün, Alan Ramponi, Ibrahim Sharaf, and Barbara Plank. 2021. Massive choice, ample tasks (MaChAmp): A toolkit for multi-task learning in NLP. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: System Demonstrations*, pages 176–197, Online. Association for Computational Linguistics.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Remi Louf, Morgan Funtowicz, Joe Davison, Sam Shleifer, Patrick von Platen, Clara Ma, Yacine Jernite, Julien Plu, Canwen Xu, Teven Le Scao, Sylvain Gugger, Mariama Drame, Quentin Lhoest, and Alexander Rush. 2020. Transformers: State-of-the-art natural language processing. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 38–45, Online. Association for Computational Linguistics.

A Examples for annotating named entities in Tafsir Al-Tabari books

1) ذُكر أن [المشركين]ORG سألوا رسول الله صلى الله عليه وسلم عن نسب ربّ العزّة, فأنـزل الله هذه السورة جوابا لهم. (112:1)
2) حدثنا [أبو كُرَيب]PER , قال: ثنا [ابن إدريس]PER , عن [عبد الملك]PER , عن [طلحة]PER , عن [مجاهد]PER , مثله. (112:4)
3) يقول تعالى ذكره لنبيه [محمد]PER صلى الله عليه وسلم : إذا جاءك نصر الله يا [محمد]PER على قومك من [قريش]ORG , والفتح: فتح [مكة]LOC (110:2)
4) قال: ثنا [ابن ثور]PER ، عن [معمر]PER ، عن [ابن طاوس]PER ، عن أبيه، قال: ما من شيء أقرب إلى الشرك من رَقّية المجانين . (113:4)
5) حدثني [يعقوب]PER ، قال: ثنا [هشيم]PER ، قال: أخبرنا [العوّام بن عبد الجبار الجولاني]PER ، قال: قَدم رجل من [أصحاب رسول الله]ORG صلى الله عليه وسلم [الشأم]LOC ، قال: فنظر إلى دور [أهل الذمة]ORG ، وما هم فيه من العيش والنضارة، وما وُسّع عليهم في دنياهم، قال: فقال: لا أبا لك أليس من ورائهم الفلق؟ قال: قيل: وما الفلق؟ قال: بيت في [جهنم]LOC إذ فُتح هَرَ [أهْلُ النار]ORG . (113:1)
6) يقول تعالى ذكره لنبيه [محمد]PER صلى الله عليه وسلم: ألم تنظر يا [محمد]PER بعين قلبك, فترى بها كَيْفَ فَعَلَ رَبُّكَ بِ [أَصْحَابِ الْفِيلِ]ORG الذين قَدِموا من [اليمن]LOC يريدون تخريب [الكعبة]LOC من [الحبشة]LOC ورئيسهم [أبرهة الحبشيّ الأشرم]PER (105:1)
7) حدثنا [ابن المثنى]PER ، قال: ثني [عبد الأعلى]PER ، قال: ثنا [داود]PER ، عن [عكرِمة]PER ، عن [ابن عباس]PER ، قال: نـزل [القرآن]OTH كله مرة واحدة في [ليلة القدر]TME في [رمضان]TME إلى السماء الدنيا، فكان الله إذا أراد أن يحدث في [الأرض]LOC شيئًا أنـزله منه حتى جمعه. (97:1)

Figure 4: Examples for annotating named entities (i.e. PER, LOC, ORG, TME, OTH) in 7 verses from the raw text of Tafsir Al-Tabari books.

B Annotation Guidelines (German version)

Guidelines für die Named Entity Recognition. Sie bauen auf den arabisierten Guidelines von Ahmed et al. (2019) auf.

B.1 Einführung: Named Entity Recognition

Unter der Named Entity Recognition (NER) versteht man die Aufgabe, Eigennamen (named entities) in Texten zu erkennen. Technisch gesehen sind hierzu zwei Schritte notwendig. Zuerst müssen in einem laufenden Text die Token gefunden werden, die zu einem Eigennamen gehören (Named Entity Detection: NED), danach können diese Eigennamen semantischen Kategorien zugeordnet werden (Named Entity Classification). Prototypisch ist dabei der Unterschied zwischen Eigennamen und Appellativa der, dass letztere eine Gattung oder eine Klasse beschreiben, während erstere einzelne Individuen oder Sammlungen von Individuen unabhängig von gemeinsamen Eigenschaften bezeichnen (Burkhardt, 2004). Die vorliegenden Guidelines sollen es Annotatoren ermöglichen, Eigennamen in Texte aus Standard und Nichtstandard-Varietäten konsistent zu annotieren. In diesen Guidelines werden die beiden Aufgaben der NED und NEC nicht unterschieden, da die Konzentration auf Beispiele in diesem Dokument, die Trennung künstlich erzeugen müsste und nicht zu erwarten ist, dass die Resultate sich dadurch verbessern würden. In Anlehnung an die oben genannten Guidelines für Zeitungssprache werden in NoSta-D-Tafsir fünf semantische Hauptklassen für klassische arabische Texte unterschieden (Personen, Organisationen, Orte, Zeiten und Andere).

B.2 Wie finde ich eine NE?

Schritt 1: Nur volle Nominalphrasen können NEs sein. Pronomen und alle anderen Phrasen können ignoriert werden.
Schritt 2: Namen sind im Prinzip Bezeichnungen für einzigartige Einheiten, die nicht über gemeinsame Eigenschaften beschrieben werden.
Beispiel:
[Der Struppi] folgt [seinem Herrchen].
Hier gibt es zwei Nominalphrasen als Kandidaten für einen Eigennamen (NE). “Der Struppi” bezeichnet eine einzige Einheit. Es kann auch mehrere Struppis geben, aber diese haben an sich keine gemeinsamen Eigenschaften, bis auf den gemeinsamen Namen, daher handelt es sich um einen Eigennamen. *"seinem Herrchen"* bezeichnet zwar (typischerweise) auch nur eine einzige Person allerdings können wir diese nur über die Eigenschaft identifizieren, dass sie ein Herrchen ist und dass dies für Struppi zutrifft. Struppi könnte auch mehrere Herrchen haben, die alle die Eigenschaften teilen, die ein Struppi-Herrchen beinhaltet (z.B. darf Struppi streicheln, muss ihn ausführen und füttern etc.)
Schritt 3: Determinierer sind keine Teile des Namens.
Beispiel: *Der [Struppi]NE folgt seinem Herrchen.*
Schritt 4: Eigennamen können mehr als ein Token beinhalten. Beispiel:
Viele Personennamen (PER für person):
[Abu Jafar Muhammad Ibn Jarir Al Tabari]PER
Buchtitle (OTH für other):
[Jami Al Bayan Al Tawil Ay Al Quran]OTH
Schritt 5: Eigennamen können auch in einander verschachtelt sein. Beispiel:

Personennamen in Buchtiteln:
[Sunan [Abi Dawud]PER]OTH
Orte (LOC für location) in Vereinsnamen (ORG für organisation):
[Hebarium Senckenbergianum [Frankfurt]LOC]ORG
Schritt 6: Titel, Anreden und Besitzer gehören NICHT zu einem komplexen Eigennamen. Besitzer können natürlich selber Eigennamen sein. Beispiel:
Referenz auf Musiktitel:
[Vivaldis]PER [Vier Jahreszeiten]OTH
Referenz auf Personen:
Landesvorsitzende Frau Vorstandsvorsitzende Dr. [Ute Wedemeier]PER
Schritt 7: Wenn das Gesamttoken einen Eigennamen darstellt, dann wird dieser annotiert. Beispiel:
Stiftungen: *[[Böll]PER-Stiftung]ORG*
Schritt 8: Kann in einem Kontext nicht entschieden werden, ob eine NP sich als Eigennamen oder Appellativ verhält, wird es nicht als NE markiert. Beispiel:
Ortsnamen vs. -beschreibungen:
...und zogen mit ihren grossen Transparenten gestern vom [Steintor] über den [Ostertorsteinweg]LOC zum [Marktplatz].
Schritt 9: Wenn ein Name als Bezeichnung für bestimmte Gegenstände in die Sprache übergegangen ist und in seiner Nutzung nicht als NE fungiert, so wird dieser nicht annotiert. Beispiel:
[Teddybär] (NICHT PER)
[Colt] (NICHT PER)
Schritt 10: Bei Aufzählungen mit Hilfe von Bindestrichen oder Vertragen eines Teils der NE auf spätere Wörter, wird die NE so annotiert, als sei sie voll ausgeschrieben.
Beispiel:
[Frühe]OTH und [Späte Bronzezeit]OTH
[Süd-]LOC und [Nordafrika]LOC

B.3 Zu welcher semantischen Klasse gehört ein Eigenname?

Wenn der Eigenname in eine der Klassen in der Liste Faustregel zur Unterscheidung einer Klassenbezeichnung und eines Namens gehört, dann annotiere die zugehörige Klasse. Sollte die gefundene NE Rechtschreibfehler enthalten, wird sie dennoch annotiert. In Zweifelsfällen hilft auch die Tabelle NoSta-D-Tafsir-TagSet und alle Untertabellen, insbesondere die Beispiele mit dem weiter.
Jahreszahlen in ORGanisationen werden markiert.
Beispiel:
[COLING]ORG [2022]TIME
[Fussball-WM]ORG [2014]TIME
Wenn der Eigennamen in KEINE der vorhandenen Klassen passt, markiere diesen mit ***UNCLEAR***, notiere dir bitte das Beispiel und schicke uns eine E-Mail an: `X.Y@email.com`. So können wir die Guidelines sukzessiv verbessern.

B.4 Faustregel zur Unterscheidung einer Klassenbezeichnung und eines Namens:

- Elemente der fraglichen Einheit verbinden die gleichen Eigenschaften → Klasse → keine NE
- Die Elemente der fraglichen Einheit verbindet nur der Name oder Element ist Einheit bezeichnet ein spezifisches Individuum → Name → NE
- *"Paleocene"* bezeichnet spezifische Epoche → NE (TME)

NoSta-D-Tafsir-Tagset

Table 5: Kategorie 'PER-Person'

Subkategorie	Beispiele
Person	*Ibn Ahmed, Saria, Al Tabari*
Künstlernamen	*Abu Nuwas*
Charaktere	*Ali Baba*
Superhelden	*Aladin, Sindbad*

Table 6: Kategorie 'LOC-Ort'

Subkategorie	**Beispiele**
Bezirke	*Makkah Aziziyah, Schöneberg*
Sehenswürdigkeiten, Moscheen	*Mada'in Saleh, Al Masjid Al Haram*
Planeten	*Erde, Mars*
Landschafts-bezeichnungen	*Al Nefud, Königsheide*
Straßen, Plätze	*Al Tariq Al Maliki Al Farsi*
Einkaufszentren	*Suq Ukadh, Nordwestzentrum*
Berge, Seen, Flüsse	*Jabal Arafat, Al Bahr Al Ahmar, Wadi Hanifa*
Kontinente	*Asien, Europa*
Länder, Staaten	*Saudi-Arabien, Hessen, Iran*
Städte	*Mekka, Babylon*
Regionen	*Al Hijaz*
Qiraat-Orte	*Al Amsar*

Table 7: Kategorie 'ORG-Organisation'

Subkategorie	**Beispiele**
Organisationen	*Ahl Al Hadith, Sunni, Shia, Ismailiten, GEFIS, EU, Landgericht Frankfurt*
Religionsgruppen	*Juden, Christen, Muslime*
Unternehmen	*Karimis, Microsoft*
Sammelbezeichung	*Umran*
Madhahib	*Kufiyun*
Qabilah	*Quraish*
Volksgruppen	*Araber, Perser, Römer*
Universitäten	*Al-Azhar University*
Bibliotheken	*Bayt Al Hikmah*

Table 8: Kategorie 'TIME'

Subkategorie	**Beispiele**
Tag	*Freitag*
Monat	*Rabi' Al Awwal*
Jahr	*570*
dd.mm.yyyy	*12.03.0570*
Jahrhundert	*5. Jahrhundert*
Epochen	*Jahiliyyah, Paleocene*

Table 9: Kategorie 'OTH-Andere'

Subkategorie	**Beispiele**
Buch-, Filmtitel etc.	*Sahih Al Bukhari, Faust*
Währungen	*Dinar, Dirham, Euro*
Sprachen	*Arabisch, Deutsch, Latein*
Buchtitel mittels Autor	*Helbig et al., ([[Helbig]PER et al.]OTH)*
Gottheiten	*Al Uzza, Al Lat, Manat, Ba'al, Nasr, Suwa', Wadd, Yaghuth*
Engel	*Jibril, Mikail, Israfil*
Dschinn	*Iblis*
Mythol. Tiere	*Hudhud*

C Sample Excerpt from Tafsir Dataset

```
# adyan: 0
# asbab: 0
# fiqh: 0
# kalam: 1
# lugha: 1
# mushkilat: 0
# mutashabih: 0
# naskh: 0
# qiraat: 0
# science: 0
# sirah: 0
# sufism: 0
# takhsis: 0
# tikrar: 0
وَقَوْلُهُ    O     O
:      O     O
وُجُوهٌ    O     O
يَوْمَئِذٍ    O     O
نَاضِرَةٌ    O     O
.      O     O
يَقُولُ    O     O
تَعَالَى    O     O
ذكره    O     O
:      O     O
وُجُوهٌ    O     B-lugha
يَوْمَئِذٍ    O     O
.      O     O
يَعْنِي    O     O
:      O     O
يَوْمَ    B-TME O
الْقِيَامَةِ    I-TME O
،      O     O
نَاضِرَةٌ    O     B-lugha
.      O     O
```

Figure 5: Tafsir Dataset in CoNLL format, showing the binary topic matrix before the sentence start, afterwards the Arabic tokens along their NER tag (1st column) and subtopic tag (2st column).

D Data Statement

In accordance with (Bender and Friedman, 2018), the following outlines the data statement for the Tafsir Dataset:

A. CURATION RATIONALE Manual annotation of literature in Classical Arabic, which is to date a low-resource language, for identification of named entities in different historical text domains, complemented with topic modeling annotation. The generation of such training data enables machine learning applications for the research fields of historical NLP and digital humanities.

B. LANGUAGE VARIETY The canonical text data of *Tafsir Al-Tabari* was collected from the online resource platform *Gawami' al-Kalim* (https://gk.islamweb.net).

C. SPEAKER DEMOGRAPHIC For various text samples in the historical collections of narrations, it is Classical Arabic speakers. Gender, age, race-ethnicity, socioeconomic status can be inferred from the extensive classical literature of biographical evaluation (*'Ilm Al-Rijal*) on narrators and their biographies (books such as *Al-Tarikh Al-Kabir* ("The Great History") by Imam Bukhari, *Kitab Al-Tabaqat Al-Kabir* ("The Book of the Major Classes") by Ibn Sa'd, or *Ikhtiyar Ma Rifat Al-Rijal* ("The Selection of the Knowledge of the Men") by Shaykh Tusi).

D. ANNOTATOR DEMOGRAPHIC Four scientific staff members and two students (age range: 25-60), gender: male and female. European with Middle Eastern background. Native language: German, Modern Standard Arabic, Classical Arabic. Socioeconomic status: university faculty and higher-education student in Classical Arabic studies.

E. SPEECH SITUATION Sopken Classical Arabic, which was later edited by the collector (here: Al-Tabari). Time frame of data between 7th century and 923 CE. Place: Middle East.

F. TEXT CHARACTERISTICS Exegetical literature: Sentences made of chain of narrators (*Isnad*) and the actual content of narrations (*Matn*) along exegetical prose elaborations for each verse of the Quran.

PROVENANCE APPENDIX N/A

E Extended Results

MLM	SEQ	CRF	Coverage
aubmindlab/bert-base-arabert	79.34	79.91	0.74
aubmindlab/bert-base-arabertv01	79.49	80.07	0.65
aubmindlab/bert-base-arabertv02	79.81	80.26	0.85
aubmindlab/bert-base-arabertv2	79.43	80.14	0.84
aubmindlab/bert-large-arabertv2	79.18	80.29	0.84
asafaya/bert-base-arabic	94.99	95.31	1.00
asafaya/bert-mini-arabic	94.02	94.50	1.00
asafaya/bert-large-arabic	94.90	94.92	1.00
asafaya/bert-medium-arabic	94.93	94.87	1.00
CAMeL-Lab/bert-base-arabic-camelbert-ca	79.56	80.31	0.85
CAMeL-Lab/bert-base-arabic-camelbert-mix	79.61	80.19	0.85
CAMeL-Lab/bert-base-arabic-camelbert-msa	79.40	80.23	0.85
UBC-NLP/ARBERT	95.04	95.29	0.88
UBC-NLP/MARBERT	94.83	94.92	0.88
aubmindlab/araelectra-base-generator	79.37	80.06	0.85
bert-base-multilingual-cased	93.89	94.36	1.00
xlm-roberta-base	94.13	94.49	1.00
xlm-roberta-large	94.36	94.88	1.00
google/rembert	94.43	94.73	1.00

Table 10: Results for CA-NER w/ and w/o CRF

MLM	NER		Topic		Subtopic	
	st	mt	st	mt	st	mt
aubmindlab/bert-base-arabertv02	95.99	95.87	26.11	13.73	21.18	20.47
aubmindlab/bert-large-arabertv2	95.53	95.26	18.94	14.43	18.28	19.44
asafaya/bert-base-arabic	95.61	94.94	20.63	11.84	19.23	18.36
asafaya/bert-large-arabic	95.65	95.80	22.15	20.46	21.68	20.58
asafaya/bert-medium-arabic	95.13	95.17	20.15	9.46	18.67	17.45
CAMeL-Lab/bert-base-arabic-camelbert-ca	96.06	95.99	24.75	15.81	19.68	17.42
UBC-NLP/ARBERT	95.46	95.45	22.16	20.37	22.05	20.56
aubmindlab/araelectra-base-generator	95.08	94.95	18.92	6.86	14.85	14.65
bert-base-multilingual-cased	95.04	94.79	23.11	11.58	18.54	16.72
xlm-roberta-large	95.54	95.22	16.97	13.80	21.18	20.46

Table 11: Multi-task learning results for each task. st=single task, mt=multitask

Topic	Macro-F1
adyan (non-Islamic religion)	27.93
asbab (occasions of revelation)	22.74
fiqh (jurisprudence)	16.66
israliyat (Judeo-Christian)	23.17
kalam (Islamic theology)	26.61
lugha (linguistics)	30.06
mushkilat (problem)	19.97
mutashabih (allegorical)	20.00
naskh (abrogation)	19.76
qiraat (recitation style)	*41.45*
sirah (biography)	21.96
sufism (mysticism)	14.87
takhsis (specification)	19.99
tikrar (repetition)	19.98
ulum (science)	18.41

Table 12: Fine-grained TM results obtained with the measure of Macro-F1 from MaChAmp on Tafsir Dataset (arabertv02).

NE category	Precision	Recall	F1
PER	*97.12*	*97.60*	*97.36*
LOC	72.53	66.93	69.62
ORG	82.00	89.31	85.50
TME	78.00	79.90	78.94
OTH	79.59	76.38	77.95

Table 13: Fine-grained NER results obtained by running the official CoNLL-2003 script on Tafsir Dataset (arabertv02).

Resource of Wikipedias in 31 Languages Categorized into Fine-Grained Named Entities

Satoshi Sekine*, Kouta Nakayama*, Masako Nomoto*
Maya Ando+, Asuka Sumida*, Koji Matsuda*
***RIKEN, AIP, +free**
1-4-1 Nihonbashi,Chuo-ku, Tokyo, 15th floor 103-0027, Japan
satoshi.sekine@riken.jp

Abstract

This paper describes a resource of Wikipedias in 31 languages categorized into Extended Named Entity (ENE), which has 219 fine-grained NE categories. We first categorized 920K Japanese Wikipedia pages according to the ENE scheme using machine learning, followed by manual validation. We then organized a shared task of Wikipedia categorization into 30 languages. The training data were provided by Japanese categorization and the language links, and the task was to categorize the Wikipedia pages into 30 languages, with no language links from Japanese Wikipedia (20M pages in total). Thirteen groups with 24 systems participated in the 2020 and 2021 tasks, sharing their outputs for resource-building. The Japanese categorization accuracy was 98.5%, and the best performance among the 30 languages ranges from 80 to 93 in F-measure. Using ensemble learning, we created outputs with an average F-measure of 86.8, which is 1.7 better than the best single systems. The total size of the resource is 32.5M pages, including the training data. We call this resource creation scheme "Resource by Collaborative Contribution (RbCC)". We also constructed structuring tasks (attribute extraction and link prediction) using RbCC under our ongoing project, "SHINRA".

1 Introduction

Wikipedia consists of a large volume of entities that are significant resources for the knowledge base (KB) used in many Natural Language Processing (NLP) applications, including Question Answering, Information Extraction, and so on. To maximize the use of such a KB, the information in Wikipedia has to be categorized and structured in a consistent manner for machines to perform inference, reasoning, and many other purposes. The current categorization and structure of Wikipedia and other KB derived from it, such as DBpedia, YAGO, and Wikidata, are extremely noisy for NLP applications. This noise is inherent to Wikipedia categories and structures because they are created by multiple independent crowd workers using a bottom-up approach. There are no consistent rules exists for most parts of Wikipedia. As a result, instead of the cumbersome Wikipedia categories and structures, we must rely on a well-defined ontology. Extended Named Entity (ENE) (ENE homepage) is one such ontology for named entities (NEs). ENE version 8 has 219 hierarchical categories, and a set of attributes is defined for each category. Our final goal is to transform Wikipedia information into the structure of ENE so that machines can use the rich information in Wikipedia.

This study reports a resource of Wikipedias in 31 languages categorized into fine-grained named entity (ENE) categories. Categorization is the first task in creating a structured KB. After developing the techniques of automatic categorization are developed, we can structure the contents of Wikipedia pages in each category. We worked on 31 languages, rather than a few major languages. We created resources for practical NLP applications in various languages, instead of platforming a few major languages.

2 Extended Named Entity

To construct a useful KB for NLP applications, a well-structured ontology is essential and must be designed in a top-down manner. The structures of the KBs in DBpedia, Freebase, and Wikidata were created by crowds in a bottom-up manner, all with similar characteristics, including inconsistent categories, imbalanced ontologies, and ad hoc attributes. This is because of the bottom-up nature of their KB designs. We need a top-down strategy to consistently design the ontology and attributes.

Proceedings of the 29th International Conference on Computational Linguistics, pages 3769–3777
October 12–17, 2022.

For a top-down designed ontology for named entities, we employed the Extended Named Entity, ENE (ENE-Homepage). ENE is a cleanly-designed named entity classification hierarchy that includes the attribute definition for each category (Sekine et al., 2002; Sekine and Nobata, 2004; Sekine, 2008). It includes 219 fine-grained categories of named entities in a hierarchy of up to four layers.

It contains not only the fine-grained categories of the typical NE categories, such as "city" and "lake" for "location", and "company" and "political party" for "organization", but also new named entity types such as "products", "event", and "natural object". These categories cover various entities that are often mentioned in encyclopedias and many other resources. Figure 1 shows ENE version 8.0. The category "Concept" is used for anything that doesn't fit into ENE categories; typically common nouns. "IGNORE" is for Wikipedia-specific titles such as "redirect", "disambiguation", and "meta information". Attributes were also designed for each category based on the investigation of the sample entities. For example, the attributes for the "airport" category include the following: "Reading", "IATA code", "ICAO code", "nickname", "name origin", "number of users per year", "number of runaways", and so on. Please refer to the ENE homepage for the complete definitions.

3 Categorized Japanese Wikipedia

We categorized 920K Japanese Wikipedia pages into one or more of 219 ENE categories. For the categorization process, we excluded less popular entities, that is, those having fewer than five incoming links (approximately 151K entities) and nonentity pages (approximately 53K pages), such as common nouns and simple numbers (CONCEPT), and Wikipeida's meta-information pages and forward pages (IGNORED). This categorization was done using the machine learning method with a training data manually created (Suzuki et al., 2018) followed by manual validation. The accuracy of the categorization was confirmed as 98.5% by senior annotators on 1,000 sample data. The remaining 1.5% covered ambiguous and difficult ones, even for human annotators. Table 1 shows the number of entities in each category on Japanese Wikipedia pages. Note that a Wikipedia page can have more than one category. For example, a novel adapted into a movie can have both categories if mentioned in the major paragraphs on the page. The detailed definition can be found on the SHINRA homepage (SHINRA HP).

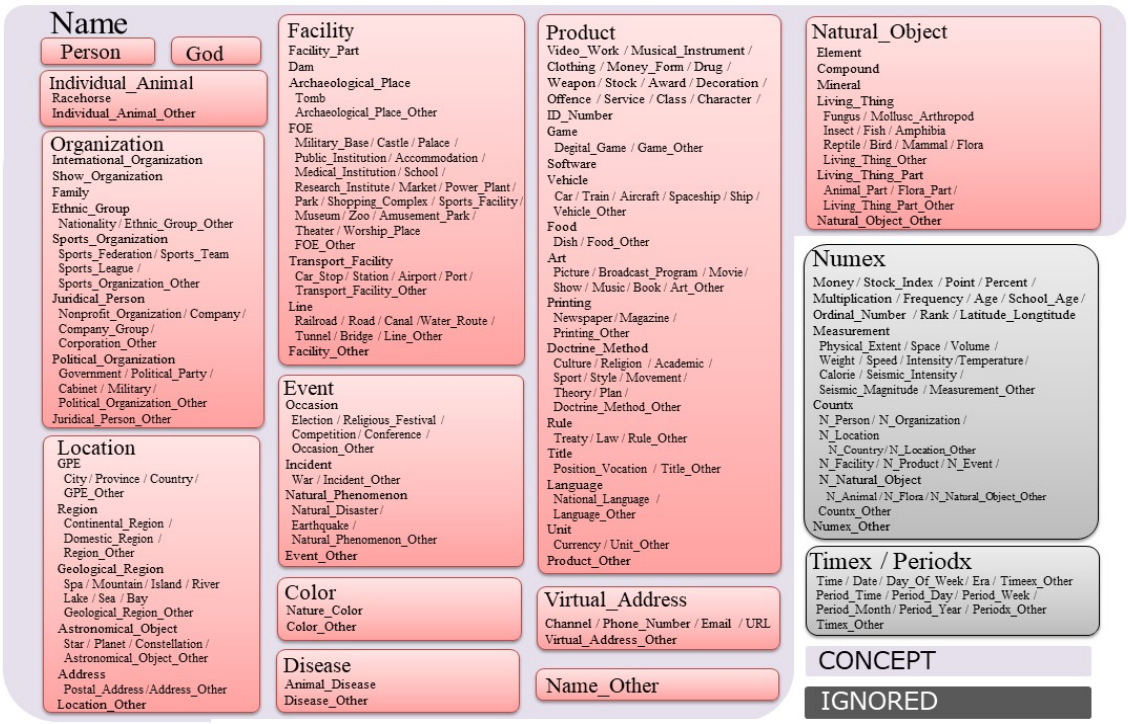

Figure 1. Extended named entity hierarchy version 8

4 Shared Tasks

We conducted Wikipedia page categorization in 30 languages shared-tasks in 2020 and 2021. We refer to them as SHINRA2020-ML and SHINRA2021-ML, respectively. We tackled the problem of categorizing Wikipedia entities in 30 languages in fine-grained categories of 219 categories, as defined in ENE (ver. 8.0). We selected 30 languages based on the number of active users from Wikipedia statistics. The languages are listed in Table 2. The Wikipedias are based on dumps in January 2019. The participants can select one or more target languages, and for each language, their automatic categorization system must categorize all Wikipedia pages in the target language(s), as the evaluation data are hidden. We provided training data for 30 languages, created by (i) the categorized Japanese Wikipedia of 920 K pages, and (ii) Wikipedia language links for 30 languages from the Japanese Wikipedia. For example, out of 2,263K German Wikipedia pages, 275K pages had language links from Japanese Wikipedia, which served as silver (i.e., a bit noisy) training data for German. Thus, the task is "to categorize the remaining 1,988K pages into 219 categories, based on the training data" (the participants are requested to categorize the training data with their system as well, for the purpose of ensemble learning). The same holds for the other 29 languages, as shown in Table 2.

4.1 Participants

Ten groups from seven countries participated in SHINRA2020-ML and three groups participated in SHINRA2021-ML. The list of participant groups and tasks in which they participated are listed in Table 3. Most of the systems in SHINA2020-ML have been described at the NTCIR-15 conference (Bui and Le-Hong, 2020; Cardoso et al., 2020; Abhishek et al., 2020; Nakayama and Sekine, 2020; Yoshioka and Koitabashi, 2020; Nishikawa and Yamada, 2020; Yoshikawa et al., 2020). The reports on SHINRA2021-ML are on the SHINRA homepage (SHINRA HP).

4.2 Evaluation Results

For each target language, a group can submit up to three runs using different methods. The system categorizes each page into one or more ENE (ver. 8.0) categories. The evaluation pages were selected from those without a link to the Japanese Wikipedia pages. The evaluation data are not disclosed because we want the systems to categorize all pages, and we can compare future results using the same data. If an estimated category is not an exact match, the system receives no score for the output. We evaluated the performance of the systems on a multi-label categorization using the micro average F1 measure, that is, the harmonic mean of precision and micro-averaged recall. Note that the distribution of the category in the test data may have differed from that of the training data because of the different characteristics of the links from the Japanese Wikipedia.

Table 4 shows statistical results. It includes 1) the F-measure of the best participating systems for each language, 2) the F-measure of the ensemble learning system., and 3) the upper bound recall by the system outputs, i.e., the percentage of entities to which the correct answer is proposed by at least one of the systems. For most languages, the ensemble system outperforms the best single system in the language (in red). The upper bound exceeds 90% in almost all languages, with an average of 96.75%.

5 Resource

This section describes the ensemble learning method, presents the sample data, and describes some statistics of the resource.

The ensemble learning method we employ is a simple voting method with minor adjustments because a participant can submit up to three systems. The systems submitted by the same participants were extremely similar; we assigned weights to the outputs, that is, one over the number of systems submitted by the participants. Then, the category with the most votes was taken as the ensemble system output.

The sample data for the resources are shown in Figure 2. "Page id" and "title" are information on the Wikipedia page. "ENE" constitutes the categorized Extended Named Entity information, which includes both the ID and name of ENE in JSON format.

The total number of entities we categorized is 32.5M pages in 30 languages. Table 6 shows the number of entities in each category for 30 languages (other than Japanese). In the table, the categories are specified at Level 2 for the categories of "Names" and Level 1 for the other categories. The two biggest categories are "Location" (24.33%) and "Person" (23.00%), followed by "Natural object" (13.85%) and "Product" (13.27%),

"Facility" (5.93%) and "Organization" (5.51%). The distribution is similar to that in the Japanese Wikipedia, which was manually validated, as shown in Table 1. We categorized a total of 920K Japanese Wikipedia entities.

Table 4 shows the average F-measure obtained using the ensemble learning system. The F-measure for the resources proposed is 86.76. We attempted to identify the error types in Table 7, which shows the confusion matrix of the system outputs and the correct answer. Most of the errors are related to the "Concept" and "IGNORED" categories (85% of the total errors). The systems often output the "Concept" even if the correct category is one of the named entities or "IGNORED". In addition, for the entities whose correct category is "IGNORED", the systems output some other categories. In addition to errors related to these two categories, the systems seem to confuse "Organization", "Location", and "Facility". One explanation is that facilities (such as schools, parks, airports, and roads) often have the properties of organization and location. Thus, the systems are robust except when identifying the "Concept" and "IGNORED" categories.

6 Related Work

Ontologies and structured KBs have been considered among the most important knowledge resources in NLP. Previously, several major projects intended to construct KBs have been undertaken. Cyc was one of the earliest projects, followed by more recent Wikipedia-based projects, such as DBpedia, YAGO, Freebase, and Wikidata. Moreover, there are shared tasks aimed at building techniques for knowledge base structuring, such as KBP and CoNLL. Here, we introduce these resources and projects and describe the points considered as issues to be solved in these projects.

Cyc ontology is a large KB constructed of commonsense knowledge (Lenat, 1995). It was one of the largest AI projects between 1980 and 1990, which mainly used human labor to construct a KB. Constructing and maintaining handmade KBs for the general domain is extremely costly. The KB suffered from a "knowledge acquisition bottleneck", which included coverage and consistency problems.

DBpedia is a more recent project that constructs structured information from semi-structured data on Wikipedia, such as infoboxes and categories (Lehmann et al., 2015). However, DBpedia is challenged by inaccuracy, low coverage, and lack of coherence. Like Cyc, infobox and categories in Wikipedia are also created by humans; however, they are non-experts in the ontology. The categories are extremely noisy. For example, in the Japanese Wikipedia, "Shinjuku Station which is a railway station, has a category, "Odakyu Electric Railway", which is a major railway company using the station. However, a station is not an instance of a railway company; therefore, this is not an appropriate category. This type of category definition (i.e., a topic rather than a hypernym relation) is allowed in Wikipedia. There are multiple instances of this in DBpedia, which heavily relies on Wikipedia. Additionally, several inconsistencies were observed in the category structure. It is not even a hierarchy and there is a loop in the category structure.

YAGO is an ontology constructed by mapping Wikipedia articles to WordNet synsets (Mahdisoltani et al., 2015). Similar to DBpedia, YAGO adopts attribute information extracted from infoboxes because no attribute is defined in the WordNet synsets.

Freebase is a project that constructs a structured knowledge base using crowdsourcing, similar to Wikipedia (Bollacker et al., 2008). However, when using the crowdsourcing approach, Freebase lacks a well-organized ontology. The resource is noisy and lacks coherence because it is created using unorganized crowds. The Freebase project was terminated and integrated into Wikidata.

Wikidata aims to be a structured knowledge base based on a crowdsourcing scheme (Vrandečić and Krötzsch 2014). It has noise and lacks coherence because it was constructed using a bottom-up approach, similar to Wikipedia.

Fine Grained Entity Recognition (FIGER) is a project that identifies 112 finely defined named entity classes, similar to ENE (Ling and Weld, 2012). The category in FIGER is biased and does not have attribute definitions for each category.

7 Conclusion

We reported a resource of Wikipedias in 31 languages categorized as ENEs. Japanese Wikipedia pages (920K pages) were categorized with an accuracy of 98.5% and 30 languages Wikipedia pages (32.5M pages) categorized with 87 F-measure. It was created through a shared-task

of Wikipedia categorization in 30 languages. We call this resource creation scheme RbCC". We have also been conducting structuring tasks (attribute extraction and link prediction) using RbCC under our ongoing project, "SHINRA".

Acknowledgments

This work was supported by JSPS KAKENHI Grant Number JP20269633.

0:Concept	51039	1.6.4.4:Public_Institution	5331	1.7.19.3:Movie	19381	1.10.5.1:Animal_Part	1944
1.0:Name_Other	5	1.6.4.5:Accommodation	892	1.7.19.4:Show	3131	1.10.5.2:Flora_Part	201
1.1:Person	269688	1.6.4.6:Medical_Institution	1615	1.7.19.5:Music	46889	1.11.0:Disease_Other	29
1.2:God	1278	1.6.4.7:School	25579	1.7.19.6:Book	21093	1.11.1:Animal_Disease	2229
1.3.0:Individual_Animal_Other	69	1.6.4.8:Research_Institute	1036	1.7.20.0:Printing_Other	753	1.12.0:Color_Other	200
1.3.1:Racehorse	3450	1.6.4.9:Market	107	1.7.20.1:Newspaper	794	1.12.1:Nature_Color	13
1.4.0:Organization_Other	3631	1.6.4.10:Power_Plant	446	1.7.20.2:Magazine	2495	2.0:Timex_Other	1
1.4.1:International_Organization	512	1.6.4.11:Park	2606	1.7.21.0:Doctrine_Method_Other	14767	2.1.0:Timeex_Other	0
1.4.2:Show_Organization	11758	1.6.4.12:Shopping_Complex	3341	1.7.21.1:Culture	303	2.1.1:Time	12
1.4.3:Family	2075	1.6.4.13:Sports_Facility	3783	1.7.21.2:Religion	936	2.1.2:Date	3964
1.4.4.0:Ethnic_Group_Other	1144	1.6.4.14:Museum	2901	1.7.21.3:Academic	1800	2.1.3:Day_Of_Week	10
1.4.4.1:Nationality	227	1.6.4.15:Zoo	445	1.7.21.4:Sport	955	2.1.4:Era	2067
1.4.5.0:Sports_Organization_Other	89	1.6.4.16:Amusement_Park	450	1.7.21.5:Style	586	2.2.0:Periodx_Other	5
1.4.5.1:Sports_Federation	865	1.6.4.17:Theater	1614	1.7.21.6:Movement	474	2.2.1:Period_Time	0
1.4.5.2:Sports_League	846	1.6.4.18:Worship_Place	9276	1.7.21.7:Theory	1808	2.2.2:Period_Day	1
1.4.5.3:Sports_Team	9107	1.6.5.0:Transport_Facility_Other	946	1.7.21.8:Plan	1818	2.2.3:Period_Week	1
1.4.6.0:Juridical_Person_Other	3	1.6.5.1:Car_Stop	5605	1.7.22.0:Rule_Other	327	2.2.4:Period_Month	1
1.4.6.1:Nonprofit_Organization	5122	1.6.5.2:Station	18296	1.7.22.1:Treaty	915	2.2.5:Period_Year	25
1.4.6.2:Company	30120	1.6.5.3:Airport	1559	1.7.22.2:Law	2289	3.0:Numex_Other	0
1.4.6.3:Company_Group	386	1.6.5.4:Port	575	1.7.23.0:Title_Other	16	3.1:Money	0
1.4.7.0:Political_Organization_Other	1256	1.6.6.0:Line_Other	1528	1.7.23.1:Position_Vocation	6317	3.2:Stock_Index	0
1.4.7.1:Government	3516	1.6.6.1:Railroad	3615	1.7.24.0:Language_Other	1544	3.3:Point	0
1.4.7.2:Political_Party	1719	1.6.6.2:Road	15550	1.7.24.1:National_Language	251	3.4:Percent	7
1.4.7.3:Cabinet	225	1.6.6.3:Canal	333	1.7.25.0:Unit_Other	496	3.5:Multiplication	0
1.4.7.4:Military	3744	1.6.6.4:Water_Route	455	1.7.25.1:Currency	319	3.6:Frequency	0
1.5.0:Location_Other	2671	1.6.6.5:Tunnel	379	1.8.0:Virtual_Address_Other	291	3.7:Age	5
1.5.1.0:GPE_Other	435	1.6.6.6:Bridge	1887	1.8.1:Channel	3348	3.8:School_Age	7
1.5.1.1:City	49028	1.7.0:Product_Other	14619	1.8.2:Phone_Number	4	3.9:Ordinal_Number	0
1.5.1.2:Province	12490	1.7.1:Video_Work	2621	1.8.3:Email	0	3.10:Rank	1
1.5.1.3:Country	1407	1.7.2:Musical_Instrument	684	1.8.4:URL	2	3.11:Latitude_Longitude	304
1.5.2.0:Region_Other	30	1.7.3:Clothing	936	1.9.0:Event_Other	1991	3.12.0:Measurement_Other	3
1.5.2.1:Continental_Region	275	1.7.4:Money_Form	273	1.9.1.0:Occasion_Other	2697	3.12.1:Physical_Extent	5
1.5.2.2:Domestic_Region	2219	1.7.5:Drug	689	1.9.1.1:Election	908	3.12.2:Space	1
1.5.3.0:Geological_Region_Other	2446	1.7.6:Weapon	11167	1.9.1.2:Religious_Festival	983	3.12.3:Volume	0
1.5.3.1:Spa	1132	1.7.7:Stock	0	1.9.1.3:Competition	17471	3.12.4:Weight	0
1.5.3.2:Mountain	4013	1.7.8:Award	3031	1.9.1.4:Conference	585	3.12.5:Speed	1
1.5.3.3:Island	2517	1.7.9:Decoration	247	1.9.2.0:Incident_Other	3465	3.12.6:Intensity	0
1.5.3.4:River	2900	1.7.10:Offense	176	1.9.2.1:War	3093	3.12.7:Temperature	0
1.5.3.5:Lake	858	1.7.11:Service	3	1.9.3.0:Natural_Phenomenon_Other	249	3.12.8:Calorie	0
1.5.3.6:Sea	302	1.7.12:Class	45	1.9.3.1:Natural_Disaster	355	3.12.9:Seismic_Intensity	1
1.5.3.7:Bay	362	1.7.13:Character	6354	1.9.3.2:Earthquake	355	3.12.10:Seismic_Magnitude	0
1.5.4.0:Astronomical_Object_Other	1516	1.7.14:ID_Number	5	1.10.0:Natural_Object_Other	1493	3.13.0:Countx_Other	4
1.5.4.1:Star	1130	1.7.15.0:Game_Other	350	1.10.1:Element	153	3.13.1:N_Person	5
1.5.4.2:Planet	3186	1.7.15.1:Degital_Game	11213	1.10.2:Compound	4493	3.13.2:N_Organization	2
1.5.4.3:Constellation	167	1.7.16:Software	4658	1.10.3:Mineral	556	3.13.3.0:N_Location_Other	0
1.5.5.0:Address_Other	0	1.7.17.0:Vehicle_Other	771	1.10.4.0:Living_Thing_Other	1410	3.13.3.1:N_Country	0
1.5.5.1:Postal_Address	1	1.7.17.1:Car	5187	1.10.4.1:Fungus	275	3.13.4:N_Facility	1
1.6.0:Facility_Other	5035	1.7.17.2:Train	4632	1.10.4.2:Mollusk_Arthropod	577	3.13.5:N_Product	0
1.6.1:Facility_Part	67	1.7.17.3:Aircraft	2619	1.10.4.3:Insect	858	3.13.6:N_Event	0
1.6.2:Dam	741	1.7.17.4:Spaceship	1531	1.10.4.4:Fish	966	3.13.7.0:N_Natural_Object_Other	0
1.6.3.0:Archaeological_Place_Other	2968	1.7.17.5:Ship	7887	1.10.4.5:Amphibia	132	3.13.7.1:N_Animal	0
1.6.3.1:Tomb	1093	1.7.18.0:Food_Other	2725	1.10.4.6:Reptile	1012	3.13.7.2:N_Flora	0
1.6.4.0:FOE_Other	2465	1.7.18.1:Dish	2888	1.10.4.7:Bird	1901	9:IGNORED	13360
1.6.4.1:Military_Base	462	1.7.19.0:Art_Other	945	1.10.4.8:Mammal	1868		
1.6.4.2:Castle	2014	1.7.19.1:Painting	408	1.10.4.9:Flora	4164		
1.6.4.3:Palace	237	1.7.19.2:Broadcast_Program	33747	1.10.5.0:Living_Thing_Part_Other	309		

Table 1. The number of entities for each category in the Japanese Wikipedia

{"pageid": "59706565", "title": "1978 Giro d'Italia, Prologue to Stage 10", "ENEs": [{"ENE_id": "1.9.1.3", "ENE_name": "Competition"}]}
{"pageid": "22059861", "title": "Tarlach Rua Mac Dónaill", "ENEs": [{"ENE_id": "1.1", "ENE_name": "Person"}]}
{"pageid": "53177250", "title": "90th Scripps National Spelling Bee", "ENEs": [{"ENE_id": "1.9.1.3", "ENE_name": "Competition"}]}
{"pageid": "13820024", "title": "The Early History of God", "ENEs": [{"ENE_id": "1.7.19.6", "ENE_name": "Book"}]}
{"pageid": "17870536", "title": "Pure type system", "ENEs": [{"ENE_id": "1.7.21.0", "ENE_name": "Doctrine_Method_Other"}]}
{"pageid": "13918760", "title": "Nyandeni Local Municipality", "ENEs": [{"ENE_id": "1.5.1.1", "ENE_name": "City"}]}
{"pageid": "14874071", "title": "BICD2", "ENEs": [{"ENE_id": "1.10.5.0", "ENE_name": "Living_Thing_Part_Other"}]}
{"pageid": "40760418", "title": "Meydan-e Sofla", "ENEs": [{"ENE_id": "1.5.1.1", "ENE_name": "City"}]}
{"pageid": "31722196", "title": "Princes' Concordat", "ENEs": [{"ENE_id": "1.7.22.1", "ENE_name": "Treaty"}]}
{"pageid": "5724950", "title": "Hatley, Quebec (township)", "ENEs": [{"ENE_id": "1.5.1.1", "ENE_name": "City"}]}

Figure 2. Sample data

Language	num. of pages	Links from ja	Ratio
English (en)	5,790,377	439,354	7.6
Spanish (es)	1,500,013	257,835	17.2
French (fr)	2,074,648	318,828	15.4
German (de)	2,262,582	274,732	12.1
Chinese (zh)	1,041,039	267,107	25.7
Russian (ru)	1,523,013	253,012	16.6
Portuguese (pt)	1,014,832	217,896	21.5
Italian (it)	1,496,975	270,295	18.1
Arabic (ar)	661,205	73,054	11.0
Japanese	1,136,222	–	–
Indonesian (id)	451,336	115,643	25.6
Turkish (tr)	321,937	111,592	34.7
Dutch (nl)	1,955,483	199,983	10.2
Polish (pl)	1,316,130	225,552	17.1
Persian (fa)	660,487	169,053	25.6
Swedish (sv)	3,759,167	180,948	4.8
Vietnamese (vi)	1,200,157	116,280	9.7
Korean (ko)	439,577	190,807	43.7
Hebrew (he)	236,984	103,137	43.5
Romanian (ro)	391,231	92,002	23.5
Norwegian (no)	501,475	135,935	27.1
Czech (cs)	420,195	135,935	25.1
Ukrainian (uk)	881,572	181,122	20.5
Hindi (hi)	129,141	30,547	23.6
Finnish (fi)	450,537	144,750	32.1
Hungarian (hu)	443,060	120,295	27.2
Danish (da)	242,523	91,811	35.6
Thai (th)	129,294	59,791	46.2
Catalan (ca)	601,473	139,032	23.1
Greek (el)	157,566	60,513	38.4
Bulgarian (bg)	248,913	89,017	35.7

Table 2. Wikipedia statistics in 31 languages

Year	Group ID	Country	Participated Language(s)
2020	CMVS	Finland	1 (ar)
	FPTAI	Vietnam	30 (all)
	HUKB	Japan	30 (all)
	PribL	Portugal	15 (ar, cs, de, en, es, fr, it, ko, nl, no, pl, pt, ru, tr, zh)
	RH312	India	6 (bg, fr, hi, id, th, tr)
	TKUIM	Taiwan	30 (all)
	Ousia	Japan	9 (ar, de, es, fr, hi, it, pt, th, zh)
	Uomfj	Australia/Japan	28 (all except el, sv)
	Vlp	Vietnam	1 (vi)
	LIAT	Japan	30 (all)
2021	HUKB	Japan	30 (all)
	KANJU	Japan	30 (all)
	junps	Japan	1 (en)

Table 3. SHINRA-ML task participants

Language	Best system F	Ensemble system F	Upper bound Recall
Arabic	90.06	92.18	97.71
Bulgarian	86.94	88.32	92.77
Catalan	89.25	86.62	95.42
Czech	81.70	83.72	94.34
Danish	82.34	81.53	92.15
German	79.68	80.93	89.68
Greek	84.85	79.72	90.04
English	86.49	87.65	93.49
Spanish	85.54	86.51	94.69
Persian	89.63	90.87	94.63
Finish	85.95	86.36	95.47
French	83.77	87.20	92.83
Hebrew	81.80	81.74	90.74
Hindi	87.81	90.76	94.79
Hungarian	89.93	91.41	96.19
Indonesian	90.71	92.22	97.28
Italian	84.08	85.51	91.85
Korean	80.08	82.57	90.68
Dutch	85.88	85.88	91.51
Norwegian	85.89	86.10	93.53
Polish	84.44	85.06	94.00
Portuguese	88.96	89.83	96.11
Romanian	93.43	93.43	98.07
Russian	81.62	84.06	90.91
Swedish	84.28	86.28	91.85
Thai	84.72	85.48	95.31
Turkish	87.85	88.13	94.18
Ukrainian	85.14	84.53	91.20
Vietnamese	90.28	89.58	95.14
Chinese	88.00	88.72	95.10
Average	**86.04**	**86.76**	**96.75**

Table 4. Performance of the system and other statistics

Name	Task	Language	Target categories
SHINRA2018	Attribute value extraction	Japanese	5 categories
SHINRA2019	Attribute value extraction	Japanese	35 categories
SHINRA2020-JP	Attribute value extraction	Japanese	78 categories
SHINRA2020-ML	Categorization	30-language	
SHINRA2021-LinkJP	Link Prediction	Japanese	7 categories
SHINRA2021-ML	Categorization	30-language	
SHINRA2022	All three tasks	Japanese	all categories

Table 5. SHINRA tasks

ID	0	1	1.1	1.2	1.3	1.4	1.5	1.6	1.7	1.8	1.9	1.1	1.11	1.12	2	3	9
Category	Concept	Name Other	Person	God	Individual Animal	Organization	Location	Facility	Product	Virtual Address	Event	Natural Object	Disease	Color	Timex	Numex	IGNORED
Arabic	52972	0	204207	513	3	26264	198716	21437	66559	2093	20277	38404	4937	97	7733	480	16849
Bulgarian	17533	0	73267	588	7	13016	53001	11304	32931	746	6704	31913	679	20	3651	51	4358
Catalan	51271	0	149258	946	7	30121	155361	41240	67885	1316	20958	64479	1652	84	4156	540	12471
Czech	50169	0	116294	796	34	28476	74478	32128	68588	1515	18266	17454	1396	37	4982	1	15654
Danish	27238	1	73171	481	12	19399	33795	16822	45794	989	8853	7121	758	31	3764	786	3577
German	196031	0	768225	2837	279	192658	396296	192032	270062	4767	82408	64601	6401	84	10350	0	76738
Greek	74786	0	67372	1342	2	28297	48940	5861	53778	1339	13157	5978	1070	85	8462	1	25595
English	315531	0	1728797	4423	5822	432236	875918	472005	1005969	43520	279123	403739	11813	334	8829	811	204327
Spanish	105543	2	376710	2265	89	90700	317901	92715	246103	4363	72964	153631	4378	187	8479	226	38106
Persian	48311	0	138148	763	16	19230	241298	46761	100016	1354	9146	36456	2171	79	8104	711	8759
Finish	42147	0	147556	784	48	34477	48877	21505	99635	1400	15659	23084	1599	31	4022	1	10071
French	129098	0	595708	2393	573	135080	416779	149286	355881	6334	87986	119710	4722	254	11182	630	60099
Hebrew	31475	1	78479	484	15	16658	20233	10751	47355	859	8045	8761	1467	39	5501	0	7122
Hindi	14388	0	22242	406	2	5113	46309	6144	20433	848	3150	3508	643	47	5728	522	2693
Hungarian	30551	1	110023	650	12	20570	137258	25902	60441	1266	18818	22436	688	18	4670	0	9952
Indonesian	26194	0	72630	669	22	21450	115722	20662	69492	1818	11991	98328	1050	32	3734	190	9556
Italian	98522	0	378736	1962	93	92536	313896	79661	325760	2873	109402	48390	4017	247	9276	7	32724
Korean	52713	0	111535	761	26	32660	46817	54853	88858	2155	15243	14963	1325	70	6036	381	11935
Dutch	120574	0	217528	1349	30	68301	342682	88123	148271	2128	47810	874290	2611	68	5067	88	36880
Norwegian	43235	0	158039	696	16	39756	73512	45573	71868	1645	25076	23477	1084	33	3988	21	13680
Polish	84346	0	354727	1805	55	79672	386271	93985	173646	2469	52842	51219	4422	49	4921	3	26274
Portuguese	80353	1	224466	1520	52	65736	240881	43866	181481	4039	48010	90099	3473	94	5829	536	24859
Romanian	22821	0	57913	582	6	13518	201132	11420	33084	1020	5905	30486	657	20	3640	1	9209
Russian	102057	1	463956	2005	57	98493	409426	76356	207009	2795	57119	53424	3288	96	7241	388	40425
Swedish	206567	0	231159	1301	288	53538	1729196	88261	119368	1459	23894	1281506	2160	74	4332	4	16006
Thai	14726	0	28041	378	2	11195	10381	8738	25335	905	6139	6944	833	32	3843	615	11570
Turkish	26188	0	83506	859	112	18642	83704	12515	59418	1755	15271	7721	1149	88	5139	237	9344
Ukranian	67899	0	178244	1381	8	39675	348825	45919	101043	1759	21852	47983	1734	67	5596	2	20769
Vietnamese	46134	0	55154	430	20	9533	245542	10573	38259	882	8636	773560	966	56	4495	5	6613
Chinese	61095	0	217583	1633	267	56013	301227	103105	133350	3371	28580	100343	1985	114	7484	249	25499
Total	2179373	7	7265091	35369	7708	1737000	7613147	1826398	4184322	100411	1114704	4403665	73143	2453	172750	7238	766215
%	6.89%	0.00%	23.00%	0.11%	0.02%	5.51%	24.33%	5.93%	13.27%	0.32%	3.51%	13.85%	0.23%	0.01%	0.55%	0.02%	2.43%

Table 6. Number of entities in each category and language (for languages other than Japanese)

Note that the statistics are based on second level categories. All sub-categories are summed together for second level categories. For example, the number of entities under Organization (1.4), such as international organizations (1.4.1), companies (1.4.6.2), political parties (1.4.7.2), and other categories under 1.4, are combined in the statistics for Organization (1.4). The total number of entities for each language might not be equal to the number of entities in Table 2 because there could be over one category for each entity.

ID	0	1.1	1.2	1.3	1.4	1.5	1.6	1.7	1.8	1.9	1.1	1.11	2	3	9
Category	Concept	Person	God	Individual Animal	Organization	Location	Facility	Product	Virtual Address	Event	Natural Object	Disease	Timex	Numex	IGNORED
0:Concept	587	-	-	-	1	2	-	24	-	2	65	1	-	-	1
1.1:Person	9	3675	-	-	4	-	-	5	-	-	-	-	-	-	3
1.2:God	-	-	14	-	-	-	-	-	-	-	-	-	-	-	-
1.3:Individual_Animal	-	1	-	-	-	-	-	1	-	-	1	-	-	-	-
1.4:Organization	15	6	-	-	760	20	12	5	-	3	-	-	-	-	3
1.5:Location	11	-	-	-	4	3371	20	1	-	-	1	-	-	-	2
1.6:Facility	12	3	-	-	11	26	906	6	-	1	1	-	-	-	3
1.7:Product	103	12	1	-	13	1	10	1824	-	9	3	-	1	-	4
1.8:Virtual_Address	1	-	-	-	5	-	-	1	28	-	-	-	-	-	-
1.9:Event	5	1	-	-	7	-	1	11	-	416	-	-	2	-	8
1.10:Natural_Object	2	-	-	-	-	-	-	1	-	-	91	-	-	-	-
1.11:Disease	-	-	-	-	-	-	-	-	-	-	-	23	-	-	-
2:Timex	-	-	-	-	-	-	-	1	-	3	-	-	46	-	-
3:Numex	-	-	-	-	-	-	-	-	-	-	-	-	-	1	1
9:IGNORED	287	171	2	-	73	181	33	138	1	45	9	-	18	2	398

Table 7. Confusion matrix of system outputs (horizontal) and correct answers (vertical) for all languages

References

Tushar Abhishek, Ayush Agarwal, Anubhav Sharma, Vasudeva Varma, and Manish Gupta (2020). Rehoboam at the NTCIR-15 SHINRA2020-ML task. In The 15th NTCIR Conference Evaluation of Information Access Technologies (NTCIR-15).

Kurt Bollacker, Colin Evans, Praveen Paritosh, Tim Sturge, and Jamie Taylor. (2008) Freebase: a collaboratively created graph database for structuring human knowledge. In SIGMOD'08: Proceedings of the 2008 ACM SIGMOD international conference on Management of data, pages 1247-1250.

The Viet Bui and Phuong Le-Hong (2020). Cross-lingual Extended Named Entity Classification of Wikipedia Articles. In The 15th NTCIR Conference Evaluation of Information Access Technologies (NTCIR-15).

Ruben Cardoso, Afonso Mendes, and Andre Lamurias (2020). Priberam Labs at the NTCIR-15 SHINRA2020-ML: Classification Task. In The 15th NTCIR Conference Evaluation of Information Access Technologies (NTCIR-15).

ENE-HP. Extended Named Entity homepage. In https://ene-project.info.

Jens Lehmann, Robert Isele, Max Jakob, Anja Jentzsch, Dimitris Kontokostas, Pablo N. Mendes, Sebastian Hellmann, Mohamed Morsey, Patrick van Kleef, Soren Auer, and Christian Bizer (2015). DBpedia - A large-scale, multilingual knowledge base extracted from Wikipedia. Semantic Web, 6(2):167-195.

Douglas Lenat (1995). Cyc: A large-scale investment in knowledge infrastructure. Communications of the ACM, 38(11):33-38.

Xiao Ling and Daniel S. Weld (2012). Fine-Grained Entity Recognition. In Proceedings of the Twenty-Sixth AAAI Conference on Artificial Intelligence, AAAI'12, pages 94-100. AAAI Press.

Farzaneh Mahdisoltani, Joanna Biega, and Fabian M Suchanek (2015). YAGO3: A Knowledge Base from Multilingual Wikipedias. CIDR.

David Nadeau, Satoshi Sekine (2007). A survey of Named Entity Recognition and Classification". Lingvisticae Investigationes 30 (1), 3-26.

Kouta Nakayama and Satoshi Sekine (2020). LIAT Team's Wikipedia Classifier at NTCIR-15 SHINRA2020-ML: Classification Task. In The 15th NTCIR Conference Evaluation of Information Access Technologies (NTCIR-15).

Sosuke Nishikawa and Ikuya Yamada (2020). Studio Uusia at the NTCIR-15 SHINRA2020-ML Task. In The 15th NTCIR Conference Evaluation of Information Access Technologies (NTCIR-15).

Satoshi Sekine (2008). Extended named entity ontology with attribute information. In the Sixth International Conference on Language Resource and Evaluation (LREC08).

Satoshi Sekine and Chikashi Nobata (2004). Definition, dictionaries, and tagger for extended named entity hierarchy. In the Fourth International Conference on Language Resources and Evaluation (LREC'04).

Satoshi Sekine, Kiyoshi Sudo, and Chikashi Nobata (2002). Extended named entity hierarchy. In the Third International Conference on Language Resources and Evaluation (LREC'02).

Satoshi Sekine, Akio Kobayashi, and Kouta Nakayama (2019). SHINRA: Structuring wikipedia by collaborative contribution. In Automated Knowledge Base Construction (AKBC).

SHINRA-HP. SHINRA project homepage. URL https://shinra-project.info.

SHINRA2020-ML-HP. SHINRA 2020-ML homepage. URL http://shinra-project.info/shinra2020ml/.

Masatoshi Suzuki, Koji Matsuda, Satoshi Sekine, Naoki Okazaki, and Kentaro Inui. (2018) A joint neural model for fine-grained named entity classification of Wikipedia articles. IEICE Transactions on Information and Systems, E101.D(1):73-81.

U.S. National Institute of Standards and Technology (NIST) . TAC Knowledge Base Population (KBP) 2017, 2018. URL https://tac.nist.gov/2017/KBP/.

Denny Vrandečić and Markus Krötzsch (2014). Wikidata: A free collaborative knowledgebase. Commun. ACM, 57(10):78-85.

Hiyori Yoshikawa, Chunpeng Ma, Aili Shen, Qian Sun, Chenbang Huang, Guillaume Pelat, Akiva Miura, Daniel Beck, Timothy Baldwin, and Tomoya Iwakura (2020). UOM-FJ at the NTCIR-15 SHINRA2020-ML task. In The 15th NTCIR Conference Evaluation of Information Access Technologies (NTCIR-15).

Masaharu Yoshioka and Yoshiaki Koitabashi (2020). HUKB at SHINRA2020-ML task. In The 15th NTCIR Conference Evaluation of Information Access Technologies (NTCIR-15).

Accuracy meets Diversity in a News Recommender System

Shaina Raza[1*], Syed Raza Bashir[2], Usman Naseem[3]
[1] University of Toronto, Toronto, ON, Canada
[2] Toronto Metropolitan University, Toronto, ON, Canada
[3] The University of Sydney, Sydney, Australia
shaina.raza@utoronto.ca, razabashir55@hotmail.com, usman.naseem@sydney.edu.au

Abstract

News recommender systems face certain challenges. These challenges arise due to evolving users' preferences over dynamically created news articles. Diversity is necessary for a news recommender system to expose users to a variety of information. We propose a deep neural network based on a two-tower architecture that learns news representation through a news item tower and users' representations through a query tower. We introduce diversity in the proposed architecture by considering a category loss function that aligns items' representation of uneven news categories. Experimental results on two news datasets reveal that our proposed architecture is more effective compared to the state-of-the-art methods and achieves a balance between accuracy and diversity.

1 Introduction

Reading the news has never been more common in people's daily lives than it is now. Big names like Yahoo!, Google, and CNN have launched online news portals that users can access from anywhere to browse various news categories and find up-to-date information. However, finding the right content is a challenge. With so much information available online, selecting relevant news has become a time-consuming and challenging task. A news recommender system (NRS) offers solutions to the information overload problem and provides relevant and interesting news recommendations to users (Raza and Ding, 2021b).

In the state-of-the-art of NRS, the news stories that users have read in the past are used to infer their interests and preferences. However, there is usually frequent content updating in a news domain. We show the news reading behaviour of a typical user in Figure 1 as an example.

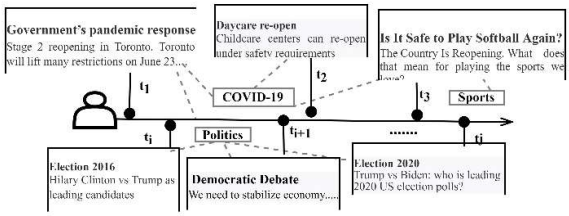

Figure 1: A user's behaviour during news reading

We see in Figure 1 that this user reads about the *'COVID-19 cases in Ontario'* during time t_1, and then read about *'Canadian children 5 to 11 could become eligible for COVID-19 vaccine…*" during t_2, and then his following action (read/click) is reading the news '*COVID-19 travel ban on tourist*s' during t_3. We also observe that there is a transition in user preferences to other topics, for example, the elections, and politics in the later time. This shows that a user in an NRS generally changes his/her interests over time. Some of the user interests are long-term which shows a habit or personality, while some of the user interests are short-term, which may erupt due to some trending news, a sudden interest in an event and so. It is important to consider the accuracy as well as the diversity of news items while providing recommendations to users in an NRS (Raza and Ding, 2021a). This is a motivation for this research.

A common practice in the recommender systems is to design a two-tower architecture (Wang et al., 2019), where, first, a retrieval model retrieves a subset of related items from a large corpus in response to a user's query, and, then a ranking model ranks the retrieved items based on users' actions (clicks or ratings). The quality of retrieved items plays a critical role in the retrieval stage. The retrieval and ranking mechanism is also used in two-tower architectures (Yang et al., 2020).

In a typical two-tower architecture (Yi et al., 2019; Yang et al., 2020; Wang et al., 2019), there is usually no mechanism for the information interaction between the two towers. Usually, the

Proceedings of the 29th International Conference on Computational Linguistics, pages 3778–3787
October 12–17, 2022.

items or users are represented by their IDs or titles (for example, movie ID, news ID, user ID) to represent each tower. By missing the chance to include rich information related to news items (e.g., news body, categories, etc.) or user contexts (e.g., a situation when the user interacts with the news, such as time, and place) in the two-tower architecture, we are providing recommendations that are not quite relevant to users' preferences.

An issue with a typical recommender system is that they only focus on the relevancy of users' preferences and there is usually no consideration of diversity aspects while making recommendations (Raza and Ding, 2021a). For example, if we consider a real-time news recommendation scenario, we find that there are different categories of news (for example, sports, entertainment, politics, weather, and such), and the number of news items in each category varies (i.e., are imbalanced). As a result, the items in a single news category may account for most news recommendations. Consequently, the news recommendations that are produced may be too narrow and users' diversified interests are not addressed.

To address the above issues (incorporating rich user-item interactions and diversity of recommendations), we present a novel two-tower architecture for an NRS. We summarize our contributions as:

- We present a two-tower architecture for NRS and supplement the item tower with rich side information (meta-data) related to news items. We also consider the user context(s) in the query tower. We represent each user query by an augmented vector, which consists of the user's query and the news item features. The augmented vector is then updated based on the output representation vector of the other tower for a positive sample. In this way, the augmented vector implicitly models the information interaction between the two towers.

- To introduce diversity in the two-tower architecture, we include a category loss function during the training phase that aligns the representation of news items from a variety of news categories.

Extensive experiments on two news datasets show that our proposed approach has two major advantages: (i) it provides deeper insights into the information interaction of two-tower models in an NRS, and (ii) it provides diversified news recommendations (along with relevant recommendations) from a variety of news categories. Our goal is to provide news recommendations that are relevant to users' past preferences (interests) and are diversified at the same time

2 Related Work

Deep learning has demonstrated great success in recommender systems, such as in movie recommendations (WeAreNetflix, 2018), social networks (Ojagh et al., 2020) and many other application domains. Recommending news is particularly challenging (Raza and Ding, 2021b). This is because of the dynamic nature of the news domain and changing users' preferences. The state-of-the-art NRSs (Wang et al., 2018; An et al., 2019; Zhu et al., 2019) has shown tremendous performance in both academia and industry, however, a few challenges need to be addressed. First, these models do not extract enough news data from a reader's history. There are many pieces of information, other than news ID or title, that may be more descriptive (e.g., the news story) or better reflect a reader's interests (e.g., topics, categories) than titles or IDs. Second, the focus in these methods is usually on the relevancy and not on the diversity aspect.

Some works consider the two-tower deep neural networks to learn representation from content features in language models (Chidambaram et al., 2018; Logeswaran and Lee, 2018). These two-tower models are also used in recommender systems to leverage content features on the item side (Yi et al., 2019). Nevertheless, these models are usually focused on the relevancy of retrieved items, which is appreciated, however, they are not used to address the diversity aspect. In this work, we use a two-tower architecture for an NRS and try to incorporate both accuracy (relevancy) and diversity of news items while making recommendations to the users.

Maximal marginal relevance (MMR) is a classical technique to increase the diversity of documents retrieved against a query in an information retrieval system. MRR is also used in recommender systems to include diversity during the re-ranking phase of recommended items (Ziegler et al., 2005).

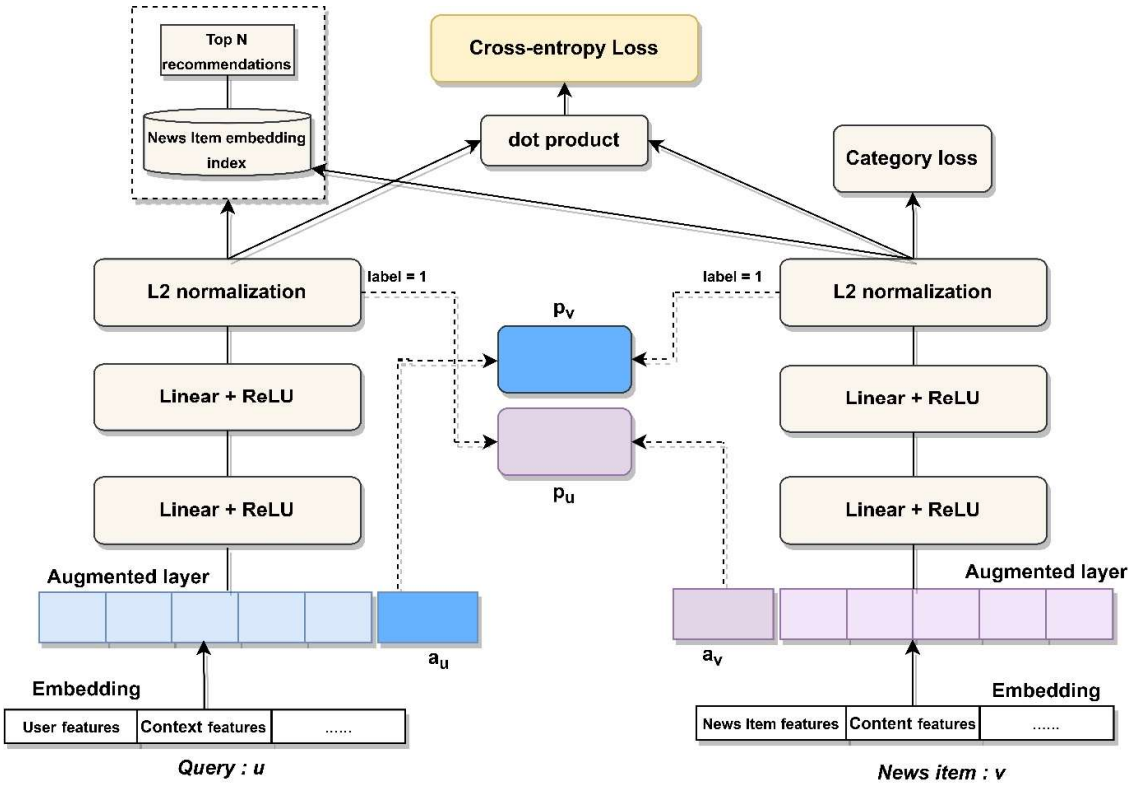

Figure 2: Network architecture for our proposed two-tower NRS

Some work (Raza and Ding, 2020; Qin and Zhu, 2013) considers diversity in the recommendation process by using the regularization terms on items' feature space. Dueling Bandit Gradient Descent (DBGD) (Afsar et al., 2021) is an online learning-to-rank algorithm based on multi-arm bandit algorithms and is used to model the exploration-versus-exploitation trade-off for relative feedback. DBGD is recently used in a state-of-the-art NRS (Zheng et al., 2018) to improve recommendation diversity. However, the learning efficiency of this model is limited in high-dimensional parameter space. Second, this method assumes only binary feedback because there is no way of directly observing the reward of users' actions. In this work, we also consider diversity during the model training time, but we use a category loss function for this purpose. Our intuition is that news items under different news categories are highly imbalanced and the recommendations are usually produced considering one such major category. We try to provide recommendations by considering news items from a diverse set of news categories.

3 Methodology

3.1 Problem Formulation

Considering the news item set $\{v_i\}_{i=1}^{N}$ and a query set as $\{u_j\}_{j=1}^{M}$, where N represents the number of news items, and M is the number of users, we refer to the news recommendation problem R, as selecting the candidate news items from the entire news corpus given a certain query.

We refer to a query as the feedback given by the user. We present the query-item feedback as a matrix $R \in \mathbb{R}^{N \times M}$. If the query j gives positive feedback on a news item i, then we consider it as $R_{ij} = 1$ (positive feedback), otherwise $R_{ij} = 0$. We represent each news item by the news content features, including news ID and side information, such as news title, body, and category. We also represent a user query with contextual factors, such as the time, and place when the user interacts with the item. By providing more information related to the query and the item, we can model rich interactions between two towers in a two-tower architecture.

3.2 Proposed two-tower architecture

We show our proposed framework for the two-tower architecture in Figure 2 and explain its work next:

Embedding Layer: The first layer is the embedding layer in the two-tower architecture. We define an embedding matrix E $\in \mathbb{R}^{K\times D}$, where E is the embedding matrix, K is the embedding dimension and D refers to the number of unique features. Each piece of information or feature in u_j and v_i goes through the embedding layer and is mapped to a low-dimensional dense vector, $e_j \in \mathbb{R}^K$, where e_j is the jth column of E.

Augmented Layer: First, we create two input feature vectors $\mathbb{z}_u$ and $\mathbb{z}_v$ that contains the information about the current query vector u and the news item vector v. Then, we create two augmented vectors a_u and a_v by the IDs, corresponding to u and v respectively. These vectors a_u and a_v are then concatenated with $\mathbb{z}_u$ and $\mathbb{z}_v$ vectors to obtain the information from the feature vectors. We show $\mathbb{z}_u$ and $\mathbb{z}_v$ as shown in Equations (1) and (2) respectively:

$$\mathbb{z}_u = [e_{123} \; ||e_{To}||e_{time}|| \dots ||a_u] \quad (1)$$

$$\mathbb{z}_v = [e_{345} \; ||e_{sport}||e_{NYTimes}|| \dots ||a_v] \quad (2)$$

Where e_f, f is a feature that is related to u (e.g., place when the user interacts with the item) or v (e.g., news category sports or source NYTimes related to a news item), and the notation || refers to the vector concatenation operation.

The concatenated vectors $\mathbb{z}_u$ and $\mathbb{z}_v$ are fed into two towers (query and news item), which consist of the fully connected layers with the ReLU activation function. These layers receive the information between two towers through the augmented vectors a_u and a_v (a_u and a_v provides information about users' positive interactions). The output from the fully connected layers goes through ℓ_2 normalization layer that gives the augmented representations of query q_u and news item p_v. We define these steps formally as shown in Equation (3):

$$p = \ell_2 norm(h_L) \qquad (3)$$

$$h_L = \text{RELU}(W_l h_{L-1} + b_l)$$

$$h_1 = \text{RELU}(W_1 z + b_1)$$

where notation ℓ refers to loss, $\ell_2 norm$ is ℓ_2 −normalization, subscript L in h refers to layer, h is the intermediate representation, z denotes z_u and z_v , p refers to p_u and p_v ; W_l denotes the weight matrix in l^{th} layer, b refers to the bias vector. The representation p, is the output vector of the ℓ_2 normalization layer.

Loss function: We define the loss function as the mean square error between the augmented vector and query/item embedding for each sample of which label equals 1. The goal of the loss function is to use the augmented vector to fit all positive interactions in the tower belonging to the corresponding query/ news item. Formally, the loss functions are defined in Equations (4) and (5).

$$\ell_u = \frac{1}{T}\sum_{(u,v,y)\in Tr} [ya_u + (1-y)\, p_u - p_u]^2 (4)$$

$$\ell_v = \frac{1}{T}\sum_{(u,v,y)\in Tr} [ya_v + (1-y)p_v - p_v]^2 \quad (5)$$

where ℓ_u refers to the loss function with query vector and ℓ_v is a loss function associated with a news item vector. Tr is a training dataset, T refers to query-item pairs Tr, $y \in \{0,1\}$ is the label. If the label $y = 1$, it means the augmented vectors a_u and a_v approach the query embedding p_u and the news item embedding p_v, otherwise $y = 0$. We apply the stop gradient strategy to stop the gradient of ℓ_u and ℓ_v from flowing back into p_u and p_v respectively.

Once the augmented vectors a_u and a_v are obtained, they can model the information interaction between the two towers, and these vectors are considered as the input feature of the two towers. Finally, the output of the model is the inner product of the query embedding and news item embeddings, as shown in Equation (6):

$$s(u,v) =< p_u, p_v > \quad (6)$$

where $s(u,v)$ refers to the score provided by the proposed model.

News categories and diversity: In real-time news recommendation scenarios, the categories of news items are usually diverse (e.g., sports, politics, entertainment and so) and the number of news items under each new category is usually uneven. To incorporate diversity, we need to consider the recommendations from diverse news categories. To accomplish this, we propose another loss function related to the news category during the training phase that transfers the knowledge learned in one news category to the other news categories.

In particular, the news item representation p_v of a major news category $Cate^{major} = \{{p_v}^{major}\}$ (having the largest amount of data) is taken and transferred to other category sets

$Cate^2, Cate^3, Cate^4, ..$ and so. We define a loss function as the distance between the second-order statistics (covariances) of the major category and other news categories' features (Bello et al., 2008), shown in Equation (7):

$$\ell_{cate} = \sum_{i=2}^{n} ||\mathbb{C}\,(Cate^{major} - \mathbb{C}(Cate^{i})||_F^2) \quad (7)$$

where $\mathbb{C}\,(\cdot)$ denotes the covariance matrix, $||.||_F^2$ refers to the square matrix Frobenius norm and n is the number of news categories.

3.3 Model Training

We treat the news recommendation problem as a binary classification problem and use a random negative sampling technique, following the standard practice (Wu et al., 2020; Wu et al., 2021a; An et al., 2019) in most NRS. In particular, for each query in the positive query-item pair (the label = 1), we randomly sample N news items from the news corpus to create $\mathbb{S}$ negative query-item pairs (label = 0) with this query and add these $\mathbb{S}+1$ pairs to the training dataset. In the training process, we use binary cross-entropy to calculate the loss for the pairs, as shown in equation (8):

$$\ell_p = \frac{1}{T}\sum_{(u,v,y)\in Tr} [y log\sigma(\prec p_u, p_v \succ) + (1-y)\log(1-\sigma(\prec p_u, p_v \succ))] \quad (8)$$

$$Tr = D \times (\mathbb{S}+1)$$

where $\sigma\,(\cdot)$ refers to the sigmoid function, D denotes the number of positive feedback query-item pairs and Tr refers to the total number of training pairs.

Final loss function: The final loss function is calculated as shown in Equation (9):

$$\ell_{total} = \ell_p + \gamma_1\ell_u + \gamma_2\ell_v + \gamma_3\ell_{Cate} \quad (9)$$

where $\gamma_1, \gamma_2, \gamma_3$ refers to the tunable parameters.

4 Experiment

4.1 Datasets

MIcrosoft News Dataset (MIND): MIND (Wu et al., 2020) is a benchmark dataset consisting of anonymized behaviour logs from Microsoft News.

New York Times (NYTimes): we collected the news articles and the anonymized readers' interactions using NYTimes API. A sample of the dataset can be accessed here[1]. Both datasets consist of English news articles. We generated the training samples from the click histories and impression logs according to the format given in the MIND paper (Wu et al., 2020). The basic details for both datasets are shown in Table 1.

Dataset	MIND -small	NYTimes
Duration	6 weeks (12th Oct. 2019 - 22nd Nov. 2019)	2 years (1st Jan. 2017 - 31st Dec. 2018)
Readers	50,000	240,000
News	161,013	15,000
Clicks	156,925	2,000,000
News information	ID, headline, snippet (abstract), category, subcategory, publication timestamp	
Reader information	ID, interaction timestamp, click history, impressions	

Table 1: Dataset details

4.2 Evaluation methodology and metrics

Following the standard evaluation methodology and metrics in NRS (Wu et al., 2020; Wang et al., 2018), we conduct a time-based splitting and use the following evaluation metrics.

Accuracy metrics: Normalized Discounted Cumulative Gain (NDCG) and F1-score (harmonic mean of precision and recall).

Diversity metric: we use GiniIndex (GINI) (Sun et al., 2019b) for diversity, as it is a commonly used diversity metric in this kind of problem like diversity and fairness (Wu et al., 2021b).

tradeoff (Raza and Ding, 2020; Raza and Ding, 2021a)*:* We consider the trade-off between the mean F1-score (accuracy) and mean GINI index (diversity) scores. Our trade-off metric is defined in Equation 10:

$$tradeoff = 2 * \frac{(accuracy * diversity)}{(accuracy + diversity)} \quad (10)$$

Both the F1 and GINI scores are in the range [0,1]. We take the mean of top @ 10, 20 and 50 for calculating means of accuracy and diversity. We show the results in percentages.

4.3 Baselines Methods

We use the following baseline methods:

BERT4Rec (Sun et al., 2019a) with bidirectional self-attention to model user behaviour sequences. We use Bayesian Personalized Ranking (BPR) (Rendle et al., 2012) as the loss function.

[1] https://github.com/deeplearningnrs/D2NN

		MIND			NYTimes		
Model	**k**	**NDCG**	**F1**	**Gini**	**NDCG**	**F1**	**Gini**
Our approach	10	43.80%	36.90%	**87.40%**	51.00%	38.00%	**78.30%**
	20	50.10%	44.60%	*82.70%*	53.50%	45.00%	73.90%
	50	**63.50%**	**62.10%**	77.40%	**66.70%**	**65.50%**	73.30%
	trade-off		**68.91%**			**69.18%**	
BERT4Rec	10	41.80%	24.50%	74.30%	43.00%	25.00%	74.00%
	20	56.60%	42.20%	72.40%	58.00%	42.50%	71.80%
	50	*60.10%*	*57.90%*	70.60%	*62.00%*	*60.10%*	69.00%
	trade-off		*63.62%*			*64.24%*	
Two-tower	10	37.20%	44.50%	60.00%	37.80%	45.20%	56.00%
	20	38.10%	48.50%	57.90%	39.60%	49.50%	52.50%
	50	39.10%	49.70%	54.50%	42.00%	51.20%	51.30%
	trade-off		51.99%			51.25%	
Youtube-DNN	10	31.40%	35.30%	69.40%	33.00%	36.10%	65.00%
	20	34.50%	39.40%	58.90%	35.20%	40.40%	55.70%
	50	41.40%	46.40%	57.90%	42.00%	47.00%	54.20%
	trade-off		51.52%			50.34%	
LSTUR	10	39.70%	30.70%	81.20%	35.80%	22.10%	*76.40%*
	20	44.90%	32.60%	79.70%	42.00%	28.90%	75.90%
	50	55.10%	49.30%	72.50%	52.40%	34.20%	69.90%
	trade-off		58.69%			45.93%	
MultiVAE	10	32.90%	32.00%	58.90%	38.90%	34.20%	53.40%
	20	42.30%	39.00%	58.50%	44.00%	40.20%	51.10%
	50	43.40%	42.60%	57.60%	45.70%	44.50%	50.00%
	trade-off		48.98%			47.09%	
RepeatNet	10	37.80%	22.00%	74.70%	38.00%	25.00%	74.30%
	20	39.90%	24.10%	69.20%	41.00%	24.50%	65.80%
	50	43.20%	40.80%	65.30%	44.50%	41.00%	62.50%
	trade-off		50.22%			49.52%	
SASRecF	10	31.70%	24.00%	77.50%	32.00%	24.40%	74.00%
	20	32.70%	31.80%	75.80%	33.50%	32.00%	70.00%
	50	35.40%	32.10%	72.20%	36.80%	33.40%	68.80%
	trade-off		44.44%			44.97%	
ENMF	10	27.20%	14.50%	64.60%	28.90%	18.30%	63.70%
	20	28.10%	23.50%	57.90%	31.20%	25.00%	55.00%
	50	31.40%	29.70%	54.50%	33.40%	32.40%	52.40%
	trade-off		38.45%			40.04%	

Table 2: Performance of all models (Bold means best result, italic is second best, and underline is third-best)

MultiVAE (Liang et al., 2018), a collaborative filtering method with variational autoencoders. We use the cross-entropy as the loss function type.

ENMF (Chen et al., 2020) is an efficient matrix factorization method without sampling. We use the cross-entropy as the loss function for this model.

SASRecF (Zhang et al., 2019) is a feature-level self-attention model. We choose the BPR as the loss function.

RepeatNet (Ren et al., 2019) is a session-based recommender. We choose the BPR as loss function.

LSTUR (An et al., 2019) is an NRS that addresses long-short term users' preferences. We minimize the summation of negative log-likelihood of all positive samples during training,

Two-tower Model (Huang et al., 2013), is a standard two-tower model in retrieval tasks to leverage rich content features.

YouTubeDNN (Covington et al., 2016) is also a two-tower approach that feeds vectors into a multi-layer feed-forward neural network.

4.4 Hyperparameters

We implemented these models in TensorFlow. The embedding dimension and batch size were fixed to 32 and 256. We use the Adam optimizer. Other hyperparameters of all models were individually tuned to achieve optimal results to ensure a fair comparison. The dimensions of augmented vectors a_u and a_v were both set to $d = 32$, the tuning

parameter γ_1, γ_2 were set to 0.5 and γ_3 to 1. We set top@ k to 10, 20 and 50, as it is normally good practice to retrieve a relatively large number of candidate news items to rank.

5 Results and Analysis

5.1 Overall results

The comparison between our model and baselines is shown in Table 2. These scores are calculated using top@ 10, 20 and 50. We expect a good tradeoff score to be above 50% as it is a harmonic mean score.

Overall, we see in Table 2 that our proposed model has the highest accuracy, diversity and trade-off values on both datasets (MIND and NYTimes). This is shown by the highest F1-score, NDCG, Gini index and trade-off values of our approach among all the baseline methods. The superiority of our model is attributed to its design which has the following properties: 1) it considers the rich side information from the news content, 2) it considers the contexts in users' queries to better capture the sequential patterns of users' clicks, and 3) it considers news item representation of uneven categories and provide diversified recommendations.

Among baselines, the general performance of BERT4Rec is better than other baselines. BERT4Rec has also shown good performance in the general recommendation tasks through bi-directional contexts. The accuracy and tradeoff scores of BERT4Rec are also quite high.

Next, comes the performance of two-tower and Youtube-DNN methods in terms of tradeoff scores, both of which are based on two-tower architecture. The two-tower model performs better than Youtube-DNN in most scores. These methods provide the advantage of feature interactions, so a higher accuracy from these models was also expected. However, the diversity scores of these models are around 50-60%, which is not too high, probably because, they do not consider the diversity from uneven categories as we are incorporating into our approach.

The performance of LSTUR, in terms of tradeoff score, comes next. This model is constructed from the start to model news and user-specific information in their default configuration. In other models, incorporating news and user-modelling information may be mandated. LSTUR performs better in the MIND dataset, which is the default dataset (An et al., 2019) in the original paper. Compared to the accuracy scores of LSTUR, we see good diversity scores from this model (after our proposed model). The MIND dataset considers session-based information, so some diversity scores from LSTUR is expected on this dataset. LSTUR also shows good diversity in NYTimes. LSTUR also performs better than Youtube-DNN on MIND dataset.

MultiVAE is a collaborative filtering system and a non-linear probabilistic model. In terms of accuracy, the MultiVAE performs average in our experiments, and the model's diversity scores are not quite high. This is most likely because CF models (such as MultiVAE) mainly focus on the personalized recommendation strategy (Su and Khoshgoftaar, 2009), which identifies similarities between users/items to serve relevant product recommendations. As a result, we can expect the model to provide some accurate recommendations, but not too high diversity.

According to our results, the session-based recommenders i.e., SASRecF, and RepeatNet all have low-to-moderate accuracy. however, we see that these models show some higher diversity (above 50%). When it comes to providing diverse recommendations, usually the session-based recommender systems perform well (Karatzoglou and Hidasi, 2017). This is probably because of the ability of these models to recommend new and less similar items that users interacted within a session.

ENMF, a collaborative filtering method, has low-to-average accuracy and diversity scores resulting in average trade-off scores in our experiments. This suggests that we should probably extend a typical model to include more contexts, sequential information, or other recommendation models in order to provide better recommendations to users.

Overall, we see better results with the MIND dataset compared to NYTimes dataset. Due to brevity concerns and better overall results with the MIND dataset, we present the results of the subsequent experiments using the MIND dataset.

In the later experiments, we are only reporting the results on MIND dataset based on better results of all models on this dataset.

5.2 Accuracy-diversity trade-off

In this section, we showcase the accuracy-diversity trade-off achieved by our model. Figure 3 shows that as accuracy (mean F1-score) increases,

diversity (GINI) decreases, indicating an inherent relationship between these two evaluation aspects, which has also been validated in previous research (Adomavicius and Kwon, 2008; Raza and Ding, 2020; Raza and Ding, 2021a; Isufi et al., 2021).

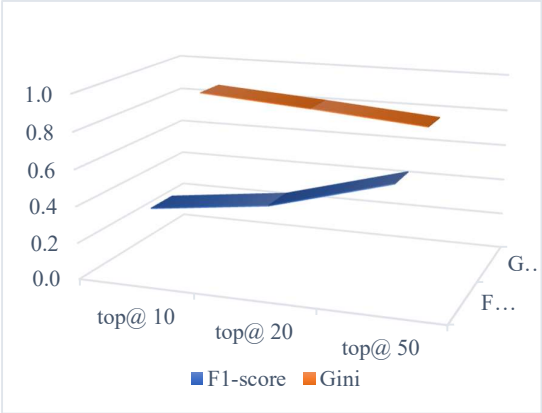

Figure 3: Accuracy and diversity trade-off of our model

As shown in Table 2, our model has the highest overall accuracy and diversity among all baselines, which is supported by a balanced trade-off score. Figure 3 also shows that as we increase the length of the recommendation list (top@ k), the accuracy of our model increases, whereas the diversity decreases. This increase in accuracy is due to the recall, which increases as the recommended items increase.

We also test the effectiveness of using different evaluation modes, which are discussed next:

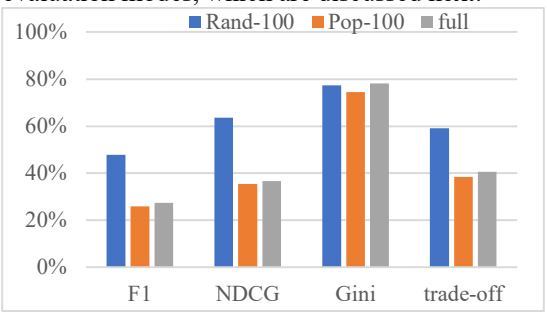

Figure 4: Model performance using different modes

random X (rand-X): randomly sample *X* negative items for each positive item in the testing set. *popularity X (pop-X):* sample *X* negative items for each positive item in the testing set based on item popularity. *full ranking*: evaluating the model on item sets. Rand-100 and pop-100 are negative sampling techniques. We report the results with *X*=100 based on best results, on the average of top @k (10, 20 and 50) and show scores in Figure 4.

Figure 4 shows that the rand-100 has the highest accuracy (F1, NDCG) score. We also see that the rand-100 mode gives us the highest diversity score (GINI-index). The pop-100 evaluation mode considers the 100 negative items for each positive item in the testing set based on item popularity and shows good results too, this is much like a collaborative filtering scenario, where some diversity is often compromised (Boim et al., 2011). The model variant with full evaluation mode shows good accuracy after the rand-100 variant. Overall, we find that the random negative sampling technique is a useful evaluation technique to achieve a balance between accuracy and diversity.

5.3 Effectiveness of side information

We also test the effectiveness of our model with and without the news side information (news body, news category, title). we consider the news body as a piece of side information in this experiment. The results are shown below in Table 3:

Metric	All features	Without news body
F1@10	36.90%	25.90%
F1@20	44.60%	21.40%
F1@50	62.10%	15.20%
Gini@10	87.40%	77.40%
Gini@20	82.70%	72.70%
Gini@50	77.40%	69.20%
tradeoff	**60.50%**	32.40%

Table 3. Model performance for side information.

As can be seen in Table 3, more content features improve the model performance compared to when we do not consider the news body. This signifies the importance of including more content-based features in the item tower that will interact with the query tower. Similarly, including more contextual features also improves model performance, we could not show this result due to limited space.

6 Conclusion

This paper proposes a two-tower architecture to model the information interaction between the two towers (query and news items). Extensive experiments on two datasets show the better performance of the proposed approach in achieving a balance between accuracy and diversity. In future, we like to conduct experiments on more real-world news data and make a deeper neural network. We like to evaluate our approach using more diversity metrics, such as normalized topic coverage and novelty. We also like to include more users' feedback like click-through rate.

References

Gediminas Adomavicius and Young Ok Kwon. 2008. Overcoming accuracy-diversity tradeoff in recommender systems: A variance-based approach. In *2008 Workshop on Information Technologies and Systems, WITS 2008*, pages 151–156. arXiv, September.

M Mehdi Afsar, Trafford Crump, Behrouz Far, M Mehdi Afsar, Trafford Crump, and Behrouz Far. 2021. Reinforcement learning based recommender systems: A survey. *arXiv preprint arXiv:2101.06286*.

Mingxiao An, Fangzhao Wu, Chuhan Wu, Kun Zhang, Zheng Liu, and Xing Xie. 2019. Neural news recommendation with long- And short-term user representations. In *ACL 2019 - 57th Annual Meeting of the Association for Computational Linguistics, Proceedings of the Conference*, pages 336–345.

Juan Pablo Bello, Elaine Chew, and Douglas Turnbull. 2008. *ISMIR 2008: Proceedings of the 9th International Conference of Music Information Retrieval*. Lulu. com.

Rubi Boim, Tova Milo, and Slava Novgorodov. 2011. Diversification and refinement in collaborative filtering recommender. *International Conference on Information and Knowledge Management, Proceedings*:739–744.

Chong Chen, Min Zhang, Yongfeng Zhang, Yiqun Liu, and Shaoping Ma. 2020. Efficient neural matrix factorization without sampling for recommendation. *ACM Transactions on Information Systems (TOIS)*, 38(2):1–28.

Laming Chen, Guoxin Zhang, and Hanning Zhou. 2018. Fast greedy map inference for determinantal point process to improve recommendation diversity. In *Proceedings of the 32nd International Conference on Neural Information Processing Systems*, pages 5627–5638.

Muthuraman Chidambaram, Yinfei Yang, Daniel Cer, Steve Yuan, Yun-Hsuan Sung, Brian Strope, and Ray Kurzweil. 2018. Learning cross-lingual sentence representations via a multi-task dual-encoder model. *arXiv preprint arXiv:1810.12836*.

Paul Covington, Jay Adams, and Emre Sargin. 2016. Deep neural networks for youtube recommendations. In *Proceedings of the 10th ACM conference on recommender systems*, pages 191–198.

Po-Sen Huang, Xiaodong He, Jianfeng Gao, Li Deng, Alex Acero, and Larry Heck. 2013. Learning deep structured semantic models for web search using clickthrough data. In *Proceedings of the 22nd ACM international conference on Information & Knowledge Management*, pages 2333–2338.

Elvin Isufi, Matteo Pocchiari, and Alan Hanjalic. 2021. Accuracy-diversity trade-off in recommender systems via graph convolutions. *Information Processing and Management*, 58(2):102459, March.

Alexandros Karatzoglou and Balázs Hidasi. 2017. Deep learning for recommender systems. In *RecSys 2017 - Proceedings of the 11th ACM Conference on Recommender Systems*, pages 396–397. June.

Dawen Liang, Rahul G Krishnan, Matthew D Hoffman, and Tony Jebara. 2018. Variational autoencoders for collaborative filtering. In *Proceedings of the 2018 World Wide Web Conference*, pages 689–698.

Lajanugen Logeswaran and Honglak Lee. 2018. An efficient framework for learning sentence representations. *arXiv preprint arXiv:1803.02893*.

Soroush Ojagh, Mohammad Reza Malek, Sara Saeedi, and Steve Liang. 2020. A location-based orientation-aware recommender system using IoT smart devices and Social Networks. *Future Generation Computer Systems*, 108:97–118.

Lijing Qin and Xiaoyan Zhu. 2013. Promoting diversity in recommendation by entropy regularizer. *IJCAI International Joint Conference on Artificial Intelligence*:2698–2704.

Shaina Raza and Chen Ding. 2020. A Regularized Model to Trade-off between Accuracy and Diversity in a News Recommender System. In *2020 IEEE International Conference on Big Data (Big Data)*, pages 551–560.

Shaina Raza and Chen Ding. 2021a. Deep Neural Network to Tradeoff between Accuracy and Diversity in a News Recommender System. In *2021 IEEE International Conference on Big Data (Big Data)*, pages 5246–5256.

Shaina Raza and Chen Ding. 2021b. News recommender system: a review of recent progress, challenges, and opportunities. *Artificial Intelligence Review*:1–52.

Pengjie Ren, Zhumin Chen, Jing Li, Zhaochun Ren, Jun Ma, and Maarten De Rijke. 2019. Repeatnet: A repeat aware neural recommendation machine for session-based recommendation. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, pages 4806–4813.

Steffen Rendle, Christoph Freudenthaler, Zeno Gantner, and Lars Schmidt-Thieme. 2012. BPR: Bayesian personalized ranking from implicit feedback. *arXiv preprint arXiv:1205.2618*.

Xiaoyuan Su and Taghi M Khoshgoftaar. 2009. A Survey of Collaborative Filtering Techniques. *Advances in Artificial Intelligence*, 2009:1–19.

Fei Sun, Jun Liu, Jian Wu, Changhua Pei, Xiao Lin, Wenwu Ou, and Peng Jiang. 2019a. BERT4Rec: Sequential Recommendation with Bidirectional Encoder Representations from Transformer. In *Proceedings of the 28th ACM International Conference on Information and Knowledge Management*, pages 1441–1450, New York, NY, USA. Association for Computing Machinery.

Wenlong Sun, Sami Khenissi, Olfa Nasraoui, and Patrick Shafto. 2019b. Debiasing the human-recommender system feedback loop in collaborative filtering. In *Companion Proceedings of The 2019 World Wide Web Conference*, pages 645–651.

Hongwei Wang, Fuzheng Zhang, Xing Xie, and Minyi Guo. 2018. DKN: Deep knowledge-aware network for news recommendation. In *The Web Conference 2018 - Proceedings of the World Wide Web Conference, WWW 2018*, pages 1835–1844, New York, New York, USA, April. Association for Computing Machinery, Inc.

Ruoxi Wang, Zhe Zhao, Xinyang Yi, Ji Yang, Derek Zhiyuan Cheng, Lichan Hong, Steve Tjoa, Jieqi Kang, Evan Ettinger, and Ed H Chi. 2019. Improving Relevance Prediction with Transfer Learning in Large-scale Retrieval Systems.

WeAreNetflix. 2018. Netflix Research: Analytics. (Accessed on 02/04/2022).

Chuhan Wu, Fangzhao Wu, Yang Yu, Tao Qi, Yongfeng Huang, and Xing Xie. 2021a. UserBERT: Contrastive User Model Pre-training.

Fangzhao Wu, Ying Qiao, Jiun-Hung Chen, Chuhan Wu, Tao Qi, Jianxun Lian, Danyang Liu, Xing Xie, Jianfeng Gao, Winnie Wu, and Ming Zhou. 2020. MIND: A Large-scale Dataset for News Recommendation. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 3597–3606, Stroudsburg, PA, USA, July. Association for Computational Linguistics.

Yao Wu, Jian Cao, and Guandong Xu. 2021b. Fairness in Recommender Systems : Evaluation Approaches Fairness in Recommender Systems : Evaluation Approaches and Assurance Strategies.

Ji Yang, Xinyang Yi, Derek Zhiyuan Cheng, Lichan Hong, Yang Li, Simon Xiaoming Wang, Taibai Xu, and Ed H Chi. 2020. Mixed negative sampling for learning two-tower neural networks in recommendations. In *Companion Proceedings of the Web Conference 2020*, pages 441–447.

Xinyang Yi, Ji Yang, Lichan Hong, Derek Zhiyuan Cheng, Lukasz Heldt, Aditee Kumthekar, Zhe Zhao, Li Wei, and Ed Chi. 2019. Sampling-bias-corrected neural modeling for large corpus item recommendations. In *RecSys 2019 - 13th ACM Conference on Recommender Systems*, pages 269–277.

Tingting Zhang, Pengpeng Zhao, Yanchi Liu, Victor S. Sheng, Jiajie Xu, Deqing Wang, Guanfeng Liu, and Xiaofang Zhou. 2019. Feature-level deeper self-attention network for sequential recommendation. In *IJCAI International Joint Conference on Artificial Intelligence*, volumes 2019-Augus, pages 4320–4326.

Guanjie Zheng, Fuzheng Zhang, Zihan Zheng, Yang Xiang, Nicholas Jing Yuan, Xing Xie, and Zhenhui Li. 2018. DRN: A deep reinforcement learning framework for news recommendation. In *The Web Conference 2018 - Proceedings of the World Wide Web Conference, WWW 2018*, pages 167–176, New York, New York, USA, April. Association for Computing Machinery, Inc.

Qiannan Zhu, Xiaofei Zhou, Zeliang Song, Jianlong Tan, and Li Guo. 2019. DAN: Deep attention neural network for news recommendation. In *33rd AAAI Conference on Artificial Intelligence, AAAI 2019, 31st Innovative Applications of Artificial Intelligence Conference, IAAI 2019 and the 9th AAAI Symposium on Educational Advances in Artificial Intelligence, EAAI 2019*, volume 33, pages 5973–5980.

Cai-Nicolas Ziegler, Sean M. McNee, Joseph A. Konstan, and Georg Lausen. 2005. Improving recommendation lists through topic diversification. In *Proceedings of the 14th international conference on World Wide Web*, number January, pages 22–32.

Dynamic Nonlinear Mixup with Distance-based Sample Selection

Shaokang Zhang[1,2], **Lei Jiang**[1,2] *, **Jianlong Tan**[1,2]
[1]Institute of Information Engineering, Chinese Academy of Sciences, Beijing, China
[2]School of Cyber Security, University of Chinese Academy of Sciences, Beijing, China
zhangshaokang@iie.ac.cn; jianglei@iie.ac.cn; tanjianlong@iie.ac.cn

Abstract

Data augmentation with mixup has shown to be effective on the NLP tasks. Although its great success, the mixup still has shortcomings. First, vanilla mixup randomly selects one sample to generate the mixup sample for a given sample. It remains unclear how to best choose the input samples for the mixup. Second, linear interpolation limits the space of synthetic data and its regularization effect. In this paper, we propose the dynamic nonlinear mixup with distance-based sample selection, which not only generates multiple sample pairs based on the distance between each sample but also enlarges the space of synthetic samples. Specifically, we compute the distance between each input data by cosine similarity and select multiple samples for a given sample. Then we use the dynamic nonlinear mixup to fuse sample pairs. It does not use a linear, scalar mixing strategy, but a nonlinear interpolation strategy, where the mixing strategy is adaptively updated for the input and label pairs. Experiments on the multiple public datasets demonstrate that dynamic nonlinear mixup outperforms state-of-the-art methods.

1 Introduction

Deep neural networks have achieved great success in NLP tasks, such as text classification (Zhang et al., 2015), machine translation (Sutskever et al., 2014), and dialogue tasks (Serban et al., 2016). These models usually require a large amount of labeled data, but labeling data is time-consuming and expensive. Data augmentation is a common technology to solve the data scarcity problem. Some of them are based on rules (Wei and Zou, 2019) and models (Edunov et al., 2018) to generate similar text. Augmented data are trained by advanced training methods (Park et al., 2021). On the other hand, mixup (Zhang et al., 2017b) trains the classifier on the synthetic data which are generated by the linear interpolation of the input and label pairs.

Recently, many variants of mixup are proposed and successfully applied to NLP tasks. It mainly includes two categories: input-level mixup (Yoon et al., 2021) and hidden-level mixup (Sun et al., 2020; Guo, 2020). Hidden-level mixup is more popular because of the discrete nature of text data and variable sequence lengths. It usually fuses the hidden vectors like embeddings or intermediate representations. While the hidden-level mixup is effective, it still has some issues. First, mixup produces the sample pairs by randomly choosing a sample for a given sample. How to select the optimal input and label pairs is unclear. Second, the space of synthetic data is limited due to its linear nature. Although nonlinear mixup (Guo, 2020) is proposed and utilizes the matrix mixing policy for the sample pairs, the matrix is fixed during the training process and it is difficult to get the appropriate parameters.

In this paper, we propose a dynamic nonlinear mixup with distance-based sample selection method to address the above problems. First, we compute the distance between each input sample by cosine similarity. Then we pick the Top-K samples with the largest distance and the Top-K samples with the smallest distance for a given sample. Second, we obtain mixed inputs by performing a dynamic mixing strategy on these sample pairs. Unlike nonlinear mixup (Guo, 2020), where the fixed matrix fusion method is applied to the sample pairs, our method uses the vector mixing policy and dynamically updates its parameters. The feature space of the generated data is further enlarged by this way. Since the dimensions of the one-hot label and vector mixing policy are different, we learn the label embedding instead of the original label. The mixing policy of labels is adaptively learned based on the mixed input. In summary, the main contributions of our work are summarized as

*Corresponding author

Proceedings of the 29th International Conference on Computational Linguistics, pages 3788–3797
October 12–17, 2022.

follows:

• We propose the dynamic nonlinear mixup, which expands the space of the generated data by the dynamic vector mixing policy. Besides, we introduce distance-based sample selection to generate more mixup samples, which improves the generalization ability of the model.

• We conducted comparative experiments on four datasets. The experimental results demonstrate that our method outperforms other state-of-the-art methods, especially in low-resource scenarios.

2 Related work

Traditional data augmentation methods are categorized into adding noise and back-translation. Easy Data Augmentation (EDA) (Wei and Zou, 2019) proposes four methods to add noise: synonym replacement, random insertion, random deletion, and random swap. It can create a large number of augmented training samples but may lose semantic information due to random deletion. Back-translation (BT) (Zhang and Zong, 2016; Edunov et al., 2018) trains target-to-source system to generate the source language. BT keeps the semantics of the original text, but the augmentation data depends on the quality of the translation model.

Another class of methods is based on the technique of interpolation. A data augmentation method mixup (Zhang et al., 2017b) is proposed and has shown good performance in the image classification. Mixup trains the classifier on the synthetic data which are generated by the linear interpolation of the input and label pairs. In NLP tasks, it usually uses hidden-level mixup. Word-Mixup and senMixup (Guo et al., 2019) are two linear interpolation methods that are implemented on the word embeddings and sentence embeddings. Sun et al. (2020) incorporate mixup to transformer-based pre-trained architecture to boost the performance of NLU tasks. Park and Caragea (2022) propose a novel mixup strategy for pre-trained language models by both the Area Under the Margin (AUM) statistic and the saliency map of each sample. Chen et al. (2020) propose a data augmentation method called Tmix for semi-supervised learning, which interpolates labeled and unlabeled samples in the hidden space to improve the performance of text classification. BatchMixup (Yin et al., 2021) utilizes interpolation strategy in a mini-batch to improve model performance on the pre-trained language model RoBERTa (Liu et al., 2019). Yoon et al. (2021) propose the Saliency-Based Span Mixup (SSMix), which generates a sentence while reserving the locality of two original texts by span-based mixing and keeping more tokens related to the prediction relying on saliency information.

The above methods just randomly select one sample for a given sample. The number of generated samples is insufficient. We choose Top-K largest distance samples and Top-K smallest distance samples according to the distance between each sample. It can generate more sample pairs for mixup. Besides, mixup uses linear interpolation which limits the space of generated data. We utilize dynamic nonlinear mixup which expands the space of the generated data. Similar to us, the nonlinear mixup (Guo, 2020) uses the fixed matrix mixing strategy. Finding an appropriate hyperparameter is difficult. In contrast, our method utilizes a vector mixing strategy that can be dynamically updated throughout the training process.

3 Approach

In this section, we first present the mixup, followed by distance-based sample selection, and the details of the dynamic nonlinear mixup. Finally, the training strategy is introduced. Figure 1 gives an overview of our method.

3.1 Mixup

Mixup is first introduced for image classification, which uses the linear interpolation to generate the synthetic data based on the input and label pairs. Specifically, a sample pair (x_i, y_i) and (x_j, y_j), where x and y denote the input samples and corresponding labels, respectively. The synthetic sample is generated as follows:

$$\begin{aligned} x_{ij} &= \lambda x_i + (1-\lambda)x_j \\ y_{ij} &= \lambda y_i + (1-\lambda)y_j \end{aligned} \tag{1}$$

where λ is the mixing-ratio and sampled from a Beta(α,α) distribution with the hyper-parameter $\alpha \in (0, \infty)$. Inspired by its great success in the image domain, this method is introduced into text classification. Unlike the image data, the text is composed of discrete and variable-length tokens. Generally, we extract the sentence embedding of the sample pairs by CNN, LSTM, or Transformer. Then, the sample pairs are linearly interpolated.

3.2 Distance-based Sample Selection

For sample (x_i, y_i), mixup randomly selects another sample (x_j, y_j) to generate the new sample

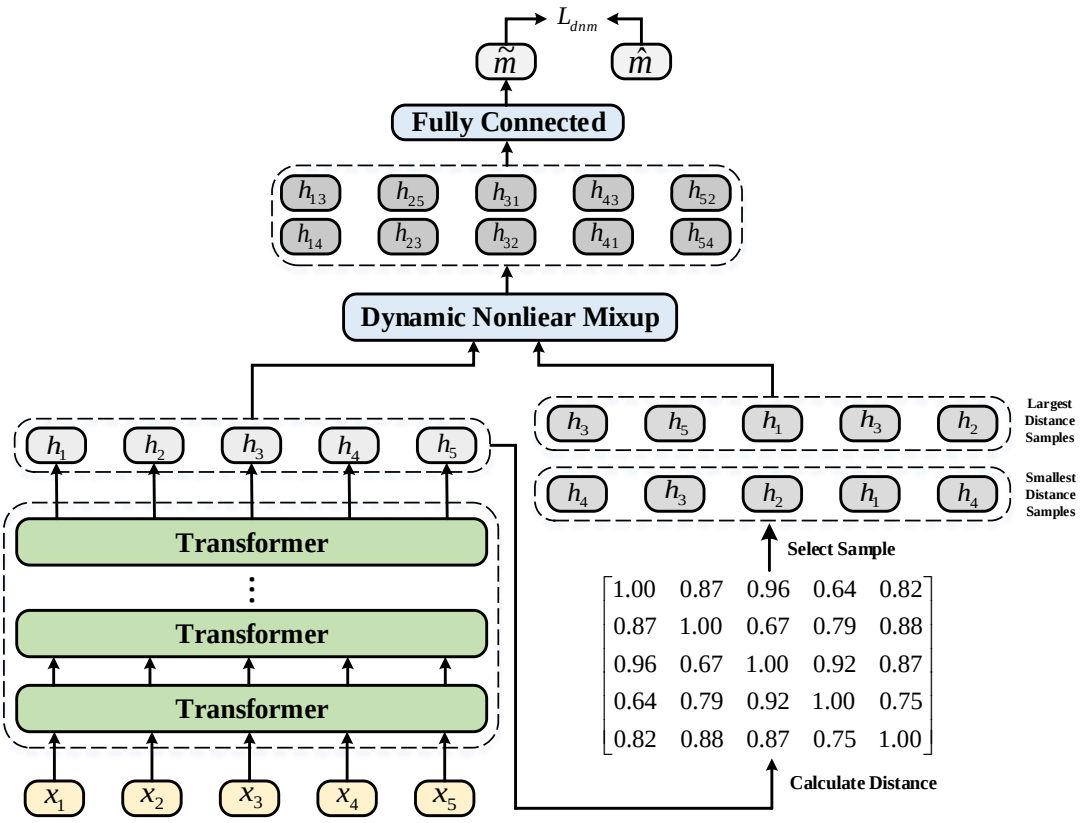

Figure 1: The architecture of the dynamic nonlinear mixup with distance-based sample selection, where $\tilde{m}$ is the predicted value and $\hat{m}$ is the ground truth. L_{dnm} is loss. For easy explanation, we select Top-1 largest distance samples and Top-1 smallest distance samples.

(x_{ij}, y_{ij}). The number of generated samples is limited because only one sample is picked. Intuitively, if the distance between a given sample and another sample is larger, the potential generation space of generated sample is larger. If only the samples with the largest distance are selected, the distribution of synthetic samples may not cover the middle area, which will reduce the performance of the model. In this paper, We propose the distance-based sample selection method which combines Top-K largest distance samples and Top-K smallest distance samples. We assume that there is training data $\mathbf{X} = \{x_i, y_i\}_{i=1}^{N}$, where N is the number of training data. The text representation $\mathbf{H} = \{h_i\}_{i=1}^{N}$ is obtained through the feature encoder. We compute the distance $L \in R^{N \times N}$ between each sample embedding representations by cosine similarity as the basis for selecting samples. For sample (x_i, y_i), we find Top-K largest distance samples $\mathbf{X}^{L_i} = \{x_j^{L_i}, y_j^{L_i}\}_{j=1}^{K}$ and Top-K smallest distance samples $\mathbf{X}^{S_i} = \{x_j^{S_i}, y_j^{S_i}\}_{j=1}^{K}$. We generate 2K mixed samples for a given sample (x_i, y_i) by interpolating the Top-K largest distance samples $\mathbf{X}^{L_i}$ and Top-K smallest distance samples $\mathbf{X}^{S_i}$.

3.3 Dynamic Nonlinear Mixup

To expand the space of generated samples, we propose the dynamic nonlinear mixup method. First, we use dynamic vector mixing policy to obtain the mixed input, where the parameters of the vector are learnable. Second, The mixing policy is a vector instead of a scalar, which cannot be applied to the label pairs because of dimension difference. We use the label embeddings to encode the one-hot labels. The label weights are updated based on the mixed input samples. In the following, we describe the details of this method successively.

3.3.1 Mixed Input

Although nonlinear mixup (Guo, 2020) expands the space of generated samples, the parameters of matrix mixing policy are constant. In this paper, we propose the dynamic nonlinear mixup which updates the mixing policy during the training process. We assume that $v_1, v_2 \in R^{1 \times d}$ denote the weight vectors, where d is the dimension. In order to make

the probability between 0 and 1, we normalize the vectors v_1 and v_2 in each dimension to get $\hat{v_1}$ and $\hat{v_2}$ by softmax. The input of dynamic nonlinear mixup is computed as follows:

$$h_{ij} = \hat{v_1} \circ h_i + \hat{v_2} \circ h_j \tag{2}$$

where h_{ij} is the input representation of the synthetic sample. $\circ$ denotes the element-wise product. The parameters of the vector are updated throughout the training process. Compared with the matrix mixing method, the space of the synthetic samples is wider and the model has better generalization ability. Note that our method mixes samples at the [CLS] hidden state representations level of the top layer of the pre-trained language model.

3.3.2 Mixed Label

Since our mixing strategy is a vector instead of a scalar, it cannot be applied to one-hot label because the vector and label have different dimensions. To overcome this problem, we use label embedding (Zhang et al., 2017a) which adopts different vectors to encode each category. $M \in R^{c \times d}$ is the label embedding matrix, where c is the number of categories. The mixed label $\hat{m}_{ij}$ is computed as follows:

$$\begin{aligned} \Phi_i &= (\hat{v_1} \circ h_i)W \\ \Phi_j &= (\hat{v_2} \circ h_j)W \\ \hat{m}_{ij} &= \hat{\Phi}_i \circ m_i + \hat{\Phi}_j \circ m_j \end{aligned} \tag{3}$$

where $W \in R^{d \times d}$ is the weight matrix. Similarly, we also normalize the weight vector Φ_i, Φ_j in each dimension to get $\hat{\Phi}_i, \hat{\Phi}_j$. m_i is the label embedding of the i-th sample. In this way, the target weight vector is generated based on the mixed input sample, which makes the model automatically obtain the appropriate synthetic sample target. Finally, the input and label of the generated samples are $(h_{ij}, \hat{m}_{ij})$.

3.4 Training strategy

We obtain the predicted d-dimensional class vector $\tilde{m}_{ij}$ by feeding the mixing input into fully connected layer:

$$\tilde{m}_{ij} = fc(h_{ij}) \tag{4}$$

We use cosine similarity to measure the correlation between the predicted vector $\tilde{m}_{ij}$ and true vector $\hat{m}_{ij}$. Since the cosine value is closer to 1, the similarity of the two vectors is higher. We use the mean squared error (MSE) to compute the loss:

$$L_{ij} = MSE(Cos(\tilde{m}_{ij}, \hat{m}_{ij}), 1) \tag{5}$$

We add up the losses of all synthetic samples to get L_{dnm}. Finally, the loss is as follows:

$$L_{dnm} = \frac{1}{N \times 2K} \sum_{i=1}^{N} \sum_{j=1}^{2K} L_{ij} \tag{6}$$

In testing phase, the cosine value is calculated between the predicted class vector $\tilde{m}_{ij}$ and the set of label embedding matrix M, and the class corresponding to the maximum value is regarded as the predicted label. Algorithm 1 describes the training procedure of our method.

Algorithm 1 Training procedure of our method

Input: Labeled data D, label embedding matrix M, weight vector v_1, v_2, weight matrix W, F is feature encoder, N is the batch size, $iters$ is the total number of iterations.

Let $t = 0$.
while t <$iters$ **do**
  t = t+1
  $Total\_Loss = 0$
  $Loss = 0$
  T_b = RandomSelect(D,N)
  Obtain the hidden layer representation $H_{T_b} = F(T_b)$
  Compute the distance L according to H_{T_b}
  for $i = 1$ to N **do**
    Select Top-K largest distance samples $\mathbf{X}^{L_i}$
    Select Top-K smallest distance samples $\mathbf{X}^{S_i}$
    Generate 2K mixed inputs using Eq. (2)
    Generate 2K mixed labels using Eq. (3)
    Compute the predicted value of each mixed input using Eq. (4)
    Compute the dynamic nonlinear mixup loss of each mixed sample using Eq. (5)
    $Loss = Loss + Loss^{L_i} + Loss^{S_i}$
  end for
  $Total\_Loss = Total\_Loss + Loss$
  Update the model parameters by minimizing $Total\_Loss$
end while

4 EXPERIMENT

4.1 Dataset Preparation

We use four datasets to evaluate our method. They are single-sentence classification tasks. **Subj** is a subjective dataset and the label is subjective or objective (Pang and Lee, 2004). **MR** is a movie review dataset with positive or negative (Pang and Lee, 2005). **SST-1** is the Stanford Sentiment Treebank with five categories of very positive, positive, neural, negative, and very negative (Socher et al., 2013). **SST-2** removes the neural label of SST-1 and is a binary classification task. Table 1 summarizes the all datasets.

Data	c	l	Train	Test	Val
Subj	2	23	8000	1000	1000
MR	2	20	8528	1068	1066
SST-2	2	19	6920	1821	872
SST-1	5	18	8544	2210	1101

Table 1: Statistics of the experimental datasets. c: number of labels. l: average sentence length.

4.2 Implementation Details

We utilize the WordPiece to split the sentences into tokens. In our experiment, we adopt the BERT_{base}(uncased) to obtain the sentence embedding. The maximum sequence length, batch size, step, and dropout is 128, 20, 4000, and 0.1, respectively. The learning rate is 2e-5. The dimension of the label embedding is 768. For each dataset, the average and standard error of the accuracy are calculated over 5 runs with different random seeds. Besides, we explore the effectiveness of our method under low-resource scenarios, and we select 50, 200, 500, 1000, and 2000 training data. For the hyper-parameter K, we select the optimal parameters on the three datasets (Table6 and Table7). Finally, We set K=5 in all our experiments. Additionally, all parameters are optimized by the adaptive momentum algorithm.

4.3 Baselines

We consider the following approaches for comparisons:

WordMixup: it applies linear interpolation at word embedding level (Guo et al., 2019).

SenMixup: it applies linear interpolation at sentence embedding level, namely the layer before the softmax layer (Guo et al., 2019).

NonlinearMixup: it uses the matrix mixing method to obtain the synthetic data (Guo, 2020).

BERT: it fine-tunes vanilla BERT on training data.

BERT+WordMixup: it fuses the the BERT-based word embeddings by the mixup.

BERT+SenMixup: it uses mixup to fuse the BERT-based sentence embeddings.

BERT+NonlinearMixup: it utilizes the BERT-based word embeddings and NonlinearMixup to train model.

We compare our method with other state-of-the-art methods on the four datasets and the experimental results are shown in Table 2, Table 3, Table 4, and Table 5. As can be seen, our method has achieved the best performance on most tasks. Besides, the results reveal several interesting observations. (1) The first is that the training data is sufficient. WordMixup, SenMixup, and NonlinearMixup use CNN or LSTM to obtain the embedding representation and achieve good performance. The classification accuracy of vanilla BERT all surpasses WordMixup, SenMixup, and NonlinearMixup. It shows that BERT model can produce better word embedding vectors. Comparing with BERT, BERT+WordMixup outperforms 0.7% on Subj dataset, 0.4% on MR dataset, 0.3% on SST-2 dataset and 0.3% on SST-1 dataset, respectively. BERT+SenMixup also outperforms BERT. Mixup is beneficial to improve the performance of large pre-trained language models. Comparing with BERT+NonlinearMixup, our method exceeds 0.7% on Subj dataset, 0.4% on MR dataset, 0.9% on SST-2 dataset and 1.0% on SST-1 dataset, respectively. The distance-based sample selection method can generate multiple sample pairs, and the dynamic nonlinear mixup expands the space of generated samples. Therefore, the generalization ability of the model is improved. (2) The second is that the training data is scarce. The classification accuracy of WordMixup, SenMixup, and NonlinearMixup dropped sharply. BERT can achieve better performance due to the pre-training on the large-scale unlabeled corpus. The classification accuracy of BERT+WordMixup and BERT+SenMixup also outperforms BERT. Compared with BERT+NonlinearMixup, our method exceeds 1.0% on Subj dataset, 2.3% on MR dataset, 1.8% on SST-2 dataset, and 2.4% on SST-1 dataset for 50 training data, respectively. The performance improvement with less data is higher than using full training data. It shows that our method is more effective in low-resource settings.

Training Data	50	200	500	1000	2000	all
WordMixup	72.4±0.6	80.6±0.7	84.1±0.6	87.4±0.5	91.6±0.4	94.3±0.6
SenMixup	73.1±0.7	81.4±0.5	84.8±0.4	88.1±0.3	92.4±0.2	95.0±0.3
NonlinearMixup	74.5±0.2	82.2±0.3	86.1±0.2	89.7±0.3	93.2±0.2	94.1±0.1
BERT	89.5±0.4	93.1±0.6	93.5±0.4	94.3±0.5	94.9±0.2	96.0±0.3
BERT+WordMixup	91.8±0.3	93.9±0.2	94.7±0.4	94.9±0.2	96.0±0.5	96.7±0.3
BERT+SenMixup	92.1±0.4	93.8±0.3	94.2±0.3	94.8±0.4	95.6±0.4	96.6±0.5
BERT+NonlinearMixup	92.4±0.5	94.3±0.4	94.9±0.4	94.6±0.2	95.9±0.3	97.3±0.4
Ours	**93.4±0.2**	**95.2±0.5**	**95.6±0.4**	**95.8±0.3**	**96.9±0.4**	**98.0±0.3**

Table 2: Classification accuracy (%) on the Subj dataset.

Training Data	50	200	500	1000	2000	all
WordMixup	65.7±0.6	68.7±0.6	70.5±0.4	72.2±0.7	76.5±0.3	79.7±0.5
SenMixup	66.3±0.4	67.9±0.5	69.4±0.3	73.1±0.5	77.1±0.5	80.3±0.6
NonlinearMixup	67.4±0.2	69.5±0.3	71.1±0.1	74.6±0.3	78.2±0.5	83.4±0.4
BERT	76.6±0.5	81.0±0.5	82.2±0.6	82.9±0.5	84.7±0.4	87.8±0.3
BERT+WordMixup	77.2±0.5	81.3±0.4	82.3±0.4	83.9±0.3	85.0±0.5	88.2±0.4
BERT+SenMixup	77.0±0.4	81.4±0.3	82.6±0.3	83.7±0.5	85.4±0.5	88.1±0.3
BERT+NonlinearMixup	77.3±0.5	81.7±0.6	83.0±0.4	84.3±0.4	85.6±0.3	88.8±0.5
Ours	**79.6±0.3**	**82.6±0.4**	**83.9±0.4**	**85.2±0.4**	**86.4±0.3**	**89.2±0.4**

Table 3: Classification accuracy (%) on the MR dataset.

Training Data	50	200	500	1000	2000	all
WordMixup	66.3±0.6	73.2±0.7	79.4±0.4	78.7±0.5	83.2±0.6	87.1±0.3
SenMixup	67.2±0.7	74.0±0.5	78.7±0.3	77.9±0.6	83.4±0.3	87.2±0.4
NonlinearMixup	68.1±0.3	75.2±0.4	80.2±0.5	79.3±0.5	84.3±0.5	88.6±0.3
BERT	82.6±0.5	85.8±0.3	86.6±0.2	87.9±0.3	89.7±0.4	90.7±0.3
BERT+WordMixup	83.1±0.4	86.4±0.5	87.6±0.3	88.6±0.4	89.6±0.4	91.0±0.3
BERT+SenMixup	83.0±0.5	86.1±0.4	87.8±0.3	88.4±0.2	89.9±0.3	91.1±0.2
BERT+NonlinearMixup	83.3±0.5	86.7±0.6	**88.4±0.4**	88.9±0.5	90.9±0.3	91.6±0.6
Ours	**85.1±0.3**	**87.7±0.4**	88.1±0.3	**89.5±0.2**	**91.4±0.3**	**92.5±0.2**

Table 4: Classification accuracy (%) on the SST-2 dataset.

Training Data	50	200	500	1000	2000	all
WordMixup	26.3±0.6	32.4±0.7	35.4±0.5	38.3±0.6	42.5±0.8	48.2±1.0
SenMixup	27.1±0.7	33.0±0.4	36.2±0.5	37.8±0.5	41.7±0.3	48.6±0.2
NonlinearMixup	27.8±0.3	33.8±0.3	36.7±0.5	38.9±0.4	43.5±0.2	49.3±0.4
BERT	33.3±0.5	39.7±0.4	45.6±0.3	46.1±0.4	48.8±0.2	52.6±0.3
BERT+WordMixup	33.7±0.4	39.3±0.5	45.1±0.3	48.0±0.6	49.4±0.4	52.9±0.2
BERT+SenMixup	33.8±0.4	38.8±0.3	44.8±0.4	47.4±0.2	49.3±0.3	53.0±0.2
BERT+NonlinearMixup	34.9±0.3	40.6±0.4	46.6±0.5	47.5±0.4	50.3±0.4	53.4±0.3
Ours	**37.3±0.3**	**41.8±0.2**	**47.7±0.5**	**48.6±0.2**	**51.4±0.3**	**54.4±0.2**

Table 5: Classification accuracy (%) on the SST-1 dataset.

4.4 Regularization Effect

We plot the training set loss and testing set accuracy using all training data on Subj and SST-1 datasets in Figure 2. Compared with BERT+NonlinearMixup, our method can reduce the training loss faster and have higher classification accuracy. Besides, the classification accuracy has not dropped even through a long training time. It shows that our

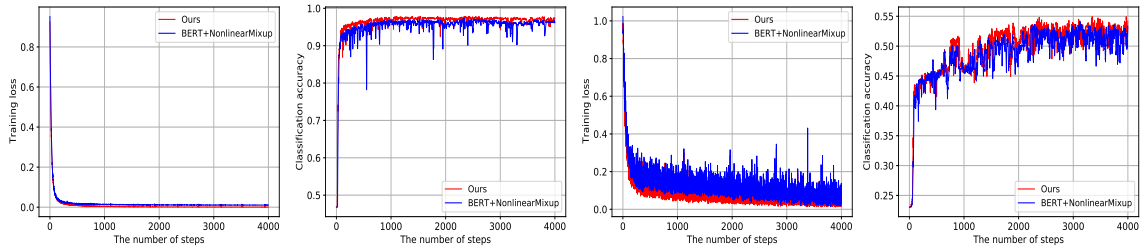

(a) (b) (c) (d)

Figure 2: Training set loss and testing set accuracy on Subj dataset (a) (b) and SST-1 dataset (c) (d) when using all training data.

method can effectively prevent overfitting because of more training samples and larger generated sample space.

4.5 Selection of Top-K

We investigate the effects of the K value on Subj, MR, and SST-1 datasets using 50 and all training data in Table 6 and Table 7. We vary the K value with 1, 3, 5, 7, 9, respectively. We find that the classification accuracy increases first and then decreases as K value increases. The best performance is achieved when K = 5 on most tasks. Compared with all training data, the improvement of model performance is more obvious in 50 training data. Moreover, as K value increases, the generated samples cannot improve the performance of the model due to overfitting. It is necessary to find an appropriate parameter K.

Top-K	Subj	MR	SST-1
1	92.5±0.4	77.2±0.3	35.2±0.3
3	91.7±0.3	78.3±0.2	36.1±0.4
5	**93.4±0.2**	**79.6±0.3**	**37.3±0.3**
7	93.1±0.3	78.8±0.3	36.9±0.5
9	92.9±0.2	79.2±0.4	36.6±0.4

Table 6: Classification accuracy under different K values on Subj, MR, and SST-1 datasets when using 50 training data.

Top-K	Subj	MR	SST-1
1	97.2±0.4	88.7±0.4	53.3±0.3
3	97.6±0.3	88.5±0.5	53.8±0.4
5	**98.0±0.3**	89.2±0.4	**54.4±0.2**
7	97.9±0.5	**89.4±0.2**	54.0±0.3
9	97.4±0.4	89.1±0.5	53.5±0.4

Table 7: Classification accuracy under different K values on Subj, MR, and SST-1 datasets when using all training data.

4.6 Effects of Input Weights and Output Weights

We construct four different ways to observe the effects of weights on Subj, MR, and SST-1 datasets using 50 and all training data in Table 8 and Table 9: tuned input weights (TIW), tuned output weights (TOW), fixed input weights (FIW), and fixed output weights (FOW). The input weights are shown in Eq. (2), and the output weights are shown in Eq. (3). The experimental results show that when the input weights or output weight is fixed, the prediction accuracy of our method is reduced. Compared with FIW+FOW, TIW+TOW exceeds 2.3% on Subj dataset, 2.8% on MR dataset, and 3.1% on SST-1 dataset for 50 training data, respectively. Compared with FIW+FOW, TIW+TOW exceeds 1.5% on Subj dataset, 1.4% on MR dataset, and 2.1% on SST-1 dataset for all training data, respectively. It shows that our method is more suitable for low resource situations.

Method	Subj	MR	SST-1
TIW+TOW	**93.4±0.2**	**79.6±0.3**	**37.3±0.3**
TIW+FOW	91.7±0.3	77.9±0.2	35.1±0.2
FIW+TOW	92.1±0.2	78.2±0.4	34.9±0.3
FIW+FOW	91.1±0.1	76.8±0.3	34.2±0.2

Table 8: Effects of input weights and output weights on Subj, MR, and SST-1 datasets when using 50 training data.

Method	Subj	MR	SST-1
TIW+TOW	**98.0±0.3**	**89.2±0.4**	**54.4±0.2**
TIW+FOW	96.7±0.3	88.4±0.3	53.1±0.3
FIW+TOW	96.5±0.4	88.1±0.2	53.0±0.4
FIW+FOW	96.2±0.2	87.8±0.3	52.3±0.3

Table 9: Effects of input weights and output weights on Subj, MR, and SST-1 datasets when using all training data.

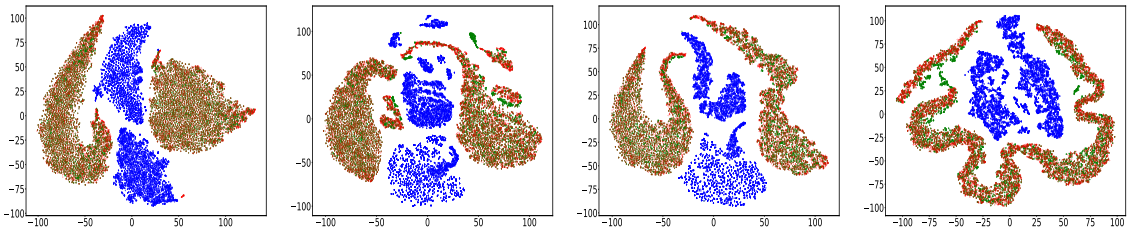

(a) Subj (b) MR (c) SST-2 (d) SST-1

Figure 3: The t-SNE visualization of original samples and generated samples. The red, green and blue points denote original samples, the generated samples with the Top-k largest distance and the generated samples with the Top-k smallest distance.

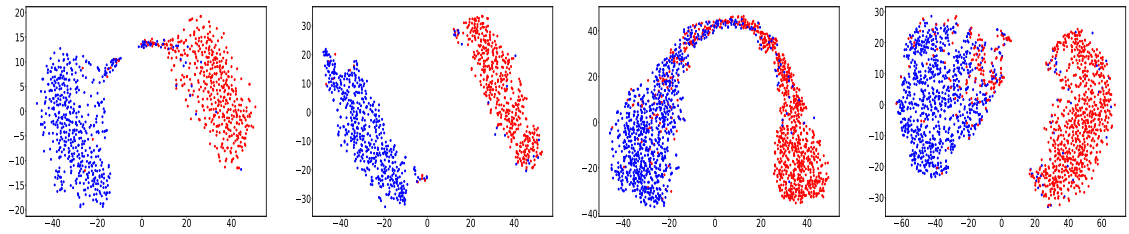

(a) BERT (b) Ours (c) BERT (d) Ours

Figure 4: The t-SNE visualization of the Subj task (a) (b) and SST-2 task (c) (d). The red and blue points denote different labels.

4.7 Ablation Studies

To analyze the effect of distance-based sample selection (DSS), largest distance samples (LDS), and smallest distance samples (SDS), and dynamic nonlinear mixup (DNM), we conduct the ablation experiments using 50 and all training data in Table 10 and Table 11. The w/o DSS randomly selects a sample for a given sample. Our method can improve the classification accuracy of the model by generating more sample pairs. Furthermore, we study the contributions of the largest distance samples and smallest distance samples. We can find the samples with the largest distance has a greater impact than samples with the smallest distance. The w/o DNM follows the original mixing strategy to fuse the sentence embeddings. When we interpolate sample pairs by mixup, the classification accuracy of w/o DNM degrades on Subj, MR, and SST-1 datasets. It shows that dynamic nonlinear mixing strategy can expand the space of generated samples.

Method	Subj	MR	SST-1
Ours	**93.4±0.2**	**79.6±0.3**	**37.3±0.3**
w/o DSS	91.3±0.5	78.3±0.3	36.1±0.3
w/o LDS	90.5±0.3	76.4±0.4	32.4±0.3
w/o SDS	91.6±0.4	77.9±0.3	35.2±0.4
w/o DNM	92.4±0.4	78.5±0.2	35.9±0.5

Table 10: Results of ablation on Subj, MR, and SST-1 datasets when using 50 training data.

Method	Subj	MR	SST-1
Ours	**98.0±0.3**	**89.2±0.4**	**54.4±0.2**
w/o DSS	97.4±0.4	88.7±0.2	53.5±0.4
w/o LDS	93.9±0.2	86.5±0.4	50.5±0.5
w/o SDS	97.1±0.3	88.2±0.3	53.1±0.3
w/o DNM	97.3±0.3	88.3±0.2	53.6±0.4

Table 11: Results of ablation on Subj, MR, and SST-1 datasets when using all training data.

4.8 Feature Visualization

To prove the idea in Section 3.2, we visualize the original samples and generated samples by the t-SNE (Van der Maaten and Hinton, 2008) in Figure 3. The distribution of the generated samples with the Top-k largest distance is similar to the original samples, and the generated samples with the Top-k smallest distance are mainly distributed in the middle area of the original samples. Besides, we also select test samples and visualize the feature of the last layer in Figure 4. We perform the

visualization on Subj and SST-2 datasets using all training data. The vanilla BERT mixes samples with different labels (Figure 4a and Figure 4c). Our method can better separate samples with different labels (Figure 4b and Figure 4d). The boundary of text classification is clear. It proves that our method can improve the performance of the model.

5 Conclusion

In this paper, we propose the dynamic nonlinear mixup with distance-based sample selection method to enhance the performance of pre-trained language models. First, we introduce distance-based sample selection to choose the Top-K largest distance samples and Top-K smallest distance samples for a given sample. More generated samples improve the generalization ability of the model. Second, we utilize the dynamic nonlinear mixing policy on the input sample pairs to enlarge the space of the synthetic samples. The mixed labels are constructed through the learned label embeddings and mixed input so that the mixed labels are updated adaptively. Experiments on the Subj, MR, SST-2, and SST-1 datasets demonstrate that our method outperforms the state-of-the-art methods. For future work, we plan to further enhance performance by exploring different weighting schemes for origin samples and augmented samples.

References

Jiaao Chen, Zichao Yang, and Diyi Yang. 2020. Mixtext: Linguistically-informed interpolation of hidden space for semi-supervised text classification. *arXiv preprint arXiv:2004.12239*.

Sergey Edunov, Myle Ott, Michael Auli, and David Grangier. 2018. Understanding back-translation at scale. *arXiv preprint arXiv:1808.09381*.

Hongyu Guo. 2020. Nonlinear mixup: Out-of-manifold data augmentation for text classification. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pages 4044–4051.

Hongyu Guo, Yongyi Mao, and Richong Zhang. 2019. Augmenting data with mixup for sentence classification: An empirical study. *arXiv preprint arXiv:1905.08941*.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Roberta: A robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692*.

Bo Pang and Lillian Lee. 2004. A sentimental education: Sentiment analysis using subjectivity summarization based on minimum cuts. *arXiv preprint cs/0409058*.

Bo Pang and Lillian Lee. 2005. Seeing stars: Exploiting class relationships for sentiment categorization with respect to rating scales. *arXiv preprint cs/0506075*.

Jungsoo Park, Gyuwan Kim, and Jaewoo Kang. 2021. Consistency training with virtual adversarial discrete perturbation. *arXiv preprint arXiv:2104.07284*.

Seo Yeon Park and Cornelia Caragea. 2022. On the calibration of pre-trained language models using mixup guided by area under the margin and saliency. *arXiv preprint arXiv:2203.07559*.

Iulian Serban, Alessandro Sordoni, Yoshua Bengio, Aaron Courville, and Joelle Pineau. 2016. Building end-to-end dialogue systems using generative hierarchical neural network models. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 30.

Richard Socher, Alex Perelygin, Jean Wu, Jason Chuang, Christopher D Manning, Andrew Y Ng, and Christopher Potts. 2013. Recursive deep models for semantic compositionality over a sentiment treebank. In *Proceedings of the 2013 conference on empirical methods in natural language processing*, pages 1631–1642.

Lichao Sun, Congying Xia, Wenpeng Yin, Tingting Liang, Philip S Yu, and Lifang He. 2020. Mixup-transformer: dynamic data augmentation for nlp tasks. *arXiv preprint arXiv:2010.02394*.

Ilya Sutskever, Oriol Vinyals, and Quoc V Le. 2014. Sequence to sequence learning with neural networks. *Advances in neural information processing systems*, 27.

Laurens Van der Maaten and Geoffrey Hinton. 2008. Visualizing data using t-sne. *Journal of machine learning research*, 9(11).

Jason Wei and Kai Zou. 2019. Eda: Easy data augmentation techniques for boosting performance on text classification tasks. *arXiv preprint arXiv:1901.11196*.

Wenpeng Yin, Huan Wang, Jin Qu, and Caiming Xiong. 2021. Batchmixup: Improving training by interpolating hidden states of the entire mini-batch. In *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*, pages 4908–4912.

Soyoung Yoon, Gyuwan Kim, and Kyumin Park. 2021. Ssmix: Saliency-based span mixup for text classification. *arXiv preprint arXiv:2106.08062*.

Honglun Zhang, Liqiang Xiao, Wenqing Chen, Yongkun Wang, and Yaohui Jin. 2017a. Multi-task label embedding for text classification. *arXiv preprint arXiv:1710.07210*.

Hongyi Zhang, Moustapha Cisse, Yann N Dauphin, and David Lopez-Paz. 2017b. mixup: Beyond empirical risk minimization. *arXiv preprint arXiv:1710.09412*.

Jiajun Zhang and Chengqing Zong. 2016. Exploiting source-side monolingual data in neural machine translation. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 1535–1545.

Xiang Zhang, Junbo Zhao, and Yann LeCun. 2015. Character-level convolutional networks for text classification. *Advances in neural information processing systems*, 28.

MultiCoNER: A Large-scale Multilingual dataset for Complex Named Entity Recognition

Shervin Malmasi, Anjie Fang*, Besnik Fetahu*, Sudipta Kar*, Oleg Rokhlenko
Amazon.com,
Seattle, WA, USA
{malmasi,njfn,besnikf,sudipkar,olegro}@amazon.com

Abstract

We present MULTICONER, a large multilingual dataset for Named Entity Recognition that covers 3 domains (Wiki sentences, questions, and search queries) across 11 languages, as well as multilingual and code-mixing subsets. This dataset is designed to represent contemporary challenges in NER, including low-context scenarios (short and uncased text), syntactically complex entities like movie titles, and long-tail entity distributions. The 26M token dataset is compiled from public resources using techniques such as heuristic-based sentence sampling, template extraction and slotting, and machine translation. We applied two NER models on our dataset: a baseline XLM-RoBERTa model, and a state-of-the-art GEMNET model that leverages gazetteers. The baseline achieves moderate performance (macro-F1=54%), highlighting the difficulty of our data. GEMNET, which uses gazetteers, improvement significantly (average improvement of macro-F1=+30%). MULTICONER poses challenges even for large pre-trained language models, and we believe that it can help further research in building robust NER systems.

MULTICONER is publicly available,[1] and we hope that this resource will help advance research in various aspects of NER.

1 Introduction

Named Entity Recognition (NER) is a core task in Natural Language Processing which involves identifying entities in text, and recognizing their types (e.g., classifying entities as a person or location). Recently, the development of Transformer-based NER approach have results in new state-of-the-art (SOTA) results on well-known benchmark datasets like CoNLL03 and OntoNotes (Devlin et al., 2019). Despite these strong results, there remain a number of practical challenges that may not be represented by these existing datasets.

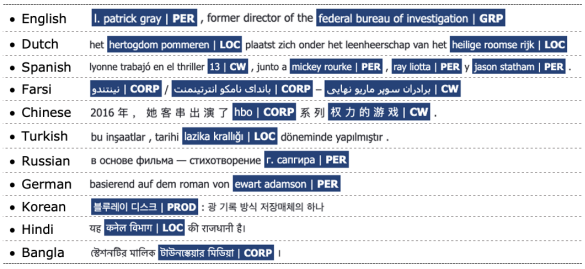

Figure 1: Some examples for all the languages incorporated in MULTICONER.

As noted by Augenstein et al. (2017), increasingly higher scores on these datasets are driven by several factors:

- Well-formed data, with punctuation and capitalized nouns, makes the NER task easier, providing the model with additional contextual cues.
- Texts from articles and the news domain contain long sentences with rich context around entities, providing valuable signals about the boundaries and types of entities.
- Data from the news domain[2] contains "easy" entities such as country, city, and person names, allowing pre-trained models to perform well due to their existing knowledge of such tokens.
- Memorization effects, due to large overlap of entities between the train and test sets also increases performance.

Accordingly, models trained on existing benchmark datasets such as CoNLL03 tend to perform significantly worse on unseen entities or noisy text (Meng et al., 2021).

*These authors contributed equally in this work.

[1] https://registry.opendata.aws/multiconer/

[2] e.g. CoNLL03 (Sang and De Meulder, 2003)

Proceedings of the 29th International Conference on Computational Linguistics, pages 3798–3809
October 12–17, 2022.

1.1 Contemporary Challenges in NER

There are many challenging scenarios for NER outside of the news domain. We categorize the challenges typically encountered in NER according to several dimensions: (i) available context around entities, (ii) named entity surface form complexity, (iii) frequency distribution of named entity types, (iv) dealing with multilingual and code-mixed textual snippets, and (v) out-of-domain adaptability.

Context News domain text often features long sentences that reference multiple entities. In other applications, such as voice input to digital assistants or search queries issued by users, the input length is constrained and the context is less informative. Datasets featuring such low context settings are needed to assess model performance.

Additionally, capitalization and punctuation features are large drivers of success in NER (Mayhew et al., 2019). However, inputs such as search queries from users, or voice commands transcribed using ASR, lack these surface features. To understand model performance in such cases, an uncased dataset is needed.

Entity Complexity Datasets in existing NER benchmarks are often dominated by entities representing persons, locations, and organizations. Such entities are often composed of proper nouns, or have names with simple syntactic structures. However, not all entities are so simple in structure: some entity types (e.g., creative works) can be linguistically complex. They can be complex noun phrases (`Eternal Sunshine of the Spotless Mind`), gerunds (`Saving Private Ryan`), infinitives (`To Kill a Mockingbird`), or full clauses (`Mr. Smith Goes to Washington`). Syntactic parsing of such nouns is hard, and most current parsers and NER systems fail to recognize them. The top system from WNUT 2017 achieved 8% recall for creative work entities (Aguilar et al., 2017). Corpora including such challenging entities are needed for evaluation of model performance in such cases.

Entity Distributions In many domains, entities have a large long-tail distribution, with millions of possible values (e.g., location names). This makes it hard to build representative training data, as a small dataset can only cover a portion of the potentially infinite entity space. A very large test set is required for comprehensive evaluation.

Furthermore, some domains have entity spaces that are continuously growing. While all entity types are open classes (i.e., new ones are added), some groups have a faster growth rate, e.g., new books, songs, and movies are released weekly. Assessing true model generalization requires test sets with many entities that are unseen in the training set, in order to mimic an open-world setting.

Multilinguality and Code-Mixing The recent success of multilingual models have greatly boosted task performance in languages with fewer resources, by leveraging transfer learning from high resource languages. However, there are limits to what can be achieved with cross-lingual transfer, and training data in additional languages is necessary to make progress in this field. The availability of a NER dataset that addresses all the above challenges across many languages will enable new research directions in multilingual model evaluation, as well as for few- and zero-shot cross-lingual transfer scenarios.

Code-mixing, where entities and the main text may be in different languages, is another related research area in multilingual NER where additional resources can help. Code-mixed data is increasing online, especially in social media platform where multiple languages are used in a single post. Such a dataset is needed to study this area and evaluate truly multilingual NER systems.

Domain Adaptation A robust NER model is expected to perform effectively in several domains, such as well written sentences, questions, web search queries, etc. While well written sentences can be easy for NER, shorter questions and queries can be challenging. It is important to study how to adapt existing NER into newly emerging domains. However, most of the existing NER benchmarks only focus on data in a single domain limiting its usage for studying domain adaptation.

MULTICONER We address the aforementioned challenges by presenting MULTICONER, a multilingual dataset that features a large number of entities (including complex ones) in three distinct domains that represent different challenges. Some key facts about MULTICONER:

- Textual snippets in MULTICONER are low in context, allowing to assess NER model's capability in detecting ambiguous named entities;

Source	Gold Sentence
English – Wiki	[heat vision and jack]$_{CW}$, a 1999 television pilot
Spanish – Wiki	reina consorte de [escocia]$_{LOC}$ como esposa de [jacobo v]$_{PER}$.
English – QA	when was the [nokia 2.2]$_{PROD}$ released
English – Search Query	cast of [dr. devil and mr. hare]$_{CW}$
Russian – QA	где было [королевы крика]$_{CW}$ снято
Code-Mixed (KO/EN)	[symphony no. 7 in e major]$_{CW}$ 란 무엇입니까?

Table 1: Examples from MULTICONER: entities are in brackets, followed by their type.

- Named entities contain a highly diverse distribution from simple `Location` (`LOC`) to highly complex entities `Creative Work` (`CW`);
- Using open knowledge bases such as Wikipedia and Wikidata, we generate textual snippets that contain highly diverse named entities;
- Through a combination of localized versions of Wikipedia and Wikidata, and as well as automated text translation approaches, we generate NER data for 11 languages and 3 domains that can be used to test cross-lingual and cross domain NER performance. Some examples are presented in Figure 1.

2 MULTICONER Dataset Overview

The MULTICONER dataset was designed in order to address the NER challenges described in §1.1. It represents 3 domains (wiki sentences, questions, and search queries) and includes 11 languages, including multilingual and code-mixed subsets. MULTICONER was collected and released as part of the SemEval 2022 Task#11, serving more than 236 participants across all the different languages (Malmasi et al., 2022).

2.1 NER Taxonomy

MULTICONER leverages the WNUT 2017 (Derczynski et al., 2017) taxonomy entity types, which defines the following NER tag-set with 6 classes:

- `PERSON` (`PER` for short, names of people)
- `LOCATION` (`LOC`, locations/physical facilities)
- `CORPORATION` (`CORP`, corporations/businesses)
- `GROUPS` (`GRP`, all other groups)
- `PRODUCT` (`PROD`, consumer products)
- `CREATIVE-WORK` (`CW`, movie/song/book titles)

This taxonomy allows us to capture a wide array of entities, including those with more complex entity structure, such as creative works.

2.2 Languages and Subsets

Bangla	(BN)	Hindi	(HI)	German	(DE)
Chinese	(ZH)	Korean	(KO)	Turkish	(TR)
Dutch	(NL)	Russian	(RU)	Farsi	(FA)
English	(EN)	Spanish	(ES)		

Table 2: The languages included in MULTICONER, along with their 2-letter codes.

There are 11 languages included in MULTICONER (cf. Table 2). These languages were chosen to include a diverse typology of languages and writing systems, and range from well-resourced (e.g., `EN`) to low-resourced ones (e.g., `FA`).

MULTICONER contains 13 different subsets: 11 monolingual subsets for the above languages, a multilingual subset (denoted as `MULTI`), and a code-mixed one (`MIX`).

Monolingual Subsets Each of the 11 languages has their own subset with data from all domains.

Multilingual Subset This contains randomly sampled data from all the languages mixed into a single subset. This subset is designed for evaluating multilingual models, and should ideally be used under the assumption that the language for each sentence is unknown. A more detailed description of the multilingual train/dev/test set construction is provided in §3.

Code-mixing Subset This subset contains code-mixed instances, where the entity is from one language and the rest of the text is written in another language. Like the multilingual subset, this subset should also be used under the assumption that the languages present in an instance are unknown.

2.3 Domains and Data Sources

The three domains[3] of MULTICONER are listed below. Details on the construction of the different subsets are provided in §3.

Wikipedia Sentences (LOWNER) This subset of MULTICONER, which we call Low-context Wikipedia NER (LOWNER) set, is built by sampling sentences from Wikipedia and using heuristics to identify ones that represent the NER challenges we target. More details on how we select sentence from Wikipedia are provided in §3.2.

[3]Domain can have ambiguous interpretations (van der Wees et al., 2015), in our case it reflects a combination of provenance and text genre.

Questions (MSQ-NER) The MSQ-NER subset of MULTICONER represents NER in the QA domain. It is composed of a set of natural language questions, based on the MS-MARCO QnA corpus (V2.1) (Bajaj et al., 2016).

Search Queries (ORCAS-NER) The ORCAS-NER subset of MULTICONER represents the search query domain. To build this data, we utilize 10 million Bing user queries from the ORCAS dataset (Craswell et al., 2020).

2.4 Data Splits

To ensure that obtained NER results on this dataset are *reproducible*, we create three predefined sets for training, development and testing. The entity classes within each set are approximately uniformly distributed. Table 3 shows detailed statistics for each of the 13 subtasks and data splits.

Training Data For the training data split, we limit the size to be $15,300$ sentences. The number of instances was chosen to be comparable to well-known NER datasets such as CoNLL03 (Sang and De Meulder, 2003). Majority of the instances come from the LOWNER domain, with a small sample of 100 instances from the MSQ-NER and ORCAS-NER domains. These instances represent out-of-domain adaptation. The out-of-domain instances are limited in order to be able to realistically assess the out-of-domain performance of models trained on the MULTICONER dataset.

Note that in the case of the Multilingual subset, the training split contains all the instances from the individual language splits. For Code-Mixed on the other hand, we constructed only a small training split, in this way we allow for NER models to better model this task, rather than having abundance of code-mixed instances. The Code-Mixed instances are constructed by first sampling instances from the language specific training splits, and then at random replacing the original entity surface forms into their corresponding surface forms in another language present in our dataset.

Development Data We randomly sample 800 instances per subset from the LOWNER domain (5% of the training set size), a reasonable amount of data for assessing model generalizability.

The only difference in the development data is for the Multilingual and Code-Mixed subtastks, where the development splits are constructed similar as for the training splits (see above).

Test Data Finally, the testing set represents the remaining instances that are not part of the training or development set. To avoid exceedingly large test sets, we limit the number of instances in the test set to be around 215k sentences at most (cf. Table 3). The only exception is for the Multilingual and Code-Mixed subsets. The Multilingual test split was generated from the language specific test splits, and was downsampled to contain only 471k instances. On the other hand, for the Code-Mixed subset, we sample test sentences from the language specific test splits, and replace the original entity surface forms with the surface form of the entity in another language, picked at random.

The larger size of test sets are done for two reasons: (1) to assess the generalizability of models on unseen and complex entities; and (2) assess cross-domain adaptation performance. Table 4 provides a breakdown of the number of instances for the different domains across the different subtasks.

Finally, the overlap of NEs between the test and train set is fairly small, with an overlap of 5.6% across all NE classes and languages. Such a small NE overlap ensures that the test dataset is suitable for measuring NER model generalization.

2.5 License, Availability, and File Format

The dataset is released under a CC BY-SA 4.0 license, which allows adapting the data. Details about the license are available on the Creative Commons website.[4] The data is distributed using the commonly used BIO tagging scheme in CoNLL03 format (Sang and De Meulder, 2003). The complete dataset is available for download.[5]

3 Dataset Construction

In this section, we provide a detailed description of the methods used to generate our dataset.

3.1 Entity Gazetteer Data

We require a large, multilingual, and reliable source of known entities for generating our dataset. To this end we leverage the Wikidata to obtain entity information. Instead of using all entities in the KB, or collecting entities from web sources (Khashabi et al., 2018), we instead focus on entities that map to our taxonomy.

[4] https://creativecommons.org/licenses/by-sa/4.0

[5] https://registry.opendata.aws/multiconer/

class	split	EN	DE	ES	RU	NL	KO	FA	ZH	HI	TR	BN	MULTI	MIX
	train	5,397	5,288	4,706	3,683	4,408	4,536	4,270	2,225	2,418	4,414	2,606	43,951	296
PER	dev	290	296	247	192	212	267	201	129	133	231	144	2,342	96
	test	55,682	55,757	51,497	44,687	49,042	39,237	35,140	26,382	25,351	26,876	24,601	111,346	19,313
	train	4,799	4,778	4,968	4,219	5,529	6,299	5,683	6,986	2,614	5,804	2,351	54,030	325
LOC	dev	234	296	274	221	299	323	324	378	131	351	101	2,932	108
	test	59,082	59,231	58,742	54,945	63,317	52,573	45,043	43,289	31,546	34,609	29,628	141,013	23,111
	train	3,571	3,509	3,226	2,976	3,306	3,530	3,199	713	2,843	3,568	2,405	32,846	248
GRP	dev	190	160	168	151	163	183	164	26	148	167	118	1,638	75
	test	41,156	40,689	38,395	37,621	39,255	31,423	27,487	18,983	22,136	21,951	19,177	77,328	16,357
	train	3,111	3,083	2,898	2,817	2,813	3,313	2,991	3,805	2,700	2,761	2,598	32,890	294
CORP	dev	193	165	141	159	163	156	160	192	134	148	127	1,738	112
	test	37,435	37,686	36,769	35,725	35,998	30,417	27,091	25,758	21,713	21,137	20,066	75,764	18,478
	train	3,752	3,507	3,690	3,224	3,340	3,883	3,693	5,248	2,304	3,574	2,157	38,372	298
CW	dev	176	189	192	168	182	196	207	282	113	190	120	2,015	102
	test	42,781	42,133	43,563	39,947	41,366	33,880	30,822	30,713	21,781	23,408	21,280	89,273	20,313
	train	2,923	2,961	3,040	2,921	2,935	3,082	2,955	4,854	3,077	3,184	3,188	35,120	316
PROD	dev	147	133	154	151	138	177	157	274	169	158	190	1,848	117
	test	36,786	36,483	36,782	36,533	36,964	29,751	26,590	28,058	22,393	21,388	20,878	75,871	20,255
	train	15,300	15,300	15,300	15,300	15,300	15,300	15,300	15,300	15,300	15,300	15,300	168,300	1,500
#instances	dev	800	800	800	800	800	800	800	800	800	800	800	8,800	500
	test	217,818	217,824	217,887	217,501	217,337	178,249	165,702	151,661	141,565	136,935	133,119	471,911	100,000

Table 3: MULTICONER dataset statistics for the different languages for the train/dev/test splits. For each NER class we show the total number of entity instances per class on the different data splits. The bottom three rows show the total number of sentences for each language.

lang	LOWNER	ORCAS-NER	MSQ-NER	Total
EN	100,000	100,000	17,818	217,818
DE	100,000	100,000	17,824	217,824
ES	100,000	100,000	17,887	217,887
RU	100,000	100,000	17,501	217,501
NL	100,000	100,000	17,337	217,337
KO	60,425	100,000	17,824	178,249
FA	48,792	100,000	16,910	165,702
ZH	33,776	100,000	17,885	151,661
HI	24,807	100,000	16,758	141,565
TR	19,581	100,000	17,354	136,935
BN	15,698	100,000	17,421	133,119
MULTI	200,000	200,000	71,911	471,911
MIX	42,168	15,667	42,165	100,000

Table 4: Test data statistics per domain.

We map Wikidata entities to our NER taxonomy (§2.1). This is done by traversing Wikidata's class and instance relations, and manually mapping them to our NER classes, e.g., Wikidata's `human` class maps to `PER` in our taxonomy, `song` to `CW`, and so on. Alternative names (aliases) for entities are included. The distribution of these entities is shown in Table 9 in §A.1.

3.2 Wiki Sentences

LOWNER, the Wiki sentences component of MULTICONER is obtained by parsing Wikipedia articles into sentences, and selecting suitable candidates. Figure 2 shows a diagram of the basic data processing steps, which are described below.

This process is performed for the following languages: `NL`, `EN`, `FA`, `KO`, `RU`, `ES`, `TR`. For the other languages (`BN`, `ZH`, `DE`, `HI`), we apply Machine Translation to obtain the data, as described in §3.4.

Wikipedia Parsing (A) We start by downloading the complete Wikipedia dumps for our target languages.[6] The files are parsed to first extract individual articles, which are then each parsed to remove markup and extract individual sentences. This process yields a set of sentences[7] with the interlinks intact, along with the IDs of the original article they were extracted from.

Interlink Parsing (B) In the next step, we parse the sentences to identify interlinks (links to other Wikipedia articles). We then map the interlinks in each sentence to an entity in the Wikidata KB. This mapping is provided in the KB, which links entities to their Wikipedia page names, which can be used to map linked pages to an entity ID. The identified entities are finally resolved to our NER taxonomy (using the same approach that was described in §3.1). Some interlinks point to inexisting Wikipedia articles, or the linked Wikipedia article cannot be joined to a corresponding Wikidata entry. We mark such cases as unresolvable.

Sentence Filtering (C) Next, we filtered sentences using several strategies and heuristics.

[6]Dumps are available here: https://dumps.wikimedia.org/backup-index.html

[7]e.g., > 180 million sentences for English.

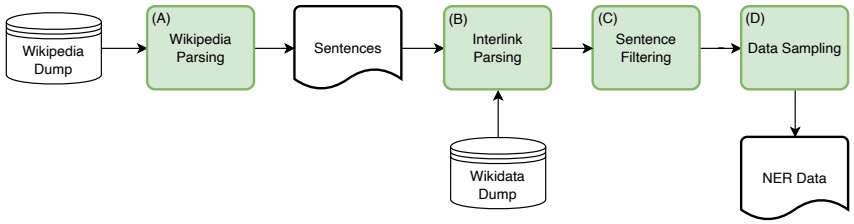

Figure 2: An overview of the different steps involved in extracting the MULTICONER data from Wikipedia dumps.

- Length filtering: short sentences (< 28 characters) and long ones (> 180 characters) are removed.
- Interlink filtering: sentences without interlinks, or with unresolvable links, are dropped. Sentences with interlinks that did not map to our taxonomy are dropped.
- Capitalization heuristic filtering: for languages that capitalize proper nouns or entities, a heuristic is used to filter out sentences that contain capitalized tokens that are not part of an interlink. This removes sentences containing potential nouns that cannot be tagged as entities by our method since they are not linked to a known entity whose type can be determined.

This filtering process removes long and high-context sentences that contain references to many entities. This step discards over 90% of the sentences retrieved in the prior steps, e.g., resulting in approx. 14 million candidate sentences for EN. Finally, to assess the NER label quality, for a small random sample of 400 sentences, we assessed the accuracy of NER gold labels, which was was measured at 94% accuracy for EN-LOWNER.

This process is very effective at yielding short, low-context sentences. Example English sentences are shown in Table 5. The sentences contain some context, but they are much shorter than the average Wikipedia sentence, and usually only contain a single entity, making them more aligned with the challenges listed in Section 1.1.

The design is considered a forerunner to the modern [food processor].
The regional capital is [Oranjestad, Sint Eustatius].
The most frequently claimed loss was an [iPad].
An [HP TouchPad] was prominently displayed in an episode of the sixth season.
The incumbent island governor is [Jonathan G. A. Johnson].
A revised edition of the book was released in 2017 as an [Amazon Kindle] book.

Table 5: Sample sentences extracted from Wikipedia. Resolved entities are in brackets.

MSQ-NER	ORCAS-NER
average retail price of <PROD>	<CW> imdb
where was <CW> filmed	best hotels <LOC>
how many miles from <LOC> to <LOC>	<PER> parents
how many kids does <PER> have	<PROD> price
when did <GRP> start	<GRP> website
when will <CORP> report earnings	<CORP> customer service

Table 6: Sample templates used to generate data for the MSQ-NER and ORCAS-NER domain. Slots are in angle brackets.

Data Sampling (D) We downsample the collected data to construct a smaller subset. Given that some of the NE classes are more prevalent (e.g. PER and LOC, account for more than 60% of named entities) , similar to stratified sampling, we sample at NE class level, with the only difference, that the number of instances per class is fixed. In this way, we create a dataset that has more uniform representation of the different NE classes. Furthermore, retaining all sentences is impractical, given the total amount of data.

Finally, we lowercase all the selected sentences to increase the difficulty of the NER task. The final stats per subset and split are shown in Table 3.

3.3 Questions and Search Queries

We apply a template-based process to generate data in the Questions and Search Query domains. This involves two broad steps: template extraction, and template slotting.

The same steps are applied to two data sources to generate the NER datasets. This process is visualized in Figure 3, and the individual steps are detailed below.

Running Named Entity Recognition (A) Similar to the work of Wu et al. (2020), we aim to templatize the input questions and search queries by first applying NER to extract entities, which are then mapped to our taxonomy.

Specifically, we apply the spaCy NER pipeline[8]

[8]We use the en_core_we_lg pre-trained pipeline in

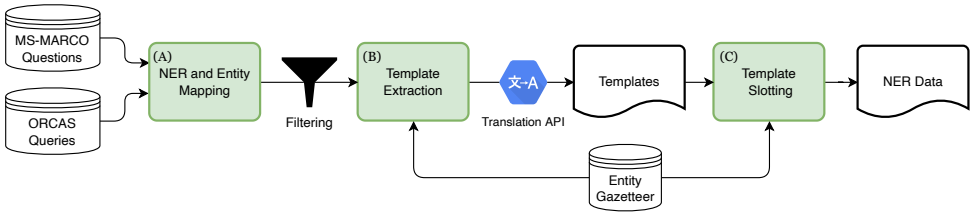

Figure 3: An overview of the data processing steps in our template-based approach to generating the NER data for the MSQ-NER and ORCAS-NER domains.

to identify entities. While this pre-trained NER system cannot correctly identify all the entities in the data, it does identify many correct ones. This process yields a sufficient amount of data for us to extract common patterns from the original input. Recognized entities are then mapped to entries in our gazetteer via string matching. Input texts that have entities that could not be mapped, or have no recognized entities are then filtered out. This process yields a set of sentences, with recognized entities that exist in our gazetteer.

Template Extraction (B) Next, we replace identified entities with their types to create templates, e.g., "`when did [iphone] come out`" is transformed to "`when did <PROD> come out`". The templates are then grouped together in order to merge all inputs having the same template, and sorted by frequency.

To minimize noise in the data, we apply frequency-based filtering, and only templates appearing $>= 5$ times are included. This process results in $3,445$ unique question templates, and $97,324$ unique search query templates. There are a wide range of question shapes and entity types. Examples are listed in Table 6.

Since these templates are all in English, we apply Machine Translation to translate them into the other 10 languages of our dataset.

Template Slotting (C) In the last step we generate the NER data by slotting the templates with random entities from the Wikipedia KB with the same class. For example, "`when did <PROD> come out`" can be slotted as "`when did [xbox 360] come out`" or "`when did [Sony Alpha DSLR-A77 II] come out`".

To maintain the same relative distribution as the original data, each template is slotted the same number of times it appeared (i.e., the template frequency) using different entities. Templates are slotted with entities from the same language, i.e., a DE template is slotted with DE entities. The slotted texts are lowercased to simulate the low-context challenges outlined in §1.1, which increases the difficulty of the task. This yields two domains of MULTICONER: MSQ-NER and ORCAS-NER.

```
EN: average cost of living in <p translate=no> <LOC> </p>
ZH: <p translate=no> <LOC> </p>的平均生活成本
DE: durchschnittliche Lebenshaltungskosten in <p
translate=no> <LOC> </p>
HI: रहनए कइ अउसत लअअ <p translate=no> <LOC> </p>
```

Table 7: Examples of template translations.

3.4 Dataset Translation

We apply automatic translation to generate two portions of our data. LOWNER sentences for four languages (Bangla, Chinese, German, Hindi) are translated from English Wiki sentences. The NER templates used for MSQ-NER and ORCAS-NER are also translated from the English templates. We do not translate any of our gazetteer entities.

We use the Google Translation API[9] to perform our translations. The input texts may contain known entity spans or slots. To prevent these spans from being translated, we leveraged the `notranslate` attribute to mark these spans and prevent them from being translated. Table 7 shows examples of template translations.

The translation quality of LOWNER, ORCAS-NER and MSQ-NER in the different languages such as German, Chinese, Bangla, and Hindi is very high, with over 90% translation accuracy (i.e., accuracy as measured by human annotators in terms of the translated sentence retaining the semantic meaning and as well have a correct syntactic structure in the target language).

3.5 Code-mixed Data Generation

We generate code-mixed data by sampling instances from the respective languages, and replacing the NE surface forms from the *source* language

spaCy where the NER model is trained using OntoNotes 5.

[9] https://cloud.google.com/translate

to a *target* language, chosen at random from any of the languages in Table 2. This results in a dataset, where the instances contain up to two languages, where the non-NE tokens are in a language that is different from the NE tokens. Note that, in some cases, some of the NE surface forms may remain in the source language if we do not possess the NE's surface form in one of the other languages from Table 2.[10]

4 NER Model Performance

To evaluate whether our new dataset poses real-world challenges (cf. §1.1), we train and test two existing NER systems: (1) XLM-RoBERTa (Conneau et al., 2020), a large multilingual Transformer model; and (2) GEMNET (Meng et al., 2021; Fetahu et al., 2021, 2022), a state of the art model that integrates gazetteers into XLM-RoBERTa.

4.1 Evaluation Metrics

We evaluate the different NER models using standard performance metrics, such as P/R/F1. We measure the performance at the class level, where we distinguish between *micro/macro* averages. The difference between *micro* and *macro* average P/R/F1, is that for unbalanced distribution *micro* performance is skewed towards the more prominent NE classes. Additionally, we consider *Mention Detection* (MD), which corresponds to the ability of models to detect NE boundaries, without taking into consideration their actual NER class.

4.2 Results

Table 8 shows the results obtained for both XLM-RoBERTa (baseline, denoted as B), and a state of the art model, GEMNET (denoted as GM). The results are shown only for the F1 score achieved on the individual NER classes, and finally the micro, macro F1 and MD scores are shown. Table 10 in §A.2 shows a detailed performance for each subtask and domain.

Baseline. For the baseline approach, we simply fine-tune XLM-RoBERTa on the language specific training data, and test on the corresponding test splits. We note that overall, across all subsets, the baseline achieves the highest performance of micro-F1=0.646 for DE, and lowest of micro-F1=0.397 for BN. This result is expected, given that the test data contains highly complex entities that are out-of-domain, and additionally are not seen in the training data.

GEMNET. The state of the art approach, GEMNET, makes use of external gazetteers, constructed from Wikidata for the task of NER. For each token GEMNET computes two representations: (i) textual representation based on XLM-RoBERTa, and (ii) gazetteer encoding, which uses a gazetteer to map to tokens to gazetteer entries, which correspondingly maps them to their NER tags. The two representations are combined using a Mixture-of-Experts (MoE) gating mechanism (Shazeer et al., 2017), which allows the model depending on the context to either assign higher weight to its textual representation or the gazetteer computed representation.

GEMNET provides a highly significant improvement over the baseline, with an average improvement of micro-F1=30%. The highest improvement is shown for languages that are considered to be low-resource, such as `TR` with micro-F1=+41.5%, and `KO` with micro-F1=+33%.

The obtained results in Table 8 show that GEMNET is highly flexible in detecting unseen entities during the training phase. Depending on its gazetteer coverage, if a named entity is matched by its gazetteers, this will allow GEMNET to correctly identify the named entity. In more detail, internally, GEMNET dynamically weighs both representation of a token (i.e., textual and gazetteer representations), to correctly determine the correct tag for a token. Note that, the gazetteers may contain noisy labels for a named entity (e.g. "`Bank of America`" can match to `CORP` and `LOC`), hence, GEMNET needs to additionally leverage the token context to determine the correct tag.

5 Conclusions and Future Work

We presented MULTICONER, a new large-scale dataset that represents a number of current challenges in NER. Results obtained on our data showed that our dataset is challenging. A XLMR based model achieves only approx. 50% F1 in average while GEMNET improves F1 performance by more than 30% using gazetteers.

These results demonstrate that MULTICONER represents challenging scenarios where even large pre-trained language models fail to achieve good performance without external resources. It is our hope that this resource will help further research for building better NER systems. This dataset can

[10] For `ZH`, the tokenization is done at the character level.

	PER		LOC		GRP		CORP		CW		PROD		Micro F1		Macro F1		MD	
	B	GM	B	GM	B	GM	B	GM	B	GM	B	GM	B	GM	B	GM	B	GM
EN	0.807	0.939	0.664	0.848	0.599	0.876	0.567	0.889	0.474	0.806	0.563	0.873	0.627	0.873	0.612	0.872	0.731	0.892
DE	0.797	0.968	0.679	0.921	0.591	0.940	0.588	0.951	0.516	0.897	0.633	0.940	0.646	0.936	0.634	0.936	0.767	0.951
ES	0.750	0.941	0.589	0.854	0.531	0.884	0.564	0.893	0.496	0.811	0.515	0.840	0.587	0.872	0.574	0.870	0.689	0.888
RU	0.666	0.839	0.632	0.780	0.539	0.818	0.600	0.870	0.534	0.803	0.576	0.805	0.597	0.817	0.591	0.819	0.699	0.834
NL	0.766	0.949	0.658	0.889	0.586	0.893	0.599	0.905	0.514	0.854	0.574	0.871	0.626	0.895	0.616	0.894	0.731	0.911
KO	0.595	0.900	0.650	0.865	0.513	0.910	0.560	0.923	0.439	0.846	0.517	0.905	0.558	0.888	0.546	0.891	0.666	0.896
FA	0.634	0.870	0.588	0.792	0.573	0.867	0.473	0.805	0.362	0.688	0.480	0.797	0.523	0.801	0.518	0.803	0.638	0.823
TR	0.549	0.894	0.497	0.860	0.404	0.896	0.480	0.897	0.374	0.849	0.441	0.914	0.468	0.883	0.457	0.885	0.614	0.893
ZH	0.532	0.884	0.627	0.889	0.371	0.866	0.552	0.902	0.434	0.818	0.552	0.861	0.531	0.870	0.511	0.870	0.664	0.895
HI	0.578	0.883	0.536	0.846	0.485	0.869	0.502	0.851	0.298	0.760	0.418	0.839	0.478	0.843	0.469	0.841	0.640	0.877
BN	0.529	0.895	0.420	0.850	0.322	0.883	0.428	0.889	0.243	0.747	0.406	0.865	0.397	0.856	0.391	0.855	0.570	0.888
MULTI	0.679	0.810	0.556	0.743	0.496	0.721	0.563	0.746	0.428	0.644	0.523	0.697	0.550	0.732	0.541	0.727	0.674	0.810
MIX	0.709	0.835	0.621	0.765	0.532	0.714	0.581	0.748	0.481	0.604	0.560	0.735	0.585	0.731	0.581	0.733	0.752	0.847
Avg.	0.661	0.893	0.594	0.839	0.503	0.857	0.543	0.867	0.430	0.779	0.520	0.842	0.552	0.846	0.542	0.846	0.680	0.877

Table 8: XLM-RoBERTa (B) baseline and GEMNET (GM) results as measured by the F1 score for the different NER tags. In the last three columns are shown the *micro*, *macro*, and *mention detection* – MD F1 performance.

serve as a good benchmark for evaluating different methods of infusing external entity knowledge into language models.

The extension of MULTICONER to additional languages is the most straightforward direction for future work. The addition of completely new domains is something we will also consider, along with the the expansion of the existing domains to include additional topics and templates.

References

Gustavo Aguilar, Suraj Maharjan, Adrian Pastor López-Monroy, and Thamar Solorio. 2017. A multi-task approach for named entity recognition in social media data. In *Proceedings of the 3rd Workshop on Noisy User-generated Text*, pages 148–153.

Isabelle Augenstein, Leon Derczynski, and Kalina Bontcheva. 2017. Generalisation in named entity recognition: A quantitative analysis. *Computer Speech & Language*, 44:61–83.

Payal Bajaj, Daniel Campos, Nick Craswell, Li Deng, Jianfeng Gao, Xiaodong Liu, Rangan Majumder, Andrew McNamara, Bhaskar Mitra, Tri Nguyen, et al. 2016. Ms marco: A human generated machine reading comprehension dataset. *arXiv preprint arXiv:1611.09268*.

Alexis Conneau, Kartikay Khandelwal, Naman Goyal, Vishrav Chaudhary, Guillaume Wenzek, Francisco Guzmán, Edouard Grave, Myle Ott, Luke Zettlemoyer, and Veselin Stoyanov. 2020. Unsupervised cross-lingual representation learning at scale. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, ACL 2020, Online, July 5-10, 2020*, pages 8440–8451. Association for Computational Linguistics.

Nick Craswell, Daniel Campos, Bhaskar Mitra, Emine Yilmaz, and Bodo Billerbeck. 2020. Orcas: 18 million clicked query-document pairs for analyzing search. *arXiv preprint arXiv:2006.05324*.

Leon Derczynski, Eric Nichols, Marieke van Erp, and Nut Limsopatham. 2017. Results of the wnut2017 shared task on novel and emerging entity recognition. In *Proceedings of the 3rd Workshop on Noisy User-generated Text*, pages 140–147.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: pre-training of deep bidirectional transformers for language understanding. In *NAACL-HLT (1)*. Association for Computational Linguistics.

Besnik Fetahu, Anjie Fang, Oleg Rokhlenko, and Shervin Malmasi. 2021. Gazetteer enhanced named entity recognition for code-mixed web queries. In *Proceedings of the 44th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 1677–1681.

Besnik Fetahu, Anjie Fang, Oleg Rokhlenko, and Shervin Malmasi. 2022. Dynamic gazetteer integration in multilingual models for cross-lingual and cross-domain named entity recognition. In *Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL 2022, Seattle, WA, United States, July 10-15, 2022*, pages 2777–2790. Association for Computational Linguistics.

Daniel Khashabi, Mark Sammons, Ben Zhou, Tom Redman, Christos Christodoulopoulos, Vivek Srikumar, Nicholas Rizzolo, Lev-Arie Ratinov, Guanheng Luo, Quang Do, Chen-Tse Tsai, Subhro Roy, Stephen Mayhew, Zhili Feng, John Wieting, Xiaodong Yu, Yangqiu Song, Shashank Gupta, Shyam Upadhyay, Naveen Arivazhagan, Qiang Ning, Shaoshi Ling, and Dan Roth. 2018. Cogcompnlp: Your swiss army knife for NLP. In *LREC*. European Language Resources Association (ELRA).

Shervin Malmasi, Anjie Fang, Besnik Fetahu, Sudipta Kar, and Oleg Rokhlenko. 2022. Semeval-2022 task 11: Multilingual complex named entity recognition (multiconer). In *Proceedings of the 16th International Workshop on Semantic Evaluation, SemEval@NAACL 2022, Seattle, Washington, United States, July 14-15, 2022*, pages 1412–1437. Association for Computational Linguistics.

Stephen Mayhew, Tatiana Tsygankova, and Dan Roth. 2019. ner and pos when nothing is capitalized. In *EMNLP/IJCNLP (1)*, pages 6255–6260. Association for Computational Linguistics.

Tao Meng, Anjie Fang, Oleg Rokhlenko, and Shervin Malmasi. 2021. GEMNET: effective gated gazetteer representations for recognizing complex entities in low-context input. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT 2021, Online, June 6-11, 2021*, pages 1499–1512. Association for Computational Linguistics.

Erik Tjong Kim Sang and Fien De Meulder. 2003. Introduction to the conll-2003 shared task: Language-independent named entity recognition. In *Proceedings of the Seventh Conference on Natural Language Learning at HLT-NAACL 2003*, pages 142–147.

Noam Shazeer, Azalia Mirhoseini, Krzysztof Maziarz, Andy Davis, Quoc V. Le, Geoffrey E. Hinton, and Jeff Dean. 2017. Outrageously large neural networks: The sparsely-gated mixture-of-experts layer. In *5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Conference Track Proceedings*. OpenReview.net.

Marlies van der Wees, Arianna Bisazza, Wouter Weerkamp, and Christof Monz. 2015. What's in a domain? analyzing genre and topic differences in statistical machine translation. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing of the Asian Federation of Natural Language Processing, ACL 2015, July 26-31, 2015, Beijing, China, Volume 2: Short Papers*, pages 560–566. The Association for Computer Linguistics.

Tongshuang Wu, Kanit Wongsuphasawat, Donghao Ren, Kayur Patel, and Chris DuBois. 2020. Tempura: Query analysis with structural templates. In *Proceedings of the 2020 CHI Conference on Human Factors in Computing Systems*, pages 1–12.

A Appendix

A.1 Gazetteer Statistics

Table 9 shows the number of entries for NE class and language. The entries are extracted from Wikidata.

lang	PER	LOC	CORP	GRP	PROD	CW
BN	42,970	31,336	1,072	8,691	990	12,152
ZH	388,910	346,879	30,323	64,031	15,919	120,831
NL	1,321,741	738,609	27,589	79,566	21,105	204,130
EN	1,797,558	1,117,951	72,105	227,822	67,113	490,523
FA	224,265	233,962	8,641	14,346	11,802	60,857
DE	1,308,532	533,551	42,321	99,468	38,735	219,801
HI	22,279	18,480	1,160	2,382	1,044	7,826
KO	148,367	72,153	9,625	23,209	8,385	55,624
RU	984,093	495,059	21,609	68,834	21,571	148,003
ES	1,389,698	480,310	29,465	113,197	25,658	228,369
TR	171,133	141,225	6,099	19,388	6,718	43,029

Table 9: Gazetteer entity statistics per class for our target languages.

A.2 Cross-Domain Evaluation Results

Table 10 shows for the different subtasks, the cross-domain evaluation results for the baseline and GEMNET approaches. We note that in all cases the GEMNET approach shows strong gains in terms of macro-F1 score across all subtasks. This is especially the case for the domains MSQ-NER and ORCAS-NER, where the models contain very little knowledge about these domains[11], hence, showing the generalizability of models in out-of-domain scenarios.

Finally, we note that in the case of the LOWNER domain, which is an in-domain evaluation scenario[12], in terms of MD, the gap between the baseline and GEMNET approach shrinks. For the Multi, the gap is only 4.1%. This shows that the baseline models for in-domain scenarios is able to correctly identify entity boundaries, even though its NER accuracy may not be optimal. For instance, for Multi the gap in terms of macro-F1 is 12.7%. showing, that models that leverage external knowledge like GEMNET, are more accurate in terms of NER accuracy and as well have higher coverage in spotting entity boundaries.

[11]The MultiCoNER training set for each of the subtasks, contains 50 instances from the MSQ-NER and ORCAS-NER domains

[12]The MultiCoNER training set consists nearly of only LOWNER instances.

	PER		LOC		GRP		CORP		CW		PROD		Micro F1		Macro F1		MD	
	B	GM	B	GM	B	GM	B	GM	B	GM	B	GM	B	GM	B	GM	B	GM
Domain – LOWNER																		
EN	0.921	0.971	0.855	0.938	0.766	0.925	0.756	0.939	0.681	0.862	0.656	0.849	0.789	0.920	0.773	0.914	0.851	0.932
DE	0.913	0.978	0.871	0.957	0.781	0.948	0.776	0.952	0.706	0.913	0.772	0.921	0.816	0.949	0.803	0.945	0.903	0.965
ES	0.897	0.944	0.797	0.866	0.725	0.871	0.792	0.910	0.667	0.802	0.627	0.761	0.759	0.864	0.751	0.859	0.821	0.883
RU	0.734	0.794	0.702	0.757	0.695	0.794	0.745	0.855	0.687	0.793	0.647	0.753	0.702	0.788	0.702	0.791	0.752	0.809
NL	0.904	0.949	0.878	0.926	0.797	0.900	0.801	0.898	0.732	0.840	0.715	0.810	0.816	0.894	0.805	0.887	0.871	0.913
KO	0.774	0.896	0.817	0.885	0.734	0.882	0.760	0.910	0.710	0.850	0.714	0.852	0.761	0.880	0.751	0.879	0.810	0.890
TR	0.813	0.897	0.825	0.875	0.807	0.906	0.798	0.906	0.684	0.831	0.640	0.805	0.768	0.871	0.761	0.870	0.818	0.884
ZH	0.869	0.917	0.855	0.924	0.534	0.795	0.740	0.859	0.659	0.816	0.655	0.834	0.743	0.868	0.719	0.858	0.811	0.901
HI	0.792	0.837	0.732	0.813	0.710	0.757	0.651	0.713	0.487	0.578	0.524	0.634	0.649	0.722	0.649	0.722	0.765	0.813
BN	0.822	0.853	0.769	0.823	0.701	0.780	0.666	0.725	0.569	0.663	0.576	0.679	0.680	0.752	0.684	0.754	0.814	0.859
MULTI	0.855	0.916	0.808	0.882	0.717	0.868	0.733	0.884	0.664	0.825	0.648	0.808	0.741	0.868	0.737	0.864	0.852	0.893
MIX	0.855	0.862	0.808	0.809	0.717	0.737	0.733	0.757	0.664	0.616	0.648	0.719	0.741	0.749	0.737	0.750	0.852	0.850
Domain – MSQ-NER																		
EN	0.781	0.921	0.613	0.823	0.366	0.788	0.408	0.801	0.348	0.795	0.355	0.852	0.598	0.842	0.479	0.830	0.733	0.860
DE	0.758	0.984	0.708	0.958	0.317	0.939	0.397	0.964	0.415	0.909	0.346	0.948	0.643	0.959	0.490	0.950	0.783	0.970
ES	0.700	0.979	0.526	0.879	0.216	0.857	0.403	0.924	0.388	0.856	0.335	0.885	0.529	0.901	0.428	0.897	0.643	0.912
RU	0.692	0.961	0.652	0.864	0.317	0.904	0.436	0.842	0.435	0.915	0.280	0.806	0.601	0.891	0.469	0.882	0.726	0.904
NL	0.745	0.980	0.511	0.895	0.273	0.831	0.450	0.947	0.436	0.922	0.342	0.937	0.546	0.919	0.460	0.919	0.680	0.932
KO	0.547	0.864	0.644	0.917	0.255	0.903	0.370	0.947	0.288	0.907	0.235	0.924	0.531	0.904	0.390	0.910	0.669	0.908
FA	0.674	0.914	0.512	0.789	0.533	0.829	0.413	0.805	0.236	0.740	0.331	0.762	0.499	0.814	0.450	0.807	0.615	0.840
TR	0.597	0.881	0.568	0.905	0.246	0.875	0.389	0.956	0.357	0.890	0.211	0.873	0.517	0.897	0.395	0.897	0.647	0.908
ZH	0.534	0.947	0.709	0.957	0.401	0.907	0.432	0.941	0.390	0.920	0.283	0.843	0.588	0.945	0.458	0.919	0.743	0.961
HI	0.725	0.955	0.715	0.925	0.464	0.925	0.572	0.929	0.360	0.899	0.280	0.827	0.656	0.928	0.519	0.910	0.802	0.952
BN	0.589	0.950	0.468	0.879	0.000	0.000	0.433	0.942	0.298	0.821	0.239	0.793	0.465	0.891	0.338	0.731	0.643	0.915
MULTI	0.628	0.775	0.571	0.751	0.271	0.503	0.401	0.602	0.323	0.539	0.306	0.463	0.531	0.712	0.417	0.605	0.688	0.817
MIX	0.629	0.857	0.477	0.764	0.445	0.733	0.521	0.786	0.349	0.666	0.532	0.777	0.496	0.763	0.492	0.764	0.738	0.891
Domain – ORCAS-NER																		
EN	0.588	0.886	0.340	0.719	0.313	0.811	0.342	0.834	0.191	0.736	0.430	0.902	0.365	0.813	0.367	0.815	0.530	0.841
DE	0.581	0.943	0.355	0.839	0.313	0.929	0.388	0.949	0.266	0.868	0.454	0.959	0.392	0.913	0.393	0.914	0.564	0.926
ES	0.524	0.927	0.296	0.815	0.260	0.902	0.323	0.875	0.229	0.811	0.333	0.929	0.331	0.876	0.327	0.876	0.490	0.888
RU	0.535	0.871	0.470	0.770	0.327	0.845	0.417	0.887	0.313	0.791	0.472	0.864	0.428	0.838	0.422	0.838	0.597	0.850
NL	0.543	0.939	0.292	0.815	0.274	0.887	0.366	0.912	0.265	0.865	0.409	0.943	0.360	0.892	0.358	0.893	0.536	0.905
KO	0.445	0.916	0.401	0.812	0.321	0.938	0.403	0.935	0.220	0.835	0.362	0.945	0.359	0.896	0.359	0.897	0.529	0.900
FA	0.498	0.870	0.386	0.759	0.450	0.884	0.327	0.788	0.202	0.641	0.399	0.822	0.361	0.790	0.377	0.794	0.535	0.816
TR	0.459	0.900	0.338	0.823	0.295	0.898	0.403	0.892	0.274	0.849	0.376	0.944	0.361	0.884	0.358	0.884	0.538	0.893
ZH	0.396	0.854	0.398	0.821	0.368	0.872	0.468	0.920	0.291	0.816	0.473	0.878	0.397	0.860	0.399	0.860	0.555	0.880
HI	0.492	0.875	0.390	0.810	0.410	0.905	0.460	0.886	0.266	0.791	0.387	0.902	0.401	0.861	0.401	0.861	0.578	0.882
BN	0.459	0.886	0.334	0.838	0.265	0.898	0.372	0.913	0.192	0.752	0.365	0.906	0.331	0.867	0.331	0.866	0.506	0.888
MULTI	0.479	0.645	0.322	0.516	0.305	0.533	0.401	0.589	0.240	0.443	0.411	0.567	0.356	0.545	0.360	0.549	0.543	0.689
MIX	0.517	0.792	0.308	0.685	0.324	0.687	0.387	0.722	0.235	0.563	0.421	0.739	0.364	0.695	0.365	0.698	0.577	0.828

Table 10: XLM-RoBERTa (B) baseline and GEMNET (G) domain results as measured by the F1 score for the different NER tags. The last three columns show the *micro*, *macro*, and *mention detection* – MD F1 performance.

Extracting a Knowledge Base of COVID-19 Events from Social Media

Shi Zong[1] **Ashutosh Baheti**[2] **Wei Xu**[2] **Alan Ritter**[2]
[1]University of Waterloo [2]Georgia Institute of Technology
s4zong@uwaterloo.ca, abaheti95@gatech.edu
{wei.xu, alan.ritter}@cc.gatech.edu

Abstract

We present a manually annotated corpus of 10,000 tweets containing public reports of five COVID-19 events, including positive and negative tests, deaths, denied access to testing, claimed cures and preventions. We designed slot-filling questions for each event type and annotated a total of 28 fine-grained slots, such as the location of events, recent travel, and close contacts. We show that our corpus can support fine-tuning BERT-based classifiers to automatically extract publicly reported events, which can be further collected for building a knowledge base. Our knowledge base is constructed over Twitter data covering two years and currently covers over 4.2M events. It can answer complex queries with high precision, such as *"Which organizations have employees that tested positive in Philadelphia?"* We believe our proposed methodology could be quickly applied to develop knowledge bases for new domains in response to an emerging crisis, including natural disasters or future disease outbreaks.[1]

1 Introduction

Since December 2019, the novel coronavirus rapidly spread across the world, and consequently, a flood of COVID-19 related information has appeared on social media. This includes reports on public figures who have tested positive/negative for the virus, which often break first on Twitter, such as Bill Gates's announcement as shown in Figure 1. Besides public figures, individual users and organizations on Twitter also report COVID-19 events around the world. For example in January 2021, many sources in different countries reported an increasing number of new cases exported from the UK (Figure 2). Being able to gather this information can potentially help experts and the general public to quickly identify issues and assess the situation near real-time, complementing officially reported data which may take longer to obtain, and does not include information at the same level of granularity as that reported in natural language on news and social media.

Figure 1: Example tweet that contains a self-reported TESTED POSITIVE event.

In this paper, we present an empirical study on the extraction of large quantities of structured knowledge related to an ongoing pandemic from Twitter. To achieve this, we construct a corpus of 10,000 tweets with rich linguistic annotations, covering five event types: positive tests, negative tests, denied access to testing, deaths, claimed methods of cure and prevention. More specifically, we annotate fine-grained semantic information for each event type by designing slot-filling questions and asking annotators to highlight text spans as answers. We show that our corpus can support training BERT-based classifiers to extract structured information automatically from Twitter. While slot F1 scores vary from 0.3 to 0.9 in individual tweets (most F1 scores are greater than 0.5), we show it is possible to achieve very high accuracy by aggregating extractions over a large corpus, exploiting redundancy of information that arises when events are widely discussed on Twitter. Although many Twitter datasets have emerged after the COVID-19 outbreak, to the best of our knowledge, our work is the first to provide complex linguistic annotations to support structured information extraction.

To demonstrate the utility of our dataset, we built COVIDKB, a knowledge base that supports

[1]Our corpus (with user-information removed), automatic extraction models, and the corresponding knowledge base are publicly available at https://github.com/viczong/extract_COVID19_events_from_Twitter.

Proceedings of the 29th International Conference on Computational Linguistics, pages 3810–3823
October 12–17, 2022.

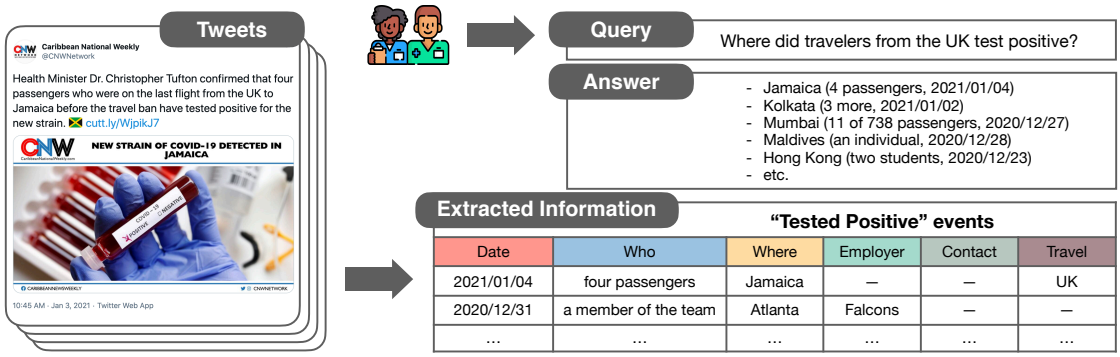

Figure 2: Overview of our COVID-19 event extraction system, which continuously extracts and indexes structured information about publicly reported events from Twitter. Users can enter structured queries to retrieve relevant tweets, such as `{location:?, travel:UK}` to find test positive cases that are exported from the UK.

structured queries over COVID-19 events, by indexing events extracted by our model over millions of tweets. Our system allows users to execute structured search queries over the extracted events, answering questions such as *"Which organizations in Houston have reports of employees who tested positive?"* or *"Who tested positive that had close contact with Boris Johnson?"* (see Figure 2). We envision CovidKB could help address the issue of information overload for professionals (Zhang et al., 2020) who need to stay on top of recent developments related to COVID-19, including journalists (Karmakharm et al., 2019), epidemiologists and public policymakers. Our extractor can also detect claims about methods of cures and prevention of the disease, which could be useful in helping to track online misinformation (Thorne et al., 2018; Stefanov et al., 2020; Hossain et al., 2020).

2 Related Work

Event Extraction from Twitter. There has been much interest in extracting events from Twitter. For example, Ritter et al. (2012) built a system for open domain event extraction. Recent work also explored extraction of cybersecurity events (Ritter et al., 2015; Chang et al., 2016), including denial of service attacks (Chambers et al., 2018) and software vulnerabilities (Zong et al., 2019). Zhou et al. (2017) use a nonparametric Bayesian mixture model for event extraction. In this work, we design event types and attributes that are specific for COVID-19 and develop automatic NLP tools for extracting structured information from tweets.

Existing COVID-19 Datasets. There have been many datasets that collect tweets related to COVID-19 (Chen et al., 2020; Banda et al., 2020). However, most are either unlabeled or provided with general-purpose NLP model predictions, rather than structured linguistic annotations of COVID-specific information, as in this work. For example, Twitter officially releases a stream with predicted entities (such as *person* and *place*) and topic labels (such as *sports* and *movies*). Qazi et al. (2020) released a COVID-19 collection of geo-located tweets that contain COVID relevant keywords and hashtags. Dimitrov et al. (2020) put together 8 million tweets with automatically generated entity linking and sentiment scores. Hu et al. (2020) presented a large-scale dataset of 40 million raw posts from Weibo with no annotations. There also exist a few datasets that contain human annotations at the time of writing. For example, Hossain et al. (2020) annotated 5,000 tweets for studying COVID-19 misconceptions. Nguyen et al. (2020) classified 10,000 tweets as informative and uninformative. Amini et al. (2021) annotated a dataset of mechanism relations from COVID-19 related scientific papers. Compared to prior work, we provide more fine-grained human annotations on text spans with predefined slots for COVID-19 events. Our annotations can support training supervised learning models that are capable of extracting structured information (Adrian Bejan and Harabagiu, 2014; Venugopal et al., 2014), similar to other influential datasets in information extraction and question answering, such as KBP (Ji et al., 2011) and SQuAD (Rajpurkar et al., 2016).

Social Media Monitoring for Public Health. Analyzing social media and other user-generated web data for monitoring public health has been an active research area. For example, Google Flu Trends (GFT) uses search engine query data to detect influenza epidemics (Ginsberg et al., 2009). Paul

et al. (2014) use the Twitter message content to forecast influenza rates. GFT has been found to over-estimate influenza-like illness (Lazer et al., 2014). In contrast to GFT, our main focus is to develop methods that process large quantities of raw tweets into a *structured* format to help people find specific information, rather than forecasting or nowcasting official statistics.

3 An Annotated Corpus for COVID-19 Event Extraction

To extract structured knowledge from tweets, we formulate the problem as a supervised slot filling task (Jurafsky and Martin, 2000; Benson et al., 2011; Ji et al., 2011). Specifically, given a tweet, annotators are asked to first identify whether it contains a relevant event, then highlight the text spans of answers that correspond to a list of pre-defined questions for each event type (detailed questions are in Table A2).

3.1 Data Collection

We consider five event types related to COVID: TESTED POSITIVE, TESTED NEGATIVE, CAN NOT TEST, DEATH, and CURE & PREVENTION. The design of these event types is inspired by the statistics reported in Johns Hopkins COVID-19 dashboard, which are of interest to the public and epidemiologists.[2] The first four types aim to extract structured information about events related to COVID-19, many of which are news stories about public figures. We have been continuously collecting Twitter data related to COVID-19 since 2020/01/15 by tracking relevant keywords using the Twitter API, such as *tested positive* for TESTED POSITIVE events (see Table A1 for a full list of our carefully selected keywords). As we will shown in Section 6.1, our fixed set of keywords are able to track the evolution of pandemic even over a period of two years, although a dynamic selection of keywords is promising to explore in future work.

Preprocessing. In this work, we mainly focus on English tweets, identified by using `langid.py` (Lui and Baldwin, 2012). We remove retweets and other duplicates, keeping the tweet that was posted earliest. Before de-duplication process, all URLs and user mentions are removed. We also use Jaccard similarity with a threshold of 0.7 to remove near-identical tweets that are posted same-day.

[2] https://coronavirus.jhu.edu/map.html

Event Type	# Anno. Total	# Event Specific	# Slots
TESTED POSITIVE	3,000	2,146	9
TESTED NEGATIVE	1,700	893	8
CAN NOT TEST	1,700	680	5
DEATH	1,800	626	6
CURE & PREV.	1,800	832	3
Total	10,000	5,177	31

Table 1: Statistics of COVID-19 Twitter Event Corpus.

3.2 Annotation Process

We randomly sample 10,000 tweets from five event types to annotate. The train and dev sets consist of 7,500 annotated tweets, that were published between 2020/01/15 and 2020/04/26. To construct the test set, we annotated 2,500 tweets, 500 for each event type, that were published from a later time period between 2020/04/27 and 2020/06/27. This simulates a real-world scenario that a model is trained on historical records and then applied to future data. Table 1 shows the overall statistics of our labeled corpus.

3.2.1 Two-phase Annotation

Given a tweet, annotators are asked to first identify whether it contains a relevant event, then highlight the text spans of answers that correspond to a list of pre-defined questions for each event type in Table A2. We hire crowd workers on Amazon's Mechanical Turk to annotate our full dataset. Each of the 10,000 tweets is annotated by 7 crowd workers in two steps. We paid crowd workers $0.4-0.5 per HIT and gave extra bonuses to annotators with high annotation quality. The hourly pay was approximately $8.55. The main portion of our annotation interface is shown in Figure A1.

Part 1: Event Specificity. Although tweets have been filtered by keywords for each event type, many of them are generic news reports, such as, *"37% of those tested under 17 for Coronavirus in California tested positive"*. Since we are interested in capturing tweets with detailed information, we first ask the annotators to judge whether a tweet refers to a specific event. For example, for tweets about positive tests, we ask the annotators whether a tweet is about an individual or a small group of people testing positive. Annotators proceed to the next step only if they answer yes to this question.

Part 2: Slot Filling. In the second step, we ask a set of pre-defined questions specifically designed for each event type, as listed in Table A2. The annotators are provided with candidate answers, which include all noun phrases and named entities

extracted by a Twitter-specific NLP tool (Ritter et al., 2011),[3] in a drop-down list. We also combine noun phrases if they are adjacent or separated by a preposition.[4] We include *author of the tweet* as an additional option for the WHO questions.[5] For each tweet, annotators have an average of 10 to 11 possible answers to choose from, and are allowed to choose more than one answer for WH-questions.

3.2.2 Inter-annotator Agreement

During annotation, we track crowd workers' performance by comparing their annotations with the majority vote of other workers and remove workers' qualifications if their F1 scores fall below 0.65.[6] For the first step of annotation on specificity, the inter-annotator agreement between crowdsourcing workers is 0.68, measured by Fleiss κ (Artstein and Poesio, 2008). We observe a 0.62 F1 score for selected text spans between annotators in our slot filling task, by using each Turker's annotation in turn as the prediction, and then compare it against answers from all other workers. Same method to calculate inter-annotator agreement for text spans has been used in Yang et al. (2018) and Lee and Sun (2019).

To further validate the quality of slot-filling annotations from the crowdsourcing workers, we hired an experienced in-house annotator to carefully re-annotate the test set (2,500 tweets total, with 500 from each event; see Section 3.1 for details). The in-house annotator is paid $15 per hour. By comparing crowdsourcing workers with our in-house annotator, we find individual annotators do miss some examples, which is similar to previous reports on linguistic annotations on relations and events, such as ACE 2005 (Min and Grishman, 2012). However, by aggregating annotations from multiple crowdsourcing workers,[7] we observe high agreement (an average of 0.72 F1 score) with our in-house annotator. We also ask the in-house annotator to examine a sample of tweets to find answer spans that are not identified as candidates by the automatic NLP tool. We find this scenario occurs in less than 2% of tweets in our dataset.

3.3 Corpus Analysis

Basic Statistics. Our annotated tweets have an average length of 34.6 tokens with a standard deviation of 15.6 tokens. We note 41.42% of the tweets have external links and 29.64% include hashtags. Examples of our annotated tweets are in Table A3.

Bots and Organizational Accounts. Among all the 9,656 unique users, 2.4% are potentially bots, as identified by the Botometer API (Varol et al., 2017). We also note that 4.1% of tweets about CURE & PREVENTION are potentially posted by bots. Estimated by the Humanizr (McCorriston et al., 2015), 18.5% of user accounts in our data belong to organizations, rather than individuals.

4 Automatic Event Extraction

We now use our annotated corpus to train and evaluate supervised learning methods for automatic COVID-19 event extraction. Each slot filling question is treated as a binary classification task: given a tweet t and the candidate span c, the classification model $f_{e,s}(t, c) \rightarrow \{0, 1\}$ predicts whether c correctly answers the question for the slot s of event type e.

4.1 Experimental Settings

Baselines. We conduct experiments with two methods for automatic COVID-19 event extraction:

(1) *Logistic Regression.* We implemented a basic logistic regression classifier using bag-of-ngram features (n = 1, 2, 3). The target chunk c is replaced with a special token before computing n-grams.

(2) *Fine-tuning BERT.* We also fine-tune a BERT based classifier (Devlin et al., 2019) that takes a tweet t as input and encloses the candidate phrase c in the tweet with a pair of special entity start `<E>` and end `</E>` markers. The BERT hidden representation of token `<E>` is then fed as input to a linear layer to produce the binary prediction. Since our dataset consists of COVID-19 related tweets, we use COVID-Twitter-BERT (CT-BERT; Müller et al., 2020), an uncased $\text{BERT}_{\text{large}}$ model pre-trained on 22.5M in-domain tweets, related to COVID-19 (0.6B tokens).

Implementation Details. By design, many slots within an event are semantically related. For example, the `age` slot is directly related to the `who` slot.

[3] github.com/aritter/twitter_nlp

[4] We notice in some cases these noun phrases are not perfect and may include extra words. Annotators are instructed that a candidate answer should only be chosen when it contains no more than three extra words.

[5] These annotations are used to develop classifiers that can detect and remove instances where users publicly report information about themselves.

[6] For more discussions on managing workers on Amazon Mechanical Turk, we recommend reading: https://homes.cs.washington.edu/~msap/notes/turking-tips.html.

[7] We consider to include a span annotation for slot-filling task if 3 out of 7 MTurk annotators agree.

During development, we found it beneficial to train the final linear layers of all slots for a given event using the shared CT-BERT parameters. All shared CT-BERT models are fine-tuned with a 2e-5 learning rate using Adam (Kingma and Ba, 2015) for 4 epochs. This model has about 345M parameters.

4.2 Results

We evaluate our model performance for event type identification and slot filling on the test data, which consists of 2,500 tweets. Event types can be directly derived from the slot-filling predictions: an event is identified if text spans are extracted for any of the pre-defined slots associated with the event types by our models. Table 2 presents F1 scores on classifying event specific tweets on the test set. Table 3 presents slot filling results of the Logistic Regression, $\text{BERT}_{\text{large}}$ and CT-BERT models, as measured by precision, recall and F1 metrics.[8]

We observe that CT-BERT gives the best overall performance, which outperforms the bag-of-ngrams baseline. CT-BERT has F1 scores ranging from 0.3 to 0.9, depending on the slot for extracting events from individual tweets. The F1 score for most slots is greater than 0.5 and the final micro average F1 achieved by CT-BERT is 0.67. While we do notice some slots have low F1 scores, these slots are normally associated with few annotations in the train set. Besides, we will show in Section 5 that the performance of our CT-BERT model is sufficient to support the development of a knowledge base, which achieves much higher accuracy for COVID-19 event extraction from Twitter by aggregating extractions over a large volume of tweets.

Event Type	BERT	CT-BERT
TESTED POSITIVE	0.90	0.89
TESTED NEGATIVE	0.72	0.77
CAN NOT TEST	0.72	0.73
DEATH	0.73	0.79
CURE & PREVENTION	0.64	0.70

Table 2: F1 scores for classifying event specific tweets.

5 COVIDKB Knowledge Base

We have built models that can extract structured information related to COVID-19 from individual tweets. To demonstrate the utility of our annotated dataset and models, we create a knowledge base (Figure 2) that enables structured search over COVID-19 events that are automatically extracted from Twitter.

[8]We omit reporting results for a few slots with less than 20 annotations in test set, such as the `duration` slot for TESTED NEGATIVE and the `when` slot for CAN NOT TEST.

TESTED POSITIVE		Logistic	BERT	CT-BERT		
Slot	**#**	**F1**	**F1**	**P**	**R**	**F1**
who	375	.48	.82	.86	.82	**.84**
close contact	61	.02	.44	.65	.61	**.63**
relation	21	0.0	.51	.83	.48	**.61**
employer	121	.15	.44	.65	.54	**.59**
recent travel	27	0.0	**.36**	.44	.26	.33
when	22	.05	.38	.47	.36	**.41**
where	176	.27	.60	.91	.49	**.64**
TESTED NEGATIVE		Logistic	BERT	CT-BERT		
Slot	**#**	**F1**	**F1**	**P**	**R**	**F1**
who	274	.23	.67	.78	.68	**.73**
close contact	27	0.0	0.0	.24	.48	**.32**
relation	56	0.0	**.55**	.77	.41	.53
where	49	0.0	**.44**	.36	.55	**.44**
when	27	0.0	0.0	.35	.41	**.38**
CAN NOT TEST		Logistic	BERT	CT-BERT		
Slot	**#**	**F1**	**F1**	**P**	**R**	**F1**
who	153	.16	.57	.77	.58	**.66**
relation	70	.08	.37	.69	.34	**.46**
symptoms	52	.06	.43	.55	.62	**.58**
where	30	.20	.44	.55	.40	**.46**
DEATH		Logistic	BERT	CT-BERT		
Slot	**#**	**F1**	**F1**	**P**	**R**	**F1**
who	139	.29	.68	.83	.76	**.79**
relation	37	0.0	.59	.96	.65	**.77**
when	33	.26	**.75**	.66	.82	.73
where	65	.22	.54	.70	.60	**.64**
age	33	.18	.78	.89	.94	**.91**
CURE & PREVENTION		Logistic	BERT	CT-BERT		
Slot	**#**	**F1**	**F1**	**P**	**R**	**F1**
opinion	152	.08	.66	.85	.59	**.69**
what	261	.22	.66	.83	.64	**.72**
who	235	.08	**.51**	.87	.37	**.51**
Micro Average F1		.25	.62		**.67**	

Table 3: Slot-filling results on the test set for logistic regression, $\text{BERT}_{\text{large}}$ and CT-BERT based classifiers. # is the count of gold annotations in the test data for each slot type. F1 in bold are highest in their row.

5.1 COVIDKB Overview

COVIDKB Statistics. Until 2022/04/01 (start dates are in Table 1), our COVIDKB knowledge base has contained around 4.2M extracted events from over 20M raw tweets and is continuously growing by processing tweets daily. Events are extracted from deduplicated tweets, which follow the same pre-processing steps in Section 3.1. Breakdowns of our extracted events are listed in Table A4.

Interacting with COVIDKB. COVIDKB supports a simple structured query interface where a user specifies one or more text-filters as a query (see Figure A2). This includes two SQL operators,

Simple Queries	P@10	P@20	P@50	P@100
(S-1) Who tested positive on 2021/06/15?	100	100	100	99
(S-2) Who is promoting cures or preventions?	90	90	96	91
(S-3) Where were people not able to access testing?	100	100	100	100
(S-4) How long did people wait for negative test results?	100	85	82	82
(S-5) Which organizations have employees who tested positive?	90	90	90	94
Advanced Queries	**P@5**	**P@10**	**P@20**	**P@50**
(A-1) Who tested positive that had close contact with Boris Johnson?	80	70	60	58
(A-2) Who tested positive that has a recent travel to Japan?	100	100	100	96
(A-3) What methods of cure and prevention do people think are effective?	80	90	85	88
(A-4) Where did people test positive who traveled from the UK?	100	100	100	100
(A-5) Which organizations have employees that tested positive in San Francisco?	100	100	90	92

Table 4: Queries used to evaluate results returned by our knowledge base, reported using Precision@K. The queries are presented here in natural language for improved readability. Simple queries can be realized as a single `GroupBy` operation; advanced queries contain both `GroupBy` and `Select`. For example, the structured query for A-1 is `{who:?, contact:'Boris Johnson'}`. All queries use the default time range (from 2020/01/15 to 2022/03/01) unless explicitly specified.

`Select` and `GroupBy`. For the event slot queried by the user, using a special token "`?`", our system returns a list of all unique answers, which were extracted from tweets that match the search criteria and sorted by mention frequency. For example, a user might enter the query `{employer:?, location:'San Francisco'}`, and the system will return a list of organizations located in San Francisco where one or more employees tested positive. This simple interface enables a rich set of informative queries over events that were automatically extracted by our classification models.

Table 4 shows a list of example queries supported by COVIDKB. Queries are randomly generated by the authors of this paper. Note that throughout this paper we present queries to our system using natural language questions for the sake of readability. In each case, translation to a structured query is straightforward. The user specifies zero or more fields to filter on (`Select`) and a single field to group the results by (`GroupBy`). As our knowledge base is continuously updating, users can further combine above structured queries with different time ranges (e.g., query S-1 in Table 4 sets the start and end dates as 2021/06/15). We do not address the problem of automatically mapping natural language questions to structured queries (Suhr et al., 2018) in this work, though there is significant prior work on this topic (Artzi and Zettlemoyer, 2011; Berant et al., 2013).

5.2 COVIDKB Evaluation

Precision of Top Extractions. We evaluate the accuracy of answers returned by our knowledge base using 10 sample queries and manually inspect the correctness of the top K extractions, sorted by frequency (tweets have been deduplicated as mentioned in Section 5.1). As reported in Table 4, our knowledge base has high precision for nearly all queries, including queries involving slots with few annotations. For example, the `duration` slot is excluded in Table 3, because there are fewer than 20 instances in the test set, whereas COVIDKB still achieves good performance on queries involving this slot, thanks to the redundancy of information in Twitter. Table 5 present outputs returned by our knowledge base.

Extracted Answer Types. In Table 6, we also show a manual analysis of the types of answers, which are correctly extracted by our system for queries that target the `who` slot. We define two answer types: (1) **Specific** entities, which are clear referents to people (mostly public figures), such as *Boris Johnson* and *Dominic Cummings*; (2) **Generic** entities, which are typically nominal references, such as *a woman*. We observe that the percentage of generic answers varies heavily depending on the query. For example, query A-1 about people who had close contact with Boris Johnson consists almost entirely of references to specific public figures, whereas A-2, about people who tested positive after traveling from Japan yields only generic references.

5.3 Error Analysis

We perform an error analysis to understand the types of errors our knowledge base contains. Two authors of this paper carefully conducted manual in-

(S-1) Who tested positive on 2021/06/15?
Teofimo Lopez tests positive for COVID-19, entire Triller PPV card pushed back to August (by @mookiealexander) https://t.co/DoaHNb9Z4T
Vaccinated Hawaiian resident tests positive for Delta coronavirus variant https://t.co/0IJ8QfpYS9
Royal Caribbean cruise ship launch, sailings postponed after crew members test positive for COVID-19… https://t.co/VVrOdS6uEX
(A-1) Who tested positive that had close contact with Boris Johnson?
#news PM Boris Johnson in self-isolation after coming into contact with a lawmaker who tested positive for COVID-19 https://t.co/Kcy2X3M6vJ
Jair Bolsanaro has tested positive for Covid-19. Noval Djokovic and Boris Johnson had it. Life sometimes comes a full circle very fast.
WH says Trump spoke with Boris Johnson and "wished him a speedy recovery" after the British PM tested positive for coronavirus.
Boris Johnson's senior adviser, Dominic Cummings, is self-isolating at home after developing #coronavirus symptoms. http://bbc.in/2WQhbsZ Last week, the PM and Health Secretary Matt Hancock both tested positive for #Covid19. WATCH: https://bbc.in/2Jv55xj #Newsnight
(A-3) What methods of cure and prevention do people think are effective?
Very good indeed but you need also to remind them keeping social distancing, another basic protective measure to prevent the spread of #covid19.
Just like washing your hands is necessary to prevent from Coronavirus, inspecting your personal protective equipment https://t.co/xjY7FRgsV1
Two men in Georgia drank disinfectants in efforts to prevent COVID-19, officials say http://a.msn.com/01/en-us/BB13kJMw?ocid=st…

Table 5: Examples of correct extractions and errors returned by our knowledge base for sample queries. We use different colors for marking the types of extracted text spans (see Section 5.3 for more details for the error types): **correct extraction**, **classification errors**, **segmentation errors**, and **ambiguous cases**.

Query ID	# Corr / # All	Specific	Generic
S-1	99 / 100	63.6%	36.4%
S-2	91 / 100	75.8%	24.2%
A-1	29 / 50	100.0%	0.0%
A-2	48 / 50	6.2%	93.8%

Table 6: Analysis of answer types in response to the queries (where applicable) in Table 4. The percentage of generic answers varies significantly.

spections for all the returned results of our sample queries in Table 4. 67 incorrect extractions were identified in 750 extractions, which can be grouped into four major categories: classification errors (58.2%), segmentation errors (37.3%), ambiguous cases (13.9%) and others (4.5%). We present some examples of these errors in Table 5.

Classification Errors. We notice our BERT based model struggles with slots that may involve subtle inferences, such as relation or close contact, although the limited number of annotations for these slots might also be a factor in this type of error. For example, in the second tweet of query A-1 in Table 5, the tweet does not imply that *Jair Bolsanaro* was in close contact with *Boris Johnson*; in the third tweet of query A-1, the model fails to identify that *Boris Johnson* and *the British PM* refer to the same person.

Segmentation Errors. In some cases the extracted items contain extra tokens because of chunker errors, for example *georgia drank disinfectants* was extracted as a cure method. We also notice our choice of only extracting noun phrase chunks does not capture verb phrases for the CURE & PREVENTION category. For example, instead of extracting *washing your hands* and *don't touch your face* as prevention methods, our system only extracts *your hands* and *your face* (see query A-3 in Table 5).

Ambiguous Cases. In some cases, it is debatable whether an extraction is correct without additional context. For instance in the last tweet of query A-1 in Table 5, we do not know if *Dominic Cummings* tested positive, although the tweet seems to indicate that he might have been infected. We consider the extraction to be an error in this case, since the tweet did not specifically mention that he tested positive.

6 Case Studies

6.1 Correlation with Official Data Sources

To investigate whether statistics of events in COVIDKB correlate with official data sources, we plot the reported global positive cases and the number of extracted tested positive events from our knowledge base over time in Figure 3. Global reported positive numbers are from Center for Systems Science and Engineering at Johns Hopkins University.[9] We use 7-days moving average when drawing two time series curves. We observe that for both two waves in 2021 and current Omicron wave (highlighted in grey in Figure 3), our extracted events follow similar trend as actual reported cases globally and also show peaks. This analysis provides evidence to support quality of the

[9] https://github.com/CSSEGISandData/COVID-19

extracted information in COVIDKB, and suggests our knowledge base may contain information that could be used to analyze emerging dynamics of the pandemic. However as mentioned previously, the main use-case for COVIDKB is to enable semantic search to help journalists, epidemiologists or other professionals quickly analyze information posted on social media.

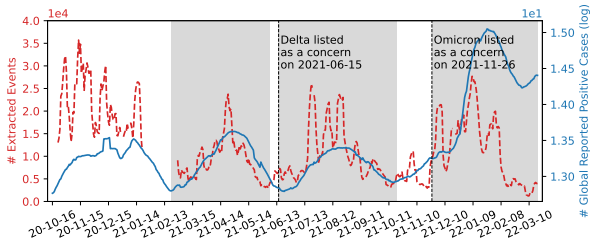

Figure 3: Number of extracted positive events and the actual global reported positive cases (log) show the similar trends in three waves (in grey). Data from 2021/01/21 to 2021/02/26 is missing due to technical issues.

6.2 Analyzing Claimed Cures and Preventions

Public's Attention Shifts over Time. Our knowledge base could also be helpful in monitoring public attention shifts regarding potential treatments and preventative measures over time. To demonstrate this, we analyze the top frequently mentioned potential cure and prevention methods that people believe are effective within different time ranges (a visualization of top 15 results are in Table A5). Time ranges are roughly divided to follow the global trends of the pandemic shown in Figure 3.

We observe people's opinions regarding certain cure and prevention methods remain unchanged throughout the whole pandemic, including *social distancing*, *hydroxychloroquine*, *(wash) your hands* and *masks*. As time proceeds, there is more focus on medical treatments. For example, *vaccine* and *vaccination* are more frequently discussed. Drugs also draw attention, especially in the last time range (from 2021/10/16 until now): we notice a variety of drugs appear in our knowledge base, including *fluvoxamine*, *monoclonal antibodies*, *AstraZeneca antibody drug* and *Israeli drug*.

We note not all above methods are actually effective for coronavirus. Researchers hold a mixed view for treatments such as *hydroxychloroquine* and *ivermectin*.[10] This type of automatically extracted information in COVIDKB could be helpful to track the spread of misinformation online.

Who is promoting cures? We also analyze the returned results from query S-2 to understand who is promoting cures. A variety of people and organizations are observed, most frequent 10 of which are *Donald Trump*, *China*, *scientists*, *CDC*, *White House*, *Jim Bakker*, *Pfizer*, *Madagascar*, *Dr. Fauci*, and *Bill Gates*.

7 Conclusion

In this paper, we presented a corpus of 10,000 tweets annotated with 5 types of events and 28 slots. We showed that our corpus supports automatic extraction of COVID-19 events using supervised learning. By aggregating extractions over millions of tweets, our approach can accurately answer a range of structured queries about events that are publicly reported in real-time on Twitter. Our knowledge base could be a useful tool for epidemiologists, journalists and policymakers to more efficiently track the spread of this new disease. This work also presents a case-study on how an information extraction system can be rapidly developed for a new domain in response to an emerging crisis. For example, our methodology could be applied to develop knowledge bases for natural disasters (Spiliopoulou et al., 2020) or future disease outbreaks.

Ethical Considerations

This study was conducted under the approval of the Institutional Review Board (IRB) of our university and complies with Twitter's terms of service. Following Twitter's policy for content redistribution, we will only release our annotated corpus that contains Tweet IDs (not Tweet Objects) and a list of character offsets corresponding to the annotated mentions. We will not release any user information or demographic data. Our event extractors produce structured representations of information that was explicitly and publicly stated. We do not derive or infer any potentially sensitive characteristics or health information that may violate users' privacy. Almost all events that are currently indexed by our knowledge base come from public news reports.

[10] For example, Ivermectin has been used in clinical trials: https://www.covid19treatmentguidelines.nih.gov/therapies/antiviral-therapy/ivermectin/. However, it is not approved or authorized by FDA: https://www.fda.gov/consumers/consumer-updates/why-you-should-not-use-ivermectin-treat-or-prevent-covid-19.

To further protect users' privacy, we specifically designed two slot-filling questions during annotation in order to detect and remove cases where users publicly report information about themselves, or a person with whom they have a close relationship.

Our knowledge base should be used with caution, as we note the Twitter users are not representative samples of the total population; posts from Twitter users are also not necessarily representative samples of public opinions (Wojcik and Hughes, 2019). As Twitter Stream API provides only 1% of all public tweets, our knowledge base naturally is not able to index all reported cases online. Our extractors may contain other unknown biases due to data collection process, for example they might perform worse on African American English. All these limitations should be taken into consideration in any application that makes use of our data.

Acknowledgements

We would like to thank the anonymous reviewers for their valuable suggestions. This material is based in part on research sponsored by the NSF (IIS-1845670) and IARPA via the BETTER program (2019-19051600004), DARPA via the ARO (W911NF-17-C-0095) in addition to an Amazon Research Award. The views and conclusions contained herein are those of the authors and should not be interpreted as necessarily representing the official policies, either expressed or implied, of IARPA, or the U.S. Government. The U.S. Government is authorized to reproduce and distribute reprints for government purposes notwithstanding any copyright annotation therein.

References

Cosmin Adrian Bejan and Sanda Harabagiu. 2014. Unsupervised event coreference resolution. *Computational Linguistics*.

Aida Amini, Tom Hope, David Wadden, Madeleine van Zuylen, Eric Horvitz, Roy Schwartz, and Hannaneh Hajishirzi. 2021. Extracting a knowledge base of mechanisms from COVID-19 papers. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*.

Ron Artstein and Massimo Poesio. 2008. Inter-coder agreement for computational linguistics. *Computational Linguistics*.

Yoav Artzi and Luke Zettlemoyer. 2011. Bootstrapping semantic parsers from conversations. In *Proceedings of the 2011 Conference on Empirical Methods in Natural Language Processing*, pages 421–432.

Juan M. Banda, Ramya Tekumalla, Guanyu Wang, Jingyuan Yu, Tuo Liu, Yuning Ding, and Gerardo Chowell. 2020. A large-scale COVID-19 Twitter chatter dataset for open scientific research – an international collaboration. *arXiv preprint arXiv:2004.03688*.

Edward Benson, Aria Haghighi, and Regina Barzilay. 2011. Event discovery in social media feeds. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*.

Jonathan Berant, Andrew Chou, Roy Frostig, and Percy Liang. 2013. Semantic parsing on freebase from question-answer pairs. In *Proceedings of the 2013 conference on empirical methods in natural language processing*, pages 1533–1544.

Nathanael Chambers, Ben Fry, and James McMasters. 2018. Detecting denial-of-service attacks from social media text: Applying NLP to computer security. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*.

Ching Yun Chang, Zhiyang Teng, and Yue Zhang. 2016. Expectation-regulated neural model for event mention extraction. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*.

Emily Chen, Kristina Lerman, and Emilio Ferrara. 2020. Tracking social media discourse about the COVID-19 pandemic: Development of a public coronavirus twitter data set. *JMIR Public Health Surveill*.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*.

Dimitar Dimitrov, Erdal Baran, Pavlos Fafalios, Ran Yu, Xiaofei Zhu, Matthäus Zloch, and Stefan Dietze. 2020. TweetsCOV19 – a knowledge base of semantically annotated tweets about the COVID-19 pandemic. *arXiv preprint arXiv:2006.14492*.

Jeremy Ginsberg, Matthew H Mohebbi, Rajan S Patel, Lynnette Brammer, Mark S Smolinski, and Larry Brilliant. 2009. Detecting influenza epidemics using search engine query data. *Nature*, 457(7232):1012–1014.

Tamanna Hossain, Robert L. Logan IV, Arjuna Ugarte, Yoshitomo Matsubara, Sean Young, and Sameer Singh. 2020. COVIDLies: Detecting COVID-19 misinformation on social media. In *Proceedings of the 1st Workshop on NLP for COVID-19 at EMNLP 2020*.

Yong Hu, Heyan Huang, Anfan Chen, and Xian-Ling Mao. 2020. Weibo-COV: A large-scale COVID-19 social media dataset from Weibo. In *Proceedings of the 1st Workshop on NLP for COVID-19 (Part 2) at EMNLP 2020*, Online. Association for Computational Linguistics.

Heng Ji, Ralph Grishman, and Hoa Dang. 2011. Overview of the TAC2011 knowledge base population track. In *TAC 2011 Proceedings Papers*.

Daniel Jurafsky and James H Martin. 2000. Speech and language processing: An introduction to natural language processing, computational linguistics, and speech recognition.

Twin Karmakharm, Nikolaos Aletras, and Kalina Bontcheva. 2019. Journalist-in-the-loop: Continuous learning as a service for rumour analysis. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing: System Demonstrations*.

Diederick P Kingma and Jimmy Ba. 2015. Adam: A method for stochastic optimization. In *International Conference on Learning Representations*.

David Lazer, Ryan Kennedy, Gary King, and Alessandro Vespignani. 2014. The parable of google flu: traps in big data analysis. *Science*, 343(6176):1203–1205.

Grace E. Lee and Aixin Sun. 2019. A study on agreement in pico span annotations. In *Proceedings of the 42nd International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR'19, page 1149–1152, New York, NY, USA. Association for Computing Machinery.

Marco Lui and Timothy Baldwin. 2012. langid.py: An off-the-shelf language identification tool. In *Proceedings of the ACL 2012 System Demonstrations*, pages 25–30, Jeju Island, Korea. Association for Computational Linguistics.

James McCorriston, David Jurgens, and Derek Ruths. 2015. Organizations are users too: Characterizing and detecting the presence of organizations on twitter. In *Proceedings of the 9th International AAAI Conference on Weblogs and Social Media*.

Bonan Min and Ralph Grishman. 2012. Compensating for annotation errors in training a relation extractor. In *Proceedings of the 13th Conference of the European Chapter of the Association for Computational Linguistics*.

Martin Müller, Marcel Salathé, and Per E Kummervold. 2020. COVID-Twitter-BERT: A natural language processing model to analyse COVID-19 content on twitter. *arXiv preprint arXiv:2005.07503*.

Dat Quoc Nguyen, Thanh Vu, Afshin Rahimi, Mai Hoang Dao, Linh The Nguyen, and Long Doan. 2020. WNUT-2020 Task 2: Identification of Informative COVID-19 English Tweets. In *Proceedings of the 6th Workshop on Noisy User-generated Text*.

Michael J Paul, Mark Dredze, and David Broniatowski. 2014. Twitter improves influenza forecasting. *PLOS Currents Outbreaks*.

Umair Qazi, Muhammad Imran, and Ferda Ofli. 2020. GeoCOV19: A dataset of hundreds of millions of multilingual COVID-19 tweets with location information. *SIGSPATIAL Special*.

Pranav Rajpurkar, Jian Zhang, Konstantin Lopyrev, and Percy Liang. 2016. SQuAD: 100,000+ questions for machine comprehension of text. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*.

Alan Ritter, Sam Clark, Mausam, and Oren Etzioni. 2011. Named entity recognition in tweets: An experimental study. In *Proceedings of the 2011 Conference on Empirical Methods in Natural Language Processing*.

Alan Ritter, Mausam, Oren Etzioni, and Sam Clark. 2012. Open domain event extraction from Twitter. In *Proceedings of the 18th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*.

Alan Ritter, Evan Wright, William Casey, and Tom Mitchell. 2015. Weakly supervised extraction of computer security events from Twitter. In *Proceedings of the 24th International Conference on World Wide Web*.

Evangelia Spiliopoulou, Salvador Medina Maza, Eduard Hovy, and Alexander G Hauptmann. 2020. Event-related bias removal for real-time disaster events. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 3858–3868.

Peter Stefanov, Kareem Darwish, Atanas Atanasov, and Preslav Nakov. 2020. Predicting the topical stance and political leaning of media using tweets. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*.

Alane Suhr, Srinivasan Iyer, and Yoav Artzi. 2018. Learning to map context-dependent sentences to executable formal queries. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*.

James Thorne, Andreas Vlachos, Christos Christodoulopoulos, and Arpit Mittal. 2018. FEVER: a large-scale dataset for Fact Extraction and VERification. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*.

Onur Varol, Emilio Ferrara, Clayton Davis, Filippo Menczer, and Alessandro Flammini. 2017. Online human-bot interactions: Detection, estimation, and characterization. In *Proceedings of the Eleventh International AAAI Conference on Web and Social Media*.

Deepak Venugopal, Chen Chen, Vibhav Gogate, and Vincent Ng. 2014. Relieving the computational bottleneck: Joint inference for event extraction with high-dimensional features. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing*.

Stefan Wojcik and Adam Hughes. 2019. Sizing up twitter users. Technical report, Pew Internet and American Life Project.

Zhilin Yang, Peng Qi, Saizheng Zhang, Yoshua Bengio, William Cohen, Ruslan Salakhutdinov, and Christopher D. Manning. 2018. HotpotQA: A dataset for diverse, explainable multi-hop question answering. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 2369–2380, Brussels, Belgium. Association for Computational Linguistics.

Edwin Zhang, Nikhil Gupta, Rodrigo Nogueira, Kyunghyun Cho, and Jimmy Lin. 2020. Rapidly deploying a neural search engine for the COVID-19 open research dataset: Preliminary thoughts and lessons learned. *arXiv preprint arXiv:2004.05125*.

Deyu Zhou, Xuan Zhang, and Yulan He. 2017. Event extraction from Twitter using non-parametric Bayesian mixture model with word embeddings. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics*.

Shi Zong, Alan Ritter, Graham Mueller, and Evan Wright. 2019. Analyzing the perceived severity of cybersecurity threats reported on social media. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*.

A Dataset

A.1 Keywords for Data Collection

We provide the keywords used for collecting data along with starting date in Table A1. Keywords in our experiments are carefully chosen to both have a wide coverage of tweets with different linguistic phenomena and have a good precision of collecting tweets that are relevant to our tasks.

Event Type	Start From	Keywords
TESTED POSITIVE	2020/01/15	(test OR tests OR tested) positive AND VIRUS
TESTED NEGATIVE	2020/02/15	(test OR tests OR tested) negative AND VIRUS
CAN NOT TEST	2020/01/15	(can't OR can not) get (tested OR test OR tests) (can't OR can not) be tested (couldn't OR could not) get (tested OR test OR tests) (couldn't OR could not) be tested
DEATH	2020/02/15	(died OR pass away OR passed away) AND VIRUS
CURE & PREVENTION	2020/03/01	(cure OR prevent) AND VIRUS

Table A1: Keywords used for each event type. We consider the following variants for VIRUS: VIRUS = (COVID19 OR COVID-19 OR corona OR coronavirus).

A.2 Data Annotation

The complete slot filling questions used for annotating COVID-19 events are listed in Table A2. We also provide the annotation interface shown to Mechanical Turk workers in Figure A1.

Event Type	Slot Name	Slot Filling Questions
TESTED POSITIVE —— TESTED NEGATIVE	who	Who tested positive (negative)?
	close contact	Who was in close contact with the person who tested positive (negative)?
	relation	Does the affected person have a relationship with the author of the tweet?
	employer	Who is the employer of the person who tested positive?
	recent travel	Where did the people who tested positive recently visit?
	when	When were positive (negative) cases reported?
	where	Where were positive (negative) cases reported?
	age	What is the age of the people who tested positive (negative)?
	duration	How long did it take to know the result of the test?
CAN NOT TEST	who	Who can not get a test?
	relation	Does the untested person have a relationship with the author of the tweet?
	when	When was the person unable to obtain a test?
	where	Where was the person unable to obtain a test?
	symptoms	Is the affected person currently experiencing any COVID-19 related symptoms?
DEATH	who	Who died from COVID-19?
	relation	Does the deceased person have a personal relationship with the author of the tweet?
	when	When was the death reported?
	where	Where was the death reported?
	age	What is the age of the person who died?
CURE & PREVENTION	opinion	Does the author of the tweet believe cure/prevention is effective?
	what	Which method of cure/prevention is mentioned?
	who	Who is promoting the cure or prevention?

Table A2: Slot filling questions used for annotating COVID-19 events.

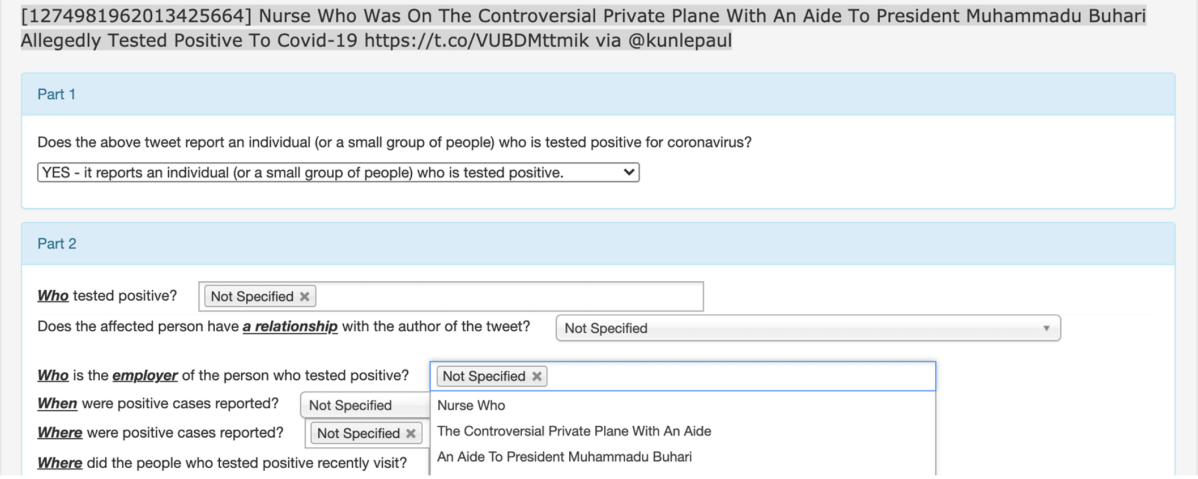

Figure A1: Main portion of the annotation interface shown to Mechanical Turk workers for annotating TESTED POSITIVE events.

A.3 Annotated Samples

Examples of our annotated tweets are presented in Table A3.

Event Type	Tweet	Annotations
POSITIVE	*#Karnataka | A 26-year-old man returning from #Greece tested positive for #COVID19, becoming the fifth positive case in the state, a health official said on Thursday. #CoronavirusPandemic #COVID #COVID19india* `[URL]`	WHO AGE WHERE RECENT V.
NEGATIVE	*Live updates: Boris Johnson tested negative for Covid-19 on leaving hospital, says Downing Street #coronavirus*	WHO
DEATH	*'#TopChef Masters' winner Floyd #Cardoz dies after #coronavirus diagnosis' "World-renowned chef Floyd Cardoz died Wednesday in New Jersey at age 59." "Cardoz admitted himself to the hospital on March 17 after feeling feverish."*	WHO AGE WHERE WHEN
CAN NOT TEST	*Nurse working in ITU couldn't get tested, & was told that the test was "very expensive", so he couldn't have a test.* `[URL]` ...	WHO

Table A3: Examples of our annotated tweets.

B COVIDKB Knowledge Base

B.1 Statistics of Our Knowledge Base

We report the number of extracted events along with the breakdown statistics for each slot in Table A4.

Event Types	# Extracted	Number of Events per Slot											
		who	relation	when	where	age	close contact	employer	recent travel	duration	symptoms	opinion	what
TESTED POS	2,354,363	2,098,964	164,126	81,053	602,552	32,361	122,952	264,275	84,157	–	–	–	–
TESTED NEG	411,071	387,354	47,325	17,044	28,447	851	7,733	–	–	9,049	–	–	–
CAN NOT TEST	30,552	26,468	17,432	94	7,637	–	–	–	–	–	14,881	–	–
DEATH	779,074	629,323	91,121	164,282	230,672	143,270	–	–	–	–	–	–	–
CURE & PREV.	665,422	319,077	–	–	–	–	–	–	–	–	–	270,493	461,290
Total	4,240,482	3,461,186	320,004	262,473	869,308	176,482	130,685	264,275	84,157	9,049	14,881	270,493	461,290

Table A4: Number of extracted events, with a breakdown for each slot in our knowledge base. Slot filling questions that are not applied to specific event types are marked with "–".

B.2 Interface of Our Knowledge Base

Our structured query interface of the knowledge base is presented in Figure A2.

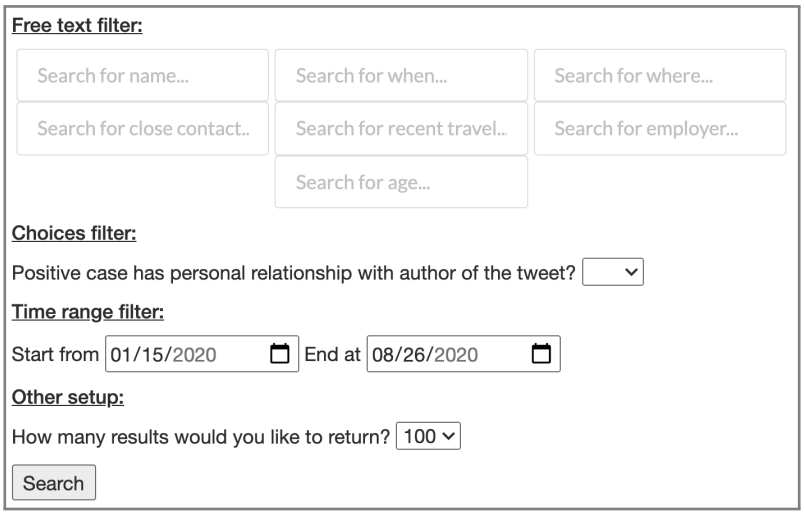

Figure A2: Structured query interface of our knowledge base.

B.3 Public Attention Shifts for Cure and Prevention Methods over Time

We present the top 15 frequently mentioned potential cure and prevention methods that people believe are effective within different time ranges in Table A5. Larger fonts indicate more frequent terms.

(A-3) What methods of cure and prevention do people think are effective?	
Before 2021/01/01	**From 2021/02/15 to 2021/06/15 (First Wave in 2021)**
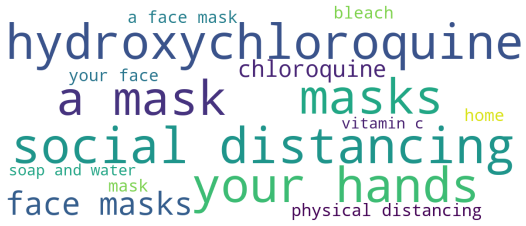	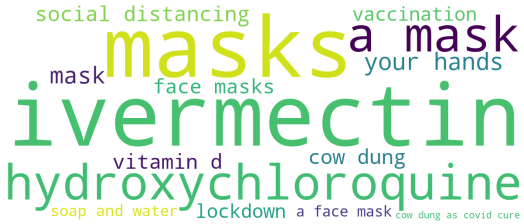
From 2021/06/16 to 2021/10/15 (Second Wave in 2021)	**From 2021/10/16 to 2022/04/01**
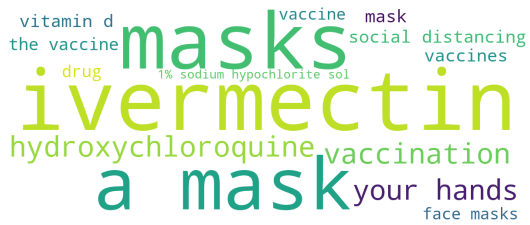	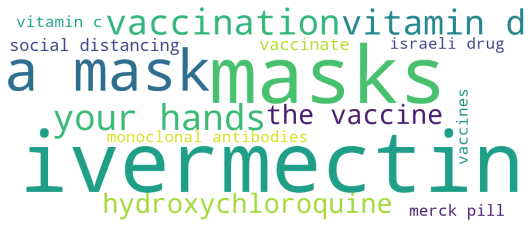

Table A5: Top 15 most frequent potential cure and prevention methods that people think are effective over different time ranges.

Accounting for Language Effect in the Evaluation of Cross-lingual AMR Parsers

Shira Wein
Georgetown University
sw1158@georgetown.edu

Nathan Schneider
Georgetown University
nathan.schneider@georgetown.edu

Abstract

Cross-lingual Abstract Meaning Representation (AMR) parsers are currently evaluated in comparison to gold English AMRs, despite parsing a language other than English, due to the lack of multilingual AMR evaluation metrics. This evaluation practice is problematic because of the established effect of source language on AMR structure. In this work, we present three multilingual adaptations of monolingual AMR evaluation metrics and compare the performance of these metrics to sentence-level human judgments. We then use our most highly correlated metric to evaluate the output of state-of-the-art cross-lingual AMR parsers, finding that Smatch may still be a useful metric in comparison to gold English AMRs, while our multilingual adaptation of S2match (XS2match) is best for comparison with gold in-language AMRs.

1 Introduction

The Abstract Meaning Representation (AMR; Banarescu et al., 2013) formalism captures the meaning of a sentence or phrase as a rooted, directed acyclic graph. Nodes correspond to concepts and the labeled edges reflect the relations between concepts. For example, the annotation in Figure 1 features the AMR annotation of the sentence "we will try not to make a mistake" in both PENMAN (text-based) and graph form. The edge labels can be arguments (core roles, denoted as `:argN`), or one of a number of non-core roles such as `:location` or `:manner`.

Cross-lingual AMR parsers convert non-English text to (English-focused) AMR graphs. As there are no existing multilingual AMR evaluation metrics and due to the limited availability of non-English gold AMR annotations, these cross-lingual AMR parsers have only ever been evaluated in comparison to gold *English* AMRs. This established approach of comparison to English AMRs using the monolingual Smatch metric needs to be considered more carefully because previous work has established that the source language has a dramatic effect on the cross-lingual AMRs (Damonte, 2019; Wein and Schneider, 2021, 2022).

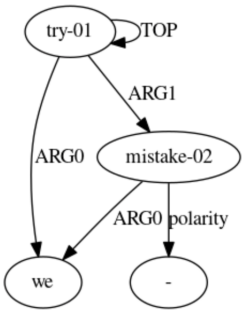


```
(v2 / try-01
    :ARG0 (v1 / we)
    :ARG1 (v3 / mistake-02
        :polarity -
        :ARG0 v1))
```

Figure 1: An AMR annotation for the sentence "We will try not to make a mistake," in PENMAN text-based notation and as a rooted graph. The diagram was made on the AMREager website (Damonte et al., 2017).

We argue that cross-lingual AMR parsing should represent the *semantics*, beyond the lexicon, faithfully to the source language. Enabled by work developing sizable Spanish (Wein et al., 2022) and Chinese (Li et al., 2016) gold AMR corpora, we propose that cross-lingual AMR parsers should be evaluated on gold AMRs that match the language being parsed. In this work, we adapt three monolingual AMR evaluation metrics to a multilingual setting, and evaluate the performance of these metrics in comparison to human judgments of cross-lingual sentence similarity. We show that our adaptation of S2match (Opitz et al., 2020) which leverages LaBSE (Feng et al., 2022) embeddings is highly correlated with, and most correlated to, human judgment of similarity.

Proceedings of the 29th International Conference on Computational Linguistics, pages 3824–3834
October 12–17, 2022.

Additionally, using our new cross-lingual evaluation metric, we evaluate the performance of cross-lingual AMR parsers and compare that with evaluations using the previously-used Smatch metric. This provides a new, informative ranking of existing cross-lingual AMR parsers and offers insight into the applicability of the monolingual Smatch metric for cross-lingual evaluation.

Ours is the first work to address the evaluation of cross-lingual AMR parsers with respect to the language being parsed. Our contributions include:

- Three multilingual adaptations of monolingual AMR evaluation metrics.
- Human judgments on Spanish-English and Chinese-English sentence pairs, corresponding to gold AMR pairs.
- An assessment of the effectiveness of existing monolingual and proposed multilingual metrics to evaluate similarity between cross-lingual AMR pairs, and correlations with human judgments for these metrics.
- An evaluation of state-of-the-art cross-lingual AMR parsers using both our new XS2match metric and Smatch, in comparison to gold English *and* gold Spanish AMRs.

Our code is available online at `https://github.com/shirawein/Crossling-AMR-Eval` to promote ease of cross-lingual AMR evaluation using our metrics.

2 Background

Abstract Meaning Representation parsers produce AMR annotations from natural text. "Cross-lingual AMR parsing" refers to parsing a non-English sentence into a standard English AMR (Damonte, 2019). Damonte and Cohen (2018) introduced the task of cross-lingual AMR parsing and developed non-English parsers by projecting from English annotations to their non-English counterparts through the use of parallel corpora. Current approaches to cross-lingual AMR parsing evaluate AMRs produced from text in four languages: Mandarin Chinese, Spanish, Italian, and German (§5.1). Language-specific AMR parsers have also been developed to parse from Mandarin Chinese (Wang et al., 2018), Portuguese (Anchiêta and Pardo, 2018), and Indonesian (Roaffa Ilmy and Leylia Khodra, 2021).

AMR parsers have traditionally been evaluated using the Smatch metric (Cai and Knight, 2013). In order to compare a pair of semantic graphs (system and gold), Smatch aligns their nodes, searching for a maximal alignment via hill climbing. With these alignments, triples representing edges of the two graphs are compared to compute an F-score.

Cross-lingual AMR parsers have been evaluated on gold English AMR graphs created from English sentences, paired with sentences that were manually translated from those English sentences into other languages. The English AMRs come from the AMR 2.0 dataset (Knight et al., 2017), and the non-English sentence translations from the AMR 2.0 - Four Translations dataset (Damonte and Cohen, 2020) with translations into Mandarin Chinese, Italian, Spanish, and German. Thus, cross-lingual AMR parsers have been evaluated by comparing the system-produced AMR parsed from the non-English sentence to the gold English AMR corresponding to the translated sentence—which does not take into account any effect the source language might have on AMR structure. We discuss this concern in §3.

In addition to the resources noted above, recent work has produced gold Spanish AMRs for AMR 2.0 - Four Translations Spanish sentences (Wein et al., 2022). As a result, we can compare the system output to in-language (Spanish) gold AMRs, and develop metrics to enable this comparison.

3 Developing Cross-lingual AMR Evaluation Metrics

When comparing English and non-English AMRs, the concepts themselves are in different languages and the structure of the AMR will also differ, as it is affected by the syntax and semantics of the language being parsed from (Damonte, 2019; Wein and Schneider, 2021; Blloshmi et al., 2020). Therefore, to be able to evaluate the similarity of AMRs in two different languages, there are likely changes that need to be made to the monolingual metric.

A naive assumption is that AMR should be structurally the same for parallel sentences regardless of language, because AMR encodes meaning and translation preserves meaning. However, previous work has demonstrated that this is not the case, and that even when lexical items are made to be monolingual, the source language has a marked effect on the AMR structure itself (for at least English and Chinese): Wein and Schneider (2022) reported Smatch scores consistently below 50% between English and Chinese parallel gold AMR graphs, even when all Chinese tokens are replaced by their

Original gold English AMR:

```
(s / surge-01
  :ARG1 (a / and
          :op1 (s2 / speed-01)
          :op2 (a2 / accident))
  :mod (a3 / as-well))
```

Original gold Spanish AMR:

```
(c0 / aumentar-01
    :manner (c1 / también)
    :ARG1 (c2 / y
              :op1 (c3 / exceder-01
                       :ARG1 (c4 / velocidad))
              :op2 (c5 / accidente)))
```

Our translated version of the gold Spanish AMR:

```
(c0 / increase
    :manner (c1 / also)
    :ARG1 (c2 / and
              :op1 (c3 / exceed
                       :ARG1 (c4 / speed))
              :op2 (c5 / accident)))
```

Figure 2: Parallel gold English and Spanish AMRs for the sentence "Speeding and accidents have surged as well" from Knight et al. (2017) and Wein et al. (2022) respectively, followed by our translated version of the gold Spanish AMR per the cross-lingual Smatch (XSmatch) method.

corresponding English AMRs. Therefore, we alter existing metrics to be able to compare (English) AMRs parsed from non-English sentences to gold in-language AMRs.

We consider the applicability of existing AMR metrics in cross-lingual parser evaluation and adapt them to function multilingually. Cross-lingual AMR parsers are currently evaluated via Smatch. Here we consider three metrics: Smatch, SemBleu, and S2match.

As mentioned in §2, Smatch aligns the semantic graphs via hill climbing. S2match (Opitz et al., 2020) incorporates word embeddings into Smatch to account for similarity of concept nodes without the same token being used. SemBleu (Song and Gildea, 2019) is based on the machine translation metric BLEU (Papineni et al., 2002). SemBleu does not involve variable alignment and instead converts the graph to a bag of k-grams.

Broadly, our approach to adapting these monolingual metrics is that we alter Smatch (§3.1) and SemBleu (§3.2) by translating the lexical material in the AMR graphs into English, and S2match (§3.3) by using cross-lingual embeddings.

3.1 XSmatch

In order to make Smatch (Cai and Knight, 2013) multilingual, we translate individual tokens within the non-English AMR to English. We use the EasyNMT package[1] for translation, which was also the translation package used in the cross-lingual parser of Uhrig et al. (2021). Specifically, we employ the Opus-MT model. Recall that Smatch (like the other evaluation metrics) compares AMR graphs rather than strings. Therefore, we are translating individual elements of the AMR and not the sentence itself. The elements of the AMR which we translate are the words in the instance and attribute triples. We also remove the word senses (numeric affixes to the concepts) for ease of translation and comparison. An example parallel gold English and gold Spanish AMR, plus our corresponding translated version of the gold Spanish AMR, can be seen in Figure 2.

We also developed a version of Smatch that aligns concepts across AMRs in different languages via fast_align (Dyer et al., 2013), and found that using machine translation was more reliable.

3.2 XSemBleu

To adapt SemBleu to function cross-lingually, we again translate the tokens in one of the AMRs, and additionally truncate the tokens (truncate after translation for the non-English AMR, and also truncate for the English AMR). SemBleu does not break the AMR into triples, so we instead translate the entire non-English AMR to an English AMR by iterating token by token over the AMR and determining whether the current token needs to be translated. For example, parentheses, digits, and roles starting with a colon do not need to be translated. This approach to translation is more intensive than the translation required for individual tokens in XSmatch. Therefore, we aim to account for translation discrepancies and errors (e.g. part-of-speech discrepancies) by truncating the translations to the first `n` tokens. In this case we use `n=5`. We use the default weights and smoothing function.

We suspected that SemBleu may be a better fit for cross-lingual AMR comparison because SemBleu prioritizes content over graph structure. One potential issue with SemBleu is that the nodes with higher connectivity are disproportionately weighted (Opitz et al., 2020).

[1] https://github.com/UKPLab/EasyNMT

3.3 XS2match

The current implementation of S2match relies on an external text file of embeddings, with the token being paired to an embedding in the file, and the embedding being retrieved from the text file for each token. To transport S2match to a multilingual format, we make use of the LaBSE (Feng et al., 2022) preprocesser and encoder. Where the existing S2match approach retrieved the embedding for a token from a text file, we elicit a constant tensor of the word, preprocess it, and encode it to a LaBSE embedding. Finally, we normalize the embedding and convert it to a numpy vector.

Consequently, we adjust the similarity computation when comparing the individual units of the two AMRs to matrix multiplication of two vectors, with one vector transposed.

We also trialed our approach with multilingual BERT (Devlin et al., 2018) and found that LaBSE was a more effective solution, though it takes longer to run than using multilingual BERT.

A benefit of XS2match is that, unlike XSmatch and XSemBleu, it does not rely on neural machine translation practices that could unduly benefit a parser using the same translation tool (e.g. Uhrig et al. (2021)) through exact lexical matching. When using XSmatch or XSemBleu to evaluate cross-lingual parser performance, it is worth verifying whether the translation approach is the same for the metric and the parser.

4 Analysis of Metrics

To compare cross-lingual AMR metrics, we need a source of ground truth about how the sentences in a translation pair relate to one another. For this we utilize human judgments of cross-lingual similarity. Because AMR is a meaning representation, the similarity scores of cross-lingual AMR pairs ideally should correlate with the similarity scores of their associated sentence pair. In line with previous work (Opitz et al., 2020), we determine the accuracy of our AMR metrics by calculating Pearson's correlation between system output and human judgments of cross-lingual sentence similarity. Specifically, we use gold AMRs as input to the metrics and calculate how correlated the metric-based similarity scores are to the human similarity ratings for the corresponding sentence pair. We normalize the AMRs by removing all wikification (links to the associated Wikipedia pages for entities in the AMR).

4.1 Collection of Human Judgments

We collect human judgments for 100 Spanish-English sentence pairs and 150 Mandarin Chinese-English sentence pairs which have associated gold AMRs. Both sets of data are doubly annotated by speakers fluent in both English and Chinese / Spanish.

We use both language pairs because Spanish and Chinese are notably syntactically distinct languages, and vary noticeably in cross-lingual AMR performance (§5.1). We also only use sentences which have associated gold AMRs, as opposed to existing sentence similarity metrics (Agirre et al., 2016), because we want to avoid introducing noise by relying on automatic parsers when comparing the AMR similarity with sentence similarity, or biasing our later assessment of cross-lingual parsers towards the parsers being used.

The sentences used come from the Chinese annotations of *The Little Prince* (Li et al., 2016) and the Spanish annotations (Wein et al., 2022) of AMR 2.0 - Four Translations (Damonte and Cohen, 2020). The parallel English sentences for both the Chinese and Spanish sentences are very related in meaning to the non-English sentences, so it was necessary to construct a dataset with varying degrees of sentence similarity (with all sentences still having associated gold AMRs).

In order to construct a Spanish-English dataset of varying similarities, 100 Spanish sentences from different genres in Damonte and Cohen (2020) were chosen. Then, a portion (25%) of the sentences were paired with English (from Knight et al., 2017) sentences with minimal to no similarity. Half of the sentences were paired with English sentences having a moderate amount of similarity / some divergence, as determined by being from the same relative part of a text and discussing the same topic without being a parallel sentence. The remaining 25% of the sentences were then paired with their parallel English sentences.

A similar approach was used when construct-

Annotator	0	1	2	3	4	5
Zh-Eng Anno. 1	35	15	5	17	41	37
Zh-Eng Anno. 2	40	9	3	10	25	63
Es-Eng Anno. 1	41	17	7	7	1	27
Es-Eng Anno. 2	34	15	14	8	6	23

Table 1: Distribution of human judgments of sentence similarity from 0-5 for each annotator. Zh-Eng annotators provided 150 judgments and Es-Eng annotators provided 100 judgments.

	Smatch	XSmatch	SemBleu	XSemBleu	XS2match	BERTscore
Zh-Eng Anno. 1	0.43	0.40	0.20	0.42	0.51	0.76
Zh-Eng Anno. 2	0.38	0.40	0.21	0.40	0.50	0.72
Zh-Eng Anno. Sum	0.41	0.41	0.21	0.42	0.51	0.75
Zh-Eng BERTscore	0.46	0.39	0.25	0.38	0.52	1.00
Es-Eng Anno. 1	0.69	0.79	0.37	0.60	0.77	0.87
Es-Eng Anno. 2	0.72	0.82	0.39	0.63	0.81	0.86
Es-Eng Anno Sum	0.72	0.82	0.38	0.63	0.80	0.88
Es-Eng BERTscore	0.74	0.82	0.41	0.62	0.79	1.00

Table 2: Pearson's correlation scores between the evaluation metrics (in the columns, along with BERTscores) and the human judgments of similarity (in the rows, with BERTscore, again). For each language pair, being Chinese-English and Spanish-English, we get the correlation with each of the two annotators as well as the sum of the similarity judgments.

ing the Chinese-English dataset, with 66% of the dataset being mostly parallel and 33% of the dataset being mostly divergent.

We asked human annotators to provide a score from 0 to 5 of how similar the content of a Spanish-English or Chinese-English sentence pair is. We use the task instructions from Agirre et al. (2016) as the basis for our instructions, "where 0 represents two sentences that are unrelated in meaning, and 5 indicates that the two sentences are perfect paraphrases of each other". We also added in degrees of similarity to the instructions to add clarity:

- (0) Completely unrelated
- (1) Not equivalent but share few subjects
- (2) Not equivalent but share some details
- (3) Roughly equivalent
- (4) Equivalent except for some details
- (5) Completely equivalent

We find that agreement for our sentence similarity protocol is high, with the correlation between annotator judgments being 0.93 for both the Spanish-English annotations and the Chinese-English annotations.[2] The distribution of the sentence similarity scores is not uniform (table 1).[3]

4.2 Results of Correlation Analysis

In order to assess the applicability of our metrics for cross-lingual AMR evaluation, we calculate Pearson's correlation for the human sentence similarity judgments and the AMR metrics. We also compare with the sentence-based metric BERTscore as a point of reference. These results can be seen in table 2.

[2]Annotator agreement for the SemEval task (Agirre et al., 2016) is not reported.

[3]For the Chinese-English sentences, we initially collected judgments from a third annotator, but that annotator's interpretation of similarity was skewed towards saying most of the valid translations were completely equivalent, so the data was not informative for studying degrees of similarity. As a result we used the data from two other annotators.

First, note that the use of translation is beneficial in SemBleu for both language pairs and in Smatch for Spanish-English. Applying translation to Chinese-English data has little effect on Smatch, for reasons discussed later in this section.

Comparing the three cross-lingual metrics, the two with the highest correlation to human judgment of sentence similarity are XSmatch and XS2match. While the correlation for Spanish-English is similar for those two multilingual metrics, though slightly higher via XSmatch, the correlation for Chinese-English is substantially higher using XS2match. As a result, we recommend that XS2match is likely the best metric to use for cross-lingual AMR parser evaluation.

Notably, though perhaps unsurprisingly, correlation with the Chinese-English human annotations is lower for all metrics than correlation with the Spanish-English human annotations. This is likely not due to any issues with the human annotation itself, because the annotations still correlate well with BERTscore judgment of similarity, as seen in the final column of table 2. Nonetheless, the Chinese-English human annotations are less correlated with BERTscore than the Spanish-English human annotations. Instead, the lower correlation with the Chinese-English annotations is likely due to lower performance on Chinese for the automatic machine translation systems and embeddings, as well as a greater degree of dissimilarity between the Chinese and English parallel AMRs than between the Spanish and English parallel AMRs. This greater degree of dissimilarity for certain AMR pairs has been studied previously (Xue et al., 2014) and is also evidenced here by the difference in the Smatch column in table 2. The baseline Smatch similarity, with no multilingual component, is already much more correlated with human judgments for Spanish-English than for Chinese-English.

The monolingual Smatch score is already highly correlated with sentence similarity (for English-Spanish in particular, but for both language pairs) because of structural similarity between the AMRs and matching between a subset of non-lexical nodes. For example, the Smatch scores aren't relying on lexical items as much as they are relying on the entities, e.g. shared name entities. This presence of names and named entities may also affect these correlation scores across languages because the Spanish-English text is from the news domain, which includes many country and person names, whereas the Chinese-English text is *The Little Prince*, which includes fewer of these named entities. This finding is a benefit of our approach to consider two different languages and text domains in our correlation analysis; as a result our recommendation to use XS2match for cross-lingual AMR evaluation is a more robust one.

Even with the translation and truncation practices, XSemBleu correlation does not exceed XS2match correlation for either language pair. We hoped that SemBleu might be able to overcome structural differences between cross-lingual AMR pairs, but the undesirable presence of bias in the metric, which cannot be overcome without introducing a different bias (Opitz et al., 2020), likely led to the consequence of correlating less with the human annotations than the other metrics. Still, XSemBleu correlates fairly well with both language pairs.

We also measure correlation with scores from the BERTscore metric (Zhang et al., 2020), which uses the sentences directly and not the AMR graphs. BERTscore uses BERT-based models to compare embeddings of the words in the candidate and reference sentence via cosine similarity. We use BERTscore with the `bert-base-multilingual-cased` model as is the default for multilingual pairs. The last column of table 2 shows that BERTscore achieves very strong correlations with human judgments, which can be interpreted as validating those judgments. Recall that our ultimate goal is to arrive at a cross-lingual AMR metric to compare AMR parsers, not to compare the raw sentences, so we use BERTscore here to validate the human judgments. Reassuringly, the AMR metrics are not too far behind BERTscore.[4] Rows 4 and 8 compare the AMR metrics with BERTscore, showing that they are about as well correlated with each other as the metrics are with human judgments.

We also verify that sentence length is not a confounding variable in these judgments, with the correlation between average sentence length and human similarity score being only 0.07.

5 Evaluating Cross-lingual AMR Parsers

Now that we have assessed the metrics discussed in §4 on gold AMRs in comparison to human judgments, we are interested in seeing how existing cross-lingual AMR parsers perform on our recommended cross-lingual metric versus on monolingual Smatch.

5.1 Approach to Parser Evaluation

We compare the performances of four state-of-the-art cross-lingual AMR parsers: SGL (Procopio et al., 2021), Bilingual Information for Cross-lingual AMR Parsing ("BI") (Cai et al., 2021), XLPT-AMR (Xu et al., 2021), and Translate then Parse ("TP") (Uhrig et al., 2021).

The SGL semantic parser (Procopio et al., 2021) is a seq2seq architecture trained for neural machine translation. SGL as a cross-lingual AMR parser works well in a zero-shot setting (without seeing any non-English AMR examples in training). Using their mBART + AP (where AP stands for annotation projection) model, SGL reports Smatch scores of 73.3, 73.9, 73.4, and 64.9 for German, Spanish, Italian, and Mandarin Chinese respectively on machine translations of the test set.

Cai et al.'s (2021) AMR parser (which here we call "BI" because of its use of bilingual information) introduces translated and non-English texts into the training of a seq2seq parser, to better predict non-English concepts. BI reports Smatch scores of 64.0, 65.4, 67.3, and 56.5 for German, Spanish, Italian, and Chinese respectively.

XLPT-AMR (Xu et al., 2021) approaches zero-shot AMR parsing via multi-task learning. XLPT-AMR reports Smatch scores of 70.5, 71.8, and 70.8 for German, Spanish, and Italian respectively; XLPT-AMR was not evaluated on Chinese data.

Translate then Parse (Uhrig et al., 2021) takes a simple approach to cross-lingual AMR parsing: translating the non-English sentence to English and then parsing with an English AMR parser (`amrlib`).[5] Translate then Parse ("TP") claims Ger-

[4] This is unsurprisingly especially true for XS2match, which uses LaBSE embeddings (BERT-based cross-lingual sentence embeddings).

[5] https://github.com/bjascob/amrlib

	BI	XLPT-AMR	SGL	TP
Consensus	0.737	0.733	0.655	0.756
DFA	0.703	0.685	0.652	0.722
Bolt	0.671	0.676	0.608	0.708
Proxy	0.776	0.785	0.737	0.808
Xinhua	0.651	0.682	0.685	0.724
Average	**0.708**	**0.712**	**0.669**	**0.744**

Table 3: XS2match scores in comparison to gold Spanish AMRs for each parser on every subset of data in the evaluation. The column labels are automatic AMR parsers and the row labels are the five genres of data in the Spanish AMR corpus.

man, Spanish, Italian, and Chinese Smatch scores of 67.6, 72.3, 70.7, and 59.1, respectively.

While these cross-lingual parsers have previously only been evaluated in comparison to gold English AMRs via Smatch, we now perform three comparisons on a substantial subset of the AMR 2.0 (Knight et al., 2017) (and AMR 2.0 - Four Translations (Damonte and Cohen, 2020)) dataset: (1) Smatch evaluation in comparison to gold English AMRs, as was used for evaluation of these parsers in previous work, (2) Smatch evaluation in comparison to gold Spanish AMRs, and (3) XS2match evaluation in comparison to gold Spanish AMRs.

In this section, we consider the subset (486 sentences) of AMR 2.0 - Four Translations that is annotated in the Spanish AMR corpus (Wein et al., 2022). We then retrieve the parser data, either by contacting the authors of the work or by running the parser ourselves, for all of those Spanish sentences. Though it is the traditional form of cross-lingual AMR parser evaluation, we compare the system output to the English gold AMRs via Smatch, so that we have a direct comparison on this subset of data with our two completely novel sets of evaluation for these parsers: in comparison to gold Spanish AMRs, via Smatch as well as XS2match.[6]

5.2 Analysis of Results

English AMRs have been viewed as a proxy for evaluating parser output for Spanish sentences, but Spanish AMRs should be the true gold standard as they are not corrupted by translation divergences. In this subsection, we empirically assess how much this difference makes for comparing and ranking parser performance. To do this, we compare cross-lingual parser performance via the traditional method of comparing output to gold English AMRs, as well as using the existing monolingual method of Smatch and our proposed multilingual method of XS2match (S2match with multilingual embeddings) in comparison to the gold Spanish AMRs.

[6]We perform this comparison exclusively with Spanish gold AMRs because we want to focus on the sentences that have been used by previous work in the evaluation of cross-lingual AMR parsing, namely, the AMR 2.0 - Four Translations dataset. Only the Spanish sentences in this dataset have gold AMRs.

Metric	BI	XLPT-AMR	SGL	TP
Eng. Smatch	0.682	0.680	0.582	0.696
Span. Smatch	0.378	0.378	0.382	0.408
Span. XS2match	0.708	0.712	0.669	0.744

Table 4: Average evaluation scores for each of the three metrics considered for the cross-lingual parsers.

Since we are comparing the system parse of a Spanish sentence to a parallel gold AMR, the higher the score (regardless of metric), the more similar the output is to the gold Spanish AMR, and thus the better the system output. We calculate the average scores for each AMR parser by retrieving the score for each of the five texts included in the evaluation dataset and averaging them with the same weight. We opt for a macro-average by text because of comparable text sizes.

Table 4 shows the average score by metric for each of the four parsers. Monolingual Smatch puts the comparison to gold Spanish AMRs at a disadvantage because the system output is parsed into an English AMR. Therefore the lexical similarity is not considered between the two AMRs, and monolingual Smatch is not an effective tool for comparing cross-lingual AMR parser output to gold AMRs of the same language as the source sentence.

With the intent to find a method to compare to gold AMRs in the source sentence, we have already found that XS2match is a good choice for this type of evaluation in §4.2. We find that the Spanish XS2match scores are slightly higher for all parsers than the English Smatch scores, which indicates that it is actually not only a more *justified* comparison than a comparison against English AMRs as it accounts for source language, but also a *fairer and more accurate* comparison because the parallel AMRs are indeed being judged as more parallel.

Ultimately we find that Smatch, when comparing to English gold AMRs, provides a similar ranking and scores for the cross-lingual AMR parsers as XS2match does when comparing to Spanish AMR. Note that in table 4, the system-level comparison by average English Smatch and average Spanish XS2match is very comparable. When considering

a ranking of parser performance, our empirical results suggest that monolingual Smatch serves as a reasonable proxy of parser ranking in the absence of in-language gold AMRs. However, it is still important to note that absolute scores from monolingual Smatch are artificially depressed in the cross-lingual scenario, meaning that faithfulness to the original Spanish sentence is being rewarded. Therefore, this monolingual evaluation does not provide a sufficient substitute for the in-language comparison via XS2match due to the fact that monolingual Smatch against English AMRs does not account for the dramatic effect and importance of language on AMR structure (Wein and Schneider, 2021).

Figures 3 and 4 show scatterplots of the average performance of the four parsers. Notably, the SGL parser, which is zero-shot, performs the worst of the four parsers when using XS2match in comparison to Spanish gold AMRs (and in comparison to English gold AMRs via Smatch). XLPT-AMR is also zero-shot but performs slightly better than SGL on the English Smatch and Spanish XS2match comparisons. BI, which incorporates additional bilingual informations, performs similarly to XLPT-AMR, achieving only slightly lower scores.

Translate then Parse ("TP") performs best on all three evaluations. The XS2match scores for each parser on every text can be seen in table 3. While the four metrics achieve similarly high scores, the output from the Translate then Parse system consistently produces the highest similarity score via XS2match across all five texts. This suggests that using the highly accurate machine translation via EasyNMT as a pre-processing step, before involving any AMR parsing, is an effective way of capturing the linguistic information of the source sentence. This is perhaps surprising because the sentence is immediately translated, but less surprising due to the challenging nature of cross-lingual AMR parsing, given that none of the cross-lingual parsers are trained on gold non-English AMRs. The ability of the machine translation system to account for cross-linguistic divergence, as noted by Uhrig et al. (2021), enables an effective monolingual English AMR parser to work well in this setting.

6 Background on Other Evaluation Metrics

Other metrics which we did not adapt in this paper have been proposed for AMR evaluation.

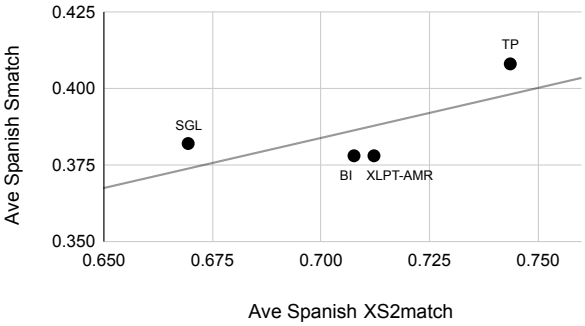

Figure 3: Average Spanish Smatch vs Average Spanish XS2match

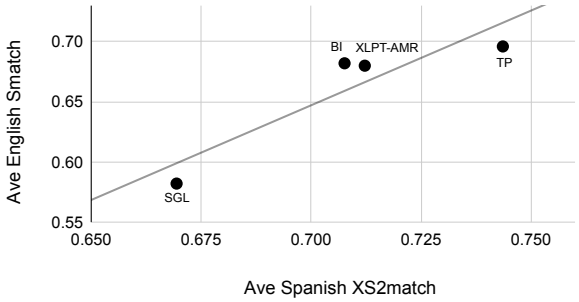

Figure 4: Average English Smatch vs Average Spanish XS2match

A document-level version of Smatch (Naseem et al., 2022) (as opposed to sentence-level) aligns the roots of its sentence-level AMR subgraphs. Similarly, Cai and Lam (2019) produces a version of Smatch designed to specifically consider core semantics, called Smatch-weighted.

SEMA (Anchiêta et al., 2019) extends Smatch by taking a breadth-first search approach to computing the maximum score; Smatch relies on one-to-one variable matching. The evaluation is limited and the metric is only shown to be *stricter* than Smatch.

Another existing monolingual AMR metric we did not consider in this work is MF_β. MF_β (Opitz and Frank, 2021) measures how easily an AMR can be reconstructed by AMR parsers and measures the grammaticality of the produced text. MF_β evaluation is more suited to AMR-to-text generation evaluation than to text-to-AMR parsing.

Goodman (2019) presents four AMR normalization techniques to ensure that isomorphic AMRs are evaluated as equivalent.

The BAMBOO suite (Opitz et al., 2021) houses various AMR similarity metrics to be able to assess the strengths and weaknesses of each metric.

7 Conclusion and Future Work

Our analysis of evaluation of cross-lingual AMR parsers indicates the usefulness of XS2match as a multilingual evaluation method. We recommend this approach as a way to compare AMRs parsed from non-English sentences to their gold non-English equivalents, while exploring additional alternatives in our work. We also find that using Smatch in comparison to gold English AMRs may be a useful tool for ranking cross-lingual AMR parser performance in the absence of in-language gold AMRs. With the future production of non-English gold AMRs, the evaluation of cross-lingual AMR parsers using our proposed metric will be more robust, accounting for the effect of source language on AMR.

8 Acknowledgements

Thank you to Yang Janet Liu, Yilun Zhu, Siyao Logan Peng, and Roy Ilany for providing human judgments. Thanks to Rexhina Blloshmi, Yitao Cai, Luigi Procopio, and Dongqin Xu for providing system results. We thank anonymous reviewers for their feedback. This work is supported by a Clare Boothe Luce Scholarship.

References

Eneko Agirre, Carmen Banea, Daniel Cer, Mona Diab, Aitor Gonzalez-Agirre, Rada Mihalcea, German Rigau, and Janyce Wiebe. 2016. SemEval-2016 task 1: Semantic textual similarity, monolingual and cross-lingual evaluation. In *Proceedings of the 10th International Workshop on Semantic Evaluation (SemEval-2016)*, pages 497–511, San Diego, California. Association for Computational Linguistics.

Rafael T Anchiêta, Marco AS Cabezudo, and Thiago AS Pardo. 2019. SEMA: an extended semantic evaluation metric for AMR. *arXiv preprint arXiv:1905.12069*.

Rafael Torres Anchiêta and Thiago Alexandre Salgueiro Pardo. 2018. A rule-based AMR parser for Portuguese. In *Ibero-American Conference on Artificial Intelligence*, pages 341–353. Springer.

Laura Banarescu, Claire Bonial, Shu Cai, Madalina Georgescu, Kira Griffitt, Ulf Hermjakob, Kevin Knight, Philipp Koehn, Martha Palmer, and Nathan Schneider. 2013. Abstract Meaning Representation for sembanking. In *Proceedings of the 7th Linguistic Annotation Workshop and Interoperability with Discourse*, pages 178–186, Sofia, Bulgaria. Association for Computational Linguistics.

Rexhina Blloshmi, Rocco Tripodi, and Roberto Navigli. 2020. XL-AMR: Enabling cross-lingual AMR parsing with transfer learning techniques. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 2487–2500, Online. Association for Computational Linguistics.

Deng Cai and Wai Lam. 2019. Core semantic first: A top-down approach for AMR parsing. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 3799–3809, Hong Kong, China. Association for Computational Linguistics.

Shu Cai and Kevin Knight. 2013. Smatch: an evaluation metric for semantic feature structures. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 748–752, Sofia, Bulgaria. Association for Computational Linguistics.

Yitao Cai, Zhe Lin, and Xiaojun Wan. 2021. Making better use of bilingual information for cross-lingual AMR parsing. In *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*, pages 1537–1547, Online. Association for Computational Linguistics.

Marco Damonte. 2019. *Understanding and Generating Language with Abstract Meaning Representation*. Ph.D. thesis, University of Edinburgh.

Marco Damonte and Shay Cohen. 2020. Abstract Meaning Representation 2.0 - Four Translations. Technical Report LDC2020T07, Linguistic Data Consortium, Philadelphia, PA.

Marco Damonte and Shay B. Cohen. 2018. Cross-lingual Abstract Meaning Representation parsing. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*,

pages 1146–1155, New Orleans, Louisiana. Association for Computational Linguistics.

Marco Damonte, Shay B. Cohen, and Giorgio Satta. 2017. An incremental parser for Abstract Meaning Representation. In *Proceedings of EACL*.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. BERT: pre-training of deep bidirectional transformers for language understanding. *CoRR*, abs/1810.04805.

Chris Dyer, Victor Chahuneau, and Noah A. Smith. 2013. A simple, fast, and effective reparameterization of IBM Model 2. In *Proceedings of the 2013 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 644–648, Atlanta, Georgia. Association for Computational Linguistics.

Fangxiaoyu Feng, Yinfei Yang, Daniel Cer, Naveen Arivazhagan, and Wei Wang. 2022. Language-agnostic BERT Sentence Embedding. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 878–891, Dublin, Ireland. Association for Computational Linguistics.

Michael Wayne Goodman. 2019. AMR normalization for fairer evaluation. *arXiv preprint arXiv:1909.01568*.

Kevin Knight, Bianca Badarau, Laura Banarescu, Claire Bonial, Madalina Bardocz, Kira Griffitt, Ulf Hermjakob, Daniel Marcu, Martha Palmer, Tim O'Gorman, and Nathan Schneider. 2017. Abstract Meaning Representation (AMR) Annotation Release 2.0. Technical Report LDC2017T10, Linguistic Data Consortium, Philadelphia, PA.

Bin Li, Yuan Wen, Weiguang Qu, Lijun Bu, and Nianwen Xue. 2016. Annotating *the Little Prince* with Chinese AMRs. In *Proceedings of the 10th Linguistic Annotation Workshop held in conjunction with ACL 2016 (LAW-X 2016)*, pages 7–15, Berlin, Germany. Association for Computational Linguistics.

Tahira Naseem, Austin Blodgett, Sadhana Kumaravel, Tim O'Gorman, Young-Suk Lee, Jeffrey Flanigan, Radu Fernandez Astudillo, Ramón Florian, Salim Roukos, and Nathan Schneider. 2022. DocAMR: Multi-sentence AMR representation and evaluation. In *Proc. of NAACL-HLT*, Seattle, Washington.

Juri Opitz, Angel Daza, and Anette Frank. 2021. Weisfeiler-Leman in the BAMBOO: Novel AMR graph metrics and a benchmark for AMR graph similarity. *Transactions of the Association for Computational Linguistics*, 9:1425–1441.

Juri Opitz and Anette Frank. 2021. Towards a decomposable metric for explainable evaluation of text generation from AMR. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*, pages 1504–1518, Online. Association for Computational Linguistics.

Juri Opitz, Letitia Parcalabescu, and Anette Frank. 2020. AMR similarity metrics from principles. *Transactions of the Association for Computational Linguistics*, 8:522–538.

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. BLEU: a method for automatic evaluation of machine translation. In *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics*, pages 311–318, Philadelphia, Pennsylvania, USA. Association for Computational Linguistics.

Luigi Procopio, Rocco Tripodi, and Roberto Navigli. 2021. SGL: Speaking the graph languages of semantic parsing via multilingual translation. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 325–337, Online. Association for Computational Linguistics.

Adylan Roaffa Ilmy and Masayu Leylia Khodra. 2021. Parsing Indonesian sentence into Abstract Meaning Representation using machine learning approach. *arXiv e-prints*, pages arXiv–2103.

Linfeng Song and Daniel Gildea. 2019. SemBleu: A robust metric for AMR parsing evaluation. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 4547–4552, Florence, Italy. Association for Computational Linguistics.

Sarah Uhrig, Yoalli Garcia, Juri Opitz, and Anette Frank. 2021. Translate, then parse! A strong

baseline for cross-lingual AMR parsing. In *Proceedings of the 17th International Conference on Parsing Technologies and the IWPT 2021 Shared Task on Parsing into Enhanced Universal Dependencies (IWPT 2021)*, pages 58–64, Online. Association for Computational Linguistics.

Chuan Wang, Bin Li, and Nianwen Xue. 2018. Transition-based Chinese AMR parsing. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 2 (Short Papers)*, pages 247–252, New Orleans, Louisiana. Association for Computational Linguistics.

Shira Wein, Lucia Donatelli, Ethan Ricker, Calvin Engstrom, Alex Nelson, and Nathan Schneider. 2022. Spanish Abstract Meaning Representation: Annotation of a general corpus. *arXiv preprint arXiv:2204.07663*.

Shira Wein and Nathan Schneider. 2021. Classifying divergences in cross-lingual AMR pairs. In *Proceedings of The Joint 15th Linguistic Annotation Workshop (LAW) and 3rd Designing Meaning Representations (DMR) Workshop*, pages 56–65, Punta Cana, Dominican Republic. Association for Computational Linguistics.

Shira Wein and Nathan Schneider. 2022. Effect of source language on AMR structure. In *Proceedings of the 16th Linguistic Annotation Workshop (LAW)*, Marseille, France.

Dongqin Xu, Junhui Li, Muhua Zhu, Min Zhang, and Guodong Zhou. 2021. XLPT-AMR: Cross-lingual pre-training via multi-task learning for zero-shot AMR parsing and text generation. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 896–907.

Nianwen Xue, Ondřej Bojar, Jan Hajič, Martha Palmer, Zdeňka Urešová, and Xiuhong Zhang. 2014. Not an interlingua, but close: Comparison of English AMRs to Chinese and Czech. In *Proceedings of the Ninth International Conference on Language Resources and Evaluation (LREC'14)*, pages 1765–1772, Reykjavik, Iceland. European Language Resources Association (ELRA).

Tianyi Zhang, Varsha Kishore, Felix Wu, Kilian Q. Weinberger, and Yoav Artzi. 2020. BERTScore: Evaluating text generation with BERT. In *Proc. of ICLR*, Online.

QSTS: A Question-Sensitive Text Similarity Measure for Question Generation

Sujatha Das Gollapalli and **See-Kiong Ng**
Institute of Data Science
National University of Singapore
{idssdg,seekiong}@nus.edu.sg

Abstract

While question generation (QG) has received significant focus in conversation modeling and text generation research, the problems of comparing questions and evaluation of QG models have remained inadequately addressed. Indeed, QG models continue to be evaluated using traditional measures such as BLEU, METEOR, and ROUGE scores which were designed for other text generation problems. We propose *QSTS*, a novel **Q**uestion-**S**ensitive **T**ext **S**imilarity measure for questions that characterizes their target intent based on question class, named-entity, and semantic similarity information.

We show that *QSTS* addresses several shortcomings of existing measures that depend on n-gram overlap scores and obtains superior results compared to traditional measures on publicly-available QG datasets. We also collect a novel dataset *SimQG* for enabling question similarity research in QG contexts. *SimQG* contains questions generated by state-of-the-art QG models along with human judgements on their relevance with respect to passage contexts as well as the given reference questions. Using *SimQG*, we showcase the key aspect of *QSTS* that differentiates it from all existing measures. *QSTS* is not only able to characterize similarity between two questions, but is also able to score questions with respect to passage contexts. Thus *QSTS* is, to our knowledge, the first metric that enables the measurement of QG performance in a reference-free manner.

1 Introduction

Automatic Question Generation (QG), the task of generating natural language questions for a given input text passage continues to garner significant research focus in the NLP community (Wang et al., 2020b; Huang et al., 2021) due its potential application in education (Srivastava and Goodman, 2021), tutoring (Lindberg et al., 2013) and interactive dialog systems (Wang et al., 2020a).

In current research, in lieu of human evaluation, the standard practice for evaluating the performance of QG models involves the use of Question Answering (QA) datasets containing pairs of (reference question, passage context) elements. For evaluating QG, the machine-generated question for a given passage context is compared with the given reference question by applying metrics such as BLEU (Papineni et al., 2002), METEOR Lavie and Agarwal (2007), and ROUGE (Lin, 2004).

The above widely-used measures were originally developed for evaluating tasks such as summarization and translation and are based on overlap of n-grams between a given reference text and the model-generated text. Though studies have indicated that these measures do not correlate well with human judgements of fluency, relevance, and coherence (Callison-Burch et al., 2006; Liu et al., 2016; Nema and Khapra, 2018) these measures are easy to compute and continue to be used for various natural language generation (NLG) tasks. Recently though, research studies are addressing metrics learning for NLG using transformers and these learnt metrics were shown to obtain state-of-the-art performance in evaluation (Zhang et al., 2020; Sellam et al., 2020).

We posit that the existing metrics for measuring text generation tasks are inadequate for comparing questions due to their inability to incorporate various features that characterize questions. The first among these features is the question class (alternatively referred to as answer type) which places constraints on the answer to a given question.[1] For example, for the question, "Who was Lincoln?", in context of a passage on the former US president, a correct answer is most likely looking for a description referring to his job/role/occupation whereas the answer to the question "What is humidity?" is a definition. We argue that question class as well as named entities, when present in a question, directly affect the intent of the question and need to be

[1] https://cogcomp.seas.upenn.edu/Data/QA/QC/definition.html

Proceedings of the 29th International Conference on Computational Linguistics, pages 3835–3846
October 12–17, 2022.

Question Pairs	QBLEU	BLEURT	*QSTS*
(What was the title of Bob Dylan's first album?; What was Bob Dylan's first album called?)	0.744	0.816	0.928
(What was the name of Vincent's brother; Who was Vincent's brother?)	0.473	0.667	0.874
(Where in Germany was the composer Beethoven born?; Which city in Germany is the place of birth of Beethoven?)	0.278	0.709	0.548
B1:0.457, B4:0.000, Meteor: 0.344, Rouge: 0.485			
(Who was Columbus?; Where is Columbus?)	0.508	0.671	0.0
(What was the name of Vincent's brother?; What was the name of Vincent's painting?)	0.832	0.519	0.0
(When did Freddie Mercury die?; How did Freddie Mercury die?)	0.779	0.799	0.0
B1: 0.733, B4: 0.683, Meteor: 0.456, Rouge: 0.663			

Table 1: Illustrative question pairs are shown with system-level scores for BLEU-1 (B1) and BLEU-4 (B4), METEOR, and ROUGE and as pair-level scores for QBLEU (Nema and Khapra, 2018), BLEURT (Sellam et al., 2020), and *QSTS*

handled differently from other words in a question.

The metrics currently in use are based on word overlap and do not capture the semantics of questions as can be seen in the representative examples of similar and dissimilar questions in Table 1. The system-level scores of the traditional metrics (BLEU, METEOR, ROUGE), are shown in this table along with QBLEU values (the extension of BLEU scores for questions proposed by Nema, et al (2018)), as well as BLEURT scores that measure semantic similarity between two texts using BERT (Sellam et al., 2020; Devlin et al., 2019).

In Table 1, we note that none of the existing metrics are able to accurately assess the similarity or difference between the given question pairs and instead tend to assign high scores to dissimilar questions and low scores to simple rewritings of the questions with the same intent. In the rightmost column of Table 1, we show the values of our proposed **Q**uestion-**S**ensitive **T**ext **S**imilarity (*QSTS*) scores assigned to these question pairs that are more representative. We discuss the design of *QSTS* in the rest of this paper. Our contributions are as follows:

1. We propose *QSTS*, a **Q**uestion-**S**ensitive **T**ext **S**imilarity measure for comparing questions. Unlike existing measures, *QSTS* explicitly represents the question class and named entities present in a given question pair and combines them with dependency tree information and word embeddings to provide a more representative and interpretable measure of the semantic similarity between the two questions.

2. We evaluate *QSTS* on publicly-available datasets of similar questions available for QG/QA research. Our experiments indicate that our proposed measure provides a more accurate representation of question similarity compared to traditional measures employed for characterizing QG model performance.

3. We present the potential use of *QSTS* in reference-free evaluation for QG. The *QSTS* metric is able to reasonably characterize question quality of model-generated questions using passages that were used for generating them. This capability is representative of the human ability to judge whether a given question is fluent and relevant in the context of a given passage unlike existing measures that need reference questions for evaluation.

   We demonstrate reference-free evaluation for QG using *QSTS* on a novel dataset, *SimQG*. *SimQG* contains human judgements for machine-generated questions from latest QG models for a selection of about 500 (reference question, passage) pairs from SQuAD (Rajpurkar et al., 2016). *SimQG* and an implementation of *QSTS* in Python have been made available for academic research.[2]

Organization: We present the details of comput-

[2] https://github.com/NUS-IDS/coling22_QSTS

ing *QSTS* in Section 2. Our novel dataset *SimQG* is described in Section 3. Experiments and results are described in Section 4 while closely-related work is summarized in Section 5. Finally, we conclude the paper with a summary and remarks on future directions in Section 6.

2 Question-Sensitive Text Similarity

A necessary aspect to capture while comparing two questions is a measure of whether the target intent behind the two questions is the same. As highlighted in the examples from Table 1, measuring simple lexical overlap between n-grams of two questions is insufficient for this purpose. In comparison, word and sentence representations (Pennington et al., 2014; Peters et al., 2018) are known to capture similarity between words despite the lexical mismatch. Extending this idea further, metrics based on contextual representations were developed for measuring similarity for text generation tasks such as translation and image captioning (Zhang et al., 2020; Sellam et al., 2020).

However, note that simple changes to words has significant changes in question meanings ("Who was Columbus" vs. "Where was Columbus?") and embedding spaces learnt purely from word co-occurrence and contextual information from large corpora suffer from the drawback of overestimating scores to word pairs representing entities as well as question cues.[3] Consequently, these measures tend to overestimate similarity in case of non-similar questions as shown in the last three examples in Table 1. We address the above issues by modeling three different question-specific aspects in *QSTS*:

Question Class (QC) or Answer-Type for a question refers to the constraints the question imposes on the "sought after answer" (Li and Roth, 2002). Li and Roth (2002) designed a two-level question class taxonomy (Footnote 1) for representing questions in TREC question answering tasks[4] where the answers to questions can be assigned one of six coarse classes namely, Abbreviation, Entity, Description, Human, Location, and Numeric value. These six classes are further organized into 50 fine classes for a more specific classification of the answer type. For example, the coarse class "Human" includes fine classes for an individual, a group of individuals, a description of an individual, as well as the title assigned to an individual. Question class information has also been used to improve question answering and question generation performance (Tayyar Madabushi et al., 2018; Zhou et al., 2019).

We use question class information in *QSTS* to characterize if the two questions under consideration are seeking the same answer type. With an accurate question-class classifier, both questions in the first row of Table 1 are assigned the same question class in the QC taxonomy (referring to "creative pieces and inventions") whereas the two questions in the fourth row are assigned classes corresponding to "description of an individual" versus "location" automatically capturing their different semantics. We can directly measure the question class similarity (**`qcsim`**) using the δ function, where $\delta_{ij} = 1$, if $qc(q_i) = qc(q_j)$ and 0 otherwise where qc_i stands for the question class for question q_i. To incorporate the hierarchical nature of the QC taxonomy, we modify this function to assign partial score of 0.5 if the coarse class matches for the two questions and 0.75 if one of question fine classes involves the catch-all "other" class. For example, the question classes assigned to the two questions in the third row correspond to "Location:Other" and "Location:city", respectively.

Named-Entities when present in a question constrain the question with reference to the mentioned entity. For instance, if "Columbus" in the question "When was Columbus born?" is replaced by another name, it will become a completely different question. Therefore, similar to question classes, named entities require a *hard* measurement. To account for multi-word names and partial matches, we isolate the tokens in a given reference question referring to named entities and look for their presence in the generated question.[5]

The named-entity similarity (**`nesim`**) is measured as the fraction of named-entity tokens in the reference question that are present in the generated question. That is, for a given reference question, "Who was Abraham Lincoln?" and the question "Who was Lincoln?", the named-entity similarity score is computed as $\frac{1}{2}$.

Semantic Similarity forms the third component of *QSTS*. We use the dependency parse of ques-

[3] For instance, based on GLoVe embeddings (Pennington et al., 2014), ("Lincoln", "Columbus"), and ("who", "when") have cosine similarity values of 0.659 and 0.608, respectively.

[4] https://trec.nist.gov/data/qa.html

[5] We use proper nouns in parts-of-speech tags and named-entity tags to identify name tokens.

tions to compute the semantic similarity between them. Dependency trees of sentences capture syntactic dependencies among the words in a sentence such as subject-object, modifier, and clausal links. Dependency-tree based kernels are widely-used in measuring sentence similarity (Croce et al., 2011; Özateş et al., 2016).

Let e=(h, rel, t) represent a directed edge in the dependency tree of a given question where the typed relation rel exists between two tokens, h and t. Given the dependency edges of two questions (reference and generated), we match the dependency edges of the reference question ($E(q_r)$) with those of the generated question ($E(q_g)$) and pick the best matching or the most similar edge $\forall e \in E(q_r)$. The edge similarity $esim(e_{m_r}, e_{n_g})$ is computed as

$$\delta(rel_{m_r}, rel_{n_g}) \Big[\frac{sim(h_{m_r}, h_{n_g}) + sim(t_{m_r}, t_{n_g})}{2} \Big]$$

In the above formulation, $\delta(rel_{m_r}, rel_{n_g})$ refers to the Kronecker δ function that assigns a value of 1 if the two relation types are the same and zero otherwise and $sim(a, b)$ is computed using cosine similarity of the word embeddings if a, b are non-name tokens. However, if either the head or tail of the edge is a name token, we use exact match on that side of the edge. That is, for the two edges e_{m_r}=$(h_{m_r}, rel_{m_r}, t_{m_r})$ and e_{nj}=$(h_{n_g}, rel_{n_g}, t_{n_g})$, if h_{m_r} is a name token, $esim(e_{m_r}, e_{n_g}) =$

$$\delta(rel_{m_r}, rel_{n_g}) \delta(h_{m_r}, h_{n_g}) sim(t_{m_r}, t_{n_g})$$

The same principle applies if t_{m_g} is a name token. The above formulation ensures that name tokens are not treated like regular tokens and a *hard* match is enforced while at the same time the edge similarity values stay in the range $[0, 1]$.

The final semantic similarity (**`semsim`**) of the two questions is the average similarity of the edges in the reference question that match best with the edges in the generated question. Since named entity tokens and question cue words are handled separately, only edges involving content words are considered in this computation. Additionally, we ignore edges representing less informative dependency relations such as "punctuation", "possessive modifier" and seven others in line with previous works (Özateş et al., 2016).

QSTS: Note that each of the similarity functions, `qcsim`,`nesim`,`semsim` assign normalized scores between $[0, 1]$ for an independent aspect of matching the two questions. These three scores can be summarized using the geometric mean (Fleming and Wallace, 1986) to obtain a single score between $[0, 1]$ for **Q**uestion-**S**ensitive **T**ext **S**imilarity as

$$QSTS(r, g) = (\text{qcsim}_{rg} * \text{nesim}_{rg} * \text{semsim}_{rg})^{1/3} \quad (1)$$

The *QSTS* score is directional, the `nesim` and `semsim` computations are with respect to a given reference question. That is, for `nesim`, we compute how many of the name tokens in a reference question are seen in the given/generated question and in `semsim`, we compute the best matching edges from $E(q_g), \forall e \in E(q_r)$. Note that this directional nature enables the computation of these two scores for (question, passage) pairs as well. To estimate if a question is valid for a passage, we can check if the named entities (when present) in the question can be found in the passage and if the dependency edges of the question are also supported in the passage. In this manner, *QSTS* provides for a **reference-free** evaluation of a question, given a passage context.

3 The *SimQG* dataset

Current models for QG are evaluated using QA datasets containing (passage, reference-question) pairs. Model-generated questions are compared against these reference questions using traditional metrics. It is our contention that given a passage context and questions generated by QG models against that context, several valid questions may be possible and it may not be representative to only compare generated questions against a specific given reference. To demonstrate this claim, we collected a novel dataset containing human judgements of machine-generated questions against their associated passage contexts and the reference questions available for these contexts.

Our novel dataset is based on SQuAD (Rajpurkar et al., 2016), a widely-used dataset in both QA and QG studies. About 500 (question, passage) pairs were randomly sampled from the test portion of the SQuAD dataset (used in (Zhou et al., 2018)). Recent QG models from ProphetNet (Qi et al., 2020), T5 (Raffel et al., 2020), and one of the early neural models based on Gated Self-Attention (GSA) networks (Zhao et al., 2018) were used for obtaining machine-generated questions. By choosing machine-generated questions from models with QG performance ranging from high (ProphetNet)

to significantly low (GSA), we seek to include questions in our dataset with varying degrees of answerability, fluency and relevance (Pan et al., 2020; Wang et al., 2020a).

Our annotation task on the crowdsourcing platform Amazon Mechanical Turk (AMT) was set up along the lines of previous QG works (Pan et al., 2020; Wang et al., 2020a). Each passage along with the machine generated question was examined by three independent crowdworkers to characterize if the question is (1) **Fluent**: Is the question grammatically correct, natural sounding, and semantically valid for the given passage context (yes=1.0, acceptable=0.5, no=0.0)?; (2) **Answerable**: Is the answer to the generated question present in the passage (yes=1.0, no=1.0)?; (3) **Relevant**: is the question relevant to the passage and only based on the content in the passage (yes=1.0, no=0.0)?

The workers were also asked to compare the machine-generated question with the reference question provided in SQuAD and to identify whether the question is similar to the reference question (score=1.0), similar but has less/more information compared to the reference question (score=0.5), different but has the same answer as the reference question (score=0), or related but different (score=0) and finally very different from the reference question (score=0).

By averaging worker scores for each question, we obtain relevance/fluency/answerability scores for each (machine-generated question, passage) pair as well as a similarity score for pairs of (machine-generated, reference) question pairs all in the range $[0, 1]$ and by suitably thresholding at 0.5, we can obtain pairs of similar and dissimilar questions for our study as well as questions that are not fluent, answerable, or relevant. We refer to the dataset collected above as the *SimQG* dataset.

A summary of *SimQG* is provided in Table 2. As seen in this table, all three QG models generate reasonably fluent questions. In accordance with the published QG performance numbers of these models on SQuAD dataset, the number of non-fluent and non-answerable questions is the highest for GSA, lowest for ProphetNet (PrptNet) and in-between for the T5-based model (Qi et al., 2020; Zhao et al., 2018).[6] In all three models, the number of non-relevant machine-generated questions is very low (2-6%) whereas the machine-generated question was considered not similar to the reference question in 40-50% of the cases. We posit that this high disparity is indicative of why QG models need evaluation measures that are not based only on reference questions.

QGModel	!Flu	!Rel	!Ans	!Sim
PrptNet (300)	0.66%	2%	4.66%	45%
GSA (100)	7%	4%	19%	47%
T5 (100)	4%	6%	12%	38%
All(500)	2.6%	3.2%	9%	44%

Table 2: Summary of *SimQG* dataset. #Qs is the number of questions whereas !Flu, !Rel, !Ans, !Sim columns refer to the percentages of questions that are not fluent, not relevant, not answerable, and not similar to the given reference question.

Additional notes on data collection: On the AMT platform, we required the crowdworkers to have greater than 95% HIT approval rate, a minimum of 10,000 HITs, and be located in the United States and clear a qualification test to be able to work on our task. Each worker was paid $\$0.30$ per HIT. We met the *ethics, quality, and reliability* considerations for our collected dataset as follows: As part of the AMT data collection process, the *anonymity* and *privacy* of the crowdworkers is already ensured. Furthermore, the settings for the HIT approval rates, and location of the worker, described previously are set similar to previous QA/QG data collection efforts to ensure the English language skills of the data annotators and thus the *quality* of the collected dataset. A total of 7 workers helped in creating our dataset. About 47% of the workers who attempted the qualification test were able to obtain a score of 80% or more and gained the eligibility to work on our task. Their annotations can, therefore, be considered reasonably reliable on average.

4 Experiments

Baseline Measures: We demonstrate the performance of our proposed *QSTS* measure by comparing with several existing measures. The first set of measures are traditionally employed in various text generation tasks including QG and comprise of BLEU, METEOR, and ROUGE scores (Papineni et al., 2002; Lavie and Agarwal, 2007; Lin, 2004). All these measures are based on n-gram overlap between the generated text and the reference text (of the same "type", for example, two summaries, or two sentences).

[6] https://github.com/patil-suraj/question_generation

The QBLEU metric was designed specifically for QG systems and includes the notion of answerability, that is, does the question include enough information to enable answer retrieval for the given question (Nema and Khapra, 2018)? To this end, various weights are estimated and incorporated for question cue words, content words, named entities and combined linearly to assign answerability score for a question. Answerability is further combined with the traditional BLEU score to obtain a QBLEU score.[7]

A recent research direction involves the use of transformers for learning metrics for text generation tasks (Zhang et al., 2020; Sellam et al., 2020). Based on its state-of-the-art performance on various NLG tasks compared to other variants such as BERTscore, we include BLEURT as one of our baselines for comparing questions. To the best of our knowledge, BLEURT has not been specifically evaluated for matching questions and we seek to bridge this gap as part of our experiments.[8]

Datasets: We used two existing QA/QG datasets with paraphrase information for evaluation. The first is the ComQA dataset that includes about 3.3K paraphrase pairs (Abujabal et al., 2019) while the second is the recently-compiled FIRS dataset, containing approximately 5K question pairs (Deschamps et al., 2021). The FIRS dataset includes rewrites of a given question which also include an extra fact from a knowledge base. That is, the rewritten question has the same intent as the original question but includes additional facts of relevant named entities. We randomly selected one of questions from each paraphrase clusters in ComQA as the reference question whereas in FIRS, the original question forms the reference question.

In addition, we evaluate on questions from the Quora Question Pairs (QQP) dataset.[9] QQP is a large dataset of about 400k question pairs obtained from Quora and includes duplicate and non-duplicate labels indicative of whether the intent of the two questions is the same. Note that this dataset is not used for QG since passage contexts and answers are unavailable. Moreover, the labels are known to be noisy in this dataset, and the questions on community forums are stylistically different from standard QG (António Rodrigues et al., 2017). Despite these differences, we study a randomly selected 5% sample of the QQP dataset separated into duplicate pairs (QQP-Dup) and non-duplicate pairs (QQP-ND). Finally, we provide evaluation on *SimQG*, the dataset specifically collected by us to model QG contexts (Section 3).

Question Class Identification: We trained our question class classifier on the widely-used TREC dataset (Li and Roth, 2002). A T5-large model[10] fine-tuned for this task obtains a test performance on par with state-of-the-art results with a classification accuracy of 92% on the fine-level classes (50 classes) and an accuracy of 97% on the six coarse classes (Reimers and Gurevych, 2019). When computing *QSTS* scores for question pairs where the question classes cannot be assumed to be the same (such as QQP-ND and *SimQG*), predictions with this model were used.

Other Settings: For computing *QSTS* scores, we need the dependency parse, parts-of-speech and named-entity tags for questions. We used the Stanza library for this purpose.[11] Since the name-tokens and question class information are treated separately, we avoid contextual and sentence-level embeddings that are time-consuming to estimate (Peters et al., 2018; Reimers and Gurevych, 2019) and instead directly use word embeddings from GloVe that only involves lookup (Pennington et al., 2014). All QG and QC experiments, and metrics that involve deep learning models were performed on a single GPU of an Nvidia Tesla cluster and take time between 1-12 hours based on the experiment setting and dataset size. The code for *QSTS* and the *SimQG* dataset have been released for academic research.

4.1 Results and Observations

Comparison of Measures: We compare *QSTS* against existing baseline measures on similar questions from ComQA and FIRS, as well as duplicate and non-duplicate question datasets QQP-Dup and QQP-ND, respectively. An ideal measure should assign high scores (close to 1) to similar questions and low scores (close to 0) to dissimilar ones. We show the system-level BLEU, METEOR, and ROUGE score as well as average and standard deviation of QBLEU, BLEURT, and *QSTS* scores

[7] https://github.com/PrekshaNema25/Answerability-Metric

[8] https://github.com/google-research/bleurt

[9] https://quoradata.quora.com/First-Quora-Dataset-Release-Question-Pairs

[10] https://huggingface.co/t5-large

[11] https://stanfordnlp.github.io/stanza/

Dataset	B1	B4	METEOR	ROUGE	QBLEU	BLEURT	*QSTS*
ComQA	0.602	0.287	0.373	0.566	0.594±0.151	**0.696**±0.110	0.692±0.298
FIRS	0.557	0.430	0.485	0.695	0.554±0.187	0.728±0.110	**0.866**±0.231
QQP-Dup	0.561	0.277	0.334	0.545	0.449±0.231	0.711±0.112	**0.754**±0.283
QQP-ND	0.342	**0.158**	0.204	0.344	0.289±0.252	0.491±0.169	0.388±0.391

Table 3: Question Similarity Metrics Evaluated on Existing Datasets

in Table 3.

Performance on Paraphrase datasets: We see in Table 3 that *QSTS* is significantly better than other measures on FIRS and QQP-Dup datasets, and is on par with the BLEURT measure on ComQA. We analyzed ComQA further to gain insight into where *QSTS* breaks down. From Equation 1, the *QSTS* score is zero when any of the component scores, `qcsim`, `nesim`, and `semsim`, is zero. That is, when the question classes, named-entities, or the content words of the two questions do not match. In ComQA, the *QSTS* score was zero for 16.9% question pairs with the `qcsim`, `nesim`, and `semsim` scores being independently zero in 5%, 8.6%, and 4.6% of the question pairs, respectively.

Since question paraphrases should, ideally, have the same question class but `qcsim` is zero for 5% of the cases, we can attribute these mismatches to the errors made by the question class predictor. However, we also note that this could be caused by erroneous pairs present in ComQA such as ("when did Judy Garland first marry?"; "who was Judy Garlands first married to?") where the question classes are indeed different and were predicted correctly as "NUM:date" and "HUM:ind" by our question class predictor.

Furthermore, ComQA also has instances where mentions of the same named-entity have typos and other differences. For example, pairs such as ("what is muhamad alis real name?"; "what is mahummad ali birth name?") and (" who was the german fascist leader during world war 2?"; "what man was the leader of germany during ww2?").

Finally, about 8% of the questions in ComQA appear to be in search-engine style ("the first american in outer space?") and do not have any of the question cue words.[12] Noisy inputs affect the type of dependency edges between content words and may result in zero `semsim` scores. Overall, *QSTS* is not fully-equipped to handle noisy paraphrases since errors in the component scores are penalized severely (Equation 1).

[12] why/who/where/how/which/when/where

Performance on Non-Paraphrases: On the QQP-ND dataset containing non-duplicate question pairs, simple n-gram based measures seem to do better than embedding-based BLEURT and *QSTS* measures which overestimate the similarity scores. As mentioned in Section 2, the *QSTS* scores are directional and also depend on the predictions from the question class classifier. Therefore, in the given non-duplicate pair from QQP-ND, ("How does one become an angel investor?"; "How do I get a job at Goldman Sachs?"), the *QSTS* scores change from 0 to 0.718 depending on which question forms the "reference". Moreover, QC predictions may not always be accurate considering the stylistic differences in QQP questions when compared to those from TREC. Questions in QQP include conjunctions such as "How do you bake pork chops in an oven and how long should you bake them?" and multi-sentence questions such as "I have completed my MBA with . . . in PSU. I want to work abroad, how do I start?".

We note that accurate measurement of dissimilar questions is not a big concern in QG contexts. For a given passage context, a good QG model is unlikely to generate a question comprising of completely arbitrary words in contrast with some pairs in QQP-ND such as ("How can I get perfect idea about best golf carts?"; "Is it hard to get a job in US after MIS without prior work experience?")

Reference-free Evaluation on *SimQG*: Using the human-assigned scores for similarity between machine-generated and reference question pairs in *SimQG*, we threshold at 0.5 to obtain pairs of questions considered similar and for these questions, we compute *QSTS* scores between the machine-generated questions and the corresponding passages. In other words, how many machine-generated questions can we correctly assign a score value ≥ 0.5 when the passage is used instead of comparing with the reference?

Similarly, for the set of machine-generated questions judged as non-fluent, non-relevant, and non-answerable by humans, how many questions are correctly assigned scores <0.5 based on the pas-

sage. The results of these computations are illustrated in Table 4.

Setting	#Qs	NoQC	withQC
Similar	280	66.1%	56.4%
Non-Fluent	13	30.7%	76.9%
Non-Relevant	16	56.3%	81.3%
Non-Answerable	45	40.0%	66.6%

Table 4: Percentages of similar, non-fluent, non-relevant and non-answerable questions correctly identified in *SimQG* by *QSTS* in the reference-free setting. #Qs refers to the number of questions whereas NoQC and withQC refer to with and without the question class information, respectively.

The percentage of questions correctly scored using reference-free *QSTS* is about 56% when question class information obtained from reference questions is incorporated ("withQC" column in Table 4). The percentage is, however, significantly higher (66%) when question class information is not considered ("NoQC"). This difference suggests that valid questions are being generated for a given passage context despite having different question classes. However, when question class information is incorporated into *QSTS* computation, we are able to determine non-fluent, non-relevant, and non-answerable questions with significantly higher accuracy as observed in Table 4.

In practice, it may not be unreasonable to assume that the expected question class is known *a priori*, considering the current state-of-the-art QG performance is obtained in the answer-aware (as opposed to answer agnostic) setting when the answer span is assumed to be known and used as a signal while learning QG (Pan et al., 2019). Even without explicit question class information, *QSTS* can correctly identify relevant and non-relevant questions with reasonable accuracies. Given that this is the first method to do so without a known reference question, this is an exciting result.

In contrast, the traditional measures as well as QBLEU and BLEURT expect similar types of texts for their computation. In our experiments, when QBLEU and BLEURT measures are computed using generated questions and passages as inputs, both measures were unable to correctly assign scores $>= 0.5$ to any of the similar questions. That is, the percentage correct values in the top row of Table 4 are zeros for both these measures. Anecdotal examples of question, passage pairs scored with our *QSTS* measure are provided in Table 5 for illustration. *QSTS* correctly assigns high scores (indicating relevant) to the top-two (question, passage) pairs and lower scores (less than 0.5 indicating non-relevant) to the bottom two pairs.

Finally, the average *QSTS* scores for the test split of SQuAD (Zhou et al., 2018) with ProphetNet (Qi et al., 2020), T5 (Footnote 6), and GSA (Zhao et al., 2018) models are shown below.

Model	*QSTS*	BLEU-4
ProphetNet	0.506 ($\pm$ 0.386)	25.80
T5	0.407 ($\pm$ 0.376)	21.32
GSA	0.344 ($\pm$ 0.370)	16.38

The BLEU-4 scores published for these models are shown in the rightmost column of the table and though these published numbers use different data splits for SQuAD compared to ours, we would like to highlight that the overall performance trend of these models as seen by their BLEU-4 scores is also captured by *QSTS*.

In summary, *QSTS* presents as a viable alternative to traditional measures for evaluating QG in terms of its interpretable score components. Moreover, *QSTS* enables a reference-free evaluation for real-world QG scenarios where precompiled lists of reference questions are unavailable.

Limitations: Although *QSTS* addresses several problems with existing QG metrics (Table 1), we note the following caveats that need further work.

1. The *QSTS* function is sensitive to the component scores. Though geometric mean is suggested for summarizing normalized scores (Fleming and Wallace, 1986), and yields higher performance compared to other mean functions in our experiments, other combining functions can be investigated in future.

2. New question-type ontologies are being developed to cover contexts different from extractive QA such as questions within dialog (Cao and Wang, 2021; Svikhnushina et al., 2022; Malhotra et al., 2022). High-accuracy question-class predictors need to be trained for using *QSTS* in these contexts.

3. None of the existing metrics as well as our proposed measure directly incorporate notions such as fluency, interesting-ness, and answerability that humans are able to assess naturally.

Question: What was the title of Bob Dylan's first album?	0.679
Passage: After the eponymous first album, Bob Dylan went on to become the breakthrough songwriter of 'The Freewheelin'	
Question: What was the name of Vincent's brother?	0.889
Passage: Vincent's brother, Theo disagreed vehemently with the placement of Irises.	
Question: Where in Germany was the composer Beethoven born?	0.488
Passage: The composer ludwig van beethoven went deaf in his final years.	
Question: Where is Columbus?	0.0
Passage: Lincoln the 16th president of the United States was born in Kentucky.	

Table 5: *QSTS* scores are shown for anecdotal question-passage pairs

5 Related Work

Models for question generation are being rapidly developed in current NLP research. We refer our readers to a survey article by Pan, et al (2019) for an overview on challenges, existing approaches, and applications for this task. Similar to the standard practice in NLG tasks, question generation has been evaluated using n-gram overlap based metrics such as BLEU (Papineni et al., 2002), METEOR Lavie and Agarwal (2007) and ROUGE (Lin, 2004). While previous studies have found these metrics inadequate for tasks such as summarization, paraphrase generation, and translation (Callison-Burch et al., 2006; Shen et al., 2022), Nema, at al. (2018) specifically study their drawbacks in context of question generation models. Indeed, similar to our approach, they isolate various types of tokens in questions and assign tuned weights to question cue-words, content words, function words, and named entities to compute "answerability" for a question.

In parallel studies, the notion of unsupervised evaluation metrics were studied for dialog systems and machine translation (Liu et al., 2016; Fomicheva et al., 2020) while metric learning was explored for several NLG tasks using transformers in BERTscore, BBScore, and BLEURT (Zhang et al., 2020; Sellam et al., 2020; Shen et al., 2022).

We have highlighted cases where these existing metrics fall short for question comparison and specifically propose reference-free evaluation possibilities for question generation. Reference-free evaluation was previously studied for NLG tasks such as machine translation (Agrawal et al., 2021) and essay grading (Fomicheva et al., 2020).

6 Conclusions and Future Work

We discussed existing metrics for question generation evaluation and highlighted cases where a deeper understanding of question semantics need to be modeled by metrics for a more representative evaluation. As an alternative, we designed the question-sensitive text similarity metric (*QSTS*) that comprises of interpretable scoring components. We also underscored the need for reference-free evaluation in QG systems. The potential of *QSTS* in serving this purpose was demonstrated on a novel dataset *SimQG*, compiled from human judgements on (machine-generated question, passage) pairs. *QSTS* provides, to our knowledge, the first approach to characterizing QG system performance in practical deployments where reference questions may not always be available.

As can be seen in experiments, there is still a large room for improvement for question similarity computation as well as QG evaluation. In future, we hope to pursue these directions further as well as study metric learning approaches for a reference-free evaluation.

Acknowledgments

This research/project is supported by the National Research Foundation, Singapore under its Industry Alignment Fund – Pre-positioning (IAF-PP) Funding Initiative. Any opinions, findings and conclusions or recommendations expressed in this material are those of the author(s) and do not reflect the views of National Research Foundation, Singapore. We thank our anonymous reviewers for their insightful comments. Content related to the error analysis on ComQA and limitations was added as a result of these reviews.

Ethics Statement

This research was conducted in conformance with the ACM Code of Ethics.

References

Abdalghani Abujabal, Rishiraj Saha Roy, Mohamed Yahya, and Gerhard Weikum. 2019. ComQA: A community-sourced dataset for complex factoid question answering with paraphrase clusters. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 307–317, Minneapolis, Minnesota. Association for Computational Linguistics.

Sweta Agrawal, George Foster, Markus Freitag, and Colin Cherry. 2021. Assessing reference-free peer evaluation for machine translation. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 1158–1171, Online. Association for Computational Linguistics.

João António Rodrigues, Chakaveh Saedi, Vladislav Maraev, João Silva, and António Branco. 2017. Ways of asking and replying in duplicate question detection. In *Proceedings of the 6th Joint Conference on Lexical and Computational Semantics (*SEM 2017)*, pages 262–270, Vancouver, Canada. Association for Computational Linguistics.

Chris Callison-Burch, Miles Osborne, and Philipp Koehn. 2006. Re-evaluating the role of Bleu in machine translation research. In *11th Conference of the European Chapter of the Association for Computational Linguistics*, pages 249–256, Trento, Italy. Association for Computational Linguistics.

Shuyang Cao and Lu Wang. 2021. Controllable open-ended question generation with a new question type ontology. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 6424–6439, Online. Association for Computational Linguistics.

Danilo Croce, Alessandro Moschitti, and Roberto Basili. 2011. Structured lexical similarity via convolution kernels on dependency trees. In *Proceedings of the 2011 Conference on Empirical Methods in Natural Language Processing*, pages 1034–1046, Edinburgh, Scotland, UK. Association for Computational Linguistics.

Arthur Deschamps, Sujatha Das Gollapalli, and See-Kiong Ng. 2021. On generating fact-infused question variations. In *Proceedings of the International Conference on Recent Advances in Natural Language Processing (RANLP 2021)*, pages 335–345, Held Online. INCOMA Ltd.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT 2019, Minneapolis, MN, USA, June 2-7, 2019, Volume 1 (Long and Short Papers)*, pages 4171–4186. Association for Computational Linguistics.

Philip J. Fleming and John J. Wallace. 1986. How not to lie with statistics: The correct way to summarize benchmark results. *Commun. ACM*, 29(3):218–221.

Marina Fomicheva, Shuo Sun, Lisa Yankovskaya, Frédéric Blain, Francisco Guzmán, Mark Fishel, Nikolaos Aletras, Vishrav Chaudhary, and Lucia Specia. 2020. Unsupervised quality estimation for neural machine translation. *Transactions of the Association for Computational Linguistics*, 8:539–555.

Qingbao Huang, Mingyi Fu, Linzhang Mo, Yi Cai, Jingyun Xu, Pijian Li, Qing Li, and Ho-fung Leung. 2021. Entity guided question generation with contextual structure and sequence information capturing. *Proceedings of the AAAI Conference on Artificial Intelligence*, 35(14).

Alon Lavie and Abhaya Agarwal. 2007. METEOR: An automatic metric for MT evaluation with high levels of correlation with human judgments. In *Proceedings of the Second Workshop on Statistical Machine Translation*, pages 228–231, Prague, Czech Republic. Association for Computational Linguistics.

Xin Li and Dan Roth. 2002. Learning question classifiers. In *Proceedings of the 19th International Conference on Computational Linguistics - Volume 1*, COLING '02, page 1–7, USA. Association for Computational Linguistics.

Chin-Yew Lin. 2004. ROUGE: A package for automatic evaluation of summaries. In *Text Summarization Branches Out*, pages 74–81, Barcelona, Spain. Association for Computational Linguistics.

David Lindberg, Fred Popowich, John Nesbit, and Phil Winne. 2013. Generating natural language questions to support learning on-line. In *European Workshop on Natural Language Generation*.

Chia-Wei Liu, Ryan Lowe, Iulian Serban, Mike Noseworthy, Laurent Charlin, and Joelle Pineau. 2016. How NOT to evaluate your dialogue system: An empirical study of unsupervised evaluation metrics for dialogue response generation. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 2122–2132, Austin, Texas. Association for Computational Linguistics.

Ganeshan Malhotra, Abdul Waheed, Aseem Srivastava, Md Shad Akhtar, and Tanmoy Chakraborty. 2022. Speaker and time-aware joint contextual learning for dialogue-act classification in counselling conversations. In *Proceedings of the Fifteenth ACM International Conference on Web Search and Data Mining*, WSDM '22, page 735–745, New York, NY, USA. Association for Computing Machinery.

Preksha Nema and Mitesh M. Khapra. 2018. Towards a better metric for evaluating question generation systems. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 3950–3959, Brussels, Belgium. Association for Computational Linguistics.

Şaziye Betül Özateş, Arzucan Özgür, and Dragomir Radev. 2016. Sentence similarity based on dependency tree kernels for multi-document summarization. In *Proceedings of the Tenth International Conference on Language Resources and Evaluation (LREC'16)*, pages 2833–2838, Portorož, Slovenia. European Language Resources Association (ELRA).

Liangming Pan, Wenqiang Lei, Tat-Seng Chua, and Min-Yen Kan. 2019. Recent advances in neural question generation. *CoRR*, abs/1905.08949.

Liangming Pan, Yuxi Xie, Yansong Feng, Tat-Seng Chua, and Min-Yen Kan. 2020. Semantic graphs for generating deep questions. In *ACL*.

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. Bleu: a method for automatic evaluation of machine translation. In *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics*, pages 311–318, Philadelphia, Pennsylvania, USA. Association for Computational Linguistics.

Jeffrey Pennington, Richard Socher, and Christopher Manning. 2014. GloVe: Global vectors for word representation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1532–1543, Doha, Qatar. Association for Computational Linguistics.

Matthew E. Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer. 2018. Deep contextualized word representations. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 2227–2237, New Orleans, Louisiana. Association for Computational Linguistics.

Weizhen Qi, Yu Yan, Yeyun Gong, Dayiheng Liu, Nan Duan, Jiusheng Chen, Ruofei Zhang, and Ming Zhou. 2020. ProphetNet: Predicting future n-gram for sequence-to-SequencePre-training. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 2401–2410, Online. Association for Computational Linguistics.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J. Liu. 2020. Exploring the limits of transfer learning with a unified text-to-text transformer. *Journal of Machine Learning Research*, 21(140):1–67.

Pranav Rajpurkar, Jian Zhang, Konstantin Lopyrev, and Percy Liang. 2016. SQuAD: 100,000+ questions for machine comprehension of text. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 2383–2392, Austin, Texas. Association for Computational Linguistics.

Nils Reimers and Iryna Gurevych. 2019. Sentence-BERT: Sentence embeddings using Siamese BERT-networks. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 3982–3992, Hong Kong, China. Association for Computational Linguistics.

Thibault Sellam, Dipanjan Das, and Ankur Parikh. 2020. BLEURT: Learning robust metrics for text generation. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7881–7892, Online. Association for Computational Linguistics.

Lingfeng Shen, Haiyun Jiang, Lemao Liu, and Shuming Shi. 2022. Revisiting the evaluation metrics of paraphrase generation.

Megha Srivastava and Noah Goodman. 2021. Question generation for adaptive education. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 2: Short Papers)*, pages 692–701, Online. Association for Computational Linguistics.

Ekaterina Svikhnushina, Iuliana Voinea, Anuradha Welivita, and Pearl Pu. 2022. A taxonomy of empathetic questions in social dialogs. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 2952–2973, Dublin, Ireland. Association for Computational Linguistics.

Harish Tayyar Madabushi, Mark Lee, and John Barnden. 2018. Integrating question classification and deep learning for improved answer selection. In *Proceedings of the 27th International Conference on Computational Linguistics*, pages 3283–3294, Santa Fe, New Mexico, USA. Association for Computational Linguistics.

Jian Wang, Junhao Liu, Wei Bi, Xiaojiang Liu, Kejing He, Ruifeng Xu, and Min Yang. 2020a. Improving knowledge-aware dialogue generation via knowledge base question answering. *AAAI*.

Liuyin Wang, Zihan Xu, Zibo Lin, Haitao Zheng, and Ying Shen. 2020b. Answer-driven deep question generation based on reinforcement learning. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 5159–5170, Barcelona, Spain (Online). International Committee on Computational Linguistics.

Tianyi Zhang, Varsha Kishore, Felix Wu, Kilian Q. Weinberger, and Yoav Artzi. 2020. Bertscore: Evaluating text generation with BERT. In *8th International*

Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, April 26-30, 2020. OpenReview.net.

Yao Zhao, Xiaochuan Ni, Yuanyuan Ding, and Qifa Ke. 2018. Paragraph-level neural question generation with maxout pointer and gated self-attention networks. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 3901–3910, Brussels, Belgium. Association for Computational Linguistics.

Qingyu Zhou, Nan Yang, Furu Wei, Chuanqi Tan, Hangbo Bao, and Ming Zhou. 2018. Neural question generation from text: A preliminary study. In *Natural Language Processing and Chinese Computing*, pages 662–671, Cham. Springer International Publishing.

Wenjie Zhou, Minghua Zhang, and Yunfang Wu. 2019. Question-type driven question generation. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 6032–6037, Hong Kong, China. Association for Computational Linguistics.

Noun-MWP: Math Word Problems Meet Noun Answers

Taehun Cha* and **Jaeheun Jung*** and **Donghun Lee**
Department of Mathematics
Korea University
{cth127,wodsos,holy}@korea.ac.kr

Abstract

We introduce a new type of problems for math word problem (MWP) solvers, named Noun-MWPs, whose answer is a non-numerical string containing a noun from the problem text. We present a novel method to empower existing MWP solvers to handle Noun-MWPs, and apply the method on Expression-Pointer Transformer (EPT). Our model, N-EPT, solves Noun-MWPs significantly better than other models, and at the same time, solves conventional MWPs as well. Solving Noun-MWPs may lead to bridging MWP solvers and traditional question-answering NLP models.

1 Introduction

Question-answering (QA) (Woods, 1968) and QA problems involving mathematics (Bobrow, 1964) are one of the classic NLP problems in computer science. Later, mathematics education community investigated a class of problems coined as Math Word Problems (MWPs) (Nesher and Katriel, 1977; Caldwell and Goldin, 1979; Ballew and Cunningham, 1982). Research in solving MWPs with computers started from purely rule-based approaches (Hosseini et al., 2014) and statistics-based approaches (Kushman et al., 2014), through hybrids (Roy and Roth, 2015), and more recently deep-learning-based approaches (Wang et al., 2017, 2018; Li et al., 2019).

Proportionally, the number of publicly available dataset for MWPs increased; for example, `Alg514` (Kushman et al., 2014), `DRAW` (Upadhyay and Chang, 2015), `MAWPS` (Koncel-Kedziorski et al., 2016), `Math23K` (Wang et al., 2017), `AllArith` (Roy and Roth, 2017), `Dolphin18K` (Huang et al., 2016), and `ASDiv-A` (Miao et al., 2020). The mathematical depth of the problems in MWP datasets vary, yet the majority of datasets require only numerical answers. On the other hand, math problems whose answer is a non-numeric substring of the inputs are frequently seen in real-world math problems in mathematics education.

MWP
Chloe was organizing her bookcase making sure each of the shelves had exactly 6 books on it. If she had 5 shelves of mystery books and 4 shelves of picture books, how many books did she have in total?
$\rightarrow (6 \times 5) + (6 \times 4)$
Extractive QA
In 1517, the seventeen-year-old King sailed to Castile. There, his Flemish court . . . . In May 1518, Charles traveled to Barcelona in Aragon. Where did Charles travel to first, Castile or Barcelona?
$\rightarrow arg\ min\{Castile : 1517, Barcelona : 1518\}$
Noun-MWP
Sihyeon collected 64 stamps, and Soma collected 1 fewer stamps than Sihyeon. Junwoo collected 7 bundles of 10 stamps each. Who among the three has collected the fewest stamps?
$\rightarrow arg\ min\{Sihyeon : 64, Soma : 64 - 1, Junwoo : 7 \times 10\}$

Table 1: Highlighted examples comparing MWP, Noun-MWP, and Extractive QA problems. Relevant numbers, nouns, and a novel element in Noun-MWP are marked in red, blue, and green, respectively.

Our Contribution

- We bring attention to Noun-MWP, a class of MWPs whose answer is a noun-substring of the input string.
- We propose a systematic approach to empower existing MWP solvers to solve Noun-MWPs as well.
- We implement and empirically validate our approach using a Korean Noun-MWP dataset.

*equal contribution

Proceedings of the 29th International Conference on Computational Linguistics, pages 3847–3857
October 12–17, 2022.

2 Related Works

Our work emphasizes the need for a new class of MWPs to complement the MWPs represented by existing datasets. From a similar stance, ASDiv-A dataset (Miao et al., 2020) emphasizes the need to have greater lexical diversity in MWP datasets, and S-VAMP dataset (Patel et al., 2021) emphasizes the underrepresented adversarial variations of the MWPs. Noun-MWP may be considered as a next-level adversarial variations of pre-existing MWPs, since as of now, adversarial generation methods of MWPs (Kumar et al., 2021) do not exceed the current implicit limit of MWPs focused on problems with numerical answers.

Noun-MWPs can also be seen as extractive QA problems that need mathematics-based reasoning to reach the correct solution. DROP dataset (Dua et al., 2019), as shown in the middle panel of Table 1, introduces basic mathematical reasoning in QA tasks. The direct approach to populate mathematical operations with arguments (Andor et al., 2019) faces limitation due to combinatorial explosion of search space in possible mathematical expressions, and injecting custom math operations and their templates (Geva et al., 2020) alleviate this drawback somewhat. To circumvent this, we start from MWP perspective and use Noun-MWPs as a gateway to solve extractive QA problems whose solutions contain more complicated mathematical expressions.

3 Methodology

3.1 Problem Definition

We consider Noun-MWP as a special type of math word problems whose correct answer is a string containing a noun from the problem context. In particular, we consider problems requiring algebraic operations $+, \times, -, \div$ and comparison between numbers. The difficulty of these problems roughly corresponds to the problems taught in elementary school level mathematics classes. Table 1 and Table 2 contain two examples.

To infer a correct solution from the problem in Table 2, a well-constructed series of mathematical inference and language processing is needed. First of all, how much milk Eunhye drank must be inferred from '3/7' and 'rest.' Then, the inferred number '4/7' must be assigned to the corresponding noun 'Eunhye.' Finally, based on the query, the correct noun answer must be determined. For an NLP model to solve a collection of noun-MWPs, both MWP and extractive QA capabilities are needed, especially in handling multi-step computation comprised of multiple mathematical operations correctly.

Problem(KR)
아린이는 우유 한 병 전체의 3/7을 마시고, 나머지는 은혜가 마셨습니다. 둘 중 우유를 더 적게 마신 사람은 누구일까요?
Problem(EN)
Arin drank 3/7 of the whole bottle of milk, and Eunhye drank the rest. Which of the two drank less milk?
Candidates : Arin, Eunhye Expressions : {Arin : $3/7$, Eunhye : $1 - 3/7$} Query : Which of the two drank less milk?

Table 2: A Noun-MWP example question presented in original Korean language and its translation to English, and the sample solution comprised of candidates (in blue), mathematical expressions, and query.

3.2 Solving Noun-MWP

In this section, we sketch the key idea of solving Noun-MWPs starting with a MWP solver. We decompose the process of solving Noun-MWP into three conceptual steps:

Candidate selection: Extract the candidate nouns from the context and generate the list of candidates, as shown in Table 2.

Expression assignment: Generate mathematical expressions that produce numbers. These numbers may be assigned to the candidates.

Query interpretation: Interpret the query into a representation compatible to the underlying MWP solver. Resulting representation allows the MWP solver to find the most suitable candidate to answer.

By implementing the three-step process on a multi-equation MWP solver, it is possible to empower the solver with necessary information to solve Noun-MWPs while preserving its original capacity to solve MWPs.

3.3 N-EPT: Noun problem solving EPT

To demonstrate the validity of our approach, we use EPT (Kim et al., 2020) as the baseline multi-equation MWP solver and empower it to solve Noun-MWPs. The original EPT solves the MWPs with the binary tree-based expression tokens, which consist of an operation and operands. They used

TAR(target) Contrast to special token VAR(variable) of original EPT, we treat it as an unary operator paired with target noun entity appeared in the problem. This allows the model to infer the candidates and equations to assign values to the noun entities.
arg Special command to generate list of pairs (noun, value) after the expression assignment. This works to separate classic MWPs, which don't require noun information, from our problem.
find and **ord(order)** Binary list operators extracting answer from the list(resp. sorted list) satisfying query. For example, (ord,R_i, n) means the n-th smallest element of the list stored at R_i. Similarly the (find,R_i, n) works on R_i and choose element indexed by n.

Table 3: Details on new operations of N-EPT, in constrast to EPT.

$+, -, \times, \div$ as operations, and applied an attention block to extract operands from the problem text. Maintaining the backbone structure of the EPT, our resulting MWP solver, named N-EPT[1], contains the following key changes:

- **Additional Operation Tokens**: We introduce four more operations, *TAR, arg, find, ord* to extend the expressive power of EPT to cover Noun-MWPs. Tokens and their operations are detailed in Table 3.
- **Noun attention**: An attention module is added to predict candidate nouns from the question text.
- **Pretrained word embedding**: N-EPT uses pretrained T5 embedding. For example, a pretrained embedding of a word *subtract* is used to represent subtracting operation.
- **T5 encoder-decoder**: In contrast to EPT, which uses ALBERT encoder and transformer decoder, N-EPT uses pretrained T5 encoder and decoder (KETI AIRC, 2021). This allows utilizing pretrained word embeddings and language-specific pretrained models.

[1]The code is available on https://github.com/invigorator96/NounMWP

Additional procedures were added to handle agglutinative characteristic of Korean language – for example, postpositions included in predicted nouns are omitted to facilitate exact answer matching. An implementation example is in Appendix A.

4 Experimental Study

4.1 Dataset

We construct a novel dataset comprised of Noun-MWPs for empirical validation. We collect Noun-MWP problems and their answers from elementary school level mathematics textbooks (최순미, 2020; 한헌조, 2018; 김은영, 2018) in South Korea, and label their mathematical expression manually. In addition, problems from a Korean MWP dataset curated by TUNiB (Keum et al., 2022) are also selected and relabeled. We validated the annotated solution labels against the answers from the source textbooks, by checking whether a rule based solver provided with the solution label can produce the same answer as the answers from the source textbooks. The resulting dataset contains 604 question-expression-answer triplets,whose basic statistics are in Table 4.

Ratio of Problems by Required Operations				
$+$	$-$	$\times$	$\div$	Simple Assignment
20.9%	8.4%	18.4%	9.3%	55.8%

Ratio of Problems by Expression Length				
≤ 8	$9 \sim 10$	$11 \sim 12$	$13 \sim 14$	≥ 15
46.5%	39.4%	9.6%	4.1%	0.3%

Table 4: Statistics of our Noun-MWP dataset

Simple Assignment cases in Table 4 correspond to the problems solved with only simple comparison or sorting, without any arithmetic operation. Our dataset includes problems requiring more than two operations to solve, so the percentages do not sum up to 100. The expression length of a problem represents the complexity of mathematical reasoning to deduce the correct answer, and is an intuitive gauge of problem difficulty.

4.2 Noun-MWP Performance

Using the Korean Noun-MWP dataset, we validate the performance of N-EPT implementation

via 5-fold cross validation. As shown in Table 5, N-EPT achieves over 80% in cross validated accuracy measured via exact match score, regardless of the choice of T5 encoder-decoder size – small, base, or large. Detailed methods used in training N-EPT and in-depth performance results are in Appendix B.

	N-EPT	KE-T5	KLUE-RoBERTa
Small	81.67%	−%	45.67%
(Std.Err)	(2.11%)	(-%)	(4.52%)
Base	**84.33%**	6.80%	46.17%
(Std.Err)	(2.44%)	(9.58%)	(5.21%)
Large	82.50%	28.17%	47.67%
(Std.Err)	(0.75%)	(2.44%)	(3.22%)

Table 5: 5-fold CV accuracy in Noun-MWPs, by N-EPT and two SOTA QA models.

Conventional MWP solvers cannot handle Noun-MWPs, as those solvers generate numerical answers. As an alternative, we use state-of-the-art (SOTA) QA models as benchmark algorithms, since QA models are capable of produce non-numerical answers based on the given question.

The first benchmark QA model is KE-T5, a SOTA Korean-pretrained version of T5 (Raffel et al., 2020) that achieves 86.27% accuracy on KorQuAD 1.1 dataset (Korean extractive QA benchmark) (KETI AIRC, 2021). KE-T5 fails to answer a great majority of Noun-MWP questions; whereas N-EPT with the same model specification consistently performs much better. As our N-EPT implementation contains KE-T5, the difference in performance between the two suggests that the expressive power added to KE-T5 by changes in Section 3.3 is very effective in solving Noun-MWPs.

We also use KLUE-RoBERTa (Park et al., 2021), a SOTA Korean-pretrained variant of RoBERTa (Liu et al., 2019) that achieves 75.58% accuracy on KLUE-MRC (Korean machine reading comprehension) benchmark as another benchmark QA model. KLUE-RoBERTa may be handling a smaller subset of problems in our Noun-MWP dataset than N-EPT, as it shows consistent performance that peaks at 47.67%, much less than 81.67%, the worst performance of N-EPT. Significant empirical advantage of N-EPT over both QA models suggest that empowering a conventional MWP solver is an effective approach to solve Noun-MWPs.

4.3 Sanity Check with Conventional MWPs

We backtest N-EPT on `MAWPS` dataset to verify that our approach retains the capacity of the original MWP solver (EPT) to handle conventional MWPs. Since `MAWPS` dataset is in English, we modify N-EPT structure by substituting KE-T5 with Google T5. Meanwhile, as the original implementation of EPT uses ALBERT, we also use our own implementation of EPT with T5 as another benchmark. We report the mean and the standard deviation of 5-fold cross-validated accuracy in Table 6.

	ALBERT model size		
	base	large	xlarge
EPT(orig.)	83.41%	84.51%	−%
(Std.Err)	(0.32%)	(1.37%)	(-%)
	T5 model size		
	small	base	large
EPT(T5)	83.35%	84.03%	85.12%
(Std.Err)	(1.70%)	(1.80%)	(1.92%)
N-EPT	83.27%	84.49%	85.50%
(Std.Err)	(1.85%)	(1.59%)	(1.53%)

Table 6: 5-fold CV accuracy in MAWPS dataset. EPT(orig.) values are from the original source (Kim et al., 2020).

Similar performance between EPT with ALBERT and EPT with T5 suggests that the choice of the encoder-decoder model has negligible effect on performance. Meanwhile, unlike ALBERT that shows instabilities in a larger model specification, T5 is stable throughout all specifications.

We implement EPT with T5 by ablating N-EPT, such that it does not have the other changes shown in Section 3.3 to empower EPT for Noun-MWPs besides adding T5. Therefore, the similar performance of the two suggests that N-EPT successfully retains the expressive power of EPT in handling conventional MWPs.

5 Discussion and Future Work

N-EPT demonstrates how to empower a conventional MWP solver to handle Noun-MWPs. Considering Noun-MWPs are a staple of basic math education, they deserve to be included in the benchmark for a novel MWP solver to gauge its ability to understand both the mathematics and the natural language in the problems.

Despite the ability to handle both conventional

MWPs and Noun-MWPs, N-EPT leaves much to be improved against harder questions. For example, problems whose expression length is longer or whose question string contains more candidates are often incorrectly answered by N-EPT. Detailed plots are placed in Appendix C, and select examples where N-EPT incorrectly answers are given in Appendix D.

6 Conclusion

We introduce Noun-MWP, a class of problems that are frequently used in mathematics education, and yet to be handled by MWP solvers. We present a three-step method, Candidate Selection, Expression Assignment, and Query Interpretation, to empower existing MWP solvers to handle Noun-MWP as well as the conventional MWPs. As a proof-of-concept implementation, we construct N-EPT using EPT as its baseline MWP solver, and the new Noun-MWP dataset consists of 604 questions-expression-answer pairs to validate the model's performance. By modifying the original EPT to handle noun information and additional operations, our N-EPT significantly outperforms benchmark models in Noun-MWPs and retains the capacity of the original MWP solver. Noun-MWPs may serve as a novel testbed for MWP solvers to gauge their understanding of mathematical logic and natural language.

Acknowledgements

This work is partly supported by Institute for Information & communications Technology Promotion (IITP) grant funded by the Korea government (MSIT) (No. 2021-0-02153, Large scale deep learning based algorithm for mathematics problem solving), and by the National Research Foundation of Korea (NRF) grant funded by the Korea government (MSIT) (No. 2020R1G1A1102828). We thank Hanyoung Kim, Dongwon Kim, Hansol Jeon, Daeseong Kim, Yanggee Kim, Hyunsang Hwang, Keunsuk Cho, and Changhae Jung for their assistance in data collection and processing.

References

Daniel Andor, Luheng He, Kenton Lee, and Emily Pitler. 2019. Giving BERT a calculator: Finding operations and arguments with reading comprehension. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 5947–5952, Hong Kong, China. Association for Computational Linguistics.

Hunter Ballew and James W Cunningham. 1982. Diagnosing strengths and weaknesses of sixth-grade students in solving word problems. *Journal for research in mathematics education*, 13(3):202–210.

Daniel G Bobrow. 1964. A question-answering system for high school algebra word problems. In *Proceedings of the October 27-29, 1964, fall joint computer conference, part I*, AFIPS '64 (Fall, part I), pages 591–614, New York, NY, USA. Association for Computing Machinery.

Janet H Caldwell and Gerald A Goldin. 1979. Variables affecting word problem difficulty in elementary school mathematics. *Journal for Research in Mathematics Education*, 10(5):323–336.

Dheeru Dua, Yizhong Wang, Pradeep Dasigi, Gabriel Stanovsky, Sameer Singh, and Matt Gardner. 2019. DROP: A reading comprehension benchmark requiring discrete reasoning over paragraphs. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 2368–2378, Minneapolis, Minnesota. Association for Computational Linguistics.

Mor Geva, Ankit Gupta, and Jonathan Berant. 2020. Injecting numerical reasoning skills into language models. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 946–958, Online. Association for Computational Linguistics.

Mohammad Javad Hosseini, Hannaneh Hajishirzi, Oren Etzioni, and Nate Kushman. 2014. Learning to solve arithmetic word problems with verb categorization. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 523–533, Doha, Qatar. Association for Computational Linguistics.

Danqing Huang, Shuming Shi, Chin-Yew Lin, Jian Yin, and Wei-Ying Ma. 2016. How well do Computers Solve Math Word Problems? Large-Scale Dataset Construction and Evaluation. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 887–896, Berlin, Germany. Association for Computational Linguistics.

KETI AIRC. 2021. KE-T5: Korean English T5. https://github.com/AIRC-KETI/ke-t5.

Bitna Keum, Myeonghyeon Ryu, Yoseph Ham, Minsuh Seo, Heechang Jo, Hangyeol Kim, and Kyubyong Park. 2022. Kmwp, korean math word problems. https://github.com/tunib-ai/KMWP.

Bugeun Kim, Kyung Seo Ki, Donggeon Lee, and Gahgene Gweon. 2020. Point to the Expression: Solving Algebraic Word Problems using the Expression-Pointer Transformer Model. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 3768–3779.

Rik Koncel-Kedziorski, Subhro Roy, Aida Amini, Nate Kushman, and Hannaneh Hajishirzi. 2016. MAWPS: A Math Word Problem Repository. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 1152–1157, San Diego, California. Association for Computational Linguistics.

Vivek Kumar, Rishabh Maheshwary, and Vikram Pudi. 2021. Adversarial examples for evaluating math word problem solvers. In *Findings of the Association for Computational Linguistics: EMNLP 2021*, pages 2705–2712, Punta Cana, Dominican Republic. Association for Computational Linguistics.

Nate Kushman, Yoav Artzi, Luke Zettlemoyer, and Regina Barzilay. 2014. Learning to Automatically Solve Algebra Word Problems. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 271–281, Baltimore, Maryland. Association for Computational Linguistics.

Jierui Li, Lei Wang, Jipeng Zhang, Yan Wang, Bing Tian Dai, and Dongxiang Zhang. 2019. Modeling Intra-Relation in Math Word Problems with Different Functional Multi-Head Attentions. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 6162–6167, Florence, Italy. Association for Computational Linguistics.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. RoBERTa: A Robustly Optimized BERT Pretraining Approach.

Shen-yun Miao, Chao-Chun Liang, and Keh-Yih Su. 2020. A diverse corpus for evaluating and developing English math word problem solvers. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 975–984, Online. Association for Computational Linguistics.

Pearla Nesher and Tamar Katriel. 1977. A Semantic Analysis of Addition and Subtraction Word Problems in Arithmetic. *Educational Studies in Mathematics*, 8(3):251–269.

Sungjoon Park, Jihyung Moon, Sungdong Kim, Won Ik Cho, Ji Yoon Han, Jangwon Park, Chisung Song, Junseong Kim, Youngsook Song, Taehwan Oh, Joohong Lee, Juhyun Oh, Sungwon Lyu, Younghoon Jeong, Inkwon Lee, Sangwoo Seo, Dongjun Lee, Hyunwoo Kim, Myeonghwa Lee, Seongbo Jang, Seungwon Do, Sunkyoung Kim, Kyungtae Lim, Jongwon Lee, Kyumin Park, Jamin Shin, Seonghyun Kim, Lucy Park, Lucy Park, Alice Oh, Jung-Woo Ha (NAVER AI Lab), Kyunghyun Cho, and Kyunghyun Cho. 2021. KLUE: Korean Language Understanding Evaluation. In *Proceedings of the Neural Information Processing Systems Track on Datasets and Benchmarks*, volume 1.

Arkil Patel, Satwik Bhattamishra, and Navin Goyal. 2021. Are NLP models really able to solve simple math word problems? In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 2080–2094, Online. Association for Computational Linguistics.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J. Liu. 2020. Exploring the limits of transfer learning with a unified text-to-text transformer. *Journal of Machine Learning Research*, 21(140):1–67.

Subhro Roy and Dan Roth. 2015. Solving general arithmetic word problems. In *Conference on Empirical Methods in Natural Language Processing, EMNLP 2015*, pages 1743–1752. experts.illinois.edu.

Subhro Roy and Dan Roth. 2017. Unit Dependency Graph and Its Application to Arithmetic Word Problem Solving. *Proceedings of the AAAI Conference on Artificial Intelligence*, 31(1).

Shyam Upadhyay and Ming-Wei Chang. 2015. Draw: A challenging and diverse algebra word problem set. Technical report, Citeseer.

Lei Wang, Yan Wang, Deng Cai, Dongxiang Zhang, and Xiaojiang Liu. 2018. Translating a math word problem to a expression tree. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 1064–1069, Brussels, Belgium. Association for Computational Linguistics.

Yan Wang, Xiaojiang Liu, and Shuming Shi. 2017. Deep neural solver for math word problems. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 845–854, Copenhagen, Denmark. Association for Computational Linguistics.

W A Woods. 1968. Procedural semantics for a question-answering machine. In *Proceedings of the December 9-11, 1968, fall joint computer conference, part I*, AFIPS '68 (Fall, part I), pages 457–471, New York, NY, USA. Association for Computing Machinery.

Qinzhuo Wu, Qi Zhang, Jinlan Fu, and Xuanjing Huang. 2020. A knowledge-aware sequence-to-tree network for math word problem solving. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 7137–7146, Online. Association for Computational Linguistics.

김은영. 2018. *기적의 수학 문장제*. 길벗스쿨, Seoul,KR.

징검다리 교육연구소 최순미. 2020. *나 혼자 푼다! 수학 문장제 - 서술형 기본서 :초등*. 이지스에듀, Seoul,KR.

김성국 한헌조. 2018. *씨투엠수학독해*. (주)씨투엠에듀, Suwon,KR.

Appendix

A Implementation Example

The sequence of expressions in Table A1 shows how N-EPT uses its operations to solve the sample Noun-MWP question shown in Table 2. Note that

Label	Expression	
R_0	$[TAR, Arin]$	Candidate selection
R_1	$[TAR, Eunhye]$	
R_2	$[=, R_0, 3/7]$	Expression assignment
R_3	$[-, 1, R_0]$	
R_4	$[=, R_1, R_3]$	
R_5	$[arg]$	Query interpretation
R_6	$[ord, R_5, 1]$	
R_7	$[END]$	

Table A1: N-EPT expression sequence to solve the Noun-MWP in Table 2.

R_n stands for the n-th token. The solver first select candidates with R_0 and R_1, and assign proper expressions to each candidates with $R_2 \sim R_4$. Then a special command R_5 construct a list of (candidate, expression) pairs, and finally, R_6 determine the query.

B Experiment Details

Five A6000 GPUs are utilized to run all folds parallelly, and we checked accuracy score on test set for every 20 training epochs. We used all different batch sizes for each model but used gradient accumulation technique to update gradients of 64 problems uniformly. After hyperparameter tuning with learning rate from [3e-4, 5e-5, 1e-5] with an ADAMW optimizer and dropout probability from [0.0, 0.1], 3e-4 and 0.1 were optimal for each. We also used label smoothing, gradient clipping and warm up technique as an original EPT did.

As a result, N-EPT achieved following results.

500epoch column presents final accuracy of all folds while *max* column shows the best accuracy on each fold. Though the *Base* model achieved the best performance among others, the *Large* model presented the smallest standard deviation on both columns. It can be an evidence of robustness of the *Large* model.

	N-EPT	
	500epoch	max
Small	79.67 ±2.33%	81.67 ±2.11%
Base	84.00 ±2.44%	**84.33** ±2.44%
Large	81.00 ±0.97%	82.50 ±0.75%

Table B2: 5-fold cross validated performance of new algorithm on Noun-MWPs.

C Impact of question complexity

We checked the impact of expression length by visualizing error rate and error count for each length. We used *base* model and gathered test sets in all five folds. It is natural to assume that error rate grows proportionally to expression length, because a question with longer expression is harder to solve. Figure C1 below shows consistent result with our assumption(Orange bar shows error rates for each category).

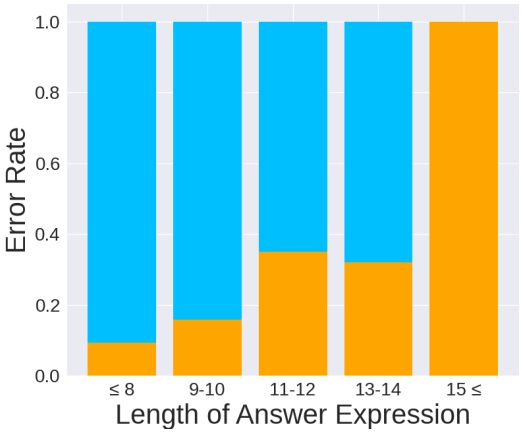

Figure C1: Impact of expression length to error rate.

There are two reasons to explain this result. The first is the relationship between expression length and question complexity. It is consistent with the report on (Wu et al., 2020), which showed decreasing performance of various models with respect to the equation length on Math23K dataset. The second is the sparsity of data with long expression in our dataset. Recall the Table 4 showing that problems in each category is highly skewed to short problems. Similar phenomena is observed when we visualize the error rate with the number of targets, candidates for answer.

Error rate is low with five targets because easy question requiring a 'find' operator consists most

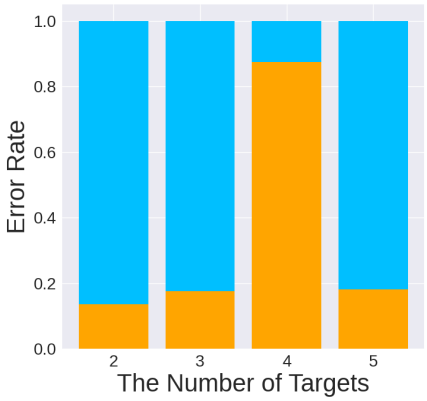

Figure C2: Impact of the number of TAR to error rate.

of the category. When there's too many targets to catch, model tends to miss some of them. Examples are presented in Appendix D. Since the distribution of the number of target is also highly skewed, same explanations mentioned above can be applied here, too.

To check the absolute count of error case and the number of problems of each category, see Figure C3 and C4.

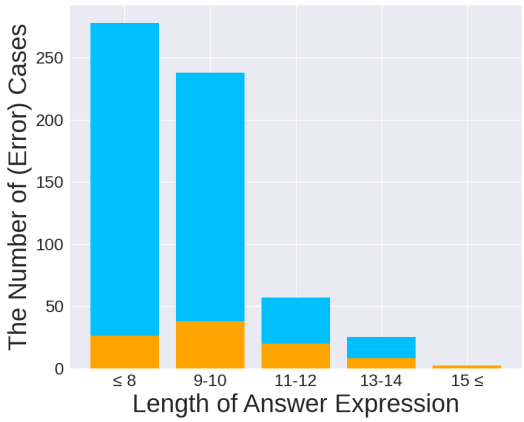

Figure C3: Impact of expression length to the number of error cases.

D Error Analysis

In this section, we make qualitative analysis on error cases. We implemented same procedure from Appendix C. Our model showed worse performance on complicated question requiring longer expressions. The Table D3 shows an example.

Though our model succeeded to find candidate nouns (*Sujin, Cheolmin* and *Youngsoo*), catch the intention of the query(*the most*) and compute the

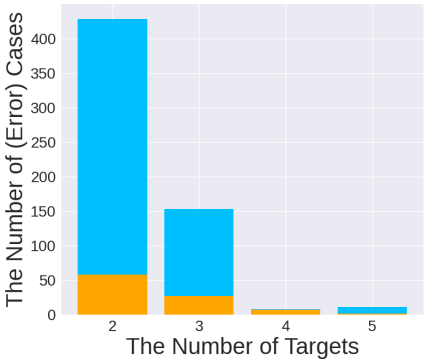

Figure C4: Impact of the number of TAR to the number of error cases.

Problem(EN)
Sujin drank 300mL of milk for two days, and Cheolmin drank 1,000mL more than three times the milk Sujin drank. Young-soo drank 1,000mL of milk every day for 12 days. Who drank the most milk?
Predicted expression
X_0: (TAR, Sujin), X_1: (TAR, Cheolmin), X_2: (TAR, Youngsoo) 300×2, $X_0 = X_2$, $X_0 \times 12$, $X_1 \times 12$ (arg), (ord, R_i, -1)

Table D3: Error case of our model

X_0 value(300×2), it failed to assign it to X_0 and successively failed to generate proper expressions. This shows the difficulty of generating long sequences, which generative models usually goes through.

	count	percent
Candidate Selection	55	58.5%
Expression Assignment	22	23.4%
Query Interpretation	17	18.1%
Total	94	100.0%

Table D4: Counts and percentage of error cases for each step. We added the number of error cases for all five folds.

In addition to the length issue, various reasons made error cases. As shown in Table D4, more than half of the error cases came from the Candidate Selection step. To improve the model, it would be necessary to enhance the Candidate Selection step, which is left to future research. We conclude the section with sample problems that our model failed to give a correct answer. The selected problems are sampled from various test folds in 5-fold cross validation experiment.

Problem(KR) 국제 어린이 학교에 중국 학생이 53명, 미국 학생은 60명 있습니다. 한 모둠에 10명씩 모았을 때 일본 학생은 5모둠에 5명이 남고, 한국 학생은 4모둠에 9명이 남습니다. 학생이 두 번째로 많은 나라는 어디인지 써 보세요.
Problem(EN) There are 53 Chinese students and 60 American students in international children's schools. When 10 students are gathered in a group, 5 Japanese students are left in 5 groups, and 9 Korean students are left in 4 groups. Write down which country has the second largest number of students.
True expressions
Candidate selection: X_0:Chinese, X_1: American, X_2: Japanese, X_3: Korean
expression assignment: $X_0 = 53$, $X_1 = 60$, $X_2 = 5 \times 10 + 5$, $X_3 = 4 \times 10 + 9$
Query interpretation: (arg),(ord,R_i,-2)
Answer: Japanese
Predicted expressions
Candidate selection: X_0:Chinese, X_1: American, X_2: Korean
expression assignment: $X_0 = 53$, $X_1 = 60$, $X_2 = 9$
Query interpretation: (arg),(ord,R_i,2)
Answer: Chinese

Table D5: Incorrect sample problem. The model failed to find candidate **Japanese**, successively the expressions and query. Note that this is the problem with longest expression with expression length 15.

Problem(KR) 학교에서 국어, 수학, 영어, 과학, 사회의 순서로 시험을 봤습니다. 두 번째로 시험을 본 과목은 무엇입니까?
Problem(EN) At school, I took the test in the order of Korean, math, English, science, and social studies. What subject did I take the exam for the second time?
True expressions
Candidate selection: X_0:Korean, X_1: math, X_2: English, X_3: Science, X_4:Social studies
expression assignment: $X_0 = 1$, $X_1 = 2$, $X_2 = 3$,$X_3 = 4$,$X_4 = 5$
Query interpretation: (arg),(find,R_i,2)
Answer: math
Predicted expressions
Candidate selection: X_0:Korean, X_1: math, X_2: English, X_3: Science, X_4:Social studies
expression assignment: $X_0 = 1$, $X_1 = 2$, $X_2 = 3$,$X_3 = 4$,$X_4 = 5$
Query interpretation: (arg),(find,R_i,3)
Answer: English

Table D6: Incorrect sample problem. The model successfully found candidates and intended expressions, but failed to find index operand of **find** operator.

Problem(KR) 선희는 30분에 10/3km를 걷고, 진혜는 30분에 2와 2/3km를 걷습니다. 더 천천히 걷는 사람은 누구일까요?
Problem(EN) Sun-hee walks 10/3 kilometers in 30 minutes, and Jin-hye walks 2 and 2/3 kilometers in 30 minutes. Who walks more slowly?
True expressions
Candidate selection: X_0:Sun-hee, X_1: Jin-hye
expression assignment: $X_0 = (10/3)$,$X_1 = (2 + (2/3))$
Query interpretation: (arg),(ord,R_i,1)
Answer: Jin-hye
Predicted expressions
Candidate selection: X_0:Sun-hee, X_1: Jin-hye
expression assignment: $X_0 = (10/3) \div 30$, $X_1 = 2 + (2/3)$
Query interpretation: (arg),(ord,R_i,1)
Answer: Sun-hee

Table D7: Incorrect sample problem. The model successfully found candidates and question token, but not correct expressions. The model tried to find the velocity of Sun-hee but not for Jin-hye.

Problem(KR) A는 한 변의 길이가 5cm인 정사각형, B는 가로가 5cm, 세로가 8cm인 직사각형, C는 한 변의 길이가 3cm인 정칠각형입니다. 이중에서 , 둘레가 가장 긴 도형을 찾으세요.
Problem(EN) A is a square with a side length of 5 cm, B is a rectangle with a width of 5 cm and a height of 8 cm, and C is a regular heptagon with a side length of 3 cm. Find the shape with the longest circumference.
True expressions
Candidate selection: X_0: A, X_1: B, X_2: C
expression assignment: $X_0 = 5 \times 4$, $X_1 = 2 \times (5 + 8)$, $X_2 = 3 \times 7$
Query interpretation: (arg),(ord,R_i, -1)
Answer: B
Predicted expressions
Candidate selection: X_0: A, X_1: B, X_2: C
expression assignment: $X_0 = 5 \times 10$, $X_1 = 2 \times (5 + 8)$, $X_2 = 3 \times 5$
Query interpretation: (arg),(ord,R_i, -1)
Answer: A

Table D8: Incorrect sample problem. The prior knowledge of square,rectangle and regular heptagon is required to solve this problem, but our model did not learn it. As a consequence, the expression assignment is not correct. Note that the model succeeded to get circumference of rectangle.

ViNLI: A Vietnamese Corpus for Studies on Open-Domain Natural Language Inference

Tin Van Huynh, Kiet Van Nguyen, Ngan Luu-Thuy Nguyen
Faculty of Information Science and Engineering, University of Information Technology,
Ho Chi Minh City, Vietnam
Vietnam National University, Ho Chi Minh City, Vietnam
{tinhv,kietnv,ngannlt}@uit.edu.vn

Abstract

Over a decade, the research field of computational linguistics has witnessed the growth of corpora and models for natural language inference (NLI) for rich-resource languages such as English and Chinese. A large-scale and high-quality corpus is necessary for studies on NLI for Vietnamese, which can be considered a low-resource language. In this paper, we introduce ViNLI (**Vi**etnamese **N**atural **L**anguage **I**nference), an open-domain and high-quality corpus for evaluating Vietnamese NLI models, which is created and evaluated with a strict process of quality control. ViNLI comprises over 30,000 human-annotated premise-hypothesis sentence pairs extracted from more than 800 online news articles on 13 distinct topics. In this paper, we introduce the guidelines for corpus creation which take the specific characteristics of the Vietnamese language in expressing entailment and contradiction into account. To evaluate the challenging level of our corpus, we conduct experiments with state-of-the-art deep neural networks and pre-trained models on our dataset. The best system performance is still far from human performance (a 14.20% gap in accuracy). The ViNLI corpus is a challenging corpus to accelerate progress in Vietnamese computational linguistics. Our corpus is available publicly for research purposes[1].

1 Introduction

Although over 98 million people speak Vietnamese globally[2], Vietnamese is considered a low-resource language for natural language processing (NLP) research because of the lack of human-annotated corpora. To help accelerate NLP progress, Nguyen et al. (2020b, 2022) and Doan et al. (2021) collected a large number of human-annotated data to benchmark Vietnamese NLP tasks. We built the ViNLI corpus for evaluating natural language inference (NLI) models. NLI is an emerging and important task in natural language understanding which is to predict the semantic relation between two sentences. Several English corpora were released for the NLI task (Bowman et al., 2015; Williams et al., 2018). Recently, NLI has also witnessed corpus-creation efforts in other languages such as OCNLI (Hu et al., 2020), KorNLI (Ham et al., 2020), and IndoNLI (Mahendra et al., 2021). Quyen et al. (2022) proposed the Vietnamese-English NLI task[3] with three labels: agree, disagree, and neutral.

To contribute to the progress of NLP research for Vietnamese, we introduce a high-quality, open-domain corpus for Vietnamese NLI. Inspired by the success of NLI corpora (Bowman et al., 2015; Hu et al., 2020; Mahendra et al., 2021), we follow a strict annotation process and design the guidelines specific to Vietnamese characteristics to make the corpus realistic and high-quality. However, SNLI (Bowman et al., 2015) uses image captions as the main data resource, which are often short, simple texts and limited in linguistic phenomena. Therefore, we use a practical resource with various topics to capture diverse inferences of the Vietnamese NLI task. The premises in our corpus are sentences extracted from 800 news articles on 13 different topics.

In addition, most of the previous corpora require annotators to create only one hypothesis sentence for each premise, for example, SNLI (Bowman et al., 2015), MultiNLI (Williams et al., 2018), and XNLI (Conneau et al., 2018). However, in reality, human reasoning is very diverse in many semantic ways from a given sentence. Hence, we asked annotators to generate two hypothesis sentences for a premise sentence to capture many layers of semantic inference in our corpus. A Similar approach has been implemented in the OCNLI (Hu et al., 2020) and IndoNLI (Mahendra et al., 2021) corpora.

[1] UIT@NLP Group: https://nlp.uit.edu.vn/

[2] https://www.worldometers.info/world-population/vietnam-population/

[3] https://vlsp.org.vn/vlsp2021/eval/nli

Proceedings of the 29th International Conference on Computational Linguistics, pages 3858–3872
October 12–17, 2022.

ViNLI not only has three inference labels as in most previous corpora but also has one more label. The OTHER label is added to separate the inference types, which is different from the meaning of the NEUTRAL label because its purpose is to distinguish pairs of sentences that are unrelated in terms of semantic information, such as events, subjects, and objects. Table 1 shows several samples of Vietnamese NLI.

ViNLI is created and annotated by Vietnamese native speakers with solid linguistic backgrounds. Annotators are trained carefully to familiarize themselves with the corpus creation guidelines following a strict training process.

To evaluate the challenge of the corpus to models, we employ three deep neural networks, including CBoW (Mikolov et al., 2013), BiLSTM (Hochreiter and Schmidhuber, 1997), and ESIM (Chen et al., 2017). We also evaluate the SOTA pre-trained language models: BERT (Devlin et al., 2019), XLM-R (Conneau et al., 2020), and PhoBERT (Nguyen and Nguyen, 2020), which have achieved impressive performances on different NLP tasks.

Contributions of this study are as follows: (1) We introduce the ViNLI corpus, an open-domain, high-quality corpus consisting of over 30,376 human-annotated sentence pairs for evaluating the NLI task. (2) We conduct experiments on NLI models including neural network-based and pre-trained transformer-based models. (3) We analyze the corpus and the experimental results in different linguistic aspects to gain more insights into Vietnamese NLI.

2 Related Work

In this section, we review existing corpora and SOTA models for natural language inference.

2.1 Related Corpora

Early NLI corpora were created mainly for the task of Recognizing Textual Entailment (RTE) (Dagan et al., 2005; Toledo et al., 2012). These human-annotated corpora have contributed to evaluating statistical and logical NLI models. However, the main limitation of these corpora is the small size (less than a few thousand samples), which limits the ability to assess neural network-based models. For example, the SICK (Sentences Involving Compositional Knowledge) corpus (Marelli et al., 2014a) was used for the SemEval 2014 task with only 4,500 training samples. To overcome this limitation, the SICK corpus (Marelli et al., 2014b) was increased in scale to 10K samples.

Since 2015, many NLI benchmark corpora have been created to evaluate the effectiveness of machine learning models. In particular, Bowman et al. (2015) introduced SNLI, a first, large-scale, human-annotated corpus containing 570K English samples for evaluating NLI models. Then, a series of other large-scale NLI corpora appeared: STS-B (Cer et al., 2017) and QQP (Chen et al., 2018) for English with sizes of 8.5K and 404K samples, respectively. 2018 witnessed the release of two large-scale corpora, MultiNLI (Williams et al., 2018) comprising 433K samples, and XNLI (Conneau et al., 2018) with more than 112K samples. While the MultiNLI corpus was built for English, the XNLI was translated into 15 different languages. Besides, with the rapid growth of NLI research for English, NLP research communities witnessed the emergence of corpora for other languages, such as OCNLI (Hu et al., 2020) for Chinese, SICK-NL (Wijnholds and Moortgat, 2021) for Dutch, KorNLI (Ham et al., 2020) for Korean, IndoNLI (Mahendra et al., 2021) for Indonesian, NLI En-Hi (Khanuja et al., 2020) for Hindi-English, and FarsTail (Amirkhani et al., 2020) for Persian. Recently, the Adversarial NLI corpus was introduced by Nie et al. (2020), a data collection via human-and-model-in-the-loop training, which has brought new challenges to SOTA NLI models.

Quyen et al. (2022) introduced the bilingual (Vietnamese-English) NLI corpus[4] annotated with three labels: agree, disagree, and neutral, including approximately 16,200 sentence pairs in the medical domain. An open-domain, large-scale, high-quality corpus similar to SNLI or MultiNLI is necessary for Vietnamese NLI. Following corpus-development efforts, we create the ViNLI corpus, a high quality resource, for developing Vietnamese NLI models, which can improve other NLP tasks: machine translation, question answering, and text summarization.

2.2 Related Models

NLP has witnessed a rapid growth of large-scale corpora and deep learning models for NLI. Besides traditional machine learning models such as Skip-gram, CBOW (Mikolov et al., 2013), deep learning models such as RNN (Elman, 1990), Bi-RNN

[4]https://vlsp.org.vn/vlsp2021/eval/nli

Premise	Majority label All Labels Topic	Hypothesis
Hai cặp nam nữ bị cảnh sát bắt quả tang phê ma tuý nhảy múa trong tiếng nhạc công suất lớn ở căn hộ chung cư. *(Two male and female couples were caught by the police with narcotics and dancing to loud music in the apartment.)*	**Entailment** E E E E E Law	Có tổng cộng bốn người bị công an bắt giữ vì có hành vi sử dụng chất kích thích trái phép. *(A total of four people were arrested by the police for using illegal drugs.)*
Theo kế hoạch, Proace City Electric sẽ bán ra ở châu Âu từ cuối năm 2021. *(As planned, Proace City Electric will be sold in Europe from the end of 2021.)*	**Neutral** N N N N N Vehicles	Thị trường châu Âu vô cùng ưa chuộng dòng xe Proace City Electric. *(The European market extremely favors the Proace City Electric car.)*
Tương tự, đa số nhà đầu tư cá nhân cũng dự đoán giá vàng tăng. *(Similarly, the majority of individual investors are also predicting an increase in the price of gold.)*	**Contradiction** C C C C C Business	Giá vàng ngày càng giảm là điều đáng lo ngại được dự đoán bởi các nhà đầu tư cá nhân. *(The falling gold price is worrisome, which is predicted by individual investors.)*
Cổ phiếu UPS tăng hơn 10% khi lợi nhuận vượt dự báo của Wall Street. *(UPS shares rose more than 10% as earnings beat Wall Street forecasts.)*	**Other** O O O O O Business	NFT là một đơn vị dữ liệu trên sổ cái kỹ thuật số được gọi là blockchain. *(NFT is a data unit on a digital ledger called the blockchain.)*

Table 1: Several examples extracted from ViNLI with topic labels, gold inference labels and the four validation inference labels (E: Entailment, C: Contradiction, N: Neutral, and O: Other).

(Schuster and Paliwal, 1997), BiLSTM (Hochreiter and Schmidhuber, 1997), Dr-BiLSTM (Ghaeini et al., 2018), ESIM (Chen et al., 2017) have achieved positive results on NLI corpora. Recently, transformer-based models like BERT (Devlin et al., 2019) achieved significant progress in performance on various NLP tasks, including NLI on many well-known corpora, SNLI (Bowman et al., 2015) and MultiNLI (Liu et al., 2019a). Besides, the variants of BERT such as RoBERTa (Liu et al., 2019b), XLNet (Yang et al., 2019), and XLM-R (Conneau et al., 2020) also obtained outstanding results on the following corpora: MultiNLI (Williams et al., 2018), XNLI (Conneau et al., 2018), QQP (Chen et al., 2018), STS-B (Cer et al., 2017). However, the models have not been explored for Vietnamese NLI.

3 Corpus Creation

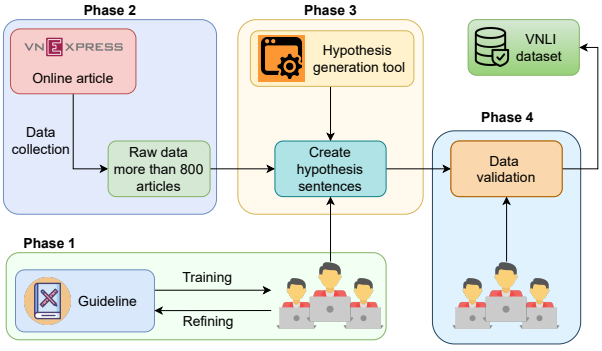

Figure 1: The process of corpus creation.

We build the ViNLI corpus following a strict process for quality control (see Figure 1). This process includes four primary phases: (3.1) annotator recruitment and training, (3.2) premise selection, (3.3) hypothesis generation, and (3.4) data validation. To obtain in-depth insights into the characteristics of the corpus, we analyze the corpus in terms of different linguistic aspects (see Section 3.5).

3.1 Annotator Recruitment and Training

Twenty-nine annotators with strong linguistic backgrounds contributed to the creation of premise-hypothesis pairs for the corpus. Figure 2 depicts our process of annotator training. The annotators must undergo this strict training phase before taking part in the official annotation process. First, the annotators are trained to familiarize themselves with the corpus-creation guidelines (see Section 3.3). Next, each annotator is required to create hypothesis sentences for the annotator-training set that contains 100 premise sentences. The labels (ENTAILMENT, CONTRADICTION, NEUTRAL, and OTHER) of their premise-hypothesis pairs on the set are masked. Two of the Twenty-eight other annotators are asked to give the labels for the same sentences. If the proportion of the labels agreed upon by the three annotators is over 0.95, the annotator is selected to participate in the official annotation process. Otherwise, that annotator needs to learn from their annotation mistakes and goes through the training phase again with another annotator-training set. Besides, we also discuss the annotation disagreements and identify complicated examples to refine the corpus-creation

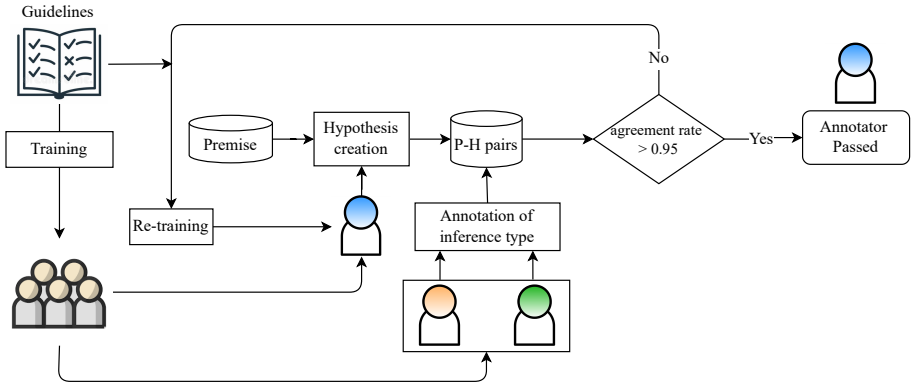

Figure 2: The strict process of training the annotators before creating our corpus.

guidelines.

3.2 Premise Selection

To capture the natural linguistic phenomena of Vietnamese NLI for news texts, the premises in our corpus ViNLI were extracted from 800 public articles on the reputable Vietnamese online newspaper VnExpress[5] with thirteen topics, shown in Table 9 in Appendix A. We selected articles with lengths of 3 to 5 paragraphs and chose the topic sentences of each paragraph as premise sentences. The reason we choose the topic sentence is because it clearly describes the main content of the whole paragraph and thus, is the most crucial sentence in a paragraph. As a result, each news article provides us with 3 to 5 premises.

3.3 Hypothesis Generation

We design four labels (ENTAILMENT, CONTRADICTION, NEUTRAL, and OTHER) instead of three (ENTAILMENT, CONTRADICTION, and NEUTRAL) as in SNLI, MultiNLI, or OCNLI datasets (Bowman et al., 2015; Williams et al., 2018; Hu et al., 2020). Because people also encounter pairs of sentences unrelated to each other in reality, and with only three labels is not possible to distinguish such cases. We ask annotators to create hypothesis sentences for the four following labels.

- **ENTAILMENT**: Create a hypothesis sentence that is definitely true to the content or situation of the premise sentence.
- **CONTRADICTION**: Create a hypothesis sentence that is definitely false with the content or situation of the premise sentence.
- **NEUTRAL**: Create a hypothesis sentence that might be true with the content or situation of the premise sentence.
- **OTHER**: Create a hypothesis sentence that is entirely unrelated to the content or situation of the premise sentence.

The OTHER hypothesis type is different from the NEUTRAL one. To create a NEUTRAL hypothesis, the annotators must rely on events, subjects, and objects in the premises. Along with this, annotators make the hypothesis that might be true with the content or situation of the premise. However, the OTHER hypothesis refers to an entirely different situation in the premise. Particularly, there is no connection of events, subjects, and objects between the premise and hypothesis.

Double Hypothesis: For each premise, annotators are required to create eight hypothesis sentences, two for each inference label (ENTAILMENT, CONTRADICTION, NEUTRAL, and OTHER). This strategy is similar to the multi-hypothesis strategy approached in the OCNLI and IndoNLI data creation protocol[6]. This data collection strategy requires annotators to mine the information in the premise sentence in many different semantic ways. In other words, the hypothesis sentence of each label has diverse content when the annotators are interested in many semantic aspects of the premises.

Besides, annotators were paid roughly 0.022 USD per premise-hypothesis pair. They must generate hypotheses based on the following guidelines:

[5] https://vnexpress.net/

[6] In OCNLI, three hypothesis sentences per label are created for each premise, resulting in a total of nine hypotheses. In IndoNLI, two hypothesis sentences per label are created for each premise, resulting in a total of six hypotheses

(1) Each premise has eight hypotheses (two hypotheses for each label). (2) Annotators are encouraged to write hypotheses in their own words without copying words or phrases from the premise. (3) Annotators may apply our general data-generation rules to create hypotheses. This rule set includes eight rules to generate CONTRADICTION sentences and eleven rules to create ENTAILMENT sentences, which are shown in Table 2 and Table 3 (see examples in Table 10 and Table 11 in Appendix D), respectively.

No.	Rule	Ratio
1	Use negative words (no, not, never, nothing, hardly, etc.)	22%
2	Replace words with antonyms	37%
3	Opposite of quantity	6%
4	Opposite of time	11%
5	Create a sentence that has the opposite meaning of a presupposition	11%
6	Wrong reasoning about an object (House, car, river, sea, person, etc.)	18%
7	Wrong reasoning about an event	27%
8	Others	4%

Table 2: Data-generation rules for creating CONTRADICTION hypothesis sentences.

No.	Rule	Ratio
1	Change active sentences into passive sentences and vice versa.	47%
2	Replace words with synonyms.	75%
3	Add or remove modifiers that do not radically alter the meaning of the sentence.	73%
4	Replace Named Entities with a word that stands for the class.	12%
5	Turn nouns into relative clauses	6%
6	Turn the object into relative clauses	7%
7	Turn adjectives into relative clauses	2%
8	Replace quantifiers with others that have a similar meaning.	13%
9	Create a presupposition sentence	8%
10	Create conditional sentences	2%
11	Others	2%

Table 3: Data-generation rules for creating ENTAILMENT hypothesis sentences.

3.4 Data Validation

To ensure the quality of annotating inference labels for premise-hypothesis pairs, we performed a round of data validation for the ViNLI corpus (see Table 9 in Appendix A). Each premise-hypothesis pair in the development and test dataset is annotated with inference labels by five different annotators. Annotators participating in the validation phase are those who joined in the hypothesis generation phase. The annotators who give inference labels must be different from the person who generate hypothesis in the hypothesis generation phase. Each premise-hypothesis pair is paid 0.013 USD.

We choose the final gold label for each premise-hypothesis pair by majority vote. Similar to the previous corpora (SNLI, MultiNLI, and OCNLI), if not at least three of five labels are the same for a pair, the gold labels are marked as '-'. And then, those pairs are either removed from the corpus or not used during model training and evaluation. The results of the validation phase are shown in Table 4. The statistics show 99.4% of sentence pairs receiving three or more identical labels, higher than the validation results of the well-known corpora such as SNLI, MultiNLI, and OCNLI, illustrating that the ViNLI has high quality and reliability.

3.5 Corpus Analysis

Before conducting corpus analysis, we divided ViNLI randomly into three sets: 80% for a training set (Train), 10% for a development set (Dev), and 10% for a test set (Test). Our corpus statistics are presented in Table 9 (in Appendix A).

The distribution of premise-hypothesis pairs and the average length of premise and hypothesis sentences (words) are illustrated in Table 9 (in Appendix A). We intentionally distribute the premise-hypothesis pairs of each topic evenly distributed on the Dev and Test sets. This division makes it possible to evaluate models fairly without bias toward any topic. In addition, the average length distribution of premise and hypothesis sentences in the Train, Dev, and Test sets is similar, with 24.5 words and 18.1 words for premise and hypothesis, respectively.

Length Distribution: The distribution of the premise and hypothesis sentences according to their length is shown in Figure 3 (in Appendix B). This Figure shows the same distribution as in Table 9. Most hypothesis sentences are shorter than premise sentences, which is similar to SNLI (Bowman et al., 2015). However, this indicates that the length of hypothesis sentences is shorter, but they still guarantee clear representations of their semantic reasoning from the sentence premises. The shortest and longest lengths to generate a hypothesis are 4 and 68 words, respectively. The number of hypothesis sentences with 10-23 word lengths occupies

Statistic	SNLI*	MultiNLI*	XNLI*	OCNLI†	IndoNLI§	ViNLI
Language	English	English	15 languages	Chinese	Indonesian	Vietnamese
Text genre	Image captions	Multi genre	Multi genre	Multi genre	Multi genre	Newswire
#pairs in total	570,152	432,702	7,500	56,525	17,736	30,376
#pairs relabeled	56,951	40,000	7,500	9,913	7,497	6,000
% relabeled	10.0%	9.2%	100%	17.5%	42.3%	19.8%
Pair w/unanimous gold label	58.3%	58.2%	nan	60.1%	nan	77.9%
4+ labels agree	nan	nan	nan	82.5	nan	91.5%
3+ labels agree	**98.0%**	**98.2%**	**93.0%**	**98.6%**	**98.6%**	**99.4%**
Individual label = gold label	89.0%	88.7%	nan	87.5%	90.0%	94.1%
Individual label = author's label	85.8%	85.2%	nan	80.8%	87.6%	91.1%
Gold label = author's label	91.2%	92.6%	nan	89.3%	92.3	96.4%
Gold label ≠ author's label	6.8%	5.6%	nan	9.3%	6.3	3.6%
No gold label (no 3 labels match)	2.0%	1.8%	nan	1.4%	1.4%	0.6%

Table 4: Agreement result of the validation phase in ViNLI compared with other corpora. *The numbers of SNLI, MultiNLI, XNLI corpora are extracted from the scientific papers (Bowman et al., 2015; Williams et al., 2018; Conneau et al., 2018). For XNLI, the number is calculated on a subset of Dev and Test in English. †For OCNLI, the number was calculated from Hu et al. (2020) by averaging 4 different protocols. §For IndoNLI, the agreement is calculated from Mahendra et al. (2021) by averaging 2 different protocols.

the largest proportion in our corpus.

Word Overlap: Taking the motivation from IndoNLI (Mahendra et al., 2021), we calculated the word overlap between the premise and hypothesis sentences in ViNLI. Higher word overlap rates can help predict the inference labels more correctly, which has been illustrated in the study of McCoy et al. (2019). Particularly, we use the Jaccard to calculate the unordered word overlap rate of premise-hypothesis pairs and the LCS index (the Longest Common Sub-sequence) to observe the level of word overlap in order. Before calculating the Jaccard and LCS, we used the VnCoreNLP toolkit (Vu et al., 2018) to perform word segmentation for Vietnamese, as shown in Table 5. With the Jaccard, the label ENTAILMENT has the highest rate of word overlap compared to the other labels, while this ratio is very low for the OTHER label. This is understandable because the hypothesis sentences of the OTHER label in ViNLI are created with content unrelated to its premise sentence. The word overlap rate in the order of premise and hypothesis in the ENTAILMENT label is also the highest, and the OTHER label is the lowest when compared with the CONTRADICTION and NEUTRAL labels. Compared with the IndoNLI corpus created by the lay annotators (Mahendra et al., 2021), the Jaccard and LCS are significantly lower, which can make ViNLI more interesting and challenging for evaluating NLI models.

New Word Rate: To evaluate word diversity in the hypothesis, we measure the new word rate, which is the proportion of hypothesis words not present in the premise. To detect Vietnamese words, we use the word segmentation tool VnCoreNLP (Vu et al., 2018). In Table 5, the new word rate in the ENTAILMENT hypotheses is the lowest at 46.59%. The word diversities of the CONTRADICTION (53.96%), NEUTRAL (61.79%), and OTHER (85.93%) labels are higher than that of the ENTAILMENT label. Compared with the IndoNLI corpus (Mahendra et al., 2021), the new word rate is remarkably higher, making ViNLI more diverse words to challenge NLI models.

In addition, we further analyze the tendency of using part-of-speech (POS) of the new words which annotators used to write hypotheses based on premises. Table 5 shows that annotators use nouns and verbs the most to create hypotheses. Before performing this statistic, we used PhoNLP (Nguyen and Nguyen, 2021) to identify the POS of words.

Data-Generation Rules Analysis: To understand the linguistic behaviors of annotators in creating ViNLI, we analyze the data-generation rules which annotators use to generate hypotheses. We randomly selected 100 ENTAILMENT premise-hypothesis pairs and 100 CONTRADICTION premise-hypothesis pairs in the corpus for analysis. For the CONTRADICTION label (see Table 2), the annotators use the "replace words with antonyms" rule with 37%, whereas the "opposite of quantity" rule is the lowest with 6%. For the ENTAILMENT label (see Table 3), "replace words with synonyms" and "add or remove modifiers that do not radically alter the meaning of the sentence" are the two most common rules used to generate

Label	Jaccard (%)	LCS	New word rate (%)	Part-Of-Speech (%)					
				Noun	Verb	Adjective	Preposition	Adjunct	Other
Entailment	29.88	52.90	46.59	31.45	24.97	6.67	8.39	8.71	19.81
Contradiction	23.30	48.90	53.96	30.79	23.53	7.40	7.61	11.27	19.40
Neutral	20.19	50.34	61.79	33.42	22.89	8.02	8.59	8.62	18.46
Other	6.18	45.53	85.93	42.16	21.91	7.73	8.02	4.87	15.31

Table 5: Word overlap between premise and hypothesis sentences.

hypotheses with 75% and 73%, respectively. While "create conditional sentences" rule is the least common to create hypotheses sentences, with only 2%. "Others" only accounts for a small part of our data.

During data generation, annotators may use one rule or more to generate a hypothesis. Our analysis on using rules for hypothesis creation (see Figure 5 in Appendix E) found that approximately two-thirds (66%) of the hypothesis sentences of the CONTRADICTION label are created by one rule, while data generation with two rules is about a third (32%). Surprisingly, very few annotators use more than two rules (only 2%) to generate hypotheses for the label CONTRADICTION. Unlike the CONTRADICTION label, the hypothesis sentences of the ENTAILMENT label (see Figure 6 in Appendix E) are usually created by combining two to three rules with 86%. Meanwhile, generating hypotheses using one rule or more than three rules for the ENTAILMENT label is only 14%.

4 Empirical Evaluation

4.1 Baseline Models and Settings

To evaluate the difficulty of ViNLI, we experiment with simple models (Random Guess and CBoW (Mikolov et al., 2013)) and more complex models (ESIM (Chen et al., 2017), BiLSTM (Hochreiter and Schmidhuber, 1997), PhoBERT (Nguyen and Nguyen, 2020), mBERT (Devlin et al., 2019), and XLM-R (Conneau et al., 2020)) as baseline models.

mBERT and XLM-R are multilingual language models pre-trained on multilingual documents, including Vietnamese. PhoBERT is a Pre-trained language model for Vietnamese that uses RoBERTa architecture with 135M parameters for the base version and 370M for the large version.

All baseline models are trained using Adam optimal function (Kingma and Ba, 2015) and on Tesla P100-PCIE-16GB of Google Colab[7]. In addition to identifying word boundaries, white space is also used to separate syllables that constitute words in Vietnamese texts. Models have different inputs, which can be word-based or syllable-based representations. Particularly, we implement the CBoW, ESIM, and BiLSTM models with the pre-trained embedding PhoW2V (Nguyen et al., 2020a) for Vietnamese. We experiment with two 300-dimensional versions of PhoW2V, including the syllable and word levels. When using PhoW2V with word-level, we use the VnCoreNLP toolkit (Vu et al., 2018) for word segmentation in Vietnamese.

The hyper-parameters of CBoW, ESIM, BiLSTM models are set up as follows: *learning_rate = 0.001, batch_size = 16, sequence_lenght = 80, epochs = 10*. To train transformer models like BERT, XLM-R, and PhoBERT, we used Hugging Face's Transformers library[8]. Besides, we set the hyper-parameters as follows *learning_rate = 1e-5, epochs = 10, batch_size = 16*.

We conduct experiments on two label sets: a three-label set (ENTAILMENT, CONTRADICTION, and NEUTRAL) and a four-label set (ENTAILMENT, CONTRADICTION, NEUTRAL, and OTHER).

4.2 Evaluation Metrics

Following SNLI (Bowman et al., 2015), we use accuracy as the primary evaluation metric. We also calculate F1-score (macro average) as the second evaluation metric to obtain more insights.

4.3 Human Performance

Following Hu et al. (2020), we hired five native Vietnamese speakers to annotate a subset of 300 samples (Test_{300}) extracted randomly from the Test set. These people did not know anything about the NLI task before, and we trained them on task definition and the meaning of inference labels to choose a label for each premise-hypothesis pair. The majority voting of five labels chooses the final label for each pair. Human performances are achieved with accuracy-based performances of

[7]https://colab.research.google.com/

[8]https://huggingface.co/docs/transformers/index

95.34% (for 03 labels: ENTAILMENT, CONTRADICTION, NEUTRAL) and 95.78% (for 04 labels: ENTAILMENT, CONTRADICTION, NEUTRAL, OTHER).

4.4 Experimental Results

Table 6 presents the performances of the baseline models on the Dev and Test sets. Overall, transformer-based models (mBERT, PhoBERT, and XLM-R) outperform others (Random Guess, CBOW, ESIM, and BiLSTM). XLM-R$_{Large}$ achieves the best results in both experiments with different label sets: three labels (ENTAILMENT, CONTRADICTION, NEUTRAL) and four labels (ENTAILMENT, CONTRADICTION, NEUTRAL, OTHER). On the three-label corpus, XLM-R$_{Large}$ achieved the highest accuracy-based performances on Dev and Test sets, with 83.02% and 81.36%, respectively. Besides, PhoBERT$_{Large}$ also achieves impressive results with 75.93% accuracy on the Test set.

On the corpus with four labels, the performances of the models tend to be quite similar to the three labels experiments. The performance of the XLM-R$_{Large}$ model also has the highest accuracy, with 86.77% on the Dev set and 85.99% on the Test set. The best performance of the syllable-level model (XLM-R) is significantly higher than the best performance of the word-level model (PhoBERT), similar to the Vietnamese MRC shared task (Nguyen et al., 2022). The accuracy and F1 achieve roughly the same results due to the experiments on the balanced corpus. Furthermore, most of the model performances on the four-label experiments are higher than those on the three-label experiments because the OTHER label is easier to recognize than other labels (see Table 7).

XLM-R$_{Large}$ also achieves the best accuracy on the Test$_{300}$ set. However, when compared with human performance, this model is still significantly lower, by 14.20% on Test$_{300}$ on three labels and 6.93% on Test$_{300}$ on four labels.

5 Result Analysis

To gain more insights, we analyze the two best-performance models, including XLM-R$_{Large}$ and PhoBERT$_{Large}$ on different linguistic aspects. In this section, XLM-R and PhoBERT stand for XLM-R$_{Large}$ and PhoBERT$_{Large}$, respectively.

How do inference labels affect the performance?

Table 7 shows the analysis of accuracy on each label in the Dev set on three labels and the Dev set on four labels of the two best-performance models (XLM-R and PhoBERT). Both XLM-R and PhoBERT perform very well on the OTHER label with more than 97% accuracy. When adding the OTHER label, the accuracy-based performances of XLM-R on CONTRADICTION and NEUTRAL are improved; however, the accuracy of ENTAILMENT is decreased from 89.31% to 86.33%.

Labels	Three-label Dev		Four-label Dev	
	PhoBERT	XLM-R	PhoBERT	XLM-R
Entailment	77.94	89.31	76.45	86.33
Contradiction	76.57	80.76	71.46	82.98
Neutral	77.53	79.12	77.66	80.45
Other	-	-	97.35	97.34

Table 7: Model performance per label in ViNLI.

How do new words affect the performance? We aim to analyze the effect of the new words in hypothesis sentences on the model accuracy. Figure 4 (in Appendix C) shows the accuracy of PhoBERT and XLM-R models according to the new word rate in the three-labels and four-labels dev set. Figure 4a shows that the accuracy of the PhoBERT model significantly decreases from around 84% to about 67% as the new word rate in hypothesis sentences increases from less than 20% to more than 80%. In contrast, the accuracy of the XLM-R model is relatively stable.

With the results of the four-label Dev set in Figure 4b, the accuracy of the models is quite similar to the trend of the three-label dev set when the new word rate is less than or equal to 60%. However, the performance of the models with the new word rate of more than 60% on the four-label Dev set is higher than that of the three-label Dev set since all pairs of OTHER labels have the highest new word rate (see Table 5). Moreover, the model performances on OTHER achieve the most (see Table 7) compared to other labels.

How do data-generation rules affect the performance? Table 8 analyzes the influence of using data-generation rules that the annotators generate the hypotheses on the model performance. We analyze our experimental results of PhoBERT and XLM-R on data-generation rules (as described in Subsection 3.5). Our experiments show that the ENTAILMENT hypotheses with more than two rules cause the performance of the models to be lower when these hypotheses are generated with only one rule. With the CONTRADICTION hy-

Model		Three Labels						Four Labels					
		Dev		Test		Test$_{300}$		Dev		Test		Test$_{300}$	
		Acc	**F1**	**Acc**	**F1**	**Acc**	**F1**	**Acc**	**F1**	**Acc**	**F1**	**Acc**	**F1**
Random Guess		33.36	32.49	32.51	33.01	34.28	34.27	25.47	24.27	24.51	24.85	25.19	25.17
Syllable	CBoW	45.54	45.13	44.96	44.62	43.86	43.43	46.29	45.73	45.97	45.46	43.58	43.88
	ESIM	48.55	48.24	47.43	46.76	46.92	46.41	48.58	48.45	47.44	47.17	43.58	42.99
	BiLSTM	48.07	48.10	46.42	46.35	46.92	46.84	48.89	48.36	48.55	48.06	48.31	48.07
	mBERT	67.41	67.46	64.84	64.83	64.91	64.85	73.91	73.83	73.45	73.62	75.34	75.78
	XLM-R$_{Base}$	72.02	71.99	71.59	71.51	71.93	71.55	76.97	76.93	76.83	77.01	77.71	78.31
	XLM-R$_{Large}$	**83.02**	**82.98**	**81.36**	**81.31**	**81.14**	**81.12**	**86.77**	**86.76**	**85.99**	**86.10**	**88.85**	**89.13**
Word	CBoW	49.05	48.64	45.80	45.41	42.11	41.37	54.63	54.48	53.12	52.87	51.01	50.60
	ESIM	49.84	49.75	48.18	48.12	40.79	40.25	48.68	47.99	48.37	47.83	46.95	46.14
	BiLSTM	48.91	48.71	46.77	46.59	43.42	42.52	50.48	49.89	49.78	49.22	49.32	48.31
	PhoBERT$_{Base}$	75.07	75.08	72.87	72.79	71.05	70.31	79.79	79.75	78.00	78.05	77.70	77.98
	PhoBERT$_{Large}$	**77.33**	**77.34**	**75.93**	**75.87**	**77.19**	**77.19**	**80.72**	**80.72**	**80.67**	**80.69**	**80.74**	**81.11**
Human performance		-	-	-	-	**95.34**	**95.33**	-	-	-	-	**95.78**	**95.79**

Table 6: Human and machine performances on the Dev and Test sets of our corpus ViNLI.

potheses, the models have more difficulty than those generated by the annotator with only one rule, whereas PhoBERT and XLM-R achieve higher accuracy if the hypotheses are generated from two or more rules.

Label	**#Rule**	**PhoBERT**	**XLM-R**
Entailment	1-2	81.82	89.09
	more than 2	77.78	84.44
Contradiction	1	71.21	81.81
	more than 1	76.47	91.18

Table 8: Effects of data-generation rules on the models.

How do multiple topics affect the performance? To observe the impact of multiple topics (open-domain) in ViNLI, we calculate the accuracy of the two highest-performance models (PhoBERT and XLM-R models) in terms of 13 different topics. Table 12 (in Appendix F) shows that XLM-R consistently outperformed PhoBERT on all topics. The models achieve different results on various topics. While the models reach the best performances in Tourism and Entertainment, Business and World are the most challenging for the models.

6 Conclusion and Future Work

In this paper, we introduced ViNLI, an open-domain, high-quality corpus for evaluating Vietnamese NLI models. By a strict annotation scheme with high annotator agreements, 30,376 premise-hypothesis pairs of the corpus were annotated by humans with solid linguistic backgrounds, which is the largest Vietnamese NLI corpus to date. The performances of the two powerful models (XLM-R and PhoBERT) illustrate that our corpus is challenging for the pre-trained language models in Vietnamese, the best of which underperforms humans by over 14% (on three labels). ViNLI is available freely for research purposes in developing Vietnamese NLU models.

Taking advantages of state-of-the-art models on large-scale, high-quality NLI corpora (SNLI, MultiNLI, XNLI, OCNLI, KorNLI, and IndoNLI), we hope that our corpus will accelerate progress on Vietnamese NLI and other NLP tasks. Based on the findings of our work, we continue to enhance the quality and quantity of the corpus with different data sources and expand our corpus with adversarial samples (Kang et al., 2018; Nie et al., 2020). Moreover, future studies will concentrate on exploiting Vietnamese models using BERTology (Rogers et al., 2020) (e.g., SBERT (Reimers and Gurevych, 2019)) and improving NLP applications such as Vietnamese machine reading comprehension models and retriever-reader question answering systems.

Acknowledgements

This research is funded by Vietnam National University HoChiMinh City (VNU-HCM) under grant number DS2022-26-01. Tin Van Huynh was funded by Vingroup JSC and supported by the Master, PhD Scholarship Programme of Vingroup Innovation Foundation (VINIF), Institute of Big Data, code VINIF.2021.ThS.49. We would like to thank anonymous reviewers from ACL Rolling Review and COLING 2022, which their comments help us improve the quality of our paper. In addition, we would like to thank our annotators for their cooperation.

References

Hossein Amirkhani, Mohammad Azari Jafari, Azadeh Amirak, Zohreh Pourjafari, Soroush Faridan Jahromi, and Zeinab Kouhkan. 2020. Farstail: A persian natural language inference dataset. *arXiv preprint arXiv:2009.08820*.

Samuel Bowman, Gabor Angeli, Christopher Potts, and Christopher D Manning. 2015. A large annotated corpus for learning natural language inference. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 632–642.

Daniel Cer, Mona Diab, Eneko Agirre, Inigo Lopez-Gazpio, and Lucia Specia. 2017. Semeval-2017 task 1: Semantic textual similarity-multilingual and cross-lingual focused evaluation. *arXiv preprint arXiv:1708.00055*.

Qian Chen, Xiaodan Zhu, Zhen-Hua Ling, Si Wei, Hui Jiang, and Diana Inkpen. 2017. Enhanced lstm for natural language inference. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1657–1668.

Zihan Chen, Hongbo Zhang, Xiaoji Zhang, and Leqi Zhao. 2018. Quora question pairs. *URL https://www.kaggle. com/c/quora-question-pairs*.

Alexis Conneau, Kartikay Khandelwal, Naman Goyal, Vishrav Chaudhary, Guillaume Wenzek, Francisco Guzmán, Edouard Grave, Myle Ott, Luke Zettlemoyer, and Veselin Stoyanov. 2020. Unsupervised cross-lingual representation learning at scale. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 8440–8451, Online. Association for Computational Linguistics.

Alexis Conneau, Ruty Rinott, Guillaume Lample, Adina Williams, Samuel Bowman, Holger Schwenk, and Veselin Stoyanov. 2018. Xnli: Evaluating cross-lingual sentence representations. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 2475–2485.

Ido Dagan, Oren Glickman, and Bernardo Magnini. 2005. The pascal recognising textual entailment challenge. In *Machine Learning Challenges Workshop*, pages 177–190. Springer.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, Minneapolis, Minnesota. Association for Computational Linguistics.

Long Doan, Nguyen Luong Tran, Thai Hoang, Dat Quoc Nguyen, et al. 2021. Phomt: A high-quality and large-scale benchmark dataset for vietnamese-english machine translation. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 4495–4503.

Jeffrey L Elman. 1990. Finding structure in time. *Cognitive science*, 14(2):179–211.

Reza Ghaeini, Sadid A Hasan, Vivek Datla, Joey Liu, Kathy Lee, Ashequl Qadir, Yuan Ling, Aaditya Prakash, Xiaoli Z Fern, and Oladimeji Farri. 2018. Dr-bilstm: Dependent reading bidirectional lstm for natural language inference. *arXiv preprint arXiv:1802.05577*.

Jiyeon Ham, Yo Joong Choe, Kyubyong Park, Ilji Choi, and Hyungjoon Soh. 2020. Kornli and korsts: New benchmark datasets for korean natural language understanding. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: Findings*, pages 422–430.

Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long short-term memory. *Neural computation*, 9(8):1735–1780.

Hai Hu, Kyle Richardson, Liang Xu, Lu Li, Sandra Kübler, and Lawrence S Moss. 2020. Ocnli: Original chinese natural language inference. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: Findings*, pages 3512–3526.

Dongyeop Kang, Tushar Khot, Ashish Sabharwal, and Eduard Hovy. 2018. Adventure: Adversarial training for textual entailment with knowledge-guided examples. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 2418–2428.

Simran Khanuja, Sandipan Dandapat, Sunayana Sitaram, and Monojit Choudhury. 2020. A new dataset for natural language inference from code-mixed conversations. In *Proceedings of the The 4th Workshop on Computational Approaches to Code Switching*, pages 9–16, Marseille, France. European Language Resources Association.

Diederik P Kingma and Jimmy Ba. 2015. Adam: A method for stochastic optimization. in yoshua bengio and yann lecun editors. In *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*.

Xiaodong Liu, Pengcheng He, Weizhu Chen, and Jianfeng Gao. 2019a. Multi-task deep neural networks for natural language understanding. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 4487–4496.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019b. Roberta: A robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692*.

Rahmad Mahendra, Alham Fikri Aji, Samuel Louvan, Fahrurrozi Rahman, and Clara Vania. 2021. Indonli: A natural language inference dataset for indonesian. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 10511–10527.

M Marelli, Luisa Bentivogli, M Baroni, R Bernardi, Stefano Menini, and R Zamparelli. 2014a. Semeval-2014 task 1: Evaluation of compositional distributional semantic models on full sentences through semantic relatedness and textual entailment.

Marco Marelli, Stefano Menini, Marco Baroni, Luisa Bentivogli, Raffaella Bernardi, Roberto Zamparelli, et al. 2014b. A sick cure for the evaluation of compositional distributional semantic models. In *Lrec*, pages 216–223. Reykjavik.

Tom McCoy, Ellie Pavlick, and Tal Linzen. 2019. Right for the wrong reasons: Diagnosing syntactic heuristics in natural language inference. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 3428–3448.

Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. 2013. Distributed representations of words and phrases and their compositionality. In *Advances in neural information processing systems*, pages 3111–3119.

Anh Tuan Nguyen, Mai Hoang Dao, and Dat Quoc Nguyen. 2020a. A Pilot Study of Text-to-SQL Semantic Parsing for Vietnamese. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 4079–4085.

Dat Quoc Nguyen and Anh Tuan Nguyen. 2020. PhoBERT: Pre-trained language models for Vietnamese. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 1037–1042.

Kiet Nguyen, Vu Nguyen, Anh Nguyen, and Ngan Nguyen. 2020b. A Vietnamese dataset for evaluating machine reading comprehension. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 2595–2605, Barcelona, Spain (Online). International Committee on Computational Linguistics.

Linh The Nguyen and Dat Quoc Nguyen. 2021. PhoNLP: A joint multi-task learning model for Vietnamese part-of-speech tagging, named entity recognition and dependency parsing. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Demonstrations*, pages 1–7.

Van Kiet Nguyen, Son Quoc Tran, Luan Thanh Nguyen, Tin Van Huynh, Son T Luu, and Ngan Luu-Thuy Nguyen. 2022. Vlsp 2021 shared task: Vietnamese machine reading comprehension. *arXiv preprint arXiv:2203.11400*.

Yixin Nie, Adina Williams, Emily Dinan, Mohit Bansal, Jason Weston, and Douwe Kiela. 2020. Adversarial nli: A new benchmark for natural language understanding. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 4885–4901.

Ngo The Quyen, Hoang Tuan Anh, Nguyen Thi Minh Huyen, and Nguyen Lien. 2022. VLSP 2021 - vnNLI Challenge: Vietnamese and English-Vietnamese Textual Entailment. *VNU Journal of Science: Computer Science and Communication Engineering*, 38(1).

Nils Reimers and Iryna Gurevych. 2019. Sentence-bert: Sentence embeddings using siamese bert-networks. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 3982–3992.

Anna Rogers, Olga Kovaleva, and Anna Rumshisky. 2020. A primer in bertology: What we know about how bert works. *Transactions of the Association for Computational Linguistics*, 8:842–866.

M. Schuster and K.K. Paliwal. 1997. Bidirectional recurrent neural networks. *IEEE Transactions on Signal Processing*, 45(11):2673–2681.

Assaf Toledo, Sophia Katrenko, Stavroula Alexandropoulou, Heidi Klockmann, Asher Stern, Ido Dagan, and Yoad Winter. 2012. Semantic annotation for textual entailment recognition. In *Mexican International Conference on Artificial Intelligence*, pages 12–25. Springer.

Thanh Vu, Dat Quoc Nguyen, Dai Quoc Nguyen, Mark Dras, and Mark Johnson. 2018. VnCoreNLP: A Vietnamese natural language processing toolkit. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Demonstrations*, pages 56–60, New Orleans, Louisiana. Association for Computational Linguistics.

Gijs Wijnholds and Michael Moortgat. 2021. Sick-nl: A dataset for dutch natural language inference. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*, pages 1474–1479.

Adina Williams, Nikita Nangia, and Samuel Bowman. 2018. A broad-coverage challenge corpus for sentence understanding through inference. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 1112–1122.

Zhilin Yang, Zihang Dai, Yiming Yang, Jaime Carbonell, Russ R Salakhutdinov, and Quoc V Le. 2019. Xlnet: Generalized autoregressive pretraining for language understanding. *Advances in neural information processing systems*, 32.

A Topic Statistics

Table 9 presents statistics in terms of 13 topics, mean premise length (words)[9] and mean hypothesis length (words)[10].

Topic/Label	Train	Dev	Test	Total
Technology	1,912	232	232	2,376
Tourism	1,896	232	228	2,356
Education	1,936	232	231	2,399
Entertainment	1,640	231	231	2,102
Science	1,792	232	228	2,252
Business	1,616	231	228	2,075
Law	1,680	231	231	2,142
Health	2,048	231	232	2,511
World	2,040	232	229	2,501
Sports	2,088	230	231	2,549
News	1,576	231	231	2,038
Vehicles	2,288	232	228	2,748
Life	1,864	232	231	2,327
Entailment	6,094	739	750	7,583
Contradiction	6,094	764	737	7,595
Neural	6,094	752	777	7,623
Other	6,094	754	727	7,575
Total (pairs)	24,376	3,009	2,991	30,376
MPL (words)	24.5	24.6	24.3	24.5
MHL (words)	18.3	17.9	18.1	18.1

Table 9: ViNLI statistics in terms of different topics, Mean Premise Length (MPL) and Mean Hypothesis Length (MHL).

B Length Distribution

The distribution of the premise and hypothesis sentences according to their length is shown in Figure 3. The length of premise and hypothesis sentences is counted by the number of words (A Vietnamese word consists of one or more syllables). We use the VnCoreNLP toolkit (Vu et al., 2018) for Vietnamese word segmentation.

[9]Mean premise length is the mean average of word-based lengths of premise sentences in Train/Dev/Test sets or ViNLI (total).

[10]Mean hypothesis length is the mean average of word-based lengths of hypothesis sentences in Train/Dev/Test sets or ViNLI (total).

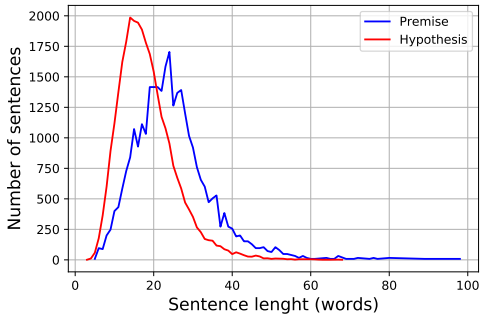

Figure 3: The distribution of sentence length.

C Effect of New Words

To observe the influence of the new word rate on the performance of models, we analyze the accuracy of PhoBERT$_{Large}$ and XLM-R$_{Large}$ models according to the new word rate. The analysis of the three-label and four-label Dev sets is shown in Figure 4.

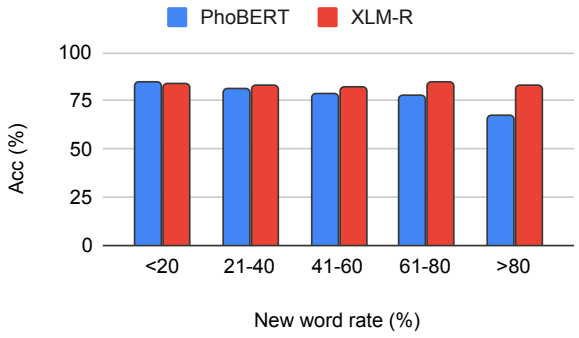

(a) Dev set with three labels.

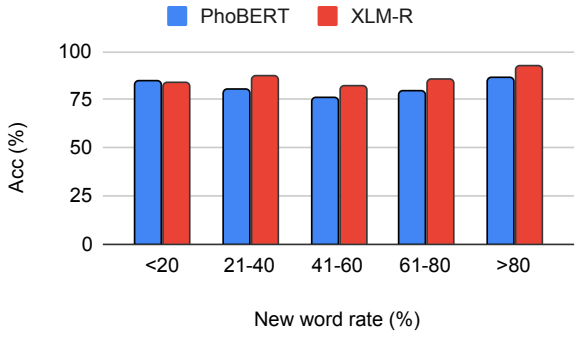

(b) Dev set with four labels.

Figure 4: Model accuracy on the Dev set according to new word rate.

D Data-Generation Rules

To understand the linguistic behaviors of annotators in creating ViNLI, we analyze data-generation rules which annotators use to generate hypothe-

ses. We randomly selected 100 ENTAILMENT premise-hypothesis pairs and 100 CONTRADICTION premise-hypothesis pairs in the corpus for analysis. For the CONTRADICTION label (see Table 2), the annotators use the "replace words with antonyms" rule with 37%, whereas the "opposite of quantity" rule is the lowest with 6%. For the ENTAILMENT label (see Table 3), "replace words with synonyms" and "add or remove modifiers that do not radically alter the meaning of the sentence" are the two most common rules used to generate hypotheses with 75% and 73%, respectively. While "create conditional sentences" rule is the least common to create hypotheses, with only 2%. "Others" only accounts for a small part of our data.

E Rules Combination for Creating Hypotheses

To create more diverse and challenging data, annotators may use one rule or more to generate a hypothesis. Figure 5 and Figure 6 show the proportion of using data-generation rules to create sentences for contradiction and entailment, respectively. Whereas most contradiction hypothesis sentences use one rule, entailment hypothesis sentences are created mainly based on two and three rules. Table 10 and Table 11 present several samples of contradiction and entailment rules for creating premise (P) - hypothesis (H) pairs, respectively.

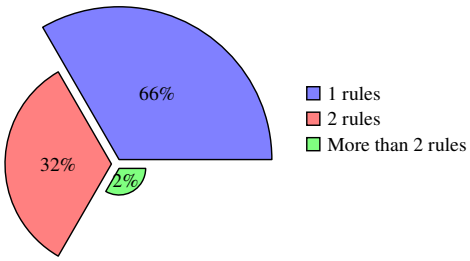

Figure 5: The ratio of combining different rules to create contradiction sentences in ViNLI.

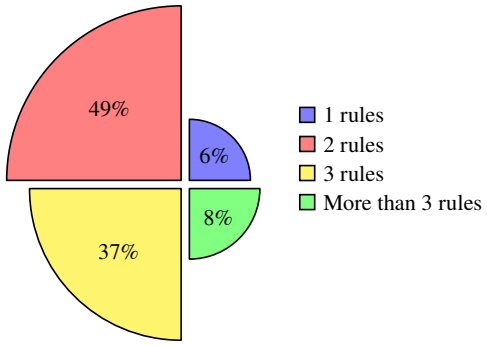

Figure 6: The ratio of combining different rules to create entailment sentences in ViNLI.

F Effects of Multiple Topics

To observe the impact of multiple topics (open-domain) in the ViNLI corpus, we calculate the accuracy of the two highest-performance models (PhoBERT$_{Large}$ and XLM-R$_{Large}$ models) on 13 different topics of ViNLI. The results are shown in Table 12.

Topic	Three-label Dev		Four-label Dev	
	PhoBERT	XLM-R	PhoBERT	XLM-R
Technology	76.88	83.24	77.59	86.64
Tourism	**84.88**	**89.53**	81.47	**91.81**
Education	79.43	81.14	82.76	83.62
Entertainment	81.14	88.57	**83.12**	90.91
Science	73.10	79.53	78.88	85.34
Business	**72.09**	80.23	**75.32**	**80.95**
Law	77.71	81.14	82.25	85.28
Health	80.92	83.82	82.68	88.74
World	72.99	**79.31**	79.31	87.93
Sports	75.86	85.63	83.04	83.91
News	76.30	81.50	82.68	88.31
Vehicles	78.16	83.33	81.90	87.93
Life	75.86	82.18	78.45	86.64

Table 12: Analyzing the model performances on different topics.

Rule	Example	Per.
Use negative words (no, not, never, nothing, hardly, etc.)	**P**: Cơ quan chức năng đã lập biên bản vụ việc. *(Authorities recorded the minutes of the incident.)* **H**: Cơ quan chức năng **không** tiến hành xử lí vụ việc. *(Authorities did **not** process the case.)*	22%
Replace words with antonyms	**P**: AAPP cho biết thêm chính quyền quân sự Myanmar đang **giam** 4.120 người, trong đó có 20 người bị kết án tử hình. *(The AAPP added that Myanmar's military junta is **holding** 4,120 people, of which 20 are sentenced to death.)* **H**: Đã có 4.120 người được chính quyền quân sự quân sự Myanmar **trả tự do**. *(There have been 4,120 people **released** by the military junta of Myanmar.)*	37%
Opposite of quantity	**P**: Suốt cuộc diễu hành kéo dài khoảng **một tiếng**, Tổng thống Bolsonaro và đa số người ủng hộ ông đều không đeo khẩu trang. *(During the parade, which lasted about **an hour**, President Bolsonaro and most of his supporters were not wearing masks.)* **H**: Cuộc diễu hành kéo dài khoảng **10 tiếng**, người tham gia và cả Tổng thống Bolsonaro đều không đeo khẩu trang. *(The parade lasted about **10 hours**; participants and President Bolsonaro were not wearing masks.)*	6%
Opposite of time	**P**: Miss Universe 2020 kéo dài khoảng 12 ngày, chung kết diễn ra tối **16/5** tại Hollywood, bang Florida. *(Miss Universe 2020 lasts about 12 days, and the final will take place on the evening of **May 16** in Hollywood, Florida.)* **H**: Miss Universe 2020 sẽ được tổ chức trong khoảng thời gian **từ 20-25/5** tại Mỹ. *(Miss Universe 2020 will be held **between May 20 and 25** in the US.)*	11%
Create a sentence that has the opposite meaning of a presupposition	**P**: **Giám đốc điều hành Apple, Lisa Jackson**, cho biết khó khăn của việc sử dụng năng lượng sạch. *(**Apple CEO Lisa Jackson** said the difficulty of using clean energy.)* **H**: Apple **chưa thể bổ nhiệm ai cho chức vụ giám đốc điều hành**. *(Apple **has not been able to appoint anyone for the position of CEO**.)*	11%
Wrong reasoning about an object (House, car, river, sea, person, etc.)	**P**: Trong thông báo hôm 24/5, **Honda Việt Nam** công bố chiến dịch thu hồi mẫu xe ga nhập khẩu. *(In an announcement on May 24, **Honda Vietnam** announced a campaign to recall imported scooter models.)* **H**: **Honda Hàn Quốc** thông báo triệu hồi các mẫu xe ga nhập khẩu. *(**Honda Korea** announced the recall of imported scooter models.)*	18%
Wrong reasoning about an event	**P**: Trong lần gặp lại này, **Zverev vượt trội đối thủ** ở giao bóng. *(In this meeting, **Zverev outperformed his opponent** in serving.)* **H**: **Đối thủ có kỹ năng giao bóng vượt xa** Zverev. *(**The opponent's serving skill far exceeded** Zverev.)*	27%
Others	**P**: Phía Tập đoàn Trung Nam cho biết, với 64,9% cổ phần còn lại họ vẫn giữ vai trò quyết định trong điều hành, định hướng phát triển của dự án điện gió Trung Nam. *(Trung Nam Group said that with the remaining 64.9% stake, they still play a decisive role in the management and development orientation of the Trung Nam wind power project.)* **H**: **Do** nắm giữ ít cổ phần **nên** Trung Nam Group mất quyền quyết định đối với các dự án quan trọng. *(**Because of** holding fewer shares, Trung Nam Group lost decision-making power on important projects.)* **Explanation**: Causal, Although clauses, etc., clauses can be used to create the contradiction hypothesis sentence with the premise sentence. This is a case of Others.	4%

Table 10: Examples of contradiction rules for creating premise (**P**) - hypothesis (**H**) pairs. Simply, we only mention one rule to be applied in each example.

Rule	Example	Per.
Change active sentences into passive sentences and vice versa.	**P**: Giá các mặt hàng dầu **đều tăng**. (*Prices of oil commodities **have increased**.*) **H**: Giá xăng dầu **được** tất cả các cửa hàng xăng dầu trên toàn quốc **điều chỉnh tăng lên**. (*Oil prices **are adjusted to increase by** all petrol stations nationwide.*)	47%
Replace words with synonyms.	**P**: Nadal tốn 130 phút để **vượt qua** Sinner 7-5, 6-4. (*Nadal took 130 minutes to **beat** Sinner 7-5, 6-4.*) **H**: Sau hơn hai tiếng, Nadal **chiến thắng** trước Sinner với tỷ số 2-0. (*After more than two hours, Nadal **won** against Sinner with a score of 2-0.*)	75%
Add or remove modifiers that do not radically alter the meaning of the sentence.	**P**: Châu Nhuận Phát **sinh ngày 18/5/1955** trong gia đình nghèo. (*Chau Nhuan Phat **was born on May 18, 1955** in a poor family.*) **H**: Châu Nhuận Phát **là con của một gia đình** có hoàn cảnh khó khăn. (*Chau Nhuan Phat **is the son of a family** with difficult circumstances.*)	73%
Replace Named Entities with a word that stands for the class.	**P**: Hacker rao bán dữ liệu của hơn 533 triệu **tài khoản Facebook**, bao gồm số điện thoại và một số thông tin cá nhân. (*Hacker sells data of more than 533 million **Facebook accounts**, including phone numbers and some personal information.*) **H**: Thông tin cá nhân của hơn 533 triệu **tài khoản mạng xã hội** đã bị Hacker rao bán. (*Personal information of more than 533 million **social network accounts** is sold by Hackers.*)	12%
Turn nouns into relative clauses	**P**: **Tháng 8-9** là thời điểm ốc béo nhất. (***August-September** is the fattest time of snails.*) **H**: **Tháng 8-9 là mùa mà** những người đầu bếp sẽ dễ dàng lựa chọn những con ốc béo và thơm nhất. (***August-September is the season when** chefs will easily choose the fattest and most fragrant snails.*)	6%
Turn the object into relative clauses	**P**: Wernery cho biết lạc đà được tiêm **xác của nCoV** để sản sinh kháng thể. (*Wernery said camels are injected with **the carcass of nCoV** to produce antibodies.*) **H**: Lạc đà được tiêm **xác của nCoV, là dung dịch** có khả năng tạo ra khoáng thể chống lại virus. (*Camels are injected with **the carcass of nCoV, which is a solution** capable of creating antibodies against the virus.*)	7%
Turn adjectives into relative clauses	**P**: Quần đảo Lofoten của Na Uy là một trong những **địa điểm đẹp nhất trên trái đất**. (*Norway's Lofoten Islands are some of **the most beautiful places on earth**.*) **H**: Quần đảo Lofoten của Na Uy là **địa điểm du lịch, nơi được mệnh danh là đẹp nhất trên trái đất**. (*Norway's Lofoten Islands are **tourist destinations that have been dubbed the most beautiful place on earth**.*)	2%
Replace quantifiers with others that have a similar meaning.	**P**: Công an xác minh, giờ ra chơi sáng 13/5, **một nam sinh và một nữ sinh** lớp 9B trong lúc đùa nghịch đã cắn tay nhau. (*Police verified that at recess on the morning of May 13, **a male student and a female student** in class 9B bit each other's hands while frolicking.*) **H**: **Hai học sinh** của lớp 9B đã cắn nhau vào giờ ra chơi. (***Two students** from class 9B were biting each other during break time.*)	13%
Create a presupposition sentence	**P**: Cũng theo Goal, **Marcelo không phải là cầu thủ duy nhất của Real** bất bình với Zidane. (*According to Goal, **Marcelo is not the only player of Real** to be angry with Zidane.*) **H**: **Marcelo đá cho đội tuyển Real Madrid**. (***Marcelo plays for the Real Madrid team**.*)	8%
Create conditional sentences	**P**: Do ảnh hưởng của Covid-19, doanh nghiệp không xuất khẩu được nên khoảng 50.000 tấn hành tím tới kỳ thu hoạch của nông dân xã Vĩnh Châu không có nơi tiêu thụ. (*Due to the impact of Covid-19, businesses could not export, so about 50,000 tons of purple onions until the harvest period of Vinh Chau commune farmers have no place to consume.*) **H**: **Nếu** không bị ảnh hưởng bởi Covid-19, doanh nghiệp **sẽ** xuất khẩu được khoảng 50.000 tấn hành tím. (***If** not affected by Covid-19, the enterprise **will** be able to export about 50,000 tons of purple onions.*)	2%
Others	**P**: Nạn nhân không bị nguy hiểm đến tính mạng nhưng chưa thể làm việc với cơ quan điều tra. (*The victim's life is not in danger, but the victim has not been able to work with the investigative agency.*) **H**: **Mặc dù** không bị nguy hiểm đến tính mạng nhưng nạn nhân vẫn chưa thể làm việc với cơ quan điều tra. (***Although** the victim's life is not in danger, the victim is still unable to work with the investigative agency.*) **Explanation**: Causal, Although clauses, etc., can be used to create the entailment hypothesis sentence with the premise sentence. This is a case of Others.	2%

Table 11: Examples of entailment rules for creating premise (**P**) - hypothesis (**H**) pairs. Simply, we only mention one rule to be applied in each example.

INFERES : A Natural Language Inference Corpus for Spanish Featuring Negation-Based Contrastive and Adversarial Examples

Venelin Kovatchev
School of Information
The University of Texas at Austin
venelin@utexas.edu

Mariona Taulé
Centre de Llenguatge i Computació
Institut de Recerca en Sistemes Complexos
Universitat de Barcelona
mtaule@ub.edu

Abstract

In this paper, we present INFERES - an original corpus for Natural Language Inference (NLI) in European Spanish. We propose, implement, and analyze a variety of corpus-creating strategies utilizing expert linguists and crowd workers. The objectives behind INFERES are to provide high-quality data, and, at the same time to facilitate the systematic evaluation of automated systems. Specifically, we focus on measuring and improving the performance of machine learning systems on negation-based adversarial examples and their ability to generalize across out-of-distribution topics.

We train two transformer models on INFERES (8,055 gold examples) in a variety of scenarios. Our best model obtains 72.8% accuracy, leaving a lot of room for improvement. The "hypothesis-only" baseline performs only 2%-5% higher than majority, indicating much fewer annotation artifacts than prior work. We find that models trained on INFERES generalize very well across topics (both in- and out-of-distribution) and perform moderately well on negation-based adversarial examples.

1 Introduction

In the task of Natural Language Inference (NLI), an automated system has to determine the meaning relation that holds between two texts. The model has to make a three-way choice between *entailment*: a hypothesis (h) is true given a premise (p) (e.g. **1.**); *contradiction*: a hypothesis (h) is false given a premise (p) (e.g. **2.**); or *neutral*: the truth value of the hypothesis (h) cannot be determined solely based on the premise (p) (e.g.: **3.**).

1. *p)* John goes to work every day with a car.
h) John has a job.

2. *p)* John goes to work every day with a car.
h) John takes the bus to go to work.

3. *p)* John goes to work every day with a car.
h) John has a Porsche.

NLI (formerly known as Recognizing Textual Entailment (RTE)) is one of the core tasks in the popular benchmarks for Natural Language Understanding GLUE (Wang et al., 2018) and Super GLUE (Wang et al., 2019). Hundreds of machine learning systems compete on these benchmarks, improving the state of NLU.

One key limitation of NLI research is that most of the existing corpora are only for English. Limited research has been done on multilingual and non-English corpora (Peñas et al., 2006; Conneau et al., 2018; Amirkhani et al., 2020; Ham et al., 2020; Hu et al., 2020; Mahendra et al., 2021).

Another well-known issue with NLI is the quality of the existing datasets and the limitations of the models trained on them. On most NLI corpora, state-of-the-art transformer based models can obtain quantitative results (Accuracy and F1) that equal or exceed human performance. Despite this high performance, researchers have identified numerous limitations and potential problems. Poliak et al. (2018) found that annotation artifacts in the datasets enable the models to predict the label by only looking at the hypothesis. NLI models are often prone to adversarial attacks (Williams et al., 2018) and may fail on instances that require specific linguistic capabilities (Hossain et al., 2020; Saha et al., 2020).

In this paper we address both of these shortcomings in NLI research. We present INFERES - to the best of our knowledge, the first original NLI corpus for Spanish, not adapted from another language or task. We study prior work for strategies that can reduce annotation artifacts and increase the linguistic variety of the corpus, resulting in a dataset that is more challenging for automated systems to solve. We also design the corpus in a way that facilitates systematic evaluation of automated systems on: 1) negation-based adversarial examples; 2) out-of-distribution examples.

We propose, implement, and analyze three dif-

Proceedings of the 29th International Conference on Computational Linguistics, pages 3873–3884
October 12–17, 2022.

ferent strategies for the generation and annotation of text pairs. In the *generation* strategy, expert linguists write original hypotheses given a premise. In the *rewrite* strategy, expert linguists create contrastive and adversarial examples by rewriting and re-annotating "generated" pairs. In the *annotation* strategy, we first generate text pairs in a semi-automated manner and then use crowd annotators to determine the meaning relation. The final INFERES corpus contains 8,055 gold standard premise-hypothesis pairs. The core part of the corpus is expert-generated and we make an additional effort to ensure the quality of the data and the linguistic diversity of the examples.

We provide two baseline for INFERES by fine-tuning multilingual BERT and BETO (Spanish BERT) transformer models. On the full dataset, BETO obtains 72.8% accuracy, indicating that the classification task is non-trivial. Both mBERT and BETO perform poorly in the "hypothesis-only" condition, indicating fewer annotation artifacts in the corpus compared to prior work. Both systems generalize well across the different topics in INFERES both "in-distribution" and "out-of-distribution". We notice a substantial drop in performance when evaluating negation-based adversarial examples, however the systems still outperform majority and "hypothesis-only".

INFERES expands the scope of the NLI research in Spanish, provides new set of naturally occurring contrastive and adversarial examples, and facilitates the study of negation and coreference in the context of NLI. As part of the corpus creation, we also present and analyze three unique strategies for creating examples. All our data and baseline models are being released to the community[1].

The rest of this article is organized as follows. Section 2 discusses the related work. Section 3 formulates our objectives and introduces the different corpus-creation strategies. Section 4 describes the final corpus and basic statistical data regarding it. Section 5 presents the machine learning experimental setup and results. Section 6 is devoted to a discussion of the results and their implications. Finally, Section 7 concludes the article.

2 Related Work

The task of Recognizing Textual Entailment (RTE) was proposed in Dagan et al. (2006) as a binary classification ("entailment" / "non-entailment"). The RTE competition ran for seven editions (Bar Haim et al., 2006; Giampiccolo et al., 2007, 2008; Bentivogli et al., 2009, 2010, 2011). RTE was later reformulated as a three-way decision and ultimately renamed Natural Language Inference in the SNLI (Bowman et al., 2015) and the MNLI (Williams et al., 2018) corpora. Both the RTE and the NLI tasks form part of the Natural Language Understanding benchmarks GLUE (Wang et al., 2018) and Super-GLUE (Wang et al., 2019). The NLU benchmarks attracted a lot of attention from the community and by 2020 the state-of-the-art systems reported human level performance. Parrish et al. (2021) proposed a "linguist-in-the-loop" corpus creation to improve the quality of the data.

The "super-human" performance of NLI systems has been questioned by a number of researchers. Poliak et al. (2018) found that annotation artifacts in the datasets enable the models to predict the label by only looking at the hypothesis. McCoy et al. (2019) and Gururangan et al. (2018) demonstrate that state-of-the-art NLI systems often rely on heuristics and annotation artifacts.

Systematic approaches to evaluation propose different sets of stress-tests for NLI and NLU systems (Kovatchev et al., 2018a; Naik et al., 2018; Wallace et al., 2019; Kovatchev et al., 2019; Ribeiro et al., 2020; Kovatchev et al., 2020). The attacks can be inspired by linguistic phenomena or empirical use cases. Systematic evaluations show that NLI and other NLU systems often underperform on complex linguistic phenomena such as conjunction (Saha et al., 2020), negation (Hossain et al., 2020), and coreference (Kovatchev et al., 2022). Researchers also experimented with creating contrastive examples that differ only slightly from training examples, but have a different label (Glockner et al., 2018; Kaushik et al., 2020; Gardner et al., 2020). Adversarially created datasets such as Adversarial NLI (Nie et al., 2020) and Dynabench NLI (Kiela et al., 2021) demonstrate that there is a lot of room for improvement regarding NLI datasets and models.

Most of the available resources for NLI research are in English. Conneau et al. (2018) present XNLI, a multilingual dataset created by translating English NLI examples into other languages. The interest in multilingual NLI has resulted in the creation of some novel non-English resources such as the Korean NLI corpus (Ham et al., 2020), Chinese NLI corpus (Hu et al., 2020), Persian NLI corpus

[1] At https://github.com/venelink/inferes InferES is also added as a HuggingFace dataset

(Amirkhani et al., 2020), Indonesian NLI corpus (Mahendra et al., 2021), and indigenous languages of the Americas NLI corpus (Ebrahimi et al., 2022). For Spanish, the only available resources are the Spanish portion of XNLI and the SPARTE corpus for RTE (Peñas et al., 2006) which was adapted from Question Answering data.

3 Objectives and Corpus Creation

When creating INFERES we experimented with different strategies for obtaining gold examples. To the best of our knowledge, this is the first time various annotation strategies are combined and compared in a single NLI corpus. We adopt three different approaches, used in prior work: our **generation** strategy is similar to the original RTE and NLI corpus creation; our **rewrite** strategy is inspired from work in generating adversarial and contrastive examples; our **annotation** strategy scales well with data and allows us to compare expert- and crowdcrated datasets. Our aim was to provide interesting and diverse examples that cover a large range of use cases and linguistic phenomena. We hope that INFERES can be used not only to train automated systems, but also to better understand the nature of inference. We formulated three main **objectives**:

O1 To create a native NLI dataset for the Spanish language. The existing resources are either an adaptation from a different task or a translation from English.

O2 To promote better data quality and corpus creation practices. We aim to create a more challenging dataset and simultaneously reduce the number of annotation artifacts.

O3 To facilitate the research on negation and coreference in the context of NLI. More specifically, we focus on contrastive and adversarial examples.

3.1 Premise Extraction

In the first step of the process, we extracted a set of candidate premises. We decided to use a single sentence premise, similar to SNLI and MNLI datasets. We defined two requirements for our premise sentences: 1) that they cover a range of different topics; and 2) that they be complex enough to entail or contradict multiple possible hypotheses.

Choice of topics As a source for premises we used the Spanish version of Wikipedia from October 2019. We chose six topics, covering five different domains: history, art, sports, technology, and politics. We also selected the topics in pairs hypothesizing that this selection might facilitate the creation of contrastive examples, specifically in the context of coreference.

- famous historical figures:
  Pablo Picasso (ES: Pablo Picasso)
  Christopher Columbus (ES: Cristobal Colón)
- types of "games":
  Olympic games (ES: Juegos Olímpicos)
  Video games (ES: videojuegos)
- types of multinational "unions":
  The European Union (ES: Unión Europeo)
  The Union of Soviet Socialist Republics
  (ES: Unión Sovética)

Extraction process We extracted the main Wikipedia article for each topic and preprocessed it (sentence segmentation and tokenization) using Spacy (Honnibal and Montani, 2017). We split the text by paragraphs and discarded paragraphs that contained only one sentence or more than five sentences. Then, from each paragraph, we selected a single sentence, prioritizing sentences containing negation[2] where possible, otherwise selecting a sentence at random. We ensured that each selected sentence had a length between 15 and 45 tokens.

Post-processing At the end of the extraction process, we had 471 candidate-premise sentences as follows: 82 for Picasso, 60 for Columbus, 68 for the Olympic games, 73 for video games, 107 for the EU, and 81 for the USSR. For each sentence, we also kept the corresponding paragraph to enable experimental setup where we provide an additional context to the machine learning models at train and test time. We also used the "context paragraphs" when generating "neutral" pairs. One of the authors manually inspected all 471 candidate-premise sentences. They manually resolved problems with sentence segmentation, removed URLs and internal wikipedia document references, and explicitly resolved any coreferential and anaphorical ambiguities (i.e.: replaced pronouns and coreferential entities with an unambiguous referent).

[2]To check for negation, we used a simple keyword based search, using a list of the most common negative particles, adverbs, and verbs in Spanish. The list is available at https://github.com/venelink/inferes

3.2 Expert "Generation" Strategy

Task formulation We formulated two separate *generation* tasks: the generation of entailment pairs and generation of contradiction pairs. We defined the tasks as follows:

Entailment: *Given a premise, write two different sentences that are true.*

Contradiction: *Given a premise, write two different sentences that are false.*

Our guidelines enforced a strict definition of contradiction and required our generators to write sentences that explicitly contradict the premise, rather than implicitly rely on event and actor coreference[3]. We asked the generators to provide multiple examples, requiring the use of different strategies. We further instructed the corpus generators to: 1) generate hypotheses that have a low lexical overlap with the premise; 2) generate one affirmative and one negated sentence for each relation; 3) where possible, replace named entities with pronouns or other instances of coreference. Our instructions aim to encourage generators to come up with difficult and diverse examples. We also ensure a high frequency of entailment pairs containing negation and of affirmative contradiction pairs. For a reference, the readers can see an example of generated entailment and contradiction hypotheses in **4.**

4. (PREMISE) En la década de 1980 el soporte habitual para el software era el cartucho en las videoconsolas, y el disco magnético o la cinta de casete en los ordenadores.[4]

(ENTAILMENT) Es poco probable que en los 1980s las videoconsolas y los ordenadores utilizaran el mismo soporte.[5]

(CONTRACITION) Aunque el cartucho se había utilizado en el pasado, en los 80 ya se consideraba desfasado.[6]

Sentence "generators" Four graduate students of linguistics were trained for this task by the authors of the paper. They received detailed instructions, examples, and a two-hour interactive training session prior to the start of the corpus creation. The students met with the authors of the paper on a weekly basis to discuss challenging or interesting examples. To further increase the diversity of the corpus, we recruited 24 undergraduate students for a two-hour hypothesis generation session preceded by a one-hour interactive training session. All generators were native speakers of European Spanish.

"Generated" portion of the corpus We distributed the premises extracted in Section 3.1 between the four graduate students balancing the number of entailment and contradiction pairs per topic and per premise. For any given premise, a single expert would generate only one of the relations, never both. Some premises were used more than once. Our selection strategy aimed to create maximum diversity in the data and reduce the potential bias from a relatively small number of data generators. The four graduate students created 2,284 pairs from the original 471 premises. The 24 undergraduate students generated further 872 pairs. The final corpus from the *generation* strategy contains 3,156 pairs, split equally between entailment and contradiction. We describe the process of obtaining "neutral" text pairs in Section 3.5.

3.3 Expert "Rewrite" Strategy

Task formulation The *rewrite* strategy is based on the pairs from the *generation* strategy. We defined the task as follows:

Given an existing premise–hypothesis pair, modify both the premise and the hypothesis so that:

1. the resulting sentence has a substantial difference in meaning from original

2. where possible, change the negation status of a sentence. That is, an affirmative sentence would become negated, while a negated sentence would become affirmative

3. where possible, replace some words in the original sentences with synonyms and/or coreferential entities

We further instructed the "rewriters" not to resort only to simple negation. An example of the *rewrite* strategy can be seen in **5.** and **6.**: when rewriting the premise, our expert replaced "descartaba" (EN:"ruled out") with "aceptaba" (EN: "accepted"); when rewriting the hypothesis, they changed "inaceptable" (EN: "unacceptable") for "viable" (EN: "feasible"). The resulting adversarial examples include lexical and morphological negation and are more complex than the "simple negation" benchmark of Hossain et al. (2020).

[3]For a discussion of the definition of contradiction in the context of NLI, we refer the reader to Gold et al. (2019).

[4]EN: "In the 1980s, the usual medium for software was the cartridge in video consoles, and the magnetic disk or cassette tape in computers."

[5]EN: "It is unlikely that in the 1980s video game consoles and computers used the same medium."

[6]EN: "Although the cartridge had been used in the past, by the 1980s it was already considered outdated."

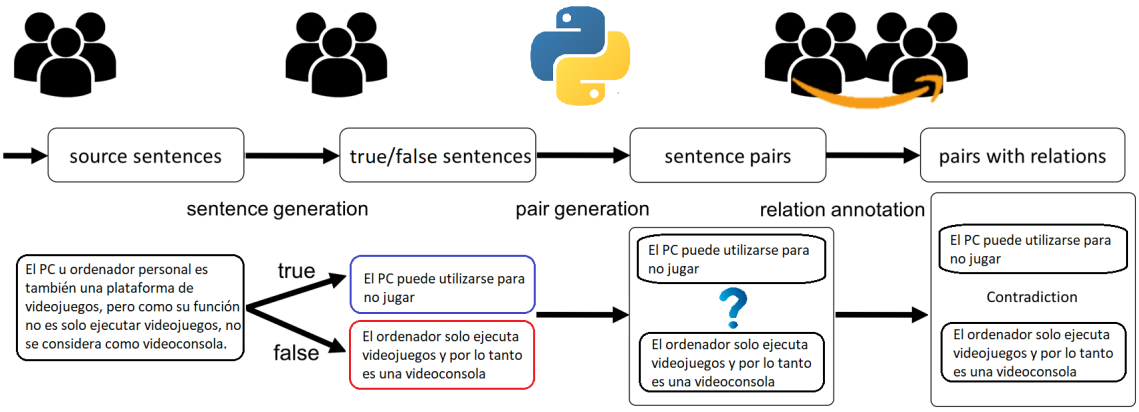

Figure 1: Pipeline for the *annotation* corpus creation strategy.

5. (PR) La reina llamó entonces a Colón, diciéndole que no **descartaba** totalmente su plan.[7]

   (HYP) La reina le hizo saber a Colón que su plan no era del todo **inaceptable**.[8]

6. (PR_RW) La reina llamó entonces a Colón, diciéndole que no **aceptaba** totalmente su plan.[9]

   (HYP_RW) La reina le hizo saber a Colón que su plan no era del todo **viable**.[10]

"Rewritten" portion of the corpus The *rewrite* process was carried out by two graduate students of linguistics. After rewriting both the premise and the hypothesis, we create three new combinations involving an original or rewritten hypothesis. In the provided example, those are the pairs 5.(Pr)–6.(Hyp_rw), 6.(Pr_rw)–5.(Hyp) and 5.(Hyp)–6.(Hyp_rw).[11] Our "rewriters" then annotated the relations between the new pairs (in the example, the relations are "contradiction", "neutral", and "entailment" respectively). As a souorce, we selected 20 entailment and 20 contradiction per topic, a total of 240 "generated" pairs. We ensured equal distribution of the original "generators" and created 720 new adversarial "rewrite" pairs.

3.4 Crowd "Annotation" Strategy

Task formulation For the crowd *annotation* strategy we adopted the three-step approach proposed by Gold et al. (2019). The authors first semi-automatically generated a large pool of premise-hypothesis pairs. Then, they applied stratified subsampling. Finally, they recruited crowd workers to annotate the meaning relations between the texts. Figure 1 illustrates the *annotation* strategy. We choose this approach since it's compatible with our *generation* and *rewrite* strategies. We were interested in comparing and combining expert- and crowd-created corpora, which, to the best of our knowledge has not been done before for NLI.

Creating a sentence pool The first step of the process was identical to the *generation* strategy. We chose 20 of the original premises, five from Picasso, Columbus, Olympic games, and video games. We chose premises that contain multiple predicates and would allow for creativity in generating entailment and contradiction pairs. In Figure 1, these premises are called "source sentences". [12]

We recruited 26 undergraduate students of linguistics and provided them with one-hour interactive training for the task of generating entailment and contradiction pairs. Each student generated 20 entailment and 20 contradiction pairs from the same 20 "source sentences". In Figure 1, these hypotheses are called "true/false sentences". [13][14] At this step, the students generated a "sentence pool" of of 1,040 "true/false sentences".

Pair generation In the second step we combined the sentences from the "sentence pool" in pairs using three different selection strategies. The "true-

[7]EN: The queen then called Columbus, telling him that she did not fully **rule out** his plan.

[8]EN: The queen let Columbus know that his plan was not entirely **unacceptable**.

[9]EN: The queen then called Columbus, telling him that she did not fully **accept** his plan.

[10]EN: The queen let Columbus know that his plan was not entirely **feasible**.

[11]We do not to use 5.(Pr)–6(Pr_rw) due to sentence length.

[12]EN: "The PC or personal computer is also a video game platform, but since its function is not only to play video games, it is not considered a video game console."

[13]EN: (true) "The PC can be used for non-gaming."

[14]EN: (false) "The computer only runs video games and is therefore a game console."

true" strategy combines two "true" sentences derived from the same premise. The "false-false" strategy combines two "false" sentences from the same premise. The "true-false" strategy combines one true and one false sentence derived from the same premise. Unlike Gold et al. (2019), we do not include a random pairing and do not downsample "false-false" and "true-false" strategies. We randomly selected 2,000 of the pairs for annotation, ensuring equal distribution of strategies.

Pair annotation In the third step, we asked crowd workers to annotate the textual relation between pairs. We used the WARP-Text (Kovatchev et al., 2018b) annotation interface for the annotation. Following Gold et al. (2019), we created two separate binary annotation tasks - one for entailment and one for contradiction. For entailment, we included each pair twice, changing the order of P and H to reflect the directional nature of the relation. If a sentence was annotated as not-entailment and not-contradiction, we marked it as "neutral".

We use three annotators for each example. Following prior work (Marelli et al., 2014; Gold et al., 2019), we calculated the agreement as the average % of annotators that voted for the majority label. We obtained 86.9% agreement for the "entailment" relation and 85.6% agreement for the "contradiction" relation. We also calculated the Fleiss' kappa score, obtaining a "moderate agreement" of 55.[15] Our agreement and label distribution of labels are consistent with the results reported by Gold et al. (2019) for 10 annotators. Since 56% of the pairs were labeled "neutral", we kept all "entailment" and "contradiction" pairs and randomly downsampled the "neutral" to obtain a balance between the classes. The *annotate* portion contains 1,290 pairs.

3.5 Generating Neutral Pairs

Using our *generation* strategy, we created "entailment" and "contradiction" pairs. Using our *rewrite* strategy, we created pairs with all three relations. However the "neutral" class was underrepresented compared to the other two. To create a balanced dataset for training automated systems, we needed a separate strategy to introduce more pairs with the "neutral" label. In this subsection, we describe four different rule-based strategies that we used to generate "neutral" pairs in an automated manner.

[15]The lower kappa is likely due to the label imbalance in the corpus: only about 22% of the pairs contain entailment and 20% contain contradiction.

Shuffling existing P and H (same topic) We matched each premise to two random hypotheses, generated for different premises on the same topic.

Matching existing P with random contexts (same topic) In Section 3.1, we kept a "context" paragraph for each premise that we extracted. We matched each premise to two random "contexts" on the same topic. We then randomly selected a sentence from each of those contexts.

Matching existing H with random contexts (same topic) Similar to the previous strategy, we randomly matched each hypothesis to a sentence from a "context" paragraph on the same topic.

Shuffling existing P and H (different topics) Typically, the premise and the hypothesis have at least some degree of semantic similarity. We argue that an automated NLI solution should be able to label unrelated pairs. We created a small fixed number of unrelated pairs by matching texts and hypotheses from different topics.

Validating neutral pairs We selected 240 pairs, 60 from each of the four strategies, to manually validate the quality of the "neutral" pairs. One of our "sentence generators" performed a two-stage annotation on each pair. At the first stage they annotated *"whether the premise and hypothesis are semantically releated"*. At the second stage they annotated *"whether the meaning relation is neutral, despite a potential semantic relatedness"*. 55% of the "neutral" pairs had some semantic relation (e.g., shared topic or named entities), and 26% had a *strong* relation. 237 out of the 240 examples (98.75%) were annotated as "neutral". Two pairs were found to be "entailment" and one - "contradiction".

The "neutral" portion of the corpus Through generation and downsampling, we obtain a total of 1,291 "neutral" sentences for INFERES . We use 298 of them to re-balance the *rewrite* portion of the corpus and the remaining 1,893 to complete the *generation* portion of the corpus.

4 INFERES

We combined the examples from all corpus creation strategies to create INFERES - a corpus of NLI for Spanish containing 8,055 text pairs. Table 1 shows the distribution of pairs and labels based on the creation strategy. Note that for *generation* and *rewrite* strategies, the "neutral" examples were at least in part generated automatically to ensure

label balance. For "annotate" the "neutral" pairs are naturally occurring. More than half of the corpus, 5,029 text pairs, was created using the *generate* strategy. This is the core part of the corpus, in which we have incorporated multiple strategies for ensuring the quality and the linguistic diversity of the examples. 1,716 pairs were created using the *rewrite* strategy and 1,290 pairs were generated using the *annotate* strategy. All six topics are represented roughly equally. In the *generate* and *rewrite* portions, we aimed to ensure that each original premise has the same number of hypotheses, distributed equally across relations.

Strategy	Pairs	Ent	Cnt	Neu
Full	8,055	2,399	2,687	2,969
Generate	5,029*	1,574	1,582	1,893*
Rewrite	1,716*	398	712	606*
Annotate	1,290	427	393	470

Table 1: Distribution of labels INFERES by strategy

We measured the vocabulary size and the lexical overlap between the premise and hypothesis. The full INFERES has a vocabulary size of 12,877 unique types. On average, 22.6% of the tokens from the premise also appear in the hypothesis. 33.4% of the tokens from the hypothesis also appear in the premise. The two numbers differ since we count the number of non-unique tokens, including repetition, and we normalize them using a different denominator (length of premise/hypothesis). The overlap is comparable with prior work for English (20% and 38% for MNLI and 18% and 34% for linguist-in-the-loop NLI).

5 Machine Learning Experiments

To demonstrate the utility of INFERES , we carried out a set of machine learning experiments. The design of INFERES allows us to test NLI models under a variety of conditions: standard train/test split, hypothesis-only condition, performance on negation-based adversarial examples, and performance by topic in- and out-of-distribution.

Machine learning models We used two transformer based models, pretrained for Spanish: the multilingual version of BERT (Devlin et al., 2019) and the Spanish version of BERT, BETO (Cañete et al., 2020). We used the version of the models available on HuggingFace (Wolf et al., 2020) as of May 2022 and finetuned them on INFERES . After experimenting with different hyperparameter settings, we empirically found the best performance using a PolinomialDecay learning rate scheduler and training the model for five epochs. We kept the rest of the hyperparameters at their default values and used ADAM optimizer[16]. All reported results are the average of five different random initializations.

Condition	MB	mBERT	BETO
Full Dataset	36.8	69.6	72.8
Hyp. Only	36.8	38.8	42.3
Adv. Negation	41.5	52	51.2

Table 2: Performance of multilingual-BERT and Spanish BERT (BETO) on INFERES across different conditions. MB: "majority baseline". 'Full": standard train/test split."Hyp. Only": hypothesis-only. "Adv. Negation": negation-based adversarial examples.

Full corpus performance In a standard "full corpus" condition, we used 80% of INFERES for training, 10% for validation, and 10% for testing. We use the examples generated by all three strategies for both training and testing. As shown in Table 2, BETO obtained 72.8% accuracy and outperformed mBERT (69.6%). While both models reached a fair performance on the test set, the results were much lower than the "super-human" performance that state-of-the-art transformer models obtain on popular benchmarks for English. For a reference, the official performance for the Spanish portion of XNLI is 82% for BETO-cased, and 78.5% for mBERT[17].

Hypothesis-only performance In the "hypothesis-only" condition, the models do not have access to the "premise" during training or testing. NLI explores the meaning relation between the two texts - the premise and the hypothesis. If a model is exposed only to one of the texts, its performance should not exceed that of a random guess, roughly equal to predicting the most common class.

Prior work has shown that existing datasets for English contain a large number of "annotation artifacts" and models are able to obtain much higher performance than chance. Poliak et al. (2018) report 55% accuracy for the "hypothesis-only" condi-

[16]The code for the experiments and all hyperparameters are available at https://github.com/venelink/inferes

[17]See https://github.com/dccuchile/beto

tion on the MNLI corpus using non-transformer models, an increase of 20% over the majority baseline. Parrish et al. (2021) show that their "linguist-in-the-loop" approach is less biased in the hypothesis-only condition, but they also report accuracy over 50% on the full dataset using a ROBERTA transformer.

In the "hypothesis-only" condition on INFERES , BETO obtained 42.3%, and mBERT – 38.8%, which is respectively 5.5% and 2% higher than the majority baseline. The relatively small improvement over the majority baseline indicates that the hypothesis-only artifacts in INFERES are substantially fewer than in previous work.

Performance on negation-based adversarial examples For this experiment, we trained the models on the *generation* and *annotation* portion of the corpus and evaluated them on *rewrite*. The setup is similar to the one from Hossain et al. (2020). The performance of mBERT and BETO drops significantly when facing adversarial examples (See Table 2). The models obtain 52% and 51.1% accuracy, about 10% higher than the majority baseline. Hossain et al. (2020) report that on two of the three datasets they use, BERT performs worse than the majority baseline. Our experimental setup is arguably more difficult, since INFERES contains multiple negation strategies rather than just negating the main verb. The *rewrite* portion of the corpus can provide further insight into the use of negation in NLI and we hope it would facilitate further research and improvement in the area.

In- and Out-of-distribution generalization by topic We also carried out a set of experiments to determine the ability of models to generalize across the six different topics. We evaluated the models in two different conditions.

The in-distribution (ID) condition is an extension of the "full corpus" condition. We split the full corpus containing all six topics in an 80/10/10 ratio. Then, when evaluating the performance of the models, we split the test set in six sub-sets, based on their topic and we measured the model accuracy on each sub-set. To ensure that the variation of the model performance is not due to a sampling bias, we re-trained each model five times, using a different 80/10/10 random split each time. In Table 3, we report the average accuracy across the five different splits. Both models obtained the highest ID performance on the topics of "Picasso" and "The European Union" and the lowest ID performance on "The USSR".

Top	mBERT		BETO	
	IND	OOD	ID	OOD
All	69.6		72.8	
1	73.1±4	67.9±2	78.5±4	73.8±.6
2	68.1±4	69.9±1	74.6±3	72.0±.7
3	70.8±3	70.9±1	74.2±3	73.3±.7
4	71.6±2	68.9±.7	69.4±5	69.7±.7
5	77.1±3	76.1±.5	78.3±4	77.2±.9
6	65.3±4	69.8±.9	68.6±5	69.6±.9

Table 3: In-distribution (ID) and Out-of-distribution (OOD) performance of mBERT and BETO on different topics within INFERES . ID: model trained on data covering all 6 topics. OOD: model trained on 5 topics and evaluated on the unseen 6th. 1 (Picasso), 2 (Columbus), 3 (Olympics), 4 (Videogames), 5 (EU), 6 (USSR).

For the out-of-distribution (OOD) condition, we designed a transfer learning experiment, in which we trained mBERT and BETO on five of the topics and evaluate on the sixth. The results presented in Table 3 demonstrate that the models are able to generalize well across the topics, even in a transfer learning setup. For both models, the OOD performance on most topics drops between 1% and 5% compared to ID. The performance on "Olympics" for mBERT and on "videogames" for BETO was almost identical between conditions. For "The USSR", both models obtained higher performance for OOD. We inspected the matter further and noticed that due to the corpus size, the ID random split is not very stable (the ID test set only contains between 100 and 120 instances of each topic) and the average can be affected by outliers. The OOD results are much more stable due to the test set having over 1,000 examples per topic. This can be seen in the difference in standard deviation: between 2% and 5% for ID, and below 1% for OOD. Our experiments demonstrate that the models are able to generalize well even to topics they have never seen during training, a promising finding for the overall generalizability of NLI models.

6 Discussion

In Section 3, we formulated three main objectives behind INFERES . In this section, we want to revisit those objectives and briefly discuss the importance of our work for the NLP and NLI communities.

Our **first objective** was *"To create a native NLI dataset for the Spanish language"*. To the best of

our knowledge, INFERES is the first native NLI dataset for Spanish which is not adapted or translated. We described the creation process and validated that it can be used to train different machine learning models. We have successfully contributed a new resource to the Spanish NLP community and we hope that INFERES can facilitate the further creation of tools and resources for that language.

Our **second objective** was *"To promote better data quality and corpus creation practices."*. We proposed, implemented, analyzed, and compared several different strategies for creating text pairs. The resulting dataset proves non-trivial to state-of-the-art NLP models with an overall accuracy in the low 70s. This leaves a lot of room for improvement and future research. The results using a "hypothesis-only" baseline indicate that INFERES contains fewer annotation artifacts than prior work. At the same time, models trained on the dataset are able to generalize well across different topics, even in our "out-of-distribution" condition. Overall, we can conclude that INFERES is of high quality and achieves the objective of promoting better data by design.

Our **third objective** was *"To facilitate the research on negation and coreference in the context of NLI."*. Our *rewrite* strategy was focused on creating naturally occurring contrastive and adversarial examples based on negation and coreference. We followed prior work on evaluating systems' performance and demonstrated that those examples are non-trivial to solve. However, the two models are still able to outperform the majority baseline by over 10%. These findings indicate that the problem is not unsolvable and the models are learning something about complex negation from the data. INFERES can facilitate the resaerch of negation in Spanish both in the context of NLI and in isolation. We leave quantifying the importance of coreference in the *rewrite* section for future work.

Overall, we have achieved all our objectives: 1) we created a novel dataset for Spanish; 2) we used different generation and annotation strategies to obtain a challenging corpus with fewerr annotation artifacts; 3) we created a set of high-quality contrastive and adversarial examples based on negation and coreference. We believe that INFERES is an important contribution to Spanish NLP, and also to researchers interested in NLI, negation, and coreference. We hope that this dataset can be used to train and evaluate more accurate automated systems, but also to better understand the nature of those linguistic phenomena.

7 Conclusions

We presented INFERES - a new corpus of Natural Language Inference for Spanish. To the best of our knowledge, this is the first original Spanish NLI corpus that is not a translation or an adaptation of an existing dataset. We explored several different strategies for corpus creation and put the emphasis on creating diverse and non-trivial examples, that are also linguistically interesting. More specifically, we created contrastive and adversarial examples involving complex negation and coreference.

We provided two baseline transformer-based systems finetuned on the dataset. We demonstrated that INFERES is challenging and contains fewer anotation artifacts than prior work. We also evaluated the performance of automated systems on adversarial examples and the ability of the models to generalize across topics in- and out-of-distribution. The results validated the quality and the difficulty of the corpus. INFERES leaves a room for analysis and improvement.

Our work opens several directions for future work: studying and improving the performance of NLI models for Spanish; expanding the research on negation in Spanish, and specifically the complex and lexical negation; evaluating the importance of coreference in the context of NLI. We believe that INFERES will be useful both to Spanish researchers and to the general NLP community.

Acknowledgements

We want to thank M. Antónia Martí and Montse Nofre for their suggestions and support for this work. We are grateful to Patricia Grau Francitorra, Eugenia Verjovodkina, Víctor Bargiela Zotes, and Xavier Bonet Casals for the annotations and the discussions about the annotation process. We also want to thank the anonymous reviewers for their feedback and suggestions.

This work was partially funded by the project "FairTransNLP-Language: Analysing Toxicity and Stereotypes in Language for Unbiased, Fair and Transparent Systems (PID2021-124361OB-C33)" funded by Ministerio de Ciencia e Innovación (Spain) and co-funded by the European Regional Development Fund (FEDER). Most of this work was carried out during the first author's APIF grant at the University of Barcelona.

References

Hossein Amirkhani, Mohammad AzariJafari, Zohreh Pourjafari, Soroush Faridan-Jahromi, Zeinab Kouhkan, and Azadeh Amirak. 2020. Farstail: A persian natural language inference dataset.

Roy Bar Haim, Ido Dagan, Bill Dolan, Lisa Ferro, Danilo Giampiccolo, Bernardo Magnini, and Idan Szpektor. 2006. The second pascal recognising textual entailment challenge.

Luisa Bentivogli, Peter Clark, Ido Dagan, and Danilo Giampiccolo. 2009. The Fifth PASCAL Recognizing Textual Entailment Challenge. In *TAC*.

Luisa Bentivogli, Peter Clark, Ido Dagan, and Danilo Giampiccolo. 2010. The Sixth PASCAL Recognizing Textual Entailment Challenge. In *TAC*.

Luisa Bentivogli, Peter Clark, Ido Dagan, and Danilo Giampiccolo. 2011. The Seventh PASCAL Recognizing Textual Entailment Challenge. In *TAC*.

Samuel R Bowman, Gabor Angeli, Christopher Potts, and Christopher D Manning. 2015. A large annotated corpus for learning natural language inference. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 632–642.

José Cañete, Gabriel Chaperon, Rodrigo Fuentes, Jou-Hui Ho, Hojin Kang, and Jorge Pérez. 2020. Spanish pre-trained bert model and evaluation data. In *PML4DC at ICLR 2020*.

Alexis Conneau, Ruty Rinott, Guillaume Lample, Adina Williams, Samuel Bowman, Holger Schwenk, and Veselin Stoyanov. 2018. XNLI: Evaluating cross-lingual sentence representations. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 2475–2485, Brussels, Belgium. Association for Computational Linguistics.

Ido Dagan, Oren Glickman, and Bernardo Magnini. 2006. The PASCAL recognising textual entailment challenge. In *Machine learning challenges. evaluating predictive uncertainty, visual object classification, and recognising tectual entailment*, pages 177–190. Springer.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, Minneapolis, Minnesota. Association for Computational Linguistics.

Abteen Ebrahimi, Manuel Mager, Arturo Oncevay, Vishrav Chaudhary, Luis Chiruzzo, Angela Fan, John Ortega, Ricardo Ramos, Annette Rios, Ivan Vladimir Meza Ruiz, Gustavo Giménez-Lugo, Elisabeth Mager, Graham Neubig, Alexis Palmer, Rolando Coto-Solano, Thang Vu, and Katharina Kann. 2022. AmericasNLI: Evaluating zero-shot natural language understanding of pretrained multilingual models in truly low-resource languages. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 6279–6299, Dublin, Ireland. Association for Computational Linguistics.

Matt Gardner, Yoav Artzi, Victoria Basmov, Jonathan Berant, Ben Bogin, Sihao Chen, Pradeep Dasigi, Dheeru Dua, Yanai Elazar, Ananth Gottumukkala, Nitish Gupta, Hannaneh Hajishirzi, Gabriel Ilharco, Daniel Khashabi, Kevin Lin, Jiangming Liu, Nelson F. Liu, Phoebe Mulcaire, Qiang Ning, Sameer Singh, Noah A. Smith, Sanjay Subramanian, Reut Tsarfaty, Eric Wallace, Ally Zhang, and Ben Zhou. 2020. Evaluating models' local decision boundaries via contrast sets. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 1307–1323, Online. Association for Computational Linguistics.

Danilo Giampiccolo, Hoa Trang Dang, Bernardo Magnini, Ido Dagan, Elena Cabrio, and Bill Dolan. 2008. The Fourth PASCAL Recognizing Textual Entailment Challenge. In *TAC*.

Danilo Giampiccolo, Bernardo Magnini, Ido Dagan, and Bill Dolan. 2007. The third pascal recognizing textual entailment challenge. In *Proceedings of the ACL-PASCAL workshop on textual entailment and paraphrasing*, pages 1–9. Association for Computational Linguistics.

Max Glockner, Vered Shwartz, and Yoav Goldberg. 2018. Breaking NLI systems with sentences that require simple lexical inferences. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 650–655, Melbourne, Australia. Association for Computational Linguistics.

Darina Gold, Venelin Kovatchev, and Torsten Zesch. 2019. Annotating and analyzing the interactions between meaning relations. In *Proceedings of the 13th Linguistic Annotation Workshop*, pages 26–36, Florence, Italy. Association for Computational Linguistics.

Suchin Gururangan, Swabha Swayamdipta, Omer Levy, Roy Schwartz, Samuel Bowman, and Noah A. Smith. 2018. Annotation artifacts in natural language inference data. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 2 (Short Papers)*, pages 107–112, New Orleans, Louisiana. Association for Computational Linguistics.

Jiyeon Ham, Yo Joong Choe, Kyubyong Park, Ilji Choi, and Hyungjoon Soh. 2020. KorNLI and KorSTS: New benchmark datasets for Korean natural language understanding. In *Findings of the Association for*

Computational Linguistics: EMNLP 2020, pages 422–430, Online. Association for Computational Linguistics.

Matthew Honnibal and Ines Montani. 2017. spaCy 2: Natural language understanding with Bloom embeddings, convolutional neural networks and incremental parsing. To appear.

Md Mosharaf Hossain, Venelin Kovatchev, Pranoy Dutta, Tiffany Kao, Elizabeth Wei, and Eduardo Blanco. 2020. An analysis of natural language inference benchmarks through the lens of negation. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 9106–9118, Online. Association for Computational Linguistics.

Hai Hu, Kyle Richardson, Liang Xu, Lu Li, Sandra Kübler, and Lawrence Moss. 2020. OCNLI: Original Chinese Natural Language Inference. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 3512–3526, Online. Association for Computational Linguistics.

Divyansh Kaushik, Eduard Hovy, and Zachary Lipton. 2020. Learning the difference that makes a difference with counterfactually-augmented data. In *International Conference on Learning Representations*.

Douwe Kiela, Max Bartolo, Yixin Nie, Divyansh Kaushik, Atticus Geiger, Zhengxuan Wu, Bertie Vidgen, Grusha Prasad, Amanpreet Singh, Pratik Ringshia, Zhiyi Ma, Tristan Thrush, Sebastian Riedel, Zeerak Waseem, Pontus Stenetorp, Robin Jia, Mohit Bansal, Christopher Potts, and Adina Williams. 2021. Dynabench: Rethinking benchmarking in NLP. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 4110–4124, Online. Association for Computational Linguistics.

Venelin Kovatchev, Trina Chatterjee, Venkata S Govindarajan, Jifan Chen, Eunsol Choi, Gabriella Chronis, Anubrata Das, Katrin Erk, Matthew Lease, Junyi Jessy Li, Yating Wu, and Kyle Mahowald. 2022. longhorns at DADC 2022: How many linguists does it take to fool a question answering model? a systematic approach to adversarial attacks. In *Proceedings of the First Workshop on Dynamic Adversarial Data Collection*, pages 41–52, Seattle, WA. Association for Computational Linguistics.

Venelin Kovatchev, Darina Gold, M. Antonia Marti, Maria Salamo, and Torsten Zesch. 2020. Decomposing and comparing meaning relations: Paraphrasing, textual entailment, contradiction, and specificity. In *Proceedings of the 12th Language Resources and Evaluation Conference*, pages 5782–5791, Marseille, France. European Language Resources Association.

Venelin Kovatchev, M. Antònia Martí, and Maria Salamó. 2018a. ETPC - a paraphrase identification corpus annotated with extended paraphrase typology and negation. In *Proceedings of the Eleventh International Conference on Language Resources and Evaluation (LREC 2018)*, Miyazaki, Japan. European Language Resources Association (ELRA).

Venelin Kovatchev, M. Antònia Martí, and Maria Salamó. 2018b. WARP-text: a web-based tool for annotating relationships between pairs of texts. In *Proceedings of the 27th International Conference on Computational Linguistics: System Demonstrations*, pages 132–136, Santa Fe, New Mexico. Association for Computational Linguistics.

Venelin Kovatchev, M. Antonia Marti, Maria Salamo, and Javier Beltran. 2019. A qualitative evaluation framework for paraphrase identification. In *Proceedings of the International Conference on Recent Advances in Natural Language Processing (RANLP 2019)*, pages 568–577, Varna, Bulgaria. INCOMA Ltd.

Rahmad Mahendra, Alham Fikri Aji, Samuel Louvan, Fahrurrozi Rahman, and Clara Vania. 2021. IndoNLI: A natural language inference dataset for Indonesian. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 10511–10527, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Marco Marelli, Stefano Menini, Marco Baroni, Luisa Bentivogli, Raffaella Bernardi, and Roberto Zamparelli. 2014. A SICK cure for the evaluation of compositional distributional semantic models. In *Proceedings of the Ninth International Conference on Language Resources and Evaluation (LREC'14)*, pages 216–223, Reykjavik, Iceland. European Language Resources Association (ELRA).

Tom McCoy, Ellie Pavlick, and Tal Linzen. 2019. Right for the wrong reasons: Diagnosing syntactic heuristics in natural language inference. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 3428–3448, Florence, Italy. Association for Computational Linguistics.

Aakanksha Naik, Abhilasha Ravichander, Norman Sadeh, Carolyn Rose, and Graham Neubig. 2018. Stress test evaluation for natural language inference. In *Proceedings of the 27th International Conference on Computational Linguistics*, pages 2340–2353, Santa Fe, New Mexico, USA. Association for Computational Linguistics.

Yixin Nie, Adina Williams, Emily Dinan, Mohit Bansal, Jason Weston, and Douwe Kiela. 2020. Adversarial NLI: A new benchmark for natural language understanding. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 4885–4901, Online. Association for Computational Linguistics.

Alicia Parrish, William Huang, Omar Agha, Soo-Hwan Lee, Nikita Nangia, Alexia Warstadt, Karmanya Aggarwal, Emily Allaway, Tal Linzen, and Samuel R. Bowman. 2021. Does putting a linguist in the loop

improve NLU data collection? In *Findings of the Association for Computational Linguistics: EMNLP 2021*, pages 4886–4901, Punta Cana, Dominican Republic. Association for Computational Linguistics.

Anselmo Peñas, Álvaro Rodrigo, and Felisa Verdejo. 2006. Sparte, a test suite for recognising textual entailment in spanish. In *Computational Linguistics and Intelligent Text Processing*, pages 275–286, Berlin, Heidelberg. Springer Berlin Heidelberg.

Adam Poliak, Jason Naradowsky, Aparajita Haldar, Rachel Rudinger, and Benjamin Van Durme. 2018. Hypothesis only baselines in natural language inference. In *Proceedings of the Seventh Joint Conference on Lexical and Computational Semantics*, pages 180–191, New Orleans, Louisiana. Association for Computational Linguistics.

Marco Tulio Ribeiro, Tongshuang Wu, Carlos Guestrin, and Sameer Singh. 2020. Beyond accuracy: Behavioral testing of NLP models with CheckList. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 4902–4912, Online. Association for Computational Linguistics.

Swarnadeep Saha, Yixin Nie, and Mohit Bansal. 2020. ConjNLI: Natural language inference over conjunctive sentences. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 8240–8252, Online. Association for Computational Linguistics.

Eric Wallace, Shi Feng, Nikhil Kandpal, Matt Gardner, and Sameer Singh. 2019. Universal adversarial triggers for attacking and analyzing NLP. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 2153–2162, Hong Kong, China. Association for Computational Linguistics.

Alex Wang, Yada Pruksachatkun, Nikita Nangia, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel Bowman. 2019. Superglue: A stickier benchmark for general-purpose language understanding systems. In *Advances in Neural Information Processing Systems*, volume 32. Curran Associates, Inc.

Alex Wang, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel Bowman. 2018. GLUE: A multi-task benchmark and analysis platform for natural language understanding. In *Proceedings of the 2018 EMNLP Workshop BlackboxNLP: Analyzing and Interpreting Neural Networks for NLP*, pages 353–355, Brussels, Belgium. Association for Computational Linguistics.

Adina Williams, Nikita Nangia, and Samuel Bowman. 2018. A broad-coverage challenge corpus for sentence understanding through inference. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 1112–1122, New Orleans, Louisiana. Association for Computational Linguistics.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Remi Louf, Morgan Funtowicz, Joe Davison, Sam Shleifer, Patrick von Platen, Clara Ma, Yacine Jernite, Julien Plu, Canwen Xu, Teven Le Scao, Sylvain Gugger, Mariama Drame, Quentin Lhoest, and Alexander Rush. 2020. Transformers: State-of-the-art natural language processing. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 38–45, Online. Association for Computational Linguistics.

ParaZh-22M: a Large-Scale Chinese Parabank via Machine Translation

Wenjie Hao[1], **Hongfei Xu**[1], **Deyi Xiong**[2], **Hongying Zan**[1,3], **Lingling Mu**[1*]
[1] Zhengzhou University, [2] Tianjin University, [3] Peng Cheng Laboratory
haowj9977@163.com, hfxunlp@foxmail.com, dyxiong@tju.edu.cn
{iehyzan,iellmu}@zzu.edu.cn

Abstract

Paraphrasing, i.e., restating the same meaning in different ways, is an important data augmentation approach for natural language processing (NLP). Zhang et al. (2019b) propose to extract sentence-level paraphrases from multiple Chinese translations of the same source texts, and construct the PKU Paraphrase Bank of 0.5M sentence pairs. However, despite being the largest Chinese parabank to date, the size of PKU parabank is limited by the availability of one-to-many sentence translation data, and cannot well support the training of large Chinese paraphrasers. In this paper, we relieve the restriction with one-to-many sentence translation data, and construct ParaZh-22M, a larger Chinese parabank that is composed of 22M sentence pairs, based on one-to-one bilingual sentence translation data and machine translation (MT). In our data augmentation experiments, we show that paraphrasing based on ParaZh-22M can bring about consistent and significant improvements over several strong baselines on a wide range of Chinese NLP tasks, including a number of Chinese natural language understanding benchmarks (CLUE) and low-resource machine translation. [1]

1 Introduction

A paraphrase is a restatement of meaning with different expressions (Bhagat and Hovy, 2013). Paraphrasing has been proven to be an effective data augmentation approach for many NLP tasks, ranging from linguistically controlled paraphrase generation (Iyyer et al., 2018; Chen et al., 2019; Li et al., 2019; Sun et al., 2021), style transfer (Krishna et al., 2020), to applications like low-resource machine translation (Khayrallah et al., 2020) and automatic MT evaluation (Thompson and Post, 2020; Bawden et al., 2020).

Zhang et al. (2019b) extract sentence-level paraphrases from multiple Chinese translations of the same source texts, and create the largest Chinese paraphrase bank (PKU Parabank) to date, which contains 509,832 pairs of paraphrased sentences. However, the amount of one-to-many sentence translation data constrains the size of PKU parabank, and it cannot meet the requirement to train large Chinese paraphrasers.

Inspired by Wieting and Gimpel (2018) and Hu et al. (2019a,b), we propose to relax the restriction that requires one-to-many translation data on the construction of large-scale Chinese parabanks, by utilizing bilingual one-to-one translation data of larger scales and MT, and construct ParaZh-22M. Specifically, we leverage the huge Chinese-English machine translation data from WMT 2021 (Akhbardeh et al., 2021) of 30.4M sentence pairs, apply **strict rules** to ensure the data quality, and translate the English side of the parallel corpus to Chinese with the cutting-edge deep Transformers and **several approaches** to ensure the translation quality. We pick the machine translated Chinese sentences considering both diversity and semantic consistency, and pair with the corresponding original Chinese references to form paraphrase pairs. Compared to the PKU parabank, ParaZh-22M is ∼40 times as large, involves a broader range of paraphrase phenomena and domains, and can support the training of large Chinese paraphrasers.

Our main contributions are as follows:

- We propose to relieve the need of one-to-many translation data for the construction of Chinese parabank, and construct a Chinese parabank of 22M sentence pairs based on one-to-one sentence translation data and advanced MT models, which involves many domains and is ∼40 times as large as the previous largest PKU Chinese parabank;
- We test the effects of data augmentation via

* Corresponding author.

[1]We opensource our dataset at https://github.com/haowj9977/parazh-22M.

Proceedings of the 29th International Conference on Computational Linguistics, pages 3885–3897
October 12–17, 2022.

paraphrasing based on our parabank on a wide range of Chinese NLP tasks, including short/long text classification, natural language inference, keyword recognition, and low-resource machine translation, and show that paraphrasing based on our parabank is able to achieve consistent and significant improvements over several baselines.

2 Construction of ParaZh-22M

Zhang et al. (2019b) extract sentence-level paraphrases from multiple Chinese translations of the same source texts. Their approach requires one-to-many translation data, which is hard to collect. Instead, we try to relieve this restriction in Chinese parabank construction, and build the parabank based on one-to-one sentence-level translation data. Specifically, we translate the translation of Chinese sentences in the parallel data back to Chinese with the cutting-edge Neural Machine Translation (NMT) technology, and construct the parabank by pairing the machine translated Chinese sentences with the corresponding original Chinese sentences.

We suggest that the semantic consistency and quality of the parallel bank is ensured by the cutting-edge NMT algorithm, as the translation quality of advanced NMT methods is already close to that of translation agencies in high resource scenarios (Akhbardeh et al., 2021).

The construction of ParaZh-22M can be divided into 4 steps: 1) data collection, 2) data processing and cleaning, 3) training of NMT models, and 4) paraphrase generation.

2.1 Data Collection

We leverage bilingual parallel sentence data for the construction of the Chinese parabank and the training of NMT models, and monolingual data to further boost the performance of NMT via back-translation (Sennrich et al., 2016a). We select several datasets from WMT 2021 Chinese-English news translation task (Akhbardeh et al., 2021), and statistics are shown in Table 1. Even though the data is collected for the news translation task in WMT, they indeed involve many domains, e.g., the ParaCrawl corpus is the extraction of parallel sentences from the web regardless of their domains.

Bilingual parallel corpus To ensure the quality of the training NMT models and the parabank, we manually check the quality of each dataset provided by WMT 2021 for the Chinese-English news translation task, and take three datasets into consideration: the United Nations Parallel Corpus v1.0 (Ziemski et al., 2016), News Commentary v16, and ParaCrawl dataset (Bañón et al., 2020).

	Dataset	Size
bilingual	United Nations Parallel Corpus	15.9M
	ParaCrawl	14.2M
	News Commentary v16	0.3M
	total	30.4M
monolingual	News Crawl	10.6M

Table 1: Statistics of the bilingual and monolingual data. Size: the number of sentence pairs (for bilingual data) / sentences (for monolingual data). The monolingual data contain 10.6M sentences per language.

The United Nations Parallel Corpus (Ziemski et al., 2016) contains over 15.9M English-Chinese sentence pairs, which is composed of official records and other parliamentary documents of the United Nations that are in the public domain. The current version of the corpus contains content that was produced and manually translated between 1990 and 2014.

The News Commentary dataset is a collection of news about general politics, economics and science. Its English-Chinese section has about 0.3M sentence pairs.

The ParaCrawl dataset (Bañón et al., 2020) contains about 14.2M English-Chinese parallel sentences, constructed through web crawling software. Although its quality is slightly worse than the other two datasets in our manual evaluation, the ParaCrawl dataset involves many domains and provides a large number of training samples. In our experiments on the Zh→En task with base Transformers, using ParaCrawl dataset can bring about $+5.4$ and $+3.8$ BLEU improvements on the newstest 2020 and newstest 2021 test sets respectively.

Monolingual corpus Back translation is a simple and effective approach to improve the performance of MT with monolingual data (Sennrich et al., 2016a; Fadaee and Monz, 2018; Edunov et al., 2018; Wang et al., 2019b; Dou et al., 2020; Wei et al., 2020a; Marie et al., 2020). To further boost the performance of our NMT models (§ 2.3), and to obtain more accurate probability estimation in dual scoring (§ 2.4), we collect monolingual data of both languages, and augment the parallel data with the back translated monolingual data for the training of NMT models. Using back-translation data for NMT models' training also helps improve

the translation diversity and alleviate the overfitting issue on the parallel data. The back translated data are not used for the construction of the parabank, to avoid introducing back-translation noise into the parabank and to ensure the quality of the parabank.

Specially, we extract ∼10.6M sentences for both English and Chinese from the monolingual News Crawl dataset, which provides article texts from various online news.

2.2 Data Processing and Cleaning

The quality of the dataset affects the performance of NMT and decides the quality of the parabank.

As many data are crawled from the web, we first standardize the texts with the following pipeline:

1. removing sentences with encoding errors;
2. replacing full-width characters with their corresponding half-width characters;
3. normalizing punctuation;
4. converting all named and numeric character HTML references (e.g., >, >, &#x3e) to their corresponding Unicode characters;
5. converting Traditional Chinese to Simplified Chinese through OpenCC. [2]

For the training of NMT models, we tokenize and truecase the English part with Moses (Koehn et al., 2007), and segment Chinese sentences into words using jieba. [3]

To clean the English-Chinese parallel corpus: 1) we only retain the most frequent instance when the source sentence has multiple translations in the data, 2) we remove the training instances where low frequency tokens take a large part of the sentence pairs, and 3) we remove sentence pairs with abnormally large source-vs-target length ratios. After the data cleaning, around 26.4M sentence pairs are left for the training of NMT models and the construction of the parabank.

We perform independent Byte Pair Encoding (BPE) (Sennrich et al., 2016b) for English and Chinese corpus with 32k merge operations to address the unknown word issue.

[2] https://github.com/BYVoid/OpenCC
[3] https://github.com/fxsjy/jieba

2.3 Training of NMT models

To construct the parabank, we only need to translate the English sentences into Chinese with NMT, but we have trained two NMT models for the forward and reverse translation directions for back translation (Sennrich et al., 2016a) and dual scoring (§ 2.4).

We employ the Transformer translation model (Vaswani et al., 2017) for NMT, as it has achieved the state-of-the-art performance in MT evaluations (Akhbardeh et al., 2021). We first use the parallel data to train 2 NMT models. Then we use greedy decoding to construct synthetic parallel data by back-translating the monolingual data (Sennrich et al., 2016a; Edunov et al., 2018). The back-translation data is then mixed with the original parallel data (the monolingual sentences at the target side and the greedy decoding texts at the source side). We fine-tune the NMT models trained in the first step on the mixed data for another $300k$ steps for improved performance.

To obtain good translation quality, we adopted the Transformer Big setting with 1024 and 4096 as the embedding dimension and the number of hidden units of the feed-forward layer respectively, together with a 12-layer deep encoder (Bapna et al., 2018; Wang et al., 2019a; Wu et al., 2019; Wei et al., 2020b; Zhang et al., 2019a; Xu et al., 2020a; Li et al., 2020; Huang et al., 2020; Xiong et al., 2020; Mehta et al., 2021; Li et al., 2021; Xu et al., 2021b). Parameters were initialized under the Lipschitz constraint (Xu et al., 2020a) to ensure the convergence of deep encoders. Since these NMT models are used to translate tens of millions of sentences (monolingual data for back-translation and the MT training set for the construction of the parabank), we used a 6-layer decoder instead of a deeper one to preserve the decoding efficiency (Kasai et al., 2021; Xu et al., 2021a). The number of warm-up steps was set to $8k$. We used a batch size of around $25k$ target tokens achieved by gradient accumulation (Xu et al., 2020b), and trained the models for $300k$ steps, which takes about 50 hours to train a model on 4 Nvidia A100 GPUs. We averaged the last 20 checkpoints saved with an interval of $1,500$ training steps.

The newstest 2019 was used as the development set, and newstest 2020 and newstest 2021 as the test set. The beam size of the decoder was set to 4, and translation quality was evaluated by case-sensitive BLEU (Papineni et al., 2002) with the SacreBLEU

Model	Zh→En		En→Zh	
	newstest20	newstest21	newstest20	newstest21
NMT	30.53	24.74	42.38	32.85
BT-NMT	30.97	25.18	52.42	42.99

Table 2: BLEU scores of our NMT models on the WMT 20 and WMT 21 news translation test sets.

toolkit (Post, 2018; Bawden et al., 2020). Results are shown in Table 2.

Table 2 shows that our BT-NMT models can obtain comparably strong translation performance.

2.4 Paraphrase Generation

Language filtering We find that there are some English words in the Chinese part of the parallel data, which may affect the quality of the constructed parabank. To address this issue, we remove sentence pairs where a large percentage of English words appear in its Chinese sentence. Specifically, we check the percentage of English characters in Chinese sentences, and sentence pairs will be dropped if the proportion is larger than 60%.

Generating paraphrase candidates The English sentences are semantically consistent with the corresponding Chinese sentences in the parallel data. So we can obtain paraphrases of the original Chinese sentences by translating the English sentences into Chinese with MT.

We use the En→Zh BT-NMT model for the translation. For each English sentence, we use a beam size of 15 and collect all Chinese beam search candidates. Then, we pair each MT candidate with the corresponding original Chinese sentence to get a candidate Chinese paraphrase pair.

We find that En→Zh translation is more challenging than that for the construction of English parabanks although the NMT model obtains a high BLEU score for character-level evaluation (used for Chinese translations), and approaches like sampling/constrained-decoding (Post and Vilar, 2018) further drop the performance (by ∼5 BLEU), causing semantic changes. Hence, we put a higher priority on translation quality to ensure the semantic consistency without using diversity-oriented approaches, such as sampling and constrained decoding. We suggest that our work provides a valuable reference for the construction of many other languages' parabanks with MT when ensuring MT quality is a problem.

Edit-distance ratio filtering To effectively ensure the diversity of the parabank, we compute the edit-distance ratio (the edit distance divided by the length) between the beam search candidate and the corresponding original Chinese sentence, and use a minimum edit-distance ratio of 12% to filter the paraphrase pairs. We note that, as the parallel data are large, it is easy to further filter out a large subset with an edit-distance threshold larger than ours.

Dual scoring filtering The En→Zh model may leave some source tokens untranslated, leading to the under-translation issue (Tu et al., 2016). Measuring the round-trip translation consistency has been proven to be an effective way to address this and to improve the translation quality (Goto and Tanaka, 2017). Instead of selecting the beam search candidate with the highest decoding probability ($p_{forward}$), we also take the force decoding probability of the reverse model (Zh→En) $p_{reverse}$ into consideration. We re-rank the beam search candidates by summing the forward and reverse probabilities.

$$p_{dual} = p_{forward} + p_{reverse} \quad (1)$$

During filtering, we first select the candidate with the highest p_{dual} from beam search results for each remaining Chinese sentence. Then we derive the per-token probability of all instances of the dataset based on p_{dual}, and only retain ∼22M sentence pairs with the highest per-token probability to further ensure the quality, obtaining the final Chinese parabank, ParaZh-22M.

3 Evaluation of ParaZh-22M

We compare the constructed ParaZh-22M with two existing Chinese paraphrase datasets: PKU Paraphrase Bank (Zhang et al., 2019b) and Chinese Paraphrase from Quora (Wang et al., 2021).

PKU Paraphrase Bank Zhang et al. (2019b) construct the PKU parabank by extracting multiple

Corpus	Source materials	Size (pairs)	Length (words)	Domain
PKU Paraphrase Bank	One-to-many translation	509K	**23.05**	Literature
Chinese Paraphrase from Quora	English Quora	263K	9.80	Question
ParaZh-22M (ours)	One-to-one translation	**22M**	22.16	**General**

Table 3: Information and statistics of Chinese parabanks.

Chinese translations of the same source texts (written in English as well as other European languages). The sentence pairs are from literary work.

Chinese Paraphrase from Quora Wang et al. (2021) transfer English retelling corpus, Quora, to Chinese with machine translation engines.

3.1 Statistics

We provide the basic information of Chinese parabanks on source materials, domain, size (the number of sentence pairs), and the average sentence length (the number of Chinese words segmented by jieba) in Table 3.

Table 3 shows that: 1) ParaZh-22M is two orders of magnitude larger than the others, in terms of the number of paraphrases, it is 84 times as large as the Chinese Paraphrase from Quora (Wang et al., 2021) and 43 times as large as PKU Paraphrase Bank (Zhang et al., 2019b). 2) the average number of words of ParaZh-22M is similar to that of the PKU Paraphrase Bank, and ParaZh-22M has more words than the Chinese Paraphrase from Quora on average. And 3) as ParaZh-22M is constructed upon bilingual data which involve many domains and rich styles (for the use of 14.2M ParaCrawl data), it is more general than the other two paraphrase corpora (PKU Paraphrase Bank is constructed based on literature work while Chinese Paraphrase from Quora are translations of English Quora), and can be easily adapted to different domains.

We suggest that: 1) the large size of ParaZh-22M is crucial to support the training of large neural paraphraser models, 2) it is easy to filter out a large subset for the use of a special task given an edit-distance threshold, and 3) covering a wide range of domains makes the application of ParaZh-22M domain-agnostic, leading to robust performance.

3.2 Manual Evaluation

There lacks an ideal evaluation metric that takes both semantic consistency and diversity into account for paraphrasing. Semantic consistency, fluency and diversity are all important, while the diversity evaluation is normally against the consistency evaluation, e.g., a lower BLEU indicating higher diversity but lower semantic consistency (in MT). So we manually evaluate ParaZh-22M and PKU Paraphrase Bank (Zhang et al., 2019b) in terms of semantic consistency, literal fluency, and sentential diversity to measure their quality. We design our evaluation criteria following Wieting and Gimpel (2018); Wang et al. (2021), and specifics are shown in Table 4. For each evaluation criterion, we design 5 levels to distinguish the quality of sentence pairs.

We randomly sampled 800 sentence pairs from each dataset, and employed 8 native Chinese linguistic experts to rate them. Each sample is rated by 2 experts, and the final score is the average of their ratings. Results are shown in Table 5.

For the evaluation of ParaZh-22M, Table 5 shows that: 1) 93.9% of ParaZh-22M samples are strongly semantically consistent (with a score no less than 4, indicating that the semantic meaning of the sentence pair is nearly equivalent, or only may differ in some unimportant details). 2) 97.9% of ParaZh-22M samples are fluent (with at most one grammatical error), and 3) 97.9% of ParaZh-22M samples have at least one lexical variation.

Compared to the PKU Paraphrase Bank, ParaZh-22M achieves much higher scores in semantic consistency and literal fluency evaluation, while obtaining a slightly lower score in sentential diversity. We conjecture this might be because: 1) we pay more attention to optimizing the translation quality when constructing the parabank (§ 2), which gives the correctness (semantic consistency and fluency) a higher priority than the diversity, and 2) Zhang et al. (2019b) use one-to-many parallel data for the construction of the parabank, while we only use one-to-one translation data.

We evaluated the inter-annotator agreement with kappa (Artstein and Poesio, 2008), and obtained a kappa value of 0.87, suggesting that a high agreement is achieved with our evaluation criteria and our evaluation is reliable.

Score	Semantic Consistency	Literal Fluency	Sentential Diversity
5	Sentences have exactly the same meaning with all the same details.	The sentence pair has no grammatical error.	The sentences have more than one grammatical variation or more than two lexical variations.
4	Sentences are mostly equivalent, but some unimportant details can differ.	The sentence pair has one grammatical error.	The sentences have grammatical variation slightly.
3	Sentences are roughly equivalent, with some important information missing or that differs slightly.	The sentence pair has two grammatical errors.	The sentences have unchanged grammatical structure but two lexical variations.
2	Sentences are not equivalent, even if they share slight details.	The sentence pair has three grammatical errors.	The sentences have unchanged grammatical structure but one lexical variation.
1	The sentences are totally different.	The sentence pair has more than three grammatical errors.	The sentence pair has basically unchanged grammatical structure and lexical variation.

Table 4: Manual evaluation criteria of semantic consistency, literal fluency, and sentential diversity.

Score	**Semantic Consistency**		**Literal Fluency**		**Sentential Diversity**	
	Ours	PKU	Ours	PKU	Ours	PKU
= 5.0	69.1	34.3	82.4	72.0	56.9	65.3
$\geq$ 4.0	93.9	66.5	97.9	97.3	70.8	74.9
$\geq$ 3.0	98.4	87.8	99.9	99.5	84.0	87.3
$\geq$ 2.0	99.6	94.8	100.0	100.0	97.9	98.5
AVG score	**4.64±0.64**	3.89±1.11	**4.82±0.41**	4.72±0.50	4.11±1.19	**4.28±1.10**

Table 5: Manual evaluation results of our corpus and PKU Paraphrase Bank on semantic consistency, literal fluency, and sentential diversity. **Medium:** the cumulative percentages of samples with the scores. **Bottom:** the average score and the standard deviation of each criterion.

4 Using ParaZh-22M in Chinese NLP

We examine the effectiveness of data augmentation based on ParaZh-22M on a number of Chinese NLP tasks, including long/short text classification, natural language inference, keyword recognition from CLUE (a Chinese Language Understanding Evaluation benchmark) (Xu et al., 2020c) and the CCMT 2022 low-resource Chinese→Thai machine translation task, by paraphrasing the original training set.

4.1 Chinese Paraphraser

ParaZh-22M contains a large number of Chinese paraphrase examples, but cannot be directly used to augment the training sets of NLP tasks. To paraphrase arbitrary Chinese sentences, we train a Chinese paraphrase model, i.e., a Chinese paraphraser, on ParaZh-22M.

Like back translation, we use the machine translated Chinese sentences as the source input of the model, and the original Chinese sentences from the parallel data as the target when training the paraphraser on ParaZh-22M.

We used the same vocabulary and BPE as the Chinese part of NMT data (§ 2.2). We employed a base Transformer as the paraphraser. Specifically, we used 6 encoder and decoder layers, an embedding size of 512, 8 attention heads, a feed-forward layer of 2048 hidden units, and shared the encoder-decoder embeddings. The model was trained for $100k$ steps. The average of the last 5 checkpoints saved with an interval of $1,500$ training steps is served as the paraphraser.

	TNEWS					IFLYTEK				
Model	Baseline	PKU	Δ	Ours	Δ	Baseline	PKU	Δ	Ours	Δ
ALBERT-tiny	53.55	53.52	-0.03	**53.74**	**+0.19**	48.76	52.59	+3.83	**54.52**	**+5.76**
BERT-base	56.09	57.11	+1.02	**57.19**	**+1.10**	60.37	60.52	+0.15	**61.52**	**+1.15**
BERT-wwm-ext-base	56.77	57.55	+0.78	**57.69**	**+0.92**	59.88	59.75	-0.13	**61.79**	**+1.91**
avg	/	/	+0.59	/	**+0.74**	/	/	+1.28	/	**+2.94**

Table 6: Results (accuracy) on the validation sets of TNEWS and IFLYTEK tasks. "Δ" indicates the improvements over the baseline. "avg" is the average improvement of data augmentation over three baselines.

For fair comparison, we also trained a paraphraser under the same setting on the PKU parabank.

4.2 Text Classification

We conducted experiments on two text classification tasks of the CLUE benchmark (Xu et al., 2020c): TNEWS for short texts, and IFLYTEK for long texts.

The TNEWS task has 15 categories (finance, technology, sports, etc.), including 53.3k training instances and 10k validation data. The IFLYTEK task has 119 classes (food, car rental, education, etc.), with 12.1k training samples and 2.6k validation data.

We augmented the training data of these 2 tasks by paraphrasing the input sentences with our paraphraser, and constructed the synthetic data D_p by pairing paraphrases with the tag of the corresponding original sentence. We concatenated D_p with the original training set D_o as the augmented training set D_{aug}, and trained the same baseline model on the augmented training set.

We used ALBERT-tiny, BERT-base, BERT-wwm-ext-base as our baselines. ALBERT-tiny is a tiny version of ALBERT with only 4 layers and a hidden size of 312. BERT-base has 12 layers and uses a hidden size of 768. BERT-wwm-ext-base has the same configuration as BERT-base, but is pre-trained with whole word masking. We evaluated these models on the validation sets (as the test sets are not publicly available). Results are shown in Table 6.

Table 6 shows that: 1) paraphrasing based on both the PKU parabank and ParaZh-22M can lead to improvements on average, and 2) data augmentation based on ParaZh-22M leads to consistent and significant improvements over all baselines on both datasets, and brings about more accuracy improvements than based on the PKU parabank, showing the advantages of ParaZh-22M for both short and long text classification.

4.3 Natural Language Inference

We also examined the effects of paraphrasing based on ParaZh-22M on the natural language inference (NLI) task, and conducted experiments on CMNLI dataset. NLI aims to predict the relation (neutral, entailment, and contradiction) between sentence pairs. The CMNLI contains 391k training samples, and 12k validation instances.

We used the same baseline models described in § 4.2. For data augmentation, as each CMNLI training instance has a sentence pair, we investigate 4 cases: 1) augmentation by paraphrasing the first sentence ($S1$), 2) augmentation by paraphrasing the second sentence ($S2$), 3) augmentation by paraphrasing both sentences ($S12$), and 4) the combination of case 1 and case 2 ($S1 + S2$). We concatenated the paraphrased training set with the original training set. Results are shown in Table 7.

Table 7 shows that: even though paraphrasing based on the PKU parabank brings about more improvements in the S1+S2 and S2 settings with the ALBERT-tiny model than based on ParaZh-22M, data augmentation with ParaZh-22M leads to consistent and significant improvements over all baselines, and works better with larger models and stronger baselines (BERT-base and BERT-wwm-ext-base) than with the PKU parabank.

4.4 Keyword Recognition

The keyword recognition task requires the model to distinguish real keywords of paper abstracts from fake keywords. Chinese Scientific Literature (CSL) dataset (Xu et al., 2020c) contains Chinese paper abstracts and their real keywords from core journals of China, covering multiple fields of natural sciences and social sciences, with fake keywords generated through TF-IDF. CSL datasets provide 20k samples for training and 3k samples for validation.

We used the same baselines as in § 4.2. When paraphrasing the abstract, we performed beam de-

Model	Baseline	D_{aug}	S1+S2	Δ	S12	Δ	S1/S2	Δ
ALBERT-tiny	70.26	PKU	**73.67**	**+3.41**	72.01	+1.75	71.88/73.27	+1.62/+3.01
		Ours	73.07	+2.81	72.88	+2.62	72.56/72.45	+2.30/+2.19
BERT-base	79.47	PKU	79.55	+0.08	79.95	+0.48	79.90/79.81	+0.43/+0.34
		Ours	80.27	+0.80	**80.80**	**+1.33**	80.57/80.30	+1.10/+0.83
BERT-wwm-ext-base	80.92	PKU	79.98	-0.94	80.50	-0.42	80.04/80.08	-0.88/-0.84
		Ours	81.23	+0.31	81.21	+0.29	**81.28**/81.16	**+0.36**/+0.24

Table 7: Results (accuracy) on the validation set of CMNLI task.

Model	Baseline	PKU	Δ	Ours	Δ
ALBERT-tiny	74.34	76.66	+2.32	**77.20**	**+2.86**
BERT-base	79.63	79.03	-0.60	**80.90**	**+1.27**
BERT-wwm-ext-base	80.60	79.30	-1.30	**81.00**	**+0.40**
avg	/	/	+0.14	/	**+1.51**

Table 8: Results (accuracy) on the validation set of CSL task.

coding with a beam size of 15 with the paraphraser, and selected the beam search candidate that contains all corresponding keywords and has the highest decoding probability. We did not augment training instances when no beam search candidate contains the keywords. The synthetic training set D_p was then combined with the original training set D_o. Results are shown in Table 8.

Table 8 shows that: 1) data augmentation based on both the PKU parabank and ParaZh-22M can bring about improvements on average, 2) the accuracy improvements with ParaZh-22M are consistent and significant with all baselines, including in challenging cases (with BERT-base and BERT-wwm-ext-base), and are larger than with the PKU parabank, demonstrating the effectiveness of ParaZh-22M in challenging settings.

4.5 Low-Resource Machine Translation

We conducted experiments on the CCMT 2022 Chinese→Thai low-resource translation task. Its training set has 200k sentence pairs. As the evaluation does not release both the development set and the test set, we held out the last 2000 sentence pairs of the training set, and equally divided them into 2 parts for validation and test respectively. We paraphrased the Chinese sentences of the training data and paired with the corresponding Thai sentences.

We employed a 6-layer and a 12-layer Transformer as our baselines. Following Sennrich and Zhang (2019), we used an embedding dimension of 256, 4 attention heads, 1024 as the hidden dimension of the feed-forward layer, a dropout probability of 0.1, and applied 16k BPE operations enforced by sentence piece (Kudo and Richardson, 2018).

Model	Baseline	PKU	Δ	Ours	Δ
6-layer	7.15	6.35	-0.80	**7.75**	**+0.60**
12-layer	10.65	7.31	-3.34	**14.74**	**+4.09**

Table 9: Results (BLEU) on CCMT 2022 Zh→Th translation task.

We set the a beam size to 4, and evaluated translation quality via BLEU with the average of the last 5 checkpoints saved in an interval of 1,500 training steps. Results are shown in Table 9.

Table 9 shows that: 1) paraphrasing based on ParaZh-22M can lead to consistent and significant improvements in the low-resource translation task with both settings, and 2) the improvements with the 12-layer model (+4.09 BLEU) in the MT task without using pre-trained models are much larger than in CLUE tasks with pre-trained models and than with the 6-layer model with fewer parameters.

5 Related Work

Data augmentation via paraphrasing is beneficial for many NLP tasks, such as question answering (Dong et al., 2017), semantic parsing (Berant and Liang, 2014; Su and Yan, 2017) and machine translation (Cho et al., 2014; Khayrallah et al., 2020), especially in low-resource scenarios. Paraphrasing relies heavily on large scale paraphrase datasets.

Construction of English paraphrase data Most paraphrase corpus construction studies are for English (Suzuki et al., 2017; Mallinson et al., 2017). Given the development of NMT, Wieting and Gim-

pel (2018) leverage large amounts of bilingual parallel data to generate paraphrases via MT. Hu et al. (2019a) add lexical constraints during NMT decoding to enrich the diversity. Hu et al. (2019b) cluster over constrained sampling decoding candidates to generate diverse paraphrases. Compared to Wieting and Gimpel (2018) and Hu et al. (2019a,b), we assign a higher priority to the quality of machine translation than the diversity to ensure the translation correctness and the semantic consistency, without using constrained decoding or sampling that hampers the translation quality.

Chinese paraphrase corpus To our knowledge, existing Chinese parabanks are much smaller than large scale English parabanks. Zhang et al. (2019b) extract sentence-level paraphrases from multiple Chinese translations of the same source text, obtaining the PKU Paraphrase Bank of 509,832 paraphrase pairs. Wang et al. (2021) translate the question retelling Quora corpus into Chinese with multiple MT engines, and construct a Chinese parabank of 263,729 sentence pairs. Compared to their work, ParaZh-22M is much larger and involves many domains.

6 Conclusion

In this paper, we relieve the requirement of one-to-many translation data for the construction of Chinese parabank, and construct a Chinese parabank of 22M sentence pairs, ParaZh-22M, utilizing one-to-one sentence-level parallel data and MT technology. ParaZh-22M involves many domains and is over 40 times as large as the previous largest PKU Chinese Paraphrase Bank. Human evaluation on semantic consistency, fluency and sentential diversity shows the good quality of ParaZh-22M.

We test the effects of data augmentation via paraphrasing based on ParaZh-22M on a wide range of Chinese NLP tasks, including short/long text classification, natural language inference, keyword recognition, and low-resource machine translation. Our experiment results show that paraphrasing based on ParaZh-22M is able to achieve consistent and significant improvements over several baselines in all evaluations, demonstrating the contribution of ParaZh-22M to Chinese NLP tasks.

Acknowledgements

We thank anonymous reviewers for their insightful comments. Wenjie Hao, Hongfei Xu, Hongyin Zan and Lingling Mu acknowledge the support of the National Social Science Fund of China (Grant No. 17ZDA138 and Grant No. 14BYY096). Deyi Xiong is partially supported by the Key Research and Development Program of Yunnan Province (Grant No. 202203AA080004) and Zhejiang Lab (Grant No. 2022KH0AB01).

References

Farhad Akhbardeh, Arkady Arkhangorodsky, Magdalena Biesialska, Ondřej Bojar, Rajen Chatterjee, Vishrav Chaudhary, Marta R. Costa-jussa, Cristina España-Bonet, Angela Fan, Christian Federmann, Markus Freitag, Yvette Graham, Roman Grundkiewicz, Barry Haddow, Leonie Harter, Kenneth Heafield, Christopher Homan, Matthias Huck, Kwabena Amponsah-Kaakyire, Jungo Kasai, Daniel Khashabi, Kevin Knight, Tom Kocmi, Philipp Koehn, Nicholas Lourie, Christof Monz, Makoto Morishita, Masaaki Nagata, Ajay Nagesh, Toshiaki Nakazawa, Matteo Negri, Santanu Pal, Allahsera Auguste Tapo, Marco Turchi, Valentin Vydrin, and Marcos Zampieri. 2021. Findings of the 2021 conference on machine translation (wmt21). In *Proceedings of the Sixth Conference on Machine Translation*, pages 1–88, Online. Association for Computational Linguistics.

Ron Artstein and Massimo Poesio. 2008. Survey article: Inter-coder agreement for computational linguistics. *Computational Linguistics*, 34(4):555–596.

Marta Bañón, Pinzhen Chen, Barry Haddow, Kenneth Heafield, Hieu Hoang, Miquel Esplà-Gomis, Mikel L. Forcada, Amir Kamran, Faheem Kirefu, Philipp Koehn, Sergio Ortiz Rojas, Leopoldo Pla Sempere, Gema Ramírez-Sánchez, Elsa Sarrías, Marek Strelec, Brian Thompson, William Waites, Dion Wiggins, and Jaume Zaragoza. 2020. ParaCrawl: Web-scale acquisition of parallel corpora. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 4555–4567, Online. Association for Computational Linguistics.

Ankur Bapna, Mia Chen, Orhan Firat, Yuan Cao, and Yonghui Wu. 2018. Training deeper neural machine translation models with transparent attention. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 3028–3033. Association for Computational Linguistics.

Rachel Bawden, Biao Zhang, Lisa Yankovskaya, Andre Tättar, and Matt Post. 2020. A study in improving BLEU reference coverage with diverse automatic paraphrasing. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 918–932, Online. Association for Computational Linguistics.

Jonathan Berant and Percy Liang. 2014. Semantic parsing via paraphrasing. In *Proceedings of the 52nd Annual Meeting of the Association for Computational*

Linguistics (Volume 1: Long Papers), pages 1415–1425, Baltimore, Maryland. Association for Computational Linguistics.

Rahul Bhagat and Eduard Hovy. 2013. What is a paraphrase? *Computational Linguistics*, 39(3):463–472.

Mingda Chen, Qingming Tang, Sam Wiseman, and Kevin Gimpel. 2019. Controllable paraphrase generation with a syntactic exemplar. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 5972–5984, Florence, Italy. Association for Computational Linguistics.

Kyunghyun Cho, Bart van Merriënboer, Caglar Gulcehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. 2014. Learning phrase representations using RNN encoder–decoder for statistical machine translation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1724–1734, Doha, Qatar. Association for Computational Linguistics.

Li Dong, Jonathan Mallinson, Siva Reddy, and Mirella Lapata. 2017. Learning to paraphrase for question answering. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 875–886, Copenhagen, Denmark. Association for Computational Linguistics.

Zi-Yi Dou, Antonios Anastasopoulos, and Graham Neubig. 2020. Dynamic data selection and weighting for iterative back-translation. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 5894–5904, Online. Association for Computational Linguistics.

Sergey Edunov, Myle Ott, Michael Auli, and David Grangier. 2018. Understanding back-translation at scale. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 489–500, Brussels, Belgium. Association for Computational Linguistics.

Marzieh Fadaee and Christof Monz. 2018. Back-translation sampling by targeting difficult words in neural machine translation. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 436–446, Brussels, Belgium. Association for Computational Linguistics.

Isao Goto and Hideki Tanaka. 2017. Detecting untranslated content for neural machine translation. In *Proceedings of the First Workshop on Neural Machine Translation*, pages 47–55, Vancouver. Association for Computational Linguistics.

J. Edward Hu, Rachel Rudinger, Matt Post, and Benjamin Van Durme. 2019a. ParaBank: Monolingual bitext generation and sentential paraphrasing via lexically-constrained neural machine translation. In *Proceedings of AAAI*, volume 33, pages 6521–6528, Hawaii, USA.

J. Edward Hu, Abhinav Singh, Nils Holzenberger, Matt Post, and Benjamin Van Durme. 2019b. Large-scale, diverse, paraphrastic bitexts via sampling and clustering. In *Proceedings of the 23rd Conference on Computational Natural Language Learning (CoNLL)*, pages 44–54, Hong Kong, China. Association for Computational Linguistics.

Xiao Shi Huang, Felipe Perez, Jimmy Ba, and Maksims Volkovs. 2020. Improving transformer optimization through better initialization. In *Proceedings of the 37th International Conference on Machine Learning*, volume 119 of *Proceedings of Machine Learning Research*, pages 4475–4483. PMLR.

Mohit Iyyer, John Wieting, Kevin Gimpel, and Luke Zettlemoyer. 2018. Adversarial example generation with syntactically controlled paraphrase networks. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 1875–1885, New Orleans, Louisiana. Association for Computational Linguistics.

Jungo Kasai, Nikolaos Pappas, Hao Peng, James Cross, and Noah Smith. 2021. Deep encoder, shallow decoder: Reevaluating non-autoregressive machine translation. In *International Conference on Learning Representations*.

Huda Khayrallah, Brian Thompson, Matt Post, and Philipp Koehn. 2020. Simulated multiple reference training improves low-resource machine translation. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 82–89, Online. Association for Computational Linguistics.

Philipp Koehn, Hieu Hoang, Alexandra Birch, Chris Callison-Burch, Marcello Federico, Nicola Bertoldi, Brooke Cowan, Wade Shen, Christine Moran, Richard Zens, Chris Dyer, Ondřej Bojar, Alexandra Constantin, and Evan Herbst. 2007. Moses: Open source toolkit for statistical machine translation. In *Proceedings of the 45th Annual Meeting of the Association for Computational Linguistics Companion Volume Proceedings of the Demo and Poster Sessions*, pages 177–180, Prague, Czech Republic. Association for Computational Linguistics.

Kalpesh Krishna, John Wieting, and Mohit Iyyer. 2020. Reformulating unsupervised style transfer as paraphrase generation. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 737–762, Online. Association for Computational Linguistics.

Taku Kudo and John Richardson. 2018. SentencePiece: A simple and language independent subword tokenizer and detokenizer for neural text processing. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 66–71, Brussels, Belgium. Association for Computational Linguistics.

Bei Li, Ziyang Wang, Hui Liu, Quan Du, Tong Xiao, Chunliang Zhang, and Jingbo Zhu. 2021. Learning light-weight translation models from deep transformer. *Proceedings of the AAAI Conference on Artificial Intelligence*, 35(15):13217–13225.

Bei Li, Ziyang Wang, Hui Liu, Yufan Jiang, Quan Du, Tong Xiao, Huizhen Wang, and Jingbo Zhu. 2020. Shallow-to-deep training for neural machine translation. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 995–1005, Online. Association for Computational Linguistics.

Zichao Li, Xin Jiang, Lifeng Shang, and Qun Liu. 2019. Decomposable neural paraphrase generation. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 3403–3414, Florence, Italy. Association for Computational Linguistics.

Jonathan Mallinson, Rico Sennrich, and Mirella Lapata. 2017. Paraphrasing revisited with neural machine translation. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 1, Long Papers*, pages 881–893, Valencia, Spain. Association for Computational Linguistics.

Benjamin Marie, Raphael Rubino, and Atsushi Fujita. 2020. Tagged back-translation revisited: Why does it really work? In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 5990–5997, Online. Association for Computational Linguistics.

Sachin Mehta, Marjan Ghazvininejad, Srinivasan Iyer, Luke Zettlemoyer, and Hannaneh Hajishirzi. 2021. Delight: Deep and light-weight transformer. In *International Conference on Learning Representations*.

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. Bleu: a method for automatic evaluation of machine translation. In *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics*, pages 311–318, Philadelphia, Pennsylvania, USA. Association for Computational Linguistics.

Matt Post. 2018. A call for clarity in reporting BLEU scores. In *Proceedings of the Third Conference on Machine Translation: Research Papers*, pages 186–191, Brussels, Belgium. Association for Computational Linguistics.

Matt Post and David Vilar. 2018. Fast lexically constrained decoding with dynamic beam allocation for neural machine translation. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 1314–1324, New Orleans, Louisiana. Association for Computational Linguistics.

Rico Sennrich, Barry Haddow, and Alexandra Birch. 2016a. Improving neural machine translation models with monolingual data. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 86–96, Berlin, Germany. Association for Computational Linguistics.

Rico Sennrich, Barry Haddow, and Alexandra Birch. 2016b. Neural machine translation of rare words with subword units. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1715–1725, Berlin, Germany. Association for Computational Linguistics.

Rico Sennrich and Biao Zhang. 2019. Revisiting low-resource neural machine translation: A case study. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 211–221, Florence, Italy. Association for Computational Linguistics.

Yu Su and Xifeng Yan. 2017. Cross-domain semantic parsing via paraphrasing. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 1235–1246, Copenhagen, Denmark. Association for Computational Linguistics.

Jiao Sun, Xuezhe Ma, and Nanyun Peng. 2021. AESOP: Paraphrase generation with adaptive syntactic control. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 5176–5189, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Yui Suzuki, Tomoyuki Kajiwara, and Mamoru Komachi. 2017. Building a non-trivial paraphrase corpus using multiple machine translation systems. In *Proceedings of ACL 2017, Student Research Workshop*, pages 36–42, Vancouver, Canada. Association for Computational Linguistics.

Brian Thompson and Matt Post. 2020. Automatic machine translation evaluation in many languages via zero-shot paraphrasing. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 90–121, Online. Association for Computational Linguistics.

Zhaopeng Tu, Zhengdong Lu, Yang Liu, Xiaohua Liu, and Hang Li. 2016. Modeling coverage for neural machine translation. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 76–85, Berlin, Germany. Association for Computational Linguistics.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Ł ukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc.

Qiang Wang, Bei Li, Tong Xiao, Jingbo Zhu, Changliang Li, Derek F. Wong, and Lidia S. Chao. 2019a. Learning deep transformer models for machine translation. In *Proceedings of the 57th Conference of the Association for Computational Linguistics*, pages 1810–1822, Florence, Italy. Association for Computational Linguistics.

Shuo Wang, Yang Liu, Chao Wang, Huanbo Luan, and Maosong Sun. 2019b. Improving back-translation with uncertainty-based confidence estimation. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 791–802, Hong Kong, China. Association for Computational Linguistics.

Yasong Wang, Mingtong Liu, Yujie Zhang, Jin'an Xu, and Chen Yufeng. 2021. Research on the construction and application of paraphrase parallel corpus. *Acta Scientiarum Naturalium Universitatis Pekinensis*, 57(1):68–74.

Hao-Ran Wei, Zhirui Zhang, Boxing Chen, and Weihua Luo. 2020a. Iterative domain-repaired back-translation. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 5884–5893, Online. Association for Computational Linguistics.

Xiangpeng Wei, Heng Yu, Yue Hu, Yue Zhang, Rongxiang Weng, and Weihua Luo. 2020b. Multiscale collaborative deep models for neural machine translation. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 414–426, Online. Association for Computational Linguistics.

John Wieting and Kevin Gimpel. 2018. ParaNMT-50M: Pushing the limits of paraphrastic sentence embeddings with millions of machine translations. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 451–462, Melbourne, Australia. Association for Computational Linguistics.

Lijun Wu, Yiren Wang, Yingce Xia, Fei Tian, Fei Gao, Tao Qin, Jianhuang Lai, and Tie-Yan Liu. 2019. Depth growing for neural machine translation. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 5558–5563, Florence, Italy. Association for Computational Linguistics.

Ruibin Xiong, Yunchang Yang, Di He, Kai Zheng, Shuxin Zheng, Chen Xing, Huishuai Zhang, Yanyan Lan, Liwei Wang, and Tieyan Liu. 2020. On layer normalization in the transformer architecture. In *Proceedings of the 37th International Conference on Machine Learning*, volume 119 of *Proceedings of Machine Learning Research*, pages 10524–10533. PMLR.

Hongfei Xu, Qiuhui Liu, Josef van Genabith, Deyi Xiong, and Jingyi Zhang. 2020a. Lipschitz constrained parameter initialization for deep transformers. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 397–402, Online. Association for Computational Linguistics.

Hongfei Xu, Josef van Genabith, Qiuhui Liu, and Deyi Xiong. 2021a. Probing word translations in the transformer and trading decoder for encoder layers. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 74–85, Online. Association for Computational Linguistics.

Hongfei Xu, Josef van Genabith, Deyi Xiong, and Qiuhui Liu. 2020b. Dynamically adjusting transformer batch size by monitoring gradient direction change. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 3519–3524, Online. Association for Computational Linguistics.

Liang Xu, Hai Hu, Xuanwei Zhang, Lu Li, Chenjie Cao, Yudong Li, Yechen Xu, Kai Sun, Dian Yu, Cong Yu, Yin Tian, Qianqian Dong, Weitang Liu, Bo Shi, Yiming Cui, Junyi Li, Jun Zeng, Rongzhao Wang, Weijian Xie, Yanting Li, Yina Patterson, Zuoyu Tian, Yiwen Zhang, He Zhou, Shaoweihua Liu, Zhe Zhao, Qipeng Zhao, Cong Yue, Xinrui Zhang, Zhengliang Yang, Kyle Richardson, and Zhenzhong Lan. 2020c. CLUE: A Chinese language understanding evaluation benchmark. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 4762–4772, Barcelona, Spain (Online). International Committee on Computational Linguistics.

Peng Xu, Dhruv Kumar, Wei Yang, Wenjie Zi, Keyi Tang, Chenyang Huang, Jackie Chi Kit Cheung, Simon J.D. Prince, and Yanshuai Cao. 2021b. Optimizing deeper transformers on small datasets. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 2089–2102, Online. Association for Computational Linguistics.

Biao Zhang, Ivan Titov, and Rico Sennrich. 2019a. Improving deep transformer with depth-scaled initialization and merged attention. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 898–909, Hong Kong, China. Association for Computational Linguistics.

Bowei Zhang, Weiwei Sun, Xiaojun Wan, and Zongming Guo. 2019b. PKU paraphrase bank: A sentence-level paraphrase corpus for chinese. In *Natural Language Processing and Chinese Computing*,

pages 814–826, Cham. Springer International Publishing.

Michal Ziemski, Marcin Junczys-Dowmunt, and Bruno Pouliquen. 2016. The united nations parallel corpus v1.0. In *LREC 2016*.

ESimCSE: Enhanced Sample Building Method for Contrastive Learning of Unsupervised Sentence Embedding

Xing Wu[1,2,3], **Chaochen Gao**[1,2]*, **Liangjun Zang**[1], **Jizhong Han**[1], **Zhongyuan Wang**[3], **Songlin Hu**[1,2]†
[1]Institute of Information Engineering, Chinese Academy of Sciences, Beijing, China
[2]School of Cyber Security, University of Chinese Academy of Sciences, Beijing, China
[3]Kuaishou Technology, Beijing, China
{wuxing,gaochaochen,zangliangjun,hanjizhong,husonglin}@iie.ac.cn
wangzhongyuan@kuaishou.com

Abstract

SimCSE [1] adopts *dropout* as data augmentation and encodes an input sentence *twice* into two corresponding embeddings to build a positive pair. Since SimCSE is a Transformer-based encoder that directly encodes the length information of sentences through positional embeddings, the two embeddings in a positive pair contain the same length information. Thus, a model trained with these positive pairs is biased, tending to consider that sentences of the same or similar length are more similar in semantics. To alleviate it, we apply a simple but effective repetition operation to modify the input sentence. Then we pass the input sentence and its modified counterpart to the pre-trained Transformer encoder, respectively, to get the positive pair. Additionally, we draw inspiration from the computer vision community and introduce momentum contrast to enlarge the number of negative pairs without additional calculations. The proposed modifications are applied to positive and negative pairs separately, and build a new sentence embedding method, termed Enhanced SimCSE (ESimCSE). We evaluate the proposed ESimCSE on several benchmark datasets w.r.t the semantic text similarity (STS) task. Experimental results show that ESimCSE outperforms SimCSE by an average Spearman correlation of 2.02% on BERT-base. Our code are available at https://github.com/caskcsg/ESimCSE.

1 Introduction

Recently, researchers have proposed using contrastive learning to learn better unsupervised sentence embeddings (Wu et al.; Zhang et al., b; Liu et al., 2021; Gao et al., 2021; Yan et al., 2021). Contrastive learning aims to learn effective sentence embeddings based on the assumption that effective sentence embeddings should bring similar sentences closer while pushing away dissimilar ones. It generally uses various data augmentation methods (Shleifer, 2019; Wei and Zou, 2019; Wu et al., 2019) to generate different views for each sentence randomly, and assumes a sentence is semantically more similar to its augmented counterpart than any other sentence. Among these methods, the most representative one is SimCSE (Gao et al., 2021), which performs on par with previously supervised counterparts. SimCSE implicitly hypothesizes *dropout* acts as a minimal data augmentation method. Specifically, SimCSE composes N sentences in a batch and feeds each sentence to the pre-trained BERT *twice* with two independently sampled dropout masks. Then the embeddings derived from the same sentence constitute a "positive pair", while those derived from two different sentences constitute a "negative pair".

Length Diff	Avg. Similarity Diff
> 3	16.34
≤ 3	**18.18** (+1.84)

Table 1: The average similarity difference between the model (SimCSE-BERT) predictions and the normalized ground truths.

Using dropout as a minimal data augmentation method is simple and effective, but there is a weak point. SimCSE models are built on Transformer blocks, which will encode a sentence's length information through positional embeddings. In a positive pair, two embeddings are derived from the same sentence to contain the same length information. In contrast, in a negative pair, two embeddings in a negative pair are derived from two different sentences and generally contain different length information. Therefore, positive and negative pairs are different in their length information, acting as a feature to distinguish them. The semantic simi-

*The first two authors contribute equally.
†Corresponding author.
[1]We focus on unsupervised sentence embedding, so SimCSE in this article refers to unsupervised SimCSE.

Proceedings of the 29th International Conference on Computational Linguistics, pages 3898–3907
October 12–17, 2022.

Method	Text	Similarity
original sentence	I like this apple because it looks so fresh and it should be delicious.	1.0
random insertion	I **don't** like this apple because **but** it looks so **not** fresh and it should be **dog** delicious.	0.69
random deletion	I like this ~~apple~~ because it looks so ~~fresh~~ and it should be ~~delicious~~.	0.32
word repetition	I like **like** this apple because it looks so **so** fresh and **and** it should be delicious.	0.99
word repetition	I **I** like this apple **apple** because it looks **looks** so fresh **fresh** and it should be delicious **delicious**.	0.98

Table 2: An example of semantic similarity after different methods change a sentence's length.

larity model trained with these pairs can be biased, which probably considers that two sentences of the same or similar lengths are more similar in semantics. To confirm it, we evaluate on seven standard semantic textual similarity datasets with the SimCSE-BERT$_{base}$ model published by (Gao et al., 2021). We partition each STS test set into two groups based on whether the sentence pairs' length difference is ≤ 3. We calculate the similarity differences between the model predictions and the normalized ground truths for each group. As shown in Table 1, the average similarity difference of seven datasets is higher when the length difference is ≤ 3, which verifies our assumption. Comparison details on each dataset can refer to Table 7.

To alleviate this problem, we propose a simple but effective enhancement method to SimCSE. For each positive pair, we expect to change the length of a sentence without changing its semantic meaning. Existing methods to change the length of a sentence generally use random insertion and random deletion. However, inserting randomly selected words into a sentence may introduce extra noise, which will probably distort the meaning of the sentence; deleting keywords from a sentence will also change its semantics substantially. Such operations are detrimental to SimCSE learning, which is also discussed in a contemporaneous work (Chuang et al., 2022). Therefore, we propose a safer method, termed "word repetition", which randomly duplicates some words in a sentence. For example, as shown in Table 2, either random insertion or random deletion may generate a sentence that deviates far from the meaning of the original sentence. On the contrary, the method of "word repetition" maintains the meaning of the original sentence quite well.

Apart from the optimization above for positive pairs construction, we further explore how to optimize the construction of negative pairs. Since contrastive learning is carried out between positive pairs and negative pairs, theoretically, more negative pairs can lead to a better comparison between the pairs (Chen et al.). And thus, a potential optimization direction is to leverage more negative pairs, encouraging the model towards more refined learning. However, according to (Gao et al., 2021), larger batch size is not always a better choice. For example, for the SimCSE-BERT$_{base}$ model, the optimal batch size is 64, and other settings of the batch size will lower the performance. Therefore, we tend to figure out how to expand the negative pairs more effectively. In the community of computer vision, to alleviate the GPU memory limitation when expanding the batch size, a feasible way is to introduce the momentum contrast (He et al.), which is also applied to natural language understanding (Fang et al.). Momentum contrast allows us to reuse the encoded embeddings from the immediate preceding mini-batches to expand the negative pairs by maintaining a queue. It always enqueues the sentence embeddings of the current mini-batches and meanwhile dequeues the "oldest" ones. As the enqueued sentence embeddings come from the preceding mini-batches, we keep a momentum updated encoder by taking the moving average of its parameters and use the momentum encoder to generate enqueued sentence embeddings. Note that, we turn off *dropout* when using the momentum encoder, which can narrow the gap between training and prediction.

The above two optimizations are proposed separately for building positive and negative pairs. We finally combine both with SimCSE, termed Enhanced SimCSE (ESimCSE). We illustrate the schematic diagram of ESimCSE in Figure 1. The proposed ESimCSE is evaluated on the semantic

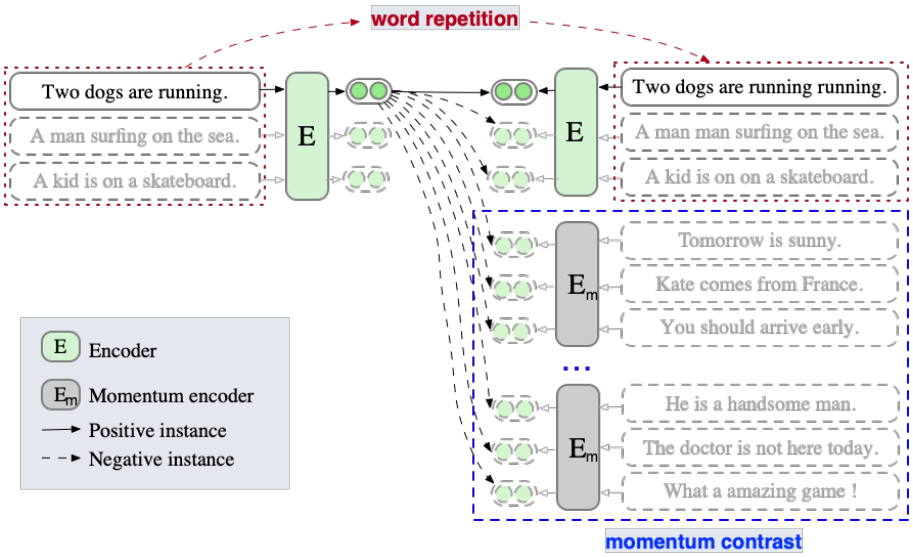

Figure 1: The schematic diagram of the ESimCSE method.

text similarity (STS) task with 7 STS-B test sets. Experimental results show that ESimCSE can improve the similarity measuring performance in different model settings over the previous state-of-the-art SimCSE. Specifically, ESimCSE gains an average increase of Spearman's correlation over SimCSE by +2.02% on BERT$_{base}$.

Our contributions can be summarized as follows:

- We observe that SimCSE constructs each positive pair with two sentences of the same length, which can bias the learning process. We propose a simple but effective "word repetition" method to alleviate the problem.
- We propose to use the momentum contrast method to increase the number of negative pairs involved in the loss calculation, which encourages the model towards more refined learning.
- We conduct extensive experiments on several benchmark datasets w.r.t semantic text similarity task. The experimental results well demonstrate that both proposed optimizations bring improvements to SimCSE.

2 Background: SimCSE

Given a set of paired sentences $\left\{x_i, x_i^+\right\}_{i=1}^m$, where x_i and x_i^+ are semantically related and will be referred to positive pairs. The core idea of SimCSE is to use identical sentences to build the positive pairs, i.e., $x_i^+ = x_i$. Note that in Transformer, there is a dropout mask placed on fully-connected layers and attention probabilities. And thus, the key ingredient is to feed the same input x_i to the encoder twice by applying different dropout masks z_i and z_i^+ and output two separate sentence embeddings to build a positive pair as follows:

$$\mathbf{h}_i = f_\theta\left(x_i, z_i\right), \mathbf{h}_i^+ = f_\theta\left(x_i, z_i^+\right) \quad (1)$$

With h_i and h_i^+ for each sentence in a mini-batch with batch size N, the contrastive learning objective w.r.t x_i is formulated as follows,

$$\ell_i = -\log \frac{e^{\text{sim}\left(\mathbf{h}_i, \mathbf{h}_i^+\right)/\tau}}{\sum_{j=1}^N e^{\text{sim}\left(\mathbf{h}_i, \mathbf{h}_j^+\right)/\tau}} \quad (2)$$

where τ is a temperature hyperparameter and $\text{sim}\left(\mathbf{h}_i, \mathbf{h}_i'\right)$ is the similarity metric, which is typically the cosine similarity function.

3 Proposed Enhanced SimCSE

In this section, we first introduce the word repetition method to construct better positive pairs. Then we introduce the momentum contrast method to expand negative pairs.

3.1 Word Repetition

The word repetition mechanism randomly duplicates some words/sub-words in a sentence. Here we take sub-word repetition as an example. Given a sentence s, after processing by a sub-word tokenizer, we get a sub-word sequence x =

$\{x_1, x_2, ..., x_N\}$, N being the length of sequence. We define the number of repeated tokens as

$$dup\_len \in [0, max(2, int(dup\_rate * N))] \quad (3)$$

where dup_rate is the maximal repetition rate, which is a hyperparameter. Then dup_len is a randomly sampled number in the set defined above, which will introduce more diversity when extending the sequence length. After dup_len is determined, we use uniform distribution to randomly select dup_len sub-words that need to be repeated from the sequence, which composes the dup_set as follows,

$$dup\_set = uniform([1, N], num = dup\_len) \quad (4)$$

For example, if the 1st sub-word is in dup_set, then sequence x becomes $x^+ = \{x_1, x_1, x_2, ..., x_N\}$. And different from SimCSE which passes x to the pre-trained BERT twice, E-SimCSE passes x and x^+ independently.

3.2 Momentum Contrast

The momentum contrast allows us to reuse the encoded sentence embeddings from the immediate preceding mini-batches by maintaining a queue of a fixed size. Specifically, the embeddings in the queue are progressively replaced. When the output sentence embeddings of the current mini-batch is enqueued, the "oldest" ones in the queue are removed if the queue is full. Note that we use a momentum-updated encoder to encode the enqueued sentence embeddings. Formally, denoting the parameters of the encoder as θ_{e} and those of the momentum-updated encoder as θ_{m}, we update θ_{m} in the following way,

$$\theta_{\mathrm{m}} \leftarrow \lambda\theta_{\mathrm{m}} + (1 - \lambda)\theta_{\mathrm{e}} \quad (5)$$

where $\lambda \in [0, 1)$ is a momentum coefficient parameter. Note that only the parameters θ_{e} are updated by back-propagation. And here we introduce θ_m to generate sentence embeddings for the queue, because the momentum update can make θ_{m} evolve more smoothly than θ_{e}. As a result, though the embeddings in the queue are encoded by different encoders (in different "steps" during training), the difference among these encoders can be made small.

With sentence embeddings in the queue, the loss function of ESimCSE is further modifed as follows,

$$\ell_i = -\log \frac{e^{\mathrm{sim}(\mathbf{h}_i, \mathbf{h}_i^+)/\tau}}{\sum_{j=1}^{N} e^{\mathrm{sim}(\mathbf{h}_i, \mathbf{h}_j^+)/\tau} + \sum_{m=1}^{M} e^{\mathrm{sim}(\mathbf{h}_i, \mathbf{h}_m^+)/\tau}} \quad (6)$$

where h_m^+ is denotes a sentence embedding in the momentum-updated queue, and M is the size of the queue.

4 Experiment

4.1 Experiment Setup

Our experimental language is English. For a fair comparison, our experimental setup mainly follows SimCSE. We use 1-million sentences randomly drawn from English Wikipedia for training[2]. The semantic textual similarity task measures the capability of sentence embeddings, and we conduct our experiments on seven standard semantic textual similarity (STS) datasets. STS12-STS16 datasets (Agirre et al., d,e,b,a,c) do not have train or development sets, and thus we evaluate the models on the development set of STS-B (Cer et al.) to search for better settings of the hyper-parameters. The SentEval toolkit[3] is used for evaluation, and Spearman correlation coefficient [4] is used to report the model performance. All the experiments are conducted on Nvidia 3090 GPUs.

4.2 Training Details

We start from pre-trained checkpoints of BERT(uncased) or RoBERTa(cased) using both the base and the large versions, and we add an MLP layer on top of the [CLS] representation to get the sentence embedding. We implement ESimCSE based on Huggingface's transformers package[5]. We train our models for one epoch using the Adam optimizer with the batch size $= 64$ and the temperature $\tau = 0.05$ in Eq. (3). The learning rate is set as 3e-5 for ESimCSE-BERT$_{base}$ model and 1e-5 for other models. The dropout rate is $p = 0.1$ for base models, $p = 0.15$ for large models. For the momentum contrast, we empirically choose a relatively large momentum λ

[2]https://huggingface.co/datasets/princeton-nlp/datasets-for-simcse/resolve/main/wiki1m_for_simcse.txt

[3]https://github.com/facebookresearch/SentEval

[4]https://en.wikipedia.org/wiki/Spearman%27s_rank_correlation_coefficient

[5]https://github.com/huggingface/transformers,version 4.2.1.

Model	STS12	STS13	STS14	SICK15	STS16	STS-B	SICK-R	Avg.
IS-BERT$_{base}$ △	56.77	69.24	61.21	75.23	70.16	69.21	64.25	66.58
CT-BERT$_{base}$ △	61.63	76.80	68.47	77.50	76.48	74.31	69.19	72.05
ConSERT$_{base}$ ♡	64.64	78.49	69.07	79.72	75.95	73.97	67.31	72.74
BERT$_{base}$-flow♢	63.48	72.14	68.42	73.77	75.37	70.72	63.11	69.57
SG-OPT-BERT$_{base}$ ♠	66.84	80.13	71.23	81.56	77.17	77.23	68.16	74.62
Mirror-BERT$_{base}$ ♯	69.10	81.10	73.00	81.90	75.70	78.00	69.10	75.40
SimCSE-BERT$_{base}$ ♣	68.40	82.41	74.38	80.91	78.56	76.85	72.23	76.25
ESimCSE-BERT$_{base}$	**73.40**	**83.27**	**77.25**	**82.66**	**78.81**	**80.17**	**72.30**	**78.27**
ConSERT$_{large}$ ♡	70.69	82.96	74.13	82.78	76.66	77.53	70.37	76.45
BERT$_{large}$-flow♢	65.20	73.39	69.42	74.92	77.63	72.26	62.50	70.76
SG-OPT-BERT$_{large}$ ♠	67.02	79.42	70.38	81.72	76.35	76.16	70.20	74.46
SimCSE-BERT$_{large}$ ♣	70.88	84.16	76.43	**84.50**	**79.76**	79.26	73.88	78.41
ESimCSE-BERT$_{large}$	**73.21**	**85.37**	**77.73**	84.30	78.92	**80.73**	**74.89**	**79.31**
Mirror-RoBERTa$_{base}$ ♯	66.60	82.70	74.00	82.40	79.70	79.60	69.70	76.40
SimCSE-RoBERTa$_{base}$ ♣	**70.16**	81.77	73.24	81.36	**80.65**	80.22	68.56	76.57
ESimCSE-RoBERTa$_{base}$	69.90	**82.50**	**74.68**	**83.19**	80.30	**80.99**	**70.54**	**77.44**
SimCSE-RoBERTa$_{large}$ ♣	72.86	83.99	75.62	84.77	**81.80**	81.98	71.26	78.90
ESimCSE-RoBERTa$_{large}$	**73.20**	**84.93**	**76.88**	**84.86**	81.21	**82.79**	**72.27**	**79.45**

Table 3: Sentence embedding performance on 7 semantic textual similarity (STS) test sets. ♣ : results from official published model by (Gao et al., 2021).♡ : results from (Yan et al., 2021). ♠ : results from (Kim et al., 2021). ♢ : results from (Li et al., 2020). △ : results are reproduced and reevaluated by (Gao et al., 2021). ♯ : results from (Liu et al., 2021)

.

= 0.995. In addition, following SimCSE's code, we evaluate the model every 125 training steps on the development set of STS-B and keep the best checkpoint for the final evaluation on test sets. We use sub-word repetition instead of word repetition, further discussed in the ablation study section.

4.3 Main Results

Table 3 shows the models' performance on seven semantic textual similarity (STS) test sets. We mainly select SimCSE for comparison, since it is the current state-of-the-art and shares the same setting as our approach. In addition, we also use IS-BERT (Zhang et al., a), CT-BERT (Carlsson et al., 2021), ConSERT (Yan et al., 2021), SG-OPT (Kim et al., 2021), BERT-flow (Li et al., 2020), Mirror-BERT (Liu et al., 2021) as baselines. It can be seen that ESimCSE improves the measurement of semantic textual similarity in different settings over SimCSE. Specifically, ESimCSE outperforms SimCSE by +2.02% on BERT$_{base}$, +0.90% on BERT$_{large}$, +0.87% on RoBERTa$_{base}$, +0.55% on RoBERTa$_{large}$, respectively.

Model	STS-B
SimCSE ♣	82.45
+ word repetition	84.09 (+1.64)
+ momentum contrast	83.98 (+1.53)
ESimCSE	**84.85** (+2.40)

Table 4: Improvement on STS-B development sets that word repetition or momentum contrast brings to SimCSE. ♣ : results from official published model by (Gao et al., 2021).

5 Ablation Study

This section investigates how different settings affect ESimCSE's performance. All results are compared on BERT$_{base}$ scale models and are evaluated on the development set of STS-B unless otherwise specified.

5.1 The Importance of Word Repetition and Momentum Contrast

We explore how much improvement it can bring to SimCSE when only using word repetition or momentum contrast. As shown in Table 4, either word

Length-extension Method	STS-B
+Inserting Stop-words	81.72
+Inserting [MASK]	83.08
+Inserting Masked Prediction	84.18
+Word Repetition	84.40
+Sub-word Repetition	**84.85**

Table 5: Effects of sentence-length-extension method.

repetition or momentum contrast can bring substantial improvements to SimCSE. It means that both proposed methods to enhance the positive pairs and negative pairs are effective. Better yet, these two modifications can be superimposed (ESimCSE) to get further improvements.

5.2 Effect of Sentence-Length-Extension Method

In addition to sub-word repetition, we also explore three other methods to increase sentence length:

- **Word Repetition** is similar to sub-word repetition, except that the repetition operation occurs *before* tokenization. For example, given a word "microbiology", word repetition will produce "microbiology microbiology", while sub-word repetition will produce "micro micro ##biology" or "micro ##biology ##biology".
- **Inserting Stop-words** inserts a random stop-word after the selected word instead of repeating the selected word.
- **Inserting [MASK]** inserts a [MASK] token after the selected word. We can regard [MASK] as a dynamic context-compatible word placeholder.
- **Inserting Masked Prediction** inserts a [MASK] token after the selected word and uses the masked language model to predict the top-1 substitution. The substitution is used to replace the inserted [MASK] token.

As shown in Table 5, sub-word repetition achieves the best performance, and word repetition can also bring a good improvement, which shows that more fine-grained repetition can better alleviate the bias brought by the length difference of positive pairs. Inserting [MASK] can also improve slightly, but inserting stop words will decrease the effect. Inserting masked prediction also brings a good improvement, but this method requires a pre-trained masked language model to predict replacements, bringing high additional computational overhead.

5.3 Batching Sentences of Similar Length in Training

Apart from sentence-length-extension methods, we explore whether batching sentences of similar length in training will alleviate the bias towards identical sequence length in inference. We divide training sentences into buckets by length and batch them within each bucket. We explore two different settings:

- We divide the training set into two coarse-grained buckets based on whether the sentence length is greater than buc_len, where $buc\_len \in [3, 8]$;
- We divide the training set by sentence length into 6 fine-grained buckets: $\{\leq 3, 4, 5, 6, 7, \geq 8\}$, which we use $buc\_len = 3 \sim 8$ for short.

We list the experimental results in Table 6. Dividing the training set into buckets does not bring significant improvements and even decreases in some settings. We believe that after being divided into buckets, shuffle can only be performed within a bucket, leading to an insufficient comparison in contrastive learning. In contrast, the effect of word repetition is much better.

5.4 The Relationship between The Similarity and Length Difference

We further explore the relationship between the similarity and length difference of sentence pairs on ESimcSE, compared with that of SimCSE in the Introduction. As STS12-STS16 datasets do not have train or development sets, and thus we evaluate the models on the test set of each dataset. We partition each STS test set into two groups based

buc_len	wr	3	4	5
STS-B	84.09	81.92	82.00	82.66
buc_len	6	7	8	$3 \sim 8$
STS-B	82.00	82.13	83.00	82.18

Table 6: Effects of different bucket lengths buc_len. "wr" means using word repetition method instead of bucketing sentences. "$3 \sim 8$" means fine-grained buckets setting: $\{\leq 3, 4, 5, 6, 7, \geq 8\}$.

Model	LD	STS12	STS13	STS14	SICK15	STS16	STS-B	SICK-R	Avg.
SimCSE	> 3	8.93	15.74	11.90	19.68	28.91	21.33	7.86	16.34
	≤ 3	9.29	22.81	19.53	19.92	24.08	22.12	9.53	18.18
ESimCSE	> 3	13.48	23.73	17.14	25.98	34.71	26.22	10.44	21.67
	≤ 3	12.52	28.56	24.13	24.17	29.32	25.63	12.35	22.38

Table 7: The difference between the model predicted cosine similarity and the true label on each dataset's test set. "LD" is short for length difference.

Model	Sim <q,s1 >	Sim <q,s2 >
SimCSE	26.39	27.07(**+0.68**)
ESimCSE	36.82	36.87(**+0.05**)

Table 8: Effect of repeated words on the average similarity of two sets

dup_rate	0.08	0.12	0.16	0.2
STS-B	83.5	83.62	82.01	83.01
dup_rate	0.24	0.28	0.32	0.36
STS-B	84.24	82.96	**84.85**	83.84

Table 9: Effects of repetition rate dup_rate.

on whether the sentence pairs' length difference is ≤ 3. Then we calculate the similarity differences between the model predictions and the normalized ground truths for each group. As listed in Table 7, ESimCSE significantly reduces the average similarity difference gap between > 3 and ≤ 3, from 1.84 to 0.71, alleviating the learning bias we mentioned in the Introduction.

5.5 Will Word Repetition Bring New Bias ?

We further explore whether word repetition will mislead the model to be more inclined to consider sentences with repeated overlaps are more similar. We conduct a detection experiment on wiki data with the following settings:

1. We randomly select a sentence as a query, such as q = "I like **this** apple because it **looks very** fresh"
2. We use the query to randomly recall a candidate sentence with 13%-17% overlap tokens, such as s1 = "**This** is a very tall tree and it **looks like** a giant"
3. We apply the word-repetition operation on the overlap tokens in the candidate sentence and produce a word-repeated sentence, such as s2 = "**This this** is a **very very** tall tree and it **looks looks** like a giant."
4. We calculate the similarity of <q, s1 >and <q, s2 >and compare them.

We experiment on 100 different query sentences and calculate their average similarity. As shown in Table 8, compared to the 0.68 increase of the SimCSE, ESimCSE-BERT only increased by 0.05. Therefore, word repetition does not bring a new bias to the learning process.

5.6 Effect of Hyperparameters

Repetition Rate To quantitatively study the effect of repetition rate on the model performance, we slowly increase the repetition rate parameter dup_rate from 0.08 to 0.36, with each increase by 0.04. As shown in Table 9, when $dup\_rate = 0.32$, ESimCSE achieves the best performance, a larger or smaller dup_rate will cause performance degradation, which is consistent with our intuition.

Momentum Queue Size The size of the momentum contrast queue determines the number of negative pairs involved in the loss calculation. We experiment with the queue size equals to different multiples of the batch size. The experimental results are listed in Table 10. The optimal result is reached when the queue size was 2.5 times the batch size. A smaller queue size will reduce the effect. This is intuitive because more negative pairs participate in the loss calculation to compare positive pairs more fully. But a too large queue size also reduces the effect. We guess that is because the negative pairs in the momentum contrast are generated by the past "steps" during training, and a larger queue will use the outputs of more outdated encoder models which are quite different from the current one. And thus that will reduce the reliability of the loss calculation.

Queue Size	STS-B
$1 \times batch\_size$	83.83
$1.5 \times batch\_size$	83.81
$2 \times batch\_size$	83.03
$2.5 \times batch\_size$	**84.85**
$3 \times batch\_size$	82.66

Table 10: Effects of queue size of momentum contrast.

5.7 Performance on Transfer Tasks

Following (Gao et al., 2021), we further evaluate ESimCSE on transfer tasks, to see the transferability of the sentence embeddings from ESimCSE. The transfer tasks include: MR (movie review) (Pang and Lee, 2005), CR (product review) (Hu and Liu, 2004), SUBJ (subjectivity status) (Pang and Lee, 2004) , MPQA (opinion-polarity) (Wiebe et al., 2005), SST-2 (binary sentiment analysis) (Socher et al., 2013), TREC (question-type classification) (Voorhees and Tice, 2000) and MRPC (paraphrase detection) (Dolan and Brockett, 2005). For more details, one can refer to SentEval[6]. As shown in Table 11, compared with the performance of SimCSE, ESimCSE slightly increases the transferability of embedding. As our optimizations are focused on semantic textual similarity tasks, the ability of ESimcse on transfer tasks remains stable relative to SimCSE.

6 Related Work

Unsupervised sentence representation learning has been widely studied. (Socher et al.; Hill et al.; Le and Mikolov) propose to learn sentence representation according to the internal structure of each sentence. (Kiros et al.; Logeswaran and Lee) predict the surrounding sentences of a given sentence based on the distribution hypothesis. (Pagliardini et al.) propose Sent2Vec, a simple unsupervised model allowing to compose sentence embeddings using word vectors along with n-gram embeddings. Recently, contrastive learning has been explored in unsupervised sentence representation learning and has become a promising trend (Zhang et al., b; Wu et al.; Meng et al.; Liu et al., 2021; Gao et al., 2021; Yan et al., 2021; Chuang et al., 2022). Those contrastive learning based methods for sentence embeddings are generally based on the assumption that a good semantic representation should be able to bring similar sentences closer while pushing away dissimilar ones. Therefore, those methods use various data augmentation methods to randomly generate two different views for each sentence and design an effective loss function to make them closer in the semantic representation space. Among these contrastive methods, the most related ones to our work are unsup-ConSERT (Yan et al., 2021) and unsup-SimSCE (Gao et al., 2021). ConSERT explores various effective data augmentation strategies(e.g., adversarial attack, token shuffling, Cutoff, dropout) to generate different views for contrastive learning and analyze their effects on unsupervised sentence representation transfer. Unsup-SimSCE, the current state-of-the-art unsupervised method uses only standard dropout as minimal data augmentation, and feed an identical sentence to a pre-trained model twice with independently sampled dropout masks to generate two distinct sentence embeddings as a positive pair. Unsup-SimSCE is very simple but works surprisingly well, performing on par with previously supervised counterparts. However, we find that SimCSE constructs each positive pair with two sentences of the same length, which can mislead the learning of sentence embeddings. So we propose a simple but effective method temed "word repetition" to alleviate it. We also propose to use the momentum contrast method to increase the number of negative pairs involved in the loss calculation, which encourages the model towards more refined learning.

7 Conclusion and Future Work

In this paper, we propose optimizations to construct positive and negative pairs for SimCSE and combine them with SimCSE, which is termed ESimCSE. Through extensive experiments, the proposed ESimCSE achieves considerable improvements on standard semantic text similarity tasks over SimCSE.

In the future, we will focus on designing a more refined objective function to improve the discrimination between different negative pairs. Also we will make attempt to optimize the performance on both semantic textual similarity tasks and transfer tasks.

[6] https://github.com/facebookresearch/SentEval

References

Eneko Agirre, Carmen Banea, Claire Cardie, Daniel Cer, Mona Diab, Aitor Gonzalez-Agirre, Weiwei Guo, Inigo Lopez-Gazpio, Montse Maritxalar, Rada

Model	MR	CR	SUBJ	MPQA	SST	TREC	MRPC	Avg.
SimCSE ♣	81.18	**86.46**	94.45	**88.88**	85.50	89.80	74.43	85.81
ESimCSE	**81.32**	86.22	**94.74**	88.74	**85.50**	**91.00**	**74.90**	**86.06**

Table 11: Results on transfer tasks of different sentence embedding models, in terms of accuracy. ♣ : results from (Gao et al., 2021).

Mihalcea, et al. a. Semeval-2015 task 2: Semantic textual similarity, english, spanish and pilot on interpretability.

Eneko Agirre, Carmen Banea, Claire Cardie, Daniel Cer, Mona Diab, Aitor Gonzalez-Agirre, Weiwei Guo, Rada Mihalcea, German Rigau, and Janyce Wiebe. b. Semeval-2014 task 10: Multilingual semantic textual similarity.

Eneko Agirre, Carmen Banea, Daniel Cer, Mona Diab, Aitor Gonzalez Agirre, Rada Mihalcea, German Rigau Claramunt, and Janyce Wiebe. c. Semeval-2016 task 1: Semantic textual similarity, monolingual and cross-lingual evaluation.

Eneko Agirre, Daniel Cer, Mona Diab, and Aitor Gonzalez-Agirre. d. Semeval-2012 task 6: A pilot on semantic textual similarity.

Eneko Agirre, Daniel Cer, Mona Diab, Aitor Gonzalez-Agirre, and Weiwei Guo. e. * sem 2013 shared task: Semantic textual similarity.

Fredrik Carlsson, Amaru Cuba Gyllensten, Evangelia Gogoulou, Erik Ylipää Hellqvist, and Magnus Sahlgren. 2021. Semantic re-tuning with contrastive tension. In International Conference on Learning Representations.

Daniel Cer, Mona Diab, Eneko Agirre, Inigo Lopez-Gazpio, and Lucia Specia. Semeval-2017 task 1: Semantic textual similarity-multilingual and cross-lingual focused evaluation. arXiv preprint arXiv:1708.00055.

Ting Chen, Simon Kornblith, Mohammad Norouzi, and Geoffrey Hinton. A simple framework for contrastive learning of visual representations. In International conference on machine learning.

Yung-Sung Chuang, Rumen Dangovski, Hongyin Luo, Yang Zhang, Shiyu Chang, Marin Soljačić, Shang-Wen Li, Wen-tau Yih, Yoon Kim, and James Glass. 2022. Diffcse: Difference-based contrastive learning for sentence embeddings. arXiv preprint arXiv:2204.10298.

William B Dolan and Chris Brockett. 2005. Automatically constructing a corpus of sentential paraphrases. In Proceedings of the Third International Workshop on Paraphrasing (IWP2005).

Hongchao Fang, Sicheng Wang, Meng Zhou, Jiayuan Ding, and Pengtao Xie. Cert: Contrastive self-supervised learning for language understanding. arXiv preprint arXiv:2005.12766.

Tianyu Gao, Xingcheng Yao, and Danqi Chen. 2021. Simcse: Simple contrastive learning of sentence embeddings. arXiv preprint arXiv:2104.08821.

Kaiming He, Haoqi Fan, Yuxin Wu, Saining Xie, and Ross Girshick. Momentum contrast for unsupervised visual representation learning. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition.

Felix Hill, Kyunghyun Cho, and Anna Korhonen. Learning distributed representations of sentences from unlabelled data. arXiv preprint arXiv:1602.03483.

Minqing Hu and Bing Liu. 2004. Mining and summarizing customer reviews. In Proceedings of the tenth ACM SIGKDD international conference on Knowledge discovery and data mining, pages 168–177.

Taeuk Kim, Kang Min Yoo, and Sang-goo Lee. 2021. Self-guided contrastive learning for bert sentence representations. arXiv preprint arXiv:2106.07345.

Ryan Kiros, Yukun Zhu, Russ R Salakhutdinov, Richard Zemel, Raquel Urtasun, Antonio Torralba, and Sanja Fidler. Skip-thought vectors. In Advances in neural information processing systems.

Quoc Le and Tomas Mikolov. Distributed representations of sentences and documents. In International conference on machine learning.

Bohan Li, Hao Zhou, Junxian He, Mingxuan Wang, Yiming Yang, and Lei Li. 2020. On the sentence embeddings from pre-trained language models. arXiv preprint arXiv:2011.05864.

Fangyu Liu, Ivan Vulić, Anna Korhonen, and Nigel Collier. 2021. Fast, effective, and self-supervised: Transforming masked language models into universal lexical and sentence encoders. arXiv preprint arXiv:2104.08027.

Lajanugen Logeswaran and Honglak Lee. An efficient framework for learning sentence representations. arXiv preprint arXiv:1803.02893.

Yu Meng, Chenyan Xiong, Payal Bajaj, Saurabh Tiwary, Paul Bennett, Jiawei Han, and Xia Song. Coco-lm: Correcting and contrasting text sequences for language model pretraining. arXiv preprint arXiv:2102.08473.

Matteo Pagliardini, Prakhar Gupta, and Martin Jaggi. Unsupervised learning of sentence embeddings using compositional n-gram features. arXiv preprint arXiv:1703.02507.

Bo Pang and Lillian Lee. 2004. A sentimental education: Sentiment analysis using subjectivity summarization based on minimum cuts. arXiv preprint cs/0409058.

Bo Pang and Lillian Lee. 2005. Seeing stars: Exploiting class relationships for sentiment categorization with respect to rating scales. arXiv preprint cs/0506075.

Sam Shleifer. 2019. Low resource text classification with ulmfit and backtranslation. arXiv preprint arXiv:1903.09244.

Richard Socher, Eric Huang, Jeffrey Pennin, Christopher D Manning, and Andrew Ng. Dynamic pooling and unfolding recursive autoencoders for paraphrase detection. Advances in neural information processing systems.

Richard Socher, Alex Perelygin, Jean Wu, Jason Chuang, Christopher D Manning, Andrew Y Ng, and Christopher Potts. 2013. Recursive deep models for semantic compositionality over a sentiment treebank. In Proceedings of the 2013 conference on empirical methods in natural language processing, pages 1631–1642.

Ellen M Voorhees and Dawn M Tice. 2000. Building a question answering test collection. In Proceedings of the 23rd annual international ACM SIGIR conference on Research and development in information retrieval, pages 200–207.

Jason Wei and Kai Zou. 2019. Eda: Easy data augmentation techniques for boosting performance on text classification tasks. arXiv preprint arXiv:1901.11196.

Janyce Wiebe, Theresa Wilson, and Claire Cardie. 2005. Annotating expressions of opinions and emotions in language. Language resources and evaluation, 39(2):165–210.

Xing Wu, Shangwen Lv, Liangjun Zang, Jizhong Han, and Songlin Hu. 2019. Conditional bert contextual augmentation. In International conference on computational science, pages 84–95. Springer.

Zhuofeng Wu, Sinong Wang, Jiatao Gu, Madian Khabsa, Fei Sun, and Hao Ma. Clear: Contrastive learning for sentence representation. arXiv preprint arXiv:2012.15466.

Yuanmeng Yan, Rumei Li, Sirui Wang, Fuzheng Zhang, Wei Wu, and Weiran Xu. 2021. Consert: A contrastive framework for self-supervised sentence representation transfer. arXiv preprint arXiv:2105.11741.

Zhang, Ruidan He, Zuozhu Liu, Kwan Hui Lim, and Lidong Bing. a. An unsupervised sentence embedding method by mutual information maximization. In Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing.

Yan Zhang, Ruidan He, Zuozhu Liu, Kwan Hui Lim, and Lidong Bing. b. An unsupervised sentence embedding method by mutual information maximization. arXiv preprint arXiv:2009.12061.

Measuring Robustness for NLP

Yu Yu * and **Abdul Rafae Khan** * and **Jia Xu** *
* School of Engineering and Science, Steven Institute of Technology, NJ 07030, USA
yyu50@stevens.edu, akhan4@stevens.edu, jxu70@stevens.edu

Abstract

The quality of Natural Language Processing (NLP) models is typically measured by the accuracy or error rate of a predefined test set. Because the evaluation and optimization of these measures are narrowed down to a specific domain like news and cannot be generalized to other domains like Twitter, we often observe that a system reported with human parity results generates surprising errors in real-life use scenarios. We address this weakness with a new approach that uses an NLP quality measure based on robustness. Unlike previous work that has defined robustness using Minimax to bound worst cases, we measure robustness based on the consistency of cross-domain accuracy and introduce the coefficient of variation and (ϵ, γ)-Robustness. Our measures demonstrate higher agreements with human evaluation than accuracy scores like BLEU on ranking Machine Translation (MT) systems.

1 Introduction

Evaluation criteria serve as learning objectives and model assessment standards and are crucial for NLP research. Conventional evaluation methods compute the accuracy or errors compared to the reference on a predefined test set, such as BLEU or TER. Since the evaluation results highly depend on the test set, they may not apply to real-world test inputs that come from an unknown distribution. For example, one may query a legal document while the system is trained on the news. Therefore, it is necessary to define a measure that can give an idea of how robust system performance will be on unseen test data so that we can predict whether our model is generalizable to new domains.

There has been influential investigation into defining robustness instead of accuracy alone (Bastani et al., 2016; Hein and Andriushchenko, 2017; Weng et al., 2018). Robustness of a machine learning model can be described as the characteristic of how accurate the model is in making its predictions when tested on a new (but similar) dataset. For example, one definition of robustness claims "If a testing sample is *similar* to a training sample, then the testing error is close to the training error" (Xu and Mannor, 2012). Such testing samples, which are samples dissimilar from training samples are known as adversarial examples. Many studies on robustness measures (Heinze-Deml and Meinshausen, 2017; Araujo et al., 2019; Carlini and Wagner, 2017) focus on the worst-case scenarios with adversarial inputs.

However, because the worst-case appears very infrequently, in particular, if artificially simulated (Wang et al., 2020; Jin et al., 2020; Alzantot et al., 2018), a worst-case analysis has the inherent problem that if the worst case is far from the typical, then the quantification assigns a numerical value to cases that occur rarely, like outliers. For instance, if a system accuracy has a very small variance in a million test cases but fails dramatically in one, then Minimax will label this system less robust than a system with a high accuracy variance. While focusing on the worst-case can be important for some areas of computer science and engineering like astronautics, but in our view, it is usually not desirable in the NLP context. In our view, robustness is a notion that should address typical behavior, not atypical and rare behavior. Note that if the worst-case differs a lot from the typical, a statistical notion of robustness should make this apparent. Additionally, a useful formalization should ignore atypical behavior of our system - because only focusing on a worst-case is not practical.

In this paper, we introduce novel definitions of robustness for NLP systems. Instead of defining robustness in terms of worst-case performance, we define it in terms of the typical behavior of the system and the consistency in their performance of the system. Under this definition, the more inconsistent the model predictions are, the less robust the

Proceedings of the 29th International Conference on Computational Linguistics, pages 3908–3916
October 12–17, 2022.

model is. Specifically, we simulate unknown test domains using leaving-one-out cross-validation to measure the variance of model accuracy on a set of test domains. More precisely, we introduce the coefficient of variation to quantify the variants of accuracy that is non-linear.

We further expand this notion to a probabilistic definition, called (ϵ, γ)-Robustness, where the higher probability of consistent behavior, the more robust the system. We provide statistical guarantees on the performance of an NLP model through the following assertion:

"I certify that, with high probability, my NLP model's performance will not change too much, given test sets from different domains and/or with various types of noise."

Both variation of coefficient and (ϵ, γ)-Robustness takes a new direction away from the worst-case result to a general consistency of the model quality. They are meta-evaluation methods that measure the consistency of user-defined quality metrics. Thus, any standard evaluation metrics can be applied to our paradigm. Our robustness measures are evaluated by comparing with human rankings on system outputs and by simulating unknown test domains using cross-validation. Our experimental results show that our robustness measures have higher agreements with human rankings on machine translation (MT) system submissions than BLEU scores.

In summary, the main contributions of this work include:

1. introducing a definition of robustness which is different from the typical notion of performance in worst-case scenarios;
2. developing a probabilistic definition of robustness based on Chebyshev's inequality;
3. experimenting on four different NMT models using cross-validation techniques to simulate the scenarios of unknown test data;
4. evaluating using human assessment on WMT submission systems;
5. carrying out human annotation experiments as a comparison with our measure;

The rest of the paper is as follows: In Section 2, we describe the three definitions of robustness. In Section 2.4, we connect the coefficient of variation with (ϵ, γ)-Robustness and show their relationship.

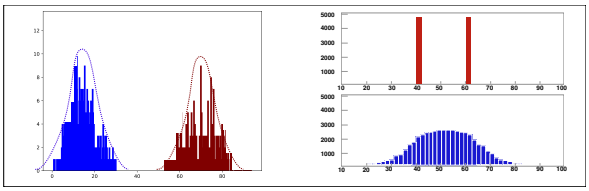

Figure 1: *Left:* Shortage of Variance; *Right:* Shortage of coefficient of variation.

Section 3 describes the six algorithms to compute robustness metrics we used during our experimentation. Section 4 describes our evaluation methods including cross-validation and manual comparison. Section 5 describes the experiments that evaluate our robustness measures. In Section 6, we discuss the previous literature on robustness in machine learning. Section 7 concludes the paper.

2 Definition of robustness

We define robustness as the consistency in the behavior of a machine-learned system. We think of it as the standard or typical behavior of the system. The more this behavior deviates from the typical, the less robust the system is defined to be. Notice that this definition does not give a notion of whether the performance of the system is good or bad. A system with consistent terrible performance is still a robust system under this definition. For example, in terms of MT, this definition becomes the consistency in translation performance for a trained MT system.

2.1 Variance

We take the NLP model accuracy (e.g, BLEU score) as a random variable. We introduce measuring the variance of, for instance, the BLEU score to indicate the consistency of the translation quality over the combination of various test sets. This random variable will give us a value that quantifies how stable is a translation system over different test sets.

2.2 Coefficient of Variation

However, the same variance value measured on the datasets with different means will carry a different meaning. For example, as left of Figure 1 shows, the variance measured on a dataset which has the mean of 10% in the BLEU score indicates a much higher inconsistency than that with the mean of 70%. Therefore, we can scale the variance by the mean, and finally, use the coefficient of the varia-

tion to measure the consistency over the accuracy scores across test sets in the dataset pool. The coefficient of variation is a scaled variance. Nonetheless, the variance or the coefficient of variation is not sufficient to express the consistency of the accuracy. For example, as right of Figure 1 shows, the accuracy can only have two values: 0 and 1 but have the same variance as values following the normal distribution. Thus looking at the distribution itself is crucial to decide on the robustness of a model.

2.3 (ϵ, γ)-Robustness

We follow the direction of the probabilistic robustness (Xu and Mannor, 2012) and introduce the notion of (ϵ, γ)-robust to consider all cases.

Briefly speaking, we want to measure the probability of upper bounding the NLP model accuracy gap between any test set and their average.

We call an NLP system (ϵ, γ)-robust, if for every test set drawn from a distribution D, its prediction error (or accuracy) X is centered around the mean error (or accuracy) μ, which is bounded through a parameter ϵ with a probability of $1 - \frac{\sigma^2}{\epsilon^2} \cdot \gamma$.

$$Pr[|X - \mu| < \epsilon] = 1 - \frac{\sigma^2}{\epsilon^2} \cdot \gamma \quad (1)$$

This definition is a relaxation of Chebyshev's inequality by adding $\gamma \in [0, 1]$. In the above formulation, the lower the value of γ, the more robust is the system. $1 - \gamma$ value indicates how much tighter we can bound the prediction accuracy around its mean than the Chebyshev's bound. A robust system can be provided with a tighter bound, while the Chebyshev in Equation 2 bounds a fragile system. Below is the inference.

$$Pr[|X - \mu| \geq \epsilon] \leq \frac{\sigma^2}{\epsilon^2} \quad (2)$$

$$\Leftrightarrow 1 - Pr[|X - \mu| \geq \epsilon] \geq 1 - \frac{\sigma^2}{\epsilon^2} \quad (3)$$

$$\Leftrightarrow Pr[|X - \mu| < \epsilon] \geq 1 - \frac{\sigma^2}{\epsilon^2} \quad (4)$$

Chebyshev's inequality in Equation 2 provides an upper bound to the probability that the difference between X and the mean will exceed a given threshold. If we put "$1-$" in front of both sides, we have Equation 3, thus Equation 4, which shows a lower bound of the difference between X and μ.

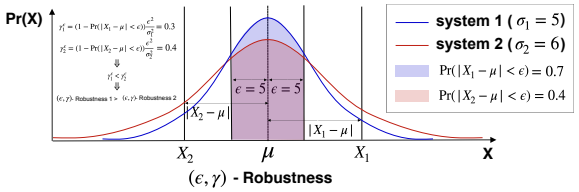

Figure 2: Illustration of the (ϵ, γ)-Robustness

Algorithm 1 Plot (ϵ, γ)-Robustness

Input: translation accuracy (e.g. BLEU or human eval) of each sentence of a test set on a given translation model

Output: 100 (ϵ, γ) values

1: **for** ϵ in 1 to 100 **do**
2: output $\gamma := (1 - Pr[|X - \mu| = \epsilon]) \cdot \frac{\epsilon^2}{\sigma^2}$
3: **end for**
4: **return**

In robustness measure, we are interested in coming up with a bound tighter than the Chebyshev's bound. To make it scalable and interpretable, we can add $\gamma \in [0, 1]$ to express how much tighter bound we can provide than Chebyshev's. If $\gamma = 1$ then we have the worst case, the system is not robust at all; if γ is approaching to 0, then it is getting very tightly bounded.

Therefore, we introduce γ **to be an indicator of how robust a system is**. ϵ is a parameter that we can explore. Algorithm 1 shows the algorithm of plotting the ϵ, γ-Robustness. γ relates to the probability that accuracy X is within a given threshold, and ϵ controls the width of such threshold, as illustrated in the example of Figure 2.

2.4 Relating the (ϵ, γ)-Robustnes to the coefficient of variation

ϵ is a hyper-parameter related to our robustness metrics. We show the (ϵ, γ)-Robustnes can be connected with the coefficient of variatioin by simply tuning ϵ.

Replace ϵ by ϵ', $\epsilon = \epsilon' \cdot \frac{\sigma}{\mu}$ to Chebyshev's inequality:

$$Pr[|X - \mu| \geq \epsilon] \leq \frac{\sigma^2}{\epsilon^2} \quad \text{Eq. 2}$$

$$\Rightarrow Pr[|X - \mu| \geq \epsilon' \cdot \frac{\sigma}{\mu}] \leq \frac{\sigma^2}{\epsilon'^2 \cdot \frac{\sigma}{\mu}^2} \quad (5)$$

$$\Rightarrow Pr[|X - \mu| \geq \epsilon' \cdot \frac{\sigma}{\mu}] \leq \frac{\mu^2}{\epsilon'^2} \quad (6)$$

Replace ϵ by ϵ', $\epsilon = \epsilon' \cdot \frac{\sigma}{\mu}$ to the $\epsilon - \gamma$ robust definition

$$Pr[|X - \mu| < \epsilon' \cdot \frac{\sigma}{\mu}] = 1 - \frac{\mu^2}{\epsilon'^2} \cdot \gamma \quad (7)$$

3 Robustness metrics

Using our definition of robustness (see Section 2), we create three different robustness metrics. The first metric (Algorithm 2) computes the variance for all the samples in the test pool. The second metric (Algorithm 3) scales variance by the mean of all samples. The third metric (Algorithm 4) computes the (ϵ, γ)-Robustness given a pre-defined parameter ϵ.

Algorithm 2 Robustness Metrics I: Variance

Require: Error function $\epsilon(\cdot)$; a test set pool T containing N samples (ts).
$V(T) = \frac{1}{I} \sum_i \{\epsilon(t_i; s, A) - \frac{1}{I} \sum_j \epsilon(t_j; s, A)\}^2$

Algorithm 3 Robustness Metrics II: Coefficient of Variation

Require: Error function $\epsilon(\cdot)$; a test set pool T containing N samples (ts).
$COV(T) = \frac{\frac{1}{I} \sum_i \{\epsilon(t_i;s,A) - \frac{1}{I} \sum_j \epsilon(t_j;s,A)\}^2}{\frac{1}{I} \sum_j \epsilon(t_j;s,A)}$

Algorithm 4 Robustness Metrics III: (ϵ, γ)-Robustness

Require: Error function $\epsilon(\cdot)$; a test set pool T containing N samples (ts), hyper-parameter ϵ.
$\mu = \frac{1}{I} \sum_i \epsilon(t_i; s, A)$
$\sigma = \sqrt{\frac{\sum (t_i - \mu)^2}{I-1}}$
$Pr[|t_i - \mu| < \epsilon] = \frac{|\{t_j | t_j - \mu < \epsilon\}|}{I}$
$(\epsilon, \gamma) - Robustness(T) = (1 - Pr[|t_i - \mu| = \epsilon]) \cdot \frac{\epsilon^2}{\sigma^2}$

We have a collection of test samples, where each test sample is t_i, and there are I many test samples in the test pool. An error function $\epsilon(t; s, A)$ indicates the translation error (1-accuracy) on a test sample t_i on s according to evaluation criterion A. The variance measures the "consistency" of the translation accuracy among all the test samples.

Bootstrapping Considering the test scores on one test sample can be unstable and inaccurate, we create a more robust method to compute robustness by bootstrapping subsamples from the entire test pool. We randomly subsample the test pool into M bootstraps and then compute the average robustness score (variance/coefficient of variation/(ϵ, γ)-Robustness) across bootstraps.

Algorithm 5 Robustness Metrics by Bagging

Require: Error function $\epsilon(\cdot)$; a test set pool T containing I samples: $t_1 \cdots t_I$; block size b, each block $B_1 \cdots B_M$ with number of blocks M, universe size N.
Initialize m empty blocks.
for $b\_ = 1$ to $b \cdot M$ **do**
  choose L at random from the set of blocks with current min number of elements
  S : set of elements in the universe not in L
  $L = L \cup t$, where $t \in S$ chosen uniformly at random
end for
for $m \in M$ **do**
  Compute $R(B_m)$ according to Algorithm 2/ Algorithm 3/Algorithm 4:
  $R(B_m) = V(B_m)/COV(B_m)/(\epsilon, \gamma) - Robustness(B_m)$
end for
$AV(T) = \frac{1}{M} \sum_m R(B_m)$

In practice, random sampling requires the times of bootstrapping to be relatively large to achieve a thorough coverage of the test pool. To complement, we create two bootstrapping methods to selectively design the subsampled bootstraps. First, for each bootstrap, we randomly sample from the elements not present in the current bootstrap, which is described in Algorithm 5. Second, we subsample from the set of elements with least sampled frequency (Papakonstantinou et al., 2014), which is described in Algorithm 6.

4 Evaluation of the robustness measures

4.1 Cross-validation

We use the leave-one-out error stability to show that the robustness measure can be generalized if we exclude a left-out test set from all four datasets where we measure the robustness. More precisely, for a given translation model, we randomly select one leaving-one-out test set and then combine all other tests to compute the variance, the mean, and the coefficient of variation on the left-out datasets, as shown in Figure 3.

Algorithm 6 Robustness Metrics by Design Bagging

Require: Error function $\epsilon(\cdot)$; a test set pool T containing I samples: $t_1 \cdots t_I$; block size b, each block $B_1 \cdots B_M$ with number of blocks M, universe size N.

Initialize m empty blocks.

for $b\_ = 1$ to $b \cdot M$ **do**

choose L at random from the set of blocks with current min number of elements

S : set of elements in the universe not in L **that appear least frequently**

$L = L \cup t$, where $t \in S$ chosen uniformly at random

end for

for $m \in M$ **do**

Compute $R(B_m)$ according to Algorithm 2/ Algorithm 3/Algorithm 4:

$R(B_m) = V(B_m)/COV(B_m)/(\epsilon, \gamma) - Robustness(B_m)$

end for

$AV(T) = \frac{1}{M} \sum_m R(B_m)$

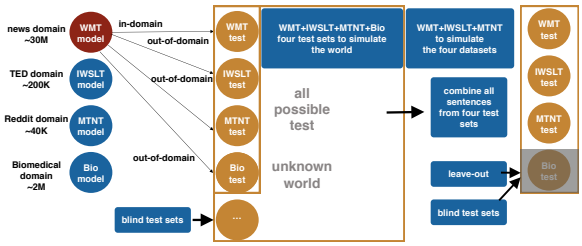

Figure 3: Leaving-one-out to verify robustness metrics.

4.2 Correlation with manual evaluation: pair-wise system comparison

Human evaluation is another way to evaluate the robustness measure. More precisely, we have human linguists come up with translation test sentences and evaluate the consistency of the translation model 1 (θ) and translation model 2 (θ'). The human evaluators can ask as many as translation sentences until she/he decides on the ranking of the performance between model 1 and model 2. A perfect robustness estimator ρ would satisfy that the human ranking of the robustness of two NLP systems is the same as the ranking by our robustness measures:

$$\rho_{human}(\theta) < \rho_{human}(\theta') \iff \rho(\theta) < \rho(\theta') \quad (8)$$

In other words, we do not necessarily need the actual value of ρ to verify that it gives us enough information to compare the two models. There are some problems with this approach, though, not least because we need to evaluate this on multiple models.

5 Experiments

Data & Tools We evaluate our robustness metrics in machine translation and sentiment classification. For machine translation, we use four different English-French NMT models trained on WMT'14, Biomedical'18, ISLWT'17, and MTNT'18 datasets. The models are trained using 35M, 2M, 200K, and 40K sentences, respectively. The WMT'14 model is the pre-trained model provided by Facebook research (Ott et al., 2019) and the remaining were trained in house using ConvS2S Toolkit (Gehring et al., 2017) till convergence. The development data used for the in house trained models include Khresmoi for Biomedical, test2014 and test2015 for IWSLT, and MTNT'18 development for MTNT. The four different test data were also from the same domains, including WMT (newstest14), Biomedical (EDP2018), IWSLT (test2017), and MTNT (MTNT2018).

For sentiment classification, we use Amazon product review dataset (Blitzer et al., 2007). Specifically, we train four models using pre-processed balanced reviews from DVD, kitchen, electronics and books domains where each review is labeled with 0 or 1. We randomly split 15% of the sentences as validation set and the rest as training set. We evaluate the classification model on unprocessed reviews from 21 domains different from the training domain, and compute the mean, var and coefficient of variance of these 21 test accuracy scores. Then we leave the apparel, baby, beauty, grocery and music domain out to compute the mean, variance, and the coefficient of variation on other 20 domains. Different from machine translation that uses sentence level performance score (i.e. BLEU) to compute robustness, we use domain/document level performance score (i.e. accuracy) to compute robustness metrics in sentiment classification.

Results As addressed in Section 3, We carry out three different robustness experiments corresponding to robustness metrics with no sampling method, robustness by bagging, and robustness by design bagging, respectively. For every experiment, we use each of the training models and translated all the four test sets. We calculate the sentence level BLEU scores and use one of the robustness metrics

to compute the change in the BLEU scores. We use a combined test data from all the four domains. We want our model of robustness measure to work on any test domain. In order to simulate a blind test domain, we apply leave-one-out testing, where we leave out one domain from the four domains. Therefore we calculate the mean, variance, and the coefficient of variation on the five testing environments.

Results in Table 1 and Table 2 use all training samples to measure. The rest four tables experiment with bagging techniques on sub-sampled test data where we create 30 bootstraps each containing 60% of the data. We calculate the mean and variance of each of the bootstrap and finally average values of these measures to calculate the coefficient of variation. For machine translation, we can observe that for models, including WMT, the change in the coefficient of variation when testing on different leave-one-out scenarios, does not change a lot compared to IWSLT. For sentiment classification, the model trained on kitchen domain has the lowest coefficient of variation compared to all other three models, and the coefficient of variation of the model trained on electronics domain varies largely when testing on different leave-one-out scenarios. Table 3 and table 4 use the bagging algorithm (Breiman, 1996) to make the intersection between bootstraps as less as possible. The mean, variance, and coefficient of variation were calculated the same way as in traditional bagging experiment.

Results using design bagging are mentioned in Table 5 and Table 6. A similar trend is observed where the difference in coefficient of variation across different leave-one-out experiments is the smallest for WMT and is much higher for MTNT.

To compute our definition of (ϵ, γ)-Robustness, we used the same setup of four different trained models and four different test domains. For each possible value of ϵ (between 0 and 100), we try to find the value of γ, which satisfies the Equation 2. Figure 4 shows the normalized γ values for each corresponding epsilon value for the four models. We can observe the γ values for WMT are much smaller than the values for other models, and the values for MTNT are the highest. This shows that the WMT model is the most robust among all models, and MTNT is the least robust. Similarly, our (ϵ, γ)-Robustness can be applied with human evaluation scores, as shown in Figure 5.

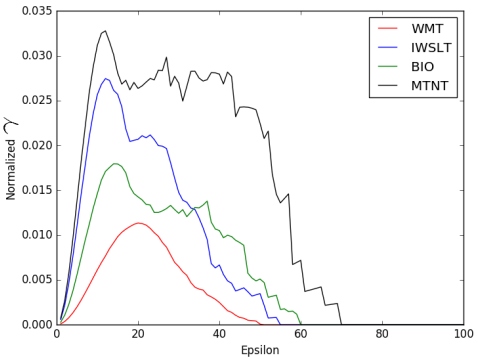

Figure 4: Normalized (ϵ, γ)-Robustness plot on our models.

Based on these four NMT systems in Figure 5, we rank them using (ϵ, γ)-Robustness and find the rank 100% agrees with the rank given by human, while the rank given by the BLEU score 75% agrees to human, as shown in Table 7. This suggests that (ϵ, γ)-Robustness evaluates robustness of systems better than accuracy-based metrics (i.e. BLEU).

Finally, as in Section 4.2, to better compare our method, we conduct a human evaluation for pairwise comparison of the models. We create a small web-based application where the human annotator is assigned two NMT models selected at random. The human annotators do not know any details about the model or the training data used for each model. She/He can only use these models to get two translation outputs for a given input sentence. This step can be repeated as many times as possible until the human annotator decides which model is more "consistent" in its translations. Table 8 mentions the pair-wise results for the four models using the human annotators. For comparison, we have also mentioned the coefficient of variation and the (ϵ, γ)-Robustness results. We observe that only one out of the six scenarios is different (16.66% error rate) for the human annotators and our robustness metrics.

6 Related Work

There have been substantial amount of work trying to improve the robustness of NLP models. Among those, a majority of focus lies on the vulnerability of NLP models to input perturbations, such as generating adversarial examples. For instance, Wang et al. (2020) uses controllable attributes irrelevant to task labels to generate diverse adversarial texts. Li et al. (2020) uses pre-trained masked language

Model	Leave-One-Out Test Set																	
	None			WMT'15			IWSLT'17			MTNT'19			BIO'18			MTNT'18		
	Mean	Var	COR	Mean	Var	COR	Mean	Var	COR	Mean	Var	COR	Mean	Var	COR	Mean	Var	COR
WMT'14	32.52	723.56	0.83	31.33	684.75	0.84	32.35	735.06	0.84	33.86	714.98	0.79	32.51	747.96	0.84	32.52	723.56	0.83
IWSLT'17	10.35	206.43	1.39	10.55	211.85	1.38	8.71	171.72	1.50	11.44	219.58	1.30	10.6	215.8	1.39	10.35	206.43	1.39
MTNT'19	6.97	173.38	1.89	7.19	179.26	1.86	6.2	156.24	2.02	6.73	166.43	1.92	7.61	186.79	1.80	6.97	173.38	1.89
BIO'18	15.36	330.83	1.18	15.7	328.59	1.15	15.37	339.74	1.20	15.89	327.4	1.14	14.66	327.51	1.23	15.36	330.83	1.18
APPERTIUM	1.83	49.20	3.83	1.68	44.11	3.95	1.77	48.53	3.94	2.04	52.74	3.56	1.72	46.72	3.98	1.95	53.79	3.76

Table 1: Robustness Metrics I & II in machine translation. Sentence level BLEU scores when a single test set is left out. *Var* is variance of BLEU scores and *COR* is the coefficient of variance.

Model	Leave-One-Out Test Set																	
	None			apparel			baby			beauty			grocery			music		
	Mean	Var	COR	Mean	Var	COR	Mean	Var	COR	Mean	Var	COR	Mean	Var	COR	Mean	Var	COR
dvd	64.33	2.04	3.18	65.02	2.05	3.16	64.91	2.08	3.21	64.13	2.16	3.37	63.76	2.07	3.25	65.09	2.08	3.19
books	64.33	2.07	3.23	65.10	2.06	3.17	64.95	2.11	3.25	64.04	2.09	3.26	63.70	2.03	3.19	65.03	2.04	3.14
kitchen	58.94	0.60	1.02	59.28	0.59	1.0	59.25	0.62	1.05	59.06	0.64	1.08	58.44	0.60	1.02	59.34	0.63	1.06
electronics	64.45	2.10	3.26	64.94	2.01	3.10	64.91	2.10	3.23	64.17	2.15	3.36	63.77	2.10	3.30	65.03	2.06	3.17

Table 2: Robustness Metrics I & II in sentiment classification. Domain level accuracy scores when a single test set is left out. *Var* is variance of accuracy scores and *COR* is the coefficient of variance.

Model	Leave-One-Out (Bagging)																	
	None			WMT'15			IWSLT'17			MTNT'19			BIO'18			MTNT'18		
	Mean	Var	COR	Mean	Var	COR	Mean	Var	COR	Mean	Var	COR	Mean	Var	COR	Mean	Var	COR
WMT'14	32.09	706.49	0.83	30.72	667.16	0.84	31.56	701.76	0.84	32.79	694.34	0.80	31.95	725.51	0.84	32.48	723.32	0.83
IWSLT'17	8.56	185.86	1.59	8.08	182.21	1.67	6.70	146.26	1.80	8.86	194.29	1.57	8.60	192.21	1.61	10.33	205.54	1.39
MTNT'19	5.76	150.79	2.13	5.52	147.63	2.20	4.72	124.94	2.37	5.21	136.62	2.24	6.19	159.87	2.04	7.00	173.80	1.88
BIO'18	12.67	306.96	1.38	12.03	295.04	1.43	11.79	301.49	1.47	12.27	297.29	1.41	11.86	298.77	1.46	15.32	329.75	1.19
APPERTIUM	2.83	81.19	3.19	2.70	74.86	3.20	2.80	80.72	3.21	3.01	82.15	3.01	2.78	80.18	3.22	2.91	85.65	3.18

Table 3: Robustness Metrics by Bagging in machine translation. Sentence level BLEU scores of bagging of test with a single test set is left out. *Var* is variance of BLEU scores and *COR* is the coefficient of variance.

Model	Leave-One-Out (Bagging)																	
	None			apparel			baby			beauty			grocery			music		
	Mean	Var	COR	Mean	Var	COR	Mean	Var	COR	Mean	Var	COR	Mean	Var	COR	Mean	Var	COR
dvd	58.95	0.012	0.02	59.27	0.026	0.043	59.15	0.014	0.025	58.73	0.018	0.03	58.53	0.018	0.032	59.65	0.021	0.036
books	50.21	0.035	0.069	50.87	0.02	0.04	50.61	0.021	0.042	50.43	0.023	0.047	49.92	0.037	0.075	50.72	0.027	0.053
kitchen	50.88	0.013	0.026	51.25	0.015	0.03	51.15	0.019	0.038	50.39	0.025	0.049	50.22	0.024	0.047	51.06	0.022	0.043
electronics	42.38	0.015	0.037	42.02	0.028	0.068	42.13	0.017	0.041	42.58	0.022	0.053	42.68	0.016	0.038	41.79	0.016	0.039

Table 4: Robustness Metrics by Bagging in sentiment classification. Domain level accuracy scores when a single test set is left out. *Var* is variance of accuracy scores and *COR* is the coefficient of variance.

Model	Leave-One-Out (Design Bagging)																	
	None			WMT'15			IWSLT'17			MTNT'19			BIO'18			MTNT'18		
	Mean	Var	COR	Mean	Var	COR	Mean	Var	COR	Mean	Var	COR	Mean	Var	COR	Mean	Var	COR
WMT'14	31.94	703.93	0.83	30.85	667.02	0.84	31.61	707.62	0.84	32.80	693.68	0.80	31.87	722.14	0.84	32.51	723.56	0.83
IWSLT'17	8.55	185.95	1.59	8.10	182.49	1.67	6.70	145.45	1.80	8.82	192.42	1.57	8.59	192.10	1.61	10.35	206.56	1.39
MTNT'19	5.76	150.24	2.13	5.51	146.54	2.20	4.76	126.83	2.37	5.19	136.47	2.25	6.16	160.23	2.05	6.97	173.48	1.89
BIO'18	12.69	307.07	1.38	12.03	295.87	1.43	11.82	303.21	1.47	12.27	297.13	1.41	11.88	298.30	1.45	15.35	330.40	1.18
APERTIUM	2.84	80.88	3.17	2.69	74.30	3.20	2.82	82.10	3.22	3.00	81.75	3.01	2.77	80.18	3.24	2.92	85.08	3.16

Table 5: Robustness Metrics by Design Bagging in machine translation. Sentence level BLEU scores of design bagging of test with a single test set is left out. *Var* is variance of BLEU scores and *COR* is the coefficient of variance.

Model	Leave-One-Out (Design Bagging)																	
	None			apparel			baby			beauty			grocery			music		
	Mean	Var	COR	Mean	Var	COR	Mean	Var	COR	Mean	Var	COR	Mean	Var	COR	Mean	Var	COR
dvd	57.80	0.02	0.035	59.27	0.028	0.047	59.03	0.021	0.035	57.93	0.019	0.032	57.58	0.018	0.031	59.29	0.027	0.046
books	59.54	0.025	0.042	61.20	0.033	0.055	60.92	0.024	0.04	59.78	0.022	0.037	59.34	0.021	0.035	61.15	0.032	0.053
kitchen	54.61	0.008	0.015	55.40	0.009	0.016	55.42	0.008	0.016	54.88	0.008	0.016	53.99	0.007	0.014	55.17	0.011	0.021
electronics	57.40	0.008	0.014	57.99	0.007	0.012	59.37	0.019	0.032	56.97	0.005	0.009	56.84	0.005	0.009	58.00	0.008	0.014

Table 6: Robustness Metrics by Design Bagging in sentiment classification. Domain level accuracy scores when a single test set is left out. *Var* is variance of accuracy scores and *COR* is the coefficient of variance.

model to generate contextualized adversarial examples. Niu et al. (2020) evaluates robustness to input perturbations for neural machine translation. While many NLP models achieve better performance af-

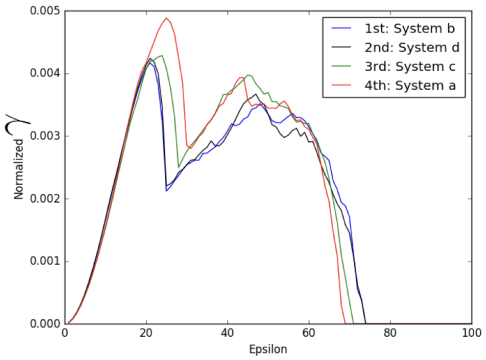

Figure 5: WMT'18 submission systems, human evaluation, (ϵ, γ)-Robustness measured on human evaluation scores.

Model	BLEU	rank$_{BLEU}$	rank$_{\gamma}$	rank$_{Human}$
System A	0.379	4	4	4
System B	0.322	**2**	**1**	**1**
System C	0.362	3	3	3
System D	0.320	**1**	**2**	**2**

Table 7: (ϵ, γ)-Robustness 100% agrees with human ranking, while BLEU 75% agrees with human.

Model 1	Model 2	CV	γ	Human	Agree?
WMT	BIO	WMT	WMT	WMT	YES
BIO	IWSLT	BIO	BIO	IWSLT	NO
WMT	MTNT	WMT	WMT	WMT	YES
WMT	IWSLT	WMT	WMT	WMT	YES
BIO	MTNT	BIO	BIO	BIO	YES
IWSLT	MTNT	IWSLT	IWSLT	IWSLT	YES

Table 8: Robustness Metrics pair-wise comparison on each two models.

ter retraining with adversarial examples, the lack of an attack-agnostic evaluation metric leaves the evaluation of model's intrinsic robustness difficult especially when seeing new adversarial attacks.

Another line of work examines the robustness of NLP models among various domains/distributions. Hendrycks et al. (2020) compared the robustness of pretrained Transformers and found that pretraining models on diverse data helps to improve out-of-distribution generalization. Müller et al. (2019) tests several techniques such as subword regularization, defensive model distillation to improve generalization of machine translation models. These work detect the model performance drop under domain shifts but did not give notion of robustness as consistent performance among domains or distributions. In contrast, we propose three robustness metrics that are easy to measure quantitatively using bagging or design bagging.

Some literature propose evaluation metrics for robustness from the perspectives of statistics or input perturbations (Weng et al., 2018; Niu et al., 2020; Mangal et al., 2019), however, they either focus on the worst-scenario of adversarial inputs or disregard the full distribution of performance scores.

7 Conclusion

We introduce variance, coefficient of variation and (ϵ, γ)-Robustness to measure the robustness of an NLP model's typical behavior. Our robustness metrics outperform BLEU in MT system performance rankings and highly agree with human robustness assessment. Our work demonstrates a successful step towards general robustness evaluation and optimization goals.

Acknowledgments

We appreciate the National Science Foundation (NSF) Award No. 1747728 and NSF CRAFT Award, Grant No. 22001 to fund this research. We are also thankful for the support of the Google Cloud Research Program. We especially thank JSALT'19 for providing the platform with top research and collaborative opportunities to inspire and initiate this work. We sincerely thank Jackie C.K. Cheung, Paul Mitchel, Graham Neubig, and Jacob Eisenstein for the initial discussion of this work.

References

Moustafa Alzantot, Yash Sharma, Ahmed Elgohary, Bo-Jhang Ho, Mani Srivastava, and Kai-Wei Chang. 2018. Generating natural language adversarial examples. *arXiv preprint arXiv:1804.07998*.

Alexandre Araujo, Laurent Meunier, Rafael Pinot, and Benjamin Negrevergne. 2019. Robust neural networks using randomized adversarial training. *arXiv preprint arXiv:1903.10219*.

Osbert Bastani, Yani Ioannou, Leonidas Lampropoulos, Dimitrios Vytiniotis, Aditya Nori, and Antonio Criminisi. 2016. Measuring neural net robustness with constraints. *Advances in neural information processing systems*, 29:2613–2621.

John Blitzer, Mark Dredze, and Fernando Pereira. 2007. Biographies, bollywood, boom-boxes and blenders: Domain adaptation for sentiment classification. In *Proceedings of the 45th annual meeting of the association of computational linguistics*, pages 440–447.

Leo Breiman. 1996. Bagging predictors. *Machine learning*.

Nicholas Carlini and David Wagner. 2017. Towards evaluating the robustness of neural networks. In *2017 ieee symposium on security and privacy (sp)*, pages 39–57. IEEE.

Jonas Gehring, Michael Auli, David Grangier, Denis Yarats, and Yann N. Dauphin. 2017. Convolutional sequence to sequence learning. *CoRR*, abs/1705.03122.

Matthias Hein and Maksym Andriushchenko. 2017. Formal guarantees on the robustness of a classifier against adversarial manipulation. *arXiv preprint arXiv:1705.08475*.

Christina Heinze-Deml and Nicolai Meinshausen. 2017. Conditional variance penalties and domain shift robustness. *arXiv preprint arXiv:1710.11469*.

Dan Hendrycks, Xiaoyuan Liu, Eric Wallace, Adam Dziedzic, Rishabh Krishnan, and Dawn Song. 2020. Pretrained transformers improve out-of-distribution robustness. *arXiv preprint arXiv:2004.06100*.

Di Jin, Zhijing Jin, Joey Tianyi Zhou, and Peter Szolovits. 2020. Is bert really robust? a strong baseline for natural language attack on text classification and entailment. In *Proceedings of the AAAI conference on artificial intelligence*, volume 34, pages 8018–8025.

Dianqi Li, Yizhe Zhang, Hao Peng, Liqun Chen, Chris Brockett, Ming-Ting Sun, and Bill Dolan. 2020. Contextualized perturbation for textual adversarial attack. *arXiv preprint arXiv:2009.07502*.

Ravi Mangal, Aditya V Nori, and Alessandro Orso. 2019. Robustness of neural networks: A probabilistic and practical approach. In *2019 IEEE/ACM 41st International Conference on Software Engineering: New Ideas and Emerging Results (ICSE-NIER)*, pages 93–96. IEEE.

Mathias Müller, Annette Rios, and Rico Sennrich. 2019. Domain robustness in neural machine translation. *arXiv preprint arXiv:1911.03109*.

Xing Niu, Prashant Mathur, Georgiana Dinu, and Yaser Al-Onaizan. 2020. Evaluating robustness to input perturbations for neural machine translation. *arXiv preprint arXiv:2005.00580*.

Myle Ott, Sergey Edunov, Alexei Baevski, Angela Fan, Sam Gross, Nathan Ng, David Grangier, and Michael Auli. 2019. fairseq: A fast, extensible toolkit for sequence modeling. In *Proceedings of NAACL-HLT 2019: Demonstrations*.

Periklis Papakonstantinou, Jia Xu, and Zhu Cao. 2014. Bagging by design (on the suboptimality of bagging). In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 28.

Tianlu Wang, Xuezhi Wang, Yao Qin, Ben Packer, Kang Li, Jilin Chen, Alex Beutel, and Ed Chi. 2020. Catgen: Improving robustness in nlp models via controlled adversarial text generation. *arXiv preprint arXiv:2010.02338*.

Tsui-Wei Weng, Huan Zhang, Pin-Yu Chen, Jinfeng Yi, Dong Su, Yupeng Gao, Cho-Jui Hsieh, and Luca Daniel. 2018. Evaluating the robustness of neural networks: An extreme value theory approach. *arXiv preprint arXiv:1801.10578*.

Huan Xu and Shie Mannor. 2012. Robustness and generalization. *Mach. Learn.*, 86(3):391–423.

CSL: A Large-scale Chinese Scientific Literature Dataset

Yudong Li[1,2], **Yuqing Zhang**[1*], **Zhe Zhao**[3], **Linlin Shen**[2], **Weijie Liu**[3], **Weiquan Mao**[3], **and Hui Zhang**[4]

[1] China University of Geosciences (Beijing), School of Information Engineering
[2] Shenzhen University, School of Computer Science and Software Engineering
[3] Tencent AI Lab
[4] Information Technology Center for National Science and Technology Infrastructure, Beijing, China

Abstract

Scientific literature serves as a high-quality corpus, supporting a lot of Natural Language Processing (NLP) research. However, existing datasets are centered around the English language, which restricts the development of Chinese scientific NLP. In this work, we present CSL, a large-scale **C**hinese **S**cientific **L**iterature dataset, which contains the titles, abstracts, keywords and academic fields of 396k papers. To our knowledge, CSL is the first scientific document dataset in Chinese. The CSL can serve as a Chinese corpus. Also, this semi-structured data is a natural annotation that can constitute many supervised NLP tasks. Based on CSL, we present a benchmark to evaluate the performance of models across scientific domain tasks, i.e., summarization, keyword generation and text classification. We analyze the behavior of existing text-to-text models on the evaluation tasks and reveal the challenges for Chinese scientific NLP tasks, which provides a valuable reference for future research. Data and code are available at https://github.com/ydli-ai/CSL.

1 Introduction

With the increase in the publication of papers, Natural Language Processing (NLP) tools that assist users in writing, searching, and archiving scientific literature have grown increasingly important. For instance, paper/citation recommendation (Beel et al., 2016; Cohan et al., 2020), topic classification (Beltagy et al., 2019) and summarization (Cohan and Goharian, 2018) systems have been developed. The construction of these automatic systems primarily relies on academic resources such as large-scale corpus (Lo et al., 2020; Saier and Färber, 2020), citation graphs (Sinha et al., 2015; Tang et al., 2008; Zhang et al., 2019) and supervised scientific datasets (Li et al., 2016; Jurgens et al., 2018). These resources, however, are mostly centered around the English language, which restricts the development of techniques for addressing non-English scientific NLP tasks. Until recently, progresses in the NLP research for Chinese resources and models has lagged behind their English counterparts.

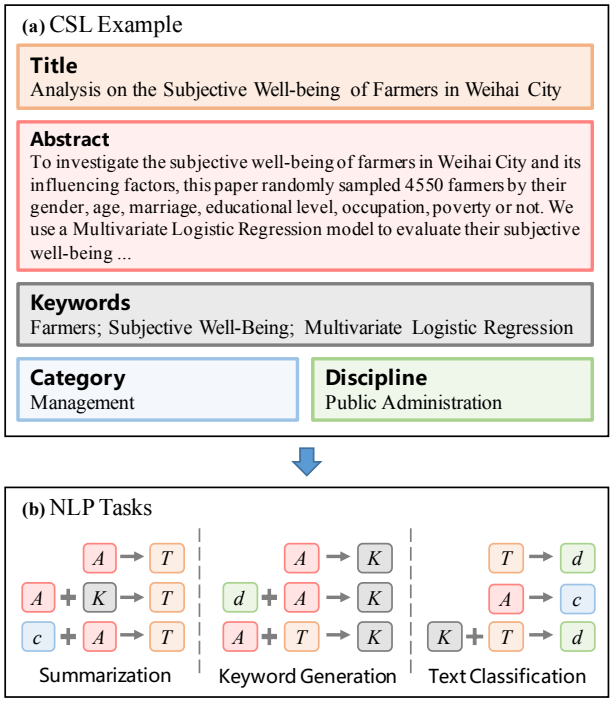

Figure 1: (a) An example of paper meta-information (translated into English). (b) Examples of NLP tasks constructed from CSL. The arrow indicates the input and output of the task, for example, "$A \rightarrow T$" represents the task feeding abstract to produce title. A: abstract; T: title; K: keywords; c: category; d: discipline.

To fill the gap of non-English scientific resources and spur the Chinese scientific NLP research, in this paper, we introduce CSL: a large-scale **C**hinese **S**cientific **L**iterature dataset. CSL contains 396,209 Chinese papers' meta-information, including title, abstract, keywords, academic category and discipline. Papers are collected from comprehensive Chinese academic journals covering a wide range of distribution. In particular, we divide them into

Proceedings of the 29th International Conference on Computational Linguistics, pages 3917–3923
October 12–17, 2022.

Category	#d	len(T)	len(A)	num(K)	#Samples	Discipline Examples
Engineering	27	19.1	210.9	4.4	177,600	*Mechanics, Architecture, Electrical Science*
Science	9	20.7	254.4	4.3	35,766	*Mathematics, Physics, Astronomy, Geography*
Agriculture	7	17.1	177.1	7.1	39,560	*Crop Science, Horticulture, Forestry*
Medicine	5	20.7	269.5	4.7	36,783	*Clinical Medicine, Dental Medicine, Pharmacy*
Management	4	18.7	157.7	6.2	23,630	*Business Management, Public Administration*
Jurisprudence	4	18.9	174.4	6.1	21,554	*Legal Science, Political Science, Sociology*
Pedagogy	3	17.7	179.4	4.3	16,720	*Pedagogy, Psychology, Physical Education*
Economics	2	19.5	177.2	4.5	11,558	*Theoretical Economics, Applied Economics*
Literature	2	18.8	158.2	8.3	10,501	*Chinese Literature, Journalism*
Art	1	17.8	170.8	5.4	5,201	*Art*
History	1	17.6	181.0	6.0	6,270	*History*
Strategics	1	17.5	169.3	4.0	3,555	*Military Science*
Philosophy	1	18.0	176.5	8.0	7,511	*Philosophy*
All	67				396,209	

Table 1: Detailed statistics of the CSL dataset. #d: The number of disciplines in the category. len(T): Average length of each title; len(A): Average length of each abstract; num(K): Average number of keywords.

13 first-level categories and 67 second-level disciplines. In addition to the difference in language, it provides broader and more fine-grained research fields than existing academic resources (Lo et al., 2020; Saier and Färber, 2019).

Scientific literature metadata contains abundant semantic information, making it a natural annotated data source with the potential to provide many NLP tasks. For example, predicting the title with abstract constitutes a summarization task. As the data and task examples shown in Figure 1, such combinations can constitute abundant tasks. These tasks can drive models in real-world applications and are essential for a lot of academic NLP research. To better understand the challenges posed by Chinese scientific NLP, we build a benchmark consisting of a series of CSL-derived tasks, i.e., summarization, keyword generation and category/discipline classification. We also provide a toolkit that allows users to design evaluation tasks according to their needs.

We implement some state-of-the-art Chinese text-to-text models and evaluate on the proposed benchmark. We also demonstrate the effectiveness of the CSL dataset as pre-training corpus. Specifically, we pre-train T5 with paper abstracts, namely CSL-T5. It outperforms the model trained on the general-domain corpus, which can be used as a strong baseline for the proposed benchmark. The experiment results show that though existing models can achieve acceptable performance on scientific NLP tasks, it still needs future efforts to reach a practical level.

The main contributions of this paper are summarized as follows:

- We release the first large-scale Chinese Scientific Literature dataset (CSL), which can be used for many different purposes, e.g., pre-training corpus and scientific-related tasks.
- Based on the CSL, we build a benchmark that represents real-world scenarios of automatic analyzing scientific literature.
- We implement text-to-text models to provide baselines. The experimental results highlight the model's difficulties in Chinese scientific NLP tasks.

2 The CSL Dataset

2.1 Data Collection

We obtain the paper's meta-information from the National Engineering Research Center for Science and Technology Resources Sharing Service (NSTR) [1] dated from 2010 to 2020. Then, we filter data by the Catalogue of Chinese Core Journals, which is an academic journal evaluation standard published by Peking University Library. It selects 1,980 core journals from the Chinese journals, widely recognized by the Chinese academic community.

According to the Catalogue and collected data, we divide academic fields into 13 first-level categories (e.g., Engineering, Science) and 67 second-level disciplines (e.g., Mechanics, Mathematics). We use the journal's instructions to assign journals to categories and disciplines, and only journals that focus on a single academic field are kept. For the guideline of human annotation, we follow the Disciplines of Conferring Academic Degrees (GB/T 13745-2009). We ask volunteers to read the intro-

[1] https://nstr.escience.net.cn

Dataset	Instances	Language	Peer Review	Source	Academic Disciplines
S2ORC (2020)	8.1M	English	not all	MAG, arXiv, PubMed	20 (multi)
PubMed Central OAS	2.3M	English	not all	PubMed	2 (bio, LS)
unarXive (2020)	1.0M	English	not all	MAG, arXiv	4 (physics, math, CS, other)
Saier and Färber, 2019	1.0M	English	not all	arXiv	3 (physics, math, CS)
arXiv CS (2018)	90k	English	not all	arXiv	1 (CS)
AAN (2013)	25k	English	all	ACL Anthology	1 (comp ling)
CSL (ours)	396k	Chinese	all	Chinese Core Journals	67 (multi)

Table 2: A comparison of CSL with other publicly-available scientific literature datasets. Note that we provide the first dataset in Chinese, which also has the more fine-grained discipline annotation. bio: biomedicine; LS: life science; CS: computer science; comp ling: computational linguistics.

duction of the journal and find the closest discipline from the guideline. As a result, papers can be labeled with categories and disciplines based on the journal in which they were published. For example, papers from the "Chinese Journal of Computers" are categorized into the category "Engineering" and the discipline "Computer Science".

In total, we collect 396,209 instances for the CSL dataset, represented as tuples $< T, A, K, c, d >$, where T is the title, A is the abstract, K is a list of keywords, c is the category label and d is the discipline label. Due to the ethical concern, we only use the paper's publicly available meta-information and do not access the full text.

2.2 Data Analysis

The paper distribution over categories and the examples of disciplines are shown in Table 1. A total of 67 disciplines are collected by CSL, covering almost all research fields. Each discipline contains 3000-10000 samples.

Table 2 presents an overview of existing academic datasets. In comparison, the CSL dataset has the following features: **(1) Wider discipline coverage.** Existing datasets mainly focus on specific domains, while CSL covers almost all research domains. Also, CSL has more fine-grained discipline labels. **(2) New data source.** It can be seen that existing datasets are largely built on digital libraries like arXiv [2], PubMed [3], ACL Anthology [4] and MAG (Sinha et al., 2015), resulting in some overlap. CSL presents a new data source in Chinese that complements existing resources. **(3) Higher quality and accuracy.** Digital libraries contain pre-print platforms, and therefore some papers are not peer-reviewed. CSL is collected from published journal papers and is potentially of higher quality. In addition, CSL directly accesses the database without PDF/LaTeX parsing, which has near-perfect accuracy.

2.3 Evaluation Benchmark

The CSL contains meta-information provided by authors when submitting their papers, and the connections between them can constitute many NLP tasks. In this section, we build a benchmark to facilitate the development of Chinese scientific literature NLP. It contains diverse tasks, ranging from classification to text generation, representing many practical scenarios. We randomly select 10,000 samples and split the datasets into training set, validation set and test set according to the ratio, 0.8 : 0.1 : 0.1. This split is shared across different tasks, which allows multi-task training and evaluation. From CSL, many possible combinations can also constitute tasks. We provide a toolkit for users to design tasks by their needs.

Text Summarization (TS) The paper title can be seen as a summary of the paper abstract. We build a summarization task predicting the paper title from the abstract. Existing Chinese text summarization resources are mainly concentrated in the news domain (Hu et al., 2015; Liu et al., 2020), and we provide the first text summarization task in the academic domain.

Keyword Generation (KG) In this task, the model is asked to predict a list of keywords from a given paper title and abstract. This task is similar to the Paper Topic Classification (Cohan et al., 2020), but instead of predicting topics in a set of candidates, the goal is to generate keywords that correspond to the paper. We construct a dataset of paper's keywords, title and abstract. In English, there are related datasets such as SemEval (Kim et al., 2013) and KP20k (Meng et al., 2017). To

[2] https://arxiv.org
[3] http://www.pubmed.gov
[4] https://aclanthology.org

the best of our knowledge, CSL provides the first Chinese keyword generation task.

Text Classification This task is predicting the category and discipline based on other information about the paper. We build a dataset for **category classification ($\mathbf{CTG_{cls}}$)**, which predicts the category with the paper title. Besides, we build a **discipline classification ($\mathbf{DCP_{cls}}$)** task that predicts the discipline with the paper abstract.

3 Experiments

3.1 Baseline Models

For baselines, we evaluate multi-task learning models trained on CSL tasks. We use the text-to-text (i.e., feed text to produce text) method to unify downstream tasks in different formats. Specifically, these tasks are represented as the language generation task guided by a textual prompt. We adopt several widely used text-to-text models, including T5 (Raffel et al., 2020), PEGASUS (Zhang et al., 2020), and BART (Lewis et al., 2019). Since there are few publicly available versions of them, we conduct pre-training on the Chinese corpus from scratch. In addition, we train a T5 using CSL paper abstracts as the corpus, namely CSL-T5, to provide a pre-training model that adapts to the Chinese scientific domain.

3.2 Settings

For pre-training Chinese text-to-text models, we follow the architecture, optimization, and hyperparameter choices described in the papers. Following Google's Chinese BERT (Devlin et al., 2019), we use the tokenizer with a vocabulary of 21,128 Chinese characters. Models are pre-trained on the CLUE Corpus Small (Xu et al., 2020) for 1M steps with the batch size of 512. We progressively train CSL-T5 basis on pre-trained T5, using the paper abstract as the corpus for 20k steps with the same hyperparameters.

Experiments are conducted on UER-py framework (Zhao et al., 2019) [5].The learning rate is set to $3e^{-4}$ for T5; $1e^{-5}$ for BART and PEGASUS. The batch size is 32. For multi-task training, we combine the training sets of each task for training 5 epochs. We use a prompt to specify which task the model should perform, e.g., "to category" for category classification. Then, we fine-tune the models on the task to be evaluated for 3 epochs with early stopping. All results are reported with greedy decoding.

Models	CTG_{cls}	DCP_{cls}	TS	KG
	Acc.	Acc.	R-L	Bpref.
T5	**83.6**	67.1	49.8	54.2
PEGASUS	81.7	69.4	49.4	55.2
BART	79.2	65.7	47.8	49.9
T5 (single)	82.3	63.2	49.2	54.1
CSL-T5	82.9	**70.8**	**52.1**	**55.9**

Table 3: The test performances of baseline models on CSL downstream tasks. T5 (single) is the result of fine-tuning T5 on each task separately, and the remaining columns are the results of multi-task learning.

Prompt: to title **Input Text:** 通过对美国职业排球运动员进行非结构性访谈研究美国职业排球运动员对赞助商和赞助行为的态度... 赞助商应尊重运动员的情感和观点,从而使双方都能获得长远利益. Through interviews, research was conducted on the attitudes of American professional volleyball players regarding sponsors and sponsorship activities ... Sponsors should respect athletes' feelings and opinions in order for both sides to profit in the long run. **Prediction:** 美国职业排球运动员对赞助商和赞助行为的态度研究 Research on American Professional Volleyball Players' Attitudes Towards Sponsors and Sponsorship Behaviors **Ground Truth:** 美国排球运动员对赞助的态度分析 Analysis of American Volleyball Players' Attitudes towards Sponsorship
Prompt: to keywords **Input Text:** 通过对祁连山自然保护区周边农牧民经济状况的调查发现阻碍经济发展的问题... 提出了发展生态旅游等适合本地区经济发展的模式. Problems with economic development were discovered during an investigation of the economic conditions of farmers and herders in the Qilian Mountain Nature Reserve ... Ecotourism and other models for local economic development were proposed. **Prediction:** 祁连山自然保护区; 农牧民; 经济发展模式 Qilian Mountain Nature Reserve; Peasants and herdsmen; Economic development model **Ground Truth:** 祁连山自然保护区; 周边经济; 发展模式 Qilian Mountain Nature Reserve; Peripheral economy; Development model

Table 4: Samples of text summarization and keyword generation of CSL-T5.

[5]https://github.com/dbiir/UER-py

3.3 Overall Performance

The experimental results are shown in Table 3. Output samples of text summarization and keyword generation tasks are shown in Table 4. For text classification, we report accuracy. We use ROUGE-L (Lin and Hovy, 2003) for the summarization task, which is commonly used for language generation tasks. For keyword generation, we use Bpref. (Buckley and Voorhees, 2004), which evaluates both the accuracy and order of generated keywords. We can observe that baseline models can achieve acceptable results, where T5 outperforms other models. However, it is still not satisfactory for real-world applications, and future efforts are needed. We also find that domain-adaptive training can further improve T5's performance. Similar experiments are also done by Beltagy et al. (2019) and Gururangan et al. (2020), it partially demonstrates the value of CSL corpus for pre-training. The model and corpus will be publicly available.

To discover the effect of multi-task training, we fine-tune T5 with each task individually. From the comparison between T5 and T5-single, multi-task learning slightly outperforms individually fine-tuned models. We speculate that since the CSL tasks are homogeneous (derived from the same dataset), it is easier for models to share knowledge across different tasks.

CSL can create a large number of tasks by different combinations of tasks' input and output. It provides a natural playground for observing which tasks are mutually reinforcing when learned together. CSL could also be useful for cross-task research (Ye et al., 2021; Bragg et al., 2021). For example, exploring which tasks the model learns can help it quickly adapt to new tasks. We leave that for future exploration.

4 Conclusion

This paper presents the first Chinese scientific literature dataset, CSL, which can serve as a pre-training corpus and can derive abundant NLP tasks. Based on CSL, we build an evaluation benchmark to explore the challenges posed by automatic analysis of Chinese scientific documents. Experimental results find difficulties in existing models in the Chinese scientific domain and point out the future directions.

Limitations and future work. In the current version of CSL, to provide accurate category/discipline labels, we only use journals focused on one domain, which resulted in some data loss. In future work, we will provide multi-label datasets to cover cross domain papers, and annotate CSL with more attributes like Chinese-English parallel data for academic machine translation. Also, the versatile NLP task derived from CSL constitutes a naturally cross-task scenario. In the future, we will explore the role of CSL in cross-task and few-shot research.

Ethical Considerations

The corpus we use is released by the Chinese government aimed at sharing academic resources, which has been anonymized wherever necessary. We are licensed to use some of paper's metadata for NLP research. Therefore, our dataset does not involve any privacy or copyright issues.

Acknowledgments

This research was supported by National Natural Science Foundation of China under grant no. 91959108 and Guangdong Basic and Applied Basic Research Foundation under Grant 2020A1515111199. We appreciate the data preprocessing and annotation effort done by Zichun Cao, Peixia Zhang, and Zhihui Wu in Prof. Zhang's lab at CUGB.

References

Joeran Beel, Bela Gipp, Stefan Langer, and Corinna Breitinger. 2016. Paper recommender systems: a literature survey. *International Journal on Digital Libraries*, 17(4):305–338.

Iz Beltagy, Kyle Lo, and Arman Cohan. 2019. Scibert: A pretrained language model for scientific text. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 3615–3620.

Jonathan Bragg, Arman Cohan, Kyle Lo, and Iz Beltagy. 2021. Flex: Unifying evaluation for few-shot nlp. *Advances in Neural Information Processing Systems*, 34.

Chris Buckley and Ellen M Voorhees. 2004. Retrieval evaluation with incomplete information. In *Proceedings of the 27th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 25–32.

Arman Cohan, Sergey Feldman, Iz Beltagy, Doug Downey, and Daniel S Weld. 2020. Specter: Document-level representation learning using citation-informed transformers. In *Proceedings*

of the 58th Annual Meeting of the Association for Computational Linguistics, pages 2270–2282.

Arman Cohan and Nazli Goharian. 2018. Scientific document summarization via citation contextualization and scientific discourse. *International Journal on Digital Libraries*, 19(2):287–303.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. Bert: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186.

Michael Färber, Alexander Thiemann, and Adam Jatowt. 2018. A high-quality gold standard for citation-based tasks. In *Proceedings of the Eleventh International Conference on Language Resources and Evaluation (LREC 2018)*.

Suchin Gururangan, Ana Marasović, Swabha Swayamdipta, Kyle Lo, Iz Beltagy, Doug Downey, and Noah A Smith. 2020. Don't stop pretraining: Adapt language models to domains and tasks. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 8342–8360.

Baotian Hu, Qingcai Chen, and Fangze Zhu. 2015. Lcsts: A large scale chinese short text summarization dataset. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 1967–1972.

David Jurgens, Srijan Kumar, Raine Hoover, Dan McFarland, and Dan Jurafsky. 2018. Measuring the evolution of a scientific field through citation frames. *Transactions of the Association for Computational Linguistics*, 6:391–406.

Su Nam Kim, Olena Medelyan, Min-Yen Kan, and Timothy Baldwin. 2013. Automatic keyphrase extraction from scientific articles. *Language resources and evaluation*, 47(3):723–742.

Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Ves Stoyanov, and Luke Zettlemoyer. 2019. Bart: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. *arXiv preprint arXiv:1910.13461*.

Jiao Li, Yueping Sun, Robin J Johnson, Daniela Sciaky, Chih-Hsuan Wei, Robert Leaman, Allan Peter Davis, Carolyn J Mattingly, Thomas C Wiegers, and Zhiyong Lu. 2016. Biocreative v cdr task corpus: a resource for chemical disease relation extraction. *Database*, 2016.

Chin-Yew Lin and Eduard Hovy. 2003. Automatic evaluation of summaries using n-gram co-occurrence statistics. In *Proceedings of the 2003 Human Language Technology Conference of the North American Chapter of the Association for Computational Linguistics*, pages 150–157.

Xiaojun Liu, Chuang Zhang, Xiaojun Chen, Yanan Cao, and Jinpeng Li. 2020. Clts: A new chinese long text summarization dataset. In *CCF International Conference on Natural Language Processing and Chinese Computing*, pages 531–542. Springer.

Kyle Lo, Lucy Lu Wang, Mark Neumann, Rodney Kinney, and Daniel S Weld. 2020. S2orc: The semantic scholar open research corpus. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 4969–4983.

Rui Meng, Sanqiang Zhao, Shuguang Han, Daqing He, Peter Brusilovsky, and Yu Chi. 2017. Deep keyphrase generation. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 582–592.

Dragomir R Radev, Pradeep Muthukrishnan, Vahed Qazvinian, and Amjad Abu-Jbara. 2013. The acl anthology network corpus. *Language Resources and Evaluation*, 47(4):919–944.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J Liu. 2020. Exploring the limits of transfer learning with a unified text-to-text transformer. *Journal of Machine Learning Research*, 21:1–67.

Tarek Saier and Michael Färber. 2019. Bibliometric-enhanced arxiv: A data set for paper-based and citation-based tasks. In *Proceedings of the 8th International Workshop on Bibliometric-enhanced Information Retrieval (BIR 2019) co-located with the 41st European Conference on Information Retrieval (ECIR 2019)*, volume 2345, page 14–26.

Tarek Saier and Michael Färber. 2020. unarxive: a large scholarly data set with publications' full-text, annotated in-text citations, and links to metadata. *Scientometrics*, 125(3):3085–3108.

Arnab Sinha, Zhihong Shen, Yang Song, Hao Ma, Darrin Eide, Bo-June Hsu, and Kuansan Wang. 2015. An overview of microsoft academic service (mas) and applications. In *Proceedings of the 24th international conference on world wide web*, pages 243–246.

Jie Tang, Jing Zhang, Limin Yao, Juanzi Li, Li Zhang, and Zhong Su. 2008. Arnetminer: extraction and mining of academic social networks. In *Proceedings of the 14th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 990–998.

Liang Xu, Xuanwei Zhang, and Qianqian Dong. 2020. Cluecorpus2020: A large-scale chinese corpus for pre-training language model. *arXiv preprint arXiv:2003.01355*.

Qinyuan Ye, Bill Yuchen Lin, and Xiang Ren. 2021. Crossfit: A few-shot learning challenge for cross-task generalization in nlp. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 7163–7189.

Fanjin Zhang, Xiao Liu, Jie Tang, Yuxiao Dong, Peiran Yao, Jie Zhang, Xiaotao Gu, Yan Wang, Bin Shao, Rui Li, et al. 2019. Oag: Toward linking large-scale heterogeneous entity graphs. In *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, pages 2585–2595.

Jingqing Zhang, Yao Zhao, Mohammad Saleh, and Peter Liu. 2020. Pegasus: Pre-training with extracted gap-sentences for abstractive summarization. In *International Conference on Machine Learning*, pages 11328–11339. PMLR.

Zhe Zhao, Hui Chen, Jinbin Zhang, Wayne Xin Zhao, Tao Liu, Wei Lu, Xi Chen, Haotang Deng, Qi Ju, and Xiaoyong Du. 2019. Uer: An open-source toolkit for pre-training models. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP): System Demonstrations*, pages 241–246.

Singlish Message Paraphrasing: A Joint Task of Creole Translation and Text Normalization

Zhengyuan Liu[†], **Shikang Ni**[‡,*], **Ai Ti Aw**[†], **Nancy F. Chen**[†]
[†]Institute for Infocomm Research, A*STAR, Singapore
[‡]University of Cambridge, UK
{liu_zhengyuan,nfychen}@i2r.a-star.edu.sg

Abstract

Within the natural language processing community, English is by far the most resource-rich language. There is emerging interest in conducting translation via computational approaches to conform its dialects or creole languages back to standard English. This computational approach paves the way to leverage generic English language backbones, which are beneficial for various downstream tasks. However, in practical online communication scenarios, the use of language varieties is often accompanied by noisy user-generated content, making this translation task more challenging. In this work, we introduce a joint paraphrasing task of creole translation and text normalization of Singlish messages, which can shed light on how to process other language varieties and dialects. We formulate the task in three different linguistic dimensions: lexical level normalization, syntactic level editing, and semantic level rewriting. We build an annotated dataset of Singlish-to-Standard English messages, and report performance on a perturbation-resilient sequence-to-sequence model. Experimental results show that the model produces reasonable generation results, and can improve the performance of downstream tasks like stance detection.

1 Introduction

While the development of natural language processing (NLP) has been focused on major languages such as English, Chinese, and French, there is emerging research interest in similar languages, varieties, and dialects (Zampieri et al., 2020). The distinction of language variations generally comes from geographical, historic, communicative settings, and social group dimensions (Coseriu, 1981). In particular, creole languages are formed in conditions when major languages (e.g., English, French) are adopted in another culture or region, and they often mix with existing languages and evolve into other varieties in their own right. Such examples include the French-based Haitian (Degraff, 1992), English-based Australian Kriol (Harris et al., 1993), and Colloquial Singaporean English (Singlish) (Ho et al., 1993). To adopt computational NLP solutions on dialects or creoles, applying existing models trained with major language resources will result in degraded performance, and collecting and annotating sufficient data for task-specified domain adaptation is time-consuming and labor-intensive. Similar to studies on multilingual scenarios (Balahur and Turchi, 2012; Eriguchi et al., 2018), one straightforward and effective approach is to conform the varieties to their base languages by machine translation (Zbib et al., 2012), then other NLP systems (e.g., sentiment analysis, information retrieval, reading comprehension) that take base languages as input could be applied. However, it would be challenging to apply this approach to dialects or creoles that include certain deviated grammar and local vocabularies with systems trained on the corresponding standard languages, especially to the under-resourced language varieties where machine translation performance is subpar (Nguyen and Chiang, 2017; Honnet et al., 2018).

Singlish, namely the colloquial Singapore English, is used in the daily lives of Singaporeans (Ho et al., 1993). Despite the obvious attribute of inheriting a large vocabulary base and foundational grammatical rules from English, Singlish imports terms and features from regional dialects including Mandarin, Malay, Hokkien, Teochew, Cantonese, and Tamil (Deterding, 2007), making its lexicon, syntax, and semantics deviate significantly from English (Wee, 2008; Wang et al., 2017). Thus, Singlish is an expressive language studded with colorful multicultural slangs yet manifested in an extremely condensed form (Leimgruber, 2011). As a result, a person only familiar with American or

[*] The contribution of Shikang Ni is conducted during his student internship in Institute for Infocomm Research, A*STAR, Singapore.

Proceedings of the 29th International Conference on Computational Linguistics, pages 3924–3936
October 12–17, 2022.

Singlish Sentence Hhh. I kaypoh meh. I thought both of us also e kaypoh type. Like dat i dun ask lor...
Converted English Sentence Haha. Am I a busybody? I thought both of us are also the busybody type. Like that then I don't ask...
Singlish Sentence I thk boh bian one. What other topics you want them to talk to u abt ? metaverse meh ?
Converted English Sentence I think it is unavoidable. What other topics do you want them to talk to you about? like Metaverse?

Table 1: Two examples of the Singlish-to-English paraphrasing. Text in blue and red are involved with lexical normalization and creole translation, respectively.

British English, might have a difficult time understanding Singlish (see examples shown in Table 1). This also holds true for computational approaches, where mainstream and popular language models cannot be directly applied to Singlish.

Moreover, in practical use cases (e.g., social platform communication, SMS messages), the challenge is further complicated caused by various noise types in user-generated content, such as typos, spelling variations, phonetic substitutions, and ad hoc abbreviations (Sproat et al., 2001) (see examples shown in Table 1). This would further complicate the dialect or creole translation tasks. While various statistical and neural-based models are proposed for content de-noising in the form of text normalization (Supranovich and Patsepnia, 2015; Muller et al., 2019), much prior work only conduct word-level correction (Baldwin et al., 2015; van der Goot et al., 2021), and non-canonical English varieties are less studied.

Therefore, in this paper, we introduce a joint task of creole language translation and text normalization of Singlish messages. Since the deviations of Singlish from English come from both the lexical and the grammatical levels (Leimgruber, 2011), the task is conducted in a sentence paraphrasing manner. Based on the linguistic characteristics of Singlish, and user behavior of online communication, we further categorize the paraphrasing into three sub-tasks: lexical level normalization, syntactic level editing, and semantic level rewriting. Guided by this linguistic hierarchy, we build a dataset of Singlish-to-English paired messages annotated by human linguistic experts. We then evaluate a neural sequence-to-sequence approach on the paraphrasing task by fine-tuning state-of-the-art language backbones, and further optimize it with linguistic-featured input perturbation. We empirically show that the model can produce reasonable results, and downstream tasks can benefit from such text paraphrasing.[1] While our work focuses on Singlish, a special variant of English, the paraphrasing task formulation, linguistic analysis, and annotation protocol are general and can be extended to studying and processing other creole languages and dialects.

2 Related Work

NLP for Similar Languages Language variation (e.g., different dialects or national varieties of the same language) poses challenges for NLP applications, such as machine comprehension and dialogue systems. As a result, there is much of recent interest in computational processing of creoles and dialects (Zampieri et al., 2020). Related research areas include **language and dialect identification** (Suzuki et al., 2002; Lui et al., 2014; Zampieri et al., 2019) and **machine translation** (Altintas and Cicekli, 2002; Wang et al., 2016). Examples of machine translation between different dialects of the same language include British-American English (Zhao et al., 2000), Cantonese-Mandarin Chinese (Zhang, 1998), and European–Brazilian Portuguese (Costa-jussà et al., 2018). For closely related languages and dialects, many differences occur at the morphological level, thus word-for-word mapping, manual language-specific rules, and phrase-based statistical systems were proposed and applied (Hajič et al., 2000; Nakov and Tiedemann, 2012; Aharoni et al., 2019). Recently, data-driven neural approaches yield further improvement (Costa-jussà et al., 2018), and show the potential of transfer learning from one language pair to another (Nguyen and Chiang, 2017).

Text Normalization Online user-generated content is a valuable NLP resource, but it is often noisy and non-canonical. Most existing models are developed on canonical languages. Such models do not cope well with the disfluencies and informal phenomena (Karpukhin et al., 2019). Text normalization converts such noisy input to a 'normal' format (Sproat et al., 2001), while preserving the original meaning. Eisenstein (2013) studied several underlying factors that cause non-standard text like illiteracy and pragmatics. Since noise often comes

[1] Interested readers can contact corresponding authors for the data and code.

Level 1: Lexical Level Normalization
Lexical Variations in User-Generated Content Tackling the common user-generated lexical variations, including lower-case and upper-case (E.g., 'mrt' → 'MRT', 'TOdAy' → 'TODAY'), spelling typo (e.g., 'domian' → 'domain', 'r0bust' → 'robust'), single-word abbreviations (e.g., 'pple' → 'people', 'coz' → 'because'), phonetic substitutions (e.g., 'tym' → 'time', '4U' → 'for you'), and other non-standard spellings (e.g., 'goooood' → 'good'). **Lexical Variations in Singlish** Tackling the Singlish lexical variations, such as special short forms (e.g., 'yck' → 'YCK (Yio Chu Kang)'), and colloquial words (e.g.,'cheapo' → 'cheapskate', 'gahmen' → 'government'). **Non-English Word Borrowing** Replacing the non-English words borrowed from other languages that have a word-to-word mapping. E.g., 'mei mei' → 'sister' (Mandarin), 'pa tuo' → 'dating' (Cantonese), 'ta pau' → 'take-away' (Cantonese), 'huat' → 'to prosper' (Hokkien), and 'makan' → 'food' (Malay).
Level 2: Syntactic Level Editing
Missing Pronoun & Copula Recovering the appropriate pronouns, and the necessary verbs (e.g. "m typing a sms" → "I am typing a SMS", "oh cat so cute" → "oh the cat is so cute"). **Non-Standard Syntax & Grammar** Fixing the non-standard grammar in colloquial Singlish sentences, such as the topic prominence phenomenon (e.g., "A bit late lah, I came there." → "I came there a bit late."). **Missing Punctuation** Inserting the punctuation to where it is necessary (e.g., "Is that your book" → "Is that your book?").
Level 3: Semantic Level Rewriting
Colloquial Wording Some wording is different from colloquial Singlish and English, thus it needs to paraphrase the sentence while retaining the same semantic meaning (e.g. "Call aint going." → "The call is not coming through.") **Discourse Particles** Some clausal-final discourse particles indicate much semantic information (e.g., 'leh' marks a tentative request, 'lah' is a mood marker, and appeals for accommodation). For instance, "U leh, i going back liao." → "What about you? I am going back." **Non-English Spans & Code-Switching** Some non-English spans and the code-switching require clause or sentence level translation (e.g. "You sian? Let's go shopping!" → "Are you feeling bored? Let's go shopping!", "makan where?" → "where should we eat?").

Table 2: Three sub-tasks of the Singlish message paraphrasing.

from character/token level manipulation, early studies utilized lexical-based methods like dictionary lookup, word similarity, and N-gram probabilities (Han and Baldwin, 2011; Supranovich and Patsepnia, 2015). MoNoise (van der Goot and van Noord, 2017) built a pipeline that is similar to a ranking-retrieval approach. Recently, Muller et al. (2019) enhanced the BERT (Devlin et al., 2019) architecture so that the language model is able to add/remove tokens for word correction, and Bucur et al. (2021) applied a pre-trained language model for multilingual lexical normalization.

3 Singlish Message Paraphrasing Corpus

While unsupervised models show impressive results on tasks like semantic similarity matching by leveraging feature-rich pre-trained backbones (Devlin et al., 2019), their performance on language generation is still subpar. To foster data-driven approaches via supervised learning, we construct a human-annotated corpus for the Singlish message paraphrasing. The raw Singlish messages are extracted from the NUS Short Message Service (SMS) Corpus (Chen and Kan, 2013), which contains a total of 56K message samples from real-world mobile chats. We choose this resource since their data collection process employs Singaporean participants, and the SMS conversations cover a wide range of topics.[2] The annotation target is to convert the messages from colloquial Singapore English to standard American English, and the human-annotated references are expected to be understandable to non-Singaporean high school students.

3.1 Paraphrasing Sub-task Definition

Online communication between creole users is often a mix of language-specific usage and noisy content generation. The paraphrasing task of Singlish

[2]All samples we use are from the published anonymized dataset, and do not contain any personal information.

messages thus requires text editing from multiple aspects. Combining our analysis of real-word Singlish messages, and previous linguistics and lexical normalization studies (Wee, 2008; van der Goot and van Noord, 2017), we categorize this paraphrasing task into three sub-tasks: lexical level normalization, syntactic level editing, and semantic level rewriting.[3]

Lexical Level Normalization Lexical normalization is to uniform the non-standard tokens and borrowed words via infilling and replacement. We first tackle the common English lexical variations like typos, abbreviations, phonetic substitutions, and misspellings which are ubiquitous in online communication platforms (Sproat et al., 2001; Supranovich and Patsepnia, 2015). Moreover, in Singlish messages and conversations, there are special short forms, discourse particles, and words borrowed from other languages (e.g., Mandarin, Malay) (Leimgruber, 2011). In addition to an existing localized vocabulary,[4] the annotators were asked to collect a list of such special words, and some examples are shown in Table 2.

While standard word recovering often can be done independently without sentence understanding, in some cases, the context is necessary for disambiguation. For instance, the word *'goooood'* in *"U r so goooood."* and *"really? oh my goooood!"* should be converted to *'good'* and *'god'*, respectively. Another case is converting some phonetic substitutions, for example: *"This is a gift 4U!"* → *"This is a gift for you!"*. In addition, it is difficult to obtain a complete collection of all non-standard tokens and borrowed words. Therefore, non-computational vocabulary-based methods are not sufficient for lexical normalization (Muller et al., 2019; Bucur et al., 2021), and in our setting, it becomes one sub-task of the sequence-to-sequence modeling.

Syntactic Level Editing The grammar of Singapore English differs from the standard English markedly. For example, some pronouns and BE verbs are often omitted (e.g., *"the weather hot lah"*). Another language-specific phenomenon is the feature 'topic prominence', where the topic span (e.g., noun phrases) is re-ordered to the beginning of the sentence (e.g., *"this book last year i read"* → *"I read this book last year."*). This construction in colloquial Singlish is adapted from Chinese and Malay (Leimgruber, 2011), and the topic prominence can be further highlighted by the insertion of a break or a discourse particle between the topic and the clause (e.g *"Too slow (lah), I find that building"*). Moreover, in online communications, users tend to omit punctuation, especially at the end of sentences (e.g., question marks). Thus, we also take punctuation into consideration in the annotation protocol.

For the sub-task syntactic level editing, sentences are converted to a standard American English grammar, which often requires changes of more than one word or span.

Discourse Particle: 'leh' marks a tentative suggestion or request.
Original Text: Still eating. Got free mcflurry. U (leh), going back liao.. **Paraphrased Text:** I am still eating. I got a free McFlurry. What about you? I am going back.
Discourse Particle: 'hor' attempts to garner support for a proposition. **Original Text:** I go can (hor).. **Paraphrased Text:** Is it alright for me to go?

Table 3: Two examples of clause-final discourse particles in colloquial Singlish.

Semantic Level Rewriting As the narrative and wording of the same meaning are different from Singlish and English, sometimes it needs rewriting the sentence while retaining the same semantic meaning. Particularly, the usage of clausal-final discourse particles (e.g., *'leh'*, *'hor'*), which originates from Hokkien and Cantonese, is one of the most well-known features of Singlish, and some fillers indicate much semantic information (Leimgruber, 2011). For instance, as shown in Table 3, the discourse particle *'hor'* conveys inquiring meaning, thus the sentence *"I can go (hor)"* should be converted to *"Is it alright for me to go?"*. More discourse particles and their examples are shown in Appendix Table 10.

Moreover, in real-world online communication and inter-language scenarios, some non-English spans (involving Singlish and cross-language usage where the syntax is mostly on one language) and the code-switching phenomenon require clause or sentence level translation and rephrasing instead of word level replacement. For instance, the sentence

[3]While colloquial Singapore English presents various lexical and grammar features, here we focus more on those which significantly affect language understanding. Features like tense agreement (Leimgruber, 2011) are not considered if the context is insufficient, to reduce annotation variance.

[4]http://www.singlishdictionary.com/

"makan where?" (the word *'makan'* in Malay means *'eat'*) should be converted to *"where should we eat?"*. This is also included in the semantic level rewriting sub-task.

3.2 Corpus Construction

3.2.1 Sample Pre-processing

We draw our source material from the texts of the NUS Short Message Service (SMS) Corpus (Chen and Kan, 2013), which consists of 56K messages originating from Singaporeans and university students. In our data pre-processing, we first filtered out raw samples that are shorter than 20 characters, as well as duplicated items, resulting in a 45K data size. We then sampled the representative Singlish messages, to refine the subset for annotation. Here the sentence-level perplexity value calculated with a language model GPT-2 (Radford et al., 2019) is used as the criteria, where a high perplexity score indicates the text is more distinct than the canonical English.[5] We then ranked all samples accordingly, and kept those above average.

3.2.2 Data Annotation

The annotation of paraphrasing task is conducted with a group of 6 linguistic experts from a local university, who are proficient in both Singlish and English. To facilitate the process and reduce the annotation variance across different annotators, they are asked to complete the three sub-tasks hierarchically from the low level (lexical normalization) to the high level (semantic rewriting) with minimum changes. Moreover, as the SMS corpus retains the original order of messages in a conversation, we did not shuffle them, thus annotators can refer to the context for better paraphrasing.

Since it is challenging to perfectly paraphrase all Singlish messages, we allow participants to assign confidence scores to their annotations, which indicate the level of agreement between two of any annotators on the same sample, according to the sus-tasks defined in Section 3.1. Confidence scores are from low (0) to high (5) agreement level, and the score of 0 is labeled when the necessary context for rephrasing is missing or the whole source sentence is written in a non-English language (e.g., Tamil, Malay), and such samples are excluded in the training set.

[5]https://huggingface.co/docs/transformers/perplexity

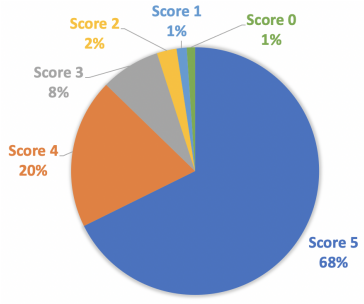

Figure 1: Distribution of confidence scores on the annotated samples. Samples with score 0 are considered as invalid annotation, and excluded from the dataset.

3.2.3 Annotation Analysis

The confidence score distribution of our annotated set is shown in Figure 1. We observed that the scores under average usually result from the semantic level rewriting, especially for non-English span translation. Moreover, to assess the text level variance among annotators, 1500 samples across all confidence levels are randomly selected and annotated by two annotators. Following the common automatic evaluation in machine translation (Aharoni et al., 2019), we use BLEU (Papineni et al., 2002) as the metric, and the average score calculated on those samples is 69.7, which shows a reasonable human annotation agreement.

4 Automatic Paraphrasing via Sequence-to-Sequence Modeling

4.1 Base Neural Architecture

Unlike previous lexical normalization work (Aw et al., 2006; Baldwin et al., 2015), the syntactic and semantic level sub-tasks in our setting require a higher capability of contextual understanding and sentence generation. Therefore, we introduce sequence-to-sequence modeling for Singlish message paraphrasing. Define x as the input text, and y as the target output. A neural encoding-decoding model G is used. The goal is formulated to maximize $P(y|x;\theta_G)$, where θ_G are the learnable parameters. In our settings, the base architecture is a Transformer-based auto-regressive language model, since the Transformer (Vaswani et al., 2017) shows strong capabilities of contextual modeling and generation, and is widely adopted in various natural language processing tasks (Radford et al., 2019; Lewis et al., 2020). The encoder consists of a stack of Transformer layers. Each layer has two sub-components: a multi-head layer with self-attention

mechanism, and a position-wise feed-forward layer. A residual connection is employed between each pair of the two sub-components, followed by layer normalization. The decoder also consists of a stack of Transformer layers. In addition to the two sub-components in the encoding layers, the decoder inserts another component that performs multi-head attention over hidden representations from the last encoding layer. Then, the decoder generates tokens from left to right in an auto-regressive manner. The architecture and formula details are described in (Vaswani et al., 2017).

With the parallel message pairs, we conduct supervised learning with token-level maximum likelihood estimation. At the training stage, the cross-entropy loss is calculated between the decoder's output and the reference sentence:

$$l(\theta) = -\Sigma_i \log(p(y_i|y_{1:i-1}, x; \theta_G)) \quad (1)$$

4.2 Linguistic-featured Input Perturbation

While fine-tuning language backbones bring about impressive performance on cross-language translation (Liu et al., 2020), they are vulnerable to noisy input. Moreover, data-driven approaches may overfit to superficial lexical features rather than learn how to paraphrase from a semantic aspect, especially when the training data are limited.

To tackle these two challenges, we adopt a simple yet effective model enhancement via input perturbation, inspired by the linguistic characteristics of the Singlish messages, and the denoising sequence-to-sequence pre-training scheme (Lewis et al., 2020). There are two operations: (1) Word Perturbation: To simulate lexical variations in real-world online user-generated content, we randomly remove or replace one character of each word (with a 10% probability). Here the word perturbation is only conducted on words from a common English word vocabulary,[6] excluding terminology words, named entities, fillers, and special short forms. Moreover, recent studies show that character-level noise makes models more robust towards spelling variations (Aepli and Sennrich, 2022). (2) Sentence Perturbation: To simulate the grammatical features like topic prominence, and enhance contextual modeling of the language backbone, we randomly inject noise by exchanging a bi-gram pair in each sentence with a 10% probability.

[6]https://github.com/first20hours/google-10000-english

Corpus	Size	Task Type
English Tweets (Baldwin et al., 2015)	3K	Lexical
English Message (Aw et al., 2006)	5K	Lexical
Our Singlish SMS Corpus	20K	All levels

Table 4: Statistics of the corpora used in our setting. The English tweets and message corpora are only for lexical normalization, and are used at the warm-up training stage. See corpus combination results in Table 9.

5 Experiments

5.1 Training Datasets

The Singlish message corpus built in Section 3 is used for model training and evaluation. The training, validation, and test set size are 20000, 1000, and 1000, respectively. In addition to the Singlish dataset, we include two English lexical normalization corpora (Aw et al., 2006; Baldwin et al., 2015) in the warm-up training stage. Data statistics are shown in Table 4. For samples we annotated, the average sentence number per message is 2.13, and the average word number is 18.85.[7] While these two datasets are much smaller and only focus on lexical normalization, in our pilot experiments, we empirically observed that warm-up training with the additional data brings 2-3% relative improvement consistently (see Table 9).

5.2 Experiment Setup

In our experimental setting, we first trained and evaluated the base model: a Vanilla Transformer (Vaswani et al., 2017) (6 encoder and 6 decoder layers, with 768 hidden size and a fixed token embedding layer). We then incorporated prior language knowledge to the base model by loading the *BART-base*, *BART-large*, and *mBART* (Lewis et al., 2020; Liu et al., 2020), and fine-tuned the backbones with parallel pairs. For automatic evaluation metrics, we adopted the common methods used for language generation based on n-gram overlap: BLEU (Papineni et al., 2002), METEOR (Banerjee and Lavie, 2005), ROUGE (Lin, 2004), and the semantic-based metric BERTScore (Zhang et al., 2020) upon similarity of contextualized sentence representations.

All models were implemented with PyTorch and Hugging Face Transformers [8]. AdamW optimizer (Kingma and Ba, 2015) was used. The batch size and learning rates were set at 16 and 2e-5, respec-

[7]https://www.nltk.org/api/nltk.tokenize.html

[8]https://github.com/huggingface/transformers

Model Type	BLEU	METEOR	ROUGE-1	ROUGE-2	ROUGE-L	BERTScore
Unaltered Singlish Messages (Lower bound)	35.6	61.1	63.7	42.5	68.7	72.2
Vanilla Transformer (Vaswani et al., 2017)	47.7	72.3	75.4	65.3	74.1	78.5
BART-base w/ Fine Tuning	55.0	79.7	83.5	70.2	85.7	84.1
BART-base w/ Fine Tuning + Input Perturbation	57.8	81.1	84.6	72.3	86.1	86.5
BART-large w/ Fine Tuning	58.4	81.0	85.0	73.5	86.3	86.4
BART-large w/ Fine Tuning + Input Perturbation	**61.5**	**81.6**	**85.8**	**75.8**	**87.0**	**87.9**
mBART-large-50 w/ Fine Tuning	57.2	80.3	84.6	72.3	85.3	86.0
mBART-large-50 w/ Fine Tuning + Input Perturbation	60.2	81.1	85.2	74.9	85.9	86.7

Table 5: Automatic evaluation scores on the Singlish message paraphrasing task. The proposed text perturbation is conducted on all input samples at the training stage. ROUGE and BERTScore reported here are F1 scores.

tively. We added label smoothing (weight λ = 0.1) on the cross-entropy loss (Müller et al., 2019). Warm-up training step number was 2000. We used early stopping (patience = 5) if validation performance did not improve. Test results were reported with the best validation checkpoints. Beam search size was set at 5. Other information such as environment details and trainable parameter sizes are shown in Appendix Table 12.

5.3 Automatic Evaluation Results

We first calculate the evaluation metrics between **Unaltered Singlish Messages** and annotated text. This serves as a lower bound performance, as no paraphrasing is conducted (see Table 5), which also demonstrates Singlish bears unique usages from standard English. Compared to the **Vanilla Transformer**, leveraging pre-trained language backbones significantly improves the performance, and the **BART-large** outperforms the **BART-base**. Adopting the input perturbation further yields certain improvements. To evaluate the effectiveness of leveraging a multilingual backbone, we also applied **multilingual BART** (mBART) (Liu et al., 2020). However, it did not show any additional performance gains. Presumably, this is because borrowing features and non-English words/spans in Singlish are not well represented in the multilingual pre-training process and data resources. For example, Hokkien and Malay are not in the supported language list of mBART. Moreover, in Singlish messages, the Mandarin words are not expressed in Chinese characters but in Pinyin, which are currently not included in the pre-training of most multilingual backbones.

5.4 Human Evaluation Results

Aside from automatic evaluation, we conducted a human evaluation to complement objective metrics. Following prior work (Wieting and Gimpel, 2018), each text candidate is scored on a five-grade scale of [1, 5], where 1 means the paraphrasing is unacceptable, and 5 means it can be taken as a ground truth. We randomly selected 100 test samples, and asked the linguistic experts to score the corresponding human-written and model-generated outputs. Details of the assessment interface are shown in Appendix Figure 3. Six raters conducted the human evaluation independently, and the average scores are summarized and shown in Table 6. While input perturbation brings significant improvement, there is still space for models to reach human reference performance.

Model Type	Avg. Rating Score
Human Reference	3.87
BART-large w/ Fine Tuning	3.12
+ additional Input Perturbation	3.41

Table 6: Human evaluation results. 100 samples were randomly selected from the test set and assessed by 6 linguistic experts. All rating scores are averaged.

To gain further insights into the limitations of automatic paraphrasing, we conduct text-level analysis on some samples. As shown in Table 7, neural generators produce reasonable changes in the lexical normalization and syntactic editing, while semantic rewriting is still relatively challenging, especially for code-switching and some non-English spans (see more examples in Appendix Table 11). Considering the insufficient language modeling of borrowing words and special abbreviations and the limited corpus with human annotation, we speculate an augmented unsupervised pre-training process is beneficial for tackling this challenge, and it can be one of the future work.

5.5 Experiment on Different Sample Groups

The confidence scores described in Section 3.2.2 are labeled during the annotation process. They can present the inter-annotator agreement level, and partially reflect the sample difficulty of paraphrasing.

Model Type	Text Content
Source Input	Yup... Okay. Cya tmr.. So long nvr write already... Dunno whether tmr can come up with 500 words
Human Reference	Yes... Okay. See you tomorrow... It has been so long since I have written... I do not know whether tomorrow I can come up with 500 words.
BART-large w/ Fine Tuning	Yes... Okay. See you tomorrow... So long never write already... Do not know whether tomorrow I can come up with 500 words.
BART-large w/ Input Perturbation	Yes... Okay. See you tomorrow... I have not written in so long already... I do not know whether tomorrow I can come up with 500 words.

Table 7: One Singlish message example and the generated text from human annotation and neural models. Text spans colored in blue are appropriate changes, and in purple are sub-optimal changes.

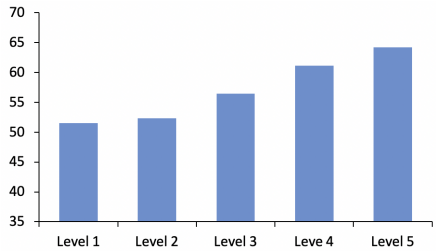

Figure 2: BLEU scores calculated on test sample groups with different confidence scores (range from 1 to 5). Samples with score 0 are considered as invalid annotation, and excluded from the test set.

Model Type	Precision	Recall	F1 Score
Training and Evaluation on Original Samples			
BERTweet-base	68.6	72.1	70.0
RoBERTa-base	70.8	71.7	71.1
Training and Evaluation on Processed Samples			
BERTweet-base	70.4	72.5	**71.2** [1.7% ↑]
RoBERTa-base	71.3	74.3	**72.7** [2.3% ↑]

Table 8: Results on the stance detection task. We processed the corpus with our message paraphrasing model for comparison to raw samples. Values in bracket denote the relative performance increase.

Therefore, we calculate BLEU scores on test sample groups with different confidence scores. As shown in Figure 2, the BLEU scores of level 4 and 5 are larger than those of level 1 and 2, and this demonstrates that samples with lower confidence scores are generally more challenging for both human and automatic paraphrasing.

5.6 Experiment on Corpus Combination

To assess the effectiveness of joint training of lexical normalization and sentence paraphrasing as well as their difference, we further conduct an experiment upon the single and mixed training data combination, as shown in Table 9. From the result, we observed that: (1) models only trained on lexical normalization corpora could not provide strong baseline performance on our Singlish paraphrasing task. (2) compared with single training on our Singlish corpus, training on the merged dataset yields 2-3% relative improvement at all fronts.

5.7 Experiment on Downstream Task

When applied to online communication, the paraphrasing model is able to reduce text noise such as non-canonical wording and misspelling, and it is potentially beneficial for various downstream tasks where noisy samples are ubiquitous. In this article, we choose stance detection of English tweets (Mohammad et al., 2016) for experimentation, which is generally formulated as a classification problem of 3 types (i.e. *Favor*, *Against*, and *None*). The corpus consists English samples with non-standard lexical and syntactic features. We ran the paraphrasing model (*BART-large w/ Fine Tuning + Input Perturbation*) on both training and test tweet samples. Then following previous work, two strong and representative baselines *BERTweet* (Nguyen et al., 2020) and *RoBERTa* (Liu et al., 2019) are trained on the processed corpus (more configuration details are shown in Appendix Table 12), and we reported F1, precision and recall scores on the test set. As shown in Table 8, while the two models perform slightly differently, their classification performance obtained improvement on all fronts after the de-noising transformation (especially a 2.3% relative F1 score increase). This suggests that while our paraphrasing model is trained on Singlish messages, it is still useful for tasks that are not in Singlish since it learned re-writing from context and can reduce the input noise significantly.

6 Conclusions

In this paper, we analyzed the representative linguistic features of colloquial Singapore English, and proposed a joint task of creole language translation and text normalization. We formulated the

Model Type	BLEU	METEOR	ROUGE-1	ROUGE-2	ROUGE-L	BERTScore
Unaltered Singlish Messages (Lower bound)	35.6	61.1	63.7	42.5	68.7	72.2
Training on two English lexical normalization corpora						
BART-base w/ Fine Tuning	45.9	65.7	77.7	63.1	80.2	77.1
BART-base w/ Fine Tuning + Input Perturbation	47.0	67.3	79.1	64.8	82.2	78.8
Training on our Singlish message corpus						
BART-base w/ Fine Tuning	53.3	78.9	81.8	69.2	83.9	80.3
BART-base w/ Fine Tuning + Input Perturbation	56.3	80.1	83.0	71.4	84.5	84.8
Training on the merged dataset						
BART-base w/ Fine Tuning	55.0	79.7	83.5	70.2	85.7	84.1
BART-base w/ Fine Tuning + Input Perturbation	57.8	81.1	84.6	72.3	86.1	86.5

Table 9: Corpus combination experiment with automatic evaluation scores. Models are trained separately on English lexical normalization and our Singlish message data. The proposed text perturbation is conducted on all input samples at the training stage. ROUGE and BERTScore reported here are F1 scores.

paraphrasing task of Singlish-to-standard English into three sub-tasks: lexical level normalization, syntactic level editing, and semantic level rewriting. Based on this linguistic hierarchy, we constructed an annotated dataset and reported baseline performance via fine-tuning language backbones, and further robustified the neural models with linguistically-inspired input perturbation. Experiment on a downstream stance detection task showed better performance when the input (colloquial English or Singlish) is de-noised by our paraphrasing model, suggesting that models developed using the data we build could help normalize noisy user-generated text. Our task definition, annotation protocol, constructed corpus, and reported base results pave the way for future studies on creole and colloquial language processing.

Acknowledgments

This research was supported by funding from the Institute for Infocomm Research (I2R) under A*STAR ARES, Singapore. We thank Hai Leong Chieu for the insightful discussions, and Jia Yi Chan, Nabilah Binte Md Johan, Siti Maryam Binte Ahmad Subaidi, and Siti Umairah Md Salleh for linguistic resource construction, and human evaluation. We also thank the anonymous reviewers for their precious feedback to help improve and extend this piece of work.

References

Noëmi Aepli and Rico Sennrich. 2022. Improving zero-shot cross-lingual transfer between closely related languages by injecting character-level noise. In *Findings of the Association for Computational Linguistics: ACL 2022*, pages 4074–4083, Dublin, Ireland. Association for Computational Linguistics.

Roee Aharoni, Melvin Johnson, and Orhan Firat. 2019. Massively multilingual neural machine translation. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 3874–3884.

Kemal Altintas and Ilyas Cicekli. 2002. A machine translation system between a pair of closely related languages. In *Proceedings of the 17th International Symposium on Computer and Information Sciences (ISCIS 2002)*, pages 192–196. Citeseer.

AiTi Aw, Min Zhang, Juan Xiao, and Jian Su. 2006. A phrase-based statistical model for sms text normalization. In *Proceedings of the COLING/ACL 2006 Main Conference Poster Sessions*, pages 33–40.

Alexandra Balahur and Marco Turchi. 2012. Multilingual sentiment analysis using machine translation? In *Proceedings of the 3rd workshop in computational approaches to subjectivity and sentiment analysis*, pages 52–60.

Timothy Baldwin, Marie-Catherine de Marneffe, Bo Han, Young-Bum Kim, Alan Ritter, and Wei Xu. 2015. Shared tasks of the 2015 workshop on noisy user-generated text: Twitter lexical normalization and named entity recognition. In *Proceedings of the Workshop on Noisy User-generated Text*, pages 126–135.

Satanjeev Banerjee and Alon Lavie. 2005. Meteor: An automatic metric for mt evaluation with improved correlation with human judgments. In *Proceedings of the acl workshop on intrinsic and extrinsic evaluation measures for machine translation and/or summarization*, pages 65–72.

Ana-Maria Bucur, Adrian Cosma, and Liviu P Dinu. 2021. Sequence-to-sequence lexical normalization with multilingual transformers. In *Proceedings of the Seventh Workshop on Noisy User-generated Text (W-NUT 2021)*, pages 473–482.

Tao Chen and Min-Yen Kan. 2013. Creating a live, public short message service corpus: the nus sms corpus. *Language Resources and Evaluation*, 47(2):299–335.

Eugenio Coseriu. 1981. Los conceptos de dialecto, nivel y estilo de lengua y el sentido propio de la dialectología. *LEA: Lingüística española actual*, 3(1):1–32.

Marta R Costa-jussà, Marcos Zampieri, and Santanu Pal. 2018. A neural approach to language variety translation. In *Proceedings of the Fifth Workshop on NLP for Similar Languages, Varieties and Dialects (VarDial 2018)*, pages 275–282.

Michel Anne Frederic Degraff. 1992. *Creole grammars and acquisition of syntax: The case of Haitian*. Ph.D. thesis, University of Pennsylvania.

David Deterding. 2007. *Singapore English*. Edinburgh University Press.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies,*, pages 4171–4186.

Jacob Eisenstein. 2013. What to do about bad language on the internet. In *Proceedings of the 2013 conference of the North American Chapter of the association for computational linguistics: Human language technologies*, pages 359–369.

Akiko Eriguchi, Melvin Johnson, Orhan Firat, Hideto Kazawa, and Wolfgang Macherey. 2018. Zero-shot cross-lingual classification using multilingual neural machine translation. *arXiv preprint arXiv:1809.04686*.

Jan Hajič, Jan Hric, and Vladislav Kuboň. 2000. Machine translation of very close languages. In *Proceedings of the sixth conference on Applied natural language processing*, pages 7–12.

Bo Han and Timothy Baldwin. 2011. Lexical normalisation of short text messages: Makn sens a# twitter. In *Proceedings of the 49th annual meeting of the association for computational linguistics: Human language technologies*, pages 368–378.

John Harris et al. 1993. Losing and gaining a language: the story of kriol in the northern territory. *Language and culture in Aboriginal Australia*, page 145.

Mian Lian Ho, John Talbot Platt, et al. 1993. *Dynamics of a contact continuum: Singaporean English*. Clarendon Press.

Pierre-Edouard Honnet, Andrei Popescu-Belis, Claudiu Musat, and Michael Baeriswyl. 2018. Machine translation of low-resource spoken dialects: Strategies for normalizing swiss german. In *Proceedings of the Eleventh International Conference on Language Resources and Evaluation (LREC 2018)*.

Vladimir Karpukhin, Omer Levy, Jacob Eisenstein, and Marjan Ghazvininejad. 2019. Training on synthetic noise improves robustness to natural noise in machine translation. *W-NUT 2019*, page 42.

Diederik P Kingma and Jimmy Ba. 2015. Adam: A method for stochastic optimization. In *Proceedings of the 3rd International Conference for Learning Representations*.

Jakob RE Leimgruber. 2011. Singapore english. *Language and Linguistics Compass*, 5(1):47–62.

Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Veselin Stoyanov, and Luke Zettlemoyer. 2020. Bart: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. In *Proceedings of the ACL 2020*, pages 7871–7880.

Chin-Yew Lin. 2004. Rouge: A package for automatic evaluation of summaries. In *Text summarization branches out*, pages 74–81.

Yinhan Liu, Jiatao Gu, Naman Goyal, Xian Li, Sergey Edunov, Marjan Ghazvininejad, Mike Lewis, and Luke Zettlemoyer. 2020. Multilingual denoising pre-training for neural machine translation. *Transactions of the Association for Computational Linguistics*, 8:726–742.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Roberta: A robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692*.

Marco Lui, Jey Han Lau, and Timothy Baldwin. 2014. Automatic detection and language identification of multilingual documents. *Transactions of the Association for Computational Linguistics*, 2:27–40.

Saif Mohammad, Svetlana Kiritchenko, Parinaz Sobhani, Xiaodan Zhu, and Colin Cherry. 2016. A dataset for detecting stance in tweets. In *Proceedings of the Tenth International Conference on Language Resources and Evaluation (LREC'16)*, pages 3945–3952.

Benjamin Muller, Benoît Sagot, and Djamé Seddah. 2019. Enhancing bert for lexical normalization. In *The 5th Workshop on Noisy User-generated Text (W-NUT)*.

Rafael Müller, Simon Kornblith, and Geoffrey Hinton. 2019. When does label smoothing help? *arXiv preprint arXiv:1906.02629*.

Preslav Nakov and Jörg Tiedemann. 2012. Combining word-level and character-level models for machine translation between closely-related languages. In *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 301–305.

Dat Quoc Nguyen, Thanh Vu, and Anh-Tuan Nguyen. 2020. Bertweet: A pre-trained language model for english tweets. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 9–14.

Toan Q Nguyen and David Chiang. 2017. Transfer learning across low-resource, related languages for neural machine translation. *IJCNLP 2017*, page 296.

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. Bleu: a method for automatic evaluation of machine translation. In *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics*, pages 311–318.

Alec Radford, Jeff Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. 2019. Language models are unsupervised multitask learners.

Richard Sproat, Alan W Black, Stanley Chen, Shankar Kumar, Mari Ostendorf, and Christopher Richards. 2001. Normalization of non-standard words. *Computer speech & language*, 15(3):287–333.

Dmitry Supranovich and Viachaslau Patsepnia. 2015. Ihs_rd: Lexical normalization for english tweets. In *Proceedings of the Workshop on Noisy User-generated Text*, pages 78–81.

Izumi Suzuki, Yoshiki Mikami, Ario Ohsato, and Yoshihide Chubachi. 2002. A language and character set determination method based on n-gram statistics. *ACM Transactions on Asian Language Information Processing (TALIP)*, 1(3):269–278.

Rob van der Goot, Alan Ramponi, Arkaitz Zubiaga, Barbara Plank, Benjamin Muller, Iñaki San Vicente Roncal, Nikola Ljubešić, Özlem Çetinoğlu, Rahmad Mahendra, Talha Çolakoğlu, Timothy Baldwin, Tommaso Caselli, and Wladimir Sidorenko. 2021. MultiLexNorm: A shared task on multilingual lexical normalization. In *Proceedings of the Seventh Workshop on Noisy User-generated Text (W-NUT 2021)*, pages 493–509, Online. Association for Computational Linguistics.

Rob van der Goot and Gerardus van Noord. 2017. Monoise: Modeling noise using a modular normalization system. *Computational Linguistics in the Netherlands Journal*, 7:129–144.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *In Advances in Neural Information Processing Systems*, pages 6000–6010.

Hongmin Wang, Yue Zhang, GuangYong Leonard Chan, Jie Yang, and Hai Leong Chieu. 2017. Universal dependencies parsing for colloquial singaporean english. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1732–1744.

Pidong Wang, Preslav Nakov, and Hwee Tou Ng. 2016. Source language adaptation approaches for resource-poor machine translation. *Computational Linguistics*, 42(2):277–306.

Lionel Wee. 2008. *Singapore English: morphology and syntax*, pages 2250–2264. De Gruyter Mouton.

John Wieting and Kevin Gimpel. 2018. Paranmt-50m: Pushing the limits of paraphrastic sentence embeddings with millions of machine translations. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 451–462.

Marcos Zampieri, Shervin Malmasi, Yves Scherrer, Tanja Samardžić, Francis Tyers, Miikka Silfverberg, Natalia Klyueva, Tung-Le Pan, Chu-Ren Huang, Radu Tudor Ionescu, et al. 2019. A report on the third vardial evaluation campaign. Association for Computational Linguistics.

Marcos Zampieri, Preslav Nakov, and Yves Scherrer. 2020. Natural language processing for similar languages, varieties, and dialects: A survey. *Natural Language Engineering*, 26(6):595–612.

Rabih Zbib, Erika Malchiodi, Jacob Devlin, David Stallard, Spyros Matsoukas, Richard Schwartz, John Makhoul, Omar Zaidan, and Chris Callison-Burch. 2012. Machine translation of arabic dialects. In *Proceedings of the 2012 conference of the north american chapter of the association for computational linguistics: Human language technologies*, pages 49–59.

Tianyi Zhang, Varsha Kishore, Felix Wu, Kilian Q. Weinberger, and Yoav Artzi. 2020. Bertscore: Evaluating text generation with BERT. In *8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, 2020*.

Xiaoheng Zhang. 1998. Dialect mt: a case study between cantonese and mandarin. In *COLING 1998 Volume 2: The 17th International Conference on Computational Linguistics*.

Liwei Zhao, Karin Kipper, William Schuler, Christian Vogler, Norman Badler, and Martha Palmer. 2000. A machine translation system from english to american sign language. In *Conference of the Association for Machine Translation in the Americas*, pages 54–67. Springer.

A Appendix

Discourse Particle	Original Text	Paraphrased Text
'leh' marks a tentative suggestion or request.	Still eating. Got free mcflurry. U (leh), going back liao..	I am still eating. I got a free McFlurry. What about you? I am going back.
'hor' attempts to garner support for a proposition.	I go can (hor)..	Is it alright for me to go?
'wot' marks obviousness and contradiction.	Datz (wot) i tld u 2 go 4 sleep...	That is why I told you to go to sleep...
'lor' indicates obviousness or resignation.	Yar (lor)... How u noe? U used dat route too?	Yes... How do you know? you used that route too?
'mah' marks information as obvious.	Lol that is my lunch timing (mah) wat time u want?	Lol that is my lunch time. What time do you want?
'meh' indicates skepticism.	s quite 4 me... U dun find it accurate (meh)...	It is quite for me... Don't you find it accurate?
'liao' present past tense.	They have done the test (liao)	They have already done the test.
'lah' is a mood marker, and appeals for accommodation.	She should be quite safe (lah)	She should be quite safe.
'one' indicates topic prominence (a syntax to emphasize the topic–comment structure).	This country weather very hot (one).	The weather is very warm in this country.

Table 10: Selected clause-final discourse particles of colloquial Singlish. In our annotation protocol, the particles that only convey (weak) emphasis will not trigger the semantic rewriting, such as *'ya'* and *'mah'*.

Model Type	Text Content
Source Input	Hehe cumi dnt mind much for litle quarels in ua lyf.. J0y n sadnes b0th are a part of lyf.. Try to smile in pain N atlast i'm wid u f0reva as a frnd..
Human Reference	Hehe Cumi did not mind much for the little quarrels in your life... Joy and sadness are part of life... Try to smile in pain and at least I am with you forever as your friend.
BART-large w/ Fine Tuning	Hehe cumi don't mind much for little quarels in your life.. Jokes and sadnes are a part of life.. Try to smile in pain and at last i'm with you f0reva as a friend.
BART-large w/ Input Perturbation	Hehe Cumi do not mind much for little quarels in your life.. Joy and sadness both are a part of life.. Try to smile in pain and at last I am with you forever as a friend..
Source Input	Woot.wan ask u ST de interview how de ah. will ask simi kind of thing.
Human Reference	Woot. I want to ask you how is the ST interview like. What kind of thing will they ask?
BART-large w/ Fine Tuning	Woot. Want to ask you ST the interview. Will ask simi kind of thing.
BART-large w/ Input Perturbation	Woot. I want to ask you ST how the interview. I will ask some kind of thing.
Source Input	Ahh I know lol. Like that if im the nominated one, walk pass also paiseh la
Human Reference	Ah, I know lol. If I am nominated, it will be embarrassing to walk past.
BART-large w/ Fine Tuning	I know lol. Like that if I am the nominated one, walk pass also sorry.
BART-large w/ Input Perturbation	Ahh, I know lol. Like that if I am the nominated one, walk pass also sorry.
Source Input	Haha. I was kidding lah. Yaloh. Yaloh. I'll try dessert also. Tml uintro their zao pai cai ar. Haha.
Human Reference	Haha. I was kidding. Yes. Yes. I will try dessert also. Tomorrow you will introduce their most famous dish. Haha.
BART-large w/ Fine Tuning	Haha. I was kidding. Yaloh. I will try dessert also. Tomorrow uintro their zao pai cai. Haha.
BART-large w/ Input Perturbation	Haha. I was kidding. Yes. I will try dessert also. Tomorrow I will sample their zao pai cai. Haha.
Source Input	baobei still syncing... dropbox so slow.. dar wait awhile morr
Human Reference	Baby, I am still syncing it... Dropbox is so slow... Dear, wait for a while.
BART-large w/ Fine Tuning	Baobei, it is still syncing...Dropbox is so low... Darling wait awhile more..
BART-large w/ Input Perturbation	Baby it is still syncing... Dropbox is so slow... darling, wait a while..

Table 11: Examples of Singlish message paraphrasing, and the generated text from human annotation and models. To improve the readability, here we only color the spans with sub-optimal changes in purple.

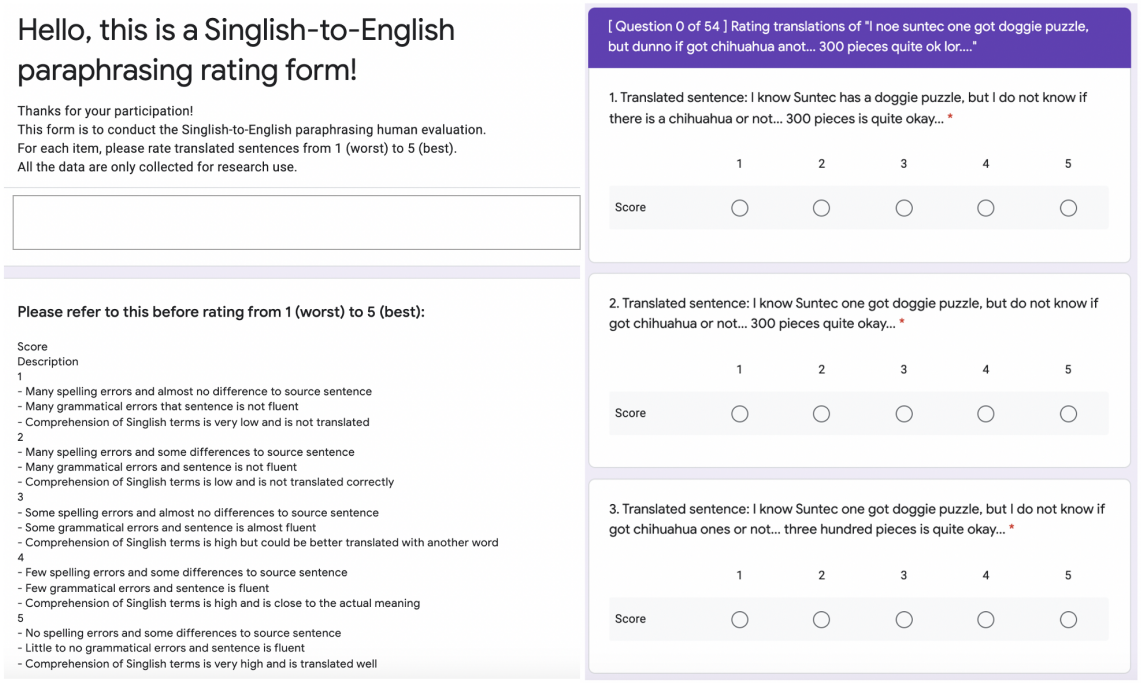

Figure 3: The rating form template for the human evaluation described in Section 5.4. Text candidates are shuffled for each sample to reduce the order bias, and we average the scores from all raters.

Environment Details	
GPU Model	Single Tesla V100 with 16 GB memory; CUDA version 10.1.
Library Version	Pytorch==1.7.1; Transformers==4.8.2.
Computational Cost	Average 1.5 hours training time for one round. Average 3 rounds for each reported result (calculating mean of the result scores).
Hyper-parameter	**Setting Detail**
Paraphrasing Task	
Neural Generator	Vanilla Transformer (12-layer, 768-hidden, 16-heads, 135M parameters). BART-base (12-layer, 768-hidden, 16-heads, 139M parameters). BART-large 24-layer, 1024-hidden, 16-heads, 406M parameters. mBART-large-50 (24-layer, 1024-hidden, 16-heads, 610M parameters).
Learning Rate and Batch Size	We set the learning rate (2e-5) and batch size (16) according to regular language model fine-tuning strategy (Lewis et al., 2020).
Beam Search Size	We evaluated beam search sizes from 3 to 10, and 5 provided the best balance of performance and inference speed.
Label Smoothing Weight	We set the label smoothing weight λ at 0.1 for fine-tuning on language generation work (Lewis et al., 2020).
BERTScore Metrics	We use the RoBERTa-base version of BERTScore (Zhang et al., 2020).
Stance Classification Task	
Corpus	The corpus we used for stance detection is from a published work (Mohammad et al., 2016), where all data are anonymized, and only for research use.
Neural Classifier	Bertweet-base (12-layer, 768-hidden, 12-heads, 130M parameters). RoBERTa-base (12-layer, 768-hidden, 12-heads, 125M parameters). Bertweet-base (12-layer, 768-hidden, 12-heads, 130M parameters).
Learning Rate and Batch Size	We set the learning rate (2e-5) and batch size (32) according to regular language model fine-tuning strategy (Devlin et al., 2019).

Table 12: Details of the experimental environment and the hyper-parameter setting.

CINO: A Chinese Minority Pre-trained Language Model

Ziqing Yang[†], **Zihang Xu**[†], **Yiming Cui**[‡†*], **Baoxin Wang**[†], **Min Lin**, **Dayong Wu**[†], **Zhigang Chen**[†§]

[†]State Key Laboratory of Cognitive Intelligence, iFLYTEK Research, Beijing, China
[‡]Research Center for SCIR, Harbin Institute of Technology, Harbin, China
[§]Jilin Kexun Information Technology Co., Ltd., Changchun, China
[†]{zqyang5,zhxu13,ymcui,bxwang2,dywu2,zgchen}@iflytek.com
[‡]ymcui@ir.hit.edu.cn

Abstract

Multilingual pre-trained language models have shown impressive performance on cross-lingual tasks. It greatly facilitates the applications of natural language processing on low-resource languages. However, there are still some languages that the current multilingual models do not perform well on. In this paper, we propose CINO (Chinese Minority Pre-trained Language Model), a multilingual pre-trained language model for Chinese minority languages. It covers Standard Chinese, Yue Chinese, and six other ethnic minority languages. To evaluate the cross-lingual ability of the multilingual model on ethnic minority languages, we collect documents from Wikipedia and news websites, and construct two text classification datasets, WCM (Wiki-Chinese-Minority) and CMNews (Chinese-Minority-News). We show that CINO notably outperforms the baselines on various classification tasks. The CINO model and the datasets are publicly available at http://cino.hfl-rc.com.

1 Introduction

The multilingual pre-trained language model (MPLM) is known for its ability to understand multiple languages, and its surprising zero-shot cross-lingual ability (Wu and Dredze, 2019). The zero-shot cross-lingual transfer ability enables the MPLM to be applied on the target languages with limited or even no annotated data by fine-tuning the MPLM on the source language with rich annotated data. MPLMs greatly facilitate transferring the current NLP technologies to low-resource languages and reduce the cost of developing NLP applications for low-resource languages.

The existing public MPLMs such as mBERT (Devlin et al., 2019), XLM (Conneau and Lample, 2019) and XLM-R (Conneau et al., 2020) can handle 100 languages, but there are still some challenges on low-resource languages understanding:

- The size of pre-training corpora of some low-resource languages is small compared to the high-resource languages. This bias towards high-resource languages may harm the performance on low-resource languages.
- There are thousands of living languages in the world, but many languages have not been covered in the existing MPLMs, especially indigenous or ethnic minority languages. For example, Tibetan, a language spoken mainly by Tibetans around Tibetan Plateau, is absent from the CC-100 corpus. Therefore, the XLM-R tokenizer can not tokenize Tibetan scripts correctly, and XLM-R is not good at understanding Tibetan texts.

Recently, more advanced MPLMs have been proposed, such as ERNIE-M (Ouyang et al., 2021), VECO (Luo et al., 2021) and Unicoder (Huang et al., 2019). These models focus on multilingual training objectives, such as leveraging parallel sentences to improve the alignment between different languages, and have improved notably over XLM-R. However, these models have not paid attention to the low-resource languages, so the problem remains unsolved.

For the above reasons, it is necessary to develop multilingual pre-trained language models for low-resource and ethnic minority languages. In this paper, we focus on Chinese minority languages. In China, Standard Chinese (Mandarin Chinese) is the predominant language. Besides Standard Chinese, we consider several most spoken minority languages. These languages are in different language families with varying writing systems, as summarized in Table 1.

Although each of the listed minority languages is spoken by at least millions of people, their digital corpus resources are quite limited. For example, in

*Email corresponding.

Proceedings of the 29th International Conference on Computational Linguistics, pages 3937–3949
October 12–17, 2022.

ISO Code	Language Name	Language Family	Writing System
zh	Standard Chinese (Mandarin)	Sino-Tibetan	Chinese characters
yue	Yue Chinese (Cantonese)	Sino-Tibetan	Chinese characters
bo	Tibetan	Sino-Tibetan	Tibetan script
mn	Mongolian	Mongolic	Traditional Mongolian script
ug	Uyghur	Turkic	Uyghur Arabic alphabet
kk	Kazakh	Turkic	Kazakh Arabic alphabet
za	Zhuang	Kra-Dai	Latin alphabet
ko	Korean	Isolate	Hangul

Table 1: Families and writing systems of the languages covered by CINO.

the CC-100 corpus used by XLM-R, the size of the Uyghur (ug) corpus is 0.4 GB, which is about 1% of the Chinese (Simplified) corpus (46.9 GB); also, there are no Tibetan (bo) or (traditional) Mongolian (mn) corpora in the CC-100.

We propose a multilingual pre-trained language model named CINO (**C**hinese M**ino**rity Pre-trained Language Model), which covers Standard Chinese, Yue Chinese (Cantonese) and six ethnic minority languages. As far as we know, this is the first multilingual pre-trained language model for the Chinese minority languages. CINO largely has the same structure as XLM-R and has been adapted for minority languages by resizing its vocabulary and adopting a fast masked language modeling objective for the pre-training.

The reason for training a multilingual pre-trained model rather than multiple monolingual pre-trained models is threefold. First, a multilingual model is more convenient than multiple monolingual models. Second, for low-resource languages, multilingual pre-training leads to better performance than monolingual pre-training (Conneau et al., 2020; Wu and Dredze, 2020). Third, a multilingual pre-trained model provides cross-lingual transfer ability, which reduces the data annotation cost for low-resource languages. Studies have also shown that pre-training with more languages leads to better cross-lingual performance on low-resource languages (Conneau et al., 2020).

The public natural language understanding tasks in Chinese minority languages are extremely limited. In this work, we construct two multilingual datasets from two data sources to support evaluating the zero-shot cross-lingual ability of MPLMs on the Chinese minority languages: (1) The **WCM** (**W**iki-**C**hinese-**M**inority) dataset is a multilingual text classification dataset built from Wikipedia corpora, with 10 classes, consisting of 63k examples. (2) **CMNews** (**C**hinese **M**inority **N**ews) dataset is a multilingual news classification dataset with 8 classes, built from the crawled news and the pre-existing news datasets, consisting of 57k examples.

To evaluate CINO from different perspectives, we run experiments on Tibetan News Classification Corpus (TNCC), Korean news topic classification (YNAT), WCM, and CMNews. Results show that CINO has acquired the ability of minority language understanding and outperforms the existing baselines on the Chinese minority languages.

To summarize, our contributions are:

- We introduce CINO, the first multilingual pre-trained language model for Chinese minority languages. Besides Standard Chinese, CINO covers Yue Chinese and six ethnic minority languages.
- We construct two multilingual text classification datasets for Chinese minority languages. They are used for evaluating the cross-lingual and multilingual abilities of the ethnic minority language model.
- Experiments show that CINO achieves notable improvements over the baselines. Furthermore, by making the model public, CINO will be a useful resource on Chinese minority languages and facilitate related research.

2 Related Work

2.1 Pre-trained Language Models

Multilingual Pre-trained Language Models. Devlin et al. (2019) introduced the first multilingual pre-trained language model mBERT trained with Masked Language Modeling (MLM). Conneau and Lample (2019) proposed Translation Language Modeling (TLM) to train the multilin-

gual model with cross-lingual supervision. Since then, various kinds of multilingual pre-training objectives have been proposed. Unicoder (Huang et al., 2019) trains the model with the objectives including cross-lingual word recovery, cross-lingual paraphrase classification and cross-lingual MLM. InfoXLM (Chi et al., 2021) proposed a pre-training task based on contrastive learning from an information-theoretic perspective. Pan et al. (2021) also introduced an alignment method based on contrastive learning. Cao et al. (2020) proposed an explicit word-level alignment procedure. ERNIE-M (Ouyang et al., 2021) integrates back-translation into the pre-training process. VECO (Luo et al., 2021) uses a cross-attention module to build the interdependence between languages explicitly. In this work, we only use non-parallel data and an objective similar to MLM for pre-training CINO.

Non-English Pre-trained Language Models and Benchmarks. Many pre-trained models have been trained on English corpora, or corpora that are heavily biased toward English. To make NLP techniques accessible to people from different cultures, researchers have developed pre-trained models and benchmarks targeting different languages: FlauBERT and the FLUE benchmark for French (Le et al., 2020), KLUE-BERT and the KLUE benchmark for Korean (Park et al., 2021), IndoBERT and the IndoLEM benchmark for Indonesian (Koto et al., 2020), and there are Chinese-BERT-wwm (Cui et al., 2021) and Arabic BERT AraBERT (Antoun et al., 2020). However, there are no pre-trained language models targeting Chinese ethnic minority languages.

2.2 Language Diversity in China

There are 56 ethnic groups and more than 80 languages in China. Standard Chinese (Mandarin) is the official language, spoken mainly by ethnic Han Chinese, which accounts for more than 90% of the total population. Ethnic minorities have their own languages. According to the study in Moseley (2010), the ethnic minority languages Mongolian, Uyghur, Kazakh, Tibetan,Yi, and Korean are safe (five of them are covered by CINO), which are spoken by about 25 million people, while the rest are in unsafe or endangered status.

Besides the ethnic minority languages, there are dialects and varieties of Chinese across the country. In this work, we consider Yue Chinese (also known as Cantonese), a widely used group of varieties of Chinese in Southern China and have been carried by immigrants to Southeast Asia and many other parts of the world.

Some languages in Table 1 are spoken and widely used in more than one country, such as Korean, Mongolian and Kazakh. In this work, we named them as *minority* languages based on their status in China.

3 CINO Model

In this section, we present the CINO model structure and the pre-training methodology. We denote by N the number of pre-training languages, $\mathcal{C}_i$ the monolingual corpus of the ith language ($i = 1, \ldots, N$). Let n_i be the number of sentences and l_i be the mean sequence length in $\mathcal{C}_i$. Let c_i represent the total number of tokens of $\mathcal{C}_i$.

3.1 Model Structure

CINO is a multilingual transformer-based model with the same architecture as XLM-R. For the CINO-base, it has 12 layers, 768 hidden states, and 12 attention heads; for the CINO-large, it has 24 layers, 1024 hidden states, and 16 attention heads. The main differences between CINO and XLM-R are the word embeddings and the tokenizer. We start from the word embeddings and the tokenizer of XLM-R and adapt them for the minority languages by vocabulary extension and vocabulary pruning, as depicted in Figure 1.

Vocabulary Extension. The original XLM-R tokenizer does not recognize Tibetan scripts and Traditional Mongolian scripts, so we extend the XLM-R tokenizer and XLM-R word embeddings matrix with additional tokens.

We train sentence-piece tokenizers for Tibetan and Mongolian on their monolingual pre-training corpora respectively. Each of the tokenizers has a vocabulary size of 16,000. Then we merge the vocabularies from the Tibetan and Mongolian tokenizers into the original XLM-R tokenizer. The merged tokenizer has a vocabulary size of 274,701.

To extend the word embeddings, we resize the original word embeddings matrix of shape $V \times D$ to $V' \times D$ by appending new rows, where D is the hidden size, V is the original vocabulary size, V' is the new vocabulary size. The new rows represent the word vectors of the new tokens from the merged tokenizer. They are initialized with a Gaussian distribution of mean 0.0 and variance 0.02.

Vocabulary Pruning. Next, we prune the word

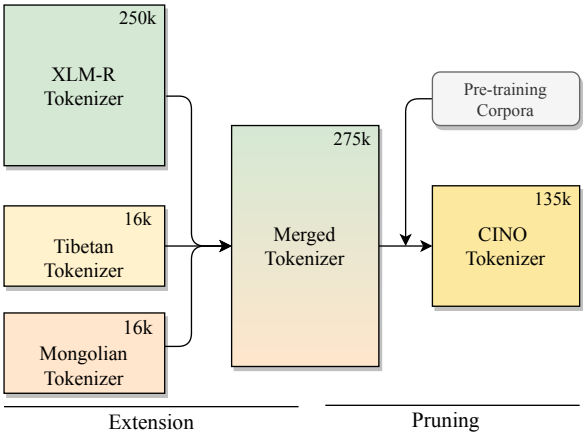

Figure 1: We extend the XLM-R tokenizer with a Tibetan tokenizer and a Mongolian tokenizer, then remove the redundant tokens to obtain the CINO tokenizer.

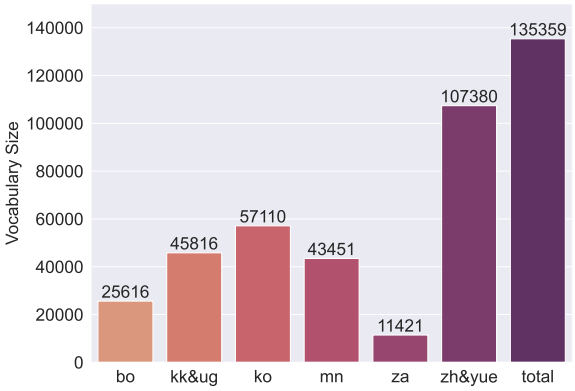

Figure 2: The vocabulary size counted from the corpus of each language. We merge the vocabularies of the languages that have similar writing systems.

embeddings matrix to reduce the model size. We tokenize the pre-training corpora with the merged tokenizer, and remove all the tokens that have not appeared in the corpora from the merged tokenizer's vocabulary and the word embeddings matrix. The above process discards 139,342 tokens.

Finally, we obtain the CINO model structure with a vocabulary size of 135,359, a model size of 728 MB for the base model, 1.7 GB for the large model, 68% and 79% size of XLM-R-base and XLM-R-large, respectively. A smaller vocabulary size leads to not only a memory-friendly model but also a faster model by reducing the cost of computing the log-softmax in the MLM task. The time cost of each iteration in pre-training is reduced by approximately 35% by reducing the vocabulary size from 270k to 140k.

3.2 Pre-training

We adopt the MLM objective for pre-training. In addition, we apply the following strategies for balancing training data and faster pre-training.

3.2.1 Resampling Strategy

To balance the data size between high-resource and low-resource languages, Conneau and Lample (2019) and Chi et al. (2021) have applied a multinomial sampling strategy. An example in the ith language is sampled with the probability

$$p_i = \frac{n_i^\alpha}{\sum_k^N n_k^\alpha}, \quad (1)$$

where $\alpha \in (0, 1]$ is a hyperparameter.

However, if the mean sequence lengths of different corpora are different, it may lead to an undesired data bias.[1] To see this, we use $\tilde{c}_i$ to denote the number of tokens seen during training. We have $\tilde{c}_i \propto p_i l_i$ and $\tilde{c}_i = K c_i$ for all $i = 1 \dots N$ if $\alpha = 1$. K is a constant that only depends on the number of training steps. If two languages i and j that have the same number of tokens, i.e., $c_i = c_j$, but with $n_i > n_j$ and $l_i < l_j$. With the sampling ratio in (1), we get $\tilde{c}_i < \tilde{c}_j$ if $\alpha < 1$ although the original corpora are of the same size. To remedy this, we introduce the dependence on the mean sequence length l_i. The sampling probability is

$$p_i = \frac{n_i^\alpha / l_i^\beta}{\sum_k^N n_k^\alpha / l_k^\beta}, \quad (2)$$

where $\beta \in [0, 1]$. Setting $\beta = 1 - \alpha$, the number of training tokens in the ith language is

$$\tilde{c}_i \propto p_i l_i \propto n_i^\alpha l_i^{1-\beta} = (n_i l_i)^\alpha = c_i^\alpha. \quad (3)$$

Therefore, corpora of equal size will be trained with an equal number of tokens.

3.2.2 Fast Masking Language Modeling

Table 1 shows that the languages we consider have distinguished writing systems, which implies that the vocabulary of each language only takes up a fraction of the whole vocabulary, as shown in Figure 2. By taking advantage of this fact, the computational costs can be reduced if the model only makes MLM predictions over the vocabulary of the specific language of the input examples rather than the whole vocabulary.

[1] In most cases, we could join short sequences to form long sequences of a uniform length. But some corpora we use consist of short sentences. Joining them as a long sequence leads to semantically incoherence.

Suppose the example is in the ith language. We denote by $\mathcal{V}$ the full vocabulary, and $\mathcal{V}_i \subset \mathcal{V}$ the vocabulary of the ith language, which is obtained by tokenizing the ith language's monolingual corpus. Let (c, x) denote the input text sequence, where x is the masked token, and c is the context. By limiting the prediction of the masked token to $\mathcal{V}_i$, the MLM loss of the masked token x is

$$\mathcal{L}_{\text{MLM}}^{(i)} = -\log \frac{\exp(g(c) \cdot E(x))}{\sum_{x' \in \mathcal{V}_i} \exp\left(g(c) \cdot E(x')\right)}, \quad (4)$$

where $g(\cdot)$ is the transformer encoder and $E(\cdot)$ is the look-up operation that returns the embeddings.

In order to calculate the loss (4) efficiently, during training, we group examples by language so that each batch contains examples in a single language.

With the objective (4) for pre-training, we have observed 10% time reduction and no significant performance drop compared to the original MLM objective, which predicts over the whole vocabulary. Combined with the speedup by vocabulary pruning, the pre-training time cost is reduced by about 40% in total.

4 Text Classification Datasets for Minority Languages

Multilingual tasks have been used widely to evaluate the cross-lingual transferability of multilingual models (Hu et al., 2020). Nevertheless, the pre-existing multilingual datasets hardly cover the Chinese ethnic minority languages. For example, Tibetan, Mongolian and Uyghur have never appeared in any task in the XTREME benchmark. To evaluate the cross-lingual transferability of CINO, we construct two text classification datasets **WCM** (**W**ikipedia-**C**hinese-**M**inority) and **CMNews** (**C**hinese-**M**inority-**N**ews).

4.1 WCM Dataset

Data Collection and Annotation. WCM is based on the data from Wikipedia. It covers seven languages: Mongolian, Tibetan, Uyghur, Kazakh, Korean, Cantonese, and Standard Chinese. We build the dataset from the Wikipedia page dumps and the Wikipedia category dumps[2] of the languages in question.

To annotate the data, we first generate a category graph for each language. Each node represents a category, and each edge stands for the affiliation between a pair of categories. By referring to the category system of Chinese Wikipedia, we choose ten categories for the classification task: Art, Geography, History, Nature, Science, Personage, Technology, Education, Economy, and Health. Then, we start from the categories of each page and backtrack along the routes in the category graph until reaching one of the ten target categories, and we set this category as the label of that page. Owing to some affiliation conflicts, like one subcategory belonging to two categories simultaneously, we reconstructed the graph by removing certain edges between the 10 target categories and their subcategories which are assessed as unreasonable by our human evaluation team.

Data Cleaning. After getting the labeled data, we apply several strategies to improve the quality of the datasets. We remove dirty data like large blocks of URLs and file paths. Then, the examples are filtered by their lengths (after being tokenized by the CINO tokenizer) by removing those examples shorter than 20 or longer than 1024 tokens.[3]

Subsampling. Since there are both high-resource languages like Korean and low-resource languages like Uyghur, we down-sample the data in the high-resource languages and the high-resource categories to balance the numbers of examples among different languages and different categories. We fix the size of the training set (Chinese articles) to 32K and downsample the datasets of the languages with abundant articles to about $5\% \sim 20\%$ size of the training set. Similarly, we also downsampled some categories if they dominate in some languages. We did not apply the above process to Uyghur due to its extreme scarcity.

Finally, we obtain 63,137 examples. WCM contains the train/dev/test set for Standard Chinese and only test sets for other languages. The detailed distribution is listed in Appendix C.

4.2 CMNews Dataset

Data Collection and Annotation. To collect the minority language examples, we crawl the news from the news websites in ethnic minority languages and record the category to which each news item belongs. To collect the Chinese news, we reuse the pre-existing dataset SogouCS News (Wang et al., 2008) and CAIL 2018 (Xiao et al., 2018). We select the appropriate categories and

[2] https://dumps.wikimedia.org/other

[3] We discard examples that are too long because long examples likely cover multiple topics while we assign a single label to each example.

Dataset		mn	bo	ug	kk	ko	yue	zh	Total
WCM	# Samples	27	5	4	52	43	49	20	200
	# Correctly Labeled	24	4	4	49	34	43	19	177
	Matching Acc	88.9%	80%	100%	94.2%	79.1%	87.8%	95.0%	88.5%
CMNews	# Samples	11	34	24	14	10	23	84	200
	# Correctly Labeled	8	31	24	14	10	20	80	187
	Matching Acc	72.7%	91.2%	100%	100%	100%	87.0%	95.2%	93.5%

Table 2: Results of human evaluation of the sampled examples from WCM and CMNews.

down-sample the two datasets to make the whole dataset more balanced.

After gathering the raw data from all the languages, we first merge the categories that have similar meanings (for example, we merge the categories *Finance* and *Economy*). Since the definition of news category may vary from website to website and language to language, we remove the categories that are not consistent in different languages by manually checking a sampled subset. We also remove the categories that do not appear in more than two languages. Finally, we obtain a dataset containing eight categories: Education, Sports, Health, Tourism, Legal, Economy, Culture, and Society.

Data Cleaning. The crawled news is much cleaner than the Wikipedia pages, and each document naturally belongs to only one category. Therefore we only perform length filtering by keeping the documents that contain more than 30 tokens after tokenization.

The dataset contains 56,764 examples in total. We split the dataset into a training set and a development set. The detailed distribution is listed in Appendix C.

4.3 Human Evaluation

To assess the quality of the datasets, we randomly sample 200 examples from WCM and 200 examples from CMNews and manually check whether the contents of the examples match their labels. The results are shown in Table 2. **Matching Acc** denotes how many examples match their labels under human evaluation. We find that 88.5% of the sampled examples from WCM and 93.5% of the sampled examples from CMNews are correctly labeled, which shows CMNews has less noise.

5 Experiments

5.1 Pre-training Setup

Pre-training Data. We randomly sample a subset dataset from the public base version of WuDaoCorpora (Yuan et al., 2021) as the Standard Chinese corpus; the corpora of the minority languages are in-house data, consisting of short monolingual sentences. The total corpora size is 28 GB. The statistics of the pre-training corpora are listed in Appendix A.

Experiment Settings. CINO is trained with the fast MLM objective (4) with the masking probability is 0.2 and the max sequence length 256. We initialize the parameters of CINO with XLM-R. We use the AdamW optimizer (Loshchilov and Hutter, 2019) with the peak learning rate of 2e-4 for the base model and 1e-4 for the large model. The learning rate is scheduled with 10k and 5k warmup steps followed by a linear decay for the base and the large model respectively. The sampling hyperparameter α is set to 0.7. We train the model with the batch size of 4,096 for 150k steps for the base model, and the batch size of 8,192 for 75k steps for the large model. The pre-training is performed on 16 NVIDIA A100 GPUs. The full pre-training hyperparameters are summarized in Appendix B.1.

5.2 Downstream Evaluation

How does CINO perform on the newly introduced languages? How does CINO perform on the languages pre-existing in XLM-R? Does CINO show multilingual and cross-lingual abilities? To answer these questions, we evaluate CINO on (1) Tibetan News Classification Corpus (Qun et al., 2017) (**TNCC**); (2) Korean news topic classification (Park et al., 2021) (**YNAT**); (3) **WCM** and **CMNews**. On TNCC and YNAT,[4] we evaluate the in-language model performance, i.e., we train and evaluate the model on the same language. On WCM and CMNews, we evaluate the cross-lingual ability. We describe the details in Section 5.4.

For each task and each model, we run the experiment five times with different seeds and report the mean metrics. The fine-tuning hyperparameters of

[4]The splitting sizes of TNCC and YNAT are listed in Appendix C.

each experiment are listed in Appendix B.2.

5.3 Baselines

Besides the common multilingual pre-trained models **mBERT** and **XLM-R**, we compare CINO models with the following baselines on some tasks.

XLM-R-Ext. We extend and prune the vocabulary of XLM-R as described in Section 3.1. This model is the un-pretrained CINO. The embeddings of Tibetan and Mongolian are randomly initialized, and the other parameters are the same as XLM-R.

KLUE-BERT-base. This is a Korean pre-trained model proposed in Park et al. (2021). Although KLUE-BERT-base is a base-sized model, it outperforms other large models on the YNAT task except for XLM-R-large.

TextCNN is a simple and light-weight model for text classification tasks (Kim, 2014). The word embedding dimension is set to 300. After the embedding layer, we apply three convolution layers in parallel with the number of out-channels 100, kernel size 3,4, and 5, respectively. Finally, we concatenate the outputs from the convolution layers and apply a two-layer fully-connected network with ReLU activation to perform the classification. We train the TextCNN from scratch with randomly initialized model parameters and word embeddings.

Word2vec (Tibetan). We first train the word embeddings using word2vec (Mikolov et al., 2013a,b) on the TNCC training set. The embedding dimension is set to 300. To perform the classification task, we average the word embeddings of each sample, then feed the results to a trainable linear layer that outputs the logits.

Model	TNCC Dev		TNCC Test	
	Acc	Macro-F1	Acc	Macro-F1
TextCNN	69.4	65.7	62.8	66.6
Word2vec (Tibetan)	70.1	67.7	70.2	68.0
base models				
mBERT	22.9	4.8	22.8	5.5
mBERT (p.t.)	63.9	56.2	61.8	56.4
XLM-R-base	35.1	20.2	31.1	21.1
XLM-R-base (p.t.)	34.2	21.5	31.4	19.9
XLM-R-Ext-base	55.7	43.2	55.0	42.1
CINO-base	74.8	71.4	73.1	70.0
large models				
XLM-R-large	35.7	26.4	32.8	27.3
XLM-R-Ext-large	31.6	13.0	29.2	12.2
CINO-large	**76.3**	**73.7**	**75.4**	**72.9**

Table 3: Model performance on the Dev and Test sets of Tibetan text classification task TNCC. *p.t.* is short for *pre-tokenized*.

Model	YNAT Dev	
	Acc	Macro-F1
mBERT (Park et al., 2021)	-	82.6†
XLM-R-base (Park et al., 2021)	-	84.5†
XLM-R-large (Park et al., 2021)	-	87.3†
KLUE-RoBERTa-large (Park et al., 2021)	-	85.9†
KLUE-BERT-base (Park et al., 2021)	-	87.0†
base models		
mBERT	82.9	82.8
XLM-R-base	85.1	85.0
KLUE-BERT-base	87.0	**87.1**
CINO-base	86.1	85.9
large models		
XLM-R-large	87.0	86.8
CINO-large	**87.3**	87.0

Table 4: Model performance on the Dev set of Korean text classification task YNAT. The results marked with † are taken from the KLUE paper (Park et al., 2021). The rest results are from our experiments.

5.4 Results and Discussions

5.4.1 TNCC

How does CINO perform on the newly introduced language? We evaluate CINO on TNCC, a Tibetan classification dataset with 12 classes. The original work (Qun et al., 2017) proposes a news title classification and a news document classification. Here we conduct the news document classification only. The task is to predict the topic of each document. Because there are no official splits available, we split the dataset into a training set, a development set and a test set with a ratio of 8:1:1. Since the texts in the dataset have been pre-tokenized (spaces have been added between words), we remove the spaces between words and tokenize the texts with the pre-trained tokenizer unless otherwise specified. We select the best checkpoint based on its macro-F1 score. We also report the accuracy score for reference.

The results are listed in Table 3. Compared among the pre-trained models, XLM-R series have low scores since the vocabulary is not adapted for the Tibetan language and has not been pre-trained on the Tibetan corpus. While XLM-R-Ext-base has an extended vocabulary and significantly outperforms XLM-R-base even without being pre-trained on the target language. Finally, by pre-training on the minority languages corpora, CINO is adapted to the new language and outperforms XLM-R and XLM-R-Ext notably.

mBERT achieves better results when fine-tuned

	Model	bo	kk	ko	mn	ug	yue	zh	Avg (Minorities)	Avg (All)
WCM	*base models*									
zh → min.	XLM-R-base	19.0	16.7	43.2	15.2	23.3	58.3	78.1	29.3	36.2
	CINO-base	36.2	43.2	**44.9**	39.1	**33.4**	59.7	78.0	42.6	47.6
	large models									
	XLM-R-large	18.4	32.9	43.8	22.2	27.8	**60.0**	77.3	34.2	40.3
	CINO-large	**40.6**	**44.8**	**44.8**	**41.6**	28.8	59.8	**79.2**	**43.3**	**48.4**
	Model	**bo**	**kk**	**ko**	**mn**	**ug**	**yue**	**zh**	**Avg (Minorities)**	**Avg (All)**
CMNews	*base models*									
min. → zh	XLM-R-base	38.1	69.6	88.3	35.1	77.5 (67.7/88.6)	**87.8**	58.6	66.1	65.0
	CINO-base	85.5	79.2	89.0	77.3	77.4 (77.0/78.0)	86.9	68.8	82.6	80.6
	large models									
	XLM-R-large	30.1	80.8	88.9	30.8	**85.1** (76.4/91.0)	87.5	63.6	67.2	66.7
	CINO-large	**86.8**	**83.0**	**90.3**	**79.4**	78.8 (68.4/91.3)	**87.9**	**71.2**	**84.4**	**82.5**

Table 5: Model performance on the WCM and CMNews. The metric on each language is macro-F1. **Avg (Minorities)** is the mean score over languages other than zh; **Avg (All)** is the mean score over all languages. We bold any score within 0.1 of the best on each language. The results in the parentheses are the min and the max values of five runs.

on the pre-tokenized data (but there are still many tokens being mapped to `[UNK]`). Due to the difference in the tokenization algorithms used by mBERT and XLM-R, XLM-R does not benefit from using pre-tokenized data.

TextCNN and Word2vec (Tibetan) surprisingly achieve competitive scores and outperforms XLM-R-Ext-base. It is possibly due to the difficulty in the optimization of large models such as XLM-R with limited training data. As we continue increasing the model size, the performance gets worse, as can be seen from comparing the scores of XLM-R-base-Ext and XLM-R-large-Ext.

5.4.2 YNAT

How does CINO perform on the minority languages pre-existing in XLM-R? We evaluate CINO on YNAT, a Korean text classification dataset with 7 classes. We select the best checkpoint based on its macro-F1 score. The results are listed in Table 4. CINO-base outperforms XLM-R-base, while CINO-large is better than XLM-R-large by our reimplementation but lower than the score reported in Park et al. (2021). CINO-large is also comparable to KLUE-BERT-base.

Notice that Korean is not a low-resource language in XLM-R (the size of the Korean corpus is 54 GB in the CC-100), thus XLM-R may have learned Korean well. To significantly outperform XLM-R and KLUE-BERT-base, we expect that longer training time and more data are required.

5.4.3 WCM and CMNews

Does CINO show multilingual and cross-lingual abilities? We use these two datasets to evaluate the cross-lingual and multilingual abilities. We take macro-F1 as the metric on each language, and the *Avg* is the arithmetic mean of the macro-F1 scores.

On the WCM dataset, we train models on the Chinese training set and test it on all the languages, so the results show how well the model transfers the knowledge from Chinese to the minority languages; the best checkpoint of each run is selected based on its score on Chinese; On the CMNews dataset, we train models on the minority languages and the Chinese data is zero-shot; the best checkpoint is selected based on its score on minority languages. The results are listed in Table 5.

On WCM, **Avg (Minorities)** score shows that CINO has superior zero-shot performance over XLM-R. By inspecting the detailed performance on each language, we see that CINO most significantly outperforms XLM-R on Tibetan, Kazakh, Mongolian and Uyghur, which have been insufficiently pre-trained in XLM-R.

On CMNews, because CINO has been adapted to minority languages, it learns more effectively than XLM-R by leveraging the examples in all the languages. **zh** score shows that CINO transfers better than XLM-R. CINO also outperforms XLM-R on almost all the minority languages except for **ug**, where there is a large gap. To find out the reason, we list the min and the max **ug** scores of five runs. We see that there is a large variance.

CINO-large achieves the highest score among all runs, but its average score is lower than XLM-R-large. The unstable performance may be the main reason that explains the gap.

6 Discussion on Limitations

Coverage of ethnic minority languages. Due to the scarcity of minority language corpora, CINO only covers Standard Chinese and some of the most popular minority languages and dialects. While being spoken by millions of people, some languages, such as the Yi language, are omitted in this study since we can not find sufficient data for pre-training.

Pre-training objectives. In our early trials of multilingual pre-training, we leveraged both monolingual and bilingual parallel data, and combined the MLM objective with a cross-lingual alignment objective, similar to the TLM objective used in Chi et al. (2021) and Conneau and Lample (2019). Intuitively, parallel data contain more information than monolingual data. However, we have not observed significant improvements over pre-training with only monolingual data and the MLM objective. The performance of CINO may be improved if parallel data can be effectively used.

Languages from different cultures. Among the languages in Table 1, some are cross-border languages. The cross-border languages are spoken in more than one country and are influenced by local cultures. How well does the model that has been trained on the corpus collected in one country transfer to the corpus collected in another country? If the writing systems of the language are different (for example, Mongolian is written in Cyrillic in Mongolia, while it is written in traditional Mongolian script in China), to what extent do writing systems influence the model performance? We expect future work to address these questions.

7 Conclusion

In this paper, we introduce CINO, a multilingual pre-trained language model for Chinese minority languages. It takes the same structure as XLM-R but with a different vocabulary and is pre-trained with an adapted MLM objective to reduce computational costs. We build multilingual text classification datasets WCM from Wikipedia and CMNews from ethnic minority news for zero-shot ability evaluation on the Chinese minority languages. We evaluate CINO on several text classification tasks. The results show that CINO achieves notable improvements over the existing baselines.

Acknowledgments

We would like to thank all anonymous reviewers for their thorough review and for providing constructive comments to improve our paper.

References

Wissam Antoun, Fady Baly, and Hazem Hajj. 2020. AraBERT: Transformer-based model for Arabic language understanding. In *Proceedings of the 4th Workshop on Open-Source Arabic Corpora and Processing Tools, with a Shared Task on Offensive Language Detection*, pages 9–15, Marseille, France. European Language Resource Association.

Steven Cao, Nikita Kitaev, and Dan Klein. 2020. Multilingual alignment of contextual word representations. In *8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, April 26-30, 2020*. OpenReview.net.

Zewen Chi, Li Dong, Furu Wei, Nan Yang, Saksham Singhal, Wenhui Wang, Xia Song, Xian-Ling Mao, Heyan Huang, and Ming Zhou. 2021. InfoXLM: An information-theoretic framework for cross-lingual language model pre-training. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 3576–3588, Online. Association for Computational Linguistics.

Alexis Conneau, Kartikay Khandelwal, Naman Goyal, Vishrav Chaudhary, Guillaume Wenzek, Francisco Guzmán, Edouard Grave, Myle Ott, Luke Zettlemoyer, and Veselin Stoyanov. 2020. Unsupervised cross-lingual representation learning at scale. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 8440–8451, Online. Association for Computational Linguistics.

Alexis Conneau and Guillaume Lample. 2019. Cross-lingual language model pretraining. In *Advances in Neural Information Processing Systems 32: Annual Conference on Neural Information Processing Systems 2019, NeurIPS 2019, December 8-14, 2019, Vancouver, BC, Canada*, pages 7057–7067.

Yiming Cui, Wanxiang Che, Ting Liu, Bing Qin, and Ziqing Yang. 2021. Pre-training with whole word masking for chinese bert. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 29:3504–3514.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for*

Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers), pages 4171–4186, Minneapolis, Minnesota. Association for Computational Linguistics.

Junjie Hu, Sebastian Ruder, Aditya Siddhant, Graham Neubig, Orhan Firat, and Melvin Johnson. 2020. Xtreme: A massively multilingual multi-task benchmark for evaluating cross-lingual generalization. *CoRR*, abs/2003.11080.

Haoyang Huang, Yaobo Liang, Nan Duan, Ming Gong, Linjun Shou, Daxin Jiang, and Ming Zhou. 2019. Unicoder: A universal language encoder by pre-training with multiple cross-lingual tasks. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 2485–2494, Hong Kong, China. Association for Computational Linguistics.

Yoon Kim. 2014. Convolutional neural networks for sentence classification. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1746–1751, Doha, Qatar. Association for Computational Linguistics.

Fajri Koto, Afshin Rahimi, Jey Han Lau, and Timothy Baldwin. 2020. IndoLEM and IndoBERT: A benchmark dataset and pre-trained language model for Indonesian NLP. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 757–770, Barcelona, Spain (Online). International Committee on Computational Linguistics.

Hang Le, Loïc Vial, Jibril Frej, Vincent Segonne, Maximin Coavoux, Benjamin Lecouteux, Alexandre Allauzen, Benoit Crabbé, Laurent Besacier, and Didier Schwab. 2020. FlauBERT: Unsupervised language model pre-training for French. In *Proceedings of the 12th Language Resources and Evaluation Conference*, pages 2479–2490, Marseille, France. European Language Resources Association.

Ilya Loshchilov and Frank Hutter. 2019. Decoupled weight decay regularization. In *7th International Conference on Learning Representations, ICLR 2019, New Orleans, LA, USA, May 6-9, 2019*. OpenReview.net.

Fuli Luo, Wei Wang, Jiahao Liu, Yijia Liu, Bin Bi, Songfang Huang, Fei Huang, and Luo Si. 2021. VECO: Variable and flexible cross-lingual pre-training for language understanding and generation. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 3980–3994, Online. Association for Computational Linguistics.

Tomás Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013a. Efficient estimation of word representations in vector space. In *1st International Conference on Learning Representations, ICLR 2013, Scottsdale, Arizona, USA, May 2-4, 2013, Workshop Track Proceedings*.

Tomás Mikolov, Ilya Sutskever, Kai Chen, Gregory S. Corrado, and Jeffrey Dean. 2013b. Distributed representations of words and phrases and their compositionality. In *Advances in Neural Information Processing Systems 26: 27th Annual Conference on Neural Information Processing Systems 2013. Proceedings of a meeting held December 5-8, 2013, Lake Tahoe, Nevada, United States*, pages 3111–3119.

C. Moseley. 2010. *Atlas of the World's Languages in Danger*. Memory of peoples Series. UNESCO Publishing.

Xuan Ouyang, Shuohuan Wang, Chao Pang, Yu Sun, Hao Tian, Hua Wu, and Haifeng Wang. 2021. ERNIE-M: Enhanced multilingual representation by aligning cross-lingual semantics with monolingual corpora. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 27–38, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Lin Pan, Chung-Wei Hang, Haode Qi, Abhishek Shah, Saloni Potdar, and Mo Yu. 2021. Multilingual BERT post-pretraining alignment. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 210–219, Online. Association for Computational Linguistics.

Sungjoon Park, Jihyung Moon, Sungdong Kim, Won Ik Cho, Jiyoon Han, Jangwon Park, Chisung Song, Junseong Kim, Yongsook Song, Taehwan Oh, Joohong Lee, Juhyun Oh, Sungwon Lyu, Younghoon Jeong, Inkwon Lee, Sangwoo Seo, Dongjun Lee, Hyunwoo Kim, Myeonghwa Lee, Seongbo Jang, Seungwon Do, Sunkyoung Kim, Kyungtae Lim, Jongwon Lee, Kyumin Park, Jamin Shin, Seonghyun Kim, Lucy Park, Alice Oh, Jung-Woo Ha, and Kyunghyun Cho. 2021. Klue: Korean language understanding evaluation.

Nuo Qun, Xing Li, Xipeng Qiu, and Xuanjing Huang. 2017. End-to-end neural text classification for tibetan. In *Chinese Computational Linguistics and Natural Language Processing Based on Naturally Annotated Big Data - 16th China National Conference, CCL 2017, - and - 5th International Symposium, NLP-NABD 2017, Nanjing, China, October 13-15, 2017, Proceedings*, volume 10565 of *Lecture Notes in Computer Science*, pages 472–480. Springer.

Radim Řehůřek and Petr Sojka. 2010. Software Framework for Topic Modelling with Large Corpora. In *Proceedings of the LREC 2010 Workshop on New Challenges for NLP Frameworks*, pages 45–50, Valletta, Malta. ELRA. http://is.muni.cz/publication/884893/en.

Canhui Wang, Min Zhang, Shaoping Ma, and Liyun Ru. 2008. Automatic online news issue construction in web environment. In *Proceedings of the 17th*

International Conference on World Wide Web, WWW 2008, Beijing, China, April 21-25, 2008, pages 457–466. ACM.

Shijie Wu and Mark Dredze. 2019. Beto, bentz, becas: The surprising cross-lingual effectiveness of BERT. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 833–844, Hong Kong, China. Association for Computational Linguistics.

Shijie Wu and Mark Dredze. 2020. Are all languages created equal in multilingual BERT? In *Proceedings of the 5th Workshop on Representation Learning for NLP*, pages 120–130, Online. Association for Computational Linguistics.

Chaojun Xiao, Haoxi Zhong, Zhipeng Guo, Cunchao Tu, Zhiyuan Liu, Maosong Sun, Yansong Feng, Xianpei Han, Zhen Hu, Heng Wang, and Jianfeng Xu. 2018. CAIL2018: A large-scale legal dataset for judgment prediction. *ArXiv preprint*, abs/1807.02478.

Sha Yuan, Hanyu Zhao, Zhengxiao Du, Ming Ding, Xiao Liu, Yukuo Cen, Xu Zou, Zhilin Yang, and Jie Tang. 2021. Wudaocorpora: A super large-scale chinese corpora for pre-training language models. *AI Open*, 2:65–68.

A Statistics of the Pre-training Corpora

The corpus size and mean sequence length for pre-training are listed in Table 6. The sequence lengths are obtained by counting the tokens after tokenization. For Standard Chinese (zh), we concatenate or truncate each example to the max sequence length, while for other languages, we do not concatenate the examples but keep them unchanged.

Language	# Tokens	Mean Sequence Length
bo	130M	13.4
kk	238M	60.7
ko	170M	20.0
mn	337M	25.7
ug	1B	23.1
yue	276M	12.6
za	23M	58.1
zh	1.2B	254

Table 6: Corpus size and mean sequence length of each language in the pre-training data.

B Hyperparameters

B.1 Pre-training Hyperparameters

Hyperparameter	Base Model	Large Model
Batch Size	4,096	8,192
Warmup Steps	10k	5k
Training Steps	150k	75k
Peak Learning Rate	2e-4	1e-4
Max Length	256	256
MLM probability	0.2	0.2
Adam ϵ	1e-8	1e-8
Adam β_1	0.9	0.9
Adam β_2	0.999	0.999
Gradient Clipping	1.0	1.0
Weight Decay	0	0
Sampling α	0.7	0.7

Table 7: Hyperparameters used for pretraining CINO models.

Table 7 presents the full set of the hyperparameters used for pre-training CINO models.

B.2 Fine-tuning Hyperparameters

The hyperparameters for fine-tuning on the downstream tasks is listed in Table 9. The batch size is 32 for all experiments except Word2vec (Tibetan), of which batch size is 16. The learning rate is scheduled with 10% warmup steps followed by a linear decay.

We use Gensim (Řehůřek and Sojka, 2010) to train the Word2vec embeddings, and set `min_count` $= 1$, `vector_size` $= 300$. Other parameters take the default values.

Dataset	# Train	# Dev	# Test	# Classes
TNCC	7,359	191	923	12
YNAT	45,678	9,106	-	7

Table 8: Number of examples in TNCC and YNAT.

C Statistics of the Datasets

The sizes of TNCC and YNAT are shown in Table 8. Detailed data distribution of WCM is listed in Table 10. Detailed data distribution of CMNews is listed in Table 11.

Model	TNCC		YNAT		WCM		CMNews	
	LR	Epochs	LR	Epochs	LR	Epochs	LR	Epochs
Word2vec (Tibetan)	3e-2	20	-	-	-	-	-	-
TextCNN	1e-4	40	-	-	-	-	-	-
mBERT	3e-5	40	2e-5	5	-	-	-	-
KLUE-BERT-base	-	-	3e-5	3	-	-	-	-
XLM-R-base	5e-5	40	3e-5	3	1e-5	20	3e-5	5
CINO-base	5e-5	40	3e-5	3	1e-5	20	3e-5	5
XLM-R-large	3e-5	40	2e-5	3	1e-5	20	3e-5	5
CINO-large	3e-5	40	2e-5	3	1e-5	20	3e-5	5

Table 9: Hyperparameters used for downstream fine-tuning.

Category	mn	bo	ug	kk	ko	yue	zh-train	zh-test	zh-dev
Arts	135	141	3	348	806	387	2657	335	331
Geography	76	339	256	572	1197	1550	12854	1644	1589
History	66	111	0	491	776	499	1771	248	227
Nature	7	0	7	361	442	606	1105	110	134
Natural Science	779	133	20	880	532	336	2314	287	317
Personage	1402	111	0	169	684	1230	7706	924	953
Technology	191	163	8	515	808	329	1184	152	134
Education	6	1	0	1392	439	289	936	118	130
Economy	205	0	0	637	575	445	922	109	113
Health	106	111	6	893	299	272	551	73	67
Total	2973	1110	300	6258	6558	5943	32000	4000	3995

Table 10: Number of examples in each category and language in WCM.

Split	Category	bo	kk	ko	mn	ug	yue	zh
Train	**Education**	626	364	378	187	423	880	1979
	Sports	66	133	321	556	1216	70	1978
	Health	1309	153	40	31	240	1358	2000
	Tourism	1128	12	43	102	1078	0	1998
	Legal	433	283	283	294	19	22	2000
	Economy	399	107	192	510	0	1080	1877
	Culture	1834	231	228	118	0	0	1995
	Society	898	149	147	543	1132	169	1935
	Total	6693	1432	1632	2341	4108	3579	15762
Dev	**Education**	418	243	253	125	282	587	1000
	Sports	44	89	215	371	811	48	1000
	Health	874	103	28	21	160	906	1000
	Tourism	752	8	30	68	719	0	1000
	Legal	289	190	189	196	14	15	1000
	Economy	266	72	129	341	0	721	1000
	Culture	1223	155	152	80	0	0	1000
	Society	600	100	99	362	756	113	1000
	Total	4466	960	1095	1564	2742	2390	8000

Table 11: Number of examples in each category and language in CMNews.

One Word, Two Sides: Traces of Stance in Contextualized Word Representations

Aina Garí Soler, Matthieu Labeau, Chloé Clavel
LTCI, Télécom-Paris, Institut Polytechnique de Paris, France
{aina.garisoler,matthieu.labeau,chloe.clavel}@telecom-paris.fr

Abstract

The way we use words is influenced by our opinion. We investigate whether this is reflected in contextualized word embeddings. For example, is the representation of "animal" different between people who would abolish zoos and those who would not? We explore this question from a Lexical Semantic Change standpoint. Our experiments with BERT embeddings derived from datasets with stance annotations reveal small but significant differences in word representations between opposing stances.

1 Introduction

Our opinions are reflected in the way we talk. People with opposing stances on a particular topic may use different words when talking about it. For example, only people against the use of face masks during the COVID-19 pandemic would sometimes refer to them as "muzzles". In this paper, however, we do not investigate *what* words are used by each side. Instead, we compare how speakers who disagree on a subject use the *same* words. Specifically, we want to know whether contextual models capture a difference between the representation of a word (e.g., "mask") when it is used by people who are in favor *vs.* against a certain target (e.g., the compulsory use of face masks).

We address this question from the perspective of Lexical Semantic Change (LSC). Work on LSC typically tries to detect word meaning changes across two or more periods of time (Tahmasebi et al., 2021), but its techniques have also been employed to identify synchronic differences in word usage, for instance across different ages, genders, professions (Gonen et al., 2020), domains (Yin et al., 2018; Schlechtweg et al., 2019), or cultures (Garimella et al., 2016). As opposed to related studies that investigate LSC between different viewpoints (Azarbonyad et al., 2017; Rodriguez et al., 2021), our goal is not to explore the usage of specific words, and we do not evaluate our method based on the ranking of words by meaning stability. We rather want to determine whether vector representations reflect a higher similarity in word usage within a stance than between different stances (see example in Figure 1). We explore this question relying on datasets annotated with stance information. Before that, we test different context-sensitive embedding models on a simulated scarce-data setting. This allows us to select a robust representation type that can identify the words that are used most differently between stances.

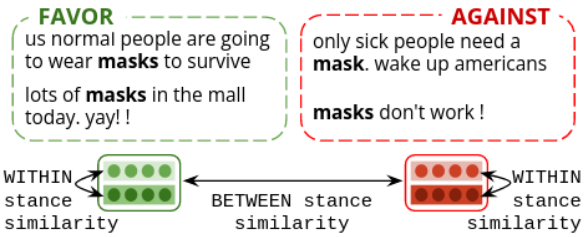

Figure 1: Example instances of "mask" from the Covid19 stance dataset (Glandt et al., 2021). We compare the within- and the between-stance usage similarity.

Our long-term goal is to detect differences in word usage between speakers in a conversation, which could point to their level of conceptual alignment (Stolk et al., 2016); that is, the extent to which dialog participants "mean the same things when using the same words" (Schober, 2005). In this study we present a first step in this direction. Representations that are sensitive to opinion differences could be useful to identify disagreements and misalignment in dialog.

2 Methodology

In this section we introduce the data and the models used in our experiments. We also describe our

Proceedings of the 29th International Conference on Computational Linguistics, pages 3950–3959
October 12–17, 2022.

similarity measure and the criteria for evaluation.[1]

2.1 Data

The datasets we use are in English and contain stance information in the form of sentences that are labeled as being in FAVOR or AGAINST a specific target. We exclude sentences with no (clear) stance (NONE), when present. **SemEval2016** (Mohammad et al., 2016b,a) contains tweets on six varied targets. We use 3,253 sentences.[2] **Covid19** (Glandt et al., 2021) is another dataset with 3,918 tweets centered on four targets related to the COVID-19 pandemic. **P-stance** (Li et al., 2021) is a large dataset containing 21,574 tweets about three US politicians. Finally, IBM-ArgQ-Rank-30kArgs (Gretz et al., 2020), hereafter **ArgQ**, is a collection of arguments on 71 targets which are annotated for stance, stance clarity and argument quality. We use 29,972 arguments that have a clear stance (with a confidence score[3] above 0.6, following Bar-Haim et al. (2020)).

We want to organize the data in a way that allows us to investigate whether instances of the same word have a higher similarity within a stance than between stances. To this end, we preprocess and organize the data as follows.

Preprocessing The ArgQ dataset was originally intended for argument quality detection, and several arguments mention their stance explicitly. To mitigate the potential biases that this could cause, we apply a strategy that we call *sentence trimming* which automatically omits this part of a sentence. We describe it in detail in Appendix A. Then we tokenize, postag and lemmatize sentences in all datasets. [4]

Sentence Sets For a given target, we randomly split the sentences of each stance (f or a) into two equally-sized sets P and Q. With these sets, we run four *comparisons*, two within-stance: WITHIN-FAVOR (P_f vs Q_f) and WITHIN-AGAINST (P_a vs Q_a); and two between-stance: BETWEEN-1 (P_f vs Q_a) and BETWEEN-2 (P_a vs Q_f).

[1]Our code and data are available at https://github.com/ainagari/1word2sides.

[2]We omit the target "Climate Change is a Real Concern" because it only has 26 AGAINST tweets.

[3]This score reflects the extent to which annotators agreed on the stance of an argument. It is calculated as a weighted average of the annotators' decisions and it ranges from 0 to 1.

[4]We use the default `nltk` functions, except for tweets, which we tokenize with `nltk`'s TweetTokenizer. Lemmatization is done with `nltk`'s WordNet Lemmatizer.

2.2 Vector Representations

We want to generate vector representations for sets of word instances within a stance (e.g., in P_f). For example, we want to obtain one representation of the word "woman" from sentences in favor of the "Feminist Movement" (SemEval2016) and compare it to the representation of "woman" in sentences expressing a stance against this target.

In LSC detection, static embeddings tend to perform better than contextualized ones (Schlechtweg et al., 2020). A typical approach is to learn static embeddings separately for each time period, corpus or viewpoint, and then compare them either by aligning them (Hamilton et al., 2016) or with a nearest-neighbors-based approach (Gonen et al., 2020). In these studies, even in those dealing with short-term change detection (Stewart et al., 2017; Del Tredici et al., 2019), it is common to have a fairly large amount of instances of a given word available. However, the number of available sentences per word within a stance in our data is limited.[5] We therefore experiment with three different types of contextualized embeddings:

À la carte embeddings (ALC) (Khodak et al., 2018) have been used to detect differences in word usage across viewpoints (Rodriguez et al., 2021). The model consists in applying a linear transformation to the averaged pre-trained embeddings of the context words surrounding the target word. We use an ALC model relying on 300*d* GloVe embeddings (Pennington et al., 2014) trained on 840B tokens from Common Crawl.

Context2vec (c2v) (Melamud et al., 2016) is a biLSTM model that generates embeddings for the context surrounding a word. It is optimized so that the representation of a context is similar to that of potential filler words. We use a 600*d* model trained on the ukWaC corpus (Baroni et al., 2009).

BERT (Devlin et al., 2019). We use contextualized representations generated with the 768*d* `bert-base-uncased` model. We explain how we choose the best layer in Section 2.3.

We denote the vocabulary of a sentence set (e.g. P) as V_P. We include in the vocabulary all nouns and verbs appearing in at least three different sen-

[5]As an example, Schlechtweg et al. (2020) have an average of 788 instances per lemma and time period; and Gonen et al. (2020) study words that appear at least 200 times in their corpus. In our data, the average amount of instances of a word in one side of a comparison is 14.

tences in P. In tweets, mentions and hashtags are treated as nouns. Stopwords are excluded. We treat all instances of a lemma with a specific part of speech (PoS) as the same word. We extract a vector representation $\mathbf{w}_P$ for every word w in V_P. For c2v and BERT, this is done by averaging the representations of all w instances in P.

2.3 Testing Representations

Before our experiments on stance, we first identify the vector representations that are best suited to reflect lexical semantic similarity between small sets of sentences. Following Schlechtweg and Schulte im Walde (2020), we use SemCor (Miller et al., 1993), a sense-annotated corpus, to create a dataset that simulates lexical semantic change. We additionally control for the amount of sentences available for each lemma. The process of creation of this dataset is explained in more detail in Appendix B.

The dataset consists of 576 lemmas: 245 nouns, 241 verbs, 69 adjectives and 21 adverbs. For every lemma we have two sets of 25 instances each, P and Q. To simulate situations of scarce data, we create X-sized subsets of P and Q (P_X, Q_X). We experiment with different values of X ($X \in \{3, 5, 10, 20, 25\}$). As in Schlechtweg and Schulte im Walde (2020), we determine the "true" semantic distance between two groups P_X and Q_X by calculating the Jensen-Shannon divergence (JSD) between their sense distributions.

Similarity predictions for a word w are obtained by simply calculating the cosine similarity between the representations of that lemma in each sentence set, $cos(\mathbf{w}_{P_X}, \mathbf{w}_{Q_X})$. We report the Kendall's tau-b correlation coefficient between JSD and the similarities predicted by each representation type. Results of this experiment are presented in Section 3.1.

2.4 Similarity Calculation

To calculate the global similarity in word usage for a comparison between two sets of sentences P and Q, we first identify the words that are common in both sets, $V_P \cap V_Q$. $V_P \cap V_Q$ contains words that are not necessarily central to the target that is being discussed. We therefore calculate a similarity based only on a subset of $V_P \cap V_Q$, which we call V_{PQ}. The similarity score is the average cosine similarity of all words in V_{PQ}:

$$sim(P, Q) = \frac{\sum_{w \in V_{PQ}} cos(\mathbf{w}_P, \mathbf{w}_Q)}{|V_{PQ}|} \quad (1)$$

This similarity measure is intended to reflect the extent to which words are used in the same way and in the same senses in two sentence sets. We experiment with three definitions of V_{PQ}. In all of them, we take care of using the same amount of words for all four comparisons within a target. In *all*, we include the top k most frequent words in $V_P \cap V_Q$, where k corresponds to the smallest size of $V_P \cap V_Q$ available for that target. Frequency is determined from the union of sentences in P and Q. We also use the top 10 words in $V_P \cap V_Q$ with highest tf-idf scores in that target (*tf-idf*). Tf-idf scores are calculated on the ensemble of stance datasets, treating all sentences about the same target as one document. Finally, we also use the 10 words in $V_P \cap V_Q$ with lowest tf-idf (*rev-tf-idf*). This subset contains words that are less relevant to the target, and therefore we expect BETWEEN- and WITHIN-stance similarities to have closer values in this setting. Note that 25% of comparisons (in SemEval2016 and ArgQ) have less than 20 words in common. In these cases, *tf-idf* and *rev-tf-idf* are partially calculated with the same words.

2.5 Evaluation

We expect WITHIN-stance comparisons to exhibit a higher average similarity than BETWEEN-stance comparisons. To measure the extent to which this holds, we use pairwise accuracy: we check for how many (WITHIN, BETWEEN) comparison pairs the BETWEEN comparison has a lower similarity. With 4 comparisons per target, our experiments involve a total of 332 (WITHIN, BETWEEN) pairs. Results on stance data are presented in Section 3.2.

3 Results

3.1 Selecting a Representation Type

Results on SemCor are shown in Figure 2. In plots *a* and *b*, we see the correlations obtained by the different representation types on various amounts of data (X). Naturally, performance is worse with lower values of X. This is especially the case of ALC embeddings, which at X=25 continue to improve. In the case of c2v and BERT, however, we do not observe big improvements after X=10. In this scarce-data setting, the performance of ALC

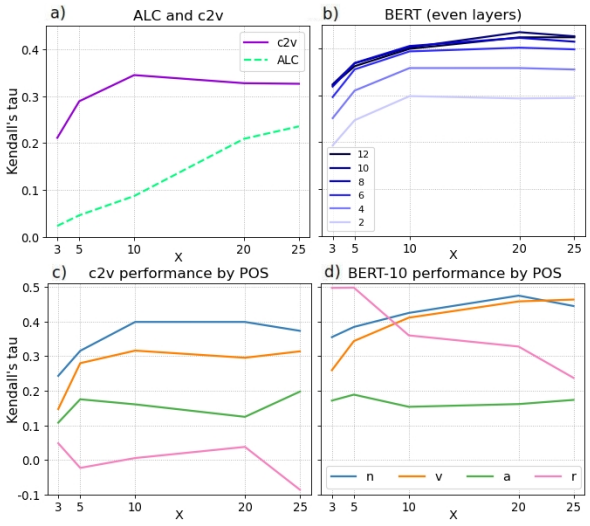

Figure 2: *a* and *b*: Kendall's tau obtained by different vector representations on SemCor. We only include even layers for BERT for better readability. *c* and *d*: Performance of c2v and BERT (10th layer) by PoS.

embeddings is much lower than that of c2v and BERT. Overall, BERT representations from the 10th layer work best. We therefore use embeddings from this layer for our experiments on stance data. We also look at the performance of the best two models (c2v and the 10th layer in BERT) by PoS (plots *c* and *d*): we find that nouns and verbs, the PoS included in our stance experiments, are generally better represented. We also make interesting observations regarding the other PoS. Despite the lower performance, adjective representations seem to be less affected by a smaller number of sentences. When it comes to BERT adverb representations, similarity estimations are more reliable at lower values of X. These differences in PoS should be taken into account when deriving type-level vectors from BERT representations.

3.2 Results on Stance

Pairwise accuracy obtained with the 10th BERT layer with different definitions of V_{PQ} is found in Table 1. We see that, especially for *all* and *tf-idf*, pairwise accuracy is remarkably high in all datasets. This shows that contextualized word representations from BERT reflect differences in the way words are used between two opposing stances.

When using the 10 words with lowest tf-idf (*rev-tf-idf*) performance decreases, but is still high in P-stance and ArgQ. We run chi-square goodness-of-fit tests on *rev-tf-idf* predictions to determine their likelihood under the null hypothesis (H_0 : acc $= 0.5$). P-values are significant for all datasets together ($p < 0.001$) but not for the set of Twitter datasets ($p = 0.08$, $\alpha = 0.05$).[6] It seems BERT representations do, to some extent, encode differences in words that are less relevant to the target. However, if for some reason not all words can be used (if there are too many), then it is preferable to select a subset carefully (e.g. with tf-idf).

Dataset	all	tf-idf	rev-tf-idf
SemEval2016	0.90	0.85	0.60
Covid19	0.88	0.81	0.50
P-stance	1.00	1.00	0.83
ArgQ	1.00	0.98	0.95
Global	0.99	0.96	0.90

Table 1: Pairwise accuracy by dataset and with different V_{PQ}. *Global* corresponds to all datasets put together.

	all	tf-idf	rev-tf-idf
a) W vs W	0.013	0.010	0.023
b) B vs B	0.013	0.010	0.023
c) W vs B	0.047	0.027	0.041

Table 2: Differences in similarity between comparisons.

We also examine the words that have the highest and the lowest similarities in BETWEEN comparisons; we provide this information in Appendix C. The words that are used most differently between stances tend to be nouns that are central to the topic (e.g. "religion" in "Atheism"), while the most similar words are often non-topical ("man" or "take"). In the middle of the distribution, in targets with a small common vocabulary (<30) we find words that are relevant to the topic, but in a less obvious way (e.g. "world" and "community" for the target "Missionary work"). In targets with a larger vocabulary we find a combination of relevant and non-relevant words.

We investigate how large the differences in similarity are between WITHIN (W) and BETWEEN (B) comparisons. We investigate this by looking at the differences in similarity (in absolute value) across comparison pairs: a) between WITHIN-FAVOR and WITHIN-AGAINST (W vs W), b) between BETWEEN-1 and BETWEEN-2 (B vs B), and c) the average difference found in the four WITHIN vs BETWEEN pairings (W vs B). We expect the latter to have a larger difference in similarity than

[6] This could be due to particularities of the language used in Twitter. We leave the use of models specialized on tweets (e.g. BERTweet (Nguyen et al., 2020)) for future work.

a) and b), where comparisons are of the same type. Results are shown in Table 2. We report the average of these values on all the data. Differences in similarity are quite low overall, indicating that the contrast (i.e., the extent to which WITHIN comparisons display a higher similarity than BETWEEN comparisons) is subtle. Values are, however, between 1.8 and 3.6 times larger for the W vs B comparison pairs. For all V_{PQ} definitions, the difference values in these comparison pairs are significantly different from those in a) and b) ($p < 0.001$).[7]

4 Conclusion and Future Work

We have shown that BERT word representations are sensitive to the opinion expressed in the sentences they are derived from. Differences in similarity found between concurring and conflicting stances are small, but significant; and words with the highest differences tend to be central to the topic. Our approach can serve to identify points of discrepancy with regard to a target, and it can be useful for stance detection and debate analysis. Our experiments on SemCor provide valuable insight on the sufficient amount of word instances needed to obtain quality representations. This is relevant for low-resource LSC and, more generally, for inferring word vectors from little data.

In future work, we plan to apply this methodology to dialog. Sets P and Q would each correspond to the utterances of one speaker in a conversation. The similarity measure would act as an approximation of the conceptual or *stance alignment* between the two participants, indicating whether speakers share opinions and use words in a similar way.

Acknowledgements

We thank the anonymous reviewers for their helpful feedback and insightful comments. This work was supported by the Télécom Paris research chair on Data Science and Artificial Intelligence for Digitalized Industry and Services (DSAIDIS).

References

Hosein Azarbonyad, Mostafa Dehghani, Kaspar Beelen, Alexandra Arkut, Maarten Marx, and Jaap Kamps. 2017. Words Are Malleable: Computing Semantic Shifts in Political and Media Discourse. In *Proceedings of the 2017 ACM on Conference on Information and Knowledge Management*, CIKM '17, page 1509–1518, New York, NY, USA. Association for Computing Machinery.

Roy Bar-Haim, Lilach Eden, Roni Friedman, Yoav Kantor, Dan Lahav, and Noam Slonim. 2020. From Arguments to Key Points: Towards Automatic Argument Summarization. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 4029–4039, Online. Association for Computational Linguistics.

Marco Baroni, Silvia Bernardini, Adriano Ferraresi, and Eros Zanchetta. 2009. The WaCky wide web: a collection of very large linguistically processed web-crawled corpora. *Journal of Language Resources and Evaluation*, 43(3):209–226.

Marco Del Tredici, Raquel Fernández, and Gemma Boleda. 2019. Short-Term Meaning Shift: A Distributional Exploration. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 2069–2075, Minneapolis, Minnesota. Association for Computational Linguistics.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, Minneapolis, Minnesota. Association for Computational Linguistics.

Aparna Garimella, Rada Mihalcea, and James Pennebaker. 2016. Identifying Cross-Cultural Differences in Word Usage. In *Proceedings of COLING 2016, the 26th International Conference on Computational Linguistics: Technical Papers*, pages 674–683, Osaka, Japan. The COLING 2016 Organizing Committee.

Kyle Glandt, Sarthak Khanal, Yingjie Li, Doina Caragea, and Cornelia Caragea. 2021. Stance Detection in COVID-19 Tweets. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 1596–1611, Online. Association for Computational Linguistics.

Hila Gonen, Ganesh Jawahar, Djamé Seddah, and Yoav Goldberg. 2020. Simple, Interpretable and Stable Method for Detecting Words with Usage Change across Corpora. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 538–555, Online. Association for Computational Linguistics.

Shai Gretz, Roni Friedman, Edo Cohen, Assaf Toledo, Dan Lahav, Ranit Aharonov, and Noam Slonim. 2020. A Large-Scale Dataset for Argument Quality Ranking: Construction and Analysis. *Proceedings of the*

[7] According to Wilcoxon or paired t-tests depending on normality (determined by Shapiro-Wilk tests).

AAAI Conference on Artificial Intelligence, 34:7805–7813.

William L. Hamilton, Jure Leskovec, and Dan Jurafsky. 2016. Diachronic Word Embeddings Reveal Statistical Laws of Semantic Change. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1489–1501, Berlin, Germany. Association for Computational Linguistics.

Mikhail Khodak, Nikunj Saunshi, Yingyu Liang, Tengyu Ma, Brandon Stewart, and Sanjeev Arora. 2018. A La Carte Embedding: Cheap but Effective Induction of Semantic Feature Vectors. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 12–22, Melbourne, Australia. Association for Computational Linguistics.

Yingjie Li, Tiberiu Sosea, Aditya Sawant, Ajith Jayaraman Nair, Diana Inkpen, and Cornelia Caragea. 2021. P-Stance: A Large Dataset for Stance Detection in Political Domain. In *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*, pages 2355–2365, Online. Association for Computational Linguistics.

Oren Melamud, Jacob Goldberger, and Ido Dagan. 2016. context2vec: Learning Generic Context Embedding with Bidirectional LSTM. In *Proceedings of the 20th SIGNLL Conference on Computational Natural Language Learning*, pages 51–61, Berlin, Germany. Association for Computational Linguistics.

George A. Miller, Claudia Leacock, Randee Tengi, and Ross T. Bunker. 1993. A Semantic Concordance. In *Human Language Technology: Proceedings of a Workshop Held at Plainsboro, New Jersey, March 21-24, 1993*.

Saif Mohammad, Svetlana Kiritchenko, Parinaz Sobhani, Xiaodan Zhu, and Colin Cherry. 2016a. A Dataset for Detecting Stance in Tweets. In *Proceedings of the Tenth International Conference on Language Resources and Evaluation (LREC'16)*, pages 3945–3952, Portorož, Slovenia. European Language Resources Association (ELRA).

Saif Mohammad, Svetlana Kiritchenko, Parinaz Sobhani, Xiaodan Zhu, and Colin Cherry. 2016b. SemEval-2016 Task 6: Detecting Stance in Tweets. In *Proceedings of the 10th International Workshop on Semantic Evaluation (SemEval-2016)*, pages 31–41, San Diego, California. Association for Computational Linguistics.

Dat Quoc Nguyen, Thanh Vu, and Anh Tuan Nguyen. 2020. BERTweet: A pre-trained language model for English Tweets. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 9–14, Online. Association for Computational Linguistics.

Jeffrey Pennington, Richard Socher, and Christopher Manning. 2014. Glove: Global Vectors for Word Representation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1532–1543, Doha, Qatar. Association for Computational Linguistics.

Pedro L Rodriguez, Arthur Spirling, and Brandon M Stewart. 2021. Embedding regression: Models for context-specific description and inference. Technical report, Working Paper Vanderbilt University.

Dominik Schlechtweg, Anna Hätty, Marco Del Tredici, and Sabine Schulte im Walde. 2019. A Wind of Change: Detecting and Evaluating Lexical Semantic Change across Times and Domains. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 732–746, Florence, Italy. Association for Computational Linguistics.

Dominik Schlechtweg, Barbara McGillivray, Simon Hengchen, Haim Dubossarsky, and Nina Tahmasebi. 2020. SemEval-2020 Task 1: Unsupervised Lexical Semantic Change Detection. In *Proceedings of the Fourteenth Workshop on Semantic Evaluation*, pages 1–23, Barcelona (online). International Committee for Computational Linguistics.

Dominik Schlechtweg and Sabine Schulte im Walde. 2020. Simulating Lexical Semantic Change from Sense-Annotated Data. In *The Evolution of Language: Proceedings of the 13th International Conference (EvoLang13)*.

Michael F Schober. 2005. Conceptual alignment in conversation. *Other minds: How humans bridge the divide between self and others*, pages 239–252.

Ian Stewart, Dustin Arendt, Eric Bell, and Svitlana Volkova. 2017. Measuring, Predicting and Visualizing Short-Term Change in Word Representation and Usage in VKontakte Social Network. *Proceedings of the International AAAI Conference on Web and Social Media*, 11(1):672–675.

Arjen Stolk, Lennart Verhagen, and Ivan Toni. 2016. Conceptual alignment: How brains achieve mutual understanding. *Trends in cognitive sciences*, 20(3):180–191.

Nina Tahmasebi, Lars Borin, and Adam Jatowt. 2021. Survey of computational approaches to lexical semantic change detection. In Nina Tahmasebi, Lars Borin, Adam Jatowt, Yang Xu, and Simon Hengchen, editors, *Computational approaches to semantic change*. Language Science Press.

Zi Yin, Vin Sachidananda, and Balaji Prabhakar. 2018. The global anchor method for quantifying linguistic shifts and domain adaptation. *Advances in neural information processing systems*, 31.

A Sentence Trimming

Sentence trimming is intended to omit a part of a sentence in the ArgQ dataset where stance is expressed explicitly. These sentences often start with the same words as the target. For example, for the target "Homeschooling should be banned", we find the sentence "'Homeschooling should not be banned because it is a right for parents to educate their children in their comfort of home". If the beginning of a sentence contains the same words as the target (with the optional addition of *not* and *n't*) and is followed by the token *because (of)*, *as*, *since*, a comma or a stop, we omit the first part of the sentence up to and including that token. In the example above, this results in the sentence "it is a right for parents to educate their children in their comfort of home". This procedure modifies 3,223 sentences. Some sentences with an explicit stance remain, but their number is importantly reduced. These include sentences starting with the target followed by connectors expressing effect (e.g., *so that*, *so as to*), which cannot be easily trimmed into a correct sentence or NP.

B Dataset for Testing Representations

In this section we describe in detail how we collect the data from SemCor (see Section 2.3). We randomly select 50 instances for every lemma that appears at least 50 times in SemCor. These instances are randomly split into two sets of 25 sentences each, P and Q. The X-sized subset of P, P_X, consists of the X first sentences in P. This approach results in a dataset with rather low JSD, especially for larger values of X. For example, for $X = 25$, the mean JSD is 0.22 and only 2% of lemmas have JSD > 0.5. To have a stronger representation of high JSD values, we maximize JSD for certain lemmas. We do this for a subset of the lemmas for which it is possible to find a P-Q split with zero sense overlap, such that JSD $= 1$. Enforcing these splits for $\sim$17% of all lemmas, the mean JSD for $X = 25$ goes up to 0.33.

C Highest- and Lowest-Similarity Words

Table 3 contains, for every target in our study, the words that differed the most and the least between FAVOR and AGAINST statements. Interestingly, among the top five most different words across all targets, we find a majority of nouns (85.9% nouns and 14.1% verbs). In the bottom five, instead, verbs are more common (38.1% nouns and 61.9% verbs).

Data	Target	Sentences	Most different words	Least different words
SemEval 2016	Feminist Movement	779	woman, men, equality woman, men, gender	come, leave, believe go, take, tell
	Hillary Clinton	728	@hillaryclinton, #hillaryclinton, woman @ hillaryclinton, #hillaryclinton, campaign	keep, world, go make, take, come
	Donald Trump	447	@realdonaldtrump, trump, #makeamericagreatagain @realdonaldtrump, trump, donald	want, give, take want, one, time
	Atheism	588	religion, #god, believe #freethinker, religion, god	man, think, go take, make, come
	Legalization of Abortion	711	abortion, woman, right abortion, woman, right	think, know, say take, carry, effect
Covid19	Face masks	1,361	mask, wear, people wear, mask, people	love, look, shut care, find, care
	Stay at home orders	590	#covid19, #coronavirus, virus #covid19, #coronavirus, virus	day, order, thing let, must, see
	Fauci	1,102	#drfauci, #coronavirus, #covid19 #drfauci, #covid19, #coronavirus	force, work, right leave, history, work
	School closures	865	@imbhupendrasinh, @vijayrupanibjp, school school, kid, @realdonaldtrump	time, do, need come, way, show
P-stance	Donald Trump	7,953	@realdonaldtrump, #donaldtrump, country @realdonaldtrump, #trump, say	color, head, pay arm, apply, wish
	Bernie Sanders	6,325	@berniesanders, bernie, #democraticdebate @berniesanders, bernie, sander	check, note, ill assume, knock, sick
	Joe Biden	7,296	#democraticdebate, @joebiden, #demdebate #democraticdebate, @joebiden, biden	name, sign, like dirt, tear, air
ArgQ	Marriage	413	marriage, people, couple marriage, couple, people	union, make, need create, become, thing
	Vow of celibacy	418	celibacy, vow, church celibacy, vow, people	need, take, way nothing, way, time
	Stay-at-home dads	392	home, dad, raise home, dad, men	make, provide, life time, allow, make
	Assisted suicide	392	suicide, assist, people suicide, assist, people	help, take, make death, take, make
	Fast food	416	food, eat, ban food, people, ban	health, make, issue world, make, time
	Urbanization	404	area, urbanization, city urbanization, people, area	space, create, grow population, make, create
	Missionary work	434	people, missionary, work work, people, missionary	make, take, way make, want, need
	Libertarianism	381	libertarianism, government, people libertarianism, government, people	lead, give, provide take, one, work
	Human cloning	416	clone, cloning, human cloning, clone, human	life, need, way make, thing, life
	Blockade of the Gaza Strip	506	strip, gaza, blockade strip, gaza, blockade	stop, right, state state, get, give
	Gender-neutral language	368	language, gender, people language, gender, people	offend, way, time make, feel, way
	Compulsory voting	405	voting, compulsory, vote vote, compulsory, people	make, way, want take, mean, could
	Zero-tolerance policy in schools	454	school, tolerance, student school, student, policy	lead, way, time way, make, time
	Payday loans	442	loan, people, need loan, money, people	situation, take, need take, make, give
	Whaling	423	whale (N), whaling, whale (V) whale (N), whale (V), whaling	help, way, need part, need, world
	Capital punishment	467	punishment, capital, death capital, punishment, crime	justice, make, serve way, give, time
	Cosmetic surgery for minors	494	minor, surgery, child surgery, minor, decision	thing, involve, give need, adult, cause

Data	Target	Sentences	Most different words	Least different words
ArgQ	School uniform	474	school, student, uniform school, uniform, student	stop, take, allow make, give, feel
	Foster care	529	child, kid, care child, parent, care	may, service, find become, make, put
	Polygamy	493	polygamy, legalize, marriage polygamy, marriage, woman	make, take, one way, make, time
	Prostitution	499	prostitution, legalize, prostitute prostitution, legalize, woman	give, allow, want choice, involve, want
	Zoos	395	animal, zoo, live animal, zoo, habitat	life, allow, make provide, keep, take
	The right to keep and bear arms	407	keep, bear, arm bear, keep, weapon	law, take, remove person, must, take
	Social media	330	medium, people, allow medium, people, allow	create, make, lose see, world, time
	Multi-party system	390	system, people, multiparty party, system, government	bring, need, allow choose, population, thing
	Nuclear weapons	542	weapon, country, use weapon, country, war	maintain, keep, life mean, make, world
	Homeschooling	395	child, homeschooling, school child, homeschooling, education	give, time, keep help, teacher, way
	Telemarketing	437	telemarketing (N), telemarketing (V), telemarketers telemarketing (V), telemarketing (N), telemarketers	allow, need, take money, work, time
	Entrampment	400	law, crime, entrapment crime, entrapment, commit	get, make, allow place, time, know
	Homeopathy	352	medicine, homeopathy, remedy homeopathy, medicine, people	harm, condition, placebo treat, cause, allow
	Intelligence tests	462	intelligence, people, person person, test, child	way, base, focus show, type, know
	Austerity regime	412	regime, austerity, economy regime, austerity, debt	spend, time, make reduce, pay, allow
	Child actors	435	actor, child, use actor, child, use	take, show, play take, make, lead
	Mandatory retirement	475	retirement, work, worker retirement, workforce, worker	make, position, force keep, provide, give
	Sex selection	400	selection, child, parent selection, baby, sex	allow, could, decide bear, right, way
	Economic sanctions	389	sanction, country, nation sanction, country, people	leader, make, take make, punish, help
	Intellectual property rights	415	property, right, product property, right, people	come, make, time time, take, think
	Use of public defenders	415	lawyer, defender, use defender, lawyer, defend	get, require, way person, mean, allow
	Guantanamo Bay detention camp	444	guantanamo, bay, detention guantanamo, detection, camp	serve, way, use law, make, usa
	Women in combat	370	combat, woman, men combat, woman, men	prohibit, could, make war, may, make
	Naturopathy	536	medicine, naturopathy, treatment naturopathy, medicine, treatment	lead, take, life seek, allow, make
	Church of Scientology	401	scientology, church, ban scientology, church, ban	member, believe, practice need, allow, practice
	Embryonic stem cell research	396	stem, cell (N), cell (V) cell, stem, research	help, need, use people, need, life
	Affirmative action	438	action, people, job action, people, discrimination	way, get, make school, way, work
	Cannabis	543	cannabis, marijuana, legalize cannabis, marijuana, drug	take, time, way may, allow, take

Data	Target	Sentences	Most different words	Least different words
ArgQ	Vocational education	418	education, school, subsidize education, subsidize, people	lead, make, way work, go, give
	Racial profiling	412	profiling, criminal, people profiling, people, crime	make, person, life stop, time, way
	Private military companies	392	company, ban, government company, government, military	could, make, time security, need, might
	Flag burning	426	burning, flag, burn flag, burning, burn	protect, freedom, make lead, protect, state
	Surrogacy	431	surrogacy, baby, woman surrogacy, woman, surrogate	right, become, term give, make, could
	Student loans	369	student, loan, education loan, student, subsidize	everyone, put, make afford, work, make
	Safe spaces	388	space, people, student space, people, others	life, may, thing make, allow, nothing
	Algorithmic trading	387	trading, people, market trading, computer, market	access, allow, base field, risk, lead
	Olympic games	409	olympic, game, olympics olympic, game, athlete	money, world, time give, time, take
	Journalism	357	journalism, news, subsidize journalism, subsidize, news	medium, need, could could, need, support
	Cosmetic surgery	425	surgery, people, appearance surgery, people, ban	make, take, lead feel, need, way
	Targeted killing	409	target, people, kill target, killing, people	use, state, take enemy, take, put
	Organ trade	408	trade, organ, sell trade, organ, legalize	give, death, way need, create, help
	Space exploration	381	space, exploration, subsidize space, exploration, planet	thing, support, country thing, find, use
	Factory farming	410	farm, factory, food factory, food, farming	space, allow, keep produce, keep, allow
	Pride parades	394	parade, pride, gay parade, pride, lgbt	right, allow, make way, want, bring
	Collectivism	440	collectivism, group, people collectivism, people, society	need, one, way take, lead, way
	Television	387	television, people, watch television, news, entertainment	way, thing, keep could, way, make
	School prayer	424	school, prayer, religion prayer, school, religion	allow, take, person part, time, place
	Autonomous cars	445	car, road, drive car, road, drive	cause, way, need take, use, time
	Holocaust denial	456	holocaust, denial, deny holocaust, denial, deny	speech, allow, go allow, world, say
	Executive compensation	375	executive, compensation, company executive, company, compensation	give, deserve, lead level, work, allow
	Three-strikes laws	490	law, strike, crime law, strike, people	take, make, need give, put, allow
	Atheism	360	atheism, god, religion atheism, religion, people	base, allow, make way, provide, lead
	Wikipedia	395	wikipedia, subsidize, information wikipedia, wikipedia, subsidize	could, need, take provide, way, give
	Judicial activism	385	judge, law, activism judge, activism, law	use, need, way allow, rule, could

Table 3: Words with the highest and lowest differences for every target with representations from the 10th layer of BERT. The two rows for each target correspond to BETWEEN-1 and BETWEEN-2, respectively. Target names in ArgQ have been abbreviated for convenience. For example, the target "Marriage" was originally "We should abandon marriage".

Prepositions Matter in Quantifier Scope Disambiguation

Aleksander Leczkowski
Faculty of Psychology
University of Warsaw
ale.leczkowski@gmail.com

Justyna Grudzińska
Faculty of Philosophy
University of Warsaw
j.grudzinska@uw.edu.pl

Manuel Vargas Guzmán
Faculty of Philosophy
University of Warsaw
m.vargas-guzman@uw.edu.pl

Aleksander Wawer
Institute of Computer Science
Polish Academy of Sciences
axw@ipipan.waw.pl

Aleksandra Siemieniuk
Faculty of Polish Studies
University of Warsaw
aleksandra.siemieniuk@uw.edu.pl

Abstract

Although it is widely agreed that world knowledge plays a significant role in quantifier scope disambiguation (QSD), there has been only very limited work on how to integrate this knowledge into a QSD model. This paper contributes to this scarce line of research by incorporating into a machine learning model our knowledge about relations, as conveyed by a manageable closed class of function words: prepositions. For data, we use a scope-disambiguated corpus created by AnderBois, Brasoveanu and Henderson, which is additionally annotated with prepositional senses using Schneider et al's Semantic Network of Adposition and Case Supersenses (SNACS) scheme. By applying Manshadi and Allen's method to the corpus, we were able to inspect the information gain provided by prepositions for the QSD task. Statistical analysis of the performance of the classifiers, trained in scenarios with and without preposition information, supports the claim that prepositional senses have a strong positive impact on the learnability of automatic QSD systems.

1 Introduction

QSD is a problem in natural language processing that arises in connection with sentences that contain multiple quantified NPs:

(1) Every kid climbed a tree.

Sentence (1) can be understood to mean that every kid climbed a possibly different tree. This is the so-called surface scope reading where the first quantified NP has wider scope than the second, corresponding to the surface ordering of the two NPs in the sentence: *every kid* > *a tree*. The other, and usually less preferred, reading is the one in which there is a single tree that all the kids climbed. This is the inverse scope reading where the second quantified NP has wider scope than the first, reversing the order of the two NPs in the sentence: *a tree* > *every kid*. Many studies on quantifier scope have dealt with the issue of generating the set of possible scope readings for a sentence like (1), both from a theoretical perspective (May, 1978; Cooper, 1983; May, 1985; Hendriks, 1993; Steedman, 2012; Barker and Shan, 2014) and computationally (Woods, 1987; Hobbs and Shieber, 1987; Bos, 1996; Copestake et al., 2001; Egg et al., 2001; Bos et al., 2004; Koller et al., 2008; Evang and Bos, 2013; Sayeed, 2016). A much smaller number of studies have focused on statistical and automatic QSD and the problem of identifying the set of factors relevant to scope preferences (Higgins and Sadock, 2003; AnderBois et al., 2012; Manshadi and Allen, 2011; Manshadi et al., 2013). These studies have shown, mostly in line with what was proposed in the semantics literature and borne out in psycholinguist work (Ioup, 1975; Micham et al., 1980; Gillen, 1991; Kurtzman and MacDonald, 1993; Tunstall, 1998; Anderson, 2004; Radó and Bott, 2011; Dotlačil and Brasoveanu, 2015; Capelier-Mourguy et al., 2015), that the grammatical role (i.e., subject and object) and lexical realization of a quantifier have an effect on scope-taking; linear precedence in a sentence has an effect as

Proceedings of the 29th International Conference on Computational Linguistics, pages 3960–3970
October 12–17, 2022.

well.[1]

The above factors are certainly not sufficient to predict quantifier scope. It has been repeatedly stressed in previous work that world knowledge also plays a significant part in real world QSD, and any successful model for the QSD task should make use of it (Saba and Corriveau, 2001; Srinivasan and Yates, 2009; Manshadi and Allen, 2011; AnderBois et al., 2012; Tsiolis, 2020). To the best of our knowledge, however, Srinivasan and Yates (2009) have been unique in using a model explicitly geared towards world knowledge (in particular, numerical typicality) in the QSD task. Drawing on Saba and Corriveau (2001), they decided on the preferred scoping by comparing the size of two classes, e.g., *Person* and *City*, standing in a relation such as the *living-in* relation. For example, the surface scope reading is dispreferred in a sentence such as *A person lives in every city* because it would require a person to live in an atypically large number of cities. The present study seeks to contribute to this scarce line of research through incorporating into a QSD model our knowledge about relations, as conveyed by a manageable closed class of function words: prepositions.

Relations between objects (but also times, events) are often signaled with prepositions (Pustejovsky, 1991; Srikumar and Roth, 2013; Abzianidze and Bos, 2017; Schneider et al., 2018). Prepositions serve, among other things, to convey place and time (*There is a restaurant* ***at*** *every corner*, *John taught* ***on*** *each Monday*), to express configurational relationships like possession or part/whole (*someone* ***with*** *every key*, *a day* ***of*** *every month*), and to indicate semantic roles in predicate–argument structure like agent or instrument (*a study sponsored* ***by*** *a consumer group*, *a store filled* ***with*** *lots of food*). Recent work argues that certain prepositional senses are special in that they encode dependencies that have an effect on scope-taking (Grudzińska and Zawadowski, 2019, 2020). For example, the preposition *of* expressing 'part-whole sense' — as in *a day of every month* — introduces a dependency between each whole (month) and its respective parts (days). By quantifying over this dependency, we obtain the inverse scope reading for the example in question: for every month, there is a different day that belongs to it (*every month* > *a day*). The surface scope reading (*a day* > *every month*) is excluded because of what we know about parts and wholes, namely that we can have many parts (days) belonging to the same whole (month), but a single part (day) cannot belong to more than one whole (month). Conversely, the 'whole-containing-part sense' of the preposition *of* — as in *a group of four homeowners* — encodes a dependency between a group and its respective members, thus only allowing surface scope (*a group* > *four homeowners*). Furthermore, universal quantification in locative and temporal prepositional phrases tends to support inverse scope. For example, the locative preposition *at* — as in *a restaurant at every corner* — implies 'disjointness' (objects do not occupy more than one place at a time), and hence can be interpreted as a dependency between each corner and the respective restaurants located at that corner. Quantifying over this dependency yields the inverse scope reading: for every corner, there is a different restaurant located at it. The surface scope reading is excluded because one restaurant cannot occupy more than one place (every corner) simultaneously.

Our study takes its theoretical inspiration from the above work and contributes to research on automatic QSD by examining the previously unexplored predictors of quantifier scope: prepositions and their senses. For the experiments undertaken in this study, we use a scope-disambiguated corpus created by AnderBois et al. (2012), additionally annotated with prepositional senses using the Semantic Network of Adposition and Case Supersenses (SNACS) scheme proposed in Schneider et al. (2018). Our results indicate that prepositional senses have a strong role in the QSD task and encourage further research and deeper analysis in this area. The structure of the paper is as follows. Section (2) introduces our corpus and discusses its annotation process. Section (3) explains the methodology of our study. In (4), we introduce our experimental setup and discuss the features we have used in our models. Section (5) presents our results and (6) concludes with a summary and some directions for future work.

2 Corpus

The present study uses a scope-disambiguated corpus which was created for the purposes of the 2012

[1]The effect of linear precedence has been debated in previous works, with some authors arguing against it (Ioup, 1975; Micham et al., 1980; Kurtzman and MacDonald, 1993), and it needs to be further explored, especially in freer word order languages with case marking (as is the case, e.g., in Sayeed et al. (2019)).

study by AnderBois, Brasoveanu and Henderson (2012). It consists of 680 sentences with multiple quantified NPs from the reasoning section of the Law School Admission Test — the so-called logic puzzles. Logic puzzles provide a good corpus for QSD for they use quantifiers frequently, providing a fair number of sentences containing scopally interacting quantifiers.

Every sentence of AnderBois et al.'s corpus is labeled with the relative scope of the quantified NPs involved. Scope is coded numerically, with **1** corresponding to widest scope and smaller numbers indicating narrower scope; cases with no relative scope like logical equivalence (e.g., two universals or two existentials) are co-tagged with the same number. The scope predictors incorporated into the annotation in the corpus include sentence order (it is not explicitly tagged, since it can be recovered from the linear order of the tags themselves), grammatical function (**S**ubject, **O**bject, **A**djunct, etc.) and lexical realization of quantifiers. The beginning of the tag is marked by & and the end is marked by ♯:

(2) Hannah visits at least one&**3_O_at.least.one**♯ city in each&**2_in_each**♯ of the three&**1_of_the.three**♯ countries.

Since the sentences in the corpus were chosen for quantified NPs, they would be expected to provide no bias with respect to prepositions. The most common prepositions in English identified by Litkowski and Hargraves (2007) do indeed overlap in eight cases with those in the corpus, although in some cases the frequency distribution is different: *of*, *in*, *on*, *at*, *to*, *for*, *with*, and *from*. While individual prepositions in prepositional phrases are tagged separately in the corpus (as illustrated by example (2)), the prepositional senses are not. It is, however, prepositional senses (rather than prepositions) that induce or block inverse scope. For example, as discussed above, the 'part-whole sense' of the preposition *of* induces inverse scope, while its inverse 'whole-containing-part sense' blocks it. For our study, we additionally annotated the corpus with prepositional senses, using the Semantic Network of Adposition and Case Supersenses (SNACS) scheme proposed in Schneider et al. (2018; 2020).

2.1 Preposition-sense annotation

The SNACS scheme provides a hierarchy of 50 supersenses, divided into three main subhierarchies that loosely correspond to adverbial adjuncts, event arguments, and adnominal complements:

- CIRCUMSTANCE: TIME, LOCUS, MEANS, MANNER, PATH, ...
- PARTICIPANT: AGENT, THEME, RECIPIENT, BENEFICIARY, INSTRUMENT ...
- CONFIGURATION: WHOLE, ORG, QUANTITYITEM, POSSESSION, STUFF ...

Furthermore, the scheme deploys the construal analysis proposed in Hwang et al. (2017), i.e., it introduces a distinction between a SCENE ROLE (marked by SS), which expresses the preposition's meaning in context, and a FUNCTION (marked by SS2), which denotes the preposition's lexical meaning. Both SCENE ROLE and FUNCTION are drawn from the supersense hierarchy and are often identical. The SNACS scheme was applied to prepositions in the STREUSLE corpus, a collection of online consumer reviews taken from the English Web Treebank (Bies et al., 2012). Each preposition token in the STREUSLE corpus is annotated with SS and SS2 (SS⇝SS2):

(3) Dan arrived at 10 am. TIME⇝TIME

(4) The team at Max's is great. ORG⇝LOCUS

In example (3), the preposition *at* is unambiguously temporal — SS and SS2 are congruent. In example (4), there is an overlap between organizational belonging meaning (marked by ORG) and locational meaning (marked by LOCUS) — SS and SS2 differ. The construal analysis helps with cases where multiple supersenses seem to fit and contributes to reducing disagreement among annotators, who are not forced to pick a single label in cases of meaning overlap.

Our scope-disambiguated corpus has been annotated with prepositional senses by three annotators, all non-native speakers with linguistic training. The annotators were familiar with the annotation manual — guidelines for English including description of the 50 supersenses, with examples and criteria for borderline cases (Schneider et al., 2020). Across all targets, there was good agreement on SS between the three annotators, $k = .68, p = .000$,

and there was very good agreement on SS2 between the three annotators, $k = .79, p = .000$. Agreement was higher on the function slot than on the scene role slot. This is expected considering the fact that the function of a preposition reflects its prototypical and more stable meaning, whereas the scene role depends on context and can vary more from person to person. Our results are only slightly lower than the SNACS IAA numbers found in Schneider et al. (2018) ($k = .73$ and $k = .80$). Our agreement is so strong most likely due to the simple literal language of the logic puzzles. The remaining differences were adjudicated in meetings involving all of the three annotators.

2.2 Scope annotation

The number of tagged quantified NPs in a sentence ranges from two to eight in AnderBois et al.'s corpus. Manshadi and Allen (2011, 2013) developed a method that can deal with an arbitrary number of quantifiers per sentence in the QSD task. They define the task as learning to build a partial order over the set of quantifiers in the sentence. In adapting the scope coding in AnderBois et al.'s corpus to Manshadi and Allen's method, we thus consider three relations between each pair of quantifiers $\{q_1, q_2\}$, with q_1 occurring before q_2 in a given sentence: *wide scope* ($q_1 > q_2$), *narrow scope* ($q_2 > q_1$) and *incomparability* (q_1, q_2). The three relations are used in order to determine the quantifier scoping of each sentence from the corpus, based on the relative scopings provided. Determining the scopes of the tagged quantifiers in example (2) is straightforward. The third quantifier outscopes the second ($q_3 > q_2$); the second one outscopes the first one ($q_2 > q_1$). Moreover, since outscoping is a transitive relation, the third one also outscopes the first one ($q_3 > q_1$). Hence, the formula describing the sentence's scoping looks as follows: $q_3 > q_2 > q_1$. Each of the 680 sentences in the corpus is annotated following that method. According to the formula $\sum_i n_i * (n_i - 1)/2$, where n_i denotes the number of quantified NPs in a sentence, there are 1451 relations between quantifiers in the corpus.

3 Method

As mentioned above, Manshadi and Allen build their method on the fact that quantifier scopings (QS) form partial orders. Hence, they define QSD as a task of creating partial orders and show that it is equivalent to a pairwise comparison problem (see Manshadi (2014) for definitions and proofs).

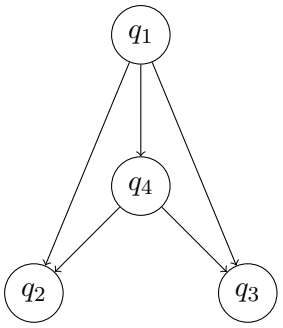

Figure 1: TDAG representing quantifier scopings in example (5): $q_1 > q_4 > q_2, q_3$.

3.1 Manshadi and Allen's approach

Partial orders can be represented as Directed Acyclic Graphs (DAGs). In fact, since outscoping is a transitive relation, Transitive Directed Acyclic Graphs (TDAGs) have a one-to-one correspondence with quantifier scopings — each has exactly one TDAG representing it. Hence, every sentence's QS is analysed in its transitive closure form and TDAGs are used for visualisation purposes.

Figure 1 depicts a TDAG which is a representation of a typical, for the examples in the corpus, quantifier scoping: $q_1 > q_4 > q_2, q_3$.[2] A sentence from the corpus which is defined by this order is provided in example (5).

(5) Each&**1_S_each**♯ member of the Kim family sits in a&**3_in_a_Locus_Locus**♯ seat adjacent to, and in the same row&**3_in_the.same_Locus_Locus**♯ as, at least one other&**2_as_at.least.one.other_ComparisonRef_ComparisonRef**♯ member of the family.

Since the QSD task is reduced to a problem of pairwise comparisons, a sentence containing n quantifiers results in $n * (n - 1)/2$ samples. There are four quantified NPs in example (5) which results in six different observations for the classifier. For each pair of quantifiers (an observation), a classifier has to predict one of three relations: *wide scope*, *narrow scope* or *incomparability*. From the perspective of a TDAG (G), those relations are defined as follows for every pair $\{q_1, q_2\}$, where q_1 precedes q_2 in a given sentence:

[2] This notation is equivalent to $q_1 > q_4 > q_2$ and $q_1 > q_4 > q_3$.

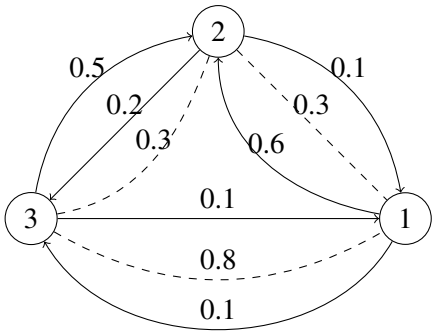

Figure 2: Example of a preference graph (Manshadi, 2014, p. 136).

1. *wide scope* (+) if $(q_1, q_2) \in$ G
2. *narrow scope* ($-$) if $(q_2, q_1) \in$ G
3. *incomparability* (ϵ) otherwise

That is, if two quantifiers q_1 and q_2 are characterised by *wide scope*, there is a directed edge from q_1 to q_2. In the case of *narrow scope*, a directed edge goes from q_2 to q_1. *Incomparability* is represented by the lack of an edge between two nodes.

A ternary soft classifier [3] predicts probabilities for each observation, for each of the three possible classes. Once those probabilities are predicted, a preference graph for each sentence's scoping can be built, as in Figure 2.

The goal at this point is to find a subgraph of that preference graph which satisfies the following condition: that it maximizes the sum of weights with the constraint that the resulting subgraph is a TDAG. The algorithm that performs this task is presented in the following section, and it is based on finding an approximately optimal ordering (Cohen et al. 1999).

3.2 Approximation algorithm

Let $(u, v)^+$, $(u, v)^-$ and $(u, v)^\epsilon$ be the probabilities that the nodes u and v are in a wide scope, narrow scope, and incomparability relation, respectively. The algorithm takes a preference graph Γ_p and stores its vertices in a set V. The difference between outgoing and incoming edges for each vertice is computed (lines 3 and 4) and the highest value is selected (i.e., the node with the widest scope) to store it as t (line 5). From lines 6 to 8, a rank (starting from 0) is assigned to t, the algorithm checks before if there is a node with the current rank that outscopes t, in which case the rank is incremented by one. From lines 9 to 11, edges (v, t) are added to the final graph G by checking all v nodes from previous ranks that have a wide scope relation with t. Finally, t is removed from V and added to G. The process repeats until V is empty.

Algorithm 1 Creates a TDAG

Input: a preference graph Γ_p
Output: a transitive directed acyclic graph G

1: $V \leftarrow$ get_vertices(Γ_p), $r \leftarrow 0$, $G \leftarrow \emptyset$
2: **while** V is non-empty **do**
3: **for each** $u \in V$ **do**
4: $\pi(u) \leftarrow \sum_{v \in V}(u, v)^+ - \sum_{v \in V}(u, v)^-$
5: $t \leftarrow \text{argmax}_{u \in V}\pi(u)$
6: **if** $\exists v \in G : \rho(v) = r$ **and** $(v, t)^+ > (v, t)^\epsilon$ **then**
7: $r \leftarrow r + 1$
8: $\rho(t) \leftarrow r$
9: **for each** $v \in G$ **do**
10: **if** $\rho(v) < r$ **and** $(v, t)^+ > (v, t)^\epsilon$ **then**
11: $G \leftarrow G \cup \{(v, t)\}$
12: $V \leftarrow V - \{t\}$
13: $G \leftarrow G \cup \{t\}$
14: **end while**

4 Experimental setup

A Support Vector Machines (SVM) classifier, Python's scikit-learn implementation (Pedregosa et al., 2011), was fitted to the data (n = 1451) in order to predict probabilities of three different relations between each pair of quantifiers: *wide scope*, *narrow scope* or *incomparability*. Once the probabilities were predicted by the classifier, in order to restore a full sentence's quantifier scoping, a predicted TDAG was built.[4]

4.1 Features

A small set of features was selected for the purpose of the experiment: only those that were manually annotated in the corpus or could be computed in a simple manner. Listed below are the extracted features, each with a brief explanation:

- *Quantifier lexicalization* — quantifier lexicalizations are combined into groups in order to limit the dimensionality of this feature. For instance, all bare numerals are grouped together, all exactly-modified numerals (e.g., *exactly one*) are combined together, superlative

[3] Note that a hard classifier cannot be used here as then there is no guarantee that the resulting predicted graph will be either acyclic or transitive.

[4] Therefore, the breakdown of data was made at the sentence level and not at the observation level. Otherwise, this could result in observations from the same sentence being placed in both the training and test set, which would not allow restoring sentence's quantifier scoping.

and comparative modified numerals (e.g., *at least/most three* and *more/less than three*) are assigned to one group, and so on.

- *Complex* — a binary feature that denotes whether a quantifier lexicalization consists of one token (e.g., *one*) or more than one token (e.g., *more than one*).
- *Grammatical function* — whether a tagged NP plays the role of, for instance, a *subject* or an *object*.
- *Appositive* — a binary feature which denotes whether a tagged NP is followed by an appositive (e.g., *four people - Grace, Heather, Josh, and Maria*).
- *Prepositions* — preposition lexicalizations or preposition supersenses depending on the system (see Section 4.4 for an explanation). We focus on SUPER SS2 and SUPER SS⇝SS2 combinations only, i.e., we drop the less stable and more idiomatic SUPER SS.

→ *Distance* — a gap between a pair of quantifiers in a given observation. For instance, if a sentence has three tagged quantificational expressions q_1, q_2 and q_3, occurring in that order in the sentence, then the distance between q_1 and q_2 equals 1 and the distance between q_1 and q_3 equals 2.

Since an observation is a pair of quantifiers, each feature[5] was defined twice for a given observation.

Linear precedence, a much-discussed predictor of quantifier scope, is not provided here as a separate feature. It is inherently encoded due to the manner in which the task is formulated, as each observation is a pair of quantifiers $\{q_1, q_2\}$, where q_1 occurs before q_2 in a given sentence.

Feature selection was performed using the Mutual Information (MI) measure. First, all of the features with MI equal to zero were deleted. In fact, this led only to the removal of features that were duplicated as a result of defining an observation as a pair. For instance, supersense (SS2) ENDTIME occurs only once in the data and only as a property of the first quantifier in an observation — denoted as ENDTIME_1; hence, feature ENDTIME_2 was deleted. Second, features occurring only once in the corpus were deleted as well; as a result, ENDTIME_1 was also removed.

There are 27 different prepositions, 26 different SS2 supersenses and 67 different SS⇝SS2 combinations in the corpus. Since each observation is a combination of two quantifiers, these numbers correspond to 54, 52 and 134 different columns in the feature vector. After the feature selection, we get 39, 41 and 82, respectively.

4.2 Training and optimization

Training and optimisation were performed using nested cross–validation. Hyperparameter selection was executed in the inner loop using the 5-fold technique. Kernel, among other SVM's hyperparameters, was considered in the optimization process and selected from *linear*, *polynomial*, *rbf* and *sigmoid*. The outer loop was repeated 30 times with different random data splits — Monte Carlo cross-validation. This way a standard 20 percent of the data was used in both inner and outer loops for the purpose of the validation of the models and the final results are an average of 30 independent runs.

4.3 Evaluation

Three different evaluation metrics, adapted to the QSD task by Manshadi and Allen (2011), were used in order to assess the performance of the models and all three of them, similarity, precision and recall, are based on the notion of the similarity of two graphs which represent gold $G_g = (V, E_g)$ and predicted $G_p = (V, E_p)$ sentence's quantifier scopings. Let $G^+ = (V, E^+)$ be the transitive closure of the graph and $\overline{G} = (V, \overline{E})$ be the complement of the undirected version of G, where V denotes the set of nodes and E corresponds to the set of edges. The most general one of the three, the similarity metric (Equation 1), was used for hyperparameter selection during the optimization process as well as for the purpose of statistical testing.

$$\sigma^+ = \frac{|E_p^+ \cap E_g^+| \cup |\overline{E}_p^+ \cap \overline{E}_g^+|}{|V|(|V|-1)/2} \qquad (1)$$

The similarity measure treats outscoping and incomparability relations equally. In practice, it is the outscoping relation that should be most important in classification. That is because if the outscoping relation is mislabeled, it leads to a different interpretation of the sentence. Hence, Manshadi and Allen also adapt to the task a form of precision (Equation 2) and recall (Equation 3) which

[5] Except for the feature *distance* which is a property of the relation between quantifiers, not a property of a quantifier itself.

are based on the number of outscoping relations identified correctly.

$$P^{+} = \frac{|E_p^{+} \cap E_g^{+}|}{|E_p^{+}|} \quad (2)$$

$$R^{+} = \frac{|E_p^{+} \cap E_g^{+}|}{|E_g^{+}|} \quad (3)$$

One might point out that precision or recall should be the metric selected to assess the model's performance during optimisation and to report final results. Note, however, that there are a number of sentences in our corpus where the incomparability relation is the dominant or only relation present. Sentences defined by QS that only consist of incomparability relations are not considered in the computation of precision and recall. Hence, even though informative, those metrics do not result in a comprehensive evaluation.

4.4 Models

Four different training scenarios are conducted in order to assess the impact of prepositions and preposition supersenses on the learnability of the QSD system:

- BASELINE — models trained using all of the features defined, except information about prepositions.
- PREP — models trained using all of the features defined, including preposition lexicalizations, but not preposition supersenses.
- SUPER SS2 — models trained using all of the features defined, including preposition supersenses (SS2 only), but except preposition lexicalizations.
- SUPER SS⇝SS2 — models trained using all of the features defined, including preposition supersenses (SS⇝SS2 combinations), but except preposition lexicalizations.

The performance of these four systems allows us, first, to study the effect of preposition information on the ability of a system to learn a QSD task and, second, to assess whether this impact is better captured when provided with the SNACS supersense hierarchy.

An additional baseline pseudo-model — WIDE — is presented as a reference. It always predicts the most frequent label in the training set.[6]

[6]In total, out of 1451 observations, 307 represent *narrow scope*, 828 *wide scope* and 316 *incomparability*.

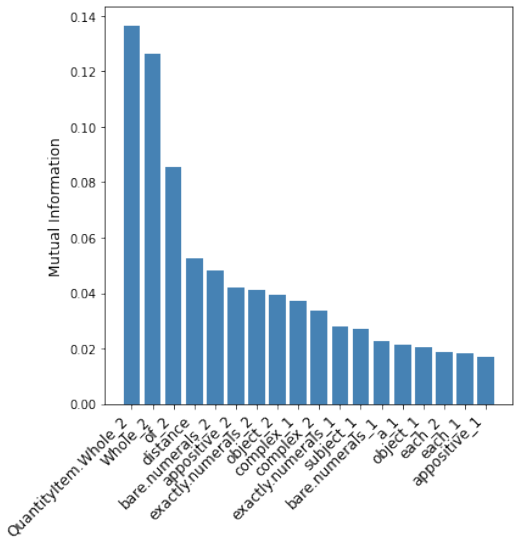

Figure 3: Most significant features according to the Mutual Information analysis.

5 Results [7]

5.1 Feature Importance

Figure 3 presents the 18 most informative features. Both the grammatical roles of *subject* and *object* and certain lexical realizations of quantifiers (*each*, *a*, bare numerals and exactly-modified numerals) rank high in the results, in line with previous findings. One other feature related to lexical realization, *complex*, also ranks high. The feature *appositive* signals the referential function of the NP to which it is related. Its high ranking is in line with the well-known fact that referentially used NPs tend to take the widest scope possible. Notably, the preposition *of* (when present in the second tagged NP in a given observation) ranks third. As expected, the 'part-whole sense' of the preposition *of*, marked by WHOLE and the corresponding role-function combination QUANTITYITEM⇝WHOLE, appears to be even more informative, ranking the highest.

Based on previous findings, one might expect the features of *subject* and *object* to rank higher in the analysis. However, previous studies (e.g., AnderBois et al. (2012)) that reported grammatical function to have a strong impact on scope-taking only focused on sentences with two quantified NPs and did not consider the incomparability relation between quantifiers. Thus, experimental setups previously employed might have inflated the role

[7]The information and code necessary to replicate the results reported in this section are available at the GitHub link: https://github.com/ALeczkowski/prep_matter_in_qsd

of *subject* and *object* in the QSD task.

5.2 Experiments

The results of the experiments are presented in Table 1. The first observation is that standard deviation is substantial in the case of each model and each metric. This is a result of the diverse data set (e.g., sentences with 2 vs. 8 quantified NPs) and shows the importance of reporting final results as a mean over a significant number of runs (in this case, 30). Second, the WIDE pseudo-model that classifies each observation as *wide scope*[8] achieves significant results, e.g., the similarity measure equals 64.82, which is a result of the domination of that relation in the data. Third, in the case of each model, precision and recall are higher than the similarity metric. This contradicts expectations and is also different than in Manshadi and Allen's experiments, where the opposite is the case. It appears that models perform classification better with respect to outscoping relations than when it comes to *incomparability*.[9] Last but not least, by looking at raw numbers, it may be noticed that adding prepositions to the feature vector which is fed to the classifier improves the performance of that classifier on all metrics. Adding preposition supersenses, instead of preposition lexicalizations, results in further improvement, both in the case of SS2 and SS $\rightsquigarrow$ SS2 combinations. However, tests need to be performed in order to determine whether those differences are statistically significant. Since SUPER SS2 is characterised by both better performance and lower standard deviation than SUPER SS$\rightsquigarrow$SS2,[10] only the former is selected for statistical testing.

Since, when it comes to the similarity metric, homogeneity of variance is present in the three compared groups (Bartlett test; p = 0.31) and each group's results are normally distributed (Shapiro-Wilk test; p = 0.82, 0.29 and 0.32 for BASELINE, PREP and SUPER SS2, respectively), one-way ANOVA is performed to determine whether there are any statistically significant differences between compared systems. The test is statistically significant with $F(2, 87) = 9.21$ and $p = 0.000$. Table 2 presents the p-values of the post hoc t-tests performed in order to inspect particular differences.

Model	Similarity		Precision		Recall	
	Mean	SD	Mean	SD	Mean	SD
WIDE	64.82	3.46	73.73	3.29	69.00	3.39
BASE	80.53	3.13	84.96	2.91	86.24	2.78
PREP	81.99	2.45	86.42	2.43	88.05	2.14
SS2	**83.57**	2.45	**88.50**	1.88	**89.88**	1.76
SS$\rightsquigarrow$SS2	83.45	2.69	88.23	2.15	89.48	1.92

Table 1: Mean results and standard deviation of each of the four systems — BASELINE, PREP, SUPER SS2 and SUPER SS$\rightsquigarrow$SS2 — and the WIDE measure.

Model	BASELINE	PREP
PREP	0.056	-
SUPER SS2	**0.000**	0.056

Table 2: P-values for pairwise t-tests with Holm correction for multiple comparisons; similarity metric.

It is not the case that providing information about prepositions to the models significantly improves the performance of those models — the p-value for the comparison between BASELINE and PREP is bigger than 0.05. However, as predicted, providing the model with information about preposition supersenses, only the SS2 part, significantly improves the performance with respect to BASELINE — p-value < 0.05 but not with respect to PREP — p-value > 0.05.[11]

6 Summary and Future Work

This study dealt with the QSD task following the methodology established by Manshadi and Allen (Manshadi and Allen, 2011; Manshadi et al., 2013; Manshadi, 2014) which allows to consider any sentence, with no restriction on the number of quantifiers involved, in a ternary classification task. Applying this method to the scope-disambiguated corpus (AnderBois et al., 2012), additionally tagged

[8] In each of 30 different data splits, *wide scope* was the most frequent relation in the training set.

[9] One possible explanation of that fact might be that no dependency parser was used in the experiment. Hence, no information was extracted about, for instance, conjuncts occurring between quantifiers, which is a strong predictor of the incomparability relation.

[10] One possible explanation for the lower performance of SUPER SS$\rightsquigarrow$SS2 is that the SS$\rightsquigarrow$SS2 combinations are too fine-grained for the size of the data.

[11] We also experimented with a model that includes both preposition lexicalizations and preposition supersenses (just the SS2 part). Performance of this system is as follows (mean, SD): (83.01, 2.29), (87.52, 2.17), (89.12, 1.88) for similarity, precision and recall, respectively. Statistical analysis of this system's results led to exactly the same conclusions as when the model including only supersenses was used. That is, it performed significantly better with respect to the BASELINE but not with respect to the system involving only preposition lexicalizations. Thus, there is no theoretical reason to believe that preposition lexicalizations would encode any relevant information that is not already captured by preposition supersenses.

with the SNACS scheme (Schneider et al., 2020), allowed us to investigate the question of whether information encoded by prepositions, or preposition senses to be exact, proves useful in the QSD task, as inspected with SVM.

Our results confirm the formulated hypothesis — preposition senses, but not preposition lexicalizations, positively impact the learnability of the models and, hence, it may be inferred that they do convey *world knowledge* in a manner beneficial for the algorithm. Note that, out of 1679 tagged quantified NPs in the corpus, only around a third (581 to be exact) are nested in prepositional phrases; this fact further strengthens our conclusions.

The fact that the methodology followed in this paper reduces the QSD problem to a pairwise comparisons task has its benefits. For instance, it significantly expands our sample from 680 sentences into 1451 pairwise comparisons. However, it comes at the price of a simplification, which might lead to information loss since each pair of quantified NPs in a sentence is treated independently of other NPs in that sentence, which in reality is not the case. Another way to approach the QSD problem would be to treat it on a sentence level, making use of modern deep learning techniques such as pre-trained transformer neural networks. However, that might require larger quantifier scope-disambiguated corpora which do not yet exist.[12]

Acknowledgements

First of all, we would like to thank Scott AnderBois, Adrian Brasoveanu and Robert Henderson for sharing their scope-disambiguated corpus with us. We would also like to express our great thanks to the Law School Admission Council for allowing us to use the materials in the corpus for our research purposes. We are also grateful to the three anonymous reviewers for helpful comments and suggestions. The research of the first two authors and the fifth one is funded by the National Science Center, Poland (Grant No. DEC-2019/35/B/HS1/01541).

[12]We tested a solution based on the Universal Sentence Encoder (Cer et al., 2018). For each quantifier, we concatenated vectors: embedding of the quantifier expression and embedding of the entire sentence (2 x 512). The vectors were then classified using the SVM classifier with a radial kernel. This solution achieved an accuracy of 0.45 (the most frequent baseline was at 0.47) when predicting relative quantifier scope. The results are averages in a 10-fold cross-validation.

References

Lasha Abzianidze and Johan Bos. 2017. Towards universal semantic tagging. In *Proceedings of IWCS*.

Scott AnderBois, Adrian Brasoveanu, and Robert Henderson. 2012. The pragmatics of quantifier scope: A corpus study. In *Proceedings of Sinn und Bedeutung*, pages 15–28.

Catherine Anderson. 2004. *The Structure and Real-time Comprehension of Quantifier Scope Ambiguity*. Ph.D. thesis, Northwestern University.

Chris Barker and Chung-chieh Shan. 2014. *Continuations and natural language*, volume 53. Oxford Studies in Theoretical Linguistics.

Ann Bies, Justin Mott, Colin Warner, and Seth Kulick. 2012. *English Web Treebank*, volume 53. Technical Report LDC2012T13, Linguistic Data Consortium, Philadelphia, PA.

Johan Bos. 1996. Predicate logic unplugged. In *Proceedings of the 10th Amsterdam Colloquium*, pages 133–143.

Johan Bos, Stephen Clark, Mark Steedman, James R. Curran, and Julia Hockenmaier. 2004. Wide-coverage semantic representations from a CCG parser. In *Proceedings of COLING 2004*, pages 1240–1246.

Arthur Capelier-Mourguy, Philippe Blache, Christian Retoré, and Laurent Prévot. 2015. Quantifier scope: A formal and experimental study. In *CJC-SC: Colloque des Jeunes Chercheurs en Sciences Cognitives*, pages 67–69.

Daniel Cer, Yinfei Yang, Sheng-yi Kong, Nan Hua, Nicole Limtiaco, Rhomni St. John, Noah Constant, Mario Guajardo-Cespedes, Steve Yuan, Chris Tar, Brian Strope, and Ray Kurzweil. 2018. Universal sentence encoder for English. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, Brussels, Belgium. Association for Computational Linguistics.

W. W. Cohen, R. E. Schapire, and Y. Singer. 1999. Learning to order things. *Journal of Artificial Intelligence Research*, 10:243–270.

Robin Cooper. 1983. *Quantification and semantic theory*. Dordrecht: Reidel.

Ann Copestake, Alex Lascarides, and Dan Flickinger. 2001. An algebra for semantic construction in constraint-based grammars. In *Proceedings of the 39th annual meeting of the Association for Computational Linguistics*, pages 140–147.

Jakub Dotlačil and Adrian Brasoveanu. 2015. The manner and time course of updating quantifier scope representations in discourse. *Language, Cognition and Neuroscience*, 30(3):305–323.

Markus Egg, Alexander Koller, and Joachim Niehren. 2001. The constraint language for lambda structures. *Journal of Logic, Language, and Information*, 10(4):457–485.

Kilian Evang and Johan Bos. 2013. Scope disambiguation as a tagging task. In *Proceedings of the 10th International Conference on Computational Semantics*, pages 314–320.

Kathryn Gillen. 1991. *The Comprehension of Doubly Quantified Sentences*. Ph.D. thesis, University of Durham.

Justyna Grudzińska and Marek Zawadowski. 2019. Inverse linking, possessive weak definites and Haddock descriptions: A unified dependent type account. *Journal of Logic, Language and Information*, 28(2):239–260.

Justyna Grudzińska and Marek Zawadowski. 2020. A scope-taking system with dependent types and continuations. In *Logic and Algorithms in Computational Linguistics 2018 (LACompLing2018)*, pages 155–176. Springer.

Herman Hendriks. 1993. *Studied flexibility: Categories and types in syntax and semantics*. Institute for Logic, Language and Computation.

Derrick Higgins and Jerrold M. Sadock. 2003. A machine learning approach to modeling scope preferences. *Computational Linguistics*, 29(1):73–96.

Jerry R. Hobbs and Stuart M. Shieber. 1987. An algorithm for generating quantifier scopings. *Computational Linguistics*, 13:47–63.

Jena D. Hwang, Archna Bhatia, Na-Rae Han, Tim O'Gorman, Vivek Srikumar, and Nathan Schneider. 2017. Double trouble: The problem of construal in semantic annotation of adpositions. In *Proceedings of *SEM*, pages 178–188.

Georgette Ioup. 1975. Some universals for quantifier scope. In *Syntax and Semantics*, volume 4, pages 37–58. Academic Press, New York.

Alexander Koller, Michaela Regneri, and Stefan Thater. 2008. Regular tree grammars as a formalism for scope underspecification. In *Proceedings of ACL-08: HLT*, pages 218–226.

Howard S. Kurtzman and Maryellen C. MacDonald. 1993. Resolution of quantifier scope ambiguities. *Cognition*, 48(3):243–279.

Ken Litkowski and Orin Hargraves. 2007. Word-sense disambiguation of prepositions. In *Proceedings of SemEval*, pages 24–29.

Mehdi Manshadi and James Allen. 2011. Unrestricted quantifier scope disambiguation. In *Proceedings of Association for Computational Linguistics'11, Workshop on Graph-based Methods for NLP (TextGraph–6)*, pages 51–59.

Mehdi Manshadi, Daniel Gildea, and James Allen. 2013. Plurality, negation, and quantification: Towards comprehensive quantifier scope disambiguation. In *Proceedings of of the 51st Annual Meeting of the Association for Computational Linguistics*, pages 64–72.

Mohammad Hafezi Manshadi. 2014. *Dealing with quantifier scope ambiguity in natural language understanding*. University of Rochester.

Robert May. 1978. *The grammar of quantification.* Ph.D. thesis, Massachusetts Institute of Technology.

Robert May. 1985. *Logical Form: Its structure and derivation*, volume 12. MIT press.

Dennis L. Micham, Jack Catlin, Nancy J. VanDerveer, and Katherine A. Loveland. 1980. Lexical and structural cues in quantifier scope relations. *Journal of Psycholinguistics Research*, 48(9):367–377.

F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. 2011. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830.

James Pustejovsky. 1991. The syntax of event structure. *Cognition*, 41(1–3):47–81.

Janina Radó and Oliver Bott. 2011. Underspecified representations of scope ambiguity? In *Proceedings of the 18th Amsterdam Colloquium*, pages 180–189.

Walid S. Saba and Jean-Pierre Corriveau. 2001. Plausible reasoning and the resolution of quantifier scope ambiguities. *Studia Logica*, 67(2):271–289.

Asad Sayeed. 2016. Representing the effort in resolving ambiguous scope. In *Proceedings of Sinn und Bedeutung*, pages 604–621.

Asad Sayeed, Matthias Lindemann, and Vera Demberg. 2019. Verb-second effect on quantifier scope interpretation. In *Proceedings of the Workshop on Cognitive Modeling and Computational Linguistics*, pages 134–139.

Nathan Schneider, Jena D. Hwang, Vivek Srikumar, Jakob Prange, Austin Blodgett, Sarah R. Moeller, Adi Shalev, Omri Abend, Archna Bhatia, Na-Rae Han, and Tim O'Gorman. 2020. Adposition and case supersenses v2.5: Guidelines for english. *arxiv.org/pdf/1704.02134v6*.

Nathan Schneider, Jena D. Hwang, Vivek Srikumar, Jakob Prange, Austin Blodgett, Sarah R. Moeller, Aviram Stern, Adi Bitan, and Omri Abend. 2018. Comprehensive supersense disambiguation of English prepositions and possessives. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics*, pages 185–196. Association for Computational Linguistics.

Vivek Srikumar and Dan Roth. 2013. Modeling semantic relations expressed by prepositions. *Transactions of the Association for Computational Linguistics*, 1:231–242.

Prakash Srinivasan and Alexander Yates. 2009. Quantifier scope disambiguation using extracted pragmatic knowledge: Preliminary results. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, pages 1465–1474.

Mark Steedman. 2012. *Taking scope: The natural semantics of quantifiers*. Mit Press.

KC Tsiolis. 2020. Quantifier scope disambiguation. Summary of research conducted in summer 2020 under the supervision of Jackie Cheung.

Susanne Lynn Tunstall. 1998. *The interpretation of quantifiers: semantics and processing*. Ph.D. thesis, University of Massachusetts Amherst.

William A. Woods. 1987. Semantics and quantification in natural language question answering. *Advances in Computers*, 17:1–87.

Modelling Commonsense Properties using Pre-Trained Bi-Encoders

Amit Gajbhiye[*], **Luis Espinosa-Anke**[*◇], **Steven Schockaert**[*]
[*]CardiffNLP, Cardiff University, United Kingdom
[◇]AMPLYFI, United Kingdom
{gajbhiyea, espinosa-ankel, schockaerts1}@cardiff.ac.uk

Abstract

Grasping the commonsense properties of everyday concepts is an important prerequisite to language understanding. While contextualised language models are reportedly capable of predicting such commonsense properties with human-level accuracy, we argue that such results have been inflated because of the high similarity between training and test concepts. This means that models which capture concept similarity can perform well, even if they do not capture any knowledge of the commonsense properties themselves. In settings where there is no overlap between the properties that are considered during training and testing, we find that the empirical performance of standard language models drops dramatically. To address this, we study the possibility of fine-tuning language models to explicitly model concepts and their properties. In particular, we train separate concept and property encoders on two types of readily available data: extracted hyponym-hypernym pairs and generic sentences. Our experimental[1] results show that the resulting encoders allow us to predict commonsense properties with much higher accuracy than is possible by directly fine-tuning language models. We also present experimental results for the related task of unsupervised hypernym discovery.

1 Introduction

Pre-trained language models (Devlin et al., 2019) have been found to capture a surprisingly rich amount of knowledge about the world (Petroni et al., 2019). Focusing on commonsense knowledge, Forbes et al. (2019) used BERT to predict whether a given concept (e.g. *teddy bear*) satisfies a given commonsense property (e.g. *is dangerous*). To this end, they convert the input into a simple sentence (e.g. "*A teddy bear is dangerous*") and treat the task as a standard sentence classification task. Remarkably, they found this approach to surpass human performance. Shwartz and Choi (2020) moreover found that language models can, to some extent, capture commonsense properties that are rarely expressed in text, thus mitigating the issue of reporting bias that has traditionally plagued initiatives for learning commonsense knowledge from text (Gordon and Durme, 2013).

Despite these encouraging signs, however, modelling commonsense properties remains highly challenging. A key concern is that language models are typically fine-tuned on a training set that contains the same properties as those in the test set. For instance, the test data from Forbes et al. (2019) includes the question whether *peach* has the property *eaten in summer*, while the training data asserts that *apple*, *banana*, *orange* and *pear* all have this property. To do well on this task, the model does not actually need to capture the knowledge that peaches are eaten in summer; it is sufficient to capture that *peach* is similar to *apple*, *banana*, *orange* and *pear*. For this reason, we propose new training-test splits, which ensure that the properties occurring in the test data do not occur in the training data. Our experiments show that the ability of language models to predict commonsense properties drops dramatically in this setting.

Our aim is to develop a strategy for modelling the commonsense properties of concepts. Given the limitations that arise when language models are used directly, a natural approach is to pre-train a language model on some kind of auxiliary data. Unfortunately, resources encoding the commonsense properties of concepts tend to be prohibitively noisy. To illustrate this point, Table 1 lists the properties of some everyday concepts according to three well-known resources: ConceptNet (Speer et al., 2017), which is a large commonsense knowledge graph, COMET-2020[2] (Hwang et al., 2021), which

[1]Code and datasets are available at https://github.com/amitgajbhiye/biencoder_concept_property

[2]We used the demo at https://mosaickg.apps.allenai.org/model_comet2020_entities.

Proceedings of the 29th International Conference on Computational Linguistics, pages 3971–3983
October 12–17, 2022.

	ConceptNet	COMET-2020	Ascent++
banana	yellow, good to eat	one of the main ingredients, eaten as a snack, one of many fruits, found in garden, black	rich, ripe, yellow, green, brown, sweet, great, black, useful, safe, delicious, healthy, nutricious, ...
lion	a feline	found in jungle, one of many animals, one of many species, two legs, very large	free, extinct, hungry, close, unique, active, nocturnal, old, dangerous, great, happy, right, ...
airplane	good for quickly travelling long distances	flying, air travel, flying machine, very small, flight	heavy, new, important, white, safe, unique, full, larger, clean, slow, low, unstable, electric, ...

Table 1: Properties of some example concepts, according to three commonsense knowledge resources.

predicts triples using a generative language model that was trained on several commonsense knowledge graphs, and Ascent++ (Nguyen et al., 2021), which is a commonsense knowledge base that was extracted from web text. Given the noisy nature of such resources, we rely on a database with hypernyms instead. The underlying intuition is that hypernyms can be extracted from text relatively easily, while fine-grained hypernyms often implicitly describe commonsense properties. For instance, Microsoft Concept Graph (Ji et al., 2019) lists *potassium rich food* as a hypernym of *banana* and *large and dangerous carnivore* as a hypernym of *lion*. We also experiment with GenericsKB (Bhakthavatsalam et al., 2020), a large collection of generic sentences (e.g. "*Coffee contains minerals and antioxidants which help prevent diabetes*"), to obtain concept-property pairs for pre-training. Given such pre-training data, we then train a concept encoder Φ_{con} and a property encoder Φ_{prop} such that $\sigma(\Phi_{\mathsf{con}}(c) \cdot \Phi_{\mathsf{prop}}(p))$ indicates the probability that concept c has property p.

In summary, our main contributions are as follows: (i) we propose a new evaluation setting which is more realistic than the standard benchmarks for predicting commonsense properties; (ii) we analyse the potential of hypernymy datasets and generic sentences to act as pre-training data; and (iii) we develop a simple but effective bi-encoder architecture for modelling commonsense properties.

2 Related Work

Several authors have analysed the extent to which language models such as BERT capture commonsense knowledge. As already mentioned, Forbes et al. (2019) evaluated the ability of BERT to predict commonsense properties from the McRae dataset (McRae et al., 2005), which we also use in our experiments. The same dataset was used by Weir et al. (2020) to analyse whether BERT-based language models could generate concept names from their associated properties; e.g. given the input "*A* ⟨*mask*⟩ *has fur, is big, and has claws*", the model is expected to predict that ⟨*mask*⟩ corresponds to the word *bear*. Conversely, Apidianaki and Garí Soler (2021) considered the problem of generating adjectival properties from prompts such as "mittens are generally ⟨*mask*⟩". Note that the latter two works evaluated pre-trained models directly, without fine-tuning, whereas the experiments Forbes et al. (2019) involved fine-tuning the language model on a task-specific training set first. When the main motivation is to probe the abilities of language models, avoiding fine-tuning has the advantage that any observed abilities reflect what is captured by the pre-trained language model itself, rather than learned during the fine-tuning phase. However, Li et al. (2021) argue that the extent to which pre-trained language models capture commonsense knowledge is limited, suggesting that some form of fine-tuning is essential in practice. Interestingly, this remains the case for large language models. For instance, their model had 7 billion parameters, while West et al. (2021) report that the predictions from GPT-3 (Brown et al., 2020) had to be filtered by a so-called critic model when distilling a commonsense knowledge graph.

The strategy taken by COMET (Bosselut et al., 2019) is to fine-tune a GPT model (Radford et al.) on triples from commonsense knowledge graphs. Being based on an autoregressive language model, COMET can be used to predict concepts that take the form of short phrases, which is often needed when reasoning about events (e.g. to express motivations or effects). However, as illustrated in Table 1, COMET is less suitable for modelling the commonsense properties of concepts. Other approaches have focused on improving the commonsense rea-

soning abilities of general purpose language models. For instance, Zhou et al. (2021) introduce a self-supervised pre-training tasks to encourage language models to better capture the commonsense relations between everyday concepts.

A final line of related work concerns the modelling of hypernymy. Several authors have proposed specialised embedding models for this task (Dasgupta et al., 2021; Le et al., 2019). Most relevant to our work, Takeoka et al. (2021) fine-tune a BERT-based language model to predict the validity of a concept–hypernym pair. Inspired by the effectiveness of Hearst patterns (Hearst, 1992), they use prompts of the form "[HYPERNYM] such as [CONCEPT]" (and similar for other Hearst patterns). The extent to which the pre-trained BERT model captures hypnernymy has also been studied. For instance, Hanna and Mareček (2021) use prompts where the prediction of the $\langle mask \rangle$ token can be interpreted as the prediction of a hypernym, to avoid the need for fine-tuning the model.

3 Methodology

Let a set of concept–property pairs $\mathcal{K}$ be given, where $(c, p) \in \mathcal{K}$ means that concept c is asserted to have the property p. We write $\mathcal{C}$ and $\mathcal{P}$ for the sets of concepts and properties in $\mathcal{K}$, i.e. $\mathcal{C} = \{c \mid (c, p) \in \mathcal{K}\}$ and $\mathcal{P} = \{p \mid (c, p) \in \mathcal{K}\}$. We use the term "property" in a broad sense, covering both semantic attributes, as in the pair $(banana, sweet)$, and hypernyms, as in the pair $(banana, fruit)$. This is motivated by the fact that hypernyms often encode knowledge about semantic attributes, as in the pair $(banana, sweet\ fruit)$. In particular, our hypothesis is that, by treating hypernyms and semantic attributes in a unified way, we can pre-train a model on readily available hypernym datasets and use it to predict semantic attributes.

We want to train a model that can predict for a given pair (c, p) whether c has property p. Two general strategies can be pursued when developing such models. The first strategy is to use a so-called cross-encoder, which amounts to fine-tuning a single language model to predict whether a given input (c, p) represents a valid pair or not. The second strategy is to use a so-called bi-encoder, which amounts to the idea that c and p are separately encoded, with the resulting vectors then being used to predict whether (c, p) is a valid pair. In this paper, we pursue the latter strategy. This is primarily motivated by the fact that the concept and property encoders enable a wider range of applications. A cross-encoder can only be used to predict whether a given pair (c, p) is valid or not. In contrast, a bi-encoder model can also be used to efficiently find the properties p of a given concept c. Moreover, the resulting concept and property embeddings may themselves be useful as static representations of word meaning, e.g. as label embeddings for zero-shot or few-shot learning (Socher et al., 2013; Ma et al.; Xing et al., 2019; Li et al., 2020; Yan et al., 2021). Finally, bi-encoders can be trained more efficiently than cross-encoders.

Datasets To train our model, we need a large set of concept–property pairs $\mathcal{K}$. Unfortunately, high-quality knowledge of this kind is not readily available. Part of the underlying issue is that properties of concepts are rarely explicitly stated in text, which is why directly using information extraction techniques is not straightforward. However, initiatives for extracting hypernyms from text have been much more successful, starting with the seminal work of Hearst (1992). A key observation is that fine-grained hypernyms often express commonsense properties, typically as a mechanism for refining hypernyms that would otherwise be too broad. For instance, Microsoft Concept Graph (Ji et al., 2019) lists *vitamin C rich food* as a hypernym of *strawberry*, as a refinement of the more general hypernym *food*. By pre-training our model on concept–hypernym pairs, we may thus expect it to learn about commonsense properties as well. To directly test this hypothesis, we use a set of such concept–hypernym pairs as our pre-training set $\mathcal{K}$. Specifically, we collect the 100K concept–hypernym pairs from Microsoft Concept Graph[3] with the highest confidence score[4] We will refer to this dataset as MSCG.

As a second strategy, we attempt to convert the MSCG dataset into a set of concept–property pairs. To this end, we look for pairs (c, h_1) and (c, h_2) where h_2 is a suffix of h_1. Specifically, if $h_1 = mh_2$ and m is an adjectival phrase, then we assume that m describes a property of c. For instance, MSCG contains the pairs (*strawberry*, *vitamin C rich food*) and (*strawberry*, *food*). Based on this, we would include the pair (*strawberry*, *vitamin C rich*) in $\mathcal{K}$. Clearly this is a heuristic strategy, which

[3] https://concept.research.microsoft.com/Home/Download

[4] Specifically, we used those pairs maximising the *Relations* frequency.

may produce non-sensical or misleading pairs. For instance, according to MSCG, strawberry is a *low-sugar berry*, but this does not entail that strawberry has the property *low-sugar* in general. However, we may expect most of the pairs that are generated with this strategy to be meaningful. A total of 8186 concept–property pairs were obtained in this way. We refer to the resulting dataset as PREFIX.

Finally, going beyond concept-hypernym pairs, we derive a dataset from GenericsKB (Bhakthavatsalam et al., 2020), which contains generic sentences such as "*Bananas are an important food staple in the tropics*". Due to the regular structure of such sentences, we can easily convert them into concept–property pairs; e.g. the aforementioned sentence would become (*banana*, *important food staple in the tropics*). We collect a set of 100K such pairs, by processing the sentences with the highest confidence (i.e. the ones which are most likely to be generic sentences) whose length is at most 7. The reason why we focus on shorter sentences is because they are more likely to capture salient information. We refer to this dataset as GKB.

Training Objective Given the pairs in $\mathcal{K}$, we pretrain two encoders, Φ_{con} and Φ_{prop}, using binary cross-entropy. In particular, the loss function for a given mini-batch is defined as follows:

$$\mathcal{L} = -\sum_{(c,p)\in\mathcal{K}_{\mathsf{batch}}} \log\sigma\big(\Phi_{\mathsf{con}}(c)\cdot\Phi_{\mathsf{prop}}(p)\big) - \sum_{(c,p)\in\mathcal{N}_{\mathsf{batch}}} \log\big(1-\sigma\big(\Phi_{\mathsf{con}}(c)\cdot\Phi_{\mathsf{prop}}(p)\big)\big)$$

Here $\mathcal{K}_{\mathsf{batch}}$ represents the subset of $\mathcal{K}$ that is in the current mini-batch. For efficiency reasons, we sample these mini-batches as follows. First, we sample a subset $\mathcal{C}_{\mathsf{batch}}$ of K concepts from $\mathcal{C}$. Then, for each concept c in $\mathcal{C}_{\mathsf{batch}}$ we sample one property $p \in \mathcal{P}$ such that $(c,p) \in \mathcal{K}$. Let $\mathcal{P}_{\mathsf{batch}}$ be the set of properties that are thus obtained. The sets of positive examples $\mathcal{K}_{\mathsf{batch}}$ and negative examples $\mathcal{N}_{\mathsf{batch}}$ are then defined as follows:

$$\mathcal{K}_{\mathsf{batch}} = (\mathcal{C}_{\mathsf{batch}} \times \mathcal{P}_{\mathsf{batch}}) \cap \mathcal{K}$$
$$\mathcal{N}_{\mathsf{batch}} = (\mathcal{C}_{\mathsf{batch}} \times \mathcal{P}_{\mathsf{batch}}) \setminus \mathcal{K}$$

In other words, the positive examples are the pairs from $\mathcal{K}$ that involve a concept from $\mathcal{C}_{\mathsf{batch}}$ and a property from $\mathcal{P}_{\mathsf{batch}}$. The negative examples are all the other pairs that we can form by taking a concept from $\mathcal{C}_{\mathsf{batch}}$ and a property from $\mathcal{P}_{\mathsf{batch}}$. This in-batch negative sampling strategy ensures that after encoding $|\mathcal{C}_{\mathsf{batch}}|$ concepts and $|\mathcal{P}_{\mathsf{batch}}|$ properties, we can take $|\mathcal{C}_{\mathsf{batch}}| \times |\mathcal{P}_{\mathsf{batch}}|$ training examples into account. Given that the encoders Φ_{con} and Φ_{prop} correspond to fine-tuned language models, and the encoding steps are thus time-consuming, in-batch negative sampling enables a significant speed-up compared to naive strategies in which positive and negative examples are sampled independently. Similar strategies are commonly used in neural information retrieval (Gillick et al., 2019).

Concept and Property Encoders The encoders Φ_{con} and Φ_{prop} correspond to fine-tuned encoder-only language models, such as BERT (Devlin et al., 2019). An important design decision is how the input to these language models is presented. For the concept encoder, the input corresponds to a string of the form "[prefix] c [suffix]", which is usually referred to as the prompt. How this prompt is chosen often has a substantial impact on the performance of a model. For instance, language models have been reported to under-perform if the input is too short (Bouraoui et al., 2020; Jiang et al., 2020). Given the importance of the choice of prompt, several strategies for automatically learning a suitable prompt have been proposed (Shin et al., 2020; Liu et al., 2021). In practice, however, carefully chosen manually designed prompts often outperform such automatically learned prompts (Ushio et al., 2021; Logan et al., 2021). For this reason, we have manually generated a number of templates and evaluated their performance on a held-out portion of the MSCG dataset. Based on this analysis[5], we use the following prompt:

$$\langle cls\rangle \text{ [CONCEPT] means } \langle mask\rangle\langle sep\rangle$$

where $\langle cls\rangle$, $\langle mask\rangle$ and $\langle sep\rangle$ are special tokens from the BERT vocabulary, while [CONCEPT] represents the concept to be modelled. The embedding of the concept is taken to be the contextualised vector of the $\langle mask\rangle$ token, i.e. the representation of this token in the final layer of the language model. We use the same prompt for concepts and properties. However, note that concepts and properties are encoded using different encoders. Intuitively, we think of $\Phi_{\mathsf{con}}(c)$ as a representation of a prototypical instance of concept c, while we view $\Phi_{\mathsf{prop}}(p)$ as a representation of the property p itself. This is why, even when $p = c$, we would expect $\Phi_{\mathsf{con}}(c)$ and $\Phi_{\mathsf{prop}}(c)$ to be different. Under this

[5]Details can be found in Appendix A.

view, $\sigma(\Phi_{\mathsf{con}}(c) \cdot \Phi_{\mathsf{prop}}(p))$ captures the probability that a prototypical instance of c would have the property p. In other words, by using different encoders for concepts and properties, we can capture the default nature of the pairs in $\mathcal{K}$ in a natural way.

4 Experiments

In our experiments, we primarily focus on commonsense property classification, i.e. predicting whether some concept has a given property. We also demonstrate the usefulness of the concept and property encoders on the task of hypernym discovery. Finally, we also present a qualitative analysis.

Training Details We pre-train the concept and property encoders on the datasets introduced in Section 3. We also consider variants in which these datasets are combined; e.g. we write MSCG+PREFIX for the dataset combining the pairs from MSCG and PREFIX. To pre-train our model, we construct separate validation data in the same way. In particular, for MSCG, we select the validation split by taking the next 10K most confident pairs from Microsoft Concept Graph (i.e. after removing the pairs from the MSCG dataset itself), and similar for the other datasets. We train the model for 100 epochs, using early stopping with a patience of 20. We use the AdamW optimizer (Loshchilov and Hutter, 2019) with a learning rate of $2e{-}6$ and set the batch size to 8. We use BERT-base-uncased as our default language model (Devlin et al., 2019), although we have also experimented with BERT-large-uncased, RoBERTa-base and RoBERTa-large (Liu et al., 2019).

4.1 Commonsense Property Classification

Datasets For commonsense property classification, we use the extended version of the McRae dataset (McRae et al., 2005) that was introduced by Forbes et al. (2019). This dataset involves a set $\mathcal{C}$ of 514 concepts and a set $\mathcal{P}$ of 50 properties. For each concept c and property p, the dataset specifies whether c has property p. The set $\mathcal{C}$ is split into a training set $\mathcal{C}_{\mathsf{train}}$ and a test set $\mathcal{C}_{\mathsf{test}}$[6]. During training, the models have access to the ground truth of every pair in $\mathcal{C}_{\mathsf{train}} \times \mathcal{P}$. The models are then tested on all pairs in $\mathcal{C}_{\mathsf{test}} \times \mathcal{P}$. We report the results in terms of the F1 score of the positive label.

As argued in the introduction, by training and testing on the same set of properties, we may not be able to faithfully test a model's ability to predict commonsense properties. For this reason, we consider an alternative setting where the set of properties is instead split into a training set $\mathcal{P}_{\mathsf{train}}$ and a test set $\mathcal{P}_{\mathsf{test}}$. During training, the model then gets access to the ground truth for the pairs in $\mathcal{C} \times \mathcal{P}_{\mathsf{train}}$, while the model is evaluated on the pairs in $\mathcal{C} \times \mathcal{P}_{\mathsf{test}}$. We use 5-fold cross-validation for this setting. Our hypothesis is that this setting will be more difficult, as it would be harder to find properties in the training data that are similar to those from the test set. However, there are nonetheless some similarities between these properties. We therefore also consider a setting in which both the concepts and properties are split into train and test fragments. The model is then trained on the pairs in $\mathcal{C}_{\mathsf{train}} \times \mathcal{P}_{\mathsf{train}}$ and evaluated on the pairs in $\mathcal{C}_{\mathsf{test}} \times \mathcal{P}_{\mathsf{test}}$. We again use a form of cross-validation. In particular, we split $\mathcal{C}$ into three folds: $\mathcal{C}_1$, $\mathcal{C}_2$ and $\mathcal{C}_3$. We similarly split $\mathcal{P}$ into three folds: $\mathcal{P}_1$, $\mathcal{P}_2$ and $\mathcal{P}_3$. In the first iteration, we train on the pairs in $(\mathcal{C}_1 \cup \mathcal{C}_2) \times (\mathcal{P}_1 \cup \mathcal{P}_2)$ and test on the pairs in $\mathcal{C}_3 \times \mathcal{P}_3$. This process is repeated nine times (as we have three ways to choose the concept test split and three ways to choose the property test split).

We have also used the CSLB Concept Property Norms[7], as a second benchmark for commonsense property classification. This dataset covers 638 concepts and 3350 properties. Similar as for McRae, the dataset indicates which concepts have which properties, but there are no standard splits. Moreover, the dataset does not explicitly contain negative examples. For this reason, for each positive example (c, p), we introduce 20 negative examples by replacing p with another property p' (such that (c, p') is not a positive example). This strategy is imperfect, as there will inevitably be some false negatives, but it should still allow us to compare the relative performance of different models. Mirroring the settings from the McRae dataset, we consider a concept-based training-test split (*Con*), a property-based split (*Prop*), and a setting where both concepts and properties are split into training and test sets (*Con+Prop*). For consistency, we use the same cross-validation strategies as for the McRae dataset. In particular, for *Con* we use a fixed split (with 90% of the concepts used for training and 10% for testing). For *Prop*, we use 5-fold cross-validation, while for *Con+Prop* we used the

[6] The split is available at https://github.com/mbforbes/physical-commonsense.

[7] https://cslb.psychol.cam.ac.uk/propnorms

Language Model	Pre-training dataset	McRae			CSLB		
		Con	Prop	C+P	Con	Prop	C+P
Random baseline		26.0	26.5	26.0	8.6	8.4	8.6
Always true		30.3	30.0	30.0	9.1	9.1	9.1
BERT-large sentence classifier (Forbes et al., 2019)		74	-	-	-	-	-
Human performance (Forbes et al., 2019)		67	-	-	-	-	-
BERT-base	No pre-training	77.7	30.7	25.2	51.8	34.1	22.4
BERT-base	MSCG	79.9	46.6	41.6	54.0	36.8	28.9
BERT-base	PREFIX	78.3	44.8	41.0	52.2	33.2	24.3
BERT-base	GKB	79.3	**50.7**	**46.0**	52.1	37.2	30.2
BERT-base	MSCG+PREFIX	80.2	47.8	43.2	53.6	37.3	29.7
BERT-base	MSCG+GKB	80.4	50.3	43.6	54.8	37.1	28.9
BERT-base	MSCG+PREFIX+GKB	79.8	49.6	44.5	54.5	39.1	32.6
BERT-large	No pre-training	75.3	36.6	25.5	54.3	36.4	17.7
RoBERTa-base	No pre-training	41.0	9.4	0.0	38.1	28.7	9.6
RoBERTa-large	No pre-training	73.7	26.9	9.4	55.3	37.8	24.8
BERT-large	MSCG+PREFIX+GKB	**80.5**	49.3	45.5	57.7	41.8	36.4
RoBERTa-base	MSCG+PREFIX+GKB	75.6	42.4	38.1	49.9	36.4	24.3
RoBERTa-large	MSCG+PREFIX+GKB	80.1	46.5	42.5	**59.0**	**42.5**	**36.0**

Table 2: Results in terms of F1 score (percentage) for commonsense property prediction.

3×3 fold cross-validation strategy.

Results The results for commonsense property classification are summarised in Table 2. We include the following baselines. First, the *Random* baseline predicts the positive label with 50% chance. Similarly, *Always true* predicts the positive label in all cases. Next, for the concept-split of the McRae dataset, we compare with the method from Forbes et al. (2019), where each pair (c, p) was converted into a natural language sentence. For instance, (*apple*, *is electrical*) is converted to the sentence "*An apple requires electricity*", which is then fed to a BERT classifier. Due to its manual nature, this method cannot be applied to new properties. We also include the estimate of human performance that was reported by Forbes et al. (2019). Finally, we consider a variant of our model which is directly trained on the McRae and CSLB training data, without the pre-training step.

The next set of results compare the performance of the different pre-training datasets. For these results, all models were initialised with BERT-base. We can clearly see that the pre-trained bi-encoder models outperform the variant without pre-training in nearly all settings (with the results for PREFIX on the CSLB property-split the only exception). This confirms our hypothesis that Microsoft Concept Graph and GenericsKB capture useful information about commonsense properties. Comparing the different corpora, PREFIX achieves the weakest results, which can be explained by the much smaller size of this dataset. However, combining PREFIX+MSCG outperforms MSCG in all but one case. Furthermore, as expected, the property-split (*Prop*) is considerably harder than the standard concept-split (*Con*), with the *C+P* setting being even harder. In fact, for the latter setting, the BERT-base model without pre-training cannot outperform the random classifier for McRae. Note that for CSLB, outperforming the random classifier is easier, given that more training data is available for that dataset. Crucially, while the best baselines only slightly underperform the pre-trained models for the concept-split, much larger differences are seen for the other splits. Overall, these findings confirm our hypothesis that predicting commonsense properties remains a highly challenging problem.

Finally, the table also shows results for some other language models. While the large models generally outperform their base counterparts, the differences are relatively small, and the improvements are not consistent. This finding is in accordance with the conclusion from Li et al. (2021) that even large language models are limited in the amount of commonsense knowledge they capture, and in particular that finding the right pre-training task is crucial. The RoBERTa results without pre-training are particularly disappointing, with learning failing completely in some cases. Even with the pre-training datasets, BERT-base outperforms RoBERTa base, and BERT-large outperforms RoBERTa-large.

	Con	Prop	C+P
Skip-gram ($k = 1$)	70.8	25.0	17.5
Skip-gram ($k = 3$)	53.4	9.5	5.7
GloVe ($k = 1$)	68.8	20.3	21.7
GloVe ($k = 3$)	51.4	6.8	4.9
BERT-base ($k = 1$)	**72.0**	**28.2**	**27.0**
BERT-base ($k = 3$)	55.6	14.6	19.1

Table 3: Evaluation of a nearest neighbour strategy for the McRae dataset (F1 score percentage).

Analysis As we have argued, models can perform well on the *Con* setting by simply transferring knowledge about similar concepts from the training data. This is analysed in more detail in Table 3, which shows the performance of a nearest neighbour classifier. To classify a test pair (c, p) we find the k concepts $c_1, ..., c_k$ from the training split that are most similar to c in terms of cosine similarity. Then we predict the positive label for (c, p) if the majority of $(c_1, p), ..., (c_k, p)$ are assigned the positive label. We test this approach for $k = 1$ and $k = 3$, using embeddings from GloVe (Pennington et al., 2014) and Skip-gram[8] (Mikolov et al., 2013), and using the embeddings predicted by our BERT-base encoder pre-trained on MSCG+PREFIX+GKB. For the *Prop* setting, we similarly predict the label of (c, p) based on the labels of the training pairs $(c, p_1), ..., (c, p_k)$, with $p_1, ..., p_k$ the k properties from the training data that are most similar to p. Finally, for *C+P*, we predict the majority label among the training pairs (c_i, p_j) with $i, j \in \{1, ..., k\}$, where $c_1, ..., c_k$ are the training concepts most similar to c and $p_1, ..., p_k$ are the training concepts most similar to p. The results in Table 3 clearly support our hypothesis about the concept-split. In particular, the nearest neighbour classifier is highly effective for the concept-split (for $k = 1$), outperforming the estimate of human performance from Forbes et al. (2019) for all embedding types, and approaching the performance of the language models without our pre-training task. In contrast, for the *Prop* and *C+P* settings, the nearest neighbour classifier performs, at best, similarly to the random classifier.

4.2 Hypernym Discovery

Given an input word (e.g. *cat*), the aim of the *hypernym discovery* task is to retrieve a set of valid hypernyms (e.g. *animal*, *mammal*, *feline*, etc.). We use this task to analyse the quality of the pre-trained concept and property encoders when used without any fine-tuning on task-specific training data. We use the data from the SemEval 2018 Hypernym Discovery task (Camacho-Collados et al., 2018), focusing on the concept-only split (i.e. without considering named entities). There are three variants of this task: an open-domain setting (referred to as *general*) and two domain-specific settings, focusing on the *music* and *medical* domains. Each variant is associated with a large vocabulary of candidate terms, consisting of 218,753 terms for *general*, 69,118 terms for *music* and 93,888 terms for *medicine*. To solve this task, each word from the vocabulary is encoded using Φ_{prop}. We then use maximum inner product search to efficiently find those words w from the vocabulary that maximise $\Phi_{\text{con}}(t) \cdot \Phi_{\text{prop}}(w)$ for a given target word t. From the retrieved list of words, we remove those that contain the term t itself and those that end with an adjective. For this experiment, we use BERT-large encoders pre-trained on MSCG+PREFIX+GKB. We compare our method with the following baselines for this task: APSyn (Santus et al., 2016), balAPInc (Kotlerman et al., 2010), SLQS (Santus et al., 2014), ADAPT (Maldonado and Klubička, 2018) and Apollo (Onofrei et al., 2018). We report the published results from the SemEval task Camacho-Collados et al. (2018) (where ADAPT only participated in the *general* setting and Apollo only participated in the *music* and *medical* settings). The latter systems achieved the best per-

		MAP	MRR	P@5
General	APSyn	1.7	3.7	1.7
	balAPInc	1.7	3.9	1.7
	SLQS	0.7	1.7	0.7
	Apollo	2.7	6.1	2.8
	Ours	**3.8**	**7.0**	**3.1**
Music	APSyn	1.1	2.6	1.3
	balAPInc	1.4	3.6	1.6
	SLQS	0.6	1.3	0.7
	ADAPT	1.9	**5.3**	1.9
	Ours	**2.3**	5.1	**2.6**
Medical	APSyn	0.7	1.4	0.7
	balAPInc	0.9	2.1	1.1
	SLQS	0.3	0.7	0.3
	ADAPT	**8.1**	**20.6**	**8.3**
	Ours	4.0	9.0	3.9

Table 4: Result of the hypernym discovery experiment.

[8] We used the 300 dimensional Skip-gram vectors trained on Google News and GloVe vectors trained on Common Crawl, available from https://radimrehurek.com/gensim/models/word2vec.html.

formance among the unsupervised methods[9]. Following Camacho-Collados et al. (2018), we report Mean Average Precision (MAP), Mean Reciprocal Rank (MRR), and Precision at 5 ($P@5$), in percentage terms. Table 4 shows that our method outperforms all baselines for *General*, performs similar to ADAPT for *Music* and worse than ADAPT for *Medical*. This is remarkable, given that our method was not designed or tuned for this task. The underperformance on *Medical* can be explained by the lack of training examples from this domain in the pre-training data. As can be observed, the results for all models are low. An error analysis, presented below, revealed that this is largely due to the fact that many correct hypernyms are not included in the ground truth.

Error Analysis Table 5 shows some of the predictions of our model for the *General* setting of the hypernym discovery task. The first set of results shows examples where many of the predicted hypernyms are intuitively correct. However, only few of these hypernyms are covered by the ground truth; ground truth predictions are shown in bold. This illustrates the rather noisy nature of the dataset, and serves as an explanation for the low overall F1 score of the different unsupervised models. The second set of results in Table 5 covers cases where most of the predictions are incorrect. In some cases, e.g. for *children*, the model predicts semantic properties rather than hypernyms, which shows that simply filtering predictions that end with an adjective is not always sufficient. The case of *broiler chicken* shows that the model sometimes predicts terms that are semantically related, but which are clearly not hypernyms (nor semantic attributes). As a variant of this observation, the case of *sigma* shows that the model sometimes tends to predict co-hyponyms.

4.3 Qualitative Analysis

As a qualitative analysis, we use our pre-trained models to predict which properties are associated with a given concept. We consider the set of all properties that appear at least 10 times in an extended version of the PREFIX+GKB dataset[10], leading to a set of 5223 candidate properties. We again use maximum inner product search to efficiently identify the properties whose embeddings are closest to the concept embedding $\phi_{\mathsf{con}}(c)$. Table 6 shows the seven nearest properties for a number of selected concepts, where we used BERT-base pre-trained on MSCG+PREFIX+GKB. Specifically, the table first revisits the examples from Table 1. Subsequently, the table lists physical concepts, for which we expected predicting properties to be easier, and abstract concepts, for which we expected the task to be harder. Finally, we included adjectives to explore whether our model can be used for learning property entailment.

The results contain a mixture of hypernyms and semantic attributes, which is a reflection of how the model was trained. For physical concepts, the results are generally meaningful, with a few exceptions. For instance, *military vehicle* is incorrectly listed as a hypernym of *airplane*. Regarding the abstract concepts, the top predictions are mostly meaningful, but we can also see terms that are semantically related but are neither hypernyms nor semantic attributes; e.g. we see *parties* as a property of *celebration*. Finally, for the adjectives, we see several instances where the entailment direction is reversed, for instance when *dessert* is mentioned as a property of *sugary*.

5 Conclusions

We studied the problem of modelling the commonsense properties of concepts. We argued that the standard evaluation setting does not faithfully assess the extent to which models capture knowledge about commonsense properties, and proposed two new evaluation settings. These new settings were found to be highly challenging for language models, with performance being close to random. We furthermore found that pre-training a bi-encoder model on hypernymy data or generic sentences can lead to substantial performance gains. However, there remains a lot of room for further improvements, which will likely require novel insights.

Acknowledgments

This work has been supported by EPSRC grant EP/V025961/1. We acknowledge the support of the Supercomputing Wales project, which is part-funded by the European Regional Development Fund (ERDF) via Welsh Government.

[9]The hypernym discovery datasets are strongly biased in which hypernyms were preferred by the annotators. Such biases can only be learned from the task-specific training data, which is why we do not compare with supervised methods.

[10]This extended dataset involves 500K pairs from Microsoft Concept Graph and 500K sentences from GenericsKB; analysis about this extended dataset is provided in Appendix B.

Hyponym	Top-5 Predicted Hypernyms
liberty	principle, notion, ideal, universal value, humanitas
longbow	hunting weapon, **weapon**, bow and arrow, wieldy, choptank
wine	**drink**, **beverage**, liquidity, alcoholic beverage, drinking alcohol
manslaughter	culpable homicide, murder charge, offence, justifiable homicide, first-degree murder
shopping	chore, specific activity, everyday, simple interest, pursuit
running	aerobic, cardio, endurance training, aerobic exercise, sport
computer industry	sector, sunrise industry, growth industry, field of operation, game industry
learner	understander, student, realizer, know-all, nonjoinder
snow	weather condition, weather, cold weather, bad weather, wet-weather
bounty hunter	vigilante, hired gun, bandit, bondman, trail boss
metre	**unit of length**, unit of measure, measuring unit, quantity unit, derived unit
hero	protagonist, archetype, archetypic, personage, literaty character
website	resource, e-resource, information source, **medium**, source
violin	**string instrument**, **musical instrument**, second fiddle, **bowed instrument**, **stringed instrument**
arms	head and shoulders, legs, straighten, stiffen, bare bones
cooking ingredient	composition, culinary, adjunct, importune, condiment
children	learn, memorize, make fun, come to life, lose track
broiler chicken	chicken cordon bleu, chicken stock, hot chicken, kung pao chicken, chicken broth
observation	qualitative, empirical research, qualitative analysis, data collection, qualitative research
sigma	lambda, upsilon, fraternity, epsilon, alpha and omega
apartment	tenantless, adjacent, low-rent, homeplace, residential building
wetsuit	drysuit, nonsuit, life-jacket, diving equipment, diving suit
yesterday	thisday, tomorrow, timea, timeless, evermore
taxi	off-license, car rental, bus service, bike rental, cab fare

Table 5: Error analysis for hypernym discovery on the general dataset. Correctly predicted hypernyms are shown in bold.

Concept	Predicted properties
banana	food, fruit, fresh, plant, edible, tropical, commercially important
lion	animal, mammal, wildcat, carnivore, species, very territorial, mammalian
airplane	vehicle, aircraft, stationary, application, object, military vehicle, automotive
straw	material, combustible, porous, stuff, fibrous, located in wood, has sections
ice	cold, has temperature, has surfaces, located in freezers, has density, authorization, albums
yacht	boat, vehicle, vessel, recreational, ship, expensive, aircraft
coffee	beverages, drinks, beverage, drink, liquid, liquids, located in supermarkets
steel	material, non-ferrous, non ferrous, rigid, product, industrial, heavy
fire	causes burns, creates heat, produces heat, causes damage, can have effects, generates heat, produce crops
beer	beverage, drink, alcoholic, liquor, liquid, beverages, drinks
democracy	principle, idea, democratic, ideology, concept, morality, value, moral
disappointment	negative, feeling, emotion, emotional, feelings, positive, depression
promotion	marketing, achievement, activity, corporate, factor, acts, activities
celebration	event, festivity, occasion, social events, parties, events, activities
forgiveness	moral, value, love, virtue, emotion, benign, principle
lawyer	professional, adult, allied, profession, consultant, closely related, expert
stressful	situation, factor, emotional, difficult, unexpected, uncomfortable, traumatic
poisonous	poison, harmless, harmful, dangerous, toxin, aggressive, sharp
sugary	dessert, taste, food, delicious, chocolate, frozen dessert, candy
rewarding	activities, clocks, happiness, treatments, approval, actions, human activities
modern	style, genre, contemporary, fashion, broad, musical style, english
alcoholic	alcoholic, liquor, drink, beverage, mixed, alcohol, addictive, aggressive

Table 6: Qualitative analysis, showing the top neighbours of the embeddings of selected concepts.

References

Marianna Apidianaki and Aina Garí Soler. 2021. ALL dolphins are intelligent and SOME are friendly: Probing BERT for nouns' semantic properties and their prototypicality. In *Proceedings of the Fourth BlackboxNLP Workshop on Analyzing and Interpreting Neural Networks for NLP*, pages 79–94.

Sumithra Bhakthavatsalam, Chloe Anastasiades, and Peter Clark. 2020. Genericskb: A knowledge base of generic statements. *CoRR*, abs/2005.00660.

Antoine Bosselut, Hannah Rashkin, Maarten Sap, Chaitanya Malaviya, Asli Celikyilmaz, and Yejin Choi. 2019. COMET: Commonsense transformers for automatic knowledge graph construction. In *Proceedings*

of the 57th Annual Meeting of the Association for Computational Linguistics, pages 4762–4779, Florence, Italy. Association for Computational Linguistics.

Zied Bouraoui, Jose Camacho-Collados, and Steven Schockaert. 2020. Inducing relational knowledge from bert. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pages 7456–7463.

Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. 2020. Language models are few-shot learners. *Advances in neural information processing systems*, 33:1877–1901.

Jose Camacho-Collados, Claudio Delli Bovi, Luis Espinosa-Anke, Sergio Oramas, Tommaso Pasini, Enrico Santus, Vered Shwartz, Roberto Navigli, and Horacio Saggion. 2018. SemEval-2018 Task 9: Hypernym Discovery. In *Proceedings of SemEval*, New Orleans, LA, United States.

Shib Sankar Dasgupta, Michael Boratko, Shriya Atmakuri, Xiang Lorraine Li, Dhruvesh Patel, and Andrew McCallum. 2021. Word2box: Learning word representation using box embeddings. *CoRR*, abs/2106.14361.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, Minneapolis, Minnesota. Association for Computational Linguistics.

Maxwell Forbes, Ari Holtzman, and Yejin Choi. 2019. Do neural language representations learn physical commonsense? In *Proceedings of the 41th Annual Meeting of the Cognitive Science Society*, pages 1753–1759.

Daniel Gillick, Sayali Kulkarni, Larry Lansing, Alessandro Presta, Jason Baldridge, Eugene Ie, and Diego Garcia-Olano. 2019. Learning dense representations for entity retrieval. In *Proceedings of the 23rd Conference on Computational Natural Language Learning (CoNLL)*, pages 528–537, Hong Kong, China. Association for Computational Linguistics.

Jonathan Gordon and Benjamin Van Durme. 2013. Reporting bias and knowledge acquisition. In *Proceedings of the 2013 workshop on Automated knowledge base construction, AKBC@CIKM 13, San Francisco, California, USA, October 27-28, 2013*, pages 25–30. ACM.

Michael Hanna and David Mareček. 2021. Analyzing BERT's knowledge of hypernymy via prompting. In *Proceedings of the Fourth BlackboxNLP Workshop on Analyzing and Interpreting Neural Networks for NLP*, pages 275–282, Punta Cana, Dominican Republic. Association for Computational Linguistics.

Marti A. Hearst. 1992. Automatic acquisition of hyponyms from large text corpora. In *COLING 1992 Volume 2: The 14th International Conference on Computational Linguistics*.

Jena D. Hwang, Chandra Bhagavatula, Ronan Le Bras, Jeff Da, Keisuke Sakaguchi, Antoine Bosselut, and Yejin Choi. 2021. (comet-) atomic 2020: On symbolic and neural commonsense knowledge graphs. In *Thirty-Fifth AAAI Conference on Artificial Intelligence, AAAI 2021, Thirty-Third Conference on Innovative Applications of Artificial Intelligence, IAAI 2021, The Eleventh Symposium on Educational Advances in Artificial Intelligence, EAAI 2021, Virtual Event, February 2-9, 2021*, pages 6384–6392. AAAI Press.

Lei Ji, Yujing Wang, Botian Shi, Dawei Zhang, Zhongyuan Wang, and Jun Yan. 2019. Microsoft concept graph: Mining semantic concepts for short text understanding. *Data Intell.*, 1(3):238–270.

Zhengbao Jiang, Frank F. Xu, Jun Araki, and Graham Neubig. 2020. How can we know what language models know? *Transactions of the Association for Computational Linguistics*, 8:423–438.

Lili Kotlerman, Ido Dagan, Idan Szpektor, and Maayan Zhitomirsky-Geffet. 2010. Directional distributional similarity for lexical inference. *Natural Language Engineering*, 16(4):359–389.

Matthew Le, Stephen Roller, Laetitia Papaxanthos, Douwe Kiela, and Maximilian Nickel. 2019. Inferring concept hierarchies from text corpora via hyperbolic embeddings. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 3231–3241, Florence, Italy. Association for Computational Linguistics.

Aoxue Li, Weiran Huang, Xu Lan, Jiashi Feng, Zhenguo Li, and Liwei Wang. 2020. Boosting few-shot learning with adaptive margin loss. In *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 12573–12581.

Xiang Lorraine Li, Adhiguna Kuncoro, Cyprien de Masson d'Autume, Phil Blunsom, and Aida Nematzadeh. 2021. A systematic investigation of commonsense understanding in large language models. *CoRR*, abs/2111.00607.

Xiao Liu, Yanan Zheng, Zhengxiao Du, Ming Ding, Yujie Qian, Zhilin Yang, and Jie Tang. 2021. GPT understands, too. *arXiv preprint arXiv:2103.10385*.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. RoBERTa: A robustly optimized BERT pretraining approach. *CoRR*, abs/1907.11692.

Robert L. Logan, Ivana Balazevic, Eric Wallace, Fabio Petroni, Sameer Singh, and Sebastian Riedel. 2021. Cutting down on prompts and parameters: Simple few-shot learning with language models. *CoRR*, abs/2106.13353.

Ilya Loshchilov and Frank Hutter. 2019. Decoupled weight decay regularization. In *ICLR*.

Yukun Ma, Erik Cambria, and Sa Gao. Label embedding for zero-shot fine-grained named entity typing. In *Proceedings of COLING 2016, the 26th International Conference on Computational Linguistics: Technical Papers*, pages 171–180.

Alfredo Maldonado and Filip Klubička. 2018. ADAPT at SemEval-2018 task 9: Skip-gram word embeddings for unsupervised hypernym discovery in specialised corpora. In *Proceedings of The 12th International Workshop on Semantic Evaluation*, pages 924–927, New Orleans, Louisiana. Association for Computational Linguistics.

Ken McRae et al. 2005. Semantic feature production norms for a large set of living and nonliving things. *Behavior research methods*, 37:547–559.

Tomas Mikolov, Wen-tau Yih, and Geoffrey Zweig. 2013. Linguistic regularities in continuous space word representations. In *Proceedings of the 2013 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 746–751, Atlanta, Georgia. Association for Computational Linguistics.

Tuan-Phong Nguyen, Simon Razniewski, Julien Romero, and Gerhard Weikum. 2021. Refined commonsense knowledge from large-scale web contents. *CoRR*, abs/2112.04596.

Mihaela Onofrei, Ionuț Hulub, Diana Trandabăț, and Daniela Gîfu. 2018. Apollo at SemEval-2018 task 9: Detecting hypernymy relations using syntactic dependencies. In *Proceedings of The 12th International Workshop on Semantic Evaluation*, pages 898–902, New Orleans, Louisiana. Association for Computational Linguistics.

Jeffrey Pennington, Richard Socher, and Christopher Manning. 2014. GloVe: Global vectors for word representation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1532–1543.

Fabio Petroni, Tim Rocktäschel, Sebastian Riedel, Patrick S. H. Lewis, Anton Bakhtin, Yuxiang Wu, and Alexander H. Miller. 2019. Language models as knowledge bases? In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing, EMNLP-IJCNLP 2019, Hong Kong, China, November 3-7, 2019*, pages 2463–2473. Association for Computational Linguistics.

Alec Radford, Karthik Narasimhan, Tim Salimans, and Ilya Sutskever. Improving language understanding by generative pre-training.

Enrico Santus, Alessandro Lenci, Tin-Shing Chiu, Qin Lu, and Chu-Ren Huang. 2016. Unsupervised measure of word similarity: how to outperform co-occurrence and vector cosine in vsms. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 30.

Enrico Santus, Alessandro Lenci, Qin Lu, and Sabine Schulte Im Walde. 2014. Chasing hypernyms in vector spaces with entropy. In *Proceedings of the 14th Conference of the European Chapter of the Association for Computational Linguistics, volume 2: Short Papers*, pages 38–42.

Taylor Shin, Yasaman Razeghi, Robert L. Logan IV, Eric Wallace, and Sameer Singh. 2020. AutoPrompt: Eliciting Knowledge from Language Models with Automatically Generated Prompts. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 4222–4235, Online. Association for Computational Linguistics.

Vered Shwartz and Yejin Choi. 2020. Do neural language models overcome reporting bias? In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 6863–6870, Barcelona, Spain (Online). International Committee on Computational Linguistics.

Richard Socher, Milind Ganjoo, Christopher D. Manning, and Andrew Y. Ng. 2013. Zero-shot learning through cross-modal transfer. In *Proceedings of the 27th Annual Conference on Neural Information Processing Systems*, pages 935–943.

Robyn Speer, Joshua Chin, and Catherine Havasi. 2017. Conceptnet 5.5: An open multilingual graph of general knowledge. In *Proceedings of the Thirty-First AAAI Conference on Artificial Intelligence, February 4-9, 2017, San Francisco, California, USA*, pages 4444–4451. AAAI Press.

Kunihiro Takeoka, Kosuke Akimoto, and Masafumi Oyamada. 2021. Low-resource taxonomy enrichment with pretrained language models. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 2747–2758, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Asahi Ushio, Jose Camacho-Collados, and Steven Schockaert. 2021. Distilling relation embeddings from pretrained language models. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 9044–9062, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Nathaniel Weir, Adam Poliak, and Benjamin Van Durme. 2020. Probing neural language models for human tacit assumptions. In *Proceedings of the 42th*

Annual Meeting of the Cognitive Science Society - Developing a Mind: Learning in Humans, Animals, and Machines, CogSci 2020, virtual, July 29 - August 1, 2020.

Peter West, Chandra Bhagavatula, Jack Hessel, Jena D. Hwang, Liwei Jiang, Ronan Le Bras, Ximing Lu, Sean Welleck, and Yejin Choi. 2021. Symbolic knowledge distillation: from general language models to commonsense models. *CoRR*, abs/2110.07178.

Chen Xing, Negar Rostamzadeh, Boris N. Oreshkin, and Pedro O. Pinheiro. 2019. Adaptive cross-modal few-shot learning. In *Proceedings of the Annual Conference on Neural Information Processing Systems*, pages 4848–4858.

Kun Yan, Zied Bouraoui, Ping Wang, Shoaib Jameel, and Steven Schockaert. 2021. Aligning visual prototypes with BERT embeddings for few-shot learning. In *Proceedings of the International Conference on Multimedia Retrieval*, pages 367–375.

Wangchunshu Zhou, Dong-Ho Lee, Ravi Kiran Selvam, Seyeon Lee, and Xiang Ren. 2021. Pre-training text-to-text transformers for concept-centric common sense. In *9th International Conference on Learning Representations, ICLR 2021, Virtual Event, Austria, May 3-7, 2021*. OpenReview.net.

A Prompt Analysis

Previous work has found that the prompt which is used can have a material impact on the performance of BERT-based encoders (Bouraoui et al., 2020; Jiang et al., 2020; Shin et al., 2020; Liu et al., 2021; Ushio et al., 2021; Logan et al., 2021). To analyse the impact of the prompt in our setting, and make a suitable choice, we experimented with a number of different, manually chosen prompts. For these experiments, we used the most confident 11,000 concept-property pairs of the MSCG dataset for training, and the next 1200 concept-property pairs for tuning. The batch size is set to 8. We used the AdamW optimizer and learning rate $2e{-}6$, using early stopping with a patience of 20. The results in Table 7 are reported in terms of the F1 score (percentage) of the positive label. For the first two results in the table, a different prompt was used for the concept and property encoders. The property prompts corresponding to these two configurations are (not shown in the table):

- $\langle cls \rangle$ Property: [CONCEPT] $\langle sep \rangle$
- $\langle cls \rangle$ Yesterday, I saw a thing which is [PROPERTY] $\langle sep \rangle$

For the first six configurations in the table, we use the average of the embeddings of all tokens, in the final layer of the BERT-base model, as the embedding of the concept and property. For the remaining seven configurations, we use the embedding of the $\langle mask \rangle$ token in the final layer instead. The results show that many of the prompts lead to a relatively similar performance, as long as the prompt is sensible. The example with the nine mask tokens (Prompt 5) show that without a semantically informative prompt the performance drops somewhat. A similar observation can be made for the prompt about the spaceship (Prompt 10). Earlier work has suggested that longer prompts tend to perform better. To some extent this is confirmed by our results. For instance, Prompt 12 outperforms the similar but shorter Prompts 9 and 11, although Prompt 13, which is an even longer variant, performs worse. Moreover, we can see that some of the shortest prompts nonetheless perform well. All things being equal, having a shorter prompt is desirable, as it means we can use larger batch sizes and faster training. For this reason, we have decided, based on these results, to use Prompt 7, whose performance is close to that of the best-performing prompt, despite also being one of the shortest ones.

B Size of the Pre-Training Corpus

The MSCG corpus was obtained by taking the 100K pairs from Microsoft Concept Graph (Ji et al., 2019) with the highest confidence. Similarly, GKB was obtained by taking the 100K sentences with the highest confidence in GenericsKB (Bhakthavatsalam et al., 2020). This choice represents a trade-off: choosing more pairs would increase the overall amount of training data, which could improve the performance of the encoders. However, this would also mean including less reliable pairs, which might have a negative effect. In particular, both Microsoft Concept Graph and GenericsKB have been extracted from text corpora. In both cases, it can be clearly observed that the pairs/sentences with the lowest confidence are often rather noisy. To analyse this trade-off, Table 8 shows the results of an experiment where we used the top 500K pairs in MSCG and the 500K most confidence sentences in GenericsKB. Similarly, PREFIX was derived from the larger MSCG dataset for these experiments. The results show a small improvement for MSCG. However, for the *Prop* and *C+P* settings, the GKB results are actually worse for the 500K setting. These results suggest that the optimal setting might use more than 100K pairs from

	Prompt	F1
1.	⟨*cls*⟩ Concept: [CONCEPT] ⟨*sep*⟩	85.6
2.	⟨*cls*⟩ Yesterday, I saw another [CONCEPT] ⟨*sep*⟩	86.1
3.	⟨*cls*⟩ The notion we are modelling is [CONCEPT] ⟨*sep*⟩	86.7
4.	⟨*cls*⟩ The notion we are modelling: [CONCEPT] ⟨*sep*⟩	87.3
5.	⟨*cls*⟩ ⟨*mask*⟩ ⟨*mask*⟩ ⟨*mask*⟩ ⟨*mask*⟩ ⟨*mask*⟩ [CONCEPT] ⟨*mask*⟩ ⟨*mask*⟩ ⟨*mask*⟩ ⟨*mask*⟩ ⟨*sep*⟩	84.8
6.	⟨*cls*⟩ The notion we are modelling is called CONCEPT ⟨*sep*⟩	86.0
7.	⟨*cls*⟩ CONCEPT means ⟨*mask*⟩ ⟨*sep*⟩	87.1
8.	⟨*cls*⟩ CONCEPT ⟨*sep*⟩ ⟨*mask*⟩ ⟨*sep*⟩	86.6
9.	⟨*cls*⟩ The notion we are modelling is CONCEPT ⟨*sep*⟩ ⟨*mask*⟩ ⟨*sep*⟩	86.8
10.	⟨*cls*⟩ The spaceship we are modelling is CONCEPT ⟨*sep*⟩ ⟨*mask*⟩ ⟨*sep*⟩	85.8
11.	⟨*cls*⟩ We are modelling CONCEPT ⟨*sep*⟩ ⟨*mask*⟩ ⟨*sep*⟩	86.4
12.	⟨*cls*⟩ The notion we are modelling this morning is CONCEPT ⟨*sep*⟩ ⟨*mask*⟩ ⟨*sep*⟩	87.0
13.	⟨*cls*⟩ As I have mentioned earlier, the notion we are modelling this morning is CONCEPT ⟨*sep*⟩ ⟨*mask*⟩ ⟨*sep*⟩	86.3

Table 7: Performance of different prompts on a held-out portion of the MSCG dataset, in terms of F1-score percentage. BERT-base was used as the language model in these experiments.

		Con	Prop	C+P
500K	MSCG	80.1	48.6	42.8
	PREFIX	78.1	45.0	41.7
	GKB	80.6	48.8	43.7
	MSCG+PREFIX	79.8	49.1	43.4
	MSCG+GKB	**80.7**	48.8	41.5
	MSCG+PREFIX+GKB	80.3	47.5	41.0
100K	MSCG	79.9	46.6	41.6
	PREFIX	78.3	44.8	41.0
	GKB	79.3	**50.7**	**46.0**
	MSCG+PREFIX	80.2	47.8	43.2
	MSCG+GKB	80.4	50.3	43.6
	MSCG+PREFIX+GKB	79.8	49.6	44.5

Table 8: Evaluation on the McRae dataset of a variant in which 500K pairs from Microsoft Concept Graph and GenericsKB were used. Results are reported in terms of F1 score percentage. BERT-base was used as the language model in these experiments.

Microsoft Concept Graph, but fewer than 500K sentences from GenericsKB. However, the results also show that any performance gains arising from optimising the selection of the pre-training data are likely to be small.

COIN – An Inexpensive and Strong Baseline for Predicting Out of Vocabulary Word Embeddings

Andrew Schneider
Near Miss Management
schneider@nearmissmgmt.com

Lihong He
IBM Research, Almaden
lihong.he@ibm.com

Zhijia Chen
Temple University
zhijia.chen@temple.edu

Arjun Mukherjee
University of Houston
arjun@cs.uh.edu

Eduard Dragut
Temple University
edragut@temple.edu

Abstract

Predicting word embeddings for out of vocabulary words remains an important challenge for NLP tools. Word embedding models only include terms that occur a sufficient number of times in their training corpora. Word embedding vector models attempt to approximate information about a word not in their vocabularies. We propose a fast method for predicting vectors for out of vocabulary terms that makes use of the surrounding terms of the unknown term and the hidden context layer of the word2vec model. We propose this method as a *strong baseline* in the sense that 1) while it does not surpass all state-of-the-art methods, it surpasses several techniques for vector prediction on benchmark tasks, 2) even when it underperforms, the margin is small retaining competitive performance in downstream tasks, and 3) it is inexpensive to compute, requiring no additional training stage. We also show that our technique can be incorporated into existing methods to achieve a new state-of-the-art on the word vector prediction problem.

1 Introduction

In recent years, distributive models of lexical semantics, i.e., word embedding models, have proven to be a very useful tool for representing natural language terms (i.e., words and common phrases such as 'New York') as real numbered vectors. These models, such as *word2vec* (Mikolov et al., 2013a,b), *GloVe* (Pennington et al., 2014), and *FastText* (Bojanowski et al., 2017), use a large corpus of documents to learn an embedded vector representation of lexical units based on their co-occurrence within sentences. Word embedding vectors capture semantic features of the terms, for example synonymous terms will be nearby in the embedded vector space. This makes word embeddings more powerful than other lexical representations such as one-hot vector encodings. Furthermore, distributive models are derived using unsupervised algorithms, which are efficient to run on large corpora. A major limitation, however, is that to derive a high quality word embedding vector for a term, it must occur more than a certain times in the training corpus (Bahdanau et al., 2018). Terms that do not meet this threshold are not included in the model's vocabulary.

When using a word embedding model, a user may encounter a large number of Out of Vocabulary (OOV) terms for various reasons. When we encounter OOV terms in a document, a common strategy is to simply ignore the unknown term. Alternately we can represent the term by a zero vector, or by the average of all known word vectors, but these strategies clearly do not extract any meaningful information from the OOV item.

A number of recent efforts propose techniques for predicting high quality word embedding vectors for OOV terms from the information available on the term *in situ* (Lazaridou et al., 2017; Herbelot and Baroni, 2017; Khodak et al., 2018; Li et al., 2017; Luong et al., 2013; Lazaridou et al., 2013; Schick and Schütze, 2019b). An important downside of these methods in practice is that they take an unpredictable amount of time to develop and deploy. A question arise in this context for deployment in an NLP pipeline: *Can one employ (and deploy) a simpler solution (with minimal accuracy loss) until one creates a more advanced solution?* The goal of this work is to seek answers to this question in the context of OOV terms.

In this work, we present a context-based method to predict the word embedding vectors for OOV terms. Our method makes use of context information that is learned internally in the course of training a word embedding model such as *word2vec*. We call this the **COIN** (**Co**ntext **In**formation) method. We advocate COIN as a *competitive baseline* in the following sense:

Accurate (compared to state-of-the-art models). The performance of COIN is close, both in direct comparison and downstream tasks, to state-of-the-

Proceedings of the 29th International Conference on Computational Linguistics, pages 3984–3993
October 12–17, 2022.

art models. We compare COIN with four such models (ADD, N2V, ALC, and FCM) on the benchmark tasks DefNonce and CRW; COIN vectors meet or exceed the accuracy of all except for FCM.

Inexpensive. Generating a COIN vector prediction requires only a pretrained word2vec model and runs on a regular personal laptop (without the requirement of GPU). The only computational cost of COIN is the equivalent of a few look-ups of word–vector mappings. It is able to generate high quality predicted word vectors for OOV terms from as few as one observed occurrence without any additional training. While the state-of-the-art method FCM requires to train a set of word vectors and then in a second step trains a model for OOV predictions.

Fast. COIN is much faster than other state-of-the-art models. It gives users immediate feedback (in seconds). While most supervised techniques for word embedding for OOV terms require hours of computation before a user can analyze it. In our experiments, the COIN method takes 131 seconds to generate the vectors of OOV terms for evaluation on the DefNonce task, and it only takes 21 seconds to run all of the 2, 4, and 6 sentence evaluations on the Chim task. In comparison, FCM takes nearly 7 hours of training time on the DefNonce task.

Balancing gains with cost. OOV words are important in downstream tasks (Chen et al., 2019; Conneau and Kiela, 2018; Garneau et al., 2019; Lourentzou et al., 2019; Serrà et al., 2017) and not as a standalone exercise. We report a study on 7 SentEval tasks, getting a .002 prediction difference between COIN at 73.9% and FCM at 74.1%. With the compute and energy demands of many modern NLP methods growing exponentially, one needs to consider weighing energy costs with the performance gains of a new model before deploying it (Henderson et al., 2020). In our setting, the gain of energy hungry methods is marginal compared to COIN. However, *many of the alternative approaches require a second step than training the initial word embedding model*. All of the training required for COIN is accomplished with the original word embedding step as a holistic model minimizing total parameters, improved performance and a better model design.

We can draw a parallel from daily life. Toyota Corolla is an excellent means of transportation for millions of people. Ferrari is another excellent means of transportation, but only available to the few who can afford it. One may view COIN (and methods in its class) as the "Corolla" among the methods of predicting embeddings for OOVs and accept that its performance may fall short of the "Ferrari's" (e.g., fastText, ELMo and Bert), one also needs to accept that the "Ferrari's" requires more resources: fuel (i.e., energy) and maintenance (i.e., hardware). Often scientists and engineers need a "Corolla" to deliver a proof of concept. There are times when their hardware infrastructures are ill-suited to run the "Ferrari's," e.g., lacking tens of GPUs. Our goal in this paper is to argue that there is a place for "Corolla's," even though we all seek shiny "Ferrari's."

COIN vectors are suitable on their own for word vector prediction: They can be used to initialize a trained method. We show that using COIN vectors as the first step in existing vector predicting systems can set a new state-of-the-art on benchmark tasks. We also notice that COIN vectors may reduce the training cost for those methods, improving the accuracy of the component techniques. We show these on FCM in Section 5.5.

In summary, the contributions of this paper are:

- propose COIN, a fast and inexpensive method for predicting a word vector for OOV terms based on context information.
- show that COIN is effective and efficient on word vector prediction tasks, working as a strong baseline and occasionally beating other state-of-the-art models.
- show that with COIN vectors as an initialization for existing techniques improves their accuracy.

2 Word Embeddings

Word embeddings are a technique for learning a vector representation of terms from a corpus based on their distributional features, that is, the contexts within which a term is observed (Bengio et al., 2003; Mikolov et al., 2013a,b; Pennington et al., 2014; Bojanowski et al., 2017). Word embedding techniques belong among a class of machine learning algorithms where the model is trained to minimize error on a dummy task and the true purpose of training is to extract an intermediate, internal representation of the input data. In the word embedding case, the dummy task is that of predicting the surrounding terms of a target term (or vice versa) and the true desired output is the internal representation of the input terms, i.e., the word embedding vectors. The various methods for learning word embeddings

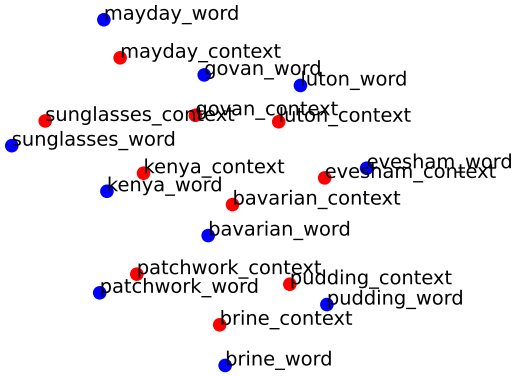

Figure 1: A t-SNE visualization of the relative locations of word and context vectors for a sample vocabulary. The word and context spaces are aligned using absolute orientation with translation and scaling (Dev et al., 2021). For most terms the word and context vectors are respectively well separated.

present variations on a core idea: each term is represented by two latent vectors: a *word vector* and a *context vector*. In training, the word embedding algorithm maximizes the conditional probability of the word vector given the set of context vectors of the terms found in its context window.

Word2vec is one of the most popular technique to learn word embeddings (Mikolov et al., 2013a). In *word2vec* model, the word vectors are not always close to the context vectors. In Figure 1 we show a visualization of the word and context vectors for some terms from the vocabulary. We find that the context and word vectors are quite far from each other for most terms. This general difference between vectors demonstrates that summing the context vectors will lead to a very different result from summing over the word vectors. This suggests that *context vectors encapsulate information that is not captured by the word vectors*, which may be useful to predict embeddding vectors for OOV terms. We aim to show this in this work. We note here that Figure 1 was generated after the word and context vector were aligned. We align the word and context vectors using absolute orientation with translation and scaling (Dev et al., 2021). Other orientation algorithms are available, but this gave the best average Cosine similarity between the vectors in our experiments. We also note that the context and word vectors as plotted in Figure 1 are even farther apart before alignment was performed.

Following our study of word embeddings, we observe that the word embedding vector $\mathbf{w}_i$ of a word is trained to be similar to the *context vectors* of the words in its context window, $\mathbf{c}_{j \in C_i}$, rather than the *word vectors*, which have previously been used in the prediction task. Intuition suggests that by using the context vectors of the context words we can improve the accuracy of our predicted vectors.

3 Related Work

Previous techniques to predict embedding vectors for OOV terms usually follow two theoretical lines: 1) Using morphological or form features of the OOV term, and 2) Extracting information from the surrounding context terms.

Form based methods in category 1) use some characteristics of the representation of the term, such as letter combinations. (Luong et al., 2013; Lazaridou et al., 2013) use morphological features. (Bojanowski et al., 2017) develops the *FastText* model, which is similar to word2vec in design, which operates on character n-grams rather than whole words. In some tasks, however, an OOV term may not have any character representation, or it may have an arbitrary representation, for example, the OOV term may be represented by underscores: "___" (Herbelot and Baroni, 2017) or by a made-up word form (Lazaridou et al., 2017). In the downstream task there may be no available form for the term, such as "filling in the missing word."

For the context based works in category 2), (Lazaridou et al., 2017) develop the additive method, which takes the sum of the word vectors of the surrounding terms. The approach of (Herbelot and Baroni, 2017) initializes each OOV term as the sum of the word vectors of the known terms in the context sentences, downsampling frequent words, and then runs a modified word2vec training procedure on the OOV terms only, with a highly accelerated learning rate. (Khodak et al., 2018) finds a linear transformation $\mathbf{A}$ and uses it to transform the sum of the context word vectors to the word vector of OOV terms. Recently (Schick and Schütze, 2019a) propose a hybrid approach using both form and context. It learns a neural network model, which encodes information about the form (n-grams) and likelihood the form will contribute to an understanding of the meaning of the term, as well as a linear transformation similar to (Khodak et al., 2018). (Schick and Schütze, 2019a) further proposes a method to identify high quality contexts for inferring the OOV term vector.

COIN vectors squarely fall into 2), context based methods. Our key observation is that there exists an additional set of vectors, one for each in-vocabulary

item, which are trained internally by the word2vec model. In word2vec, each term is represented by two latent vectors: a *word vector* and a *context vector*. These vectors are better suited, by design, for deducing OOV term vectors from the surrounding context words. When the model is being trained, these additional *context vectors* are used to represent the terms when they are found in the context window of the target term. These vectors can be easily extracted from a word2vec model.

Recently *contextualized word vectors* (CWV) methods such as ELMo and BERT have received much attention in the NLP community (Horn, 2017; Peters et al., 2018; Devlin et al., 2018). CWV prediction shares some similarity with OOV vector prediction, but they are intrinsically distinct problems. CWVs derive a vector representation for a term based on the surrounding context in each usage, thus there is not a generalized concept of a word vector for a lexical item as is the case with traditional word vectors. While in traditional word vector models (like word2vec), each term gets one vector, which is used regardless of the surrounding terms. We include them in our empirical study on the OOV benchmark tasks.

4 Methodology

Based on our observation discussed previously, we suggest that *the context vectors of the context words* can be used to predict the word embedding vectors for OOV terms. We introduce our method, COIN, which uses context vectors only to solve the prediction problem for OOV terms.

4.1 Problem Definition

For some word embedding model M, let $\mathcal{V}$ be the vocabulary of terms for which we have a trained word vector. Then there exists a mapping $\mathbf{w} : \mathcal{V} \rightarrow \mathbb{R}^n$ where n is the dimension of the embedded vectors. For a term $t \notin \mathcal{V}$, let $\hat{\mathbf{w}}(t) \in \mathbb{R}^n$ be the optimal word vector for t based on its semantic properties and the properties of the embedding space. Assuming $\hat{\mathbf{w}}(t)$ is known in our problem. We define the problem as follows:

Given a set of terms $\mathcal{U}$, where $\mathcal{V} \cap \mathcal{U} = \emptyset$, find a function f such that $\forall t \in \mathcal{U}$, $f(t) \approx \hat{\mathbf{w}}(t)$.

4.2 Proposed Approach

To find a COIN vector for an OOV term, we take the sum of the context vectors of the words in its contexts. Let $\mathbf{c} : \mathcal{V} \rightarrow \mathbb{R}^k$ be the mapping of terms to context vectors in M. For the n-th occurrence of term t_i, its context window C_i^n is the set of $2 \times k_w$ terms that are found up to k_w places before and after t_i, excluding t_i itself, for context window size k_w. Then for term t_i we define the COIN vector prediction $\hat{f}_{COIN}(t_i)$ as:

$$\hat{f}_{COIN}(t_i) = \sum_j \sum_{t' \in C_i^j} \mathbf{c}(t')$$

where each C_i^j is one observed context window.

There are some small modifications to the basic COIN pattern that are easily introduced. These include: stop word removal, word weighting, and principal component removal. Stop word removal (denoted **nsw**) ignores stop words in the vector summation, and generally gives some improvement in accuracy. We apply the stop words provided in python NLTK in our experiments. Word weighting multiplies each vector by a function of its term, so that certain words have more importance in the final sum. Here we consider SIF (Arora et al., 2017) which weights each term by the inverse of its frequency. Principal component removal (Mu and Viswanath, 2018) removes the top principal component from each vector, which has been shown to improve the performance of word vectors on similarity tasks. On the CRW task we show that a combination of all three yields good results.

Additionally, we consider separately how the various setting for learning word embeddings via *word2vec* and the resulting *context vectors* are suited for estimating OOV word vectors. We consider the skip-gram and CBOW architectures, using both HS and NS respectively for updating. Section 5.6 discusses the difference among these models.

5 Evaluation

In this section, we compare our method COIN to a number of state-of-the-art models to show the efficiency and effectiveness of COIN. We also show that COIN vectors can be used in conjunction with an existing, trained method to yield improved results over either method alone

5.1 Models in Comparison

We compare COIN against recent OOV prediction models and CWV methods in the literature:
ADD (Lazaridou et al., 2017) produces an OOV prediction as the sum of the word vectors of all the neighboring terms. We also consider **ADD-nsw**, a same model with stop words removed.

N2V (Herbelot and Baroni, 2017) updates a set of OOV terms initialized by an additive method, by running an accelerated skip-gram training.

ALC (Khodak et al., 2018) learns a linear transformation on the set of ADD vectors that minimizes the ℓ_2 distance to the known word embedding vector for a set of training terms.

FastText (Bojanowski et al., 2017) is a morphological embedding algorithm that extends *word2vec*. It represents each word as an n-gram of characters instead of learning vectors for words directly. We use its implementation in gensim[1].

FCM is the Form Context Model of (Schick and Schütze, 2019b); it is a two part model that learns both a form and context component.

HiCE. HiCE (Hu et al., 2019) is an attention-based hierarchical context encoder that uses both the contexts and morphological features of an OOV word.

BERT. We take the embedding output from the last hidden layer of BERT (Devlin et al., 2018) as the vector representation for a word in our experiment.

ELMo (Peters et al., 2018) is a deep contextualized word representation; the representation for a word depends on the entire context in which it is used.

Model Parameter Settings. For all the methods (e.g., COIN, ADD, N2V, ALC, and FCM) that depend on a pre-trained *word2vec* model, we use the one provided by (Herbelot and Baroni, 2017). We run the experiments using the source codes and settings published in the corresponding referenced paper. For the method FastText, we set embedding size to 400 (same as the embedding size in the *word2vec* model), parameter window is set to 5, the minimum count is set to 1, and the model is trained for 5 epochs. We quote the HiCE experiment results from (Hu et al., 2019). BERT and ELMO are context dependent CWV methods that generate different word embeddings for the same word in different sentences. We use the pre-trained BERT base model from Hugging Face[2] which contains 12 hidden layers, and the pre-trained ELMo model from AllenNLP[3]. The BERT and ELMo embeddings for the OOV words are averaged by their context words' embeddings.

[1] https://radimrehurek.com/gensim/

[2] Facehttps://huggingface.co/bert-base-uncased

[3] http://docs.allennlp.org/v0.9.0/api/allennlp.modules.elmo.html

name	DefNonce	enwiki	WWC
dimension	400	400	300
vocab size	259,376	560,881	230,130
min count	50	50	50
source	Wikipedia	Wikipedia	WWC
model type	skip-gram	CBOW skip-gram	CBOW skip-gram
updating method	NS	NS HS	NS HS
window size	5	5	5

Table 1: Specifications of training word2vec embeddings in three datasets. The sample number in negative sampling (NS) is 5. The value of alpha is set to 0.025 in skip-gram, 0.05 in CBOW.

5.2 Benchmark Tasks

We evaluate the quality of our COIN embeddings on the common OOV benchmark tasks: Definitional Nonce (DefNonce), Contextual Rare Words (CRW), and Chimeras (Chim).

5.2.1 DefNonce Task

DefNonce (Herbelot and Baroni, 2017) is a set of sentences harvested from Wikipedia articles which are designed to be maximally informative. The first sentence of articles describing one word topics are extracted. The target term is the article title. Sentences that contain at least 10 words are randomly sampled creating 700 training and 300 test instances. The FastText model is trained on these training instances. The goal for this evaluation task is to learn a word vector for the target term that is close to the known word vector for that term. (Herbelot and Baroni, 2017) provide a set of pre-trained word2vec vectors trained on a $1.6B$ word Wikipedia snapshot using the skip-gram architecture, with negative sampling. The word2vec model parameters are shown in Table 1 (See DefNonce).

Experimental Results on DefNonce. Table 2 gives the results of this evaluation. Accuracy on this task is measured by two values: Mean Reciprocal Rank (MRR), where higher is better, and Median Rank, where lower is better. We show the form and context components of FCM, denoted FCM-form and FCM-ctxt respectively, along with the full FCM model. Of all the models FCM performs the best on this task. However if we limit our focus to methods which only use context information (see Part 2), COIN is more accurate than either ALC or FCM-ctxt, and COIN-nsw is the leader in

both measures. With COIN-nsw, for all the terms in the test set, half are among the 90 closest terms, out of a vocabulary size of 259,376.

Comparing FCM with FCM-ctxt and FCM-form, it seems that much of the improvement for FCM comes from the form information, which is suggested by the relative performance of FCM-ctxt and FCM-form. We also observe that while the MRR for FCM-form is high, so is the Median Rank, suggesting that this model performs very well on some terms, presumably morphologically rich examples, but poorly on a good number of other terms.

Between the two CWV methods, ELMo performs better. It gives a competitive Median Rank, the second lowest, better than COIN's, but not as good as FCM's. Its MRR is not as competitive. Our explanation is that ELMo performs well when sentences have informative context for OOV words. But its predicted vector representation of an OOV term is far from the "true" representation when sentences contain less informative context.

Comparison of Runtime. We repeat the experiments of each method 10 times and report their averaged runtimes together with the standard deviations (STD) in the last column of Table 2. We ran the experiments on a PC with an Intel i7-8700K CPU @ 3.70GHz, 64GB RAM @ 2133 MHz, and a NVDIA GeForce GTX 1080 GPU. We find that COIN and its variant COIN-nsw have a good balance between accuracy and runtime. For example, FCM achieves the best performance with 0.1754 MRR and 49 Median Rank, but it requires over 5 hours, which is orders of magnitude larger than that of COIN (415 seconds) and COIN-nsw (423 seconds). FastText runs the fastest (30 seconds) but with a dramatic sacrifice in accuracy (0.0095 MRR and 2202 Median Rank). Put together, the running time and accuracy reported in Table 2, support our claim that COIN is a strong, inexpensive baseline for predicting OOV word embeddings.

	method	MRR. ($\times 10^{-2}$)	Med. Rank	Time in sec avg (STD)
Part 1	FastText	0.95	2202	30 (0.39)
	FCM-ctxt*	6.56	184	/
	FCM-form*	12.98	404	/
	FCM*	**17.54**	**49**	19,236 (682)
Part 2	N2V*	4.91	623	1,166 (12.99)
	ALC*	7.06	165	842 (12.83)
	ADD	0.95	3881	414 (1.72)
	ADD-nsw	3.62	876	429 (9.83)
	COIN	9.43	100	415 (1.38)
	COIN-nsw	**9.46**	**90**	423 (4.85)
Part 3	BERT	2.34	242	518 (9.74)
	ELMo	4.25	62	1,090 (12.81)

*Performance of these methods are quoted from the referenced paper. Our re-runs may differ slightly.

Table 2: Results on the DefNonce task. Methods are divided in into three groups. Methods in Part 1 make use of context and other information, the ones in Part 2 only use context information, Part 3 is CWV methods.

5.2.2 CRW Task

CRW (Khodak et al., 2018) is a subset of 562 pairs of words from the Rare Word dataset (Luong et al., 2013) combined with 255 sentences for each rare word sampled from the Westbury Wikipedia Corpus (WWC) (Shaoul and Westbury, 2010). Each pair of words has been manually annotated with a similarity score. The 255 context sentences are partitioned into eight disjoint subsets of sizes 1, 2, 4, . . . , 64, 128. The goal is to match the human similarity scores between pairs of words in the CRW, measured using Spearman correlation. The authors provide a set of word2vec word embeddings trained on a subset of the WWC from which all sentences containing a rare word have been removed. However they only include the trained word embeddings, without the context vectors needed for our technique. Therefore we train our own set of embeddings using their original training data, trying to follow their parameters as closely as possible (See column WWC in Table 1). We also train FastText embeddings on this training data.

Experimental Results on CRW. Figure 2 gives the Spearman correlation values across the sample sizes, for various model designs. The basic COIN vectors perform better than ADD on this task. Following (Khodak et al., 2018) we explore SIF-weighted vectors, with stop words removed and top principal component removal. The resulting COIN vectors, COIN-nsf, perform about as well as ALC on this task across context sizes. FCM performs the best for all context sizes. (Schick and Schütze, 2019b) points out that the Rare Word dataset was designed to contain many morphologically analyzable words. Form based methods, such as FCM, likely benefit from this design.

5.2.3 Chim Task

Chim (Lazaridou et al., 2017) constructs novel concepts by combining the contexts of pairs of existing terms to derive a "chimera" of their attributes, for example "alligator" and "rattlesnake." Human annotators rank the similarity of the chimeric concept to a set of six probe words (see the example in

input terms:		*probes:*					
alligator	rattlesnake	crocodile	iguana	gorilla	banner	buzzard	shovel
		2.29	3.29	3.43	2.0	3.71	2.14
sentences:							
1. animals such as capybara jaguars jacare ___ and hyacinth macaws are particularly vulnerable							
2. nadirpur stared at it as though it were a ___ his face quite drained							

Table 3: Chimera example for input pairs and probes. Similarities have been determined by human judges.

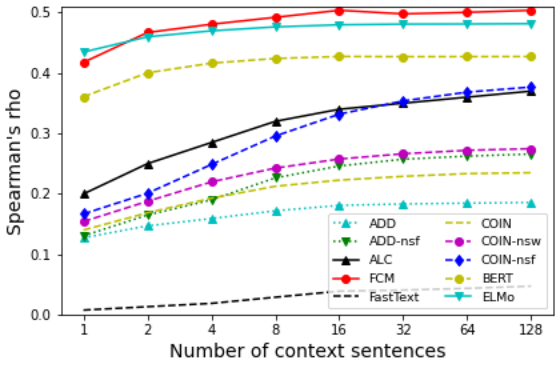

Figure 2: Spearman correlation results for CRW task.

	method	2 sent.	4 sent.	6 sent.
Part 1	FastText	0.178	0.174	0.129
	FCM-ctx	0.337	0.359	0.422
	FCM-AM-ctx	0.342	0.376	**0.436**
	HiCE	**0.378**	**0.405**	0.431
Part 2	N2V	0.332	0.367	0.389
	ALC	0.363	0.384	0.394
	ADD	0.303	0.340	0.337
	ADD-nsw	**0.354**	**0.379**	**0.416**
	COIN	0.299	0.290	0.359
	COIN-nsw	0.336	0.323	0.395

Table 4: Results on the Chim task reported as Spearman correlation. All methods are grouped into two parts: Part 1 and 2, as in Table 2. Neither ELMo nor BERT can be used in this task as there is no sentence for the probe words to learn their embeddings.

Table 3). Context sentences are provided for each concept in sizes 2, 4, and 6 (half for each of the existing terms). The goal of this trial is to match the similarities of the probes to the chimeras as given by the human judges, measured by Spearman correlation. The word2vec model used on this task is the same as the one on task DefNonce. The FastText model is trained on WikiText-103 (Merity et al., 2016), following the way in (Hu et al., 2019).

Experimental Results on Chim. Table 4 gives the results. CWV methods cannot be used in this task as there is no sentence for the probe words to learn their embeddings. A big challenge of this task is that the randomly selected sentences are not necessarily informative about the concepts. For the relatively larger sample size of six sentences a selective model such as FCM-AM-ctx (Schick and Schütze, 2019a) is able to focus on the contexts that will be most helpful. Among the context-based methods, ADD-nsw performs the best. But the performance of our method, COIN-nsw, is very close to that of ADD-nsw.

We observe that COIN is much better in the DefNonce task but not in Chim task. A possible explanation is that the sentences for the OOV words in Chim task are randomly selected and thus, are not necessarily informative about the nature of the OOV words. In COIN, we get the vectors of OOV words based on the context words in the sentences and use these vectors to calculate the similarity between words. In contrast, the gold standard similarity between words is from human rankings. We hypothesize that the intrinsic less information of COIN vectors (from these sentences) may be the reason for COIN not performing that well in Chim.

5.3 Downstream Tasks

Following (Schick and Schütze, 2019b), we conduct an experimental study using the SentEval evaluation toolkit (Conneau and Kiela, 2018) which contains various types of sentence classification tasks, including sentiment analysis (MR, SST2, and SST5) (Socher, 2013; Hosseinia et al., 2019; Schneider and Dragut, 2015), product reviews (CR) (Hu and Liu, 2004; Hosseinia et al., 2020), subjectivity/objectivity (SUBJ) (Pang and Lee, 2004), opinion polarity (MPQA) (Hosseinia et al., 2021; Tumarada et al., 2021; Wiebe et al., 2005; Yang et al., 2020), and paraphrase detection (MRPC) (Dolan et al., 2004; Aljebreen et al., 2021). We replace the OOV terms with their COIN and FCM predictions, and our results mirror those reported in (Schick and Schütze, 2019b), showing no significant change among the OOV prediction methods on the results (COIN 73.9% to FCM 74.1%).

method	MRR.	Median Rank
FCM	0.1724 (0.1726)	53.8 (50)
COIN+FCM	**0.1731** (0.1735)	**52.7 (46)**
p-value	0.0003	0.0008

Table 5: Results on the DefNonce task. FCM initialized with COIN vectors improves both MRR. and Median Rank. We give the average and best (in parenthesis) outcomes for both MMR. and Median Rank.

5.4 Efficiency and Effectiveness of COIN

According to the comparison between COIN and other models, our method COIN is very competitive, typically outperforming all but FCM in many of the tasks. However a significant advantage to our technique is that it requires no training beyond learning the initial word embedding model, while FCM requires a significant amount of training time and computing resources. In our experiments, training the FCM model took 5.34 hours on average. We stress that COIN requires no such additional training time. The only computational cost of COIN is the equivalent of a few look-ups of word–vector mappings (in seconds). Specifically on the 300 test words of the DefNonce task, the COIN method takes 131 seconds from loading the word2vec model, generating the COIN vectors to evaluation. The COIN method takes 21 seconds to run all of the 2, 4, and 6 sentence evaluations on the Chim task. *This makes our technique substantially more practical to use, particularly in development stages, giving the user a strong starting point.* In a real world situation when we encounter an OOV term without having previously trained a model for one of the trained methods on a corpus of similar documents, COIN can quickly give a predicted word vector based on the context at hand, with accuracy close to the cutting edge techniques. *In a scenario where it is unknown if a more complicated model may make a difference on the downstream result, a user can use COIN to provide a first look while training another OOV prediction model, whereby the COIN model provides a good basis to extend and build upon.*

5.5 Initialization with COIN Vectors

The process of deriving COIN vectors is very quick and efficient, therefore they provide a good model for initializing other prediction methods. Typically these methods begin with an additive model for OOV terms, then further refine the embeddings through some training strategy (Herbelot and Baroni, 2017; Khodak et al., 2018; Schick and Schütze, 2019b). COIN vectors can be used in place of the additive vectors as the first level approximation. In this study our goal is to show that COIN vectors can be used in conjunction with an existing, trained method to yield improved results over either method alone. We focus here on the FCM method and investigate a modified design which uses COIN vectors to initialize the model.

On the DefNonce task, COIN achieves better results than the FCM-context piece alone as shown in Table 2. We train the models as described in (Schick and Schütze, 2019b) using the vectors provided by (Herbelot and Baroni, 2017). We train the standard FCM model and our combined model, COIN+FCM, on the same training data used in their paper, for 10 epochs. Selecting the best performing epoch for each algorithm on the training set, we then run this model on the test data. We repeat the experiments of FCM and COIN+FCM 10 times, and report their average and best performance in Table 5. We notice that using COIN vectors to initialize the model improves the performance in both MRR and Median Rank. The improvement is slight but consistent in every randomized run. Thus, we calculate the p-value to verify that the performance differences are significant. With a p-value = 0.0003 for the MRR. and p-value = 0.0008 for the Meidan Rank, the results are significant at the 0.05 level.

We note that while the training data is the same as used in (Schick and Schütze, 2019b), the FCM performances here differ slightly with that in Table 2, this is because the results here are averaged over 10 repeats, and the outcome of each run varies due to inherent randomness in the model. We also run the CRW task comparing the FCM model and COIN+FCM model. The results show that COIN+FCM achieves a slight improvement at most context sizes. The largest difference comes when the number of contexts is very small. This makes sense under the assumption that the COIN vectors are closer to the desired target than ADD vectors, and therefore fewer training examples are required to fit the model.

In summary, FCM can be initialized by COIN without additional cost and this combination yields more accurate results on the benchmark tasks.

5.6 Embedding Model Selection

There are various hyperparameters and model choices in training word embeddings which may

dataset	type	update	MRR.	Median Rank
enwiki	CBOW	NS	0.0614	157
	CBOW	HS	**0.1148**	102
	S-G	NS	0.0802	173
	S-G	HS	0.0711	270
WWC	CBOW	NS	0.1037	**72**
	CBOW	HS	**0.1075**	109
	S-G	NS	0.0727	104
	S-G	HS	0.0759	133

Table 6: Results of different choices from model types (CBOW or S-G) and updating methods (HS or NS) for COIN on task DefNonce. S-G is short for skip-gram.

play a role in how the context vectors relate to the word vectors in OOV prediction. In this section we consider different settings for word2vec models to compare their performance. Specifically we explore the model types (CBOW or skip-gram) and updating methods (HS or NS). We train these vectors on two corpora: a snapshot of Wikipedia from January 2019 and the subset of the WWC used by (Khodak et al., 2018) in their CRW study. The model parameters are described in Table 1 (See enwiki and WWC). We repeat the DefNonce evaluation for comparison between model selection. The results here are not directly comparable to those given in the previous sections.

Results are shown in Table 6. One observation is that CBOW generally yields better vectors for COIN than skip-gram. CBOW with HS on enwiki gives the highest MRR of all COIN models. For this model in particular, the much larger vocabulary of enwiki does not present a liability on this task. The Median Rank is also competitive among all the vector predictions. The lowest Median Rank is achieved on the WWC data with CBOW and NS.

6 Conclusion

In this work we present the COIN method for predicting word embedding vectors for OOV terms using the context vectors of context words. We advertise COIN as an inexpensive and strong baseline. We show that COIN performs close to the existing state-of-the-art techniques, while being much faster as it requires no additional training. We also show how COIN can be used along with existing techniques to give a new state-of-the-art on vector prediction tasks, and how it can be used to help downstream tasks such as sentiment analysis. Besides, we explore different model settings for learning the word vectors. The results of COIN are robust across model choices, with generally better performance from CBOW trained embeddings.

Acknowledgements: This work was supported in part by the U.S. National Science Foundation BIGDATA 1838145 and 1838147 grants, and a gift from NVIDIA Corporation.

References

Abdullah Aljebreen, Weiyi Meng, and Eduard Dragut. 2021. Segmentation of tweets with urls and its applications to sentiment analysis. *AAAI*, 35(14):12480–12488.

Sanjeev Arora, Yingyu Liang, and Tengyu Ma. 2017. A simple but tough-to-beat baseline for sentence embeddings. In *ICLR 2017*.

Dzmitry Bahdanau, Tom Bosc, Stanisław Jastrzębski, Edward Grefenstette, Pascal Vincent, and Yoshua Bengio. 2018. Learning to compute word embeddings on the fly.

Yoshua Bengio, Réjean Ducharme, Pascal Vincent, and Christian Janvin. 2003. A neural probabilistic language model. *JMLR*, 3:1137–1155.

Piotr Bojanowski, Edouard Grave, Armand Joulin, and Tomas Mikolov. 2017. Enriching word vectors with subword information. *TACL*, 5:135–146.

Hao Chen, Susan McKeever, and Sarah Jane Delany. 2019. The use of deep learning distributed representations in the identification of abusive text. *ICWSM*, 13(01):125–133.

Alexis Conneau and Douwe Kiela. 2018. SentEval: An evaluation toolkit for universal sentence representations. In *LREC*.

Sunipa Dev, Safia Hassan, and Jeff M Phillips. 2021. Closed form word embedding alignment. *Knowledge and Information Systems*, 63(3):565–588.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. In *NAACL-HLT*.

William Dolan, Chris Quirk, Chris Brockett, and Bill Dolan. 2004. Unsupervised construction of large paraphrase corpora: Exploiting massively parallel news sources.

Nicolas Garneau, Jean-Samuel Leboeuf, and Luc Lamontagne. 2019. Predicting and interpreting embeddings for out of vocabulary words in downstream tasks. *arXiv preprint arXiv:1903.00724*.

Peter Henderson, Jieru Hu, Joshua Romoff, Emma Brunskill, Dan Jurafsky, and Joelle Pineau. 2020. Towards the systematic reporting of the energy and carbon footprints of machine learning. *JMLR*, 21(248):1–43.

Aurélie Herbelot and Marco Baroni. 2017. High-risk learning: acquiring new word vectors from tiny data. In *EMNLP*, pages 304–309.

Franziska Horn. 2017. Context encoders as a simple but powerful extension of word2vec. In *Repl4NLP*, pages 10–14.

Marjan Hosseinia, Eduard Dragut, Dainis Boumber, and Arjun Mukherjee. 2021. On the usefulness of personality traits in opinion-oriented tasks. In *RANLP*, pages 547–556.

Marjan Hosseinia, Eduard Dragut, and Arjun Mukherjee. 2019. Pro/con: Neural detection of stance in argumentative opinions. In *SBP-BRiMS*, pages 21–30, Cham.

Marjan Hosseinia, Eduard Dragut, and Arjun Mukherjee. 2020. Stance prediction for contemporary issues: Data and experiments. In *SocialNLP*, Online.

Minqing Hu and Bing Liu. 2004. Mining and summarizing customer reviews. In *ACM SIGKDD*, pages 168–177.

Ziniu Hu, Ting Chen, Kai-Wei Chang, and Yizhou Sun. 2019. Few-shot representation learning for out-of-vocabulary words. *arXiv preprint arXiv:1907.00505*.

Mikhail Khodak, Nikunj Saunshi, Yingyu Liang, Tengyu Ma, Brandon Stewart, and Sanjeev Arora. 2018. A la carte embedding: Cheap but effective induction of semantic feature vectors. In *ACL 2018*, pages 12–22.

Angeliki Lazaridou, Marco Marelli, and Marco Baroni. 2017. Multimodal word meaning induction from minimal exposure to natural text. *Cognitive Science*, 41(S4):677–705.

Angeliki Lazaridou, Marco Marelli, Roberto Zamparelli, and Marco Baroni. 2013. Compositionally derived representations of morphologically complex words in distributional semantics. In *ACL 2013*, pages 1517–1526.

Quanzhi Li, Sameena Shah, Xiaomo Liu, and Armineh Nourbakhsh. 2017. Data sets: Word embeddings learned from tweets and general data. In *ICWSM*, pages 428–436.

Ismini Lourentzou, Kabir Manghnani, and ChengXiang Zhai. 2019. Adapting sequence to sequence models for text normalization in social media. In *ICWSM*, pages 335–345. AAAI Press.

Thang Luong, Richard Socher, and Christopher D. Manning. 2013. Better word representations with recursive neural networks for morphology. In *CoNLL*, pages 104–113.

Stephen Merity, Caiming Xiong, James Bradbury, and Richard Socher. 2016. Pointer sentinel mixture models. *arXiv preprint arXiv:1609.07843*.

Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013a. Efficient estimation of word representations in vector space. In *ICLR 2013*.

Tomas Mikolov, Ilya Sutskever, Kai Chen, Gregory S. Corrado, and Jeffrey Dean. 2013b. Distributed representations of words and phrases and their compositionality. In *NIPS 2013*, pages 3111–3119.

Jiaqi Mu and Pramod Viswanath. 2018. All-but-the-top: Simple and effective postprocessing for word representations. In *ICLR*.

Bo Pang and Lillian Lee. 2004. A sentimental education: Sentiment analysis using subjectivity summarization based on minimum cuts. *arXiv preprint cs/0409058*.

Jeffrey Pennington, Richard Socher, and Christopher D. Manning. 2014. Glove: Global vectors for word representation. In *EMNLP 2014*, pages 1532–1543.

Matthew E. Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer. 2018. Deep contextualized word representations. In *NAACL*.

Timo Schick and Hinrich Schütze. 2019a. Attentive mimicking: Better word embeddings by attending to informative contexts. In *NAACL 2019*.

Timo Schick and Hinrich Schütze. 2019b. Learning semantic representations for novel words: Leveraging both form and context. In *AAAI 2019*.

Andrew Schneider and Eduard Dragut. 2015. Towards debugging sentiment lexicons. In *ACL*, pages 1024–1034.

Joan Serrà, Ilias Leontiadis, Dimitris Spathis, Gianluca Stringhini, Jeremy Blackburn, and Athena Vakali. 2017. Class-based prediction errors to detect hate speech with out-of-vocabulary words. In *WOAH6*, pages 36–40.

Cyrus Shaoul and Chris Westbury. 2010. The Westbury Lab Wikipedia Corpus. `http://www.psych.ualberta.ca/~westburylab/downloads/westburylab.wikicorp.download.html`.

Richard et al. Socher. 2013. Recursive deep models for semantic compositionality over a sentiment treebank. In *EMNLP*, pages 1631–1642.

Kishore Tumarada, Yifan Zhang, Fan Yang, Eduard Dragut, Omprakash Gnawali, and Arjun Mukherjee. 2021. Opinion prediction with user fingerprinting. In *RANLP*, pages 1423–1431.

Janyce Wiebe, Theresa Wilson, and Claire Cardie. 2005. Annotating expressions of opinions and emotions in language. *LREC*, 39(2):165–210.

Fan Yang, Eduard Dragut, and Arjun Mukherjee. 2020. Predicting personal opinion on future events with fingerprints. In *COLING*, pages 1802–1807.

DynGL-SDP: Dynamic Graph Learning for Semantic Dependency Parsing

Bin Li[1,2], **Miao Gao** [1,2], **Yunlong Fan**[1,2],
Yikemaiti Sataer[1,2], **Zhiqiang Gao**[1,2,*], **Yaocheng Gui**[3]
[1]School of Computer Science and Engineering, Southeast University, Nanjing 210096, China
[2]Key Laboratory of Computer Network and Information Integration, Ministry of Education, Nanjing 210096, China
[3]School and Institute of Modern Posts, Nanjing University of Posts and Telecommunications, Nanjing 210003, China
{lib, miaogao, fanyunlong, yikmat, zqgao}@seu.edu.cn
guiyc@njupt.edu.cn

Abstract

A recent success in semantic dependency parsing shows that graph neural networks can make significant accuracy improvements, owing to its powerful ability in learning expressive graph representations. However, this work learns graph representations based on a static graph constructed by an existing parser, suffering from two drawbacks: (1) the static graph might be error-prone (e.g., noisy or incomplete), and (2) graph construction stage and graph representation learning stage are disjoint, the errors introduced in the graph construction stage cannot be corrected and might be accumulated to later stages. To address these two drawbacks, we propose a dynamic graph learning framework and apply it to semantic dependency parsing, for jointly learning graph structure and graph representations. Experimental results show that our parser outperforms the previous parsers on the SemEval-2015 Task 18 dataset in three languages (English, Chinese, and Czech).

1 Introduction

Semantic dependency parsing (SDP) represents a sentence as a directed acyclic graph (DAG), also called semantic dependency graph (SDG), to capture between-word semantic relationships that are more closely related to the meaning of the sentence. SDP has been widely applied in many downstream tasks of natural language processing (NLP), including sentiment analysis (Lin et al., 2019), abstractive summarization (Jin et al., 2020a), natural language understanding (Wu et al., 2021), etc.

Several semantic dependency parsers are presented in recent years. Their parsing mechanisms are either transition-based or graph-based. A transition-based parser generates a sequence of transition actions to incrementally build an SDG (Wang et al., 2018; Kurita and Søgaard, 2019; Lindemann et al., 2020; Fernández-González and Gómez-Rodríguez, 2020). Transition-based parsers can parse a sentence efficiently. However, they are more prone to suffer from error propagation since the transition prediction stage is sequential and greedy. Graph-based parsers overcome the shortcomings of transition-based parsers. They score each edge (or combination of them) of a possible SDG and globally search for the highest-scoring SDG (Peng et al., 2017; Dozat and Manning, 2018; Wang et al., 2019; Jia et al., 2020; He and Choi, 2020; Wang et al., 2021; Li et al., 2022). A recent success in SDP is the model of Li et al. (2022), which is a graph-based model relying on graph neural networks (GNNs). Their model utilizes an existing parser to construct an initial static graph, and then use GNNs to learn node embeddings based on the static graph to predict dependency relationships between words, as shown in Figure 1(a). This model achieves state-of-the-art performance in three languages (English, Chinese, and Czech), owing to the powerful ability of GNNs in learning expressive graph representations.

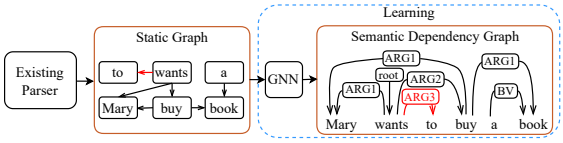

(a) Static graph-based disjoint learning for SDP

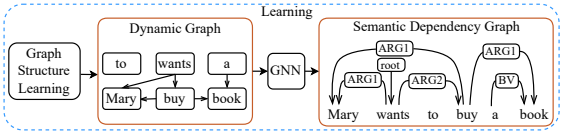

(b) Dynamic graph-based joint learning for SDP

Figure 1: Static graph-based SDP and dynamic graph-based SDP for example sentence "*Mary wants to buy a book*". The edge and dependency colored red are erroneous.

Despite the promising performance of Li et al. (2022), there are still two drawbacks in their model: (1) the initial graph constructed by the existing

*Corresponding Author

Proceedings of the 29th International Conference on Computational Linguistics, pages 3994–4004
October 12–17, 2022.

parser is static and might be error-prone (e.g., noisy or incomplete), the learned node embeddings based on the static graph may lead to performance degradation to some extent; (2) graph construction stage and graph representation learning stage are disjoint, the errors introduced in the graph construction stage cannot be corrected and might be accumulated to later stages.

To address these two drawbacks mentioned above, we propose a dynamic graph learning framework and apply it to SDP, as shown in Figure 1(b), for jointly learning graph structure and graph representations in this paper. Two GNNs variants, Graph Convolutional Network (GCN) (Kipf and Welling, 2016) and Graph Attention Network (GAT) (Veličković et al., 2018) have been investigated in DynGL-SDP. Experiments are conducted on the SemEval-2015 Task 18 dataset in three languages (English, Chinese, and Czech). Experimental results demonstrate the effectiveness of DynGL-SDP, which outperforms previous studies and achieves a new state-of-the-art performance. In addition, DynGL-SDP shows more advantages with respect to parsing speed.

Contributions The major contributions of this paper are summarized as follows: (1) we propose a dynamic graph learning framework for jointly learning graph structure and graph representations; (2) we apply the framework in SDP to propose a graph-based semantic dependency parser; (3) we conduct sufficient experiments and show that our parser achieves a new state-of-the-art result in three languages. Our code is publicly available at `https://github.com/LiBinNLP/DynGL-SDP`.

2 Related Work

In this section, the studies of SDP and dynamic graph learning will be summarized as follows.

2.1 Semantic Dependency Parsing

Several SDP models are presented in recent years. Their parsing mechanisms are either transition-based or graph-based.

A transition-based parser predicts a sequence of transition actions to incrementally build an SDG. Wang et al. (2018) presented a neural transition-based parser using *Bi-LSTM Subtraction* and *Incremental Tree-LSTM* to represent the key components of the transition system, using a variant of list-based arc-eager transition algorithm for dependency graph parsingto better capture the semantics of segments and internal sub-graph structures. Kurita and Søgaard (2019) presented a transition-based parser, using reinforcement learning algorithm to iteratively apply the syntactic dependency parser to build a DAG structure sequentially. Lindemann et al. (2020) developed a transition-based parser for *Apply-Modify* dependency parsing. They introduced the *stack-pointer* model to predict transitions. Fernández-González and Gómez-Rodríguez (2020) presented a transition-based parser, using *pointer network* to choose a transition among *ROOT*, *Attach-p*, and *Shift*.

Transition-based parsers can parse a sentence efficiently using a linear or quadratic number of transitions. However, they are more prone to suffer from error propagation since the transition prediction stage is sequential and greedy, an erroneous action can affect future predictions.

Graph-based parsers overcome the shortcomings of transition-based parsers, therefore they draw more attention. A graph-based parser scores each edge (or combination of them) of a possible SDG and searches for the highest-scoring SDG. Peng et al. (2017) developed a graph-based parser which explored two multitask learning approaches with a parameterization and factorization that implicitly to model the relationship between multiple formalisms. Dozat and Manning (2018) presented a biaffine attention-based parser, which extended the syntactic parser of Dozat et al. (2017) to train on and generate an SDG. Wang et al. (2019) extended the parser of Dozat and Manning (2018) and added the second-order information for scoring each dependency edge. Either mean-field variational inference or loopy belief propagation is utilized for approximate decoding. Jia et al. (2020) presented a semi-supervised model based on conditional random field autoencoder to learn a dependency graph. He and Choi (2020) significantly improved the performance by introducing contextual string embeddings (called Flair) in the basis of Dozat and Manning (2018). Wang et al. (2021) utilized reinforcement learning algorithm to find better concatenations of embeddings, and then fed it into the parser of Dozat and Manning (2018).

Recently, Li et al. (2022) presented a GNN-based parser. They used an existing parser to construct an initial SDG, and then utilized GNNs to learn node embeddings to predict dependencies,

achieving state-of-the-art performance. However, the initial SDG constructed by the existing parser is static and might be error-prone (e.g., noisy or incomplete), this may leads to error accumulation and performance degradation.

2.2 Dynamic Graph Learning

Graph learning is the process of learning the representations of a graph, GNNs are the most prominent approaches for graph learning. GNNs take in the original feature and adjacency matrix, and output node embeddings as graph representations (Hamilton et al., 2017; Kipf and Welling, 2016; Veličković et al., 2018). Due to the powerful ability in learning graph representations, GNNs have been applied to various downstream tasks, including node prediction (Hamilton et al., 2017), link prediction (Teru et al., 2020), and graph classification (Ying et al., 2018).

Despite GNNs' powerful ability in graph learning, unfortunately, they can only be used when graph-structured data is available. Many NLP tasks may only have sequential data, there is no graph structure available. To address this limitation, several dynamic graph learning frameworks are presented, for jointly learning graph structure and graph representations (Chen et al., 2020; Jin et al., 2020b; Fu and He, 2021).

Inspired by the ideas of these frameworks, we propose a dynamic graph learning framework, and then apply it to SDP, to generate an SDG from word sequence rather than an initial static graph.

3 DynGL-SDP

DynGL-SDP is a graph-based parser using the dynamic graph learning framework. An overview of DynGL-SDP is shown as Figure 2. Given sentence s with n words $[w_1, w_2, \ldots, w_n]$, there are four stages to parse it as an SDG: (1) contextualized representation learning—a bidirectional long short-term memory network (BiLSTM) is used to learn the contextualized representation of each word; (2) graph structure learning—a graph structure learning module is used to learn the adjacency matrix of a potential SDG; (3) graph representation learning—the contextualized representations and the learned adjacency matrix will be fed into the GNNs, to learn expressive node embeddings; (4) dependency relationship learning—the concatenation of node embeddings and contextualized representations are fed into biaffine attention-based parser, to learn dependency relationships between words.

3.1 Contextualized Representation Learning

We concatenate word and feature embeddings, and feed them into a BiLSTM to obtain contextualized representations.

$$x_i = e_i^{(word)} \oplus e_i^{(feat)} \tag{1}$$

$$c_i = BiLSTM(x_i) \tag{2}$$

where x_i is the concatenation ($\oplus$) of the word and feature embeddings of word w_i, c_i is the contextualized representation of w_i.

Word Embedding 100-dimensional word embeddings from GloVe (Pennington et al., 2014) are used for English; 300-dimensional word embeddings from fasttext (Grave et al., 2018) are used for Chinese and Czech.

Feature Embedding Four types of feature embeddings are used, their dimensions (denoted as d) are equal: (1) Part-of-speech (POS) tag: POS tag embedding $E^{(pos)}$ is randomly generated, $E^{(pos)} \in R^{n \times d}$, where n is the number of POS tags; (2) Lemma: lemma embedding $E^{(lemma)}$ is also randomly generated. $E^{(lemma)} \in R^{l \times d}$, where l is the number of lemmas; (3) Character: character embedding is generated using Char-LSTM (Kim et al., 2016) that convolved over three-character embeddings at each time step; (4) BERT: BERT embedding (Kenton and Toutanova, 2019) is extracted from the pretrained BERT$_{\text{base}}$ model.

3.2 Graph Structure Learning

The purpose of the graph structure learning module is to learn the adjacency matrix A of a potential SDG. Two multi-layer perceptrons (MLP) are used to capture head and dependent representations of each word, as Equation 3 and 4:

$$h_i^{(adj-head)} = MLP^{(adj-head)}(c_i) \tag{3}$$

$$h_i^{(adj-dep)} = MLP^{(adj-dep)}(c_i) \tag{4}$$

We can then use biaffine classifier (as Equation 5), to compute the score of a possible edge between w_i and w_j, as Equation 6:

$$Biaff(x_1, x_2) = x_1^T U x_2 + W(x_1 \oplus x_2) + b \tag{5}$$

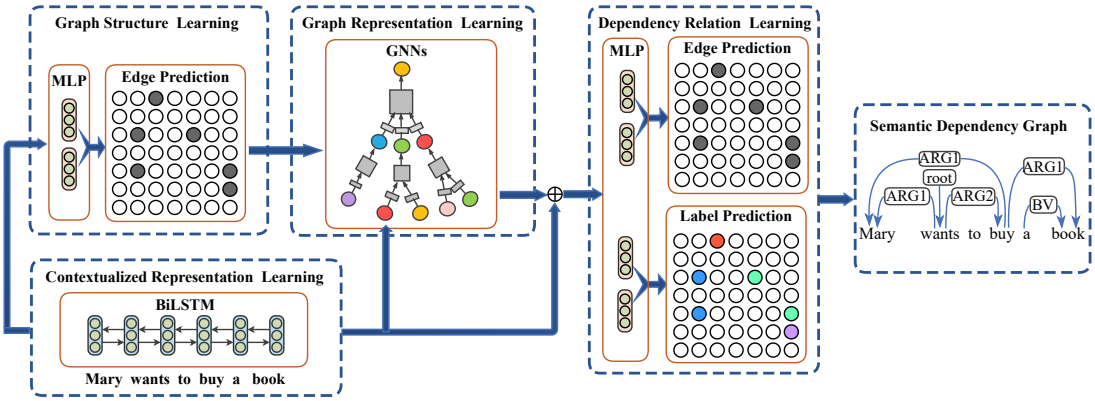

Figure 2: Overall architecture of the proposed DynGL-SDP.

$$s_{i,j}^{(adj)} = Biaff^{(adj)}(h_i^{(adj-dep)}, h_j^{(adj-head)}) \tag{6}$$

A directed edge from w_i to w_j exists when A_{ij} is 1. A_{ij} in A is computed as Equation 7:

$$\hat{A}_{i,j} = \begin{cases} 1, & s_{i,j}^{(adj)} > 0 \\ 0, & s_{i,j}^{(adj)} \leq 0 \end{cases} \tag{7}$$

3.3 Graph Representation Learning

The graph representation learning module utilizes GNNs to learn the each word's representation (i.e. node embedding) that contains graph structure information. K-layer GNNs are employed, which take in the contextualized representations and the learned adjacency matrix A and output the embedding matrix of final layer as node embeddings R. $R^{(k)}$ in k^{th}-layer is computed as Equation 8:

$$R^{(k)} = GNNLayer^{(k-1)}(R^{(k-1)}, \hat{A}) \tag{8}$$

When *GNNLayer* is implemented in *GCN*, the representation of node i in k^{th} layer $r_i^{(k)}$ is computed as Equation 9:

$$r_i^{(k)} = \sigma\left(W \sum_{j \in N(i)} r_j^{(k-1)} + B r_i^{(k-1)}\right) \tag{9}$$

where W and B are parameter matrices; $N(i)$ are neighbors of node i; σ is active function; $r_i^{(0)} = c_i$.

When *GNNLayer* is implemented in *GAT*, $r_i^{(k)}$ is computed as Equation 10:

$$r_i^{(k)} = \sigma\left(W \sum_{j \in N(i)} a_{ij}^{(k-1)} r_j^{(k-1)} + B r_i^{(k-1)}\right) \tag{10}$$

where $a_{ij}^{(k-1)}$ is attention coefficient of node i to its neighbour j at $(k-1)^{th}$ layer.

3.4 Dependency Relationship Learning

The dependency relationship learning module follows the biaffine attention-based parser (Dozat and Manning, 2018), which has two components: edge prediction and label prediction. For each word w_i, the node embedding r_i and the contextualized representation c_i are concatenated to represent it, as shown in Equation 11. For each of the two components, we use MLP to split the final word representation z_i into two parts—a head representation, and a dependent representation, as shown in Equation 12 – 15:

$$z_i = r_i \oplus c_i \tag{11}$$

$$h_i^{(edge-head)} = MLP^{(edge-head)}(z_i) \tag{12}$$

$$h_i^{(label-head)} = MLP^{(label-head)}(z_i) \tag{13}$$

$$h_i^{(edge-dep)} = MLP^{(edge-dep)}(z_i) \tag{14}$$

$$h_i^{(label-dep)} = MLP^{(label-dep)}(z_i) \tag{15}$$

Two biaffine classifiers are used to predict edges and labels, as Equation 16 and 17 :

$$s_{i,j}^{(edge)} = Biaff^{(edge)}(h_i^{(edge-dep)}, h_j^{(edge-head)}) \tag{16}$$

$$s_{i,j}^{(label)} = Biaff^{(label)}(h_i^{(label-dep)}, h_j^{(label-head)}) \quad (17)$$

where $s_{i,j}^{(edge)}$ and $s_{i,j}^{(label)}$ are scores of the edge and label between the word w_i and w_j. U, W, and b are learned parameters.

For edge prediction component, U will be $(d \times 1 \times d)$-dimensional, so that $s_{i,j}^{(edge)}$ will be a scalar. An edge between w_i and w_j exists where $s_{i,j}$ is positive. For label prediction component, U will be $(d \times c \times d)$-dimensional, where c is the number of labels, so that $s_{i,j}^{(label)}$ is a vector that represents the probability distribution of each label. The most probable label will be assigned to the edge between w_i and w_j.

$$\hat{y}_{i,j}^{(edge)} = \{s_{i,j}^{(edge)} > 0\} \quad (18)$$

$$\hat{y}_{i,j}^{(label)} = \arg\max s_{i,j}^{(label)} \quad (19)$$

3.5 Learning

We can train the system by summing the losses from the three modules, back propagating error to the parser. Cross entropy function is used as the loss function, which is computed as Equation 20:

$$CE(p, q) = -\sum_x p(x) \log q(x) \quad (20)$$

We define the loss function of graph structure learning module (as Equation 21), edge prediction module (as Equation 22) and label prediction module (as Equation 23):

$$\mathcal{L}^{(adj)}(\theta_1) = CE(\hat{A}_{i,j}, A_{i,j}) \quad (21)$$

$$\mathcal{L}^{(edge)}(\theta_2) = CE(\hat{y}_{i,j}^{(edge)}, y_{i,j}^{(edge)}) \quad (22)$$

$$\mathcal{L}^{(label)}(\theta_3) = CE(\hat{y}_{i,j}^{(label)}, y_{i,j}^{(label)}) \quad (23)$$

where θ_1, θ_2, and θ_3 are the parameters of three modules.

Then the Adaptive Moment Estimation (Adam) (Kingma and Ba, 2015) method is used to optimize the summed loss function $\mathcal{L}$:

$$\mathcal{L} = \alpha\mathcal{L}^{(adj)} + \beta\mathcal{L}^{(edge)} + (1-\alpha-\beta)\mathcal{L}^{(label)} \quad (24)$$

where α and β are two tunable interpolation constants, where $(\alpha + \beta) \in (0, 1)$.

4 Experiments

4.1 Dataset and Evaluation Metrics

We conduct experiments on the SemEval-2015 Task 18 dataset, which covers three languages (English, Chinese, and Czech) and contains three different formalisms (DELPH-IN MRS (DM) (Flickinger et al., 2012), Predicate-Argument Structure (PAS) (Miyao and Tsujii, 2004), and Prague Semantic Dependencies (PSD) (Hajic et al., 2012)). Three formalisms (DM, PAS, and PSD) are all available for English; only PAS formalism is available for Chinese; only PSD formalism is available for Czech.

Dataset Split The dataset split for three languages is the same as Li et al. (2022), which is shown in Appendix A.1.

Evaluation Metric Labeled F-measure score (LF1) (including ROOT edges) is used as the metric to evaluate our parser's performance on the ID and OOD test sets for each formalism as well as the macro-average over the three of them.

4.2 Hyperparameters

The hyperparameter configuration for our final system is given in Appendix A.2. Following Wang et al. (2019), Adam method is used for optimizing our model, annealing the learning rate by 0.5 for every 10,000 steps, and switched the optimizer to AMSGrad (Reddi et al., 2019) after 5,000 steps without improvement. We train the model for 100,000 iterations with batch sizes of 6,000 tokens and terminated training early after 10,000 iterations with no improvement on the development set.

4.3 Baseline Approaches

We compare DynGL-SDP with previous state-of-the-art approaches. We group them into three groups: transition-based models, graph-based models, and hybrid models.

Transition-based models Turku is from Kanerva et al. (2015). WCGL (Wang et al., 2018) is a neural transition-based model. SemPointer (Fernández-González and Gómez-Rodríguez, 2020) is a transition-based model using Pointer Network. Lindemann et al. (2019) and Lindemann et al. (2020) are compositional semantic parser for SDP and abstract meaning representation.

Models	English							
	DM		PAS		PSD		Avg	
	ID	OOD	ID	OOD	ID	OOD	ID	OOD
Peking (2015)	89.1	81.8	91.3	87.2	75.7	73.3	85.3	80.8
Lisbon (2015)	88.2	81.8	90.9	86.9	76.4	74.8	85.2	81.2
PTS17 (2017): Basic	89.4	84.5	92.2	88.3	77.6	75.3	86.4	82.7
PTS17 (2017): Basic	90.4	85.3	92.7	89.0	78.5	76.4	87.2	83.6
WCGL (2017): Basic	90.3	84.9	91.7	87.6	78.6	75.9	86.9	82.8
D&M (2018): Basic	91.4	86.9	93.9	90.8	79.1	77.5	88.1	85.0
MF (2019): Basic	93.0	88.4	94.3	91.5	80.9	78.9	89.4	86.3
LBP (2019): Basic	92.9	88.4	94.3	91.5	81.0	78.8	89.4	86.2
Lindemann et al. (2019): Basic	91.2	85.7	92.2	88.0	78.9	76.2	87.4	83.3
SemPointer (2020): Basic	92.5	87.7	94.2	91.0	81.0	78.7	89.2	85.8
GNNSDP(GCN) (2022): Basic	93.3	88.0	94.8	91.1	85.6	83.6	91.2	87.6
GNNSDP(GAT) (2022): Basic	93.0	87.9	94.8	91.6	85.4	83.3	91.1	87.6
DynGL-SDP(GCN) Basic	**93.7**	**89.3**	**94.9**	**91.7**	**85.9**	**84.1**	**91.5**	**88.5**
DynGL-SDP(GAT) Basic	**93.8**	**89.2**	**95.1**	**92.0**	**85.9**	**83.8**	**91.6**	**88.3**
D&M (2018): +Char+Lemma	93.7	88.9	93.9	90.6	81.0	79.4	89.5	86.3
MF (2019): +Char+Lemma	94.0	89.7	94.1	91.3	81.4	79.6	89.8	86.9
LBP (2019): +Char+Lemma	93.9	89.5	94.2	91.3	81.4	79.5	89.8	86.8
Jia et al. (2020): +Lemma	93.6	89.1	-	-	-	-	-	-
SemPointer (2020): +Char+Lemma	93.9	89.6	94.2	91.2	81.8	79.8	90.0	86.9
GNNSDP(GCN) (2022): +Char+Lemma	94.2	**90.1**	94.9	91.4	86.4	84.9	91.8	88.8
GNNSDP(GAT) (2022): +Char+Lemma	94.4	89.9	**95.0**	91.8	86.2	84.6	91.9	88.8
DynGL-SDP(GCN) +Char+Lemma	**95.0**	**90.1**	**95.0**	**92.0**	**86.6**	**85.0**	**92.2**	**89.0**
DynGL-SDP(GAT) +Char+Lemma	**94.9**	**90.5**	**95.3**	**92.1**	**86.7**	**85.0**	**92.3**	**89.2**
Lindemann et al. (2019): +BERT$_{large}$	94.1	90.5	94.7	92.8	82.1	81.6	90.3	88.3
Lindemann et al. (2020): +BERT$_{large}$	93.9	90.4	94.7	92.7	81.9	81.6	90.2	88.2
SemPointer (2020): +Char+Lemma+BERT$_{base}$	94.4	91.0	95.1	93.4	82.6	82.0	90.7	88.8
He et al. (2020): +Char+Lemma+BERT$_{base}$+Flair	94.6	90.8	96.1	94.4	86.8	79.5	92.5	88.2
ACE-Fine-tune (2021) +AutoConcat	95.6	92.6	95.8	**94.6**	83.8	83.4	91.7	90.2
GNNSDP(GCN) (2022): +Char+Lemma+BERT$_{base}$	95.1	91.1	95.7	93.2	**87.7**	**87.3**	92.8	90.5
GNNSDP(GAT) (2022): +Char+Lemma+BERT$_{base}$	95.3	91.9	96.0	94.3	87.0	86.7	92.8	91.0
DynGL-SDP(GCN) +Char+Lemma+BERT$_{base}$	**95.8**	**92.7**	**96.2**	94.2	**87.8**	87.0	**93.3**	**91.3**
DynGL-SDP(GAT) +Char+Lemma+BERT$_{base}$	**95.9**	**92.7**	**96.2**	94.3	**87.7**	87.2	**93.3**	**91.4**

Table 1: Comparison of labeled F1 scores achieved by our model and previous parsers on English dataset. Jia et al. (2020) only reports the full-supervised result on DM formalism. The F1 scores of Baseline and our model are averaged over 5 runs. ID denotes the in-domain (Wall Street Journal Corpus) test set and OOD denotes the out-of-domain (Brown Corpus) test set. +Char, +Lemma, +BERT, +Flair, and +AutoConcat mean augmenting the token embeddings with character-level, lemma embeddings, BERT embeddings, Flair embeddings, and automated concatenation of 11 types of pretrained embeddings.

Graph-based models Lisbon is from Almeida and Martins (2015). PTS17 (Peng et al., 2017) is a multitask learning based parser across three formalisms. D&M (Dozat and Manning, 2018) is a biaffine attention-based parser. MF and LBP (Wang et al., 2019) are a second-order model using mean field variational inference or loopy belief propagation. Jia et al. (2020) is a semi-supervised parser, only the full-supervised result on DM formalism is reported in their paper. He and Choi (2020) uses not only BERT but also contextual string embeddings (called Flair). ACE-Fine-tune (Wang et al., 2021) adds automated concatenation of 11 types of pretrained embeddings to the biaffine attention-based parser. GNNSDP (Li et al., 2022) is a GNN-based parser, which is the previous state-of-the-art parser.

Hybrid models Peking is a hybrid model from Du et al. (2015). Riga is from Barzdins et al. (2015).

4.4 Main Results

To perform a fair comparison, we group SDP models in three blocks according to the embeddings provided to the models: (1) just basic pre-trained word embeddings and POS tag embeddings (Basic), (2) character and pre-trained lemma embeddings augmentation (+Char+Lemma) and (3) pretrained BERT embeddings augmentation (+Char+Lemma+BERT).

4.4.1 Results on English

Table 1 shows the comparison of DynGL-SDP and previous studies on the SemEval-2015 Task 18 English dataset. From the result, we have the following observations:

- DynGL-SDP implemented with two GNN variants outperforms all existing parsers on three formalisms of English dataset in three embedding settings.
- Compared to the previous best one (GNNSDP (2022)) in each embedding setting, the best performing DynGL-SDP makes 0.4%, 0.4%, and 0.5% averaged LF1 improvements on in-domain test sets, 0.9%, 0.4%, and 0.4% averaged LF1 improvements on out-of-domain test sets.
- The performances of most parsers have been generally improved when the token embeddings are augmented with more feature embeddings.
- The performances of two DynGL-SDP variants implemented with GCN and GAT are relatively close.
- We note that ACE-Fine-tune (2021) performs better than DynGL-SDP in out-of-domain test set of PAS formalism. A reasonable explanation is that 11 types of pretrained embeddings are used in their model, improving the model's generalization ability.

4.4.2 Results on Chinese and Czech

Table 2 shows the comparison of DynGL-SDP and previous studies on SemEval-2015 Task 18 Chinese and Czech test sets. From the result, we have observed that:

- DynGL-SDP outperforms the previous parsers on Chinese and Czech. The best performing DynGL-SDP makes 0.33% averaged LF1 improvement on Chinese in-domain test set in three embedding settings, 0.28% averaged LF1 improvement on Czech in-domain and out-of-domain test sets in three embedding settings.
- The LF1 scores of two DynGL-SDP variants implemented with GCN and GAT are relatively close on Chinese and Czech.

Models	Chinese PAS ID	Czech PSD ID	Czech PSD OOD
Turku(2015)	79.6	75.3	63.7
Riga(2015)	82.5	75.3	61.3
Peking(2015)	83.4	78.5	64.4
Lisbon(2015)	82.0	79.3	63.5
D&M(2018): Basic	87.4	86.9	77.8
GNNSDP(GCN)(2022): Basic	88.3	88.2	**79.1**
GNNSDP(GAT)(2022): Basic	88.0	87.8	78.9
DynGL-SDP(GCN): Basic	**88.8**	**88.7**	78.9
DynGL-SDP(GAT): Basic	**88.9**	**88.9**	79.0
D&M (2018): +Char+Lemma	87.8	87.6	78.9
GNNSDP(GCN) (2022): +Char+Lemma	**88.5**	88.8	80.2
GNNSDP(GAT) (2022): +Char+Lemma	88.3	88.9	80.2
DynGL-SDP(GCN): +Char+Lemma	**88.5**	**89.7**	**80.3**
DynGL-SDP(GAT): +Char+Lemma	88.3	**90.0**	80.0
GNNSDP(GCN) (2022): +Char+Lemma+BERT	90.1	89.6	**80.7**
GNNSDP(GAT) (2022): +Char+Lemma+BERT	90.4	89.3	80.4
DynGL-SDP(GCN): +Char+Lemma+BERT	**90.8**	**90.1**	80.4
DynGL-SDP(GAT): +Char+Lemma+BERT	**90.8**	**90.1**	80.4

Table 2: Comparison of labeled F1 scores achieved by DynGL-SDP and previous studies on Chinese and Czech datasets. Only the PAS formalism and in-domain (ID) test set are available for Chinese, the PSD formalism, in-domain and out-of-domain (OOD) test sets are available for Czech.

- On the Chinese dataset, the LF1 score of DynGL-SDP has degraded on the contrary when the token embeddings are augmented with character-level and lemma embeddings. The reason is that the character and lemma are not available for Chinese.

In summary, DynGL-SDP outperforms the previous parsers in three languages and three semantic dependency formalisms. Outstanding performances of DynGL-SDP demonstrate that the proposed dynamic graph learning framework is able to learn the expressive graph representations without depending on an initial static graph.

5 Analysis

5.1 Performance on Different Sentence Lengths

Here we want to investigate the performances of DynGL-SDP (ours) and GNNSDP (the previous best one) on different sentence lengths. We split the ID and OOD test sets of DM formalism into 6 and 7 groups (one group with 10 tokens) and evaluate DynGL-SDP and GNNSDP on them. The GNN module in these two parsers is implemented with GAT. The results in different groups are shown in

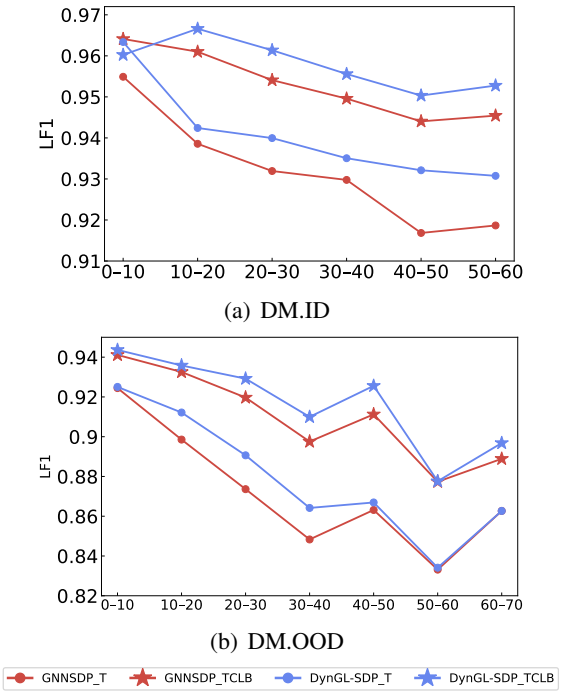

Figure 3: LF1 scores of different sentence lengths in DM formalism on English dataset. *_T represents that only the POS tag embedding is used. *_TCLB represents that the POS tag, character-level, lemma, and BERT embeddings are used.

Figure 3.

From the result, we can see that DynGL-SDP outperforms GNNSDP in different groups and two embedding settings. Furthermore, the performances of both parsers degrade as sentence length gets longer, highly suggesting that parsing longer sentences is still a challenge.

5.2 Parsing Speed

Not only accuracy but also parsing speed determine whether a parser can be applied to downstream tasks. Therefore we compare DynGL-SDP and GNNSDP with respect to parsing speed on an Nvidia GeForce RTX2080Ti server.

To avoid the influence of preprocessing stage, the annotated tokens, POS tags, and lemmas in the dataset are directly used without preprocessing. The result is shown in table 3.

From the result, we can see that:

- DynGL-SDP performs better than GNNSDP with respect to parsing speed in three embedding settings.
- The parsing speed of two parsers slows down when more features are added.

Models	EN DM	CHS PAS	CZ PSD
GNNSDP(GCN):Basic	1153	1032	997
DynGL-SDP(GCN):Basic	**1974**	**1922**	**1828**
GNNSDP(GCN):+Char+Lemma	821	819	775
DynGL-SDP(GCN):+Char+Lemma	**1551**	**1543**	**1411**
GNNSDP(GCN):+Char+Lemma+BERT	346	297	276
DynGL-SDP(GCN):+Char+Lemma+BERT	**677**	**559**	**554**

Table 3: Parsing speed (sentences/second) of DynGL-SDP (ours) and GNNSDP (the previous best one) on English (EN), Chinese (CHS) and Czech (CZ). Parsing speed of each parser is averaged over 5 runs.

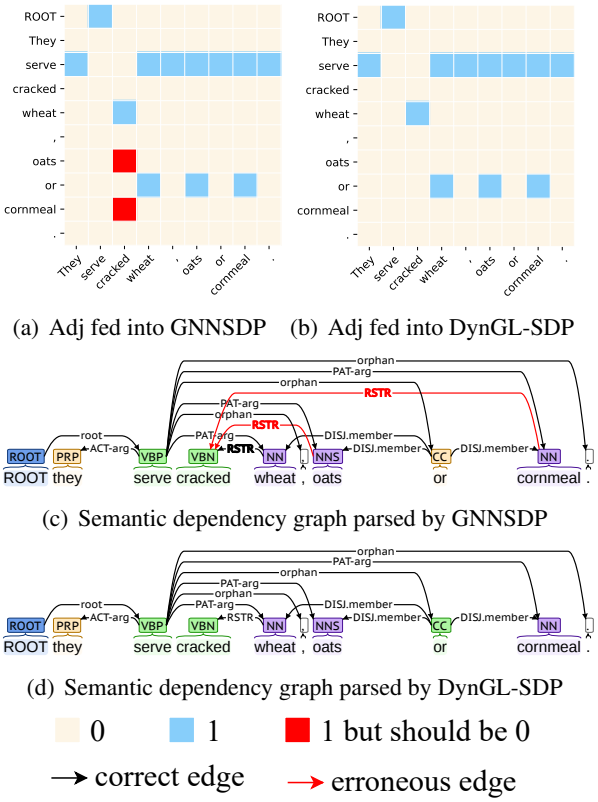

Figure 4: 4(a) and 4(b) are two adjacency matrices (Adj) fed into two parsers. The words in the left are head words, words in the bottom are dependent words. 4(c) and 4(d) are parsing results of two parsers.

5.3 Case Study

We provide a parsing example to show why DynGL-SDP can outperform GNNSDP using dynamic graph learning. Figure 4(a) and Figure 4(b) represent the adjacency matrices fed into GNNSDP and DynGL-SDP. Figure 4(c) and Figure 4(d) are parsing results of GNNSDP and DynGL-SDP for the English sentence "*They serve cracked wheat, oats or cornmeal.*" (sent_id = 40504062, in OOD test set of PSD formalism). The two parsers are implemented with GCN and trained in the basic

embedding setting.

From Figure 4(a), we can see that there are two erroneous values (red square) in this adjacency matrix, indicating that the graph structure input into GNNSDP is noisy. Using learned node embeddings based on the noisy graph leads to two erroneous dependent edges in SDG parsed by GNNSDP (red edge labeled RSTR).

Benefiting from the dynamic graph learning framework, the learned graph structure input into DynGL-SDP is correct. Therefore DynGL-SDP produces a correct SDG.

6 Conclusions

In this paper, we propose a dynamic graph learning framework and apply it in semantic dependency parsing. Experimental results show that our model achieves a new state-of-the-art performance on the SemEval-2015 Task 18 dataset in three languages (English, Chinese, and Czech). The outstanding performance of our model demonstrates that the proposed dynamic graph learning framework is able to learn the expressive graph representations without depending on an initial static graph.

Acknowledgments

This research was sponsored by the Foundation of Jiangsu Provincial Double-Innovation Doctor Program (Grant No. JSSCBS20210507) and NUPTSF (Grant No. NY220176 and NY221106).

References

Mariana SC Almeida and André FT Martins. 2015. Lisbon: Evaluating turbo semantic parser on multiple languages and out-of-domain data. In *Proceedings of the 9th International Workshop on Semantic Evaluation (SemEval 2015)*, pages 970–973.

Guntis Barzdins, Peteris Paikens, and Didzis Gosko. 2015. Riga: from framenet to semantic frames with c6. 0 rules. In *Proceedings of the 9th International Workshop on Semantic Evaluation (SemEval 2015)*, pages 960–964.

Yu Chen, Lingfei Wu, and Mohammed Zaki. 2020. Iterative deep graph learning for graph neural networks: Better and robust node embeddings. *Advances in Neural Information Processing Systems*, 33.

Timothy Dozat and Christopher D Manning. 2018. Simpler but more accurate semantic dependency parsing. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 484–490.

Timothy Dozat, Peng Qi, and Christopher D Manning. 2017. Stanford's graph-based neural dependency parser at the conll 2017 shared task. In *Proceedings of the CoNLL 2017 Shared Task: Multilingual Parsing from Raw Text to Universal Dependencies*, pages 20–30.

Yantao Du, Fan Zhang, Xun Zhang, Weiwei Sun, and Xiaojun Wan. 2015. Peking: Building semantic dependency graphs with a hybrid parser. In *Proceedings of the 9th international workshop on semantic evaluation (SemEval 2015)*, pages 927–931.

Daniel Fernández-González and Carlos Gómez-Rodríguez. 2020. Transition-based semantic dependency parsing with pointer networks. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7035–7046.

Dan Flickinger, Yi Zhang, and Valia Kordoni. 2012. Deepbank. a dynamically annotated treebank of the wall street journal. In *Proceedings of the 11th International Workshop on Treebanks and Linguistic Theories*, pages 85–96.

Dongqi Fu and Jingrui He. 2021. Sdg: A simplified and dynamic graph neural network. In *Proceedings of the 44th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 2273–2277.

Edouard Grave, Piotr Bojanowski, Prakhar Gupta, Armand Joulin, and Tomas Mikolov. 2018. Learning word vectors for 157 languages. In *Proceedings of the International Conference on Language Resources and Evaluation (LREC 2018)*.

Jan Hajic, Eva Hajicová, Jarmila Panevová, Petr Sgall, Ondřej Bojar, Silvie Cinková, Eva Fucíková, Marie Mikulová, Petr Pajas, Jan Popelka, et al. 2012. Announcing prague czech-english dependency treebank 2.0. In *Proceedings of the Eighth International Conference on Language Resources and Evaluation (LREC'12)*, pages 3153–3160.

William L Hamilton, Rex Ying, and Jure Leskovec. 2017. Inductive representation learning on large graphs. In *Proceedings of the 31st International Conference on Neural Information Processing Systems*, pages 1025–1035.

Han He and Jinho Choi. 2020. Establishing strong baselines for the new decade: Sequence tagging, syntactic and semantic parsing with bert. In *The Thirty-Third International Flairs Conference*.

Zixia Jia, Youmi Ma, Jiong Cai, and Kewei Tu. 2020. Semi-supervised semantic dependency parsing using crf autoencoders. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 6795–6805.

Hanqi Jin, Tianming Wang, and Xiaojun Wan. 2020a. Semsum: Semantic dependency guided neural abstractive summarization. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pages 8026–8033.

Wei Jin, Yao Ma, Xiaorui Liu, Xianfeng Tang, Suhang Wang, and Jiliang Tang. 2020b. Graph structure learning for robust graph neural networks. In *2020 ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*.

Jenna Kanerva, Juhani Luotolahti, and Filip Ginter. 2015. Turku: Semantic dependency parsing as a sequence classification. In *Proceedings of the 9th International Workshop on Semantic Evaluation (SemEval 2015)*, pages 965–969.

Jacob Devlin Ming-Wei Chang Kenton and Lee Kristina Toutanova. 2019. Bert: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of Annual Conference of the North American Chapter of the Association for Computational Linguistics:Human Language Technologies (NAACL-HLT)*, pages 4171–4186.

Yoon Kim, Yacine Jernite, David Sontag, and Alexander M Rush. 2016. Character-aware neural language models. In *Thirtieth AAAI conference on artificial intelligence*.

Diederik P Kingma and Jimmy Ba. 2015. Adam: A method for stochastic optimization. In *Proceedings of the 3rd International Conference on Learning Representations (ICLR 2015)*.

Thomas N Kipf and Max Welling. 2016. Semi-supervised classification with graph convolutional networks. *arXiv preprint arXiv:1609.02907*.

Shuhei Kurita and Anders Søgaard. 2019. Multi-task semantic dependency parsing with policy gradient for learning easy-first strategies. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 2420–2430.

Bin Li, Yunlong Fan, Yikemaiti Sataer, Zhiqiang Gao, and Yaocheng Gui. 2022. Improving semantic dependency parsing with higher-order information encoded by graph neural networks. *Applied Sciences*, 12(8).

Peiqin Lin, Meng Yang, and Jianhuang Lai. 2019. Deep mask memory network with semantic dependency and context moment for aspect level sentiment classification. In *Proceedings of the 2019 International Joint Conferences on Artificial Intelligence*, pages 5088–5094.

Matthias Lindemann, Jonas Groschwitz, and Alexander Koller. 2019. Compositional semantic parsing across graphbanks. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 4576–4585.

Matthias Lindemann, Jonas Groschwitz, and Alexander Koller. 2020. Fast semantic parsing with well-typedness guarantees. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 3929–3951.

Yusuke Miyao and Jun'ichi Tsujii. 2004. Deep linguistic analysis for the accurate identification of predicate-argument relations. In *Proceedings of the 20th International Conference on Computational Linguistics (COLING 2004)*, pages 1392–1398.

Hao Peng, Sam Thomson, and Noah A Smith. 2017. Deep multitask learning for semantic dependency parsing. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 2037–2048.

Jeffrey Pennington, Richard Socher, and Christopher D Manning. 2014. Glove: Global vectors for word representation. In *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*, pages 1532–1543.

Sashank J Reddi, Satyen Kale, and Sanjiv Kumar. 2019. On the convergence of adam and beyond. *arXiv preprint arXiv:1904.09237*.

Komal Teru, Etienne Denis, and Will Hamilton. 2020. Inductive relation prediction by subgraph reasoning. In *International Conference on Machine Learning*, pages 9448–9457. PMLR.

Petar Veličković, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Liò, and Yoshua Bengio. 2018. Graph attention networks. In *International Conference on Learning Representations*.

Xinyu Wang, Jingxian Huang, and Kewei Tu. 2019. Second-order semantic dependency parsing with end-to-end neural networks. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 4609–4618.

Xinyu Wang, Yong Jiang, Nguyen Bach, Tao Wang, Zhongqiang Huang, Fei Huang, and Kewei Tu. 2021. Automated concatenation of embeddings for structured prediction. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 2643–2660.

Yuxuan Wang, Wanxiang Che, Jiang Guo, and Ting Liu. 2018. A neural transition-based approach for semantic dependency graph parsing. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 32.

Zhaofeng Wu, Hao Peng, and Noah A Smith. 2021. Infusing finetuning with semantic dependencies. *Transactions of the Association for Computational Linguistics*, 9:226–242.

Zhitao Ying, Jiaxuan You, Christopher Morris, Xiang Ren, Will Hamilton, and Jure Leskovec. 2018. Hierarchical graph representation learning with differentiable pooling. *Advances in neural information processing systems*, 31.

A Appendix

A.1 Dataset Split

The dataset split is shown in Table 4.

Language	Train	Dev	Test	
			ID	OOD
English	33,964	1,692	1,410	1,849
Chinese	25,336	3,000	8,976	-
Czech	39,057	3,000	1,670	316

Table 4: The number of sentences contained in each divided dataset of three languages. Only the in-domain test set is available for Chinese; the in-domain and out-of-domain test sets are available for English and Czech.

- For English, 33,964 sentences from Sections 00-19 of the Wall Street Journal corpus as training data, 1,692 sentences from Section 20 as development data, 1,410 sentences from Section 21 as in-domain (ID) test data, and 1,849 sentences sampled from the Brown Corpus as the out-of-domain (OOD) test data.

- For Chinese, the top 3,000 sentences as the development data, the remaining 25,336 sentences as the training data, and 8,976 sentences as ID test data.

- For Czech, the top 3,000 sentences as the development data, the remaining 39,057 sentences as the training data, 1,670 sentences as the ID test data, and 316 sentences as the OOD test data.

A.2 Hyperparameter Values

The hyperparameter configuration for our final system is given in Table 5. 100-dimensional pretrained GloVe embeddings are used for English, in which the token "*unk*" represents the out-of-vocabulary tokens. 300-dimensional pretrained fasttext embeddings are used for Chinese and Czech, in which the token "*UNK*" represents the out-of-vocabulary tokens. Word embeddings of each language will be linearly transformed to be 125-dimensional. Only words or lemmas that occurred 7 times or more will be included in the word and lemma embedding matrix.

Layer	Hyper-parameter	Value
Word Embedding	English	100
	Chinese/Czech	300
Feature Embedding	POS/Lemma	100
	Char/BERT	100
LSTM	layers	3
	hidden size	400
	dropout	0.33
GNN	GCN/GAT layers	3
	GCN/GAT hidden	600
	GCN/GAT dropout	0.33
MLP	adj-head/dep hidden	600
	edge-head/dep hidden	600
	label-head/dep hidden	600
Trainer	optimizer	Adam
	learning rate	$1e^{-2}$
	Adam (β_1, β_2)	0.95
	decay rate	0.75
	decay step length	5000
	Loss(α, β)	0.2, 0.2

Table 5: Final hyperparameter configuration.

Knowledge Is Flat: A Seq2Seq Generative Framework for Various Knowledge Graph Completion

Chen Chen[1,*], **Yufei Wang**[2,*], **Bing Li**[3] and **Kwok-Yan Lam**[1†]
Nanyang Technological University, Singapore[1]
Macquarie University, Sydney, Australia[2]
Centre for Frontier AI Research (CFAR), A*STAR, Singapore[3]
{S190009,kwokyan.lam}@ntu.edu.sg,
yufei.wang@students.mq.edu.au
li_bing@ihpc.a-star.edu.sg

Abstract

Knowledge Graph Completion (KGC) has been recently extended to multiple knowledge graph (KG) structures, initiating new research directions, e.g. static KGC, temporal KGC and few-shot KGC (Ji et al., 2022). Previous works often design KGC models closely coupled with specific graph structures, which inevitably results in two drawbacks: 1) structure-specific KGC models are mutually incompatible; 2) existing KGC methods are not adaptable to emerging KGs. In this paper, we propose **KG-S2S**, a Seq2Seq generative framework that could tackle different verbalizable graph structures by unifying the representation of KG facts into "flat" text, regardless of their original form. To remedy the KG structure information loss from the "flat" text, we further improve the input representations of entities and relations, and the inference algorithm in **KG-S2S**. Experiments on five benchmarks show that **KG-S2S** outperforms many competitive baselines, setting new state-of-the-art performance. Finally, we analyze **KG-S2S**'s ability on the different relations and the *Non-entity Generations* [1].

1 Introduction

Knowledge graph completion (KGC) has been a fundamental task to discover unobserved facts from various knowledge graph (KG) structures, including static KGC (SKGC), temporal KGC (TKGC) and few-shot KGC (FKGC) (Ji et al., 2022). As shown in Figure 1, TKGC (in orange) contains temporal facts with timestamps, while FKGC (in green) predicts the facts with relations that only have limited or zero training instances.

Typically, the solutions for KGC are graph-based, i.e., treating entities and relations as *nodes* and *linkages*. The training and inference of SKGC models rely on various *transitional relations* over graph paths (Trouillon et al., 2016; Dettmers et al., 2018; Vashishth et al., 2020). TKGC and FKGC methods are further integrated with non-trivial components or learning paradigms to handle the extra temporal information or training requirements. Concretely, TKGC models (Dasgupta et al., 2018; Goel et al., 2020; Lacroix et al., 2020) either construct temporal-specific sub-KG or add additional temporal embeddings into existing SKGC methods. FKGC models apply the additional training scheme (e.g., meta-learning) between the frequent relations and the infrequent ones to the SKGC models (Xiong et al., 2018; Niu et al., 2021). Such a methodological discrepancy leads to a great maintenance cost and being inadaptable to emerging knowledge queries, ingestion, and presents. Naturally, a research question has been raised: *Can we adapt the different forms of KG facts and solve these KGC tasks in a unified framework?*

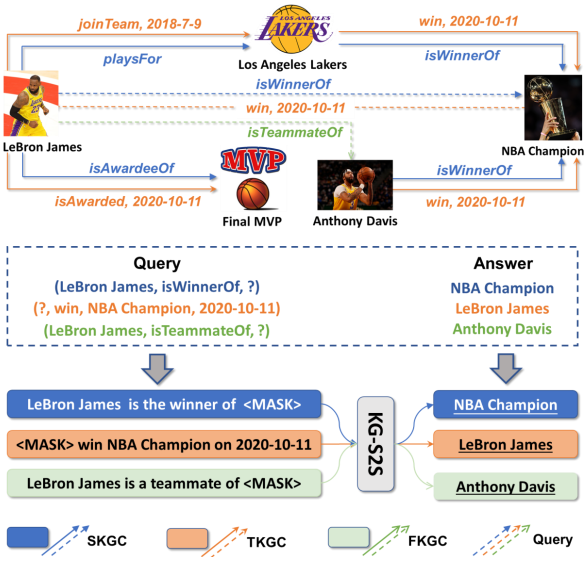

Figure 1: Running examples of Static (SKGC), Temporal (TKGC) and Few-shot (FKGC) Knowledge Graph Completion tasks. Our proposed **KG-S2S** is an unified Seq2Seq framework adaptable to all of these tasks.

[*] First two authors contribute equally.
[†] Corresponding author
[1] Our source code is available at https://github.com/chenchens190009/KG-S2S

Proceedings of the 29th International Conference on Computational Linguistics, pages 4005–4017
October 12–17, 2022.

Recently, Seq2Seq Pre-trained Language Models (PLM) have shown state-of-the-art performances and high technical homogeneity when dealing with different NLP tasks. Albeit having heterogeneous input and output, the Seq2Seq PLMs covert those tasks into "text-to-text" format, taking the text as inputs and producing another text as outputs (Raffel et al., 2020; Xie et al., 2022a). In addition, PLMs have embedded massive real-world knowledge from the pre-training (Petroni et al., 2019), which is potentially beneficial for the KGC tasks, especially in the data-sparsity scenarios.

Inspired by this, we propose **KG-S2S**, a simple yet effective Seq2Seq PLM framework adaptable to various KG structures. Given a KG query, **KG-S2S** directly generates the target entity text using the common PLM fine-tuning practices. Firstly, to remedy the KG structure information loss caused by the naïve "text-to-text" format, we improve **KG-S2S** via 1) the input representations of entities and relations using *Entity Description*, *Soft Prompt* and *Seq2Seq Dropout*; 2) the constrained inference algorithm empowered by the *Prefix Constraints*; Secondly, we treat all the KG elements (i.e., entity, relation and timestamp) as "flat" text (Figure 1) which enables **KG-S2S** to **i)** handle various *verbalizable* knowledge graph structures; **ii)** generate non-entity text and find novel entities for KGs. We make several improvements on the preliminary attempts of concurrent works (Saxena et al., 2022; Xie et al., 2022b) using Seq2Seq for KGC. Our model adds special treatments to input entity/relation textual representation. This helps to better capture subtle yet key tokens and facilitate the ability to ingest other graph structures.

We conduct experiments on **WN18RR**, **FB15K-237** and **FB15K-237N** for SKGC, **ICEWS14** for TKGC and **NELL-One** for FKGC. **KG-S2S** outperforms several competitive baseline models, including graph-based and PLM-based models, and sets new state-of-the-art performance on all three settings. We conduct ablation studies to show the effectiveness of the proposed components, compare **KG-S2S** with graph-based KGC models at the relation level and finally showcase the *Non-entity Generation* from **KG-S2S** to present its potential in producing novel knowledge triples.

2 Related work

KGC has been studied in the Static, Temporal and Few-shot settings. Previous works often focus on a single setting, while **KG-S2S** fits all three settings without any architecture modifications.

Static KGC Early SKGC models assign trainable embeddings to each entity and relations (Bordes et al., 2013; Sun et al., 2019). A score function is proposed to evaluate the scores of triples with these embedding. These models learn structural information of a knowledge graph, regardless of the textual information of the entities and relations. Recently, Yao et al. (2019); Wang et al. (2021a); Xie et al. (2022b); Saxena et al. (2022) proposed to encode entity and relation textual knowledge into the model by using PLMs. Instead of calculating scores from embeddings, they train PLMs to produce plausibility scores for KG text representation.

Temporal KGC Many TKGC models incorporate additional time-specific parameters upon existing KGC methods. Leblay and Chekol (2018), based on Bordes et al. (2013), represents each timestamp with independent embeddings. Dasgupta et al. (2018) resembles Wang et al. (2014), regarding timestamps as hyperplanes for entities to project. Lacroix et al. (2020) considers the score of each triple as canonical decomposition of order 4 tensors in complex domain. Goel et al. (2020) suggests learning dynamic embeddings for entity and relations, transforming part of the embedding with sinusoidal activation of learned frequencies. Han et al. (2021) proposes a systematic framework to improve existing temporal embedding models.

Few-shot KGC For one-shot learning on relations, Xiong et al. (2018) attempts to seek a matching metric that can be used to discover similar triples given one reference triple. Chen et al. (2019) discovers two kinds of relation-specific meta-information: relation meta, and gradient meta. It uses meta-learning methods to transfer meta-information to low-resource relations. With the help of textual information and PLMs, Wang et al. (2021a) outperforms other few-shot baseline models on the zero-shot relations.

3 Proposed Method

This section first formulates *Knowledge Graph Completion* tasks in Sec. 3.1, then discusses our proposed **KG-S2S** method in Sec. 3.2, 3.3 and 3.4. Figure 2 shows the overview of **KG-S2S**.

Figure 2: The overview of **KG-S2S**. Given the query (LeBron James, is the winner of, ?, $*$), we represent "LeBron James" by joining entity name and description. *Soft Prompt distinguishes* similar relation name and *disentangles* relation-specific information. *Seq2Seq Dropout* randomly masks input words to avoid over-fitting. In the inference, *Prefix Constraints* (Pre. Con.) forces the decoding algorithms to *only* generate valid entity text in the decoder.

3.1 Knowledge Graph Completion

A *Knowledge Graph* (KG) $(\mathcal{E}, \mathcal{R}, \mathcal{T})$ includes an entity set $\mathcal{E}$, a relation set $\mathcal{R}$ and a tuple list $\mathcal{T} = [(h, r, t, m)_1, \cdots, (h, r, t, m)_n]$ where $h, t \in \mathcal{E}$ is head and tail entity, $r \in \mathcal{R}$ is the tuple relation and m is the KG meta-information. *Knowledge Graph Completion* (KGC) predicts the missing entities for the queries $(?, r, t, m)$ or $(h, r, ?, m)$.

The meta-information m denotes different form of contents in different KG settings. As shown in Figure 2, m is represented as null in SKGC, timestamps (e.g., "Jun-19-2014") in TKGC and typing (e.g., Player-Championship) in the KGs providing typing information. Using text representation, KGs with different structures can be converted into an unified format.

3.2 A Seq2Seq Framework for KGC

A Seq2Seq Framework, including an *encoder* and a *decoder*, can be viewed as:

$$P(Y|X) = \prod_{t=1}^{m} P(y_t|X, Y_{<t}) \qquad (1)$$

where X is the input sequence to the Seq2Seq *encoder*, Y is the auto-regressively generated output sequence (i.e., from left to right) and y_0 is the special Begin-of-Sequence Symbol. To apply this Seq2Seq Framework to KGC, given query $(?, r, t, m)$ or $(h, r, ?, m)$ and corresponding ground-truth answer gt, we first encode r, t and m into text. We represent "?" with "<mask>" at the corresponding position to distinguish between $(?, r, t, m)$ and $(h, r, ?, m)$. We then concatenate the text together into X and train **KG-S2S** to generate gt as output sequence Y. The **KG-S2S** training is straightforward: unlike StAR (Wang et al., 2021a) which applied composite objective over the encoder-only PLM, we follow the common practices in fine-tuning Seq2Seq PLMs (i.e., Cross-Entropy Loss), directly training **KG-S2S** with positive examples (negative sampling trick is unnecessary to **KG-S2S**). However, this architecture remains two main challenges: *i)* How to *effectively* represent the query in the **KG-S2S** encoder? *ii)* How to *accurately* generate entity text as the answer to the query? Sec. 3.3 and Sec 3.4 answer the above two questions, respectively.

3.3 Entity & Relation Representation

Encoding query $(?, r, t, m)$ and $(h, r, ?, m)$ into "flat" text allows **KG-S2S** to handle various KGs. However, the "flat" text could introduce KG structure loss. To remedy this issue, we further improve **KG-S2S** using the following components.

Entity Description Intuitively, one could represent an entity using its name text (e.g., Anthony Davis) in either compositional or non-compositional form (Li et al., 2018a). However, as **KG-S2S** is initialized from the PLM weights, some specific types of entities (e.g., person, locations) may refer to multiple real-world entities in the large-scale PLM training corpus, introducing noisy ambiguity to **KG-S2S**. To avoid this risk, we additionally introduce entity descriptions to enrich the context information of entities. For example, the textual description about "Lebron James" could be "is an American NBA star". Previous research (Zuo et al., 2018; Lovelace et al., 2021) have shown the utility of the descriptions when integrated with traditional graph-based KGC models. Likewise, we add entity descriptions for both queries and ground-truth answers. At the encoder side, we concatenate entity names and descriptions as the entity representation. At the decoder side, we *train* **KG-S2S** to jointly predict entity names and entity descriptions under the cross-entropy loss. We find using entity descriptions on both sides of **KG-S2S** is beneficial.

KG Soft Prompt In traditional graph-based KGC models (e.g., TransE), KG entities and relations are represented with separated embeddings, while **KG-S2S** represents entities and relations using the shared Seq2Seq PLM parameters. As a consequence, the KG knowledge/patterns regarding similar surface relations or entities (e.g., "film costume design by" and "film production design by") could be mixed together. To tackle this issue and inspired by the recent *Soft Prompt* (Lester et al., 2021), which is a set of trainable embeddings directly fed into the Seq2Seq PLM input, we propose to add additional trainable prompt embeddings for specific entities and relations into X. The separated parameter space could potentially *disentangle* the general KG and element-specific knowledge for **KG-S2S**. However, as entities are equipped with descriptions and *Entity Soft Prompt* introduces a large number of parameters, we only apply the *Soft Prompt* to relations. Specifically, as shown in Figure 2, similar to a recent BERT-based KGC model (Lv et al., 2022), we insert the *Relation Soft Prompt* embeddings $P_{e1}, P_{e2}, P_{r1}, P_{r2} \in \mathbb{R}^{|\mathcal{R}| \times d}$, where d is the **KG-S2S** hidden size, before and after the textual entity and relation name.

Seq2Seq Dropout In our preliminary experiments, we find that **KG-S2S** often learns fast (measured by validation MRR) in the early stage of the model training. We hypothesize that, unlike other NLG tasks where different instances have little textual overlapping, the entity descriptions remain unchanged in different training queries in KGC, which could easily lead to over-fitting. We attempt to increase the original *encoder dropout* for **KG-S2S** training, however, it has little impact on the final performance. Therefore, we impose a more strict *Seq2Seq Dropout* where we randomly select and mask $p\%$ of the input tokens in X when calculating the encoder self-attention module and decoder cross-attention module. Note that the *Relation Soft Prompt* and the "<mask>" token are excluded from this selection process. Compared with the original *encoder dropout*, Seq2Seq dropout takes effect at both encoder and decoder sides. This introduces more diversity to the input query text, thus, better capability of preventing over-fitting.

3.4 KGC Inference

The traditional KGC models g answer a query $(?, r, t, m)$ by first finding the score $g(x, r, t, m)$ $\forall x \in \mathcal{E}$ and then ranking all entities based on $g(x, r, t, m)$. Naturally, at the inference stage, **KG-S2S** could compute a score for every $x \in \mathcal{E}$. However, this could be computationally expensive because $|\mathcal{E}|$ could be very large (e.g., $|\mathcal{E}|$ is 68,544 in **NELL-One**). Instead, in **KG-S2S**, its encoder takes X as input and then the **KG-S2S** decoder generates the text of entity predictions that are mapped into specific entity ids. These generated entities are further ranked based on their corresponding log cross-entropy loss. We assign -∞ for all entities not generated in the decoding stage.

Algorithm 1 Next Candidates (NC): Given Entity Prefix Trie $\mathcal{T}$, Query-GT Prefix Trie Mapping $\mathcal{D}$, query q and the generated tokens Gen; return candidate tokens.

procedure NC($\mathcal{T}, \mathcal{D}, q, Gen$)
  $\mathcal{T}_q \leftarrow \mathcal{D}.get(q)$
  $cand \leftarrow \mathcal{T}.next(pre = Gen)$
  $rm \leftarrow \mathcal{T}_q.next(pre = Gen)$
  $cand \leftarrow cand.remove(rm)$
  return $cand$

Decoding Methods Different from general text generation tasks where only one optimal output sequence is required, in KGC, given a query $(?, r, t, m)$ or $(h, r, ?, m)$, there could be multiple valid entities. To generate K valid entity candidates, we deploy the standard beam search algorithm with beam width K because it naturally produces different entity text in each beam with high likelihood. In contrast, random sampling often provides low-quality answers due to its randomness in decoding and the outputs of diverse beam search are distorted due to its diversity encouragement term.

Prefix Constraints The flexible auto-regressive generation may produce entities that do not exist in $\mathcal{E}$, which could reduce the number of valid entity candidates in the decoding. To avoid this scenario, we propose *Prefix Constraints* to control the **KG-S2S** decoder to generate valid tokens given prefix sequences p. For example, given $\mathcal{E}$ = {"Grammy Award for Best Rock Song", "Grammy Award for Best Music Video"} and p = [Grammy, Award, for, Best], the *Prefix Constraints* only allow "Rock" and "Music" to be generated in the next step. To enable effective decoding, we propose to use Trie (Cormen et al., 2009) to extract appropriate next tokens. As suggested in Algorithm 1, given the generated prefix, we first extract all possible

tokens using the Entity Prefix Trie $\mathcal{T}$ and then remove the entities that are the ground-truth to the query q in the training data using the Query-GT Prefix Trie $\mathcal{T}_q$.

4 Experiment

In this section, we evaluate **KG-S2S** against competitive baselines in the following KGC datasets: **WN18RR** (Dettmers et al., 2018) (SKGC), **FB15K-237** (Toutanova and Chen, 2015) (SKGC) **FB15K-237N** (Lv et al., 2022) (SKGC) and **ICEWS14** (García-Durán et al., 2018) (TKGC) and **NELL-One** (Xiong et al., 2018) (FKGC).

4.1 Experimental Settings

Dataset **WN18RR** and **FB15K-237** are improved version of WN18 and FB15k (Bordes et al., 2013) respectively, where all inverse relations are removed to avoid data leakage. **FB15K-237N** further removes **FB15K-237**'s concatenated relations caused by Freebase mediator nodes (Akrami et al., 2020) to avoid Cartesian production relation issue. **ICEWS14** refers to 2014 political facts from the Integrated Crisis Early Warning System database (Boschee et al., 2015). **NELL-One** is a few-shot KGC dataset derived from NELL (Carlson et al., 2010). Following Wang et al. (2021a), we reformat **NELL-One** so that the dev/test relations never appear in the train set. More details can be found in Appendix A.

Implementation details We initialize **KG-S2S** using the T5-base model (Raffel et al., 2020), and optimize **KG-S2S** with Adam (Kingma and Ba, 2015). We use T5 default settings in our experiments for all benchmarks and follow the *filtered setting* proposed in Bordes et al. (2013) to evaluate our model. More implementation details and optimal hyperparameters can be found in Appendix B.

Evaluation Protocol We remove the duplicated entities from the output. In the non-constrained decoding method, we further remove non-entity generations. The performance of our model is reported on the standard KGC metrics: Mean Reciprocal Rank (MRR), and Hits@1,3,10 (Hits@1,5,10 in **NELL-One** to follow previous works). For each test triple (h, r, t, m), we rank all entities for the query $(h, r, ?, m)$ and $(?, r, t, m)$. We then aggregate the ranking for ground-truth entity and report the mean reciprocal rank (MRR) and the proportion of ground-truth entities ranked in the top n (H@n). To handle the equal score scenarios, we use the RANDOM mode proposed in Sun et al. (2020) to determine the rank of entities. Model is selected by MRR value on valid set.

4.2 Experimental results

Static KGC We compare our results with various graph-based and PLM-based methods on the SKGC settings. Experimental results are summarized in Table 1. On **WN18RR** and **FB15K-237**, **KG-S2S** achieves state-of-the-art or competitive performance. In the comparison of PLM-based methods, **KG-S2S** outperforms previous work by a substantial margin. Specifically, we see 13% (from 0.508 to 0.574) relative MRR improvement on **WN18RR**, and 16% (from 0.296 to 0.336) on **FB15K-237**. Compared with graph-based methods, **KG-S2S** consistently obtains performance gain on **WN18RR**, though maintaining a modest result on **FB15K-237**.

According to Akrami et al. (2020); Lv et al. (2022), **FB15K-237** contains many oversimplified unrealistic cartesian product relations (CPR), which improperly improves the model accuracy. For instance, the multiary fact "average low temperature in Tokyo is 34 degrees Fahrenheit in January" has been decomposed into multiple CPR facts (Tokyo, climate./month, January) and (Tokyo, climate./average_min_temp, 34), which are obviously unrealistic and semantically meaningless. We note that RotatE achieves higher overall performance than **KG-S2S** on **FB15K-237**.

However, after breaking down the performance on CPRs and non-CPRs in Table 2, we surprisingly find that our proposed **KG-S2S** has distinct advantages on non-CPR (MRR 0.363 vs. 0.338). That is, leading performance of RotatE is due to the facts with CPR, while **KG-S2S** has demonstrated its advantages in realistic relations (i.e., non-CPRs). Methodologically, RotatE is a typical graph-based model, while KG-S2S regards KGs as plain text with structure-aware components. Graph-based models are good at predicting simple structure yet inferior in absorbing KGs text. This explains why RotatE performs better on FB15k-237 dataset which is rich in cartesian product relations (CPRs, simple synthesized yet less textually meaningful relations), while worse on non-CPR datasets like FB15K-237N. This further motivates us to compare **KG-S2S** with other KGC methods on **FB15K-237N** which only has facts with non-CPR.

	WN18RR				FB15K-237				FB15K-237N			
	MRR	H@1	H@3	H@10	MRR	H@1	H@3	H@10	MRR	H@1	H@3	H@10
Graph-Based Methods												
TransE (Bordes et al., 2013)	.243	.043	.441	.532	.279	.198	.376	.441	.255	.152	.301	.459
DistMult (Yang et al., 2015)	.444	.412	.470	.504	.281	.199	.301	.446	.209	.143	.234	.330
ComplEx (Trouillon et al., 2016)	.449	.409	.469	.530	.278	.194	.297	.450	.249	.180	.276	.380
ConvE (Dettmers et al., 2018)	.456	.419	.470	.531	.312	.225	.341	.497	.273	.192	.305	.429
RotatE (Sun et al., 2019)	.476	.428	.492	.571	.338	.241	.375	.533	.279	.177	.320	.481
CompGCN (Vashishth et al., 2020)	.479	.443	.494	.546	.355	.264	.390	.535	.316	.231	.349	.480
PLM-Based Methods												
KG-BERT (Yao et al., 2019)	.216	.041	.302	.524	-	-	-	.420	.203	.139	.201	.403
MTL-KGC (Kim et al., 2020)	.331	.203	.383	.597	.267	.172	.298	.458	.241	.160	.284	.430
StAR (Wang et al., 2021a)	.401	.243	.491	**.709**	.296	.205	.322	.482	-	-	-	-
PKGC (Lv et al., 2022)	-	-	-	-	-	-	-	-	.307	.232	.328	.471
GenKGC (Xie et al., 2022b)	-	.287	.403	.535	-	.192	.355	.439	-	-	-	-
KGT5 (Saxena et al., 2022)	.508	.487	-	.544	.276	.210	-	.414	-	-	-	-
KG-S2S (Ours)	**.574**	**.531**	**.595**	.661	**.336**	**.257**	**.373**	**.498**	**.353**	**.282**	**.385**	**.495**

Table 1: Results of static KGC. **WN18RR** and **FB15K-237** results are taken from Wang et al. (2021a). **FB15K-237N** results are taken from (Lv et al., 2022). The uncovered results of graph-based methods are obtained through hyperparameter tuning with LibKGE (Broscheit et al., 2020) and PLM-based methods through official implementations. The best PLM-based method results are in bold and the second best results are in underline.

	relations	MRR	H@1	H@3	H@10
RotatE	CPR	.337	.232	.374	.552
	non-CPR	.340	.254	.376	.504
	all	.338	.241	.375	.533
KG-S2S	CPR	.318	.234	.355	.493
	non-CPR	.363	.292	.398	.504
	all	.336	.257	.373	.498

Table 2: Evaluation of cartesian product relations (CPRs) and non-cartesian product relations (non-CPRs) on **FB15K-237**

	MRR	H@1	H@3	H@10
Graph-Based Methods				
TTransE (Leblay and Chekol, 2018)	.255	.074	-	.601
HyTE (Dasgupta et al., 2018)	.297	.108	.416	.655
ATiSE (Xu et al., 2019)	.550	.436	.629	.750
DE-SimplE (Goel et al., 2020)	.526	.418	.592	.725
Tero (Xu et al., 2020)	.562	.468	.621	.732
TComplEx (Lacroix et al., 2020)	.560	.470	.610	.730
TNTComplEx (Lacroix et al., 2020)	.560	.460	.610	.740
T+TransE (Han et al., 2021)	.553	.437	.627	**.765**
T+SimplE (Han et al., 2021)	.539	.439	.594	.730
PLM-Based Methods				
KG-S2S (Ours)	**.595**	**.516**	**.642**	.737

Table 3: Results of temporal KGC on **ICEWS14**. All the results are from original papers.

As shown in Table 1, **KG-S2S** obtains the best results compared with graph-based and PLM-based baselines at all metrics. In particular, **KG-S2S** achieves an absolute Hit@1 increase of 5.0% over second best method Wang et al. (2021a).

The overall SKGC results confirms that, by taking advantage of entity and relation textual representation, **KG-S2S** is capable of capturing more accurate semantics of KG facts, and employ them for inference.

Temporal KGC To evaluate **KG-S2S**'s ability of handling additional meta-information in KG, we conduct the experiment on the TKGC benchmark **ICEWS14**. The results are shown in Table 3. Our proposed **KG-S2S** obtains a new state-of-the-art result on MRR and Hit@1,3 while achieving comparative performance on Hit@10. This result confirms that **KG-S2S** can learn additional temporal meta-information from pure textual form. We observe that our result on Hit@10 is lower than several existing methods. This could be explained by the low quality of entities in **ICEWS14**, which only includes the "sector" and "country" of the entities. These entity descriptions are much less informative than the ones in the SKGC benchmark. We believe that the performance of **KG-S2S** could be further improved when more informative entity descriptions are available.

Few-shot KGC Finally, we verify **KG-S2S**'s ability in few-shot learning in the **NELL-One** benchmark, as shown in Table 4. Following Wang et al. (2021a), we conduct the evaluation under zero-shot setting (i.e, evaluation relations never appear in the training set). Surprisingly, **KG-S2S** is able to achieve superior performance than all the variations of previous graph-based models, which transfer knowledge from the training data to the evaluation relations (i.e., one-shot and five-shot meta learning). In addition, compared with the PLM-based StAR model, **KG-S2S** also obtains higher performance with considerable margins in

	N-Shot	MRR	H@1	H@5	H@10
Graph-Based Methods					
GMatching$_{\text{ComplEx}}$ ♣	Five	.20	.14	.26	.31
MetaR♡	Five	.26	.17	.35	.44
GMatching$_{\text{TransE}}$ ♣	One	.17	.12	.21	.26
GMatching$_{\text{DistMult}}$ ♣	One	.17	.11	.22	.30
GMatching$_{\text{ComplEx}}$ ♣	One	.19	.12	.26	.31
MetaR♡	One	.25	.17	.34	.40
MTransH△	One	**.31**	.21	**.41**	.48
PLM-Based Methods					
StAR♠	Zero	.26	.17	.35	.45
KG-S2S (Ours)	Zero	**.31**	**.22**	**.41**	**.49**

Table 4: Results of few-shot KGC on NELL-One. △ (Niu et al., 2021). ♣ (Xiong et al., 2018), ♡ (Chen et al., 2019) and ♠ (Wang et al., 2021a).

terms of all the metrics. In particular, Hit@1 performance is boosted from 0.17 to 0.22, around 29% relative improvement. This remarkable performance gain could own to the following aspects: 1) the prior knowledge contained in PLM; 2) **KG-S2S**'s capability to transfer the knowledge from the training relations to the unseen ones.

	PW	Description		Soft Prompt		S2S. Drop	MRR	H@10
		SRC	TGT	REL	ENT			
Baseline	✓	-	-	-	-	-	.280	.416
	✓	✓	-	-	-	-	.326	.453
	✓	✓	✓	-	-	-	.350	.478
	✓	✓	✓	✓	-	-	.350	.486
	✓	✓	✓	-	✓	-	.338	.468
KGT5	-	✓	✓	-	-	-	.226	.335
	-	✓	✓	✓	-	✓	.233	.341
KG-S2S	✓	✓	✓	✓	-	✓	**.353**	**.495**

Table 5: Ablation for the **KG-S2S** Input Components on **FB15K-237N**. PW denotes pretrained weight. SRC and TGT denote source and target description. REL and ENT denote relation-specific and entity-specific soft prompts. S2S.Drop denotes Seq2Seq dropout.

4.3 Ablation Study

In this section, we conduct ablation studies to show the contributions of each of our proposed components. Table 5 shows the impact of input components and Figure 3 shows the ablation study of decoding components in **KG-S2S**.

Source and target description The descriptions of entities enrich their context information and resolve the ambiguity issue. As shown in Table 5, adding source and target description can separately improve MRR (i.e., from 0.280 to 0.350) and Hit@10 (i.e., from 0.416 to 0.478). This suggests that to achieve optimal performance, it is important to inject entity descriptions at **KG-S2S**'s encoder and decoder, *simultaneously*.

Soft Prompt Soft prompt allows **KG-S2S** to recognize entities and relations as atomic concepts. Adding relation Soft Prompt successfully boosts Hit@10 from 0.478 to 0.486. However, the Entity Soft Prompt has a negative effect for **KG-S2S**, degrading MRR and Hit@10 by 0.12 (from 0.350 to 0.338) and 0.1 (0.478 to 0.468), respectively. We argue this phenomenon occurs for at least two reasons: 1) The entity descriptions have already enriched the entity context information and consequently made entities distinguishable; 2) Entity Soft Prompt introduces massive amounts of embeddings, which may weaken **KG-S2S**'s ability to learn from natural language.

Seq2Seq Dropout Seq2Seq dropout applies a random masking mechanism on the encoder input mask. It is observed that Seq2Seq dropout is able to deliver consistent improvement on both MRR and Hit@10. This result practically justifies the effectiveness of this implementation. We believe the advance is derived from the diversified input data generated by Seq2Seq dropout, which helps **KG-S2S** to avoid potential over-fitting risk.

Campared with KGT5 KGT5 is trained on a random initialized Seq2Seq structure to fully adapt KG training data. However, learnt from large pretraining corpus, pretrained weights contains rich linguistic knowledge and simply dropping them may weaken the model's ability of ingesting nature language. The large performance gap between KGT5 and **KG-S2S** indicates pretrained weights is critical for KGC models with Seq2Seq backbones .

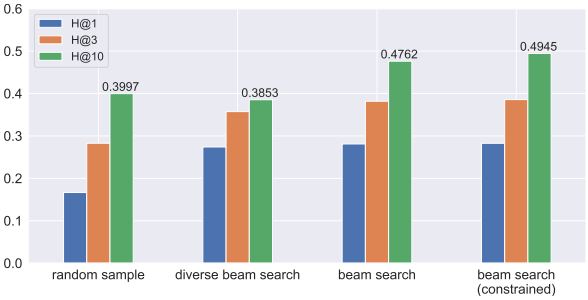

Figure 3: Comparison for decoding methods on **FB15K-237N**.

Impact of Decoding Methods In Figure 3, we investigate the impact of decoding methods, including random sampling, diverse beam search, beam search and *Prefix Constraints* beam search (de-

scribed in Sec. 3.4) in **KG-S2S**. The performances of random sampling and diverse beam search are much worse than the standard beam search algorithm. This is mostly because random selection in sampling and diversity encouragement terms in diverse beam search negatively affect the quality of generated entity text. Whilst standard beam search always keeps and derives the candidates with the highest beam score from **KG-S2S**. We find that applying our *Prefix Constraints* to the beam search algorithm further improves the **KG-S2S** performance (i.e., 0.02 Hit@10 improvement). *Prefix Constraints* control **KG-S2S** to only generate valid entity text with little computation overhead.

4.4 Discussion

Comparison with previous SOTA PLM-based methods From Table 1 and Table 4, **KG-S2S** outperforms previous SOTA encoder-only StAR methods on MRR and Hit@1,3. We argue two advantages contribute to this result: 1) Pretraining / finetuning consistency. StAR employs composite training objectives at the entity level, while its backbones (i.e. BERT (Devlin et al., 2019) and RoBERTa (Liu et al., 2019)) are trained with token-level cross-entropy loss. This mismatch may weaken the representation ability of PLM. In contrast, **KG-S2S** follows the common PLM finetuning practices, allowing better knowledge transfer from PLM. 2) Information interaction. StAR uses a two-branch Siamese architecture (Chopra et al., 2005) to encode the query text and answer text as two separated vectors, and calculates their dot-production as the score for ranking. Instead of compressing the them separately, **KG-S2S** interacts query and answer in the cross-attention module of **KG-S2S** decoder, auto-regressively. With more textual exposure and interaction, **KG-S2S** decoder trends to predict more accurate entities.

Relations Analysis Figure 4 shows the top-3 and bottom-3 relations regarding the MRR difference between **KG-S2S** and RotatE. The top-3 relations are *(sports team) location*, *(location) contains* and *(Netflix) genre*, which refer to the real-world knowledge. The possible reason is such knowledge has already been obtained from the pre-training corpus by the Seq2Seq PLM. In contrast, the bottom-3 relations are *(film) production company*, *(film) executive produced by* and *(film) produced by*, which, surprisingly, are all relevant to the film industry. This could be because these relations are all linked

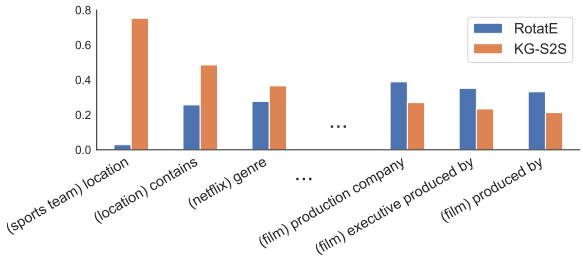

Figure 4: The top-3 (left) / bottom-3 (right) relations regarding the MRR difference between **KG-S2S** and RotatE on the **FB15K-237N** benchmark. To maintain stable results, we select the relations with at least 0.5% facts, and sort them by the MRR difference.

to the person and company names that may have multiple references (i.e., different people and companies could share the same names) in the PLM pre-training corpus. In addition, we find that some of the relations are semantically overlapping. For example, the **FB15K-237N** includes both relation *(film) executive produced by* and *(film) produced by*. After being trained with the fact *(Hulk, (film) executive produced by, Stan Lee)*, **KG-S2S** generates *Stan Lee* as the top-1 candidate for the query *(Hulk, (film) produced by, ?)*. However, the ground-truth entity set doesn't include *Stan Lee*. This scenario has no effect on the traditional graph-based KGC models because they do not access the text at all. Similar cases also occur between *(film) written by* and *(film) story by*, *(people) profession* and *(people) specialization of*. This issue is caused by the fact that previous KGC benchmarks i) are not fully verified by experts; ii) are based on the closed-world assumption (CWA) (Keet, 2013). We leave KGC benchmarks improvement as future work.

Queries	Prediction	GT
(RoboCop, (film) genre, ?)	Superhero film	Thriller
(Amber Riley, profession, ?)	Vocalist	Actor-GB
(? (location) contains, Israel)	Greater Middle East	Eurasia
(?, ethnicity, M. Night Shyamalan)	Malayalam people	Indian American

Table 6: Case study for Non-entity generations. GT stands for ground-truth answer.

Non-entity Generation Without *Prefix Constraints* module, **KG-S2S** can generate non-entity text. As shown in Table 6, some of the non-entity generations are also meaningful answers to the query. In the first example, Superhero film and Thriller are both semantically correct answer. In the second one, Amber Riley is actually considered as an actor and a vocalist by the public. In addition, the third and fourth examples show that **KG-S2S**

can derive more fine-grained answers. For example, *Israel* is specifically located in the *Greater Middle East* and *M. Night Shyamalan* is an *Indian American*, born in a Malayalam-speaking Indian city. These newly generated entities could be potentially applied to improve the KGC model performance via a data augmentation procedure (Wang et al., 2022). The expert knowledge to determine the plausibility of non-entity generations is given by the corresponding entries from Wikipedia, e.g. the Wikipedia profile for Amber Riley [2].

The Effect of Beam Width Beam width determines the number of generations for each query, thus it has potentially significant impact on the **KG-S2S** performance. In Figure 5, we study how beam width affects the final performance by evaluating **KG-S2S** under different beam width. In general, **KG-S2S** achieves higher MRR as the beam width increases, whilst the performance gain becomes flat after 40 beams (red bar). As inference time goes linearly with beam width, we choose beam size 40 in **KG-S2S** to trade-off between model performance and inference cost.

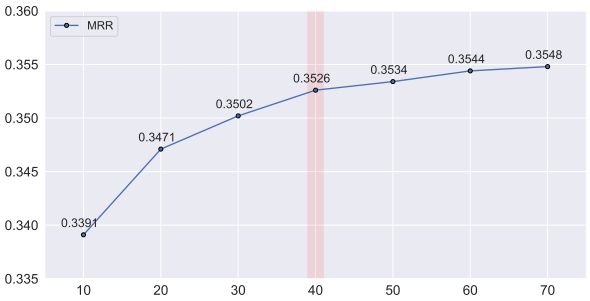

Figure 5: The effect of beam width to KS-S2S

Parameter Size Table 7 compares the model performance and parameter size between **KG-S2S** and StAR. Compared with StAR (354M trainable parameters), **KG-S2S** is based on a smaller T5-base backbone (220M trainable parameters, 1.6x less), while it achieves better performance with a relatively large margin (MRR 0.274 vs. 0.353). We further run **KG-S2S** using T5-small backbone (60M trainable parameters, 5.9x less). This variant of **KG-S2S** obtain slightly lower result (0.351 on MRR), but still outperforms StAR with substantial margin. The results suggest **i)** **KG-S2S** is not sensitive to the size of PLM; **ii)** **KG-S2S** is parameter-efficient.

[2] https://en.wikipedia.org/wiki/Amber_Riley

Model	Size	MRR	H@10
StAR	354M	.274	.455
KG-S2S (small)	60M	.351	.485
KG-S2S (base)	220M	.353	.495

Table 7: Comparison of model performance and parameter size between **KG-S2S** and StAR on **FB15K-237N**.

Parameter Growth Since **KG-S2S** represents KG elements (e.g. entities, relations and timestamps) as simple textual sequences, all KG tuples share the same vocabulary and language model parameters. As the training KG grows, the parameters of **KG-S2S** only increase due to the relations *Soft Prompt* with the complexity of $O(|\mathcal{R}| \cdot d)$ where d is the hidden size of Seq2Seq Pre-trained language model. On the contrary, traditional graph-based models represent entities, relations and other meta-information with distinct embeddings and the parameter growth of these models is $O((|\mathcal{E}| + |\mathcal{R}|)d)$. As $|\mathcal{R}| \ll |\mathcal{E}|$, the growth could be negligible and the parameter size of **KG-S2S** remains nearly constant given KGs with any size.

5 Conclusion and Future Work

In this paper, we present **KG-S2S** for various knowledge graph completion tasks. By converting different kinds of KG structures into "text-to-text" format, **KG-S2S** can directly produce the text of target predicted entities. Experimental results demonstrate that **KG-S2S** outperforms competitive baseline models in various KGC settings. In the future, we would explore extending **KG-S2S** to other Seq2Seq PLMs, such as BART (Lewis et al., 2020) and MASS (Song et al., 2019). In addition, it is interesting to combine **KG-S2S** with other knowledge-intensive NLP tasks, such as conversation recommendation (Li et al., 2018b) and commonsense generation (Wang et al., 2021b) in the Seq2Seq framework, and see if the KG knowledge could benefit these downstream tasks.

Acknowledgement

We thank the anonymous reviewers for their insightful suggestions to improve this paper. This work was supported by the National Research Foundation, Singapore under its Strategic Capability Research Centres Funding Initiative. Yufei Wang receives a MQ Research Excellence Scholarship and a CSIRO's DATA61 Top-up Scholarship.

References

Farahnaz Akrami, Mohammed Samiul Saeef, Qingheng Zhang, Wei Hu, and Chengkai Li. 2020. Realistic re-evaluation of knowledge graph completion methods: An experimental study. In *Proceedings of the 2020 International Conference on Management of Data, SIGMOD Conference 2020, online conference [Portland, OR, USA], June 14-19, 2020*, pages 1995–2010. ACM.

Antoine Bordes, Nicolas Usunier, Alberto García-Durán, Jason Weston, and Oksana Yakhnenko. 2013. Translating embeddings for modeling multi-relational data. In *Advances in Neural Information Processing Systems 26: 27th Annual Conference on Neural Information Processing Systems 2013. Proceedings of a meeting held December 5-8, 2013, Lake Tahoe, Nevada, United States*, pages 2787–2795.

Elizabeth Boschee, Jennifer Lautenschlager, Sean O'Brien, Steve Shellman, James Starz, and Michael Ward. 2015. ICEWS Coded Event Data. *Harvard Dataverse*.

Samuel Broscheit, Daniel Ruffinelli, Adrian Kochsiek, Patrick Betz, and Rainer Gemulla. 2020. Libkge - A knowledge graph embedding library for reproducible research. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations, EMNLP 2020 - Demos, Online, November 16-20, 2020*, pages 165–174. Association for Computational Linguistics.

Andrew Carlson, Justin Betteridge, Bryan Kisiel, Burr Settles, Estevam R. Hruschka, and Tom M. Mitchell. 2010. Toward an architecture for never-ending language learning. In *Proceedings of the Twenty-Fourth AAAI Conference on Artificial Intelligence*, AAAI'10, page 1306–1313. AAAI Press.

Mingyang Chen, Wen Zhang, Wei Zhang, Qiang Chen, and Huajun Chen. 2019. Meta relational learning for few-shot link prediction in knowledge graphs. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing, EMNLP-IJCNLP 2019, Hong Kong, China, November 3-7, 2019*, pages 4216–4225. Association for Computational Linguistics.

S. Chopra, R. Hadsell, and Y. LeCun. 2005. Learning a similarity metric discriminatively, with application to face verification. In *2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05)*, volume 1, pages 539–546 vol. 1.

Thomas H Cormen, Charles E Leiserson, Ronald L Rivest, and Clifford Stein. 2009. *Introduction to algorithms*. MIT press.

Shib Sankar Dasgupta, Swayambhu Nath Ray, and Partha P. Talukdar. 2018. Hyte: Hyperplane-based temporally aware knowledge graph embedding. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing, Brussels, Belgium, October 31 - November 4, 2018*, pages 2001–2011. Association for Computational Linguistics.

Tim Dettmers, Pasquale Minervini, Pontus Stenetorp, and Sebastian Riedel. 2018. Convolutional 2d knowledge graph embeddings. In *Proceedings of the Thirty-Second AAAI Conference on Artificial Intelligence, (AAAI-18), the 30th innovative Applications of Artificial Intelligence (IAAI-18), and the 8th AAAI Symposium on Educational Advances in Artificial Intelligence (EAAI-18), New Orleans, Louisiana, USA, February 2-7, 2018*, pages 1811–1818. AAAI Press.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT 2019, Minneapolis, MN, USA, June 2-7, 2019, Volume 1 (Long and Short Papers)*, pages 4171–4186. Association for Computational Linguistics.

Alberto García-Durán, Sebastijan Dumancic, and Mathias Niepert. 2018. Learning sequence encoders for temporal knowledge graph completion. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing, Brussels, Belgium, October 31 - November 4, 2018*, pages 4816–4821. Association for Computational Linguistics.

Rishab Goel, Seyed Mehran Kazemi, Marcus Brubaker, and Pascal Poupart. 2020. Diachronic embedding for temporal knowledge graph completion. In *The Thirty-Fourth AAAI Conference on Artificial Intelligence, AAAI 2020, The Thirty-Second Innovative Applications of Artificial Intelligence Conference, IAAI 2020, The Tenth AAAI Symposium on Educational Advances in Artificial Intelligence, EAAI 2020, New York, NY, USA, February 7-12, 2020*, pages 3988–3995. AAAI Press.

Zhen Han, Gengyuan Zhang, Yunpu Ma, and Volker Tresp. 2021. Time-dependent entity embedding is not all you need: A re-evaluation of temporal knowledge graph completion models under a unified framework. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing, EMNLP 2021, Virtual Event / Punta Cana, Dominican Republic, 7-11 November, 2021*, pages 8104–8118. Association for Computational Linguistics.

Shaoxiong Ji, Shirui Pan, Erik Cambria, Pekka Marttinen, and Philip S. Yu. 2022. A survey on knowledge graphs: Representation, acquisition, and applications. *IEEE Trans. Neural Networks Learn. Syst.*, 33(2):494–514.

C. Maria Keet. 2013. *Closed World Assumption*, pages 415–415. Springer New York, New York, NY.

Bosung Kim, Taesuk Hong, Youngjoong Ko, and Jungyun Seo. 2020. Multi-task learning for knowledge graph completion with pre-trained language

models. In *Proceedings of the 28th International Conference on Computational Linguistics, COLING 2020, Barcelona, Spain (Online), December 8-13, 2020*, pages 1737–1743. International Committee on Computational Linguistics.

Diederik P. Kingma and Jimmy Ba. 2015. Adam: A method for stochastic optimization. In *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*.

Timothée Lacroix, Guillaume Obozinski, and Nicolas Usunier. 2020. Tensor decompositions for temporal knowledge base completion. In *8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, April 26-30, 2020*. OpenReview.net.

Julien Leblay and Melisachew Wudage Chekol. 2018. Deriving validity time in knowledge graph. In *Companion of the The Web Conference 2018 on The Web Conference 2018, WWW 2018, Lyon , France, April 23-27, 2018*, pages 1771–1776. ACM.

Brian Lester, Rami Al-Rfou, and Noah Constant. 2021. The power of scale for parameter-efficient prompt tuning. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 3045–3059, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Veselin Stoyanov, and Luke Zettlemoyer. 2020. BART: denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, ACL 2020, Online, July 5-10, 2020*, pages 7871–7880. Association for Computational Linguistics.

Bing Li, Xiaochun Yang, Bin Wang, Wei Wang, Wei Cui, and Xianchao Zhang. 2018a. An adaptive hierarchical compositional model for phrase embedding. In *IJCAI*, pages 4144–4151.

Raymond Li, Samira Ebrahimi Kahou, Hannes Schulz, Vincent Michalski, Laurent Charlin, and Chris Pal. 2018b. Towards deep conversational recommendations. In *Advances in Neural Information Processing Systems*, volume 31. Curran Associates, Inc.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Roberta: A robustly optimized BERT pretraining approach. *CoRR*, abs/1907.11692.

Justin Lovelace, Denis Newman-Griffis, Shikhar Vashishth, Jill Fain Lehman, and Carolyn Rosé. 2021. Robust knowledge graph completion with stacked convolutions and a student re-ranking network. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 1016–1029, Online. Association for Computational Linguistics.

Xin Lv, Yankai Lin, Yixin Cao, Lei Hou, Juanzi Li, Zhiyuan Liu, Peng Li, and Jie Zhou. 2022. Do pre-trained models benefit knowledge graph completion? a reliable evaluation and a reasonable approach. In *Findings of the Association for Computational Linguistics: ACL 2022*, pages 3570–3581, Dublin, Ireland. Association for Computational Linguistics.

Guanglin Niu, Yang Li, Chengguang Tang, Ruiying Geng, Jian Dai, Qiao Liu, Hao Wang, Jian Sun, Fei Huang, and Luo Si. 2021. Relational learning with gated and attentive neighbor aggregator for few-shot knowledge graph completion. In *SIGIR '21: The 44th International ACM SIGIR Conference on Research and Development in Information Retrieval, Virtual Event, Canada, July 11-15, 2021*, pages 213–222. ACM.

Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Köpf, Edward Z. Yang, Zachary DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. 2019. Pytorch: An imperative style, high-performance deep learning library. In *Advances in Neural Information Processing Systems 32: Annual Conference on Neural Information Processing Systems 2019, NeurIPS 2019, December 8-14, 2019, Vancouver, BC, Canada*, pages 8024–8035.

Fabio Petroni, Tim Rocktäschel, Sebastian Riedel, Patrick Lewis, Anton Bakhtin, Yuxiang Wu, and Alexander Miller. 2019. Language models as knowledge bases? In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 2463–2473, Hong Kong, China. Association for Computational Linguistics.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J. Liu. 2020. Exploring the limits of transfer learning with a unified text-to-text transformer. *Journal of Machine Learning Research*, 21(140):1–67.

Apoorv Saxena, Adrian Kochsiek, and Rainer Gemulla. 2022. Sequence-to-sequence knowledge graph completion and question answering. *CoRR*, abs/2203.10321.

Kaitao Song, Xu Tan, Tao Qin, Jianfeng Lu, and Tie-Yan Liu. 2019. MASS: masked sequence to sequence pre-training for language generation. In *Proceedings of the 36th International Conference on Machine Learning, ICML 2019, 9-15 June 2019, Long Beach, California, USA*, volume 97 of *Proceedings of Machine Learning Research*, pages 5926–5936. PMLR.

Zhiqing Sun, Zhi-Hong Deng, Jian-Yun Nie, and Jian Tang. 2019. Rotate: Knowledge graph embedding by relational rotation in complex space. In *7th International Conference on Learning Representations, ICLR 2019, New Orleans, LA, USA, May 6-9, 2019*. OpenReview.net.

Zhiqing Sun, Shikhar Vashishth, Soumya Sanyal, Partha P. Talukdar, and Yiming Yang. 2020. A re-evaluation of knowledge graph completion methods. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, ACL 2020, Online, July 5-10, 2020*, pages 5516–5522. Association for Computational Linguistics.

Kristina Toutanova and Danqi Chen. 2015. Observed versus latent features for knowledge base and text inference. In *Proceedings of the 3rd workshop on continuous vector space models and their compositionality*, pages 57–66.

Théo Trouillon, Johannes Welbl, Sebastian Riedel, Éric Gaussier, and Guillaume Bouchard. 2016. Complex embeddings for simple link prediction. In *Proceedings of the 33nd International Conference on Machine Learning, ICML 2016, New York City, NY, USA, June 19-24, 2016*, volume 48 of *JMLR Workshop and Conference Proceedings*, pages 2071–2080. JMLR.org.

Shikhar Vashishth, Soumya Sanyal, Vikram Nitin, and Partha P. Talukdar. 2020. Composition-based multi-relational graph convolutional networks. In *8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, April 26-30, 2020*. OpenReview.net.

Bo Wang, Tao Shen, Guodong Long, Tianyi Zhou, Ying Wang, and Yi Chang. 2021a. Structure-augmented text representation learning for efficient knowledge graph completion. In *WWW '21: The Web Conference 2021, Virtual Event / Ljubljana, Slovenia, April 19-23, 2021*, pages 1737–1748. ACM / IW3C2.

Yufei Wang, Ian Wood, Stephen Wan, Mark Dras, and Mark Johnson. 2021b. Mention flags (MF): Constraining transformer-based text generators. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 103–113, Online. Association for Computational Linguistics.

Yufei Wang, Can Xu, Qingfeng Sun, Huang Hu, Chongyang Tao, Xiubo Geng, and Daxin Jiang. 2022. PromDA: Prompt-based data augmentation for low-resource NLU tasks. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 4242–4255, Dublin, Ireland. Association for Computational Linguistics.

Zhen Wang, Jianwen Zhang, Jianlin Feng, and Zheng Chen. 2014. Knowledge graph embedding by translating on hyperplanes. In *Proceedings of the Twenty-Eighth AAAI Conference on Artificial Intelligence, July 27 -31, 2014, Québec City, Québec, Canada*, pages 1112–1119. AAAI Press.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Rémi Louf, Morgan Funtowicz, Joe Davison, Sam Shleifer, Patrick von Platen, Clara Ma, Yacine Jernite, Julien Plu, Canwen Xu, Teven Le Scao, Sylvain Gugger, Mariama Drame, Quentin Lhoest, and Alexander M. Rush. 2020. Transformers: State-of-the-art natural language processing. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations, EMNLP 2020 - Demos, Online, November 16-20, 2020*, pages 38–45. Association for Computational Linguistics.

Tianbao Xie, Chen Henry Wu, Peng Shi, Ruiqi Zhong, Torsten Scholak, Michihiro Yasunaga, Chien-Sheng Wu, Ming Zhong, Pengcheng Yin, Sida I. Wang, Victor Zhong, Bailin Wang, Chengzu Li, Connor Boyle, Ansong Ni, Ziyu Yao, Dragomir R. Radev, Caiming Xiong, Lingpeng Kong, Rui Zhang, Noah A. Smith, Luke Zettlemoyer, and Tao Yu. 2022a. Unifiedskg: Unifying and multi-tasking structured knowledge grounding with text-to-text language models. *CoRR*, abs/2201.05966.

Xin Xie, Ningyu Zhang, Zhoubo Li, Shumin Deng, Hui Chen, Feiyu Xiong, Mosha Chen, and Huajun Chen. 2022b. From discrimination to generation: Knowledge graph completion with generative transformer. *CoRR*, abs/2202.02113.

Wenhan Xiong, Mo Yu, Shiyu Chang, Xiaoxiao Guo, and William Yang Wang. 2018. One-shot relational learning for knowledge graphs. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing, Brussels, Belgium, October 31 - November 4, 2018*, pages 1980–1990. Association for Computational Linguistics.

Chengjin Xu, Mojtaba Nayyeri, Fouad Alkhoury, Jens Lehmann, and Hamed Shariat Yazdi. 2019. Temporal knowledge graph embedding model based on additive time series decomposition. *CoRR*, abs/1911.07893.

Chengjin Xu, Mojtaba Nayyeri, Fouad Alkhoury, Hamed Shariat Yazdi, and Jens Lehmann. 2020. Tero: A time-aware knowledge graph embedding via temporal rotation. In *Proceedings of the 28th International Conference on Computational Linguistics, COLING 2020, Barcelona, Spain (Online), December 8-13, 2020*, pages 1583–1593. International Committee on Computational Linguistics.

Bishan Yang, Wen-tau Yih, Xiaodong He, Jianfeng Gao, and Li Deng. 2015. Embedding entities and relations for learning and inference in knowledge bases. In *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*.

Liang Yao, Chengsheng Mao, and Yuan Luo. 2019. KG-BERT: BERT for knowledge graph completion. *CoRR*, abs/1909.03193.

Yukun Zuo, Quan Fang, Shengsheng Qian, Xiaorui Zhang, and Changsheng Xu. 2018. Representation learning of knowledge graphs with entity attributes and multimedia descriptions. In *Fourth IEEE International Conference on Multimedia Big Data, BigMM 2018, Xi'an, China, September 13-16, 2018*, pages 1–5. IEEE.

A Dataset

ICEWS14 This dataset doesn't include any entity descriptions. As a result, we find the original data source[3] and create the description by combining 'sector' and 'country' entries for each entity.

NELL-One To conduct zero-shot learning for this dataset, we follow Wang et al. (2021a) to reformat the raw dataset so that the relations in the dev/test sets do not appear in the train set. Additionally, we observe that textual representations of entities and relations are written in lower letters. To avoid pretrain-finetune data format mismatch, we further capitalize the surface words for each entity name. Dataset statistics are shown in Table 8.

Dataset	Setting	$\|\mathcal{E}\|$	$\|\mathcal{R}\|$	|Train|	|Valid|	|Test|
WN18RR	SKGC	40,943	11	86,835	3,034	3,134
FB15K-237	SKGC	14,541	237	272,115	17,535	20,466
FB15K-237N	SKGC	14,541	93	87,282	7,041	8,226
ICEWS14	TKGC	6,869	230	72,826	8,941	8,963
NELL-One	FKGC	68,544	358	189,635	1,004	2,158

Table 8: Statistics of the Datasets.

All of these datasets are open-source English-written sources without any offensive content. They are introduced only for research use.

B Inplementation details

We implement our **KG-S2S** using PyTorch (Paszke et al., 2019) and HuggingFace (Wolf et al., 2020), and assess it on a single GPU (Tesla V100).

Model Input and Output We follow the T5 standard unsupervised training paradigm. We form the query texts by masking the target entities with T5 default special tokens. Answer texts are also wrapped by the T5 special tokens. We use square brackets around descriptions to distinguish them from entity names. A special separation token "|" is inserted to separate entity, relation and meta-information. During the inference stage, our model generates the raw text, and we remove the wrapping special tokens and corresponding entity descriptions with regular expression, remaining the entity names as model predictions. Practical results suggest that the predicted entities can be determined by the entity names, so it is unnecessary to generate all the descriptions. Consequently, we perform an early stopping generation strategy, that is, the generation process will be stopped if the model outputs reach maximum entity name length.

Seq2Seq Dropout Seq2Seq dropout is applied on the encoder input mask, randomly flipping the values from 1 to 0. Note that Seq2Seq dropout excludes the positions carrying special meanings, i.e. separation tokens, mask tokens and soft prompt.

Hyperparameters In terms of hyperparameters, we select the batch size from {32, 64, 128}, learning rate from {5e-3, 1e-3, 5e-4}, description length from {10, 40, 80}, Seq2Seq dropout from {0.0, 0.1, 0.2, 0.3}. The optimal configurations are displayed in Table 9

	batch size	learning rate	SRC/TGT desc.	S2S.Drop.
WN18RR	64	1e-3	40/40	0.1
FB15K-237	32	1e-3	80/80	0.2
FB15K-237N	32	1e-3	80/80	0.2
ICEWS14	32	5e-4	40/40	0.1
NELL-One	128	5e-4	0/0	0.0

Table 9: Hyperparameters for **KG-S2S**. SRC/TGT desc. denotes source and target description length. S2S.Drop denotes Seq2Seq dropout.

[3] https://dataverse.harvard.edu/dataverse/icews

Modelling Frequency, Attestation, and Corpus-Based Information with OntoLex-FrAC

Christian Chiarcos[1,2] and **Elena-Simona Apostol**[3,4] and **Besim Kabashi**[5] and **Ciprian-Octavian Truică**[3,4]
[1]Applied Computational Linguistics, Goethe University Frankfurt, Germany
[2]Institute for Digital Humanities, University of Cologne, Germany
[3]Department of Information Technology, Uppsala University, Sweden
[4]Faculty of Automatic Control and Computers, University Politehnica of Bucharest, Romania
[5]Computational and Corpus Linguistics, University of Erlangen-Nuremberg, Germany
chiarcos@cs.uni-frankfurt.de, elena-simona.apostol@it.uu.se
besim.kabashi@fau.de, ciprian-octavian.truica@it.uu.se

Abstract

OntoLex-Lemon has become a de facto standard for lexical resources in the web of data. This paper provides the first overall description of the emerging OntoLex module for Frequency, Attestations, and Corpus-Based Information (OntoLex-FrAC) that is intended to complement OntoLex-Lemon with the necessary vocabulary to represent major types of information found in or automatically derived from corpora, for applications in both language technology and the language sciences.

1 Background

The OntoLex-Lemon vocabulary has become the dominant vocabulary for modelling machine-readable dictionaries on the web of data, i.e., by means of RDF. And indeed, publishing lexical resources in RDF has a number of advantages, including the ease of integration of dictionary information not only with ontologies and knowledge graphs (this was the original domain of application), but also with other lexical data.

Figure 1 illustrates the OntoLex-Lemon core vocabulary. Primary data structures are `ontolex:LexicalEntry` (lexeme), `ontolex:Form` (word form), `ontolex:LexicalSense` (word sense), and `ontolex:LexicalConcept` (lexicalization-independent concept), so that lexical entries can be described, but also fine-grained differences in meaning and surface form.

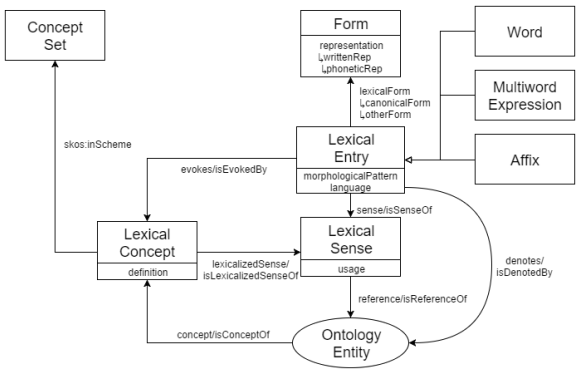

Figure 1: OntoLex-Lemon core module

While these aspects are advanced, stable and widely used, there is no complete module described in the current literature that enables interoperability and integration between lexical and textual resources and the distributional semantics of words, lexical senses and concepts, and collocation properties. By employing the usage of L(L)OD (Linked Linguistic Open Data) technologies, we describe the consolidation of OntoLex-FrAC (Frequency, Attestation, and Corpus Information), an OntoLex-Lemon model that *(1)* addresses the requirements of corpus-based lexicography (frequency and collocation information) and digital philology (linking lexical resources with corpus data), and *(2)* provides a standard for encoding, storing, and exchanging vector representations of words along with their lexical concepts, senses, and lemmas.

2 Core Concepts

So far, the development of FrAC has been conducted in a bottom-up fashion, where uses cases were analyzed and sub-vocabularies for different phenomena have been proposed. This includes frequency and attestations (Chiarcos et al., 2020), embeddings and similarity (Chiarcos et al., 2021) and collocations (Chiarcos et al., 2022). We complement these efforts with a top-down perspective, and we suggest three top-level classes to structure the model as a whole. In addition to that, we provide an OWL2/DL ontology to formalize the vocabulary. Restructuring the module entails a number of minor revisions regarding naming and scope of properties and classes, however, we aimed to stay faithful to the original definitions while integrating them into a more coherent overall picture.

Proceedings of the 29th International Conference on Computational Linguistics, pages 4018–4027
October 12–17, 2022.

The FrAC vocabulary is about information from or derived from corpora that can be included in machine-readable dictionaries and other forms of lexical or ontological resources, i.e., information about lexical forms (which can be counted), lexical entries (which can be illustrated with attestations or corpus examples), lexical senses or lexical concepts (which can be found as annotations in corpora). For these, FrAC introduced a generalization over the OntoLex core elements (and any other entity FrAC-related information is to be expressed about), and introduced the notion of `frac:Observable`, i.e., a lexical unit that can be observed in natural language, e.g., in a corpus. The corpus class was a another vocabulary element introduced with FrAC, and it is understood here in the more general sense of structured (collections of) primary data.[1] In addition to representing the primary data itself, it can also provide the total number of tokens in the corpus `frac:total`.

The different FrAC sub-vocabularies then defined different concepts that define the relation between observables and the corpus (or anyURI) object. A novel contribution of our paper is that we introduce a generalization over these FrAC-specific classes. In analogy with `frac:Observable`, we refer to this as `frac:Observation`. An observation in this understanding is any information found in, based on or created from a corpus, and the observations supported by the FrAC vocabulary are corpus frequency, attestation, collocation, similarity and embeddings. We consider aggregate observations (frequency, collocations, embeddings, similarity clusters) to be observations in their own right, as long as their characteristics are solely defined by the underlying data. FrAC observations have a number of common properties:

(1) `rdf:value`: value of an observation, with characteristics depending on the specific observation class.

(2) `dc:description`: human-readable characterization of the methods involved in the observation. FrAC does not provide a vocabulary for provenance – if such information is to be provided, we recommended to use Prov-O (Lebo et al., 2013)

(3) `frac:corpus` link from the observation to the structured data from which the observation was created.

3 FrAC Observations

We propose four main classes as subclasses of `frac:Observation`, i.e., frequency, attestation, collocations, embeddings, and similarity as summarized in Fig. 2.

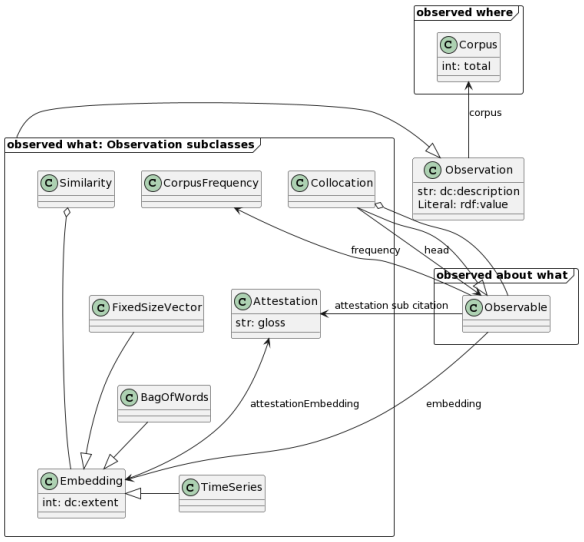

Figure 2: Revised FrAC vocabulary with the top-level classes `frac:Observable`, `frac:Observation` and `frac:Corpus`

3.1 Attestation

For attestation, the linking of lexical resources with corpus evidence, we distinguish three primary fields of application, i.e., lexicography (the use of references to corpora by a lexicographer to furnish evidence with reference to examples for the existence of a given lexical phenomena at a certain time period), language technology (linking a lexicon with the corpus from which information is derived), and corpus linguistics (linking a corpus [excerpt] with the lexical units or semantic annotations it provides). FrAC attestations are designed to support the different requirements in a unified way. FrAC defines `frac:Attestation` as an exact or normalized quotation or excerpt from a source document that exhibits a particular lexical entry, form, sense, lexeme or features such as spelling variation, morphology, syntax, collocation, register. An attestation should have a quotation or an attestation gloss and must define a locus object to identify the source of this material.

In the revised FrAC model, the attestation gloss – originally an independent property – is mod-

[1] In this more general sense, 'corpus' is also used in neighboring fields such as law (e.g., for Justinian's *Corpus Juris Civilis*) or archeology (e.g., for the Corpus Vasorum Antiquorum, a database of Greek vases). FrAC corpus thus comprises, but is not restricted to the sense of 'text corpus' (or speech corpus), i.e., a structured and/or electronically available and/or linguistically annotated collection of texts (or multimedia content).

elled as `rdf:value`. In its usage in lexicography, the attestation gloss differs from the quotation (`frac:quotation`) as it may include additional (human-readable) metadata about siglia, lines or versions that the actual primary data it refers to (the quotation) might not display. Similarly, the locus object (originally any URI as an object of `frac:locus`) is modelled as a corpus (`frac:Corpus` object of the `frac:corpus` property). This is in line with the fact that FrAC is underspecified as to the exact nature of `frac:corpus` objects, i.e., whether they represent the URI that resolves to the corpus that contains the lexical unit attested, or whether they represent the relevant excerpt of a corpus that contains the lexical unit, or whether they represent a metadata entry that stands in for a corpus which might not even exist in electronic form.

Any observable can be linked with its attestation by means of `frac:attestation`, defined as a subproperty of a more general `frac:citation` property – which posits no constraints on its range and which has been introduced to accommodate the needs of lexicographers who want to include attestations from secondary sources (Khan and Boschetti, 2018). The object of `frac:citation` is thus any URI, but for objects other than attestations, FrAC users are encouraged to follow any of the existing vocabularies for bibliographic data in RDF (Saur, 1998; Peroni and Shotton, 2012).

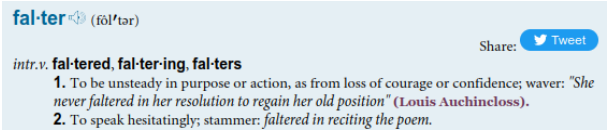
fal·ter (fôl′tər)

Share: Tweet

intr.v. **fal·tered, fal·ter·ing, fal·ters**

1. To be unsteady in purpose or action, as from loss of courage or confidence; waver: *"She never faltered in her resolution to regain her old position"* **(Louis Auchincloss).**

2. To speak hesitatingly; stammer: *faltered in reciting the poem.*

Figure 3: Attestations in the American Heritage Dictionary (accessed 2022-05-17)

Figure 3 shows a sample entry from the (online) American Heritage Dictionary (AHDictionary) of the English Language, and the attestation for its second sense can be modelled as follows:

```
:le_falter_vi
  a ontolex:LexicalEntry ;
  ontolex:sense :ls_falter_vi_2 .
:ls_falter_vi_2
  a ontolex:LexicalSense ;
  rdfs:comment "To speak hesitatingly,
      ..." ;
  frac:attestation [
    a frac:Attestation ;
    rdf:value "faltered in reciting the
        poem" ] .
```

While this attestation does not point to a corpus, the original 1969 edition of the dictionary uses the Brown corpus as a basis for its attestations, and for an example that *would have* come from the Brown corpus, we could give the link to the relevant subsections of the corpus in its online edition provided by SketchEngine.

```
... a frac:Attestation ;
    frac:corpus <https://app.
        sketchengine.eu/#concordance?
        corpname=preloaded%2Fbrown_1&
        keyword=falter&showresults=1> ;
    rdf:value "faltered in reciting the
        poem"
```

3.2 Frequency

The frequency distribution of linguistic elements is one of the most fundamental corpus-based statistics. In general, frequency information is critical to corpus studies, linguistic analysis, and NLP. We can distinguish between absolute and relative frequencies. Relative frequencies are generally normalized and computed as frequencies per a pre-defined number of linguistic elements. The FrAC module considers both absolute and relative frequency in order to facilitate different necessities. However, in terms of modelling, the focus is on absolute frequencies that are defined in relation to a particular corpus. The `frac:CorpusFrequency` class gives the absolute number of attestations, i.e., `rdf:value`, of a single `frac:Observable` considering a specific language resource, i.e., `frac:corpus`. Auxiliary filter conditions can be added to extend the `frac:CorpusFrequency` class with views of different sub-corpora.

An example is to restrict the subcorpus to a particular period. By means of OWL restrictions, a corpus-specific subclass of `frac:CorpusFrequency` can be created, say, `my:XYZCorpusFrequency` for which values for `dc:description`, `frac:corpus` and other parameters are defined as fixed. If then, the object of `frac:frequency` is defined as a `my:XYZCorpusFrequency`, these values do not have to be repeated, but are, instead, inherited from the class definition. As an example for frequency, we again, resort to the Brown corpus as provided by SketchEngine:

```
:BrownCorpusFreq
 rdfs:subClassOf
  frac:CorpusFrequency ,
   [ a owl:Restriction ;
    owl:onProperty frac:corpus ;
    owl:hasValue <https://app.
        sketchengine.eu/#concordance?
```

```
        corpname=preloaded%252Fbrown_1>
        ] .
```

For *falter*, the number of hits returned by querying SketchEngine can be modelled as a `:BrownCorpusFreq` in FrAC, then:

```
:le_falter_vi
  frac:frequency
    [ a :BrownCorpusFreq ;
      rdf:value "6" ] .
```

When re-defining corpus frequency as a `frac:Observation`, no semantic changes are necessary, except that `rdf:value` and `frac:corpus` are inherited now rather than defined for corpus frequency.

3.3 Collocation

A collocation is an expression containing two or more juxtaposition words that statistically appear together more frequently than by chance. The individual words of a collocation are characterized by the property of limited compositionality with each other, since they are predictable and, when they occur together, e.g. in the case of multi-word expressions, compound nouns, etc., they can have meanings that are different from their meanings when they occur alone or in other word combinations. Thus, some words can be freely combined with each other, others tend to combine only with certain words. They are word combinations that lie in a range between free and fixed. Collocation analysis is used in natural language processing, especially in automatic machine translation, in text generation, e.g. to make the output text as natural as possible, and to avoid untypical word combinations (Manning and Schütze, 1999; Evert, 2008).

In FrAC, collocations are modeled as an aggegate (`rdfs:Container`) of `frac:Observables`. Fixed word order collocations are defined using `rdf:Seq`) as a sequence, while variable word order collocations are defined using `rdf:Bag` as an ordered set. Collocations obtained by quantitative methods are characterized by their method of creation (`dc:description`), first word (`frac:head`), collocation strength (`rdf:value`), and the corpus used to create them (`frac:corpus`). Furthermore, collocations share these characteristics with other types of contextual relations. In previous FrAC proposals, these were thus inherited from the abstract class `frac:ContextualRelation` for the relation between two or more lexical elements. In our revised FrAC vocabulary,

LARGE CTy 82; CF 208; CTe 57	EF	IF	RF	TC	DI
LARGE AMOUNT OF DATA aha	2	2	*	2	4

Figure 4: Collocation analysis for the head word *large* and the collocation *(a) large amount of data* over the Brown corpus according to Kjellmer (1994) as given by Johansson (1998, p.339)

`frac:ContextualRelation` has been superseded by `frac:Observation`. A FrAC collocation is thus an aggregate (bag or sequence) of observables based on their co-occurrence within the same context window and characterized the head word of the collocation (`frac:head`) and the collocation score (`frac:cscore`) in a particular source corpus (`frac:corpus`).

Collocations are `frac:Observables`, they can also be given an `frac:attestation`, `frac:embedding` or `frac:frequency`. Using the embeddings, we can determine nested collocation by computing a similarity metric (e.g., cosine similarity). The collocation score (`frac:cscore`) is a subproperty of `rdf:value` that provides a specific corpus dependent collocation score. In FrAC we define multiple symmetric and asymmetric collocation metrics as sub-properties of `frac:cscore`, e.g., `frac:rel_freq` for the relative frequency (asymmetric), `frac:pmi` for the pointwise mutual information (symmetric), `frac:chi2` for Person's Chi-square test (asymmetric), etc. For asymmetric collocations scores the `frac:head` property is used to identify the elements' order.

As an example for collocation analysis over the Brown corpus, we refer to the classical work by Kjellmer (1994), who provides (candidate) collocations along with different scores in a tabular format (Fig. 4). Kjellmer's work differs from more recent works in collocation analysis in that he focuses on absolute frequencies rather than designated collocation scores. For this kind of data, it is sufficient to resort to `frac:CorpusFrequency` (resp., designated subclasses such as `:Inclusive-BrownFrequency`, with fixed values for `frac:corpus` and `dc:description`), and as collocations are both observations and observables, this is possible in FrAC:

```
:coll_laod
  a frac:Collocation, rdf:Seq ;
  rdf:_1 :le_large ;     # lexical
  rdf:_2 :le_amount ;    # entries
  rdf:_3 :le_of ;
  rdf:_4 :le_data ;
  frac:head :le_large ; # head
  frac:frequency  # frequencies
```

```
[ a :InclusiveBrownFrequency ; rdf:
    value "2" ] .
```

More conventional scores can be expressed with the designated subproperties of `frac:cscore` (or, if this is unambiguous, `rdf:value`).

3.4 Embeddings Subclasses

In the context of FrAC, the notion of embedding has been understood in a sense established in mathematics. An embedding is a structure-preserving projection (mapping) from a given domain into a numerical representation. The most popular example of embeddings in language technology is a more restricted form of embeddings in that sense, i.e., the topological space of the resulting embeddings is represented by fixed-size vectors (resp., tensors as aggregates of such vectors), but as there are other forms of numerical representations that serve similar or identical functions, FrAC introduces a more general class of `frac:Embedding` along with a specific sub-class of embeddings `frac: FixedSizeVector` to for embeddings as typically found in NLP. Other embedding subclasses are `frac:BagOfWords` (for unweighted or weighted bags of words), and `frac:TimeSeries` (for sequences of fixed-size vectors). Both representations are similar to embeddings in the NLP sense in that they represent a projection into a numerical feature space and that the primary function of this projection is to provide distance measurements. For bags of words, these are represented by confidence scores for weighted bag of words models (or booleans for unweighted bags of words) for *every word in the vocabulary* (at least, this would be a possible mathematical interpretation; in practice, such data is not represented as a vector, but as a hashtable – or, for unweighted bags of words, a set –, so that only words with positive scores are listed). Mathematically, bags of words could also be described as infinite-size embeddings (if the vocabulary is not completely known in advance), and indeed, earlier methods for dimensionality reduction motivated embeddings as a compact form of bags of words (Schütze, 1992, with slightly different wording).

Time series data is another form of infinite-size embeddings, but here, it is an infinite-size series of finite-size vectors. In language technnology, a stream of text, mapped to word embeddings, is such a structure – but normally not stored. A lexicographically more relevant use case is in sensor data, e.g., for the recording of gestures for sign languages. Such recordings can then be compared with each other using techniques such as dynamic time warping (Gold and Sharir, 2018), and then be the basis for automated clustering, etc.

3.5 Word and Concept Embeddings

In FrAC, any observable can be assigned an embedding. This includes lexical form, lemmas (lexical entries), word senses (lexical senses), lexical concepts and other entities, multi-word expressions (as lexical entries) and groups of observables (FrAC collocations). This also partially answers the question on why embeddings (esp., fixed-size vectors for NLP embeddings) are a necessary data structure for FrAC. In many use cases, word embeddings are created on the fly and not shared across different applications – but as their creation involves a non-deterministic element, they cannot be easily compared across languages or corpora. For this reason (and because, historically, the creation of embeddings from large-scale corpora was a matter of weeks or months of processing), applications often use precompiled embeddings for either subsequent fine-tuning or directly. As far as word embeddings are concerned, it does – again – not seem to be necessary to represent these in RDF. The typical structure of an embedding file is a table, with the first column representing the token, the following columns representing the embedding with one value per cell. As long as applications refer to the same embedding file with the same parameters (same length, same tokenization, same normalization for strings [e.g., lowercasing], same vector normalization function – e.g., to the spans of either $[0, ..., 1]$ or $[-1, ..., 1]$ –, same underlying corpus data), they will operate in the same embedding space, and with libraries such as TextTorch (TorchText, 2022) or repositories like HuggingFace (Wolf et al., 2020), there is an established infrastructure to retrieve identical word embeddings using standard identifiers. However, this can nevertheless be problematic, especially if different applications retrieve their embeddings from different sources. While the retrieval of, say, GloVe embeddings (Pennington et al., 2014) via TextTorch or via the original provider *should* lead to the same result, this cannot be automatically validated as neither source provides machine-readable metadata – nor is there any transparent relation between both methods of access unless TextTorch code is manually inspected.

RDF metadata can help here to resolve ambiguities. And ambiguities exist, and will intensify with the adoption of current techniques and data in new programming languages and future ecosystems. A classical example are the infamous 'Collobert & Weston embeddings' which is a term applied to two different and unrelated sets of embeddings also known as 'SENNA embeddings' on the one hand and 'Turian embeddings' on the other (Collobert, 2011).

Another objective to provide embeddings in RDF is that the basic table format in which embeddings are shared is insufficient if these embeddings are detached from the definition of the elements they are assigned to. An important category here are embeddings of lexical concepts and lexical senses as derived, for example, from underlying word embeddings, and thus residing in the same feature space, e.g., the classical AutoExtend embeddings (Rothe and Schütze, 2017) whose synset identifiers are ambiguous as to which WordNet version they refer to. While this can be solved with using persistent URIs as synset identifiers, for lexical resources for which no publicly accessible, resolvable or persistent URIs can be provided, an alternative solution is to *bundle* embeddings and the underlying knowledge graph into a data structure from which both the graph and the embeddings can be accessed, and FrAC provides the vocabulary to provide that from the RDF perspective.

3.6 Contextualized Embeddings

Another aspect in which static word embeddings have been superseded by more recent developments is the rise of transformer architectures operating with subsymbolic embeddings and the processing of text spans rather than static lexemes. *Contextualized* embeddings for a phrase, a lexical unit or another observable can be represented in FrAC as (a property of the) attestation of the observable in a corpus: `frac:attestationEmbedding` assigns an attestation an embedding. For encoding multiple contextual embeddings for a particular lexical entry, say, *tree*, it is necessary to create one attestation with one attestation embedding property. If possible, the attestation object should be linked to the respective passage in the corpus, but in the spirit of the open world assumption in RDF semantics, this information is optional, so that a minimal encoding of contextual embeddings in a lexical resource can use the following template:

```
:le_tree a ontolex:LexicalEntry ,
           frac:Observable ;
  frac:embedding [
    a frac:FizedSizeVector;
    dc:extent "50";
    rdf:value "[ 0.0001, ... ]" ];
  frac:attestation [
    frac:attestationEmbedding [
      a frac:FixedSizeVector;
      dc:extent "50";
      rdf:value "[ 0.5352, ... ]" ] ].
```

From the perspective of a corpus, both contextualized and context-free embeddings can be encoded correspondingly. If we use the CoNLL-RDF vocabulary for identifying tokens in a corpus (FrAC can be used with any vocabulary for this purpose, e.g., Web Annotation or NIF), and the token `doc:s1_5` has already been defined, then, it can just be linked with the attestation:

```
... frac:attestation [
      frac:corpus doc:s1_5;
      frac:attestationEmbedding
        [ ... ] ] .
```

For the token `doc:s1_5`, we can now easily retrieve various kinds of embeddings:

```
doc:s1_5 ^frac:corpus ?att.
?att frac:attestationEmbedding
       ?contextualEmbedding .
?att ^frac:attestation
   [ a ontolex:LexicalEntry;
     frac:embedding ?entryEmbedding ].
doc:s1_5
  ^frac:corpus/^frac:attestation
   [ a ontolex:LexicalForm;
     frac:embedding ?formEmbedding ].
doc:s1_5
  ^frac:corpus/^frac:attestation
   [ a ontolex:LexicalSense;
     frac:embedding ?senseEmbedding ].
doc:s1_5
  ^frac:corpus/^frac:attestation
   [ a ontolex:LexicalConcept;
     frac:embedding ?conceptEmbedding ].
```

These partial queries operate on separate attestations for every kind of observables. However, the model is generic enough to also follow indirect links: Using OntoLex core data structures, a sense attestation can server as an anchor to retrieve embeddings for lexical sense, but also lexical concept or lexical entry: For the Brown corpus, a concrete application can be seen in the SemCor corpus (Fellbaum et al., 1997), a layer of semantic annotation (WordNet senses), and with the following query we can retrieve AutoExtend synset embeddings and contextualized sense embeddings:

```
SELECT ?contextualEmbedding
       ?synsetEmbedding
WHERE {
  ?att frac:attestationEmbedding [
```

```
  rdf:value ?contextualEmbedding ] .
?att ^frac:attestation ?sense.
?sense
  a ontolex:LexicalSense;
    ontolex:isLexicalizedSenseOf
      ?synset.
?synset a ontolex:LexicalConcept;
  frac:embedding/rdf:value
      ?synsetEmbedding .
```

Such data can then be used, for example, to train a mapping from contextual embeddings to synset embeddings.

In their earlier formulation of the FrAC vocabulary, `frac:Embedding` had the following attributes: *(1)* `dc:extent` dimensionality of embeddings (for fixed-size vectors), or the number of data points per observation (in time series data) *(2)* `rdf:value` value of the embeddings, according to the examples, this should be a JSON literal, e.g., an array (of floats) or a hashtable of keys (e.g., context words) and numerical weights. *(3)* `dc:description` human-readable description of embedding type and parameters *(4)* `frac:corpus` URI of the underlying corpus data

With the revised upper model, embedding inherits `rdf:value`, `dc:description` and `frac:corpus` from `frac:Observation`. As part of the generalization, the restriction of `rdf:value` to JSON literals is abandoned – and this may, indeed, conincide with external requirements to the FrAC vocabulary, as it seems easier at times to encode embeddings (fixed size vectors or bags of words) just as plain strings, as such data can be more easily created from existing resources.

3.7 Similarity

Similarity relates to computing the strength of the semantic relationships between different elements, e.g., forms, lexemes, and phrases. There are various similarity metrics, but in the FrAC context, similarity is obtained through a numerical description of the contexts for each of the analysed elements, i.e., their embeddings (Sect. 3.4).

In FrAC, similarity is represented using the `frac:Similarity` class, an aggregate (set, or bag) of FrAC observables, that represents a relation between two or more embeddings (`frac:Embeddings`). In the revised FrAC model, the earlier `frac:ContextualRelation` superclass of `frac:Similarity` from which its properties were inherited has been replaced by `frac:Observation`, no further renaming necessary. FrAC similarity can be applied to both similarity relations (sets of two observables) and similarity clusters (sets of two or more observables) characterized by a single value.

The value of a similarity is an `rdf:value` calculated according to the employed Embedding model, e.g., the number of shared dimensions – in a bag-of-word model. This value is published together with its corresponding metadata, *(2)* one or more source corpora, i.e., `frac:corpus`, *(2)* the description of the comparison method, i.e., `dc:description`.

We can use `frac:Similarity` for different scenarios, two being exemplified in FrAC, i.e., the similarity between two words, and similarity clusters. Similarity clusters are useful in computational linguistics for tasks that rely on cognate recognition and language similarity. When applied to a particular corpus, similarity cluster offers a generalization score over all the pairs of similes. This generalization method can use different approaches, e.g., the minimal similarity between all members in the cluster, or a score given by the clustering algorithm. The used approach must be explained in `dc:description`.

A very simple example for Similarity is cosine similarity, as can be calculated between fixed size vectors. Using the AutoExtend embeddings, the cosine similarity between any two lexical concepts can be modelled as follows:

```
:ls_abc a ontolex:LexicalSense ;
  frac:embedding :ls_abc_embedding .
:ls_xyz a ontolex:LexicalSense ;
  frac:embedding :ls_xyz_embedding .
[ dc:description
  "cosine similarity" ]
  a frac:Similarity, rdfs:Bag ;
  rdfs:member :ls_abc_embedding ,
              :ls_xyz_embedding ;
  rdf:value "0.0036" .
```

4 Consolidation and Outlook

The revised FrAC vocabulary proposed in this paper introduces the novel class `frac:Observation` as a generalization over different kinds of phenomena that can be observed from or derived from corpus data and that are relevant for lexical resources. With small changes to previously proposed vocabulary elements, this revised top-level structure can be seamlessly applied to use cases and sample data featured in previous publications on FrAC.

The following changes have been proposed: *(1)* merge the `frac:locus` property of `frac:Attestation` into `frac:corpus`; *(2)* extend the understanding of `frac:Corpus` / range of `frac:corpus` to cover any piece of structured (collections of) primary data, including parts thereof; *(3)* merge `frac:quotation` and `rdf:value`; *(4)* abandon previous restrictions on the range of `rdf:value`; and *(5)* merge `frac:ContextualRelation` with the newly created class `frac:Observation`

Aside from inheriting common characteristics from a newly created generalization, we claim that this model is equivalent in expressivity to the current formulation of FrAC as available from the public draft of the vocabulary. but that it features a more systematic structure in that the common pattern exhibited for modelling the different types of observations and corpus-derived information is now explicitly encoded in the model, making the overall model both easier to describe, more compact and easier to formalize in RDFS semantics.

We illustrate the applicability of this model to a number of examples, mostly with reference to the Brown corpus. We argue that with this in formation, it becomes possible for the first time to encode both the majority of lexical information derived from the Brown corpus in a unified way, and to thus integrate those resources on a technical level. With FrAC, all these different aspects can be encoded in RDF and this representation can be the basis to define interoperable APIs for different web services, APIs or applications to produce, consume or integrate such data. At the moment, the state of the art in this area is probably best represented by the *proprietary* SketchEngine APIs, whose responses are, however, do not come with any guarantees for long-term stability or reproducibility. Furthermore, a number of aspects are not well-supported by SketchEngine: This includes, for example, the online reference to individual attestations (SketchEngine only provides resolvable URIs for query responses, but not for the individual matches), or the retrieval of embeddings (provided by SketchEngine but only as data dumps, not integrated in the API). Another possible application is to provide dumps of corpus-derived information (of attestations, embeddings, collocations, similarity clusters or frequency lists) along with the associated lexical graph.

As it provides uniform data structures on the basis of web standards, FrAC represents the fundament to develop consistent access protocols for the unified access, public exposure, exchange and integration of heterogeneous data as currently provided, for example, via the Linguistic Linked Open Data cloud (Chiarcos et al., 2011; Declerck et al., 2020), libraries such as NLTK (Bird, 2006) or via portals such as HuggingFace (Lhoest et al., 2021). At the same time, FrAC accomodates the needs of digital lexicography and the language sciences, and has partially been motivated by applicability to philological data (Chiarcos et al., 2020) and multimedia content (Chiarcos et al., 2011). As a result of applying OntoLex and FrAC, resources developed in lexicography become accessible and re-usable in the context of language technology, and resources and solutions developed in language technology become applicable to lexicographic and linguistic data and research challenges.

The second novel contribution of this paper is that we provide an OWL2/DL ontology as formalization of the revised FrAC vocabulary. This allows to automatically validate FrAC data, to detect inconsistencies and to perform reasoning (inferences) over FrAC data. In particular, types (classes) of observations and observables can be automatically (RDFS-)inferred from domain and range constraints of properties such as `frac:frequency`, `frac:attestation`, `frac:embedding`, etc., so that this information is in fact optional in data exchange. Likewise, formal OWL2/DL axioms allow users to define application-specific subclasses of `frac:CorpusFrequency`, etc., so that these can be used as a short hand for specific bundles of observations with `frac:corpus`, `rdf:value`, `dc:description` and other properties that constrain the corpus under consideration or that define hyperparameters used in the extraction process.

It is important to note that – to the best of our knowledge – no RDF vocabulary of similar scope was in existence prior to FrAC, and that the integration with OntoLex facilitates a relatively wide application across different disciplines and research networks. It is less certain whether there are comparable pre-RDF vocabularies in existence. We mentioned the SketchEngine API and exchange formats, but as far as *open (community) standards* are concerned, we are not aware of any related work of similar scope. Nevertheless, aspects of corpus-driven lexicography have been addressed in the Lexical Markup Framework (Romary et al., 2019,

for attestation) and the TEI guidelines (Burnard, 2013, for collocation), but we are not aware of any formal standard for embeddings or machine-readable similarity scores. By extending OntoLex with a designated module for frequency, attestation and corpus-based information in lexical resources, we are thus breaking novel ground.

Acknowledgments

The research described in this paper was conducted in the context of the Cost Action CA18209 *Nexus Linguarum. European network for Web-centred linguistic data science*. Ciprian-Octavian Truică was partially funded through the OPTIM Research project (POCU grant no. 62461/03.06.2022, SMIS code 153735). Moreover, the authors would like to thank Maxim Ionov, Anas Fahad Khan, and Julia Bosque-Gil for contributing to the development of OntoLex-FrAc as well as to all OntoLex-FrAC contributors.

References

Steven Bird. 2006. Nltk: The natural language toolkit. In *Proceedings of the COLING/ACL 2006 Interactive Presentation Sessions*, pages 69–72.

Lou Burnard. 2013. The evolution of the text encoding initiative: from research project to research infrastructure. *Journal of the Text Encoding Initiative*, (5).

Christian Chiarcos, Thierry Declerck, and Maxim Ionov. 2021. Embeddings for the lexicon: Modelling and representation. In *Proceedings of the 6th Workshop on Semantic Deep Learning (SemDeep-6)*, pages 13–19.

Christian Chiarcos, Katerina Gkirtzou, Maxim Ionov, Besim Kabashi, Anas Fahad Khan, and Ciprian-Octavian Truică. 2022. Modelling collocations in OntoLex-FrAC. In *Proceedings of GlobaLex-2022*, Marseille, France.

Christian Chiarcos, Sebastian Hellmann, and Sebastian Nordhoff. 2011. Towards a linguistic linked open data cloud: The open linguisticsworking group. *TAL Traitement Automatique des Langues*, 52(3):245–275.

Christian Chiarcos, Maxim Ionov, Jesse de Does, Katrien Depuydt, Fahad Khan, Sander Stolk, Thierry Declerck, and John Philip McCrae. 2020. Modelling frequency and attestations for ontolex-lemon. In *Proceedings of the 2020 Globalex Workshop on Linked Lexicography*, pages 1–9.

Ronan Collobert. 2011. SENNA. https://ronan.collobert.com/senna/, accesased 2022-05-16.

Thierry Declerck, John Philip McCrae, Matthias Hartung, Jorge Gracia, Christian Chiarcos, Elena Montiel-Ponsoda, Philipp Cimiano, Artem Revenko, Roser Sauri, Deirdre Lee, et al. 2020. Recent developments for the linguistic linked open data infrastructure. In *Proceedings of the 12th Language Resources and Evaluation Conference*, pages 5660–5667.

Stefan Evert. 2008. Corpora and collocations. In Kytö M Lüdeling A, editor, *Corpus Linguistics. An International Handbook*, pages 1212–1248. Mouton de Gruyter, Berlin, New York.

Christiane Fellbaum, Joachim Grabowski, and Shari Landes. 1997. Analysis of a hand-tagging task. In *In Proceedings of ANLP-97 Workshop on Tagging Text with Lexical Semantics, Why, What, and How?*, Washington, D.C.

Omer Gold and Micha Sharir. 2018. Dynamic time warping and geometric edit distance: Breaking the quadratic barrier. *ACM Trans. Algorithms*, 14(4).

Stig Johansson. 1998. Göran kjellmer. a dictionary of english collocations, based on the brown corpus. *International Journal of Corpus Linguistics*, 3(2):338–348.

Anas Fahad Khan and Federico Boschetti. 2018. Towards a Representation of Citations in Linked Data Lexical Resources. In *Proceedings of the XVIII EURALEX International Congress: Lexicography in Global Contexts*, pages 137–147, Ljubljana, Slovenia. Ljubljana University Press.

Göran Kjellmer. 1994. *A Dictionary of English Collocations. Based on the Brown Corpus*. Clarendon Press. 3 vols.

Timothy Lebo, Satya Sahoo, Deborah McGuinness, Khalid Belhajjame, James Cheney, David Corsar, Daniel Garijo, Stian Soiland-Reyes, Stephan Zednik, and Jun Zhao. 2013. Prov-o: The prov ontology. *W3C Recommendation*.

Quentin Lhoest, Albert Villanova del Moral, Yacine Jernite, Abhishek Thakur, Patrick von Platen, Suraj Patil, Julien Chaumond, Mariama Drame, Julien Plu, Lewis Tunstall, Joe Davison, Mario Šaško, Gunjan Chhablani, Bhavitvya Malik, Simon Brandeis, Teven Le Scao, Victor Sanh, Canwen Xu, Nicolas Patry, Angelina McMillan-Major, Philipp Schmid, Sylvain Gugger, Clément Delangue, Théo Matussière, Lysandre Debut, Stas Bekman, Pierric Cistac, Thibault Goehringer, Victor Mustar, François Lagunas, Alexander Rush, and Thomas Wolf. 2021. Datasets: A community library for natural language processing. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 175–184. Association for Computational Linguistics.

Christopher Manning and Hinrich Schütze. 1999. *Foundations of Statistical Natural Language Processing*. MIT Press Ltd.

Jeffrey Pennington, Richard Socher, and Christopher D Manning. 2014. Glove: Global vectors for word representation. In *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*, pages 1532–1543.

Silvio Peroni and David Shotton. 2012. Fabio and cito: ontologies for describing bibliographic resources and citations. *Web Semantics: Science, Services and Agents on the World Wide Web*, 17:33–43.

Laurent Romary, Mohamed Khemakhem, Fahad Khan, Jack Bowers, Nicoletta Calzolari, Monte George, Mandy Pet, and Piotr Bański. 2019. Lmf reloaded. *arXiv preprint arXiv:1906.02136*.

Sascha Rothe and Hinrich Schütze. 2017. Autoextend: Combining word embeddings with semantic resources. *Computational Linguistics*, 43(3):593–617.

KG Saur. 1998. IFLA study group on the functional requirements for bibliographic records.

Hinrich Schütze. 1992. Dimensions of meaning. In *Supercomputing'92: Proceedings of the 1992 ACM/IEEE Conference on Supercomputing*, pages 787–796. IEEE.

TorchText. 2022. TorchText 0.4.0 documentation. Technical report.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Remi Louf, Morgan Funtowicz, Joe Davison, Sam Shleifer, Patrick von Platen, Clara Ma, Yacine Jernite, Julien Plu, Canwen Xu, Teven Le Scao, Sylvain Gugger, Mariama Drame, Quentin Lhoest, and Alexander Rush. 2020. Huggingface's transformers: State-of-the-art natural language processing. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 38–45. Association for Computational Linguistics.

Contrast Sets for Stativity of English Verbs in Context

Daniel Chen & Alexis Palmer
Department of Linguistics
University of Colorado
{daniel.chen-1, alexis.palmer}@colorado.edu

Abstract

For the task of classifying verbs in context as **dynamic** or **stative**, current models approach human performance, but only for particular data sets. To better understand the performance of such models, and how well they are able to generalize beyond particular test sets, we apply the **contrast set** (Gardner et al., 2020) methodology to stativity classification. We create nearly 300 contrastive pairs by perturbing test set instances just enough to change their labels from one class to the other, while preserving coherence, meaning, and well-formedness. Contrastive evaluation shows that a model with near-human performance on an in-distribution test set degrades substantially when applied to transformed examples, showing that the stative vs. dynamic classification task is more complex than the model performance might otherwise suggest. Code and data are freely available.[1]

1 Introducing stativity

Aspectual properties of verbs, and the clauses they inhabit, have the potential to support a range of natural language processing tasks, such as event ordering (Modi and Titov, 2014) and temporal relation classification (Costa and Branco, 2012), as well as contributing to speaker choices around situational construal (Trott et al., 2020). At the same time, verb and situational aspect are a complex set of interacting properties, in which the meaning of the verb, the nature of its arguments, adverbial modifiers, and grammatical features such as verb tense and nominal definiteness can all play a role in determining the aspectual make-up of a clause.

This sensitivity of aspectual categorization to small shifts in linguistic form is one reason that automatic prediction of aspectual classes is an especially challenging computational problem. In this paper we explore the stability of automatic classification for one particular facet of aspect: stativity of English verbs. Stativity reflects the degree to which a verb represents a static situation versus a situation that reflects some degree of dynamicity. Dynamic verbs typically involve some change of state. (1-3) below show examples of the three classes relevant for our study: DYNAMIC verbs, STATIVE verbs, and verbs for which annotators CANNOT_DECIDE.

(1) Table 7 shows results from the latest experiments. STATIVE

(2) Dr. Smith showed her students how to work with the new GPUs. DYNAMIC

(3) The earlier paper shows the effectiveness of incorporating linguistic features. CANNOT_DECIDE

In (1), the table is static, and the results exist in the table; no change of state is indicated. (2) highlights the dynamic sense of *show*, in which the Agent is giving a demonstration. (3) allows two readings. In the STATIVE reading, the result about linguistic features is a static property of the paper; it simply exists in the paper. In the DYNAMIC reading, the paper demonstrates the effectiveness of linguistic features through an argument that develops and progresses over the course of the paper.

Most verbs in English have a strong predominant category (stative or dynamic), yet allow for variable interpretation, depending on context. A smaller number of verbs (e.g. *show*), are highly flexible, with no strong statistical tendency in either direction (Friedrich and Palmer, 2014a; Falk and Martin, 2016). Because of this variability, automatic classification of aspectual properties requires contextual input and instance-level classification.

To better understand the ability of systems to automatically determine stativity, we produce contrast sets (section 2) for English verb stativity with 298 transformed instances. The contrast set instances and their ground-truth labels are extracted from the SitEnt corpus (section 2.2), and the transformed instances are produced using a range of

[1] https://github.com/dchensta/se_contrast

Proceedings of the 29th International Conference on Computational Linguistics, pages 4028–4036
October 12–17, 2022.

linguistically-motivated transformation strategies, detailed in section 3.

Using a standard modeling configuration (section 4), we show that classification performance on the transformed instances is substantially lower than on the original instances. According to Friedrich (2017), observed annotator agreement for this task ranges from 79% to 82%. On the original instances, the model achieves micro-averaged accuracy of nearly 80%, approaching human agreement for this task.[2] On the transformed instances, micro-averaged accuracy is well below 60%.

2 Building contrast sets for stativity

For many NLP tasks, neural models, especially those built on large language models, have been shown to be sensitive to annotation artifacts in the data on which they are trained and evaluated. High performance of classifiers often hinges on preserving these properties at evaluation time, and testing on out-of-distribution data can result in such dramatic performance decreases that the models no longer reliably perform the task they have been trained to do (Gururangan et al., 2018; Poliak et al., 2018; Geva et al., 2019, among others). These findings have given rise to methodologies for more careful evaluation of classification capability. Several different methods for improved evaluation have been proposed (Ribeiro et al., 2020; Gardner et al., 2020, among others).

2.1 Contrast sets

In this work, we follow the contrast set methodology (Gardner et al., 2020). The core idea is to create **contrastive** evaluation data sets by having experts make small perturbations to instances in the original test sets. In our case, we vary the lexical aspect of the main verb so that the preferred label changes from DYNAMIC (4) to STATIVE (5):

(4) Mary ran the Buenos Aires Marathon.

(5) Mary was a participant in the Buenos Aires Marathon.

These perturbations need to strike a delicate balance. The changes should be large enough to change the gold label for the instance, yet small enough to retain meaning, coherence, and validity. We also aim to use a variety of strategies, so as not to introduce *new* unintended annotation artifacts.

[2]Note that cross-validation accuracy on the much larger training set ranges from 77-80%.

	DYN	STAT	CD
Training	18,357	15,507	8,445
Test	376	217	57
Contrast: Test_Orig	172	120	0
Contrast: Test_Trans	120	172	0

Table 1: Distribution of DYNAMIC (DYN), STATIVE (STAT), and CANNOT_DECIDE (CD) labels.

Once contrast sets have been built, we compare the performance of the model in question on the *transformed* test instances (with their new labels) to the performance of the same model on the *original* version of those same instances. Significant performance degradation on the transformed test data calls into question whether the model has learned to classify the phenomena modeled in the annotated training data. The **contrast set** consists of the paired original and transformed test instances.

2.2 Data

We use data from the SitEnt (situation entities) corpus (Friedrich and Palmer, 2014b; Friedrich, 2017). The corpus[3] combines data from MASC (Ide et al., 2008) with Wikipedia texts, creating a collection of documents from 13 different genres. Texts are segmented into clauses, and each clause is triply-annotated for stativity of the main verb, genericity of the main referent, habituality of the situation described, and finally, a clause-level situation type label (following Smith (2003), these labels distinguish between events, states, generics, generalizing sentences, facts, propositions, reports, questions, imperatives, and undecided). Gold labels come from a majority vote across the three annotators.

For model training, we use Friedrich et al. (2016)'s original training split, which consists of 324 documents, with 42,309 clauses. As a basis for building contrast sets, we select four documents from the original test set, each from a different genre: news, essay, journal, and Wikipedia. For each sentence, we create one contrast set by transforming the first clause in the sentence.[4] No contrast sets are created for clauses labeled CAN-

[3]https://github.com/annefried/sitent/tree/master/annotated_corpus

[4]To test the viability of using first clauses only, we created contrast sets for all clauses in the Wikipedia document and compared classifier performance for the full document vs. only initial clauses. There was no significant difference.

NOT_DECIDE. We produce 292 contrast sets.[5]

Table 1 shows the distribution of the three labels for our data set. **Test** refers to the four selected documents; **Contrast: Test_Orig** refers to the original versions for the 292 contrast sets; and **Contrast: Test_Trans** refers to the transformed counterparts of the original instances, with flipped labels. Note that DYNAMIC to STATIVE transformations outnumber STATIVE to DYNAMIC transformations.

3 Transformation strategies

After building the contrast sets, we perform an analysis of the linguistic properties of the various strategies used. Most transformations hinge on the main verb, either replacing the lexical item or changing the role of the verb so that it moves to a different structural and semantic configuration.

3.1 DYNAMIC –> STATIVE

Example sentences showing the DYNAMIC to STATIVE transformations can be found in Table 2.

1. THOUGHT VERB - Demote a dynamic verb from main to secondary verb by moving it into the subordinate THEME role for verbs of thinking, believing, or feeling.
2. COPULA - Replace main verb with a simple predication headed by a copular verb.
3. DESCRIPTIVE VERB - Replace the dynamic action with a descriptive verb, effectively reconfiguring the dynamic action as stative properties of the subject noun.
4. LIGHT VERB - Use the possessive [6] light verb construction with *have* to make *have* the new main verb.
5. SEMI-MODAL - Use a semi-modal verb (e.g. *need to*, *ought to*) marking deontic modality, which concerns the speaker's requirements and desires, as a "thought" or "emotion" from the speaker, who can be an unspecified authority with no referent.
6. DOWNGRADE TO PPL - Remove main verb from the clause by transforming it into a perfect passive participle that favors a descriptive, adjectival reading over a verbal reading.
7. ORDER - Switch the order of the clauses and insert a descriptive verb as the new main verb.

3.2 STATIVE –> DYNAMIC

Example sentences showing each of the STATIVE to DYNAMIC transformations can be found in Table 3.

1. NEW PARTICIPANT - Choose a synonymous verb that introduces an agent who participates in a dynamic synonym of the original verb.
2. INSERT VERB - Replace a stative verb (typically the copula) with a dynamic verb or insert a dynamic verb as the new main verb, of which the original stative verb is a dependent.
3. BECOMING - Replace standard copula with an inflected form of the verbs *to become* or *to get*, and their synonyms. This preserves the copula's predicating structure while reformulating the event as dynamic.
4. UPGRADE - Upgrade a perfect passive participle or subordinate STATIVE verb to the main verb of the clause by adding a helping verb or deleting the main STATIVE verb.
5. HEAVY VERB - Replace a light verb construction like *have* with a heavy, dynamic verb.

4 Model, results, and discussion

Having built contrast sets, we now evaluate performance compared to the original instances.

4.1 Model

Our straightforward classification model first learns a contextualized representation for each clause using BERT (Devlin et al., 2019), followed by a regression layer to classify the clause representations as DYNAMIC, STATIVE, or CD. The logistic regression model is trained using the liblinear solver and L2 regularization. Running 5-fold cross-validation over the full training set, using the trained logistic regression model, yielded accuracy scores ranging from 77.7% to 80.13%, only slightly below observed human agreement, which Friedrich (2017) reports as ranging from 79% to 82%.

4.2 Results: Classifying contrast sets

Table 4 shows the model's performance on the original test set instances and the transformed instances. These figures include all clauses for the Wikipedia text and only initial clauses for the other three texts, and only clauses whose original labels are either STATIVE or DYNAMIC. On the original instances, the model achieves micro-averaged accuracy of nearly 80%, approaching human agreement for this task. On the transformed instances, micro-averaged accuracy is well below 60%.

[5] This number is on par with the data sets described in Gardner et al. (2020), which range from 70 to 1000 contrast sets per task.

[6] verbs of possession have a STATIVE reading

Strategy	Original Instance (DYN)	Transformed Instance (ST)	#
THOUGHT VERB	During that time, the panel **said**,	During that time, the panel **believed** that	55
COPULA	Although it **affected** Youngstown and the surrounding area more than it affected other regions,	Although it **was** in Youngstown and the surrounding area more than in other regions,	48
DESCRIPTIVE VERB	"Your actions and failure to act **led to** violations of Senate rules	"Your actions and failure to act **constituted** direct violations of Senate rules	47
LIGHT VERB	**Scoring** higher than 21	**Having** a score higher than 21	14
SEMI-MODAL	The players' initial cards may be **dealt** face up	The players' initial cards **need to** consistently be dealt either face up	5
DOWNGRADE TO PPL	When **examining** the areas history, culture, and economic situation	**Based** on the area's history, culture, and economic situation	2
ORDER	the player or the dealer **wins** by having a score of 21 or by having the highest score	having a score of 21 **means** the player or the dealer wins	1

Table 2: Examples of DYNAMIC –> STATIVE transformations, along with the number of times each transformation strategy was used in the contrast sets.

Strategy	Original Instance (ST)	Transformed Instance (DYN)	#
NEW PARTICIPANT	11 plus the value of any other card will always **be** less than or equal to 21.	Players **add** 11 plus the value of any other card to get less than or equal to 21.	81
INSERT VERB	Since the 1960s, blackjack has **been** a high-profile target of advantage players, particularly card counters,	Since the 1960s, blackjack has **functioned** as a high-profile target of advantage players, particularly card counters,	19
BECOMING	One such bonus **was** a ten-to-one payout	One such bonus **became** a ten-to-one payout	13
UPGRADE PPL	Other casino games inspired by blackjack **include** Spanish 21 and pontoon.	Other casino games were **inspired** by blackjack, including Spanish 21 and pontoon.	5
HEAVY VERB	After receiving their initial two cards, players **have** the option of getting a "hit",	After receiving their initial two cards, players may **pursue** the option of getting a "hit",	2

Table 3: Examples of STATIVE –> DYNAMIC transformations, along with the number of times each transformation strategy was used in the contrast sets.

4.3 Results: Transformation strategies

Finally, we look at contrast set classification accuracy across different transformation strategies. In the STATIVE to DYNAMIC direction, the strategy most often classified correctly (56%) is NEW PARTICIPANT. In the reverse direction, the strategy of replacing the DYNAMIC main verb with a COPULA has the highest accuracy (52%). Notably, even the highly stative nature of the copula doesn't always result in the model recognizing the transformed clause as stative. (Detailed results in Appendix A.)

Both THOUGHT VERB and DESCRIPTIVE VERB for converting DYNAMIC clauses to STATIVE perform poorly. This indicates that verbs of feeling, thinking, and wanting (THOUGHT VERB) and certain verbs like *signify*, *constitute*, and *include*, are not uniformly treated as STATIVE by the classifier. Other descriptive verbs, like *contain* and *resemble* do get accurately classified. Some verbs like *appear* sometimes get classified correctly as STATIVE, as in "All other cards appear as the numeric value", other times as DYNAMIC, as in "Cards appear either from one or two handheld decks, from a dealer's shoe, or from a shuffling machine."

The frequency of transformation strategy in the contrast sets does not correspond to high classification accuracy. In future, we will look at how frequent such constructions are in the training data, and whether their association with stativity labels matches linguistic expectations. We suspect that the different types of stativity may also play a role. For example, the stativity commonly associated with descriptions of mental processes is different from the attributional or predicational stativity often seen with copular constructions.

5 Related work on lexical aspect and verb stativity

Aspectual structure is complex and well-studied in the linguistics literature (Vendler, 1967; Comrie, 1976; Moens and Steedman, 1988; Smith, 1991, among many others). Classically, aspectual analysis involves the semantic properties of stativity, telicity, durativity, and iterativity. Croft et al. (2016) expand on this set of properties in their discussion of aspectual annotations within the Rich Event Description framework. Donatelli et al. (2018) propose methods for expanding the Abstract Meaning

Document	# Clauses Full Text	# Contrast Sets	Test Orig: Correct	Test Trans: Correct	Test Orig: Acc	Test Trans: Acc	Diff by # Clauses	Prop Diff
Wikipedia	114	102	70	60	67.39	58.82	-10	-9.8%
News	149	47	42	16	89.36	34.04	-26	-55.32%
Journal	67	19	11	11	57.89	57.89	-0	-0%
Essay	316	130	114	70	87.68	53.85	-44	-33.85%
Total or Microavg.	646	298	237	157	79.53	52.68		

Table 4: Accuracy on **Contrast: Test_Orig** and **Contrast: Test_Trans**, by document. "Diff by Clause" shows the # of clauses misclassified after transformation, and "Prop Diff" shows the percentage of misclassified contrast sets.

Representation framework with aspectual features, and such aspectual information is a key feature of the Uniform Meaning Representation framework (Van Gysel et al., 2021). In addition, aspectual properties are relevant at both the clause level and the level of individual verbs.

Our current focus is the verb-level property of **stativity**, sometimes referred to as *inherent lexical aspect*.[7] Stativity reflects the degree to which a verb represents a static situation versus a situation that reflects some degree of dynamicity. Dynamic verbs typically involve some change of state.

Computational approaches to stativity. Early approaches to computational analysis of verb stativity employ rule-based approaches based on known linguistic tests for stativity, such as the progressive test.[8] Klavans and Chodorow (1992) produce a type-level stativity rating for English verbs, based on the frequency with which verbs occur in various tenses in the Brown and Reader's Digest corpora. Dorr and Olsen (1997) treat stativity as one of several aspectual properties derivable from logical representations of verb meaning in the Lexical Conceptual Structure (LCS) framework (Jackendoff, 1983, 1990). Siegel and McKeown (Siegel, 1999; Siegel and McKeown, 2000) use a wide range of linguistic indicators to derive type-level stativity values for English verbs. Friedrich and Palmer (2014a) extend Siegel and McKeown's work to incorporate distributional features and perform classification in context. Kober et al. (2020) use distributional semantics to classify both stativity and telicity across genres. Falk and Martin (2016) take a more fine-grained approach to lexical aspect classification for French verbs in context, categorizing verbs across a set of 13 different verbal *readings*. Hermes et al. (2018) take a distributional approach to classifying German verbs for Aktionsart, and (Egg et al., 2019) provide a new annotated corpus and classification experiments for multiple components of aspect for German verbs.

Another important line of research (Govindarajan et al., 2019; Gantt et al., 2022) takes a broader view of event meaning, treating stativity as one of a number of aspectual features which together compose the meaning of an event. Similarly, work on clause-level semantic aspect classification (*aka* situation entity classification) (Friedrich et al., 2016; Becker et al., 2017; Dai and Huang, 2018) considers stativity as a key semantic property for determining clause-level aspect. Finally, Chen et al. (2021) use a sequence of rules to assign tense and aspect values to both verbal events and event nominals, making use of co-occurrence cues of part-of-speech tags, special lexical items, and semantic configurations that help the classifier select the right shade of aspect for a given situation.

6 Conclusions

We apply the contrast set methodology to the task of classifying English verbs in context as stative or dynamic. We see a serious performance degradation on the transformed examples, suggesting the model has not learned a clean decision boundary for stativity. This first analysis suggests a need to more clearly define features that may bias clauses toward stative or dynamic readings.

The study would benefit from more data, across a wider range of text types. We would also like to investigate the effectiveness of contrastive evaluation for other semantic properties, using recently-developed methods for partially-automatic contrast set creation (Li et al., 2020; Bitton et al., 2021; Ross et al., 2021, among others).

[7] *Aktionsart* (Vendler, 1957) also models lexical aspect. Both *Aktionsart* and stativity are subject to coercion at the clause-level, as described in the introduction.

[8] Generally, static verbs in English cannot occur in progressive form: e.g. **I am knowing Thai*. This is one of the most robust of the linguistic tests, but it too is subject to exceptions: e.g. *I am liking this song!*

7 Acknowledgements

This project is supported, in part, by a grant from the Center for Humanities & the Arts at the University of Colorado Boulder.

References

Maria Becker, Michael Staniek, Vivi Nastase, Alexis Palmer, and Anette Frank. 2017. Classifying semantic clause types: Modeling context and genre characteristics with recurrent neural networks and attention. In *Proceedings of the 6th Joint Conference on Lexical and Computational Semantics (*SEM 2017)*, pages 230–240, Vancouver, Canada. Association for Computational Linguistics.

Yonatan Bitton, Gabriel Stanovsky, Roy Schwartz, and Michael Elhadad. 2021. Automatic generation of contrast sets from scene graphs: Probing the compositional consistency of GQA. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 94–105, Online. Association for Computational Linguistics.

Daniel Chen, Martha Palmer, and Meagan Vigus. 2021. AutoAspect: Automatic annotation of tense and aspect for uniform meaning representations. In *Proceedings of The Joint 15th Linguistic Annotation Workshop (LAW) and 3rd Designing Meaning Representations (DMR) Workshop*, pages 36–45, Punta Cana, Dominican Republic. Association for Computational Linguistics.

Bernard Comrie. 1976. *Aspect: An introduction to the study of verbal aspect and related problems*, volume 2. Cambridge university press.

Francisco Costa and António Branco. 2012. Aspectual type and temporal relation classification. In *Proceedings of the 13th Conference of the European Chapter of the Association for Computational Linguistics*, pages 266–275, Avignon, France. Association for Computational Linguistics.

William Croft, Pavlina Pešková, and Michael Regan. 2016. Annotation of causal and aspectual structure of events in RED: a preliminary report. In *Proceedings of the Fourth Workshop on Events*, pages 8–17, San Diego, California. Association for Computational Linguistics.

Zeyu Dai and Ruihong Huang. 2018. Building context-aware clause representations for situation entity type classification. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 3305–3315, Brussels, Belgium. Association for Computational Linguistics.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. *arXiv:1810.04805 [cs]*. ArXiv: 1810.04805.

Lucia Donatelli, Michael Regan, William Croft, and Nathan Schneider. 2018. Annotation of tense and aspect semantics for sentential AMR. In *Proceedings of the Joint Workshop on Linguistic Annotation, Multiword Expressions and Constructions (LAW-MWE-CxG-2018)*, pages 96–108, Santa Fe, New Mexico, USA. Association for Computational Linguistics.

Bonnie J. Dorr and Mari Broman Olsen. 1997. Deriving verbal and compositonal lexical aspect for NLP applications. In *35th Annual Meeting of the Association for Computational Linguistics and 8th Conference of the European Chapter of the Association for Computational Linguistics*, pages 151–158, Madrid, Spain. Association for Computational Linguistics.

Markus Egg, Helena Prepens, and Will Roberts. 2019. Annotation and automatic classification of aspectual categories. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 3335–3341, Florence, Italy. Association for Computational Linguistics.

Ingrid Falk and Fabienne Martin. 2016. Automatic identification of aspectual classes across verbal readings. In *Proceedings of the Fifth Joint Conference on Lexical and Computational Semantics*, pages 12–22, Berlin, Germany. Association for Computational Linguistics.

Annemarie Friedrich. 2017. *States, events, and generics: computational modeling of situation entity types*. Ph.D. thesis, Universität des Saarlandes.

Annemarie Friedrich and Alexis Palmer. 2014a. Automatic prediction of aspectual class of verbs in context. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 517–523, Baltimore, Maryland. Association for Computational Linguistics.

Annemarie Friedrich and Alexis Palmer. 2014b. Situation entity annotation. In *Proceedings of LAW VIII - The 8th Linguistic Annotation Workshop*, pages 149–158, Dublin, Ireland. Association for Computational Linguistics and Dublin City University.

Annemarie Friedrich, Alexis Palmer, and Manfred Pinkal. 2016. Situation entity types: automatic classification of clause-level aspect. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1757–1768, Berlin, Germany. Association for Computational Linguistics.

William Gantt, Lelia Glass, and Aaron Steven White. 2022. Decomposing and Recomposing Event Structure. *Transactions of the Association for Computational Linguistics*, 10:17–34.

Matt Gardner, Yoav Artzi, Victoria Basmov, Jonathan Berant, Ben Bogin, Sihao Chen, Pradeep Dasigi, Dheeru Dua, Yanai Elazar, Ananth Gottumukkala, Nitish Gupta, Hannaneh Hajishirzi, Gabriel Ilharco, Daniel Khashabi, Kevin Lin, Jiangming Liu, Nelson F. Liu, Phoebe Mulcaire, Qiang Ning, Sameer

Singh, Noah A. Smith, Sanjay Subramanian, Reut Tsarfaty, Eric Wallace, Ally Zhang, and Ben Zhou. 2020. Evaluating models' local decision boundaries via contrast sets. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 1307–1323, Online. Association for Computational Linguistics.

Mor Geva, Yoav Goldberg, and Jonathan Berant. 2019. Are we modeling the task or the annotator? an investigation of annotator bias in natural language understanding datasets. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 1161–1166, Hong Kong, China. Association for Computational Linguistics.

Venkata Govindarajan, Benjamin Van Durme, and Aaron Steven White. 2019. Decomposing generalization: Models of generic, habitual, and episodic statements. *Transactions of the Association for Computational Linguistics*, 7:501–517.

Suchin Gururangan, Swabha Swayamdipta, Omer Levy, Roy Schwartz, Samuel Bowman, and Noah A. Smith. 2018. Annotation artifacts in natural language inference data. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 2 (Short Papers)*, pages 107–112, New Orleans, Louisiana. Association for Computational Linguistics.

Jürgen Hermes, M.P.R. Richter, and Claes Neuefeind. 2018. Supervised classification of aspectual verb classes in german - subcategorization-frame-based vs window-based approach: A comparison. In *ICAART*.

Nancy Ide, Collin Baker, Christiane Fellbaum, Charles Fillmore, and Rebecca Passonneau. 2008. MASC: the manually annotated sub-corpus of American English. In *Proceedings of the Sixth International Conference on Language Resources and Evaluation (LREC'08)*, Marrakech, Morocco. European Language Resources Association (ELRA).

Ray Jackendoff. 1983. *Semantics and Cognition*. The MIT Press.

Ray Jackendoff. 1990. *Semantic Structures*. The MIT Press.

Judith L. Klavans and Martin Chodorow. 1992. Degrees of stativity: The lexical representation of verb aspect. In *COLING 1992 Volume 4: The 14th International Conference on Computational Linguistics*.

Thomas Kober, Malihe Alikhani, Matthew Stone, and Mark Steedman. 2020. Aspectuality across genre: A distributional semantics approach. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 4546–4562, Barcelona, Spain (Online). International Committee on Computational Linguistics.

Chuanrong Li, Lin Shengshuo, Zeyu Liu, Xinyi Wu, Xuhui Zhou, and Shane Steinert-Threlkeld. 2020. Linguistically-informed transformations (LIT): A method for automatically generating contrast sets. In *Proceedings of the Third BlackboxNLP Workshop on Analyzing and Interpreting Neural Networks for NLP*, pages 126–135, Online. Association for Computational Linguistics.

Ashutosh Modi and Ivan Titov. 2014. Inducing neural models of script knowledge. In *Proceedings of the Eighteenth Conference on Computational Natural Language Learning*, pages 49–57, Ann Arbor, Michigan. Association for Computational Linguistics.

Marc Moens and Mark Steedman. 1988. Temporal ontology and temporal reference. *Computational Linguistics*, 14(2):15–28.

Adam Poliak, Jason Naradowsky, Aparajita Haldar, Rachel Rudinger, and Benjamin Van Durme. 2018. Hypothesis only baselines in natural language inference. In *Proceedings of the Seventh Joint Conference on Lexical and Computational Semantics*, pages 180–191, New Orleans, Louisiana. Association for Computational Linguistics.

Marco Tulio Ribeiro, Tongshuang Wu, Carlos Guestrin, and Sameer Singh. 2020. Beyond accuracy: Behavioral testing of NLP models with CheckList. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 4902–4912, Online. Association for Computational Linguistics.

Alexis Ross, Tongshuang Wu, Hao Peng, Matthew E. Peters, and Matt Gardner. 2021. Tailor: Generating and perturbing text with semantic controls. *CoRR*, abs/2107.07150.

Eric V. Siegel. 1999. Corpus-based linguistic indicators for aspectual classification. In *Proceedings of the 37th Annual Meeting of the Association for Computational Linguistics*, pages 112–119, College Park, Maryland, USA. Association for Computational Linguistics.

Eric V. Siegel and Kathleen R. McKeown. 2000. Learning methods to combine linguistic indicators:improving aspectual classification and revealing linguistic insights. *Computational Linguistics*, 26(4):595–627.

Carlota Smith. 1991. The parameter of aspect. *Studies in Linguistics and Philosophy*, 43:27–64.

Carlota S. Smith. 2003. *Modes of Discourse: The local structure of texts*. Cambridge University Press.

Sean Trott, Tiago Timponi Torrent, Nancy Chang, and Nathan Schneider. 2020. (Re)construing meaning in NLP. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 5170–5184, Online. Association for Computational Linguistics.

Jens E. L. Van Gysel, Meagan Vigus, Jayeol Chun, Kenneth Lai, Sarah Moeller, Jiarui Yao, Tim O'Gorman, Andrew Cowell, William Croft, Chu-Ren Huang, Jan Hajič, James H. Martin, Stephan Oepen, Martha Palmer, James Pustejovsky, Rosa Vallejos, and Nianwen Xue. 2021. Designing a Uniform Meaning Representation for Natural Language Processing. *KI - Künstliche Intelligenz*, 35(3):343–360.

Zeno Vendler. 1957. Verbs and times. *The Philosophical Review*, 66(2):143–160.

Zeno Vendler. 1967. Facts and events. *Linguistics in philosophy*, pages 122–146.

A Classification results by transformation strategy

Table 5 shows the distribution of correct labels assigned to transformed clauses using DYNAMIC to STATIVE strategies. Table 6 shows the same for STATIVE to DYNAMIC transformations.

Strategy	**Test Trans: Size**	**Test Trans: Correct**	**Acc**
THOUGHT VERB	55	10	18.18%
COPULA	48	25	52.08%
DESCRIPTIVE VERB	47	17	36.17%
LIGHT VERB	14	4	28.57%
SEMI-MODAL	5	0	0%
DOWNGRADE TO PPL	2	1	50%
ORDER	1	0	0%
Totals	172	57	33.14%

Table 5: Successful DYNAMIC > STATIVE transformation strategies, evaluated by accuracy of correctly identifying the contrast label.

Strategy	**Test Trans: Size**	**Test Trans: Correct**	**Acc**
NEW PARTICIPANT	81	45	55.55%
INSERT VERB	19	9	47.37%
BECOMING	13	9	69.23%
UPGRADE PPL	5	2	40%
HEAVY VERB	2	1	50%
Totals	120	66	55%

Table 6: Successful STATIVE > DYNAMIC transformation strategies, evaluated by accuracy of correctly identifying the contrast label.

Multilingual and Multimodal Topic Modelling with Pretrained Embeddings

Elaine Zosa and **Lidia Pivovarova**
Department of Computer Science
University of Helsinki
Helsinki, Finland
firstname.lastname@helsinki.fi

Abstract

This paper presents M3L-Contrast—a novel multimodal multilingual (M3L) neural topic model for comparable data that maps texts from multiple languages and images into a shared topic space. Our model is trained jointly on texts and images and takes advantage of pretrained document and image embeddings to abstract the complexities between different languages and modalities. As a multilingual topic model, it produces aligned language-specific topics and as multimodal model, it infers textual representations of semantic concepts in images. We demonstrate that our model is competitive with a zero-shot topic model in predicting topic distributions for comparable multilingual data and significantly outperforms a zero-shot model in predicting topic distributions for comparable texts and images. We also show that our model performs almost as well on unaligned embeddings as it does on aligned embeddings.

1 Introduction

Topic modelling is an unsupervised method initially designed for text data that extracts latent themes in documents through the co-occurrence statistics of the words in the documents. In most probabilistic topic models, a topic is a distribution over a vocabulary and a document is a distribution over topics (Blei et al., 2003). Multilingual topic models extend basic topic models for multilingual data by jointly training on multiple languages (Mimno et al., 2009; Hao and Paul, 2018). These models learn aligned language-specific topics and have been used in different cross-lingual applications such as multilingual news clustering (De Smet and Moens, 2009) and comparing discourses from different cultures in news and social media (Shi et al., 2016; Gutiérrez et al., 2016).

Most topic models are designed for textual data but there is also a rich body of work on applying topic modelling to images resulting in multimodal topic models (Barnard et al., 2003; Feng and Lapata, 2010; Roller and Im Walde, 2013). These models use natural language supervision to improve the semantic representation of images. Augmenting topical information from text with topic information from visual inputs also produces better semantic representations of words.

Neural topic models have been proposed to improve on classical topic models and have resulted in models that are more computationally efficient and produces more coherent topics (Srivastava and Sutton, 2017). Moreover, the neural topic modelling framework has given rise to models that take advantage of information from external sources such as word embeddings (Dieng et al., 2020) and contextualised language models (Bianchi et al., 2021a,b; Hoyle et al., 2020; Mueller and Dredze, 2021).

In this work, we present a novel neural multilingual *and* multimodal topic model that takes advantage of pretrained document and image embeddings to abstract the complexities between languages and modalities. Our work is based on the contextualized topic model (CTM, Bianchi et al., 2021a,b), a family of topic models that uses contextualized document embeddings as input.

We show that while ZeroshotTM (Bianchi et al., 2021b), a cross-lingual variant of CTM, can predict relevant topic distributions of documents in languages it has not seen during training, this ability does not transfer well to unseen modalities (e.g. images). Moreover, since ZeroshotTM only sees monolingual data, it produces monolingual topics that are inferred from documents in a single language. This approach does not take into account possible biases in worldviews that are hidden in different languages.

Our approach, which we refer to as M3L-Contrast, trains jointly on multilingual texts and images with a contrastive objective. We show that our model produces better topic distributions for comparable texts and images compared to Ze-

Proceedings of the 29th International Conference on Computational Linguistics, pages 4037–4048
October 12–17, 2022.

roshotTM even with unaligned embeddings and our model also improves on a classical multilingual topic model for comparable multilingual data. The main contributions of this work are:

1. We present a neural multimodal *and* multilingual topic model for comparable data that maps images and texts into a shared topic space;
2. we show that contrastive learning is effective in mapping embeddings from unaligned encoders into a shared topic space *and* improves on the alignment of aligned embeddings;
3. we present a multilingual topic model for comparable multilingual data that uses pretrained embeddings and improves on a classical topic model for comparable data.[1]

2 Related Work

2.1 Neural topic models

Neural topic models (NTMs) refer to a class of topic models that use neural networks to estimate the parameters of the topic-word and document-topic distributions. Using a variational autoencoder (VAE) to map documents into latent topic spaces was proposed by Srivastava and Sutton (2017) and demonstrated in the ProdLDA model that exhibited better topic coherences and faster training than classical models. This has led to other VAE-based topic models that can incorporate information from external sources such as the Embedded Topic Model (Dieng et al., 2020) which uses pretrained word embeddings, the Contextualised Topic Model (Bianchi et al., 2021a) which uses contextualised embeddings and the BERT-based Autoencoder as Teacher (Hoyle et al., 2020) model that distills large language models to improve topic coherence.

2.2 Multilingual topic models

Multilingual topic models infer aligned language-specific topics from a multilingual dataset. To align topics across languages, some degree of supervision is required to establish the link between the languages. In most cases, the languages are linked either at a word level or at a document level (Hao and Paul, 2020). Models that use word-level supervision require a translation dictionary to link words from different languages (Jagarlamudi and Daumé, 2010; Hao and Paul, 2018; Yang et al., 2019). Document-level supervision requires a comparable dataset where a document in one language is linked to a thematically similar document in another language (Mimno et al., 2009; De Smet and Moens, 2009).

The Polylingual Topic Model (PLTM, Mimno et al., 2009) is widely-used classical multilingual topic model for comparable data. To our knowledge, the Neural Multilingual Topic Model (Wu et al., 2020), a model that uses word-level supervision, is the only neural multilingual topic model so far. ZeroshotTM (Bianchi et al., 2021b), while not a multilingual model, is capable of zero-shot cross-lingual topic inference: it can predict topic distributions for documents in unseen languages if the model is trained on embeddings from a multilingual encoder. However, ZeroshotTM requires *aligned* embeddings for zero-shot topic modelling.

2.3 Multimodal topic models

Multimodal topic models use data from different modalities to infer topics. The most popular pairing is texts and images. Some text-and-image topic models use labelled image datasets to learn natural language representations of images using a supervised topic modelling approach (Barnard et al., 2003; Zheng et al., 2014). Other models extract 'visual words' from images using image feature extractors such as SIFT and images are represented as a bag of 'visual words' in the same manner that documents are represented as a bag of textual words (Feng and Lapata, 2010; Virtanen et al., 2012; Roller and Im Walde, 2013). (An et al., 2020) trained visual and textual topic models from neural network representations for multimodal depression detection but does not map text and images into the same topic space.

2.4 Contrastive learning

Contrastive learning is a self-supervised technique that uses different views of the same data to learn better data representations (Jaiswal et al., 2021; Liu et al., 2021). In contrastive training the goal is to minimize the distance between positive samples while separating them from negative samples. Contrastive training is popular in multimodal settings such as web-scale text-image alignment (Radford et al., 2021; Jia et al., 2021), audio-visual alignment (Khorrami and Räsänen, 2021) and biomedical imaging (Zhang et al., 2020).

In neural topic modelling, contrastive learning has recently been used to improve on the Adversar-

[1] Our code is available at https://github.com/ezosa/M3L-topic-model

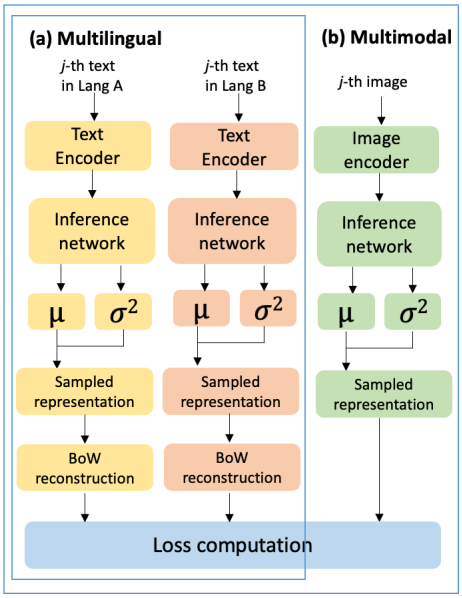

Figure 1: Proposed M3L-Contrast topic model. (a) Multilingual topic model with language-specific encoders and inference networks; (b) Extension to the multimodal setting. The loss function is detailed in Equation 1.

ial Topic Model (Wang et al., 2019) by adding a contrastive objective to the training loss and taking a more principled approach to sampling positive and negative samples (Nguyen and Luu, 2021).

3 Multilingual and Multimodal Model

3.1 Neural multilingual topic model

We first propose a neural multilingual topic model for comparable multilingual data that uses pre-trained document embeddings. Our multilingual model is based on ZeroshotTM (Bianchi et al., 2021b), a zero-shot cross-lingual topic model. However, we are not aiming for a zero-shot model. Instead, our model infers aligned *language-specific topics* for each language present in the dataset. Moreover, our approach does not require the pre-trained document embeddings to be aligned beforehand. This property makes it advantageous in settings where a multilingual encoder that includes our desired language might not exist such as in low-resource settings.

Figure 1(a) shows the multilingual model architecture. The model uses independent inference networks for each language. To align language-specific topics, the model minimizes the Kullback-Leibler (KL) divergence between the topic distributions of comparable documents from different languages and, in addition, uses a contrastive loss to map similar instances close to each other in the topic space and keep non-related instances apart.

For each tuple of aligned documents in the comparable multilingual dataset, we encode the documents from each language using their own separate encoders (whether aligned or non-aligned) and then the embeddings are passed to language-specific inference networks that infers the mean, μ, and variance, σ^2, of the Gaussian distribution from which we sample latent document-topic distributions. At this point, the languages are independent of each other and have not yet shared any information.

After sampling topic distributions for each document, we induce a shared topic space by minimizing the pairwise KL divergence between the language-specific distributions whose parameters are estimated from their own inference network. We also add a contrastive objective so that aligned examples are kept away from other examples in the topic space. We use InfoNCE (Van den Oord et al., 2018) as our contrastive loss. The positive pairs are all possible combinations of document pairs from the same tuple and negative pairs are all other pairs of documents from different tuples within a batch. For instance, for a comparable dataset with two languages and batch size N, we would have N positive pairs and $N^2 - N$ negative pairs per batch. For three languages, that would be $3N$ positive pairs and $3(N^2 - N)$ negative pairs, etc.

Thus, the loss consists of the three components: the reconstruction loss; the KL divergence between topic distributions; and the contrastive loss. Formally, the loss function is written as:

$$\mathcal{L} = \sum_{l=0}^{L} \mathbb{E}_q[w^\top \log(\text{softmax}(\beta_l \theta_d))] - \sum_{\substack{a,b=0 \\ a \neq b}}^{n} \mathbb{KL}(p(\theta_i^a | x_i^a) || q(\theta_i^b | x_i^b)) - s \sum_{\substack{a,b=0 \\ a \neq b}}^{n} \log \frac{\exp((\theta_i^a \cdot \theta_i^b)/\tau)}{\sum_{j=0}^{N} \sum_{c,d=0}^{n} \exp((\theta_i^c \cdot \theta_j^d)/\tau)} \quad (1)$$

The first term is the sum of the bag-of-words (BoW) reconstruction losses of each language in the corpus. We refer the reader to (Srivastava and Sutton, 2017) for further details on the reconstruction loss.

The second term is the sum of the KL divergences between the language-specific document distributions, $p()$ and $q()$, whose mean and variance are estimated from language-specific infer-

ence networks; θ refers to the sampled topic representation of a document in a tuple where i is the tuple index, a and b are the indices of the documents inside the tuple and n is the size of the tuple. Lastly, x refers to a document embedding.

The third term is the InfoNCE loss where $(\theta_i^a \cdot \theta_i^b)$ are positive pairs (they belong to the same tuple) and $(\theta_i^c \cdot \theta_j^d)$ are negative pairs (they are from different tuples). N is the batch size, τ is the temperature and s is a constant to give additional weight to the contrastive loss.

3.2 Extension to multimodal setting

We now extend the proposed multilingual topic model to the *multimodal* setting. Figure 1(b) shows the architecture of the proposed multilingual *and* multimodal topic model.

We can think of the multimodal case as a generalization of the multilingual model. The loss function in Equation 1 remains essentially the same. Since a BoW representation is not available for images, the reconstruction loss is computed only on texts and the first loss term is unchanged. In the second term of the loss function, x can be a document *or* image embedding and θ is the sampled topic distribution for that embedding.

Since the document or image embeddings abstract the modality of the data, the topic distributions are now modality-agnostic. Thus, the third term is also unchanged, except for the tuple size n. A multimodal dataset with one language and one image view would have N positive and $N^2 - N$ negative pairs, the same as in the bilingual case. For two languages *and* one image, we would have $3N$ positive pairs and $3(N^2 - N)$ negative pairs, as in the trilingual case.

We refer to our proposed topic model as **M3L-Contrast** for *multimodal multilingual (M3L) topic model with contrastive learning*.

4 Experimental Setup

4.1 Dataset

We run experiments on our proposed model on a dataset of aligned English and German Wikipedia articles and images. We take aligned articles from the Wikipedia Comparable Corpora[2] and align them with images from the Wikipedia-based Image Text dataset (WIT) (Srinivasan et al., 2021) [3]. We use articles instead of the image descriptions in WIT because topic models are designed for full documents, rather than snippets of text.

We randomly select 20,000 tuples for training. Since articles can be associated with more than one image and we want fixed-size tuples during training, we randomly select one image per article pair. For testing, we randomly select 1,000 article pairs. We consider *all* images aligned with the paired articles, which results in 3,278 unique tuples in the test set.

4.2 Evaluation

We evaluate M3L-Contrast in the multilingual setting and the multimodal setting, separately. In the multilingual case we train M3L-Contrast on multilingual articles *without* images and for the multimodal case on the multilingual articles *and* images.

We want document-topic distributions for multilingual articles and images from the same tuple to be similar to each other *and* distinct from other examples. Thus, we evaluate the alignment of topic distributions using retrieval tasks.[4]

Texts and images are fed one at a time to their own language-specific and modality-specific inference networks to obtain topic distributions. For the multilingual setting, we match an English article to the most similar German article in terms of the Jensen-Shannon divergence (JSD) between their respective document-topic distributions. For the multimodal setting, we match English articles to images and German articles to images, separately. We use mean reciprocal rank (MRR) to measure text retrieval performance and uninterpolated average precision (UAP, Manning and Schütze, 1999) to measure text-image retrieval performance because multiple images can be associated with an article.

We also report the averaged JSD between the topic distributions for all data pairs from the same tuple. Lastly, we compute language-specific topic coherences with respect to the training data using normalised pointwise mutual information (NPMI, Röder et al., 2015)[5].

[2] https://linguatools.org/tools/corpora/wikipedia-comparable-corpora/

[3] https://github.com/google-research-datasets/wit

[4] We are aware that topic distributions do not outperform raw embeddings in retrieval tasks but the point of this evaluation is not to improve cross-lingual or cross-modal retrieval but to evaluate the alignment of the topic distributions.

[5] Computed using the Gensim library (Rehurek and Sojka, 2010).

Model	Coherence↑ EN	DE	EN-DE text MRR↑	JSD↓
PLTM	0.064	0.044	0.333	0.067
ZeroshotTM	0.113	0.096	**0.997**	**0.012**
ZeroshotTM-KD	0.109	0.092	0.390	0.081
M3L-Contrast	**0.119**	**0.097**	0.684	0.036

Table 1: Language-specific topic coherences (NPMI) and cross-lingual retrieval peformance (MRR and JSD).

4.3 Baselines

PLTM (Mimno et al., 2009) We implement PLTM with Gibbs sampling.

ZeroshotTM (Bianchi et al., 2021b) We train separate models on the English and German articles using the authors' original implementation.[6]

ZeroshotTM-KD (Pivovarova and Zosa, 2021) We adapt ZeroshotTM for multilingual or multimodal settings using knowledge distillation (Hinton et al., 2015). This method uses the parameters learned by the teacher model as priors for the student. We train four separate teacher-student pairs: (1) Model trained on English articles as teacher, German as student; (2) German articles as teacher, English as student; (3) English articles as teacher, images as student; and (4) German articles as teacher, images as student.

4.4 Configurations

We report the performance of the neural topic models using CLIP (Radford et al., 2021) as our multimodal multilingual encoder for a fair comparison.[7] CLIP is a pretrained vision-language model trained on web-scale data that encodes text and images into a common embedding space. We train all models with 100 topics for 100 epochs. Other hyperparameters are discussed in the Appendix. We use batch size 32 for M3L-Contrast. In Section 6 we show the performance of M3L-Contrast for different encoder combinations (aligned and unaligned), different batch sizes and topic numbers.

5 Results and Discussion

5.1 Multilingual setting

Table 1 shows the cross-lingual retrieval performance and averaged JSD of aligned articles. ZeroshotTM is the clear winner with an MRR of 0.997 and the lowest JSD. M3L-Contrast, while it does not outperform ZeroshotTM, shows encouraging results given that it has to infer twice as many topics as ZeroshotTM (bilingual case). It also outperforms PLTM, a classical multilingual topic model and the only other model, aside from M3L-Contrast, trained on multilingual articles. Moreover, M3L-Contrast also has the best topic coherences.

5.2 Multimodal setting

Table 2 shows the results for text-image matching. M3L-Contrast performs the best with UAP of 0.125 and 0.102 for matching English and German articles to images, respectively, and has the lowest JSDs. In a reversal of the results for cross-lingual retrieval, ZeroshotTM performs the worst with the lowest UAP scores and highest JSDs. ZeroshotTM-KD only slightly outperforms ZeroshotTM, indicating that the success of M3L-Contrast can be attributed to *joint* training and cannot be achieved with the teacher-student sequential training scheme.

These results indicate that ZeroshotTM without any modifications is not suitable for multimodal settings. One likely reason is that multimodal encoders like CLIP suffer from the so-called 'modality gap' where embeddings for different modalities are mapped to separate regions in the embedding space (Liang et al., 2022).

Our results also indicate that for a joint multimodal and multilingual neural topic model, it could be beneficial to use a hybrid model that uses separate inference networks for different modalities and a shared network for the same modality. We leave this for future work.

5.3 Error analysis

To further investigate differences between the models we checked some examples from the test set. We show two article-image tuples and their predicted topics in Table 3. The table contains the top topic for the aligned English and German articles (titles shown) and an image associated with them.

The first example article is about pepper[8]. ZeroshotTM predicts relevant topics for the English and German articles but off-topic for the image. ZeroshotTM-KD (a teacher model trained on English articles and a student on images) predicts a

[6] https://github.com/MilaNLProc/contextualized-topic-models

[7] *clip-ViT-B-32* for images and *clip-ViT-B-32-multilingual-v1* for texts.

[8] https://en.wikipedia.org/wiki/Capsicum_pubescens

Model	Coherence↑ EN	DE	EN-images UAP↑	JSD↓	DE-images UAP↑	JSD↓
ZeroshotTM	0.113	0.096	0.034	0.445	0.039	0.435
ZeroshotTM-KD	0.109	0.092	0.082	0.128	0.093	0.146
M3L-Contrast	**0.122**	**0.097**	**0.125**	**0.130**	**0.102**	**0.147**

Table 2: Language-specific topic coherences (NPMI) and text-image retrieval performance (UAP). JSD is the averaged JS divergence between topic distributions of aligned articles and images. Only M3L-Contrast is jointly trained on **multilingual articles and images**.

		EN article	DE article	Image
Article title: **Capsicum pubescens**				
	ZeroshotTM	**21:** plant, leaves, flowers, tall	**31:** bird, south, america, species	**67:** university, library, museum, research
	ZeroshotTM-KD ***(EN teacher)***	**44:** plant, leaves, flowers, plants, genus	-	**44:** plant, leaves, flowers, plants, genus
	M3L-Contrast	**65:** plant, plants, leaves, flowers	**65:** beschreibung (*description*), pflanzen (*plant*), selten (*rare*), stehen (*stand*)	**65:** plant, plants, leaves, flowers
Article title: **Microexpression/Mikroexpression**				
	ZeroshotTM	**13**: include, cause, may, cases, occur	**13:** include, cause, may, cases, occur	**9:** bishop, catholic, pope, church, roman
	ZeroshotTM-KD ***(EN teacher)***	**32:** blood, symptoms, disease, cell, bone	-	**69:** album, released, song, single, group
	M3L-Contrast	**84:** theory, term, example, social, defined	**84:** begriff (*concept*), definition (*definition*), beispiel (*example*), theorie (*theory*), zahl (*number*)	**5:** film, award, series, actress, born

Table 3: Top topics of Wikipedia article pairs and a related image. The numbers indicate the topic indices.

relevant topic for the English article and the image but it has not been trained on German. M3L-Contrast predicts relevant topics for the English and German articles and the image. Though the table shows English topic labels for the image, it is equally possible to produce German image labels.

In the second example, the article about microexpressions[9] is illustrated with an image of a woman presenting basic emotions. ZeroshotTM predicts slightly relevant topics for the English and German articles but off-topic for the image. ZeroshotTM-KD also predicts a relevant topic for the English

[9] https://en.wikipedia.org/wiki/Microexpression

article. Although the image topic is different from the topic of the article, it is still somewhat relevant in that the model may have associated images of women with pop stars. M3L-Contrast predicts relevant topics for the English and German articles. For the image, it predicts a topic about actresses likely because the image depicts a woman.

We found similar behaviour in other cases: English and German articles are usually assigned with the same topic while the image often has a different topic. In many cases M3L-Contrast finds an aligned topic for an image while the other two models fail.

5.4 Visualizing the topic space

To investigate the structure of the multimodal multilingual topic space, we use 2D visualizations presented in Figures 2a and 2b, for ZeroshotTM and M3L-Contrast, respectively. These figures show the proximities of multilingual texts and images, represented by their predicted topic distributions, in the topic spaces induced by the respective models and mapped into two dimensions with tSNE.[10]

In Figure 2a—the ZeroshotTM topic space—the topic distributions of the aligned articles are very similar to their counterparts (most of the points representing English articles are hidden under the German articles). The images, however, tend to be isolated instead of being close to their textual counterparts. This supports the modality gap hypothesis and explains why ZeroshotTM performs poorly in the text-image retrieval task.

In Figure 2b, the topic space induced by M3L-Contrast, articles and images tend to group together in terms of *themes*—exactly the behaviour we want from a topic model. No single modality or language is isolated by itself. This explains why M3L-Contrast performs better than ZeroshotTM in text-image retrieval. On the other hand, the English and German articles are not as close to each other as in ZeroshotTM. This supports our claim that joint training takes into account data from all languages and adjusts for possible discrepancies between worldviews across languages, even though this property results in worse performance in cross-lingual text retrieval.

6 Ablation Study

6.1 Topic numbers

In Figure 3 we show the models' performance on cross-lingual text retrieval (MRR) for $[25, 50, 100]$ topics. ZeroshotTM performs best for all topic numbers followed by M3L-Contrast. Figure 4 shows the results for text-image retrieval (UAP). M3L-Contrast performs best for all topic numbers while ZeroshotTM performs worst. In general, performance improves as the topic number increases.

6.2 Batch sizes for M3L-Contrast

We check batch sizes $[16, 32, 64, 128]$. Figure 5 shows the effect of increasing batch sizes for M3L-Contrast trained only on multilingual articles. We find that batch size 32 is the best for the multilingual setting. We also run similar experiments with multilingual text *and* images (Figure 6). In the multimodal setting, size 32 performs the best when English articles are matched to images while 64 is best for German. This is why we used batch size 32 for our experiments with M3L-Contrast.

6.3 Encoder combinations for M3L-Contrast

To show how unaligned encoders perform with M3L-Contrast, we experiment with two encoder combinations: (1) a multilingual text encoder and an unaligned image encoder; and (2) unaligned monolingual text encoders and an unaligned image encoder. For this experiment, we train M3L-Contrast with multilingual articles *and* images and evaluate the models on cross-lingual retrieval and text-image retrieval. Encoder details are in the Appendix. Results are shown in Table 4.

For cross-lingual text retrieval, using a multilingual encoder performs best since this model is trained specifically on multilingual texts and has a larger embedding dimension than CLIP (768 and 512, respectively). For English text-image retrieval, it is expected that CLIP is the best since the text and image embeddings are already aligned (first row). CLIP image embeddings performed better than ResNet on all measures.

It is encouraging that M3L-Contrast with unaligned text and image embeddings still outperform ZeroshotTM and ZeroshotTM-KD (compare with Table 2, top part) even though those models use aligned embeddings. This shows that contrastive

[10]These figures are available as interactive plots in the code repository of this paper.

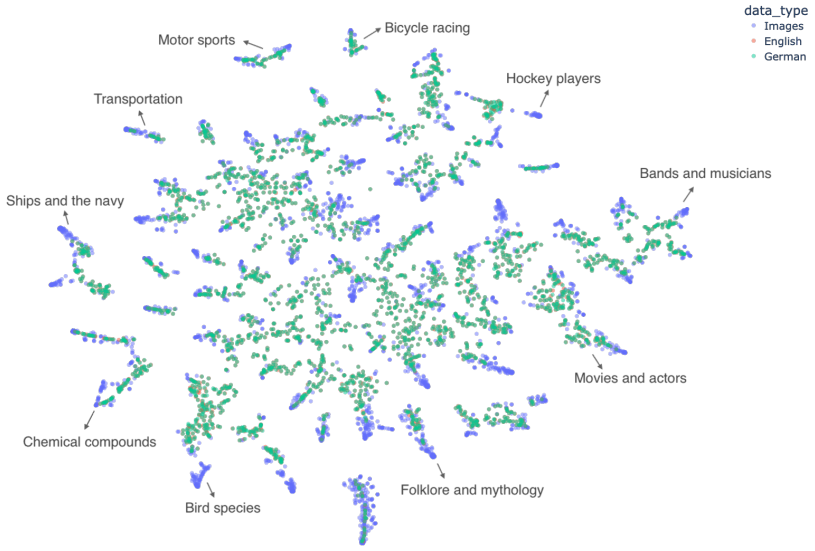

(a) ZeroshotTM

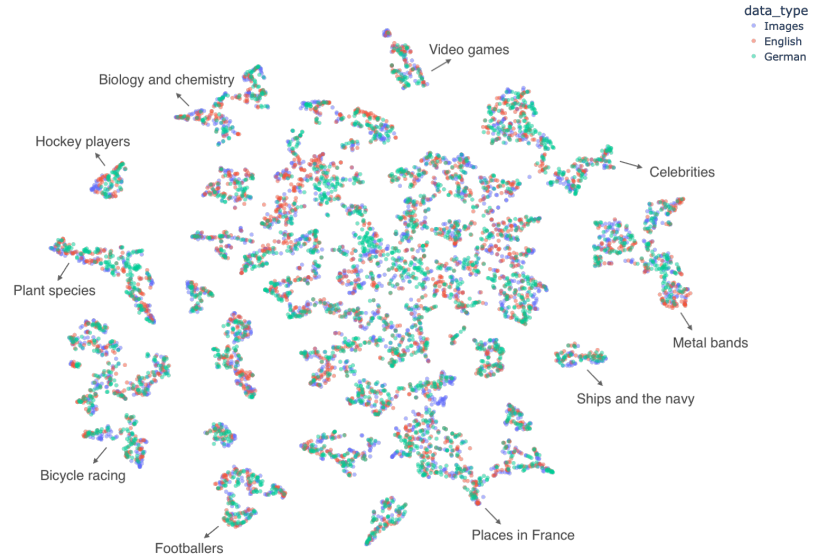

(b) M3L-Contrast

Figure 2: tSNE visualizations of topic distributions of multilingual texts and images inferred by ZeroshotTM and M3L-Contrast, respectively. Annotations are added manually. Best viewed in color.

learning is effective in mapping unanligned embeddings into a shared topic space and that it is not necessary to use aligned embeddings in multimodal topic modelling.

7 Conclusion

We presented M3L-Contrast, a multimodal and multilingual neural topic model based on ZeroshotTM that uses pretrained document and image embeddings. M3L-Contrast is trained jointly on multilingual texts and images and does not require aligned embeddings. Since it is a *multilingual* topic model it produces aligned language-specific topics. As a *multimodal* topic model, it maps texts and images into a shared topic space and infers textual representations, through the topic words, of the semantic concepts present in the images.

We show that in the multilingual setting, M3L-

Encoders		EN-DE text		EN-images		DE-images	
Text	**Image**	**MRR↑**	**JSD↓**	**UAP↑**	**JSD↓**	**UAP↑**	**JSD↓**
CLIP	CLIP	0.613	0.035	**0.125**	0.130	0.102	0.147
multilingual SBERT	CLIP	**0.716**	**0.029**	0.119	0.137	**0.114**	0.147
monolingual SBERTs	CLIP	0.407	0.052	0.118	**0.129**	0.102	**0.141**
multilingual SBERT	ResNet	0.659	0.028	0.053	0.160	0.047	0.167
monolingual SBERTs	ResNet	0.347	0.050	0.053	0.145	0.052	0.157

Table 4: Effect of different encoder combinations for M3L-Contrast trained on **multilingual text and images** compared to CLIP with 100 topics.

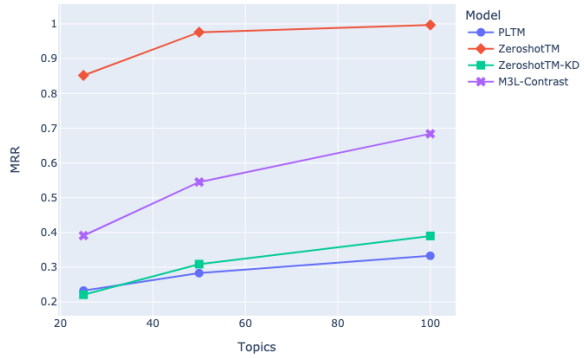

Figure 3: Effect of increasing topic numbers on cross-lingual retrieval performance (MRR).

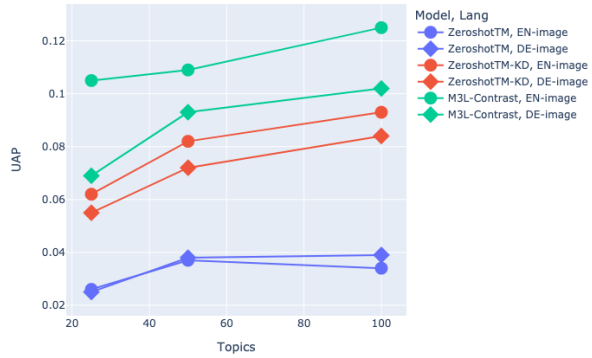

Figure 4: Effect of increasing topic numbers on text-image retrieval performance (UAP).

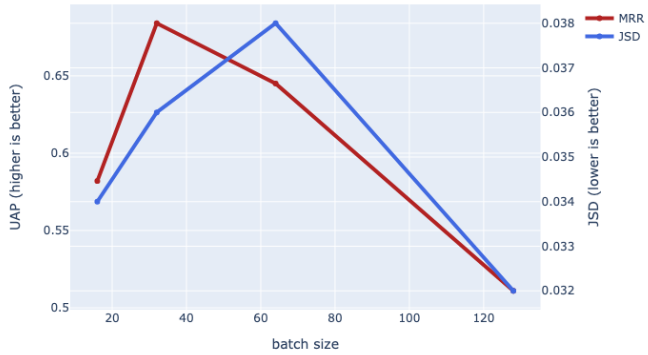

Figure 5: Effect of different batch sizes on M3L-Contrast trained on multilingual text data.

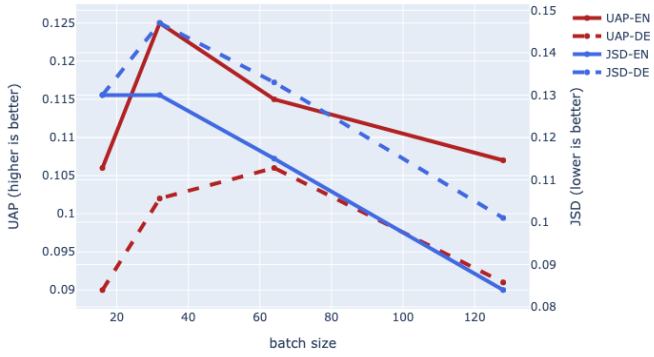

Figure 6: Effect of different batch sizes for M3L-Contrast trained on multilingual text *and* images.

Contrast improves on PLTM, a classical multilingual topic model, and that it is competitive with ZeroshotTM in the alignment of topic distributions for comparable documents in different languages. In the multimodal setting, our model significantly improves on ZeroshotTM in aligning comparable texts and images in the topic space. Moreover, with unaligned text and image embeddings our model still performs better than ZeroshotTM that uses aligned embeddings.

Our proposed architecture can easily be extended to include other modalities beyond image and text. We also believe that M3L-Contrast will be useful in a low-resource setting, where aligned embeddings can be difficult to obtain.

References

Minghui An, Jingjing Wang, Shoushan Li, and Guodong Zhou. 2020. Multimodal topic-enriched auxiliary learning for depression detection. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 1078–1089.

Kobus Barnard, Pinar Duygulu, David Forsyth, Nando De Freitas, David M Blei, and Michael I Jordan. 2003. Matching words and pictures.

Federico Bianchi, Silvia Terragni, and Dirk Hovy. 2021a. Pre-training is a hot topic: Contextualized document embeddings improve topic coherence. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 2: Short Papers)*, pages 759–766, Online. Association for Computational Linguistics.

Federico Bianchi, Silvia Terragni, Dirk Hovy, Debora Nozza, and Elisabetta Fersini. 2021b. Cross-lingual contextualized topic models with zero-shot learning. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*, pages 1676–1683, Online. Association for Computational Linguistics.

Abeba Birhane, Vinay Uday Prabhu, and Emmanuel Kahembwe. 2021. Multimodal datasets: misogyny, pornography, and malignant stereotypes. *arXiv preprint arXiv:2110.01963*.

David M Blei, Andrew Y Ng, and Michael I Jordan. 2003. Latent dirichlet allocation. *Journal of machine Learning research*, 3(Jan):993–1022.

Wim De Smet and Marie-Francine Moens. 2009. Cross-language linking of news stories on the web using interlingual topic modelling. In *Proceedings of the 2nd ACM workshop on Social web search and mining*, pages 57–64.

Adji B Dieng, Francisco JR Ruiz, and David M Blei. 2020. Topic modeling in embedding spaces. *Transactions of the Association for Computational Linguistics*, 8:439–453.

Yansong Feng and Mirella Lapata. 2010. Visual information in semantic representation. In *Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, pages 91–99.

Zhihui Guo, Pramod Sharma, Andy Martinez, Liang Du, and Robin Abraham. 2022. Multilingual molecular representation learning via contrastive pre-training. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 3441–3453.

E Dario Gutiérrez, Ekaterina Shutova, Patricia Lichtenstein, Gerard de Melo, and Luca Gilardi. 2016. Detecting cross-cultural differences using a multilingual topic model. *Transactions of the Association for Computational Linguistics*, 4:47–60.

Shudong Hao and Michael Paul. 2018. Learning multilingual topics from incomparable corpora. In *Proceedings of the 27th international conference on computational linguistics*, pages 2595–2609.

Shudong Hao and Michael Paul. 2020. An empirical study on crosslingual transfer in probabilistic topic models. *Computational Linguistics*, 46(1):95–134.

Geoffrey Hinton, Oriol Vinyals, and Jeff Dean. 2015. Distilling the knowledge in a neural network. In *NIPS Deep Learning and Representation Learning Workshop*.

Alexander Miserlis Hoyle, Pranav Goel, and Philip Resnik. 2020. Improving neural topic models using knowledge distillation. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1752–1771.

Jagadeesh Jagarlamudi and Hal Daumé. 2010. Extracting multilingual topics from unaligned comparable corpora. In *European Conference on Information Retrieval*, pages 444–456. Springer.

Ashish Jaiswal, Ashwin Ramesh Babu, Mohammad Zaki Zadeh, Debapriya Banerjee, and Fillia Makedon. 2021. A survey on contrastive self-supervised learning. *Technologies*, 9(1):2.

Chao Jia, Yinfei Yang, Ye Xia, Yi-Ting Chen, Zarana Parekh, Hieu Pham, Quoc Le, Yun-Hsuan Sung, Zhen Li, and Tom Duerig. 2021. Scaling up visual and vision-language representation learning with noisy text supervision. In *International Conference on Machine Learning*, pages 4904–4916. PMLR.

Khazar Khorrami and Okko Räsänen. 2021. Evaluation of audio-visual alignments in visually grounded speech models. In *Interspeech*, pages 1231–1235. International Speech Communication Association.

Weixin Liang, Yuhui Zhang, Yongchan Kwon, Serena Yeung, and James Zou. 2022. Mind the gap: Understanding the modality gap in multi-modal contrastive representation learning. *arXiv preprint arXiv:2203.02053*.

Xiao Liu, Fanjin Zhang, Zhenyu Hou, Li Mian, Zhaoyu Wang, Jing Zhang, and Jie Tang. 2021. Self-supervised learning: Generative or contrastive. *IEEE Transactions on Knowledge and Data Engineering*.

Christopher Manning and Hinrich Schütze. 1999. Topics in information retrieval. *Chapter 15 in Foundations of Statistical Natural Language Processing*.

David Mimno, Hanna Wallach, Jason Naradowsky, David A Smith, and Andrew McCallum. 2009. Polylingual topic models. In *Proceedings of the 2009 conference on empirical methods in natural language processing*, pages 880–889.

Aaron Mueller and Mark Dredze. 2021. Fine-tuning encoders for improved monolingual and zero-shot polylingual neural topic modeling. In *NAACL-HLT*.

Thong Nguyen and Anh Tuan Luu. 2021. Contrastive learning for neural topic model. *Advances in Neural Information Processing Systems*, 34.

Aaron Van den Oord, Yazhe Li, and Oriol Vinyals. 2018. Representation learning with contrastive predictive coding. *arXiv e-prints*, pages arXiv–1807.

Lidia Pivovarova and Elaine Zosa. 2021. Visual topic modelling for NewsImage task at MediaEval 2021. In *Working Notes Proceedings of the MediaEval 2021 Workshop*. MediaEval.

Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, et al. 2021. Learning transferable visual models from natural language supervision. In *International Conference on Machine Learning*, pages 8748–8763. PMLR.

Radim Rehurek and Petr Sojka. 2010. Software framework for topic modelling with large corpora. In *In Proceedings of the LREC 2010 workshop on new challenges for NLP frameworks*. Citeseer.

Nils Reimers and Iryna Gurevych. 2020. Making monolingual sentence embeddings multilingual using knowledge distillation. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 4512–4525, Online. Association for Computational Linguistics.

Michael Röder, Andreas Both, and Alexander Hinneburg. 2015. Exploring the space of topic coherence measures. In *Proceedings of the eighth ACM international conference on Web search and data mining*, pages 399–408.

Stephen Roller and Sabine Schulte Im Walde. 2013. A multimodal lda model integrating textual, cognitive and visual modalities. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*, pages 1146–1157.

Bei Shi, Wai Lam, Lidong Bing, and Yinqing Xu. 2016. Detecting common discussion topics across culture from news reader comments. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 676–685.

Krishna Srinivasan, Karthik Raman, Jiecao Chen, Michael Bendersky, and Marc Najork. 2021. WIT: Wikipedia-based image text dataset for multimodal multilingual machine learning. *arXiv preprint arXiv:2103.01913*.

Akash Srivastava and Charles Sutton. 2017. Autoencoding variational inference for topic models. In *5th International Conference on Learning Representations*.

Seppo Virtanen, Yangqing Jia, Arto Klami, and Trevor Darrell. 2012. Factorized multi-modal topic model. In *Proceedings of the 28th Conference on Uncertainty in Artificial Intelligence (UAI)*, page 843–851.

Rui Wang, Deyu Zhou, and Yulan He. 2019. Atm: Adversarial-neural topic model. *Information Processing & Management*, 56(6):102098.

Xiaobao Wu, Chunping Li, Yan Zhu, and Yishu Miao. 2020. Learning multilingual topics with neural variational inference. In *CCF International Conference on Natural Language Processing and Chinese Computing*, pages 840–851. Springer.

Weiwei Yang, Jordan Boyd-Graber, and Philip Resnik. 2019. A multilingual topic model for learning weighted topic links across corpora with low comparability. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 1243–1248.

Yuhao Zhang, Hang Jiang, Yasuhide Miura, Christopher D Manning, and Curtis P Langlotz. 2020. Contrastive learning of medical visual representations from paired images and text. *arXiv preprint arXiv:2010.00747*.

Yin Zheng, Yu-Jin Zhang, and Hugo Larochelle. 2014. Topic modeling of multimodal data: an autoregressive approach. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1370–1377.

Appendix

Data preprocessing

We follow the training data preprocessing of Bianchi et al. (2021b) for the BoW input: removing stopwords and retaining the 2000 most frequent words of each language as our vocabularies. We use the English and German stopword lists from NLTK[11].

Hyperparameters

The neural topic models are trained on a single Nvidia V100 GPU (35 minutes) while PLTM is trained on a single Intel Xeon CPU (3 hours). During testing, we averaged the inferred topic distributions for each article/image from 20 samples. For all the neural models we used Adam optimizer with a learning rate of 2^{-3}. We use a batch size of 64 except for M3L-Contrast. For M3L-Contrast, we set the temperature τ to 0.07 following (Guo et al., 2022). We set the contrastive weight s to 50 based on initial experiments. Tuning τ and s are saved for future work.

Inference network

We use the same inference network structure as ZeroshotTM (Bianchi et al., 2021b): one fully-connected hidden layer followed by softplus layer with 100 dimensions. We save the investigation of other inference network structures for future work.

Encoder Details

We use SentenceBERT to encode all our data (Reimers and Gurevych, 2020) [12]. For a fairer comparison, we set the maximum sequence length of all text encoders to 128 tokens. The multilingual text encoder is *paraphrase-multilingual-mpnet-base-v2*. For the monolingual encoders, the English encoder is *all-mpnet-base-v2* and the German encoder is *T-Systems-onsite/erman-roberta-sentence-transformer-v2*. ResNet embeddings are provided in this Kaggle challenge: https://www.kaggle.com/competitions/wikipedia-image-caption.

Potential impact and risks

Our models are currently for research purposes only. We do not advise that it be used in production settings. Our models might associate images of people and objects with negative and insensitive stereotypes if the training data has these associations. Since we use CLIP to encode texts and images in our experiments, our models might also perpetuate the harmful stereotypes found in the CLIP training data discussed in (Birhane et al., 2021). The same issue applies to the other pre-trained encoders we use in our experiments.

[11] https://www.nltk.org/

[12] https://www.sbert.net/docs/pretrained_models.html

Zero-Shot Script Parsing

Fangzhou Zhai, Vera Demberg and **Alexander Koller**
Dept. of Language Science and Technology
Saarland Informatics Campus
Saarland University
{fzhai, vera, koller}@coli.uni-saarland.de

Abstract

Script knowledge (Schank and Abelson, 1977) is useful for a variety of NLP tasks. However, existing resources only cover a small number of activities, limiting its practical usefulness. In this work, we propose a zero-shot learning approach to **script parsing**, the task of tagging texts with scenario-specific event and participant types, which enables us to acquire script knowledge without domain-specific annotations. We (1) learn representations of potential event and participant mentions by promoting class consistency according to the annotated data; (2) perform clustering on the event / participant candidates from unannotated texts that belongs to an unseen scenario. The model achieves 68.1/74.4 average F1 for event / participant parsing, respectively, outperforming a previous CRF model that, in contrast, has access to scenario-specific supervision. We also evaluate the model by testing on a different corpus, where it achieved 55.5/54.0 average F1 for event / participant parsing.

1 Introduction

Script knowledge is a type of commonsense knowledge that captures how people conduct everyday activities (Schank and Abelson, 1977). It expresses that in a certain **scenario**, **participants** tend to act out **events** in a certain order; an example from the scenario FIXING A FLAT TIRE is shown in Fig. 1. Humans use script knowledge to fill in events that are not explicitly mentioned in a text, and script knowledge is useful for many downstream NLP applications, including referent prediction (Ahrendt and Demberg, 2016; Modi et al., 2017), discourse classification (Lee et al., 2020), and story generation (Zhai et al., 2019, 2020).

A key challenge in dealing with script knowledge is coverage: it is costly and time-consuming to spell out the prototypical events and participants of a scenario and how they can be expressed in language. Existing resources are mostly crowd-sourced (Regneri et al., 2010; Modi et al., 2016); they annotate stories from a limited number of scenarios with script events and participants (cf. figure 1). **Script parsers**, which predict these event and participant labels given a text, can achieve high accuracies on scenarios that were seen in training (Ostermann et al., 2017; Zhai et al., 2021). Nonetheless, these parsers only operate on known scenarios: they could only predict event types on the same scenario as they were trained. The limited coverage restricts the practical usefulness of script parsers and thus the practical usefulness of script knowledge for downstream tasks in general.

In this paper, we acquire script knowledge by tackling the task of **zero-shot script parsing**: we present the first system which accurately performs

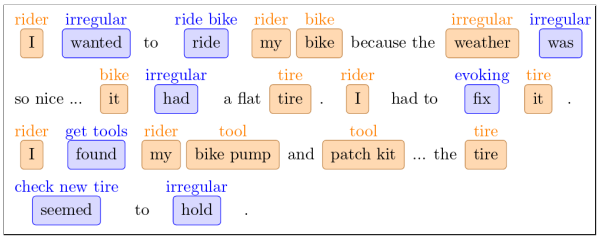

Figure 1: A story about FIXING A FLAT TIRE from InScript. Script parsing identifies events and participants from texts. The picture is taken from Zhai et al. (2021).

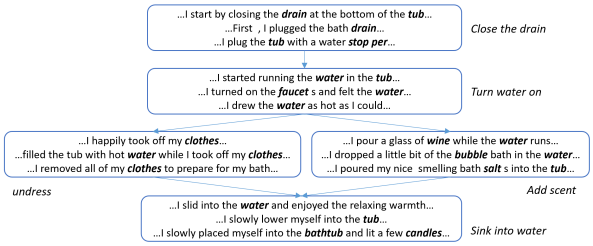

Figure 2: A part of the temporal script graph for TAKING A BATH inferred from our parsing result. Each node illustrates 3 random verbalizations from the cluster. The gold classes are shown on the side. Further edges that could be inferred by transitivity are omitted. We see one could either *undress* first or *add scent* (to the bath tub) first before *sink into water*.

Proceedings of the 29th International Conference on Computational Linguistics, pages 4049–4060
October 12–17, 2022.

script parsing on scenarios that were not seen at training time. For instance, given training data that talks about TAKING A BATH and GOING TO A RESTAURANT, the parser labels events and participants in the FIXING A FLAT TIRE story of Fig. 1, allowing us to arrange script knowledge into a graphical form as Fig. 2. This offers a way of overcoming the coverage limitations of script knowledge, by generalizing from the training scenarios to arbitrary other ones.

Our method learns to extract script-specific representations from general-purpose pretrained word embeddings, and then uses agglomerative clustering at inference time to group together natural-language phrases that refer to the same event or participant of the unseen script. Finally, we evaluate our model on MCScript, a different corpus than the training data.

Our model achieves a micro-F1 score on zero-shot event labeling of up to 68.1 and on participant labeling of up to 74.4, on par with the supervised model of Ostermann et al. (2017) that assumes training data for the same scenario. We find that our method yields script graphs with reasonable event clusters that are temporally ordered in a reasonable way; the majority of errors on event labeling are due to issues with the granularity of events. We also find in probing tasks that our model learns to amplify information about sentence ordering from the pretrained embeddings, while suppressing low-level information about morphology and syntax, which are less relevant for the script parsing task. In order to investigate its potential for practical use, we also evaluated the model on a different, unannotated corpus, where the model achieved $55.5/54.0$ average F1 for event / participant parsing.

2 Related work

Scripts were introduced as an approach to capturing commonsense knowledge in AI by Schank and Abelson (1977); see also Barr and Feigenbaum (1981). Much research in NLP has simplified the acquisition of script knowledge to identifying "event chains" in narrative text (Chambers and Jurafsky, 2008, 2009). Event chains represent typical sequences of events, each represented by one predicate, and can be learned from large corpora. Other work has followed in this tradition (Jans et al., 2012; Modi and Titov, 2014; Pichotta and Mooney, 2014; Rudinger et al., 2015).

In this paper, we instead build upon work by Regneri et al. (2010, 2011), who explicitly capture script knowledge about a given scenario in a *temporal script graph* (see Fig. 2). A TSG specifies the abstract events and participants that make up a script with their temporal ordering; each of these events and participants can be expressed in language in many different ways. A TSG is also more expressive than mere event sequences, because it encodes different manners how a scenario play out in real life. Regneri et al. learned script graphs by crowdsourcing. We instead rely on manually script-annotated corpora (Modi et al., 2016; Wanzare et al., 2016).

Trained with scenario-specific supervision, script parsing can be performed accurately. Ostermann et al. (2017) developed a linear CRF model to perform script parsing as a sequence labelling task. Zhai et al. (2021) developed a hierarchical model for supervised script parsing, making use of pre-trained contextualized word embeddings. The model learns patterns at the word level, as well as the narrative level with respective sequence models. These existing approaches are limited to scenarios for which training data is available, whereas our work focuses on unseen scenarios.

Zero-shot learning is a family of methods that establish a classifier for **unseen** classes, based on labelled data from **seen** classes. One common approach is to learn a latent representation space that all instances embed into, thus the knowledge of the source domain, encoded in the labelled training instances, could be transferred to the target domain. It tackles data scarcity in various situations, such as machine translation for low-resource languages (e.g. Pham et al., 2019; Zhang et al., 2020; Johnson et al., 2017), generation (Duan et al., 2019; Philip et al., 2020), text classification (see, e.g. Yin et al., 2019) and question answering (e.g. Banerjee and Baral, 2020).

3 Data

We work with **InScript** (Modi et al., 2016) and **MCScript** (Ostermann et al., 2018, 2019).

3.1 InScript

InScript is a crowdsourced corpus of around 100 stories about each of 10 scenarios (see Fig. 1 for an example). The authors were asked to write a story about a given scenario (such as GOING GROCERY SHOPPING) "as if to a child", step by step. InScript was then hand-annotated with event and

participant classes; it also contains coreference and dependency annotations.

3.2 Preprocessing

Following Ostermann et al. (2017), we distinguish between (1) events that are 'related to the scenario', or commonly seen in a typical instantiation of the scenario, which we call **regular events**, and (2) the ones that take place in the course of a specific story, but are not directly related to the scenario, which we call **irregular events**. For example, in Figure 1, *I found my bike pump* describes the regular event 'get tools', whereas *the weather **was** nice* is irregular. We collapse all the subclasses of irregular events[1] into a single irregular event class for each scenario. 12,902 (33.5%) event instances in InScript are regular. We also distinguish **regular participants** from **irregular participants** in a similar manner: participants like 'rain' in Fig. 1 are considered irregular to the FIXING A FLAT TIRE scenario, as they are not directly relevant to the scenario per se. Irregular participant instances take a smaller proportion of 19.6%.

3.3 MCScript

MCScript is a question answering dataset that focuses on script knowledge. Here we use its background text (not the questions), which comprises around 20 stories on each of 200 scenarios. These stories are not annotated with script events and participants; they are stylistically similar to those in InScript, as they also consist of relatively simple language and focus on explaining the scenario in detail.

We use MCScript to evaluate our model trained on InScript. To this end, we annotate 20 random scenarios from MCScript to specify script events and participants: 10 as validation set and 10 as test set. The set of labels we used is consistent with that of InScript, in that we adopt the same set of special labels (for example label **Unrel** for events and participants not relevant to the scenario). The annotation was performed by two experts. Similar preprocessing is performed to MCScript.

[1]These labels are UNREL (unrelated to the scenario), RELNSCR (related to the scenario but not a script event / participant), OTHER and UNKNOWN.

4 Zero-shot Script Parsing

4.1 Task Description

Our parser is tasked to predict event and participant annotations for a scenario that was not seen in training. Thus, our model must learn to group verbs and noun phrases from an unseen scenario into abstract events and participants, without knowing what the gold label set of events and participants are.

The basic idea of our zero-shot script parser is as follows. We will learn a transformation φ which maps pretrained general-purpose word embeddings into a representation space that is suitable for script parsing. Identifying verb tokens as *candidates* for event descriptions and noun and pronoun tokens as candidates for participant descriptions, we will train φ such that candidates that describe the same event or participant are close together in the representation space, whereas candidates for different events and participants are distant. To parse a text from an unseen scenario, we will apply φ to the word embeddings of all candidates and perform clustering to group them into events and participants.

We train and evaluate the model under a few different settings. **(1) InScript.** We split InScript into eight training, one validation, and one test scenario. During inference, the model takes the unannotated stories of the test scenario as input and labels them with events and participants that are consistent with the gold annotations. We rotate the validation / test scenario to perform a ten-fold cross-validation. **(2) MCScript.** We use all 10 scenarios in InScript as the training set, whereas validation and test are performed on the newly annotated MCScript validation set and test set.

4.2 Regular candidate identification

Throughout the paper, we will focus on *regular* candidates, because irregular candidates are not our primary goal, as the target of script acquisition is identifying candidates that describe the scenario, i.e. the **regular** candidates; furthermore, irregular candidates make a diverse group of instances without much semantic similarity to each other. Thus they would not cluster easily in the representation space. Therefore, we ignore irregular candidates in training. During inference, we evaluate against the original gold standard.

We train a classifier to distinguish regular and irregular candidates so the latter could be excluded from training. The classifier is also used in the test

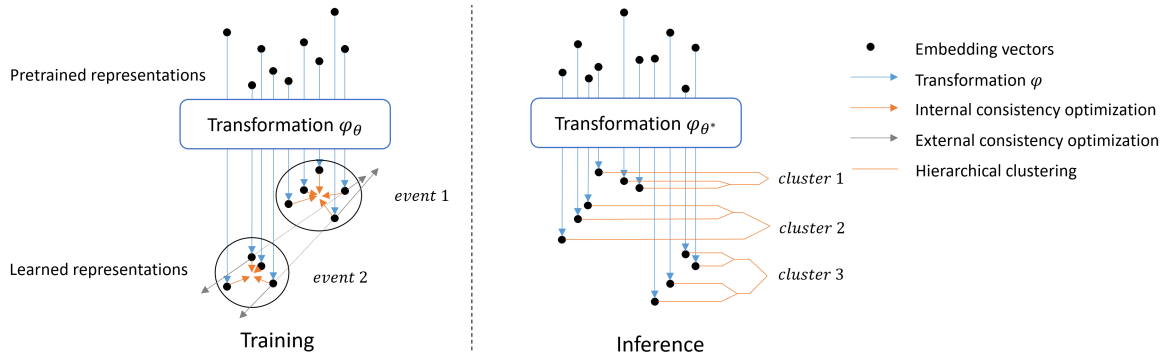

Figure 3: The overall framework. We learn a representation from annotated corpus and apply it to unannotated texts. The coreference and dependency terms are not visualized.

phase to exclude irregular candidates from clustering. We use the supervised script parser proposed in Zhai et al. (2021), but train it only to distinguish regular candidates from irregular candidates. We obtain training data for this task by grouping the original labels into one of REGULAR_EVENT, IRREGULAR_EVENT, REGULAR_PARTICIPANT and IRREGULAR_PARTICIPANT. The train / validation set of this model is constructed from the respective training set specified for each settings in §4.1. These classifiers achieve on average 85 points F1-score.

4.3 Training

We now describe how to learn φ. For any given text that we want to parse, we run XLNet (Yang et al., 2019) to obtain contextualized word embeddings $f(c)$ for each event and participant candidate c. We also considered using BERT (Kenton and Toutanova, 2019) / ROBERTA (Liu et al., 2019), but our input length exceeds the 512 word-piece limit hardwired in the pre-trained versions of these models.

Afterwards, we train φ to minimize distances within the same event and participant class and maximize them between different ones (§4.3.1); the general framework is illustrated in Fig. 3. We will then describe several extensions to the loss function (§4.3.2–§4.3.3) and then discuss replacing XLNet embeddings with more specialized word embeddings (§4.3.4)[2].

4.3.1 Learning script-neutral representations

Let $\mathcal{C}$ be the set of all event candidates or the set of all participant candidates in a text, and let $\pi(\mathcal{C})$ be a partition of $\mathcal{C}$ which clusters candidates into equivalence classes; at training time, each class contains the candidates that are labeled with the same event or participants. We define $\pi(c)$ as the element of the partition to which the candidate c belongs. Given a pre-trained embedding function f and the transformation φ_θ that we want to learn, we consider the average distance between instances belonging to different clusters:

$$d_{ext}(\pi(\mathcal{C});\theta) = \underset{\substack{c,c' \in \mathcal{C}: \\ \pi(c) \neq \pi(c')}}{mean} d(\varphi_\theta(f(c)), \varphi_\theta(f(c')))$$

The transformation φ is implemented by fine-tuning the last layer of the encoder.

We would like to push the embeddings of two candidates from different classes apart if they are too close to each other. We do so by maximizing the *external consistency* of the partition π:

$$\gamma_{ext}(\pi(\mathcal{C});\theta) = \underset{\substack{c,c': \pi(c) \neq \pi(c'), \\ d(\varphi_\theta(f(c)), \varphi_\theta(f(c'))) \\ < \sigma_1 d_{ext}(\pi(\mathcal{C});\theta)}}{mean} d(\varphi_\theta(f(c)), \varphi_\theta(f(c')))$$

$\sigma_1 \in (0, 1)$ is a threshold that quantifies being 'too close'. This definition captures the intuition that φ should map candidates from different classes to dissimilar vectors.

Likewise, consider the average distances between embeddings of candidates from same classes:

$$d_{int}(\pi(\mathcal{C});\theta) = \underset{c,c': \pi(c)=\pi(c')}{mean} d(\varphi_\theta(f(c)), \varphi_\theta(f(c')))$$

We would like to pull the embeddings of two candidates from the same class towards each other if they are too far away. In a similar spirit, we maximize the *internal consistency* of π:

[2]Our code and data are available at https://github.com/coli-saar/A3_USSP_coling22

(a) There is a **bus** stop down the street from my house . If you take **it** going south , **it** leads to the city...

(b) ...$I_{passenger}$ **fed** my *coins*$_{money}$ into the slot where you put your money...
...$I_{passenger}$ boarded the bus and **paid** for my ride with my *change*$_{money}$...

(c) ...the bus *arrived*$_{bus\_stops}$ at the **bus stop** closest to the beach...
...I would need the bus to *stop*$_{bus\_stops}$ next to **the hospital**...

Figure 4: Examples.(a) coreference chain. (b) events sharing similar participants. (c) participants that have similar event dependents.

$$\gamma_{int}(\pi(\mathcal{C});\theta) = 1 - \underset{\substack{c,c':\pi(c)=\pi(c'),\\ d(\varphi_\theta(f(c)),\varphi_\theta(f(c')))\\ >\sigma_2 d_{int}(\pi;\theta)}}{mean} d(\varphi_\theta(f(c)),\varphi_\theta(f(c'))) \quad (1)$$

We write $d(\cdot,\cdot)$ for the distance function in the representation space. Empirically, the following variant of cosine distance worked well:

$$\sqrt{1 - cos(\angle(v,w))}$$

Here $\angle(v,w)$ denotes the angle between v and w.

We obtain an overall consistency measure γ, which we maximize in training; λ_i is a hyperparameter that balances the terms. [3]

$$\gamma(\pi(\mathcal{C});\theta) = \gamma_{ext}(\pi(\mathcal{C});\theta) + \lambda_i \gamma_{int}(\pi(\mathcal{C});\theta)$$

4.3.2 Coreference

We can now further refine this baseline consistency model with script-specific knowledge. First, within a text, noun phrases that refer to the same entity form a **coreference chain**: for example, all mentions of the bus in the scenario TAKING A BUS (Fig. 4a). Therefore, these noun phrases should belong to the same participant cluster and have similar representations.

We capture this intuition as follows. Let $\eta(\mathcal{C}^c)$ be the set of all coreference chains that consists of participant candidates $\mathcal{C}^c$. Like π above, $\eta(\mathcal{C}^c)$ also specifies an equivalence relation, in that two candidates are in the same class iff they are in the same coreference chain. We can thus formulate a coreference-based consistency measure as

$$\beta(\theta) := \gamma_{int}(\eta(\mathcal{C}^c);\theta)$$

Note that minimizing this quantity imposes only a soft constraint; coreferent entities are rewarded for being in the same class, not forced into them. This increases robustness against noise in the coreference annotations.

4.3.3 Event-participant dependencies

Second, events and participants in a script are tightly linked: if two verbs have arguments from the same participant class, they tend to describe the same event (Fig. 4b); and if two noun phrases are arguments of the same event, they tend to describe the same participant (Fig. 4c).

Let c_p be the set of event candidates that have participant p as an argument; we encourage φ to map the elements of c_p to similar representations. Let $\xi(\mathcal{C}_e^d)$ be the set of all c_p, namely a partition of event candidates based on dependencies. Analogously, let $\xi(\mathcal{C}_p^d)$ be the set of participant candidate sets that depend on the same events. We can formulate a dependency-based consistency measure as

$$\alpha(\theta) = \gamma_{int}(\xi(\mathcal{C}_e^d);\theta) + \gamma_{int}(\xi(\mathcal{C}_p^d);\theta)$$

The final training objective, with hyperparameters λ_c, λ_d and cluster assignment $\pi^*(\mathcal{C}^e)$ of event candidates and $\pi^*(\mathcal{C}^p)$ of participant candidates specified by the annotations in InScript, is [4]

$$\theta^* = \underset{\theta}{argmax}[\gamma(\pi^*(\mathcal{C}_e^d);\theta) + \lambda_p \gamma(\pi^*(\mathcal{C}_p^d);\theta) + \lambda_c \beta(\theta) + \lambda_d \alpha(\theta)] \quad (2)$$

4.3.4 Specialized word embeddings

We further investigated whether our zero-shot approach can benefit by using more specialized word embeddings as input instead of the general-purpose XLNet embeddings. We thus replaced f with the representations from the pre-final layer of the supervised script parser of Zhai et al. (2021) trained and validated on our training data. These representations are also based on XLNet, but then trained to

[3] This function could be seen as a variant of triplet loss introduced by (Dong and Shen, 2018), but relaxed with the thresholds and uses a different distance metric. Empirically, these measures improve the stability of training and result in a moderate performance improvement.

[4] As was correctly pointed out by one of our reviewers, untyped dependencies can be noisy for our purpose. We do not use typed dependencies as typing the links amplifies the data sparsity, making it difficult to generalize. Empirically, using untyped dependencies still granted a moderate performance improvement.

predict InScript events and participants on known scenarios.

4.4 Inference

4.4.1 Clustering

At inference time, we first determine the event and participant candidates by taking the nouns, pronouns and verbs, and classify them for regularity. We then acquire a representation $\varphi_{\theta^*}(f(c))$ for each candidate c and group them into classes by clustering (cf. Fig. 3).

We use agglomerative clustering, a bottom-up hierarchical clustering algorithm that iteratively merges the most similar pair of clusters. It terminates when either the number of clusters decreases to a pre-defined quantity or the minimum dis-similarity between the current clusters goes beyond a predefined threshold. As the number of event and participant classes vary across scenarios, we do not fix the number of cluster, but instead define a dissimilarity threshold estimated from the training scenarios.

4.4.2 Protagonists

We give special treatment to the **protagonist** of each scenario – for example, the passenger in TAKING A TRAIN or the customer in GROCERY SHOPPING. The protagonist is the most frequent participant in all scenarios and always makes the largest class of participant candidates. We thus identify it by following the longest coreference chain, instead of feeding these referents to the neural pipeline. Therefore, the identification of protagonists is excluded from training and performed symbolically in the reference phase. This simple heuristic yields an F-score of 98 at inference time for the protagonist class.

4.4.3 Coreference Chains at Inference Time

During the training phase we encourage the embedding vectors associated with candidates from the same coreference chain to be more similar to each other. But during the inference phase, we could also directly perform coreference resolution on the text. We take into account this information by refining the distance in §4.3 to

$$\sqrt{1 - cos(\angle(v, w)) - \lambda I_c(v, w)}$$

here $I_c(v, w) = 1 \Leftrightarrow$ the candidates associated with vectors v, w are in the same coreference chain. λ is a hyper-parameter.

5 Evaluation

On InScript, we evaluate our method with 10-fold cross-validation where each fold is the data associated with a scenario. Note that the texts in the validation and test data are always from scenarios that were unseen in training. On MCScript, the model is evaluated on the MCScript test set.

5.1 Metric

Given a cluster assignment, what we are interested in is how well the predicted clusters align with gold classes. However, the gold classes are unknown to us; the outcome of clustering includes a number of indexed clusters, like $cluster_1, cluster_2$, and we do not know which gold cluster should these clusters be compared to. To tackle this issue, we need a 'best' assignment of the clusters to the gold classes, with which we can evaluate the 'accuracy' of the clustering results as if it were a classification task. One approach is to find the assignment that maximizes this accuracy. This is a *linear assignment* problem, which is solved in cubic time by the Hungarian algorithm (see, e.g. Kuhn, 1955), thus tractable given the scale of our problem. We call the F1 score evaluated according to this optimal assignment **Hungarian F1**, and use it as our main evaluation metric. This metric allows us to compare the results of the clustering-based parsers to that of the classification-based parsers.

5.2 Baselines

We compare the results of our zero-shot parser to a number of baselines. First, we compare against the supervised script parsers of Zhai et al. (2021) and Ostermann et al. (2017). For the former, we take the performance report from the original paper; for the latter, we retrain the model to evaluate on the train-test split of Zhai et al. This data split defines a supervised task, thus the performance of these parsers are not directly comparable to ours.

Second, we compare against a baseline where we cluster event and participant candidates at inference time based on the bare XLNet embeddings, rather than the ones that were transformed by our learned φ_{θ^*}. Finally, in addition to our *full* model, as specified by equation 2 with the specialized embeddings of §4.3.4, we also present results for ablated versions without the extensions regarding event-participant dependencies (*dep*), coreference (*coref*), and specialized embeddings (*specialized*).

model	gold regularity	task	events		participants	
			macro F1	micro F1	macro F1	micro F1
Ostermann et al. (2017)	✓	supervised	58.1	66.0	n/a	n/a
Zhai et al. (2021)	X	supervised	75.1	85.7	80.3	90.3
Bare XLNet	X	zero-shot	40.2	53.2	39.3	60.5
w/o dep, coref, specialized	X	zero-shot	$46.0_{\pm 2.8}$	$58.4_{\pm 2.7}$	$47.5_{\pm 2.6}$	$\mathbf{75.7}_{\pm 1.8}$
w/o dep, coref	X	zero-shot	$48.6_{\pm 5.2}$	$62.7_{\pm 3.7}$	$44.5_{\pm 3.3}$	$71.8_{\pm 2.1}$
w/o dep	X	zero-shot	$51.0_{\pm 3.7}$	$66.8_{\pm 4.3}$	$\mathbf{52.0}_{\pm 3.7}$	$74.8_{\pm 2.9}$
Full model	X	zero-shot	$\mathbf{53.4}_{\pm 1.8}$	$\mathbf{68.1}_{\pm 2.3}$	$51.7_{\pm 1.6}$	$74.4_{\pm 1.4}$
Bare XLNet	✓	zero-shot	43.1	51.6	43.9	61.0
w/o dep, coref, specialized	✓	zero-shot	$46.1_{\pm 1.9}$	$55.4_{\pm 2.2}$	$51.1_{\pm 2.7}$	$\mathbf{75.3}_{\pm 1.3}$
w/o dep, coref	✓	zero-shot	$55.3_{\pm 2.8}$	$65.8_{\pm 2.8}$	$52.5_{\pm 3.1}$	$73.6_{\pm 2.1}$
w/o dep	✓	zero-shot	$56.7_{\pm 3.3}$	$67.4_{\pm 3.7}$	$\mathbf{53.6}_{\pm 2.9}$	$74.2_{\pm 2.0}$
Full model	✓	zero-shot	$\mathbf{57.6}_{\pm 1.3}$	$\mathbf{68.1}_{\pm 1.3}$	$52.8_{\pm 1.4}$	$73.7_{\pm 1.4}$

Table 1: Results on InScript. We show the average over ten-fold cross validation and five training runs when feasible. These quantities are the Hungarian versions of F1 defined in §5.1. Some models train and inference according to the regularity annotations in InScript, instead of the predictions of our regular candidate identifier. Ostermann et al. and Zhai et al. use a data split where the models see the test scenario during training; the other variants use the zero-shot split described in §3.

model	gold regularity	events		participants	
		macro F1	micro F1	macro F1	micro F1
Bare XLNet	X	24.1	25.0	17.8	19.5
w/o dep, coref, specialized	X	$44.2_{\pm 0.57}$	$51.2_{\pm 0.83}$	$37.0_{\pm 1.4}$	$39.0_{\pm 0.72}$
w/o dep, coref	X	$45.7_{\pm 0.63}$	$50.2_{\pm 0.63}$	$37.9_{\pm 1.1}$	$41.9_{\pm 1.0}$
w/o dep	X	$48.2_{\pm 1.4}$	$53.3_{\pm 1.3}$	$40.8_{\pm 1.8}$	$45.4_{\pm 1.7}$
Full model	X	$\mathbf{49.6}_{\pm 1.3}$	$\mathbf{55.5}_{\pm 1.6}$	$\mathbf{42.6}_{\pm 1.6}$	$\mathbf{54.0}_{\pm 2.0}$

Table 2: Results on MCScript averaged from five parallel training runs. All models train and inference based on the regularity predictions of our regular candidate identifier.

For each of the clustering-based methods, we report two results: one where we assume gold information about whether an event or participant candidate is regular, and one where this is predicted by the classifier from §4.2. All variants use the same number of trials for hyperparameter tuning. Afterwards, we do 5 parallel training sessions to test the models' robustness against random initializations, and report mean and std.

6 Results

6.1 Evaluation on InScript

The results on InScript are shown in Table 1. All variants of our model outperform clustering based on raw XLNet embeddings by a considerable margin. Our model also performs on par with Ostermann's, although we do not have access to scenario-specific supervision whereas Ostermann's does, and our model additionally performs participant parsing. In general, we obtain a higher micro-F1 for participants than for events. This is due to the more skewed distribution of the sizes of the participant class sizes than those of the event classes.

The model extensions boost parsing accuracy significantly. Access to coreference information improves participant parsing performance. Dependency information grants a performance boost in event parsing. A closer inspection shows that with dependency information, the parser is better at grouping together event candidates with different verbs but same arguments. For example, event *sink into water* in TAKING A BATH could be evoked by *slide into water*, *sink into water*, *slip into the tub*, *lower into the tub*, etc. The verbs in these event candidates all share arguments *I* and *water* or *tub*, which our parser correctly clusters together. Without dependency information, the parser mostly groups together candidates whose predicate is 'sink', the most frequent verbalization of the event.

The performance of our script parser differs from fold to fold: we get 70.1 micro-F1 for participant parsing on TAKING A BATH, but only 43.8 on BORROWING A BOOK FROM LIBRARY. These differ-

ences result from two factors. (1) Similarities between the training scenarios and the test scenario. (2) Differences in the qualities of the original annotation among different scenarios.

6.2 Evaluation on MCScript

The results on MCScript is given in table 2. Note that the test scenarios only have around 20 stories each, as opposed to the 100 stories available for each InScript scenario. Shifting to a different domain and having fewer stories per scenario available incurred a considerable performance drop (on average over 10 points F1 score). The performance drop is especially noticeable for participant parsing, which was unexpected. A closer look indicates that participants in MCScript scenarios follow much more skewed distributions than in InScript: instances from the less frequent half of the participant classes in MCScript take a proportion of only 8% whereas this quantity is 18% for InScript. As a result, there are more classes that only have a handful of instances than InScript, worsening the already low-resource setting.

All our model variants still outperform the XLNet baseline by a large margin, with the full model achieving the best performance, at 49.6/55.5 macro/micro F1 for events, and 42.6/54.0 macro/micro F1 for participants. Each model component contributes to this performance according to the ablation study.

7 Further analysis

7.1 Temporal script graphs

Given the clustering results, we can induce temporal script graphs for unseen scenarios. We establish temporal order as follows: we say event e_1 **precedes** event e_2 if, and only if, in stories where they both occur, the proportion where e_1 occurs before e_2 is beyond a threshold ζ. If neither e_1 precedes e_2 nor e_2 precedes e_1, we decide they could follow arbitrary order.

If we view the construction of temporal script graphs as a task of retrieving temporally ordered event pairs, and evaluate our results against that inferred from annotations in InScript, our clustering results yields 75 points F1 score. Observe that the model has learned that each event can be expressed in many different ways that are semantically similar only in the context of the scenario (Fig. 2).

7.2 Probing Script-specific Embeddings

We conjectured that the transformation φ was needed to distill the relevant information for script parsing out of the pretrained XLNet embeddings. We investigate whether this is true by freezing the embeddings $\varphi(f(c))$ (zero-shot) and the pretrained embeddings $f(c)$ (XLNet) and training models for a variety of NLP tasks that take these embeddings as input.

We probe with the following tasks. (1) POS tagging and (2) named entity recognition; these mostly depend on the token itself and its local context. (3) Noun phrase chunking, which is determined by sentence-level syntax. (4) Sentence ordering, where we randomly shuffle the order of the sentences in a story and train a binary classifier to detect whether the story is shuffled. The task would need information across the entire story to conduct. (1)-(3) are formulated as sequence labelling tasks; (4) is a binary classification. The experiments are conducted on InScript, with the same data split as is used to train our representation. InScript includes POS annotations; for NER and chunking, the labels are generated with Spacy (Honnibal and Montani, 2017, model *en_core_web_trf*).

In each of these probing tasks, both representations use the same amount of GPU budget. See Fig. 5 for the results. The transformed representation $\varphi(f(c))$ incurs performance drops on most tasks, compared to the general-purpose embeddings $f(c)$. However, the performance on sentence ordering sees a noticable improvement. This supports our hypothesis that φ amplifies higher level features, which are more important to script parsing than to generic language modelling. In comparison, lower-level information about morphology and syntax are deemphasized.

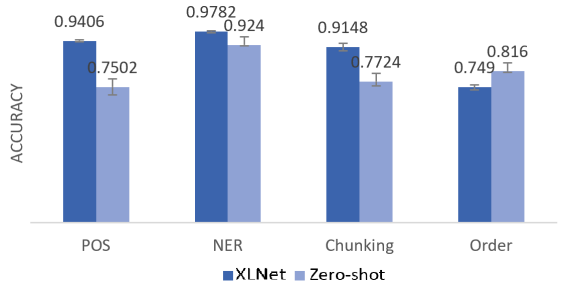

Figure 5: Performance on probing tasks. Our representation clearly favours the sentence ordering task. The error bars show one standard deviation. All differences between these pairs are significant at $\alpha = 0.05$ according to independent T-test.

7.3 Error analysis

We now have a look at the errors made by our parser. The errors fall into the following categories.

Granularity

Many events could be divided into multiple subevents, forming a hierarchy of events. For example, in the TAKING A BATH scenario, we have *prepare for bath*, *undress* and *grab a towel*. A similar phenomenon is observed for the participants. This fact manifested itself into various types of errors. For example, the set of event labels in InScript often consists of events of different granularities, frequently rendering multiple cluster assignments feasible (e.g. ... *I **took** a clean towel with me* ... in either *prepare for bath* or *grab a towel*). As a result, the parser sometimes confuses one event cluster with another that includes it, or group together different events that actually fit together (*turn on water* and *fill tub with water*). Granularity accounts for two thirds the event errors and one sixth participant errors.

Shared predicate or argument

Some wrongly clustered events share the verb or some arguments with another class, especially when light verbs are involved, which makes the distinction harder. For example, in TAKING A TRAIN, a few *get ticket* events (e.g. "I took the ticket from him") are predicted as *conductor checks ticket* (e.g. "I gave the ticket to him").

8 Conclusion

We have presented the first approach to script parsing without scenario-specific knowledge. We do this by clustering specialized word representations which are trained by optimizing cluster consistency; the model is further improved by the use of coreference and event-participant dependency information. The model greatly outperforms a baseline with general-purpose word embeddings, and performs on par with an earlier supervised model.

Our model thus makes it possible, for the first time, to label large quantities of unannotated data with script information. In future work, we plan to experiment also with corpora that contain a larger variety of naturally occurring texts than MCScript, in which the sentences are relatively simple, pertinent to the scenario and are often in the correct temporal order.

Acknowledgement

This research was funded by the German Research Foundation (DFG) as part of SFB 1102 (Project-ID 232722074) "Information Density and Linguistic Encoding".

References

Simon Ahrendt and Vera Demberg. 2016. Improving event prediction by representing script participants. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 546–551.

Pratyay Banerjee and Chitta Baral. 2020. Self-supervised knowledge triplet learning for zero-shot question answering. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 151–162, Online. Association for Computational Linguistics.

Avron Barr and Edward Feigenbaum. 1981. *The Handbook of Artificial Intelligence: Volume 2*. William Kaufman Inc, Los Altos, CA.

James Bergstra and Yoshua Bengio. 2012. Random search for hyper-parameter optimization. *Journal of Machine Learning Research*, 13(Feb):281–305.

Nathanael Chambers and Dan Jurafsky. 2008. Unsupervised learning of narrative event chains. *Proceedings of ACL-08: HLT*, pages 789–797.

Nathanael Chambers and Dan Jurafsky. 2009. Unsupervised learning of narrative schemas and their participants. In *Proceedings of the Joint Conference of the 47th Annual Meeting of the ACL and the 4th International Joint Conference on Natural Language Processing of the AFNLP: Volume 2-Volume 2*, pages 602–610. Association for Computational Linguistics.

Xingping Dong and Jianbing Shen. 2018. Triplet loss in siamese network for object tracking. In *Proceedings of the European conference on computer vision (ECCV)*, pages 459–474.

Xiangyu Duan, Mingming Yin, Min Zhang, Boxing Chen, and Weihua Luo. 2019. Zero-shot cross-lingual abstractive sentence summarization through teaching generation and attention. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 3162–3172, Florence, Italy. Association for Computational Linguistics.

Matt Gardner, Joel Grus, Mark Neumann, Oyvind Tafjord, Pradeep Dasigi, Nelson F. Liu, Matthew Peters, Michael Schmitz, and Luke S. Zettlemoyer. 2017. Allennlp: A deep semantic natural language processing platform.

Matthew Honnibal and Ines Montani. 2017. spaCy 2: Natural language understanding with Bloom embeddings, convolutional neural networks and incremental parsing. To appear.

Bram Jans, Steven Bethard, Ivan Vulić, and Marie Francine Moens. 2012. Skip n-grams and ranking functions for predicting script events. In *Proceedings of the 13th Conference of the European Chapter of the Association for Computational Linguistics*, pages 336–344, Avignon, France. Association for Computational Linguistics.

Melvin Johnson, Mike Schuster, Quoc V. Le, Maxim Krikun, Yonghui Wu, Zhifeng Chen, Nikhil Thorat, Fernanda Viégas, Martin Wattenberg, Greg Corrado, Macduff Hughes, and Jeffrey Dean. 2017. Google's multilingual neural machine translation system: Enabling zero-shot translation. *Transactions of the Association for Computational Linguistics*, 5:339–351.

Jacob Devlin Ming-Wei Chang Kenton and Lee Kristina Toutanova. 2019. Bert: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of NAACL-HLT*, pages 4171–4186.

Diederik P Kingma and Jimmy Ba. 2015. Adam: A method for stochastic optimization. In *ICLR (Poster)*.

Harold W Kuhn. 1955. The hungarian method for the assignment problem. *Naval research logistics quarterly*, 2(1-2):83–97.

I-Ta Lee, Maria Leonor Pacheco, and Dan Goldwasser. 2020. Weakly-supervised modeling of contextualized event embedding for discourse relations. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 4962–4972, Online. Association for Computational Linguistics.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Roberta: A robustly optimized bert pretraining approach.

Ashutosh Modi, Tatjana Anikina, Simon Ostermann, and Manfred Pinkal. 2016. Inscript: Narrative texts annotated with script information. In *Proceedings of the Tenth International Conference on Language Resources and Evaluation (LREC'16)*, pages 3485–3493.

Ashutosh Modi and Ivan Titov. 2014. Inducing neural models of script knowledge. In *Proceedings of the Eighteenth Conference on Computational Natural Language Learning*, pages 49–57, Ann Arbor, Michigan. Association for Computational Linguistics.

Ashutosh Modi, Ivan Titov, Vera Demberg, Asad Sayeed, and Manfred Pinkal. 2017. Modeling semantic expectation: Using script knowledge for referent prediction. *Transactions of the Association for Computational Linguistics*, 5:31–44.

Simon Ostermann, Ashutosh Modi, Michael Roth, Stefan Thater, and Manfred Pinkal. 2018. Mcscript: A novel dataset for assessing machine comprehension using script knowledge. In *Proceedings of the Eleventh International Conference on Language Resources and Evaluation (LREC 2018)*.

Simon Ostermann, Michael Roth, and Manfred Pinkal. 2019. Mcscript2. 0: A machine comprehension corpus focused on script events and participants. In *Proceedings of the Eighth Joint Conference on Lexical and Computational Semantics (* SEM 2019)*, pages 103–117.

Simon Ostermann, Michael Roth, Stefan Thater, and Manfred Pinkal. 2017. Aligning script events with narrative texts. In *Proceedings of the 6th Joint Conference on Lexical and Computational Semantics (* SEM 2017)*, pages 128–134.

Ngoc-Quan Pham, Jan Niehues, Thanh-Le Ha, and Alexander Waibel. 2019. Improving zero-shot translation with language-independent constraints. In *Proceedings of the Fourth Conference on Machine Translation (Volume 1: Research Papers)*, pages 13–23, Florence, Italy. Association for Computational Linguistics.

Jerin Philip, Alexandre Berard, Matthias Gallé, and Laurent Besacier. 2020. Monolingual adapters for zero-shot neural machine translation. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 4465–4470, Online. Association for Computational Linguistics.

Karl Pichotta and Raymond Mooney. 2014. Statistical script learning with multi-argument events. In *Proceedings of the 14th Conference of the European Chapter of the Association for Computational Linguistics*, pages 220–229, Gothenburg, Sweden. Association for Computational Linguistics.

Michaela Regneri, Alexander Koller, and Manfred Pinkal. 2010. Learning script knowledge with web experiments. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics*, pages 979–988, Uppsala, Sweden. Association for Computational Linguistics.

Michaela Regneri, Alexander Koller, Josef Ruppenhofer, and Manfred Pinkal. 2011. Learning script participants from unlabeled data. In *Proceedings of the International Conference Recent Advances in Natural Language Processing 2011*, pages 463–470.

Rachel Rudinger, Pushpendre Rastogi, Francis Ferraro, and Benjamin Van Durme. 2015. Script induction as language modeling. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 1681–1686, Lisbon, Portugal. Association for Computational Linguistics.

Roger C Schank and Robert P Abelson. 1977. *Scripts, plans, goals, and understanding: An inquiry into human knowledge structures*. Psychology Press.

Lilian DA Wanzare, Alessandra Zarcone, Stefan Thater, and Manfred Pinkal. 2016. Descript: A crowdsourced database of event sequence descriptions for the acquisition of high-quality script knowledge. In *Proceedings of the Tenth International Conference on Language Resources and Evaluation (LREC'16)*, pages 3494–3501.

Zhilin Yang, Zihang Dai, Yiming Yang, Jaime Carbonell, Russ R Salakhutdinov, and Quoc V Le. 2019. Xlnet: Generalized autoregressive pretraining for language understanding. In *Advances in neural information processing systems*, pages 5753–5763.

Wenpeng Yin, Jamaal Hay, and Dan Roth. 2019. Benchmarking zero-shot text classification: Datasets, evaluation and entailment approach. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 3914–3923, Hong Kong, China. Association for Computational Linguistics.

Fangzhou Zhai, Vera Demberg, and Alexander Koller. 2020. Story generation with rich details. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 2346–2351, Barcelona, Spain (Online). International Committee on Computational Linguistics.

Fangzhou Zhai, Vera Demberg, Pavel Shkadzko, Wei Shi, and Asad Sayeed. 2019. A hybrid model for globally coherent story generation. In *Proceedings of the Second Workshop on Storytelling*, pages 34–45, Florence, Italy. Association for Computational Linguistics.

Fangzhou Zhai, Iza Škrjanec, and Alexander Koller. 2021. Script parsing with hierarchical sequence modelling. In *Proceedings of* SEM 2021: The Tenth Joint Conference on Lexical and Computational Semantics*, pages 195–201.

Biao Zhang, Philip Williams, Ivan Titov, and Rico Sennrich. 2020. Improving massively multilingual neural machine translation and zero-shot translation. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 1628–1639, Online. Association for Computational Linguistics.

A Implementation and Optimization

The model was implemented with AllenNLP 1.2 Gardner et al. (2017). The pre-trained XLNet model we used was *xlnet-base-cased* (`https://github.com/zihangdai/xlnet/`). The training is further regularized with weight decay. The optimization is performed with adam (Kingma and Ba, 2015) in conjunction with early-stopping which monitors validation loss; the hyper-parameter tuning is performed with random hyper-parameter search (Bergstra and Bengio, 2012). Optimization takes on average 5 hours on a singe Tesla v100.

The implementations of agglomerative clustering and Hungarian algorithm are from the *scipy* library. Table 3 shows the hyper-parameters for the best performing single run on each fold (full model, predicted regularity).

B Examples

Table 4 illustrates a couple of sample clusters. The candidates vary in their surface forms.

fold	lr	weight decay	λ_{ei}	λ_{pi}	λ_p	σ_1	σ_2	λ_c	λ_d
0	1.10E-05	0.000155	1.03	0.0147	0.343	0.887	0.263	0.00372	0.00795
1	2.06E-05	0.00677	0.457	0.882	0.125	0.392	0.391	0.0101	0.0485
2	0.000126	0.000896	0.0105	0.0148	0.0144	0.903	0.728	0.00533	0.0118
3	2.91E-05	0.000166	0.0237	0.0649	0.256	0.373	0.544	0.0619	0.00867
4	7.08E-05	0.00051	0.15	0.0711	0.745	0.681	0.562	0.0121	0.0216
5	0.000353	0.00121	1.51	0.0371	0.00211	0.171	0.692	0.0872	0.016
6	0.000423	1.07E-05	0.204	0.018	0.177	0.113	0.632	0.242	0.00517
7	7.88E-05	0.00303	0.734	0.0136	0.517	0.189	0.948	0.0047	0.111
8	2.72E-05	7.32E-05	0.568	0.00151	0.0589	0.676	0.566	0.00424	0.0475
9	2.09E-05	5.19E-05	0.0327	0.555	0.243	0.788	0.95	0.0866	0.0119

Table 3: Hyper-parameters. Note that these are the hyper-parameter combinations that yielded the best performance in a round of random hyper-parameter search. Thus the quantities in this table do not represent the best choices of each single hyper-parameter.

ground truth	text
turn water on bath	... I might drain the tub and **put** in more water ...
sink into water	... I turn off the faucet and **sink** into bliss ...
sink into water	... Then I **slid** into the water and enjoyed the relaxing warmth for twenty or more minutes...
sink into water	... I gingerly **lowered** myself into the nice warm water and immediately began to relax...
sink into water	... I eased my way into the tub and let myself **sink** into the water ...
sink into water	... I slowly **sunk** the rest of my body , and closed my eyes...
sink into water	... the tub was full and ready . I **slipped** into the tub and soaked in the bliss...
washing tools	...then I lather up with either **soap** or shower gel ...
water	...After I scrub really good and finish singing , I pour **water** continuously on my body...
washing tools	...I pour water continuously on my body until all the **soap** was he s off...
washing tools	...I cleaned my hair with some **shampoo** and washed my body with a wash cloth and rinsed...
washing tools	...shampooed my hair and applied some **conditioner** then washed my body...
washing tools	...applied some condition er then washed my body using some **liquid body wash**...
washing tools	...After I have washed everything , I rinse the **soap** from my body with the water in the tub...
washing tools	...on the corner of the bathtub . I lather ed **it** up and washed my arms , my legs...
washing tools	...take a wash cloth and **soap** or body wash to give yourself a good scrub down...
washing tools	...You can put the **shampoo** in your hair...
washing tools	...place your head under the faucet to rinse out the **soap** . Enjoy your bath !
washing tools	...washed myself with a **wash cloth** and soap . Then I leaned my head against...
washing tools	...stepped into the bath tub . I used **soap** and a wash cloth to clean myself...

Table 4: Example output clusters. Top: event; bottom: participant. The table presents a random selection of instances from these clusters as the original output could contain hundreds of instances.

Word Sense Disambiguation with Knowledge-Enhanced and Local Self-Attention-based Extractive Sense Comprehension

Guobiao Zhang[1] , Wenpeng Lu[1,*], Xueping Peng[2] , Shoujin Wang[3] , Baoshuo Kan[1] , Rui Yu[1]
[1]School of Computer, Qilu University of Technology (Shandong Academy of Sciences), Jinan, China
[2]Australian Artificial Intelligence Institute, University of Technology Sydney, Sydney, Australia
[3]Data Science Institute, University of Technology Sydney, Sydney, Australia
guobiao.zhang@foxmail.com, lwp@qlu.edu.cn
{xueping.peng,shoujin.wang}@uts.edu.au
10431200583@stu.qlu.edu.cn, rui.yu1996@foxmail.com

Abstract

Word sense disambiguation (WSD), identifying the most suitable meaning of ambiguous words in the given contexts according to a predefined sense inventory, is one of the most classical and challenging tasks in natural language processing. Reformulating WSD as a text span extraction task is an effective approach, which accepts a sentence context of an ambiguous word together with all definitions of its candidate senses simultaneously, and requires to extract the text span corresponding with the right sense. However, the approach merely depends on a short definition to learn sense representation, which neglects abundant semantic knowledge from related senses and leads to data-inefficient learning and suboptimal WSD performance. To address the limitations, we propose a novel WSD method with **K**nowledge-**E**nhanced and **L**ocal self-attention-based **E**xtractive **S**ense **C**omprehension (KELESC). Specifically, a knowledge-enhanced method is proposed to enrich semantic representation by incorporating additional examples and definitions of the related senses in WordNet. Then, in order to avoid the huge computing complexity induced by the additional information, a local self-attention mechanism is utilized to constrain attention to be local, which allows longer input texts without large-scale computing burdens. Extensive experimental results demonstrate that KELESC achieves better performance than baseline models on public benchmark datasets.[1]

1 Introduction

Word sense disambiguation (WSD) is to identify a proper sense with an ambiguous word in a given context according to a predefined sense inventory, which is one of the most typical and challenging tasks in natural language processing (NLP) and play a critical role for human language understanding (Conia and Navigli, 2021). For instance, the noun word *plant* conveys different senses in *industrial plant* and *plant seeds*. WSD has been able to determine accurate meanings of ambiguous words, which is beneficial to a variety of downstream NLP applications, such as machine translation, information extraction and retrieval (Song et al., 2021; Pasini and Navigli, 2020).

In recent years, with the rapid development of deep learning, the performance of WSD with neural networks-based methods has great improvement. The early neural networks-based models have cast WSD as a multi-label classification task, which disambiguated all polysemous words with a unified classier (Kågebäck and Salomonsson, 2016; Raganato et al., 2017a). However, these models have focused on modeling contexts containing ambiguous words from sense-labeled training data, which ignored the rich semantic knowledge in lexical resources, such as WordNet and BabelNet (Navigli et al., 2021), and resulted in their inability to outperform the traditional *word expert* supervised methods (Song et al., 2021).

Due to the semantic knowledge in a lexical dictionary including sense definitions (glosses), examples, relations, etc., defined by professional lexicographers, which is essential and valuable for WSD, some works (Banerjee and Pedersen, 2002; Basile et al., 2014) have attempted to integrate gloss information into neural WSD models in order to leverage the lexical knowledge. GAS (Luo et al., 2018) has incorporated glosses into WSD, which jointly encoded glosses and contexts, and captured their relations with a memory network. GlossBERT (Huang et al., 2019) has utilized glosses in WordNet together with the annotated data to construct *context-gloss* pairs, reformulated WSD as a text matching task. BEM (Blevins and Zettlemoyer, 2020) has been a bi-encoder method that encodes the target word and candidate glosses inde-

*Corresponding author
[1]The source code of this paper can be obtained from https://github.com/Stubborn-z/KELESC

Proceedings of the 29th International Conference on Computational Linguistics, pages 4061–4070
October 12–17, 2022.

pendently and optimizes the encoders in the same representation space. Although these works considered the gloss information in WordNet, they neglected to explore the knowledge contained in semantic relations such as hypernyms. Therefore, EWISER (Bevilacqua and Navigli, 2020) has been proposed to enhance WSD by integrated synset embeddings and semantic relations including hypernyms and hyponyms. ESR (Song et al., 2021) has further enhanced sense representations by incorporating synonyms, example sentences and sense glosses of hypernyms.

Although the methods mentioned above have achieved great successes, they have treated WSD as multi-label classification or text matching tasks, which focused on modeling the relations between a context and each specified candidate sense. None of them considers all candidate senses of an ambiguous word simultaneously, which is not consistent with human behaviors, as humans always justify the right sense by comparing all possible senses with the context. In order to simulate the cognitive process of human, ESC (Barba et al., 2021) has reformulated WSD as a text span extraction task, called extractive sense comprehension, which accepted a context of an ambiguous word together with all definitions of its candidate senses. Although ESC demonstrated the superiority over the competitive methods, it merely relied on a short definition (gloss) to represent a sense, which was insufficient to learn an ideal sense representation and inevitably hinder the improvement of WSD performance.

To address the above-mentioned limitations, we propose a novel WSD method with **K**nowledge-**E**nhanced and **L**ocal self-attention-based **E**xtractive **S**ense **C**omprehension (KELESC), inspired by ESR (Song et al., 2021). Specifically, a knowledge-enhanced method is proposed to enrich semantic representation by incorporating additional examples and definitions of the related senses in WordNet. Then, in order to avoid the huge computing complexity induced by the additional information, a local self-attention mechanism is utilized to constrain attentions to be local, which allows longer input texts without large-scale computing burdens (Beltagy et al., 2020; Manakul and Gales, 2021). Extensive experiments have been conducted to verify the effectiveness of the proposed model on public benchmark datasets. In summary, this paper makes the following contributions:

- We propose a novel end-to-end WSD model with Knowledge-Enhanced and Local self-attention-based Extractive Sense Comprehension (KELESC). The model reformulates WSD as a text extraction task, fully utilizes lexical knowledge to enhance sense representation, and considers all candidate senses simultaneously instead of one by one to identify the right sense.

- We devise a knowledge enhancement method to enrich semantic representation by incorporating additional sense information of related senses in WordNet. Besides, we exploit a local self-attention mechanism to reduce the computing burden of training the model.

- Extensive experiments are conducted on public datasets to demonstrate the superiority of our proposed model on all-words English WSD tasks by making comparisons with the baseline models.

2 Related Work

The existing works on WSD can be categorized into three groups: knowledge-based, supervised and neural-based methods.

2.1 Knowledge-based WSD methods

These methods focus on leveraging semantic knowledge contained in lexical resources to identify the right sense (Luo et al., 2018). They mainly exploit two kinds of knowledge: sense definitions and structure of semantic network. For sense definitions' knowledge, Lesk algorithm and its variants are the typical works, which select the right sense according to the overlap of contexts and sense definitions (Lesk, 1986). For structure knowledge of semantic network, Personalized PageRank (Agirre et al., 2014; Scozzafava et al., 2020), BabelNet (Navigli and Ponzetto, 2012) and structural semantic interconnections (Navigli and Velardi, 2005) are the representative methods, which construct semantic graphs with senses and their relations, and utilize graph-based algorithms to choose the most important sense as the right one. With the support of lexical knowledge, knowledge-based methods achieve satisfied WSD coverage while their accuracy usually is worse than the others.

2.2 Traditional supervised WSD methods

These methods utilize manually feature engineering to train a special classifier for each polysemous word, i.e., *word expert*. IMS system first proposes instance and feature extractors to extract instances and their features, then trains an independent classifier for each word type on sense-annotated SemCor corpus (Zhong and Ng, 2010). Iacobacci et al. (2016) investigate how word embeddings has been utilized in WSD, which combines word embeddings and traditional manual features to enhance the original IMS system. Although these traditional supervised methods show better performance on WSD, they are confused by the manually engineered features and sense-annotated training dataset. Besides, they train a dedicated classifier for each word, which are hard to be applied on all-word WSD tasks.

2.3 Neural-based WSD methods

These approaches usually train a unified classifier based on neural networks to disambiguate all of the polysemous words. The early neural-based models mainly focus on modeling the relations of sentence context and sense labels contained in training datasets. For example, (Kågebäck and Salomonsson, 2016) and (Raganato et al., 2017a) employ bidirectional LSTM and encoder-decoder architecture to train unified models for all-word WSD tasks. However, they neglect to utilize the valuable semantic knowledge contained in lexical resources such as WordNet. Thus, The GAS model attempts to incorporate gloss information into an end-to-end WSD model (Luo et al., 2018). GlossBERT (Huang et al., 2019) also leverages glosses in WordNet to construct *context-gloss* pairs, reformulates WSD as a text matching task to model the matching relations of sense glosses and the contexts of ambiguous words. BEM (Blevins and Zettlemoyer, 2020) proposes a jointly optimized bi-encoder (the context encoder and the gloss encoder), which encode the context and sense glosses, and choose the nearest sense with the context according to gloss and context embeddings. However, these methods merely utilize the gloss information in WordNet, which still ignore the semantic relation knowledge such as hypernyms. Therefore, EWISER (Bevilacqua and Navigli, 2020) is proposed to integrate sense embeddings together with hypernyms and hyponyms relations to enhance WSD performance. And, ESR (Song et al., 2021) further incorporates synonyms, example sentences and sense glosses of hypernyms to enhance sense representations. All methods mentioned above focus on modeling relations between a context and each specified candidate sense individually, while human usually determines the sense by comparing all possible senses with a context simultaneously. This means that there are still some room to improve the neural-based WSD methods. In order to simulate the cognitive progress of human, that is, to comparing all candidate senses simultaneously, ESC (Barba et al., 2021) reformulates WSD as a text span extraction task, which accepts a context of an ambiguous word together with the definitions of all possible senses, and choose the text span of the right sense by comparing all sense definitions at once. ESC has shown the superiority on WSD task, however, it merely utilizes a short definition to learn a sense, which is insufficient.

3 Methodology

In this section, we first give the task definition. Then, we detail our proposed model, **K**nowledge-**E**nhanced and **L**ocal self-attention-based **E**xtractive **S**ense **C**omprehension (KELESC) for WSD.

3.1 Task Definition

Given the context with target word with glosses, example sentences and hypernym glosses of all candidate senses, the task of the paper is to identify the text span that indicates the right sense. Specifically, we represent the context of target word $\hat{w}$ as $C = \{w_1^c, \cdots, w_m^c\}$, where m is the number of words in the context. For the k-th candidate sense of the target word $\hat{w}$, its gloss, example sentence and hypernym gloss are represented as $G^k = \{w_1^{gk}, \cdots, w_{|gk|}^{gk}\}$, $ES^k = \{w_1^{ek}, \cdots, w_{|ek|}^{ek}\}$, and $HG^k = \{w_1^{hk}, \cdots, w_{|hk|}^{hk}\}$. $|gk|$, $|ek|$ and $|hk|$ indicate their lengths. Given the concatenation of the context C and the information G, ES, HG of all candidate senses, our model will identify the interval $[i_{cor}, j_{cor}]$, which indicates the start and end positions of the text span corresponding with the gloss, example sentence and hypernym gloss of the right sense of $\hat{w}$.

3.2 Model Architecture

The overall structure of KELESC model is shown in Figure 1. KELESC model consists of three core modules: (1) a knowledge enhancement module,

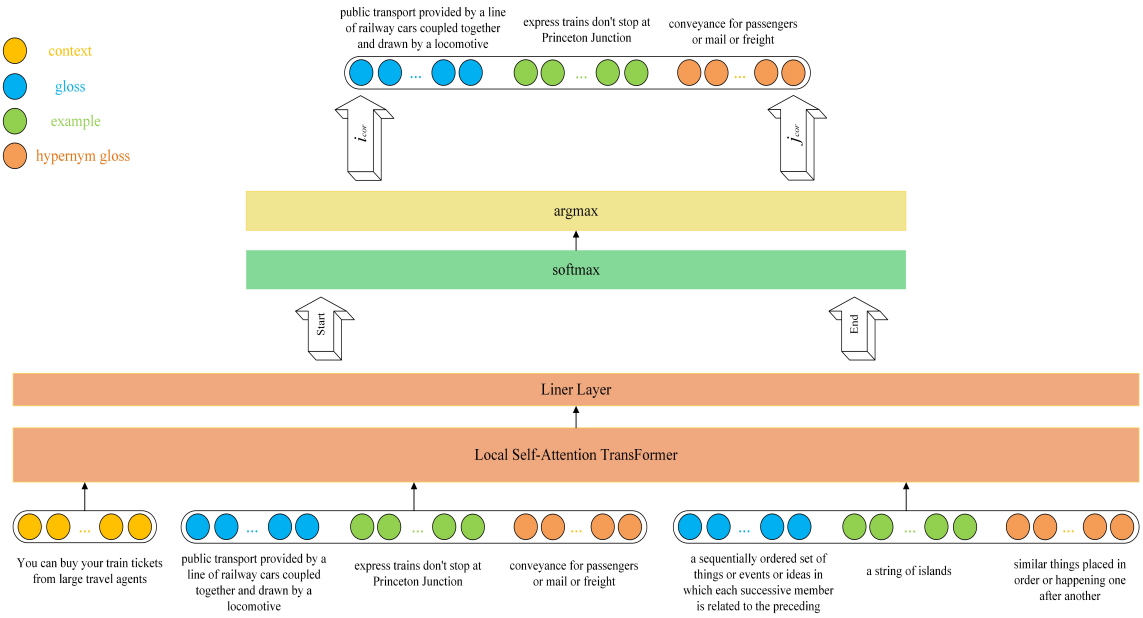

Figure 1: Overview structure of KELESC model. Concatenate the context of target word together with the gloss, example sentence, hypernym gloss of each candidate sense as the input of our model. $Start$ and End represent the logits for each word, which indicates whether it is the start or end of the text span of the right sense of the target word, respectively. i_{cor} and j_{cor} is the start and end indices of the correct sense, respectively.

Context Sentence		You can buy your *train* tickets from large travel agents.
Sense#1	Gloss	public transport provided by a line of railway cars coupled together and drawn by a locomotive.
	Example sentence	express trains don't stop at Princeton Junction.
	Hypernym gloss	conveyance for passengers or mail or freight.
Sense#2	Gloss	a sequentially ordered set of things or events or ideas in which each successive member is related to the preceding.
	Example sentence	train of mourners.
	Hypernym gloss	similar things placed in order or happening one after another.
Context and enhanced knowledge		You can buy your train tickets from large travel agents. public transport provided by a line of railway cars coupled together and drawn by a locomotive. express trains don't stop at Princeton Junction. a sequentially ordered set of things or events or ideas in which each successive member is related to the preceding. a string of islands. similar things placed in order or happening one after another.

Table 1: An example of knowledge enhancement of the target word *train*.

which utilizes gloss, example sentence and hypernym gloss to enhance the representation of each candidate sense, (2) a local self-attention transformer, which encodes the entire input texts with local self-attention transformer, (3) a span prediction module, which extracts the text span with the highest probability of expressing the correct sense of the target word.

3.2.1 Knowledge Enhancement

Recent studies have shown that lexical knowledge in WordNet is essential and valuable for accurate sense representation learning (Song et al., 2021). To this end, KELESC model devises a knowledge enhancement module to explore and integrate the richer lexical knowledge. As shown in Table 1, for each candidate sense of a target word, the module collects its gloss, example sentence and hypernym glosses together to enrich sense representation. Specifically, the gloss is a short definition of the current sense in WordNet. The example sentence is a sentence instance that conveys the corresponding sense. The hypernym gloss refers to the sense definition of the hypernym synsets of the current sense, which describes high-level semantic information. The original context and all candidate senses of the the target word with its glosses, example sentences and hypernym glosses are concatenated together, which is fed into our model.

3.2.2 Local Self-Attention Transformer

In order to effectively encode the input text, we adopt the pre-trained transformer-based model, i.e., BART$_{large}$, as it works well on long sequence modeling and comprehension tasks (Lewis et al., 2020; Beltagy et al., 2020). As shown in Figure 1 and Table 1, we concatenate the gloss, example sentence and hypernym gloss of each candidate sense together for a target word, marked as $A = \{G^1, ES^1, HG^1, \cdots, G^{|s|}, ES^{|s|}, HG^{|s|}\}$, where $|s|$ is the number of candidate senses of the target word. The context sentence and enhanced knowledge are fed into the transformer as the input. There could be some exceptions: if there is no example sentence in WordNet, we ignore it; if there are multiple example sentences, we only select the first one.

Specifically, we use the tags $< s >$ and $< /s >$ to surround the entire input sequence. The context sentence C and the enhanced lexical knowledge A are segmented by the special symbol $< /d >$ and the target word $\hat{w}$ is surrounded by $< t >$ and $< /t >$. The entire input of the transformer is as follows:

$$
\begin{aligned}
input = & < s > w_1^c \cdots < t > \hat{w} < /t > \cdots w_m^c \\
& < /d > w_1^{g1} \cdots w_{|g1|}^{g1} w_1^{e1} \cdots w_{|e1|}^{e1} w_1^{h1} \cdots w_{|h1|}^{h1} \cdots \\
& w_1^{gn} \cdots w_{|gn|}^{gn} w_1^{en} \cdots w_{|en|}^{en} w_1^{hn} \cdots w_{|hn|}^{hn} < /s >
\end{aligned}
$$

where $input$ is tokenized as $T = \{t_1, t_2, \cdots, t_n\}$, and n denotes the length of input token sequence.

The sense representation is enriched by our proposed knowledge enhancement module. However, it inevitably results in the longer sequence of input text, which the memory requirement and computing complexity of the transformer-based model could be quadratic with the length of input sequence. It increases the huge burden during the model training. To alleviate this problem, we introduce a local self-attention mechanism proposed by (Manakul and Gales, 2021). It is noteworthy that the mechanism in KELESC focuses on the encoder part. Our local self-attention transformer-based module adopts a fixed window $\widehat{w}$ around each token which only focuses on the ones lying in the window on each side. As shown in Figure 2, we set a window size to 1 as a toy example to show the local self-attention mechanism in an encoding layer. The outputs of encoding layer in local self-attention transformer are calculated as bellow:

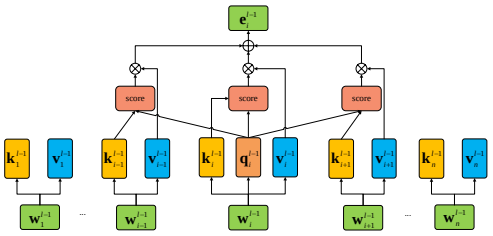

Figure 2: Local self-attention with a window size $\widehat{w}$ of 1, $\mathbf{w}_i^{l-1}$ represents the embedding of the i-th token w_i generated by the previous encoding layer $(l-1)$. $\mathbf{k}_i^{l-1}$, $\mathbf{v}_i^{l-1}$ and $\mathbf{q}_i^{l-1}$ represent the vector of key, value and query, respectively. $\mathbf{w}_i^l$ is the embedding of w_i obtained with local self-attention mechanism in the current layer (l).

$$
\mathbf{w}_i^l = \text{sum}\left(\text{softmax}\left(\frac{\mathbf{Q}_i^{l-1}{\mathbf{K}_i^{l-1}}^\top}{\sqrt{\mathbf{d}_k}}\right)\mathbf{v}_i^{l-1}\right)
$$

where $\mathbf{Q}_i^{l-1}=[\mathbf{q}_i^{l-1}]_{2*\widehat{w}+1}$ is the local query matrix, $\mathbf{K}_i^{l-1}=[\mathbf{k}_{i-\widehat{w}}^{l-1},\cdots,\mathbf{k}_{i-1}^{l-1},\mathbf{k}_i^{l-1},\mathbf{k}_{i+1}^{l-1},\cdots,\mathbf{k}_{i+\widehat{w}}^{l-1}]$ is the local key matrix, and $\mathbf{v}_i^{l-1}=[\mathbf{v}_{i-\widehat{w}}^{l-1},\cdots,\mathbf{v}_{i-1}^{l-1},\mathbf{v}_i^{l-1},\mathbf{v}_{i+1}^{l-1},\cdots,\mathbf{v}_{i+\widehat{w}}^{l-1}]$ is the local value matrix. $\mathbf{d}_k$ is the dimension of the embedding vector. By stacking multiple layers of this local self-attention transformer, a large receptive range will be obtained, in which the top layer can access all input positions and has the ability to build a representation containing the whole input information. In this way, the memory requirement and computational complexity of the model increases linearly with the length of the input sequence. This reduce the training burden greatly.

After passing through the last layer of the local self-attention transformer, we obtain the hidden states representation of the final layer:

$$
\mathbf{h}_1, \mathbf{h}_2, \cdots, \mathbf{h}_n = \text{Transformer}(T),
$$

where $\mathbf{h} \in \mathbb{R}^d$, d represents the dimension of each hidden state. All these representations of hidden units form the final matrix $\mathbf{H}$, i.e., $\mathbf{H} = [\mathbf{h}_1, \mathbf{h}_2, \cdots, \mathbf{h}_n] \in \mathbb{R}^{d \times n}$, which is further transferred to a liner layer:

$$
\mathbf{Z} = \mathbf{W}^\top \mathbf{H} + \mathbf{b},
$$

where $\mathbf{W} \in \mathbb{R}^{d \times 2}$ and $\mathbf{b} \in \mathbb{R}^2$ are trainable parameters.

3.2.3 Loss Function

For the target word $\hat{w}$, the correct start and end positions are represented as:

$$
\begin{aligned}
\textbf{Start} &= [\mathbf{Z}_{11} \ \cdots \ \mathbf{Z}_{1n}], \\
\textbf{End} &= [\mathbf{Z}_{21} \ \cdots \ \mathbf{Z}_{2n}],
\end{aligned}
$$

where **Start** and **End** indicate the logits for each token, denoting whether it is the start or the end of the text span corresponding with the correct sense of the target word $\hat{w}$, respectively.

We add the two cross-entropy loss functions for the start and end positions to train the model:

$$\mathcal{L}_s = -\textbf{Start}_{i_{cor}} + \log \sum_{v=1}^{l} \exp(\textbf{Start}_v),$$

$$\mathcal{L}_e = -\textbf{End}_{j_{cor}} + \log \sum_{v=1}^{l} \exp(\textbf{End}_v),$$

$$\mathcal{L} = \mathcal{L}_s + \mathcal{L}_e.$$

where $\textbf{Start}_{i_{cor}}$ and $\textbf{End}_{j_{cor}}$ are the scores correspond to the correct start and end positions.

3.2.4 Prediction

Following the work of Barba et al. (2021), our model outputs a pair of (i_{cor}, j_{cor}), which indicate the start and end positions of the right sense in the input text. To assure the pair is exactly matched with the text span of any sense, the model selects its output by comparing their probability. First, the logits **Start** and **End** are fed into softmax to obtain the probability distribution. Then, we perform a product operation on the probability distributions of the start and end positions to generate the probability of pair (i_{cor}, j_{cor}) that starts at i and ends at j :

$$\mathbf{P}(i_{cor}) = \text{softmax}(\textbf{Start}),$$

$$\mathbf{P}(j_{cor}) = \text{softmax}(\textbf{End}),$$

$$\mathbf{P}(i_{cor}, j_{cor}) = \mathbf{P}(i_{cor}) \times \mathbf{P}(j_{cor}),$$

where $\mathbf{P}(i_{cor})$ and $\mathbf{P}(j_{cor})$ indicates the probability that i_{cor} is the correct start position or the j_{cor} is the correct end position, respectively. $\mathbf{P}(i_{cor}, j_{cor})$ represents the probability of span that starts at i_{cor} and ends at j_{cor} across all the other spans in the input T.

Finally, the model outputs the pair with max probability, as follows:

$$output = \text{argmax}\ \mathbf{P}(i_{cor}, j_{cor}).$$

4 Experiment

4.1 Datasets

Following the existing works, we evaluate our proposed model on English all-words WSD task through a public unified evaluation framework (Raganato et al., 2017b). SemCor is selected as our training corpus (Miller et al., 1994), the smallest SemEval-2007 dataset (**SE07**) (Pradhan et al., 2007) is chosen as development set, and the rest are used as test datasets, including Senseval-2 (**SE2**) (Edmonds and Cotton, 2001), Senseval-3 (**SE3**) (Snyder and Palmer, 2004), SemEval-2013 (**SE13**) (Navigli et al., 2013), SemEval-2015 (**SE15**) (Moro and Navigli, 2015). The four test datasets are concatenated together marked as **ALL**. F1 score is used as the evaluation measure to report the performance.

4.2 Baselines

According to the exploitation of lexical knowledge, we categorize the baselines into three groups.

The first group includes the methods without any lexical knowledge, which merely rely on the training data and don't utilize any lexical knowledge, such as glosses and hypernyms. In this group, we first consider the MFS baseline, which simply adopts the most frequent sense in training datasets as the right sense of each word. Then, BiLSTM (Kågebäck and Salomonsson, 2016) is adopted, which is a early neural-based method and trains bidirectional LSTM to obtain a unified model for all-word WSD task. Besides, we select BERT_{base} (Devlin et al., 2019) as another baseline, which learns a linear classifier based on frozen BERT representations .

The second group involves the neural-based methods which exploit glosses of candidate senses, i.e., GAS (Luo et al., 2018), LMMS (Loureiro and Jorge, 2019), GlossBERT (Huang et al., 2019), ARES (Scarlini et al., 2020), BEM (Blevins and Zettlemoyer, 2020), ESCHER (Barba et al., 2021). These models utilize glosses to represent the corresponding senses. GAS is the first model to incorporate glosses into neural-based WSD, which jointly optimizes the representations of contexts and glosses of ambiguous words. Both LMMS and ARES are the nearest neighbors methods (k-NN), which identify the right sense according to the similarity between context and sense representation. LMMS generates sense representation from sense-annotated data, which is further enhanced with sense glosses in WordNet. ARES generates sense representation by leveraging the contexts in SemCor and the glosses in WordNet, which is further enriched with synset embeddings. Gloss-

Model	Dev Set	Test Sets				Concatenation of all Datasets				
	SE07	SE2	SE3	SE13	SE15	Nouns	Verbs	Adj.	Adv.	ALL
Baselines without any lexical knowledge										
MFS baseline	54.5	65.6	66.0	63.8	67.1	67.7	49.8	73.1	80.5	65.5
BiLSTM	-	71.1	68.4	64.8	68.3	69.5	55.9	76.2	82.4	68.4
BERT$_{base}$	68.6	75.9	74.4	70.6	75.2	75.7	63.7	78.0	85.8	73.7
Baselines with gloss information										
GAS*	-	72.0	70.0	66.7	71.6	71.7	57.4	76.5	83.5	70.1
LMMS*	68.1	76.3	75.6	75.1	77.0	-	-	-	-	76.8
GlossBERT*	72.5	77.7	75.2	76.1	80.4	79.8	67.1	79.6	87.4	77.0
ARES*	71.0	78.0	77.1	77.3	83.2	80.6	68.3	80.5	83.5	77.9
BEM*	74.5	79.4	77.4	79.7	81.7	81.4	68.5	83.0	**87.9**	79.0
ESCHER*	76.3	81.7	77.8	82.2	**83.2**	83.9	69.3	83.8	86.7	80.7
Baselines with gloss and other knowledge										
EWISER†	71.0	78.9	**78.4**	78.9	79.3	81.7	66.3	81.2	85.8	78.3
ESR$^{\dagger}_{base}$	75.4	80.6	78.2	79.8	82.8	82.5	**69.5**	82.5	87.3	79.8
KELESC†	**76.7**	**82.2**	78.1	**82.2**	83.0	**84.3**	69.4	**84.0**	86.7	**81.2**

Table 2: Comparison of F1 scores (%) on the English all-words WSD task. $*$ indicates that the model exploit glosses of candidate sense, † indicates that the model utilizes sense glosses as well as other knowledge in WordNet. We bold the best score for each column.

BERT reformulates WSD as a text matching task, which evaluates the matching degree between sense glosses and the contexts of ambiguous words to identify the right sense. BEM utilizes two encoders to represent contexts and candidate senses independently, and identify the right sense by finding the nearest sense embedding for the context embedding. ESCHER reformulates WSD as a text span extraction task, which is optimized to extract the text span of the gloss expressing the right sense when the model is fed with a sentence containing an ambiguous word and all its candidate sense glosses.

The third group consist of the methods which exploit more lexical knowledge, such as hypernyms, example sentence and gloss information. EWISER (Bevilacqua and Navigli, 2020) learns sense information from WordNet, which considers semantic relations between senses, such as hypernyms and hyponyms. ESR$_{base}$ (Song et al., 2021) further enhance sense representations by incorporating synonyms, example phrases or sentence and sense gloss of hypernyms.

4.3 Parameter Settings

We select BART$_{large}$ (Lewis et al., 2020) as our based model, whose encoder and decoder have 12 layers, respectively. In the encoder, the original self-attention is replaced by the local self-attention with a window size of 512 to avoid the huge computing complexity induced by the additional information. The optimizer is RAdam (Liu et al., 2020). Besides, we set batch size to 900 tokens, learning rate to 2e-6, and weight decay to 0.01. F1 score is calculated on validation dataset every 2000 steps, and stop training is applied if the model does not improve in 15 successive times. Our model is trained on one A100 GPU, which takes about 10 hours.

4.4 Overall Results

We evaluate the performance of our method by comparing it with the baselines. The overall results on English all-words WSD task are summarized in Table 2. According to the table, we have several observations.

First, the methods exploiting gloss information (i.e., the second group) usually outperform the methods without any lexical knowledge (i.e., the first group), except for BERT$_{base}$ and GAS. This demonstrates that gloss information is critical and essential for WSD, which is beneficial for learning better sense representations. Besides, the exception of BERT$_{base}$ and GAS may be caused that GAS is realized with BiLSTM whose learning ability is much weaker than BERT.

Second, the methods exploiting gloss and other lexical knowledge (i.e., the third group) outperform most of the methods in second group and all methods in first group. This is because that the methods in third group incorporate more lexical knowledge, such as hypernyms and example sentence, to enhance neural-based models, and the enhanced knowledge is useful for sense representation, which can further improve WSD performance.

Third, our model consistently outperform all competitive baseline methods on **ALL**. Our model also achieves the best performance on **SE07**, **SE2**, **SE13**, **Nouns** and **Adj.**. The reason for the superiority of our model is two-fold. One is that our model reformulates WSD as a text extraction task, which can accept and perceive all candidate sense, simultaneously. The other is that our model enhances sense representation with more lexical knowledge including hypernyms and examples. Among the baselines, ESCHER is the most similar to our model. Both models reformulate WSD as a text span extraction task. However, our model is better than ESCHER and increases F1 score by 0.5% on ALL. This is because that our model utilize more lexical knowledge than ESCHER.

4.5 Ablation Study

To evaluate the effectiveness of different lexical knowledge in our model, i.e., example sentence and hypernym gloss, we conduct ablation studies by removing them one by one to observe the change of overall performance.

Reserved Lexical Knowledge	**ALL** (%)
Example sentence + hypernym gloss	81.2
Example sentence	81.0
Hypernym gloss	80.8

Table 3: Comparison of ablated models on ALL.

As shown in Table 3, if we remove the gloss of hypernyms from our model, this leads to 0.2% drop from 81.2% to 81.0%. And, if we remove the example sentence, there is 0.4% drop from 81.2% to 80.8%. The above results indicate that the role of example sentence is more important than the gloss of hypernyms in our model. One explanation is that the example sentence is more semantically representative for the target sense.

4.6 Window Size in Local Self-Attention

In order to evaluate model training complexity and effectiveness, we employ different configurations of local self-attention. At the same time, we compare local self-attention with self-attention. The results are shown in Table 4:

Model	**Window**	**GiB**	**ALL** (%)
Self-attention	Full	31.2	80.8
Local self-attention	128	18.0	80.3
Local self-attention	256	20.8	80.8
Local self-attention	512	23.4	81.2

Table 4: Comparison of memory requirement and performance with different window sizes.

In Table 4, we observe that local self-attention mechanism can significantly reduce the memory usage, which is beneficial for accelerating training speed and reducing the training burden. Moreover, we find that the performance with window size of 128 is 80.3, which is 0.5% lower than the original self-attention, which is due to the fact that the window is too small and the model cannot effectively model the sense representation.

5 Conclusion and Future Work

In this paper, we proposed a novel WSD method with knowledge-enhanced and local self-attention-based extractive sense comprehension. Specifically, a knowledge-enhanced method was devised to enrich semantic representation by incorporating additional examples and definitions of the related senses in WordNet. Then, in order to avoid the huge computing complexity induced by the additional information, a local self-attention mechanism was utilized to constrain attentions to be local, which allowed longer input texts without large-scale computing burdens. Extensive experimental results had demonstrated the effectiveness of the proposed model on public benchmark datasets.

Although our model achieved better performance, it still could be improved. Currently, we utilized example sentence and hypernym gloss. There are many other unexplored semantic relations in WordNet and BabelNet. We leave it as future work to explore more semantic relations in more lexical resources to further enhance WSD performance. Besides, a detailed qualitative analysis on rare senses and frequent ones should be considered. We will attempt to evaluate the performance on

different situations to further enhance our model.

Acknowledgements

The work is partly supported by National Nature Science Foundation of China (61502259), and Key Program of Science and Technology of Shandong (2020CXGC010901), and Studio Project of the Research Leader in Jinan (2019GXRC062).

References

Eneko Agirre, Oier López de Lacalle, and Aitor Soroa. 2014. Random walks for knowledge-based word sense disambiguation. *Computational Linguistics*, 40(1):57–84.

Satanjeev Banerjee and Ted Pedersen. 2002. An adapted Lesk algorithm for word sense disambiguation using wordnet. In *Proceeding of the International Conference on Intelligent Text Processing and Computational Linguistics*, pages 136–145.

Edoardo Barba, Tommaso Pasini, and Roberto Navigli. 2021. ESC: Redesigning WSD with extractive sense comprehension. In *Proceedings of the Conference of the North American Chapter of the Association for Computational Linguistics*, pages 4661–4672.

Pierpaolo Basile, Annalina Caputo, and Giovanni Semeraro. 2014. An enhanced Lesk word sense disambiguation algorithm through a distributional semantic model. In *Proceedings of the 25th International Conference on Computational Linguistics*, pages 1591–1600.

Iz Beltagy, Matthew E Peters, and Arman Cohan. 2020. Longformer: The long-document transformer. *arXiv preprint arXiv:2004.05150*.

Michele Bevilacqua and Roberto Navigli. 2020. Breaking through the 80% glass ceiling: Raising the state of the art in word sense disambiguation by incorporating knowledge graph information. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 2854–2864.

Terra Blevins and Luke Zettlemoyer. 2020. Moving down the long tail of word sense disambiguation with gloss informed bi-encoders. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 1006–1017.

Simone Conia and Roberto Navigli. 2021. Framing word sense disambiguation as a multi-label problem for model-agnostic knowledge integration. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics*, pages 3269–3275.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the Conference of the North American Chapter of the Association for Computational Linguistics*, pages 4171–4186.

Philip Edmonds and Scott Cotton. 2001. Senseval-2: overview. In *Proceedings of the 2nd International Workshop on Evaluating Word Sense Disambiguation Systems*, pages 1–5.

Luyao Huang, Chi Sun, Xipeng Qiu, and Xuan-Jing Huang. 2019. GlossBERT: BERT for word sense disambiguation with gloss knowledge. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing*, pages 3509–3514.

Ignacio Iacobacci, Mohammad Taher Pilehvar, and Roberto Navigli. 2016. Embeddings for word sense disambiguation: An evaluation study. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics*, pages 897–907.

Mikael Kågebäck and Hans Salomonsson. 2016. Word sense disambiguation using a bidirectional LSTM. In *Proceedings of the 5th Workshop on Cognitive Aspects of the Lexicon*, pages 51–56.

Michael Lesk. 1986. Automatic sense disambiguation using machine readable dictionaries: How to tell a pine cone from an ice cream cone. In *Proceedings of the 5th International Conference on Systems Documentation*, pages 24–26.

Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Veselin Stoyanov, and Luke Zettlemoyer. 2020. BART: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7871–7880.

Liyuan Liu, Haoming Jiang, Pengcheng He, Weizhu Chen, Xiaodong Liu, Jianfeng Gao, and Jiawei Han. 2020. On the variance of the adaptive learning rate and beyond. In *Proceedings of the 2020 International Conference on Learning Representations*.

Daniel Loureiro and Alipio Jorge. 2019. Language modelling makes sense: Propagating representations through WordNet for full-coverage word sense disambiguation. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 5682–5691.

Fuli Luo, Tianyu Liu, Qiaolin Xia, Baobao Chang, and Zhifang Sui. 2018. Incorporating glosses into neural word sense disambiguation. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics*, pages 2473–2482.

Potsawee Manakul and Mark Gales. 2021. Long-Span summarization via local attention and content selection. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing*, pages 6026–6041.

George A Miller, Martin Chodorow, Shari Landes, Claudia Leacock, and Robert G Thomas. 1994. Using a semantic concordance for sense identification. In *Proceedings of the Workshop on Human Language Technology*.

Andrea Moro and Roberto Navigli. 2015. Semeval-2015 task 13: Multilingual all-words sense disambiguation and entity linking. In *Proceedings of the 9th International Workshop on Semantic Evaluation*, pages 288–297.

Roberto Navigli, Michele Bevilacqua, Simone Conia, Dario Montagnini, and Francesco Cecconi. 2021. Ten years of BabelNet: A survey. In *Proceedings of the 30th International Joint Conference on Artificial Intelligence*, pages 4559–4567.

Roberto Navigli, David Jurgens, and Daniele Vannella. 2013. Semeval-2013 task 12: Multilingual word sense disambiguation. In *Proceedings of the 7th International Workshop on Semantic Evaluation*, pages 222–231.

Roberto Navigli and Simone Paolo Ponzetto. 2012. BabelNet: The automatic construction, evaluation and application of a wide-coverage multilingual semantic network. *Artificial Intelligence*, 193:217–250.

Roberto Navigli and Paola Velardi. 2005. Structural semantic interconnections: A knowledge-based approach to word sense disambiguation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 27(7):1075–1086.

Tommaso Pasini and Roberto Navigli. 2020. Train-O-Matic: Supervised word sense disambiguation with no (manual) effort. *Artificial Intelligence*, 279:103215.

Sameer Pradhan, Edward Loper, Dmitriy Dligach, and Martha Palmer. 2007. Semeval-2007 task-17: English lexical sample, SRL and all words. In *Proceedings of the 4th International Workshop on Semantic Evaluations*, pages 87–92.

Alessandro Raganato, Claudio Delli Bovi, and Roberto Navigli. 2017a. Neural sequence learning models for word sense disambiguation. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 1156–1167.

Alessandro Raganato, Jose Camacho-Collados, and Roberto Navigli. 2017b. Word sense disambiguation: A unified evaluation framework and empirical comparison. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics*, pages 99–110.

Bianca Scarlini, Tommaso Pasini, and Roberto Navigli. 2020. With more contexts comes better performance: Contextualized sense embeddings for all-round word sense disambiguation. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing*, pages 3528–3539.

Federico Scozzafava, Marco Maru, Fabrizio Brignone, Giovanni Torrisi, and Roberto Navigli. 2020. Personalized pagerank with syntagmatic information for multilingual word sense disambiguation. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics: System Demonstrations*, pages 37–46.

Benjamin Snyder and Martha Palmer. 2004. The english all-words task. In *Proceedings of the Third International Workshop on the Evaluation of Systems for the Semantic Analysis of Text*, pages 41–43.

Yang Song, Xin Cai Ong, Hwee Tou Ng, and Qian Lin. 2021. Improved word sense disambiguation with enhanced sense representations. In *Findings of the Association for Computational Linguistics: EMNLP 2021*, pages 4311–4320.

Zhi Zhong and Hwee Tou Ng. 2010. It makes sense: A wide-coverage word sense disambiguation system for free text. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics: System Demonstrations*, pages 78–83.

A Novel Multi-Task Learning Approach for Context-Sensitive Compound Type Identification in Sanskrit

Jivnesh Sandhan[1], Ashish Gupta[2], Hrishikesh Terdalkar[1], Tushar Sandhan[1], Suvendu Samanta[1], Laxmidhar Behera[1,3] and Pawan Goyal[2]

[1]IIT Kanpur, [2]IIT Kharagpur, [3]IIT Mandi

jivnesh@iitk.ac.in, ashishgupta2598@gmail.com, pawang@cse.iitkgp.ac.in

Abstract

The phenomenon of compounding is ubiquitous in Sanskrit. It serves for achieving brevity in expressing thoughts, while simultaneously enriching the lexical and structural formation of the language. In this work, we focus on the *Sanskrit Compound Type Identification (SaCTI)* task, where we consider the problem of identifying semantic relations between the components of a compound word. Earlier approaches solely rely on the lexical information obtained from the components and ignore the most crucial contextual and syntactic information useful for SaCTI. However, the SaCTI task is challenging primarily due to the implicitly encoded context-sensitive semantic relation between the compound components.

Thus, we propose a novel multi-task learning architecture which incorporates the contextual information and enriches the complementary syntactic information using morphological tagging and dependency parsing as two auxiliary tasks. Experiments on the benchmark datasets for SaCTI show 6.1 points (Accuracy) and 7.7 points (F1-score) absolute gain compared to the state-of-the-art system. Further, our multilingual experiments demonstrate the efficacy of the proposed architecture in English and Marathi languages.[1]

1 Introduction

A compound is defined as a collection of one or more entities that act as a single meaningful entity. The process of decoding an implicit semantic relation between the components of a compound in Sanskrit is called *Sanskrit Compound Type Identification (SaCTI)*. Alternatively, it is also termed as *Noun Compound Interpretation (NCI)* (Ponkiya et al., 2021, 2020). In the literature, the NCI problem has been formulated in two related but different ways. Let's take *mango juice* as an example. In the first formulation, the relation between the two components is labeled from a set of semantic relations (*MADEOF*) (Dima and Hinrichs, 2015; Fares et al., 2018; Ponkiya et al., 2021). The second formulation uses paraphrasing to illustrate the semantic relations (*a juice made from mango*) (Lapata and Keller, 2004; Ponkiya et al., 2018a, 2020).

In this work, we use the first formulation that frames the task as a multi-class classification problem. The task is challenging and often depends upon the context or world knowledge about the entities involved (Krishna et al., 2016). For instance, the semantic type of the compound *rāma-īśvaraḥ* can be classified into one of the following semantic types depending on the context: *Karmadhāraya*[2], *Bahuvrīhi* and *Tatpuruṣa*. Although the compound has the same components as well as the final form, the implicit relationship between the components can be decoded only with the help of available contextual information (Kulkarni and Kumar, 2013; Krishna et al., 2016). Due to such instances, the downstream Natural Language Processing (NLP) applications for Sanskrit such as question answering (Terdalkar and Bhattacharya, 2019) and machine translation (Aralikatte et al., 2021), etc. show sub-optimal performance when they stumble on compounds. For example, while translating *rāma-īśvaraḥ* into English, depending on the semantic type, there are three possible meanings: (1) Lord who is pleasing (in *Karmadhāraya*) (2) the one whose God is Rama (in *Bahuvrīhi*) (3) Lord of Rama (in *Tatpuruṣa*). Therefore, the SaCTI task can be seen as a preliminary pre-requisite to building a robust NLP technology for Sanskrit. Further, this dependency on contextual information rules out the possibility of storing and doing a lookup to identify a compound's semantic types.

[1]The code and datasets are publicly available at: https://github.com/ashishgupta2598/SaCTI

[2]There are 4 broad semantic types of compounds: *Avyayībhāva*, *Bahuvrīhi*, *Dvandva*, and *Tatpuruṣa*. *Karmadhāraya* is considered as sub-type of *Tatpuruṣa*. We encourage readers to refer Krishna et al. (2016) for more details on these semantic types.

Proceedings of the 29th International Conference on Computational Linguistics, pages 4071–4083
October 12–17, 2022.

With the advent of recent contextual models (Peters et al., 2018; Devlin et al., 2019; Conneau et al., 2020), there has been upsurge in performance of various downstream NLP applications (Kondratyuk and Straka, 2019; Liu et al., 2019; Yang et al., 2019). Nevertheless, there have been no efforts to build context-dependent models in SaCTI.[3] This may be attributed to the fact that while most of the natural language technology is built for resource-rich languages such as English (Joshi et al., 2020), compounding is not a predominant phenomenon in them (Krishna et al., 2016). There is also lack of task-specific context-sensitive labeled data.

Earlier approaches (Kulkarni and Kumar, 2013; Krishna et al., 2016; Sandhan et al., 2019) for SaCTI solely rely on lexical information obtained from the components and are blind to potentially useful contextual and syntactic information. The context is the most feasible, cheaply available information. As per Paṇini's grammar (Paṇini, 500 BCE; Kulkarni and Kumar, 2013), the morphological features have direct correlation with the semantic types. Sometimes, the dependency information also helps is disambiguation and can provide a medium to enrich contextual information. Thus, we propose a novel multi-task learning approach which (1) incorporates the contextual information, and (2) enriches the complementary syntactic information using morphological tagging and dependency parsing auxiliary tasks without any additional manual labeling. Summarily, our key contributions are:

- We propose a novel context-sensitive multi-task learning architecture for SaCTI (§ 2).
- We illustrate that morphological tagging and dependency parsing auxiliary tasks are helpful and serve as a proxy for explainability of system predictions (§ 4) for the SaCTI task.
- We report results with 7.71 points (F1) absolute gains compared to the current state-of-the-art system by Krishna et al. (2016) (§ 3.2).
- We show the efficacy of the proposed approach in English and Marathi (§ 4).
- We release our codebase and pre-processed datasets (including newly annotated Marathi dataset (§ 3.1)) and web-based tool (§ 4) for using our pretrained models.

[3]Refer to related work section (§ 5) for more details.

2 The proposed system

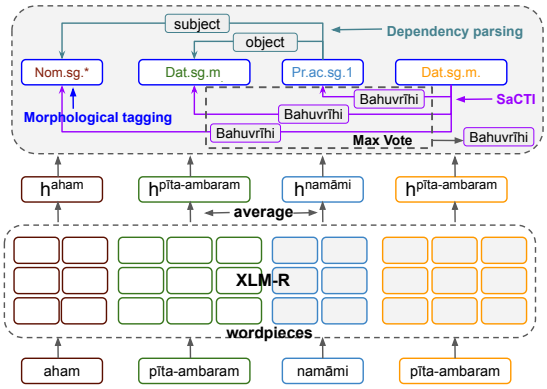

Figure 1: Illustration of the proposed multi-task learning architecture with an example "*aham pīta-ambaram namāmi*" (Translation: "I pray to Pītāmbara (Lord Vishnu)."") where '*pīta-ambaram*' is a compound word belonging to the *Bahuvrīhi* semantic type as per the given context. We feed this context and the compound word at the end as an input to the system. Each token is split into wordpieces using a multi-lingual tokenizer (Kudo and Richardson, 2018). This sequence of wordpieces is passed to multi-lingual pretrained XLM-R encoder (Conneau et al., 2020). The hidden representation of each token is the average of its wordpieces' representations obtained from the encoder. We apply our multi-task learning architecture which consists of three tasks, namely, Sanskrit compound type identification (SaCTI), morphological tagging and dependency parsing over the hidden representations. We formulate SaCTI as a pair-wise (a context word and the compound) classification task where the objective is to predict the semantic type of the target compound word (*Bahuvrīhi*). At test time, we apply the maximum voting policy to select a single prediction from the set of semantic relations predicted by such n pairs.

Figure 1 illustrates the proposed multi-task learning architecture with an example context, "*aham pīta-ambaram namāmi*" (Translation: "I pray to Pītāmbara (Lord Vishnu)."") where '*pīta-ambaram*' is a compound belonging to the *Bahuvrīhi* semantic type as per the given context. As shown in Figure 1, we feed this context along with the compound word concatenated at the end, as an input to the system, and obtain hidden representations from the multi-lingual encoder as described below. On top of the hidden representations as obtained via the encoder module, we apply our multi-task learning architecture consisting of three tasks: SaCTI, morphological tagging, and dependency parsing. We formulate the SaCTI task as a pair-wise (a context word paired with the compound word) classification task

where the objective is to predict the semantic type of the target compound word (*Bahuvrīhi* for *pīta-ambaram* compound word in this example) for all the pairs as shown in Figure 1. At test time, we apply the maximum voting policy to select a single prediction from the set of semantic types predicted by these pairs. We formally discuss the details of each component below.

Multilingual Encoder: Sanskrit is a morphologically rich and low-resource language. In order to build powerful contextual representations for Sanskrit words, morphological richness poses the out-of-vocabulary problem and low-resource nature poses an unlabelled data scarcity problem. Thus, we opt for a multi-lingual encoder (Conneau et al., 2020, XLM-R) to mitigate these issues.

Given a compound c_p and its context $C = [c_1, c_2, ..., c_n]$ such that p^{th} position $(1 \leq p \leq n)$ in the context is the compound word, we append the compound word to the context such that $C = [c_1, c_2, ..., c_n, c_{n+1}]$ where $c_{n+1} = c_p$. Each token (c_i) is further split into wordpieces $(c_i = [c_i^1, c_i^2, ..., c_i^{m_i}])$ using a multi-lingual subword tokenizer (Kudo and Richardson, 2018; Kudo, 2018, Sentencepiece). Next, we pass the overall sequence $(C = [c_1^1, c_1^2, ..., c_n^{m_n}, c_{n+1}^1 ..., c_{n+1}^{m_{n+1}}])$ of wordpieces corresponding to the context C into the pretrained transformer. Finally, we obtain the contextual representation of all tokens as $h = (h_1, h_2, h_3, ..., h_n, h_{n+1})$ where

$$h_i = \frac{1}{m_i} \sum_{k=1}^{m_i} Transformer(c_i^k) \quad (1)$$

SaCTI: Our context-sensitive classifier uses Bi-Affine attention, henceforth referred to as BiAFF. Given the hidden representations of ith, j^{th} context words as h_i, h_j from the multi-lingual encoder, the scoring function $s_{i,j}$ indicates the system's belief that the latter (j^{th}) (Eqn. 2) should be related to the former (i^{th}) in identifying the semantic type of the latter, where $q_i^T z_i$ indicates bias to capture the prior of contextual information in the ith word.

$$s_{i,j} = z_i^T U z_j + q_i^T z_i \quad (2)$$

where $z_i = MLP(h_i)$, U and q_i are learnable parameters, MLP denotes a multi-layered perceptron. Similarly, a score for k^{th} possible semantic type relation between every pair of ith context word $(\forall i \in [1, n])$ and the compound is computed by:

$$r_{i,k} = z_i^{'T} U_k^{'} z_{n+1}^{'} + q_k^{'T} [z_i^{'}; z_{n+1}^{'}] + b_k^{'} \quad (3)$$

where $z_i^{'} = MLP^{'}(h_i)$, $U^{'}$, $b_k^{'}$ and $q_k^{'}$ are learnable parameters. Finally, model maximizes the following objective function during training.

$$\sum_{i=1}^{n} p(y_{n+1}|c_i, \theta) + p(y_{n+1}^l|c_i, y_{n+1}, \theta) \quad (4)$$

where y_{n+1} is the target compound appended in context C, y_{n+1}^l is the semantic type of $(y_{n+1}/c_i, \theta)$, θ denotes system's parameters, $p(y_{n+1}|c_i, \theta) \propto exp(s_{i,n+1})$, $p(y_{n+1}^l|c_i, y_{n+1}, \theta) \propto exp(r_{i,l})$. At test time, we apply maximum voting policy to select a single prediction from the set of semantic relations predicted for n pairs $(c_i, c_{n+1}), \forall i \in [1, n]$.

Morphological tagging: The primary motivation behind using morphological tagging as an auxiliary task aligns well with grammatical rules from Paṇini's grammar (Paṇini, 500 BCE; Kulkarni and Kumar, 2013). For instance, *Avyayībhāva* compounds are in neuter gender. *Tatpuruṣa* is a function of 'case' attribute of morphological features. The number attribute of a compound depends on the semantic type of the compound. Also, there are constraints based on inflection/derivational suffix. Summarily, these morphological features have direct correlation with the semantic classes. In our proposed system, the morphological tagging task leverages hidden representations from the multi-lingual encoder and decodes the pseudo-labels[4] using a fully connected layer followed by a softmax layer.[5] In this process, morphological information useful for the SaCTI task is enriched in the hidden representations. This can be seen as an implicit way to encode the grammatical rules in the system.

Dependency parsing: Lowe (2015) argues that compounding is mostly a syntactic phenomenon. *Bahuvrīhi* compounds are "exocentric" in nature, which attribute a property to an entity external to the compound with the adjective relationship. Sometimes, syntactic information can provide a complementary signal useful for compound type disambiguation. For instance, consider the following example: *aham nīla-utpalaḥ saraḥ paśyāmi* (Translation: I watch the pond having a blue-lotus.) Here, *nīla-utpalam* qualifies to be *Bahuvrīhi* due to

[4]The benchmark datasets do not have a gold standard morphological information. We use predicted morphological information as pseudo-labels (§ 3.1).

[5]Note that all the parameters present in the multi-lingual encoder are trainable during the task-specific training.

presence of its referent *saraḥ* with an adjective relationship. However, in the absence of *saraḥ* in the context, ambiguity pops up in between *Bahuvrīhi* and *Tatpuruṣa*.[6] This motivates us to investigate the usefulness of syntactic information for the SaCTI task. The benchmark datasets do not have a gold standard dependency information. We use predicted dependency trees as pseudo-labels (§ 3.1). Our dependency parsing component leverages Bi-Affine parser (Dozat and Manning, 2017) over hidden representations from multi-lingual encoder.

3 Experiments

3.1 Dataset and metrics

Table 1 reports the unique number of compounds, data statistics and the number of semantic types for the respective datasets used in this work. We restrict to binary compounds (compounds with two components) in all the datasets. These datasets consist of components, context and semantic type of a compound. The SaCTI datasets for Sanskrit are available with two levels of annotations: coarse (4 broad types) and fine-grained (15 sub-types).

Datasets	#Unique	#Train	#Dev	#Test	#Types
SaCTI-base	8,594	9,356	2,339	2,994	4 (15)
SaCTI-large	48,132	59,133	6,571	7,301	4 (15)
English	4,676	4,163	1,041	1,301	7
Marathi	368	659	99	114	4

Table 1: Data statistics for all the datasets

Sanskrit: We evaluate on two available context-sensitive benchmark datasets: *SaCTI-base* and *SaCTI-large*. We follow the same experimental settings as Krishna et al. (2016) in *SaCTI-base* to keep our results comparable with their state-of-the-art results. *SaCTI-base* is a subset of *SaCTI-large* dataset. In due course of time, more annotated data resulted in *SaCTI-large* dataset.

English: We use instance-based (context-dependent) noun-noun compound dataset released by Fares (2016). The compounds used in this dataset are extracted from the Wall Street Journal (WSJ) portion in the Penn Treebank (PTB).

Marathi: We create an annotated context-sensitive compound data for Marathi due to its unavailability. We extract compound words from Marathi grammar textbooks. For Marathi, we restrict to the same 4 semantic types as in Sanskrit (coarse setting). In our dataset, the context corresponding to 75% data points is automatically leveraged from Wikipedia. In order to increase the difficulty level of the task, we ask one of the annotators to create the remaining 25% data points in such a way that the same compound with a different context leads to a different semantic type. Next, we provide these compound words and context information to 3 annotators (A, B, C)[7] using a web-based platform. Refer to Appendix B for annotation interface. All annotators have their minimum academic qualification as Master in Arts in Marathi. These annotators have to choose the correct label from the multiple-choice options.[8] Finally, we use the maximum voting policy amongst annotators to get the label for each data point. Initially, we start with 1,000 data points for annotation. Out of 1,000, we drop 128 data points where none of the 2 annotators have an agreement. The pair-wise inter-annotator agreement between annotators in terms of Cohen Kappa (κ) is as follows: $A - B : 0.40, B - C : 0.20$ and $A - C : 0.35$. The $\kappa \in [0.2, 0.4]$ is considered as fair agreement (McHugh, 2012).

Psuedo-labels for auxiliary tasks: The benchmark datasets do not have a gold standard dependency and morphological information. We use predicted labels as pseudo-labels. For Sanskrit, we obtain pseudo-labels from the Trankit model (Nguyen et al., 2021) trained on STBC dataset (Krishna et al., 2020) and morphological pseudo-labels from the LemmaTag model (Kondratyuk et al., 2018) trained on Hackathon dataset (Krishnan et al., 2020). For English, we use pseudo-labels from English XLM-R model[9] for morphological and dependency parsing task. For Marathi, we do not find any such data or pretrained model to obtain pseudo-labels. Therefore, we do not activate morphological tagging and dependency parsing components in the proposed system while training on Marathi data.

Hyper-parameter settings: For the implementation of the proposed system, we modify on top of codebase by Nguyen et al. (2021). We use the following hyper-parameter settings for the best config-

[6] The ambiguity is whether I am seeing the blue lotus or the pond having a blue lotus.

[7] Note that the annotator who created the context for 25% data points is different from these 3 annotators.

[8] In case of confusion, we gave additional option to mark as *"Not sure"*.

[9] https://spacy.io/models/en

Datasets	System	Sanskrit (coarse)								Sanskrit (fine-grained)							
		w/o context				w/ context				w/o context				w/ context			
		A	P	R	F1	A	P	R	F1	A	P	R	F1	A	P	R	F1
SaCTI-base	ISCLS19	77.68	76.00	71.00	73.00	77.68	76.00	71.00	73.00	70.64	67.53	63.18	64.58	70.64	67.53	63.18	64.58
	COLING16	77.39	**78.00**	72.00	**74.00**	77.39	78.00	72.00	74.00	-	-	-	-	-	-	-	-
	SanALBERT	72.01	72.10	70.00	71.40	77.40	77.31	73.00	74.20	69.39	62.58	58.13	58.22	75.40	70.14	61.22	62.04
	IndicALBERT	71.87	51.60	53.90	52.00	78.63	77.47	75.83	76.47	68.06	53.05	53.19	52.30	71.93	57.29	56.49	55.25
	mBERT	76.12	77.43	71.68	73.10	81.42	83.12	73.54	78.09	75.00	67.65	72.69	68.56	78.36	77.41	69.59	69.37
	XLM-R	78.19	73.54	73.10	73.31	81.00	82.01	77.00	79.10	73.75	66.38	63.96	65.16	77.94	71.72	70.18	70.94
	Ours	**80.21**	72.31	**74.50**	73.38	**83.45**	**79.65**	**83.87**	**81.71**	**78.25**	**72.94**	**73.81**	**72.68**	**82.47**	**76.87**	**79.08**	**77.20**
SaCTI-large	ISCLS19	90.69	75.66	72.09	73.76	90.69	75.66	72.09	73.76	76.66	71.62	65.40	68.09	76.66	71.62	65.40	68.09
	SanALBERT	88.17	66.65	61.31	62.85	87.84	64.15	65.44	63.63	79.38	69.90	75.34	67.57	80.73	71.62	74.60	72.69
	IndicALBERT	87.77	68.82	49.58	56.05	92.95	84.98	74.90	79.32	79.03	68.00	64.14	63.67	83.13	74.23	79.77	76.11
	mBERT	92.29	78.51	77.45	77.41	93.52	83.13	80.82	81.83	81.56	70.82	76.91	72.74	80.98	70.00	79.13	72.80
	XLM-R	92.61	79.91	79.00	79.42	93.85	86.64	79.67	82.78	81.84	74.46	77.93	75.68	83.12	73.97	81.07	76.20
	Ours	**93.54**	**81.30**	**81.65**	**81.47**	**94.78**	**83.89**	**87.61**	**85.64**	**82.85**	**74.94**	**78.12**	**76.49**	**84.73**	**78.53**	**80.30**	**77.13**

Table 2: Evaluation on Sanskrit datasets in two levels of annotations (coarse and fine-grained) and two settings (w/o context and w/ context). The best results are bold. Results are averaged over 4 runs. The significance test between the best baseline XLM-R and the proposed system in terms of Recall/Accuracy metrics: $p < 0.01$ (as per t-test). We could not perform significance test with COLING16 (SaCTI-base-coarse-w/o) and report its results on all the datasets due to unavailability of its predictions and codebase. ISCLS19 and COLING16 baselines cannot utilize the context information; therefore, we report the same numbers in w/context as w/o context.

uration of the proposed system: number of epochs as 70, batch size 50 and a embedding dropout rate of 0.3 with a learning rate of 0.001. In our multi-task loss function, we penalize dependency component's loss function by 0.01 to prioritize the performance on SaCTI. This penalty is identified based on hyper-parameter tuning. The rest of the hyper-parameters are kept the same as Nguyen et al. (2021). For multi-lingual baselines, we used Huggingface's transformers repository (Wolf et al., 2020). We release our codebase and datasets publicly under the licence CC-BY 4.0. All the artifacts used in this work are publicly available for the research purpose.

Computing infrastructure used: We use a single GPU with Tesla P100-PCIE, 16 GB GPU memory, 3584 GPU Cores computing infrastructure for our experiments. Our proposed system takes approximately 1 hour for training SaCTI-base coarse w/o context setting dataset.

Evaluation metrics: Following Krishna et al. (2016); Sandhan et al. (2019), we report macro averaged **P**recision, **R**ecall and **F1**-score for all our experiments. We also report micro-averaged **A**ccuracy. We use Scikit-learn software (Pedregosa et al., 2011) to compute these metrics.

Baselines: We consider two context agnostic systems where Sandhan et al. (2019, **ISCLS19**) formulate SaCTI as a purely neural-based multi-class classification approach using static word embeddings of components of a compound and Krishna et al. (2016, **COLING16**) deploy a hybrid system which leverages linguistically involved handcrafted feature engineering with distributional information from Adaptor Grammar (Johnson et al., 2006). The COLING16 system is the current state-of-the-art system for the SaCTI task. Next, we opt for multi-lingual contextual language models due to the lack of sufficiently large unlabelled data available for Sanskrit. We consider three multi-lingual pretrained language models, namely, Kakwani et al. (2020, **IndicALBERT**) which is ALBERT model trained on 12 Indic languages excluding Sanskrit, BERT (Devlin et al., 2019, **mBERT**) trained on 104 languages having largest Wikipedia's excluding Sanskrit and Conneau et al. (2020, **XLM-R**) trained on 100 languages including Sanskrit. Finally, we consider a mono-lingual ALBERT model (Sandhan et al., 2022, **SanALBERT**) trained on Sanskrit corpus (Hellwig, 2010) from scratch. In all the contextual baselines, we pass the classification token [CLS] representation of a sentence pair (compound word and its context separated by [SEP] token) to the classification head. **Ours:** This is our proposed system from § 2.

3.2 Results

Table 2 shows the performance for the best performing configurations of all the baselines on the test set of benchmark datasets for SaCTI. We report results on two levels of annotations (coarse and fine-grained) and two settings (w/o context and w/

context).[10] Except for ISCLS19 and COLING16 systems, all systems utilize available context along with components of a compound.[11] XLM-R reports the best performance among all the baselines while using the context information.

Our proposed system outperforms all the competing systems in terms of all the evaluation metrics and reports 6.1 points (A) and 7.7 points (F1) absolute gain with respect to the current state-of-the-art system COLING16 (on SaCTI-base coarse w/context dataset). COLING16 outperforms the proposed system on SaCTI-base coarse w/o context. Notably, our proposed system outperforms the strong baseline XLM-R with large margins in fine-grained setting on SaCTI-base dataset (low-resourced setting). This confirms the usefulness of the proposed system in low-resourced settings with fine-grain labels. The large performance gap between the proposed system with context and COLING16/ISCLS19 baselines illustrates the efficacy of using contextual information and syntactic information. Summarily, we mark new state-of-the-art results with the help of the novel architecture, where the contextual component is integrated with syntax-based auxiliary tasks such as morphological tagging and dependency parsing. We find a similar trend in performance for the SaCTI-large dataset.

4 Analysis

In this section, we dive deep into the proposed system architecture for a detailed analysis as well as generalizability. We use SaCTI-base coarse dataset in the w/ context setting for the analysis.

(1) Ablation analysis: Here, we investigate the contribution of various components towards the overall improvements of the proposed system. Table 3 reports ablations in terms of all the evaluation metrics when a particular component is inactivated from the proposed system. For example, "-DP" denotes the system where the dependency parsing component is removed from the proposed system. We see that elimination of any of the components deteriorates the performance. Table 3 illustrates that 'context' component is the most critical towards improvements. Also, the deletion of the 'BiAFF' component has the second largest impact on the final performance.[12]

System	A	P	R	F1
Ours	83.45	79.65	83.87	81.71
-context	80.21	72.31	74.50	73.38
-BiAFF	81.00	82.01	77.00	79.10
-morph	82.87	80.00	81.21	80.60
-DP	81.89	79.35	81.62	80.26
-morph -DP	81.50	82.34	77.67	79.93

Table 3: Ablations of the proposed system in terms of all the metrics. Each ablation deletes a single component from the proposed system. For example, "-DP" deletes the dependency parsing task from the proposed system.

(2) How effective is the proposed system in reducing confusion between conflicting classes? Figure 2 illustrates the confusion matrices in w/ context and w/o context scenarios. We observe a similar trend in both the scenarios. (1) Both systems mis-classify the predictions into the most populated type (*Tatpuruṣa*). This can be attributed to the imbalanced nature of the dataset.[13] (2) The confusion between *Avyayībhāva* and *Tatpuruṣa* is due to these compounds having their first component as an indeclinable word. Notably, the system with context is able to reduce confusion by 15%. (3) One of the reasons for conflict between *Tatpuruṣa* and *Bahuvrīhi* is due to the specific subcategory of both the classes where the first component is a negation. With the help of enriched information, a system with context can reduce this miss-classification by 7%. Summarily, the system with contextual information always performs superior to one with no context. This substantiates the importance of contextual information in the proposed system.

(3) How well can we generalize the proposed system for other languages? The primary motivation is to illustrate the efficacy of our language agnostic approach. The semantic type of compounds of the language of interest need not be similar to that of Sanskrit for the model to work. It is purely language agnostic model. To study the generalization ability of the proposed system, we consider 2 additional languages, namely, English (en) and Marathi (ma). We choose English due to its availability of context-sensitive annotated

[10] For the systems that require context, we feed the compound word only as the context.

[11] These baselines cannot utilize the context; therefore, we report the same numbers in w/context as w/o context.

[12] In the absence of the proposed 'BiAFF' component, we use [CLS] token for the sentence-level prediction, where this system is similar to XLM-R + DP + morph.

[13] In this work, we do not consider any strategy to tackle imbalanced classification. We plan to address this in future.

	English								Marathi							
	w/o context				w/ context				w/o context				w/ context			
System	A	P	R	F1	A	P	R	F1	A	P	R	F1	A	P	R	F1
ISCLS19	67.43	71.81	64.38	66.26	67.43	71.81	64.38	66.26	68.62	70.78	52.89	56.85	68.62	70.78	52.89	56.85
IndicALBERT	68.23	57.25	59.70	57.96	70.22	68.01	69.31	68.65	67.65	33.33	45.15	38.08	60.00	45.17	40.62	40.95
mBERT	70.98	70.00	69.87	69.97	72.48	76.08	73.47	74.30	77.45	58.70	61.28	59.48	71.05	51.45	53.34	52.18
BERT	74.19	72.79	71.12	71.62	74.71	75.83	74.48	75.12	78.43	71.21	68.04	69.20	75.43	67.07	66.07	66.43
XLM-R	72.21	71.53	68.88	69.20	74.33	77.12	76.86	76.57	76.47	67.36	62.90	63.33	74.56	65.50	62.07	62.75
Ours	**74.69**	**74.79**	**75.19**	**75.19**	**77.81**	**79.17**	**79.19**	**79.12**	**78.12**	**71.98**	**70.00**	**70.57**	**80.43**	**66.54**	**77.00**	**69.12**

Table 4: Evaluation on English and Marathi languages. The best results are bold. The significance test between the best baseline XLM-R and our system in terms of Recall/Accuracy metrics: $p < 0.01$ (as per t-test). ISCLS19 do not have power to utilize the context information; therefore, we report the same numbers in w/context as w/o context.

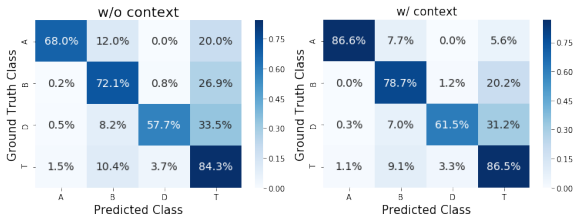

(a) (b)

Figure 2: The confusion matrix for the proposed system trained (a) w/o context (b) w/ context. Semantic types: ***A**vyayībhāva*, ***B**ahuvrīhi*, ***D**vandva*, and ***T**atpuruṣa*

data and Marathi due to its closeness to Sanskrit. We freshly created annotated task-specific context-sensitive data for Marathi (§ 3.1) as no such dataset was previously available. Table 4 reports the results for these two languages. For English, all the baselines (with context) improve over their counterpart (without context). However, we do not find similar trend in Marathi possibly due to (1) lack of sufficient task-specific data, and (2) lack of both the auxiliary task [14]. For both languages, our system consistently outperforms all the competing systems. Across both the languages, it shows the average absolute gain of 4.7 points (A) and 4.4 points (F1) compared to the strong baseline XLM-R. Summarily, these empirical results prove the proposed approach's efficacy in languages other than Sanskrit.

(4) Multi-lingual training and zero-shot cross-lingual transfer experiments for Marathi: Here, we investigate the transferability of the SaCTI task for low-resourced languages. We experiment with the Marathi language (w/ context). Since the label space of Marathi is the same as that of the SaCTI coarse dataset, this makes it possible to experiment with cross-lingual zero-shot transfer and multi-lingual training. There is an isomorphic semantic type system with 4 types for Marathi as is the case for most of the Indian languages, due to a close connection with / inheritance from Sanskrit. In Table 5, we consider the mono-lingual (training on Marathi) results as a baseline. In multi-lingual training, we train the proposed system with mix of Marathi and SaCTI-base coarse dataset and evaluate on test set of Marathi.[15] In the zero-shot transfer, we leverage the model trained on the SaCTI-base coarse dataset to get predictions on the test set of Marathi. In multi-lingual training experiment, we observe substantial improvements (3.4 points F1) over mono-lingual training. However, zero-shot cross-lingual transfer does not show encouraging results, where system predicts *Tatpuruṣa* type for majority test samples (80% predictions).

Tasks	A	P	R	F1
ours (Marathi)	80.43	66.54	77.00	69.12
zero-shot transfer	48.09	33.85	48.36	31.08
multi-lingual training	**83.10**	**71.00**	**80.48**	**73.70**

Table 5: Performance of the multilingual training and cross-lingual zero-shot transfer on Marathi.

(5) Probing analysis: Here, we probe the attention modules of the proposed system to investigate (1) How do different context words contribute towards final prediction? (2) Do these attentions serve as a proxy for explainability of correct/incorrect predictions? Figure 3 illustrates attention heatmaps for the SaCTI (Blue) and dependency parsing (Purple) tasks. The SaCTI heatmap shows how different context words contribute towards final prediction (Blue) and dependency parsing heatmap (Purple) serves as proxy for interpretation. We notice in the SaCTI attentions that all

[14] We could not activate auxiliary tasks due to lack of datasets for Marathi.

[15] Here, we use pretrained models of Sanskrit for both the auxiliary tasks to obtain psuedo-labels for Marathi.

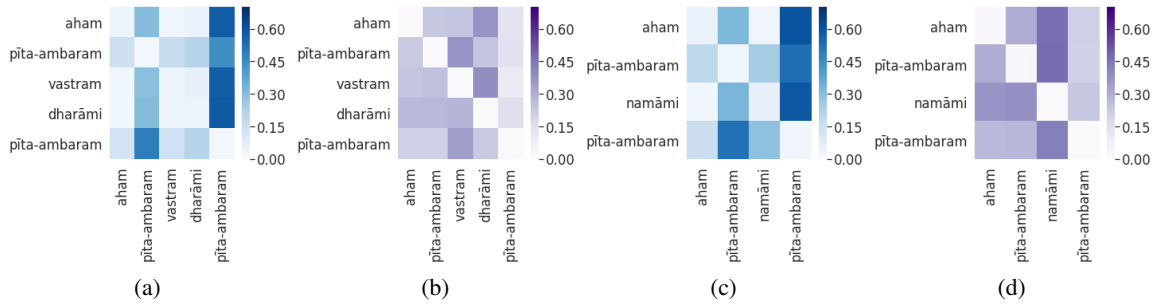

Figure 3: Attention heatmaps for the SaCTI (Blue) and dependency parsing (Purple) tasks. The SaCTI heatmap shows how different context words contribute towards final prediction (Blue) and dependency parsing heatmap (Purple) serve as proxy for interpretation. We illustrates how the same compound (*pīta-ambaram*) in two different contexts [(a-b) *aham* ***pīta-ambaram*** *vastram dharāmi* (I wear a yellow cloth) and (c-d) *aham* ***pīta-ambarām*** *namāmi* (I pray to the Lord Viṣnu)] leads to different semantic type predictions (*Tatpuruṣa*: yellow cloth and *Bahuvrīhi*: Lord Viṣnu). In the dependency heatmap of the first case, *pīta-ambaram* focuses on *vastram* (cloth) and in latter case it focuses on *namāmi* (the action of praying).

words mostly focus on the target compound word. Figure 3 illustrate how the same compound (*pīta-ambaram*) in two different contexts [(a-b) *aham* ***pīta-ambaram*** *vastram dharāmi* (I wear a yellow cloth) and (c-d) *aham* ***pīta-ambarām*** *namāmi* (I pray to the Lord Viṣnu)] leads to different semantic type predictions (*Tatpuruṣa*: yellow cloth and *Bahuvrīhi*: Lord Viṣnu). In the dependency heatmap of the first case, *pīta-ambaram* focuses on *vastram* (cloth) and in latter case, it focuses on *namāmi* (the action of praying). As per the grammatical rules, the morphological tagging task correctly predicts the gender information as neuter and masculine in these cases, respectively. Thus, this probing analysis suggests that auxiliary tasks not only help add complementary signals to the system but also serve as a proxy for explainability.

(6) Additional auxiliary tasks: With our proposed multi-task learning approach, we experiment with a few more additional sequence labeling auxiliary tasks (on SaCTI-base w/ context dev set), namely, the prediction of the case grammatical category (C), lemma prediction (L) and prediction of a relation (R) between modifier and its headword. The results in Table 6 show that except for the relation prediction task, all the remaining auxiliary tasks report improvements over the base system (with no auxiliary task). However, none of the combinations of these auxiliary tasks could outperform the proposed combination of morphological parsing and dependency parsing tasks. Therefore, we do not consider these additional auxiliary tasks in our final system.

Tasks	A	P	R	F1
BiAFF	87.99	85.48	87.90	86.64
+case (C)	87.64	85.65	88.90	87.19
+morph (M)	88.01	88.90	85.75	87.26
+relation (R)	86.61	85.53	84.99	85.22
+lemma (L)	87.43	88.27	86.21	87.18
+Dep. parse (DP)	89.14	86.49	90.10	88.01
M+C	87.55	88.30	85.28	86.72
M+C+L	87.08	87.74	84.98	86.30
M+C+R	86.10	86.04	83.28	84.55
M+DP	**88.11**	**86.12**	**89.23**	**88.43**

Table 6: The comparison (on SaCTI-base w/ context dev set) in between auxiliary tasks. '+' denotes a system where the corresponding task is integrated with BiAFF.

(7) Web-based tool: We deploy our pretrained models as a web-based tool which facilitates the following advantages: (1) A naive user with no prior deep-learning expertise can use it for pedagogical purposes. (2) It can serve as a semi-supervised annotation tool keeping a human in the loop for the error corrections. (3) Our tool helps the user interpret the model prediction using model confidence on each semantic type and the probing analysis. Refer to Appendix A for our web-based tool's interface. (4) It can be used for any general purpose classification task.

5 Related work

English Noun Compound Interpretation Prior to the deep learning era, various machine learning-based approaches have been proposed for Noun Compound Identification (Kim and Baldwin, 2005; Ó Séaghdha and Copestake, 2009; Tratz and Hovy,

2010). With the advent of deep-learning based approaches, Dima and Hinrichs (2015) and Fares et al. (2018) proposed a neural-based architecture where concatenated representations of a compound were fed to a feed-forward network to predict a semantic relation between the compound's components. Shwartz and Waterson (2018) proposed an approach that combines labeling with paraphrasing. Recently, Ponkiya et al. (2021) proposed a novel approach using semantic label repository (Ponkiya et al., 2018b, FrameNet) where continuous label space embeddings are used to predict unseen labels. To the best of our knowledge, a context has never been used for the classification task for NCI. In paraphrasing line of modeling, Ponkiya et al. (2020) formulates paraphrasing as "fill-in-the-blank" problem to predict the "missing" predicate or preposition using pretrained language models.

Sanskrit Compound Type Identification task has garnered considerable attention of the researchers in the last decade. In order to decode the meaning of a Sanskrit compound, it is essential to figure out its constituents (Gérard, 2010; Mittal, 2010; Hellwig and Nehrdich, 2018), how the constituents are grouped (Kulkarni and Kumar, 2011), identify the semantic relation between them (Kumar, 2012) and finally generate the paraphrase of the compound (Kumar et al., 2009). Satuluri and Kulkarni (2013) and Kulkarni and Kumar (2013) proposed a rule-based approach where around 400 rules mentioned in Pāṇini's grammar (Paṇini, 500 BCE) were analysed from the perspective of compound generation and type identification, respectively. Recently, Sandhan et al. (2019) investigated whether a purely engineering data-driven approach competes with the performance of a linguistically motivated hybrid approach by Krishna et al. (2016). Summarily, no attention has been given to incorporating contextual information, which is crucial and cheaply available. We address this research gap and mark the new state-of-the-art results with substantial improvements.

6 Conclusion and Discussion

This work focused on Sanskrit compound type identification, where the task is to decode the semantic information hidden in the compound, which can be *context-sensitive*. This poses a limitation to the existing *context agnostic* approaches, thus we propose a novel multi-task learning architecture which incorporates the contextual information and also enriches it with complementary syntactic information using morphological tagging and dependency parsing auxiliary tasks. Our probing analysis showcased that these auxiliary tasks also serve as a proxy for model prediction explainability. To the best of our knowledge, this is the first time that the importance of these auxiliary tasks has been showcased for SaCTI. Our experiments on benchmark datasets showed that the proposed system provides stunning improvements with 6.1 points (A) and 7.7 points (F1) absolute gain compared with the current state-of-the-art system. Our fine-grained analysis showcased some light on the inner engineering of the proposed system. Our multi-lingual experiments on English and Marathi languages proved the efficacy of the proposed system in other languages.

We limit our study to the *purely engineering* data-driven settings. We plan to extend the current work by augmenting logical rules (Li and Srikumar, 2019; Nandwani et al., 2019) derived from Pāṇinian grammar in the proposed approach.

Ethics Statement: We do not foresee any ethical concerns with the work presented in this manuscript.

Acknowledgements

We thank Amba Kulkarni for providing annotations for SaCTI datasets. We are grateful to Pavankumar Satuluri for helping us with insightful discussion. We would like to thank Narein Rao, Rathin Singha and the anonymous reviewers for their constructive feedback towards improving this work. We thank Om Adideva Paranjay and Ayush Daksh for helping us with the data pre-processing step. We would like to thank Jayesh Lakade, Prachi Rahangade and Aakash Biswas for helping us with the Marathi compound raw data collection. We thank Varsha Sandhan, Chitra Manjarekar, Khyati Deshpande and Rachit Bodhare for helping us with the Marathi data annotations. The work of the first author is supported by the TCS Fellowship under the Project TCS/EE/2011191P.

References

2022. Bootstrap 5.1. https://getbootstrap.com/docs/5.1/getting-started/introduction/.

Rahul Aralikatte, Miryam de Lhoneux, Anoop Kunchukuttan, and Anders Søgaard. 2021. Itihasa: A large-scale corpus for Sanskrit to English translation.

In *Proceedings of the 8th Workshop on Asian Translation (WAT2021)*, pages 191–197, Online. Association for Computational Linguistics.

Alexis Conneau, Kartikay Khandelwal, Naman Goyal, Vishrav Chaudhary, Guillaume Wenzek, Francisco Guzmán, Edouard Grave, Myle Ott, Luke Zettlemoyer, and Veselin Stoyanov. 2020. Unsupervised cross-lingual representation learning at scale. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 8440–8451, Online. Association for Computational Linguistics.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, Minneapolis, Minnesota. Association for Computational Linguistics.

Corina Dima and Erhard Hinrichs. 2015. Automatic noun compound interpretation using deep neural networks and word embeddings. In *Proceedings of the 11th International Conference on Computational Semantics*, pages 173–183, London, UK. Association for Computational Linguistics.

Timothy Dozat and Christopher D. Manning. 2017. Deep biaffine attention for neural dependency parsing. In *5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Conference Track Proceedings*. OpenReview.net.

Murhaf Fares. 2016. A dataset for joint noun-noun compound bracketing and interpretation. In *Proceedings of the ACL 2016 Student Research Workshop*, pages 72–79, Berlin, Germany. Association for Computational Linguistics.

Murhaf Fares, Stephan Oepen, and Erik Velldal. 2018. Transfer and multi-task learning for noun–noun compound interpretation. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 1488–1498, Brussels, Belgium. Association for Computational Linguistics.

Huet Gérard. 2010. Sanskrit segmentation. In *SALA XXVIII*.

Oliver Hellwig. 2010. Dcs-the digital corpus of sanskrit. *Heidelberg (2010-2022). URL http://www.sanskrit-linguistics.org/dcs/index.php*.

Oliver Hellwig and Sebastian Nehrdich. 2018. Sanskrit word segmentation using character-level recurrent and convolutional neural networks. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 2754–2763, Brussels, Belgium. Association for Computational Linguistics.

Mark Johnson, Thomas L. Griffiths, and Sharon Goldwater. 2006. Adaptor grammars: A framework for specifying compositional nonparametric bayesian models. In *NIPS*.

Pratik Joshi, Sebastin Santy, Amar Budhiraja, Kalika Bali, and Monojit Choudhury. 2020. The state and fate of linguistic diversity and inclusion in the NLP world. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 6282–6293, Online. Association for Computational Linguistics.

Divyanshu Kakwani, Anoop Kunchukuttan, Satish Golla, Gokul N.C., Avik Bhattacharyya, Mitesh M. Khapra, and Pratyush Kumar. 2020. IndicNLPSuite: Monolingual Corpora, Evaluation Benchmarks and Pre-trained Multilingual Language Models for Indian Languages. In *Findings of EMNLP*.

Su Nam Kim and Timothy Baldwin. 2005. Automatic interpretation of noun compounds using WordNet similarity. In *Second International Joint Conference on Natural Language Processing: Full Papers*.

Dan Kondratyuk and Milan Straka. 2019. 75 languages, 1 model: Parsing Universal Dependencies universally. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 2779–2795, Hong Kong, China. Association for Computational Linguistics.

Daniel Kondratyuk, Tomáš Gavenčiak, Milan Straka, and Jan Hajič. 2018. LemmaTag: Jointly tagging and lemmatizing for morphologically rich languages with BRNNs. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 4921–4928, Brussels, Belgium. Association for Computational Linguistics.

Amrith Krishna, Ashim Gupta, Deepak Garasangi, Pavankumar Satuluri, and Pawan Goyal. 2020. Keep it surprisingly simple: A simple first order graph based parsing model for joint morphosyntactic parsing in Sanskrit. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 4791–4797, Online. Association for Computational Linguistics.

Amrith Krishna, Pavankumar Satuluri, Shubham Sharma, Apurv Kumar, and Pawan Goyal. 2016. Compound type identification in Sanskrit: What roles do the corpus and grammar play? In *Proceedings of the 6th Workshop on South and Southeast Asian Natural Language Processing (WSSANLP2016)*, pages 1–10, Osaka, Japan. The COLING 2016 Organizing Committee.

Sriram Krishnan, Amba Kulkarni, and Gérard Huet. 2020. Validation and normalization of dcs corpus using sanskrit heritage tools to build a tagged gold corpus.

Taku Kudo. 2018. Subword regularization: Improving neural network translation models with multiple subword candidates. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 66–75, Melbourne, Australia. Association for Computational Linguistics.

Taku Kudo and John Richardson. 2018. SentencePiece: A simple and language independent subword tokenizer and detokenizer for neural text processing. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 66–71, Brussels, Belgium. Association for Computational Linguistics.

Amba Kulkarni and Anil Kumar. 2011. Statistical constituency parser for sanskrit compounds. In *Proceedings of the International Conference on Natural Language Processing*.

Amba Kulkarni and Anil Kumar. 2013. Clues from aṣṭādhyāyī for compound type identification. In *Proceedings of the Sanskrit Computational Linguistics Symposium*.

Anil Kumar. 2012. *An automatic Sanskrit compound processing*. Ph.D. thesis, University of Hyderabad.

Anil Kumar, V. Sheebasudheer, and Amba P. Kulkarni. 2009. Sanskrit compound paraphrase generator. In *The workshop on Panini and Paninians of 16th and 17th centuries, Pondicherry*.

Mirella Lapata and Frank Keller. 2004. The web as a baseline: Evaluating the performance of unsupervised web-based models for a range of NLP tasks. In *Proceedings of the Human Language Technology Conference of the North American Chapter of the Association for Computational Linguistics: HLT-NAACL 2004*, pages 121–128, Boston, Massachusetts, USA. Association for Computational Linguistics.

Tao Li and Vivek Srikumar. 2019. Augmenting neural networks with first-order logic. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 292–302, Florence, Italy. Association for Computational Linguistics.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Roberta: A robustly optimized bert pretraining approach.

John Lowe. 2015. The syntax of sanskrit compounds. *Language*, 91:e71–e115.

Mary McHugh. 2012. Interrater reliability: The kappa statistic. *Biochemia medica : časopis Hrvatskoga društva medicinskih biokemičara / HDMB*, 22:276–82.

Vipul Mittal. 2010. Automatic Sanskrit segmentizer using finite state transducers. In *Proceedings of the ACL 2010 Student Research Workshop*, pages 85–90, Uppsala, Sweden. Association for Computational Linguistics.

Yatin Nandwani, Abhishek Pathak, Parag Singla, et al. 2019. A primal dual formulation for deep learning with constraints. In *Advances in Neural Information Processing Systems*, pages 12157–12168.

Minh Van Nguyen, Viet Dac Lai, Amir Pouran Ben Veyseh, and Thien Huu Nguyen. 2021. Trankit: A light-weight transformer-based toolkit for multilingual natural language processing. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: System Demonstrations*.

Diarmuid Ó Séaghdha and Ann Copestake. 2009. Using lexical and relational similarity to classify semantic relations. In *Proceedings of the 12th Conference of the European Chapter of the ACL (EACL 2009)*, pages 621–629, Athens, Greece. Association for Computational Linguistics.

Paṇini. 500 BCE. Ashtādhyāyī.

F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. 2011. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830.

Matthew E. Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer. 2018. Deep contextualized word representations. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 2227–2237, New Orleans, Louisiana. Association for Computational Linguistics.

Girishkumar Ponkiya, Diptesh Kanojia, Pushpak Bhattacharyya, and Girish Palshikar. 2021. FrameNet-assisted noun compound interpretation. In *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*, pages 2901–2911, Online. Association for Computational Linguistics.

Girishkumar Ponkiya, Rudra Murthy, Pushpak Bhattacharyya, and Girish Palshikar. 2020. Looking inside noun compounds: Unsupervised prepositional and free paraphrasing. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 4313–4323, Online. Association for Computational Linguistics.

Girishkumar Ponkiya, Kevin Patel, Pushpak Bhattacharyya, and Girish Palshikar. 2018a. Treat us like the sequences we are: Prepositional paraphrasing of noun compounds using LSTM. In *Proceedings of the*

27th International Conference on Computational Linguistics, pages 1827–1836, Santa Fe, New Mexico, USA. Association for Computational Linguistics.

Girishkumar Ponkiya, Kevin Patel, Pushpak Bhattacharyya, and Girish K Palshikar. 2018b. Towards a standardized dataset for noun compound interpretation. In *Proceedings of the Eleventh International Conference on Language Resources and Evaluation (LREC 2018)*, Miyazaki, Japan. European Language Resources Association (ELRA).

Armin Ronacher. Opening the falsk. *URL: http://mitsuhiko.pocoo.org/flask-pycon-2011.pdf*.

Jivnesh Sandhan, Om Adideva, Digumarthi Komal, Laxmidhar Behera, and Pawan Goyal. 2022. Evaluating neural word embeddings for sanskrit. In *the 18th World Sanskrit Conference*. Canberra, Australia.

Jivnesh Sandhan, Amrith Krishna, Pawan Goyal, and Laxmidhar Behera. 2019. Revisiting the role of feature engineering for compound type identification in Sanskrit. In *Proceedings of the 6th International Sanskrit Computational Linguistics Symposium*, pages 28–44, IIT Kharagpur, India. Association for Computational Linguistics.

Pavankumar Satuluri and Amba Kulkarni. 2013. Generation of sanskrit compounds. In *Proceedings of the International Conference on Natural Language Processing*.

Vered Shwartz and Chris Waterson. 2018. Olive oil is made *of* olives, baby oil is made *for* babies: Interpreting noun compounds using paraphrases in a neural model. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 2 (Short Papers)*, pages 218–224, New Orleans, Louisiana. Association for Computational Linguistics.

Hrishikesh Terdalkar and Arnab Bhattacharya. 2019. Framework for question-answering in Sanskrit through automated construction of knowledge graphs. In *Proceedings of the 6th International Sanskrit Computational Linguistics Symposium*, pages 97–116, IIT Kharagpur, India. Association for Computational Linguistics.

Stephen Tratz and Eduard Hovy. 2010. A taxonomy, dataset, and classifier for automatic noun compound interpretation. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics*, pages 678–687, Uppsala, Sweden. Association for Computational Linguistics.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Remi Louf, Morgan Funtowicz, Joe Davison, Sam Shleifer, Patrick von Platen, Clara Ma, Yacine Jernite, Julien Plu, Canwen Xu, Teven Le Scao, Sylvain Gugger, Mariama Drame, Quentin Lhoest, and Alexander Rush. 2020. Transformers: State-of-the-art natural language processing. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 38–45, Online. Association for Computational Linguistics.

Zhilin Yang, Zihang Dai, Yiming Yang, Jaime Carbonell, Russ R Salakhutdinov, and Quoc V Le. 2019. Xlnet: Generalized autoregressive pretraining for language understanding. In *Advances in Neural Information Processing Systems*, volume 32. Curran Associates, Inc.

A Web-based tool

We deploy our pretrained models as a web-based tool. Figure 4 illustrates the web-based tool integrated with our best performing system. Input: "*aham pīta-ambaram namāmi*" (Translation: "I pray to Pītāmbara (Lord Vishnu)."") where '*pīta-ambaram*' is a compound word. Our interface shows predicted morphological tags (color-coded with violet boxes), type-wise system confidence (bar plot), attention heatmaps. Our tool helps the user interpret the model prediction using model confidence on each semantic type.

B Marathi annotation details

We have built a user-friendly annotation tool for general-purpose classification tasks. The tool is a Flask-based (Ronacher) user-friendly web application styled by Bootstrap 5 (boo, 2022), and sports a simple administrative interface that lets the administrators easily control the class labels and classification context as well as export annotations. Figure 5 shows our web-based mobile friendly annotation interface, where the task is to select the correct option from multiple-choices for the given compound with the context. There are 4 broad categories of semantic types in Marathi. In case of confusion, we gave additional option to mark as *"Not sure"*. We also provide an option to add a comment to convey additional information about ambiguity or the concern for the corresponding example.

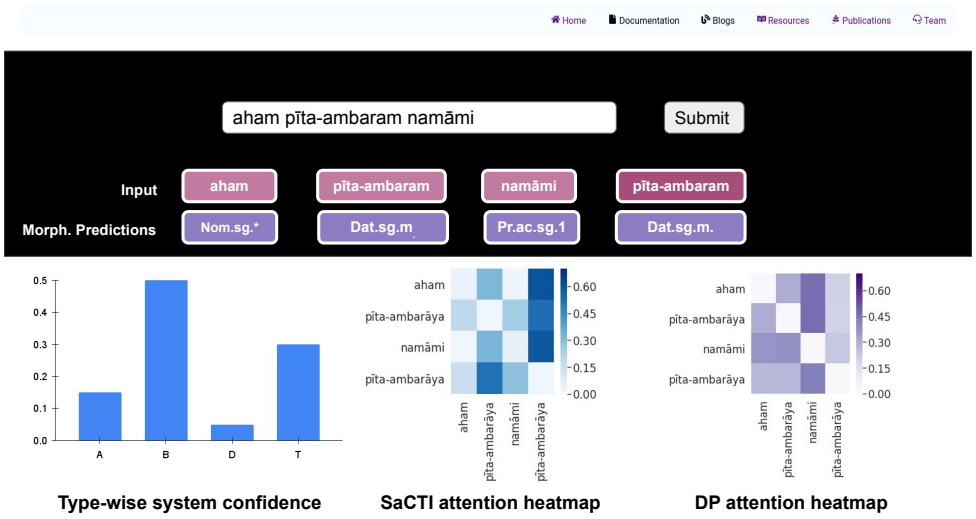

Figure 4: Illustration of web-based tool integrated with our best performing pretrained system. Input: "*aham pīta-ambaram namāmī*" (Translation: "I pray to Pītāmbara (Lord Vishnu).") where '*pīta-ambaram*' is a compound word. Our interface shows predicted morphological tags (color-coded with violet boxes), type-wise system confidence (bar plot), attention heatmaps.

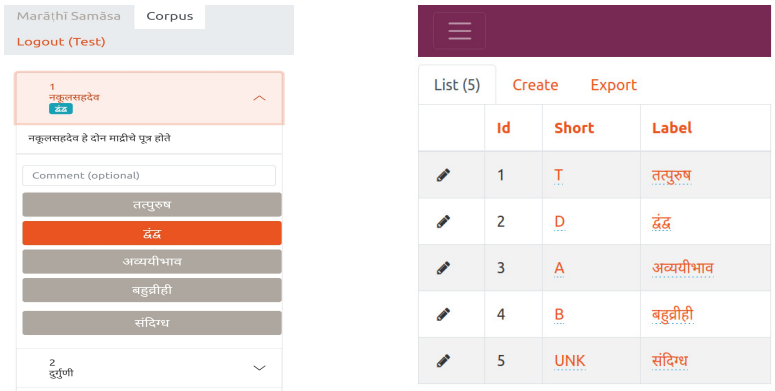

(a) (b)

Figure 5: Illustration of our web-based mobile friendly (a) annotation interface, where the task is to select the correct option from multiple-choices for the given compound with the context. (b) Administrative interface, where the administrators can easily control the class labels and classification context as well as export annotations.

Testing Large Language Models on Compositionality and Inference with Phrase-Level Adjective-Noun Entailment

Lorenzo Bertolini, Julie Weeds and **David Weir**
University of Sussex
Brighton, UK
{l.bertolini, juliwe, d.j.weir}@sussex.ac.uk

Abstract

Previous work has demonstrated that pre-trained large language models (LLM) acquire knowledge during pre-training which enables reasoning over relationships between words (e.g, hyponymy) and more complex inferences over larger units of meaning such as sentences. Here, we investigate whether lexical entailment (LE, i.e. hyponymy or the *is a* relation between words) can be generalised in a compositional manner. Accordingly, we introduce PLANE (**P**hrase-**L**evel **A**djective-**N**oun **E**ntailment), a new benchmark to test models on fine-grained compositional entailment using adjective-noun phrases. Our experiments show that knowledge extracted via In–Context and transfer learning is not enough to solve PLANE. However, a LLM trained on PLANE can generalise well to out–of–distribution sets, since the required knowledge can be stored in the representations of subwords (SW) tokens.

1 Introduction

Composition and entailment are crucial features of human language and reasoning. The first refers to the ability to combine units of meaning, like words or phrases, into larger constructs, such as sentences or paragraphs. Entailment, on the other hand, refers to the notion of inference. A linguistic element A (e.g. a word or phrase) is said to entail an element B if, assuming A is true, so is B. Word-level entailment is often referred to as lexical entailment (LE), *hypernym* detection, or the *is a* relation (Weeds et al., 2014; Vulić and Mrkšić, 2018; Kober et al., 2021), and refers to examples such as dog *entails* ($\models$) animal and gun $\not\models$ weapon. Yet entailment does not occur just between two words, and has been a long standing problem in NLP (Dagan et al., 2005; MacCartney and Manning, 2008; Marelli et al., 2014; Nie et al., 2020). When occurring between two sentences, it is usually referred to as natural language inference (NLI).

Although arguments have been made in favour of a more probabilistic interpretation of the task (see Pavlick and Callison-Burch (2016); Pavlick and Kwiatkowski (2019), inter alia), NLI benchmarks generally abandoned the rigid binary classification for a three way classification, usually involving a `neutral` or `UNK` label[1]. With a few exceptions, e.g., Baroni et al. (2012); Kartsaklis and Sadrzadeh (2016); Kober et al. (2021), NLI is still the main method adopted by the NLP community to jointly study the compositional and inferential abilities of a model. However, commonly used benchmarks frequently contain spurious statistical associations that a model can use to solve the task (Poliak et al., 2018; Dasgupta et al., 2018; McCoy et al., 2019). These cues might be as simple as the presence of negation or lexical overlap (Dasgupta et al., 2018), but can be more complex, and exploit similar syntactic substructures between premise and hypothesis (McCoy et al., 2019).

Popular alternatives to training and testing models on datasets containing significant biases are prompting (Petroni et al., 2019; Do and Pavlick, 2021; Hanna and Mareček, 2021) and In–Context learning (Brown et al., 2020). The success of these paradigms has grown in parallel with the popularity of large language models (LLMs) based on Transformers (Vaswani et al., 2017). LLMs architectures are usually pre-trained with mask or next-sentence prediction tasks, and later fine-tuned on other downstream tasks. Pre-trained LLMs have been successfully used with a prompt-based framework to extract factual information (Petroni et al., 2019) (e.g. Dante, `born_in`, Italy), LE relations (Bouraoui et al., 2020; Hanna and Mareček, 2021) (e.g. car, *is a*, vehicle) and study the more complex entailment in Winograd-style schemata (Do and Pavlick, 2021). Here, we study the impact that pre-training, NLI tuning and supervised learning have on the

[1]Usually matched with `entailment` and `non-entailment/contradiction`.

Proceedings of the 29th International Conference on Computational Linguistics, pages 4084–4100
October 12–17, 2022.

performance of a LLMs tested on compositional entailment, using adjective–noun phrases. That is, we investigate at which stage a LLMs might learn that *red car* $\models$ *vehicle*, as well as *red car* $\models$ *red vehicle*; whilst *fake gun* $\not\models$ *weapon*, even though *fake gun* $\models$ *fake weapon*.

Our main contributions are as follow. First, in Section 3, we introduce PLANE (**P**hrase–**L**evel **A**djective–**N**oun **E**ntailment), a large and automatically annotated resource to evaluate models on phrase–level compositional entailment for the English language. We then provide consistent evidence that knowledge acquired by LLMs during the pre-training phase (Section 4), and during finetuning on NLI tasks (Section 5) is weak, yielding poor and unstable performances on PLANE. In contrast, we show in Section 6 how, in a supervised setting, a model like BERT can effectively generalise to out-of-distribution test sets, and how crucial the role of subword (SW) tokens is to this ability. Finally, our work underlines how the different logical functions associated with the three macro classes of adjectives, frequently ignored or oversimplified, can pose notably different challenges to these models.

2 Related Work

Prompting Among the vast literature on prompting LLMs, the work from Hanna and Mareček (2021) is closely related to ours, and provides evidence that BERT retains information on the hyponym-hypernym relation occurring between two words. The work also shows how crucial the structure of the prompt can be. Garí Soler and Apidianaki (2020) provide evidence on the rich representations that BERT has about scalar adjectives and their intensity. Do and Pavlick (2021) propose a set of detailed entailment-based experiments, using both prompting and finetuning paradigm. Here, Winograd-like scenarios are used to carefully construct sentences that challenge LLM's internal association between two entities. Results strongly suggest that, once a model is not able to rely on those learned associations, the task becomes challenging even after finetuning.

Phrase entailment Compared to NLI, phrase-level entailment (PLE) has received significantly less attention. Baroni et al. (2012) present a set of experiments on compositional entailment considering adjective (e.g., `BIG` *dog* $\models$ *dog*) and quantifier modifications (e.g., `ALL` *dogs* $\models$ `SOME` *dogs*). However, instances were strictly limited to AN $\models$ N, and the class of the modifying adjectives was not discussed or differentiated in the results. Kartsaklis and Sadrzadeh (2016) introduced a manually annotated dataset for PLE, using subject-verb, verb-object, and subject-verb-object phrases. Negative samples were built by reversing each entailment item. In contrast, in our dataset, the label of an item can not be inferred by directional clues (i.e. hyponym-hypernym vs hypernym-hyponym) or by the absence of the hypernym relation between constituent words (e.g. *big cat* $\not\models$ *dog* because *cat* $\not\models$ *dog*). Kober et al. (2021) showed how automatically constructed compound-noun and AN compositional items can be used as a data augmentation method to enhance LE. However, this work filtered out intensional adjectives and assumed that for all other adjectives, $N \models h(N) \implies AN \models h(N)$. AN phrases were also studied within the context of fully formed sentences. The main example is the work from Pavlick and Callison-Burch (2016), that introduced the AddOne dataset. Overall, AddOne resemble the standard NLI benchmark, with sentence as premise and hypothesis, used to formulate a three way (`entailment, non-entailment, UNK`) classification task. However, in this case premise and hypothesis differ only by the presence or absence of a single adjective. Apidianaki and Garí Soler (2021) probed BERT with AddOne to study how it encodes the property of a noun. In contrast, we study the different entailment relations which are valid for different classes of adjectives.

3 PLANE

In this section, we describe the PLANE benchmark. We first outline how each of the three classes of adjectives, intersective (I), subsective (S) and intensional (O), affects the relation between a noun and its hypernym, as well as the noun itself. We then describe the sources used to gather adjectives, nouns, AN phrases, and hypernyms, and the procedure used to generate entailment items.

3.1 Adjective Classes

Adjectives can be divided into three macro classes: intersective (I), subsective (S) and intensional (O). From an entailment perspective, the distinction is based on how they modify a noun, N, with respect to itself as well as with respect to it's hypernyms ($hyps(N)$) (McCrae et al., 2014; Lalisse

	Inference Type (IT)	Intersective (I)	Subsective (S)	Intensional (O)
1	AN $\models$ N	✓	✓	✗
2	AN $\models$ h(N)	✓	✓	✗
3	AN $\models$ Ah(N)	✓	✗	✓

Table 1: PLANE annotation rules. Schema of how the interaction between each adjective class and inference type shapes the truth value – positive (✓) or negative (✗) – of a true noun (N) – hypernym (h(N))) entailment ($\models$) pair.

and Asudeh, 2015). We focus on three inference types, summarised in Table 1, all starting from an adjective-noun (AN) phrase.

AN phrases containing **intersective** (**I**) adjectives (e.g., *red*, *dead* and *Finnish*) describe a subset of entities subsumed by the noun itself and also a subset of entities which all have that adjective as a property. For example, a *red car* is both a *car* and a *red thing*. Thus, AN phrases containing intersective adjectives satisfy all of the forms of inference types (IT) shown in Table 1. Continuing our example, *red car* $\models$ *car* (IT 1), *red car* $\models$ *vehicle* (IT 2) and *red car* $\models$ *red vehicle* (IT 3).

Phrases with **subsective** (**S**) adjectives (e.g., *small*, *intelligent* and *strong*), describe a subset of entities subsumed by the noun but not a subset of entities which have that adjective as a property. For example, a *small elephant* is an *elephant* but it is not necessarily a *small thing*. Thus, AN phrases containing subsective adjectives satisfy IT 1 and 2 inferences but not IT 3 inferences listed in Table 1. In our example, whilst *a small elephant* $\models$ *elephant* and *small elephant* $\models$ *animal*; *small elephant* $\not\models$ *small animal*.

Intensional (**O**) adjectives (e.g. *fake*, *former*, *possible*) have the exact opposite behaviour of subsective. When an intensional adjective modifies a noun, it negates some of its core properties (e.g. *fake gun* $\not\models$ *gun*) and thus IT 1 inferences do not hold. Inferences with IT 2 also do not hold for intensional adjectives since the modification also directly applies to the hypernym of the noun (e.g., *fake gun* $\not\models$ *weapon*). However, since the adjective modification describes a subset of entities fully disjoint from the noun itself, this new set is usually contained within the subset of entities described using the hypernym of the noun modified by the adjective (e.g., *fake Glock* $\models$ *fake gun* $\models$ *fake weapon*) and thus IT 3 holds.

As in LE, we consider PLE as a binary classification task. We note that an argument on the probabilistic nature of PLE as in Pavlick and Kwiatkowski (2019) could be made. In our modelling scheme, *former president* $\models$ *politician*, and *small mouse* $\models$ *small animal* are formally false (McCrae et al., 2014); but, in the real world, might be judged to be unknown or true. We take the position that these cases require additional knowledge in order to judge them to be true. A *small mouse* $\models$ *small animal* because our knowledge suggests that *mouse* $\models$ *small animal*, and the modification of *mouse* by *small* does not change this. In this work, we assume that only LEs between unigrams are known a priori. We then consider whether LLMs contain the knowledge which will enable us to reason over necessary entailment between AN phrases. Therefore, in our binary classification task, the `negative` label covers all cases which might be judged in the real world to be false, unknown or dependent on additional knowledge.

We now present how the evaluation dataset has been constructed, starting from the source of adjectives (A), adjective-noun (AN) phrases and hypernyms ($hyps(N)$).

3.2 Sources

Adjectives Our main source is the list provided by Lalisse and Asudeh (2015), consisting of 300 items in English. Each adjective is tagged with its class, whether it is weakly or strongly polysemous, and/or context dependent[2]. Further intensional adjectives were added from the dictionary in Kennard et al. (2014). After filtering out all adjectives tagged as context-dependent, wee remained with a total of 312 unique items.

Adjective-Noun phrases To collect compositional and realistic AN phrases[3] we parsed a clean Wikipedia dump (Wilson, 2015) via Spacy[4] (Honnibal and Johnson, 2015). We then filtered out all phrases where the identified adjective was not in the adjective list previously described.

[2]The class of an adjective can vary according to the context or the noun it modifies. Deep, for example, can be intersective, as in *deep lake*, or subsective, like in *deep thinker*)

[3]See Appendix A for further analysis.

[4]We used the `en_core_web_lg` model.

Hypernyms We used Wordnet (Fellbaum, 1998) via the NLTK API to collect nouns' hypernyms. We first filtered out AN phrases that were potentially mislabelled by Spacy as containing a noun, by searching for `noun` synsets. We then queried Wordnet for hypernyms of the noun ($hyps(\texttt{N})$), up to a maximum path distance of 3 and always following the first synset. For AN phrases containing an intensional (O) adjective, this procedure was limited to direct hypernyms (i.e. hypernyms with path distance 1 from the noun). This is to mitigate the fact that IT 2 and 3 inferences might not be always false/true for this class of adjectives. As an example, consider the phrase *alleged thief*. In line with our previous discussion, *alleged thief* `is_not_a` *thief* and *alleged thief* `is_a` *alleged criminal*. However, as we move up the hypernym hierarchy, we find *alleged thief* `is_a` *person*, and *alleged thief* `is_not_a` *alleged person*.

We then filtered out any hypernyms that were already in bigram or multi-word-expressions (MWE) form. Although they present an interesting resource for future investigation, here we focus on the set of unigram hypernyms, to control more precisely the automatic construction of items and mitigate the possibility of including idiomatic phrases. Lastly, test items were further restricted to instances containing nouns occurring at least once within each adjective class. This was done to control for results determined solely by possible strong/weak noun–adjective associations.

Inference Types Once the hypernyms ($hyps(\texttt{N})$) for each `AN` were collected, we automatically constructed all possible positive (✓) and negative (✗) items following the rules presented in Table 1. This converts triplets of the IT <`A`, `N`, $h(\texttt{N})$> where $h(\texttt{N}) \in hyps(\texttt{N})$ into triplets of the IT $<$ c_1, c_2, `label`> where c_1 is the `AN` phrase, c_2 is one of `N`, $h(\texttt{N})$ or `A`$h(\texttt{N})$ and `label` indicates whether an entailment holds between c_1 and c_2.

The final PLANE dataset contains 312 unique adjective, ∼7800 unique nouns and approximately 1.9M unique inference items. The complete benchmark and code for the experiments are openly available[5]

4 In–Context Learning

In this section we investigate the ability of multiple LLMs to solve compositional entailment without any target training. To do so, we adopt an In-Context learning paradigm. With a similarly aim, Hanna and Mareček (2021) evaluated a model's performance on LE by testing if it was able to unmask a prompt *P* such as "A *x* is a `[MASK]`" with a correct hypernym of *x*. Given the phrasal nature of our investigation, we structure our prompts to ask the model whether a particular instance is a *positive* (✓) or *negative* (✗) example of an entailment pair.

Results from Hanna and Mareček (2021) and preliminary Zero-Shot experiments (See Appendix B.1) suggest the performance of a model may be largely affected by its lack of understanding of the task, or particular words in the prompt. Thus, we experiment with a Two-Shot NLI-like format, providing models with some solved examples and background knowledge about entailment, involving the lexical items in the hypothesis. More specifically, we adopt a prompt *P* consisting of two 'labelled' premises and one 'unlabelled' hypothesis, e.g.,:

p_1 : A big car is a good example of a car.

p_2 : A big car is a poor example of a big vehicle.

h : A big car is a `[MASK]` example of a vehicle.

As in the example, each of the three components of *P* (i.e. the two premises and the hypothesis) has a unique inference type (IT). We structure the prompts in this way for two reasons: i) to independently study each <`A`, `N`, $h(\texttt{N})$ > triplets generating every $< c_1, c_2,$ `label` >; ii) investigate if a context that facilitate the identification of an adjective's class, also yields better performances. In the example above, even if a model has no knowledge on the adjective *big*, but knows how subsective (S) adjectives work, it can directly infer from the premises the class of *big*, and, hence, the correct label for the hypothesis. However, if p_2 and h were inverted, the only way a model could solve the instance would be knowing how subsective adjectives work and that *big* is subsective. Lastly, to investigate potential recency effects of the two premises, we query each model with the presented prompt and one with inverted p_1 and p_2. For example, given a hypothesis with IT 3, we consider both premises with IT 1,2 and premises with IT 2,1.

Since *P* contains labelled examples, models can observe the expected label within the given sample. We hence define a set of label's verbalisers for positive (✓), and one for the negative (✗) labels.

[5] https://github.com/lorenzoscottb/PLANE

	Internal	External
$p_{1,2}$	A $\{c_1\}$ is a {verbaliser} example of a $\{c_2\}$.	A $\{c_1\}$ is a type of $\{c_2\}$:{verbaliser}.
h	A $\{c_1\}$ is a [MASK] example of a $\{c_2\}$.	A $\{c_1\}$ is a type of $\{c_2\}$:[MASK].

Table 2: Prompt templates (PT). The two premises (p_*) – hypothesis (h) structures used in the Two–Shot experiment. $c_{1,2}$ refer to the head and tail components of a given inference type (IT) (see Table 1 for reference).

We experiment with two prompt templates, and three label verbalisers, presented in Table 2 and 3 respectively. Given a prompt P, its label l, we define the task as the ability of an LLM to generate, as first prediction for the [MASK], the token t that corresponds to the correct verbaliser for l. Performance is computed via F1 score, since it is possible that t will be different from either of the correct verbalisers.

	✓	✗
GP	good	poor
TF	true	false
PN	positive	negative

Table 3: Labels' verbalisers. Tokens used to verbalise positive (✓) and negative (✗) labels in the Two–Shot experiment.

Selected Models We focus on three families of Transformer networks: BERT (Devlin et al., 2019), RoBERTa (Liu et al., 2019) and Distillation based (Sanh et al., 2019) (i.e. DistillBERT and DistillRoBERTa). For RoBERTa and BERT, we considered both base and large models (cased for BERT). Models for this and later experiments were all implemented via Hugging Face (Wolf et al., 2020). All experiments were run on a NVIDIA GeForce RTX 3090.

Results Results from individual models are presented in Table 4, divided by adjective classes. Overall, the performance is fairly poor, with all models presenting low average scores, and remarkable variances across classes. As discussed below, part of this is surely generated by the different verbalisers and prompt templates adopted in the experiment. However, a non-irrelevant part of this variance seems to be directly explained by the adjective class itself. As we can see, all models follow the trend in results associated with each class, finding intersective (I) adjective examples easiest, followed by intensional (O) and then subsective (S). Indeed, the first might not come as a surprise, especially as I adjectives are always associated with a positive label. Yet the fact that rare and anomalously–behaving adjectives such as the intensional (O) ones seem to be easier to deal with than subsective (S), the predominant class in human language, is more unexpected.

PT and verbaliser analysis Across models, prompt templates (PT) and verbalisers have remarkably differed effects on each class and IT (see Figure 6 in Appendix B.2 for summary). Most PT–verbaliser combinations yield almost flawless performances on intersective (I) adjectives, suggesting LLMs are generally keen to choose the same label appearing in both premises. In this class, the variance derives almost entirely from the PN verbaliser. As intersective adjectives are associated just with positive (✓) labels, this evidence suggest a possible association of PN with negative solutions.

Results from subsective (S) items point to similar conclusions. First of, almost all PT–verbaliser combinations struggle to solve instances where the hypothesis has IT 3. That is, when the hypothesis presents the opposite label to both premises. Moreover, PN seems to be again associated with a tendency towards negative labels, especially if combined with the External PT. Such combination is the only one improving the performance on IT 3, but severely damages all other inferences.

In intensional (O) adjectives, where most IT have negative (✗) labels, this association partially affects the TF verbaliser too. However, most models still fail where the hypothesis has opposite label to both premises (IT 3). Overall, this suggests that, whenever presented with premises sharing the same label, regardless of which, models tend to overcome possible internal associations, and opt to repeat the presented label.

Concluding, we note conflicting observations on the recency effect, expected to emerge when p_2 and h share the same label (see Figure 7 in Appendix B.2). The effect has a mostly positive impact on the GP verbaliser (in S and O classes), but contradictory effects on the others, especially PN.

Adj. Class	BERT-base	BERT-large	DistillBERT	DistillRoBERTa	RoBERTa-base	RoBERTa-large
I	69.9 ± 38	78.2 ± 34	70.9 ± 36	83.79 ± 32	97.6 ± 5	99.4 ± 1
S	40.1 ± 36	48 ± 36	34.4 ± 40	19.6 ± 27	38.3 ± 32	41.4 ± 42
O	59.3 ± 40	54. 4 ± 37	52.6 ± 46	61.6 ± 40	48.1 ± 34	44.8 ± 38
Average	56.5 ± 40	60.2 ± 43	52.6 ± 43	55 ± 42	61.3 ± 37	61.9 ± 42

Table 4: Two–Shot learning results. Mean F1 scores (± standard deviation, obtained collapsing prompts' and verbalisers' results) of individual models on the Two–Shot learning experiment, divided by adjective class.

5 Transfer Learning

Evidence from Section 4 suggest In–Context learning is too susceptible to internal correlations and biases to be reliable. Since models trained to classify text for entailment are very popular, we next investigate whether tuning a LLM for sentence level entailment can provide enough information to reliably solve phrase-level entailments from PLANE. For comparison, we re-use the same test from the Two–Shot experiment, re-framing the task as a standard NLI text classification. We replace the standard premises-hypothesis input sentences with a $< c_1,c_2 >$ pair, and evaluate a model's performance in classifying each scenario as presenting an entailment or not. We adopt F1 scores, since, in contrast to PLANE's binary classification, NLI also has a third label (2). This label, often referred to as neutral or `UNK`, usually denotes instances where annotators could not agree on the presence or absence of entailment (1).

Selected Models In the experiment, we use Liu et al. (2019) and Nie et al. (2020) RoBERTa models, both fine-tuned to run NLI-like tasks, and a RoBERTa-base model we tuned on the AddOne benchmark from Pavlick and Callison-Burch (2016). As mentioned, AddOne was designed to study AN composition in the context of full sentences, using premises and hypothesis that differ by a single adjective.

Results Table 5 summarises the results, which appear contrasting. Nie et al. (2020)'s performance is fairly in line with average results of RoBERTa models in the Two–Shot setting (see Table 4). However, in this setting, subsective (S) items seem to obtain a far better performance, especially with respect to intensional (O), suggesting a strong shift towards positive solutions. On the other hand, it appears NLI tuning had a negative impact on Liu et al. (2019)' model. The very high performance observed for intensional adjective strongly suggest a strong preference for contradiction label, as suggested by the error analysis (see Figure 9 in C.1 for visual summary. As for the model tuned on AddOne, the same analysis confirmed that the poor performance across the board depends on a strong preference for neutral labelling. Interestingly, we found that all models share a pattern of predictions for neutral (2) labels (see Figure 9 in C.1 for visual summary). When presented with subsective (S) adjectives, neutral mislabelling is more frequent with positive items, whilst the opposite is true for intensional (O) ones.

Adj. Class	Liu et al. 19	Nie et al. 20	AddOne
I	17.1	90.3	35.9
S	24.1	58.8	32.2
O	57.4	31.1	25.5
Average	32.8	60	31.2

Table 5: Testing NLI models results. F1 scores, divided by adjective class, of RoBERTa models tuned on different NLI benchmarks, and tested on phrase-level entailment. The test set consists of PLANE items used in the Two–Shot experiment.

Variance analysis As mentioned in the introduction, multiple work (e.g. Dasgupta et al. (2018); McCoy et al. (2019)) have shown how biases can arise from syntactic structures. To investigate if the structure of an instance (i.e., the IT) has an impact on each model's performance, we investigate the results divided by adjective class and ITs. The results are summarised in Figure 1.

First off, the image clearly shows the preference of Liu et al. (2019) for negative labels and Nie et al. (2020) for positive ones. Interestingly, we can also see how, at least for these two models, these preferences are strongly accentuated under inference type 1. This effect could be related to the lexical overlap heuristic described in McCoy et al. (2019). This heuristic refers to those instances where the hypothesis (h) contains multiple words from the premise (p), especially within its first tokens. Inference type 1 (i.e. AN $\models$ N) could elicit this bias since h is simply a partial repetition of p.

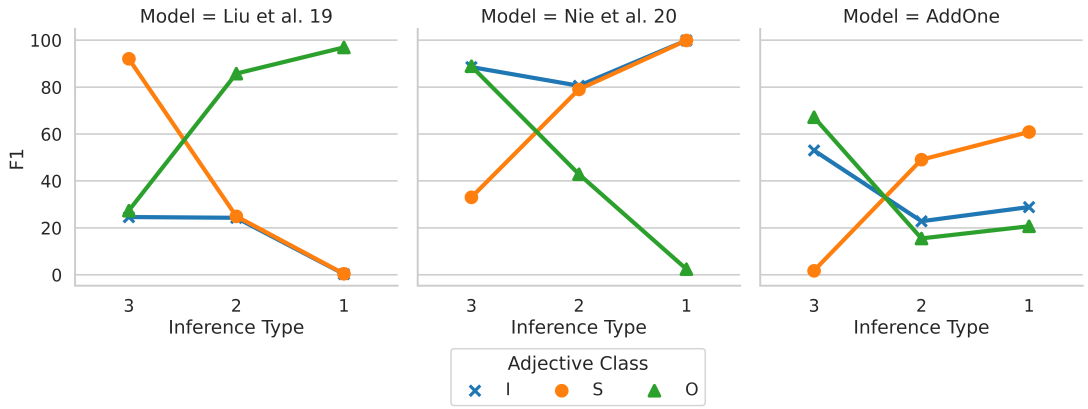

Figure 1: Transfer–Learning variance analysis. Visualisation of the variance observed in different models tuned on NLI-based datasts (column), with respect to each adjective class (hue) and inference type (x axis).

However, McCoy et al. (2019) found that in MNLI (Williams et al., 2018) – Liu et al. (2019)'s training data – such heuristic was mainly associated with a positive label, which is in contrast with our results. It would however partially explain why this behaviour is not expressed by the model trained on AddOne, where the lexical overlap is close to 100% by design, so that a model can not use the heuristic at all. Lastly, it is worth noting how in the model trained on AddOne, subsective (S) adjective display an almost specular pattern to intersective (I) and intensional (O). Observing opposite patterns between S and O is not surprising, as they have opposed labels with respect to each IT (see Table 1). What is unexpected is that I adjectives, always associated with positive labels, produce results almost identical to those of O, where only IT 3 presents a positive label.

6 Supervised Learning

As suggested in McCoy et al. (2019), and supported by preliminary experiments (see Appendix D), drawing the test set from the same distribution of the train set likely over–simplifies the task for LLMs. Hence, to study the performance of a model in a supervised setting, we focus on it's ability to generalise out of distribution (GOoD). Furthermore, we conducted an experiment using a setting where structural cues as the inference types (IT) have been removed (One–IT). As LLMs' vocabularies contain a significant amount of subwords (SW) tokens, together with word tokens, we provide an analysis on the impact of SWs on the model's performance. Following the work of Hanna and Mareček (2021), Do and Pavlick (2021), Apidianaki and Garí Soler (2021), we focus the supervised experiments on a BERT-base model.

6.1 Generalise Out of Distribution (GOoD)

We use PLANE to generate splits where the vocabularies (i.e. adjective, noun, and hypernyms) used in the training and test set do not overlap. That is, each adjective, noun, and hypernym is unique to either the train or the test set. We frame the task as a sequence classification. Following preliminary experiment (see Appendix D), input length is set to 12. We collect 5 different (and openly available) train–test splits, and train the model for 1 epoch. Results are displayed in the left column of Table 6.

Compared to previous results, the performance is strong, remarkably more stable, and is well above chance. The training regime still contains potential structural biases (the ITs), that can facilitate the solution. Yet those cues are useless if not correctly combined with the class of an adjective. Given that single word memorisation is excluded by design, one could assume an effect of pre-training. However, this seems unlikely, given earlier results. Another possibility is that inferences are being made which rely in some way on the constituent subword (SW) tokens of otherwise unseen lexical items.

Training Setting	GOoD	One–IT
Accuracy	.85 ± .05	.86 ± .01

Table 6: Fientuning results. Accuracy (mean ± standard deviations) obtained by BERT, when finetuned on different PLANE–generated splits, in the full generalise out of distribution (GOoD) and One–IT GOoD setting.

Subwords analysis To study the impact of subwords, we compute the accuracies obtained in each test split by BERT, divided by adjective class, and

compare them against the percentage of test instances containing SWs. The results are displayed in Figure 2.

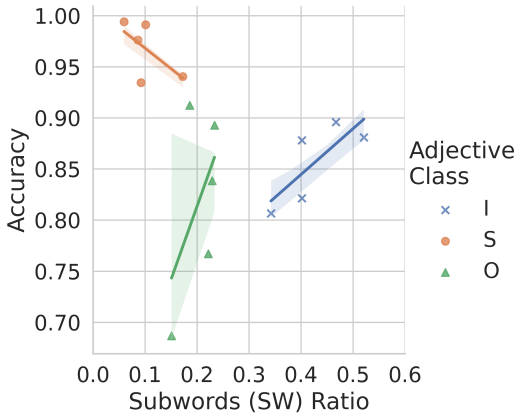

Figure 2: Subword (SW) analysis in GOoD training setting. Analysis of the relation between the amount of sequences containing WP and the accuracy obtained by BERT in each of the five GOoD test splits, divided by adjective class.

We begin noticing that each class seems to cluster around fairly specific SW ratios, which might already facilitate the correct classification of a given input. In sequences with subsective (S) adjectives, SWs are actually all related to nouns and/or hypernyms. This seems to create strong biases that, in the absence of SWs in the adjective position, would suggest to the model that the adjective is subsective, and, hence, the solution. The negative impact that subwords have on S instances might be further explained by the fact that up to 60% of the N/h(N) SWs set overlaps with SWs used in I and 0 adjectives.

A similar overlap also affects intensional (O) adjectives. Up to 65% of adjective subwords overlap with the subwords (SW) used by nouns and hypernyms, and circa 28% also overlap with SWs used for I adjectives. This suggests that, although minimal, an increase in subwords could help the model to identify the correct class of an instance.

Intersective (I) adjectives present the highest ratio of subwrds. Despite the set of adjectives and nouns/hypernyms SWs have similar length, the overlap is very low – between 10 and 7%. This would allow the model to directly exploit SWs to deduct the correct class of an adjective.

6.2 One–IT

The test sets from previous experiment still contained structural cues (ITs) that could assist the model. To study the impact of those cues, we collect new training and test sets, using solely IT 3. We focus on IT 3 as it is the only subset of PLANE where intersective and subsective adjectives, the two largest classes, present opposite labels. Similarly to previous experiment, we balance the number of positive and negative labels, and assure that nouns and hypernyms do not act as cues. We sample five train-test splits, and train with same settings of previous experiment. Results are presented in the right column of Table 6.

The absence of structural cues yields very similar results to the ones from previous experiment, with lower standard deviation and seemingly more stable. Results divided by single split and classes are displayed in Figure 3. SW analysis is also carried out for this training regime, adopting the same setting as in Section 6.1.

Subwords analysis In this setting, Intersective (I) adjectives reached a lower performance, and present a weaker correlation between accuracy and SW ratio. A possible explanation involves the large overlap – circa 50% – in the set of SWs used for adjectives and nouns/hypernyms. Furthermore, the number of instances with and without SWs are remarkably similar, making it potentially difficult to use subwords' presence as cue.

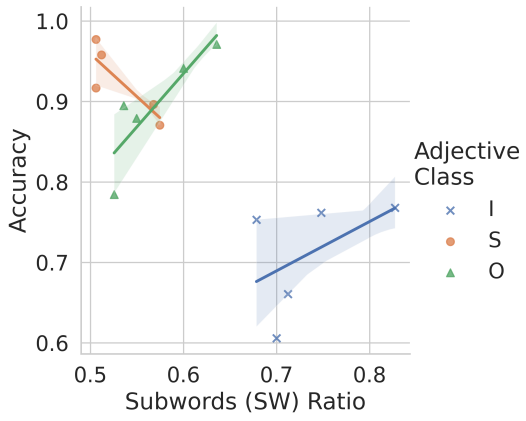

Figure 3: Subword (SW) analysis in One–IT training setting. Analysis of the relation between the amount of sequences containing WP and the accuracy obtained by BERT in each of the five One–IT GOoD test splits, divided by adjective class.

In this experiment, we did found a set of subsec-

tive (S) adjectives containing subwords. However, this set is very small, so the absence of SWs in the adjective position could bias the decision towards negative (✗) labels. Such minimal increase could however act as distractor, explaining the steeper slope of the regression (orange) line.

Once again, O class has the most marked interaction between accuracy and SW ratio. However, in this case, instances with a SW ratio similar to S items do not seem affected. A possible explanation is the very restricted set of SWs (33) used for these adjectives. This smalls set could facilitate an adjective's classification, hence producing a correct solution.

7 Discussion and Conclusion

Adjectives can be grouped in three macro classes. From a logical and linguistic perspective, these classes shape the truth value of a lexical entailment (LE) pair as *dog* $\models$ *animal* in multiple ways, depending on the class and the structure of said inference, as presented in Table 1. This versatility provides a valuable resource to study composition and inference with great detail and control, but was often oversimplified. As previous evidence suggest large language models (LLM) are able to retain word-level entailment information (Petroni et al., 2019; Hanna and Mareček, 2021), we designed a resource to study if LLMs can tackle fine–grained compositional inference, with AN entailment.

Results based on In–Context learning suggest that LLMs' performance is too unstable, and frequently relying on pre-existing word associations or labelling patterns. Conclusions are not so different with models tuned to classify text for entailment. As Section 5 strongly suggests, after tuning a model for sentence–level inference, the knowledge is hardly transferable to the same task at phrase level. These evidence are likely connected to how AN phrases behave within the context of fully formed sentences (Pavlick and Callison-Burch, 2016). From a logical stand, *Japanese economy* $\models$ *economy*. Yet, given a sentence as "*Bush travels Monday to Michigan to make remarks on the Japanese economy.*"[6], potential annotators might say it does not entail "*Bush travels Monday to Michigan to make remarks on the economy.*". Of course this and similar scenarios are influenced by complex commonsense and pragmatical knowledge. Yet this opens interesting questions on how AN phrases in and out of context are related to each other, whether a model should be able to correctly reason over both, and, most importantly, what can we do to make that happen.

Experiments with supervised learning and out–of–distribution test sets suggest that a LLM such as BERT can become robustly efficient, even in absence of structural cues. Our results strongly suggest that the solution is aided by subwords (SW) tokens. Aside from leaking some information to the test set, SW might create biases related to how they distribute in different adjective classes. This solution is computationally efficient and effective, but might pose some limits. This solution is simple, computationally efficient and effective. However, it is unlikely that it provides a theoretically sound model of natural language from the perspective of composition, especially since SW are rarely morphologically grounded (Hofmann et al., 2021, 2022). From a practical perspective, it also poses questions as to how we should define out-of-distribution sets when working with LLMs.

To conclude, we introduced PLANE, an extensive annotated resource to train and test models on compositional phrase-level entailment, using adjective-noun phrases. We provided evidence that knowledge learnt via pre-training or NLI tuning is insufficient to solve the task, and showed how, in a supervised setting, a model like BERT can learn to generalise out of distribution examples, adopting strategies connected to SW tokens. Future work will focus on extending In–Context learning to autoregressive LLMs, using PLANE to evaluate LE models on composition, and investigate a three-way or probabilistic labelling system.

Ethical and Broader Impact Statement

As the work has a mainly theoretical focus, authors do not foresee a significant ethical issue related to the set of experiment. However, we note that a number of intersective (I) adjectives refer to nationality (e.g. English, Italian, Japanese) and religious faith (e.g. Christian, Jewish). It is possible that phrases containing biases and/or stereotypes contained in the WikiDump we adopted might have accidentally ended up in the final version of PLANE. As for the broader impact, we believe our work makes two key contributions: i) offers a tool to investigate in grater detail adjective-noun phrases with respect to inference; ii) provides analyses and evidences in support of the need of taking into account the

[6] Example from Pavlick and Callison-Burch (2016)

distinction between adjective classes, as they pose clearly different challenges to the tested models.

Acknowledgements

This research was supported by the EPSRC project *Composition and Entailment in Distributed Word Representations* (grant no. 2129720), and the EU Horizon 2020 project HumanE-AI (grant no. 952026). We also thank the anonymous reviewers for their helpful comments and suggestions.

References

Marianna Apidianaki and Aina Garí Soler. 2021. ALL dolphins are intelligent and SOME are friendly: Probing BERT for nouns' semantic properties and their prototypicality. In *Proceedings of the Fourth BlackboxNLP Workshop on Analyzing and Interpreting Neural Networks for NLP*, pages 79–94, Punta Cana, Dominican Republic. Association for Computational Linguistics.

Marco Baroni, Raffaella Bernardi, Ngoc-Quynh Do, and Chung-chieh Shan. 2012. Entailment above the word level in distributional semantics. In *Proceedings of the 13th Conference of the European Chapter of the Association for Computational Linguistics*, pages 23–32, Avignon, France. Association for Computational Linguistics.

Zied Bouraoui, José Camacho-Collados, and Steven Schockaert. 2020. Inducing relational knowledge from BERT. In *The Thirty-Fourth AAAI Conference on Artificial Intelligence, AAAI 2020, The Thirty-Second Innovative Applications of Artificial Intelligence Conference, IAAI 2020, The Tenth AAAI Symposium on Educational Advances in Artificial Intelligence, EAAI 2020, New York, NY, USA, February 7-12, 2020*, pages 7456–7463. AAAI Press.

Tom B. Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, T. J. Henighan, Rewon Child, Aditya Ramesh, Daniel M. Ziegler, Jeff Wu, Clemens Winter, Christopher Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. 2020. Language models are few-shot learners. *ArXiv*, abs/2005.14165.

Ido Dagan, Oren Glickman, and Bernardo Magnini. 2005. The pascal recognising textual entailment challenge. In *MLCW*.

Ishita Dasgupta, Demi Guo, Andreas Stuhlmüller, Samuel J. Gershman, and Noah D. Goodman. 2018. Evaluating compositionality in sentence embeddings. *Proceedings of the 40th Annual Conference of the Cognitive Science Society, pages 1596–1601, Madison, WI.*

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, Minneapolis, Minnesota. Association for Computational Linguistics.

Nam Do and Ellie Pavlick. 2021. Are rotten apples edible? challenging commonsense inference ability with exceptions. In *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*, pages 2061–2073, Online. Association for Computational Linguistics.

Christiane Fellbaum. 1998. *WordNet: An Electronic Lexical Database*. Bradford Books.

Aina Garí Soler and Marianna Apidianaki. 2020. BERT knows Punta Cana is not just beautiful, it's gorgeous: Ranking scalar adjectives with contextualised representations. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 7371–7385, Online. Association for Computational Linguistics.

Michael Hanna and David Mareček. 2021. Analyzing BERT's knowledge of hypernymy via prompting. In *Proceedings of the Fourth BlackboxNLP Workshop on Analyzing and Interpreting Neural Networks for NLP*, pages 275–282, Punta Cana, Dominican Republic. Association for Computational Linguistics.

Valentin Hofmann, Janet Pierrehumbert, and Hinrich Schütze. 2021. Superbizarre is not superb: Derivational morphology improves BERT's interpretation of complex words. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 3594–3608, Online. Association for Computational Linguistics.

Valentin Hofmann, Hinrich Schuetze, and Janet Pierrehumbert. 2022. An embarrassingly simple method to mitigate undesirable properties of pretrained language model tokenizers. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 385–393, Dublin, Ireland. Association for Computational Linguistics.

Matthew Honnibal and Mark Johnson. 2015. An improved non-monotonic transition system for dependency parsing. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 1373–1378, Lisbon, Portugal. Association for Computational Linguistics.

Dimitri Kartsaklis and Mehrnoosh Sadrzadeh. 2016. A compositional distributional inclusion hypothesis. In *Proceedings of the 9th International Conference on Logical Aspects of Computational Linguistics. Celebrating 20 Years of LACL 1996—2016 - Volume 10054*, LACL 2016, page 116–133, Berlin, Heidelberg. Springer-Verlag.

Neha Nayak Kennard, Mark Kowarsky, Gabor Angeli, and Christopher D. Manning. 2014. A dictionary of nonsubsective adjectives.

Daniel Keysers, Nathanael Schärli, Nathan Scales, Hylke Buisman, Daniel Furrer, Sergii Kashubin, Nikola Momchev, Danila Sinopalnikov, Lukasz Stafiniak, Tibor Tihon, Dmitry Tsarkov, Xiao Wang, Marc van Zee, and Olivier Bousquet. 2020. Measuring compositional generalization: A comprehensive method on realistic data. In *International Conference on Learning Representations*.

Thomas Kober, Julie Weeds, Lorenzo Bertolini, and David Weir. 2021. Data augmentation for hypernymy detection. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*, pages 1034–1048, Online. Association for Computational Linguistics.

Mathias Lalisse and ash Asudeh. 2015. Distinguishing intersective and non-intersective adjectives in compositional distributional semantics.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Roberta: A robustly optimized bert pretraining approach. *ArXiv*, abs/1907.11692.

Bill MacCartney and Christopher D. Manning. 2008. Modeling semantic containment and exclusion in natural language inference. In *Proceedings of the 22nd International Conference on Computational Linguistics (Coling 2008)*, pages 521–528, Manchester, UK. Coling 2008 Organizing Committee.

Marco Marelli, Stefano Menini, Marco Baroni, Luisa Bentivogli, Raffaella Bernardi, and Roberto Zamparelli. 2014. A SICK cure for the evaluation of compositional distributional semantic models. In *Proceedings of the Ninth International Conference on Language Resources and Evaluation (LREC'14)*, pages 216–223, Reykjavik, Iceland. European Language Resources Association (ELRA).

Tom McCoy, Ellie Pavlick, and Tal Linzen. 2019. Right for the wrong reasons: Diagnosing syntactic heuristics in natural language inference. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 3428–3448, Florence, Italy. Association for Computational Linguistics.

John P. McCrae, Francesca Quattri, Christina Unger, and Philipp Cimiano. 2014. Modelling the semantics of adjectives in the ontology-lexicon interface. In *Proceedings of the 4th Workshop on Cognitive Aspects of the Lexicon (CogALex)*, pages 198–209, Dublin, Ireland. Association for Computational Linguistics and Dublin City University.

Yixin Nie, Adina Williams, Emily Dinan, Mohit Bansal, Jason Weston, and Douwe Kiela. 2020. Adversarial NLI: A new benchmark for natural language understanding. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 4885–4901, Online. Association for Computational Linguistics.

Ellie Pavlick and Chris Callison-Burch. 2016. Most "babies" are "little" and most "problems" are "huge": Compositional entailment in adjective-nouns. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 2164–2173, Berlin, Germany. Association for Computational Linguistics.

Ellie Pavlick and Tom Kwiatkowski. 2019. Inherent disagreements in human textual inferences. *Transactions of the Association for Computational Linguistics*, 7:677–694.

Fabio Petroni, Tim Rocktäschel, Sebastian Riedel, Patrick Lewis, Anton Bakhtin, Yuxiang Wu, and Alexander Miller. 2019. Language models as knowledge bases? In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 2463–2473, Hong Kong, China. Association for Computational Linguistics.

Adam Poliak, Jason Naradowsky, Aparajita Haldar, Rachel Rudinger, and Benjamin Van Durme. 2018. Hypothesis only baselines in natural language inference. In *Proceedings of the Seventh Joint Conference on Lexical and Computational Semantics*, pages 180–191, New Orleans, Louisiana. Association for Computational Linguistics.

Victor Sanh, Lysandre Debut, Julien Chaumond, and Thomas Wolf. 2019. Distilbert, a distilled version of bert: smaller, faster, cheaper and lighter.

Yulia Tsvetkov and Shuly Wintner. 2011. Identification of multi-word expressions by combining multiple linguistic information sources. In *Proceedings of the 2011 Conference on Empirical Methods in Natural Language Processing*, pages 836–845, Edinburgh, Scotland, UK. Association for Computational Linguistics.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Ł ukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc.

Aline Villavicencio, Valia Kordoni, Yi Zhang, Marco Idiart, and Carlos Ramisch. 2007. Validation and evaluation of automatically acquired multiword expressions for grammar engineering. In *Proceedings*

of the 2007 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning (EMNLP-CoNLL), pages 1034–1043, Prague, Czech Republic. Association for Computational Linguistics.

Ivan Vulić and Nikola Mrkšić. 2018. Specialising word vectors for lexical entailment. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 1134–1145, New Orleans, Louisiana. Association for Computational Linguistics.

Julie Weeds, Daoud Clarke, Jeremy Reffin, David Weir, and Bill Keller. 2014. Learning to distinguish hypernyms and co-hyponyms. In *Proceedings of COLING 2014, the 25th International Conference on Computational Linguistics: Technical Papers*, pages 2249–2259, Dublin, Ireland. Dublin City University and Association for Computational Linguistics.

Adina Williams, Nikita Nangia, and Samuel Bowman. 2018. A broad-coverage challenge corpus for sentence understanding through inference. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 1112–1122, New Orleans, Louisiana. Association for Computational Linguistics.

Benjamin Wilson. 2015. The unknown perils of mining wikipedia. Https://blog.lateral.io/2015/06/the-unknown-perils-of-mining-wikipedia/.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Remi Louf, Morgan Funtowicz, Joe Davison, Sam Shleifer, Patrick von Platen, Clara Ma, Yacine Jernite, Julien Plu, Canwen Xu, Teven Le Scao, Sylvain Gugger, Mariama Drame, Quentin Lhoest, and Alexander Rush. 2020. Transformers: State-of-the-art natural language processing. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 38–45, Online. Association for Computational Linguistics.

A PLANE: PMI analysis

To further control for non—compositional items, we performed a PMI analysis on PLANE's phrases. Tsvetkov and Wintner (2011) showed how higher values of PMI can indicate the presence of a multi–word–expression (MWE), whilst values below zero tend to refer to words that should not really co-occur. Villavicencio et al. (2007) compared the probability distributions of PMI scores from a set of MWE and non—MWE n—grams. The results showed how the distribution of MWE was significantly more skewed towards the upper bound, whilst non—MWE would distribute more normally across observed scores. The distributions of PMI scores of PLANE's phrases, divided by adjective class, are presented in Figure 4.

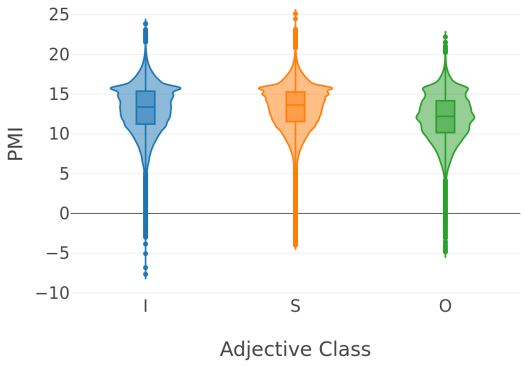

Figure 4: PMI scores by adjective class. Distribution of the PMI scores for each adjective–noun phrase in PLANE.

The median values of all three classes are notably distant from 0 and upper–bound outliers. Phrases containing intersective (I), and subsective (S) adjectives have a strikingly similar distributions, skewed towards higher values. On the contrary, phrases built with intensional (O) adjectives present a slightly lower average PMI score, and appear more evenly distributed. A manual inspection of a subset from phrases with a PMI equal to or higher than 15 didn't identify any idioms or MWE. The same observation holds for the circa 0.1% of phrases with a score equal to, or lower than, 0.

B In–Context Learning

B.1 Zero-Shot Preliminary experiment

As in Section 4, our preliminary Zero–Shot experiment focused on an unmasking problem. We adopted the same prompt templates of Table 2. However, in this case no contextual examples were included within each prompt, so models were not expose to either of the possible labels' verbalisers. We hence built a conversion table V by manually collecting sensible tokens from the set of commonly retrieved ones. Table 7, present the collected conversion table V, mapping potential verbalisers to the `positive` (✓) and `negative` (✗) labels.

✓	✗
good	poor
true	false
positive	negative
great	bad
possible	impossible
plausible	implausible
acceptable	unacceptable
strong	weak

Table 7: Verbalisers adopted for positive (✓) and negative (✗) samples in the Zero–Shot experiment.

Results Results divided by adjective class and model are presented in Table 8. RoBERTa models appear to perform the best, showing also the least amount of variance between the base and large variation of the model. BERT models are the second best performing family. Interestingly, BERT-base seems to outperform its large counterpart. Distillation based models clearly produce the worse results across the all board. Ignoring DistilBERT, results appear to follow the same pattern: performance is the highest on the intersective (I) class, followed by subsective (S) and then intentional (O).

Prompt template analysis Results in Figure 5 provide the mean F1 performance divided by adjective class (coloured lines), forms (x axis), and prompt template (PT, column). Error bars refers to standard deviations, and illustrate the variance produced by collapsing each model's performance.

As it can be clearly appreciated by the Figure, the vast majority of the variance can be attributed to the prompt template (PT). Whilst under the `Internal` PT models where partially able to interpret the given task, the `External` PT made it almost impossible to produce a correct prediction. Lastly, the fact that, regardless of PT and adjective class, ITs associated with negative labels shows a performance close to zero strongly suggest that, without contextual information, most models strongly prefer positive solutions.

Adj. Class	BERT-base	BERT-large	DistilBERT	DistilRoBERTa	RoBERTa-base	RoBERTa-large
I	35.8 ± 25.2	24.1 ± 16.9	7.8 ± 5.4	5.3 ± 3.7	42.5 ± 29.9	42.2 ± 29.8
S	21.9 ± 14.8	13.9 ± 9.2	8.3 ± 5.6	5.0 ± 3.5	23.1 ± 16.2	23.1 ± 16.3
O	5.9 ± 2.2	6.4 ± 1.5	3.3 ± 1.3	1.8 ± 0.3	8.6 ± 4.9	7.9 ± 5.2
Average	21.2 ± 14.1	14.8 ± 9.2	6.5 ± 4.1	4.0 ± 2.5	24.7 ± 17.0	24.4 ± 17.1

Table 8: Zero–Shot learning results. Individual model's performances (mean F1 ± standard deviation from prompt template (PT)), divided by adjective class.

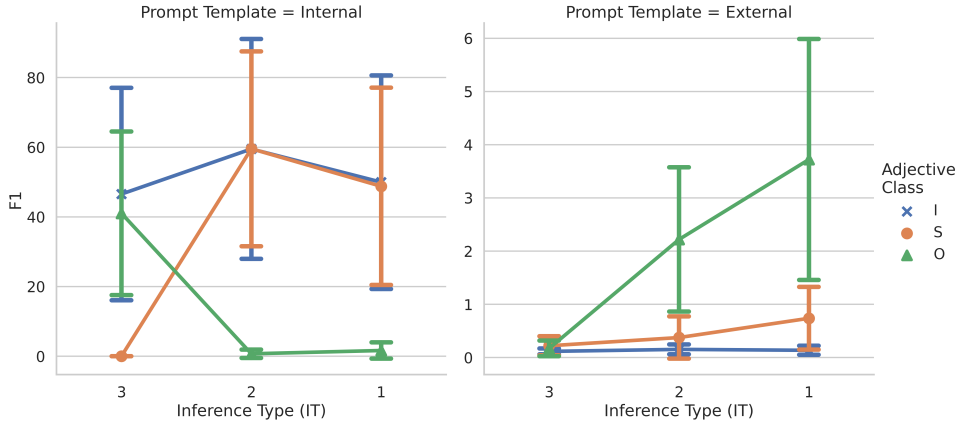

Figure 5: Variance analysis of the Zero–Shot experiment. The graph displays the mean F1 and standard deviation (bars, obtained by collapsing models' performance) obtained on different adjective classes. X-axis refers to the inference type (IT) of the test-items.

B.2 Two-Shot: visualising variance and recency effect

The analysis on the variance generated by prompt templates (PT) and verbalisers is presented in Figure 6. Mean F1 performance is provided, collapsed by models' and premises' permutations [7]. Each column represent an adjective class (I, S and O, respectively), whilst the rows identify the two PT: `Internal` and `External`, respectively (see Table 2).

Figure 7 present the results in further detail, divided by single permutations of a sequence's premises to visualise possible recency effect.

C Transfer Learning

C.1 visualisation of error analysis

The section provides a visual summary of the error analysis in Section 5 via Figure 9.

D In–Distribution Compositional Generalisation

The In–Distribution generalisation experiment presents the same setting as Section 6, with fundamental difference that test set do not contain out–of–distribution items. Following Keysers et al. (2020)'s notation, given a dataset, we identify a set of *atoms*, single words (i.e., adjectives, nouns, hypernyms) and inference types (IT), and a set of *compounds*, which are combination of these three elements. Hence, for this experiment, we generated training and test splits with overlapping atoms, and disjoint compound distributions. Results in term of accuracy against maximum sequence length are presented in Figure 8.

First, cutting the maximum length of the input sequence to 6 tokens produces chance–level performance. As two–third of test items are composed of only 6 words, this suggested that: i) a consistent portion of the input sequences gets split into WordPieces (WP); ii) our splitting algorithm successfully generated sets without biases or c_1–`label` association the model could use to solve the task. As soon as input sequence length reaches 12, the task becomes, as predictable, trivial. WP might still play a minor role – accuracy is still not 1 with sequence length of 8). However, since the atoms, and the adjectives especially, are shared between train and test, the model technically has all the information need to infer the correct label: combining the adjective with inference type (IT)

[7] That is, when testing items with hypothesis of IT 3, we combine results for premise with IT sequences 1,2 2,1.

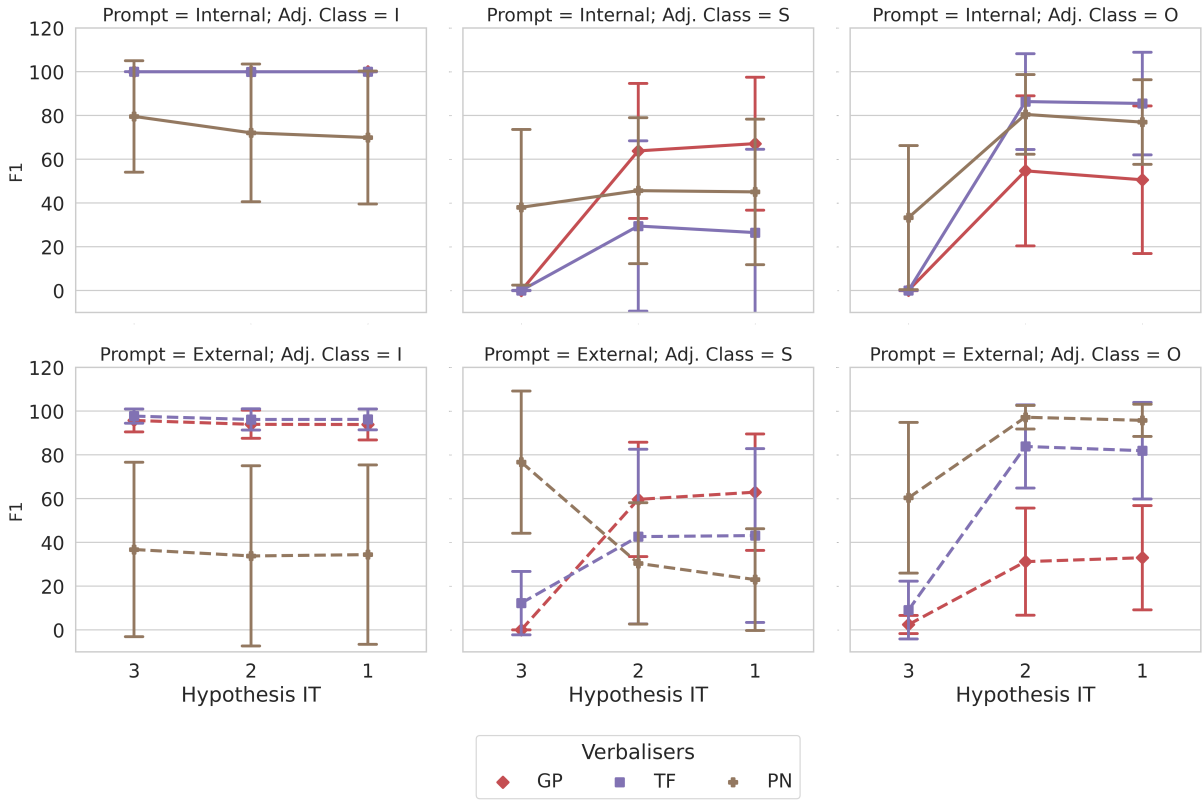

Figure 6: Variance analysis of the Two–Shot experiment. Each graph displays the mean F1 and standard deviation (shown via error bars, generate by collapsing models' performance) obtained by different verbalisers, on a specific combination of prompt template (rows) and adjective class (columns). X-axis refers to the inference type (IT) presented in the hypothesis of the test-items.

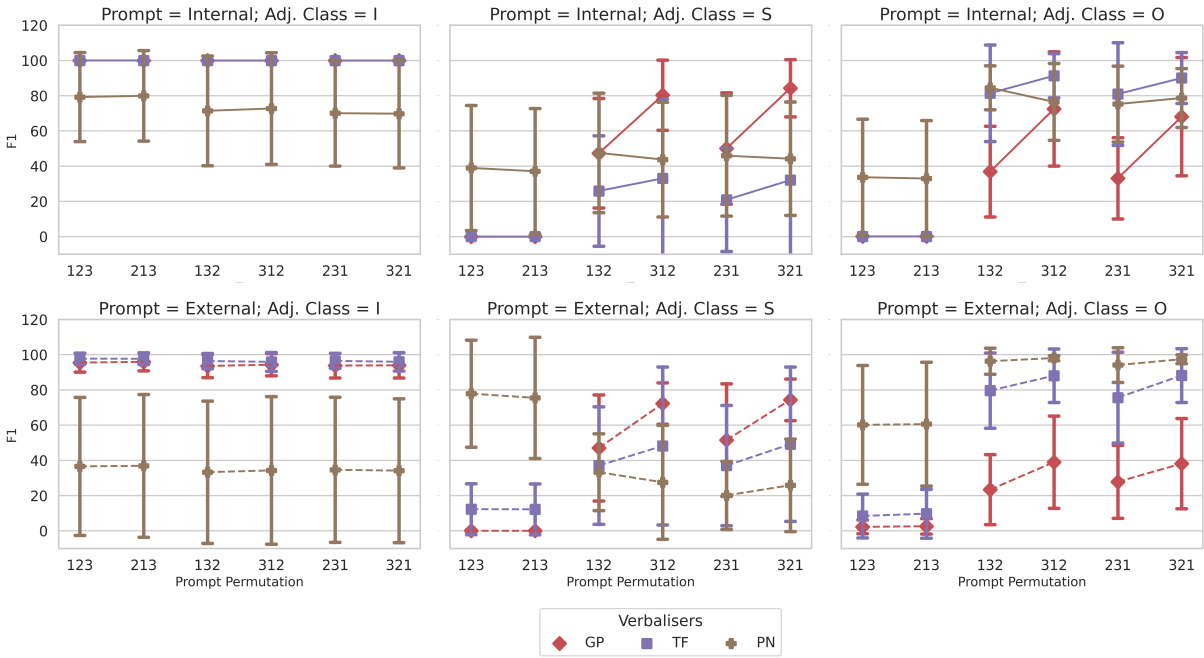

Figure 7: Variance and recency effect analysis of the Two–Shot experiment. Each graph displays the mean F1 and standard deviation (shown via error bars, generate by collapsing models' performance) obtained by different verbalisers, on a specific combination of prompt template (rows) and adjective class (columns). X-axis refers to the inference type (IT) sequence presented in the test-items.

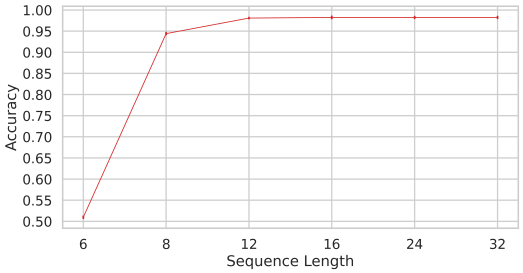

Figure 8: In–Distribution generalisation results. Impact of the input sequence length on accuracy in the compositional generalisation experiment.

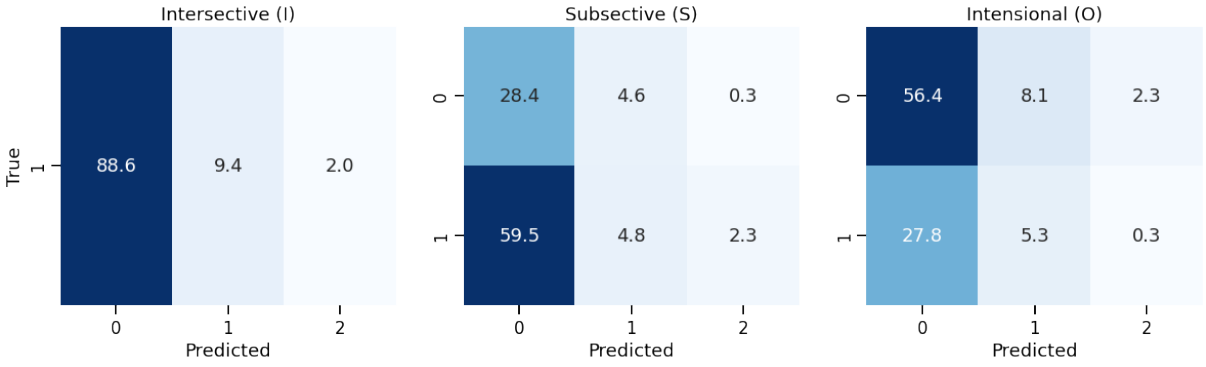

(a) Liu et al. (2019)

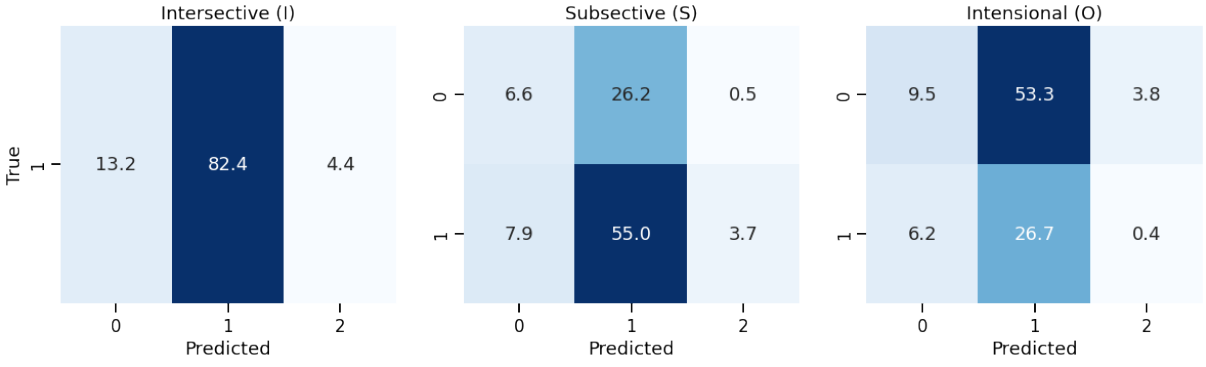

(b) Nie et al. (2020)

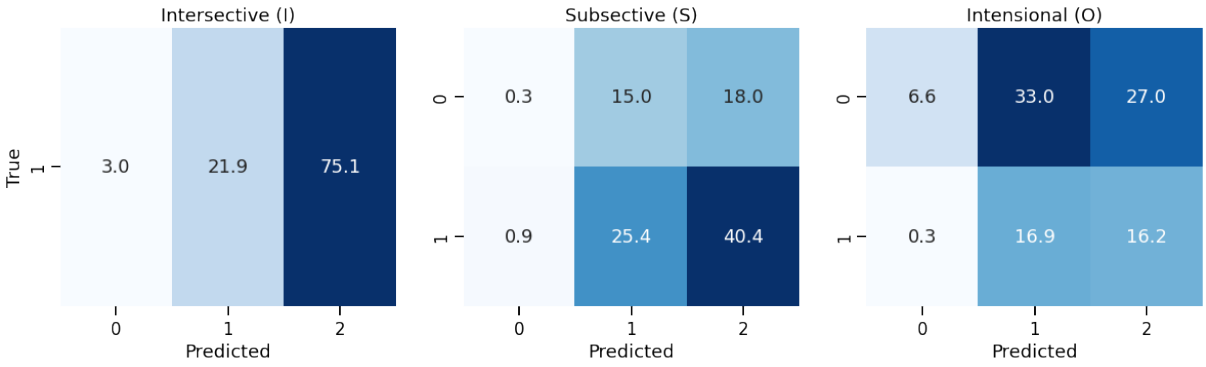

(c) AddOne

Figure 9: Testing NLI models error analysis'. Confusion matrices of the three RoBERTa models tuned on different NLI benchmarks, and tested on PLANE instances from the Two-Shot experiment.

Does BERT Recognize an Agent? Modeling Dowty's Proto-Roles with Contextual Embeddings

Mattia Proietti
CoLing Lab, University of Pisa
m.proietti2@studenti.unipi.it

Gianluca E. Lebani
Ca' Foscari University of Venice
gianluca.lebani@unive.it

Alessandro Lenci
CoLing Lab, University of Pisa
alessandro.lenci@unipi.it

Abstract

Contextual embeddings build multidimensional representations of word tokens based on their context of occurrence. Such models have been shown to achieve a state-of-the-art performance on a wide variety of tasks. Yet, the community struggles in understanding what kind of semantic knowledge these representations encode. We report a series of experiments aimed at investigating to what extent one of such models, BERT, is able to infer the semantic relations that, according to Dowty's Proto-Roles theory, a verbal argument receives by virtue of its role in the event described by the verb.

This hypothesis were put to test by learning a linear mapping from the BERT's verb embeddings to an interpretable space of semantic properties built from the linguistic dataset by White et al. (2016). In a first experiment we tested whether the semantic properties inferred from a typed version of the BERT embeddings would be more linguistically plausible than those produced by relying on static embeddings. We then move to evaluate the semantic properties inferred from the contextual embeddings both against those available in the original dataset, as well as by assessing their ability to model the semantic properties possessed by the agent of the verbs participating in the so-called causative alternation.

1 Introduction

In the last two decades, word embeddings have become one of the most widely used tools for the encoding of lexical meaning in computational models of language. Different flavours of such models have been proposed, all of which have in common the idea of representing lexical elements as multidimensional vectors inferred from their context of occurrence (for a review, see Lenci, 2018).

The last wave of word embeddings followed the transformer-based models breakthrough (Vaswani et al., 2017), that resulted in the development of the so-called **contextual embeddings**. These representations are generated by models like BERT (Devlin et al., 2019) or GPTs (Radford and Narasimhan, 2018; Radford et al., 2019) and derive their name by their ability to keep track of the different contexts in which a word occurs, giving different vector representations for the same word appearing with different surrounding neighbours (for review, see Liu et al., 2020; Ethayarajh, 2019). This has been a major improvement over **static embeddings** obtained from models such as LSA (Landauer and Dumais, 1997), GloVe (Pennington et al., 2014) and Word2Vec (Mikolov et al., 2013), allowing this kind of representation to reach state-of-the-art performance in a great variety of Natural Language Processing (NLP) tasks.

Notwithstanding their wide usage, mainly due to their great empirical successes, the community still struggles to understand what kind of information word embeddings are actually able to encode about language structure, and how they do it. The problem has been even sharpened with contextual embeddings, which are considered to be more entangled representations, usually bigger in dimensions than previous versions, and are obtained from deeper neural models, whose inner working is more complex. Due to this fact, to better understand and explain what kind of structure these models are able to represent is becoming more and more desirable and several research lines started to spring out with this purpose. To the present days the encoding of syntactic knowledge in these model has been more studied than their ability to deal with semantic facets of language (Rogers et al., 2020), but the number of studies in that direction, usually carried

Proceedings of the 29th International Conference on Computational Linguistics, pages 4101–4112
October 12–17, 2022.

out by means of probing tasks developed on top of pre-trained architectures (Vulić et al., 2020), is also growing (Chersoni et al., 2021; Ettinger, 2020).

In the present paper we focus on the modelling of the semantic content of what Dowty (Dowty, 1989, 1991) labelled as **Thematic Proto-Roles**, that are clusters of entailment properties that an arguments derives solely by virtue of its role in the event described by a predicate. Following previous work by Lebani and Lenci (2021) we use a linear transformation mapping between embeddings of verbs represented in BERT's space and a set of interpretable vectors derived from the Universal Decompositional Semantics Dataset on Proto-Roles properties (White et al., 2016), in which human ratings about argument properties have been annotated and collected. However, the present work differs from Lebani and Lenci (2021) under several respects: i.) we deal with contextual embeddings, focusing on those yielded by BERT, ii.) we experiment with representations at the token-level, iii.) we successfully apply sPCA as a de-noising technique, iv.) and we qualitatively address the phenomenon of the causative-inchoative alternation, for which the notion of Semantic (Proto-) Role is crucially relevant.

The goal of this paper is twofold: i.) to test whether the BERT contextual embeddings of a verb encode semantic information concerning the Proto-Role properties held by its arguments, and ii.) to test whether this knowledge can be distilled by means of a linear mapping, thus leading the way to the development of full-scale systems able to extract this knowledge for a wide range of verbs.

The following pages are organized as follows: in Section 2, we quickly review the literature investigating the semantic content of vector representations, before discussing the notion of thematic role and its empirical foundations. We describe and test our method in Sections 4 and 5, respectively, while Section 6 is devoted to a general discussion of the merits and limitations of the use of the BERT's embeddings to model Proto-Role information.

2 The Semantics of Word Embeddings

Notwithstanding the wide usage of word embeddings in NLP and related fields, the literature trying to characterize the semantic properties of these representation is quite scarce. Concerning the efforts in trying to understand whether and to what extent thematic roles information is encoded in contextual embeddings, Tenney et al. (2019b), (building on works by Teichert et al., 2017 and Rudinger et al., 2018) proposed a suite of classification tasks aimed at investigating how these representation encode sentence structure across a range of syntactic, semantic, local, and long-range phenomena. Crucially, these authors report a very small improvement over non-contextual baselines. Thematic role information seems to be recoverable with this strategy, but to an extent which is not that notable.

Ettinger (2020) tested BERT on a suite of diagnostics drawn from human language experiments, among which the most relevant to our scopes is the semantic role sensitivity and event knowledge task, that tests the model ability to discern between good and bad sentence fillers on the basis on the required semantic role. Results showed that the model is not that accurate at matching human predictions, even if some of the information appear to be encoded. Klafka and Ettinger (2020) developed a suite of probing tasks with the aim of assessing what kind of semantic information about the surrounding words is encoded in a contextual embedding. For instance, a task of this suite can implement the question "*What does the embedding of the verb tell us about the animacy of the subject noun?*" as a binary task for a MLP classifier trained and tested on the embeddings of a single word (Klafka and Ettinger, 2020, p. 4802). Relevant for our purposes, these scholars report that much information about subject's animacy can be recovered by inspecting the embedding of the verb.

However, our work is more strongly related, both in goals and methods, to those by Fagarasan et al. (2015), Utsumi (2020), Chersoni et al. (2021) and above all to Lebani and Lenci (2021). All these authors have used a linear transformation to learn a mapping between an embedding space and a space derived from human judgements. Fagarasan et al. (2015) learned a mapping towards the short normalized descriptions (*feature norm*) collected by McRae et al. (2005) in order to learn to predict perceptual features for novel words. Both Utsumi (2020) and Chersoni et al. (2021) applied that strategy to decode word vectors in terms of the brain based semantic features collected in Binder et al. (2016). Finally, Lebani and Lenci (2021) focused on finding fine-grained Proto-Role information by learning the mapping between several static embeddings and an entailment space based on the same ratings we use in this experiment, i.e. the White

et al. (2016) dataset that is introduced in the next section. All in all, these works show that both contextual and static embeddings encode a wide range of (psycho)linguistic relevant information that can be inferred by means of a simple linear mapping.

3 Thematic Proto-Roles

Theories about the role played by the arguments of the verb at the syntax-semantics interface have come in many flavours. One of the most debated and controversial matter in theoretical linguistics concerns the definition of the notion of **semantic role** and the development of a reliable method for the identification of such categories.

In contrast with the traditional view of semantic roles as discrete primitive semantic categories, David Dowty proposed to reduce the total number of roles to two prototypical notions, which he called **Proto-Agent** and **Proto-Patient** (Dowty, 1989, 1991). In Dowty's view, roles are defined by a set of entailments determined by the meaning of the verb. Some properties contribute to characterize an argument as *Proto-Agent*, while others as *Proto-Patient*, but they can be present in different degrees and in mixed configurations. These configurations correspond to classical intermediate roles such as experiencer, theme, and so on.

This approach has some important advantages over classical views of semantic roles, which have always been in tension between the choice of the right number of roles and the right mapping between grammatical and semantic role. Dowty's theory does not discard completely the possibility of identifying a different role for each different peculiar argument, placing a distinction between *specific roles* and *linguistic roles*. While the former are specific for each verb (e.g., to build has two main arguments: the *builder* and the *buildee*), the latter are generalizations aimed at capturing common traits about different specific roles. For example *builder, killer, worker, seller* are all instances of *Proto-Agent*, but all at a different degree.

Furthermore, concerning the selection of the argument, Dowty sets a straightforward rule in his **Argument Selection Principle** stating that: "*In predicates with grammatical subject and object, the argument for which the predicate entails the greatest number of* Proto-Agent *properties will be lexicalized as subject of the predicate; the argument having the greatest number of* Proto-Patient *entailments will be lexicalized as the direct object.*" (Dowty, 1991, p. 576). This claim received empirical validation on a cognitive perspective by Kako (2006), among others. This scholar proved, through a series of experiments, not only that the hypothesis has psychological validity, but also that "*speakers can make inferences about these properties from grammatical roles alone [...]*" (Kako, 2006, p. 34). Inspired by these findings Reisinger et al. (2015) built a crowd-sourcing experiment to test the theory against a large amount of data and substantially confirmed the results by Kako (2006). The latter approach has been the precursor of the dataset by White et al. (2016) that we adopt here to build a semantic space based on human judgments.

3.1 Human Judgements about Proto-Roles

As will be explained in more detail in Section 4, in order to infer whether the BERT contextual embeddings are able to encode some information about semantic roles, we studied the output of a mapping from the contextual embeddings of a group of selected verbs and the ratings produced by a group of speakers. In our experiment we rely on the judgments collected by White et al. (2016). This dataset was built by asking a group of native speakers to read a series of sentences with a highlighted argument and to answer, on a five points Likert scale, to a group of Dowty-inspired questions on the target argument. For example, to know how plausible is for an argument to have a property like `awareness`, the subjects were asked: "*ARG was/were aware of being involved in PRED?*".

The paradigm used by White et al. (2016) was developed from that described by Reisinger et al. (2015). In the latter work the authors annotated sentences from PropBank (Palmer et al., 2005) while White et al. (2016) used the English Web Treebank (Silveira et al., 2014), which is annotated following the Universal Dependencies guidelines (de Marneffe et al., 2021) and covers a greater variety of genres. Furthermore, White et al. (2016) revised the inventory of questions and the method to present them and used redundant annotations. The semantic decomposition principle behind the whole paradigm is well suited to Dowty's theory of Proto-Roles, and vice-versa, due to their common target of reducing semantics categories to smaller dimensions of meaning. This reduction allows not only linguists to better describe the categories, but also naive speakers to understand the questions to characterize the semantic roles.

	⟨nsubj, awareness⟩	⟨nsubj, change of state⟩	⟨nsubj, volition⟩	⟨dobj, awareness⟩	⟨dobj, change of state⟩	⟨dobj, volition⟩
to affect	0	0.625	0	0.688	0.75	0.187
to amaze	0.25	0.25	0.25	1	0.708	0.792
to bring	0.922	0.422	0.828	0.562	0.472	0.319
to fill	0.875	0.25	0.875	0.5	0.875	0.562
to give	0.899	0.352	0.887	0.062	0.312	0.081
to ignore	1	0.875	1	0.75	0.5	0.125
to include	0.458	0.51	0.433	0.451	0.461	0.446
to kill	0.925	0.65	0.875	0.575	0.937	0.042
to put	0.833	0.492	0.84	0.275	0.75	0.11
to tell	0.99	0.357	0.959	0.968	0.561	0.714

Table 1: Portion of the entailment-based vector space (adapted from Lebani and Lenci (2021)).

4 General Methodology

Our ultimate goal is to probe the kind of distributional knowledge encoded in contextual embeddings in order to assess whether BERT (and arguable other models of the same family) is able to encode Proto-Role semantic information. As such, we opted for a methodology that has been tested and proven in our reference literature (Fagarasan et al., 2015; Utsumi, 2020; Chersoni et al., 2021; Lebani and Lenci, 2021). Similarly to what has been done by Lebani and Lenci (2021), indeed, we created a linear mapping between a semantic space composed of BERT embeddings and a vector space derived from the ratings collected in the White et al. (2016)'s Proto-Roles dataset.

Model We tested BERT (Devlin et al., 2019) in its `bert-large-cased` version as released in the Hugging Face python library (Wolf et al., 2019). This deep encoder architecture has 24 layers, 1,024 hidden units per layer, 16 attention heads and a total of 336M of parameters. It is pre-trained with masked language modeling and next sentence prediction tasks. As we want to know the semantic properties that BERT encodes in its native representations, we did not fine-tune the model.

Corpus The sentences annotated by White et al. (2016) were extracted from the English Web Treebank (Silveira et al., 2014) corpus, which is available in the Universal Dependencies repository.[1] From the training set of this corpus we extracted a list of 2226 pre-tokenized sentences that were later processed with BERT. From these sentences we extracted only the verb embeddings either at type or token level. For the type-level experiment, verb vectors were averaged across different contexts (Bommasani et al., 2020)[2].

Ratings-based semantic spaces We built different rating-based semantic spaces for the type-level and for the token-level analyses. For the type-level analysis we followed Lebani and Lenci (2021) and built a unique semantic space for both arguments, as shown in Table 1. For the evaluation of the token-level embeddings, on the other side, we built different spaces for the `nsubj` and for `dobj` syntactic roles, choosing to ignore the passive subjects in order to remove excessive sparsity. The latter procedure left us with 1972 token instances for the `nsubj` space and 797 token instances for the `dobj` space. The dimensions of these spaces correspond to the 14 properties tested by the authors, as ranked by annotators for each token. We indexed each token with the id of its sentence, in order to retrieve it and compare different occurrences of the same verb type.

Learning Algorithm As a mapping strategy, in the wake of previous works (Chersoni et al., 2021; Fagarasan et al., 2015; Lebani and Lenci, 2021)

[1] https://universaldependencies.org/

[2] We did not need to average between word pieces, as also suggested in Bommasani et al. (2020), since we used sentences that were pre-tokenized at the word level.

we used the Partial Least Squares (PLS) regression implementation in the Scikit-learn Python library (Pedregosa et al., 2011), with the number of components set to 10 and within a ten-fold cross-validation. We evaluated the predicted vectors by calculating its Spearman's rank correlation with the original ones, both row-wise and column-wise.

To check the quality of our model, we generated a matrix for each experiment with values randomly sampled from the interval $[0, 1]$, shaping it like the corresponding BERT space dimensions. We treated the performance of the mapping learned from these randomly generated spaces as a baseline.

5 Experiments

5.1 Experiment 1: Type-level

For the *type-level* analysis, we reproduced on the BERT vectors the experimental settings as in Lebani and Lenci (2021), mainly in order to obtain a set of comparable results. The application of the same filtering strategy resulted in a vector space composed of 155 rows, one for each verb lemma, and 41 columns, corresponding to features made of <*grammatical_function,property*> pairs, which were first aggregated by averaging between different annotators judgments and instances of the same lemma and than scaled to fit the range [0,1].

After constructing the BERT type embeddings as described in Section 4, we moved on to learning the mapping and, as we wanted to have a grasp of the differences in performance across the whole BERT model, we tested each of the layers averaging between them in groups of four (e.g., layers 1-4, layers 5-8, etc). Even if we weren't able to identify significant differences across groups, we found a peak around the group of layers 13-16. We believe that these results are consistent with the findings in Tenney et al. (2019a), where it is shown that syntactic and grammatical information (e.g., word order, POS) is better encoded at lower layers, while semantics features (e.g., semantic roles, coreference) are better represented at higher layers, although the latter seems to be more equally distributed than the former across the whole model. Correlation results for our best performing group of layers – comprising layers 13-16 – are reported below in Table 2, while scores for other groups can be found in Table 3 in Appendix A. We obtained average values directly comparable to the best performing model found in Lebani and Lenci (2021), which is a Skip-Gram model with negative sampling and syntactic typing (SGNS.syn). All the groups of layers we tested performed equally or slightly better in absolute terms than the SGNS.syn used in Lebani and Lenci (2021). In particular, our best performing group of layers reaches average correlation values of $\rho = 0.74$ row-wise (i.e., correlations by verb) and $\rho = 0.39$ column-wise (i.e., correlations by property), against the respective values of SGNS.syn of $\rho = 0.71$ and $\rho = 0.31$. However, we encountered the same problem of Lebani and Lenci (2021) when evaluating the mapping row-wise, that is an unexpected high baseline, which in our case set itself at $\rho = 0.64$[3]. We tried to overcome this limit in the second experiment made at the *token-level*. Overall, the interpretation suggested by those results seems to be that, when reduced to static embeddings, BERT contextual vectors perform only slightly better, if at all, than those of a classic non-contextual Distributional Semantic Model with proper hyper-parameters settings, when it comes to retrieve fine-grained information about thematic Proto-Roles.

Model	Row-wise	Column-wise
BERT$_{13_16}$	0.74	0.39
baseline_BERT	0.64	-0.07
SGNS.syn	0.71	0.31
baseline_SGNS	0.62	0.035

Table 2: Average Spearman's scores for BERT, group of layers 13-16, and SGNS.syn (the best performing static model in Lebani and Lenci, 2021), with relative baselines.

Regarding correlations obtained by property at the type-level, which are shown in detail in Figure 1, we found that the `dobj` argument seems to be the one easier to model but, interestingly enough, it shows higher values for *Proto-Agent* properties, reaching the highest in `awareness` with a $\rho = 0.62$. There are three properties specifically related to the *Proto-Patient* that are scored relatively high for the `dobj` argument: `change of possession` ($\rho = 0.59$), `change of state` ($\rho = 0.52$) and `was used` ($\rho = 0.49$). As for the `nsubj` argument, the best modeled properties seem to be those that characterize a *Proto-Agent* role, that is to say `sentient` ($\rho = 0.53$), `volition` ($\rho = 0.49$), `awareness` ($\rho = 0.47$)

[3]The baseline scores attested in our trials belonged to the $[0.62, 0.66]$ interval, coherently with the baseline score $\rho = 0.62$ reported by Lebani and Lenci (2021).

and was for benefit (ρ = 0.43). Overall, these results are consistent with those found in Lebani and Lenci (2021) but partially deviate from those we obtained at the token-level, as it will be shown in the next sections.

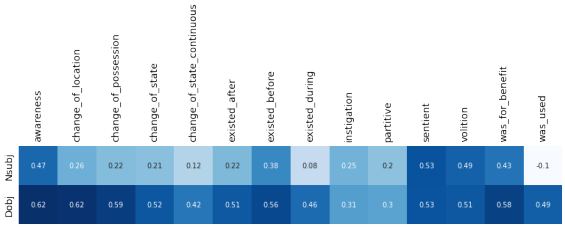

Figure 1: Detailed properties correlation values at the type-level for nsubj and dobj.

5.2 Experiment 2: Token-level

As already mentioned, we created two different mappings at the token-level, one for nsubj argument and one for the dobj argument. Differently from the type-level experiment, in which every data point was an abstract representation of a decontextualised verb lemma described by the 41 features made of the union of grammatical function and Proto-Role property, here we deal with words tokens in context. Since not every verb occurrence received annotations both for nsubj and dobj in White et al. (2016), we created the two sub-spaces, in which each instance is described by the set of fourteen properties elicited with the questions of the SPR2 protocol found in White et al. (2016). In this case we ran the experiment on single layers chosen among those of the best performing group in the previous experiment, which was that of layers 13-16, and we report here the results for layer 16. Simply reproducing the mapping on those spaces with the same settings as the type-level gave us results really similar to the first experiment.

Concerning the high baseline problem, we took into account the hypothesis put forward by Lebani and Lenci (2021). They considered a possible justification of these results the fact that "*Subjects tend to have proto-agent properties, while object tend to have proto-patient properties. From this association*[...] *follows the fact that the vectors of our target entailment-based space are, to a certain extent, bound to share a similar structure in which some dimensions tend to be consistently scored higher than others.*" (Lebani and Lenci, 2021). This proposal is confirmed by measuring the cosine similarity among the vectors of the entailment-based spaces. In fact, a look at the average cosine similarity in the semantic spaces gave us values of cos = 0.85 and cos = 0.77, respectively for the nsubj and dobj, showing that indeed there is a high similarity score among the vectors, which can introduce noise and alter the learning process of our model, thereby allowing the baseline to reach high correlations. Thus, we tried to use a dimensionality reduction technique such as Sparse Principal Components Analysis (sPCA) in its Scikit-learn implementation, which is based on Mairal et al. (2009). The goal was to introduce sparsity in our data and reduce the noise, without reducing the number of dimensions and losing interpretability. This technique is mostly used for de-noising purposes in the field of computer vision, but rarely employed in NLP (Drikvandi and Lawal, 2020). Differently from classic PCA, sPCA does not yield orthogonal dimensions in the space where it is applied, but seems to succeed in reducing similarity among the instances of our ratings-based spaces. As a matter of fact, the average cosine value, for both nsubj and dobj space, resulted in cos $\approx$ 0 after the application of this technique. Furthermore, the sparsity of the loadings generated with this method allowed us to have a better grasp of which principal component represented which variable.

As shown in Figure 2, we obtained average correlations of ρ = 0.50 for nsubj and ρ = 0.40 for dobj at the row-level, with the baseline keeping itself at $\rho \approx 0$ in both cases[4] . It should be noted that the reported manipulations with sPCA affect only the row-wise analysis, which was indeed the only one suffering from the high baseline problem. The average values obtained column-wise remain the same, and the same has to be said for the fine grained analysis of single properties. In fact, the correlation values obtained by the new components yielded by sPCA overlap perfectly with the original variables. At the column level we got ρ = 0.43 for the nsubj space and ρ = 0.38 for the dobj space. Despite the fact that these correlations do not reach outstanding values, they are significantly higher than the baseline both row-wise and column-wise, and the ones obtained with BERT and SGNS type

[4]We also experimented with standard PCA. We searched for a number of components capable of accounting for the 85% of the variance, thus obtaining a different number of components for our spaces: 6 for nsubj and 8 for dobj. The correlations with this strategy are lower than those obtained by using sPCA: ρ = 0.42 (by row) and ρ = 0.33 (by columns) for the nsubj; of ρ = 0.38 (by row) and ρ = 0.34 (by column) for the dobj.

vectors.

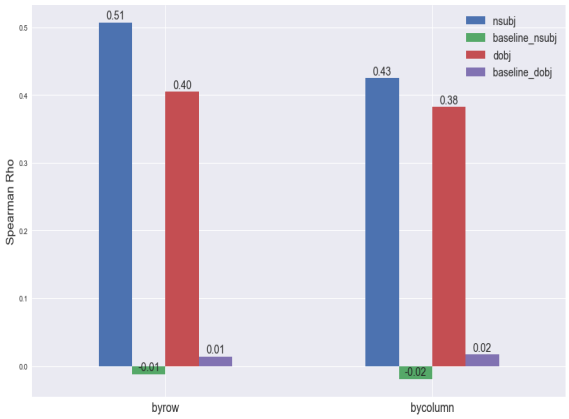

Figure 2: Average correlations at the token-level.

5.3 Experiment 3: Modeling the causative-inchoative alternation

As a third experiment, we decided to make a more qualitative analysis focusing on the so called causative alternation. This linguistic phenomenon is directly tied with theories of semantic/thematic roles and Dowty's theory is no exception. Our hypothesis here is that verbs participating to the causative alternation, occurring both in transitive and intransitive frames, should entail a set of properties more skewed toward the *Proto-Agent* role when appearing in transitive contexts and should incline toward those entailments typical of the *Proto-Patient* in their intransitive occurrences. For example, the verb *to break* can appear in transitive sentences like *John broke the window* or in intransitive ones as *The window broke*, entailing different properties about the respective subjects. In fact, the two subjects are supposed to be realizations of different underlying Proto-Roles, a *Proto-Agent* in the former case, a *Proto-Patient* in the latter.

Our aim was to test BERT embeddings to know whether they are able to encode some information about that alternation. We focused again on the token level, using the `nsubj` space previously created to train a PLS regression model on transitive verbs. We selected 100 sentences containing 50 alternating verb types, thus having 50 pairs of causative alternation examples. Target verbs for this experiment have been selected following the Levin (1993)'s classification as coded in VerbNet (Schuler, 2006). Sentences containing these target verbs have been extracted manually from a variety of sources, comprising VerbNet frames examples, FrameNet (Baker et al., 1998) examples, and enTenTen corpus through Sketch Engine (Kilgarriff et al., 2014). We found that causative alternation is indeed well modeled in the majority of cases (35 out of 50 pairs of sentences, 70%), as can be seen from Figure 3, which shows a visual representation of a portion of our predicted vector space.

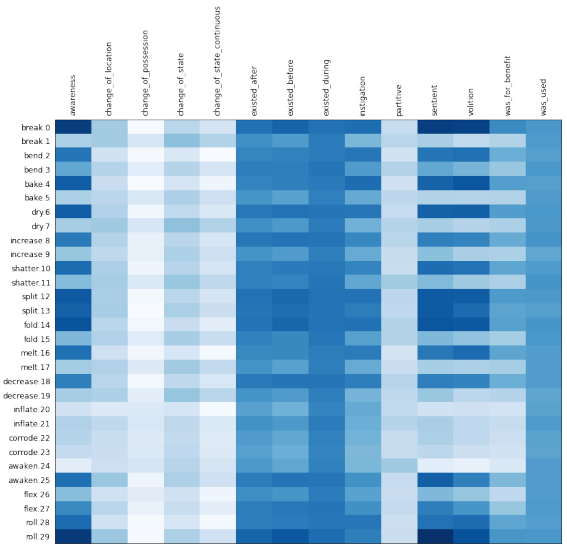

Figure 3: Visualization of the first 30 alternating verbs in our predicted space. Even ids are for transitive frames, odd for intransitive ones.

The alternation is clearly visible in the difference of intensity in those slots of the heatmap corresponding to the *Proto-Agent* properties, mostly in `awareness`, `sentient`, `volition`, `instigation` and, to a lesser extent, to those corresponding to the *Proto-Patient*, mostly in `change of state`, `change of state continuous`, `was used`. It should be noted that our model fails to catch the alternation in some pairs of verbs (*awaken, flex* and *roll* in the portion showed in Figure 3)[5]. However, these failures, which represent the 30% of the predicted outcomes (15/35), pave the way to further considerations that we discuss in the next section.

6 General Discussion

6.1 Ability to recover fine-grained Proto-Role information in BERT's embeddings

All the three reported experiments show that it is indeed possible to recover Proto-Role information about the arguments from verb embeddings, as demonstrated by the average correlation values

[5]Due to an error, ids for the verb *inflate* are switched. Thus it seems that the intransitive has higher scores in *Proto-Agent* properties than the transitive, which is the contrary.

obtained both row-wise and column-wise, which are significantly higher than those of the baseline, mostly in the token-level experiment in which a preliminary sPCA transformation has been applied. An even more fine-grained analysis can be conducted taking into account single properties. Although not directly comparable, the results at the token-level partially contradict the trend obtained at the type-level, such that in the former the `nsubj` argument shows higher correlation values than the `dobj`, while for the latter the contrary happens. However, what they have in common is the fact that in general *Proto-Agent* properties seem to be better modeled in both experiments and for both arguments. As a matter of fact, at the token-level this happen not only, for the `nsubj`, as expected, but also for the `dobj` even if, in this second experiment, we obtained higher correlations for the `nsubj` argument, as can be seen in Figure 4. In particular, three properties seem to be pretty well modeled: `awareness`, `sentient` and `volition`, which are the core entailments of the *Proto-Agent* role, and are strongly related to the animacy of typical subject arguments.

On the contrary, our model struggles to cope with *Proto-Patient* properties at the token-level in both spaces. Whether this evidence means that *Proto-Agent* properties are better represented in BERT or in just the rating-based space it is not easy to say. But, from a theoretical point of view it should be considered that the individuation of good examples of properties for the *Proto-Patient* has been an issue ever since the statement of Dowty's theory. In fact, Dowty himself claimed that "*Proto-Patient properties are harder to isolate entirely*" (Dowty, 1991, p. 576) than those of the *Proto-Agent*.

Moreover, both Reisinger et al. (2015) and Kako (2006) found out in their experiments that "*Proto-Agent properties have a greater effect than Proto-Patient properties*"(Reisinger et al., 2015, p. 481). All these cues might suggest that the individuation and the modeling of *Proto-Patient* properties might be a more difficult matter than *Proto-Agent* ones and that the latter have a more solid stand from several point of views: theoretical analysis, cognitive and corpus-level testing, and probably even in the knowledge encoding operated by BERT. Also, it should be taken into consideration that some further developments of Dowty's theory dispensed with *Proto-Patient* properties at all, building only on those of the *Proto-Agent* and characterizing its opposite role in negative terms (see, for example, the theory elaborated by Grimm, 2011).

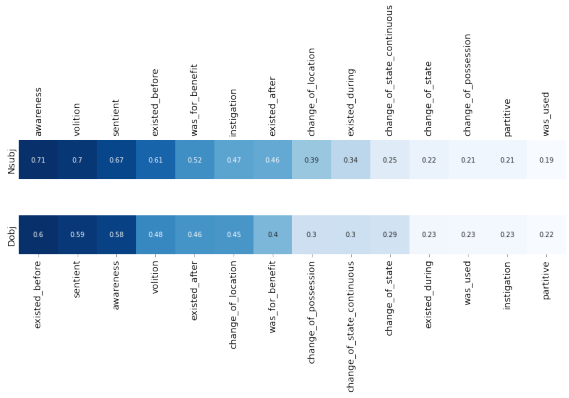

Figure 4: Detailed single properties correlations for `nsubj` and `dobj`.

6.2 Modeling the causative-inchoative alternation

As it has been shown, we have been able through our strategy to model the causative-inchoative alternation in terms of Proto-Role properties prediction. In fact in 70% of the pairs we predicted, the transitive version of the verb scored higher values in *Proto-Agent* properties. In particular, those cases representing prototypical instances of such phenomenon are almost perfectly predicted. Consider as an example, the pair of sentences regarding the verb *to break* (*break.0* and *break.1* in Figure 3). The corresponding sentences are taken directly from VerbNet example frames and are: *Tony broke the window* and *The window broke*, respectively for the transitive and the intransitive frame. The properties seem to be well predicted not only for those concerning proto-agency, but also for those entailments of the *Proto-Patient*. That is to say, while the first verb, *break.0*, which is transitive, shows a greater intensity (i.e., higher predicted values) than *break.1* in the slots corresponding to `awareness`, `instigation`, `volition`, `sentient`, `existed after`, `existed before` and `was for benefit`, the reverse is true if we consider *Proto-Patient* entailments. In fact, the intransitive *break.1* has greater values in `change of possession`, `change of state`, `change of state continuous`, `partitive`. This is a recurrent pattern among all the predicted space. Moreover, Figure 5 shows how on average *Proto-Agent* properties are scored higher in the predicted

subspace formed by only the transitive use of the verbs. Vice-versa almost all the *Proto-Patient* properties have higher mean scores among the intransitive use, even if with a much smaller difference. This is consistent with the assumption that subjects of inaccusative verbs are less agentive than those of their transitive counterpart. However, there are a few notable exceptions to this trend. Three properties in particular seem to contradict the assumptions of the theory: `change of location`, `partitive` and `was used`. The first is assumed to be typical of *Proto-Agent* and, instead, shows a higher average score for the transitive use. The second and the third, supposed to be *Proto-Patient* properties, reach the same values in both sub-spaces. Among the 15/50 pairs (30%) which our model failed to predict, there are 6 instances in which the subject of the intransitive verb is more animate than that of the transitive verb. We regard this fact as a possible influence in the prediction, due to the fact that animacy and agency, and consequently their characterizing properties, are two strictly related concepts and they might even overlap in some circumstances, for example in the determination of subjecthood. Given the contextual nature of the BERT embeddings, it is no surprise that verb representations are adjusted in relation to the other elements of the sentence, incorporating at each occurrence particular information about surrounding words. In particular, the fact that animacy information about the subject is projected into the verb embedding and is recoverable from it has been shown by Klafka and Ettinger (2020).

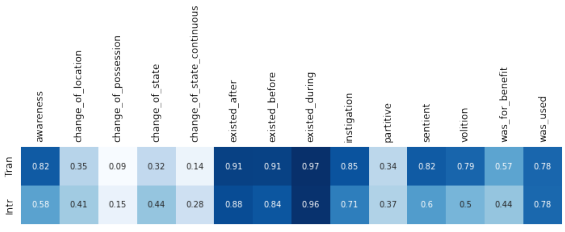

Figure 5: Average predicted properties for transitive and intransitive use.

7 Conclusions

Although our strategy has been proven to be good at modeling the Proto-Roles phenomenon in BERT embeddings to a certain extent, some intrinsic limitations of our work have to be taken into account. Firstly, we used a linear regression (PLS) model as a strategy to build the mapping, but, due to the complexity of the type of information enquired and that of the BERT space, more complex, non-linear transformations, like a Multi Layer Perceptron, might be a better choice for the task. Secondly, the data we used are the best at our disposal, but they are not necessarily the best possible in absolute and might be further improved, by both revising the questions and the set of properties. Thirdly, we obtained mixed results between the token and the type levels concerning which is the best modeled Proto-Role.

Notwithstanding these limitations, we have shown that fine-grained information about Proto-Roles properties of the arguments is recoverable inspecting the embeddings of the verbs yielded by BERT. Also, our results suggest that there might be a discrepancy between the properties of the two Proto-Roles and that *Proto-Agent* properties are better modeled and predicted. We have also been able to show how different Proto-Roles entailments can be predicted in verbs participating to the causative-inchoative alternation. Additionally, we successfully employed sPCA to reduce the noise in our data, which might be a promising cue about future usages of this technique in the field of NLP.

It is worth emphasizing that the main goal of this research is to test BERT's ability to capture some crucial aspects of the verbal argument structure. Even if there can be practical applications of our method (e.g., it can be used as a starting point for Semantic Role labeling or, crucially, Semantic Proto-role labeling; Reisinger et al. 2015; Teichert et al. 2017), our main interest is more theoretical and methodological. Many of the probing tasks that are used today, indeed, do not focus on the prototypical nature of semantic roles, which is precisely a fundamental pillar and the major innovation of Dowty's theory and of the present work. Moreover, our analysis of the causative/inchoative alternation is just a first example of a series of tasks that we plan to develop to characterize the knowledge acquired by these models to explore key aspects of the syntax-semantic interface and of verb argument realization. Finally, we will also extend our approach to other contextual embeddings models, like GPT (Radford and Narasimhan, 2018; Radford et al., 2019) and XLNet (Yang et al., 2019).

References

Collin F. Baker, Charles J. Fillmore, and John B. Lowe. 1998. The Berkeley FrameNet project. In *36th Annual Meeting of the Association for Computational Linguistics and 17th International Conference on Computational Linguistics, Volume 1*, pages 86–90, Montreal, Quebec, Canada. Association for Computational Linguistics.

Jeffrey R. Binder, Lisa L. Conant, Colin J. Humphries, Leonardo Fernandino, Stephen B. Simons, Mario Aguilar, and Rutvik H. Desai. 2016. Toward a brain-based componential semantic representation. *Cognitive Neuropsychology*, 33:130 – 174.

Rishi Bommasani, Kelly Davis, and Claire Cardie. 2020. Interpreting Pretrained Contextualized Representations via Reductions to Static Embeddings. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 4758–4781, Online. Association for Computational Linguistics.

Emmanuele Chersoni, Enrico Santus, Chu-Ren Huang, and Alessandro Lenci. 2021. Decoding word embeddings with brain-based semantic features. *Computational Linguistics*, 47(3):663–698.

Marie-Catherine de Marneffe, Christopher D. Manning, Joakim Nivre, and Daniel Zeman. 2021. Universal Dependencies. *Computational Linguistics*, 47(2):255–308.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, Minneapolis, Minnesota. Association for Computational Linguistics.

David R. Dowty. 1989. On the semantic content of the notion of 'thematic role'. In Gennaro Chierchia, Barbara H. Partee, and Raymond Turner, editors, *Properties, Types and Meaning: Volume II: Semantic Issues*, pages 69–129. Springer Netherlands, Dordrecht.

David R. Dowty. 1991. Thematic proto-roles and argument selection. *Language*, 67:547 – 619.

Reza Drikvandi and Olamide O. Lawal. 2020. Sparse principal component analysis for natural language processing. *Annals of Data Science*, pages 1–17.

Kawin Ethayarajh. 2019. How contextual are contextualized word representations? Comparing the geometry of BERT, ELMo, and GPT-2 embeddings. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 55–65, Hong Kong, China. Association for Computational Linguistics.

Allyson Ettinger. 2020. What BERT is not: Lessons from a new suite of psycholinguistic diagnostics for language models. *Transactions of the Association for Computational Linguistics*, 8:34–48.

Luana Fagarasan, Eva Maria Vecchi, and Stephen Clark. 2015. From distributional semantics to feature norms: grounding semantic models in human perceptual data. In *Proceedings of the 11th International Conference on Computational Semantics*, pages 52–57, London, UK. Association for Computational Linguistics.

Scott Grimm. 2011. Semantics of case. *Morphology*, 21:515–544.

Edward Kako. 2006. Thematic role properties of subjects and objects. *Cognition*, 101:1–42.

Adam Kilgarriff, Vít Baisa, Jan Bušta, Miloš Jakubíček, Vojtěch Kovář, Jan Michelfeit, Pavel Rychlý, and Vít Suchomel. 2014. The Sketch Engine: ten years on. *Lexicography*, pages 7–36.

Josef Klafka and Allyson Ettinger. 2020. Spying on your neighbors: Fine-grained probing of contextual embeddings for information about surrounding words. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 4801–4811, Online. Association for Computational Linguistics.

Thomas K. Landauer and Susan T. Dumais. 1997. A Solution to Plato's Problem: The Latent Semantic Analysis Theory of Acquisition, Induction, and Representation of Knowledge. *Psychological Review*, 104(2):211–240.

Gianluca E. Lebani and Alessandro Lenci. 2021. Investigating dowty's proto-roles with embeddings. *Lingue e Linguaggio*, 20:165–197.

Alessandro Lenci. 2018. Distributional Models of Word Meaning. *Annual Review of Linguistics*, 4(1):151–171.

Beth Levin. 1993. *English Verb Classes and Alternations A Preliminary Investigation*. University of Chicago Press, Chicago and London.

Qi Liu, Matt J. Kusner, and Phil Blunsom. 2020. A survey on contextual embeddings. *ArXiv*, abs/2003.07278.

Julien Mairal, Francis Bach, Jean Ponce, and Guillermo Sapiro. 2009. Online dictionary learning for sparse coding. In *Proceedings of the 26th Annual International Conference on Machine Learning*, ICML '09, page 689–696, New York, NY, USA. Association for Computing Machinery.

Ken McRae, George S. Cree, Mark S. Seidenberg, and Chris McNorgan. 2005. Semantic feature production norms for a large set of living and nonliving things. *Behavior Research Methods*, 37:547–559.

Tomás Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013. Efficient estimation of word representations in vector space. In *1st International Conference on Learning Representations, ICLR 2013, Scottsdale, Arizona, USA, May 2-4, 2013, Workshop Track Proceedings*.

Martha Palmer, Daniel Gildea, and Paul Kingsbury. 2005. The Proposition Bank: An annotated corpus of semantic roles. *Computational Linguistics*, 31(1):71–106.

Fabian Pedregosa, Gaël Varoquaux, Alexandre Gramfort, Vincent Michel, Bertrand Thirion, Olivier Grisel, Mathieu Blondel, Peter Prettenhofer, Ron Weiss, Vincent Dubourg, et al. 2011. Scikit-learn: Machine learning in python. *Journal of Machine Learning Research*, 12:2825–2830.

Jeffrey Pennington, Richard Socher, and Christopher Manning. 2014. GloVe: Global vectors for word representation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1532–1543, Doha, Qatar. Association for Computational Linguistics.

Alec Radford and Karthik Narasimhan. 2018. Improving language understanding by generative pre-training. In *arxiv*.

Alec Radford, Jeff Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. 2019. Language Models are Unsupervised Multitask Learners. *OpenAI blog*, 1(8):9.

Drew Reisinger, Rachel Rudinger, Francis Ferraro, Craig Harman, Kyle Rawlins, and Benjamin Van Durme. 2015. Semantic proto-roles. *Transactions of the Association for Computational Linguistics*, 3:475–488.

Anna Rogers, Olga Kovaleva, and Anna Rumshisky. 2020. A primer in BERTology: What we know about how BERT works. *Transactions of the Association for Computational Linguistics*, 8:842–866.

Rachel Rudinger, Adam Teichert, Ryan Culkin, Sheng Zhang, and Benjamin Van Durme. 2018. Neural-Davidsonian semantic proto-role labeling. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 944–955, Brussels, Belgium. Association for Computational Linguistics.

Karin Kipper Schuler. 2006. *VerbNet: A Broad-Coverage, Comprehensive Verb Lexicon*. Ph.D. thesis, University of Pennsylvania.

Natalia Silveira, Timothy Dozat, Marie-Catherine de Marneffe, Samuel Bowman, Miriam Connor, John Bauer, and Chris Manning. 2014. A gold standard dependency corpus for English. In *Proceedings of the Ninth International Conference on Language Resources and Evaluation (LREC'14)*, pages 2897–2904, Reykjavik, Iceland. European Language Resources Association (ELRA).

Adam Teichert, Adam Poliak, Benjamin Van Durme, and Matthew Gormley. 2017. Semantic Proto-Role Labeling. In *AAAI Conference on Artificial Intelligence (AAAI)*.

Ian Tenney, Dipanjan Das, and Ellie Pavlick. 2019a. BERT rediscovers the classical NLP pipeline. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 4593–4601, Florence, Italy. Association for Computational Linguistics.

Ian Tenney, Patrick Xia, Berlin Chen, Alex Wang, Adam Poliak, R. Thomas McCoy, Najoung Kim, Benjamin Van Durme, Samuel R. Bowman, Dipanjan Das, and Ellie Pavlick. 2019b. What do you learn from context? Probing for sentence structure in contextualized word representations. *ArXiv*, abs/1905.06316.

Akira Utsumi. 2020. Exploring what is encoded in distributional word vectors: A neurobiologically motivated analysis. *Cognitive science*, 44 6:e12844.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Ł ukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc.

Ivan Vulić, Edoardo Maria Ponti, Robert Litschko, Goran Glavaš, and Anna Korhonen. 2020. Probing pretrained language models for lexical semantics. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 7222–7240, Online. Association for Computational Linguistics.

Aaron Steven White, Drew Reisinger, Keisuke Sakaguchi, Tim Vieira, Sheng Zhang, Rachel Rudinger, Kyle Rawlins, and Benjamin Van Durme. 2016. Universal decompositional semantics on Universal Dependencies. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 1713–1723, Austin, Texas. Association for Computational Linguistics.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Rémi Louf, Morgan Funtowicz, and Jamie Brew. 2019. HuggingFace's Transformers: State-of-the-art Natural Language Processing. *ArXiv*, abs/1910.03771.

Zhilin Yang, Zihang Dai, Yiming Yang, Jaime G. Carbonell, Ruslan Salakhutdinov, and Quoc V. Le. 2019. Xlnet: Generalized autoregressive pretraining for language understanding. *CoRR*, abs/1906.08237.

A Appendix

Here we report the results obtained computing the average Spearman's Rho between predicted vectors and original ones at the type-level, both by row and by column. We indicate each group with the formula BERT$_{x_y}$ in which x is the first layer of the group and y is the last one.

Model	By Row	By column
BERT$_{1\_4}$	0.71	0.31
BERT$_{5\_8}$	0.73	0.36
BERT$_{9\_12}$	0.72	0.34
BERT$_{13\_16}$	**0.74**	**0.39**
BERT$_{17\_20}$	0.73	0.36
BERT$_{21\_24}$	0.72	0.34
baseline_BERT	0.64	-0.07
SGNS.syn	0.71	0.31
baseline_SGNS	0.62	0.035

Table 3: Average correlation values obtained for each group of BERT layers and for the best performing model in Lebani and Lenci (2021), a SGNS.syn, with relative baselines. Analysis at the type-level.

Towards Structure-aware Paraphrase Identification with Phrase Alignment Using Sentence Encoders

Qiwei Peng David Weir Julie Weeds
University of Sussex
Brighton, UK
{qiwei.peng, d.j.weir, j.e.weeds}@sussex.ac.uk

Abstract

Previous works have demonstrated the effectiveness of utilising pre-trained sentence encoders based on their sentence representations for meaning comparison tasks. Though such representations are shown to capture hidden syntax structures, the direct similarity comparison between them exhibits weak sensitivity to word order and structural differences in given sentences. A single similarity score further makes the comparison process hard to interpret. Therefore, we here propose to combine sentence encoders with an alignment component by representing each sentence as a list of predicate-argument spans (where their span representations are derived from sentence encoders), and decomposing the sentence-level meaning comparison into the alignment between their spans for paraphrase identification tasks. Empirical results show that the alignment component brings in both improved performance and interpretability for various sentence encoders. After closer investigation, the proposed approach indicates increased sensitivity to structural difference and enhanced ability to distinguish non-paraphrases with high lexical overlap.

1 Introduction

Sentence meaning comparison measures the semantic similarity of two sentences. Specifically, the task of paraphrase identification binarises the similarity as paraphrase or non-paraphrase depending on whether they express similar meanings (Bhagat and Hovy, 2013). This task benefits many natural language understanding applications, like plagiarism identification (Chitra and Rajkumar, 2016) and fact checking (Jiang et al., 2020), where it is important to detect same things said in different ways.

The difference in sentence structures is important for distinguishing their meanings. However, as shown in Table 1 and 3, many existing paraphrase identification datasets exhibit high correlation between positive pairs and the degree of their lexical overlap, such as the Microsoft Research Paraphrase Corpus (MSRP) (Dolan and Brockett, 2005). Models trained on them tend to mark sentence pairs with high word overlap as paraphrases despite clear clashes in meaning. In light of this, Zhang et al. (2019b) utilised word scrambling and back translation to create the Paraphrase Adversaries from Word Scrambling (PAWS) datasets which are mainly concerned with word order and structure by creating paraphrase and non-paraphrase pairs with high lexical overlap. As also shown in these two tables, sentence pairs in the PAWS datasets demonstrate much higher lexical overlap and lower correlation, which requires models to pay more attention to word order and sentence structure to successfully distinguish non-paraphrases from paraphrases.

Recently, various pre-trained sentence encoders have been proposed to produce high-quality sentence embeddings for downstream usages (Reimers and Gurevych, 2019; Thakur et al., 2021; Gao et al., 2021). Such embeddings are compared to derive a similarity score for different meaning comparison tasks, including paraphrase identification. Though widely used, sentence encoders still face challenges from different aspects in case of meaning comparison. Pre-trained models are observed to capture structural information to some extent (Clark et al., 2019; Hewitt and Manning, 2019; Jawahar et al., 2019). However, as we will demonstrate in this work, their direct comparison of two sentence vectors performs poorly on PAWS datasets indicating weak sensitivity to structural difference, though they achieve good performance on other general paraphrase identification datasets like MSRP. In addition, the single similarity score derived from the comparison of two vectors is difficult to interpret. This thus motivates us to find a better way of utilising sentence encoders for meaning comparison.

Elsewhere, researchers have worked on decom-

Proceedings of the 29th International Conference on Computational Linguistics, pages 4113–4123
October 12–17, 2022.

Dataset	Sentence A	Sentence B	Label
MSRP	The Toronto Stock Exchange opened on time and slightly lower.	The Toronto Stock Exchange said it will be business as usual on Friday morning.	N
	More than half of the songs were purchased as albums, Apple said.	Apple noted that half the songs were purchased as part of albums.	Y
PAWS	What factors cause a good person to become bad?	What factors cause a bad person to become good?	N
	The team also toured in Australia in 1953.	In 1953, the team also toured in Australia.	Y

Table 1: Example sentence pairs taken from both MSRP and PAWS datasets. Y stands for paraphrases while N stands for non-paraphrases.

posing sentence-level meaning comparison into comparisons at a lower level, such as word and phrase-level, which largely increased the interpretability (He and Lin, 2016; Chen et al., 2017; Zhang et al., 2019a). Alignment is the core component in these proposed systems, where sentence units at different levels are aligned through either training signals or external linguistic clues, after which a matching score is derived for sentence-level comparison. Here, we argue that, instead of comparing sentence meaning by using sentence embeddings, it would be better to combine sentence encoders with alignment components in a structure-aware way to strengthen the sensitivity to structural difference and to gain interpretability.

An important aspect of sentence meaning is its predicate-argument structure, which has been utilised in machine translation (Xiong et al., 2012) and paraphrase generation (Ganitkevitch et al., 2013; Kozlowski et al., 2003). Given the importance of detecting structural differences in paraphrase identification tasks, we propose to represent each sentence as a list of predicate-argument spans where span representations are derived from sentence encoders, and to decompose sentence-level meaning comparison into the direct comparison between their aligned predicate-argument spans by taking advantage of the Hungarian algorithm (Kuhn, 1956; Crouse, 2016). The sentence-level score is then derived by aggregation over their aligned spans. Without re-training, the proposed alignment-based sentence encoder can be used with enhanced structure-awareness and interpretability.

As pre-trained sentence encoders produce contextualised representations, two phrases of different meaning might be aligned together due to their similar syntactic structure and contexts. For example:

a) *Harris announced on twitter that he will quit.*

b) *James announced on twitter that he will quit.*

Unsurprisingly, the span *Harris announced* will be aligned to the span *James announced* with a high similarity score given that they share exactly the same context and syntactic structure. However, it might be problematic to consider this high similarity score when we calculate the overall score given clear clashes in the meaning at sentence-level. In this regard, we further explore how the contextualisation affects paraphrase identification by comparing aligned phrases based on their de-contextualised representations.

Empirical results show that the inclusion of the alignment component leads to improvements on four paraphrase identification tasks and demonstrates increased ability to detect non-paraphrases with high lexical overlap, plus an enhanced sensitivity to structural difference. Upon closer investigation, we find that applying de-contextualisation to aligned phrases could further help to recognise such non-paraphrases.

In summary, our contributions are as follows:

1) We propose an approach that combines sentence encoders with an alignment component by representing sentences as lists of predicate-argument spans and decomposing sentence-level meaning comparison into predicate-argument span comparison.
2) We provide an evaluation on four different paraphrase identification tasks, which demonstrates both the improved sensitivity to structures and the interpretability at inference time.
3) We further introduce a de-contextualisation step which can benefit tasks that aim to identify non-paraphrases of extremely high lexical overlap.

2 Related Work

2.1 Sentence Encoders

Sentence encoders have been studied extensively in years. Kiros et al. (2015) abstracted the skip-gram

model (Mikolov et al., 2013) to the sentence level and proposed Skip-Thoughts by using a sentence to predict its surrounding sentences in an unsupervised manner. InferSent (Conneau et al., 2017), on the other hand, leveraged supervised learning to train a general-purpose sentence encoder with BiLSTM by taking advantage of natural language inference (NLI) datasets. Pre-trained language models like BERT (Devlin et al., 2019) are widely used to provide a single-vector representation for the given sentence and demonstrate promising results across a variety of NLP tasks. Inspired by InferSent, Sentence-BERT (SBERT) (Reimers and Gurevych, 2019) produces general-purpose sentence embeddings by fine-tuning BERT on NLI datasets. However, as investigated by Li et al. (2020), sentence embeddings produced by pre-trained models suffer from anisotropy, which severely limits their expressiveness. They then proposed a post-processing step to map sentence embeddings to an isotropic distribution which largely improves the situation. Similarly, Su et al. (2021) proposed a whitening operation for post-process, which aims to alleviate the anisotropy problem. Gao et al. (2021), on the other hand, proposed the SimCSE model by fine-tuning pre-trained sentence encoders with a contrastive learning objective (Chen et al., 2020) along in-batch negatives (Henderson et al., 2017; Chen et al., 2017) on NLI datasets, improving both the performance and the anisotropy problem. Though sentence encoders have achieved promising performance, the current way of utilising them for meaning comparison tasks has known drawbacks and could benefit from the fruitful developments of the alignment component, which have been widely used in modelling sentence pair relations.

2.2 Alignment in Sentence Pair Tasks

Researchers have been investigating sentence meaning comparison for years. One widely used method involves decomposing the sentence-level comparison into comparisons at a lower level. MacCartney et al. (2008) aligned phrases based on their edit distance and applied the alignment to NLI tasks by taking average of aligned scores. Shan et al. (2009) decomposed sentence-level similarity score into the direct comparison between events and content words based on WordNet (Miller, 1995). Sultan et al. (2014) proposed a complex alignment pipeline based on various linguistic features, and predicted the sentence-level semantic similarity by taking the proportion of their aligned content words. The alignment between two syntactic trees are used along with other lexical and syntactic features to determine whether two sentences are paraphrases with SVM (Liang et al., 2016).

Similar ideas are combined with neural models to construct alignments based on the attention mechanism (Bahdanau et al., 2015). They can be seen as learning soft alignments between words or phrases in two sentences. Pang et al. (2016) proposed MatchPyramid where a word-level alignment matrix was learned, and convolutional networks were used to extract features for sentence-level classification. More fine-grained comparisons between words are introduced by PMWI (He and Lin, 2016) to better dissect the meaning difference. Wang et al. (2016) put focus on both similar and dissimilar alignments by decomposing and composing lexical semantics over sentences. ESIM (Chen et al., 2017) further allowed richer interactions between tokens. These models are further improved by incorporating context and structure information (Liu et al., 2019), as well as character-level information (Lan and Xu, 2018). Recently, Pre-trained models are exploited to provide contextualised representations for the PMWI (Zhang et al., 2019a). Instead of relying on soft alignments, some other models tried to take the phrase alignment task as an auxiliary task for sentence semantic assessments (Arase and Tsujii, 2019, 2021), and to embed the Hungarian algorithm into trainable end-to-end neural networks to provide better aligned parts (Xiao, 2020). Considering pre-trained sentence encoders are often directly used to provide fixed embeddings for meaning comparison, in this work, we propose to combine them with the alignment component at inference time so that it can be used with enhanced structure-awareness without re-training.

3 Our Approach

Instead of generating a single-vector representation for meaning comparison based on sentence encoders, we propose to represent each sentence as a list of predicate-argument spans and use sentence encoders to provide its span representations. The comparison between two sentences is then based on the alignment between their predicate-argument spans. As depicted in Figure 1, the approach can be considered as a post-processing step and consists of the following main components:

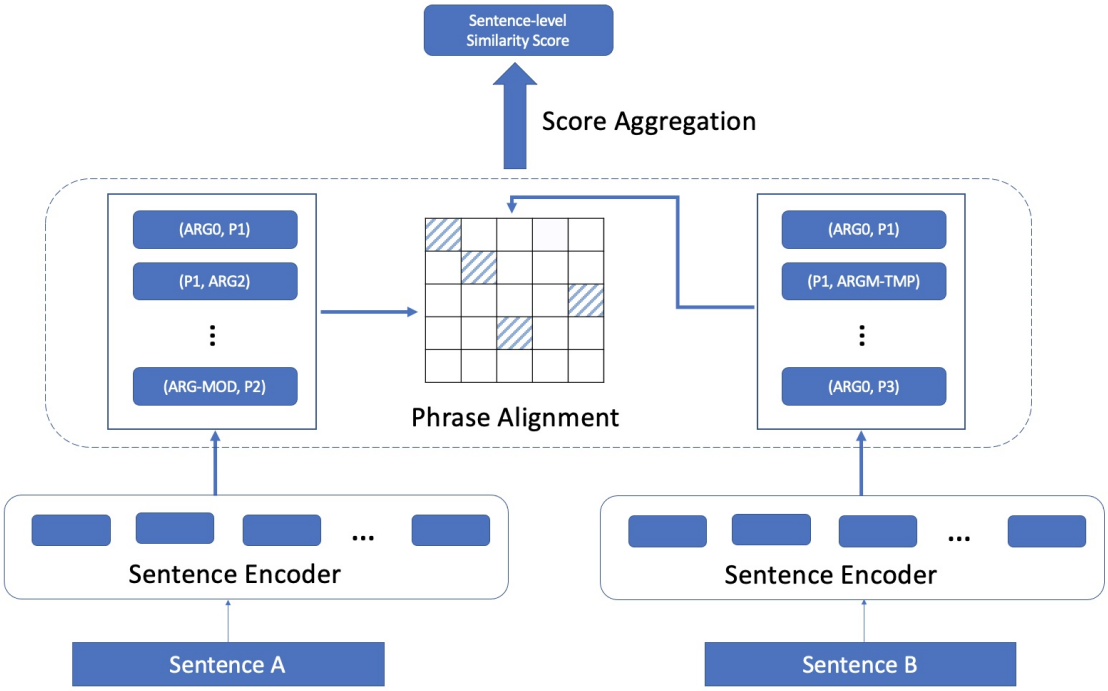

Figure 1: The proposed approach for paraphrase identification that combines sentence encoders with the phrase alignment at inference time. Predicate-argument spans are first extracted from sentences. Span representations are then derived from contextualised token representations. We perform Hungarian algorithm to align extracted phrase spans and obtain the sentence-level similarity score by aggregation over aligned spans. The alignment matrix is useful for interpretation.

Sentence Encoders: The input sentences are first fed into sentence encoders to produce contextualised token representations that will later be used to create context-aware phrase representations from the last hidden layer. The phrase representation will be the basic unit of our meaning comparison method.

Predicate Argument Spans (PAS): For each sentence, we first apply a BERT-based semantic role labelling (SRL) tagger provided by AllenNLP (Gardner et al., 2018) to obtain both predicates and relevant arguments for each sentence. To generate predicate argument spans, we group the predicate and its arguments together and order them according to their original position in the sentence. Following is an example of predicate-argument spans from a sentence:

> *James ate some cheese whilst thinking about the play.*

Two predicates, *ate* and *thinking*, are extracted by the tagger. As shown in Figure 2, a number of arguments with different relations are discovered for each predicate. We further group them into predicate-argument spans. For the given sentence, we will have three spans for the predicate *ate*: (*James*, *ate*), (*ate*, *some*, *cheese*), (*ate*, *whilst*, *thinking*, *about*, *the*, *play*) and two spans for the predicate *thinking*: (*James*, *thinking*), (*thinking*, *about*, *the*, *play*). If no predicate or associated argument is found, we take the whole sentence itself as a long span.

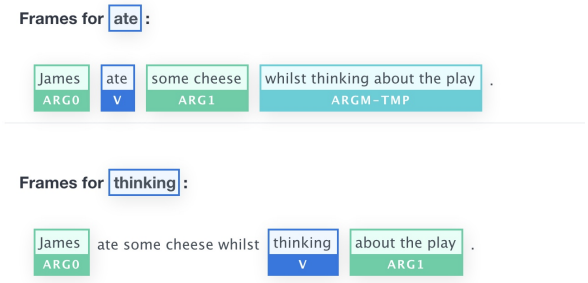

Figure 2: The extracted predicates and relevant semantic arguments for the given example sentence. Outputs are produced by the AllenNLP SRL tagger.

Phrase Alignment: After obtaining all predicate-argument spans, we derive their span representations based on the used encoder. As previous works have shown, aligning with contextual information could achieve better performance and help with disambiguation (Arase and Tsujii, 2020; Dou

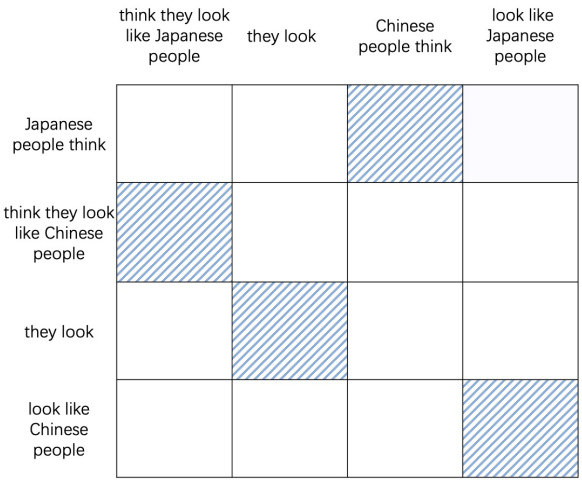

Figure 3: The predicate-argument span alignment between the example pair taken from PAWS_QQP, sentence A: *do Chinese people think they look like Japanese people?* and sentence B: *do Japanese people think they look like Chinese people?*

and Neubig, 2021). We take the mean-pooling over all tokens in the span to produce a contextualised span representation for later alignment. The tokenization strategy in BERT generates sub-tokens, whereas in the produced spans, we have word tokens. To align them properly, we use the same tokenizer to break the original word into sub-tokens and represent it as a list of sub-tokens in the span if a sub-token exists. Given two collections of predicate-argument span representations, $p = \{p_1, p_2, ..., p_M\}$ and $q = \{q_1, q_2, ..., q_N\}$, we are trying to find the best alignments between them. This can be viewed as a standard assignment problem that has been extensively handled by Hungarian algorithm (Kuhn, 1956). A similarity matrix, C, is constructed for each pair of sentences where the row has one collection of spans and the column has another. The value for each entry cell, C_{mn}, is the cosine similarity score between the two span, p_m and q_n. The task of finding the best alignment is to find alignments among two collections that give the maximum score:

$$max \sum_{m} \sum_{n} C_{mn} X_{mn} \quad (1)$$

X is a boolean matrix where $X[m, n] = 1$ if span m is assigned to span n. We apply the modified Jonker-Volgenant algorithm[1] (Crouse, 2016) to find the best alignments that maximise the overall score. After discovering the optimal X, we obtain a collection of aligned span pairs associated with their cosine similarity scores, $A = \{A_1, A_2, ..., A_l\}$. One alignment example taken from PAWS is given in Figure 3.

Score Aggregation: To produce a sentence-level similarity score for the given pair, we simply take the mean-pooling over scores of all aligned parts:

$$Score_{ij} = MeanPooling(A_1, .., A_l) \quad (2)$$

The similarity score between sentence i and sentence j is the average score of their aligned spans, and will be used for determining whether the sentence pair is paraphrase or non-paraphrase. The alignment matrix, as shown in Figure 3, is useful to explain how the overall score is derived and why.

4 Experiments

We follow the same two-step procedure in previous work for evaluation (Li et al., 2020; Thakur et al., 2021). For vanilla sentence encoders, we first generate fixed sentence embeddings, and then derive sentence-level similarity scores by calculating the cosine similarity between two embeddings. For sentence encoders combined with the alignment component, we derive sentence-level similarity scores by aggregation over cosine scores of all aligned spans where span representations are derived from sentence encoders. Otherwise specifically stated, the alignment is performed between predicate-argument spans (PAS). Their performances under these two scenarios are evaluated and compared. We here experiment with three widely used sentence encoders, BERT-base (Devlin et al., 2019), Sentence-BERT (SBERT) (Reimers and Gurevych, 2019) and SimCSE (Gao et al., 2021).

Datasets	Train	Dev	Test
PAWS_QQP	11,986	-	677
PAWS_Wiki	49,401	8,000	8,000
MSRP	3,668	408	1,725
TwitterURL	37,976	4,224	9,334

Table 2: Statistics of all four datasets used in this work.

4.1 Datasets

In this work, we evaluate the proposed approach on four different paraphrase identification tasks.

[1]We use its Scipy implementation: https://docs.scipy.org/doc/scipy/reference/generated/scipy.optimize.linear_sum_assignment.html

The statistics of these datasets are listed in Table 2. Below we give some basic descriptions:

- **PAWS_QQP**: In order to assess the sensitivity to word order and syntactic structure, Zhang et al. (2019b) proposed a paraphrase identification dataset which has extremely high lexical overlap by applying back translation and word scrambling to sentences taken from the Quora Question Pairs (Wang et al., 2017).
- **PAWS_Wiki**: Similar to PAWS_QQP, the same technique is applied to sentences obtained from Wikipedia articles to construct sentence pairs (Zhang et al., 2019b). Both PAWS datasets aim to measure sensitivity of models on word order and sentence structure.
- **Microsoft Research Paraphrase Corpus (MSRP)**: This corpus constructs sentence pairs by clustering news articles with an SVM classifier and human annotations (Dolan and Brockett, 2005). It has 4,076 train data and 1,725 test data. In this paper, we adopt the same split strategy as stated in GLUE (Wang et al., 2019).
- **TwitterURL**: Lan et al. (2017) proposed the TwitterURL corpus where sentence pairs in the dataset are collected by linking tweets that share the same URL of news articles. The corpus contains multiple references of both formal well-edited and informal user-generated texts.

Datasets	Lexical Overlap		
	Positive	Negative	Overall
PAWS_QQP	95.24%	96.79%	96.35%
PAWS_Wiki	84.50%	84.99%	84.77%
MSRP	55.95%	42.60%	51.48%
TwitterURL	29.28%	7.73%	11.94%

Table 3: The lexical overlap between sentence pairs across different datasets. We report both the overall figure and the figures for each class. We calculate the lexical overlap in terms of Jaccard Similarity with ngram=1.

The percentage of lexical overlap between sentence pairs in terms of their labels are summarised in Table 3. It shows that sentence pairs taken from the PAWS datasets generally have higher lexical overlap. Compared to datasets like MSRP and TwitterURL, where negative examples have lower lexical overlap than positive examples, the two PAWS datasets exhibit similar degrees of lexical overlap regardless of their labels. In light of this, we expect that models that are sensitive to word order and sentence structure would demonstrate greater improvements on the PAWS datasets in comparison to models without such sensitivity. Specifically, we put our focus on the PAWS datasets and explore whether different models capture structural differences.

4.2 Implementation Details

For sentence encoders used in this work, we generate sentence embeddings according to their default strategies. For BERT-base[2] and SBERT[3], we use the mean-pooling over the last hidden layer as its sentence representation, and for SimCSE[4], we use the CLS token after the trained MLP layer. For all experiments in this work, no training process is involved. In order to calculate evaluation metrics like accuracy and F1 score, we find optimal thresholds for different metrics on the development set, and apply them on test sets to binary the cosine similarity as paraphrase or non-paraphrase. Given PAWS_QQP does not have development set, we randomly sample 20% of its training data as the development set following the same class distribution. All experiments are conducted on RTX 3090 GPUs.

4.3 Evaluation

The main results are summarised in Table 4, and we report the F1 score of the positive class as well as the overall accuracy. It shows that, with our proposed approach, the performance of different sentence encoders can generally be improved. In addition, significant improvements are observed on PAWS datasets after we introduce the alignment component. This demonstrates the effectiveness of our proposed alignment-based sentence encoder and validates the improved sensitivity to word order and sentence structure. Furthermore, we find that the performance of different models, regardless of combining with the alignment component, is similar or competitive on MSRP and TwitterURL datasets. It suggests that both datasets are inadequate when used to detect the model's structure-

[2]We use its huggingface implementation: https://huggingface.co/bert-base-uncased

[3]https://github.com/UKPLab/sentence-transformers

[4]https://github.com/princeton-nlp/SimCSE

	PAWS_QQP (F1/ACC)	PAWS_Wiki (F1/ACC)	TwitterURL (F1/ACC)	MSRP (F1/ACC)	AVG (F1/ACC)
BERT	37.13/72.97	61.28/56.75	63.24/85.65	80.50/70.38	60.54/71.44
+ Alignment	**47.46/75.18**	**63.08/62.58**	**65.26/86.52**	**80.96/70.61**	**64.19/73.72**
SBERT	33.95/74.74	61.83/60.63	65.61/87.04	81.68/73.39	60.61/73.95
+ Alignment	**52.75/77.70**	**62.52/64.51**	**66.60/87.33**	**82.10/73.80**	**65.99/75.84**
SimCSE	36.16/75.48	61.32/62.58	**69.20/87.74**	**82.80/74.61**	62.37/75.10
+ Alignment	**57.49/79.17**	**65.00/65.99**	67.83/87.27	81.70/73.68	**68.01/76.53**

Table 4: Results on four paraphrase identification tasks, we report both the F1 score of the positive class and the overall accuracy. Cells marked bold have the best performance in each column.

Models	PAWS_QQP (F1/ACC)	PAWS_Wiki (F1/ACC)
BERT-TokenLevel	40.13/73.41	62.33/62.36
BERT-RandomSpan	19.91/73.71	61.30/57.19
BERT-ContinuousRandom	39.86/74.74	61.25/57.66
BERT-PAS	**47.46/75.18**	**63.08/62.58**
SBERT-TokenLevel	47.51/75.04	61.65/64.15
SBERT-RandomSpan	38.89/73.12	61.07/59.08
SBERT-ContinuousRandom	46.56/74.74	61.28/58.49
SBERT-PAS	**52.75/77.70**	**62.52/64.51**
SimCSE-TokenLevel	50.74/74.00	62.03/63.28
SimCSE-RandomSpan	34.31/73.56	61.24/57.70
SimCSE-ContinuousRandom	40.74/77.25	61.30/57.24
SimCSE-PAS	**57.49/79.17**	**65.00/65.99**

Table 5: Evaluation using different span types for alignment. We report the F1 score of the positive class and the overall accuracy.

awareness for the structural information is not required to achieve high scores on them. Accordingly, compared to its alignment version, the lack of sensitivity to structural differences translates the slightly better performance on TwitterURL and MSRP obtained by SimCSE into much worse performance on the PAWS datasets. This further supports our previous arguments and demonstrates the advantages of introducing the alignment component to enhance structure-awareness.

5 Analysis

To better understand the improvements, we have conducted several experiments to investigate different aspects of the proposed approach. Given we are mostly interested in the performance on the two PAWS datasets, we only experiment and report the results on the PAWS_QQP and PAWS_Wiki in the following experiments.

5.1 Comparison to Other Span Strategies

In this experiment, we consider three more scenarios with different span types, and investigate the impact of the predicate-argument span. The alignment between different tokens are widely used in previous works, so here, instead of aligning predicate-argument spans, we directly conduct alignment at token-level. Two further strategies are explored regarding phrase-level alignment. Firstly, we randomly sample words from the sentence to make a span, where the words in each span might not necessarily be sequential. In the RandomSpan strategy, no linguistically-meaningful structures are preserved. Secondly, we randomly sample continuous word sequences to build a span, where the span must contain sequential texts. In this ContinuousRandom strategy, only sequential relations are preserved. The length of the sampled spans is arbitrary. To make a fair comparison, the number of sampled spans is the same as that of the predicate-argument spans in the sentence. As demonstrated in Table 5, the alignment between predicate-argument spans outperforms all the others. In other words, the model's sensitivity to word order and structural differences can be greatly improved by comparing two sentences' predicate-argument structures.

5.2 Large Improvements in Recall

We have observed significant improvements on PAWS datasets by introducing the alignment between predicate-argument spans in previous experiments. It is crucially important to understand how the improvement is obtained. In this experiment, we look into the recall of positive and negative pairs. In PAWS_Wiki, we find that almost all sentence pairs are classified as positive by vanilla models given the near-zero recall for the negative class as shown in Table 6. Despite utterly incorrect pre-

	PAWS_QQP (recall of +)	PAWS_Wiki (recall of -)
BERT	32.46	0.09
+ Alignment	**36.65**	**25.96**
SBERT	24.08	9.14
+ Alignment	**47.64**	**29.53**
SimCSE	25.65	0.27
+ Alignment	**50.26**	**52.28**

Table 6: Results on PAWS_QQP and PAWS_Wiki. For PAWS_QQP, we report the recall of positive class and for PAWS_Wiki, we report the recall of negative class.

Models	PAWS_QQP (F1/recall of +)	PAWS_Wiki (F1/recall of -)
BERT-Alignment	47.46/36.65	63.08/25.96
+ decontext	**52.50/43.98**	**63.39/45.09**
SBERT-Alignment	52.75/47.64	62.52/29.53
+ decontext	**65.43/64.40**	**66.63/64.38**
SimCSE-Alignment	57.49/50.26	65.00/52.28
+ decontext	**65.16/68.06**	**67.32/54.14**

Table 7: The results on PAWS datasets after applying de-contextualisation. We report the F1 score of the positive class on both datasets, the recall of positive class on PAWS_QQP, and the recall of negative class on PAWS_Wiki.

dictions, it spuriously lowers the performance gap (on PAWS_Wiki) in terms of the F1 score of the positive class as shown in Table 4. After applying the alignment process to sentence encoders, we notice significant improvements in the recall of negative class. About 70% of sentence pairs in the PAWS_QQP have negative labels, which makes vanilla models difficult to distinguish paraphrases from non-paraphrases and mark most of sentence pairs as negative, as evidenced by the low recall for positives in the table. Similarly, we observe significant improvements in recall after introducing the alignment component. The large improvements in recall demonstrate the enhanced ability to distinguish non-paraphrases from paraphrases. Moreover, as shown in Table 4, the improvements in recall are not at the expense of their general performance, since we are still improving on F1 scores and the overall accuracy.

5.3 De-contextualisation

As pre-trained sentence encoders produce contextualised representations, two phrases of different meaning might be aligned for similar syntactic structure and contexts with a high similarity score. In the example shown in Section 1, *Harris announced* will be aligned with *James announced* with a high similarity score given their identical syntactic structure and contexts. However, does such high similarity score make sense when it comes to the task of paraphrase identification? Comparing the meaning of two phrases in the context of their use often helps disambiguate. In this case, the highly similar context instead downplays the difference, while it is the minor difference that changes the whole sentence meaning. This problem is exacerbated in the PAWS datasets given that both PAWS_QQP and PAWS_Wiki have extremely high lexical overlap, with 96.35% and 84.77% respectively, as shown in Table 3. Such high lexical overlap indicates a similar context, and thus a high similarity score between aligned phrases. In this experiment, we align phrases based on their contextualised representations as before but de-contextualise them by sending these phrases, without context, through sentence encoders to produce context-agnostic representations. A similarity score at sentence-level is then derived from cosine similarities between context-agnostic representations.

We show the results in Table 7. It clearly shows that, in spite of losing contextual information, the model with de-contextualisation process appears to improve the performance significantly. Additionally, it suggests that contextualisation might be harmful in situations where we focus on small differences that might change the meaning of the whole.

6 Conclusion

In this work, we propose an approach that combines sentence encoders with an alignment component by representing sentences as lists of predicate-argument spans and decomposing sentence-level meaning comparison into predicate-argument span comparison. Experiments with three widely used sentence encoders show that such method leads to improvements on various paraphrase identification tasks and increases the sensitivity to word order and structural differences between two sentences. The alignment matrix can further be utilised for interpretation. We then demonstrate that applying de-contextualisation to aligned phrases could help to recognise non-paraphrases of extremely high lexical overlap. Our future work includes exploring other alignment algorithms and more application scenarios for alignment-based sentence encoders.

Acknowledgement

We thank all anonymous reviewers for their insightful comments. We would like to further thank Bowen Wang and Wing Yan Li for helpful discussions and proofreading.

References

Yuki Arase and Jun'ichi Tsujii. 2019. Transfer fine-tuning: A BERT case study. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 5393–5404, Hong Kong, China. Association for Computational Linguistics.

Yuki Arase and Junichi Tsujii. 2021. Transfer fine-tuning of bert with phrasal paraphrases. *Computer Speech Language*, 66:101164.

Yuki Arase and Jun'ichi Tsujii. 2020. Compositional phrase alignment and beyond. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1611–1623.

Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. 2015. Neural machine translation by jointly learning to align and translate. In *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*.

Rahul Bhagat and Eduard Hovy. 2013. What is a paraphrase? *Computational Linguistics*, 39(3):463–472.

Qian Chen, Xiaodan Zhu, Zhen-Hua Ling, Si Wei, Hui Jiang, and Diana Inkpen. 2017. Enhanced lstm for natural language inference. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1657–1668.

Ting Chen, Simon Kornblith, Mohammad Norouzi, and Geoffrey Hinton. 2020. A simple framework for contrastive learning of visual representations. In *International conference on machine learning*, pages 1597–1607. PMLR.

A Chitra and Anupriya Rajkumar. 2016. Plagiarism detection using machine learning-based paraphrase recognizer. *Journal of Intelligent Systems*, 25(3):351–359.

Kevin Clark, Urvashi Khandelwal, Omer Levy, and Christopher D Manning. 2019. What does bert look at? an analysis of bert's attention. In *Proceedings of the 2019 ACL Workshop BlackboxNLP: Analyzing and Interpreting Neural Networks for NLP*, pages 276–286.

Alexis Conneau, Douwe Kiela, Holger Schwenk, Loïc Barrault, and Antoine Bordes. 2017. Supervised learning of universal sentence representations from natural language inference data. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 670–680.

David F. Crouse. 2016. On implementing 2d rectangular assignment algorithms. *IEEE Transactions on Aerospace and Electronic Systems*, 52(4):1679–1696.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, Minneapolis, Minnesota. Association for Computational Linguistics.

William B. Dolan and Chris Brockett. 2005. Automatically constructing a corpus of sentential paraphrases. In *Proceedings of the Third International Workshop on Paraphrasing (IWP2005)*.

Zi-Yi Dou and Graham Neubig. 2021. Word alignment by fine-tuning embeddings on parallel corpora. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*, pages 2112–2128.

Juri Ganitkevitch, Benjamin Van Durme, and Chris Callison-Burch. 2013. Ppdb: The paraphrase database. In *Proceedings of the 2013 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 758–764.

Tianyu Gao, Xingcheng Yao, and Danqi Chen. 2021. Simcse: Simple contrastive learning of sentence embeddings. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 6894–6910.

Matt Gardner, Joel Grus, Mark Neumann, Oyvind Tafjord, Pradeep Dasigi, Nelson F Liu, Matthew E Peters, Michael Schmitz, and Luke Zettlemoyer. 2018. Allennlp: A deep semantic natural language processing platform. In *Proceedings of Workshop for NLP Open Source Software (NLP-OSS)*, pages 1–6.

Hua He and Jimmy Lin. 2016. Pairwise word interaction modeling with deep neural networks for semantic similarity measurement. In *Proceedings of the 2016 conference of the north American chapter of the Association for Computational Linguistics: human language technologies*, pages 937–948.

Matthew Henderson, Rami Al-Rfou, Brian Strope, Yun-Hsuan Sung, László Lukács, Ruiqi Guo, Sanjiv Kumar, Balint Miklos, and Ray Kurzweil. 2017. Efficient natural language response suggestion for smart reply. *arXiv preprint arXiv:1705.00652*.

John Hewitt and Christopher D Manning. 2019. A structural probe for finding syntax in word representations. In *Proceedings of the 2019 Conference of the North*

American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers), pages 4129–4138.

Ganesh Jawahar, Benoît Sagot, and Djamé Seddah. 2019. What does bert learn about the structure of language? In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 3651–3657.

Shan Jiang, Simon Baumgartner, Abe Ittycheriah, and Cong Yu. 2020. *Factoring Fact-Checks: Structured Information Extraction from Fact-Checking Articles*, page 1592–1603. Association for Computing Machinery, New York, NY, USA.

Ryan Kiros, Yukun Zhu, Ruslan Salakhutdinov, Richard S Zemel, Antonio Torralba, Raquel Urtasun, and Sanja Fidler. 2015. Skip-thought vectors. In *Proceedings of the 28th International Conference on Neural Information Processing Systems-Volume 2*, pages 3294–3302.

Raymond Kozlowski, Kathleen F McCoy, and K Vijay-Shanker. 2003. Generation of single-sentence paraphrases from predicate/argument structure using lexico-grammatical resources. In *Proceedings of the second international workshop on Paraphrasing*, pages 1–8.

Harold W Kuhn. 1956. Variants of the hungarian method for assignment problems. *Naval research logistics quarterly*, 3(4):253–258.

Wuwei Lan, Siyu Qiu, Hua He, and Wei Xu. 2017. A continuously growing dataset of sentential paraphrases. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 1224–1234.

Wuwei Lan and Wei Xu. 2018. Character-based neural networks for sentence pair modeling. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 2 (Short Papers)*, pages 157–163.

Bohan Li, Hao Zhou, Junxian He, Mingxuan Wang, Yiming Yang, and Lei Li. 2020. On the sentence embeddings from pre-trained language models. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 9119–9130, Online. Association for Computational Linguistics.

Chen Liang, Praveen K Paritosh, Vinodh Rajendran, and Kenneth D Forbus. 2016. Learning paraphrase identification with structural alignment. In *IJCAI*, pages 2859–2865.

Linqing Liu, Wei Yang, Jinfeng Rao, Raphael Tang, and Jimmy Lin. 2019. Incorporating contextual and syntactic structures improves semantic similarity modeling. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 1204–1209.

Bill MacCartney, Michel Galley, and Christopher D. Manning. 2008. A phrase-based alignment model for natural language inference. In *Proceedings of the 2008 Conference on Empirical Methods in Natural Language Processing*, pages 802–811, Honolulu, Hawaii. Association for Computational Linguistics.

Tomás Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013. Efficient estimation of word representations in vector space. In *1st International Conference on Learning Representations, ICLR 2013, Scottsdale, Arizona, USA, May 2-4, 2013, Workshop Track Proceedings*.

George A Miller. 1995. Wordnet: a lexical database for english. *Communications of the ACM*, 38(11):39–41.

Liang Pang, Yanyan Lan, Jiafeng Guo, Jun Xu, Shengxian Wan, and Xueqi Cheng. 2016. Text matching as image recognition. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 30.

Nils Reimers and Iryna Gurevych. 2019. Sentence-bert: Sentence embeddings using siamese bert-networks. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 3982–3992.

Jian-fang Shan, Zong-tian Liu, and Wen Zhou. 2009. Sentence similarity measure based on events and content words. In *2009 Sixth International Conference on Fuzzy Systems and Knowledge Discovery*, volume 7, pages 623–627.

Jianlin Su, Jiarun Cao, Weijie Liu, and Yangyiwen Ou. 2021. Whitening sentence representations for better semantics and faster retrieval. *arXiv preprint arXiv:2103.15316*.

Md Arafat Sultan, Steven Bethard, and Tamara Sumner. 2014. Dls @ cu: Sentence similarity from word alignment. In *SemEval@ COLING*, pages 241–246.

Nandan Thakur, Nils Reimers, Johannes Daxenberger, and Iryna Gurevych. 2021. Augmented SBERT: Data augmentation method for improving bi-encoders for pairwise sentence scoring tasks. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 296–310, Online. Association for Computational Linguistics.

Alex Wang, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel R. Bowman. 2019. GLUE: A multi-task benchmark and analysis platform for natural language understanding. In *7th International Conference on Learning Representations, ICLR 2019, New Orleans, LA, USA, May 6-9, 2019*. OpenReview.net.

Zhiguo Wang, Wael Hamza, and Radu Florian. 2017. Bilateral multi-perspective matching for natural language sentences. In *Proceedings of the 26th International Joint Conference on Artificial Intelligence*, pages 4144–4150.

Zhiguo Wang, Haitao Mi, and Abraham Ittycheriah. 2016. Sentence similarity learning by lexical decomposition and composition. In *Proceedings of COLING 2016, the 26th International Conference on Computational Linguistics: Technical Papers*, pages 1340–1349.

Han Xiao. 2020. Hungarian layer: A novel interpretable neural layer for paraphrase identification. *Neural Networks*, 131:172–184.

Deyi Xiong, Min Zhang, and Haizhou Li. 2012. Modeling the translation of predicate-argument structure for smt. In *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 902–911.

Yinan Zhang, Raphael Tang, and Jimmy Lin. 2019a. Explicit pairwise word interaction modeling improves pretrained transformers for english semantic similarity tasks. *arXiv preprint arXiv:1911.02847*.

Yuan Zhang, Jason Baldridge, and Luheng He. 2019b. Paws: Paraphrase adversaries from word scrambling. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 1298–1308.

CILex: An Investigation of Context Information for Lexical Substitution Methods

Sandaru Seneviratne[1], **Elena Daskalaki**[1], **Artem Lenskiy**[1], **Hanna Suominen**[1,2]
[1]The Australian National University (ANU) / Canberra, ACT, Australia
[2]University of Turku / Turku, Finland
{sandaru.seneviratne, eleni.daskalaki,
artem.lenskiy, hanna.suominen}@anu.edu.au

Abstract

Lexical substitution, which aims to generate substitutes for a target word given a context, is an important natural language processing task useful in many applications. Due to the paucity of annotated data, existing methods for lexical substitution tend to rely on manually curated lexical resources and contextual word embedding models. Methods based on lexical resources are likely to miss relevant substitutes whereas relying only on contextual word embedding models fails to provide adequate information on the impact of a substitute in the entire context and the overall meaning of the input. We proposed CILex, which uses contextual sentence embeddings along with methods that capture additional *C*ontext *I*nformation complimenting contextual word embeddings for *Lex*ical substitution. This ensured the semantic consistency of a substitute with the target word while maintaining the overall meaning of the sentence. Our experimental comparisons with previously proposed methods indicated that our solution is now the state-of-the-art on both the widely used LS07 and CoInCo datasets with $P@1$ scores of 55.96% and 57.25% for lexical substitution. The implementation of the proposed approach is available at https://github.com/sandaruSen/CILex under the MIT license.

1 Introduction

Lexical substitution is an important Natural Language Processing (NLP) task, which aims to generate and rank suitable candidate words to replace a given target word, while maintaining the meaning of the given sentence. Lexical substitution is used in a wide range of NLP tasks like data augmentation, paraphrase generation, word sense induction, or text simplification (Shardlow, 2014; Amrami and Goldberg, 2018).

Through the years, different approaches have been introduced for lexical substitution but, due to the paucity of annotated data, most of the lexical substitution systems rely on unsupervised methods based on lexical resources or pre-trained language models (Lacerra et al., 2021). Earlier, methods typically relied entirely on manually curated lexical resources like WordNet (Miller, 1995). The synonyms obtained from such resources were then ranked based on their suitability evaluated by a similarity metric and predefined rules. Some approaches used vector-based modelling and distributional vectors based on syntactic context to obtain the most suitable synonyms (Melamud et al., 2015b). Recent advances in contextual language models like Bidirectional Encoder Representations from Transformers (BERT) (Devlin et al., 2019), Embeddings from Language Models (ELMo) (Peters et al., 2018), and XLNet (Yang et al., 2019) have resulted in major breakthroughs in NLP. Because these models carry contextual information and have the ability of context-sensitive modelling of word probabilities, they have achieved the state-of-the-art (SOTA) results in lexical substitution as well. Some recent research efforts have improved lexical substitution by modifying the architecture of contextual embedding models (Zhou et al., 2019) whereas others integrated lexical resources to contextual embeddings to obtain the most suitable set of substitutes (Michalopoulos et al., 2022).

Methods based on lexical resources may fail to obtain the most relevant substitutes given that they predominantly focus on synonyms, hypernyms, and hyponyms. Moreover, they fail to consider the influence of the substitute on the global context of the given sentence (Zhou et al., 2019). Even though the contextual word embedding models consider the given context, they are unable to provide sufficient knowledge about the effect of the substitute on the overall meaning of a sentence.

To address these issues, the aim of this paper was to investigate the effect of introducing contextual sentence embeddings alongside contextual word

Proceedings of the 29th International Conference on Computational Linguistics, pages 4124–4135
October 12–17, 2022.

embeddings in the lexical substitution task. We first analysed the impact of the addition of contextual sentence information and then, investigated other methods to improve lexical substitution (Zhou et al., 2019; Michalopoulos et al., 2022). The proposed solution achieved the SOTA results on the LS07 and CoInCo datasets.

The main contributions of the paper were as follows:

- Analysis of the impact of adding sentence context for lexical substitution.
- Analysis of methods, which incorporate lexical resources and additional context information to improve lexical substitution.
- A lexical substitution solution, which outperformed previous SOTA methods, and its release at `https://github.com/sandaruSen/CILex` under the MIT license.

2 Related Work

Researchers have identified different subtasks under lexical substitution, namely substitution generation, substitution selection, and substitution ranking (Shardlow, 2014). Out of these, substitution generation and substitution ranking are considered as the two main subtasks, where the former focuses on generating possible substitutes for a target word given the context, and the latter aims to rank the substitutes (Giuliano et al., 2007; Martinez et al., 2007). Ranking of the substitutes may include ranking of the generated substitutes by the lexical substitution method or a much simpler ranking problem with ranking of the set of substitutes obtained from the human-annotated data given in the dataset (Erk and Padó, 2010; Thater et al., 2011).

Early efforts on lexical substitution relied mainly on manually curated lexical resources like WordNet (Miller, 1995) which evolved to the use of unsupervised methods and models based on distributional similarity. Word embeddings, such as word2vec (Mikolov et al., 2013) were used to obtain substitutes by selecting words with embeddings residing near the target word. The embedding similarity obtained from these models was used to rank the substitutes (Melamud et al., 2015b). The model context2vec, introduced by Melamud et al. (2016), produced the contextual embeddings for a given target word by combining the output of two bidirectional Long Short-Term Memory Networks (LSTMs) using a feedforward neural network. This model was successfully applied for the ranking of given substitutes in the lexical substitution task. ELMo used a similar approach with bidirectional LSTMs where the embedding of a given word was created based on the meaning of the context it appeared (Peters et al., 2018). ELMo was used in the lexical substitution task to rank the candidates by calculating the cosine similarity between the contextual embeddings from the ELMo for the target word and all the substitutes for the target word (Garí Soler et al., 2019).

The introduction of transformers resulted in major advances in a wide range of NLP tasks (Vaswani et al., 2017). Transformer-based language models trained on extra large corpora like BERT (Devlin et al., 2019) and a robustly optimised BERT (RoBERTa) (Liu et al., 2019) used a masked language modelling objective where tokens were replaced by a special token [MASK] in the training process. Further improving on the BERT-based language models, XLNet was introduced; it used an autoregressive pre-training method with a permutation-based language modelling objective without corrupting the input with masks (Yang et al., 2019). These contextual embedding models were extensively used for lexical substitution.

The authors in Zhou et al. (2019) relied on contextual word embeddings for lexical substitution. They modified the BERT architecture with a dropout embedding policy where the target word was partially masked with the aim of providing some information of the target word in the prediction. To evaluate the fitness of possible candidates, the authors introduced a validation score which was computed using representations in the top four layer's of BERT. The proposed method achieved the SOTA results for lexical substitution. Arefyev et al. (2020a) presented an extensive analysis on different contextual embedding models for lexical substitution. The authors, in addition to the model probability predictions for the target word, computed the word embedding similarity of the target with all the words in the model's vocabulary for final predictions. Their experimental comparisons indicated that XLNet had superior performance compared to other contextual word embedding models like ELMo, BERT, and RoBERTa at providing substitutes given no changes in the basic architecture of the models.

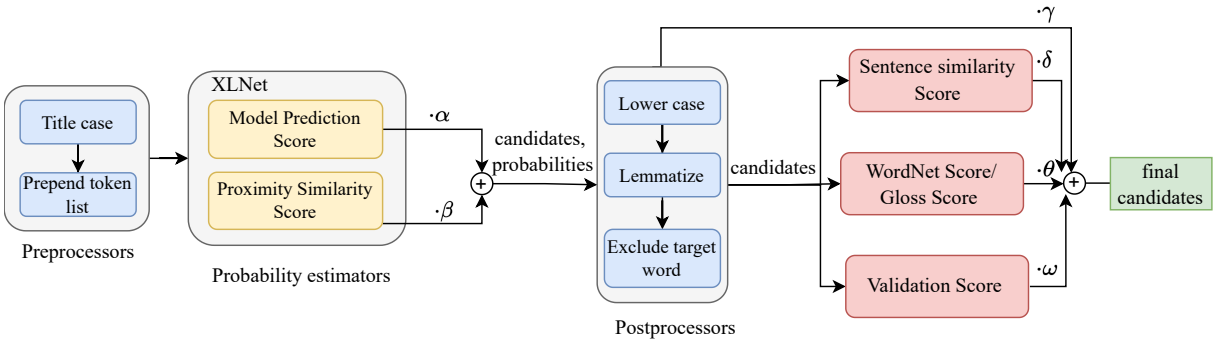

Figure 1: A flowchart of the proposed solution.

Michalopoulos et al. (2022) presented a framework, which integrated external knowledge from WordNet to BERT for lexical substitution. The authors computed a proposal score based on BERT and WordNet, a gloss sentence similarity score based on WordNet definitions, a sentence similarity score using a contextual sentence embedding model, and the validation score introduced by Zhou et al. (2019) to obtain the final set of substitutes. Compared to Zhou et al. (2019) and Arefyev et al. (2020a), additionally their approach integrated both WordNet and contextual sentence embeddings for lexical substitution. These authors' most recent methods on lexical substitution relied mostly on contextual word embedding models (Zhou et al., 2019; Arefyev et al., 2020a) and the use of a variety of methods that provide contextual information (Michalopoulos et al., 2022).

In our study, we specifically focused on the added value of sentence context for lexical substitution based on contextual sentence embeddings. We based our experiments on Arefyev et al. (2020a) which gave evidence of XLNet outperforming other contextual embedding models for lexical substitution, and analysed the impact of adding sentence context information. Additionally, we introduced a WordNet-based metric and investigated the methods proposed by Zhou et al. (2019); Michalopoulos et al. (2022) for lexical substitution.

3 The CILex Solution

In this study, we investigated the impact of sentence context and the methods that capture context information, and proposed a lexical substitution solution called CILex (Figure 1). We followed Arefyev et al. (2020a) and Michalopoulos et al. (2022) as the basis of our work.

3.1 Preprocessing Methods

To address the performance degradation of XLNet model for short contexts (Arefyev et al., 2020a), we explored two main preprocessing steps, namely, converting to title case and prepending strings to the input. Following Arefyev et al. (2020a), we tested out prepending two types of strings (a random set of strings followed by a meaningful string) to increase the input length and assessed the impact in the performance of our models. We observed a slight improvement in the results when only a meaningful string was prepended. When prepending with XLNet, to ensure the separation of the input and the string and to define the beginning of a sentence, use of a special end-of-document (<*eod*>) token (Arefyev et al., 2020b) and conversion of the first word in a sentence to title case were performed.

3.2 Contextual Word Embedding-based Scores

Model Prediction Score. Given a target word x and its context c, we obtained the probability provided by the XLNet model $P(w|c)$ as the model prediction (w is any word from the XLNet vocabulary). We used the XLNet model following Arefyev et al. (2020a) which gave evidence of XLNet outperforming other contextual word embedding models like BERT, ELMo, and RoBERTa without fine-tuning.

Proximity Similarity Score. In addition to the model prediction, we obtained the probability of possible substitutes based on their proximity to the target word $P(w|x)$ through embedding similarity which was computed using the inner product of the embedding of the target word and the embedding of the respective word $(embedding_x \cdot embedding_w^\top)$ (Arefyev et al., 2020a).

These probability scores were linearly combined for each word in the vocabulary to obtain $S_{\texttt{XLNet}}$ which is a representation of the model prediction and the embedding similarity (Eq. (1)).

$$S_{\texttt{XLNet}} = \alpha P(w|c) + \beta P(w|x) \quad (1)$$

where α is the weight for model prediction score and β is the weight for embedding similarity score. The values for parameters α and β can be fine-tuned.

Based on $S_{\texttt{XLNet}}$ score, the words were ranked to obtain the top 20 possible substitutes.

3.3 Contextual Sentence Embedding-based Scores

To evaluate the suitability of the possible candidates and their influence in the global context of the given sentence, we used contextual sentence embeddings with the assumption that contextual sentence embeddings are capable of ensuring that the possible substitutes do not change the overall meaning of the sentence.

Given a sentence s with a target word x_i, we obtained an updated sentence (s') by replacing the target word with a possible substitute. An updated sentence can be denoted as $s' = (x_1, \ldots, x'_i, \ldots)$. For each possible substitute, a sentence similarity score was then calculated using cosine similarity using the sentence embeddings for the original sentence s and the updated sentence s':

$$S_{\texttt{sent}} = \texttt{cos}(s, s'). \quad (2)$$

To obtain sentence embeddings, we experimented with a general sentence embedding model based on RoBERTa (*stsb-roberta-large*) (Reimers et al., 2019), unlike Michalopoulos et al. (2022) who used a fine-tuned RoBERTa sentence embedding model.

To investigate the added value of sentence context, the scores from the XLNet model and the sentence similarity model were linearly combined to obtain the candidate score S for the possible substitutes with γ and δ as the weights for $S_{\texttt{XLNet}}$ and $S_{\texttt{sent}}$ scores respectively. The model which relied only on $S_{\texttt{XLNet}}$ and $S_{\texttt{sent}}$ was defined as *CILex1*.

$$S = \gamma S_{\texttt{XLNet}} + \delta S_{\texttt{sent}}. \quad (3)$$

3.4 Additional Context Information-based Scores

Gloss Sentence Similarity Score. As introduced by Michalopoulos et al. (2022), we computed a gloss sentence similarity score $S_{\texttt{gloss}}$ based on WordNet and BERT (*bert-large-uncased*) which captured additional context information of the target word. For target words and possible substitutes, lists of potential definitions were obtained from WordNet. By computing the similarity score between the given sentence and the definitions, the most suitable definitions for each target word and substitute was obtained. For each substitute, a gloss similarity score was obtained by computing the cosine similarity between the best definition embedding of the target word d_t and best definition embedding of the substitute d_w.

$$S_{\texttt{gloss}} = \texttt{cos}(d_t, d_w). \quad (4)$$

WordNet Similarity Score. Similar to the $S_{\texttt{gloss}}$ score, we introduced a new score $S_{\texttt{wordnet}}$ based on WordNet and BERT (*bert-large-uncased*). Unlike $S_{\texttt{gloss}}$ score, we obtained lists of potential definitions only for the target words, from which the most suitable definition for the target word was obtained computing cosine similarity score between the given sentence and the definitions. By replacing the target word in the given sentence by possible substitutes, a list of updated sentences were obtained. For each substitute, wordnet based similarity score was obtained by computing the cosine similarity between the best definition of the target word d_t and the updated sentence s'.

$$S_{\texttt{wordnet}} = \texttt{cos}(d_t, s'). \quad (5)$$

Validation Score. We also used the validation score $S_{\texttt{val}}$ in Zhou et al. (2019) by computing the cosine similarities between the BERT-based contextual embeddings (*bert-large-uncased*) of the top four layers of every token in the original sentence and the modified sentence.

For each word filtered based on $S_{\texttt{XLNet}}$ score, $S_{\texttt{sent}}$, $S_{\texttt{gloss}}$, $S_{\texttt{wordnet}}$, and $S_{\texttt{val}}$ scores were calculated. The scores were then linearly interpolated to obtain the candidate score S for the possible substitutes with γ, δ, θ, and ω as the weights for $S_{\texttt{XLNet}}$, $S_{\texttt{sent}}$, $S_{\texttt{wordnet}}$, and $S_{\texttt{val}}$ scores respectively for *CILex2* (Eq. (6)). For *CILex3*, $S_{\texttt{wordnet}}$ score was replaced using $S_{\texttt{gloss}}$ (Eq. (7)) [1].

$$S = \gamma S_{\texttt{XLNet}} + \delta S_{\texttt{sent}} + \theta S_{\texttt{wordnet}} + \omega S_{\texttt{val}}. \quad (6)$$

$$S = \gamma S_{\texttt{XLNet}} + \delta S_{\texttt{sent}} + \theta S_{\texttt{gloss}} + \omega S_{\texttt{val}}. \quad (7)$$

[1] Both $S_{\texttt{wordnet}}$ and $S_{\texttt{gloss}}$ are based on WordNet and BERT models, and therefore not considered together.

Sentence	Candidate Substitutes with Weights
If we **take** the factual context in which the term is used into consideration ...	consider 2; accept 1; include 1; think about 1
It shouldn't **take** that long .	last 2; be 1; engage for 1
If you don't **take** the risk of dying by driving to the store ...	tolerate 1; run 1; undergo 1; accept 1; risk 1

Table 1: Three example instances for the target word $take$ from LS07 dataset.

4 Experiments

This section gives an overview of the datasets used in the experiments, evaluation metrics, the experimental setup, and the results from performance evaluations.

4.1 Datasets in Experiments

We evaluated CILex1, CILex2, and CILex3 on the two most widely used English datasets for lexical substitution task: the SemEval 2007 task dataset (LS07) (McCarthy and Navigli, 2007) and the Concepts in Context (CoInCo) (Kremer et al., 2014) dataset.

LS07 dataset is from the English Internet Corpus and consists of $2,010$ sentences for 201 target words with 10 sentences per target word. The annotators were asked to provide up to 3 candidate words for each target word.

CoInCo dataset is from the "Manually Annotated Sub-Corpus" and it consists of $15,415$ sentences with $3,874$ target words. For each target word, 6 candidate substitutes were provided by the annotators.

Each candidate substitute in the datasets was assigned a weight, which corresponds to the frequency it was chosen by the annotators. Table 1 provides an example of three instances of the target word $take$ with the candidate substitutes provided by the annotators in LS07 dataset.

4.2 Experimental Setup

We evaluated CILex on the two subtasks of lexical substitution: substitution generation and substitution ranking.

In the substitution generation task, possible substitutes for the target word were obtained from our proposed approach. We based our evaluation on the metrics proposed in the SemEval 2007 task (McCarthy and Navigli, 2007), in particular, we used *best* and *best-m* to evaluate the quality of the best predictions of the system and out of ten (*oot*) and *oot-m* to assess the coverage of the gold substitutes in the top ten best predictions respectively.[2] We also used $precision@1$ ($P@1$) and $precision@3$ ($P@3$) as evaluation metrics to have a thorough comparison with Zhou et al. (2019); Arefyev et al. (2020a); Michalopoulos et al. (2022). We computed the $P@k$, $k = \{1, 3\}$ as follows:

$$P@k = \frac{\text{acceptable substitutes in the system top-}k}{\text{substitutes in the system top-}k}.$$

To evaluate the statistical significance of the $P@1$ score of CILex methods, we used Wilcoxon Signed-Rank Test (Wilcoxon, 1992) and Pearson correlation (Benesty et al., 2009).

The substitution ranking task was performed based on the substitutes provided in the dataset. Following the previous works, we pooled all the candidate substitutes for the target word in the given instance across the dataset based on the target lemma and Part Of Speech (POS) tag and removed multi-words from the list (Melamud et al., 2015b; Arefyev et al., 2020a; Michalopoulos et al., 2022). The filtered out list was the input to the system as candidates to be ranked. As the gold standard, we used the given candidate substitutes. The proposed approach was then used to rank the possible candidates. CILex was evaluated for the candidate ranking task using the Generalised Average Precision (GAP) score (Kishida, 2005) where candidates are ranked based on their weights; candidates with higher weights should be ranked higher.

We experimented with weights when combining the scores together. Scores from the XLNet model (Eq. (1)) were computed by setting the α parameter to 1 and β parameter to 10 following (Arefyev et al., 2020a). When integrating the XLNet score $S_{\texttt{XLNet}}$ with sentence similarity score $S_{\texttt{sent}}$, empirically, we changed the γ parameter to 1, 0.5, and 0.05 keeping δ at 1. We obtained the best results when γ was 0.05. For θ and ω, we used 0.05 and

[2]For brevity, details of all the evaluation metrics are not described. More information can be found at McCarthy and Navigli (2007).

Method	*best*	*best-m*	*oot*	*oot-m*	*P*@1	*P*@3
			LS07 dataset			
Substitute Vector (Melamud et al., 2015a)	12.7	21.7	36.37	52.03	-	-
PIC (Roller and Erk, 2016)	-	-	-	-	19.7	14.8
Transfer Learning (Hintz and Biemann, 2016)	17.2	-	48.8	-	40.8	-
BERT-based substitution (Zhou et al., 2019)	20.3	34.2	55.4	68.4	51.1	-
BERT-based substitution*	12.8	22.1	43.9	59.7	31.7	-
XLNet+embs (Arefyev et al., 2020a)	21.32	37.80	55.04	73.90	50.56	36.29
LexSubCon (Michalopoulos et al., 2022)	21.1	35.5	51.3	68.6	51.7	-
CILex1	22.15	39.02	54.98	74.15	53.38	37.58
CILex2	23.17	40.98	55.51	73.90	55.43	38.15
CILex3	**23.31**	**40.98**	**56.32**	**74.88**	**55.96**	**38.5**
			CoInCo dataset			
Substitute Vector	8.1	17.4	26.7	46.2	-	-
BERT-based substitution	14.5	33.9	45.9	69.9	56.3	-
BERT-based substitution*	11.8	24.2	36.0	56.8	43.5	-
XLNet+embs	15.09	33.02	45.06	71.85	52.57	39.67
LexSubCon	14.0	29.7	38.0	59.2	50.5	-
CILex1	15.96	35.04	45.84	72.12	55.73	41.34
CILex2	16.30	35.73	46.55	72.84	56.77	42.3
CILex3	**16.39**	**35.80**	**46.87**	**72.98**	**57.25**	**42.49**

Table 2: Results of the best implementations of our approach and previous state-of-the-art models for LS07 and CoInCo datasets (Higher the value, better the performance). We reproduced the results of Arefyev et al. (2020a) and included reproduced results of the BERT-based substitution method (Zhou et al., 2019) by Michalopoulos et al. (2022) which is shown in *. Best values are bolded. (Results for the entire dataset can be found at Appendix A.)

0.5. The linear model parameters for LS07 dataset were fine-tuned against CoInCo dataset and vice versa (Arefyev et al., 2020a). We conducted our experiments on a RTX 3090 graphics card with 24 GB memory and CUDA 11.4.

4.3 Experimental Results from Performance Evaluations

Substitution Generation. Our proposed approaches outperformed all the previous SOTA lexical substitution methods for both datasets (Table 2). We compared our best performing approaches (Eq. (3), Eq. (6), Eq. (7)) with the previous best results from substitute vector-based method (Melamud et al., 2015a), PIC (Roller and Erk, 2016), transfer learning-based method (Hintz and Biemann, 2016), BERT for lexical substitution (Zhou et al., 2019), XLNet+embs method (Arefyev et al., 2020a), and LexSubCon method (Michalopoulos et al., 2022). *CILex3* outperformed the most recent method by Michalopoulos et al. (2022) on both datasets by a ∼4% improvement on LS07 dataset and ∼6.75% improvement on CoInCo dataset. *CILex3* also showed an improvement of ∼5% and ∼4.5% on LS07 and CoInCo respectively compared to (Arefyev et al., 2020a).

Method	**LS07**	**CoInCo**
Transfer Learning	51.9	-
Vector Space Modelling	52.5	47.8
PIC	52.4	48.3
Supervised Learning	55.0	-
Substitute Vector	55.1	50.2
context2vec	56.0	47.9
CILex1	56.81	51.68
CILex3	57.83	53.57
CILex2	58.25	53.92
BERT-based	58.6	55.2
XLNet+embs	60.5	55.64
LexSubCon	60.6	58.0

Table 3: Comparison of GAP scores (%) for the candidate ranking task. The results from the transfer learning (Hintz and Biemann, 2016), vector space modelling (Kremer et al., 2014), PIC (Roller and Erk, 2016), supervised learning (Szarvas et al., 2013), substitute vector (Melamud et al., 2015a), context2vec (Melamud et al., 2016), XLNet+embs (Arefyev et al., 2020a), BERT-based lexical substitution (Zhou et al., 2019), and LexSubCon (Michalopoulos et al., 2022) are presented.

Method	*best*	*best-m*	*oot*	*oot-m*	$P@1$	$P@3$	$R@10$	*Runtime*
				LS07 dataset				
$S_{\texttt{XLNet}}$ and $S_{\texttt{sent}}$	**22.15**	**39.02**	54.98	**74.15**	**53.38**	37.58	48.67	32 min 27 sec
$S_{\texttt{XLNet}}$ and $S_{\texttt{sent}}$*	21.76	38.70	**55.27**	73.90	52.38	**37.6**	**48.79**	32 min 29 sec
$S_{\texttt{XLNet}}$ and $S_{\texttt{wordnet}}$	21.53	38.37	54.59	72.76	50.85	35.39	48.22	25 min 57 sec
$S_{\texttt{XLNet}}$ and $S_{\texttt{gloss}}$	20.97	36.59	50.72	69.35	50.5	30.4	44.0	45 min 11 sec
$S_{\texttt{XLNet}}$ and $S_{\texttt{val}}$	21.98	38.46	54.39	72.93	52.73	36.78	48.15	59 min 37 sec
				CoInCo dataset				
$S_{\texttt{XLNet}}$ and $S_{\texttt{sent}}$	**15.96**	**35.04**	45.84	72.12	**55.73**	41.34	37.21	4 hrs 54 min
$S_{\texttt{XLNet}}$ and $S_{\texttt{sent}}$*	15.71	34.39	**46.07**	**72.80**	54.63	**41.88**	**37.31**	4 hrs 59 min
$S_{\texttt{XLNet}}$ and $S_{\texttt{wordnet}}$	15.23	33.39	44.48	70.84	53.07	38.14	35.77	3 hrs 18 min
$S_{\texttt{XLNet}}$ and $S_{\texttt{gloss}}$	14.95	32.35	41.46	66.22	52.37	34.54	33.27	6 hrs 28 min
$S_{\texttt{XLNet}}$ and $S_{\texttt{val}}$	15.63	34.39	44.76	70.57	54.64	39.97	36.26	8 hrs 47 min

Table 4: Ablation study of the proposed approach with $S_{\texttt{XLNet}}$ as the basis and different methods to obtain additional context information. In *, we used the fine-tuned RoBERTa model to compute sentence similarity scores.

All the proposed CILex solutions gave statistically significant improvement of $P@1$ score compared to (Arefyev et al., 2020a) for both datasets (P<0.05). Both CILex2 and CILex3 were statistically significantly better than CILex1. However, based on Pearson's correlation results for CILex2 and CILex3, we could observe a high level of correlation between the two methods.

Candidate Ranking. Our proposed approaches provided competitive results on both LS07 and CoInCo datasets for candidate ranking task (Table 3). We could observe that CILex approaches outperformed the transfer learning (Hintz and Biemann, 2016), vector space modelling (Kremer et al., 2014), PIC (Roller and Erk, 2016), supervised learning (Szarvas et al., 2013), substitute vector (Melamud et al., 2015a), and context2vec (Melamud et al., 2016) methods. However, BERT-based lexical substitution (Zhou et al., 2019), XLNet+embs (Arefyev et al., 2020a), and LexSubCon (Michalopoulos et al., 2022) reported better results than the proposed approach for candidate ranking.

Ablation Study on Substitution Generation. We conducted an ablation study to evaluate the effect of contextual sentence embeddings and different methods that capture context information introduced in (Zhou et al., 2019; Michalopoulos et al., 2022) (Table 4). We have presented results for the recall of the top 10 predictions ($R@10$) and run time for each experiment as additional metrics.

The results from our analysis of the contribution of fine-tuned contextual sentence embeddings and general contextual embeddings indicated that fine-tuned sentence embedding model on the dataset does not necessarily perform well for lexical substitution. We used $S_{\texttt{XLNet}}$ as the basis and experimented with two contextual sentence embedding models based on RoBERTa and the fine-tuned sentence embedding model based on RoBERTa (Michalopoulos et al., 2022). Based on our experiments, we observed that both models gave similar results for LS07 and CoInCo datasets.

We further performed experiments to analyse the relative contribution of additional context information obtained by $S_{\texttt{wordnet}}$, $S_{\texttt{gloss}}$, and $S_{\texttt{val}}$ with respect to the output from $S_{\texttt{XLNet}}$. Our results implied that the model achieves the worst performance when gloss sentence similarity score was used as additional context information. Based on our results, we also observed an increase in the final results when sentence similarity score was used to obtain additional context information.

To identify the efficient methods of integrating contextual information for lexical substitution, we reported the runtimes for our experiments (Table 4). The runtimes indicated that use of $S_{\texttt{wordnet}}$ is comparatively efficient. However, considering the computational vs performance trade-off, desired scores can be used for lexical substitution.

	Number of successful predictions			
Dataset	**0**	**1**	**2**	**3**
LS07	23.19	43.27	28.36	5.16
CoInCo	22.00	37.98	30.53	9.47

Table 5: The percentage of no. of samples in each dataset based on the number of successful predictions in the top three predictions.

Sentence	Gold Substitutes	Top Three Predictions
Nevertheless she gave me what i can only **describe** as as appraising glance	call, see, recount, imagine, detail, assess as	description, explain, define
Just another **wild** and crazy guy.	uninhibited, turbulent, rowdy, restless, peculiar, intense, insane, impassioned, fierce, adventurous	crazy, reckless, random
The federal complaint offers many **details** of the alleged conspiracy ...	specific, point, fact, tidbit, snippet, item, issue, facet, count, account	information, description, outline
He **said**.	state, remark, declare, comment, cite, ask, answer	speak, tell, reply

Table 6: Four example instances from CoInCo dataset whose top three model predictions were not in the gold substitutes and the extracted possible substitutes from WordNet. The target words are bolded.

4.4 Analysis of the Substitution Generation Results

We conducted experiments to further analyse the performance of the CILex architecture. For each dataset, we calculated the percentage of successful predictions as the samples for which the model returned at least one prediction that was included in the gold substitutes, considering the top three predictions (Table 5). We observed that for 76.8% of the LS07 dataset and 77.99% of the CoInCo dataset CILex provided at least one successful prediction. We further analysed the 23.19% and 22% of LS07 and CoInCo datasets for which our method did not yield a successful prediction.

For the target words in the samples which did not yield at least one successful prediction, we extracted synonyms, hypernyms, and hyponyms from WordNet as possible substitutes and checked if at least one of the model predictions was in the extracted WordNet substitutes. Results from CILex indicated that 51.64% of LS07 and 25.69% of CoInCo which did not yield a successful prediction, contained predictions that were included in the extracted WordNet substitutes. This implied that, even though for certain samples there were no successful predictions based on the gold substitutes, they included predictions with a certain relevance to the target word. We manually checked the remaining samples, for which our method did not yield a successful prediction and which were not included in the WordNet substitutes, as illustrated in Table 6. In the analysed instances, we observed predictions which could be considered as possible substitutes for the given target word.

5 Discussion

In this paper, we analysed the impact of introducing contextual sentence embeddings and methods which provide additional context information for lexical substitution, thereby ensuring that the substitutes are semantically consistent while preserving the overall meaning of the sentences. The results from the proposed CILex solution outperformed previous SOTA methods for lexical substitution on LS07 and CoInCo datasets. Our results indicated that accounting for sentence context information has improved the performance on the substitution generation task. This is demonstrated by our approach (i.e., CILex) outperforming (Arefyev et al., 2020a) SOTA contextual word embedding-based method on two datasets. However, interestingly based on our results, the performance did not improve in the candidate ranking task, which requires further investigation.

The results from our ablation study on the methods, which provide additional context information, indicated that they improve the lexical substitution task. Analogously with Michalopoulos et al. (2022), our results implied that the model achieves the worst performance when gloss sentence similarity score was used as additional context information. This could be mainly due to WordNet being manually curated and definitions for words obtained from WordNet might not reflect the meaning of the words in the given context. Furthermore, for certain words, definitions may not be available on WordNet. Based on our results, we also observed an increase in the final results when sentence similarity score was used to obtain additional context information as opposed to wordnet score, gloss sen-

tence similarity score, and validation score. This is likely because the sentence similarity can appropriately identify if a substitute fits the context and ensures that the overall meaning of the sentence is unchanged. However, when compared with Michalopoulos et al. (2022), their approach yielded an improvement in results when the validation score introduced by (Zhou et al., 2019) was used. These observations hindered us to come to a conclusion as to which of the components contributes most to lexical substitution and illustrated that different components contributed differently.

Our experiments gave evidence of the importance of the initial model/method used to obtain the first set of substitutes. The proposed CILex solution, which relied on an XLNet-based method to obtain the initial set of substitutes, outperformed Lexsubcon (Michalopoulos et al., 2022) which used a BERT-based method to obtain the initial set of substitutes. CILex showed an improvement of $\sim$4% on LS07 dataset and $\sim$6.75% on CoInCo dataset. These insights guided us to conclude that the initial model used to obtain the possible candidates also had a direct impact to the lexical substitution which requires a thorough investigation.

The proposed CILex solutions are based on pre-trained language models and therefore are generalisable. For domain specific applications, the proposed approach can be easily transferable by replacing the pre-trained language models with respective domain specific models.

Further analysis of the substitution generation results indicated that samples which did not have successful predictions may contain potential substitutes based on WordNet. This illustrated the impact of the annotation subjectivity in the interpretation of the performance of the lexical substitution task.

As future work, we intend to extend our experiments and analyse methods that can provide context information to improve lexical substitution. Moreover, the impact on the candidate ranking task will be explored further as future work. We also plan to look into the applicability of the proposed approach for other tasks like word sense induction (Amrami and Goldberg, 2018) and word sense disambiguation.

6 Conclusion

We have presented and released a solution for lexical substitution investigating the impact of sentence context obtained from contextual sentence embeddings. We have introduced and further integrated methods which capture additional context information as proposed by Zhou et al. (2019); Michalopoulos et al. (2022). The unified solution has achieved the SOTA results on two benchmark datasets; LS07 and CoInCo.

We have also analysed and evaluated effects of different methods that provide contextual information and their contribution for lexical substitution. The results have demonstrated the importance of sentence context information obtained using contextual sentence embeddings in lexical substitution.

7 Ethical Considerations

We proposed a solution for lexical substitution and analysed the impact of adding sentence context using contextual sentence embeddings. Additionally, we also incorporated scores proposed in (Zhou et al., 2019; Michalopoulos et al., 2022) for lexical substitution to evaluate the significance of each of them.

The proposed approach was tested and validated on two benchmark datasets: LS07 dataset and CoInCo dataset. According to the National Statement on Ethical Conduct in Human Research (2007) — Updated 2018 (National Health and Medical Research Council, 2018), a new ethics approval was not required for our experiments and, to the best of our knowledge, both datasets were created ethically. No new data or annotations were collected as part of our study.

Acknowledgement

This research was funded by and has been delivered in partnership with Our Health in Our Hands (OHIOH), a strategic initiative of the ANU, which aims to transform health care by developing new personalised health technologies and solutions in collaboration with patients, clinicians and healthcare providers. We gratefully acknowledge the funding from the ANU School of Computing for the first author's PhD studies. We wish to thank the reviewers for their valuable comments. We also thank the authors of Arefyev et al. (2020a); Michalopoulos et al. (2022) for releasing their code and clarifying the questions we came across.

References

Asaf Amrami and Yoav Goldberg. 2018. Word sense induction with neural biLM and symmetric patterns.

In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 4860–4867, Brussels, Belgium. Association for Computational Linguistics.

Nikolay Arefyev, Boris Sheludko, Alexander Podolskiy, and Alexander Panchenko. 2020a. Always keep your target in mind: Studying semantics and improving performance of neural lexical substitution. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 1242–1255, Barcelona, Spain (Online). International Committee on Computational Linguistics.

Nikolay Arefyev, Boris Sheludko, Alexander Podolskiy, and Alexander Panchenko. 2020b. A comparative study of lexical substitution approaches based on neural language models. *arXiv preprint arXiv:2006.00031*.

Jacob Benesty, Jingdong Chen, Yiteng Huang, and Israel Cohen. 2009. Pearson correlation coefficient. In *Noise reduction in speech processing*, pages 1–4. Springer.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, Minneapolis, Minnesota. Association for Computational Linguistics.

Katrin Erk and Sebastian Padó. 2010. Exemplar-based models for word meaning in context. In *Proceedings of the ACL 2010 Conference Short Papers*, pages 92–97, Uppsala, Sweden. Association for Computational Linguistics.

Aina Garí Soler, Anne Cocos, Marianna Apidianaki, and Chris Callison-Burch. 2019. A comparison of context-sensitive models for lexical substitution. In *Proceedings of the 13th International Conference on Computational Semantics - Long Papers*, pages 271–282, Gothenburg, Sweden. Association for Computational Linguistics.

Claudio Giuliano, Alfio Gliozzo, and Carlo Strapparava. 2007. FBK-irst: Lexical substitution task exploiting domain and syntagmatic coherence. In *Proceedings of the Fourth International Workshop on Semantic Evaluations (SemEval-2007)*, pages 145–148, Prague, Czech Republic. Association for Computational Linguistics.

Gerold Hintz and Chris Biemann. 2016. Language transfer learning for supervised lexical substitution. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 118–129, Berlin, Germany. Association for Computational Linguistics.

Kazuaki Kishida. 2005. *Property of average precision and its generalization: An examination of evaluation indicator for information retrieval experiments*. National Institute of Informatics Tokyo, Japan.

Gerhard Kremer, Katrin Erk, Sebastian Padó, and Stefan Thater. 2014. What substitutes tell us - analysis of an "all-words" lexical substitution corpus. In *Proceedings of the 14th Conference of the European Chapter of the Association for Computational Linguistics*, pages 540–549, Gothenburg, Sweden. Association for Computational Linguistics.

Caterina Lacerra, Tommaso Pasini, Rocco Tripodi, and Roberto Navigli. 2021. Alasca: an automated approach for large-scale lexical substitution. In *Proceedings of the Thirtieth International Joint Conference on Artificial Intelligence, IJCAI-21*, pages 3836–3842.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Roberta: A robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692*.

David Martinez, Su Nam Kim, and Timothy Baldwin. 2007. MELB-MKB: Lexical substitution system based on relatives in context. In *Proceedings of the Fourth International Workshop on Semantic Evaluations (SemEval-2007)*, pages 237–240, Prague, Czech Republic. Association for Computational Linguistics.

Diana McCarthy and Roberto Navigli. 2007. Semeval-2007 task 10: English lexical substitution task. In *Proceedings of the fourth international workshop on semantic evaluations (SemEval-2007)*, pages 48–53.

Oren Melamud, Ido Dagan, and Jacob Goldberger. 2015a. Modeling word meaning in context with substitute vectors. In *Proceedings of the 2015 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 472–482, Denver, Colorado. Association for Computational Linguistics.

Oren Melamud, Jacob Goldberger, and Ido Dagan. 2016. context2vec: Learning generic context embedding with bidirectional LSTM. In *Proceedings of The 20th SIGNLL Conference on Computational Natural Language Learning*, pages 51–61, Berlin, Germany. Association for Computational Linguistics.

Oren Melamud, Omer Levy, and Ido Dagan. 2015b. A simple word embedding model for lexical substitution. In *Proceedings of the 1st Workshop on Vector Space Modeling for Natural Language Processing*, pages 1–7, Denver, Colorado. Association for Computational Linguistics.

George Michalopoulos, Ian McKillop, Alexander Wong, and Helen Chen. 2022. LexSubCon: Integrating knowledge from lexical resources into contextual embeddings for lexical substitution. In *Proceedings*

of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), pages 1226–1236, Dublin, Ireland. Association for Computational Linguistics.

Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. 2013. Distributed representations of words and phrases and their compositionality. *Advances in neural information processing systems*, 26.

George A Miller. 1995. Wordnet: a lexical database for english. *Communications of the ACM*, 38(11):39–41.

National Health and Medical Research Council. 2018. National Statement on Ethical Conduct in Human Research (2007). https://www.nhmrc.gov.au/about-us/publications/national-statement-ethical-conduct-human-research-2007-updated-2018. [Online; accessed 06-January-2022].

Matthew E. Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer. 2018. Deep contextualized word representations. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 2227–2237, New Orleans, Louisiana. Association for Computational Linguistics.

Nils Reimers, Iryna Gurevych, Nils Reimers, Iryna Gurevych, Nandan Thakur, Nils Reimers, Johannes Daxenberger, Iryna Gurevych, Nils Reimers, Iryna Gurevych, et al. 2019. Sentence-bert: Sentence embeddings using siamese bert-networks. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing*, pages 671–688. Association for Computational Linguistics.

Stephen Roller and Katrin Erk. 2016. PIC a different word: A simple model for lexical substitution in context. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 1121–1126, San Diego, California. Association for Computational Linguistics.

Matthew Shardlow. 2014. A survey of automated text simplification. *International Journal of Advanced Computer Science and Applications*, 4(1):58–70.

György Szarvas, Róbert Busa-Fekete, and Eyke Hüllermeier. 2013. Learning to rank lexical substitutions. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*, pages 1926–1932, Seattle, Washington, USA. Association for Computational Linguistics.

Stefan Thater, Hagen Fürstenau, and Manfred Pinkal. 2011. Word meaning in context: A simple and effective vector model. In *Proceedings of 5th International Joint Conference on Natural Language Processing*, pages 1134–1143, Chiang Mai, Thailand. Asian Federation of Natural Language Processing.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. *Advances in neural information processing systems*, 30.

Frank Wilcoxon. 1992. Individual comparisons by ranking methods. In *Breakthroughs in statistics*, pages 196–202. Springer.

Zhilin Yang, Zihang Dai, Yiming Yang, Jaime Carbonell, Ruslan Salakhutdinov, and Quoc V. Le. 2019. Xlnet: Generalized autoregressive pretraining for language understanding.

Wangchunshu Zhou, Tao Ge, Ke Xu, Furu Wei, and Ming Zhou. 2019. BERT-based lexical substitution. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 3368–3373, Florence, Italy. Association for Computational Linguistics.

A Results of the Entire Dataset

The CILex solutions are based on pre-trained contextual embedding models and therefore no training is performed. Hence, we have also provided the results of the proposed approaches for the entire dataset (Tables 7, 8, and 9).

Method	*best*	*best-m*	*oot*	*oot-m*	$P@1$	$P@3$
			LS07 dataset			
XLNet+embs (Arefyev et al., 2020a)	20.88	36.92	53.60	71.74	49.53	34.9
CILex1	21.50	37.68	53.53	72.02	51.92	36.25
CILex2	22.47	39.43	53.96	71.81	53.77	36.76
CILex3	**22.59**	**39.22**	**54.65**	**72.37**	**54.22**	**37.33**
			CoInCo dataset			
XLNet+embs	14.11	31.70	44.03	71.62	51.5	39.5
CILex1	15.37	34.71	45.09	72.25	56.12	42.24
CILex2	15.67	35.20	45.73	72.88	57.09	43.12
CILex3	**15.67**	**34.97**	**46.03**	**73.07**	**57.35**	**43.27**

Table 7: Results of the best implementations of our approach for the whole dataset (trial and test) and the reproduced results of XLNet+embs method (Arefyev et al., 2020a).

Method	*best*	*best-m*	*oot*	*oot-m*	$P@1$	$P@3$	$R@10$	*Runtime*
				LS07 dataset				
$S_{\texttt{XLNet}}$ and $S_{\texttt{sent}}$	**21.50**	**37.68**	**53.53**	**72.02**	**51.92**	**36.25**	**47.6**	37 min 5 sec
$S_{\texttt{XLNet}}$ and $S_{\texttt{wordnet}}$	20.83	36.99	53.08	70.41	49.23	34.18	47.1	27 min 55 sec
$S_{\texttt{XLNet}}$ and $S_{\texttt{gloss}}$	20.26	35.24	49.27	66.99	48.63	29.21	43.04	55 min 46 sec
$S_{\texttt{XLNet}}$ and $S_{\texttt{val}}$	21.29	37.19	52.78	70.41	50.97	35.1	46.95	1 hr 7 min
				CoInCo dataset				
$S_{\texttt{XLNet}}$ and $S_{\texttt{sent}}$	**15.37**	**34.71**	**45.09**	**72.25**	**56.12**	**42.24**	**36.33**	7 hr 58 min
$S_{\texttt{XLNet}}$ and $S_{\texttt{wordnet}}$	14.53	32.78	43.40	70.72	52.77	38.58	34.62	4 hr 52 min
$S_{\texttt{XLNet}}$ and $S_{\texttt{gloss}}$	14.32	31.82	40.74	66.50	52.41	35.41	32.42	10 hr 6 min
$S_{\texttt{XLNet}}$ and $S_{\texttt{val}}$	15.02	33.91	43.85	70.51	54.59	40.61	35.29	13 hr 58 min

Table 8: Ablation study of the proposed approach for the complete dataset (trial and test).

Method	**LS07**	**CoInCo**
CILex3	56.91	53.55
CILex2	57.42	53.93
BERT-based	57.9	55.5
XLNet+embs	59.61	55.64
CILex1	59.9	55.6
LexSubCon	60.3	58.0

Table 9: Comparison of GAP scores (%) for the candidate ranking task on the entire dataset (trial and test). The results of CILex methods, reproduced results of XLNet+embs (Arefyev et al., 2020a), reported results of BERT-based lexical substitution (Zhou et al., 2019), and LexSubCon by Michalopoulos et al. (2022) are presented.

Emotion Enriched Retrofitted Word Embeddings

Sapan Shah[1,2], **Sreedhar Reddy**[1], **and Pushpak Bhattacharyya**[2]

[1]TCS Research, Tata Consultancy Services, Pune
[2]Indian Institute of Technology Bombay, Mumbai
{sapan.hs,sreedhar.reddy}@tcs.com
pb@cse.iitb.ac.in

Abstract

Word embeddings learned using the distributional hypothesis (e.g., GloVe, Word2vec) are good at encoding various lexical-semantic relations. However, they do not capture the emotion aspects of words. We present a novel retrofitting method for updating the vectors of emotion bearing words like fun, offence, angry, etc. The retrofitted embeddings achieve better inter-cluster and intra-cluster distance for words having the same emotions, e.g., the *joy* cluster containing words like fun, happiness, etc., and the *anger* cluster with words like offence, rage, etc., as evaluated through different cluster quality metrics. For the downstream tasks on sentiment analysis and sarcasm detection, simple classification models, such as SVM and Attention Net, learned using our retrofitted embeddings perform better than their pre-trained counterparts (about 1.5% improvement in F1-score) as well as other benchmarks. Furthermore, the difference in performance is more pronounced in the limited data setting.

1 Introduction

Word embedding models inspired from the distributional hypothesis (Harris, 1954) have one major limitation: they mix semantic similarity with other types of semantic relatedness (Hill et al., 2015). For instance, consider `cheap` and `expensive`. Though opposite in meaning, the distributional vectors of these words are similar since they occur in nearly identical contexts. This is problematic for many applications such as text simplification, dialogue state tracking, etc. To address this, researchers have proposed various models that leverage knowledge resources to improve word embeddings. At a high level, these models are categorized into two types: Joint specialization models (Yu and Dredze, 2014; Liu et al., 2015); and Retrofitting (post-processing) models (Faruqui et al., 2015; Mrkšić et al., 2016). Joint specialization models typically modify the optimization objective of distributional models by integrating external knowledge into the objective function. In contrast, retrofitting models first generate training data from knowledge resources in the form of constraints and then modify the pre-trained embeddings in a post-processing step so that they respect the constraints. These approaches focus mainly on constraints from relations such as synonymy, antonymy, hypernymy, etc., that are present in WordNet, Paraphrase database, etc.

word pair	GloVe	RETripletGBal
(angry, offence)	0.2339	0.3924
(angry, enjoy)	0.3400	0.2950
(fun, closeness)	0.2232	0.3688
(fun, miserable)	0.3105	0.2812

Table 1: Cosine similarity between words from same and different emotion categories: pre-trained GloVe vs. embeddings retrofitted by our method RETripletGBal

While pre-trained embeddings and their retrofitted versions are good at encoding various lexico-semantic relations, they do not consider the emotion content of words. For example, consider words such as `angry` and `offence` that evoke *anger* emotion and words such as `fun` and `enjoy` eliciting *joy*. Table 1 shows cosine similarity as computed using pre-trained GloVe embeddings. Even though `angry` and `offence` evoke the same emotion (*anger*), their cosine similarity is lower than that between `angry` and `enjoy`, a pair of words eliciting different emotions, pointing to the shortcomings of existing embedding models. Recently, a few attempts have used affective lexicons (Khosla et al., 2018; Seyeditabari et al., 2019) or task-dependent distant supervision (Tang et al., 2016; Agrawal et al., 2018) to induce emotion embeddings. While they work well for some tasks, they do not generalize well across tasks and have not been tested extensively for intrinsic quality.

Proceedings of the 29th International Conference on Computational Linguistics, pages 4136–4148
October 12–17, 2022.

Emotion	anger	joy	sadness	fear	anticipation	surprise	trust	disgust
#words	543	651	1153	772	623	318	1197	1024

Table 2: EmoLex statistics: number of words annotated with Plutchik's eight basic emotion categories

In this work, we present a novel retrofitting method to learn emotion enriched embeddings. For knowledge, it relies on word-level emotion annotations available in the NRC word-emotion association lexicon (known as EmoLex). The central idea is: if words w_a and w_p are associated with the same emotion category t and word w_n is not associated with t, then w_a is semantically closer to w_p than w_n in the context of the emotion category t. This can be stated as an inequality constraint: $sim_t(w_a, w_p) > sim_t(w_a, w_n)$. Such emotion inequality constraints containing word triplets (w_a, w_p, w_n) are generated for all emotion categories present in EmoLex. We use these constraints as training data to learn a non-linear transformation function that maps original word vectors to a vector space respecting these constraints. The transformation function is learned in a similarity metric learning setting using a multi-layer feed-forward network.

The embeddings retrofitted using our method achieve better clustering for emotion bearing words. For the downstream tasks on sentiment analysis and sarcasm detection, they perform better than their pre-trained counterparts and other benchmarks, with significant gains in limited data setting. The main contributions of this work are:

1. A novel retrofitting method to learn emotion enriched embeddings in a similarity metric learning setting (Section 3).

2. A detailed evaluation of word embeddings for their emotion content using clustering experiments (Section 4.1).

3. A detailed evaluation on sentiment analysis and sarcasm detection showing the efficacy of our retrofitting method (Section 4.2).

2 Constraints from NRC EmoLex

A large body of work has focussed on understanding and modelling human emotions. For instance, Plutchik's wheel of emotions (Plutchik, 1980), Ekman's model (Ekman, 1992), Parrot's tree-structured emotions (Parrott, 2001), etc. The model proposed by Plutchik arranges emotions in circles with the length of radius indicating the intensity of emotions. It proposes eight basic or primary emotions: joy, trust, fear, surprise, sadness, disgust, anger, and anticipation.

Various lexical resources have been proposed in the literature to capture the emotion aspect of words, e.g. (Mohammad, 2018a,b). In this work, we focus on NRC EmoLex (Mohammad and Turney, 2013). It contains a list of English words and their associations with Plutchik's eight basic emotions (Plutchik, 1980). Since words are ambiguous in their meaning and may evoke multiple emotions, each word in EmoLex has been associated with a set of emotions. For example, `playful` is associated with three emotion categories: *trust*, *surprise* and *joy*. Table 2 shows the total number of words annotated with each emotion category[1].

We obtain a set of inequality constraints from EmoLex in the form of triplets. Each triplet contains three words (w_a, w_p, w_n) in which we refer to w_a, w_p and w_n as the *anchor*, *positive* and *negative* words, respectively. The corresponding inequality constraint is: similarity between w_a and w_p shall be greater than the similarity between w_a and w_n, by at least a margin m. The margin m is set in the range $[0, 2]$ corresponding to a minimum versus maximum separation on cosine distance. For example, consider the following word-emotion pairs in EmoLex: (`lonely`, *sadness*), (`playful`, *joy*), and (`sorrow`, *sadness*). With `lonely` as the anchor word, `sorrow` can be considered the positive word since both these words belong to the same emotion category *sadness*. The word `playful` is then considered as a negative word since it is annotated with a different emotion category *joy*. This gives rise to the following constraint in the context of *sadness* category: $sim_{sadness}(lonely, sorrow) > sim_{sadness}(lonely, playful) + m_{sadness}$. Such constraints are obtained in the context of all the eight emotion categories by considering each word from the corresponding emotion category as the potential anchor and then generating the positive and negative words.

[1] EmoLex contains emotion and sentiment annotations for 14,182 words. Out of these, it has a total of 4,463 emotion bearing words i.e. words that are marked with at least one emotion category.

3 Retrofitting Method

Our goal is to learn a transformation function that maps pre-trained word embeddings to a vector space that respects the emotion inequality constraints. As explained earlier, an inequality constraint is created using a word triplet (w_a, w_p, w_n) and the corresponding emotion category-specific *margin* m. Thus, a natural way to create training data for this task is to generate a set of four tuples (w_a, w_p, w_n, m) using all possible inequality constraints. However, training data generation using all inequality constraints has a drawback. We first explain this drawback and ways to mitigate it, followed by our retrofitting model to learn the transformation function.

3.1 Training: Batch of triplets To Batch of words

Let's define the set of triplets that satisfy inequality constraints (hence zero loss) as *easy triplets*, and conversely, the set of triplets that do not satisfy the constraints (hence leading to non-zero loss) as *active triplets*. The constraint generation method described in Section 2 produces $O(n^3)$ triplets (n = #words in EmoLex), which by construction leads to training data explosion. Moreover, many of these triplets may trivially satisfy the inequality constraint (i.e., easy triplets) in pre-trained input vector space. In fact, the set of active triplets keeps on changing as the training progress, and just after a few batch updates in stochastic gradient descent, a majority ($> 99\%$) of the triplets become easy triplets resulting in zero loss. The gradients from these inactive triplets start vanishing at this point, leading to considerably slow training.

The stagnant-training problem described above is well studied in the computer vision community, where triplet loss has been successfully applied in metric learning settings for applications such as face verification (Schroff et al., 2015), person re-identification (Hermans et al., 2017), etc. Various approaches for selecting the right set of triplets (referred as triplet mining or sampling) are broadly categorized into offline (Gordo et al., 2016) and online mining (Hermans et al., 2017). In this work, we focus only on online mining as it generally leads to better training convergence than the offline approach. In online mining, we first sample a mini-batch consisting only of raw images (words in our case). The set of active triplets is then generated on-the-fly from the mini-batch. Various policies to sample active triplets from a given batch include BatchHard, BatchHardNegative, and BatchAll. For a given anchor image a from class X, the BatchHard (**BH**) policy selects the hardest positive image p (farthest from a in terms of distance metric) from among the rest of the images of X in the batch. It then selects the hardest negative image n (closest to a in terms of distance metric) from the set of images belonging to classes other than X. The BatchHardNegative (**BHN**) policy relaxes positive image mining by considering all possible in-batch positive images p and then selects the hardest negative n. The BatchAll (**BA**) policy considers all possible in-batch positive images and in-batch negative images for the given anchor a and then selects active triplets from the complete set.

In a nutshell, to mitigate the stagnant-training problem, instead of sampling a mini-batch of triplets from a huge set created offline, we first sample mini-batch of individual words and then generate triplets from the mini-batch on-the-fly using online triplet mining policies.

3.2 Retrofitting model

Our retrofitting model takes pre-trained word embeddings as input and updates them using a non-linear transformation function $\mathbf{T}(\boldsymbol{x_w})$ such that the emotion aspects of words, as induced by the inequality constraints, are respected. The transformation function is learned in a similarity metric learning setting. Figure 1 shows the architecture for learning our retrofitting model.

1. Training data generation: A training instance for our model consists of a word and its emotion category. The data generation component samples words from EmoLex to create a mini-batch b of size n for training. We experiment with two variants: (1) *Uniform* variant samples an equal number of words from all the eight emotion categories; (2) *Weighted* variant, on the other hand, samples words from a category proportional to the number of words annotated with that category in Emolex.

2. Transformation function: We take the d-dimensional pre-trained embeddings of words present in the mini-batch b as input and pass them through the transformation function to compute retrofitted embeddings, i.e., $x_w^t = \mathrm{T}(x_w)$. This function is realized using a multi-layer feed-forward neural network with a corresponding set of network weights W_T.

3. Triplet mining: The retrofitted embeddings

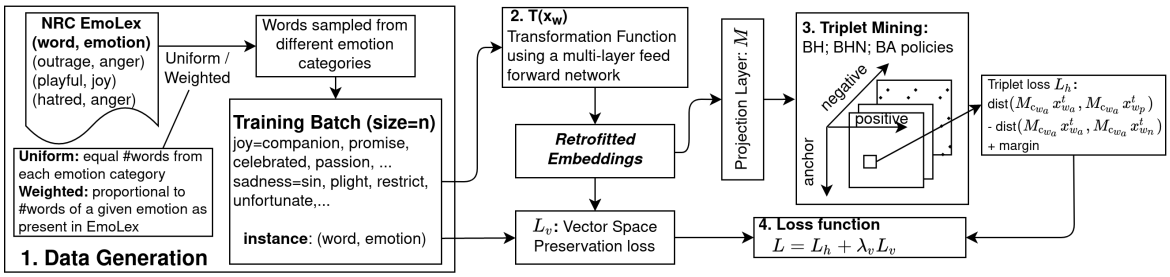

Figure 1: Architecture for our retrofitting model

computed by $\mathrm{T}(x_w)$ are passed to the triplet mining component, which samples the set of active triplets A from batch b according to the selected online triplet mining policy.

4. Loss function: Active triplets obtained from the triplet mining component are used to compute triplet loss from the mini-batch b. It is defined in terms of a margin based hinge loss function,

$$L_h = \sum_{A} \Big(dist(M_{c_{w_a}} x^t_{w_a}, M_{c_{w_a}} x^t_{w_p}) - dist(M_{c_{w_a}} x^t_{w_a}, M_{c_{w_a}} x^t_{w_n}) + margin \Big)_+ \quad (1)$$

Here, $dist$ is a cosine-distance function; $(x)_+ = \max(0, x)$; and $margin$ is a hyper-parameter, set in $[0, 2]$. In Emolex, a word may be tagged with multiple emotion categories e.g. `lonely` is tagged with both *sadness* and *anger*. Thus, while generating *sadness* related constraints from the word `lonely` (e.g. the constraint in $(lonely, sorrow, playful)$), we need a way to extract the sadness aspect. Similarly, when generating *anger* related constraints, we need to consider the anger aspect. To account for this, we first project the retrofitted embeddings to an emotion category-specific vector space using a linear transformation matrix $M_{c_{w_a}} \in \mathbb{R}^{d \times d}; c_{w_a} \in \{1, 2, .., 8\}$ (learned jointly with T). The $dist$ function in Eq. 1 is then applied to the projected retrofitted embeddings.

Vector Space Preservation: Pre-trained embeddings contain useful semantic relations between words as captured by the distributional hypothesis. The transformation function learned by our model should preserve these relations while also respecting the emotion inequality constraints. To address this, we use a regularization term which penalizes vector space transformations that drastically change the topology of input vector space, similar to (Mrkšić et al., 2016; Glavaš and Vulić, 2018). It measures the Euclidean distance between the pre-trained vector x_i and its transformed version $\mathrm{T}(x_i)$ for all words present in batch b,

$$L_v = \sum_{w \in b} \|x_w - \mathrm{T}(x_w)\|_2 \quad (2)$$

The final loss function used by our model is then: $L = L_h + \lambda_v L_v$, where λ_v is a hyper-parameter that determines how strictly the topology of original vector space is preserved. The loss function also includes weight decay for parameters W_T and M.

Since the retrofitting function $\mathrm{T}(x_w)$ is formulated as a representation learning problem (similarity metric learning setting), it can be used to transform pre-trained embeddings of all words present in a given vocabulary post training.

4 Experimental Results

To evaluate our retrofitting method, we experimented with 300-dimensional pre-trained embeddings in GloVe[2] (Pennington et al., 2014) and Word2vec[3] (Mikolov et al., 2013). Due to space constraints, we discuss only GloVe results here (Word2vec results are present in Appendix B). As explained earlier, we used triplet constraints extracted from EmoLex to learn retrofitted embeddings. We refer to our method as **RETriplet** hereafter. Although we report the complete hyper-parameter grid search details in Appendix A, the hyper-parameter λ_v for the vector space preservation loss in Eq. 2 needs special attention. Setting the right value for λ_v is extremely important to learn a meaningful retrofitting model. If we set it very high, RETriplet may not focus on the inequality constraints in triplet loss, thereby learning retrofitted embeddings nearly identical to their pre-trained version. Conversely, a low value of λ_v may produce embeddings that largely satisfy emotion

[2]https://nlp.stanford.edu/data/glove.42B.300d.zip
[3]https://code.google.com/archive/p/word2vec/

constraints but may not preserve the topology of input vector space, possibly leading to degraded performance on downstream tasks. To account for this trade-off, we devise the following scheme and select two configurations: (1) We use adjusted rand index (ARI, a clustering evaluation metric, described in Section 4.1) to measure the quality of retrofitted embeddings and select the configuration that gives the highest value for ARI (referred as **RETripletG**); (2) we compute the average cosine distance between pre-trained and retrofitted embeddings for words in EmoLex and filter configurations having distance < 0.15. We then choose the configuration with the highest ARI from the filtered list (referred as **RETripletGBal**).

Retrofitting approaches proposed in the literature use *attract* and *repel* constraints, extracted from WordNet, Paraphrase database, etc., to update pre-trained embeddings. The attract constraints pull similar (e.g., synonyms, hypernyms, etc.) word pairs close together. While the repel constraints push non-similar (e.g., antonyms) word pairs away from each other. We compare RETriplet with the following,

Counterfit (Mrkšić et al., 2016): It defines the loss function as a weighted sum of terms that brings attract word pairs closer and pushes repel word pairs apart. It also includes a vector space regularization term.

Attract-Repel (AR) (Mrkšić et al., 2017): The counterfit method updates embeddings of attract and repel words without considering their relations to other words. AR addresses this problem by performing context-sensitive vector updates. For each word in attract pairs, it finds the closest (in terms of cosine distance) in-batch word to generate negative examples (conversely farthest for repel words). It then uses these negative examples to form a hinge loss function for context-sensitive updates.

Post-specialization: The methods described above *locally* update vectors of only those words that are present in constraints (i.e., seen words), whereas vectors for all other words remain intact. To address this, post-specialization methods use retrofitted embeddings of seen words to learn a global specialization function which then updates vectors of unseen words. We use the generative adversarial network architecture proposed by Ponti et al. (2018) for post specialization with AR as the local method (referred to as **AR+PS**).

We also learn emotion enriched embeddings using the methods described above by extracting attract and repel constraints from EmoLex. Two words annotated with the same emotion category in EmoLex are added to the attract set, e.g. *(angry, offence)* since both `angry` and `offence` are marked with the *anger* category. In contrast, two words, when annotated with different emotion categories, are added to the repel set, e.g. *(fun, miserable)* since `fun` is marked with *joy* and `miserable` with *sadness*. The generated attract and repel sets are then used to learn retrofitted embeddings. They are referred by appending **+EL** to the retrofitting method, e.g., AR+EL for embeddings retrofitted using AR with EmoLex constraints.

We also compare our method with the following emotion enriched embeddings: (1) **EWE** (Agrawal et al., 2018): It first creates noisy emotion labelled data using distant supervision and then applies recurrent neural network to learn emotion embeddings; (2) **Aff2vec** (Khosla et al., 2018): It appends valence (V), arousal (A) and dominance (D) values of words as present in Warriner's VAD lexicon (Warriner et al., 2013) to the counterfitted GloVe embeddings, resulting in 303-dimensional affective embeddings; (3) **EEArmin** (Seyeditabari et al., 2019): It applies counterfit method directly on the *(word, emotion)* pairs in EmoLex; (4) **SentiEmbs** (Yu et al., 2017): embeddings refined for sentiment using valence values present in Warriner's lexicon.

4.1 Clustering Experiments

Since our main objective is to investigate word embeddings for their emotion content, it is natural to ask, do words that evoke the same emotion have similar embeddings? In other words, are words with similar emotion content clustered together in the vector space? To study this, we extract all words present in EmoLex and their emotion labels to create a dataset for clustering. The embeddings of words are then used as features to perform K-means clustering with the number of means (k) set to 8. Since the true labels are available, we apply various external cluster validity indices to measure clustering quality. In particular, we use adjusted rand index (ARI), Fowlkes Mallows score (FMS), adjusted mutual information score (AdjustedMIS), V-measure, and entropy (refer Scikit-learn user guide). In addition to good cluster quality, retrofitted embeddings shall also preserve the topology of pre-trained vector space. To quantify this, we compute the average cosine distance between

Embeddings	ARI↑	FMS↑	AdjustedMIS↑	V-measure↑	Entropy↓	VDist↓
GloVe	0.0456	0.1542	0.0863	0.0888	1.8092	0
counterfit	0.0897	0.1969	0.1634	0.1657	1.6404	0.1740
AR	0.0749	0.1802	0.1479	0.1502	1.6717	0.0977
AR+PS	0.0853	0.1911	0.1607	0.1630	1.6444	0.1257
counterfit+EL	0.1530	0.2532	0.1953	0.1976	1.5680	0.0308
AR+EL	0.2071	0.3126	0.3966	0.3984	1.1594	0.3068
AR+PS+EL	0.1567	0.2689	0.2579	0.2600	1.4495	0.2029
EWE	0.0556	0.1630	0.1083	0.1108	1.7605	0.0085
Aff2vec	0.0824	0.1877	0.1574	0.1598	1.6517	NA
EEArmin	0.3764	0.4566	0.5501	0.5514	0.7856	1.0152
SentiEmbs	0.0009	0.2974	0.0135	0.0176	1.9817	0.4329
RETripletGBal	0.0951	0.2000	0.1639	0.1662	1.6373	0.0946
RETripletG	0.1616	0.2602	0.3031	0.3050	1.3271	0.4445

Table 3: External cluster validity indices (with k=8) for pre-trained GloVe and its retrofitted versions (↓: lower values are better; ↑: higher values are better) - Overall, RETripletGBal and counterfit+EL provide substantially good clustering while preserving the topology of pre-trained vector space. The embeddings in red are not desirable as they drastically change the pre-trained vector space (high VDist) and may not perform well on affective end-tasks.

pre-trained and retrofitted embeddings for words in EmoLex. It is referred to as VDist (lower values are better).

As shown in Table 3, the scores for the pre-trained GloVe baseline are lowest across all clustering indices. This indicates that there is a scope of improvement for injecting emotion content into pre-trained embeddings. The embeddings from pair-wise retrofitting methods with synonymy and antonymy constraints (i.e., counterfit, AR, AR+PS) reasonably improve clustering quality while maintaining a fairly good VDist (< 0.18). When used with the attract and repel constraints from EmoLex (+EL setting), both AR+EL and AR+PS+EL embeddings achieved extremely good clustering. However, their VDist is very high, pointing to the fact that they did not maintain semantic relations present in GloVe. The EWE embeddings perform poorly on clustering indices as they are identical to their pre-trained version (VDist=0.0085). The SentiEmbs embeddings do not provide good clustering since they are optimized only for coarse-grained sentiments. On the other hand, the EEArmin embeddings have completely overfitted for clustering, with extremely poor VDist. Though Aff2vec embeddings achieve reasonably good clustering, we could not compute VDist due to the three extra dimensions appended for VAD. The embeddings in counterfit+EL and RETripletGBal provide the right balance overall with substantially good cluster quality along with low values for VDist. The counterfit+EL embeddings, however, do not perform well on downstream tasks, as reported later in Table 5.

Figure 2 shows t-SNE plots for EmoLex words using pre-trained GloVe, RETripletGBal, and RETripletG, marking the median point for each emotion category. These points are very close to each other for pre-trained GloVe as it only uses the distributional hypothesis to learn embeddings, not considering the emotion content of words. On the other hand, RETripletG (selected based only on clustering quality) provides extremely good separation but at the expense of losing semantic relations present in GloVe. RETripletGBal embeddings not only provide reasonably good separation but also preserve the topology of the input vector space. The cosine similarity values computed using RETripletGBal are well-calibrated for emotion content, as evident for the exemplar pairs in Table 1.

4.2 Evaluation on Downstream tasks

We evaluate emotion enriched embeddings on two affective end-tasks: (1) Sentiment analysis using binary (SST2) and graded (SST5) Stanford sentiment treebank, and tweet messages from SemEval 2017 (task 4A); (2) Sarcasm detection using sit-com utterances in Mustard++. Table 4 reports statistics for these datasets. Similar to EWE (Agrawal et al., 2018), we use a probing framework (Conneau et al., 2018; Eichler et al., 2019) to evaluate embeddings for their performance on the downstream tasks. In particular, we apply two classification models: sup-

Task	Dataset	#class	size	#token	Type	Vocab	Source
Sentiment analysis	SST2	2	9613	162783	sentence	17630_1	(Socher et al., 2013)
	SST5	5	11855	199120	sentence	19631_1	(Socher et al., 2013)
	SemEval	3	61854	1174626	tweet	23005_2	(Rosenthal et al., 2017)
Sarcasm detection	Mustard++	2	1202	14219	utterance	2632_1	(Ray et al., 2022)

Table 4: Dataset statistics for downstream tasks (subscript in **Vocab** indicate minimum frequency threshold)

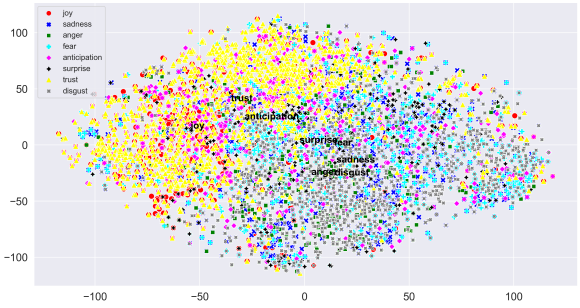

(a) Pre-trained GloVe

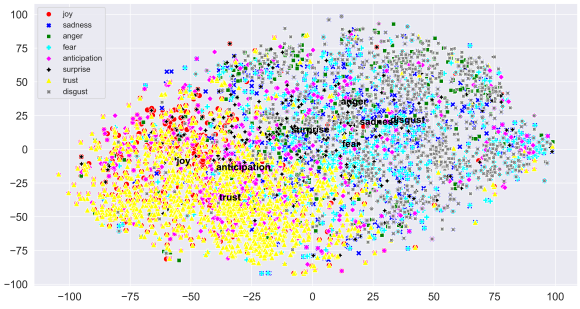

(b) RETripletGBal embeddings

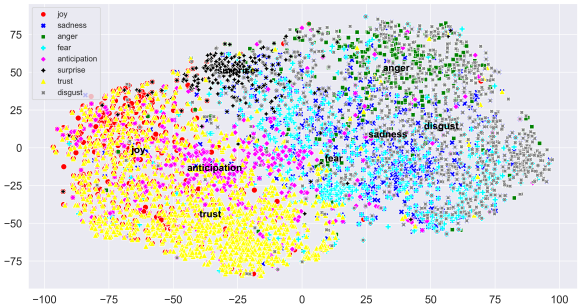

(c) RETripletG embeddings

Figure 2: t-SNE plots for emotion bearing words

port vector machine (SVM), and attention network (AttnNet). The embeddings of tokens present in a given sentence/utterance/tweet are averaged to compute input features for SVM. Whereas the token embeddings as a sequence are passed as input to an attention layer followed by softmax to compute cross-entropy loss for AttnNet.

Table 5 reports the micro F1-scores for SVM and AttnNet. The pre-trained GloVe embeddings seems to be a hard baseline to beat on the sentiment analysis task. While the pair-wise retrofitting methods (counterfit, AR, AR+PS) have been shown to improve tasks such as dialogue state tracking, text simplification, etc., they have not been extensively tested for sentiment analysis. Surprisingly, embeddings from these methods could not beat the baseline even though they are updated to respect the synonymy and antonymy constraints. When retrofitted using attract and repel constraints from EmoLex, their (+EL variants) performance degraded even further. This degradation is partly attributable to the in-batch sampling of negative examples. Unlike synonym constraints where distinct word pairs are not interrelated, word pairs in Emolex attract constraints are interrelated due to their emotion labels. For example, consider in-batch attract pairs such as (enjoy, fun), (happy, thankful), and (loving, delightful), having the common emotion label *joy*. While generating negative examples for `enjoy`, pairs such as (enjoy, happy) and (enjoy, delightful) may be inappropriately considered as candidates, leading to spurious training data. The EWE embeddings trained using distant supervision are nearly identical to their pre-trained version (VDist=0.009), leading to no improvement in end-task. Though retrofitted for emotions, both Aff2vec and EEArmin embeddings could not beat the pre-trained baseline, possibly due to drastic changes to the topology of input vector space (high VDist). SentiEmbs, though optimized for sentiments, unexpectedly could not perform well on any datasets. RETripletGBal embeddings learned using our method achieved the highest F1-score for both SVM and AttnNet on the sentiment analysis task. For the sarcasm detection task (Mus++ in Table 5), the embeddings learned using our method performed better than their pre-trained counterparts (about 1.5% improvement in F1-score) and achieved the highest F1-score with SVM.

4.2.1 Limited data experiments

To evaluate embeddings in a low resource setting, we sample datasets of various sizes, such as 10%,

Embeddings	SVM				AttnNet			
	SST2	SST5†	SemEval†	Mus++	SST2	SST5†	SemEval†	Mus++
GloVe	**0.8034**	0.4122	0.6131	0.5333	0.7705	0.4072	0.6375	0.5125
counterfit	0.7996	**0.4181**	**0.6236**	0.5105	0.7419	0.4005	0.6274	0.5375
AR	0.8029	0.3846	0.5782	0.5063	0.721	0.3937	0.635	0.4833
AR+PS	0.8018	0.4041	0.6031	0.4979	**0.7853**	**0.4204**	0.6306	0.5417
counterfit+EL	0.7985	0.4032	0.6112	0.5021	0.7326	0.3842	**0.6391**	0.5125
AR+EL	0.7902	0.405	0.607	0.4979	0.7348	0.3923	0.6365	0.5042
AR+PS+EL	0.7628	0.3842	0.5711	0.4979	0.7721	0.3624	0.6171	0.4875
EWE	0.7974	0.402	0.6049	0.5523	0.7738	0.4068	0.6237	0.5292
Aff2vec	0.7831	0.3893	0.5725	0.523	0.7381	0.4023	0.6241	**0.5457**
EEArmin	0.7644	0.3805	0.5604	0.5397	0.76	0.3928	0.6226	**0.5458**
SentiEmbs	0.7397	0.3633	0.5511	0.5356	0.6985	0.3543	0.5409	0.5125
RETripletGBal	**0.816**	**0.4339**	**0.6305**	**0.5542**	**0.7946**	**0.4267**	**0.6405**	0.5292
RETripletG	0.7705	0.3946	0.6101	**0.5667**	0.7787	0.3973	0.6288	0.4792

Table 5: Micro F1-scores for SVM and AttnNet with various embeddings as input (**Bold+Underline**: highest; **Bold**: next highest); †: Wilcoxon's signed rank test with $\alpha = 0.5$ indicates RETripletGBal is better than GloVe

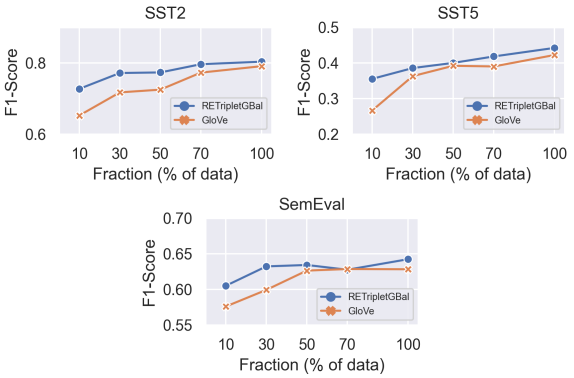

Figure 3: Data size vs. micro F1-score for Pre-trained GloVe and RETripletGBal in limited data setting

30%, etc., from the original sentiment analysis datasets. We then compare pre-trained GloVe with RETripletGBal in terms of micro F1-score across the data sizes. As we can see in Figure 3, RETripletGBal performs significantly better than pre-trained GloVe in a low data regime ($< 60\%$ data). The difference in performance reduces nearly after 80% data size. This points to the fact that the external knowledge from EmoLex as captured by our retrofitting method helps improve the end-task, especially in the limited data scenario.

5 Related Work

Large language models with contextualized word embeddings (e.g., BERT and its variants) have lately received a lot of attention in the NLP community. Nonetheless, their static counterparts are still actively explored, e.g., combining static and contextualized embeddings to improve end-tasks (Alharbi and Lee, 2021; Alghanmi et al., 2020), inducing knowledge bases (Dufter et al., 2021), bilingual lexicon induction (Zhang et al., 2021), etc. In this work, we focus on static word embeddings that are learned primarily using the distributional hypothesis. A major limitation with these embeddings is that they do not differentiate semantic similarity from other types of relatedness (Hill et al., 2015). This problem is addressed by borrowing semantic relations from resources such as WordNet, Paraphrase Database, etc., in the form of constraints. These constraints are then used by joint specialization (Yu and Dredze, 2014; Liu et al., 2015) or retrofitting models (Faruqui et al., 2015; Mrkšić et al., 2016; Shah et al., 2020) to improve word embeddings. These models, however, focus mainly on synonymy, antonymy, and hypernymy relations. Recently, a few attempts, such as Aff2vec from Khosla et al. (2018) and emotion embeddings from Seyeditabari et al. (2019), incorporate knowledge present in affective lexicons to learn emotion enriched embeddings.

The contrastive learning approach similar to our work has recently been applied to learn transformer based sentence embeddings in SBERT (Reimers and Gurevych, 2019) and zero-shot image classification in CLIP (Radford et al., 2021). However, these methods are not specialized to learning emotion enriched embeddings.

There is a large body of work that focuses on

learning task-specific *affective* embeddings. These methods first use distant supervision to create a noisy labelled dataset and then use it to update word embeddings or learn them from scratch. For instance, sentiment-aware embeddings using tweet data (Tang et al., 2014, 2016); affective embeddings using tweet emojis (Felbo et al., 2017); emotion enriched embeddings using product reviews data (Agrawal et al., 2018). Since embeddings learned from these methods are tied to the dataset used for distant supervision, they may not work well for other related affective end-tasks. Moreover, they are not very accurate due to noisy labelling.

The emotion-enriched embeddings learned by our method are not only accurate compared to the methods described above, as evident from the clustering experiments, they also work well on the related affective end-tasks.

6 Summary and Future work

We present a novel retrofitting method to learn emotion enriched embeddings using triplet constraints from EmoLex. These constraints are used as training data to learn a retrofitting function in a similarity metric learning setting. The embeddings learned by our method perform better than their pre-trained counterparts and other benchmarks in both intrinsic clustering evaluation and the extrinsic downstream tasks in sentiment analysis and sarcasm detection. As future work, we plan to extend our triplet constraint-based approach to other resources such as VAD lexicon (Warriner et al., 2013; Mohammad, 2018a). We also plan to develop a similar approach for contextualized word embeddings.

References

Ameeta Agrawal, Aijun An, and Manos Papagelis. 2018. Learning emotion-enriched word representations. In *Proceedings of the 27th International Conference on Computational Linguistics*, pages 950–961, Santa Fe, New Mexico, USA. Association for Computational Linguistics.

Israa Alghanmi, Luis Espinosa Anke, and Steven Schockaert. 2020. Combining BERT with static word embeddings for categorizing social media. In *Proceedings of the Sixth Workshop on Noisy User-generated Text (W-NUT 2020)*, pages 28–33, Online. Association for Computational Linguistics.

Abdullah I. Alharbi and Mark Lee. 2021. Multi-task learning using a combination of contextualised and static word embeddings for Arabic sarcasm detection and sentiment analysis. In *Proceedings of the Sixth Arabic Natural Language Processing Workshop*, pages 318–322, Kyiv, Ukraine (Virtual). Association for Computational Linguistics.

Alexis Conneau, German Kruszewski, Guillaume Lample, Loïc Barrault, and Marco Baroni. 2018. What you can cram into a single $&!#* vector: Probing sentence embeddings for linguistic properties. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 2126–2136, Melbourne, Australia. Association for Computational Linguistics.

Philipp Dufter, Nora Kassner, and Hinrich Schütze. 2021. Static embeddings as efficient knowledge bases? In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 2353–2363, Online. Association for Computational Linguistics.

Max Eichler, Gözde Gül Şahin, and Iryna Gurevych. 2019. LINSPECTOR WEB: A multilingual probing suite for word representations. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP): System Demonstrations*, pages 127–132, Hong Kong, China. Association for Computational Linguistics.

Paul Ekman. 1992. An argument for basic emotions. *Cognition and Emotion*, 6:169–200.

Manaal Faruqui, Jesse Dodge, Sujay Kumar Jauhar, Chris Dyer, Eduard Hovy, and Noah A. Smith. 2015. Retrofitting word vectors to semantic lexicons. In *Proceedings of the 2015 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 1606–1615, Denver, Colorado. Association for Computational Linguistics.

Bjarke Felbo, Alan Mislove, Anders Søgaard, Iyad Rahwan, and Sune Lehmann. 2017. Using millions of emoji occurrences to learn any-domain representations for detecting sentiment, emotion and sarcasm. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 1615–1625, Copenhagen, Denmark. Association for Computational Linguistics.

Goran Glavaš and Ivan Vulić. 2018. Explicit retrofitting of distributional word vectors. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 34–45, Melbourne, Australia. Association for Computational Linguistics.

Albert Gordo, Jon Almazán, Jérôme Revaud, and Diane Larlus. 2016. Deep image retrieval: Learning

global representations for image search. In *Computer Vision - ECCV 2016 - 14th European Conference, Amsterdam, The Netherlands, October 11-14, 2016, Proceedings, Part VI*, volume 9910 of *Lecture Notes in Computer Science*, pages 241–257. Springer.

Zellig Harris. 1954. Distributional structure. *Word*, 10(23):146–162.

Alexander Hermans, Lucas Beyer, and Bastian Leibe. 2017. In Defense of the Triplet Loss for Person Re-Identification. *arXiv preprint arXiv:1703.07737*, abs/1703.07737.

Felix Hill, Roi Reichart, and Anna Korhonen. 2015. SimLex-999: Evaluating semantic models with (genuine) similarity estimation. *Computational Linguistics*, 41(4):665–695.

Sopan Khosla, Niyati Chhaya, and Kushal Chawla. 2018. Aff2Vec: Affect–enriched distributional word representations. In *Proceedings of the 27th International Conference on Computational Linguistics*, pages 2204–2218, Santa Fe, New Mexico, USA. Association for Computational Linguistics.

Diederik P. Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. Cite arxiv:1412.6980Comment: Published as a conference paper at the 3rd International Conference for Learning Representations, San Diego, 2015.

Quan Liu, Hui Jiang, Si Wei, Zhen-Hua Ling, and Yu Hu. 2015. Learning semantic word embeddings based on ordinal knowledge constraints. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 1501–1511, Beijing, China. Association for Computational Linguistics.

Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013. Distributed representations of words and phrases and their compositionality. In *Proceedings of the 26th International Conference on Neural Information Processing Systems - Volume 2*, NIPS'13, pages 3111–3119, USA. Curran Associates Inc.

Saif M. Mohammad. 2018a. Obtaining reliable human ratings of valence, arousal, and dominance for 20,000 english words. In *Proceedings of The Annual Conference of the Association for Computational Linguistics (ACL)*, Melbourne, Australia.

Saif M. Mohammad. 2018b. Word affect intensities. In *Proceedings of the 11th Edition of the Language Resources and Evaluation Conference (LREC-2018)*, Miyazaki, Japan.

Saif M. Mohammad and Peter D. Turney. 2013. Crowdsourcing a word-emotion association lexicon. *Computational Intelligence*, 29(3):436–465.

Nikola Mrkšić, Diarmuid Ó Séaghdha, Blaise Thomson, Milica Gašić, Lina M. Rojas-Barahona, Pei-Hao Su, David Vandyke, Tsung-Hsien Wen, and Steve Young. 2016. Counter-fitting word vectors to linguistic constraints. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 142–148, San Diego, California. Association for Computational Linguistics.

Nikola Mrkšić, Ivan Vulić, Diarmuid Ó Séaghdha, Ira Leviant, Roi Reichart, Milica Gašić, Anna Korhonen, and Steve Young. 2017. Semantic specialisation of distributional word vector spaces using monolingual and cross-lingual constraints. *Transactions of the Association for Computational Linguistics*, 5:309–324.

W. Gerrod Parrott. 2001. Emotions in social psychology : essential readings.

Jeffrey Pennington, Richard Socher, and Christopher Manning. 2014. Glove: Global vectors for word representation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1532–1543. Association for Computational Linguistics.

Robert Plutchik. 1980. A general psychoevolutionary theory of emotion. *Theories of emotion*.

Edoardo Maria Ponti, Ivan Vulić, Goran Glavaš, Nikola Mrkšić, and Anna Korhonen. 2018. Adversarial propagation and zero-shot cross-lingual transfer of word vector specialization. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 282–293, Brussels, Belgium. Association for Computational Linguistics.

Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, Gretchen Krueger, and Ilya Sutskever. 2021. Learning transferable visual models from natural language supervision. In *Proceedings of the 38th International Conference on Machine Learning, ICML 2021, 18-24 July 2021, Virtual Event*, volume 139 of *Proceedings of Machine Learning Research*, pages 8748–8763. PMLR.

Anupama Ray, Apoorva Nunna, and Pushpak Bhattacharyya. 2022. A multimodal corpus for emotion recognition in sarcasm. In *Proceedings of the 13th Edition of the Language Resources and Evaluation Conference (LREC-2022)*, Marseille, France.

Nils Reimers and Iryna Gurevych. 2019. Sentence-BERT: Sentence embeddings using Siamese BERT-networks. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 3982–3992, Hong Kong, China. Association for Computational Linguistics.

Sara Rosenthal, Noura Farra, and Preslav Nakov. 2017. SemEval-2017 task 4: Sentiment analysis in Twitter. In *Proceedings of the 11th International Workshop on Semantic Evaluation (SemEval-2017)*, pages 502–518, Vancouver, Canada. Association for Computational Linguistics.

Florian Schroff, Dmitry Kalenichenko, and James Philbin. 2015. Facenet: A unified embedding for face recognition and clustering. In *IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2015, Boston, MA, USA, June 7-12, 2015*, pages 815–823. IEEE Computer Society.

Scikit-learn user guide. Clustering performance evaluation. Online; accessed 01-February-2022.

Armin Seyeditabari, Narges Tabari, Shafie Gholizadeh, and Wlodek Zadrozny. 2019. Emotional embeddings: Refining word embeddings to capture emotional content of words. *ArXiv*, abs/1906.00112.

Sapan Shah, Sreedhar Reddy, and Pushpak Bhattacharyya. 2020. A retrofitting model for incorporating semantic relations into word embeddings. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 1292–1298, Barcelona, Spain (Online). International Committee on Computational Linguistics.

Richard Socher, Alex Perelygin, Jean Wu, Jason Chuang, Christopher D Manning, Andrew Ng, and Christopher Potts. 2013. Recursive deep models for semantic compositionality over a sentiment treebank. In *Proceedings of the 2013 conference on empirical methods in natural language processing*, pages 1631–1642.

Duyu Tang, Furu Wei, Bing Qin, Nan Yang, Ting Liu, and Ming Zhou. 2016. Sentiment embeddings with applications to sentiment analysis. *IEEE Transactions on Knowledge and Data Engineering*, 28(2):496–509.

Duyu Tang, Furu Wei, Nan Yang, Ming Zhou, Ting Liu, and Bing Qin. 2014. Learning sentiment-specific word embedding for Twitter sentiment classification. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1555–1565, Baltimore, Maryland. Association for Computational Linguistics.

Amy Beth Warriner, Victor Kuperman, and Marc Brysbaert. 2013. Norms of valence, arousal, and dominance for 13,915 english lemmas. *Behavior Research Methods*, 45:1191–1207.

Liang-Chih Yu, Jin Wang, K. Robert Lai, and Xuejie Zhang. 2017. Refining word embeddings for sentiment analysis. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 534–539, Copenhagen, Denmark. Association for Computational Linguistics.

Mo Yu and Mark Dredze. 2014. Improving lexical embeddings with semantic knowledge. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 545–550, Baltimore, Maryland. Association for Computational Linguistics.

Jinpeng Zhang, Baijun Ji, Nini Xiao, Xiangyu Duan, Min Zhang, Yangbin Shi, and Weihua Luo. 2021. Combining static word embeddings and contextual representations for bilingual lexicon induction. In *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*, pages 2943–2955, Online. Association for Computational Linguistics.

A Training details

This section details the hyper-parameters in our retrofitting method and the best combinations selected thereof. The transformation function $\mathrm{T}(x_w)$ in RETriplet is implemented using a multi-layer feed-forward neural network. The hyper-parameters are:- number of hidden layers: $\{2, 3, 4\}$, size of hidden layer: $\{300, 400, 500\}$, activations: $\{tanh, ReLU\}$, dropout: 0.2 and L2 regularization: 0.0005. We use Adam (Kingma and Ba, 2014) optimization algorithm with, learning rate: $\{0.001, 0.0005\}$ and batch size: $\{64, 128, 256\}$. We experiment with two data generation schemes described earlier: Uniform and Weighted. For the hinge loss function in Eq. 1, we use cosine as the distance metric with two margin values, i.e. $\{0.2, 0.6\}$. The margin is set to the same value for all emotion categories. For triplet mining, during initial experiments, we observed that both the BatchHard and BatchAll mining policies performed equally well, with BatchAll having better convergence during initial epochs. Hence, we primarily experimented with the BatchAll mining policy. The hyper-parameter for vector space preservation loss λ_v is varied as $\{0.1, 0.2, 0.3, 0.4, 0.5, 0.6, 0.7, 0.8, 0.9, 1, 1.5\}$. We set aside 10% words in EmoLex for validation and use early stopping with patience 10. For experimentation, we used CPU machines with 64GB RAM and 20 core CPUs. Each configuration on an average took 80 minutes to run.

For both GloVe and Word2vec, we select two configurations to generate retrofitted embeddings. One configuration is selected only on the basis of clustering quality metric (ARI). Whereas, the second configuration takes vector space preservation into account in addition to the clustering quality. Table 6 reports these configurations.

	GloVe		Word2vec	
hyperparameter	RETripletGBal	RETripletG	RETripletWBal	RETripletW
#layers	2	3	2	3
#hidden units	300	300	300	200
activation	ReLU	ReLU	ReLU	ReLU
dropout	0.2	0.2	0.2	0.2
L2-regularization	0.0005	0.0005	0.0005	0.0005
batch-size	128	128	128	128
learning rate	0.0005	0.0005	0.0005	0.0005
data generation	Weighted	Uniform	Weighted	Weighted
triple mining	BatchAll	BatchAll	BatchAll	BatchAll
λ_v	0.5	0.1	0.7	0.4

Table 6: Selected hyper-parameter configurations for retrofitted embeddings (1) GloVe:- RETripletG has the best ARI; RETripletGBal has the best ARI with VDist < 0.15 (2) Word2vec:- RETripletW has the best ARI; RETripletWBal has the best ARI with VDist < 0.15

B Experimental results for Word2vec

Table 7 reports clustering experiments for Word2vec pre-trained baseline and their retrofitted versions. Table 8 reports results for sentiment analysis and sarcasm detection tasks for SVM and Attention network with Word2vec as the base embeddings.

Embeddings	ARI↑	FMS↑	AdjustedMIS↑	V-measure↑	Entropy↓	VDist↓
Word2vec	0.0553	0.1641	0.1019	0.1044	1.7753	0.0
counterfit	0.0762	0.1814	0.1495	0.1518	1.6682	0.1803
AR	0.0794	0.186	0.1538	0.1561	1.6601	0.2556
AR+PS	0.0913	0.2051	0.159	0.1613	1.6559	0.1326
counterfit+EL	0.1628	0.2618	0.2029	0.2051	1.5507	0.0232
AR+EL	0.2008	0.3066	0.39	0.3917	1.1729	0.37
AR+PS+EL	0.1228	0.2527	0.3349	0.3369	1.3078	0.1931
EWE	-	-	-	-	-	-
Aff2vec	0.0914	0.1978	0.1567	0.1591	1.6548	NA
EEArmin	0.3655	0.4468	0.5495	0.5507	0.7964	0.9986
SentiEmbs	0.0007	0.3000	0.0085	0.0126	1.9896	0.4382
RETripletWBal	0.1493	0.2545	0.2086	0.2109	1.5448	0.1371
RETripletW	0.1768	0.2764	0.2784	0.2804	1.3885	0.3633

Table 7: External cluster validity indices for pre-trained Word2vec and its retrofitted versions (↓: lower values are better; ↑: higher values are better) - Overall, RETripletWBal and counterfit+EL provide substantially good clustering while preserving the topology of pre-trained vector space. The embeddings in red are not desirable as they drastically change the pre-trained vector space (high VDist) and may not perform well on affective end-tasks. *EWE embeddings not available for Word2vec.

Embeddings	SVM				AttnNet			
	SST2	SST5	SemEval	Mus++	SST2	SST5	SemEval	Mus++
Word2vec	0.8144	0.4262	0.6209	0.5481	**0.7985**	0.4136	**0.6342**	0.525
counterfit	0.8127	**0.4281**	**0.6298**	0.5063	0.7408	0.4023	0.6277	0.5125
AR	0.8018	0.409	0.5995	0.5105	0.7842	0.3787	0.6307	0.5083
AR+PS	0.8023	0.4176	0.5995	0.5397	0.7924	**0.4249**	0.6281	0.5667
counterfit+EL	0.816	0.4262	0.6245	0.5272	0.7567	0.3778	**0.6385**	0.5458
AR+EL	0.8127	0.4208	0.6243	0.5314	0.7776	0.3697	0.6306	0.5375
AR+EL+PS	0.7968	0.3959	0.6012	0.523	0.7452	0.4072	0.6186	0.5125
EWE	-	-	-	-	-	-	-	-
Aff2vec	**0.8166**	0.407	0.6119	0.5146	0.7414	0.3692	0.6299	**0.575**
EEArmin	0.771	0.3887	0.5964	**0.5523**	0.7479	0.3566	0.6197	0.5542
SentiEmbs	0.7567	0.3656	0.5716	**0.5649**	0.7205	0.3661	0.5462	0.4958
RETripletWBal	**0.8221**	**0.438**	**0.6323**	**0.5523**	**0.8051**	**0.419**	0.6323	**0.5792**
RETripletW	0.7979	0.4145	0.6153	0.5105	0.7979	0.3982	0.6307	0.5

Table 8: Micro F1-scores for SVM and AttnNet with various embeddings as input: Experiments with Word2vec as baseline (**Bold+Underline**: highest; **Bold**: next highest) *EWE embeddings not available for Word2vec

Metaphor Detection via Linguistics Enhanced Siamese Network

Shenglong Zhang **Ying Liu**
Tsinghua University, Beijing, China, 100084
zsl18@mails.tsinghua.edu.cn
yingliu@mail.tsinghua.edu.cn

Abstract

In this paper we present MisNet, a novel model for word level metaphor detection. MisNet converts two linguistic rules, i.e., Metaphor Identification Procedure (MIP) and Selectional Preference Violation (SPV) into semantic matching tasks. MIP module computes the similarity between the contextual meaning and the basic meaning of a target word. SPV module perceives the incongruity between target words and their contexts. To better represent basic meanings, MisNet utilizes dictionary resources. Empirical results indicate that MisNet achieves competitive performance on several datasets.

1 Introduction

Metaphor is an omnipresent figurative language in daily communication. Conceptual Metaphor Theory proposes that metaphor is a mapping mechanism between the source domain and the target domain (Lakoff and Johnson, 2008).

e.g. 1 *The scream **pierced** the night.*

In e.g. 1, the literal meaning of the verb ***pierce*** is "some sharp object goes into or on through something" . However, the contextual meaning is "break silence". Here, the source domain is a highly abstract action, while the target domain can present the corresponding meaning in a more concrete way. In general, there exists two types of metaphors, i.e., novel metaphors and conventional metaphors.

e.g. 2 *He **attacked** the government's defence policy.*

Semantic Shift Theory shows that new lexical senses can derive from metaphors (Blank, 2013). Once a metaphorical usage of a word is accepted by most people, the metaphorical lexical sense is fixed. Thus a polysemant may have metaphorical marginal meanings (Bloomfield, 1994). In e.g. 2, the metaphorical target word ***attack*** means ***criticize***, which is also a sense of it. It is a conventional metaphor for the metaphorical meaning has been fixed. While e.g. 1 is a novel metaphor, for the metaphorical usage is temporary.

Linguistic rules instruct us how to identify metaphors. According to Metaphor Identification Procedure (MIP) (Crisp et al., 2007; Steen, 2010), a metaphor is identified if the contextual meaning of the target word contrasts with one of its more basic meaning. More basic meanings are: **1**) More concrete; what they evoke is easier to imagine, see, hear, feel, smell, and taste; **2**) related to bodily action; **3**) more precise (as opposed to vague); **4**) historically older(Group, 2007; Do Dinh et al., 2018). The basic meaning of ***pierce*** in e.g. 1 contrasts with its contextual meaning so it is a metaphor.

Researchers are divided on how to represent basic meanings. Gao et al. (2018) and Mao et al. (2019) used dynamic ELMo embeddings and static GloVe embeddings to encode contextual target meanings and basic target meanings respectively. Choi et al. (2021) proposed that a target used alone is literal. Su et al. (2021) and Wan et al. (2021) used the gloss (brief definition) of a target to represent its literal meaning. All the methods are not linguistically intuitive, because we do not know whether the basic meaning is accurately encoded through embedding and a basic meaning is not simply the average of its gloss. Since basic meanings are not properly represented, MIP may be invalid. Consequently, conventional metaphors are ignored by previous studies as well(Tong et al., 2021).

Another linguistic rule is Selectional Preference Violation (SPV) (Wilks, 1975, 1978), which suggests that a metaphor is identified by noticing semantic incongruity between a target word and its context. In e.g. 1, ***pierce*** rarely occurs in the context consisting of ***scream*** and ***night***. There comes a contextual contrast in such a collocation. Notice that SPV becomes invalid when faced with a conventional metaphor, because the context is also usual for the metaphor word like ***attack*** in e.g. 2.

To better use linguistic rules, we propose to use

Proceedings of the 29th International Conference on Computational Linguistics, pages 4149–4159
October 12–17, 2022.

the sentence where the target adopts its basic meaning for a better representation. This idea is in line with mainstream language models: you shall know a word by the company it keeps (Firth, 1957).

In this paper, we propose a novel metaphor detection model named **M**etaphor **I**dentification from **S**iamese **Net**work (MisNet). MisNet adopts a siamese framework, consisting of two separate encoders . We regard MIP as a representation based semantic matching task between target in the given sentence and target in the basic usage. MIP is accomplished across two encoders. We model SPV as an interaction based semantic matching task between the target word and its context to measure semantic incongruity. SPV is implemented within a single encoder. Based on the fusion of MIP and SPV modules, MisNet makes a final prediction.

The contributions of this paper can be summarized as follows:

- We model two linguistic rules, i.e., MIP and SPV as two semantic matching tasks. Our model is linguistically intuitive and also extensible.
- We use basic usage to better encode the basic meaning for a target word, thus our model can avoid the invalidation of MIP and SPV. It is also proficient in tackling conventional metaphors.
- Experimental results show that our method achieves competitive performance on several datasets over existing approaches.
- Our code is available on GitHub[1].

2 Related Work

Recently, metaphor detection has attracted lots of attention. With the development of NLP technologies, various methods have been applied. In general, these approaches can be categorized into three types: feature engineering based, RNN based and transformer based methods.

Feature engineering based methods use linguistic features such as word concreteness, word abstractness (Turney et al., 2011), and word class etc., as input of a certain machine learning model like Logistic Regression and SVM (Shutova and Sun, 2013; Assaf et al., 2013; Tsvetkov et al., 2014; Wan et al., 2020). RNN based models use RNN as a feature extractor to form contextual representations to identify metaphors(Wu et al., 2018; Gao et al., 2018; Mao et al., 2019; Le et al., 2020). Though great improvements have been made, RNN based models mostly use static word representations like Word2Vec and GloVe, so they are not adept at metaphors which convey complex contextual senses. Also, due to the nature of RNN, these models cannot be paralleled.

Transformer based methods use a Pretrained Language Model (PLM) like BERT (Devlin et al., 2019) and RoBERTa (Liu et al., 2019), as the backbone of the model, yielding promising results in metaphor detection. Su et al. (2020) converted metaphor detection into a machine reading comprehension task with various linguistic features incorporated, achieving best reported results in ACL 2020 metaphor detection shared task. Choi et al. (2021) used a late-interaction mechanism to encode the contextual meaning and the literal meaning of a target word. Song et al. (2021) focused on verb metaphor detection. They used dependency parsing to extract the objects and subjects of the given verbs to use syntactic relations. Lin et al. (2021) utilized contrastive learning to distinguish metaphors from literal usages. They also used self-training strategy to generate pseudo-labels, which largely expanded existing public datasets. Also, some recent researches noticed that external dictionary resources could greatly help metaphor detection. Wan et al. (2021) and Su et al. (2021) used glosses to interpret target words. They took the average embeddings of a gloss as the corresponding target representation. However, the meaning of a word is not simply the average of its definition. There still leaves much space to better represent literal meanings.

3 Proposed Model

Some researchers use sequence labeling to detect metaphors, i.e., label all n words in a sentence in one go (Gao et al., 2018; Mao et al., 2019). In this paper, we use word classification paradigm to detect metaphors. We regard each word in the given sentence as target in order, then take a target along with its given sentence to predict the metaphoricity of the target for n times.

3.1 SPV & MIP : Semantic Matching

Semantic matching intends to measure the similarity between two given texts, of which Interaction-

[1]https://github.com/SilasTHU/MisNet

based Models and Representation-based Models are two main paradigms. In this paper, we use the mentioned two semantic matching models to implement SPV and MIP.

Interaction-based Model for SPV: for an Interaction-based Model (IM), two texts are concatenated as input, where each token within the input can fully interact with the others (Yang et al., 2019; Rao et al., 2019). Vanilla BERT (Devlin et al., 2019) and RoBERTa (Liu et al., 2019) take two texts as model input to compute a similarity score, so they are also IMs. SPV suggests to notice the incongruity between the target word and its context, which can be measured through the semantic similarity between them. As Fig. 1 (a) shows, we adopt an IM to implement SPV, because the target and its context are from a same sequence, such that they are naturally concatenated. In our model, they are two texts to be matched. Hence they can interact thoroughly through multi-head self-attention (Vaswani et al., 2017) in BERT. Finally, we manage to retrieve the contextual target embedding $\boldsymbol{h}_t$ and context embedding $\boldsymbol{h}_c$ to calculate the similarity.

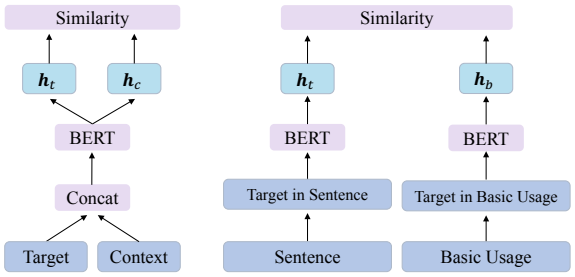

(a) Interaction-based Model (b) Representation-based Model

Figure 1: Interaction-based Model and Representation-based Model in semantic matching task.

Representation-based Model for MIP: for a Representation-based Model (RM), two texts are input into different encoders to get their representations, so the two texts do not interact with each other during encoding (Conneau et al., 2017; Reimers and Gurevych, 2019). MIP prompts us to determine whether the target has a more basic usage. Thus we calculate the semantic similarity between target in the given sentence and that in the basic usage. As Fig. 1 (b) shows, we model MIP as an RM because using two separate encoders for the given sentence and the basic usage can avoid unnecessary interactions. Hence the contextual meaning and the basic meaning of a target can be better captured. We then obtain the contextual target embedding $\boldsymbol{h}_t$ and the basic embedding $\boldsymbol{h}_b$, based on which the similarity is computed.

Attack *Step 0*

NOUN
1. An aggressive and violent act.
eg. *He was killed in an arson attack.*
2. Public criticism or opposition.
eg. *The Opposition Leader intensified his attack on the Prime Minister.*

VERB *Step 1*
1. Take aggressive military action. *Step 2*
eg. *In February the Germans attacked Verdun.*
2. Criticize or oppose publicly.
eg. *He attacked her opinion.*

Figure 2: Basic Usage Retrieving Strategy for *attack (verb)*. **Step 0.** Find the term of the target word. **Step 1.** Locate at the same POS tag. **Step 2.** Take the example sentence under the first gloss as a basic usage since dictionary editors tend to place more basic meanings in the front.

3.2 MisNet Architecture

Combine MIP and SPV: Using SPV to detect a metaphor depends on the incongruity of the target and its context. However, a conventional metaphorical target does not own a paradoxical context (the context is usually common for the target), so SPV may be invalid (Haagsma and Bjerva, 2016; Do Dinh et al., 2018). MIP is utilized based on the basic meaning and the contextualized meaning of a target. Since we use basic usages to encode basic meanings, our MIP module is suitable for both conventional and novel metaphors. We leverage the combination of MIP and SPV for better metaphor detection, which is proved to be a better method by experimental results.

Fig. 3 shows the architecture of MisNet. MisNet adopts a siamese framework to combine MIP and SPV. The left part encodes the given sentence, while the right uses the target, the POS tag, and the basic usage. MIP is implemented across the left and the right encoders, while SPV functions within the left one.

Left Encoder Input: the left encoder input is the given sentence in the dataset:

$$\mathbf{L} = (\text{[CLS]}, \text{given\_sentence}, \text{[SEP]}), \quad (1)$$

where [CLS] and [SEP] are the two special tokens of BERT.

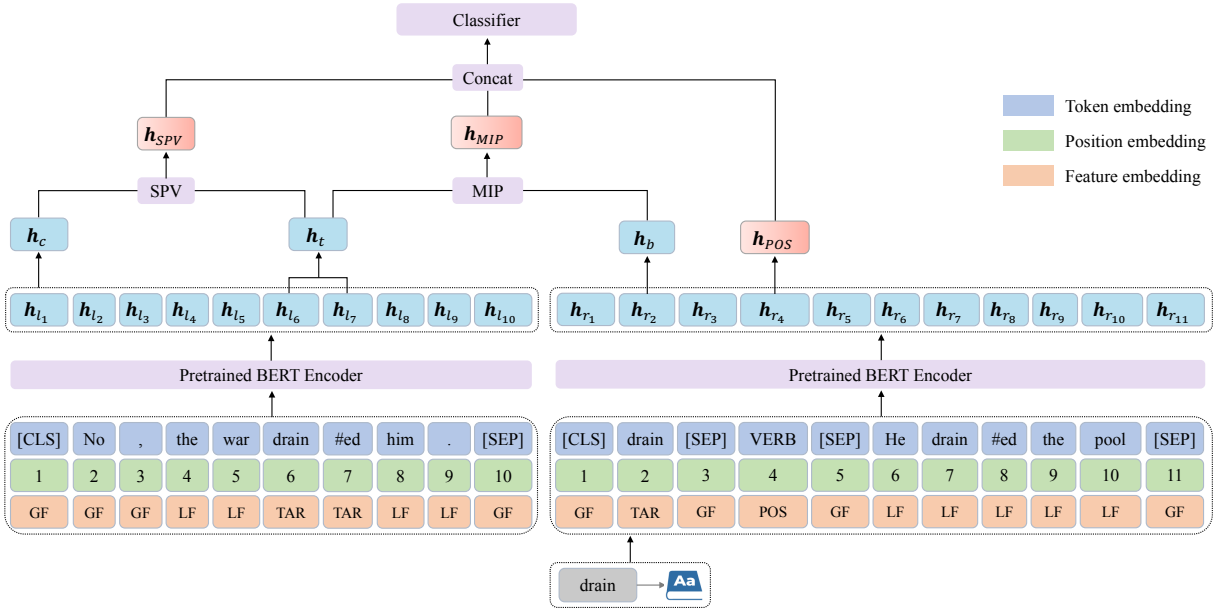

Figure 3: MisNet architecture. The two BERT encoders share weights. $\boldsymbol{h}_c$, $\boldsymbol{h}_t$, $\boldsymbol{h}_b$ are context embedding, contextual target meaning, and basic meaning respectively. GF, LF, POS, TAR denote global feature, local feature, POS feature and target word.

Right Encoder Input: we concatenate the target word, target POS tag and the basic usage as the right encoder input. The basic direct usage (or a more basic usage, which is still in line with MIP rule.) for the target word is retrieved via Basic Usage Retrieving Strategy as Fig. 2 shows. The right input is:

$$\mathbf{R} = (\text{[CLS]}, \text{target\_word}, \text{[SEP]}, \text{POS}, \text{[SEP]}, \text{basic\_usage}, \text{[SEP]}). \quad (2)$$

If we fail to retrieve the basic usage, we just use the target word and its POS tag.

Different parts of the input have different impacts on metaphor detection. Self-Attention mechanism in BERT can benefit semantic representations for input tokens(Vaswani et al., 2017), but it may not be sufficient to notice the differences among various input parts. To treat them differently, for both the left and the right input, we add input type feature embeddings to the BERT input layer. We design four features and embed them into fixed-length vectors:

- **POS Feature**: the POS tag of the target word. It only exists in the right input.
- **Target Feature**: the target word. The left and the right input have a same target word.
- **Local Feature**: following Su et al. (2020) and Choi et al. (2021), we set the clause where the target word lies as local context. For simplicity, a clause is separated by commas, dots, exclamation marks, and question marks etc. Since a basic usage is usually short[2], we regard the whole basic usage as local feature.
- **Global Feature**: other tokens except the POS tag, the target word, and its local feature.

After tokenization via Byte-Pair Encoding (BPE) algorithm (Radford et al., 2019), $\mathbf{L}$ is cut into n tokens, while $\mathbf{R}$ has m tokens. The final input for BERT is token embeddings, positional embeddings, plus feature embeddings. Then we use BERT to get contextualized representations:

$$\mathbf{H}_L = \text{BERT}(\mathbf{L}) = (\boldsymbol{h}_{l_1}, \boldsymbol{h}_{l_2}, \cdots, \boldsymbol{h}_{l_n}), \quad (3)$$
$$\mathbf{H}_R = \text{BERT}(\mathbf{R}) = (\boldsymbol{h}_{r_1}, \boldsymbol{h}_{r_2}, \cdots, \boldsymbol{h}_{r_m}), \quad (4)$$

where $\mathbf{H}_L \in \mathbb{R}^{n \times d}$ and $\mathbf{H}_R \in \mathbb{R}^{m \times d}$ are the embedding matrices of $\mathbf{L}$ and $\mathbf{R}$ respectively. d is the hidden dimension in BERT.

Based on $\mathbf{H}_L$, we can get the contextual meaning of the target word, which is denoted by $\boldsymbol{h}_t$. If the target word is cut into k tokens by BPE, we just take the average:

$$\boldsymbol{h}_t = \frac{1}{k} \sum_{i=u}^{u+k-1} \boldsymbol{h}_{l_i}, \quad (5)$$

[2]It is difficult to get the exact position for a target word in its basic usage, because it may not be rendered in the original form and lemmatization is not always accurate.

where u is the start location for target in the left input. Similarly, based on $\mathbf{H}_R$, we get the basic meaning of the target word, which is denoted by $\boldsymbol{h}_b$. Notice that we do not need to know the exact position of the target word in the basic usage, because transformer encoder will apply self-attention mechanism to make the target word in $\mathbf{R}$ focus on the relevant parts automatically (Vaswani et al., 2017).

For the left input, we take the average of the embedding matrix $\mathbf{H}_L$ to get the context embedding:

$$\boldsymbol{h}_c = \text{Mean}(\mathbf{H}_L). \tag{6}$$

MIP layer compares the basic meaning vector $\boldsymbol{h}_b$ and the contextual target meaning vector $\boldsymbol{h}_t$. We use a linear transformation to implement MIP:

$$\boldsymbol{h}_{\text{MIP}} = W_{\text{MIP}}^{\top}[\boldsymbol{h}_t; \boldsymbol{h}_b; |\boldsymbol{h}_t - \boldsymbol{h}_b|; \boldsymbol{h}_t * \boldsymbol{h}_b] + b_{\text{MIP}}, \tag{7}$$

where $[\cdot]$ is a readout method. $|\cdot|$ means absolute value. ; is concatenation, and $*$ denotes hadamard product. We combine these methods to readout different representations. W_{MIP} and b_{MIP} are weight and bias of MIP layer respectively. Similarly, we conduct SPV on context vector $\boldsymbol{h}_c$ and contextual target meaning vector $\boldsymbol{h}_t$:

$$\boldsymbol{h}_{\text{SPV}} = W_{\text{SPV}}^{\top}[\boldsymbol{h}_c; \boldsymbol{h}_t; |\boldsymbol{h}_c - \boldsymbol{h}_t|; \boldsymbol{h}_c * \boldsymbol{h}_t] + b_{\text{SPV}}, \tag{8}$$

where W_{SPV} and b_{SPV} are weight and bias of SPV layer respectively. POS information plays an important role in metaphor detection, so we extract POS vector $\boldsymbol{h}_{\text{POS}}$ from the right encoder. Finally, we combine MIP, SPV, and POS information to decide whether the target word is metaphorical:

$$\boldsymbol{y} = \sigma\left(W^{\top}[\boldsymbol{h}_{\text{MIP}}; \boldsymbol{h}_{\text{SPV}}; \boldsymbol{h}_{\text{POS}}] + b\right), \tag{9}$$

where W and b are weight and bias. σ is a softmax function. $\boldsymbol{y} \in \mathbb{R}^2$ indicates the predicted label distribution.

3.3 Training Objective

For a classification task, we use cross entropy loss as our optimization criterion:

$$\mathcal{L} = -\frac{1}{N}\sum_{i=1}^{N} w_{y_i} y_i \log(\hat{y}_i), \tag{10}$$

where N is the count of training samples. y_i and $\hat{y}_i$ denote the ground truth label and the predicted score for the i-th sample respectively. w_{y_i} is class weight to alleviate data unbalance problem.

4 Experiments

4.1 Datasets

Following most metaphor identification works, we use four widely-used public datasets. The statistic information is shown in Table 1.

VUA All (Steen, 2010): The largest metaphor dataset drawn from VU Amsterdam Metaphor Corpus (VUA). VUA collects sentences from the BNC-Baby Corpus, including four genres: academic, conversation, fiction, and news. VUA All dataset labels each word in each POS for each sentence.

VUA Verb (Steen, 2010): VUA Verb is a subset of VUA All. VUA Verb dataset only has verb targets.

MOH-X (Mohammad et al., 2016): MOH-X dataset focuses on the verb track. MOH-X collects metaphorical and literal usages for verbs from WordNet. Each verb in MOH-X has multiple senses, of which at least one is metaphorical.

TroFi (Birke and Sarkar, 2006, 2007): TroFi dataset only includes verb targets. The literal and metaphorical usages for 50 English verbs are drawn from The 1987-89 Wall Street Journal Corpus.

Dataset	#Sent.	#Target	%Met.	Avg. Len
VUA All_{tr}	6,323	116,622	11.19	18.4
VUA All_{val}	1,550	38,628	11.62	24.9
VUA All_{te}	2,694	50,175	12.44	18.6
VUA Verb_{tr}	7,479	15,516	27.90	20.2
VUA Verb_{val}	1,541	1,724	26.91	25.0
VUA Verb_{te}	2,694	5,873	29.98	18.6
MOH-X	647	647	48.69	8.0
TroFi	3,737	3,737	43.54	28.3

Table 1: Datasets information. **#Sent.**: Number of sentences. **#Target**: Number of target words. **%Met.**: Percentage of metaphors. **Avg. Len**: Average sentence length.

4.2 Baselines

RNN_ELMo (Gao et al., 2018) and **RNN_BERT** (Mao et al., 2019): two RNN based sequence labeling models. They concatenate embeddings of ELMo (or BERT) and GloVe to represent a word.

RNN_HG and **RNN_MHCA** (Mao et al., 2019): RNN_HG uses MIP to compare literal target meanings and contextual target meanings, which are represented by GloVe and ELMo embeddings respectively. RNN_MHCA is based on SPV, with multi-head contextual attention utilized.

MUL_GCN (Le et al., 2020): MUL_GCN adopts a multi-task learning framework to tackle metaphor

Model	VUA All				VUA Verb				MOH-X (10 fold)			
	Pre.	Rec.	F1	Acc.	Pre.	Rec.	F1	Acc.	Pre.	Rec.	F1	Acc.
RNN_ELMo	71.6	73.6	72.6	93.1	68.2	71.3	69.7	81.4	79.1	73.5	75.6	77.2
RNN_BERT	71.5	71.9	71.7	92.9	66.7	71.5	69.0	80.7	75.1	81.8	78.2	78.1
RNN_HG	71.8	76.3	74.0	93.6	69.3	72.3	70.8	82.1	79.7	79.8	79.8	79.7
RNN_MHCA	73.0	75.7	74.3	93.8	66.3	**75.2**	70.5	81.8	77.5	83.1	80.0	79.8
MUL_GCN	74.8	75.5	75.1	93.8	72.5	70.9	71.7	83.2	79.7	80.5	79.6	79.9
RoBERTa_SEQ†	80.4	74.9	77.5	-	79.2	69.8	74.2	-	-	-	-	-
DeepMet†	*82.0*	71.3	76.3	-	*79.5*	70.8	74.9	-	-	-	-	-
MelBERT	80.1	*76.9*	*78.5*	-	78.7	72.9	*75.7*	-	-	-	-	-
MrBERT	**82.7**	72.5	77.2	*94.7*	**80.8**	71.5	**75.9**	**86.4**	*80.0*	**85.1**	*82.1*	*81.9*
MisNet	80.4	**78.4**	**79.4**	**94.9**	78.3	*73.6*	**75.9**	*86.0*	**84.2**	*84.0*	**83.4**	**83.6**

Table 2: Results on VUA All, VUA Verb, and MOH-X. Best in **bold** and second best in *italic underlined*. The top block exhibits RNN based methods, while the middle block includes the transformer based. The † results are reproduced by Choi et al. (2021).[3]

detection and word sense disambiguation simultaneously. It also uses Graph Convolution Network with Bi-LSTM to encode dependency relations.

RoBERTa_SEQ (Leong et al., 2020): a sequence labeling baseline model provided by ACL 2020 metaphor detection shared task. RoBERTa_SEQ takes a sentence as input, and uses a softmax classifier to predict the metaphoricity for each token.

DeepMet (Su et al., 2020): the winning model in ACL 2020 metaphor detection shared task. It models metaphor identification as a reading comprehension task, with query features, fine-grained POS features, and context features etc. incorporated.

MelBERT (Choi et al., 2021): a model based on RoBERTa. It utilizes siamese network as well. However, MelBERT assumes that the target word used alone is literal.

MrBERT (Song et al., 2021): MrBERT regards metaphor detection as a relation classification task. It extracts dependency relations among verbs and subjects or objects, and embeds relations into BERT input.

4.3 Implementation Details

Gao et al. (2018) expanded VUA All dataset with POS tags. We retrieve basic usages from a digital Oxford dictionary[4] following Su et al. (2021). Since MOH-X does not have a training, validation, and test split, we perform 10-fold cross validation on it. Also, following previous studies (Choi et al., 2021; Song et al., 2021), we conduct zero-shot transfer on TroFi dataset to examine the generalization ability of MisNet. We use the RoBERTa (Liu et al., 2019) implementation for BERT, provided by HuggingFace[5]. It has stacked 12-layer transformer encoders, each with 12 attention heads. The hidden dimension in each layer is 768. Both hidden dimensions in MIP and SPV layer are 768.

VUA All: for VUA All dataset the learning rate is 3e-5. The epoch number is 15 and the training batch size is 64. Since VUA All suffers from data unbalance, we use different class weights in cross entropy loss function, 1 for literal samples and 5 for metaphors. **VUA Verb:** for VUA Verb, we remove POS tag from the input since it provides few information when only training on verbs. The training batch size is 64 with a 3e-5 learning rate. The class weights are 1 for literal samples and 4 for metaphors. We train for 15 epochs. **MOH-X:** MOH-X is a balanced dataset, such that we do not apply different class weights. The batch size is 16 with a 3e-5 learning rate, and we train for 15 epochs. All the experiments adopt AdamW (Peters et al., 2019) optimizer. For VUA All and VUA Verb, we take the best model on validation set to do testing. For MOH-X, we take the best score in each fold, and calculate the average over total 10 folds. All experiments are done in PyTorch 1.10 and cuda 11.2, on a single NVIDIA RTX 3090 GPU. Our code, saved model weights, and datasets are available for more details.

5 Results and Analysis

5.1 Overall Results

Following Mao et al. (2019), we mainly focus on F1 score. As shown in Table 2, MisNet obtains

[3]In ACL 2020 shared task, participants can manipulate training dataset or perform ensemble learning, making the original results incomparable.

[4]https://www.lexico.com/

[5]https://huggingface.co/roberta-base

Model	Academic				Conversation				Fiction				News			
	Pre.	Rec.	F1	Acc.	Pre.	Rec.	F1	Acc.	Pre.	Rec.	F1	Acc.	Pre.	Rec.	F1	Acc.
RNN_ELMo	78.2	80.2	79.2	92.8	64.9	63.1	64.0	94.6	61.4	69.1	65.1	93.1	72.7	71.2	71.9	91.6
RNN_BERT	76.7	76.0	76.4	91.9	64.7	64.2	64.4	94.6	66.5	68.6	67.5	*93.9*	71.2	72.5	71.8	91.4
RNN_HG	76.5	**83.0**	79.6	92.7	63.6	**72.5**	67.8	*94.8*	61.8	74.5	67.5	93.4	71.6	*76.8*	74.1	91.9
RNN_MHCA	79.6	80.0	79.8	*93.0*	64.0	71.1	67.4	*94.8*	64.8	70.9	67.7	93.8	74.8	75.3	75.0	*92.4*
RoBERTa_SEQ†	*86.0*	77.3	81.4	-	70.5	69.8	70.1	-	73.9	72.7	73.3	-	82.2	74.1	*77.9*	-
DeepMet†	**88.4**	74.7	81.0	-	*71.6*	71.1	*71.4*	-	**76.1**	70.1	73.0	-	**84.1**	67.6	75.0	-
MelBERT	85.3	*82.5*	**83.9**	-	70.1	71.7	70.9	-	74.0	*76.8*	*75.4*	-	81.0	73.7	77.2	-
MisNet	85.1	*82.5*	*83.8*	**94.5**	**71.8**	*72.0*	**71.9**	**95.7**	*74.5*	**77.5**	**76.0**	**95.5**	*82.6*	**77.0**	**79.7**	**94.1**

Model	Verb				Adjective				Adverb				Noun			
	Pre.	Rec.	F1	Acc.	Pre.	Rec.	F1	Acc.	Pre.	Rec.	F1	Acc.	Pre.	Rec.	F1	Acc.
RNN_ELMo	68.1	71.9	69.9	-	56.1	60.6	58.3	-	67.2	53.7	59.7	94.8	59.9	60.8	60.4	-
RNN_BERT	67.1	72.1	69.5	87.9	58.1	51.6	54.7	88.3	64.8	61.1	62.9	94.8	63.3	56.8	59.9	88.6
RNN_HG	66.4	75.5	70.7	*88.0*	59.2	**65.6**	62.2	89.1	61.0	66.8	63.8	94.5	60.3	*66.8*	63.4	88.4
RNN_MHCA	66.0	*76.0*	70.7	87.9	61.4	61.7	61.6	*89.5*	66.1	60.7	63.2	*94.9*	69.1	58.2	63.2	*89.8*
RoBERTa_SEQ†	74.4	75.1	74.8	-	*72.0*	57.1	63.7	-	77.6	63.9	70.1	-	**76.5**	59.0	66.6	-
DeepMet†	**78.8**	68.5	73.3	-	**79.0**	52.9	63.3	-	*79.4*	66.4	72.3	-	**76.5**	57.1	65.4	-
MelBERT	74.2	75.9	*75.1*	-	69.4	60.1	*64.4*	-	**80.2**	*69.7*	**74.6**	-	*75.4*	66.5	**70.7**	-
MisNet	*77.5*	**77.7**	**77.6**	**91.4**	68.8	*65.2*	**67.0**	**91.2**	76.4	**70.5**	*73.3*	**96.3**	74.4	**67.2**	*70.6*	**91.6**

Table 3: Breakdown results for genre and POS on VUA All. Best in **bold** and second best in *italic underlined*.

competitive results. In VUA All dataset, MisNet gains as most as 7.7 and 3.1 F1 score improvements compared with RNN based models and transformer based models respectively. Also, MisNet gains nearly 1.0 F1 score over the strongest baseline MelBERT, and achieves highest recall and accuracy scores. MisNet can fully utilize POS information, such that it has a strong ability to distinguish the impossible cases, like conjunctions and exclamations, from the potential ones.

In VUA Verb dataset, we remove POS tag in input because it provides little information when there is only one word class, i.e., we only make judgements via MIP and SPV layers. We get improvements by as most as 6.9 and 1.7 F1 scores compared with RNN based methods and transformer based methods respectively. It is worth mentioning that the strongest baseline MrBERT uses dependency parsing to extract subjects and objects for verbs, but MisNet still obtains promising results only via semantic matching methods, which shows the importance to properly utilize linguistic rules.

We attain improvements by 1.3 F1 scores against the strongest baseline MrBERT in MOH-X dataset, and achieve best precision and accuracy scores. Compared with RNN based methods, the performance is improved by as most as 7.8 F1 scores as well. We notice that MisNet performs better on MOH-X than VUA Verb: MOH-X is built upon WordNet via extracting metaphorical and literal usages of certain verbs, which means most metaphors in MOH-X are conventional metaphors. MisNet can get benefits from basic usages while the other baselines may fail to capture the basic meanings. However, verbs in VUA Verb dataset are much more complex, including auxiliary verbs, link verbs and etc. Predictions for VUA Verb are much harder.

5.2 VUA All Breakdown Results

Table 3 shows two breakdown analysis on VUA All dataset. In the genre track, MisNet outperforms the previous baselines in conversation, fiction, and news. We achieve a promising result on academic as well. All the methods perform better on academic and news, which have formal language usages so the patterns beneath are easy to perceive.

In the POS track, we find that MisNet achieves largest improvements on verb and adjective, with 2.5 and 2.6 F1 scores gained respectively. Verbs and adjectives are often used metaphorically, so there are more positive samples in VUA All dataset. Also, verb samples take the biggest portion in VUA All dataset, which makes the training on verbs more thorough. The performance on adverb is mediocre, because adverbs are very different internally. For instance, adverbs of time, place, and degree etc., can rarely be metaphors. Such complexity makes adverbs more difficult to judge.

5.3 Zero-shot Transfer on TroFi

We conduct zero-shot transfer on TroFi dataset, i.e., using TroFi only for testing. As Table 4 shows, MisNet outperforms all the baselines. Notably, the baselines with ♠ are trained on an expanded version of VUA All (Choi et al., 2021), so they have more training data. MrBERT explicitly utilizes dependency relations to benefit verb metaphor detection. However, we still attain best results in all metrics, which indicates that MisNet has strong generalization ability.

Model	TroFi (Zero-shot)			
	Pre.	Rec.	F1	Acc.
RoBERTa_SEQ♠	53.6	70.1	60.7	-
DeepMet♠	*53.7*	72.9	61.7	-
MelBERT♠	53.4	74.1	62.0	-
MrBERT	**53.8**	*75.0*	*62.7*	*61.1*
MisNet	**53.8**	**76.2**	**63.1**	**61.2**

Table 4: Zero-shot transfer results on TroFi dataset. We use MisNet trained on VUA All dataset.

5.4 Effectiveness Study

We conduct ablation experiments to test the effectiveness of different modules and features in MisNet, as Table 5 illustrates. In each ablation setting, the performance drops, which demonstrates the capability of each part. MIP module is more important than SPV as is observed. A conventional metaphor may be normal for its frequent context, so SPV becomes invalid. But MIP can notice the discrepancy between the contextual target meaning and its basic meaning. POS provides useful information for MisNet to filter out the impossible cases, without which the model performs worse. When basic usages are aborted, MisNet may fail to represent basic meanings, such that some metaphors cannot be detected. When feature embeddings are removed, MisNet works quite badly. Our designed feature embeddings can help model to treat different parts differently to better utilize features.

We also evaluate the impacts from different readout methods. We replace the readout method in Eq. 7 and Eq. 8 with candidates from Table 5. We find that $|\boldsymbol{u}-\boldsymbol{v}|$ and $(\boldsymbol{u};\boldsymbol{v})$ are two crucial components, without which the performance drops significantly. $|\boldsymbol{u}-\boldsymbol{v}|$ can directly reveal the difference between two representations, while $(\boldsymbol{u};\boldsymbol{v})$ can preserve all the original information. However, the default setting $(\boldsymbol{u};\boldsymbol{v};|\boldsymbol{u}-\boldsymbol{v}|;\boldsymbol{u}*\boldsymbol{v})$ is the best since all components work as an ensemble.

Ablation	Pre.	Rec.	F1	Acc.
-MIP	**83.1**	72.8	77.6	94.8
-SPV	81.2	76.0	78.5	94.8
-POS	79.1	77.4	78.2	94.6
-Basic Usage	81.2	75.4	78.2	94.8
-Feature Embedding	78.4	77.7	78.0	94.6
MisNet♣	80.4	**78.4**	**79.4**	**94.9**
Readout Method	**Pre.**	**Rec.**	**F1**	**Acc.**
$(\boldsymbol{u};\boldsymbol{v})$	81.5	76.1	78.7	**94.9**
$(\lvert\boldsymbol{u}-\boldsymbol{v}\rvert)$	**82.5**	74.5	78.3	**94.9**
$(\lvert\boldsymbol{u}*\boldsymbol{v}\rvert)$	73.9	80.4	77.1	94.0
$(\lvert\boldsymbol{u}-\boldsymbol{v}\rvert;\boldsymbol{u}*\boldsymbol{v})$	75.5	**81.2**	78.3	94.4
$(\boldsymbol{u};\boldsymbol{v};\boldsymbol{u}*\boldsymbol{v})$	82.4	74.2	78.1	94.8
$(\boldsymbol{u};\boldsymbol{v};\lvert\boldsymbol{u}-\boldsymbol{v}\rvert)$	79.5	77.4	78.5	94.7
$(\boldsymbol{u};\boldsymbol{v};\lvert\boldsymbol{u}-\boldsymbol{v}\rvert;\boldsymbol{u}*\boldsymbol{v})$♣	80.4	78.4	**79.4**	**94.9**

Table 5: Effectiveness study on VUA All dataset. ♣ are the default MisNet settings.

Table 6 shows the quality analysis results. The top block indicates that MisNet can better detect conventional metaphors by using basic usages, which confirms our assumptions at the beginning. The middle block includes indirect metaphors, of which the metaphoricity is predicted upon preceding words. Metaphors in the bottom block can be very confusing. If we do not use a wider context, we can't distinguish accurately. However, MisNet only takes sentence-level inputs, thus we cannot properly handle these situations. We leave it as a future work.

MisNet	-Basic U.	Label	Sentence
✓	✗	M	The ban on emergency work was *tightened*.
✓	✗	M	A new minister would *operate* inside the DoE.
✓	✗	M	A financial *crash* of global proportions.
✓	✗	M	Raising the federal debt *ceiling*.
✗	✗	M	Er, just maybe the size of *this*.
✗	✗	M	I'm gonna play with *that* and see what.
✗	✗	M	She *bought* it.
✗	✗	M	Thought you might want a *lift*.

Table 6: Quality analysis on VUA All dataset. Target words in *red italic*. **M** means **Metaphor**.

6 Conclusion

In this paper, we propose a novel metaphor detection model named MisNet, which uses MIP to compare the discrepancy between contextual target word meaning and its basic meaning, and utilizes

SPV to measure the incongruity between the target and its context. MisNet takes basic usages to encode basic target meanings, which can prevent the invalidation of MIP and SPV when dealing with conventional metaphors. Empirical results show that our method achieves competitive performance on several datasets.

Acknowledgements

This work is supported by 2018 National Major Program of Philosophy and Social Science Fund of China (18ZDA238) and Beijing Social Science Fund (16YYB021).

References

Dan Assaf, Yair Neuman, Yohai Cohen, Shlomo Argamon, Newton Howard, Mark Last, Ophir Frieder, and Moshe Koppel. 2013. Why "dark thoughts" aren't really dark: A novel algorithm for metaphor identification. pages 60–65.

Julia Birke and Anoop Sarkar. 2006. A clustering approach for nearly unsupervised recognition of nonliteral language. In *11th Conference of the European Chapter of the Association for Computational Linguistics*, pages 329–336, Trento, Italy. Association for Computational Linguistics.

Julia Birke and Anoop Sarkar. 2007. Active learning for the identification of nonliteral language. In *Proceedings of the Workshop on Computational Approaches to Figurative Language*, pages 21–28, Rochester, New York. Association for Computational Linguistics.

Andreas Blank. 2013. *Why do new meanings occur? A cognitive typology of the motivations for lexical semantic change*. De Gruyter Mouton.

L. Bloomfield. 1994. *Language*. Motilal Banarsidass Publishers.

Minjin Choi, Sunkyung Lee, Eunseong Choi, Heesoo Park, Junhyuk Lee, Dongwon Lee, and Jongwuk Lee. 2021. MelBERT: Metaphor detection via contextualized late interaction using metaphorical identification theories. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 1763–1773, Online. Association for Computational Linguistics.

Alexis Conneau, Douwe Kiela, Holger Schwenk, Loïc Barrault, and Antoine Bordes. 2017. Supervised learning of universal sentence representations from natural language inference data. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 670–680, Copenhagen, Denmark. Association for Computational Linguistics.

Peter Crisp, Raymond Gibbs, Alice Deignan, Graham Low, Gerard Steen, Lynne Cameron, Elena Semino, Joe Grady, Alan Cienki, Zoltan Kövecses, et al. 2007. Mip: A method for identifying metaphorically used words in discourse. *Metaphor and Symbol*, 22(1):1–39.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, Minneapolis, Minnesota. Association for Computational Linguistics.

Erik-Lân Do Dinh, Hannah Wieland, and Iryna Gurevych. 2018. Weeding out conventionalized metaphors: A corpus of novel metaphor annotations. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 1412–1424, Brussels, Belgium. Association for Computational Linguistics.

John R Firth. 1957. A synopsis of linguistic theory, 1930-1955. *Studies in linguistic analysis*.

Ge Gao, Eunsol Choi, Yejin Choi, and Luke Zettlemoyer. 2018. Neural metaphor detection in context. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 607–613, Brussels, Belgium. Association for Computational Linguistics.

Pragglejaz Group. 2007. Mip: A method for identifying metaphorically used words in discourse. *Metaphor and symbol*, 22(1):1–39.

Hessel Haagsma and Johannes Bjerva. 2016. Detecting novel metaphor using selectional preference information. In *Proceedings of the Fourth Workshop on Metaphor in NLP*, pages 10–17, San Diego, California. Association for Computational Linguistics.

George Lakoff and Mark Johnson. 2008. *Metaphors we live by*. University of Chicago press.

Duong Le, My Thai, and Thien Nguyen. 2020. Multi-task learning for metaphor detection with graph convolutional neural networks and word sense disambiguation. *Proceedings of the AAAI Conference on Artificial Intelligence*, 34(05):8139–8146.

Chee Wee (Ben) Leong, Beata Beigman Klebanov, Chris Hamill, Egon Stemle, Rutuja Ubale, and Xianyang Chen. 2020. A report on the 2020 VUA and TOEFL metaphor detection shared task. In *Proceedings of the Second Workshop on Figurative Language Processing*, pages 18–29, Online. Association for Computational Linguistics.

Zhenxi Lin, Qianli Ma, Jiangyue Yan, and Jieyu Chen. 2021. CATE: A contrastive pre-trained model for metaphor detection with semi-supervised learning.

In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 3888–3898, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Roberta: A robustly optimized bert pretraining approach.

Rui Mao, Chenghua Lin, and Frank Guerin. 2019. End-to-end sequential metaphor identification inspired by linguistic theories. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 3888–3898, Florence, Italy. Association for Computational Linguistics.

Saif Mohammad, Ekaterina Shutova, and Peter Turney. 2016. Metaphor as a medium for emotion: An empirical study. In *Proceedings of the Fifth Joint Conference on Lexical and Computational Semantics*, pages 23–33, Berlin, Germany. Association for Computational Linguistics.

Matthew E. Peters, Sebastian Ruder, and Noah A. Smith. 2019. To tune or not to tune? adapting pretrained representations to diverse tasks. In *Proceedings of the 4th Workshop on Representation Learning for NLP (RepL4NLP-2019)*, pages 7–14, Florence, Italy. Association for Computational Linguistics.

Alec Radford, Jeff Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. 2019. Language models are unsupervised multitask learners.

Jinfeng Rao, Linqing Liu, Yi Tay, Wei Yang, Peng Shi, and Jimmy Lin. 2019. Bridging the gap between relevance matching and semantic matching for short text similarity modeling. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 5370–5381, Hong Kong, China. Association for Computational Linguistics.

Nils Reimers and Iryna Gurevych. 2019. Sentence-BERT: Sentence embeddings using Siamese BERT-networks. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 3982–3992, Hong Kong, China. Association for Computational Linguistics.

Ekaterina Shutova and Lin Sun. 2013. Unsupervised metaphor identification using hierarchical graph factorization clustering. In *Proceedings of the 2013 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 978–988, Atlanta, Georgia. Association for Computational Linguistics.

Wei Song, Shuhui Zhou, Ruiji Fu, Ting Liu, and Lizhen Liu. 2021. Verb metaphor detection via contextual relation learning. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 4240–4251, Online. Association for Computational Linguistics.

Gerard Steen. 2010. *A method for linguistic metaphor identification: From MIP to MIPVU*, volume 14. John Benjamins Publishing.

Chang Su, Kechun Wu, and Yijiang Chen. 2021. Enhanced metaphor detection via incorporation of external knowledge based on linguistic theories. In *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*, pages 1280–1287, Online. Association for Computational Linguistics.

Chuandong Su, Fumiyo Fukumoto, Xiaoxi Huang, Jiyi Li, Rongbo Wang, and Zhiqun Chen. 2020. DeepMet: A reading comprehension paradigm for token-level metaphor detection. In *Proceedings of the Second Workshop on Figurative Language Processing*, pages 30–39, Online. Association for Computational Linguistics.

Xiaoyu Tong, Ekaterina Shutova, and Martha Lewis. 2021. Recent advances in neural metaphor processing: A linguistic, cognitive and social perspective. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 4673–4686, Online. Association for Computational Linguistics.

Yulia Tsvetkov, Leonid Boytsov, Anatole Gershman, Eric Nyberg, and Chris Dyer. 2014. Metaphor detection with cross-lingual model transfer. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 248–258, Baltimore, Maryland. Association for Computational Linguistics.

Peter Turney, Yair Neuman, Dan Assaf, and Yohai Cohen. 2011. Literal and metaphorical sense identification through concrete and abstract context. In *Proceedings of the 2011 Conference on Empirical Methods in Natural Language Processing*, pages 680–690, Edinburgh, Scotland, UK. Association for Computational Linguistics.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Ł ukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc.

Hai Wan, Jinxia Lin, Jianfeng Du, Dawei Shen, and Manrong Zhang. 2021. Enhancing metaphor detection by gloss-based interpretations. In *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*, pages 1971–1981, Online. Association for Computational Linguistics.

Mingyu Wan, Kathleen Ahrens, Emmanuele Chersoni, Menghan Jiang, Qi Su, Rong Xiang, and Chu-Ren Huang. 2020. Using conceptual norms for metaphor detection. In *Proceedings of the Second Workshop on Figurative Language Processing*, pages 104–109, Online. Association for Computational Linguistics.

Yorick Wilks. 1975. A preferential, pattern-seeking, semantics for natural language inference. *Artificial intelligence*, 6(1):53–74.

Yorick Wilks. 1978. Making preferences more active. *Artificial intelligence*, 11(3):197–223.

Chuhan Wu, Fangzhao Wu, Yubo Chen, Sixing Wu, Zhigang Yuan, and Yongfeng Huang. 2018. Neural metaphor detecting with CNN-LSTM model. In *Proceedings of the Workshop on Figurative Language Processing*, pages 110–114, New Orleans, Louisiana. Association for Computational Linguistics.

Runqi Yang, Jianhai Zhang, Xing Gao, Feng Ji, and Haiqing Chen. 2019. Simple and effective text matching with richer alignment features. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 4699–4709, Florence, Italy. Association for Computational Linguistics.

Fast and Accurate End-to-End Span-based Semantic Role Labeling as Word-based Graph Parsing

Shilin Zhou, Qingrong Xia, Zhenghua Li*, Yu Zhang, Yu Hong, Min Zhang
Institute of Artificial Intelligence, School of Computer Science and Technology,
Soochow University, Suzhou, China
{slzhou.cs,yzhang.cs}@outlook.com; kirosummer.nlp@gmail.com
{zhli13,hongy,minzhang}@suda.edu.cn

Abstract

This paper proposes to cast end-to-end span-based SRL as a word-based graph parsing task. The major challenge is how to represent spans at the word level. Borrowing ideas from research on Chinese word segmentation and named entity recognition, we propose and compare four different schemata of graph representation, i.e., *BES*, *BE*, *BIES*, and *BII*, among which we find that the *BES* schema performs the best. We further gain interesting insights through detailed analysis. Moreover, we propose a simple constrained Viterbi procedure to ensure the legality of the output graph according to the constraints of the SRL structure. We conduct experiments on two widely used benchmark datasets, i.e., CoNLL05 and CoNLL12. Results show that our word-based graph parsing approach achieves consistently better performance than previous results, under all settings of end-to-end and predicate-given, without and with pre-trained language models (PLMs). More importantly, our model can parse 669/252 sentences per second, without and with PLMs respectively.

1 Introduction

As a fundamental natural language processing (NLP) task, semantic role labeling (SRL) uses predicate-argument structure to represent the shallow semantic meaning of sentences. SRL structure is shown to be helpful for many downstream NLP tasks, such as machine translation (Liu and Gildea, 2010; Marcheggiani et al., 2018) and question answering (Wang et al., 2015).

There exist two forms of concrete SRL formalism in the community, i.e., word-based (also known as dependency-based SRL) and span-based, depending on whether an argument consists of a single word or a word span. Compared with word-based SRL, span-based SRL is more complex due to difficulties in determining argument boundaries. Figure 1 shows the span-based SRL structure for two predicates. Semantic roles of arguments are distinguished with edge labels, such as "A0" (agent) and "A1" (patient). This work focuses on the end-to-end span-based SRL task, and proposes a unified model to simultaneously recognize predicates and arguments in the input sentence.

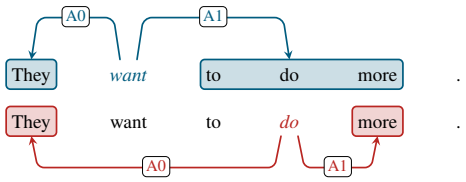

Figure 1: An example of span-based SRL, where "*want*" and "*do*" are two predicates.

In recent years, span-based SRL has achieved substantial performance boost due to the tremendous progress made by deep neural network models, especially by pre-trained language models (PLMs). Currently, there are mainly two mainstream approaches, i.e., BIO-based (Zhou and Xu, 2015) sequence labeling and span-based graph parsing (He et al., 2018).

The BIO-based sequence labeling approach first identifies the predicates and then finds arguments for each predicate independently by labeling every word with BIO tags, like "B-A0" or "I-A0". Its major weakness is that a sentence has to be encoded and decoded for multiple times, each time for one predicate (Zhou and Xu, 2015; Shi and Lin, 2019), thus proportionally reducing the training and inference efficiency.[1] Zhou and Xu (2015) concatenate an indicator embedding to each input token, where the focused predicate corresponds to 1, and others to 0. Shi and Lin (2019) append the focused predicate word to the end of the sentence before getting into BERT (Devlin et al., 2019).

* Corresponding author

[1] Some BIO-based approaches, for example Strubell et al. (2018), only encode the input sentence once without using predicate indicators, but this leads to inferior performance.

Proceedings of the 29th International Conference on Computational Linguistics, pages 4160–4171
October 12–17, 2022.

The span-based graph parsing approach directly considers all word spans as candidate argument nodes and links them to predicate nodes (He et al., 2018; Li et al., 2019). However, this approach also suffers from a severe inefficiency problem, since there are $O(n)$ candidate predicates and $O(n^2)$ candidate arguments, leading to a big search space of $O(n^3)$. Previous works usually employ heuristic pruning techniques to improve efficiency.

Inspired by recent works on semantic dependency graph parsing (SDGP) (Oepen et al., 2014; Dozat and Manning, 2018; Wang et al., 2019), this work for the first time proposes a word-based graph parsing approach for end-to-end span-based SRL. End-to-end means that all predicates and arguments in a sentence are inferred simultaneously and by a single model. The key challenge is how to represent span-based arguments in word-based graphs in which nodes correspond to single words. Once this is solved, we can build our parser on the shoulder of existing word-based graph parsing models. This work employs the second-order model of Wang et al. (2019). In summary, our work makes the following contributions:

- We propose a new word-based graph parsing approach for end-to-end span-based SRL. Via a straightforward simplification, our approach can be applied to the *predicate-given* setting.
- Borrowing ideas from research on Chinese word segmentation (CWS) and named entity recognition (NER), we propose and investigate several graph schemata. We find the *BES* schema is steadily superior to others and obtain interesting insights via detailed analysis.
- Inevitably, graph parsing models may output illegal graph that cannot be properly transformed into SRL structure. To deal with this, we propose a simple constrained Viterbi procedure for post-processing illegal graphs.
- We conduct experiments on the CoNLL05 and CoNLL12 benchmark datasets. Our proposed approach achieves consistently better performance than previous results, under all settings of end-to-end and predicate-given, with and without PLMs. More importantly, our parser is much more faster than previous parsers and can analyze 669/252 sentences per second, without and with PLMs.

We release our code, configuration files, and models at `https://github.com/zsLin177/SRL-as-GP`.

2 Related Works

Span-based SRL. As two mainstream neural models, the BIO-based and span-based graph parsing approaches handle SRL in different ways.

The BIO-based approach usually predicts predicates first and then recognizes arguments for each predicate via sequence labeling. For each predicate, Zhou and Xu (2015) indicates the position of the predicate via indicator embedding, and then encode the sentence using multi-layer BiLSTMs, and finally apply a CRF layer to find the best label sequence. Shi and Lin (2019) append the focused predicate word to the original sentence, and then feed the sentence into BERT, and then apply BiLSTM for further encoding.

The span-based graph parsing approach is proposed by He et al. (2018). The idea is directly predicting relations between candidate predicates (single words) and arguments (word spans) in a graph. Li et al. (2019) apply the approach to the word-based SRL task.

Besides the two mainstream approaches, researchers have explored other interesting directions. Zhang et al. (2021) propose a two-step span recognition approach, i.e., first identifying a head word and then extending the word into a span. Blloshmi et al. (2021) cast the SRL task under the predicate-given setting as a sequence-to-sequence task like machine translation. Given a predicate, its SRL structure is converted into a token sequence. Their approach achieves competitive performance by using BART (Lewis et al., 2020).

Concurrently, Zhang et al. (2022) cast span-based SRL as *a tree parsing approach*. Given a predicate, the word span corresponding to an argument is represented as latent trees. The sentence is encoded once without using predicate indicators, but each predicate require an independent decoding process.

Syntax-enhanced SRL. Due to the close connection between syntax parsing and SRL, there has been a lot of works on syntax-enhanced SRL. Strubell et al. (2018) and Zhou et al. (2020) jointly handle syntactic parsing and SRL under the multi-task learning framework. Xia et al. (2019) inject auto-parsed syntactic trees into SRL as extra features . In contrast, our work is a pure modeling study, and does not use external syntactic knowledge.

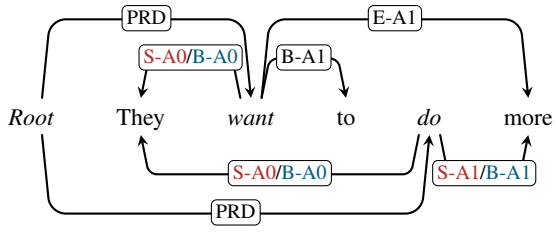

(a) *BES* and *BE*

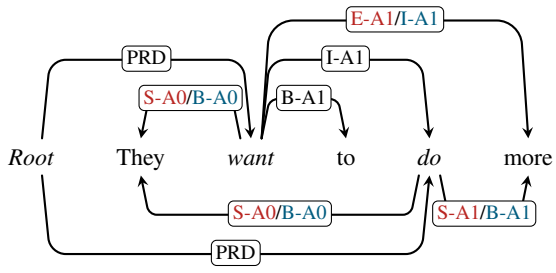

(b) *BIES* and *BII*

Figure 2: Proposed four different schemata. Labels in black are the shared part. Red and blue labels belong to *BES*, *BIES* and *BE*, *BII* respectively.

SDGP. In contrast to predicate-argument structure, SDGP belongs to another category of semantic representation formalism, using word-based graphs to represent semantics of sentences (Oepen et al., 2014, 2015). The specific forms include DM (Ivanova et al., 2012), PSD (Hajič et al., 2012), PAS (Miyao and Tsujii, 2004), etc.

Straightforwardly, graph parsing is a mainstream approach for SDGP. Dozat and Manning (2018) propose an efficient first-order graph parser to find an optimal graph from a fully connected graph. Wang et al. (2019) extend the model of Dozat and Manning (2018) by introducing second-order information. They compare two approximate high-order inference methods, i.e., mean filed variational inference and loopy belief propagation.

Word-based graph parsing for word-based SRL. As far as we know, Li et al. (2020) for the first time propose to treat word-based SRL as a SGDP task. Since arguments correspond to single words in word-based SRL, the two tasks are very similar. They employ the SGDP model of Wang et al. (2019) straightforwardly. Moreover, their study focuses on the *predicate-given* setting. First, they use a separate sequence labeling model to predict predicates. The SDGP model is then applied to recognize arguments.

3 Proposed Graph Schemata

This work proposes to cast end-to-end span-based SRL as a word-based graph parsing task. The key challenge is to design a suitable graphical schema so that all predicates and their span-based arguments can be represented simultaneously in one graph without ambiguity. And the graph can be transformed to its corresponding SRL structure without performance loss.

3.1 SRL-to-Graph Transformation

We design four different schemata for transforming span-based SRL structures into word-based graphs. The basic idea is linking words in an argument to the corresponding predicate, and labeling the edges according to both semantic role labels and word positions in the argument.

Specifically, we add a pseudo "*Root*" node at the beginning of the sentence and link all the predicates to it with "PRD" as the edge label. This allows our model to simultaneously predict predicates and arguments in an end-to-end manner.

Borrowing ideas from research on CWS and NER, we propose and investigate two strategies for attaching argument words to corresponding predicates, i.e., *boundary-attach* and *all-attach*. The boundary-attach strategy connects only the start and end words of an argument to its predicate word, while the all-attach strategy connects all words of an argument to the predicate word. For each strategy, we design two concrete schemata, as follows.

Boundary-attach: *BES* and *BE*. Figure 2(a) shows the two schemata. When an argument contains multiple words, we attach only the start and end words to its corresponding predicates, using "B-r" and "E-r" as the edge labels, where r is the original semantic role label. As shown in Figure 2(a), the two schemata handle the argument "to do more" in the same way.

When an argument corresponds to a single word, for example, the argument "They", the *BE* schema simply uses "B-r" as the label, while the *BES* schema uses "S-r" to make a distinction. Our experiments show that such distinction consistently improves performance.

All-attach: *BIES* and *BII*. Figure 2(b) shows the two schemata. Each word in an argument is attached to its corresponding predicate. In the *BII* schema, the first word is labeled as "B-r", and the following words, if any, are labeled as "I-r",

where the prefix "I−" means being inside an argument.

Analogous to *BES*, *BIES* further distinguishes the end word in an argument using "E−r", and single-word arguments using "S−r".

In fact, there is another variant schema that belongs to the all-attach category, which is *BIS*. Due to space limitation, we do not introduce it in detail since our preliminary experiments show its performance lags behind the best schema by a large margin.

3.2 Graph-to-SRL Recovery

In the evaluation stage, given an input sentence, our graph parsing model outputs an optimal graph according to the underlying schema. Then, the job is to recover SRL structure. If the output graph is legal (i.e., without label conflicts), the recovery is quite straightforward. Taking the *BES* schema for example, all children nodes (words) of the pseudo "*Root*" are treated as predicates. Then, for each predicate, we recover all its arguments based on the edge labels. An argument corresponds to either a paired labels, such as "B−A0" and "E−A0", or a single label such as "S−A0".

Unfortunately, it is quite complex to guarantee legality of output graphs. To handle this issue, we propose a simple yet effective post-processing recipe based on constrained Viterbi decoding in Section 5.

4 Model

Based on our designed graphical schema, we can address span-based SRL as a word-based graph parsing task. Following Dozat and Manning (2018) and Wang et al. (2019), the framework of our model consists of two stages: 1) predicting all edges and 2) assigning labels for edges.

4.1 Encoder

BiLSTM. Under the setting without PLMs, we use BiLSTM as our encoder. The input of the i-th word w_i is the concatenation of word embedding $\mathbf{e}_i^{word}$, lemma embedding $\mathbf{e}_i^{lemma}$, and charLSTM representation vector:

$$\mathbf{x}_i = \mathbf{e}_i^{word} \oplus \mathbf{e}_i^{lemma} \oplus \mathbf{e}_i^{char} \quad (1)$$

where $\mathbf{e}_i^{char}$ is the output vector of a one-layer BiLSTM that encodes the character sequence (Lample et al., 2016). Then, a three-layer BiLSTM encoder

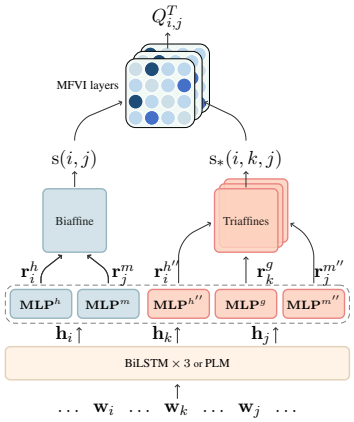

Figure 3: Illustration of our model. $s_*(i,k,j)$ corresponds to the second-order scores, where $* \in \{sib, cop, grd\}$.

produces a context-aware vector representation for each word.

$$\mathbf{h}_i = \mathbf{f}_i \oplus \mathbf{b}_i \quad (2)$$

where $\mathbf{f}_i$ and $\mathbf{b}_i$ respectively denote the output vectors of top-layer forward and backward LSTMs for w_i.

PLM. Under the setting with PLMs, we adopt ELMo (Peters et al., 2018) and BERT (Devlin et al., 2019) to get contextual word representation to boost the performance of our model.

$$\mathbf{h}_i = \mathbf{PLM}(w_i) \quad (3)$$

Concretely, we use the outputs of all three layers in ELMO, and of the top four layers in BERT, and then apply weighted sum to obtain the final output vector for each token.

4.2 Edge prediction

In SDGP, the prediction of edge is treated as a binary 0/1 classification task, where 1 means that there exists an edge between the given word pair and 0 otherwise. Here, for each edge $i \rightarrow j$ [2], we need to compute the logit score logit_{ij}. Then we can get the probability of existence of each edge Q_{ij} by applying the Sigmoid function. During inference, the edges that have $Q_{ij} > 0.5$ will be retained.

To facilitate computation and modeling, the first-order model of Dozat and Manning (2018) makes a strong assumption that edges are mutually independent and thus it only considers the information

[2]For convenience, we abbreviate the edge $i \rightarrow j$ as (i, j) in the remaining part of the paper.

(a) sibling (b) co-parent (c) grandchild

Figure 4: Three types of second-order sub-trees.

between the current two words when computing logits. However, in our case, the edges in the resulting graph usually have a strong correlation. For example, in our *BE* schema, a "B−∗" edge usually calls for a "E−∗" edge, and vice-versa, to form a complete argument. So, in this work, we extend first-order to second-order by adding three types of sub-trees, as shown in Figure 4. And we compute the logit by mean field variational inference (MFVI) following Wang et al. (2019).

The logit comes from two parts. The first part is the first-order score $\mathrm{s}(i,j)$. We use two MLPs to get representation vectors of a word as a head or a modifier respectively, and then use a BiAffine to compute edges' first-order scores as follows:

$$\begin{aligned}\mathbf{r}_i^{\mathrm{h}};\mathbf{r}_i^{\mathrm{m}} &= \mathrm{MLP}^{\mathrm{h}}\left(\mathbf{h}_i\right);\mathrm{MLP}^{\mathrm{m}}\left(\mathbf{h}_i\right)\\ \mathrm{s}(i,j) &= \begin{bmatrix}\mathbf{r}_j^{\mathrm{m}}\\ 1\end{bmatrix}^{\top}\mathbf{W}\mathbf{r}_i^{\mathrm{h}}\end{aligned} \tag{4}$$

where $\mathbf{W}\in\mathbb{R}^{(d+1)\times d}$.

The other part comes from second-order sub-trees. First, we use three new MLPs to get representations of each word for playing different roles in second-order sub-trees as follows:

$$\mathbf{r}_i^{\mathrm{h}''};\mathbf{r}_i^{\mathrm{m}''};\mathbf{r}_i^{\mathrm{g}} = \mathrm{MLP}^{\mathrm{h}''/\mathrm{m}''/\mathrm{g}}\left(\mathbf{h}_i\right) \tag{5}$$

where $\mathbf{r}_i^{h''}$, $\mathbf{r}_i^{m''}$, and $\mathbf{r}_i^{g}$ denote the representation vectors of w_i as head, modifier, and grandchild respectively. Then, a TriAffine scorer (Zhang et al., 2020) taking the three vectors as input is applied to compute the score of the corresponding second-order structure:

$$\mathrm{TriAFF}(\mathbf{v}_1,\mathbf{v}_2,\mathbf{v}_3) = \begin{bmatrix}\mathbf{v}_3\\ 1\end{bmatrix}^{\top}\mathbf{v}_1{}^{\top}\mathbf{W}'\begin{bmatrix}\mathbf{v}_2\\ 1\end{bmatrix} \tag{6}$$

where $\mathbf{W}'\in\mathbb{R}^{(d'+1)\times d'\times(d'+1)}$ and $\mathbf{v}_{1,2,3}\in\mathbb{R}^{d'}$. Finally, scores of the three types of sub-trees can be computed as follows respectively:

$$\mathrm{s}_{sib}(i,j,k) = \mathrm{TriAFF}_{sib}(\mathbf{r}_i^{\mathrm{h}''},\mathbf{r}_j^{\mathrm{m}''},\mathbf{r}_k^{\mathrm{m}''}) \tag{7}$$

$$\mathrm{s}_{cop}(i,j,k) = \mathrm{TriAFF}_{cop}(\mathbf{r}_i^{\mathrm{h}''},\mathbf{r}_j^{\mathrm{m}''},\mathbf{r}_k^{\mathrm{h}''}) \tag{8}$$

$$\mathrm{s}_{grd}(i,j,k) = \mathrm{TriAFF}_{grd}(\mathbf{r}_i^{\mathrm{h}''},\mathbf{r}_j^{\mathrm{m}''},\mathbf{r}_k^{\mathrm{g}}) \tag{9}$$

It should be noted that for symmetrical sibling sub-trees and co-parent sub-trees, we compute their corresponding scores only once, i.e., $\mathrm{s}_{sib}(i,j,k) = \mathrm{s}_{sib}(i,k,j)$ and $\mathrm{s}_{cop}(i,j,k) = \mathrm{s}_{cop}(k,j,i)$.

For a given edge (i,j), MFVI aggregates the final logit_{ij}^T and Q_{ij}^T from the corresponding first-order score and second-order scores iteratively as follows:

$$\begin{aligned}\mathcal{M}_{ij}^{t-1} &= \sum_{k\neq i,j} Q_{ik}^{t-1}\mathrm{s}_{sib}(i,j,k)\\ &\quad + Q_{kj}^{t-1}\mathrm{s}_{cop}(i,j,k)\\ &\quad + Q_{jk}^{t-1}\mathrm{s}_{grd}(i,j,k)\\ \mathrm{logit}_{ij}^t &= \mathrm{s}(i,j) + \mathcal{M}_{ij}^{t-1}\\ Q_{ij}^t &= \sigma(\mathrm{logit}_{ij}^t)\end{aligned} \tag{10}$$

where $t\in[1,T]$ is the iteration number. $\mathcal{M}_{ij}$ is an intermediate variable that stores message from second-order sub-tree scores. Q_{ij}^0 is initialized by applying Sigmoid on $\mathrm{s}(i,j)$. Through T times of update, we get the final logit_{ij}^T and probability Q_{ij}^T.

4.3 Label prediction

Similar to edge scoring, we use two extra MLPs and a set of Biaffines to compute the label scores:

$$\begin{aligned}\mathbf{r}_i^{\mathrm{h}'};\mathbf{r}_i^{\mathrm{m}'} &= \mathrm{MLP}^{\mathrm{h}'}\left(\mathbf{h}_i\right);\mathrm{MLP}^{\mathrm{m}'}\left(\mathbf{h}_i\right)\\ \mathrm{s}(i,j,\ell) &= \begin{bmatrix}\mathbf{r}_j^{\mathrm{m}'}\\ 1\end{bmatrix}^{\top}\mathbf{W}_{\ell}^{\mathtt{label}}\begin{bmatrix}\mathbf{r}_i^{\mathrm{h}'}\\ 1\end{bmatrix}\\ \mathrm{p}(\ell|i,j) &= \frac{\exp\left(\mathrm{s}(i,j,\ell)\right)}{\sum_{\ell'\in\mathcal{L}}\exp\left(\mathrm{s}(i,j,\ell')\right)}\end{aligned} \tag{11}$$

where $\mathrm{s}(i,j,\ell)$ is the score of the label ℓ for the edge (i,j). $\mathrm{p}(\ell|i,j)$ is the probability after softmax over all labels. Each label has its own Biaffine parameters $\mathbf{W}_{\ell}^{\mathtt{label}}\in\mathbb{R}^{(d+1)\times(d+1)}$.

4.4 Training

The loss of our system comes from both edge and label prediction modules. Given one sentence X and its gold graph G, the fully connected graph of X is denoted as C.

$$\begin{aligned}L_e(\theta) &= -\sum_{(i,j)\in G}\log Q_{ij}^T - \sum_{(i,j)\in C\backslash G}\log\left(1-Q_{ij}^T\right)\\ L_l(\theta) &= -\sum_{(i,j)\in G}\log\mathrm{p}(\hat{\ell}|i,j)\end{aligned} \tag{12}$$

where θ denotes model parameters; $C\backslash G$ is the set of incorrect edges; $\hat{\ell}$ is the gold label of edge. The

loss of the final model is the weighted sum of the two parts:

$$L(\theta) = \lambda L_l(\theta) + (1-\lambda)L_e(\theta) \quad (13)$$

where $\lambda = 0.06$ in our model.

5 Conflict resolution

During inference, we first use the edge prediction module to build the graph skeleton, and then use the label prediction module to assign labels to predicted edges. After that, we use a simple procedure to check whether the generated graph is legal. Concretely, for each predicate, we scan the edges of the predicate from left to right. For example, in the *BES* schema, a "B−∗" edge must be followed by a "E−∗" edge; "S−∗" edge and "E−∗" can be followed by a "B−∗" edge or "S−∗" edge. If the generated graph is legal, we can directly recover the corresponding SRL structure through Graph-to-SRL procedure described in 3.2.

However, since the label prediction module handles each edge independently, the resulting graph may contain conflicts, as shown in the upper part of Figure 5(a) [3]. First, if two consecutive edges are both labeled as "E−∗", such as the two "E−A0" edge, then it is impossible to recover the corresponding arguments. Another conflicting scene is when there exists a single outlier edge labeled as "B−∗" or "E−∗", such as "E−A1" edge in the figure.

Constrained Viterbi. We propose to employ constrained decoding to handle conflicts. Concretely, when conflicts occur during recovering arguments for a predicate in the output graph, we re-label all words in the sentence for the predicate. However, it is non-trivial to apply constrained Viterbi to our SDGP framework as a post-processing step.

Here we use the *BES* schema as an example, and the process for other schemata is similar. In the first stage, Q_{ij}^T means the probability that the edge appears in the final graph; while in the second stage, $\mathrm{p}(\ell|i,j)$ means the probability that the edge should be labeled as $\ell \in \mathcal{L} = \{\texttt{B−⋆}, \texttt{E−⋆}, \texttt{S−⋆}\}$. We can see that $\mathcal{L}$ does not include "I" and "O", meaning that the word is inside an argument or outside any arguments respectively. The two labels are indispensable for the sequence labeling procedure.

To solve this issue, we add two pseudo labels "O/I" into the label set, and redistribute the label

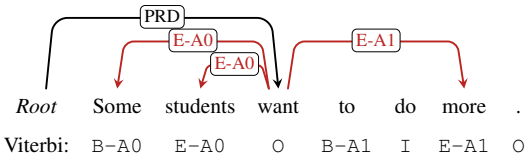

(a) A conflicting example in *BES*. Edges in red cause conflicts, and the sequence below is the corrected sequence via our constrained Viterbi.

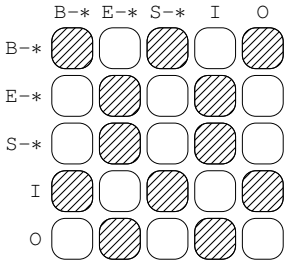

(b) The transition matrix of *BES*.

Figure 5: A conflicting example and the transition matrix in *BES* schema. The rows indicate the beginning of the transition and the columns indicate the ending. Cells with fence denote the prohibited transitions. I and O are two pseudo labels.

probability distribution as follows.

$$\begin{aligned} \mathrm{p}'(\ell|i,j) &= Q_{ij}^T \cdot \mathrm{p}(\ell|i,j) \\ \mathrm{p}'(\texttt{O}|i,j) &= \mathrm{p}'(\texttt{I}|i,j) = 1 - Q_{ij}^T \end{aligned} \quad (14)$$

where $\mathrm{p}'(\ell|i,j)$ is the probability for the normal label such as "B−A0". $\mathrm{p}'(\texttt{O}|i,j)$ and $\mathrm{p}'(\texttt{I}|i,j)$ share the same probability because they both mean that there is no edge pointing to the word, but "I" has an extra indication that there is an unpaired "B−∗" in the left side. Thus, we can solve the conflicts by controlling the transition matrix.

For example, as shown in Figure 5(b), we disallow transitions from "E−∗" to "E−∗". So, the "Some" and "students" are re-labeled as "B−A0" and "E−A0". And finally we get the correct argument span "Some students" with semantic role "A0".

6 Experiments

Data and evaluation. Experiments are conducted on CoNLL05 (Palmer et al., 2005) and larger-scale CoNLL12 (Pradhan et al., 2012), which are two widely used span-based SRL datasets. Following previous works on span-based SRL, we omit predicate sense prediction (Zhou and Xu, 2015; He et al., 2017). We use the official

[3] Here we only take the *BES* schema as a representative for discussion, and others can be viewed in § A.

Schema	WSJ			Brown		
	P	R	F_1	P	R	F_1
BES	**85.28**	83.66	**84.46**	**74.10**	70.76	**72.39**
BE	83.97	83.56	83.76	71.82	70.19	70.99
BIES	82.63	**83.92**	83.27	70.22	**72.03**	71.11
BII	81.65	83.44	82.54	67.72	70.74	69.20
+BERT						
BES	**87.15**	**88.44**	**87.79**	**79.44**	80.85	**80.14**
BE	86.37	87.93	87.14	78.18	79.91	79.04
BIES	85.91	88.17	87.03	77.59	**81.76**	79.62
BII	85.31	87.57	86.43	76.90	81.03	78.91

Table 1: Results on CoNLL05 datasets with respect to proposed four schemata. The variation between the 3 runs on WSJ and Brown is about 0.1 and 0.2, respectively. And it varies little between different schemata.

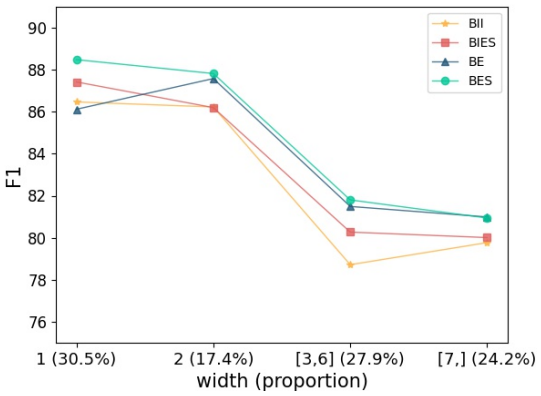

Figure 6: Analysis of the arguments with different width. The horizontal axis denotes the width of arguments and the proportion of arguments of the same width in the data set. The vertical axis denotes the F_1 value.

evaluation scripts[4]. We choose seeds randomly to run our model for 3 times and report the average results.

Hyper-parameter settings. We employ 300-dimension English word embeddings from GloVe (Pennington et al., 2014) for our experiments. We adopt most hyper-parameters of the SDGP work of Wang et al. (2019), except that we reduce the dimension of Char-LSTM from 400 to 100 to save the memory, which only slightly influence performance. For experiments with PLMs, we adopt ELMo (Peters et al., 2018) and BERT (Devlin et al., 2019) as our encoder. Following most of previous works (He et al., 2018; Xia et al., 2019), for ELMo, we froze its parameters during training. For BERT, we fine-tune its parameters for 10 epochs. The initial learning rate for models with and without BERT are 5e-5 and 1e-3 respectively. The hyper-parameter λ in the loss function (Eq. 13) is set to 0.06 in all experiments, based on preliminary experiment results.

6.1 Schema Comparison

Overall results. First, to compare the proposed four schemata and find which one is better, we conduct experiments on CoNLL05 datasets under the *end-to-end* setting. Table 1 shows results of different schemata. First, by comparing the two different attaching strategy, i.e., all-attach (*BII*, *BIES*) and boundary-attach (*BE*, *BES*), we can find that the schemata resulted from boundary-attach have better P and F_1 results. We think this may be because *BII* and *BIES* connect all words in arguments to predicates. So the final graph contains much more edges than that generated by *BE* and *BES*. Therefore, given two words, the corresponding model tends to build an edge between them, compared with not building an edge, resulting in a higher R but a lower P. Second, by comparing schemata that with and without "S−r", i.e., *BII* vs. *BIES* and *BE* vs. *BES*, we find that it is always better to use a separate "S−r" to label the edge corresponding to the single-word argument. Therefore, from the overall point of view, we can get the conclusion that *BES* > *BE* > *BIES* > *BII*.

Performance regarding argument width. Here we define an argument's width as the number of words included. As we know, different schemata have different attaching and labeling methods to represent arguments of the same width. Therefore, analyzing the performance of different schemata on the same width argument will help us to deeply explore the advantages and disadvantages of schemata.

As shown in Figure 6, we divide arguments into four categories according to their width, and report F_1 values for each category. The proportion of each category in the gold-standard data is also reported. First, we can see that *BES* and *BIES* perform much better on 1-width arguments. This further shows that it is necessary to use "S−r" alone to represent arguments of width 1. Then, we can clearly find that *BE* and *BES* perform better than *BII* and *BIES* on arguments containing multiple words. And we know that *BE* and *BES* are resulted from the boundary-attach strategy which pays more attention to boundary information. So, we may con-

[4] http://www.cs.upc.edu/~srlconll/st05/st05.html

Model	Type	Sents/sec
He et al. (2018)	SGP	44
Strubell et al. (2018)	BIO-based	45
Zhang et al. (2022)	TP	214
Zhang et al. (2022)$_{\text{BERT}}$		113
Ours	WGP	**669**
Ours$_{\text{BERT}}$		252

Table 2: Speed comparison on CoNLL05. "SGP" and "WGP"denote the span-based graph parsing and word-based graph parsing approach respectively; "TP" means the tree parsing approach.

clude that boundary information is more helpful to the recognition of multi-word arguments.

Through analyzing the performance of different schemata, we find that *BES* is more suitable for converting span-based SRL into word-based graphs than other schemata. So, the rest of the experiments are conducted in *BES* schema.

6.2 Efficiency

Table 2 compares different models in terms of decoding speed. For fair comparison, we re-run all previous models on the same GPU environment (Nvidia GeForce 1080 Ti 11G). The results are averaged over 3 runs. In terms of batch size during evaluation, our model and Strubell et al. (2018) use 5000 tokens (about 134 sentences), while He et al. (2018) and Li et al. (2019) use 40 sentences by default.

We can see that our model improves the efficiency of previous span-based SRL models by large margin. Compared with the span-based graph parsing approach (He et al., 2018; Li et al., 2019), our graph-based parser only has a $O(n^2)$ search space. As for the BIO-based model of Strubell et al. (2018), the encoder contains 12 self-attention layers, and they adopts a pipeline framework by first predicting all predicates via sequence labeling and then recognizing arguments, leading to its low parsing speed. And when augmented with BERT, our methods can still parse about 250 sentences per second.

As discussed in Section 2, Zhang et al. (2022) reduce the SRL to a tree parsing task and get good results. However, they have to build a dependency tree for each predicate, which greatly reduces the efficiency of their approach. Specifically, the speed of our model is respectively three and twice times as fast as theirs under the setting without and with BERT.

6.3 Comparison with previous results

End-to-end. Our work mainly focuses on the *end-to-end* setting, i.e, requiring predicting predicates and arguments simultaneously. So we first go into this scenario. The first part of the Table 3 shows the comparison with previous works under the *end-to-end* setting.

First, when compared with models without PLMs, our model surpasses previous approaches with the large gap, getting comparable results with recently released work (Zhang et al., 2022). Then, most previous works usually use ELMo to improve the performance. In order to make a fair comparison, we also report the results with ELMo. We can find that our model also reaches better results, with +0.25 F_1 on WSJ, +0.77 F_1 on Brown, and +0.44 F_1 on CoNLL12-test when using ELMo. And when augmented with the more powerful PLM BERT, the performance of our model can be further improved. It shows that our method not only has high efficiency, but also performs better than previous works.

As discussed in Section 2, please kindly notice that Strubell et al. (2018) and Zhou et al. (2020) use extra syntactic knowledge to boost SRL performance. We only list their syntax-agnostic results here for fair comparison. It is worth noting that Strubell et al. (2018) incidentally wrongly used the official script in the end-to-end setting, leading to much higher precision scores. We reported this issue to their github repository and they confirmed this mistake. In this work, we report their results by evaluating their released models with the correct evaluation process.

Predicate-given. Recent works (Jindal et al., 2020; Zhang et al., 2021; Blloshmi et al., 2021) usually assume that predicates have been given, thus they only need to recognize the arguments and semantic roles. To compare with these works, we also report the results under the *predicate-given* setting. In our work, following Cai et al. (2018), during training procedure, the model is informed which word is the predicate using a predicate embedding. The embedding is added to the input vector.

Finally, from the second part of the Table 3, we can see that our model reaches the best results on most test datasets when compared with models without PLMs. When it comes to models with PLMs, we can see that the BIO-based Shi and Lin (2019) is a strong baseline. Our model lags behind

Model		CoNLL05-WSJ				CoNLL05-Brown			CoNLL12			
		Dev.F_1	P	R	F_1	P	R	F_1	Dev.F_1	P	R	F_1
The end-to-end setting												
He et al. (2017)†		80.3	80.2	82.3	81.2	67.6	69.6	68.5	75.5	78.6	75.1	76.8
Strubell et al. (2018)† *		81.72	81.77	83.28	82.51	68.58	70.10	69.33	-	-	-	-
He et al. (2018)‡		81.6	81.2	83.9	82.5	69.7	71.9	70.8	79.4	79.4	80.1	79.8
Li et al. (2019)‡		-	-	-	83.0	-	-	-	-	-	-	-
Zhou et al. (2020)		82.27	-	-	-	-	-	-	-	-	-	-
Zhang et al. (2022)		**83.91**	83.26	**86.20**	**84.71**	70.70	**74.16**	**72.39**	**81.16**	79.27	**83.24**	**81.21**
Ours		83.17	**85.28**	83.66	84.46	**74.10**	70.76	**72.39**	80.79	**82.10**	79.76	80.91
Strubell et al. (2018)† *	+ ELMo	84.73	83.86	85.98	84.91	73.01	75.61	74.31	-	-	-	-
He et al. (2018)‡	+ ELMo	85.3	84.8	87.2	86.0	73.9	78.4	76.1	83.0	81.9	84.0	82.9
Li et al. (2019)‡	+ ELMo	-	85.2	87.5	86.3	74.7	78.1	76.4	-	**84.9**	81.4	83.1
Ours	+ ELMo	85.72	86.19	86.91	86.55	76.57	77.77	77.17	83.72	83.53	83.56	83.54
Zhang et al. (2022)	+ BERT	**87.03**	87.00	**88.76**	**87.87**	79.08	**81.50**	**80.27**	**85.53**	**84.53**	**86.41**	**85.45**
Ours	+ BERT	86.79	**87.15**	88.44	87.79	**79.44**	80.85	80.14	84.74	83.91	85.61	84.75
The predicate-given setting												
He et al. (2017)†		81.6	83.1	83.0	83.1	72.9	71.4	72.1	81.5	81.7	81.6	81.7
Strubell et al. (2018)†		-	84.70	84.24	84.47	73.89	72.39	73.13	-	-	-	-
He et al. (2018)‡		-	-	-	83.9	-	-	73.7	-	-	-	82.1
Tan et al. (2018)†		83.1	84.5	85.2	84.8	73.5	74.6	74.1	82.9	81.9	83.6	82.7
Zhou et al. (2020)		83.16	-	-	-	-	-	-	-	-	-	-
Zhang et al. (2021)		84.45	85.30	85.17	85.23	74.98	73.85	74.41	82.83	83.09	83.71	83.40
Zhang et al. (2022)		**84.65**	85.47	**86.40**	**85.93**	74.92	**75.00**	74.96	83.39	83.02	**84.31**	83.66
Ours		84.39	**87.01**	84.36	85.66	**77.86**	72.53	**75.10**	**83.83**	**85.74**	82.95	**84.32**
Li et al. (2019)‡	+ ELMo	-	87.9	87.5	87.7	80.6	80.4	80.5	-	85.7	86.3	86.0
Shi and Lin (2019)†	+ BERT	-	88.6	89.0	88.8	81.9	82.1	82.0	-	85.9	87.0	86.5
Jindal et al. (2020)†	+ BERT	87.1	87.7	88.0	87.9	80.3	80.1	80.2	86.6	86.3	86.8	86.6
Zhang et al. (2021)	+ BERT	87.38	87.70	88.15	87.93	81.52	81.36	81.44	86.27	86.00	86.84	86.42
Blloshmi et al. (2021)	+ BART	-	-	-	-	-	-	-	-	**87.8**	86.8	87.3
Zhang et al. (2022)	+ BERT	**88.05**	89.00	**89.03**	**89.02**	82.81	**82.35**	**82.58**	**87.52**	87.52	**87.79**	**87.66**
Ours	+ BERT	87.54	**89.03**	88.53	88.78	**83.22**	81.81	82.51	86.97	87.26	87.05	87.15

Table 3: Results on CoNLL05 and CoNLL12 datasets. We mark BIO-based models by † and span-based graph ones by ‡. For Strubell et al. (2018) and Zhou et al. (2020), we list their syntax-agnostic results to compare fairly. Moreover, we mark the results of Strubell et al. (2018) by * to indicate that we report corrected evaluation results after re-testing their released models.

them slightly on WSJ, but is much higher than them on other datasets. And even compared with recent seq-to-seq model (Blloshmi et al., 2021), which uses more powerful BART (Lewis et al., 2020), our model still has strong competitiveness.

Comparison with Zhang et al. (2022). As discussed in Section 2, Zhang et al. (2022) propose a tree parsing approach to span-based SRL, which also appears in COLING-2022. We can see that performance of our model is slightly inferior to theirs, possibly due to more careful hyper-parameter tuning according to personal discussion between the two first authors. For example, Zhang confirms that fine-tuning BERT for 20 iterations leads to higher performance, while we only did 10 iterations.

7 Conclusions

This paper proposes four new graph representation schemata for transforming raw span-based SRL structures to word-based graphs. Based on the schema, we cast the span-based SRL as a word-based graph parsing task and present a fast and accurate parser. Moreover, we propose a simple post-processing method based on constrained Viterbi to handle conflicts in the output graphs. Experiments show that our parser 1) is much more efficient than previous parsers, and can parse over 600 sentences per second; 2) reaches consistently better performance than previous results on CoNLL05, CoNLL12 datasets. The in-depth comparison between four schemata shows that the boundary information counts a lot when recognizing arguments. In addition, distinguishing single-word arguments from multi-words arguments can also improve the final performance. These clear findings may help researchers think about SRL from a new perspective in the future.

8 Acknowledgments

We thank our anonymous reviewers for their valuable suggestions. This work was supported by National Natural Science Foundation of China (No. 62176173 and 62076174) and a project funded by the Priority Academic Program Development of Jiangsu Higher Education Institutions.

References

Rexhina Blloshmi, Simone Conia, Rocco Tripodi, and Roberto Navigli. 2021. Generating senses and roles: An end-to-end model for dependency- and span-based semantic role labeling. In *Proceedings of IJCAI*, pages 3786–3793.

Jiaxun Cai, Shexia He, Zuchao Li, and Hai Zhao. 2018. A full end-to-end semantic role labeler, syntactic-agnostic over syntactic-aware? In *Proceedings of ACL*, pages 2753–2765.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. Bert: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of NAACL-HLT*, pages 4171–4186.

Timothy Dozat and Christopher D. Manning. 2018. Simpler but more accurate semantic dependency parsing. In *Proceedings of ACL*, pages 484–490.

Jan Hajič, Eva Hajičová, Jarmila Panevová, Petr Sgall, Ondřej Bojar, Silvie Cinková, Eva Fučíková, Marie Mikulová, Petr Pajas, Jan Popelka, Jiří Semecký, Jana Šindlerová, Jan Štěpánek, Josef Toman, Zdeňka Urešová, and Zdeněk Žabokrtský. 2012. Announcing Prague Czech-English Dependency Treebank 2.0. In *Proceedings of LREC*, pages 3153–3160.

Luheng He, Kenton Lee, Omer Levy, and Luke Zettlemoyer. 2018. Jointly predicting predicates and arguments in neural semantic role labeling. In *Proceedings of ACL*, pages 364–369.

Luheng He, Kenton Lee, Mike Lewis, and Luke Zettlemoyer. 2017. Deep semantic role labeling: What works and what's next. In *Proceedings of ACL*, pages 473–483.

Angelina Ivanova, Stephan Oepen, Lilja Øvrelid, and Dan Flickinger. 2012. Who did what to whom? A contrastive study of syntacto-semantic dependencies. In *Proceedings of LAW*, pages 2–11.

Ishan Jindal, Ranit Aharonov, Siddhartha Brahma, Huaiyu Zhu, and Yunyao Li. 2020. Improved semantic role labeling using parameterized neighborhood memory adaptation. *arXiv preprint arXiv:2011.14459*.

Guillaume Lample, Miguel Ballesteros, Sandeep Subramanian, Kazuya Kawakami, and Chris Dyer. 2016. Neural architectures for named entity recognition. In *Proceedings of NAACL-HLT*, pages 260–270.

Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Veselin Stoyanov, and Luke Zettlemoyer. 2020. BART: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. In *Proceedings of ACL*, pages 7871–7880.

Zuchao Li, Shexia He, Hai Zhao, Yiqing Zhang, Zhuosheng Zhang, Xi Zhou, and Xiang Zhou. 2019. Dependency or span, end-to-end uniform semantic role labeling. In *Proceedings of AAAI*, pages 6730–6737.

Zuchao Li, Hai Zhao, Rui Wang, and Kevin Parnow. 2020. High-order semantic role labeling. In *Findings of EMNLP*, pages 1134–1151.

Ding Liu and Daniel Gildea. 2010. Semantic role features for machine translation. In *Proceedings of COLING*, pages 716–724.

Diego Marcheggiani, Jasmijn Bastings, and Ivan Titov. 2018. Exploiting semantics in neural machine translation with graph convolutional networks. In *Proceedings of NAACL-HLT*, pages 486–492.

Yusuke Miyao and Jun'ichi Tsujii. 2004. Deep linguistic analysis for the accurate identification of predicate-argument relations. In *Proceedings of COLING*.

Stephan Oepen, Marco Kuhlmann, Yusuke Miyao, Daniel Zeman, Silvie Cinková, Dan Flickinger, Jan Hajic, and Zdenka Uresova. 2015. Semeval 2015 task 18: Broad-coverage semantic dependency parsing. In *Proceedings of SemEval*, pages 915–926.

Stephan Oepen, Marco Kuhlmann, Yusuke Miyao, Daniel Zeman, Dan Flickinger, Jan Hajič, Angelina Ivanova, and Yi Zhang. 2014. SemEval 2014 task 8: Broad-coverage semantic dependency parsing. In *Proceedings of SemEval*, pages 63–72.

Martha Palmer, Daniel Gildea, and Paul Kingsbury. 2005. The proposition bank: An annotated corpus of semantic roles. *Computational linguistics*, 31(1):71–106.

Jeffrey Pennington, Richard Socher, and Christopher Manning. 2014. GloVe: Global vectors for word representation. In *Proceedings of EMNLP*, pages 1532–1543.

Matthew E. Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer. 2018. Deep contextualized word representations. In *Proceedings of NAACL-HLT*, pages 2227–2237.

Sameer Pradhan, Alessandro Moschitti, Nianwen Xue, Olga Uryupina, and Yuchen Zhang. 2012. CoNLL-2012 shared task: Modeling multilingual unrestricted coreference in OntoNotes. In *Proceedings of EMNLP-CoNLL*, pages 1–40.

Peng Shi and Jimmy Lin. 2019. Simple bert models for relation extraction and semantic role labeling. *arXiv preprint arXiv:1904.05255*.

Emma Strubell, Patrick Verga, Daniel Andor, David Weiss, and Andrew McCallum. 2018. Linguistically-informed self-attention for semantic role labeling. In *Proceedings of EMNLP*, pages 5027–5038.

Zhixing Tan, Mingxuan Wang, Jun Xie, Yidong Chen, and Xiaodong Shi. 2018. Deep semantic role labeling with self-attention. In *Proceedings of AAAI*.

Hai Wang, Mohit Bansal, Kevin Gimpel, and David McAllester. 2015. Machine comprehension with syntax, frames, and semantics. In *Proceedings of ACL-IJCNLP*, pages 700–706.

Xinyu Wang, Jingxian Huang, and Kewei Tu. 2019. Second-order semantic dependency parsing with end-to-end neural networks. In *Proceedings of ACL*, pages 4609–4618.

Qingrong Xia, Zhenghua Li, Min Zhang, Meishan Zhang, Guohong Fu, Rui Wang, and Luo Si. 2019. Syntax-aware neural semantic role labeling. In *Proceedings of AAAI*, pages 7305–7313.

Yu Zhang, Zhenghua Li, and Min Zhang. 2020. Efficient second-order TreeCRF for neural dependency parsing. In *Proceedings of ACL*, pages 3295–3305.

Yu Zhang, Qingrong Xia, Shilin Zhou, Yong Jiang, Guohong Fu, and Min Zhang. 2022. Semantic role labeling as dependency parsing: Exploring latent tree structures inside arguments. In *Proceedings of COLING*.

Zhisong Zhang, Emma Strubell, and Eduard Hovy. 2021. Comparing span extraction methods for semantic role labeling. In *Proceedings of SPNLP*, pages 67–77.

Jie Zhou and Wei Xu. 2015. End-to-end learning of semantic role labeling using recurrent neural networks. In *Proceedings of ACL-IJCNLP*, pages 1127–1137.

Junru Zhou, Zuchao Li, and Hai Zhao. 2020. Parsing all: Syntax and semantics, dependencies and spans. In *Findings of EMNLP*, pages 4438–4449.

Appendices

A More examples of other schemata

In order to save space, we only show the transition matrix of *BES* in the main body. Here, we present the transition matrix of others in Figure 7. In addition, we provide more examples to improve the comprehensibility of our schemata and the constrained viterbi. Figure 8, 9, and 10 show the outputs of models using different schemata respectively. For example, in Figure 8, there is missing an edge from "pilling" to "falling" in the raw output. After the viterbi procedure, an `I-AM-ADV` edge will be added since we disallow the transition from `O` to `I-*`. Thus we can get the legal SRL structure.

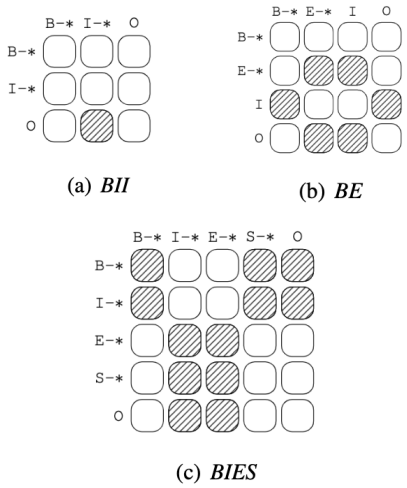

Figure 7: Transition matrices of *BII*, *BE*, and *BIES*.

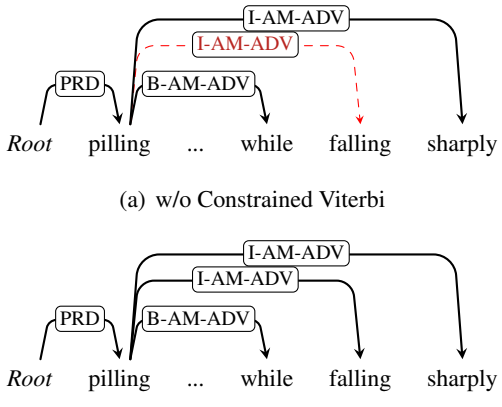

Figure 8: *BII* schema.

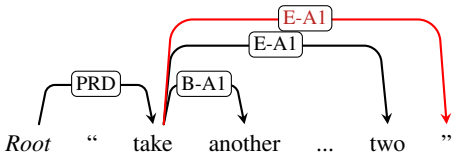

(a) w/o Constrained Viterbi

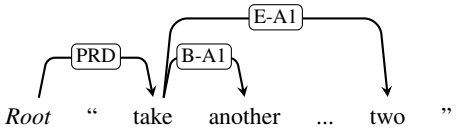

(b) w/ Constrained Viterbi

Figure 9: *BE* schema.

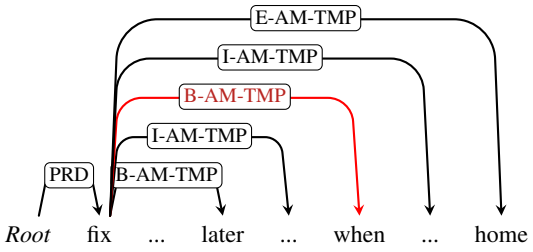

(a) w/o Constrained Viterbi

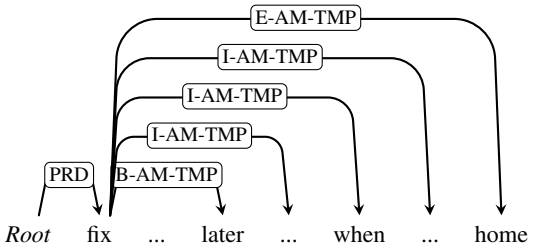

(b) w/ Constrained Viterbi

Figure 10: *BIES* schema.

Unsupervised Lexical Substitution with Decontextualised Embeddings

Takashi Wada[1,2] **Timothy Baldwin**[1,3] **Yuji Matsumoto**[2] **Jey Han Lau**[1]
[1] School of Computing and Information Systems, The University of Melbourne
[2] RIKEN Center for Advanced Intelligence Project (AIP)
[3] Department of Natural Language Processing, MBZUAI
twada@student.unimelb.edu.au, tb@ldwin.net
yuji.matsumoto@riken.jp, jeyhan.lau@gmail.com

Abstract

We propose a new unsupervised method for lexical substitution using pre-trained language models. Compared to previous approaches that use the generative capability of language models to predict substitutes, our method retrieves substitutes based on the similarity of contextualised and *decontextualised* word embeddings, i.e. the average contextual representation of a word in multiple contexts. We conduct experiments in English and Italian, and show that our method substantially outperforms strong baselines and establishes a new state-of-the-art without any explicit supervision or fine-tuning. We further show that our method performs particularly well at predicting low-frequency substitutes, and also generates a diverse list of substitute candidates, reducing morphophonetic or morphosyntactic biases induced by article–noun agreement.[1]

1 Introduction

There has been growing interest in developing automatic writing support systems to assist humans to write documents. One relevant task to this research goal is lexical substitution, where given a target word and its surrounding context, a system suggests a list of word substitutions that can replace the target word without changing its core meaning. For instance, given the target word *great* and the context *He is a great artist*, the model might suggest alternative words such as *outstanding*, *terrific*, or *distinguished*. Writers can use such suggestions to improve the fluency of their writing, reduce lexical repetition, or search for better expressions that represent their ideas more creatively.

As with other NLP tasks, recent studies have shown that masked language models such as BERT (Devlin et al., 2019) perform very well on lexical substitution, even without any task-specific supervision. A common approach is to employ language models as generative models and predict substitutes based on their generative capability. However, this approach has some limitations. First, it is extremely difficult for language models to predict rare words — especially those that are segmented into multiple subword tokens — since the models inevitably assign them very low probabilities. Second, word prediction is highly affected by morphosyntactic constraints from the surrounding context, which overshadows the (arguably more important) question of semantic fit. For instance, if the target word is *increase* in the context ... *with an increase in* ..., language models tend to suggest words that also start with a vowel sound due to the presence of *an* before the target word, missing other possible substitutes such as *hike* or *boost*. In fact, this problem is even more pronounced in languages where words have grammatical gender (e.g. Italian nouns) or a high degree of inflection (e.g. Japanese verbs).

In this paper, we propose a new approach that explicitly deals with these limitations. Instead of generating words based on language model prediction, we propose to find synonymous words based on the similarity of contextualised and *decontextualised* word embeddings, where the latter refers to the "average" contextual representation of a word in multiple contexts. Experiments on English and Italian lexical substitution show that our fully unsupervised method outperforms previous models by a large margin. Furthermore, we show that our model performs particularly well at predicting low-frequency words, and also generates more diverse substitutes with less morphophonetic or morphosyntactic bias, e.g. as a result of article–noun agreement in English and Italian.

2 Method

2.1 Our Approach

Given a sentence that contains a target word x and its surrounding context c, we first feed the sentence

[1]Code is available at: https://github.com/twadada/lexsub_decontextualised.

Proceedings of the 29th International Conference on Computational Linguistics, pages 4172–4185
October 12–17, 2022.

into a pre-trained transformer model (Vaswani et al., 2017) such as BERT and generate the contextualised representations of x: $f^\ell(x,c) \in \mathbb{R}^d$, where ℓ ($\leq L$) denotes the layer of the model. We propose to predict substitutes of x by retrieving words that have similar representations to $f^\ell(x,c)$. To this end, we calculate $S(y|x,c)$: the score of y being a substitute of x in context c, as follows:

$$S(y|x,c) = \cos(f(y), f(x,c)), \quad (1)$$
$$f(x,c) = \sum_{\ell \in Z} f^\ell(x,c),$$
$$f(y) = \frac{1}{N} \sum_{i}^{N} \sum_{\ell \in Z} f^\ell(y, c'_i),$$

where $f(y) \in \mathbb{R}^d$ denotes the *decontextualised* word embedding of y; Z is a set of selected layers; and $\cos(a,b)$ denotes the cosine similarity between a and b. To obtain $f(y)$, we randomly sample N sentences $(c'_1, c'_2 ..., c'_N)$ that contain y from a monolingual corpus, and take the average of the contextualised representations of y given c'_i: $f^\ell(y, c'_i)$. We pre-compute $f(y)$ for each word y in our pre-defined vocabulary $\tilde{V}$, which consists of *lexical items* (i.e. no subwords) and contains different words from the pretrained model's original vocabulary V. If y is segmented into multiple subwords (using the pretrained model's tokeniser), we average its subword representations — this way we can include low-frequency words in $\tilde{V}$ and generate diverse substitutes. We obtain $f(x,c)$ and $f(y)$ by summing representations across different layers $\ell \in Z$ to capture various lexical information.[2]

2.2 Multi-Sense Embeddings

Representing $f(y)$ in Eqn. (1) as a simple average of the contextualised representations of y is clearly limited when y has multiple meanings, since the representations will likely vary depending on its usage. For instance, Wiedemann et al. (2019) show that BERT representations of polysemous words such as *bank* create distinguishable clusters based on their usages. To address this issue, we first group the N sentences into K clusters based on the usages of y, and for each cluster k, we obtain the decontextualised embedding $f^k(y)$ by averaging the contextualised representations, i.e.,

$$f^k(y) = \frac{1}{|C^k|} \sum_{c' \in C^k} \sum_{\ell \in Z} f^\ell(y, c'),$$

where C^k denotes the set of the sentences that belong to the cluster k. To obtain clusters, we apply K-means (Lloyd, 1982; Arthur and Vassilvitskii, 2007) to the L2-normalised representations of y in N sentences.[3] We expect that if y has multiple senses, $f^k(y)$ will to some degree capture the different meanings.[4] This methodology has been shown to be effective by Chronis and Erk (2020) on context-independent word similarity tasks. With $f^k(y)$, we can refine the similarity score $S(y|x,c)$ in Eqn. (1) as follows:

$$S(y|x,c) = \max_k \cos(f^k(y), f(x,c)).$$

In this way, we can compare x with y based on the sense that is most relevant to x. Furthermore, we capture *global similarity* between x and y as:

$$S(y|x,c) = \max_k \lambda \cos(f^k(y), f(x,c)) + (1-\lambda)\cos(f^k(y), f^{j_c}(x)), \quad (2)$$
$$j_c = \operatorname*{argmax}_j \cos(f^j(x), f(x,c)), \quad (3)$$

where the second term in Eqn. (2) corresponds to the global similarity, which compares x and y outside of context c.[5] However, it still considers c in Eqn. (3) to retrieve the cluster that best represents the meaning of x given c.

While Eqn. (2) generally generates high-quality substitutes, we found that it sometimes retrieves words that share the same root word as x and yet do not make good substitutes (e.g. *pay* and *payer*). This is mainly due to the fact that the vocabulary $\tilde{V}$ contains a large number of derivationally-related words, some of which are out-of-vocabulary (OOV) in the original vocabulary V (e.g. *pay ##er*). To address this problem, we add a simple heuristic where y is discarded if the normalised edit distance[6] between x and y is less than a threshold (0.5 for our English and Italian experiments).[7]

2.3 Reranking

In Eqn. (2), the context c affects the representation of x but not y. Ideally, however, we want to con-

[2] We also tried taking the weighted sum of the different-layer embeddings, but we did not see noticeable improvement.

[3] We concatenate $f^\ell(y,c)$ across multiple layers $\ell \in Z$.

[4] Note that the number of clusters K is fixed across all words, forcing the model to "split" and "lump" senses (Hanks, 2012) to varying degrees.

[5] To obtain $f^j(x)$, we compute the decontextualised embedding of x and apply K-means, as we do to compute $f^k(y)$. When x is not included in our pre-defined vocabulary $\tilde{V}$, we set λ to 1 and ignore the second term in Eqn. (2).

[6] The distance normalised by the maximum string length.

[7] We tuned this threshold based on English development data (i.e. the development split of SWORDS).

sider the context c on both sides to find the words that best fit the context. Therefore, we first generate top-M candidates based on Eqn. (2), and rerank them using the following score:

$$\mathrm{S(y|x,c)} = \frac{1}{|Z|} \sum_{\ell \in Z} \cos(f^{\ell}(y,c), f^{\ell}(x,c)), \quad (4)$$

where $f^{\ell}(y,c)$ denotes the contextualised representation of y given c, which can be obtained by replacing x in c with y and feeding it into the model. In Eqn. (4), we calculate the similarity at each layer $\ell \in Z$ and take the average, which yields small yet consistent improvements over averaging the embeddings first and then calculating the similarity.[8] We limit the use of this scoring method to the M candidates only, since it is computationally expensive to calculate $f^{\ell}(y,c)$ for every single word y in $\tilde{V}$. Previously, a similar method was employed by Lee et al. (2021) but they used the last layer only (i.e. $Z = \{L\}$). We show that using multiple layers substantially improves the performance. Following Lee et al. (2021), we set M to 50.

2.4 Comparison to Previous Approaches

Our approach contrasts with previous approaches (Zhou et al., 2019; Lee et al., 2021; Yang et al., 2022) that employ BERT as a generative model and predict lexical substitutes based on the generation probability $P(y|x,c)$:

$$P(y|x,c) = \frac{\exp(E_y f^{\hat{L}}(x,c) + b_y)}{\sum_{\acute{y} \in V} \exp(E_{\acute{y}} f^{\hat{L}}(x,c) + b_{\acute{y}})}, \quad (5)$$

where $E_y \in \mathbb{R}^d$ denotes the output embedding of y, which is usually tied with the input word embedding; $f^{\hat{L}}(x,c)$ is the representation at the very last layer of the model;[9,10] and b_y is a scalar bias. While this approach is straightforward and well motivated, its predictions are highly influenced by morphosyntax, as discussed in Section 1. Moreover, Eqn. (5) shows three additional limitations compared to our approach: (1) the prediction is conditioned on the last layer only, despite previous studies showing that different layers capture different information, with the last layer usually containing less semantic information than the lower or middle layers (Bommasani et al., 2020; Tenney et al., 2019); (2) y is represented by the single vector E_y, which may not work well when y has multiple meanings — we alleviate this by clustering the embeddings (Section 2.2); and (3) the model is not capable of generating OOV words, unless we force the model to decode multiple subwords, e.g. by using multiple mask tokens or duplicating x. Our approach, in contrast, can include rare words in the pre-defined vocabulary $\tilde{V}$ and generate diverse substitutes (Section 4.4).

3 Experiments

3.1 Data and Evaluation

We conduct experiments in two evaluation settings: generation and ranking. In the generation setting, systems produce lexical substitutes given target words and sentences, while in the ranking setting, they are also given substitute candidates and rank them based on their appropriateness.

For the generation task, we base our experiments on SWORDS (Lee et al., 2021), the largest English lexical substitution dataset, which extends and improves CoInCo (Kremer et al., 2014) by introducing a new annotation scheme: in CoInCo, the annotators were asked to come up with substitutes by themselves, whereas in SWORDS, the annotators were given substitute candidates preretrieved from a thesaurus, and only had to made binary judgements ("good" or "bad").[11] A word is regarded as *acceptable* if it is judged to be good by more than five out of ten annotators, and *conceivable* if selected by at least one annotator. In this way, SWORDS provides more comprehensive lists of substitutes, including many low-frequency words that are good substitutes and yet difficult for humans to suggest — these words are of particular interest to us. For the evaluation metrics, the authors use the harmonic mean of the precision and recall given the gold and top-10 system-generated substitutes.[12] As gold substitutes, they use either the acceptable or conceivable words, and calculate the corresponding scores F_a and F_c, respectively.

[8] In Eqn. (1), we obtained similar results by averaging the embeddings or cosine similarities across layers.

[9] Note that this does not always correspond to the last layer of transformer: $f^{L}(x,c)$. E.g., BERT calculates $f^{\hat{L}}(x,c)$ by applying a feed forward network and layer normalisation to $f^{L}(x,c)$, whereas for XLNET, $f^{\hat{L}}(x,c) = f^{L}(x,c)$.

[10] When x consists of multiple subwords, the representation of the first or longest token is usually used.

[11] The annotators were asked if they would consider using the substitute candidate to replace the target word as the author of the context.

[12] More precisely, their evaluation script lemmatises the top-50 substitutes first and then extracts the top-10 distinct words.

They also propose to measure those scores in both ***strict*** and ***lenient*** settings, which differ in that in the lenient setting, candidate words that are not scored under SWORDS are filtered out and discarded.

In the ranking task, we evaluate models on the traditional SemEval-2007 Task 10 ("SemEval-07") data set (trial+test) (McCarthy and Navigli, 2007), as well as SWORDS. For the evaluation metric, we follow previous work in using Generalized Average Precision (GAP; Kishida (2005)):

$$GAP = \frac{\sum_{i=1}^{N} I(\alpha_i) p_i}{\sum_{i=1}^{R} I(\beta_i) \bar{\beta}_i}, \quad p_i = \frac{\sum_{k=1}^{i} \alpha_k}{i}, \quad (6)$$

where α_i and β_i denote the gold weight of the i-th item in the predicted and gold ranked lists respectively, with N and R indicating their sizes; $I(\alpha_i)$ is a binary function that returns 1 if $\alpha_i > 0$, and 0 otherwise; and $\bar{\beta}_i$ is the average weight of the gold ranked list from the 1st to the i-th items. In our task, the weight corresponds to the aptness of the substitute, which is set to zero if it is not in the gold substitutes. Following previous work (Melamud et al., 2015; Arefyev et al., 2020), we ignore multiword expressions in SemEval-07.[13]

3.2 Models

As shown in Eqn. (2), our approach requires only the vector representations of words and hence is applicable to any text encoder model. Therefore, we test our method with various pre-trained models, including five masked language models: BERT (Devlin et al., 2019), mBERT, SpanBERT (Joshi et al., 2020), XLNET (Yang et al., 2019), and MPNet (Song et al., 2020); one encoder-decoder model: BART (Lewis et al., 2020); and two discriminative models: ELECTRA (Clark et al., 2020) and DeBERTa-V3 (He et al., 2021).[14] We also evaluate two sentence-embedding models: MPNet-based sentence transformer (Reimers and Gurevych, 2019) and SimCSE (Gao et al., 2021), both of which are fine-tuned on semantic downstream tasks such as MNLI and achieve good performance on sentence-level tasks. Finally, we also evaluate the encoder of the fine-tuned mBART on English-to-Many translation (Tang et al., 2021). Note that the discriminative models and embedding models (e.g. NMT-encoder, SimCSE) cannot generate words and hence are incompatible with the previous approach described in Section 2.4.

[13]We run the evaluation code at `https://github.com/orenmel/lexsub` with the *no-mwe* option.

[14]See Appendix A for the details of all models.

Models	Lenient		Strict	
	F_a	F_c	F_a	F_c
HUMANS	*48.8*	*77.9*	–	–
CoInCo	*34.1*	*63.6*	–	–
GPT-3	**34.6**	49.0	22.7	36.3
BERT-K[15]	32.4	55.4	19.2	30.4
(w/o rerank)[15]	31.8	54.9	15.7	24.4
BERT-M	30.9	48.3	16.2	25.4
(w/o rerank)	30.9	48.1	10.7	16.5
Zhou et al. (2019)[15]	32.0	55.4	17.4	27.5
Yang et al. (2022)[16]	31.9	54.9	16.7	28.4
OURS				
BERT	33.2	64.1	21.1	34.9
(w/o rerank)	33.0	63.8	20.7	34.4
(w/o rerank, heuristic)	33.6	64.0	20.2	32.4
mBERT	27.0	52.7	12.4	22.6
SpanBERT	32.6	61.4	20.9	34.0
MPNet	33.8	63.8	22.0	34.1
XLNet	34.4	65.3	23.3	37.4
ELECTRA	33.5	64.2	23.2	36.7
DeBERTa-V3	33.6	**65.8**	**24.5**	**39.9**
BART (Enc)	33.6	62.8	21.9	34.8
BART (Dec)	33.5	60.5	21.4	34.0
BART (Enc-Dec)	33.7	64.9	23.5	37.2
SBERT (MPNet)	**34.6**	64.0	21.8	33.5
SimCSE (BERT)	33.4	64.3	21.6	35.7
NMT (mBART)	28.7	55.6	13.4	22.2
OURS (Rank Candidates)				
XLNet	35.2	72.9	–	–
BART	34.8	72.4	–	–
DeBERTa-V3	35.1	72.2	–	–

Table 1: The results for the generation task in the lenient and strict settings. The best scores are boldfaced.

We use the same vocabulary $\tilde{V}$ for all models, which consists of the 30,000 most common words[17] in the OSCAR corpus (Ortiz Suárez et al., 2020). We set the number of sentences we sample from OSCAR to calculate the decontextualised embeddings, i.e. N, to 300; the clustering size K to 4; and λ in Eqn. (2) to 0.7. For the set of transformer layers, Z, we employ all layers except for the first and

[15]These results differ slightly from the reported scores in Lee et al. (2021), due to a bug in their code.

[16]Updated from the original scores by the authors after they fixed some critical issues in their evaluation setup.

[17]We discard tokens that contain numerals, punctuation, or capital letters. As such, $\tilde{V}$ includes more lexical items (with less noise and no subwords) than the original vocabulary V.

last two, i.e. $\{3, 4, ..., L-2\}$. We tune all hyperparameters on the development split of SWORDS.

3.3 Results

Table 1 shows the results on SWORDS, along with baseline scores from previous work (with some bug fixes, as noted). The first row, **HUMANS**, indicates the agreement of two independent sets of annotators on (a subset of) SWORDS, and approximates the upper bound for this task. The second row, **CoInCo**, shows the accuracy of the gold standard substitutes in CoInCo, which are suggested by human annotators without access to substitution candidates[18] — this approximates how well humans perform when asked to elicit candidates themselves. The remainder of the rows above OURS denote baseline systems, all of which employ generative approaches. The first baseline uses **GPT-3** (Brown et al., 2020), and achieves the state-of-the-art in the strict setting. It generates substitutes based on "in-context learning", where the model first reads several triplets of target sentences, queries, and gold-standard substitutes retrieved from the development set, and then performs on-the-fly inference on the test set. As such, it is not exactly comparable to the other fully unsupervised models. **BERT-K** generates substitutes based on Eqn. (5) by feeding the target sentence into BERT, and **BERT-M** works the same except that the target word is replaced by [MASK]. Both models further rerank the candidates based on Eqn. (4), using the last layer only; we show the performance without reranking as "w/o rerank" in Table 1. Yang et al. (2022) and Zhou et al. (2019) also use BERT to generate substitutes, and rerank them using their own method.

The rows below OURS indicate the performance of our approach using various off-the-shelf models. Our method with BERT substantially outperforms all the BERT-based baselines, even without the edit-distance heuristic (Section 2.2) or reranking method (Eqn. (4)). The best performing models are DeBERTa-V3, XLNet, and BART (Enc-Dec), all of which outperform the weakly-supervised GPT-3 model by a large margin in the strict setting; and XLNet even outperforms CoInCo in the lenient setting. The last three rows show the performance of the top-3 models when they are given the candidate words and rank them based on Eqn. (4), which emulates how the SWORDS annotators judged the words. The result shows that all the models still lag behind HUMANS, suggesting there is still substantial room for improvement. Interestingly, BART performs best when we average the scores obtained by its encoder and decoder, suggesting each layer captures complementary information. It is also intriguing to see that the discriminative models (DeBERTa-V3 and ELECTRA) perform much better than BERT, albeit they are not trained to generate words and not compatible with the previous generative approach. The sentence embedding models (SBERT, SimCSE) perform no better than the original models, which contrasts with their strong performance in sentence-level tasks. The multilingual models (mBERT, NMT) perform very poorly, even though the NMT model was fine-tuned on large English-X parallel corpora.

Models	S-07	SW
HUMANS	—	*66.2*
Arefyev et al. (2020) (XLNet)	61.3[19]	—
Michalopoulos et al. (2022) (LMs+WN)	60.3	—
Lacerra et al. (2021a) (BERT)	58.2	—
Lacerra et al. (2021a) (BERT, sup)	60.5	—
Zhou et al. (2019) (BERT)	60.5[20]	53.5[15]
Lee et al. (2021) (BERT)	56.6	56.9
OURS (Eqn. (4))		
BERT	58.6	60.7
mBERT	45.4	52.0
SpanBERT	59.3	60.8
MPNet	61.5	59.5
XLNet	63.8	**62.9**
ELECTRA	64.4	62.3
DeBERTa-V3	**65.0**	**62.9**
BART (Enc)	62.9	61.9
BART (Dec)	62.6	60.8
BART (Enc-Dec)	64.1	62.7
SBERT (MPNet)	61.0	62.5
SimCSE (BERT)	58.4	60.9
NMT (mBART, Enc)	46.0	51.5

Table 2: GAP scores on SemEval-07 and SWORDS. "LMs+WN" employs multiple language models and WordNet, and "BERT, sup" is a supervised model.

Table 2 shows the results for the ranking task on SemEval-07 and SWORDS. Michalopoulos et al. (2022) harness WordNet (Fellbaum, 1998) to obtain synsets of the target word and also their glosses, and employ BERT and RoBERTa (Liu et al., 2019)

[18] Since all of these words are in the substitute candidates of SWORDS, it cannot be evaluated under the strict setting.

[19] The original score reported by Arefyev et al. (2020) is 59.6, but we found we could improve this result by appending unscored OOV words to the ranked list in random order.

[20] Similar to Lacerra et al. (2021a) and Arefyev et al. (2020), we were unable to reproduce this score.

to rank candidates. The models proposed by Lacerra et al. (2021a) are different from the others in that they fine-tune BERT on lexical substitution data sets. They propose unsupervised (BERT) and supervised (BERT, sup) models, which are fine-tuned on automatically-generated or manually-annotated data. Table 2 shows that our method with BERT performs comparably with the unsupervised model of Lacerra et al. (2021a) without any fine-tuning, and outperforms Zhou et al. (2019) and Lee et al. (2021) (except for the score of Zhou et al. (2019) on SemEval-07, which couldn't be reproduced in previous work). Just like the generation task, DeBERTa-V3 achieves the best performance on both data sets and establishes a new state-of-the-art. Other models also follow a similar trend to the generation results, e.g. BART performs best by combining its encoder and decoder, and the multilingual models perform very poorly. We hypothesise that their poor performance is mainly caused by suboptimal segmentation of English words. This hypothesis is also supported by the fact that DeBERTa-V3 has by far the largest vocabulary V of all models.[21]

3.4 Results on Italian Lexical Substitution

We further conduct an additional experiment on Italian, based on the data set from the EVALITA 2009 workshop (Toral, 2009). We report F scores given top-10 predictions as in the English generation task, plus two traditional metrics used in the workshop, namely *oot* and *best*, which compare the top-10 and top-1 predictions against the gold substitutes.[22] We lemmatise all the generated words to make them match the gold substitutes, following the SWORDS evaluation script.[23] We use the same hyper-parameters as for the English experiments, and Table 3 shows the results. Hintz and Biemann (2016) is a strong baseline that retrieves substitute candidates from MultiWordNet (Pianta et al., 2002) and ranks them using a supervised ranker model. We also implement BERT-K using an Italian BERT model (Schweter, 2020), with and without the reranking method. The results show that our approach substantially outperforms the baselines, confirming its effectiveness. However, our reranking method is not as effective as in English, which we attribute to the influence of grammatical gender in Italian (which we return to in Section 4.2). The heuristic improves best-P but harms F and oot-P, meaning it removes good candidates as well as bad ones, possibly because we used the threshold tuned on English.

[21] Note that V differs from $\tilde{V}$, the pre-defined vocabulary we used for all models. Appendix A compares the size of the model's original vocabulary V across different models.

[22] We report precision only, as it is the same as recall under those metrics when predictions are made for every sentence.

[23] We used the Italian lemmatiser (it_core_news_sm 3.2.0) in spaCy (ver. 3.2.2) (Honnibal et al., 2020).

	F	best-P	oot-P
Hintz and Biemann (2016)	—	16.2	41.3
BERT-K	14.3	14.4	39.1
(w/o rerank)	15.6	17.4	43.3
OURS (BERT)	17.3	19.9	47.5
(w/o rerank)	17.5	19.1	48.4
(w/o rerank, heuristic)	17.5	17.4	48.7
OURS (ELECTRA)	19.0	21.0	51.2
(w/o rerank)	18.9	**21.3**	51.0
(w/o rerank, heuristic)	**19.2**	20.2	**52.1**

Table 3: The result of Italian lexical substitution.

	BERT	BART	XLNet	DeBERTa
$\lambda = 1$	34.1	26.2	32.9	35.8
$\lambda = 0$	32.8	34.1	34.1	35.6
$K = 1$	32.9	32.0	33.7	35.7
k is random	30.6	29.2	30.3	32.0
w/o heuristic	32.4	32.0	33.4	35.7
$\lambda = 0.7$	34.4	34.0	35.0	36.9
+ rerank	**34.9**	**37.2**	**37.4**	**39.9**

Table 4: Ablation studies of our method. The scores denote F_c in the strict setting on SWORDS.

4 Analysis

4.1 Ablation Studies

We perform ablation studies on SWORDS to see the effect of λ and the K-clustered embeddings, and also the heuristic based on edit distance. Table 4 shows the results. Overall, our method with $\lambda = 0.7$ performs better than $\lambda = 1$ or $\lambda = 0$, confirming the benefit of considering both in-context and out-of-context similarities. One interesting observation is that while BERT and DeBERTa perform better with $\lambda = 1$ than with $\lambda = 0$, the opposite trend is observed for XLNet and especially BART (and hence the optimal value for λ is smaller than 0.7). This suggests that BART representations are highly influenced by context, containing much information that is not relevant to the semantics

	a	*an*	*un*	*una*	*la/le*	*il/i*
BERT-K	94.2	56.0	93.3	92.5	93.8	92.0
(w/o r)	91.1	54.0	90.0	87.5	90.3	87.4
OURS (BERT)	88.4	24.0	81.9	89.2	92.6	85.2
(w/o r)	86.8	18.0	70.5	66.7	69.7	64.6
(w/o r, h)	86.8	44.0	62.4	65.0	69.4	60.7
Gold	86.8	26.9	55.6	63.3	69.0	67.0

Table 5: The percentage of generated and gold substitutes whose initial sound or grammatical gender agrees with the corresponding English or Italian articles. "r" and "h" denote rerank and heuristic, respectively. The closest numbers to Gold are underlined.

Models	*a*	*an*
BERT-K	94.2	56.0
BERT-M	99.5	88.0
BERT	88.4	24.0
SpanBERT	97.9	46.0
MPNet	91.6	52.0
XLNet	83.2	38.0
BART (Enc)	89.5	62.0
BART (Dec)	91.1	46.0
BART (Enc-Dec)	89.5	58.0
DeBERTa-V3	86.8	32.0
ELECTRA	88.4	36.0

Table 6: The percentage of substitutes whose initial sound agrees with the corresponding English articles.

of the target word; we further confirm this in the next section. When we set the cluster size K to 1, the performance of all the models drops sharply, indicating the effectiveness of the clustered embeddings. When we retrieve the cluster of y at random instead of the closest one to x in Eqn. (2), the performance decreases substantially, suggesting each cluster captures different semantics. The heuristic consistently improves the performance, filtering out derivationally-related yet semantically-dissimilar words to the target word. Lastly, our reranking method substantially improves the performance of all the models, demonstrating that it is important to incorporate the target context c into both the target and candidate word representations.

4.2 Effects of Morphosyntactic Agreement

Compared to previous generative approaches, our method does not depend on the generation probabilities of language models, and hence we expect it to be less sensitive to morphosyntactic agreement effects. To investigate this, we analyse the performance on noun target words which immediately follow one of the following articles: *a* or *an* in English, and *una*, *la*, *le*, *un*, *il* or *i* in Italian. The first three Italian articles are used with feminine nouns, and the rest with masculine ones. Our hypothesis is that generative methods will be highly biased by these articles, despite the gold standard being *semantically* annotated, and thus largely oblivious to local morphosyntactic agreement effects.

Table 5 shows the percentage of top-10 predicted candidates that agree with the article.[24] It demonstrates that the prediction of BERT-K is highly affected by the proceeding article as expected, resulting in substitutes which don't satisfy this constraint being assigned low probabilities. In contrast, the results of our method are more balanced and close to the gold standard.[25] Conversely, our reranking method actually *increases* the bias greatly, suggesting that the contextualised embeddings $f^\ell(x, c)$ and $f^\ell(y, c)$ in Eqn. (4) become similar when x and y collocate similarly with the words in the context c — overall, this leads to better results, but actually hurts in cases of local agreement effects biasing the results. This is one reason why reranking was not as effective in Italian as in English, as agreement effects are stronger in Italian.

Table 6 shows the result when we use different pre-trained models in English. First, it shows that BERT-M is more sensitive to the articles than BERT-K, indicating the strong morphophonetic agreement effect on the masked word prediction. Among the pre-trained language models used by our method, SpanBERT and BART are the most sensitive to the article *a* and *an*, respectively. This suggests that the embeddings $f(x, c)$ obtained from these models are highly sensitive to the context c, partly explaining why BART performs very poorly with $\lambda = 1$, as shown in Section 4.1. Lastly, Table 7 shows examples of predicted substitutes when the article *an* comes before the target word. It shows that BERT-K and OURS with BART tend to retrieve words that start with a vowel sound, as quan-

[24] We retrieve Italian gender information using a dictionary API (https://github.com/sphoneix22/italian_dictionary), and English phonetic information using CMUdict (https://github.com/cmusphinx/cmudict) accessed via NLTK (Steven et al., 2009).

[25] Note that the big jump in results for *an* is based on a small number of instances (5 sentences).

Context	The loan may be extended by the McAlpine group for an additional year with an **increase** in the conversion price to $2.50 a share.
Gold (Conceivable)	*boost*, *gain*, *raise*, *hike*, *rise*, swell, surge, upsurge, enlargement, growth, addition, escalation, expansion, upgrade, cumulation, swelling, exaggeration, step-up
BERT-K	increased, **rise**, enhancement, increasing, **addition**
OURS (BERT)	**rise**, **raise**, change, reduce, reduction
OURS (BART)	**rise**, uptick, **hike**, improvement, upping
OURS (DeBERTa-V3)	**boost**, **rise**, **raise**, **hike**, reduction
Context	Under an **accord** signed yesterday, the government and Union Bank of Finland would become major shareholders in the new company, each injecting 100 million Finnish markkaa ($23.5 million).
Gold (Conceivable)	*arrangement*, *agreement*, *pact*, contract, deal, treaty
BERT-K	understanding, **agreement**, **pact**, **arrangement**, agreeing
OURS (BERT)	**agreement**, **pact**, **treaty**, understanding, **deal**
OURS (BART)	**agreement**, **deal**, **pact**, **arrangement**, **treaty**
OURS (DeBERTa-V3)	**pact**, **agreement**, **deal**, **arrangement**, **treaty**

Table 7: Top-5 predictions when the article *an* comes before the target word (*increase* or *accord*). Gold shows a list of "conceivable" words sorted by their annotated scores (with "acceptable" words shown in italic, and multiword expressions omitted from the table). Words included in Gold are boldfaced.

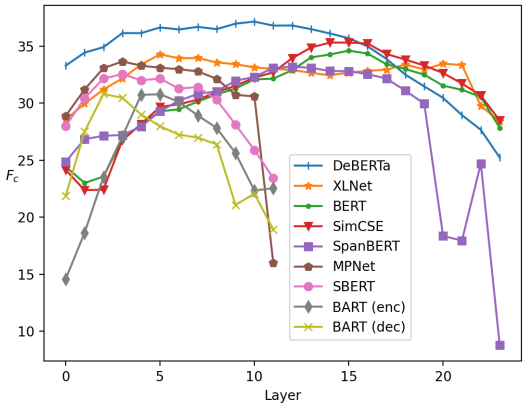

Figure 1: Layer-wise performance (F_c) on SWORDS.

titatively described in Table 6.

4.3 Layer-Wise Performance

We analyse the performance on SWORDS using different layers in Figure 1 (w/o rerank).[26] First, we clearly see that middle layers perform better than the first or last ones, for all models.[27] The performance of BERT peaks at layer 16, in contrast with previous findings that the first quarter of layers perform best on context-independent word similarity tasks (Bommasani et al., 2020), likely because lexical substitution critically relies on context.[28] Our method using multiple layers performs mostly as well as using the best layer without the need to perform model-wise layer selection (see Table 11 in Appendix C). The last layer performs very poorly for all models, highlighting the limitation of the previous approach which uses the last layer only (Eqn. (5)). The downward trend is particularly evident for MPNet, BART (dec), and SpanBERT; for BART and SpanBERT, we attribute this to the fact that their last-layer representations of the word at position t are used to predict the next word w_{t+1}, or u neighbouring words $\{w_{t-u}..,w_{t-1}\}$ or $\{w_{t+1}..,w_{t+u}\}$.[29] This training objective may also lead to their sensitivity to articles before the target word, as shown in Section 4.2. Interestingly, the sentence-embedding models (SBERT and SimCSE) are no exception to the downward trend, which is somewhat counter-intuitive given that their last layer representations are fine-tuned (and used during inference) to perform semantic downstream tasks. Importantly, they do not perform better than the original models (MPNet and BERT), although in the ranking task, both models benefit moderately from fine-tuning (see Appendix B).

[26] We perform qualitative analysis in Appendix D.

[27] We see a similar trend in the ranking task (Appendix B).

[28] In fact, Tenney et al. (2019) show that high-level semantic information is encoded in higher layers.

[29] SpanBERT does this for Span Boundary Objective.

	# Matched Words			F_c
	low	med	high	
BERT-K	121	144	2002	30.4
BERT (5k)	31	52	2017	28.1
BERT (10k)	72	111	2249	32.6
BERT (20k)	164	220	2194	34.5
BERT (30k)	241	267	2095	34.9
BART (30k)	380	274	2123	37.2
XLNet (30k)	395	**292**	2106	37.4
DeBERTa (30k)	**429**	287	**2262**	**39.9**

Table 8: The number of correctly predicted substitutes, grouped by their frequency in monolingual data. The number in brackets shows the size of the vocabulary $\tilde{V}$.

K	1	2	4	8	16
F_a	22.8	23.1	23.2	23.3	**23.4**
F_c	36.0	36.7	36.7	**36.9**	36.7

Table 9: Results with different numbers of clusters.

4.4 Analysis of Word Frequency

One of the strengths of our approach is that it can generate low-frequency substitutes that are OOV words in the original vocabulary. To confirm this, we analyse how well our method can generate low-frequency words from different vocabulary sizes $\tilde{V}$. Table 8 shows the results, in which we experiment with our BERT-based model with the vocabulary sizes of 5k, 10k, 20k, and 30k. The columns under "# Matched Words" show the numbers of correctly-predicted words, grouped by frequency range: low, med, and high denote words with frequency $<$50k, 50k–100k, and $>$100k in a large web corpus. The table shows that our method with 30k words generates nearly twice as many low-frequency substitutes as the baseline. Our method with 10k words still outperforms BERT-K in F_c, demonstrating its effectiveness. The last three rows show the performance of our method using other models, further demonstrating its ability to predict low-frequency words.

4.5 Effects of Cluster Size

Finally, we analyse the effect of the cluster size K for ELECTRA, as shown in Table 9. While a larger cluster size yields better performance, the improvement is marginal. Rather than using a fixed K, in future work we are interested in dynamically selecting the number of clusters per word.

5 Related Work

In the pre-BERT era, most lexical substitution methods employed linguistic resources such as WordNet (Fellbaum, 1998) to obtain substitute candidates (Szarvas et al., 2013; Hintz and Biemann, 2016). However, recent studies have shown that pre-trained language models such as BERT outperform these models without any external lexical resources. For instance, Zhou et al. (2019) feed a target sentence into BERT while partially masking the target word using dropout (Srivastava et al., 2014), and generate substitutes based on the probability distribution at the target word position. The masking strategy was shown to be effective on SemEval-07 but not on SWORDS. Similarly, Yang et al. (2022) feed two sentences into BERT, concatenating the target sentence with itself but with the target word replaced by [MASK], and predict words based on the mask-filling probability. Michalopoulos et al. (2022) augment pre-trained language models with WordNet and outperform Zhou et al. (2019). Lacerra et al. (2021a) fine-tune BERT on lexical substitution data sets that are automatically generated using BERT. They show that this approach is effective at ranking, and that adding manually-annotated data further boosts performance. Lacerra et al. (2021b) fine-tune BART on human-annotated data, and make it generate a list of substitutes given a target sentence in an end-to-end manner. They show that this generative approach rivals Zhou et al. (2019). Note that all of these recent models are evaluated on English only.

6 Conclusion

We present a new unsupervised approach to lexical substitution using pre-trained language models. We showed that our method substantially outperforms previous methods on English and Italian data sets, establishing a new state-of-the-art. By comparing performance on lexical substitution using different layers, we found that middle layers perform better than first or last layers. We also compared the substitutes predicted by the previous generative approach and our method, and showed that our approach works better at predicting low-frequency substitutes and reduces morphophonetic or morphosyntactic biases induced by article–noun agreement in English and Italian.

References

Nikolay Arefyev, Boris Sheludko, Alexander Podolskiy, and Alexander Panchenko. 2020. Always keep your target in mind: Studying semantics and improving performance of neural lexical substitution. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 1242–1255, Barcelona, Spain (Online). International Committee on Computational Linguistics.

David Arthur and Sergei Vassilvitskii. 2007. K-means++: The advantages of careful seeding. In *Proceedings of the Eighteenth Annual ACM-SIAM Symposium on Discrete Algorithms*, SODA '07, page 1027–1035, USA. Society for Industrial and Applied Mathematics.

Rishi Bommasani, Kelly Davis, and Claire Cardie. 2020. Interpreting pretrained contextualized representations via reductions to static embeddings. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 4758–4781, Online. Association for Computational Linguistics.

Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel Ziegler, Jeffrey Wu, Clemens Winter, Chris Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. 2020. Language models are few-shot learners. In *Advances in Neural Information Processing Systems*, volume 33, pages 1877–1901. Curran Associates, Inc.

Gabriella Chronis and Katrin Erk. 2020. When is a bishop not like a rook? When it's like a rabbi! Multi-prototype BERT embeddings for estimating semantic relationships. In *Proceedings of the 24th Conference on Computational Natural Language Learning*, pages 227–244, Online. Association for Computational Linguistics.

Kevin Clark, Minh-Thang Luong, Quoc V. Le, and Christopher D. Manning. 2020. ELECTRA: Pre-training text encoders as discriminators rather than generators. In *International Conference on Learning Representations*.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, Minneapolis, Minnesota. Association for Computational Linguistics.

Christiane Fellbaum. 1998. *WordNet: An Electronic Lexical Database*. Bradford Books.

Tianyu Gao, Xingcheng Yao, and Danqi Chen. 2021. SimCSE: Simple contrastive learning of sentence embeddings. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 6894–6910, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Patrick Hanks. 2012. The Corpus Revolution in Lexicography. *International Journal of Lexicography*, 25(4):398–436.

Pengcheng He, Jianfeng Gao, and Weizhu Chen. 2021. DeBERTaV3: Improving DeBERTa using ELECTRA-style pre-training with gradient-disentangled embedding sharing. *CoRR*, abs/2111.09543.

Gerold Hintz and Chris Biemann. 2016. Language transfer learning for supervised lexical substitution. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 118–129, Berlin, Germany. Association for Computational Linguistics.

Matthew Honnibal, Ines Montani, Sofie Van Landeghem, and Adriane Boyd. 2020. spaCy: Industrial-strength natural language processing in Python.

Mandar Joshi, Danqi Chen, Yinhan Liu, Daniel S. Weld, Luke Zettlemoyer, and Omer Levy. 2020. SpanBERT: Improving pre-training by representing and predicting spans. *Transactions of the Association for Computational Linguistics*, 8:64–77.

Kazuaki Kishida. 2005. Property of average precision and its generalization: An examination of evaluation indicator for information retrieval experiments. *NII Technical Report*.

Gerhard Kremer, Katrin Erk, Sebastian Padó, and Stefan Thater. 2014. What substitutes tell us — analysis of an "all-words" lexical substitution corpus. In *Proceedings of the 14th Conference of the European Chapter of the Association for Computational Linguistics*, pages 540–549, Gothenburg, Sweden. Association for Computational Linguistics.

Caterina Lacerra, Tommaso Pasini, Rocco Tripodi, and Roberto Navigli. 2021a. ALaSca: an automated approach for large-scale lexical substitution. In *Proceedings of the 30th International Joint Conference on Artificial Intelligence*. International Joint Conferences on Artificial Intelligence.

Caterina Lacerra, Rocco Tripodi, and Roberto Navigli. 2021b. GeneSis: A generative approach to substitutes in context. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 10810–10823, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Mina Lee, Chris Donahue, Robin Jia, Alexander Iyabor, and Percy Liang. 2021. Swords: A benchmark for lexical substitution with improved data coverage and

quality. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 4362–4379, Online. Association for Computational Linguistics.

Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Veselin Stoyanov, and Luke Zettlemoyer. 2020. BART: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7871–7880, Online. Association for Computational Linguistics.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. RoBERTa: A robustly optimized BERT pretraining approach. *CoRR*, abs/1907.11692.

S. Lloyd. 1982. Least squares quantization in pcm. *IEEE Transactions on Information Theory*, 28(2):129–137.

Diana McCarthy and Roberto Navigli. 2007. SemEval-2007 task 10: English lexical substitution task. In *Proceedings of the 4th International Workshop on Semantic Evaluations*, SemEval '07, page 48–53, USA. Association for Computational Linguistics.

Oren Melamud, Omer Levy, and Ido Dagan. 2015. A simple word embedding model for lexical substitution. In *Proceedings of the 1st Workshop on Vector Space Modeling for Natural Language Processing*, pages 1–7, Denver, Colorado. Association for Computational Linguistics.

George Michalopoulos, Ian McKillop, Alexander Wong, and Helen Chen. 2022. LexSubCon: Integrating knowledge from lexical resources into contextual embeddings for lexical substitution. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1226–1236, Dublin, Ireland. Association for Computational Linguistics.

Pedro Javier Ortiz Suárez, Laurent Romary, and Benoît Sagot. 2020. A monolingual approach to contextualized word embeddings for mid-resource languages. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 1703–1714, Online. Association for Computational Linguistics.

Emanuele Pianta, Luisa Bentivogli, and Christian Girardi. 2002. MultiWordNet: Developing an aligned multilingual database. In *Proceedings of the First International Conference on Global WordNet*, pages 293–302, Mysore, India.

Nils Reimers and Iryna Gurevych. 2019. Sentence-BERT: Sentence embeddings using Siamese BERT-Networks. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics.

Stefan Schweter. 2020. Italian BERT and ELECTRA models. https://doi.org/10.5281/zenodo.4263142.

Kaitao Song, Xu Tan, Tao Qin, Jianfeng Lu, and Tie-Yan Liu. 2020. MPNet: Masked and permuted pre-training for language understanding. In *Advances in Neural Information Processing Systems*, volume 33, pages 16857–16867.

Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. 2014. Dropout: A simple way to prevent neural networks from overfitting. *Journal of Machine Learning Research*, 15:1929–1958.

Bird Steven, Edward Loper, and Ewan Klein. 2009. *Natural Language Processing with Python*. O'Reilly Media Inc.

György Szarvas, Chris Biemann, and Iryna Gurevych. 2013. Supervised all-words lexical substitution using delexicalized features. In *Proceedings of the 2013 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 1131–1141, Atlanta, Georgia. Association for Computational Linguistics.

Yuqing Tang, Chau Tran, Xian Li, Peng-Jen Chen, Naman Goyal, Vishrav Chaudhary, Jiatao Gu, and Angela Fan. 2021. Multilingual translation from denoising pre-training. In *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*, pages 3450–3466, Online. Association for Computational Linguistics.

Ian Tenney, Dipanjan Das, and Ellie Pavlick. 2019. BERT rediscovers the classical NLP pipeline. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 4593–4601, Florence, Italy. Association for Computational Linguistics.

Antonio Toral. 2009. The lexical substitution task at EVALITA 2009. In *EVALITA workshop, 11th Congress of Italian Association for Artificial Intelligence*, Reggio Emilia, Italy.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Ł ukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, editors, *Advances in Neural Information Processing Systems 30*, pages 5998–6008. Curran Associates, Inc.

Gregor Wiedemann, Steffen Remus, Avi Chawla, and Chris Biemann. 2019. Does BERT make any sense? interpretable word sense disambiguation with contextualized embeddings. In *Proceedings of the 15th Conference on Natural Language Processing (KONVENS 2019): Long Papers*, pages 161–170, Erlangen,

Germany. German Society for Computational Linguistics & Language Technology.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Rémi Louf, Morgan Funtowicz, Joe Davison, Sam Shleifer, Patrick von Platen, Clara Ma, Yacine Jernite, Julien Plu, Canwen Xu, Teven Le Scao, Sylvain Gugger, Mariama Drame, Quentin Lhoest, and Alexander M. Rush. 2020. Transformers: State-of-the-art natural language processing. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 38–45, Online. Association for Computational Linguistics.

Xi Yang, Jie Zhang, Kejiang Chen, Weiming Zhang, Zehua Ma, Feng Wang, and Nenghai Yu. 2022. Tracing text provenance via context-aware lexical substitution. In *Proceedings of the 36th AAAI Conference on Artificial Intelligence*, Virtual.

Zhilin Yang, Zihang Dai, Yiming Yang, Jaime Carbonell, Russ R Salakhutdinov, and Quoc V Le. 2019. XLNet: Generalized autoregressive pretraining for language understanding. In *Advances in Neural Information Processing Systems*, volume 32.

Wangchunshu Zhou, Tao Ge, Ke Xu, Furu Wei, and Ming Zhou. 2019. BERT-based lexical substitution. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 3368–3373, Florence, Italy. Association for Computational Linguistics.

A Details of Pre-trained Models

Table 10 describe the details of the pre-trained models used in our experiments. We sourced these models from the Transformers library (Wolf et al., 2020) except for SpanBERT, which we obtained from the original GitHub repository (`https://github.com/facebookresearch/SpanBERT`).

B Layer-Wise Ranking Performance

Figure 2 shows the layer-wise performance in the ranking task. Similar to the generation results (Figure 1), middle layers perform better than the first or last layers. It also shows that sentence-embedding models (SimCSE/SBERT) outperform their original models (BERT/MPNet) for several layers, different from the generation results where they perform similarly. This suggests that fine-tuning on semantic downstream tasks improves the capacity of the model to differentiate subtle semantic differences between synonymous words, but not their ability to retrieve relevant words from a large pool of words; it also suggests that optimal representations for these objectives might differ.

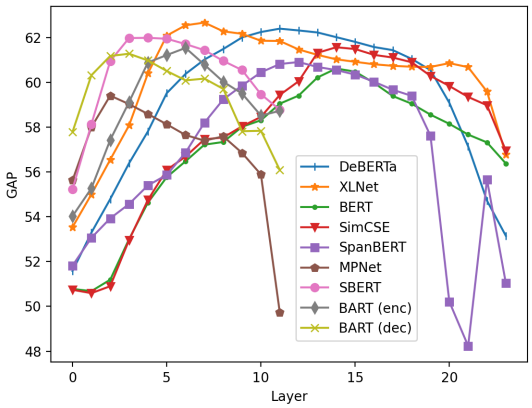

Figure 2: Layer-wise performance (GAP) on SWORDS.

C Effectiveness of Using Multiple Layers

Table 11 shows the generation and ranking performance of our model on SWORDS using different layers. It shows that our method using multiple layers $\ell \in Z$ performs comparably or even better than selecting the best layer tuned on the *test* set for each model. It also shows that the best-performing layer differs across models, suggesting they capture lexical information in a different manner.

D Examples of Generated Substitutes

Table 12 shows examples of substitutes generated by our method using different layers (without reranking). It shows that the words retrieved by each layer are very different, indicating that each layer encodes very different information about the input word. For instance, given the target word *care*, the first layer of BERT and BART-Enc/Dec retrieves a large number of words that contain the target word as a sub-morpheme (e.g. *aftercare*, *carefree*).[30] This is presumably because the first-layer representations are highly affected by the input word embedding, and hence result in retrieving words that share the same subword token (e.g. *care ##free*) regardless of the semantic similarity. The last layer also performs poorly (as previously shown in Figure 1), e.g. BART-DEC (L12) retrieves *participation* as the closest word to the target word *interest*. This is because the last-layer representations of BART-decoder are used to directly predict the next word *in* after *interest* in the target sentence, and in fact, most of the retrieved words (e.g. *uptick, faith, surge*) are those that often collocate with *in*.

[30] Since the edit distances between these words and the target word *care* are not greater than the threshold (0.5), they weren't filtered out by our heuristic.

Models	# Layer	Emb Size	$\|V\|$	Model Path
BERT	24	1024	30522	bert-large-uncased
mBERT	12	768	105879*	bert-base-multilingual-uncased
SpanBERT	24	1024	30522	spanbert-large-cased
MPNet	12	768	30527	microsoft/mpnet-base
XLNet	24	1024	32000	xlnet-large-cased
ELECTRA	24	1024	30522	google/electra-large-discriminator
DeBERTa-V3	24	1024	128000	microsoft/deberta-v3-large
BART (Enc/Dec)	12	1024	50265	facebook/bart-large
SBERT (MPNet)	12	768	30527	sentence-transformers/all-mpnet-base-v2
SimCSE (BERT)	24	1024	30522	princeton-nlp/sup-simcse-bert-large-uncased
NMT (mBART, Enc)	12	1024	250054*	facebook/mbart-large-50-one-to-many-mmt
BERT (Italian)	12	768	31102	dbmdz/bert-base-italian-xxl-uncased
ELECTRA (Italian)	12	768	31102	dbmdz/electra-base-italian-xxl-cased-discriminator

Table 10: Details of the pre-trained models used in this paper. $|V|$ denotes the original vocabulary size of each model. *The vocabularies of mBERT and mBART contain a great number of non-English words.

	Generation Performance (F_c)					Ranking Performance (GAP)				
Layer	First	Middle	Last	Best	$\ell \in Z$	First	Middle	Last	Best	$\ell \in Z$
BERT	24.4	32.2	27.8	**34.6** (16)	34.4	50.8	59.1	56.4	60.6 (15)	**60.7**
SpanBERT	24.9	33.1	8.8	**33.2** (13)	31.1	51.8	60.8	51.0	**60.9** (13)	60.8
MPNet	28.9	33.1	16.0	33.7 (4)	**33.8**	55.6	58.1	49.7	59.4 (3)	**59.5**
XLNet	28.6	33.0	28.4	34.3 (6)	**35.0**	53.5	61.9	56.8	62.7 (8)	**62.9**
DeBERTa-V3	33.3	36.8	25.2	**37.2** (11)	36.9	51.6	62.4	53.2	62.4 (12)	**62.9**
BART (Enc)	14.6	**30.8**	22.6	**30.8** (6)	30.0	54.0	61.2	58.7	61.5 (7)	**61.9**
BART (Dec)	21.9	28.0	18.9	**30.8** (3)	29.0	57.8	60.5	56.1	**61.3** (4)	60.8

Table 11: Generation and ranking performance of our approach on SWORDS using the first, middle ($\frac{L}{2}$th), last (Lth), or best layer tuned on the *test* set (the corresponding layer denoted in brackets); or using multiple layers $\ell \in Z$: {3, 4, ..., $L-2$} (L = 12 for MPNet and BART, and 24 otherwise). Generation and ranking performance across all layers is illustrated in Figure 1 and Figure 2.

Oddly, BART-Enc predicts a large number of substitutes that consist of multiple words (segmented by the tokeniser), none of which are relevant to the target word, e.g. *aswell*, *todo*, and *inbetween* as substitutes for *interest*. In fact, the number of such words increases (and the performance decreases) as the hyper-parameter λ gets bigger (which increases the influence of $f(x, c)$ on the predictions). One possible interpretation is that the last layer representations of the BART encoder may contain vague contextual information rather than the lexical information of the input word, since they are used by the decoder to predict various words (esp. masked words) during pre-training. Lastly, another interesting observation is that, for the target word *interest*, the last layer representations of BERT and BART-enc retrieve a lot of words that start from a vowel sound, despite the absence of the article *an* before *interest*, suggesting that the embeddings contain some morphophonetic information.

Context	I say I do not **care** about law, I care about service and she should care about money.
Gold (Conceivable)	*worry*, think, mind, desire, love, tend, cherish, consider, stress, concern, bother, watch
BERT (L3-22)	matter, caregiving, carefree, **worry**, **concern**, know, pay, caregivers, **love**, like
BART (L3-10)	**concern**, matter, carelessness, caretaker, carelessly, carefree, caretakers, **worry**, **bother**, **mind**
BERT (L1)	caregiving, carefree, caregiver, caregivers, aftercare, childcare, healthcare, skincare, custody, affections
BERT (L12)	matter, caregiving, **worry**, carefree, fret, **love**, despise, loathe, resent, pay
BERT (L24)	matter, **worry**, pay, **concern**, know, look, give, take, **bother**, **think**
BART-Enc (L1)	aftercare, caregiving, caretaker, carefree, carelessly, carelessness, caretakers, healthcare, caregivers, medicare
BART-Enc (L6)	**concern**, todo, caretaker, interest, carelessness, careless, disinterested, disdain, pay, carelessly
BART-Enc (L12)	todo, aswell, beleive, inbetween, zealand, pay, **concern**, usefull, noone, ofcourse
BART-Dec (L1)	caregiving, carelessness, caretaker, carelessly, carefree, aftercare, caretakers, **concern**, healthcare, **worry**
BART-Dec (L6)	**concern**, **bother**, **worry**, commit, reckon, dispose, shit, grieve, pay, strive
BART-Dec (L12)	**worry**, damn, **concern**, inquire, complain, passionate, shit, **think**, talk, whine
Context	"I'm starting to see more business transactions," says Andrea West of American Telephone & Telegraph Co., noting growing **interest** in use of 900 service for stock sales, software tutorials and even service contracts.
Gold (Conceivable)	*interestedness*, enthusiasm, demand, attraction, popularity, excitement, curiosity, activity, importance, notice, significance, involvement, relevance, note, gain, passion, influence, accrual, concernment
BERT (L3-22)	**curiosity**, **enthusiasm**, intrigued, desire, concern, fascination, **passion**, attention, **excitement**, fondness
BART (L3-10)	fascination, **enthusiasm**, appetite, **curiosity**, **excitement**, concern, inclination, eagerness, desire, **involvement**
BERT (L1)	concern, **importance**, **curiosity**, investment, **involvement**, attention, focussed, fascination, focus, **significance**
BERT (L12)	**curiosity**, **enthusiasm**, concern, fascination, confidence, unease, belief, **excitement**, desire, **passion**
BERT (L24)	appetite, attracting, attractiveness, actively, **demand**, attention, intrigued, **popularity**, **enthusiasm**, flocking
BART-Enc (L1)	fascination, concern, intrigue, **involvement**, **relevance**, investment, stake, **curiosity**, **enthusiasm**, trustworthiness
BART-Enc (L6)	**enthusiasm**, fascination, appetite, **curiosity**, intrigued, **excitement**, intrigue, inclination, eagerness, enjoyment
BART-Enc (L12)	todo, aswell, inbetween, **enthusiasm**, fascination, eagerness, appetite, intrigued, attention, inclination
BART-Dec (L1)	fascination, **involvement**, intrigue, concern, participation, **curiosity**, investment, **excitement**, intrigued, **importance**
BART-Dec (L6)	fascination, appetite, **involvement**, engagement, affinity, uptake, **demand**, inclination, participation, appreciation
BART-Dec (L12)	participation, uptick, delight, decline, increase, spike, faith, decrease, grounding, surge

Table 12: Examples of substitutes predicted by our method (w/o rerank) using different layers. Gold shows a list of "conceivable" words sorted by their annotated scores (with "acceptable" words shown in italic, and multiword expressions omitted from the table). Words included in Gold are boldfaced.

Transparent Semantic Parsing with Universal Dependencies using Graph Transformations

Wessel Poelman, Rik van Noord, Johan Bos
Center for Language and Cognition
University of Groningen
The Netherlands
contact@wesselpoelman.nl
{r.i.k.van.noord,johan.bos}@rug.nl

Abstract

Even though many recent semantic parsers are based on deep learning methods, we should not forget that rule-based alternatives might offer advantages over neural approaches with respect to transparency, portability, and explainability. Taking advantage of existing off-the-shelf Universal Dependency parsers, we present a method that maps a syntactic dependency tree to a formal meaning representation based on Discourse Representation Theory. Rather than using lambda calculus to manage variable bindings, our approach is novel in that it consists of using a series of graph transformations. The resulting UD semantic parser shows good performance for English, German, Italian and Dutch, with F-scores over 75%, outperforming a neural semantic parser for the lower-resourced languages. Unlike neural semantic parsers, our UD semantic parser does not hallucinate output, is relatively easy to port to other languages, and is completely transparent.

1 Introduction

Semantic parsing is the task of mapping natural language sentences to a formal meaning representation such as Abstract Meaning Representations (Banarescu et al., 2013) or Discourse Representation Structures (Bos et al., 2017). The current trend in this area is strongly geared towards using methods based on deep learning. The best performing parsers use pre-trained language models (van Noord et al., 2020; Zhou et al., 2021; Bevilacqua et al., 2021; Bai et al., 2022). But a good performance is perhaps not the only thing that matters. A drawback of neural semantic parsers is that their output lacks explainability: why are the meaning representations composed in the way they are? Moreover, they require vast amounts of training data, and are usually specific for a particular language. In addition, their performance usually decreases for longer input sentences.

In other words, it may look like we have made a lot of progress, but viewed from a different perspective, we might actually have made a step back. This is especially so with regards to transparency and interpretability of semantic parsers. In this paper we describe a semantic parsing system for Discourse Representation Structures — the formal meaning representations proposed by Discourse Representation Theory (Kamp and Reyle, 1993; Abzianidze et al., 2017) — that is based on Universal Dependencies (UD, de Marneffe et al., 2021). The first advantage of the UD framework is that it has been developed for numerous languages (using a cross-linguistically consistent annotation scheme) and that several high-performing parsers have been developed for UD. This will make it easier, as we will show, to develop semantic parsers for languages other than English, in our case German, Italian and Dutch. The second advantage of using UD as input describing the syntactic structure of the sentence, is that it provides us with explainable support of the output of the meaning representation, based on the derivation provided by the UD parser.

The innovative contribution of the system, *UD-boxer*, that we describe here is in the way the meaning representations are computed. Even though the original Discourse Representation Structure (DRS) is formally an ordered pair of a set of discourse referents and a set of conditions, we recast the DRS as a directed acyclic graph. Through graph transformation rules our system changes the input UD syntactic representation step-by-step into a fully fledged formal meaning representation.

2 Related Work

Our aims are similar to those of Reddy et al. (2016) and Reddy et al. (2017), who map UD to logical forms in three steps: (1) enriching the UD tree with missing syntactic information and long-distance dependencies, turning it into a graph, (2) binarization of the dependency graph, (3) substituting the

Proceedings of the 29th International Conference on Computational Linguistics, pages 4186–4192
October 12–17, 2022.

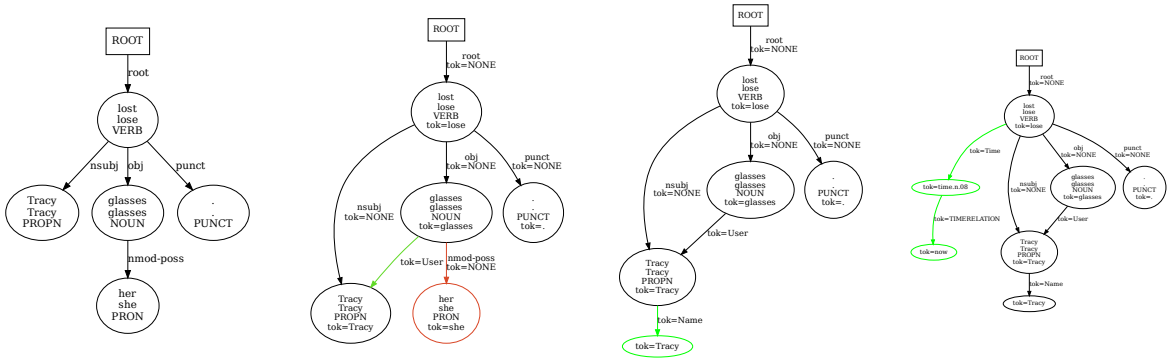

Figure 1: Graph transformations for *Tracy lost her glasses*, from left to right: initial UD graph, connecting the User role, expanding the proper name, and adding and connecting tense information. The *token* attribute refers to the semantic label or concept for a given node or edge.

words and using typed lambda expressions that encode the lexical semantics and dependency labels for λ-expressions that either copy, invert or merge lambda-expression to compose predicate-argument structures, and (4) applying β-conversion to get a reduced, normalized logical form.

The main difference of our work with that of Reddy et al. (2017) is that we do not require complicated operations involving logical variables. By making the target meaning representations free of variables in the form of a graph, the mapping from UD to meaning representation is solely based on a sequence of graph transformations (Zhizhkun, 2006). This allows us to apply our method on languages other than English.

Similar in spirit to our work, but different in execution, is recent work by Shen and Evang (2022), who present a DRS parser that is competitive in accuracy with recent sequence-to-sequence models and at the same time *compositional*. This latter property makes their system transparent and more explainable. Shen and Evang (2022) recast DRS parsing as a sequence labeling task, and achieve a good performance with F-scores of 84.4 for English, 78.3 for German, 80.4 for Italian, and 72.1 for Dutch on PMB 3.0.0 data (and therefore not directly comparable with our results, working with a more recent version of the PMB).

3 Method

3.1 Overall Idea

Our semantic parsing method is based on the insight provided by Bos (2021) that Discourse Representation Structures can be represented as relatively simple directed acyclic graphs without resorting to variables. In a Discourse Representation Graph (DRG) the nodes denote entities (represented by a WordNet synset) and constants (names, numbers, dates, etc.), the edges denote thematic roles, comparison operators, and discourse relations. Our semantic mapping comprises a series of transformations from UD to DRG, exemplified by Figure 1 and Figure 2. Our full system is publicly available: https://github.com/WPoelman/ud-boxer

We use semantically annotated data from the Parallel Meaning Bank (Abzianidze et al., 2017). The PMB provides a large set of English, German, Italian and Dutch sentences paired with Discourse Representation Structures (DRSs). Since release 4.0.0[1] the PMB also provides DRSs in a variable-free variant, using relative indices instead of variables, following the procedure outlined in Bos (2021). This notation corresponds directly to a DRG. There is a straightforward mapping from DRG to DRS of formulas of first-order logic (see Figure 2). We use this format in developing our system. The PMB has gold, silver (partially annotated) and bronze (no manual annotations) standard data available. We only use gold for UD-boxer, while for Neural Boxer, a strong neural parser based on van Noord et al. (2020), we use all available data for German, Italian and Dutch. However, for training Neural Boxer on English data we only use the gold and silver data (van Noord et al., 2018). The data splits are shown in Table 1. Note that English also has an extra evaluation set of 830 instances. This was the hidden test set in the shared task of Abzianidze et al. (2019) and now serves as an extra test set.

[1]The data of the Parallel Meanng Bank is available here: https://pmb.let.rug.nl/data.php.

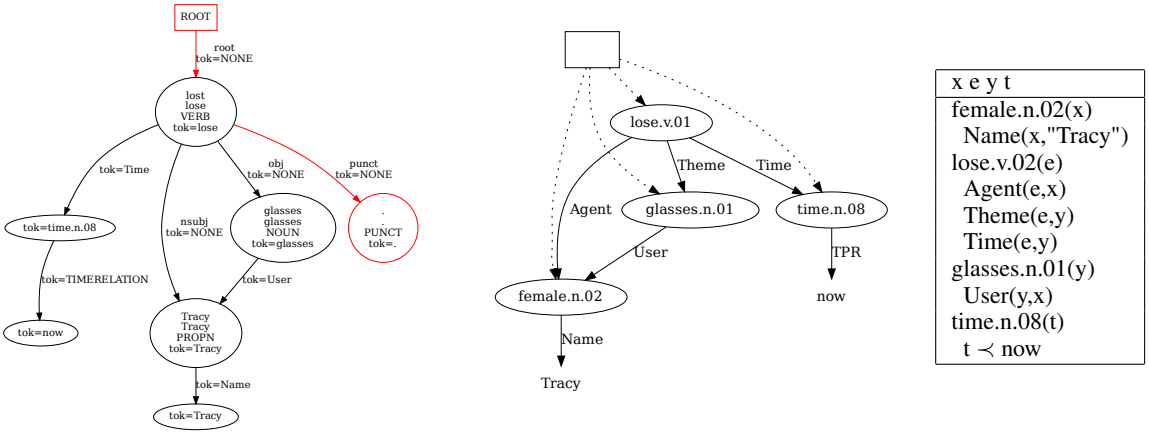

Figure 2: Removing redundant nodes followed by substitution semantic symbols for syntactic relations, and corresponding DRS in box format for *Tracy lost her glasses*.

	Gold			Silver	Bronze
	Train	Dev	Test	Train	Train
English	7,668	1,169	1,048	127,303	151,493
German	1,738	559	547	6,355	156,286
Italian	685	540	461	4,088	100,963
Dutch	539	491	437	1,440	28,265

Table 1: Number of documents for the four languages for PMB release 4.0.0.

3.2 System Overview

The overall system expects a sentence as input and consists of three main steps: (1) creating a UD parse; (2) applying the graph transformation rules; and (3) substituting syntactic labels for semantic entities. The final output is a DRG and can be exported to various formats.

The first step is implemented by using existing off-the-shelf UD parsers. This is a modular part of the system — any UD parser can be plugged in. In the context of this paper we used two state-of-the-art UD parsers: Stanza (Qi et al., 2020) and Trankit (Nguyen et al., 2021), both of which go head to head in their performance for English and also achieve good results for the other languages of our study.

The second step, the graph transformation, is carried out by using GREW, a graph rewriting framework specifically designed for linguistic graphs and trees (Bonfante et al., 2018; Guillaume, 2021). The focus of this step is to apply *structural* changes to the graph: adding, removing or combining nodes and edges. Some node and edge label substitution might be carried out already during graph transformation, but most of that is left to the final step. Figure 3 shows a language-neutral transformation rule that connects a thematic role to an entity, whereas Figure 4 is an example of a language-specific rule.

```
rule connect_user {
    pattern {
        USER [upos=PROPN|NOUN];
        * -[1=nsubj]-> USER;
        REL: TARGET -[1=nmod, 2=poss]->
              INDICATOR;
    }
    without {
        TARGET -[token=User]-> USER;
    }
    commands {
        add_edge TARGET -[token=User]->
                 USER;
        del_edge REL;
        del_node INDICATOR;
    }
}
```

Figure 3: Example of language-neutral rule (in GREW syntax) to connect the User role to an entity.

The final step, substitution, involves labeling nodes and edges that do not have valid DRS labels yet, as well as connecting box nodes (the only structural part of this step). Currently, this step is comprised of applying simple mappings extracted from the training data and leveraging syntactic and morphological information from the UD parse. This is also a modular component and additional approaches can be added. Existing systems that go beyond syntax are a good candidate to be added here, e.g., named entity recognition systems.

```
rule box_negation_det {
    pattern {
        N [lemma=no|not|never];
        * -[1=advmod|det]-> N;
    }
    without {
        P [token=NEGATION];
    }
    commands {
        del_node N;

        add_node NEGATION_BOX;
        NEGATION_BOX.token = NEGATION;
    }
}
```

Figure 4: Example of English specific rule to introduce a negation box. The box gets connected in the substitution and labeling step.

The transformation rules as well as the mappings for the substitution are developed using the training data and tested on the development set. Currently there are 19 language independent rules and four specific rules per language. These specific rules deal with either negation or quantifiers (e.g. *all*, *every*, *none*). Rules were developed by analyzing the UD graph and gold SBN graph side-by-side for a given example sentence. We then aimed to create the most general and simple rule(s) that (structurally) transformed the UD graph into the SBN graph.

The node and edge mappings are extracted when the graph transformations are applied and result in a graph isomorphic to the gold-standard graph. The UD information is then extracted and stored per triple (from node, edge, to node) if the mapping was correct. This creates a positive feedback loop, as the rules improve, the labeling improves as well. This process was bootstrapped by creating a tiny set of initial mappings from dependency relations to DRS roles and operators. Our approach here serves as a baseline of sorts, since word sense disambiguation and edge labeling are only done with the most frequent occurrences in the training data.

3.3 Evaluating DR Graphs

Counter is the standard evaluation tool for DRSs (van Noord et al., 2018). However, it is specifically designed for the *clausal* notation of DRS. This notation does not use a graph-like structure directly, but works with clauses that can have three or four components. It is therefore not suitable for the (simpler) DRGs that our system produces.

SMATCH (Cai and Knight, 2013) was created for evaluating Abstract Meaning Representations, which are directed acyclic graphs, like DRGs. SMATCH supports the Penman notation (Kasper, 1989), converts a graph into a set of triples, while automatically performing role inversion when needed. By converting a DRG into Penman format, we can simply use SMATCH to compare system output with the gold standard, for which SMATCH computes an F-score based on matching triples. A DRG in Penman format is shown in Figure 5.

```
(b0 / box
    :member (e1 / entity
        :lemma female :pos n :sense 02
        :Name "Tracy")
    :member (e2 / entity
        :lemma lose :pos v :sense 02
        :Agent e1 :Theme e3 :Time e4)
    :member (e3 / entity
        :lemma glasses :pos n :sense 01
        :User e1)
    :member (e4 / entity
        :lemma time :pos n :sense 08
        :TPR now))
```

Figure 5: Penman format for a Discourse Representation Graph for *Tracy lost her glasses.*

As Figure 5 shows, we split up WordNet synsets components to support a more fine-grained evaluation. This also gives us flexibility in the evaluation process, where we can toggle between evaluating word sense disambiguation (strict) or not (lenient). In this paper, we use only strict evaluation.

3.4 Comparison System

For comparison with our system we train a neural DRG parser based on BERT (Devlin et al., 2019), following van Noord et al. (2020).[2] This is a bi-LSTM sequence-to-sequence model, which uses (frozen) BERT embeddings to initialize the encoder. Specifically, we use `bert-base-cased` for English and `bert-base-multilingual-cased` for the other languages. The word-level decoder is trained from scratch. We do not apply any preprocessing nor postprocessing, simply taking the input sentence and output DRS in sequential box notation as is. We follow the procedure described in van Noord et al. (2020) by first pretraining on gold + non-gold data, after which we fine-tune on just the gold data.

[2]Detailed instructions can be found here: https://github.com/RikVN/Neural_DRS/blob/master/AllenNLP.md#sbn-experiments.

	English		German		Italian		Dutch	
	Dev	**Test**	**Dev**	**Test**	**Dev**	**Test**	**Dev**	**Test**
UD-Boxer (Stanza)	82.1 (0.3)	82.0 (0.0)	**78.4 (0.0)**	77.3 (0.0)	76.2 (1.9)	78.4 (0.9)	75.5 (0.0)	**75.8 (0.0)**
UD-Boxer (Trankit)	81.9 (0.3)	81.8 (0.0)	**78.4 (0.0)**	**77.5 (0.0)**	**77.8 (0.0)**	**79.1 (0.0)**	**75.8 (0.0)**	**75.8 (0.0)**
Neural Boxer (gold)	82.8 (4.6)	84.0 (3.7)	64.2 (0.4)	63.8 (0.2)	55.5 (1.5)	55.7 (1.5)	51.2 (0.2)	51.1 (0.4)
Neural Boxer (best)	**92.5 (2.0)**	**92.5 (2.3)**	74.6 (0.4)	74.7 (0.5)	75.6 (0.0)	75.4 (0.0)	71.9 (0.9)	71.6 (1.0)

Table 2: Average macro F1-scores on the dev and test set of the four languages in PMB 4.0.0. The number in parentheses indicates the percentage of ill-formed DRSs in the output. For Neural Boxer, *best* indicates that it was trained on gold, silver and bronze data (German, Italian, and Dutch) or only on gold and silver data (English). UD-Boxer (Stanza) and UD-Boxer (Trankit) obtain an F-score of 81.3 and 81.5 on the English *evaluation* set.

4 Results

Table 2 shows the main results for UD-boxer. We show the results of using two syntactic parsers (Stanza and Trankit) and compare to the performance of the neural system, trained on just the gold PMB data and on gold and non-gold data. Our system is not competitive with the best Neural Boxer for English, but clearly outperforms this model for German, Italian and Dutch. However, when only using gold data, the performance of our model is quite close to Neural Boxer for English, while producing considerably fewer ill-formed DRSs.

In a manual analysis, we found that the few errors made by UD-Boxer were all caused by unexpected sentence roots in the UD output of the Stanza parser. A case in point is the input *All of my friends like computer games* where Stanza decides that it is a noun phrase with *All* as the root, whereas Trankit assigns *like* as the root. Currently no transformation rules deal with such cases. The graph gets malformed because the root is cut away at some point since it is a determiner and those can generally be left out of DRSs.

But the majority of ill-formed output is produced by the neural parsers. An example, which also exhibits hallucination of semantic information, is the output DRG for *She died of tuberculosis* (Figure 6), where the disease changed into a heart attack and is recognized as an anonymous geological formation (there is a role Name without a constant). This strange phenomenon occurs often in output from both neural parsers. The problem with detecting these anomalies is that often these graphs score well, even though they are semantically ill-formed.

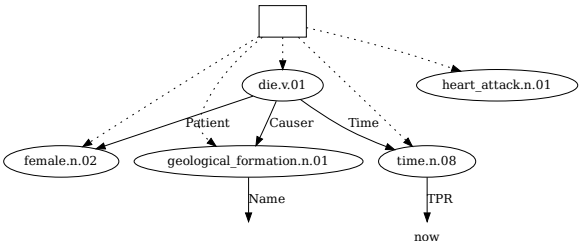

Figure 6: Erroneously (wrong type of disease, incorrect named entity) and ill-formed output (node missing) of Neural Boxer for *She died of tuberculosis.*

5 Conclusion

Our results show competitive performance between UD-Boxer and Neural Boxer for English, when a limited amount of training data is available for the neural approach. In addition, it shows strong cross-lingual performance using a few simple language-specific rules and only gold training data to extract the mappings. Adding more transformation rules and creating a label substitution component based on machine learning will likely push the performance of UD-Boxer even higher. The phenomena we have in mind are named entities, numeral expressions, time and date expressions, and discourse relations, for which only simple rules have been defined so far. Since UD-Boxer is a modular system, it is rather straightforward to add such rules.

An important difference between the two semantic parsers is that UD-Boxer is guaranteed to output a well-formed meaning representation, provided that the input UD is accurate. Because Neural Boxer is following a seq2seq transformer approach, the output does not always correspond to a graph and therefore requires ad-hoc postprocessing rules to make such output interpretable. Another serious deficiency of neural parsers is that it sometimes hallucinates semantic material without warning. A transparent semantic parser, even with a slightly lower performance in some cases, might be a good alternative for certain applications, in particular when lower-resourced languages are involved.

Acknowledgements

The Tesla K40 GPU used in this work was kindly donated to us by the NVIDIA Corporation. We thank the Center for Information Technology of the University of Groningen for providing access to the Peregrine high performance computing cluster. We also thank the three anonymous reviewers for their valuable feedback.

References

Lasha Abzianidze, Johannes Bjerva, Kilian Evang, Hessel Haagsma, Rik van Noord, Pierre Ludmann, Duc-Duy Nguyen, and Johan Bos. 2017. The Parallel Meaning Bank: Towards a multilingual corpus of translations annotated with compositional meaning representations. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 2, Short Papers*, pages 242–247, Valencia, Spain. Association for Computational Linguistics.

Lasha Abzianidze, Rik van Noord, Hessel Haagsma, and Johan Bos. 2019. The first shared task on discourse representation structure parsing. In *Proceedings of the IWCS Shared Task on Semantic Parsing*, Gothenburg, Sweden. Association for Computational Linguistics.

Xuefeng Bai, Yulong Chen, and Yue Zhang. 2022. Graph pre-training for AMR parsing and generation. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 6001–6015, Dublin, Ireland. Association for Computational Linguistics.

Laura Banarescu, Claire Bonial, Shu Cai, Madalina Georgescu, Kira Griffitt, Ulf Hermjakob, Kevin Knight, Philipp Koehn, Martha Palmer, and Nathan Schneider. 2013. Abstract Meaning Representation for sembanking. In *Proceedings of the 7th Linguistic Annotation Workshop and Interoperability with Discourse*, pages 178–186, Sofia, Bulgaria. Association for Computational Linguistics.

Michele Bevilacqua, Rexhina Blloshmi, and Roberto Navigli. 2021. One spring to rule them both: Symmetric amr semantic parsing and generation without a complex pipeline. *Proceedings of the AAAI Conference on Artificial Intelligence*, 35(14):12564–12573.

G. Bonfante, B. Guillaume, and G. Perrier. 2018. *Application of Graph Rewriting to Natural Language Processing*. Wiley.

Johan Bos. 2021. Variable-free discourse representation structures. *Semantics Archive*.

Johan Bos, Valerio Basile, Kilian Evang, Noortje Venhuizen, and Johannes Bjerva. 2017. The Groningen Meaning Bank. In Nancy Ide and James Pustejovsky, editors, *Handbook of Linguistic Annotation*, volume 2, pages 463–496. Springer.

Shu Cai and Kevin Knight. 2013. Smatch: an evaluation metric for semantic feature structures. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 748–752, Sofia, Bulgaria. Association for Computational Linguistics.

Marie-Catherine de Marneffe, Christopher D. Manning, Joakim Nivre, and Daniel Zeman. 2021. Universal Dependencies. *Computational Linguistics*, 47(2):255–308.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, Minneapolis, Minnesota. Association for Computational Linguistics.

Bruno Guillaume. 2021. Graph Matching and Graph Rewriting: GREW tools for corpus exploration, maintenance and conversion. In *EACL 2021 - 16th conference of the European Chapter of the Association for Computational Linguistics*, Kiev/Online, Ukraine.

Hans Kamp and Uwe Reyle. 1993. *From Discourse to Logic; An Introduction to Modeltheoretic Semantics of Natural Language, Formal Logic and DRT*. Kluwer, Dordrecht.

Robert T. Kasper. 1989. A flexible interface for linking applications to Penman's sentence generator. In *Speech and Natural Language: Proceedings of a Workshop Held at Philadelphia, Pennsylvania, February 21-23, 1989*.

Minh Van Nguyen, Viet Dac Lai, Amir Pouran Ben Veyseh, and Thien Huu Nguyen. 2021. Trankit: A lightweight transformer-based toolkit for multilingual natural language processing. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: System Demonstrations*, pages 80–90, Online. Association for Computational Linguistics.

Peng Qi, Yuhao Zhang, Yuhui Zhang, Jason Bolton, and Christopher D. Manning. 2020. Stanza: A Python natural language processing toolkit for many human languages. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics: System Demonstrations*.

Siva Reddy, Oscar Täckström, Michael Collins, Tom Kwiatkowski, Dipanjan Das, Mark Steedman, and Mirella Lapata. 2016. Transforming dependency structures to logical forms for semantic parsing. *Transactions of the Association for Computational Linguistics*, 4:127–140.

Siva Reddy, Oscar Täckström, Slav Petrov, Mark Steedman, and Mirella Lapata. 2017. Universal semantic parsing. In *Proceedings of the 2017 Conference on*

Empirical Methods in Natural Language Processing, pages 89–101, Copenhagen, Denmark. Association for Computational Linguistics.

Minxing Shen and Kilian Evang. 2022. DRS parsing as sequence labeling. In *Proceedings of the 11th Joint Conference on Lexical and Computational Semantics*, pages 213–225, Seattle, Washington. Association for Computational Linguistics.

Rik van Noord, Lasha Abzianidze, Hessel Haagsma, and Johan Bos. 2018. Evaluating scoped meaning representations. In *Proceedings of the Eleventh International Conference on Language Resources and Evaluation (LREC 2018)*, pages 1685–1693. European Language Resources Association (ELRA).

Rik van Noord, Lasha Abzianidze, Antonio Toral, and Johan Bos. 2018. Exploring neural methods for parsing discourse representation structures. *Transactions of the Association for Computational Linguistics*, 6:619–633.

Rik van Noord, Antonio Toral, and Johan Bos. 2020. Character-level representations improve DRS-based semantic parsing even in the age of BERT. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 4587–4603. Association for Computational Linguistics.

Valentin Zhizhkun. 2006. *Graph Transformations for Natural Language Processing*. Ph.D. thesis, University of Amsterdam.

Jiawei Zhou, Tahira Naseem, Ramón Fernandez Astudillo, Young-Suk Lee, Radu Florian, and Salim Roukos. 2021. Structure-aware fine-tuning of sequence-to-sequence transformers for transition-based AMR parsing. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 6279–6290, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Multilingual Word Sense Disambiguation with Unified Sense Representation

Ying Su[1], **Hongming Zhang**[1,2], **Yangqiu Song**[1], **Tong Zhang**[1]
[1]HKUST, [2]Tencent AI lab, Seattle
ysuay@connect.ust.hk, {hzhangal,yqsong}@cse.ust.hk,
tongzhang@ust.hk

Abstract

As a key natural language processing (NLP) task, word sense disambiguation (WSD) evaluates how well NLP models can understand the lexical semantics of words under specific contexts. Benefited from the large-scale annotation, current WSD systems have achieved impressive performances in English by combining supervised learning with lexical knowledge. However, such success is hard to be replicated in other languages, where we only have limited annotations. In this paper, based on the multilingual lexicon BabelNet describing the same set of concepts across languages, we propose building knowledge and supervised-based Multilingual Word Sense Disambiguation (MWSD) systems. We build unified sense representations for multiple languages and address the annotation scarcity problem for MWSD by transferring annotations from rich-sourced languages to poorer ones. With the unified sense representations, annotations from multiple languages can be jointly trained to benefit the MWSD tasks. Evaluations of SemEval-13 and SemEval-15 datasets demonstrate the effectiveness of our methodology.

1 Introduction

As a critical natural language understanding task, word sense disambiguation (WSD) aims at classifying words into pre-defined senses. With such a disambiguation process, machines can understand the precise meanings of words. Previous researches have demonstrated that a sound WSD system could benefit many downstream NLP tasks, such as machine translation (Pu et al., 2018; Liu et al., 2018) and information extraction (Bovi et al., 2015).

Existing researches on word sense disambiguation mostly focus on English only. By leveraging lexical knowledge such as gloss (Iacobacci et al., 2016; Luo et al., 2018; Huang et al., 2019; Blevins and Zettlemoyer, 2020) or graph structure (Banerjee et al., 2003; Kumar et al., 2019; Bevilacqua and Navigli, 2020) and supervised training over large-scale annotations, these models have achieved impressive performance on the standard English WSD task. However, though the English WSD task (Raganato et al., 2017) and multilingual WSD (MWSD) task (Navigli et al., 2013; Moro and Navigli, 2015) are of the same form as shown in Figure 1, this progress can not be easily applied across languages as the paucity of annotated training data and immense labor in handling diverse lexical knowledge of multiple languages separately.

WSD Example. Language: English

*Context: Detailed studies of the **plan** were well underway*.

plan#NOUN: plan%1:09:00:: **plan%1:09:01::** plan%1:06:00::

MWSD Example. Language: French

*Context: Le groupe des Nations_Unies a des projets de **plans** pour la réduction des émissions*.

plan#NOUN: **bn:00062759n** bn:00062766n bn:00005439n

Figure 1: Examples of WSD and Multilingual WSD (MWSD) task. The target words are indicated with the bold font in contexts. Candidate sense keys are listed below each context, and the one in blue is the correct sense.

BabelNet (Navigli and Ponzetto, 2012) is a multilingual semantic lexicon and contains a set of multilingually lexicalized concepts. Similar to WordNet (Miller, 1998), a Babel synset defines a concept shared by a group of words across languages with the same meaning. Based on the multilingual lexicon source BabelNet, we propose to build multilingual word sense disambiguation systems by inducing lexical knowledge and annotations from rich sourced language (e.g., English) to scarce sourced ones. First, as defined in BabelNet, words in each synset have the same sense, and the sense is described by lexical knowledge gloss despite the language forms. An example is

Proceedings of the 29th International Conference on Computational Linguistics, pages 4193–4202
October 12–17, 2022.

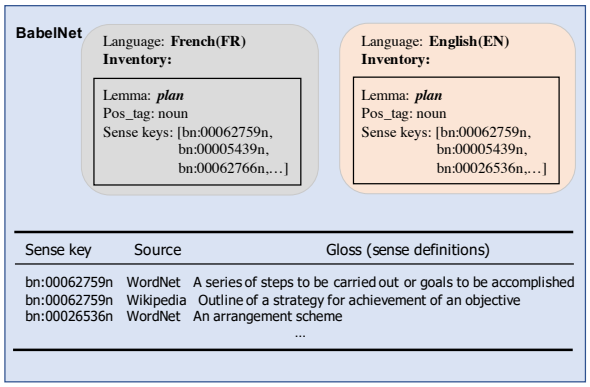

Figure 2: BabelNet contains inventories for multiple languages. Each word in a language has several senses, and different words across languages may share the same senses. For each sense across languages, glosses from various sources such as WordNet and Wikipedia are collected to describe its meaning.

shown in Figure 2. The knowledge can be injected into supervised MWSD systems. Second, the annotations acquired from rich sourced languages through machine translation and alignment tools, can be used as weak supervision. By utilizing the lexical knowledge and weak annotations, we can build a decent MWSD system for scarce sourced languages without further human effort.

To summarize, the contributions of this paper are two-fold: (1) We propose to build an MWSD system mBERT-UNI for multiple languages with transferred annotations from rich sourced languages and unified synsets with lexical knowledge, addressing the data paucity problem on the MWSD task; (2) Our system can be easily combined with other data generation efforts such as MuLaN (Barba et al., 2020), further boosting the system performance. Experiments results on benchmark SemEval-13 (Navigli et al., 2013) and SemEval-15 (Moro and Navigli, 2015) demonstrate the effectiveness of our methodology. Our code is open-resourced[1].

2 Related Work

This section introduces previous efforts on multilingual word sense disambiguation, which can be categorized into two streams: data-driven systems and knowledge-based systems.

2.1 Data-driven Systems

In the last decades, many efforts in the field of multilingual word sense disambiguation have been devoted to mitigating the knowledge acquisition bottleneck problem (Gale et al., 1992; Pasini, 2020), which is hard to acquire sense-annotated corpora for multiple languages. To mitigate the paucity of annotations, many researchers have focused on automatically creating high-quality, sense-annotated training corpora (Pasini and Navigli, 2020). OMSIT (Taghipour and Ng, 2015) proposed a semi-automatic approach to acquire one million training instances from MultiUN dataset (Eisele and Chen, 2010). OneSec (Scarlini et al., 2019) proposed to generate multilingual sense-annotated datasets on a large scale by mapping Wikipedia categories to word senses. MuLaN (Barba et al., 2020) utilized contextualized word embeddings to transfer sense annotations from labeled datasets SemCor (Miller et al., 1993) and WNG (Langone et al., 2004) to the unlabeled corpus from Wikipedia across languages. Hauer et al. (2021) proposed a label propagation approach for constructing multilingual sense-annotated corpora by machine translation. XL-WSD (Pasini et al., 2021) further enriches the annotations across 18 languages from six different linguistic families. Similar to Hauer et al. (2021), our automatic corpora generation method takes advantage of machine translation and alignment tools, while it is easy and feasible to use without additional resources. Moreover, we also induce lexical knowledge in building sense representations.

2.2 Knowledge-based Systems

Besides annotated corpora, lexical knowledge such as sense inventories is another key component in word sense disambiguation systems. Lexical knowledge sources such as WordNet and BabelNet provide rich lexical knowledge, e.g., gloss or graph structure. Such knowledge has been exploited and shows decent performance in many supervised systems (Kumar et al., 2019; Loureiro and Jorge, 2019; Scarlini et al., 2020; Blevins and Zettlemoyer, 2020). Readers can refer to (Bevilacqua et al., 2021) for more details.

In this paper, we aim to induce lexical knowledge into MWSD systems. Based on the synsets and lexical knowledge in multilingual lexicon BabelNet, we propose to build unified sense representations that can be shared across languages. The sense representations can be incorporated into supervised systems to improve the performance of MWSD tasks.

[1] https://github.com/suytingwan/multilingual-WSD

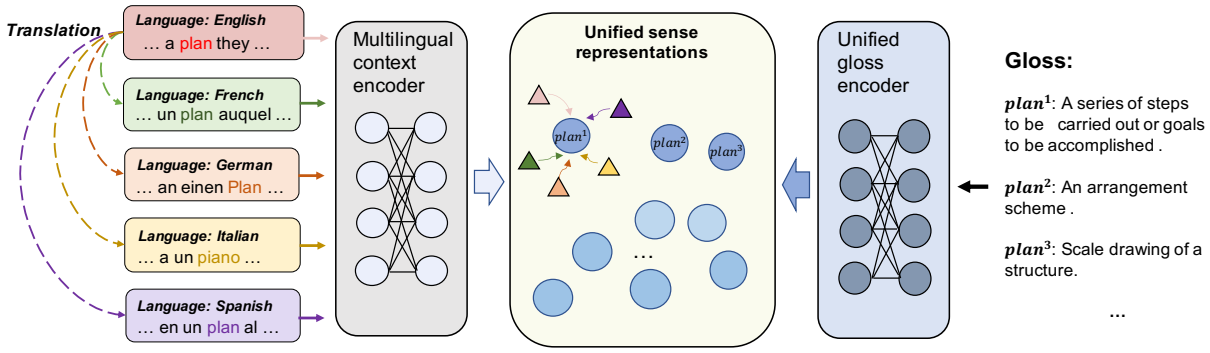

Figure 3: Architecture Overview of the mBERT-UNI model. We apply multilingual BERT for the encoders. The context encoder takes the multilingual context as input and generates representations for the target words. The gloss encoder takes the glosses as input and generates unified sense representations. The similarity scores between target word embedding and representation of candidate senses are calculated for each language. The sense with the highest score is the predicted sense by the model.

3 Approach

In this section, we first present the formal definition of the multilingual WSD task and used notations. After that, we present the details of the proposed system mBERT-UNI, a supervised framework incorporating lexical unified representation space for the MWSD task. From the overview in Figure 3, we can see that mBERT-UNI can be decomposed into four parts: (1) To address the data paucity issue, we first translate the annotated English corpus SemCor into other languages and use alignment tool to generate sense annotations; (2) A context encoder encodes the target words in multilingual context; (3) A gloss encoder encodes the glosses to produce unified sense representations; (4) The annotated corpus in several languages and unified sense representations are bound with a joint training setting.

3.1 Task Description and Notations

In the multilingual setting, the WSD task is to disambiguate the senses for a sequence of words $\{w_1, \cdots, w_m\}$ in a sentence S. The sentences come from various languages $L \in \{L_1, L_2, ..., L_n\}$. For each word w, the goal is to map it to a pre-defined sense $s \in \mathcal{S}_m$, where $\mathcal{S}_m = \{s_1, s_2, ..., s_k\}$ is the set of pre-defined candidate senses for w. The meaning of each sense is defined by the gloss. The candidate senses have a corresponding gloss set defined as $G = \{g_1, g_2, ..., g_k\}$. For the MWSD task, multiple languages have different inventories but share the same set of synsets and glosses as defined in BabelNet (Navigli and Ponzetto, 2012).

3.2 Multilingual Corpora Preparation

We use machine translation and alignment tools to acquire annotated training data for multiple languages. Machine translation has been developed for decades and has achieved remarkable progress (Wu et al., 2016; Tiedemann et al., 2020). Following (Luan et al., 2020), we use google translation to acquire parallel training corpora from English to other languages. Specifically, we translate SemCor (Miller et al., 1993) into the target languages with the Google translation tool[2] (Wu et al., 2016).

There are many research alignment methods to acquire aligned words across languages based on parallel corpora (Dyer et al., 2013; Östling and Tiedemann, 2016; Luan et al., 2020; Dou and Neubig, 2021). We use the early FastAlign tool[3] (Dyer et al., 2013) to align the words across languages for simplicity. Through the process, we propagate the annotations from English to multiple languages. The weak supervision signal in the transferred annotations can be further utilized in supervised systems.

An example of the context translation alignment and sense mapping between English and French is shown in Figure 4. Note that our method is language-independent and thus can be applied to many languages. For evaluating the MWSD system on SemEval-13 and SemEval-15, we apply the method to four languages, German (DE), French (FR), Spanish (ES), and Italian (IT).

3.3 Model Overview

The MWSD system mainly consists of an mBERT-UNI model, which is built upon the biencoder

[2]https://translate.google.com
[3]https://github.com/clab/fast_align

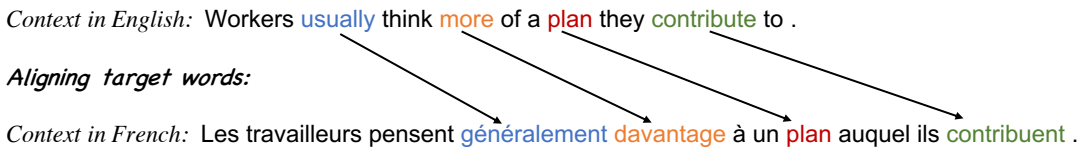

Figure 4: Example of training corpora preparation. The training corpora are first translated into multiple languages by the google translation tool. Then FastAlign tool is applied to find the alignment of labeled target words in English context to other languages. The matched aligned words in different languages are shown in the same color.

model for the English WSD task (Blevins and Zettlemoyer, 2020), where one encoder for encoding multilingual context and the other encoder for encoding unified gloss knowledge. The difference is that we apply the model to construct unified representations which can be used across languages. Both encoders apply the multilingual BERT (mBERT) transformer. mBERT (Kenton and Toutanova, 2019) is trained on 104 languages and is commonly used for cross-lingual semantic representation.

The model first extracts representations of context word and candidate sense representations. For a target word w in the language L, the context encoder generates its representation as e_w^L. Specifically, it is the average pooling of target word tokens in the output.

For sense representations, the gloss set of the corresponding target word is fed as input to the gloss encoder. Though different languages vary in the form of contexts, they share the synsets which can be described by gloss knowledge in a single language, e.g., English. The hidden state of [CLS] token in the gloss encoder output is the vector representation of the gloss. The representation of the candidate gloss set is $\{e_{g_1}, \cdots, e_{g_k}\}$.

The similarity scores between embedding of target word and embeddings of its candidate sense set are calculated by the dot product:

$$score^L(w, g_i) = e_w^L \cdot e_{g_i}, i \in [1, k].$$

The candidate sense with the highest score is the predicted sense produced by the system.

The system is trained with cross-entropy loss over the scores after a softmax activation under a supervised paradigm:

$$p_i = \frac{\exp(score^L(w, g_i))}{\sum_{i=1}^{k} \exp(score^L(w, g_i))},$$

$$loss^L(w) = -\sum_{i=1}^{k} \left[y_i \log\left(p_i\right) \right],$$

where y_i is 1 if the ith sense is the correct sense otherwise y_i is 0.

Joint Training Setting. As annotations are scarce for low-resourced languages, we further design the joint language setting to see if the unified sense representation can connect annotations across languages to benefit the MWSD task. Under the joint language training setting, the inputs to the context encoder can be from different sources and languages. In contrast, the gloss encoder still generates representation for the Babel synsets.

4 Experiment Setup

In this section, we introduce the experiment details including the evaluation dataset, the evaluation metric, training corpora, baseline methods, and implementation details.

4.1 Dataset and Evaluation Metric

Following the previous work (Barba et al., 2020), we evaluated the systems with the updated version of SemEval-13 and SemEval-15 [4](WordNet split). Specifically, SemEval-13 contains four low-resourced languages: Italian (IT), Spanish (ES), French (FR), and German (DE), and SemEval-15 contains Italian and Spanish. As no development dataset is provided, we randomly sample a small amount of the test instances as a development set for model selection. The instance number and the distribution of word sense number(#sense) on the test dataset are shown in Table 1. The word #sense distribution is calculated separately on word level and instance level. The test instance with the higher word sense is more difficult than those with lower word sense because there are more senses to be disambiguated. The F1 score(%) is used as the evaluation metric.

[4]https://github.com/SapienzaNLP/mwsd-datasets

Dataset	Inst num	Word avg	Inst avg
SemEval-13-IT	1,490	3.80	5.51
SemEval-13-ES	1,260	4.20	5.52
SemEval-13-FR	1,449	2.36	3.03
SemEval-13-DE	1,076	1.60	2.17
SemEval-15-IT	1,007	4.38	5.27
SemEval-15-ES	1,043	6.17	6.19

Table 1: Distribution of instance numbers, average word #sense on word level, and average word #sense on instance level for SemEval-13 and SemEval-15 test datasets.

Language	EN	DE	FR	IT	ES
Translated	226k	169k	181k	181k	179k
MuLaN	–	245k	311k	416k	452k
MuLaN*	–	221k	270k	343k	394k

Table 2: Number of training instances for our translation based dataset, original MuLaN dataset, and filtered MuLaN* dataset.

4.2 Training Corpora

We utilize two types of automatically generated training datasets in our experiments, the dataset generated by our proposed translation-based method, and the dataset generated by a label propagation method MuLaN (Barba et al., 2020). Details of the dataset are shown in Table 2.

Translated Corpora: SemCor (Miller et al., 1993) is one of the largest annotated English WSD datasets, which contains 226,036 training instances covering 33,362 senses. We use SemCor 3.0 as the translation source. Due to differences in morphology between languages and inaccuracy brought by the alignment tool, a small amount of the annotated senses in English cannot be transferred to other languages. As a result, we get a comparable number of training instances.

MuLaN: MuLaN is one of the most representative works in automatically constructing the training corpus for the MWSD task. MuLaN has a broader coverage of sense keys as it utilizes the BabelNet inventory than SemCor which utilizes the WordNet inventory. Since we utilize the WordNet split of inventories and evaluation datasets, some of the words in the original MuLaN dataset are not in the used inventories. For fair comparison on mBERT-UNI model by inducing lexical gloss knowledge, we keep the instances with target words existing in the provided inventory, resulting in a filtered dataset MuLaN*.

4.3 Baselines

We compare the proposed mBERT-UNI model with the following baseline methods:

1. **BabelNet S1**: This baseline tags the target word with its most common sense. Following the ranking in BabelNet inventory, the top one ranked sense is the most common sense (MCS). The left senses are least common sense (LCS). We denote this frequency-based baseline as "BabelNet S1."

2. **mBERT-CLS**: The model is built on mBERT (Kenton and Toutanova, 2019). The pre-trained language model first extracts feature representation for target words in context sentences. On top of the frozen mBERT representation, a linear classifier is trained to classify the senses of target words. The model cannot be used to disambiguate unseen senses from the training dataset. Therefore, the model always predicts the most common sense for unseen senses as a back-off strategy.

4.4 Implementation Details

Both encoders in mBERT-CLS and mBERT-UNI models are initialized with a pre-trained Bert-base-multilingual-uncased model, which has 110M parameters. For both models, we use the Cross-Entropy loss as the training loss, and Adam (Kingma and Ba, 2015) as the optimization algorithm.

For mBERT-CLS, we fed the concatenation of the last four layers' output from mBERT encoder to a linear classifier. As discussed in (Blevins and Zettlemoyer, 2020), finetuning the mBERT-CLS does not improve the performance on the English WSD classification task. Therefore, we keep mBERT frozen and only train the linear classifier during training. The model is trained with a fixed learning rate $2 \cdot 10^{-5}$ for 50 epochs. The training batch size is 128.

For mBERT-UNI, the unified representation are generated from gloss knowledge in English, collecting from BabelNet and WordNet. For each sense key, BabelNet may have several gloss definitions and we select the source from WordNet for simplicity. The whole model is trained with the learning rate 10^{-5} for 20 epochs. We set the batch size at 40. The experiments are run on RTX 2080 and

Model	SemEval-13				SemEval-15	
	IT	ES	FR	DE	IT	ES
BabelNet S1	53.22	60.32	60.04	76.58	45.38	39.31
mBERT-CLS (Trans)	69.53	**70.32**	67.43	67.57	61.67	57.62
mBERT-UNI (SemCor)	65.70	67.14	**78.61**	79.74	67.23	64.91
mBERT-UNI (Trans)	**70.94**	69.68	77.29	**80.48**	**71.00**	**67.11**

Table 3: Results of mBERT-UNI on SemEval-13 and SemEval-15 test datasets. The training corpora generated from translation are briefly denoted as Trans.

the average running time for each experiment is 40 hours. For collecting glosses of word senses, we use BabelNet API [5].

5 Result Analysis

In this section, we analyze the performances of our proposed mBERT-UNI model in two parts. We first introduce the effects of mBERT-UNI on MWSD task with our generated translated corpora. After that, we present further experiment results on the MWSD task under various settings.

5.1 Results of mBERT-UNI

We present the performance of mBERT-UNI and other baseline methods in Table 3. From the results, we can make the following observations:

1. Compared with BabelNet S1, knowledge and learning-based methods (i.e., mBERT-CLS and mBERT-UNI) can perform better in most languages. Such results show that even though we do not have any annotations for these languages, the corpus we translate from English can serve as a strong weak-supervision signal.

2. The only exception is German, in which BabelNet S1 outperforms mBERT-CLS with translation. As shown in Table 1, this is potentially because words in German typically have much fewer candidate senses than in other languages. As a result, in most instances, simply predicting the most common sense will lead to the correct answer. In this case, the effect of learning is not as significant as in other languages. Even so, by carefully modeling the unified sense representations, the proposed model can still outperform the BabelNet S1 method by a 3.9 % F1 score.

3. Compared with the mBERT-CLS system, the proposed mBERT-UNI model outperforms on five out of six datasets because of additional lexical knowledge from sense representations with the same translated corpora. Though mBERT-CLS has captured the transferred supervised signal from translated corpora, it is still not enough to disambiguate the senses well. By utilizing lexical knowledge from the unified sense definitions, mBERT-UNI can better disambiguate the word senses under a supervised setting.

4. The translated corpora benefit the MWSD system with external multilingual data. Compared to the mBERT-UNI system trained on original English SemCor and trained on the translated corpora, we can find that the system achieves performance gain on five out of six test datasets. This shows that though the machine translation and alignment tools may induce noise in the corpora preparation process, the resulting multilingual corpora still benefits the system on MWSD tasks. Future work may exploit in the direction of acquiring multilingual corpora of higher quality through automatic methods that can still benefit the system.

5.2 Further Analysis on mBERT-UNI

In this section, we conduct further analysis to show the effects of leveraging an additional corpus MuLaN (Barba et al., 2020) on mBERT-UNI, the effects of joint learning, and the performance on Least Common Sense (LCS). Details are as follows.

5.2.1 Effect of Adding MuLaN Corpora

To see if the knowledge brought by the unified sense representations can be helpful under a supervised paradigm with extra training corpora, we conduct experiments on MuLaN. The results are shown in Table 4.

By incorporating the unified sense representation, previous data generation methods such as MuLaN can further boost the performance of MWSD tasks. From the results, we can see even with the

[5] https://babelnet.org/guide

Model	SemEval-13				SemEval-15	
	IT	ES	FR	DE	IT	ES
BabelNet S1	53.22	60.32	60.04	76.58	45.38	39.31
SensEmBERT (Scarlini et al., 2020)	69.80	73.40	77.80	79.20	-	-
OneSeC (Scarlini et al., 2019)	63.45	61.59	65.10	75.84	-	-
MuLaN (Barba et al., 2020)	**77.45**	77.70	80.12	82.09	70.31	68.73
mBERT-CLS (MuLaN*)	69.73	75.87	78.54	82.62	68.82	67.50
mBERT-UNI (MuLaN*)	75.64	**80.24**	81.64	83.27	72.99	**70.47**
mBERT-UNI (Trans+MuLaN*)	76.98	79.44	**82.68**	**83.83**	**74.58**	68.94

Table 4: Results of mBERT-UNI with extra data corpora MuLaN on SemEval-13 and SemEval-15 test dataset. mBERT-CLS (MuLaN*) is the performance on filtered dataset MuLaN*. mBERT-CLS (MuLaN*) is the performance of our implementation with filtered MuLaN as training data. MuLaN is the performance from original paper.

filtered training corpora of smaller size, mBERT-UNI still achieves performance gain over five out of six test datasets compared to MuLaN (Barba et al., 2020). Though the unified sense representations are built based on glosses from the English language only, it can still benefit multiple languages since words share a set of synsets. Future research may continue to find if enriching the sense representations with resources from different languages would still benefit the system.

Moreover, the unified sense representations encoded with the gloss knowledge from BabelNet, are of high quality. SensEmBERT (Scarlini et al., 2020) produced BERT-based sense embeddings by exploiting mostly the semantic relations in BabelNet and Wikipedia for multiple languages separately. Compared with SensEmBERT, our unified sense representation can be simply acquired from the single lexical knowledge source WordNet and even achieves higher performance on the MWSD task.

The mBERT-UNI also supports merging multiple sources of data generation efforts. Combining MuLaN* with our generated dataset, mBERT-UNI can boost the performance on four out of six test datasets. The only exception is Spanish (ES). As shown in Table 1, the test instances in ES are more challenging than in other languages, and thus they are potentially more vulnerable to the noise in the automatically generated training corpora.

5.2.2 Effect of Joint Training

In this section, we conduct experiments to study the effect of the proposed joint learning setting. We are interested in two questions: (1) Whether the joint learning setting can help models solve the MWSD problem or not? (2) Whether the joint learning setting will have a negative effect on English WSD or not. To answer the question, we conduct experiments on training with monolingual datasets and multilingual datasets.

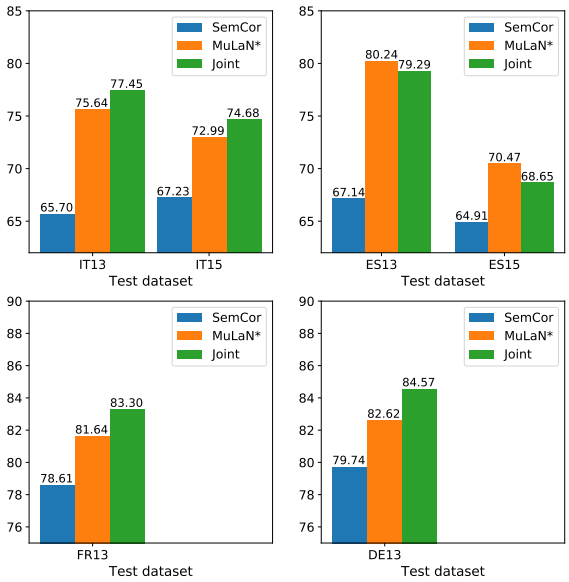

Figure 5: Results of joint learning on the MWSD task. For joint training setting of each language, training data contains SemCor (English) and MuLaN* with the corresponding language part, e.g., SemCor and MuLaN* (Italian) for Italian (*IT13* and *IT15*).

To answer the first question, we present the performances on the MWSD task in Figure 5. We can see that joint learning can achieve better performance on four out of the six datasets and comparable performance on the other two (ES13, ES15). For each language, we combine the MuLaN* with English SemCor as a new training dataset. This result shows that with the unified sense representation, jointly training instances from different languages can improve the annotation usage efficiency across languages. For language ES, the higher

Model	SemEval-13				SemEval-15	
	IT	ES	FR	DE	IT	ES
mBERT-CLS (Trans)	60.02	63.11	57.16	45.58	49.26	53.21
mBERT-UNI (Trans)	62.59	61.92	67.38	59.83	63.90	64.15
mBERT-CLS (MuLaN*)	61.45	68.74	68.36	64.38	59.02	63.61
mBERT-UNI (MuLaN*)	68.24	75.81	72.77	66.38	66.83	63.47

Table 5: Results of Least Common Senses (LCS) on SemEval-13 and SemEval-15 test dataset.

Dataset	SemCor	+IT	+ES	+FR	+DE
ALL	75.70	75.94	75.82	75.76	75.31
Δ	-	+0.24	+0.12	+0.06	-0.39

Table 6: Results of joint learning on the ALL test dataset of English WSD task. *+IT* means that training dataset contains the SemCor (English) and MuLaN* (Italian).

word #sense of the test instances may account for the performance drop under a joint training setting than using MuLaN* alone.

To answer the second question, we report the performance of the mBERT-UNI model trained with English SemCor as well as another joint trained setting on the all-words English WSD datasets proposed by (Raganato et al., 2017). The test dataset "ALL" covers all five datasets, including semeval 2007 (Pradhan et al., 2007), senseval-2 (Palmer et al., 2001), senseval-3 (Snyder and Palmer, 2004), semeval2012 (Navigli et al., 2013), and semeval2015 (Moro and Navigli, 2015). We show the results in Table 6. We can see that the overall performance in English is comparable in different settings. Since the MuLaN dataset is specially designed for other languages and the propagated annotations mainly come from SemCor, the joint training does not benefit the English WSD task much. However, joint training enables a single mBERT-UNI to generate unified sense representations, which can be used in disambiguating word senses in multiple languages. In future work, the unified sense representation may be applied in cross-lingual representation learning.

5.2.3 LCS Analysis

In this section, we analyze the influence of unified sense representation on the performance of the least common senses (LCS). We split the test instances into two parts, one part with annotation of BabelNet S1 and one part with annotations except BabelNet S1 as least common senses. Compared with most common senses, less common ones are more difficult to disambiguate for MWSD systems because of fewer training instances on average.

We show the performances on the two groups of different systems are shown in Table 5. Comparing mBERT-CLS and mBERT-UNI, adding the sense representations can help improve models' performance significantly on the least common senses. The improvement is consistent on different training corpora for both the translated corpora and MuLaN. It can be concluded that unified sense representation with lexical knowledge improves the ability of deep models to disambiguate the least common senses. This is because mBERT-UNI can still generate and learn unique sense representations for the least common senses even with no or very few training instances. However, while the systems achieve decent performance on the overall performance, disambiguating the lease common senses is still a challenging problem.

6 Conclusion and Future Work

In this paper, to build feasible knowledge and supervised based systems for multilingual word sense disambiguation, we propose to construct unified sense representation by utilizing Babel synsets, and transferred annotations from rich source languages by machine translation and alignment tools. With the unified representations, previous data generation efforts can be combined and further boost the performance. Moreover, annotations from different languages can be jointly trained and benefit the multilingual word sense disambiguation task. Experiments on standard evaluation multilingual word sense disambiguation benchmarks demonstrate the effectiveness of the proposed method.

Future work can be extended on how to induce more lexical knowledge from various languages to improve the representation learning. Moreover, based on the fact that multiple languages share a set of concepts described by Babel synsets, the generated representations may benefit cross lingual representation for other natural language understanding tasks.

References

Satanjeev Banerjee, Ted Pedersen, et al. 2003. Extended gloss overlaps as a measure of semantic relatedness. In *Ijcai*, volume 3, pages 805–810. Citeseer.

Edoardo Barba, Luigi Procopio, Niccolo Campolungo, Tommaso Pasini, and Roberto Navigli. 2020. Mulan: Multilingual label propagation for word sense disambiguation. In *Proc. of IJCAI*, pages 3837–3844.

Michele Bevilacqua and Roberto Navigli. 2020. Breaking through the 80% glass ceiling: Raising the state of the art in word sense disambiguation by incorporating knowledge graph information. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 2854–2864.

Michele Bevilacqua, Tommaso Pasini, Alessandro Raganato, Roberto Navigli, et al. 2021. Recent trends in word sense disambiguation: A survey. In *Proceedings of the Thirtieth International Joint Conference on Artificial Intelligence, IJCAI-21*. International Joint Conference on Artificial Intelligence, Inc.

Terra Blevins and Luke Zettlemoyer. 2020. Moving down the long tail of word sense disambiguation with gloss informed bi-encoders. In *ACL*.

Claudio Delli Bovi, Luis Espinosa Anke, and Roberto Navigli. 2015. Knowledge base unification via sense embeddings and disambiguation. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 726–736.

Zi-Yi Dou and Graham Neubig. 2021. Word alignment by fine-tuning embeddings on parallel corpora. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*, pages 2112–2128.

Chris Dyer, Victor Chahuneau, and Noah A Smith. 2013. A simple, fast, and effective reparameterization of ibm model 2. In *Proceedings of the 2013 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 644–648.

Andreas Eisele and Yu Chen. 2010. Multiun: A multilingual corpus from united nation documents. In *LREC*.

William A Gale, Kenneth W Church, and David Yarowsky. 1992. A method for disambiguating word senses in a large corpus. *Computers and the Humanities*, 26(5):415–439.

Bradley Hauer, Grzegorz Kondrak, Yixing Luan, Arnob Mallik, and Lili Mou. 2021. Semi-supervised and unsupervised sense annotation via translations. In *Proceedings of the International Conference on Recent Advances in Natural Language Processing (RANLP 2021)*, pages 504–513.

Luyao Huang, Chi Sun, Xipeng Qiu, and Xuan-Jing Huang. 2019. Glossbert: Bert for word sense disambiguation with gloss knowledge. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 3509–3514.

Ignacio Iacobacci, Mohammad Taher Pilehvar, and Roberto Navigli. 2016. Embeddings for word sense disambiguation: An evaluation study. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 897–907.

Jacob Devlin Ming-Wei Chang Kenton and Lee Kristina Toutanova. 2019. Bert: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of NAACL-HLT*, pages 4171–4186.

Diederik P Kingma and Jimmy Ba. 2015. Adam: A method for stochastic optimization. In *ICLR (Poster)*.

Sawan Kumar, Sharmistha Jat, Karan Saxena, and Partha Talukdar. 2019. Zero-shot word sense disambiguation using sense definition embeddings. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 5670–5681.

Helen Langone, Benjamin R Haskell, and George A Miller. 2004. Annotating wordnet. In *Proceedings of the Workshop Frontiers in Corpus Annotation at HLT-NAACL 2004*, pages 63–69.

Frederick Liu, Han Lu, and Graham Neubig. 2018. Handling homographs in neural machine translation. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 1336–1345.

Daniel Loureiro and Alipio Jorge. 2019. Language modelling makes sense: Propagating representations through wordnet for full-coverage word sense disambiguation. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 5682–5691.

Yixing Luan, Bradley Hauer, Lili Mou, and Grzegorz Kondrak. 2020. Improving word sense disambiguation with translations. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 4055–4065.

Fuli Luo, Tianyu Liu, Qiaolin Xia, Baobao Chang, and Zhifang Sui. 2018. Incorporating glosses into neural word sense disambiguation. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 2473–2482.

George A Miller. 1998. *WordNet: An electronic lexical database*. MIT press.

George A Miller, Claudia Leacock, Randee Tengi, and Ross T Bunker. 1993. A semantic concordance. In *HUMAN LANGUAGE TECHNOLOGY: Proceedings of a Workshop Held at Plainsboro, New Jersey, March 21-24, 1993*.

Andrea Moro and Roberto Navigli. 2015. Semeval-2015 task 13: Multilingual all-words sense disambiguation and entity linking. In *Proceedings of the 9th international workshop on semantic evaluation (SemEval 2015)*, pages 288–297.

Roberto Navigli, David Jurgens, and Daniele Vannella. 2013. Semeval-2013 task 12: Multilingual word sense disambiguation. In *Second Joint Conference on Lexical and Computational Semantics (* SEM), Volume 2: Proceedings of the Seventh International Workshop on Semantic Evaluation (SemEval 2013)*, pages 222–231.

Roberto Navigli and Simone Paolo Ponzetto. 2012. Babelnet: The automatic construction, evaluation and application of a wide-coverage multilingual semantic network. *Artificial Intelligence*, 193:217 – 250.

Robert Östling and Jörg Tiedemann. 2016. Efficient word alignment with markov chain monte carlo. *The Prague Bulletin of Mathematical Linguistics*.

Martha Palmer, Christiane Fellbaum, Scott Cotton, Lauren Delfs, and Hoa Trang Dang. 2001. English tasks: All-words and verb lexical sample. In *Proceedings of SENSEVAL-2 Second International Workshop on Evaluating Word Sense Disambiguation Systems*, pages 21–24.

Tommaso Pasini. 2020. The knowledge acquisition bottleneck problem in multilingual word sense disambiguation. In *IJCAI*, pages 4936–4942.

Tommaso Pasini and Roberto Navigli. 2020. Train-o-matic: Supervised word sense disambiguation with no (manual) effort. *Artificial Intelligence*, 279:103215.

Tommaso Pasini, Alessandro Raganato, Roberto Navigli, et al. 2021. Xl-wsd: An extra-large and cross-lingual evaluation framework for word sense disambiguation. In *Proceedings of the AAAI Conference on Artificial Intelligence*. AAAI Press.

Sameer Pradhan, Edward Loper, Dmitriy Dligach, and Martha Palmer. 2007. Semeval-2007 task-17: English lexical sample, srl and all words. In *Proceedings of the fourth international workshop on semantic evaluations (SemEval-2007)*, pages 87–92.

Xiao Pu, Nikolaos Pappas, James Henderson, and Andrei Popescu-Belis. 2018. Integrating weakly supervised word sense disambiguation into neural machine translation. *Transactions of the Association for Computational Linguistics*, 6:635–649.

Alessandro Raganato, Jose Camacho-Collados, and Roberto Navigli. 2017. Word sense disambiguation: A unified evaluation framework and empirical comparison. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 1, Long Papers*, pages 99–110.

Bianca Scarlini, Tommaso Pasini, and Roberto Navigli. 2019. Just "onesec" for producing multilingual sense-annotated data. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 699–709.

Bianca Scarlini, Tommaso Pasini, and Roberto Navigli. 2020. Sensembert: Context-enhanced sense embeddings for multilingual word sense disambiguation. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pages 8758–8765.

Benjamin Snyder and Martha Palmer. 2004. The english all-words task. In *Proceedings of SENSEVAL-3, the Third International Workshop on the Evaluation of Systems for the Semantic Analysis of Text*, pages 41–43.

Kaveh Taghipour and Hwee Tou Ng. 2015. One million sense-tagged instances for word sense disambiguation and induction. In *Proceedings of the nineteenth conference on computational natural language learning*, pages 338–344.

Jörg Tiedemann, Santhosh Thottingal, et al. 2020. Opus-mt–building open translation services for the world. In *Proceedings of the 22nd Annual Conference of the European Association for Machine Translation*. European Association for Machine Translation.

Yonghui Wu, Mike Schuster, Zhifeng Chen, Quoc V Le, Mohammad Norouzi, Wolfgang Macherey, Maxim Krikun, Yuan Cao, Qin Gao, Klaus Macherey, et al. 2016. Google's neural machine translation system: Bridging the gap between human and machine translation. *arXiv preprint arXiv:1609.08144*.

A Transition-based Method for Complex Question Understanding

Yu Xia[1,*], **Wenbin Jiang**[2], **Yajuan Lyu**[2], **Sujian Li**[1,†]
[1]MOE Key Lab of Computational Linguistics, Peking University, Beijing, China
[2]Baidu Inc., Beijing, China
{yuxia, lisujian}@pku.edu.cn
{jiangwenbin, lvyajuan}@baidu.com

Abstract

Complex Question Understanding (CQU) parses complex questions to Question Decomposition Meaning Representation (QDMR) which is a sequence of atomic operators. Existing works are based on end-to-end neural models which do not explicitly model the intermediate states and lack interpretability for the parsing process. Besides, they predict QDMR in a mismatched granularity and do not model the step-wise information which is an essential characteristic of QDMR. To alleviate the issues, we treat QDMR as a computational graph and propose a transition-based method where a *decider* predicts a sequence of actions to build the graph node-by-node. In this way, the partial graph at each step enables better representation of the intermediate states and better interpretability. At each step, the *decider* encodes the intermediate state with specially designed encoders and predicts several candidates of the next action and its confidence. For inference, a *searcher* seeks the optimal graph based on the predictions of the *decider* to alleviate the error propagation. Experimental results demonstrate the parsing accuracy of our method against several strong baselines. Moreover, our method has transparent and human-readable intermediate results, showing improved interpretability.

1 Introduction

The task of complex question understanding (CQU) aims at converting complex questions which require multi-hop reasoning into consecutive trivial sub-questions. An example of CQU is shown in Figure 1. To answer the question "return me the author in the University of Michigan whose papers have more than 5000 total citations", CQU models decompose it into several trivial sub-questions (*e.g.* "return authors"), and the final answer is obtained by consecutively answering the sub-questions. To capture the meaning of questions over unstructured sources such as text and images, Question Decomposition Meaning Representation (QDMR) (Wolfson et al., 2020) is proposed where questions are represented through a sequence of atomic executable operators, and the final answer can be obtained by answering the operator sequences in order. QDMR has been shown to improve the performance and interpretability for multi-hop question answering (Hasson and Berant, 2021; Subramanian et al., 2020; Talmor et al., 2021).

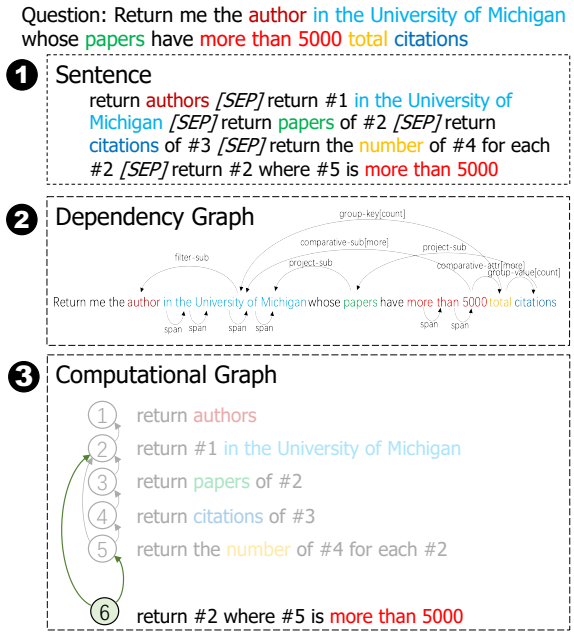

Figure 1: An example of CQU consists of a complex question and different modeling of QDMR.

Existing works for CQU can be roughly divided into two categories: the seq2seq-based autoregressive parser (Wolfson et al., 2020) and the dependency-based non-autoregressive parser (Hasson and Berant, 2021). However, these approaches are based on end-to-end models which do not explicitly model the intermediate states and lack interpretability for the parsing process. Besides, they

[*] This work was done during internship at Baidu Inc.
[†] Corresponding author.

Proceedings of the 29th International Conference on Computational Linguistics, pages 4203–4211
October 12–17, 2022.

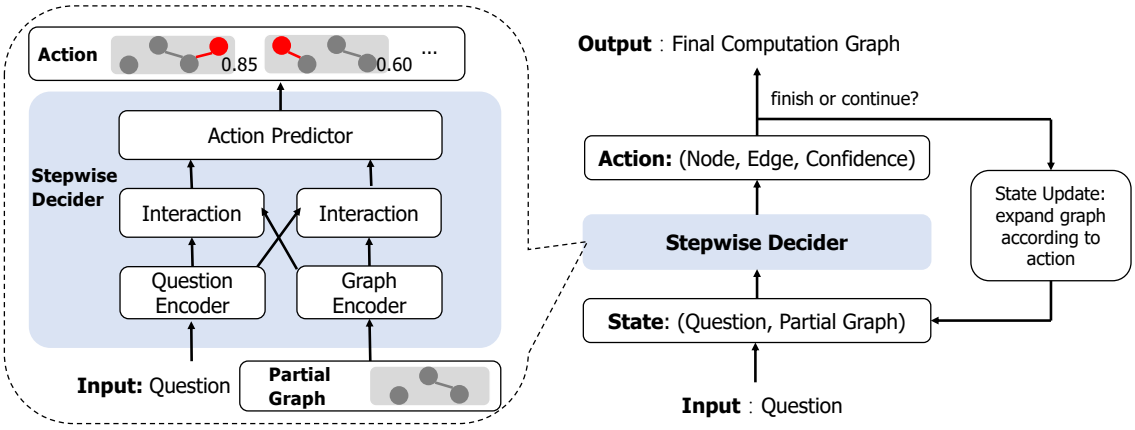

Figure 2: An overview of our transition-based framework.

predict QDMR in a mismatched granularity. The former category generates QDMR as a sentence (as shown in Figure 1 ❶) and adopts the seq2seq model to decode the QDMR token-by-token. This token-level modeling is sub-optimal since it ignores the inherent operator-level structure of QDMR and thus performs worse when the QDMR has longer operators and when the question is informative. The latter category maps QDMR to a dependency graph over the question tokens (as shown in Figure 1 ❷) and adopts a non-autoregressive parser to decode the entire dependency graph in a single step. It predicts all operators in the QDMR simultaneously and ignores the interaction between operators, trading off performance for computational efficiency. (Hasson and Berant, 2021) also tries combining the two approaches by exploiting the graph supervision to train the encoder in the seq2seq model. To sum up, these methods have drawbacks in modeling the intermediate states and the step-wise information which is a distinct characteristic of QDMR.

To alleviate the shortcomings of the above methods while preserving their advantages, we treat QDMR as a computational graph and propose a transition-based method where a *decider* predicts a sequence of actions to build the graph node-by-node. At each transition step, one new node and its referencing edges are decided given the question and the previously generated partial graph. In this way, the partial graph at each step enables better representation of the intermediate states and better interpretability. We illustrate the proposed method in Fig. 2. The generated graph starts from empty and expands incrementally in a node-by-node manner. At each step, given the question and the current state, the *step-wise decider* encodes them with specially designed encoders and predicts several candidates for the next action which includes a node, its connecting edges, and its confidence. After each step, the partial graph is either expanded according to the action or finalized as a QDMR. For inference, a *searcher* seeks the optimal graph based on the predictions of the *decider* to alleviate the error propagation.

To verify the effectiveness of our proposed method, abundant experiments are conducted on the BREAK dataset, which contains 83,978 examples from ten QA datasets across three modalities. Experimental results show that our method outperforms strong baselines and achieves the state-of-the-art on the BREAK dataset. We further analyze the interpretability of our method. Overall, our work makes the following major contributions:

1. To the best of our knowledge, we are the first to investigate the transition-based method for CQU by modeling the intermediate states which facilitate better encoding and better interpretability.

2. Experiments on BREAK demonstrate the parsing accuracy of our method against strong baselines. Moreover, further analysis and visualization verify the interpretability.

2 Method

2.1 Problem Definition

Given a question with n tokens, $q = (q_1, q_2, \ldots, q_n)$, the goal of CQU is to parse it to its QDMR. In this work, we treat QDMR as a computational graph $G = \langle V, E \rangle$ where V

and E are the node set and edge set. A node $v^i \in V$ is a sequence of $|v^i|$ tokens $v^i = v^i_1 ... v^i_{|v^i|}$, where token v^i_j is either a question token $\in \mathcal{V}_q$ (or some inflection of it), a word from a constant predefined lexicon $\in \mathcal{V}_{const}$, or a reference token $\in \mathcal{V}^i_{ref} = \{\#1, ..., \#(i-1)\}$ referring to a previous step. A directed edge $e^{ij} \in E$ pointing from v^i to v^j is the reference token $\#j$ in v^i.

2.2 Overview

To address the aforementioned challenges in this task, we propose a neural transition-based model for CQU which decides a new node along with its connections to existing nodes at each step to incrementally build a computational graph. The construction process is briefly illustrated in Fig. 2. At each step i, given the question q and the current state $s_i = \{G^i\}$ where $G^i = \langle V^i, E^i \rangle$ denotes the generated partial graph, the *stepwise decider* predicts several candidates of the next action $a^{i+1} = \langle v^{i+1}, \Delta E^i, p^{i+1} \rangle$, where v^{i+1} denotes the next node, ΔE^i denotes the edges start from v^{i+1} and p^{i+1} denotes the confidence of the action. Then, we expand the graph from G^i to $G^{i+1} = \langle V^i + v^{i+1}, E^i + \Delta E^i \rangle$ and update the current state according to the predicted action. We repeat the above iteration until the end action is predicted. In inference, we adopt a *searcher* to maintain and seek the optimal graphs based on the node confidence at each step.

2.3 State Representation

At each step i, given the question q and the current state $s_i = \{G^i\}$ where q denotes the question and $G^i = \langle V^i, E^i \rangle$ denotes the partial graph, we firstly use two encoders to obtain their representation respectively. Then, we feed them into a dual interaction layer to update them dynamically.

2.3.1 Question Encoder

We feed the input question $q = (q_1, q_2, \ldots, q_n)$ into the Transformer encoder of a pretrained seq2seq model (e.g. BART) to get the contextual representation matrix $H^q \in \mathbb{R}^{n \times d_h}$, where d_h is the dimension of the hidden states in BART and n is the length of question. In this way, question q can be represented as $H^q = \{\mathbf{h}^q_1, \mathbf{h}^q_2, \ldots, \mathbf{h}^q_n\}$, where $\mathbf{h}^q_i$ is the contextual representation of the i-th token of q. We call H^q static question representation to distinguish it with the dynamically updated question representation H^{q^i} introduced in section 2.3.3.

2.3.2 Graph Encoder

We treat the incrementally expanding graph as a sequence of actions in the chronological order of when they are added in. We adopt the order given in the dataset. We utilize the Transformer decoder to serve as the graph encoder . Concretely, we use the masked self-attention mechanism to ensure that the representation of the node and edges at step i takes all previous nodes and edges in G^{i-1} into consideration.

Formally, given the graph G^i, we get the input tokens of the linearized graph by seperating actions with special tokens to indicate their boundaries: $([\mathrm{A}_1], v^1_1, \ldots, v^1_{|v^1|}, [\mathrm{A}_2], \ldots, [\mathrm{A}_i], v^i_1, \ldots, v^i_{|v^i|})$. If v^i is the last action in the graph, we append a special token [END] to indicate the termination of the parsing process. We feed the input tokens into the graph encoder to get the contextual representation matrix $H^g \in \mathbb{R}^{p \times d_h}$, where p is the length of the input tokens. In this way, G^i can be represented as $H^{G^i} = \{\mathbf{h}^{G^i}_1, \mathbf{h}^{G^i}_2, \ldots, \mathbf{h}^{G^i}_p\}$, where $\mathbf{h}^{G^i}_i$ is the contextual representation of the i-th token of the input tokens. We repeat the above encoding every time a new token in the next action i.e. v^{i+1}_j is generated to integrate the partial action semantic.

2.3.3 Interaction Layer

We observe that different nodes tend to use different parts of the question, and that question tokens already present in the partial graph are less likely to be chosen in the later nodes. To model this observation, we apply the scaled dot-product attention proposed in (Vaswani et al., 2017) to dynamically update the question representation according to the generated partial graph.

$$\begin{aligned} \alpha^i_1 &= \mathrm{softmax}(\frac{W^K_1 H^q (W^Q_1 H^{G^i})^\mathsf{T}}{\sqrt{d_k}}), \\ H^{q^i} &= \alpha^i_1 W^V_1 H^q \end{aligned} \tag{1}$$

where $\{W^Q_1, W^K_1\} \in \mathbb{R}^{d_h \times d_h}$ denote learnable matrices that transform the graph and question representation into the query and key subspace respectively. The attention weights over all question tokens $\alpha^i \in \mathbb{R}^{|q|}$ softly indicate whether a token is already present in the partial graph. $W^V_1 \in \mathbb{R}^{d_h \times d_h}$ denotes the learnable matrix that projects the question representation into the value subspace, and the projected value vectors are averaged according to α^i to get the updated question tokens representation H^i. Similarly, to make the representation of

G^i attends to all question tokens in q, we apply the source-attention mechanism which takes the output of the question encoder as the key:

$$\alpha_2^i = \text{softmax}(\frac{W_2^K H^{G^i}(W_2^Q H^q)^\mathsf{T}}{\sqrt{d_k}}), \quad (2)$$
$$\hat{H}^{G^i} = \alpha_2^i W_2^V H^{G^i}$$

Finally, we apply MaxPooling over all question tokens representation and node tokens representation to obtain the final representation for action prediction:

$$h_{\text{final}} = \text{MaxPooling}(H^{q^i}, \hat{H}^{G^i}) \quad (3)$$

where the MaxPooling is performed on the first dimension and $h_{\text{final}} \in \mathbb{R}^{d_h}$.

2.4 Action Prediction

At each step i, given the representation of the current state h_{final}, we use the action predictor to predict several candidates of the next action $a^{i+1} = \langle v^{i+1}, \Delta E^i, p^{i+1} \rangle$, where v^{i+1} denotes the next node, ΔE^i denotes the edges start from v^{i+1} and p^{i+1} denotes the confidence of the action.

2.4.1 Node Prediction

Note that as mentioned in 2.1 , a node v^{i+1} is a sequence of $|v^{i+1}|$ tokens $v^{i+1} = (v_1^{i+1}, \ldots, v_{|v^{i+1}|}^{i+1})$, where token v_j^{i+1} is either a question token $\in \mathcal{V}_q$ (or some inflection of it), a word from a constant predefined lexicon $\in \mathcal{V}_{const}$, or a reference token $\in \mathcal{V}_{ref}^{i+1} = \{\#1, \ldots, \#i\}$ referring to a previous step. Among them, the reference token $\#j$ also belongs to the edge tokens e^{ij}. Therefore, we decompose the prediction of the reference tokens into two stages. In the node prediction stage, we predict #R indicating a placeholder for the edge token. Then, in the edge prediction stage, we predict the exact number of the reference token to replace R and get $\#j$. The probability over vocabulary $\mathcal{V} = \mathcal{V}_q \bigcup \mathcal{V}_{const} \bigcup \{\#\text{R}\}$ can be obtained by:

$$P(v_z^{i+1}|v_{<z}^{i+1}, q, G^i, \theta) = \text{softmax}(W^P h_{\text{final}} + b^P) \quad (4)$$

where $W^P \in \mathbb{R}^{d_k \times |\mathcal{V}|}, b^P \in \mathbb{R}^{|\mathcal{V}|}$ are learnable parameters that transform the final representation into the probability over $\mathcal{V}$. The model is parameterized by θ. $v_{<z}^{i+1}$ denotes the partial action $\{v_0^{i+1}, \ldots, v_{j-1}^{i+1}\}$.

2.4.2 Edge Prediction

Instead of treating the reference tokens as static tokens in the vocabulary as the previous works, which shares the embeddings among different examples and thus ignores their semantics. We obtain their representation dynamically according to the constructed partial graph. Specifically, we average the representation of $v_1^i \ldots v_{|v^i|}^i$ to obtain the representation of $\#i$ which is denoted as $h_{\#i}$. Then, we adopt a bilinear function to compute the similarity between each reference representation $h_{\#i}$ and the final representation h_{final}. The probability over $\mathcal{V}_{ref}^{i+1}$ can be obtained by:

$$P(v_z^{i+1}|v_{<z}^{i+1}, q, G^i, \theta) = \text{softmax}(E^{\text{ref}} W^{\text{ref}} h_{\text{final}})$$
$$j = \text{argmax}_{j \in \mathcal{V}_{ref}^{i+1}} P(j|v_{<z}^{i+1}, q, G^i, \theta) \quad (5)$$

where $E^{\text{ref}} \in \mathbb{R}^{i \times d_h}$ denotes the embedding matrix of the reference tokens. $W^{\text{ref}} \in \mathbb{R}^{d_h \times d_h}$ is a learnable matrix. We use the predicted reference number j to replace R for v_z^{i+1} if it is a #R.

2.4.3 Action Confidence

We apply beam search with beam size K_1 sampling on $P(v_z^{i+1}|v_{<z}^{i+1}, q, G^i, \theta)$ to get top K_1 candidate actions. The confidence of each candidate action a^{i+1} is defined as the probability of the predicted sequence i.e. the product of the probabilities of the predicted tokens $v_1^{i+1} \ldots v_{|v^{i+1}|}^{i+1}$:

$$P(a^{i+1}|q, G^i, \theta) = \prod_{z=1}^{|v^{i+1}|} P(v_z^{i+1}|v_{<z}^{i+1}, q, G^i, \theta) \quad (6)$$

2.5 Training

We train our transition-based model with the standard maximum likelihood estimate using teacher forcing. In other words, we maximize the sum of the stepwise action confidence. The loss *w.r.t* an example is defined as follows:

$$P(G|q, \theta) = \prod_{i=1}^{|G|} P(a^{i+1}|q, G^i, \theta)$$
$$\mathcal{L}(G|q, \theta) = -\frac{1}{m} \sum_{i=1}^{m} logP(a^{i+1}|q, G^i, \theta) \quad (7)$$

where $G = (a^1, \ldots, a^m) = ([\text{A}_1], v_1^1, \ldots, v_{|v^1|}^1, [\text{A}_2], \ldots, [\text{A}_m], \ldots, v_{|v^m|}^m, [\text{END}])$ and m denotes the number of the action in G.

2.6 Inference

The whole inference procedure is shown in Algorithm 1. At line $1 \sim 3$, we first obtain the question representation and initialize the output with an empty graph. In the loop from line 5, we predict an action at each transition step until the [END] is generated. At each transition step, we first initialize the predicted action v_0^{i+1} with $[A_i]$. Then, in the loop from line 9, we generate a token at each step until the $[\mathrm{A}_{i+1}]$ or [END] indicating the termination of an action is predicted. At line $10 \sim 15$, we obtain the partial graph representation and the final representation. At line $18 \sim 21$, we first predict v_z^{i+1} by sampling on Eq. 4. Then, we decide the exact number by Eq. 5 if v_z^{i+1} is a #R token. At line 24, we obtain topK_2 a^{i+1} by sampling on Eq. 7. Finally, we add the action predictions to G^i and get the new graph G^{i+1}.

Note that in the above inference procedure, we adopt a *searcher* which seeks for high-probability output graphs to relieve the error propagation and guide the direction of the graph expansion. Inspired by the traditional beam search which decodes a single token at each search step, we replace the original generation probability with the action confidence defined in Eq. 6. At step i, we maintain K_2 candidate actions sampling on the probability $P(a^{i+1}|q, G^i, \theta)$ where K_2 denotes the beam size.

Algorithm 1 Inference procedure of our framework

Require: a question with n tokens $q = (q_1, q_2, ..., q_n)$.
Ensure: topK_2 computational graphs.

1: $H^q \leftarrow$Question-Encoder(q);
2: $G^0 \leftarrow (V^0, E^0), V^0 \leftarrow \emptyset, E^0 \leftarrow \emptyset$;
3: $i \leftarrow 1$;
4: // Generating topK_2 Action Sequence
5: **while** the last action a^{i+1} is not [END] **do**
6: $\quad v_0^{i+1} \leftarrow [A_i]$;
7: $\quad z \leftarrow 1$;
8: $\quad$ // Generating topK_1 Action Candidates
9: $\quad$ **while** the last token v_{z-1}^{i+1} is not in $\{[\mathrm{A}_{i+1}], [\mathrm{END}]\}$ **do**
10: $\quad\quad G^i \leftarrow G^i + \{v^{i+1}\}$;
11: $\quad\quad$ // Partial Graph Encoding
12: $\quad\quad H^{G^i} \leftarrow$Graph-Encoder$(G^i, q)$;
13: $\quad\quad$ // Interaction
14: $\quad\quad H^{q^i} \leftarrow$Attention$(H^q, H^{G^i})H^q$;
15: $\quad\quad \hat{H}^{G^i} \leftarrow$Attention$(H^{G^i}, H^q)H^{Gi}$;
16: $\quad\quad h_{\text{final}} \leftarrow \mathrm{MaxPooling}(H^{q^i}, \hat{H}^{G^i})$;
17: $\quad\quad$ // Node & Edge Prediction
18: $\quad\quad$ get topK_1 v_z^{i+1} by a search step on Eq. 4;
19: $\quad\quad$ **if** v_z^{i+1}=#R **then**
20: $\quad\quad\quad$ get the number of #R by Eq. 5.
21: $\quad\quad$ **end if**
22: $\quad\quad z \leftarrow z + 1$;
23: $\quad$ **end while**
24: $\quad$ get topK_2 a^{i+1} by a search step on Eq. 7;
25: $\quad G^{i+1} \leftarrow G^i + \{a^{i+1}\}$;
26: $\quad i \leftarrow i + 1$;
27: **end while**

3 Experiment

3.1 Experimental Setup

Datasets and Metrics Our evaluation is conducted on the dataset BREAK. The question-QDMR pairs are crowd-sourced based on questions sampled from ten widely-used QA datasets. It consists of 83,978 examples including 60,150 examples with QDMR and 23,828 examples with high-level QDMR. We do not include examples with high-level QDMR for the sake of a fair comparison with the previous work.The QA datasets included in BREAK and the statistics are listed in Table 1. The train/dev/test sets are partitioned following the original datasets. The distribution over QDMR sequence length is listed in Table 4. Note that the gold answers of the test set are not publicly available, so we report performance on the development set.

Metrics Following the previous work (Hasson and Berant, 2021), we use Normalized Exact Match (NormEM) and Logical Form Exact Match (LF-EM) as the evaluation metrics. **Normalized Exact Match (NormEM)**: The predicted and gold QDMRs are first normalized by a rule-based procedure, and then exact string match is computed between the two normalized QDMRs. The value for each sample is either 0 or 1. **Logical Form Exact Match (LF-EM)**: The predicted and gold QDMRs are first converted to the logical form by a rule-based procedure, and then exact string match is computed between the two logical form QDMRs. The value for each sample is either 0 or 1.

Implementation Details We follow the previous work (Hasson and Berant, 2021) for implementation. We use BART-base (Lewis et al., 2019) as our backbone. The models are implemented in Pytorch (Paszke et al., 2019) and are trained on a single

	BREAK									
	ACADEMIC	ATIS	CLEVR	COMQA	GEO	CWQ	DROP	NLVR2	SPIDER	Total
Train	195	4042	9453	3546	547	1985	7683	9915	6955	44321
Dev	-	457	2215	988	50	475	1268	1805	502	7760
Test	-	407	2267	986	280	528	1279	1797	525	8069

Table 1: The question distribution of each QD dataset in BREAK.

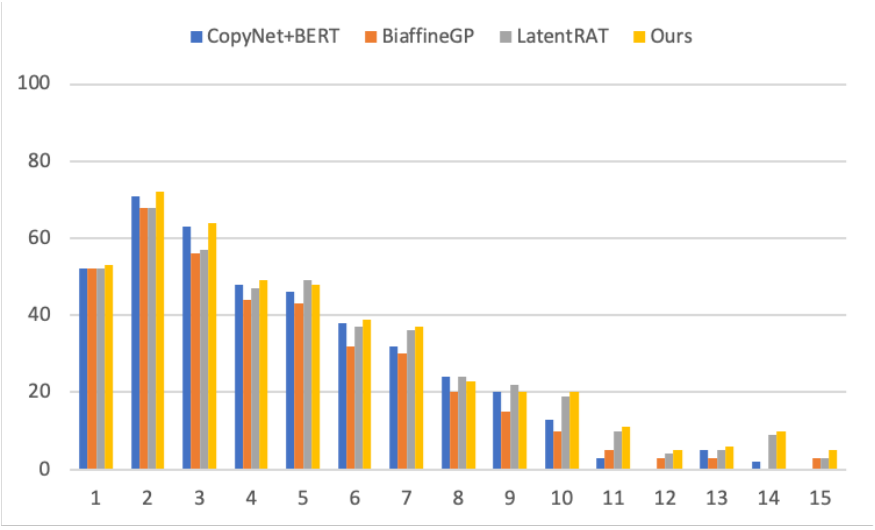

Figure 3: The average LF-EM of our method and baselines for different lengths of QDMR on the dev set.

Model	NormEM	LF-EM
CopyNet+BERT	37.3	47.4
BiaffineGP	-	45.3
Latent-RAT	35.6	46.9
BART	38.1	47.7
Ours w/o Beam Search	38.9	48.1
Ours w/ Beam Search	**41.2**	**49.7**

Table 2: NormEM and LF-EM on the dev set.

GeForce RTX 3090 GPU. We set the batch size as 32, and the max training epoch number as 20 with early stopping (patience=5). We utilize Adam optimizer (Kingma and Ba, 2014) with a dynamic learning rate according to the slanted triangular schema. The beam size of both inner and outer beam search is set to 5.

Baselines We compare our framework with various previous works in terms of parsing accuracy and interpretability.

CopyNet+BERT is a seq2seq model consisting of a BERT encoder and an LSTM decoder with a copy mechanism. **BiaffineGP** is based on the biaffine dependency parser of (Dozat and Manning, 2018) except that it predicts a DAG and not a tree. Besides, it applies an Integer Linear Programming layer on top of it to eliminate constraint violations in the output graph. **Latent-RAT** is based on RAT transformer layers (Shaw et al., 2018; Wang et al., 2019) to predict the graph structure using the encoder and predict the QDMR sequence using the decoder. **BART** is based on the pretrained seq2seq model BART.

3.2 Results

3.2.1 Main Results

Table 2 shows the overall performance of our method and all the baselines on the development set of BREAK. Our method achieves the best results among the recently available methods. Specifically, our method without beam search achieves comparable performance to BART with advantages of 0.8 NormEM and 0.4 LF-EM and outperforms CopyNet+BERT by 1.6 NormEM and 0.7 LF-EM. We attribute the performance gain to the better modeling of the question and the current state. Enhanced by beam search, our method exceeds BART by 3.1 NormEM and 2.0 LF-EM which demonstrates the potential of increasing the search space.

Question	Wrong Prediction	Ours

How many was the difference between Sobieski's force and the Turks and Tatars?	Latent-RAT:
	1.select(sub=Sobieski)
	2.project(projection=force of #REF;sub=#1)
	3.select(sub=Turks)
	4.select(sub=the Tatars)
	5.arithmetic[difference](left=#2;right=#3)
	6.arithmetic[difference](left=#4;right=#5)
		1.select(sub=Sobieski)
		2.project(projection=force of #REF;sub=#1)
		3.project(projection=size of #REF; sub=#2)
		4.select(sub=the Turks and Tatars)
		5.project(projection=the force of #REF; sub=#4)
		6.project(projection=size of #REF;sub=#5)
		5.arithmetic[difference](left=#6; right=#3)

How many year after Knopf was founded was it officially incorporated?	BiaffineGP:
	1.project(projection=Knopf was founded years; sub=#1)
	2.select(sub=was it officially incorporated)
	3.project(projection=years;sub=#2);
	4.arithmetic[difference](left=#3;right=#1)
		1.select(sub=Knopf was founded)
		2.select(sub=Knopf was officially incorporated)
		3.project(projection=year of #REF; sub=#1)
		4.project(projection=year of #REF; sub=#2)
		5.arithmetic[difference](left=#4; right=#3)

Table 3: Two cases from the dev set. The outputs are converted to Logical Form for comparison with BiaffineGP. One can refer to (Hasson and Berant, 2021) for the conversion details.

Length	1-2	3-4	5-6	7-8	9+
Percentage(%)	10.7	44.9	27.0	10.1	7.4

Table 4: The distribution over the length of QDMR actions.

	Token	Action
Train	11.59	4.75
Dev	11.35	4.90

Table 5: The average length of token sequences and action sequences in QDMR.

3.2.2 Length Analysis

In order to explore how much does our transition-based framework contribute to examples with longer steps, we plot and compare the average LF-EM of different methods for each possible number of steps in QDMR. From Figure 3 we can see that, as the number of steps increases, our method exceeds the baselines greater. It verifies that our method handles complex decompositions better.

Table 5 shows the average length of token sequences used in seq2seq models and of action sequences used in our method. As shown in the table, the action sequence is much shorter than the token sequence i.e. reduced from 11 to 5 in length. In other words, representing QDMR as an action sequence has the advantage of more compact encoding which makes the modeling of long-distance dependency easier. Therefore, it is more appropriate for QDMR generation.

3.2.3 Interpretability Analysis

In order to verify the interpretability of our method, we print the beam search process together with the log probabilities of different action sequences. From Figure 4 we can see that, the sequence with the highest log probability (-2.44) matches the gold decomposition. We note that although the last three sequences do not match with the gold decomposition, they are logically equivalent to the provided gold one which can also get the correct answer to the question.

3.2.4 Case Study

We show two examples from the development set to illustrate the effectiveness of our model by comparing the results of different models in Table 3. The first example shows that beam search helps for searching the optimal graph. Latent-RAT fails to predict the correct structure and starts to deviate from step 3. In contrast, our method seeks the optimal graph in a larger search space and predicts the correct structure. The second example shows that our modeling of the question and the current state helps the model decide a step more accurately. In the example, BiaffineGP predicts the whole graph in an end-to-end manner. Our method predicts more accurately with the help of the better representation.

4 Related Work

Complex Question Understanding Complex question understanding is proposed by (Wolfson et al., 2020) as a standalone language understanding task. They introduce a formalism named QDMR to represent the meaning of questions that relies on question decomposition and is agnostic to

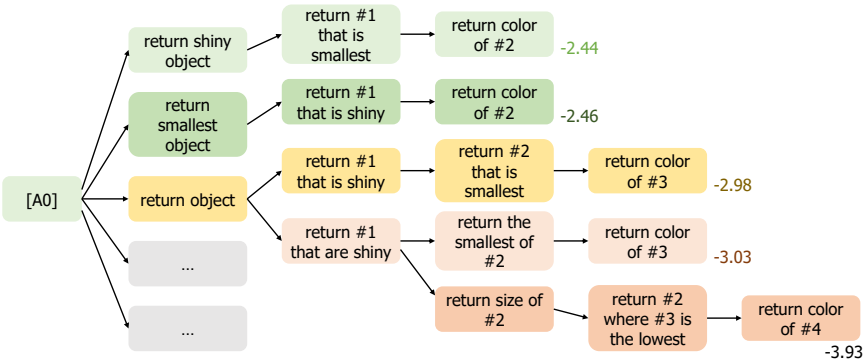

Figure 4: The beam search process of our method corresponds to the question "What color is the smallest, shiny object?". For simplicity, we do not draw the unselected nodes. The log probability of the final graph is listed at the end of the sequence.

the information modality. Existing approaches can be divided into two categories: the seq2seq-based method and the dependency-based method. The seq2seq-based method (Hasson and Berant, 2021) treats QDMR as a sentence and adopts a seq2seq model to decode the textual QDMR one token at each step. This token-wise modeling seems sub-optimal for generating QDMR which ignores its inherent structure. The dependency-based method (Hasson and Berant, 2021) maps QDMR to a dependency graph over question tokens and adopts a non-autoregressive graph parser to predict the entire graph in a single step. It predicts all steps of QDMR simultaneously which does not model the interaction between different predictions, trading off performance for efficiency. There is also work combining the two categories by exploiting the graph supervision to train a seq2seq model. It adds an auxiliary loss term where the graph is decoded from the encoder representations. However, the above methods do not explicitly model the step-wise information which is an essential and distinct characteristic of QDMR. Before CQU was proposed, some work has explored decomposing questions to facilitate answering complex questions that require discrete reasoning (Talmor and Berant, 2018). IBM Watson (Ferrucci et al., 2010) decomposes questions into sub-questions in multiple ways or not at all. DECOMPRC (Min et al., 2019) recasts sub-question generation as a span prediction problem which requires only 400 decomposition examples to train a competitive model. (Iyyer et al., 2017; Talmor and Berant, 2018) have also decomposed questions to create a sequential question answering task. Despite the initial success, their decomposition methods remain preliminary and they conduct experiments on a much more limited set of questions than in BREAK.

Semantic Parsing Semantic parsing is a larger area of work that aims at parsing natural language utterances into logical forms(Zelle and Mooney, 1996; Zettlemoyer and Collins, 2012; Liang et al., 2013). They are usually executed over structured knowledge bases such as relational databases(Yu et al., 2018) and graph KBs(Yih et al., 2016). Our work is inspired by the idea of transition-based systems from semantic parsing(Chen et al., 2018). CQU differs from semantic parsing in that it produces meaning representation expressed in natural language which is easy to annotate at scale and can be potentially converted to other meaning representations based on the task at hand. Besides, CQU focuses on representing the semantics of complex questions which is important for QA systems and for probing models for reasoning.

5 Conclusion

In this paper, to model the intermediate states and the step/operator-wise semantic, we view QDMR as a computational graph and propose a transition-based method where a *decider* wrapped with a *searcher* incrementally constructs the graph. Experimental results show that our framework outperforms the state-of-the-art CQU model by 3.1 NormEM and 2.0 LF-EM. Further visualization also demonstrates the interpretability of our method by giving transparent and human-readable intermediate results.

Acknowledgements

This work is partially supported by Na- tional Key Research and Development Project (2019YFB1704002) and National Natural Science Foundation of China (61876009). This work is also supported by Baidu-PKU Joint Project.

References

Bo Chen, Le Sun, and Xianpei Han. 2018. Sequence-to-action: End-to-end semantic graph generation for semantic parsing. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 766–777, Melbourne, Australia. Association for Computational Linguistics.

Timothy Dozat and Christopher D Manning. 2018. Simpler but more accurate semantic dependency parsing. *arXiv preprint arXiv:1807.01396*.

David Ferrucci, Eric Brown, Jennifer Chu-Carroll, James Fan, David Gondek, Aditya A Kalyanpur, Adam Lally, J William Murdock, Eric Nyberg, John Prager, et al. 2010. Building watson: An overview of the deepqa project. *AI magazine*, 31(3):59–79.

Matan Hasson and Jonathan Berant. 2021. Question decomposition with dependency graphs. *arXiv preprint arXiv:2104.08647*.

Mohit Iyyer, Wen-tau Yih, and Ming-Wei Chang. 2017. Search-based neural structured learning for sequential question answering. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1821–1831.

Diederik P Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*.

Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Ves Stoyanov, and Luke Zettlemoyer. 2019. Bart: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. *arXiv preprint arXiv:1910.13461*.

Percy Liang, Michael I Jordan, and Dan Klein. 2013. Learning dependency-based compositional semantics. *Computational Linguistics*, 39(2):389–446.

Sewon Min, Victor Zhong, Luke Zettlemoyer, and Hannaneh Hajishirzi. 2019. Multi-hop reading comprehension through question decomposition and rescoring. *arXiv preprint arXiv:1906.02916*.

Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, et al. 2019. Pytorch: An imperative style, high-performance deep learning library. *Advances in neural information processing systems*, 32.

Peter Shaw, Jakob Uszkoreit, and Ashish Vaswani. 2018. Self-attention with relative position representations. *arXiv preprint arXiv:1803.02155*.

Sanjay Subramanian, Ben Bogin, Nitish Gupta, Tomer Wolfson, Sameer Singh, Jonathan Berant, and Matt Gardner. 2020. Obtaining faithful interpretations from compositional neural networks. *arXiv preprint arXiv:2005.00724*.

Alon Talmor and Jonathan Berant. 2018. The web as a knowledge-base for answering complex questions. *arXiv preprint arXiv:1803.06643*.

Alon Talmor, Ori Yoran, Amnon Catav, Dan Lahav, Yizhong Wang, Akari Asai, Gabriel Ilharco, Hannaneh Hajishirzi, and Jonathan Berant. 2021. Multimodalqa: Complex question answering over text, tables and images. *arXiv preprint arXiv:2104.06039*.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. *Advances in neural information processing systems*, 30.

Bailin Wang, Richard Shin, Xiaodong Liu, Oleksandr Polozov, and Matthew Richardson. 2019. Rat-sql: Relation-aware schema encoding and linking for text-to-sql parsers. *arXiv preprint arXiv:1911.04942*.

Tomer Wolfson, Mor Geva, Ankit Gupta, Matt Gardner, Yoav Goldberg, Daniel Deutch, and Jonathan Berant. 2020. Break it down: A question understanding benchmark. *Transactions of the Association for Computational Linguistics*, 8:183–198.

Wen-tau Yih, Matthew Richardson, Christopher Meek, Ming-Wei Chang, and Jina Suh. 2016. The value of semantic parse labeling for knowledge base question answering. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 201–206.

Tao Yu, Rui Zhang, Kai Yang, Michihiro Yasunaga, Dongxu Wang, Zifan Li, James Ma, Irene Li, Qingning Yao, Shanelle Roman, et al. 2018. Spider: A large-scale human-labeled dataset for complex and cross-domain semantic parsing and text-to-sql task. *arXiv preprint arXiv:1809.08887*.

John M Zelle and Raymond J Mooney. 1996. Learning to parse database queries using inductive logic programming. In *Proceedings of the national conference on artificial intelligence*, pages 1050–1055.

Luke S Zettlemoyer and Michael Collins. 2012. Learning to map sentences to logical form: Structured classification with probabilistic categorial grammars. *arXiv preprint arXiv:1207.1420*.

Semantic Role Labeling as Dependency Parsing: Exploring Latent Tree Structures Inside Arguments

Yu Zhang[☆], **Qingrong Xia**[☆☾], **Shilin Zhou**[☆], **Yong Jiang**[☼], **Guohong Fu**[☆*], **Min Zhang**[☆]
[☆]Institute of Artificial Intelligence, School of Computer Science and Technology,
Soochow University, Suzhou, China
[☾]Huawei Cloud, China
[☼]DAMO Academy, Alibaba Group, China
{yzhang.cs,slzhou.cs}@outlook.com; xiaqingrong@huawei.com
yongjiang.jy@alibaba-inc.com; {ghfu,minzhang}@suda.edu.cn

Abstract

Semantic role labeling (SRL) is a fundamental yet challenging task in the NLP community. Recent works of SRL mainly fall into two lines: 1) BIO-based; 2) span-based. Despite ubiquity, they share some intrinsic drawbacks of not considering internal argument structures, potentially hindering the model's expressiveness. The key challenge is arguments are flat structures, and there are no determined subtree realizations for words inside arguments. To remedy this, in this paper, we propose to regard flat argument spans as latent subtrees, accordingly reducing SRL to a tree parsing task. In particular, we equip our formulation with a novel span-constrained TreeCRF to make tree structures span-aware and further extend it to the second-order case. We conduct extensive experiments on CoNLL05 and CoNLL12 benchmarks. Results reveal that our methods perform favorably better than all previous syntax-agnostic works, achieving new state-of-the-art under both *end-to-end* and *w/ gold predicates* settings.

1 Introduction

Semantic role labeling (SRL) is a fundamental yet challenging task in the NLP community, involving predicate and argument identification, as well as semantic role classification. As SRL can provide informative linguistic representations, it has been widely adopted in downstream tasks like question answering (Berant et al., 2013; Yih et al., 2016), information extraction (Christensen et al., 2010; Lin et al., 2017), and machine translation (Liu and Gildea, 2010; Bazrafshan and Gildea, 2013), etc.

Recent works of SRL mainly fall into two lines: 1) BIO-based; 2) span-based. The former views SRL as a sequence labeling task (Zhou and Xu, 2015; Strubell et al., 2018; Shi and Lin, 2019). For each predicate, each token is tagged with a label starting with BIO prefixes indicating if it is at the **B**eginning, **I**nside, or **O**utside of an argument.

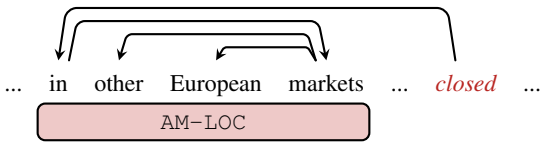

Figure 1: An argument example (below) and its related subtree structure (above) for the predicate "*closed*".

The latter (He et al., 2018a; Ouchi et al., 2018; Li et al., 2019), in contrast, opts to jointly predict all predicate and argument span pairs using a span-graph formulation.

Despite ubiquity, there are some drawbacks that limit the expressiveness of the two methods. First, framing predicate-argument structures as a BIO-tagging scheme is less effective as it lacks explicit modeling of span-level representations, so that long adjacencies of argument phrases can be ignored (Cohn and Blunsom, 2005; Jie and Lu, 2019; Zhou et al., 2020d; Xu et al., 2021). Second, span-based method seeks to pick very few (typically $<10\%$) positive examples from $O(n^3)$ candidate predicate-argument pairs, thus suffering from severe class imbalance problem (Li et al., 2021). To alleviate this issue, span-based method relies on heavy pruning (He et al., 2018a) to reduce the searching space, potentially impairing the performance.

Meanwhile, both formulations share some common flaws in terms of lacking explicit modeling of internal argument structures, which appear to be beneficial to SRL. Taking Fig. 1 as an example, internal dependencies of words ("in other European markets") inside the span provide strong clues for recognizing it as a locative modifier ("AM-LOC") of the predicate "*closed*". Besides, the predicate-argument relation can be naturally reflected by the dependency from the predicate to the span head-word ("*closed* $\xrightarrow{\text{AM-LOC}}$ in"), and we can properly recognize the argument span boundaries by retrieving all descendants of the subtree. Such observa-

*Corresponding author

Proceedings of the 29th International Conference on Computational Linguistics, pages 4212–4227
October 12–17, 2022.

tions have motivated many attempts on utilizing relations inside arguments (Gildea and Hockenmaier, 2003; Johansson and Nugues, 2008a,c; Xia et al., 2019; Li et al., 2019, *inter alia*). However, stuck on the fact that span-style SRL has no determined internal structure realizations, existing works have to resort to making use of external human-annotated syntax knowledge to bridge the gap (Shi et al., 2020; Li et al., 2021).

Our main goal in this work is to explicitly take internal argument structures into account meanwhile keeping our framework *end-to-end*. To this end, we propose to model flat arguments as latent subtrees, thus paving the way for reducing SRL to dependency parsing seamlessly: *we view predicate-argument structures as partially-observed trees where exact subtrees for each argument are not realized yet.* In this way, we reframe span-style SRL as parsing word-to-word relations by encoding all predicate-argument relations into a unified dependency graph. Unlike span-based methods (He et al., 2018a), a dependency graph contains no more than $O(n^2)$ possible dependencies, so that the class imbalance issue can be side-stepped effortlessly. Specifically, we make use of TreeCRF (Eisner, 2000; Zhang et al., 2020), which provides a viable way for probabilistic modeling of tree structures, to learn the partially-observed trees and marginalize the latent structures out during training. Unlike canonical TreeCRF, which enumerates all possible trees, in our setting, we have to impose many span constraints to reflect the argument boundaries on subtrees correctly. To accommodate this, we further design a novel span-constrained TreeCRF to adapt it to our learning procedure, which explicitly prohibits invalid edges across different arguments as well as multi-head subtrees (Nivre et al., 2014; Zhang et al., 2021a).

There are further advantages to our reduction. Conversion to tree structures enables us to easily conduct global optimization (Eisner, 1996; McDonald et al., 2005) in polynomial time, which has already been shown to often lead to improved results and more meaningful predictions (Toutanova et al., 2008; Täckström et al., 2015; FitzGerald et al., 2015; Li et al., 2020) compared to local unconstrained methods. On the other hand, by drawing on the experience in the parsing literature, we can further extend our method to some well-studied high-order methods (McDonald and Pereira, 2006) without any obstacle. We experiment with sibling factors in this work and find significant gains, in line with many parsing works (Zhang et al., 2020; Fonseca and Martins, 2020). Our contributions can be summarized as follows:[1]

- Aware of the benefits of internal argument structures, we propose to model flat argument spans as latent subtrees, thereby reducing SRL to dependency parsing seamlessly.
- We propose a novel span-constrained TreeCRF to learn the converted trees and further extend it to the second-order case.
- Experiments on CoNLL05 and CoNLL12 benchmarks reveal that our proposed methods outperform existing works significantly, achieving new state-of-the-art results under the syntax-agnostic setting.

2 Overview

In span-style SRL, an argument of a predicate corresponds to one word or multiple continuous words. In the latter case, each word in the argument span is treated as equal, and the internal structure of a multi-word argument, i.e., the relationship between words inside the argument, is usually overlooked due to the lack of corresponding annotations.

In this work, we propose to explicitly model internal structures of multi-word arguments and treat arguments as latent subtrees. Our approach deals with each predicate separately, and assumes each corresponds to a single-root tree. Consequently, each argument subtree is attached to the predicate. During the training process, all possible structures are enumerated and accumulated to compose the argument representation. While decoding, we seek to find a 1-best tree and recover arguments from the subtrees belonging to the resulting structure. We highlight four key points.

i Our approach is syntax-agnostic. The tree structures are modeled and predicted solely to serve the SRL task without referring to any linguistic syntax knowledge.

ii The predicate identification subtask is handled as a simple classification procedure.

iii For argument identification, argument boundaries are decided by subtrees attached to the predicate, and edge labels are used for role disambiguation.

iv We adopt a consistent scoring architecture for the two subtasks and train them jointly.

[1]Our code is publicly available at https://github.com/yzhangcs/crfsrl.

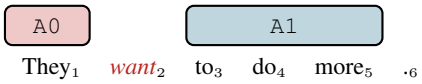

(a) Original structure: arguments of the predicate are located in the upper half-plane of the sentence and do not overlap with each other.

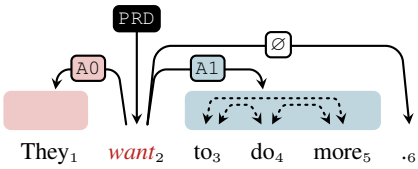

(b) Training: convert the predicate-argument structure to a dependency tree with (dotted) latent annotations; non-argument spans are assigned "∅" for distinction.

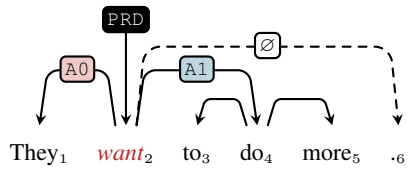

(c) Decoding: realize a tree rooted at the predicate with the arc labeled as "PRD"; (dashed) arcs labeled as "∅" are discarded.

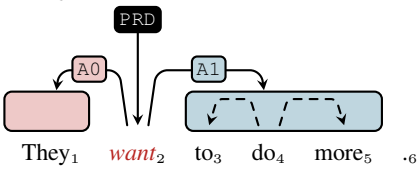

(d) Recovery: collapse all (dashed) subtrees governed by the predicate into flat argument spans.

Figure 2: Illustration of our SRL→Tree conversion (Fig. 2a and Fig. 2b), and its inverse Tree→SRL process (Fig. 2c and Fig. 2d). We emphasize the predicate "*want*" in the figures for clarity. The two arguments with roles "A0" and "A1" are framed by red and blue rectangles, respectively.

2.1 SRL → Tree Conversion

Formally, given an input sentence $\boldsymbol{x} = x_1, \ldots, x_n$, we first seek to obtain tree structures for each predicate $p \in \boldsymbol{x}$, which are taken as materials of training a parser. We define a directed acyclic dependency tree $\boldsymbol{t}$ by assigning a head $h \in \{x_0, x_1, \ldots, x_n\}$ together with a relation label $r \in \mathcal{R}$ to each modifier $m \in \boldsymbol{x}$, where a dummy word x_0 is attached before $\boldsymbol{x}$ as the pseudo root node.[2]

For predicate p, the first step is to link x_0 to p. To facilitate predicate identification, we assign a special label PRD (resp. ∅ for non-predicate) to the dependency $x_0 \rightarrow p$. Then, we make all corresponding latent argument subtrees descendants of p. As we showcase in Fig. 2a, this takes advantage of the non-overlapping constraint for arguments belonging to the same predicate (Punyakanok et al., 2004; Li et al., 2019). For an argument with a consecutive word span $x_i, \ldots, x_j$ and a semantic role $r \in \mathcal{R}$, we restrict all possible subtrees are single-rooted at a potential headword h within the span, which is also not realized yet. The semantic role r is assigned as the label of the dependency pointing from p to the headword. We adopt a similar strategy for non-argument spans, except that we set the label to ∅ for distinction and remove the single-root restriction.

By enumerating all possible subtrees and combing them together, the resulting tree set T_p is exponential in size. During training, we develop a span-constrained Inside algorithm to perform the enumeration (§ 3.2). Fig. 2b gives a brief example of the conversion process.

2.2 Tree → SRL Recovery

Supposing we have trained a parsing model, during the decoding phase, what we need is to recover predicate-argument structures from the outputs of the parser.

We first find all predicates via simple local label classification: a word p is recognized as a predicate if the dependency $x_0 \rightarrow p$ is labeled as PRD. Subsequently, we obtain the highest-scoring tree $\boldsymbol{t}^*$ (Fig. 2c) for p using Eisner algorithm (Eisner, 2000) with complexity $O(n^3)$:

$$\boldsymbol{t}^* = \arg \max_{\boldsymbol{t}: x_0 \xrightarrow{\text{PRD}} p \in \boldsymbol{t}} \text{s}(\boldsymbol{x}, \boldsymbol{t}) \quad (1)$$

where $\text{s}(\boldsymbol{x}, \boldsymbol{t})$ is the tree score, and the tree is restricted to be rooted at p. Arguments for the predicate are then recovered by collapsing subtrees headed by p into flat spans.

Concretely, we take each modifier h of p as the headword of a potential argument. If the label r of $p \rightarrow h$ is not "∅", i.e., non-argument, then an entire argument span comprises h and its descendants and takes r as the semantic role. The resulting SRL output is the collection of all predicates and corresponding recovered arguments. A recovery example is demonstrated in Fig. 2d.

[2] In this work, we assume all dependency trees are *projective*, i.e., without any crossing arcs. This property allows us to associate the subtree with its continuous argument span (Kong et al., 2015).

3 Methodology

Now we elaborate the architecture of our proposed model for training the parser. Following Dozat and Manning (2017); Zhang et al. (2020), our model consists of a contextualized encoder and a (second-order) scoring module. We further propose a span-aware TreeCRF to compute the probabilities of the converted partially-observed trees.

3.1 Neural Parameterization

Given the sentence $\boldsymbol{x} = x_0, x_1, \ldots, x_n$, we first obtain the hidden representation of each token x_i via a deep contextualized encoder.

$$\mathbf{h}_0, \mathbf{h}_1, \ldots, \mathbf{h}_n = \texttt{Encoder}(x_0, x_1, \ldots, x_n) \quad (2)$$

In this work, we experiment with two alternative encoders, i.e., BiLSTMs (Gal and Ghahramani, 2016) and pretrained language models (PLMs) (Devlin et al., 2019). More setting details are available in § A.

(Second-order) Tree parameterization Following Dozat and Manning (2017), we decompose a tree $\boldsymbol{t}$ into two separate $\boldsymbol{y}$ and $\boldsymbol{r}$, where $\boldsymbol{y}$ is a skeletal tree, and $\boldsymbol{r}$ is the related strictly-ordered label sequence. For each head-modifier pair $h \rightarrow m \in \boldsymbol{y}$, we score them using two MLPs followed by a Biaffine layer (Cai et al., 2018):

$$\begin{aligned} \mathbf{r}_i^{\text{head/mod}} &= \texttt{MLP}^{\text{head/mod}}(\mathbf{h}_i) \\ \text{s}(h \rightarrow m) &= \texttt{BiAF}\left(\mathbf{r}_h^{\text{head}}, \mathbf{r}_m^{\text{mod}}\right) \end{aligned} \quad (3)$$

The score of the dependency $h \rightarrow m$ with label $r \in \mathcal{R}$ is calculated analogously. We use two extra MLPs and $|\mathcal{R}|$ Biaffine layers to obtain all label scores.

We also make use of adjacent-sibling information (McDonald and Pereira, 2006) to enhance the first-order biaffine parser further. Following Wang et al. (2019); Zhang et al. (2020), we employ three extra MLPs as well as a Triaffine layer for second-order subtree scoring,

$$\begin{aligned} \mathbf{r}_i^{\text{head/mod/sib}} &= \texttt{MLP}^{\text{head/mod/sib}}(\mathbf{h}_i) \\ \text{s}(h \rightarrow s, m) &= \texttt{TriAF}\left(\mathbf{r}_h^{\text{head}}, \mathbf{r}_m^{\text{mod}}, \mathbf{r}_s^{\text{sib}}\right) \end{aligned} \quad (4)$$

where s and m are two adjacent modifiers of h, and s populates between h and m.

Under the first-order factorization (McDonald et al., 2005), the score of $\boldsymbol{y}$ becomes

$$\text{s}(\boldsymbol{x}, \boldsymbol{y}) = \sum_{h \rightarrow m \in \boldsymbol{y}} \text{s}(h \rightarrow m) \quad (5)$$

For the second-order case (McDonald and Pereira, 2006), we further incorporate adjacent-sibling subtree scores into tree scoring:

$$\text{s}(\boldsymbol{x}, \boldsymbol{y}) = \sum_{h \rightarrow m} \text{s}(h \rightarrow m) + \sum_{h \rightarrow s, m} \text{s}(h \rightarrow s, m) \quad (6)$$

The probabilities of skeletal tree $\boldsymbol{y}$ and its label sequence $\boldsymbol{r}$ are parameterized as

$$\begin{aligned} P(\boldsymbol{y} \mid \boldsymbol{x}) &= \frac{\exp\left(\text{s}(\boldsymbol{x}, \boldsymbol{y})\right)}{Z(\boldsymbol{x}) \equiv \sum_{\boldsymbol{y}' \in Y(\boldsymbol{x})} \exp\left(\text{s}(\boldsymbol{x}, \boldsymbol{y}')\right)} \\ P(\boldsymbol{r} \mid \boldsymbol{x}, \boldsymbol{y}) &= \prod_{h \xrightarrow{r} m \in \boldsymbol{t}} P(r \mid \boldsymbol{x}, h \rightarrow m) \end{aligned} \quad (7)$$

$Y(\boldsymbol{x})$ is the set of all possible legal unlabeled trees, and $Z(\boldsymbol{x})$ is known as the partition function. Each label r is independent of tree $\boldsymbol{y}$ and other labels, thus $P(r \mid \boldsymbol{x}, h \rightarrow m)$ is locally normalized over all $r' \in \mathcal{R}$.

Finally, we define the probability of the labeled tree $\boldsymbol{t}$ as the product of the probabilities of its two sub-components.

$$P(\boldsymbol{t} \mid \boldsymbol{x}) = P(\boldsymbol{y} \mid \boldsymbol{x}) \cdot P(\boldsymbol{r} \mid \boldsymbol{x}, \boldsymbol{y}) \quad (8)$$

3.2 Span-constrained TreeCRF

Training objective During training, we seek to maximize the probability of converted trees T_p for each predicate p. Accordingly, we define the following loss function:

$$\mathcal{L} = -\sum_p \log P(T_p \mid \boldsymbol{x}) \quad (9)$$

in which $P(T_p \mid \boldsymbol{x})$ can be further expanded as

$$\begin{aligned} P(T_p \mid \boldsymbol{x}) &= \sum_{\boldsymbol{t} \in T_p} \underbrace{P(\boldsymbol{y} \mid \boldsymbol{x}) \cdot P(\boldsymbol{r} \mid \boldsymbol{x}, \boldsymbol{y})}_{P(\boldsymbol{t} \mid \boldsymbol{x})} \\ &= \frac{1}{Z(\boldsymbol{x})} \sum_{\boldsymbol{t} \in T_p} \underbrace{\exp(\text{s}(\boldsymbol{x}, \boldsymbol{y})) \cdot P(\boldsymbol{r} \mid \boldsymbol{x}, \boldsymbol{y})}_{\exp(\text{s}(\boldsymbol{x}, \boldsymbol{t}))} \end{aligned} \quad (10)$$

The proposed Inside The calculation of the denominator $Z(\boldsymbol{x})$ in Eq. 10 can be accomplished by the canonical Inside algorithm. As for the numerator, we make a slight change of the formula and define the labeled tree score as:

$$\text{s}(\boldsymbol{x}, \boldsymbol{t}) = \text{s}(\boldsymbol{x}, \boldsymbol{y}) + \log P(\boldsymbol{r} \mid \boldsymbol{x}, \boldsymbol{y}) \quad (11)$$

In this way, the numerator is exactly the summation of exponential scores of all legal labeled trees.[3]

[3] It is noteworthy that we do not assign any label to $h \rightarrow m \in \boldsymbol{y}$ while $h \notin \{x_0, p\}$, i.e., any dependency inside an argument span, thus its logarithmic label probability is set to 0 and does not contribute to tree scoring.

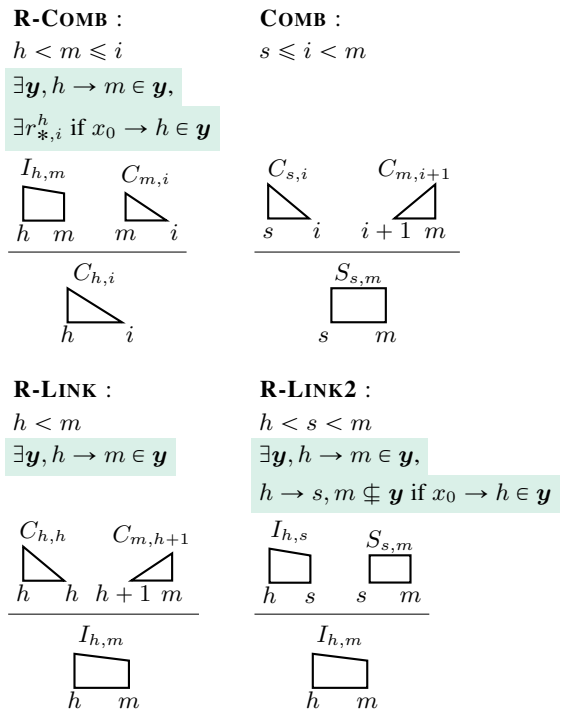

Figure 3: Deduction rules for our span-constrained Inside algorithm (**R-COMB** and **R-LINK**) and its second-order extension (**COMB** and **R-LINK2**). Our modified rule constraints are highlighted in *green color*. The condition $x_0 \rightarrow h \in \boldsymbol{y}$ means h is a predicate with x_0 as the parent. $r^{h}_{*,i}$ denotes an argument span that takes h as the predicate and ends with i. We show only R-rules, omitting the symmetric L-rules as well as initial conditions for brevity.

This differs from the traditional case of partial tree learning (Li et al., 2016) from two perspectives where the common Inside algorithm is not adequate to: 1) we impose span constraints to force the converted latent subtrees to reflect argument spans, and 2) we require the subtree ought to be single-rooted at one potential headword in the span.

To resolve this, in this work, we propose a span-constrained Inside algorithm to accommodate these constraints. We illustrate the deduction rules (Pereira and Warren, 1983) of our tailored algorithm and its second-order extension in Fig. 3.[4] Basically, we avoid the arc $h \rightarrow m$ crossing different argument spans by prohibiting merging to the relevant incomplete span $I_{h,m}$ (**R-LINK**). To prevent multiple headwords in the same argument, inspired by Zhang et al. (2021a), for predicate h, we only allow merging to the complete span $C_{h,i}$ if i is at the endpoint of an argument (**R-COMB**).

[4]We refer interested readers to § B for more details on the exact meaning of the operations in the Inside algorithm.

	#Train	#Dev	#Test	#OOD	#roles
CoNLL05	39,832	1,346	2,416	2,399	54
CoNLL12	75,187	9,603	9,479	-	63

Table 1: Data statistics for CoNLL05 and CoNLL12 datasets.

For the second-order case, we further prohibit the subtree $h \rightarrow s, m$ once s and m are located in the same argument (**R-LINK2**), since this case implies that the argument can be split into two more smaller headed spans with respect to s and m, which is not what we expect.

Time complexity analysis The proposed span-constrained Inside shares the same asymptotic time complexity of $O(n^3)$ as its canonical counterpart (Eisner, 2000). Besides, we draw on the recent development of parallelization techniques (Eisner, 2016; Zhang et al., 2020; Rush, 2020) and further reduce the complexity of the parallelized algorithm to $O(n^2)$ on GPUs. In practice, we find that our models are efficient enough compared to BIO-based and Span-based models. We make comprehensive speed comparisons in § 4.3.

4 Experiments

We measure our proposed first-order CRF and second-order CRF2O models on two SRL benchmarks: CoNLL05 and CoNLL12. Full implementation details are given in § A.

Data Table 1 lists the statistics of the datasets. For CoNLL05, we follow standard splits of Carreras and Màrquez (2005): sections 02-21 of WSJ corpus as Train data, section 24/23 as Dev/Test data, and three sections (CK01-03) of the Brown corpus as out-of-domain (OOD) data. For CoNLL12, following He et al. (2018a), we extract data from OntoNotes (Pradhan et al., 2013) and follow the data splits of the CoNLL12 shared task (Pradhan et al., 2012).[5] We adopt the same splits for both *end-to-end* and *w/ gold predicates* settings. We use the official scripts provided by CoNLL05 shared task[6] for evaluation.

4.1 Main results

Table 2 gives our main results. By default, our models work in an *end-to-end* fashion, i.e., predicting

[5]The list of file IDs for Train/Dev/Test data is available on the task webpage.

[6]https://www.cs.upc.edu/~srlconll

	CoNLL05							CoNLL12			
	Dev	WSJ			Brown			Dev	Test		
	F_1	P	R	F_1	P	R	F_1	F_1	P	R	F_1
He et al. (2017)	80.3	80.2	82.3	81.2	67.6	69.6	68.5	75.5	78.6	75.1	76.8
He et al. (2018a)	81.6	81.2	83.9	82.5	69.7	71.9	70.8	79.4	79.4	80.1	79.8
Li et al. (2019)	-	-	-	83.0	-	-	-	-	-	-	-
Zhou et al. (2020a)	82.27	-	-	-	-	-	-	-	-	-	-
CRF	83.70	83.18	85.38	84.27	70.40	72.97	71.66	81.03	**79.47**	82.80	81.10
CRF2O	**83.91**	**83.26**	**86.20**	**84.71**	**70.70**	**74.16**	**72.39**	**81.16**	79.27	**83.24**	**81.21**
Li et al. (2019)$_{\text{ELMo}}$	-	85.2	87.5	86.3	74.7	78.1	76.4	-	**84.9**	81.4	83.1
Zhou et al. (2022)$_{\text{BERT}}$	86.79	**87.15**	88.44	87.79	**79.44**	80.85	80.14	84.74	83.91	85.61	84.75
CRF$_{\text{BERT}}$	86.82	86.98	88.28	87.63	79.19	80.92	80.05	85.35	84.47	86.24	85.35
CRF2O$_{\text{BERT}}$	**87.03**	87.00	**88.76**	**87.87**	79.08	**81.50**	**80.27**	**85.53**	84.53	**86.41**	**85.45**
CRF$_{\text{RoBERTa}}$	87.31	87.20	88.67	87.93	79.29	81.48	80.38	86.08	84.98	86.86	85.91
CRF2O$_{\text{RoBERTa}}$	**87.46**	**87.35**	**89.34**	**88.33**	**79.95**	**82.32**	**81.12**	**86.34**	**85.30**	**87.02**	**86.15**
w/ gold predicates											
He et al. (2017)	81.6	83.1	83.0	83.1	72.9	71.4	72.1	81.5	81.7	81.6	81.7
Ouchi et al. (2018)	82.5	84.7	82.3	83.5	**76.0**	70.4	73.1	82.9	**84.4**	81.7	83.0
Tan et al. (2018)	83.1	84.5	85.2	84.8	73.5	74.6	74.1	82.9	81.9	83.6	82.7
Strubell et al. (2018)	-	84.7	84.24	84.47	73.89	72.39	73.13	-	-	-	-
Zhou et al. (2020a)	83.16	-	-	-	-	-	-	-	-	-	-
Zhang et al. (2021b)	84.45	85.30	85.17	85.23	74.98	73.85	74.41	82.83	83.09	83.71	83.40
CRF	84.42	85.38	85.56	85.47	75.05	74.05	74.55	83.22	83.21	83.85	83.53
CRF2O	**84.65**	**85.47**	**86.40**	**85.93**	74.92	**75.00**	**74.96**	**83.39**	83.02	**84.31**	**83.66**
Strubell et al. (2018)$_{\text{ELMo}}$	85.26	86.21	85.98	86.09	77.1	75.61	76.35	83.23	84.39	82.21	83.28
Shi and Lin (2019)$_{\text{BERT}}$	-	88.6	89.0	88.8	81.9	82.1	82.0	-	85.9	87.0	86.5
Jindal et al. (2020)$_{\text{BERT}}$	-	88.7	88.0	87.9	80.3	80.1	80.2	-	86.3	86.8	86.6
Zhang et al. (2021b)$_{\text{BERT}}$	87.38	87.70	88.15	87.93	81.52	81.36	81.44	86.27	86.00	86.84	86.42
Zhou et al. (2022)$_{\text{BERT}}$	87.54	**89.03**	88.53	88.78	**83.22**	81.81	82.51	86.97	87.26	87.05	87.15
Conia and Navigli (2020)$_{\text{BERT}}$	-	-	-	-	-	-	-	-	86.9	87.7	87.3
Blloshmi et al. (2021)$_{\text{BART}}$	-	-	-	-	-	-	-	-	**87.8**	86.8	87.3
CRF$_{\text{BERT}}$	87.76	88.93	88.58	88.76	82.87	81.67	82.27	87.33	87.45	87.56	87.51
CRF2O$_{\text{BERT}}$	**88.05**	89.00	**89.03**	**89.02**	82.81	**82.35**	**82.58**	**87.52**	87.52	**87.79**	**87.66**
CRF$_{\text{RoBERTa}}$	88.21	89.29	88.99	89.15	83.22	82.42	82.82	87.97	87.99	88.22	88.11
CRF2O$_{\text{RoBERTa}}$	**88.49**	**89.45**	**89.63**	**89.54**	**83.89**	**83.39**	**83.64**	**88.29**	**88.11**	**88.53**	**88.32**

Table 2: Results on CoNLL05 and CoNLL12 data. All results are averaged over 4 runs with different random seeds.

all predicates and their associated arguments simultaneously. However, we note that reporting the results of using gold predicates is a more prevalent practice in the SRL community (He et al., 2018a; Shi and Lin, 2019). Therefore, for comprehensive comparisons, in addition to listing most *end-to-end* results of previous works we are aware of, we also conduct experiments with gold predicates, which is achieved by only parsing trees rooted at the pre-specified predicates.[7]

The two major rows show the results of *end-to-end* and *w/ gold predicates* settings, indicating very consistent trends. We can clearly see that under the *end-to-end* setting, our LSTM-based CRF models outperform previous works by a large margin on all datasets. The second-order CRF2O further improves over CRF by 0.2, 0.4 and 0.7 F_1 scores on three CoNLL05 datasets, respectively. On CoNLL12, CRF2O shows smaller but steady gains. As revealed in § 4.2, we attribute the improvements brought by CRF and CRF2O to better performing at global consistency and long-range dependencies.

The results under the *w/ gold predicates* setting are presented in the second major row. Many PLM-based results comparable to ours are available in this setting. Among them, the BIO-based parser of Shi and Lin (2019) achieves 88.8, 82.0 and 86.5 F_1 scores on CoNLL05 WSJ, Brown and CoNLL12 Test data. The dependency (word)-based parser of Zhou et al. (2022) achieves 88.78, 82.51 and 87.15 F_1 scores. Meanwhile, the results of our first-order CRF model with BERT is $88.76_{\pm 0.18}$, $82.27_{\pm 0.26}$ and $87.51_{\pm 0.11}$. The performance gap between CRF and recent state-of-the-art parsers are negligibly small. We note that we do not uti-

[7]We eliminate the invalid $x_0 \rightarrow p$ simply via setting the dependency score to $-\infty$.

	Dev				Test	
	P	R	F_1	CM	F_1	CM
BIO	86.80	86.38	86.59	69.24	88.22	71.95
SPAN	87.68	86.75	87.21	68.43	88.44	70.22
CRF	87.89	87.62	87.76	71.59	88.76	73.01
FIRST	87.44	86.60	87.02	70.35	87.81	71.14
LAST	86.99	87.00	86.99	70.29	87.67	71.08
FLAT	85.63	82.26	83.91	63.41	83.32	62.22
CRF2O	**88.02**	**88.09**	**88.05**	**72.57**	**89.02**	**73.74**

Table 3: Finetuning results on CoNLL05 Dev and Test data under the setting of *w/ gold predicates*.

lize any word/predicate embeddings as well as LSTM layers for simplicity, which may potentially hinder the results. Despite this fact, our second-order CRF2O achieves $89.02_{\pm 0.17}$, $82.58_{\pm 0.47}$, and $87.66_{\pm 0.05}$, which outperforms the systems of Shi and Lin (2019) by 0.2, 0.6 and 1.2 F_1 scores and achieves new state-of-the-art on both CoNLL05 and CoNLL12 datasets. This implies that imposing stronger structure constraints can still bring remarkable improvements for span-style SRL even when empowered with very expressive encoders. In the bottom lines, we provide the results of utilizing RoBERTa, we can see that CRF and CRF2O augmented with RoBERTa can obtain further gains on top of BERT.

We highlight that we do not include any syntax-aware work (Xia et al., 2019; Zhou et al., 2020a) in Table 2, which has shown to deliver substantial gains for SRL (see further discussions in § C). It is still an open question to be investigated whether the benefits brought by our methods are orthogonal to linguistic syntax knowledge. We focus on pure syntax-agnostic models in this paper. So we do not list the results of this line of works in order to make fair comparisons.

4.2 Analysis

To better understand which empowers our proposed CRF and CRF2O and in what aspects they are helpful, we conduct detailed analyses on CoNLL05 Dev data. Considering that there exist many differences in model/training settings, we re-implemented the following two methods based on two widely used libraries HanLP[8] (He and Choi, 2021) and SuPar[9] for fair comparisons:

- **BIO**: BIO-based method of Zhou and Xu (2015). Following Zhang et al. (2021b), we employ linear-chain CRF (Lafferty et al., 2001) to conduct global inference during training.
- **SPAN**: span-based method of He et al. (2018a). We borrow the settings of Strubell et al. (2018) and make use of Biaffine layers for span scoring.

We adopt the same experimental setups for all implementations, i.e., finetuning on BERT and assuming all predicates are given. Results are shown in Table 3. It is clear that under the same settings, our CRF expands the advantages over BIO and SPAN, and CRF2O further improves the performance.

Impact of latent subtrees First of all, we consider three variants of our first-order CRF to verify the necessity of modeling arguments as latent trees: 1) FIRST, similar to CRF but always takes the first word as argument headword; 2) LAST, denoting the last word accordingly; 3) FLAT, similar to FIRST but directly attach other argument words to the first word. The first two variants fix the position of argument headwords. In Table 3, we observe that FIRST and LAST perform quite similarly and steadily inferior to CRF. This agrees with Zhang et al. (2021b), highlighting the importance of headwords in recognizing arguments. In contrast to CRF, FLAT completely excludes latent representations during training and restricts the height of the converted trees to 2. We can see that FLAT achieves 83.91 F_1 on Dev, a dramatic performance drop against CRF (87.76). Overall, as we expect, it seems that totally latent argument representations empower CRF a lot, performing best compared to other variants.

Structural consistency To quantify the benefits of our methods in making global decisions for SRL structures, we report the percentage of completely correct predicates (CM) (He et al., 2018a) in Table 3. We show that BIO with linear-chain CRF significantly outperforms SPAN, but still falls short of our CRF by 1.5. By explicitly modeling sibling information, CRF2O provides stronger structure constraints and goes further beyond CRF by 0.9. In terms of the performance broken down by argument length, as shown in Fig. 4a, SPAN lags largely behind BIO over length$\geqslant$8. We guess this is mainly because of their aggressive argument pruning strategy. And as expected, CRF and CRF2O demonstrate steady improvements over BIO and SPAN. We owe this to the superiority of our formulations in modeling subtree structures, thus providing more powerful argument representations and rich inter- and intra-argument dependency interactions.

[8] https://github.com/hankcs/HanLP
[9] https://github.com/yzhangcs/parser

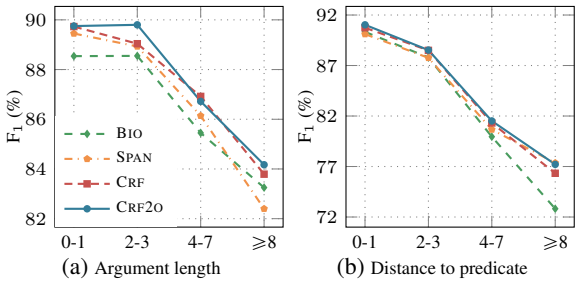

Figure 4: F_1 scores breakdown by argument length (Fig. 4a) and predicate-argument distance (Fig. 4b).

		Sents/s
Strubell et al. (2018)	BIO	50
He et al. (2018a)	SPAN	49
Li et al. (2019)	SPAN	20
Ours	CRF	**242**
	CRF2O	214
	CRF_{BERT}	136
	$\text{CRF2O}_{\text{BERT}}$	113

Table 4: Speed comparison on CoNLL05 Test data. We also list the speed of our TreeCRF models using BERT (CRF_{BERT} and $\text{CRF2O}_{\text{BERT}}$).

Long-range dependencies Fig. 4b shows the results broken down by predicate-argument distance. It is clear that the gaps between BIO and other methods become larger as the distance increases. This is reasonable since BIO lacks explicit connections for non-adjacent predicate-argument pairs, whereas ours provides direct word-to-word bilexical mappings. SPAN shows competitive results but is still inferior to ours. we speculate this is due to their inferiority in ultra-long arguments, as illustrated in Fig. 4a.

4.3 Efficiency

Table 4 compares different models in terms of parsing speed. We obtain the speed of previous works by rerunning their released code. For fair comparisons, all models are run on Intel Xeon E5-2650 v4 CPU and Nvidia GeForce GTX 1080 Ti GPU, and do not use any PLM layers.

We can clearly see that our CRF and CRF2O can parse about 242 and 214 sentences per second respectively, much faster than all previous works. In line with Strubell et al. (2018), our CRF and CRF2O consume 5,000 tokens (roughly 200 sentences) per mini-batch. However, Strubell et al. (2018) use up to 12 Transformer layers, much deeper than our 3-layer BiLSTM encoder. This explains their less efficiency from the side, as encoder layers might take up a major part of the running time, while the relative more efficient Viterbi decoding does not dominate the time-consuming. Moreover, our models are based on highly parallelized implementations (Zhang et al., 2020). We speculate that the model speed of Strubell et al. (2018) can be further improved with dedicated optimization. As for He et al. (2018a); Li et al. (2019), we adopt their default setting of 40 sentences per batch. They need to obtain the representations of all candidate argument spans, leading to high GPU memory usage. This limits us to enlarge the batch size further and significantly slows down the parsing speed. In the bottom lines of Table 4, we can see that our CRF and CRF2O enhanced with BERT achieve speeds of 136 and 113 sentences per second respectively. Overall, we can conclude that our proposed CRF and CRF2O are efficient enough and readily applicable to real-life systems.

5 Related Works

Span-style SRL Pioneered by Gildea and Jurafsky (2000), syntax has long been considered indispensable for span-style SRL (Punyakanok et al., 2008). With the advent of the neural network era, syntax-agnostic models make remarkable progress (Zhou and Xu, 2015; Tan et al., 2018; Cai et al., 2018), mainly owing to powerful model architectures like BiLSTM (Gal and Ghahramani, 2016) or Transformer (Vaswani et al., 2017). Meanwhile, other researchers also pay attention to the utilization of syntax trees, including serving as guidance for argument pruning (He et al., 2018b), as input features (Marcheggiani and Titov, 2017; Xia et al., 2019; Mohammadshahi and Henderson, 2021), or as supervisions for joint learning (Swayamdipta et al., 2018). However, to our best knowledge, very few works have been devoted to mining internal structures of shallow SRL representations. As exceptions, He et al. (2018a); Zhang et al. (2021b) take into account headwords while recognizing arguments. Beyond this, this work proposes to model full argument subtree structures rather than merely headwords and find more competitive results.

Parsing with latent variables Henderson et al. (2008, 2013) design a latent variable model to deliver syntactic and semantic interactions under the setting of joint learning. In more common situations where gold treebanks may lack, Naradowsky et al. (2012); Gormley et al. (2014) use LBP for the inference of semantic graphs and treat latent trees as global factors (Smith and Eisner, 2008)

to provide soft beliefs for reasonable predicate-arguments structures. This work differs in that we make hard constraints on syntax tree structures to conform to the SRL structures, and take only subtrees attached to predicates as latent variables. The intuition behind latent tree models (Meila and Jordan, 2000; Chu et al., 2017; Kim et al., 2017) is to utilize tree structures to provide rich structural interactions for problems with prohibitive high complexity. This idea is also common in many other NLP tasks like text summarization (Liu and Lapata, 2018), sequence labeling (Zhou et al., 2020d), and AMR parsing (Zhou et al., 2020c).

Reduction to tree parsing Researchers have investigated several ways to recover SRL structures from tree structures, due to their high coupling nature (Palmer et al., 2005). Early efforts of Cohn and Blunsom (2005) derive predicate-arguments from pruned phrase structures by using a CKY-style TreeCRF to learn parameters. Johansson and Nugues (2008a) and Choi and Palmer (2010) investigate retrieving semantic boundaries from dependency outputs. Their devised heuristics rely heavily on the quality of output trees, leading to inferior results. Our reduction is also inspired by works on other NLP tasks, including named entity recognition (NER) (Yu et al., 2020), nested NER (Fu et al., 2021; Lou et al., 2022), semantic parsing (Sun et al., 2017; Jiang et al., 2019), and EUD parsing (Anderson and Gómez-Rodríguez, 2021). As the most relevant work, Shi et al. (2020) also propose to reduce SRL to syntactic dependency parsing by integrating syntactic-semantic relations into a single dependency tree by means of joint labels. However, their approach shows non-improved results, possibly due to the label sparsity problem and high back-and-forth conversion loss. Also, they use gold treebank supervisions, while ours does not rely on any hand-annotated syntax data.

6 Discussions and Future Works

The basic idea of this work is to mimic SRL structures with a combination of multiple latent trees. This new perspective sheds light on some natural extensions of our work to other tightly related semantic parsing tasks, e.g., AMR (Zhang et al., 2019a) and UCCA (Jiang et al., 2019).[10] Tasks fall into this type exhibit very flexible graph representation schemes (e.g., *reentrancy* and *discontinuity*) (Zhang et al., 2019b), which are intractable by principled decoding algorithms like dynamic programming. We believe that employing structured inference in spirit of our approaches can provide considerable help in getting rid of greedy span/dependency selections and finding globally optimal structures.

We prefer to reduce SRL to dependency-based tree parsing rather than another paradigm, i.e., constituency parsing, partly because dependencies provide a more transparent bilexical governor-dependent encoding of predicate-argument relations (Hacioglu, 2004). We also do not pursue the way of jointly modeling dependencies and phrasal structures with lexicalized trees (Eisner and Satta, 1999; Yang and Tu, 2022; Lou et al., 2022) as our approach enjoys a lower time complexity of $O(n^3)$. Nonetheless, we admit potential advantages of this kind of modeling (Liu et al., 2022) and leave this as our future work.

There are other interesting perspectives deserve further explorations: given that span-style SRL substantially benefits from our formulation of recovering SRL structures from trees, *can the induced dependency trees learn plausible syntactic structures?* Or in other words, can they agree with linguistic-motivated annotations (Marcus et al., 1993)? We conduct thorough analyses in spirit of Gormley et al. (2014); Li et al. (2021) and give affirmative answers. Due to space limitations, we refer readers to § D and § E for details.

7 Conclusions

In this paper, we propose to reduce span-style SRL to dependency parsing by viewing flat phrasal arguments as latent subtrees, and design a novel span-constrained TreeCRF to accommodate the span structures. Taking inspirations from the parsing literature, we also build a second-order extension and find further gains. Our models are syntax-agnostic and do not rely on any external linguistic syntax knowledge. Experimental results show that, our proposed methods outperform all previous comparable works, achieving new state-of-the-art on both CoNLL05 and CoNLL12 benchmarks. Extensive analyses confirm that our approach enjoys some merits of global structural constraints, meanwhile maintaining acceptable time complexity. Furthermore, we find our modeling of latent subtrees provides effective assistance in terms of long-range dependencies and global consistency.

[10] We thank an anonymous reviewer for pointing out the connection.

8 Acknowledgments

We would like to thank the anonymous reviewers for their valuable comments, and Prof. Zhenghua Li for very helpful feedbacks, suggestions and idea discussions. This work was supported by National Natural Science Foundation of China with Grant No. 62176173, 62076173, U1836222 and a project funded by the Priority Academic Program Development (PAPD) of Jiangsu Higher Education Institutions.

References

Mark Anderson and Carlos Gómez-Rodríguez. 2021. Splitting EUD graphs into trees: A quick and clatty approach. In *Proceedings of IWPT*, pages 167–174, Online.

Marzieh Bazrafshan and Daniel Gildea. 2013. Semantic roles for string to tree machine translation. In *Proceedings of ACL*, pages 419–423, Sofia, Bulgaria.

Jonathan Berant, Andrew Chou, Roy Frostig, and Percy Liang. 2013. Semantic parsing on Freebase from question-answer pairs. In *Proceedings of EMNLP*, pages 1533–1544, Seattle, Washington, USA.

Rexhina Blloshmi, Simone Conia, Rocco Tripodi, and Roberto Navigli. 2021. Generating senses and roles: An end-to-end model for dependency- and span-based semantic role labeling. In *Proceedings of IJCAI*, pages 3786–3793.

Jiaxun Cai, Shexia He, Zuchao Li, and Hai Zhao. 2018. A full end-to-end semantic role labeler, syntactic-agnostic over syntactic-aware? In *Proceedings of COLING*, pages 2753–2765, Santa Fe, New Mexico, USA.

Jiong Cai, Yong Jiang, and Kewei Tu. 2017. CRF autoencoder for unsupervised dependency parsing. In *Proceedings of EMNLP*, pages 1638–1643, Copenhagen, Denmark.

Xavier Carreras and Lluís Màrquez. 2005. Introduction to the CoNLL-2005 shared task: Semantic role labeling. In *Proceedings of CoNLL*, pages 152–164, Ann Arbor, Michigan.

Jinho Choi and Martha Palmer. 2010. Retrieving correct semantic boundaries in dependency structure. In *Proceedings of LAW*, pages 91–99, Uppsala, Sweden.

Janara Christensen, Mausam, Stephen Soderland, and Oren Etzioni. 2010. Semantic role labeling for open information extraction. In *Proceedings of WS*, pages 52–60, Los Angeles, California.

Shanbo Chu, Yong Jiang, and Kewei Tu. 2017. Latent dependency forest models. In *Proceedings of AAAI*, pages 3733–3739.

Trevor Cohn and Philip Blunsom. 2005. Semantic role labelling with tree conditional random fields. In *Proceedings of CoNLL*, pages 169–172, Ann Arbor, Michigan.

Michael Collins. 2003. Head-driven statistical models for natural language parsing. *CL*, pages 589–637.

Simone Conia and Roberto Navigli. 2020. Bridging the gap in multilingual semantic role labeling: a language-agnostic approach. In *Proceedings of COLING*, pages 1396–1410, Barcelona, Spain (Online).

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of NAACL-HLT*, pages 4171–4186, Minneapolis, Minnesota.

Timothy Dozat and Christopher D. Manning. 2017. Deep biaffine attention for neural dependency parsing. In *Proceedings of ICLR*, Toulon, France.

Jason Eisner. 1996. Three new probabilistic models for dependency parsing: An exploration. In *Proceedings of COLING*, pages 340–345.

Jason Eisner. 2000. *Bilexical Grammars and their Cubic-Time Parsing Algorithms*, pages 29–61. Dordrecht.

Jason Eisner. 2016. Inside-outside and forward-backward algorithms are just backprop (tutorial paper). In *Proceedings of WS*, pages 1–17, Austin, TX.

Jason Eisner and Giorgio Satta. 1999. Efficient parsing for bilexical context-free grammars and head automaton grammars. In *Proceedings of ACL*, pages 457–464, College Park, Maryland, USA.

Nicholas FitzGerald, Oscar Täckström, Kuzman Ganchev, and Dipanjan Das. 2015. Semantic role labeling with neural network factors. In *Proceedings of EMNLP*, pages 960–970, Lisbon, Portugal.

Erick Fonseca and André F. T. Martins. 2020. Revisiting higher-order dependency parsers. In *Proceedings of ACL*, pages 8795–8800, Online.

Yao Fu, Chuanqi Tan, Mosha Chen, Songfang Huang, and Fei Huang. 2021. Nested named entity recognition with partially-observed treecrfs. In *Proceedings of AAAI*, pages 12839–12847, Online.

Yarin Gal and Zoubin Ghahramani. 2016. Dropout as a bayesian approximation: Representing model uncertainty in deep learning. In *Proceedings of ICML*, pages 1050–1059.

Daniel Gildea and Julia Hockenmaier. 2003. Identifying semantic roles using Combinatory Categorial Grammar. In *Proceedings of EMNLP*, pages 57–64.

Daniel Gildea and Daniel Jurafsky. 2000. Automatic labeling of semantic roles. In *Proceedings of ACL*, pages 512–520, Hong Kong.

Matthew R. Gormley, Margaret Mitchell, Benjamin Van Durme, and Mark Dredze. 2014. Low-resource semantic role labeling. In *Proceedings of ACL*, pages 1177–1187, Baltimore, Maryland.

Kadri Hacioglu. 2004. Semantic role labeling using dependency trees. In *Proceedings of COLING*, pages 1273–1276, Geneva, Switzerland.

Jan Hajič, Massimiliano Ciaramita, Richard Johansson, Daisuke Kawahara, Maria Antònia Martí, Lluís Màrquez, Adam Meyers, Joakim Nivre, Sebastian Padó, Jan Štěpánek, Pavel Straňák, Mihai Surdeanu, Nianwen Xue, and Yi Zhang. 2009. The CoNLL-2009 shared task: Syntactic and semantic dependencies in multiple languages. In *Proceedings of CoNLL*, pages 1–18, Boulder, Colorado.

Wenjuan Han, Yong Jiang, and Kewei Tu. 2017. Dependency grammar induction with neural lexicalization and big training data. In *Proceedings of EMNLP*, pages 1683–1688, Copenhagen, Denmark.

Han He and Jinho D. Choi. 2021. The stem cell hypothesis: Dilemma behind multi-task learning with transformer encoders. In *Proceedings of EMNLP*, pages 5555–5577, Online and Punta Cana, Dominican Republic.

Luheng He, Kenton Lee, Omer Levy, and Luke Zettlemoyer. 2018a. Jointly predicting predicates and arguments in neural semantic role labeling. In *Proceedings of ACL*, pages 364–369, Melbourne, Australia.

Luheng He, Kenton Lee, Mike Lewis, and Luke Zettlemoyer. 2017. Deep semantic role labeling: What works and what's next. In *Proceedings of ACL*, pages 473–483, Vancouver, Canada.

Shexia He, Zuchao Li, Hai Zhao, and Hongxiao Bai. 2018b. Syntax for semantic role labeling, to be, or not to be. In *Proceedings of ACL*, pages 2061–2071, Melbourne, Australia.

James Henderson, Paola Merlo, Gabriele Musillo, and Ivan Titov. 2008. A latent variable model of synchronous parsing for syntactic and semantic dependencies. In *Proceedings of CoNLL*, pages 178–182, Manchester, England.

James Henderson, Paola Merlo, Ivan Titov, and Gabriele Musillo. 2013. Multilingual joint parsing of syntactic and semantic dependencies with a latent variable model. *CL*, pages 949–998.

Wei Jiang, Zhenghua Li, Yu Zhang, and Min Zhang. 2019. HLT@SUDA at SemEval-2019 task 1: UCCA graph parsing as constituent tree parsing. In *Proceedings of SemEval*, pages 11–15, Minneapolis, Minnesota, USA.

Yong Jiang, Wenjuan Han, and Kewei Tu. 2016. Unsupervised neural dependency parsing. In *Proceedings of EMNLP*, pages 763–771, Austin, Texas.

Zhanming Jie and Wei Lu. 2019. Dependency-guided LSTM-CRF for named entity recognition. In *Proceedings of EMNLP-IJCNLP*, pages 3862–3872, Hong Kong, China.

Ishan Jindal, Ranit Aharonov, Siddhartha Brahma, Huaiyu Zhu, and Yunyao Li. 2020. Improved semantic role labeling using parameterized neighborhood memory adaptation.

Richard Johansson and Pierre Nugues. 2008a. Dependency-based semantic role labeling of PropBank. In *Proceedings of EMNLP*, pages 69–78, Honolulu, Hawaii.

Richard Johansson and Pierre Nugues. 2008b. Dependency-based syntactic–semantic analysis with PropBank and NomBank. In *Proceedings of CoNLL*, pages 183–187, Manchester, England.

Richard Johansson and Pierre Nugues. 2008c. The effect of syntactic representation on semantic role labeling. In *Proceedings of COLING*, pages 393–400, Manchester, UK.

Yoon Kim, Carl Denton, Luong Hoang, and Alexander M. Rush. 2017. Structured attention networks. In *Proceedings of ICLR*.

Diederik P. Kingma and Jimmy Ba. 2015. Adam: A method for stochastic optimization. In *Proceedings of ICLR*, San Diego, CA, USA.

Dan Klein and Christopher Manning. 2004. Corpus-based induction of syntactic structure: Models of dependency and constituency. In *Proceedings of ACL*, pages 478–485, Barcelona, Spain.

Lingpeng Kong, Alexander M. Rush, and Noah A. Smith. 2015. Transforming dependencies into phrase structures. In *Proceedings of NAACL-HLT*, pages 788–798, Denver, Colorado.

John D. Lafferty, Andrew McCallum, and Fernando C. N. Pereira. 2001. Conditional random fields: Probabilistic models for segmenting and labeling sequence data. In *Proceedings of ICML*, pages 282–289, Williams College, Williamstown, MA, USA.

Guillaume Lample, Miguel Ballesteros, Sandeep Subramanian, Kazuya Kawakami, and Chris Dyer. 2016. Neural architectures for named entity recognition. In *Proceedings of NAACL-HLT*, pages 260–270, San Diego, California.

Phong Le and Willem Zuidema. 2015. Unsupervised dependency parsing: Let's use supervised parsers. In *Proceedings of NAACL-HLT*, pages 651–661, Denver, Colorado.

Tao Li, Parth Anand Jawale, Martha Palmer, and Vivek Srikumar. 2020. Structured tuning for semantic role labeling. In *Proceedings of ACL*, pages 8402–8412, Online.

Zhenghua Li, Min Zhang, Yue Zhang, Zhanyi Liu, Wenliang Chen, Hua Wu, and Haifeng Wang. 2016. Active learning for dependency parsing with partial annotation. In *Proceedings of ACL*, pages 344–354, Berlin, Germany.

Zuchao Li, Shexia He, Hai Zhao, Yiqing Zhang, Zhuosheng Zhang, Xi Zhou, and Xiang Zhou. 2019. Dependency or span, end-to-end uniform semantic role labeling. In *Proceedings of AAAI*, pages 6730–6737.

Zuchao Li, Hai Zhao, Shexia He, and Jiaxun Cai. 2021. Syntax role for neural semantic role labeling. *CL*, pages 529–574.

Yankai Lin, Zhiyuan Liu, and Maosong Sun. 2017. Neural relation extraction with multi-lingual attention. In *Proceedings of ACL*, pages 34–43, Vancouver, Canada.

Ding Liu and Daniel Gildea. 2010. Semantic role features for machine translation. In *Proceedings of COLING*, pages 716–724, Beijing, China.

Tianyu Liu, Yuchen Jiang, Ryan Cotterell, and Mrinmaya Sachan. 2022. A structured span selector. In *Proceedings of NAACL*, pages 2629–2641, Seattle, United States.

Yang Liu and Mirella Lapata. 2018. Learning structured text representations. *TACL*, pages 63–75.

Ilya Loshchilov and Frank Hutter. 2019. Decoupled weight decay regularization. In *Proceedings of ICLR*, New Orleans, LA, USA.

Chao Lou, Songlin Yang, and Kewei Tu. 2022. Nested named entity recognition as latent lexicalized constituency parsing. In *Proceedings of ACL*, pages 6183–6198, Dublin, Ireland.

Diego Marcheggiani and Ivan Titov. 2017. Encoding sentences with graph convolutional networks for semantic role labeling. In *Proceedings of EMNLP*, pages 1506–1515, Copenhagen, Denmark.

Mitchell P. Marcus, Beatrice Santorini, and Mary Ann Marcinkiewicz. 1993. Building a large annotated corpus of English: The Penn Treebank. *CL*, pages 313–330.

Ryan McDonald, Koby Crammer, and Fernando Pereira. 2005. Online large-margin training of dependency parsers. In *Proceedings of ACL*, pages 91–98, Ann Arbor, Michigan.

Ryan McDonald and Fernando Pereira. 2006. Online learning of approximate dependency parsing algorithms. In *Proceedings of EACL*, pages 81–88, Trento, Italy.

Marina Meila and Michael I. Jordan. 2000. Learning with mixtures of trees. *JMLR*, pages 1–48.

Alireza Mohammadshahi and James Henderson. 2021. Syntax-aware graph-to-graph transformer for semantic role labelling.

Jason Naradowsky, Sebastian Riedel, and David Smith. 2012. Improving NLP through marginalization of hidden syntactic structure. In *Proceedings of EMNLP*, pages 810–820, Jeju Island, Korea.

Joakim Nivre, Yoav Goldberg, and Ryan McDonald. 2014. Squibs: Constrained arc-eager dependency parsing. *CL*, pages 249–257.

Hiroki Ouchi, Hiroyuki Shindo, and Yuji Matsumoto. 2018. A span selection model for semantic role labeling. In *Proceedings of EMNLP*, pages 1630–1642, Brussels, Belgium.

Martha Palmer, Daniel Gildea, and Paul Kingsbury. 2005. The Proposition Bank: An annotated corpus of semantic roles. *CL*, pages 71–106.

Giovanni Paolini, Ben Athiwaratkun, Jason Krone, Jie Ma, Alessandro Achille, RISHITA ANUBHAI, Cicero Nogueira dos Santos, Bing Xiang, and Stefano Soatto. 2021. Structured prediction as translation between augmented natural languages. In *Proceedings of ICLR*.

Fernando C. N. Pereira and David H. D. Warren. 1983. Parsing as deduction. In *Proceedings of ACL*, pages 137–144, Cambridge, Massachusetts, USA.

Sameer Pradhan, Alessandro Moschitti, Nianwen Xue, Hwee Tou Ng, Anders Björkelund, Olga Uryupina, Yuchen Zhang, and Zhi Zhong. 2013. Towards robust linguistic analysis using OntoNotes. In *Proceedings of CoNLL-WS*, pages 143–152, Sofia, Bulgaria.

Sameer Pradhan, Alessandro Moschitti, Nianwen Xue, Olga Uryupina, and Yuchen Zhang. 2012. CoNLL-2012 shared task: Modeling multilingual unrestricted coreference in OntoNotes. In *Proceedings of CoNLL-WS*, pages 1–40, Jeju Island, Korea.

Vasin Punyakanok, Dan Roth, and Wen-tau Yih. 2008. The importance of syntactic parsing and inference in semantic role labeling. *CL*, pages 257–287.

Vasin Punyakanok, Dan Roth, Wen-tau Yih, and Dav Zimak. 2004. Semantic role labeling via integer linear programming inference. In *Proceedings of COLING*, pages 1346–1352, Geneva, Switzerland.

Alexander Rush. 2020. Torch-struct: Deep structured prediction library. In *Proceedings of ACL*, pages 335–342, Online.

Yikang Shen, Yi Tay, Che Zheng, Dara Bahri, Donald Metzler, and Aaron Courville. 2021. StructFormer: Joint unsupervised induction of dependency and constituency structure from masked language modeling. In *Proceedings of ACL-IJCNLP*, pages 7196–7209, Online.

Peng Shi and Jimmy Lin. 2019. Simple bert models for relation extraction and semantic role labeling.

Tianze Shi, Igor Malioutov, and Ozan Irsoy. 2020. Semantic role labeling as syntactic dependency parsing. In *Proceedings of EMNLP*, pages 7551–7571, Online.

David Smith and Jason Eisner. 2008. Dependency parsing by belief propagation. In *Proceedings of EMNLP*, pages 145–156, Honolulu, Hawaii.

Emma Strubell, Patrick Verga, Daniel Andor, David Weiss, and Andrew McCallum. 2018. Linguistically-informed self-attention for semantic role labeling. In *Proceedings of EMNLP*, pages 5027–5038, Brussels, Belgium.

Weiwei Sun, Junjie Cao, and Xiaojun Wan. 2017. Semantic dependency parsing via book embedding. In *Proceedings of ACL*, pages 828–838, Vancouver, Canada.

Swabha Swayamdipta, Sam Thomson, Kenton Lee, Luke Zettlemoyer, Chris Dyer, and Noah A. Smith. 2018. Syntactic scaffolds for semantic structures. In *Proceedings of EMNLP*, pages 3772–3782, Brussels, Belgium.

Oscar Täckström, Kuzman Ganchev, and Dipanjan Das. 2015. Efficient inference and structured learning for semantic role labeling. *TACL*, pages 29–41.

Zhixing Tan, Mingxuan Wang, Jun Xie, Yidong Chen, and Xiaodong Shi. 2018. Deep semantic role labeling with self-attention. In *Proceedings of AAAI*, pages 4929–4936.

Kristina Toutanova, Aria Haghighi, and Christopher D. Manning. 2008. A global joint model for semantic role labeling. *CL*, pages 161–191.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in NIPS*, pages 5998–6008, Long Beach, California, USA.

Xinyu Wang, Jingxian Huang, and Kewei Tu. 2019. Second-order semantic dependency parsing with end-to-end neural networks. In *Proceedings of ACL*, pages 4609–4618, Florence, Italy.

Qingrong Xia, Zhenghua Li, Min Zhang, Meishan Zhang, Guohong Fu, Rui Wang, and Luo Si. 2019. Syntax-aware neural semantic role labeling. In *Proceedings of AAAI*, pages 7305–7313.

Lu Xu, Zhanming Jie, Wei Lu, and Lidong Bing. 2021. Better feature integration for named entity recognition. In *Proceedings of NAACL-HLT*, pages 3457–3469, Online.

Songlin Yang, Yong Jiang, Wenjuan Han, and Kewei Tu. 2020. Second-order unsupervised neural dependency parsing. In *Proceedings of COLING*, pages 3911–3924, Barcelona, Spain (Online).

Songlin Yang and Kewei Tu. 2022. Combining (second-order) graph-based and headed-span-based projective dependency parsing. In *Findings of ACL*, pages 1428–1434, Dublin, Ireland.

Songlin Yang, Yanpeng Zhao, and Kewei Tu. 2021. Neural bi-lexicalized PCFG induction. In *Proceedings of ACL-IJCNLP*, pages 2688–2699, Online.

Wen-tau Yih, Matthew Richardson, Chris Meek, Ming-Wei Chang, and Jina Suh. 2016. The value of semantic parse labeling for knowledge base question answering. In *Proceedings of ACL*, pages 201–206, Berlin, Germany.

Juntao Yu, Bernd Bohnet, and Massimo Poesio. 2020. Named entity recognition as dependency parsing. In *Proceedings of ACL*, pages 6470–6476, Online.

Liwen Zhang, Ge Wang, Wenjuan Han, and Kewei Tu. 2021a. Adapting unsupervised syntactic parsing methodology for discourse dependency parsing. In *Proceedings of ACL-IJCNLP*, pages 5782–5794, Online.

Sheng Zhang, Xutai Ma, Kevin Duh, and Benjamin Van Durme. 2019a. AMR parsing as sequence-to-graph transduction. In *Proceedings of ACL*, pages 80–94, Florence, Italy.

Sheng Zhang, Xutai Ma, Kevin Duh, and Benjamin Van Durme. 2019b. Broad-coverage semantic parsing as transduction. In *Proceedings of EMNLP-IJCNLP*, pages 3786–3798, Hong Kong, China.

Yu Zhang, Zhenghua Li, and Min Zhang. 2020. Efficient second-order TreeCRF for neural dependency parsing. In *Proceedings of ACL*, pages 3295–3305, Online.

Zhisong Zhang, Emma Strubell, and Eduard Hovy. 2021b. Comparing span extraction methods for semantic role labeling. In *Proceedings of SPNLP*, pages 67–77, Online.

Jie Zhou and Wei Xu. 2015. End-to-end learning of semantic role labeling using recurrent neural networks. In *Proceedings of ACL-IJCNLP*, pages 1127–1137, Beijing, China.

Junru Zhou, Zuchao Li, and Hai Zhao. 2020a. Parsing all: Syntax and semantics, dependencies and spans. In *Findings of EMNLP*, pages 4438–4449, Online.

Junru Zhou, Zhuosheng Zhang, Hai Zhao, and Shuailiang Zhang. 2020b. LIMIT-BERT : Linguistics informed multi-task BERT. In *Findings of EMNLP*, pages 4450–4461, Online.

Qiji Zhou, Yue Zhang, Donghong Ji, and Hao Tang. 2020c. AMR parsing with latent structural information. In *Proceedings of ACL*, pages 4306–4319, Online.

Shilin Zhou, Qingrong Xia, Zhenghua Li, Yu Zhang, and Min Zhang. 2022. Fast and accurate span-based semantic role labeling as graph parsing. In *Proceedings of COLING*.

Yang Zhou, Yong Jiang, Zechuan Hu, and Kewei Tu. 2020d. Neural latent dependency model for sequence labeling.

Hao Zhu, Yonatan Bisk, and Graham Neubig. 2020. The return of lexical dependencies: Neural lexicalized PCFGs. *TACL*, pages 647–661.

A Implementation Details

In this work, we set up two alternative model architectures, i.e, LSTM-based and PLM-based. For the LSTM-based model, we directly adopt most settings of Dozat and Manning (2017) with some adaptions. The input vector of each token $x_i \in \boldsymbol{x}$ is the concatenation of three parts,

$$\mathbf{e}_i = \left[\mathbf{e}_i^{\text{word}}; \mathbf{e}_i^{\text{lemma}}; \mathbf{e}_i^{\text{char}}\right]$$

where $\mathbf{e}_i^{\text{word}}$ and $\mathbf{e}_i^{\text{lemma}}$ are word and lemma embeddings, and $\mathbf{e}_i^{\text{char}}$ is the outputs of a CharLSTM layer (Lample et al., 2016). We set the dimension of lemma and CharLSTM representations to 100 in our setting. We next feed the input embeddings into 3-layer BiLSTMs (Gal and Ghahramani, 2016) to get contextualized representations with dimension 800.

$$\mathbf{h}_0, \mathbf{h}_1, \ldots, \mathbf{h}_n = \texttt{BiLSTMs}(\mathbf{e}_0, \mathbf{e}_1, \ldots, \mathbf{e}_n)$$

Other dimension settings are kept the same as biaffine parser (Dozat and Manning, 2017). Following Zhang et al. (2020), we set the hidden size of Triaffine layer to 100 for CRF2O additionally. The training process continues at most 1,000 epochs and is early stopped if the performance on Dev data does not increase in 100 consecutive epochs. In practice, we observe that the training procedure is often stopped within 300 epochs (∼12 hours), which is efficient enough.

For PLM-based models, we opt to directly fine-tune the PLM layers without cascading word embedding and LSTM layers for the sake of simplicity. We use "`bert-large-cased`" for BERT, and "`roberta-large`" for RoBERTa respectively. We train the model for 20 epochs with roughly 1,000 tokens per batch and use AdamW (Kingma and Ba, 2015; Loshchilov and Hutter, 2019) with $\beta_1 = 0.9, \beta_2 = 0.9$ and $\lambda = 0$ for parameter optimization . The learning rate is 5×10^{-5} for PLMs, and 10^{-3} for the rest components. We adopt the warmup strategy in the first 10% of the training steps, and then apply a linear decay to the learning rate in the remaining steps.

Algorithm 1 The Second-order Inside Algorithm.

1: **Define:** $I, S, C \in \mathbf{R}^{n \times n \times b}$
2: $\triangleright$ b is batch size
3: **Initialize:** $C_{i,i} = \mathbf{0}, 0 \leqslant i \leqslant n$
4: **for** $w = 1$ **to** n **do** $\triangleright$ span width
5: *Parallelization on* $0 \leqslant i; j = i + w \leqslant n$
$I_{i,j} \leftarrow \log(\exp(C_{i,i} + C_{j,i+1})$
6: $\quad + \sum_{i<r<j} \exp(I_{i,r} + S_{r,j} + \mathrm{s}(i,r,j)))$
$\quad + \mathrm{s}(i,j)$
$I_{j,i} \leftarrow \log(\exp(C_{j,j} + C_{i,j-1})$
7: $\quad + \sum_{i<r<j} \exp(I_{j,r} + S_{r,i} + \mathrm{s}(i,r,j)))$
$\quad + \mathrm{s}(j,i)$
8: $S_{i,j} \leftarrow \log \sum_{i \leqslant r<j} \exp(C_{i,r} + C_{j,r+1})$
9: $C_{i,j} \leftarrow \log \sum_{i<r \leqslant j} \exp(I_{i,r} + C_{r,j})$
10: $C_{j,i} \leftarrow \log \sum_{i \leqslant r<j} \exp(I_{j,r} + C_{r,i})$
11: **end for**
12: **return** $C_{0,n}$

B The Inside Algorithm

We give the pseudocode of the common second-order Inside algorithm (McDonald and Pereira, 2006) in Alg. 1 as additional explanations to Fig. 3. The difference between the common second-order Inside algorithm and our proposed span-constrained one lies in the rule constraints green highlighted in Fig. 3.

In Line 3, $C_{i,i}$ corresponds to the axiom items with initial score $\mathbf{0}$. Line 6 corresponds to two merge operations in Fig. 3. The incomplete span $I_{i,j}$ ( ) is obtained by summing over either all pairs of complete span $C_{i,i}$ and $C_{j,i+1}$ (**R-LINK**) or pairs of the incomplete span $I_{i,r}$ and the sibling span $S_{j,r}$ (**R-LINK2**). In Line 8, the sibling span $S_{i,j}$ ( ) is obtained by summing over all pairs of complete span $C_{i,r}$ and $C_{j,r+1}$ (**COMB**). Line 9 describes the similar merging operation on all pairs of the incomplete span $I_{i,r}$ and the complete span $C_{r,j}$, resulting a complete span

	WSJ			Brown		
	P	R	F_1	P	R	F_1
SA$^{\diamondsuit}$	84.17	83.28	83.72	72.98	70.1	71.51
SA$^{\diamondsuit}_{\text{ELMo}}$	86.21	85.98	86.09	77.1	75.61	76.35
G2G$^{\clubsuit}_{\text{BERT}}$	86.40	87.79	87.08	78.76	80.06	79.40
LIMIT$^{\clubsuit}_{\text{BERT}}$	86.62	89.12	87.85	79.58	83.05	81.28
ParsingAll$^{\clubsuit}_{\text{BERT}}$	86.77	88.49	87.62	79.06	81.67	80.34
ParsingAll$^{\clubsuit}_{\text{XLNet}}$	87.65	89.66	88.64	80.77	83.92	82.31
CRF$_{\text{BERT}}$	86.98	88.28	87.63	79.19	80.92	80.05
CRF2O$_{\text{BERT}}$	87.00	88.76	87.87	79.08	81.50	80.27
CRF$_{\text{RoBERTa}}$	87.20	88.67	87.93	79.29	81.48	80.38
CRF2O$_{\text{RoBERTa}}$	87.35	89.34	88.33	79.95	82.32	81.12
w/ gold predicates						
ParsingAll$^{\clubsuit}_{\text{BERT}}$	89.04	88.79	88.91	81.89	80.98	81.43
ParsingAll$^{\clubsuit}_{\text{XLNet}}$	89.89	89.74	89.81	85.35	84.57	84.96
TANL$^{\diamondsuit}_{\text{T5}}$	-	-	89.3	-	-	82.0
CRF$_{\text{BERT}}$	88.93	88.58	88.76	82.87	81.67	82.27
CRF2O$_{\text{BERT}}$	89.00	89.03	89.02	82.81	82.35	82.58
CRF$_{\text{RoBERTa}}$	89.29	88.99	89.15	83.22	82.42	82.82
CRF2O$_{\text{RoBERTa}}$	89.45	89.63	89.54	83.89	83.39	83.64

Table 5: Comparisons with other less comparable works on CoNLL05 WSJ and Brown data. $^{\clubsuit}$ means using linguistic syntax knowledge; $^{\diamondsuit}$ means different evaluation methods. SA: Strubell et al. (2018); ParsingAll: Zhou et al. (2020a); LIMIT: Zhou et al. (2020b); G2G: Mohammadshahi and Henderson (2021); TANL: Paolini et al. (2021).

$C_{i,j}$ ($\overset{i}{\ulcorner}\!\!\!\rightarrow\!\!\!\underset{j}{\urcorner}$) (**R-COMB**). Line 7 and Line 10 is the symmetric L-rules, which are omitted in Fig. 3.

C More Comparisons

In Table 5, for reference, we list the results of some works with different experimental settings and therefore less comparable. For example, Paolini et al. (2021) and Strubell et al. (2018)[11] adopt different evaluation metrics, resulting in slightly higher F_1 values than official tools. Nonetheless, we find that our CRF2O with RoBERTa achieves 89.54 F_1 on WSJ data under the *w/ gold predicates* setting, showing very competitive results when compared with T5-based model of Paolini et al. (2021). Zhou et al. (2020a) propose a joint-learning framework, integrating both (dependency/constituency) syntactic parse trees and dependency-based SRL resources to enhance their models. Their ablation studies show that using syntax trees brought an overall improvement of 1.6 F_1 score on CoNLL05 Dev data. We believe that we could achieve similar or even higher results than their syntax-aware XLNet-based models by incorporating human-annotated syntax knowledge. However, exploring different ways of injecting syntax is not the core of this paper. We take this as our future work.

[11] Under the *end-to-end* setting, different from the standard pratice (He et al., 2018a), Strubell et al. (2018) only ran the evaluation tool once, resulting in slightly higher precision values. See discussions in their code issue.

	P	R	F_1
CRF	75.28	75.24	75.26
CRF$_{\text{BERT}}$	84.70	84.39	84.54
w/ gold syntax			
Johansson and Nugues (2008c)	-	-	84.32
Li et al. (2019)$_{\text{ELMo}}$	-	-	89.20
CRF$_{\text{BERT}}$	**93.56**	**93.22**	**93.39**

Table 6: Results for dependency-based evaluation on CoNLL09 Test data under *w/o.* and *w/ gold syntax* settings.

D Dependency-based evaluation

Observing that our CRF model can conveniently determine dependencies from predicates to span headwords as by-products of constructing arguments, we therefore conduct dependency-based evaluation on CoNLL09 Test data (Hajič et al., 2009) to measure the quality of induced dependencies. As CoNLL09 data shares the same text content with CoNLL05, we directly make use of the model trained on CoNLL05 to obtain the results of CoNLL09 Test. Following Johansson and Nugues (2008b); Li et al. (2019), we also compare our CRF outputs with the upper bound of utilizing gold syntax tree to determine the headwords of predicted arguments. Since CoNLL05 contains only verbal predicates, we discard all nominal predicate-argument structures under the guidance of POS tags starting with `N*`. Word senses and self-loops are removed as well.

Results are listed in Table 6, from which we can draw some observations: 1) after using BERT, CRF outperforms LSTM-based model (75.26) by a large margin, implying BERT provides fruitful prior knowledge for dependency induction; 2) our CRF with BERT achieves 84.54 F_1 on CoNLL09 Test, exhibiting very promising performance even when compared to models using gold syntax (Johansson and Nugues, 2008b; Li et al., 2019). This indicates that the dependencies induced by CRF are highly in line with gold dependency-based annotations, illuminating potential extensions of our work on supervised dependency-based SRL.

Rules	Models	WSJ
Stanford	NL-PCFGs (Zhu et al., 2020)	40.5
	NBL-PCFGs (Yang et al., 2021)	39.1
	StructFormer (Shen et al., 2021)	46.2
	CRF	48.0
	CRF_{BERT}	**65.4**
	w/ gold POS tags (for reference)	
Collins	DMV (Klein and Manning, 2004)	39.4
	MaxEnc (Le and Zuidema, 2015)	65.8
	NDMV (Jiang et al., 2016)	57.6
	CRFAE (Cai et al., 2017)	55.7
	L-NDMV (Han et al., 2017)	59.5
	NDMV2o (Yang et al., 2020)	**67.5**

Table 7: Grammar induction results of our CRF model under different head-finding rules.

E Grammar Induction

To gain further insights, we make use of the scores defined in Eq. 5 to extract full dependency tree structures. Surprisingly, we find they are highly in agreement with expert-designed grammars (Marcus et al., 1993) when examined on the grammar induction task (Klein and Manning, 2004).

We show precise grammar induction results in Table 7. The results are not comparable to typical methods like DMV (Klein and Manning, 2004) or CRFAE (Cai et al., 2017), as they use gold POS tags as guidance, and we use Stanford Dependencies rather than Collins rules (Collins, 2003). Under similar settings, however, our learned task-specific trees perform significantly better than recent works.

Another interesting observation is that the gap between the BERT-based model and the LSTM-based model is much larger than that on SRL results. This implies LSTMs tend to be more fitted to SRL structures, while BERT is able to provide a strong inductive bias for syntax induction.

Noisy Label Regularisation for Textual Regression

Yuxia Wang[♠] **Timothy Baldwin**[♠♡] **Karin Verspoor**[♠♢]

♠ The University of Melbourne, Melbourne, Victoria, Australia
♡ MBZUAI, Abu Dhabi, UAE
♢RMIT University, Melbourne, Victoria, Australia
yuxiaw@student.unimelb.edu.au
tb@ldwin.net karin.verspoor@rmit.edu.au

Abstract

Training with noisy labelled data is known to be detrimental to model performance, especially for high-capacity neural network models in low-resource domains. Our experiments suggest that standard regularisation strategies, such as weight decay and dropout, are ineffective in the face of noisy labels. We propose a simple noisy label detection method that prevents error propagation from the input layer. The approach is based on the observation that the projection of noisy labels is learned through memorisation at advanced stages of learning, and that the Pearson correlation is sensitive to outliers. Extensive experiments over real-world human-disagreement annotations as well as randomly-corrupted and data-augmented labels, across various tasks and domains, demonstrate that our method is effective, regularising noisy labels and improving generalisation performance.

1 Introduction

Modern deep neural networks (DNNs) have millions or billions of trainable parameters, far more than the number of examples they are trained on. To avoid over-fitting, they are heavily reliant on large-scale training, including data derived through methods such as self supervision, data augmentation, and self labelling (Devlin et al., 2019; Wei and Zou, 2019; Wang et al., 2020c). However, such methods inevitably introduce noise, through biased data, unnatural inputs, or incorrect labels.

Weak perturbations applied to inputs can improve model performance by forcing DNNs to learn noise-invariant latent representations (Tang and Eliasmith, 2010; Goodfellow et al., 2016). But training with noisy labels has been shown to be detrimental to generalisation performance across tasks including image classification (Tanaka et al., 2018), dialogue generation (Akama et al., 2020), and entity–relation extraction (Chen et al., 2020). DNNs fit noisy labels by "memorising" each example — over-fitting corrupted training sets, and yielding poor generalisation (Arpit et al., 2017).

Given this background, our focus in this paper is on how to alleviate memorisation and improve generalisation when training with noisy labels. Previous related work has proposed three directions: (1) regularisation techniques (Arpit et al., 2017); (2) augmenting the loss function with an explicit representation of the distribution of noise (Sukhbaatar et al., 2015; Patrini et al., 2017); and (3) explicit detection of noisy labels (Tanaka et al., 2018; Nguyen et al., 2020; Lee and Chung, 2020; Desmond et al., 2020). However, the vast majority of this work has focused on classification tasks, and there has been very little work in the context of regression tasks and low-resource domains. In this work, we fill the gap by targeting noisy label regularisation for text regression, in the form of semantic text similarity (STS), sentiment analysis, and machine translation quality assessment.

Our work makes three contributions: (1) empirical clarification of the role of explicit regularisation in noisy label training in a regression setting; (2) proposal of an effective noisy label detection method for continuous labels; and (3) extensive experiments across various regression tasks under both real-world and synthetic noisy labels, including state-of-the-art results on MedSTS. The code associated with this paper is available at: https://github.com/yuxiaw/Regularise-Regression-Noisy-Labels.

2 Related Work

2.1 Regularisation of Noisy Labels in Classification Settings

Regularisation: Explicit regularisation techniques such as dropout, weight decay, or data augmentation can help to alleviate over-fitting, improving model generalisation (Arpit et al., 2017; Tanaka et al., 2018). They do not, however, prevent classi-

Proceedings of the 29th International Conference on Computational Linguistics, pages 4228–4240
October 12–17, 2022.

fier degradation caused by noisy labels (Harutyunyan et al., 2020). Gradient descent with early stopping and its variants are provably robust to noisy labels (Li et al., 2020; Hu et al., 2020). While empirically verified to be effective for image classification, their performance in textual regression tasks is unknown.

Noise Distribution Matrix: An alternative approach is to correct the loss function with a noise distribution transition matrix (Sukhbaatar et al., 2015; Patrini et al., 2017; Yao et al., 2019; Tanno et al., 2019). Formally, let l and l^{GT} be the noisy and ground-truth labels. The noise transition matrix T is defined as $t_{ij} = p(l = j|l^{GT} = i)$, where the element of the i-th row and the j-th column t_{ij} represents the probability of mis-annotating golden class i to incorrect label j. The cross entropy loss is modified to $\mathcal{L}(\boldsymbol{\theta}, X, Y) = \frac{1}{n}\sum_{n=1}^{n} \log(\mathbf{y}_i^{\intercal}\mathbf{s}(\boldsymbol{\theta}, \mathbf{x}_i))$, where $\mathbf{s}(\boldsymbol{\theta}, \cdot)$ is the classifier. In classification tasks, the probability of misclassification between classes $p(l = j|l^{GT} = i)$ is well-defined. However, it is not clear how to define the matrix T for continuous output variables in a regression setting.

Noisy Label Detection has been explored extensively in classification settings, especially for images, but has received very little attention for textual regression problems. Noisy instances are typically identified based on model prediction (Zheng et al., 2020; Ye et al., 2021), such as comparing predicted labels (pseudo-labels) l^P with annotated labels l^A during training (Tanaka et al., 2018; Berthelot et al., 2019; Nguyen et al., 2020; Lee and Chung, 2020; Desmond et al., 2020), and the label distribution confidence (Liu et al., 2020). However, exact label match ($l^A = l^P$) is too strict a requirement for regression tasks. We relax the criterion to a range controllable by a threshold τ, i.e. $|l^A - l^P| < \tau$. This makes it identical to the loss-based criterion $(l^A - l^P)^2 < \tau$ in regression using mean-squared error loss (MSE). Specifically, instances that result in small loss can be considered to be clean (Shen and Sanghavi, 2019).

2.2 Detect Noisy Labels in Regression

To our knowledge, the only research addressing noisy labels in a textual regression setting is: (1) Wang et al. (2022), who select high-disagreement labels using the predictive variance of uncertainty models; and (2) Takamoto et al. (2020), who identify outliers based on the absolute difference between teacher model predictions and target labels.

Noise filtering also relates to data sampling in active learning. It aims to select the most informative/useful data points from an unlabelled pool, leveraging the least labelling effort to reach the best performance.

Sampling in Active learning: Regression tasks are also under-researched in the active learning literature (Elreedy et al., 2019; Zhang et al., 2020). Cai et al. (2013) sample data associated with the maximum gradient of the loss function, typically based on squared error, and Sugiyama (2006) aims to minimise the conditional expectation of the generalisation error. These are akin to the loss criteria in noisy label identification.

Separately, Wu (2019) considers representativeness and diversity in initial data collection and sequential query selection, and Wu and Huang (2022) select the most beneficial samples to label based on three emotion primitives: valence, arousal, and dominance for affect estimation. However, these methods are too domain-specific to adapt to general-purpose regression tasks.

3 Task and Datasets

In this paper, we investigate text regression across three separate tasks, and a total of 10 datasets.

3.1 Tasks

The three tasks we target in this research are STS, sentiment analysis, and machine translation quality estimation, which we outline below.

Semantic textual similarity (**STS**) assesses the degree of semantic equivalence between two (short) texts (Corley and Mihalcea, 2005). The aim is to predict a similarity score for a sentence pair $(S1, S2)$, generally in the range $[0, 5]$, where 0 indicates complete dissimilarity and 5 indicates equivalence in meaning. As an example:

S1: *Total minutes spent in timed codes: 10 mins.*
S2: *Total minutes spent in timed codes: 33 mins.*

is labelled 4, as the two texts differ only in very specific content (underlined).

Sentiment analysis (**SA**) rating involves predicting a sentiment score for a review S, in the range 1 (extremely negative) to 5 (extremely positive).

Machine translation quality estimation, based on the direct assessment (**DA**) approach (Graham et al., 2017), aims to predict a normalised quality score for text pair $(S1, S2)$, where $S2$ is machine translated from $S1$. As such, it is similar to STS, but differs in that it is cross-lingual.

3.2 Datasets

We evaluate on different-sized datasets across various domains for STS and SA, and two identically-sized datasets for DA, as summarised in Table 1.

For STS, we use: three large-scale general datasets — STS-B (Cer et al., 2017), SICK-R (Marelli et al., 2014), and STS-G (Wang et al., 2020c); and two small clinical data sets — MedSTS (Wang et al., 2018) and N2C2-STS (Wang et al., 2020a).

For SA, we use: a large-scale product review dataset — Yelp (Sabnis, 2018); and two small datasets of movie and paper reviews — 10Movie (Benlahbib, 2019) and PeerRead (Kang et al., 2018). We augment 10Movie with 700 examples from IMDB movie reviews (Maas et al., 2011) (see Appendix A.1 for details of the label conversion process), and also augment PeerRead with 399 Spanish paper reviews (Keith et al., 2017) which we automatically translate into English.

For DA, we employ two language pairs from WMT2020 (Specia et al., 2020), namely ru-en and ro-en, which are low- and medium-resource language pairs, respectively.

As evaluation metrics, we use Pearson's correlation (r) and Spearman's correlation (ρ) between the predicted and gold standard scores.

3.3 Notation and Loss Function

Throughout this paper, raw examples, column vectors, and matrices are denoted in lower-case italics, bold, and upper-case italics, respectively (e.g. x, $\mathbf{x}$ and X). $\boldsymbol{\theta}_{encoder}$ and $\boldsymbol{\theta}_{reg}$ represent parameters of the transformer encoder and task-specific regression layers, and $f(\boldsymbol{\theta}, \cdot)$ refers to the whole model. Assuming a dataset with N instances $\mathcal{D} = \{(x_1, y_1), \cdots, (x_i, y_i), \cdots, (x_N, y_N)\}$, where (x_i, y_i) is the ith instance of $\mathcal{D}$, $y_i \in [0, 5]$, $\mathbf{x}_i = s(\boldsymbol{\theta}_{encoder}, x_i)$ is the embedding of x_i. The loss function is the empirical risk of the mean square error (MSE): $\mathcal{L} = \frac{1}{N}\sum_{i=1}^{N}(f(\boldsymbol{\theta}, x_i) - y_i)^2$.

4 Case Study

We first examine the susceptibility of DNNs to overfit random labels (Zhang et al., 2017), based on the clinical N2C2-STS data set using BERT (Devlin et al., 2019). Then we conduct ablation experiments using various regularisation techniques, to observe whether they can reduce the degradation caused by noisy labels.

Dataset	Size (Train, Test, Dev)	Range	Domain
SICK-R (2014)	4500, 4927, 500	$[1, 5]$	general
STS-B (2017)	5749, 1379, 1500	$[0, 5]$	general
STS-G (2020)	28518, —, —	$[0, 5]$	general
MedSTS (2018)	750, 318, —	$[0, 5]$	clinical
N2C2-STS (2019)	1642, 412, —	$[0, 5]$	clinical
Yelp (2018)	5000, —, —	[1,5]	product
10Movie+IMDB (2019)	1400, 300, —	[1,5]	movie
PeerRead+Spanish (2018)	1638, 290, —	[1,5]	paper
WMT ru-en (2020)	7000, 1000, 1000	$[0, 100]$	low-resource
WMT ro-en (2020)	7000, 1000, 1000	$[0, 100]$	med-resource

Table 1: STS/SA rating/DA datasets. "Train", "Test", "Dev" = number of text pairs; "Range" = label range. In practice, DA is normalised by z-scoring.

Hypothesis Arpit et al. (2017) empirically showed that explicit regularisation, especially dropout coupled with adversarial training, can reduce memorisation of noise without reducing a model's ability to learn. Zhang et al. (2017), on the other hand, argued that it is neither necessary nor sufficient for controlling generalisation error in deep learning. Overall, explicit regularisation may improve generalisation performance, but does not explicitly deal with noisy labels.

4.1 Experiment

Regression Model Structure: The regression model used here and in Section 6 takes the hidden state of the `[CLS]` token output for the single sentence or sentence pair from BERT, $\mathbf{h} \in R^d$. This is fed through a two-layer MLP, structured as:

$$\mathbf{h}' = \tanh(\mathbf{W}\mathbf{h} + \mathbf{b}) \quad (1)$$

$$\hat{y} = \mathbf{w}^\intercal \mathbf{h}' + b \quad (2)$$

where $\hat{y}$ is the predicted score, and $\mathbf{W} \in \mathbb{R}^{d \times d}$, $\mathbf{b}, \mathbf{w} \in \mathbb{R}^d$, and $b \in \mathbb{R}$ are trainable parameters of task-specific layers, denoted as "CLS-BERT".

Corrupted Training Set: To generate partially-noisy training data, we corrupt training set $\mathcal{D}$ by randomly selecting M instances and replacing their labels with $s \in [0, 5]$ sampled from a uniform distribution, forming noisy subset $\mathcal{D}_{noisy}$, leaving the clean partition $\mathcal{D}_{clean}$. Thus the corrupted training set is $\mathcal{D}' = \mathcal{D}_{noisy} \cup \mathcal{D}_{clean}$ (where $|\mathcal{D}| = |\mathcal{D}'| = N$).

Experimental Setup: We randomly split the 1,642 instances in the N2C2-STS training set into 1,242 and 400 instances, as training set $\mathcal{D}$ and a validation set. $M = \alpha \cdot N$ is decided by noise ratio $\alpha \in \{0.2, 0.4, 1.0\}$ to generate three corrupted training sets, denoted *corrupt2*, *corrupt4*, and *corrupt10*, respectively, with corresponding

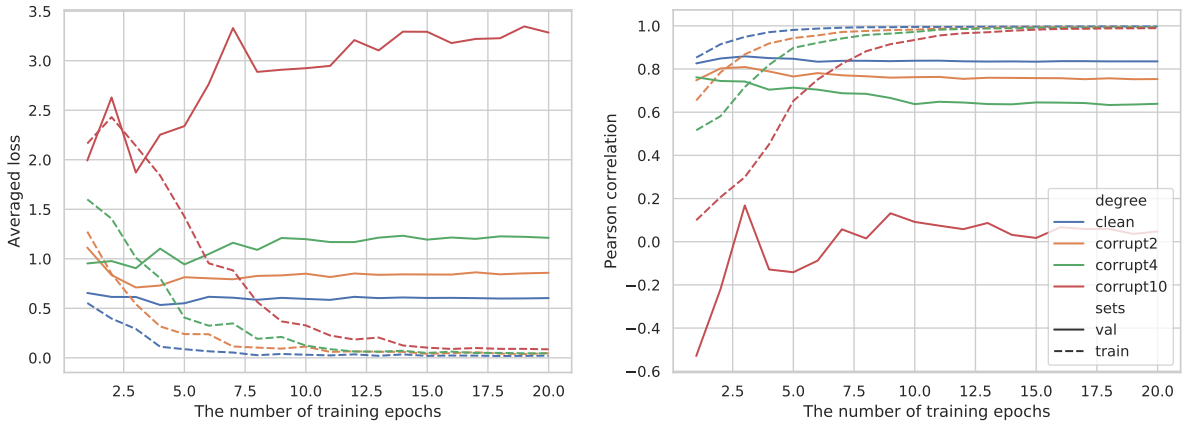

Figure 1: The loss (left) and r (right) on N2C2 train and validation sets over four different degrees of corrupted training set: clean ($\mathcal{D}$), corrupt2, corrupt4 and corrupt10.

clean partitions denoted as *clean2*, *clean4* and $\oslash$ below. Note that the same validation set is used for all experiments in this section and Section 6.1.

The model is trained on *bert-base-uncased*, optimising with a linear scheduler with warmup proportion = 0.1, train batch size = 16, learning rate = 2e-5, and training epochs = 20.

4.1.1 Results

BERT is structured with layer normalisation and residual connections (He et al., 2016), noting that pre-training has been shown to be beneficial for generalisation by alleviating exposure bias. We are thus interested in whether using pre-trained BERT is robust to the effects of random noisy labels.

Using the baseline regression model, we fine-tune over the varyingly-corrupted training sets, each for 20 epochs. As per Figure 1, training with noisy labelled data is detrimental to generalisation performance, and more training exacerbates the effect, especially for noisier data.

As we increase the amount of noise, the training loss decreases and the model takes longer to converge. Particularly on fully-corrupted training set *corrupt10*, the training loss rises first and then starts to fit random labels, taking eight epochs to reach the same training accuracy as the *clean* set $\mathcal{D}$ in the first epoch. This shows that pre-trained BERT can reduce fitting to random labels early in training, and in general slows down convergence. However, since the random labels are fixed across epochs, iterating over the training set multiple times leads to (over)fitting the random labels perfectly.

4.2 Explicit Regularisation

We use the following regularisation techniques:

- **Early stop ("ES")**: return the trained model where the lowest validation error is obtained (based on Pearson's correlation).
- **Weight decay ("WD")**: update parameters by $\theta_t = (1-\beta)\theta_{t-1} - \alpha g_t$, where $\beta \in \{0.01, 0.05, 0.1\}$ is a weight decay coefficient, α is the learning rate, and g_t is the gradient at update step t.
- **Dropout ("DP")**: replace Eq (1) with $\mathbf{h}' = \text{dropout}(\tanh(\mathbf{W}\mathbf{h} + \mathbf{b}))$.
- **Data augmentation**: perform data augmentation via back translation ("BT") or segment reordering ("SR" = randomly permute the order of segments separated by commas or semicolons) following Wang et al. (2020b).
- **Cross Domain Pre-fine-tuning**: fine-tune the model with general-purpose STS-B ("STSB") training set for 3 epochs before fine-tuning on the clinical STS data.

4.2.1 Results

We present the results in Table 2. Comparing rows 1 and 2 (no regularisation vs. early stopping), early stopping improves performance over both clean and corrupted training data, especially on *corrupt4*. Therefore we combine it with the other strategies. Weight decay (WD) (rows 3–5) has negligible impact, but markedly improves *corrupt2* through dropout and data augmentation (rows 6–8 and 11–13), and *corrupt4* through back-translation (rows 7 and 12). Pre-fine-tuning provides large gains in accuracy on both clean and corrupted data sets (row 9), especially coupled with early stopping, weight decay and dropout (rows 10 and 14).

In sum, explicit regularisation improves generalisation performance, not just on corrupted data

ID	Train set setting	$\mathcal{D}$			corrupt2			corrupt4			clean2			clean4		
		r	ρ	loss	r	ρ	loss	r	ρ	loss	r	ρ	loss	r	ρ	loss
1	baseline	.835	.830	0.603	.754	.740	0.858	.639	.636	1.212	.845	.835	0.570	.836	.818	0.598
2	ES	.859	.837	0.614	.809	.797	0.710	.762	.724	0.953	.858	.835	0.529	.851	.822	0.592
3	ES + WD (0.01)	.857	.835	0.518	.810	.799	0.677	.762	.724	0.953	.854	.835	0.535	.852	.816	**0.532**
4	ES + WD (0.05)	.857	.835	0.613	.818	.805	0.670	.760	.722	0.955	.852	.831	0.559	.853	.818	0.544
5	ES + WD (0.1)	.857	.837	0.603	.808	.803	0.680	.769	.731	0.952	.852	.830	0.561	**.854**	**.825**	0.569
6	ES + DP	.858	.839	0.570	.831	.799	0.717	.758	.726	**0.849**	.857	.828	0.519	.840	.818	0.626
7	ES + BT	.858	.838	0.526	.833	.810	0.673	.775	.734	1.124	.850	.825	0.551	.847	.823	0.650
8	ES + SR	.855	.836	0.526	.832	.796	0.615	.761	.749	0.999	**.858**	**.838**	0.541	.849	.823	0.630
9	ES + STSB	.867	.842	0.491	.846	.817	0.745	.783	.778	0.890	.857	.823	0.533	.852	.822	0.571
10	ES + WD (0.1) + STSB	**.867**	**.843**	**0.488**	**.850**	.822	0.724	.783	.778	0.890	.857	.823	0.531	.852	.822	0.571
11	ES + DP + WD (0.1)	.853	.835	0.588	.837	.801	0.679	.756	.737	1.167	.855	.823	**0.517**	.842	.820	0.622
12	ES + DP + WD (0.1) + BT	.859	.833	0.508	.833	.802	0.631	**.796**	.763	0.949	.843	.830	0.587	.839	.817	0.643
13	ES + DP + WD (0.1) + SR	.853	.826	0.538	.833	.787	**0.589**	.790	.763	1.080	.856	.835	0.524	.834	.812	0.627
14	ES + DP + WD (0.1) + STSB	.864	.841	0.520	.848	**.823**	0.731	.779	**.780**	0.943	.853	.822	0.561	.845	.813	0.589

Table 2: Averaged loss, r and ρ on N2C2 validation set with various combinations of regularisation techniques ("ES" = early stopping, "WD" = weight decay, "DP" = dropout, "BT" = data augmentation with back translation, "SR" = data augmentation with segment reordering, "STSB" = fine-tuning over STS-B), over five different training sets ("$\mathcal{D}$" = all clean, "corrupt2" = 0.2 corrupted, "corrupt4" = 0.4 corrupted, "clean2" = clean complementary set of corrupt2, "clean4" = clean complementary set of corrupt4). The best result in each column is bolded.

but on clean data as well. The fact that the best accuracy is still much worse than training on fully clean $\mathcal{D}$, and also worse than on their clean components, clean2 and clean4, also confirms that explicit regularisation can't control the generalisation error caused by noisy labels. It additionally suggests that removing noisy examples could lead to improvements, which we verify in Section 5.

5 Noisy Label Detection

We propose a two-step method to identify noisy labels from training data based on iterative predictions, followed by three different strategies for training with noisy examples, namely: (1) **DIS**: discard noisy examples; (2) **REP**: repair noisy labels with pseudo labels; and (3) **RES**: resample the same number of instances from the "clean" set, to make up for discarded noisy examples.

5.1 Prediction criterion

We relax the requirement of exact match between the pseudo label $\hat{p}$ and the annotated label y by measuring the absolute difference, and them to match if the difference is within a predefined range $|y - \hat{p}| \leq \tau$, which is a tuneable hyper-parameter. The pseudo label is obtained by averaging predictions over multiple training iterations (see line 9 of Algorithm 1).

If τ is small, precision will be low and recall of noisy instances will be high, whereas if τ is large, recall will be low, but precision will be high, negatively affecting training quality. To achieve a balance between precision and recall, we use Pearson's correlation to assess the "noisiness" of each noisy candidate, where a relatively small τ ensures high recall.

Algorithm 1 Train on Noisy Labelled Data

1: **Input:** Training and validation set $\mathcal{D}_{train}$, $\mathcal{D}_{val}$
2: $\mathcal{D}_{clean} \leftarrow \mathcal{D}_{train}$
3: **for** i in range(1, $epochs$) **do**
4: $\quad M_i \leftarrow train(\mathcal{D}_{clean})$
5: $\quad$ **if** $acc(M_i, \mathcal{D}_{val}) \geq acc(M_{best}, \mathcal{D}_{val})$ **then**
6: $\quad\quad M_{best} \leftarrow M_i$
7: $\quad$ **end if**
8: $\quad \hat{\mathbf{y}}_i \leftarrow M_{best}(\mathbf{x})$
9: $\quad$ obtain pseudo labels $\hat{\mathbf{p}}_i \leftarrow \frac{1}{i}\sum_{j=0}^{i} \hat{\mathbf{y}}_j$
10: $\quad$ get noisy candidate set by Prediction Criterion f_1
11: $\quad \mathcal{C}_{clean}, \mathcal{C}_{noise} = f_1(\mathcal{D}_{train}, \hat{\mathbf{p}}_i, \tau)$
12: $\quad$ determine noisy set by Pearson Criterion f_2
13: $\quad \mathcal{D}_{clean}, \mathcal{D}_{noise} = f_2(\mathcal{C}_{clean}, \mathcal{C}_{noise}, m, K, \varepsilon)$
14: $\quad \mathcal{D}_{clean}$ by **Repairing**
15: $\quad \mathcal{D}_{clean} \leftarrow \text{repair}(\mathcal{D}_{noise}, \hat{\mathbf{p}}_i) \cup \mathcal{D}_{clean}$
16: $\quad \mathcal{D}_{clean}$ by **Resampling**
17: $\quad \mathcal{D}_{clean} \leftarrow \text{sample}(\mathcal{D}_{clean}, size(\mathcal{D}_{noise})) \cup \mathcal{D}_{clean}$
18: **end for**
19: **Output:** M_{best}

5.2 Pearson correlation criterion

In Table 2, we found that the smallest loss did not always mean the best performance r. The former is measured for an individual example, while the latter considers correlation of the combined set of instances. We therefore propose to use a summary evaluation metric for selection of noisy instances, adopting Pearson's correlation (r) due to its sensitivity to outliers (Wilcox, 2004; Chok, 2010; Mathur et al., 2020). That is, Pearson correlation is strongly sensitive to linear relationships: r is maximised when two variables are linearly

related to each other, whereas Spearman correlation is maximised when two variables are monotonically related, whether the relationship is linear or not. A single outlier influences the results of r (Rousselet and Pernet, 2012).

Under this criterion, for each noisy candidate (x, y) identified by the prediction criterion (line 13), we randomly select m clean examples from the clean subset $\mathcal{C}_{clean}$ (the training set with noisy candidates filtered out) and calculate the correlation r, then add (x, y) to this set and calculate the new correlation as r'. If a large perturbation is observed — i.e., $|r - r'| > \varepsilon$, e.g. > 0.01 — x is considered to be noisy; otherwise it is considered to be clean.

However, when most of the sampled labels are clustered together with no obvious relationship, r tends to be unjustified, and when points are distributed uniformly, the score is fair (see Figure 1 in Rousselet and Pernet (2012)). This is attributed to the ordinary least square solution: one badly positioned point can have a dramatic influence on the results (Hubert et al., 2008). This instability leads to a less powerful statistical test. To smooth the number (denote as A), we project A of the lower-order into higher-order A^K by repeating this process K times. This can mitigate the large variance and inconsistency in the correlation (Song et al., 2021). We make the final decision by voting: only if all K votes agree that the given training instance is noisy is it removed.

Though the first training epoch is trained on the whole corrupted training set, it does not impact generalisation significantly, because memorised features are not learned in the early stages of training. It also benefits domains with limited training data. That is, even when initialising with pre-trained weights, the STS model is not accurate enough to filter noisy labels accurately in the first iteration, leading to a high percentage of instances being filtered out and exacerbating data sparsity.

5.3 Time Complexity

In terms of computational efficiency on large-scale datasets, despite the two-step detection process and repeatedly calculating Pearson's correlation criterion K times, time complexity varies linearly — $\mathcal{O}(N) \times K$. This is negligible when compared with the training time.

6 Experiments

We first evaluate the noise detection method on two synthetically-corrupted versions of N2C2-STS ("N2C2") (*corrupt2* and *corrupt4*), where we have perfect knowledge of the noisy and clean subset, and then apply our methods to real-world STS, SA, and DA datasets, where noisy labels are unknown.

6.1 Train on Randomly Corrupted Data

Setup We employ the optimal combination of regularisation methods from Section 4 as a strong baseline, namely row 14 of Table 2 (early stopping + dropout + weight decay + pre-fine-tuning), and set $wd = 0.01$, $tolerance = 0.75$, $threshold_r = 0.01$, $K = 5$, and $m = 8$ for noise detection. Other hyper-parameters are as per Section 4. Experimental results are averaged over ten runs based on ten random seeds to account for variance for small test sets.[1]

Result We highlight three findings from Table 3: (1) on randomly corrupted labelled data, noise filtering improves validation performance, particularly for high-degree corruption, decreasing the generalisation error by a large margin; (2) the strategies of DIS and RES perform largely the same, better than REP, so we use DIS in most cases in our following experiments; and (3) precision and recall at noise detection impact on the end-task performance, while the second step of Pearson correlation-based filtering critically improves precision. For example, after the first training epoch on corrupt4 with "discard", precision is 53.82% and recall is 68.15%, and with the Pearson correlation filtering the precision improves to 65.98%, leading to the improvement in row 3.[2]

Discarding automatically-detected noisy labels can maintain improved performance after fast convergence. Considering *corrupt4* in Figure 2, we observe that training with noisy labels for more iterations reduces the validation performance, and the loss also peaks. Through discarding noisy examples by our method during training, the performance can be improved and maintained, and the loss correspondingly drops. This finding is especially vital in the absence of a clean validation set. In such cases, early stopping becomes invalid,

[1] For reproduction purposes, we use seeds = [30 32 40 42 43 45 90 101 3405 3407] for all small test sets.

[2] Precision and recall can be obtained only in the unrealistic scenario of the noisy subset being explicitly known, and are unavailable for real-world data.

	criterion	corrupt2			corrupt4		
		r	ρ	loss	r	ρ	loss
BASE	NA	0.841 ± 0.005	0.815 ± 0.006	0.661 ± 0.048	0.797 ± 0.019	0.785 ± 0.009	0.911 ± 0.102
DIS	prediction	0.844 ± 0.004	0.820 ± 0.006	0.628 ± 0.068	0.808 ± 0.012	0.791 ± 0.011	0.818 ± 0.121
DIS	two-step	**0.847** ± 0.004	**0.824** ± 0.006	**0.586** ± 0.038	**0.822** ± 0.006	**0.803** ± 0.006	0.687 ± 0.095
REP	two-step	0.842 ± 0.005	0.816 ± 0.006	0.655 ± 0.053	0.799 ± 0.018	0.782 ± 0.013	0.861 ± 0.092
RES	two-step	0.843 ± 0.006	0.822 ± 0.007	0.593 ± 0.044	**0.822** ± 0.016	0.798 ± 0.013	**0.653** ± 0.043

Table 3: Results on N2C2 validation set trained on partially corrupted N2C2 train sets: *corrupt2* and *corrupt4* under three noisy label training strategies; "prediction" means we only use the first-step criterion; BASE = baseline.

	ro-en dev			ro-en test			ru-en dev			ru-en test		
	r	ρ	loss	r	ρ	loss	r	ρ	loss	r	ρ	loss
BASE	0.834	0.791	**0.309**	0.832	0.778	**0.290**	0.629	0.612	**0.512**	0.645	0.612	**0.531**
DIS	**0.841**	**0.807**	0.319	**0.838**	**0.793**	0.292	0.640	**0.617**	0.576	0.653	0.627	0.599
RES	0.838	0.801	0.328	0.837	0.789	0.299	**0.644**	0.612	0.538	**0.660**	**0.630**	0.554

Table 4: Results for DA-style quality estimation over the two WMT language pairs.

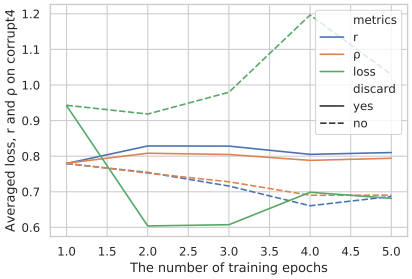

Figure 2: Averaged loss, r, and ρ on N2C2 validation set with (solid line) and without (dotted line) discarding noisy labels using *corrupt4* over the first five epochs.

and we generally train for a constant number of iterations. Identifying and filtering noisy labels can prevent the continuous decline.

6.2 Training on Real-world Noisy Labels

Real-world label noise is a natural outcome of the dataset collection process, and emanates from three primary sources (Algan and Ulusoy, 2021): (1) conflicting opinions of multiple annotators due to diverse interpretations and varying level of expertise, e.g. machine translation quality assessment; (2) inherent uncertainty due to domain complexity such as in the clinical domain; and (3) to collect large amounts of data, textual regression tasks tend to resort to various data augmentation strategies, which is known to result in noisy labels.

6.2.1 Human Disagreement Labels

DA datasets contain examples with highly ambiguous labels due to its subjectivity and the nature of language ambiguity (Wang et al., 2022). Disagreements among annotators very often persist even if more ratings are collected and more context is provided to the raters (Pavlick and Kwiatkowski, 2019). To evaluate the effectiveness of identifying high-disagreement labels, we perform our noise detection method on two DA language pairs: ru-en and ro-en.

Setup: We fine-tune BERT based on *bert-base-multilingual-cased* with maximum sequence length of 128 for five epochs. Hyper-parameter $tolerance$ is set as 1.0 in DIS for both, and 0.75 for ru-en in RES based on the validation set. Other settings are the same as Section 6.1.

Results: In Table 4, over both ru-en and ro-en, employing either discarding or resampling can improve the correlation r/ρ by more than one point on average, indicating that the approach can filter high-ambiguity labels that confuse the model, thus boosting accuracy. It also shows that the lowest loss does not always result in the best performance in terms of r/ρ.

6.2.2 Complicated Domain Labels

In clinical STS, some examples are complicated for clinicians to reach consensus, leading to low inter-annotator agreement, such as Cohen's $\kappa = 0.6$ for N2C2 training labels. In contrast to the disagreements mentioned above, the issue is more varying levels of domain knowledge and the inherent complexity of mapping textual similarity onto a single scalar. Additionally, the N2C2 ground-truth lacks an adjudication process, and gold scores are de-

ID	Train Data	PreF	DIS	r	ρ	loss
1	N2C2 train	NA	NA	0.860 ± 0.008	0.805 ± 0.009	0.759 ± 0.038
2	N2C2 train	stsg	No	0.860 ± 0.003	0.816 ± 0.005	0.746 ± 0.017
3	N2C2 train	stsg	Yes	0.878 ± 0.010	0.810 ± 0.007	$\mathbf{0.585} \pm 0.077$
4	SR train	stsg	No	0.868 ± 0.005	0.816 ± 0.009	0.825 ± 0.035
5	SR train	stsg	Yes	$\mathbf{0.885} \pm 0.010$	$\mathbf{0.824} \pm 0.014$	0.595 ± 0.082

Table 5: Averaged loss, r/ρ on the N2C2 test set **w/o** noise-filtering using N2C2 train and segment-reordered N2C2 train based on CLS-BERT pre-fine-tuned (PreF) on large-scale general STS corpus STS-G.

rived by simple averaging of scores from two annotators (Mahajan et al., 2020). We aim to recognise strong-disagreement labels from the original N2C2 training data and the segment reordered version (see Section 4.2), to improve generalisation.

Setup: We employ a larger-scale general-purpose STS labelled dataset STS-G (Wang et al., 2020c) to learn a general-domain STS model. Other settings are the same as Section 6.1, except for training epoch=3.

Results: Comparing rows 1 and 2 in Table 5, pre-fine-tuning on STS-G doesn't improve performance, and just diminishes the standard deviation. By applying noise-filtering, the loss decreases and correlation improves appreciably (row 3). We speculate that STS-G is large enough to capture general STS task properties, so multiple iterations of training on the N2C2-STS training set don't lead to any gains, but stabilise at a local minimum. Noisy label training helps escape the local minimum by filtering suspected examples, bringing about the large drop in loss, and boosting performance. Fine-tuning over segment-reordered training data augments corruption, making noisy labels more noticeable and easier to detect, thus resulting in the best r/ρ (row 5).

Analysis: Can our method recognise high-disagreement labels? Is detection more accurate with segment-reordered text? As noisy labels are unknown, and individual labels from different annotators are not available either, we manually analysed the first 400 instances in the training set, finding 44 labels that we don't agree with, 16 of which overlap with the detected noisy labels with N2C2 original training data after the first epoch (prec.=0.43, recall=0.36). In the setting of SR, we observed the overlap with our annotations increases to 24 (prec.=0.43, recall=0.55), so the final result improves with an increase in detection accuracy.

6.2.3 Data-Augmented Labels

We apply our method to denoising instances generated through data augmentation, either through synthetic generation based on annotation guidelines, or through rule-based conversion.

Clinical STS with Synthetic Labels Wang et al. (2020b) show that a hierarchical convolutional network based on BERT ("HConvBERT"), in which the BERT-base bottom eight encoder layers are frozen and only parameters of the top four layers are updated, is beneficial to training over a small-scale data set. Since the size of clinical STS datasets is small, we experiment with HConvBERT (hconv) in addition to CLS-BERT (cls).

Synthetic Data Generation: To generate clinical sentences, following Wang et al. (2020c), we sample discharge summaries from MIMIC-III (Johnson et al., 2016) and segment them into 27 parts based on section subtitles (topics). We select the topics of medications, illnesses, diagnoses, and follow-up instructions at which the clinical proxy is not expert, and split into sentences. As medication-related examples emphasise specific medication names, different rules are applied than for other topics. All medication sentences are grouped by medication name; this is always their first word. If S1 and S2 are sampled from the same name group, a similarity score of 3 is assigned, with +0.5 for every increase of 0.2 in $l/L1$ and $l/L2$, where l is the number of shared tokens between S1 in length of L1 and S2 in L2. Otherwise, it is labelled as 1.

For other topics, when sampling sentences from two different topics, the label is set to the range $[0, 1]$. Sentences sampled from the same topic are theoretically in the range $[1, 5]$, but in practice are generally in the range $[1, 2]$ because high similarity under random pairing is unlikely. To obtain pairs in the range $[2, 4.5]$, we randomly sample two sentences from the same topic, and use one as a "template" (S1). We then randomly replace a sequence of d words in the template with the same position (word index) sequence of the second sentence, forming S2. This pair (S1,S2) is labelled as 4.5 if only 10% words are replaced in S1, i.e. $d/L1$=0.1, and 4, 3.5, 3 and 2 with more words replaced, $d/L1$=0.2, 0.3, 0.4 and 0.5, respectively.

In total, we generate 1534 cases ("syn1534"), including 416 medication cases (200 in range $[1, 2]$ and 416 in range $[3, 5]$) and 1118 cases of other topics (200 in range $[0, 1]$, 200 in range $[1, 2]$ and 718 in range $[2, 5]$).

Train Data	Model	PreF	Dis	r	ρ	loss
N2C2 train	cls	NA	NA	0.860 ± 0.008	0.805 ± 0.009	0.759 ± 0.038
N2C2 train	cls	stsb	NA	0.852 ± 0.006	0.813 ± 0.007	0.792 ± 0.054
+ syn1534	cls	stsb	No	0.854 ± 0.004	0.799 ± 0.004	0.788 ± 0.039
+ syn1534	cls	stsb	Yes	0.868 ± 0.003	0.785 ± 0.005	0.669 ± 0.040
N2C2 train	hconv	stsb	NA	0.872 ± 0.003	$\mathbf{0.828} \pm 0.003$	0.717 ± 0.026
+ syn1534	hconv	stsb	No	0.867 ± 0.002	0.817 ± 0.005	0.760 ± 0.018
+ syn1534	hconv	stsb	Yes	$\mathbf{0.882} \pm 0.003$	0.805 ± 0.007	$\mathbf{0.669} \pm 0.076$
MedSTS train	cls	NA	NA	0.833 ± 0.004	0.764 ± 0.006	0.411 ± 0.024
MedSTS train	hconv	stsb	NA	0.850 ± 0.001	0.784 ± 0.004	0.372 ± 0.012
+ syn700	hconv	stsb	No	0.856 ± 0.002	0.789 ± 0.002	0.360 ± 0.019
+ syn700	hconv	stsb	Yes	$\mathbf{0.858} \pm 0.003$	$\mathbf{0.801} \pm 0.006$	$\mathbf{0.357} \pm 0.026$

Table 6: Averaged loss, r and ρ on the N2C2 test set (upper half) and MedSTS test set (bottom half) **w/o** noise-filtering trained *w/o* synthetic data.

	r	ρ	loss
Paper			
Base	0.664 ± 0.007	0.664 ± 0.006	1.203 ± 0.036
FT	0.674 ± 0.003	0.679 ± 0.003	1.223 ± 0.035
FT+Dis	$\mathbf{0.681} \pm 0.006$	$\mathbf{0.686} \pm 0.005$	$\mathbf{1.192} \pm 0.025$
Movie			
Base	0.791 ± 0.010	0.760 ± 0.006	0.533 ± 0.024
FT	0.810 ± 0.006	0.774 ± 0.005	$\mathbf{0.510} \pm 0.018$
FT+Dis	$\mathbf{0.815} \pm 0.005$	$\mathbf{0.777} \pm 0.006$	0.526 ± 0.032

Table 7: r/ρ on PeerRead (top) and 10Movie (bottom) test set **w/o** noise-filtering using PeerRead+Spanish and 10Movie+IMDB for paper and movie domains based on HConvBERT fine-tuned on Yelp (Base), FT=fine-tune.

The STS model is first fine-tuned on STS-B to capture general-domain effects, then on the N2C2 training set combined with syn1534, and the MedSTS training set combined with a random sub-set of syn1534 of size 700 ("syn700"), to match the size of the MedSTS training set (750).

Results: Table 6 shows that combining synthetically-generated sentence pairs with gold-standard training data can improve performance, and discarding noisy labels results in further gains in accuracy on both N2C2-STS (upper half) and MedSTS (bottom half). We exceed previous state-of-the-art results on MedSTS $r = 0.848$ (Peng et al., 2019) $\rightarrow r = 0.858$. Further, pre-trained HConvBERT performs better than CLS-BERT.

Domain SA rating with Converted Labels Additionally, we evaluate denoising on SA rating. Two small-scale SA datasets in the academic paper and movie domains are augmented through rule-based conversion and machine translation (see Section 3), which inevitably introduce noise into the training sets.

Following the experimental setting of clinical STS with corrupted labels, we first fine-tune the regressor using the large-scale Yelp dataset (5,000 instances) based on HConvBERT, referred to as "HConvYelp", then adapt it to the respective datasets by continuous fine-tuning combined with noise filtering. The results in Table 7 show that employing noise-filtering consistently improves performance for both datasets, particularly for the paper reviews, which are most domain-removed from the product reviews.

6.3 Overall Take-away

Through extensive experiments over randomly-corrupted and real-world noisy labels, we have demonstrated that our denoising method is effective at preventing memorisation, regularising noisy labels and improving generalisation performance on various regression tasks. The limitation of our method is that, while it is effective at detecting extreme outliers, it struggles to detect instances with weak disagreement, due to the fact that Pearson's correlation is stable over distributions with moderate skewness (Chok, 2010). As such, it shows more impressive improvement in knowledge-rich domains like clinical notes and academic papers than general-purpose domains (see Section A.2).

7 Conclusion

Regularisation strategies improve model generalisation performance in a range of contexts, but are not able to effectively address generalisation degradation caused by training with noisy labels. In this paper, we have proposed a noisy label training method for text regression tasks, based on identifying noise through iterative prediction and targeted evaluation criteria, followed by discarding or repairing of noisy labels. Extensive experiments on three rating tasks demonstrate the effectiveness of our approach.

Acknowledgements

We thank the three anonymous reviewers for their helpful comments, and Yimeng Chen and Minghan Wang for various discussions. Yuxia Wang is supported by scholarships from The University of Melbourne and China Scholarship Council (CSC).

References

Reina Akama, Sho Yokoi, Jun Suzuki, and Kentaro Inui. 2020. Filtering noisy dialogue corpora by connectivity and content relatedness. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 941–958, Online. Association for Computational Linguistics.

Görkem Algan and Ilkay Ulusoy. 2021. Image classification with deep learning in the presence of noisy labels: A survey. *Knowledge Based Systems*, 215:106771.

Devansh Arpit, Stanislaw Jastrzebski, Nicolas Ballas, David Krueger, Emmanuel Bengio, Maxinder S. Kanwal, Tegan Maharaj, Asja Fischer, Aaron C. Courville, Yoshua Bengio, and Simon Lacoste-Julien. 2017. A closer look at memorization in deep networks. In *Proceedings of the 34th International Conference on Machine Learning*, pages 233–242. PMLR.

Abdessamad Benlahbib. 2019. 1000 movie reviews (review + attached rating + sentiment polarity) for reputation generation. https://data.mendeley.com/datasets/38j8b6s2mx/1.

David Berthelot, Nicholas Carlini, Ian Goodfellow, Nicolas Papernot, Avital Oliver, and Colin A Raffel. 2019. Mixmatch: A holistic approach to semi-supervised learning. In *Advances in Neural Information Processing Systems*, pages 5049–5059.

Wenbin Cai, Ya Zhang, and Jun Zhou. 2013. Maximizing expected model change for active learning in regression. In *2013 IEEE 13th International Conference on Data Mining*, pages 51–60.

Daniel Cer, Mona Diab, Eneko Agirre, Iñigo Lopez-Gazpio, and Lucia Specia. 2017. SemEval-2017 task 1: Semantic textual similarity multilingual and crosslingual focused evaluation. In *Proceedings of the 11th International Workshop on Semantic Evaluation (SemEval-2017)*, pages 1–14, Vancouver, Canada.

Daoyuan Chen, Yaliang Li, Kai Lei, and Ying Shen. 2020. Relabel the noise: Joint extraction of entities and relations via cooperative multiagents. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 5940–5950, Online. Association for Computational Linguistics.

Nian Shong Chok. 2010. *Pearson's versus Spearman's and Kendall's correlation coefficients for continuous data*. Ph.D. thesis, University of Pittsburgh.

Courtney Corley and Rada Mihalcea. 2005. Measuring the semantic similarity of texts. In *Proceedings of the ACL Workshop on Empirical Modeling of Semantic Equivalence and Entailment*, pages 13–18. Association for Computational Linguistics.

Michael Desmond, Catherine Finegan-Dollak, Jeff Boston, and Matt Arnold. 2020. Label noise in context. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics: System Demonstrations*, pages 157–186, Online. Association for Computational Linguistics.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, Minneapolis, Minnesota.

Dina Elreedy, Amir F. Atiya, and Samir I. Shaheen. 2019. A novel active learning regression framework for balancing the exploration-exploitation trade-off. *Entropy*, 21(7):651.

Ian Goodfellow, Yoshua Bengio, Aaron Courville, and Yoshua Bengio. 2016. *Deep Learning*, volume 1. MIT press Cambridge.

Yvette Graham, Timothy Baldwin, Alistair Moffat, and Justin Zobel. 2017. Can machine translation systems be evaluated by the crowd alone. *Natural Language Engineering*, 23(1):3–30.

Hrayr Harutyunyan, Kyle Reing, Greg Ver Steeg, and Aram Galstyan. 2020. Improving generalization by controlling label-noise information in neural network weights. In *Proceedings of the 37th International Conference on Machine Learning*, pages 4071–4081. PMLR.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. 2016. Deep residual learning for image recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 770–778.

Wei Hu, Zhiyuan Li, and Dingli Yu. 2020. Simple and effective regularization methods for training on noisily labeled data with generalization guarantee. In *International Conference on Learning Representations*.

Mia Hubert, Peter J Rousseeuw, and Stefan Van Aelst. 2008. High-breakdown robust multivariate methods. *Statistical Science*, 23(1).

Alistair EW Johnson, Tom J Pollard, Lu Shen, H Lehman Li-Wei, Mengling Feng, Mohammad Ghassemi, Benjamin Moody, Peter Szolovits, Leo Anthony Celi, and Roger G Mark. 2016. MIMIC-III: a freely accessible critical care database. *Scientific Data*, 3(1):1–9.

Dongyeop Kang, Waleed Ammar, Bhavana Dalvi, Madeleine van Zuylen, Sebastian Kohlmeier, Eduard Hovy, and Roy Schwartz. 2018. A dataset of peer reviews (PeerRead): Collection, insights and NLP applications. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 1647–1661, New Orleans, Louisiana. Association for Computational Linguistics.

Brian Keith, Exequiel Fuentes, and Claudio Meneses. 2017. A hybrid approach for sentiment analysis applied to paper. In *Proceedings of ACM SIGKDD Conference*.

Jisoo Lee and Sae-Young Chung. 2020. Robust training with ensemble consensus. In *International Conference on Learning Representations*.

Mingchen Li, Mahdi Soltanolkotabi, and Samet Oymak. 2020. Gradient descent with early stopping is provably robust to label noise for overparameterized neural networks. In *International Conference on Artificial Intelligence and Statistics*, pages 4313–4324. PMLR.

Yun-Peng Liu, Ning Xu, Yu Zhang, and Xin Geng. 2020. Label distribution for learning with noisy labels. In *Proceedings of the Twenty-Ninth International Joint Conference on Artificial Intelligence, IJCAI 2020*, pages 2568–2574. ijcai.org.

Andrew Maas, Raymond E Daly, Peter T Pham, Dan Huang, Andrew Y Ng, and Christopher Potts. 2011. Learning word vectors for sentiment analysis. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, pages 142–150.

Diwakar Mahajan, Ananya Poddar, Jennifer J Liang, Yen-Ting Lin, John M Prager, Parthasarathy Suryanarayanan, Preethi Raghavan, and Ching-Huei Tsou. 2020. Identification of semantically similar sentences in clinical notes: Iterative intermediate training using multi-task learning. *JMIR Medical Informatics*, 8(11):e22508.

Marco Marelli, Luisa Bentivogli, Marco Baroni, Raffaella Bernardi, Stefano Menini, and Roberto Zamparelli. 2014. SemEval-2014 task 1: Evaluation of compositional distributional semantic models on full sentences through semantic relatedness and textual entailment. In *Proceedings of the 8th International Workshop on Semantic Evaluation (SemEval 2014)*, pages 1–8, Dublin, Ireland. Association for Computational Linguistics.

Nitika Mathur, Timothy Baldwin, and Trevor Cohn. 2020. Tangled up in BLEU: Reevaluating the evaluation of automatic machine translation evaluation metrics. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 4984–4997, Online. Association for Computational Linguistics.

Duc Tam Nguyen, Chaithanya Kumar Mummadi, Thi Phuong Nhung Ngo, Thi Hoai Phuong Nguyen, Laura Beggel, and Thomas Brox. 2020. Self: Learning to filter noisy labels with self-ensembling. In *International Conference on Learning Representations*.

Giorgio Patrini, Alessandro Rozza, Aditya Krishna Menon, Richard Nock, and Lizhen Qu. 2017. Making deep neural networks robust to label noise: A loss correction approach. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1944–1952.

Ellie Pavlick and Tom Kwiatkowski. 2019. Inherent disagreements in human textual inferences. *Transactions of the Association for Computational Linguistics*, 7:677–694.

Yifan Peng, Shankai Yan, and Zhiyong Lu. 2019. Transfer learning in biomedical natural language processing: An evaluation of BERT and ELMo on ten benchmarking datasets. In *Proceedings of the 18th BioNLP Workshop and Shared Task*, pages 58–65, Florence, Italy. Association for Computational Linguistics.

Guillaume A Rousselet and Cyril R Pernet. 2012. Improving standards in brain-behavior correlation analyses. *Frontiers in Human Neuroscience*, 6:119.

Omkar Sabnis. 2018. Yelp review dataset. https://www.kaggle.com/omkarsabnis/yelp-reviews-dataset.

Yanyao Shen and Sujay Sanghavi. 2019. Learning with bad training data via iterative trimmed loss minimization. In *International Conference on Machine Learning*, pages 5739–5748. PMLR.

Yurun Song, Junchen Zhao, and Lucia Specia. 2021. Sentsim: Crosslingual semantic evaluation of machine translation. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 3143–3156.

Lucia Specia, Frédéric Blain, Marina Fomicheva, Erick Fonseca, Vishrav Chaudhary, Francisco Guzmán, and André F. T. Martins. 2020. Findings of the WMT 2020 shared task on quality estimation. In *Proceedings of the Fifth Conference on Machine Translation*, pages 743–764, Online. Association for Computational Linguistics.

Masashi Sugiyama. 2006. Active learning in approximately linear regression based on conditional expectation of generalization error. *Journal of Machine Learning Research*, 7:141–166.

Sainbayar Sukhbaatar, Joan Bruna, Manohar Paluri, Lubomir Bourdev, and Rob Fergus. 2015. Training convolutional networks with noisy labels. In *International Conference on Learning Representations*.

Makoto Takamoto, Yusuke Morishita, and Hitoshi Imaoka. 2020. An efficient method of training small models for regression problems with knowledge distillation. In *3rd IEEE Conference on Multimedia Information Processing and Retrieval*, pages 67–72. IEEE.

Daiki Tanaka, Daiki Ikami, Toshihiko Yamasaki, and Kiyoharu Aizawa. 2018. Joint optimization framework for learning with noisy labels. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 5552–5560.

Yichuan Tang and Chris Eliasmith. 2010. Deep networks for robust visual recognition. In *Proceedings of the 27th International Conference on Machine Learning (ICML-10)*, pages 1055–1062.

Ryutaro Tanno, Ardavan Saeedi, Swami Sankaranarayanan, Daniel C. Alexander, and Nathan Silberman. 2019. Learning from noisy labels by regularized estimation of annotator confusion. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 11244–11253. Computer Vision Foundation / IEEE.

Yanshan Wang, Naveed Afzal, Sunyang Fu, Liwei Wang, Feichen Shen, Majid Rastegar-Mojarad, and Hongfang Liu. 2018. MedSTS: a resource for clinical semantic textual similarity. *Language Resources and Evaluation*, pages 1–16.

Yanshan Wang, Sunyang Fu, Feichen Shen, Sam Henry, Ozlem Uzuner, and Hongfang Liu. 2020a. The 2019 N2C2/OHNLP track on clinical semantic textual similarity: Overview. *JMIR Medical Informatics*, 8(11):e23375.

Yuxia Wang, Daniel Beck, Timothy Baldwin, and Karin Verspoor. 2022. Uncertainty estimation and reduction of pre-trained models for text regression. *Transactions of the Association for Computational Linguistics*, 10:1–17.

Yuxia Wang, Fei Liu, Karin Verspoor, and Timothy Baldwin. 2020b. Evaluating the utility of model configurations and data augmentation on clinical semantic textual similarity. In *Proceedings of the 19th SIGBioMed Workshop on Biomedical Language Processing*, pages 105–111, Online. Association for Computational Linguistics.

Yuxia Wang, Karin Verspoor, and Timothy Baldwin. 2020c. Learning from unlabelled data for clinical semantic textual similarity. In *Proceedings of the 3rd Clinical NLP Workshop*, Online. EMNLP.

Jason Wei and Kai Zou. 2019. EDA: Easy data augmentation techniques for boosting performance on text classification tasks. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 6381–6387, Hong Kong, China.

Rand Wilcox. 2004. Inferences based on a skipped correlation coefficient. *Journal of Applied Statistics*, 31(2):131–143.

Dongrui Wu. 2019. Pool-based sequential active learning for regression. *IEEE Transactions of Neural Networks Learning Systems*, 30(5):1348–1359.

Dongrui Wu and Jian Huang. 2022. Affect estimation in 3d space using multi-task active learning for regression. *IEEE Transactions of Affective Computing*, 13(1):16–27.

Jiangchao Yao, Hao Wu, Ya Zhang, Ivor W. Tsang, and Jun Sun. 2019. Safeguarded dynamic label regression for noisy supervision. In *The Thirty-Third AAAI Conference on Artificial Intelligence*, pages 9103–9110.

Shuquan Ye, Dongdong Chen, Songfang Han, and Jing Liao. 2021. Learning with noisy labels for robust point cloud segmentation. In *2021 IEEE/CVF International Conference on Computer Vision*, pages 6423–6432. IEEE.

Chiyuan Zhang, Samy Bengio, Moritz Hardt, Benjamin Recht, and Oriol Vinyals. 2017. Understanding deep learning requires rethinking generalization. In *International Conference on Learning Representations*.

Hongjing Zhang, S. S. Ravi, and Ian Davidson. 2020. A graph-based approach for active learning in regression. In *Proceedings of the 2020 SIAM International Conference on Data Mining*, pages 280–288. SIAM.

Songzhu Zheng, Pengxiang Wu, Aman Goswami, Mayank Goswami, Dimitris N. Metaxas, and Chao Chen. 2020. Error-bounded correction of noisy labels. In *Proceedings of the 37th International Conference on Machine Learning*, pages 11447–11457. PMLR.

A Appendix

A.1 Collecting Labels for Regression

Unlike computer vision tasks, which can make use of messy user tags or search engines and social media, it's hard to obtain usable continuous labels for semantic understanding tasks. Textual regression tasks tend to resort to data augmentation strategies or synthetic generation approaches to obtain labels.

IMDB Label Conversion: IMDB binary-class labels are converted to a rating score by label assignment rules — a negative label corresponds to random selection from $[1, 1, 5, 2, 2, 5]$ and a positive label from $[3, 3.5, 4, 4.5]$.

PeerRead Label: In the PeerRead training set, the ultimate score for rejection or acceptance of a paper is based on more than ten individual aspects such as originality and clarity, but often fewer than 5 aspect scores are available. Scores from other annotated aspects are averaged to fill in these missing aspects, introducing bias.

A.2 General STS with Heterogeneous Labels

We investigate the performance of our method in combining two heterogeneously-labelled datasets (SICK-R and STS-B), expanding the training set size but introducing noise.

STS-B is labelled in the range $[0, 5]$ in accordance with the standard STS formulation, while SICK-R is annotated in the range $[1, 5]$ with an emphasis on semantic relatedness rather than semantic similarity, leading to label misalignment in both label semantics and range between the two datasets. For example, completely irrelevant cases are scored 1.0 in SICK-R but 0 in STS-B, and for:

> **S1:** *A brown dog is attacking another animal in front of the man in pants.*
> **S2:** *Two dogs are fighting.*

the gold score is 3.5 in SICK-R, but for STS-B the score would be in $[1, 2]$.

Two alternatives exist to incorporate SICK-R into STS-B training: (1) fine-tuning jointly over combined training sets; and (2) fine-tuning first on STS-B, then on SICK-R. We investigate both. Consistent with the findings of Section 4, we pre-fine-tune over "STS-B", with training epochs=3, lr=2e-5, and without early stopping. We experiment with the full SICK-R training set ("SICK-R-full") and a subsample of 3000 instances ("SICK-R-3000").

As presented in Table 8, using noise detection and discarding noisy instances can improve the performance under both joint and separate fine-tuning. Overall, in this scenario, from a strong baseline, the improvement is modest even though the training data volume is doubled in the case of the full SICK-R training set. This is largely because the "clean" data from SICK-R for STS-B purposes is mostly distributed in the range of $[4, 5]$, i.e., highly related pairs are also highly similar in the meaning, but this is generally the range where STS predictions are reliable, based on STS-B training. Put differently, even though the noise filtering method was able to discard noisy labels, it was ineffectual due to a lack of clean instances in the critical range $[2, 4]$ where STS-B models perform poorly (Mahajan et al., 2020).

This provides a valuable insight: it is vital to integrate examples in label ranges where the model is deficient. Further, given our findings, only joint fine-tuning is used in Section 6.

Train Data	Manner	Discard	r	ρ	loss
STS-B	NA	NA	0.900	0.896	0.444
STS-B + SICK-R-3000	joint	No	0.900	0.896	0.440
STS-B + SICK-R-3000	joint	Yes	0.901	0.896	0.438
STS-B + SICK-R-3000	separate	No	0.874	0.881	1.515
STS-B + SICK-R-3000	separate	Yes	0.888	0.890	1.241
STS-B + SICK-R-full	joint	No	0.901	0.897	0.445
STS-B + SICK-R-full	joint	Yes	**0.903**	**0.898**	**0.430**
STS-B + SICK-R-full	separate	No	0.886	0.882	1.402
STS-B + SICK-R-full	separate	Yes	0.889	0.889	1.167

Table 8: Averaged loss, r and ρ on STS-B validation set **w/o** noise discarding combining SICK-R-3000 (sampled 3000 instances from SICK-R train set) or SICK-R-full (full train set) by joint and separate fine-tuning.

Detecting Suicide Risk in Online Counseling Services: A Study in a Low-Resource Language

Amir Bialer[†] **Daniel Izmaylov**[†] **Avi Segal**[†] **Oren Tsur**[†]

{amirbial,zmaylov}@post.bgu.ac.il avisegal@gmail.com orentsur@bgu.ac.il

Yossi Levi-Belz[‡]
yossil@ruppin.ac.il

Kobi Gal[†§]
kobig@bgu.ac.il

[†]Ben-Gurion University of the Negev [‡]Ruppin Academic Center [§]University of Edinburgh

Abstract

With the increased awareness of situations of mental crisis and their societal impact, online services providing emergency support are becoming commonplace in many countries. Computational models, trained on discussions between help-seekers and providers, can support suicide prevention by identifying at-risk individuals. However, the lack of domain-specific models, especially in low-resource languages, poses a significant challenge for the automatic detection of suicide risk. We propose a model that combines pre-trained language models (PLM) with a fixed set of manually crafted (and clinically approved) set of suicidal cues, followed by a two-stage fine-tuning process. Our model achieves 0.91 ROC-AUC and an F2-score of 0.55, significantly outperforming an array of strong baselines even early on in the conversation, which is critical for real-time detection in the field. Moreover, the model performs well across genders and age groups.

1 Introduction

The World Health Organization (WHO) lists suicide as one of the most salient causes of death world-wide, causing more deaths than breast cancer or war (WHO, 2019). With close to 1 million lives taken directly by suicide every year, and over 25 million suicide attempts, suicide incurs a lasting impact on families and communities. Identifying individuals at risk ahead of time and providing them with psychological and medical support is a key step in suicide prevention (Joiner et al., 2018).

In this paper, we tackle the task of detecting Suicide Ideation (SI). Specifically, we aim at the detection of SI of individuals contacting online counseling hot-lines in low-resource and morphologically-rich languages. We focus on anonymous data from online, text-based (chat), support services. Such services are available in many countries, allowing for confidential and immediate help to those in distress, and play a critical role in suicide prevention (Bantilan et al., 2021; Jashinsky et al., 2014; Joiner et al., 2007). Empirical evidence suggests that at-risk individuals seek help in close proximity to actual suicide attempts (Zalsman et al., 2021), thus it is critical to identify suicide risks as early as possible during the session.

There is a growing body of work on assessing suicide risk from English texts (whether in social media posts or from counseling sessions), but there is an acute lack of NLP resources that could be used for detection of suicide risk in other languages (Lee et al., 2020). We directly address this gap by focusing on suicide risk detection from online counseling services in Hebrew, which is a low-resource and morphologically-rich language that challenges traditional NLP tasks (Seker et al., 2021).

Our proposed approach for the SI detection task, called SI-BERT, extends a generic pretrained model with a small set of Out of Vocabulary (OOV) tokens, pretraining the language model on a masked LM task and fine tuning for the SI classification task. We further train a logistic regression model, based on a manually crafted lexicon of suicide ideation terms (vetted by domain experts). We then create an Ensemble model by training SI-BERT together with the lexicon. Both SI-BERT and the Ensemble model outperform alternative approaches ranging from W2V (Word2Vec) embeddings and feed-forward networks to Hebrew psychological lexicon. Additionally, the Ensemble model achieves 82% ROC-AUC even when processing only the first few utterances (20%) of the help-seeker, suggesting it can be used to enhance early detection of suicide risk when deployed in the field. We analyze the approach with respect to different demographics, speaker focus and text truncation, and provide a few examples, qualitatively illustrating the benefits of our model. Our work goes the first step in helping counselors identify and treat at-risk individuals in real time.

Proceedings of the 29th International Conference on Computational Linguistics, pages 4241–4250
October 12–17, 2022.

Paper	*Embedding+Model*	*Language*	*Setting*
(Cheng et al., 2017)	LIWC+SVM	Chinese	social media (Weibo)
(Allen et al., 2019)	LIWC+CNN	English	social media (Reddit)
(Matero et al., 2019)	BERT without Pretraining	English	social media (Reddit)
(Ophir et al., 2020)	ELMO+Questionnaires+ANN	English	social media (Facebook)
(Lee et al., 2020)	W2V+LSTM+Lexicons	Korean	social media (Naver Cafe)
(Bantilan et al., 2021)	TF-IDF+XGBoost	English	phone counseling
(Xu et al., 2021)	Knowledge Graph+W2V+LSTM	Chinese	online counseling

Table 1: Sample of relevant approaches used for suicide risk classification from text.

2 Related Work

Our work extends past approaches to suicide risk detection in texts as well as NLP classification tasks in low-resource languages. For a systematic review of the use of machine learning for suicide risk detection from text we refer the reader to Ji et al. (2020) and to (Bernert et al., 2020). For a comprehensive survey of the application of the BERT architecture in different scenarios we refer the reader to (Rogers et al., 2020). In the remainder of this section we briefly survey recent works in each of these fields.

Detection of suicide risk There are limited works in suicide detection in conversations between help-seekers and counselors. Xu et al. (2021) used a classifier based on a knowledge graph of logical relationships of events related to suicide ideation. They combined this graph with Word2Vec embeddings to detect suicide risk in an online counseling service in Hong Kong. Their model achieved 81.5% ROC-AUC for suicide risk detection. Bantilan et al. (2021) combined TF-IDF embeddings with an XGBoost model in transcribed phone calls from a counseling service in English. Their approach achieved a 73% ROC-AUC performance in the phone call based counseling. None of these approaches addressed early detection, and are outperformed by our own approach in terms of ROC-AUC.

There is a body of work on suicide risk detection for English text from social media posts (Guntuku et al., 2017; De Choudhury et al., 2013; Zirikly et al., 2019). Online counseling chats are quite different from such settings in that they often include complete conversations between help-seekers and counselors, rather than single utterances, and exhibit temporal and mental-state dynamics.

Ophir et al. (2020), used ANNs to predict at-risk individuals from Facebook posts and psychological questionnaires. Matero et al. (2019) achieved top results in a suicide ideation detection task in social media (Zirikly et al., 2019) by adapting a BERT model to process input from Twitter and Reddit posts. They found that at-risk individuals use a distinct vocabulary in comparison to the rest of the population. Lee et al. (2020) tackled suicide ideation detection in social media posts in Korean which they describe as a low resource language. In their work they claim domain lexicons are highly beneficial for the task when available. Their classifying model is based on word embeddings, lexicons, attention, and LSTM. These works inspired our approach to combine expert based lexicons with the language model.

For convenience, the main approaches and the settings on which they were developed are summarized in Table 1.

Using Transformers for low-resource and morphologically rich languages Deep neural architectures, especially the Transformer, require massive corpora for adequate training. These pretrained models are then fine-tuned for specific classification tasks, e.g., (Devlin et al., 2018; Sun et al., 2019; Pierse and Lu, 2020; Gururangan et al., 2020). However, fine tuning is suboptimal in the cases where the domain of the classification task is unique, especially in low resource and morphologically-rich languages (Klein and Tsarfaty, 2020; Seker et al., 2021; Nzeyimana and Rubungo, 2022). Our approach tackles those issues.

Hebrew NLP There is an increasing number of Hebrew tools available for modeling NLP tasks. Shapira et al. (2021) released a Hebrew Psychological Lexicons (HPL) that contains 30% of the terms that exist in LIWC, while also containing unique psychological terms that can help detect psychological aspects such as emotional state. Other tools include generic Hebrew BERT models such as HeBERT (Chriqui and Yahav, 2021) and AlephBERT (Seker et al., 2021). AlephBERT was trained

on a larger dataset and achieved better results than HeBERT, making it our PLM of choice.

3 The Dataset

Sahar (https://sahar.org.il), Hebrew acronym for "Aid and Attention Online", is the leading chatline in Israel, focusing on suicide prevention, and emotional distress relief. Relieving the emotional distress of help-seekers is a crucial step in the process of suicide prevention (Overholser et al., 1997; Surís et al., 1996). The organization handles more than $10,000$ chat sessions a year, and these numbers have increased significantly during the COVID-19 pandemic (Zalsman et al., 2021).

Sahar's counselors are volunteers over 24 years old that completed a special training program by licensed clinicians. They are trained to use a special support language that is based on the conversation dynamics. During shifts, there are also therapists on duty who monitor the conversations and provide professional support if needed to the counselors. At the end of each session, the counselor is asked to summarize the conversation and to complete a short survey. A conversation is defined as exhibiting SI if the counselor answered "Yes" to the question "Did the subject of suicide come up in the conversation?"

The Sahar corpus The Sahar corpus contains 44,506 chat sessions which took place in the span of five years (2017-January 2022). Of these, 17,564 are labeled with a True/False to designate whether the chat exhibited SI. Seventeen percent $(3097/17564)$ of the labeled sessions are flagged with a positive SI label. For the remainder of this paper, the term Sahar dataset will refer to the 17,564 labeled sessions in the corpus. A session includes the utterances generated by the help-seekers and the counselor, delimited by a separating token. Table 2 shows general statistics related to our dataset.

Dataset statistics	
Total Num. Sessions	$44,506$
Num. Labeled Sessions	$17,564$
SI positive label ratio	17%
Mean(Median) number of tokens in a chat	$617(566)$

Table 2: General statistics for Sahar Corpus

Beyond the SI label, counselors are requested to select the prominent topics in each conversation (one to three topics chosen from a predefined list).

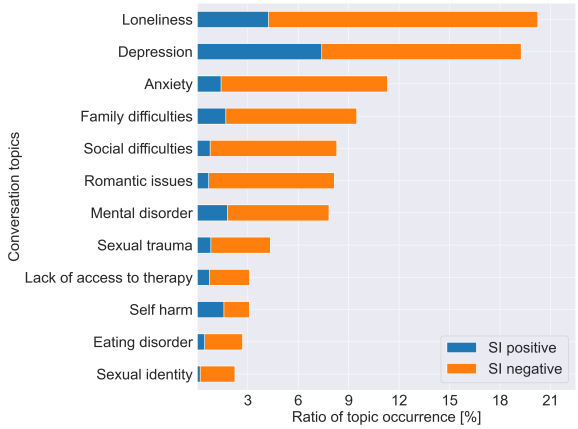

Figure 1: Conversation topics distribution in Sahar corpus

Figure 1 shows the discussed topics distribution in the data. As shown in the figure, loneliness and depression are the most common topics discussed in Sahar platform and a large share of those conversation exhibits positive SI. These findings coincide with psychology literature which depicts depression (Gijzen et al., 2021; Moitra et al., 2021) and loneliness (McClelland et al., 2020) as powerful predictors for suicide ideation. An interesting observation from Figure 1 is, that SI is prevalent across all of the reported topics, and is not exclusive to a certain topic.

Unfortunately, we are not able to share the Sahar corpus due to its sensitive and private content. Nor are we able to share the trained language model, since it was shown language models can be manipulated to reveal training data (e.g (Carlini et al., 2021)). We do provide however a repository with the experiments' code and lexicons used in this paper to support transparency and reproducibility.

4 Computational Approach

Our approach is based on an Ensemble method that extends a generic PLM to the SI domain, and leverages it with a fixed set of manually crafted (and clinically approved) set of suicidal cues. We expand on each component of the Ensemble model in turn.

4.1 The SI-BERT Classifier

SI-BERT is a model designed and configured for SI detection in online Hebrew chats. It utilizes AlephBERT (Seker et al., 2021), which is the best performing Hebrew BERT model to date and is publicly available. SI-BERT augments the generic AlephBERT model by (a) adding domain-specific to-

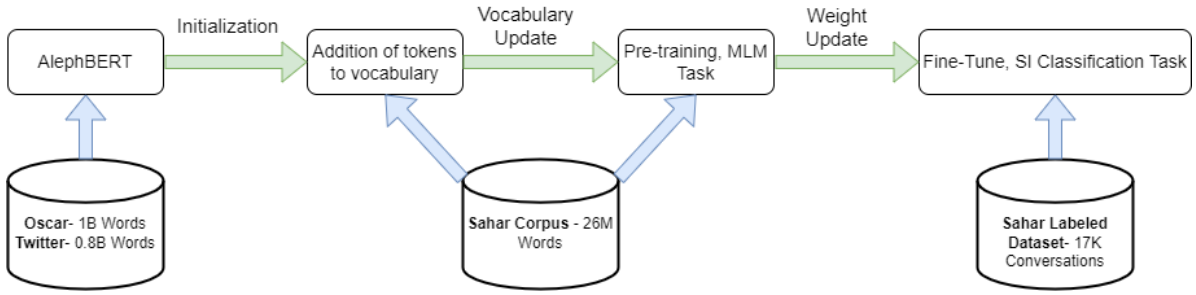

Figure 2: SI-BERT Architecture

kens to the vocabulary; (b) pretraining on a masked language model (MLM) task over the Sahar corpus; and (c) fine-tuning for the SI classification task. The SI-BERT architecture is illustrated in Figure 2. The term 'SI-BERT' at each step refers to the model obtained by the previous step. In the remainder of this section we expand on each of these steps.

Adding Domain Specific Tokens It is well established that enriching the vocabulary of pretrained models with Domain Specific Tokens (DST) improves performance for domain specific tasks (Tai et al., 2020; Beltagy et al., 2019; Honda et al., 2021).

We consider words which are OOV to be domain specific tokens, since by definition they exist in the domain corpus and not in the language model's vocabulary. The language model's vocabulary was constructed by taking the most common tokens from it's training corpus. Manual examination of the most frequent DST finds that many of these tokens are highly related to suicide and mental distress e.g, "suicidal", "desperate", "depressed", "abandoned" (translated from Hebrew). We hypothesize that adding a relatively small number of domain specific tokens will improve performance for suicide ideation classification task.

Therefore, the DST list we used is the δ most frequently appearing words in the Sahar corpus that were not in the vocabulary of the pretrained model. We added this list to SI-BERT's vocabulary, changing it's size from $|V|$ to $|V| + \delta$. We set $\delta = 1000$ based on performance on a held-out validation set. Analyzing the number of OOV token in Sahar corpus, we find 5% of the words to be OOV (before the addition of the DST list). After adding the DST to the vocabulary we observed a decrease of 20% in the number of OOV words in Sahar corpus.

Pretraining with MLM task In this step, we pretrain SI-BERT with a Masked Language Model (MLM) task. MLM is an unsupervised task in which a share of the tokens in each utterance is masked, and SI-BERT is trained to predict the masked words. This task has been shown to improve the performance of BERT and other language models performance for downstream tasks (Gururangan et al., 2020; Pierse and Lu, 2020). The training for the MLM task was conducted for 200 epochs on the complete Sahar corpus. An additional advantage of the MLM task is that it retrains the weights of the SI-BERT following the additions of tokens in the previous step (Tai et al., 2020).

Fine-Tuning In this step we fine-tune SI-BERT for the SI classification task. We add a binary classification head to SI-BERT. The classification head is a neural network layer which consists of two neurons with a softmax activation (binary classification). We compile the model with a cross entropy loss. We fine tune our model with the labeled dataset described in section 3. The BERT model is designed to process 512 tokens. We fine tune the model with the help-seeker's text and used the first 512 tokens of each session as input to the model. For a discussion and justification of these decisions, see subsection 6.4. In practice, 21% of the sessions were truncated when inputted to the model.

4.2 Suicide Ideation Lexicon

We extracted 200 randomly selected positive SI sessions from the Sahar dataset and constructed a list of phrases that explicitly mention suicide ideation. The list contains 67 phrases such as: "suicide", "cut wrists", "want to die" and other variations.

This list was vetted by psychologists with expertise in suicide ideation. The set of 200 positive sessions was removed from the test sets used in the evaluation. Each session is mapped to a vector of length 67, where the ith element in the vector is

the number of occurrences of phrase i in a given session. The vectors are scaled to [0,1] range and fed into a logistic regression model. We publicly share the lexicons in the article's repository.

4.3 Ensemble Model

The Ensemble model combines SI-BERT and the lexicon by feeding their predictions to a fully connected layer activated with a sigmoid function.

5 Evaluation

We compared the Ensemble model to the baseline models described below. The input to all models is a pre-processed chat that concatenates the utterances of the help-seeker and removes non-Hebrew characters and URLs. (See subsection 6.4 for a comparison with the case of also including the utterances of the counselor). For each bag of words based model (W2V, TF-IDF, HPL), the embeddings were scaled to [0,1] range and fed into a logistic regression model.

Fine-tuned AlephBERT (FT-BERT) We used the publicly available model of AlephBERT for text classification and fine tuned it on the labels in the Sahar dataset.

SI-BERT The SI-BERT PLM described in subsection 4.1.

Expert-based SI Lexicon (SI-Lexicon) The expert-based SI lexicon that is described in subsection 4.2.

W2V embeddings + Logistic Regression (W2V-LR) Word To Vector (W2V) is an algorithm which uses a neural network to map each word to a vector with a fixed size (Mikolov et al., 2013). We trained a W2V model on our corpus (embedding dimension=300, as used in the original paper (Mikolov et al., 2013)). The model was used to generate an embedding of all words in the session.

TF-IDF + Logistic Regression (TF-IDF-LR) Term Frequency–Inverse Document Frequency (TF-IDF) is a term weighting scheme commonly used to represent textual documents as vectors (Sammut and Webb, 2010). Each session is vectorized in this manner.

Hebrew Psychological Lexicon + Logistic Regression (HPL-LR) Each session is mapped to a vector of length 276 (number of lexicons in HPL). The ith element in the vector, is the number of occurrences of phrases in lexicon i in a given session.

We focus on two metrics to evaluate a model's performance: (a) ROC-AUC is the most commonly used metric in suicide detection tasks (Bernert et al., 2020). Its main advantage is that it doesn't depend on class distribution in the dataset. (b) The F2 metric computes a weighted harmonic average between the precision and recall scores. It assigns more than twice the weight (compared to the standard F1 metric) to the recall score which is sensitive to false-negative classifications. False negatives are critical in the SI detection task since missing people at risk has life-threatening consequences.

6 Results

In this section we report and discuss results from four perspectives: (i) Entire sessions results, (ii) Early detection, (iii) Results on different demographic groups, and (iv) Using the turn-taking structure vs. focusing on the utterances of the help-seeker. All results are reported using 5-fold cross validation. We keep the label imbalance unchanged in order to increase the potential use of the models for real time SI detection, discussing the False Positive-False Negative trade-off. The input to all models used in subsection 6.1 - subsection 6.3 consists of the concatenated utterances of the help-seeker in each session, while in subsection 6.4 we also provide results for a turn-taking scenario.

6.1 Entire Sessions Results

Table 3 compares the different models using several metrics, including ROC-AUC, and F2, which are commonly used for settings that suffer from high class imbalance (Forman and Scholz, 2010).

As shown in the table, the Ensemble model significantly outperforms the other models in terms of F1, F2 and ROC-AUC metrics. SI-BERT outperforms FT-BERT, and the other baselines. The SI-Lexicon achieves the highest precision, slightly better that the Ensemble, and does very well compared to the non-PLM approaches in the F1 and F2 metrics. However, it achieves only modest recall. This reflects the fact that the lexicon model was manually crafted by mental clinicians and tailored to detect explicit use of suicide ideation. However, there still exist SI positive sessions that cannot be captured by a static list of phrases.

The Ensemble model achieves a significant improvement in recall compared to the models it is

Model	*ROC-AUC[%]*	*F1[%]*	*F2[%]*	*Precision[%]*	*Recall[%]*
W2V-LR	86(0.25)	42(1.35)	33(1.50)	75(1.86)	29(1.50)
TF-IDF-LR	82(0.44)	43(0.30)	34(0.45)	75(1.50)	30(0.50)
HPL-LR	78(0.51)	28(0.94)	21(0.87)	66(2.53)	18(0.81)
SI-Lexicon	82(0.67)	51(0.58)	42(0.59)	**78(1.54)**	38(0.60)
FT-BERT	84(0.47)	47(1.35)	39(1.63)	70(1.61)	42(1.74)
SI-BERT	87(0.37)	55(1.21)	49(1.54)	71(1.52)	45(1.69)
Ensemble	**91(0.45)***	**61(0.89)***	**55(1.32)***	76(2.21)	**51(1.59)***

Table 3: SI classification results listing average performance with standard error in parenthesis. Bold highlights highest value, * marks a model is significantly better than the rest with $p < 0.05$ under Wilcoxon signed rank test.

comprised of, with only a slight decrease in precision performance. We wish to stress that this trade-off is of major significance in this specific domain. To further illustrate this point, Figure 3 presents false-negative ratios (out of test-set size) for the top performing models. As shown in the figure, the Ensemble model achieves the lowest false-negative (8.6%) given that the positive SI samples account for 17% of the test data. Most importantly, the Ensemble model reduces the false negatives ratio from 10.96% to 8.60% (11% decrease). In the field, such an improvement provides a meaningful contribution to suicide prevention, especially in early stages of the session, as we show in the following subsection.

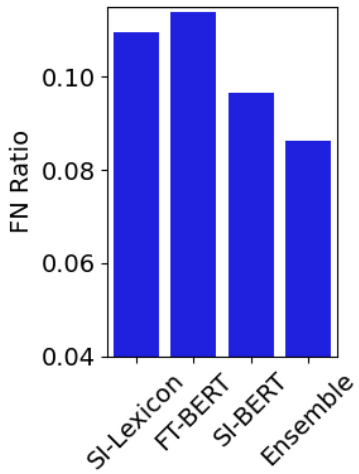

Figure 3: False-negative ratio (out of test-set size) for top-performing SI detection approaches.

6.2 Early Detection

Detecting at-risk individuals as early as possible during the session contributes to suicide prevention and reduces the load on the volunteers. To this end, Figure 4 shows the ROC-AUC performance of the top-performing SI-detection approaches when analyzing the first $\{5, 10, 20, 40, 60, 80, 100\}$ percent of the session (using 5-fold cross validation). As expected, all of the approaches improve as they process more information, with the Ensemble model constantly outperforming all of the other approaches.

Two key findings that stand out from Figure 4 are: (a) There is a consistent gap in performance between SI-BERT and FT-BERT, especially at early stages of the conversation. (b) SI-Lexicon performs poorly at an early stage of the conversation. We hypothesize that help seekers tend to be implicit, before allowing themselves to express their suicidal tendency explicitly[1]. Specifically, we found that while 73% of SI positive sessions contained an explicit SI phrase from the lexicon, only 38% of the SI positive sessions contained an explicit SI phrase in the early 20% of the session. This strengthens our conclusion that the lexicon model is insufficient for early detection of suicide risk.

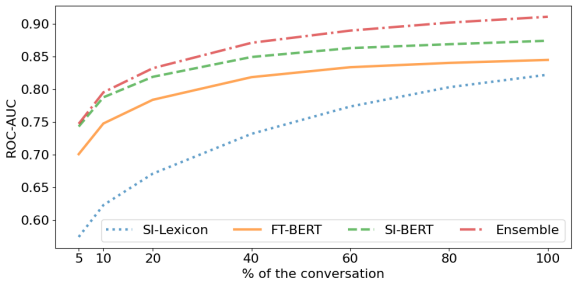

Figure 4: Classification results for early detection of top-performing SI detection approaches.

6.3 Demographic Analysis

Suicide risk, technological proficiency and linguistic norms vary across demographics. Therefore, we evaluate the performance of our model over different demographics.

[1]Furthermore, the explicit expressions could be a response to the counselor sensing implicit cues before directing the conversation a certain way.

Age/Gender	*Samples*	*+ Label*	*ROC-AUC*	*F2*	*Tokens*	*Types*	*OOV Tokens*	*OOV Types*
10-17	$4,179(23\%)$	15%	90	52	1.1M	54K	6%	6%
18-30	$9,066(52\%)$	19%	90	55	2.8M	141K	8%	7%
31-64	$4,164(24\%)$	17%	91	56	1.2M	107K	9%	6%
65+	$145(< 1\%)$	12%	93	58	29K	8K	12%	4%
Female	$12,074$	18	90	55	$3.6M$	$120K$	8%	8%
Male	$5,343$	17	91	55	$1.6M$	$83K$	8%	7%

Table 4: Ensemble model performance evaluation for subgroups of different age and gender.

	10-17	*18-30*	*31-64*	*65+*
10-17	–	0.28	0.39	0.86
18-30	0.60	–	0.55	0.91
31-64	0.54	0.37	–	0.89
65+	0.21	0.14	0.16	–

Table 5: Percentage of the number of unique tokens of each age group with respect to other age groups.

Age The Ensemble model consistently outperforms all other models across all age groups. Results of the Ensemble for different age groups are presented in Table 4 (Top), along with descriptive statistics, highlighting the differences between the sub-corpora in terms of size, types, tokens and OOV types and tokens.

The different linguistic norms each age group exhibits is captured in Table 5 through the relative size of each group's unique vocabulary. For example, while the sub-corpora 10-17 and 31-64 are a similar share of the data (23% & 24%, see Table 4) and a similar label break down (15% & 17%), 39% of the tokens used by the the 10-17 help-seekers are not used by help-seekers 31-64 years of age. Similarly, 54% of the tokens used by the 31-64 group are not used by the individuals on the 10-17 group. These trends are even more pronounced if one considers only OOV types.

Given the large variance in vocabularies between groups, the consistency of our results further demonstrates the robustness of our model.

Gender Examining the two main gender categories[2] we verify that the model achieves similar performance for both genders (see Table 4, Bottom). This result is far from trivial for two reasons: (a) Hebrew is a heavily gendered language (e.g., (Vainapel et al., 2015)). This means most (non-past tense) verbs and adjectives have different morphological inflection depending on the speaker's gender[3], and (b) The number of samples for each gender varies greatly, with female individuals making a vast majority of help seekers.

6.4 Speaker and Text Truncation

One major limitation of the BERT architecture is the constraint it enforces on length of the input. Consequently, most of the sessions cannot be processed fully. A straight forward way to tackle this constraint is to feed the model only part of the session – exhausting the 512-tokens buffer size. We considered three alternative protocols: (i) using utterances of both help-seekers and the counselor, (ii) using only utterances made by the help-seeker, and (iii) using only utterances made by the counselor. The latter protocol is used for comparative reason (also assuming that the responses of the counselor bear relevant signal). In each of these three settings, we considered two options (a) using the first 512 tokens ("keep head"), and (b) using the 512 trailing tokens ("keep tail").

Results of the Ensemble for each of the six settings are presented in Figure 5. The best performance is obtained using the head of the the utterances made by the help-seeker. This result, together with the ability to perform relatively well in early stages (subsection 6.2) are encouraging, given the high stakes of the task and the limited resources (personnel) available to the emergency services.

7 Discussion

The results in the previous section demonstrate the need for adequate adaptation of PLMs into a specific domain with obvious challenges in processing

[2]The gender the help-seeker identifies with is implicitly self disclosed since Hebrew is a gendered language and first-person verbs often takes different suffix according to the speaker gender (see examples in footnote 3).

[3]For example, consider the Hebrew inflections of the adjective 'lonely': 'boded' (M) vs. 'bodeda' (F), or the verb 'going (to)': 'holex' (M) vs. 'holexet' (F).

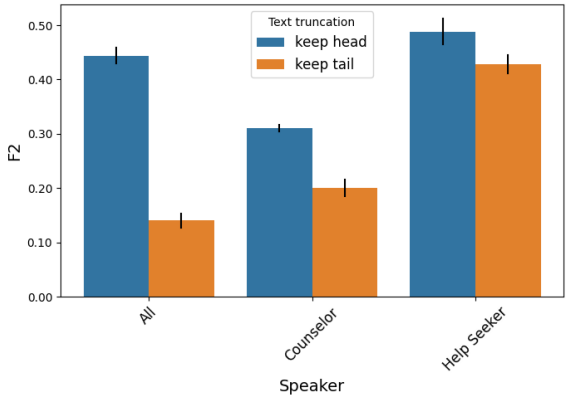

Figure 5: SI-BERT F2 performance for different text truncation methods and speaker text. Error bars mark standard error.

low-resource languages. Domain-tailored lexicons serve as strong baselines, with high precision and competitive F1 and F2 scores but they fall short in terms of recall and AUC. As PLMs tend to capture more nuanced expressions of SI – combining both approaches and careful fine-tuning improved the ability to detect SI early on in the chat.

We conclude this work discussing three illustrative examples (see Table 6) and reflecting on a few limitations of our approach.

I	*I don't want to die*
II	*I feel like life is too much for me*
III	*I had a spare time and bad thoughts, I decided to take a couple of sleeping pills and take a nap. I slept all day and now I'm dizzy.*

Table 6: Three illustrative utterances (translated from Hebrew). Note that these utterance are presented without a conversational context.

In utterance I (Table 6), the speaker explicitly rejects a suicidal intent, however, both the lexicon and SI-BERT (and the Ensemble, of course) classify this utterance as positive ideation. While the lexical approach matches the token *die* and wrongfully ignores the negation, our collaborators – clinical psychologists with suicide detection as their research focus – approve the classification, citing psychological studies on the distinction between suicide ideation and intent (Bagley, 1975; Beck et al., 1979; McAuliffe, 2002).

While it does not match any of the lexical items in the predefined lexicons, the second utterance (Table 6, II) is a classic example of SI. SI-BERT and the Ensemble (as well as the online counselor) correctly label it a positive SI, demonstrating the benefits of the domain-specific contextual model.

The third utterance is not considered by experts to exhibit SI. Indeed, SI-BERT (and the Ensemble) correctly classifies it as a negative example. On the other hand, FT-BERT assigns it a positive label. We hypothesize that the combination of "bad thoughts" and "(a couple of) sleeping pills" triggered the naive FT-BERT, while SI-BERT better captures the nuanced context. This example further demonstrates the benefits of fine-tuning the vanilla AlephBERT not only for the classification task but also fine-tuning the language model on the domain-specific data through a masked LM task.

We end this section briefly mentioning some limitations of the approach. First, our approach does not explicitly account for the discourse structure of the sessions. It may be that encoding the full conversational context may improve performance. Second, the Ensemble approach relies on a handcrafted lexicon requiring considerable human effort. Many psychological lexicons already exist in other languages and have played a considerable role in prior SI research, see (Lee et al., 2020). Investing further effort in lexicon creation may have further reduced the false negative rate. Third, the lack of a benchmark dataset for the SI detection task makes it difficult to compare with prior work and approaches.

8 Conclusion and Future Work

Accurate and early detection of users' suicide risk in text-based counseling services is essential to ensure that at-risk individuals are given timely and proper treatment. This paper provides an automatic approach to risk detection from chats in Hebrew, a low-resource and morpholigically rich language. Our approach adapted a generic Hebrew language model by (i) adding out-of-vocabulary tokens, and (ii) performing additional pre-training of the LM on a masked language modeling task over the specific domain. Finally, we fine-tuned the model for the suicide risk detection task. We combined this model with a lexicon of hand crafted suicide ideation phrases that were vetted by experts. Our Ensemble model outperformed several competitive approaches, including a generic language model and the stand-alone lexicon. Our model performed consistently well for different demographics (age, gender). These encouraging results suggest the model can be deployed successfully, providing the

much needed support to volunteers and health-care professionals in their mission of reducing suicide rates.

In future work, we wish to integrate the discursive structure into the model, and include latent information about the cognitive state of the help-seeker.

References

Kristen Allen, Shrey Bagroy, Alex Davis, and Tamar Krishnamurti. 2019. Convsent at clpsych 2019 task a: Using post-level sentiment features for suicide risk prediction on reddit. In *Proceedings of the Sixth Workshop on Computational Linguistics and Clinical Psychology*, pages 182–187.

Christopher Bagley. 1975. Suicidal behaviour and suicidal ideation in adolescents: A problem for counsellors in education. *British Journal of Guidance and Counselling*, 3(2):190–208.

Niels Bantilan, Matteo Malgaroli, Bonnie Ray, and Thomas D Hull. 2021. Just in time crisis response: suicide alert system for telemedicine psychotherapy settings. *Psychotherapy research*, 31(3):289–299.

Aaron T Beck, Maria Kovacs, and Arlene Weissman. 1979. Assessment of suicidal intention: the scale for suicide ideation. *Journal of consulting and clinical psychology*, 47(2):343.

Iz Beltagy, Kyle Lo, and Arman Cohan. 2019. Scibert: A pretrained language model for scientific text. *arXiv preprint arXiv:1903.10676*.

Rebecca A Bernert, Amanda M Hilberg, Ruth Melia, Jane Paik Kim, Nigam H Shah, and Freddy Abnousi. 2020. Artificial intelligence and suicide prevention: a systematic review of machine learning investigations. *International journal of environmental research and public health*, 17(16):5929.

Nicholas Carlini, Florian Tramer, Eric Wallace, Matthew Jagielski, Ariel Herbert-Voss, Katherine Lee, Adam Roberts, Tom Brown, Dawn Song, Ulfar Erlingsson, et al. 2021. Extracting training data from large language models. In *30th USENIX Security Symposium (USENIX Security 21)*, pages 2633–2650.

Qijin Cheng, Tim MH Li, Chi-Leung Kwok, Tingshao Zhu, and Paul SF Yip. 2017. Assessing suicide risk and emotional distress in chinese social media: a text mining and machine learning study. *Journal of medical internet research*, 19(7):e243.

Avihay Chriqui and Inbal Yahav. 2021. Hebert & hebemo: a hebrew bert model and a tool for polarity analysis and emotion recognition. *arXiv preprint arXiv:2102.01909*.

Munmun De Choudhury, Michael Gamon, Scott Counts, and Eric Horvitz. 2013. Predicting depression via social media. In *Seventh international AAAI conference on weblogs and social media*.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*.

George Forman and Martin Scholz. 2010. Apples-to-apples in cross-validation studies: pitfalls in classifier performance measurement. *Acm Sigkdd Explorations Newsletter*, 12(1):49–57.

Mandy WM Gijzen, Sanne PA Rasing, Daan HM Creemers, Filip Smit, Rutger CME Engels, and Derek De Beurs. 2021. Suicide ideation as a symptom of adolescent depression. a network analysis. *Journal of Affective Disorders*, 278:68–77.

Sharath Chandra Guntuku, David B Yaden, Margaret L Kern, Lyle H Ungar, and Johannes C Eichstaedt. 2017. Detecting depression and mental illness on social media: an integrative review. *Current Opinion in Behavioral Sciences*, 18:43–49.

Suchin Gururangan, Ana Marasović, Swabha Swayamdipta, Kyle Lo, Iz Beltagy, Doug Downey, and Noah A Smith. 2020. Don't stop pretraining: adapt language models to domains and tasks. *arXiv preprint arXiv:2004.10964*.

Ukyo Honda, Yoshitaka Ushiku, Atsushi Hashimoto, Taro Watanabe, and Yuji Matsumoto. 2021. Removing word-level spurious alignment between images and pseudo-captions in unsupervised image captioning. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*, pages 3692–3702, Online. Association for Computational Linguistics.

Jared Jashinsky, Scott H Burton, Carl L Hanson, Josh West, Christophe Giraud-Carrier, Michael D Barnes, and Trenton Argyle. 2014. Tracking suicide risk factors through twitter in the us. *Crisis*.

Shaoxiong Ji, Shirui Pan, Xue Li, Erik Cambria, Guodong Long, and Zi Huang. 2020. Suicidal ideation detection: A review of machine learning methods and applications. *IEEE Transactions on Computational Social Systems*.

Thomas Joiner, John Kalafat, John Draper, Heather Stokes, Marshall Knudson, Alan L Berman, and Richard McKeon. 2007. Establishing standards for the assessment of suicide risk among callers to the national suicide prevention lifeline. *Suicide and Life-Threatening Behavior*, 37(3):353–365.

Thomas E Joiner, Skip Simpson, Megan L Rogers, Ian H Stanley, and Igor I Galynker. 2018. Whether called acute suicidal affective disturbance or suicide crisis syndrome, a suicide-specific diagnosis would enhance clinical care, increase patient safety, and mitigate clinician liability. *Journal of Psychiatric Practice®*, 24(4):274–278.

Stav Klein and Reut Tsarfaty. 2020. Getting the ##life out of living: How adequate are word-pieces for modelling complex morphology? In *Proceedings of the 17th SIGMORPHON Workshop on Computational Research in Phonetics, Phonology, and Morphology*, pages 204–209, Online. Association for Computational Linguistics.

Daeun Lee, Soyoung Park, Jiwon Kang, Daejin Choi, and Jinyoung Han. 2020. Cross-lingual suicidal-oriented word embedding toward suicide prevention. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 2208–2217.

Matthew Matero, Akash Idnani, Youngseo Son, Salvatore Giorgi, Huy Vu, Mohammad Zamani, Parth Limbachiya, Sharath Chandra Guntuku, and H Andrew Schwartz. 2019. Suicide risk assessment with multi-level dual-context language and bert. In *Proceedings of the sixth workshop on computational linguistics and clinical psychology*, pages 39–44.

Carmel M McAuliffe. 2002. Suicidal ideation as an articulation of intent: a focus for suicide prevention? *Archives of Suicide Research*, 6(4):325–338.

Heather McClelland, Jonathan J Evans, Rebecca Nowland, Eamonn Ferguson, and Rory C O'Connor. 2020. Loneliness as a predictor of suicidal ideation and behaviour: a systematic review and meta-analysis of prospective studies. *Journal of affective disorders*, 274:880–896.

Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013. Efficient estimation of word representations in vector space. *arXiv preprint arXiv:1301.3781*.

Modhurima Moitra, Damian Santomauro, Louisa Degenhardt, Pamela Y Collins, Harvey Whiteford, Theo Vos, and Alize Ferrari. 2021. Estimating the risk of suicide associated with mental disorders: A systematic review and meta-regression analysis. *Journal of psychiatric research*, 137:242–249.

Antoine Nzeyimana and Andre Niyongabo Rubungo. 2022. Kinyabert: a morphology-aware kinyarwanda language model. *arXiv preprint arXiv:2203.08459*.

Yaakov Ophir, Refael Tikochinski, Christa SC Asterhan, Itay Sisso, and Roi Reichart. 2020. Deep neural networks detect suicide risk from textual facebook posts. *Scientific reports*, 10(1):1–10.

James C Overholser, Stacy R Freiheit, and Julia M DiFilippo. 1997. Emotional distress and substance abuse as risk factors for suicide attempts. *The Canadian journal of psychiatry*, 42(4):402–408.

Nuo Wang Pierse and Jingwen Lu. 2020. Aligning the pretraining and finetuning objectives of language models. *arXiv preprint arXiv:2002.02000*.

Anna Rogers, Olga Kovaleva, and Anna Rumshisky. 2020. A primer in BERTology: What we know about how BERT works. *Transactions of the Association for Computational Linguistics*, 8:842–866.

Claude Sammut and Geoffrey I Webb. 2010. Tf–idf. *Encyclopedia of machine learning*, pages 986–987.

Amit Seker, Elron Bandel, Dan Bareket, Idan Brusilovsky, Refael Shaked Greenfeld, and Reut Tsarfaty. 2021. Alephbert: A hebrew large pre-trained language model to start-off your hebrew nlp application with. *arXiv preprint arXiv:2104.04052*.

Natalie Shapira, Dana Atzil-Slonim, Daniel Juravski, Moran Baruch, Dana Stolowicz-Melman, Adar Paz, Tal Alfi-Yogev, Roy Azoulay, Adi Singer, Maayan Revivo, et al. 2021. Hebrew psychological lexicons. In *Proceedings of the Seventh Workshop on Computational Linguistics and Clinical Psychology: Improving Access*, pages 55–69.

Chi Sun, Xipeng Qiu, Yige Xu, and Xuanjing Huang. 2019. How to fine-tune bert for text classification? In *China National Conference on Chinese Computational Linguistics*, pages 194–206. Springer.

Joan-Carles Surís, Nuria Parera, and Conxita Puig. 1996. Chronic illness and emotional distress in adolescence. *Journal of adolescent health*, 19(2):153–156.

Wen Tai, HT Kung, Xin Luna Dong, Marcus Comiter, and Chang-Fu Kuo. 2020. exbert: Extending pre-trained models with domain-specific vocabulary under constrained training resources. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 1433–1439.

Sigal Vainapel, Opher Y Shamir, Yulie Tenenbaum, and Gadi Gilam. 2015. The dark side of gendered language: The masculine-generic form as a cause for self-report bias. *Psychological assessment*, 27(4):1513.

WHO. 2019. Suicide in the world: global health estimates. Technical report, World Health Organization.

Zhongzhi Xu, Yucan Xu, Florence Cheung, Mabel Cheng, Daniel Lung, Yik Wa Law, Byron Chiang, Qingpeng Zhang, and Paul SF Yip. 2021. Detecting suicide risk using knowledge-aware natural language processing and counseling service data. *Social Science & Medicine*, 283:114176.

Gil Zalsman, Yael Levy, Eliane Sommerfeld, Avi Segal, Dana Assa, Loona Ben-Dayan, Avi Valevski, and J John Mann. 2021. Suicide-related calls to a national crisis chat hotline service during the covid-19 pandemic and lockdown. *Journal of psychiatric research*.

Ayah Zirikly, Philip Resnik, Ozlem Uzuner, and Kristy Hollingshead. 2019. Clpsych 2019 shared task: Predicting the degree of suicide risk in reddit posts. In *Proceedings of the sixth workshop on computational linguistics and clinical psychology*, pages 24–33.

Does Meta-learning Help mBERT for Few-shot Question Generation in a Cross-lingual Transfer Setting for Indic Languages?

Aniruddha Roy[1], Rupak Kumar Thakur,[2*], Isha Sharma[1], Ashim Gupta[3],
Amrith Krishna[4], Sudeshna Sarkar[1] and Pawan Goyal[1]
[1]IIT Kharagpur, [2]Google India [3]University of Utah, [4]Uniphore
{aniruddha.roy,ishasharma}@iitkgp.ac.in,rupakthakur@google.com
sudeshna@cse.iitkgp.ac.in,pawang@cse.iitkgp.ac.in

Abstract

Few-shot Question Generation (QG) is an important and challenging problem in the Natural Language Generation (NLG) domain. Multilingual BERT (mBERT) has been successfully used in various Natural Language Understanding (NLU) applications. However, the question of how to utilize mBERT for few-shot QG, possibly with cross-lingual transfer, remains. In this paper, we try to explore how mBERT performs in few-shot QG (cross-lingual transfer) and also whether applying meta-learning on mBERT further improves the results. In our setting, we consider mBERT as the base model and fine-tune it using a seq-to-seq language modeling framework in a cross-lingual setting. Further, we apply the model agnostic meta-learning approach to our base model. We evaluate our model for two low-resource Indian languages, Bengali and Telugu, using the TyDi QA dataset. The proposed approach consistently improves the performance of the base model in few-shot settings and even works better than some heavily parameterized models in some settings. Human evaluation also confirms the effectiveness of our approach.

1 Introduction

QG can be defined as the task of generating an appropriate question based on the answer tokens and the context. The previous state-of-the-art QG models are built using neural networks (Du et al., 2017; Zhou et al., 2017; Zhao et al., 2018; Nema et al., 2019), and are trained on high-resource languages with availability of vast amount of manually annotated data for training. Collecting and annotating such vast data for training on low-resource languages can be challenging and costly. Cross-lingual transfer learning has shown its effectiveness in many NLP applications (Kumar et al., 2019; Chi et al., 2019; Asai et al., 2021; Xie et al., 2018) for addressing data scarcity, because it allows us to transmit domain knowledge from a high resource annotated source language to domain of desired target language by fine-tuning with data from a target domain with low resource availability. mBERT (Devlin et al., 2018) has been successfully used in various NLU tasks (Wu and Dredze, 2019; Hu et al., 2020). However, utilizing mBERT for generation tasks with cross-lingual transfer remains unexplored, specifically for QG.

In this paper, we examine the application of mBERT for QG with cross-lingual transfer. Specifically, we ask: 1) Despite the successful usage of various multilingual auto-regressive language models (Xue et al., 2020; Liu et al., 2020; Maurya et al., 2021), can mBERT, an encoder-based model with fewer parameters than these auto-regressive models, be used for QG with cross-lingual transfer? 2) In few-shot cross-lingual transfer settings, fine-tuning may cause colossal distribution gap and severe forgetting (French, 1999), along with an overfitting problem. Can applying meta-learning further improve the results? Meta-learning has shown its effectiveness in various NLP applications such as Dialogue Generation (Qian and Yu, 2019), Machine Translation (Park et al., 2021; Gu et al., 2018), and Natural Language Understanding (Nooralahzadeh et al., 2020; Roy et al., 2022) as it has the capacity to swiftly adapt to unseen training instances while leveraging limited resources, thus it may be helpful in this case as well.

To address these two questions, we use mBERT as the base model, and following (Dong et al., 2019), we fine-tune it as a sequence-to-sequence LM (unidirectional decoding conditioned on bidirectional encoding). We then apply the model agnostic meta-learning approach (Finn et al., 2017) to our base model, and we call our approach meta-QG. The goal of our proposed approach is to determine the best initialization of the model param-

*The author contributed to the paper when he was a student of IIT Kharagpur

Proceedings of the 29th International Conference on Computational Linguistics, pages 4251–4257
October 12–17, 2022.

eters for the QG task, which can help the model to easily adapt to target languages which are low-resource. In our method, there are two phases, i.e., meta-train phase and adaptation phase. The objective of the meta-train phase is to learn an optimal parameter initialization, so we create pseudo QG tasks on the source language. To minimize the language distribution gap between the meta-train and adaption phase, we mix English with an Indian language and consider both as the source languages. During the adaptation phase, we apply the model obtained using meta-train phase on the target language in zero-shot or few-shot settings. For evaluation, we apply our model on two low-resource Indian languages- Telugu and Bengali. We show that our approach gives consistent gains over the base model for Meteor, BLEU-4, and Rouge-L scores. Additionally, we also compare our approach with the heavily parameterized models mt5-base (Xue et al., 2020) (580M) and mBART-50 (Liu et al., 2020) (680M), and the results obtained demonstrate that our approach outperforms mt5-base for both the languages, and performs better than mBART-50 for Bengali in few-shot ($n \leq 16$) settings. Human evaluation also indicates that the proposed approach is very effective.

2 Methodology

QG task is defined as to generate a (syntactically and semantically correct) question based on a paragraph and the relevant sequence of answer tokens present in it. In our cross-lingual transfer setting, we denote the source language labelled training data as D^S_{train} and the target language test data as D^T_{test}. The aim of our QG meta-learning algorithm is to train a model with D^S_{train} using minimum or zero resource of target language labelled data, such that it performs well on D^T_{test}. Our base model in detail, as well as our proposed approach, is described below.

2.1 Base model

For the base model, we make use of multilingual BERT ($mBERT$) and fine-tune it (Dong et al., 2019) as a sequence-to-sequence LM for our QG task. In our work, we consider passage and answer as source segment and question as target segment, and we join these two segment with special tokens $[SEP]$. We randomly mask some tokens in the target sequence and fine-tune the model to recover the masked tokens in a sequence-to-sequence manner. Basically, the model considers partial sentence $y_1 : y_{t-1}$ from the ground truth (bidirectional encoding) to generate the t-th token y_t, which was masked (unidirectional decoding). We use beam search during decoding, taking beam size as 3.

2.2 Applying Meta-learning

Next, we discuss how we apply model-agnostic meta learning (Finn et al., 2017, MAML) for the proposed task. First we take the source languages and, using them, create a set of tasks which we refer to as pseudo-meta-QG tasks. Then, we train the base model on these using pseudo-meta-QG training. Lastly, we adapt the meta-trained model to the test examples of the target language (in zero-shot and few-shot settings). We discuss this in detail below.

Pseudo-meta-QG Tasks creation: We create pseudo-meta-QG tasks (Wu et al., 2020) from the source languages' labeled data. Let us assume that source language's training data, D^S_{train} has P examples denoted as $\{x^{(i)}\}^P_{i=1}$. For each example $x^{(i)}$, a pseudo-meta-QG task τ_i is created in the form of a pseudo train set $D^{\tau_i}_{train}$ and a test set $D^{\tau_i}_{test}$. Here, $D^{\tau_i}_{test} = x^{(i)}$, and $D^{\tau_i}_{train}$ is obtained by retrieving k examples from D^S_{train} which most closely resemble the selected test example. We use the input representation from the base model (mBERT) to calculate (cosine) similarity between any two examples. The pseudo-meta-QG tasks τ_i are defined as follows per training example:

$$\tau_i = (D^{\tau_i}_{train}, D^{\tau_i}_{test}), i \in 1, 2, ..., P. \quad (1)$$

Pseudo-meta-QG training setup: Given the base model M_θ (mBERT) with parameters θ and pseudo-meta-QG tasks $\{\tau_i\}^P_{i=1}$, we obtain θ'_i (one set of parameters per pseudo-meta-QG task τ_i) by doing an inner-update on each τ_i. Specifically, it performs few ($n = 2$) gradient steps on $D^{\tau_i}_{train}$ (pseudo train set), and helps to obtain new model parameters from the base model parameters θ. Our equation for inner-update is as follows:

$$\theta'_i = \theta - lr_{inner} \nabla_\theta \mathcal{L}^{\tau_i}_{D_{\text{train}}}(\theta) \quad (2)$$

Here, θ denotes parameters of the base model M_θ, lr_{inner} is inner learning rate, and $\mathcal{L}^{\tau_i}_{D_{\text{train}}}$ is the loss of pseudo training set $D^{\tau_i}_{train}$ of task τ_i. After the inner-update, a meta-update is performed on the

pseudo test set $D_{test}^{\tau_i}$ of τ_i . This step first calculates the pseudo test loss $\mathcal{L}_{D_{test}^{\tau_i}}$ by evaluation of the modified parameters (θ_i') on $D_{test}^{\tau_i}$. After that, we update the model by optimization of the loss on $D_{test}^{\tau_i}$ in terms of θ. There are multiple iterations involved in this step and the meta-update equation is defined as:

$$\begin{aligned} \theta &\leftarrow \theta - lr_{meta} \sum_i \nabla_\theta \mathcal{L}_{D_{\text{test}}}^{\tau_i}(\theta_i') \\ &= \theta - lr_{meta} \sum_i grad_i \end{aligned} \tag{3}$$

Here lr_{meta} is the learning rate of meta-update and $grad_i$ is the meta-gradient on task τ_i. We can expand it as:

$$grad_i = \nabla_\theta \mathcal{L}_{D_{\text{test}}}^{\tau_i}\left(\theta_i'\right) = \nabla_{\theta_i'} \mathcal{L}_{D_{\text{test}}^{\tau_i}}\left(\theta_i'\right) \nabla_\theta\left(\theta_i'\right) \tag{4}$$

In Equation 4, $\nabla_\theta(\theta_i')$ refers to the Jacobian matrix and it will introduce higher order gradient. Following (Finn et al., 2017; Wu et al., 2020), to reduce the computational cost, we use identity matrix in place of Jacobian matrix. Therefore, $grad_i$ can be computed as:

$$grad_i = \nabla_{\theta_i'} \mathcal{L}_{D_{\text{test}}^{\tau_i}}\left(\theta_i'\right) \tag{5}$$

Finally, we obtain the base model's updated parameters as θ^*.

Adaptation: In the adaptation phase, we apply the source trained model (parameters θ^*) to the target language's test samples in a zero-shot or few-shot setting. We follow Wu et al. (2020)'s adaptation approach for our zero-shot setting. In few-shot setting, we fine-tune the source-trained model on few-shot examples from the training data of target language. Specifically, we subsample the target language training dataset to obtain the small few-shot datasets of size [2,4,8,16]. We randomly sample five datasets for each shot.

3 Experiments

We evaluate our meta-learning based QG model in zero-shot and few-shot settings. This section covers details about the dataset used in our experiments followed by the implementation details with evaluation metrics. **Dataset:** We conduct experiments on low resource Indian languages having minimum amount of annotated data for QG. We use TyDi QA[1] (Clark et al., 2020) Gold passage dataset for our experiments. The dataset contains triplets of passage, question and answer for 9 languages. We evaluate our method on Bengali and Telugu dataset. The sizes of the Bengali and Telugu dataset (train, dev), in terms of number of examples, are (2390, 113), and (5563, 669), respectively. For cross-lingual knowledge transfer, we additionally use English triplets from the same dataset (train = 3696; dev = 440). One should note that since the aforementioned dataset contains no test data, we consider development set as test data for all our experiments. For evaluating Bengali, we consider English and Telugu as the source languages, while we use English and Bengali as the source languages for Telugu. The purpose of mixing one Indian languages is to learn different language distributions rather than single-source distribution. Please note that we follow the same zero-shot and few-shot approach to our base models for fair comparison.

Experiment Setup: We implement our algorithm using PyTorch 1.1.0. Our base model uses BERT base multilingual cased with 12 Transformer blocks, 12 self-attention heads and 768 hidden dimension, GELU activation, and dropout is 0.1. The maximum sequence length is set to 512 for the input. For the creation of pseudo-meta-QG task, we take only two $k = 2$ similar examples during meta training and zero-shot adaptation phase. Each meta-training step performs two inner-update and a meta-update on a batch of 16 tasks. We train our model up to 6,000 meta-training steps. As described in Wu and Dredze (2019), we freeze the embedding and the first three layers of the base transformer model, while the other layers are further fine-tuned for each task. Other hyper-parameter settings are same as in Devlin et al. (2018). We use Adam (Kingma and Ba, 2015) optimizer with learning rates of lr_{inner}, lr_{meta} = $3e{-}5$ for both inner-update and meta-update steps. We set learning rate of lr_{adapt} = $1e{-}5$ for gradient updates during adaptation phase. For few-shot experiment, we fine-tune the meta-trained model up to 60 steps.

We evaluate the systems using BLEU (Papineni et al., 2002), METEOR (Denkowski and Lavie, 2014), and ROUGE-L (Lin, 2004) scores [2]. During the training phase, we train our model using the source language's training data and save the model based on the accuracy of the source lan-

[1] https://github.com/google-research-datasets/tydiqa

[2] We use (Du et al., 2017)'s script for evaluating our model

Model	Setting	Bengali			Telugu		
		BLEU-4	Meteor	Rouge-L	BLEU-4	Meteor	Rouge-L
mt5-base	0-shot	1.38	9.62	7.15	0.00	15.80	11.21
mBART-50		**4.31**	**20.92**	15.87	**3.52**	**27.15**	17.56
mBERT		3.24	16.37	27.88	2.27	17.82	15.03
meta-QG (Ours)		3.99	18.35	**29.45**	1.92	20.19	**20.19**
mt5-base	2-shot	1.73	13.80	9.97	1.15	21.96	12.56
mBART-50		5.01	**27.98**	21.00	**10.02**	**33.52**	**39.21**
mBERT		3.24	16.37	27.88	2.27	17.82	15.03
meta-QG (Ours)		**5.22**	25.45	**33.51**	4.86	31.77	31.83
mt5-base	4-shot	1.71	15.31	10.80	1.59	25.65	14.11
mBART-50		4.71	23.84	19.14	**10.38**	**36.04**	**37.12**
mBERT		3.24	16.37	27.88	2.27	17.82	15.03
meta-QG (Ours)		**5.38**	**26.23**	**34.48**	5.19	34.13	28.54
mt5-base	8-shot	2.95	19.52	13.81	3.88	29.25	19.25
mBART-50		5.01	**27.73**	20.91	**21.02**	**38.88**	**43.74**
mBERT		4.58	22.47	30.74	10.49	32.31	33.05
meta-QG (Ours)		**5.54**	27.40	**32.80**	10.19	36.58	34.57
mt5-base	16-shot	4.85	24.84	17.56	6.23	33.15	26.22
mBART-50		5.67	**27.91**	22.54	**26.46**	**39.44**	**50.72**
mBERT		6.35	23.39	33.99	12.05	32.75	34.94
meta-QG (Ours)		**8.45**	26.77	**37.17**	12.83	35.93	37.78

Table 1: Performance for zero-shot and few-shot cross-lingual question generation for Bengali and Telugu. We consider English and Telugu as source language and evaluate Bengali as target language, while for evaluation of target language Telugu we use English and Bengali as source language. The improvements in BLEU-4 by meta-QG were statistically significant ($p < 0.05$ as per $t-$test) for all settings wrt mt5-base and mBERT and for Bengali 16-shot setting wrt mBART-50.

guage's dev dataset. We carry out the training procedure for four random seeds. For few-shot setting, we randomly sample 5 datasets, and average over 4 training random seeds.

Results: In Table 1, we compare our model to the various base models in zero-shot and few-shot settings to verify the effectiveness of cross-lingual knowledge transfer from source languages to target languages. We see that meta-QG outperforms the base mBERT model for all the settings except Telugu 0-shot BLEU-4 and Telugu 8-shot BLEU-4. Interestingly, it also outperforms the heavily parameterized mt5-base model for all the settings. mBART-50, however, shows its superior quality and outperforms all the other methods for Telugu, except zero-shot Rouge-L, where meta-QG gives better scores. For Bengali, meta-QG still holds an edge over mBART-50, which was quite encouraging. The improvements in BLEU-4 by meta-QG were statistically significant ($p < 0.05$ as per $t-$test) for all settings wrt mt5-base and mBERT and for Bengali 16-shot setting wrt mBART-50 . A detailed error analysis is presented in the Appendix.

Human Evaluation: We also perform human evaluation using a similar procedure as used by (Chi et al., 2019; Maurya et al., 2021). We randomly sample 35 test data-points in both Telugu and Bengali languages and employ three metrics: fluency, relatedness, and correctness. Fluency measure is self-explanatory. The degree to which the generated questions are related to the input context is measured by relatedness, correctness assesses the meaning and semantics of the generated output. While fluency and correctness mainly deal with the generation quality, relatedness is the most critical among these for the task. We present the generated questions by all the competing models (after random shuffling) to three language experts and ask them to rate the questions on a 5-point Likert scale (1: very bad and 5: very good) for all the metrics. The results show that our approach consistently outperforms mBERT and mt5-base for all the metrics. mBART-50 achieves better scores in

Fluency and Correctness due to its superior generation capability. However, meta-QG performs better in relatedness for Bengali, the most critical metric. The final numbers are in Table 2 in the Appendix. These were calculated by averaging all the experts' responses for each parameter.

4 Conclusion

In this work, we make use of mBERT for QG task in few-shot cross-lingual transfer setting, and interestingly, we find that it actually performs better than mt5-base for all the settings, and better than mBART-50 for 16-shot setting in Bengali. We then explore the use of meta-learning with mBERT as the base model (meta-QG) and find that it achieves significant performance improvements compared to the mBERT as well as mt5-base, and surprisingly also outperforms mBART-50 for Bengali. In the future, we plan to extend this framework to other Natural Language Generation tasks, and also plan to study the effectiveness of data augmentation approaches.

Acknowledgement

We thank NLTM, BHASHINI, under the Ministry of Electronics and Information Technology, Govt of India, for their funding and support. We want to thank the human annotators who volunteered to be part of the human evaluation. We also thank Sovan Kumar Sahoo, Souvic Chakraborty, Anurag Roy, Santanu Pal, Ankit Bagde, Kousshik Raj, Nithish Kannen, and Nikhil Reddy, who helped in various aspects to complete the project.

References

Akari Asai, Jungo Kasai, Jonathan Clark, Kenton Lee, Eunsol Choi, and Hannaneh Hajishirzi. 2021. XOR QA: Cross-lingual open-retrieval question answering. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 547–564, Online. Association for Computational Linguistics.

Zewen Chi, Li Dong, Furu Wei, Wenhui Wang, Xian-Ling Mao, and Heyan Huang. 2019. Cross-lingual natural language generation via pre-training.

Jonathan H. Clark, Eunsol Choi, Michael Collins, Dan Garrette, Tom Kwiatkowski, Vitaly Nikolaev, and Jennimaria Palomaki. 2020. Tydi qa: A benchmark for information-seeking question answering in typologically diverse languages. *Transactions of the Association for Computational Linguistics*.

Michael Denkowski and Alon Lavie. 2014. Meteor universal: Language specific translation evaluation for any target language. In *Proceedings of the Ninth Workshop on Statistical Machine Translation*, pages 376–380, Baltimore, Maryland, USA. Association for Computational Linguistics.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*.

Li Dong, Nan Yang, Wenhui Wang, Furu Wei, Xiaodong Liu, Yu Wang, Jianfeng Gao, Ming Zhou, and Hsiao-Wuen Hon. 2019. Unified language model pre-training for natural language understanding and generation.

Xinya Du, Junru Shao, and Claire Cardie. 2017. Learning to ask: Neural question generation for reading comprehension. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics, ACL 2017, Vancouver, Canada, July 30 - August 4*, volume 1:, page 13421352.

C. ; Abbeel Finn, P. ;, and S. Levine. 2017. Model-agnostic meta-learning for fast adaptation of deep networks. In *ICML, 11261135*.

Robert French. 1999. Catastrophic forgetting in connectionist networks. *Trends in cognitive sciences*, 3:128–135.

Jiatao Gu, Yong Wang, Yun Chen, Kyunghyun Cho, and Victor O. K. Li. 2018. Meta-learning for low-resource neural machine translation.

Junjie Hu, Sebastian Ruder, Aditya Siddhant, Graham Neubig, Orhan Firat, and Melvin Johnson. 2020. Xtreme: A massively multilingual multi-task benchmark for evaluating cross-lingual generalisation. In *International Conference on Machine Learning*, pages 4411–4421. PMLR.

D. P. Kingma and J. Ba. 2015. Adam: A method for stochastic optimization. In *ICLR*.

Vishwajeet Kumar, Nitish Joshi, Arijit Mukherjee, Ganesh Ramakrishnan, and Preethi Jyothi. 2019. Cross-lingual training for automatic question generation. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 4863–4872, Florence, Italy. Association for Computational Linguistics.

Chin-Yew Lin. 2004. ROUGE: A package for automatic evaluation of summaries. In *Text Summarization Branches Out*, pages 74–81, Barcelona, Spain. Association for Computational Linguistics.

Yinhan Liu, Jiatao Gu, Naman Goyal, Xian Li, Sergey Edunov, Marjan Ghazvininejad, Mike Lewis, and Luke Zettlemoyer. 2020. Multilingual denoising pre-training for neural machine translation. *Transactions of the Association for Computational Linguistics*, 8:726–742.

Kaushal Kumar Maurya, Maunendra Sankar Desarkar, Yoshinobu Kano, and Kumari Deepshikha. 2021. Zmbart: An unsupervised cross-lingual transfer framework for language generation.

Preksha Nema, Akash Kumar Mohankumar, Mitesh M. Khapra, Balaji Vasan Srinivasan, and Balaraman Ravindran. 2019. Let's ask again: Refine network for automatic question generation.

Farhad Nooralahzadeh, Giannis Bekoulis, Johannes Bjerva, and Isabelle Augenstein. 2020. Zero-shot cross-lingual transfer with meta learning.

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. Bleu: a method for automatic evaluation of machine translation. In *Proceedings of the 40th annual meeting of the Association for Computational Linguistics*, pages 311–318.

Cheonbok Park, Yunwon Tae, Taehee Kim, Soyoung Yang, Mohammad Azam Khan, Eunjeong Park, and Jaegul Choo. 2021. Unsupervised neural machine translation for low-resource domains via meta-learning.

Kun Qian and Zhou Yu. 2019. Domain adaptive dialog generation via meta learning. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 2639–2649, Florence, Italy. Association for Computational Linguistics.

Aniruddha Roy, Isha Sharma, Sudeshna Sarkar, and Pawan Goyal. 2022. Meta-ed: Cross-lingual event detection using meta-learning for indian languages. *ACM Trans. Asian Low-Resour. Lang. Inf. Process.*

Qianhui Wu, Zijia Lin, Guoxin Wang, Hui Chen, Börje F. Karlsson, Biqing Huang, and Chin-Yew Lin. 2020. Enhanced meta-learning for cross-lingual named entity recognition with minimal resources.

S. Wu and M. Dredze. 2019. *Beto, bentz, becas: The surprising cross-lingual effectiveness of BERT*. CoRRabs/1904.09077.

Jiateng Xie, Zhilin Yang, Graham Neubig, Noah A. Smith, and Jaime Carbonell. 2018. Neural cross-lingual named entity recognition with minimal resources.

Linting Xue, Noah Constant, Adam Roberts, Mihir Kale, Rami Al-Rfou, Aditya Siddhant, Aditya Barua, and Colin Raffel. 2020. mt5: A massively multilingual pre-trained text-to-text transformer. *arXiv preprint arXiv:2010.11934*.

Yao Zhao, Xiaochuan Ni, Yuanyuan Ding, and Qifa Ke. 2018. Paragraph-level neural question generation with maxout pointer and gated self-attention networks. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 3901–3910, Brussels, Belgium. Association for Computational Linguistics.

Qingyu Zhou, Nan Yang, Furu Wei, Chuanqi Tan, Hangbo Bao, and Ming Zhou. 2017. Neural question generation from text: A preliminary study.

A Appendix

A.1 Human Evaluation Results: Complete Table

Model	Flu.		Rel.		Cor.	
	bn	te	bn	te	bn	te
mt5-base	3.17	2.76	2.35	2.80	3.28	2.65
mBART-50	**4.42**	**4.32**	2.69	**3.23**	**4.29**	**3.91**
mBERT	3.01	3.40	2.17	2.49	2.74	3.08
meta-QG	3.49	3.49	**2.96**	2.85	3.79	3.51

Table 2: Human evaluation results of 16-shot cross-lingual question generation for Bengali and Telugu. The three metrics are Fluency (Flu.), Relatedness (Rel.), and Correctness (Cor.) respectively.

A.2 Case Study

Table 3 shows few example sentences with the corresponding questions generated by the base mBERT model as well as the proposed meta-QG approach. For the examples 3a and 3d, we find that mBERT does not generate a question where entity names are getting repeated, possibly due to some bias towards generating entity names from the reference context. However, meta-QG overcomes this issue and generates better questions. The questions generated by mBERT in 3b and 3c are better than the other two examples, but there are minor issues, such as 'Surya Sen' instead of 'Surya Sen's' (3b: missing morphological marker in Bengali) and only the surname (3c).

Reference 3a (Bengali): চিত্রা বন্দ্যোপাধ্যায়ের স্বামীর নাম কী ? **Translation**: What is the name of Chitra Bandyopadhyay's husband? **meta-QG output**: চিত্রা বন্দ্যোপাধ্যায়ের স্বামীর নাম কী ? **Translation**: What is the name of Chitra Bandyopadhyay's husband? **mBERT output**: চিত্রা বা চিত্রা বা চিত্রা ছিলেন ? **Translation**: Was it Chitra or Chitra or Chitra?
Reference 3b (Bengali): মাস্টারদা সূর্যকুমার সেনের বাবার নাম কী ছিল ? **Translation**: What was the name of Masterda Suryakumar Sen's father? **meta-QG output**: সূর্য সেনের বাবার নাম কী ? **Translation**: What is the name of Surya Sen's father? **mBERT output**: সূর্য সেন বাবার নাম কী ? **Translation**: What is the name of Surya Sen father?
Reference 3c (Bengali): বিখ্যাত জ্যোতির্বিজ্ঞানী নিকোলাউস কোপের্নিকুসের জন্ম কবে হয় ? **Translation**:When was the famous astronomer Nicolaus Copernicus born? **meta-QG output**: নিকোলাস কোপারনিকাস জন্ম কবে ? **Translation:** When was Nicholas Copernicus born? **mBERT output** : কোপারনিকাসের জন্ম কবে ? **Translation:** When was Copernicus born?
Reference 3d (Bengali): বিখ্যাত বাংলাদেশী চলচ্চিত্র পরিচালক মোরশেদুল ইসলামের প্রথম পরিচালিত চলচ্চিত্রের নাম কী ? **Translation:** What is the name of the first film directed by famous Bangladeshi film director Morshedul Islam? **meta-QG output**: মোরশেদুল ইসলামের প্রথম চলচ্চিত্রের নাম কী ? **Translation:** What is the name of the first film of Morshedul Islam? **mBERT output** : মোরশেদুল ইসলাম বা মোরশেদুল ইসলাম বা মোরশেদ কে ছিলেন ? **Translation:** Who was Morshedul Islam or Morshedul Islam or Morshed?

Table 3: Some example outputs by the base mBERT model as well as the proposed meta-QG approach

Revisiting Syllables in Language Modelling and their Application on Low-Resource Machine Translation

Arturo Oncevay[ϵ,χ] **Kervy Dante Rivas Rojas**[ρ,α]
Liz Karen Chavez Sanchez[χ] **Roberto Zariquiey**[ρ,χ]
[ϵ]School of Informatics, University of Edinburgh, Scotland
[ρ]Pontificia Universidad Católica del Perú ([α]IA-PUCP | [χ]Chana Field Station), Peru
a.oncevay@ed.ac.uk,rzariquiey@pucp.edu.pe

Abstract

Language modelling and machine translation tasks mostly use subword or character inputs, but syllables are seldom used. Syllables provide shorter sequences than characters, require less-specialised extracting rules than morphemes, and their segmentation is not impacted by the corpus size. In this study, we first explore the potential of syllables for open-vocabulary language modelling in 21 languages. We use rule-based syllabification methods for six languages and address the rest with hyphenation, which works as a syllabification proxy. With a comparable perplexity, we show that syllables outperform characters and other subwords. Moreover, we study the importance of syllables on neural machine translation for a non-related and low-resource language-pair (Spanish–Shipibo-Konibo). In pairwise and multilingual systems, syllables outperform unsupervised subwords, and further morphological segmentation methods, when translating into a highly synthetic language with a transparent orthography (Shipibo-Konibo). Finally, we perform some human evaluation, and discuss limitations and opportunities.

1 Introduction

In language modelling (LM), we learn distributions over sequences of words, subwords or characters, and the last two can allow an open-vocabulary generation (Sutskever et al., 2011). We rely on subword segmentation as a widespread approach to generate rare subword units (Sennrich et al., 2016). However, the lack of a representative corpus, in terms of the word vocabulary, can constrain the unsupervised segmentation (e.g. with scarce monolingual texts (Joshi et al., 2020)). As an alternative, we could use character-level modelling, since it also has access to subword information (Kim et al., 2016), but we face long-term dependency issues and require longer training time to converge. Similar issues are extended to other generation tasks, such as machine translation (MT).

In this context, we focus on syllables, which are speech units: "A syl-la-ble con-tains a sin-gle vowel u-nit". syllables can be defined as a group of segments that is pronounced as a single articulatory movement. Syllables are fundamental phonological units since they participate in important word prosodic patterns, such as stress assignment. In this sense, syllables are more linguistically relevant units than characters, and behave as a mapping function to reduce the length of the sequence with a larger "alphabet" or syllabary. Their extraction can be rule-based and corpus-independent, but data-driven methods or hyphenation using dictionaries can approximate them as well.

We assess whether syllables are useful for encoding and/or decoding a diverse set of languages on two generation tasks. First, for LM, we study 21 languages, to cover different levels of orthographic depth, which is the degree of grapheme-phoneme correspondence (Borgwaldt et al., 2005) and a factor that can increase complexity to syllabification (Marjou, 2021).[1] Whereas for MT, we focus on the distant and low-resource language-pair of Spanish–Shipibo-Konibo. We choose Shipibo-Konibo[2] because it is an endangered language with scarce textual corpora, which limits unsupervised segmentation methods, and has a transparent orthography, which could be beneficial to syllabification. Also, we consider multilingual MT systems, as they outperformed pairwise systems for the chosen language pair (Mager et al., 2021).

2 Related work

The closest LM study to ours is from Mikolov et al. (2012) for subword-grained prediction in English, where they used syllables as a proxy to split words with low frequency, reduce the vocabulary and compress the model size. Besides, syllable-aware LM

[1]E.g., English has a deep orthography (weak correspondence), whereas Finnish is transparent (Ziegler et al., 2010).

[2]See Appendix A for more details about the language.

Proceedings of the 29th International Conference on Computational Linguistics, pages 4258–4267
October 12–17, 2022.

was addressed by Assylbekov et al. (2017) for English, German, French, Czech, Spanish and Russian, and by Yu et al. (2017) for Korean. However, in both cases, the syllables were composed with convolutional filters into word-level representations for closed-vocabulary generation. Besides, for subword-aware open-vocabulary LM, Blevins and Zettlemoyer (2019) incorporated morphological supervision with a multi-task objective.

For syllable-based MT, there are mostly studies for related paired languages, such as Indic languages (in statistical MT without subword-based baselines: Kunchukuttan and Bhattacharyya (2016)), Tibetan–Chinese (Lai et al., 2018), and Myanmar–Rakhine (Myint Oo et al., 2019). Instead, Spanish–Shipibo-Konibo is a non-related language-pair. The only distant pair was English–Myanmar (ShweSin et al., 2019), but they did not compare it with unsupervised subwords. Neither of these studies analysed multilingual settings.

3 Open-vocabulary language modelling with a comparable perplexity

Open-vocabulary output We generate the same input unit (e.g. characters, syllables or other subwords) as an open-vocabulary LM task, where there is no prediction of an "unknown" or out-of-vocabulary word-level token (Sutskever et al., 2011). We thereby differ from previous works, and refrain from composing the syllable representations into words to evaluate only word-level perplexity.

Character-level perplexity For a fair comparison across all granularities, we evaluate all results with character-level perplexity:

$$\text{ppl}^c = \exp\left(\mathcal{L}_{\text{LM}}(\mathbf{s}) \cdot \frac{|\mathbf{s}^{\text{seg}}| + 1}{|\mathbf{s}^c| + 1}\right) \qquad (1)$$

where $\mathcal{L}_{\text{LM}}(\mathbf{s})$ is the cross entropy of a string $\mathbf{s}$ computed by the neural LM, and $|\mathbf{s}^{\text{seg}}|$ and $|\mathbf{s}^c|$ refer to the length of $\mathbf{s}$ in the chosen segmentation and character-level units, respectively (Mielke, 2019). The extra unit considers the end of the sequence.

3.1 Experimental setup

Languages and datasets Corpora are listed in Table 4 in Appendix B. We first choose WikiText-2-raw (en_w; Merity et al., 2016), which contains around two million word-level tokens extracted from Wikipedia articles in English. Furthermore, we employ 20 Universal Dependencies (UD; Nivre et al., 2020) treebanks, similarly to Blevins and Zettlemoyer (2019).[3] Finally, we include the Shipibo-Konibo (shp) side of the parallel corpora provided by the AmericasNLP shared task on MT (Mager et al., 2021), which is also used in §4.

Syllable segmentation (SYL) For splitting syllables in different languages, we used rule-based syllabification tools for English, Spanish, Russian, Finnish, Turkish and Shipibo-Konibo, and dictionary-based hyphenation tools for all languages except the ones mentioned above. We list the tools in Appendix C.

Segmentation baselines Besides characters (CHAR) and the annotated morphemes in the UD treebanks (MORPH), we consider Polyglot (POLY)[4], which includes models for unsupervised morpheme segmentation trained with Morfessor (Virpioja et al., 2013). Moreover, we employ an unsupervised subword segmentation baseline of Byte Pair Encoding (BPE; Sennrich et al., 2016)[5] with different vocabulary sizes from 2,500 to 10,000 tokens, with 2,500 steps. We also fix the parameter to the syllabary size. Appendix C includes details about the segmentation format.

Model and training Following other open-vocabulary LM studies (Mielke and Eisner, 2019; Mielke et al., 2019), we use a low-compute version of an LSTM neural network, named Average SGD Weight-Dropped (Merity et al., 2018). See the hyperparameter details in Appendix E.

3.2 Results and discussion

Table 1 shows the ppl^c values for the different levels of segmentation we considered in the study, where we did not tune the neural LM model for a specific segmentation. We observe that syllables always result in better perplexities than other granularities, even for deep orthography languages such as English or French. The results obtained by the BPE baselines are relatively poor as well, and they could not beat characters in any dataset, even though we searched for an optimal vocabulary size for the BPE algorithm. The advantage of using syllables is that we do not need to tune a

[3] The languages are chosen given the availability of an open-source syllabification or hyphenation tool. We prefer to use the UD treebanks, instead of other well-known datasets for language modelling (e.g. Multilingual Wikipedia Corpus (Kawakami et al., 2017)), because they provide morphological annotations, which are fundamental for this study.

[4] polyglot-nlp.com

[5] We use: https://github.com/huggingface/tokenizers

	CHAR	MORPH	POLY	SYL	BPE_{best}
en_w*	2.48 ±0.0	-	2.8 ±0.0	**1.96** ±0.0	2.91 ±0.0
bg	3.56 ±0.03	4.09 ±0.05	4.69 ±0.01	**2.87** ±0.0	5.19 ±0.01
ca	2.84 ±0.0	3.11 ±0.02	3.26 ±0.01	**2.21** ±0.0	3.31 ±0.0
cs	3.32 ±0.0	3.11 ±0.01	4.18 ±0.01	**2.66** ±0.0	4.24 ±0.0
da	4.25 ±0.01	4.42 ±0.04	5.6 ±0.0	**3.1** ±0.01	6.21 ±0.03
de	3.5 ±0.04	3.36 ±0.08	3.79 ±0.0	**2.48** ±0.0	3.86 ±0.02
en*	4.11 ±0.01	4.39 ±0.08	5.67 ±0.01	**2.82** ±0.07	5.65 ±0.04
es*	3.16 ±0.01	3.71 ±0.04	3.95 ±0.01	**2.51** ±0.0	3.98 ±0.0
fi*	3.77 ±0.01	4.05 ±0.12	4.76 ±0.01	**3.1** ±0.0	5.27 ±0.01
fr	3.09 ±0.01	3.67 ±0.02	3.82 ±0.01	**2.3** ±0.01	3.87 ±0.01
hr	3.52 ±0.02	3.92 ±0.01	4.34 ±0.0	**2.8** ±0.0	4.52 ±0.02
it	2.8 ±0.0	3.19 ±0.0	3.43 ±0.01	**2.27** ±0.01	3.61 ±0.0
lv	4.55 ±0.02	5.31 ±0.0	6.82 ±0.02	**3.59** ±0.0	7.19 ±0.0
nl	3.83 ±0.05	3.69 ±0.1	4.44 ±0.01	**2.76** ±0.01	4.83 ±0.01
pl	4.03 ±0.01	4.77 ±0.22	5.96 ±0.04	**3.19** ±0.0	5.99 ±0.0
pt	3.31 ±0.01	3.46 ±0.03	4.03 ±0.01	**2.56** ±0.0	4.24 ±0.01
ro	3.4 ±0.02	3.89 ±0.04	4.25 ±0.01	**2.72** ±0.0	4.71 ±0.01
ru*	3.28 ±0.0	2.93 ±0.01	4.05 ±0.0	**2.69** ±0.01	4.04 ±0.0
sk	6.16 ±0.05	5.1 ±0.07	7.61 ±0.08	**4.62** ±0.01	10.51 ±0.03
tr*	4.16 ±0.05	4.86 ±0.05	6.41 ±0.07	**3.66** ±0.03	6.98 ±0.1
uk	4.92 ±0.02	6.45 ±0.11	8.11 ±0.03	**4.24** ±0.02	9.23 ±0.02
shp*	4.48±0.01	-	-	**2.15**±0.02	3.50±0.03

Table 1: Character-level perplexity (↓) in test. We show the mean and standard deviation for three runs with different seeds. BPE shows the best result from models with different vocabulary sizes. SYL presents the syllabification-based result if it is available (*), or the hyphenation otherwise.

hyper-parameter to extract a different set of subword pieces.

As a significant outcome, we note that syllables did not fail to beat characters, at least in an open-vocabulary LM task, which extends the findings of Assylbekov et al. (2017). One potential reason is that character-level modelling requires a larger model capacity due to the longer sequences, however, that is also an advantage towards syllables. Besides, other subword pieces with a closer sequence length to syllables (BPE, MORPH or POLY) were still outperformed.

Finally, in Appendix D, we further discuss the relationship between the syllable type/token ratio with the word vocabulary growth and perplexity.

4 Low-resource Machine Translation

After observing the positive impact on LM, we focus on syllables for MT, which adds complexity to the process, as there is at least one extra language involved. In contrast to prior work, we (i) study syllable-based MT for a distant and low-resource language-pair, Spanish–Shipibo-Konibo; (ii) compare syllables against the most widespread unsupervised segmentation method (BPE) with automatic metrics and human evaluation; and (iii) analyse the applicability of syllables on multilingual translation systems. The last element is significant, as a multilingual setting is the state-of-the-art approach for leveraging low-resource language-pairs performance (Siddhant et al., 2022). Moreover, we decided to apply syllabification only on Shipibo-Konibo, a highly synthetic[6] language with scarce textual data and with a transparent orthography[7].

For this reason, we focus in three settings. First, MONO, a pairwise system where each source and target is segmented with a different method. Second, JOINT, another pairwise system where the BPE baseline is jointly trained with the source and target data. Third, O2M, a multilingual one-to-many[8] system, where the BPE baseline is jointly trained with all the languages (we added Spanish–English in our experiments).

4.1 Experimental setup

Data For Spanish–Shipibo-Konibo (es–shp), we use the dataset provided by the AmericasNLP workshop (Mager et al. (2021); Galarreta et al. (2017); Gómez Montoya et al. (2019)), and perform the same split as Mager et al. (2022) for the dev and test subsets, to make the results comparable to their morphological segmentation experiments. For the multilingual case, we use the Spanish–English (es–en) train set from EuroParl (Koehn, 2005) and newscommentary-v8, and the NEWSTEST2013.ES-EN (Bojar et al., 2013) evaluation sets.

Segmentation (i) BPE (Sennrich et al., 2016) is our baseline segmentation method, and we use the implementation of SentencePiece (Kudo and Richardson, 2018). Similar to Mager et al. (2022), we fix the best vocabulary size at 5000 pieces for the MONO setting, after trying different values from 1k to 10k. JOINT and O2M use 5000 and 16000 pieces, respectively.

(ii) Syllabification (SYL) for Shipibo-Konibo is adapted from Alva and Oncevay (2017). The original method uses syllables to verify whether a word is composed by consistent syllables for spell-checking. In our experiments, when a word can not be syllabify-ed, we split it into characters for the

[6] With a high ratio of number of morphemes per word.

[7] We attempted to use syllables on Spanish and English as well, but with negative results. With large data, unsupervised segmentation methods like BPE can obtain more significant and overlapping subwords from source and target.

[8] We do not consider the many-to-one direction due to resource constraints, and because we observed that the improvements by syllables are noted when decoding Shipibo-Konibo.

MONO setting, and we use the joint-BPE segmentation model for the JOINT and O2M settings.

Model and training We reproduce Mager et al. (2022)'s settings, by using the fairseq toolkit (Ott et al., 2019), and a Transfomer model (Vaswani et al., 2017) with smaller dimensions (Guzmán et al., 2019). For the multilingual O2M setting, we use a sampling approach with 5 of temperature (Aharoni et al., 2019). See details in Appendix E.

Evaluation We use chrF (Popović, 2015) from SACREBLEU (Post, 2018)[9] and also perform a human evaluation of 100 samples per system (BPE and Syl), following the annotation protocol used in the AmericasNLP shared task (Mager et al., 2021).

4.2 Results and discussion

Table 2 shows the translation performance in all settings, and we observe that syllables are statistically better than the BPE baseline when translating from Spanish into Shipibo-Konibo, but not in the other direction. This fact indicates that syllables support the decoding more than the encoding step of a language with a transparent orthography. Also, the JOINT setting reduces the gap between BPE and SYL, probably due to the shared roots between the two languages (i.e., loanwords from Spanish into Shipibo-Konibo). Furthermore, we note that the impact of syllables is not minimised in a multilingual system (O2M), where the performance for es→shp has drastically improved, and the other language-pair (es→en) retains a comparable result.

Moreover, our MONO experiments are comparable with the study of Mager et al. (2022), where they tested several unsupervised and supervised morphological segmentation methods against BPE for MT in four polysynthetic languages (including Shipibo-Konibo). Our result with syllables in es→shp outperforms all other approaches, such as LMVR (Ataman et al., 2017), with a 38.99 chrF score. This indicates that syllables are a robust alternative to morphologically-aware methods when we are dealing with limited data and translating into a polysynthetic language.

4.3 Human evaluation

We also conducted a small human evaluation of system outputs using a 5-points scale for the adequacy and fluency of the Spanish→Shipibo-Konibo translation, which is the translation direction that benefited from the syllable segmentation. The annotation protocol and annotator's information is provided in Appendix F.

Figure 1 shows the scores annotated for adequacy and fluency, where we compare BPE and SYL in the O2M setting, which obtained the best performance for both segmentation methods. We observe that the adequacy is very poor for both systems (1-2), but there is an advantage for SYL in the smaller batch of highest adequacy (5), with 3% more of the total samples. Regarding fluency, both systems mostly obtain a low score (2), but there is a consistent advantage for SYL over BPE in the highest value (5), with 6.5% more of the total samples. The differences are very small to determine whether a segmentation works better than the other from human judgement, but they are consistent with the automatic evaluation provided previously. A larger sample, an extra annotator, or more robust

	BPE	SYL	BPE	SYL
	es→shp		**es→en**	
MONO	37.62±1.87	**41.27***±0.54		
JOINT	40.41±0.82	**41.74***±0.95		
O2M	48.30	**51.25***	53.99	53.85
	shp→es			
MONO	**33.37**±0.79	32.85*±1.22		
JOINT	**34.55**±0.56	33.13*±0.75		

Table 2: chrF scores in the test subsets. MONO: single BPE model (5k pieces) for each source and target. JOINT: joint BPE model (5k) for both source and target. O2M: joint BPE model (16k) for ES, EN and SHP. For the first two settings, we run three experiments and present the mean and standard deviation. The latter only has one run due to resource constraints, and we report es–en scores as a reference. Syllabification (SYL) is only applied on the Shipibo-Konibo side, and (*) indicates a p-value ≤ 0.05 against the BPE baseline.

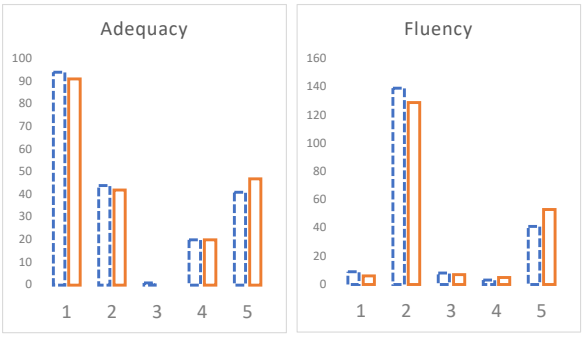

Figure 1: Adequacy and fluency scores (1-5) for 200 outputs of two approaches: BPE (dashed blue) and SYL (solid orange), from the best es→shp O2M system.

[9] chrF2+numchars.6+space.false+v.1.5.0.

systems could aid to clarify other potential benefits.

5 Limitations and opportunities

Syllables only cannot offer a universal solution to the subword segmentation problem for all languages, as the syllabification tools are language-dependent. Besides, the analysis should be extended to more scripts and morphological types. Furthermore, we do not encode any semantics in the syllable-vector space, with a few exceptions like in Korean (Choi et al., 2017). Nevertheless, our results confirm that syllables are reliable for LM and MT, and building a syllable splitter might require less effort than annotating morphemes to train a robust supervised tool[10].

Specifically for MT, syllables could be useful when: (i) we are dealing with extremely low-resource data, which affects unsupervised word segmentation, (ii) we are translating into a language with a high synthesis, which has been observed as a factor that impacts on MT performance (Oncevay et al., 2022), and (iii) we are working with a language with a transparent orthography. This is the scenario for several languages from the Americas, where their writing systems have been recently standardised for documentation and revitalisation purposes (Mager et al., 2018), and some resources for MT have been compiled (Mager et al., 2021).

6 Conclusion

We have proved that syllables are valuable for generation tasks such as: (i) Open-vocabulary LM, where they behave positively even for languages with deep orthography, and overcome character and subword baselines. (ii) Low-resource and multilingual MT, outperforming BPE pieces when we translate into a language with a transparent orthography and complex morphology (high synthesis), even when the language-pair is not related.

Acknowledgements

The first author acknowledges the support of NVIDIA Corporation with the donation of a Titan Xp GPU used for the study. The last author acknowledges the Max Planck Institute for Evolutionary Anthropology, Department of Linguistic and Cultural Evolution, for its support to the development of the Chana Field Station in the Amazonian region of Peru, and the support of CONCYTEC-ProCiencia, Peru, under the contract 183-2018-FONDECYT-BM-IADT-MU from the funding call E041-2018-01-BM.

[10]For instance, the syllabification tool that we used for English is based on five general rules from: https://www.howmanysyllables.com/divideintosyllables. Their implementation should take less effort than annotating a UD treebank or building a Finite-State-Transducer for morphological analysis.

References

Roee Aharoni, Melvin Johnson, and Orhan Firat. 2019. Massively multilingual neural machine translation. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 3874–3884, Minneapolis, Minnesota. Association for Computational Linguistics.

Carlo Alva and Arturo Oncevay. 2017. Spell-checking based on syllabification and character-level graphs for a Peruvian agglutinative language. In *Proceedings of the First Workshop on Subword and Character Level Models in NLP*, pages 109–116, Copenhagen, Denmark. Association for Computational Linguistics.

Zhenisbek Assylbekov, Rustem Takhanov, Bagdat Myrzakhmetov, and Jonathan N. Washington. 2017. Syllable-aware neural language models: A failure to beat character-aware ones. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 1866–1872, Copenhagen, Denmark. Association for Computational Linguistics.

Duygu Ataman, Matteo Negri, Marco Turchi, and Marcello Federico. 2017. Linguistically motivated vocabulary reduction for neural machine translation from turkish to english. *arXiv preprint arXiv:1707.09879*.

Terra Blevins and Luke Zettlemoyer. 2019. Better character language modeling through morphology. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 1606–1613, Florence, Italy. Association for Computational Linguistics.

Ondřej Bojar, Christian Buck, Chris Callison-Burch, Christian Federmann, Barry Haddow, Philipp Koehn, Christof Monz, Matt Post, Radu Soricut, and Lucia Specia. 2013. Findings of the 2013 Workshop on Statistical Machine Translation. In *Proceedings of the Eighth Workshop on Statistical Machine Translation*, pages 1–44, Sofia, Bulgaria. Association for Computational Linguistics.

Susanne R Borgwaldt, Frauke M Hellwig, and Annette MB De Groot. 2005. Onset entropy matters–letter-to-phoneme mappings in seven languages. *Reading and Writing*, 18(3):211–229.

Elisabeth Borleffs, Ben AM Maassen, Heikki Lyytinen, and Frans Zwarts. 2017. Measuring orthographic

transparency and morphological-syllabic complexity in alphabetic orthographies: a narrative review. *Reading and writing*, 30(8):1617–1638.

Sanghyuk Choi, Taeuk Kim, Jinseok Seol, and Sanggoo Lee. 2017. A syllable-based technique for word embeddings of Korean words. In *Proceedings of the First Workshop on Subword and Character Level Models in NLP*, pages 36–40, Copenhagen, Denmark. Association for Computational Linguistics.

Ana-Paula Galarreta, Andrés Melgar, and Arturo Oncevay. 2017. Corpus creation and initial SMT experiments between Spanish and Shipibo-konibo. In *Proceedings of the International Conference Recent Advances in Natural Language Processing, RANLP 2017*, pages 238–244, Varna, Bulgaria. INCOMA Ltd.

Héctor Erasmo Gómez Montoya, Kervy Dante Rivas Rojas, and Arturo Oncevay. 2019. A continuous improvement framework of machine translation for Shipibo-konibo. In *Proceedings of the 2nd Workshop on Technologies for MT of Low Resource Languages*, pages 17–23, Dublin, Ireland. European Association for Machine Translation.

Francisco Guzmán, Peng-Jen Chen, Myle Ott, Juan Pino, Guillaume Lample, Philipp Koehn, Vishrav Chaudhary, and Marc'Aurelio Ranzato. 2019. The FLORES evaluation datasets for low-resource machine translation: Nepali–English and Sinhala–English. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 6098–6111, Hong Kong, China. Association for Computational Linguistics.

Pratik Joshi, Sebastin Santy, Amar Budhiraja, Kalika Bali, and Monojit Choudhury. 2020. The state and fate of linguistic diversity and inclusion in the NLP world. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 6282–6293, Online. Association for Computational Linguistics.

Kazuya Kawakami, Chris Dyer, and Phil Blunsom. 2017. Learning to create and reuse words in open-vocabulary neural language modeling. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1492–1502, Vancouver, Canada. Association for Computational Linguistics.

Yoon Kim, Yacine Jernite, David Sontag, and Alexander M. Rush. 2016. Character-aware neural language models. In *Proceedings of the Thirtieth AAAI Conference on Artificial Intelligence*, AAAI'16, pages 2741–2749. AAAI Press.

Diederik P. Kingma and Jimmy Ba. 2015. Adam: A method for stochastic optimization. In *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*.

Philipp Koehn. 2005. Europarl: A parallel corpus for statistical machine translation. In *Proceedings of Machine Translation Summit X: Papers*, pages 79–86, Phuket, Thailand.

Taku Kudo and John Richardson. 2018. SentencePiece: A simple and language independent subword tokenizer and detokenizer for neural text processing. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 66–71, Brussels, Belgium. Association for Computational Linguistics.

Anoop Kunchukuttan and Pushpak Bhattacharyya. 2016. Orthographic syllable as basic unit for SMT between related languages. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 1912–1917, Austin, Texas. Association for Computational Linguistics.

Wen Lai, Xiaobing Zhao, and Wei Bao. 2018. Tibetan-Chinese neural machine translation based on syllable segmentation. In *Proceedings of the AMTA 2018 Workshop on Technologies for MT of Low Resource Languages (LoResMT 2018)*, pages 21–29, Boston, MA. Association for Machine Translation in the Americas.

Manuel Mager, Ximena Gutierrez-Vasques, Gerardo Sierra, and Ivan Meza-Ruiz. 2018. Challenges of language technologies for the indigenous languages of the Americas. In *Proceedings of the 27th International Conference on Computational Linguistics*, pages 55–69, Santa Fe, New Mexico, USA. Association for Computational Linguistics.

Manuel Mager, Arturo Oncevay, Abteen Ebrahimi, John Ortega, Annette Rios, Angela Fan, Ximena Gutierrez-Vasques, Luis Chiruzzo, Gustavo Giménez-Lugo, Ricardo Ramos, Ivan Vladimir Meza Ruiz, Rolando Coto-Solano, Alexis Palmer, Elisabeth Mager-Hois, Vishrav Chaudhary, Graham Neubig, Ngoc Thang Vu, and Katharina Kann. 2021. Findings of the AmericasNLP 2021 shared task on open machine translation for indigenous languages of the Americas. In *Proceedings of the First Workshop on Natural Language Processing for Indigenous Languages of the Americas*, pages 202–217, Online. Association for Computational Linguistics.

Manuel Mager, Arturo Oncevay, Elisabeth Mager, Katharina Kann, and Thang Vu. 2022. BPE vs. morphological segmentation: A case study on machine translation of four polysynthetic languages. In *Findings of the Association for Computational Linguistics: ACL 2022*, pages 961–971, Dublin, Ireland. Association for Computational Linguistics.

Xavier Marjou. 2021. OTEANN: Estimating the transparency of orthographies with an artificial neural network. In *Proceedings of the Third Workshop on Computational Typology and Multilingual NLP*, pages 1–9, Online. Association for Computational Linguistics.

Stephen Merity, Nitish Shirish Keskar, and Richard Socher. 2018. Regularizing and optimizing LSTM language models. In *International Conference on Learning Representations*.

Stephen Merity, Caiming Xiong, James Bradbury, and Richard Socher. 2016. Pointer sentinel mixture models. *arXiv preprint arXiv:1609.07843*.

Sabrina J. Mielke. 2019. Can you compare perplexity across different segmentations? Available in: http://sjmielke.com/comparing-perplexities.htm.

Sabrina J. Mielke, Ryan Cotterell, Kyle Gorman, Brian Roark, and Jason Eisner. 2019. What kind of language is hard to language-model? In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 4975–4989, Florence, Italy. Association for Computational Linguistics.

Sabrina J. Mielke and Jason Eisner. 2019. Spell once, summon anywhere: A two-level open-vocabulary language model. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, pages 6843–6850.

Tomáš Mikolov, Ilya Sutskever, Anoop Deoras, Hai-Son Le, Stefan Kombrink, and Jan Cernocky. 2012. Subword language modeling with neural networks. *preprint (http://www. fit. vutbr. cz/imikolov/rnnlm/char. pdf)*, 8:67.

Thazin Myint Oo, Ye Kyaw Thu, and Khin Mar Soe. 2019. Neural machine translation between Myanmar (Burmese) and Rakhine (Arakanese). In *Proceedings of the Sixth Workshop on NLP for Similar Languages, Varieties and Dialects*, pages 80–88, Ann Arbor, Michigan. Association for Computational Linguistics.

Joakim Nivre, Marie-Catherine de Marneffe, Filip Ginter, Jan Hajič, Christopher D. Manning, Sampo Pyysalo, Sebastian Schuster, Francis Tyers, and Daniel Zeman. 2020. Universal Dependencies v2: An evergrowing multilingual treebank collection. In *Proceedings of the 12th Language Resources and Evaluation Conference*, pages 4034–4043, Marseille, France. European Language Resources Association.

Arturo Oncevay. 2021. Peru is multilingual, its machine translation should be too? In *Proceedings of the First Workshop on Natural Language Processing for Indigenous Languages of the Americas*, pages 194–201, Online. Association for Computational Linguistics.

Arturo Oncevay, Duygu Ataman, Niels Van Berkel, Barry Haddow, Alexandra Birch, and Johannes Bjerva. 2022. Quantifying synthesis and fusion and their impact on machine translation. In *Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 1308–1321, Seattle, United States. Association for Computational Linguistics.

Myle Ott, Sergey Edunov, Alexei Baevski, Angela Fan, Sam Gross, Nathan Ng, David Grangier, and Michael Auli. 2019. fairseq: A fast, extensible toolkit for sequence modeling. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics (Demonstrations)*, pages 48–53, Minneapolis, Minnesota. Association for Computational Linguistics.

Maja Popović. 2015. chrF: character n-gram F-score for automatic MT evaluation. In *Proceedings of the Tenth Workshop on Statistical Machine Translation*, pages 392–395, Lisbon, Portugal. Association for Computational Linguistics.

Matt Post. 2018. A call for clarity in reporting BLEU scores. In *Proceedings of the Third Conference on Machine Translation: Research Papers*, pages 186–191, Brussels, Belgium. Association for Computational Linguistics.

Rico Sennrich, Barry Haddow, and Alexandra Birch. 2016. Neural machine translation of rare words with subword units. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1715–1725, Berlin, Germany. Association for Computational Linguistics.

Yimon ShweSin, Win Pa Pa, and KhinMar Soe. 2019. UCSYNLP-lab machine translation systems for WAT 2019. In *Proceedings of the 6th Workshop on Asian Translation*, pages 195–199, Hong Kong, China. Association for Computational Linguistics.

Aditya Siddhant, Ankur Bapna, Orhan Firat, Yuan Cao, Mia Xu Chen, Isaac Caswell, and Xavier Garcia. 2022. Towards the next 1000 languages in multilingual machine translation: Exploring the synergy between supervised and self-supervised learning.

Ilya Sutskever, James Martens, and Geoffrey Hinton. 2011. Generating text with recurrent neural networks. In *Proceedings of the 28th International Conference on International Conference on Machine Learning*, ICML'11, pages 1017–1024, USA. Omnipress.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Ł ukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, editors, *Advances in Neural Information Processing Systems 30*, pages 5998–6008. Curran Associates, Inc.

Sami Virpioja, Peter Smit, Stig-Arne Grönroos, and Mikko Kurimo. 2013. Morfessor 2.0: Python implementation and extensions for morfessor baseline. In *Aalto University publication series*. Department of Signal Processing and Acoustics, Aalto University.

Seunghak Yu, Nilesh Kulkarni, Haejun Lee, and Jihie Kim. 2017. Syllable-level neural language model for agglutinative language. In *Proceedings of the*

First Workshop on Subword and Character Level Models in NLP, pages 92–96, Copenhagen, Denmark. Association for Computational Linguistics.

Roberto Zariquiey, Harald Hammarström, Mónica Arakaki, Arturo Oncevay, John Miller, Aracelli García, and Adriano Ingunza. 2019. Obsolescencia lingüística, descripción gramatical y documentación de lenguas en el perú: hacia un estado de la cuestión. *Lexis*, 43(2):271–337.

Johannes C Ziegler, Daisy Bertrand, Dénes Tóth, Valéria Csépe, Alexandra Reis, Luís Faísca, Nina Saine, Heikki Lyytinen, Anniek Vaessen, and Leo Blomert. 2010. Orthographic depth and its impact on universal predictors of reading: A cross-language investigation. *Psychological science*, 21(4):551–559.

A The Shipibo-Konibo language

Shipibo-Konibo (shp) is the largest and most vital language within the Pano language family. With more than 30,000 speakers, the Shipibo-Konibo are among the largest indigenous groups in Peru. Shipibo-Konibo people mainly live in the Peruvian Amazonia (in the regions of Ucayali, Loreto, Huánuco and Madre de Dios), but there are also large groups of Shipibo-Konibo people living in the Peruvian coast (particularly in Lima and Ica).

Shipibo-Konibo is a well-documented language, although the publicly available data on this language is rather small (Zariquiey et al., 2019). It has a complex morphology due to its high synthesis (high ratio of morphemes per word, mostly by suffixation) and it agglutinative nature. Its orthography can be considered transparent, because its alphabet was recently standardised by the Peruvian Government (Alva and Oncevay, 2017), and the datasets we are using in all experiments are provided with the most recent writing standard (Mager et al., 2021). Machine translation research on Shipibo-Konibo has focused on the development of new parallel corpora (Galarreta et al., 2017; Gómez Montoya et al., 2019), the application of multilingual models (Oncevay, 2021), or the impact of morphological segmentation methods (Mager et al., 2022). However, neither of them has focused on syllables as a unit for segmentation. For this study, we adapt the syllabification function proposed by Alva and Oncevay (2017), which was used for spell-checking.

B Dataset details

Table 4 shows the size of the training, validation and test splits for all the datasets used in the LM task, while Table 3 shows details of the Spanish–Shipibo-Konibo and Spanish–English parallel corpora used in the MT task.

	train	dev	test
es–shp	13,102	587	1,030
es–en	2,140,175	5,003	3,000

Table 3: Total number of sentences in train, dev and test splits for the language-pairs used in the MT experiments.

C Segmentation details

Tools We list the tools for rule-based syllabification and dictionary-based hyphenation:

	Train			Valid			Test		
	Word	Syl	Char	Word	Syl	Char	Word	Syl	Char
en_w	2,089	4,894	10,902	218	505	1,157	246	568	1,304
bg	125	386	710	16	50	92	16	49	90
ca	436	1,123	2,341	59	152	317	61	157	327
cs	1,158	3,546	6,868	157	482	933	172	524	1,012
da	81	215	442	10	28	57	10	27	56
de	260	735	1,637	12	34	75	16	45	102
en	210	488	1,061	26	61	133	26	61	132
es	376	1,060	2,043	37	103	198	12	33	64
fi	165	595	1,224	19	67	137	21	76	155
fr	360	837	1,959	36	84	197	10	23	54
hr	154	484	930	20	62	119	23	75	145
it	263	762	1,504	11	32	64	10	28	57
lv	113	349	690	19	58	115	20	59	116
nl	187	488	1,074	12	30	66	11	31	68
pl	102	293	589	13	37	73	13	37	74
pt	192	551	1,040	10	29	54	9	27	51
ro	183	549	1,056	17	51	98	16	48	94
ru	867	2,707	5,411	118	364	722	117	360	717
sk	80	232	437	12	39	76	13	41	80
tk	38	126	242	10	33	63	10	33	64
uk	88	289	501	12	41	71	16	56	99
shp	43	141	398						

Table 4: Total number of tokens (in thousands) at word, syllable and character-level for all the splits.

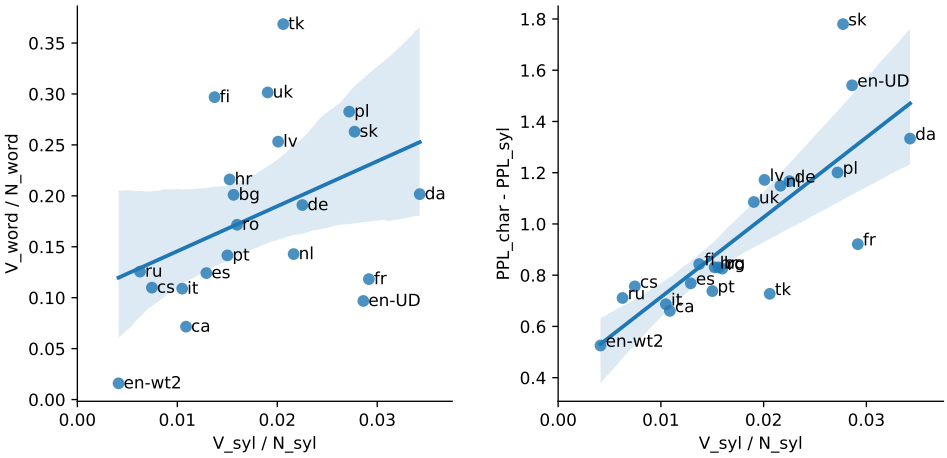

(a) V_{syl}/N_{syl} vs. V_{word}/N_{word} (b) V_{syl}/N_{syl} vs. $\Delta\,ppl^{c}{}_{char\text{-}syl}$

Figure 2: Left (a): Vocabulary growth rate of syllables (x-axis) versus words (y-axis). Right (b): Vocabulary growth rate of syllables (x-axis) versus the difference of ppl^{c} obtained by characters and syllables (y-axis).

- English syllabification: Extracted from https://www.howmanysyllables.com/
- Spanish syllabification: https://pypi.org/project/pylabeador/
- Russian syllabification: https://github.com/Koziev/rusyllab
- Finnish syllabification: https://github.com/tsnaomi/finnsyll
- Turkish syllabification: https://github.com/MeteHanC/turkishnlp
- Shipibo-Konibo syllabification: Alva and Oncevay (2017)
- Hyphenation: PyPhen (https://pyphen.org/), which is based on Hunspell dictionaries.

Format For syllables in the LM task, we separate the subwords as: "A @ syl la ble @ con tains @ a @ sin gle @ vow el @ u nit", where "@" is a special token that indicates the word boundary. We also evaluated syllables with a segmentation format like in Sennrich et al. (2016): "A syl@ la@ ble con@ tains a ...", but we obtained lower performance in general. Whereas in the MT task, we adopt the segmentation format used by SentencePiece (Kudo and Richardson, 2018) for syllables: "_A _syl la

ble _con tains _a _sin gle _vow el _u nit".

D Type/token ratio of syllables in LM

In Figure 2a, we show a scatter plot of the token/type growth rate of syllables versus words for all languages and corpora. In other words, the ratio of syllable-types (syllabary or V_{syl}) per total number of syllable-tokens (N_{syl}) versus the type/token ratio of words (V_{word}/N_{word}) in the train set. The figure suggests at least a weak relationship, which agrees with the notion that a low word-vocabulary richness only requires a low syllabary richness for expressivity. Also, a richer vocabulary can use a richer syllabary or just longer words, so the distribution of the vocabulary richness could be larger.

We expected that the syllabary growth rate (V_{syl}/N_{syl}) for a low phonemic language like English would be relatively high, but wikitext-2 (en-wt2) is located in the bottom-left corner of the plot, probably caused by its large amount of word-tokens. However, we observed a large V_{syl}/N_{syl} for the English (en-UD) and French (fr) treebanks, despite their low V_{word}/N_{word} ratio, which is an expected pattern for deep orthographies.

We also observe that languages with a more transparent orthography, like Czech (cs) or Finnish (fi), are located in the left side of the figure, whereas Turkish (tr) is around the middle section. Nevertheless, our study does not aim to analyse the relationship between the level of phonemic orthography with the V_{syl}/N_{syl} ratio. For that purpose, we might need an instrument to measure how deep or shallow a language orthography is (Marjou, 2021; Borgwaldt et al., 2005; Borleffs et al., 2017), and a multi-parallel corpus for a more fair comparison.

Finally, in Figure 2b we observe a stronger relationship of the syllable type/token ratio with the difference of CHAR's ppl^c minus SYL's ppl^c. In other words, if our dataset possesses a rich syllabary, we are fairly approximating the amount of word-level tokens, which reduces the ppl^c gain.

E Model and Training

LM In contrast with the default settings, we use a smaller embedding size of 500 units for faster training. Additionally, we have 3 layers of depth, 1152 of hidden layer size and a dropout of 0.15. We train for 25 epochs with a batch size of 64, a learning rate of 0.002 and Adam optimiser (Kingma and Ba, 2015) with default parameters. We fit the model using the one cycle policy and an early stopping of 4. We run our experiments in a NVIDIA Titan Xp.

MT Similar to Mager et al. (2022), we use a small Transformer model for our low-resource MT settings, following Guzmán et al. (2019): "with 5 encoder and 5 decoder layers, where the number of attention heads, embedding dimension and inner-layer dimension are 2, 512 and 2048, respectively". For the pairwise systems, we train up to 100 epochs with an early stopping policy of 5 (validating every 5 epochs), whereas for the multilingual systems we train up to 30 epochs. For all the experiments, we use 4 NVIDIA GeForce GTX 1080 Ti GPUs.

F Human evaluation

F.1 Annotation protocol

Adapted and summarised from the AmericasNLP shared task (Mager et al., 2021): The expert received the source sentence in Spanish, the reference in Shipibo-Konibo, and an anonymized system output, which includes the baseline (BPE) and our syllable-based system (SYL). The expert received only 200 samples (per system, same entries) that were randomly selected and shuffled. They were asked to annotate **Adequacy** (Does the output sentence express the meaning of the reference?) from 1 to 5 (extremely bad, bad, neutral, sufficiently good, excellent), and **Fluency** (Is the output sentence easily readable and looks like a human-produced text?) from 1 to 5 as well.

F.2 About the annotator

The annotator is a native speaker of Shipibo-Konibo, a certified and professional translator, and a bilingual teacher in Peru. The annotator has experience in translating corpus for MT research, and performing human evaluation for Spanish–Shipibo-Konibo. This expertise is almost unique for Shipibo-Konibo, and we could not identify a second annotator with the same expertise to obtain inter-annotation agreement.

Aligning Multilingual Embeddings for Improved Code-switched Natural Language Understanding

Barah Fazili
IIT Bombay
barah@cse.iitb.ac.in

Preethi Jyothi
IIT Bombay
pjyothi@cse.iitb.ac.in

Abstract

Multilingual pretrained models, while effective on monolingual data, need additional training to work well with code-switched text. In this work, we present a novel idea of training multilingual models with alignment objectives using parallel text so as to explicitly align word representations with the same underlying semantics across languages. Such an explicit alignment step has a positive downstream effect and improves performance on multiple code-switched NLP tasks. We explore two alignment strategies and report improvements of up to 7.32%, 0.76% and 1.9% on Hindi-English Sentiment Analysis, Named Entity Recognition and Question Answering tasks compared to a competitive baseline model.

1 Introduction

Large pretrained multilingual models have enabled cross-lingual transfer on a number of downstream natural language understanding (NLU) tasks. Apart from serving as a good starting point to train models for tasks in low-resource languages, multilingual models (Devlin et al., 2018) (Conneau et al., 2019) have also been used to achieve zero-shot cross-lingual transfer on target languages with no task-specific labeled data. However, compared to monolingual inputs, the effectiveness of multilingual models on code-switched inputs—i.e., inputs with two or more languages appearing within or across sentences in a conversation—has not been explored enough.

In this work, we aim at explicitly modifying representations from pretrained multilingual models to be more amenable to code-switched inputs. We do this with the help of parallel text in the two component languages and alignment objectives that explicitly encourage representations to be better aligned across the two languages. We conjecture that modifying multilingual embeddings to be better aligned across the two languages will help the model deal better with tokens switching languages within a code-switched sentence. We start with a pretrained multilingual BERT (mBERT) baseline model (Devlin et al., 2018) and design two alignment objectives to be used with parallel text to align the multilingual embeddings. This "aligned" mBERT model is then further fine-tuned with small amounts of code-switched labeled data in the target task. We find such an aligned model to be more accurate on multiple downstream tasks involving code-switched inputs.

The two main highlights of this work can be summarized as follows:

- We propose two alignment-based objectives to be used with mBERT and parallel text in English and Hindi. The aligned models are fine-tuned and further evaluated on code-switched Hindi-English NER, SA and QA tasks. Compared to the baseline mBERT, we obtain clear improvements on all three downstream tasks.
- We investigate how our model behaves in the following two settings: 1) Using a bilingual lexicon instead of parallel text 2) Using Romanized Hindi instead of the native Devanagari script for Hindi.

We also present visualizations that clearly show that the alignment objective helps bring representations for aligned words in Hindi and English closer together.

2 Methodology

We explore two different objectives to encourage cross-lingual contextual alignment in the mBERT model. For this, we need access to parallel text in the component languages corresponding to the code-switched language of interest. We propose both a sequence-level alignment objective that is contrastive in nature, and a word-level alignment objective that is based on minimizing distances between aligned word embeddings.

Proceedings of the 29th International Conference on Computational Linguistics, pages 4268–4273
October 12–17, 2022.

2.1 Contrastive Loss for Sentence-level Alignments

Contrastive learning has been widely used in computer vision as a self-supervised technique to learn visual representations (Chen et al., 2020). Such contrastive objectives are becoming more popular for text-based tasks as well (Gao et al., 2021). We use a contrastive alignment objective with parallel text to improve cross-lingual alignment and potentially yield improved representations for code-switched text.

Consider a batch consisting of N pairs of parallel sentences $\{(x_1, y_1), \ldots, (x_N, y_N)\}$ extracted from a parallel corpus $\mathcal{C}$, where x_i and y_i denote sequences of words in two different languages. Our aim is to improve the alignment of a multilingual model f with respect to $\mathcal{C}$. Let $f(x_i)$ denote the contextual embedding of the word x_i due to the multilingual model f. The contrastive alignment objective is given by:

$$L_c = \frac{1}{2N}\sum_{i=1}^{N} -\log \frac{e^{(\mathrm{S}(f(x_i),f(y_i))/\tau)}}{\sum_{\substack{k=1\\k\neq i}}^{N} e^{(\mathrm{S}(f(x_i),f(y_k))/\tau)}} + \frac{1}{2N}\sum_{i=1}^{N} -\log \frac{e^{(\mathrm{S}(f(x_i),f(y_i))/\tau)}}{\sum_{\substack{k=1\\k\neq i}}^{N} e^{(\mathrm{S}(f(x_k),f(y_i))/\tau)}} + \eta \sum_{i=1}^{N} R_i(f) \quad (1)$$

where S is a similarity function, τ is a temperature hyperparameter and $R_i(f)$ is a regularization term with a scaling factor of η that is defined as:

$$R_i(f) = 2 - \mathrm{S}(f(x_i), f_0(x_i)) - \mathrm{S}(f(y_i), f_0(y_i)) \quad (2)$$

Here, f_0 denotes the initial pretrained model prior to alignment.

The contrastive objective in Equation (1) forces positive pairs ((x_i, y_i)) to be closer to each other and negative pairs ($(x_i, y_k), \forall k \neq i$) to be pushed further apart. The regularization term ensures that the aligned embeddings do not deviate too much from their initialization. The alignment algorithm using the contrastive objective is further elaborated in the following steps:

1. $f(x_i)$ is the embedding of the [CLS] token in mBERT after passing the entire sequence x_i as its input. For a given batch of N parallel pairs, the loss in Equation 1 is computed over all positive pairs, (x_i, y_i). There are two loss terms associated with each positive pair (x_i, y_i), each consisting of similarity scores between (x_i, y_k) (excluding y_i) and (x_k, y_i) (excluding x_i), respectively.

2. The similarity function between embeddings, denoted as S, is a cosine similarity function. The similarity scores are further scaled by a positive temperature hyperparameter.

3. The regularization term is composed of one loss term per (x_i, y_i) instance and explicitly penalizes divergences in embeddings from the initial pretrained model f_0.

4. The composite loss per batch is finally normalized by the number of positive instances considered per batch i.e. $2N$ pairs.

2.2 Multilingual Loss for Word-level Alignments

While the contrastive loss operates at the level of sentences, we also consider an alignment objective that operates at the level of individual words. This could be considered a more aggressive alignment technique since it encourages every aligned word in parallel sentences to be close together. For every parallel sentence pair (x_i, y_i), we first use an off-the-shelf alignment tool called `awesome-align` (Dou and Neubig, 2021)[1] to extract word alignments. We further filter the aligned pairs based on an alignment prediction probability (set to 0.9 in our experiments) to ensure that we only use high-quality word alignments. If there are N parallel sentences $\{(x_1, y_1), \ldots, (x_N, y_N)\}$ in a batch and $a(x_i, y_i)$ represents a list of index tuples $\{(1, j), \ldots, (m, n)\}$ denoting the aligned word indices in the parallel sentence pair (x_i, y_i), the alignment objective can be written as:

$$L_m = \frac{1}{B}\sum_{i=1}^{N} \sum_{\substack{(m,n)\in\\ \{a(x_i,y_i)\}}} \mathrm{S}(f(x_{i,m}), f(y_{i,n})) + R_i(f)$$

where $x_{i,m}$, $y_{i,n}$ denotes the mth and n^{th} word in x_i and y_i, respectively, and B denotes the total number of successfully aligned word-pairs in the batch.

[1] https://github.com/neulab/awesome-align

$R_i(f)$ is the regularization term defined in Equation (2). Note that $f(x_{i,m})$ in L_m refers to a contextual embedding, while $f(x_i)$ in the contrastive loss L_c is the embedding of the [CLS] token.[2]

3 Experiments and Results

3.1 Experimental Setup

3.1.1 Dataset Details

We evaluated our aligned models on three downstream tasks in code-switched Hindi-English — SA (Sentiment Analysis), NER (Named Entity Recognition) and QA (Question Answering) — from the GLUECoS (Khanuja et al., 2020) benchmark. Tasks in the GLUECoS benchmark can be grouped into two categories, sequence labeling tasks (NER, etc.) and tasks requiring deeper semantic understanding (sentiment analysis, etc.). We evaluated our techniques on three tasks, NER, SA and QA, spanning both categories.[3]

NER and QA datasets contain Hindi in the Romanized form, while the SA evaluation sets use the native Devanagari script for Hindi. As an evaluation metric, we use F1 scores for all three tasks. For the cross-lingual alignment training phase, we used parallel text in English-Hindi from the IIT Bombay English-Hindi Corpus (Kunchukuttan et al., 2017). Alternatively, we also experimented with using a bilingual lexicon, MUSE (Lample et al., 2018), instead of parallel text.

GLUECoS NER is sourced from a Twitter NER corpus (Singh et al., 2018) with 2467/308/307 train/dev/test instances. The sentiment analysis dataset is taken from the ICON 2017 shared task; Sentiment Analysis for Indian Languages (SAIL) (Patra et al., 2018) and has 10080/1260/1260 instances in the train/dev/test splits. The QA dataset (Chandu et al., 2018) includes 259/54 instances in the train/dev sets, respectively.

3.1.2 Model Implementation

We use Multilingal BERT (Devlin et al., 2018) (base) as our baseline pretrained multilingual model, that is also the baseline of choice for the GLUECoS benchmark. Subsequent works reporting results on GLUECoS (e.g., Santy et al. (2021)) also used mBERT as their base model. This motivated us to stick to mBERT so that we could reproduce the baseline numbers and contextualize our improvements better compared to prior work.

We train mBERT with the alignment objectives in two different ways: 1) Train all 12 mBERT layers with the alignment objective and 2) Only train a newly-introduced linear layer on top of mBERT and freezing the remaining mBERT layers. The new linear layer will have the same number of input and output dimensions as the last layer in mBERT (i.e., 768 in mBERT base). For training the linear layer, we use the AdamW optimizer at a learning rate of 0.001 with early stopping (and patience set to 10). For training all the mBERT layers, we choose a smaller learning rate of $5e-5$. For the contrastive objective L_c, we used a validation set to tune the scaling factor for the regularization term η and the temperature values. For the multilingual alignment L_m, we only tuned the scaling factor η for the regularization term.

SA (Devanagari)	dev	test
Baseline	$60.3_{\pm 0.00}$	$64.2_{\pm 0.03}$
L_c (|| Devanagari)	$59.4_{\pm 0.01}$	$66.3_{\pm 0.04}$
L_m (|| Devanagari)	$\mathbf{61.0}_{\pm 0.01}$	$\mathbf{68.9}_{\pm 0.03}$
L_c (MUSE Devanagari)	$60.7_{\pm 0.00}$	$67.8_{\pm 0.04}$
L_m (MUSE Devanagari)	$59.6_{\pm 0.01}$	$65.9_{\pm 0.02}$

Table 3.1: F-scores after intermediate pretraining of standard mBERT using various alignment schemes on the GLUECoS SA task. || refers to the use of parallel text, and MUSE is the bilingual lexicon. L_c,L_m refer to the contrastive and multilingual alignment schemes.

3.2 Results on Downstream Tasks

Table 3.1 lists the F scores on the GLUECoS SA task. The alignment training was done either using parallel text from the IITB Parallel Corpus or the bilingual lexicon from MUSE. This alignment training phase was followed by finetuning on the code-switched Hindi-English SA training data. We see significant improvements in F1 scores for all alignment training schemes. The best F1 score on SA Devanagari is achieved with multilingual alignment over the IITB parallel corpus.

Table 3.2 shows results on both NER and QA. The alignment training is different from SA (in Table 3.1) with only training a newly-added linear

[2]Code and data supporting our work is available at: https://github.com/BarahFazili/AlignmentForCS.

[3]We did not consider the remaining two sequence labeling tasks of GLUECoS — LID-tagging and POS-tagging — since they already yielded fairly high baseline mBERT scores (>95 and >87 for LID and POS, respectively).

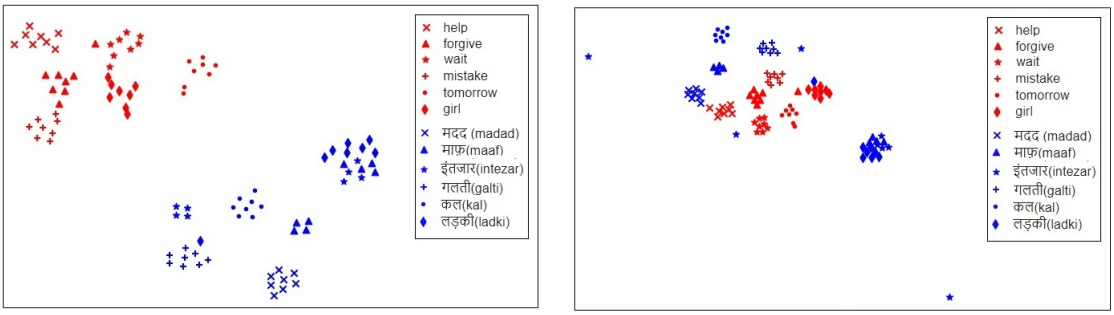

(a) Baseline mBERT

(b) Aligned mBERT

Figure 3.1: t-SNE plots using 48 instances of words in each language before and after aligning mBERT using L_m.

layer on top of frozen mBERT layers.[4] Baseline (rand) refers to adding a randomly initialized linear layer on top of the baseline, that is subject to no alignment training and only task-specific finetuning. We observe clear performance improvements on both NER and QA.

Visualizing the alignments. Fig 3.1 visualizes the change in embeddings after the alignment training. We selected 6 pairs of parallel Hindi-English words and created a set of 48 parallel sentences in monolingual Hindi and English; each of the six pairs appear in eight sentences each (in their corresponding scripts). Embeddings for these words across all 96 sentences were extracted from both the baseline mBERT and our multilingual aligned mBERT and plotted in 2D using t-SNE. As seen in Fig 3.1, the parallel words are now closer to each other in the aligned plot irrespective of the underlying language.

[4]Backpropagating through all mBERT layers significantly degrades performance for QA. Conversely, training only a linear layer for SA while freezing mBERT layers did not help.

System	NER dev	QA dev
Baseline	$78.7_{\pm 0.01}$	$73.5_{\pm 2.75}$
Baseline (rand)	$78.5_{\pm 0.01}$	$71.8_{\pm 1.67}$
L_m (|| Roman)	$79.3_{\pm 0.00}$	$74.0_{\pm 2.20}$
L_c (|| Roman)	$79.0_{\pm 0.01}$	$74.3_{\pm 2.99}$
L_m (|| Devanagari)	$79.2_{\pm 0.00}$	$72.3_{\pm 0.69}$
L_c (|| Devanagari)	$78.8_{\pm 0.01}$	$74.0_{\pm 1.36}$
L_c (MUSE Roman)	-	$74.9_{\pm 1.44}$
L_m (MUSE Roman)	-	$72.6_{\pm 1.91}$

Table 3.2: F scores after intermediate pretraining of linear layer added on top of frozen standard mBERT using various alignment schemes on the GLUECoS NER,QA and SA tasks

4 Related Work

Large pretrained multilingual models, such as mBERT (Devlin et al., 2019) and XLM-R (Conneau et al., 2020) have achieved state-of-the-art results on monolingual and cross-lingual benchmark tasks. However, their efficacy on code-switched tasks have not been sufficiently explored. (Winata et al., 2021) observed that pretrained multilingual models do not necessarily guarantee effective representations for code-switched text.

Prior work has explored different ways of adapting multilingual pretrained models to be effective for code-switched data. Prasad et al. explore bilingual intermediate pretraining to derive large and consistent performance gains on three different NLP tasks on code-switched text. Santy et al. finetune mBERT with synthetic code switched data generated using random lexical substitution and code-switching constraints based on linguistic theories. Chakravarthy et al. (2020) also pretrain mBERT on code-switched text and adopt other data augmentation techniques to derive performance gains. Aguilar et al. (2021) focus on the role of tokenization and propose a hybrid technique that processes in-vocabulary and out-of-vocabulary tokens differently and observe improvements on three different code-switched NLP tasks. Gupta et al. (2021) use unsupervised self-training to predict pseudolabels on the target task and retain high-confidence predictions as labeled samples that are further used to finetune the model. This leads to a boost in performance on the task of code-switched sentiment analysis.

We note that the L2 alignment technique in Wu and Dredze (2020) is the same as our word-level multilingual alignment objective except for regularizing model parameters rather than the model out-

put embeddings (before and after alignment). Wu and Dredze (2020) show two variants of contrastive alignment termed "weak" and "strong". The weak variant strictly uses negative pairs from the other language, while the strong variant uses negative pairs from both within and outside the language. We used the weak variant in our work to avoid overfitting since we did not have a lot of data for alignment training.

Our work departs from prior work on improving NLU for code-switched inputs in that it is the first to explore the use of alignment objectives with parallel text to modify the multilingual representations and make them more suitable for code-switched tasks. Recent work from Deshpande et al. (2021) corroborates our findings and establishes a strong correlation between embedding alignments and downstream performance on cross-lingual transfer. While they present a post-hoc empirical analysis of what factors benefit cross-lingual transfer the most, we explicitly use an alignment-based training for better alignment between languages and improve downstream task performance.

5 Conclusion

In this work, we propose aligning multilingual embeddings using sentence-level (contrastive) and word-level (non-contrastive) objectives. Such an explicit alignment leads to improved performance on three code-switched Hindi-English NLP tasks: SA, NER and QA. Future work will explore the use of alignment objectives in a multi-task framework with the target tasks.

Acknowledgements

The authors thank the anonymous reviewers for their constructive suggestions to improve the draft. The second author gratefully acknowledges support from Google Research India towards research on code-mixing.

References

Gustavo Aguilar, Bryan McCann, Tong Niu, Nazneen Rajani, Nitish Keskar, and Thamar Solorio. 2021. Char2subword: Extending the subword embedding space using robust character compositionality.

Sharanya Chakravarthy, Anjana Umapathy, and Alan W Black. 2020. Detecting entailment in code-mixed Hindi-English conversations. In *Proceedings of the Sixth Workshop on Noisy User-generated Text (W-NUT 2020)*, pages 165–170, Online. Association for Computational Linguistics.

Khyathi Chandu, Ekaterina Loginova, Vishal Gupta, Josef van Genabith, Günter Neumann, Manoj Chinnakotla, Eric Nyberg, and Alan W. Black. 2018. Code-mixed question answering challenge: Crowdsourcing data and techniques. In *Proceedings of the Third Workshop on Computational Approaches to Linguistic Code-Switching*, pages 29–38, Melbourne, Australia. Association for Computational Linguistics.

Ting Chen, Simon Kornblith, Mohammad Norouzi, and Geoffrey Hinton. 2020. A simple framework for contrastive learning of visual representations.

Alexis Conneau, Kartikay Khandelwal, Naman Goyal, Vishrav Chaudhary, Guillaume Wenzek, Francisco Guzmán, Edouard Grave, Myle Ott, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Unsupervised cross-lingual representation learning at scale.

Alexis Conneau, Kartikay Khandelwal, Naman Goyal, Vishrav Chaudhary, Guillaume Wenzek, Francisco Guzmán, Edouard Grave, Myle Ott, Luke Zettlemoyer, and Veselin Stoyanov. 2020. Unsupervised cross-lingual representation learning at scale.

Ameet Deshpande, Partha Talukdar, and Karthik Narasimhan. 2021. When is bert multilingual? isolating crucial ingredients for cross-lingual transfer.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. Bert: Pre-training of deep bidirectional transformers for language understanding.

Zi-Yi Dou and Graham Neubig. 2021. awesomealign.

Tianyu Gao, Xingcheng Yao, and Danqi Chen. 2021. Simcse: Simple contrastive learning of sentence embeddings.

Akshat Gupta, Sargam Menghani, Sai Krishna Rallabandi, and Alan W Black. 2021. Unsupervised self-training for sentiment analysis of code-switched data.

Simran Khanuja, Sandipan Dandapat, Anirudh Srinivasan, Sunayana Sitaram, and Monojit Choudhury. 2020. Gluecos : An evaluation benchmark for code-switched nlp.

Anoop Kunchukuttan, Pratik Mehta, and Pushpak Bhattacharyya. 2017. The iit bombay english-hindi parallel corpus.

Guillaume Lample, Alexis Conneau, Marc'Aurelio Ranzato, Ludovic Denoyer, and Hervé Jégou. 2018. Word translation without parallel data. In *International Conference on Learning Representations*.

Braja Gopal Patra, Dipankar Das, and Amitava Das. 2018. Sentiment analysis of code-mixed indian languages: An overview of sail$_{code}$ − $mixedsharedtask@icon$ − 2017.

Archiki Prasad, Mohammad Ali Rehan, Shreya Pathak, and Preethi Jyothi. 2021. The effectiveness of intermediate-task training for code-switched natural language understanding.

Sebastin Santy, Anirudh Srinivasan, and Monojit Choudhury. 2021. Bertologicomix: How does code-mixing interact with multilingual bert? In *AdaptNLP EACL 2021*.

Vinay Singh, Deepanshu Vijay, Syed Sarfaraz Akhtar, and Manish Shrivastava. 2018. Named entity recognition for Hindi-English code-mixed social media text. In *Proceedings of the Seventh Named Entities Workshop*, pages 27–35, Melbourne, Australia. Association for Computational Linguistics.

Genta Indra Winata, Samuel Cahyawijaya, Zihan Liu, Zhaojiang Lin, Andrea Madotto, and Pascale Fung. 2021. Are multilingual models effective in code-switching?

Shijie Wu and Mark Dredze. 2020. Do explicit alignments robustly improve multilingual encoders? In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 4471–4482, Online. Association for Computational Linguistics.

Fashioning Local Designs from Generic Speech Technologies in an Australian Aboriginal Community

Éric Le Ferrand,[1,2] **Steven Bird,**[1] **and Laurent Besacier**[2]
[1]Northern Institute, Charles Darwin University, Australia
[2]Laboratoire Informatique de Grenoble, Université Grenoble Alpes, France

Abstract

An increasing number of papers have been addressing issues related to low-resource languages and the transcription bottleneck paradigm. After several years spent in Northern Australia, where some of the strongest Aboriginal languages are spoken, we could observe a gap between the motivations depicted in research contributions in this space and the Northern Australian context. In this paper, we address this gap in research by exploring the potential of speech recognition in an Aboriginal community. We describe our work from training a spoken term detection system to its implementation in an activity with Aboriginal participants. We report here on one side how speech recognition technologies can find their place in an Aboriginal context and, on the other, methodological paths that allowed us to reach better comprehension and engagement from Aboriginal participants.

1 Introduction

A consistent theme in recent NLP research has been *doing more with less* (Wiesner et al., 2022; Gao et al., 2021; Baevski et al., 2021; Schneider et al., 2019; Menon et al., 2019). It is popular to describe new pipelines to solve a wide range of tasks for under-resourced languages (Godard et al., 2018; Anastasopoulos et al., 2018; Settle et al., 2017; Mitra et al., 2016; Lane and Bird, 2019). However, the motivations behind the design of a computational method are not systematically well justified according to the needs of the target speech communities.

The category of under-resourced languages encompasses a wide range of contexts, not simply in terms of the quantity of data available but also in terms of local speech communities' sociolinguistic and political situation (Bird, 2022). Often, the focus has been to generalise a given method across languages, where the proposed system is at the core of the argument instead of the benefits that it could have for the speakers. We could ask whether the same language technology would be equally applicable to Marathi, spoken by millions in a major metropolis, and Miriwoong, with only a few elderly speakers in a remote Australian Aboriginal community (cf. Kuhn, 2022).

Universal solutions dominate NLP: research and results are often provided without taking into account the global situation of the languages involved or the views of the speech communities about the preservation of their language. Instead, it is common to assert that an improvement in Word Error Rate yielded by a given speech recognition system is the answer to the transcription bottleneck and, therefore, the problem of scaling up language documentation (van Esch et al., 2019; Foley et al., 2018).

Most of the world's languages are primarily oral (Ong, 1982; Walsh and Yallop, 1993). Writing is often not a priority, and very few people are skilled in transcribing their language. Written resources often only exist in limited spaces where there is a collaboration between westerners and local communities, such as schools, ranger programs, tourism, and academia. In such cases, writing would seem to primarily serve institutional agendas (cf. Dobrin et al., 2009; Perley, 2012; Nevins, 2013). Accordingly, we must ask ourselves to what extent automatic transcription technologies have a place in research that respects local self-determination. Bird (2022) calls for a *local turn*, for the need to work with local speech communities from the ground up. In other words, outsiders who enter communities with their expertise need to begin with local concerns and local knowledge practices, and only later begin to explore ways in which language technologies can be added into the mix. For example, a local person might want non-indigenous colleagues to learn and use the local language, rather than assuming that all work is conducted in English. We have found that such an approach enlarges the opportunities

Proceedings of the 29th International Conference on Computational Linguistics, pages 4274–4285
October 12–17, 2022.

for collaboration, while simultaneously generating language resources.

This paper extends our previous work on collaborative transcription (Le Ferrand et al., 2022), where the language documentation pipeline we designed failed. We were confronted with different ways of knowing and different expectations in terms of language work. In this work, learning from our past failure, we describe our approach, from the training of a transcription system to the design of collaborative transcription activities with Aboriginal participants. We first describe our speech recognition method based on syllable spotting. We then present the design of the app used that bridges the output of the syllable spotting system to the people, taking into account existing practices. We also explain our method to engage with participants to address their interests in terms of language work. Finally, we detail the application of the proposed transcription activities and discuss the success and flaws of this work.

2 Background

2.1 Decolonising practices

Research contributions around speech processing for low-resource languages have often followed the work of documentary linguistics, where some automation is added to support manual annotations (Adams et al., 2018; Godard et al., 2018; Foley et al., 2018). The 7000+ world languages are often mentioned and language technologies appear as a way to prevent their loss (Adda et al., 2016; Duong, 2017; Jimerson and Prud'hommeaux, 2018). Special workshops like the zero resource challenge[1] and the introduction of a surprise language have pushed in this direction allowing the creation of computational solutions that bypass the need of the speech communities of language experts. Recent studies have also shown that the languages (Schwartz, 2022) or the speech communities (Caselli et al., 2021) are rarely ate the core of the argument in the ACL anthology's publications.

Documentary linguistics is often the preliminary step of language description and analysis (Hanke, 2017). Documentation and description communicate with each other to allow western scholars to have a better comprehension of Indigenous languages. There are no clear benefits for the speech community, and extra work needs to be provided to share the benefits of a research project (Chelliah and De Reuse, 2010). The NHMRC Guidelines[2] for Ethical Conduct in research with Aboriginal and Torres Strait Islander Peoples and Communities set out principles of equity and reciprocity, where the outcome of the research should benefit both parties. Recent research practices, including documentary linguistics, started to fully commit to these standards by adopting a community-based approach (e.g. Rodríguez Louro and Collard, 2021; Ryder et al., 2021; Taylor et al., 2020). Community-based research has the community at its core and is meant to be conducted for and with the participation of community members (Rice, 2011).

2.2 Community-based projects

Community-based research around software design is a small but growing area. Projects have been based on research Human-Computer Interaction (HCI) or NLP from a language learning perspective. On the HCI side, research has contributed to responding to local issues by designing tools in collaboration with the community (Soro et al., 2017; Hardy et al., 2016; Leong et al., 2019). Cross-cultural collaboration is challenging. From this kind of project have also emerged engagement methods to facilitate the conversation with Indigenous communities about technology design (Zaman et al., 2016; Taylor et al., 2020). On the NLP side, the research contributions have been language-specific or bounded to a specific context. For instance, Pine et al. (2022) have described speech synthesis systems in several Indigenous Canadian languages responding to a call from the language learners. Projects that did not initially have a community-based component sometimes ended up serving community-based projects. Uí Dhonnchadha and Van Genabith (2006) for instance, created a POS tagger for gaellig Irish. The system has been then incorporated into an Irish learning game (Xu et al., 2022). In either case, the majority of the work done in this area is based on writing (e.g. Lane and Bird, 2019; Schwartz et al., 2019; Finn et al., 2022). The only speech-based projects are around speech synthesis (Harrigan et al., 2019; Pine et al., 2022). Speech recognition seems to be rarely involved in community-based projects.

[1] https://www.zerospeech.com/

[2] https://www.nhmrc.gov.au/about-us/resources/ethical-conduct-research-aboriginal-and-torres-strait-islander-peoples-and-communities#block-views-block-file-attachments-content-block-1

2.3 Context

Our work is grounded in Bininj country in West Arnhem, Northern Territory in the Australian Top End. Bininj country is part of the Indigenous Protected Area of Arnhem Land where the land and sea are managed by Aboriginal groups.[3] The main language of communication is Kunwinjku (ISO gup) which is spoken by approximately 2500 people (Marley, 2021). There is a standard orthography that has been introduced by linguists but it is not widely used by the members of the community.

The first and second authors have several years of experience with the Bininj community, have some expertise in Kunwinjku, the local language, and have both been adopted by Traditional owners of the land. In this case, adoption means the attribution of a *skin name* that connects an individual to the rest of the community (cf. Christie, 2008, p.35).

2.4 Learning from failure

This work is the continuation of Le Ferrand et al. (2022). We previously designed a spoken term detection prototype to detect whole words in untranscribed speech collections in Kunwinjku. We then used an app to bridge the output of our prototype to the people to allow local communities to verify the guesses of our system and therefore be part of transcription works. We faced many challenges that we tried to build on in this work.

This previous work focused on the collection of data to enhance the performance of the system. The design ended up being irrelevant and redundant for the participants. From here we realised the need for further discussion with the community to set up activities that are relevant to their agenda, interests and practices.

The app presented displayed only four buttons: one to play the query, one to play the utterance and two to give a feedback on whether the query has been spotted in the utterance or not. While testing the app, we realised how the audio files extracted from their contexts were confusing for the participants. Besides, the fact of validating system guesses in random utterances was disconnected from the idea of transcription which led most participants to overthink the task.

In projects around cross-cultural technology design, shallow information is provided about the extent of the collaboration and the challenges encountered. Yet, studies have described ways of knowing in Indigenous communities that differs from the western approach to knowledge (Descola, 2005; Foley, 2003). Such differences appear as the main reason behind the failed attempt of app design where the proposed task lose all meaning in Bininj context.

From our first failed attempt, the challenges were two-fold. We first needed to figure out a way to solve the comprehension issues we have faced. Then, we needed to improve the relevance of this work for Bininj participants. The key was to find out how to design transcription technologies based on existing practices. From the language learning sessions we had with some of our Aboriginal collaborators, we noticed, for instance, how they teach us breaking down words into syllables to decompose the pronunciation of a given item. This led us to think about replacing word spotting with syllable spotting, allowing participants to reproduce their word decomposition strategy to build up the transcription from the syllables spotted. From here, the focus needed to be given on incorporating this transcription strategy into an activity that matters to the people.

3 Transcription by syllables

3.1 Data

To build the system, we are using a corpus in Kunwinjku built from several sources. The training and validation sets consist of 35.45 min and 7.39 min respectively of spontaneous speech made of guided tours of Aboriginal towns and utterances for language description purposes. Two different sets are used for testing: one set of 19.43 min of spontaneous utterances and one set of 4.43 min of elicited words recorded in isolation.

To build our list of valid syllables, we used a word list built from the Bible in Kunwinjku. We then applied on each word syllable segmentation rules resulting in a set of 584 unique syllables with relative frequency values associated.

3.2 Experimental setup

Le Ferrand et al. (2021) introduced a method of spoken term detection for very low-resource languages based on phone recognition. Their method is based on Allosaurus (Li et al., 2020), a universal phone recognizer. We preferred this method in this work due to its flexibility in terms of query selection and its speed compared to Dynamic Time Warp-

[3] https://www.awe.gov.au/agriculture-land/land/indigenous-protected-areas

ing, which is usually used for very low-resource languages.

We first trained the phone recognizer using our train set and generated confusion matrix from the validation and test sets. A confusion matrix consists of a phone transcription and the top k (we use k=5) most likely alternatives per phone with a likelihood score associated. To spot syllables, we expressed the syllables extracted from the bible as a finite state automaton after conversion from graphs to phones and explored every possible path in the phone matrix that corresponded to a valid transition in the lexicon. Ultimately we extracted the resulting syllables with the mean of the phones' scores that are used as a likelihood measure to filter the syllables spotted based on a threshold *T*.

To increase the accuracy of the method of Le Ferrand et al. (2021) which only relies on the likelihood scores output by allosaurus, we used the frequency information in our syllable list to more precisely select our candidates. To do so, we average the likelihood score L_s of a detected syllable with its unigram probability P_s weighted with a constant α as:

$$L_s + \alpha P_s \quad (1)$$

We then optimised, on the validation set, α varying a range of values between 0 and 10 with a 0.1 step and a syllable detection threshold T between 0 and 10 with a step at 0.01. We then spotted syllables on the test set with the parameters which provided the best F-score on validation. We also report results without the frequency where only the threshold T is optimized on the validation set.

3.3 Experimental results

Our best results on the validation set have been obtained with $T = 0.39$ when unigram probability is added. For our baseline without unigram probability, the best threshold has been obtained with $T = 0.35$ We report the results in Table 1.

We can see here that the frequency information has an impact on the overall performances in both scenarios with an F-score nearly 4 points higher in the results with frequency. Better performances are obtained on the test set made of utterances. Two elements can explain it. First, the phone recognition model has been trained on similar data to the utterance test set which leads to better phone recognition performances. Then, the chance of a given syllable being pronounced several times is higher in longer utterances which means that it has higher chance to be spotted.

Results with likelihood score alone			
Sets	Recall	Precision	F-score
Words	41.71%	24.40%	30.79%
Utterances	47.21%	36.26%	41.02%
+ unigram probability			
Sets	Recall	Precision	F-score
Words	43.08%	28.09%	34.23%
Utterances	46.56%	41.50%	43.88%

Table 1: Experimental results (syllable spotting) on the two test sets

4 App design

4.1 Prototype

We designed a simple interface to display the syllables from our spoken term detection systems to our participants (see Figure 1). Our goal here was not to design a final product but to present a simple interface that works well enough to see if the proposed syllable concatenation mechanism makes sense from a Bininj perspective. We bridged the output of the system to a transcription interface by creating one button per syllable spotted for a given audio recording. The buttons display the orthography of the syllables spotted. They play the corresponding pronunciation when clicked. There is one *play* button to play the audio to transcribe and one text area with an associated *play* button to look for syllables that have not been spotted. The user needs to use the keyboard to make guesses on missing syllables and needs to click on the *play* button to check the pronunciation of their guesses.

Figure 1: Preliminary version of the app

We organised a testing session with one participant in Gunbalanya: IG, a 25 year old local artist and tour guide. We spotted syllables in a 3.35min recording made of elicited speech of Bible stories. Because of the quality of the audio, most of the syllables were correctly spotted. We explained to IG that we wanted to write down Kunwinjku and we needed his help to spell the words.

IG rapidly understood the task and started point-

ing syllables on the screen while we were writing with pen and paper IG's feedback. He clicked several times on the different syllables displayed and progressively gave feedback. When a syllable was not spotted, he could with some hesitancy, write with the keyboard syllables guesses in the text area.

The main observations made during the pilot study were IG's quick comprehension of the task, his hesitancy while using the keyboard and his confidence while reporting the orthography. At the end of the activity, he told us that he was expecting the text area to produce a new syllable button he could use.

4.2 Design and features

The quick comprehension of IG showed the potential of the proposed transcription mechanism which made us pursue this direction. Based on the first trial, we designed a proper transcription interface based on syllable spotting (see Figure 2). The core of the interface was the same that our first trial: we have a play button on the top of the screen playing the target audio to transcribe. We have one button per detected syllable associated with a wav file containing their pronunciation. The syllables can be dragged and dropped to the black box at the bottom of the screen. The user can listen to the final concatenation of the syllables with the associated play button and validate the transcription created with a thumb up button.

We needed to find a way to allow the user to add undetected syllables manually. To do so, we initially added a side menu accessible through a plus button on the side of the screen. The menu consisted of a scrolling list that contained the 584 syllables. We added a text area at the top of the list that allowed the user to retrieve a syllable from its first letters (see Figure 3). Following the principle of the regular syllable button, the user could click on the syllable to hear the pronunciation and click on the associated plus button once their choice was made. The syllable was then added as a regular syllable button. To avoid the use of the keyboard, we changed this syllable search mechanism by removing the text area and by replacing the list of syllables with expandable sub-lists labelled with the first graph[4] of the syllables it contains (see Figure 4). The user can then search for a syllable by expanding the lists and select a syllable by listening to it and clicking on the associated plus button. The app and databases were stored in a laptop accessed remotely by a tablet with wifi.

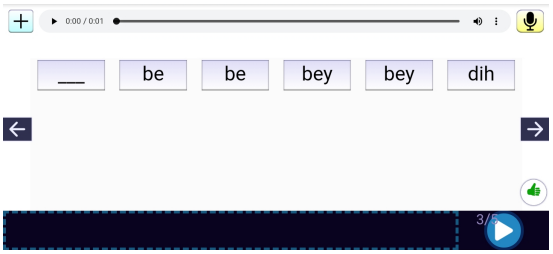

Figure 2: Final version of the app

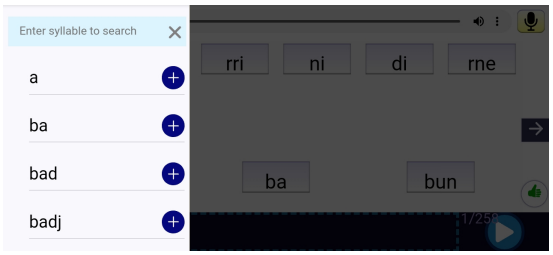

Figure 3: Initial syllable search mechanism

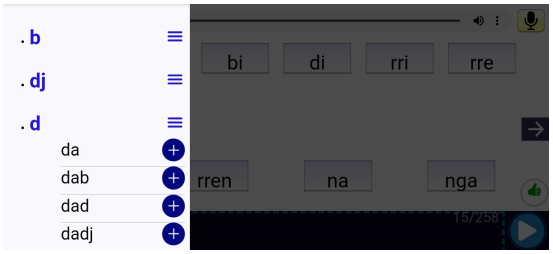

Figure 4: Updated syllable search mechanism

5 User testing

Due to Covid-19 restrictions, no trips to remote communities were possible. However, we have been able to work individually with Kunwinjku speakers in transit in Darwin at the university. We incorporated our syllable spotting based transcription task in a more global resource creation workflow. We could test it with two participants from Bininj country. In order to engage with the participants, we organised the testing phase in two sections. In the first one, we discussed and elicited knowledge about topics of interest based on previous conversations, in the second, we used the interface to transcribe the knowledge recorded. Therefore, besides the focus given to the design of the app and spoken term detection system, time of this

[4]We are not talking in terms of individual letter but graph or group of graphs that correspond to a single phone in Kunwinjku

project has been dedicated to the study of cultural elements to enable more efficient collaboration.

5.1 Activity description

Elicitation of knowledge

> *Ngabenbekken nahni wurdwurd nawu kabirrihre minj Kundebi kabirrikarrme. Burrkyak. Kabirridjalngeybun. Minj kabirridebikarren, burrkyak.*
>
> "I hear these children going about – they don't have Kundebi. No. They just use people's names. They don't use Kundebi with each other, no." (Etherington, 2006)

Language shift is not a new phenomenon. Language variation in Kunwinjku has been the subject of recent research (Marley, 2021) and has been one of the concerns raised by Bininj Elders. *Kundebi* specifically has been described as a language feature that the community is proud of and that is being progressively lost by the young generation (Garde, 2013; Etherington, 2006). It has also been mentioned in the same terms by some Elders during some of our fieldtrips. *Kundebi* refers to the way a speaker A refers to an individual C while talking to an addressee B. For example, a speaker A is talking to their elder sister's child B about their elder sister C. A is usually referring to B using the term *djedje* "nephew" and to C using the term *yabok* "sister". Listener B however usually refers to C using the term *morlah* "mother's elder sister". The kundebi term *berlunghkowarre* is then used to summarize these three relationships and could be translated as "my sister, your mother's elder sister, you are my sister's child" (Garde, 2013).

In order to respond to people's priority in terms of language work, we have decided to first focus the activity on the creation of written resources around *Kundebi*. To do so, while working with a Kunwinjku speaker, we would talk about common acquaintances, identify the way we both would refer to them and then identify and record the corresponding *Kundebi* terms. We used an activity sheet (see Figure 5) to draw the relationships we wanted to elicit (for instance, E for first author, G for the participant and J for the person we are talking about). The recording is directly stored on our laptop. The speed of the pipeline, described in Section 3.2, also allowed us to directly spot the syllables in the audio. Some of our participants expressed the fact that they were not confident with

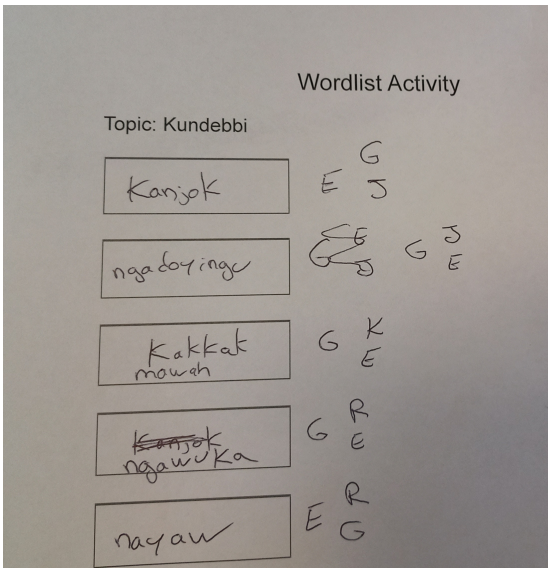

Figure 5: activity sheet filled

Kundebi and would feel more comfortable talking about *Kunbalak*. *Kunbalak* is a sub-language used for forbidden relationships to show respect. It is identical to regular Kunwinjku syntactically but would use different lexical items. For instance *Birriwam* "they went" becomes *birridokang* in *Kunbalak* (Manakgu, 1996). To elicit *Kunbalak* we would just ask for the conversion of regular Kunwinjku terms.

Use of the app

After recording a few terms with a speaker, we presented the transcription interface to them. The terms previously recorded and the syllables spotted have been automatically loaded into the app database. After showing the interface's different features to the participants, we asked them to drag and drop the syllables to build the transcription of the previously recorded terms. After actively working around Kunwinjku and building expertise about the proper way to write the language through the years, we let the participants use their own expertise on what they think is the orthography without questioning their authority.

5.2 Fieldwork

We tested our pipeline with two participants. JB (30s) and GB (30s).

We could present our activity to JB on three different occasions. We could identify and record some *kundebi* terms during the first trial. The activity has then been interrupted by upset child. During this first trial, she briefly started to point syllables on the screen without properly using the app.

She told us afterwards that the *kundebi* terms we recorded should be double-checked by an Elder, and she would feel more confident talking about *Kunbalak* instead. During the second and third trials, we could easily identify and record some *Kunbalak* terms. While using the app, we faced minor technical difficulties with the manual syllable addition feature. However, JB could take control of the tablet to transcribe some of the terms recorded. One of the issues we faced was the playback of syllables that include a glottal stop which was hard to identify in syllables in isolation (the difference between *ma* and *mah*, for instance). The activity was trialled with the first version of the syllable search (see Figure 3). The keyboard generated by the text area would take most of the space on the screen. JB needed to ask for our support to know how to proceed. At the end of the second trial, while no instruction had been explicitly given, she started to drag and drop the syllables available on the screen to explore the different words that are possible with them. We asked about her thoughts about the activity, and she responded that she liked it and would like to get more confident in writing in Kunwok and download the app later.

We could test the activity with GB, the second participant. We first recorded a few *Kundebi* terms. We wrote on paper the relationship to elicit, which made him understand the activity was about constructing a word list. After recording a few terms, we gave GB the tablet and asked him to transcribe the words. For each term, he listened to the audio first and pressed the syllable displayed on the screen. He was able to add new syllables manually without too much difficulty. For one particular term: *nayaw*, we discussed rather the term should be written *nayaw* or *nayawu*. While listening for a given syllable, he sometimes asked for confirmation about what he heard (for instance, "Is this *ka*?"). We discussed his thought about the task at the end of the activity. He showed enthusiasm about the incremental construction of the transcription. During the activity, he rephrased the syllable concatenation process by "putting pieces of language together".

No more participants were available for the time for this project. However, to sustain this work in the future, we deployed it in a laptop to be brought to the community by future scholars or language workers, as soon as COVID-19 restrictions are eased.

Figure 6: Picture of a participant using the app

6 Discussion and Limitations

The design and testing of the activity have shown promising results among a few participants, which gave us a glance at the potential of syllable spotting for the design of language related activity for Aboriginal people.

Syllable Spotting: It has been shown in the literature that traditional ASR is hardly applicable to Aboriginal languages due to the lack of resources available to train robust systems. Sub-word detection has been seen as a way to avoid out-of-vocabulary (Szoke et al., 2008; Parlak and Saraclar, 2008; Van Heerden et al., 2017) and, in our case, to allow a denser transcription than word spotting specifically for a polysynthetic language like Kunwinjku. Adding information on frequency, not surprisingly, allowed us to boost our performance (F-score) from 40% to nearly 44% for the syllables displayed on the screen for a given utterance.

Enabling mutual comprehension: Our main objective, starting from our previous work, was to enable a better comprehension in our cross-cultural setting. Part of this process consisted of getting familiar with cultural components that have been raised by the community (namely, *Kundebi* and *Kunbalak*). This also consisted of finding methods to trigger a conversation about these topics. For the rest, strategies have been found to help the participants to understand our contribution is this work. For instance, the support of the activity sheet made clear that the ultimate goal of the activity was to build a word list. Then the syllable concatenation mechanism allowed the participants to leverage existing language patterns from the aural space into writing.

Aligning agendas: Asking the participants about traditional knowledge allowed them to di-

rectly use their expertise and navigate in familiar territory. Talking about *Kundebi* and *Kunbalak* gave a sense of clarity regarding our function in Aboriginal land because of the continuation between previous conversations and the current activity. Yet the extent of our contribution being seen as beneficial for Bininj people from a language preservation perspective is still unclear. Writing in language is not a traditional practice in this community. People are often literate in English but not in their language. We then needed to find a space where the orthography made sense (Lewis and Simons, 2016). Documents written in Kunwinjku exist in Bininj country through the ranger program, the schools or in facilities where exists an interaction between Bininj and westerners (art centres, clinics, etc.). While we thought that the proposed activity could enable the continuation of the creation of these resources by Aboriginal participants, the proposed app has probably mainly been seen as a way to enhance writing skills.

App design: There were two main challenges related to the design of the app. The first one was enabling syllable concatenation, prioritising information from the oral space. Then we needed to efficiently retrieve syllables that had not been spotted. The first challenge was easily solved by the syllable playback features possible with the progressive collection of syllables throughout this project. Then we designed a basic search mechanism. The first search mechanism to add new syllables relied on the keyboard, which we knew was problematic (cf. Section 4.1). We believe that the new design would lead to better efficiency, but it could not be properly tested.

Activity flaws: The lack of good quality data available in Kunwinjku did not allow us to build a robust speech synthesis system that would have been relevant to the interface. Instead, we recorded in isolation syllables which sometimes lacked clarity. While ultimately, some of the most common syllables have been recorded by a native speaker, many were still pronounced by the first author, whose pronunciation might not be accurate. For instance, in the pilot study, while writing the word *djurra* (IPA djura) "paper", first author's pronunciation of the syllable *rra* has not been accepted by IG and selected instead "*da*" which was closer to the pronunciation of the word according to him. The case of the glottal stop has also been mentioned as a challenge in the literature (Wigglesworth et al., 2021). The glottal stops included in some syllables were not clearly audible out of context, which made them hard to differentiate from similar syllables without glottal stops (*ma* and *mah*, for instance).

Limitations: There is a limited number of Kunwinjku speakers, and recruiting a large number of participants for such work was not easy. The current pandemic did not facilitate our work, and we know that it is hard to draw final conclusions with activities conducted with only three participants. Further research needs to be done, including proper testing in Bininj country to consolidate our observations. The activity setup was also grounded for JB and GB in an academic environment with access to facilities that we do not necessarily have access to in remote locations (access to the internet, workplaces etc...). Besides, we can ask ourselves about the sustainability of such a work grounded in an interaction between Aboriginal participants and scholars in a very controlled environment. To be sure that our methods can be used in the long term, we imagine setting up a remote server to enable remote access on tablets so that people can keep interacting with the app without outside intervention.

7 Conclusion

Generic speech recognition methods for under-resourced languages offer the potential to support small speech communities. Yet the translation of such methods into community-based projects is rare. We have presented a study on the creation and testing of a syllable spotting-based transcription interface to enable the creation of written resources by the members of an Aboriginal community in the Australian Top End. Based on the challenges encountered in previous work, we went from word spotting to syllable spotting to reach a denser transcription and enabled a transcription method closer to existing practices. With the help of collaborators, we designed a transcription interface that allowed the users to build the transcription of given audio using the syllable spotted by our system. We reported the testing of the app with three participants at different stages of development, including lessons learnt from their interaction with the transcription activity and the app design.

Research guidelines push scholars to decolonise their practices and to go towards self-determination. Yet the translation of guidelines to real-life applications is unclear, specifically in cross-cultural collaborations with different ways of knowing. This

work allowed us to highlight methodological paths that improved the engagement and comprehension of the participants. The activity sheet, for instance, made clear that the activity was about creating a wordlist which was not necessarily clear based on our explanation. Dividing the activity between an elicitation part and a transcription part allowed us to hook the interest of the participants with a task they were familiar with and allowed us to clarify the context of our work in contrast to the sparse transcription of random sentences explored previously (Le Ferrand et al., 2022). All participants frequently used the playback of the syllables in isolation and their concatenation, confirming its engaging aspect.

Documentary linguistics has often been undertaken by non-indigenous linguists where the collaboration with the community did not go further than the collection of spoken data (First Languages Australia, 2014). In this work, we initially wanted to counterbalance these practices by enabling community-based language documentation. Yet keeping a language strong does not need to be about language documentation, and Bininj people who took part in this work did not seem to buy into documentary linguistics practices. Instead, they seemed to see the interface as a literacy learning tool. Keeping language strong is seen as building capabilities instead of creating and storing language material. Community-based implies an active role of the community in the work we conducted, and following their view in terms of language work is then crucial. The cross-cultural challenges we encountered required extra work to enable a common ground we could build on. Now that comprehension issues are solved, that we have a better comprehension of people agenda and COVID-19 restrictions start to be eased, more iteration can happen to allow the community to take control of the design of the proposed tool to better fit their agenda and practices.

8 Acknowledgements

We are grateful to the Bininj people of Northern Australia for the opportunity to work in their community, and particularly to artists at Injalak Arts and Craft (Gunbalanya), so as the Bininj and Daluk from Mamadawerre and Kabulwarnamyo who spent time helping us in the university facilities. Our thanks to several anonymous reviewers for helpful feedback on earlier versions of this paper. The final version of the transcription building app has been co-designed by Dr. Cat Kutay, Lecturer at Charles Darwin University and Melko Nguyen as part of her final master's project.

This research was covered by a research permit from the Northern Land Council, ethics approved from CDU and was supported by the Australian government through a PhD scholarship, and grants from the Australian Research Council and the Indigenous Language and Arts Program. All the participants have been paid at the regular rate for Aboriginal people consultancy.

References

Oliver Adams, Trevor Cohn, Graham Neubig, Hilaria Cruz, Steven Bird, and Alexis Michaud. 2018. Evaluating phonemic transcription of low-resource tonal languages for language documentation. In *LREC 2018 (Language Resources and Evaluation Conference)*, pages 3356–3365.

Gilles Adda, Sebastian Stüker, Martine Adda-Decker, Odette Ambouroue, Laurent Besacier, David Blachon, Hélene Bonneau-Maynard, Pierre Godard, Fatima Hamlaoui, Dmitry Idiatov, et al. 2016. Breaking the unwritten language barrier: The bulb project. *Procedia Computer Science*, 81:8–14.

Antonios Anastasopoulos, Marika Lekakou, Josep Quer, Eleni Zimianiti, Justin DeBenedetto, and David Chiang. 2018. Part-of-speech tagging on an endangered language: a parallel Griko-Italian resource. In *Proceedings of the 27th International Conference on Computational Linguistics*, pages 2529–2539.

Alexei Baevski, Wei-Ning Hsu, Alexis Conneau, and Michael Auli. 2021. Unsupervised speech recognition. *Advances in Neural Information Processing Systems*, 34:27826–27839.

Steven Bird. 2022. Local languages, contact languages, and other high-resource scenarios. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics*, pages 7817–7829.

Tommaso Caselli, Roberto Cibin, Costanza Conforti, Enrique Encinas, and Maurizio Teli. 2021. Guiding principles for participatory design-inspired natural language processing. In *Proceedings of the 1st Workshop on NLP for Positive Impact*, pages 27–35.

Shobhana Chelliah and Willem De Reuse. 2010. *Handbook of Descriptive Linguistic Fieldwork*. Springer Science & Business Media.

Michael Christie. 2008. Yolngu studies: A case study of aboriginal community engagement. *Gateways: International Journal of Community Research and Engagement*, 1:31–47.

Philippe Descola. 2005. *Par-delà nature et culture*, volume 1. Gallimard Paris.

Lise M Dobrin, Peter K Austin, and David Nathan. 2009. Dying to be counted: The commodification of endangered languages in documentary linguistics. *Language Documentation and Description*, 6:37–52.

Long Duong. 2017. *Natural Language Processing for Resource-poor Languages*. Ph.D. thesis, University of Melbourne.

Steve Etherington. 2006. *Learning to be Kunwinjku: Kunwinjku People Discuss their Pedagogy*. Ph.D. thesis, Charles Darwin University.

Aoife Finn, Peter-Lucas Jones, Keoni Mahelona, Suzanne Duncan, and Gianna Leoni. 2022. Developing a part-of-speech tagger for te reo māori. In *Proceedings of the Fifth Workshop on the Use of Computational Methods in the Study of Endangered Languages*, pages 93–98.

First Languages Australia. 2014. Angkety Map: Digital resource report. Technical report. https://www.firstlanguages.org.au/images/fla-angkety-map.pdf; accessed Oct 2021.

Ben Foley, Josh Arnold, Rolando Coto-Solano, Gautier Durantin, T. Mark Ellison, Daan van Esch, Scott Heath, František Kratochví, Zara Maxwell-Smith, David Nash, Ola Olsson, Mark Richards, Nay San, Hywel Stoakes, Nick Thieberger, and Janet Wiles. 2018. Building speech recognition systems for language documentation: The CoEDL Endangered Language Pipeline and Inference System. In *Proceedings of the 6th International Workshop on Spoken Language Technologies for Under-Resourced Languages*, pages 205–209. ISCA.

Dennis Foley. 2003. Indigenous epistemology and indigenous standpoint theory. *Social Alternatives*, 22(1):44–52.

Tianyu Gao, Adam Fisch, and Danqi Chen. 2021. Making pre-trained language models better few-shot learners. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 3816–3830.

Murray Garde. 2013. *Culture, Interaction and Person Reference in an Australian Language: An Ethnography of Bininj Gunwok Communication*, volume 11. John Benjamins Publishing.

Pierre Godard, Marcely Zanon Boito, Lucas Ondel, Alexandre Berard, François Yvon, Aline Villavicencio, and Laurent Besacier. 2018. Unsupervised word segmentation from speech with attention. In *Proceedings of Interspeech 2018*, pages 2678–2682.

Florian Hanke. 2017. *Computer-Supported Cooperative Language Documentation*. Ph.D. thesis, University of Melbourne.

Dianna Hardy, Elizabeth Forest, Zoe McIntosh, Trina Myers, and Janine Gertz. 2016. Moving beyond "just tell me what to code" inducting tertiary ICT students into research methods with Aboriginal participants via games design. In *Proceedings of the 28th Australian Conference on Computer-Human Interaction*, pages 557–561.

Atticus Harrigan, Timothy Mills, and Antti Arppe. 2019. A preliminary plains cree speech synthesizer. In *Proceedings of the Workshop on Computational Methods for Endangered Languages*, volume 1, pages 64–73.

Robbie Jimerson and Emily Prud'hommeaux. 2018. ASR for documenting acutely under-resourced indigenous languages. In *Proceedings of the Eleventh International Conference on Language Resources and Evaluation (LREC 2018)*, pages 4161–4166.

Roland Kuhn. 2022. The Indigenous Languages Technology Project at the National Research Council of Canada, and its context. In *Language Technologies and Language Diversity*, pages 85–104. Linguapax International.

William Lane and Steven Bird. 2019. Towards a robust morphological analyzer for Kunwinjku. In *Proceedings of the The 17th Annual Workshop of the Australasian Language Technology Association*, pages 1–9.

Éric Le Ferrand, Steven Bird, and Laurent Besacier. 2021. Phone based keyword spotting for transcribing very low resource languages. In *Proceedings of the The 19th Annual Workshop of the Australasian Language Technology Association*, pages 79–86.

Éric Le Ferrand, Steven Bird, and Laurent Besacier. 2022. Learning from failure: Data capture in an australian aboriginal community. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics*, pages 4988–4998.

Tuck Wah Leong, Christopher Lawrence, and Greg Wadley. 2019. Designing for diversity in aboriginal australia: Insights from a national technology project. In *Proceedings of the 31st Australian Conference on Human-Computer-Interaction*, pages 418–422.

M Paul Lewis and Gary F Simons. 2016. *Sustaining Language Use*. SIL International Publications.

Xinjian Li, Siddharth Dalmia, Juncheng Li, Matthew Lee, Patrick Littell, Jiali Yao, Antonios Anastasopoulos, David R Mortensen, Graham Neubig, Alan W Black, et al. 2020. Universal phone recognition with a multilingual allophone system. In *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 8249–8253. IEEE.

Andrew Manakgu. 1996. Kunbalak. *Stories for Kunwinjku young people in mother-inlaw language, ordinary Kunwinjku and English. Kunbarllanjnja (Oenpelli): Kunwinjku Language Centre*.

Alexandra Marley. 2021. "I speak my language my way!"—young people's kunwok. *Languages*, 6(2):88.

Raghav Menon, Herman Kamper, Ewald van der Westhuizen, John Quinn, and Thomas Niesler. 2019. Feature exploration for almost zero-resource ASR-free keyword spotting using a multilingual bottleneck extractor and correspondence autoencoders. *Proceedings of Interspeech 2019*, pages 3475–3479.

Vikramjit Mitra, Andreas Kathol, Jonathan D Amith, and Rey Castillo García. 2016. Automatic speech transcription for low-resource languages-the case of Yoloxóchitl Mixtec (Mexico). In *Proceedings of Interspeech 2016*, pages 3076–3080.

M. Eleanor Nevins. 2013. *Lessons from Fort Apache: Beyond Language Endangerment and Maintenance*. Wiley.

Walter Ong. 1982. *Orality and Literacy: The Technologizing of the Word*. Routledge.

Siddika Parlak and Murat Saraclar. 2008. Spoken term detection for Turkish broadcast news. In *2008 IEEE International Conference on Acoustics, Speech and Signal Processing*, pages 5244–5247. IEEE.

Bernard Perley. 2012. Zombie linguistics: Experts, endangered languages and the curse of undead voices. *Anthropological Forum*, 22:133–149.

Aidan Pine, Dan Wells, Nathan Brinklow, Patrick Littell, and Korin Richmond. 2022. Requirements and motivations of low-resource speech synthesis for language revitalization. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 7346–7359.

Keren Rice. 2011. Documentary linguistics and community relations. *Language Documentation and Conservation*, 5:187–207.

Celeste Rodríguez Louro and Glenys Collard. 2021. Working together: Sociolinguistic research in urban Aboriginal Australia. *Journal of Sociolinguistics*, 25:785–807.

Courtney Ryder, Tamara Mackean, Kate Hunter, Julieann Coombes, Andrew JA Holland, and Rebecca Ivers. 2021. Yarning up about out-of-pocket healthcare expenditure in burns with aboriginal families. *Australian and New Zealand journal of public health*, 45(2):138–142.

Steffen Schneider, Alexei Baevski, Ronan Collobert, and Michael Auli. 2019. wav2vec: Unsupervised pre-training for speech recognition. In *Proceedings of Interspeech 2019*, pages 3465–3469.

Lane Schwartz. 2022. *Primum Non Nocere*: Before working with Indigenous data, the ACL must confront ongoing colonialism. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics*, pages 724–731.

Lane Schwartz, Emily Chen, Benjamin Hunt, and Sylvia LR Schreiner. 2019. Bootstrapping a neural morphological analyzer for st. lawrence island yupik from a finite-state transducer. In *Proceedings of the Workshop on Computational Methods for Endangered Languages*, volume 1, pages 87–96.

Shane Settle, Keith Levin, Herman Kamper, and Karen Livescu. 2017. Query-by-example search with discriminative neural acoustic word embeddings. *Proceedings of Interspeech 2017*, pages 2874–2878.

Alessandro Soro, Margot Brereton, Jennyfer Lawrence Taylor, Anita Lee Hong, and Paul Roe. 2017. A cross-cultural noticeboard for a remote community: design, deployment, and evaluation. In *IFIP Conference on Human-Computer Interaction*, pages 399–419. Springer.

Igor Szoke, Lukás Burget, Jan Cernocky, and Michal Fapso. 2008. Sub-word modeling of out of vocabulary words in spoken term detection. In *2008 IEEE Spoken Language Technology Workshop*, pages 273–276. IEEE.

Jennyfer Lawrence Taylor, Wujal Wujal Aboriginal Shire Council, Alessandro Soro, Michael Esteban, Andrew Vallino, Paul Roe, and Margot Brereton. 2020. Crocodile language friend: Tangibles to foster children's language use. In *Extended Abstracts of the 2020 CHI Conference on Human Factors in Computing Systems*, pages 1–14.

E. Uí Dhonnchadha and J. Van Genabith. 2006. A part-of-speech tagger for Irish using finite-state morphology and constraint grammar disambiguation. In *Proceedings of the Fifth International Conference on Language Resources and Evaluation (LREC'06)*, pages 2241–2244, Genoa, Italy.

Daan van Esch, Ben Foley, and Nay San. 2019. Future directions in technological support for language documentation. In *Workshop on the Use of Computational Methods in the Study of Endangered Languages*, pages 14–22.

Charl Van Heerden, Damianos Karakos, Karthik Narasimhan, Marelie Davel, and Richard Schwartz. 2017. Constructing sub-word units for spoken term detection. In *2017 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 5780–5784. IEEE.

Michael Walsh and Colin Yallop, editors. 1993. *Language and Culture in Aboriginal Australia*. Aboriginal Studies Press.

Matthew Wiesner, Desh Raj, and Sanjeev Khudanpur. 2022. Injecting text and cross-lingual supervision in few-shot learning from self-supervised models. In *ICASSP 2022-2022 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 8597–8601. IEEE.

Gillian Wigglesworth, Melanie Wilkinson, Yalmay Yunupingu, Robyn Beecham, and Jake Stockley. 2021. Interdisciplinary and intercultural development of an early literacy app in Dhuwaya. *Languages*, 6:106.

Liang Xu, Elaine Uí Dhonnchadha, and Monica Ward. 2022. Faoi gheasa an adaptive game for irish language learning. In *Proceedings of the Fifth Workshop on the Use of Computational Methods in the Study of Endangered Languages*, pages 133–138.

Tariq Zaman, Heike Winschiers-Theophilus, Franklin George, Alvin Yeo Wee, Hasnain Falak, and Naska Goagoses. 2016. Using sketches to communicate interaction protocols of an indigenous community. In *Proceedings of the 14th Participatory Design Conference: Short Papers, Interactive Exhibitions, Workshops-Volume 2*, pages 13–16.

Few-Shot Pidgin Text Adaptation via Contrastive Fine-Tuning

Ernie Chang[1], Jesujoba O. Alabi[2,4], David Ifeoluwa Adelani[3,4], and Vera Demberg[1]
[1] Department of Language Science and Technology, Saarland University, Germany
[2] Inria, France
[3] Spoken Language Systems (LSV), Saarland Informatics Campus, Germany
[4] Masakhane NLP
{cychang,vera}@coli.uni-saarland.de
jesujoba.alabi@inria.fr
didelani@lsv.uni-saarland.de

Abstract

The surging demand for multilingual dialogue systems often requires a costly labeling process for each language addition. For low resource languages, human annotators are continuously tasked with the adaptation of resource-rich language utterances for each new domain. However, this prohibitive and impractical process can often be a bottleneck for low resource languages that are still without proper translation systems nor parallel corpus.

In particular, it is difficult to obtain task-specific low resource language annotations for the English-derived creoles (e.g. Nigerian and Cameroonian Pidgin). To address this issue, we utilize the pretrained language models i.e. BART which has shown great potential in language generation/understanding – we propose to finetune the BART model to generate utterances in Pidgin by leveraging the proximity of the source and target languages, and utilizing *positive* and *negative* examples in constrastive training objectives. We collected and released the first parallel Pidgin-English conversation corpus in two dialogue domains and showed that this simple and effective technique is suffice to yield impressive results for English-to-Pidgin generation, which are two closely-related languages.

1 Introduction

Task-oriented dialog systems are becoming prevalent in various daily activities such as ticket booking and restaurant reservations (Peng et al., 2020). In a typical task-oriented dialog system, the natural language generation (NLG) module plays a crucial role by converting semantic representations into responses in natural language (Langkilde and Knight, 1998). As such, NLG plays a significant impact on the users' experience.

Unfortunately, these dialogue systems are mostly built for high resource languages such as English, which makes it less user-friendly in regions where English is not widely-spoken or is spoken differently (Khalil, 2020; Yusupujiang and Ginzburg, 2021). This includes African countries such as Nigeria and Cameroon where the demand for these technologies are growing at a rapid rate (Khalil, 2020). There is then a need to adapt[1]/translate these existing well-developed dialogue systems in high resource languages into its target language counterparts. However, techniques in machine translation are still under-developed as some language pairs are completely devoid of parallel corpus (Adelani et al., 2021; Chang et al., 2020). This situation is further exacerbated by the fact that the targeted NLG requires a domain-specific translation and annotation.

To this end, we present a preliminary study on the domain-specific adaptation of high resource English language model into the English-related Pidgin languages consisting of the Nigerian (Naija) and Cameroonian Pidgin (Yaounde). In particular, we explore on a low resource scenario where the goal is to translate the English utterance into conversation (dialogue) Pidgin sentences. This scenario consists of a few parallel data and a larger quantity of monolingual text in both languages. To facilitate the research, we collected the first parallel English-Pidgin dialogue corpus and release it along with the public Nigerian (Naija) (Ogueji and Ahia, 2019) and Cameroonian Pidgin spoken corpus (Green et al., 2016). We further propose a simple technique that allows to finetune a pretrained English language model into generating Pidgin that can be used to collect dialogue text in dialogue systems. To do so, we employ the technique of contrastive learning where the off-the-shelf English language model adjusts its useful prior knowledge into producing Pidgin, which is a closely-related language (Ayafor, 2008; Faraclas, 2008). This process is enhanced by having the model to discern

[1] We use the word "adapt" here since the translation process can slightly change the content for the regional end users.

Proceedings of the 29th International Conference on Computational Linguistics, pages 4286–4291
October 12–17, 2022.

between positive/negative examples as defined by (1) the in/out-domain English utterances, and (2) the English and Pidgin texts. Overall, we made the following contributions:

1. We release the first parallel data consisting of $\backsim$ 200 Nigerian/Cameroonian Pidgin and English pairs in multiple dialogue domains including the restaurant and drone simulation.

2. We showed the efficacy of the proposed constrastive finetuning technique as being both simple and effective in creating natural Pidgin text with high-fidelity.

2 Related Work

Contrastive learning has been widely used in various tasks – language modeling (Huang et al., 2018), unsupervised word alignment (Liu and Sun, 2015), caption generation (Mao et al., 2016; Vedantam et al., 2017), and machine translation (Yang et al., 2019). Representations are learned with contrasting positive pairs and negative pairs: Chopra et al. (2005); Weinberger and Saul (2009); Schroff et al. (2015) leverage a triplet loss to separate positive examples from negative examples in metric learning. Chen et al. (2020) shows that contrastive learning can boost the performance of self/semi-supervised learning in computer vision tasks.

In natural language processing, contrastive learning has been widely used. (Mikolov et al., 2013) predicts neighbouring words from context with noise-contrastive estimation (Gutmann and Hyvärinen, 2012) while (Logeswaran and Lee, 2018) samples two contiguous sentences for positive pairs and the sentences from other document as negative pairs. Our work is in the same vein where positive/negative examples are provided to enhance cross-lingual sentence representations.

3 Relations of Pidgin to English

Variations of Nigerian Pidgin are spoken across West and Central Africa in countries such as Benin, Ghana, and Cameroon (Faraclas, 2013) as a result of contacts between Africans and English-speaking sailors and traders (Gilman, 1980)[2]. Altogether, these languages evolved into a closely related group of languages whose vocabulary is predominantly English, spoken by Africans in West Africa and by their descendants in the Western Hemisphere (Gilman, 1980).

Importantly, we *postulate that the relatedness of the Pidgin language to English can be utilized* to formulate an effective contrastive learning setup such that an English language model can be readily adapted into its Pidgin variant. This assumption is not unfounded as English is the lexifier language of Pidgin language (Gilman, 1980), thereby consisting of a large amount of loaned English vocabulary. This motivates our use of an English language model as a strong prior to generate Pidgin text.

4 Methodology

In order to adapt the English language model to generate Pidgin dialogue texts, we propose a contrastive learning framework to expose the model to various valid or incorrect output sequences for a given input sentence. Specifically, we train the model in a two-stage process where for each stage its contrastive loss goes below the threshold 0.1:

(1) **Stage 1** is the *domain-targeting phase* where the language model is finetuned to generate in-domain dialogue utterances. As such, the primary source of positive samples are the English dialogue utterances. Further, we include a general-domain, non-conversation English texts. Non-conversation English texts are closer in the embedding space to the dialogue English texts, so they serve as a meaningful source of negative examples that help to pull the embedding projections to separate in/out domain texts.

(2) On the other hand, **stage 2** is the *language-converting phase* that primarily aims to ensure that the model learns to distinguish Pidgin from English sentences. In this phase, all monolingual Pidgin texts are defined to be the positive samples. In order to induce the language model to generate the corresponding Pidgin texts in stage 2, one source of negative samples are thus the randomly sampled English sentences that exclude the dialogue English texts. We obtained these monolingual English sentences from a general domain and force the model to predict them to be negative. Lastly, when available for the setting, we consider the few-shot Pidgin dialogue utterances obtained from the dialogue system can be used as positive samples. We display some examples in Figure 1.

Both stages follow the contrastive learning framework (Chen et al., 2020), where we train the model to learn the representations of the ground

[2]It is also possible that the English of the sailors was itself pidginized before contact with the Africans, as a result of the multilingual nature of the ships' crews.

Stage 1:
+ **(Dialogue English)**: There is a pub Blue Spice in the riverside area.
− **(English)**: Mauritius was voted Vice President.
Stage 2:
+ **(Pidgin)**: Na for di main general hospital for di regional capital, Mekelle dem dey treat sick pipo.
+ **(Dialogue Pidgin)**: One pub Blue Soice dey for riversde area.
− **(English)**: We treat our youth differently because they are just coming up.

Figure 1: Positive (+) and negative (−) examples.

truth sentence by contrasting the positive pairs with the negative pairs. We project the source and target text sequences onto the latent embedding space. Then we maximize the similarity between the pair of source and target sequence; while minimizing the similarity between the negative pairs as follows:

$$\mathcal{L}_{cont}(\theta) = \sum_{i=1}^{N} \log \frac{\exp(\text{sim}(z_x^{(i)}, z_y^{(i)})/\tau)}{\sum_{z_y^{(j)} \in S} \exp(\text{sim}(z_x^{(i)}, z_y^{(j)})/\tau)} \quad (1)$$

$\mathbf{z}^{(i)} = [\mathbf{h}_1^{(i)} \cdots \mathbf{h}_T^{(i)}] \in \mathbb{R}^{d \times T}$ is a concatenation of the decoder hidden states $\mathbf{h}_t^{(i)}$ of the target sentence $y^{(i)}$ across the sequence of length T for sequence S. $\text{Sim}(\cdot, \cdot)$ is a cosine similarity function and τ is the temperature factor set to 0.5.

Sequence-to-sequence Finetuning. To fine-tune the pretrained BART (Lewis et al., 2020) models for generation, we assume a dataset where each example (x, y) is an (English, Pidgin) pair. We train the student model using the standard cross entropy loss:

$$\mathcal{L}_{\text{seq}} = -\sum_{t=1}^{T} \log p(y_{t+1}|y_{1:t}, x) \quad (2)$$

where T in the target sequence length p is the model's predicted probability for the correct word. As such, the total objective for each batch update is given as $\mathcal{L}_{\text{total}} = \mathcal{L}_{\text{seq}} + \mathcal{L}_{\text{cont}}$.

5 Pidgin Dialogue Corpus

To create the Pidgin dialogue corpus, we consider two publicly available English dialogue datasets (i.e. E2E (Novikova et al., 2017) of restaurant domain and DroneParrot[3])) and translate the English utterances into Pidgin. We show the statistics for each datasets in Table 1 and observe that the Pidgin language generally has less characters (e.g. "kontri" as opposed to "country") – owing to the evolution of its make-shift morphology where words are spelled in a reduced way based on its spoken, colloquial form (Gramley and Pätzold, 2004).

	Domain	Split	Naija	Yaounde
Para	E2E	Train	40	-
		Dev	30	-
		Test	30	10
	Drone	Train	40	-
		Dev	30	-
		Test	30	-
Mono	General	Pd	57,549	3,108
		En	57,549	

Table 1: Statistics of the dataset. For each part of the dataset, the number of sentences for both monolingual (*Mono*) and parallel (*Para*) data in two dialogue domains.

While the language tend to have a smaller vocabulary set than its lexifier (i.e. English)[4], Pidgin diverges profusely in terms of syntax. Thus, the difference in word-ordering is where the anticipated challenges come from in terms of utterance adaptation.

6 Experiments

Training Details. We fine-tune the large BART (Lewis et al., 2020) model for 200 steps using an Adam optimizer (Kingma and Ba, 2014) with $\beta_1 = 0.9$, $\beta_2 = 0.999$, 0.1 weight decay, 0.1 dropout, 0.1 attention dropout, 0.1 label smoothing, 6% warmup steps and a learning rate of 3e-5. The final outputs are generated using beam search with a beam size of 3.

Compared Approaches. We compare the proposed technique with contrastive learning setup with various methods including simply using the source as target (**COPY**):

NMT: It is a semi/un-supervised technique that takes sentences from bilingual/monolingual corpora in two different languages and maps them into the same latent space via iterative back-translation (Lample et al., 2018). In addition, we include the *supervised* bidirectional training objectives.

XLM-R: This approach finetunes the pretrained multilingual language model in Conneau et al. (2020) on parallel data following the proposed objectives as in **Ours** as we now discussed.

[3]Dialogue corpus for drone-human communication.

[4]English is the lexifier language of Pidgin where a large set of vocabulary are loaned.

Do.	Method	Un-Naija	Naija	Yaounde	N+Y
E2E	COPY	**16.84**	16.84	0.00	-
	NMT	0.00	28.26	0.00	0.00
	XLM-R	3.57	33.69	2.51	2.65
	BART	2.12	49.86	3.76	3.38
	+Stage-1	7.42	52.42	4.29	3.56
	+Stage-2 (Ours)	8.46	**56.35**	**5.73**	**4.41**
Drone	COPY	0.28	0.28	0.00	-
	NMT	0.00	9.18	0.00	0.00
	XLM-R	1.39	20.71	1.37	1.39
	Ours	**3.28**	**34.62**	**2.63**	**2.55**

Table 2: English to Pidgin translation performance in BLEU-4 with Naija (N) and Yaounde (Y) in both domains (*Do.*).

Ours: As our proposed approach, the pretrained language model BART (Lewis et al., 2020) is finetuned as in §4 where **BART** simply uses the objective $\mathcal{L}_{\text{seq}}$, and **Stage-1** and **Stage-2** are incrementally added.

Experimental Scenarios. To validate our approach, we finetune **BART** with both Naija and Yaounde utterances. In Table 2, **Un-Naija** means that only the monolingual English and Naija texts are provided. **Naija** and **Yaounde** means training on parallel/monolingual data and testing on their respective languages; while **N+Y** means training on both Naija/Yaounde data and testing on Yaounde.

7 Results and Analysis

In Table 2, we display the results for experiments in both unsupervised and few-shot scenarios. We observe that even with an extremely small amount of annotations, the proposed technique (**Ours**) consisting of both training stages outperforms all benchmarks (**NMT** and **XLM-R**) by as much as 28.09 BLEU – which also goes to show that having multilingual representation (**XLM-R**) is not more beneficial. Next, we show the effectiveness of the contrastive learning setup where **BART** without contrastive losses is substantially lower than **BART+Stage-1** by 2.56 BLEU points, and 6.49 points lower than **Ours**. This shows that providing the positive/negative examples can help project the embedding space into more accurate representations for each examples. Further, we observe a consistent lower scores for the Yaounde Pidgin – which corroborates with past findings that indicates its influence from other sources of European languages such as the Portuguese and French (Gilman, 1980). However, when both *Naija* and *Yaounde* corpus are combined, we observe a drop in performance, which suggests that the morphological differences between the two languages (i.e. Naija and Yaounde) are interfering with the latent representations, causing it to performing sub-par.

Model	E2E		Drone	
	Naturalness	Wrong	Naturalness	Wrong
Reference	4.19	0	4.21	0
NMT	4.31	14	4.26	13
XLM-R	4.06	18	4.19	22
Ours	**4.66**	**13**	**4.35**	**9**

Table 4: Human evaluation on the generated Pidgin texts (100 instances) for all models in the few-shot scenario. Annotators were asked to evaluate the *naturalness* (0-5) and *wrong* (i.e. # hallucinated slots w.r.t. the slot-value pairs) of the texts.

Human Evaluations. We also perform human evaluation with two experts who are well-versed in both English and Pidgin and show the results on *naturalness* and *wrong* in Table 4. We display some examples of generation in Figure 3. *Wrong* is a content selection metric that measures how accurate the generated Pidgin texts are adhering to the dialogue semantic frames. We observe that results on both metrics are consistent with the BLEU-4 scores where our proposed approach consistently generate more natural text with fewer mistakes on both *Naija* and *Yaounde*.

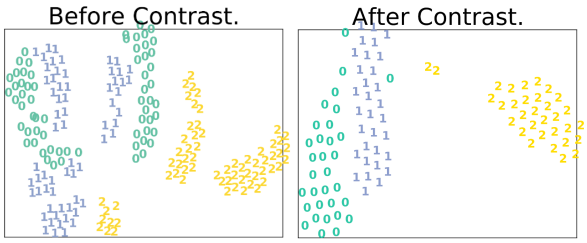

Figure 2: t-SNE projection *before* and *after* contrastive training for dialogue English (0), Naija (1) and general Naija (2).

Embedding Space Separation. To get a clear picture of the sentence representations, we visualize the respective embedding projections in Figure 2. We show the *before* and *after* projections of English and Pidgin utterances, which shows that the clusters of *dialogue English and Pidgin*, and *general Pidgin* sentences are more separated.

8 Limitations

While there are substantial improvements over the translation model baseline, we also observe the limitation of the approach in overcoming extreme low resource conditions as in Yaounde, which does not share much similarities with English. Yaounde is mostly spoken in Cameroon, and therefore evolved differently from Naija Pidgin. This influences how effective the contrastive objectives are in regulating

Reference
If na pub wey get rating of 5 over 5 you wan pick Bue Spice.
Blue Spice be pub for riverside near Rainbow Vegetarian Café.
Ours
Giraffe be pub wey dey riverside near Rainbow Vegetarian Café .
Blue Spice be family friendly pub wey dey riverside near Rainbow Vegetarian Café .
NMT
Blue Spice be family friendly pub wey dey serve Chinese for riverside near Rainbow
One pub near Rainbow Vegetarian Café wey dem dey call Blue Spice.
XLM-R
Blue Spice wey dey serve English food.
One pub wey dem dey call Giraffe wey dey family friendly de

Table 3: Samples of Naija outputs and references in the restaurant domain.

the latent space of the models, which in turn limit the applications of the approach. Moreover, we also notice some discrepancies between the contents of reference and generated outputs, meaning that the models tend to hallucinate fluent, unrelated sentences. This is therefore one other aspect that the approach is unable to address.

9 Conclusion

In this paper, we show that the proposed two-stage training approach can help to adapt the high resource English conversation texts into natural, high-fidelity Pidgin sentences in low resource scenarios. Moreover, we conclude that while Naija and Yaounde are two similar languages, augmenting with either languages provide no further improvements, while separating the representations with contrastive objectives is hugely beneficial.

Acknowledgements

This research was funded in part by the German Research Foundation (DFG) as part of SFB 248 "Foundations of Perspicuous Software Systems". We sincerely thank the anonymous reviewers for their insightful comments that helped us to improve this paper. We also thank Cyrille Vivian Tchakounte and Anuoluwapo Aremu for their involvements in translating the text data we used.

References

David Ifeoluwa Adelani, Jade Abbott, Graham Neubig, Daniel D'souza, Julia Kreutzer, Constantine Lignos, Chester Palen-Michel, Happy Buzaaba, Shruti Rijhwani, Sebastian Ruder, et al. 2021. Masakhaner: Named entity recognition for african languages. *arXiv preprint arXiv:2103.11811*.

Miriam Ayafor. 2008. Cameroon pidgin english (kamtok): Morphology and syntax. In *A handbook of varieties of English*, pages 2101–2120. De Gruyter Mouton.

Ernie Chang, David Ifeoluwa Adelani, Xiaoyu Shen, and Vera Demberg. 2020. Unsupervised pidgin text generation by pivoting english data and self-training. *arXiv preprint arXiv:2003.08272*.

Ting Chen, Simon Kornblith, Mohammad Norouzi, and Geoffrey Hinton. 2020. A simple framework for contrastive learning of visual representations. *International Conference on Machine Learning, ICML*.

Sumit Chopra, Raia Hadsell, and Yann LeCun. 2005. Learning a similarity metric discriminatively, with application to face verification. *2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05)*.

Alexis Conneau, Kartikay Khandelwal, Naman Goyal, Vishrav Chaudhary, Guillaume Wenzek, Francisco Guzmán, Édouard Grave, Myle Ott, Luke Zettlemoyer, and Veselin Stoyanov. 2020. Unsupervised cross-lingual representation learning at scale. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 8440–8451.

Nicholas Faraclas. 2008. Nigerian pidgin english: morphology and syntax. *Varieties of English: Africa, South and Southeast Asia*, 4:340–367.

Nicholas Faraclas. 2013. *Nigerian pidgin*. De Gruyter Mouton.

Charles Gilman. 1980. The origin of cameroonian pidgin dialects. *Anthropological Linguistics*, 22(9):363–372.

Stephan Gramley and Kurt-Michael Pätzold. 2004. *A survey of modern English*. Routledge.

Melanie Green, Miriam Ayafor, Gabriel Ozón, et al. 2016. A spoken corpus of cameroon pidgin english: Pilot study. *Oxford Text Archive Core Collection*.

Michael U Gutmann and Aapo Hyvärinen. 2012. Noise-contrastive estimation of unnormalized statistical models, with applications to natural image statistics. *The journal of machine learning research*.

Jiaji Huang, Yi Li, Wei Ping, and Liang Huang. 2018. Large margin neural language model. *Empirical Methods in Natural Language Processing, EMNLP*.

Shittu Adeshina Khalil. 2020. Artificial intelligence: The impact of chatbots in the nigerian banking sector.

Diederik P Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*.

Guillaume Lample, Alexis Conneau, Ludovic Denoyer, and Marc'Aurelio Ranzato. 2018. Unsupervised machine translation using monolingual corpora only. In *International Conference on Learning Representations*.

Irene Langkilde and Kevin Knight. 1998. Generation that exploits corpus-based statistical knowledge. In *COLING 1998 Volume 1: The 17th International Conference on Computational Linguistics*.

Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Veselin Stoyanov, and Luke Zettlemoyer. 2020. BART: denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. *Annual Meeting of the Association for Computational Linguistics, ACL*.

Yang Liu and Maosong Sun. 2015. Contrastive unsupervised word alignment with non-local features. *Proceedings of the Twenty-Ninth AAAI Conference on Artificial Intelligence,*.

Lajanugen Logeswaran and Honglak Lee. 2018. An efficient framework for learning sentence representations. *International Conference on Learning Representations*.

Junhua Mao, Jonathan Huang, Alexander Toshev, Oana Camburu, Alan L. Yuille, and Kevin Murphy. 2016. Generation and comprehension of unambiguous object descriptions. *2016 IEEE Conference on Computer Vision and Pattern Recognition, CVPR*.

Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. 2013. Distributed representations of words and phrases and their compositionality. *Advances in neural information processing systems*.

Jekaterina Novikova, Ondřej Dušek, and Verena Rieser. 2017. The e2e dataset: New challenges for end-to-end generation. In *Proceedings of the 18th Annual SIGdial Meeting on Discourse and Dialogue*, pages 201–206.

Kelechi Ogueji and Orevaoghene Ahia. 2019. Pidginunmt: Unsupervised neural machine translation from west african pidgin to english. *arXiv preprint arXiv:1912.03444*.

Baolin Peng, Chenguang Zhu, Chunyuan Li, Xiujun Li, Jinchao Li, Michael Zeng, and Jianfeng Gao. 2020. Few-shot natural language generation for task-oriented dialog. *arXiv preprint arXiv:2002.12328*.

Florian Schroff, Dmitry Kalenichenko, and James Philbin. 2015. Facenet: A unified embedding for face recognition and clustering. *Proceedings of the IEEE conference on computer vision and pattern recognition*.

Ramakrishna Vedantam, Samy Bengio, Kevin Murphy, Devi Parikh, and Gal Chechik. 2017. Context-aware captions from context-agnostic supervision. *IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2017*.

Kilian Q Weinberger and Lawrence K Saul. 2009. Distance metric learning for large margin nearest neighbor classification. *Journal of Machine Learning Research*.

Zonghan Yang, Yong Cheng, Yang Liu, and Maosong Sun. 2019. Reducing word omission errors in neural machine translation: A contrastive learning approach. *Annual Meeting of the Association for Computational Linguistics, ACL*.

Zulipiye Yusupujiang and Jonathan Ginzburg. 2021. Data collection design for dialogue systems for low-resource languages. In *Conversational Dialogue Systems for the Next Decade*, pages 387–392. Springer.

Penalizing Divergence: Multi-Parallel Translation for Low-Resource Languages of North America

Garrett Nicolai **Changbing Yang** **Miikka Silfverberg**
University of British Columbia
first.last@ubc.ca

Abstract

This paper explores a special case in multilingual machine translation: so called multi-parallel translation, where the target data for all language pairs are identical. While multi-parallelism offers benefits which are not available in a standard translation setting, translation models can easily overfit when training data are limited. We introduce a regularizer, the divergence penalty, which penalizes the translation model when it represents source sentences with identical target translations in divergent ways. Experiments on very low-resourced Indigenous North American languages show that an initially deficient multilingual translator can improve by 4.9 BLEU through mBART pre-training, and 5.5 BLEU points with the strategic addition of monolingual data, and that a divergence penalty leads to further increases of 0.4 BLEU. Further experiments on Germanic languages demonstrate a improvement of 0.5 BLEU when applying the divergence penalty. An investigation of the neural encoder representations learned by our translation models shows that the divergence penalty encourages models to learn a unified neural interlingua.

1 Introduction

Bilingual neural translation models typically require millions of parallel sentences to achieve adequate quality. A vast majority of the world's languages lack sufficient parallel corpora, and translation efforts for low-resource languages have turned to multilingual methods, leveraging linguistic similarity to augment a deficient signal with plentiful, albeit sometimes noisy, data from related languages (Aharoni et al., 2019; Goyal et al., 2020). In this paper, we explore a very specific multilingual translation setting: multi-parallel translation. Here, models are trained on documents, such as the proceedings of the European Parliament, collections of subtitles, and the Bible. Each sentence has translations in many languages, providing not just a bilingual signal, but one that is bilingual in many directions.

We explore a massively multi-parallel document: the Bible, which has hundreds of translations. While it represents a very restricted domain, the Bible is the only parallel document available for many languages and multi-parallel translation is, therefore, of key importance for low-resource NLP.

Earlier work shows that multi-parallel translation systems can in practice deliver poor results when available training data for individual languages are very small (Mueller et al., 2020). In this setting, the performance of the translation models degrades when the number of source languages is increased. Adapting pre-trained multilingual models such as mBART has also not led to significant progress (Lee et al., 2022). For languages that are not closely-related to the languages in the model, high-quality translation remains an unsatisfied goal.

We hypothesize that this drop in performance is a consequence of an inability of the model to learn a neural interlingua (Johnson et al., 2017), that is, a shared semantic representation for source languages. This prevents knowledge transfer between languages, degrading translation performance. To counteract this tendency, we present the divergence penalty—a modification to the standard loss of a multilingual translation system, which encourages identical encoder and decoder representations for parallel source sentences.

Our experiments on Indigenous North American languages show several avenues for improving the quality of the embedding space. We are able to stabilize the embeddings through mBART pre-training[1] and the addition of monolingual corpora, with gains of up to 5.5 BLEU in the super-low setting. It can be supplemented, however, with our di-

[1] In contrast to Lee et al. We hypothesize that the multi-parallel setting may be responsible, but it is beyond the scope of this paper.

Proceedings of the 29th International Conference on Computational Linguistics, pages 4292–4298
October 12–17, 2022.

vergence penalty, which further coerces the embeddings to adopt interlingual representations. Both visual inspection of t-SNE plots (Hinton and Roweis, 2002) and quantitative examination of learned representations show that the divergence penalty encourages the encoder and decoder representations to cluster according to semantics of the source sentence, regardless of source language.

This is not the first work to explore approaches to strengthening the embedding space of a multilingual translation model. Mullov et al. (2021) use cross-lingual word embeddings. Others have used explicit neural interlinguas (Zhu et al., 2020), and multiple encoders with tied attention (Vázquez et al., 2019). More closely related to our work, Yang et al. (2021) apply an agreement objective which encourages similar representations for artificially code-switched sentence variants and the original sentences. Pan et al. (2021) use contrastive learning to encourage shared representations for semantically similar sentences which resembles our divergence penalty. Finally, Arivazhagan et al. (2019) introduce an auxiliary loss which enforces multilingual similarity to improve zero-shot translation results. All of the aforementioned works, however, investigate translation in a substantially higher-resourced setting and none of them investigate multi-parallel translation.

A second class of related research falls into the broad category of data augmentation for low-resource translation. Sennrich and Zhang (2019) demonstrate that neural translation can learn in low-resource settings without significant modifications, but that these systems can be very sensitive to hyper-parameter tuning. Currey et al. (2017) demonstrate that the expedient method of copying source data to the target can improve low-resource translation quality, while Madaan and Sadat (2020) further leverage back-translations to boost the signal of low-resource translations. Likewise, Rubino et al. (2020) extol the virtues of monolingual data and back-translations in low-resource settings.

2 Methods

In a multi-parallel translation setting, each sentence in the training data has been translated into several languages. A standard multilingual transformer model learns from sentences in isolation, and does not leverage co-dependencies between source sentences in a multi-parallel scenario. We introduce the *divergence penalty* as an auxiliary loss which penalizes models which do not learn similar encoder and decoder representations for parallel sentences. Within each training batch, we identify sentences that are parallel with each other (via their targets), and compute a pairwise cosine comparison of their representation vectors at each position in the sentence. In a true interlingua, these representations would be identical.

The transformer modifies the embedding space at several points, and we compare three variants that utilize a snapshot of the embeddings at a specific point. First, we calculate the cosine distance after context-attention has been applied to the final layer of the encoder (EP). Secondly, we calculate the distance on the output distributions (DP). Finally, we sum the two together (BOTH). Given a sequence $\mathbf{r} = \mathbf{r}_1, ..., \mathbf{r}_n$ of encoder, decoder or joint encoder and decoder representations, the divergence penalty takes the form:

$$\mathcal{L}_{DIV}(\mathbf{r}) = \frac{\sum_{i=1}^{n}\sum_{j=1}^{n}(1 - \mathbf{r}_i^\top \mathbf{r}_j)}{n^2}$$

We then weight this distance (via a tunable hyper-parameter $\alpha_{DIV} \in [0, 1]$), and add it to the batch loss. Batches with parallel sources propagate higher loss if the model has learned divergent representations. Batches that have no multi-parallel sentences see no modification. During training, all parallel translations of the same target sentence are added into the same batch.

Training is performed using the Fairseq (Ott et al., 2019) implementation of transformers, with 3 layers and 4 attentional heads. Embedding dimensions are set at 512, while the feed-forward size is 1024. The model optimizes a label-smoothed cross entropy using Adam(0.9, 0.98), and an inverse square-root learning rate schedule (5e-4 - 1e-9). The model is run for 50 epochs, with the best model chosen via validation loss. These settings closely follow Nicolai et al. (2021). α_{DIV} is tuned for each model, with values in [0, 0.3] typically leading to the best results.

3 Data and Architectures

Our experiments are conducted on Bible data (Nicolai et al., 2021) for three Indigenous language families of North America: Algic, Athabaskan, and Inuit-Aleut. The target language in most of our experiments is English although we also train many-to-many translation systems. Family data sets are constructed by concatenating individual language

Family	Language	Train	Test
Algic	Algonquin	7133	-
	Arapaho	1024	-
	Cree	30269	394
	Mikmaq	7133	394
	Ojibwa	9795	-
	Potawatomi	1870	-
	Siksika	965	-
Algic	Apache	7131	-
	Carrier	8667	-
	Dane-zaa	616	-
	Gwich'in	7132	-
	Navajo	30276	394
	Tlicho	8667	394
	Tsilhqot'in	602	-
Inuit-Aleut	Inuinnaqtun	4289	-
	Inuktitut	30275	394
	Inupiatum	7132	394
	Yupik	30276	394
Germanic	Afrikaans	30249	394
	Bokmål	6719	365
	Danish	30276	394
	Dutch	30216	394
	German	30107	394
	Icelandic	7000	390
	Low German	7116	394
	Nynorsk	6719	365
	Swedish	29870	394
	Swiss German	7120	393

Table 1: The number of Bible verses used for training and testing for different languages. The New Testament consists of approximately 7130 verses, while a full translation is approximately 30275 verses. Numbers are approximate due to verse-splitting techniques.

Bibles, and prepending a language tag. We additionally perform experiments on Germanic Bibles (McCarthy et al., 2020).

Some languages have a full Bible translation available, while most only have a subset. See Table 1 for details. Evaluation is performed on languages with full Bible translations, as well as one ultra low-resource language for each family: Inupiatum (Inuit-Aleut), Mikmaq (Algic), and Tlicho (Athabaskan). For the sake of this paper, we consider "low-resource" to represent languages that have complete Bible translations of 30,000 sentences. "Ultra low-resource" languages are those that have only 7,000 sentences in the New Testament.

For development and testing, we sample each book of the New Testament at a rate of 1% for test, and a further 1% for development. All datasets are segmented using a joint source-target Byte Pair Encoding with a vocabulary size of 16,000. Explorations varying the vocabulary size suggested that this was a reasonable, stable choice for these data sets.[2]

Models are learned using a modified version of the Fairseq (Ott et al., 2019) implementation of transformers. Each model is trained with 3 layers and 4 attentional heads, with an embedding size of 512, and a feed-forward size of 1024. Models are optimized using Adam(0.9, 0.98), and an inverse square root learning schedule starting at 1e-7. Models are trained for a maximum of 50 epochs, with a batch size of 2000 tokens. Dropout is 50%, while attentional dropout is 30%.

4 Experiments

In our experiments, we investigate the performance of different multi-parallel configurations on our Indigenous and Germanic data. We start by training baseline bilingual X-to-English (2L) and multilingual F-to-English systems (M2E), where X is an Indigenous or Germanic language and F is one of the Indigenous language families or the Germanic family. We also continue training on mBART (Liu et al., 2020) for a maximum of 50 epochs.[3]

Indigenous languages Table 2 shows that apart from Mikmaq, Indigenous M2E translators see a sharp decrease in performance from their bilingual analogues–even for ultra low-resource languages. The average BLEU score drops from 10.5 points for bilingual translations models to 6.1 for multilingual models. For Indigenous languages, mBART shows the importance of the language model in multilingual translation. A strong target language model, even in another domain, is enough to learn a model that improves notably over the bilingual one.

Turning to the second sub-table (Raw) in Table 2, we observe that instituting a divergence penalty on the encoder (EP) restores almost all of the quality of the higher-resource languages, and improves ultra low-resource translation performance by 1 BLEU. The decoder penalty (DP) results in a slight 0.2

[2]Our code and datasets are available at `anonymized/for/review`.

[3]Since mBART does not contain any of our Indigenous languages, we tokenize them with the English tokenizer, and use a language identifier to guide the source-to-English translation. Inuktitut is written in its own script, which likely explains its underperformance.

	Baselines				Raw			+E2E			
Lang	2L	M2E	M2M	mBART	EP	DP	BOTH	M2E	+EP	+DP	+BOTH
Cree	17.7	13.7	3.4	**19.4**	13.7	13.7	13.8	16.1	16.1	17.1	16.3
Navajo	12.1	4.1	2.7	12.0	10.0	4.6	10.8	12.6	12.6	**12.9**	12.6
Yupik	12.7	4.1	2.4	16.8	13.6	4.1	13.7	**14.8**	14.3	14.3	14.7
Inuktitut	13.0	3.7	2.6	1.6	14.0	3.7	13.3	14.7	14.7	**15.0**	14.9
Average Full	13.9	6.4	2.8	12.5	12.8	6.5	12.9	14.6	14.4	**14.8**	14.6
Mikmaq	4.2	9.8	3.0	7.9	8.4	9.8	8.4	9.8	9.8	9.6	**10.4**
Tlicho	7.6	3.7	2.5	9.4	9.1	4.2	9.8	11.3	11.3	**12.1**	11.3
Inupiatum	5.9	3.5	2.3	14.4	11.5	4.1	11.8	12.4	12.2	**13.0**	12.3
Average NT	5.9	5.7	2.6	10.6	9.7	6.0	10.0	11.2	11.1	**11.6**	11.3
Average All	10.5	6.1	2.7	11.6	11.5	6.3	11.7	13.1	13.0	**13.4**	13.2

Table 2: Translation results. The baselines compare other data augmentation strategies - 2L is a bilingual model; M2E is plain many-to-English translation; M2M is many-to-many translation, and mBART is the 25 language mBART model. The Raw columns apply the divergence penalties to the encoder (EP), decoder (DP) and a combination of both (BOTH) of the baseline M2E model. +E2E adds in a source-target copy of the English Bible.

BLEU improvement over the initial multilingual model. This makes sense, as the decoder requires a strong encoder representation to learn successful translations. Combining encoder and decoder penalties (BOTH) gives a 5.6 BLEU improvement over the M2E model and a 1.2 BLEU improvement over bilingual models. The improvement over bilingual models for ultra low-resource languages is substantial at 4.1 BLEU. Furthermore, for two of our ultra low-resource languages, BOTH also outperforms mBART, suggesting that while a strong target language model is important, focused embedding space modification can also lead to improvements.

Rather than augment the mBART architecture with our penalty, we instead mimic the language model through the addition of monolingual English-to-English (E2E) examples: from each English target sentence in our training data, we generate a new translation example with identical source and target sentence, and append the example to our multilingual training set.[4] When applying the divergence penalty, English is treated as an additional source language in the training set.

Adding monolingual data mimics mBART, pushing both low and higher-resource languages beyond the divergence penalty alone. Adding the encoder penalty on top of E2E does not improve results. However, the decoder penalty leads to a further average improvement of 0.3 BLEU, while the combination BOTH fares slightly worse. We hypothesize that monolingual data and the encoder penalty behave similarly, anchoring multilingual representations in encoder space and allowing translations to cluster around the English representations. We explore this further in Section 5.

[4]We also experimented with back-translation, but the quality of the back-translated training data were very poor, and hurt model quality, overall. We hypothesize that this is a result of the small size of our training sets, which do not allow us to learn a back-translation model of sufficient quality.

Lang.	2L	M2E	+EP	+DP	+BOTH
Afrikaans	27.4	25.7	25.7	25.9	25.7
Danish	26.5	26.5	26.5	25.9	26.5
German	25.5	25.9	25.9	25.8	25.9
Dutch	26.4	25.9	25.9	26.2	25.9
Swedish	23.7	23.9	23.9	24.7	23.9
Ave. Full	**25.9**	25.6	25.6	25.7	25.6
Swiss German	12.5	21.4	21.4	21.3	21.4
Low German	12.0	19.2	19.2	19.7	19.2
Nynorsk	12.6	22.3	22.3	22.9	22.3
Bokmål	12.2	22.4	22.4	23.2	22.4
Icelandic	11.3	18.9	18.9	19.2	18.9
Ave. NT	12.1	20.8	20.8	**21.3**	20.8
Ave. All	19.0	23.2	23.2	**23.5**	23.2

Table 3: Germanic results.

Germanic languages For our Germanic language experiments, all models are trained with E2E data because this strategy was found to be universally beneficial for Indigenous languages. In contrast to Indigenous languages, none of the Germanic languages see a drop in performance from bilingual to multilingual translation models as demonstrated in Table 3. Multilingual models substantially improve performance for the ultra low-resource Germanic languages where average performance improves by 8.7 points BLEU. For Germanic languages, EP does not result in improvements but DP improves performance for ultra low-resource languages by an average 0.5 points BLEU. A combination of EP and DP does not provide further gains in BLEU.

5 Analysis

To understand how the encoder and decoder penalties affect multi-parallel translation models, we use t-SNE to plot the encoder representations of source

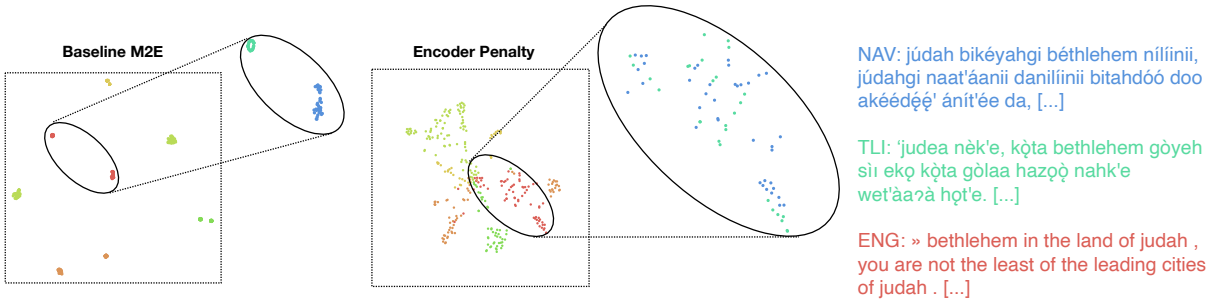

Figure 1: Example t-SNE plots of multilingual encoder representations for baseline M2E models and models with encoder penalty. The square plot represents encoder representations for 4 different sentences - each point represents a single encoder representation, and each sentence is represented by a different color. When we zoom in on the red cluster (representing the sentence shown on the right), we show Navajo representations in light blue and Tlicho in green.

sentences for our baseline multilingual Athabaskan model and a model trained with the encoder penalty in Figure 2.[5] Sentences are color-coded. For example, red dots include representations for one Tlicho and one Navajo source sentence with identical English translations. When we Zoom into the red cluster, showing Navajo representations in light blue and Tlicho in green, we can see that representations for the baseline multilingual system (**Baseline M2E**) form very tight clusters. However, in many cases, these cluster according to source language. This indicates that the model has failed to learn truly multilingual representations, which would instead cluster according to semantics of the source *sentence*, rather than source *language*.

When we add in the encoder penalty (**Encoder Penalty**), we see a significant correction. Tlicho and Navajo representations for equivalent source sentences now intermingle in a joint red cluster. Models supplemented by monolingual data also seem to cluster representations by meaning (included in Appendix A), demonstrating a similar tendency which supports our conclusion that both monolingual data augmentation and the encoder penalty strengthen learning of shared multilingual representations. The decoder penalty does not seem to have a similar effect (see appendix A).

The visual interpretation of t-SNE plots is confirmed via mathematical analysis. We calculate the centroid of each cluster (formed by representations for a particular source language and sentence ID), and determine the average cosine distance between centroids for clusters having the same sentence ID across languages. Formally, let $R(i, l)$ be the set of representations for language l and sentence ID i, and let $\mu(i, l)$ be the centroid of $R(i, l)$. We then compute: $d = \sum_{i=0}^{n} \text{dist}(\mu(i, l_1), \mu(i, l_2))/n$, where dist is cosine distance and n is our number of sentences. We compute these numbers over the entire test set for language l_1 and l_2. Comparing Navajo and Tlicho for the Athabaskan languages family, the baseline M2E model has an average distance of 0.1950, while the EP decreases the value dramatically, to 0.0018.

[5]We first use PCA to project into $\mathbb{R}^{10}$, and then use t-SNE to further project the results to $\mathbb{R}^2$. t-SNE plots ran for a maximum of 150,000 iterations, with perplexity of 30.

6 Conclusion

Multi-parallel translation has the ability to leverage cross-lingual information to supplement a weak low-resource translation signal, but not all languages can benefit. We have introduced a divergence penalty that forces multi-parallel models to learn shared embedding spaces that improve the quality of the translation. On its own, the penalty improves the quality of ultra low-resource Indigenous translation by 4.1 BLEU over a bilingual model, and by more than 4.3 BLEU over a deficient multilingual alternative. Furthermore, monolingual data provides a strong target for multilingual embeddings, but is complemented by our penalty for a further increase of 0.4 BLEU. This trend continues even when the the number of large translation corpora is increased.

References

Roee Aharoni, Melvin Johnson, and Orhan Firat. 2019. Massively multilingual neural machine translation. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 3874–3884.

Naveen Arivazhagan, Ankur Bapna, Orhan Firat,

Roee Aharoni, Melvin Johnson, and Wolfgang Macherey. 2019. The missing ingredient in zero-shot neural machine translation. *arXiv preprint arXiv:1903.07091*.

Anna Currey, Antonio Valerio Miceli-Barone, and Kenneth Heafield. 2017. Copied monolingual data improves low-resource neural machine translation. In *Proceedings of the Second Conference on Machine Translation*, pages 148–156.

Vikrant Goyal, Sourav Kumar, and Dipti Misra Sharma. 2020. Efficient neural machine translation for low-resource languages via exploiting related languages. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics: Student Research Workshop*, pages 162–168.

Geoffrey Hinton and Sam T Roweis. 2002. Stochastic neighbor embedding. In *NIPS*, volume 15, pages 833–840. Citeseer.

Melvin Johnson, Mike Schuster, Quoc V Le, Maxim Krikun, Yonghui Wu, Zhifeng Chen, Nikhil Thorat, Fernanda Viégas, Martin Wattenberg, Greg Corrado, et al. 2017. Google's multilingual neural machine translation system: Enabling zero-shot translation. *Transactions of the Association for Computational Linguistics*, 5:339–351.

En-Shiun Annie Lee, Sarubi Thillainathan, Shravan Nayak, Surangika Ranathunga, David Ifeoluwa Adelani, Ruisi Su, and Arya D McCarthy. 2022. Pre-trained multilingual sequence-to-sequence models: A hope for low-resource language translation? *arXiv preprint arXiv:2203.08850*.

Yinhan Liu, Jiatao Gu, Naman Goyal, Xian Li, Sergey Edunov, Marjan Ghazvininejad, Mike Lewis, and Luke Zettlemoyer. 2020. Multilingual denoising pre-training for neural machine translation. *Transactions of the Association for Computational Linguistics*, 8:726–742.

Pulkit Madaan and Fatiha Sadat. 2020. Multilingual neural machine translation involving Indian languages. In *Proceedings of the WILDRE5–5th Workshop on Indian Language Data: Resources and Evaluation*, pages 29–32.

Arya D McCarthy, Rachel Wicks, Dylan Lewis, Aaron Mueller, Winston Wu, Oliver Adams, Garrett Nicolai, Matt Post, and David Yarowsky. 2020. The Johns Hopkins University Bible corpus: 1600+ tongues for typological exploration. In *Proceedings of The 12th Language Resources and Evaluation Conference*, pages 2884–2892.

Aaron Mueller, Garrett Nicolai, Arya D. McCarthy, Dylan Lewis, Winston Wu, and David Yarowsky. 2020. An analysis of massively multilingual neural machine translation for low-resource languages. In *Proceedings of the 12th Language Resources and Evaluation Conference*, pages 3710–3718, Marseille, France. European Language Resources Association.

Carlos Mullov, Ngoc-Quan Pham, and Alexander Waibel. 2021. Unsupervised transfer learning in multilingual neural machine translation with cross-lingual word embeddings. *arXiv preprint arXiv:2103.06689*.

Garrett Nicolai, Edith Coates, Ming Zhang, and Miika Silfverberg. 2021. Expanding the JHU Bible Corpus for machine translation of the indigenous languages of north america. In *Proceedings of the Workshop on Computational Methods for Endangered Languages*, volume 1, pages 1–5.

Myle Ott, Sergey Edunov, Alexei Baevski, Angela Fan, Sam Gross, Nathan Ng, David Grangier, and Michael Auli. 2019. Fairseq: A fast, extensible toolkit for sequence modeling. In *Proceedings of NAACL-HLT 2019: Demonstrations*.

Xiao Pan, Mingxuan Wang, Liwei Wu, and Lei Li. 2021. Contrastive learning for many-to-many multilingual neural machine translation. *arXiv preprint arXiv:2105.09501*.

Raphael Rubino, Benjamin Marie, Raj Dabre, Atushi Fujita, Masao Utiyama, and Eiichiro Sumita. 2020. Extremely low-resource neural machine translation for Asian languages. *Machine Translation*, 34(4):347–382.

Rico Sennrich and Biao Zhang. 2019. Revisiting low-resource neural machine translation: A case study. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 211–221.

Raúl Vázquez, Alessandro Raganato, Jörg Tiedemann, and Mathias Creutz. 2019. Multilingual NMT with a language-independent attention bridge. In *Proceedings of the 4th Workshop on Representation Learning for NLP (RepL4NLP-2019)*, pages 33–39.

Jian Yang, Yuwei Yin, Shuming Ma, Haoyang Huang, Dongdong Zhang, Zhoujun Li, and Furu Wei. 2021. Multilingual agreement for multilingual neural machine translation. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 2: Short Papers)*, pages 233–239, Online. Association for Computational Linguistics.

Changfeng Zhu, Heng Yu, Shanbo Cheng, and Weihua Luo. 2020. Language-aware interlingua for multilingual neural machine translation. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 1650–1655.

A Encoder Space Plots

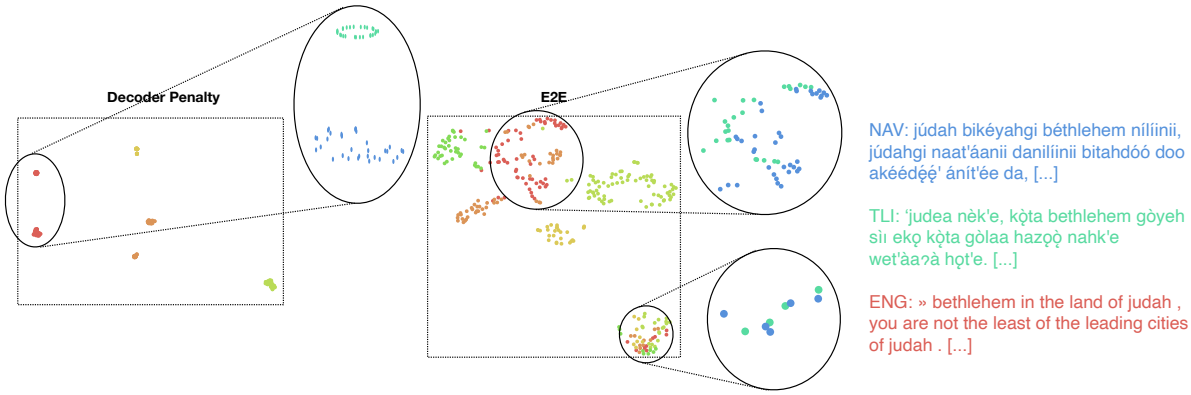

Figure 2: Example t-SNE plots of multilingual encoder representations for M2E models with decoder penalty (on the left) and augmented with English monolingual data (on the right). Each color in the original plot encodes representations for parallel source sentences both in Tlicho and Navajo. When we zoom in on the red cluster, we show Navajo representations in light blue and Tlicho in green. We can see that the decoder penalty does not help the model to learn shared encoder representations. Instead the representations of the Tlicho and Navajo sentences form distinct clusters. When we instead apply mononlingual data augmentation, the representations for the Tlicho and Navajo sentences cluster by meaning and we get shared multilingual representations. For a reason unknown to us, function words, punctuation and language tags form a tight cluster in the lower right corner of the plot when using monolingual data augmentation.

Assessing Digital Language Support on a Global Scale

Gary F. Simons
SIL International
gary_simons@sil.org

Abbey L. Thomas
The University of Texas at Dallas
abbey.thomas@utdallas.edu

Chad K. White
SIL International
chad_white@sil.org

Abstract

The users of endangered languages struggle to thrive in a digitally-mediated world. We have developed an automated method for assessing how well every language recognized by ISO 639 is faring in terms of digital language support. The assessment is based on scraping the names of supported languages from the websites of 143 digital tools selected to represent a full range of ways that digital technology can support languages. The method uses Mokken scale analysis to produce an explainable model for quantifying digital language support and monitoring it on a global scale.

1 Introduction

The users of endangered languages struggle to thrive in a digitally-mediated world. The opportunities afforded by digital technology differ drastically depending on the language being used. This has been dubbed the "digital language divide" (Mikami, 2008; Young, 2015; Soria, 2016; Matsakis, 2019). As digital modes of communicating and accessing information become increasingly necessary in daily life, lack of digital language support (DLS) for a language means that its speakers must use other languages to participate in the global information society or be left out.

Linguists have been writing for decades about the role digital technology could play in language revitalization (Warschauer, 1998; Buszard-Welcher, 2001; Eisenlohr, 2004; Galla, 2009; Holton, 2011; Cru, 2016). Language technologists are recognizing the inequities facing the vast majority of the world's languages (Bird, 2020; Blasi et al., 2022) and are embracing the challenges of bringing greater equity in DLS (Joshi et al., 2019; Bapna et al., 2022; Edunov et al., 2022).

However, in a world where most people are multilingual and each language fits into its functional niche within an ecology of languages (Lewis and Simons, 2016), full digital support for every language is not a realistic goal nor what those multilingual individuals are necessarily looking for (Bird, 2022). The goal of our research is to develop a method for measuring DLS in every language, so that it will be possible to provide an empirical view of the digital state of the world's languages and to observe the progress as so-called low-resource languages move toward crossing the digital language divide.

2 Related Work

Our primary inspiration has been the seminal work by Kornai (2013) on developing a method for assessing the digital vitality of any language. He proposes a four-way classification of languages as digitally Thriving, Vital, Heritage, or Still, "roughly corresponding to the amount of digital communication that takes place in the language." His method harvests data from the Web, then uses supervised classification to automatically label all known languages. In practice, he adds a fifth level, Borderline, to represent languages that show signs of crossing the gap from Still to Vital. He and his colleagues have applied this method to the languages of India (Kornai and Bhattacharyya, 2014), the former Soviet Union (Kornai, 2015), and the Uralic family (Acs et al., 2017).

In reviewing Kornai's method, Gibson (2015, 2016) focused on the huge gap between Still and Vital. He argues that two additional levels are needed to fill this gap: one for when the needed elements (like a keyboarding solution) are in place for potential digital language use, and another for when digital language use is indeed taking off. We follow Gibson's lead in adding two levels, but use names that achieve better congruence with the geometry of the S-curve model that emerges from our method (see Figure 3).

3 Requirements

Following Kornai's (2013) lead, we seek to develop an automated method for assessing digital

Proceedings of the 29th International Conference on Computational Linguistics, pages 4299–4305
October 12–17, 2022.

language vitality that is based on feature data harvested from the Web. In this way, it can be run periodically to monitor changes in digital vitality for every language. We were motivated to develop an alternative to Kornai's method of analysis in order to meet three requirements:

Digital vitality should be orthogonal to non-digital vitality. We exclude features like population and language vitality from the feature data. Kornai notes that the EGIDS level as reported in Ethnologue (Lewis and Simons, 2010; Eberhard et al., 2022) is "the best predictor of digital status." But digital vitality is distinct from non-digital vitality. For instance, our method reports the "dead" language Latin to be the 80th most digitally vital language in the world. By contrast, Aimaq with nearly two million speakers is found to be digitally Still.

The assessments should be explainable. A standard critique of machine learning models based on black-box methods is that the models cannot explain why they produce the answers they do (Arrieta et al., 2020; Miller, 2019). Kornai (2013) bases his results on the majority outcome from 100 runs of a black-box model that yields a slightly different result each time. Users will be more likely to trust results if they are deterministic and explainable.

The assessment scale should measure a single underlying trait. The data features used by Kornai (2013) covered a variety of digital uses. Some had to do with quantifying the extent to which the language has been documented in digital archives by researchers. Others, like the sizes of Wikipedias, had to do with quantifying the extent of digital language use by the language community itself. Still others looked at specific software products and recorded which languages they support. These strike us as three distinct traits, each of which should be assessed in its own right: digital language preservation, digital language use (DLU), and digital language support (DLS). Of these, the latter two are what speak to monitoring the digital vitality of a language as it moves toward crossing the digital language divide. DLU and DLS are distinct traits that should be assessed separately—speakers of unsupported languages may nevertheless use it digitally (for instance, making do in texting; see Eberhard and Mangulamas (2022)), while speakers of supported languages may choose to use digital resources in another language they know.

We have chosen to focus on DLS since the data for monitoring that phenomenon are openly accessible—the developers of digital tools are usually keen to advertise all of the languages they support. By contrast, data on actual digital use is typically not shared on a language-by-language basis by the vendors concerned. A comparable effort to assess DLU on a global scale is much needed, though we anticipate that it will be significantly harder to acquire the needed data.

4 Methodology

The method we have adopted for building an explainable model of DLS is Mokken scale analysis (Mokken, 1971; Schuur, 2003). Mokken's method is a generalization of the more widely known Guttman scaling (Guttman, 1950). In the latter, the items in a scale form a strict hierarchy. If a subject has an item on the scale, then all lower items also apply. A subject's score on the scale is thus the highest item that is true for the subject.

Intuitively, DLS has these properties. If a language has a good virtual assistant (like Siri), then we can infer that it also has good machine translation—but having good machine translation does not imply having a good virtual assistant. Similarly, if a language has good machine translation, we can guess that it must also have good spell checking, though we cannot assume that the reverse would hold. In a Guttman scale, an exception to the hierarchical ordering is considered an error, but in an arena like DLS we can expect there to be exceptions. Mokken scaling is a method for placing the items of a supposed hierarchical scale into their optimal order, while providing metrics that allow one to evaluate how well the hierarchical model fits.

4.1 Categories of Digital Language Support

The method uses the following seven categories of DLS. They are listed below from easiest (most commonly supported) to hardest (least commonly supported) as determined by the results of our analysis:[1]

- Content — A service offering content in many languages (like Wikipedia, news sites, or Bible sites)[2]

[1] This aspect of the analysis is explained in subsection 5.2 and illustrated in Figure 2.

[2] Having digital content in a language could also be viewed as an evidence of digital language use. We treat the fact that a service offers content in a language as a Boolean indicator of

- Encoding — A system component for representing languages (like keyboards and fonts)
- Surface — A tool with surface-level processing (like spell checking or stemming)
- Localized — A tool with a localized user interface (like operating system, browser, or messaging)
- Meaning — A tool with meaning-level processing (like machine translation)
- Speech — A tool for speech processing (like speech-to-text or text-to-speech)
- Assistant — An intelligent virtual assistant (like Siri or Alexa)

For each category, we sought to identify the top ten tools of its kind globally. In order to ensure that we included the major tools in use outside the English-speaking world, we also included the top five tools in each of the ten most populous countries of the world.[3] The reference authority for these rankings was the *similarweb* service.[4] Then we added any tools found from other sources that supported more than 10% of the median number of languages supported by the top tools in the category. In order for a tool to be used in our analysis, we required there to be a URL from which the names or ISO 639 codes of supported languages could be scraped.

The full sample consists of URLs for 143 digital tools across the seven categories of DLS.[5] The number of tools in each category is shown in Table 1 as the maximum number in the range for level 4.

4.2 Harvesting the feature data

The method works by scraping each URL in the sample to discover what languages each tool supports. The harvested language names are mapped to their corresponding ISO 639-3 code[6] by means of a manually maintained table of name-to-code mappings. After the mapping of the harvested language names, the resulting feature data is a logical matrix with rows for 7,829 ISO 639-3 codes, columns for the 143 digital tools, and a Boolean value at the intersection indicating whether the given language is supported by the given tool.

4.3 Scoring the DLS categories as subscales

When a language is not supported by any tools in a given DLS category it is scored as 0; otherwise, the number of tools supporting that language is converted to a level score on a four-level subscale. The correspondence between the number of tools supporting the language and the level on the subscale is shown in Table 1. The score corresponds to the quartile in the distribution of the number of tools supporting each language; only the languages that are supported by at least one tool in the category are included in that distribution.[7]

Category	Levels			
	1	**2**	**3**	**4**
Assistant	1	2	3–4	5–11
Speech	1	2–3	4–8	9–23
Meaning	1	2	3–6	7–14
Localized	1	2	3–12	13–47
Surface	1	2	3	4–15
Encoding	1	2	3	4–10
Content	1	2	3	4–23

Table 1: Number of tools supporting a language in each level of the subscales for the DLS categories

4.4 From category levels to scale items

In constructing the Mokken scale, the levels of the categories become items in the scale. These items are named Content1, Content2, and so on. Within each subscale, the items form a strict hierarchy, in which being scored at a higher level on the subscale implies also having at least as much support as the lower levels of the same subscale. Thus the count of languages for item Content3 also includes the languages for Content4, and so on going down. The bar graph in Figure 1 shows the items listed from top to bottom in ascending order of the number of languages with at least that level of support in the named category.

5 Results

5.1 Evaluating fit of the model

Mokken scale analysis allows us to evaluate the degree to which the scale depicted in Figure 1 forms

support for the language. To measure digital language use, we would quantify the amount of digital content in each language.

[3]This sampling method allows us to discover widely-used tools that support just one large language, but it admittedly misses tools that have been custom-built for a single smaller language.

[4]https://similarweb.com

[5]A complete list of the 143 digital tools is provided at https://github.com/sil-ai/dls-results.

[6]https://iso639-3.sil.org/code_tables

[7]The quartile boundaries are extended upward to accommodate ties; thus in every case, Level 1 contains more than 25% of the languages with that kind of support.

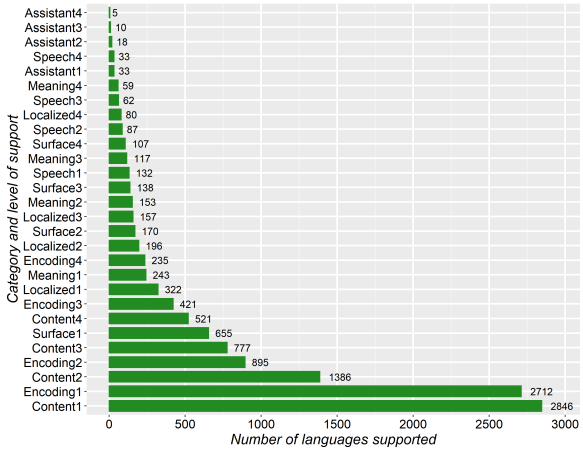

Figure 1: Number of languages supported at each category and level of digital language support

a hierarchical scale. This is done using Loevinger's (1948) coefficient of homogeneity, H.[8] H compares the actual Guttman errors to the expected number of errors if the items were not related in a scale. A value of 1.0 indicates no errors; any value above 0.5 is indicative of a strong scale (Sijtsma and Molenaar, 2002).

Item	H
Assistant	0.987
Speech	0.942
Meaning	0.920
Localized	0.924
Surface	0.885
Encoding	0.707
Content	0.685
Full scale	0.825

Table 2: Coefficient of homogeneity, H, for DLS scale

The results in Table 2 show that the proposed DLS scale is a very strong scale, especially among the categories of support that are hardest to achieve. Thus the total score on all 7 categories (i.e., 0 to 28) serves to quantify the DLS for a given language.

5.2 Relative difficulty of DLS items

Mokken analysis is based on Item Response Theory (IRT)—a methodology developed for educational and psychological testing (Lord, 1980). In IRT, logistic regression is used to derive an Item Response Function (IRF) for each test item; it returns the probability that a subject would produce a positive (or correct) response on that item, given their total score on the rest of the test items. The difficulty of an item is defined as the score (on the rest of the test) at which the subject has a 50% chance of giving a positive response for the item. Figure 2 plots the difficulty for each of the scale items listed in Figure 1. For instance, a language has a 50% chance of getting its first spell-checker (Surface1) if it has 3.6 other DLS items, but the first virtual assistant (Assistant1) cannot be expected until it has 23.4 other DLS items.

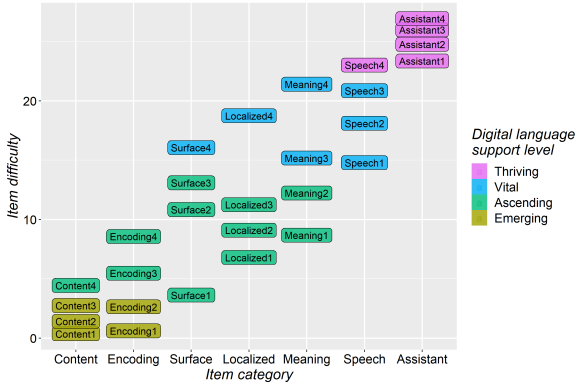

Figure 2: Difficulty of the DLS categories and levels

5.3 DLS as a growth curve

Figure 3 plots the DLS score for 7,829 ISO 639 languages. The vertical axis is the measure of DLS as a proportion: the DLS score achieved divided by the maximum possible score.[9] The horizontal axis is the rank of the language by DLS score, but converted to a log scale and flipped so that lowest DLS is on the left and highest is on the right.

The pattern that emerges is an S-curve as is typical in studies of growth in innovation. We follow the geometry of the fitted curve to assign each language to one of the five summary levels:

- Still — a score of 0
- Emerging — at the bottom where the slope is more horizontal than vertical
- Ascending — below the midpoint where the slope is more vertical than horizontal
- Vital — above the midpoint where the slope is more vertical than horizontal

[8] We have performed these calculations using the "mokken" package (van der Ark, 2007, 2012) in R (R Core Team, 2022).

[9] The DLS scores are also adjusted by scoring each item as the probability returned by its IRF. In educational testing, scoring each positive response as a probability is a way of controlling for random guessing on questions that are too hard for the subject. In the application to DLS it can control for "random" developments that do not have the underpinnings of the expected lower categories of support, such as when there is a one-time philanthropic gesture by a large company or the potentially unsustainable efforts of a solitary developer.

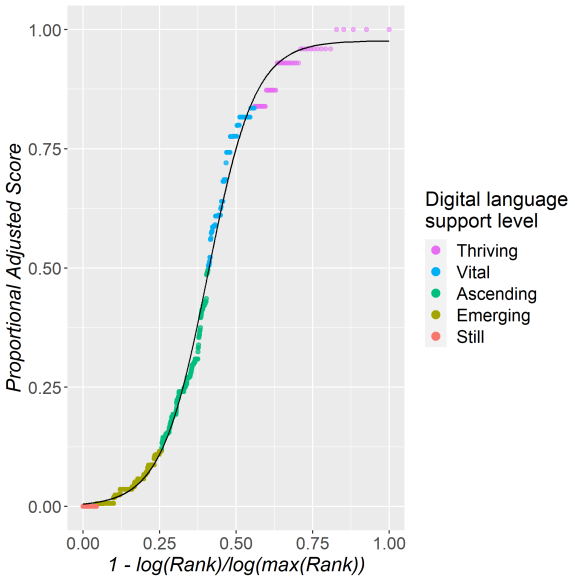

Figure 3: The growth of DLS as a logistic function.

- Thriving — at the top where the slope is more horizontal than vertical

By comparing Figures 2 and 3 one sees what components of DLS correspond to the summary levels.

Table 3 reports the number of languages at each summary level along with the names of example languages, the first being from the upper end of the range and the second from the lower.[10]

Level	Languages	Examples
Thriving	33	English, Hungarian
Vital	95	Nepali, Tongan
Ascending	401	Greenlandic, Hunsrik
Emerging	3304	Dogri, Michif
Still	3996	Aimaq, Yurok

Table 3: Number of languages per DLS level

6 Conclusion

We have presented a method that produces an explainable model for quantifying DLS. We are currently working with Ethnologue to add reporting on DLS in its description of languages, beginning with the next edition. Regularly updating the assessments should serve to document the digital trajectory of every known language.

[10] A sampling of the detailed results produced by the system is provided at https://github.com/sil-ai/dls-results.

Acknowledgements

This research has been funded by the Ethnologue program of SIL International for the purpose of developing a way to add digital language vitality to its reporting on the state of the world's languages. At the outset of the project, SIL International licensed the software developed by Kornai (2013) from the Hungarian Academy of Sciences. We are deeply indebted to Andras Kornai and his PhD student at the time, Katalin Pajkossy, for their help and encouragement as we first replicated his results before developing the method described in this paper. Others who made significant contributions include Steve Moitozo, Erica Oldaker (née Swindle), Steve Woolston, and Daniel Whitenack.

References

J. Acs, K. Pajkossy, and A. Kornai. 2017. Digital vitality of Uralic languages. *Acta Linguistica Academica*, 64(3):327–345.

A. B. Arrieta, N. D. Rodríguez, J. Del Ser, A. Bennetot, S. Tabik, A. Barbado, S. García, S. Gil-Lopez, D. Molina, R. Benjamins, R. Chatila, and F. Herrera. 2020. Explainable artificial intelligence (xai): Concepts, taxonomies, opportunities and challenges toward responsible ai. *Information Fusion*, 58:82–115.

A. Bapna, I. Caswell, J. Kreutzer, O. Firat, D. van Esch, A. Siddhant, M. Niu, P. Baljekar, X. Garcia, W. Macherey, T. Breiner, V. Axelrod, J. Riesa, Y. Cao, M. X. Chen, K. Macherey, M. Krikun, P. Wang, A. Gutkin, A. Shah, Y. Huang, Z. Chen, Y. Wu, and M. Hughes. 2022. Building machine translation systems for the next thousand languages. Technical Report arXiv:2205.03983.

S. Bird. 2020. Decolonising Speech and Language Technology. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 3504–3519, Barcelona, Spain (Online). International Committee on Computational Linguistics.

S. Bird. 2022. Local languages, third spaces, and other high-resource scenarios. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 7817–7829, Dublin, Ireland. Association for Computational Linguistics.

Damian Blasi, Antonios Anastasopoulos, and Graham Neubig. 2022. Systematic inequalities in language technology performance across the world's languages. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 5486–5505, Dublin, Ireland. Association for Computational Linguistics.

L. Buszard-Welcher. 2001. Can the Web help save my language? In L. Hinton and K. Hale, editors, *The Green Book of Language Revitalization in Practice*, pages 331–345. Academic Press, San Diego, CA.

J. Cru, editor. 2016. *Digital Media and Language Revitalisation*. Linguapax Review 2016. Linguapax International.

D. M. Eberhard and M. Mangulamas. 2022. Who texts what to whom and when? Patterning of texting in four multilingual minoritized language communities and a preliminary proposal for the language repertoire matrix. *International Journal of the Sociology of Language*, 2022(276):169–205.

D. M. Eberhard, G. F. Simons, and C. D. Fennig, editors. 2022. *Ethnologue: Languages of the World*, twenty-fifth edition. SIL International, Dallas.

S. Edunov, P. Guzman, J. Pino, and A. Fan. 2022. Teaching AI to translate 100s of spoken and written languages in real time. *Meta AI*.

P. Eisenlohr. 2004. Language revitalization and new technologies: Cultures and electronic mediation and the refiguring of communities. *Annual Review of Anthropology*, 18(3):339–361.

C. K. Galla. 2009. Indigenous language revitalization and technology: From traditional to contemporary domains. In J. Reyhner and L. Lockard, editors, *Indigenous Language Revitalization: Encouragement, Guidance, and Lessons Learned*, pages 167–182. Northern Arizona University, Flagstaff, AZ.

M. L. Gibson. 2015. A framework for measuring the presence of minority languages in cyberspace. In *Linguistic and Cultural Diversity in Cyberspace. Proceedings of the 3rd International Conference*, pages 61–70, Moscow. Interregional Library Cooperation Centre.

M. L. Gibson. 2016. Assessing digital vitality: Analytical and activist approaches. In *Proceedings of the LREC 2016 Workshop "CCURL 2016—Towards an Alliance for Digital Language Diversity"*, pages 46–51.

L. Guttman. 1950. The principal components of scale analysis. In S.A. Stouffer, editor, *Measurement and Prediction*, pages 312–361. Wiley, New York.

G. Holton. 2011. The role of information technology in supporting minority and endangered languages. In P.K. Austin and J. Sallabank, editors, *The Cambridge Handbook of Endangered Languages*, pages 371–399. Cambridge University Press, Cambridge, UK.

P. Joshi, C. Barnes, S. Santy, S. Khanuja, S. Shah, A. Srinivasan, S. Bhattamishra, S. Sitaram, M. Choudhury, and K. Bali. 2019. Unsung challenges of building and deploying language technologies for low resource language communities. In *Proc. of the 16th International Conference on Natural Language Processing,*, pages 211–219, Hyderabad, India.

A. Kornai. 2013. Digital language death. *PLoS ONE*, 8(10):e77056.

A. Kornai. 2015. A new method of language vitality assessment. In *Linguistic and Cultural Diversity in Cyberspace. Proceedings of the 3rd International Conference*, pages 132–138, Moscow. Interregional Library Cooperation Centre.

A. Kornai and P. Bhattacharyya. 2014. Indian subcontinent language vitalization. In *Proc. 2014 LREC Workshop on Indian Language Data: Resources and Evaluation (WILDRE2)*, pages 24–27.

M. P. Lewis and G. F. Simons. 2010. Assessing endangerment: Expanding Fishman's GIDS. *Revue Roumaine de Linguistique*, 55(2):103–120.

M. P. Lewis and G. F. Simons. 2016. *Sustaining language use: Perspectives on community-based language development*. SIL International, Dallas.

J. Loevinger. 1948. The technic of homogeneous tests compared with some aspects of "scale analysis" and factor analysis. *Psychological Bulletin*, 45(6):507.

F. M. Lord. 1980. *Applications of Item Response Theory to Practical Testing Problems*. Lawrence Erlbaum Associates, Mahwah, NJ.

L. Matsakis. 2019. Bridging the Internet's Digital Language Divide. *Wired, 13 June 2019*.

Y. Mikami. 2008. Digital language divide: Measuring linguistic diversity on the Internet. In *UNESCO/UNU Conference on Globalization and Languages: Building on our Rich Heritage*, pages 27–28, Tokyo, Japan.

T. Miller. 2019. Explanation in artificial intelligence: Insights from the social sciences. *Artificial Intelligence*, 267:1–38.

R. J. Mokken. 1971. *A Theory and Procedure of Scale Analysis: With Applications in Political Research*, volume 1. Walter de Gruyter.

R Core Team. 2022. *R: A language and environment for statistical computing*. Vienna, Austria.

W. H. Schuur. 2003. Mokken scale analysis: Between the Guttman scale and parametric item response theory. *Political Analysis*, 11(2):139–163.

K. Sijtsma and I. W. Molenaar. 2002. *Introduction to nonparametric item response theory*. Sage, Thousand Oaks, CA.

C. Soria. 2016. What is digital language diversity and why should we care? In J. Cru, editor, *Digital Media and Language Revitalisation. Linguapax Review 2016*, pages 13–28. Linguapax International.

L. A. van der Ark. 2007. Mokken scale analysis in R. *Journal of Statistical Software*, 20(11):1–19.

L. A. van der Ark. 2012. New developments in Mokken scale analysis in R. *Journal of Statistical Software*, 48(5):1–27.

M. Warschauer. 1998. Technology and indigenous language revitalization: Analyzing the experience of Hawai‘i. *Canadian Modern Language Review*, 55:140–161.

H. Young. 2015. The digital language divide: How does the language you speak shape your experience of the internet? *The Guardian, 28 May 2015*.

Persian Natural Language Inference: A Meta-learning approach

Heydar Soudani*
Sharif University of Technology
heydars@ce.sharif.edu

Mohammad Hassan Mojab*
Sharif University of Technology
mhmojab@ce.sharif.edu

Hamid Beigy
Sharif University of Technology
beigy@sharif.edu

Abstract

Incorporating information from other languages can improve the results of tasks in low-resource languages. A powerful method of building functional natural language processing systems for low-resource languages is to combine multilingual pre-trained representations with cross-lingual transfer learning. In general, however, shared representations are learned separately, either across tasks or across languages. This paper proposes a meta-learning approach for inferring natural language in Persian. Alternately, meta-learning uses different task information (such as QA in Persian) or other language information (such as natural language inference in English). Also, we investigate the role of task augmentation strategy in forming additional high-quality tasks. We evaluate the proposed method using four languages and an auxiliary task. Compared to the baseline approach, the proposed model consistently outperforms it, improving accuracy by roughly six percent. We also examine the effect of finding appropriate initial parameters using zero-shot evaluation and CCA similarity. Our code is publicly available at https://github.com/HassanMojab/MetaNLI.

1 Introduction

In natural language processing (NLP), the goal is to improve models for the processing and production of human languages. As part of NLP, several tasks are defined, each covering a different level of natural language understanding. Meanwhile, natural language inference (NLI) is considered an appropriate and rigorous measure of language comprehension. This task requires recognizing the consequences of natural language sentences, which indicates how well it understands the language (MacCartney, 2009).

NLI aims to determine the inferential relationship between a premise p and a hypothesis h. The problem involves a three-class classification in which every pair (p, h) falls into one of three categories: entailment, contradiction, and neutral. If the hypothesis can be inferred from the premise, pair (p, h) will be assigned to the entailment class. For a hypothesis that contradicts the premise, pair (p, h) will be assigned to the contradiction and neutral otherwise (Amirkhani et al., 2020).

The Persian language lacks sufficient linguistic resources when it comes to natural language understanding. The lack of data can be addressed by collecting annotated data, but this process is both time-consuming and expensive (Nooralahzadeh et al., 2020). FarsTail (Amirkhani et al., 2020) is currently available for Persian, which is created using the same method as SciTail (Khot et al., 2018). It contains 10,367 samples. Also, ParsiNLU (Khashabi et al., 2021) is created for high-level tasks in Persian, and for NLI, it consists of 2700 samples. As it turns out, this amount of data is too small compared with resource-rich languages (such as English, which has only 550,000 samples in the SNLI (Bowman et al., 2015) dataset).

Researchers have tried to solve the data scarcity problem by using cross-language methods. Recent work on cross-lingual learning has mainly focused on transfer between languages already covered by pre-trained representations (Wu and Dredze, 2019). Nonetheless, these techniques do not readily transfer to low-resource languages in which (1) large monolingual corpora are unavailable for pre-training, and (2) sufficient labeled data is lacking for fine-tuning downstream tasks (Xia et al., 2021).

The results of experimental studies for Persian using different embedding methods including word2vec (Mikolov et al., 2013), fastText (Bojanowski et al., 2017), ELMo (Peters et al., 2018), and BERT (Devlin et al., 2019) and various models, such as, DecompAtt (Parikh et al., 2016), ESIM (Chen et al., 2016), HBMP (Talman et al.,

*Equal contribution

Proceedings of the 29th International Conference on Computational Linguistics, pages 4306–4319
October 12–17, 2022.

2019), and ULMFiT (Howard and Ruder, 2018) is reported in FarsTail (Amirkhani et al., 2020). Although this cross-lingual information sharing has enabled success in various natural language processing tasks, it raises the question of how we can achieve more effective collaborative learning between languages or even between different tasks.

Recently, meta-learning has shown to be effective for a variety of machine learning tasks, including NLP (Koch et al., 2015; Ravi and Larochelle, 2016; Qian and Yu, 2019). This paper uses a meta-learning-based method for learning parameters in the joint space of tasks and languages. Auxiliary languages include English, Spanish, French, and German, while QA is the auxiliary task.

Alternatively, an essential prerequisite for the successful application of meta-learning is a task distribution from which a large number of tasks can be sampled to train the meta-learner. However, in NLP, datasets are usually considered as tasks (Nooralahzadeh et al., 2020; Qi and Du, 2020). There are two main problems with treating entire datasets as tasks. The first problem is overfitting, in which a meta-learner is overfitted to a small number of training tasks since there is only a small number of supervised datasets for each NLP problem. A second concern is that the heterogeneity of NLP datasets may result in learning episodes that lead to memorization overfitting, where a meta-learner ignores the support set and fails to adapt (Murty et al., 2021). To improve the quality and quantity of tasks, we use the DReCa (Murty et al., 2021) approach as our data augmentation strategy.

In this paper, we employ meta-learning algorithms to enhance the Persian NLI task. Our models are evaluated on the FarsTail dataset. Experimental results show that we push Persian NLI accuracy forward by more than 6% and zero-shot accuracy by about 4%, setting a new state-of-the-art result for this task. In summary, the main contributions of our research are:

- We have enabled effective parameter sharing across multiple languages and tasks by providing a meta-learning approach. To the best of the authors' knowledge, this is the first study of the interaction between several languages and tasks at different levels of abstraction to solve a high-level problem in the Persian language. The evaluation results are based on the FarsTail dataset as a reference dataset in the Persian language. The datasets available in the XTREME benchmark (Hu et al., 2020) have also been used for auxiliary languages and tasks.

- We examine a metadata augmentation strategy named DReCa (Murty et al., 2021) that takes as input a set of tasks (entire datasets). We then decompose them to approximate some of the latent reasoning categories underlying these datasets, such as various syntactic constructs within a dataset or semantic categories such as quantifiers and negation.

- We also evaluate the trained model in zero-shot mode, which means that the target language (Persian) data is never used during the training process. This test indicates the generality of the model.

The rest of the paper is arranged as follows: Section 2 briefly describes related work. Section 3 introduces our method, and in section 4, we explain the details of the experimental setup. Section 5 presents practical results. The results analysis and some justification are described in section 6. We conclude the paper and summarize future directions in section 7.

2 Related Work

In this section, we briefly outline related work in three areas. The first area is models based on cross-lingual algorithms. In the second area, we highlight methods based on meta-learning. Finally, we summarize existing data augmentation strategies.

2.1 Models based on Cross-lingual

Cross-lingual learning is a method for transferring knowledge from one natural language to another (Pikuliak et al., 2021). Pre-trained models are one of the most widely used examples of cross-lingual learning. Since these models have achieved good results, so Wu and Dredze (2019) explored the broader cross-lingual potential of mBERT (multilingual BERT) as a zero-shot language transfer model with five NLP tasks, including NLI, covering a total of 39 languages. Also, Wang et al. (2019) provides a comprehensive study of the contribution of different components in mBERT to its cross-lingual ability. In addition, it examines the impact of the linguistic properties of the languages, the architecture of the model, and the learning objectives.

Conneau and Lample (2019) proposed two methods for learning cross-lingual language models, one using monolingual data and the other using parallel data and a new cross-lingual language model objective. Singh et al. (2019) introduced a cross-lingual data augmentation method that substitutes part of the input text with its translation into another language.

Huertas-Tato et al. (2021) designed a new architecture called Siamese Inter-Lingual Transformer (SILT) to align multilingual embeddings for NLI efficiently. The paper points out that transformer models are unable to generalize to other domains and have problems with multilingual and interlinguistic scenarios. A new network has been developed to overcome these weaknesses by combining three parts: a multilingual transformer as pretrained embedding, an alignment matrix to compute the similarity between two sentences, and a multi-head self-attention block to interpret input strings.

Despite the advances that Cross-lingual methods have made, building NLP systems in these settings are challenging for several reasons. First, the target language does not contain sufficient annotated data for effective fine-tuning. Secondly, pre-trained multilingual representations are not directly transferable due to language disparities (Xia et al., 2021). In contrast to these methods, we consider setting up training models simultaneously on multiple languages and tasks.

2.2 Meta-learning

Meta-learning addresses the problem of learning to learn. By examining many learning problems, a meta-learner learns a model (Liu et al., 2020). Specifically, the meta-learner uses a meta training set $\mathbb{MS} = \{(\mathbb{S}_i^s, \mathbb{T}_i^s)\}_{i=1}^{N^s}$, where $(\mathbb{S}_i^s, \mathbb{T}_i^s)$ are the training (support) and test (query) set of the ith learning problem and N^s is the number of learning problems used for training; and a meta test set $\mathbb{MT} = \left\{\left(\mathbb{S}_i^t, \mathbb{T}_i^t\right)\right\}_{i=1}^{N^t}$, where $\left(\mathbb{S}_i^t, \mathbb{T}_i^t\right)$ are the support and query set of the ith test learning problem, while N^t is the number of learning problems used for the test. Given $\mathbb{MS}$, the meta-learner learns how to map a pair $(\mathbb{S}, \mathbb{T})$ into an algorithm that leverages $\mathbb{S}$ to optimally solve $\mathbb{T}$.

Due to the lack of well-defined task distribution, meta-learning has not yet succeeded in NLP, leading to attempts that treat datasets as tasks. An ad hoc task distribution causes problems with quantity and quality. Murty et al. (2021) provide a way to break down heterogeneous tasks such as datasets into a set of appropriate subtasks. With this method, data is transferred to the feature space using a pretrained model. They use k-means to decompose data into k clusters and create tasks by combining these clusters.

Recently, however, the combination of cross-lingual techniques in meta-learning frameworks has also been extensively studied. To train a model for low-resource languages on NLI and QA tasks, Nooralahzadeh et al. (2020) uses the MAML algorithm and auxiliary languages. van der Heijden et al. (2021) study the text documents classification problem in monolingual and multilingual modes, using different algorithms such as, Prototypical Networks (Snell et al., 2017), MAML (Finn et al., 2017), Reptile (Nichol et al., 2018), and ProtoMAML (Triantafillou et al., 2019). Also, Tarunesh et al. (2021) examine the interaction between different languages and tasks to learn an appropriate common feature space.

Additionally, transfer-learning can be helpful for low-resource languages. Xia et al. (2021) introduce a meta-learning-based framework called MetaXL for extremely low-resource languages. MetaXL learns an intelligent representational conversion from several auxiliary languages to the target language, bringing the feature space of these languages closer together for more efficient conversion. The main idea is to use a Representation Transformation network between the main model layers which are trained only with the target language.

To the best of our knowledge, this paper is the first attempt to study meta-learning for solving the NLI problem in the Persian language. Also, we are pioneers in using task-language pairs as meta-learning tasks in the Persian language.

2.3 Task Augmentation

Machine learning algorithms usually assume that the train and test data have the same distribution. In contrast, the meta-learning framework treats tasks as training examples and trains a model to adapt to all of them. Meta-learning also assumes that the training and new tasks are drawn from the same distribution of tasks $p(\tau)$. In NLP, datasets are typically treated as tasks, and meta-learners are then overfitting their adaptation mechanisms. NLP datasets are highly heterogeneous, which causes

many learning episodes to have a poor transfer between their support and query sets, which dissuades meta-learners from adapting (Murty et al., 2021).

To deal with overfitting challenges, Yin et al. (2019) propose a meta-regularizer to mitigate memorization overfitting, but don't study learner overfitting. Rajendran et al. (2020) study task augmentation for mitigating meta-learners overfitting in the context of few-shot label adaptation. SMLMT method (Bansal et al., 2020) creates new self-supervised tasks that improve meta-overfitting, but this does not directly address the dataset-as-tasks problem. In contrast, the DReCa method (Murty et al., 2021) addresses the dataset-as-tasks problem and focuses on using clustering as a way to subdivide and fix tasks that already exist. In this paper, we use DReCa as a task augmentation strategy for our method since it mitigates meta-overfitting without any additional unlabeled data.

	NLI	QA
FA	FarsTail (10.3K)	PersianQA (9K)
EN	XNLI (392k)	—
ES	tr. XNLI (392k)	—
DE	tr. XNLI (392k)	—
FR	tr. XNLI (392k)	—

Table 1: Overview of datasets from a variety of sources. For the NLI task, we use the XNLI dataset for English, and its translated versions (tr.) for Spanish(ES), German(DE), and French(FR) provided in XTREME. For each dataset, the number of training instances is also mentioned.

3 The Proposed Methodology

In our setting, firstly, we prepare a set of task-language pairs to provide meta-learning tasks. Afterward, we sample some tasks in each episode and feed them to the meta-learner. In the rest of this section, we describe the proposed task sampling strategy, the proposed meta-learning algorithm, and the proposed task augmentation strategy.

3.1 The Proposed Task Sampling Strategy

In meta-learning, task selection has a profound impact on model performance. For this reason, we create a queue of tasks first. We can create this queue using different scenarios such as selecting languages for a target task (Gu et al., 2018), selecting tasks for a target language (Dou et al., 2019), and picking from various auxiliary languages and auxiliary tasks. In the meta-training section, we sample some tasks from the queue. Formally, the queue's tasks are represented by $\mathcal{D}$. We need to sample tasks from $\mathcal{M}$, which is a Multinomial distribution over $P_{\mathcal{D}}(i)$s. Thus, we investigate temperature-based heuristic sampling (Aharoni et al., 2019), which defines the probability of any dataset as a function of its size as,

$$P_{\mathcal{D}}(i) = q_i^{1/\tau} / \left(\sum_{k=1}^{n} q_k^{1/\tau} \right) \quad (1)$$

where $P_{\mathcal{D}}(i)$ is the probability of sampling the ith task, q_i is the size of ith task, and τ is the temperature parameter. With $\tau = 1$, tasks are randomly sampled proportionately to their dataset sizes, and with $\tau \rightarrow \infty$, they follow a uniform distribution.

3.2 The Proposed Meta-learning Algorithms

Meta-learning is the process of building a model that can solve a new task with only a few labeled examples by training on a variety of tasks with rich annotations. The key idea is to train the model's initial parameters such that the model has maximal performance on a new task after the parameters have been updated through zero or a couple of gradient steps (Yin, 2020). MAML (model-agnostic meta-learning) (Finn et al., 2017) is one of the most significant algorithms. We describe one episode of the MAML algorithm in Appendix A.1. MAML is quite challenging to train since there are two levels of training. Therefore, we use the following two optimization-based and metric-based meta-learning algorithms in this work.

Reptile (Nichol et al., 2018) is a first-order optimization-based algorithm that moves weights toward a manifold of the weighted averages of task-specific parameters $\theta_i^{(m)}$. It samples training tasks from $p(\mathcal{T}) : \tau_1, \cdots, \tau_i, \cdots, \tau_n$. For each training task, it generates an episode that just contains the support set data. For training task τ_i, let's assume the original parameters θ have gone through m steps of updating and become $\theta_i^{(m)}$ (i.e., $\theta_i^{(m)} = \text{AdamW}\left(L_{\tau_i}, \theta, m\right)$ (2)), then Reptile updates θ as follows (Yin, 2020):

$$\theta \leftarrow \theta + \beta \frac{1}{|\{\mathcal{T}\}|} \sum_{\tau_i \sim \mathcal{M}} \left(\theta_i^{(m)} - \theta \right) \quad (3)$$

Prototypical Networks (Snell et al., 2017) is a metric-based meta-learning algorithm. Prototypical networks learn a metric space in which classification can be performed by computing distances to

prototype representations of each class. In general, they are composed of an embedding network f_θ and a distance function $d(x_1, x_2)$. Using the following equation, the embedding network encodes the support set samples S_c and computes prototypes μ_c per class based on the mean sample encodings for that class.

$$\mu_c = \frac{1}{|S_c|} \sum_{(x_i, y_i) \in S_c} f_\theta(x_i). \quad (4)$$

A Prototypical network classifies a new sample according to the following rule.

$$p(y = c \mid x) = \frac{\exp\left(-d\left(f_\theta(x), \mu_c\right)\right)}{\sum_{c' \in C} \exp\left(-d\left(f_\theta(x), \mu_{c'}\right)\right)} \quad (5)$$

Thus, we define the *distance-based cross entropy* (DCE) loss as follows:

$$\text{Loss}(DCE) = -\log P(y = c \mid x) \quad (6)$$

To ensure that the feature space is robust to noise, we also use the Cross Entropy (CE) loss (more details can be found in Appendix A.3.1).

3.3 The Task Augmentation Strategy

First, we use the dataset-as-tasks strategy, which is the most common method for selecting tasks for meta-learning in NLP applications. Next, we employ DReCa to form additional high-quality tasks. The goal of DReCa is to take a heterogeneous task (such as a dataset) and produce a decomposed set of tasks. Given a training task T_i^{tr}, DReCa first groups examples by their labels and then embeds examples within each group with an embedding function EMBED(.). Concretely, for each N-way classification task T_i^{tr}, it forms groups $g_l^i = \{(\text{EMBED}(x_i), y_i) \mid y_i = l\}$. Then, it proceeds to refine each label group into K clusters via k-means clustering to break down T_i^{tr} into groups $\left\{C^j\left(g_l^i\right)\right\}_{j=1}^{K}$ for $l = 1, 2, \ldots, N$. These cluster groups can be used to produce K^N potential DReCa tasks. Each task is obtained by choosing one of the K clusters for each of the N label groups and taking their union.

4 Experimental Setup

4.1 Datasets

We use FarsTail (Amirkhani et al., 2020) for the target dataset. FarsTail is the only large-scale Persian corpus for the NLI task, with 10,367 samples. The samples are generated from 3,539 multiple-choice questions with the least amount of annotators' interventions or selected from natural sentences that already exist independently in the wild, similarly to the SciTail dataset (Khot et al., 2018).

We also use XTREME (Hu et al., 2020) as an auxiliary dataset. XTREME is a multilingual multitask benchmark consisting of classification, structured prediction, QA, and retrieval tasks. We use this benchmark to prepare NLI data for auxiliary languages. Note that large-scale datasets for NLI were only available in English. However, the authors of XTREME developed a custom-built translation system to get translated datasets for NLI. Furthermore, we consider the QA as an auxiliary task. Therefore, we use PersianQA (Ayoubi and Davoodeh, 2021), which is a Persian reading comprehension dataset for QA containing over 9000 entries. Table 1 summarizes the employed dataset specifications.

4.2 Baselines

On the FarsTail dataset, Amirkhani et al. (2020) presents results of various traditional and deep learning-based methods. According to the results of this paper, the highest test accuracy is obtained by using a translation-based approach, i.e., *Translate-Source* with fastText embeddings. In *Translate-Source*, the Persian-translated MultiNLI training set is combined with FarsTail training data for training an ESIM model. Furthermore, FarsTail's authors reported mBERT fine-tuning results in FarsTail webpage[1]. Therefore, we use these results as baselines.

4.3 Implementation Details

In this study, we aim to compare the effects of meta-learning algorithms on classification accuracy with those of fine-tuning and non-episodic algorithms. To make a fair comparison, we first fine-tune our pre-trained models using training data of the auxiliary task in a non-episodic approach. Afterward, we fine-tune the obtained model using the training data of the target task. In this approach, we use mBERT (Devlin et al., 2019), and XLM-R (Conneau et al., 2020), which are known as the state-of-the-art multilingual pre-trained models, and ParsBERT (Farahani et al., 2021) as a monolingual transformer-based model for the Persian language.

[1] https://github.com/dml-qom/FarsTail

In the meta-learning approach, we use the XLM-R model with output layers tailored for each task and train it with Reptile and Prototypical algorithms. To select the hyperparameters of the Reptile algorithm, we utilize the experiments done in Tarunesh et al. (2021). Appendix A.2 provides further details. The Prototypical algorithm is used only in cross-lingual experiments, and we use Euclidean distance as its distance function. The auxiliary languages are arranged in two scenarios. In the first scenario, support and query set data are generated from auxiliary languages, while in the second scenario, the query set is drawn from both auxiliary and target languages. Detailed information is provided in Appendix A.3.2.

Furthermore, we fine-tune the obtained models on Persian training data using the following two methods. The first method is **non-episodic**, which involves fine-tuning models in batches. The second method is **episodic**, in which episodes are constructed first, and then the models are fine-tuned according to the algorithm used.

5 Results

The meta-learning model is tested on different combinations and configurations of the auxiliary tasks. The accuracy results of the Reptile algorithm are presented in Table 2. In addition to the zero-shot and fine-tuning results, we report the accuracy of another scenario. In this scenario, training data of the target language is placed in the meta-training stage along with other auxiliary tasks and cooperate in a training process. Consequently, this scenario does not involve fine-tuning phase. The results of the mentioned scenario are shown in the last column of Table 2.

Table 3 shows the accuracy scores using the Prototypical Network. In the first section of this table, we generate both support and query sets from Persian language data, without using auxiliary tasks. In the second section of this table, the results of the first multi-lingual scenario (where both the support and query sets are generated from auxiliary languages data) are reported in rows 5 to 12. In rows 13 to 16, we show the results of the second multi-lingual scenario (where the support set is drawn from auxiliary language data and the query set is drawn from both auxiliary and Persian language data). Lastly, we added the DReCa strategy and presented the results in rows 17 to 20.

Additionally, we conducted zero-shot evaluations of both algorithms. Zero-shot results are presented in the first accuracy column of Tables 2 and 3. The confusion matrices of the best-performing models for both Reptile and Prototypical algorithms are also depicted in Appendix A.4.

6 Discussion and Analysis

Table 2 shows that the multi-lingual models are always better than the multi-task models due to the fact that tasks like NLI (which require deeper semantic representations) are more likely to benefit from combining data from different languages. We found that our meta-learned models perform better than baselines and non-episodic models. The reason is that the goal of standard meta-learning is to find a model that generalizes well to a new target task. In addition, we compared two different meta-learning algorithms to evaluate their superiority in this paper. From Tables 2 and 3, we can see that Prototypical performed better than Reptile. It is because Prototypical networks use class representations instead of example representations. Therefore, it finds a suitable representation for each class during the meta-train stage.

As part of another experiment, we combined data from the target language with data from other auxiliary tasks for meta-training. Based on the results of these experiments (last column of Table 2 for Reptile and rows 13 to 16 of Table 3 for Prototypical), the model's accuracy has decreased. This is due to the fact that target language data is small when compared with auxiliary language data. So, unbalanced training data confuses the training process and decreases the model's accuracy. In any case, the cooperation of the target language during the training process is a excellent idea for future work.

As indicated in the last two columns of Table 3, episodic fine-tuning is significantly superior to non-episodic fine-tuning. It demonstrates that episodic training is effective even on single language data and creates a generality in the level of training and test data.

We examined the proximity between the feature spaces of the auxiliary languages and the target language quantitatively and qualitatively. At first, we collect representations of the auxiliary and target languages from non-episodic, Reptile, and Prototypical models. In Fig. 1, we present a 2-component PCA visualization for comparison. We also evaluated the models using a distance metric

Row	Model	Shot	Aux. Tasks	Zero-shot	Non-episodic f.t.	Add NLI-fa in m.t.
			Baselines			
1	Translate-Source*	—	—	—	78.13	—
2	mBERT*	—	—	—	83.38	—
			Non-episodic approach			
3	ParsBERT	—	—	—	74.64	—
4	mBERT	—	—	—	81.95	—
5		—	NLI-en	56.53	81.38	—
6		—	NLI-(en, es, de, fr)	67.88	82.34	—
7	XLM-R	—	—	—	81.97	—
8		—	NLI-en	**69.49**	**86.55**	—
9		—	NLI-(en, es, de, fr)	69.09	84.69	—
			Meta-learning approach			
10	XLM-R	1	NLI-en	64.19	84.31	83.37
11		4		70.96	87.17	86.00
12		8		70.70	87.11	86.65
13		16		71.03	87.43	86.52
14		1	NLI-(en, es, de, fr)	65.17	85.21	83.91
15		4		**72.27**	87.57	85.74
16		8		71.61	**88.35**	**88.22**
17		16		71.22	88.02	87.76
18		1	QA-fa	34.18	81.48	81.58
19		4		34.18	81.38	84.96
20		8		33.79	82.14	83.59
21		16		34.18	82.23	84.70
22		1	NLI-en, QA-fa	46.42	83.53	85.16
23		4		66.02	86.52	86.26
24		8		64.26	86.98	86.46
25		16		46.88	86.52	86.13

Table 2: Average test accuracy of the Reptile algorithm with baselines and non-episodic approach results on the Persian NLI task. The first accuracy column shows results before fine-tuning on the Persian NLI train-set (called zero-shot). In the second accuracy column, we provided results after fine-tuning (f.t.) on the Persian NLI train-sets. The last accuracy column reports results of using the Persian NLI train-set in the meta-training phase (m.t.). The data with * comes from FarsTail's paper and webpage.

Row	Model	Shot	Support	Query	Zero-shot	Non-episodic f.t.	Episodic f.t.
			Without auxiliary tasks				
1	XLM-R	1	NLI-fa	NLI-fa	—	70.38	79.30
2		4			—	81.97	85.22
3		8			—	83.98	84.64
4		16			—	85.29	85.74
			With auxiliary tasks				
5	XLM-R	1	NLI-en	NLI-en	68.10	84.83	86.07
6		4			70.57	86.72	87.50
7		8			70.77	86.72	87.37
8		16			**73.18**	87.76	88.54
9		1	NLI-(en, es, de, fr)	NLI-(en, es, de, fr)	69.15	85.01	85.97
10		4			70.25	86.78	87.63
11		8			71.09	**88.48**	**89.39**
12		16			72.20	88.15	88.28
13		1	NLI-(en, es, de, fr)	NLI-(en, es, de, fr, fa)	—	84.15	85.12
14		4			—	86.33	86.78
15		8			—	86.33	86.46
16		16			—	86.78	87.24
17	XLM-R+ DReCa	8	NLI-en	NLI-en	70.44	87.96	88.87
18		16			71.94	87.24	88.74
19		8	NLI-(en, es, de, fr)	NLI-(en, es, de, fr)	71.16	87.74	88.48
20		16			71.61	87.30	88.22

Table 3: Average test accuracy on the Persian NLI task using Prototypical algorithm with and without auxiliary tasks. The last accuracy column reports results after episodic fine-tuning (f.t.) on the Persian NLI train-set.

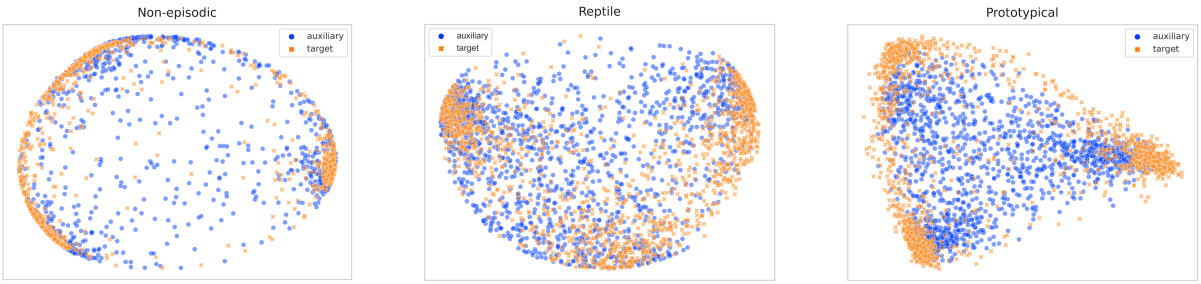

Figure 1: PCA visualization of non-episodic, Reptile and Prototypical models to examine the closeness of the auxiliary and target languages feature spaces.

commonly used in vision and NLP tasks (Huttenlocher et al., 1993; Dubuisson and Jain, 1994; Patra et al., 2019; Xia et al., 2021). Informally, the Hausdorff distance measures the distance between data representations of auxiliary languages and the target language. Given a set of representations of the auxiliary language $\mathcal{S} = \{s_1, s_2, \ldots, s_m\}$ and a set of representations of the target language $\mathcal{T} = \{t_1, t_2, \ldots, t_m\}$ we compute the Hausdorff distance as follows:

$$\max \left\{ \max_{s \in \mathcal{S}} \min_{t \in \mathcal{T}} d(s,t), \max_{t \in \mathcal{T}} \min_{s \in \mathcal{S}} d(s,t) \right\} \quad (7)$$

where cosine distance is used as the inner distance, i.e.,

$$d(s,t) \triangleq 1 - \cos(s,t) \quad (8)$$

Compared to the non-episodic method, we observe a drastic drop of Hausdorff distance from 0.18 to 0.05 for Prototypical, and also we see a minor decline of Hausdorff distance from 0.18 to 0.13 for the Reptile. Both qualitative visualization and quantitative metrics confirm that meta-learning approaches bring the distributions of auxiliary and target language data closer together, thus increasing the accuracy on the target language.

The advantage of meta-learning methods is that they obtain the appropriate initial parameters for the target language, as mentioned. The zero-shot test is used as a criterion to evaluate this point, and it shows that meta-learning-based models are more accurate than other methods. The generality of the initial parameters can also be assessed via canonical correlation analysis (CCA) (Raghu et al., 2017; Morcos et al., 2018). Using this criterion, we compare the output of each layer before and after fine-tuning, and the results are presented in Fig. 2. The meta-learning models have a higher CCA similarity, which indicates the model obtained more general parameters before fine-tuning.

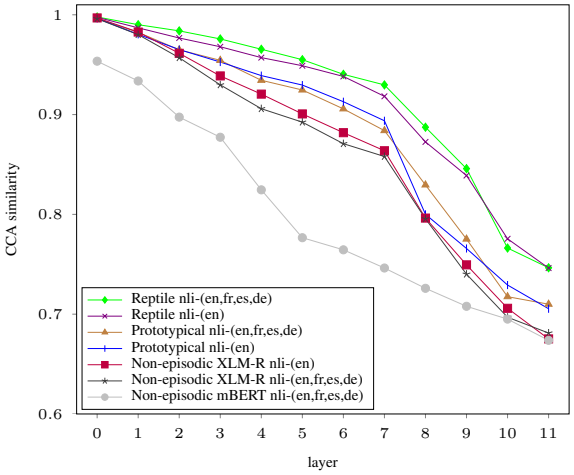

Figure 2: CCA similarity for each transformer layer. We calculate the similarity before and after fine-tuning on the FarsTail training data.

In the next experiment, we apply the DReCa strategy and train the model with the Prototypical algorithm. DReCa is a task augmentation strategy used when a small number of auxiliary tasks are available. The identification of clusters requires a more detailed examination of the method. However, in this paper, we are not interested in obtaining meaningful clusters but in generating more auxiliary tasks. So we used the DReCa strategy for the best results obtained with the Prototypical method, hoping to get more improvements. According to Table 3, there is a slight improvement when we divided a single English NLI auxiliary task into eight auxiliary tasks. Whereas DReCa did not help the model when we used several auxiliary tasks. It illustrates that task augmentation in meta-algorithms affects the model's accuracy. However, defining the appropriate task augmentation strategy still needs research.

7 Conclusion

We present effective use of meta-learning to benefit from other tasks or languages. We advantageously leverage this approach to improve NLI in Persian as a low-resource language. We found that our meta-learning model outperformed competitive baseline models. In response to the concept of treating entire datasets as tasks, we use DReCa as a general-purpose task augmenting approach. Finally, zero-shot evaluations illustrate the generality of the results obtained by meta-learning. This work will be extended to other cross-lingual NLP tasks in Persian in the future. Furthermore, we would like to use a self-supervised approach to provide a useful starting point for parameters.

References

Roee Aharoni, Melvin Johnson, and Orhan Firat. 2019. Massively multilingual neural machine translation. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*. Association for Computational Linguistics.

Hossein Amirkhani, Mohammad Azari Jafari, Azadeh Amirak, Zohreh Pourjafari, Soroush Faridan Jahromi, and Zeinab Kouhkan. 2020. FarsTail: A persian natural language inference dataset. *arXiv preprint arXiv:2009.08820*.

Sajjad Ayoubi and Mohammad Yasin Davoodeh. 2021. Persianqa: a dataset for persian question answering.

Trapit Bansal, Rishikesh Jha, Tsendsuren Munkhdalai, and Andrew McCallum. 2020. Self-supervised meta-learning for few-shot natural language classification tasks. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*. Association for Computational Linguistics.

Piotr Bojanowski, Edouard Grave, Armand Joulin, and Tomas Mikolov. 2017. Enriching word vectors with subword information. *Transactions of the Association for Computational Linguistics*.

Samuel R. Bowman, Gabor Angeli, Christopher Potts, and Christopher D. Manning. 2015. A large annotated corpus for learning natural language inference. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics.

Qian Chen, Xiaodan Zhu, Zhenhua Ling, Si Wei, Hui Jiang, and Diana Inkpen. 2016. Enhanced LSTM for natural language inference. *arXiv preprint arXiv:1609.06038*.

Alexis Conneau, Kartikay Khandelwal, Naman Goyal, Vishrav Chaudhary, Guillaume Wenzek, Francisco Guzmán, Edouard Grave, Myle Ott, Luke Zettlemoyer, and Veselin Stoyanov. 2020. Unsupervised cross-lingual representation learning at scale. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*. Association for Computational Linguistics.

Alexis Conneau and Guillaume Lample. 2019. Cross-lingual language model pretraining. *Advances in Neural Information Processing Systems*.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, Minneapolis, Minnesota. Association for Computational Linguistics.

Zi-Yi Dou, Keyi Yu, and Antonios Anastasopoulos. 2019. Investigating meta-learning algorithms for low-resource natural language understanding tasks. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*. Association for Computational Linguistics.

Marie-Pierre Dubuisson and Anil K. Jain. 1994. A modified hausdorff distance for object matching. In *Proceedings of 12th International Conference on Pattern Recognition*. IEEE.

Mehrdad Farahani, Mohammad Gharachorloo, Marzieh Farahani, and Mohammad Manthouri. 2021. ParsBERT: Transformer-based model for persian language understanding. *Neural Process. Lett.*

Chelsea Finn, Pieter Abbeel, and Sergey Levine. 2017. Model-agnostic meta-learning for fast adaptation of deep networks. In *International Conference on Machine Learning*. PMLR.

Jiatao Gu, Yong Wang, Yun Chen, Victor O. K. Li, and Kyunghyun Cho. 2018. Meta-learning for low-resource neural machine translation. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics.

Jeremy Howard and Sebastian Ruder. 2018. Universal language model fine-tuning for text classification. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics*. Association for Computational Linguistics.

Junjie Hu, Sebastian Ruder, Aditya Siddhant, Graham Neubig, Orhan Firat, and Melvin Johnson. 2020. XTREME: A massively multilingual multi-task benchmark for evaluating cross-lingual generalization. In *International Conference on Machine Learning*. PMLR.

Javier Huertas-Tato, Alejandro Martín, and David Camacho. 2021. SML: a new semantic embedding alignment transformer for efficient cross-lingual natural language inference. *arXiv preprint arXiv:2103.09635*.

Daniel P Huttenlocher, Gregory A. Klanderman, and William J Rucklidge. 1993. Comparing images using the hausdorff distance. *IEEE Transactions on pattern analysis and machine intelligence*.

Daniel Khashabi, Arman Cohan, Siamak Shakeri, Pedram Hosseini, Pouya Pezeshkpour, Malihe Alikhani, Moin Aminnaseri, Marzieh Bitaab, Faeze Brahman, Sarik Ghazarian, Mozhdeh Gheini, Arman Kabiri, Rabeeh Karimi Mahabagdi, Omid Memarrast, Ahmadreza Mosallanezhad, Erfan Noury, Shahab Raji, Mohammad Sadegh Rasooli, Sepideh Sadeghi, Erfan Sadeqi Azer, Niloofar Safi Samghabadi, Mahsa Shafaei, Saber Sheybani, Ali Tazarv, and Yadollah Yaghoobzadeh. 2021. ParsiNLU: A Suite of Language Understanding Challenges for Persian. *Transactions of the Association for Computational Linguistics*.

Tushar Khot, Ashish Sabharwal, and Peter Clark. 2018. SciTaiL: A textual entailment dataset from science question answering. In *Thirty-Second AAAI Conference on Artificial Intelligence*.

Gregory Koch, Richard Zemel, Ruslan Salakhutdinov, et al. 2015. Siamese neural networks for one-shot image recognition. In *ICML deep learning workshop*.

Bo Liu, Hao Kang, Haoxiang Li, Gang Hua, and Nuno Vasconcelos. 2020. Few-shot open-set recognition using meta-learning. In *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition*.

Ilya Loshchilov and Frank Hutter. 2018. Fixing weight decay regularization in adam.

Bill MacCartney. 2009. *Natural language inference*. Stanford University.

Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. 2013. Distributed representations of words and phrases and their compositionality. In *Advances in Neural Information Processing Systems*.

Ari Morcos, Maithra Raghu, and Samy Bengio. 2018. Insights on representational similarity in neural networks with canonical correlation. In *Advances in Neural Information Processing Systems 31*. Curran Associates, Inc.

Shikhar Murty, Tatsunori B. Hashimoto, and Christopher Manning. 2021. DReCa: A general task augmentation strategy for few-shot natural language inference. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*.

Alex Nichol, Joshua Achiam, and John Schulman. 2018. On first-order meta-learning algorithms. *arXiv preprint arXiv:1803.02999*.

Farhad Nooralahzadeh, Giannis Bekoulis, Johannes Bjerva, and Isabelle Augenstein. 2020. Zero-shot cross-lingual transfer with meta learning. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*. Association for Computational Linguistics.

Ankur Parikh, Oscar Täckström, Dipanjan Das, and Jakob Uszkoreit. 2016. A decomposable attention model for natural language inference. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics.

Barun Patra, Joel Ruben Antony Moniz, Sarthak Garg, Matthew R. Gormley, and Graham Neubig. 2019. Bilingual lexicon induction with semi-supervision in non-isometric embedding spaces. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*. Association for Computational Linguistics.

Matthew E. Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer. 2018. Deep contextualized word representations. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*. Association for Computational Linguistics.

Matúš Pikuliak, Marián Šimko, and Mária Bieliková. 2021. Cross-lingual learning for text processing: A survey. *Expert Systems with Applications*.

Kunxun Qi and Jianfeng Du. 2020. Translation-based matching adversarial network for cross-lingual natural language inference. In *Proceedings of the AAAI Conference on Artificial Intelligence*.

Kun Qian and Zhou Yu. 2019. Domain adaptive dialog generation via meta learning. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*. Association for Computational Linguistics.

Maithra Raghu, Justin Gilmer, Jason Yosinski, and Jascha Sohl-Dickstein. 2017. Svcca: Singular vector canonical correlation analysis for deep learning dynamics and interpretability. In *Advances in Neural Information Processing Systems 30*. Curran Associates, Inc.

Janarthanan Rajendran, Alexander Irpan, and Eric Jang. 2020. Meta-learning requires meta-augmentation. In *Advances in Neural Information Processing Systems*. Curran Associates, Inc.

Sachin Ravi and Hugo Larochelle. 2016. Optimization as a model for few-shot learning. In *5th International Conference on Learning Representations*.

Jasdeep Singh, Bryan McCann, Nitish Shirish Keskar, Caiming Xiong, and Richard Socher. 2019. XLDA: cross-lingual data augmentation for natural language inference and question answering. *arXiv preprint arXiv:1905.11471*.

Jake Snell, Kevin Swersky, and Richard Zemel. 2017. Prototypical networks for few-shot learning. In *Proceedings of the 31st International Conference on Neural Information Processing Systems*.

Aarne Talman, Anssi Yli-Jyrä, and Jörg Tiedemann. 2019. Sentence embeddings in nli with iterative refinement encoders. *Natural Language Engineering*.

Ishan Tarunesh, Sushil Khyalia, Vishwajeet Kumar, Ganesh Ramakrishnan, and Preethi Jyothi. 2021. Meta-learning for effective multi-task and multilingual modelling. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*. Association for Computational Linguistics.

Eleni Triantafillou, Tyler Zhu, Vincent Dumoulin, Pascal Lamblin, Utku Evci, Kelvin Xu, Ross Goroshin, Carles Gelada, Kevin Swersky, Pierre-Antoine Manzagol, et al. 2019. Meta-dataset: A dataset of datasets for learning to learn from few examples. *arXiv preprint arXiv:1903.03096*.

Niels van der Heijden, Helen Yannakoudakis, Pushkar Mishra, and Ekaterina Shutova. 2021. Multilingual and cross-lingual document classification: A meta-learning approach. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*. Association for Computational Linguistics.

Zihan Wang, Stephen Mayhew, Dan Roth, et al. 2019. Cross-lingual ability of multilingual bert: An empirical study. *arXiv preprint arXiv:1912.07840*.

Shijie Wu and Mark Dredze. 2019. Beto, bentz, becas: The surprising cross-lingual effectiveness of BERT. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*. Association for Computational Linguistics.

Mengzhou Xia, Guoqing Zheng, Subhabrata Mukherjee, Milad Shokouhi, Graham Neubig, and Ahmed Hassan Awadallah. 2021. MetaXL: Meta representation transformation for low-resource cross-lingual learning. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*. Association for Computational Linguistics.

Mingzhang Yin, George Tucker, Mingyuan Zhou, Sergey Levine, and Chelsea Finn. 2019. Meta-learning without memorization. *arXiv preprint arXiv:1912.03820*.

Wenpeng Yin. 2020. Meta-learning for few-shot natural language processing: A survey. *arXiv preprint arXiv:2007.09604*.

A Appendix

A.1 MAML Description

MAML is one of the most popular meta-learning algorithms and has proven its effectiveness in various fields (e.g., computer vision). MAML is able to find good initialization parameter values and adapt to new tasks quickly. This algorithm can be performed in one episode by following these steps:

- Make a copy of the model with its initial parameters θ.
- Use the training set D_i^{train} to train the model as
$$\hat{\theta} = \theta - \alpha \nabla_\theta \mathcal{L}_i \left(\theta, \mathcal{D}_i^{train}\right) \quad (9)$$
- Apply the model with the updated parameters $\hat{\theta}$ to the validation set D_i^{val}.
- Use the loss on the validation set to update the initial parameters θ
$$\theta = \theta - \beta \nabla \theta \sum_i \mathcal{L}_i \left(\hat{\theta}, \mathcal{D}_i^{val}\right) \quad (10)$$

A.2 Hyperparameters

Models are implemented using the PyTorch[2] framework. ParsBERT, mBERT, and XLM-R implementations are taken from the HuggingFace library [3].

In our experiments, we used the AdamW optimizer (Loshchilov and Hutter, 2018) with learning rate 1e-5 to perform the inner loop of the Reptile algorithm (2), which is known as meta-step. The hyperparameters for the Reptile algorithm are listed in Table 4.

The hyperparameters for the Prototypical algorithm are also shown in Table 5. Some parameters are calculated based on a grid search, such as Distance Cross-Entropy (DCE) and Cross-Entropy (CE) coefficients, and others are chosen similarly to the Reptile algorithm.

The number of iterations parameter varies according to the value of the shot and is chosen to ensure that all instances in the dataset appear at least once in each epoch.

A.3 Prototypical Networks

A.3.1 Loss Function

As we mentioned in section 2.2, the primary loss function of the Prototypical algorithm is DCE.

[2] https://pytorch.org/
[3] https://huggingface.co/

Hyperparameter	Value
epochs	2
number of iterations	20000
sequence length (for NLI)	128
sequence length (for QA)	384
dropout	0.1
optimizer	AdamW
learning rate	1e-5
update steps (m)	3
number of class per episode (way)	2
queue length	4
temperature parameter (τ)	1

Table 4: Hyperparameters for the Reptile algorithm

Hyperparameter	Value
epochs	2
number of iterations	20000
sequence length (for NLI)	128
dropout	0.1
optimizer	AdamW
learning rate	1e-5
number of class per episode (way)	3
DCE coefficient (λ_1)	1.0
CE coefficient (λ_2)	1.0

Table 5: Hyperparameters for the Prototypical algorithm

Since a prototype consists of distribution information from instances associated with it, the choice of these instances may introduce noise in the learned representation if the neural network is trained only by using the DCE loss. We use CE loss in addition to the DCE loss to make the feature space robust to noise. As a whole, we train the model with a combination of DCE loss and CE loss given by the following equation.

$$\text{Loss(overall)} = \lambda_1 \,\text{Loss}(DCE) + \lambda_2 \,\text{Loss}(CE) \tag{11}$$

A.3.2 Scenarios

We considered two scenarios for making the episodes. In the first scenario, the model is trained only on auxiliary languages, then fine-tuned using the target language. Therefore, only auxiliary languages are used to generate support and query sets. An episode of the first scenario is shown in Table 6.

In the second scenario, in addition to auxiliary languages, we also used the target language for training. So, the support set is constructed from auxiliary language data, and the query set is generated from both auxiliary and Persian language data. Table 7 shows an episode of the second scenario.

A.4 Confusion Matrices

The confusion matrices for the top-performing models (8-shot with four auxiliary languages) are depicted in Fig. 3, showing the success of this method in improving the accuracy in all classes, especially the neutral class.

Example	Category
Support set (or Query set)	
In the midst of this amazing amalgam of cultures is a passion for continuity ⇒ *A passion for continuity is not the most important of these cultures*	neutral
The river plays a central role in all visits to Paris. ⇒ *The river is central to all vacations to Paris*	entailment
For the moment, he sought refuge in retreat, and left the room precipitately. ⇒ *He stayed put and sat on the floor.*	contradiction

Table 6: Example for a 3-way 1-shot episode in the first scenario. In this example we select support set and query set samples from English dataset. As support and query sets are generated similarly, only one set is shown in this table.

Example	Category
Support set	
Recuerda que una vez mencionó que su padre era médico? ⇒ *Ella mencionó que su padre era médico hace mucho tiempo*	neutral
Dies ist etwas anderes als eine Cantina-Leuchte ⇒ *Dies ist sicherlich keine Cantina-Leuchte*	entailment
Ensuite, il enfonce un tube respiratoire dans la gorge du patient mort. ⇒ *Le patient vit toujours.*	contradiction
English Translation	
You remember her once mentioning that her father was a doctor? ⇒ She mentioned her father being a doctor a long time ago.	neutral
This is something other than a cantina fixture. ⇒ This is certainly not a cantina fixture.	entailment
Next he shoves a breathing tube down the dead patient 's throat . ⇒ The patient is still alive.	contradiction
Query set	
Une pièce qualifie Frank Lloyd Wright de terrible ingénieur. ⇒ *Piece a également déclaré que Wright était un bien meilleur concepteur.*	neutral
Sus rápidos oídos captaron el sonido del tren que se acercaba. ⇒*Escuchó que el tren se acercaba rápidamente.*	entailment
از قرن دوازدهم به بعد ارقام عربی برای نخستین بار در ایتالیا کاربرد یافت. ⇒فرانسه اولین کشوری بود که از ارقام عربی استفاده کرد.	contradiction
English Translation	
A piece calls Frank Lloyd Wright an awful engineer. ⇒Piece also stated Wright was a much better designer.	neutral
Her quick ears caught the sound of the approaching train. ⇒She heard the train approaching fast.	entailment
From the twelfth century onwards, Arabic numerals were first used in Italy. ⇒France was the first country to use Arabic numerals.	contradiction

Table 7: Example for a 3-way 1-shot episode in the second scenario. In this example, the support set samples are selected from French, Spanish, and German datasets, respectively, and the query set samples are selected from French, Spanish, and Persian datasets, respectively.

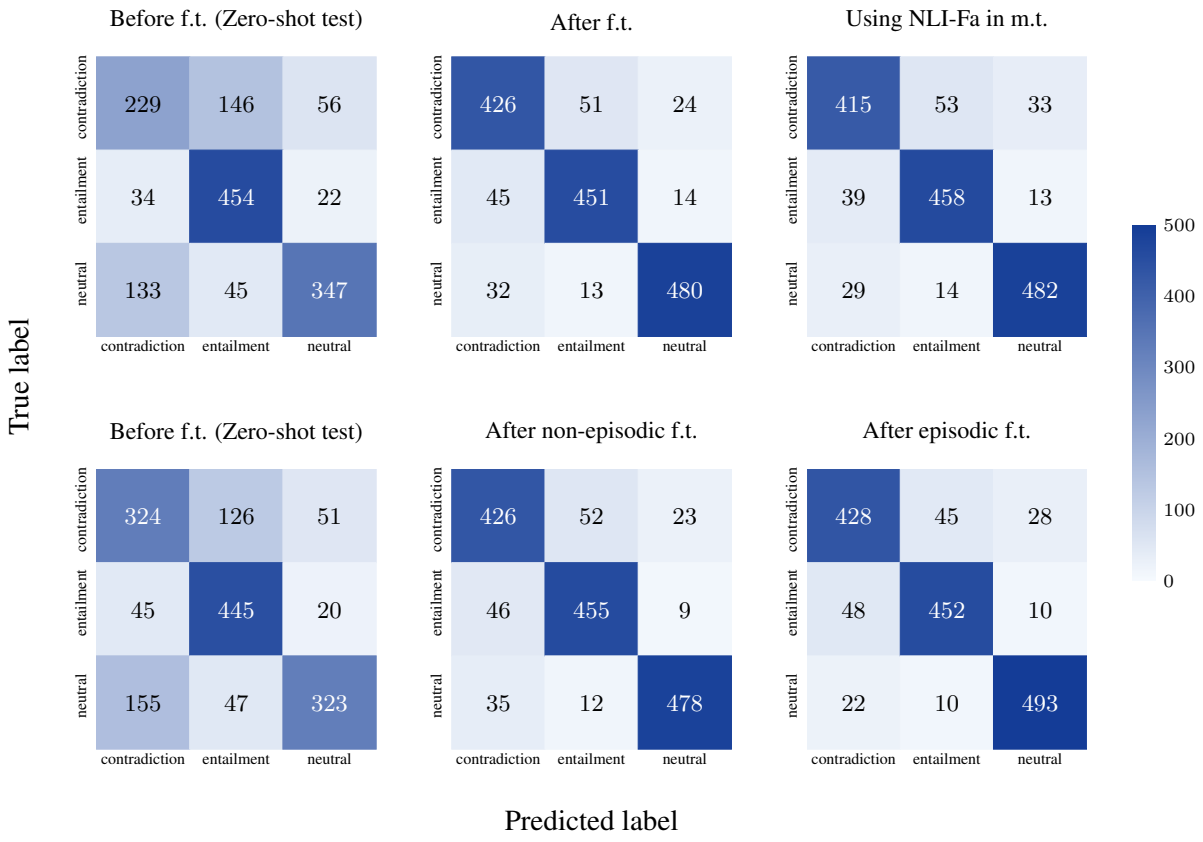

Figure 3: Confusion matrices of the best-obtained model (8-shot with four auxiliary languages) in both meta-learning algorithms on the FarsTail test set. (Top): Reptile algorithm results. (Bottom): Prototypical algorithm results.

Global Readiness of Language Technology for Healthcare: What would it Take to Combat the Next Pandemic?

♡Ishani Mondal,*♡Kabir Ahuja,*♡Mohit Jain, ♢Jacki O'Neil, ♡Kalika Bali, ♡Monojit Choudhury
♡Microsoft Research Labs, Bangalore, India
♢Microsoft Africa Research Institute, Nairobi, Kenya
{t-imonda, t-kabirahuja, kalikab, monojitc}@microsoft.com

Abstract

The COVID-19 pandemic has brought out both the best and worst of language technology (LT). On one hand, conversational agents for information dissemination and basic diagnosis have seen widespread use, and arguably, had an important role in fighting against the pandemic. On the other hand, it has also become clear that such technologies are readily available for a handful of languages, and the vast majority of the global south is completely bereft of these benefits. What is the state of LT, especially conversational agents, for healthcare across the world's languages? And, what would it take to ensure global readiness of LT before the next pandemic? In this paper, we try to answer these questions through survey of existing literature and resources, as well as through a rapid chatbot building exercise for 15 Asian and African languages with varying amount of resource-availability. The study confirms the pitiful state of LT even for languages with large speaker bases, such as Sinhala and Hausa, and identifies the gaps that could help us prioritize research and investment strategies in LT for healthcare.

1 Introduction

The world witnessed one of the worst pandemics in early 2020, COVID-19, infecting over 250 million people globally. Scientists and technologists from various fields joined hands, lending support to deal with this global crisis. Language Technology (LT), particularly the conversational agents (aka chatbots), played a crucial role during the pandemic by facilitating correct information dissemination (Li et al., 2020; Maniou and Veglis, 2020) and early disease screening (Judson et al., 2020a; Martin et al., 2020b). Nevertheless, today practically useful chatbots and other benefits of LT are available only in a handful of languages (Joshi et al., 2020). Despite impressive gains made by the Massively Multilingual Transformer based Language Models (MMLM) (Devlin et al., 2019; Lample and Conneau, 2019; Aharoni et al., 2019; Conneau et al., 2020; Xue et al., 2021) on standard NLP benchmark tasks (Pan et al., 2017; Conneau et al., 2018; Yang et al., 2019; Ruder et al., 2021), the real-world implications of such advancements remain largely unexplored. Joshi et al. (2020) has highlighted such a disparity and proposed a language hierarchy that comprises of the languages of world classified into six classes based on their resource-availability. In this hierarchy, Class 5 represents the most resource-rich languages for whom benefits of LT are readily available; and class 0 denotes the most under-resourced languages.

In this paper, we ask the following two questions: (1) Today, in which languages can we build practically useful LT systems, especially chatbots, that could serve as beneficial assistants during the pandemic? (2) How should we prioritize research and resource building investments so that LT is globally ready before the next pandemic?

In order to answer these questions, we review the existing literature and resources on COVID-19 chatbots, and classify them based on the languages they support and the solutions they provide. Quite unsurprisingly, the survey reveals a strong disparity in LT solutions between resource-rich and resource-poor languages. In order to quantify this gap and measure the pandemic-readiness of various languages today, we select 15 Asian and African languages (except English) with various degrees of resource-availability, and attempt to build COVID-19 FAQ bots for them. Since building an end-to-end chatbot is a substantial engineering effort, we scope the problem down to building an intent classifier for these languages, which forms the core of the Natural Language Understanding (NLU) unit. We also experiment with entity recognition for a subset of these languages. Our code and datasets have been made publicly available to foster future research.[1]

*Equal contribution

[1] https://github.com/kabirahuja2431/

Proceedings of the 29th International Conference on Computational Linguistics, pages 4320–4335
October 12–17, 2022.

Our study shows that despite using the best available commercial multilingual chatbot frameworks (e.g., Google Dialogflow[2], Microsoft Bot Framework (MS Bot)[3]), advanced Machine Translation (MT) systems[4], and state-of-the-art massively multilingual language models (mBERT (Devlin et al., 2019) and XLM-R (Conneau et al., 2020)), there is a 20-30% drop in performance for class 0 - 2 languages as compared to English. The drop is large for all the African languages (e.g., Hausa and Somali) and some of the Asian languages (e.g., Marathi and Sinhala). Note that our experiments were limited to languages which are supported by at least one of the chatbot frameworks, MT systems or MMLMs. There are thousands of other languages which are supported by none. We have experimented with one such language (Kikuyu) and observed near random performance. Therefore, we hypothesize that such languages are not at all ready for the next pandemic.

We extrapolate our findings at global scale and construct a global LT readiness map for pandemic-response and healthcare. Based on this map as well as error analysis of the chatbot experiments, we identify a set of research problems as well as resource-prioritization strategies which we believe are key to ensure global LT readiness before the next pandemic. More specifically, the purpose of our work is about comparing the systematic inequalities that exist across different languages while deploying chatbots for emergency situations, as well as showing that certain geographical regions are in a disadvantaged position because even the languages that are spoken by a large portion of their population are ill-supported by current LT. Finally, we provide recommendations on how this gap can be bridged by suggesting investment strategies for building LT systems which is otherwise a tough ethical question.

The rest of the paper is organized as follows: Sec 2 presents the literature survey on LT response to COVID-19, specifically focusing on chatbots built for the pandemic. Sec 3 describes the chatbot building experiments, where in 3.1 we motivate the choice of languages for the experiment, in 3.2 and 3.5 we discuss the intent and entity detection experiments respectively. In Sec 4 we present the global LT readiness map and in Sec 5, we conclude with our recommendations.

2 Literature Survey

In the recent years, NLP for Healthcare has witnessed a major uptake and an impressive volume of work has significantly pushed the research forward by developing sophisticated domain-specific language models (Alsentzer et al., 2019; Lee et al., 2020; Ji et al., 2021). These models have been adopted to serve different axes of healthcare such as patient provider communication (Min et al., 2020; Si et al., 2020), information dissemination (Maniou and Veglis, 2020), and self-care management and therapy (Morris et al., 2018; Kadariya et al., 2019; Park et al., 2019; Kamita et al., 2019). The role of healthcare chatbots becomes crucial along all these axes because of the recent adoption of telehealth technology services (Bhat et al., 2021).

Chatbots have received a considerable interest during the recent COVID-19 pandemic. Due to the worldwide spread and severity of the virus and subsequent global response, we believe that the study of COVID-19 chatbots can provide us an accurate picture of the global-readiness of LT. We surveyed COVID-19 chatbots that are mentioned in the literature and/or deployed in the real-world[5].

2.1 Use-Cases and Technological Support

From the survey, two primary use-cases of COVID-19 chatbots emerge – (1) information dissemination: answering pandemic-related questions asked by the users (Li et al., 2020; Desai, 2021; Prasannan et al., 2020; Mehfooz et al., 2020; Trang and Shcherbakov, 2021), and (2) symptom-screening: assessing risk factors associated with the symptoms provided by the user for quick diagnosis (Ferreira et al., 2020; Martin et al., 2020a; Judson et al., 2020b; Quy Tran et al., 2021). Existing commercial frameworks such as DialogFlow, Watson Assistant and MS Bot have been used primarily for building a majority of these chatbots (Li et al., 2020; Sophia and Jacob, 2021). However, open-source bot frameworks like Rasa (Quy Tran et al., 2021; Nguyen and Shcherbakov, 2021) have also been gaining

Covid19HealthBots

[2] https://cloud.google.com/dialogflow

[3] https://dev.botframework.com/

[4] https://translate.google.co.in/ , https://www.bing.com/translator

[5] Besides healthcare, NLP has also proved beneficial in providing aid during natural disasters like earthquakes and floods (Lewis, 2010; Rudra et al., 2015; Ghosh et al., 2019; Tsai et al., 2019; Basu et al., 2019). Strassel and Tracey (2016) leveraged existing LT for resource-poor languages to fight against the natural disasters. Though this study is limited to pandemic readiness, we believe the state of LT for disaster-readiness across the globe would be very similar.

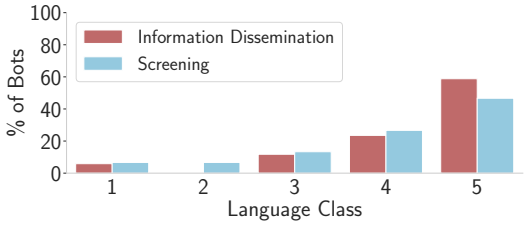

(a) Percentage of papers covering the development of COVID-19 chatbots per language class

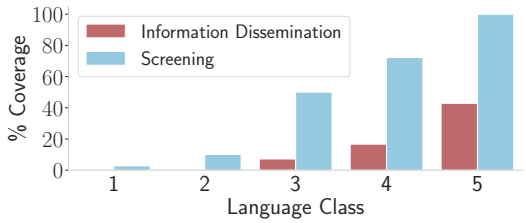

(b) Percentage of languages of each class for which at-least one paper addressed the development of a chatbot.

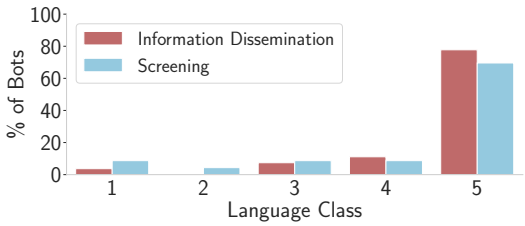

(c) Percentage of bots deployed by governmental agencies , provider organizations etc. per language class, to serve people during the pandemic.

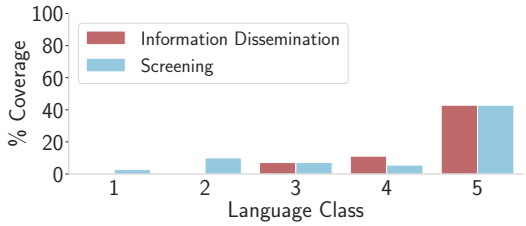

(d) Coverage of languages across different classes in the bots deployed by governmental agencies , provider organizations etc.

Figure 1: Bots developed (for both research and public deployment) for different language classes and their coverage.

traction in the community. The inbuilt NLU engines supported by these frameworks makes chatbot development easy, hence there is a significant uptake in utilizing these to develop new chatbots. Pre-trained LMs were also leveraged for COVID symptom identification (Oniani and Wang, 2020) and question answering (Park et al., 2020).

2.2 Language Diversity

Which languages are supported by these COVID bots? Of the 20 COVID-related bots mentioned in the existing literature and 34 others deployed by different countries to combat the pandemic, 26 ($\approx 50\%$) are exclusively for English, followed by German having 10 deployed bots. In Figure 1, we show the distribution of the chatbots by the language classes defined in (Joshi et al., 2020). As expected, for all the cases, we observe that chatbots were available primarily and almost exclusively for languages in class 5. We observe a slightly higher presence of class 4 and 3 languages in research papers on COVID chatbots (Fig. 1a). For instance, there are three research papers each for Hindi and Vietnamese, both class 4 languages. (Mabrouk et al., 2021) has been recently introduced to help in information dissemmination about covid in African languages. To the best of our knowledge, we could not find any publication or deployed bot for class 0 languages. This skew is more prominent when we consider the coverage of languages of different classes, i.e., the fraction of languages in each class for which at least one COVID-19 conversation system was developed (Figure 1b and 1d). This lack of attention to a large number of languages has also been highlighted by Anastasopoulos et al. (2020) who strongly advocated for the development of language resources for improving access to COVID-19 related information in 26 lesser-resourced languages, particularly from Africa and South and South-East Asia.

3 Rapid Chatbot Building Exercise

How quickly can one build a pandemic response chatbot in a language based on the best publicly available systems? In order to answer this question we have to understand the pandemic-readiness of various languages. To do this, we made an attempt to build chatbots for answering frequently asked questions about COVID-19 using Google Dialogflow, Microsoft Bot Framework (MS Bot), as well as two of the most popular Massively Multilingual Language Models (MMLM) – mBERT and XLM-R. Since building an end-to-end chatbot is complex, we chose to conduct rapid prototyping experiments for intent recognition in 16 languages, and entity recognition in 3 languages.

3.1 Language Selection Criteria

For our experiments, we chose a few languages from each language class (Joshi et al. (2020)) such that at least one language per class is supported by either of the two commercial chatbot frameworks, leading to the set: *English, Chinese* from Class 5, *Hindi, Korean* from Class 4, *Bengali, Malay* from Class 3, *Swahili, Hausa, Marathi, Amharic, Zulu* from Class 2, *Assamese, Gujarati, Kikuyu, Somali*

from Class 1, and *Sinhala* from Class 0.

3.2 Intent Recognition

Intent Recognition is an essential component of conversational systems. Given a user query, the task is to classify it into one of the pre-defined intent categories (Braun et al., 2017).

3.2.1 Dataset Creation and Characteristics

For training and evaluation, we curate a set of 147 queries categorized into one of the 14 intents: 1) `Airborne` (how COVID spreads by air), 2) `ClarifyCovid` (difference between COVID and other diseases), 3) `Country` (country-wise infection statistics), 4) `CovidTwice` (possibility of reinfection), 5) `ExplainSymptom` (COVID symptoms) 6) `Incubation` (how many days of incubation required), 7) `Length` (longevity of infection), 8) `Mask` (ways of wearing mask), 9) `Protection` (ways to protect against infection), 10) `Quarantine` (quarantine requirement of US), 11) `Spread` (how COVID spreads), 12) `Testing` (available COVID tests), 13) `Medication` (about drugs to protect from COVID), and 14) `Treatment` (about treatments or therapies related to COVID). Examples and definitions of each intent are present in Table 3.3.

We refer to the FAQs provided by the UN (Department of Operational Support, 2020) and user queries in the dataset released by Anastasopoulos et al. (2020), to identify the 14 types of questions that a user may ask. We manually paraphrase the questions to generate queries (Mean = 10.5, S.D. = 4.36 queries per intent) for each intent in English. Two annotators with native English proficiency independently classified these queries; the inter-annotator agreement (κ) was 0.89. We asked a few native speakers of each of the selected languages to translate these 147 queries manually. The dataset is split into train and test sets, using a stratified split over the intents, giving a total of 76 and 71 queries in train and test set respectively.

3.2.2 Strategies of Developing Chatbots

We consider three training and inference strategies, emulating the possible scenarios for developing such chatbots in practical settings (Table 3.4).

Train on English Data: In this strategy, we develop our bots by training them on the English queries, and evaluate the intent detection performance in different languages by automatically translating the test queries into English (e.g., similar to Gupta et al. (2021)).

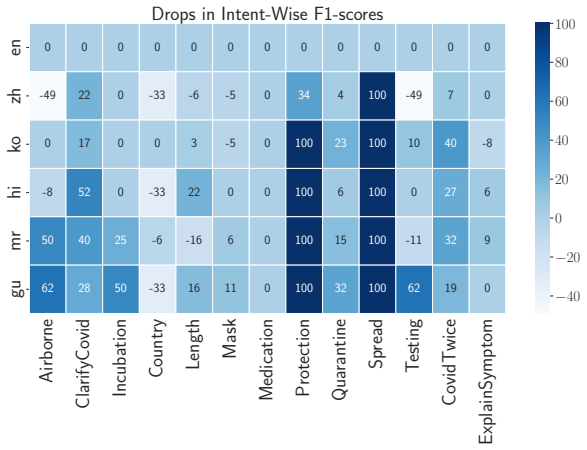

Figure 2: Relative drops (relative to English) in intent wise F1 scores for different languages in the *Train on Manual Translations* setup (in LUIS). Negative values indicate increase in the scores relative to English.

Train on MT Translations: Here we build target language intent classifier models from training data in different languages, which is obtained by automatically translating the English training data. The classifier is then tested on the manually translated test data in the corresponding target language. A similar method was adopted by Balahur and Turchi (2012) for sentiment analysis.

Train on Manual Translations: In this setup, we use the manual translations of the English training dataset to train our bots in different languages. Like the previous setups, here again we use the manually translated data to evaluate the intent detection performance of the developed chatbots. Jennifer Bot (Li et al., 2020) used a similar setup to extend their English bot to Spanish. Note that this is the most expensive setup in terms of data creation cost.

3.3 Intent Definitions and Descriptions

The different intents used for our experiments are described in table 5. We provide definitions and examples for each of the different intents used.

3.4 Bot Building Strategies

3.4.1 Experimental Setup

Commercial Frameworks: We use Google Dialogflow and MS Bot Framework to train and evaluate the FAQ bots in different languages. For Dialogflow we use the ES Console, and for MS Bot, we use Microsoft's Language Understanding

Intent Type	Example in English	Definition
Airborne	Can the virus that causes COVID-19 be transmitted through the air?	Queries related to how much COVID is carried by air
ClarifyCovid	How do I know if it is COVID-19 or just the flu?	Queries related to difference betwen COVID and other diseases
Incubation	Can someone in incubation infect other people?	Queries related to situations where a person is infected with COVID and is going through incubation phase
Length	How long does the illness make you poorly for?	Queries regarding longevity of COVID infection
Mask	Should I wear a mask while exercising?	Queries about wearing mask
Protection	Ways to keep safe from COVID-19	Queries about the ways of protection from COVID
Quarantine	Will I avoid coronavirus, if I self-isolate?	Queries about the effect of quarantining after getting infected with COVID
Spread	Aside from inhalation, are there other ways coronavirus can spread?	Queries about the spreading process of COVID
Testing	Where can I get my test done?	Queries about the testing process of COVID
CovidTwice	If you get COVID-19, can you get it again?	Queries about whether COVID can infect someone more than once
ExplainSymptom	I have a sharp pain here in the chest	User explaining COVID related symptoms
Country	How many people in Italy have COVID-19?	Querying about the statistics of infection in different countries
Medication	Do any of the drugs reduce mortality?	Querying about the medication to survive from COVID
Treatment	Which vaccines are good to protect against the virus?	Querying about the treatment strategies associated with COVID

Table 1: Different intents with definitions and examples present in our dataset.

Bot Building Setup	Training Strategy	Testing Strategy
Train on English Data	Train set comprises of the English queries	Test set comprises of English queries where the manually written queries in target language are translated to English using MT system
Train on MT Translations	Train set comprises of the English queries translated to target language using MT System	Test set comprises of manually written queries in target language
Train on Manual Translations	Train set comprises of manually written queries in target language	Test set comprises of manually written queries in target language

Table 2: Different strategies for building the chatbots.

Service (LUIS)[7] framework. Dialogflow and LUIS supports 7 and 6 out of our 16 selected languages, respectively. For the unsupported languages, we could only experiment with *Train on English Data*.

Pre-trained MMLMs: We evaluate two popular MMLMs, namely mBERT (*bert-base-multilingual-cased*) and XLMR (*xlm-roberta-base*), for our intent detection experiments. XLM-R supports all but Kikuyu, Somali and Zulu, while mBERT supports all but Amharic, Assamese, Hausa, Kikuyu and Zulu. For these models, we only evaluate the *Train on Manual Translations* setup. We experiment with two different approaches for building intent classifiers with these models: i) Using k-Nearest Neighbors on the sentence embeddings obtained through the MMLM to classify the intents as done in Caron et al. (2021), ii) Training an end-to-end classifier by fine-tuning the pre-trained MMLM. We report the best scores out of these two setups for both MMLMs (details in A.3).

Evaluation: We report the relative accuracy drop δ_l for each target language l from English (en), defined as $(A_{en} - A_l)/(A_{en}) \times 100$, where A_l is the accuracy of intent classification for l on the held-out test set[8]. Thus, lower the value of δ_l, better is the state-of-the-art of LT for the language l.

3.4.2 Results and Analysis

Table 3 presents the intent classification results which reports the relative drop of the model's accuracy with respect to English. While the relative drop δ_l is reported, we also mark the values with a † wherein the *absolute* accuracy, A_l falls below 67%. We use this as a minimum viable threshold of

[7] https://www.luis.ai/

[8] Absolute accuracies are not reported since we do not intend to compare the performances of commercial frameworks.

Class	Languages	Train on English Data		Train on MT Translations		Train on Manual Translations			
		DF	LUIS	DF	LUIS	DF	LUIS	mBERT	XLM-R
5	Chinese	0.60	5.00	17.50†	18.40†	0.04	(5.20)	(5.63)	(8.50)
4	Hindi	12.50	0.05	16.25†	25.01†	13.02	10.50	12.72†	19.15†
	Korean	6.50	13.71†	31.20†	11.55	23.75†	10.00	5.63†	10.88†
3	Bengali	20.50†	13.15†	26.25†	×	11.80	×	7.04†	2.13†
	Malay	21.24†	11.87	19.53†	×	12.50	×	4.77	14.89†
2	Swahili	28.08†	18.00†	×	×	×	×	32.39†	19.15†
	Hausa	40.97†	34.00†	×	×	×	×	×	29.79†
	Marathi	21.23†	14.00†	28.08†	28.7†	16.25†	17.76†	16.90†	29.79†
	Amharic	43.06†	34.82†	×	×	×	×	×	12.39†
	Zulu	30.56†	11.28	×	×	×	×	×	×
1	Assamese	19.52†	18.00†	×	×	×	×	×	29.79†
	Gujarati	15.55	10.03	×	22.88†	×	22.88†	4.77	19.15†
	Kikuyu*	97.60†	76.87†	×	×	×	×	×	×
	Somali	40.56†	27.58†	×	×	×	×	25.35†	×
0	Sinhala	35.00†	19.00†	34.93†	×	15.65	×	61.97†	19.15†

Table 3: δ_l for each language for the *Intent Recognition* task using the three different strategies. × indicates that the framework does not support end-to-end chatbot development for that language. Drops that lead to accuracy below 67% are marked by †, indicating the case where the bot mis-recognizes 1 out of every 3 queries. *Owing to non-availability of standard MT for Kikuyu, we used *Safarini*[6] app from Android playstore for translation. Note: The values mentioned in the parantheses indicate that we observe relative gain instead of drop.

Lang	Issue	Actual Example	Misclassified Translated Example
Si	**Terminology Mismatch**	I have hay fever though too.	I also have gonorrhea.
Bn	**Fluency**	Is SARS-CoV-2 airborne?	Does SARS-CoV-2 sit in the air?
Hu	**Relevance**	I got the virus. How long does it go on for?	I Nasami Cutar. How long will it take?
Hu	**Fluency, Terminology Mismatch**	How long should I wear a mask?	How long will I impose sanctions?
Hu	**Terminology Mismatch**	Is it healthy to wear a mask during swimming?	Is it safe, can I wear fascist sanctions when I swim?

Table 4: Excerpts of test instances showing bottlenecks of MT systems in the *Train on English Data* setup.

the performance, as below this the model will misclassify more than 1 out of every 3 queries which might not be useful for real world deployments. As expected, we observe high δ_l for languages belonging to class 3 or lower, with most of the accuracies below the acceptable limit.

Comparison across the three setups: We observe that for classes 4 and 5, *Translate on English Data* performs at par or even better than the most expensive *Train on Manual Translations* setup. This may be because the MT translations from these languages to English is highly accurate. On the other hand, for languages belonging to class 3 or lower, *Train on Manual Translations* led to better performance, arguably due to poorer performance of the MT system. Unfortunately, the *Train on Manual Translations* method is the most expensive in terms of data curation cost, hence may be the hardest to implement in the midst of a pandemic. The problem becomes worse because a majority of class 3 and lower languages are not supported by current chatbot frameworks. Even when supported, their performance is below the acceptable limit (e.g., Marathi, Gujarati). One of the reasons is the difficulty in correctly identifying technical intents like `Airborne` and `Incubation` in such low-resource languages (Figure 2). Since a few of these low-resource languages are present in the pre-training dataset of mBERT and XLM-R, we can evaluate them for *Train on Manual Translations*. There is a similar pattern in accuracy drop for MMLMs, however the accuracy begin to fall below the acceptable limit (67%) from class 4 languages onward. There is a remarkable drop in mBERT's accuracy for Sinhala (class 0). In general, we

find mBERT to outperform XLM-R, except for Swahili and Sinhala. This may be due to the better representation of these languages in the pre-training corpus of XLM-R (CommonCrawl Corpus). This strongly indicates the importance of the pre-training dataset size for developing LT, both in terms of absolute size as well as relative size to other languages (Wu and Dredze, 2020). As expected, the performance in *Train on MT Translations* setup is the worst among the three; except for Korean in LUIS, all values lie below the acceptable limit, which could be a compounded effect of poor translation quality and inferior NLU solutions. To conclude, all languages in class 3-5 had at least one solution yielding an acceptable accuracy, while all languages in class 0-2, except Gujarati, Sinhala and Zulu, had no acceptable solution.

Lost in Translation: Table 4 shows the intent misclassification errors due to the errors in MT translations. The manual translation in the target language correspond to the 'Actual Example' in English, and the phrase translated back to English for the *Train on English Data* setup is reported under the 'Misclassified Translated Example.' We categorize the translation errors as Terminology Mismatch, Fluency and Relevance (Li et al., 2020). We find that domain-specific terms often get translated incorrectly into English (4). In a few cases, the translations result in unnatural queries resulting in loss of fluency, such as *Does SARS-CoV-2 sit in the air?*. All these factors lead to poor performance of *Train on English Data* setup for low-resource languages. We find that *Terminology Mismatch* is the most common issue affecting the performance[9]. Interestingly, technical terms like *incubation* does not exist in a few of our target languages, hence the manually written test queries in these languages just had the English term written in that language's script. In such cases, we found lesser performance drop compared to languages when equivalent vocabulary exists in the target language. E.g., high drops in F1-score for intents like "`Quarantine`" and "`Incubation`" in Hausa (76%, 100% respectively) and Amharic (56%, 100%) justify this, whereas for Zulu, where the human translator used English terms in their queries resulted in much lower drop in F1 scores (20%, 0%). See appendix for the intent-wise F1 scores for different languages.

Implications: Based on our experimental results, we wish to explore how to prioritize the resource-investment strategies to push the state of current LT forward. Resource-poor languages mostly underperform across all the three set-ups, so then *should we invest more towards developing better translation systems or focus more on improving the current NLU solutions for different languages?* We observe that a good quality translation system can support building bots from scratch in a new language, and often performs on-par with the *Train on Manual Translations* setup for high-resource languages (e.g., Korean, Hindi) and sometimes even for low-resource languages (e.g., Gujarati). Building bots from scratch in a new language is resource-intensive, requiring rapid prototyping, which may be infeasible during a crisis since massive data collection efforts need to be made. Therefore, a generic way to ensure pandemic-readiness in a language is by ensuring reasonably accurate MT systems similar to that for class 4 languages. Improving representation of low-resource languages in the pre-training datasets of existing multilingual models (specifically on domain-specific corpora as done by (Gu et al., 2021; Zhang et al., 2020)) is yet another way to ensure preparedness, as it can lead to improved performance of the MMLMs on these languages (Wu and Dredze, 2020). Unlabelled language data for low resource languages can also be leveraged to build Machine Translation systems in these languages when used in conjunction with the parallel data in high resource languages for training massively multilingual models. (Siddhant et al., 2022; Bapna et al., 2022).

3.5 Entity Recognition

We also evaluate the developed chatbots on another core task of NLU, i.e. entity recognition (Ali, 2020) on English, Hindi and Bengali. To train and evaluate different COVID bots on this task, we use a set of 200 user related queries (obtained by augmenting existing dataset of 147 queries). Entity types were identified from a subset of labels from the

[9]While investigating the correlation between the translation quality (BLEU (Papineni et al., 2002)) between the reference original English query and the back-translated English string from each manually written query in target language) and the intent classification results, we did obtain a positive spearman correlation coefficient with the value of 0.36. and a p-value 0.21. With a p-value of 0.21 the correlation is not statistically significant which we suspect might be due to BLEU not being a great measure of translation quality (Sulem et al., 2018) to measure the subtleties discussed above.

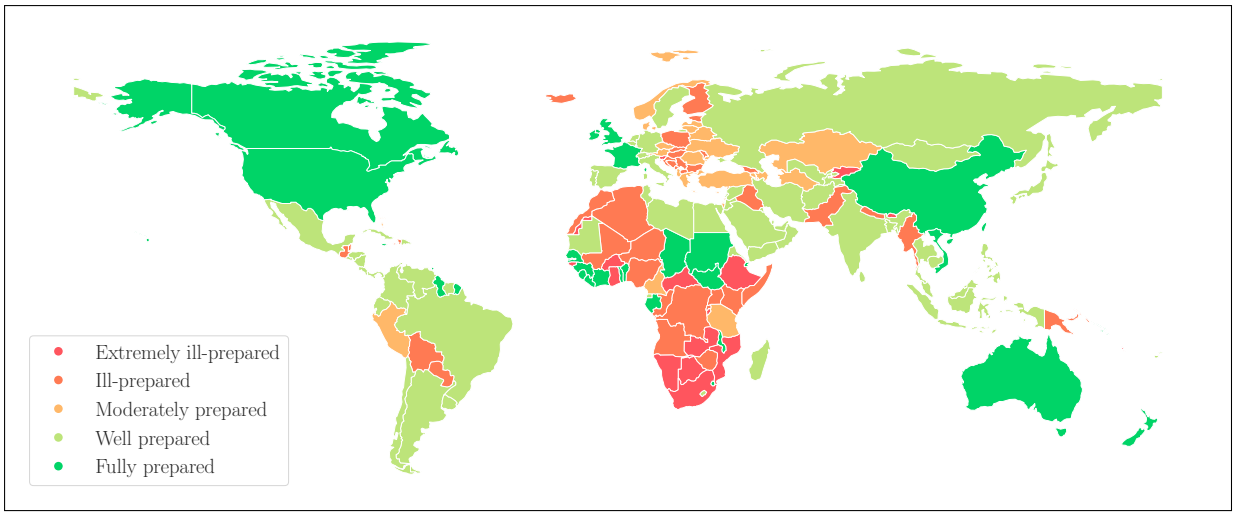

Figure 3: World map showing the Readiness of each country in terms of fighting the next pandemic using LT.

CORD-19 NER dataset (Lu Wang et al., 2020), and the queries were accordingly tagged by two native speakers of Bengali and Hindi. Overall, our dataset had a mix of *medical* and *non-medical* entities. The final set of *medical* entity types consists of: `Covid` (COVID-related entities), `PhysicalScience` (technical terms related to bio-molecular mechanism of the disease), and `Disease` (any form of illness or symptoms). The *non-medical* entity types are: `BodyPart` (name of the body part), `Country` (country name), `Duration` (length in days), `Protection` (ways to protect against COVID, such as 'mask', 'gloves'), and `InfoSource` (source of information). `Country` (country name). For generating the equivalent translations, we manually aligned the entity tags in two languages: Hindi (supported by DialogFlow and LUIS) and Bengali (supported by DialogFlow). In majority of the cases, we observe that domain-specific entities such as `incubation, ACE-2 Cells, biochemical assays` are hard to predict by these models on languages other than English. For instance, for `Covid` entity, we observe significant F1-score drop of 24.6% for Hindi and 42.9% for Bengali. However, for non-medical entities, these models were found to perform comparatively better, e.g., drop in F1-score on `Country` tag was 5.2% for Hindi and 8.9% for Bengali.

4 Measuring Global Readiness

Although our current work focuses on analyzing pandemic-preparedness of only 16 languages, here we try to generalize our findings to other languages by introducing a *Readiness Score* for every language which empirically measures the preparedness of current LT to serve its speakers in a pandemic-like emergency situation. The definition of readiness is based on the assumption that one has access to the best available LT by considering the highest intent detection accuracy of A_l^* for a language across different frameworks and training setups. We then define the readiness of a language l as its relative accuracy with respect to English as $r_l = \frac{A_l^* - A_{random}}{A_{en}^* - A_{random}}$, where A_{en}^* denotes the best case accuracy on English, and A_{random} is the accuracy of a random classifier: $A_{random} = 100/numberOfIntents$.

We would like to interpolate r_l for all the languages of the world, and hence would need more training examples than the 16 languages that we currently have. We select a set of 11 proxy languages (details in Appendix A.5). This has been done in order to ensure the coverage of the features of major language families in the world[10] while training the model. For these languages, we compute *proxy* accuracies $\tilde{A}_l^*$ by building and evaluating chatbots on MT translated data. We then train a Gaussian Process Regression model for predicting readiness scores with the r_l values for the 27 languages as our training set. We use geographical and genetic features from the URIEL database (Littell et al., 2017) to represent the languages. The predictive model, which has an average absolute prediction error of 5%, is then used to estimate the readiness scores of 116 new languages supported by major MMLMs (mBERT and XLM-R) and/or translators (Google and Microsoft). For all

[10] *Ethnologue 24 (2021)*: `https://en.wikipedia.org/wiki/List_of_language_families`

other languages, we set $r_l = 0$, as one can expect near random performance without any LT, as we did see for Kikuyu (Table 3). The estimated final r_l scores for all the languages were used to extrapolate the pandemic-readiness of each country c, as follows. We use the country-wise language and speaker demographic data[11] to calculate the country-wise readiness (similar to Blasi et al. (2021)), $r_c = \sum_{l \in \mathcal{L}_c} s_{c,l} r_l$, where $\mathcal{L}_c$ is the set of languages spoken in country c, and $s_{c,l}$ is the fraction of c's population forming native speakers of the language l. The r_c values were clustered to generate five classes (Extremely ill-prepared: 0-0.33, Ill-prepared: 0.33-0.74, Moderately prepared: 0.74-0.83, Well prepared: 0.83-0.92, Fully prepared: 0.92-1) using Jenks' natural breaks optimization (Jenks, 1967). These classes were used to generate a readiness heatmap of the world (Fig 3).

Observations: From Figure 3, one can observe that South and East African countries are Extremely ill-prepared, due to the high dominance of low-resource languages. For instance, people in Zambia's speak Bemba, Chewa and Luzi, all of which are severely under-resourced. As pointed out in Anastasopoulos et al. (2020), these regions might also be worse-hit in a pandemic situation, and therefore, require immediate attention. For Ill-prepared regions such as Bolivia in South America, and Guatemala in Latin America, r_l values are slightly better due to the abundance of Spanish speakers, however there is a sizeable population speaking under-served languages such as Q'eqchi and Guarani. Countries that fall within fully to moderately prepared categories typically have large native speaker population of one or more of the class 5 languages (English, French, Chinese, Arabic) and/or well-supported languages (e.g., Korean, Bengali, Malay). It is important to note that while approximating readiness of a language, we assumed same value for all its diverse linguistic variants and dialects, which in certain cases results in overestimation of r_c. High r_c for north and central African countries (e.g., Libya, Egypt and Sudan) might be due to sizeable population of a resource-rich language Arabic. However, Arabic has several dialects, which vary from the Modern Standard Arabic at various linguistic levels, and consequently the performance of LT systems for such dialects also vary considerably (Zbib et al., 2012; Alsharhan and Ramsay, 2020). It holds true for Spanish and Portuguese spoken in Latin America (Lipski, 2014) and French dialects of Western Africa.

5 Conclusion and Recommendation

From our chatbot development experiences, we uncover a set of interesting insights to arrive at the following recommendations which can improve the state of preparedness of languages to develop useful technologies during the next pandemic.

— Our experiments showed that low-resource Indian languages (such as Marathi, Bengali) were benefited due to the presence of a geographically and/or linguistically closely related well-resourced language (Hindi). This notion of such "bridge" languages has been explored before in the context of MT (Paul et al., 2013) and zero/few-shot transfer in MMLMs (Lauscher et al., 2020). We recommend the community to target bridge languages for the regions that are currently poorly prepared from an LT perspective.

— Drawing insights from the brittleness of MT for domain-specific terms (*airborne, incubation*) or newly-coined terms (*COVID*), we believe that commercial and open-source bot frameworks can benefit from domain adaptation techniques (Chu and Wang, 2018), or techniques to inject new terms to existing solutions.

Our study confirms that except English, only a few European and Asian languages push forward the state-of-the-art research in LT for healthcare. Our preliminary investigation suggests that instead of demographic demand, it is the economic prowess of the users of a language that drives the investment towards developing sophisticated LT solutions for a given language. For instance, Swahili, even though considered as the *lingua franca* of Africa, is still under-served by commercial chatbot frameworks. Similar trends were observed for Hausa which has a considerably large speaker base compared to Dutch (resource-rich)[12].

We believe that these findings will play a crucial role in making the community aware of the disparity that needs to be addressed before the next pandemic hits.

[11] Infoplease Languages Spoken in Each Country of the World: https://bit.ly/3HoAs9K

[12] https://en.wikipedia.org/wiki/List_of_languages_by_number_of_native_speakers

References

Roee Aharoni, Melvin Johnson, and Orhan Firat. 2019. Massively multilingual neural machine translation. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 3874–3884, Minneapolis, Minnesota. Association for Computational Linguistics.

Nazakat Ali. 2020. Chatbot: A conversational agent employed with named entity recognition model using artificial neural network. *CoRR*, abs/2007.04248.

Emily Alsentzer, John Murphy, William Boag, Wei-Hung Weng, Di Jindi, Tristan Naumann, and Matthew McDermott. 2019. Publicly available clinical BERT embeddings. In *Proceedings of the 2nd Clinical Natural Language Processing Workshop*, pages 72–78, Minneapolis, Minnesota, USA. Association for Computational Linguistics.

Eiman Alsharhan and Allan Ramsay. 2020. Investigating the effects of gender, dialect, and training size on the performance of arabic speech recognition. *Language Resources and Evaluation*, 54(4):975–998.

Antonios Anastasopoulos, Alessandro Cattelan, Zi-Yi Dou, Marcello Federico, Christian Federmann, Dmitriy Genzel, Franscisco Guzmán, Junjie Hu, Macduff Hughes, Philipp Koehn, Rosie Lazar, Will Lewis, Graham Neubig, Mengmeng Niu, Alp Öktem, Eric Paquin, Grace Tang, and Sylwia Tur. 2020. TICO-19: the translation initiative for COvid-19. In *Proceedings of the 1st Workshop on NLP for COVID-19 (Part 2) at EMNLP 2020*, Online. Association for Computational Linguistics.

Alexandra Balahur and Marco Turchi. 2012. Multilingual sentiment analysis using machine translation? In *Proceedings of the 3rd Workshop in Computational Approaches to Subjectivity and Sentiment Analysis*, pages 52–60, Jeju, Korea. Association for Computational Linguistics.

Ankur Bapna, Isaac Caswell, Julia Kreutzer, Orhan Firat, Daan van Esch, Aditya Siddhant, Mengmeng Niu, Pallavi Baljekar, Xavier Garcia, Wolfgang Macherey, Theresa Breiner, Vera Axelrod, Jason Riesa, Yuan Cao, Mia Xu Chen, Klaus Macherey, Maxim Krikun, Pidong Wang, Alexander Gutkin, Apurva Shah, Yanping Huang, Zhifeng Chen, Yonghui Wu, and Macduff Hughes. 2022. Building machine translation systems for the next thousand languages.

Moumita Basu, Anurag Shandilya, Prannay Khosla, Kripabandhu Ghosh, and Saptarshi Ghosh. 2019. Extracting resource needs and availabilities from microblogs for aiding post-disaster relief operations. *IEEE Transactions on Computational Social Systems*, 6(3):604–618.

Karthik S Bhat, Mohit Jain, and Neha Kumar. 2021. Infrastructuring telehealth in (in)formal patient-doctor contexts. *Proc. ACM Hum.-Comput. Interact.*, 5(CSCW2).

Damián Blasi, Antonios Anastasopoulos, and Graham Neubig. 2021. Systematic inequalities in language technology performance across the world's languages.

Daniel Braun, Adrian Hernandez Mendez, Florian Matthes, and Manfred Langen. 2017. Evaluating natural language understanding services for conversational question answering systems. In *Proceedings of the 18th Annual SIGdial Meeting on Discourse and Dialogue*, pages 174–185, Saarbrücken, Germany. Association for Computational Linguistics.

Mathilde Caron, Hugo Touvron, Ishan Misra, Hervé Jégou, Julien Mairal, Piotr Bojanowski, and Armand Joulin. 2021. Emerging properties in self-supervised vision transformers.

Tianqi Chen and Carlos Guestrin. 2016. Xgboost: A scalable tree boosting system. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '16, page 785–794, New York, NY, USA. Association for Computing Machinery.

Chenhui Chu and Rui Wang. 2018. A survey of domain adaptation for neural machine translation. In *Proceedings of the 27th International Conference on Computational Linguistics*, pages 1304–1319, Santa Fe, New Mexico, USA. Association for Computational Linguistics.

Alexis Conneau, Kartikay Khandelwal, Naman Goyal, Vishrav Chaudhary, Guillaume Wenzek, Francisco Guzmán, Edouard Grave, Myle Ott, Luke Zettlemoyer, and Veselin Stoyanov. 2020. Unsupervised cross-lingual representation learning at scale. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 8440–8451, Online. Association for Computational Linguistics.

Alexis Conneau, Ruty Rinott, Guillaume Lample, Adina Williams, Samuel Bowman, Holger Schwenk, and Veselin Stoyanov. 2018. XNLI: Evaluating cross-lingual sentence representations. In *Proceedings of EMNLP 2018*, pages 2475–2485.

UN Department of Operational Support. 2020. Covid-19 frequently asked questions.

Sharmishta Suhas Desai. 2021. Chatbot for covidvaccine using deep learning.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, Minneapolis, Minnesota. Association for Computational Linguistics.

Thiago Castro Ferreira, Milena Soriano Marcolino, Isaias Ramos, Raquel Oliveira Prates, Leonardo B. Ribeiro, Zilma Reis, Adriana Silvina Pagano, Wagner Meira, and Antônio Luiz Pinho Ribeiro. 2020. Ana: A brazilian chatbot assistant about covid-19.

Promila Ghosh, M. Raihan, Md. Tanvir Islam, and Md. Ekhlasur Rahaman. 2019. Safeguard: A prototype of an application programming interface to save the disaster affected people. In *2019 10th International Conference on Computing, Communication and Networking Technologies (ICCCNT)*, pages 1–5.

Yu Gu, Robert Tinn, Hao Cheng, Michael Lucas, Naoto Usuyama, Xiaodong Liu, Tristan Naumann, Jianfeng Gao, and Hoifung Poon. 2021. Domain-specific language model pretraining for biomedical natural language processing. *ACM Trans. Comput. Healthcare*, 3(1).

Ankur Gupta, Yash Varun, Prarthana Das, Nithya Muttineni, Parth Srivastava, Hamim Zafar, Tanmoy Chakraborty, and Swaprava Nath. 2021. Truthbot: An automated conversational tool for intent learning, curated information presenting, and fake news alerting.

George F Jenks. 1967. The data model concept in statistical mapping. *International yearbook of cartography*, 7:186–190.

Shaoxiong Ji, Tianlin Zhang, Luna Ansari, Jie Fu, Prayag Tiwari, and E. Cambria. 2021. Mentalbert: Publicly available pretrained language models for mental healthcare. *ArXiv*, abs/2110.15621.

Pratik Joshi, Sebastin Santy, Amar Budhiraja, Kalika Bali, and Monojit Choudhury. 2020. The state and fate of linguistic diversity and inclusion in the NLP world. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 6282–6293, Online. Association for Computational Linguistics.

Timothy J Judson, Anobel Y Odisho, Jerry J Young, Olivia Bigazzi, David Steuer, Ralph Gonzales, and Aaron B Neinstein. 2020a. Implementation of a digital chatbot to screen health system employees during the COVID-19 pandemic. *Journal of the American Medical Informatics Association*, 27(9):1450–1455.

Timothy J. Judson, Anobel Y. Odisho, Jerry J Young, Olivia Bigazzi, David J. Steuer, Ralph Gonzales, and Aaron B. Neinstein. 2020b. Implementation of a digital chatbot to screen health system employees during the covid-19 pandemic. *Journal of the American Medical Informatics Association : JAMIA*, 27:1450 – 1455.

Dipesh Kadariya, Revathy Venkataramanan, Hong Yung Yip, Maninder Kalra, Krishnaprasad Thirunarayanan, and Amit Sheth. 2019. kbot: Knowledge-enabled personalized chatbot for asthma self-management. In *2019 IEEE International Conference on Smart Computing (SMARTCOMP)*, pages 138–143.

Takeshi Kamita, Tatsuya Ito, Atsuko Matsumoto, Tsunetsugu Munakata, and Tomoo Inoue. 2019. A chatbot system for mental healthcare based on sat counseling method. *Mobile Information Systems*, 2019:9517321.

Guillaume Lample and Alexis Conneau. 2019. Cross-lingual language model pretraining. In *NeurIPS*.

Anne Lauscher, Vinit Ravishankar, Ivan Vulić, and Goran Glavaš. 2020. From zero to hero: On the limitations of zero-shot language transfer with multilingual Transformers. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 4483–4499, Online. Association for Computational Linguistics.

Jinhyuk Lee, Wonjin Yoon, Sungdong Kim, Donghyeon Kim, Sunkyu Kim, Chan Ho So, and Jaewoo Kang. 2020. Biobert: a pre-trained biomedical language representation model for biomedical text mining. *Bioinformatics*, 36:1234 – 1240.

William Lewis. 2010. Haitian Creole: How to build and ship an MT engine from scratch in 4 days, 17 hours, & 30 minutes. In *Proceedings of the 14th Annual conference of the European Association for Machine Translation*, Saint Raphaël, France. European Association for Machine Translation.

Yunyao Li, Tyrone Grandison, Patricia Silveyra, Ali Douraghy, Xinyu Guan, Thomas Kieselbach, Chengkai Li, and Haiqi Zhang. 2020. Jennifer for COVID-19: An NLP-powered chatbot built for the people and by the people to combat misinformation. In *Proceedings of the 1st Workshop on NLP for COVID-19 at ACL 2020*, Online. Association for Computational Linguistics.

John M. Lipski. Castile and the hydra: the diversification of spanish in latin america.

John M. Lipski. 2014. *2. The Many Facets of Spanish Dialect Diversification in Latin America*, pages 38–75. University of Chicago Press.

Patrick Littell, David R. Mortensen, Ke Lin, Katherine Kairis, Carlisle Turner, and Lori Levin. 2017. URIEL and lang2vec: Representing languages as typological, geographical, and phylogenetic vectors. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 2, Short Papers*, pages 8–14, Valencia, Spain. Association for Computational Linguistics.

Ilya Loshchilov and Frank Hutter. 2019. Decoupled weight decay regularization. In *International Conference on Learning Representations*.

Lucy Lu Wang, Kyle Lo, Yoganand Chandrasekhar, Russell Reas, Jiangjiang Yang, Darrin Eide, Kathryn Funk, Rodney Kinney, Ziyang Liu, William Merrill, Paul Mooney, Dewey Murdick, Devvret Rishi, Jerry Sheehan, Zhihong Shen, Brandon Stilson, Alex D. Wade, Kuansan Wang, Chris Wilhelm, Boya

Xie, Douglas Raymond, Daniel S. Weld, Oren Etzioni, and Sebastian Kohlmeier. 2020. Cord-19: The covid-19 open research dataset. *ArXiv*, page arXiv:2004.10706v2. 32510522[pmid].

Aymen Ben Elhaj Mabrouk, Moez Ben Haj Hmida, Chayma Fourati, Hatem Haddad, and Abir Messaoudi. 2021. A multilingual african embedding for faq chatbots. *ArXiv*, abs/2103.09185.

Theodora A. Maniou and Andreas Veglis. 2020. Employing a chatbot for news dissemination during crisis: Design, implementation and evaluation. *Future Internet*, 12(7).

Alistair Martin, Jama Nateqi, Stefanie Gruarin, Nicolas Munsch, Isselmou Abdarahmane, and Bernhard Knapp. 2020a. An artificial intelligence-based first-line defence against covid-19: digitally screening citizens for risks via a chatbot. *bioRxiv*.

Alistair Martin, Jama Nateqi, Stefanie Gruarin, Nicolas Munsch, Isselmou Abdarahmane, Marc Zobel, and Bernhard Knapp. 2020b. An artificial intelligence-based first-line defence against covid-19: digitally screening citizens for risks via a chatbot. *Scientific Reports*, 10(1):19012.

Fahad Mehfooz, Sakshi Jha, Sahil Singh, Shreya Saini, and Nidhi Sharma. 2020. Medical chatbot for novel covid-19. *ICT Analysis and Applications*, 154:423 – 430.

Do June Min, Veronica Perez-Rosas, Shihchen Kuo, William H. Herman, and Rada Mihalcea. 2020. Upstage: Unsupervised context augmentation for utterance classification in patient-provider communication. In *Proceedings of the 5th Machine Learning for Healthcare Conference*, volume 126 of *Proceedings of Machine Learning Research*, pages 895–912. PMLR.

Robert R Morris, Kareem Kouddous, Rohan Kshirsagar, and Stephen M Schueller. 2018. Towards an artificially empathic conversational agent for mental health applications: System design and user perceptions. *J Med Internet Res*, 20(6):e10148.

Trang Nguyen and Maxim Shcherbakov. 2021. Enhancing rasa nlu model for vietnamese chatbot. 9:33–36.

David Oniani and Yanshan Wang. 2020. A qualitative evaluation of language models on automatic question-answering for covid-19. In *Proceedings of the 11th ACM International Conference on Bioinformatics, Computational Biology and Health Informatics*, BCB '20, New York, NY, USA. Association for Computing Machinery.

Xiaoman Pan, Boliang Zhang, Jonathan May, Joel Nothman, Kevin Knight, and Heng Ji. 2017. Cross-lingual name tagging and linking for 282 languages. In *Proceedings of ACL 2017*, pages 1946–1958.

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. Bleu: a method for automatic evaluation of machine translation. In *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics*, pages 311–318, Philadelphia, Pennsylvania, USA. Association for Computational Linguistics.

Ho-yeon Park, Gun-doo Moon, and Kyoung-jae Kim. 2020. Classification of covid-19 symptom for chatbot using bert. *Solid State Technology*, 63(6):19185–19188.

SoHyun Park, Jeewon Choi, Sungwoo Lee, Changhoon Oh, Changdai Kim, Soohyun La, Joonhwan Lee, and Bongwon Suh. 2019. Designing a chatbot for a brief motivational interview on stress management: Qualitative case study. *J Med Internet Res*, 21(4):e12231.

Michael Paul, Andrew Finch, and Eiichrio Sumita. 2013. How to choose the best pivot language for automatic translation of low-resource languages. *ACM Transactions on Asian Language Information Processing*, 12(4).

Praveen Prasannan, Stephy Joseph, and Rajeev R R. 2020. A chatbot in Malayalam using hybrid approach. In *Proceedings of the 17th International Conference on Natural Language Processing (ICON): System Demonstrations*, pages 28–29, Patna, India. NLP Association of India (NLPAI).

Ban Quy Tran, Thai Van Nguyen, Thang Duc Phung, Viet Tan Nguyen, Dat Duy Tran, and Son Tung Ngo. 2021. Fu covid-19 ai agent built on attention algorithm using a combination of transformer, albert model, and rasa framework. In *2021 10th International Conference on Software and Computer Applications*, ICSCA 2021, page 22–31, New York, NY, USA. Association for Computing Machinery.

Sebastian Ruder, Noah Constant, Jan Botha, Aditya Siddhant, Orhan Firat, Jinlan Fu, Pengfei Liu, Junjie Hu, Dan Garrette, Graham Neubig, and Melvin Johnson. 2021. Xtreme-r: Towards more challenging and nuanced multilingual evaluation.

Koustav Rudra, Subham Ghosh, Niloy Ganguly, Pawan Goyal, and Saptarshi Ghosh. 2015. Extracting situational information from microblogs during disaster events: A classification-summarization approach. In *Proceedings of the 24th ACM International on Conference on Information and Knowledge Management*, CIKM '15, page 583–592, New York, NY, USA. Association for Computing Machinery.

Shijing Si, Rui Wang, Jedrek Wosik, Hao Zhang, David Dov, Guoyin Wang, and Lawrence Carin. 2020. Students need more attention: Bert-based attention model for small data with application to automatic patient message triage. In *Proceedings of the 5th Machine Learning for Healthcare Conference*, volume 126 of *Proceedings of Machine Learning Research*, pages 436–456. PMLR.

Aditya Siddhant, Ankur Bapna, Orhan Firat, Yuan Cao, Mia Xu Chen, Isaac Caswell, and Xavier Garcia. 2022. Towards the next 1000 languages in multilingual machine translation: Exploring the synergy between supervised and self-supervised learning. *CoRR*, abs/2201.03110.

J.Jinu Sophia and T.Prem Jacob. 2021. Edubot-a chatbot for education in covid-19 pandemic and vqabot comparison. In *2021 Second International Conference on Electronics and Sustainable Communication Systems (ICESC)*, pages 1707–1714.

Stephanie Strassel and Jennifer Tracey. 2016. LORELEI language packs: Data, tools, and resources for technology development in low resource languages. In *Proceedings of the Tenth International Conference on Language Resources and Evaluation (LREC'16)*, pages 3273–3280, Portorož, Slovenia. European Language Resources Association (ELRA).

Elior Sulem, Omri Abend, and Ari Rappoport. 2018. BLEU is not suitable for the evaluation of text simplification. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 738–744, Brussels, Belgium. Association for Computational Linguistics.

Nguyen Thi Thu Trang and Maxim Shcherbakov. 2021. Enhancing rasa nlu model for vietnamese chatbot.

Meng-Han Tsai, James Yichu Chen, and Shih-Chung Kang. 2019. Ask diana: A keyword-based chatbot system for water-related disaster management. *Water*, 11(2).

Alex Wang, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel Bowman. 2018. GLUE: A multi-task benchmark and analysis platform for natural language understanding. In *Proceedings of the 2018 EMNLP Workshop BlackboxNLP: Analyzing and Interpreting Neural Networks for NLP*, pages 353–355, Brussels, Belgium. Association for Computational Linguistics.

Shijie Wu and Mark Dredze. 2020. Are all languages created equal in multilingual BERT? In *Proceedings of the 5th Workshop on Representation Learning for NLP*, pages 120–130, Online. Association for Computational Linguistics.

Linting Xue, Noah Constant, Adam Roberts, Mihir Kale, Rami Al-Rfou, Aditya Siddhant, Aditya Barua, and Colin Raffel. 2021. mT5: A massively multilingual pre-trained text-to-text transformer. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 483–498, Online. Association for Computational Linguistics.

Yinfei Yang, Yuan Zhang, Chris Tar, and Jason Baldridge. 2019. PAWS-X: A cross-lingual adversarial dataset for paraphrase identification. In *Proceedings of EMNLP 2019*, pages 3685–3690.

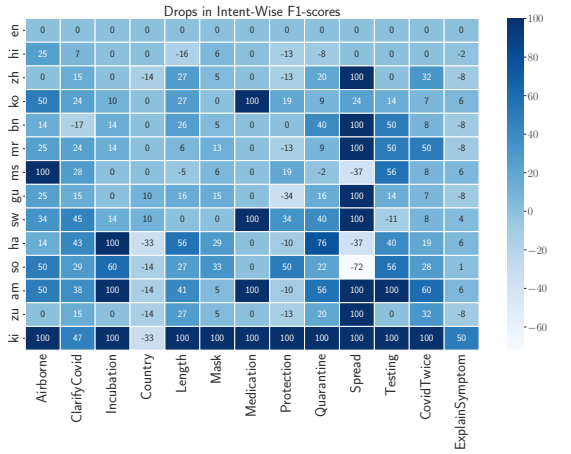

Figure 4: Relative drops (relative to English) in intent wise F1 scores for different languages in the *Train on English* setup (in LUIS). Negative values indicate increase in the scores relative to English.

Rabih Zbib, Erika Malchiodi, Jacob Devlin, David Stallard, Spyros Matsoukas, Richard Schwartz, John Makhoul, Omar F. Zaidan, and Chris Callison-Burch. 2012. Machine translation of Arabic dialects. In *Proceedings of the 2012 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 49–59, Montréal, Canada. Association for Computational Linguistics.

Rong Zhang, Revanth Reddy Gangi Reddy, Md Arafat Sultan, Vittorio Castelli, Anthony Ferritto, Radu Florian, Efsun Sarioglu Kayi, Salim Roukos, Avirup Sil, and Todd Ward. 2020. Multi-stage pretraining for low-resource domain adaptation. *ArXiv*, abs/2010.05904.

A Example Appendix

A.1 Intent Definitions and Descriptions

The different intents used for our experiments are described in table 5. We provide definitions and examples for each of the different intents used.

A.2 Bot Building Strategies

A.3 MMLM Training Setup

For our experiments with Multilingual Pre-trained Transformers we consider mBERT (*bert-base-multilingual-cased*) and XLMR (*xlm-roberta-base*) for training intent classifiers. As mentioned in the main text we explore two methodologies to train and evaluate these MMLMs, a detailed description with hyperparameters is given below:

1. KNN using Pre-trained Embeddings:
Since the scale of our data is on the lower side, training an end-to-end classifier might be prone to

Intent Type	Example in English	Definition
Airborne	Can the virus that causes COVID-19 be transmitted through the air?	Queries related to how much COVID is carried by air
ClarifyCovid	How do I know if it is COVID-19 or just the flu?	Queries related to difference betwen COVID and other diseases
Incubation	Can someone in incubation infect other people?	Queries related to situations where a person is infected with COVID and is going through incubation phase
Length	How long does the illness make you poorly for?	Queries regarding longevity of COVID infection
Mask	Should I wear a mask while exercising?	Queries about wearing mask
Protection	Ways to keep safe from COVID-19	Queries about the ways of protection from COVID
Quarantine	Will I avoid coronavirus, if I self-isolate?	Queries about the effect of quarantining after getting infected with COVID
Spread	Aside from inhalation, are there other ways coronavirus can spread?	Queries about the spreading process of COVID
Testing	Where can I get my test done?	Queries about the testing process of COVID
CovidTwice	If you get COVID-19, can you get it again?	Queries about whether COVID can infect someone more than once
ExplainSymptom	I have a sharp pain here in the chest	User explaining COVID related symptoms
Country	How many people in Italy have COVID-19?	Querying about the statistics of infection in different countries
Medication	Do any of the drugs reduce mortality?	Querying about the medication to survive from COVID
Treatment	Which vaccines are good to protect against the virus?	Querying about the treatment strategies associated with COVID

Table 5: Different intents with definitions and examples present in our dataset.

Bot Building Setup	Training Strategy	Testing Strategy
Train on English Data	Train set comprises of the English queries	Test set comprises of English queries where the manually written queries in target language are translated to English using MT system
Train on MT Translations	Train set comprises of the English queries translated to target language using MT System	Test set comprises of manually written queries in target language
Train on Manual Translations	Train set comprises of manually written queries in target language	Test set comprises of manually written queries in target language

Table 6: Different strategies for building the chatbots.

over-fitting. We fit a k-Nearest Neighbors (KNN) classifier on the sentence embeddings obtained using the pre-trained model for the queries in training data. At test time, we similarly obtain the representation of the user query and find its nearest neighbors among the training queries to predict its intent. The optimal value for k was empirically found to be 1 and for sentence embeddings, we take the average of the representation of each token of the sentence in the last layer of MMLM.

We also tried fine-tuning the pre-trained model with the training queries using a Masked Language Modelling (MLM) objective. Additionally, we also fine-tuning on a much larger COVID-19 queries dataset in english : COQB (Li et al., 2020) along with our training queries, as has been pointed by Lauscher et al. (2020) can be an effective strategy for few shot transfer. We use 3 epochs to fine-tune the models with a learning rate of 5e-5 and Adam-W optimizer (Loshchilov and Hutter, 2019). A masking probability of 15% was used during the MLM training and maximum sequence length was taken to be 32.

2. Fine-tuning an End to End Classifier :
We also try fine-tuning the MMLMs end-to-end by adding a classification head on top of the pre-trained network to classify the input query into one of the 14 intents. We adapt the sequence classification scripts for GLUE benchmark (Wang et al., 2018) provided by hugging face[13] on our dataset. We fine-tune the classifier for 20 epochs, with the same learning rate and optimizer as the MLM fine-tuning in the first point with a batch size of 8.

For every language we use the best accuracies

[13] https://huggingface.co/transformers/v2.9.1/examples.html

		Medical			Non-Medical				
Lang	Bot	Covid	PhysicalScience	Disease	BodyPart	Country	Duration	Protection	InfoSource
Hi	DF	24.6	50.1	+3	32	7.52	5.2	11.3	+30.1
	LUIS	24.56	+41	94	21	5.31	+4	12.06	+72.72
Bn	DF	42.9	43.1	52	34	6.3	8.9	20.4	8.34

Table 7: Relative drop in entity-type wise F1-score in *Entity Recognition* task using DialogFlow (DF) and LUIS.

obtain from either of these two strategies [14]. All the experiments were run on 4 NVIDIA V-100 GPUs with 32 GB memory.

A.4 Language Readiness Analysis

Results and Analysis

Initially, we have plotted the readiness measures of each language used in our training data on the scatter plot in Figure 5 with the language class on X-axis and readiness measure in Y-Axis. It clearly shows that the African languages such as `Somali`, `Amharic`, `Hausa`, `Zulu` are below the trend-line in terms of readiness. In fact, some of the European languages such as `Icelandic`, `Hungarian`, `Estonic`, `Finnish` also require some attention. Primarily, we observe that the readiness measure is not a direct function of the language class from this plot. As we can see that even though majority of the class 4 and 5 are near the trend line, the observation is similar for Class 1 as well.

Therefore, we also resort to understanding how much does the trend hold true for the language families of these corresponding languages? So, we approximate each of the language family by taking the average scores of each language falling into that class and plot those in Figure 6. It was interesting to observe that the English-major language families such as `Austroasiatic`, `Koreanic` and `Sino-Tibetan` are well-served, and consequently lie above the trend line. Overall, `Indo-European` language families are well near the trend line and then the resource-poor language families are `Afroasiatic`, `Niger-Congo` and `Uralic`, the worst being the `Afroasiatic` language family.

A.5 Details on Language Readiness Prediction

In section 4, we discussed the estimation of readiness values of different languages. We first extended our 16 languages that we considered for intent recognition experiments with proxy scores for an additional 11 languages, namely, `French (fr)`, `Arabic (ar)`, `German (de)`, `Spanish (es)`, `Portuguese (pr)`, `Vietnamese (vi)`, *Hungarian (hu)*, `Finnish (fi)`, `Czech (cs)`, `Estonian (et)`, and `Icelandic (is)`. Finally, it covers six primary language families in the world, such as: 1) `Indo-European`, 2) `Sino-Tibetan`, 3) `Afroasiatic`, 4) `Niger-Congo`, 5) `Koreanic` and 6) `Austroasiatic`. To estimate the readiness values of the remaining 116 languages supported by the Translators (Google and Bing) and MMLMs (mBERT and XLMR), we used the available readiness data for the 27 to build a regression model. We used Gaussian Processes to model the readiness prediction problem, due its efficiency on the small sized datasets. Radial Basis Function (RBF) with added noise level for each instance (White Kernel), was used, and the *length scale* of RBF and noise level were tuned using L-BFGS algorithm with 5 restarts for the optimizer. The model selection was done using a Leave One Out strategy, where we move one language to validation set and train on the remaining, repeating this for all the languages and measuring average accuracy. Besides Gaussian Process Regression (GPR), we also experimented with Linear Regression, Lasso Regression and XGBoost (Chen and Guestrin, 2016), but observed inferior validation accuracies.

A.6 Global Pandemic Readiness Measurement

In section 4 of the paper, we have talked about how to actually take the speaker-base values into account while calculating the readiness scores for each country in the world and the final r_l scores obtained on all the languages are used to extrapolate the readiness of each country c in the world. We had also experimented in a way such that all the languages spoken in the country are weighted

[14] technically 4, as in the KNN case we consider no fine-tuning, fine-tuning on Train Queries, and fine-tuning on Train and COQB queries

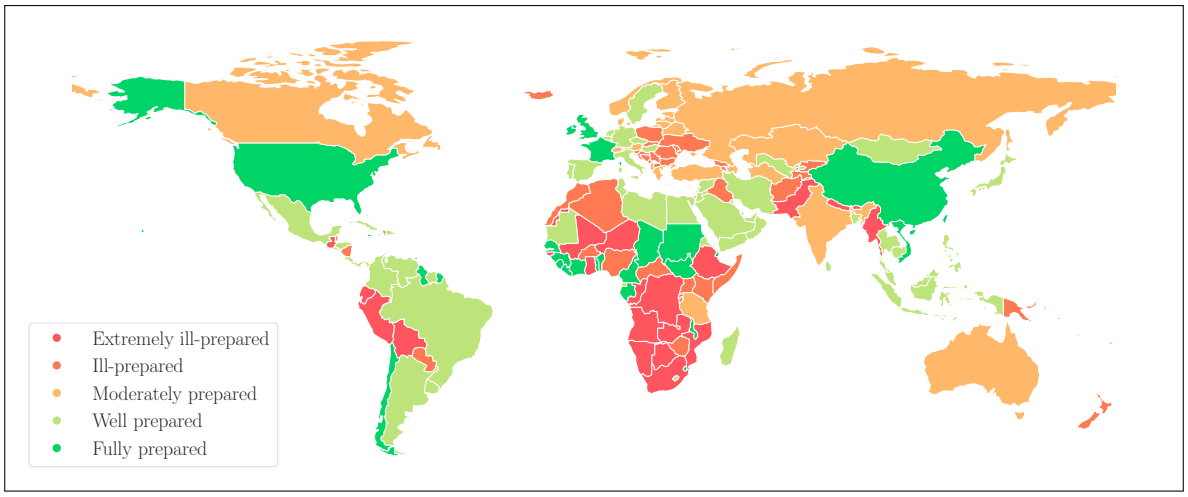

Figure 5: World Map showing the Readiness of each country in terms of combating the next pandemic using LT. Their Levels of Preparedness are shown as legends in the bottom left corner. This map was generated by providing uniform weightage to all the languages spoken in a country, i.e. excluding the percentage of speaker-base for a particular language in a country (Readiness measure calculated using Equation 2).

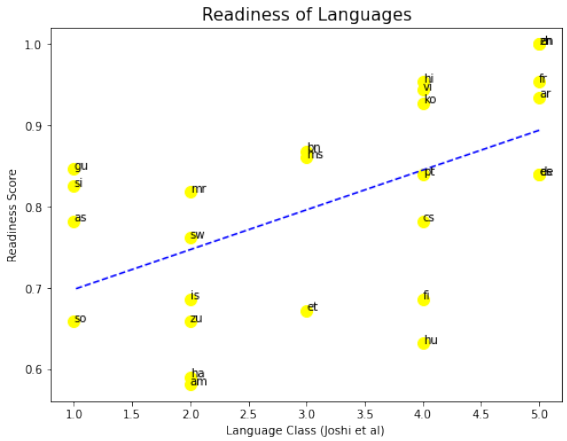

Figure 6: shows the readiness scores of the languages which are used in our training data for readiness measurement using GPR

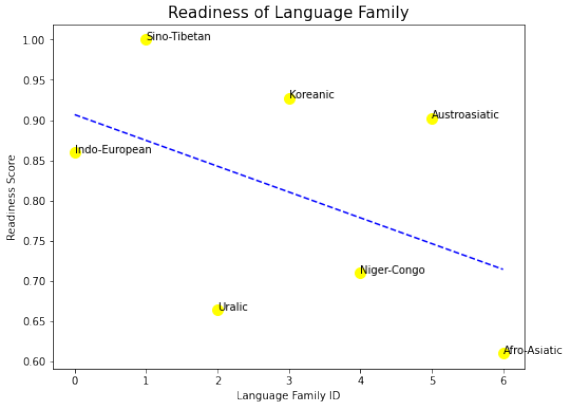

Figure 7: shows the readiness scores of the language families of the corresponding languages which are used in our training data for readiness measurement using GPR

equally while calculating the readiness of a country. This is similar to the linguistic utility defined by Blasi et al. (2021) in their work for a country c we calculate linguistic readiness r_c^{ling} as:

$$r_c^{ling} = \frac{1}{|\mathcal{L}_c|} \sum_{l \in \mathcal{L}_c} r_l \quad (1)$$

The r_c^{ling} values have been plotted in Figure 5. Based on our observations on these values we make the following observations highlighting the difference between demographic and linguistic readiness of different countries.

Observations: The map shown in 5 provides us an idea of how each country in the world would be able to effectively combat the pandemic by leveraging LT solutions. However, this is when we are actually considering uniform speaker-base for each language in a country. Overall, it can be observed that some of the Asian countries like India falls in the ***moderately prepared*** zone now which was initially treated as ***well prepared***. This is due to the presence of class 4 language Hindi (having a readiness score of 0.9536) with a considerably high speaker-base (46.19%). Also, similar trend is observed in Canada (home to the speakers of various languages like `English`, `French`, `Punjabi`, `Italian`, `Spanish`, `German`, `Cantonese`, `Arabic`, `Tagalog`).

Adapting Pre-trained Language Models to African Languages via Multilingual Adaptive Fine-Tuning

Jesujoba O. Alabi[*], **David Ifeoluwa Adelani**[*], **Marius Mosbach, and Dietrich Klakow**
Spoken Language Systems (LSV), Saarland University, Saarland Informatics Campus, Germany
{jalabi,didelani,mmosbach,dklakow}@lsv.uni-saarland.de

Abstract

Multilingual pre-trained language models (PLMs) have demonstrated impressive performance on several downstream tasks for both high-resourced and low-resourced languages. However, there is still a large performance drop for languages unseen during pre-training, especially African languages. One of the most effective approaches to adapt to a new language is *language adaptive fine-tuning* (LAFT) — fine-tuning a multilingual PLM on monolingual texts of a language using the pre-training objective. However, adapting to a target language individually takes a large disk space and limits the cross-lingual transfer abilities of the resulting models because they have been specialized for a single language. In this paper, we perform *multilingual adaptive fine-tuning* on 17 most-resourced African languages and three other high-resource languages widely spoken on the African continent to encourage cross-lingual transfer learning. To further specialize the multilingual PLM, we removed vocabulary tokens from the embedding layer that corresponds to non-African writing scripts before MAFT, thus reducing the model size by around 50%. Our evaluation on two multilingual PLMs (AfriBERTa and XLM-R) and three NLP tasks (NER, news topic classification, and sentiment classification) shows that our approach is competitive to applying LAFT on individual languages while requiring significantly less disk space. Additionally, we show that our adapted PLM also improves the zero-shot cross-lingual transfer abilities of parameter efficient fine-tuning methods.

1 Introduction

Recent advances in the development of multilingual pre-trained language models (PLMs) like mBERT (Devlin et al., 2019), XLM-R (Conneau et al., 2020), and RemBERT (Chung et al., 2021) have led to significant performance gains on a wide range of cross-lingual transfer tasks. Due to the *curse of multilinguality* (Conneau et al., 2020) — a trade-off between language coverage and model capacity — and non-availability of pre-training corpora for many low-resource languages, multilingual PLMs are often trained on about 100 languages. Despite the limitations of language coverage, multilingual PLMs have been shown to transfer to several low-resource languages unseen during pre-training. Although, there is still a large performance gap compared to languages seen during pre-training.

One of the most effective approaches to adapt to a new language is *language adaptive fine-tuning* (LAFT) — fine-tuning a multilingual PLM on monolingual texts in the target language using the same pre-training objective. This has been shown to lead to big gains on many cross-lingual transfer tasks (Pfeiffer et al., 2020), and low-resource languages (Muller et al., 2021; Chau & Smith, 2021), including African languages (Alabi et al., 2020; Adelani et al., 2021). Nevertheless, adapting a model to each target language individually takes large disk space, and limits the cross-lingual transfer abilities of the resulting models because they have been specialized to individual languages (Beukman, 2021).

An orthogonal approach to improve the coverage of low-resource languages is to include them in the pre-training data. An example for this approach is AfriBERTa (Ogueji et al., 2021), which was trained from scratch on 11 African languages. A downside of this approach is that it is resource intensive in terms of data and compute.

Another alternative approach is parameter efficient fine-tuning like Adapters (Pfeiffer et al., 2020) and sparse fine-tuning (Ansell et al., 2021), where the model is adapted to new languages by using a sparse network trained on a small monolingual corpus. Similar to LAFT, it requires adaptation for every new target language. Although it takes little disk space, all target language-specific parameters need to be stored.

In this paper, we propose *multilingual adaptive fine-tuning (MAFT)*, a language adaptation to multiple languages at once. We perform language adaptation on the 17 most-resourced African languages (Afrikaans, Amharic, Hausa, Igbo, Malagasy, Chichewa, Oromo, Naija, Kinyarwanda, Kirundi, Shona, Somali, Sesotho, Swahili, isiXhosa, Yorùbá, isiZulu) and three other high-resource language widely spoken on the continent

[*] Equal contribution.

Proceedings of the 29th International Conference on Computational Linguistics, pages 4336–4349
October 12–17, 2022.

(English, French, and Arabic) *simultaneously* to provide a single model for cross-lingual transfer learning for African languages. To further specialize the multilingual PLM, we follow the approach of Abdaoui et al. (2020) and remove vocabulary tokens from the embedding layer that correspond to non-Latin and non-Ge'ez (used by Amharic) scripts before MAFT, thus effectively reducing the model size by 50%.

Our evaluation on two multilingual PLMs (AfriBERTa and XLM-R) and three NLP tasks (NER, news topic classification and sentiment classification) shows that our approach is competitive to performing LAFT on the individual languages, with the benefit of having a single model instead of a separate model for each of the target languages. Also, we show that our adapted PLM improves the zero-shot cross-lingual transfer abilities of parameter efficient fine-tuning methods like Adapters (Pfeiffer et al., 2020) and sparse fine-tuning (Ansell et al., 2021).

As an additional contribution, and in order to cover more diverse African languages in our evaluation, we create a new evaluation corpus, ANTC – **A**frican **N**ews **T**opic **C**lassification – for Lingala, Somali, Naija, Malagasy, and isiZulu from pre-defined news categories of VOA, BBC, Global Voices, and Isolezwe newspapers. To further the research on NLP for African languages, we make our code and data publicly available.[1] Additionally, our models are available via HuggingFace.[2]

2 Related Work

Multilingual PLMs for African languages The success of multilingual PLMs such as mBERT (Devlin et al., 2019) and XLM-R (Conneau et al., 2020) for cross-lingual transfer in many natural language understanding tasks has encouraged the continuous development of multilingual models (Luo et al., 2021; Chi et al., 2021; Ouyang et al., 2021; Chung et al., 2021; He et al., 2021). Most of these models cover 50 to 110 languages and only few African languages are represented due to lack of large monolingual corpora on the web. To address this under-representation, regional multilingual PLMs have been trained from scratch such as AfriBERTa (Ogueji et al., 2021) or adapted from existing multilingual PLM through LAFT (Alabi et al., 2020; Pfeiffer et al., 2020; Muller et al., 2021; Adelani et al., 2021). AfriBERTa is a relatively small multilingual PLM (126M parameters) trained using the RoBERTa architecture and pre-training objective on 11 African languages. However, it lacks coverage of languages from the southern region of the African continent, specifically the southern-Bantu languages. In our work, we extend to those languages since only a few of them have large (>100MB size) monolingual corpus. We also do not specialize to a single language but apply MAFT which allows multilingual adaptation and preserves downstream performance on both high-resource and low-resource languages.

Adaptation of multilingual PLMs It is not unusual for a new multilingual PLM to be initialized from an existing model. For example, Chi et al. (2021) trained InfoXLM by initializing the weights from XLM-R before training the model on a joint monolingual and translation corpus. Although they make use of a new training objective during adaptation. Similarly, Tang et al. (2020) extended the languages covered by mBART (Liu et al., 2020b) from 25 to 50 by first modifying the vocabulary and initializing the model weights of the original mBART before fine-tuning it on a combination of monolingual texts from the original 25 languages in addition to 25 new languages. Despite increasing the number of languages covered by their model, they did not observe a significant performance drop on downstream tasks. We take inspiration from these works for applying MAFT on African languages, but we do not modify the training objective during adaptation nor increase the vocabulary.

Compressing PLMs One of the most effective methods for creating smaller PLMs is distillation where a small student model is trained to reproduce the behaviour of a larger teacher model. This has been applied to many English PLMs (Sanh et al., 2019; Jiao et al., 2020; Sun et al., 2020; Liu et al., 2020a) and a few multilingual PLMs (Wang et al., 2020, 2021). However, it often leads to a drop in performance compared to the teacher PLM. An alternative approach that does not lead to a drop in performance has been proposed by Abdaoui et al. (2020) for multilingual PLM. They removed unused vocabulary tokens from the embedding layer. This simple method significantly reduces the number of embedding parameters thus reducing the overall model size since the embedding layer contributes the most to the total number of model parameters. In our paper, we combine MAFT with the method proposed by Abdaoui et al. (2020) to reduce the overall size of the resulting multilingual PLM for African languages. This is crucial especially because people from under-represented communities in Africa may not have access to powerful GPUs in order to fine-tune large PLMs. Also, Google Colab[3] (free-version), which is widely used by individuals from under-represented communities without access to other compute resources, cannot run large models like e.g. XLM-R. Hence, it is important to provide smaller models that still achieve competitive downstream performance to these communities.

Evaluation datasets for African languages One of the challenges of developing (multilingual) PLMs for African languages is the lack of evaluation corpora. There have been many efforts by communities like

[1] https://github.com/uds-lsv/afro-maft
[2] https://huggingface.co/Davlan
[3] https://colab.research.google.com/

Domain	Number of sentences			Classes	Number of classes
	Train	Dev	Test		
Newly created datasets					
Lingala (lin)	1,536	220	440	Rdc, Politiki/Politique, Bokengi/Securite, Justice, Bokolongono/Santé/Medecine	5
Naija (pcm)	1,165	167	333	Entertainment, Africa, Sport, Nigeria, World	5
Malagasy (mlg)	4544	650	1299	Politika (Politics), Kolontsaina (Culture), Zon'olombelona (Human Rights), Siansa_sy_Teknolojia (Science and Technology) ,Tontolo_iainana (Environment)	5
Somali (som)	10,072	1,440	2879	Soomaaliya (Somalia), Wararka (News), Caalamka (World), Maraykanka (United States), Afrika (Africa)	6
isiZulu (zul)	2,961	424	847	Ezemidlalo (Sports), Ezokungcebeleka (Recreation), Imibono (Ideas), Ezezimoto (Automotive), Intandokazi (Favorites)	5
Existing datasets					
Amharic (amh)	36,029	5,147	10,294	Local News, Sport, Politics, International News, Business, Entertainment	6
English (eng)	114,000	6,000	7,600	World, Sports, Business, Sci/Tech	4
Hausa (hau)	2,045	290	582	Africa, World, Health, Nigeria, Politics	5
Kinyarwanda (kin)	16,163	851	4,254	Politics, Sport, Economy, Health, Entertainment, History, Technology, Tourism, Culture, Fashion, Religion, Environment, Education, Relationship	14
Kiswahili (swa)	21,096	1,111	7,338	Uchumi (Economic), Kitaifa (National), Michezo (Sports), Kimataifa (International), Burudani (Recreation), Afya (Health)	6
Yorùbá (yor)	1,340	189	379	Nigeria, Africa, World, Entertainment, Health, Sport, Politics	7

Table 1: Number of sentences in training, development and test splits. We provide automatic translation of some of the African language words to English (in Parenthesis) using Google Translate.

Masakhane to address this issue (∀ et al., 2020; Adelani et al., 2021). We only find two major evaluation benchmark datasets that cover a wide range of African languages: one for named entity recognition (NER) (Adelani et al., 2021) and one for sentiment classification (Muhammad et al., 2022). In addition, there are also several news topic classification datasets (Hedderich et al., 2020; Niyongabo et al., 2020; Azime & Mohammed, 2021) but they are only available for a few African languages. Our work contributes novel news topic classification datasets (i.e. ANTC) for additional five African languages: Lingala, Naija, Somali, isiZulu, and Malagasy.

3 Data

3.1 Adaptation corpora

We perform MAFT on 17 African languages Afrikaans, Amharic, Hausa, Igbo, Malagasy, Chichewa, Oromo, Naija, Kinyarwanda, Kirundi, Shona, Somali, Sesotho, Swahili, isiXhosa, Yorùbá, isiZulu) covering the major African language families and 3 high resource languages (Arabic, French, and English) widely spoken in Africa. We selected the African languages based on the availability of a (relatively) large amount of monolingual texts. We obtain the monolingual texts from three major sources: the mT5 pre-training corpus which is based on Common Crawl Corpus[4] (Xue et al., 2021), the British Broadcasting Corporation (BBC) News, Voice of America News[5] (Palen-Michel et al., 2022), and some other news websites based in Africa. Table 9 in the Appendix provides a summary of the monolingual data, including their sizes and sources. We pre-processed the data by removing lines that consist of numbers or punctuation only, and lines with less than six tokens.

3.2 Evaluation tasks

We run our experiments on two sentence level classification tasks: news topic classification and sentiment classification, and one token level classification task: NER. We evaluate our models on English as well as diverse African languages with different linguistic characteristics.

3.2.1 Existing datasets

NER For the NER task we evaluate on the MasakhaNER dataset (Adelani et al., 2021), a manually annotated dataset covering 10 African languages (Amharic, Hausa, Igbo, Kinyarwanda, Luganda, Luo, Naija, Kiswahili, Wolof, and Yorùbá) with texts from the news domain. For English, we use data from the CoNLL 2003 NER task (Tjong Kim Sang & De Meulder, 2003) also containing texts from the news domain. For isiXhosa, we use the data from Eiselen (2016). Lastly, to evaluate on Arabic we make use of the ANERCorp dataset (Benajiba et al., 2007; Obeid et al., 2020).

[4] https://commoncrawl.org/

[5] https://www.voanews.com

News topic classification We use existing news topic datasets for Amharic (Azime & Mohammed, 2021), English – AG News corpus – (Zhang et al., 2015), Kinyarwanda – KINNEWS – (Niyongabo et al., 2020), Kiswahili – new classification dataset– (David, 2020), and both Yorùbá and Hausa (Hedderich et al., 2020). For dataset without a development set, we randomly sample 5% of their training instances and use them as a development set.

Sentiment classification We use the NaijaSenti multilingual Twitter sentiment analysis corpus (Muhammad et al., 2022). This is a large code-mixed and monolingual sentiment analysis dataset, manually annotated for 4 Nigerian languages: Hausa, Igbo, Yorùbá and Pidgin. Additionally, we evaluate on the Amharic, and English Twitter sentiment datasets by Yimam et al. (2020) and Rosenthal et al. (2017), respectively. For all datasets above, we only make use of tweets with positive, negative and neutral sentiments.

3.2.2 Newly created dataset: ANTC corpus

We created a novel dataset, ANTC — African News Topic Classification for five African languages. We obtained data from three different news sources: VOA, BBC[6], Global Voices[7], and isolezwe[8]. From the VOA data we created datasets for Lingala and Somali. We obtained the topics from data released by Palen-Michel et al. (2022) and used the provided URLs to get the news category from the websites. For Naija, Malagasy and isiZulu, we scrapped news topic from the respective news website (BBC Pidgin, Global Voices, and isolezwe respectively) directly base on their category. We noticed that some news topics are not mutually exclusive to their categories, therefore, we filtered such topics with multiple labels. Also, we ensured that each category has at least 200 samples. The categories include but are not limited to: Africa, Entertainment, Health, and Politics. The pre-processed datasets were divided into training, development, and test sets using stratified sampling with a ratio of 70:10:20. Table 1 provides details about the dataset size and news topic information.

4 Pre-trained Language Models

For our experiments, we make use of different multilingual PLMs that have been trained using a masked language model objective on large collections of monolingual texts from several languages. Table 2 shows the number of parameters as well as the African languages covered by each of the models we consider.

1. XLM-R (Conneau et al., 2020) has been pre-trained on 100 languages including eight African languages. We make use of both XLM-R-base and XLM-R-large for MAFT with 270M and 550M parameter sizes respectively. Although, for our main experiments, we make use of XLM-R-base.

2. AfriBERTa (Ogueji et al., 2021) has been pre-trained only on African languages. Despite its smaller parameter size (126M), it has been shown to reach competitive performance to XLM-R-base on African language datasets (Adelani et al., 2021; Hedderich et al., 2020).

3. XLM-R-miniLM (Wang et al., 2020) is a distilled version of XLM-R-large with only 117M parameters.

PLM	# Lang.	African languages covered
XLM-R-base (270M)	100	afr, **amh**, **hau**, **mlg**, orm, **som**, **swa**, **xho**
AfriBERTa-large (126M)	11	**amh**, **hau**, **ibo**, **kin**, run, orm, **pcm**, **som**, **swa**, tir, **yor**
XLM-R-miniLM (117M)	100	afr, **amh**, **hau**, **mlg**, orm, **som**, **swa**, **xho**
XLM-R-large (550M)	100	afr, **amh**, **hau**, **mlg**, orm, **som**, **swa**, **xho**
AfroXLMR* (117M-270M)	20	afr, **amh**, **hau**, **ibo**, **kin**, run **mlg**, nya, orm, **pcm**, **sna**, **som**, sot, **swa**, **xho**, **yor**, **zul**

Table 2: Language coverage and size for pre-trained language models. Languages in **bold** have evaluation datasets for either NER, news topic classification or sentiment analysis.

Hyper-parameters for baseline models We fine-tune the baseline models for NER, news topic classification and sentiment classification for 50, 25, and 20 epochs respectively. We use a learning rate of 5e-5 for all the task, except for sentiment classification where we use 2e-5 for XLM-R-base and XLM-R-large. The maximum sequence length is 164 for NER, 500 for news topic classification, and 128 for sentiment classification. The adapted models also make use of similar hyper-parameters.

5 Multilingual Adaptive Fine-tuning

We introduce MAFT as an approach to adapt a multilingual PLM to a new set of languages. Adapting PLMs has been shown to be effective when adapting to a new domain (Gururangan et al., 2020) or language (Pfeiffer et al., 2020; Alabi et al., 2020; Muller et al., 2021; Adelani et al., 2021). While previous work on multilingual adaptation has mostly focused on autoregressive sequence-to-sequence models such as mBART (Tang et al., 2020), in this work, we adapt non-autoregressive masked PLMs on monolingual corpora covering 20 languages. Crucially, during adaptation we use the same objective that was also used during pre-training. The models resulting from MAFT can then be fine-tuned on supervised NLP downstream tasks. We name the model resulting after applying MAFT to XLM-R-base and XLM-R-miniLM as *AfroXLMR-base* and *AfroXLMR-mini*, respectively. For adaptation, we train

[6] https://www.bbc.com/pidgin
[7] https://mg.globalvoices.org/
[8] https://www.isolezwe.co.za

Model	Size	amh	ara	eng	hau	ibo	kin	lug	luo	pcm	swa	wol	xho	yor	avg
Finetune															
XLM-R-miniLM	117M	69.5	76.1	91.5	74.5	81.9	68.6	64.7	11.7	83.2	86.3	51.7	69.3	72.0	69.3
AfriBERTa	126M	73.8	51.3	89.0	90.2	87.4	73.8	78.9	70.2	85.7	88.0	61.8	67.2	81.3	76.8
XLM-R-base	270M	70.6	77.9	92.3	89.5	84.8	73.3	79.7	74.9	87.3	87.4	63.9	69.9	78.3	79.2
XLM-R-large	550M	76.2	79.7	93.1	90.5	84.1	73.8	81.6	73.6	89.0	89.4	67.9	72.4	78.9	80.8
MAFT + Finetune															
XLM-R-miniLM	117M	69.7	76.5	91.7	87.7	83.5	74.1	77.4	17.5	85.5	86.0	59.0	72.3	75.1	73.5
AfriBERTa	126M	72.5	40.9	90.1	89.7	87.6	75.2	80.1	69.6	86.5	87.6	62.3	71.8	77.0	76.2
XLM-R-base	270M	76.1	79.7	**92.8**	91.2	87.4	78.0	82.9	75.1	89.6	88.6	67.4	71.9	82.1	81.8
XLM-R-base-v70k	140M	70.1	76.4	91.0	91.4	86.6	77.5	83.2	**75.4**	89.0	88.7	65.9	72.4	81.3	80.7
XLM-R-base+LAFT	270M x 13	**78.0**	79.1	91.3	**91.5**	**87.7**	77.8	**84.7**	75.3	**90.0**	**89.5**	**68.3**	**73.2**	**83.7**	**82.3**

Table 3: **NER** model comparison, showing F1-score on the test sets after 50 epochs averaged over 5 runs. Results are for all 4 tags in the dataset: PER, ORG, LOC, DATE/MISC. For LAFT, we multiplied the size of XLM-R-base by the number of languages as LAFT results in a single model per language.

on a combination of the monolingual corpora used for AfriMT5 adaptation by Adelani et al. (2022). Details for each of the monolingual corpora and languages are provided in Appendix A.1.

Hyper-parameters for MAFT The PLMs were trained for 3 epochs with a learning rate of 5e-5 using huggingface transformers (Wolf et al., 2020). We use of a batch size of 32 for AfriBERTa and a batch size 10 for the other PLMs.

5.1 Vocabulary reduction

Multilingual PLMs come with various parameter sizes, the larger ones having more than hundred million parameters, which makes fine-tuning and deploying such models a challenge due to resource constraints. One of the major factors that contributes to the parameter size of these models is the embedding matrix whose size is a function of the vocabulary size of the model. While a large vocabulary size is essential for a multilingual PLM trained on hundreds of languages, some of the tokens in the vocabulary can be removed when they are irrelevant to the domain or language considered in the downstream task, thus reducing the vocabulary size of the model. Inspired by Abdaoui et al. (2020), we experiment with reducing the vocabulary size of the XLM-R-base model before adapting via MAFT. There are two possible vocabulary reductions in our setting: *(1) removal of tokens before MAFT or (2) removal of tokens after MAFT*. From our preliminary experiments, we find approach (1) to work better. We call the resulting model, *AfroXLMR-small*.

To remove non-African vocabulary sub-tokens from the pretrained XLM-base model, we concatenated the monolingual texts from 19 out of the 20 African languages together. Then, we apply `sentencepiece` to the Amharic monolingual texts, and concatenated texts separately using the original XLM-R-base tokenizer. The frequency of all the sub-tokens in the two separate monolingual corpora is computed, and we select the top-k most frequent tokens from the separate corpora. We used this separate sampling to ensure that a considerable number of Amharic sub-tokens are captured in the new vocabulary, we justify the choice of this approach in Section 5.3. We assume that the top-k most frequent tokens should be representative of the vocabulary of the whole 20 languages. We chose $k = 52.000$ from the Amharic sub-tokens which covers 99.8% of the Amharic monolingual texts, and $k = 60.000$ which covers 99.6% of the other 19 languages, and merged them. In addition, we include the top 1000 tokens from the original XLM-R-base tokenizer in the new vocabulary to include frequent tokens that were not present in the new top-k tokens.[9] We note that our assumption above may not hold in the case of some very distant and low-resourced languages as well as when there are domain differences between the corpora used during adaptation and fine-tuning. We leave the investigation of alternative approaches for vocabulary compression for future work.

5.2 Results and discussion

5.2.1 Baseline results

For the baseline models (top rows in Tables 3, 4, and 5), we directly fine-tune on each of the downstream tasks in the target language: NER, news topic classification and sentiment analysis.

Performance on languages seen during pre-training For NER and sentiment analysis we find XLM-R-large to give the best overall performance. We attribute this to the fact that it has a larger model capacity compared to the other PLMs. Similarly, we find AfriBERTa and XLM-R-base to give better results on languages they have been pre-trained on (see Table 2), and in most cases AfriBERTa tends to perform better than XLM-R-base on languages they are both pre-trained on, for example `amh`, `hau`, and `swa`. However, when the languages are unseen by AfriBERTa (e.g. `ara`, `eng`, `wol`, `lin`, `lug`, `luo`, `xho`, `zul`), it performs much worse than XLM-R-base and in some cases even worse than the XLM-R-miniLM. This shows that it may be better

[9]This introduced just a few new tokens which are mostly English tokens to the new vocabulary. We end up with 70.609 distinct sub-tokens after combining all of them.

Model	Size	amh	eng	hau	kin	lin	mlg	pcm	som	swa	yor	zul	avg
Finetune													
XLM-R-miniLM	117M	70.4	94.1	77.6	64.2	41.2	42.9	67.6	74.2	86.7	68.8	56.9	67.7
AfriBERTa	126M	70.7	93.6	90.1	75.8	55.4	56.4	81.5	**79.9**	87.7	**82.6**	71.4	76.8
XLM-R-base	270M	71.1	94.1	85.9	73.3	56.8	54.2	77.3	78.8	87.1	71.1	70.0	74.6
XLM-R-large	550M	72.7	94.5	86.2	75.1	52.2	63.6	79.4	79.2	87.5	74.8	78.7	76.7
MAFT + Finetune													
XLM-R-miniLM	117M	69.5	94.1	86.7	72.0	51.7	55.3	78.1	77.7	87.2	74.0	60.3	73.3
AfriBERTa	126M	68.8	93.7	89.5	76.5	54.9	59.7	**82.2**	**79.9**	87.7	80.8	76.4	77.3
XLM-R-base	270M	71.9	**94.6**	88.3	**76.8**	**58.6**	64.7	78.9	79.1	87.8	80.2	**79.6**	78.2
XLM-R-base-v70k	140M	70.4	94.2	87.7	76.1	56.8	64.4	76.1	79.4	87.4	76.9	77.4	76.9
XLM-R-base+LAFT	270M x 11	**73.0**	94.3	**91.2**	76.0	56.9	**67.3**	77.4	79.4	**88.0**	79.2	79.5	**78.4**

Table 4: **News topic classification** model comparison, showing F1-score on the test sets after 25 epochs averaged over 5 runs. For LAFT, we multiplied the size of XLM-R-base by the number of languages.

Model	Size	amh	eng	hau	ibo	pcm	yor	avg
Finetune								
XLM-R-miniLM	117M	51.0	62.8	75.0	78.0	72.9	73.4	68.9
AfriBERTa-large	126M	51.7	61.8	81.0	**81.2**	75.0	80.2	71.8
XLM-R-base	270M	51.4	66.2	78.4	79.9	76.3	76.9	71.5
XLM-R-large	550M	52.4	67.5	79.3	80.8	77.6	78.1	72.6
MAFT+Finetune								
XLM-R-miniLM	117M	51.3	63.3	77.7	78.0	73.6	74.3	69.7
AfriBERTa	126M	53.6	63.2	81.0	80.6	74.7	80.4	72.3
XLM-R-base	270M	53.0	65.6	80.7	80.5	77.5	79.4	72.8
XLM-R-base-v70k	140M	52.2	65.3	80.6	81.0	77.4	78.6	72.5
XLM-R-base+LAFT	270M x 6	**55.0**	65.6	**81.5**	80.8	74.7	**80.9**	**73.1**

Table 5: **Sentiment classification** model comparison, showing F1 evaluation on test sets after 20 epochs, averaged over 5 runs. We obtained the results for the baseline model results of "hau", "ibo", "pcm", and "yor" from Muhammad et al. (2022). For LAFT, we multiplied the size of XLM-R-base by the number of languages as LAFT results in a single model per language.

to adapt to a new African language from a PLM that has seen numerous languages than one trained on a subset of African languages from scratch.

LAFT is a strong baseline The results of applying LAFT to the XLM-R-base model are shown in the last row of Tables 3, 4, and 5. We find that applying LAFT on each language individually provides a significant improvement in performance across all languages and tasks we evaluated on. Sometimes, the improvement is very large, for example, +7.4 F1 on Amharic NER and +9.5 F1 for Zulu news-topic classification. The only exception is for English since XLM-R has already seen large amounts of English text during pre-training. Additionally, LAFT models tend to give slightly worse result when adaptation is performed on a smaller corpus.[10]

5.2.2 Multilingual adaptive fine-tuning results

While LAFT provides an upper bound on downstream performance for most languages, our new approach is often competitive to LAFT. On average, the difference on NER, news topic and sentiment classification is −0.5, −0.2, and −0.3 F1, respectively. Crucially, compared to LAFT, MAFT results in a single adapted model which can be applied to many languages while LAFT results in a new model for each language. Below, we discuss our results in more detail.

PLMs pre-trained on many languages benefit the most from MAFT We found all the PLMs to improve after we applied MAFT. The improvement is the largest for the XLM-R-miniLM, where the performance improved by +4.2 F1 for NER, and +5.6 F1 for news topic classification. Although, the improvement was lower for sentiment classification (+0.8). Applying MAFT on XLM-R-base gave the overall best result. On average, there is an improvement of +2.6, +3.6, and +1.5 F1 on NER, news topic and sentiment classification, respectively. The main advantage of MAFT is that it allows us to use the same model for many African languagesinstead of many models specialized to individual languages. This significantly reduces the required disk space to store the models, without sacrificing performance. Interestingly, there is no strong benefit of applying MAFT to AfriBERTa. In most cases the improvement is < 0.6 F1. We speculate that this is probably due to AfriBERTa's tokenizer having a limited coverage. We leave a more detailed investigation of this for future work.

[10]We performed LAFT on eng using VOA news corpus with about 906.6MB, much smaller than the CC-100 eng corpus (300GB)

Model	amh		ara		eng		yor	
	#UNK	F1	#UNK	F1	#UNK	F1	#UNK	F1
AfroXLMR-base	0	76.1	0	79.7	0	92.8	24	82.1
Afro-XLM-R70k (i)	3704	67.8	1403	76.3	44	90.6	5547	81.2
Afro-XLM-R70k (ii)	3395	70.1	669	76.4	54	91.0	6438	81.3

Table 6: Numbers of UNKs when the model tokenizers are applied on the NER test sets.

More efficient models using vocabulary reduction Applying vocabulary reduction helps to reduce the model size by more than 50% before applying MAFT. We find a slight reduction in performance as we remove more vocabulary tokens. Average performance of XLM-R-base-v70k reduces by -1.6, -1.5 and -0.6 F1 for NER, news topic, and sentiment classification compared to the XLM-R-base+LAFT baseline. Despite the reduction in performance compared to XLM-R-base+LAFT, they are still better than XLM-R-miniLM, which has a similar model size, with or without MAFT. We also find that their performance is better than that of the PLMs that have not undergone any adaptation. We find the largest reduction in performance on languages that make use of non-Latin scripts i.e. `amh` and `ara` — they make use of the Ge'ez script and Arabic script respectively. We attribute this to the vocabulary reduction impacting the number of `amh` and `ara` subwords covered by our tokenizer.

In summary, we recommend XLM-R-base+MAFT (i.e. AfroXLMR-base) for all languages on which we evaluated, including high-resource languages like English, French and Arabic. If there are GPU resource constraints, we recommend using XLM-R-base-v70k+MAFT (i.e. AfroXLMR-small).

5.3 Ablation experiments on vocabulary reduction

Our results showed that applying vocabulary reduction reduced the model size, but we also observed a drop in performance for different languages across the downstream tasks, especially for Amharic, because it uses a non-Latin script. Hence, we compared different sampling strategies for selecting the top-k vocabulary subtokens. These include: (i) concatenating the monolingual texts, and selecting the top-70k sub-tokens (ii) the exact approach described in Section 5.1. The resulting tokenizers from the two approaches are used to tokenize the sentences in the NER test sets for Amharic, Arabic, English, and Yorùbá. Table 6 shows the number of UNKs in the respective test set after tokenization and the F1 scores obtained on the NER task for the languages. The table shows that the original AfroXLMR tokenizer obtained the least number of UNKs for all languages, with the highest F1 scores. Note that Yorùbá has 24 UNKs, which is explained by the fact that Yorùbá was not seen during pre-training. Furthermore, using approach (i), gave 3704 UNKs for Amharic, but with approach (ii) there was a significant drop in the number of UNKs and an improvement in F1 score. We noticed a drop in the vocabulary coverage for the other languages as we increased the Amharic sub-tokens. Therefore, we concluded that there is no sweet spot in terms of the way to pick the vocabulary that covers all languages and we believe that this is an exciting area for future work.

5.4 Scaling MAFT to larger models

To demonstrate the applicability of MAFT to larger models, we applied MAFT to XLM-R-large using the same training setup as XLM-R-base. We refer to the new PLM as AfroXLMR-large. For comparison, we also trained individual LAFT models using the monolingual data[11] from Adelani et al. (2021). Table 7 shows the evaluation result on NER. Averaging over all 13 languages, AfroXLMR-large improved over XLM-R-large by $+2.8$ F1, which is very comparable to the improvement we obtained between AfroXLMR-base (81.8 F1) and XLM-R-base (79.2 F1). Surprisingly, the improvement is quite large ($+3.5$ to $+6.3$ F1) for seven out of ten African languages: `yor`, `luo`, `lug`, `kin`, `ibo`, and `amh`. The most interesting observation is that AfroXLMR-large, on average, is either competitive or better than the individual language LAFT models, including languages not seen during the MAFT training stage like `lug`, `luo` and `wol`. This implies that AfroXLMR-large (a single model) provides a better alternative to XLM-R-large+LAFT (for each language) in terms of performance on downstream tasks and disk space. AfroXLMR-large is currently the largest masked language model for African languages, and achieves the state-of-the-art compared to all other multilingual PLM on the NER task. This shows that our MAFT approach is very effective and scales to larger PLMs.

6 Cross-lingual Transfer Learning

The previous section demonstrates the applicability of MAFT in the fully-supervised transfer learning setting. Here, we demonstrate that our MAFT approach is also very effective in the zero-shot cross-lingual transfer setting using parameter-efficient fine-tuning methods.

Parameter-efficient fine-tuning methods like *adapters* (Houlsby et al., 2019) are appealing because of their modularity, portability, and composability across languages and tasks. Often times, language adapters are trained on a general domain corpus

[11]For languages not in MasakhaNER, we use the same monolingual data in Table 9.

Model	Size	amh	ara	eng	hau	ibo	kin	lug	luo	pcm	swa	wol	xho	yor	avg
XLM-R-large	550M	76.2	79.7	**93.1**	90.5	84.1	73.8	81.6	73.6	89.0	89.4	67.9	72.4	78.9	80.8
XLM-R-large+LAFT	550M x 13	**79.9**	**81.3**	92.2	**91.7**	**87.7**	78.4	86.2	**78.2**	**91.1**	90.3	68.8	72.7	82.9	83.2
AfroXLMR-large	550M	79.7	80.9	92.2	91.2	**87.7**	**79.1**	**86.7**	78.1	91.0	**90.4**	**69.6**	**72.9**	**85.2**	**83.4**

Table 7: **NER** model comparison on XLM-R-large, XLM-R-large+LAFT and XLM-R-large+MAFT (i.e AfroXLMR-large), showing F1-score on the test sets after 50 epochs averaged over 5 runs. Results are for all 4 tags in the dataset: PER, ORG, LOC, DATE/MISC.

Model	amh	hau	ibo	kin	lug	luo	pcm	swa	wol	yor	avg
XLM-R-base (fully-supervised)	69.7	91.0	86.2	**73.8**	80.5	75.8	**86.9**	**88.7**	69.6	78.1	81.2
mBERT (MAD-X) (Ansell et al., 2021)	-	83.4	71.7	65.3	67.0	52.2	72.1	77.6	65.6	74.0	69.9
mBERT (MAD-X on **news domain**)	-	86.0	77.6	69.9	73.3	56.9	78.5	80.2	68.8	75.6	74.1
XLM-R-base (MAD-X on **news domain**)	47.5	85.5	83.2	72.0	75.7	57.8	76.8	84.0	68.2	72.2	75.0
AfroXLMR-base (MAD-X on **news domain**)	47.7	88.1	80.9	73.0	80.1	59.2	79.9	86.9	69.1	75.6	77.0
mBERT (LT-SFT) (Ansell et al., 2021)	-	83.5	76.7	67.4	67.9	54.7	74.6	79.4	66.3	74.8	71.7
mBERT (LT-SFT on **news domain**)	-	86.4	80.6	69.2	76.8	55.1	80.4	82.3	71.6	76.7	75.4
XLM-R-base (LT-SFT on **news domain**)	54.1	87.6	81.4	72.7	79.5	60.7	81.2	85.5	73.6	73.7	77.3
AfroXLMR-base (LT-SFT on **news domain**)	54.0	88.6	83.5	**73.8**	**81.0**	60.7	81.7	86.4	**74.5**	**78.7**	78.8

Table 8: **Cross-lingual transfer using LT-SFT** (Ansell et al., 2021) and evaluation on **MasakhaNER**. The full-supervised baselines are obtained from Adelani et al. (2021) to measure performance gap when annotated datasets are available. Experiments are performed on 3 tags: PER, ORG, LOC. Average (avg) excludes amh. The best zero-shot transfer F1-scores are underlined.

like Wikipedia. However, when there is a mismatch between the target *domain* of the task and the *domain* of the language adapter, it could also impact the cross-lingual performance.

Here, we investigate how we can improve the cross-lingual transfer abilities of our adapted PLM – AfroXLMR-base by training language adapters on the same domain as the target task. For our experiments, we use the MasakhaNER dataset, which is based on the news domain. We compare the performance of language adapters trained on *Wikipedia* and *news* domains. In addition to adapters, we experiment with another parameter-efficient method based on Lottery-Ticket Hypothesis (Frankle & Carbin, 2019) i.e. LT-SFT (Ansell et al., 2021).

For the adapter approach, we make use of the MAD-X approach (Pfeiffer et al., 2020) – an adapter-based framework that enables cross-lingual transfer to arbitrary languages by learning modular language and task representations. However, the evaluation data in the target languages should have the same task and label configuration as the source language. Specifically, we make use of MAD-X 2.0 (Pfeiffer et al., 2021) where the last adapter layers are dropped, which has been shown to improve performance. The setup is as follows: (1) We train language adapters via masked language modelling (MLM) individually on source and target languages, the corpora used are described in Appendix A.2; (2) We train a task adapter by fine-tuning on the target task using labelled data in a source language. (3) During inference, task and language adapters are stacked together by substituting the source language adapter with a target language adapter.

We also make use of the Lottery Ticket Sparse Fine-tuning (LT-SFT) approach (Ansell et al., 2021), a parameter-efficient fine-tuning approach that has been shown to give competitive or better performance than the MAD-X 2.0 approach. The LT-SFT approach is based on the Lottery Ticket Hypothesis (LTH) that states that each neural model contains a sub-network (a "winning ticket") that, if trained again in isolation, can reach or even surpass the performance of the original model. The LTH is originally a compression approach, the authors of LT-SFT re-purposed the approach for cross-lingual adaptation by finding sparse sub-networks for tasks and languages, that will later be composed together for zero-shot adaptation, similar to Adapters. For additonal details we refer to Ansell et al. (2021).

6.1 Experimental setup

For our experiments, we followed the same setting as Ansell et al. (2021) that adapted mBERT from English CoNLL03 (Tjong Kim Sang & De Meulder, 2003) to African languages (using MasakhaNER dataset) for the NER task.[12] Furthermore, we extend the experiments to XLMR-base and AfroXLMR-base. For the training of MAD-X 2.0 and sparse fine-tunings (SFT) for African languages, we make use of the monolingual texts from the news domain since it matches the domain of the evaluation data. Unlike, Ansell et al. (2021) that trained adapters and SFT on monolingual data from Wikipedia domain except for luo and pcm where the dataset is absent, we show that the domain used for training language SFT is also very important. For a

[12]We excluded the MISC and DATE from CoNLL03 and MasakhaNER respectively to ensure same label configuration.

fair comparison, we reproduced the result of Ansell et al. (2021) by training MAD-X 2.0 and LT-SFT on mBERT, XLM-R-base and AfroXLMR-base on target languages with the news domain corpus. But, we still make use of the pre-trained English language adapter[13] and SFT[14] for mBERT and XLM-R-base trained on the Wikipedia domain. For the AfroXLMR-base, we make use of the same English adapter and SFT as XLM-R-base because the PLM is already good for English language. We make use of the same hyper-parameters reported in the LT-SFT paper.

Hyper-parameters for adapters We train the task adapter using the following hyper-parameters: batch size of 8, 10 epochs, "pfeiffer" adapter config, adapter reduction factor of 8, and learning rate of 5e-5. For the language adapters, we make use of 100 epochs or maximum steps of 100K, minimum number of steps is 30K, batch size of 8, "pfeiffer+inv" adapter config, adapter reduction factor of 2, learning rate of 5e-5, and maximum sequence length of 256. For a fair comparison with adapter models trained on Wikipedia domain, we used the same hyper-parameter settings (Ansell et al., 2021) for the news domain.

6.2 Results and discussion

Table 8 shows the results of MAD-X 2.0 and LT-SFT, we compare their performance to fully supervised setting, where we fine-tune XLM-R-base on the training dataset of each of the languages, and evaluate on the test-set. We find that both MAD-X 2.0 and LT-SFT using news domain for African languages produce better performance (+4.2 on MAD-X and +3.7 on LT-SFT) than the ones trained largely on the wikipedia domain. This shows that the domain of the data matters. Also, we find that training LT-SFT on XLM-R-base gives better performance than mBERT on all languages. For MAD-X, there are a few exceptions like `hau`, `pcm`, and `yor`. Overall, the best performance is obtained by training LT-SFT on AfroXLMR-base, and sometimes it give better performance than the fully-supervised setting (e.g. as observed in `kin` and `lug`, `wol` `yor` languages). On both MAD-X and LT-SFT, AfroXLMR-base gives the best result since it has been firstly adapted on several African languages and secondly on the target domain of the target task. This shows that the MAFT approach is effective since the technique provides a better PLM that parameter-efficient methods can benefit from.

7 Conclusion

In this work, we proposed and studied MAFT as an approach to adapt multilingual PLMs to many African languages with a single model. We evaluated our approach on 3 different NLP downstream tasks and additionally contribute novel news topic classification dataset for 4 African languages. Our results show that MAFT is competitive to LAFT while providing a single model compared to many models specialized for individual languages. We went further to show that combining vocabulary reduction and MAFT leads to a 50% reduction in the parameter size of a XLM-R while still being competitive to applying LAFT on individual languages. We hope that future work improves vocabulary reduction to provide even smaller models with strong performance on distant and low-resource languages. To further research on NLP for African languages and reproducibility, we have uploaded our language adapters, language SFTs, AfroXLMR-base, AfroXLMR-small, and AfroXLMR-mini models to the HuggingFace Model Hub[15].

Acknowledgments

Jesujoba Alabi was partially funded by the BMBF project SLIK under the Federal Ministry of Education and Research grant 01IS22015C. David Adelani acknowledges the EU-funded Horizon 2020 projects: ROXANNE under grant number 833635 and COMPRISE (`http://www.compriseh2020.eu/`) under grant agreement No. 3081705. Marius Mosbach acknowledges funding from the Deutsche Forschungsgemeinschaft (DFG, German Research Foundation) – Project-ID 232722074 – SFB 1102. We also thank DFKI GmbH for providing the infrastructure to run some of the experiments. We are also grateful to Stella Biderman for providing the resources to train AfroXLMR-large. We thank Alan Ansell for providing his MAD-X 2.0 code. Lastly, we thank Benjamin Muller, the anonymous reviewers of AfricaNLP 2022 workshop and COLING 2022 for their helpful feedback.

References

Amine Abdaoui, Camille Pradel, and Grégoire Sigel. Load what you need: Smaller versions of mutililingual BERT. In *Proceedings of SustaiNLP: Workshop on Simple and Efficient Natural Language Processing*, pp. 119–123, Online, November 2020. Association for Computational Linguistics. doi: 10.18653/v1/2020.sustainlp-1.16. URL https://aclanthology.org/2020.sustainlp-1.16.

David Adelani, Jesujoba Alabi, Angela Fan, Julia Kreutzer, Xiaoyu Shen, Machel Reid, Dana Ruiter, Dietrich Klakow, Peter Nabende, Ernie Chang, Tajuddeen Gwadabe, Freshia Sackey, Bonaventure F. P. Dossou, Chris Emezue, Colin Leong, Michael Beukman, Shamsuddeen Muhammad, Guyo Jarso, Oreen Yousuf, Andre Niyongabo Rubungo, Gilles Hacheme, Eric Peter Wairagala, Muhammad Umair Nasir, Benjamin Ajibade, Tunde Ajayi, Yvonne

[13] https://adapterhub.ml/

[14] https://huggingface.co/cambridgeltl

[15] https://huggingface.co/models?sort=downloads&search=Davlan%2Fafro-xlmr

Gitau, Jade Abbott, Mohamed Ahmed, Millicent Ochieng, Anuoluwapo Aremu, Perez Ogayo, Jonathan Mukiibi, Fatoumata Ouoba Kabore, Godson Kalipe, Derguene Mbaye, Allahsera Auguste Tapo, Victoire Memdjokam Koagne, Edwin Munkoh-Buabeng, Valencia Wagner, Idris Abdulmumin, Ayodele Awokoya, Happy Buzaaba, Blessing Sibanda, Andiswa Bukula, and Sam Manthalu. A few thousand translations go a long way! leveraging pre-trained models for African news translation. In *Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pp. 3053–3070, Seattle, United States, July 2022. Association for Computational Linguistics. doi: 10.18653/v1/2022.naacl-main.223. URL https://aclanthology.org/2022.naacl-main.223.

David Ifeoluwa Adelani, Jade Abbott, Graham Neubig, Daniel D'souza, Julia Kreutzer, Constantine Lignos, Chester Palen-Michel, Happy Buzaaba, Shruti Rijhwani, Sebastian Ruder, Stephen Mayhew, Israel Abebe Azime, Shamsuddeen H. Muhammad, Chris Chinenye Emezue, Joyce Nakatumba-Nabende, Perez Ogayo, Aremu Anuoluwapo, Catherine Gitau, Derguene Mbaye, Jesujoba Alabi, Seid Muhie Yimam, Tajuddeen Rabiu Gwadabe, Ignatius Ezeani, Rubungo Andre Niyongabo, Jonathan Mukiibi, Verrah Otiende, Iroro Orife, Davis David, Samba Ngom, Tosin Adewumi, Paul Rayson, Mofetoluwa Adeyemi, Gerald Muriuki, Emmanuel Anebi, Chiamaka Chukwuneke, Nkiruka Odu, Eric Peter Wairagala, Samuel Oyerinde, Clemencia Siro, Tobius Saul Bateesa, Temilola Oloyede, Yvonne Wambui, Victor Akinode, Deborah Nabagereka, Maurice Katusiime, Ayodele Awokoya, Mouhamadane MBOUP, Dibora Gebreyohannes, Henok Tilaye, Kelechi Nwaike, Degaga Wolde, Abdoulaye Faye, Blessing Sibanda, Orevaoghene Ahia, Bonaventure F. P. Dossou, Kelechi Ogueji, Thierno Ibrahima DIOP, Abdoulaye Diallo, Adewale Akinfaderin, Tendai Marengereke, and Salomey Osei. MasakhaNER: Named entity recognition for African languages. *Transactions of the Association for Computational Linguistics*, 9:1116–1131, 2021. doi: 10.1162/tacl_a_00416. URL https://aclanthology.org/2021.tacl-1.66.

Jesujoba Alabi, Kwabena Amponsah-Kaakyire, David Adelani, and Cristina España-Bonet. Massive vs. curated embeddings for low-resourced languages: the case of Yorùbá and Twi. In *Proceedings of the 12th Language Resources and Evaluation Conference*, pp. 2754–2762, Marseille, France, May 2020. European Language Resources Association. ISBN 979-10-95546-34-4. URL https://aclanthology.org/2020.lrec-1.335.

Alan Ansell, Edoardo Maria Ponti, Anna Korhonen, and Ivan Vulić. Composable sparse fine-tuning for cross-lingual transfer, 2021.

Israel Abebe Azime and Nebil Mohammed. An amharic news text classification dataset. *ArXiv*, abs/2103.05639, 2021.

Yassine Benajiba, Paolo Rosso, and José Miguel BenedíRuiz. Anersys: An arabic named entity recognition system based on maximum entropy. In Alexander Gelbukh (ed.), *Computational Linguistics and Intelligent Text Processing*, pp. 143–153, Berlin, Heidelberg, 2007. Springer Berlin Heidelberg. ISBN 978-3-540-70939-8.

Michael Beukman. Analysing the effects of transfer learning on low-resourced named entity recognition performance. In *3rd Workshop on African Natural Language Processing*, 2021. URL https://openreview.net/forum?id=HKWMFqfN8b5.

Ethan C. Chau and Noah A. Smith. Specializing multilingual language models: An empirical study. In *Proceedings of the 1st Workshop on Multilingual Representation Learning*, pp. 51–61, Punta Cana, Dominican Republic, November 2021. Association for Computational Linguistics. doi: 10.18653/v1/2021.mrl-1.5. URL https://aclanthology.org/2021.mrl-1.5.

Zewen Chi, Li Dong, Furu Wei, Nan Yang, Saksham Singhal, Wenhui Wang, Xia Song, Xian-Ling Mao, Heyan Huang, and Ming Zhou. InfoXLM: An information-theoretic framework for cross-lingual language model pre-training. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pp. 3576–3588, Online, June 2021. Association for Computational Linguistics. doi: 10.18653/v1/2021.naacl-main.280. URL https://aclanthology.org/2021.naacl-main.280.

Hyung Won Chung, Thibault Fevry, Henry Tsai, Melvin Johnson, and Sebastian Ruder. Rethinking embedding coupling in pre-trained language models. In *International Conference on Learning Representations*, 2021. URL https://openreview.net/forum?id=xpFFI_NtgpW.

Alexis Conneau, Kartikay Khandelwal, Naman Goyal, Vishrav Chaudhary, Guillaume Wenzek, Francisco Guzmán, Edouard Grave, Myle Ott, Luke Zettlemoyer, and Veselin Stoyanov. Unsupervised cross-lingual representation learning at scale. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pp. 8440–8451, Online, July 2020. Association for Computational Linguistics. doi: 10.18653/v1/2020.acl-main.747. URL https://aclanthology.org/2020.acl-main.747.

Davis David. Swahili : News classification dataset. *Zenodo*, December 2020. doi: 10.5281/zenodo.5514203. URL https://doi.org/10.5281/zenodo.5514203. The news version contains both train and test sets.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pp. 4171–4186, Minneapolis, Minnesota, June 2019. Association for Computational Linguistics. doi: 10.18653/v1/N19-1423. URL https://aclanthology.org/N19-1423.

Roald Eiselen. Government domain named entity recognition for South African languages. In *Proceedings of the Tenth International Conference on Language Resources and Evaluation (LREC'16)*, pp. 3344–3348, Portorož, Slovenia, May 2016. European Language Resources Association (ELRA). URL https://aclanthology.org/L16-1533.

∀, Wilhelmina Nekoto, Vukosi Marivate, Tshinondiwa Matsila, Timi Fasubaa, Taiwo Fagbohungbe, Solomon Oluwole Akinola, Shamsuddeen Muhammad, Salomon Kabongo Kabenamualu, Salomey Osei, Freshia Sackey, Rubungo Andre Niyongabo, Ricky Macharm, Perez Ogayo, Orevaoghene Ahia, Musie Meressa Berhe, Mofetoluwa Adeyemi, Masabata Mokgesi-Selinga, Lawrence Okegbemi, Laura Martinus, Kolawole Tajudeen, Kevin Degila, Kelechi Ogueji, Kathleen Siminyu, Julia Kreutzer, Jason Webster, Jamiil Toure Ali, Jade Abbott, Iroro Orife, Ignatius Ezeani, Idris Abdulkadir Dangana, Herman Kamper, Hady Elsahar, Goodness Duru, Ghollah Kioko, Murhabazi Espoir, Elan van Biljon, Daniel Whitenack, Christopher Onyefuluchi, Chris Chinenye Emezue, Bonaventure F. P. Dossou, Blessing Sibanda, Blessing Bassey, Ayodele Olabiyi, Arshath Ramkilowan, Alp Öktem, Adewale Akinfaderin, and Abdallah Bashir. Participatory research for low-resourced machine translation: A case study in African languages. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, Online, 2020. URL https://www.aclweb.org/anthology/2020.findings-emnlp.195.

Jonathan Frankle and Michael Carbin. The lottery ticket hypothesis: Finding sparse, trainable neural networks. In *International Conference on Learning Representations*, 2019. URL https://openreview.net/forum?id=rJl-b3RcF7.

Suchin Gururangan, Ana Marasović, Swabha Swayamdipta, Kyle Lo, Iz Beltagy, Doug Downey, and Noah A. Smith. Don't stop pretraining: Adapt language models to domains and tasks. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pp. 8342–8360, Online, July 2020. Association for Computational Linguistics. doi: 10.18653/v1/2020.acl-main.740. URL https://aclanthology.org/2020.acl-main.740.

Pengcheng He, Jianfeng Gao, and Weizhu Chen. Debertav3: Improving deberta using electra-style pretraining with gradient-disentangled embedding sharing. *ArXiv*, abs/2111.09543, 2021.

Michael A. Hedderich, David Adelani, Dawei Zhu, Jesujoba Alabi, Udia Markus, and Dietrich Klakow. Transfer learning and distant supervision for multilingual transformer models: A study on African languages. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pp. 2580–2591, Online, November 2020. Association for Computational Linguistics. doi: 10.18653/v1/2020.emnlp-main.204. URL https://aclanthology.org/2020.emnlp-main.204.

Neil Houlsby, Andrei Giurgiu, Stanislaw Jastrzebski, Bruna Morrone, Quentin De Laroussilhe, Andrea Gesmundo, Mona Attariyan, and Sylvain Gelly. Parameter-efficient transfer learning for NLP. In Kamalika Chaudhuri and Ruslan Salakhutdinov (eds.), *Proceedings of the 36th International Conference on Machine Learning*, volume 97 of *Proceedings of Machine Learning Research*, pp. 2790–2799. PMLR, 09–15 Jun 2019. URL https://proceedings.mlr.press/v97/houlsby19a.html.

Xiaoqi Jiao, Yichun Yin, Lifeng Shang, Xin Jiang, Xiao Chen, Linlin Li, Fang Wang, and Qun Liu. TinyBERT: Distilling BERT for natural language understanding. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pp. 4163–4174, Online, November 2020. Association for Computational Linguistics. doi: 10.18653/v1/2020.findings-emnlp.372. URL https://aclanthology.org/2020.findings-emnlp.372.

Weijie Liu, Peng Zhou, Zhiruo Wang, Zhe Zhao, Haotang Deng, and Qi Ju. FastBERT: a self-distilling BERT with adaptive inference time. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pp. 6035–6044, Online, July 2020a. Association for Computational Linguistics. doi: 10.18653/v1/2020.acl-main.537. URL https://aclanthology.org/2020.acl-main.537.

Yinhan Liu, Jiatao Gu, Naman Goyal, Xian Li, Sergey Edunov, Marjan Ghazvininejad, Mike Lewis, and Luke Zettlemoyer. Multilingual denoising pre-training for neural machine translation. *Transactions of the Association for Computational Linguistics*, 8:726–742, 2020b. doi: 10.1162/tacl_a_00343. URL https://aclanthology.org/2020.tacl-1.47.

Fuli Luo, Wei Wang, Jiahao Liu, Yijia Liu, Bin Bi, Songfang Huang, Fei Huang, and Luo Si. VECO: Variable and flexible cross-lingual pre-training for language understanding and generation. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language*

Processing (Volume 1: Long Papers), pp. 3980–3994, Online, August 2021. Association for Computational Linguistics. doi: 10.18653/v1/2021.acl-long.308. URL https://aclanthology.org/2021.acl-long.308.

Shamsuddeen Hassan Muhammad, David Ifeoluwa Adelani, Sebastian Ruder, Ibrahim Said Ahmad, Idris Abdulmumin, Bello Shehu Bello, Monojit Choudhury, Chris Chinenye Emezue, Saheed Abdullahi Salahudeen, Aremu Anuoluwapo, Alípio Jeorge, and Pavel Brazdil. Naijasenti: A nigerian twitter sentiment corpus for multilingual sentiment analysis. *ArXiv*, abs/2201.08277, 2022.

Benjamin Muller, Antonios Anastasopoulos, Benoît Sagot, and Djamé Seddah. When being unseen from mBERT is just the beginning: Handling new languages with multilingual language models. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pp. 448–462, Online, June 2021. Association for Computational Linguistics. doi: 10.18653/v1/2021.naacl-main.38. URL https://aclanthology.org/2021.naacl-main.38.

Rubungo Andre Niyongabo, Qu Hong, Julia Kreutzer, and Li Huang. KINNEWS and KIRNEWS: Benchmarking cross-lingual text classification for Kinyarwanda and Kirundi. In *Proceedings of the 28th International Conference on Computational Linguistics*, pp. 5507–5521, Barcelona, Spain (Online), December 2020. International Committee on Computational Linguistics. doi: 10.18653/v1/2020.coling-main.480. URL https://aclanthology.org/2020.coling-main.480.

Ossama Obeid, Nasser Zalmout, Salam Khalifa, Dima Taji, Mai Oudah, Bashar Alhafni, Go Inoue, Fadhl Eryani, Alexander Erdmann, and Nizar Habash. CAMeL tools: An open source python toolkit for Arabic natural language processing. In *Proceedings of the 12th Language Resources and Evaluation Conference*, pp. 7022–7032, Marseille, France, May 2020. European Language Resources Association. ISBN 979-10-95546-34-4. URL https://aclanthology.org/2020.lrec-1.868.

Kelechi Ogueji, Yuxin Zhu, and Jimmy Lin. Small data? no problem! exploring the viability of pretrained multilingual language models for low-resourced languages. In *Proceedings of the 1st Workshop on Multilingual Representation Learning*, pp. 116–126, Punta Cana, Dominican Republic, November 2021. Association for Computational Linguistics. doi: 10.18653/v1/2021.mrl-1.11. URL https://aclanthology.org/2021.mrl-1.11.

Xuan Ouyang, Shuohuan Wang, Chao Pang, Yu Sun, Hao Tian, Hua Wu, and Haifeng Wang. ERNIE-M: Enhanced multilingual representation by aligning cross-lingual semantics with monolingual corpora. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pp. 27–38, Online and Punta Cana, Dominican Republic, November 2021. Association for Computational Linguistics. doi: 10.18653/v1/2021.emnlp-main.3. URL https://aclanthology.org/2021.emnlp-main.3.

Chester Palen-Michel, June Kim, and Constantine Lignos. Multilingual open text 1.0: Public domain news in 44 languages. *CoRR*, abs/2201.05609, 2022. URL https://arxiv.org/abs/2201.05609.

Jonas Pfeiffer, Ivan Vulić, Iryna Gurevych, and Sebastian Ruder. MAD-X: An Adapter-Based Framework for Multi-Task Cross-Lingual Transfer. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pp. 7654–7673, Online, November 2020. Association for Computational Linguistics. doi: 10.18653/v1/2020.emnlp-main.617. URL https://aclanthology.org/2020.emnlp-main.617.

Jonas Pfeiffer, Ivan Vulić, Iryna Gurevych, and Sebastian Ruder. UNKs everywhere: Adapting multilingual language models to new scripts. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pp. 10186–10203, Online and Punta Cana, Dominican Republic, November 2021. Association for Computational Linguistics. doi: 10.18653/v1/2021.emnlp-main.800. URL https://aclanthology.org/2021.emnlp-main.800.

Sara Rosenthal, Noura Farra, and Preslav Nakov. Semeval-2017 task 4: Sentiment analysis in twitter. In *Proceedings of the 11th international workshop on semantic evaluation (SemEval-2017)*, pp. 502–518, 2017.

Victor Sanh, Lysandre Debut, Julien Chaumond, and Thomas Wolf. Distilbert, a distilled version of bert: smaller, faster, cheaper and lighter. *ArXiv*, abs/1910.01108, 2019.

Kathleen Siminyu, Godson Kalipe, Davor Orlic, Jade Z. Abbott, Vukosi Marivate, Sackey Freshia, Prateek Sibal, Bhanu Neupane, David Ifeoluwa Adelani, Amelia Taylor, Jamiil Toure Ali, Kevin Degila, Momboladji Balogoun, Thierno Ibrahima Diop, Davis David, Chayma Fourati, Hatem Haddad, and Malek Naski. Ai4d - african language program. *ArXiv*, abs/2104.02516, 2021.

Zhiqing Sun, Hongkun Yu, Xiaodan Song, Renjie Liu, Yiming Yang, and Denny Zhou. MobileBERT: a compact task-agnostic BERT for resource-limited devices. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pp. 2158–2170, Online, July 2020. Association for Computational Linguistics. doi: 10.18653/v1/2020.acl-main.195. URL https://aclanthology.org/2020.acl-main.195.

Y. Tang, C. Tran, Xian Li, Peng-Jen Chen, Naman Goyal, Vishrav Chaudhary, Jiatao Gu, and Angela Fan. Multilingual translation with extensible multilingual pretraining and finetuning. *ArXiv*, abs/2008.00401, 2020.

Erik F. Tjong Kim Sang and Fien De Meulder. Introduction to the CoNLL-2003 shared task: Language-independent named entity recognition. In *Proceedings of the Seventh Conference on Natural Language Learning at HLT-NAACL 2003*, pp. 142–147, 2003. URL https://aclanthology.org/W03-0419.

Wenhui Wang, Furu Wei, Li Dong, Hangbo Bao, Nan Yang, and Ming Zhou. Minilm: Deep self-attention distillation for task-agnostic compression of pretrained transformers. In H. Larochelle, M. Ranzato, R. Hadsell, M. F. Balcan, and H. Lin (eds.), *Advances in Neural Information Processing Systems*, volume 33, pp. 5776–5788. Curran Associates, Inc., 2020.

Wenhui Wang, Hangbo Bao, Shaohan Huang, Li Dong, and Furu Wei. MiniLMv2: Multi-head self-attention relation distillation for compressing pretrained transformers. In *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*, pp. 2140–2151, Online, August 2021. Association for Computational Linguistics. doi: 10.18653/v1/2021.findings-acl.188. URL https://aclanthology.org/2021.findings-acl.188.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Remi Louf, Morgan Funtowicz, Joe Davison, Sam Shleifer, Patrick von Platen, Clara Ma, Yacine Jernite, Julien Plu, Canwen Xu, Teven Le Scao, Sylvain Gugger, Mariama Drame, Quentin Lhoest, and Alexander Rush. Transformers: State-of-the-art natural language processing. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pp. 38–45, Online, October 2020. Association for Computational Linguistics. doi: 10.18653/v1/2020.emnlp-demos.6. URL https://aclanthology.org/2020.emnlp-demos.6.

Linting Xue, Noah Constant, Adam Roberts, Mihir Kale, Rami Al-Rfou, Aditya Siddhant, Aditya Barua, and Colin Raffel. mt5: A massively multilingual pre-trained text-to-text transformer. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pp. 483–498, Online, June 2021. Association for Computational Linguistics. doi: 10.18653/v1/2021.naacl-main.41. URL https://aclanthology.org/2021.naacl-main.41.

Seid Muhie Yimam, Hizkiel Mitiku Alemayehu, Abinew Ayele, and Chris Biemann. Exploring Amharic sentiment analysis from social media texts: Building annotation tools and classification models. In *Proceedings of the 28th International Conference on Computational Linguistics*, pp. 1048–1060, Barcelona, Spain (Online), December 2020. International Committee on Computational Linguistics. doi: 10.18653/v1/2020.coling-main.91. URL https://aclanthology.org/2020.coling-main.91.

Xiang Zhang, Junbo Jake Zhao, and Yann LeCun. Character-level convolutional networks for text classification. In *NIPS*, 2015.

A Appendix

A.1 Monolingual corpora for LAFT and MAFT

For training the MAFT models, we make use of the aggregation of monolingual data from Table 9.

For the LAFT models, we make use of existing XLMR-base+LAFT models from the MasakhaNER paper (Adelani et al., 2021). However, for other languages not present in MasakhaNER (`ara`, `mlg`,`orm`, `sna`, `som`, `xho`), we make use of the mC4 corpus except for `eng` — we use the VOA corpus. For a fair comparison across models, when training the XLM-R-large+LAFT models, we use the same monolingual corpus used to train XLM-R-base+LAFT models.

A.2 News corpora for language adapters and SFTs

Table 10 provides the news corpus we used to train language adapters and SFTs for the cross-lingual settings.

Language	Source	Size (MB)	No. of sentences
Afrikaans (afr)	mC4 (subset) (Xue et al., 2021)	752.2MB	3,697,430
Amharic (amh)	mC4 (subset), and VOA	1,300MB	2,913,801
Arabic (ara)	mC4 (subset)	1,300MB	3,939,375
English (eng)	mC4 (subset), and VOA	2,200MB	8,626,571
French (fra)	mC4 (subset), and VOA	960MB	4,731,196
Hausa (hau)	mC4 (all), and VOA	594.1MB	3,290,382
Igbo (ibo)	mC4 (all), and AfriBERTa Corpus (Ogueji et al., 2021)	287.5MB	1,534,825
Malagasy (mlg)	mC4 (all)	639.6MB	3,304,459
Chichewa (nya)	mC4 (all), Chichewa News Corpus (Siminyu et al., 2021)	373.8MB	2,203,040
Oromo (orm)	AfriBERTa Corpus, and VOA	67.3MB	490,399
Naija (pcm)	AfriBERTa Corpus	54.8MB	166,842
Rwanda-Rundi (kin/run)	AfriBERTa Corpus, KINNEWS & KIRNEWS (Niyongabo et al., 2020), and VOA	84MB	303,838
chiShona (sna)	mC4 (all), and VOA	545.2MB	2,693,028
Somali (som)	mC4 (all), and VOA	1,000MB	3,480,960
Sesotho (sot)	mC4 (all)	234MB	1,107,565
Kiswahili (swa)	mC4 (all)	823.5MB	4,220,346
isiXhosa (xho)	mC4 (all), and Isolezwe Newspaper	178.4MB	832,954
Yorùbá (yor)	mC4 (all), Alaroye News, Asejere News, Awikonko News, BBC, and VON	179.3MB	897,299
isiZulu (zul)	mC4 (all), and Isolezwe Newspaper	700.7MB	3,252,035

Table 9: Monolingual Corpora (after pre-processing – we followed AfriBERTa (Ogueji et al., 2021) approach) , their sources and size (MB), and number of sentences.

Language	Source	Size (MB)	No. of sentences
Amharic (amh)	VOA (Palen-Michel et al., 2022)	19.9MB	72,125
Hausa (hau)	VOA (Palen-Michel et al., 2022)	46.1MB	235,614
Igbo (ibo)	BBC Igbo (Ogueji et al., 2021)	16.6MB	62,654
Kinyarwanda (kin)	KINNEWS (Niyongabo et al., 2020)	35.8MB	61,910
Luganda (lug)	Bukedde	7.9MB	67,716
Luo (luo)	Ramogi FM news and MAFAND-MT (Adelani et al., 2022)	1.4MB	8,684
Naija (pcm)	BBC	50.2MB	161,843
Kiswahili (swa)	VOA (Palen-Michel et al., 2022)	17.1MB	88,314
Wolof (wol)	Lu Defu Waxu, Saabal, Wolof Online, and MAFAND-MT (Adelani et al., 2022)	2.3MB	13,868
Yorùbá (yor)	BBC Yorùbá	15.0MB	117,124

Table 10: Monolingual News Corpora used for language adapter and SFT training, their sources and size (MB), and number of sentences.

Noun Class Disambiguation in Runyankore and Related Languages

Joan Byamugisha
IBM Research Africa
45 Juta Street, Braamfontein
Johannesburg, South Africa
joan.byamugisha@ibm.com

Abstract

Bantu languages are spoken by communities in more than half of the countries on the African continent by an estimated third of a billion people. Despite this populous and the amount of high quality linguistic research done over the years, Bantu languages are still computationally under-resourced. The biggest limitation to the development of computational methods for processing Bantu language text is their complex grammatical structure, chiefly in the system of noun classes. We investigated the use of a combined syntactic and semantic method to disambiguate among singular nouns with the same class prefix but belonging to different noun classes. This combination uses the semantic generalizations of the types of nouns in each class to overcome the limitations of relying only on the prefixes they take. We used the nearest neighbors of a query word as semantic generalizations, and developed a tool to determine the noun class based on resources in Runyankore, a Bantu language indigenous to Uganda. We also investigated whether, with the same Runyankore resources, our method had utility in other Bantu languages, Luganda, indigenous to Uganda, and Kinyarwanda, indigenous to Rwanda. For all three languages, the combined approach resulted in an improvement in accuracy, as compared to using only the syntactic or the semantic approach.

1 Introduction

Over the last three decades, there has been an increase in the development of computational resources for Bantu languages. Much of this work applies the knowledge gained on Bantu language linguistics to digitize textual resources and develop software tools for text processing for a single language or for a group of languages. Among the textual resources created are the SAWA English-Kiswahili parallel corpus for machine learning (De Pauw et al., 2011) and the labelled and unlabelled Runyankore datasets (Byamugisha, 2020). The text processing resources include morphological tools, such as morphological analyzers for isiZulu (Bosch and Pretorius, 2003, 2004), isiXhosa, seSwati, and isiNdebele (Bosch et al., 2008); text generation tools, such as a morphological generator for isiZulu (Bosch and Pretorius, 2003) and surface realizers for isiZulu (Keet et al., 2017) and Runyankore (Byamugisha et al., 2017a,b); part-of-speech taggers for Kiswahili, Ciluba, Northern Sotho, and isiZulu (De Pauw et al., 2012); and noun pluralization tools for isiZulu and Runyankore (Byamugisha et al., 2016), chiShona, isiXhosa, Kikuyu, Kinyarwanda, and Luganda (Byamugisha et al., 2018).

Despite these efforts, Bantu languages are still among the most computationally under-resourced languages in the world. This is due to their complex grammatical structure, mainly the noun class system, verb morphology, and agglutinative morphology. In this paper, we focus on the noun class system only, the hallmark of Bantu nominal morphology (Katamba, 2003). This system places every noun in a language into a class, based on the semantics of a noun first (such as whether the noun is of a human or non-human entity), then the morphology of a noun next (which is based on the prefix of a noun) (Katamba, 2003). The importance of the noun class to computational text processing goes beyond classifying nouns, and also determines the formulation of other parts of speech, such as adjectives, verbs, possessives, determinants, grammatical number, etc. because a noun class is central to an extensive system of concordial agreement that determines morphological composition (Katamba, 2003).

Given that there exists in some noun class systems class prefixes that are not unique among different noun classes, a problem of ambiguity exists when determining a noun class using a class prefix only, as is the case with morphological approaches. We refer to the process of determining the correct

Proceedings of the 29th International Conference on Computational Linguistics, pages 4350–4359
October 12–17, 2022.

noun class under these circumstances as noun class disambiguation, which, to the best of our knowledge, has not yet been solved. Katamba (2003) state that using a noun's semantics and extending beyond morphology to syntax by considering concords, can overcome the limitations of morphological approaches, but it requires large resources that capture the context in which a noun is used. We therefore investigated whether it is possible to extend morphological approaches to syntactic approaches by including the syntax of an entire sentence, and further combine this with the semantics of a noun in order to undertake noun class disambiguation. We used the following questions:

1. To what extent can a noun's semantic generalizations, in the form of nearest neighbors, work to disambiguate among noun classes to identify its noun class correctly?

2. Can the presence of sub-word information in word vectors in one language contribute enough semantic information to improve noun class disambiguation in another Bantu language?

We investigated the applicability of a combined syntactic and semantic approach as a means of noun class disambiguation among singular nouns, using word vectors pre-trained using FastText (Bojanowski et al., 2016; Joulin et al., 2016) on two one million sentence datasets in Runyankore, one dataset unlabelled and the other labelled with morphological information including the noun class (Byamugisha, 2020). The syntactic method relies both on the morphology of a noun based on its prefix and on the syntax of other grammatical units in a sentence which are determined by a noun class. The semantic method uses the noun class labels of the nearest neighbors of a query word. Only singular nouns were used in this investigation because there is ground-truth data from the singular wordlists used in the noun pluralization tools with which to evaluate the results, and these same tools apply the knowledge on the singular/plural pairings to solve the problem of plurals computationally. We started with two datasets in Runyankore, the same language as the word vectors used, and obtained accuracies of 80.54% and 85.23% on two test sets. We then investigated whether, using a combined approach with the same Runyankore word vectors, a correct noun class determination can be made for another Bantu language. Using Luganda and Kinyarwanda, we obtained accuracies of 73.97% and 63.64%, respectively. To the best of our knowledge, this is the first computational attempt to use both the syntactic and semantic characteristics of a noun to disambiguate among noun classes with the same class prefix.

The rest of this paper is arranged as follows: Section 2 provides information on Bantu languages, with a focus on the noun class system; Section 3 details the materials, methods, and results from using a combined syntactic and semantic approach to determine a noun class; Section 4 discusses the implications of this work; and we conclude in Section 5.

2 Bantu Language Noun Class System

Bantu languages are a group of languages indigenous to Africa (Nurse and Philippson, 2003). They extend from the south, below Nigeria, to most of central, east, and southern Africa, as shown in Figure 1 (Nurse and Philippson, 2003). There are Bantu-speaking communities in 27 of the continent's 54 countries, with about 240 million speakers (Nurse and Philippson, 2003). The exact number of languages classified as Bantu ranges from 300 to 680, based on different criteria by different authors (Nurse and Philippson, 2003).

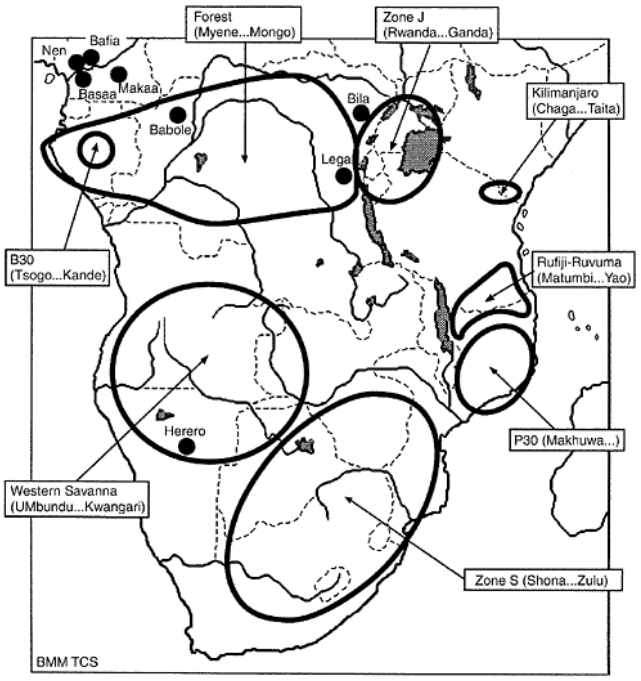

Figure 1: The spread of Bantu languages across Africa (Nurse and Philippson, 2003)

Bantu languages assign all nouns to a class re-

ferred to as a noun class (NC); and there are over 20 noun classes, though some NCs have fallen into disuse in most languages (Nurse and Philippson, 2003; Mohlala, 2003; Maho, 1999). The semantic generalizations of the types of nouns in each class are shown in Table 1 (Keet and Khumalo, 2014; Baertlein and Ssekitto, 2014; Kimenyi, 2004; Jeon et al., 2015; Zentz, 2016; Taraldsen, 2010; Mohlala, 2003; Katamba, 2003; Maho, 1999).

Noun Class	Description of Associated Nouns
1 and 2	People and kinship
3 and 4	Plants, nature, and some parts of the body
5 and 6	Fruits, liquids, some parts of the body, and paired things
7 and 8	Inanimate objects
9 and 10	Tools and animals
11	Long thin stringy objects, languages, and inanimate objects
12 and 13	Diminutives
14	Abstract concepts
15	Infinitives and parts of the body
16, 17, and 18	Locative classes
19	Diminutives
20, 21, and 22	Augmentatives
23	Locative class

Table 1: Classification of Bantu nouns into noun classes (the 'and' indicates that the two classes are a singular/plural pairing)

The simple noun comprises a prefix and a stem (Katamba, 2003); for example, *omuntu* 'person' in Runyankore, which can be analyzed as the prefix *o-mu-* and stem *-ntu*. Therefore, in addition to the semantic categorization of nouns shown in Table 1, nouns are categorized morphologically also, according to the prefixes they take (Katamba, 2003). In many Bantu languages, the class prefix may be preceded by a formative referred to as the augment, pre-prefix, or initial vowel (Katamba, 2003; Maho, 1999). In the above example of *omuntu*, the class prefix *o-mu-* possesses the augment *o*. The augment is not found in all Bantu languages (Katamba, 2003; Maho, 1999).

Noun classes do not only classify nouns, but are at the heart of an extensive system of concordial agreement (Katamba, 2003), where each class determines the agreement with: concord patterns; nominal prefix in nouns, locatives, and adjectives; numeral prefix; pronominal prefix for substitutives, connectives, possessives, demonstratives, and determinatives; initial prefix in absolutive verb forms; and the verbal infix (Katamba, 2003; Maho, 1999; Zentz, 2016; Nurse and Philippson, 2003; Tayebwa, 2014). Given how central the noun class system is to Bantu language computational linguistics, it is important to identify a means of determining the noun class of a noun.

3 Noun Class Disambiguation Using Syntax and Semantics

In this research, we aimed to find out to what extent determining the semantics of a noun is beneficial to disambiguating among noun classes, in order to determine its noun class correctly. In Byamugisha (2020), it was shown that from word vectors trained on one million Runyankore sentences in an unsupervised manner, the nearest neighbors (co-occurrence vectors) obtained for a query word reflect the semantic generalizations of noun classes shown in Table 1. We, therefore, used a noun's nearest neighbors as a representation of its semantic generalizations. We investigated further whether, with the pre-trained word vectors in one Bantu language, the nearest neighbors of a query word in a different Bantu language improve the disambiguation, and, consequently, determination of a noun class in this other Bantu language. Four criteria are relied upon when selecting languages for this research: (1) the availability of linguistic information about its noun class; (2) the presence of an augment in a class prefix of a noun, as is the case with Runyankore; (3) the placement of people and kinship nouns in the same noun class (instead of two different classes), as is the case in Runyankore's noun class system; and (4) the availability of ground-truth data with which to evaluate the output. Based on this, Luganda and Kinyarwanda were selected from the existing computational resources for seven Bantu languages in Byamugisha et al. (2018).

Our approach to disambiguating among noun classes of a noun also extends beyond the morphology of a noun and considers the presence of the concords in a sentence that are indicative of a noun class. We combine this method with a noun's semantics as defined by its nearest neighbors. The details on each of these two methods are explained in the following sections.

3.1 Syntactic Approach

As explained by Katamba (2003), the assignment of nouns to a class is partially morphological, using rules based on class prefixes (which may or may not be unique) according to the noun class system of a particular language. A syntactic approach to disambiguating among noun classes involves

analyzing the syntax of an entire sentence to confirm whether other grammatical units such as the possessive, subject, or object concords (which are always unique in a noun class system) are indicative of a particular noun class. Runyankore, Luganda, and Kinyarwanda have different noun class systems, and, therefore, require different morphological rules for each language. The class prefixes for the noun class systems of these languages are shown in Table 2.

Noun Class	Runyankore	Luganda	Kinyarwanda
1	o-mu-	o-mu-	u-mu-
2	a-ba-	a-ba-	a-ba-
3	o-mu-	o-mu-	u-mu-
4	e-mi-	e-mi-	i-mi-
5	e-ri-/ei-	e-li-/e-	i-ri-/i-
6	a-ma-	a-ma-	a-ma-
7	e-ki-	e-ki-	i-ki-/i-cy-/i-gi-
8	e-bi-	e-bi-	i-bi-
9	em-/em-	e-n-	i-/i-n-/i-nz-
10	em-/em-	e-n-	i-/i-n-/i-nz-
11	o-ru-	o-lu-	u-ru-
12	a-ka-	a-ka-	a-ka-/a-ga-
13	o-tu-	o-tu-	u-tu-/u-du-
14	o-bu-	o-bu-	u-bu-
15	o-ku-	o-ku-	u-ku-/u-gu-
16	a-ha-	wa-	a-ha-
17	o-ku-	ku-	N/A
18	o-mu-	mu-	N/A
19	N/A	N/A	N/A
20	o-gu-	o-gu-	N/A
21	a-ga-	a-ga-	N/A
22	N/A	N/A	N/A
23	N/A	e-	N/A

Table 2: The noun classes for Runyankore (Asiimwe, 2014), Luganda (Baertlein and Ssekitto, 2014), and Kinyarwanda (Kimenyi, 2004), showing the class prefixes. The dashes between the letters in the prefix illustrate separation between the augment and prefix; and 'N/A', the NC is not present in that language.

Table 2 shows the noun classes and the class prefixes for Runyankore, Luganda, and Kinyarwanda. They have different numbers of noun classes–20 in Runyankore, 21 in Luganda, and 16 in Kinyarwanda–because some noun classes have fallen into disuse in most languages (Nurse and Philippson, 2003; Mohlala, 2003; Maho, 1999). None of these languages has classes 19 and 22.

Morphological rules, based on a class prefix, are used first to attempt to determine a noun's noun class. However, there are some prefixes that are the same for different noun classes. In Table 2, these are classes 1, 3, and 18 with prefix *o-mu-* and classes 15 and 17 with prefix *o-ku-* in Runyankore; classes 1 and 3 with prefix *o-mu-* and classes 5, 9, and 23 with prefix *e-* in Luganda; and classes 1 and 3 with prefix *u-mu-* and classes 5 and 9 with prefix *i-* in Kinyarwanda. This results in ambiguity during noun class determination, which cannot be resolved by a morphological approach only. On the other hand, concords can be used to disambiguate between nouns belonging to different classes but with the same class prefix because their concords differ (Katamba, 2003; Maho, 1999). Table 3 shows the subject concords in Runyankore's noun class system.

Noun Class	Class Prefix	Subject Concord
1	o-mu-	-a-
2	a-ba-	-ba-
3	o-mu-	-gu-
4	e-mi-	-gi-
5	ei-/e-ri-	-ri-
6	a-ma-	-ga-
7	e-ki-	-ki-
8	e-bi-	-bi-
9	e-n-/e-m-	-e-
10	e-n-/e-m-	-zi
11	o-ru-	-ru-
12	a-ka-	-ka-
13	o-tu-	-tu-
14	o-bu-	-bu-
15	o-ku-	-ku
16	a-ha-	-ha
17	o-ku-	-ha-
18	o-mu-	-ha-
20	o-gu-	-gu-
21	a-ga-	-ga-

Table 3: The Subject concords of the Runyankore noun class system, showing that concords are unique across classes with the same prefix

Table 3 shows that for classes 1, 3, and 18 with prefix *o-mu-* and classes 15 and 17 with prefix *o-ku-*, the subject concords are unique among them, and can thus be used to disambiguate among these classes. We, therefore, extended beyond morphology, to syntax, by including an entire sentence in our approach.

3.2 Semantic Approach

According to Katamba (2003), the assignment of nouns to a class is also partially based on semantic generalizations of the types of nouns in each class, as shown in Table 1. In Byamugisha (2020), the nearest neighbors obtained from word vectors trained on one million Runyankore sentences were found to have a high level of semantic relatedness. Table 4 from Byamugisha (2020) shows the nearest neighbors for query words with the prefix *o-mu-* but belonging to different noun classes.

The examples in Table 4 present a case of noun class ambiguity using three query words (*omuntu*,

Query Word	Nearest Neighbors
omuntu (person)	*omugyesi* (reaper), *omutaahi* (companion), *omukoreesa* (overseer), *omushomesa* (teacher), *omukuru* (elder)
omuti (tree)	*omutumba* (banana tree), *omwani* (coffee tree), *omuzaabibu* (grape or grapevine), *omucungwa* (orange), *omugusha* (sorghum)
omukono (arm)	*omunwa* (mouth), *omutwe* (head), *eriino* (tooth), *enkokora* (elbow), *okuguru* (leg)

Table 4: Nearest Neighbors for query words with the prefix *o-mu-* (Byamugisha, 2020)

omuti, and *omukono*) which all have the same noun prefix, *o-mu-*. However, they belong to different noun classes, with *omuntu* in class 1, semantically for people according to Table 1, and *omuti* and *omukono* in class 3. The nearest neighbors of these query words reflect a semantic distinction among them, which cannot be determined syntactically. On the other hand, the nearest neighbors of *omukono* in Table 4 belong to different noun classes morphologically: *omunwa* and *omutwe* in class 3, *eriino* in class 5, *enkokora* in class 9, and *okuguru* in class 15 according to Table 2. Therefore, whilst the nearest neighbors help to exclude class 1 for *omukono*, the disambiguation is made syntactically, by going beyond a noun's prefix and extending to the concords in a sentence. Our approach thus combines syntax and semantics in order to leverage the benefits of both to disambiguate among noun classes.

3.3 Materials

The materials used to develop and evaluate a combined syntactic and semantic approach to noun class disambiguation were obtained from existing computational resources. These included: (1) pre-trained word vectors in Runyankore from Byamugisha (2020); (2) a classifier trained on one million sentences, in Runyankore labelled for parts-of-speech and morphology (including the noun class assignment of nouns, subject and object concords, and adjective prefixes) from Byamugisha (2020); (3) a dataset of singular nouns and their correct noun classes in Runyankore to act as ground-truth during development, obtained from Set1r and Set2r in Byamugisha et al. (2016); and (4) a dataset of singular nouns and their correct noun classes in Runyankore, Luganda, and Kinyarwanda, to act as ground-truth during evaluation, obtained from SetI and SetC in Byamugisha et al. (2018). Table 5 shows the language and number of nouns in each of these datasets.

Dataset	Language	Number of Nouns
Set1r	Runyankore	92
Set2r	Runyankore	2542
SetI	Runyankore	81
SetC	Runyankore	88
SetI	Luganda	75
SetC	Luganda	78
SetI	Kinyarwanda	70
SetC	Kinyarwanda	-

Table 5: Details on datasets used

The datasets[1] in Table 5 contain a noun and its noun class. These ground-truth datasets contain singular nouns only (with the exception of mass nouns), thus, we consider only singular nouns so far. The need for noun class disambiguation is evident in the statistical characteristics of these datasets: 25.0% of nouns in Set1R have the prefix *om-* and 51.09% have the prefix *e-*; 39.89% of nouns in Set2r have the prefix *om-*, while 34.97% have the prefix *e-*. For Runyankore, 17.05% of nouns in SetI have the prefix *om-* and 40.91% have the prefix *e-*; while 28.4% of nouns in SetC have the prefix *om-* while 58.02% have the prefix *e-*. For Luganda, 23.07% of nouns in SetI have prefix *om-* and 38.46% have prefix *e-*; while 16.0% of nouns in SetC have prefix *om-* and 45.33% have prefix *e-*. For Kinyarwanda, with SetI only, 18.57% of nouns have prefix *um-* and 44.29% have prefix *e-*.

3.4 Methods

We used an iterative approach to develop a tool that uses a noun's syntax and semantics to disambiguate first, and then determine its noun class. This involved testing different versions of the tool for Runyankore using Set1r and Set2r, adding new functionality, and then testing again. The tool was developed as a Python module using the Gensim library[2] to load the pre-trained word-vectors of a language model for Runyankore from Byamugisha (2020). The FastText library[3] was used to train a classifier on a one million sentence labeled dataset also from Byamugisha (2020). The dataset was

[1] Set1r and Set2r can be obtained from https://github.com/ThesisResources/RunyankorePluralizer, while SetI and SetC can be found at https://github.com/runyankorenlg/Generic-Pluralizer.

[2] For more details on the Gensim Python library, see https://pypi.org/project/gensim/.

[3] For more details on the FastText Python library, see https://pypi.org/project/fasttext/.

split into 70% for training, 20% for validation, and 10% for testing. We used the default values for the training parameters of the FastText library. Figure 2 shows the architecture of the method used for noun class disambiguation.

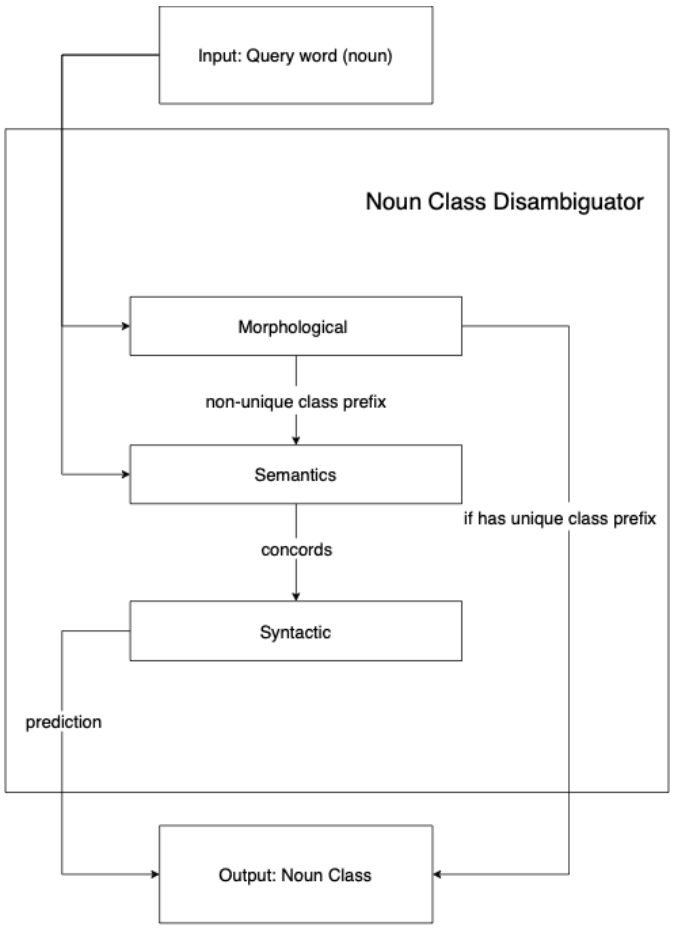

Figure 2: Method taken for noun class disambiguation

Figure 2 shows how a noun class of a noun is determined, as well as where disambiguation is required. Given a noun, a morphological step is performed first, but this results in a noun class only if the class prefix of the noun is unique (see tables 2 and 3 for the class prefixes). On the other hand, if the class prefix is not unique, then the noun is regarded as a query word and undergoes the semantic method, during which the following happens:

- Extracting a vocabulary from the pre-trained language model, and including all the sub-word information in the model as part of the vocabulary;
- Finding the noun's nearest neighbors from the pre-trained language model, which should include the sub-word results in the vocabulary; and
- Predicting, from the top-n nearest neighbors obtained, a label for each nearest neighbor using the classifier.

The results from the semantic method go through the syntactic method, where concords in an entire sentence can be used to determine the final label because concords are unique among noun classes (Katamba, 2003; Maho, 1999) (see Table 3 for subject concords in Runyankore). This involves:

- Excluding items from the list of nearest neighbors if the label predicted for their concord (sub-word) is not consistent with the same noun class; and
- Predicting the final noun class based on the final list of nearest neighbors

Table 6 shows the improvement in accuracy on the two test sets as different versions of the tool were implemented.

Version	Set1r	Set2r
Classical nearest neighbors	52.22	41.13
Annoy nearest neighbors	54.22	41.26
Top 10 Annoy nearest neighbors + no verbs	63.33	54.67
Top 110 Annoy nearest neighbors + no verbs	73.86	65.72
Top 110 Annoy nearest neighbors + no verbs + retrofitted	76.14	68.45
Morphological and top 110 Annoy nearest neighbors + no verbs + retrofitted + syntactic	85.23	80.54

Table 6: Improvements in accuracy of the different versions developed to disambiguate among noun classes on two Runyankore test sets

The first version of the tool was based only on the semantics, and used the classical method to obtain the nearest neighbors. There was an improvement in performance when Annoy[4] nearest neighbors were used instead of the classical method, as shown in Table 6. Next, we used a set of ten Annoy nearest neighbors, omitting any verbs from this set because the verb root does not contain any noun class information, and this also improved the accuracy. We investigated the optimum number of nearest neighbors that result in the highest accuracy, starting from 10 up to 1000. We found 110 to be

[4]For details on the Annoy algorithm, see https://pypi.org/project/annoy/1.0.3/.

the optimum number that results in the accuracies shown in Table 6. This number is also large enough to include the concords required for the syntactic approach.

The next version of the tool added on to the previous one by using the retrofit algorithm by Faruqui et al. (2015) in an attempt to enable words to be related semantically in a better manner by having similar vector representations. We extracted a lexicon of nouns from a Runyankore dictionary (Taylor, 2009), arranged according to their noun classes, and used it to retrofit the pre-trained word vectors. We then calculated the accuracy at this point and obtained the results shown in Table 6. This represents the current combination of methods associated with the semantic and syntactic approaches. The final version starts with a morphological method to handle nouns with unique class prefixes in the input datasets, followed by the semantic and syntactic methods when disambiguation is required, in order to obtain the final accuracies of 85.23% and 80.54% on Set1r and Set2r, respectively.

Next, we report on the results from evaluating this combined approach on Runyankore, Luganda, and Kinyarwanda.

3.5 Results on Evaluation

We carried out an evaluation of the tool developed to disambiguate and determine a noun's class along two lines of enquiry: (1) whether a combined approach achieves better results than a morphological or semantic method alone; and (2) whether, without changing the underlying resources that are the basis of the semantic approach, a combined approach can still achieve better results than the individual approaches when applied to languages other than Runyankore. Table 7 shows the results of the evaluation for Runyankore, Luganda, and Kinyarwanda ("NCS" refers to "Noun Class System"). The metric for evaluation is accuracy, determined as the percentage of correct noun classes determined over the test sets.

The results in Table 7 show clearly that a combined syntactic and semantic approach achieves better results than the individual approaches. Though not shown in Table 7, accuracies of 84.88% and 70.67% were obtain on SectC for Runyankore and Luganda, respectively (there was no dataset for SetC in Kinyarwanda). Additionally, the results show that an improvement in performance is achieved in Luganda and Kinyarwanda despite

Approach	Runyankore	Luganda	Kinyarwanda
Morphological only	69.23	57.53	43.94
Semantic only	66.67	47.95	40.91
Combination, with Runyankore NCS	87.18	67.12	31.82
Combination, with language NCS	87.18	73.97	63.64

Table 7: Results of the evaluation on the morphological only, semantic only, and combined syntactic and semantic approaches on Runyankore, Luganda, and Kinyarwanda datasets

the underlying semantic resources belonging to Runyankore. This is an important result given the under-resourced state of Bantu languages, where the reuse of resources during text processing will always be advantageous, because developing the same resources in Runyankore for another Bantu language requires significant time and effort (Byamugisha, 2020).

While it is not surprising that the best results are achieved when the syntactic approach is based on the concords of a test language's noun class system, it is nonetheless interesting that there is a 10% improvement in accuracy for Luganda even when the syntactic approach uses Runyankore's noun class system. This can be explained by how similar their noun class systems are (see Table 2), suggesting, possibly, a potential for direct reuse of the tool among closely related languages.

4 Discussion

We developed an approach to disambiguate among noun classes in order to determine a noun's class that combines a noun's syntax and semantics in order to achieve the best results. Our approach is in line with literature on the Bantu noun class system that states that the assignment of nouns to a class is based on semantic generalizations of the types of nouns in each class, as well as morphologically, according to their class prefixes, and syntactically, according to the concords they take (Katamba, 2003). While our work is focused on Bantu languages, it is important to note that noun class systems are a strong areal feature in Africa, with an estimated two-thirds of the languages on the continent having noun classes (Katamba, 2003). The theory on which our approach is based might be applicable

to another family of languages in Africa.

That a combined syntactic and semantic approach is necessary during text processing for Bantu languages is not novel, as this is the basis of the tools on noun pluralization for Runyankore and isiZulu (Byamugisha et al., 2016) and chiShona, isiXhosa, Kikuyu, Kinyarwanda, and Luganda (Byamugisha et al., 2018). What is emphasized by our results, however, is that, for the semantics, a little goes a long way. Only 385 singular nouns were used in the generation of the Runyankore datasets (Byamugisha, 2020) on which the word vectors were trained; yet, as seen in the results in Table 6, an accuracy of over 80% was achieved on Set2r that comprises over 2000 nouns.

The main reason for needing a semantic approach is to overcome noun class ambiguity, a situation where nouns belonging to different noun classes have the same class prefix. Therefore, though useful, the reuse of the resources in one language to determine the semantics of another language is limited if the source language does not account for the nouns affected in the target language. For example, in addition to classes 1 and 3 having the same prefix in Luganda and Kinyarwanda, classes 5, 9, and 23 in Luganda also have the same prefix, *e-*, and classes 5 and 9 in Kinyarwanda also have the same prefix, *i-*; yet their semantic distinctions are not all captured in the Runyankore resources. Having language-specific resources is the desired outcome, though the ability to reuse resources provides a benefit.

The increase in accuracy on test sets in languages that are different from the resources on which the semantics are determined is also a notable finding. Though the selection of Luganda and Kinyarwanda was purposively based on the availability of resources, their results cannot be removed entirely from their grouping in Guthrie zones as compared to Runyankore. Guthrie zones are a referential classification of Bantu languages which groups together languages with similar linguistic features (such as phonetic, semantic, and syntactic) that are geographically colocated, without presupposing their historical relatedness (Schadeberg, 2003). Guthrie zones categorize Bantu languages into 16 geographic zones, which are labeled using the letters A, B, C, D, E, F, G, H, J, K, L, M, N, P, R, and S; and these are further subdivided into decades (zone J.10, where Runyankore belongs, contains individual languages labeled from J.11 to J.19, while J.20, J.30, etc. represent different zones) (Nurse and Philippson, 2003; Schadeberg, 2003; Maho, 2009). Figure 3 shows how this classification covers the Bantu languages throughout the African continent.

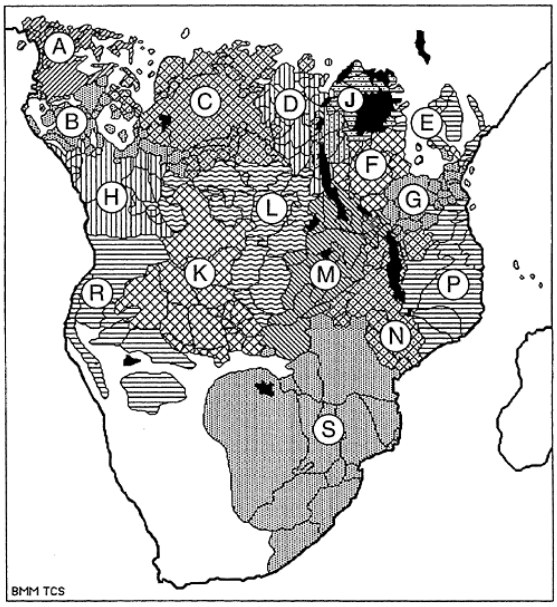

Figure 3: The spread and classification of Bantu languages into Guthrie zones (Maho, 2009)

Runyankore and Luganda are in zone J.10, while Kinyarwanda is in zone J.60 (Maho, 2009). When considered from the perspective of Guthrie zone classification, the results in Table 7 provide evidence, though limited, for the reuse of language resources both within and across Guthrie zones, with the best results obtained for the former. However, this needs to be investigated further before any conclusions can be made.

5 Conclusion

In this paper, we have presented the use of a combined syntactic and semantic approach to address noun class ambiguity when determining a singular noun's class. The semantic approach, which is based on the type of the noun, is necessary to perform noun class disambiguation, which addresses the main limitation of the morphological approach, that is, the presence of nouns with the same prefix but belonging to different noun classes. We used a noun's nearest neighbors in word vectors as representations of a noun's semantic generalizations, and a noun's concords, together with its morphology, as representations of a noun's syntactic association. We developed a tool based on our

combined approach using pre-trained word vectors in Runyankore, and evaluated the accuracy of the tool with two Runyankore datasets, achieving up to 87.18% accuracy. We also evaluated the reusability of our approach to other Bantu languages, Luganda and Kinyarwanda, whilst relying on Runyankore resources for the semantic approach. We achieved accuracies of 73.97% in Luganda and 63.64% in Kinyarwanda. Additionally, for all three languages, the combined approach performed better than individual morphological and semantic approaches.

References

Allen Asiimwe. 2014. *Definiteness and Specificity in Runyankore-Rukiga*. Ph.D. thesis, Stellenbosch University, Western Cape, South Africa.

Elizabeth Baertlein and Martin Ssekitto. 2014. Luganda nouns inflectional morphology and tests. *Linguistic Portfolios*, 3.

Piotr Bojanowski, Edouard Grave, Armand Joulin, and Tomas Mikolov. 2016. Enriching word vectors with subword information. *arXiv preprint arXiv:1607.04606*.

Sonja Bosch, Laurette Pretorius, and Axel Fleisch. 2008. Experimental bootstrapping of morphological analyzers for Nguni languages. *Nordic Journal of African Studies*, 17(2):66–88.

Sonja E. Bosch and Laurette Pretorius. 2003. Building a computational morphological analyser/generator for Zulu using the Xerox finite-state tools. In *Workshop on Finite-State Methods in Natural Language Processing, 10th Conference of the European Chapter of the Association for Computational Linguistics*, pages 27–34, Budapest. Association for Computational Linguistics.

Sonja E. Bosch and Laurette Pretorius. 2004. Software tools for morphological tagging of Zulu corpora and lexicon development. In *4th International Language Resources and Evaluation Conference*, pages 1251–1254, Lisbon, Portugal.

Joan Byamugisha. 2020. Generating varied training corpora in Runyankore using a combined semantic and syntactic, pattern-grammar-based approach. In *13th International Conference on Natural Language Generation*, pages 273–282.

Joan Byamugisha, C. Maria Keet, and Brian DeRenzi. 2017a. Evaluation of a Runyankore grammar engine for healthcare messages. In *10th International Conference on Natural Language Generation (INLG 2017)*, volume 4, pages 105–113, Santiago de Compostela, Spain. ACL.

Joan Byamugisha, C. Maria Keet, and Brian DeRenzi. 2017b. Toward an nlg system for Bantu languages: first steps with Runyankore (demo). In *10th International Conference on Natural Language Generation (INLG 2017)*, volume 4, Santiago de Compostela, Spain. ACL.

Joan Byamugisha, C. Maria Keet, and Brian DeRenzi. 2018. Pluralizing nouns in agglutinating Bantu languages. In *27th International Conference on Computational Linguistics (COLING 2018)*, pages 2633–2643, Santa Fe, New Mexico, USA. ACL.

Joan Byamugisha, C. Maria Keet, and Langa Khumalo. 2016. Pluralizing nouns in isiZulu and related languages. In *17th International Conference on Intelligent Text Processing and Computational Linguistics (CICLing 2016)*, volume 9626, pages 271–283, Konya, Turkey. Springer LNCS.

Guy De Pauw, Gilles-Maurice de Schryver, and Janneke van de Loo. 2012. Resource-light Bantu part-of-speech tagging. In *Workshop on Language Technology for Normalisation of Less-Resourced Languages (SaLTMiL 8 - AfLaT 2012)*, pages 85–92, Istanbul, Turkey.

Guy De Pauw, Peter W. Wagacha, and Gilles-Maurice de Schryver. 2011. Exploring the SAWA corpus: Collection and deployment of a parallel corpus English-Swahili. *Language Resources and Evaluation*, 45(3):331–344.

Manaal Faruqui, Jesse Dodge, Sujay K Jauhar, Chris Dyer, Eduard Hovy, and Noah A Smith. 2015. Retrofitting word vectors to semantic lexicons. *arXiv:1411.4166 [cs]*.

Lisa Jeon, Jessica Li, Samantha Mauney, Anai Navarro, and Jonas Wittke. 2015. A basic sketch grammar of Gikuyu. *Rice Working Papers in Linguistics*, 6.

Armand Joulin, Edouard Grave, Piotr Bojanowski, and Tomas Mikolov. 2016. Bag of tricks for efficient text classification. *arXiv preprint arXiv:1607.01759*.

Francis Katamba. 2003. Bantu nominal morphology. In *The Bantu Languages: Routledge Language Family Series 4*, chapter 7, pages 103–120. Taylor and Francis Routledge, London.

C. M. Keet, M. Xakaza, and L. Khumalo. 2017. Verbalising owl ontologies in isiZulu with python. In *14th Extended Semantic Web Conference (ESWC'17)*, volume 10577, pages 59–64, Portoroz, Slovenia. Springer LNCS.

C. Maria Keet and Langa Khumalo. 2014. Towards verbalizing logical theories in isiZulu. In *4th Workshop on Controlled Natural Languages (CNL'14)*, volume 8625, pages 78–89, Galway, Ireland. Springer LANI.

Alex Kimenyi. 2004. Kinyarwanda morphology. In Geert Booij, Christian Lehmann, Joachim Mudgan, and Stavros Skopeteas, editors, *Morphology: An International Handbook for Inflection and Word Formation*, volume 17.2. De Gruyter.

Jouni Maho. 1999. *A Comparative Study of Bantu Noun Classes*. Ph.D. thesis, Goteborg University, Goteborg, Sweden.

Jouni Filip Maho. 2009. NUGL online: The online version of the updated Guthrie list, a referential classification of the Bantu languages.

Linkie Mohlala. 2003. The Bantu attribute noun class prefixes and their suffixal counterparts, with special reference to Zulu. Master's thesis, University of Pretoria, Pretoria, South Africa.

Derek Nurse and Gerard Philippson. 2003. Introduction. In *The Bantu Languages: Routledge Language Family Series 4*, chapter 1, pages 1–9. Taylor and Francis Routledge, London.

C. Thilo Schadeberg. 2003. Historical linguistics. In *The Bantu Languages: Routledge Language Family Series 4*, chapter 9, pages 143–181. Taylor and Francis Routledge, London.

Knut Tarald Taraldsen. 2010. The nanosyntax of Nguni noun class prefixes and concords. *Lingua*, 120(6):1522–1548.

Doreen Daphine Tayebwa. 2014. Demonstrative determiners in Runyankore-Rukiga. Master's thesis, Norwegian University of Science and Technology, Norway.

C. Taylor. 2009. *A Simplified Runyankore-Rukiga-English Dictionary*. Fountain Publishers, Kampala, Uganda.

Jason Zentz. 2016. *Forming Wh-Questions in Shona: A Comparative Bantu Perspective*. Ph.D. thesis, Yale University.

Improving Low-resource RRG Parsing with Cross-lingual Self-training

Kilian Evang Laura Kallmeyer Jakub Waszczuk
Kilu von Prince Tatiana Bladier Simon Petitjean
Heinrich Heine University Düsseldorf, Germany
first.last@hhu.de

Abstract

This paper considers the task of parsing low-resource languages in a scenario where parallel English data and also a limited seed of annotated sentences in the target language are available, as for example in bootstrapping parallel treebanks. We focus on constituency parsing using Role and Reference Grammar (RRG), a theory that has so far been understudied in computational linguistics but that is widely used in typological research, i.e., in particular in the context of low-resource languages. Starting from an existing RRG parser, we propose two strategies for low-resource parsing: first, we extend the parsing model into a cross-lingual parser, exploiting the parallel data in the high-resource language and unsupervised word alignments by providing internal states of the source-language parser to the target-language parser. Second, we adopt self-training, thereby iteratively expanding the training data, starting from the seed, by including the most confident new parses in each round. Both in simulated scenarios and with a real low-resource language (Daakaka), we find substantial and complementary improvements from both self-training and cross-lingual parsing. Moreover, we also experimented with using gloss embeddings in addition to token embeddings in the target language, and this also improves results. Finally, starting from what we have for Daakaka, we also consider parsing a related language (Dalkalaen) where glosses and English translations are available but no annotated trees at all, i.e., a no-resource scenario wrt. syntactic annotations. We start with cross-lingual parser trained on Daakaka with glosses and use self-training to adapt it to Dalkalaen. The results are surprisingly good.[1]

1 Introduction

Treebanks play an increasingly important role in typological research, where linguists are oftentimes faced with (primarily) oral low-resource languages and where grammatical frameworks such as Role and Reference Grammar (RRG; Van Valin and Foley, 1980; Van Valin, 2005) are a common choice (Toratani and González Vergara, 2020). RRG is a non-transformational linguistic theory strongly inspired by typological concerns. Its development was guided by the question of what a linguistic theory would "look like if it were based on the analysis of languages with diverse structures such as Lakhota, Tagalog and Dyirbal [...]?" (Van Valin, 2005, p. 1). RRG assumes constituency structures to be organized in layers, *viz.* nucleus (containing the predicate), core (containing the nucleus and the arguments of the predicate) and clause (the core and extracted arguments). Furthermore, each layer can have modifiers (termed periphery elements) and operators. An example from Daakaka (an Oceanic language, von Prince, 2015) is given in Fig. 1, using the annotation scheme from Bladier et al.

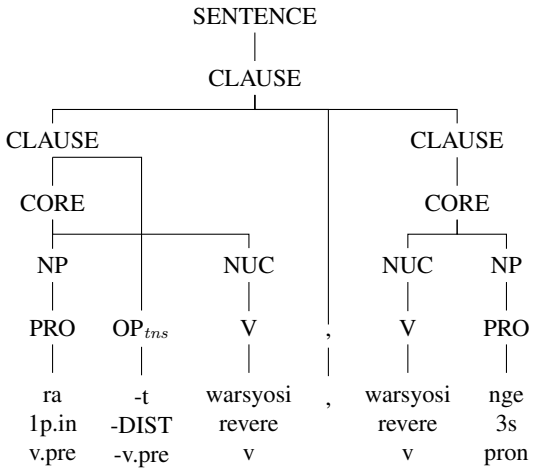

"We revered, revered him."

Figure 1: RRG annotation of a Daakaka sentence with glosses, parts of speech, and translation. Glosses: 1p.in– first person plural inclusive; DIST–distal TAM marker (for past and counterfactual contexts); 3s: third person singular

[1]Our experimental code is available at https://gitlab.com/treegrasp/rrgproj2.

Proceedings of the 29th International Conference on Computational Linguistics, pages 4360–4371
October 12–17, 2022.

(2022).

With respect to complex constructions, i.e., combinations of more than one CLAUSE, CORE or NUC, RRG distinguishes not just coordination and subordination but, in addition, cosubordination. The latter has the general form [[ ]X [ ]X]X, an example is the combination of the two CLAUSEs in Figure 1 into a larger CLAUSE. In such a construction, an operator that applies to categories X takes scope over both X daughters while being realized only once. The TAM marker in Figure 1 for instance assigns tense to both CLAUSEs. The various ways of combining two NUC, CORE or CLAUSE constituents differ with respect to whether one of the two depends on the other, how tight the two units are concerning time and location of the two events, and to what extent the two units share operators such as tense, aspect, modality etc., all of which can be explained by the respective combination of layer level (NUC, CORE or CLAUSE) and construction type (coordination, subordination or cosubordination).

In this paper, we show a way to train RRG parsers on only a little amount of annotated data in the target language or, in the case of Dalkalaen, even no annotated data, yielding parse trees that can presumably reduce annotation effort considerably when used as a starting point for treebanking. Furthermore, the resulting parse trees might be sufficiently good to be the basis for (semi-)automatic investigations of syntactic properties of low-resource languages.

The scenario underlying our work is that we have a limited amount of data (translated to English and possibly glossed), and that a small subset of it (a few hundred or at most a few thousand sentences) is annotated with RRG trees. This is a realistic scenario for the result from typological fieldwork and grammar description. Consequently, and in contrast to most other proposals for parsing low-resource languages, (i) we are not aiming at dependency parsing but, instead, using a constituency scheme often used in typological research, and (ii) we cannot use a large language model trained on the target language, but (iii) we can make use of additional data typically included in fieldwork output, namely glosses and translations.

We propose to improve parsing in this scenario by cross-lingually injecting information from English translations (and glosses, if available) into the target-language parser, combined with self-training. The main research question we address in this paper is to what extent this method improves performance, and in what situations, i.e., at how many annotated trees a language stops being "low resource enough" to be helped by this method. To answer this, we test our methods on various simulated degrees of "low resource-ness", from 100 training trees to over 4 000. To get more robust data, we test our method not only on a real low-resource language (Daakaka), but also on four other languages for which parallel RRG treebanks are available (German, French, Russian, and Farsi). In all cases, we use English as the source language. Finally, we conduct experiments on a language related to Daakaka, namely Dalkalaen, where we have glosses and English translations but no syntactic annotations (except for test data). The aim is to test whether extending a cross-lingual parser (with English as source language) to a related language while keeping the source language leads to useful results. The hypothesis is that parsing a no-resource language (with glosses and English translations, i.e., no-resource concerning syntactic annotations) benefits from knowledge about the English translation and from knowledge learned from a related language.

In the remainder of the paper, we will first discuss related work (Sec. 2), then explain our parsing architecture, including grammar extraction, RRG parsing, cross-lingual transfer, and self-training (Sec. 3). Then, in Sec. 4, we will describe our experimental setup, and Sec. 5 will discuss the results of the experiments. We conclude in Sec. 6.

2 Related Work

The problem of parsing low-resource languages has been addressed both for dependency and for constituency parsing for different scenarios concerning available data on the target side and available high resource parallel data (Zeman and Resnik, 2008; Vania et al., 2019). Many approaches assume that there is enough unlabeled data for the target language to train a language model (i.e., the term 'low resource' refers only to syntactically annotated data). Schuster et al. (2019) for instance use monolingual language models for both source and target language and use a mapping between decontextualized variants of these vectors to guide the cross-lingual transfer. Mulcaire et al. (2019); Kitaev et al. (2019) use polyglot language models trained on source and target data as input to

crosslingual parsing. In contrast to this, we assume a scenario where there is not enough target data to train a language model.

There is also variation concerning the way the information from the source parse is injected into the target parsing process. Many approaches project the source parse onto the target sentence via alignments, sometimes even using multiple source languages (Agić et al., 2016). Instead of projecting a parse of a source sentence to a target sentence, McDonald et al. (2011) use parallel English data to select among the k best parses of a target language sentence by comparing to the parallel English parse. In our case, we use only English parallel source data, and we project supertag information along word alignments.

Some approaches use delexicalized parsers (McDonald et al., 2011; Das et al., 2017), but it has also been shown that lexicalization, in particular when covering aspects shared between source and target language helps (Falenska and Çetinoğlu, 2017). In this vein, we experiment with including glosses in the target language, which means that the target language tokens contain information that is similar to the information captured in the aligned English tokens.

Self-training as a means of training data augmentation for parsing has been proposed in a number of papers (McClosky et al., 2006; Reichart and Rappoport, 2007; Rehbein, 2011; Rotman and Reichart, 2019), though not in the context of the above-mentioned approaches to cross-lingual transfer in low resource parsing.

3 Method

RRG Parsing Following earlier work on RRG parsing (Bladier et al., 2020b), we adopt a formalization of RRG as a Tree Wrapping Grammar (TWG; Kallmeyer et al., 2013), a tree rewriting grammar formalism in the spirit of Tree-Adjoining Grammar (TAG; Joshi and Schabes, 1997). In their parsing architecture, training trees are first decomposed by a rule-based algorithm into TWG elementary trees (supertags) and bilexical dependencies.[2] A neural model is then trained to predict supertag and dependency head probability distributions for each word in a sentence. At test time, an A* parsing algorithm takes these distributions as input and computes the optimal TWG derivation and derived tree. Because RRG trees contain crossing branches, a decrossing step before supertag extraction and a recrossing step between parsing and evaluation on the gold data is required. Figure 2 shows an example.

Note that decrossing is rather local, since crossing branches usually occur within a single group of layers CLAUSE – CORE – NUC (resp. XP – CORE_X – NUC_X), as in Fig. 1 and 2. The decrossing algorithm of Bladier et al. (2020b), which we use, differs from the graph decrossing method proposed by Boyd (2007) in that the tree structures undergo only minimal changes to largely preserve the original tree structure and that it follows handwritten rules to decross the nodes uniformly, e.g. the discontinuous OP_{tns} node under the CLAUSE is always re-attached to the closest lower CORE node. The co-anchoring for PPs without a clear meaning contribution (e.g. *made for the stairs*) and particle verbs (e.g. *pick up*) is simulated by including the corresponding internal structure of the dependent supertag into the head supertag (see Fig. 2). The supertags are extracted with features that indicate the original parent node, which facilitates the rule-based recrossing step after parsing.

For the steps of decrossing, supertag extraction, A* parsing, and recrossing, we use the system developed by (Bladier et al., 2020a,b). Our cross-lingual extension is in the neural supertag and arc scoring module. The idea is illustrated in Figure 3: we train a monolingual system for the source language and then use its internal representations as an additional input to a system for the target language. The representations are fed through a cross-lingual attention mechanism to take into account word alignment information. The resulting cross-lingual system takes a source-language sentence and an aligned target-language sentence as input (both are available in the parallel treebanking scenario) and produces a parse for the latter. We now describe the monolingual scoring module and our cross-lingual extension in detail.

Monolingual Scoring Module The scoring module of Bladier et al. (2020b) takes a sequence of word embeddings $(x_i)_{i=1}^N$ as input, to which a 2-layer BiLSTM transducer is applied to provide contextualized word representations $(h_i)_{i=1}^N$. To these, two additional 2-layer BiLSTMs are applied to obtain supertag-specific ($h^{(sp)}$) and dependency-

[2]Note that the term 'dependencies' is used in a formal sense here, i.e., denoting directed edges between tokens. These edges mark combinations of the respective supertags via (wrapping) substitution or adjunction. They correspond only to a certain extent to dependencies in the linguistic sense.

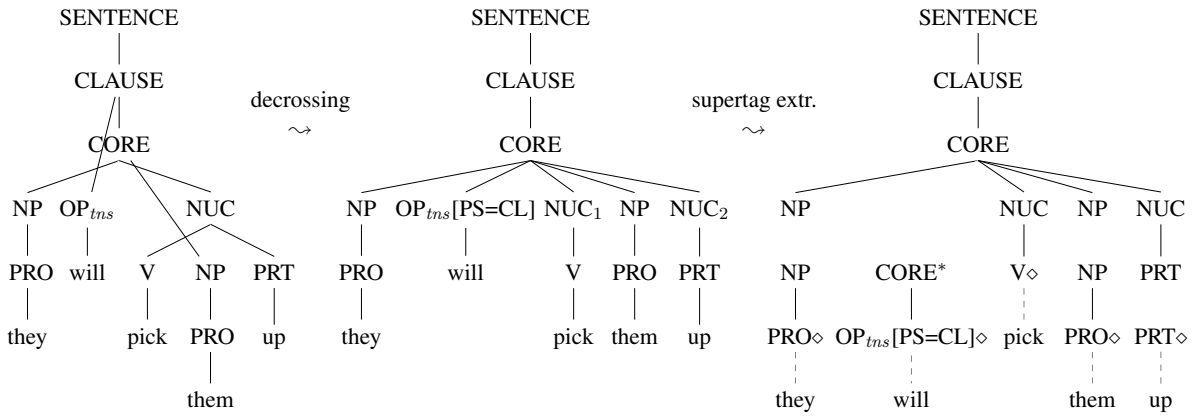

Figure 2: Example for decrossing and subsequent supertag extraction. [PS=CL] indicates that the OP$_{tns}$ node was originally immediately below CLAUSE. ◇ indicates the position of the lexical anchor.

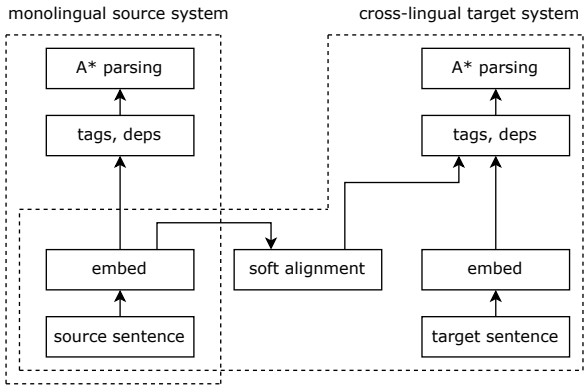

Figure 3: Architecture overview

specific ($h^{(dp)}$) word representations, respectively:

$$(h_1^{(sp)}, \ldots, h_N^{(sp)}) = \texttt{BiLSTM}_s(h_1, \ldots, h_N) \quad (1)$$

$$(h_1^{(dp)}, \ldots, h_N^{(dp)}) = \texttt{BiLSTM}_d(h_1, \ldots, h_N) \quad (2)$$

$(h_1^{(sp)}, \ldots, h_n^{(sp)})$ are used to predict supertags:

$$\Pr(sup(i)) = \texttt{softmax}(\texttt{Linear}_s(h_i^{(sp)})) \quad (3)$$

Finally, the dependency parsing component is based on biaffine scoring (Dozat and Manning, 2017), in which the head and dependent representations are obtained by applying two feed-forward networks to $(h_1^{(dp)}, \ldots, h_N^{(dp)})$, $\texttt{hd}_i = \texttt{FF}_{hd}(h_i^{(dp)})$ and $\texttt{dp}_i = \texttt{FF}_{dp}(h_i^{(dp)})$. The score of word j becoming the head of word i is then defined as follows (M is a matrix and b is a bias vector):

$$\phi(i,j) = \texttt{dp}_i^T M\ \texttt{hd}_j + b^T \texttt{hd}_j \quad (4)$$

Cross-lingual Embeddings via Soft Alignment Our cross-lingual system requires on input (i) a target sentence $(y_i)_{i=1}^M$ of length M, (ii) a corresponding source sentence $(x_i)_{i=1}^N$ of length N, and (iii) a soft word alignment function $a(j,i)$, obtained using standard unsupervised word alignment. $a(j,i)$ provides the probability of aligning the j-th target word with the i-th source word (0 in the source sentence is used for unaligned target words), and it holds that for each $j \in 1..M$ the sum of all $a(j,i)$ ($i \in 0..N$) is 1. The alignment enables an attention mechanism that projects the source supertag/dependency representations via the word alignments onto the corresponding target words. The source-side representations are provided by a monolingual parser trained on a large dataset available for the source language. Formally, let $(h_i^{(src\_sp)})_{i=1}^N$ and $(h_i^{(src\_dp)})_{i=1}^N$ be the source supertag and dependency representations (calculated as in Eq. 1 and Eq. 2). They are projected along the alignment function to obtain the target supertag/dependency projections:

$$h_j^{(prj\_sp)} = \sum\nolimits_i a(j,i) h_i^{(src\_sp)} \quad (5)$$

$$h_j^{(prj\_dp)} = \sum\nolimits_i a(j,i) h_i^{(src\_dp)} \quad (6)$$

These projections are then concatenated with the target language supertag and dependency representations ($h_j^{(trg\_sp)}$ and $h_j^{(trg\_dp)}$, resp.), calculated as in Eq. 1, 2 but without pre-trained embeddings:

$$h_j^{(hbr\_sp)} = [h_j^{(prj\_sp)}; h_j^{(trg\_sp)}] \quad (7)$$

$$h_j^{(hbr\_dp)} = [h_j^{(prj\_dp)}; h_j^{(trg\_dp)}] \quad (8)$$

where for two given vectors v and w, $[v;w]$ denotes their concatenation. The resulting "hybrid" representations, $h_j^{(hbr\_sp)}$ and $h_j^{(hbr\_dp)}$, are from

this point on used as in the monolingual model in order to determine the supertag distribution (Eq. 3) and dependency head scores (Eq. 4).

Gloss Embeddings Language documentation data seldom comes in parsed form, but often in glossed form. For example, in the Daakaka and Dalkalaen data we use (cf. Section 4), each word is annotated with parts of speech as well as morpheme-by-morpheme glosses, where content morphemes are represented by English translations, and function morphemes by acronyms indicating their function (see bottom of Figure 1 for an example). These glosses contain valuable information for parsing, as they put words into morphosyntactic classes and contain specific lexical information as well. We experiment with making this information available to the parser by concatenating the character-based word embeddings with character-based part-of-speech and morpheme-by-morpheme gloss embeddings.

Self-training Self-training is a technique used to improve learning on limited amounts of training data. Applied to parsing, the idea is to train on what little data is available first and use the resulting model to parse unannotated data. Some of the resulting parses are then selected and added to the training data. The expanded training data is used to train a new model. This process can be repeated for multiple "rounds" of self-training (McClosky et al., 2006; Reichart and Rappoport, 2007; Huang and Harper, 2009; Kurniawan et al., 2021). Selection of the added data is crucial; generally the idea is to add those instances where the parser is especially confident and that are thus likely to be correct.

4 Experimental Setup

Data Our choice of data is mainly driven by the availability of RRG-annotated resources. RRGbank (Bladier et al., 2018) contains a subset of the English Wall Street Journal Corpus (Marcus et al., 1993) annotated with RRG trees. RRGparbank (Evang et al., 2021; Bladier et al., 2022) contains George Orwell's novel *1984* in the original English as well as translations to French, German, Russian, and Farsi, along with sentence alignments between English and every other language. The English data as well as parts in other languages are annotated. In addition, we use 6 499 sentences in Daakaka, first published as von Prince (2013a) and described in von Prince (2015), which come with glosses, part-of-speech tags, and English translations, and 1 871 of which have been annotated with RRG trees following the RRGparbank annotation guidelines. Furthermore, we use 3 393 sentences in Dalkalaen, first published as von Prince (2013b), also including glosses and English translations, of which 102 sentences have been annotated with RRG trees.

We use all English trees in RRGbank and RRGparbank to train the English source model. For German, French, Russian, Farsi, Daakaka, and Dalkalaen, we focus on sentences with 25 tokens or less that are 1:1-aligned with an English sentence (for Daakaka and Dalkalaen, this is almost all sentences). Of these, we use 80% for training, 10% for development, and 10% for testing. Note that low resource language corpora are often oral corpora, and they therefore tend to contain shorter sentences. This is the reason why only very few of the Daakaka and Dalkalaen sentences are longer than 25, and this is also why a sentence length limit of 25 simulates this low resource scenario adequately. For Dalkalaen, we use the unannotated sentences for self-training, while the 101 annotated sentences are used for testing. In the case of Daakaka, we ran experiments with and without gloss embeddings (in addition to the token embeddings), while for Dalkalaen we always used gloss and token embeddings.

We randomly downsample the training data according to the degree of "low resource-ness" we are simulating in each experiment.

English Source Model The English source model uses FastText word embeddings (Bojanowski et al., 2017). We tuned its hyperparameters using 80% of the data for training and 10% for validation. We then trained it on the entire English dataset.

Word Alignments For all sentence pairs (a sentence in the target language and the aligned English sentence), we computed a soft word alignment matrix using `efmaral` (Östling, 2015) with default settings (code modified to output matrices). For the experiments using gloss embeddings, we align to the glosses instead of the target-language words.

Target Models After a phase of hyperparameter tuning on the development data (cf. Appendix A), we trained 1) monolingual models and 2) cross-lingual models on the (decrossed and decomposed) training data for all target languages, except for Dalkalaen.

Source	Language	Total # Sentences	Total # Trees	≤ 25 tokens, 1:1-aligned # Sentences	≤ 25 tokens, 1:1-aligned ∅ Length	≤ 25 tokens, 1:1-aligned # Trees	≤ 25 tokens, 1:1-aligned # Supertags
RRGbank	English	49 208	3 765	n/a	n/a	n/a	n/a
RRGparbank	English	6 737	6 737	n/a	n/a	n/a	n/a
	German	6 661	5 822	5 026	13.0	4 482	3 128
	French	7 261	3 243	4 550	12.9	2 339	1 815
	Russian	6 669	6 001	5 638	11.2	5 315	3 293
	Farsi	6 604	1 253	4 726	12.3	1 040	946
von Prince (2013a)	Daakaka	6 499	1 871	6 279	10.3	1 845	1 170
von Prince (2013b)	Dalkalaen	3 393	102	3 272	10.2	101	229

Table 1: Data overview. The versions of RRGbank and RRGparbank used are the 2022-03-17 snapshots.

Self-training We experiment with up to 5 rounds of self-training. In each round, we use the current parsing model to parse all sentences that are not yet part of the training data. We then add the 500 output trees with the lowest weights as reported by the A* parser to the training data. With poorly performing initial models, it sometimes happens that many parse failures occur and less than 500 parses are found in total. In these cases, we add all parses.

For Daakaka and for the RRGparbank languages, self-training starts from a parser trained on a set of annotated trees, as described in the previous section. For Dalkalaen, the starting point was the parser trained on all annotated Daakaka data (train, dev, and test), with glosses.

Evaluation We applied the neural scoring model to the annotated part of the development/test data, then computed RRG trees using the A* parser. Failed parses were replaced by dummy trees consisting of a SENTENCE root that directly dominates POS tags taken from the highest-scoring supertags. The resulting trees were recrossed and compared to the manually annotated dev/test trees using the EVALB metric (Collins, 1997), ignoring function tags such as -PERI or -TNS. We report the F1 score.

5 Results and Discussion

Impact of Training Data Size We are interested in low-resource scenarios, so we first look at how the number of available training trees impacts parser accuracy for different languages. This provides the baseline against which we will later evaluate our cross-lingual embedding and self-training methods. We look at various degrees of "low resource-ness", simulated by randomly downsampling the training set: 100 training sentences, 500, 1000, 2000, 3000, 4000, and finally the scenario where we use as many training trees as possible (varies by language). The blue dots in Fig. 4 show the results. In all cases, we see a considerable improvement in each step up to 2000 training sentences, and after that (for those languages where we have data), results improve only slightly. Note that the f-scores in general are rather low (the best ones around 80%) compared to state of the art constituency parsing. This is due to the fact that, since we are simulating a low resource scenario, we do not use any pretrained word embeddings, even for the languages where these would be available.

Impact of Cross-lingual Embeddings Fig. 4 (orange crosses) also shows the effect of including cross-lingual embeddings in the model. In general, the less training data we have in the target language, the more the cross-lingual information from the aligned English sentences is helpful. Furthermore, in none of the cases, the f-score of the cross-lingual model is below the one of the monolingual model, except for one slight outlier for German at training data size 500. (Note however that on the test data, cross-lingual transfer, with 500 or 1 000 training sentences, is helpful for German, see Table 2.) The only language where cross-lingual embeddings improve parsing performance only very little, even when using only 100 sentences of training data, is Daakaka when using glosses. We think that the reason for this is that the helpful information coming from English words, in particular the implicit information on English supertags, is already to a certain extent provided by the glosses, therefore the cross-lingual model does not contribute a lot of useful additional features.

It is striking that monolingual parsing for German with only little training data (500 or more sentences) is better than for instance for French, even though German has more syntactic variation and therefore more supertags. We suspect that the reason is that French has more ambiguities in high frequency lexical items, in particular ambiguities

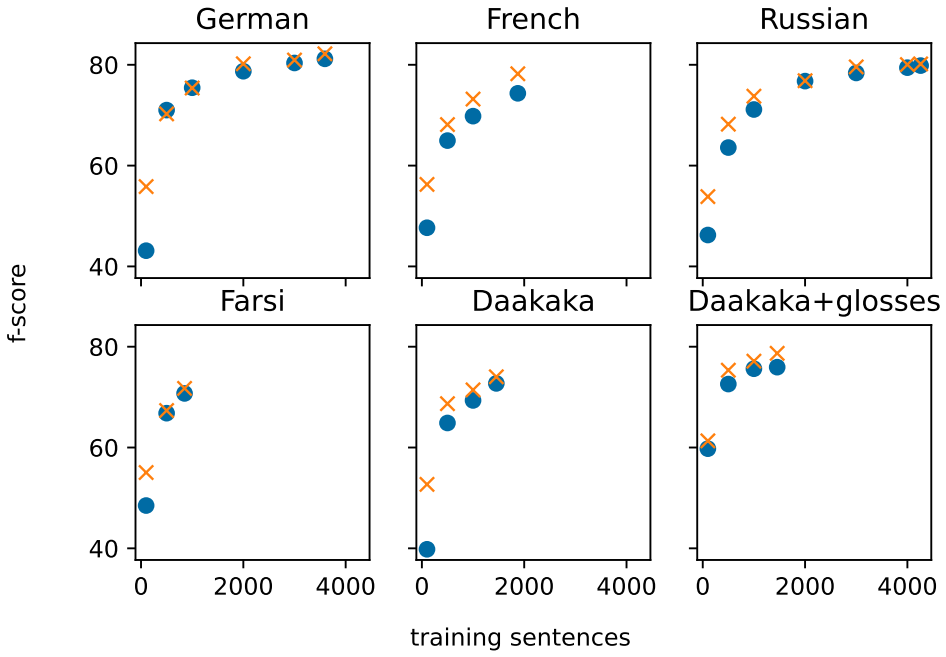

Figure 4: Parser accuracies on the development sets at 100, 500, 1000, 2000, 3000, 4000, and all available training sentences. Dots represent monolingual models, X's represent cross-lingual ones.

that come with different syntactic constructions, for instance the use of *de* and *à* as prepositions ('of' and 'to' respectively in English, 'von' and 'an' in German) or as clause linkage markers preceding an infinitive ('to' in English, 'zu' in German). With more training data and pretrained embeddings as input, this difference is reversed; Bladier et al. (2022) report better parsing results on the RRGparbank data for French and Russian than for German.

Impact of Self-training Now let us look at whether self-training improves the parsing performance, again for monolingual as well as cross-lingual parsing. Figure 5 plots the f-scores for the different languages and for different sizes of training data. Overall, we oftentimes see a positive effect of self-training, and this effect is stronger in the *extremely low* and *very low* resource scenarios (100 and 500 training sentences resp.). The effect is less visible with Daakaka with glosses, but the experiments on the test data (see below) will show that also for this scenario, self-training is actually helpful. It is surprising that, even when starting with only 100 sentences, which means that many constructions in the target language are not present in the training data, self-training improves results, at least for approximately the first two rounds.

In the cases where we start with 1000 training sentences or even with all available training data (note that these have different sizes), the effect of self training is less clear. In some cases, parsing performance even decreases slightly. This means that the trees added during self-training do not contain new knowledge about possible supertag combinations while containing probably errors that decrease parsing performance when used in training. Note however that these numbers are from just one run on the dev data. On the test data, when averaging over several runs, self-training helps for all languages when starting with 1000 training sentences (see below).

Zero-shot parsing with self-training We now investigate the scenario where we want to parse language data that has glosses and translations, but no syntactic annotations at all, not even a seed. We focus on the scenario where we do have access to a seed training set for a related language that also has translations, and glosses following a similar schema. This is also a realistic scenario in linguistic fieldwork, where linguists often investigate multiple related languages within a region. We take the example of Daakaka, for which we have a moderately sized training set, and the closely related language Dalkalaen, for which we have only created a small annotated test set (102 sentences). We apply our cross-lingual model to this scenario by first training our model on all Daakaka data

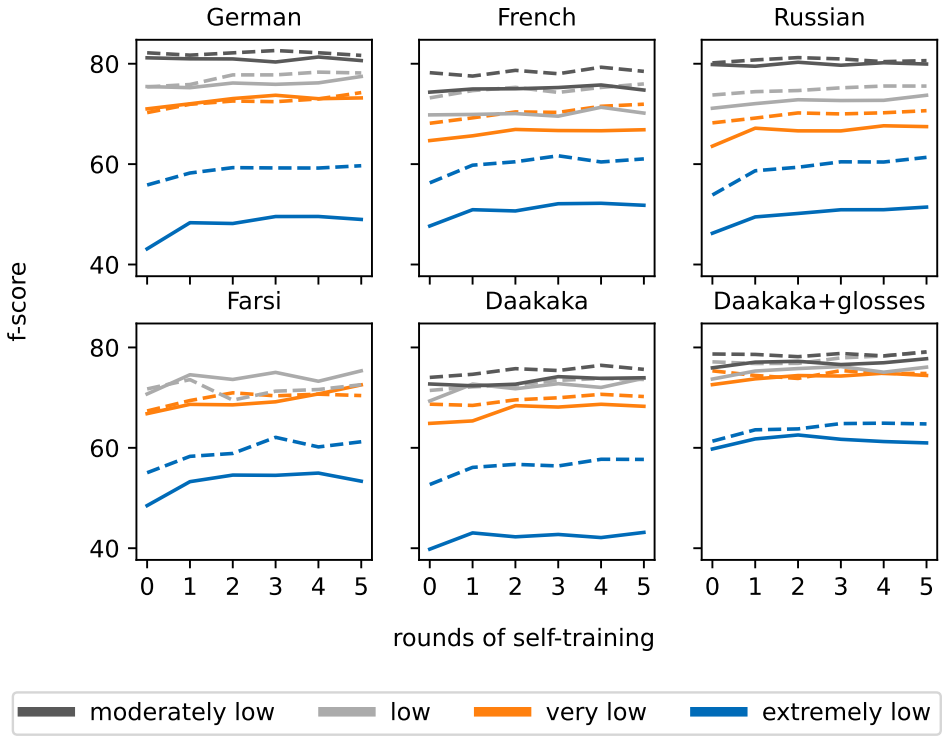

Figure 5: Effect of self-training in the *extremely low* (100 annotated sentences), *very low* (500), *low* (1 000; 851 for Farsi), and *moderately low* (3 594 for German, 1 875 for French, 4 259 for Russian, 1 455 for Daakaka) resource scenarios after 0, 1, 2, 3, 4, and 5 rounds of self-training, adding up to 500 sentences in each round (it can be less due to parse failures). Solid curves represent monolingual and dashed curves cross-lingual models.

(train, dev, and test), and then using the unannotated Dalkalaen data for self-training. The results are shown at the bottom of Table 2. It can be seen that even in this "no-resource" scenario, a performance comparable with the low-resource scenario can be reached.

Test Results After developing our models on the development data, we tested them on the test data. Here, we focus on the very low resource (500 annotated training sentences) and low resource (1000 sentences; 851 for Farsi) scenarios. We run each experiment five times with different random seeds (these affect the initial model parameters and the downsampling of the training data) and give the average f-scores in Table 2; the numbers in bold are the best results for a specific size of training data. The results confirm even more clearly that both cross-lingual transfer as well as self-training improve parsing results. In both cases (500 and 1 000 resp. 851 training sentences) the cross-lingual model systematically outperforms the monolingual model, sometimes by a large margin (see the column with 0 rounds of self-training). Furthermore, for all languages, self-training leads to further improvement, though not always with the same number of self-training rounds. For French in the case of 500 training sentences, after the first three rounds, the maximum score is already reached, and for Farsi and Daakaka with glosses in the very low resource scenario, the scores decrease after 4 rounds of self-training. For the other languages and training data sizes, continuing to 5 rounds brings further slight improvement. In most cases, the best score is reached with the combination of cross-lingual parsing and self-training, except for Farsi, where mono-lingual parsing benefits substantially from self-training while cross-lingual parsing improves only very little.

6 Conclusions

In this paper, we investigated constituency parsing, more precisely RRG parsing, for low-resource languages in a scenario where English translations, a limited set of annotated sentences in the target language and possibly also glosses are available. RRG is a theory that is widely used in typological research. We extended an existing RRG parser into a cross-lingual parser, exploiting the parallel

language	German						French					
self-training	0	1	2	3	4	5	0	1	2	3	4	5
very low, mono	68.9	70.1*	70.9*	71.5*	71.9*	72.2	62.6	64.9*	65.0	66.3*	66.3	66.4
very low, cross	69.0	70.2$^{*}_{\dagger}$	72.0$^{*}_{\dagger}$	73.3$^{*}_{\dagger}$	73.5$^{*}_{\dagger}$	**74.0**$_{\dagger}$	69.9$_{\dagger}$	70.4$_{\dagger}$	71.6$^{*}_{\dagger}$	71.9$_{\dagger}$	72.1$_{\dagger}$	**72.2**$_{\dagger}$
low, mono	74.3	74.7*	75.1	75.8*	75.8	76.0	69.4	70.6*	71.4*	71.5	72.0*	72.0
low, cross	74.7$_{\dagger}$	76.1$^{*}_{\dagger}$	77.5$^{*}_{\dagger}$	77.7$_{\dagger}$	78.2$^{*}_{\dagger}$	**78.3**$_{\dagger}$	74.1$_{\dagger}$	74.4$_{\dagger}$	75.3$_{\dagger}$	**75.9**$^{*}_{\dagger}$	75.7$_{\dagger}$	75.4$_{\dagger}$
language	Russian						Farsi					
self-training	0	1	2	3	4	5	0	1	2	3	4	5
very low, mono	65.8	67.3*	68.0*	68.6	69.2*	69.2	66.7	69.7*	70.2	71.1	**71.8**	71.0
very low, cross	69.9$_{\dagger}$	70.4$^{*}_{\dagger}$	71.4$^{*}_{\dagger}$	71.9$_{\dagger}$	72.0$^{*}_{\dagger}$	**72.2**$_{\dagger}$	69.2$_{\dagger}$	70.4	70.9	71.2	70.5	70.0
low, mono	71.7	73.1*	73.4	73.9*	74.3	74.4	71.3	74.1*	75.1*	74.8	74.9	**75.7**
low, cross	74.3$_{\dagger}$	75.1$^{*}_{\dagger}$	75.3$_{\dagger}$	75.9$^{*}_{\dagger}$	75.9$_{\dagger}$	**76.0**$_{\dagger}$	73.0$_{\dagger}$	73.2	73.4	74.2	73.9	73.9
language	Daakaka						Daakaka+glosses					
self-training	0	1	2	3	4	5	0	1	2	3	4	5
very low, mono	63.0	62.8	64.7*	64.9	65.1	65.1	67.9	69.5*	70.1*	70.7*	70.9	70.5
very low, cross	66.0$_{\dagger}$	67.0$^{*}_{\dagger}$	67.7$_{\dagger}$	68.3$_{\dagger}$	68.4$_{\dagger}$	**68.8**$_{\dagger}$	70.2$_{\dagger}$	70.7$_{\dagger}$	71.7$^{*}_{\dagger}$	72.2$^{*}_{\dagger}$	**72.4**$_{\dagger}$	72.2$_{\dagger}$
low, mono	67.6	68.1*	68.7	68.7	69.2	69.5	71.9	71.5	72.4*	72.9	73.4*	73.3
low, cross	70.4$_{\dagger}$	70.8$_{\dagger}$	70.8$_{\dagger}$	71.2$^{*}_{\dagger}$	71.1$_{\dagger}$	**71.3**$_{\dagger}$	73.1$_{\dagger}$	73.7$^{*}_{\dagger}$	74.3$_{\dagger}$	74.2$_{\dagger}$	74.5$_{\dagger}$	**74.7**$_{\dagger}$
language							Dalkalaen+glosses					
self-training							0	1	2	3	4	5
zero, cross							69.0	71.8*	72.4	73.0*	**73.6**	73.2

Table 2: Test results for the *very low* and *low* resource scenarios for German, French, Russian, Farsi, and Daakaka, as well as the *zero* resource scenario for Dalkalaen. All f-scores are averaged over 5 runs with different random initializations. *indicates that the result is significantly better ($p \leq .05$) than the corresponding result with one less round of self-training according to a permutation test (cf. Dror et al., 2018). $_{\dagger}$indicates the same for cross-lingual results compared to the corresponding monolingual model.

English data and unsupervised word alignments. Furthermore, we also adopted self-training, i.e., iteratively expanding the training data by adding the most confident new parses in each round. Our experiments showed that both self-training and cross-lingual parsing yield substantial and in almost all cases complementary improvements. Further experiments showed that even when only translations and glosses but no annotated sentences are available, self-training starting from a cross-lingual parser for a related language, based also on tokens and glosses, leads to considerable improvements and surprisingly good results.

Acknowledgments

We would like to thank the three anonymous reviewers for their valuable feedback. This work was carried out as a part of the research project TreeGraSP[3] funded by a Consolidator Grant of the European Research Council (ERC).

References

Željko Agić, Anders Johannsen, Barbara Plank, Héctor Martínez Alonso, Natalie Schluter, and Anders Søgaard. 2016. Multilingual projection for parsing truly low-resource languages. *Transactions of the Association for Computational Linguistics*, 4:301–312.

Tatiana Bladier, Kilian Evang, Valeria Generalova, Laura Kallmeyer, Robin Möllemann, Natalia Moors, Rainer Osswald, and Simon Petitjean. 2022. RRGparbank: A parallel role and reference grammar treebank. In *Proceedings of LREC*. To appear.

Tatiana Bladier, Laura Kallmeyer, Rainer Osswald, and Jakub Waszczuk. 2020a. Automatic extraction of tree-wrapping grammars for multiple languages. In *Proceedings of the 19th International Workshop on Treebanks and Linguistic Theories*, pages 55–61, Düsseldorf, Germany. Association for Computational Linguistics.

Tatiana Bladier, Andreas van Cranenburgh, Kilian Evang, Laura Kallmeyer, Robin Möllemann, and Rainer Osswald. 2018. RRGbank: a role and reference grammar corpus of syntactic structures extracted from the Penn Treebank. In *Proceedings of the 17th International Workshop on Treebanks and Linguistic Theories (TLT 2018)*. Linköping Electronic Conference Proceedings.

Tatiana Bladier, Jakub Waszczuk, and Laura Kallmeyer. 2020b. Statistical parsing of tree wrapping grammars. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 6759–6766, Barcelona, Spain (Online). International Committee on Computational Linguistics.

Piotr Bojanowski, Edouard Grave, Armand Joulin, and Tomas Mikolov. 2017. Enriching word vectors with

[3] https://treegrasp.phil.hhu.de

subword information. *Transactions of the Association for Computational Linguistics*, 5:135–146.

Adriane Boyd. 2007. Discontinuity revisited: An improved conversion to context-free representations. In *Proceedings of the Linguistic Annotation Workshop*, pages 41–44, Prague, Czech Republic. Association for Computational Linguistics.

Stephen Clark and James R. Curran. 2007. Wide-coverage efficient statistical parsing with CCG and log-linear models. *Computational Linguistics*, 33(4):493–552.

Michael Collins. 1997. Three generative, lexicalised models for statistical parsing. In *35th Annual Meeting of the Association for Computational Linguistics and 8th Conference of the European Chapter of the Association for Computational Linguistics*, pages 16–23, Madrid, Spain. Association for Computational Linguistics.

Ayan Das, Affan Zaffar, and Sudeshna Sarkar. 2017. Delexicalized transfer parsing for low-resource languages using transformed and combined treebanks. In *Proceedings of the CoNLL 2017 Shared Task: Multilingual Parsing from Raw Text to Universal Dependencies*, pages 182–190, Vancouver, Canada. Association for Computational Linguistics.

Timothy Dozat and Christopher D. Manning. 2017. Deep biaffine attention for neural dependency parsing. In *ICLR*.

Rotem Dror, Gili Baumer, Segev Shlomov, and Roi Reichart. 2018. The hitchhiker's guide to testing statistical significance in natural language processing. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1383–1392, Melbourne, Australia. Association for Computational Linguistics.

Kilian Evang, Tatiana Bladier, Laura Kallmeyer, and Simon Petitjean. 2021. Bootstrapping Role and Reference Grammar treebanks via Universal Dependencies. In *Proceedings of Universal Dependencies Workshop 2021 (UDW 2021)*.

Agnieszka Falenska and Özlem Çetinoğlu. 2017. Lexicalized vs. delexicalized parsing in low-resource scenarios. In *Proceedings of the 15th International Conference on Parsing Technologies*, pages 18–24, Pisa, Italy. Association for Computational Linguistics.

Zhongqiang Huang and Mary Harper. 2009. Self-training PCFG grammars with latent annotations across languages. In *Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing*, pages 832–841, Singapore. Association for Computational Linguistics.

Aravind K Joshi and Yves Schabes. 1997. Tree-adjoining grammars. In *Handbook of formal languages*, pages 69–123. Springer.

Laura Kallmeyer, Rainer Osswald, and Robert D. Van Valin, Jr. 2013. Tree Wrapping for Role and Reference Grammar. In *Formal Grammar 2012/2013*, volume 8036 of *LNCS*, pages 175–190. Springer.

Nikita Kitaev, Steven Cao, and Dan Klein. 2019. Multilingual constituency parsing with self-attention and pre-training. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 3499–3505, Florence, Italy. Association for Computational Linguistics.

Kemal Kurniawan, Lea Frermann, Philip Schulz, and Trevor Cohn. 2021. PPT: Parsimonious parser transfer for unsupervised cross-lingual adaptation. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*, pages 2907–2918, Online. Association for Computational Linguistics.

Mitchell P. Marcus, Beatrice Santorini, and Mary Ann Marcinkiewicz. 1993. Building a large annotated corpus of English: The Penn Treebank. *Computational Linguistics*, 19(2):313–330.

David McClosky, Eugene Charniak, and Mark Johnson. 2006. Effective self-training for parsing. In *Proceedings of the Human Language Technology Conference of the NAACL, Main Conference*, pages 152–159, New York City, USA. Association for Computational Linguistics.

Ryan McDonald, Slav Petrov, and Keith Hall. 2011. Multi-source transfer of delexicalized dependency parsers. In *Proceedings of the 2011 Conference on Empirical Methods in Natural Language Processing*, pages 62–72, Edinburgh, Scotland, UK. Association for Computational Linguistics.

Phoebe Mulcaire, Jungo Kasai, and Noah A. Smith. 2019. Low-resource parsing with crosslingual contextualized representations. In *Proceedings of the 23rd Conference on Computational Natural Language Learning (CoNLL)*, pages 304–315, Hong Kong, China. Association for Computational Linguistics.

Robert Östling. 2015. *Bayesian Models for Multilingual Word Alignment*. Ph.D. thesis, Stockholm University.

Kilu von Prince. 2013a. *Daakaka, The Language Archive*. MPI for Psycholinguistics, Nijmegen.

Kilu von Prince. 2013b. *Dalkalaen, The Language Archive*. MPI for Psycholinguistics, Nijmegen.

Kilu von Prince. 2015. *A Grammar of Daakaka*. Mouton de Gruyter, Berlin, Boston.

Ines Rehbein. 2011. Data point selection for self-training. In *Proceedings of the Second Workshop on Statistical Parsing of Morphologically Rich Languages*, pages 62–67, Dublin, Ireland. Association for Computational Linguistics.

Roi Reichart and Ari Rappoport. 2007. Self-training for enhancement and domain adaptation of statistical parsers trained on small datasets. In *Proceedings of the 45th Annual Meeting of the Association of Computational Linguistics*, pages 616–623, Prague, Czech Republic. Association for Computational Linguistics.

Guy Rotman and Roi Reichart. 2019. Deep contextualized self-training for low resource dependency parsing. *Transactions of the Association for Computational Linguistics*, 7:695–713.

Tal Schuster, Ori Ram, Regina Barzilay, and Amir Globerson. 2019. Cross-lingual alignment of contextual word embeddings, with applications to zero-shot dependency parsing. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 1599–1613, Minneapolis, Minnesota. Association for Computational Linguistics.

Kiyoko Toratani and Carlos González Vergara. 2020. Role and reference grammar. https://rrg.caset.buffalo.edu/rrg/RRGBib_2020.pdf. Accessed: 2021-11-15.

Robert D. Van Valin, Jr. 2005. *Exploring the Syntax-Semantics Interface*. Cambridge University Press.

Robert D. Van Valin, Jr. and William A. Foley. 1980. Role and reference grammar. In E. A. Moravcsik and J. R. Wirth, editors, *Current approaches to syntax*, volume 13 of *Syntax and semantics*, pages 329–352. Academic Press, New York.

Clara Vania, Yova Kementchedjhieva, Anders Søgaard, and Adam Lopez. 2019. A systematic comparison of methods for low-resource dependency parsing on genuinely low-resource languages. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 1105–1116, Hong Kong, China. Association for Computational Linguistics.

Daniel Zeman and Philip Resnik. 2008. Cross-language parser adaptation between related languages. In *Proceedings of the IJCNLP-08 Workshop on NLP for Less Privileged Languages*.

A Hyperparamters

The hyperparameters of our models are given in Table 3.

	mono	cross
character-level LSTM embedding layers		
character size	25	25
depth	1	1
output size for words	300	300
output size: glosses	100 (0)	100 (0)
output size: POS	100 (0)	100 (0)
output dropout	0.1	0.1
common contextualization layer		
input size	500 (300)	500 (300)
output size	200	200
depth	2	2
dropout	0.1	0.1
output dropout	0.1	0.1
contextualization layer for taggers		
input size	400	800
output size	200	200
depth	2	2
dropout	0.1	0.1
output dropout	0.1	0.1
contextualization layer for biaffine layer		
input size	400	800
output size	200	200
depth	2	2
dropout	0.1	0.1
output dropout	0.1	0.1
auxiliary POS tagging layer		
input size	400	400
supertagging layer		
input size	400	400
biaffine dependency parsing layer		
input size	400	400
hidden size	100	100
output size	100	100
dropout	0.1	0.1
A* decoder (partage-twg)		
top-n tags given	15	15
top-n dep. heads	15	15
β probability cutoff factor (Clark and Curran, 2007)	0.01	0.01

Table 3: Hyperparameters in monolingual and cross-lingual models. Numbers in parentheses apply to models with no gloss embeddings. Contextualization layers for the cross-lingual model have twice as big inputs because they concatenate source-language and target-language representations.

A Simple and Effective Method to Improve Zero-Shot Cross-Lingual Transfer Learning

Kunbo Ding[1] [*], **Weijie Liu**[1,2] [†], **Yuejian Fang**[1], **Weiquan Mao**[2], **Zhe Zhao**[2]
Tao Zhu[2], **Haoyan Liu**[2], **Rong Tian**[2], **Yiren Chen**[2]
[1]Peking University, Beijing, China [2]Tencent Research, Beijing, China
kunbo_ding@stu.pku.edu.cn, dataliu@pku.edu.cn, fangyj@ss.pku.edu.cn
{weiquanmao, nlpzhezhao, mardozhu, haoyanliu, rometian, yirenchen}@tencent.com

Abstract

Existing zero-shot cross-lingual transfer methods rely on parallel corpora or bilingual dictionaries, which are expensive and impractical for low-resource languages. To disengage from these dependencies, researchers have explored training multilingual models on English-only resources and transferring them to low-resource languages. However, its effect is limited by the gap between embedding clusters of different languages. To address this issue, we propose Embedding-Push, Attention-Pull, and Robust targets to transfer English embeddings to virtual multilingual embeddings without semantic loss, thereby improving cross-lingual transferability. Experimental results on mBERT and XLM-R demonstrate that our method significantly outperforms previous works on the zero-shot cross-lingual text classification task and can obtain a better multilingual alignment.

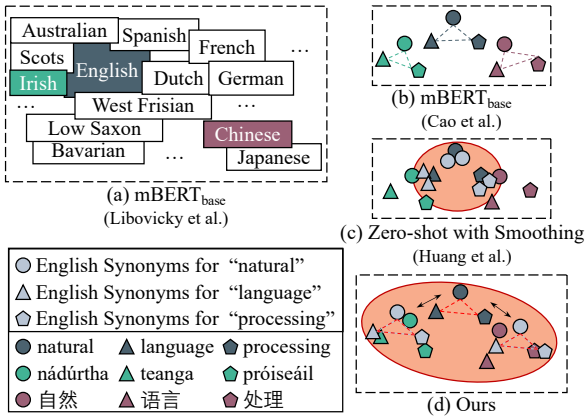

Figure 1: (a) Different languages clusters in mBERT. (b) The relative positions of "nature", "language" and "processing" are similar in English, Chinese and Irish (Cao et al., 2020). (c) Using synonym augmentation to train a robust region covering words in other languages. (d) We align different languages and construct a suitable robust region by pushing the embeddings away and pulling the relative distance among words.

1 Introduction

In recent years, advances in multilingual models such as mBERT (Devlin et al., 2019), XLM (Conneau and Lample, 2019), XLM-R (Conneau et al., 2020), etc., after being fine-tuned with annotated data, have enabled significant improvements in many cross-lingual tasks. However, due to the lack of annotated data, some tasks in low-resource languages have not enjoyed this technological advancement. To solve this issue, the academic and industrial community began to focus on zero-shot cross-lingual transfer learning (Huang et al., 2019; Artetxe et al., 2020), which aims to fine-tune multilingual models with annotated data in high-resource languages and obtain a nice performance in low-resource language tasks.

Some works aligned word embeddings between high- and low-resource languages through additional parallel sentence pairs (Artetxe and Schwenk, 2019; Wei et al., 2021; Chi et al., 2021; Pan et al., 2021) or bilingual dictionaries (Cao et al., 2020; Qin et al., 2020; Liu et al., 2020), so that high-resource fine-tuned models can be transferred to low-resource languages. Although this approach has achieved excellent results in many languages, parallel corpora and bilingual dictionaries are still prohibitively expensive, rendering it impracticable in some minority languages.

To disengage from the dependence on parallel corpora or bilingual dictionaries (Wu and Dredze, 2019; Hu et al., 2020), some studies have found that syntactic features in high-resource languages can improve zero-shot cross-lingual transfer learning (Meng et al., 2019; Subburathinam et al., 2019; Ahmad et al., 2021a,b). Libovický et al. (2020) found that the embeddings of different languages are clustered according to their language families, as shown in Figure 1a and 1b, which demonstrated that different languages are not aligned perfectly

[*] Contribution during internship at Tencent Inc.
[†] Corresponding author: Weijie Liu.

Proceedings of the 29th International Conference on Computational Linguistics, pages 4372–4380
October 12–17, 2022.

in mBERT (Deshpande et al., 2021). Huang et al. (2021) tried adversarial training and randomized smoothing with English synonym augmentation to build robust regions for embeddings in the multilingual models, as illustrated in Figure 1c. In this way, models can output similar predictions for different language embeddings in the same robust region even they are not well aligned. However, the transferability of English synonym augmentation is limited because its robust region remains close to the English cluster, as shown in Figure 1c.

In this work, we select English as a high-resource language and follow the studies that do not require additional parallel corpora or bilingual dictionaries to improve cross-lingual transfer learning performance with minimal cost. For this purpose, three strategies are proposed to enlarge the robust region of English embeddings. The first strategy is called *Embedding-Push*, which pushes the embedding of English to other language clusters. The second is *Attention-Pull*, which constrains the relative position of the word embeddings to prevent the meaning from straying. The last strategy, named *Robust target*, introduces a Virtual Multilingual Embedding (VME) to help the model build a suitable robust region, as shown in Figure 1d.

Experimental results on mBERT and XLM-R demonstrate that our method effectively improves the zero-shot cross-lingual transfer on classification tasks and outperforms a series of previous works. In addition, case studies show that our method improves the model through multilingual word alignment. Compared with existing works, our method has the following advantages. First, our method only needs English resources, which is suitable for low-resource languages. Second, our method can induce alignments in many languages without specifying the target language. Finally, our method is simple to implement and achieves effective experimental results. Our code is publicly available[1].

2 Method

Given an English training batch $\mathcal{B}$, for a specific $\boldsymbol{x} \in \mathcal{B}$ consisting of words (x_1, x_2, x_3), we first follow Huang et al. (2021) to generate an augmented example $\boldsymbol{x^a} = (x_1^a, x_2^a, x_3^a)$ by randomly replacing x_i with x_i^a from the pre-defined English synonym set (Alzantot et al., 2018). Then, we introduce three objective functions to get the Virtual Multilingual Embedding (VME) that provides a suitable robust region for zero-shot cross-lingual classification task as shown in Figure 2. We describe the details in the following subsections.

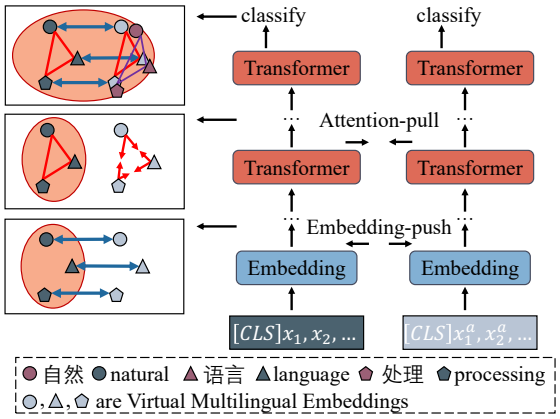

Figure 2: The two networks have tied weights. VMEs expand robust regions (orange circle) by aligning semantic-similar words in other languages. Note that VMEs do not specify the target language but improve multilingual performance, as shown in section 3.3.

2.1 Embedding-push target

The Embedding-Push target aims to make English embeddings leave their original cluster and robust region by pushing away $(\boldsymbol{x}, \boldsymbol{x^a})$ in the embedding space. The pushed embedding can be viewed as the VME. The loss function is (1).

$$\ell_{EPT} = -\frac{1}{|\mathcal{B}|}\sum_{\boldsymbol{x}\in\mathcal{B}}\left(M(E_{\boldsymbol{x}}) - M(E_{\boldsymbol{x^a}})\right)^2 \quad (1)$$

where $E_{\boldsymbol{x}}, E_{\boldsymbol{x^a}}$ denote the embedding output of $\boldsymbol{x}$ and $\boldsymbol{x^a}$, M is the mean-pooling method.

2.2 Attention-pull target

The self-attention matrices contain rich linguistic information (Clark et al., 2019) and can be regarded as a 1-hop graph attention between the hidden states of words (Vaswani et al., 2017; Veličković et al., 2018). The attention matrix represents the information transfer score between each pair of words, we regard it as the pulling force, so the attention matrix determines the relative linguistic positions of words in a sentence. We introduce the Attention-Pull target to encourage the relative linguistic position among (x_1^a, x_2^a, x_3^a) to be similar to (x_1, x_2, x_3) by fitting the middle layer multi-head attention matrices, as (2).

$$\ell_{APT} = \frac{1}{|\mathcal{B}|H}\sum_{\boldsymbol{x}\in\mathcal{B}}\sum_{i}^{H}\left(A_{\boldsymbol{x}}^i - A_{\boldsymbol{x^a}}^i\right)^2 \quad (2)$$

[1]https://github.com/KB-Ding/EAR

Model	en	ar	bg	de	el	es	fr	hi	ru	sw	th	tr	ur	vi	zh	avg.
mBERT†	80.8	64.3	68.0	70.0	65.3	73.5	73.4	58.9	67.8	49.7	54.1	60.9	57.2	69.3	67.8	65.4
+ADV†	81.9	64.9	68.3	71.7	66.5	74.4	74.5	59.6	68.8	48.8	50.6	61.7	59.2	70.0	69.4	66.0
+RS-RP†	82.6	65.4	68.7	70.5	67.2	75.0	74.1	59.8	69.5	48.4	50.5	59.7	57.9	70.5	69.7	66.0
+RS-DA†	81.0	66.4	69.9	71.8	68.0	74.7	74.2	62.7	70.6	51.1	55.7	62.9	60.9	71.8	71.4	67.6
+Syntax‡	81.6	65.4	69.3	70.7	66.5	74.1	73.2	60.5	68.8	-	-	62.4	58.7	69.9	69.3	-
+ **Ours**	**83.2**	**67.4**	**71.0**	**72.9**	**68.3**	**75.7**	**75.2**	**64.0**	**71.6**	**51.3**	**56.7**	**63.6**	**61.4**	**72.4**	**71.5**	**68.4**
XLM-R*	84.0	72.6	**78.9**	77.0	76.5	78.6	78.2	70.3	76.4	65.0	72.4	73.4	67.6	75.5	75.0	74.8
+RS-DA*	83.5	73.2	78.2	77.1	76.9	79.2	79.0	72.3	**76.9**	66.5	73.2	73.1	68.2	**76.4**	75.1	75.3
+ **Ours**	**84.6**	**74.5**	78.8	**77.5**	**77.0**	**79.4**	**79.5**	**72.6**	76.8	**66.7**	**73.9**	**74.7**	**68.7**	**76.4**	**75.8**	**75.8**

Table 1: Zero-shot cross-lingual transfer results on the XNLI. We bold the highest accuracy scores (%). "†" and "‡" are taken from (Huang et al., 2021) and (Ahmad et al., 2021a), respectively. "∗" is the result of our reimplementation.

Model	en	de	es	fr	ja	ko	zh	avg.
mBERT†	94.0	85.7	87.4	87.0	73.0	69.6	77.0	82.0
+ADV†	93.7	86.5	88.5	87.8	76.1	75.3	80.4	84.0
+RS-RP†	**94.5**	87.4	90.0	89.5	77.9	77.5	82.0	85.5
+RS-DA†	93.5	87.8	88.8	88.8	79.3	78.3	81.5	85.4
+Syntax‡	94.0	85.9	89.1	88.2	75.8	76.3	80.7	84.3
+Ours	94.2	**87.9**	**90.3**	**89.7**	**79.9**	**79.2**	**82.4**	**86.2**
XLM-R*	94.4	88.9	89.8	89.2	78.2	78.4	81.4	85.7
+RS-DA*	94.7	88.8	89.7	90.0	78.7	80.2	82.3	86.3
+Ours	**95.1**	**89.0**	**90.3**	**90.1**	**80.5**	**81.7**	**83.1**	**87.1**

Table 2: Experimental results on the PAWS-X across 7 languages. "†" and "‡" are taken from (Huang et al., 2021) and (Ahmad et al., 2021a), respectively. "∗" is the result of our reimplementation.

where H is the number of attention head. Let L denote the sequence length, $A^i \in \mathbb{R}^{L\times L}$ is the attention matrix corresponding to the i-th head. ℓ_{APT} alleviates the semantic loss of the VME.

2.3 Robust target

The robust target aims to build a robust region with the VME for the classification task. The hidden state of [CLS] in the last layer is taken to classify, as (3). The model is trained by (4).

$$P_n = \text{softmax}(\boldsymbol{W} h_n^{[\text{CLS}]} + \boldsymbol{b}) \quad (3)$$

$$\ell_{CE} = -\frac{1}{|\mathcal{B}|}\sum_{\boldsymbol{x}\in\mathcal{B}} (y \log P_{\boldsymbol{x}} + y \log P_{\boldsymbol{x}^a}) \quad (4)$$

where $\boldsymbol{W}$ and $\boldsymbol{b}$ are trainable parameters. P_n is the prediction for n. y denotes the gold label for each $\boldsymbol{x} \in \mathcal{B}$. The final training objective is to minimize three targets as (5):

$$\ell = \ell_{CE} + \alpha\ell_{EPT} + \beta\ell_{APT} \quad (5)$$

where α and β are hyperparameters.

3 Experiment

3.1 Dataset and setup

We use $\text{mBERT}_{\text{base}}$ and $\text{XLM-R}_{\text{base}}$ to evaluate our method on XNLI (Conneau et al., 2018) and PAWS-X (Yang et al., 2019) tasks, covering 17 languages. We consider English as the source language and other languages in test sets as low-resource target languages. More training details are in Appendix A. We set α=1, β=0.1 and apply the Attention-Pull target at the 6-th layer. The analysis of hyperparameters is in Appendix B. We measure results with accuracy.

3.2 Baseline methods

For XLM-R, we consider **RS-DA** as a strong baseline because it achieves the best performance. For mBERT, we consider all the following baselines.

Adv: Huang et al. (2021) uses adversarial training to build a robust region for cross-lingual transfer. They consider the most effective perturbation in each iteration.

RS-RP: Huang et al. (2021) perturbs sentence embeddings with randomly sampled δ to smooth the classifier and build robust regions.

RS-DA: Huang et al. (2021) augments training data with English synonym replacement to train a smooth classifier and build robust regions.

Syntax: Ahmad et al. (2021a) provides syntax features to mBERT by graph attention networks, which helps cross-lingual transfer.

3.3 Main results

As illustrated in Table 1 and Table 2. We can observe that: 1) Our method achieves up to 4.2% and 1.4% improvement on mBERT and XLM-R, respectively, outperforming existing works and demonstrating the effectiveness of our method. 2) Multiple low-resource languages benefit from our method. Based on mBERT, our method improves not only English-like languages such as **es** and **de** but also English-dissimilar (Littell et al., 2017) languages such as **tr** and **ko**. This result indicates that the VME we proposed helps align different languages in semantic space. 3) We avoid training

Model	en	ar	bg	de	el	es	fr	hi
Ours	83.2	67.4	71.0	72.9	68.3	75.7	75.2	64.0
w/o EPT	82.8	67.0	71.2	72.7	67.6	75.5	75.1	63.4
w/o APT	82.4	66.5	70.8	72.8	68.5	76.0	75.1	63.4
w/o both	82.1	66.4	70.0	72.3	67.7	75.1	74.9	62.8
Model	**ru**	**sw**	**th**	**tr**	**ur**	**vi**	**zh**	**avg.**
Ours	71.6	51.3	56.7	63.6	61.4	72.4	71.5	**68.4**
w/o EPT	71.2	51.1	56.0	63.4	60.7	72.5	71.4	68.1
w/o APT	70.9	50.0	57.0	62.8	61.9	72.0	72.3	68.2
w/o both	70.8	48.3	54.6	61.0	61.0	71.5	71.5	67.4

Table 3: Ablation experimental results of our method on the XNLI task. Experiments are based on mBERT.

each target language separately and achieves the best results in one epoch using the English-trained VME.

3.4 Ablation study

As shown in Table 3, we perform ablation studies on Embedding-Push Target (**EPT**) and Attention-Pull Target (**APT**). We find that both EPT and APT are effective, but they can not perform well alone. Besides, removing the APT causes improvement in some languages, such as **zh** and **ur**. We attribute this to the fact that the EPT-guided VME is unstable without the APT, which improves performance in some languages but drops in more languages such as **en**, **ar**, **ru**, etc., resulting in poor average performance. Thus EPT and APT need to be combined for better performance.

4 Analysis

4.1 Case study

To study the effects of VME, we do the T-SNE visualization for the word embeddings of parallel sentences, as shown in Figure 3. Compared with the RS-DA, our fine-tuned model aligns better across languages, and words are closer to their translations, leading to correct predictions. This observation shows that the VME can effectively help cross-lingual word alignment and improve the performance of the model. We choose Arabic for the case study because it can represent a class of languages far apart from English.

4.2 Effect of EPT

To study the impact of EPT, we do the T-SNE visualization using the embedding layer of mBERT. As shown in Figure 4, some synonyms such as "coupled / pair" and "energy / electricity" are pushed away in the embedding layer trained with EPT, and some synonyms are still close to their original words. It indicates that the EPT push away

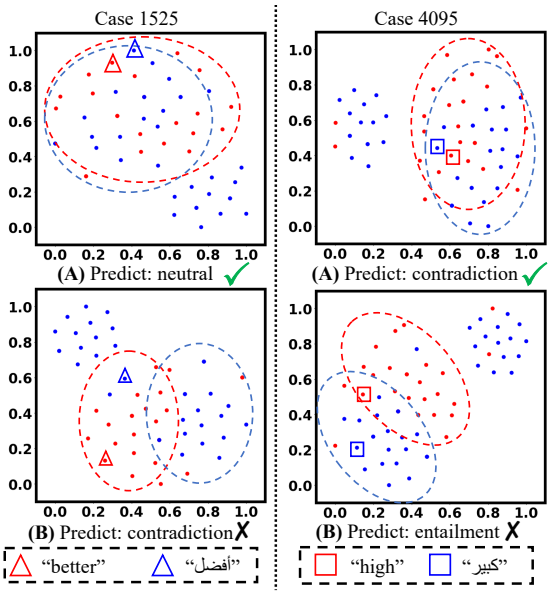

Figure 3: T-SNE visualization for word embeddings of English and Arabic translated sentences in XNLI test sets. Blue dots are Arabic words. Red dots are English words. (A) mBERT trained with our method. (B) mBERT trained with RS-DA.

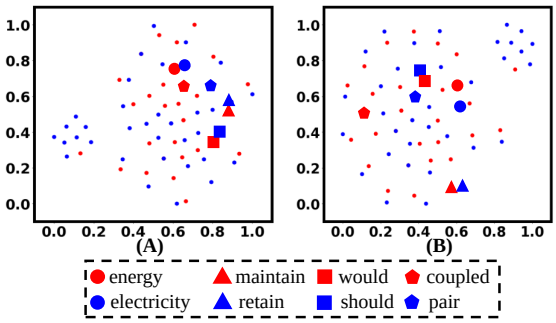

Figure 4: Visualization for English synonyms in the XNLI dataset using the embedding layer of mBERT. (A) Untrained. (B) Trained with our method.

Model	en	es	de	fr	bg	ru	el	th
EPT + APT	83.2	75.7	72.9	75.2	71.0	71.6	68.3	56.7
NT + APT	82.7	75.6	72.5	75.2	70.6	71.0	67.9	55.9
EPT + SRPT	83.1	76.0	73.2	74.9	70.7	71.4	68.9	56.5
Model	**sw**	**vi**	**ar**	**zh**	**hi**	**ur**	**tr**	**avg.**
EPT + APT	51.3	72.4	67.4	71.5	64.0	61.4	63.6	**68.4**
NT + APT	50.6	72.3	66.9	71.8	63.4	61.0	62.9	68.0
EPT + SRPT	50.5	72.3	67.0	71.8	63.3	60.9	63.1	68.2

Table 4: Results on the XNLI task when replacing some targets, based on the mBERT. We sort languages according to their differences from English (Littell et al., 2017), from top left (small) to bottom right (big).

synonyms selectively. We also try to replace the EPT in (5) with the Noise Target (NT), which perturbs word embeddings with Gaussian noise (Cohen et al., 2019). As shown in Table 4, we find

Source Language	en	ar	bg	de	el	es	fr	hi	ru	sw	th	tr	ur	vi	zh	avg.
en	**83.2**	67.4	71.0	72.9	68.3	75.7	**75.2**	64.0	71.6	**51.3**	56.7	63.6	61.4	72.4	71.5	68.4
de	79.6	**68.7**	71.9	**77.7**	**68.8**	**76.2**	74.9	64.2	72.4	50.1	55.2	**64.0**	**62.8**	73.0	72.6	**68.8**
ru	78.5	68.2	**73.3**	73.1	**68.8**	74.8	73.9	**65.8**	**75.7**	49.3	**57.2**	**64.0**	62.4	**73.4**	**73.7**	**68.8**

Table 5: Results of our method on the XNLI task when training mBERT with three source languages.

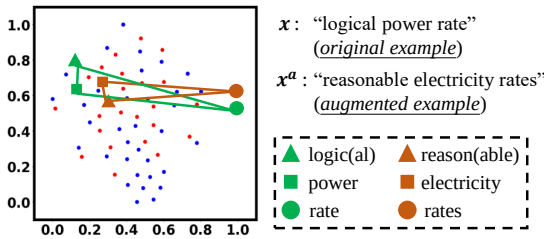

Figure 5: T-SNE visualization on the outputs of the mBERT trained with our method. The original words (x) and synonyms (x^a) are from the XNLI training sets.

scale	size of dictionary	XNLI result
1.0	49975	**68.424**
0.75	37481	**68.392**
0.5	24987	**68.218**
0.25	12493	**68.080**

Table 6: Results on the XNLI task when using the scaled English synonym dictionaries for data augmentation.

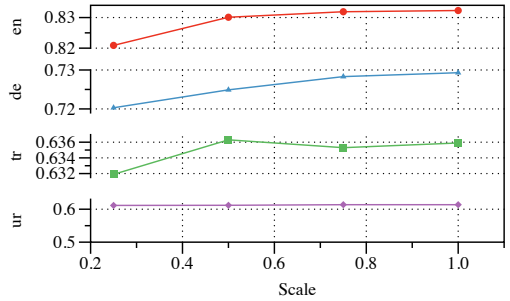

Figure 6: Results on XNLI test sets of four languages when using scaled synonym dictionaries in our method.

that the EPT setting outperforms NT. One possible explanation could be that the noise in NT affects all English tokens and thus may hurt performance.

4.3 Effect of APT

To investigate the effects of APT, we replace the APT in (5) with the Sentence Representation Pull Target (SRPT). SRPT uses the mean squared error between sentence embeddings of $\boldsymbol{x}$ and $\boldsymbol{x^a}$ as the objective. Formally, $\ell_{SRPT} = \frac{1}{|\mathcal{B}|}\sum_i^{|\mathcal{B}|}(\text{Sent}(\boldsymbol{x}) - \text{Sent}(\boldsymbol{x^a}))^2$, where Sent(x) represents the mean-pooled sentence embeddings (Reimers and Gurevych, 2019) obtained by the middle layer of the model. Results in Table 4 show that: 1) The average performance of SPRT is lower than that of APT. 2) The SRPT mainly improves performance on English-like languages, such as **es**, **de**, and **el**, while drops that of most English-dissimilar languages, such as **tr**, **hi**, **sw**, **ur**, etc. This phenomenon shows that SRPT suffers heavily from English training resources, biasing the VME towards English-like languages, which hurts the overall zero-shot cross-lingual transferability.

We perform T-SNE visualization on the outputs of the mBERT trained with our method. As shown in Figure 5, the synonym is still in the same relative position as the original word, which proves the effectiveness of APT.

4.4 Effect of source language

In addition to **en**, both **de** and **ru** show preference as source languages in cross-lingual learning (Turc et al., 2021). We translate the training set into **de** and **ru** using OPUS-MT (Tiedemann and Thottingal, 2020) models, as shown in Table 5, the performance of our method can be further improved.

4.5 Effect of dictionary size

The data augmentation in our method relies on the size of pre-defined synonym dictionary. As shown in Table 6 and Figure 6, we can observe that: 1) The overall performance decreases as the dictionary size decreases. 2) Some languages are not sensitive to the dictionary size, such as **tr** and **ur**. 3) The performance of **en**, **de**, and **tr** degrades significantly when the dictionary size is scaled from 0.5 to 0.25. This phenomenon may be related to some important synonyms in the dictionary, which are effective for cross-lingual transfer learning.

5 Conclusion

To get rid of the dependence on parallel corpora, enable cross-lingual transfer to low-resource languages, we propose Embedding-Push, Attention-Pull, and Robust targets to combat the influence of language clusters in multilingual models. Experimental results demonstrate that our method outperforms previous works and obtains better-aligned embeddings when trained with only English.

References

Wasi Ahmad, Haoran Li, Kai-Wei Chang, and Yashar Mehdad. 2021a. Syntax-augmented multilingual BERT for cross-lingual transfer. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 4538–4554.

Wasi Uddin Ahmad, Nanyun Peng, and Kai-Wei Chang. 2021b. Gate: Graph attention transformer encoder for cross-lingual relation and event extraction. *Proceedings of the AAAI Conference on Artificial Intelligence*, 35(14):12462–12470.

Moustafa Alzantot, Yash Sharma, Ahmed Elgohary, Bo-Jhang Ho, Mani Srivastava, and Kai-Wei Chang. 2018. Generating natural language adversarial examples. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 2890–2896.

Mikel Artetxe, Sebastian Ruder, and Dani Yogatama. 2020. On the cross-lingual transferability of monolingual representations. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 4623–4637.

Mikel Artetxe and Holger Schwenk. 2019. Massively multilingual sentence embeddings for zero-shot cross-lingual transfer and beyond. *Transactions of the Association for Computational Linguistics (TACL)*, 7:597–610.

Steven Cao, Nikita Kitaev, and Dan Klein. 2020. Multilingual alignment of contextual word representations. In *International Conference on Learning Representations*.

Fredrik Carlsson, Amaru Cuba Gyllensten, Evangelia Gogoulou, Erik Ylipää Hellqvist, and Magnus Sahlgren. 2021. Semantic re-tuning with contrastive tension. In *International Conference on Learning Representations*.

Zewen Chi, Li Dong, Furu Wei, Nan Yang, Saksham Singhal, Wenhui Wang, Xia Song, Xian-Ling Mao, Heyan Huang, and Ming Zhou. 2021. InfoXLM: An information-theoretic framework for cross-lingual language model pre-training. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 3576–3588.

Kevin Clark, Urvashi Khandelwal, Omer Levy, and Christopher D. Manning. 2019. What does BERT look at? an analysis of BERT's attention. In *Proceedings of the 2019 ACL Workshop BlackboxNLP: Analyzing and Interpreting Neural Networks for NLP*, pages 276–286.

Jeremy Cohen, Elan Rosenfeld, and Zico Kolter. 2019. Certified adversarial robustness via randomized smoothing. In *Proceedings of the 36th International Conference on Machine Learning*, pages 1310–1320. PMLR.

Alexis Conneau, Kartikay Khandelwal, Naman Goyal, Vishrav Chaudhary, Guillaume Wenzek, Francisco Guzmán, Edouard Grave, Myle Ott, Luke Zettlemoyer, and Veselin Stoyanov. 2020. Unsupervised cross-lingual representation learning at scale. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics (ACL)*, pages 8440–8451.

Alexis Conneau and Guillaume Lample. 2019. Cross-lingual language model pretraining. In *Advances in Neural Information Processing Systems (NeurIPS)*.

Alexis Conneau, Ruty Rinott, Guillaume Lample, Adina Williams, Samuel Bowman, Holger Schwenk, and Veselin Stoyanov. 2018. XNLI: Evaluating cross-lingual sentence representations. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 2475–2485.

Ameet Deshpande, Partha Talukdar, and Karthik Narasimhan. 2021. When is BERT multilingual? isolating crucial ingredients for cross-lingual transfer. *CoRR*, abs/2110.14782.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics (NAACL)*, pages 4171–4186.

Junjie Hu, Sebastian Ruder, Aditya Siddhant, Graham Neubig, Orhan Firat, and Melvin Johnson. 2020. XTREME: A massively multilingual multi-task benchmark for evaluating cross-lingual generalisation. In *Proceedings of the 37th International Conference on Machine Learning*, volume 119, pages 4411–4421.

Haoyang Huang, Yaobo Liang, Nan Duan, Ming Gong, Linjun Shou, Daxin Jiang, and Ming Zhou. 2019. Unicoder: A universal language encoder by pre-training with multiple cross-lingual tasks. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 2485–2494.

Kuan-Hao Huang, Wasi Ahmad, Nanyun Peng, and Kai-Wei Chang. 2021. Improving zero-shot cross-lingual transfer learning via robust training. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 1684–1697.

Jindřich Libovický, Rudolf Rosa, and Alexander Fraser. 2020. On the language neutrality of pre-trained multilingual representations. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 1663–1674.

Patrick Littell, David R. Mortensen, Ke Lin, Katherine Kairis, Carlisle Turner, and Lori Levin. 2017. URIEL and lang2vec: Representing languages as typological,

geographical, and phylogenetic vectors. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 2, Short Papers*, pages 8–14.

Zihan Liu, Genta Indra Winata, Zhaojiang Lin, Peng Xu, and Pascale Fung. 2020. Attention-informed mixed-language training for zero-shot cross-lingual task-oriented dialogue systems. *Proceedings of the AAAI Conference on Artificial Intelligence*, 34(05):8433–8440.

Tao Meng, Nanyun Peng, and Kai-Wei Chang. 2019. Target language-aware constrained inference for cross-lingual dependency parsing. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 1117–1128.

Amil Merchant, Elahe Rahimtoroghi, Ellie Pavlick, and Ian Tenney. 2020. What happens to BERT embeddings during fine-tuning? In *Proceedings of the Third BlackboxNLP Workshop on Analyzing and Interpreting Neural Networks for NLP*, pages 33–44.

Lin Pan, Chung-Wei Hang, Haode Qi, Abhishek Shah, Saloni Potdar, and Mo Yu. 2021. Multilingual BERT post-pretraining alignment. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 210–219. Association for Computational Linguistics.

Libo Qin, Minheng Ni, Yue Zhang, and Wanxiang Che. 2020. Cosda-ml: Multi-lingual code-switching data augmentation for zero-shot cross-lingual nlp. In *Proceedings of the Twenty-Ninth International Joint Conference on Artificial Intelligence, IJCAI-20*, pages 3853–3860.

Nils Reimers and Iryna Gurevych. 2019. Sentence-BERT: Sentence embeddings using Siamese BERT-networks. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 3982–3992.

Ananya Subburathinam, Di Lu, Heng Ji, Jonathan May, Shih-Fu Chang, Avirup Sil, and Clare Voss. 2019. Cross-lingual structure transfer for relation and event extraction. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 313–325. Association for Computational Linguistics.

Jörg Tiedemann and Santhosh Thottingal. 2020. OPUS-MT – building open translation services for the world. In *Proceedings of the 22nd Annual Conference of the European Association for Machine Translation*, pages 479–480. European Association for Machine Translation.

Iulia Turc, Kenton Lee, Jacob Eisenstein, Ming-Wei Chang, and Kristina Toutanova. 2021. Revisiting the primacy of english in zero-shot cross-lingual transfer. *CoRR*, abs/2106.16171.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Ł ukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc.

Petar Veličković, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Liò, and Yoshua Bengio. 2018. Graph attention networks. In *International Conference on Learning Representations*.

Xiangpeng Wei, Rongxiang Weng, Yue Hu, Luxi Xing, Heng Yu, and Weihua Luo. 2021. On learning universal representations across languages. In *9th International Conference on Learning Representations, ICLR 2021, Virtual Event, Austria, May 3-7, 2021*.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Remi Louf, Morgan Funtowicz, Joe Davison, Sam Shleifer, Patrick von Platen, Clara Ma, Yacine Jernite, Julien Plu, Canwen Xu, Teven Le Scao, Sylvain Gugger, Mariama Drame, Quentin Lhoest, and Alexander Rush. 2020. Transformers: State-of-the-art natural language processing. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 38–45.

Shijie Wu and Mark Dredze. 2019. Beto, bentz, becas: The surprising cross-lingual effectiveness of BERT. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 833–844.

Yinfei Yang, Yuan Zhang, Chris Tar, and Jason Baldridge. 2019. PAWS-X: A cross-lingual adversarial dataset for paraphrase identification. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 3687–3692.

A Implementation details

Dataset XNLI is the cross-lingual natural language inference task. PAWS-X is used to determine whether two sentences paraphrase each other. The augmentation datasets are obtained from Huang et al. (2021). They augmented 3 and 10 examples for each sentence in XNLI and PAWS-X by synonym replacement, respectively. The pre-defined English synonym set is from Alzantot et al. (2018). The scripts for splitting training, test, and validation sets are provided by XTREME (Hu et al., 2020).

Layer	en	ar	bg	de	el	es	fr	hi	ru	sw	th	tr	ur	vi	zh	avg.
3	83.23	67.17	71.44	73.33	68.06	75.99	74.89	63.27	70.94	51.00	56.61	63.23	61.04	72.30	71.66	68.28
6	83.05	67.01	70.88	72.63	67.98	76.05	74.91	62.99	71.82	51.28	56.81	63.53	61.44	72.48	71.48	**68.29**
9	82.87	67.56	71.22	73.05	68.36	75.81	74.63	63.65	71.14	50.96	56.75	62.97	61.00	72.55	71.68	68.28
12	83.05	67.05	70.56	72.81	68.22	75.55	75.35	63.35	71.48	50.82	56.71	63.11	60.30	72.48	71.68	68.17

Table A.1: Results of the XNLI task when we apply the Attention-Pull target at different layers of mBERT.

β	en	ar	bg	de	el	es	fr	hi	ru	sw	th	tr	ur	vi	zh	avg.
0.1	83.23	67.41	71.04	72.93	68.28	75.75	75.19	63.99	71.64	51.28	56.73	63.59	61.38	72.44	71.50	**68.42**
0.2	83.15	67.05	71.38	73.79	68.34	75.99	75.01	63.53	71.58	50	56.41	63.21	60.52	72.40	71.88	68.32
0.3	83.09	67.47	71.52	72.99	68.44	75.65	75.03	63.57	71.42	50.76	55.87	63.29	61.26	72.63	71.58	68.30
0.5	83.01	67.15	70.58	72.95	68.10	75.87	74.99	62.99	71.64	50.34	56.37	63.61	60.82	72.57	72.18	68.21
0.7	82.69	66.83	71.08	72.87	68.14	75.89	74.43	63.15	71.00	51.60	56.57	63.15	60.88	72.16	71.82	68.15
0.9	82.51	66.83	71.00	72.87	68.50	75.65	75.01	62.99	71.30	50.68	55.77	63.29	61.38	72.42	71.54	68.12

Table A.2: The experimental results of the XNLI task based on mBERT when β takes different values, where α=1.

α	en	ar	bg	de	el	es	fr	hi	ru	sw	th	tr	ur	vi	zh	avg.
0.6	82.85	67.35	71.64	73.03	68.32	75.65	74.57	63.37	71.56	50.56	56.39	63.67	61.16	72.44	72.02	68.30
0.8	82.87	67.23	70.96	73.17	68.68	75.23	74.87	63.53	71.26	50.86	56.45	63.21	61.30	72.55	72.02	68.28
1	83.23	67.41	71.04	72.93	68.28	75.75	75.19	63.99	71.64	51.28	56.73	63.59	61.38	72.44	71.50	**68.42**
1.2	83.19	67.03	71.08	72.97	67.86	75.87	74.75	63.49	71.50	51.60	56.37	63.21	60.88	72.59	71.28	68.24
1.4	83.09	67.03	71.44	73.35	68.78	75.79	74.51	63.23	71.50	51.14	56.35	63.45	60.72	72.75	71.60	68.32
1.6	83.19	67.05	71.50	73.23	68.18	76.25	74.77	63.43	71.06	51.10	56.43	62.95	60.52	72.38	71.98	68.27
1.8	83.29	67.09	71.44	73.51	68.54	75.85	74.79	63.83	71.36	50.78	56.47	63.23	60.86	72.59	71.98	68.37

Table A.3: The experimental results of the XNLI task based on mBERT when α takes different values, where β=0.1.

Setup The $\text{mBERT}_{\text{base}}$ and $\text{XLM-R}_{\text{base}}$ are obtained from Huggingface's *transformers* package (Wolf et al., 2020). The maximum sequence length is set as 128. The learning rate is set as 2e-5. Our method is trained for one epoch with the batch size of 32. other models are trained following Hu et al. (2020) and Huang et al. (2021).

Input construction Both XNLI and PAWS-X are sentence pair classification tasks. Taking mBERT as an example, for each s_1, s_2 and augmented s_1^a, s_2^a in the training data, we set $\boldsymbol{x}$ as [CLS]s_1[SEP]s_2[SEP], $\boldsymbol{x^a}$ as [CLS]s_1^a[SEP]s_2^a[SEP]. Then, we take $\boldsymbol{x}$ and $\boldsymbol{x^a}$ as the input of our method in Figure 2, [CLS] token is used for classification.

B Hyperparameter analysis

There are three main hyperparameters in our method that need to be adjusted. 1) We need to determine which layer is most effective for applying Attention-Pull target. 2) We need to determine the weight of β in the final loss. 3) We need to determine the weight of α in the final loss. We conduct experiments on XNLI task based on mBERT.

For 1), we first set α=1 and β=1, then apply the Attention-Pull target on the {3, 6, 9, 12} layers respectively, and the results are shown in Table A.1. We find that applying the Attention-Pull target to all layers works well. The most significant improvement is achieved at the 6-th layer and the minimal improvement is achieved at the last layer, which may be related to the quality of sentence representation at different layers of the model (Carlsson et al., 2021; Merchant et al., 2020).

For 2), we apply the Attention-Pull target at the 6-th layer and set α=1, then select β from {0.1, 0.2, 0.3, 0.5, 0.7, 0.9}. The experimental results are shown in Table A.2. First, we find that model performance improved when using any of the above β values. Second, we also find that the improvement becomes significant as β decreases, we attribute this phenomenon to the fact that the Attention-Pull target should not over-focus on features of the English corpus but should help the VME capture features in other language clusters. Note that this result does not mean that the Attention-Pull target is unnecessary, as ablation experiments in section 3.4 show that the Attention-Pull target can improve the model. Finally, the best experimental result is obtained when β=0.1.

For 3), we apply the Attention-Pull target at the 6-th layer and set β=0.1, then select α from {0.6,

Model	en	es	de	fr	bg	ru	el	th
EPT + APT	84.6	79.4	77.5	79.5	78.8	76.8	77.0	73.9
NT + APT	84.4	79.4	77.2	79.0	78.8	76.7	76.4	73.4
EPT + SRPT	84.4	80.0	77.8	79.2	78.5	76.8	77.1	74.2
Model	**sw**	**vi**	**ar**	**zh**	**hi**	**ur**	**tr**	**avg.**
EPT + APT	66.7	76.4	74.5	75.8	72.6	68.7	74.7	**75.8**
NT + APT	67.3	76.3	73.6	75.2	72.2	67.7	74.1	75.5
EPT + SRPT	65.2	76.6	73.5	75.8	72.5	68.7	74.4	75.6

Table A.4: Results on the XNLI task when replacing some targets, based on the XLM-R.

0.8, 1.0, 1.2, 1.4, 1.6, 1.8}. Results are shown as Table A.3. We find that the best performance is achieved when α is 1.0. The performance is also improved when using other α values, which shows that the Embedding-Push target can robustly improve the cross-lingual transferability of models. Therefore, in our main experiments, we set α=1.0, β=0.1 and apply the Attention-Pull target at the 6-th layer.

C Analysis on XLM-R

We perform analysis based on XLM-R, the results are shown in Table A.4.

Towards Multi-Sense Cross-Lingual Alignment of Contextual Embeddings

Linlin Liu[12*] **Thien Hai Nguyen**[1] **Shafiq Joty**[2] **Lidong Bing**[1†] **Luo Si**[1]
[1]DAMO Academy, Alibaba Group
[2]Nanyang Technological University, Singapore
{linlin.liu, thienhai.nguyen, l.bing, luo.si}@alibaba-inc.com
srjoty@ntu.edu.sg

Abstract

Cross-lingual word embeddings (CLWE) have been proven useful in many cross-lingual tasks. However, most existing approaches to learn CLWE including the ones with contextual embeddings are sense agnostic. In this work, we propose a novel framework to align contextual embeddings at the sense level by leveraging cross-lingual signal from bilingual dictionaries only. We operationalize our framework by first proposing a novel sense-aware cross entropy loss to model word senses explicitly. The monolingual ELMo and BERT models pretrained with our sense-aware cross entropy loss demonstrate significant performance improvement for word sense disambiguation tasks. We then propose a sense alignment objective on top of the sense-aware cross entropy loss for cross-lingual model pretraining, and pretrain cross-lingual models for several language pairs (English to German/Spanish/Japanese/Chinese). Compared with the best baseline results, our cross-lingual models achieve 0.52%, 2.09% and 1.29% average performance improvements on zero-shot cross-lingual NER, sentiment classification and XNLI tasks, respectively. [1]

1 Introduction

Cross-lingual word embeddings (CLWE) provide a shared representation space for knowledge transfer between languages, yielding state-of-the-art performance in many cross-lingual natural language processing (NLP) tasks. Most of the previous works have focused on aligning static embeddings. To utilize the richer information captured by the pretrained language model, more recent approaches attempt to extend previous methods to align contextual representations.

Aligning the dynamic and complex contextual spaces poses significant challenges, so most of the existing approaches only perform coarse-grained alignment. Schuster et al. (2019) compute the average of contextual embeddings for each word as an anchor, and then learn to align the *static* anchors using a bilingual dictionary. In another work, Aldarmaki and Diab (2019) use parallel sentences in their approach, where they compute sentence representations by taking the average of contextual word embeddings, and then they learn a projection matrix to align sentence representations. They find that the learned projection matrix also works well for word-level NLP tasks. Besides, unsupervised multilingual language models (Devlin et al., 2018; Artetxe and Schwenk, 2019; Conneau et al., 2019; Liu et al., 2020) pretrained on multilingual corpora have also demonstrated strong cross-lingual transfer performance. However, studies (Wang et al., 2020; Cao et al., 2020; Efimov et al., 2022; Tien and Steinert-Threlkeld, 2022) have shown that adjusting the unsupervised multilingual language model with parallel sentences can help further improve cross-lingual performance.

Though contextual word embeddings are intended to provide different representations of the same word in distinct contexts, Schuster et al. (2019) find that the contextual embeddings of different senses of one word are much closer compared with that of different words. This contributes to the anisomorphic embedding distribution of different languages and causes problems for cross-lingual alignment. For example, it will be difficult to align the English word *bank* and its Japanese translations 銀行 and 岸 that correspond to its two different senses, since the contextual embeddings of different senses of *bank* are close to each other while those of 銀行 and 岸 are far. Zhang et al. (2019) propose two solutions to handle multi-sense words: 1) remove multi-sense words and then align anchors in the same way as Schuster et al. (2019);

* Linlin Liu is under the Joint PhD Program between Alibaba and Nanyang Technological University.
† Corresponding author.
[1]Our code is available at https://github.com/ntunlp/multisense_embedding_alignment.git.

Proceedings of the 29th International Conference on Computational Linguistics, pages 4381–4396
October 12–17, 2022.

2) generate cluster level average anchor for contextual embeddings of multi-sense words and then learn a projection matrix in an unsupervised way with MUSE (Conneau et al., 2017). They do not make good use of the bilingual dictionaries, which are usually easy to obtain, even in low-resource scenarios. Moreover, their projection-based approach still cannot handle the anisomorphic embedding distribution problem.

In this work, we propose a novel sense-aware cross entropy loss to model multiple word senses explicitly, and then leverage a sense level translation task on top of it for cross-lingual model pretraining. The proposed sense level translation task enables our models to provide more isomorphic and better aligned cross-lingual embeddings. We only use the cross-lingual signal from bilingual dictionaries for supervision. Our pretrained models demonstrate consistent performance improvements on zero-shot cross-lingual NER, sentiment classification and XNLI tasks. Though pretrained on less data, our model achieves the state-of-the-art result on zero-shot cross-lingual German NER task. To the best of our knowledge, we are the first to perform sense-level contextual embedding alignment with only bilingual dictionaries.

2 Background: prediction tasks of language models

Next token prediction and masked token prediction are two common tasks in neural language model pretraining. We take two well-known language models, ELMo (Peters et al., 2018) and BERT (Devlin et al., 2018), as examples to illustrate these two tasks (architectures are shown in §A).

Next token prediction ELMo uses next token prediction tasks in a bidirectional language model. Given a sequence of N tokens $(t_1, t_2, \ldots, t_N)$, it first prepares a context independent representation for each token by using a convolutional neural network over the characters or by word embedding lookup (*a.k.a. input embeddings*). These representations are then fed into L layers of LSTMs to generate the contextual representations: $\boldsymbol{h}_{i,j}$ for token t_i at layer j. The model assigns a learnable *output embedding* $\boldsymbol{w}$ for each token in the vocabulary, which has the same dimension as $\boldsymbol{h}_{i,L}$. Then, the forward language model predicts the token at position k with:

$$\begin{aligned} & p(t_k|t_1, t_2, \ldots, t_{k-1}) \\ &= \text{softmax}(\boldsymbol{h}_{k-1,L}^{\mathsf{T}} \boldsymbol{w}_{k'}) \\ &= \frac{\exp(\boldsymbol{h}_{k-1,L}^{\mathsf{T}} \boldsymbol{w}_{k'})}{\sum_{i=1}^{V} \exp(\boldsymbol{h}_{k-1,L}^{\mathsf{T}} \boldsymbol{w}_i)} \end{aligned} \tag{1}$$

where k' is the index of token t_k in the vocabulary, V is the size of the vocabulary, and $(\boldsymbol{w}_1, \ldots, \boldsymbol{w}_V)$ are the output embeddings for the tokens in the vocabulary. The backward language model is similar to the forward one, except that tokens are predicted in the reverse order. Since the forward and backward language models are very similar, we will only describe our proposed approach in the context of the forward language model in the subsequent sections.

Masked token prediction The Masked Language Model (MLM) in BERT is a typical example of masked token prediction. Given a sequence $(t_1, t_2, \ldots, t_N)$, this approach randomly masks a certain percentage (15%) of the tokens and generates a masked sequence $(m_1, m_2, \ldots, m_N)$, where $m_k = [mask]$ if the token at position k is masked, otherwise $m_k = t_k$. BERT first prepares the context independent representations $(\boldsymbol{x}_1, \boldsymbol{x}_2, \ldots, \boldsymbol{x}_N)$ of the masked sequence via token embeddings. It is then fed into L layers of transformer encoder (Vaswani et al., 2017) to generate "bidirectional" contextual token representations. The final layer representations are then used to predict the masked token at position k as follows:

$$\begin{aligned} & p(m_k = t_k|m_1, \ldots, m_N) \\ &= \text{softmax}(\boldsymbol{h}_{k,L}^{\mathsf{T}} \boldsymbol{w}_{k'}) \\ &= \frac{\exp(\boldsymbol{h}_{k,L}^{\mathsf{T}} \boldsymbol{w}_{k'})}{\sum_{i=1}^{V} \exp(\boldsymbol{h}_{k,L}^{\mathsf{T}} \boldsymbol{w}_i)} \end{aligned} \tag{2}$$

where k', V, $\boldsymbol{h}$ and $\boldsymbol{w}$ are similarly defined as in Eq. 1. Unlike ELMo, BERT ties the input and output embeddings.

3 Proposed framework

We first describe our proposed sense-aware cross entropy loss to model multiple word senses explicitly in language model pretraining. Then, we present our joint training approach with sense alignment objective for cross-lingual mapping of contextual word embeddings. The proposed framework can be applied to most of the recent neural language

models, such as ELMo, BERT and their variants. See Table 1 for a summary of the main notations used in this paper.

Notation	Description
t_k	k-th token in sentence
$t_{k,s}$	s-th sense of t_k
k'	index of token t_k in vocabulary
L	number of LSTM/Transformer layers
V	size of vocabulary
S	maximum number of senses per token
$\boldsymbol{h}_{k,j}$	contextual representation of token t_k in layer j
$\boldsymbol{h}_{k^*,L}$	contextual representation used in softmax function for predicting t_k
v_i	i-th word in vocabulary
$v_{i,s}$	s-th sense of v_i
$\boldsymbol{w}_i$	output embedding of v_i
$\boldsymbol{w}_{i,s}$	context-dependent output embedding (i.e. sense vector) of $v_{i,s}$
$\boldsymbol{c}_{i,s}$	sense cluster center of $v_{i,s}$
$\boldsymbol{C}_i$	sense cluster centers of v_i
d	dimension of contextual representations
$\boldsymbol{P}$	projection matrix for dimension reduction

Table 1: Summary of the main notations.

3.1 Sense-aware cross entropy loss

Limitations of original training objectives The training tasks with Eq. 1 and 2 maximize the normalized dot product of contextual representations ($\boldsymbol{h}_{k-1,L}$ or $\boldsymbol{h}_{k,L}$) with a weight vector $\boldsymbol{w}_{k'}$. The only difference is that $\boldsymbol{h}_{k-1,L}$ in Eq. 1 encodes the information of previous tokens in the sequence, while $\boldsymbol{h}_{k,L}$ in Eq. 2 encodes the information of the masked sequence. Therefore, without loss of generality, we use $\boldsymbol{h}_{k^*,L}$ to denote the contextual representation for predicting the next or masked token t_k.

Even though contextual language models like ELMo and BERT provide a different token representation for each distinct context, the learned representations are not guaranteed to be sense separated. For example, Schuster et al. (2019) computed the average of ELMo embeddings for each word as an *anchor*, and found that the average cosine distance between contextual embeddings of multi-sense words and their corresponding anchors are much smaller than the average distance between anchors, which mean that the embeddings of different senses of one word are relatively near to each other comparing to that of different words. We also observed the same with BERT embeddings. This finding suggests that sense clusters of a multi-sense word's appearances are not well separated in the embedding space, and the current contextual language models still have room for improvement by considering finer-grained word sense disambiguation.

Notice that there is only one weight vector $\boldsymbol{w}_{k'}$ for predicting the token t_k in the original training tasks. Ideally, we should treat the appearances of a multi-sense word in different contexts as different tokens, and train the language models to predict different senses of the word. In the following, we propose a novel sense-aware cross entropy loss to explicitly model different senses of a word in different contexts.

Sense-aware cross entropy loss Given a sequence $(t_1, t_2, \ldots, t_N)$, our proposed framework generates contextual representations ($\boldsymbol{h}_{k,j}$ for token t_k in layer $j \in \{1, \ldots, L\}$) in the same way as the standard LMs. Different from existing methods, our approach maintains multiple *context-dependent output embeddings* (henceforth, sense vectors) for each token. Specifically, let S be the maximum number of senses per token. Each word v_i in the vocabulary contains S separate sense vectors $(\boldsymbol{w}_{i,1}, \boldsymbol{w}_{i,2}, \ldots, \boldsymbol{w}_{i,S})$, where each $\boldsymbol{w}_{i,s}$ corresponds to a different sense (see Appendix for some interesting visualization examples). Following the notation in §2, we use k' to denote the index of the output token t_k in the vocabulary. Therefore, the sense vectors of t_k can be represented by $(\boldsymbol{w}_{k',1}, \boldsymbol{w}_{k',2}, \ldots, \boldsymbol{w}_{k',S})$, which are randomly initialized and of the same dimension as $\boldsymbol{h}_{k^*,L}$. Note that we untie the input and output embeddings in our framework.

We propose a word sense selection method shown in Algorithm 1 to select the most likely sense vector when training with sense-level cross entropy loss. Figure 1 shows the architecture of our proposed models. Assuming sense s' is selected for token t_k (which means sense vector $\boldsymbol{w}_{k',s'}$ should be used), we have the following new prediction task:

$$\begin{aligned} & p(t_{k,s'}|context) \\ &= \text{softmax}(\boldsymbol{h}_{k^*,L}^\top \boldsymbol{w}_{k',s'}) \\ &= \frac{\exp(\boldsymbol{h}_{k^*,L}^\top \boldsymbol{w}_{k',s'})}{\sum_{i=1}^{V}\sum_{s=1}^{S}\exp(\boldsymbol{h}_{k^*,L}^\top \boldsymbol{w}_{i,s})} \end{aligned} \tag{3}$$

The sense-aware cross entropy loss for word sense prediction is defined as follows:

$$\mathcal{L}_{\text{SENSE}} = -\log(p(t_{k,s'}|context)) \tag{4}$$

Word sense selection algorithm Word sense selection when training the language model can be

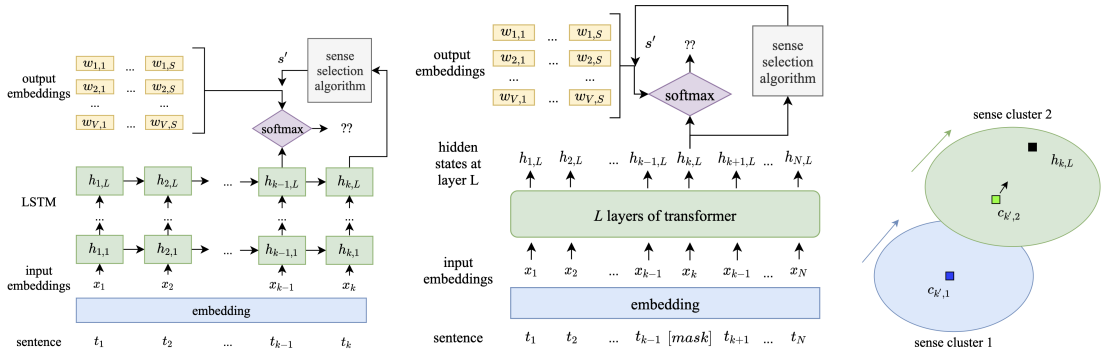

(a) Sense-aware next token prediction. (b) Sense-aware masked token prediction. (c) Word sense selection.

Figure 1: Our proposed framework for sense-aware next token and masked token prediction tasks. Since the backward language model for next token prediction is similar to the forward, we only show the forward one in (a) for simplicity. Figure (c) shows an example of word sense selection, where the two sense clusters of t_k (assume its vocabulary index is k') are shifting in space. Center vectors $\boldsymbol{c}_{k',1}$ and $\boldsymbol{c}_{k',2}$ are used to locate cluster centers. Given $\boldsymbol{h}_{k,L}$, the algorithm performs dimension reduction on both $\boldsymbol{h}_{k,L}$ and center vectors, and then finds the most close cluster center $\boldsymbol{c}_{k',2}$, so we know the output embedding corresponding to sense 2 ($\boldsymbol{w}_{k',2}$) should be used in the loss function. $\boldsymbol{c}_{k',2}$ also makes a small step towards $\boldsymbol{h}_{k,L}$.

handled as a non-stationary data stream clustering problem (Aggarwal et al., 2004; Khalilian and Mustapha, 2010; Abdullatif et al., 2018). The most intuitive way to select the corresponding sense vector for $\boldsymbol{h}_{k^*,L}$ is to select the vector $\boldsymbol{w}_{k',s}$ with the maximum dot product value $\boldsymbol{h}_{k^*,L}^{\top}\boldsymbol{w}_{k',s}$, or cosine similarity value $cossim(\boldsymbol{h}_{k^*,L}, \boldsymbol{w}_{k',s})$. However, our experiments show that these methods do not work well due to curse of dimensionality, suboptimal learning rate and noisy $\boldsymbol{h}_{k^*,L}$. We apply an online k-means algorithm to cluster different senses of a word in Algorithm 1. For each sense vector $\boldsymbol{w}_{i,s}$, we maintain a cluster *center* $\boldsymbol{c}_{i,s}$ which is of the same dimension as $\boldsymbol{w}_{i,s}$. Therefore, each token v_i in the vocabulary has S such cluster center vectors, denoted by $\boldsymbol{C}_i = (\boldsymbol{c}_{i,1}, \boldsymbol{c}_{i,2}, \ldots, \boldsymbol{c}_{i,S})$. When predicting token t_k in a given sequence, we apply Algorithm 1 to select the best sense vector based on $\boldsymbol{h}_{k,L}$ (see Figure 1). Notice that $\boldsymbol{h}_{k,L}$ is different from $\boldsymbol{h}_{k^*,L}$ for next token prediction (Figure 1a) for which $\boldsymbol{h}_{k^*,L} = \boldsymbol{h}_{k-1,L}$. The cluster centers $\boldsymbol{C}_i$ are **not** neural network parameters; instead, they are randomly initialized using a normal distribution $\mathcal{N}(0, \sigma^2)$ and updated through Algorithm 1. In addition, we also maintain a projection matrix $\boldsymbol{P}$ for dimension reduction to facilitate effective sense clustering. $\boldsymbol{P} \in \mathbb{R}^{d \times d'}$ projects $\boldsymbol{h}_{k,L}$ and $\boldsymbol{c}_{i,s}$ from dimension d to d', and is shared by all tokens in vocabulary. Similar to $\boldsymbol{C}$, $\boldsymbol{P}$ is also randomly initialized with normal distribution $\mathcal{N}(0, 1)$, and then updated through Algorithm 2. Both Algorithm 1 and 2 run in parallel, and are interrupted when the language model stops training.

Algorithm 1 Word sense selection

Hyper-parameters: number of senses S, sense learning rate α
Initialize the set of all sense cluster centers $\boldsymbol{C}$
repeat
  input: $\boldsymbol{h}_{k,L}$, vocabulary index k' of the token to predict
  Lookup sense cluster centers for k': $\boldsymbol{C}_{k'} = \{\boldsymbol{c}_{k',1}, \boldsymbol{c}_{k',2}, \ldots, \boldsymbol{c}_{k',S}\}$
  $\boldsymbol{P}$ = updated projection matrix from Alg. 2
  if cosine similarity between $\boldsymbol{c}_{k',s'}\boldsymbol{P}$ and $\boldsymbol{h}'_k\boldsymbol{P}$ is the largest among the vectors in $\boldsymbol{C}_{k'}$ **then**
    $\boldsymbol{c}_{k',s'} = (1-\alpha)\boldsymbol{c}_{k',s'} + \alpha\boldsymbol{h}_{k,L}$
    output: s'($\boldsymbol{w}_{k',s'}$ should be selected)
  end if
until interrupted

Some rationales behind our algorithm design are the following:

- Directly computing cosine similarity between $\boldsymbol{c}_{k',s}$ and $\boldsymbol{h}_{k,L}$ suffers from the curse of dimensionality. We maintain $\boldsymbol{P}$ for dimension reduction. Although many algorithms use random projection for dimension reduction, we find using PCA components can help improve clustering accuracy.
- Since the neural model parameters keep being updated during training, the sense clusters become non-stationary, i.e., their locations keep changing. Experiments shows that when using $\boldsymbol{P}$ for dimension reduction, a slightly larger projection dimension d' will make the clustering algorithm

Algorithm 2 Projection matrix $\boldsymbol{P}$ update

Hyper-parameters: projection dimension d', update interval M, queue size Q
Initialize $\boldsymbol{P}$ with $\mathcal{N}(0,1)$, queue $H = \emptyset$, $m = 0$
repeat
  input: $\boldsymbol{h}_{k,L}$
  $m = m + 1$
  Add $\boldsymbol{h}_{k,L}$ to queue H
  if $size(H) > Q$ **then**
    Pop the oldest element from queue H.
  end if
  if $m >= M$ **then**
    $\boldsymbol{P}$ = the first d' PCA components of H
    $m = 0$
  end if
  output: $\boldsymbol{P}$
until interrupted

less sensitive to cluster location change. We use $d' = 16$ for ELMo, and $d' = 14$ for BERT. We also notice that the sense clustering works well even if $\boldsymbol{P}$ is updated sporadically. We can set a relatively large update interval in Algorithm 2 to reduce computation cost.

- A separate sense learning rate α should be set for the clustering algorithm. A large α makes the algorithm less robust to noise, while a small α leads to slow convergence.
- It is essential to use the current token's contextual representation $\boldsymbol{h}_{k,L}$ for sense selection even though we use $\boldsymbol{h}_{k^*,L} = \boldsymbol{h}_{k-1,L}$ in the next token prediction task. If we use $\boldsymbol{h}_{k-1,L}$ for sense selection, experiments show that most of the variance comes from input embedding $\boldsymbol{x}_{k-1}$. This introduces too much noise for word sense clustering.

Dynamic pruning of redundant word senses To make the training more efficient, we keep track of relative sense selection frequency for each token in the vocabulary. Assume token v_i has initial senses $(v_{i,1}, v_{i,2}, \ldots, v_{i,S})$, for which we compute the relative frequency $\rho(v_{i,s})$ such that $0 \leq \rho(v_{i,s}) \leq 1$ and $\sum_s \rho(v_{i,s}) = 1$. A lower $\rho(v_{i,s})$ means the sense is less frequently selected compared with others. We check the relative frequencies after every E training steps, and if $\rho(v_{i,s}) < \beta$ (a threshold hyper-parameter), $v_{i,s}$ is removed from the list of senses of v_i.

Remark on model size and parameters The sense cluster centers $\boldsymbol{C}$ and the projection matrix $\boldsymbol{P}$ are only used to facilitate sense selection during model pretraining, which are not neural model parameters. The sense vectors $\boldsymbol{w}_{i,s}$ will no longer be used after pretraining, which can also be discarded. Therefore, our models and the original models have exactly the same number of parameters when transferred to downstream tasks.

Remark on model complexity The computational complexity of our algorithm is linear with respect to the size of data, so our method is scalable to train on very large datasets.

3.2 Joint training with sense level translation

Training language model with sense-aware cross entropy loss helps to learn contextual token representations that are sufficiently distinct for different senses (§4.1). In this subsection, we extend it to cross-lingual settings and present a novel approach to learn cross-lingual contextual word embeddings at the sense level. Our approach uses a bilingual seed dictionary,[2] and can be applied to both next and masked token prediction tasks.

For training the cross-lingual LM, we concatenate the (non-parallel) corpora of two languages, L_1 and L_2, and construct a joint vocabulary $O = O^{L_1} \cup O^{L_2}$, where O^{L_1} and O^{L_2} are the vocabularies of L_1 and L_2, respectively. Algorithm 1 is used to model the senses of tokens in the joint vocabulary. In addition to predicting the correct monolingual sense $p(t_{k,s'}|context)$ in Eq. 3, we also train the model to predict its sense level translation. Let v_j be the translation of t_k and sense v_{j,s^*} of v_j be the best sense level translation under the given context, we add the following sense-level translation prediction task to maximize probability of v_{j,s^*}.

$$\begin{aligned} & p(v_{j,s^*}|context) \\ &= \text{softmax}(\boldsymbol{h}_{k^*,L}^{\mathsf{T}} \boldsymbol{w}_{j,s^*}) \\ &= \frac{\exp(\boldsymbol{h}_{k^*,L}^{\mathsf{T}} \boldsymbol{w}_{j,s^*})}{\sum_{i=1}^{V} \sum_{s=1}^{S} \exp(\boldsymbol{h}_{k^*,L}^{\mathsf{T}} \boldsymbol{w}_{i,s})} \end{aligned} \tag{5}$$

where $\boldsymbol{w}_{j,s^*}$ is the corresponding sense vector of v_{j,s^*}.

Similar to the previous subsection, we maintain sense cluster centers $\boldsymbol{C}_i$ for each token $v_i \in O$ and the shared projection matrix $\boldsymbol{P}$ to select the best translation sense. Assume t_k has T translations in dictionary, and each translation has S senses, then there are $T \times S$ possible sense level translations for t_k in the given context. If the $cossim(\boldsymbol{h}_{k,L}\boldsymbol{P}, \boldsymbol{c}_{j,s^*}\boldsymbol{P})$ value is the largest among the $T \times S$ sense cluster centers, then we select v_{j,s^*}

[2] If not provided, it can be learned in an unsupervised way, e.g., MUSE (Conneau et al., 2017).

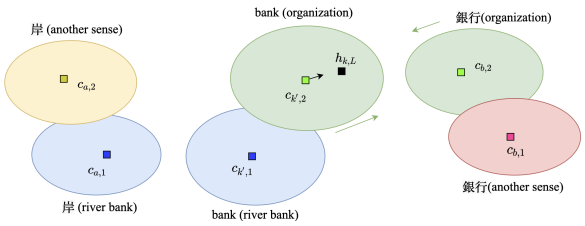

Figure 2: An example of English-Japanese sense-level joint training, which shows two possible Japanese translations (銀行 and 岸) of the English word *bank*. $\boldsymbol{h}_{k,L}$ is a contextual representation of *bank* in finance context and $\boldsymbol{c}_{k',2}$ is the cluster center for this sense. $\boldsymbol{c}_{a,1}$, $\boldsymbol{c}_{a,2}$, $\boldsymbol{c}_{b,1}$, $\boldsymbol{c}_{b,2}$ are different sense cluster centers of the two Japanese translations, among which $\boldsymbol{c}_{b,2}$ is the closest to $\boldsymbol{h}_{k,L}$ after dimension reduction through PCA. Our sense level objective (Eq. 6) moves sense clusters for *bank (organization)* and 銀行*(organization)* closer to each other.

as the closest translation. An example is shown in Figure 2. If token t_k has at least one translation in the dictionary, the translation cross entropy loss can be computed as:

$$\mathcal{L}_{\text{TRAN}} = -\log(p(v_{j,s^*}|context)) \quad (6)$$

If token t_k has no translation in the seed dictionary, we use Eq. 4 as the only loss. The joint training loss is defined as follows:

$$\mathcal{L}_{\text{JOINT}} = \begin{cases} \frac{\mathcal{L}_{\text{SENSE}}+\mathcal{L}_{\text{TRAN}}}{2}, & \text{if } t_k \text{ has translations} \\ \mathcal{L}_{\text{SENSE}}, & \text{otherwise} \end{cases} \quad (7)$$

Further alignment (optional) Our sense-aware pretraining tries to move similar senses of two different languages close to each other as illustrated in Figure 2. This process makes the sense distributions of the two languages more isomorphic (some sense vector visualization examples are shown in §D). Applying the linear projection approach proposed by Schuster et al. (2019) on top of the language model pretrained with our framework can further improve cross-lingual transfer on some tasks. See §B for more details of our implementation.

4 Experiments

4.1 Experiments using monolingual models

To verify the effectiveness of our proposed sense-aware cross entropy loss, we implement the monolingual models on top of ELMo and BERT with the changes described in §3.1, which are named **SaELMo** (Sense-aware ELMo) and **SaBERT** (Sense-aware BERT) respectively. The algorithm for dynamic pruning of redundant word senses is optional, which is implemented on SaELMo only.

Model	SE2	SE3	SE07	SE13	SE15
ELMo	0.555	0.576	0.446	0.544	0.538
SaELMo (ours)	**0.575**	**0.586**	**0.470**	**0.560**	**0.583**
BERT-Tiny	0.596	0.539	**0.466**	0.536	0.572
SaBERT-Tiny (ours)	**0.611**	**0.546**	0.446	**0.550**	**0.579**

Table 2: Word sense disambiguation (F1 scores).

Pretraining settings We use the one billion word language modeling benchmark data (Chelba et al., 2013) to pretrain all the monolingual models. The corpus is preprocessed with the provided scripts, and then converted to lowercase. We do not apply any subword tokenization. We use similar hyper-parameters as Peters et al. (2018) to train the ELMo and SaELMo models, and similar hyper-parameters as Devlin et al. (2018) to train 4-layer BERT-Tiny and SaBERT-Tiny. Next sentence prediction task is disabled in BERT-Tiny and SaBERT-Tiny, since this task is irrelevant to our proposed changes. See §C.1 for a complete list of hyper-parameters.

Word sense disambiguation (WSD) Since our context-aware cross entropy loss is designed to learn word senses better in the context, we first conduct experiments to compare our monolingual model with the original models on the WSD task (Raganato et al., 2017), which is a task to associate words in context with the most suitable entry in a pre-defined sense inventory. We use SemCor 3.0 (Miller et al., 1993) as training data, and Senseval/SemEval series (Edmonds and Cotton, 2001; Moro and Navigli, 2015; Navigli et al., 2013; Pradhan et al., 2007; Snyder and Palmer, 2004) as test data. We use the pretrained models to compute the average of contextual representations for each sense in training data, and then classify the senses of the target words in test sentences by finding the nearest neighbour from all senses entries without pre-filtering senses by lemma.[3] WSD results are presented in Table 2. SaELMo shows significant performance improvements over the baseline ELMo model in all of the five test sets. SaBERT-Tiny also outperforms BERT-Tiny except on SE07,

[3]We use the evaluation code from https://github.com/drgriffis/ELMo-WSD-reimplementation.git.

which is the smallest among the five test sets.

4.2 Experiments using bilingual models

As discussed in §3.1, our cross-lingual framework is designed to address the same problem identified in the training objectives of ELMo and Transformer-based language models. To verify its effectiveness, we implement the bilingual models on top of ELMo, named **Bi-SaELMo** that does not use linear projection for further alignment and **Bi-SaELMo+Proj** that uses the linear projection. Sense vectors and cluster center vectors are not shared between the forward and backward language models. We use **ELMo+Proj** and **Joint-ELMo+Proj** as our baseline models, where ELMo+Proj is proposed by Schuster et al. (2019) and Joint-ELMo+Proj is implemented following the framework recently proposed by Wang et al. (2020). Wang et al. (2020) combine joint training and projection, and claim their framework is applicable to any projection method, so we implement the same projection method as Schuster et al. (2019) did for Joint-ELMo+Proj. We also report results of **ELMo** and **Joint-ELMo**, which are the counterparts of ELMo+Proj and Joint-ELMo+Proj without using linear projection.

Pretraining settings To pretrain language models, we sample a 500-million-token corpus for each language from the English, German, Spanish, Japanese and Chinese Wikipedia dump. The dictionaries used for pretraining models and learning the projection matrix were downloaded from the MUSE (Conneau et al., 2017) GitHub page[4]. We also add JMDict (Breen, 2004) to the *en-jp* MUSE dictionary. Bilingual models were pretrained on *en-de*, *en-es*, *en-jp* and *en-zh* concatenated data with similar parameters as the monolingual models. ELMo and ELMo+Proj were pretrained on monolingual data, while the projection matrix of ELMo+Proj was learned using bilingual data. See §C.2 for a complete list of hyper-parameters.

Zero-shot cross-lingual NER A Bi-LSTM-CRF model implemented with the Flair framework (Akbik et al., 2018) is used for this task. For the CoNLL-2002 (Tjong Kim Sang, 2002) and CoNLL-2003 (Sang and De Meulder, 2003) datasets, the NER model was trained on English data, and evaluated on Spanish and German test data. For the OntoNotes 5.0 (Weischedel et al., 2013) dataset, the NER model was trained on all English data and evaluated on all Chinese data. We report the average F1 of 5 runs in Table 3. The results show that all of the models using linear projection outperform their counterparts (not using linear projection), since minimizing token level distance is more important for cross-lingual NER tasks. Our sense-aware pretraining makes sense distributions of two languages more isomorphic, which further improves linear projection performance. Our model Bi-SaELMo+Proj demonstrates consistent performance improvement in all the three languages. Moreover, our model outperforms fine-tuned XLM/XLM-R and Multilingual BERT on German data even though it is pretrained on less data.

Model	de	es	zh
ELMo	16.30	16.14	0.28
Joint-ELMo	56.49	58.91	53.47
ELMo+Proj (Schuster et al., 2019)	69.57	60.02	63.15
Joint-ELMo+Proj (Wang et al., 2020)	71.59	65.19	59.08
Bi-SaELMo (ours)	63.83	60.65	55.83
Bi-SaELMo+Proj (ours)	**72.19**	**65.86**	**63.44**
For references, but not our baselines, since they are trained on much larger datasets and/or parallel sentences.			
XLM Finetune (Conneau and Lample, 2019)	67.55	63.18	-
mBERT Finetune (Pires et al., 2019)	69.74	73.59	-
XLM-R_{base} Finetune (Liang et al., 2020)	70.40	75.20	-
mBERT Feature+Proj (Wang et al., 2020)	70.54	75.77	-
mBERT Align (Kulshreshtha et al., 2020)	71.23	75.93	-

Table 3: Zero-shot cross-lingual NER (F1).

Zero-shot cross-lingual sentiment classification We use the multi-lingual multi-domain Amazon review data (Prettenhofer and Stein, 2010) for evaluation on cross-lingual sentiment classification. The ratings in review data are converted into binary labels. The average of contextual word representations is used as the document/sentence representation for each review text/summary, which is then fed into a two-dense-layer model for sentiment classification. All the models are trained on English, and evaluated on German and Japanese test data in the same domain. We report the average accuracy of 5 runs in Table 4. Different from the NER task, the linear projection approach for cross-lingual alignment does not work for this task, since it may add noise to embedding features. Our model Bi-SaELMo demonstrates consistent improvements in all of the 6 evaluation tasks. The performance of Bi-SaELMo is significantly better than Joint-ELMo, which shows that our sense-level transla-

[4] https://github.com/facebookresearch/MUSE

Model	de			jp		
	books	music	dvd	books	music	dvd
ELMo	52.94	63.61	57.78	50.37	51.59	54.32
Joint-ELMo	71.72	75.22	64.25	66.64	68.50	58.54
ELMo+Proj (Schuster et al., 2019)	49.92	50.29	49.94	50.57	49.59	50.65
Joint-ELMo+Proj (Wang et al., 2020)	75.74	72.25	72.25	62.50	59.77	57.65
Bi-SaELMo (ours)	**77.46**	**75.32**	**74.97**	**68.16**	**69.48**	**64.04**
Bi-SaELMo+Proj (ours)	70.84	66.25	68.99	62.17	55.91	61.57

Table 4: Zero-shot sentiment classification accuracy.

Model	de	es	zh
ELMo	34.07	33.41	35.77
Joint-ELMo	60.12	63.73	57.82
ELMo+Proj (Schuster et al., 2019)	55.51	58.92	53.17
Joint-ELMo+Proj (Wang et al., 2020)	63.33	64.71	58.34
PROC-B+SpecNorm (Aboagye et al., 2022)	62.40	-	-
Bi-SaELMo (ours)	60.98	62.75	60.40
Bi-SaELMo+Proj (ours)	**64.77**	**65.05**	**60.44**

Table 5: Zero-shot XNLI accuracy.

tion pretraining objective improves cross-lingual embedding alignment.

Zero-shot cross-lingual natural language inference (XNLI) We use XNLI (Conneau et al., 2018) and MultiNLI (Williams et al., 2018) data for evaluation on this task. The Bi-LSTM baseline model[5] was trained on MultiNLI English training data, and then evaluated on XNLI German, Spanish, Chinese test data. We report the average zero-shot XNLI accuracy of 2 runs in Table 5. Our models show consistent improvements over the baselines on all of the three data sets. For zero-shot transfer to Chinese, both of our models outperform the best baseline by more than 2 points, which again demonstrates the effectiveness of our framework on distant language pairs.

5 Related work

Cross-lingual word embedding demonstrates strong performance in many cross-lingual transfer tasks(Wu and Dredze, 2019; Li et al., 2020b,a; Zhang et al., 2021). The projection-based approach has a long line of research on aligning static embeddings (Mikolov et al., 2013; Xing et al., 2015; Smith et al., 2017; Joulin et al., 2018; Aboagye et al., 2022). It assumes that the embedding spaces of different languages have an isomorphic structure, and fit an orthogonal matrix to project multiple monolingual embedding spaces to a shared space. Many studies (Schuster et al., 2019; Aldarmaki and Diab, 2019) have extended the projection-based approach to contextual representation alignment. Besides, there are many discussions on the limitations of the projection-based approach, arguing that the isomorphic assumption is not true in general (Nakashole and Flauger, 2018; Patra et al., 2018; Søgaard et al., 2018; Ormazabal et al., 2019), so non-linear mapping methods are also explored in recent work (Mohiuddin et al., 2020; Ganesan et al., 2021). Joint training is another line of research and early methods (Gouws et al., 2015; Luong et al., 2015; Ammar et al., 2016) learn static word embeddings of multiple languages simultaneously. Extending joint training to cross- or multi-lingual language model pretraining has gained more attention recently. As discussed above, unsupervised multilingual language models (Devlin et al., 2018; Artetxe and Schwenk, 2019; Conneau and Lample, 2019; Conneau et al., 2019; Liu et al., 2020, 2021) also demonstrate strong cross-lingual transfer performance.

There has been some work on sense-aware language models/embeddings (Rothe and Schütze, 2015; Pilehvar and Collier, 2016; Hedderich et al., 2019), and most of them require WordNet (Miller, 1998) or other additional resource for supervision. Šuster et al. (2016) utilize both monolingual and bilingual information from parallel corpora to learn multi-sense word embeddings. Peters et al. (2019) embed WordNet knowledge into BERT with attention mechanism. Levine et al. (2019) pretrain SenseBERT to predict both the masked words and their WordNet supersenses. Similar to our framework, there are also some unsupervised approaches, but most of them are used to learn static embeddings. Huang et al. (2012) learn word representations with both local and global context, and then apply a clustering algorithm to learn multi-prototype vectors. Neelakantan et al. (2014) propose an extension to the Skip-gram model that leverage k-means clustering algorithm learns multiple embeddings per word type. Lee and Chen (2017) leverage reinforcement learning for modularized unsupervised sense level embedding learning. Boyd-Graber et al. (2020) use Gumbel softmax for sense disambiguation when learning sense embeddings.

6 Conclusions

In this paper, we have introduced a novel sense-aware cross entropy loss to model word senses explicitly, then we have further proposed a sense-level alignment objective for cross-lingual model pretraining using only bilingual dictionaries. The results of the experiments show the effectiveness

[5] https://github.com/NYU-MLL/multiNLI

of our monolingual and bilingual models on WSD, zero-shot cross-lingual NER, sentiment classification and XNLI tasks. In future work, we will study how to extend our method to multilingual models.

Broader Impact

NLP has achieved significant success for many popular languages, such as English and German. However, most of the low-resource languages in the world do not receive enough attention from the NLP community. Cross-lingual word embedding is an efficient tool to help overcome the resource barrier and enable the advances in NLP to benefit a wider range of population. This makes NLP more inclusive of low-resource languages (and their speakers), and can also help preventing online bullying, detecting fake news, etc. in multiple languages. In this work, we proposed a novel framework for cross-lingual contextual word embedding alignment, which further improves the performance of cross-lingual transfer learning. Our findings and proposed techniques are potentially useful for future research on both monolingual and cross-lingual language model pretraining.

Acknowledgements

This research is partly supported by the Alibaba-NTU Singapore Joint Research Institute, Nanyang Technological University. Linlin Liu would like to thank the support from Interdisciplinary Graduate School, Nanyang Technological University.

References

Amr Abdullatif, Francesco Masulli, and Stefano Rovetta. 2018. Clustering of nonstationary data streams: A survey of fuzzy partitional methods. *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery*, 8(4):e1258.

Prince Osei Aboagye, Jeff Phillips, Yan Zheng, Junpeng Wang, Chin-Chia Michael Yeh, Wei Zhang, Liang Wang, and Hao Yang. 2022. Normalization of language embeddings for cross-lingual alignment. In *International Conference on Learning Representations*.

Charu C Aggarwal, Jiawei Han, Jianyong Wang, and Philip S Yu. 2004. A framework for projected clustering of high dimensional data streams. In *Proceedings of the Thirtieth international conference on Very large data bases-Volume 30*, pages 852–863.

Alan Akbik, Duncan Blythe, and Roland Vollgraf. 2018. Contextual string embeddings for sequence labeling. In *COLING 2018, 27th International Conference on Computational Linguistics*, pages 1638–1649.

Hanan Aldarmaki and Mona Diab. 2019. Context-aware crosslingual mapping. *arXiv preprint arXiv:1903.03243*.

Waleed Ammar, George Mulcaire, Yulia Tsvetkov, Guillaume Lample, Chris Dyer, and Noah A Smith. 2016. Massively multilingual word embeddings. *arXiv preprint arXiv:1602.01925*.

Mikel Artetxe and Holger Schwenk. 2019. Massively multilingual sentence embeddings for zero-shot cross-lingual transfer and beyond. *Transactions of the Association for Computational Linguistics*, 7:597–610.

Jordan Boyd-Graber, Fenfei Guo, Leah Findlater, and Mohit Iyyer. 2020. Which evaluations uncover sense representations that actually make sense? In *Proceedings of the 12th Language Resources and Evaluation Conference*, pages 1727–1738, Marseille, France. European Language Resources Association.

Jim Breen. 2004. Jmdict: a japanese-multilingual dictionary. In *Proceedings of the workshop on multilingual linguistic resources*, pages 65–72.

Steven Cao, Nikita Kitaev, and Dan Klein. 2020. Multilingual alignment of contextual word representations. *arXiv preprint arXiv:2002.03518*.

Ciprian Chelba, Tomas Mikolov, Mike Schuster, Qi Ge, Thorsten Brants, Phillipp Koehn, and Tony Robinson. 2013. One billion word benchmark for measuring progress in statistical language modeling.

Alexis Conneau, Kartikay Khandelwal, Naman Goyal, Vishrav Chaudhary, Guillaume Wenzek, Francisco Guzmán, Edouard Grave, Myle Ott, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Unsupervised cross-lingual representation learning at scale. *arXiv preprint arXiv:1911.02116*.

Alexis Conneau and Guillaume Lample. 2019. Cross-lingual language model pretraining. In *Advances in Neural Information Processing Systems*, pages 7057–7067.

Alexis Conneau, Guillaume Lample, Marc'Aurelio Ranzato, Ludovic Denoyer, and Hervé Jégou. 2017. Word translation without parallel data. *arXiv preprint arXiv:1710.04087*.

Alexis Conneau, Ruty Rinott, Guillaume Lample, Adina Williams, Samuel R. Bowman, Holger Schwenk, and Veselin Stoyanov. 2018. Xnli: Evaluating cross-lingual sentence representations. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*.

Philip Edmonds and Scott Cotton. 2001. Senseval-2: overview. In *Proceedings of SENSEVAL-2 Second International Workshop on Evaluating Word Sense Disambiguation Systems*, pages 1–5.

Pavel Efimov, Leonid Boytsov, Elena Arslanova, and Pavel Braslavski. 2022. The impact of cross-lingual adjustment of contextual word representations on zero-shot transfer. *arXiv preprint arXiv:2204.06457*.

Ashwinkumar Ganesan, Francis Ferraro, and Tim Oates. 2021. Learning a reversible embedding mapping using bi-directional manifold alignment. In *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*, pages 3132–3139, Online. Association for Computational Linguistics.

Stephan Gouws, Yoshua Bengio, and Greg Corrado. 2015. Bilbowa: Fast bilingual distributed representations without word alignments. In *Proceedings of the 32nd International Conference on Machine Learning*.

Michael A. Hedderich, Andrew Yates, Dietrich Klakow, and Gerard de Melo. 2019. Using multi-sense vector embeddings for reverse dictionaries. In *Proceedings of the 13th International Conference on Computational Semantics - Long Papers*, pages 247–258, Gothenburg, Sweden. Association for Computational Linguistics.

Eric Huang, Richard Socher, Christopher Manning, and Andrew Ng. 2012. Improving word representations via global context and multiple word prototypes. In *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 873–882, Jeju Island, Korea. Association for Computational Linguistics.

Armand Joulin, Piotr Bojanowski, Tomas Mikolov, Hervé Jégou, and Edouard Grave. 2018. Loss in translation: Learning bilingual word mapping with a retrieval criterion. *arXiv preprint arXiv:1804.07745*.

Madjid Khalilian and Norwati Mustapha. 2010. Data stream clustering: Challenges and issues. *arXiv preprint arXiv:1006.5261*.

Saurabh Kulshreshtha, Jose Luis Redondo Garcia, and Ching-Yun Chang. 2020. Cross-lingual alignment methods for multilingual BERT: A comparative study. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 933–942, Online. Association for Computational Linguistics.

Guang-He Lee and Yun-Nung Chen. 2017. MUSE: Modularizing unsupervised sense embeddings. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 327–337, Copenhagen, Denmark. Association for Computational Linguistics.

Yoav Levine, Barak Lenz, Or Dagan, Dan Padnos, Or Sharir, Shai Shalev-Shwartz, Amnon Shashua, and Yoav Shoham. 2019. Sensebert: Driving some sense into bert. *arXiv preprint arXiv:1908.05646*.

Juntao Li, Ruidan He, Hai Ye, Hwee Tou Ng, Lidong Bing, and Rui Yan. 2020a. Unsupervised domain adaptation of a pretrained cross-lingual language model. *arXiv preprint arXiv:2011.11499*.

Xin Li, Lidong Bing, Wenxuan Zhang, Zheng Li, and Wai Lam. 2020b. Unsupervised cross-lingual adaptation for sequence tagging and beyond. *arXiv preprint arXiv:2010.12405*.

Yaobo Liang, Nan Duan, Yeyun Gong, Ning Wu, Fenfei Guo, Weizhen Qi, Ming Gong, Linjun Shou, Daxin Jiang, Guihong Cao, Xiaodong Fan, Ruofei Zhang, Rahul Agrawal, Edward Cui, Sining Wei, Taroon Bharti, Ying Qiao, Jiun-Hung Chen, Winnie Wu, Shuguang Liu, Fan Yang, Daniel Campos, Rangan Majumder, and Ming Zhou. 2020. XGLUE: A new benchmark datasetfor cross-lingual pre-training, understanding and generation. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 6008–6018, Online. Association for Computational Linguistics.

Linlin Liu, Xin Li, Ruidan He, Lidong Bing, Shafiq Joty, and Luo Si. 2021. Knowledge based multilingual language model. *arXiv preprint arXiv:2111.10962*.

Yinhan Liu, Jiatao Gu, Naman Goyal, Xian Li, Sergey Edunov, Marjan Ghazvininejad, Mike Lewis, and Luke Zettlemoyer. 2020. Multilingual denoising pre-training for neural machine translation.

Minh-Thang Luong, Hieu Pham, and Christopher D Manning. 2015. Bilingual word representations with monolingual quality in mind. In *Proceedings of the 1st Workshop on Vector Space Modeling for Natural Language Processing*, pages 151–159.

Christopher D. Manning, Mihai Surdeanu, John Bauer, Jenny Finkel, Steven J. Bethard, and David McClosky. 2014. The Stanford CoreNLP natural language processing toolkit. In *Association for Computational Linguistics (ACL) System Demonstrations*, pages 55–60.

Tomas Mikolov, Quoc V Le, and Ilya Sutskever. 2013. Exploiting similarities among languages for machine translation. *arXiv preprint arXiv:1309.4168*.

George A Miller. 1998. *WordNet: An electronic lexical database*. MIT press.

George A Miller, Claudia Leacock, Randee Tengi, and Ross T Bunker. 1993. A semantic concordance. In *Proceedings of the workshop on Human Language Technology*, pages 303–308. Association for Computational Linguistics.

Tasnim Mohiuddin, M Saiful Bari, and Shafiq Joty. 2020. LNMap: Departures from isomorphic assumption in bilingual lexicon induction through non-linear mapping in latent space. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 2712–2723, Online. Association for Computational Linguistics.

Andrea Moro and Roberto Navigli. 2015. Semeval-2015 task 13: Multilingual all-words sense disambiguation and entity linking. In *Proceedings of the 9th international workshop on semantic evaluation (SemEval 2015)*, pages 288–297.

Ndapa Nakashole and Raphael Flauger. 2018. Characterizing departures from linearity in word translation. *arXiv preprint arXiv:1806.04508*.

Roberto Navigli, David Jurgens, and Daniele Vannella. 2013. Semeval-2013 task 12: Multilingual word sense disambiguation. In *Second Joint Conference on Lexical and Computational Semantics (* SEM), Volume 2: Proceedings of the Seventh International Workshop on Semantic Evaluation (SemEval 2013)*, pages 222–231.

Arvind Neelakantan, Jeevan Shankar, Alexandre Passos, and Andrew McCallum. 2014. Efficient non-parametric estimation of multiple embeddings per word in vector space. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1059–1069, Doha, Qatar. Association for Computational Linguistics.

Aitor Ormazabal, Mikel Artetxe, Gorka Labaka, Aitor Soroa, and Eneko Agirre. 2019. Analyzing the limitations of cross-lingual word embedding mappings. *arXiv preprint arXiv:1906.05407*.

Barun Patra, Joel Ruben Antony Moniz, Sarthak Garg, Matthew R Gormley, and Graham Neubig. 2018. Bliss in non-isometric embedding spaces.

Matthew E. Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer. 2018. Deep contextualized word representations. In *Proc. of NAACL*.

Matthew E Peters, Mark Neumann, Robert L Logan IV, Roy Schwartz, Vidur Joshi, Sameer Singh, and Noah A Smith. 2019. Knowledge enhanced contextual word representations. *arXiv preprint arXiv:1909.04164*.

Mohammad Taher Pilehvar and Nigel Collier. 2016. De-conflated semantic representations. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 1680–1690, Austin, Texas. Association for Computational Linguistics.

Telmo Pires, Eva Schlinger, and Dan Garrette. 2019. How multilingual is multilingual bert? *arXiv preprint arXiv:1906.01502*.

Sameer Pradhan, Edward Loper, Dmitriy Dligach, and Martha Palmer. 2007. Semeval-2007 task-17: English lexical sample, srl and all words. In *Proceedings of the fourth international workshop on semantic evaluations (SemEval-2007)*, pages 87–92.

Peter Prettenhofer and Benno Stein. 2010. Cross-language text classification using structural correspondence learning. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics*, pages 1118–1127, Uppsala, Sweden. Association for Computational Linguistics.

Alessandro Raganato, Jose Camacho-Collados, and Roberto Navigli. 2017. Word sense disambiguation: A unified evaluation framework and empirical comparison. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 1, Long Papers*, pages 99–110, Valencia, Spain. Association for Computational Linguistics.

Sascha Rothe and Hinrich Schütze. 2015. AutoExtend: Extending word embeddings to embeddings for synsets and lexemes. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 1793–1803, Beijing, China. Association for Computational Linguistics.

Erik F Sang and Fien De Meulder. 2003. Introduction to the conll-2003 shared task: Language-independent named entity recognition. *arXiv preprint cs/0306050*.

Tal Schuster, Ori Ram, Regina Barzilay, and Amir Globerson. 2019. Cross-lingual alignment of contextual word embeddings, with applications to zero-shot dependency parsing. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 1599–1613, Minneapolis, Minnesota. Association for Computational Linguistics.

Samuel L Smith, David HP Turban, Steven Hamblin, and Nils Y Hammerla. 2017. Offline bilingual word vectors, orthogonal transformations and the inverted softmax. *arXiv preprint arXiv:1702.03859*.

Benjamin Snyder and Martha Palmer. 2004. The english all-words task. In *Proceedings of SENSEVAL-3, the Third International Workshop on the Evaluation of Systems for the Semantic Analysis of Text*, pages 41–43.

Anders Søgaard, Sebastian Ruder, and Ivan Vulić. 2018. On the limitations of unsupervised bilingual dictionary induction. *arXiv preprint arXiv:1805.03620*.

Simon Šuster, Ivan Titov, and Gertjan Van Noord. 2016. Bilingual learning of multi-sense embeddings with discrete autoencoders. *arXiv preprint arXiv:1603.09128*.

Chih-chan Tien and Shane Steinert-Threlkeld. 2022. Bilingual alignment transfers to multilingual alignment for unsupervised parallel text mining. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 8696–8706, Dublin, Ireland. Association for Computational Linguistics.

Erik F. Tjong Kim Sang. 2002. Introduction to the CoNLL-2002 shared task: Language-independent

named entity recognition. In *COLING-02: The 6th Conference on Natural Language Learning 2002 (CoNLL-2002)*.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in neural information processing systems*, pages 5998–6008.

Zirui Wang, Jiateng Xie, Ruochen Xu, Yiming Yang, Graham Neubig, and Jaime G. Carbonell. 2020. Cross-lingual alignment vs joint training: A comparative study and a simple unified framework. In *International Conference on Learning Representations*.

Ralph Weischedel, Martha Palmer, Mitchell Marcus, Eduard Hovy, Sameer Pradhan, Lance Ramshaw, Nianwen Xue, Ann Taylor, Jeff Kaufman, Michelle Franchini, et al. 2013. Ontonotes release 5.0 ldc2013t19. *Linguistic Data Consortium, Philadelphia, PA*, 23.

Adina Williams, Nikita Nangia, and Samuel Bowman. 2018. A broad-coverage challenge corpus for sentence understanding through inference. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 1112–1122. Association for Computational Linguistics.

Shijie Wu and Mark Dredze. 2019. Beto, bentz, becas: The surprising cross-lingual effectiveness of bert. *arXiv preprint arXiv:1904.09077*.

Chao Xing, Dong Wang, Chao Liu, and Yiye Lin. 2015. Normalized word embedding and orthogonal transform for bilingual word translation. In *Proceedings of the 2015 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 1006–1011.

Wenxuan Zhang, Ruidan He, Haiyun Peng, Lidong Bing, and Wai Lam. 2021. Cross-lingual aspect-based sentiment analysis with aspect term code-switching. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 9220–9230, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Zheng Zhang, Ruiqing Yin, Jun Zhu, and Pierre Zweigenbaum. 2019. Cross-lingual contextual word embeddings mapping with multi-sense words in mind. *arXiv preprint arXiv:1909.08681*.

A Prediction tasks of language models

Next token prediction and masked token prediction are two common tasks in neural language model (LM) pretraining. We take two well-known language models, ELMo and BERT, as examples to illustrate these two tasks, which are shown in Figure 3.

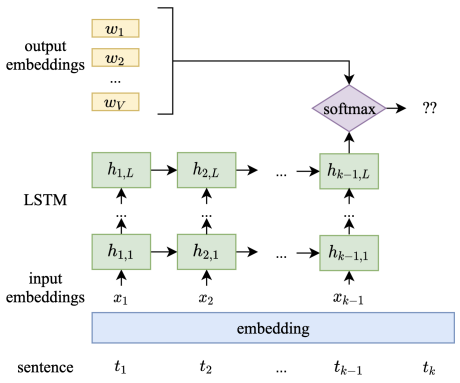

(a) Next token prediction.

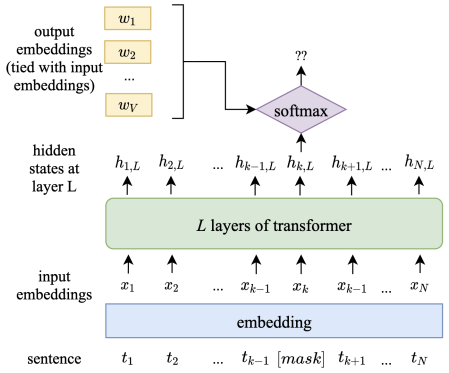

(b) Masked token prediction.

Figure 3: Next token and masked token prediction tasks of language models. For simplicity, we only show the forward language model in next token prediction.

B Further alignment (optional)

Applying the linear projection approach proposed by Schuster et al. (2019) on top of our framework can further improve cross-lingual transfer on some tasks. After our cross-lingual model is finished training on the concatenated corpora of two languages, L_1 and L_2, it is used to generate contextual token embeddings for the word pairs in the seed dictionary $\mathcal{D} = \{(t_i^{L_1}, t_i^{L_2})\}_{i=1}^{|\mathcal{D}|}$ [6]. Then, we compute the average of all contextual embeddings for each token $t_i^{L_j}$, denoted by $\boldsymbol{a}_i^{L_j}$. Finally, a linear projection matrix $\boldsymbol{W} \in \mathbb{R}^{d \times d}$ is learned to minimize cross-lingual embedding distance:

$$\boldsymbol{W} = \arg\min_{\boldsymbol{W}} \sum_{i=1}^{|\mathcal{D}|} ||\boldsymbol{W}\boldsymbol{a}_i^{L_1} - \boldsymbol{a}_i^{L_2}||^2 \quad (8)$$

[6] If any token t_k appears in both languages, we add that as an entry (t_k, t_k) to the dictionary as well.

C Pretraining details

C.1 Monolingual model

All the monolingual models were trained for one million steps. For better sense clustering performance, the maximum number of senses (S in word sense selection algorithm) was set to 1 for the first 20,000 steps to quickly get a reasonable initial model, and then increased to 5 afterwards when pretraining SaELMo and SaBERT-Tiny, which is controlled by hyperparameter *n_context* in our implementation. For SaELMo, we set *n_context* to 6, so that the model initialize 6 senses for each token, but only use the first sense in the 20,000 steps, and then use the other 5 senses (the first sense will be disabled) afterwards. We implement this for SaBERT-Tiny in a slightly different way, where *n_context* can be set to 5 directly to achieve the same effect. We use two NVIDIA V100 GPUs to pretrain SaELMo, which takes about 15 days to complete training. We use one NVIDIA V100 GPU to pretrain SaBERT-Tiny, which takes about 5 days. See Tables 6 and 7 for the hyperparameters used to pretrain SaELMo and SaBERT-Tiny respectively.

Hyperparameter	Value
max_word_length	50
batch_size	256
n_gpus	2
bidirectional	True
char_cnn:embedding:dim	16
char_cnn:max_characters_per_token	50
char_cnn:n_characters	261
char_cnn:n_highway	2
dropout	0.1
lstm:cell_clip	3
lstm:dim	4096
lstm:n_layers	2
lstm:proj_clip	3
lstm:projection_dim	512
lstm:use_skip_connections	True
all_clip_norm_val	10.0
n_epochs	10
unroll_steps	16
n_negative_samples_batch	8192
n_context	6
cluster_proj_dim	16
pca_sample	20,000
remove_less_freqent_contexts	0.1
learning_rate	0.2
sense_learning_rate	0.01

Table 6: Monolingual model hyperparameters: SaELMo.

Hyperparameter	Value
attention_probs_dropout_prob	50
directionality	bidi
hidden_act	gelu
hidden_dropout_prob	0.1
hidden_size	512
initializer_range	0.02
intermediate_size	2048
max_position_embeddings	512
num_attention_heads	8
num_hidden_layers	4
pooler_fc_size	512
pooler_num_attention_heads	8
pooler_num_fc_layers	3
pooler_size_per_head	128
pooler_type	first_token_transform
type_vocab_size	2
vocab_size	27654
n_context	5
context_rep_lr	0.01
pca_dim	14
contextual_warmup	20,000

Table 7: Monolingual model hyperparameters: SaBERT-Tiny.

C.2 Bilingual model

As metioned in the paper, we use Wikipedia dump to pretrain the bilingual models. The Stanford CoreNLP tokenizer (Manning et al., 2014) is used to tokenize English, German, Spanish and Chinese data. And the spaCy tokenizer is used to tokenize Japanese data. All data are converted to lowercase. We convert Chinese data to simplified font to make it consistent with evaluation task datasets.

All the language models used in cross-lingual experiments were pretrained for 600,000 steps from scratch. Similar to our monolingual models, maximum number of senses (S in word sense selection algorithm) was set to 1 for the first 20,000 steps, and the increased to 3 afterwards when pretraining Bi-SaELMo and Bi-SaELMo+Proj.[7] We use two NVIDIA V100 GPUs to pretrain each Bi-SaELMo model, which takes about 10 days to complete the training. See Table 8 for the hyperparameters used to pretrain Bi-SaELMo/Bi-SaELMo+Proj.

D Visualization of sense vectors

We visualize[8] the sense vectors of each model in a two dimensional PCA, and show some examples in Figures 4 to 7. For our English monolingual model

[7] Theoretically, in a reasonable range, it is expected that a larger S would be more helpful to capture the fine-grained senses. However, due to limited computation power, we use only 3 here, and 5 for the monolingual models.

[8] We use the tensorflow embedding projector (https://projector.tensorflow.org/) for visualization.

Hyperparameter	Value
max_word_length	50
batch_size	256
n_gpus	2
bidirectional	True
char_cnn:embedding:dim	16
char_cnn:max_characters_per_token	50
char_cnn:n_characters	261
char_cnn:n_highway	2
dropout	0.1
lstm:cell_clip	3
lstm:dim	4096
lstm:n_layers	2
lstm:proj_clip	3
lstm:projection_dim	512
lstm:use_skip_connections	True
all_clip_norm_val	10.0
n_epochs	6
unroll_steps	12
n_negative_samples_batch	8192
n_context	4
cluster_proj_dim	16
pca_sample	20,000
remove_less_freqent_contexts	0.1
learning_rate	0.2
sense_learning_rate	0.01

Table 8: Bilingual model hyperparameters: Bi-SaELMo/Bi-SaELMo+Proj.

(SaELMo), the vectors close to two different sense vectors of the word *may* are shown in (a) and (b) of Figure 4, respectively. We observe that senses are well clustered in these two subfigures, where cluster (a) corresponds to "month", and cluster (b) corresponds to "auxiliary verb".

We do the same for the English-Japanese bilingual model (Bi-SaELMo, without projection), and show the vectors close to two different sense vectors of the English word *bank* in (c) and (d) of Figure 5. We can see both English and Japanese sense vectors (*trade*, 銀行, 証券, etc.) in (c), most of which correspond to the sense "organization", though there are some noises. Similarly, most of the sense vectors in (d) correspond to sense "river bank".

Another two examples are shown in Figures 6 and 7. Our framework exhibits good sense clustering and sense level cross-lingual alignment behaviour in these examples. All sense vectors are dumped at training step 200,000, which is before pretraining complete.

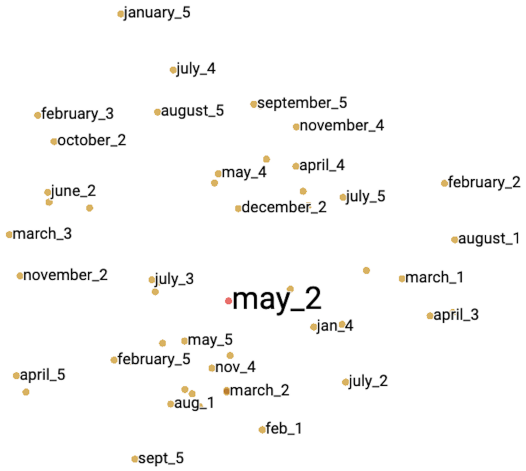

(a) "may" for month

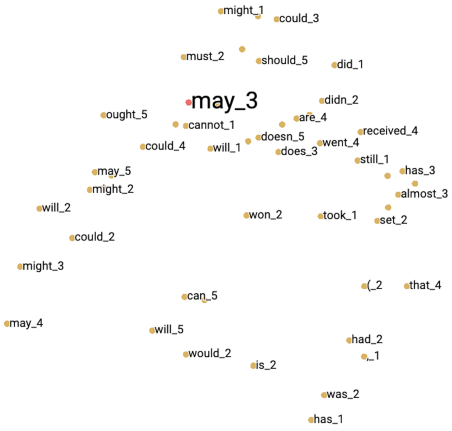

(b) "may" as auxiliary verb

Figure 4: We visualize sense vectors of English monolingual model (SaELMo) in a two dimensional PCA, and show the vectors close to two different sense vectors of word *may* in (a) and (b).

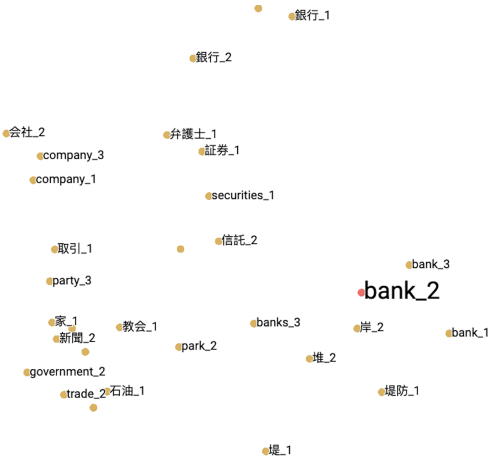

(a) "bank" for organization

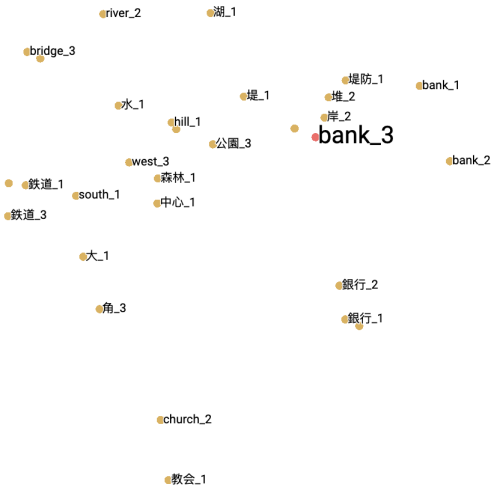

(b) "bank" for river bank

Figure 5: We visualize all sense vectors of *en-jp* bilingual model (Bi-SaELMo) in a two dimensional PCA, and show the vectors close to two different sense vectors of word *bank*.

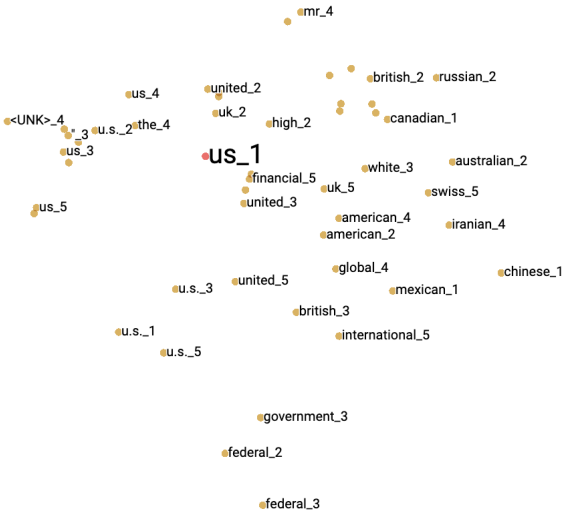

(a) "us" for country

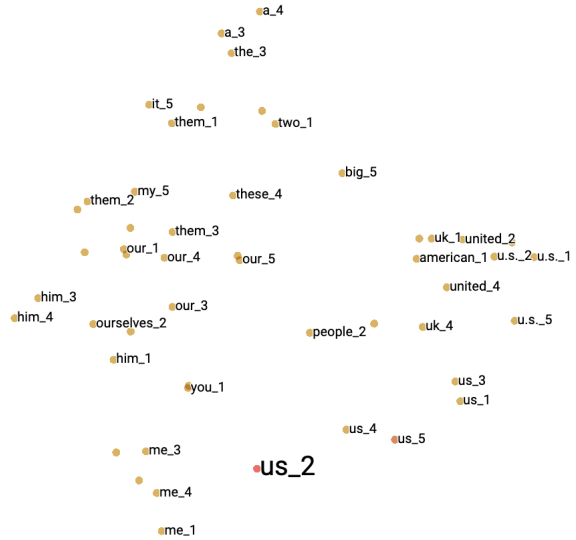

(b) "us" as pronoun

Figure 6: We visualize sense vectors of English monolingual model (SaELMo) in a two dimensional PCA, and show the vectors close to two different sense vectors of word *us* in (a) and (b).

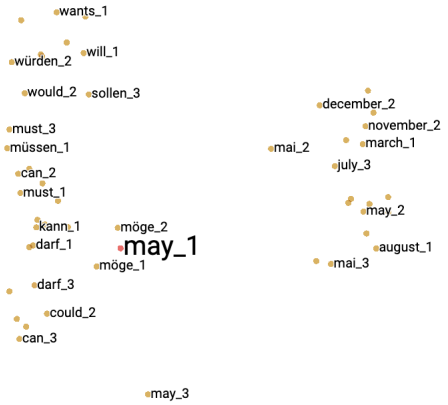

(a) "may" as auxiliary verb

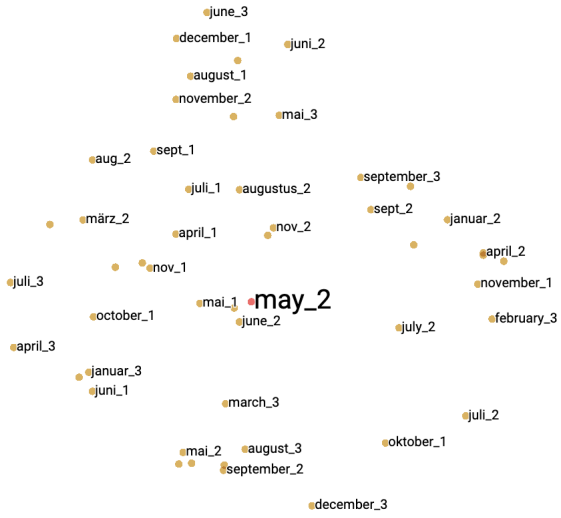

(b) "may" for month

Figure 7: We visualize all sense vectors of *en-de* bilingual model (Bi-SaELMo) in a two dimensional PCA, and show the vectors close to two different sense vectors of word *may*.

How to Parse a Creole: When Martinican Creole Meets French

Ludovic Mompelat
Indiana University
lmompela@iu.edu

Daniel Dakota
Indiana University
ddakota@iu.edu

Sandra Kübler
Indiana University
skuebler@indiana.edu

Abstract

We investigate methods to develop a parser for Martinican Creole, a highly under-resourced language, using a French treebank. We compare transfer learning and multi-task learning models and examine different input features and strategies to handle the massive size imbalance between the treebanks. Surprisingly, we find that a simple concatenated (French + Martinican Creole) baseline yields optimal results even though it has access to only 80 Martinican Creole sentences. POS embeddings work better than lexical ones, but they suffer from negative transfer.

1 Introduction

Syntactic analysis is an essential task for language documentation and language revitalization, as it allows a deeper understanding of languages. Under-resourced languages often suffer from the lack of annotated gold standard data available to develop and offer NLP solutions for the communities speaking the language. Moreover, there is a low number of researchers trained in formal linguistics and/or linguistic annotations which causes additional challenges in the creation of language resources for such languages. In recent years, research on parsing has developed a focus on under-resourced languages (Agić et al., 2016; Vania et al., 2019; Meechan-Maddon and Nivre, 2019), but creoles have received less attention.

In this study, we develop a dependency parsing model for Martinican Creole (MC), a French-based Creole, mostly spoken in Martinique, a French island in the Caribbean. Being a French territory, French and MC coexist in an unbalanced manner. The diglossic situation makes French the dominant language in many contexts, although in the past decades Martinican Creole has seen an expansion of its communicative contexts (Bernabé, 2004; Véronique, 2020). This is due to the codification and standardization processes the language underwent, especially by the GEREC (1982), and to the linguistic policies developed in an effort to safeguard and revitalize this language. However, one aspect of this revitalization process that is currently missing is the expansion of NLP tools and resources for MC (and other creole languages). Creole languages based on the same lexifier language (in our case French) are extremely similar and in many cases, mutually intelligible. Thus, developing NLP solutions for one creole language provides a basis to transfer knowledge to other related Creole languages.

The goal of this project is to investigate the best methods for developing a parser for an extremely low resource language when this language is a creole language. In our case, the creole is Martinican Creole. The main question here is whether the lexifier language, i.e., French, is similar enough to serve as basis for training a parser without further modification.

2 Research Questions

Our overarching research question is the following: Can we leverage a French treebank using transfer learning or multi-task learning approaches to create a parser model for Martinican Creole given that we only have a very small treebank ? Can we leverage the similarity between the creole and its lexifier language, French?

To answer this question, we need to answer the following questions:

1. Which types of embeddings can be used? Given the differences in spelling, are character embeddings, POS embeddings, or BERT embeddings the best representation of the input sentence? (We will not consider multilingual BERT models since the closest language is French, and we have access to large French embeddings models.)

2. In a transfer learning setting, how do we best

Proceedings of the 29th International Conference on Computational Linguistics, pages 4397–4406
October 12–17, 2022.

use the very limited Martinican Creole data? Is it worth the effort to annotate data for optimizing the parser, or can we optimize it on French? Is there enough structural and lexical similarity between French and the creole to make this possible?

3. In a transfer learning setting, how do we deal with the extreme imbalance between the large French Treebank and the small Martinican Creole Treebank? Can we prevent the parser from overfitting?

4. Can we leverage a multi-task learning model to handle the imbalance between French and the creole? More specifically, will loss weighting be able to counterbalance the treebank sizes?

5. Can we determine the linguistic characteristics of Martinican Creole that provide challenges to parsers based on standard transfer learning and on multi-task learning?

3 Related Work

Creoles are still under-researched in NLP. Noticeable work includes language model comparisons by Lent et al. (2021) between Haitian Creole, Nigerian Pidgin English, and Singaporean Colloquial English, trained with empirical risk minimization, against language models with distributionally robust ones, finding that the former performed better for Creoles. One reason postulated may be the absence of drift due to the relative stability of creoles.

Regarding French-based creoles, Haitian Creole was the subject of an extensive collaboration in Machine Translation led by Microsoft Research (Lewis, 2010) following the 2010 earthquake. Millour and Fort (2018) led a project of crowdsourcing of POS tags for Guadelupean Creole in which they describe the necessary steps and methodology to crowdsource a language for POS tagging. They were able to collect a corpus of nearly 2,500 tokens POS tagged and create a POS tagger reaching 84% accuracy.

The lack of available creole treebanks, with Nigerian Pidgin English (Caron et al., 2019) the only publicly available Universal Dependency treebank, means that best parsing strategies for Creoles are still being developed. Given the lack of available data, parsing creoles can be viewed as similar to the need to leverage related treebanks to try and increase performance on the target treebank. A common approach is to concatenate available treebanks and optimize towards the target treebank. This has demonstrated gains in both monolingual (Björkelund et al., 2017; Velldal et al., 2017) and cross-lingual (Das et al., 2017) experiments. Another successful technique is to instead train a model on a source treebank and then fine-tune on the target treebank (Shi et al., 2017; Che et al., 2017).

The most directly related works to ours are Wang et al. (2017, 2019) since they parse Singlish, an English-based Creole, by leveraging its lexifier language, English, to boost performance. Wang et al. (2017) propose a neural stacking architecture which yielded promising results which were further investigated by Wang et al. (2019). They tripled the size of their original Singlish treebank by web scraping and annotating more data and performed additional multi-task experiments for integrating English syntactic knowledge. While multi-task models showed some success, neural stacking methods were still better, as was simply concatenating English and Singlish treebanks in some experiments. Such neural stacking architectures with additional POS information also have helped in the related task of parsing Hindi-English Code-switching data (Bhat et al., 2018). As far as we know, we are the first to approach the task of dependency parsing a French-based Creole.

4 Properties of Martinican Creole

Martinican Creole (MC) is a French-based creole and part of the Atlantic Creoles language family. Syntactically, MC is an SVO language and is closely related to French, other creoles such as Guadeloupean, Marie Galante, St. Barth, Saint Lucian Creoles, and Haitian Creole, and to a lesser degree to African languages. The differences between MC and the closely related Antillean creoles are mostly lexical, they share very similar syntactic structures.

While MC originates from French, both languages show noticeable syntactic differences, especially wrt. the word order in noun phrases.

(1) Zanmi-mwen enmen liv-la (MC)
friends-my like book-the
Mes amis aiment le livre (French)
"My friends like the book"

Example (1) shows a sentence in Martinican Cre-

ole. It demonstrates that determiners like *-la* and modifying pronouns like *-mwen* are post-posed, compared to their French and English counterparts *mes* (my) and *le* (the).

Despite these differences in morpheme order, it is still relatively easy to see the direct parallels between both languages. This makes French a good candidate for a transfer learning approach to parsing MC.

MC is considered a morphologically reduced language (Hazaël-Massieux, 2002): Tense, mood and aspect features are expressed as separate morphemes/markers instead of inflections on the verbal element. There is also no morphological gender/number marking on nouns and adjective.

(2) Asiparé yo **té ké** vann prop
Apparently 3PL **PST.FUT** sell own
frè-***yo*** épi sè-***yo***.
brothers-3PL and sisters-3PL
A ce qu'il parait, ils vend**raient** ***leurs*** propres frères et ***leurs*** soeurs.
"Apparently, they would sell their own siblings"

In example (2), we see that the conditional is expressed in MC by a morpheme combination of the Past/Perfective marker *té* and the Future/Irrealis marker *ké* whereas in French, the conditional is expressed synthetically by the affix *-raient* attached to the end of the verb *vendre*. We also see that general plural nouns like *frè-yo* and *sè-yo* are not morphologically marked in MC, and neither is their accompanying adjective *prop*, whereas in French *frères* and *soeurs* and *propres* are all morphologically marked for gender and number.

Finally, while MC uses a different spelling system from French, the MC pronunciation is much closer to its spelling than in French. MC acquired most of its lexicon from French. Lexical transfer was either phonetically transparent or underwent reanalysis via several phono-lexical processes (such as agglutination (see example (3)), apheresis (see example (4)), syncope (see example (5)), etc.).

(3) **Agglutination**
diri [di.ʁi] (MC)
du riz [dy.ʁi] (French)
(some) rice

(4) **Apheresis**
limen [li.mɛ̃] (MC)
allumer [a.ly.me] (French)
to turn on

(5) **Syncope**
dòmi [dɔ.mi] (MC)
do**r**mir [dɔ.ʁmi] (French)
To sleep

In both cases, while the lexical transfer can easily be identified at the phonetic level, it is a more difficult to identify at the orthographic level, since there are significant differences in the respective spelling systems. Because of the amount of differences, it is possible that French embeddings may not be useful, since there may not be enough lexical overlap between French and MC, even on the subword level.

5 Methodology

5.1 Treebanks

French Treebank For our source treebank, we use the French GSD treebank (Guillaume et al., 2019)[1] as it is sufficiently large in size and predominantly consists of news articles, which aligns better with the newly created MC treebank.

MC Treebank The MC treebank consists of news and blog articles written in Martinican Creole by native speakers. Texts range from 2004 to 2021 and consist of two primary sources: 1) *Kréyolad*[2] collections which gather all the article contributions of Jude Duranty to the newspaper *Antilla*[3] from 2004 to 2018 and 2) the collective blog *Montray Kréyol*[4] which contains columns from numerous authors, written in French and various (mostly French-based) creoles. Selected text were annotated by the first author. The fully annotated treebank of MC consists of 240 sentences and a total of 4809 tokens.[5]

Annotation of MC Treebank We tokenized the texts using NLTK Tokenizer[6] and then annotated for POS information using INCePTION (Klie et al., 2018). INCePTION proposes an automatic POS tagger training on the annotations one makes synchronously and retrains itself whenever a new word receives a tag. We then used UD Annotatrix (Tyers

[1] Experiments training with all French treebanks were computationally more expensive and yielded poorer results.

[2] https://www.potomitan.info/duranty/kreyolad.php

[3] https://antilla-martinique.com/

[4] https://www.montraykreyol.org/

[5] The treebank will be released in the next UD cycle.

[6] https://www.nltk.org/api/nltk.tokenize.html We used the default model (English) since we did not expect any differences in punctuation.

Treebank	Train	Dev	Test
FR-GSD	13 072	1 634	1 634
MC	80	80	80

Table 1: Distribution in Train/Dev/Test sets of FR-GSD and Martinican Creole (MC) treebanks.

et al., 2017) for the dependency annotations. The treebank is not annotated for lemmas or morphological information.

5.2 Experimental Setup

Data Splits Due to the small size of the MC corpus, we split the treebank into equal size folds for train, dev, and test of 80 sentences. For more generalized results, we generate three different randomized splits and report results averaged over the three runs. For the French GSD treebank, we use the standard train/dev/test split, unless otherwise noted. Table 1 shows the sizes of the different data sets.

Parser We use the Deep Biaffine parser (Dozat and Manning, 2017) implemented in the SuPar parsing library.[7] The parser is a neural graph-based dependency parser which uses biaffine attention and biaffine classifier in combination with dimension reducing MLP layers to reduce non-relevant information.

We experiment with different input embeddings: character, POS tag, and BERT embeddings. Note that SuPar always includes word embeddings, so that we can only use (word+)POS and (word+)BERT. For all POS embeddings, we use gold POS tags. For the BERT embeddings, we use the French camemBERT (Martin et al., 2020)[8].

In addition, we also use a multi-task learning parser where each treebank is treated as a separate task (Sayyed and Dakota, 2021). Both input embeddings into the BiLSTM and the subsequent MLP layers are shared, which allows for information transfer during joint optimization between the treebanks. We also experiment with weighting treebanks with respect to their joint loss contribution, which has shown to be beneficial when data imbalances exist between treebanks (Dakota et al., 2021), as in our case. Results reported are using the scorer from CoNLL2018 shared task (Zeman et al., 2018).

[7] https://github.com/yzhangcs/parser

[8] We also experimented with the other large French LM, FlauBERT (Le et al., 2020), but this yielded worse performance

Train	Embed.	UAS	LAS
French	char	25.05	11.73
	POS	65.08	51.95
	BERT	38.23	21.63
MC	char	62.89	48.36
	POS	71.71	62.86
	BERT	63.36	49.83
FR+MC	char	72.95	60.57
	POS	80.75	71.77
	BERT	72.17	58.57

Table 2: Baselines for training on French, Martinican (MC), and concatenated French+Martinican (FR+MC).

6 Results

6.1 Baselines

We first need to establish the baselines, i.e., training on the French training set, training on the Martinican Creole training set, and concatenating these two. Here, we optimize and test on the MC dev set.

Table 2 shows the results for these baseline models. These results show that the French training data gives us the lowest results. The best model, using POS embeddings, results in an LAS of 51.95. Using character and BERT embeddings results in considerable losses (LAS: 11.73 and 21.63); this can be attributed to the significant differences in spelling between French and MC (see section 4). Training on 80 MC sentences is surprisingly successful. Again, using the POS embeddings shows the best results (LAS: 62.86). It is worth noting how beneficial the use of POS embeddings is for MC compared to subword information. One reason is simply the small data size of the MC treebank; another reason may be that some of the linguistic properties of MC are disambiguated via POS tags but not via characters. However, the concatenation of both training sets results in the highest scores overall, with an LAS of 71.77 for POS embeddings. This is particularly interesting given that the French training size is about 136 times the size of the MC training but this small amount is enough to direct the French-trained model in a beneficial direction.

6.2 Optimization

Since we operate in a very low-resource setting, the next question is whether it is worth annotating sentences to use for optimizing the parser or whether the neural architecture does not require target language specific optimization. Thus, we compare a

Dev	Embed.	UAS	LAS
French	char	20.03	9.16
	POS	58.17	45.54
	BERT	33.07	18.38
MC	char	25.05	11.73
	POS	65.08	51.95
	BERT	38.23	21.63

Table 3: MC test performance when optimizing on French and MC.

Dev.	Embed.	Finet.	UAS	LAS
French	char	no	20.03	9.16
		yes	20.03	9.16
	POS	no	58.17	45.54
		yes	58.17	45.54
	BERT	no	33.07	18.38
		yes	33.07	18.38
MC	char	no	25.05	11.73
		yes	64.71	46.83
	POS	no	65.08	51.95
		yes	72.87	60.83
	BERT	no	38.23	21.63
		yes	67.10	49.41

Table 4: Performance with and without fine-tuning on MC.

setting trained and optimized on French with a setting where we train on French and optimize on 80 MC sentences. The results are shown in Table 3.[9].

Our results show increases when the source French model has been optimized on MC as compared to French. This is true for all types of embeddings. The POS model shows a sizable improvement of more than 6 percent points for LAS while the improvements for the character and BERT models are more modest, around 2-3 percent points. However, we do not reach the MC baseline in any setting.

6.3 Fine-tuning

We next experiment with transfer learning in order to see if we can improve on the French baseline by fine-tuning on the MC training set. Given the difference in size, the MC data should not have a noticeable effect, but since the concatenation baseline proved so successful, we need to determine whether fine-tuning on the MC training has the same effect. When training on French, we have two settings: We either optimize on French or on MC. When fine-tuning on MC, we optimize on MC.

Table 4 shows the results of these experiments. Note that the results without fine-tuning are repeated from Table 3. The results show very clearly that fine-tuning is only successful when the first stage is optimized on MC. If we optimize that stage on French, fine-tuning does not result in any improvement. This is likely due to the fact that training a fully optimized French model results in overfitting, which in turns does not allow the little MC data to effectively update the parameters.

When optimizing on MC, we note that all models show a drastic improvement in performance, especially for the BERT and character embeddings. Out of the three types of embeddings, the model using character embeddings benefits the most from fine-tuning, improving from 25.05 to 64.71 for UAS and from 11.73 to 46.83 for LAS, followed by the BERT embeddings model going from 38.23 to 67.10 for UAS and 21.63 to 49.41 for LAS. The most successful model, using POS embeddings, reaches an LAS of 60.83. While this is still below the concatenation model, it shows again the usefulness of POS embeddings.

6.4 Overfitting

One reason for the lack of improvement of the model optimized on French in Table 4 may be that training a fully optimized French model results in overfitting, which in turn does not allow the little MC data to effectively update the parameters. To investigate the issue of overfitting, we perform experiments where we stop the training early. Since it is unclear how to determine good stopping points, we stopped the training at epoch 1 as well as at the 1/4, 1/2, and 3/4 of the optimal number of epochs when using the French model (trained and optimized on French) and perform fine-tuning experiments in two settings, fine-tuning the model on MC, and on the concatenated FR+MC treebank. In both cases, we optimize on MC. The results of these experiments are shown in Table 5.

When comparing between the two fine-tuning settings, we note that none of the 1/4, 1/2, 3/4 or the fully optimized models improve from the MC or MC+FR data during fine-tuning, as results are not substantially different from the ones without fine-tuning. This indicates that the more a model

[9]Note that the results of the French model optimized on MC are repeated from Table 2.

FT	Emb.	Epoch 1	1/4	1/2	3/4	Full
MC	char	41.59	9.41	9.38	9.32	9.16
MC	POS	36.82	49.81	48.79	47.23	45.54
MC	BERT	11.55	20.30	19.25	18.03	18.38
FR+MC	char	54.17	9.41	9.38	9.32	9.16
FR+MC	POS	65.33	49.81	48.79	47.23	45.54
FR+MC	BERT	11.55	20.30	19.25	18.03	18.38

Table 5: LAS when training on 1/4, 1/2, 3/4 of the best epoch of the French model, fine-tuned on MC or FR+MC.

Embed.	No weight		Weight	
	UAS	LAS	UAS	LAS
char	70.23	56.58	71.44	58.26
POS	64.67	50.46	64.99	50.12
BERT	69.39	56.33	70.76	56.78

Table 6: Results for MTL with non-weighted and weighted losses on the MC task. All weighted experiments use 0.9 for French and 0.1 for MC.

is trained and optimized on French, the less it is able to profit from having access to MC data. The only models showing noticeable benefit from fine-tuning are the epoch 1 character model fine-tuned on MC and the epoch 1 character and POS models fine-tuned on FR+MC, but both are still below their respective baselines.

When we look at the experiments with fewer epochs, we see a deterioration of the results from fewer epochs to the full number of epochs, showing clear signs of overfitting. This trend holds across all conditions but is strongest for the highest performing model using POS embeddings. Here the LAS decreases from 49.81 to 45.54. However, even the results at 1/4 epochs are far below the MC baseline.

6.5 Multi-task Learning

Another approach for information sharing is to use multi-task learning (MTL). By treating each treebank as a task, it allows them to be optimized jointly but does so by combining information with the other treebank in the process rather than sequentially as in a typical transfer learning setup. For this experiment, we have two settings, one without weighting losses, and one with loss weighting. Reducing the weights for the smaller treebank may help reduce the negative transfer that can occur. Given the small size of the MC training set, its contribution to the overall loss may be too high, leading the parser in a sub-optimal direction. This assumption has been shown to hold for a domain adaptation setting (Dakota et al., 2021), where assigning higher weights to the larger and lower weights to the smaller treebank yielded the best performance. Consequently, we assign a loss of 0.9 to the French treebank and 0.1 to the MC treebank.

The results of this experiment are shown in Table 6. Results are generally better than for the fine-tuning setting. However, the best result so far is still the baseline trained on only 80 sentences of MC and using POS (see Table 2), as none of the MTL settings reach this result. When we compare the weighted and non-weighted settings, we see an improvement of about 1.5 points (LAS: from 56.58 to 58.26) for the character model and a minimal gain for the BERT model (LAS: from 56.33 to 56.78), but a small decrease for the POS model. It is noticeable that using POS information leads to substantially worse results in comparison to the other models, thus contradicting the trends of previous experiments. This further re-enforces the notion of negative transfer when sharing POS information.

We next look at a setup where we use the FR+MC concatenated treebank as one of our tasks and the MC treebank as the other, with both optimizing on the same development set, but using different weights.f

Table 7 shows the results of this experiment (the FR/MC setting is repeated from Table 6). We see that using the combined FR+MC training set gives us a moderate boost of 2-3 percent points over the FR/MC setting. Here, the UAS improves over the best MC-only baseline, the LAS does not. Additionally, we can see that even further reducing the MC weights tends to yield better performance for LAS, suggesting that as the data imbalance becomes extreme, so does the need to downweight the smaller treebank.

Embeddings	Weights		UAS	LAS
	FR	MC		
char	0.90	0.10	71.44	58.26
POS	0.90	0.10	64.99	50.12
BERT	0.90	0.10	70.76	56.78
	FR+MC	MC		
char	0.90	0.10	73.13	59.84
POS	0.90	0.10	69.56	55.81
BERT	0.90	0.10	73.37	60.08
char	0.95	0.05	74.08	61.26
char	1.0	0.01	73.54	61.47
POS	0.95	0.05	70.77	57.66
POS	1.0	0.01	70.71	57.53
BERT	0.95	0.05	72.97	59.82
BERT	1.0	0.01	73.15	60.04

Table 7: Results for the MC task using varying weights for the MTL parser, training on either FR and MC or on FR+MC and MC and testing on MC.

7 Analysis

All the experiments described above tell us that the best method to parse Martinican Creole given a very small treebank is to concatenate the two treebanks. It is unclear why first training on French and then fine-tuning on MC does not result in a similar performance. And it is equally unclear why the POS embeddings are successful in transfer learning but not in a multi-task learning setting. We assume that the two are related and will analyze the data to shed light on these questions.

7.1 Correlation of POS Tags and Parser Errors

We first look into the correlation between specific part of speech tags and parser errors. More specifically we look at label accuracy of incoming arcs per POS tag, with a primary focus on the experiments that include the concatenated FR+MC data during training.

Table 8 presents the results[10] for the FR+MC embeddings baselines and their best respective MTL settings from Table 7. We see the same trends across most open and closed class POS tags within one setting. Since the lexical models (char and BERT) show significantly lower results than the POS model, this points to a disconnect on the lexical level (caused by the different spelling systems) that can only be overcome by adding POS information. However, in this case, we would expect a better performance of the POS model in our MTL task in comparison to the MC baseline. Since this does not happen, we assume that there are significant differences on the POS level between the two languages, causing negative transfer for the MTL model (one facet of this will be investigated in more detail in the next section.)

One notable trend is related to the accuracies for adjectives and adverbs: While the baseline POS model can parse those POS very successfully, the MTL POS model reaches accuracies that are below the MTL character and BERT models for adjectives and comparable for adverbs (again, see below for an explanation).

7.2 POS Distribution

We now have a closer look at the POS distributions between French and MC, to determine whether these ambiguity rates can give us insights into the differences between French and MC on the POS level. However, a direct comparison does not seem to be feasible since the MC treebank is too small to give us a stable picture, especially compared to the large French treebank. For this reason, we decided to use the full 240 sentences of the MC treebank and to randomly sample 240 sentences from the French treebank (averaged over 10 repetitions). While the small number will introduce some variability, the results will be more comparable across the languages.

When looking at the percentage of ambiguous words, 2.2% of the word types (in the POS lexicon) for French and 7.0% for MC are ambiguous, showing that about 3 times more MC words are ambiguous. Additionally, the percentage of ambiguous word types amounts to 13.0% when we concatenate the French and MC treebanks.

Table 9 shows the rates of ambiguous word types per POS tag. A comparison of French and MC shows that for all POS tags, the MC words are ambiguous about 3 times more often. And while French subordinating conjunctions (SConJ) and prepositions (Adp) tend to be frequently ambiguous, this ratio increases to more than 50% for MC. Additionally, the percentages for the combined treebank shows that the ambiguities are mostly additive, i.e., there is not much overlap between the ambiguous words in French and MC. This at least partly explains the difficulties of the POS models. The most extreme cases are subordinating conjunctions

[10] All numbers are averaged over the three folds.

	Noun	Verb	Adj	Propn	Adv	CConj	SConj	Adp	LAS
baseline char	56.76	63.67	50.36	60.82	57.38	66.80	60.27	68.49	60.57
baseline POS	70.52	72.03	84.45	66.78	87.21	88.12	91.34	89.23	71.77
baseline BERT	54.92	58.47	53.05	58.15	51.66	62.57	55.92	65.44	58.57
MTL char	58.03	64.45	50.90	58.53	62.64	71.08	65.38	66.55	61.47
MTL POS	53.00	58.61	45.35	56.93	59.72	68.49	65.01	68.56	57.66
MTL BERT	55.67	60.96	55.68	60.82	58.45	69.48	58.19	65.20	60.08

Table 8: Accuracy of dependency labels per POS tag for FR+MC baseline and best MTL experiments.

	Noun	Verb	Adj	Propn	Adv	CConj	SConj	Adp	Total
French	2.2%	3.5%	5%	0.5%	8.9%	8.4%	48.8%	18.6%	2.2%
MC	5.7%	9.5%	17.9%	1.7%	24.5%	37.5%	61.8%	51.8%	7.0%
FR+MC	14.5%	15.3%	34.6%	17.3%	35.8%	51.7%	70.6%	62.9%	13.0%

Table 9: Average of ambiguous word types per POS tag.

and prepositions, for which more than 50% are ambiguous in MC and more than 60% in the combined treebank. For the open class POS tags, adjectives and adverbs are the most affected. In the case of adjectives, the combined treebank shows a doubling of the ambiguity rate from MC to the combined treebank, thus indicating not only an increase in ambiguity in MC, but also a high number of words that are only considered adjectives in one of the languages but not both. This partly explains the low results for adjectives in the MTL POS setting in Table 8.

7.3 Example

Martinican creole shows a systematic ambiguity between nouns and adjectives. The word *politik* is one example, as shown in examples (6) and (7). In example (6) the word is misidentified as a noun, which leads the character model to interpret it as an *nmod* of *désizion* instead of its *amod* (see Figure 1). Having access to the gold POS tags in the POS model helps this model disambiguate it correctly.

(6) zot wè ni an désizion politik
2PL see there-is a decision political
"You saw that there is a political decision."

(7) fanm an politik
women in politics

8 Conclusion and Future Work

In this study, we built a first parser of Martinican Creole using French as the supporting language to address the extremely low-resource setting of the creole.

Our main finding is that, surprisingly, we obtain the best parsing results with our baseline model trained on a concatenation of the French and MC training sets. The success of the concatenated baseline model shows that even with as little as 80 MC sentences in the training set, the POS model is able to direct itself in the right direction.

Even the baseline POS model trained on 80 MC sentences outperforms all transfer and MTL models, the single exception being the UAS of the MTL character and BERT models. Partial explanations for these results can be found in the different spelling systems used for French and MC (see Section 4) and in the high level of ambiguity of MC, and specifically MC adjectives and adverbs. Whether POS tags are needed in neural dependency parsing is still an open question (Anderson and Gómez-Rodríguez, 2020; Zhou et al., 2020), and our findings further complicate this picture. In our case, they can reduce ambiguity in our baselines, but increase ambiguity across the two languages. Since we use gold POS tags, there remains the open question whether the same effects will occur with automatically annotated POS tags.

Our results partially contradict findings by Wang et al. (2019) for Singlish. They also found that in the low resource setting (using 900 Singlish sentences), treebank concatenation outperforms MTL. However in their work, MTL outperformed the baselines for both individual treebanks while we did not see this increase in performance across experiments. Our findings thus confirm that to improve our performances on parsing MC using French, we will need to reduce the imbalance between the two languages, by augmenting the MC data. For the future, we are planning to investi-

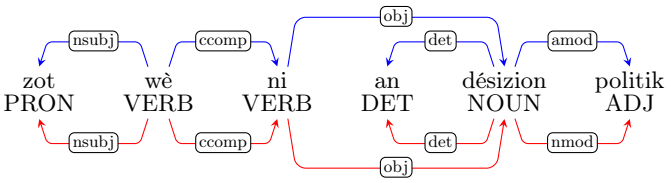

Figure 1: *zot wè ni an désizion politik* parses for POS and char predictions.

gate whether a larger MC training set will have a positive effect in the MTL setup.

However, the fact that as little as 240 annotated sentences, provided that we concatenate them with French data, can yield an LAS in the low 70es indicates that it is possible to develop parsing models for French-based creoles without extensive annotation projects.

References

Željko Agić, Anders Johannsen, Barbara Plank, Héctor Martínez Alonso, Natalie Schluter, and Anders Søgaard. 2016. Multilingual Projection for Parsing Truly Low-Resource Languages. *Transactions of the Association for Computational Linguistics*, 4:301–312.

Mark Anderson and Carlos Gómez-Rodríguez. 2020. On the frailty of universal POS tags for neural UD parsers. In *Proceedings of the 24th Conference on Computational Natural Language Learning*, pages 69–96, Online.

Jean Bernabé. 2004. Éléments d'écolinguistique appliqués à la situation martiniquaise. In *Créoles-Langages et Politiques linguistiques: Actes du XXVI e Colloque International de Linguistique Fonctionnelle-30 septembre-7 octobre 2002 à Gosier (Guadeloupe)*, pages 13–29. Peter Lang.

Irshad Bhat, Riyaz A. Bhat, Manish Shrivastava, and Dipti Sharma. 2018. Universal Dependency parsing for Hindi-English code-switching. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 987–998, New Orleans, LA.

Anders Björkelund, Agnieszka Falenska, Xiang Yu, and Jonas Kuhn. 2017. IMS at the CoNLL 2017 UD shared task: CRFs and perceptrons meet neural networks. In *Proceedings of the CoNLL 2017 Shared Task: Multilingual Parsing from Raw Text to Universal Dependencies*, pages 40–51, Vancouver, Canada.

Bernard Caron, Marine Courtin, Kim Gerdes, and Sylvain Kahane. 2019. A Surface-Syntactic UD Treebank for Naija. In *TLT 2019, Treebanks and Linguistic Theories, Syntaxfest*, Paris, France.

Wanxiang Che, Jiang Guo, Yuxuan Wang, Bo Zheng, Huaipeng Zhao, Yang Liu, Dechuan Teng, and Ting Liu. 2017. The HIT-SCIR system for end-to-end parsing of Universal Dependencies. In *Proceedings of the CoNLL 2017 Shared Task: Multilingual Parsing from Raw Text to Universal Dependencies*, pages 52–62, Vancouver, Canada.

Daniel Dakota, Zeeshan Ali Sayyed, and Sandra Kübler. 2021. Bidirectional domain adaptation using weighted multi-task learning. In *Proceedings of the 17th International Conference on Parsing Technologies and the IWPT 2021 Shared Task on Parsing into Enhanced Universal Dependencies (IWPT 2021)*, pages 93–105, Online.

Ayan Das, Affan Zaffar, and Sudeshna Sarkar. 2017. Delexicalized transfer parsing for low-resource languages using transformed and combined treebanks. In *Proceedings of the CoNLL 2017 Shared Task: Multilingual Parsing from Raw Text to Universal Dependencies*, pages 182–190, Vancouver, Canada.

Timothy Dozat and Christopher Manning. 2017. Deep biaffine attention for neural dependency parsing. In *5h International Conference on Learning Representations (ICLR 2017)*, Toulon, France.

GEREC. 1982. *Charte culturelle créole: se pwan douvan avan douvan pwan nou!* GEREC, Centre universitaire Antilles-Guyane.

Bruno Guillaume, Marie-Catherine de Marneffe, and Guy Perrier. 2019. Conversion et améliorations de corpus du français annotés en Universal Dependencies. *Revue TAL*, 60(2):71–95.

Marie-Christine Hazaël-Massieux. 2002. Les créoles à base française: une introduction. *Travaux Interdisciplinaires du Laboratoire Parole et Langage d'Aix-en-Provence (TIPA)*, 21:63–86.

Jan-Christoph Klie, Michael Bugert, Beto Boullosa, Richard Eckart de Castilho, and Iryna Gurevych. 2018. The INCEpTION platform: Machine-assisted and knowledge-oriented interactive annotation. In *Proceedings of the 27th International Conference on Computational Linguistics: System Demonstrations*, pages 5–9, Santa Fe, New Mexico.

Hang Le, Loïc Vial, Jibril Frej, Vincent Segonne, Maximin Coavoux, Benjamin Lecouteux, Alexandre Allauzen, Benoît Crabbé, Laurent Besacier, and Didier Schwab. 2020. FlauBERT: Unsupervised language

model pre-training for French. In *Proceedings of The 12th Language Resources and Evaluation Conference*, pages 2479–2490, Marseille, France.

Heather Lent, Emanuele Bugliarello, Miryam de Lhoneux, Chen Qiu, and Anders Søgaard. 2021. On language models for creoles. In *Proceedings of the 25th Conference on Computational Natural Language Learning*, pages 58–71, Online.

William Lewis. 2010. Haitian Creole: How to build and ship an MT engine from scratch in 4 days, 17 hours, & 30 minutes. In *Proceedings of the 14th Annual Conference of the European Association for Machine Translation*, Saint Raphaël, France.

Louis Martin, Benjamin Muller, Pedro Javier Ortiz Suárez, Yoann Dupont, Laurent Romary, Éric de la Clergerie, Djamé Seddah, and Benoît Sagot. 2020. CamemBERT: a tasty French language model. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, Online.

Ailsa Meechan-Maddon and Joakim Nivre. 2019. How to parse low-resource languages: Cross-lingual parsing, target language annotation, or both? In *Proceedings of the Fifth International Conference on Dependency Linguistics (Depling, SyntaxFest 2019)*, pages 112–120, Paris, France. Association for Computational Linguistics.

Alice Millour and Karën Fort. 2018. Krik: First Steps into Crowdsourcing POS tags for Kréyòl Gwadloupéyen. In *CCURL 2018* , Miyazaki, Japan.

Zeeshan Ali Sayyed and Daniel Dakota. 2021. Annotations matter: Leveraging multi-task learning to parse UD and SUD. In *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*, pages 3467–3481, Online.

Tianze Shi, Felix G. Wu, Xilun Chen, and Yao Cheng. 2017. Combining global models for parsing Universal Dependencies. In *Proceedings of the CoNLL 2017 Shared Task: Multilingual Parsing from Raw Text to Universal Dependencies*, pages 31–39, Vancouver, Canada.

Francis M. Tyers, Mariya Sheyanova, and Jonathan North Washington. 2017. UD Annotatrix: An annotation tool for Universal Dependencies. In *Proceedings of the 16th International Workshop on Treebanks and Linguistic Theories*, pages 10–17, Prague, Czech Republic.

Clara Vania, Yova Kementchedjhieva, Anders Søgaard, and Adam Lopez. 2019. A systematic comparison of methods for low-resource dependency parsing on genuinely low-resource languages. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 1105–1116, Hong Kong, China. Association for Computational Linguistics.

Erik Velldal, Lilja Øvrelid, and Petter Hohle. 2017. Joint UD parsing of Norwegian Bokmål and Nynorsk. In *Proceedings of the 21st Nordic Conference on Computational Linguistics*, pages 1–10, Gothenburg, Sweden.

Georges Daniel Véronique. 2020. Les langues créoles dans la république française: entre démarcation et revendication1. *Entre francisation et démarcation.: Usages hérités et usages renaissantistes des langues régionales de France*, page 201.

Hongmin Wang, Jie Yang, and Yue Zhang. 2019. From genesis to creole language: Transfer learning for Singlish Universal Dependencies parsing and POS tagging. *ACM Trans. Asian Low-Resour. Lang. Inf. Process.*, 19(1).

Hongmin Wang, Yue Zhang, GuangYong Leonard Chan, Jie Yang, and Hai Leong Chieu. 2017. Universal Dependencies parsing for colloquial Singaporean English. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics*, pages 1732–1744, Vancouver, Canada.

Daniel Zeman, Jan Hajič, Martin Popel, Martin Potthast, Milan Straka, Filip Ginter, Joakim Nivre, and Slav Petrov. 2018. CoNLL 2018 shared task: Multilingual parsing from raw text to Universal Dependencies. In *Proceedings of the CoNLL 2018 Shared Task: Multilingual Parsing from Raw Text to Universal Dependencies*, pages 1–21, Brussels, Belgium.

Houquan Zhou, Yu Zhang, Zhenghua Li, and Min Zhang. 2020. Is pos tagging necessary or even helpful for neural dependency parsing? In *Natural Language Processing and Chinese Computing*, pages 179–191, Cham. Springer International Publishing.

Byte-based Multilingual NMT for Endangered Languages

Mengjiao Zhang
Stevens Institute of Technology
mzhang49@stevens.edu

Jia Xu
Stevens Institute of Technology
jxu70@stevens.edu

Abstract

Multilingual neural machine translation (MNMT) jointly trains a shared model for translation with multiple language pairs. However, traditional subword-based MNMT approaches suffer from out-of-vocabulary (OOV) issues and representation bottleneck, which often degrades translation performance on certain language pairs. While byte tokenization is used to tackle the OOV problems in neural machine translation (NMT), until now its capability has not been validated in MNMT. Additionally, existing work has not studied how byte encoding can benefit endangered language translation to our knowledge. We propose a byte-based multilingual neural machine translation system (BMNMT) to alleviate the representation bottleneck and improve translation performance in endangered languages. Furthermore, we design a random byte mapping method with an ensemble prediction to enhance our model robustness. Experimental results show that our BMNMT consistently and significantly outperforms subword/word-based baselines on twelve language pairs up to +18.5 BLEU points, an 840% relative improvement.

1 Introduction

Neural Machine Translation (NMT) has achieved great success and dominates recent research on translation tasks in both academic and industry studies (Wu et al., 2016; Stahlberg, 2020; Chen et al., 2018). In particular, multilingual neural machine translation (MNMT) has been shown to benefit low-resource language translation by jointly training MNMT models with high-resource languages (Johnson et al., 2017). However, most existing NMT/MNMT models are based on word or subword tokenization, which has three main problems. First, language-specific tokenizers, such as BPE (Shaham and Levy, 2021), may introduce inaccurate segmentations (Banerjee and Bhattacharya, 2018). Second, out-of-vocabulary (OOV) words/subwords are still unavoidable and hurt translation performance. Third, some languages show a decrease in translation quality with multilingual training due to specific characteristics of the language variety, a problem known as the representation bottleneck (Dabre et al., 2020; Zoph and Knight, 2016). It essentially limits the improvement of transfer learning from high-resource NMT, like Chinese-English (Gu et al., 2018) to the low-resource NMT, like Aymara-Spanish.

Recently, byte-based NMT models show comparable performance to word/subword-based models (Shaham and Levy, 2021; Wang et al., 2020). Because modern byte encoding systems such as UTF-8 have only 256-byte entries in total, i.e., `0x00` to `0xff`, the unified byte tokenization in all languages obviates the traditional preprocessing of language-specific subword tokenization and restricts the vocabulary to a fixed and small one. Therefore, the 256-byte-sized UTF-8 vocabulary avoids the OOV issues.

Despite the advantages of byte tokenization, the byte encoding has not been investigated in multilingual NMT yet, to the best of our knowledge. In this paper, we show that byte tokenization can be naturally applied to MNMT systems with great advantages. Given the unified encoding of byte tokenization in all languages, byte encoding is able to address the representation bottleneck problem in MNMT systems effectively. For example, Figure 1 shows the overlaps of subword and byte vocabularies among multiple languages. We notice with the growth of the language number, there is almost no overlap of word vocabularies, while the byte vocabulary still has a large overlap. This large overlap of byte vocabulary enables enhanced knowledge sharing among different languages and can help the model learn more generalized representations for these languages. Taking the low resource as a common property for most endangered languages,

Proceedings of the 29th International Conference on Computational Linguistics, pages 4407–4417
October 12–17, 2022.

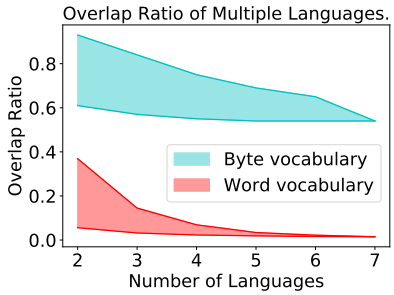

Figure 1: The vocabulary overlap ratio of byte and subword tokens. The area upper/lower bound denotes the highest/lowest ratio among all language combinations, given a language number.

we think byte-based MNMT is particularly helpful for endangered language translation. Therefore, we aim to incorporate byte tokenization in MNMT to alleviate the representation bottleneck problem in multilingual translation.

Besides incorporating byte encoding into MNMT, we aim to further investigate the generalizability of byte encoding. We observed that byte mapping can be arbitrary. For example, the characters "a"-"z" are represented with bytes 97-122 using UTF-8. However, we conjecture "a"-"z" can be any byte from 0 to 255. The byte representation does not need to be determined as a single encoding mapping. However, existing byte-based NMT systems do not consider such randomness of byte encoding. Moreover, we think language models should provide similar performance given different byte encoding methods to improve the generalizability and robustness. Therefore, we design a new encoding method that we call Random Byte Encoding by incorporating the random representation of bytes and reduce the variance of model outputs.

In this work, to address these challenges, we propose a **B**yte-based **M**ultilingual **N**eural **M**achine **T**ranslation framework (**BMNMT**). It simultaneously considers the byte randomness and the endangered languages in multilingual translation and works as follows. First, we design a novel MNMT framework that can take the byte encoding of sentences as inputs. Then, we incorporate the randomness of the byte encoding as discussed above by generating random byte mapping to replace the original byte ordering. We finally propose an ensemble prediction method by combining different encodings for reliable outputs.

Our BMNMT achieves amazing results in improving the low-resource and the endangered language translation that often does not have satisfactory results due to scarce resources and other linguistic characteristics (Levow et al., 2021; Ens et al., 2019; Liu et al., 2022). We demonstrate that our BMNMT consistently and significantly enhances the translation performance on all languages, including five high-resource languages, German, Arabic, Chinese, Farsi, Turkish to English, one low-resource language, Slovenian to English, and ten endangered languages, Asháninka, Aymara, Bribri, Guarani, Nahuatl, Otomí, Quechua, Rarámuri, Shipibo-Konibo, and Wixarika to Spanish. For example, the translation BLEU score [%] increases from 0 to 3.9 for the endangered language Shipibo-Konibo, and from 2.2 to 20.7 for the low-resource language Slovenian to English, i.e., +18 BLEU points. The contributions of this work are summarized as follows:

- We propose an effective byte-based MNMT framework to alleviate the representation bottleneck problem in word/subword-based multilingual translation, especially for endangered languages.
- We design a novel method of random byte encoding with ensemble prediction to enhance the generalizability and robustness of our byte-based MNMT model.
- We evaluate BMNMT on various training strategies. Extensive experiments validate the effectiveness, generalizability, and robustness of our model.

In the following context, we first outline the previous work in Section 2, then describe our method in Section 3, finally show our experimental results in Section 4 and conclude this work in Section 6.

2 Related Work

Multilingual Neural Machine Translation Word and subword-level tokenizations are widely used in natural language processing, including NMT/MNMT. Morishita et al. (Morishita et al., 2018) propose to incorporate hierarchical subword features to improve neural machine translation. Massively multilingual NMT models are proposed by Aharoni et al. (Aharoni et al., 2019) and Arivazhagan et al. (Arivazhagan et al., 2019). They are trained on massive language pairs and show a strong and positive impact on low-resource languages. However, these models tend to have

representation bottlenecks (Dabre et al., 2020), due to large vocabulary size and large diversity of training languages. Two MNMT systems (Tan et al., 2019; Pan et al., 2021) are proposed to solve this problem by modifying the model architectures, adding special constraints on training, or designing more complicated preprocessing methods. Pan et al. (Pan et al., 2021) adopt the contrastive learning scheme in many-to-many MNMT. Tan et al. (Tan et al., 2019) propose a distillation based approach to boost the accuracy of MNMT systems. However, these word/subword-based models still need complex preprocessing steps such as data augmentation or special model architecture design.

Byte tokenizaiton Recently, byte tokenization methods are proposed to address the OOV problems in word/subword-based models. Ruiz et al. (Ruiz Costa-Jussà et al., 2017) compare character-based and byte-based NMT systems and show that byte-based systems converge faster. Wang et al. (Wang et al., 2020) combine subwords tokenization with byte encoding and propose a byte-level BPE (BBPE). Shaham and Levy (Shaham and Levy, 2021) propose embeddingless byte-to-byte machine translation by replacing the token embedding layer in subword-based models with one-hot encoding for bytes. However, among these models, byte-level MNMT is still not studied, and the randomness of byte tokenization as we discussed above is not investigated.

Therefore, different from the previous work, we mainly focus on byte-based MNMT, while simultaneously considering the randomness of bytes and endangered languages.

3 Methods

3.1 Preliminary of Byte Representation

Any writing system can be encoded with a byte sequence (Needleman, 2000; Shaham and Levy, 2021), using pre-defined byte encoding methods, such as UTF-8 for almost all languages, GBK for simplified Chinese, and eucJP for Japanese.

Formally, we use a mapping function $f : \mathcal{C} \rightarrow \mathcal{B}^n$ to denote the mapping from characters in a raw sentence to bytes. Here, $\mathcal{C}$ is the character domain for all languages, $\mathcal{B} = (0, 1, \ldots, 255)$ is the byte domain, and n is the maximum byte number that a character maps into. Also, we define f^{-1} on a byte sequence to convert it back to the text. In this paper, we use UTF-8 as the mapping function, because it is a general encoding method and contains almost all characters in existing languages. Figure 2 shows four different languages represented in bytes, including English (en), Chinese (zh), German (de), and Turkish (tr). A character in each language is mapped into bytes. Particularly, characters of some languages such as Chinese are mapped into multiple bytes.

(a) en (b) zh (c) de (d) tr

Figure 2: Byte representation of four languages.

3.2 Problem Definition

Here, we first describe the input and output of the multilingual translation task.

Definition 1 (Multilingual Domain)**.** *We use $\mathcal{S} = \{\mathcal{S}_1, \mathcal{S}_2, \ldots, \mathcal{S}_N\}$ to denote the source language domain and use $\mathcal{T} = \{\mathcal{T}_1, \mathcal{T}_2, \ldots, \mathcal{T}_M\}$ to denote the target language domain. $\mathcal{S}_i$ or $\mathcal{T}_j$ represents a type of language. Note that $\mathcal{S}$ and $\mathcal{T}$ can have intersection.*

Definition 2 (Multilingual Sentence Pair)**.** *Given the source domain $\mathcal{S}$ and the target domain $\mathcal{T}$, we define the multilingual sentence pair set $\mathcal{L} = \{(s_i, s_j) \mid s_i \in \mathcal{S}_i$ and $s_j \in \mathcal{T}_j$ and $\mathcal{S}_i \neq \mathcal{T}_j\}$. Here, the s_i and s_j denote two parallel sentences from different languages.*

Based on the above description, we formally define the multilingual translation task as follow:

Definition 3 (Multilingual Translation)**.** *Given the multilingual sentence pair $\mathcal{L}$ as the training set and a translation model $\mathcal{M}$ with parameters $\boldsymbol{\theta}$, we aim to find the optimized parameters $\hat{\boldsymbol{\theta}}$ of $\mathcal{M}$ to minimize the following objective function:*

$$\hat{\boldsymbol{\theta}} = \arg\min_{\boldsymbol{\theta}} \sum_{(s_i, s_j) \in \mathcal{L}} -p(s_j) \log p(\hat{s}_j | s_i; \boldsymbol{\theta}). \quad (1)$$

Here, $\hat{s}_j$ is the predicted sentence of the target sentence s_j. To generate each token $\hat{x}_j^t$ in $\hat{s}_j$, we use s_i and the previous tokens $s_j^{<t}$ of s_j to calculate a probability $p(\hat{x}_j^t) = \mathcal{M}(s_i, s_j^{<t}; \boldsymbol{\theta})$. Assume s_j consists of m tokens (byte or subword): $(x_j^1, x_j^2, \ldots, x_j^m)$, the conditional probabil-

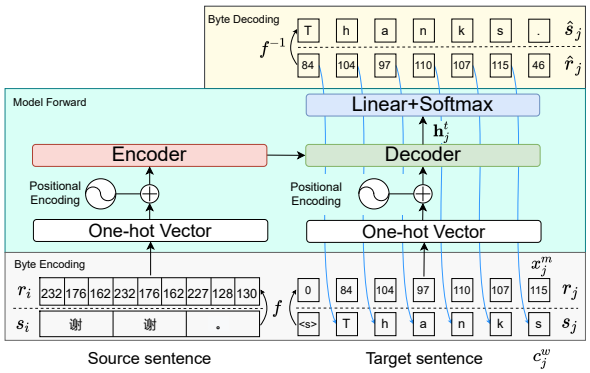

Figure 3: The overview of the proposed byte-based multilingual translation model BMNMT.

ity $p(\hat{s}_j|s_i;\boldsymbol{\theta})$ is defined as:

$$p(\hat{s}_j|s_i;\boldsymbol{\theta}) = \prod_{t=1}^{m} p(\hat{x}_j^t) = \prod_{t=1}^{m} \mathcal{M}(s_i, s_j^{<t};\boldsymbol{\theta}). \tag{2}$$

3.3 Multilingual translation with byte-level tokenization

Suppose we are given a sentence pair $(s_i, s_j) \in \mathcal{L}$ and $s_i = (c_i^1, \ldots, c_i^k), s_j = (c_j^1, \ldots, c_j^w)$, respectively. c is a character of s_i or s_j. k and w denote the character number of the raw source sentence s_i and target sentence s_j, respectively. Our proposed multilingual translation framework BMNMT contains three main parts: byte encoding, model forward, and byte decoding. The model overview of BMNMT is shown in Figure 3.

Byte encoding We first use the byte mapping function f to encode the raw sentence pairs into byte sequences. Take s_j in Figure 3 as an example. We first map w characters in each raw sentence into a new sequence r_j of m byte tokens:

$$\begin{aligned} r_j &= (f(c_j^1), f(c_j^2), \ldots, f(c_j^w)) \\ &= (x_j^1, x_j^2, \ldots, x_j^m). \end{aligned} \tag{3}$$

It is worth noting that we also consider the punctuation and the space symbol in the raw text as characters and encode them into bytes.

Model forward Our byte-level multilingual translation model is based on the state-of-the-art Transformer architecture (Vaswani et al., 2017), which includes an encoder $\texttt{Enc}$ and a decoder $\texttt{Dec}$. After getting the byte tokens of sentence pairs, we convert r_i and r_j to one-hot vectors $\mathbf{r}_i$ and $\mathbf{r}_j$. Then, the encoder encodes the source sequence into hidden representations, and the decoder outputs logits $\mathbf{h}_j^t \in \mathbb{R}^d$ for each generated token $\hat{x}_j^t$ of the target sentence. Here, d is the dimension of the decoder output. Finally, $\mathcal{M}$ calculates a probability $p(\hat{x}_j^t)$ with a fully-connected (FC) layer with Softmax:

$$\mathbf{h}_j^t = \texttt{Dec}(\texttt{Enc}(\mathbf{r}_i), \mathbf{r}_j^{<t}) \in \mathbb{R}^d, \tag{4}$$

$$p(\hat{x}_j^t) = \text{Softmax}(\mathbf{W}\mathbf{h}_j^t) \in \mathbb{R}^{256}. \tag{5}$$

Here, $\mathbf{W} \in \mathbb{R}^{256\times d}$ is the weight in the FC layer to project the output space of the decoder into the byte space.

Byte decoding After getting the probability distribution $p(\hat{x}_j^t)$ of the current output token $\hat{x}_j^t$, we then use Beam Search ($\texttt{BS}$) to sample the target byte token $\hat{x}_j^t$. Finally, after generating the entire byte sequence, we use the inverse mapping f^{-1} to retrieve the real generated text sentence $\hat{s}_j$:

$$\hat{x}_j^t = \texttt{BS}(p(\hat{x}_j^t)) \tag{6}$$

$$\hat{s}_j = f^{-1}(\hat{x}_j^1, \hat{x}_j^2, \ldots, \hat{x}_j^m) \tag{7}$$

Note that, in the inference process, we autoregressivelly output the target tokens using the previous generated tokens $\hat{s}_j^{<t}$ instead of $s_j^{<t}$, i.e., $p(\hat{x}_j^t) = \mathcal{M}(s_i, \hat{s}_j^{<t};\boldsymbol{\theta})$.

3.4 Random byte encoding and ensemble prediction

As discussed in the previous section, we use a one-hot vector to represent a byte token. For example, in Figure 3, the byte representation of character "h" is 104 under UTF-8 encoding. In the one-hot vector, the entry in 104 is one and the others are 0. However, we conjecture UTF-8 is just one of the mapping functions for bytes. The language model should not be limited by a single byte encoding method, because a single encoding method can bring possible bias in model representation. Rare resources of data and other characteristics make it harder for endangered languages on translation task. Therefore, we propose an additional random byte encoding method besides the basic byte tokenization by generating multiple random byte mappings. Then we design an ensemble prediction by training multiple models $\mathcal{M}$s with different byte mappings and output an average probability among all $\mathcal{M}$s to enhance the robustness of the translation model.

Random byte mapping To generate multiple random mappings, we define z permutation functions $g_1, \ldots, g_z$ by shuffling the original bytes $\mathcal{B} = (0, 1, \ldots, 255)$ to $\mathcal{P}_1, \ldots, \mathcal{P}_z$. Here, g_l is

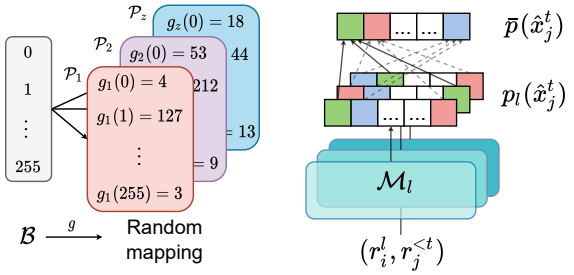

(a) Random byte mapping (b) Ensemble prediction

Figure 4: Random byte mapping and ensemble prediction. (a) Random byte mapping with z different byte permutations of UTF-8. (b) Ensemble prediction by producing z BMNMT for each byte permutation.

a one-to-one mapping from $\mathcal{B}$ to $\mathcal{P}_l$. Figure 4(a) shows one example of the random byte mapping. $g_1(0) = 4$ maps byte 0 in $\mathcal{B}$ to byte 4 in $\mathcal{P}_1$.

Ensemble prediction In the previous section, we directly use byte encoding from UTF-8. By introducing random byte encoding, we first update Equation (3) using the byte mapping g_l:

$$r_j^l = \left(g_l\left(x_j^1\right), g_l\left(x_j^2\right) \ldots, g_l\left(x_j^m\right)\right) \quad (8)$$

Then, we have z byte-level input sentence pairs $(r_i^1, r_j^1), (r_i^2, r_j^2), \ldots, (r_i^z, r_j^z)$ for each (s_i, s_j). For l-th random byte encoding, we adopt an individual multiliangual Transformer $\mathcal{M}_l$ to take the l-th byte-level language pair under mapping $g_l(\cdot)$ as the the input. As a result, we calculate z probability distributions $p_1(\hat{x}_j^t), p_2(\hat{x}_j^t), \ldots, p_z(\hat{x}_j^t)$ from Equation (5). Next, we ensemble these z distributions. $p_l(\hat{x}_j^t)$ is a vector of 256 entries. For each entry $v \in \mathcal{B}$, we find the corresponding entry $g_l(v)$ in probability distributions $p_l(\hat{x}_j^t)$ of $\mathcal{P}_l$ and calculate an average probability from $\mathcal{P}_1$ to $\mathcal{P}_z$:

$$\bar{p}^v(\hat{x}_j^t) = \frac{1}{z}\sum_{l=1}^{z} p_l^{g_l(v)}(\hat{x}_j^t). \quad (9)$$

We still take Figure 4(a) as an example. When calculating the probability of byte 0 in $\mathcal{B}$, we find byte 4 in $\mathcal{P}_1$, byte 53 in $\mathcal{P}_2, \ldots$, and byte 18 in $\mathcal{P}_z$. Finally, we use $\bar{p}(\hat{x}_j^t)$ to execute beam search instead of $p(\hat{x}_j^t)$ in Equation (6). The ensemble prediction modules are demonstrated in Figure 4(b).

4 Experimental Settings

4.1 Dataset

In our experiments, we use the English-centric IWSLT14 dataset (Cettolo et al., 2014) and the IndCorpus dataset (Chen et al., 2021).

Specifically, in IWSLT14, we select five high-resource language pairs, i.e., Chinese, Arabic, German, Farsi, Turkish to English, and one low-resource language pair, i.e., Slovenian to English. When preprocessing IWSLT14, we remove sentences longer than 800 bytes in the training set, following the settings in (Shaham and Levy, 2021). In total, about 5% samples are removed.

The IndCorpus contains ten endangered languages. We translate each language to Spanish. Because the entire test set of IndCorpus is not publicly available, we adopt the Dev set as the test data. The statistics of IWSLT14 and IndCorpus can be found in Appendix A. Note that, although some endangered languages in IndCorpus have a large number of training samples, we still regard them as endangered languages because they have the characteristics of endangered languages.

4.2 Baselines and Model Settings

To study the representation bottleneck when upgrading from monolingual translation to multilingual translation, we first include a byte-based monolingual model. Next, to validate the effectiveness of byte-based MNMT, we also incorporate two subword-based monolingual and multilingual models. Specifically, we select the following model schemes as baselines with different model architectures on two datasets:

- *B-N* (Shaham and Levy, 2021). To valid the ability of our byte-based MNMT model BM-NMT in alleviating the representation bottleneck, we select a byte-based NMT model (B-N) for monolingual translation.
- *W-N* (Mager et al., 2021). To compare the translation performance between our model and subword-based model, we select a subword-based NMT model (W-N) for monolingual translation.
- *W-M*. We implement a subword-based MNMT model (W-M) for multilingual translation based on W-N to further evaluate the effectiveness of our byte-based model on multilingual.

The framework of all models including our model are Transformers with the same architecture on each dataset. For the model scheme in IWSLT14, we follow the architecture in (Shaham and Levy, 2021). For IndCorpus, we follow the architecture in (Mager et al., 2021). For both tokenizations,

all language pairs share the same model and the same dictionary. For the subword-based models, the embedding layers are shared among source and target languages. For byte-based models, we remove the embedding layers following (Shaham and Levy, 2021). The detailed model settings on two datasets including architecture and environment are listed in Appendix B. The source code is available at the Github repo: https://github.com/MengjiaoZhang/Byte-based-multilingual-NMT.

4.3 Hyper-parameters

Following (Shaham and Levy, 2021) and (Mager et al., 2021), for subword tokenization with BPE, we use 10,000 merging steps. The dropout rates of models in IWSLT14 and IndCorpus are 0.2 and 0.4, respectively. Due to the limited training data in IndCorpus 2021, we also set attention dropout as 0.2 and ReLU dropout as 0.2 to avoid over-fitting in IndCorpus. The optimizer is Adam (Kingma and Ba, 2015) with the inverse square root learning rate scheduler. We set the warm-up steps as 4,000 and the minimum learning rate is 10^{-7}. The training epoch for IWSLT14 is 200 with early stop when observing no decrease of validation loss within 5 consecutive epochs. The epoch number for IndCorpus is 50 without early stop because we use the dev set in IndCorpus as the test data.

4.4 Evaluation Metrics

To evaluate the translation results of all models, we use the commonly adopted BLEU scores on both datasets and use an additional metric ChrF (Popović, 2015) to evaluate endangered languages in IndCorpus. Here, ChrF denotes the character n-gram F-score. We adopt ChrF because not all endangered languages in IndCorpus have a tokenization standard (Mager et al., 2021).

4.5 Training strategies

To analyze the causes of the representation bottleneck and explore model generalizability while studying the effectiveness of the basic BMNMT framework and ensemble prediction, we consider multiple scenarios related to languages resource.

Case 1. Jointly **T**rain BMNMT with both **H**igh- and **L**ow-resource language pairs without finetuning and ensemble prediction (T-HL).

Case 2. Jointly **T**rain BMNMT only with the **H**igh resource language pairs without finetuning and ensemble prediction (T-H).

Case 3. Jointly **T**rain BMNMT on the **E**ndangered language pairs without finetuning and ensemble prediction (T-E).

Case 4. **F**inetune BMNMT with **E**ndangered languages on pretrained BMNMT in Case 2 (F-E2).

Case 5. Jointly **T**rain BMNMT on **E**ndangered languages without finetuning but with ensemble prediction (T-E+P).

For the first four cases without ensemble, we use the original UTF-8 encoding. For the last case, we adopt the proposed random byte encoding method.

	Byte (BLEU)		Subword (BLEU)	
	B-N	BMNMT (Ours)	W-N	W-M
ar-en	30.8	30.4 (**-0.4**)	30.5	28.8 (-1.7)
de-en	34.4	34.2 (**-0.2**)	34.1	33.1 (-1.0)
fa-en	22.7	24.2 (**+1.5**)	21.6	23.0 (+1.4)
tr-en	22.8	22.5 (**-0.3**)	22.2	21.9 (-0.3)
zh-en	15.8	15.8 (**+0.0**)	15.9	15.8 (-0.1)
sl-en	2.2	20.7 (**+18.5**)	8.9	20.2 (+11.3)
Avg.	21.4	24.6 (**+3.2**)	22.2	23.8 (+1.6)

Table 1: Representation bottleneck analysis on Case 1 (T-HL) using the IWSLT14 dataset. Values in "()" represents the BLEU score difference between multilingual and monolingual models. Here, Slovenian (sl) is a relatively low-resource language.

5 Experimental Results

In the main paper, we report the BLEU scores of all cases. The ChrF for endangered languages in Cases 3-5 can be found in Appendix C.

5.1 Representation Bottleneck Analysis on Cases 1 (T-HL) and 2 (T-H)

To validate the effectiveness of our model in addressing the representation bottleneck in multilingual translation with both high and low resource languages, we adopt the training strategies of Cases 1 (T-HL) and 2 (T-H). First, we run the subword-based translation baselines on monolingual (W-N) and multilingual (W-M) languages and calculate the difference of BLEU scores between them. Then, we also calculate such difference between BMNMT and B-N to evaluate whether BMNMT can alleviate the representation bottleneck in MNMT.

Table 1 shows the results of Case 1 (T-HL) on IWSLT14 dataset. In this experiment, the high-resource language pairs are Arabic (ar), German

	Byte (BLEU)		Subword (BLEU)	
	B-N	BMNMT (Ours)	W-N	W-M
ar-en	30.8	31.5 (**+0.7**)	30.5	29.6 (-0.9)
de-en	34.4	35.2 (**+0.8**)	34.1	33.6 (-0.5)
fa-en	22.7	24.9 (**+2.2**)	21.6	23.7 (+2.1)
tr-en	22.8	23.8 (**+1.0**)	22.2	22.4 (+0.5)
zh-en	15.8	16.9 (**+1.1**)	15.9	16.2 (+0.4)
Avg.	25.3	26.5 (**+1.2**)	24.9	25.1 (+0.2)

Table 2: Representation bottleneck analysis on Case 2 (T-H) with all high-resource languages in IWSLT14.

	Byte (BLEU)		Subword (BLEU)	
	B-N	BMNMT (Ours)	W-N	W-M
quy-es	2.4	3.5 (**+0.9**)	3.3	2.7 (-0.6)
gn-es	3.6	4.4 (**+0.8**)	2.1	3.0 (+0.9)
nah-es	0.2	2.6 (**+2.4**)	0.8	2.1 (+1.3)
shp-es	0.0	3.9 (**+3.9**)	0.3	2.8 (+2.5)
Avg.	1.6	3.6 (**+2.0**)	1.6	2.7 (+1.1)

Table 3: Representation bottleneck analysis on Case 3 (T-E) with endangered languages in IndCorpus. Quechua (quy) is a relatively high-resource endangered language.

(de), Farsi (fa), Turkish (tr), and Chinese (zh) to English (en), while the low-resource language pair is Slovenian (sl) to English. We first notice byte-level models achieve the best performance on almost all language pairs. It proves the capability of byte tokenization in the translation task. Furthermore, as a low-resource language, the translation from Slovenian to English gains the largest benefit from BMNMT, and BMNMT has the best average BLEU score. Therefore, we can conclude that the application of byte tokenization to MNMT is able to enhance the knowledge sharing among multiple languages.

In addition, we notice that subword-based multilingual translation (W-M) suffers from representation bottleneck, because the performance of some high-resource language pairs ar-en, de-en, tr-en, and zh-en decrease compared with monolingual translation (W-N). However, our proposed byte-based multilingual translation shows a much smaller decrease than word-based models. It further proves the ability of BMNMT to alleviate the representation bottleneck problem in MNMT.

Table 2 shows the results of Case 2 (T-H). Here, to further study the influence of language resource on the representation bottleneck, we remove the low-resource language Slovenian (sl) in Case 1 (T-HL) and re-train BMNMT and other baselines. We notice that after removing the language Slovenian, all MNMT models based on byte and subword gain a higher performance in the average BLEU score. However, subword-based W-M still cannot avoid the representation bottleneck, while our BMNMT achieves improvement on all pairs against the B-N model. More importantly, BMNMT has the best performance on all language pairs.

In summary, when training MNMT with both high and low-resource languages, the low-resource languages are a main reason for the representation bottleneck. Moreover, compared to subword-based models, our proposed BMNMT can effectively alleviate this problem.

5.2 Representation Bottleneck Analysis on Endangered Languages: Case 3 (T-E)

To evaluate the ability of BMNMT to address the representation bottleneck in endangered languages, we adopt the training strategy of Case 3 (T-E). Here, we still report the BLUE score differences between monolingual and multilingual models.

Table 3 demonstrates the results of Case 3. Following the setting in Case 1, we select four language pairs Quechua (quy), Guarani (gn), Nahuatl (nah), and Shipibo-Konibo (shp) to Spanish (es) that have relatively large training sizes in IndCorpus to avoid the bias in languages of extremely low resource. Among these pairs, quy has the most training data that are comparable to high-resource languages in IWSLT14, while the others can be regarded as low-resource languages. We notice the BLEU score of quy is much lower than languages in IWSLT14. We infer it is because of the characteristics in endangered languages. However, different to the results in Case 1, the BLEU score of byte-based BMNMT on quy-es does not decrease even with multilingual training. To summarize, we think our BMNMT can help overcome the representation bottleneck of rare resource in endangered languages and bring enhancement in translation.

5.3 Generalizability Analysis: Case 4 (F-E2)

Another property we want to analyze for multilingual translation is the model generalizability. Therefore, we adopt Case 4 (F-E2) to generalize the pretrained model on high-resource languages to endangered languages. As the case indicates, we use the pretrained BMNMT and W-M in Case 2, which are trained on all high-resource languages and individually finetuned on all endangered lan-

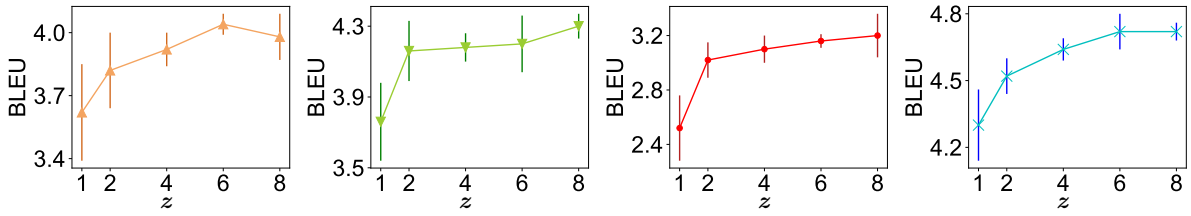

(a) quy-es (b) shp-es (c) nah-es (d) gn-es

Figure 5: Robustness analysis on Case 5 (T-E+P): Average BLEU scores of ensemble prediction with different ensemble number z.

	Byte (BLEU)		Subword (BLEU)	
	B-N	BMNMT (Ours)	W-N	W-M
gn-es	4.5	5.9 (**+1.4**)	3.6	4.0 (+0.4)
nah-es	3.4	4.5 (**+1.1**)	2.8	2.0 (-0.8)
quy-es	7.0	7.9 (**+0.9**)	5.4	5.9 (+0.4)
shp-es	0.6	2.7 (**+2.1**)	1.0	1.1 (+0.1)
aym-es	3.8	5.1 (**+1.3**)	2.8	2.7 (-0.1)
cni-es	0.3	1.8 (**+1.5**)	0.6	0.6 (+0.0)
bzd-es	0.7	2.5 (**+1.7**)	0.9	0.9 (+0.0)
oto-es	0.4	1.4 (**+1.0**)	0.4	0.4 (+0.0)
tar-es	0.2	0.7 (**+0.5**)	0.2	0.3 (+0.1)
hch-es	1.7	2.6 (**+0.9**)	1.1	0.9 (-0.2)
Avg.	2.3	3.5 (**+1.2**)	1.9	1.9 (+0.0)

Table 4: Generalizability analysis on Case 4 (F-E2) by finetuning with all endangered languages in IndCorpus based on pretrained models in Case 2.

guages. As monolingual models, B-N and W-N are pretrained on German to English and finetuned on endangered languages. It is worth noting that all endangered languages in finetuning do not occur in pretraining to validate the models' generalizability.

Table 4 shows the finetuning results on IndCorpus. We first notice that the models pretrained on multilingual languages perform better than monolingual models. It indicates that multilingual models have a stronger generalizability to endangered languages. Additionally, although the byte-based monolingual model B-N has worse performance than W-N on some language pairs, the BLEU scores of these pairs are largely increased by our BMNMT compared to W-M. It further validates the generalizability of BMNMT.

5.4 Robustness Analysis: Case 5 (T-E+P)

To evaluate the translation effectiveness and robustness of our proposed BMNMT on endangered languages, we generate multiple random byte mappings and adopt Case 5 (T-E+P) in the ensemble prediction experiment. Specifically, we first choose the number z of random byte mappings, i.e., ensemble number from $\{1, 2, 4, 6, 8\}$. For each z, we train BMNMT 5 times on IndCorpus with different permutations of bytes in UTF-8. The average and standard deviation of BLEU scores in 5 runs for each z are reported in Figure 5.

With the growth of z, the BLUE score shows an increasing trend. It proves that appropriate ensemble can improve the translation performance. Moreover, BMNMT without ensemble ($z = 1$) has the highest standard deviation. We infer byte-based translation with only one encoding method can brings noise to token representations. However, with multiple byte mapping and ensemble prediction, the translation becomes more stable. Therefore, we think that the random byte mapping and ensemble prediction can improve the robustness of byte-based translation.

In summary, based on all analysis for the representation bottleneck, generalizability, and robustness, the effectiveness of our proposed BMNMT is validated. We conclude that with the introduce of byte in MNMT and the ensemble prediction for byte mappings, the representation bottleneck can be alleviated, especially on endangered language.

6 Conclusion

Multilingual neural machine translation has been successful in enhancing low-resource languages because of knowledge sharing. To address the representation bottleneck in existing subword-based multilingual translation systems, we propose a byte-based MNMT model, BMNMT with the Transformer architecture. To improve the model generalizability and robustness, we further design an ensemble prediction method with random byte encoding. Our experimental results show that BMNMT can alleviate the representation bottleneck and has a stronger generalization ability compared with subword-based MNMT. Meanwhile, BMNMT with ensemble prediction improves the transla-

tion performance and robustness on endangered language translation tasks. Extending our byte-based method to large scale models and datasets is promising and can improve model performance, which will be our future work.

Acknowledgments

We appreciate the National Science Foundation (NSF) Award No. 1747728 and NSF CRAFT Award, Grant No. 22001 to fund this research. We are also thankful for the support of the Google Cloud Research Program. We thank Chang Lu for the comments.

References

Roee Aharoni, Melvin Johnson, and Orhan Firat. 2019. Massively multilingual neural machine translation. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 3874–3884.

Naveen Arivazhagan, Ankur Bapna, Orhan Firat, Dmitry Lepikhin, Melvin Johnson, Maxim Krikun, Mia Xu Chen, Yuan Cao, George Foster, Colin Cherry, et al. 2019. Massively multilingual neural machine translation in the wild: Findings and challenges. *arXiv preprint arXiv:1907.05019*.

Tamali Banerjee and Pushpak Bhattacharya. 2018. Meaningless yet meaningful: Morphology grounded subword-level nmt. *NAACL HLT 2018*, page 55.

Mauro Cettolo, Jan Niehues, Sebastian Stüker, Luisa Bentivogli, and Marcello Federico. 2014. Report on the 11th iwslt evaluation campaign. In *Proceedings of the 11th International Workshop on Spoken Language Translation: Evaluation Campaign*.

Mia Xu Chen, Orhan Firat, Ankur Bapna, Melvin Johnson, Wolfgang Macherey, George Foster, Llion Jones, Mike Schuster, Noam Shazeer, Niki Parmar, et al. 2018. The best of both worlds: Combining recent advances in neural machine translation. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 76–86.

Wei-Rui Chen, Muhammad Abdul-Mageed, Hasan Cavusoglu, et al. 2021. Indt5: A text-to-text transformer for 10 indigenous languages. In *Proceedings of the First Workshop on Natural Language Processing for Indigenous Languages of the Americas*, pages 265–271.

Raj Dabre, Chenhui Chu, and Anoop Kunchukuttan. 2020. A survey of multilingual neural machine translation. *ACM Computing Surveys (CSUR)*, 53(5):1–38.

Jeff Ens, Mika Hämäläinen, Jack Rueter, Philippe Pasquier, et al. 2019. Morphosyntactic disambiguation in an endangered language setting. In *22nd Nordic Conference on Computational Linguistics (NoDaLiDa) Proceedings of the Conference*. Linköping University Electronic Press.

Jiatao Gu, Yong Wang, Yun Chen, Victor OK Li, and Kyunghyun Cho. 2018. Meta-learning for low-resource neural machine translation. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 3622–3631.

Melvin Johnson, Mike Schuster, Quoc V Le, Maxim Krikun, Yonghui Wu, Zhifeng Chen, Nikhil Thorat, Fernanda Viégas, Martin Wattenberg, Greg Corrado, et al. 2017. Google's multilingual neural machine translation system: Enabling zero-shot translation. *Transactions of the Association for Computational Linguistics*, 5:339–351.

Diederik P Kingma and Jimmy Ba. 2015. Adam: A method for stochastic optimization. In *ICLR (Poster)*.

Gina-Anne Levow, Emily P Ahn, and Emily M Bender. 2021. Developing a shared task for speech processing on endangered languages. In *Proceedings of the Workshop on Computational Methods for Endangered Languages*, volume 1, pages 96–106.

Zoey Liu, Crystal Richardson, Richard Hatcher Jr, and Emily Prud'hommeaux. 2022. Not always about you: Prioritizing community needs when developing endangered language technology. *arXiv preprint arXiv:2204.05541*.

Manuel Mager, Arturo Oncevay, Abteen Ebrahimi, John Ortega, Annette Rios Gonzales, Angela Fan, Ximena Gutierrez-Vasques, Luis Chiruzzo, Gustavo Giménez-Lugo, Ricardo Ramos, et al. 2021. Findings of the americasnlp 2021 shared task on open machine translation for indigenous languages of the americas. In *Proceedings of the First Workshop on Natural Language Processing for Indigenous Languages of the Americas*, pages 202–217.

Makoto Morishita, Jun Suzuki, and Masaaki Nagata. 2018. Improving neural machine translation by incorporating hierarchical subword features. In *Proceedings of the 27th International Conference on Computational Linguistics*, pages 618–629.

Mark Needleman. 2000. The unicode standard. *Serials review*, 26(2):51–54.

Xiao Pan, Mingxuan Wang, Liwei Wu, and Lei Li. 2021. Contrastive learning for many-to-many multilingual neural machine translation. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 244–258.

Maja Popović. 2015. chrf: character n-gram f-score for automatic mt evaluation. In *Proceedings of the Tenth Workshop on Statistical Machine Translation*, pages 392–395.

Marta Ruiz Costa-Jussà, Carlos Escolano Peinado, and José Adrián Rodríguez Fonollosa. 2017. Byte-based neural machine translation. In *Proceedings of the First Workshop on Subword and Character Level Models in NLP*, pages 154–158. Association for Computational Linguistics.

Uri Shaham and Omer Levy. 2021. Neural machine translation without embeddings. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 181–186.

Felix Stahlberg. 2020. Neural machine translation: A review. *Journal of Artificial Intelligence Research*, 69:343–418.

Xu Tan, Yi Ren, Di He, Tao Qin, and Tie-Yan Liu. 2019. Multilingual neural machine translation with knowledge distillation. In *International Conference on Learning Representations*.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. *Advances in neural information processing systems*, 30.

Changhan Wang, Kyunghyun Cho, and Jiatao Gu. 2020. Neural machine translation with byte-level subwords. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pages 9154–9160.

Yonghui Wu, Mike Schuster, Zhifeng Chen, Quoc V Le, Mohammad Norouzi, Wolfgang Macherey, Maxim Krikun, Yuan Cao, Qin Gao, Klaus Macherey, et al. 2016. Google's neural machine translation system: Bridging the gap between human and machine translation. *arXiv preprint arXiv:1609.08144*.

Barret Zoph and Kevin Knight. 2016. Multi-source neural translation. *arXiv preprint arXiv:1601.00710*.

A Dataset Statistics

Tables 5 and 6 show the data statistics of IWSLT14 and endangered language dataset IndCorpus, respectively. IWSLT datasets contains English scripts of TED talks translated into other languages. All training samples in IWSLT are collected in 2014. The test samples are from TED talks from 2010 to 2012. For IndCorpus, there are ten indigenous languages from multiple countries of America.

B Model Architecture and Environment

In every Transformer model used in IWSLT14, we adopt 6 attention layers for both encoder for decoder. The number of attention heads is 4. For the subword-based models, they contain an embedding layer for tokens. The embedding dimension is 512, which is the same as the hidden dimension. The feed-forward layer is built upon the hidden layer to calculate the output digits. The dimension of feed-forward layers in the encoder and decoder is 1024. For byte-based models, they do not have the embedding layer.

ISO	Language	Train	Dev	Test
zh	Chinese	166,046	7,547	5,099
ar	Arabic	165,591	7,526	5,357
de	German	158,516	7,205	5,585
fa	Farsi	99,792	4,536	4,244
tr	Turkish	142,619	6,482	5,433
sl	Slovenian	15,859	720	2,555

Table 5: Languages from the IWSLT14 dataset with the ISO codes. The Train, Dev, and Test columns denote the number of sentence pairs of each language with English in the training, validation, and test set.

ISO	Language	Train	Dev
cni	Asháninka	3,883	883
aym	Aymara	6,531	996
bzd	Bribri	7,508	996
gn	Guarani	26,032	995
nah	Nahuatl	16,145	672
oto	Otomí	4,889	599
quy	Quechua	125,008	996
tar	Rarámuri	14,721	995
shp	Shipibo-Konibo	14,592	996
hch	Wixarika	8,966	994

Table 6: The languages featured in the IndCorpus, their ISO codes, and dataset statistics.

For the Transformer model in IndCorpus, we shrink the model size to avoid overfitting because the languages in IndCorpus only contain limited training samples. The encoder and decoder both have 5 attention layers with 2 heads. The remaining parts keep the same as the model architecture used in IWSLT14.

We use the transformer model implemented by faiseq[1]. All the program used in this work is implemented using Python 3.8, PyTorch 1.10.0, and CUDA 11.3. For the hardware environment, we run our program on a machine with Intel i9-10900KF CPU, 128G memory, and an NVIDIA GeForce RTX 3090 GPU.

C Additional Experimental Results on Endangered Languages using ChrF

Table 3 and Table 9 show the translation performance of endangered languages with ChrF in Cases 3 (T-E) and 4 (F-E2). Similar to the results in Ta-

[1] https://github.com/facebookresearch/fairseq

	IWSLT14	IndCorpus
Encoder layers	6	5
Decoder layers	6	5
Attention heads	4	2
Hidden dim d	512	512
Feed-forward dim	1024	1024

Table 7: Model architectures on the IWSLT14 and IndCorpus datasets.

	Byte (ChrF)		Subword (ChrF)	
	B-N	BMNMT (Ours)	W-N	W-M
quy-es	22.5	25.2 (**+2.7**)	21.5	20.3 (-0.8)
gn-es	21.8	23.4 (+1.6)	18.3	22.7 (**+4.4**)
nah-es	13.0	19.7 (**+6.7**)	15.0	19.6 (+4.6)
shp-es	12.0	27.0 (**+15.0**)	10.5	21.6 (+10.9)
Avg.	17.3	23.8 (**+5.5**)	16.7	21.2 (+4.5)

Table 8: Representation bottleneck analysis on Case 3 (T-E) with endangered languages. The evaluation metric in this table is ChrF.

ble 3 and Table 4 with BLEU scores, our proposed BMNMT has the best performance, and it can improve ChrF on all languages. Evaluation with these two metrics proves that the byte tokenization can alleviate the representation bottleneck even in endangered languages.

In Figure 6, we plot the ChrF scores of ensemble prediction. With the number of ensemble models increasing, the average translation performance shows the same trend while the standard deviation decrease. Both the evaluation on BLEU scores and ChrF show that the ensemble prediction with random byte mapping improve the translation performance and robustness in MNMT.

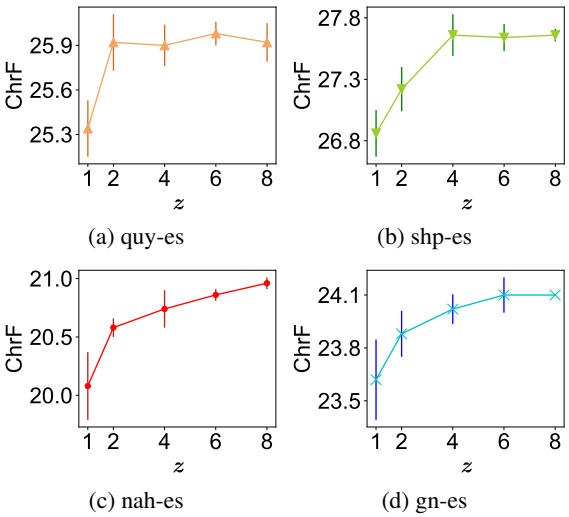

(a) quy-es (b) shp-es

(c) nah-es (d) gn-es

Figure 6: Robustness analysis on Case 5 (T-E+P): Average ChrF of ensemble prediction with different ensemble number z.

	Byte (ChrF)		Subword (ChrF)	
	B-N	BMNMT (Ours)	W-N	W-M
gn-es	25.7	27.1 (**+1.4**)	22.1	21.6 (-0.5)
nah-es	23.2	25.3 (**+2.1**)	18.1	17.7 (-0.4)
quy-es	32.1	32.9 (+0.8)	26.3	27.2 (**+0.9**)
shp-es	20.3	27.3 (**+7.0**)	16.2	17.0 (+0.8)
aym-es	22.9	25.8 (**+2.9**)	18.2	18.7 (+0.5)
cni-es	17.0	21.2 (**+4.2**)	14.1	13.7 (-0.4)
bzd-es	19.7	24.4 (**+4.7**)	14.3	15.0 (+0.7)
oto-es	17.9	19.4 (**+1.5**)	12.1	13.2 (+0.0)
tar-es	16.2	19.4 (**+3.2**)	15.1	14.7 (-0.4)
hch-es	20.3	23.6 (**+3.3**)	13.6	14.8 (+1.2)
Avg.	21.5	24.6 (**+3.1**)	17.0	17.4 (+0.4)

Table 9: Translation performance (ChrF) on IndCorpus. The results are finetune on the model trained in case 2.

BRCC and SentiBahasaRojak: The First Bahasa Rojak Corpus for Pretraining and Sentiment Analysis Dataset

Nanda Putri Romadhona[†] Sin-En Lu[†] Bo-Han Lu[†] Richard Tzong-Han Tsai[†‡*]

[†]Department of Computer Science and Information Engineering,
National Central University, Taiwan
[‡]Center for GIS, Research Center for Humanities and Social Sciences,
Academia Sinica, Taiwan
nandadona61@gmail.com
{alznn, lu110522028, thtsai}@g.ncu.edu.tw

Abstract

Code-mixing refers to the mixed use of multiple languages. It is prevalent in multilingual societies and is also one of the most challenging natural language processing tasks. In this paper, we study Bahasa Rojak, a dialect popular in Malaysia that consists of English, Malay, and Chinese. Aiming to establish a model to deal with the code-mixing phenomena of Bahasa Rojak, we use data augmentation to automatically construct the first Bahasa Rojak corpus for pre-training language models, which we name the Bahasa Rojak Crawled Corpus (BRCC). We also develop a new pre-trained model called "Mixed XLM". The model can tag the language of the input token automatically to process code-mixing input. Finally, to test the effectiveness of the Mixed XLM model pre-trained on BRCC for social media scenarios where code-mixing is found frequently, we compile a new Bahasa Rojak sentiment analysis dataset, SentiBahasaRojak[1], with a Kappa value of 0.77.

1 Introduction

Code-mixing is common in multilingual societies (Bukhari et al., 2015). People tend to use one primary language for grammar and scripting (Lal et al., 2019), and other languages as auxiliary. Code-mixing is commonly found on social media, such as Facebook, Twitter, or any other microblog services.

Malaysia reflects a multilingual society that considers Malay to be the national language but uses mixed languages in daily life. Bahasa Rojak (Bukhari et al., 2015) is one of the code-mixing examples that combines Malay and English into a certain level of language structure. Bahasa Rojak or Malaysian English (Vollmann and Wooi, 2019) is often mixed with Chinese because the ethnic Chinese population in Malaysia is quite large. Bakar and Mazzalan (2018) show that many users in Malaysia use Bahasa Rojak on Facebook. We can find many code-mixing combinations like Malay-English, Hindi-English, Spanish-English, and others on social media. People not only express their feelings on social media, but also exchange information such as the latest news from their hometowns or countries, and financial topics are also popular. Therefore, natural language processing (NLP) studies are increasingly focused on code-mixing (Thara and Poornachandran, 2018).

One of the most advanced trends in natural language processing is to prepare a large unlabeled corpus to pre-train a language model for representing the input text. The source of this unlabeled corpus is usually Wikipedia (Qiu et al., 2020). However, Wikipedia does not have any Bahasa Rojak pages, which hinders the training of a pre-trained model that can represent Bahasa Rojak input texts.

As a result, in this study, we employ data augmentation to automatically construct a new Bahasa Rojak code-mixing corpus, called *BRCC*, for pre-training language models. To find the best way of exploiting the corpus, we not only pre-train language models including BERT and XLM on BRCC, but also revise the original XLM model to make it able to handle code-mixing input text. The revised model is called *Mixed XLM*. As we mentioned, Bahasa Rojak is most frequently used in social media texts. Hence, we compile the first Bahasa Rojak sentiment analysis dataset, called *SentiBahasaRojak*, to evaluate each language model's performance and reflect its ability to represent Bahasa Rojak input texts.

*Corresponding author.

[1]Both BRCC and SentiBahasaRojak are available at https://data.depositar.io/dataset/brcc_and_sentibahasarojak

Proceedings of the 29th International Conference on Computational Linguistics, pages 4418–4428
October 12–17, 2022.

2 Related Work

2.1 Code-Mixing

Code-mixing is a term in mixed language research, representing a common phenomenon in a multilingual society. Code-mixing means that a single sentence or a single utterance contains different languages (Ho et al., 2007).

Pratapa et al. (2018b) compares three existing bilingual word embeddings and a novel method based on the skip-gram language model. They found that bilingual word embeddings obtained from a mixture of two languages rather than from multilingual monolingual texts are more suitable for code-mixing tasks. Lal et al. (2019) proved that the traditional method, which only takes surface and semantic features into account, is not effective for code-mixing sentiment analysis, so they proposed a new method called "demixing". Choosing English-Hindi as the target of code-mixing research, they used a convolutional neural network to generate sub-words and constructed a dual encoder network composed of two parallel BiLSTMs.

A recent code-mixing paper related to this paper is Qin et al. (2020), in which the authors use a code-mixing corpus to fine-tune mBERT. They also increase the size of the code-mixing corpus through data augmentation. Their method aligns the representations of the source language and multiple target languages by using contextual information. Even though there are multiple target languages, the model only needs to be pre-trained once.

2.2 Code-Mixing and Sentiment Analysis of Social Media in Malaysia

Malay is the official language of Malaysia. In the 19th century, under British colonial rule, English had a profound influence on Malay. English replaced the original writing system completely with Latin script, and many words of economic, political and technical fields in modern Malay are borrowed from English. Due to the large population of Chinese people in Malaysia, there are many Chinese schools and companies leading the spread of Chinese.

Because of cultural blending, Malaysian citizens often use code-mixing language. The phenomena have led to the emergence of "Bahasa Rojak", a new language in Malaysia that combines English, Malay, and Chinese words and structures, and often appears on social media such as Facebook and Twitter. Some Chinese users (Shafiee et al., 2019) introduce Chinese words in text-based communication. For example, Table 1 shows comments in three financial forums. These comments are presented in the form of English or code-mixing.

So far, research on sentiment analysis on social media used by Malaysians is still quite limited. Al-Saffar et al. (2018) use an emotional dictionary construction method to obtain a set of predefined features (emotional words). These features are used to build a machine-learning classifier model to determine the sentiment polarity of the given input social media text. There are also a few labeled pure Malay sentiment analysis datasets (Husein, 2018), but to the best of our knowledge, there is no labeled Bahasa Rojak dataset.

2.3 Pre-trained Language Models

BERT (Devlin et al., 2019) is a pre-trained encoder model based on the Transformer architecture. Since it is bidirectional, context semantics can be considered. Usually, we use a huge corpus, such as Wikipedia, to pre-train BERT. Since its launch in 2018, BERT has achieved leading performance on many sentence-level and token-level natural language processing tasks, such as question answering (Rajpurkar et al., 2016; Joshi et al., 2017), machine translation (McCann et al., 2017), and sentiment analysis (Socher et al., 2013).

RoBERTa (Liu et al., 2019) was made with some modifications based on the BERT model. RoBERTa removes the pre-training task of next sentence prediction, uses dynamic masking in pre-training, and adopts a larger byte-level BPE as its text encoding method. RoBERTa has more parameters than BERT and uses a larger corpus for pre-training to obtain better performance.

XLM-R (Conneau et al., 2020) is a transformer-based cross-lingual model, which combines XLM (Conneau and Lample, 2019) and RoBERTa. The difference with the monolingual XLM is that XLM-R is pre-trained on corpus containing multiple languages, so it can represent sentences containing multiple languages. The basic XLM-R uses Masked Language Model (MLM) as the pre-training task. When a bilingual parallel corpus is available, additional pre-training with the Transla-

Sentence	Translate	Language
dont surprise..Armada will touch below 40sen..	No need to be surprised, Armada will touch under 40 cents.	Manglish
The bad news is finished and the good things have come...charting showing good sign 物极必反	The bad news is finished and the good things have come...charting showing good sign things must be reversed.	Manglish + Chinese
Bukan ex Umno saja, tapi x der integrity, penipu, senyum kambing yang bodoh!	Not just ex Umno, but not integrity, liar, stupid goat smile!	Bahasa Rojak

Table 1: Sample comments on a financial forum in Malaysia

Short form	Original word
a.n.	atas nama
awk	awak
bsh	bodoh
bkn	bukan
bln	bulan

Table 2: Malay short form words and corresponding original words

Language	Passages	Tokens
Code-mixing	2M	62,703,287
Malay	2M	60,519,134
English	2M	75,032,902

Table 3: Statistics on the corpora used for pre-training, including the total number of subwords in each language based on BPE Tokenizer segmentation.

tion Language Model (TLM) task can be used to improve performance.

2.4 Transfer Learning

Recently, transfer learning methods have become popular in natural language processing, especially for low-resource tasks. Transfer learning improves failed steps by transferring the resources of related tasks, languages, or domains of high-resource source settings to low-resource target settings.

Transfer learning is not a new method to solve NLP tasks, since it has long been applied on many NLP tasks, such as latent semantic analysis (Deerwester et al., 1990), Brown clusters (Brown et al., 1992), and pre-trained word embeddings (Mikolov et al., 2013). Ruder et al. (2019) have created a taxonomy that makes it easier for researchers to design solutions based on transfer learning methods. For example, if the target task is the same as the source task and there is only annotated data of the source task, the domain adaptation method can be applied. For low-resource language tasks, the usual practice is to train on the annotated data of high-resource source language and apply to low-resource target languages, such as Farra (2019).

Note that the main purpose of adopting transfer learning methods in cross-language tasks is to transfer lexical knowledge across languages, that is, to establish a cross-language word embedding model.

3 Corpus Compilation

In this study, we mainly deal with three languages: English, Malay, and Bahasa Rojak. For each language, we construct a corpus to pre-train each language model and a sentiment analysis dataset to evaluate each language model's performance. The reason for choosing sentiment analysis is that the frequency of code-mixing on social media texts is relatively high (Thara and Poornachandran, 2018). We will continue to discuss more details about the reason for choosing sentiment analysis in 3.4.

3.1 Data Preprocessing

We use common rules to pre-process our English microblogging corpora, including removing noise or unnecessary characters, tags, URLs, certain symbols, etc. The same rules are also used to pre-process Malay microblogging corpora.

However, there are still differences between English and Malay microblog corpora. In Malay, words or sentences are more often abbreviated into shorter forms, such as dialects, word abbreviations, grammatical neglect, and many more. For example, it is common that the word *because* is written as *bcz*, which causes high noise and a distinct text structure. The short form manner in Malay becomes a serious issue in Malay's NLP research (Ariffin and Tiun, 2020). Also, we have the concern that if the short-form text in Malay is not regularized, the vocabulary size will be too large to train the model and increases the cost of training. Past studies show that normalizing such short-form

Algorithm 1: Bahasa Rojak's Data Augmentation

Input :
Source languages : $sl \leftarrow \{en, ms\}$;
Target languages : $tl \leftarrow \{en : [ms, zh], ms : [en, zh]\}$;
Set of source language sentences : $S_{sl} = \{s^n\}_{n=1}^{N}$;
Replace ratio : $[\alpha, \beta, \gamma]$, Type of replaced word : $rw\_list = [V, N, Adj]$;

Output :
Set of Code-Mixing sentences : $T = \{t^n\}_{n=1}^{N}$;

for i *in* $1...N$ **do**
  $count = 0$;
  $t^i \leftarrow s^i$; // Initialize code-mixed Sentence
  if $random() \leq \alpha$ **then**
    // phrase extraction and syntax analysis, return list
    $phrases \leftarrow get\_phrase(rw\_list, s^i)$;
    while $count \leq \beta * len(phrases)$ **do**
      $rid = random\_int(0, len(phrases))$;
      $rw = random(phrases[rid])$;
      $trans\_phrase \leftarrow Translate(rw, random(tl\{sl\})$;
      $t^i \leftarrow Replace\_and\_Aligment(phrases, trans\_phrase, rid)$;
      $count \leftarrow count + 1$;
  else
    if $random() \leq \gamma$ **then**
      $t^i \leftarrow Translate(s^i, random(tl\{sl\})$ // Translate complete sentences

Phrase Name	Pattern
Noun Phrase	{<DET|ADJ|NOUN.*>+ <DET|ADJ|NOUN.*>+}
Prepositional Phrase	{<ADP> <NP>}{<ADP> <PROPN>}
Verb Phrase	{<VERB.*> <NP|PP|CLAUSE|ADP>+$} {<VERB*> <NOUN*>} {<PART> <VERB>}

Table 4: For the regular expression patterns, we use the NLTK parser to identify noun, prepositional and verb phrases.

text increases data quality and has a positive effect on NLP research (Samsudin et al., 2013; Saloot et al., 2014; Kassim et al., 2020). Chekima and Alfred (2017) collected some Malay SMS rules to normalize short form words, and we continue to add some rules. For more examples, please refer to Table 2.

3.2 Bahasa Rojak Crawled Corpus (BRCC)

In order to pre-train a model that works in both monolingual and code-mixing environments, we first construct monolingual corpora and then derive a code-mixing corpus from them. For English and Malay corpora, we crawl English and Malay pages from Wikipedia. We mainly use our data augmentation method to generate the Bahasa Rojak code-mixing corpus, which is called BRCC (Bahasa Rojak Crawled Corpus). Each of these three corpora has 2 million passages. Table 3 shows the detailed information of the three corpora.

To generate BRCC through data augmentation, we first scrape 93,584 Bahasa Rojak passages from the Malaysia Bursa forum. In order to expand the Bahasa Rojak corpus, we modify the CoSDA-ML method to generate 2 million Bahasa Rojak passages from our English and Malay corpora. The main difference is that CoSDA-ML randomly selects words and translates them into a specific target language, while our method parses the sentence to identify phrases and then randomly selects the phrases to be translated. Algorithm 1 explains the details of our data augmentation method. For a detailed description, please refer to A.1.

Take the phrase "the book in your schoolbag" for example. Suppose we randomly choose a word to translate into another language; if "the" is selected, because of the lack of context, "the" is not suitable to be translated by itself. To alleviate this problem, our method first analyzes sentences with part-of-speech (POS) tags, and identifies noun phrases, prepositional phrases, and verb phrases through patterns composed of POS tags. Table 4 shows the regular expression patterns used with the NLTK

Original Sentence	Augmentation Sentence
Di sepanjang pesisirnya terdapat teluk dan tanjung yang berpontensi dimajukan sebagai kawasan pelancongan	Along the coast terdapat teluk dan tanjung yang berpontensi dimajukan sebagai tourist areas
He was previously offered X Men membership but he declined opting instead to work at the Muir Island research center Polaris Havok s long time lover and also a former X Man who can control magnetism	He was previously offered X Men membership but he declined opting instead to work at pusat penyelidikan Pulau Muir Polaris Havok s 長期戀人 and also bekas X Man who can 控制磁

Table 5: Samples of data augmentation results

(Bird and Loper, 2004) parser to identify the three types of phrases mentioned above. These patterns have been confirmed by native Malay speakers. All-caps words denote POS tags, angle brackets denote sub-patterns, and the rest of the symbols are used the same way as they are in regular expressions. Taking a simplified noun phrase pattern *{<ADJ>+ <NOUN>+}* as an example, this means that if the parser finds at least one adjective (ADJ) followed by at least one noun (NOUN), it finds a noun phrase.

Lastly, we randomly select a phrase to translate into the target language. The purpose of the modification is to choose "the book" instead of "the" or "book". Table 5 shows the sentences generated by our data augmentation method.

3.3 BRCC Quality

To evaluate the quality of BRCC, we conduct a test inspired by the Turing Test on two native Malay speakers. We randomly sample 500 sentences each from the BRCC corpus (using our data augmentation method to synthesize Bahasa Rojak sentences) and the KLSE forum (klse.i3investor.com), where Bahasa Rojak frequently appears. After mixing the two, we ask two native speakers to judge whether it is Bahasa Rojak, sentence by sentence, and if so, the sentence is labeled as positive, otherwise negative. As shown in Table 6, we get similar positive ratios in BRCC and KLSE, which indicates that most of the synthesized Bahasa Rojak sentences in BRCC are considered to be real Bahasa Rojak.

3.4 Sentiment Analysis Datasets

To verify that the Bahasa Rojak code-mixing corpus generated by our data augmentation method can be used to pre-train a code-mixing language model, we choose a natural language processing task for testing. According to Thara and Poornachandran (2018), people tend to using social media to share their opinions and thoughts, making code-mixing texts common on all kinds of social platforms in a multilingual society. We choose sentiment analysis of social media texts as our natural language processing task. We construct the first Bahasa Rojak sentiment analysis dataset, named SentiBahasaRojak, to evaluate the code-mixing model's performance on Bahasa Rojak sentiment analysis. This dataset contains three domains: product review, movie review, and stock review. To determine whether this code-mixing model can perform well on Bahasa Rojak and remain accurate in English or Malay, we have also compiled English and Malay datasets containing the same three domains.

3.4.1 English Sentiment Analysis Dataset

For English, we crawl product reviews from Kaggle[2] and use the IMDB dataset (Maas et al., 2011) as movie review data. As for stock reviews, we choose SemEval 2017 task 5 subtask 1 (Kar et al., 2017) and StockTwits[3]. The former is composed of financial microblogging data. Each post has been labeled with a value of -1 to 1, which corresponds to the most bearish (negative) to the most bullish (positive). To match other datasets, we convert the values into binary labels. StockTwits is a microblogging platform focusing on stock market discussions and supported by Twitter. According to StockTwits restrictions, users must label their posts as bullish or bearish. We collected all the posts of ten companies on StockTwits from 2016 to 2020.

3.4.2 Malay Sentiment Analysis Dataset

For Malay, we use the product review dataset and the movie review dataset from the Malay dataset (Husein, 2018). As for the stock review dataset, we hired experts to manually translate the dataset

[2] https://www.kaggle.com/bittlingmayer/amazonreviews

[3] https://stocktwits.com/

	BRCC Positive	BRCC Negative	KLSE Positive	KLSE Negative
Participant 1	408 (81.6%)	92 (18.4%)	438 (87.6%)	62 (12.4%)
Participant 2	442 (88.4%)	58 (11.6%)	377 (75.4%)	123 (24.6%)

Table 6: BRCC and KLSE each with 500 sentences assessed by native speakers (Participants 1 and 2). Positive means that the native speaker thinks the sentence is fluent and conforms to Bahasa Rojak's grammar, otherwise it is negative.

Dataset	**# of post**		
	Product Review	**Movie Review**	**Stock Forum**
English	2600	2600	1106
Malay	893	699	1106
sentiBahasaRojak	893	699	693

Table 7: The statistics of sentiment analysis datasets

Language	**# of tokens**
Malay (ms)	60,628,280
English (en)	1,867,773
Chinese (zh)	1,553,360
Undefined (other)	438,937

Table 8: BRCC's language auto-tagging statistics

from task 5 subtask 1 of SemEval 2017 to Malay.

3.4.3 Bahasa Rojak Sentiment Analysis Dataset (SentiBahasaRojak)

To the best of our knowledge, there is no public Bahasa Rojak sentiment analysis dataset. In this research, we employ different methods to construct three datasets of product reviews, movie reviews, and stock reviews, which collectively are called the SentiBahasaRojak sentiment analysis dataset. For product and movie reviews, since there are already publicly available Malay datasets, we use the data augmentation method mentioned in Algorithm 1 to generate Bahasa Rojak code-mixing datasets based on the Malay datasets.

For stock reviews, we intend to test the effectiveness of our Mixed XLM method on real data, so we crawl posts from Malaysia's financial and stock market websites, such as I3investor[4], and hired five native Malaysian experts who can read and write in Bahasa Rojak to manually annotate these posts. Experts must be able to distinguish the difference between bullish (positive) and bearish (negative). These five experts first annotated all the posts, and then majority voting was carried out to determine the final label of each post. The kappa value of the Bahasa Rojak stock review dataset is 0.77, which means that the annotations of these five experts have substantial agreement (Kasmuri and Basiron, 2019). Table 7 shows the number of posts in English, Malay, and Bahasa Rojak in the three domains.

[4] https://klse.i3investor.com/jsp/scl/community.jsp

4 Experiments and Results

In this section, we will explain our proposed method for dealing with code-mixing data. BERT and XLM are chosen as the baseline language models for comparison, and use a variety of sentiment analysis task datasets to evaluate each model.

We use the movie reviews, product reviews, and stock market comments described in the previous section as the experimental datasets. We will also report the accuracy and F1 score of each model for each dataset.

4.1 Baseline Model

We use the code-mixing data constructed in the previous section to pre-train the BERT and XLM models from scratch. Based on the results obtained from preliminary experiments, we remove the next sentence prediction (NSP) task when pre-training the BERT model.

We use the Masked Language Model (MLM) task to pre-train our baseline models. Due to the limitation of computing resources, our configuration is six layers, eight heads, 512 embedding dimensions, and the learning rate fixed at 2e-5. We use the Adam optimizer and adopt the early stopping method to terminate the training process.

4.2 Mixed XLM

In this work, we propose a new model called Mixed XLM. The main difference from vanilla XLM is that Mixed XLM automatically recognizes the language of each input token and handles code-mixing input, as shown in the Figure 1. In the Mixed XLM for Bahasa Rojak, we develop a language tagging algorithm to label the language of each word, as

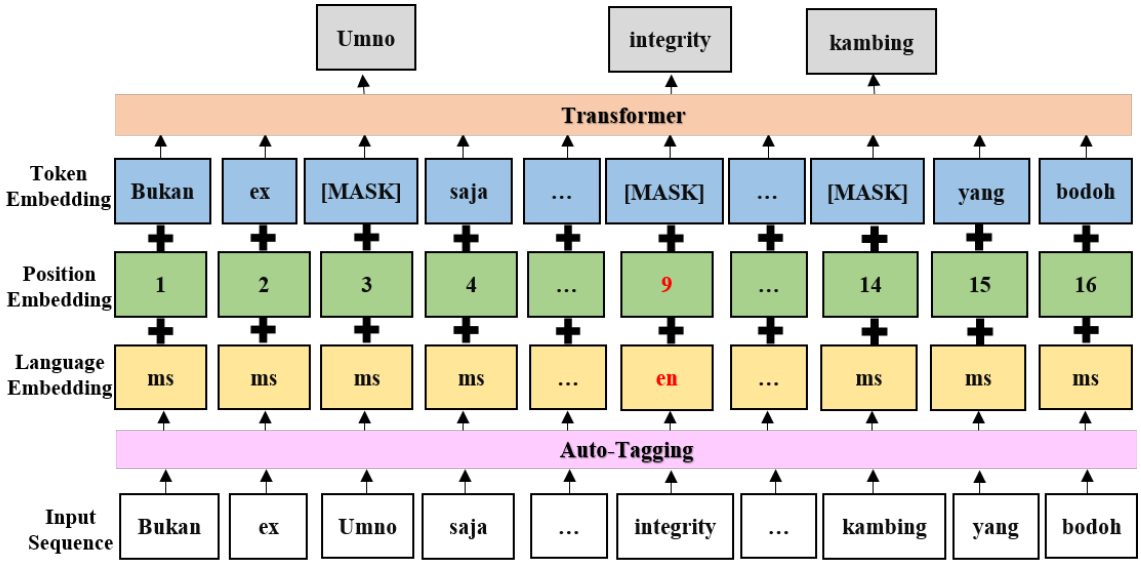

Figure 1: Input representation of Mixed XLM

shown in Algorithm 2. For each token *t*, the algorithm searches which language dictionary it is in. Suppose *t* is found in the dictionary of language *l*, then *t* is labeled as language *l*. Some words can be found in both Malay and English dictionaries, and language tagging labels them as Malay because these words are loanwords from English. We evaluated the language tagging module as having an accuracy of 0.973. The number of tokens for each language in our BRCC corpus after language tagging is shown in Table 8.

Algorithm 2: Language Tagging

Input :
Vocabulary of Malay words : V_{ms};
Vocabulary of Chinese words : V_{zh};
Vocabulary of English words : V_{en};
Input sentence : S_{cm};
Words in sentence : $S_{cm} = \{w^{(n)}\}_{n=1}^{N}$;

Output :
Language tagging : $langTag[N]$;

for $w^{(n)}$ *in* S_{cm} **do**
  if $w^{(n)}$ *in* V_{ms} **then**
    $langTag[n] = ms$
  else if $w^{(n)}$ *in* V_{zh} **then**
    $langTag[n] = zh$
  else if $w^{(n)}$ *in* V_{en} **then**
    $langTag[n] = en$
  else
    $langTag[n] = other$
  end
end

Finally, in the setting of our Mixed XLM, which is the same as BERT and XLM, we use six transformation layers, eight head layers, and 512 embedding dimensions, and use the masked language model as the pre-training task.

4.3 Evaluation of Pre-trained Language Models

We use the sentiment analysis task to evaluate our pre-trained language models. We also use 10-fold cross-validation to strengthen the credibility of the results. Table 9 shows the performance of our proposed models and baseline models, fine-tuned on SentiBahasaRojak. Remember that there are three domains in our SentiBahasaRojak, including product reviews, movie reviews, and stock market forums.

In Table 9, the best performing baseline model is XLM (EN-MS), which achieves 0.698 and 0.637 in accuracy ACC and F1 score, respectively. As for our proposed model Mixed XLM, it scores 0.718 and 0.666 when using only the Code-Mixing dataset (CM) for training. If the entire dataset (EN-MS-CM) is used for training Mixed XLM, 0.745 and 0.705 can be achieved, meaning it outperforms the best baseline model XLM (EN-MS) by 0.047 and 0.068. As shown in Table 9, we can observe that in the code-mixing sentiment analysis task, our proposed Mixed XLM achieves the best performance in all three datasets.

In addition, experiments are conducted on En-

Model	Product Review		Movie Review		Stock Market		Avg.	
	acc	f1	acc	f1	acc	f1	acc	f1
mBERT (CM)	0.652	0.603	0.661	0.653	0.563	0.496	0.625	0.584
mBERT (EN-MS-CM)	0.653	0.651	0.631	0.576	0.571	0.562	0.618	0.596
XLM (EN-MS)	0.658	0.592	0.764	0.751	0.672	0.568	0.698	0.637
Mixed XLM (CM)	0.703	0.671	0.761	0.746	0.690	0.581	0.718	0.666
Mixed XLM (EN-MS-CM)	**0.718**	**0.696**	**0.812**	**0.803**	**0.706**	**0.615**	**0.745**	**0.705**

Table 9: Results of different models on SentiBahasaRojak

Model	Product Review		Movie Review		Stock Market		Avg.	
	acc	f1	acc	f1	acc	f1	acc	f1
mBERT (CM)	0.801	0.793	0.691	0.663	**0.717**	**0.773**	0.736	0.743
mBERT (EN-MS-CM)	0.803	0.794	0.755	0.745	0.702	0.771	0.753	0.770
XLM (EN-MS)	0.813	0.812	0.701	0.689	0.675	0.746	0.730	0.749
Mixed XLM (CM)	0.807	0.804	0.792	0.771	0.643	0.712	0.747	0.762
Mixed XLM (EN-MS-CM)	**0.823**	**0.826**	**0.813**	**0.787**	0.677	0.743	**0.771**	**0.785**

Table 10: Results of different models on English

Model	Product Review		Movie Review		Stock Market		Avg.	
	acc	f1	acc	f1	acc	f1	acc	f1
mBERT (CM)	0.813	0.802	0.782	0.756	0.690	0.756	0.762	0.771
mBERT (EN-MS-CM)	0.815	0.743	0.780	0.782	0.683	0.765	0.759	0.763
XLM (EN-MS)	0.823	0.802	0.751	0.744	0.683	0.742	0.752	0.763
Mixed XLM (CM)	0.824	0.805	0.783	0.764	0.661	0.736	0.756	0.768
Mixed XLM (EN-MS-CM)	**0.828**	**0.818**	**0.805**	**0.785**	**0.696**	**0.769**	**0.776**	**0.791**

Table 11: Results of different models on Malay

glish and Malay monolingual datasets. We fine-tune all language models on monolingual datasets. Tables 10 and 11 show that our Mixed XLM model pre-trained on all corpora including BRCC achieves the highest average score on each language, which demonstrates the robustness of our approach.

5 Conclusion

In this paper, for Bahasa Rojak, we build a corpus called BRCC to pre-train Bahasa Rojak's language model, and compile a sentiment analysis dataset called SentiBahasaRojak. BRCC and SentiBahasaRojak are the first resources available for Bahasa Rojak in this area. We also propose a new pretrained model, Mixed XLM, which not only achieves the best performance on code-mixing data, but also maintains performance on monolingual data.

Our new Bahasa Rojak corpus is generated by our new data augmentation algorithm that recognizes three types of phrases in sentences and randomly selects some of those three phrases for translation to generate Bahasa Rojak sentences.

Our proposed Mixed XLM model is able to label input tokens to deal with code-mixing phenomena. As long as the Mixed XLM model is pre-trained on a code-mixing corpus, it can be used in downstream tasks containing code-mixing sentences, just as in this study, the Mixed XLM model was used in Bahasa Rojak's Sentiment Analysis.

Finally, we evaluate the Mixed XLM model pre-trained on BRCC through the sentiment analysis task on three different language settings (English, Malay, Bahasa Rojak). The sentiment analysis task includes three domains. The results show our Mixed XLM model achieves the best performance in all domains. In the monolingual setting experiment, Mixed XLM also achieves comparable performance, which proves the robustness of the model and the effectiveness of BRCC.

References

Ahmed Al-Saffar, S. Awang, Hai Tao, N. Omar, Wafaa Al-Saiagh, and M. Albared. 2018. Malay sentiment analysis based on combined classification approaches and senti-lexicon algorithm. *PLoS ONE*, 13.

SNAN Ariffin and Sabrina Tiun. 2020. Rule-based text normalization for Malay social media texts. *Interna-*

tional Journal of Advanced Computer Science and Applications, 11(10):156–162.

Mohd Syuhaidi Abu Bakar and Aliffluqman Mohd Mazzalan. 2018. Aliran pertuturan bahasa rojak dalam kalangan pengguna facebook di malaysia. *e-Academia Journal*, 7(1).

Steven Bird and Edward Loper. 2004. NLTK: The natural language toolkit. In *Proceedings of the ACL Interactive Poster and Demonstration Sessions*, pages 214–217, Barcelona, Spain. Association for Computational Linguistics.

Peter F. Brown, Vincent J. Della Pietra, Peter V. deSouza, Jenifer C. Lai, and Robert L. Mercer. 1992. Class-based *n*-gram models of natural language. *Computational Linguistics*, 18(4):467–480.

N. A. Bukhari, A. Anuar, Khairunnisa Mohad Khazin, and T. T. Aziz. 2015. English-Malay code-mixing innovation in Facebook among Malaysian university students. *Researchers World*, 6:01–10.

Khalifa Chekima and Rayner Alfred. 2017. Sentiment analysis of Malay social media text. In *International Conference on Computational Science and Technology*, pages 205–219. Springer.

Alexis Conneau, Kartikay Khandelwal, Naman Goyal, Vishrav Chaudhary, Guillaume Wenzek, Francisco Guzmán, Edouard Grave, Myle Ott, Luke Zettlemoyer, and Veselin Stoyanov. 2020. Unsupervised cross-lingual representation learning at scale. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 8440–8451, Online. Association for Computational Linguistics.

Alexis Conneau and Guillaume Lample. 2019. Cross-lingual language model pretraining. In *Advances in Neural Information Processing Systems*, volume 32. Curran Associates, Inc.

S. Deerwester, S.T. Dumais, G.W. Furnas, T.K. Landauer, and R.A. Harshman. 1990. Indexing by latent semantic analysis. *Journal of the American Society for Information Science 41*, pages 391–407.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, Minneapolis, Minnesota. Association for Computational Linguistics.

Noura Farra. 2019. *Cross-Lingual and Low-Resource Sentiment Analysis*. Columbia University.

Björn Gambäck and Amitava Das. 2016. Comparing the level of code-switching in corpora. In *Proceedings of the Tenth International Conference on Language Resources and Evaluation (LREC'16)*, pages 1850–1855, Portorož, Slovenia. European Language Resources Association (ELRA).

Souvick Ghosh, Satanu Ghosh, and Dipankar Das. 2017. Complexity metric for code-mixed social media text. *Computación y Sistemas*, 21.

Deepak Gupta, Asif Ekbal, and Pushpak Bhattacharyya. 2020. A semi-supervised approach to generate the code-mixed text using pre-trained encoder and transfer learning. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 2267–2280, Online. Association for Computational Linguistics.

Judy Woon Yee Ho et al. 2007. Code-mixing: Linguistic form and socio-cultural meaning. *The International Journal of Language Society and Culture*, 21(7):1–8.

Zolkepli Husein. 2018. Malay-dataset. `https://github.com/huseinzol05/Malay-Dataset`.

Mandar Joshi, Eunsol Choi, Daniel Weld, and Luke Zettlemoyer. 2017. TriviaQA: A large scale distantly supervised challenge dataset for reading comprehension. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1601–1611, Vancouver, Canada. Association for Computational Linguistics.

Sudipta Kar, Suraj Maharjan, and Thamar Solorio. 2017. Ritual-uh at semeval-2017 task 5: Sentiment analysis on financial data using neural networks. In *Proceedings of the 11th International Workshop on Semantic Evaluation (SemEval-2017)*, pages 877–882, Vancouver, Canada. Association for Computational Linguistics.

Emaliana Kasmuri and Halizah Basiron. 2019. Building a Malay-English code-switching subjectivity corpus for sentiment analysis. *Int. J. Advance Soft Compu. Appl*, 11(1).

Mohamad Nizam Kassim, Shaiful Hisham Mat Jali, Mohd Aizaini Maarof, Anazida Zainal, and Amirudin Abdul Wahab. 2020. Enhanced text stemmer with noisy text normalization for Malay texts. In *Smart Trends in Computing and Communications*, pages 433–444, Singapore. Springer Singapore.

Yash Kumar Lal, Vaibhav Kumar, Mrinal Dhar, Manish Shrivastava, and Philipp Koehn. 2019. De-mixing sentiment from code-mixed text. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics: Student Research Workshop*, pages 371–377, Florence, Italy. Association for Computational Linguistics.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Roberta: A robustly optimized BERT pretraining approach. Cite arxiv:1907.11692.

Andrew L. Maas, Raymond E. Daly, Peter T. Pham, Dan Huang, Andrew Y. Ng, and Christopher Potts. 2011. Learning word vectors for sentiment analysis. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, pages 142–150, Portland, Oregon, USA. Association for Computational Linguistics.

Bryan McCann, James Bradbury, Caiming Xiong, and Richard Socher. 2017. Learned in translation: Contextualized word vectors. In *Proceedings of the 31st International Conference on Neural Information Processing Systems*, NIPS'17, page 6297–6308, Red Hook, NY, USA. Curran Associates Inc.

Tomas Mikolov, Kai Chen, Greg S. Corrado, and Jeffrey Dean. 2013. Efficient estimation of word representations in vector space.

Adithya Pratapa, Gayatri Bhat, Monojit Choudhury, Sunayana Sitaram, Sandipan Dandapat, and Kalika Bali. 2018a. Language modeling for code-mixing: The role of linguistic theory based synthetic data. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1543–1553, Melbourne, Australia. Association for Computational Linguistics.

Adithya Pratapa, Monojit Choudhury, and Sunayana Sitaram. 2018b. Word embeddings for code-mixed language processing. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 3067–3072, Brussels, Belgium. Association for Computational Linguistics.

Libo Qin, Minheng Ni, Yue Zhang, and Wanxiang Che. 2020. CoSDA-ML: Multi-lingual code-switching data augmentation for zero-shot cross-lingual nlp. In *Proceedings of the Twenty-Ninth International Joint Conference on Artificial Intelligence, IJCAI-20*, pages 3853–3860. International Joint Conferences on Artificial Intelligence Organization.

XiPeng Qiu, TianXiang Sun, YiGe Xu, YunFan Shao, Ning Dai, and XuanJing Huang. 2020. Pre-trained models for natural language processing: A survey. *Science China Technological Sciences*, 63(10):1872–1897.

Pranav Rajpurkar, Jian Zhang, Konstantin Lopyrev, and Percy Liang. 2016. SQuAD: 100,000+ questions for machine comprehension of text. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 2383–2392, Austin, Texas. Association for Computational Linguistics.

Sebastian Ruder, Matthew E. Peters, Swabha Swayamdipta, and Thomas Wolf. 2019. Transfer learning in natural language processing. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Tutorials*, pages 15–18, Minneapolis, Minnesota. Association for Computational Linguistics.

Mohammad Arshi Saloot, Norisma Idris, and Rohana Mahmud. 2014. An architecture for Malay tweet normalization. *Information Processing & Management*, 50(5):621–633.

Norlela Samsudin, Mazidah Puteh, Abdul Razak Hamdan, and Mohd Zakree Ahmad Nazri. 2013. Mining opinion in online messages. *Int. J. Adv. Comput. Sci. Appl*, 4(8).

Hidayah Shafiee, Ahmad Fahmi Mahamood, Abdul Rahman Abdul Manaf, Tengku Kastriafuddin Shah Tengku Yaakob, Abdul Jalil Ramli, Zuraidi Ahmad Mokhdzar, Jamsari Jamaluddin, Maskor Bajuri, and Mohd Erpi Mohd Ali. 2019. Pengaruh bahasa rojak di media baharuterhadap bahasa kebangsaan. *International Journal of Law, Government and Communication*, 4(15):141–153.

Richard Socher, Alex Perelygin, Jean Wu, Jason Chuang, Christopher D. Manning, Andrew Ng, and Christopher Potts. 2013. Recursive deep models for semantic compositionality over a sentiment treebank. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*, pages 1631–1642, Seattle, Washington, USA. Association for Computational Linguistics.

S. Thara and P. Poornachandran. 2018. Code-mixing: A brief survey. *2018 International Conference on Advances in Computing, Communications and Informatics (ICACCI)*, pages 2382–2388.

Ralf Vollmann and Soon Tek Wooi. 2019. The sociolinguistic status of Malaysian English. *Grazer Linguistische Studien*, 91:133–150.

A Appendix

A.1 Detail of Algorithm 1

Having three different languages (Malay, English and Chinese) in a sentence is one of the features of Bahasa Rojak. In general, the matrix language in Bahasa Rojak is either English or Malay, while the other of that pair and Chinese serve as inserted language independently or jointly. Considering the characteristics of Bahasa Rojak, we collect both English and Malay data in our source language, and then translate them to our target sentences in Bahasa Rojak that consist of at least one matrix and inserted language.

In Algorithm 1, we represent the source language as sl, and $sl \leftarrow \{en, ms\}$ to denote which language is the source language. The same concept applies to target language, tl represents the target language, and $tl \leftarrow \{en : [ms, zh], ms : [en, zh]\}$ means that in the source language en, we hope to translate to a target language, ms or zh. Next, we set the substitution ratio manually, in which α represents the probability of translating a source sentence to a code-mixed sentence. β means how many phrases need to be translated during the process. Finally, γ means the probability of translating a source sentence to a target sentence completely. Note that the target language is randomly selected in our algorithm.

In our augmentation algorithm, there are two important functions: $get\_phrase()$ and $Replace\_and\_Alignment()$. In the $get\_phrase()$ function, we first use the NLTK for syntax analysis, parse each input sentence, and then use POS tagging to label the phrases. In this way, we rephrase the sentence from a word-based to phrase-based tokenization structure. After building the sentence structure, we use the regular expression patterns defined in Table 4 to extract specific phrases to translate.

As we reconstruct our Bahasa Rojak sentences in the phrase-based tokenization manner, we have to implement two kinds of phase alignment methods to perfect our code-mixing sentences. Therefore, through aligning the sentence index and replacing the translated phrase in the source language, we successfully generate our Bahasa Rojak sentences.

A.2 Code-Mixing Complexity

Due to time constraints, we only sample 1% of the data from our BRCC dataset and evaluate it with the following metrics.

Switch-Point Fraction (SPF) The switch point refers the point in a sentence where two adjacent tokens are in different languages. We follow the definition proposed by Pratapa et al. (2018a), but make slight adjustments to fit our BRCC corpus. We calculate the number of switch points in a sentence, and divide it by the total number of phrase boundaries.

Code-Mixing Index (CMI) CMI is used to measure the amount of code-mixing in a corpus to account for the language distribution (Gupta et al., 2020), which was proposed by Ghosh et al. (2017); Gambäck and Das (2016). In our BRCC dataset, we use the CMI formula from Pratapa et al. (2018a), as follows:

$$C_u(x) = \frac{(N(x) - \max_{L_i \in L}\{t_{L_i}\}(x)) + P(x)}{N(x)},$$

where N denotes the number of language tokens, x is an utterance; t_{L_i} represents the tokens in language Li, P is the number of code-switching points in utterance x. Then, we compute our data at the sentence level by averaging all sentences sampled from the BRCC dataset.

Our SPF and CMI values are 0.158 and 0.384 respectively.

WordNet-QU: Development of a Lexical Database for Quechua Varieties

Nelsi Melgarejo
Pontifical Catholic University of Peru
nelsi.melgarejo@pucp.edu.pe

Rodolfo Zevallos
Pompeu Fabra University
rodolfojoel.zevallos@upf.edu

Héctor Gómez
Pontifical Catholic University of Peru
hector.gomez@pucp.edu.pe

John E. Ortega
Northeastern University
j.ortega@northeastern.edu

Abstract

In the effort to minimize the risk of extinction of a language, linguistic resources are fundamental. Quechua, a low-resource language from South America, is a language spoken by millions but, despite several efforts in the past, still lacks the resources necessary to build high-performance computational systems. In this article, we present **WordNet-QU** which signifies the inclusion of Quechua in a well-known lexical database called *wordnet*. We propose **WordNet-QU** to be included as an extension to wordnet after demonstrating a manually-curated collection of multiple digital resources for lexical use in Quechua. Our work uses the *synset* alignment algorithm to compare Quechua to its geographically nearest high-resource language, Spanish. Altogether, we propose a total of 28,582 unique synset IDs divided according to region like so: 20510 for Southern Quechua, 5993 for Central Quechua, 1121 for Northern Quechua, and 958 for Amazonian Quechua.

1 Introduction and related work

Lexical databases and resources have been used in the past for various natural language processing (NLP) tasks ranging from information retrieval (IR) to machine translation (MT). While many recent NLP approaches rely on deep learning techniques like transformers, namely BERT (Devlin et al., 2018), where attention (Vaswani et al., 2017) is used to create a semantic representation of text, more traditional approaches relied on purely linguistic and syntactic features. More often than not, recent deep-learning approaches require a large amount of data to perform better than traditional ones (e.g. on the order of millions of words for machine translation (Koehn and Knowles, 2017; Bahdanau et al., 2014)). This makes NLP approaches with low-resource languages, languages that are measured in the thousands typically, much more difficult to solve with recent approaches thus forcing the use of traditional approaches to solve problems.

One low-resource language from South America, called Quechua, is spoken by nearly 8 million people[1] yet still does not have enough resources to effectively compete with other high-resource languages as has been shown in previous research (Ebrahimi et al., 2021; Ortega et al., 2021, 2020). Oftentimes, due to insufficient resources, scores such as BLEU (Papineni et al., 2002) and accuracy are more than three times lower. This lack of resources thus drives the need for traditional techniques such as the use of lexical databases, grammars, and other linguistic cues such as tree banks and more. One such resource that has been commonly used for traditional approaches is called *wordnet* (Fellbaum, 1998) which was originally created in the 1990s yet is still used today, especially for low-resource languages like Quechua.

The need to build digital resources is greater for endangered languages like Quechua and others since there is a clear desire to save the language from extinction. However, the desire is typically not supported by those agencies that are responsible for its survival. Berment (Berment, 2002) and others have expressed the need for further analysis and research stating that the current effort "may be insufficient to aid preservation efforts". In this work, we provide several lexical resources for Quechua to increase its inclusion in wordnet (Fellbaum, 1998). We call the collection of resources **WordNet-QU** which corresponds to its commonly-used language-pair symbol (QU) found in most corpora for NLP in Quechua. To elaborate on its inclusion, in Section 2 we provide details on how the corpus was compiled and the annotations done. Then, in Section 3, we cover the wordnet implementation of the corpus. Finally, in Section 4 we provide insight into our future downstream tasks.

[1] https://en.wikipedia.org/wiki/Quechuan_languages

Proceedings of the 29th International Conference on Computational Linguistics, pages 4429–4433
October 12–17, 2022.

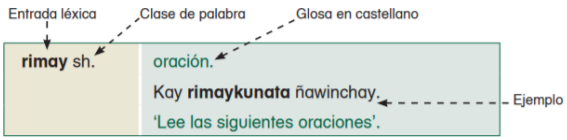

Figure 1: Example of the structure of the some dictionaries.

2 Corpus details

The corpus presented here made available publicly[2] has been created using a manually curated collection of dictionaries. The dictionaries were mostly gotten from Ministry of Education in Peru (MINEDU in Table 4, see Appendix) and consist of five regional varieties of Quechua (Southern Quechua (Collao), Southern Quechua (Chanka), Central Quechua, Northern Quechua, and Amazonian Quechua) ranging from 1976 to 2005 in the years they were collected.

In order to organize the dictionaries into a format that can be used by wordnet (Fellbaum, 1998), the corpus is structured in a format that consists of the following labels: (i) branch, (ii) variety, (iii) region, (iv) author, (v) dictionary, (vi) year, (vii) lexical entry, (viii) grammatical category, (ix) glossary entry, (x) Quechua definition, (xi) Quechua synonym, (xii) Spanish synonym and (xiii) notes or clarifications. An example of the original dictionary as found from the Ministry of Education is seen in Figure 1.

Since there are several dialects of Quechua spoken in Peru (Cerrón-Palomino, 2021), it was important to compile the corpus by variety or region. In order to better illustrate the differences in parts of speech for dialects, we break each region's dialect into the following categories: noun, adjective, adverb and verb as shown in Table 1. The variety with the highest lexical entries are Southern Quechua (South) followed by Central, Northern (North), and Amazonian (Amaz). For each of the parts of speech and varieties of Quechua, there is a corresponding Spanish glossary entry. Additionally, for the southern and central varieties, apart from the part of speech and glossary, there is a definition in Quechua and translation in Spanish. In some cases the translation is gotten from MINEEDU and in other cases native speakers translated for us.

[2] https://github.com/Llamacha/wordnet-qu

POS	Quechua variety			
	South	Central	North	Amaz
Noun	16 717	3 241	579	537
Verb	8 000	3 145	506	423
Adjective	4 116	904	157	160
Adverb	985	384	126	48
Total	**29818**	**7677**	**1368**	**1204**

Table 1: Number of words per part of speech (POS) for each Peruvian region.

3 Methodology

In order to use and distribute **WordNet-QU** we had to make it compatible with wordnet (Fellbaum, 1998). Constructing a wordnet, whether from scratch or by expanding a previous one, is a labor intensive process that requires several steps and extensive use of both human labor and automated systems. Since the creation of the first wordnet (Princeton WordNet (PWN)) in 1995 (Miller, 1995), many other wordnets have been created for several languages. For example EuroWordNet (EWN) is a multilingual wordnet project that links wordnets of multiple European languages (English, Dutch, Italian, Spanish, German, French, Czech and Estonian) (Vossen, 1997). In EWN, wordnets were created for each language separately and then linked through an index based on PWN. In the same way, BalkaNet is a multilingual wordnet project consisting of six Balkan languages (Bulgarian, Czech, Greek, Romanian, Serbian, and Turkish). (Tufis et al., 2004)

Two of the most-commonly used approaches for creating a wordnet are based on what are known as the *expand* and *merge* approaches. Both approaches use *synsets* – groups of synonyms that express the same concept in wordnet. One synset can have multiple words and one word can have multiple synsets. In the *expand* approach, a set of synsets from PWN, including their semantic database, are first translated into the target language and then relations are transferred from English and checked in a manual fashion as is done for Scottish Gaelic (Bella et al., 2020) and the French (Sagot and Fišer, 2008). The *merge* approach builds bilingual relations from scratch, without any links to English, the main language for wordnet. Both the Polish wordnet (Derwojedowa et al., 2008) and Norwegian wordnet (Fjeld and Nygaard, 2009) use the merge approach.

Model	Size	Spearman
Pre-trained Model (Wiki)	29k	0.35
WordNet-QU (Wiki + WordNet Corpus)	31k	**0.61**

Table 2: A comparison of Spearman correlation coefficients (Wissler, 1905) between human judgement and similarity scores for pre-trained model on tokens of Wikipedia alone and Wikipedia with the WordNet-QU corpus.

Our implementation is based on a few steps. The first step is to construct a wordnet for Spanish because translations for Quechua are more available in the high-resource language (Spanish) in Peru. Using the main wordnet in English, we create a Spanish wordnet using the expansion technique described above based on similarity alone. The abundance of on-line Spanish glossaries and other relationships helped when creating the Spanish wordnet. Once translated, the Spanish wordnet became what is known as our multi-lingual central repository (MCR) for Quechua. This, in turn, facilitates the next steps which are to create and align synsets to with their corresponding concept which is validated manually by a human.

3.1 Synset alignment

The most important part of creating a wordnet is the alignment of synsets to their main concept. Our algorithm focuses on a straightforward process. First, the algorithm iterates through the entire wordnet MCR in Spanish for each word from the Quechua corpus.[3] When an exact Quechua–Spanish match is found and verified (manually), all of the related words from the Quechua vocabulary are mapped to their corresponding Spanish concept. This process constitutes the creation of a Quechua synset for one or more words that exist in their Spanish counterpart. After the synset creation, part-of-speech tags are created according to their grammatical category.

3.2 Wordnet validation

In order to validate the feasibility of **WordNet-QU**, we measure the cosine similarity distance between two FastText (Grave et al., 2018) models: (1) a baseline model[4] based on Wikipedia[5] which contains Quechua text and (2) a model based on Wikipedia with the addition of the **WordNet-QU** corpus. Our FastText (Grave et al., 2018) model is trained using 31 thousand tokens and identical hyper-parameters and algorithm as the baseline (skipgram algorithm, an embedding size of 300 dimensions, a context window size of 5, and n-grams ranging from 3 to 6 characters). The cosine similarity is measured for a 1000 randomly collected synsets. The distance results are then compared to the annotator's yes/no decision of whether or not each synset corresponds to the words from **WordNet-QU**. Human judgement is found to correspond much higher with the WordNet-QU model than the pre-trained model as shown in Table 2. We leave further improvement for future work.

4 Results and future work

Variety	**Synsets**	**Def.**	**Sent.**
Southern	20 510	1 873	1 827
Central	5 993	1 191	1 191
Northern	1 121	-	-
Amazonian	958	-	-
Total	**28 582**	**3 064**	**3 018**

Table 3: A count of synsets, definitions, and sentences per variety.

We have presented the process and resources used to create a wordnet-based resource for Quechua called **WordNet-QU**. We use fastText embeddings as a manner of measuring the similarity between Quechua words and Spanish concepts which provides nearly the 29k synsets illustrated in Table 3. We make the synsets and various lexicons created available publicly. For more details on specific dialects and other information related to our processing, please consult the Appendix.

This research was focused on the development of a Quechua wordnet using synonyms between different varieties of Quechua. The dictionaries used from different sources had to be identified for there region and dialect which became an after-the-fact asset to our work.

Future lines of investigations are based on work that is planed with several renown authors in

[3] Translations from Quechua to Spanish are performed beforehand.

[4] https://fasttext.cc/docs/en/crawl-vectors.html

[5] https://www.wikipedia.org/

the field of NLP processing of Quechua to use **WordNet-QU** in downstream tasks. Some of the NLP approaches that are currently in discussion are **WordNet-QU** for Quechua–Spanish translation and **WordNet-QU** for POS tagging in treebanks.

Acknowledgements

This work has been partially supported by the Project PID2019-104512GB-I00, Ministerio de Ciencia e Innovación and Agencia Estatal de Investigación (Spain).

References

Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. 2014. Neural machine translation by jointly learning to align and translate. *arXiv preprint arXiv:1409.0473*.

Gábor Bella, Fiona McNeill, Rody Gorman, Caoimhín Ó Donnaíle, Kirsty MacDonald, Yamini Chandrashekar, Abed Alhakim Freihat, and Fausto Giunchiglia. 2020. A major wordnet for a minority language: Scottish gaelic. In *Proceedings of the 12th Language Resources and Evaluation Conference*, pages 2812–2818.

Vincent Berment. 2002. Several directions for minority languages computerization. In *COLING 2002: The 17th International Conference on Computational Linguistics: Project Notes*.

Rodolfo Cerrón-Palomino. 2021. The languages of the inkas. In *The Inka Empire*, pages 39–54. University of Texas Press.

Magdalena Derwojedowa, Maciej Piasecki, Stanis[a]aw Szpakowicz, Magdalena Zawisławska, and Bartosz Broda. 2008. Words, concepts and relations in the construction of polish wordnet. *Proceedings of GWC 2008*, pages 162–177.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*.

Abteen Ebrahimi, Manuel Mager, Arturo Oncevay, Vishrav Chaudhary, Luis Chiruzzo, Angela Fan, John Ortega, Ricardo Ramos, Annette Rios, Ivan Vladimir, et al. 2021. Americasnli: Evaluating zero-shot natural language understanding of pretrained multilingual models in truly low-resource languages. *arXiv preprint arXiv:2104.08726*.

C Fellbaum. 1998. Wordnet: Wiley online library. *The Encyclopedia of Applied Linguistics*, 7.

Ruth Vatvedt Fjeld and Lars Nygaard. 2009. Nornet-a monolingual wordnet of modern norwegian. In *NODALIDA 2009 workshop: WordNets and other Lexical Semantic Resources-between Lexical Semantics, Lexicography, Terminology and Formal Ontologies*, volume 7, pages 13–16.

Edouard Grave, Piotr Bojanowski, Prakhar Gupta, Armand Joulin, and Tomas Mikolov. 2018. Learning word vectors for 157 languages. *arXiv preprint arXiv:1802.06893*.

Philipp Koehn and Rebecca Knowles. 2017. Six challenges for neural machine translation. *arXiv preprint arXiv:1706.03872*.

George A Miller. 1995. Wordnet: a lexical database for english. *Communications of the ACM*, 38(11):39–41.

John E Ortega, Richard Castro Mamani, and Kyunghyun Cho. 2020. Neural machine translation with a polysynthetic low resource language. *Machine Translation*, 34(4):325–346.

John E Ortega, Richard Alexander Castro-Mamani, and Jaime Rafael Montoya Samame. 2021. Love thy neighbor: Combining two neighboring low-resource languages for translation. In *Proceedings of the 4th Workshop on Technologies for MT of Low Resource Languages (LoResMT2021)*, pages 44–51.

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. Bleu: a method for automatic evaluation of machine translation. In *Proceedings of the 40th annual meeting of the Association for Computational Linguistics*, pages 311–318.

Benoît Sagot and Darja Fišer. 2008. Building a free french wordnet from multilingual resources. In *OntoLex*.

Dan Tufis, Dan Cristea, and Sofia Stamou. 2004. Balkanet: Aims, methods, results and perspectives. a general overview. *Romanian Journal of Information science and technology*, 7(1-2):9–43.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. *Advances in neural information processing systems*, 30.

Piek J.T.M. Vossen. 1997. Eurowordnet: a multilingual database for information retrieval. In *Proceedings of the DELOS workshop on Cross-language Information Retrieval, March 5-7, 1997 Zurich*. Vrije Universiteit.

Clark Wissler. 1905. The spearman correlation formula. *Science*, 22(558):309–311.

Variety of Quechua	Dictionary	Author	Year
Southern (Collao)	Yachakuqkunapa Simi Qullqa	MINEDU	2005
	Diccionario quechua: Cuzco– Collao.		2005
Southern (Chanka)	Yachakuqkunapa Simi Qullqa	MINEDU	2005
	Diccionario quechua: Cuzco– Chanka		2005
Central	Chawpi Qichwapa Chimi Qullqan	MINEDU	2017
	Yachachinapaq shimikunachawpin qichwa		2005
Northern	Diccionario quechua: Cajamarca – Cañaris	MINEDU	1976
Amazonian	Shimikunata asirtachik killka Inka Castellanu	Inst. ling. de verano	2002

Table 4: Dictionaries used for the construction of the corpus.

N°	Grammatical category			
	Noun	**Verb**	**Adjective**	**Adverb**
1	9 190	4 682	2 924	474
2	1 186	578	382	32
3	374	211	112	11
4	134	88	26	8
5	37	32	9	2
6	10	3	4	1
Total	**10 931**	**5 594**	**3 457**	**528**

Table 5: Number of words per sense for each grammatical category of Southern Quechua wordnet.

N°	Grammatical category			
	Noun	**Verb**	**Adjective**	**Adverb**
1	1 702	1 974	603	210
2	362	292	136	30
3	95	67	33	15
4	19	14	8	3
5	4	2	1	3
Total	**2 182**	**2 349**	**781**	**261**

Table 6: Number of words per sense for each grammatical category of Central Quechua wordnet.

N°	Grammatical category			
	Noun	**Verb**	**Adjective**	**Adverb**
1	382	372	123	31
2	38	12	26	16
Total	**424**	**384**	**149**	**47**

Table 7: Number of words per sense for each grammatical category of Amazonian Quechua wordnet.

N°	Grammatical category			
	Noun	**Verb**	**Adjective**	**Adverb**
1	439	392	141	71
2	21	40	5	10
Total	**460**	**433**	**146**	**82**

Table 8: Number of words per sense for each grammatical category of Northern Quechua wordnet.

When the Student Becomes the Master: Learning Better and Smaller Monolingual Models from mBERT

Pranaydeep Singh and Els Lefever
LT3, Language and Translation Technology Team
Department of Translation, Interpreting and Communication – Ghent University
Groot-Brittanniëlaan 45, 9000 Ghent, Belgium
{firstname.lastname}@ugent.be

Abstract

In this research, we present pilot experiments to distil monolingual models from a jointly trained model for 102 languages (mBERT). We demonstrate that it is possible for the target language to outperform the original model, even with a basic distillation setup. We evaluate our methodology for 6 languages with varying amounts of resources and belonging to different language families.

1 Introduction

The introduction of the Transformer architecture, which aims to solve sequence-to-sequence tasks while also handling long-range dependencies relying on self-attention (Vaswani et al., 2017), has caused a huge improvement on the state-of-the-art for a wide range of natural language processing tasks. Despite the clear gain in performance, these pre-trained language models are extremely data- and computation-hungry and their sizes keep growing at an incredible speed. For example, RoBERTa (Liu et al., 2019) is trained on a text corpus of 160 GB, and the recent collaboration of Microsoft and Nvidia resulted in a language model containing 530 billion parameters that was trained on 270 billion tokens (Smith et al., 2022). Besides the ethical questions on sustainability raised by the NLP community, these huge language models also pose a lot of operational challenges, as they require massive amounts of training data, computation power and storage capacity. More importantly, the limits on training data also introduce limits on the languages these models can work with. Consequently, English has been the default language newer transformers have been trained on. More recently, much attention has been devoted to multilingual aspects, especially with the advent of joint models like mBERT (Devlin et al., 2019) and XLM (Conneau and Lample, 2019). However, the percentage of data used to train joint models for low-resourced languages when compared to English is orders of magnitudes lesser and results in poorer representations for lower-resourced languages (Wu and Dredze, 2020). Moreover, when language representations for a specific low-resourced language are needed, there are no available monolingual models for a lot of these languages, and the entire large jointly-trained model needs to be loaded. More recently, a lot of attention is paid to sustainability for transformers, resulting in various approaches like pruning, weight sharing and distillation to reduce the size of large models. Previous research has shown good results for task-specific distillation, distillation from larger English BERT models and distillation from mBERT into a smaller multilingual model (see Section 2). In this paper, we attempt to combine these two research directions and distil smaller monolingual transformers from mBERT. Our hypothesis was that distilling would help to improve the representations for a target language, as the model can focus its prediction power on the target language, instead of attempting to accommodate 101 other languages. To the best of our knowledge, this is the first research presenting results for distilling monolingual student models from mBERT. Not only are we able to successfully distil smaller monolingual models from mBERT, we also demonstrate that these smaller models outperform mBERT for the distilled language. We experiment with student models for well-resourced (Dutch, French), middle-resourced (Hindi, Hebrew) and low-resourced (Swahili, Slovenian) languages from very diverse language families and attempt to understand how a student model can outperform a teacher model in a distillation setup.

2 Related Research

While models that generate Deep Contextualized Representations like ELMo (Peters et al., 2018), BERT (Devlin et al., 2019) and GPT (Brown

Proceedings of the 29th International Conference on Computational Linguistics, pages 4434–4441
October 12–17, 2022.

et al., 2020) have pushed the state-of-the-art for downstream tasks in English, the improvements for medium- and low-resourced languages have not been as significant. Trained models on large monolingual corpora are abundant for English and other Western European languages, but they are extremely scarce for under-resourced languages (having small Wikipedias to train language models on). Although joint models trained for multiple languages like mBERT (Devlin et al., 2019) and XLM (Conneau and Lample, 2019) have been an excellent alternative to monolingual models, and even perform better than monolingual models due to additional supervision, these large joint models also come with drawbacks. Wu et al. (2020) show that a majority of languages in mBERT are under-represented and have poor performance on downstream tasks. This can be attributed to multiple reasons, like minimal monolingual data compared to English and other Western European languages, a small percentage share when it comes to mBERT's shared vocabulary and the sub-word tokenization not being suitable for all scripts. However, mBERT still performs better than monolingual models for these languages, since the monolingual models are trained on very limited pre-training data compared to mBERT. Therefore, with monolingual transformers either not existing or performing worse than mBERT, one is forced to use mBERT for monolingual tasks for a particular low-resourced language, and thus deploying representations for 101 other languages. Abdaoui et al. (2020) propose a simple alternative by loading specific sections of mBERT's vocabulary for particular languages, causing no loss in performance while decreasing the size of the deployed model considerably.

Various methodologies have been proposed to improve the sustainability of transformers. Distillation methods have shown to be very successful, but also low-rank approximation (Chen et al., 2021), weight sharing (Reid et al., 2021), pruning (Lagunas et al., 2021) and quantization (Bondarenko et al., 2021) propose excellent alternatives. While the central idea behind distillation is to reduce the model size as much as feasible while keeping the performance as close to the original model as possible, task-specific distillation (Tang et al., 2019) is a variant which focuses on a certain subset of the model's capabilities. The broader concept behind both methodologies is similar: a student model is initialised with significantly lesser parameters than the original teacher model. The student is trained with the loss for the objective at hand (masked language modeling (MLM) for generic distillation and specific tasks for task-based distillation), while also forcing the predictions to be identical to the teacher model. The underlying hypothesis being that once a larger model has learned the nuances of the data, a much smaller model can simply mimic the predictions with significantly lesser parameters. Hinton et al. (2015) further introduced the softmax temperature to emphasize learning from the entire distribution. A seminal work in the distilation of BERT-like transformers was performed by Sanh et al. (2019), who report a considerable model size reduction (40%) while retaining 97% performance on downstream tasks.

In this paper, we present pilot experiments for a new type of knowledge distillation that aims to extract representations for a particular language from a multilingual model. While there have been more advanced distillation approaches proposed, such as Patient Knowledge Distillation (Sun et al., 2019), this research is a first attempt at language-based distillation, where the basic distillation setup is validated as a proof of concept. We demonstrate that even when using basic distillation, it is possible to obtain monolingual models 3 times smaller than mBERT, while also performing consistently better on the distilled language. In addition, we show promising results for languages with a large variety in terms of available resources and belonging to different language families.

3 Distillation Methodology and Experimental Setup

We use the basic, proven distillation technique by Sanh et al. (2019), which uses mBERT as a teacher for all setups, and a 6-layer BERT architecture without the pooler and token-type embeddings as the student. We also use the 3 loss functions proposed, which are formally represented in the equations below. The distillation loss (1) ensures the similarity between the teacher output distribution t_i and the student output distribution s_i using negative log-likelihood. The cosine embedding loss (2) ensures the distributions not only have similar magnitudes but a similar directional alignment as well by penalising cosine distance. The final loss (3) is the standard cross-entropy used for most modern MLM systems, where s_i is once again the

[1]https://github.com/NirantK/hindi2vec

	Wikipedia	Downstream Task 1	Downstream Task 2	Monolingual Model
French	12M	Sentiment (Le et al., 2019)	UD POS (GSD)	CamemBERT-base
Dutch	4.4M	Sentiment (van der Burgh (2019))	UD POS (Lassy-small)	BERTje
Hindi	1.1M	News (hindi2vec: *github NirantK/hindi2vec*)	UD POS (HDTB)	indicBERT
Hebrew	1.3M	Sentiment (Amram et al., 2018)	UD POS (HTB)	AlephBERT
Swahili	0.1M	News (SNCD: *huggingface datasets/swahili_news*)	NER (Adelani et al., 2021)	SwahBERT
Slovene	0.4M	NER (Rahimi et al., 2019)	UD POS (SSJ)	SloBERTa

Table 1: Overview of the used resources for each of the 6 languages, including the monolingual Wikipedia (in million pages) used for the distillation process, as well as the two tasks used to evaluate the distilled model. For all PoS-tagging tasks (*UD POS*), datasets were retrieved from the Universal Dependencies dataset (*https://universaldependencies.org*). The last column lists the monolingual model that was used to present an upperbound score for the given task and language.

student's output distribution while y_i is the ground truth output distribution. The three losses are then aggregated with a weighted sum (4). An important factor here is the unlabelled monolingual data used for distillation. While it is often the case for distillation systems to pre-train the student model on the entire corpus to ensure optimal transfer, we want to focus on the capabilities of a single target language, and therefore only distil for $i \epsilon L$ where L is the target language in question.

$$L_{distillation} = \sum_{i \epsilon L} t_i * log(s_i) \quad (1)$$

$$L_{cosine} = \sum_{i \epsilon L} 1 - cos(t_i, s_i) \quad (2)$$

$$L_{mlm} = \sum_{i \epsilon L} y_i * log(s_i) \quad (3)$$

$$L = \alpha_1 L_{distillation} + \alpha_2 L_{mlm} + \alpha_3 L_{cosine} \quad (4)$$

A vital thing to note here is that L_{mlm} and $L_{distillation}$ can be slightly contradictory. While both enforce the same obejctive, L_{mlm} uses the ground truth while $L_{distillation}$ uses the teacher's predictions. Therefore, if the teacher predictions are often dissimilar to the ground-truth, the two losses might interfere with each other's progress. We explore this further in Section 4.

We also follow in the footsteps of the previous work for the student initialisation. It was noted by Sun et al. (2019) that initialising the student model from the teacher model by skipping alternate layers greatly speeds up the learning process and improves the student model considerably. Our final step was to reduce the vocabulary of the student model to only the target language L. Since the student vocabulary was initialized from the teacher mBERT, which contains sub-words from over a 100 languages, this majorly contributes to the large vocabulary, and therefore directly to the model size and inference speed. We use the approach suggested by Abdaoui et al. (2020), which selects a subset of the vocabulary of a large model and eliminates the unnecessary parameters from the embedding layer and tokenizer, thus significantly bringing down model parameters.

We replicate the distillation experiments for a set of 6 languages, using the entire available Wikipedia for each language. Since the data size varies considerably (ranging between 0.1 million pages for Swahili to 12 million pages for French), we train for 20,000 steps to ensure all models are trained to a similar extent. The student models for Dutch and French might therefore be improved by further training since they have more available data, but our objective here was to focus on the viability of the approach for low-resourced languages. Experiments with the high-resourced languages (Dutch, French) are only added for the sake of comparison. We use the values of 5.0 for α_1, 2.0 for α_2 and 1.0 for α_3, respectively. A starting learning rate of $5e-4$ was used, with a batch size of 8 per device, for 4 Tesla A100 GPUs, and the distillation takes approximately 48 hrs per language. Post-distillation, we reduce the model's vocabulary as described above. Next, we test the monolingual student models on two different downstream tasks for each language. For the fine-tuning, we add a

classification layer to the distilled model and train with an LR of $5e-5$ for 5 epochs with 500 warm up steps before a linear LR decay. We include tasks requiring both semantic understanding (sentiment analysis, news classification) as well as syntactic knowledge (PoS-tagging, NER).

An overview of the evaluated tasks and datasets per language is presented in Table 1. The last column refers to the models used for the upper-bound scores in Table 2. While for some languages we could use the standard monolingual BERT models as the upper bound (BERTje[1], AlephBERT[2], SwahBERT[3]), for other languages we had to use more advanced architectures like RoBERTa (CamemBERT[4], SloBERTa[5]), while for Hindi we used IndicBERT[6], which is an ALBERT model for 12 Indic Languages.

4 Results and Discussion

The results for all experiments are summarized in Table 2. The distilled language models (*Eliquare*) almost consistently outperform mBERT, even when mBERT has 3 times as many parameters. A more fair comparison is made with distilmBERT (Sanh et al., 2019), which has fewer parameters than mBERT and a similar model structure to the *Eliquare* models, but uses the entire Wikipedia for all languages for the distillation process (i.e. 237 million pages). In addition, we notice that DistilmBERT is also consistently outperformed by *Eliquare*. This outcome may seem counter-intuitive and unusual at first, since a student model should in principle not be outperforming a teacher model based on evidence from a number of distillation methodologies presented over the years. It can be explained, however, by the loss setup discussed in Section 3. If mBERT consistently makes mistakes for MLM, $L_{distillation}$ and L_{mlm} continue to contradict each other. While the MLM loss will encourage the model to improve, since the distillation loss is weighted (α_1) so heavily, it ensures the model cannot be very different from the teacher. As a result, each *Eliquare* model, even though better, is only marginally improved due to the distillation loss halting the progress. This is illustrated by Figure 1, which shows the progress of the two losses for Dutch distillation. While $L_{distillation}$ converges much faster, and maintains a much lower absolute value, L_{mlm} encounters a lot of fluctuation and has an almost four times higher mean value, even though both loss functions are near identical (cross-entropy). The distillation and MLM loss plots for the other 5 languages are provided in appendix A. The results for the upper-bound models are only

	Task 1	Task 2
French		
Upper-bound $*$	0.9338	0.9818
mBERT	0.8923	**0.9795**
distilmBERT	0.8773	0.9790
Eliquare	**0.8952**	0.9792
Dutch		
Upper-bound	0.9300	0.9630
mBERT	**0.9033**	0.9623
distilmBERT	0.8812	0.9607
Eliquare	0.8970	**0.9625**
Hindi		
Upper-bound $\star *$	0.2553	0.9208
mBERT	0.4744	0.9666
distilmBERT	0.4555	0.9597
Eliquare	**0.5066**	**0.9683**
Hebrew		
Upper-bound	0.8871	0.9620
mBERT	0.8512	0.9681
distilmBERT	0.8391	0.9597
Eliquare	**0.8567**	**0.9705**
Swahili		
Upper-bound	0.9090	0.8850
mBERT	0.8689	0.8490
distilmBERT	0.8666	0.8452
Eliquare	**0.8701**	**0.8632**
Slovene		
Upper-bound $*$	0.9410	0.9902
mBERT	0.9326	0.9791
distilmBERT	0.9268	0.9790
Eliquare	**0.9365**	**0.9822**

Table 2: Experimental results (macro-F1) for multilingual BERT (*mBERT*), distilled mBert (*distil*) and our language-specific distillation approach (*Eliquare*) for various end-tasks for 2 high-resourced (French, Dutch), 2 middle-resourced (Hindi, Hebrew) and 2 low-resourced languages (Swahili, Slovene). $\star$ Refers to non-monolingual models trained with additional similar languages, while $*$ refers to more advanced architectures (like RoBERTa) expected to perform better than BERT-based models.

[1]https://huggingface.co/GroNLP/bert-base-dutch-cased
[2]https://huggingface.co/onlplab/alephbert-base
[3]https://github.com/gatimartin/SwahBERT
[4]https://huggingface.co/camembert-base
[5]https://huggingface.co/EMBEDDIA/sloberta
[6]https://huggingface.co/ai4bharat/indic-bert

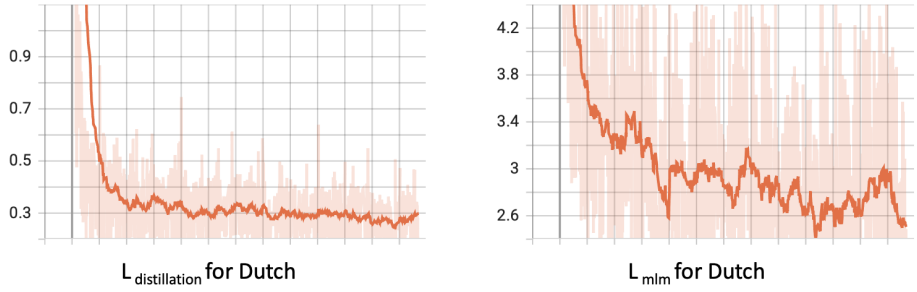

Figure 1: Distillation loss (left) and MLM loss (right) for the Dutch language distillation.

Model	# of Parameters	Inference Speed	Inference Memory
mBERT	167M	0.384s	10880 MB
distilmBERT	134M	0.165s	8798 MB
Eliquare	66M	0.066s	2944 MB

Table 3: Sustainability comparisons of *Eliquare* with standard mBERT and distilled mBERT versions. All numbers were computed on a Tesla V100 with a Batch Size of 32 and Sequence Length of 512 for inference.

provided for reference and do not serve as a fair comparison since these are specialized models for the language trained on significantly more monolingual data compared to mBERT.

While the performances of all the models maybe very close together, where the *Eliquare* set of models really shine is in a practical deployment setting. As shown in Table 3, *Eliquare* models are about 6 times faster than mBERT for inference (for a Batch Size of 32 and Sequence Length of 512 on a single Tesla V100), while being approximately 2.5 times faster than the distilled mBERT model. Moreover, they occupy around 1/3rd of the memory of mBERT with the same inference setting.

To conclude, we demonstrate that it is possible to distil better, smaller and faster monolingual models from mBERT, using a very basic distillation setup. The results show that the *Eliquare* monolingual models consistently outperform mBERT which has 3 times more parameters, and distilmBERT which has almost 2 times more parameters and uses orders of magnitude more data for the distillation process. While the *Eliquare* models may not be useful for the high-resourced languages due to the availability of large monolingual models, they present a major step towards having small monolingual models for a number of low-resourced languages.

In future research, we will investigate how to further improve the student from the teacher by diminishing the impact of the distillation loss. In addition, we will run the same set of experiments with XLM-R to investigate whether the same distillation approach can be applied to other joint multilingual models as well. Finally, we will also explore the impact of distilling multiple typologically similar languages from mBERT in parallel.

References

Amine Abdaoui, Camille Pradel, and Grégoire Sigel. 2020. Load what you need: Smaller versions of multilingual bert. In *SustaiNLP / EMNLP*.

D. Adelani, Jade Abbott, Graham Neubig, Daniel D'Souza, Julia Kreutzer, Constantine Lignos, Chester Palen-Michel, Happy Buzaaba, Shruti Rijhwani, Sebastian Ruder, Stephen Mayhew, Israel Abebe Azime, S. Muhammad, Chris C. Emezue, Joyce Nakatumba-Nabende, Perez Ogayo, Anuoluwapo Aremu, Catherine Gitau, Derguene Mbaye, J. Alabi, Seid Muhie Yimam, Tajuddeen R. Gwadabe, Ignatius Ezeani, Rubungo Andre Niyongabo, Jonathan Mukiibi, V. Otiende, Iroro Orife, Davis David, Samba Ngom, Tosin P. Adewumi, Paul Rayson, Mofetoluwa Adeyemi, Gerald Muriuki, Emmanuel Anebi, C. Chukwuneke, N. Odu, Eric Peter Wairagala, S. Oyerinde, Clemencia Siro, Tobius Saul Bateesa, Temilola Oloyede, Yvonne Wambui, Victor Akinode, Deborah Nabagereka, Maurice Katusiime, Ayodele Awokoya, Mouhamadane Mboup, D. Gebreyohannes, Henok Tilaye, Kelechi Nwaike, Degaga Wolde, Abdoulaye Faye, Blessing Sibanda, Orevaoghene Ahia, Bonaventure F. P. Dossou, Kelechi Ogueji, Thierno Ibrahima Diop, A. Diallo, Adewale Akinfaderin, T. Marengereke, and Salomey Osei. 2021. Masakhaner: Named entity recognition for african languages. *ArXiv*, abs/2103.11811.

Adam Amram, Anat Ben David, and Reut Tsarfaty. 2018. Representations and architectures in neu-

ral sentiment analysis for morphologically rich languages: A case study from Modern Hebrew. In *Proceedings of the 27th International Conference on Computational Linguistics*, pages 2242–2252, Santa Fe, New Mexico, USA. Association for Computational Linguistics.

Yelysei Bondarenko, Markus Nagel, and Tijmen Blankevoort. 2021. Understanding and overcoming the challenges of efficient transformer quantization. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 7947–7969, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Tom B. Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel M. Ziegler, Jeffrey Wu, Clemens Winter, Christopher Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. 2020. Language models are few-shot learners.

Beidi Chen, Tri Dao, Eric Winsor, Zhao Song, Atri Rudra, and Christopher Ré. 2021. Scatterbrain: Unifying sparse and low-rank attention. In *Advances in Neural Information Processing Systems (NeurIPS)*.

A. Conneau and G. Lample. 2019. Cross-lingual language model pretraining. In *Advances in Neural Information Processing Systems*, volume 32. Curran Associates, Inc.

J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, Minneapolis, Minnesota. Association for Computational Linguistics.

Geoffrey Hinton, Oriol Vinyals, and Jeff Dean. 2015. Distilling the knowledge in a neural network.

François Lagunas, Ella Charlaix, Victor Sanh, and Alexander M. Rush. 2021. Block pruning for faster transformers. *CoRR*, abs/2109.04838.

Hang Le, Loïc Vial, Jibril Frej, Vincent Segonne, Maximin Coavoux, Benjamin Lecouteux, Alexandre Allauzen, Benoît Crabbé, Laurent Besacier, and Didier Schwab. 2019. Flaubert: Unsupervised language model pre-training for french.

Y. Liu, M. Ott, N. Goyal, J. Du, M. Joshi, D. Chen, O. Levy, M. Lewis, L. Zettlemoyer, and V. Stoyanov. 2019. Roberta: A robustly optimized BERT pretraining approach. *CoRR*, abs/1907.11692.

M. Peters, M. Neumann, M. Iyyer, M. Gardner, C. Clark, K. Lee, and L. Zettlemoyer. 2018. Deep contextualized word representations. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 2227–2237, New Orleans, Louisiana. Association for Computational Linguistics.

Afshin Rahimi, Yuan Li, and Trevor Cohn. 2019. Massively multilingual transfer for NER. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 151–164, Florence, Italy. Association for Computational Linguistics.

Machel Reid, Edison Marrese-Taylor, and Yutaka Matsuo. 2021. Subformer: Exploring Weight Sharing for Parameter Efficiency in Generative Transformers. In *Findings of the Association for Computational Linguistics: EMNLP 2021*, Punta Cana, Dominican Republic. Association for Computational Linguistics.

Victor Sanh, Lysandre Debut, Julien Chaumond, and Thomas Wolf. 2019. Distilbert, a distilled version of bert: smaller, faster, cheaper and lighter. In *NeurIPS EMC^{2}Workshop*.

Shaden Smith, Mostofa Patwary, Brandon Norick, Patrick LeGresley, Samyam Rajbhandari, Jared Casper, Zhun Liu, Shrimai Prabhumoye, George Zerveas, Vijay Korthikanti, Elton Zhang, Rewon Child, Reza Yazdani Aminabadi, Julie Bernauer, Xia Song, Mohammad Shoeybi, Yuxiong He, Michael Houston, Saurabh Tiwary, and Bryan Catanzaro. 2022. Using deepspeed and megatron to train megatron-turing nlg 530b, a large-scale generative language model.

Siqi Sun, Yu Cheng, Zhe Gan, and Jingjing Liu. 2019. Patient knowledge distillation for bert model compression. *arXiv preprint arXiv:1908.09355*.

Raphael Tang, Yao Lu, Linqing Liu, Lili Mou, Olga Vechtomova, and Jimmy Lin. 2019. Distilling task-specific knowledge from bert into simple neural networks.

Benjamin van der Burgh and Suzan Verberne. 2019. The merits of universal language model fine-tuning for small datasets - a case with dutch book reviews. *CoRR*, abs/1910.00896.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. volume abs/1706.03762.

Shijie Wu and Mark Dredze. 2020. Are all languages created equal in multilingual bert? *CoRR*, abs/2005.09093.

A Loss Variation

In this appendix, we further illustrate with Figure 2 the differences in convergence for the MLM loss

and the distillation loss for the other 5 languages, and consistently observe the findings presented in Section 4. The MLM loss continues to have a much larger mean value than the distillation loss, while also having issues converging with various spikes in the loss, while the training for the distillation loss is much more stable. This is line with our hypothesis that the large α_1 values prioritize the distillation over the language modelling objective, thus not allowing the model to further improve from the teacher.

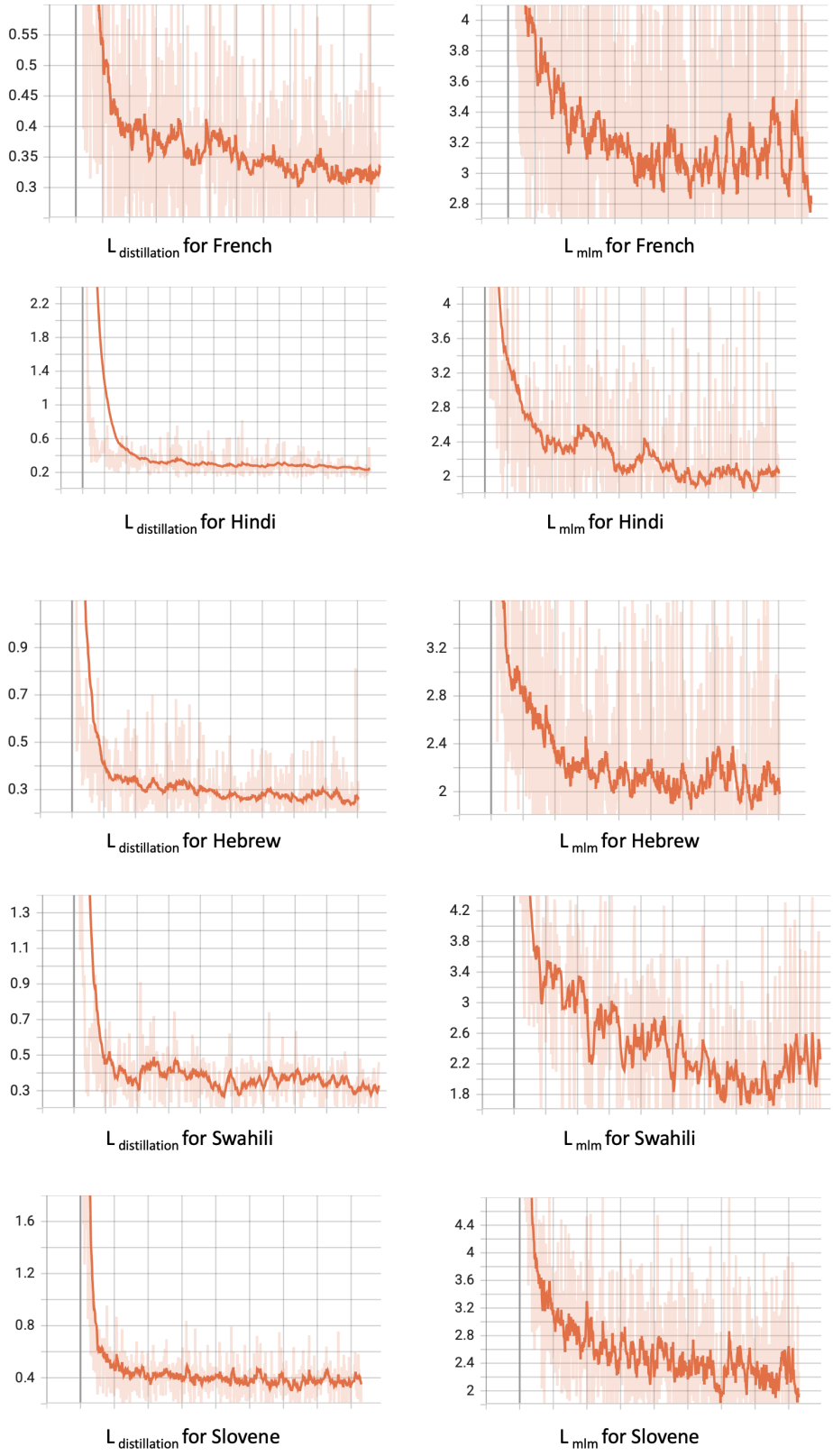

Figure 2: Distillation loss (left) and MLM loss (right) for the other 5 languages: French, Hindi, Hebrew, Swahili and Slovene.

Zero-shot Disfluency Detection for Indian Languages

Rohit Kundu, Preethi Jyothi and **Pushpak Bhattacharyya**
Department of Computer Science and Engineering
Indian Institute of Technology Bombay
{rkundu, pjyothi, pb}@cse.iitb.ac.in

Abstract

Disfluencies that appear in the transcriptions from automatic speech recognition systems tend to impair the performance of downstream NLP tasks. Disfluency correction models can help alleviate this problem. However, the unavailability of labeled data in low-resource languages impairs progress. We propose using a pretrained multilingual model, finetuned only on English disfluencies, for zero-shot disfluency detection in Indian languages. We present a detailed pipeline to synthetically generate disfluent text and create evaluation datasets for four Indian languages: Bengali, Hindi, Malayalam, and Marathi. Even in the zero-shot setting, we obtain F1 scores of 75 and higher on five disfluency types across all four languages. We also show the utility of synthetically generated disfluencies by evaluating on real disfluent text in Bengali, Hindi, and Marathi. Finetuning the multilingual model on additional synthetic Hindi disfluent text nearly doubles the number of exact matches and yields a 20-point boost in F1 scores when evaluated on real Hindi disfluent text, compared to training with only English disfluent text.

1 Introduction

Disfluencies (e.g., *filled pauses*, *repetitions*, *discourse markers*) are artefacts that are inherent to spontaneous or conversational speech. Disfluencies typically obey the following surface structure comprising: a *reparandum*, an *interruption point* (+) that marks the end of the reparandum, an *interregnum*, and finally the *repair* (Shriberg, 1994). The *reparandum* consists of one or more words that are not intended by the speaker and will be replaced or ignored. The *interregnum* consists of an editing term indicating that the *reparandum* will be edited, or it may be empty, or it can contain *fillers*, *discourse markers*, etc. The *repair* section reflects the fluent part of the utterance. Words from the *reparandum* are repeated or corrected in the *repair* section, or a new chain of thought is started in case of a *false start*.

Consider the following example that illustrates two disfluency types:

{well} [i think + {you know} i think] the idea will work

The words highlighted in red and green refer to *discourse marker* and *repetition* disfluency types, respectively. The part in blue is the fluent version of the original sentence. The example also follows the standard annotation scheme:

$$[reparandum + \{interregnum\}\ repair]$$

Disfluencies in automatically transcribed text pose a major challenge for downstream NLP tasks such as machine translation, summarization, etc. (Rao et al., 2007; Wang et al., 2010). Disfluency detection/correction is often used as a preprocessing step for NLP, where the goal is to identify/remove the disfluent words (Shriberg et al., 1992). While disfluency correction has been extensively studied for English (Honal and Schultz, 2003; Zayats et al., 2014), it has received far less attention in other languages. This is largely due to the lack of labeled data for other languages.

In this work, our main objective is to build disfluency detection models for four Indian languages — Bengali, Hindi, Malayalam, and Marathi — in the zero-shot setting with no access to labeled disfluent data in these languages. To the best of our knowledge, this is the very first study of disfluency detection across multiple Indian languages and also the very first to investigate the ability of large pretrained models to do zero-shot disfluency detection.

We specify a rule-based procedure to generate disfluencies starting from fluent sentences. It is worth noting that the synthetically generated disfluent data might not completely reflect real world disfluencies. Nevertheless, we find the synthetic data to be useful in improving disfluency detection

Proceedings of the 29th International Conference on Computational Linguistics, pages 4442–4454
October 12–17, 2022.

for low-resource Indian languages. Also, we can create near-real disfluent data by manually editing the synthetic data, which will take significantly less time than annotating from scratch. Our rule-based pipeline is targeted at Indian languages, and the same set of rules is applied to sentences in four Indian languages — Bengali, Hindi, Malayalam, and Marathi. From within these synthetic datasets, native speakers of the respective languages manually identified disfluent sentences that seemed natural. This resulted in manually-verified evaluation datasets for all the four languages. We also constructed evaluation datasets for Bengali, Hindi, and Marathi with real[1] disfluent sentences, transcribed and extracted from conversational speech in interviews. Using these datasets as evaluation benchmarks and inspired by prior work on cross-lingual zero-shot transfer using large pretrained multilingual models (Pires et al., 2019; Hu et al., 2020; Khanuja et al., 2021), we investigated the effectiveness of a large pretrained multilingual model MuRIL (Khanuja et al., 2021) on the task of zero-shot disfluency detection. MuRIL is a multilingual transformer-based model that is pretrained on large amounts of text in a number of different Indian languages. We finetuned MuRIL using labeled disfluent sentences in English (Godfrey et al., 1992) (and synthetic disfluent text) and evaluated disfluency correction for all four Indian languages in the zero-shot setting.[2]

Four MuRIL-based disfluency detection models were trained, viz. those using (1) only real English disfluent data, (2) both real English and synthetically generated Bengali disfluent data, (3) both real English and synthetically generated Hindi disfluent data, and (4) both real English and synthetically generated Marathi disfluent data. Model (2) significantly improves disfluency detection on the real Bengali disfluent data compared to model (1). We also observe similar results for the other two languages. This validates our claim that our synthetically generated data is effective in capturing (some subset of) the kinds of disfluencies that are encountered in real conversational data.

The main ideas in this work can be summarized as follows:

- We outline a common rule-based procedure that allows us to synthesize disfluencies for four Indian languages: Bengali, Hindi, Malayalam, and Marathi.

- We construct manually-verified evaluation datasets for all four Indian languages, starting from synthetically generated data. The Bengali, Hindi, Malayalam, and Marathi test sets contain 500, 575, 575, and 420 sentences, respectively.

- We also annotate real labeled disfluent datasets in Bengali, Hindi, and Marathi, containing 300, 150, and 250 sentences, respectively. These sentences were transcribed and extracted from real conversational speech. We note that this annotation process is substantially more tedious than identifying natural disfluencies starting from our synthetic data.

- We finetune a pretrained multilingual model, MuRIL, on labeled disfluent data in English and show its effectiveness at zero-shot disfluency detection for all four Indian language datasets.

- We show the utility of our synthetic disfluency generation pipeline by comparing performance of a model finetuned only on real English disfluent data versus a model finetuned on both real English and synthetically generated disfluent data of one Indian language.

- We present a detailed breakdown of performance across various disfluency types, show qualitative analyses of our model predictions and highlight some interesting aspects related to disfluencies in Indian languages (e.g., *reduplication*).

2 Related Work

There are three main categories of approaches for disfluency detection. They are based on (1) sequence tagging, (2) parsing, and (3) a noisy channel model (Kundu et al., 2022).

Sequence tagging based approaches use classification techniques to label individual words (Liu et al., 2006; Ostendorf and Hahn, 2013; Zayats et al., 2014; Ferguson et al., 2015; Hough and Schlangen, 2015; Zayats et al., 2016; Wang et al.,

[1]We use the term *real* throughout the paper to contrast the manually-edited synthetic datasets with the (real) datasets containing disfluencies annotated from conversations.

[2]Our code and datasets can be found at https://github.com/RKKUNDU/zero-shot-disfluency-detection.

2018). Parsing-based approaches detect disfluencies along with identifying the syntactic structure of the sentence (Rasooli and Tetreault, 2013; Honnibal and Johnson, 2014; Wu et al., 2015; Yoshikawa et al., 2016; Jamshid Lou and Johnson, 2020). The main idea behind a noisy channel model of disfluency is that we assume there is a fluent source sentence X to which some noise has been added, resulting in a disfluent sentence Y. The goal is to find the most likely fluent sentence given Y (Johnson and Charniak, 2004; Zwarts and Johnson, 2011; Jamshid Lou and Johnson, 2017).

Prior works (Hu et al., 2020; Khanuja et al., 2021) have used pretrained multilingual models for many zero-shot NLP tasks such as Named Entity Recognition (NER), Part of Speech (POS) tagging, Question Answering (QA), etc. However, this is the first work to attempt disfluency detection in a zero-shot setting and the very first work to study disfluency detection for multiple Indian languages.

Our work on synthetic disfluency data generation has parallels to the recent work of Passali et al. (2022) where they focus on an artificial disfluency generation algorithm. They focus broadly on *Repetitions*, *Replacements*, and *Restarts* and only focus on English. Saini et al. (2020) is another prior work that has looked into inducing disfluencies in English fluent text. Our disfluencies are much more fine-grained in construction (e.g., pronoun corrections, missing syllables, etc.) compared to prior work and apply to Indian languages.

3 Generating Synthetic Disfluencies

We focus on four major disfluency types as listed in Honal and Schultz (2003), i.e., *Fillers*, *Repetitions*[3], *Corrections*, and *False Starts*. We specify a total of nine rules across the four disfluency types to introduce disfluencies in fluent sentences. We show examples of Bengali disfluent sentences in Appendix A for all the disfluency types. Apart from what we describe in this section, there are some more fine-grained details governing how and where various disfluencies are introduced in a fluent sentence; these details are specified in our released codebase.

[3]In the literature, *Repetitions* and *Corrections* disfluency categories are considered as a single category

3.1 Fillers

We loosely use the term *Fillers* to denote *editing terms*, *discourse markers*, *filled pauses* and *interjections*. *Editing terms* are used to explicitly indicate that the previously uttered word(s) were not intended. *Discourse markers* help in beginning or keeping a turn (e.g., well) or merely serve as a form of acknowledgment (e.g., yeah). *Filled pauses* are non-lexicalized sounds without any semantic content. *Interjections* are defined as non-lexicalized sounds indicating affirmation or negation.

We simply introduce frequent *filler* phrases at randomly chosen positions. We choose frequent *filler* phrases after carefully observing conversations. We assume that there will be at most 3 *fillers* in a sentence and uniformly choose a number between 1 and 3. Thereafter, with uniform probability, we pick the location in the sentence at which the next *filler* will be inserted and also choose the *filler* phrase to be inserted with uniform probability from a pool of *filler* phrases.[4]

Speakers might tend to use *fillers* before long words. For words with 12 or more characters, we first choose a *filler* phrase with uniform probability and then place it before the long word.

3.2 Repetitions

Repetition is defined as the phenomenon of speakers repeating a word or phrase.

Word Repetition. For this rule, we pick a word uniformly at random and repeat it.

Phrase Repetition. In this rule, we repeat a phrase[5] containing 2 to 5 words. We first randomly pick a length from $[2, 3, 4, 5]$ using a weighted distribution of $[0.4, 0.3, 0.2, 0.1]$.[6] Then, we pick a phrase of the chosen length uniformly at random and repeat it.

Pronoun Repetition. We find pronouns to be commonly repeated in Indian languages. First, we accumulate a list of pronouns for each language. If any word in the fluent sentence appears in the pronoun list, then we repeat the pronoun with a predetermined probability.[7]

[4]These *filler* phrases are separately listed for all four Indian languages after consulting native speakers.

[5]Here, we mean an n-gram of consecutive words regardless of their real phrasal structure.

[6]This distribution was chosen only to signify that phrases of shorter length are more frequent than phrases of longer length.

[7]More details about the probability with which a pronoun is chosen to be repeated is specified in our code.

3.3 Corrections

Corrections involve substitutions, deletions, or insertions of words from the *reparandum* section. *Corrections* may include the *interregnum*.

Partial Word. For this rule, we introduce *partial words* before long words with 12 or more characters. Firstly, we find the orthographic syllables[8] of a long word using the Indic NLP Library[9] (Kunchukuttan, 2020). Thereafter, we create the *partial word* by joining the first n syllables where n comes from a weighted distribution in which probability $P(n)$ is proportional to $\frac{1}{n}$.

Missing Syllables. For this rule, preceding a long word of 12 or more characters, we insert the same word but with one or more syllables missing. We first find the orthographic syllables of the long word. Then, we remove n contiguous syllables from the word (where n is sampled from a weighted distribution similar to what we used for *phrase repetition)* and add this reduced form of the word prior to the original long word.

Pronoun Correction. In this rule, a pronoun gets explicitly corrected. From the pronoun lists mentioned in Section 3.2, we create groups of similar types of pronouns (e.g., all first person pronouns are in one group). For each pronoun in the fluent sentence, we find its group and pick a different pronoun from the group to serve as its correction. We also (optionally) insert a frequent filler phrase before using the correct pronoun.

Synonym Correction. In this rule, we introduce a synonym of the word before the actual word, obtained using IndoWordNet[10] (Bhattacharyya, 2010).

3.4 False Start

For the *False Start* disfluency, a sentence is aborted before it is completed, and a new idea or line of thought is introduced. To create *false starts*, we first randomly pick two different fluent sentences. Then, we split the first fluent sentence from a random position and we concatenate the first part of the split with the second fluent sentence.

[8] Orthographic syllable is a sequence of one or more consonants followed by a vowel.

[9] https://github.com/anoopkunchukuttan/indic_nlp_library

[10] https://www.cfilt.iitb.ac.in/indowordnet/

4 Dataset Details

4.1 Disfluency Datasets for Indian Languages

Real Disfluent Data. We create *real disfluent datasets* in Bengali, Hindi, and Marathi by transcribing and annotating real disfluencies from conversations in the respective languages. For this purpose, we used publicly available *Interviews* in Bengali[11], Hindi[12], and Marathi[13] from YouTube[14]. From these videos, we constructed three datasets containing 300, 150, and 250 disfluent and fluent parallel sentences in Bengali, Hindi, and Marathi, respectively.

Synthetic Disfluent Data. We also induce disfluencies in fluent text using our rule-based algorithm and create evaluation datasets for disfluency detection in Bengali, Hindi, Malayalam and Marathi. We start with fluent monolingual text from the PMIndia corpus[15] (Haddow and Kirefu, 2020). We synthesize disfluent sentences using the rules outlined in Section 3. We ask language specialists in each of the four languages to manually pick sentences from the synthetic dataset that appear like natural disfluencies (and edit the disfluent sentences if needed). We picked utterances such that there is uniform coverage across disfluency types and there is no label imbalance. We used IndicNLP (Kunchukuttan, 2020) for normalization and tokenization, and we removed all punctuation marks. Table 1 shows detailed disfluency type counts for all four datasets. The test sets for Bengali, Hindi, Malayalam, and Marathi contain 500, 575, 575, and 420 sentences, respectively.

Each of the test sets is grouped into five categories: *fillers*, *repetitions*, *corrections*, *false starts* and fluent sentences. Fluent sentences are included as a control set to check whether the model is incorrectly detecting disfluencies in fluent sentences. We also include fluent sentences with *reduplications* which are a special category in Indian languages as mentioned below.

Reduplication. *Reduplication* is the act of repeating all or part of a word for emphasis or to

[11] https://www.youtube.com/c/WBCSMadeEasyTM

[12] https://www.youtube.com/NeeleshMisraChannel

[13] https://www.youtube.com/c/GopalDarji

[14] We obtained explicit permission from the Bengali and Marathi content creators to use the data for research. The Hindi content creator allows *fair use* of specific videos for research.

[15] https://data.statmt.org/pmindia/

Type	Bn	Hi	Ml	Mr
Filler (3.1)	50	100	100	70
Word Repetition (3.2)	42	50	50	35
Phrase Repetition (3.2)	42	50	50	35
Pronoun Repetition (3.2)	41	50	50	35
Partial Word (3.3)	66	50	50	35
Missing Syllables (3.3)	34	50	50	35
Pronoun Correction (3.3)	66	50	50	35
Synonym Correction (3.3)	34	50	50	35
False Start (3.4)	50	50	50	35
Fluent Sentences with Redpl	25	25	25	20
Normal Fluent Sentences	50	50	50	50
Total	500	575	575	420

Table 1: Synthetic Dataset Statistics: Number of sentences of each disfluency type. Redpl: Reduplication, Bn: Bengali, Hi: Hindi, Ml: Malayalam, Mr: Marathi.

convey a meaning. It is widely used in Indian languages; a few examples of reduplication in Hindi are shown in Table 2 (Montaut, 2009). In the context of disfluencies, we note that *reduplications* could be mistaken for a *repetition* disfluency type. *Reduplications* are intentional repetitions which are grammatically correct and should not be flagged as disfluencies. To check for this, we include fluent sentences with *reduplication* in our test set.

4.2 English Disfluency Data

We also present results for English to check how well MuRIL performs when compared with previously published results. Switchboard[16] (Godfrey et al., 1992) in English is the most commonly used dataset for disfluency detection. Following the experimental settings in Wang et al. (2021), we split the Switchboard corpus such that the dev set consists of all sw_04[5-9]*.utt files, the test set consists of all sw_04[0-1]*.utt files, and the training set consists of all the remaining files. We do not include sentences without disfluencies in the training data, but do so in the dev, test set. Following Honnibal and Johnson (2014), we lowercase the text and remove all punctuation marks.

5 Experimental Setup

In this work, we use MuRIL (Khanuja et al., 2021) which is a BERT model (Devlin et al., 2019) pretrained on 16 Indian languages (including the four we consider) and English. MuRIL is pretrained using two language modeling objectives: Masked Language Modeling and Translation Language Modeling.

[16] https://catalog.ldc.upenn.edu/LDC97S62

[17] Orange color denotes *reduplication*

Evaluation Metrics. We test the model on Bengali, Hindi, Malayalam, Marathi and English disfluency detection tasks. Similar to prior work on detecting English disfluencies (Wang et al., 2021), we compute precision, recall, and F1 scores using word-level labels. We also use a more ambitious metric, the *exact match percentage*, where the predicted fluent sentence is compared to the reference fluent sentence and checked for an exact match. We also show BLEU scores between the fluent text predictions and the reference fluent sentences, which are calculated using *sacreBLEU* (Post, 2018).

5.1 Using only English Disfluency Data

We finetune the pretrained MuRIL checkpoint on the English disfluency detection task where the goal is to correctly label each of the tokens as *fluent* or *disfluent*. We use the *muril-base-cased* checkpoint (having 236M parameters) from *HuggingFace*[18] for all our experiments. For each of the subword tokens identified by the MuRIL tokenizer, the model predicts its label as being 0 (*fluent*) or 1 (*disfluent*).

For disfluency correction, once we have disfluency labels for each subword, we use majority voting to determine whether a word is omitted or not. For a word, if the number of its subwords tagged as disfluent is greater than the number of subwords tagged as fluent, the word is deleted; else, it is retained.

5.2 Using Synthetically Generated Indian Language Data along with English Disfluency Data

We would like to check whether our synthetically generated data in one Indian language (say, Bengali) helps improve performance on real disfluencies in Bengali, Hindi, and Marathi languages.

We generate disfluent sentences having n disfluencies where $n \in \{1, 2, 3, 4, 5\}$. Ignoring false starts, we pick one of the eight disfluency types (listed in Table 1) at random and inject disfluencies in fluent Bengali sentences from the PMIndia corpus. We generated 42500 disfluent Bengali sentences in this way, which is roughly half the number of sentences in the Switchboard corpus.[19] Sim-

[18] https://huggingface.co/google/muril-base-cased

[19] We want to augment the Switchboard data with the synthetic Bengali data, but do not want the synthetic data to dominate the corpus.

Sentence	Transliteration	Gloss	Translation	Comment
तुम कहा कहा[17] गए	tuma kahaa kahaa gae	you where where went	where did you go	Reduplication of interrogative pronoun. Here the questioner expects a list of places in response.
खाते खाते मत बोलो	khaate khaate mata bolo	eating eating do_not speak	do not speak while eating	Reduplication of verb
यह लो तुम्हारी चाय. गरम गरम है, पियो	yaha lo tumhaarii caaya. garama garama hai, piyo	this take your tea. hot hot is drink	Take your tea. It is nicely hot, drink it	Reduplication of adjective
बच्चो को एक एक टॉफ़ी दो	bacco ko eka eka taffii do	children to one one toffee give	give a toffee to each child	Reduplication of number.

Table 2: Examples of Reduplication in Hindi. Gloss: word-to-word English translation. "_" in the gloss suggests *fertility* which refers to one word mapping to multiple words in the other language.

ilarly, we construct synthetic data for the other two languages as well.

Next, we finetune the pretrained MuRIL checkpoint on the combined synthetic Bengali data and Switchboard data. The other experimental details are the same as described in Section 5.1. We evaluate this model on real disfluency detection data in Bengali, Hindi, and Marathi. We hypothesize that the performance of this model on the real Bengali disfluency detection dataset will be better than that of the model finetuned only on English data. This would indicate that our synthetically generated data contains disfluencies that mimic the ones seen in real speech.

5.3 Using Only Synthetically Generated Indian Language Data

We also finetune the pretrained MuRIL checkpoint only on the synthetic Bengali/Hindi/Marathi data. We apply the same synthetic data and experimental setup as discussed in Section 5.2.

6 Results & Analysis

This section presents the evaluation results and analyses the quantitative and qualitative performance of our models.

6.1 Performance on Real Disfluent Data

Table 3 shows a comparison of our model finetuned on only English data, and models finetuned on synthetic Bengali/Hindi/Marathi disfluent data (optionally) along with real English disfluent data.

We can see that *MuRIL - En & Syn Bn* (model finetuned using both synthetically generated Bengali data and the Switchboard corpus) outperforms *MuRIL - En* (finetuned only using the Switchboard corpus) by a significant margin of 19% in terms of exact matches, when evaluated on real Bengali disfluencies. Also, *MuRIL - En & Syn Bn* has high precision which leads to an increase of 4.67 F1 scores. Similarly, *MuRIL - En & Syn Hi* model outperforms *MuRIL - En* by a large margin of 19.92 F1 scores, when evaluated on real Hindi disfluencies. Both *MuRIL - En & Syn Bn* and *MuRIL - En & Syn Mr* also outperform *MuRIL - En* by 15.51 and 18.14 F1 scores, respectively. We observe similar trends on the real Marathi evaluation set as well.

All the models that were finetuned with additional synthetic data (irrespective of the language) nearly double the number of exact matches when evaluated on real Hindi/Marathi disfluencies, compared to the model trained with only English disfluent data. Our models trained only on synthetic disfluent data outperform the *MuRIL - En* model. We observe 1.17, 12.33, 17.48 F1 scores improvement over *MuRIL - En* model, when evaluated on real Bengali, Hindi and Marathi evaluation sets, respectively.

These results suggest that using synthetically generated disfluent sentences does enable transfer to real disfluent data and helps validate our synthetic data generation pipeline.

6.2 Disfluency Detection in Indian Languages

Table 4 presents a detailed account of the performance of MuRIL on the four manually-edited synthetic disfluency evaluation sets in Bengali, Hindi, Malayalam, and Marathi. It also provides a breakdown of performance across disfluency types. We report the exact match percentages, the BLEU scores and the F1 scores.

The overall F1 scores for Bengali, Hindi, Malayalam and Marathi are 73.14, 63.82, 67.12 and

Language	Model	Precision	Recall	F1 score	Exact Match %	BLEU
Bengali	MuRIL - En	81.13	65.60	72.54	28.33	80.2
	MuRIL - Syn Bn	91.79	61.58	73.71	43.00	81.2
	MuRIL - En & Syn Bn	**92.90**	**66.06**	**77.21**	**47.33**	**82.6**
	MuRIL - En & Syn Hi	89.90	51.03	65.11	35.67	77.4
	MuRIL - En & Syn Mr	92.72	51.15	65.93	35.33	76.7
Hindi	MuRIL - En	67.81	62.20	64.89	36.67	85.5
	MuRIL - Syn Hi	83.18	72.05	77.22	54.00	89.5
	MuRIL - En & Syn Bn	82.57	78.35	80.40	64.00	91.3
	MuRIL - En & Syn Hi	84.98	**84.65**	**84.81**	**66.00**	**93.7**
	MuRIL - En & Syn Mr	**86.38**	79.92	83.03	**66.00**	91.7
Marathi	MuRIL - En	57.78	54.93	56.32	26.40	83.8
	MuRIL - Syn Mr	**92.25**	61.50	73.80	55.60	88.2
	MuRIL - En & Syn Bn	82.77	68.78	75.13	56.80	88.9
	MuRIL - En & Syn Hi	83.14	68.31	75.00	55.60	89.5
	MuRIL - En & Syn Mr	87.54	**69.25**	**77.33**	**60.80**	**90.0**

Table 3: Performance on real Bengali, Hindi, and Marathi disfluent data. *MuRIL - En*: Finetuning MuRIL only on Switchboard data, *MuRIL - En & Syn X* (where $X \in \{Bn, Hi, Mr\}$): Finetuning MuRIL on Switchboard data and synthetic disfluency detection data in language X [Bn: Bengali, Hi: Hindi, Mr: Marathi] , *MuRIL - Syn X* (where $X \in \{Bn, Hi, Mr\}$): Finetuning MuRIL only on synthetic disfluent data in language X . We note that the BLEU scores between the original disfluent text and the fluent reference text for Bengali, Hindi, and Marathi are 62.0, 71.9 and 73.2, respectively.

70.22, respectively. Interestingly, the model is able to do a reasonable job of disfluency detection even in the zero-shot setting with no access to labeled disfluent data in the target languages. The BLEU scores between the original disfluent text and the fluent reference for Bengali, Hindi, Malayalam and Marathi are 83.2, 83.4, 78.3, 81.1, respectively. Comparing these scores to the BLEU scores obtained using the finetuned MuRIL (92.5, 90.9, 88.4 and 91.3) clearly shows that the model is effective in removing disfluencies.

6.3 Performance across Disfluency Types

Table 4 shows that our model is doing exceptionally well at detecting *Repetitions* in all the languages and our model shows the best performance in detecting *Phrase Repetitions*. In 64% of the Bengali sentences, our model did not tag a *reduplication* as disfluency, which is correct. This suggests that the model is learning the difference between *word repetition* and *reduplication*. Also, most of the fluent sentences are kept unchanged.

Even without any explicit supervision of the filler words specific to each language, we find that our model is sometimes able to accurately detect the *fillers* based on the context in which they appear. The high F1 score in Marathi for *fillers* (compared to the other three languages) can be attributed to the fact that the *fillers* exhibited a positional bias and mostly appeared at the start of the Marathi test sentences.

Our model performs fairly on detecting *partial words*. In comparison, the model does very well on detecting *missing syllables* and achieves more than 75 F1 scores in all four languages. Despite the complexity of the *pronoun correction* task, which could also involve optional editing terms, our model performs admirably and gets F1 scores of greater than 75 across all languages.

Detecting *synonym correction* correctly is a complex task and our model does not perform too well on this disfluency type. Our model yields the lowest F1 scores across all disfluency types on *false starts*. This is not very surprising because false starts are the hardest of disfluency types to detect (Shriberg, 1994). Sometimes there are ambiguities even in the gold standard utterances containing *false starts*. (Example 7 in Table 8 shows such an ambiguity.) Another reason could be that the Switchboard dataset does not contain many *false start* disfluencies.

6.4 Performance across Languages

From Table 4, we compare the performance of our model across languages for different disfluency types.

Filler detection is done best for Marathi. We observe that the model does not perform well in detecting Hindi *fillers*. Hindi differs from the other languages in that it uses fairly long phrases as *fillers*. For example, "क्या कहते है" means "what to say". Thus, our model might find it challenging to catch these long *filler* phrases. We also observe that the model's capability to detect a *filler*

Type	Bengali			Hindi			Malayalam			Marathi		
	M	B	F1	M	B	F1	M	B	F1	M	B	F1
Filler (3.1)	34.00	86.74	53.66	23.00	82.05	28.45	19.00	78.63	37.24	61.43	90.13	77.24
Word Repetition (3.2)	78.57	98.12	90.53	66.00	96.39	79.67	82.00	98.40	92.91	82.86	97.61	92.50
Phrase Repetition (3.2)	80.95	98.49	96.42	70.00	96.91	93.84	86.00	98.46	96.75	62.86	95.52	91.57
Pronoun Repetition (3.2)	73.17	97.27	86.04	74.00	95.32	84.75	86.00	98.10	92.31	74.29	97.58	88.61
Partial Word (3.3)	45.45	90.31	66.10	38.00	91.00	54.55	52.00	88.59	70.00	42.86	89.88	60.71
Missing Syllables (3.3)	70.59	96.20	83.08	82.00	98.39	89.80	62.00	92.16	76.77	62.86	95.46	82.19
Pronoun Correction (3.3)	59.09	89.49	82.11	54.00	90.00	78.54	60.00	90.08	85.71	40.00	87.11	75.74
Synonym Correction (3.3)	29.41	86.76	48.98	18.00	77.66	54.13	18.00	73.17	45.84	14.29	67.18	41.38
False Start (3.4)	20.00	75.50	37.64	8.00	80.29	27.23	2.00	70.51	15.56	17.14	79.04	35.39
Fluent Sentences with Reduplication	64.00	96.01		64.00	97.31		60.00	94.59		35.00	89.45	
Normal Fluent Sentences	90.00	98.46		98.00	99.89		96.00	99.35		94.00	99.45	
Overall	57.60	92.52	73.14	50.96	90.87	63.82	53.22	88.39	67.12	56.19	91.31	70.22

Table 4: Performance on manually-edited synthetic Bengali, Hindi, Malayalam, and Marathi disfluency datasets. We use *MuRIL - En* model for the evaluation. M: Exact Match Percentage, B: BLEU Score, F1: F1 Score. We note here that the BLEU scores between the original disfluent text and the fluent reference text for Bengali, Hindi, Malayalam and Marathi are 83.2, 83.4, 78.3, 81.1, respectively.

word varies depending on the position of the *filler*. When *hesitations* like "अं", "आं" are present at the beginning of the sentence, our model always detects them, but not necessarily when they are present in the middle.

Our model performs the best on Hindi when it comes to detecting *missing syllables*. F1 scores when detecting *repetitions* and *corrections* are comparable across languages. However, we see a huge gap across languages in the case of *false starts*. For *false starts*, the difference in performance between the Bengali and Malayalam test sets is 22.08 (in terms of F1 scores). This large difference in performance could be attributed to the following reasons:

• The number of words present in *false starts* follows different distributions for Bengali and Malayalam. In Bengali, there are fewer occurrences of *false starts* containing many words, while there are more long *false start* occurrences in Malayalam.

• The model detects Bengali *conjunctions* like "কারণ" (meaning "because"), "কিন্তু" (meaning "but"), "আর" (meaning "and") etc. as disfluencies which are part of the *false start*.

• At times, the model detects part of the *false start* as being part of some other disfluency type, such as a *repetition* or a *correction*.

6.5 Ablation Study

Table 3 shows that combining synthetic Marathi disfluent data with English disfluent data increases F1 scores by 21 points on real Marathi disfluent data compared to a model trained solely on English data. Via an ablation study, we aim to check which subset of disfluency types in the synthetic dataset is most helpful. We consider seven combinations of disfluency types (including *MuRIL - En*[20] and *MuRIL - En & Syn Mr*[21]) that we think are representative. We generate the same amount of synthetic data for all the combinations using the steps discussed in Section 5.2.

We present the results in Table 5. It is encouraging to see that all the models achieve a precision of nearly 90 when trained on synthetic data whereas *Disf-onlyEng* achieves a precision of only 57.78. In comparison to *Disf-onlyEng*, *Disf-1* (synthetic data containing only *fillers*) has a lower F1 score, which can be attributed to the very low *recall* value. *Disf-1234* improves F1 score by 8.40, which suggests that *repetitions* help. It is interesting to see that *Disf-15678* improves F1 score by 20.09. This implies that *corrections* by themselves are of sig-

[20] *MuRIL - En* and *Disf-onlyEng* are the same. This model is only trained on English disfluent data

[21] *MuRIL - En & Syn Mr* and *Disf-all* are the same. This model is trained on English data, and synthetic Marathi data comprising all the disfluency types

Model Name	Precision	Recall	F1 score	Exact Match %	BLEU
Disf-onlyEng	57.78	54.93	56.32	26.40	83.8
Disf-1	90.97	30.75	45.96	23.60	79.9
Disf-1234	**93.36**	49.53	64.72	45.60	84.8
Disf-15678	89.06	66.90	76.41	56.80	89.4
Disf-137	89.78	47.42	62.06	38.80	83.6
Disf-1357	89.56	62.44	73.58	54.00	88.2
Disf-all	87.54	**69.25**	**77.33**	**60.80**	**90.0**

Table 5: Performance of models trained on English and additional synthetic Marathi disfluent data containing a subset of disfluency types (except *Disf-onlyEng* which is trained only on English data). The numbers in the model name indicate which disfluency types were present in the synthetic data during finetuning (e.g., *Disf-137* was trained on synthetic Marathi disfluent data containing only *filler*, *phrase repetition*, and *pronoun correction*). Mapping of number to disfluency types — 1: *filler*, 2: *word repetition*, 3: *phrase repetition*, 4: *pronoun repetition*, 5: *partial word*, 6: *missing syllables*, 7: *pronoun correction*, 8: *synonym correction*. We note here that *MuRIL - En* is the same as *Disf-onlyEng* and *MuRIL - En & Syn Mr* is the same as *Disf-all*.

nificant help. We also observe that only *phrase repetition* and *pronoun correction* do not help much as *Disf-137* achieves an F1 score of only 62.06. However, when we also add *partial words*, *Disf-1357* overshoots *Disf-137* by a margin of 11.52 F1.

6.6 English Disfluency Detection

In Table 6, we present the performance of our multilingual model MuRIL on the test set of the Switchboard corpus as a sanity check of our model. We achieved an F1 score of 93.62 on the sequence labeling task, whereas the state-of-the-art model in Wang et al. (2021) reported an F1 score of 91.7. (We note that Wang et al. (2021) preprocesses *"uh"*, *"um"*, *"I mean"*, etc. tokens differently; hence, our F1 scores are not directly comparable.)

Precision	Recall	F1 Score	Accuracy
95.22	92.08	93.62	98.00

Table 6: Performance on Switchboard disfluency detection test set.

7 Conclusion

We propose the use of a pretrained multilingual model MuRIL for zero-shot disfluency detection in Indian languages. We evaluate our model on Bengali, English, Hindi, Malayalam, and Marathi disfluency detection tasks. We also show that synthetically generated Bengali/Hindi/Marathi disfluency detection data using simple rules, when combined with real English disfluency data during finetuning, helps improve F1 scores on real Bengali/Hindi/Marathi disfluencies. Our overall results support the claim that it is possible to do cross-lingual transfer of disfluency detection without any labeled data in the target languages. For future work, we intend to evaluate the model on more diverse disfluencies.

Acknowledgements

The authors thank the anonymous reviewers for their constructive suggestions that helped improve the draft. This work was done as part of the Bahubhashak Pilot Project on Speech to Speech Machine Translation under the umbrella of National Language Technology Mission of Ministry of Electronics and IT, Govt. of India.

References

Pushpak Bhattacharyya. 2010. IndoWordNet. In *Proceedings of the Seventh International Conference on Language Resources and Evaluation (LREC'10)*, Valletta, Malta. European Language Resources Association (ELRA).

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, Minneapolis, Minnesota. Association for Computational Linguistics.

James Ferguson, Greg Durrett, and Dan Klein. 2015. Disfluency detection with a semi-Markov model and prosodic features. In *Proceedings of the 2015 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 257–262, Denver, Colorado. Association for Computational Linguistics.

John J. Godfrey, Edward C. Holliman, and Jane McDaniel. 1992. Switchboard: Telephone speech corpus for research and development. In *Proceedings of the 1992 IEEE International Conference on*

Acoustics, Speech and Signal Processing - Volume 1, ICASSP'92, page 517–520, USA. IEEE Computer Society.

Barry Haddow and Faheem Kirefu. 2020. PMIndia–A Collection of Parallel Corpora of Languages of India. *arXiv preprint arXiv:2001.09907*.

Matthias Honal and Tanja Schultz. 2003. Correction of disfluencies in spontaneous speech using a noisy-channel approach. In *Interspeech*. Citeseer.

Matthew Honnibal and Mark Johnson. 2014. Joint incremental disfluency detection and dependency parsing. *Transactions of the Association for Computational Linguistics*, 2:131–142.

Julian Hough and David Schlangen. 2015. Recurrent neural networks for incremental disfluency detection. In *Proc. Interspeech 2015*, pages 849–853.

Junjie Hu, Sebastian Ruder, Aditya Siddhant, Graham Neubig, Orhan Firat, and Melvin Johnson. 2020. Xtreme: A massively multilingual multi-task benchmark for evaluating cross-lingual generalisation. In *International Conference on Machine Learning*, pages 4411–4421. PMLR.

Paria Jamshid Lou and Mark Johnson. 2017. Disfluency detection using a noisy channel model and a deep neural language model. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 547–553, Vancouver, Canada. Association for Computational Linguistics.

Paria Jamshid Lou and Mark Johnson. 2020. Improving disfluency detection by self-training a self-attentive model. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 3754–3763, Online. Association for Computational Linguistics.

Mark Johnson and Eugene Charniak. 2004. A TAG-based noisy-channel model of speech repairs. In *Proceedings of the 42nd Annual Meeting of the Association for Computational Linguistics (ACL-04)*, pages 33–39, Barcelona, Spain.

Simran Khanuja, Diksha Bansal, Sarvesh Mehtani, Savya Khosla, Atreyee Dey, Balaji Gopalan, Dilip Kumar Margam, Pooja Aggarwal, Rajiv Teja Nagipogu, Shachi Dave, et al. 2021. MuRIL: Multilingual representations for indian languages. *arXiv preprint arXiv:2103.10730*.

Anoop Kunchukuttan. 2020. The IndicNLP Library. `https://github.com/anoopkunchukuttan/indic_nlp_library/blob/master/docs/indicnlp.pdf`.

Rohit Kundu, Preethi Jyothi, and Pushpak Bhattacharyya. 2022. Survey: Exploring disfluencies for speech-to-speech machine translation.

Yang Liu, Elizabeth Shriberg, Andreas Stolcke, Dustin Hillard, Mari Ostendorf, and Mary Harper. 2006. Enriching speech recognition with automatic detection of sentence boundaries and disfluencies. *IEEE Transactions on audio, speech, and language processing*, 14(5):1526–1540.

Annie Montaut. 2009. Reduplication and echo words in Hindi/Urdu. *Annual review of South Asian languages and linguistics*, pages 21–91.

Mari Ostendorf and Sangyun Hahn. 2013. A sequential repetition model for improved disfluency detection. In *Proc. Interspeech 2013*, pages 2624–2628.

Tatiana Passali, Thanassis Mavropoulos, Grigorios Tsoumakas, Georgios Meditskos, and Stefanos Vrochidis. 2022. Lard: Large-scale artificial disfluency generation. *arXiv preprint arXiv:2201.05041*.

Telmo Pires, Eva Schlinger, and Dan Garrette. 2019. How multilingual is multilingual BERT? In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 4996–5001, Florence, Italy. Association for Computational Linguistics.

Matt Post. 2018. A call for clarity in reporting BLEU scores. In *Proceedings of the Third Conference on Machine Translation: Research Papers*, pages 186–191, Brussels, Belgium. Association for Computational Linguistics.

Sharath Rao, Ian Lane, and Tanja Schultz. 2007. Improving spoken language translation by automatic disfluency removal: evidence from conversational speech transcripts. In *Proceedings of Machine Translation Summit XI: Papers*, Copenhagen, Denmark.

Mohammad Sadegh Rasooli and Joel Tetreault. 2013. Joint parsing and disfluency detection in linear time. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*, pages 124–129, Seattle, Washington, USA. Association for Computational Linguistics.

Nikhil Saini, Jyotsana Khatri, Preethi Jyothi, and Pushpak Bhattacharyya. 2020. Generating fluent translations from disfluent text without access to fluent references: IIT Bombay@IWSLT2020. In *Proceedings of the 17th International Conference on Spoken Language Translation*, pages 178–186, Online. Association for Computational Linguistics.

Elizabeth Shriberg, John Bear, and John Dowding. 1992. Automatic detection and correction of repairs in human-computer dialog. In *Speech and Natural Language: Proceedings of a Workshop Held at Harriman, New York, February 23-26, 1992*.

Elizabeth Ellen Shriberg. 1994. *Preliminaries to a theory of speech disfluencies*. Ph.D. thesis, Citeseer.

Feng Wang, Wei Chen, Zhen Yang, Qianqian Dong, Shuang Xu, and Bo Xu. 2018. Semi-supervised disfluency detection. In *Proceedings of the 27th International Conference on Computational Linguistics*, pages 3529–3538, Santa Fe, New Mexico, USA. Association for Computational Linguistics.

Shaolei Wang, Zhongyuan Wang, Wanxiang Che, Sendong Zhao, and Ting Liu. 2021. Combining self-supervised learning and active learning for disfluency detection. *ACM Trans. Asian Low-Resour. Lang. Inf. Process.*, 21(3).

Wen Wang, Gokhan Tur, Jing Zheng, and Necip Fazil Ayan. 2010. Automatic disfluency removal for improving spoken language translation. In *2010 IEEE International Conference on Acoustics, Speech and Signal Processing*, pages 5214–5217. IEEE.

Shuangzhi Wu, Dongdong Zhang, Ming Zhou, and Tiejun Zhao. 2015. Efficient disfluency detection with transition-based parsing. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 495–503, Beijing, China. Association for Computational Linguistics.

Masashi Yoshikawa, Hiroyuki Shindo, and Yuji Matsumoto. 2016. Joint transition-based dependency parsing and disfluency detection for automatic speech recognition texts. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 1036–1041, Austin, Texas. Association for Computational Linguistics.

Vicky Zayats, Mari Ostendorf, and Hannaneh Hajishirzi. 2016. Disfluency Detection Using a Bidirectional LSTM. In *Proc. Interspeech 2016*, pages 2523–2527.

Victoria Zayats, Mari Ostendorf, and Hannaneh Hajishirzi. 2014. Multi-domain disfluency and repair detection. In *Fifteenth Annual Conference of the International Speech Communication Association*.

Simon Zwarts and Mark Johnson. 2011. The impact of language models and loss functions on repair disfluency detection. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, pages 703–711, Portland, Oregon, USA. Association for Computational Linguistics.

A Examples: Synthetic Disfluency Generation Rules

Table 7 presents a brief overview and example of each type of disfluencies covered in Section 3.

[22] Purple color denotes disfluencies

B Analysis: Inconsistency in Predicted Tags of the Subwords of a Word

Since our model works at the subword level, it could be possible that the subwords of a particular word get different tags (*disfluent*/*fluent*) as the prediction. We call this *inconsistency in tagging*.

According to our findings, only 0.02 % of subwords in the Switchboard test set were marked inconsistently. In our Bengali, Hindi, Malayalam, and Marathi test sets, we found inconsistency in just 0.21 %, 0.06 %, 0.38 %, and 0.03 % subwords. These findings indicate that our model is quite likely to consistently predict the tags across the subwords for any word.

C Qualitative Analysis

We analyze the performance of our base model (trained only on English data) on relatively difficult disfluencies, namely, *false starts* and *corrections*. We also analyse one example of a fluent sentence and we present these examples of potential interest in Table 8.

[23] Cyan color denotes that it is unclear whether the words are intended in the sentence

Type	Description	Example	Transliteration	Gloss	Translation
Filler (3.1)	Insert frequent *filler* phrases in a sentence	সমীক্ষায় দেখা যায়, এই মানে[22] তৃতীয় এবং চতুর্থ শ্রেণী থেকেই অধিকাংশ ছাত্রী স্কুলছুট হয়।	samIkShAYa dekhA YAYa, ei mAne tRRitIYa evaM chaturtha shreNI thekei adhikAMsha ChAtrI skulaChuTa haYa	in_survey see can this means third and fourth grade since most girl_student drop_out_of_school is	According to the survey, this means most girl students drop out of school from third and fourth grade.
Word Repetition (3.2)	Repeat a word unnecessarily	এখনও এখনও হয়তো অনেককে বাইরে থেকে জল আনতে যেতে হয়।	ekhanao ekhanao haYato anekake vAire theke jala Anate Yete haYa	still still maybe many outside from water to_bring to_go is	Many may still still have to fetch water from outside.
Phrase Repetition (3.2)	Repeat a phrase unnecessarily	অর্থাৎ, হতাশার আবহেও বাঁচার আশা জাগানোর সামর্থ্য এই সমাজের এই সমাজের রয়েছে।	arthAt, hatAshAra Avaheo vA.NchAra AshA jAgAnora sAmarthYa ei samAjera ei samAjera raYaeChe	in_other_words, despair in_condition to_live hope to_awaken ability this of_society this of_society there_are	In other words, this society this society has the ability to inspire hope to survive in the face of despair.
Pronoun Repetition (3.2)	Repeat a pronoun unnecessarily	আমরা আমরা রোয়াঙাবাসীকে এক্ষেত্রে নানাভাবে সাহায্য করতে পারি।	AmarA AmarA roYaAn.DAvAsIke ekShetre nAnAbhAve sAhAYYa karate pAri	We we to_Rwandans in_this_case many_ways help to_do can	We we can help Rwandans in many ways.
Partial Word (3.3)	Use *part of word* before the actual word	এর ফলে, নয়ডা অঞ্চলে আগামী বছরগুলিতে জনসংখ্যাও উল্লেখ উল্লেখযোগ্যভাবে বৃদ্ধি পাবে।	era phale, naYaA.DAanchale AgAmI vaCharagulite janasaMkhYAo ullekha ullekhaYogYabhAve vRRiddhi pAve	for_this as_a_result Noida in_region next over_the_years population_too PARTIAL_WORD significantly increase will_get	As a result, the population of the Noida region will also increase signi **signi**ficantly in the coming years.
Missing Syllables (3.3)	Missed a few syllables from the middle of a word; therefore, it is followed by the entire word.	আর্টিফিশিয়াল ইন্টেলিসেন্স ইন্টেলিজেন্সের সূত্র ধরেই বহু সমস্যারই সমাধান আমরা করতে পারি	ArTiphishiYaAla inTelinsera inTelijensera sUtra dharei vahu samasYArai samAdhAna AmarA karate pAri	Artificial of_intellince of_intelligence formula taking many problems solution we to_do can	We can solve many problems with the help of Artificial Intellince **Intelli**g**ence**.
Pronoun Correction (3.3)	Use an incorrect pronoun, then an optional *edit* phrase, then the proper pronoun.	কচ্ছে এদেরকে না মানে একে ভুঙ্গা বলে।	kachChe ederake nA mAne eke bhu NgA vale	in_Kachchh to_them no mean to_this bhunga called	They no I mean it is called bhunga in Kachchh.
Synonym Correction (3.3)	Use of imprecise synonym before the actual word	কোন্ প্রকল্প কবে শেষ হবে, সেই সময় নির্দেশিত নির্দিষ্ট করে_দেওয়া_হয়েছে।	kon prakalpa kave sheSha have, sei samaYa nirdeshita nirdiShTa kare_deoYaA_haYaeChe	which project when complete will_be, that time directed specified has_been_done	When the project will be completed, the time has been directed specified.
False Start (3.4)	Begin a sentence, then abruptly end it and begin a new sentence.	ইতিমধ্যেই রাজ্যে আজ দেশ উন্নয়নের নতুন উচ্চতা অতিক্রম করছে।	itimadhYei rAjYe Aja desha unnaYanera natuna uchchatA atikrama karaChe	already in_the_state today country of_development new height exceed doing	Already in the state Today, the country is exceeding new heights of development.

Table 7: Different types of disfluencies in our synthetic dataset

Disfluent Sentence	Transliteration	Gloss	Translation	Model Output	Comment
তোকে এটা ভাবলে তুই এটা ভাববি যে চার বছর ধরে অল্প অল্প করে কাজ করব	BT: toke eTA bhAvale tui eTA bhAvavi Ye chAra vaChara dhare alpa alpa kare kAja karava	BG: to_you it if_think you it should_think that four year for little little by work will_do	E: If you think about it you would think that I would work little by little for four years	তুই এটা ভাববি যে চার বছর ধরে অল্প করে কাজ করব	Our model is able to detect the *correction* but considers *reduplication* ("অল্প অল্প") as a disfluency. Thus the model removes one "অল্প".
তুই যেটা পড়ছিস তুই যেটা পড়লি সেদিনে সেটাকে সুন্দর করে লেখার চেষ্টা করবি	BT: tui YeTA parChisa tui YeTA parli sedine seTAke sundara kare lekhAra cheShTA karavi	BG: you what reading you what read on_that_day that beautiful by writing try you_should	E: Try to write beautifully what you are reading you read on that day	তুই যেটা পড়লি সেদিনে সেটাকে সুন্দর করে লেখার চেষ্টা করবি	Our model is able to detect the *correction*.
তুমি কি তাহলে ওই কাজটা শেষও হয়ে গেল	BT: tumi ki tAhale oi kAjaTA sheShao haye gela	BG: you what then that work end become gone	E: Are you then that work is over	ওই কাজটা শেষও হয়ে গেল	The sentence starts with a thought ("তুমি কি তাহলে" means "Are you then") and suddenly a new chain of thought is initiated ("ওই কাজটা শেষও হয়ে গেল" means "that work is over"). Our model is able to detect the *false start*.
আসলে আমি না আমি একদমই বুঝে পাচ্ছি না কী করা উচিত	BT: Asale Ami nA Ami ekadamai vujhe pAchChi nA kI karA uchita	BG: Actually I no I absolutely understand getting no what do should	E: Actually I no I have no idea what to do	আসলে আমি একদমই বুঝে পাচ্ছি না কী করা উচিত	The disfluent sentence contains *pronoun repetition* along with an *interregnum*. Our model is able to detect both.
मेरा नहीं मतलब हमारा अतीत अंदरूनी_तौर_पर हमेशा संयमपूर्वक बुना_गया है	HT: merA nahIM matalaba hamArA atIta aMdarUnI_taura_para hameshA saMyamapUrvaka bunA_gayA hai	HG: my no mean our past internally always abstemiously woven is	E: My no I mean our past has always been abstemiously woven inward	हमारा अतीत अंदरूनी तौर पर हमेशा संयमपूर्वक बुना गया है	Our model detects the wrong pronoun along with the *edit* term.
देशातील प्रत्येक शहरात प्रत्येक गावात ही स्वच्छता मोहीम सुरू आहे	MrT: deshAtIla pratyeka shaharAta pratyeka gAvAta hI svachChatA mohIma surU Ahe	MrG: of_the_country each in_city each in_village this cleanliness campaign going_on is	E: This cleaning campaign is going on in every city and every village of the country	देशातील प्रत्येक गावात ही स्वच्छता मोहीम सुरू आहे	We test our model on a fluent sentence. The fluent sentence contains the phrase "प्रत्येक शहरात प्रत्येक गावात" which has two components: "प्रत्येक शहरात" means "in each city" and "प्रत्येक गावात" means "in each village". Our model wrongly assumes that the first component is *corrected* by the second component and hence the model labels the first component as a disfluency.
ওঁরা ছয় মাস ধরে কোনও কোম্পানিতে[23] অ্যাপেল-এর সিইও টিম কুক বলেছেন এই সিদ্ধান্তের পরিণাম সুদূরপ্রসারী হবে	BT: o.NrA Chaya mAsa dhare kono kompAnite aYApela-era siio Tima kuka valeChena ei siddhAntera pariNAma sudUraprasArI have	BG: they six months for any in_company Apple's CEO Tim Cook said this of_decision consequences far-reaching will_be	E: They for six months at a company Apple's CEO Tim Cook said the decision would have far-reaching consequences	ছয় মাস ধরে কোনো কোম্পানিতে অ্যাপেল-এর সিইও টিম কুক বলেছেন এই সিদ্ধান্তের পরিণাম সুদূরপ্রসারী হবে	The disfluent sentence contains a false start "ওঁরা ছয় মাস ধরে কোনো কোম্পানিতে", which makes it ambiguous to find the intended meaning of the utterance. There could be at least three interpretations, which makes the task challenging: • "*At a company* Apple's CEO Tim Cook *has said for six months* that the decision would have far-reaching consequences" • "*At a company* Apple's CEO Tim Cook said the decision would have far-reaching consequences" • "Apple's CEO Tim Cook said the decision would have far-reaching consequences" Our model picks up the first interpretation.
মানে প্রপারলি যদি ইমপ্যাক্টটা আনতে হয় তাহলে এখন থেকে যদি ইমপ্যাক্ট আনতে হয় তাহলে আমাকে সিভিল সার্ভিসে যেতে হবে	BT: maane prapaarali yadi imapyaak.ta.taa aanate hayā taahale ekhana theke yadi imapyaak.ta aanate hayā taahale aamaake sibhila saarbhise yete have	BG: mean properly if the_impact to_bring is then now from if impact to_bring is then me civil to_service to_go will_be	E: I mean, if I have to get the impact properly, if I have to get impact from now on, I have to go to the civil service.	প্রপারলি এখন থেকে যদি ইমপ্যাক্ট আনতে হয় তাহলে আমাকে সিভিল সার্ভিসে যেতে হবে	The disfluent sentence has *code-mixing*. English words "properly", "impact", "civil", "service" are present in the Bengali sentence. This disfluent sentence contains *correction* disfluency type. Our model is able to detect all the disfluencies correctly.

Table 8: Qualitative analysis of our model predictions. BT: Bengali Transliteration, BG: Bengali Gloss, HT: Hindi Transliteration, HG: Hindi Gloss, MrT: Marathi Transliteration, MrG: Marathi Gloss, E: English Translation.

Evaluating Word Embeddings in Extremely Under-Resourced Languages: A Case Study in Bribri

Rolando Coto-Solano
Dartmouth College
rolando.a.coto.solano@dartmouth.edu

Abstract

Word embeddings are critical for numerous NLP tasks but their evaluation in actual under-resourced settings needs further examination. This paper presents a case study in Bribri, a Chibchan language from Costa Rica. Four experiments were adapted from English: Word similarities, WordSim353 correlations, odd-one-out tasks and analogies. Here we discuss their adaptation to an under-resourced Indigenous language and we use them to measure semantic and morphological learning. We trained 96 word2vec models with different hyperparameter combinations. The best models for this under-resourced scenario were Skip-grams with an intermediate size (100 dimensions) and large window sizes (10). These had an average correlation of r=0.28 with WordSim353, a 76% accuracy in semantic odd-one-out and 70% accuracy in structural/morphological odd-one-out. The performance was lower for the analogies: The best models could find the appropriate semantic target amongst the first 25 results approximately 60% of the times, but could only find the morphological/structural target 11% of the times. Future research needs to further explore the patterns of morphological/structural learning, to examine the behavior of deep learning embeddings, and to establish a human baseline. This project seeks to improve Bribri NLP and ultimately help in its maintenance and revitalization.

1 Introduction

Word embeddings are dense vectors that describe the semantics of words (Landauer et al., 1997). They are calculated by collecting the words around a specific word and using them to create a numerical vector that can determine semantic similarity. For example, the words *spinach* and *kale* share neighboring words like *garlic* and *cooked*, which would make them similar to each other and different from words with other neighbors, such as *hammer*. Embeddings can vary in size, or *dimensionality*: Word embeddings from systems like Word2Vec (Mikolov et al., 2013) and GloVe (Pennington et al., 2014) can range from size 10 to 300, while the embeddings from GPT-3 (Brown et al., 2020) can have up to 12288 dimensions.

Embeddings are an integral part of a number of NLP tasks. Because these vectors can be used to calculate cosine distance, one could calculate the distance between words like *king* and *queen*, and examine if this is similar to the distance between *man* and *woman*. Learning these patterns entails that the system is learning the gender distinctions in the words and that it is generalizing patterns across words. This would make an embedding a type of language model which could be enlisted to solve tasks like finding the "odd" word in a triad of words. For example, it could determine that *tiger* is the odd word in the set {*man*, *woman*, *tiger*} given its greater cosine distance from the other two. This is known as the *odd-one-out task*. Embeddings can also solve analogies by using vector algebra. The embedding vectors can be used for the operation *king+woman-man*, and the word closest to this result should be *queen*. These two tasks, odd-one-out and analogies, are some of the main tasks used to evaluate the training of embeddings (Schnabel et al., 2015).

The evaluation of embeddings is relevant not just for these specific semantic tasks, but because contemporary deep learning systems like Transformers (Vaswani et al., 2017), which use embeddings for their learning. Embeddings are used as semantically rich input in the training of sequence-to-sequence tasks such as question answering and machine translation. Because embeddings are so widely used it is necessary to understand their behavior in low-resource environments. This would allow researchers to better describe the lexical structures of Indigenous and other minority languages, and it would also help optimize the training of deep learning algorithms for these languages.

Proceedings of the 29th International Conference on Computational Linguistics, pages 4455–4467
October 12–17, 2022.

Making NLP for Indigenous languages is a complex task because of the chronic lack of data to train models, so determining the optimal configurations to train embeddings for them can help move these efforts forward. There is relatively little research in this regard (Adams et al., 2017; Stringham and Izbicki, 2020), so this paper seeks to add a case study for how to perform these evaluations in an Indigenous language of the Americas.

2 Methodology

This paper will examine the evaluation of word embeddings in extremely under-resourced languages by using a realistic example from an Indigenous language in Central America. The Bribri language (Glottolog `brib1243`) is spoken by approximately 7000 people in Southern Costa Rica (INEC, 2011). It is a Chibchan language and it is vulnerable (Sánchez Avendaño, 2013), which means that some children no longer speak the language. There are efforts to generate NLP tools for Bribri to aid with language maintenance and revitalization efforts, including the construction of digital dictionaries and corpora (Krohn, 2020, 2021; Flores Solórzano, 2017), tools for phonetic analysis (Coto-Solano and Flores-Solórzano, 2016, 2017), machine translation (Feldman and Coto-Solano, 2020; Mager et al., 2021), speech recognition (Coto-Solano, 2021), natural language understanding (Ebrahimi et al., 2021), parsing (Coto-Solano et al., 2021) and morphological analysis (Flores-Solórzano, 2019), as well as tools for the input of the language into digital media (Flores-Solórzano, 2010). The motivation for this paper is to contribute information to analyze the training process of deep learning tools for the language.

2.1 Data preparation

It is regularly the case that, when models are trained in low-resource conditions, what is actually presented is a truncated version of a training set from a high-resource language like English as an attempt to simulate low-resource conditions. (Baevski et al., 2020; Sennrich and Zhang, 2019). However this is not realistic because actual under-resourced environments[1] include obstacles in the creation of the training datasets. Not only is data scarce, but it might lack standardization, making the dataset more sparse than it would be for languages with standardized orthographies and numerous speakers. Here we will review the challenges found in creating a dataset to train embeddings for Bribri.

The data collected contains 90,128 words of monolingual Bribri text. It contains 10,328 unique words, 6,071 of which appear as hapax legomena. It has 10,962 sentences, with an average length of 8.2 ± 7.6 tokens. It includes text from two textbooks (Constenla et al., 2004; Jara Murillo and García Segura, 2013), a dictionary (Margery, 2005), a grammar book (Jara Murillo, 2018), several educational books (Sánchez Avendaño et al., 2021b,a), an oral corpus (Flores Solórzano, 2017) and other sources (Ebrahimi et al., 2021). There are numerous challenges in preparing the data, given that the writing has four main sources of variation: (i) Authors use different orthographic conventions, (ii) there are linguistic phenomena that cause variation, (iii) there is dialectal variation, and (iv) there is idiosyncratic variation across authors.

First, there are multiple orthographies in current use. For example, the word 'tiger' can be written in at least three different ways: *na̲mù̲*, *nãmù̲* and *nɑ̨mù̲*. The vowels in this word are nasal, but the nasal mark changes depending on the author. This problem is compounded by the fact that different authors might use different Unicode characters for the diacritics. For example, the line under the vowel can be variously represented as the combinatorial low line (U+0332), the combinatorial minus sign below (U+0320) or the combinatorial macron (U+0331). These variations were standardized in the dataset so that the three orthographic representations of 'tiger' could all contribute to the embedding for the word.

Second, because the orthography is not standardized, there is much phonological variation that finds its way into the writing. For example, the word *a̲mì* 'mother' can be pronounced as *mì* in rapid speech, and it frequently appears this way in text. This means that half of the information for the 'mother' embedding would be contained in the word *a̲mì* and half would be contained in *mì*, diluting the embedding for both forms.

Third, there is considerable dialectal variation in the dataset. This is present in every human language, but it is magnified in this small dataset. For example, the word 'road' is *ñalà* in the Amubri

[1] The standard terminology is to call these languages *low-resource*. However, they are actually *under-resourced*, as are their communities of speakers. The languages are as fully-fledged as a high-resource language, and contain ample knowledge about the world. They also tend to have speakers and experts willing to compile data. What these languages usually lack are economic resources for the datasets to be created.

dialect and *ñolò̰* in the Coroma dialect. The corpus includes 41 occurrences of *ña̲là̲* and 18 occurences of *ñolò̰*, a 69%-31% distribution. This means that a significant portion of the information for 'road' will not be included in the embedding of either alternative.

Finally, there are numerous idiosyncratic differences in spelling. Lack of standardization is no obstacle for communication and should not slow down the development of written materials, but it does impact NLP. There are numerous words that have different spellings in the same document. For example, the corpus contains 5 spellings for *ta̲î* 'much', and other documents include 9 additional variants such as *tá̲i* and *tâìn*. This variation made it so that some monolingual documents with major internal variation could not be included.

In addition to these sources of variation, Bribri has a different typology of English, which doesn't lend itself well to the word embedding architecture. Bribri is morphologically fusional. Its words regularly take suffixes, and as a result, each Bribri verb can have 23 different conjugations and can also appear with numerous clitics attached to it (Flores Solórzano, 2017). This makes the data itself more sparse than it would be for an English text with the same word count.

2.2 Embedding Evaluation

Once the corpus is in a standardized form, the next challenge is to adapt embedding evaluation techniques to work on Bribri data. There are numerous obstacles to this, including: (i) the paucity of data in the monolingual corpus and (ii) the mismatches between the English and Bribri vocabulary. This subsection will propose four experiments to evaluate Bribri embeddings taking these challenges into account.

2.2.1 Technical Aspects of Training

In order to perform the experiment a total of 96 types of models were trained using the Word2Vec algorithm (Mikolov et al., 2013) implemented in Gensim (Rehurek and Sojka, 2011). This paper performs a systematic examination of different hyperparameters. The models could vary in their type of training (Skip-Gram and CBOW), the size of the embedding (10, 25, 50, 100, 150, 200, 250, 300 dimensions), the window of words used to calculate the embedding (2, 5, 10 words) and the minimum frequency of the tokens considered when constructing the embeddings (n=1 or n=2). Each model was trained 60 times to account for potential variations in the training phase. Therefore, the calculations in the results section use a total of 5760 trained models. The models were trained using a CPU-based Google Colab environment, with a runtime of approximately 8.5 hours.

Four evaluations were performed: (i) A basic similarities test, (ii) a correlation between the Bribri similarities and the similarities in the WordSim353 set (Finkelstein et al., 2001), (iii) a series of odd-one-out tests with both semantically and structurally "odd" words, and (iv) a series of semantic and structural analogies.

2.2.2 Basic similarities

The first test served as a kind of sanity test in order to verify that the system was producing meaningful results. For each model, four similarities were calculated: two semantic and two structural. For the semantic case, the word *aláköl* 'woman' was compared to the hypothetically similar word *wə̀̀m* 'man' and the hypothetically less similar word *na̲mù̲* 'feline, tiger'. The woman-man pair should be more similar than the woman-tiger pair.

For the structural case, two pairs of verbs were selected. The system compared the perfective active form of the verb (e.g. *yö'* 'made', *ña̲'* 'ate soft food') with the perfective middle voice form of the same verb (e.g. *yö̀ne̲* 'was made', *ñà̲ne̲* 'was eaten').

The similarities were calculated for each of the 96 model types, and then the average and standard deviation for each of the four pairs was calculated.

2.2.3 WordSim353 Correlations

The second experiment explores the similarities in the Bribri set and examines if they have the same patterns as other well understood datasets. This experiment will use the word pairs in the WordSim353 dataset (Finkelstein et al., 2001), a set that contains 353 pairs of English words and a measure of their similarity. The experiment calculates the correlation between the WordSim353 pair (e.g. *tiger-cat*, similarity=7.35) and its corresponding Bribri translation (e.g. *na̲mù̲-pûs*, similarity=0.87)[2].

Converting the WordSim353 dataset to be usable in Bribri involved several challenges. First, many

[2]The similarities in WordSimPair353 go from 0 to 10, while the similarities in the Gensim Bribri data go from 0 to 1. A WordSimPair353 score of 5 would be roughly equivalent to a Gensim score of 0.5.

of the pairs had words that were not a part of Bribri culture and did not appear in the corpus (e.g. *Harvard-Yale*). Second, many of the words did not have translations into Bribri (e.g. *vodka-gin*). There are some words from foreign languages that have indeed been borrowed into Bribri (e.g. *banco* 'bank'), but this particular word appeared only once in the corpus so it couldn't be used for the experiment. In the end only 19 pairs could be translated; these are included in Appendix A. We calculated the Pearson correlation between the 19 English-Bribri pairs for the 96 model types. This was done 60 times for each model type, and then the correlations from the 60 runs were averaged for each model type.

2.2.4 Odd-One-Out Testing

The third experiment implements two types of odd-one-out testing: semantic and structural. First, 20 triads were constructed where the odd word would be different from the other two because of its semantic properties (e.g. {*aláköl*, *wẽ̀m*, NAMÙ} 'woman, man, TIGER'). Second, 20 triads of structurally related words were constructed, where the odd word was different because it had a different part of speech, had a different verbal conjugation, or belonged to a different word paradigm. One example of different verbal forms is the triad {*yö'*, *ña'*, KÙNE} 'made, ate, WAS FOUND', where the third verb is in the middle voice. One example of words that belong to different paradigms is {*e'töm*, *bõ̀töm*, MAÑÁL} 'one.flat, two.flat, THREE.HUMAN'. The first two words are counters for nouns in the flat word class, whereas the third word is a counter for human nouns, and therefore should be more salient. All the semantic and structural triads can be found in Appendix A.

The percentage of correct responses for the odd-one-out task was recorded for each of the 60 runs of each of the 96 model types, and the average for each of the model types was calculated. Two separate averages were considered: Semantic and structural averages.

2.2.5 Analogies

The final experiment is the evaluation of analogies. The first step was to use the BATS dataset (Gladkova et al., 2016) to examine which analogies in English could be translated into Bribri. Some of the semantic relationships could be translated. For example, the words for man:woman :: boy:GIRL were present in the corpus. There were numerous pairs that couldn't be used because Bribri expresses different English concepts with the same word (e.g. 'sir/madam' = *akẽ̀kë*), or because the English words are underspecified for their translation. For example, Bribri has words for maternal and paternal uncles, but it doesn't have a single word for 'uncle'. Similarly, there were many English language pairs that were not present in the corpus, such as *euler/mathematician*. A total of 20 analogies (40 pairs) were constructed based on the model from BATS; they can be found in Appendix A. These quartets include 4 hypernymic relationships (e.g. 'quetzal:bird :: corn:PLANT'), 3 "place to live" relationships (e.g. 'fish:water :: man:HOUSE'), 4 antonym relationships (e.g. 'small:big :: white:BLACK'), 3 object/action analogies (e.g. 'ball:play :: book:READ'), one holonymic set ('beam:house :: eye:HEAD') and 5 family relationships. The family relationships are particularly important because they had to be adjusted to the Iroquois family system used in Bribri (Constenla et al., 2004). The language has different words for maternal and paternal relatives, but also for different relationships with siblings (e.g. *ẽ́l* 'sibling of the same sex; cousin from the same sex as the parent', *kutà* 'sister of man', *akẽ̀* 'brother of a woman'). Therefore, it would be important to calculate if the models can learn these culturally relevant relationships.

Finding morphological/structural pairs was more complex because of the differences between English and Bribri. The two languages share some morphological phenomena such a plurals (e.g. 'child:children :: he.she:THEY'), but the rest of the pairs had to be constructed using the morphology of the language. Some analogies paired a noun with its numeral class (e.g. *aláköl*:*wẽ̀m* :: *chìchi*:E'TÖM 'woman:one.human :: dog:ONE.FLAT'), while others used verbal conjugations (e.g. *katõ̀k*:*katèkẹ* :: *yawõ̀k*:YÈKE 'toEat:eating :: toMake:MAKING'). This method of using grammatical phenomena unavailable in English has been used in other adaptations of BATS (Kang and Yang, 2018).

A total of 20 semantic and 20 structural analogies were constructed; the complete list is in Appendix A. These analogies were tested on 60 trial runs for each of the 96 model types. Three results were calculated: (i) The percentage of times that the expected result was the first result of the analogy, (ii) the percentage of times that the expected result was within the first 10 results of the analogy, and (iii) the percentage of times that the expected

Pair	Translation	Similarity
aláköl - wëm	woman - man	0.92 ± 0.04
aláköl - namù	woman - tiger	0.77 ± 0.10
yö' - yönę	made - was made	0.93 ± 0.07
ña' - ñànę	ate - was eaten	0.95 ± 0.03

Table 1: Semantic and structural similarities across all trained models

Type	Size	Window	MinFreq	r
SG	25	10	2	0.33
SG	50	10	2	0.31
SG	25	10	1	0.30
SG	50	10	1	0.30
SG	100	10	2	0.28
SG	**100**	**10**	**1**	**0.28**

Table 2: Average Pearson correlation (r) between Bribri and WordSim353 similarities for top performing models (SG: Skip-gram, MinFreq: Minimum frequency of words included in the model)

result was within the first 25 results of the analogy. These numbers were averaged for each model type.

3 Results

3.1 Basic Similarities

The first set of results confirms that the model contains some knowledge of the Bribri grammar and lexicon. Table 1 shows the similarity between two semantically related pairs ('woman - man', 'woman - tiger') and two structural pairs ('made - was made', 'ate - was eaten'), averaged over the 60 runs and the 96 models. These initial results show that the word for 'woman' is more similar to 'man' (similarity = 0.92) than it is to 'tiger' (0.77). The structural pair shows that the system might be learning the morphological relationships between verbal tokens. Here the middle voice verbs show similar distances to their corresponding perfective active voice verbs ($similarity_{\text{'made-wasMade'}} = 0.93, similarity_{\text{'ate-wasEaten'}} = 0.95$).

3.2 WordSim353 Correlations

After having performed a basic check for learning, the next step is to confirm this with more similarities and check that those resemble the similarities in equivalent English models. Figure 1 shows the average correlation between the Bribri pairs and the WordSim353 pairs for each of the 96 model types. Table 2 shows the average correlation for the top performing model types.

The first visible pattern is that Skip-grams have higher performance than CBOW models. The average correlation for CBOW is r=-0.04±0.09, lower than the r=0.17±0.09 obtained for Skip-grams. The best performing models had correlations ranging between r=0.28 and r=0.33 and all of them were trained as Skip-grams[3]. The second pattern is that the best results were obtained with the largest possible window. It might be the case that the corpus is so small that it needs a large window to be able to learn the relationships between words. A third observable pattern is that small to medium embedding sizes have better performance. The best results occur with embeddings that range between 25 and 100 dimensions; smaller and larger embeddings have relatively lower performance. Finally, for this particular test the minimum frequency of the words does not appear to affect the performance of the models: Models have relatively higher correlations regardless of whether the minimum word frequency is one or two. In general, this experiment shows that the models are learning some Bribri semantics. As an example, figure 2 shows the correlation for one of the top performing models: Skip-gram, 100 dimensions, window=10, minimum word frequency=1. As can be seen in the figure, words with higher similarities in English also have relatively higher similarities in Bribri.

3.3 Odd-One-Out

In this next step, the performance of the system will be examined for both semantics and structural relationships by calculating the performance of the odd-one-out task. Figure 3 shows the percentage of correct responses for semantic and structural triads for the 96 model types studied. Table 3 shows the average percentages for the top performing models.

The main pattern in the odd-one-out experiment is that the semantic and structural triads had similar performance. For example, the highest performing model has an average of 76% of correct responses for semantics, and a slightly lower 70% for structure. This is an encouraging result, as it seems that the system is learning not just the meaning of Bribri words, but also some concepts about the grammar of the Bribri language.

[3]Table 5 shows the 19 pairs that were considered in the correlations. One of them is the trivial case 'tiger-tiger'. This was included because it also appears in WordSim353. If this trivial pair is removed the correlations drop by about half. For example, the model with an average of r=0.33 becomes r=0.17. The better-performing models still have higher correlations.

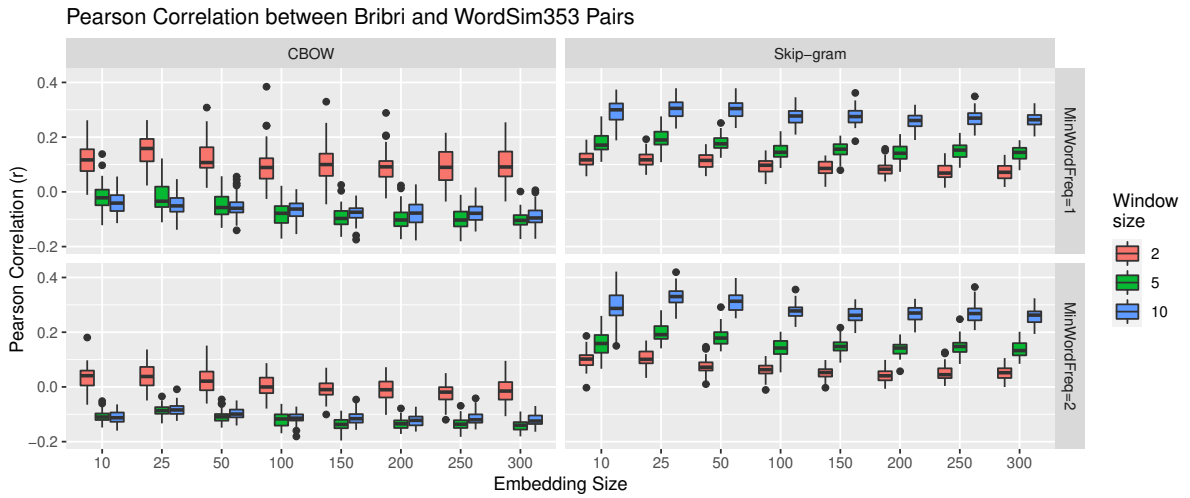

Figure 1: Pearson correlation between WordSim353 similarities and Bribri similarities

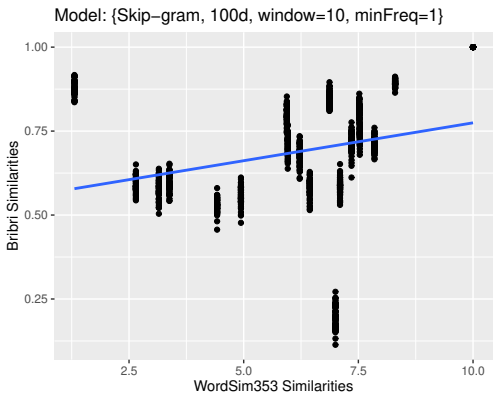

Figure 2: Correlation between WordSim353 similarities and Bribri similarities for a high performing model

Type	Size	Window	MinFreq	Sem	Str
SG	**100**	**10**	**1**	**76**	**70**
SG	200	10	1	75	71
SG	50	10	1	75	68
SG	25	10	2	75	68
SG	300	10	1	75	72
SG	150	10	1	74	71

Table 3: Average of correct responses to the Odd-One-Out task for semantic and structural triads (SG: Skip-gram; Sem: % Correct Semantic; Str: % Correct Structural)

Some of the patterns observed in the similarities experiment were also visible here. Skip-grams with a large window of 10 words were the best performing models. In the odd-one-out experiment the dimensionality doesn't seem to determine the performance. On the other hand, it does seem that models that take all the words into account have better performance; almost all high performing models take all the words in the corpus into account (minimum word frequency = 1). Notably, the highest performing model {Skip-gram, 100 dimensions, window=10, minFreq=1} is also amongst the high-performing models for similarities in table 2.

3.4 Analogies

The final experiment takes 20 semantic and 20 structural word quartets (e.g. *wẽ́m*:*aláköl* :: *kabȅ*:BÙSI 'man:woman :: boy:GIRL') and tries to calculate the fourth word by performing vector algebra with the first three, in effect performing an analogy. Figure 4 shows the average results for the 96 model types trained. The results indicate how often the target word was found as the first result of the vector algebra operation. They also indicate how often the word was found in the top 10 results and in the top 25 results.

Skip-grams performed best, but even in the best models the results for the structural analogies were very low. For semantic analogies in Skip-gram models the target word appeared 14%±6% of the time as the first result, 40%±9% of the time in the top 10 results, and 53%±8% of the time in the top 25 results. Structural analogies for Skip-grams had a much lower performance: The target word appeared only 0.3%±1% of the time as the top result, 4%±4% in the top 10 results, and 7%±5% of the time in the top 25 results. This indicates that there are limits to how much grammatical structure these systems are learning from such a small corpus. Table 4 confirms this pattern. It shows the results

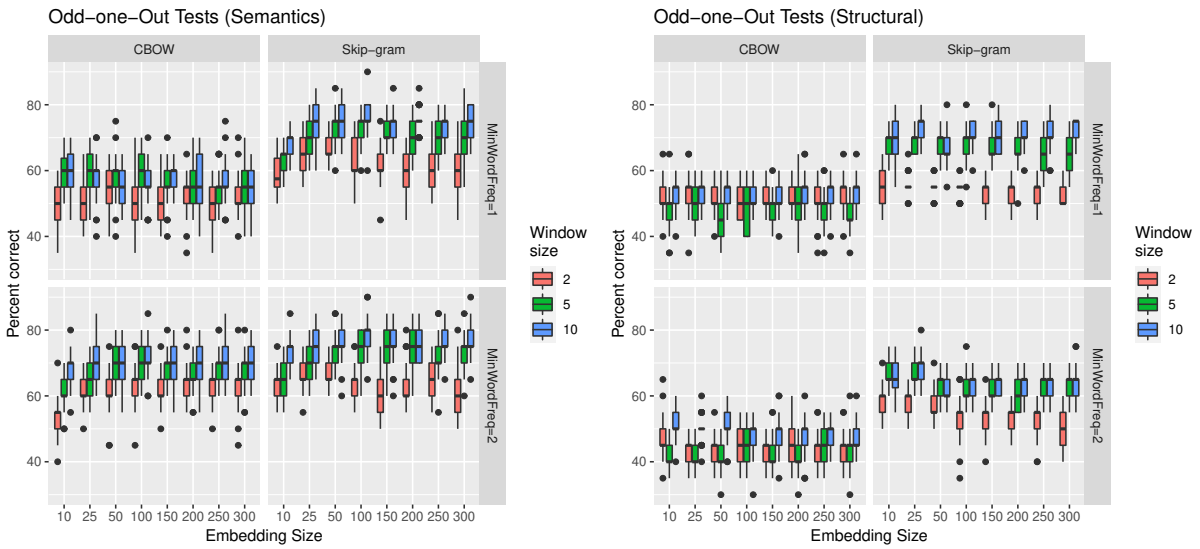

Figure 3: Percentage of correct answers to the Odd-One-Out task for semantic and structural triads

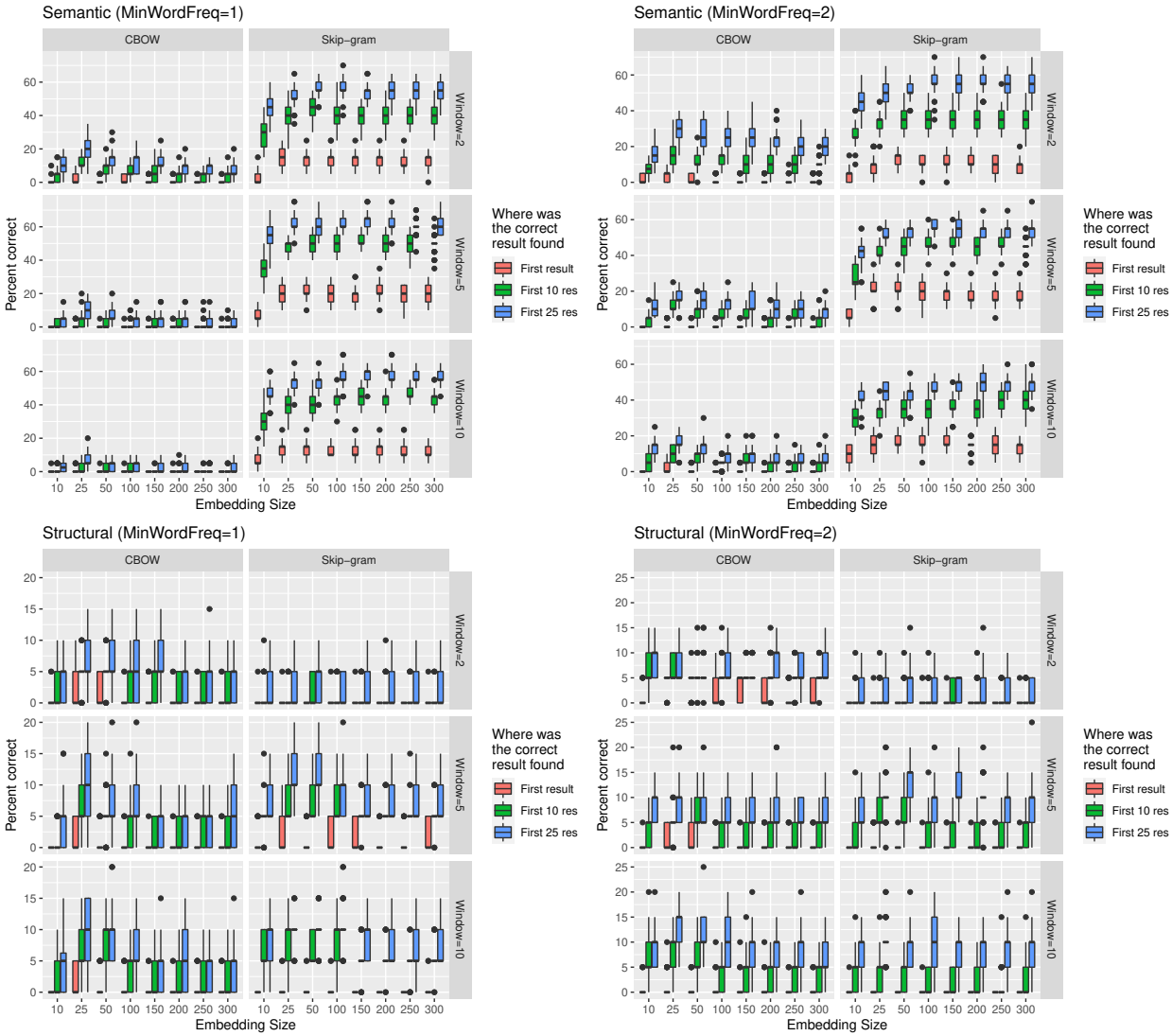

Figure 4: Position of target word in semantic and structural analogies

Type	Size	Window	MinFreq	Sem	Str
SG	100	5	1	63	9
SG	25	5	1	62	12
SG	50	5	1	60	12
SG	**100**	**10**	**1**	**57**	**11**
SG	150	5	2	56	11
SG	100	5	2	55	9

Table 4: Percent of analogies that contained the target word in the first 25 results; top-performing models (SG: Skip-gram; Sem: Semantic; Str: Structural)

for whether the target word was contained in the first 25 words. In semantic analogies the target word appears in the top 25 results between 55% and 63% of the time, but in the structural analogies the target never appears more than 12% of the time.

All of the best results were again Skip-grams. Like in the correlation experiment, medium dimensionality (i.e. around 100 dimensions) seems to be optimal for the analogies. Better-performing windows tend to be shorter than for the odd-one-out task (with more results of window=5), and most high-performing models include all the words in the corpus (mininum frequency=1). Table 4 also contains the model {Skip-gram, 100 dimensions, window=10, minFreq=1}. This model is the only one that appears amongst the top performing models for all of the tests, so it could be considered as the best amongst the examined models. It appears to have an adequate balance for learning all of the tasks presented in this section.

4 Discussion

The results indicate that there are certain combinations of hyperparameters that could provide better performance for extremely under-resourced languages. Embeddings with larger windows and low to mid-sized dimensionality (size=100) appear to learn better. Word2Vec also appears to need every word it can get, and its best performance comes when all words in the corpus are included in the training. Skip-grams might be a better alternative for this task because they avoid overfitting for very frequent words, thereby absorbing more of the information from the relatively sparse dataset at hand (Shobana and Murali, 2021). CBOW has been reported to be better at learning morphological relations, but this was not the case in the Bribri dataset, where the CBOW showed very low rates of structural learning.

Indeed, an open question in this paper is the extent to which the model is learning Bribri grammar. The results from the basic similarities and the odd-one-out experiments seem to indicate that the model learned grammar at roughly the same rate as it did lexical relationships. However, this is contradicted by the analogies experiment, where there was almost no evidence of structural knowledge in the language model. It might be the case that large windows are interfering with local structural learning (e.g. learning which counter words go with which nouns) in favor of semantic knowledge (Levy and Goldberg, 2014), and that the good performance in the odd-one-out structural tests might actually have to do with the semantics of the chosen pairs. More research and a larger test set is necessary to fully understand this effect.

Importantly, the corpus provided here appears to be enough for the system to learn general semantic patterns. Future experiments need to expand on this by providing more pairs, including pairs with more culturally specific words. Another future experiment will be to use the data to train other embeddings such as GloVe, fasttext (Bojanowski et al., 2017), BERT-type dynamic embeddings and multilingual embeddings. In the case of BERT, we need to study the effect of their high dimensionality on low-resource semantic learning. The preliminary hypothesis would be that this increase in dimensionality would have a negative impact in semantic learning. This needs to be verified because those larger embeddings are necessary for numerous deep learning techniques.

One important addition to this experiment would be a human baseline. One of the obstacles in working with Indigenous languages is that there are few people who read and write these languages. Moreover, there are few speakers of Bribri that are familiar with tasks like analogies, which are an unusual type of exercise mostly reserved for academic contexts. Therefore, carrying out such an experiment is relatively complex. This is the next step in this project, a necessary one so that the Skip-gram's performance can be placed in context.

The main ethical concern in the project has to do with data sovereignty. The results will be used to train deep learning systems in collaboration with Bribri partners, but there is currently no overarching community organization which controls the access to Bribri data. This paper has restricted itself to data that is publicly available or licensed

through Creative Commons, so no private or new data was included. However, an effort needs to be made so that the results of this project benefit the Bribri partners in particular and the Bribri community in general, by using them to produce useful NLP tools.

5 Conclusions

This paper presents an evaluation of a word embedding in a truly under-resourced environment. It presents a methodology for embedding evaluation that could be adapted to other Indigenous languages. It provides evidence that some embedding configurations have better performance when dealing with under-resourced scenarios (i.e. Skip-gram trained embeddings, with around 100-dimensions, where the skip-gram attempts to predict words in a window of size 10 and use every word in the corpus to calculate their predictions). The results also confirm the intuition that semantics might be easier to learn than morphology, particularly in morphologically complex languages with little (and sparse) data. Finally, the results here will be used to continue NLP work in the Bribri language with the objective of training deep learning models and understanding their performance and errors, with the ultimate goal of using these to contribute to efforts of language revitalization.

6 Acknowledgments

The author wishes to thank Prof. Samantha Wray, Prof. Carla Jara, Dr. Sofía Flores Solórzano and the reviewers of this paper for their help at different stages of the project. The author also wishes to thank the audience at the Universidad Michoacana de San Nicolás de Hidalgo for their comments on a previous version of this project.

References

Oliver Adams, Adam Makarucha, Graham Neubig, Steven Bird, and Trevor Cohn. 2017. Cross-lingual Word Embeddings for Low-Resource Language Modeling. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 1, Long Papers*, pages 937–947.

Alexei Baevski, Yuhao Zhou, Abdelrahman Mohamed, and Michael Auli. 2020. wav2vec 2.0: A framework for Self-Supervised Learning of Speech Representations. *Advances in Neural Information Processing Systems*, 33:12449–12460.

Piotr Bojanowski, Edouard Grave, Armand Joulin, and Tomas Mikolov. 2017. Enriching Word Vectors with Subword Information. *Transactions of the association for computational linguistics*, 5:135–146.

Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. 2020. Language Models are Few-Shot Learners. *Advances in neural information processing systems*, 33:1877–1901.

Adolfo Constenla, Feliciano Elizondo, and Francisco Pereira. 2004. *Curso Básico de Bribri*. Editorial de la Universidad de Costa Rica.

Rolando Coto-Solano. 2021. Explicit Tone Transcription Improves ASR Performance in Extremely Low-Resource Languages: A case study in Bribri. In *Proceedings of the First Workshop on Natural Language Processing for Indigenous Languages of the Americas*, pages 173–184, Online. Association for Computational Linguistics.

Rolando Coto-Solano and Sofía Flores-Solórzano. 2016. Alineación forzada sin entrenamiento para la anotación automática de corpus orales de las lenguas indígenas de costa rica. *Kánina*, 40(4):175–199.

Rolando Coto-Solano and Sofía Flores-Solórzano. 2017. Comparison of Two Forced Alignment Systems for Aligning Bribri Speech. *CLEI Electron. J.*, 20(1):2–1.

Rolando Coto-Solano, Sharid Loáiciga, and Sofía Flores-Solórzano. 2021. Towards universal dependencies for bribri. In *Proceedings of the Fifth Workshop on Universal Dependencies (UDW, SyntaxFest 2021)*, pages 16–29.

Abteen Ebrahimi, Manuel Mager, Arturo Oncevay, Vishrav Chaudhary, Luis Chiruzzo, Angela Fan, John Ortega, Ricardo Ramos, Annette Rios, Ivan Vladimir, et al. 2021. Americasnli: Evaluating zero-shot natural language understanding of pretrained multilingual models in truly low-resource languages. *arXiv preprint arXiv:2104.08726*.

Isaac Feldman and Rolando Coto-Solano. 2020. Neural Machine Translation Models with Back-Translation for the Extremely Low-Resource Indigenous Language Bribri. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 3965–3976.

Lev Finkelstein, Evgeniy Gabrilovich, Yossi Matias, Ehud Rivlin, Zach Solan, Gadi Wolfman, and Eytan Ruppin. 2001. Placing Search in Context: The Concept Revisited. In *Proceedings of the 10th International Conference on World Wide Web*, pages 406–414.

Sofía Flores-Solórzano. 2010. Teclado Chibcha: Un software lingüístico para los sistemas de escritura de las lenguas bribri y cabécar. *Revista de Filología y Lingüística de la Universidad de Costa Rica*, pages 155–161.

Sofía Flores-Solórzano. 2019. La modelización de la morfología verbal bribri - Modeling the Verbal Morphology of Bribri. *Revista de Procesamiento del Lenguaje Natural*, 62:85–92.

Sofia Margarita Flores Solórzano. 2017. *Un primer corpus pandialectal oral de la lengua bribri y su anotacion morfologica con base en el modelo de estados finitos*. Ph.D. thesis, Universidad Autónoma de Madrid.

Sofía Flores Solórzano. 2017. Corpus oral pandialectal de la lengua bribri.

Anna Gladkova, Aleksandr Drozd, and Satoshi Matsuoka. 2016. Analogy-Based Detection of Morphological and Semantic Relations with Word Embeddings: What Works and What Doesn't. In *Proceedings of the NAACL Student Research Workshop*, pages 8–15.

INEC. 2011. Población total en territorios indígenas por autoidentificación a la etnia indígena y habla de alguna lengua indígena, según pueblo y territorio indígena. In Instituto Nacional de Estadística y Censos, editor, *Censo 2011*.

Carla Victoria Jara Murillo. 2018. *Gramática de la lengua bribri*. E Digital.

Carla Victoria Jara Murillo and Alí García Segura. 2013. *Se' ttö' bribri ie Hablemos en bribri*. E Digital.

Hyungsuk Kang and Janghoon Yang. 2018. The Analogy Test Set Suitable to Evaluate Word Embedding Models for Korean. *Journal of Digital Contents Society*, 19(10):1999–2008.

Haakon Krohn. 2020. Elaboración de una base de datos en XML para un diccionario bribri–español español–bribri en la web. *Porto das Letras*, 6(3):38–58.

Haakon S. Krohn. 2021. Diccionario digital bilingüe bribri. http://www.haakonkrohn.com/bribri.

Thomas K Landauer, Darrell Laham, Bob Rehder, and Missy E Schreiner. 1997. How well can passage meaning be derived without using word order? A comparison of Latent Semantic Analysis and humans. In *Proceedings of the 19th annual meeting of the Cognitive Science Society*, pages 412–417.

Omer Levy and Yoav Goldberg. 2014. Dependency-Based Word Embeddings. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 302–308.

Manuel Mager, Arturo Oncevay, Abteen Ebrahimi, John Ortega, Annette Rios Gonzales, Angela Fan, Ximena Gutierrez-Vasques, Luis Chiruzzo, Gustavo Giménez Lugo, Ricardo Ramos, et al. 2021. Findings of the AmericasNLP 2021 Shared Task on Open Machine Translation for Indigenous Languages of the Americas. In *Proceedings of the First Workshop on Natural Language Processing for Indigenous Languages of the Americas*, pages 202–217.

Enrique Margery. 2005. *Diccionario fraseológico bribri-español español-bribri*, second edition. Editorial de la Universidad de Costa Rica.

Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013. Efficient Estimation of Word Representations in Vector Space. *arXiv preprint arXiv:1301.3781*.

Jeffrey Pennington, Richard Socher, and Christopher D Manning. 2014. GloVe: Global Vectors for Word Representation. In *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*, pages 1532–1543.

Radim Rehurek and Petr Sojka. 2011. Gensim: Python Framework for Vector Space Modelling. *NLP Centre, Faculty of Informatics, Masaryk University, Brno, Czech Republic*, 3(2):2.

Carlos Sánchez Avendaño. 2013. Lenguas en peligro en Costa Rica: vitalidad, documentación y descripción. *Revista Káñina*, 37(1):219–250.

Tobias Schnabel, Igor Labutov, David Mimno, and Thorsten Joachims. 2015. Evaluation Methods for Unsupervised Word Embeddings. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 298–307.

Rico Sennrich and Biao Zhang. 2019. Revisiting Low-Resource Neural Machine Translation: A Case Study. *arXiv preprint arXiv:1905.11901*.

J Shobana and M Murali. 2021. Improving Feature Engineering by Fine Tuning the Parameters of Skip Gram Model. *Materials Today: Proceedings*.

Nathan Stringham and Mike Izbicki. 2020. Evaluating Word Embeddings on Low-Resource Languages. In *Proceedings of the First Workshop on Evaluation and Comparison of NLP Systems*, pages 176–186.

Carlos Sánchez Avendaño, Alí García Segura, et al. 2021a. *Se' Dalì Diccionario y Enciclopedia de la Agricultura Tradicional Bribri*. Editorial de la Universidad de Costa Rica.

Carlos Sánchez Avendaño, Alí García Segura, et al. 2021b. *Se' Má Diccionario-Recetario de la Alimentación Tradicional Bribri*. Editorial de la Universidad de Costa Rica.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. *Advances in Neural Information Processing Systems*, 30.

A Experimental Items

Bribri	Translation	Equivalent English	English Similarity
01. namù - pûs	big feline - cat	tiger - cat	7.35
02. namù - namù	big feline - big feline	tiger - tiger	10
03. ya' - kukuò̱	drank - ear	drink - ear	1.31
04. ya' - kò̱	drank - mouth	drink - mouth	5.96
05. ya' - ña'	drank - ate soft things	drink - eat	6.87
06. aláala - amì	baby - mother	baby - mother	7.85
07. ya' - amì	drank - mother	drink - mother	2.65
08. chakò̱ - kàlwö	food - fruit	food - fruit	7.52
09. dù - dakarò	bird - chicken	bird - cock	7.1
10. chakò̱ - dakarò	food - chicken	food - rooster	4.42
11. dayë̀ - kañík	sea - jungle	coast - forest	3.15
12. namù - íyiwak	big feline - animal	tiger - animal	7
13. ajkuò̱ - wò̱bala	skin - eye	skin - eye	6.22
14. dalì - inúköl	merchandise - money	grocery - money	5.94
15. ñalà - ala'r	road - children	street - children	4.94
16. skél - si'	five.flat - month	five - month	3.38
17. ñíwe - shkèna	during the day - to dawn	day - dawn	7.53
18. wḗm - aláköl	man - woman	man - woman	8.3
19. ñalà - ká	road - time, space, place	street - place	6.44

Table 5: WordSim353 Correlations and translated Bribri pairs. WordSim353 English similarities go from 0 to 10. A Bribri/Gensim score of 0.5 is roughly equivalent to a WordSim353 score of 5.

Bribri	Translation
01. aláköl - wḗm - NAMÙ	woman - man - TIGER
02. yḗ - amì - Ù	father - mother - POT
03. amíla - amì - ALÀ	maternal aunt - mother - CHILD
04. wìm - sàl - DÙ	howler monkey - spider monkey - BIRD
05. chamù - ikuò̱ - DAKARÒ	banana - corn - CHICKEN
06. íyiwak - kàlwak - KUÁ	animal - bug - PLANT
07. skuë̀ - kano' - KALṍM	mouse - lowland paca - PLANTAIN
08. tkabë̀ - só - DIKÓ	snake - cockroach - PEJIBAYE DATE
09. átu - ali' - TABÈ	beans - yucca - KNIFE
10. dalôlô - sarûrû - BITSÎ	black - white - LONG
11. tsuru' - balo' - ÀRROS	cocoa - chicha drink - RICE
12. chkṍk - katṍk - YṍK	to eat - to eat hard foods - TO DRINK
13. katṍk - ñúk - CHKṍK	to eat hard food - to eat soft food - TO EAT (GENERIC)
14. ulà - kalò̱ - Ù	hand - foot - HOUSE
15. kàsir - kò̱chi - NIMÀ	peccary - pig - FISH
16. wò̱bla - yík - KALò̱	eye - nose - FOOT
17. datsi' - apàio - SI'	clothes - shirt - MOON
18. chkì - îñe - DI'	yesterday - today - RIVER, WATER
19. dë' - mík - YÖ'	arrived - went - TO MAKE
20. Amùbali - Kua'rö - TALÌRI	Town of Amubri - Town of Buenos Aires - SALITRE RIVER

Table 6: Semantic groups of Odd-One-Out Triads. Words in small caps are the "odd" word.

Bribri	Translation
01. kàl - íyök - SHKŐK	tree - soil - TO GO
02. kuá- íyiwak - SÈRKE	plant - animal - (SOMEONE) LIVES
03. ta - tö - DI'	with - ergative marker - WATER, RIVER
04. ska - ki - AKĕ́KËPA	towards - at - OLD PERSON
05. ye' - be' - DÙ	I - you - BIRD
06. sa' - se' - ÑALÀ	we (exclusive) - we (inclusive) - ROAD
07. tsîr - tsikirîrî - CHKÌ	small - yellow - YESTERDAY
08. bềrie - buáala - MÌK	big - beautiful - WHEN
09. yö' - ña' - KÙNE	made - ate - WAS FOUND (middle voice)
10. kít - ya' - SÙNE	wrote - drank - WAS SEEN (middle voice)
11. yők - inúk - YA'	to drink - to play - DRANK (perfective)
12. shkők - sauk - DË'	to walk - to see - WENT (perfective)
13. ché - yawé - SÚ	saying - drinking - SAW (perfective)
14. mi'ke - yawèke - DË'	going - drinking - WENT (perfective)
15. tkër - tulur - DUR	sitting - sitting.plural - STANDING
16. a'r - tkë'nik - TÉN	hanging - hanging.plural - STICKING IN
17. awí - awì - E'	there (near) - there (far) - THIS ONE
18. dió - dià - NE'	there (below, near) - there (below, far) - THIS ONE (ONLY HEARD)
19. e'töm - bötöm - MAÑÁL	one.long - two.long - THREE.HUMAN
20. e'köl - ból - MAÑÀTÖM	one.human - two.human - THREE.LONG

Table 7: Structural groups of Odd-One-Out Triads. Words in small caps are the "odd" word.

Bribri	Translation	Type
01. wẽ́m:aláköl :: yẽ́:AMÌ	man : woman :: father : MOTHER	Family relation
02. kabë̀:bùsi :: wẽ́m:ALÁKÖL	boy : girl :: man : WOMAN	Family relation
03. amíla:naù :: aláköl:wẽ́M	maternal aunt : maternal uncle :: woman:MAN	Family relation
04. akë̀:kutà :: wẽ́m:ALÁKÖL	brother of woman : sister of man :: man:WOMAN	Family relation
05. talà:yẽ́ :: wìke:AMÌ	paternal grandfather : father :: maternal grandmother : MOTHER	Family relation
06. balo':yàne :: ali':ÑÀNE	chicha drink : was drunk :: yucca : WAS EATEN	Object/Action
07. wẽ́m:dur :: dù:TKËR	man : stands :: bird : RESTS ON A SURFACE	Object/Action
08. bola:inùk :: uyë̀jkuö:ÀRITSÖK	ball : to play :: book : TO READ	Object/Action
09. nimà:di' :: wẽ́m:Ù	fish : water :: man : HOUSE	Place to live
10. nimà:di' :: buà:ÚK	fish : water :: iguana : BURROW	Place to live
11. aláköl:ù :: dù:KÀL	woman : house :: bird : TREE	Place to live
12. pulí:kàl :: pú:DÙ	ceiba tree : tree :: eagle : BIRD	Hypernym
13. kabék:dù :: ikuö̀:KUÁ	quetzal bird : bird :: corn : PLANT	Hypernym
14. kaë̀:dù :: átu:KUÁ	pava negra bird : bird :: beans : PLANT	Hypernym
15. dakarò:dù :: chìchi:ÍYIWAK	chicken : bird :: dog : ANIMAL	Hypernym
16. sẽ́:ù :: wö̀bla:WŐKIR	beam : house :: eyes : FACE	Holonym
17. tóttô:darềrề :: bua':SULÛ	easy : hard :: good : BAD	Antonym
18. kéwe:ukö̀ki :: wéshke:ÛRIKI	before : after :: inside : OUTSIDE	Antonym
19. kájke:dikì :: wőnik:TSÌ	above : below :: in front : BEHIND	Antonym
20. tsîr:bềrie :: sarûrû:DALÔLÔ	small : big :: white : BLACK	Antonym

Table 8: Quartets for the semantic analogies. Words in small caps are the target words.

Bribri	Translation
01. aláköl:e'köl :: chìchi:E'TÖM	woman : one.human :: dog : ONE.LONG
02. chamù:e'töm :: dawás:E'K	banana : one.long :: year : ONE.ROUND
03. dur:ië'ten :: tkër:TULUR	stand.sg : stand.pl :: sit.sg : SIT:PL
04. a'r:tkë'nik :: tén:TULUR	hanging.sg : hanging.pl :: stuck.sg : STUCK:PL
05. alà:ala'r :: ie':IE'PA	child : children :: 3sg : THEY
06. awá:awápa :: yéria:YÉRIAPA	healer : healers :: hunter : HUNTERS
07. tsîr:tsítsi :: buáala:buàmbuáala	small.sg : small.pl :: beautiful.sg : BEAUTIFUL.PL
08. bêrie:wîwî :: wáwán:WÂNWÂN	big.sg : big.pl :: few.sg : FEW.PL
09. e'köl:e'töm :: ből:BÖTÖM	one.human : one.flat :: two.human : TWO.FLAT
10. e'k:e'tökicha :: bök:BÖTÖKICHA	one.round : one.time :: two.round : TWO.TIMES
11. ma'tk:máshmash :: siê:SIÉLSIEL	red : reddish :: blue : BLUEISH
12. dalôlô:dalóshdalosh :: sarûrû:SARÚLSARUL	black : blackish :: white : WHITEISH
13. yö':yöne :: sú:SÙNE	made : was made :: saw : WAS SEEN
14. shka':shkàne :: stsë':STSËNE	walk : [someone] walked [there] :: heard : WAS HEARD
15. awí:awì :: aí:AÌ	that (near) : that (far) :: that (above, near) : THAT (ABOVE, FAR)
16. awì:aì :: awí:AÍ	that (far) : that (above, far) :: that (near) : THAT (ABOVE, NEAR)
17. awí:awì :: dió:DIÀ	that (near) : that (far) :: that (below, near) : THAT (BELOW, FAR)
18. sú:saú :: yö':YAWÓ	saw : see! :: made : MAKE!
19. të':tèke :: yö':YAWÈKE	hit : hitting :: made : MAKING
20. katök:katèke :: yawök:YÈKE	to eat (hard things) : eating (hard things) :: to make : MAKING

Table 9: Quartets for the structural analogies. Words in small caps are the target words.

Applying Natural Annotation and Curriculum Learning to Named Entity Recognition for Under-Resourced Languages

Valeriy Lobov
MIPT
lobov.va@phystech.edu

Alexandra Ivoylova
RSUH
a.m.ivoylova@gmail.com

Serge Sharoff
School of Languages
University of Leeds, UK
s.sharoff@leeds.ac.uk

Abstract

Current practices in building new NLP models for low-resourced languages rely either on Machine Translation of training sets from better resourced languages or on cross-lingual transfer from them. Still we can see a considerable performance gap between the models originally trained within better resourced languages and the models transferred from them. In this study we test the possibility of (1) using natural annotation to build synthetic training sets from resources not initially designed for the target downstream task and (2) employing curriculum learning methods to select the most suitable examples from synthetic training sets. We test this hypothesis across seven Slavic languages and across three curriculum learning strategies on Named Entity Recognition as the downstream task. We also test the possibility of fine-tuning the synthetic resources to reflect linguistic properties, such as the grammatical case and gender, both of which are important for the Slavic languages. We demonstrate the possibility to achieve the mean F1 score of 0.78 across the three basic entities types for Belarusian starting from zero resources in comparison to the baseline of 0.63 using the zero-shot transfer from English. For comparison, the English model trained on the original set achieves the mean F1-score of 0.75. The experimental results are available from https://github.com/ValeraLobov/SlavNER

1 Introduction

The use of pre-trained language models (PLMs), such as BERT (Devlin et al., 2018), has significantly improved accuracy of many NLP tasks, such as POS tagging and Named Entity Recognition (Tenney et al., 2019). It is also possible to achieve good quality transfer of the classifiers across the languages using multilingual PLMs (Conneau et al., 2020; Hu et al., 2020). However, lesser resourced languages still present a considerable problem. First, the amount of raw text data available for their pretraining is on the order of magnitudes smaller than what is available for bigger languages, such as English. Second, zero- or few-shot multilingual transfer comes with the price of a performance gap, when a model tested on lesser resourced recipient languages is less accurate than the original donor language model. This comes partly because of the linguistic differences between the donor and recipient languages and partly because of the lower quality of pre-trained embeddings obtained on smaller corpora for lesser resourced recipient languages (Vulić et al., 2020).

We propose a method to improve the accuracy of models for lesser resourced languages by what we call “natural" annotation, i.e., when some desired linguistic properties are derived from annotations arising as a by-product of a natural activity which is not directly related to the task of the model. Parallel corpora provide an example of natural annotation arising from human translations which are not produced by the their translators for the purpose of Machine Translation or Word Sense Disambiguation. Similarly, in our study we use natural annotation from Wikipedia categories, which provide a sufficient number of Named Entity (NE) examples even for lesser resourced languages. This allows production of synthetic corpora for such languages as Belarusian, which has no native NER resources. As an example, the Belarusian Wikipedia contains more than 130 thousand entries with more than four thousand entries about people (as of April 2022), sufficient for creating synthetic training sets.

Instead of the initial problem with the availability of training data, the use of synthetic corpora leads to the problem with having potentially millions of noisy annotated sentences per language. Therefore, we need to estimate their usefulness for training a model in the recipient languages. In this study we experiment with applying Curriculum

Proceedings of the 29th International Conference on Computational Linguistics, pages 4468–4480
October 12–17, 2022.

Learning methods (Bengio et al., 2009; Zhu et al., 2021) to synthetic corpora with natural annotation.

The contributions of this study are as follows:

- how to build a corpus with natural annotation from the available resources in Wikipedias;
- how to improve it by adjusting its linguistic properties;
- how to choose the curriculum learning strategy for corpora of this kind.

2 Methodology

2.1 Synthetic dataset

In this study we concentrate on building synthetic corpora for seven Slavic languages with three NE categories – PERsons, LOCations and ORGanizations. The main idea behind our experiment is:

1. to select annotated sentences in a better resourced donor language (we use the English WikiNER (Pan et al., 2017) for all experiments),
2. to produce synthetic corpora by Machine Translation of relevant annotated sentences in the recipient languages (we use Google Translate), and
3. to replace their annotations with the relevant Wikipedia entries from the recipient languages.

At the same time, Machine Translation is especially unreliable for NEs. For example, Google Translate renders ***Chain*** *was declared the winner as a result* into Slovene as ***Veriga*** *je bil zaradi tega razglašen za zmagovalca* with a literal translation of *Chain* as 'a series of connected links or things'.

Therefore, in Step 1 we select annotated WikiNER sentences with a single known NE (PER in this sentence), replace it with a pronoun placeholder into ***He*** *succeeded in purifying penicillin* and replace the placeholder to a range of known NEs from Wikipedia, for example, for this context ***Einstein/Romanova/Arhit*** *je bil zaradi tega razglašen za zmagovalca*, see Table 2. This way, each sentence in our synthetic datasets contains at least one NE (and no more than two of them for PER). This makes it quite different from natural datasets: the entity distribution is much more uniform in synthetic data.

The known NEs are determined as those with matching categories in the respective Wikipedias, see Table 1. For example, the English PER NEs are from categories like 1791 BIRTHS → *James Buchanan* → Джэймс Б'юкенен (in Belarusian).

We produced three versions of the synthetic datasets:

S1 Replacements of the parallel set of named entities;

S2 Replacements of the parallel set of named entities, while taking into account the grammatical case, number and gender;

S3 Replacements of a maximum available number of entities from the respective categories

For S1 and S2 we ensured that the NEs are available in all languages, so that the synthetic datasets can be completely parallel. We also used rules to normalize the names of the entries, e.g., Ломоносов, Михаил Васильевич ('Lomonosov, Mikhail Vasilyevich' in Russian) was normalized to Михаил Ломоносов 'Mikhail Lomonosov', which is the form used in texts.

S1 does not take into account the grammatical properties, such as the case, gender and number. Besides, it contains quite a lot of duples or quasi-duples (e.g., 'A.Lukashenko' and 'Alexander Lukashenko') which were deleted during S2 creation: this is the reason why PER number decreased for S2. Thus, S2 is generated by constraining the respective contexts for the respective categories. For example, *Romanova* is a female name, while *je bil razglašen* is a form requiring the male name, so in S2 we only generate synthetic sentences respecting these constraints:

en **Albert Einstein** was declared the winner as a result.
sl.m.sg → **Albert Einstein** je bil zaradi tega razglašen za zmagovalca
en **Anastasia Romanova** warned that the syndicalists aims were in perpetuating syndicalism itself.
sl.f.sg → **Anastazija Romanova** opozorila je, da so sindikalistični cilji ohranjanje samega sindikalizma.

The number and gender for the ORG type was detected from the syntactic properties of the head of each ORG name using udpipe (Straka et al., 2016).

Finally, S3 drops the constraint of having parallel NEs by using all of the NEs detected for a given language through the categories in Table 1 (the contexts remain parallel, though). The number of PER NEs becomes unreasonable, so we constrained it to 20,000 PER NEs per language.

Table 1: Labels of NER-related categories in Wikipedias.

Language	Wiki entries	PER Categories	LOC Categories	ORG Categories
en	6,489,550	Born	Cities|Countries	Organizations
be	217,410	Нарадзіліся	Краіны|Гарады	Арганізацыі
bg	281,108	Родени	Градове|Държави	Организации
cs	502,428	Narozen	Měst|Země	Organizace
pl	1,519,696	Urodzeni	Miasta|Kraje	Organizacje
ru	1,814,092	Родились	Города|Страны	Организации
sl	176,025	Rojeni	Mesta|Države	Organizacije
uk	1,151,062	Народились	Міста|Країни	Організації
#Parallel		4,492	1,709	239
#Total		140,000	22,709	34,329

Table 2: Synthetic dataset contents.

	PER	LOC	ORG
S1			
Entities	7,889	1,709	239
Contexts	7,694	1,535	348
S2			
Entities	4,492	3,178	239
Contexts	7,439	1,535	348
S3			
Entities	20,000	6,178	9,828
Contexts	7,439	1,535	1,087

We compare training on our synthetic corpora against two commonly used baselines with random training dataset ordering:

B1 Zero-shot transfer of a multilingual PLM with training on the original English dataset;

B2 Training on the target language dataset produced by Machine Translation of the original English dataset (The original entities were marked in the text by special symbols like '|0|' so that the labels would not be lost during MT).

Each training dataset consists of 12000 randomly generated examples from the available set of entities and contexts for particular language. The test dataset was produced by manual cleaning of approximately 10,000 tokens taken from WikiNER for each of the eight languages including English. The NE counts for this dataset are listed in Table 3.

2.2 NER setup

To test the contribution of our synthetic corpora with curriculum learning mechanisms, we rely on a competitive NER approach, which is based on XLM-R (Conneau et al., 2019). In the starting stage, we compared the multilingual BERT (Devlin et al., 2018) and XLM-R on our datasets. Subsequently, the XML-R model was used for all predictions as it showed slightly better results in all of our experiments (as also shown in other downstream applications). In addition, XLM-R offered better zero-shot transfer results, which is relevant for our B1 baseline.

2.3 Curriculum learning methods

The main purpose of using curriculum learning (CL) in the proposed method is to ensure better ordering of potentially noisy examples. Since the training dataset may contain samples from incompatible semantic categories within the same NE type (e.g., Greek philosophers doing research on penicillin), the CL model must reduce the negative effect of noisy examples and thereby make learning process more stable.

In total, we implemented three models to determine the order of significance for the examples:

C1 order data by sample size, i.e. sentence length

C2 order data by average confidence in predicted named entities

C3 order data by perplexity value

These models cover our hypotheses regarding aspects of data complexity for the named entity recognition task. C1 reflects the amount of information that the model is gradually learning. Long sentences carry significantly more information and therefore may be more difficult to mark up entities in them. Another complexity measure for NER

Table 3: Counts of NER-related categories in the test dataset.

Entities	en	be	bg	cs	pl	ru	sl	uk
PER	340	479	321	377	414	373	360	298
LOC	336	821	974	530	716	552	690	948
ORG	358	125	87	195	195	292	104	138

task samples is the average confidence of the CL model in predicted named entities (model C2). Inspired by the article (Zhu et al., 2021) we also calculate the probability that the entity word has a label from the gold-label markup. By sorting the training dataset in descending order of such probability, we thereby place incorrect examples at the end and give priority to high-confidence data.

The model needs to be based on a different architecture, for which we used Bi-LSTM-CRF. It calculates the optimal tag probability as:

$$p^*(x) = \max_{y' \in \boldsymbol{y}} \left[CRF_F(x, y') + CRF_B(x, y') \right] \quad (1)$$

where $\boldsymbol{y}$ means the original set of labels, x - input word, CRF_F and CRF_B represent the calculated probabilities from the CRF model for forward and backward pass of the Bi-LSTM network. Then the average probability among the entities words is computed as:

$$\text{C2}_{score}(S) = \frac{1}{|\boldsymbol{w}|} \sum_{i=1}^{|\boldsymbol{w}|} (p^*(w_i)) \quad (2)$$

where $\boldsymbol{w}, \boldsymbol{w} \subset S$ mean subset of words from the sentence S which are named entities.

Finally C3 is perplexity of CL model for a given sentence. Perplexity is a measurement of how well a probability model p predicts a sample. This metric is calculated below:

$$PP(p) = 2^{-\sum_x p(x) \log_2 p(x)} \quad (3)$$

So, we are curious if this metric is suitable for sorting training samples and improving the learning process in our task. As follows from the definition, the training dataset needs to be sorted in the ascending order of this metric.

Several studies in curriculum learning (Wang et al., 2019a,b; Castells et al., 2020) suggest discarding the top-N% of the most complex samples from the dataset. Therefore, in our study, we conducted several experiments to understand whether the model should use all samples or a certain percentage during training.

3 Results

Because of space constraints, here we compare the performance of CL models and of synthetic datasets for two languages – Polish and Belarusian. Belarusian is the most under-resourced language out of those tested in our study, in terms of the amount of (1) pre-trained data, (2) Wikipedia entries and (3) available training resources. Polish is a better resourced language from a different Slavic branch (West vs East for Belarusian), and also it is written in a mix of Latin and Polish-specific characters, which creates non-trivial tokenization problems, as its tokenizer is influenced by major European languages. The results for Polish reported below are similar to those for Belarusian, which emphasizes the universality of the methods for Slavic languages presented in this paper. The applicability of these methods to other language groups is beyond the scope of this article, but nevertheless this is an interesting topic for further research.

Table 4 reports summary F1 metrics on WikiNER evaluation dataset for all models and training datasets for Belarusian. Table 5 - a similar table for Polish. The main overall measure is the mean F1 scores for the three main NE categories (omitting the Other category, which does not indicate the source of errors). The best mean score for Belarusian is 0.78, for Polish 0.79. For comparison, the same English model trained on the same set achieves the mean F1-score of 0.75, while the best zero-shot transfer achieves 0.63 for Belarusian and 0.68 for Polish. The full set of results for 7 Slavic languages is presented in Figure 6 in the Appendix.

3.1 Synthetic datasets

There is considerable variation between the NE types. PER is easy to detect by using any training set, this is followed by LOC, while ORG is more difficult. There is a very considerable improvement from using more linguistic information when moving from S1 to S2. S1 is mostly not better than either of the two baselines, i.e. B1 – zero-shot transfer or B2 – direct MT of the English training

Table 4: Belarusian: F1 scores for all entities, models and datasets. The scores in bold mean the best results for a CL model within a dataset. The underlined scores are the best scores for each NE type across all models and datasets. Last row, *mean score*, represents mean F1 score for three entity types.

Entity	Model	B1	B2	S1	S2	S3
PER	C1	0.89	0.90	0.85	0.91	**0.92**
	C2	0.88	0.91	**0.89**	**0.93**	0.91
	C3	0.88	0.89	0.88	0.92	0.91
LOC	C1	0.60	0.64	0.59	0.78	**0.76**
	C2	0.55	0.58	0.51	**0.81**	0.60
	C3	0.64	0.66	**0.70**	0.79	0.57
ORG	C1	0.40	0.43	0.30	0.56	**0.62**
	C2	0.42	0.44	0.33	0.61	0.58
	C3	0.38	0.45	**0.37**	**0.62**	0.57
Mean score	C1	0.63	0.66	0.58	0.75	**0.77**
	C2	0.62	0.64	0.57	**0.78**	0.70
	C3	0.63	0.67	**0.65**	0.77	0.68

Table 5: Polish: F1 scores for all entities, models and datasets. The scores in bold mean the best results for a CL model within a dataset. The underlined scores are the best scores for each NE type across all models and datasets. Last row, *mean score*, represents mean F1 score for three entity types.

Entity	Model	B1	B2	S1	S2	S3
PER	C1	0.88	0.89	0.86	0.92	0.93
	C2	0.88	0.90	0.82	**0.93**	**0.94**
	C3	0.87	0.89	**0.87**	0.92	0.92
LOC	C1	0.64	0.67	0.54	0.76	**0.81**
	C2	0.66	0.68	0.55	0.75	0.68
	C3	0.68	0.70	**0.64**	**0.80**	0.63
ORG	C1	0.54	0.56	0.50	0.61	0.57
	C2	0.51	0.55	0.48	0.60	**0.59**
	C3	0.43	0.48	**0.55**	**0.66**	0.54
Mean score	C1	0.68	0.70	0.63	0.76	**0.77**
	C2	0.68	0.71	0.62	0.76	0.74
	C3	0.66	0.69	**0.68**	**0.79**	0.70

set. Also, surprisingly, the very simple B2 setup is often better than the popular zero-shot transfer of B1.

S2 is the best dataset overall, so the expected improvements by using larger training sets in S3 did not materialise. The most likely reason is that most frequent location entities in the test dataset are country names, which are well covered by S2, as most of them are translated across the Wikipedias; thus, adding even a much bigger set of examples with rare place names could not improve the scores (though we suppose that S3 could show better results on a test dataset with rare place names). The effect of gender disambiguation used in S2 and S3 (opposite to S1) clearly show the benefits for PER recognition, even though this is an easier task for all of the models and datasets. The most difficult category is ORG. Some organization names are rendered without translation, i.e., *General Motors*, so they are easier to recognize. ,On the other hand, many ORG names are linguistically diverse with a complicated structure, for example, *Międzynarodowe Centrum Badań nad Ochroną i Konserwacją Dziedzictwa Kulturowego* 'International Centre for the Study of the Preservation and Restoration of Cultural Property', with the problem in their detection persisting across any available NER model.

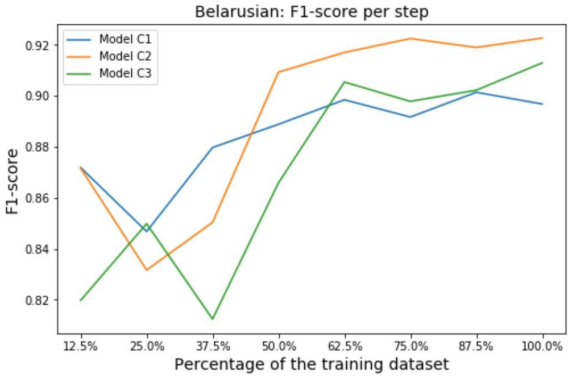

Figure 1: F1 score on evaluation dataset after each training step on belarusian S2 dataset.

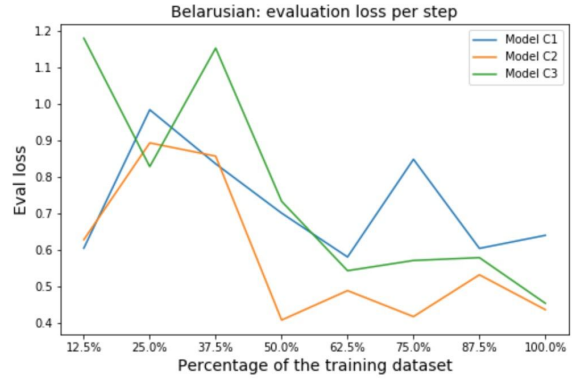

Figure 2: Evaluation loss after each training step on Belarusian S2 dataset.

3.2 CL Models

In our experiments with models, we wanted to evaluate the impact of curriculum learning on the synthetic dataset. First of all, ordering of the training samples improves target metrics compared to baseline, basic XLM models with random sampling. For locations and organizations the difference is substantial: 0.15 and 0.17 of the F1-score points respectively. For persons, the difference is not so big, because initially there are enough entities and context examples for persons, so it is easier to recognize PER entities irrespectively of the order of training.

One of the key goals of using curriculum learning in this study is to reduce the impact of noisy synthetic examples. For example, using the Belarusian dataset S2 and model C2 (average probability of predicting gold-label entity tags), these examples are listed as the highest difficulty for predicting LOC:

1. Шпаер таксама ў спісе з двума. ‘Speyer is also in the list with the two’

2. Рэвалюцыя 1897-1898 гадоў адкрыла дзверы для больш шырокіх ведаў, і пачалося шмат даследаванняў, пра якія гл. Валета ‘The 1987-1898 revolution opened the doors for wider knowledge, and many researches were started, about which see Valeta’

The C2 model reasonably placed these examples at the end of the training dataset. In the first example, the location Шпаер (‘Speyer’) is similar to the surname, so the model is not confident between deciding it is a location and a person. In the second example, the structure is not correct since the end of the sentence contains an artifact from Wikipedia "about which see [entity]". This sentence tells about the revolution, not about the location, so the context of the entity is incomplete. In the examples shown above, the model correctly identified the entities, but was not sure about them, so the curruculum learning method assigned them the higher scores.

Comparing the methods of curriculum learning, we wanted to determine which method makes learning more robust and efficient. For this experiment, we took the S2 dataset, as it provides the best performance among all other datasets. As for the experiment setup, we trained three models C1, C2, C3 on the Belarusian S2 dataset with following configuration: 1 epoch, Adam optimizer, learning rate equals 1e-5, weight decay equals 0.01. The choice of a single epoch and a low learning rate is primarily due to the fact that the NER model trained on synthetic datasets quickly overfits. For example, for Belarusian and Polish S2, the validation loss increases after 12.5% of the training dataset of the second epoch. On the charts 1 and 2 the key evaluation metrics of three CL models are shown. Since we found out that one epoch is enough for training, on the figures each training step is 12.5% of the training dataset. In particular, Figure 2 shows that the ranking of training examples based on the metric of the average probability of predicting named entities (model C2) demonstrates more stable training. Moreover, this model is the fastest gaining more than 90% of the F1 metric and has the minimum evaluation loss among all models.

Also, in the same experiment we found out that throwing out 10% of most complicated data leads to mean decrease in F1 measure by around 3% with all CL models. This means that the model

should rather see the most complex examples in order to demonstrate the best quality.

3.3 Errors analysis

We manually analyzed and compared the predictions of models finetuned on S2 and S3 datasets for Polish and Belarusian. The quality for Belarusian is comparable to the other languages in the set. There were no problems found caused by tokenization, though sometimes the models tend to misjudge the boundaries of the entities, including spaces, brackets or commas.

Most errors for S2 are predictably caused by cases: so PER entities which are not in nominative case often do not get a correct label, or case endings may not be labelled (our model labels every BPE-token), e.g., Polish *Artura Rubinsteina* 'Arthur Rubinstein' loses its PER label for endings *-a* in both name and surname and the markup looks like *'Artur, PER'* and *'#a, O'*. The same goes for other entities, for example, Polish *Kanady* 'Canada' (genitive) may not get annotated in certain contexts at all, and ORG entity *Partią Republikańską* 'Republican Party' (instrumental) has an unlabelled case ending *ą*.

Some of other easily explainable errors are caused by entity type: seas, rivers, and mountains were not present in the synthetic datasets, so the models may only partially recognize entities like Belarusian Галілейскага мора 'Sea of Galilee'.

Another quite common, but not so obvious error concerns unlabelled country names: although the models must have seen them, country names are often lost. It is worth mentioning that country names are often unlabelled in real data, such as WikiNER and SlavicNER (Piskorski et al., 2021), as well.

When the models are finetuned on the S3 dataset, the error types generally remain the same, though their quantity is slightly smaller. We noticed that sometimes the context obviously provides a false label, i.e., in Belarusian sentence which contains пад націскам Рима пазбавіў 'under the pressure of Rome [he] deprived' Rome in genitive case gets the PER label, because it is followed by a verb common for PER entities.

4 Related work

4.1 NER and synthetic corpora

The idea of using synthetic data for augmentation was a natural consequence of the development of ML models which need a lot of annotated data in order to learn. Mentions of synthetic data in NLP can be found as early as in 2000s (i.e., (Talbot, 2003)); an obvious solution was to generate training data for ML models using rule-based tools, as it was done to improve machine translation in (Hu et al., 2007). Starting from late 2000s, the idea of using synthetic corpora grew more popular and was applied in various areas of research, for example, for generation of animations for sign languages (Schnepp et al., 2010), or for Implicit Discourse Relation Recognition (Lan et al., 2013). Nowadays, there are plenty of works using synthetic data for improvement, and generating synthetic datasets is considered a common technique. For example, (Kvapilíková et al., 2020) article used unsupervised machine translation to build a synthetic dataset and improve quality on parallel corpus mining task in low-resource languages, which is especially relevant for our research. Some recent works using this approach are, for example, (Li et al., 2021), (Hosseini et al., 2021) or (Whitfield, 2021), the latter introducing GPT-2 model for data generation. Also, method proposed in (Sellam et al., 2020) uses synthetic sentences for BERT pre-training, which contain a wide variety of lexical, syntactic, and semantic diversity. The key goal for synthetic datasets is to provide the maximum variation of text data in order to make the target model more robust.

NER typically involves one of three gold standards: MUC, CoNLL, or BBN, all created by costly manual annotation. One of the first datasets for NER in the CoNLL standard was created in 2003 (Sang and De Meulder, 2003) and covered two languages (English and German); the NER-related categories consisted of PER, LOC, ORG and MISC labels. There were also several datasets in CoNLL standard created around 2010, one of them being WikiGold (Balasuriya et al., 2009) – 40K tokens of Wikipedia articles, manual annotation; the other, Web (Ratinov and Roth, 2009), contained 8K tokens taken from the Web, and the third dataset used Twitter as a resource and contained approximately 34K tokens (Ritter et al., 2011). Later additions are represented by Broad Twitter Corpus (Derczynski et al., 2016) and WiNER (Ghaddar and Langlais, 2017), although the list of NER-annotated corpora isn't exhausted by those.

There are also attempts to create synthetic

corpora with NER-annotation, mostly for low-resource languages, i.e., (Jónsson et al., 2021) for Icelandic. Cross-lingual transfer is a viable method as well and it was used for creation of a synthetic Chinese NER corpus in 2014, for example (Fu et al., 2014). There has also been a recent set of experiments on cross-lingual annotation, for example, for African languages (Adelani et al., 2021). There is also an approach to use a small number of examples via triggers (Lin et al., 2020).

4.2 Curriculum learning

Curriculum Learning (CL) is a learning technique where the order of training samples depends on their complexity for the target model. This paradigm resembles human learning: a gradual increase in the complexity of training examples makes learning process more qualitative. The original technique was proposed by (Bengio et al., 2009). But for supervised learning and NLP in particular, (Elman, 1993) article is one of the first where an idea similar to curriculum learning is applied. The author emphasized that the order of training data is important, where the "small" data comes first.

More recently, the curriculum learning approach was used quite extensively in NLP. Different measures of complexity have been proposed, depending on the task. For instance, (Tay et al., 2019) article addresses the problem of reading comprehension of long texts. Curriculum learning based on the answerability and understandability of texts effectively improves training process. In addition, (Platanios et al., 2019) apply curriculum learning to neutral machine translation to learn better and converge faster. The framework presented in the paper shows certain samples to the model at certain times according to their complexity and model competence at that moment.

NER task requires a comprehensive metric to use for curriculum learning. The study closest to our work (Zhu et al., 2021) uses several strategies to organize their training data. All of them use probability of entities from the gold label dataset calculated by the CL model. In one case, the average confidence of the model is calculated from all entities from the input sentence. In another, the averaged confidence of the model is considered only for named entities. This approach is relevant due to the fact that goal is to recognize named entities correctly, other tokens in the sentence are not so important. Furthermore, there are other techniques for filtering and ordering samples for the NER task, especially suitable for generated datasets. (Liu et al., 2021) filter data samples with specific entity based on complex criteria: for example, if this entity is too frequent in the training dataset.

4.3 Cross-lingual transfer

Cross-lingual transfer is relevant when there is sufficient data for training a model for one language and no such data for another language. A model which is trained on the data for one language and applied on the data of the other (zero-shot transfer), usually shows worse results than on donor language material; thus leading to a transfer gap, which can be measured as the difference between the performance of the same model on donor and recipient languages (Hu et al., 2020). In order to use cross-lingual transfer, language spaces must be aligned, and the models which provide higher quality vector spaces perform better. This idea has become popular since 2014, when embedding methods produced high quality spaces which are almost isomorphic across languages and which can be aligned by using small seed dictionaries (Mikolov et al., 2013).

Modern models, such as multilingual BERT or XLM-RoBERTa, are even more efficient at building cross-lingual vector spaces (Conneau et al., 2020). Some recent studies in this area showed that the transfer gap if the donor language is English normally is not bigger than 0.25 (0.14 on average) for a set of recipient languages (Hu et al., 2020; Ruder et al., 2021). For example, the Natural Language Inference (NLI) task for Slavic languages has a transfer gap of 0.07. As for the NER task with the use of cross-lingual transfer, there was a recent analysis of zero-shot transfer between English and Korean (Kim et al., 2021). More specifically on the topic of this paper, cross-lingual transfer for NER on Slavic languages has been discussed in (Sharoff, 2020), and shared tasks such as SlavicNER (Piskorski et al., 2021) are specifically aimed at NER for Slavic languages.

5 Conclusions

In this work we demonstrated how to achieve prediction quality for lesser resourced languages without any performance gap introduced by zero-

shot transfer or Machine Translation. A synthetic corpus of about 10,000 sentences produced from a combination of naturally annotated data and machine translation from a better resourced language can produce better results than training on the source dataset for the better resourced language. However, the key to this success is an accurate model of important linguistic phenomena (case, number and gender, as in our S2 and S3 datasets), as without this the synthetic corpus (our version S1) is worse than the zero-shot baseline (B1). The second major contribution is the importance of the curriculum learning strategy. Any strategy for choosing the examples helps, but the average probability (from a different model) and perplexity usually help more than simple ordering by the sentence length. Also discarding the most difficult items does not help, as the models improve when seeing more data.

The negative result of this study is that a bigger collection NEs (S3) did not improve over the smaller set (S2). More research is needed into understanding the reasons for this. Better NE selection can help in matching the test dataset, while this might cause problems in applying the models beyond the test dataset. The study is also limited to a specific set of languages as well as to a single downstream task. In our future research we want to explore Wikipedias and similar resources with natural annotation for building synthetic training sets for more languages and for other downstream tasks, such as NE linking or ontology building.

References

David Ifeoluwa Adelani, Jade Abbott, Graham Neubig, Daniel D'souza, Julia Kreutzer, Constantine Lignos, Chester Palen-Michel, Happy Buzaaba, Shruti Rijhwani, Sebastian Ruder, Stephen Mayhew, Israel Abebe Azime, Shamsuddeen H. Muhammad, Chris Chinenye Emezue, Joyce Nakatumba-Nabende, Perez Ogayo, Aremu Anuoluwapo, Catherine Gitau, Derguene Mbaye, Jesujoba Alabi, Seid Muhie Yimam, Tajuddeen Rabiu Gwadabe, Ignatius Ezeani, Rubungo Andre Niyongabo, Jonathan Mukiibi, Verrah Otiende, Iroro Orife, Davis David, Samba Ngom, Tosin Adewumi, Paul Rayson, Mofetoluwa Adeyemi, Gerald Muriuki, Emmanuel Anebi, Chiamaka Chukwuneke, Nkiruka Odu, Eric Peter Wairagala, Samuel Oyerinde, Clemencia Siro, Tobius Saul Bateesa, Temilola Oloyede, Yvonne Wambui, Victor Akinode, Deborah Nabagereka, Maurice Katusiime, Ayodele Awokoya, Mouhamadane MBOUP, Dibora Gebreyohannes, Henok Tilaye, Kelechi Nwaike, Degaga Wolde, Abdoulaye Faye, Blessing Sibanda, Orevaoghene Ahia, Bonaventure F. P. Dossou, Kelechi Ogueji, Thierno Ibrahima DIOP, Abdoulaye Diallo, Adewale Akinfaderin, Tendai Marengereke, and Salomey Osei. 2021. MasakhaNER: Named entity recognition for African languages. *Transactions of the Association for Computational Linguistics*, 9:1116–1131.

Dominic Balasuriya, Nicky Ringland, Joel Nothman, Tara Murphy, and James R Curran. 2009. Named entity recognition in wikipedia. In *Proceedings of the 2009 workshop on the people's web meets NLP: Collaboratively constructed semantic resources (People's Web)*, pages 10–18.

Yoshua Bengio, Jérôme Louradour, Ronan Collobert, and Jason Weston. 2009. Curriculum learning. In *Proceedings of the 26th Annual International Conference on Machine Learning*, ICML '09, page 41–48, New York, NY, USA. Association for Computing Machinery.

Thibault Castells, Philippe Weinzaepfel, and Jérôme Revaud. 2020. Superloss: A generic loss for robust curriculum learning. In *NeurIPS*.

Alexis Conneau, Kartikay Khandelwal, Naman Goyal, Vishrav Chaudhary, Guillaume Wenzek, Francisco Guzmán, Edouard Grave, Myle Ott, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Unsupervised cross-lingual representation learning at scale.

Alexis Conneau, Shijie Wu, Haoran Li, Luke Zettlemoyer, and Veselin Stoyanov. 2020. Emerging cross-lingual structure in pretrained language models. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 6022–6034, Online. Association for Computational Linguistics.

Leon Derczynski, Kalina Bontcheva, and Ian Roberts. 2016. Broad twitter corpus: A diverse named entity recognition resource. In *Proceedings of COLING 2016, the 26th International Conference on Computational Linguistics: Technical Papers*, pages 1169–1179.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. BERT: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*.

Jeffrey Elman. 1993. Learning and development in neural networks: the importance of starting small. *Cognition*, 48:71–99.

Ruiji Fu, Bing Qin, and Ting Liu. 2014. Generating chinese named entity data from parallel corpora. *Frontiers of Computer Science*, 8(4):629–641.

Abbas Ghaddar and Philippe Langlais. 2017. Winer: A wikipedia annotated corpus for named entity recognition. In *Proceedings of the Eighth International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 413–422.

Arian Hosseini, Siva Reddy, Dzmitry Bahdanau, R Devon Hjelm, Alessandro Sordoni, and Aaron Courville. 2021. Understanding by understanding not: Modeling negation in language models. *arXiv preprint arXiv:2105.03519*.

Junjie Hu, Sebastian Ruder, Aditya Siddhant, Graham Neubig, Orhan Firat, and Melvin Johnson. 2020. Xtreme: A massively multilingual multi-task benchmark for evaluating cross-lingual generalization. *arXiv preprint arXiv:2003.11080*.

Xiaoguang Hu, Haifeng Wang, and Hua Wu. 2007. Using rbmt systems to produce bilingual corpus for smt. In *Proceedings of the 2007 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning (EMNLP-CoNLL)*, pages 287–295.

Haukur Páll Jónsson, Vésteinn Snæbjarnarson, Haukur Barri Símonarson, and Vilhjálmur Thorsteinsson. 2021. En-is synthetic parallel named entity robustness corpus.

Jongin Kim, Nayoung Choi, Seunghyun Lim, Jungwhan Kim, Soojin Chung, Hyunsoo Woo, Min Song, and Jinho D. Choi. 2021. Analysis of zero-shot crosslingual learning between English and Korean for named entity recognition. In *Proceedings of the 1st Workshop on Multilingual Representation Learning*, pages 224–237, Punta Cana, Dominican Republic. Association for Computational Linguistics.

Ivana Kvapilíková, Mikel Artetxe, Gorka Labaka, Eneko Agirre, and Ondřej Bojar. 2020. Unsupervised multilingual sentence embeddings for parallel corpus mining. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics: Student Research Workshop*, pages 255–262, Online. Association for Computational Linguistics.

Man Lan, Yu Xu, and Zheng-Yu Niu. 2013. Leveraging synthetic discourse data via multi-task learning for implicit discourse relation recognition. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 476–485.

Jianfu Li, Yujia Zhou, Xiaoqian Jiang, Karthik Natarajan, Serguei Vs Pakhomov, Hongfang Liu, and Hua Xu. 2021. Are synthetic clinical notes useful for real natural language processing tasks: A case study on clinical entity recognition. *Journal of the American Medical Informatics Association*, 28(10):2193–2201.

Bill Yuchen Lin, Dong-Ho Lee, Ming Shen, Ryan Moreno, Xiao Huang, Prashant Shiralkar, and Xiang Ren. 2020. TriggerNER: Learning with entity triggers as explanations for named entity recognition. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 8503–8511, Online. Association for Computational Linguistics.

Zihan Liu, Feijun Jiang, Yuxiang Hu, Chen Shi, and Pascale Fung. 2021. NER-BERT: A pre-trained model for low-resource entity tagging. *arXiv preprint arXiv:2112.00405*.

Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013. Efficient estimation of word representations in vector space. *arXiv preprint arXiv:1301.3781*.

Xiaoman Pan, Boliang Zhang, Jonathan May, Joel Nothman, Kevin Knight, and Heng Ji. 2017. Cross-lingual name tagging and linking for 282 languages. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1946–1958.

Jakub Piskorski, Bogdan Babych, Zara Kancheva, Olga Kanishcheva, Maria Lebedeva, Michał Marcińczuk, Preslav Nakov, Petya Osenova, Lidia Pivovarova, Senja Pollak, Pavel Přibáň, Ivaylo Radev, Marko Robnik-Sikonja, Vasyl Starko, Josef Steinberger, and Roman Yangarber. 2021. Slav-NER: the 3rd cross-lingual challenge on recognition, normalization, classification, and linking of named entities across Slavic languages. In *Proceedings of the 8th Workshop on Balto-Slavic Natural Language Processing*, pages 122–133, Kiyv, Ukraine. Association for Computational Linguistics.

Emmanouil Antonios Platanios, Otilia Stretcu, Graham Neubig, Barnabas Poczos, and Tom M. Mitchell. 2019. Competence-based curriculum learning for neural machine translation.

Lev Ratinov and Dan Roth. 2009. Design challenges and misconceptions in named entity recognition. In *Proceedings of the Thirteenth Conference on Computational Natural Language Learning (CoNLL-2009)*, pages 147–155.

Alan Ritter, Sam Clark, Oren Etzioni, et al. 2011. Named entity recognition in tweets: an experimental study. In *Proceedings of the 2011 conference on empirical methods in natural language processing*, pages 1524–1534.

Sebastian Ruder, Noah Constant, Jan Botha, Aditya Siddhant, Orhan Firat, Jinlan Fu, Pengfei Liu, Junjie Hu, Dan Garrette, Graham Neubig, et al. 2021. Xtreme-r: Towards more challenging and nuanced multilingual evaluation. *arXiv preprint arXiv:2104.07412*.

Erik F Sang and Fien De Meulder. 2003. Introduction to the conll-2003 shared task: Language-independent named entity recognition. *arXiv preprint cs/0306050*.

Jerry C Schnepp, Rosalee Wolfe, and John McDonald. 2010. Synthetic corpora: a synergy of linguistics and computer animation. *Visual Communication and Technology Education*.

Thibault Sellam, Dipanjan Das, and Ankur Parikh. 2020. BLEURT: Learning robust metrics for text generation. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7881–7892, Online. Association for Computational Linguistics.

Serge Sharoff. 2020. Finding next of kin: Cross-lingual embedding spaces for related languages. *Journal of Natural Language Engineering*, 26:163–182.

Milan Straka, Jan Hajič, and Jana Straková. 2016. UDPipe: Trainable pipeline for processing CoNLL-U files performing tokenization, morphological analysis, POS tagging and parsing. In *Proceedings of the Tenth International Conference on Language Resources and Evaluation (LREC'16)*, Portorož, Slovenia.

David Talbot. 2003. Learning translations from comparable corpora.

Yi Tay, Shuohang Wang, Luu Anh Tuan, Jie Fu, Minh C. Phan, Xingdi Yuan, Jinfeng Rao, Siu Cheung Hui, and Aston Zhang. 2019. Simple and effective curriculum pointer-generator networks for reading comprehension over long narratives.

Ian Tenney, Dipanjan Das, and Ellie Pavlick. 2019. BERT rediscovers the classical NLP pipeline. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 4593–4601, Florence, Italy. Association for Computational Linguistics.

Ivan Vulić, Sebastian Ruder, and Anders Søgaard. 2020. Are all good word vector spaces isomorphic? In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 3178–3192, Online. Association for Computational Linguistics.

Wei Wang, Isaac Caswell, and Ciprian Chelba. 2019a. Dynamically composing domain-data selection with clean-data selection by "co-curricular learning" for neural machine translation. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 1282–1292, Florence, Italy. Association for Computational Linguistics.

Wei Wang, Ye Tian, Jiquan Ngiam, Yinfei Yang, Isaac Caswell, and Zarana Parekh. 2019b. Learning a multi-domain curriculum for neural machine translation.

Dewayne Whitfield. 2021. Using gpt-2 to create synthetic data to improve the prediction performance of nlp machine learning classification models. *arXiv preprint arXiv:2104.10658*.

Wenjing Zhu, Liu Jian, Xu Jinan, Chen Yufeng, and Zhang Yujie. 2021. Improving low-resource named entity recognition via label-aware data augmentation and curriculum denoising. In *Proceedings of the 20th Chinese National Conference on Computational Linguistics*, pages 1131–1142, Huhhot, China. Chinese Information Processing Society of China.

A Appendix

A.1 Results for 7 Slavic languages

In this section we provide Figure 6 with comprehensive results of our experiments in seven Slavic languages.

Table 6: Mean F1 scores per 3 entities types for all languages, models and datasets. The scores in bold mean the best results for a CL model within a dataset. The underlined scores are the best scores for each NE type across all models and datasets.

Language	Model	B1	B2	S1	S2	S3
Belarussian	C1	0.63	0.66	0.58	0.75	**0.77**
	C2	0.62	0.64	0.57	**0.78**	0.70
	C3	0.63	0.67	**0.65**	0.77	0.68
Ukrainian	C1	0.63	0.67	0.61	0.71	0.69
	C2	0.64	0.68	0.67	**0.75**	0.70
	C3	0.65	0.67	**0.68**	0.73	**0.72**
Russian	C1	0.61	0.63	0.60	0.73	0.70
	C2	0.63	0.62	**0.63**	**0.77**	**0.73**
	C3	0.62	0.63	0.61	0.75	0.72
Slovenian	C1	0.67	0.69	0.60	0.70	0.69
	C2	0.68	0.71	0.64	**0.78**	**0.75**
	C3	0.68	0.70	**0.65**	0.73	0.71
Polish	C1	0.68	0.70	0.63	0.76	**0.77**
	C2	0.68	0.71	0.62	0.76	0.74
	C3	0.66	0.69	**0.68**	**0.79**	0.70
Bulgarian	C1	0.62	0.64	0.61	0.74	0.70
	C2	0.65	0.66	**0.64**	0.76	**0.74**
	C3	0.66	0.67	0.63	**0.78**	0.71
Czech	C1	0.64	0.67	0.65	0.72	0.71
	C2	0.65	0.68	**0.68**	**0.76**	**0.75**
	C3	0.65	0.69	0.66	0.74	0.74

Taking Actions Separately: A Bidirectionally-Adaptive Transfer Learning Method for Low-Resource Neural Machine Translation

Xiaolin Xing, Yu Hong*, Minhan Xu, Jianmin Yao, Guodong Zhou
School of Computer Science and Technology, Soochow University, China
{actuallyxxl, tianxianer, cosmosbreak5712}@gmail.com
{jyao, gdzhou}@suda.edu.cn

Abstract

Training Neural Machine Translation (NMT) models suffers from sparse parallel data, in the infrequent translation scenarios towards low-resource source languages. The existing solutions primarily concentrate on the utilization of Parent-Child (**PC**) transfer learning. It transfers well-trained NMT models on high-resource languages (namely Parent NMT) to low-resource languages, so as to produce Child NMT models by fine-tuning. It has been carefully demonstrated that a variety of PC variants yield significant improvements for low-resource NMT. In this paper, we intend to enhance PC-based NMT by a bidirectionally-adaptive learning strategy. Specifically, we divide inner constituents (6 transformers) of Parent encoder into two "teams", i.e., T1 and T2. During representation learning, T1 learns to encode low-resource languages conditioned on bilingual shareable latent space. Generative adversarial network and masked language modeling are used for space-shareable encoding. On the other hand, T2 is straightforwardly transferred to low-resource languages, and fine-tuned together with T1 for low-resource translation. Briefly, T1 and T2 take actions separately for different goals. The former aims to adapt to characteristics of low-resource languages during encoding, while the latter adapts to translation experiences learned from high-resource languages. We experiment on benchmark corpora SETIMES, conducting low-resource NMT for Albanian (Sq), Macedonian (Mk), Croatian (Hr) and Romanian (Ro). Experimental results show that our method yields substantial improvements, which allows the NMT performance to reach BLEU4-scores of 62.24%, 56.93%, 50.53% and 54.65% for Sq, Mk, Hr and Ro, respectively.

1 Introduction

NMT has achieved significant improvements (Bahdanau et al., 2015; Vaswani et al., 2017) in recent years. Nevertheless, It heavily relies on large-scale observable parallel corpora. As a result, NMT generally fails to perform perfectly in an infrequent translation scenario, where the available parallel data for training is sparse. For example, the size of training data for NMT between English (En) and Macedonian (Mk) is about 200K, which is significantly smaller than that (582M) between English (En) and German (De). The issue has been widely known as low-resource NMT.

The existing studies attempt to overcome the issue primarily by 1) producing cross-language embeddings, 2) constructing bilingual shareable latent space for encoding, and 3) transferring well-trained models to low-resource languages. We overview the studies in Section 2. Within the aforementioned arts, nowadays, Parent-Child (PC) transfer learning (Zoph et al., 2016) represents a considerable advance in our knowledge. It allows a Parent NMT model to be fully trained and developed over high-resource languages (e.g., that for De→En), and transfers it to low-resource languages (e.g., that for Mk→En) for fine-tuning. This contributes to the construction of a Child NMT model that inherits the translation experiences of Parent model.

The recent experimental results suggest that PC transfer learning suffers from the weak perception of semantics in low-resource languages, at the very beginning of encoding. In other words, fine-tuning Parent NMT model over low-resource languages is unavoidably started from scratch. This results in less effective and inefficient representation learning. Though, it is proven that duplicating embeddings (Aji et al., 2020; Xu and Hong, 2022) of cross-language shareable tokens, synonyms and mutually-aligned tokens helps to alleviate the cold-start fine-tuning problem. This also implies that conventional methods of constructing shareable latent space (Artetxe et al., 2018; Lample et al., 2018) may produce similar but general effects.

In this paper, we intend to strengthen PC trans-

*Corresponding author.

Proceedings of the 29th International Conference on Computational Linguistics, pages 4481–4491
October 12–17, 2022.

fer learning by coupling it with space-shareable encoding. Different from the previous work, we neither use an unabridged Parent encoder for space-shareable encoding, nor directly transfer it towards low-resource translation. Instead, we divide the unabridged Parent encoder into two parts. One part engages in space-shareable encoding for alleviating cold-start fine-tuning problems. The other is straight transferred without being "brainwashed" for the pre-existing translation experiences (i.e., the ones learned during high-resource NMT). The goal is to fulfill bidirectional adaptation, i.e., 1) establishing the encoding mode that adapts to linguistic characteristics of low-resource languages, conditioned on the shareable latent space; and 2) preserving the encoding mode that adapts to original translation experiences of the Parent NMT model.

In our experiments, we follow Vaswani et al. (2017) to build a transformer-based NMT model within the encoder-decoder architecture, where both encoder and decoder comprise 6 transformer layers. We intensively train it on large-scale high-resource language pairs to produce a knowledgeable Parent NMT model. On the basis, we take Parent's encoder, and divide it into two teams: T1 (1st transformer layer) and T2 (2nd-6th transformer layers). We train T1 to perform space-shareable encoding for low-resource languages. And we carry out monolingual unsupervised learning when training T1, where the generative adversarial network and masked language model are used. When moulding the Child NMT model, we transfer Parent to low-resource languages as usual. The difference lies in that T2 in Parent performs hot-start encoding by absorbing "home-made" hidden states from T1, i.e., the ones fabricated by T1 in terms of both monolingual features of low-resource languages and distributions in shareable latent space.

We conduct experiments on corpora SETIMES, where the low-resource MT scenarios of Sq-En, Mk-En, Hr-En and Ro-En are considered. Experimental results show that our method yields substantial improvements, and achieves competitive performance compared to the state of the art.

2 Related Work

There are a variety of advanced methodologies proposed for tackling low-resource NMT. Due to page limitation, we merely overview the closely-related arts reported in recent five years.

• **Shareable Latent Space**

Recently, the impressive hypothesis is that embeddings of both high-resource and low-resource source languages can be produced conditioned on the distributions in the same latent space. This undoubtedly contributes to the construction of versatile NMT models towards different language pairs, frankly, including those in the low-resource NMT scenarios. The key issue, in this case, is to establish a shareable latent space.

Artetxe et al. (2018) and Lample et al. (2018) design unsupervised learning approaches to construct shareable latent space. The approaches actually enable an encoder to properly project low-resource languages into the latent space of high-resource languages. Iterative back translation (Sennrich et al., 2016) and denoising auto-encoder (Vincent et al., 2008) are used to fulfill space-shareable encoding. The studies demonstrate the versatility of space-shareable encoding for multilingual translation in simulated experiments, where high-resource language pairs (e.g., En-De) are used though the size of training data is reduced. Soon after, Guzmán et al. (2019) prove that space-shareable encoding fails to obtain promising performance for authentic low-resource scenarios (e.g., En-Nepali and En-Sinhala). Marchisio et al. (2020) suggest that weak isomorphism between non-family languages results in the performance degradation. To strengthen space-shareable encoding, recently, multilingual BART (Liu et al., 2020a) is used in the unsupervised NMT framework, together with denoising autoencoder (Üstün et al., 2021) and multitask learning (Ko et al., 2021).

• **Constructing Shareable Vocabulary**

The first study for alleviating weak isomorphism most probably derives from Lakew et al. (2019)'s effort, where perplexity-based similarity computation is utilized to automatically select most relevant high-resource languages for space-shareable encoding. The obtained improvements in this study imply that common linguistic units (isomorphic constituents) between high-resource and low-resource languages serve as informative seeds for harvesting embeddings of heterogeneous constituents. It raises the interest in building shareable vocabulary.

Kim et al. (2018) construct a synthetic dictionary by iteratively updating linear mapping relationships between bilingual embeddings. Aji et al. (2020) build a joint vocabulary where the matched tokens are assigned with the same embeddings, while the mismatched the randomly-initialized em-

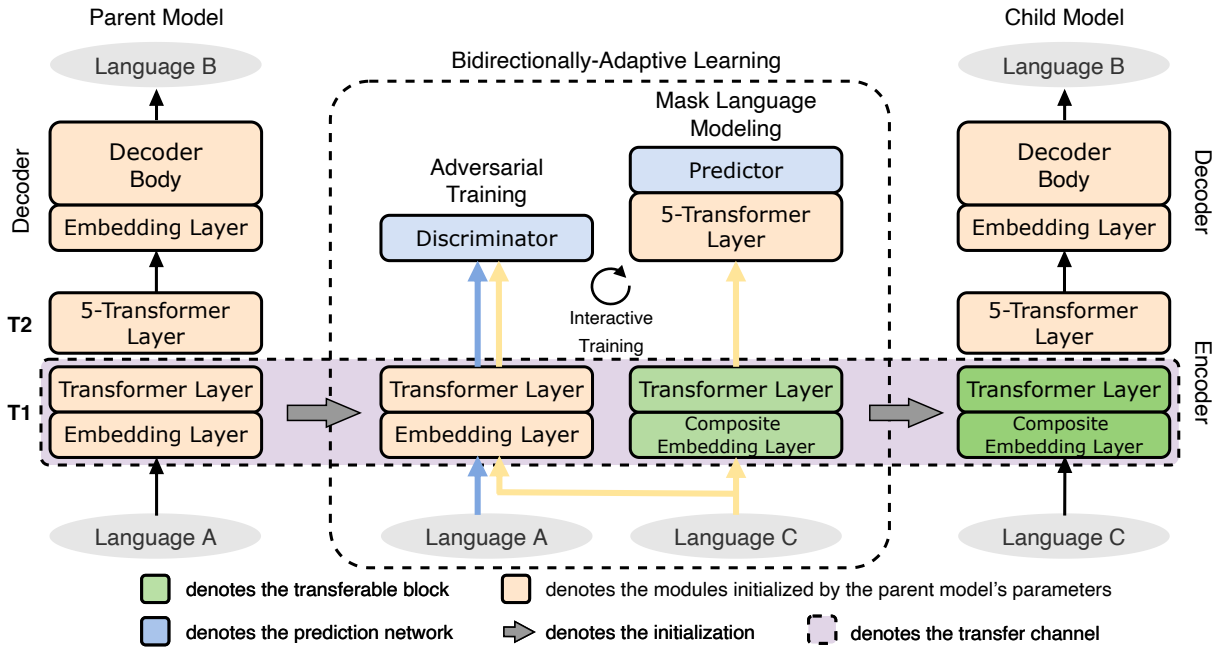

Figure 1: Architecture of bidirectionally-adaptive transfer learning for PC-based NMT, which conducts bilingual translation from high-resource language pairs (A→B) to low-resource language pairs (C→B).

beddings. Chronopoulou et al. (2021) retrain Byte-Pair-Encoding (BPE) over bilingual hybrid corpus, and use the segmented tokens by BPE to build the shareable vocabulary. Xu and Hong (2022) carry out word alignment between low-resource and high-resource languages, and share embeddings among aligned sub-tokens. This effectively expands the existing shareable vocabularies.

• **Transfer Learning for NMT**

Transfer learning approaches for NMT are primarily developed within Parent-Child (PC) framework (Zoph et al., 2016; Zhang et al., 2021). PC allows an NMT model to be trained on large-scale high-resource parallel data, and fine-tunes it on a small quantity of low-resource parallel data. It is proven that PC produces significant and increasing improvement with less warming-up time. Nowadays, PC has been successfully coupled with the aforementioned shareable vocabulary construction (Aji et al., 2020; Chronopoulou et al., 2021; Xu and Hong, 2022).

• **Pretrained Langauge Models for NMT**

Pretrained langauge models, such as ELMo (Peters et al., 2018) and BERT (Devlin et al., 2019), have been demonstrated to be effective for natural language processing. They enable the deep perception and encoding of semantics by learning that from large-scale monolingual data. Recently, Liu et al. (2020b) develop a multilingual BART, whose encoder significantly contributes to the enhancement of low-resource NMT.

3 Approach

We show the architecture of our low-resource NMT model in Figure 1. The 12-layer transformer-based encoder-decoder network (at the left side in Figure 1) serves as Parent NMT model, which contains six layers of transformer encoder and six layers of transformer decoder. From here on, we refer them as encoder and decoer layers respectively. The Parent NMT model has been intensively trained to perform translation for high-resource language pairs A and B (i.e., NMT for A→B).

We divide Parent encoder layers into two teams T1 and T2, where T1 is constituted merely with the 1st encoder layer, while T2 the rest five encoder layers. On the basis, T1 is pushed into the bidirectionally-adaptive learning channel (which is marked by the dotted rectangular box with a purple background in Figure 1). During the adaptation process, T1 engages in the bilingual encoding program launched by the generative adversarial network, so as to learn space-shareable encoding mode for both high-resource language A and low-resource language C. Besides, T1 engages in the masked language modeling program for low-resource language C, with the aim to learn language-specific encoding mode in terms of distinct characteristics of C. T1 will be trained iteratively and alternately

in the two programs. This results in a refined T1.

We construct Child NMT model by transferring Parent decoder and T2 to low-resource language pairs C and B (NMT for C→B), and connect T2 with the refined T1. On the basis, we fine-tune Child NMT model using small-scale low-resource parallel data. The Child model is deployed at the right side in Figure 1. Instead, the T1 and T2 we used come from the decoder of the Parent model (A→B) if the child translation direction is A→C. In the rest of this section, we detail all the components of our model.

3.1 Baseline Low-resource NMT

We follow Xu and Hong (2022)'s work to construct the baseline transferable NMT model for low-resource language pairs, where Parent-Child (PC) transfer mechanism (Zoph et al., 2016) is used, and Xu and Hong (2022)'s expanded version of Aji et al. (2020)'s joint vocabulary is adopted.

• **NMT Framework** We performance sentence-level NMT. Given a source language sentence s, we convert each token in it into the real-valued embedding $v_s^i \in \mathbb{R}^{512}$. This results in the distributed representations V_s of s (V_s=$\{v_s^1 ... v_s^l\}$), where "l" denotes the maximum length of input sequence. A trainable embedding layer is used for obtaining token-level embeddings, which possesses a dynamic **source-language** vocabulary mapping from tokens to embeddings.

We feed V_s into the 1st encoder layer, the one deployed ahead of other five sequentially-connected encoder layers. We use the encoder layers to obtain deep representations H_s of V_s as follows:

$$\begin{cases} H_s^{(i)} = f_e^{(i)}\left(H_s^{(i-1)}\right), & 1 < i \leq 6 \\ H_s^{(i)} = f_e^{(i)}\left(V_s\right), & i = 1 \end{cases} \quad (1)$$

where, $f_e^{(i)}$ is the i-th encoder layer of vanilla transformer (Vaswani et al., 2017; Al-Rfou et al., 2019).

Conditioned on the representations $H_s^{(6)}$ output by the encoder stack, we conduct decoding using six successively-connected decoder layers:

$$\begin{cases} h_t^{(i)} = f_d^{(i)}\left(H_s^{(6)}, h_t^{(i-1)}\right), & 1 < i \leq 6 \\ h_t^{(i)} = f_d^{(i)}\left(H_s^{(6)}, v_{t-1}\right), & i = 1 \end{cases} \quad (2)$$

where, $f_d^{(i)}$ denotes the i-th decoder layer of vanilla transformer, $h_t^{(i)}$ is the hidden state output by $f_d^{(i)}$ at a certain decoding time step t, and v_{t-1} is the embedding of the (t-1)-th token predicted at the earlier time step. Note that we obtain v_{t-1} using the trainable **target-language** embedding layer. Each target-language token is predicted by a linear layer with Softmax normalization, conditioned on $h_t^{(6)}$.

• **PC Transfer Learning** We train the aforementioned encoder-decoder network for NMT on high-resource language pairs A and B, i.e., learning to encode $H_A^{(6)}$ and decode $H_B^{(6)}$. This allows Parent NMT model to be formed. We transfer this well-trained Parent NMT model to low-resource language pairs C and B, and fine-tune it over the parallel data between C and B. By parametric inheritance and adaptive training (i.e., fine-tuning), transfer learning enables the generation of Child NMT model. Ideally, it learns to encode $H_C^{(6)}$ and decode $H_B^{(6)}$ to some extent.

• **Joint Vocabulary** During the transfer learning towards C→B NMT, the embedding layer of source language C is enhanced using the expanded joint vocabulary. In terms of the vocabulary, both morphologically-identical sub-tokens (Aji et al., 2020) and aligned sub-tokens (Xu and Hong, 2022) (between source language C and target language B) share the same embeddings, i.e., the ones learned from the process of training Parent NMT. When conducting bilingual embedding sharing between aligned sub-tokens, Xu and Hong (2022)'s element-wise mean aggregation (namely Mean-PC) is used for N-to-1 alignment cases.

3.2 Bidirectionally-adaptive Transfer Learning

We strengthen the baseline NMT model using a Bidirectionally-Adaptive Transfer Learning strategy (BATL for short). BATL adopts T1 (i.e., 1st encoder layer) of Parent NMT model, and exclusively trains it for activating its bidirectional adaptability, including the adaptation to Parent's encoding mode towards high-resource NMT (A→B), as well as that to monolingual linguistic characteristics of the low-resource language C. Generative Adversarial Network (GAN) and Masked Language Modeling (MLM) are utilized for BATL.

• **GAN-based Backward Adaptation** We construct a discriminator, and couple it with T1 for adversarial training. T1 encodes a source-language sentence, which may derive from high-resource language pairs or low-resource. Conditioned on the representation (of a sentence) output by T1, the discriminator determines whether the sentence is of high-resource language or low-resource, within

a binary classification task.

More importantly, T1 plays the role of a "counterfeiter". It produces the representation according as closely as possible to distributions in the semantic space of high-resource language, i.e., the one learned during the training for Parent NMT. Briefly, T1 counterfeits the high-resource sentence representation even if the sentence is actually of low-resource language. By contrast, the discriminator is trained to perform for anti-counterfeiting, determining the provenance of a sentence as precisely as possible. Repeatedly training T1 and the discriminator within the adversarial framework (counterfeiting versus anti-counterfeiting) will enhance both themselves. In particular, T1 learns to encode the low-resource language in the way of encoding the high-resource language, conditioned on a shareable semantic space. Coupling such a T1 into Child NMT model, frankly, contributes to the enhancement of its adaptation to Parent's translation experience, during the process of tackling the low-resource language.

• **MLM-based Forward Adaptation** T1 appears as a junior encoder when dealing with sentences of low-resource language at the very beginning, due to a lack of pragmatic and semantic knowledge in it. Consequently, the aforementioned adversarial training that directly utilizes such a junior T1, most probably, fails to form a reliable bilingual semantic space. In other words, although T1 learns to encode low-resource language in the mode of high-resource language (by GAN), it is grounded on a shallow or even inexact understanding of the former's pragmatics and semantics.

To address the issue, we construct a Masked Language Modeling channel (MLM) to enhance the capacity of T1 in encoding low-resource languages. It enables the forward adaptation of T1 to inherent linguistic characteristics of low-resource language.

MLM is conducted with the task of predicting masked tokens. Given a sentence of low-resource language, we mask about 10% tokens in it. The Masking strategy is implemented by substituting the randomly selected tokens with the special token "UNK", and initializing them with the unified embeddings. We feed the partially-masked sentence into T1 to encode each token in it, where the parameters of T1 are learnable during training. T2 is used for further encoding over the output of T1, where the parameters of T2 are frozen. Freezing T2 prompts T1 to learn low-resource languages as actively as possible.

• **Collaborative Training** We train T1 by GAN and MLM, alternatively and iteratively. First, T1 is trained by GAN, where 781 batches of hybrid monolingual data are used (i.e., 781 batches of monolingual sentences selected from the low-resource dataset, as well as 781 batches of high-resource cases). Secondly, T1 is further trained by MLM, where 1,562 batches of monolingual low-resource instances are used. This alternative training is carried out iteratively for 100 times within 25 epochs.

3.3 Shaping Low-resource NMT by BATL

We utilize the aforementioned BATL as a midway stage of transfer learning. In order to shape a concrete Child NMT, we still need to transfer T1 and the accompanying networks to low-resource language pairs, and fine-tune them on the parallel data.

• **Components of Child** comprises T1, T2 and the decoder of Parent NMT model. During assembling the components, T2 is connected behind T1, and both act as an encoder. The decoder of Parent with an embedding layer is coupled with the encoder. Briefly, Child inherits Parent's architecture.

• **Transfer Learning** includes the stages of BATL and transfer to low-resource language pairs. Within Child's components, only T1 is considered during BATL. All the components are fine-tuned on the parallel data of low-resource language pairs. It is noteworthy that although T2 is used for MLM in BATL, its parameters are frozen at the stage. During fine-tuning towards low-resource languages, all the components are trainable.

• **Loss of NMT** is calculated as follows, where the cross-entropy estimation is used:

$$\mathcal{L}_{MT} = \mathbb{E}_{x_i \in S}[-log\ p(y_i|x_i)] \tag{3}$$

where, $p(y_i|x_i)$ denotes the conditional probability that the ground-truth target-language sentence y_i is predicted given the source-language sentence x_i.

4 Experimentation

4.1 Datasets and Evaluation Metric

We experiment on SETIMES (Tiedemann, 2012)[1]. To facilitate the comparison with the previous work, we concentrate on the low resource translation tasks of Sq↔En, Mk↔En, Hr↔En and Ro↔En, where

[1]http://opus.nlpl.eu/SETIMES.php

#HighR	Fr-En	Es-En	De-En	Ru-En
Train.	747M	952M	582M	217M

Table 1: Statistics in high-resource (#HighR) parallel datasets. Note that we won't report the development and test results of Parent NMT models, and therefore the statistics for high-resource validation and test sets are omitted in this study. The high-resource NMT performance has been discussed in Tiedemann's work[3].

#Languages	Datasets	Train.
De, Sq, Mk, Hr, Ro	SETIMES	200k
Fr, Es, Ru	Europarl	200k

Table 2: Statistics in monolingual datasets.

Sq, Mk, Hr, Ro and En refer to Albanian, Macedonian, Croatian, Romanian and English, respectively. We utilize 200K sentences for training, 1K sentences for validation, and 3K sentences for testing for each language. We take into consideration various Parent NMT models for shaping Child in a series of separate experiments, where four classes of high-resource language pairs are used, including Es→En, Fr→En, De→En and Ru→En (Spanish: Es, French: Fr, German: De, Russian: Ru). All the high-resource parallel data is derived from Tatoeba[2]. Table 1 shows the scales of high-resource parallel training data.

In addition, we introduce different monolingual datasets into our experiments, which are used for GAN and MLM during the stage of BATL. The monolingual data of De, Sq, Mk, Hr and Ro is taken from the parallel data in SETIMES, while that of Fr, Es and Ru is selected from Europarl (Koehn, 2005)[4]. Table 2 shows the statistics in monolingual datasets.

We follow the previous work to evaluate all NMT models with SacreBLEU (Post, 2018).

4.2 Hyperparameter Settings

We directly use the off-the-shelf transformer-base NMT models (Tiedemann, 2020) as Parents, and the newly-developed Child NMT models inherit all the configurations and hyperparameters of Parent.

First of all, all the sentences are tokenized using SentencePiece (Kudo and Richardson, 2018) with a 100k vocabulary size.

Secondly, we use monolingual datasets to train T1. We train T1 by GAN and MLM in 25 epochs, using NVIDIA RTX 2080Ti 11GB GPU. The optimizer is set to Adam (Kingma and Ba, 2015), and the learning rate is set to 10^{-4}.

Finally, we shape a Child NMT model using the well-trained T1 as well as Parent's T2 and decoder. Fine-tuning Child NMT is conducted on the low-resource parallel data. During fine-tuning, HuggingFace Transformers library (Wolf et al., 2020) and AdamW (Loshchilov and Hutter, 2019) optimizer are used. The latter runs with a weight decay rate of 0.1. We carry out grid search in the learning rates of $\{10^{-4}, 5 \times 10^{-5}\}$ for each translation task, and adopt the best model occurred during the development process. All fine-tuning is conducted on NVIDIA RTX 3090 24GB GPU.

4.3 Models for Comparison

We compare with two baseline models, which are denoted as Baseline1 and Baseline2. Baseline1 acts as a 12-layer transformer-based encoder-decoder NMT. It is randomly initialized and trained on low-resource parallel data. Baseline2 is a variant of Baseline1 since it is enhanced by transfer learning within the Parent-Child (PC) framework.

Besides, we compare our model to different state-of-the-art NMT models, including:

• **XLM** (Conneau and Lample, 2019) is a transferable language model. It is obtained by cross-language pretraining, where parallel sentences are concatenated for joint encoding, within a masked language modeling process.

• **RE-LM** (Chronopoulou et al., 2020) learns to reuse language models across different monolingual datasets. An extended bilingual vocabulary is constructed to enhance cross-language pretraining. The obtained language models are transferred to low-resource NMT.

In addition, we involve Xu and Hong (2022)'s **Mean-PC** into the discussion, which recently improves low-resource NMT using shareable embeddings of aligned sub-tokens. Nevertheless, we fail to directly compare with it because its performance, as reported, is obtained on different corpora and source languages. We discuss Mean-PC in a separate ablation experiment, where it is reproduced and equipped with our BATL.

4.4 Main Result

We show the primary test results in Table 3. It can be observed that BATL produces substantial improvements compared to both baselines. It is note-

[2]https://opus.nlpl.eu/Tatoeba.php
[3]https://huggingface.co/Helsinki-NLP
[4]https://statmt.org/europarl/

Model	Sq-En		Mk-En		Hr-En		Ro-En	
	→	←	→	←	→	←	→	←
Baseline1	32.50	52.03	30.13	50.62	25.90	38.90	28.92	46.39
Baseline2 (Zoph et al., 2016)	38.15	54.50	33.32	54.57	29.93	41.34	32.92	48.71
XLM (Conneau and Lample, 2019)	60.90	55.10	55.00	55.50	-	-	-	-
RE-LM (Chronopoulou et al., 2020)	61.10	54.80	55.20	55.30	-	-	-	-
BATL (Ours)	**62.24**	**56.82**	**56.93**	**56.15**	**50.53**	**45.21**	**54.65**	**52.19**

Table 3: Low-resource NMT performance. BLEU scores (%) are reported in both translation directions (← and →) for each low-resource language pairs. The performance of previous work is quoted from the published literature (instead of reproduction) due to the use of the same test sets.

Model	Sq→En	Mk→En
Unabridged.	**62.24**	**56.93**
−GAN	61.95	56.45
−MLM	60.61	52.81
−Mean-PC	62.06	56.34
−BATL	61.45	56.42
−All	38.15	33.09

Table 4: Verifying the effectiveness of different components of our NMT model in ablation experiments.

worthy that we conduct transfer learning within the same framework with Baseline2, i.e., PC transfer. The additional components we use include Mean-PC (for embedding sharing among matched or aligned sub-tokens), as well as GAN and MLM (for bidirectional adaptation to low-resource and high-resource languages). This demonstrates that the considerable performance gains benefit from the collaboration between bilingual commonality perception and bidirectionally-adaptive encoding.

Compared to the state-of-the-art low-resource NMT models, our BATL-based models achieve better performance. The possible reasons behind the advantages may include the following aspects:

• The commonly-used cross-language pretraining in the previous work (XLM, RE-LM and the variant) is proceeded with a task-irrelevant scenario, where a knowledgeable multilingual pretrained model may be used for initialization, though it fails to learn the experience in MT. By contrast, we take out part of encoder of the well-trained Parent NMT, and train it to adapt different source languages during encoding. This allows the resultant representations of new languages to be compatible with the pre-existing translation mode, i.e., ensuring the task-specific cross-language training.

• MLM is used alone for cross-language pretraining in the previous work, where bilingual source sentences are concatenated, masked and encoded thereafter. This contributes to the encoding within a shareable latent space, though the exclusive linguistic characteristics of a specific source language are neglected to some extent. By contrast, we simultaneously pursue commonality and exclusive characteristics using MLM and GAN, where MLM intently encodes low-resource source languages in terms of their natures, while GAN is utilized to explore shareable encoding mode.

4.5 Ablation study

In a series of ablation experiments, we verify the effectiveness of different individual components of BATL, including GAN and MLM. Considering that we expand PC-based NMT using both Mean-PC (Xu and Hong, 2022) and BATL, we also ablate them alternatively to examine their influences. Table 4 shows the experimental results. Note that, hereafter, we merely report the forward NMT performance (i.e., that of "→" NMT), where low-resource languages (Sq and Mk) are considered as source languages.

It can be found that ablating MLM results in more significant performance degradation, compared to GAN. This implies that MLM plays a dominant role in BATL, or in other words, the adaptation to low-resource languages during the "preheated" transfer process is crucial. Note that we merely push part of encoder (i.e., T1) into MLM-based cross-language transfer learning. It is different from the previous work which uses the whole encoder. In fact, by comparing the performance of XLM and RE-LM to our model that ablates GAN (i.e., mere use of MLM) across Table 3 and 4, we can find that our local transfer strategy produces the positive effects (better performance is obtained even if GAN is disabled).

Compare to Mean-PC, ablating BATL (i.e., ablating both GAN and MLM) causes more substantial

Parent Model	Sq→En	Mk→En
De→En	61.34	55.28
Fr→En	62.18	56.24
Es→En	**62.24**	**56.93**
Ru→En	61.03	55.86

Table 5: BLEU scores (%) of Child NMT models transferred from different Parent models.

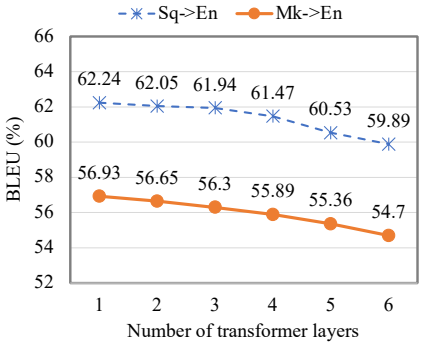

Figure 2: BLEU scores of our approach with respect to the number of self-attention layers in T1.

performance reduction for the translation scenario of Sq→En, while relatively comparable reduction for Mk→En. This illustrates that learning-centered strategy of BATL has an advantage over the knowledge sharing mechanism of Mean-PC. Frankly, both are non-negligible. It is proven by the severe performance degradation caused by disabling both BATL and Mean-PC (see the performance obtained when "ALL" is ablated in Table 4).

4.6 Discussion and Analysis

• Effects of Different Parent Models

We construct four Parent NMT models using different source languages, including De, Fr, Es and Ru. On the basis, we verify the effects of such Parents on Child NMT models. The low-resource NMT performance resulted from different Parents is shown in Table 5. It can be observed that the Parents of Fr→En and Es→En are more beneficial to transfer, helping to produce higher BLEU scores.

Ideally, the PC-based transfer learning ought to benefit from high-resource languages that derive from the consistent or similar language family, such as the relatively closer relationship between Ru and Mk. However, the experimental results fail to support this hypothesis. Our findings show that the size of high-resource training data plays a more crucial role in improving the performance of PC transfer. As shown in Table 1, the most knowledgeable Parent, i.e., that of Es→En, is obtained on 952M training data. The scale of training data is much larger than that used for Parent of Ru→En. The latter fails to obtain an equivalently strong Child model. These findings are consistent with the conclusion of Kocmi and Bojar (2018).

• Is it Necessary to Construct Multi-layer T1

Our BATL is performed merely using T1, i.e., a single encoder layer, while the considered encoder is actually constituted with 6 transformer layers. It may be questioned whether BATL induces varying effects when T1 is expanded with more layers. The following experiment demonstrates that a larger T1 negatively influences NMT performance.

In a separate experiment, we split different numbers of transformer layers from Parent's encoder, and use them to form different depths of T1s. There are six T1s constructed in total using 1 to 6 encoder layers respectively. We conduct BATL for each of them separately, and reform Child NMT models accordingly. Figure 3 shows performance curves of such models in the MT scenarios of Sq↔En and Mk↔En, where the horizontal axis indicates the number of encoder layers in T1. It can be observed that performance degrades gradually when T1 is enlarged using more encoder layers.

The experimental results imply that overly shuffling and remodeling Parent's encoder for adaptation enhancement is risky. Most of well-trained parameters (translation experience) of Parent need to be directly inherited by Child.

• Compatibility with DAE and BT

Both denoising autoencoder (DAE) and back-translation (BT) have been proven effective in low-resource NMT. They were known as data augmentation methods that build synthetic corpora using monolingual data (Artetxe et al., 2018; Lample et al., 2018). We attempt to combine DAE and BT with our BATL, and rerun all the experiments to verify whether compatibility can be achieved.

For verifying the compatibility with DAE, we combine it with BATL from behind. During training, BATL is first used to optimize T1 and then DAE runs. Within the process, there are 3M[5] monolingual sentences (of low-resource language) used for DAE, where the model is additionally trained to assist revivification of all the falsified sentences, with the role of autoencoder. The effect of combining BATL and DAE is negative, as shown in Table 6, where the low-resource NMT performance is reduced severely. Our analysis suggests that, worse

Model	Sq→En	Mk→En
Ours.	62.24	56.93
+DAE	60.97	55.76
+BT	**62.75**	**57.32**

Table 6: BLEU scores obtained in the study of compatibility, where our BATL is respectively combined with DAE and BT.

than incompatibility and redundancy, the additional utilization of DAE leads to catastrophic forgetting of translation experience. In fact, DAE nearly plays the same role as the MLM module of our BATL, and thus, the posteriori-prompted DAE breaks the compromise effects reached by MLM and GAN.

For verifying the compatibility with BT, we expand the training data of low-resource language pairs using parallel instances. As usual, such pseudo instances are obtained by translating En to a certain low-resource language (e.g., Sq or Mk) forwardly and backwardly, where an off-the-shelf NMT model[6] is used. In our experiments, there are 2M pseudo instances created by BT for expansion. We use the expanded low-source data to fine-tune the Child NMT model. The rest configuration remains unchanged. Combining BATL and BT yields additional performance gains, as shown in Table 6, where BLEU scores are up to 62.75% and 57.32% for Sq→En and Mk→En. It demonstrates that BATL can be jointly used with BT safely.

5 Conclusion

We propose a Bidirectionally-Adaptive Transfer Learning (BATL) approach to enhance low-resource NMT models. Experimental results show that our approach yields substantial improvements, compared to the state of the art. In addition, it is demonstrated that BATL is compatible with BT-based data augmentation. Combining BATL and BT obtains additional performance gains. In a series of auxiliary experiments, we analyze the effects of various Parent NMT models and multi-layer BATL on low-resource NMT, some of which are negative and therefore noteworthy for risks in real applications.

The Commonality of linguistics stands for the fundamental principle in prompting Parent-Child transfer for low-resource NMT. Different family languages hold inconsistent commonalities with a specific low-resource language. Considering this phenomenon, in the future, we will study on a multilingual Parent-Child transfer learning. A selective transfer will be developed, in terms of case-specific adhesion to different high-resource family languages. The adhesion will be perceived by modeling the relevance of topics, provenances and domains, as well as document-level structure information (e.g., monolingual discourse relationships and rhetorics).

[5]We follow the previous work to adopt a similar number of instance for running DAE (Chronopoulou et al., 2020).

[6]https://huggingface.co/Helsinki-NLP

Acknowledgements

The research is supported by National Key R&D Program of China (2020YFB1313601) and National Science Foundation of China (62076174, 62076175).

References

Alham Fikri Aji, Nikolay Bogoychev, Kenneth Heafield, and Rico Sennrich. 2020. In neural machine translation, what does transfer learning transfer? In *Proceedings of ACL*, pages 7701–7710, Online. Association for Computational Linguistics.

Rami Al-Rfou, Dokook Choe, Noah Constant, Mandy Guo, and Llion Jones. 2019. Character-level language modeling with deeper self-attention. In *Proceedings of AAAI, Honolulu, Hawaii, USA*, pages 3159–3166. AAAI Press.

Mikel Artetxe, Gorka Labaka, Eneko Agirre, and Kyunghyun Cho. 2018. Unsupervised neural machine translation. In *Proceedings of ICLR, Vancouver, BC, Canada*.

Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. 2015. Neural machine translation by jointly learning to align and translate. In *Proceedings of ICLR, San Diego, CA, USA*.

Alexandra Chronopoulou, Dario Stojanovski, and Alexander Fraser. 2021. Improving the lexical ability of pretrained language models for unsupervised neural machine translation. In *Proceedings of NAACL*, pages 173–180. Association for Computational Linguistics.

Alexandra Chronopoulou, Dario Stojanovski, and Alexander M. Fraser. 2020. Reusing a pretrained language model on languages with limited corpora for unsupervised NMT. In *Proceedings of EMNLP*, pages 2703–2711. Association for Computational Linguistics.

Alexis Conneau and Guillaume Lample. 2019. Cross-lingual language model pretraining. In *Proceedings of NIPS*, pages 7057–7067.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of NAACL*, pages 4171–4186, Minneapolis, Minnesota. Association for Computational Linguistics.

Francisco Guzmán, Peng-Jen Chen, Myle Ott, Juan Pino, Guillaume Lample, Philipp Koehn, Vishrav Chaudhary, and Marc'Aurelio Ranzato. 2019. The FLORES evaluation datasets for low-resource machine translation: Nepali–English and Sinhala–English. In *Proceedings of EMNLP*, pages 6098–6111, Hong Kong, China. Association for Computational Linguistics.

Yunsu Kim, Jiahui Geng, and Hermann Ney. 2018. Improving unsupervised word-by-word translation with language model and denoising autoencoder. In *Proceedings of EMNLP, Brussels, Belgium*, pages 862–868. Association for Computational Linguistics.

Diederik P. Kingma and Jimmy Ba. 2015. Adam: A method for stochastic optimization. In *Proceedings of ICLR, San Diego, CA, USA*.

Wei-Jen Ko, Ahmed El-Kishky, Adithya Renduchintala, Vishrav Chaudhary, Naman Goyal, Francisco Guzmán, Pascale Fung, Philipp Koehn, and Mona T. Diab. 2021. Adapting high-resource NMT models to translate low-resource related languages without parallel data. In *Proceedings of ACL*, pages 802–812. Association for Computational Linguistics.

Tom Kocmi and Ondřej Bojar. 2018. Trivial transfer learning for low-resource neural machine translation. In *Proceedings of WMT*, pages 244–252.

Philipp Koehn. 2005. Europarl: A parallel corpus for statistical machine translation. In *Proceedings of MTSummit, Phuket, Thailand*, pages 79–86.

Taku Kudo and John Richardson. 2018. SentencePiece: A simple and language independent subword tokenizer and detokenizer for neural text processing. In *Proceedings of EMNLP*, pages 66–71, Brussels, Belgium. Association for Computational Linguistics.

Surafel M. Lakew, Alina Karakanta, Marcello Federico, Matteo Negri, and Marco Turchi. 2019. Adapting multilingual neural machine translation to unseen languages. In *Proceedings of IWSLT*, Hong Kong. Association for Computational Linguistics.

Guillaume Lample, Alexis Conneau, Ludovic Denoyer, and Marc'Aurelio Ranzato. 2018. Unsupervised machine translation using monolingual corpora only. In *Proceedings of ICLR 2018, Vancouver, BC, Canada*.

Yinhan Liu, Jiatao Gu, Naman Goyal, Xian Li, Sergey Edunov, Marjan Ghazvininejad, Mike Lewis, and Luke Zettlemoyer. 2020a. Multilingual denoising pre-training for neural machine translation. *Transactions of the Association for Computational Linguistics*, 8:726–742.

Yinhan Liu, Jiatao Gu, Naman Goyal, Xian Li, Sergey Edunov, Marjan Ghazvininejad, Mike Lewis, and Luke Zettlemoyer. 2020b. Multilingual denoising pre-training for neural machine translation. *Transactions of the Association for Computational Linguistics*, 8:726–742.

Ilya Loshchilov and Frank Hutter. 2019. Decoupled weight decay regularization. In *Proceedings of ICLR, New Orleans, LA, USA*.

Kelly Marchisio, Kevin Duh, and Philipp Koehn. 2020. When does unsupervised machine translation work? In *Proceedings of WMT*, pages 571–583, Online. Association for Computational Linguistics.

Matthew E. Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer. 2018. Deep contextualized word representations. In *Proceedings of NAACL*, pages 2227–2237, New Orleans, Louisiana. Association for Computational Linguistics.

Matt Post. 2018. A call for clarity in reporting BLEU scores. In *Proceedings of WMT*, pages 186–191, Brussels, Belgium. Association for Computational Linguistics.

Rico Sennrich, Barry Haddow, and Alexandra Birch. 2016. Improving neural machine translation models with monolingual data. In *Proceedings of ACL 2016*. The Association for Computer Linguistics.

Jörg Tiedemann. 2012. Parallel data, tools and interfaces in OPUS. In *Proceedings of LREC*, pages 2214–2218. European Language Resources Association.

Jörg Tiedemann. 2020. The Tatoeba Translation Challenge – Realistic data sets for low resource and multilingual MT. In *Proceedings of WMT*, pages 1174–1182, Online. Association for Computational Linguistics.

Ahmet Üstün, Alexandre Berard, Laurent Besacier, and Matthias Gallé. 2021. Multilingual unsupervised neural machine translation with denoising adapters. In *Proceedings of EMNLP*, pages 6650–6662, Punta Cana, Dominican Republic. Association for Computational Linguistics.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Proceedings of NIPS, Long Beach, CA, USA*, pages 5998–6008.

Pascal Vincent, Hugo Larochelle, Yoshua Bengio, and Pierre-Antoine Manzagol. 2008. Extracting and composing robust features with denoising autoencoders. In *Proceedings of ICML 2008*, volume 307 of *ACM International Conference Proceeding Series*, pages 1096–1103. ACM.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Remi Louf, Morgan Funtowicz,

Joe Davison, Sam Shleifer, Patrick von Platen, Clara Ma, Yacine Jernite, Julien Plu, Canwen Xu, Teven Le Scao, Sylvain Gugger, Mariama Drame, Quentin Lhoest, and Alexander Rush. 2020. Transformers: State-of-the-art natural language processing. In *Proceedings of EMNLP*, pages 38–45, Online. Association for Computational Linguistics.

Minhan Xu and Yu Hong. 2022. Sub-word alignment is still useful: A vest-pocket method for enhancing low-resource machine translation. In *Proceedings of ACL*, pages 613–619, Dublin, Ireland. Association for Computational Linguistics.

Meng Zhang, Liangyou Li, and Qun Liu. 2021. Two parents, one child: Dual transfer for low-resource neural machine translation. In *Findings of ACL, Online*, pages 2726–2738. Association for Computational Linguistics.

Barret Zoph, Deniz Yuret, Jonathan May, and Kevin Knight. 2016. Transfer learning for low-resource neural machine translation. In *Proceedings of EMNLP*, pages 1568–1575, Austin, Texas. Association for Computational Linguistics.

HCLD: A Hierarchical Framework for Zero-shot Cross-lingual Dialogue System

Zhanyu Ma[1,2,3] **Jian Ye**[1,2,3*] **Xurui Yang**[1,2,3] **Jianfeng Liu**[1,2,3]
Institute of Computing Technology, Chinese Academy of Sciences, Beijing, China[1]
University of Chinese Academy of Sciences[2]
Beijing Key Laboratory of Mobile Computing and Pervasive Device[3]
{mazhanyu21s,jye,yangxurui20g,liujianfeng18s}@ict.ac.cn

Abstract

Recently, many task-oriented dialogue systems need to serve users in different languages. However, it is time-consuming to collect enough data of each language for training. Thus, zero-shot adaptation of cross-lingual task-oriented dialog systems has been studied. Most of existing methods consider the word-level alignments to conduct two main tasks for task-oriented dialogue system, i.e., intent detection and slot filling, and they rarely explore the dependency relations among these two tasks. In this paper, we propose a hierarchical framework to classify the pre-defined intents in the high-level and fulfill slot filling under the guidance of intent in the low-level. Particularly, we incorporate sentence-level alignment among different languages to enhance the performance of intent detection. The extensive experiments report that our proposed method achieves the SOTA performance on the public task-oriented dialog dataset.

1 Introduction

Natural language understanding (NLU) plays a significant role in task-oriented dialogue systems, which is aimed to parse dialog utterances by identifying user's intent and the arguments of the intent (Hou et al., 2021; van der Goot et al., 2021). These two tasks are known as intent detection and slot filling respectively. For instance, in the utterance "Next week's forecast" as shown in Table 1, the user intent is to query about the weather circumstances and the time argument of the query is "next week". Recently, many neural models have been proposed to jointly train these two tasks by considering the intent detection and the slot filling as sentence classification and sequence labeling task (Krishnan et al., 2021; Cao et al., 2020).

However, most works focus on monolingual datasets which are expensive to build. Furthermore, some dialogue systems, e.g., Google Home and Apple Siri, need to serve numerous users around the world in different languages and might be faced with the scarcity of dialogue data in certain languages. Thus, it's motivated to build cross-lingual dialogue systems that enable zero-shot adaptation from a high-resource language to a low-resource language without any training data in the target language. Specifically, some models (Upadhyay et al., 2018; Chen et al., 2018; Schuster et al., 2019) used cross-lingual pretrained embeddings to bridge different languages. Some approaches like Liu et al. (2019) adopted a small vocabulary of translated word-pairs to align cross-lingual embeddings by bilingual projection and alleviate the inherent discrepancies among different languages. Many existing methods (Liu et al., 2020; Qin et al., 2020) utilized pretrained cross-lingual language model (PXLM), e.g., XLM (Conneau and Lample, 2019), mBERT (Pires et al., 2019) and XLM-R (Conneau et al., 2020), to derive contextual embeddings of words in different languages.

Sentence	Next	week's	forecast
Slots	B-datetime	I-datetime	B-weather/noun
Intents	weather/find		

Table 1: An example of a sentence with slots and intents annotations from the zero-shot NLU dataset.

Nevertheless, most of existing methods consider embedding alignments among source and target languages at the word-level to fulfill the cross-lingual adaptation, but they ignore that intent detection is a sentence classification task (Liu and Lane, 2016; Liu et al., 2021), which requires higher level alignments among different languages. Furthermore, these two tasks are closely related and the slots highly depend on the intent (Goo et al., 2018; Qin et al., 2019), but their deep relations are not fully discovered by existing works.

In light of these observations, we propose a model named *HCLD* (A Hierarchical Framework

* Corresponding author.

Proceedings of the 29th International Conference on Computational Linguistics, pages 4492–4498
October 12–17, 2022.

for Zero-shot Cross-lingual Dialogue System) to alleviate these defects. Our approach is built upon a hierarchical framework to jointly accomplish intent detection and slot filling. It learns to classify pre-defined intents in the high-level of our model and fill the semantic slots under the guidance of the predicted intent in the low-level, which can help to find more related arguments of the intent to enhance the performance of slot filling. Particularly, we also adopt a pretrained language agnostic BiLSTM encoder *LASER* (Artetxe and Schwenk, 2019) to derive the sentence embeddings and directly use them for later classification task of intent detection, where the embeddings of sentences in different languages are aligned in the sentence-level. We conduct experiments on a public task-oriented dialogue dataset (Schuster et al., 2019). The results show that our proposed method achieves state-of-the-art performance on zero-shot adaptation.

2 Related Work

Zero-shot Cross-lingual Transfer The mainstream methods of zero-shot focus to learning cross-lingual embeddings. Recently, some contextual approaches (Pires et al., 2019; Conneau and Lample, 2019) built upon masked language model encourage to narrow the distance of representation in source and target language space. Apart from word-level alignments, another work (Artetxe and Schwenk, 2019) focuses on learning cross-lingual sentence embeddings to align sentences representations in different languages. MTOP (Li et al., 2021) further expend it with distant supervision in zero-shot setting for flat representations with the masked source utterance and the translated utterance as the concatenated input.

Code-switching is used as data augmentation in an alternative data alignment method known as CoSDA (Qin et al., 2020). To make model training highly multilingual, random words in the input are translated and substituted, leading to increased cross-lingual transfer ability. There were additional attempts to learn how to code-switch automatically (Liu et al., 2020).

Our work combines the cross-lingual sentence embedding and word embedding to handle intent detection and slot filling respectively.

Intent detection and slot filling Intent detection and slot filling are key tasks in the natural language understanding (NLU) of task-oriented dialogue systems. Recently, some models (Qin et al., 2021b; M'hamdi et al., 2021) consider to implement intent detection and slot filling jointly. The recent work (Zhang and Wang, 2016) first proposed a joint model to learn the correlation between intent and slots by RNN. Co-gat (Qin et al., 2021a) investigated a non-autoregressive model for joint multiple intent detection and slot filling.

Recently, information of intent has been incorporated for slot filling. The prior work (Goo et al., 2018) utilizes a slot-gated mechanism to model the relationship between two tasks. Some approaches like Wang et al. (2018) propose the bi-model to consider the cross-impact between the intent and slots based on ATIS datasets (Goo et al., 2018). Some models (Qin et al., 2019; Louvan and Magnini, 2020) predict the intent based on each word and then feed into slot filling as input.

Compared with previous works, our approach firstly predicts the intent given by multi-lingual sentence representation, and directly incorporates the intent information of sentence for slot filling. Furthermore, our model is handling simultaneously intent detection and slot filling at the cross-lingual setting.

3 Our Approach

3.1 Problem Formulation

Intent detection and slot filling are two key tasks of NLU. Given a utterance of L words $u = [w_1, w_2, \ldots, w_L]$ and a set of pre-defined intent types I and slots types S, the intent detection is aimed to predict the intent $o^I \in I$ based on the utterance u. While the slot filling is a sequence tagging problem, mapping the word sequence $[w_1, w_2, \ldots, w_L]$ into semantic slots $[s_1, s_2, \ldots, s_L]$ where $s \in S$.

3.2 HCLD

As shown in Fig. 1, our proposed hierarchical model HCLD firstly classifies pre-defined intent in the high-level architecture and then fulfill the semantic slots under the guidance of intent in the low-level. For intent detection, multi-lingual sentence embedding feeds Sentence U into a sentence representation H_u. Then a linear layer would predict an intent i based on H_u. For slot filling, word sequence [1] of U, $u = w_1, w_2, \ldots, w_L$ are passed

[1] https://huggingface.co/docs/transformers/v4.21.3/en/model_doc/xlm#transformers.XLMTokenizer

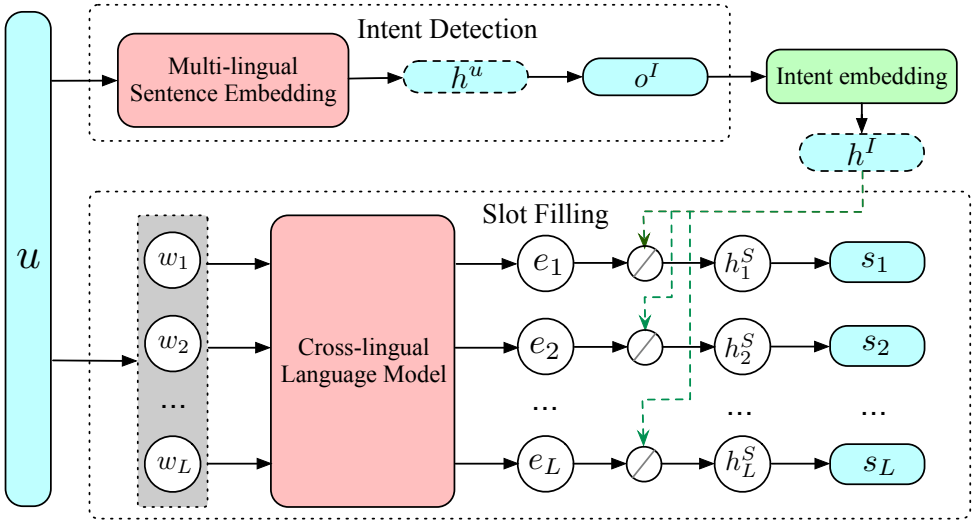

Figure 1: The framework of HCLD.

into multi-lingual language model [2] [3] and obtain contextual embeddings $e_1, e_2, \ldots, e_{L-1}, e_L$. Next, H_s^j, $i \in \{1, \ldots, L\}$ would computed by average the representations of e_j and H_i. Where H_i is obtained by intent i looking up the trainable intent embedding. Finally, the H_s^j would be used to predict the corresponding slot s_j.

In the high level, we first adopt LASER (Artetxe and Schwenk, 2019) [4] to derive sentence representations h^u. Based on multilingual neural machine translation tasks, LASER can produce sentence embeddings for multiple languages where embeddings of sentences with close meanings but in different languages can be aligned in a similar semantic space. Such sentence-level alignments can better boost the intent detection, which is a sentence classification task (Liu and Lane, 2016). Then the intent o^I is predicted by

$$y^I = Softmax(W_h^I h^u + b_h^I), \quad (1)$$

$$o^I = Argmax(y^I), \quad (2)$$

where y^I is the intent distribution and W_h^I, b_h^I are trainable parameters. In the low-level, we utilize mBERT (Pires et al., 2019), the pre-trained multilingual language model (PXLM), to produce contextual word embeddings in different languages $[e_1, e_2, \ldots, e_L]$ where word embeddings are aligned in the same semantic space.

However, PXLM suffers inconsistent contextualized representations of subwords across different languages (Qin et al., 2020). Thus, we follow CoSDA-ML (Qin et al., 2020) [5] to better align subword representations with data augmentation during the fine-tuning process of mBERT.

Furthermore, we adopt the intent information to guide the slot filling task by averaging each word embedding e_j with the corresponding intent embedding h^I as new word representations $h_j^S = \text{Average}(e_j, h^I)$, and help to fill the semantic slot $[s_1, s_2, \ldots, s_L]$. Here h_j^S is a combined representation with a word embedding and the intent embedding, and the slot would be predicted based on the h_j^S. Thus, the slot distribution for each word can be predicted as y_j^S. Recall that our proposed approach would guide the slot filling task with the information of the intent, we add a trainable intent embedding initialized with randomly parameters. A predicted intent o^I would look up the intent embedding and then receive an intent representation H^I. Next, H^I would influence the process of slot filling. The process of intent detection in the high-level can be formulated as follows,

$$H_u = LASER(U) \quad (3)$$

$$I_u = Softmax(Linear(H_u)) \quad (4)$$

Where I_u is a vector that tries to project the sentences in any language into a high-dimensional space with the goal that the same statement in any language will end up in the same neighborhood.

$$y_j^S = Softmax(W_h^S h_j^S + b_h^S), j \in \{1, \ldots, L\} \quad (5)$$

To jointly learn both tasks, the objective function

[2] https://huggingface.co/bert-base-multilingual-cased
[3] https://github.com/facebookresearch/XLM/
[4] https://github.com/facebookresearch/LASER
[5] https://github.com/kodenii/CoSDA-ML

$\mathcal{L}$ is formulated as:

$$\mathcal{L} = \mathcal{L}_{\mathcal{I}} + \mathcal{L}_{\mathcal{S}} = -\sum_{i=1}^{n^I} \hat{y}_i^I log(y_i^I) - \sum_{j=1}^{L}\sum_{i=1}^{n^S} \hat{y}_{j,i}^S log(y_{j,i}^S) \quad (6)$$

where $\mathcal{L}_{\mathcal{I}}$, $\mathcal{L}_{\mathcal{S}}$ stand for the loss function of intent detection and slot filling respectively. n^I is the number of intent types and y_i^I is the gold intent label. While in $\mathcal{L}_{\mathcal{S}}$, L is the sequence length of a sentence, n^S means the number of slot types as well as y_j^S is the gold slot label. To avoid error prorogation, we adopt the gold intent to replace the predicted intent o^I during the training period.

3.3 Dataset

We conduct our experiments on the cross-lingual task-oriented dialogue dataset (Schuster et al., 2019) which contains English, Spanish and Thai. We train and validate our model on the English dialog data with $30,521$ and $4,181$ sentences respectively, and test on $3,043$ and $1,692$ sentences in Spanish and Thai for zero-shot adaptation following Liu et al. (2019).

3.4 Implementable Details

We introduce several competitive baselines, including: Zero-shot SLU (Upadhyay et al., 2018), BiLSTEM with CRF (Liu et al., 2019) [6] (LVM), XLM (Conneau and Lample, 2019), mBERT (Pires et al., 2019), Attention-Informed Mixed-Language Training (MLT) (Liu et al., 2020) [7] and CoSDA-ML (Qin et al., 2020). We adopt classification accuracy (Acc.) to evaluate the performance of intent detection while using F1 score to measure the performance of slot fillings.

In our experiments, we use WordPiece embeddings with a vocabulary containing 110k tokens following Devlin et al. (2019). We adopt LASER to generate sentence embeddings whose dimension is 1024 while taking the base case mBERT to derive word embeddings with dimension of 768.

Notice that we take the first subword embedding as word-level representation following Qin et al. (2020) while incorporating mBERT for slot filling. We also set the size of intent embedding to 768, then train our model for 10 epochs with a batch size of 32 and a learning rate 0.001. We adopt AdamW (Loshchilov and Hutter, 2018) to optimize our HCLD and select the hyper-parameters by grid search. Besides, we adopt the gold intent instead of the predicted intent o^I in equation 2 to guide the slot filling during the early stages of training period to avoid error prorogation.

[6] https://github.com/zliucr/Crosslingual-NLU

[7] https://github.com/zliucr/mixed-language-training

4 Experiments

4.1 Overall Performance

We can make several observations from the results demonstrated in Table 2. Firstly, these methods with CoSDA-ML (*mBERT+COSDA-ML*) achieve the best performance among all the baselines, which indicates that pre-trained language models (e.g., mBERT) with data augmentation can produce better contextual word embeddings for different languages than those without data augmentation or only with simple word alignments techniques (e.g., BiLSTM with CRF/LVM).

Specifically, HCLD outperforms the second-best model on intent detection by 3.4 on Spanish, 1.5 on Thai. We conjecture that the alignments on sentence embeddings can enhance the adaption on different languages, leading to the significant improvements on intent detection. In addition, the performance of our model on slot filling also exceeds all the compared methods. We think that it can be attributed to the hierarchical architecture that fulfill the slots under the guidance of intent information.

4.2 Ablation Study

To investigate the effects of individual component, we conduct an ablation study and report the results in Table 2. Firstly, we remove the data augmentation method CoSDA-ML from our model to testify its effectiveness, it drops 11.2 and 16.3 on Spanish and Thai on slot filling respectively. Secondly, we remove the hierarchical architecture from HCLD and find that it would perform worse without the guidance from intent information. It is probably because intent can provide related information and help to find more accurate semantic slots. Thirdly, we investigate the importance of sentence alignments by replacing LASER with two pre-trained language models, i.e., mBERT or XLM, to derive the sentence embeddings. It validates the effectiveness of LASER that the performance of models with either mBERT or XLM is inferior to that with LASER which can produce aligned sentence representations in different languages. In addition, we also find that mBERT can produce better token-level embeddings for slot filling than XLM, thus

Methods	Spanish		Thai	
	Intent Acc.	Slot F1	Intent Acc.	Slot F1
Zero-shot SLU (Upadhyay et al., 2018)	46.6	15.4	35.6	12.1
XLM (Conneau and Lample, 2019)	69.4	40.0	49.3	13.3
mBERT (Pires et al., 2019)	73.7	51.7	28.2	10.6
BiLSTM with CRF (Liu et al., 2019)	88.8	44.0	64.5	17.5
BiLSTM with LVM	89.2	64.0	70.8	29.5
XLM + MLT (Liu et al., 2020)	87.5	68.6	72.6	28.0
mBERT + MLT	87.9	74.9	73.5	27.1
XLM + CoSDA-ML (Qin et al., 2020)	90.3	69.0	*86.7*	34.9
mBERT +CoSDA-ML	*94.8*	*80.4*	76.8	*37.3*
HCLD	**98.2**	**83.2**	**88.2**	**38.0**
w/o CoSDA-ML	96.8	72.0	84.2	21.7
w/o Hierarchical	97.8	77.9	87.9	31.8
w/o LASER, w/ mBERT	84.7	79.5	81.0	32.2
w/o LASER, w/ XLM	93.0	69.4	86.3	34.4
w/o mBERT, w/ XLM	96.6	72.7	88.1	36.5

Table 2: Results of zero-shot test set are compared with different baselines as well as the result of ablation study. The most competitive results from baselines are annotated with underline.

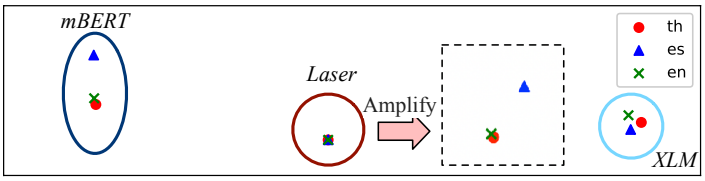

Figure 2: The visualization of sentence representations generated by LASER, XLM and mBERT. In the visulation, we select three utterances with the same meaning but in different languages, including Thai (th), Spanish (es) and English (en). In order to clearly demonstrate the points of LASER, we magnify these nodes with 10 times.

we derive word-embedding with mBERT in HCLD.

4.3 Visualization

To provide a more straightforward viewpoint to examine the sentence-level alignments, we visualize the sentence representations generated by LASER, mBERT and XLM by projecting them into a 2-dimension space. We select three sentences in English, Spanish and Thai with close meanings, which is an example from Liu et al. (2019). According to Fig. 2, we can find the three points from LASER are concentrated and even look like a point and while points from mBERT and XLM are more sparse. It validates that we can use LASER to derive aligned sentence embeddings and it has positive correlations with the results of intent detection as shown in Table 2.

5 Conclusion

In this paper, we propose a hierarchical framework to classify the pre-defined intents in the high-level and fulfill slot filling under the guidance of intent in the low-level. Particularly, we adopt sentence-level alignments to improve the performance of intent detection, and further enhance the performance of slot filling. The experiments conducted on the public dataset demonstrate the effectiveness of our proposed method and our model achieves state-of-the-art performance in the zero-shot cross-lingual scenario.

6 Acknowledgement

The research work is supported by National Key R&D Program of China (No.2017YFB1302400), Key Research and Development Program of in Shandong Province (2019JZZY020102), Industrial Internet Innovation and Development Project in 2021 (TC210A02M, TC210804D), Key Research and Development Program of Jiangsu Province (No.BE2018084), Opening Project of Beijing Key Laboratory of Mobile Computing and Pervasive Device.

References

Mikel Artetxe and Holger Schwenk. 2019. Massively multilingual sentence embeddings for zero-shot cross-lingual transfer and beyond. *Transactions of the Association for Computational Linguistics*, 7:597–610.

Xu Cao, Deyi Xiong, Chongyang Shi, Chao Wang, Yao Meng, and Changjian Hu. 2020. Balanced joint adversarial training for robust intent detection and slot filling. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 4926–4936.

Wenhu Chen, Jianshu Chen, Yu Su, Xin Wang, Dong Yu, Xifeng Yan, and William Yang Wang. 2018. Xl-nbt: A cross-lingual neural belief tracking framework. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 414–424.

Alexis Conneau, Kartikay Khandelwal, Naman Goyal, Vishrav Chaudhary, Guillaume Wenzek, Francisco Guzmán, Edouard Grave, Myle Ott, Luke Zettlemoyer, and Veselin Stoyanov. 2020. Unsupervised cross-lingual representation learning at scale. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 8440–8451, Online. Association for Computational Linguistics.

Alexis Conneau and Guillaume Lample. 2019. Cross-lingual language model pretraining. In *Advances in Neural Information Processing Systems*, pages 7059–7069.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. Bert: Pre-training of deep bidirectional transformers for language understanding. In *NAACL-HLT (1)*.

Chih-Wen Goo, Guang Gao, Yun-Kai Hsu, Chih-Li Huo, Tsung-Chieh Chen, Keng-Wei Hsu, and Yun-Nung Chen. 2018. Slot-gated modeling for joint slot filling and intent prediction. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 2 (Short Papers)*, pages 753–757.

Yutai Hou, Yongkui Lai, Cheng Chen, Wanxiang Che, and Ting Liu. 2021. Learning to bridge metric spaces: Few-shot joint learning of intent detection and slot filling. In *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*, pages 3190–3200.

Jitin Krishnan, Antonios Anastasopoulos, Hemant Purohit, and Huzefa Rangwala. 2021. Multilingual code-switching for zero-shot cross-lingual intent prediction and slot filling. In *Proceedings of the 1st Workshop on Multilingual Representation Learning*, pages 211–223.

Haoran Li, Abhinav Arora, Shuohui Chen, Anchit Gupta, Sonal Gupta, and Yashar Mehdad. 2021. Mtop: A comprehensive multilingual task-oriented semantic parsing benchmark. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*, pages 2950–2962.

Bing Liu and Ian Lane. 2016. Attention-based recurrent neural network models for joint intent detection and slot filling. *Interspeech 2016*, pages 685–689.

Zihan Liu, Jamin Shin, Yan Xu, Genta Indra Winata, Peng Xu, Andrea Madotto, and Pascale Fung. 2019. Zero-shot cross-lingual dialogue systems with transferable latent variables. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 1297–1303.

Zihan Liu, Genta I Winata, Samuel Cahyawijaya, Andrea Madotto, Zhaojiang Lin, and Pascale Fung. 2021. On the importance of word order information in cross-lingual sequence labeling. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 35, pages 13461–13469.

Zihan Liu, Genta Indra Winata, Zhaojiang Lin, Peng Xu, and Pascale Fung. 2020. Attention-informed mixed-language training for zero-shot cross-lingual task-oriented dialogue systems. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pages 8433–8440.

Ilya Loshchilov and Frank Hutter. 2018. Decoupled weight decay regularization. In *International Conference on Learning Representations*.

Samuel Louvan and Bernardo Magnini. 2020. Recent neural methods on slot filling and intent classification for task-oriented dialogue systems: A survey. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 480–496.

Meryem M'hamdi, Doo Soon Kim, Franck Dernoncourt, Trung Bui, Xiang Ren, and Jonathan May. 2021. X-metra-ada: Cross-lingual meta-transfer learning adaptation to natural language understanding and question answering. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 3617–3632.

Telmo Pires, Eva Schlinger, and Dan Garrette. 2019. How multilingual is multilingual bert? In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 4996–5001.

Libo Qin, Wanxiang Che, Yangming Li, Haoyang Wen, and Ting Liu. 2019. A stack-propagation framework with token-level intent detection for spoken language understanding. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference*

on Natural Language Processing (EMNLP-IJCNLP), pages 2078–2087.

Libo Qin, Zhouyang Li, Wanxiang Che, Minheng Ni, and Ting Liu. 2021a. Co-gat: A co-interactive graph attention network for joint dialog act recognition and sentiment classification. *Proceedings of the AAAI Conference on Artificial Intelligence*, 35(15):13709–13717.

Libo Qin, Minheng Ni, Yue Zhang, and Wanxiang Che. 2020. Cosda-ml: Multi-lingual code-switching data augmentation for zero-shot cross-lingual nlp. *arXiv preprint arXiv:2006.06402*.

Libo Qin, Fuxuan Wei, Tianbao Xie, Xiao Xu, Wanxiang Che, and Ting Liu. 2021b. GL-GIN: Fast and accurate non-autoregressive model for joint multiple intent detection and slot filling. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 178–188, Online. Association for Computational Linguistics.

Sebastian Schuster, Sonal Gupta, Rushin Shah, and Mike Lewis. 2019. Cross-lingual transfer learning for multilingual task oriented dialog. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 3795–3805, Minneapolis, Minnesota. Association for Computational Linguistics.

Shyam Upadhyay, Manaal Faruqui, Gokhan Tür, Hakkani-Tür Dilek, and Larry Heck. 2018. (almost) zero-shot cross-lingual spoken language understanding. In *2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 6034–6038. IEEE.

Rob van der Goot, Ibrahim Sharaf, Aizhan Imankulova, Ahmet Üstün, Marija Stepanović, Alan Ramponi, Siti Oryza Khairunnisa, Mamoru Komachi, and Barbara Plank. 2021. From masked language modeling to translation: Non-english auxiliary tasks improve zero-shot spoken language understanding. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 2479–2497.

Yu Wang, Yilin Shen, and Hongxia Jin. 2018. A bi-model based rnn semantic frame parsing model for intent detection and slot filling. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 2 (Short Papers)*, pages 309–314.

Xiaodong Zhang and Houfeng Wang. 2016. A joint model of intent determination and slot filling for spoken language understanding. In *IJCAI*, volume 16, pages 2993–2999.

GRADA: Graph Generative Data Augmentation for Commonsense Reasoning

Adyasha Maharana **Mohit Bansal**
Department of Computer Science
University of North Carolina at Chapel Hill
{adyasha,mbansal}@cs.unc.edu

Abstract

Recent advances in commonsense reasoning have been fueled by the availability of large-scale human annotated datasets. Manual annotation of such datasets, many of which are based on existing knowledge bases, is expensive and not scalable. Moreover, it is challenging to build augmentation data for commonsense reasoning because the synthetic questions need to adhere to real-world scenarios. Hence, we present GRADA, a graph-generative data augmentation framework to synthesize factual data samples from knowledge graphs for commonsense reasoning datasets. First, we train a graph-to-text model for conditional generation of questions from graph entities and relations. Then, we train a generator with GAN loss to generate distractors for synthetic questions. Our approach improves performance for SocialIQA, CODAH, HellaSwag and CommonsenseQA, and works well for generative tasks like ProtoQA. We show improvement in robustness to semantic adversaries after training with GRADA and provide human evaluation of the quality of synthetic datasets in terms of factuality and answerability. Our work provides evidence and encourages future research into graph-based generative data augmentation. [1]

1 Introduction

Recent work has seen the emergence of several datasets for improving commonsense reasoning of language models through tasks like question answering (QA) (Sap et al., 2019b; Talmor et al., 2019; Bisk et al., 2020) and natural language inference (Bhagavatula et al., 2020; Zellers et al., 2019; Sakaguchi et al., 2020). Some of these datasets are based on existing knowledge graphs that represent different aspects of commonsense through entities and relations. For example, annotators for SocialIQA (Sap et al., 2019b) were shown an event from the inferential knowledge graph ATOMIC (Sap et al., 2019a) and instructed to turn it into a sentence by adding names, filling placeholders and adding context, etc. For multiple-choice QA datasets, annotators are also instructed to write distractor choices for each question. These useful datasets are collected through a time-taking and money-intensive crowdsourcing process which is hard to scale. Large pretrained models like GPT2 (Radford et al., 2018) can be finetuned to generate sentences from narrow data distributions, and it has recently been leveraged to augment datasets for text classification (Anaby-Tavor et al., 2020) and question answering (Puri et al., 2020; Yang et al., 2020). However, it is challenging to generate augmentation data for commonsense reasoning because the generated questions and answers (referred to as "synthetic" in rest of the paper) need to depict plausible real-world scenarios accurately. Hence, we develop GRADA, a graph-based generative data augmentation framework to generate

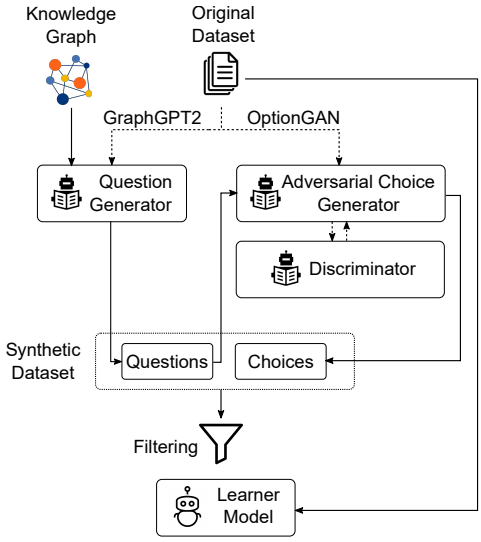

Figure 1: GRADA framework: The original dataset is used to train GraphGPT2, a graph-to-text question generator and OptionGAN, a distractor generator. The synthetic dataset is subjected to filtering and used to train the model in combniation with the original dataset.

[1] Code and synthetic data files are available at https://github.com/adymaharana/GraDA.

Proceedings of the 29th International Conference on Computational Linguistics, pages 4499–4516
October 12–17, 2022.

synthetic samples from existing knowledge graphs that encode information about the real world. We focus on generating synthetic samples for models that perform discriminative and generative commonsense question answering.

Each sample in commonsense reasoning datasets comprises a question which describes a real-world scenario and can be mapped to a set of predefined entities and relations from knowledge bases like ConceptNet and ATOMIC. For instance, the question "Besides a mattress, name something people sleep on." from the ProtoQA dataset (Boratko et al., 2020) can be mapped to the single-hop path (mattress, *RelatedTo*, people) using ConceptNet. If a pretrained language model is trained to conditionally generate questions from such input paths, we can expect it to generate sensible questions when it is provided new paths with similar relations. The model will likely generalize to unseen entity nodes and generate questions containing unique commonsense knowledge. Following this intuition, we finetune GPT2 (Radford et al., 2019) to generate questions which explicitly depict the entities and relations in input path. When trained on the aforementioned example (alongside other similar examples) and provided with the new path (mattress, *RelatedTo*, soft), our model generates "Besides a mattress, name something that's soft.", which is a valid question for probing real-world commonsense. Usually, these paths contain multiple nodes with several hops and hence are referred to as graphs in rest of the paper. In order to represent the graph, we explore both (a) encoding of linearized graph and (b) augmentation of linear encodings with structure-aware encoding of graph, and find that the latter improves the transfer of semantic knowledge from graph to text.

Synthetic questions need to be accompanied by synthetic answers and distractor choices (for multiple-choice datasets), which are similarly generated by finetuning GPT2 for conditional generation of answers/distractors from the question. However, Yang et al. (2020) report that human annotators find it hard to pick a unique/unambiguous answer in more than 50% of the synthetic dataset generated in this manner. Therefore, we explore an alternative where we finetune the generative model within a GAN framework (Nie et al., 2019a) where it is continuously challenged by a discriminator model to generate unique distractors that can fool the discriminator (see OptionGAN, Figure 1). The synthetic questions and answers thus generated are assembled into synthetic samples which are then used in a two-stage training pipeline (Mitra et al., 2019). Additionally, since the generative pipeline is only an approximate imitation of the human annotation process, we are left with several ambiguous and inaccurate samples in the synthetic pool. Hence, we retain the most informative data samples from the synthetic pool by using Question Answering Probability (Zhang and Bansal, 2019) to measure accuracy by answerability. Our contributions can be summarized as follows:

- We present a generative framework consisting of (i) a graph-to-text model to convert knowledge graphs to questions, (ii) a model finetuned with GAN loss to generate distractors for commonsense reasoning QA datasets, and (iii) combined with a filter for selecting the most informative samples from synthetic datasets.
- We improve performance on commonsense reasoning datasets, and perform ablation analysis to show the impact of various modules in our framework as well as human evaluation of synthetic dataset quality.

2 Related Work

Explicit reasoning over knowledge graphs has been a popular approach for improving commonsense understanding of QA models. Bauer et al. (2018); Lin et al. (2019); De Cao et al. (2019); Feng et al. (2020) and Lv et al. (2020) extract relevant multi-hop relational commonsense from knowledge graphs and show significant improvements over models that operate solely on text. Devlin et al. (2019); Yang et al. (2019); Ye et al. (2019) expand the rich latent knowledge of large pretrained models by finetuning on similar corpora (Havasi et al., 2010) before finetuning on the target dataset. Mitra et al. (2019) convert external resources (Koupaee and Wang, 2018) to QA samples for data augmentation. Yang et al. (2020) generate randomly initialized samples from finetuned GPT2 as augmentation data for target datasets. We ground the generated samples to real-world facts by providing knowledge graphs as input to the model.

There has been a surge of efforts in neural graph-to-text modeling in the recent years. Marcheggiani and Perez-Beltrachini (2018) encode input graphs using a graph convolutional encoder (Kipf and Welling, 2017). Koncel-Kedziorski et al. (2019)

propose the model GraphWriter which improves on the graph attention networks presented in Velickovic et al. (2018) by replacing self-attention encoder with Transformer blocks (Vaswani et al., 2017). Several recent works have shown that pretrained generative models can be finetuned with or without structure-aware graph encoding to improve graph-to-text generation (Mager et al., 2020; Ribeiro et al., 2020; Hoyle et al., 2020; He et al., 2020; Ke et al., 2021). Query or question generation has also been shown to benefit from knowledge graphs in Shen et al. (2022); Bi et al. (2020). We combine the structure-aware encoding capabilities of graph-to-text models with the rich contextual knowledge of pretrained models in GraphGPT2 and generate rich real-world scenarios from sparse sub-graphs (Shen et al., 2022; Chen et al., 2020; Kumar et al., 2019).

Good distractors are necessary for a task model to learn the right reasoning towards answering multiple-choice datasets. To this end, Liang et al. (2018) rank distractors using feature-based ensemble methods. Offerijns et al. (2020); Yang et al. (2020) finetune GPT2 to generate distractors. Chung et al. (2020) approach distractor generation as a coverage problem and select distractors for maximizing sample difficulty. Cai and Wang (2018) use adversarial training to sample high quality negative training examples for knowledge graph embeddings. In a similar line of work, we use generative adversarial networks (GANs) (Goodfellow et al., 2014) with the Gumbel-Softmax relaxation (Kusner and Hernández-Lobato, 2016; Nie et al., 2019b) and train a generator with GAN loss to imitate the creation of human-authored tricky, incorrect answer options. Most NLP applications use REINFORCE (Sutton et al., 2000) algorithm and its variants (Yu et al., 2017; Cai and Wang, 2018; Qin et al., 2018; Zhang et al., 2018) to circumvent the discrete sampling issue for text-based GANs.

3 Methods

In this section, we describe the various modules in the GRADA framework.

3.1 Graph-to-Text Generation

In the first module of our pipeline, we generate synthetic questions by using knowledge graphs as input. Given a dataset of input graphs (g_i), we finetune GPT2 with cross-entropy loss for conditional generation of questions (q_i) from the graphs i.e., $L_q = \sum_{i=1}^{N} log\, p(q_i|f(g_i))$, where $f(.)$ is the function for encoding the graph and $p(.)$ represents the probabilities. We explore linearized graph encoding as well as structure-aware encoding of graph.

Linearized Graph Input. Graph linearization is a simple way to use graphs like text when finetuning GPT2. We adopt depth-first-search to linearize the input graphs and preserve edge information to some extent by augmenting GPT2 vocabulary with special tokens for edges. GPT2 is finetuned for conditional generation of target question from this linearized graph input.

Using linearized graphs with pretrained language models (PTLMs) surpasses graph-based architectures at data-to-text generation by a large margin (Ribeiro et al., 2020). However, Mager et al. (2020) show that omitting the edge information from linearized graphs notably degrades performance, implying that graph structure is beneficial for generation. Hence, we propose GraphGPT2.

GraphGPT2 for Structure-aware Graph Input. Instead of linearizing the input graph, we encode the graph using a Transformer-based graph encoder $f_s(.)$ which preserves the graph structure by performing masked self-attention over edges and nodes. We use the Transformer-based graph encoder from Graph Writer (Koncel-Kedziorski et al., 2019) for structure-preserving encoding of graphs. First, we convert the input graphs g_i into unlabeled connected bipartite graphs $G_i = (v_i, e_i)$, where v_i is the list of entities, relations and global vertex, and e_i is the adjacency matrix describing the directed edges (Beck et al., 2018). The global vertex is connected to all entity vertices and promotes global context modelling by allowing information flow between all parts of the graph. Next, v_i is projected to a dense, continuous embedding space V_i and is sent as input to the graph encoder (see Figure 2). The encoder is composed of L stacked Transformer blocks; each Transformer block consists of a N-headed self-attention layer followed by normalization and a two-layer feed-forward network. The resulting encodings i.e. $f_s(g_i)$, are referred to as graph contextualized vertex encodings. These encodings are prepended to the embedded representation of linearized graph in the form of past key values, and sent as input to the decoder. The decoder i.e., pretrained GPT2, is finetuned to

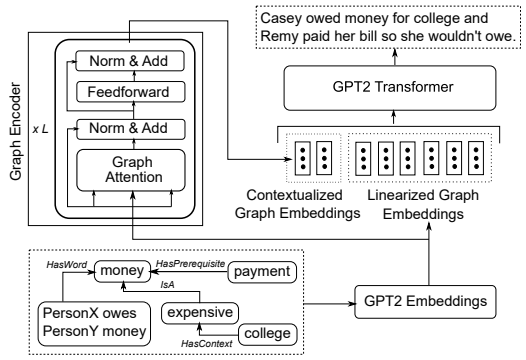

Figure 2: GraphGPT2: The Graph Encoder is composed of L Transformer blocks and its output is concatenated with GPT2 embeddings for input to GPT2.

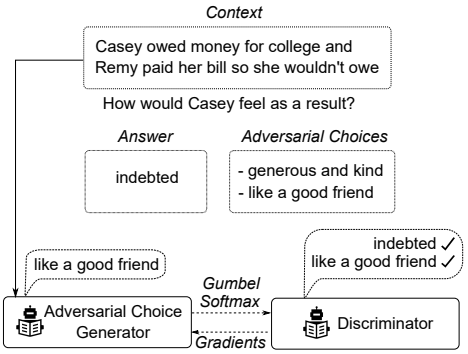

Figure 3: Training process for OptionGAN.

generate a coherent question from the combined embeddings. The graph encoder is initialized with GPT2 embeddings to force continuity in word representation across modules. Figure 2 shows the integration of graph contextualized encodings with GPT2 in GraphGPT2. The combined generative model is finetuned end-to-end for maximizing the conditional log-likelihood of target question q_i i.e. $L_q = \sum_{i=1}^{N} log\, p(q_i \,|\, [f_l(g_i); f_s(g_i)])$, where $f_l(.)$ represents the linearized graph embeddings.

During inference, both of the above models are provided with graphs that do not appear in training dataset to generate synthetic questions containing new knowledge. See Sec. 4.1 for details on creation of training and inference datasets.

3.2 Answer & Distractor Generation

We finetune a GPT2 model for conditional generation of answers from questions i.e., $L_a = \sum_{i=1}^{N} log\, p(a_i|q_i)$. However, as we discussed in Sec. 1, a similar method for conditional generation of distractors does not guarantee good distractors. Hence, we finetune GPT2 within a GAN framework to generate maximally adversarial distractors, in a bid to imitate the best human annotator.

OptionGAN for Adversarial Choices. We train a model to generate distractors (in the multiple-choice QA task) for the synthetic questions obtained from GraphGPT2 (see Figure 1) using a generator-discriminator adversarial framework. The discriminator D is a sequential classification model that takes the question q_i, concatenated with the ground truth correct answer a_i i.e., $[q_i; a_i]$ or the distractor $\hat{d}_i$ generated by generator G i.e., $[q_i; \hat{d}_i]$ as input and classifies the pair as correct or otherwise. While training, the generator runs the risk of learning to generate correct answers instead of distractors, since it's goal is to be able to fool the discriminator into classifying the question-distractor pair $[q_i; \hat{d}_i]$ as correct. To prevent this, we heavily bias the model by first pretraining it to generate only distractors using the conditional cross-entropy loss and then continue with adversarial training from the saved weights. Mathematically, we pretrain the generator G with the loss $L_g = \sum_{i=1}^{N} log\, p(d_i|q_i)$, where q_i, d_i are question and distractor, respectively. We use the question as input instead of the knowledge sub-graph, since most generated questions contain additional semantics from the latent knowledge of the pretrained generative model which is not present in the original sub-graph. Then, the pretrained generator is finetuned within an adversarial framework to produce distractors that successfully fool the discriminator, so that we get adversarial options that are as tricky as human-annotated options (see Figure 3). We use the Gumbel-Softmax relaxation (Nie et al., 2019a) while sampling from generator to allow flow of gradients through the discriminator model i.e. $z = softmax(\frac{1}{\tau}(h + g))$, where h, g and τ are the logits generated from G, Gumbel distribution sample and temperature respectively. The temperature is annealed using an exponential function during training. Following RelGAN (Nie et al., 2019a), we use the Relativistic standard GAN loss for the adversarial training i.e. $\min_G \max_D log\, sigmoid(D([q_i; a_i]) - D([q_i; \hat{d}_i]))$. Generator G is trained to minimize the loss while discriminator D is trained to maximize the loss. In practice, we use GPT2 for both roles i.e., generator as well as discriminator.

3.3 Filtering and Selection of Samples

Inspite of the careful construction of synthetic samples using knowledge graphs, the pool of synthetic

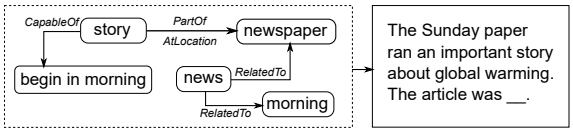

Figure 4: Example of synthetic context generated from GraphGPT2 for the CODAH dataset.

samples can be noisy and may consist of incoherent text, incorrect question-answer pairs or out-of-distribution samples. Hence, we use Question Answering Probability (QAP) (Zhang and Bansal, 2019) to measure accuracy of synthetic samples. The QAP score (μ) is the prediction probability of the true class by a model with parameters θ which has been trained on the original dataset i.e. $\mu_i = p_\theta(a_i|x_i)$. Samples with low prediction probabilities for the correct choices are either annotated incorrectly or are especially difficult instances for the model. We define a low and high threshold for the QAP filter and samples lying within this range are retained in the dataset.

See supplementary for a comparison of QAP with two other methods for filtering i.e. Energy (Liu et al., 2020) and Model Confidence & Variability (Swayamdipta et al., 2020).

4 Experimental Setup

4.1 Datasets

SocialIQA (Sap et al., 2019b) and CommonsenseQA (Talmor et al., 2019) are annotated using knowledge graphs, making them a suitable choice for testing our approach. SocialIQA is a question answering dataset based on ATOMIC (Sap et al., 2019a), containing 33,410/1954/2224 samples in training, development and test set, resp. CommonsenseQA (CQA) is a similarly crowd-sourced dataset based on ConceptNet (Speer et al., 2017) containing an official split of 9741/1221/1241 samples. Following Yang et al. (2020), we also test our method on HellaSwag-2K (Zellers et al., 2019) and CODAH (Chen et al., 2019) for low-resource scenario. HellaSwag-2K is created by sampling 2000/1000/1000 examples from HellaSWAG training and validation sets. We test our approach on the CoDAH folds (2.8k samples) released by Yang et al. (2020) for comparison. Apart from these four MCQ datasets, we also experiment with the generative QA dataset ProtoQA (9762/52/102) (Boratko et al., 2020) and find that our approach works especially well with it. See Appendix for details.

Data Preparation. To prepare graph-to-text datasets for training GraphGPT2, we map the questions to multi-hop paths in ConceptNet (Bauer et al., 2018). We use Spacy[2] to tag the questions with part-of-speech and extract verbs and nouns as concepts, retaining those that appear in ConceptNet as entities and the connecting relations (see example in Fig. 4).[3] We remove inverse relations from the set of triples. The graphs extracted in this manner are acyclic and can be linearized with a depth-first search. For SocialIQA, we map the questions to a combination of ATOMIC and ConceptNet. ATOMIC events contain nouns and verbs which are representative of the social scenario being described in the event and are further extended in the context by SocialIQA annotators. We tokenize and stem the events and contexts to extract these representative words, and compute the percentage of overlapping words in the context with respect to each event. The event with maximum overlap with context is selected as the corresponding ATOMIC subject. The ATOMIC relation is selected from the predefined map of ATOMIC relations to SocialIQA questions. This way, we recover the ATOMIC alignments of nearly 20,000 samples from training set of SocialIQA (88% acc.).

Generation of Synthetic Data. In order to prepare synthetic datasets, we create a dataset of unseen input graphs by mutating the graphs from training sets of graph-to-text datasets. One or two entities are replaced by a randomly selected entity (or relation-entity pair) with similar adjacency to other entities in the input graph, to create a mutated graph. The maximum sequence length of graph contextualized embeddings is set to 64, while that of GPT2 is set to 128. The synthetic dataset size (pre-filtering) is 100k/50k/10k/10k/50k for SocialIQA, CQA, HellaSwag-2K, Codah, and ProtoQA respectively. For generation of synthetic data for SocialIQA, we use the set of tuples from ATOMIC that do not appear in the original dataset. To prepare the synthetic dataset for CommonsenseQA, we select two adversarial choices from ConceptNet and two choices generated by OptionGAN. For ProtoQA, we find accurate answers by generating 30 sets of answers for each synthetic question, ranking the answer choices by frequency and retaining the ones that appear at least 5 times in the 30 sets. See example of synthetic context generation in Fig. 4.

[2] https://spacy.io/

[3] We use the question concept present in CQA annotations as additional concept for the questions.

Evaluation. To evaluate graph-to-text generation, we define an ORACLE score which measures the semantic relevance of synthetic question when paired with the original answer options. We replace the original question in validation set samples with the synthetic question and re-evaluate models on this modified dataset. In addition, we adopt the following NLG metrics: BLEU-4 (Papineni et al., 2002), METEOR (Banerjee and Lavie, 2005), CIDEr[4] (Vedantam et al., 2015) and BERTScore (F1 score) (Zhang et al., 2020). Models trained on the synthetic and original commmonsense reasoning datasets are evaluated using their respective task-specific accuracies (see Appendix). For ProtoQA, we report the accuracy in top-k answers where $k = 1, 3, 5$. We also perform human evaluation of the samples generated using GraphGPT2 and OptionGAN.

5 Results & Analysis

First, we present results from the complete GRADA framework followed by results from ablation experiments. Then, we discuss evaluation of the various generative models in GRADA using automated metrics as well as human annotators. Finally, we evaluate the robustness of models trained with and without GRADA to semantic adversaries and discuss upper bounds of our data augmentation pipeline. See Appendix for visualization of the quality of the synthetic datasets.

5.1 Data Augmentation Results

Results from the best GRADA model are presented in Table 1.[5] The baseline row represents results from the same task models used for GRADA but trained without any data augmentation i.e. T5-3B for ProtoQA and RoBERTa for all other datasets. We see 1-2% improvements over baseline across all multiple-choice datasets using GRADA. For the best GRADA models (selected using validation results), synthetic samples are generated from structured GraphGPT2 and OptionGAN, and filtered using QAP.[6] GRADA results in large improvements for ProtoQA i.e. 4-6% higher values on the Max Answers 1/3/5 metrics (see Appendix), suggesting the effectiveness of our approach for similar generative tasks. We see 0.3%, 0.3% and 0.26% improvement with GRADA over G-DAUG for CQA, Codah and HellaSwag-2K respectively. Our approach also performs similar to the Option Comparison Network in HyKAS (Ma et al., 2019) for CQA (row 3 in Table 1). Our approach is orthogonal to HyKAS, KG-Fusion as their instance-level approach retrieves information for each sample while GRADA augments knowledge on a global level.

Ablation results from the GRADA framework on validation sets are presented in Table 2. The first row of Table 2 presents results from baseline task models i.e., trained without data augmentation. Next, we compare results from two-stage training and see upto 1.7% ($p<0.05$ for all datasets) improvements (row 1 vs. 4 in Table 2) with the addition of synthetic data without filtering.[7] Using structured GraphGPT2 leads to 0.47% (p=0.043), 0.39% (p=0.078), 1.46% (p=0.12)[8] improvements over linearized GraphGPT2 for SocialIQA, CQA, ProtoQA and diminishing improvements for the smaller datasets. We see consistent but modest improvements which are not significant, from addition of distractors generated from OptionGAN. Even though improvements with OptionGAN are marginal, it is necessary for the completeness of the pipeline for synthetic generation. Next, adding filter to denoise the synthetic pool unequivocally improves results by large margins for all datasets except CQA. Filtering by QAP (row 5 in Table 2) provides additional benefit (p=0.069 and p=0.093 for SocialIQA and CQA, $p<0.05$ for other datasets) to downstream task models over unfiltered synthetic data augmentation (row 4).[9] See examples of high and low quality synthetic data samples filtered using QAP in Table 7. Smaller datasets benefit the most from GRADA.

Single-hop vs. Multi-hop Paths. Additionally, we finetune GraphGPT2 with sub-graphs made of single-hop paths only to generate the context. We perform data augmentation using the synthetic questions generated through this approach and compare to the GRADA results on validation sets. See results in Table 4. We observe 0.92%, 0.08%, 1.48% and 1.05% drops in performance for validation sets of SocialIQA, CQA, CODAH and Hel-

[4] https://github.com/Maluuba/nlg-eval

[5] It should be noted that the state-of-the-art UnifiedQA has 30x parameters in $\text{RoBERTa}_{\text{LARGE}}$

[6] ProtoQA is not a multiple-choice dataset, so OptionGAN is not used and we use sample perplexity as the only filter.

[7] Statistical significance is computed with 100K samples using bootstrap (Noreen, 1989; Tibshirani and Efron, 1993).

[8] p-values are larger for improvements on ProtoQA validation set which has only 52 samples.

[9] We also ran experiments with MLM pretraining (ATOMIC for SocialIQA and OMCS corpus for the rest) before finetuning on target dataset and saw <1% improvements.

Method	SocialIQA	CQA	Codah	HellaSwag-2K	ProtoQA
UnifiedQA-11B (Khashabi et al., 2020)	81.45	79.1	-	-	41.49 / 24.95 / 21.77
RoBERTa + KG Fusion (Mitra et al., 2019)	78.00	-	-	-	-
RoBERTa + HyKAS (Ma et al., 2019)	-	73.2			
BACKTRANSLATION (Yang et al., 2020)		70.2	81.8		-
G-DAUG (Yang et al., 2020)	-	72.6	84.3	75.70	-
Baseline* (No Augmentation)	76.74	72.1	82.3	73.40	35.77 / 43.81 / 49.88
GRADA	77.85	72.9	84.7	75.96	42.02 / 48.90 / 54.23

Table 1: Results on test sets of commonsense datasets and comparative results from other approaches taken from leaderboards. *We use T5-3B for ProtoQA baseline and GRADA results and RoBERTa for all other datasets.

Method	SIQA	CQA	CDH	H2K	PQA
Baseline	77.78	77.23	84.48	75.10	41.1
Synthetic Data Augmentation					
Linearized	78.21	77.55	86.07	76.40	45.63
+ Structured	78.68	77.94	86.13	76.70	46.09
+ OptionGAN	78.82	78.02	86.19	76.70	-
Filtering					
QAP*	79.12	78.06	86.81	77.60	50.34

Table 2: Ablation results on validation set of commonsense reasoning datasets. *We use sample perplexity for filtering ProtoQA samples.

Dataset	Original	GraphGPT2	
		Linearized	*Structured*
SocialIQA	75.92	55.18	57.34
CQA	77.23	57.63	58.71
CODAH	82.19	46.23	46.78
HellaSWAG-2K	76.58	41.35	41.74
ProtoQA	41.10	28.21	23.47

Table 3: ORACLE scores for question generation. Original represents the performance of baseline task models on original dataset. The columns GPT2 and GraphGPT2 represent similar evaluation with synthetic questions generated from linearized graphs and structure-aware graph encoder respectively.

laSwag respectively. The larger drops for smaller datasets suggest that multi-hop paths are effective in low-resource scenarios.

Generalization to Unseen Concepts. We looked for %overlap of entity nodes and single-hop paths (subject– relation– object) between the multi-hop KGs spanning the questions of correctly answered samples after GraDA training and the questions of synthetic data, and observed 5-60% entity overlap and <20% path overlap. This suggests GRADA also promotes reasoning capabilities of the downstream models for unseen concepts.

5.2 Generative Model Evaluation Results

ORACLE scores for the two variations of GraphGPT2 are presented in Table 3. The scores in first column refer to the validation set performance of baseline models on original datasets. These models are re-evaluated on the questions generated

Method	SIQA	CQA	CDH	H2K	PQA
Baseline	77.78	77.23	84.48	75.10	41.1
GraDA (single-hop)	78.70	77.31	85.96	76.05	45.67
GraDA (multi-hop)	79.12	78.06	86.81	77.60	50.34

Table 4: Results on validation set of commonsense reasoning datasets using single-hop vs. multi-hop graphs for GRADA pipeline.

Dataset	Question	Answer	Distractors
SocialIQA	96.1%	86.0%	50.0%
CommonsenseQA	100.0%	97.2%	25.0%
HellaSwag-2K	92.0%	88.1%	25.8%
CODAH	90.3	83.4%	30.6%
ProtoQA	97.2%	75.0%	-

Table 5: Results from human evaluation of generated questions, answers and distractors.

by GraphGPT2 (as described in Sec. 4.1). The largest improvement i.e. 2.16% (p=0.068) is observed for SocialIQA, which may be attributed to its large dataset size. We see diminishing improvements for low-resource scenarios i.e. Codah and HellaSwag-2K. We observe a similar trend when the synthetic questions are evaluated using NLG metrics (see Appendix). More importantly, since phrase-matching metrics are not ideal for NLG evaluation (Novikova et al., 2017), we also perform human evaluation to judge the quality of generation for SocialIQA and CQA as we see significant improvements from structured GraphGPT2 vs. linearized GraphGPT2. We ask annotators on Amazon Mechanical Turk[10] (AMT) to select the sentence which is more representative of the information encoded in input graph, for 100 samples from validation set. Questions generated from GraphGPT2 are preferred 46% and 53% of the times for SocialIQA and CQA resp., compared to those from linearized inputs only, showing that the addition of graph encoder improves integration of knowledge in generated text.

We perform human evaluation (AMT)

[10]Located in United States, HIT Approval Rate>98%, Number of HITs Approved>10K, $15 per hour (approx.).

G-Daug (Yang et al., 2020)	GRADA	
	Knowledge-Graph Tuple	Generated Data
A human enjoys putting rubber on furniture. They should do this before .. front of the mirror.	*S*: PersonX provides __ for PersonY's children	Taylor provided meals for Kendall's children and they all enjoyed it greatly.
There was a large, cold bite of ice on my where?	*R*: xIntent	Why did Taylor do this?
He hated flying, the controls were what?	*O*: To be helpful	[A] to be a bad friend [B] to be helpful [C] to be rude
What is a square leg made of made out of?	*S*: weasel *R*: AtLocation	The man was a weasel, he was part of a powerful what?
What country does a cow go to make a milk run?	*O*: mafia organization	[A] out of doors [b] terrarium [c] mafia organization [D] farmyard [E] backyard

Table 6: Comparison of randomly generated synthetic data from G-Daug (Yang et al., 2020) (left) and knowledge-grounded synthetic data generated using GRADA (right). (S=Subject, R=Relation, O=Object)

	High-quality synthetic samples
SIQA	Riley provided help to the community through his many charity events over the years. How would Others feel as a result? [A] selfish [B] appreciative [C] bored
CQA	When a child is upset by something, what may they do? [A] fall down [B] wish to fly [C] start crying [D] play tag [E] boy or girl
PQA	Name something you worry you're still doing when you're not supposed to. drinking, smoking, sleeping, working, using cell phone
	Low-quality synthetic samples
SIQA	Tracy raised her arm to her face to cover her eyes during the scary movie. What does Tracy need to dobefore this? [A] scared [B] be scared of the movie [C] to have a fundraiser
CQA	What will you do if you want to go public? [A] prepare for worst [B] tell family first [C] own private company [D] telegram [E] charming
PQA	Name a family tradition that has deep roots in the dialect of suzh. cooking, caroling, knitting, hunting, fishing

Table 7: High and low quality synthetic samples generated through GRADA for SIQA, CQA, ProtoQA (PQA) and ranked using QAP scores (and perplexity for PQA). Labels are marked in green.

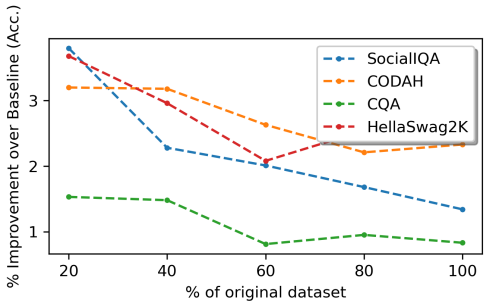

Figure 5: % improvement in accuracy over baseline with different % of original dataset. Baseline is RoBERTa finetuned on the same % of original dataset.

of answerability of the generated questions/answers/distractors on 50 randomly selected samples from the filtered augmentation data (see Table 5). Annotators were provided with the question, answer and distractors, and asked to evaluate a) if the question can be answered in a few words (b) if the question can be answered by the given answer and (c) if the distractors are wrong answers for the question. More than 90% of the questions were judged as answerable, 75-90% of the answers were judged as correct answers for the respective questions. The quality of distractors ranged from 50% for SocialIQA to 20-30% for smaller datasets. However, the overall quality of distractors is high enough to benefit data augmentation. See examples in Table 7. We also perform human evaluation for the factuality of samples generated using our method GraDA and GDaug (Yang et al., 2020). We picked a randomly sampled set of 100 synthetic QA pairs from G-Daug for the datasets CQA, Codah and HellaSWAG-2K. For a fair comparison, we collected 100 synthetic pairs from GraDA for the same datasets. We asked an annotator to evaluate if each of the synthetic QA pair adheres to a plausible real-world scenario, and found that 56% G-Daug samples were judged as factual as compared to 68% of the GraDA samples (see examples in Table 6).

5.3 Upper Bounds

We ran experiments for augmentation with 20%, 40%, 60%, 80% and 100% training data from the original set (see Fig. 5). The improvement margins from the augmentation dataset is upto 4% at 20% of the original SocialIQA dataset. We see similar trends for CODAH, HellaSwag and ProtoQA, while the improvements for CQA were <1.5%.

5.4 Robustness Evaluation

We expect that data augmentation exposes the task model to diverse language and improves its robustness to semantic adversaries in addition to boosting its performance on the target task. To evaluate this, we use the TextFooler system (Jin et al., 2020;

	OptionGAN Adversarial Choice Examples
A.	*Question*: Skylar kept their grades up by studying a lot in the weeks before finals. How would Skylar feel afterwards? *Answer*: [A] content *Adversarial Choices*: (GPT2) [B] accomplished [C] tired (OptionGAN) [B] like they don't need to study [C] afraid of studying
B.	*Question*: A group of skiers and snowboarders attend an ice event. the skiers *Answer*: [A] are doing tricks and riding around on the ice *Adversarial Choices*: (GPT2) [B] bike to the finish line [C] and snowboarders attend the event [D] are shovelling snow from the driveway (OptionGAN) [B] get on skis and ride back to the station after the race [C] walk in an ice tight formation [D] flip the snow over their heads onto the ice and back

Table 8: Examples of two scenarios were OptionGAN adversarial choices improve the synthetic question. Answers are marked in green. OptionGAN adversarial choices are marked in blue.

Yang et al., 2020; Wei and Zou, 2019) to generate adversarial text by computing word importance ranking and replacing the most influential words with their synonym in the vector space. Overall, GRADA benefits the robustness of task models and improves their failure rate by 1-3% (see Table 9).

5.5 Semantic Analysis of OptionGAN

As outlined in Sec. 1, we use OptionGAN to provide better adversarial choices for the synthetic questions generated using GraphGPT2. In this section, we perform qualitative analysis of the adversarial choices generated with and without OptionGAN in order to define the scenarios where OptionGAN provides more effective choices. First, from the analysis of a few examples, we find that OptionGAN improves the synthetic QA examples in two main ways:

- It provides wrong choices rather than the equally-correct choices (false negatives) provided by the non-adversarial choice generation (see example A in Table 8).
- It provides wrong choices that require more complex reasoning (i.e. harder true negatives) than the ones provided by the non-adversarial choice generator. (see example B in Table 8).

We conducted a larger human evaluation study of randomly picked 50 synthetic samples from the four synthetic QA datasets (SocialIQA, H2K, CQA, CODAH) and compared the synthetic adversarial choices generated with and without OptionGAN. We observe that, in 30.7% of the cases, OptionGAN choices were more adversarial than the ones generated without OptionGAN. Within those examples, nearly 60% samples were categorized into the first category and 30% were categorized into the second category, as described above. However, the improvements from OptionGAN are limited in the downstream task, suggesting that 'harder negatives' are required for effective training.

Method	**SIQA**	**CQA**	**CDH**	**H2K**	**PQA**
Baseline	21.7/10.3	14.9/12.5	31.3/16.1	19.4/10.6	5.1/16.2
GRADA	22.4/10.8	15.8/12.9	34.8/18.2	20.5/11.5	6.3/16.8

Table 9: Robustness Evaluation. Failure rate / perturbation ratio (higher is better) from TextFooler experiments are shown on development sets.

6 Conclusion

We present GRADA, a graph-based data augmentation framework for commonsense reasoning QA datasets. We train a graph-to-text question generator and GAN-based adversarial choice generator for creating synthetic data samples, which are used to augment the original datasets. GRADA promotes factuality in synthetic samples and improves results on five downstream datasets.

7 Ethical Considerations

The usage of pretrained generative models in any downstream application requires careful consideration of the real-world impact of generated text. In our approach, we provide concrete inputs for grounding the generated text to specific entities and relations which encode real-world facts, thereby reducing the possibility of propagating unintended stereotypical and social biases embedded within the pretrained models. However, since these entities and relations are derived from existing knowledge bases like ConceptNet (Speer et al., 2017), there is potential for transfer of bias present in these resources to the generated texts. Additionally, the graph-to-text generative models in GRADA pose the same risk as other data-to-text generative models (Ribeiro et al., 2020; Hoyle et al., 2020; Mager et al., 2020) i.e. the models can be made to generate incorrect facts by providing incorrect data as input. Therefore, we recommend restricting the use of GRADA to low-risk, unbiased graphs inputs.

Acknowledgments

We thank the reviewers for their useful feedback. This work was supported by DARPA MCS Grant N66001-19-2-403, ONR Grant N00014-18-1-2871, and NSF-CAREER Award 1846185. The views are those of the authors and not of the funding agency.

References

Ateret Anaby-Tavor, Boaz Carmeli, Esther Goldbraich, Amir Kantor, George Kour, Segev Shlomov, Naama Tepper, and Naama Zwerdling. 2020. Not enough data? deep learning to the rescue! In *Thirty-Fourth AAAI Conference on Artificial Intelligence*.

Satanjeev Banerjee and Alon Lavie. 2005. Meteor: An automatic metric for mt evaluation with improved correlation with human judgments. In *Proceedings of the acl workshop on intrinsic and extrinsic evaluation measures for machine translation and/or summarization*, pages 65–72.

Lisa Bauer, Yicheng Wang, and Mohit Bansal. 2018. Commonsense for generative multi-hop question answering tasks. In *Proceedings of the 2018 Conference on EMNLP*, pages 4220–4230.

Daniel Beck, Gholamreza Haffari, and Trevor Cohn. 2018. Graph-to-sequence learning using gated graph neural networks. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 273–283.

Chandra Bhagavatula, Ronan Le Bras, Chaitanya Malaviya, Keisuke Sakaguchi, Ari Holtzman, Hannah Rashkin, Doug Downey, Wen-tau Yih, and Yejin Choi. 2020. Abductive commonsense reasoning. In *8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, April 26-30, 2020*. OpenReview.net.

Sheng Bi, Xiya Cheng, Yuan-Fang Li, Yongzhen Wang, and Guilin Qi. 2020. Knowledge-enriched, type-constrained and grammar-guided question generation over knowledge bases. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 2776–2786.

Yonatan Bisk, Rowan Zellers, Ronan Le Bras, Jianfeng Gao, and Yejin Choi. 2020. Piqa: Reasoning about physical commonsense in natural language. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pages 3027–3035.

Michael Boratko, Xiang Li, Tim O'Gorman, Rajarshi Das, Dan Le, and Andrew McCallum. 2020. Protoqa: A question answering dataset for prototypical common-sense reasoning. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1122–1136.

Antoine Bosselut, Hannah Rashkin, Maarten Sap, Chaitanya Malaviya, Asli Celikyilmaz, and Yejin Choi. 2019. Comet: Commonsense transformers for automatic knowledge graph construction. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 4762–4779.

Liwei Cai and William Yang Wang. 2018. Kbgan: Adversarial learning for knowledge graph embeddings. In *Proceedings of NAACL-HLT*, pages 1470–1480.

Michael Chen, Mike D'Arcy, Alisa Liu, Jared Fernandez, and Doug Downey. 2019. Codah: An adversarially authored question-answer dataset for common sense. In *Proceedings of the 3rd Workshop on Evaluating Vector Space Representations for NLP*.

Yu Chen, Lingfei Wu, and Mohammed J Zaki. 2020. Toward subgraph guided knowledge graph question generation with graph neural networks. *arXiv preprint arXiv:2004.06015*.

Ho-Lam Chung, Ying-Hong Chan, and Yao-Chung Fan. 2020. A bert-based distractor generation scheme with multi-tasking and negative answer training strategies. *arXiv preprint arXiv:2010.05384*.

Nicola De Cao, Wilker Aziz, and Ivan Titov. 2019. Question answering by reasoning across documents with graph convolutional networks. In *Proceedings of NAACL-HLT*, pages 2306–2317.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. Bert: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186.

Yanlin Feng, Xinyue Chen, Bill Yuchen Lin, Peifeng Wang, Jun Yan, and Xiang Ren. 2020. Scalable multi-hop relational reasoning for knowledge-aware question answering. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1295–1309.

Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. 2014. Generative adversarial nets. In *Advances in neural information processing systems*, pages 2672–2680.

Catherine Havasi, Robert Speer, Kenneth Arnold, Henry Lieberman, Jason Alonso, and Jesse Moeller. 2010. Open mind common sense: Crowd-sourcing for common sense. In *Workshops at the Twenty-Fourth AAAI Conference on Artificial Intelligence*.

Bin He, Di Zhou, Jinghui Xiao, Xin Jiang, Qun Liu, Nicholas Jing Yuan, and Tong Xu. 2020. Integrating graph contextualized knowledge into pre-trained language models. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: Findings*, pages 2281–2290.

Alexander Hoyle, Ana Marasović, and Noah Smith. 2020. Promoting graph awareness in linearized graph-to-text generation. *arXiv preprint arXiv:2012.15793*.

Di Jin, Zhijing Jin, Joey Tianyi Zhou, and Peter Szolovits. 2020. Is bert really robust? natural language attack on text classification and entailment. In *AAAI*.

Pei Ke, Haozhe Ji, Yu Ran, Xin Cui, Liwei Wang, Linfeng Song, Xiaoyan Zhu, and Minlie Huang. 2021. Jointgt: Graph-text joint representation learning for text generation from knowledge graphs. In *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*, pages 2526–2538.

Daniel Khashabi, Sewon Min, Tushar Khot, Ashish Sabharwal, Oyvind Tafjord, Peter Clark, and Hannaneh Hajishirzi. 2020. Unifiedqa: Crossing format boundaries with a single qa system. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: Findings*, pages 1896–1907.

Thomas N Kipf and Max Welling. 2017. Semi-supervised classification with graph convolutional networks. In *ICLR*. OpenReview.net.

Rik Koncel-Kedziorski, Dhanush Bekal, Yi Luan, Mirella Lapata, and Hannaneh Hajishirzi. 2019. Text generation from knowledge graphs with graph transformers. In *Proceedings of the 2019 Conference of the NAACL: HLT, Volume 1 (Long and Short Papers)*, pages 2284–2293.

Mahnaz Koupaee and William Yang Wang. 2018. Wikihow: A large scale text summarization dataset. *arXiv preprint arXiv:1810.09305*.

Vishwajeet Kumar, Yuncheng Hua, Ganesh Ramakrishnan, Guilin Qi, Lianli Gao, and Yuan-Fang Li. 2019. Difficulty-controllable multi-hop question generation from knowledge graphs. In *International Semantic Web Conference*, pages 382–398. Springer.

Matt J Kusner and José Miguel Hernández-Lobato. 2016. GANs for sequences of discrete elements with the gumbel-softmax distribution. *arXiv preprint arXiv:1611.04051*.

Chen Liang, Xiao Yang, Neisarg Dave, Drew Wham, Bart Pursel, and C Lee Giles. 2018. Distractor generation for multiple choice questions using learning to rank. In *Proceedings of the thirteenth workshop on innovative use of NLP for building educational applications*, pages 284–290.

Bill Yuchen Lin, Xinyue Chen, Jamin Chen, and Xiang Ren. 2019. Kagnet: Knowledge-aware graph networks for commonsense reasoning. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 2822–2832.

Weitang Liu, Xiaoyun Wang, John Owens, and Yixuan Li. 2020. Energy-based out-of-distribution detection. *Advances in Neural Information Processing Systems*, 33.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Roberta: A robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692*.

Shangwen Lv, Daya Guo, Jingjing Xu, Duyu Tang, Nan Duan, Ming Gong, Linjun Shou, Daxin Jiang, Guihong Cao, and Songlin Hu. 2020. Graph-based reasoning over heterogeneous external knowledge for commonsense question answering. In *Proceedings of the Thirty-Fourth Conference on Association for the Advancement of Artificial Intelligence (AAAI)*.

Kaixin Ma, Jonathan Francis, Quanyang Lu, Eric Nyberg, and Alessandro Oltramari. 2019. Towards generalizable neuro-symbolic systems for commonsense question answering. In *Proceedings of the First Workshop on Commonsense Inference in Natural Language Processing*, pages 22–32.

Manuel Mager, Ramón Fernandez Astudillo, Tahira Naseem, Md Arafat Sultan, Young-Suk Lee, Radu Florian, and Salim Roukos. 2020. Gpt-too: A language-model-first approach for amr-to-text generation. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 1846–1852.

Diego Marcheggiani and Laura Perez-Beltrachini. 2018. Deep graph convolutional encoders for structured data to text generation. In *Proceedings of the 11th International Conference on Natural Language Generation*, pages 1–9.

Arindam Mitra, Pratyay Banerjee, Kuntal Kumar Pal, Swaroop Mishra, and Chitta Baral. 2019. Exploring ways to incorporate additional knowledge to improve natural language commonsense question answering. *arXiv preprint arXiv:1909.08855*.

Weili Nie, Nina Narodytska, and Ankit Patel. 2019a. Relgan: Relational generative adversarial networks for text generation. In *7th International Conference on Learning Representations, ICLR 2019, New Orleans, LA, USA, May 6-9, 2019*. OpenReview.net.

Weili Nie, Nina Narodytska, and Ankit Patel. 2019b. Relgan: Relational generative adversarial networks for text generation. In *7th International Conference on Learning Representations, ICLR 2019, New Orleans, LA, USA, May 6-9, 2019*. OpenReview.net.

Eric W Noreen. 1989. *Computer-intensive methods for testing hypotheses*. Wiley New York.

Jekaterina Novikova, Ondřej Dušek, Amanda Cercas Curry, and Verena Rieser. 2017. Why we need new evaluation metrics for NLG. In *Proceedings of*

the 2017 Conference on Empirical Methods in Natural Language Processing, pages 2241–2252, Copenhagen, Denmark. Association for Computational Linguistics.

Jeroen Offerijns, Suzan Verberne, and Tessa Verhoef. 2020. Better distractions: Transformer-based distractor generation and multiple choice question filtering. *arXiv preprint arXiv:2010.09598*.

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. Bleu: a method for automatic evaluation of machine translation. In *Proceedings of the 40th annual meeting on association for computational linguistics*, pages 311–318. Association for Computational Linguistics.

Raul Puri, Ryan Spring, Mostofa Patwary, Mohammad Shoeybi, and Bryan Catanzaro. 2020. Training question answering models from synthetic data. *arXiv preprint arXiv:2002.09599*.

Pengda Qin, Weiran Xu, and William Yang Wang. 2018. Dsgan: Generative adversarial training for distant supervision relation extraction. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 496–505.

Alec Radford, Karthik Narasimhan, Tim Salimans, and Ilya Sutskever. 2018. Improving language understanding by generative pre-training. *OpenAI Technical Report*.

Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. 2019. Language models are unsupervised multitask learners.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J Liu. 2020. Exploring the limits of transfer learning with a unified text-to-text transformer. *Journal of Machine Learning Research*, 21:1–67.

Leonardo FR Ribeiro, Martin Schmitt, Hinrich Schütze, and Iryna Gurevych. 2020. Investigating pretrained language models for graph-to-text generation. *arXiv preprint arXiv:2007.08426*.

Keisuke Sakaguchi, Ronan Le Bras, Chandra Bhagavatula, and Yejin Choi. 2020. Winogrande: An adversarial winograd schema challenge at scale. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34.

Maarten Sap, Ronan Le Bras, Emily Allaway, Chandra Bhagavatula, Nicholas Lourie, Hannah Rashkin, Brendan Roof, Noah A Smith, and Yejin Choi. 2019a. Atomic: An atlas of machine commonsense for if-then reasoning. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, pages 3027–3035.

Maarten Sap, Hannah Rashkin, Derek Chen, Ronan Le Bras, and Yejin Choi. 2019b. Social iqa: Commonsense reasoning about social interactions. In *Proceedings of the 2019 Conference on EMNLP and the 9th IJCNLP*, pages 4453–4463.

Xinyao Shen, Jiangjie Chen, Jiaze Chen, Chun Zeng, and Yanghua Xiao. 2022. Diversified query generation guided by knowledge graph. In *Proceedings of the Fifteenth ACM International Conference on Web Search and Data Mining*, pages 897–907.

Jasper Snoek, Hugo Larochelle, and Ryan P Adams. 2012. Practical bayesian optimization of machine learning algorithms. In *Advances in neural information processing systems*, pages 2951–2959.

Robyn Speer, Joshua Chin, and Catherine Havasi. 2017. Conceptnet 5.5: An open multilingual graph of general knowledge. In *Thirty-First AAAI Conference on Artificial Intelligence*.

Richard S Sutton, David A McAllester, Satinder P Singh, and Yishay Mansour. 2000. Policy gradient methods for reinforcement learning with function approximation. In *Advances in neural information processing systems*, pages 1057–1063.

Swabha Swayamdipta, Roy Schwartz, Nicholas Lourie, Yizhong Wang, Hannaneh Hajishirzi, Noah A Smith, and Yejin Choi. 2020. Dataset cartography: Mapping and diagnosing datasets with training dynamics. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 9275–9293.

Christian Szegedy, Vincent Vanhoucke, Sergey Ioffe, Jon Shlens, and Zbigniew Wojna. 2016. Rethinking the inception architecture for computer vision. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2818–2826.

Alon Talmor, Jonathan Herzig, Nicholas Lourie, and Jonathan Berant. 2019. Commonsenseqa: A question answering challenge targeting commonsense knowledge. In *Proceedings of the 2019 Conference of the NAACL: HLT, Volume 1 (Long and Short Papers)*, pages 4149–4158.

Robert J Tibshirani and Bradley Efron. 1993. An introduction to the bootstrap. *Monographs on statistics and applied probability*, 57:1–436.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in neural information processing systems*, pages 5998–6008.

Ramakrishna Vedantam, C Lawrence Zitnick, and Devi Parikh. 2015. Cider: Consensus-based image description evaluation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 4566–4575.

Petar Velickovic, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Liò, and Yoshua Bengio. 2018. Graph attention networks. In *6th International Conference on Learning Representations, ICLR 2018, Vancouver, BC, Canada, April 30 - May 3, 2018, Conference Track Proceedings*. OpenReview.net.

Jason Wei and Kai Zou. 2019. Eda: Easy data augmentation techniques for boosting performance on text classification tasks. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 6383–6389.

Yiben Yang, Chaitanya Malaviya, Jared Fernandez, Swabha Swayamdipta, Ronan Le Bras, Ji-Ping Wang, Chandra Bhagavatula, Yejin Choi, and Doug Downey. 2020. G-daug: Generative data augmentation for commonsense reasoning. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: Findings*, pages 1008–1025.

Zhilin Yang, Zihang Dai, Yiming Yang, Jaime Carbonell, Russ R Salakhutdinov, and Quoc V Le. 2019. Xlnet: Generalized autoregressive pretraining for language understanding. In *Advances in neural information processing systems*, pages 5754–5764.

Zhi-Xiu Ye, Qian Chen, Wen Wang, and Zhen-Hua Ling. 2019. Align, mask and select: A simple method for incorporating commonsense knowledge into language representation models. *arXiv preprint arXiv:1908.06725*.

Lantao Yu, Weinan Zhang, Jun Wang, and Yong Yu. 2017. Seqgan: Sequence generative adversarial nets with policy gradient. In *Thirty-First AAAI Conference on Artificial Intelligence*.

Rowan Zellers, Ari Holtzman, Yonatan Bisk, Ali Farhadi, and Yejin Choi. 2019. Hellaswag: Can a machine really finish your sentence? In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 4791–4800.

Shiyue Zhang and Mohit Bansal. 2019. Addressing semantic drift in question generation for semi-supervised question answering. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 2495–2509.

Tianyi Zhang, Varsha Kishore, Felix Wu, Kilian Q Weinberger, and Yoav Artzi. 2020. Bertscore: Evaluating text generation with bert. In *8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, April 26-30, 2020*. OpenReview.net.

Zhirui Zhang, Shujie Liu, Mu Li, Ming Zhou, and Enhong Chen. 2018. Bidirectional generative adversarial networks for neural machine translation. In *Proceedings of the 22nd Conference on Computational Natural Language Learning*, pages 190–199. Association for Computational Linguistics.

A Experiment Setup

Datasets: Social IQA (Sap et al., 2019b) and CommonsenseQA (Talmor et al., 2019) are popular datasets based on knowledge graphs, making them a suitable choice for testing our approach. Social IQA is a multiple-choice question answering dataset. Each sample consists of a context, question and three multiple choices. CommonsenseQA is also a multiple-choice QA dataset, wherein each sample consists of a context and five multiple choices. Of those 5 choices, three are taken from ConceptNet and the other two are authored by annotators. We only use the human-authored incorrect choices to train our adversarial choice generator OptionGAN. The ATOMIC knowledge graph contains 24K base events and 877K tuples describing a variety of social scenarios. We use the 710K training split introduced in Bosselut et al. (2019) to randomly sample 100K tuples as the seed subgraphs for generation of synthetic data dataset for Social IQA. For CommonsenseQA, we use the entire ConceptNet knowledge graph, subject to pruning as outlined in Talmor et al. (2019), to sample seed tuples for synthetic dataset generation. For SocialIQA, CQA, Codah and HellaSwag-2K, we use simple accuracy for model evaluation.

ProtoQA (Boratko et al., 2020) is a generative QA dataset which is evaluated by 7 different metrics[11]. We report the first 3 metrics i.e. Max Answers 1/3/5. For tables showing only one number for ProtoQA, such as the ablation table in main text, we report the Max Answer 1 metric. In order to train T5-3B for ProtoQA, we concatenate the ranked choices for each question and finetune the model for conditional generation of this concatenated string from the input question.

All of the above datasets are being for their intended purposes i.e. research only, in our work. All of these datasets are in the English language.

Data Preparation: To prepare graph-to-text datasets for training GraphGPT2, we map the questions to multi-hop paths in ConceptNet (Bauer et al., 2018). We use Spacy[12] to tag the questions with part-of-speech and extract verbs and nouns as concepts, retaining those that appear in ConceptNet as entities[13]. For SocialIQA, we map the questions to a combination of ATOMIC and

[11] https://github.com/iesl/protoqa-evaluator

[12] https://spacy.io/

[13] We use the question concept present in CQA annotations as additional concept for the questions.

ConceptNet. ATOMIC events contain nouns and verbs which are representative of the social scenario being described in the event and are further extended in the context by Social IQA annotators (see Table 6). We tokenize and stem the events and contexts to extract these representative words, and compute the percentage of overlapping words in the context with respect to each event. The event with maximum overlap with context is selected as the corresponding ATOMIC subject. The ATOMIC relation is selected from the predefined map of ATOMIC relations to Social IQA questions. This way, we recover the ATOMIC alignments of 20,000 samples from training set of SocialIQA with 88% accuracy. We introduce additional tokens to the vocabulary of GPT2, in order to represent the set of relations present in the knowledge graph. Multi-word entities are embedded using an average of embeddings across individual tokens.

Synthetic Data Generation. In order to prepare synthetic datasets, we create a dataset of unseen input graphs by mutating the graphs from training sets of graph-to-text datasets. One or two entities are replaced by a randomly selected entity (or relation-entity pair) with similar adjacency to other entities in the input graph, to create a mutated graph. The synthetic dataset size (pre-fitering) is 100k/50k/10k/10k/50k for SocialIQA, CQA, HellaSwag-2K, Codah, and ProtoQA respectively. For generation of synthetic data, we use the set of tuples from ATOMIC and ConceptNet that do not appear in SocialIQA and CommonsenseQA datasets respectively. To prepare the synthetic dataset for CommonsenseQA, we select two adversarial choices from ConceptNet and two choices generated by OptionGAN. For ProtoQA, we find accurate answers by generating 30 samples of answers for each synthetic question, ranking the answer choices by frequency and retaining the ones that appear atleast 5 times in the 30 samples. After this, the synthetic question and answer (concatenation of high-frequency answer choices) is subjected to filtering. Due to lack of option for supplementary in this submission, we have included a sample of the generated synthetic examples in Table 10.

A.1 Filtering and Selection of Samples

Inspite of the careful construction of synthetic samples using knowledge graphs, the pool of synthetic samples can be noisy and may consist of incoherent text, incorrect question-answer pairs or out-of-distribution samples. Hence, we compare the effect of three different methods to filter samples on downstream task performance.

Question Answering Probability (QAP). The QAP score (μ) (Zhang and Bansal, 2019) is the prediction probability of the true class by a model with parameters θ which has been trained on the original dataset i.e. $\mu_i = p_\theta(y_i^*|x_i)$. Samples with low prediction probabilities for the correct choices are either annotated incorrectly or are especially difficult instances for the model. We define a low and high threshold for the QAP filter and samples lying within this range are retained in the dataset.

Model Confidence and Variability. Swayamdipta et al. (2020) propose the model confidence ($\hat{\mu_i}$) and variability ($\hat{\sigma_i}$) measures to identify the effect of data samples on the model's generalization error. Specifically, $\hat{\mu_i} = \frac{1}{E}\sum_{e=1}^{E} p_\theta(y_i^*|x_i)$ and $\hat{\sigma_i} = \sqrt{\frac{\sum_{e=1}^{E}(p_\theta(y_i^*|x_i)-\hat{\mu_i})^2}{E}}$, where E is training epochs. They find that ambiguous samples i.e., high variability and mid-range confidence, contribute the most to test performance on downstream task. Following this, we define low and high thresholds for both confidence and variability in order to find the most informative samples.

Energy. Liu et al. (2020) show that the energy score can be reliably used for distinguishing between in- and out-of-distribution (OOD) samples, as compared to the traditional approach of using the softmax scores. We introduce an energy threshold to select samples which are out-of-distribution i.e. $E_i = -log\sum_j^C e^{p_\theta(y_i^j|x)}$ where C is the number of choices in the QA sample, and measure the effect of using OOD samples as augmentation data.

A.2 Training Details

Baselines: We use pretrained RoBERTa$_{\text{LARGE}}$ (Liu et al., 2019) for multiple-choice datasets and T5-3B (Raffel et al., 2020) for ProtoQA as the task models. The baseline task model is finetuned on original datasets with no data augmentation, and is used as scoring model for filtering. We use GPT2$_{\text{MEDIUM}}$ for GraphGPT2, GPT2$_{\text{SMALL}}$ as the pretrained generator and discriminator for OptionGAN. For GRADA, the model is first finetuned on synthetic samples using label smoothing (Szegedy et al., 2016) and then on original dataset. We refer the reader to Koncel-Kedziorski et al. (2019) for full implementation details of the Graph Encoder

HellaSWAG-2K	
Question	*Answer*
A close up of a gymnast is shown. a gymnast balances on beam as she sweeps __	(a) over obstacles. (b) around with other gymnast. (c) performs a front squat and a flip, and crosses her arms. (d) performing multiple back and forth moves.
"We then search for a car by its model and make. Once we get the car model __	(a) we determine what the tires are for. (b) we either buy a new or recycle it. If we want to recycle the car, we simply (c) click the buy now button. The seller will then provide a description of the car and (d) we'll add it to the computer so we can make a list of the different models we'll
A man in black robes is walking into a bar. He __	(a) is telling several anecdotes about how he has been following other people around and talking to them. (b) speaks to a group of workers and they all rise and raise their arms in the air. (c) starts singing into the microphone. (D) begins a beat down on a man standing behind him.
CODAH	
Question	*Answer*
I am feeling very hungry. I think that __	(a) I will have dinner. (b) I will drink some milk. (c) I will sleep a lot. (d) I will play catch with my grandpa.
A man with no body hair was peacefully wallowing in the sea of ocean. The man then __	(a) was surrounded by a flock of birds. (b) hung from the ceiling and sang (c) began to carpet the beach. (d) watched a movie with his headphones on.
A man excitedly planned a surprise party for his friend. He __	(a) got a shotgun. (b) put up a giant neon sign with his own hand painted on it. (c) decided to end his life in front of his friend. (d) planned to brew a cup of coffee and play chess.
ProtoQA	
Question	*Answer*
Name something you worry you're still doing when you're not supposed to.	drinking, smoking, sleeping, working, using cell phone
Besides milk, name a popular product in the dairy market.	cheese, ice cream, yogurt, butter
Name something you can disagree about.	religion, politics, parenting, weight, money
If you sent a postcard from china what would be pictured on the front?	great wall, temple, dragon
Name something a knight needs for a good day's work.	horse, armour, sword, lance, shield

Table 10: Examples of synthetic samples generated for HellaSWAG-2K, CODAH and ProtoQA datasets from the GRADA pipeline. Correct answers for multiple-choice questions are marked in green.

in GraphGPT2.

OptionGAN: It is tricky to train GAN models, especially with discrete data like text. We follow the training method in Nie et al. (2019a) to fine-tune the adversarial choice generator in a minimax game with discriminator. In addition to the training parameters mentioned in Table 18, we restrict the number of training iterations to 5000, and perform one gradient descent step on generator for every 5 gradient descent steps on discriminator.

Training & Hyperparameter Tuning. After generation of synthetic examples, we perform two-stage training of the task models. In the first phase, the model is finetuned on synthetic data only, In the second phase, the model is finetuned on the original dataset. The model trained in first phase is subject to bayesian optimization (Snoek et al., 2012) of filter parameters.

Parameter	**Bounds**
Filter Parameters	
QAP/Model Confidence Lower Threshold	[0.0, 0.49]
QAP/Model Confidence Higher Threshold	[0.51, 1.0]
Energy Lower Threhsold	[0.0, 1.0]
Energy Higher Threshold	[0.0, 1.0]
Model Variability Lower Threshold	[0.0, 0.5]
Model Variability Higher Threshold	[0.0, 0.5]
Training Parameters	
Learning Rate	[1, 10]*1e-6
Batch Size (inc)	[4, 8, 16]
Total Train Epochs	[3, 5]

Table 11: Optimization bounds for grid-search based tuning of training hyperparameters.

A.3 Human Evaluation

Generative source of the sentences are omitted when presented to annotators. The input graphs are seed tuples from ATOMIC and ConceptNet for samples from the development sets of Social IQA

Method	BLEU4	METEOR	CIDEr	BERTScore
Social IQA				
GPT2	14.58	26.41	132.84	89.12
GraphGPT2	15.37	26.95	135.91	91.83
CommonsenseQA				
GPT2	1.71	12.78	30.89	85.76
GraphGPT2	1.90	13.64	33.76	87.34

Table 12: Comparison of performance for GPT2 and GraphGPT2 on development sets.

Dataset	Wins	Loses	Tie
SocialIQA	46%	37%	17%
CommonsenseQA	53%	31%	16%

Table 13: Results from comparative human evaluation of generated questions. Wins and Loses refer to the %times synthetic question generated from structured graph input was chosen over linearized graph.

and CommonsenseQA respectively. The annotators can pick both the sentences if either of them are equally relevant in their subjective opinion. We allow for a single hit for each assignment in Amazon Mechanical Turk.

B Results

B.1 Generative Model Evaluation

As shown in Table 12, we see small improvements for BLEU-4 and METEOR, but larger improvements in other metrics from GraphGPT2 i.e., 3.07% (p=0.027), 2.87% (p=0.035) in CIDEr, and 2.71% (p=0.042), 1.58% (p=0.056) in BERTScore for Social IQA and CQA, resp. The phrase-matching metric scores are low for CQA, which may be attributed to its small sample size. However, BERTScore for CQA lies between 85-88%, showing that the model manages to convey similar meaning as human-annotated context albeit with different words.

More importantly, since phrase-matching metrics are not ideal for NLG evaluation (Novikova et al., 2017), we also perform human evaluation to judge the quality of generation for SocialIQA and CommonsenseQA as we see significant improvements from structured GraphGPT2 vs. linearized GraphGPT2. We ask annotators on Amazon Mechanical Turk[14] to select the sentence which is more representative of the information encoded in input graph, for 100 samples from validation set. Results are shown in Table 13. Samples generated from structured input are selected significantly more times than those from linearized inputs, for both SocialIQA and CQA, showing that addition of a graph encoder improves representation of knowledge in generated sample.

[14]Located in United States, HIT Approval Rate>98%, Number of HITs Approved>10K.

Method	SIQA	CQA	CDH	H2K	PQA
Baseline	77.78	77.23	84.48	75.10	41.1
Filtering					
QAP*	79.12	78.06	86.81	77.60	50.34
Confidence	79.05	77.83	86.59	77.40	-
Energy	78.76	77.79	86.38	77.10	-

Table 14: Ablation results on validation set of commonsense reasoning datasets using various filtering methods. *We use sample perplexity for filtering ProtoQA samples.

Additionally, we perform human evaluation of the samples generated using GraphGPT2 and OptionGAN. We randomly select 50 samples from the filtered augmentation datasets for each of the five datasets, and ask 2 annotators to answer 3 yes/no questions about the quality of the question, answer and distractors respectively. We present results from the survey in Table 5. More than 90% of the questions in each dataset were judged as answerable, showing the effectiveness of GraphGPT2 as well as the QAP-based filtering method. Similarly, 75-90% of the answers were judged as correct answers for the respective questions. The quality of distractors were relatively lower, ranging from 50% for larger datasets like SocialIQA to 20-30% for rest of the datasets. The inter-annotator agreement was also low (<0.6) for distractor judgements, suggesting the general difficulty of both tasks: distractor generation and measurement of distractor quality. However, the overall quality of distractors in our datasets is high enough to benefit data augmentation.

For both human evaluation annotation tasks, it was made clear in the instructions that the data is being collected for research purposes only.

B.2 Comparison of Filtering Methods

Table 14 demonstrates the effect of using various methods of filtering i.e. QAP, Energy and Model Confidence/Variability. Results are shown on the validation sets the commonsense reasoning datasets. We see largest improvements with using QAP as the filter. Similar improvements are seen with the confidence/variability scores; however, it requires scores from multiple finetuned models from various training checkpoints.

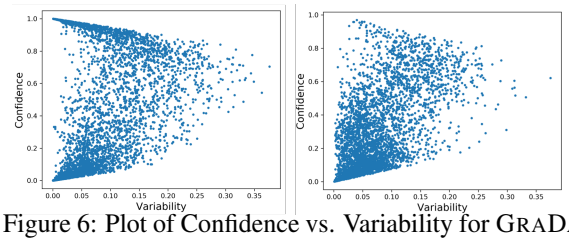

Figure 6: Plot of Confidence vs. Variability for GRADA synthetic samples for CQA (left) and H2K (right).

B.3 Robustness Evaluation

We expect that data augmentation exposes the task model to diverse language and improves its robustness to semantic adversaries in addition to boosting its performance on the target task. To evaluate this, we use the TextFooler system (Jin et al., 2020; Yang et al., 2020; Wei and Zou, 2019) to generate adversarial text by computing word importance ranking and replacing the most influential words with their synonym in the vector space. Failure rate is the %examples for which TextFooler fails to change the original model prediction, and average perturbation ratio is the average % of words replaced when TextFooler succeeds at changing the prediction. We use our best GRADA models in comparison with baseline models (Table 9). Overall, GRADA positively impacts the robustness of task models to TextFooler and improves the failure rate by >3% for Codah and upto 1% for all other datasets. We observe similar trends for the perturbation ratios too. This shows that GRADA improves semantic robustness of the models. It is also worthwhile noting that generative task models like T5-3B for ProtoQA are especially prone to adversarial attacks like TextFooler with a mere 5-6% failure rate and there needs to be more research towards improving their robustness.

B.4 Cartography Quality Evaluation

We use dataset cartography Swayamdipta et al. (2020) to visualize the quality of our synthetic datasets. Samples in top left of figure are easy, while samples towards bottom and right of the figure are difficult and ambiguous respectively. We can observe from the figure that the synthetic dataset for CQA (left) has a higher % of easy samples than HellaSwag-2K, suggesting that the quality of synthetic samples generated by GRADA improves with original dataset size. Moreover, when applying QAP filtering, using the entire synthetic dataset yields largest improvements for CQA whereas for HellaSwag-2K (right), the lower cutoff for QAP is 0.3 which filters out most of the samples present in bottom part of the plot. This suggests that in low-resource scenarios, it is important to remove inaccurate samples, while larger datasets benefit from ambiguous and inaccurate samples.

Best Parameters	Social IQA	CQA	Codah	HellaSwag-2K	ProtoQA
QAP Lower Threshold	0.49	0.32	0.43	0.49	0.27
QAP Higher Threshold	1.0	1.0	1.0	1.0	1.0

Table 15: Best Filter Hyperparameters.

	Social IQA			CommonsenseQA		
Hyperparameter	**Baseline**	**GRADA Phase 1**	**GRADA Phase 2**	**Baseline**	**GRADA Phase 1**	**GRADA Phase 2**
Learning Rate	5e-6	4e-6	3e-6	1e-5	5e-6	1e-5
Epochs	3	1	3	5	1	5
Max Gradient Norm	1.0	1.0	1.0	None	None	None
Weight Decay	0.01	0.01	0.01	0.01	0.01	0.01
Batch Size	8	8	8	16	16	16
Max Length	128	128	128	70	70	70
Warmup Ratio	0.0	0.0	0.0	0.06	0.06	0.0
LR Decay	Linear	Linear	Linear	Linear	Linear	Linear
Optimizer	AdamW	AdamW	AdamW	AdamW	AdamW	AdamW
Hardware	RTX 2080Ti	RTX 2080Ti	RTX 2080Ti	RTX 2080Ti	RTX 2080Ti	RTX 2080Ti
Single GPU Hours	5 hrs	1.5 hrs	5 hrs	2 hrs	0.5 hrs	2 hrs

Table 16: Training hyperparameters for baseline and two-stage GRADA training of SocialIQA and CQA

	CODAH			HellaSwag-2K		
Hyperparameter	**Baseline**	**GRADA Phase 1**	**GRADA Phase 2**	**Baseline**	**GRADA Phase 1**	**GRADA Phase 2**
Learning Rate	1e-5	4e-6	3e-6	5e-5	5e-6	1e-5
Epochs	5	1	5	5	1	5
Max Gradient Norm	1.0	1.0	1.0	None	None	None
Weight Decay	0.01	0.01	0.01	0.01	0.01	0.01
Batch Size	16	8	16	8	8	8
Max Length	90	90	90	128	128	128
Warmup Ratio	0.06	0.06	0.06	0.06	0.06	0.06
LR Decay	Linear	Linear	Linear	Linear	Linear	Linear
Optimizer	AdamW	AdamW	AdamW	AdamW	AdamW	AdamW
Hardware	RTX 2080Ti	RTX 2080Ti	RTX 2080Ti	RTX 2080Ti	RTX 2080Ti	RTX 2080Ti
Single GPU Hours	2 hrs*	1 hr* hrs	2 hrs*	0.5 hr	0.2 hr	0.5 hr

Table 17: Training hyperparameters for baseline and two-stage GraDA training of RoBERTa models for HellaSwag-2K and CODAH. *values reported for five-fold training

		OptionGAN		
Hyperparameter	**GraphGPT2**	**Generator**	**Discriminator**	**GAN**
Learning Rate	4e-5	1e-5	1e-5	1e-6
Epochs	5	5	3	-
Max Gradient Norm	1.0	1.0	1.0	None
Weight Decay	0.0	0.01	0.01	0.01
Batch Size	8	8	8	4
Max Length	128	128	128	128
Warmup Ratio	0.0	0.0	0.0	0.06
LR Decay	Linear	Linear	Linear	Linear
Optimizer	AdamW	AdamW	AdamW	AdamW

Table 18: Training hyperparameters for GraphGPT2, Generator, Discriminator and OptionGAN

Eureka: Neural Insight Learning for Knowledge Graph Reasoning

Alex X. Zhang[1], **Xun Liang**[1]*, **Bo Wu**[1], **Xiangping Zheng**[1],
Sensen Zhang[1], **Yuhui Guo**[1], **Jun Wang**[2], **Xinyao Liu**[3]
[1]Renmin University of China
[2]Swinburne University of Technology
[3]Hong Kong University of Science and Technology
{zhangxuanalex, xliang, wubochn, xpzheng,
sensen0126, yhguo}@ruc.edu.cn
junwang@swin.edu.au liuxinyao575599@163.com

Abstract

The human recognition system has presented the remarkable ability to effortlessly learn novel knowledge from only a few trigger events based on prior knowledge, which is called insight learning. Mimicking such behavior on Knowledge Graph Reasoning (KGR) is an interesting and challenging research problem with many practical applications. Simultaneously, existing works, such as knowledge embedding and few-shot learning models, have been limited to conducting KGR in either "seen-to-seen" or "unseen-to-unseen" scenarios. To this end, we propose a neural insight learning framework named Eureka to bridge the "seen" to "unseen" gap. Eureka is empowered to learn the seen relations with sufficient training triples while providing the flexibility of learning unseen relations given only one trigger without sacrificing its performance on seen relations. Eureka meets our expectation of the model to acquire seen and unseen relations at no extra cost, and eliminate the need to retrain when encountering emerging unseen relations. Experimental results on two real-world datasets demonstrate that the proposed framework also outperforms various state-of-the-art baselines on datasets of both seen and unseen relations.

1 Introduction

Human knowledge provides a formal understanding of the world. Knowledge graphs (KGs) that represent structural relations between entities in the form of (*head entity*, *relation*, *tail entity*) have become an increasingly popular research direction towards cognition and human-level intelligence. These triples of KGs are abbreviated using (h, r, t) in this paper. Over the last few years, the works (Bordes et al., 2013; Sun et al., 2019) on knowledge embedding have achieved impressive results in the knowledge graph reasoning (KGR) task. To successfully learn a set of relations, these methods usually require a large number of training triples and cannot infer missing facts of unseen relations due to the sparse interactions, which are essentially transductive learning processes in terms of the relations. Thus, we categorize this line of knowledge embedding research as "seen-to-seen" methods; i.e., reasoning from seen relations to seen relations. Moreover, the representations of the relations in KGs produced by knowledge embedding models always remain fixed after training. They may be sub-optimal since the real-world large-scale KGs dynamically evolve quickly with new relations emerging every day, rather than staying static.

Suppose that we would like to expand the set of relations that the knowledge embedding models can recognize. We need to collect training triples for the emerging (unseen) relations; i.e., those not in the initial training set, and then restart the aforementioned computationally costly training procedure on the enhanced training set. Not to mention the fact that the model may not perform well when only few training examples are available for the unseen relations (Xiong et al., 2018).

To alleviate the above challenge, some few-shot learning methods (Xiong et al., 2018; Zhang et al., 2020) have been proposed, which can be seen as inductive learning approaches. Their basic ideas are to predict new facts in a meta-learning framework in a setting where only few training triples for each unseen relation are available. We term them as "unseen-to-unseen" methods; i.e., reasoning from unseen relations to unseen relations. This is possible since the meta-learning framework can simulate the unseen relations during meta-training, while they are unobservable in conventional learning schemes. However, their performance will reach a plateau as the number of training examples increases, as illustrated in Figure 1. Moreover, they cannot perform as well as knowledge embedding models on the initial seen relations with suffi-

*Corresponding author

Proceedings of the 29th International Conference on Computational Linguistics, pages 4517–4527
October 12–17, 2022.

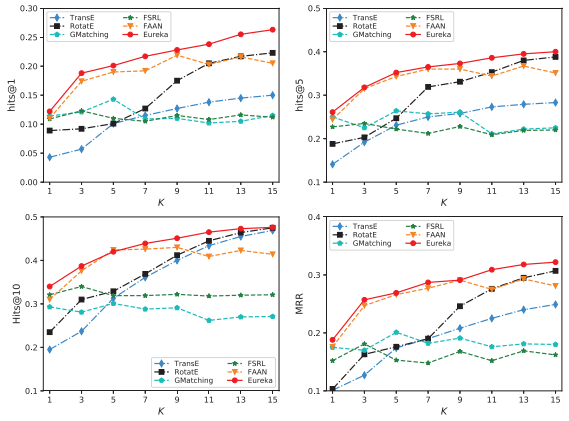

Figure 1: Impact of the size of training triples on NELL dataset. K is the number of training triples per relation. TransE and RotatE are typical knowledge embedding models; GMatching, FSRL, and FAAN are few-shot learning methods; Eureka is our model.

cient training triples, as the few-shot learners are adapted to the common parts of the different meta-tasks and forget the relation-specific information. In other words, the dramatic performance of few-shot learning methods on unseen relations comes at the cost of dysfunction on seen relations with sufficient training examples available. It is also different from the human learning system, where new concepts can be learned from very few examples at no extra cost.

Meanwhile, the human learning system exhibits the remarkable ability to effortlessly discover novel concepts during the "Eureka moment", with only one or few examples as the trigger. For example, a child, having accumulated enough knowledge (seen relations) like "CEO", "President" and so on, can easily learn and generalize the unseen concept of "Leader"from only a single knowledge set like (Gandhi, Leader_of, India) by analogy. Mimicking this behavior in artificial reasoning systems is an interesting and very challenging research problem with many practical advantages, such as developing real-time knowledge reasoning systems for downstream applications, such as language models.

Motivated by the limitations of knowledge embedding methods and few-shot KGR models, we mimic human insight learning modes in machine learning and propose a neural insight learning framework termed Eureka for KGR tasks. The whole structure of Eureka is illustrated in Figure 2. Eureka aims to tackle the problem knowledge embedding methods and few-shot KGR models encountered, under a more realistic setting, where a large set of training triples are assumed to exist for seen relations; and using these data as the sole input, we want to develop a KGR model that, is not only capable of recognizing these seen relations, but also learning unseen relations from only one training example (provided only at the testing time) without sacrificing the performance on seen relations or requiring to be retrained). We also devise a cross-domain attention (CDA) network to model the semantic interactions and bridge the gap between unseen and seen relations; for example, the unseen relation "Leaderof" has semantic similarity with the seen relations "Presidentof" and "CEOof" and modeling such similarity will help to make up for the lack of training information of unseen relations and represent the unseen relation more accurately to some extent.

Compared to prior approaches, we believe that Eureka resembles more closely the human learning behavior (w.r.t. how it learns novel concepts). Eureka is also more suitable in the real-world scenario where unseen relations do not emerge one by one but may emerge simultaneously as a set, with only few triples available for each new unseen relation.

To summarize, the contributions are as follows:

- To the best of our knowledge, Eureka is the first neural insight learning framework for KGR by mimicking human learning behaviors, which can efficiently learn new unseen relations based on one given trigger and the learned seen relations at no extra cost.
- In contrast to the previous works, Eureka bridges the "seen" to "unseen" gap with the CDA networks and provides the flexibility of inferring missing facts for both seen and unseen relations in a unified protocol.
- The extensive experimental results on two real-world datasets show the superiority of Eureka compared with the state-of-the-art baselines on both seen and unseen relations.

2 Related Work

The neural insight learning framework draws on the previous research in knowledge embedding and meta-learning.

2.1 Knowledge Embedding

Knowledge embedding aims to model multi-relational data and automatically inferring missing facts in knowledge graphs. Many of them encode both entities and relations into a continuous low dimensional vector space. TransE (Bordes et al., 2013) is a classic work that encodes both entities

and relations into a 1-D vector space. DistMult (Yang et al., 2015) and ComplEx (Trouillon et al., 2016) attempt to mine latent semantics to benefit their KG embeddings. CoKE (Wang et al., 2019) presents a novel paradigm that takes into account KGs' contextual nature and learns contextualized knowledge graph embedding based on the transformer. There are also other effective models like ConvE (Dettmers et al., 2018a), Rotate (Sun et al., 2019) and UniKER (Cheng et al., 2021). These embedding-based models rely heavily on extensive collections of training instances, and they are not able to deal with sparse triples, as presented in (Xiong et al., 2018).

2.2 Meta-Learning

Meta-learning, commonly known as *learning to learn* (Lake et al., 2015), refers to the process of improving the learning algorithm itself over multiple learning episodes. Contrary to conventional machine learning approaches where tasks are handled from scratch using a fixed learning algorithm, meta-learning provides an opportunity to dynamically adapt to new tasks with the learned algorithm.

One line of meta-learning research, which is closely related to our work, is few-shot learning. Few-shot learning methods seek to learn novel concepts with only a small number of labeled examples. Recent deep learning based few-shot learning algorithms can be classified into three groups. The first group is **model-based approaches**, which depend on a specially designed part, like memory, to quickly optimize the model parameters given few-shot training instances. MetaNet (Munkhdalai and Yu, 2017), a typical model-based approach, learns meta knowledge across tasks and generalizes rapidly via its fast parameterization. The second group is **metric-based approaches**, which try to learn a generalizable metric and the corresponding matching functions among a set of training instances. For example, prototypical networks (Snell et al., 2017) classify each instance by calculating its similarity to the prototype representation of each class, whose idea is similar to some nearest neighbor algorithms. GMatching (Xiong et al., 2018), FSRL (Zhang et al., 2020), and FAAN (Sheng et al., 2020) can also be considered as metric-based approaches. The third group is **optimization-based approaches**, which aim to learn faster by changing the optimization methods on few-shot reference instances. One example is the model-agnostic meta-learning (MAML) (Finn et al., 2017), which first proposed a framework of parameter updating for a task-specific learner and performing meta-optimization across tasks by using the above updated parameters. MetaR (Chen et al., 2019), MetaP (Jiang et al., 2021) and GANA (Niu et al., 2021) can be regarded as optimization-based approaches for few-shot KGR.

3 Preliminaries

In this section, we formally describe neural insight learning in the KGR scenario and leave technical details to the next section. According to the Gestalt theory of learning, insight learning occurs spontaneously when people discover new knowledge within their prior knowledge as a result of reasoning or problem-solving processes that reorganize or restructure that knowledge (Ash et al., 2012). In other words, there are two key points about machine insight learning in the KGR scenario. One is that the model could learn new unseen relations with a trigger based on the prior seen relations it learned, the other is that the model should achieve good performance on both seen and unseen relations as human beings do.

We first present the definition of KGR, then formalize neural insight learning in the KGR scenario. The difference between Eureka and previous relevant learning theories such as meta-learning (Hospedales et al., 2021), transfer learning (Pan and Yang, 2010), one-shot learning (Wang et al., 2020), and one-pass learning (Zhou et al., 2016) are also discussed.

Definition 1. *Given an incomplete knowledge graph $\mathcal{G}$ presented as $\{(h, r, t)\} \subseteq \mathcal{E} \times \mathcal{R} \times \mathcal{E}$, where $\mathcal{E}$ and $\mathcal{R}$ denote the entity set and relation set, the KG reasoning task aims at finding a set of missing triples; i.e., predicting relations r between two existing entities: $(h, ?, t)$, or predicting the tail entity t given the head entity and the relation: $(h, r, ?)$, or predicting the head entity h given the relation and the tail entity: $(?, r, t)$.*

In practical experimental settings, it is reasonable to predict the tail entity to test a KGR model. Usually, we aim to rank the triples with the true tail entity higher than those with the false tail entities.

Eureka consists of two stages: prior knowledge learning and trigger learning. The former mimics where the human beings acquire basic prior knowledge. The latter aims at learning new knowledge given only one trigger as the training example,

based on the prior knowledge they gained. Taking the KGR task as an example, the objective of prior knowledge learning can be presented as follows:

$$\min_{\theta} \mathbb{E}\left[\sum_{(h,r,t)\in\mathcal{G}_a\cup\bar{\mathcal{G}}_a} \ell_\theta\left(h,r,t \mid \mathcal{G}_a,\bar{\mathcal{G}}_a\right)\right], \quad (1)$$

where $\mathcal{G}_a$ is a KG full of triples containing seen relations and $\bar{\mathcal{G}}_a$ is a set of invalid triples generated by polluting the tail entities of valid triples in $\mathcal{G}_a$; $\ell_\theta\left(h,r,t \mid \mathcal{G}_a,\bar{\mathcal{G}}_a\right)$ is an arbitrary ranking loss function, and θ is the parameter of Eureka including the embeddings of entities and seen relations. This stage is very similar to conventional knowledge embedding models.

Trigger learning imitates the human ability of fast learning new knowledge based on their prior knowledge after being stimulated by a new phenomenon. We sample one new training triple episodically as a trigger for the model to acquire the unseen relation. The objective of trigger learning is defined as:

$$\min_{\varphi} \mathbb{E}_{D_{r'}}\left[\sum_{(h_i,r',t_i)\in T^{\text{test}}_{r'}\cup\bar{\mathcal{G}}_b} \frac{\ell_\varphi\left(h_i,r',t_i \mid \theta^*,\bar{\mathcal{G}}_b,T^{\text{train}}_{r'}\right)}{\left|T^{\text{test}}_{r'}\right|}\right], \quad (2)$$

where r' is a unseen relation and $D_{r'} = \{T^{\text{train}}_{r'}, T^{\text{test}}_{r'}\}$ is sampled from $\mathcal{G}_b \cup \bar{\mathcal{G}}_b$; $\mathcal{G}_b$ is a KG of unseen relations and $\bar{\mathcal{G}}_b$ is a set of invalid triples generated by polluting tail entities of triples in $\mathcal{G}_b$. The relations in $\mathcal{G}_b$ and $\mathcal{G}_a$ are disjointed; i.e., the relations in $\mathcal{G}_b$ are the unseen relations for the model trained on $\mathcal{G}_a$. Each $T^{\text{train}}_{r'}$ contains only one training triple (h_0, r', t_0) as a trigger. The $T^{\text{test}}_{r'} = \{(h_i, r', t_i)\}$ is comprised of the testing triples of r' with ground-truth tail entity and the invalid tail entities for each query (h_i, r'). θ^* is the learned optimal parameter of prior knowledge learning stage. $\ell_\varphi\left(h_i,r',t_i \mid \theta^*,\bar{\mathcal{G}}_b,T^{\text{train}}_{r'}\right)$ is the loss function of trigger learning stage and φ is the parameter to learn.

There are also some learning theories that neural insight learning looks a bit similar to. We list them as follows:

1) **Meta-Learning** (Hospedales et al., 2021): Meta-learning, also termed as *learning to learn*, refers to the paradigm of improving a learning algorithm given the experience of multiple learning episodes.

2) **Transfer Learning** (Pan and Yang, 2010): Transfer learning focuses on storing knowledge gained from a source domain and applying it to a different but related target domain.

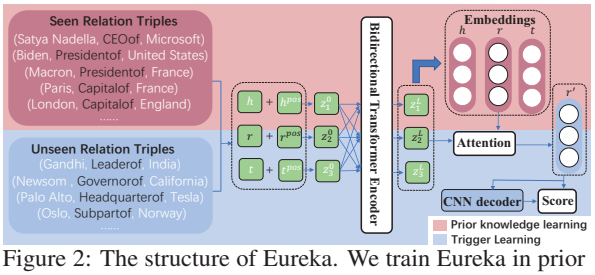

Figure 2: The structure of Eureka. We train Eureka in prior knowledge learning and trigger learning stages. Specifically, Eureka acquires the embeddings of entities and seen relations in the prior knowledge learning stage through a bidirectional transformer encoder and CNN-based decoder. With the learned representations of seen relations, Eureka then learns the unseen relations efficiently through a cross-domain attention network. Note that Eureka shares the same encoder and decoder in the two stages.

3) **One-Shot Learning/Few-Shot Learning** (Wang et al., 2020): Whereas most machine learning algorithms require training on hundreds or thousands of samples and very large datasets, one-shot/few-shot learning aims to learn information about object categories from only one/few training samples.

4) **One-Pass Learning** (Zhou et al., 2016; Hou and Zhou, 2018): One-pass learning is proposed to predict new coming samples' label and update the model based on the prediction, where coming samples are used only once and never stored.

It is obvious that we adopt a meta-learning framework to formalize and implement Eureka. However, we use a more strict setting where only one training example can be seen and the trained model is capable of performing well on the unseen relations without sacrificing its performance on the seen relations. Our neural insight learning can be termed as *trigger to learn* as the meta-learning is also known as *learning to learn*. Eureka can be also seen as a subset of transfer learning with more restrictions since it could learn unseen relation representations in the target domain based on the prior knowledge acquired from the source domain. Although one-shot learning and Eureka both use one training instance as input, the goal of one-shot learning is to learn the common knowledge across the tasks and forget the relation-specific prior knowledge while Eureka wants to remember prior knowledge. One pass learning basically refers to learning by seeing the data once. So if we learn by taking data as a single instance, mini-batch, or large batches as long as we go over them once (epoch=1), they qualify as one pass learning. However, our neural insight learning only takes one training example as input and could also satisfy the evolving streaming data

nature in the real world.

4 Neural Insight Learning

4.1 Overall Architecture

To fully mimic human insight learning behaviors, Eureka is built on a two-stage learning framework. Figure 2 shows an overview of Eureka. In the prior knowledge learning stage, Eureka takes triples sampled from the $\mathcal{G}_a$ as input, similar to the knowledge embedding models (Wang et al., 2019) based on deep neural networks. We use a bidirectional transformer encoder and a CNN-based decoder to learn the dynamic embeddings of entities and relations of $\mathcal{G}_a$ and the parameters of the encoder-decoder model. The second stage, termed trigger learning, is designed to learn new knowledge with only one training example as a trigger based on the prior knowledge Eureka gained. Thus, we adopt a meta-learning framework to implement trigger learning. During meta-training step, the trigger sampled from $\mathcal{G}_b$ only contains one training triple (h_0, r', t_0) for each unseen relation r'. The representations of r' can be produced by the trained encoder-decoder model. However, only a single training example cannot guarantee the accurate representations of r' as the previous research on knowledge embedding (Xiong et al., 2018) claimed. We use a CDA network to make up for the lack of training information of unseen relations. The CDA mechanism incorporates the embeddings of the relevant seen relations with the calibrated embeddings of r' to acquire accurate representations of unseen relations. The same CNN-based decoder is then applied to judge whether the query is true or not with the given trigger during the meta-testing step.

4.2 Prior Knowledge Learning

We expect that Eureka to learn new relations with only one trigger and preserves good performance on seen relations; i.e. the embeddings of entities and relations should ideally evolve with the newly added triggers. Thus, we need an encoder to produce dynamic embeddings for every component given its graph contexts. The pre-trained language models such as BERT (Devlin et al., 2019), GPT-2 (Radford et al., 2019), have recently made great progress in learning contextualized word embeddings with transformers (Vaswani et al., 2017). Inspired by these techniques, we employ a bidirectional transformer encoder to model graph contexts and produce the dynamic contextual embeddings of entities and seen relations. In contrast to previous sequential left-to-right or right-to-left encoding strategies for the elements in a triple (Guo et al., 2019), our model applies a multi-head self-attention mechanism to model context information, which allows each element to pay attention to all elements in the sequence. Given a triple (h, r, t), we obtain a sequence $X = (x_1, x_2, x_3)$, where x_1, x_2, x_3 represent the head entity, relation, and tail entity, respectively. Since we aim to acquire the well-learned embeddings among triples, we follow the settings of most knowledge embedding models; i.e., modeling the semantic knowledge among triples instead of paths and walks outstretching from the entity and relation like CoKE (Wang et al., 2019). For each element x_i in X, the input of our transformer encoder is constructed as:

$$\mathbf{m}_i^0 = \mathbf{x}_i^{\text{ele}} + \mathbf{x}_i^{\text{pos}}, \tag{3}$$

where $\mathbf{x}_i^{\text{ele}}$ and $\mathbf{x}_i^{\text{pos}}$ denote the element embedding and the position embedding, respectively. The former is used to identify the current element, and the latter represents its position in the sequence. We feed the input vectors into a stack of L transformer blocks to encode X:

$$\mathbf{m}_i^l = \text{Transformer}\left(\mathbf{m}_i^{l-1}\right), l = 1, 2, \cdots, L, \tag{4}$$

where $\mathbf{m}_i^l$ is the hidden state of x_i after l-th layer.

Then we are allowed to obtain a sequence of three encoded vectors $(\mathbf{m}_1^L, \mathbf{m}_2^L, \mathbf{m}_3^L)$ for the triple (h, r, t). (Fan et al., 2020) indicates that a given layer of transformers can only access low-level representations and it restricts the model from fully exploiting the sequential nature of the input. Transformers also have challenges in modeling hierarchical structures (Hahn, 2020). Thus we adopt CNN as the decoder in Eureka since CNN can explore the high-level representations and model the hierarchical structures of the interactions between entities and relations by nonlinear feature learning. The score function for the triple is designed as

$$f(h, r, t) = \text{pooling}(\sigma([\mathbf{m}_1^L, \mathbf{m}_2^L, \mathbf{m}_3^L] * \omega))^\top \mathbf{u}, \tag{5}$$

where pooling is a max-pooling operator and σ denotes an activation function. $[\mathbf{m}_1^L, \mathbf{m}_2^L, \mathbf{m}_3^L] \in \mathbb{R}^{d \times 3}$ is a matrix generated by stacking $\mathbf{m}_1^L$, $\mathbf{m}_2^L$ and $\mathbf{m}_3^L$, and d is the embedding size; $*$ denotes a convolution operator; $\omega \in \mathbb{R}^{s \times 3}$ is a set of filters with s being the number of filters; and $\mathbf{u} \in \mathbb{R}^d$ denotes a weight vector. Unlike ConvE (Dettmers et al., 2018b), the decoder of Eureka stacks ele-

ments of the triple instead of concatenating the relation and entities in the triple. The stack operation for feature maps, which is fed to a Conv2D network, increases the learning ability of latent features. The pooling operator is empowered to capture the most important semantic feature from each feature map and reduces the number of weight parameters.

Depending on the scoring function $f(h, r, t)$, we adopt a binary cross-entropy (BCE) loss as (Nguyen et al., 2018). It applies a softplus (Glorot et al., 2011) to the score of each (positive or negative) triple and uses the cross-entropy between the resulting likelihood and the triple's label as loss:

$$\mathcal{L} = \sum_{(h,r,t)\in\mathcal{G}_a^*} \log\left(1 + \exp\left(-l_{(h,r,t)} \cdot f(h, r, t)\right)\right), \tag{6}$$

in which, $l_{(h,r,t)} = \begin{cases} 1 \text{ for } (h,r,t) \in \mathcal{G}_a \\ -1 \text{ for } (h,r,t) \in \bar{\mathcal{G}}_a \end{cases}$ where $\mathcal{G}_a$ and $\bar{\mathcal{G}}_a$ are collections of valid and invalid triples; $\mathcal{G}_a^* = \left\{\mathcal{G}_a \cup \bar{\mathcal{G}}_a\right\}$; $\bar{\mathcal{G}}_a$ is generated by corrupting tail entities of valid triples in $\mathcal{G}_a$.

4.3 Trigger Learning

In this stage, we explore how to learn an unseen relation representation with only one trigger. Since we aim to predict new facts on both seen and unseen relations, Eureka shares the same encoder and decoder across the two training stages. In other words, the representation vectors $(\mathbf{m}'_1, \mathbf{m}'_2, \mathbf{m}'_3)$ for a trigger (h_0, r', t_0) is output through Equation 3 and 4. Note that Eureka cannot see r' in the prior knowledge learning stage. Thus, r' is randomly initialized before being fed to the transformer encoder. In the dynamic scenario, the representations of new relations cannot be sufficiently trained on knowledge embedding models given limited training triples and thus the embeddings of r' output by the transformer encoder can be not accurate. To empower Eureka to adapt well to unseen relations and learn from prior knowledge, we borrow the idea of transfer learning and design a CDA to model the semantic interactions between unseen and seen relations and bridge the "seen-to-unseen" gap.

In the "seen-to-unseen" reasoning scenario, we adopt a meta-learning framework to implement trigger learning. For a specific trigger (h_0, r', t_0), the CDA network can be presented as $\text{CDA}(\mathbf{m}'_1, \mathbf{m}'_2, \mathbf{m}'_3, \mathbf{W}_a)$, where $\mathbf{m}'_1, \mathbf{m}'_2, \mathbf{m}'_3$ is the output embeddings by the transformer encoder and $\mathbf{W}_a$ is a set of all seen relations' embeddings generated in the prior knowledge learning stage. The semantic interactions between seen relations and r' is modeled as follows:

$$\mathbf{w}'_i = \sum_{t=1}^{K} \text{ATTENTION}\left(\frac{\mathbf{m}'_2}{\|\mathbf{m}'_2\|}, k_t\right) \cdot \frac{\mathbf{w}_a^t}{\|\mathbf{w}_a^t\|}, \tag{7}$$

where $\left\{k_t \in \mathbb{R}^d\right\}_1^K$ is a set of learnable keys (one per seen relation) used for indexing the memory. $\mathbf{w}_a^t$ is the t-th row vector of the prior knowledge matrix $\mathbf{W}_a$, which represents the embedding of the seen relation r_t. Then the final representation vector of unseen relation r' can be computed as:

$$\mathbf{w}_i^* = \lambda_1 \odot \mathbf{m}'_2 + \lambda_2 \odot \mathbf{w}'_i, \tag{8}$$

where λ_1 and $\lambda_2 \in \mathbb{R}^d$ are weight matrices to learn; $\odot$ is the Hadamard product. Through the above process, our model is able to explicitly leverage the acquired semantic knowledge from the seen relations. Note that we only consider a closed set of entities and an open set of relations in this scenario. To be more specific, the testing triples share the same entities with the training triples while the relations of testing triples are disjointed from the relations of training triples.

Since we aim to predict the missing links in a unified protocol for both seen and unseen relations, we adopt the same decoder and loss function as the prior knowledge learning does. To be more specific, we replace $[\mathbf{m}_1^L, \mathbf{m}_2^L, \mathbf{m}_3^L]$ in Equation 5 with $[\mathbf{m}'_1, \mathbf{w}_i^*, \mathbf{m}'_3]$ to constructed trigger learning's score function and adopt the same loss function as Equation 6 implemented in $\mathcal{G}_b$.

5 Experiments

We investigate three issues with Eureka: (1) Could Eureka improve the performance of KGR on unseen relations at no extra cost? (2) Is each component in Eureka necessary? (3) How CDA works? To answer these questions, we conduct experiments on two KG datasets and systematically analyze the corresponding results.

5.1 Datasets

We use two public datasets for experiments, NELL and Wiki, which are released by (Xiong et al., 2018). NELL is derived from a system that can continuously acquire diverse structured knowledge (Mitchell et al., 2015). Wiki is constructed based on Wikidata (Vrandečić and Krötzsch, 2014). The dataset statistics are shown in Table 1. We randomly select a number of relations with more

Table 1: Statistics of the Datasets. # Entities, # Relations and # Triples denote the number of unique entities, relations and triples in the datasets, respectively.

Dataset	#Entities	# Relations	# Triples
NELL	68,545	358	181,109
Wiki	4,838,244	822	5,859,240

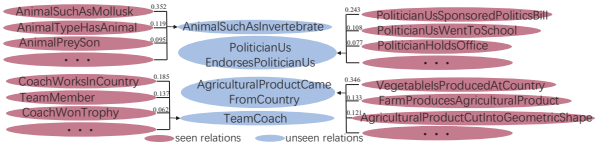

Figure 3: The most contributive seen relations in different tasks for unseen relations in NELL. Here we present top 3 seen relations and their attention weights.

than 1000 triples as seen relations, and the relations less than 500 but more than 50 triples as unseen relations. There are 67 and 183 unseen relations in NELL and Wiki data, respectively. Besides, we use 51/5/11 unseen relations for training/validation/testing in NELL and the division is set to 133/16/34 in Wiki during the few-shot learning stage. The datasets used in the prior knowledge learning stage are constructed by assigning triples of each seen relation in the ratio of 7:1:2 to the training/validation/testing set.

5.2 Baseline Methods

We select two kinds of baseline methods including knowledge embedding models and few-shot learning models: 1) **Knowledge Embedding Models.** This line of research models multi-relational structure in KGs and encodes both entities and relations into a continuous low dimensional vector space. We consider four widely used baselines as follows: TransE (Bordes et al., 2013), DistMult (Yang et al., 2015), ComplEx (Trouillon et al., 2016), RotatE (Sun et al., 2019) and UniKER(Cheng et al., 2021). For fair comparison, we consider the transformer-based model termed CoKE (Wang et al., 2019) as the baseline method. All training triples of the seen relations, as well as the trigger triples of unseen relations, are used during training. 2) **Few-Shot Learning Models.** These models concentrate on predicting new facts in KGs with only few-shot reference triples. We select four typical models; i.e., GMatching (Xiong et al., 2018), MetaR (Chen et al., 2019), FSRL (Zhang et al., 2020), FAAN (Sheng et al., 2020) and GANA (Niu et al., 2021). Note that we adopt one-shot setting for these methods for fair comparison since Eureka only gets one available trigger.

5.3 Implementation Details

At the prior knowledge learning stage, Eureka is trained, evaluated, and tested solely on the triples of seen relations. We vary the number of transformer layers in $\{2, 3, 4\}$, the number of transformer heads in $\{2, 3, 4, 5, 6\}$, the head size in $\{128, 256, 512, 1024\}$, the number of filters in $\{128, 256, 512, 1024\}$. We also apply dropout to the transformer layers with the rate in $\{0.1, 0.2, 0.3, 0.4, 0.5\}$ to avoid over-fitting. For parameter updates, we use Adam (Kingma and Ba, 2015) with the initial learning rate of 0.005 and we have the learning rate decay 5 times for each 50k training step. At the trigger learning stage, Eureka is trained, evaluated, and tested solely on the triples of unseen relations with one training data available. We also use Adam (Kingma and Ba, 2015) with the initial learning rate of 0.001 to optimize our model. Then we have the learning rate decay 5 times for each 10k training step. We evaluate all methods for every 10k training steps, and select the best models leading to the highest Hits@10 on the validation set within 500k steps as (Xiong et al., 2018); and then we get the optimal hyper-parameters, and report the final results on the testing set.

5.4 Evaluation Metrics

Following the widely used evaluation metrics in KGR tasks (Bordes et al., 2013), we adopt Hits@k, i.e., the proportion of correct entities ranked in the top k, and MRR, i.e., the mean reciprocal rank, to evaluate the overall performance of Eureka. Generally, the higher MRR and Hits@k indicate the better performance. k is set to 1, 5, and 10.

5.5 Results

We first evaluate our model on unseen relations, which predicts new facts on a query set where no seen relations are included. As shown in Table 2, Eureka shows a significant improvement margin over both knowledge embedding models and few-shot learning models. Taking the best performing few-shot learning model on NELL (FAAN) as an example, the improvement (%) of Eureka on testing MRR and Hits@10 are 23.9% and 14.5%, respectively. It, to some extent, confirms the effectiveness of the idea that the unseen relation representation can benefit from semantic interactions of seen relations and even be composed as a linear combination of the similar seen relation embeddings.

We also perform a KGR experiment on the seen

Table 2: The overall results of seen and unseen relations on testing datasets. We present the best baseline results by underline and highlight the best results of all methods in bold. For coping with the space limitation, we shortened the names of some evaluation metrics, e.g., Hits@10 is shortened as H@10. The notations are the same in all tables.

	NELL								Wiki							
	Seen relations				Unseen relations				Seen relations				Unseen relations			
Model	MRR	H@10	H@5	H@1	MRR	H@10	H@5	H@1	MRR	H@10	H@5	H@1	MRR	H@10	H@5	H@1
TransE	.254	.475	.284	.158	.101	.195	.141	.043	305	.464	.378	.267	.033	.052	.041	.022
DistMult	.235	.426	.256	.147	.095	.177	.125	.065	.285	424	.357	.221	.050	.102	.069	.019
ComplEx	.289	.453	.285	.215	.131	.223	.156	.086	.324	.468	.381	.295	.069	.122	.089	.036
RotatE	.314	**.482**	.392	.226	.103	.235	.188	.089	.337	.481	.408	.299	.055	.083	.055	.033
UniKER	.299	.463	.390	.232	.107	.230	.176	.075	.321	.480	.373	.290	.051	.101	.053	.039
CoKE	.289	.466	.384	.235	.082	.155	.092	.037	.322	.477	.395	.280	.042	.051	.032	.024
GMatching	.181	.295	.261	.131	.175	.293	.250	.114	.269	.388	.341	.205	.201	.335	.272	.123
MetaR	.231	.384	.291	.175	.172	295	.236	.096	.324	.420	.390	.281	.193	.291	.237	.155
FSRL	.181	.322	.219	.103	.152	.321	.227	.109	.161	.298	.212	.103	.197	.318	.255	.119
FAAN	.268	.421	.357	.202	.176	.310	.244	.110	.321	.466	.395	.281	.239	.380	.309	.170
GANA	.242	.389	.299	.193	.176	.317	.247	.112	.320	.435	.388	.270	.223	.370	.262	.155
Eureka (Ours)	**.332**	**.482**	**.407**	**.267**	**.188**	**.340**	**.261**	**.122**	**.339**	**.498**	**.412**	**.305**	**.257**	**.397**	**.323**	**.192**

Table 3: Results of model variants on unseen relations of NELL. The best results are highlighted in bold.

Model	MRR	H@10	H@5	H@1
AS_1.1.1	.103	.201	.147	.055
AS_1.1.2	.114	.228	.162	.061
AS_1.2.1	.108	.249	.151	.058
AS_1.2.2	.135	.261	.197	.105
AS_2	.179	.330	.250	.118
AS_3.1	.157	.295	.231	.112
AS_3.2	.160	.304	.227	.115
Eureka	**.188**	**.340**	**.261**	**.122**

relation dataset. The experimental results show that Eureka surpasses the prior state-of-the-art few-shot learning approaches, which demonstrates that Eureka is able to remember the original knowledge learned from seen relations when having acquired unseen relations. Moreover, Eureka is still competitive compared with best-performed knowledge embedding models on both NELL and Wiki. It is also worth noting that the performance of the few-shot learning model does not improve significantly with sufficient training examples as other models do. It indicates that the learning capacity of the few-shot learning approaches is limited even though the number of their training examples increases.

Thus, we have so far answered the first question; i.e., Eureka can be well adapted into the KGR task of unseen relations and outperform both embedding models and few-shot learning models by incorporating the knowledge learned from the seen relation embeddings without sacrificing performance on seen relations.

5.6 Ablation Study

In this section, we inspect effectiveness of the model components. Experimental results of model variants is shown in Table 3 :

1) We verify the significance of the transformer encoder. We remove the transformer encoder and replace it with TransE and RotatE, respectively. We conduct two group of experiments with TransE as an alternative of the transformer encoder; i.e., AS_1.1.1 and AS_1.1.2. In AS_1.1.1, the entity and relation embeddings produced in the prior knowledge learning stage remain static when used in the trigger learning stage. In AS_1.1.2, these entity and relation embeddings are fine-tuned in the trigger learning stage. AS_1.2.1 and AS_1.2.2 share the same settings with RotatE as an alternative of the transformer encoder. Experimental results demonstrate that the transformer encoder is an essential component of our model due to its ability to model the semantic interactions between entities and relations and produce the dynamic embeddings. By comparing the results of AS_1.1.1 with AS_1.1.2, we can find that it is better to allow dynamic embeddings rather than static embeddings for the encoder of Eureka since Eureka absorbs new knowledge (unseen relation triples) in the trigger learning stage and could adjust embeddings of seen relation triples to adapt the unseen domain. The comparison between AS_1.2.1 and AS_1.2.2 also leads to the same conclusion.

2) We analyze the contribution of the CNN decoder. We remove the CNN decoder and this makes Eureka in the prior training stage degenerate into a solely transformer-based model similar to CoKE. Experimental results indicate that the model per-

formance can slightly benefit from the CNN-based decoder through modeling the high-level representations and the hierarchical structures of the KGs.

3) **(AS_3)** We evaluate the effectiveness of CDA. We conduct two experiments denoted as AS_3.1, AS_3.2. In AS_3.1, we simply remove CDA to inspect the contribution of seen relations; in AS_3.2, we replace CDA with an average pooling operation on seen relations to evaluate the effectiveness of the attention mechanism. Experimental results show that both unseen relations and attention mechanism are important and contribute consistent improvements to Eureka.

5.7 Case Study for CDA

We investigate how CDA works. Since CDA adopts an attention mechanism to model the interactions between seen and unseen relations and represents unseen relations with the acquired knowledge from seen relations. We randomly select four unseen relations and present their most relevant seen relations according to the attention score in CDA as shown in Figure 3. We could find that the relevant seen relations have semantic similarity with their corresponding unseen relations. For example, for the unseen relation *AnimalSuchAsInvertebrate*, the top 3 of the selected seen relations are *AnimalSuchAsMollusk*, *AnimalTypeHasAnimal*, *AnimalPreySon*. Obviously, most mollusks belong to invertebrate animals, which confirms the effectiveness of CDA and our assumption that unseen relations can benefit from relevant semantic interactions.

5.8 Discussions

We summarize the answers to our three research issues: (1) Eureka surpasses both embedding models and few-shot learning models on seen and unseen relations. Eureka achieves better performance on unseen relations without sacrificing its performance on seen relations. Note that as the number of triggers increases, Eureka still outperforms other baselines, which is shown in Figure 1. (2) The ablation study demonstrates the effectiveness of each model variant of Eureka, i.e., the transformer encoder for allowing dynamic embeddings; CNN decoder for modeling the high-level representations and the hierarchical structures of the KGs; CDA for modeling semantic interactions between seen and unseen relations. (3) The case study shows why CDA brings a dramatic rise for experimental results, i.e., CDA assigns varying attention weights to different seen relations and selects the most relevant seen relations to represent each unseen relation more accurately.

6 Conclusion

In this work, we present a neural insight learning framework (Eureka), which mimics human insight learning modes to bridge the "seen" to "unseen" gap in the KGR tasks. We train Eureka in prior knowledge learning and trigger learning stages. Specifically, Eureka acquires the representations of entities and seen relations in the prior knowledge learning stage, and then learns the unseen relations efficiently through a CDA network with the incorporation of the embeddings of seen relations. Eureka meets our expectation of the model to not only have good performance on both relation types but also eliminate the need to retrain the original training datasets. The experimental results demonstrate that our model outperforms the state-of-the-art baselines on datasets of both seen and unseen relations. The case studies confirm the CDA network is empowered to select relevant seen relations to better represent unseen relations. We plan to investigate enhancing CDA with relations' text descriptions as future directions of this work.

Acknowledgements

This work was supported by National Natural Science Foundation of China (62072463, 71531012), Research Seed Funds of School of Interdisciplinary Studies of Renmin University of China, National Social Science Foundation of China (18ZDA309), and Opening Project of State Key Laboratory of Digital Publishing Technology of Founder Group. Xun Liang is the corresponding author of this paper.

References

Ivan K. Ash, Benjamin D. Jee, and Jennifer Wiley. 2012. Investigating insight as sudden learning. *J. Probl. Solving*, (2).

Antoine Bordes, Nicolas Usunier, Alberto García-Durán, Jason Weston, and Oksana Yakhnenko. 2013. Translating embeddings for modeling multi-relational data. In *Advances in Neural Information Processing Systems 26: 27th Annual Conference on Neural Information Processing Systems 2013. Proceedings of a meeting held December 5-8, 2013, Lake Tahoe, Nevada, United States*.

Mingyang Chen, Wen Zhang, Wei Zhang, Qiang Chen, and Huajun Chen. 2019. Meta relational learning

for few-shot link prediction in knowledge graphs. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*.

Kewei Cheng, Ziqing Yang, Ming Zhang, and Yizhou Sun. 2021. Uniker: A unified framework for combining embedding and definite horn rule reasoning for knowledge graph inference. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing, EMNLP 2021, Virtual Event / Punta Cana, Dominican Republic, 7-11 November, 2021*.

Tim Dettmers, Pasquale Minervini, Pontus Stenetorp, and Sebastian Riedel. 2018a. Convolutional 2d knowledge graph embeddings. In *Proceedings of the Thirty-Second AAAI Conference on Artificial Intelligence, (AAAI-18), the 30th innovative Applications of Artificial Intelligence (IAAI-18), and the 8th AAAI Symposium on Educational Advances in Artificial Intelligence (EAAI-18), New Orleans, Louisiana, USA, February 2-7, 2018*.

Tim Dettmers, Pasquale Minervini, Pontus Stenetorp, and Sebastian Riedel. 2018b. Convolutional 2d knowledge graph embeddings. In *Proceedings of the Thirty-Second AAAI Conference on Artificial Intelligence, (AAAI-18), the 30th innovative Applications of Artificial Intelligence (IAAI-18), and the 8th AAAI Symposium on Educational Advances in Artificial Intelligence (EAAI-18), New Orleans, Louisiana, USA, February 2-7, 2018*.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*.

Angela Fan, Thibaut Lavril, Edouard Grave, Armand Joulin, and Sainbayar Sukhbaatar. 2020. Addressing some limitations of transformers with feedback memory. *arXiv preprint arXiv:2002.09402*.

Chelsea Finn, Pieter Abbeel, and Sergey Levine. 2017. Model-agnostic meta-learning for fast adaptation of deep networks. In *Proceedings of the 34th International Conference on Machine Learning, ICML 2017, Sydney, NSW, Australia, 6-11 August 2017*, Proceedings of Machine Learning Research.

Xavier Glorot, Antoine Bordes, and Yoshua Bengio. 2011. Deep sparse rectifier neural networks. In *Proceedings of the Fourteenth International Conference on Artificial Intelligence and Statistics, AISTATS 2011, Fort Lauderdale, USA, April 11-13, 2011*, JMLR Proceedings.

Lingbing Guo, Zequn Sun, and Wei Hu. 2019. Learning to exploit long-term relational dependencies in knowledge graphs. In *Proceedings of the 36th International Conference on Machine Learning, ICML 2019, 9-15 June 2019, Long Beach, California, USA*, Proceedings of Machine Learning Research.

Michael Hahn. 2020. Theoretical limitations of self-attention in neural sequence models. *Transactions of the Association for Computational Linguistics*.

Timothy M Hospedales, Antreas Antoniou, Paul Micaelli, and Amos J Storkey. 2021. Meta-learning in neural networks: A survey. *IEEE Transactions on Pattern Analysis and Machine Intelligence*.

Chenping Hou and Zhi-Hua Zhou. 2018. One-pass learning with incremental and decremental features. *IEEE Trans. Pattern Anal. Mach. Intell.*, (11).

Zhiyi Jiang, Jianliang Gao, and Xinqi Lv. 2021. Metap: Meta pattern learning for one-shot knowledge graph completion. In *SIGIR '21: The 44th International ACM SIGIR Conference on Research and Development in Information Retrieval, Virtual Event, Canada, July 11-15, 2021*.

Diederik P. Kingma and Jimmy Ba. 2015. Adam: A method for stochastic optimization. In *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*.

Brenden M Lake, Ruslan Salakhutdinov, and Joshua B Tenenbaum. 2015. Human-level concept learning through probabilistic program induction. *Science*, (6266).

Tom M. Mitchell, William W. Cohen, Estevam R. Hruschka Jr., Partha Pratim Talukdar, Justin Betteridge, Andrew Carlson, Bhavana Dalvi Mishra, Matthew Gardner, Bryan Kisiel, Jayant Krishnamurthy, Ni Lao, Kathryn Mazaitis, Thahir Mohamed, Ndapandula Nakashole, Emmanouil A. Platanios, Alan Ritter, Mehdi Samadi, Burr Settles, Richard C. Wang, Derry Wijaya, Abhinav Gupta, Xinlei Chen, Abulhair Saparov, Malcolm Greaves, and Joel Welling. 2015. Never-ending learning. In *Proceedings of the Twenty-Ninth AAAI Conference on Artificial Intelligence, January 25-30, 2015, Austin, Texas, USA*.

Tsendsuren Munkhdalai and Hong Yu. 2017. Meta networks. In *Proceedings of the 34th International Conference on Machine Learning, ICML 2017, Sydney, NSW, Australia, 6-11 August 2017*, Proceedings of Machine Learning Research.

Dai Quoc Nguyen, Tu Dinh Nguyen, Dat Quoc Nguyen, and Dinh Phung. 2018. A novel embedding model for knowledge base completion based on convolutional neural network. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 2 (Short Papers)*.

Guanglin Niu, Yang Li, Chengguang Tang, Ruiying Geng, Jian Dai, Qiao Liu, Hao Wang, Jian Sun, Fei Huang, and Luo Si. 2021. Relational learning with

gated and attentive neighbor aggregator for few-shot knowledge graph completion. In *SIGIR '21: The 44th International ACM SIGIR Conference on Research and Development in Information Retrieval, Virtual Event, Canada, July 11-15, 2021*.

Sinno Jialin Pan and Qiang Yang. 2010. A survey on transfer learning. *IEEE Trans. Knowl. Data Eng.*, (10).

Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. 2019. Language models are unsupervised multitask learners. *OpenAI blog*, (8).

Jiawei Sheng, Shu Guo, Zhenyu Chen, Juwei Yue, Lihong Wang, Tingwen Liu, and Hongbo Xu. 2020. Adaptive Attentional Network for Few-Shot Knowledge Graph Completion. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*.

Jake Snell, Kevin Swersky, and Richard S. Zemel. 2017. Prototypical networks for few-shot learning. In *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, December 4-9, 2017, Long Beach, CA, USA*.

Zhiqing Sun, Zhi-Hong Deng, Jian-Yun Nie, and Jian Tang. 2019. Rotate: Knowledge graph embedding by relational rotation in complex space. In *7th International Conference on Learning Representations, ICLR 2019, New Orleans, LA, USA, May 6-9, 2019*.

Théo Trouillon, Johannes Welbl, Sebastian Riedel, Éric Gaussier, and Guillaume Bouchard. 2016. Complex embeddings for simple link prediction. In *Proceedings of the 33nd International Conference on Machine Learning, ICML 2016, New York City, NY, USA, June 19-24, 2016*, JMLR Workshop and Conference Proceedings.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, December 4-9, 2017, Long Beach, CA, USA*.

Denny Vrandečić and Markus Krötzsch. 2014. Wikidata: a free collaborative knowledgebase. *Communications of the ACM*, (10).

Quan Wang, Pingping Huang, Haifeng Wang, Songtai Dai, Wenbin Jiang, Jing Liu, Yajuan Lyu, Yong Zhu, and Hua Wu. 2019. Coke: Contextualized knowledge graph embedding. *arXiv preprint arXiv:1911.02168*.

Yaqing Wang, Quanming Yao, James T. Kwok, and Lionel M. Ni. 2020. Generalizing from a few examples: A survey on few-shot learning. *ACM Comput. Surv.*, (3).

Wenhan Xiong, Mo Yu, Shiyu Chang, Xiaoxiao Guo, and William Yang Wang. 2018. One-shot relational learning for knowledge graphs. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*.

Bishan Yang, Wen-tau Yih, Xiaodong He, Jianfeng Gao, and Li Deng. 2015. Embedding entities and relations for learning and inference in knowledge bases. In *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*.

Chuxu Zhang, Huaxiu Yao, Chao Huang, Meng Jiang, Zhenhui Li, and Nitesh V. Chawla. 2020. Few-shot knowledge graph completion. In *The Thirty-Fourth AAAI Conference on Artificial Intelligence, AAAI 2020, The Thirty-Second Innovative Applications of Artificial Intelligence Conference, IAAI 2020, The Tenth AAAI Symposium on Educational Advances in Artificial Intelligence, EAAI 2020, New York, NY, USA, February 7-12, 2020*.

Zhaoze Zhou, Wei-Shi Zheng, Jianfang Hu, Yong Xu, and Jane You. 2016. One-pass online learning: A local approach. *Pattern Recognit.*

CitRet: A Hybrid Model for Cited Text Span Retrieval

Amit Pandey[†*], **Avani Gupta**[*] **and Vikram Pudi**[†]
[†]Data Sciences and Analytics Center, Kohli Center on Intelligent Systems
International Institute of Information Technology, Hyderabad, India
{amit.pandey, avani.gupta}@research.iiit.ac.in
vikram@iiit.ac.in

Abstract

The paper aims to identify cited text spans in the reference paper related to the given citance in the citing paper. We refer to it as cited text span retrieval (CTSR). Most current methods attempt this task by relying on pre-trained off-the-shelf deep learning models like SciBERT. Though these models are pre-trained on large datasets, they underperform in out-of-domain settings. We introduce CitRet, a novel hybrid model for CTSR that leverages unique semantic and syntactic structural characteristics of scientific documents. This enables us to use significantly less data for finetuning. We use only 1040 documents for finetuning. Our model augments mildly-trained SBERT-based contextual embeddings with pre-trained non-contextual Word2Vec embeddings to calculate semantic textual similarity. We demonstrate the performance of our model on the CLSciSumm shared tasks. It improves the state-of-the-art results by over 15% on the F1 score evaluation.

1 Introduction

Citations are an integral part of scientific literature as they help better understand the relationships between scientific documents. Authors cite other papers to acknowledge their contributions, compare to their work, criticize, and improve upon their work. Citances often focus on the most important components of a scientific document. Moreover, citance-based summarization is also a widely studied field because it covers some insights that might not be present in abstract-based summarization (Elkiss et al., 2008).

However, a citance depends on the intention and opinion of the citing author and can be affected by epistemic value drift[1] (Cohan et al., 2015). Also, a citance in itself lacks sufficient details to capture the exact content of the referenced paper. Hence, identifying the correct context of the cited text can enable us to verify the biases (Zerva et al., 2020), overcome epistemic value drift, build dense knowledge graphs, and generate better summaries (Jaidka et al., 2019; Chandrasekaran et al., 2020). Furthermore, it also helps in qualitative analysis of the citations (Teufel et al., 2006). Motivated by these, research tasks and tracks such as BiomedSumm[2] and CLSciSumm lay significant emphasis on this fundamental and challenging problem of finding the exact cited text span. We refer to this task as cited text span retrieval (CTSR).

Most of the current methods targeting this problem are centered around fine-tuning deep neural networks. In this regard, transformer (Vaswani et al., 2017) based encoders such as BERT (Devlin et al., 2018) and SciBERT (Beltagy et al., 2019) have proven to be very effective and have outperformed standard baselines like LDA and TF-IDF. However, a major drawback of these methods is that they require large domain-specific datasets, often exceeding 1 million documents, to fine-tune.

This paper proposes CitRet, a hybrid CTSR model that performs well even in low-resourced domain-specific settings. We model the problem as a semantic textual similarity (STS) task. We exploit the distinctive semantic and syntactic structural characteristics of scientific literature, i.e., when a paper is cited, the cited text of the reference paper is often paraphrased in such a way that it still expresses the same central idea while also preserving certain keywords. Hence, we use these keywords, which are common to both the citance and the cited sentence, to find weighted contextual embeddings for the sentences. To find these weighted contextual embeddings, we use Sentence-BERT (SBERT) (Reimers and Gurevych, 2019) fine-tuned to minimize cosine similarity loss on training data. However, when the training data is scarce, these contextual embeddings fail to cap-

[*]Equal contribution

[1]An example of epistemic value drift is citing a claim as a fact.

[2]http://www.nist.gov/tac/2014/BiomedSumm/

Proceedings of the 29th International Conference on Computational Linguistics, pages 4528–4536
October 12–17, 2022.

ture out-of-domain knowledge. To overcome this, we further leverage pre-trained non-contextual embeddings like Word2Vec (Mikolov et al., 2013) to capture the general domain knowledge. We use Word Mover's Distance (WMD) (Kusner et al., 2015) to find (dis)similarity scores based on these non-contextual embeddings. This hybrid approach of utilizing contextual and non-contextual embeddings enables CitRet to generalize well over unseen datasets. Definitions of the terms used throughout the paper are:

Reference paper (RP): A scientific document of which one or more sentences have been cited by another paper(s). **Citing paper (CP)**: A document that contains one or multiple citations to an RP. **Citance**: A sentence in CP that contains the reference to the RP. **Cited sentence**: The exact piece of the text belonging to the RP that a citance refers to. **Cited text span**: Span of the cited sentence(s) belonging to the RP corresponding to a citance.

The major contributions of this work are: 1) Proposing a simple yet effective CTSR model that requires less data for fine-tuning and is computationally inexpensive. We train only on the CL-SciSumm training dataset that consists of 40 manually annotated articles and 1000 automatically annotated articles. 2) Advancing the state-of-the-art (SOTA) to identify cited text span by over 15%. 3) Empirically validating the advantage of using the semantic and syntactic structure for CTSR.

2 Related Work

The task of CTSR requires modeling the relationship (similarity) between a citing and a candidate cited sentence. Early systems proposed using features based on TF–IDF (Yeh et al., 2017; Cao et al., 2016; Prasad, 2017) and n-grams or sentence graph overlap (Aggarwal and Sharma, 2016; Klampfl et al., 2016) in order to calculate similarity scores between the citing sentence and candidate sentences. Similarity measures such as Jaccard similarity and cosine similarity were commonly used to solve this task. (Bravo et al., 2018; Debnath et al., 2018; Kim and Ou, 2019; Pitarch et al., 2019). The problem has also been posed as a binary classification problem in Davoodi et al. (2018); Yeh et al. (2017); Zerva et al. (2020). In addition to traditional features such as TF-IDF and n-grams, prior methods have also proposed using learned distributed vector space representation (word embeddings) based features since they contain the semantic similarity information at the word level. Models using both non-contextual embeddings such as Word2Vec and contextual embedding methods like BERT have been utilized to find these word embeddings. These extracted features are further used as an input to machine learning algorithms like SVM (Ma et al., 2018), random forests (Wang et al., 2018), Word Mover's Distance (Li et al., 2018), CNN (Li et al., 2019; AbuRa'ed et al., 2018) or XGBoost (Syed et al., 2019; Pitarch et al., 2019). Furthermore, many approaches even adopted voting mechanisms and ensemble techniques on top of their models to improve their metrics (Chai et al., 2020; Wang et al., 2018; Ma et al., 2018, 2019; Quatra et al., 2019). The current best performing models exploit transformers fine-tuned on very large datasets (Chai et al., 2020; Zerva et al., 2019). Chai et al. (2020) also experimented with adding document level features to the model using special tokens. Other noteworthy approaches, like Aumiller et al. (2020) formulated the task as a search problem and used a two-step approach for retrieving relevant sentences for a given citation. They first find candidate sentences using Apache Solr and BM25 and then re-rank the retrieved sentences using a computationally expensive BERT-based re-ranker.

CTSR as Semantic Textual Similarity: We model the problem as a semantic textual similarity (STS) task. To this end, learning sentence embeddings, instead of word embeddings, has shown promise and improvement in performance (Reimers and Gurevych, 2019). Using pooling strategies such as mean or max pooling of word embeddings has proven to be an efficient way of obtaining sentence embeddings. SBERT (Reimers and Gurevych, 2019) by default uses mean pooling. Chen et al. (2018) further explored generalized pooling strategies to enhance sentence embeddings. CNN-based models have also been used to encode sentences into fixed length vectors (Jiao et al., 2018). To improve performance on sentence matching tasks, Liu et al. (2020) proposed syntax- and semantics-aware BERT(SS-BERT), which implicitly integrates syntactic and semantic information of sentences. Unnam et al. (2022) showed that sentence embeddings could be further improved by employing principal component removal based denoising as a post-processing step.

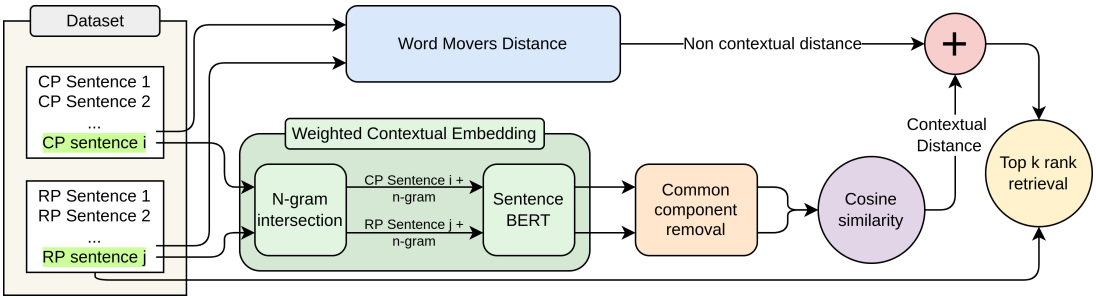

Figure 1: Illustration of the CitRet model. WMD and weighted contextual embeddings (WCEs) are calculated for an input pair. The WCEs are then denoised using the common component removal technique. These denoised WCEs are used to find cosine similarity between the sentences of the input pair. Finally, WMD and cosine scores are added, and top k similar sentences in an RP for a citance are retrieved.

3 Methodology

We formulate this task of CTSR as finding semantic textual similarity between a citance and all the sentences of an RP, i.e., to find the cited text span for a given citance, we pick the top k similar sentences in the RP. We refer to a <citance, a sentence in the RP> pair as an *input pair*. As shown in Figure 1, an input pair is first pre-processed by lowercasing the tokens, removing the stop words, and removing the special characters. Then to find the final similarity scores, CitRet employs a mix of cosine scores using weighted contextual embeddings (contextual distance) and Word Mover's Distance scores (non-contextual distance) using pre-trained non-contextual embeddings. Now, we explain each component of the pipeline in detail.

3.1 Contextual Distance

Contextual distance between the sentences is calculated using contextual sentence embeddings. The proposed model uses finetuned SBERT to learn these contextual embeddings for an input pair. SBERT returns a fixed-length dense vector for an input sentence (sentence embedding), irrespective of the length of the input sentence[3]. To yield the final sentence embeddings, CitRet follows three steps: 1) Finetuning the SBERT, 2) Finding the weighted contextual embeddings for each sentence pair, and 3) Denoising the embeddings.

3.1.1 Finetuning the SBERT

To finetune SBERT siamese networks, we use cosine similarity loss. As training examples, we pass sentence pairs annotated with cosine similarity scores on a scale of 0 to 1. For each citance, we pass 5 sentence pairs of 3 different types, i.e., one pair with the actual cited text having a similarity score of 1, two pairs with randomly selected sentences from other RP having a similarity score of 0, and two pairs with randomly selected sentences belonging to the same RP having similarity score of 0.3. This helps us model relations between the sentences of the same documents and sentences of different documents.

3.1.2 Weighted contextual embeddings

When an RP is cited, the information that can be extracted from a citance about the RP depends upon the intention, and the opinion of the citing author(s) (Zerva et al., 2020). However, when the cited sentences are referred to, some key ideas and keywords are preserved, as depicted in Figure 2. CitRet exploits this characteristic of the scientific documents to find weighted contextual embeddings for the input pair.

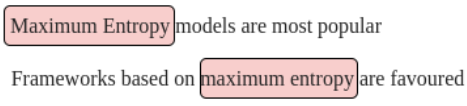

Figure 2: n-gram intersection of two sentences

SBERT takes the mean of all the word embeddings to calculate the sentence embedding. After fine-tuning SBERT on domain-specific data, it is able to learn contextual embedding for a sentence. To leverage this contextual learning capability of SBERT and to find weighted contextual embeddings (WCEs) for the sentences, we use a very simple and intuitive strategy of concatenating the common keywords to the input pair before passing it to the SBERT (we concatenate the keyword to both the sentences of the pair). These keywords are extracted by finding common n-gram intersections between the sentences of the input pair. In the example shown in Figure 2, *maximum entropy* is the common keyword (bigram). Concatenating these

[3]Please refer to Appendix (A.1) for more details.

n-grams results in the common keywords having more weight in the sentence embeddings due to the mean pooling operation. Therefore, the sentence embedding vectors of the pair come closer in the dense vector space if they share some keywords. Here, number *n* can be optimized empirically, and in our tests, we get the best results for bigrams.

3.1.3 Denoising

We further modify the WCEs that we get from the previous step by using a denoising technique adapted from *piecewise common component removal* method proposed in Ethayarajh (2018). Here, the common components refer to the common topics (discourse themes) that exist throughout the document (RP and CP) and can be considered as noise. Thus, removing these common components can be understood as downgrading the unimportant components (common discourse) and focusing on the components that have more discriminatory power. This helps in denoising the embeddings (Arora et al., 2017). Since cosine-similarity treats all dimensions equally (Reimers and Gurevych, 2019), denoising becomes critical in making it more focused. Consequently, the cosine similarity scores calculated using denoised embeddings become more relevant (Arora et al., 2017).

$$\tilde{v} = v - \sum_{i}^{m} \lambda_i \, proj_{pc_i} \, v \quad \text{, where } \lambda_i = \frac{\sigma_i^2}{\sum_{j=1}^{m} \sigma_j^2}$$

These common discourse vectors are estimated as the principal components for a set of WCEs. These principal components are calculated by singular value decomposition of $A_{l \times d}$ matrix, where l is the number of sentences in the document (RP and CP), and d is the dimension of the WCEs. To get the final denoised sentence vector $\tilde{v}$, we subtract from the original sentence vector v, the weighted sum of the projections of the vector v on the first $m(=3)$ principal components $pc_{i..m}$. The projections $proj_{pc_i} \, v$ are weighted by λ_i, where λ_i is the proportion of variance σ_i (singular value) captured by the principal component pc_i.

3.2 Non-contextual Distance

CitRet uses both the supervised and unsupervised techniques to calculate the final similarity scores to generalize well over unseen datasets. It augments contextual distance calculated using mildly-trained SBERT with non-contextual distance calculated using unsupervised WMD technique[4]. Arora et al. (2017) and Reimers and Gurevych (2019) note that even simple techniques such as computing the average of pre-trained embeddings can outperform sophisticated techniques such as BERT in unsupervised textual similarity tasks. As discussed, a cited text is usually paraphrased around a keyword in such a way that it still expresses the same central idea. Since WMD uses the high-quality Word2Vec model embeddings having a vocabulary size of 3 million, it can capture knowledge related to these general domain words that fine-tuning a deep learning model with low training data might not be able to extract (Kusner et al., 2015). Figure 3 demonstrates WMD's ability to capture relations in the general domain setting. The arrows represent the flow between two words of an input pair. It may be observed how *models* flows to *frameworks* and *popular* to *favoured*. It can be noted that the words *popular* and *favoured* are general domain words (non-scientific terms) and might not appear very frequently in a scarce domain-specific dataset. Hence, the semantic relationships between these general domain words are better captured by WMD.

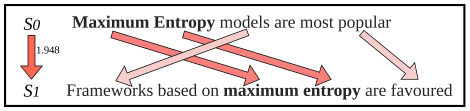

Figure 3: Flow between 2 sentences S_0 and S_1 using WMD

4 Experiments and Results

We demonstrate the performance of the proposed method on CL-SciSumm shared task (Jaidka et al., 2019; Chandrasekaran et al., 2019, 2020) task 1(a), where for each citance, we need to identify the spans of text (cited text spans) in the RP that most accurately reflect the citance. These cited text spans range from the granularity of a sentence fragment to several consecutive sentences. We pick top $k = 3$ similar candidate cited sentences for a given citance. CitRet is trained only on the CL-SciSumm training dataset that consists of 40 manually annotated articles and 1000 low quality document sets that were automatically annotated using neural networks. We do not use any external corpora to fine-tune our model. We evaluate our model's performance against gold label annotations for the CL-SciSumm test set of 20 documents.

[4]Please refer to Appendix A.2 for a detailed explanation of WMD.

Method	Recall	Precision	F1
ACL 2018	-	-	0.126
BERT 2018/19 OV+2018FT	-	-	0.120
SciBERT 2018	-	-	0.078
SciBERT-SemBERT	**0.2459**	0.1318	0.1716
SciBer-ACLBERT	0.2265	0.1244	0.1606
SBERT†	0.1879	0.1023	0.1325
SBERT + WCE†	0.1815	0.1647	0.1727
Denoising (SBERT+WCE+D)†	0.1901	0.1724	0.1808
CitRet (SBERT+WCE+D+WMD)†	0.2080	**0.1888**	**0.1979**

Table 1: Performance comparison of our model with the baseline models. The last 4 rows show the ablation study of our model marked with †. D denotes denoising step.

We consider the SOTA models of 2019 and 2020 CL-SciSumm tasks as baselines. Table 1 shows that CitRet performs the best in quantitative metrics (F1 and Precision) and outperforms 2019 SOTA (*ACL 2018*) by over 57% and 2020 SOTA (*SciBERT-SemBERT*) by over 15% on F1 score evaluation. It can be noted that using just the *SBERT + WCE* component outperformed all the baseline SOTA models that use much larger datasets (exceeding 1 million) for finetuning[5]. This empirically validates that using the semantic and syntactic structure for CTSR can significantly improve the results. It should be noted that *Denoising* and *WMD* further improve the performance.

5 Discussion

As can be observed from Table 1, the proposed method significantly improves the F1 score (+15%) and Precision(+43%) with some loss in Recall(15%). Our approach focuses on Precision (a measure of the quality of retrieval) over Recall (a measure of quantity) because, for the given task, the probability of getting false positives is very high. Hence a higher precision results in a more concise and accurate summarization.

The proposed approach is in line with the recommendation made by the task organisers to exploit the structural and semantic characteristics that are unique to scientific documents to enrich the embeddings. The paper proposes a simple and computationally inexpensive alternative to the current state-of-art model in the form of CitRet. It leverages both contextual and non-contextual embeddings. CitRet also combines a supervised model and an unsupervised model. This hybrid architecture provides performance and robustness against noisy training samples. The components of the model are lightweight (do not require extensive fine-tuning), faster, explainable, and intuitive. This highlights how other statistical machine learning techniques can be leveraged along with modern deep neural network architectures to compensate for the lack of quality training data and outperform computationally expensive architectures.

It may also be noted that while our method beats the baselines by large margins and achieves a new SOTA, the absolute values are still rather low because of the non-triviality of the task. The task becomes particularly challenging because of the low-quality training data and subjectivity of the annotators. Hence, we believe that there is a scope for further improvement, and the problem demands greater exploration.

6 Conclusion

In this paper, we propose CitRet, a novel model for cited text span retrieval. CitRet outperforms the current SOTA models by significant margins (15% F1). The proposed model is quite simple, computationally inexpensive, improves generalization, and does not require any large external datasets to fine-tune. However, considering the non-triviality of the task, this paper proposes a new approach for further exploration of the task.

Acknowledgments

We thank IHub-Data, IIIT Hyderabad[6] for financial support. We thank Mr. Narendra Babu Unnam, DSAC, IIIT Hyderabad, for his comments that greatly improved the manuscript.

References

Ahmed Ghassan Tawfiq AbuRa'ed, Àlex Bravo Serrano, Luis Chiruzzo, and Horacio Saggion. 2018. Lastus/taln+ inco@ cl-scisumm 2018-using regression and convolutions for cross-document semantic linking and summarization of scholarly literature. In *Mayr P, Chandrasekaran MK, Jaidka K, editors. BIRNDL 2018. 3rd Joint Workshop on Bibliometric-enhanced Information Retrieval and Natural Language Processing for Digital Libraries; 2018 Jul 21; Ann Arbor, MI.[place unknown]: CEUR; 2018. p. 150-63.* CEUR Workshop Proceedings.

Peeyush Aggarwal and Richa Sharma. 2016. Lexical and syntactic cues to identify reference scope of citance. In *Proceedings of the joint workshop on bibliometric-enhanced information retrieval and*

[5]Please refer to Appendix B for details of the experimental setup of the baseline models and ablation study analysis.

[6]https://ihub-data.iiit.ac.in/

natural language processing for digital libraries (BIRNDL), pages 103–112.

Waleed Ammar, Dirk Groeneveld, Chandra Bhagavatula, Iz Beltagy, Miles Crawford, Doug Downey, Jason Dunkelberger, Ahmed Elgohary, Sergey Feldman, Vu Ha, Rodney Kinney, Sebastian Kohlmeier, Kyle Lo, Tyler Murray, Hsu-Han Ooi, Matthew Peters, Joanna Power, Sam Skjonsberg, Lucy Lu Wang, Chris Wilhelm, Zheng Yuan, Madeleine van Zuylen, and Oren Etzioni. 2018. Construction of the literature graph in semantic scholar.

Sanjeev Arora, Yingyu Liang, and Tengyu Ma. 2017. A simple but tough-to-beat baseline for sentence embeddings. In *International conference on learning representations*.

Dennis Aumiller, Satya Almasian, Philip Hausner, and Michael Gertz. 2020. UniHD@CL-SciSumm 2020: Citation extraction as search. In *Proceedings of the First Workshop on Scholarly Document Processing*, pages 261–269, Online. Association for Computational Linguistics.

Iz Beltagy, Kyle Lo, and Arman Cohan. 2019. Scibert: A pretrained language model for scientific text.

Àlex Bravo, Luis Chiruzzo, Horacio Saggion, et al. 2018. Lastus/taln+ inco@ cl-scisumm 2018-using regression and convolutions for cross-document semantic linking and summarization of scholarly literature. In *BIRNDL@ SIGIR*.

Ziqiang Cao, Wenjie Li, and Dapeng Wu. 2016. Polyu at cl-scisumm 2016. In *Proceedings of the joint workshop on bibliometric-enhanced information retrieval and natural language processing for digital libraries (BIRNDL)*, pages 132–138.

Ling Chai, Guizhen Fu, and Yuan Ni. 2020. Nlp-pingantech@ cl-scisumm 2020. In *Proceedings of the First Workshop on Scholarly Document Processing*, pages 235–241.

Muthu Kumar Chandrasekaran, Guy Feigenblat, Eduard Hovy, Abhilasha Ravichander, Michal Shmueli-Scheuer, and Anita de Waard. 2020. Overview and insights from the shared tasks at scholarly document processing 2020: CL-SciSumm, LaySumm and LongSumm. In *Proceedings of the First Workshop on Scholarly Document Processing*, pages 214–224, Online. Association for Computational Linguistics.

Muthu Kumar Chandrasekaran, Michihiro Yasunaga, Dragomir Radev, Dayne Freitag, and Min-Yen Kan. 2019. Overview and results: Cl-scisumm shared task 2019.

Qian Chen, Zhen-Hua Ling, and Xiaodan Zhu. 2018. Enhancing sentence embedding with generalized pooling. In *Proceedings of the 27th International Conference on Computational Linguistics*, pages 1815–1826.

Arman Cohan, Luca Soldaini, and Nazli Goharian. 2015. Matching citation text and cited spans in biomedical literature: a search-oriented approach. In *proceedings of the 2015 conference of the North American Chapter of the association for computational linguistics: Human language technologies*, pages 1042–1048.

Elnaz Davoodi, Kanika Madan, and Jia Gu. 2018. Clscisumm shared task: On the contribution of similarity measure and natural language processing features for citing problem. In *BIRNDL@ SIGIR*.

Dipanwita Debnath, Amika Achom, and Partha Pakray. 2018. Nlp-nitmz@ clscisumm-18. In *BIRNDL@ SIGIR*, pages 164–171.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*.

Aaron Elkiss, Siwei Shen, Anthony Fader, Güneş Erkan, David States, and Dragomir Radev. 2008. Blind men and elephants: What do citation summaries tell us about a research article? *Journal of the American Society for Information Science and Technology*, 59(1):51–62.

Kawin Ethayarajh. 2018. Unsupervised random walk sentence embeddings: A strong but simple baseline. In *Rep4NLP@ACL*.

Kokil Jaidka, Michihiro Yasunaga, Muthu Kumar Chandrasekaran, Dragomir Radev, and Min-Yen Kan. 2019. The cl-scisumm shared task 2018: Results and key insights.

Xiaoqi Jiao, Fang Wang, and Dan Feng. 2018. Convolutional neural network for universal sentence embeddings. In *Proceedings of the 27th International Conference on Computational Linguistics*, pages 2470–2481, Santa Fe, New Mexico, USA. Association for Computational Linguistics.

Hyonil Kim and Shiyan Ou. 2019. Nju@cl-scisumm-19. In *BIRNDL@SIGIR*, volume 2414 of *CEUR Workshop Proceedings*, pages 247–255. CEUR-WS.org.

Stefan Klampfl, Andi Rexha, and Roman Kern. 2016. Identifying referenced text in scientific publications by summarisation and classification techniques. In *Proceedings of the joint workshop on bibliometric-enhanced information retrieval and natural language processing for digital libraries (BIRNDL)*, pages 122–131.

Matt Kusner, Yu Sun, Nicholas Kolkin, and Kilian Weinberger. 2015. From word embeddings to document distances. In *International conference on machine learning*, pages 957–966. PMLR.

Lei Li, Junqi Chi, Moye Chen, Zuying Huang, Yingqi Zhu, and Xiangling Fu. 2018. Cist@ clscisumm-18: Methods for computational linguistics scientific citation linkage, facet classification and summarization. In *BIRNDL@ SIGIR*.

Lei Li, Yingqi Zhu, Yang Xie, Zuying Huang, Wei Liu, Xingyuan Li, and Yinan Liu. 2019. Cist@clscisumm-19: Automatic scientific paper summarization with citances and facets. In *BIRNDL@SIGIR*, volume 2414 of *CEUR Workshop Proceedings*, pages 196–207. CEUR-WS.org.

Tao Liu, Xin Wang, Chengguo Lv, Ranran Zhen, and Guohong Fu. 2020. Sentence matching with syntax- and semantics-aware bert. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 3302–3312.

Shutian Ma, Heng Zhang, Jin Xu, and Chengzhi Zhang. 2018. Njust@ clscisumm-18. In *BIRNDL@ SIGIR*.

Shutian Ma, Heng Zhang, Tianxiang Xu, Jin Xu, Shaohu Hu, and Chengzhi Zhang. 2019. Ir&tm-njust @ clscisumm-19. In *BIRNDL@SIGIR*, volume 2414 of *CEUR Workshop Proceedings*, pages 181–195. CEUR-WS.org.

Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013. Efficient estimation of word representations in vector space. *arXiv preprint arXiv:1301.3781*.

Ofir Pele and Michael Werman. 2008. A linear time histogram metric for improved sift matching. In *European conference on computer vision*, pages 495–508. Springer.

Ofir Pele and Michael Werman. 2009. Fast and robust earth mover's distances. In *2009 IEEE 12th international conference on computer vision*, pages 460–467. IEEE.

Yoann Pitarch, Karen Pinel-Sauvagnat, Gilles Hubert, Guillaume Cabanac, and Ophélie Fraisier-Vannier. 2019. IRIT-IRIS at cl-scisumm 2019: Matching citances with their intended reference text spans from the scientific literature. In *BIRNDL@SIGIR*, volume 2414 of *CEUR Workshop Proceedings*, pages 208–213. CEUR-WS.org.

Animesh Prasad. 2017. Wing-nus at cl-scisumm 2017: Learning from syntactic and semantic similarity for citation contextualization. In *BIRNDL@ SIGIR (2)*.

Moreno La Quatra, Luca Cagliero, and Elena Baralis. 2019. Poli2sum@cl-scisumm-19: Identify, classify, and summarize cited text spans by means of ensembles of supervised models. In *BIRNDL@SIGIR*, volume 2414 of *CEUR Workshop Proceedings*, pages 233–246. CEUR-WS.org.

Dragomir R Radev, Pradeep Muthukrishnan, Vahed Qazvinian, and Amjad Abu-Jbara. 2013. The acl anthology network corpus. *Language Resources and Evaluation*, 47(4):919–944.

Nils Reimers and Iryna Gurevych. 2019. Sentence-bert: Sentence embeddings using siamese bert-networks. *arXiv preprint arXiv:1908.10084*.

Bakhtiyar Syed, Vijayasaradhi Indurthi, Balaji Vasan Srinivasan, and Vasudeva Varma. 2019. Helium @ cl-scisumm-19 : Transfer learning for effective scientific research comprehension. In *BIRNDL@SIGIR*, volume 2414 of *CEUR Workshop Proceedings*, pages 214–223. CEUR-WS.org.

Simone Teufel, Advaith Siddharthan, and Dan Tidhar. 2006. Automatic classification of citation function. In *Proceedings of the 2006 Conference on Empirical Methods in Natural Language Processing*, pages 103–110, Sydney, Australia. Association for Computational Linguistics.

Narendra Babu Unnam, Krishna Reddy, Amit Pandey, and Naresh Manwani. 2022. Journey to the center of the words: Word weighting scheme based on the geometry of word embeddings. In *34th International Conference on Scientific and Statistical Database Management*, SSDBM 2022, New York, NY, USA. Association for Computing Machinery.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need.

Pancheng Wang, Shasha Li, Ting Wang, Haifang Zhou, and Jintao Tang. 2018. Nudt@ clscisumm-18. In *BIRNDL@ SIGIR*.

Jen-Yuan Yeh, Tien-Yu Hsu, Cheng-Jung Tsai, and Pei-Cheng Cheng. 2017. Reference scope identification for citances by classification with text similarity measures. In *proceedings of the 6th international conference on software and computer applications*, pages 87–91.

Chrysoula Zerva, Minh-Quoc Nghiem, Nhung T. H. Nguyen, and Sophia Ananiadou. 2019. Nactem-uom @ cl-scisumm 2019. In *BIRNDL@SIGIR*, volume 2414 of *CEUR Workshop Proceedings*, pages 167–180. CEUR-WS.org.

Chrysoula Zerva, Minh-Quoc Nghiem, Nhung T. H. Nguyen, and Sophia Ananiadou. 2020. Cited text span identification for scientific summarisation using pre-trained encoders. *Scientometrics*, 125:3109–3137.

Zhuosheng Zhang, Yuwei Wu, Hai Zhao, Zuchao Li, Shuailiang Zhang, Xi Zhou, and Xiang Zhou. 2020. Semantics-aware bert for language understanding.

A Background

A.1 SBERT

Sentence-BERT (SBERT) (Reimers and Gurevych, 2019) , is a modification of the pre-trained BERT (Bidirectional Encoder Representations from Transformers) (Devlin et al., 2018) model. BERT is a popular attention mechanism-based model that takes a sentence (an arbitrary sequence of tokens)

as an input and learns contextual embeddings for each token in the sentence. Though BERT has achieved state-of-the-art performance in a wide variety of NLP tasks, its design renders it inappropriate for semantic similarity search and unsupervised tasks because BERT doesn't compute independent sentence embeddings and instead learns embeddings for each token of the sentence.

To overcome this problem, SBERT builds over the BERT's innovation of using a bidirectional encoder. SBERT leverages BERT-based siamese network architecture to embed sentences into a fixed-length vector by adding a pooling layer on top of the BERT layer. The SBERT siamese network architecture can be fined tuned using different losses such as triplet loss, contrastive loss, and cosine similarity loss. Moreover, SBERT is computationally inexpensive compared to BERT (Reimers and Gurevych, 2019).

A.2 Word Mover's Distance

Given pre-trained embeddings for the words, Word Mover's Distance (WMD) (Kusner et al., 2015) measures the distance between a pair of sentences (sequence of words). It exploits the underlying geometry of the word embeddings to represent a sentence as a weighted point cloud in the word embedding space. It formulates the problem of finding distance between two sentences as a transportation problem based on Earth Mover's Distance. It defines the dissimilarity between two sentences as the minimum amount of work (distance traveled) required to transport words from one sentence to the words of another sentence in the word embedding space. This minimum cumulative travel cost between words of two sentences is calculated by solving the following linear optimization problem.

$$\min_{T \geq 0} \sum_{i,j=1}^{n} T_{ij} c(i,j)$$

$$\text{subject to:} \sum_{j=1}^{n} T_{ij} = s_i \quad \forall i \in \{1, ..., n\}$$

$$\sum_{i=1}^{n} T_{ij} = s'_j \quad \forall j \in \{1, ..., n\}$$

Here, s and s' are the normalized bag-of-words representation of two sentences. T is a flow matrix, where the $T_{ij} \geq 0$ entry indicates how much of word i in sentence s travels to word j in sentence s'. The total outgoing flow from a word i in sentence s to all the words j in sentence s' equals to the normalised frequency of word i, i.e ($\sum_{ij} T_{ij} = s_i$). The distance between two words in the embedding space is given by $c(i, j)$ and calculated using euclidean distance between the word embeddings. The final distance between two sentences is $\sum_{ij} T_{ij} c(i, j)$.

B Detailed Experimental Setup and Analysis

We demonstrate the performance of the proposed method on CL-SciSumm shared task (Jaidka et al., 2019; Chandrasekaran et al., 2019, 2020) task 1(a), where for each citance, we need to identify the spans of text (cited text spans) in the RP that most accurately reflect the citance. These cited text spans are of the granularity of a sentence fragment, a full sentence, or several consecutive sentences (no more than 5). For this, we pick top k (we picked $k = 3$) semantically similar candidate cited sentences for a given citance by sorting their similarity scores. We evaluate the predictions against gold label annotations using F1 score.

We compare the performance of the proposed model CitRet, with the best 3 systems (of each category) submitted by NaCTeM-UoM (Zerva et al., 2019) and the best 2 systems submitted by team NLP-PINGAN-TECH (Chai et al., 2020), over CL-SciSumm test set.

The systems submitted by NaCTeM-UoM are based on BERT. Along with *BERT 2018/19 OV + 2018 FT* (a BERT model fine-tuned on the CL-SciSumm 2018-2019 dataset), they submitted models *ACL 2018* and *SciBERT 2018*. Both these models are first trained on significantly large domain-specific corpora and then fine-tuned on CL-SciSumm dataset. *ACL 2018* is trained ACL-ARC (Radev et al., 2013) whereas *SciBERT 2018* is based on SciBERT model (Beltagy et al., 2019), which is pre-trained on collection of 1.14M documents from Semantic Scholar (Ammar et al., 2018).

NLP-PINGAN-TECH team also centered their approach around fine-tuning BERT-based models using larger domain-specific datasets. Their best performing system *SciBERT-SemBERT* is an ensemble of SciBERT, SemBERT (Zhang et al., 2020) based on SciBERT, *Sci-BERT-fake-token* (tokens for position and section details like *[method][sid=xx][ssid=xx]* are added as prefixes to the sentences) and *SciBERT-special-token* (tokens for position and section details like

[method],[sid=1], etc. are added to the SciBERT dictionary to avoid split during tokenization). The other method *SciBer-ACLBERT*, submitted by the NLP-PINGAN-TECH team that achieved a high score, also leverages SciBERT and ACL corpora.

In comparison, the proposed model is trained only on the CL-SciSumm training dataset that consists of 40 manually annotated articles, which were used in the 2018 CL-SciSumm challenge as well, and 1000 document sets that were automatically annotated using neural networks. These 1000 document sets were introduced in 2019 and are of lower quality compared to the manually annotated dataset. Also, we do not use any external corpora to fine-tune our model.

Table 1 shows the performance comparison of our model with the SOTA models. The last 4 rows show the ablation study of our pipeline marked with †. It can be observed that fine-tuning SBERT using our strategy resulted in better scores than *BERT 2018/19 OV + 2018 FT*, *ACL 2018* and *SciBERT 2018*. All three models are based on BERT and SciBERT and trained on much larger datasets. This shows that learning sentence embeddings instead of token embeddings performs better for textual similarity tasks (Reimers and Gurevych, 2019).

Moreover, as evident from the ablation study, individual components of our pipeline also help in increased performance. The most significant improvement, of 30% over fine-tuned SBERT, was achieved by weighted contextual embeddings (*SBERT + WCE*). It can be noted from Table 1 that using just *SBERT + WCE* component of our pipeline outperformed all the SOTA models. This empirically validates that utilizing the unique structural characteristics of the scientific documents can significantly improve the results. Further denoising the weighted contextual embeddings (SBERT + WCE + D) for $m = 3$ improved the performance by around 5%. Moreover, augmenting the contextual embeddings-based similarity scores with WMD achieved a new SOTA by advancing the results of *SBERT + WCE + D* by over 9%.

We also performed experiments to check how the performance of the model varies with train and test sets' size. The proposed method showed improvement when we used 1000 document sets that were automatically annotated using neural networks along with the 40 manually annotated documents. We obtained 0.17790 F1 (0.1869 Recall and 0.1697 Precision) when we trained with just the manually annotated dataset that contained only 40 documents. We also experimented with the 1000 noisy training samples by randomly splitting them into the train (80%) and test (20%) sets and obtained 0.2779 F1 (0.4836 Recall and 0.195 Precision).

C Implementation details

We preprocess the sentences by lowercasing the words, removing the stopwords, removing the special characters and errors due to OCR using NLTK library and regex functions. For SBERT we use the implementation provided by Reimers and Gurevych (2019). We train the model for 3 epcohs with a batch size of 16 on Nvidia GeForce GTX 1080 Ti GPU. The total training time is around 7 minutes. We use AdamW optimizer with a learning rate of $2e^{-05}$ and weight decay of = 0.01. For WMD we use Gensim's implementation (Kusner et al., 2015; Pele and Werman, 2008, 2009). The code is available at `https://github.com/AmitPandey-Research/CitRet_Public.git`.

A Weak Supervision Approach for Predicting Difficulty of Technical Interview Questions

Arpita Kundu, Subhasish Ghosh, Pratik Saini,
Tapas Nayak and **Indrajit Bhattacharya**
TCS Research, India
{arpita.kundu1, g.subhasish, pratik.saini, nayak.tapas, b.indrajit}@tcs.com

Abstract

Predicting difficulty of questions is crucial for technical interviews. However, such questions are long-form and more open-ended than factoid and multiple choice questions explored so far for question difficulty prediction. Existing models also require large volumes of candidate response data for training. We study weak-supervision and use unsupervised algorithms for both question generation and difficulty prediction. We create a dataset of interview questions with difficulty scores for Deep Learning and use it to evaluate SOTA models for question difficulty prediction trained using weak supervision. Our analysis brings out the task's difficulty as well as the promise of weak supervision for it.

1 Introduction

For effective technical interviewing, it is important to know the question difficulty — the probability of a student from a cohort, e.g. senior undergraduate CS students, correctly answering the question. We address the problem of predicting the difficulty of interview questions for candidate cohorts.

Predicting difficulty from the question statement, answer choices and related documents has been studied for multiple choice or factoid questions for reading comprehension and exams (Wang et al., 2014; Huang et al., 2017; Pado´, 2017; Qiu et al., 2019; Benedetto et al., 2020; Yaneva et al., 2020; Benedetto et al., 2021; Cheng et al., 2021; Byrd and Srivastava, 2022). All publicly available datasets (Benedetto et al., 2021; Cheng et al., 2021; Yaneva et al., 2020; Qiu et al., 2019) also contain multiple choice or factoid questions. The nature of technical assessment questions in interviews is different. These look to assess knowledge and understanding rather than memorization of facts and are more open-ended. Answers are long-form, typically spanning 2-5 sentences.

Existing approaches, particularly recent deep models (Xue et al., 2020; Qiu et al., 2019; Benedetto et al., 2021), require large volumes of candidate response data to train the models. This is a challenge when creating a question bank for a new domain or a subject, since field tests need to be performed with real students. In contrast, we explore training question difficulty prediction models using weak supervision based on subject textbooks and Bloom's Taxonomy. This removes dependence on candidate responses and answer assessment.

We explore various strategies of creating weakly-supervised training data. Weak supervision has been explored extensively for many NLP tasks (Lison et al., 2020; Ratner et al., 2020; Ren et al., 2020; Awasthi et al., 2020). For question difficulty, the training data requires not just difficulty scores but interview questions as well. We explore pre-trained large language models (GPT3) and template-based algorithms for generating training questions. We then assign difficulty to these questions using an unsupervised algorithm that uses subject textbooks and Bloom's Taxonomy (Bloom, 1956; Anderson and Krathwohl, 2014). While Bloom's Taxonomy has been used extensively in computer educational testing (Masapanta-Carrión and Ángel Velázquez-Iturbide, 2018; Duran et al., 2018) and for *analysis* of difficulty for short answer questions (Pado´, 2017), but not in predictive models.

For evaluation, we create a dataset of interview questions with difficulty scores from an authoritative textbook on Deep Learning. We use this to evaluate the performance of state-of-the-art QDE models (Benedetto et al., 2020, 2021) when trained using weak-supervision. Our analysis highlights both the challenges of the task as well as the promise of weak-supervision for it.

Our contributions in this paper are as follows. (a) We motivate and introduce the task of difficulty prediction for technical interview questions and curate a dataset for this task. (b) We explore various forms of weak-supervision for this task and analyze the performance of state-of-the-art mod-

Proceedings of the 29th International Conference on Computational Linguistics, pages 4537–4543
October 12–17, 2022.

els. (c) We propose an unsupervised algorithm for question difficulty prediction based on text-book structure and Bloom's Taxonomy. Aside from use in weak supervision, we show that this performs competitively on its own.

2 Dataset

We created a dataset for evaluating interview question difficulty prediction. We made this dataset publicly available[1]. We focus on Deep Learning and use the book "Deep Learning" by Courville et. al. available freely online. First, annotators familiar with technical interviewing and Deep Learning generate interview questions from different chapters of this book. A chapter was given to 2 annotators who reached agreement over validity of generated questions for use in interviews.

Next, we needed to annotate these questions for difficulty on a scale of 1-10. We define higher (lower) difficulty as indicating lower (higher) probability of getting the correct answer from a candidate who has studied this book, and does not have any other exposure to this subject. Attempts to directly annotate difficulty of individual questions led to very low inter-annotator agreement. Instead, we annotated *relative difficulty* for a pair of questions with 3 possible labels: (a) Q1 MORE DIFFICULT, (b) Q2 MORE DIFFICULT and (c) EQUALLY DIFFICULT/EASY. We introduced a *difficulty explanation* label for individual questions in a pair. Possible values were (i) lot of pre-req, (ii) little pre-req, (iii) lot of mathematics, (iv) little mathematics, (v) well-highlighted answer, (vi) hard-to-find answer, (vii) about fundamental concept(s), (viii) about niche concept(s), and (ix) other. Annotators were advised to decide the pair-wise label considering the explanations for the two questions.

The final dataset has 150 unique questions from 16/20 chapters of the book. The questions are well distributed over cognitive tasks (Sec.3.2) and templates (Sec.3.1). 360 question pairs were selected for annotation after running our unsupervised difficulty prediction algorithm (Sec.3.2) to ensure non-triviality of the pair-wise decision. There were 30 unique annotators and each pair was annotated by 5 annotators. After the first round, inter-annotator agreement was 0.23 Fleiss Kappa (fair), and 60/360 questions had a tie. These were broken by 2 additional annotators. The final distribution over labels is 100 Q1, 130 Q2 and 130 EQUAL.

[1]Dataset and Customized definition of BT are publicly available at https://github.com/kunduarpita2012/Technical_question_difficulty_prediction.git

3 Weak Supervision

In this section, we address weak supervision (WS) approaches for question difficulty prediction. WS has been extensively explored for various NLP tasks. One specific challenge is that the training dataset needs not only difficulty scores for questions, but also the questions. Generation of questions is also expertise intensive and gold-standard questions are small in volume. Therefore, WS needs to generate both questions and difficulty scores.

3.1 Question Generation

To generate questions, we explore two different unsupervised approaches: (a) a pre-trained LLM (GPT3), and (b) a template-based algorithm.

GPT3 Questions: Recently, GPT3 (Brown et al., 2020) has been used for weak supervision for many NLP tasks, including question generation from context and answers (Wang et al., 2021). We use prompting with the GPT3 Interview Question preset to generate interview questions from book contexts. In the GPT3 prompt, we provide a context (part of a section) from the book, followed by a new line and an instruction — "Generate a list of questions from the above passage". This was arrived at via experimentation. This process generates diverse questions, but questions are sometimes imprecise in different ways, such as the context not containing the answer, and incompleteness.

Template Questions: To generate more precise questions of types commonly seen in interviews, we use template-based question generation (Puzikov and Gurevych, 2018; Fabbri et al., 2020; Yu and Jiang, 2021). We use the following templates: WHAT IS X?, DEFINE X., EXPLAIN X., WHAT ARE BENEFITS/ADVANTAGES/DISADVANTAGES OF X?, COMPARE X AND Y. For each template, we use precise regular expressions with dictionaries to check its applicability for a sentence. We use a concept dictionary constructed using the book index to detect occurrences of X in sentences.

3.2 Unsupervised Q. Difficulty Prediction

We now describe our unsupervised algorithm for assigning difficulty $d(q)$ to a question q. It as-

signs *context difficulty* $d^c(q)$ considering the specific part of the book from which the question is generated. It also assigns *(cognitive) task difficulty* $d^t(q)$ involved in answering the question considering Bloom's Taxonomy. The overall difficulty of the question is obtained by combining the two: $d(q) = wd^c(q) + (1-w)d^t(q)$.

Context Difficulty: Intuitively, questions from later parts of the book, and similarly later parts of a chapter / section / subsection, are likely to have more dependencies on earlier parts, and are therefore more difficult. We use the chapter no. n^0, section no. n^1 and subsection no. n^2 of a context c to assign a context difficulty score: $d(c) = \sum_{l=0}^{2} w(l)d(n^l; l)$. $w(l)$ is the weight of level l, and we use weights 1, 0.1 and 0.01 for chapters, sections and subsections respectively. The intuition behind the level weights is that two questions generated from two different chapters which are farther apart, are likely to have a greater gap between their difficulty scores than two questions generated from two different sections within a chapter. This intuition similarly extends to subsections within sections. $d(n^l; l)$ is the difficulty associated with level number n^l for level l. So that numbers closer to the end have higher difficulty, we define $d(n^l; l) = n^l / n^l_{max}$, where n^l_{max} is the maximum n^l for a level l.

Task Difficulty: Bloom's Taxonomy (BT) is a well-known resource for determining complexity of educational and assessment tasks. The *Cognitive Process* dimension of BT has levels of cognitive ability, namely REMEMBER, UNDERSTAND, (e.g., explain, classify), APPLY, ANALYZE, EVALUATE and CREATE, and has action verb dictionaries for each level. We first manually assign difficulty scores $d(l)$ to BT levels l. Then the task difficulty $d^t(q)$ of a question q is scored as $d^t(q) = \sum_l \text{sim}(q, l)d(l)$.

To customize BT for interviews, we enrich the taxonomy levels. To each level, we add a list of WH words, and a list of question templates (Sec.3.2). We made this resource public as well[1].

For $\text{sim}(q, l)$, we embed the question q and the BT level l appropriately and compute their cosine similarity. We perform POS tagging and dependency parsing on the question using Spacy. For *verb similarity*, we embed question verbs and level verbs using word2vec (Mikolov et al., 2013) and take the max pair-wise similarity. For *template similarity*, we templatize the question by masking verbs and objects, embed the question template and level templates using pre-trained sBert (Reimers and Gurevych, 2019), and take the max similarity. For *wh similarity*, we check for existence of the question wh word in the level. These three are weighted equally to get $\text{sim}(q, l)$.

Weakly Supervised Training: While there are various weak-supervision frameworks for NLP tasks (Lison et al., 2020; Ratner et al., 2020; Ren et al., 2020; Awasthi et al., 2020), we explore a simple mechanism where we fine-tune a deep model over a weakly labeled training set generated using an unsupervised model. Recent papers have shown that deep models trained using such weak supervision are able to outperform the unsupervised algorithm on the test set (Dehghani et al., 2017; Yu et al., 2021). We plan to explore more sophisticated weak supervision frameworks in future work.

4 Experiments and Analysis

In this section, we report our experiments on the interview question difficulty dataset. We test the usefulness of the following aspects for weak-supervision (WS): (a) difficulty scores predicted by our unsupervised algorithm, (b) algorithm-generated questions, and (c) questions from a related subject.

Model	Q.Subj.	Micro F1
R2DE	DL	0.50
TrQDE[-]	DL	0.51
TrQDE[DL]	DL	**0.54**
TrQDE[DL]	DL+ML	0.53
TrQDE[DL+ML]	DL+ML	0.525
UQDP	-	0.51

Table 1: Comparison of WS types. Q.Subj. indicates subject of questions in training data: deep learning (DL), machine learning (ML). Test questions are on DL. TrQDE[X] indicates TrQDE with MLM fine-tuned on book for subject X. UQDP is unsupervised algorithm for difficulty prediction. Micro-avg F1 is the maximum over threshold θ

Ftr	All	C	T	Tt	Tv	Tw
M. F1	**0.51**	0.50	0.43	0.43	0.37	0.37

Table 2: Ablation for unsupervised difficulty prediction algo. UQDP on test data. C: w/ only context difficulty, T: w/ only all aspects of task difficulty. Tt, Tv, Tw indicate template, verb, wh similarity for task difficulty.

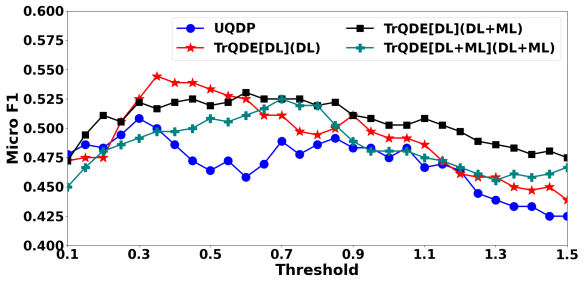

Figure 1: Performance vs threshold θ for best models

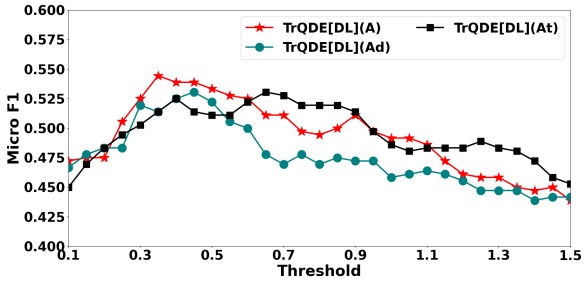

Figure 2: Performance vs threshold θ for question generation algorithms for TrQDE[DL]

All experiments were run on an A100 20GB server. We used Adam with batch-size 16, learning rate 1E-5, dropout rate 0.5 and 50 epochs.

We evaluated two state-of-the-art models for difficulty prediction for factoid / MCQ questions. R2DE (Benedetto et al., 2020) regresses on questions to predict difficulty. We consider the *ques_only* version, since we do not use answers. It uses tf-idf representation of the questions. We report performance for linear regression as the regression model, which was the best. TrQDE (Benedetto et al., 2021) uses transformers to represent the question, with a final regression layer. It fine-tunes the transformer MLM layer using the question corpus, and then further fine-tunes it for the regression task. We report performance for DistilBERT, which worked better than BERT. Here too, we used the *q only* setting. For the unsupervised difficulty prediction algorithm UQDP, we use $w = 0.8$ for combining context and task difficulty.

The primary WS training data covered all 20 chapters of the Deep Learning book, and had 2536 questions (GPT3:1647, Template:889). The secondary WS training data covered 20 chapters from "Pattern Recognition and Machine Learning" (Bishop), also available online. This had 2218 questions (GPT3:1268, Template: 950).

Since the test set has relative difficulty labels, EQUAL is predicted when the difference between a model's predicted difficulty scores for the two questions in a pair is less than or equal to a threshold θ.

The main results are shown in Tab.1. First, WS using UQDP generated difficulty scores for algorithm generated questions improves performance beyond that achieved by using UQDP alone for MLM-fine-tuned versions of TrQDE. This shows the usefulness of both aspects (a) and (b). However, R2DE and the TrQDE with just regression-layer fine-tuning cannot beat UQDP. Next, we analyze aspect (c). Note that UQDP scores difficulty of the DL questions in the training data using the DL book and those of the ML questions using the ML book. Still, including ML questions to train the regression layer does not help, even after including the ML book to fine-tune the MLM layer. The most likely explanation is that the test questions and difficulties are from DL. Including ML questions changes the train distribution, even though the subjects are quite related. Fine-tuning the MLM-layer fits the altered training distribution more closely, leading to poorer results in test.

In Fig.1, we show how micro F1 varies across threshold θ for the 3 best models. This reveals a more nuanced picture. While peak performance of DL-only training is higher, including ML questions in training results has more stable gains across θ values. However, including the ML book for MLM fine-tuning results in worse performance than both.

We investigate aspect (b) further in Fig.2 by plotting performance vs θ when training using different question generation algorithms. We see that performance is the best when using both template and GPT3 generated questions. But, interestingly, templates have better performance individually than GPT3 across θ values. This is very likely because template questions, though smaller in volume and lacking diversity, better mimic the human interview questions seen in test.

Tab.1 showed that UQDP itself has competitive performance on the test set, outperforming R2DE and TrQDE w/o MLM fine-tuning. We investigate aspect (a) further in Tab.2 by performing ablation over different UQDP features. We see that context similarity makes the most significant contribution but adding task similarity improves performance slightly. The contributions of verb and wh similarity are limited compared to template similarity.

5 Error analysis

We now report results of error analysis for our best performing model TrQDE[DL](DL).

The question pairs in the data belong to 3 groups. (A) Same-task-different-context: the questions belong to the same BT level, but are from different book contexts and are about unrelated concepts. (B) Same-context-different task: the questions are from the same context or about related concepts, but belong to different BT levels. (C) Different-context-different-task: the questions are about unrelated concepts / different context and belong to different BT levels. In our labeled data, 36%, 18% and 46% are from groups A, B and C respectively. The third group is the most challenging for relative difficulty labeling. This is so even for human annotators. The Fleiss Kappa scores for inter-annotator agreement are 0.23, 0.27 and 0.21 respectively. Note that while the tasks (question templates) are labeled by annotators when annotating question, while concepts of a question are obtained by eliminating stopwords, wh-words and prepositions using NLTK libraries.

We analyze difficulty prediction errors for each group separately. The errors are of two types. OL-1 (Ordinal Loss 1) errors occur when the predicted and true relative difficulty differ by 1, i.e. the predicted (or true) label is EQUALLY DIFFICULT/EASY and the true (or predicted) label is Q1 MORE DIFFICULT or Q2 MORE DIFFICULT. OL-2 (Ordinal Loss 2) errors occur when the predicted and true labels are the two extremes, i.e. the predicted (or true) label is Q1 MORE DIFFICULT and the true (or predicted) label is Q2 MORE DIFFICULT.

Overall, $\sim$ 32.5% of the predictions of TrQDE[DL](DL) correspond to OL-1 errors and $\sim$ 13% to OL-2. Fig.3 shows a group-wise drill-down. First, we observe that the total error is highest for group C, as expected, as is OL-2 error, demonstrating that it is the hardest group. Between groups A and B, total error is slightly higher for B, indicating that predicting task-difficulty is a bigger challenge than context difficulty.

Deeper analysis provided further insights into the prediction errors of TrQDE. One of these stems from an underlying assumption for UQDE that context difficulty is higher for later parts of a book. However, concepts introduced earlier are often revisited in later chapters in the context of related concepts. UQDE assigns context difficulty incorrectly in such cases and corrupts training data for TrQDE.

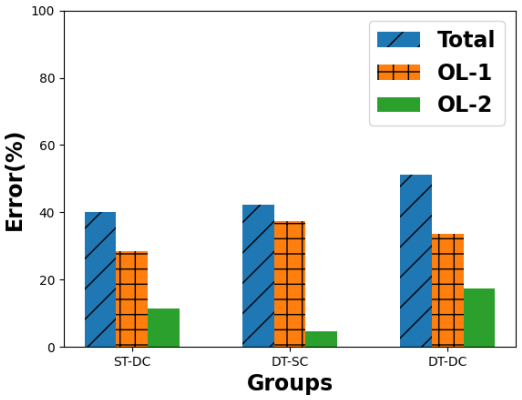

Figure 3: Prediction error of TrQDE[DL](DL) for different groups of question pairs where ST-DC, DT-SC and DT-DC represent Same-task-different-context, Same-context-different task and Different-context- different-task groups respectively.

For example, graphical models and their types are first introduced when Structured Probability Models are introduced in 3.14 under Probability and Information Theory, and discussed again when discussing Graphs for Model Structure in 16.2 Structured Probabilistic Models for Deep Learning. As a result, the question 'What are the different categories of graphical models?' from 16.2 incorrectly gets assigned a higher difficulty level. Other than this, we observe that while TrQDE learns from UQDE-assigned levels in general, sometimes it makes an incorrect prediction for test question pairs where UQDE makes the correct prediction.

6 Conclusions

In summary, we have motivated the task of difficulty prediction for technical interview questions and curated a dataset for evaluation. We have shown that weak-supervision using algorithm-generated questions and an unsupervised difficulty scoring algorithm is a promising direction for fine-tuning related state-of-the-art models for this task. The simple unsupervised algorithm itself shows competitive performance and hints at aspects that new models for this challenging problem will need to consider.

References

Lorin Anderson and David Krathwohl. 2014. *A taxonomy for learning, teaching and assessing: A revision of Bloom's*. Addison Wesley Longman.

Abhijeet Awasthi, Sabyasachi Ghosh, Rasna Goyal, and

Sunita Sarawagi. 2020. Learning from rules generalizing labeled exemplars. In *In International Conference on Learning Representations (ICLR)*.

Luca Benedetto, Andrea Cappelli, Roberto Turrin, and Paolo Cremonesi. 2020. R2de: a nlp approach to estimating irt parameters of newly generated questions. In *Proceedings of the Tenth International Conference on Learning Analytics and Knowledge (LAKE)*.

Luca Benedetto, Paolo Cremonesi, Giovanni Aradelli, Andrea Cappelli, Andrea Giussani, and Roberto Turrin. 2021. On the application of transformers for estimating the difficulty of multiple-choice questions from text. In *Proceedings of the 16th Workshop on Innovative Use of NLP for Building Educational Applications (ACL)*.

Benjamin Bloom. 1956. *The taxonomy of educational objectives, the classification of educational goals, volume Handbook I: Cognitive Domain*. David McKay.

Tom B. Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, T. J. Henighan, Rewon Child, Aditya Ramesh, Daniel M. Ziegler, Jeff Wu, Clemens Winter, Christopher Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. 2020. Language models are few-shot learners. In *Proceedings of 34th Conference on Neural Information Processing Systems (NeurIPS)*.

Matthew Byrd and Shashank Srivastava. 2022. Predicting difficulty and discrimination of natural language questions. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (ACL)*.

Yi Cheng, Siyao Li, Bang Liu, Ruihui Zhao, Sujian Li, Chenghua Lin, and Yefeng Zheng. 2021. Guiding the growth: Difficulty-controllable question generation through step-by-step rewriting. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (AACL-IJCNLP)*.

Mostafa Dehghani, Hamed Zamani, Aliaksei Severyn, Jaap Kamps, and W. Bruce Croft. 2017. Neural ranking models with weak supervision. In *Proceedings of the SIGIR*.

Rodrigo Duran, Juha Sorva, and Sofia Leite. 2018. Towards an analysis of program complexity from a cognitive perspective. In *Proceedings of the ACM Conference on International Computing Education Research (ICER)*.

Alexander Fabbri, Patrick Ng, Zhiguo Wang, Ramesh Nallapati, and Bing Xiang. 2020. Template-based question generation from retrieved sentences for improved unsupervised question answering. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics (ACL)*.

Zhenya Huang, Qi Liu, Enhong Chen, Hongke Zhao, Mingyong Gao, Si Wei, Yu Su, and Guoping Hu. 2017. Question difficulty prediction for reading problems in standard tests. In *Proceedings of the Thirty-First AAAI Conference on Artificial Intelligence (AAAI)*.

Pierre Lison, Jeremy Barnes, Aliaksandr Hubin, Aliaksandr Hubin Pierre Lison, Jeremy Barnes, and Samia Touileb. 2020. Named entity recognition without labelled data: A weak supervision approach. In *In Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics (ACL)*.

Susana Masapanta-Carrión and J. Ángel Velázquez-Iturbide. 2018. A systematic review of the use of bloom's taxonomy in computer science education. In *Proceedings of the 49th ACM Technical Symposium on Computer Science Education (SIGCSE)*.

Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013. Distributed representations of words and phrases and their compositionality. In *Proceedings of 27th Conference on Neural Information Processing Systems (NeurIPS)*.

Ulrike Pado´. 2017. Question difficulty – how to estimate without norming, how to use for automated grading. In *Proceedings of the 12th Workshop on Innovative Use of NLP for Building Educational Applications (ACL)*.

Yevgeniy Puzikov and Iryna Gurevych. 2018. E2E NLG challenge: Neural models vs. templates. In *Proceedings of the 11th International Conference on Natural Language Generation (ICNLG)*.

Zhaopeng Qiu, Xian Wu, and Wei Fan. 2019. Question difficulty prediction for multiple choice problems in medical exams. In *Proceedings of Conference on Information and Knowledge Management (CIKM)*.

Alexander Ratner, Stephen H. Bach, Henry R. Ehrenberg, Jason A. Fries, Sen Wu, and Christopher Ré. 2020. Snorkel: rapid training data creation with weak supervision. *Proceedings of the 44th International Conference on Very Large Data Bases (VLDB) Endowment*.

Nils Reimers and Iryna Gurevych. 2019. Sentence-bert: Sentence embeddings using siamese bert-networks. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing (EMNLP)*.

Wendi Ren, Yinghao Li, Hanting Su, David Kartchner, Cassie Mitchell, and Chao Zhang. 2020. Denoising multi-source weak supervision for neural text classification. In *In Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics (ACL)*.

Quan Wang, Jing Liu, Bin Wang, and Li Guo. 2014. A regularized competition model for question difficulty estimation in community question answering services. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP)*.

Shuohang Wang, Yang Liu, Yichong Xu, Chenguang Zhu, and Michael Zeng. 2021. Want to reduce labeling cost? GPT-3 can help. In *Findings of the Association for Computational Linguistics: EMNLP*.

Kang Xue, Victoria Yaneva, Christopher Runyon, and Peter Baldwin. 2020. Predicting the difficulty and response time of multiple choice questions using transfer learning. In *Proceedings of the Fifteenth Workshop on Innovative Use of NLP for Building Educational Applications at Annual Meeting of the Association for Computational Linguistics (ACL)*.

Victoria Yaneva, Le An Ha, Peter Baldwin, and Janet Mee. 2020. Predicting item survival for multiple choice questions in a high-stakes medical exam. In *Proceedings of the 12th Conference on Language Resources and Evaluation (LREC)*.

Xiaojing Yu and Anxiao Jiang. 2021. Expanding, retrieving and infilling: Diversifying cross-domain question generation with flexible templates. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics (EACL)*.

Yue Yu, Simiao Zuo, Haoming Jiang, Wendi Ren, Tuo Zhao, and Chao Zhang. 2021. Fine-tuning pretrained language model with weak supervision: A contrastive-regularized self-training approach. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL-HLT)*.

Reinforcement Learning with Large Action Spaces for Neural Machine Translation

Asaf Yehudai, Leshem Choshen, Lior Fox, Omri Abend
Department of Computer Science
The Hebrew University of Jerusalem
{first.last}@mail.huji.ac.il

Abstract

Applying Reinforcement learning (RL) following maximum likelihood estimation (MLE) pre-training is a versatile method for enhancing neural machine translation (NMT) performance. However, recent work has argued that the gains produced by RL for NMT are mostly due to promoting tokens that have already received a fairly high probability in pre-training. We hypothesize that the large action space is a main obstacle to RL's effectiveness in MT, and conduct two sets of experiments that lend support to our hypothesis. First, we find that reducing the size of the vocabulary improves RL's effectiveness. Second, we find that effectively reducing the dimension of the action space without changing the vocabulary also yields notable improvement as evaluated by BLEU, semantic similarity, and human evaluation. Indeed, by initializing the network's final fully connected layer (that maps the network's internal dimension to the vocabulary dimension), with a layer that generalizes over similar actions, we obtain a substantial improvement in RL performance: 1.5 BLEU points on average.[1]

1 Introduction

The standard training method for sequence-to-sequence tasks, specifically for NMT is to maximize the likelihood of a token in the target sentence, given a gold standard prefix (henceforth, maximum likelihood estimation or MLE). However, despite the strong performance displayed by MLE-trained models, this token-level objective function is limited in its ability to penalize sequence-level errors and is at odds with the sequence-level evaluation metrics it aims to improve. One appealing method for addressing this gap is applying policy gradient methods that allow incorporating non-differentiable reward functions, such as the ones often used for MT evaluation (Shen et al., 2016, see §2). For brevity, we will refer to these methods simply as RL.

The RL training procedure consists of several steps: (1) generating a translation with the pre-trained MLE model, (2) computing some sequence-level reward function, usually, one that assesses the similarity of the generated translation and a reference, and (3) updating the model so that its future outputs receive higher rewards. The method's flexibility, as well as its ability to address the exposure bias (Ranzato et al., 2016; Wang and Sennrich, 2020), makes RL an appealing avenue for improving NMT performance. However, a recent study (C19; Choshen et al., 2019) suggests that current RL practices are likely to improve the prediction of target tokens only where the MLE model has already assigned that token a fairly high probability.

In this work, we observe that one main difference between NMT and other tasks in which RL methods excel is the size of the action space. Typically, the size of the action space in NMT includes all tokens in the vocabulary, usually tens of thousands. By contrast, common RL settings have either small discrete action spaces (e.g., Atari games (Mnih et al., 2013)), or continuous action spaces of low dimension (e.g., MuJoCo (Todorov et al., 2012) and similar control problems). Intuitively, RL takes (samples) actions and assesses their outcome, unlike supervised learning (MLE) which directly receives a score for all actions. Therefore, a large action space will make RL less efficient, as individual actions have to be sampled in order to assess their quality. Accordingly, we experiment with two methods for decreasing the size of the action space and evaluate their impact on RL's effectiveness.

We begin by decreasing the vocabulary size (or equivalently, the number of actions), conducting experiments in low-resource settings on translating four languages into English, using BLEU both as

[1] https://github.com/AsafYehudai/Reinforcement-Learning-with-Large-Action-Spaces-for-Neural-Machine-Translation

Proceedings of the 29th International Conference on Computational Linguistics, pages 4544–4556
October 12–17, 2022.

the reward function and the evaluation metric. Our results show that RL yields a considerably larger performance increase (about 1 BLEU point on average) over MLE training than is achieved by RL with the standard vocabulary size. Moreover, our findings indicate that reducing the size of the vocabulary can improve upon the MLE model even in cases where it was not close to being correct. See §4.

However, in some cases, it may be undesirable or unfeasible to change the vocabulary. We therefore experiment with two methods that effectively, reduce the dimensionality of the action space without changing the vocabulary. We note that generally in NMT architectures, the dimensionality of the decoder's internal layers (henceforth, d) is significantly smaller than the target vocabulary size (henceforth, $|V_T|$), which is the size of the action space. A fully connected layer is generally used to map the internal representation to suitable outputs. We may therefore refer to the rows of the matrix (parameters) of this layer, as *target embeddings*, mapping the network's internal low-dimensional representation back to the vocabulary size, the actions. We use this term to underscore the analogy between the network's first embedding layer, mapping vectors of dimension $|V_T|$ to vectors of dimension d, and target embeddings that work in an inverse fashion. Indeed, it is often the case (e.g., in BERT, Devlin et al., 2019) that the weights of the source and target embeddings are shared during training, emphasizing the relation between the two.

Using this terminology, we show in simulations (§5.1) that when similar actions share target embeddings, RL is more effective. Moreover, when target embeddings are initialized based on high-quality embeddings (BERT's in our case), freezing them during RL yields further improvement still. We obtain similar results when experimenting on NMT. Indeed, using BERT's embeddings for target embeddings improves performance on the four language pairs, and freezing them yields an additional improvement on both MLE and RL as reported by both automatic metrics and human evaluation. Both initialization and freezing are novel in the context of RL training for NMT. Moreover, when using BERT's embeddings, RL's ability to improve performance on target tokens to which the pre-trained MLE model did not assign a high probability, is enhanced (§5.2).

2 Background

2.1 RL in Machine Translation

RL is used in text generation (TG) for its ability to incorporate non-differentiable signals, to tackle the exposure bias, and to introduce sequence-level constraints. The latter two are persistent challenges in the development of TG systems, and have also been addressed by non-RL methods (e.g., Zhang et al., 2019; Ren et al., 2019). In addition, RL is grounded within a broad theoretical and empirical literature, which adds to its appeal.

These properties have led to much interest in RL for TG in general (Shah et al., 2018) and NMT in particular (Wu et al., 2018a). Numerous policy gradient methods are commonly used, notably REINFORCE (Williams, 1992), and Minimum Risk Training (MRT; e.g., Och, 2003; Shen et al., 2016). However, despite increasing interest and strong results, only a handful of works studied the source of observed performance gains by RL in NLP and its training dynamics, and some of these have suggested that RL's gains are partly due to artifacts (Caccia et al., 2018; Choshen et al., 2019).

In a recent paper, C19 showed that existing RL training protocols for MT (REINFORCE and MRT) take a prohibitively long time to converge. Their results suggest that RL practices in MT are likely to improve performance only where the MLE parameters are already close to yielding the correct translation. They further suggest that observed gains may be due to effects unrelated to the training signal, but rather from changes in the shape of the distribution curve. These results may suggest that one of the drawbacks of RL is the uncommonly large action space, which in TG includes all tokens in the vocabulary, typically tens of thousands of actions or more.

To the best of our knowledge, no previous work considered the challenge of large action spaces in TG, and relatively few studies considered it in different contexts. One line of work assumed prior domain knowledge about the problem, and partitioned actions into sub-groups (Sharma et al., 2017), or similar to our approach, embedding actions in a continuous space where some metric over this space allows generalization over similar actions (Dulac-Arnold et al., 2016). More recent work proposed to learn target embeddings when the underlying structure of the action space is apriori unknown using expert demonstrations (Tennenholtz and Mannor, 2019; Chandak et al., 2019).

This paper establishes that the large action spaces are a limiting factor in the application of RL for NMT, and propose methods to tackle this challenge. Our techniques restrict the size of the embedding space, either explicitly or implicitly by using an underlying continuous representation.

2.2 Technical Background and Notation

Notation. We denote the source sentence with $X = (x_1, ..., x_S)$ and the reference sentence with $Y = (y_1, ..., y_T)$. Given X, the network generates a sentence in the target language $Y' = (y'_1, ..., y'_M)$. Target tokens are taken from a vocabulary V_T. During inference, at each step i, the probability of generating a token $y'_i \in V_T$ is conditioned on the sentence and the predicted tokens, i.e., $P_\theta(y'_i|X, y'_{<i})$, where θ is the model parameters. We assume there is exactly one valid target token, the reference token, as in practice, training is done against a single reference (Schulz et al., 2018).

NMT with RL. In RL terminology, one can think of an NMT model as an agent, which interacts with the environment. In this case, the environment state consists of the previous words $y'_{<i}$ and the source sentence X. At each step, the agent selects an action according to its policy, where actions are tokens. The policy is defined by the parameters of the model, i.e., the conditional probability $P_\theta(y'_i|y'_{<i}, X)$. Reward is given only once the agent generates a complete sequence Y'. The standard reward for MT is the sentence level BLEU metric (Papineni et al., 2002), matching the evaluation metric. Our goal is to find the parameters that will maximize the expected reward.

In this work, we use MRT (Och, 2003; Shen et al., 2015), a policy gradient method adapted to MT. The key idea of this method is to optimize at each step a re-normalized risk, defined only over the sampled batch. Concretely, the expected risk is defined as:

$$L_{risk} = \sum_{u \in U(X)} R(Y, u) \frac{P(u|X)^\beta}{\sum_{u' \in U(X)} P(u'|X)^\beta} \quad (1)$$

where u is a candidate hypothesis sentence, $U(x)$ is the sample of k candidate hypotheses, Y is the reference, P is the conditional probability that the model assigns a candidate hypothesis u given source sentence X, β a smoothness parameter and R is BLEU.

3 Methodology

Architecture. We use a similar setup as used by Wieting et al. (2019), adapting their fairSeq-based (Ott et al., 2019) codebase to our purposes.[2] Similar to their Transformer architecture we use gated convolutional encoders and decoders (Gehring et al., 2017). We use 4 layers for the encoder and 3 for the decoder, the size of the hidden state is 768 for all layers, and the filter width of the kernels is 3. Additionally, the dimension of the BPE embeddings is set to 768.

Data Prepossessing. We use BPE (Sennrich et al., 2016) for tokenization. The vocabulary size is set to 40K for the combined source and target vocabulary as done by Wieting et al. (2019). For the small target vocabulary experiments, we change the target vocabulary size to 1K and keep the source vocabulary unchanged.

Objective Functions. Following Edunov et al. (2018), we train models with MLE with label-smoothing (Szegedy et al., 2016; Pereyra et al., 2017) of size 0.1. For RL, we fine-tune the model with a weighted average of the MRT L_{risk} and the token level loss L_{mle}.

Our fine-tuning objective thus becomes:

$$L_{Average} = \alpha \cdot L_{mle} + (1 - \alpha) \cdot L_{risk} \quad (2)$$

We set α to be 0.3 shown to work best by Wu et al. (2018b). We set β to 1. We generate eight hypotheses for each MRT step (k=8) with beam search. We train with smoothed BLEU (Lin and Och, 2004) from the Moses implementation.[3] Moreover, we use this metric to report results and verify they match sacrebleu (Post, 2018).[4]

Optimization. We train the MLE objective over 200 epochs and the combined RL objective over 15. We perform early stopping by selecting the model with the lowest validation loss. We optimize with Nesterov's accelerated gradient method (Sutskever et al., 2013) with a learning rate of 0.25, a momentum of 0.99, and re-normalize gradients to a 0.1 norm (Pascanu et al., 2012).

[2] https://github.com/jwieting/beyond-bleu

[3] https://github.com/jwieting/beyond-bleu/blob/master/multi-bleu.perl

[4] https://github.com/mjpost/sacrebleu

Data. We experiment with four languages: German (De), Czech (Cs), Russian (Ru), and Turkish (Tr), translating each of them to English (En). For training data for cs-en, de-en, and ru-en, we use the WMT News Commentary v13[5] (Bojar et al., 2017). For tr-en training data, we use WMT 2018 parallel data, which consists of the SETIMES2 corpus (Tiedemann, 2012). The validation set is a concatenation of newsdev 2016 and 2017 released for WMT18. Test sets are the official WMT18 test sets. Those experiments focus on a low-resource setting. We choose this setting as RL experiments are computationally demanding and this setting is common in the literature for RL experiments like ours Wieting et al. (2019). (see data statistics in Supp. §A)

4 Reducing the Vocabulary Size

We begin by directly testing our hypothesis that the size of the action space is a cause for the long convergence time of RL for NMT. To do so, we train a model with target-side BPE taken from a much smaller vocabulary than is typically used.

We begin by training two MLE models, one with a large (17K-31K) target vocabulary (LTV) and another with a target vocabulary of size 1K (STV). The source vocabulary remains unchanged. We start with the MLE pretraining and then train each of the two models with RL.

Results (Table 1) show that the RL training with STV achieves about 1 BLEU point more than the RL training with LTV.[6] For a comparison of the models' entropy see Supp. §B. In order to verify that the improvement does not stem from the choice of α mixing RL and MLE (see Eq. 2), we repeat the training for De-En with $\alpha \in \{0, 1\}$, we find that $\alpha = 0.3$ is superior to both. Moreover, RL improves STV more than LTV when training with only the RL objective ($\alpha = 1$). This indicates that RL training contributes to the observed improvement.

We next turn to analyze what tokens are responsible for the observed performance gain. Specifically, we examine whether reducing the vocabulary size resulted in RL being able to promote target tokens that received a low rank by the pretrained MLE model. For each model, for 700K trials, we compute what rank the model assigns to the gold token y_i for a context $y'_{<i}$ and source sentence X. Formally, $\forall r \in |V_T|$, $P^r_{model} = \frac{\#\{gold\ token\ assigns\ to\ the\ r\ rank\}}{\#\{all\ trials\}}$. We then compare the rank distribution of the MLE model to that of the RL model by subtracting those two distributions. In our notation, for each rank r, $\Delta P^r = P^r_{RL} - P^r_{MLE}$. This subtraction represents how RL influences the model's ability to assign the correct token y_i for each rank. The greater the positive effect of RL is, the more probable it is that the probability will be positive for the first rank, and negative for lower ranks (due to the probability shift to first place).

Model	DE-EN	CS-EN	RU-EN	TR-EN
LTV	25.07	15.16	16.67	12.76
LTV+RL	25.67	15.33	16.9	12.98
Diff.	0.6	0.17	0.23	0.22
STV	21.83	13.79	14.63	10.37
STV+RL	23.23	14.62	15.73	11.96
Diff.	1.4	0.83	1.1	1.59

Table 1: BLEU scores for translating four languages to English using MLE pretraining followed by RL, and comparing a model with a large vocabulary (LTV) to a small one (STV). The top (bottom) block presents results for LTV (STV) with and without RL, and the difference between them (Diff.). RL with STV gains more than 1 point more (on average) over the pre-trained MLE model, than RL with LTV.

Figure 1 presents the probability difference per rank for LTV and STV. We can see that for the first rank the probability shift due to RL training with STV is more than twice the shift caused by RL training with LTV. Consequently, the probability shift for the following ranks is usually more negative for small vocabulary settings. The figure indicates that indeed the shift of probability mass to higher positions occurs substantially more when we apply RL using a smaller action space. Moreover, the STV training was able to shift probability mass from lower ranks upwards compared to LTV. An indication for that is that, within the first one hundred ranks, STV reduces the probability of 83 of them, whereas LTV of only 2.

5 Reducing the Effective Dimensionality of the Action Space

Finding that reducing the number of actions improves RL's performance, we propose a method

[5] http://data.statmt.org/wmt18/translation-task

[6] Preliminary experiments showed that altering the random seed changes the BLEU score by ±0.01 points.

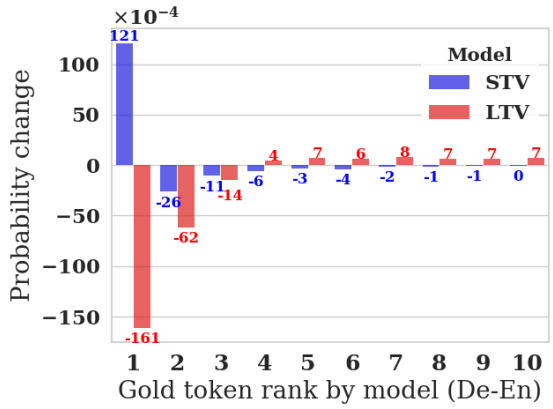

Figure 1: Comparison of probability shift due to RL training of assign y_{best} for ten first words for both LTV and STV. in blue, you can see the results with BPE of size 1,000, STV. in red are the results with BPE of size 30,000, LTV. a clear improvement of assigning y_{best} in first place for STV.

for reducing the effective number of actions, without changing the actual output. The vocabulary size might be static, as in pre-trained models (Devlin et al., 2019), and reducing it might help RL but be sup-optimal for MLE (Gowda and May, 2020), or introduce out-of-domain words (Koehn and Knowles, 2017). We propose to do so by using target embeddings that generalize over tokens that appear in similar contexts. We explore two implementations of this idea, one where we initialize the target embeddings with high-quality embeddings, and another where we freeze the learned target embeddings during RL. We also explore a combination of the two approaches. Freezing the target embeddings (decoder's last layer) can be construed as training the network to output the activations of the penultimate layer, where a fixed function then maps it to the dimension of the vocabulary.

We note that although freezing is a common procedure (Zoph et al., 2016; Thompson et al., 2018; Lee et al., 2019; Coster et al., 2021), as far as we know, it has never been applied in the use of RL for sequence to sequence models.

Denote the function that the network computes with f_θ. f_θ can be written as $h_{\theta_2} \circ g_{\theta_1}$, where $\theta = (\theta_1, \theta_2)$, g maps the input – source sentence X and model translation prefix $y'_{<i}$ – into $\mathbb{R}^d$, and h maps g's output into $\mathbb{R}^{|V_T|}$.

Using this notation, we can formulate the method as loading pre-trained MLE target embeddings to h_{θ_2} or freezing it (or both). As for many encoder-decoder architectures (including the Transformer), it holds that $d \ll |V_T|$, this can be thought of as constraining the agent to select a d-dimensional continuous action, where h_{θ_2} is a known transformation performed by the environment.

The importance of target embedding is that they allow for better generalization over actions. The intuition is as follows. Assume two tokens have the same embedding, and similar semantics, i.e., they are applicable in the same contexts (synonyms). Since they have the same target embeddings, during training the network will perform the same gradient updates when encountering either of them, except for in the last layer (since they are still considered different outputs). If the target embeddings are not frozen, encountering either of them during training will lead to very similar updates (since they have the same target embeddings), but their target embeddings may drift slightly apart, which will cause a subsequent drift in the lower layers. If the target embeddings are frozen, the gradient updates they will yield will remain the same and expedite learning. We hypothesize a similar effect during training, where tokens that have similar (but not identical) embeddings, and a similar (but not identical) distribution would benefit in training from each other. This motivates us to explore a combination of informative initialization and parameter freezing. (see formal proof in Supp. §E).

5.1 Motivating Simulation through Policy Parameterization in Large Action Spaces

In order to examine the intuition outlined above in a controlled setting, we consider a synthetic RL problem in which the action space is superficially enlarged. The task is a (contextual) multi-armed-bandit, with K actions. At each step, an input state is sampled from the environment (the "context"; a random vector sampled from a multivariate Gaussian distribution). A random, fixed, non-linear binary classifier determines whether action #1 or action #2 is rewarding based on the given context (actions 3-K are never rewarding), and the reward for each action is $r + z$ where $r = 1$ for the rewarding action and 0 for all other actions, and $z \sim \mathcal{N}(0, 0.1)$. Crucially, we duplicate each action a times, resulting in a total of $K \times a$ actions at the policy level – whereas for the environment all 'copies' of a given action are equivalent.

The problem structure, including the classifier itself, is unknown to the RL agent, which directly optimizes a policy parameterized as a fully-connected

feed-forward neural network. We control two aspects of the last layer of the policy network, resulting in a total of four variants of agents. First, the last layer can be frozen to its initial value, or learned (by RL). Second, the last layer can be initialized at random, or induce a prior regarding the duplicated actions (such that weight vectors projecting to different copies of a given action are initialized identically). We call the latter the *informative* initialization.

We stress that the informative initialization carries no information about the underlying reward structure of the problem (i.e., the classifier, and the identity of the rewarding actions), but only as to which actions are duplicated. Nevertheless, as shown in Figure 2, a prior regarding the structure of the action space is helpful on its own, leading to faster learning (compare *Informative* to *Full net*).

Results fit the intuition presented. With informative last layer initialization, learning in previous layers generalizes over the duplicated actions and boosts early stages of learning, leading to faster convergence. We note that in this setting faster learning is not only the result of learning fewer parameters. Notably, freezing the last layer with random initialization, prohibits the network from learning the task. This is due to the regime of a very large action space (output layer; width 4000) compared to the dimensionality of the hidden representations (width 300). Freezing an informative initialization, on the other hand, sets the network in a rather different regime, in which the effective size of the output layer is (much) smaller than the hidden representation (i.e #'real' actions; 10). In this regime, the network is generally expressive enough so that it can quickly learn the task even with a fixed, random readout layer (Hoffer et al., 2018).

To conclude, this example provides evidence that initializing and possibly freezing the last layer in the policy network in a way that respects the structure of the action space is helpful for learning in vast action spaces, as it supports generalization over similar or related actions. Importantly, this helps even when the (frozen) initialization does not contain task-specific information. In a more realistic scenario, actions are not simply a complete duplicate of each other, but rather are organized in some complex structure. Informative initialization, then, accounts not for duplicating weights, but for initializing them in such a way that a-priori reflects, or is congruent, with this structure. This motivates our approach – in the realistic, complicated task of MT – to freeze a learned output layer for the policy network, from a model whose embeddings have been shown to be effective across a wide range of tasks (in our case, BERT).

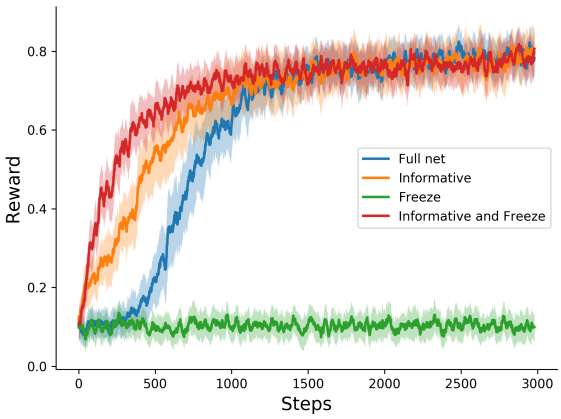

Figure 2: Simulating learning in large action spaces. Figures show a moving average over 20 steps of the underlying binary reward. Solid curves denote mean, shaded area denote ± 0.5 s.d. ($N = 50$ trials per agent, $K = 10, a = 400$, network architecture: 10-300-300-4000). Informative initialization is effective on its own, and more so when freezing is applied.

5.2 NMT Experiments

The motivating analysis and simulations indicate that it is desirable to use target embeddings that assign similar values to similar actions. Doing so can be viewed as an effective reduction in the dimensionality of the action space. We turn to experiment with this approach on NMT. We explore two approaches: (1) freezing h during RL; (2) informatively initializing h, as well as their combination. Our main results are presented in Table 2.

As a baseline, we experiment with freezing uninformative target embeddings: target embeddings are randomly initialized and frozen during both MLE and RL. Unsurprisingly, doing so does not help training, and in fact, greatly degrades it (in about 2 BLEU points in En-De).

Next, we examine whether the target embeddings of the MLE pretraining are informative enough, namely whether freezing them during RL leads to improved effectiveness. Results show a slight improvement in BLEU when doing so, which is encouraging given that the frozen embeddings' weights consist of more than half of the network's trainable parameters. Indeed, freezing the embed-

ding layers has a dramatic impact on the volume of trainable parameters, decreasing their size by more than 60%. In Supp. §D we present the number of trainable parameters in each setting.

We therefore hypothesize that, as in the simulations (§5.1), the quality of the frozen embedding space is critical for the success of this approach. As using frozen MLE embeddings improves performance, but only somewhat, we further consider target embeddings that were trained on much larger datasets, specifically BERT's embedding layer.[7]

For this set of experiments, we adjust the target vocabulary to be BERT's vocabulary of size $|V_T| = 30526$. We train RL models with and without freezing the embedding layers and with and without loading BERT embedding. We report results of MLE training with BERT's embedding when the embedding is kept frozen as it reaches superior results (see Supp. §C).

The results (Table 2) directly parallels our findings in the simulations: Initializing from BERT ($+BERT + RL$) improves performance across all language pairs, and freezing ($+RL + FREEZE$) yields an additional improvement in most settings, albeit a more modest one. Combining both methods provides additional improvement. Indicating that FREEZE and BERT are helpful both independently and in conjunction. In total, our model ($+BERT+RL+FREEZE$) achieves 1.5 BLEU points improvements over regular RL. We also report semantic similarity scores in Supp. §F.

Notably, our method surpasses the LTV+RL results (Table 1) across all languages except German, overall about 1.5 BLEU points more on average. We hypothesize that the reason for this degradation is the lower results of the MLE with BERT's vocabulary compared to the joint BPE vocabulary which is known to be superior to BPE on each language individually, especially when the source and target languages are close (Sennrich et al., 2016). These considerations are peripheral to our discussion, which specifically targets the effectiveness of the RL approach.

Finally, initializing from BERT increases RL's ability to promote tokens that were not ranked high according to the MLE model (Fig. 3).

6 Human Evaluation

We perform human evaluation, comparing the baseline RL with our proposed model. We selected 100 translations from the respective test sets of each language. The annotation was performed by two professional annotators (contractors of the project), who work in the field of translation. Both are native English speakers. The annotators assigned a score from 0 to 100, judging how well the translation conveyed the information contained in the reference (see annotation guidelines in Supp. §G). From Fig. 4, we see that our proposed model scores the highest across all language pairs. To test statistical significance, we use the Wilcoxon rank sum test

[7] HuggingFace implementation

MODEL	De-En	Cs-En	Ru-En	Rr-En
MLE	22.38	15.81	17.31	12.60
+RL	23.19	15.81	17.31	12.66
+RL+FREEZE	23.14	16.04	17.78	13.18
+BERT	23.46	16.59	18.14	14.15
+BERT+RL	24.44	17.04	**18.68**	14.37
+BERT+RL+FREEZE	**24.71**	**17.37**	18.30	**14.55**

Table 2: BLEU scores on translating four languages to English. The upper block shows the baseline scores of training only with MLE, and with MLE followed by RL. RL presents modest improvement (if any) over only using MLE. +RL+FREEZE shows some improvement due to freezing the target embeddings. The lower three rows show results when using BERT's target embeddings (informative initialization). Additional benefit is seen from freezing (+RL+BERT+FREEZE).

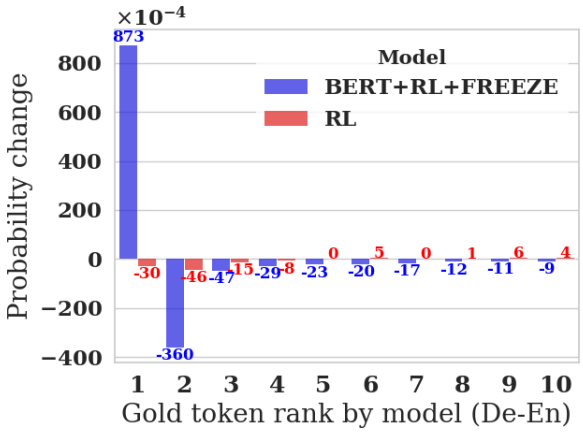

Figure 3: Comparison of the change in the rank distribution of the target token following RL in two settings, one where RL training with frozen BERT embeddings is used (blue) and the second when we used basic RL training (red). The gain in probability in the first rank indicates that the model is more probable to be correct (which is reflected in its superior performance over the pre-trained MLE model). The negative values in the following places demonstrate how RL with frozen high-quality target embeddings can improve not only when the MLE model is initially close to being correct.

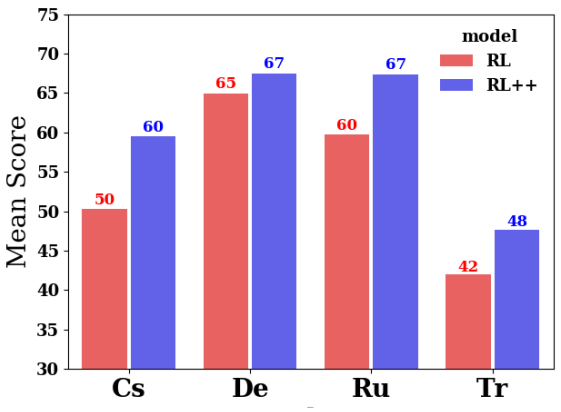

Figure 4: Average human ratings on 200 sentences from the test set for each of the respective languages. RL is the baseline RL model and $RL++$ is our model ($+BERT + RL + FREEZE$). The performance of our model is consistently better than the baseline.

to standardize score distributions fit for our setting (Graham et al., 2015). Comparing the two models' distributions, we got a p-value of $8.5e^{-5}$ indicating the improvement is significant. We emphasize that our main goal is showing that our method can improve the optimized metric (e.g., BLEU), and hence the improvement over the semantic similarity score and the human evaluation is an additional indication of our method's robustness.

7 Comparing Target Embedding Spaces

The previous section discussed how BERT's target embeddings improve RL performance, compared to target embeddings learned by MLE. We now turn to directly analyze the generalization ability of the two embeddings. We do so by comparing the embeddings of semantically related words.

We use WordNet (Miller, 1998) and spaCy[8] to compile three lists of word pairs: inflections (e.g., 'documentaries' / 'documentary', 'boxes' / 'box', 'stemming' / 'stem'), synonyms (e.g., 'luckily' / 'fortunately', 'amazement' / 'astonishment', 'purposely' / 'intentionally'), and random pairs, and compare the embeddings assigned to these pairs using BERT and MLE embeddings. Figure 5 presents the distributions of the cosine similarity of the pairs in the three lists for both embedding spaces. Results show that MLE embeddings for the different lists have almost identical distributions, demonstrating the limited informativeness of these target embeddings. In contrast, BERT embeddings only display a small overlap between the similarity distributions of inflections and random pairs. However, synonyms' distribution remains quite similar to that of random pairs. In conclusion, BERT embeddings better discern semantics overall compared to MLE embeddings, which may partly account for their superior performance. Results also indicate BERT's embeddings could be further improved.

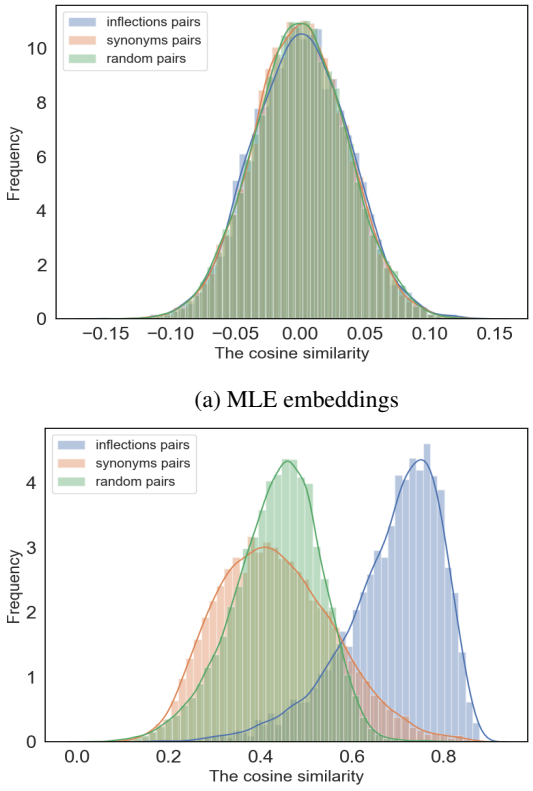

(a) MLE embeddings

(b) BERT embeddings

Figure 5: Comparison of the distribution of the cosine similarity between word pairs from three groups: random word pairs (in green), synonym pairs that do not share a stem (in orange), and pairs of synonyms that share a stem (in blue). The top figure refers to the target embeddings learned by MLE, and the bottom one to BERT embeddings. The ability of the embeddings to distinguish between these three groups is informative of their ability to map semantically related words to similar embeddings. The better discrimination ability of BERT embeddings is thus likely related to their superiority as target embeddings over MLE embeddings.

8 Conclusion

In this paper, we addressed the limited effectiveness of RL for NMT, seeking to understand its origins and offer means for tackling it. We hypothesized that this limitation arises from the size of the action spaces used in NMT and examined two ways of reducing their effective dimension. In the

[8] https://spacy.io/

first method, we experiment with smaller vocabularies, showing improved RL effectiveness. While this method constrains the size of the vocabulary, which may be limiting in some settings (Ding et al., 2019; Gowda and May, 2020), it motivates further research along these lines.

The second approach introduces a new method of using informative target embeddings and potentially freezing them during RL. We find that this method may be beneficial as well, but its effectiveness crucially depends on the quality of the employed embeddings. Indeed, we find using both simulations and NMT experiments that freezing in itself results in some improvement in RL performance, but that combined with target embeddings that generalize over words with a similar distribution, it may yield substantial gains as shown by BLEU, semantic similarity, and human evaluation. We compare the target embeddings produced by MLE and those by BERT, finding the latter to be considerably stronger. Those results in low resources settings, encourage further research aiming to address the problem of large action space for TG in richer data settings by adapting and extending our methods.

Future work will increase the exploration ability of RL training in NMT. A promising line of research towards this goal is using off-policy methods. Off-policy methods, in which observations are sampled from a different policy than the one we currently optimize, are prominent in RL (Watkins and Dayan, 1992; Sutton et al., 1998), and were also studied in the context of policy gradient methods (Degris et al., 2012; Silver et al., 2014). We believe that the adoption of such methods to enhance exploration, combined with our proposed method for using target embeddings, can be a promising path forward for the application of RL in NMT, and more generally in TG.

A different line of future work will focus on changing the network's architecture to predict a d dimension continuous action, instead of discrete actions. Such an approach may directly reduce the size of the action space without limiting the number of words that can be predicted.

9 Acknowledgements

This work was supported by the Israel Science Foundation (grant no. 2424/21), and by the Applied Research in Academia Program of the Israel Innovation Authority.

References

Ondřej Bojar, Rajen Chatterjee, Christian Federmann, Yvette Graham, Barry Haddow, Shujian Huang, Matthias Huck, Philipp Koehn, Qun Liu, Varvara Logacheva, Christof Monz, Matteo Negri, Matt Post, Raphael Rubino, Lucia Specia, and Marco Turchi. 2017. Findings of the 2017 conference on machine translation (WMT17). In *Proceedings of the Second Conference on Machine Translation*, pages 169–214, Copenhagen, Denmark. Association for Computational Linguistics.

Massimo Caccia, Lucas Caccia, William Fedus, Hugo Larochelle, Joelle Pineau, and Laurent Charlin. 2018. Language gans falling short. *arXiv preprint arXiv:1811.02549*.

Yash Chandak, Georgios Theocharous, James Kostas, Scott M. Jordan, and P. S. Thomas. 2019. Learning action representations for reinforcement learning. *ArXiv*, abs/1902.00183.

Leshem Choshen, Lior Fox, Zohar Aizenbud, and Omri Abend. 2019. On the weaknesses of reinforcement learning for neural machine translation. In *International Conference on Learning Representations*.

Mathieu De Coster, Karel D'Oosterlinck, Marija Pizurica, Paloma Rabaey, Severine Verlinden, Mieke Van Herreweghe, and Joni Dambre. 2021. Frozen pretrained transformers for neural sign language translation. In *MTSUMMIT*.

Thomas Degris, Martha White, and Richard S Sutton. 2012. Off-policy actor-critic. *arXiv preprint arXiv:1205.4839*.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. Bert: Pre-training of deep bidirectional transformers for language understanding. *Proceedings of the 2019 Conference of the North*.

Shuoyang Ding, Adithya Renduchintala, and Kevin Duh. 2019. A call for prudent choice of subword merge operations. *CoRR, vol. abs/1905.10453*.

Gabriel Dulac-Arnold, Richard Evans, Hado van Hasselt, Peter Sunehag, Timothy Lillicrap, Jonathan Hunt, Timothy Mann, Theophane Weber, Thomas Degris, and Ben Coppin. 2016. Deep reinforcement learning in large discrete action spaces.

Sergey Edunov, Myle Ott, Michael Auli, David Grangier, and Marc'Aurelio Ranzato. 2018. Classical structured prediction losses for sequence to sequence learning. *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*.

Jonas Gehring, Michael Auli, David Grangier, Denis Yarats, and Yann N. Dauphin. 2017. Convolutional sequence to sequence learning.

Thamme Gowda and Jonathan May. 2020. Finding the optimal vocabulary size for neural machine translation. In *EMNLP*.

Yvette Graham, Timothy Baldwin, Alistair Moffat, and Justin Zobel. 2015. Can machine translation systems be evaluated by the crowd alone. *Natural Language Engineering*, 23:3 – 30.

Elad Hoffer, Itay Hubara, and Daniel Soudry. 2018. Fix your classifier: the marginal value of training the last weight layer. In *International Conference on Learning Representations*.

Philipp Koehn and R. Knowles. 2017. Six challenges for neural machine translation. In *NMT@ACL*.

Jaejun Lee, Raphael Tang, and Jimmy Lin. 2019. What would elsa do? freezing layers during transformer fine-tuning.

Chin-Yew Lin and Franz Josef Och. 2004. ORANGE: a method for evaluating automatic evaluation metrics for machine translation. In *COLING 2004: Proceedings of the 20th International Conference on Computational Linguistics*, pages 501–507, Geneva, Switzerland. COLING.

George A Miller. 1998. *WordNet: An electronic lexical database*. MIT press.

V. Mnih, K. Kavukcuoglu, D. Silver, A. Graves, Ioannis Antonoglou, Daan Wierstra, and Martin A. Riedmiller. 2013. Playing atari with deep reinforcement learning. *ArXiv*, abs/1312.5602.

Franz Josef Och. 2003. Minimum error rate training in statistical machine translation. In *Proceedings of the 41st annual meeting of the Association for Computational Linguistics*, pages 160–167.

Myle Ott, Sergey Edunov, Alexei Baevski, Angela Fan, Sam Gross, Nathan Ng, David Grangier, and Michael Auli. 2019. fairseq: A fast, extensible toolkit for sequence modeling. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics (Demonstrations)*, pages 48–53.

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. Bleu: a method for automatic evaluation of machine translation. In *Proceedings of the 40th annual meeting of the Association for Computational Linguistics*, pages 311–318.

Razvan Pascanu, Tomas Mikolov, and Yoshua Bengio. 2012. On the difficulty of training recurrent neural networks.

G. Pereyra, G. Tucker, J. Chorowski, L. Kaiser, and Geoffrey E. Hinton. 2017. Regularizing neural networks by penalizing confident output distributions. *ArXiv*, abs/1701.06548.

Matt Post. 2018. A call for clarity in reporting BLEU scores. In *Proceedings of the Third Conference on Machine Translation: Research Papers*, pages 186–191, Brussels, Belgium. Association for Computational Linguistics.

Marc'Aurelio Ranzato, S. Chopra, M. Auli, and W. Zaremba. 2016. Sequence level training with recurrent neural networks. *CoRR*, abs/1511.06732.

Shuo Ren, Zhirui Zhang, Shujie Liu, M. Zhou, and S. Ma. 2019. Unsupervised neural machine translation with smt as posterior regularization. *ArXiv*, abs/1901.04112.

Philip Schulz, Wilker Aziz, and Trevor Cohn. 2018. A stochastic decoder for neural machine translation. *arXiv preprint arXiv:1805.10844*.

Rico Sennrich, Barry Haddow, and Alexandra Birch. 2016. Neural machine translation of rare words with subword units. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1715–1725, Berlin, Germany. Association for Computational Linguistics.

Pararth Shah, Dilek Z. Hakkani-Tür, Bing Liu, and G. Tür. 2018. Bootstrapping a neural conversational agent with dialogue self-play, crowdsourcing and online reinforcement learning. In *NAACL-HLT*.

Sahil Sharma, Aravind Suresh, Rahul Ramesh, and Balaraman Ravindran. 2017. Learning to factor policies and action-value functions: Factored action space representations for deep reinforcement learning. *arXiv preprint arXiv:1705.07269*.

Shiqi Shen, Yong Cheng, Zhongjun He, Wei He, Hua Wu, Maosong Sun, and Yang Liu. 2015. Minimum risk training for neural machine translation. *arXiv preprint arXiv:1512.02433*.

Shiqi Shen, Yong Cheng, Zhongjun He, Wei He, Hua Wu, Maosong Sun, and Yang Liu. 2016. Minimum risk training for neural machine translation. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1683–1692, Berlin, Germany. Association for Computational Linguistics.

David Silver, Guy Lever, Nicolas Heess, Thomas Degris, Daan Wierstra, and Martin Riedmiller. 2014. Deterministic policy gradient algorithms. In *Proceedings of the 31st International Conference on Machine Learning*, volume 32 of *Proceedings of Machine Learning Research*, pages 387–395, Bejing, China. PMLR.

Ilya Sutskever, James Martens, George Dahl, and Geoffrey Hinton. 2013. On the importance of initialization and momentum in deep learning. In *International conference on machine learning*, pages 1139–1147.

Richard S Sutton, Andrew G Barto, et al. 1998. *Introduction to reinforcement learning*, volume 135. MIT press Cambridge.

Christian Szegedy, Vincent Vanhoucke, Sergey Ioffe, Jon Shlens, and Zbigniew Wojna. 2016. Rethinking the inception architecture for computer vision. *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.

Guy Tennenholtz and Shie Mannor. 2019. The natural language of actions. In *ICML*.

Brian Thompson, Huda Khayrallah, Antonios Anastasopoulos, Arya D. McCarthy, Kevin Duh, Rebecca Marvin, Paul McNamee, Jeremy Gwinnup, Tim Anderson, and Philipp Koehn. 2018. Freezing subnetworks to analyze domain adaptation in neural machine translation. *Proceedings of the Third Conference on Machine Translation: Research Papers*.

J. Tiedemann. 2012. Parallel data, tools and interfaces in opus. In *LREC*.

Emanuel Todorov, Tom Erez, and Yuval Tassa. 2012. Mujoco: A physics engine for model-based control. In *2012 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 5026–5033. IEEE.

Chaojun Wang and Rico Sennrich. 2020. On exposure bias, hallucination and domain shift in neural machine translation.

Christopher JCH Watkins and Peter Dayan. 1992. Q-learning. *Machine learning*, 8(3-4):279–292.

John Wieting, Taylor Berg-Kirkpatrick, Kevin Gimpel, and Graham Neubig. 2019. Beyond bleu: Training neural machine translation with semantic similarity. In *Proceedings of the Association for Computational Linguistics*.

Ronald J. Williams. 1992. Simple statistical gradient-following algorithms for connectionist reinforcement learning. In *Machine Learning*, pages 229–256.

Lijun Wu, Fei Tian, T. Qin, J. Lai, and T. Liu. 2018a. A study of reinforcement learning for neural machine translation. In *EMNLP*.

Lijun Wu, Fei Tian, Tao Qin, Jianhuang Lai, and Tie-Yan Liu. 2018b. A study of reinforcement learning for neural machine translation. *arXiv preprint arXiv:1808.08866*.

Wen Zhang, Y. Feng, Fandong Meng, Di You, and Qun Liu. 2019. Bridging the gap between training and inference for neural machine translation. In *ACL*.

Barret Zoph, Deniz Yuret, Jonathan May, and Kevin Knight. 2016. Transfer learning for low-resource neural machine translation.

A Methodology

Table 3 present the train, validation and test sizes in all four languages pairs. We note that our use of the data is aligned with the license and intended use of the data.

Lang.	Train	Valid	Test
de-en	284,246	6,003	2,998
cs-en	218,384	6,004	2,983
ru-em	235,159	5,999	3,000
tr-en	207,678	6,007	3,000

Table 3: Number of sentence pairs in the training/validation/test sets for all four languages.

B Entropy of STV and LTV

As C19 suggested we can compare the peakiness of the two models by calculating their distributions entropy. Lower entropy indicates a more peaky distribution. We used KL divergence with respect to the uniform distribution in order to normalize the entropy and compare the peakiness of the two models. The STV model starts RL training with mean entropy of 0.300 and finishes with 0.269 while the LTV begins with 0.258 and finishes with 0.264. This indicates that before RL training the LTV model was slightly more peaky than the STV, but after RL training they have similar peakiness.

C Loading Bert embedding

We consider two options for initializing Bert embeddings for the MLE training, with and without freezing the embedding layer. The results were unequivocal, freezing the embedding layers has a very constructive effect on the results (table 4). We estimate that freezing the embedding layers causes such a vast improvement in performance because it enables us to avoid the catastrophic forgetting of BERT parameters. Therefore, although using BERT embedding is helpful as initialization, by freezing the parameters we allow the model to better utilize BERT's embeddings.

D Number of parameters

In Table 5 we provide a comparison of the number of trainable parameters with and without freezing the embedding layer.

Model	De-En	Cs-En	Ru-En	Tr-En
MLE	22.38	15.81	17.31	12.60
MLE+Bert W/o freeze	22.99	15.32	17.57	12.65
MLE+Bert with freeze	23.46	16.59	18.14	14.15
diff.	0.47	1.27	0.57	1.50

Table 4: Comparison of MLE models with BERT embedding with and without freezing.

# parameters	De-En	Cs-En	Ru-En	Tr-En
Freeze	30.2M	29.3M	27.9M	29.4M
W/o Freeze	77.2M	76.2M	74.8M	76.4M
Ratio	0.39	0.38	0.37	0.38

Table 5: Comparison if trainable parameters.

E Formalizing the intuition behind freezing the embedding layer

Here we want to formalize the intuition behind freezing the embedding layer. We explicitly calculate the gradients of the cross-entropy (CE) loss of the one-hot vector, y, and the distribution vector, $\hat{y}$ of the model $f_\theta = h_{\theta_2} \circ g_{\theta_1}$ output (henceforth, we will discard the parameters notation from f, g and h). We will discuss two cases, one when we freeze θ_2 and the second when we are not. We note that $\theta_2 \in \mathbb{R}^{d \times |V_T|}$ is the embedding layer where each row, ρ_i, is the representation of the k's word in the vocabulary. Moreover, $h : \mathbb{R}^d \to \mathbb{R}^{|V_T|}$ is the function defined by multiplying the output of g, denoted by $v \in \mathbb{R}^d$, by θ_2, and then taking the soft-max of the output vector, hence $\forall k \in |V_T|; h_k(v) = \frac{exp(\rho_k \cdot v)}{\sum_l exp(\rho_l \cdot v)}$. Therefore assigning to each word some probability, $\hat{y}_k$, to be the next one in the sentence.

Now, we want to investigate the update defined by the gradients of the CE loss in the setting when two words, w_1 and w_2 have the same representation, $\rho_1 = \rho_2$. We consider the case where one of them is the gold token, w.l.g. w_1. We note this case by $\Big|^1$.

We turn to examine the gradient in this setting for both cases. We start by realizing that if all the partial derivatives of the CE loss, L, exist then the gradient is the vector of all the partial derivatives meaning, $\nabla_\theta L = \begin{pmatrix} \nabla_{\theta_1} L \\ \nabla_{\theta_2} L \end{pmatrix}$ and we can separate the calculation into two parts, one with respect to θ_1 and the second with respect to θ_2.

By definition, in the case where we freeze θ_2 we will keep ρ_1 and ρ_2 the same. We will now show that in the case when we don't freeze θ_2 the update will be different.

Lemma E.1. *If θ_2 is not frozen then: Updates are differe: $\Delta\rho_1 ! = \Delta\rho_2$.*

Proof. We start by noticing that multiplying v by θ_2 is a linear transformation so for points p_1 and p_2 we will get the same derivative as $\rho_1 = \rho_2$, moreover by taking the soft-max of those identical outputs we will get the same outputs. Hence, we get that $\forall i \in [d]; \frac{\partial \hat{y}_1}{\partial v_i} = \frac{\partial \hat{y}_2}{\partial v_i}$, similarly $\frac{\partial \hat{y}_1}{\partial \rho_{1i}} = \frac{\partial \hat{y}_2}{\partial \rho_{2i}}$.

We continue by calculating the derivative of the CE. The CE loss is defined by:

$$L(y, \hat{y}) = \sum_i y_i log(\hat{y}_i) \tag{3}$$

The derivative is: $\frac{\partial L}{\partial \hat{y}_i} = \sum_i y_i \frac{1}{\hat{y}_i}$ we notice that y is a one hot vector i.e., $y_1 = 1$ and $\forall i \in [2, |V_T|]; y_i = 0$. Therefore, the derivative will be different from $i = 1$ to all other i's. Specifically, $\forall i \in [d]; \frac{\partial L}{\partial \rho_{1i}} \neq \frac{\partial L}{\partial \rho_{2i}}$. Putting it all together we get:

$$\frac{\partial L}{\partial \rho_{1i}} = \frac{\partial L}{\partial \hat{y}_1} \cdot \frac{\partial \hat{y}_1}{\partial \rho_{1i}} \neq \frac{\partial L}{\partial \hat{y}_2} \cdot \frac{\partial \hat{y}_2}{\partial \rho_{2i}} = \frac{\partial L}{\partial \rho_{2i}} \tag{4}$$

Proving that ρ_1 and ρ_2 updates are different.

□

Lemma E.2. *For both cases, the update of θ_1 is symmetric to the gold being w_1 or w_2. $\nabla_{\theta_2} L\Big|^1 = \nabla_{\theta_2} L\Big|^2$.*

Proof. Given a parameter $\lambda \in \theta_1$, we inspect the derivative of L with respect to λ. We use here Einstein summation notation.

$$\frac{\partial L}{\partial \lambda}\Big|^1 = \frac{\partial L}{\partial v_i} \cdot \frac{\partial v_i}{\partial \lambda}\Big|^1 = \frac{\partial L}{\partial v_i} \cdot \frac{\partial v_i}{\partial \lambda}\Big|^2 = \frac{\partial L}{\partial \lambda}\Big|^2 \tag{5}$$

We deduce $\frac{\partial v_i}{\partial \lambda}\Big|^1 = \frac{\partial v_i}{\partial \lambda}\Big|^2$, as the derivative of v_i is independent of the question which word is the gold. In order to justify the second equality we used, we will write the derivative of L with respect to v_i.

$$\frac{\partial L}{\partial v_i}\Big|^1 = \frac{\partial L}{\partial \hat{y}_k} \cdot \frac{\partial \hat{y}_k}{\partial v_i} \tag{6}$$

Clearly, we only need to check the elements that change by switching the gold from being w_1 to w_2 or vice versa. Therefor all the second terms that multiply by $\frac{\partial L}{\partial \hat{y}_k}$ for $k \in [3, |V_T|]$ didn't change. We already proved that $\forall i \in [d]; \frac{\partial \hat{y}_1}{\partial v_i} = \frac{\partial \hat{y}_2}{\partial v_i}$ Finally, because we switch the gold, $\frac{\partial L}{\partial \hat{y}_1}$ and $\frac{\partial L}{\partial \hat{y}_2}$ indeed switch there values but both of them are multiply by the same values as $\frac{\partial \hat{y}_1}{\partial v_i} = \frac{\partial \hat{y}_2}{\partial v_i}$. Overall, the derivative is unchanged. □

To conclude, in the motivational setting we discussed, when we freeze θ_2 we keep semantically close vectors unchanged while if we don't freeze θ_2 we enable them to change. As consequence, in further steps, this change will affect on θ_1 also. In a similar manner, as long as the representation is similar, all layers but the penultimate would update both words similarly.

F Semantic scores for the second method

Our method of freezing informative initialization of the embedding layer aims to generalize across different but semantically close actions. In order to test the ability of our model to generalize we used SIM. SIM is a measure of semantic similarity that assigns partial credit to semantically correct but lexically different translations (Wieting et al., 2019). Table 6 shows our model results and exhibits similar trends to the BLEU scours. Here we see even greater gains for cs-en and ru-en languages pairs. Those results may indicate that the model was able to predict tokens that are semantically close to the gold token.

MODEL	De-En	Cs-En	Ru-En	Tr-En
MLE	70.03	63.29	66.17	59.68
+RL	71.17	63.29	66.17	59.99
+RL+FREEZE	71.03	64.29	66.66	60.52
+BERT	71.56	64.26	66.70	61.75
+BERT+RL	72.44	65.80	**67.94**	**63.59**
+BERT+RL+FREEZE	**72.81**	**66.44**	67.66	**63.59**

Table 6: SIM scores on translating four languages to English.

G Human Evaluation Information

We recruited the service of two professional translators via translations providers.

G.1 Human Evaluation Instructions

You will be shown:

1. An English segment of text;
2. Corresponding translation into English.

There are three parts to each annotation:

1. Read the English segment;
2. Read the translation and compare its meaning to the meaning of the original English segment;
3. Give a score between 0-100 describing how close the meaning of the translation is to the meaning of the original English segment.

Noise Learning for Text Classification: A Benchmark

Bo Liu
School of Sofrware Engineering, Xi'an Jiaotong University
the CETC Key Laboratory of Smart City Model Simulation and Intelligent Technology
The Smart City Research Institute of CETC and National Center for Applied Mathematics Shenzhen(NCAMS)
boliu@stu.xjtu.edu.cn

Wandi Xu
Northeastern University
xuwandi@stumail.neu.edu.cn

Yuejia Xiang
Tencent
yuejiaxiang@tencent.com

Xiaojun Wu
the CETC Key Laboratory of Smart City Model Simulation and Intelligent Technology
The Smart City Research Institute of CETC and National Center for Applied Mathematics Shenzhen(NCAMS)
wuxiaojun@cetc.com.cn

Lejian He
Cornell University
lh628@cornell.edu

Bowen Zhang
Shenzhen Technology University
zhang_bo_wen@foxmail.com

Li Zhu*
School of Sofrware Engineering, Xi'an Jiaotong University
zhuli@xjtu.edu.cn

Abstract

Noise Learning is important in the task of text classification which depends on massive labeled data that could be error-prone. However, we find that noise learning in text classification is relatively underdeveloped: 1. many methods that have been proven effective in the image domain are not explored in text classification, 2. it is difficult to conduct a fair comparison between previous studies as they do experiments in different noise settings. In this work, we adapt four state-of-the-art methods of noise learning from the image domain to text classification. Moreover, we conduct comprehensive experiments on our benchmark of noise learning with seven commonly-used methods, four datasets, and five noise modes. Additionally, most previous works are based on an implicit hypothesis that the commonly-used datasets such as TREC, Ag-News and Chnsenticorp contain no errors. However, these datasets indeed contain 0.61% to 15.77% noise labels which we define as **intrinsic noise** that can cause inaccurate evaluation. Therefore, we build a new dataset Golden-Chnsenticorp (G-Chnsenticorp) without **intrinsic noise** to more accurately compare the effects of different noise learning methods. To the best of our knowledge, this is the first benchmark of noise learning for text classification.

* Corresponding author.

1 Introduction

The fast development of text classification cannot be achieved without massive labeled data resources, especially for supervised embedding-based methods. However, not all training data are correctly labeled in practice (Wang et al., 2018; Zlateski et al., 2018). These incorrectly labeled data are called noisy labels. To alleviate the interference caused by noisy labels, many noise learning methods have been proposed (Rolnick et al., 2017; Veit et al., 2017; Jiang et al., 2018; Yang Liu, 2019; Li et al., 2020; Curtis G. Northcutt, 2020; Garg et al., 2021). Although both CV and NLP domains have serious label noise problems, these work are mainly focused on the CV domain, only (Garg et al., 2021) is dedicated to NLP domain. So in order to support the development of noise learning in NLP, we would like to propose a noise learning benchmark in the field of text classification.

We find that the previous studies in noise learning for text classification tasks have two weaknesses. 1. The implicit hypothesis is unreasonable. 2. They lack horizontal comparison.

Unreasonable implicit hypothesis. The previous research uses a four-step approach to evaluate a new method. First, they split a dataset into training data and test data, and then add manufactured noise data to the training data following a predefined noise mode. Third, they apply the noise learning method to the training data. In the end, they evaluate the noise learning method on the test data. This approach makes an implicit assumption that the dataset is completely reliable. Based on this assumption, the noise data in the experiment is equal to

Proceedings of the 29th International Conference on Computational Linguistics, pages 4557–4567
October 12–17, 2022.

Dataset	Intrinsic Noise			Total
	Fatal	Inexact	Ambiguous	
TREC	1.94%	2.58%	3.16%	7.68%
Ag-News	0.00%	0.12%	0.49%	0.61%
Chn.	2.19%	4.63%	8.95%	15.77%

Table 1: The ratio of **intrinsic noise** in several widely used datasets. Chn. denotes Chnsenticorp.

the manufactured noise data and the evaluation given the test data is accurate. But we find that the dataset is not completely noise-free and the ratio of noise data in the dataset (i.e. **intrinsic noise**, which can be divided into three parts according to the ambiguity level: fatal noise, inexact noise, and ambiguous noise) is not negligible for noise learning task (Han et al., 2020), as shown in Table 1. Thus, the noise data in the experiment is actually equal to the superposition of the **intrinsic noise** data and the manufactured noise. Hence, the evaluation in previous studies is not robust and accurate.

Lacking horizontal comparison. The previous studies lack horizontal comparison between them as they usually use different datasets, different noise modes, different noise ratios, etc. This is not conducive to the development of noise learning for text classification.

In order to overcome these two weaknesses, we build a new dataset without **intrinsic noise** and present a benchmark of noise learning for text classification. The main contributions of this paper can be summarized as follows:

- We divide the **intrinsic noise** into three parts according to the ambiguity level. To the best of our knowledge, this is the first time **intrinsic noise** to be defined and analyzed in the noise learning task of text classification.
- We propose a new dataset without **intrinsic noise**, named G-Chnsenticorp. Experiments on this dataset would have more accurate results.
- This is the first time that a benchmark of noise learning for text classification has been established. First, we summarize the noise modes mentioned in previous works. Second, we reproduce/transform seven commonly used noise learning methods in/to the text classification task.
- We have several interesting observations and conclusions: **Intrinsic Noise** is more difficult to be learned than other noise modes; many methods do not work well when the noise ratio is higher than 30%; a small amount of white noise can benefit classification methods, etc.

2 Related Work

Plenty of previous studies have examined the factors that impact label noise learning models. Zhang et al. (2016) prove that a model of sufficient complexity can over-fit any noise. Jacot et al. (2018) analyze convergence and generalization in neural networks from the perspective of Gaussian processes in the infinite-width limit. Rolnick et al. (2017) propose a model that is extremely adaptive to specific patterns of artificial noise. Li et al. (2020) prove that gradient descent with early stopping is robust to label noise for overparameterized neural networks. From the perspective of noise mode, Algan and Ulusoy (2020) conduct a detailed analysis of the influence of label noise on model training and propose a generic framework to generate feature-dependent label noise. Hataya and Nakayama (2018) investigate the behavior of Convolutional Neural Networks (CNNs) under class-dependently simulated label noise. Flatow and Penner (2017) test the robustness of the model by randomly permuting the labels of the training set with increasing frequency.Jiang et al. (2020) establish the first benchmark of controlled real-world label noise in the CV field.

Following the work of Han et al. (2020) which divides noise learning methods into three categories optimization-based method, objective-based method, and data-based method for a more comprehensive comparative analysis, we select several commonly-used noise learning methods from each category to conduct comparison experiments.

There exist a few benchmarks (Xu et al., 2018; Jiang et al., 2020) in the field of image classification, but there is no benchmark in the field of text classification. Therefore, many works in text classification task (Jindal et al., 2019; Garg et al., 2021) only do comparative experiments with their own baseline. Moreover, very few studies consider different noise settings. A robust benchmark is much needed in the development of the field of text classification.

3 G-Chnsenticorp database

Ambiguity level. We then define three categories of intrinsic noise based on the aforementioned annotator agreement thresholds: fatal noise [90%, 100%], inexact noise [60%, 90%), and ambiguous noise [0%, 60%). Specifically, when more than 90% of the annotators agree upon a label different from the original one, we consider the sample as fatal noise. When 60% to 90% of annotators agree on a label different the original, we consider the sample as inexact noise. When less than 60% of annotators agree on the label, no matter what label it is we consider it as ambiguous. Ratios of the three intrinsic noises in TREC, Ag-News and Chnsenticorp are summarized in Table 1. We adopted two annotator agreement thresholds 90% and 60% as our guidance. (Please refer to Appendix B for the selection of threshold.) Note that annotator agreement threshold here means the ratio of agreement on labeling among the annotators.

We investigate three common datasets for text classification illustrated as follows:

- TREC (Voorhees and Tice., 1999): An question

classification dataset consisting of fact-based questions divided into broad semantic categories. There are six classes. It contains 5k+ training samples.

- Ag-News (Xiang Zhang, 2015): A large-scale, four-class topic classification dataset. It contains approximately 110K training samples.

- Chnsenticorp (Tan and Zhang, 2008): A hotel review classification dataset. It contains 5K+ positive reviews and 2K+ negative reviews.

With regard to the Chnsenticorp dataset, we recruited a team of 10 experts in hotel management as annotators to label samples from Chnsenticorp and found that there existed four categories in the dataset: positive, negative, irrelevant, and neutral. We remove irrelevant and neutral samples to construct a new binary dataset Golden-Chnsenticorp (G-Chnsenticorp) since we believe those samples are ambiguous in the task of text classification.

Based on the three types of intrinsic noise, we reconstruct the G-Chnsenticorp dataset as follows: we correct samples of fatal noise with the annotator majority label and remove samples of inexact or ambiguous noise. Note that since G-Chnsenticorp is a simple binary text sentiment classification dataset, we are confident to correct the fatal noise labels with our expert majority label and remove the other two types of noise. Ambiguous noise samples such as "Good breakfast but bad bed" are indeed noise to the binary classification dataset and challenging samples such as sarcastic reviews are not classified as ambiguous or inexact noise during annotating. The new dataset can ensure the robustness of the models trained on it. Through our annotating and modification as mentioned above, we obtain the final G-Chnsenticorp dataset which contains around 4,000 training samples.

4 Noise Generation Methods

In single-label text classification tasks, we assign a corresponding label to each sentence. For all n samples with k different types of classes, we let $D = \{(x_1, y_1), (x_2, y_2), \ldots, (x_n, y_n)\}$ as noise-free dataset, where x_i denotes the ith sentence in the dataset, $y_i \in \{1, \ldots, k\}$ denotes the class of the ith sentence. However, it is hard for us to find truly noise-free data in the dataset except for manual verification. Therefore, we first assume all pre-given data are true. Then, we use noise transfer matrix to automatically generate the corresponding noise-labeled dataset. Here, we set $D^{'} = \{(x_1, y_1^{'}), (x_2, y_2^{'}), \ldots, (x_n, y_n^{'})\}$ as the noisy dataset, where $y_i^{'}$ denotes the corresponding noise label of sentence x_i. The noise transfer matrix $\Phi(y, y^{'})$ represents the transfer distribution of the true label y and the noise label $y^{'}$, which is a $k \times k$ matrix.

Under the same assumption in other studies (Aritra Ghosh, 2017; Patrini et al., 2017; Jindal et al., 2019), the noise label $y_i^{'}$ only depends on the corresponding true label y_i, but not the input x_i or the other labels y_j or $y_j^{'}$. In our experiment, we use the noise transfer matrix Φ to generate the corresponding noise labels for the training set, but labels in the test set are not changed. Meanwhile, we use p to denote the noise rate, which is the overall probability of label errors, where $0 \leq p \leq 1$.

Noise Mode	Noise Generation Formula
Symmetric Noise	$\Phi = (1-p)I + \frac{p}{k}A$
Pairflip Noise	$\Phi = (1-p)I + pB$
Uniform Noise	$\Phi = (1-p)I + \frac{p}{k}C$
Random Noise	$\Phi = (1-p)I + pD$
White Noise	other unrelated field text

Table 2: Different generation formala of noise mode. Here, I represents the identity matrix. A denotes an all-ones matrix with zeros along the diagonal. B represents the identity matrix where the last column is transferred to the first column. C represents the matrix with zeros along the diagonal, and except for the diagonal, the values are uniformly and independently distributed. D is a matrix independent of the $k-1$ dimensional unit simplex with zeros along the diagonal.

Generally speaking, noise labels can be categorized into four types according to different noise transfer matrices. As shown in Table 2 and Figure 1: (1) Symmetric Noise (Van Rooyen et al., 2015); (2) Pairflip Noise (Han et al., 2018); (3) Random Noise (Garg et al., 2021); (4)Uniform Noise (Garg et al., 2021). Here, if i means the original category and j means the convered category, Φ[i][j] represents the probability of class i becoming to class j. Additionally, we define a new type of noise named White Noise. Referring to the practice in the image field (Rolnick et al., 2017), we first collect different fields of text data and generate white noise by randomly labeling the labels.

80%	6.7%	6.7%	6.7%
6.7%	80%	6.7%	6.7%
6.7%	6.7%	80%	6.7%
6.7%	6.7%	6.7%	80%

(a) Symmetric Noise

80%	20%	0%	0%
0%	80%	20%	0%
0%	0%	80%	20%
20%	0%	0%	80%

(b) Pairflip Noise

80%	12.9%	1.7%	5.4%
2.7%	80%	11.1%	6.2%
13.6%	5.0%	80%	1.4%
13.8%	0.1%	6.1%	80%

(c) Random Noise

80%	3.8%	10.1%	6.1%
0.5%	80%	2.6%	16.9%
6.7%	5.6%	80%	7.7%
8.2%	8.7%	3.1%	80%

(d) Uniform Noise

Figure 1: Transition matrices of different noise types (using 4 classes and noise rate p=0.2 as an example).

5 Method

Common noise learning methods can be divided into three categories: optimization-based method, objective-

based method and data-based method (Han et al., 2020).

5.1 Optimization-based Method

Optimization-based methods use two networks to make predictions on the same mini-batch data and calculate a joint loss with Co-Regularization for each training example. For the optimization policy, the key is to explore the dynamic process of optimization, which relates to memorization. (Han et al., 2020)

Here, we modify the Co-teaching (Han et al., 2018), Co-teaching+ (Yu et al., 2019) and JoCoR (Wei et al., 2020) frameworks to make the data and models compatible with natural language processing tasks. All of them are originally used in the field of computer vision.

- Co-teaching: it trains two networks simultaneously. In each batch data, both networks select their small-loss samples to cross-update parameters of the other network.
- Co-teaching+: it trains two networks simultaneously, too. Different from selecting all small-loss data in Co-teaching, Co-teaching+ only keeps prediction discrepancy data in the two networks.
- JoCoR: it also trains two networks, but updates parameters with a joint loss. To reduce divergence between two networks, JoCoR uses the joint loss and sampling discrepancy data to backward propagate in a whole.

For feature representation layers, we use three types of networks to extract features, which are FNN, CNN, and BERT.

- Co-teaching$_{FNN}$: based on the Co-teaching method, we use three-layer feedforward neural networks to extract features.
- Co-teaching$_{CNN}$: based on the Co-teaching method, we use CNN for feature representation.
- Co-teaching$_{BERT}$: based on the Co-teaching method, we use BERT for feature extraction.

5.2 Objective-based Method

The objective-based methods learn from noisy data by modifying the objective function. Specifically, the key is to design a suitably modified loss, which is noise-tolerant and guarantees statistical consistency compared to the original loss (Han et al., 2020).

Here we select LSTM$_{DN-H}$ (Garg et al., 2021), LSTM$_{DN-S}$ (Garg et al., 2021) and Peer (Yang Liu, 2019) to modify the loss function. In these three approaches, only Peer is originally used in the field of natural language processing.

- LSTM$_{DN-H}$: the network first assigns a probability score to each training data by a beta mixture model clustering the losses at an early epoch of training. Then the network is trained with these scores using the joint loss l_{DN-H}.

$$l_{DN-H} = l_{CE}(\hat{y}^{(n)}, y) + \beta \cdot B(x) \cdot l_{CE}(\hat{y}^{(c)}, y)$$

where $\hat{y}^{(n)}$ denotes the noisy label prediction, $\hat{y}^{(c)}$ denotes the clean label prediction, y denotes the input in training dataset, l_{CE} denotes the cross entropy loss, B(x) denotes the posterior probability that x has a clean label, and β is a weighting parameter between the two terms.

- LSTM$_{DN-S}$: similar to the LSTM$_{DN-H}$. The only difference is that the network uses an alternative formulation by replacing the Bernoulli R.V. B(x) with the indicator $\mathbb{1}[B(x) > 0.5]$. The l_{DN-S} loss function is as follows:

$$l_{DN-S} = l_{CE}(\hat{y}^{(n)}, y) + \beta \cdot \chi \cdot l_{CE}(\hat{y}^{(c)}, y)$$

where χ denotes the indicator $\mathbb{1}[B(x) > 0.5]$. In the experiments of LSTM$_{DN-H}$ and LSTM$_{DN-S}$ model,we use pretrained word2vec embeddings and lstm neural network layer to extract features.

- Peer: introduces a new family of loss functions called peer loss functions.This method enables training a classifier over noisy labels without using explicit knowledge of the noise rates of labels. The l_{peer} loss function is as follows:

$$l_{peer}(f(x_j), y_j) = l_1(f(x_j), y_j) - \alpha l_2(f(x_{j_1}), y_{j_2})$$

where $alpha$ is non-zero real number hyper-parameter. For each sample (x_j, y_j), randomly draw other two samples (x_{j_1}, y_{j_1}), (x_{j_2}, y_{j_2}) such that $j_1 \neq j_2$. These two samples are called the peer sample. f represents the bayes optimal classifier. l_1 and l_2 can be any standard classification-calibrated loss function, such as cross entropy loss and mean square error loss. In our experiments, we use BERT (Devlin et al., 2019) as the feature extraction layer.

5.3 Data-based Method

For data-based methods, we aim to discover the underlying noise transition pattern. The noise transfer matrix allows us to find the relationship between the clean label and the noisy label. Therefore, the key point here is to design an accurate estimator of the noise transition matrix. In this method, we select the Confident learning approach that is originally used in computer vision to solve this problem.

Confident learning (CL) is an alternative approach that instead focuses on the label quality by characterizing and identifying label errors in datasets. Based on the principles of pruning noisy data, we count with probabilistic thresholds to estimate noise and rank examples to train with confidence (Curtis G. Northcutt, 2019). The CL model inferred which samples are noisy by obtaining the predicted probabilities of the samples on different classifications. In other words, the predicted probabilities are the feature of the CL model to discriminate noisy labels. In our experiments,we use three-layer feedforward neural networks to extract predicted probabilities of the samples.

			Symmetric Noise						Pairflip Noise					
TREC	Method	Clean Data	10%	20%	30%	40%	50%	60%	10%	20%	30%	40%	50%	60%
	Co-teaching	88.40%	85.00%	82.20%	80.80%	73.80%	66.80%	64.00%	85.00%	79.60%	74.40%	53.60%	33.00%	15.80%
	Co-teaching+	84.60%	84.00%	82.40%	81.20%	78.80%	74.20%	68.40%	83.80%	84.00%	77.40%	72.40%	46.60%	29.40%
	JoCoR	84.80%	83.20%	80.20%	80.80%	77.60%	67.20%	63.80%	84.40%	78.40%	75.80%	48.20%	31.00%	28.60%
	LSTM$_{DN-H}$	94.20%	**92.20%**	89.20%	85.80%	83.30%	82.40%	81.10%	91.90%	88.80%	85.60%	84.50%	83.00%	**81.50%**
	LSTM$_{DN-S}$	**94.40%**	**92.20%**	**90.70%**	**87.80%**	**84.80%**	**83.20%**	**82.00%**	**92.10%**	**90.20%**	**88.30%**	**86.30%**	**83.40%**	**81.50%**
	Peer	78.44%	77.52%	75.03%	73.84%	73.11%	71.33%	64.89%	76.99%	75.38%	75.14%	73.03%	66.80%	27.41%
	CL	82.63%	83.57%	84.57%	81.69%	77.35%	70.74%	61.32%	83.17%	77.56%	72.34%	54.11%	30.26%	22.04%
			White Noise						Random Noise					
	Method	Clean Data	10%	20%	30%	40%	50%	60%	10%	20%	30%	40%	50%	60%
	Co-teaching	88.40%	85.80%	86.40%	87.40%	85.40%	83.00%	81.40%	83.80%	81.80%	75.80%	70.40%	67.60%	47.80%
	Co-teaching+	84.60%	83.00%	84.20%	84.40%	82.60%	82.20%	83.40%	83.20%	82.80%	80.80%	72.40%	69.80%	60.60%
	JoCoR	84.80%	85.40%	85.60%	82.60%	82.60%	82.00%	82.80%	84.40%	82.60%	75.60%	72.80%	69.20%	63.00%
	LSTM$_{DN-H}$	94.20%	94.30%	**94.40%**	**94.30%**	**94.00%**	93.50%	**93.80%**	92.00%	89.60%	86.50%	83.40%	**82.00%**	**81.40%**
	LSTM$_{DN-S}$	**94.40%**	**94.40%**	94.20%	94.20%	93.80%	**93.80%**	93.60%	**92.20%**	**91.10%**	**88.80%**	**83.50%**	81.80%	**81.40%**
	Peer	78.44%	77.15%	77.35%	76.93%	75.21%	76.60%	76.73%	74.74%	71.23%	70.05%	68.02%	65.31%	58.98%
	CL	82.63%	83.15%	83.17%	81.74%	74.80%	71.18%	65.36%	83.37%	80.56%	78.76%	70.34%	61.72%	36.87%
			50% Symmetric Noise + 50% Pairflip Noise						50% Random Noise + 50% Pairflip Noise					
	Method	Clean Data	10%	20%	30%	40%	50%	60%	10%	20%	30%	40%	50%	60%
	Co-teaching	88.40%	82.40%	81.20%	77.20%	74.20%	65.40%	47.00%	86.80%	80.20%	76.60%	70.40%	61.00%	36.20%
	Co-teaching+	84.60%	82.80%	81.40%	80.20%	72.40%	70.20%	60.60%	85.00%	78.40%	77.80%	75.40%	68.40%	30.80%
	JoCoR	84.80%	85.00%	81.20%	78.80%	75.00%	67.80%	44.00%	84.80%	80.40%	78.60%	68.60%	62.60%	34.40%
	LSTM$_{DN-H}$	94.20%	91.10%	88.70%	85.30%	83.10%	82.00%	81.00%	**91.80%**	88.50%	86.30%	**84.30%**	**83.60%**	81.00%
	LSTM$_{DN-S}$	**94.40%**	**91.60%**	**88.90%**	**86.70%**	**84.40%**	**82.80%**	**81.30%**	91.40%	**89.70%**	**88.20%**	84.10%	82.80%	**81.10%**
	Peer	78.44%	78.07%	77.65%	75.11%	73.34%	68.92%	45.20%	76.23%	74.88%	73.26%	65.06%	53.73%	28.19%
	CL	82.63%	83.20%	80.76%	77.59%	72.20%	62.31%	48.24%	84.19%	81.47%	74.33%	71.22%	60.09%	33.35%
			Symmetric Noise						Pairflip Noise					
Ag-News	Method	Clean Data	10%	20%	30%	40%	50%	60%	10%	20%	30%	40%	50%	60%
	Co-teaching	78.43%	76.95%	75.55%	74.86%	72.99%	69.33%	63.54%	77.51%	76.01%	69.59%	65.51%	47.80%	15.64%
	Co-teaching+	76.88%	76.59%	75.84%	75.53%	74.62%	71.82%	67.46%	76.83%	75.46%	74.26%	69.53%	47.08%	15.66%
	JoCoR	77.92%	77.00%	76.17%	74.45%	72.96%	70.12%	62.95%	76.99%	75.12%	70.22%	62.39%	38.14%	15.51%
	LSTM$_{DN-H}$	**93.31%**	91.54%	91.24%	91.01%	88.53%	87.92%	87.66%	91.56%	90.94%	90.55%	88.03%	87.68%	87.45%
	LSTM$_{DN-S}$	**93.31%**	**91.77%**	**91.48%**	**91.07%**	**89.42%**	**88.79%**	**88.52%**	**91.75%**	**91.30%**	**90.87%**	**89.52%**	**88.90%**	**88.37%**
	Peer	74.03%	73.77%	72.68%	72.60%	71.00%	70.59%	65.39%	73.35%	72.16%	71.63%	67.08%	36.11%	17.52%
	CL	80.30%	78.58%	69.97%	63.21%	55.19%	46.86%	37.43%	77.23%	69.32%	61.56%	54.33%	46.12%	36.28%
			White Noise						Uniform Noise					
	Method	Clean Data	10%	20%	30%	40%	50%	60%	10%	20%	30%	40%	50%	60%
	Co-teaching	78.43%	78.46%	78.41%	78.18%	78.05%	77.25%	76.05%	77.07%	76.34%	74.54%	72.61%	63.42%	62.75%
	Co-teaching+	76.88%	77.02%	76.66%	77.07%	76.45%	76.33%	76.86%	76.59 %	76.16%	75.80%	74.11%	71.79%	64.99%
	JoCoR	77.92%	77.88%	77.87%	77.68%	77.39%	76.17%	74.55%	77.09%	75.80%	73.79%	71.42%	67.59%	60.58%
	LSTM$_{DN-H}$	**93.31%**	**93.34%**	**93.25%**	**93.24%**	**93.20%**	92.89%	**93.12%**	91.59%	91.20%	90.83%	90.05%	89.74%	88.51%
	LSTM$_{DN-S}$	**93.31%**	**93.34%**	93.18%	93.11%	93.15%	**93.04%**	93.07%	**91.86%**	**91.44%**	**91.01%**	**90.23%**	**89.88%**	**88.63%**
	Peer	74.03%	74.25%	74.11%	73.68%	73.02%	72.43%	72.16%	73.87%	73.68%	72.10%	71.78%	70.31%	63.80%
	CL	80.30%	80.46%	78.73%	77.75%	74.27%	72.68%	67.35%	78.51%	69.72%	62.82%	54.06%	46.72%	37.18%
			50% Symmetric Noise + 50% Pairflip Noise						50% Uniform Noise + 50% Pairflip Noise					
	Method	Clean Data	10%	20%	30%	40%	50%	60%	10%	20%	30%	40%	50%	60%
	Co-teaching	78.43%	76.45%	75.30%	73.63%	71.24%	63.91%	40.82%	78.43%	77.09%	75.94%	73.53%	63.34%	46.24%
	Co-teaching+	76.88%	76.96%	75.79%	74.89%	73.41%	67.49%	38.71%	76.74%	75.99%	74.25%	74.00%	66.46%	43.21%
	JoCoR	77.92%	76.92%	75.17%	74.39%	75.64%	63.03%	75.95%	76.39%	75.64%	72.95%	70.00%	62.72%	47.63%
	LSTM$_{DN-H}$	**93.31%**	91.43%	90.87%	90.16%	87.95%	87.66%	87.12%	91.86%	91.47%	91.09%	90.03%	**89.85%**	**89.37%**
	LSTM$_{DN-S}$	**93.31%**	**91.59%**	**91.25%**	**90.63%**	**89.04%**	**88.20%**	**87.64%**	**92.31%**	**91.85%**	**91.20%**	**90.86%**	89.57%	89.30%
	Peer	74.03%	73.77%	73.50%	72.37%	70.74%	62.59%	46.49%	73.01%	73.16%	72.20%	71.29%	63.63%	44.93%
	CL	80.30%	77.63%	68.42%	60.30%	54.89%	45.21%	36.85%	78.69%	71.37%	64.49%	56.74%	47.55%	39.64%
			Symmetric Noise / Pairflip Noise						White Noise					
Chnsenticorp	Method	Clean Data	10%	20%	30%	40%	50%	60%	10%	20%	30%	40%	50%	60%
	Co-teaching	72.05%	66.88%	69.98%	64.39%	63.35%	55.28%	38.3%	70.18%	71.22%	68.12%	69.98%	68.94%	67.08%
	Co-teaching+	71.42%	69.98%	69.98%	63.35%	67.49%	52.17%	38.92%	71.84%	66.87%	71.64%	69.98%	69.77%	68.32%
	JoCoR	73.5%	68.94%	70.39%	65.01%	61.49%	**55.90%**	37.06%	69.77%	72.67%	70.6%	70.39%	69.36%	69.15%
	LSTM$_{DN-H}$	59.42%	58.59%	57.26%	56.17%	54.29%	53.46%	**52.23%**	59.42%	59.36%	59.21%	59.14%	59.07%	59.10%
	LSTM$_{DN-S}$	59.62%	59.00%	57.83%	56.33%	64.60%	53.55%	52.13%	59.62%	59.62%	59.61%	59.59%	59.57%	59.53%
	Peer	75.63%	73.23%	71.17%	65.31%	60.69%	42.35%	31.02%	76.11%	76.82%	73.25%	72.07%	71.53%	71.83%
	CL	**88.17%**	**87.55%**	**85.47%**	**74.27%**	**73.03%**	47.51%	28.22%	**88.59%**	**88.38%**	**88.80%**	**86.10%**	**89.00%**	**87.55%**
			Symmetric Noise / Pairflip Noise						White Noise					
G-Chnsenticorp	Method	Clean Data	10%	20%	30%	40%	50%	60%	10%	20%	30%	40%	50%	60%
	Co-teaching	75.98%	72.05%	71.01%	69.57%	66.05%	48.45%	32.20%	76.81%	75.16%	74.12%	75.36%	75.78%	75.36%
	Co-teaching+	76.19%	73.08%	71.01%	69.77%	64.80%	54.04%	35.40%	74.95%	74.95%	74.21%	76.81%	75.16%	75.57%
	JoCoR	75.78%	76.19%	70.60%	69.15%	63.56%	56.11%	38.92%	78.05%	74.32%	73.71%	75.57%	73.08%	72.26%
	LSTM$_{DN-H}$	62.11%	61.90%	59.61%	57.43%	55.15%	53.40%	**52.10%**	62.11%	62.06%	62.01%	61.98%	61.83%	61.78%
	LSTM$_{DN-S}$	62.31%	62.05%	59.16%	57.42%	54.84%	52.96%	51.89%	62.11%	62.07%	61.90%	61.75%	61.90%	61.85%
	Peer	79.15%	77.83%	74.26%	70.89%	68.35%	**60.32%**	37.11%	79.83%	79.02%	78.83%	78.49%	77.10%	76.60%
	CL	**95.44%**	**91.49%**	**85.06%**	**79.25%**	**70.95%**	50.83%	28.63%	**95.85%**	**94.81%**	**94.19%**	**95.43%**	**93.98%**	**95.02%**

Table 3: The accuracy of seven different models trained on the four datasets with different noise ratios and noise modes respectively

6 Results and Discussion

In our experiments, we evaluate the performance of the models in Table 3 on accuracy which is a widely-used metric in noise learning. We consider five different noise modes (random, symmetric, pairflip, uniform, and white) and some of their combinations on four datasets. We compare the performance of seven widely-used methods and five of them are originally used in the field of computer vision. Hence, we transform them to adapt to the task of text classification. To examine the robustness of the proposed approaches, we set the noise ratio from 0% to 60%. In detail, $p \in \{0.0, 0.1, 0.2, 0.3, 0.4, 0.5, 0.6\}$.

6.1 Effects of Noise Mode

For single mode of noise We find that when white noise is included, the results are comparable with those of the clean dataset. Even if the ratio of noise reaches the maximum, the results are still comparable. We believe that this is because neural networks learn features from text contents. Since the white-noise texts are not related to the original datasets, the results are not greatly affected.

For multi-class classification problems We find that in most cases when the noise ratio increases to 30%, the accuracy of models with pairflip noise is significantly lower than others. We argue that the main reason here is the transfer matrix. When the number of classes exceeds two, the labels with pairflip noise can only transfer to a fixed category or remain unchanged. But for symmetric, uniform, and random transfer matrices, there is a certain possibility of transferring to each category. When the number of classes is two, pairflip transfer matrix is the same as other transfer matrices. Therefore, models involving pairflip noise may have worse accuracy.

For different combinations of noise modes We select symmetric, uniform, pairflip, and random noises to do combinations. We set the total noise ratio from 10% to 60% and assign an equal portion to each noise of the combinations. For most combinations, we find that when combining two noise modes which generally have high classification accuracy, the accuracy of the combination is usually lower than that of the single noise mode. Interestingly, when two noise modes with low accuracy in each single mode are combined, the accuracy may be higher than their single mode's results. For example, in Ag-News we find that choosing the combination of 50% uniform and 50% pairflip noise can achieve better results than their single mode of noise.

6.2 Effects of Noise Rate

For all four datasets, we find that as the noise ratio increases, the performance gradually decreases in most cases. Especially when the noise ratio exceeds 30%, the accuracy drops significantly. However, there are a few exceptions. For instance, including less than 20% of symmetric or white noises would lead to higher accuracy than the clean dataset. We think this may be due to the network's robustness as suggested in (Rolnick et al., 2017) . For symmetric noise, different from other noise transfer matrices, its matrix is an equal division of probabilities except for the diagonal. Then, labels flip with a small equal probability. Due to the fault-tolerance of neural networks, the accuracy can be high. For white noise, when adding a small number of irrelevant texts, the model is consistent and better in its predicted results.

Noise Mode	Noise Ratio %				
	5	10	15	20	25
symmetric	75.97%	73.24%	72.05%	71.52%	71.01%
white	76.03%	76.32%	76.81%	75.23%	75.17%
intrinsic noise	75.83%	70.18%	68.42%	65.97%	63.09%

Table 4: Accuracy of Co-teaching with different noise modes and ratios on G-chnsenticorp

6.3 Effects of Method

For all three categories methods mentioned in Section 5, we find that the data-based method achieves high accuracy in Chnsenticorp and G-Chnsenticorp. The goal of CL is to discover the underlying noise transition pattern which is closer to our noise generation approaches. Therefore, CL can get more accurate results. Compared to Chnsenticorp and G-Chnsenticorp, TREC and Ag-News have more categories than them. This may be the reason for the decreasing results on TREC and Ag-News.

For TREC and Ag-News, the objective-based method, especially LSTM$_{DN-H}$ and LSTM$_{DN-S}$, performs better than other methods. We think that the model can identify the wrong labels by the sample loss value of the training process via modifying the loss function. Hence, based on a suitably constructed loss, it can train a robust deep classifier from the noisy training data and thus can assign correct labels on clean test data.

Among Co-teaching, Co-teaching+, and JoCoR approaches, Co-teaching+ achieves the best result. We argue that this noise learning approach can capture an arbitrary noise function so it can predict a more precise result.

6.4 Effects of Dataset

Compared with Chnsenticorp and G-Chnsenticorp, we find the results on G-Chnsenticorp are significantly better than those on Chnsenticorp. There may be two reasons for this. First, the imbalance of label distribution in Chnsenticorp may affect the results. Second, **intrinsic noise** is an important influencing factor. Interestingly, for results on symmetric or pairflip noise, we find Chnsenticorp has worse results than G-Chnsenticorp at first. But as the noise ratio exceeds 40%, the accuracy on G-Chnsenticorp drops even faster than Chnsenticorp.

Models	Noise Ratio %				
	10	20	30	40	50
Co-teaching$_{FNN}$	83.80%	81.80%	75.80%	70.40%	67.60%
Co-teaching$_{CNN}$	83.34%	80.07%	74.56%	65.44%	62.03%
Co-teaching$_{BERT}$	87.27%	83.77%	77.92%	74.35.67%	70.03%

Table 5: Accuracy of models with different complexities and ratios of random noise on G-chnsenticorp

6.5 Effects of Intrinsic Noise

Because we completely check and relabel the Chnsenticorp dataset, we can analyze the impact of **intrinsic noise** and artificial noise (e.g. symmetric) on the model. We gradually add noise samples to G-chnsenticorp to examine the effects of different **intrinsic noise** ratios on the performance of Co-teaching model.

According to Table 4, we can see that the accuracy with **intrinsic noise** is higher than that of artificial noise. This is because **intrinsic noise** contains more uncertainty than artificial noise which has a definite pattern. There can be a number of reasons for incorrect annotating of data: ambiguity of the correct label (Zhan et al., 2019), annotation speed, human errors, inexperience of annotator, etc. The noise labels generated by these behaviors have no patterns. It is more difficult for the model to capture and counteract these noise labels. Thus, the accuracy of the model with intrinsic noise is negatively affected.

6.6 Effects of Model Complexity

We also examine the influence of model complexity on its performance using different feature extraction layers. We experiment on G-chnsenticorp with random noise. Intuitively, models with different complexities should have different tolerances for noise data. Results in Table 5 also illustrate this: the accuracy of CNN and FNN drops faster than that of BERT as the noise ratio increases. Since the BERT model is pre-trained on a large-scale corpus, it has better generalization ability and can better combat the interference from noise data.

7 Conclusion

In conclusion, we firstly construct a text classification dataset without **intrinsic noise**, then do experiments on these datasets using some sota noise learining methods, and finally draw some useful conclusions about noise learning: **Intrinsic Noise** is more difficult to be learned than other noise modes; many methods do not work well when the noise ratio is higher than 30%; a small amount of white noise can benefit classification methods. This is the first time a benchmark of noise learning for text classification has been established. We construct a dataset without **intrinsic noise** for more accurate evaluations in the noise learning. We present this benchmark to summarize and compare the contributions and weaknesses of previous work in noise learing, to make up for their lack in **intrinsic noise** analysis, and hopefully to provide a reference for future research in the field of noise learning.

In future work, we will build more data without **intrinsic noise** and conduct more in-depth analysis of **intrinsic noise** in other settings. Of course, we will also research a better noise learing method based on the experimental findings of this paper.

Acknowledgements

This work is supported by national key research and development project, No. 2019YFB2102500, National Nature Science Foundations of China under Grant U20B2052.

References

Görkem Algan and Ilkay Ulusoy. 2020. Label noise types and their effects on deep learning. *arXiv preprint arXiv:2003.10471*.

P.S. Sastry Aritra Ghosh, Himanshu Kumar. 2017. Robust loss functions under label noise for deep neural networks. *arXiv preprint arXiv:1712.09482*.

Isaac L. Chuang Curtis G. Northcutt, Lu Jiang. 2019. Confident learning: Estimating uncertainty in dataset labels. *arXiv preprint arXiv:1911.00068*.

Jessy Lin Curtis G. Northcutt, Anish Athalye. 2020. Pervasive label errors in ml benchmark test sets,consequences, and benefits. In *Workshop on Dataset Curation and Security - NeurIPS 2020*.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. Bert: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186.

David Flatow and Daniel Penner. 2017. On the robustness of convnets to training on noisy labels.

Siddhant Garg, Goutham Ramakrishnan, and Varun Thumbe. 2021. Towards robustness to label noise in text classification via noise modeling. *arXiv preprint arXiv:2101.11214*.

Bo Han, Quanming Yao, Tongliang Liu, Gang Niu, Ivor W Tsang, James T Kwok, and Masashi Sugiyama. 2020. A survey of label-noise representation learning: Past, present and future. *arXiv preprint arXiv:2011.04406*.

Bo Han, Quanming Yao, Xingrui Yu, Gang Niu, Miao Xu, Weihua Hu, Ivor Tsang, and Masashi Sugiyama. 2018. Co-teaching: Robust training of deep neural networks with extremely noisy labels. *arXiv preprint arXiv:1804.06872*.

Ryuichiro Hataya and Hideki Nakayama. 2018. Investigating cnns' learning representation under label noise.

Arthur Jacot, Franck Gabriel, and Clément Hongler. 2018. Neural tangent kernel: Convergence and generalization in neural networks. *arXiv preprint arXiv:1806.07572*.

Lu Jiang, Di Huang, Mason Liu, and Weilong Yang. 2020. Beyond synthetic noise: Deep learning on controlled noisy labels. In *International Conference on Machine Learning*, pages 4804–4815. PMLR.

Lu Jiang, Zhengyuan Zhou, Thomas Leung, Li-Jia Li, and Li Fei-Fei. 2018. Mentornet: Learning data-driven curriculum for very deep neural networks on corrupted labels. In *International Conference on Machine Learning*, pages 2304–2313. PMLR.

Ishan Jindal, Daniel Pressel, Brian Lester, and Matthew Nokleby. 2019. An effective label noise model for dnn text classification. *arXiv preprint arXiv:1903.07507*.

Mingchen Li, Mahdi Soltanolkotabi, and Samet Oymak. 2020. Gradient descent with early stopping is provably robust to label noise for overparameterized neural networks. In *International Conference on Artificial Intelligence and Statistics*, pages 4313–4324. PMLR.

Giorgio Patrini, Alessandro Rozza, Aditya Krishna Menon, Richard Nock, and Lizhen Qu. 2017. Making deep neural networks robust to label noise: A loss correction approach. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1944–1952.

David Rolnick, Andreas Veit, Serge Belongie, and Nir Shavit. 2017. Deep learning is robust to massive label noise. *arXiv preprint arXiv:1705.10694*.

Songbo Tan and Jin Zhang. 2008. An empirical study of sentiment analysis for chinese documents. *Expert Systems with applications*, 34(4):2622–2629.

Brendan Van Rooyen, Aditya Krishna Menon, and Robert C Williamson. 2015. Learning with symmetric label noise: The importance of being unhinged. *arXiv preprint arXiv:1505.07634*.

Andreas Veit, Neil Alldrin, Gal Chechik, Ivan Krasin, Abhinav Gupta, and Serge Belongie. 2017. Learning from noisy large-scale datasets with minimal supervision. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 839–847.

Ellen M Voorhees and Dawn M Tice. 1999. The trec8 question answering track evaluation. In *TREC*, volume 1999, page 82.

Fei Wang, Liren Chen, Cheng Li, Shiyao Huang, Yanjie Chen, Chen Qian, and Chen Change Loy. 2018. The devil of face recognition is in the noise. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 765–780.

Hongxin Wei, Lei Feng, Xiangyu Chen, and Bo An. 2020. Combating noisy labels by agreement: A joint training method with co-regularization. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 13726–13735.

Yann LeCun Xiang Zhang, Junbo Zhao. 2015. Character-level convolutional networks for text classification. *arXiv preprint arXiv:1509.01626*.

Jun Xu, Hui Li, Zhetong Liang, David Zhang, and Lei Zhang. 2018. Real-world noisy image denoising: A new benchmark. *arXiv preprint arXiv:1804.02603*.

Hongyi Guo Yang Liu. 2019. Peer loss functions: Learning from noisy labels without knowing noise rates. *arXiv preprint arXiv:1910.03231*.

Xingrui Yu, Bo Han, Jiangchao Yao, Gang Niu, Ivor Tsang, and Masashi Sugiyama. 2019. How does disagreement help generalization against label corruption? In *International Conference on Machine Learning*, pages 7164–7173. PMLR.

Xueying Zhan, Yaowei Wang, Yanghui Rao, and Qing Li. 2019. Learning from multi-annotator data: A noise-aware classification framework. *ACM Transactions on Information Systems (TOIS)*, 37(2):1–28.

Chiyuan Zhang, Samy Bengio, Moritz Hardt, Benjamin Recht, and Oriol Vinyals. 2016. Understanding deep learning requires rethinking generalization. *arXiv preprint arXiv:1611.03530*.

Aleksandar Zlateski, Ronnachai Jaroensri, Prafull Sharma, and Frédo Durand. 2018. On the importance of label quality for semantic segmentation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1479–1487.

Appendices

A Cases of Intrinsic Noise

Table 6 shows some cases of **Intrinsic Noise**.

B Selection of Threshold

During human annotation, we analyze the annotation results of our 10 annotators. Based on the annotation accuracy on the annotators' emotional inclination (the majority agree on one label), we choose 90% and 60% as the annotator agreement thresholds using the 3σ principle of Normal Distribution.

First, we calculate the rate of annotators' agreement on the annotated labels. We find that the accuracy of annotators getting the labels with significantly emotional inclination correct is 92.08%. With regard to those labels without significantly emotional inclination, we assume that the probability of annotating the label positive or negative is 50%. The rate of annotators' agreement on the annotated labels is illustrated as Table 7. The details results are summarized in Table 8.

Based on the annotator agreement rates on emotionally inclined or not inclined samples, we can compute the accuracy of getting those samples correct as follows:

$$\begin{aligned} accuracy(a_i) &= \frac{a_i}{a_i + b_i} \\ accuracy(b_i) &= \frac{b_i}{a_i + b_i} \end{aligned} \tag{1}$$

where a_i and b_i denote the annotator agreement rate on emotionally inclined or not inclined samples respectively.

The accuracy is summarized in Table 9.

We choose the 95% Confidence interval for our results and have the following definitions on intrinsic noise:

As more than 90% of annotations are the same among the annotators, the original sample is significantly emotionally inclined (accuracy > 95%). If the majority annotation is different from the original label, we consider the sample as Fatal noise.

As less than 60% of annotations are the same among the annotators, the original sample is not significantly emotionally inclined (accuracy > 95%). Not matter what the original label is, we consider it as Ambiguous noise.

As 60% to 90% of annotations are the same among the annotators, the original sample is inexact in terms of emotional inclination. If the majority annotation is different from the original label, we consider the sample as Inexact noise.

C Effects of Hyperparameter Settings

To explore other different settings, we conduct experiments using the following uniform setup: Co-teaching model trained the fixed TREC dataset with different ratios of random noise. Here, we explore the effects of learning rates, number of training epochs, and optimizers. Table 10 shows the accuracy on different settings.

For learning rates, we observe that the optimal learning rate increases as the noise ratio increases as expected. We think this is because an appropriately large learning rate can help the model escape from local optimum and increases the model robustness to noise labels. Small learning rates tend to make the model trapped in a locally optimum or overfit the model. It also takes a longer time to train.

For training epochs, we set the same learning rate 1e-3. We find model can gradually fit all data as training epochs increase. But after 50 epochs, the model is over-fit. We have two interesting observations on fitting labels with random mode: a) we do not need to change the learning rate schedule; b) once the fitting starts, it converges quickly.

For optimizers, as the noise ratio increases, the results with RMSprop and Adam optimizers are significantly better than the simple SGD optimizer. The RMSprop and Adam optimizers have the following advantages: first, the gradient of the current batch is used for fine-tuning the final update. Second, the learning rate is adaptive for each parameter. These advantages help to be able to get rid of the local optima.

Case	Source Label	Checked Label	Sample source
What company is being bought by Yahoo and how much is the deal worth ?	HUM	HUM and NUM	TREC
What is the best college in the country ?	HUM	LOC	TREC
Mars water tops science honours.The discovery that salty, acidic water once flowed across the surface of Mars has topped a list of the 10 key scientific advances of 2004.	World	Sci/Tech	Ag-News
Pharma Groups Work on EPC Issues.Sept. 30, 2004 Reacting to calls from pharmaceutical retailers, distributors and manufacturers, EPCglobal has added a new action group to specifically study the pharmaceutical industry.	Sci/Tech	Sci/Tech and Business	Ag-News
比较实惠，旁边有易初莲花，买东西比较方便，还有麦当劳。 (This hotel is not only affordable, but also close to Etsu Lotus, which is convenient for shopping.There's also a McDonald's which is convenient for dining.)	negative	positive	Chnsenticorp
购物较方便，上外滩也近，但房间太小。没有早餐不方便，较为嘈杂，装修较老。 (This hotel is convenient for shopping and close to the Bund. But the disadvantages are small room, no breakfast, noisy environment and old decoration.)	negative	positive and negative	Chnsenticorp
作为酒店的老客户，恐怕以后要做另外的选择了——服务水平在下降，价格却一升再升，再这样下去，下次不会再入住了。(As a regular customer of the hotel, I'm afraid I'll have to make another choice in the future - the service level is declining, but the price is rising. if this continues, I won't stay there again next time.)	positive	negative	Chnsenticorp

Table 6: Examples of **intrinsic noise** in different datasets

Conditions	Rate generation formula
$n\ mod\ 2 = 1$ and $\lceil n/2 \rceil \leq i \leq n$	$rate = C_n^i p^i (1-p)^{n-i} + C_n^{n-i} p^{n-i} (1-p)^i$
$n\ mod\ 2 = 0$ and $i = \lceil n/2 \rceil$	$rate = C_n^i p^i (1-p)^{n-i}$
$n\ mod\ 2 = 0$ and $\lceil n/2 \rceil < i \leq n$	$rate = C_n^i p^i (1-p)^{n-i} + C_n^{n-i} p^{n-i} (1-p)^i$

Table 7: The conditions and the rate generation formulas of annotators' agreement. Here, n is the number of annotators, i denotes the number of annotators who get the same annotation, p denotes the accuracy of annotators getting the labels correct.

Sample type\The rate of same annotation (n=10)	100%	90%	80%	70%	60%	50%
Emotionally inclined	43.82%	37.69%	14.59%	3.35%	0.51%	0.05%
Not Emotionally inclined	0.20%	1.95%	8.79%	23.44%	41.02%	24.61%

Table 8: The rate of annotators agreement on emotionally inclined or not inclined samples. Note that this table only lists annotator agreement rate from 100% to 50% as the dataset is binary. The rates are complementary such as r90% = r10%. All percentages are rounded up.

Accuracy\ The rate of same annotation (n=10)	100%	90%	80%	70%	60%	50%
Emotionally inclined	99.56%	95.07%	62.40%	12.49%	1.22%	0.21%
Not Emotionally inclined	0.44%	4.93%	37.60%	87.51%	98.78%	99.79%

Table 9: The accuracy of getting emotionally inclined or not inclined samples correct.

Hyperparameters		Noise Ratio %				
		10	20	30	40	50
learing rates	1e-2	83.79%	82.10%	75.92%	70.97%	68.60%
	1e-3	83.80%	81.80%	75.80%	70.40%	67.60%
	1e-4	82.56%	81.44%	75.59%	70.11%	66.36%
	1e-5	82.43%	81.32%	75.24%	69.89%	66.12%
	1e-7	82.13%	80.76%	74.68%	69.43%	65.68%
training epochs	10	83.24%	79.74%	73.96%	69.36%	65.07%
	20	83.80%	81.80%	75.80%	70.40%	67.60%
	30	86.55%	81.54%	76.05%	70.94%	67.80%
	50	86.55%	81.66%	76.23%	70.93%	67.81%
	100	86.56%	81.67%	76.23%	71.04%	67.81%
optimizers	SGD	82.43%	78.37%	72.68%	67.60%	65.03%
	Adagrad	82.57%	78.45%	72.97%	67.83%	65.48%
	Momentum	83.64%	79.86%	73.52%	68.25%	66.23%
	RMSprop	85.24%	81.22%	74.26%	69.22%	67.01%
	Adam	83.80%	81.80%	75.80%	70.40%	67.60%

Table 10: Accuracy of Co-teaching model trained fixed TREC dataset with different noise ratios of random noise and hyperparameter settings

Mitigating the Diminishing Effect of Elastic Weight Consolidation

Canasai Kruengkrai **Junichi Yamagishi**
National Institute of Informatics, Japan
{canasai,jyamagishi}@nii.ac.jp

Abstract

Elastic weight consolidation (EWC, Kirkpatrick et al. 2017) is a promising approach to addressing catastrophic forgetting in sequential training. We find that the effect of EWC can diminish when fine-tuning large-scale pre-trained language models on different datasets. We present two simple objective functions to mitigate this problem by rescaling the components of EWC. Experiments on natural language inference and fact-checking tasks indicate that our methods require much smaller values for the trade-off parameters to achieve results comparable to EWC.[1]

1 Introduction

New training data may arrive after we have spent considerable time training our model on the data at hand. A simple method for exploiting both new and old training data is to mix them and retrain the model from scratch. However, this mix-and-retrain method is neither always practical nor economical, especially in academic environments where computational resources are limited.

Sequential training is a potential alternative approach but faces a difficult challenge called *catastrophic forgetting* in which the performance on old data drastically drops when we train a model on new data. There exists a line of work that has addressed this challenge (Rusu et al., 2016; Li and Hoiem, 2018; Kirkpatrick et al., 2017; Mallya et al., 2018; He and Jaeger, 2018; Zhang et al., 2020). In this paper, we are particularly interested in elastic weight consolidation (EWC, Kirkpatrick et al. 2017), which has been shown to be helpful for domain adaptation (Saunders et al., 2019; Thompson et al., 2019).

EWC adds a regularization term to the objective function to ensure that the model works well on both new and old data. We empirically find that EWC requires unexpectedly large values for the trade-off parameter (λ) between the regularizer and the loss to be effective when applying to pre-trained language models. Figure 1 shows such a phenomenon in which EWC has no effect in preventing catastrophic forgetting of the prior dataset (MNLI) with λ in the range of $[10^0, 10^4]$. We have to scale λ up to $[10^5, 10^7]$, which is an unusual range of hyperparameters. To the best of our knowledge, this phenomenon has not been reported in the literature.

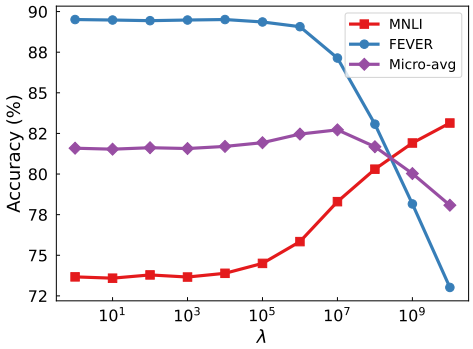

Figure 1: Accuracy vs. trade-off parameter λ. We sequentially fine-tune BERT (Bidirectional Encoder Representations from Transformers, Devlin et al. 2019) on MNLI (Williams et al., 2018) and FEVER (Thorne et al., 2018) and evaluate performance on the balanced dev sets. EWC starts to increase the accuracy of the prior dataset (MNLI) when increasing λ to 10^5 and yields the highest average accuracy at 10^7.

We propose two simple objective functions for mitigating the diminishing effect of EWC. Our objective functions rely on rescaling the components of EWC. Specifically, the first objective function involves taking the square root of the regularization term, while the second one involves using the absolute value of the gradient instead of the squared gradient. Both of our objective functions can reduce the values of the trade-off parameter λ by three to seven orders of magnitude while producing results similar to those of the original EWC.

[1]Our code is available at https://github.com/nii-yamagishilab/ewc.

Proceedings of the 29th International Conference on Computational Linguistics, pages 4568–4574
October 12–17, 2022.

2 Background

2.1 Problem formulation

We consider a supervised learning problem in which the task is to map an input $x \in \mathcal{X}$ to a label $y \in \mathcal{Y}$. We need to train a model $h_{\boldsymbol{\theta}} : \mathcal{X} \rightarrow \mathcal{Y}$ with parameters $\boldsymbol{\theta} \in \mathbb{R}^d$. Given a dataset $\mathcal{D} = \{(x_i, y_i)\}_{i=1}^{M}$, we typically estimate $\boldsymbol{\theta}$ on the basis of empirical risk minimization (ERM, Vapnik 1992):

$$J_{\text{ERM}}(\boldsymbol{\theta}) = \frac{1}{M} \sum_{(x,y)\in\mathcal{D}} \mathcal{L}(h_{\boldsymbol{\theta}}(x), y), \quad (1)$$

where $\mathcal{L}$ is the negative log likelihood loss:

$$\mathcal{L}(h_{\boldsymbol{\theta}}(x), y) = -\sum_{y\in\mathcal{Y}} \mathbb{1}\{\hat{y} = y\} \log p_{\boldsymbol{\theta}}(\hat{y}|x).$$

Our base model $h_{\boldsymbol{\theta}}$ is a neural network containing a multilayer perceptron (MLP) on top of a pre-trained language model (e.g., BERT). Thus, we define $p_{\boldsymbol{\theta}}(\hat{y}|x) = \text{softmax}(h_{\boldsymbol{\theta}}(x))$, where $h_{\boldsymbol{\theta}} = \text{MLP}(\text{BERT}(x))$. The model parameters $\boldsymbol{\theta}$ include those in the MLP and BERT.

2.2 Elastic weight consolidation

Elastic weight consolidation (EWC, Kirkpatrick et al. 2017) is based on a Bayesian framework that seeks to approximate the posterior distribution of $\boldsymbol{\theta}$ conditional on two datasets. Let $\mathcal{D}$ and $\mathcal{D}^0$ denote the current and prior datasets, respectively. We express the posterior distribution as:

$$\begin{aligned} p(\boldsymbol{\theta}|\mathcal{D}, \mathcal{D}^0) &= \frac{p(\boldsymbol{\theta}, \mathcal{D}, \mathcal{D}^0)}{p(\mathcal{D}, \mathcal{D}^0)}, \\ &= \frac{p(\mathcal{D}|\boldsymbol{\theta}, \mathcal{D}^0)p(\boldsymbol{\theta}, \mathcal{D}^0)}{p(\mathcal{D}, \mathcal{D}^0)}, \\ &= \frac{p(\mathcal{D}|\boldsymbol{\theta})p(\boldsymbol{\theta}|\mathcal{D}^0)p(\mathcal{D}^0)}{p(\mathcal{D})p(\mathcal{D}^0)} \\ &\propto p(\mathcal{D}|\boldsymbol{\theta})p(\boldsymbol{\theta}|\mathcal{D}^0), \end{aligned} \quad (2)$$

where we assume that $\mathcal{D}$ and $\mathcal{D}^0$ are conditionally independent in the third line and ignore the constant in the last line. Taking the log on both sizes of Eq. (2), we have:

$$\log p(\boldsymbol{\theta}|\mathcal{D}, \mathcal{D}^0) = \log p(\mathcal{D}|\boldsymbol{\theta}) + \log p(\boldsymbol{\theta}|\mathcal{D}^0). \quad (3)$$

The first term on the right-hand side corresponds to the log likelihood of $\mathcal{D}$, which can be computed using Eq. (1). The second term is intractable but can be approximated using a second-order Taylor expansion of the KL-divergence around the parameters of the previously trained model, $\boldsymbol{\theta}^0$:

$$\log p(\boldsymbol{\theta}|\mathcal{D}^0) \approx \frac{1}{2}\Delta\boldsymbol{\theta}^\top \mathbf{H} \Delta\boldsymbol{\theta}, \quad (4)$$

where $\Delta\boldsymbol{\theta} = \boldsymbol{\theta} - \boldsymbol{\theta}^0$ and $\mathbf{H}$ is the expected negative Hessian of the posterior distribution (Pascanu and Bengio, 2014). Computing $\mathbf{H}$ is impractical. Kirkpatrick et al. (2017) proposed approximating $\mathbf{H}$ using the diagonal of the Fisher information matrix. Let $\text{diag}(\mathbf{f})$ be the diagonal matrix with diagonal $\mathbf{f}$. We estimate $\mathbf{f}$ with the average of the squared gradient across some N subsamples $\mathcal{S}^0$:

$$\mathbf{f} = \frac{1}{N} \sum_{(x,y)\in\mathcal{S}^0} \left(\nabla_{\boldsymbol{\theta}^0} \mathcal{L}(h_{\boldsymbol{\theta}}(x), y)\right)^2. \quad (5)$$

Replacing $\mathbf{H}$ with $\text{diag}(\mathbf{f})$, we can simplify Eq. (4) as:

$$\log p(\boldsymbol{\theta}|\mathcal{D}^0) \approx \frac{1}{2} \sum_{j=1}^{d} f_j (\theta_j - \theta_j^0)^2. \quad (6)$$

Applying Eqs. (1) and (6) to Eq. (3), we obtain the EWC objective:

$$J_{\text{EWC}}(\boldsymbol{\theta}) = J_{\text{ERM}}(\boldsymbol{\theta}) + \frac{\lambda}{2} \sum_{j=1}^{d} f_j (\theta_j - \theta_j^0)^2, \quad (7)$$

where λ is the trade-off parameter.

3 Proposed method

As shown in Figure 1, EWC requires extremely large values of λ to be effective. We analyze the components of EWC and find that this problem arises from the Fisher approximation in Eq. (5). The diagonal element f_j corresponds to the jth element of the squared gradient with respect to $\boldsymbol{\theta}^0$. Since its training had already converged, the values of the gradient are typically small. When we square such a small decimal and combine it with the squared difference between the current and prior parameters, the final value can be vanishingly small.[2] We find that this issue is neither affected by datasets nor pre-trained language models. In Appendix A, we further investigate this issue on another pre-trained language model.

We propose scaling up the Fisher approximation by taking the square root to resolve the issue above. We define the square root of EWC (REWC) as:

$$J_{\text{REWC}}(\boldsymbol{\theta}) = J_{\text{ERM}}(\boldsymbol{\theta}) + \lambda\sqrt{A + \epsilon}, \quad (8)$$

[2]For example, in the MNLI⇒FEVER experiment, we find that 85.9% of non-zero f_j are less than 1e-10.

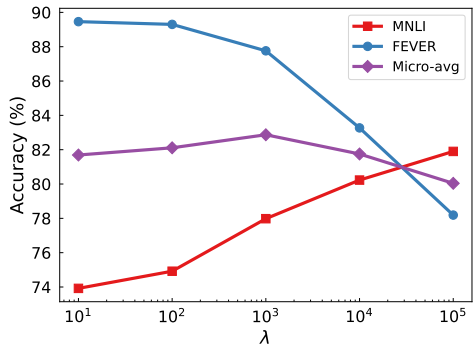

(a) REWC

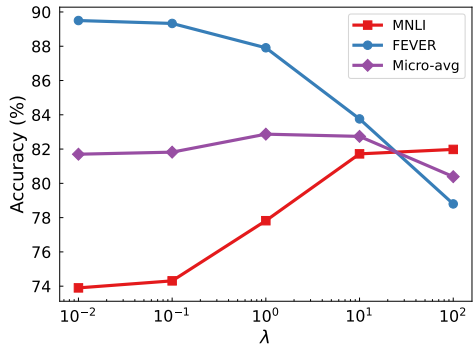

(b) AEWC

Figure 2: Accuracy vs. trade-off parameter λ of our REWC and AEWC. Both methods begin to affect accuracy with much lower λ (i.e., 10^2 and 10^{-1} for REWC and AEWC, respectively) while maintaining average accuracies similar to EWC.

where $A = \sum_{j=1}^{d} f_j(\theta_j - \theta_j^0)^2$, and ϵ is a small value (e.g., 10^{-8}) for preventing the derivative of the square root at 0.

Another solution is to use the absolute value of the gradient instead of the squared gradient. We define:

$$\mathbf{g} = \frac{1}{N} \sum_{(x,y)\in\mathcal{S}^0} \left|\nabla_{\boldsymbol{\theta}^0} \mathcal{L}(h_{\boldsymbol{\theta}}(x), y)\right|. \quad (9)$$

Note that diag($\mathbf{g}$) is positive semi-definite (like diag($\mathbf{f}$)) because all of its eigenvalues are greater than or equal to 0. Replacing the squared difference with the absolute difference yields our absolute EWC (AEWC):

$$J_{\text{AEWC}}(\boldsymbol{\theta}) = J_{\text{ERM}}(\boldsymbol{\theta}) + \lambda \sum_{j=1}^{d} g_j |\theta_j - \theta_j^0|. \quad (10)$$

Figure 2 shows the results of REWC and AEWC based on the same setting as in Figure 1.

4 Experiments

4.1 Datasets

We evaluated the objective functions described in §2 and §3 on natural language inference and fact-checking tasks. We used six datasets pre-processed by Schuster et al. (2021) as follows:

MNLI (Williams et al., 2018) is a multi-genre natural language inference dataset. The task is to determine the inference relation between two sentences. Schuster et al. (2021) converted the original labels {"entailment", "contradiction","neutral"} into {"supported", "refuted", "not enough info"}.

FEVER (Thorne et al., 2018) (Fact Extraction and VERification) verifies whether a claim is supported or refuted by an evidence sentence, or decides whether there is insufficient information to make a decision.

VITC (Schuster et al., 2021) introduces the notion of contrastive evidence to FEVER. Given a claim, two evidence sentences that are nearly identical but with different labels are created. Thus, the task becomes more challenging than that of FEVER. The dataset contains both real and synthetic examples. We used only the real ones in our experiments.

ADVERSARIAL (Thorne et al., 2019) is derived from the FEVER 2.0 shared task, containing adversarially created claims that aim to induce erroneous predictions to the FEVER-trained models.

SYMMETRIC (Schuster et al., 2019) is another dataset that challenges the FEVER-trained models. It contains synthetically created claim-evidence pairs designed to break models that often make predictions using claims only without taking evidence sentences into account.

TRIGGERS (Atanasova et al., 2020) contains adversarial claims generated by using GPT-2 (Radford et al., 2019) given the original claims and triggers, which are words that cause the model to flip its prediction.

We selected λ that yields the highest average accuracy on the development (dev) sets. To avoid a bias towards more populated datasets (e.g., VITC), we created our balanced dev sets by randomly selecting 9,000 examples from each of the original dev sets. Since the dev and test sets of MNLI are identical, we split 9,000 examples from the training set to form the dev set and used the test set for the final evaluation. Table 1 shows our dataset statistics.

Dataset	|Train|	|Dev|	|Test|
MNLI	383,702	9,000	9,832
FEVER	178,059	9,000	11,710
VITC	248,953	9,000	34,481
ADVERSARIAL	–	–	766
SYMMETRIC	–	–	712
TRIGGERS	–	–	186

Table 1: Dataset statistics in our experiments. Bottom three datasets contain only test sets adversarially created for testing robustness of fact-checking models.

4.2 Training details

We implemented our base model described in §2.1 using Hugging Face's Transformers library (Wolf et al., 2020). Specifically, the model consists of a two-layer MLP and BERT-base. Let x be the input sequence (i.e., a pair of sentences in our datasets). BERT-base encodes x into a sequence of hidden state vectors. Following common practice, we used the first hidden state vector of the special **cl**assification token (i.e., [CLS]) to represent x and fed it to the MLP followed by a softmax function.

For all experiments, we used Adafactor optimizer (Shazeer and Stern, 2018) with a gradient clipping of 1.0. Our effective batch size is 256.[3] For standard training, we randomly initialized the model parameters with $\mathcal{N}(0, 0.02)$[4], except for those of BERT-base. We trained each model for three epochs with a learning rate of 2e-5.

For sequential training, we randomly selected 1% of examples from $\mathcal{D}^0$ to represent $\mathcal{S}^0$ in Eq. (5). We also varied the subsample size from 0.1% to 10% but did not observe significant changes in performance. We initialized the current model parameters using the prior ones (i.e., $\boldsymbol{\theta}^0 \rightarrow \boldsymbol{\theta}$). Determining a learning rate can be challenging. We used a method analogous to the learning rate decay technique (Ng, 2017). Let α_0 be the initial learning rate and r be the number of prior training runs. We computed the learning rate α for the current training run as:

$$\alpha = \frac{1}{1 + (\text{decay\_rate} \times r)} \alpha_0. \qquad (11)$$

For example, consider the case of further training the MNLI-trained model on the FEVER dataset, where $\alpha_0 =$ 2e-5 and $r = 1$. We set decay_rate to 1e-2 for all sequential training experiments. Using Eq. (11), the learning rate α for the current run decreases to 1.98e-5. We conducted all the experiments on NVIDIA Tesla A100 GPUs.

[3] We used gradient accumulation with 8 batches of 32.

[4] This is the default setting in Transformers.

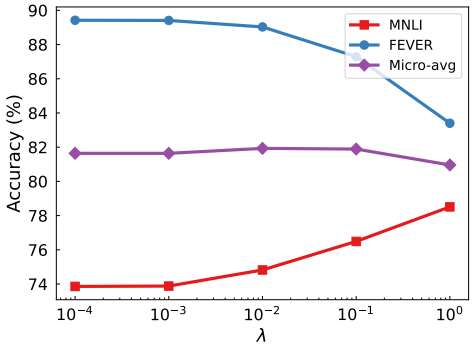

(a) EWC w/o f_j

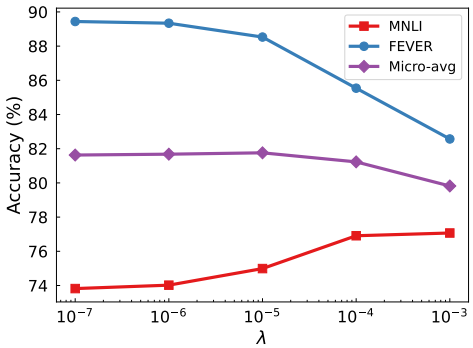

(b) AEWC w/o g_j

Figure 3: Accuracy vs. trade-off parameter λ of EWC and AEWC without f_j and g_j, respectively.

4.3 Results

Table 2 shows the results of various settings on the test sets. For sequential training, conducting experiments on all combinations takes time and considerable resources. Thus, we chose only a representative order for the datasets in accordance with their publication times. Since the MNLI and FEVER datasets were published at the same time, we decided to start with MNLI due to its generality.

We considered the mix-and-retrain method ($\cup$) with ERM as the *topline* setting. Unsurprisingly, this method yields the best performance on the prior datasets. The sequential training method ($\Rightarrow$) with ERM (i.e., vanilla fine-tuning) encounters severe catastrophic forgetting on the prior datasets. Our REWC and AEWC effectively reduce the values of λ. AEWC requires the lowest λ among the three objective functions. The performances of all the methods seem comparable on average, but each yields a different trade-off in accuracy between the prior and current datasets. Regarding the training time, AEWC is faster than REWC/EWC (though not significant) because its computation is simpler.

Training set	Obj.	λ	MNLI	FEVER	VitC	Adver.	Sym.	Trig.
MNLI	ERM	–	83.9±0.1	67.7±0.7	47.8±0.7	51.0±0.8	74.8±0.3	68.3±1.4
FEVER	ERM	–	58.8±0.2	87.4±0.1	59.7±0.1	51.4±0.7	75.3±0.2	65.4±1.4
VitC	ERM	–	62.5±1.0	65.1±0.5	78.2±1.2	28.9±0.5	65.8±1.2	69.1±2.8
MNLI ∪ FEVER	ERM	–	83.9±0.2	87.8±0.1	61.0±0.3	53.8±0.2	82.6±0.4	73.8±0.4
MNLI ⇒ FEVER	ERM	–	74.9±0.2	88.2±0.2	62.7±0.1	55.0±0.3	82.6±0.2	71.2±0.4
	EWC	10^7	79.3±0.2	86.3±0.1	61.0±0.4	53.7±0.4	80.3±0.6	67.7±1.4
	REWC	10^3	78.7±0.2	86.8±0.1	61.5±0.3	53.6±0.5	81.1±0.6	69.2±0.6
	AEWC	10^0	78.7±0.2	87.2±0.1	61.9±0.3	53.9±0.4	81.3±0.4	70.5±0.4
FEVER ∪ VitC	ERM	–	69.0±0.4	87.5±0.1	83.3±0.3	51.0±0.2	79.0±0.7	71.5±0.8
FEVER ⇒ VitC	ERM	–	66.2±0.4	75.8±0.4	84.4±0.1	39.6±0.8	71.3±0.8	70.9±0.9
	EWC	10^6	65.8±0.3	78.2±0.2	83.6±0.1	40.5±1.4	71.3±0.5	70.0±0.6
	REWC	10^2	66.2±0.3	76.7±0.2	84.2±0.1	39.7±1.5	71.4±0.3	70.5±0.6
	AEWC	10^{-1}	66.3±0.4	76.3±0.2	84.3±0.1	39.5±1.4	71.4±0.4	70.6±0.6
MNLI ∪ VitC	ERM	–	84.0±0.1	76.8±0.2	84.3±0.1	43.8±0.6	75.5±0.6	74.6±1.9
MNLI ⇒ VitC	ERM	–	76.0±0.2	72.4±0.2	85.5±0.2	40.2±0.8	73.0±0.3	71.7±1.0
	EWC	10^5	76.5±0.3	72.7±0.4	85.3±0.1	41.0±1.0	73.3±0.5	72.4±1.8
	REWC	10^2	76.7±0.3	72.9±0.3	85.1±0.1	41.1±0.9	73.5±0.3	72.8±1.9
	AEWC	10^{-1}	76.4±0.2	72.7±0.4	85.3±0.1	40.7±1.1	73.3±0.4	72.8±1.9
MNLI ∪ FEVER ∪ VitC	ERM	–	83.8±0.2	88.1±0.1	84.6±0.1	53.5±0.6	82.6±0.4	73.2±1.0
MNLI ⇒ FEVER ⇒ VitC	ERM	–	75.1±0.3	79.1±0.3	85.7±0.0	44.4±0.5	75.4±0.7	74.9±0.7
	EWC	10^6	77.5±0.3	79.1±0.2	84.0±0.2	44.2±0.4	75.1±0.5	73.3±1.6
	REWC	10^2	76.4±0.4	77.7±0.2	85.2±0.1	42.9±0.4	74.4±0.5	73.5±0.9
	AEWC	10^0	78.6±0.2	82.8±0.1	80.0±1.0	46.8±0.6	76.9±0.5	74.1±1.5

Table 2: Symbol ∪ denotes mixing training sets, while arrow ⇒ denotes using training sets sequentially. Gray color highlights the effect of *catastrophic forgetting* on the prior dataset. Blue color emphasizes the performance on the current dataset. Green color indicates the topline performance of the mix-and-retrain method. We ran each experiment five times using different random seeds and report mean and standard deviation.

Obj.	λ	MNLI	FEVER	VitC
EWC	10^7	79.3±0.2	86.3±0.1	61.0±0.4
w/o f_j	10^{-2}	75.9±0.1	88.0±0.1	62.4±0.2
AEWC	10^0	78.7±0.2	87.2±0.1	61.9±0.3
w/o g_j	10^{-5}	76.1±0.2	87.6±0.1	62.1±0.2

Table 3: Ablation studies on EWC and AEWC for MNLI ⇒ FEVER. "w/o f_j (or g_j)" denotes omitting the gradient component from the regularization term.

4.4 Discussion

We can interpret the EWC family as a weighted sum of the squared (or absolute) differences between the current and prior parameters. The gradient component helps suggest which parameter is important. To examine the benefit of gradient information, we conducted ablation studies on EWC and AEWC in the MNLI ⇒ FEVER experiment. We omitted f_j and g_j from Eqs. (7) and (10), respectively. The remaining regularization terms resemble the squared ℓ_2-norm and the ℓ_1-norm that take the prior parameters into account.

As seen in Figure 3, without the gradient component, both methods need lower λ to affect the accuracy of the prior dataset (MNLI). However, improvements on the prior dataset are marginal (∼1%) before reaching the optimal average accuracy compared to the original EWC and AEWC (∼4%). Table 3 shows the ablation results on the test sets, indicating that omitting the gradient component yields lower accuracies on the prior dataset. These results confirm that the gradient component is indeed helpful.

5 Conclusion

Without realizing the diminishing effect of EWC, we may fine-tune a pre-trained language model with a conventional range of hyperparameters and find no effect in combating catastrophic forgetting. We identified a possible cause of this issue and suggested two alternative objective functions, REWC and AEWC, that yield results comparable to the original EWC. Exploring more efficient ways for choosing an optimal λ is part of our future work.

Acknowledgments

This work is supported by JST CREST Grants (JPMJCR18A6 and JPMJCR20D3) and MEXT KAKENHI Grants (21H04906), Japan.

References

Pepa Atanasova, Dustin Wright, and Isabelle Augenstein. 2020. Generating label cohesive and well-formed adversarial claims. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 3168–3177, Online. Association for Computational Linguistics.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, Minneapolis, Minnesota. Association for Computational Linguistics.

Xu He and Herbert Jaeger. 2018. Overcoming catastrophic interference using conceptor-aided backpropagation. In *6th International Conference on Learning Representations (ICLR)*.

James Kirkpatrick, Razvan Pascanu, Neil C. Rabinowitz, Joel Veness, Guillaume Desjardins, Andrei A. Rusu, Kieran Milan, John Quan, Tiago Ramalho, Agnieszka Grabska-Barwinska, Demis Hassabis, Claudia Clopath, Dharshan Kumaran, and Raia Hadsell. 2017. Overcoming catastrophic forgetting in neural networks. *Proceeding of the National Academy of Science*, 114(13):3521–3526.

Zhenzhong Lan, Mingda Chen, Sebastian Goodman, Kevin Gimpel, Piyush Sharma, and Radu Soricut. 2020. ALBERT: A lite bert for self-supervised learning of language representations. In *International Conference on Learning Representations*.

Zhizhong Li and Derek Hoiem. 2018. Learning without forgetting. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 40(12):2935–2947.

Arun Mallya, Dillon Davis, and Svetlana Lazebnik. 2018. Piggyback: Adapting a single network to multiple tasks by learning to mask weights. In *Proceedings of the European Conference on Computer Vision (ECCV)*.

Andrew Ng. 2017. Learning rate decay. https://www.coursera.org/lecture/deep-neural-network/learning-rate-decay-hjgIA.

Razvan Pascanu and Yoshua Bengio. 2014. Revisiting natural gradient for deep networks. In *2nd International Conference on Learning Representations (ICLR)*.

Alec Radford, Jeff Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. 2019. Language models are unsupervised multitask learners.

Andrei A. Rusu, Neil C. Rabinowitz, Guillaume Desjardins, Hubert Soyer, James Kirkpatrick, Koray Kavukcuoglu, Razvan Pascanu, and Raia Hadsell. 2016. Progressive neural networks. *CoRR*, abs/1606.04671.

Danielle Saunders, Felix Stahlberg, Adrià de Gispert, and Bill Byrne. 2019. Domain adaptive inference for neural machine translation. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 222–228, Florence, Italy. Association for Computational Linguistics.

Tal Schuster, Adam Fisch, and Regina Barzilay. 2021. Get your vitamin C! robust fact verification with contrastive evidence. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 624–643, Online. Association for Computational Linguistics.

Tal Schuster, Darsh Shah, Yun Jie Serene Yeo, Daniel Roberto Filizzola Ortiz, Enrico Santus, and Regina Barzilay. 2019. Towards debiasing fact verification models. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 3419–3425, Hong Kong, China. Association for Computational Linguistics.

Noam Shazeer and Mitchell Stern. 2018. Adafactor: Adaptive learning rates with sublinear memory cost. In *Proceedings of the 35th International Conference on Machine Learning*, volume 80 of *Proceedings of Machine Learning Research*, pages 4596–4604. PMLR.

Brian Thompson, Jeremy Gwinnup, Huda Khayrallah, Kevin Duh, and Philipp Koehn. 2019. Overcoming catastrophic forgetting during domain adaptation of neural machine translation. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 2062–2068, Minneapolis, Minnesota. Association for Computational Linguistics.

James Thorne, Andreas Vlachos, Christos Christodoulopoulos, and Arpit Mittal. 2018. FEVER: a large-scale dataset for fact extraction and VERification. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 809–819, New Orleans, Louisiana. Association for Computational Linguistics.

James Thorne, Andreas Vlachos, Oana Cocarascu, Christos Christodoulopoulos, and Arpit Mittal. 2019. The FEVER2.0 shared task. In *Proceedings of the Second Workshop on Fact Extraction and VERification (FEVER)*, pages 1–6, Hong Kong, China. Association for Computational Linguistics.

Vladimir Vapnik. 1992. Principles of risk minimization for learning theory. In *Advances in Neural Information Processing Systems*, volume 4. Morgan-Kaufmann.

Adina Williams, Nikita Nangia, and Samuel Bowman. 2018. A broad-coverage challenge corpus for sentence understanding through inference. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 1112–1122, New Orleans, Louisiana. Association for Computational Linguistics.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Remi Louf, Morgan Funtowicz, Joe Davison, Sam Shleifer, Patrick von Platen, Clara Ma, Yacine Jernite, Julien Plu, Canwen Xu, Teven Le Scao, Sylvain Gugger, Mariama Drame, Quentin Lhoest, and Alexander Rush. 2020. Transformers: State-of-the-art natural language processing. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 38–45, Online. Association for Computational Linguistics.

Jeffrey O. Zhang, Alexander Sax, Amir Zamir, Leonidas Guibas, and Jitendra Malik. 2020. Side-tuning: A baseline for network adaptation via additive side networks. In *Computer Vision – ECCV 2020*, pages 698–714, Cham. Springer International Publishing.

A Additional results

We verified the diminishing effect of EWC on another pre-trained language model, A Lite BERT (ALBERT, Lan et al. 2020). Figure 4 shows the results of sequential training: MNLI ⇒ FEVER. We can still see the diminishing effect of the original EWC, while our REWC and AEWC reduce the value of λ by three and six orders of magnitude and produced similar results.

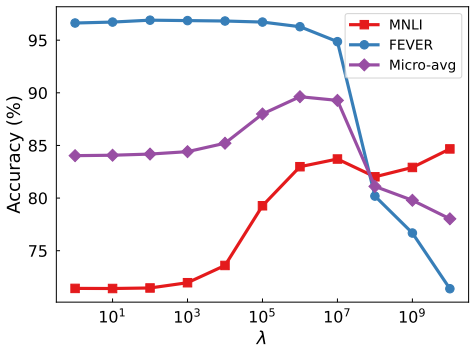

(a) EWC

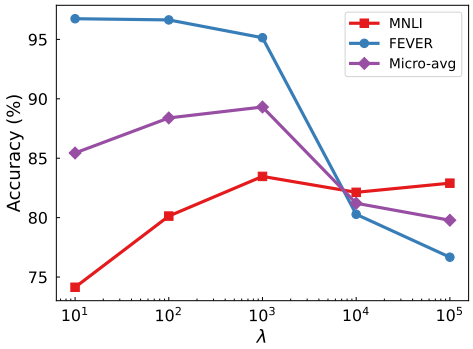

(b) REWC

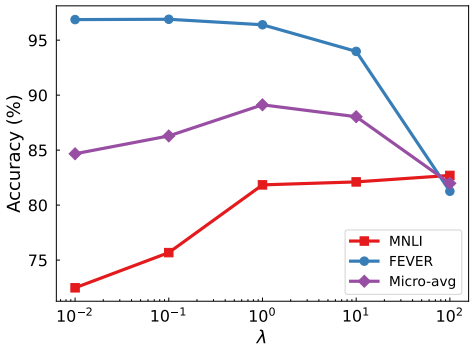

(c) AEWC

Figure 4: Accuracy vs. trade-off parameter λ on the balanced dev sets of MNLI and FEVER using ALBERT-base (Lan et al., 2020). EWC, REWC, and AEWC achieve highest average accuracies with $\lambda = 10^6$, 10^3, and 10^0, respectively.

Token and Head Adaptive Transformers for Efficient Natural Language Processing

Chonghan Lee[1], Md Fahim Faysal Khan[1], Rita Brugarolas Brufau[2], Ke Ding[2], Vijaykrishnan Narayanan[1]
[1]The Pennsylvania State University
[2]Intel
[1]{cvl5361,mzk591,vxn9}@psu.edu
[2]{rita.brugarolas.brufau,ke.ding}@intel.com

Abstract

While pre-trained language models like BERT (Devlin et al., 2019) have achieved impressive results on various natural language processing tasks, deploying them on resource-restricted devices is challenging due to their intensive computational cost and memory footprint. Previous approaches mainly focused on training smaller versions of a BERT model with competitive accuracy under limited computational resources. In this paper, we extend *Length Adaptive Transformer* (Kim and Cho, 2021) and propose to design *Token and Head Adaptive Transformer*, which can compress and accelerate various BERT-based models via simple fine-tuning. We train a transformer with a progressive token and head pruning scheme, eliminating a large number of redundant tokens and attention heads in the later layers. Then, we conduct a multi-objective evolutionary search with the overall number of floating point operations (FLOPs) as its efficiency constraint to find joint token and head pruning strategies that maximize accuracy and efficiency under various computational budgets. Empirical studies show that a large portion of tokens and attention heads could be pruned while achieving superior performance compared to the baseline BERT-based models and Length Adaptive Transformers in various downstream NLP tasks. MobileBERT(Sun et al., 2020) trained with our joint token and head pruning scheme achieves a GLUE score of 83.0, which is 1.4 higher than Length Adaptive Transformer and 2.9 higher than the original model.

1 Introduction

The Transformer (Vaswani et al., 2017) has become the dominant neural architecture used in natural language processing. Especially, pre-trained models using the Transformer architecture as their backbone, e.g., BERT (Devlin et al., 2019), RoBERTa (Liu et al., 2019), and GPT-3 (Brown et al., 2020), have shown significant accuracy improvement across various natural language processing tasks. However, the amount of computation required for these large neural networks is enormous, which leads to a substantial increase in energy consumption and carbon footprint. For example, an article points out that in order to train a GPT-3 networks using standard NVIDIA gpus, it would roughly take 190,000 KWh of energy which is equivalent to producing around 85,000 Kg CO_2 considering America's average carbon emission intensity (Katyanna Quach, 2020). Despite their remarkable performance, these power hungry models pose a great hindrance along the pathway to sustainable systems. Hence, it is crucial to make the large-scale networks efficient and sustainable to be deployed on various hardware with different computational budgets and adopted for real-world applications.

There have been efforts to improve the efficiency of the large language models (see section 6 for a discussion in detail.) One of the effective approaches has focused on removing redundant model parameters. Recent works (Voita et al., 2019; Michel et al., 2019) show that only a small subset of heads are important and mainly attributed to the final decision making and some heads could even be pruned at test time to improve efficiency. Other works successfully improve the efficiency of BERT-based models by pruning out word token vectors. Length Adaptive Transformer (Kim and Cho, 2021) based on PoWER-BERT (Goyal et al., 2020) demonstrates that training an adaptive transformer with progressive word token pruning improves accuracy-efficiency trade-off that can satisfy any target computational budget. In this paper, we focus on developing a framework capable of training an adaptive Transformer that jointly eliminates redundant attention heads and word tokens to maximize accuracy and minimize required computational resources. We perform a multi-objective evolutionary search to find a full Pareto frontier of joint token

Proceedings of the 29th International Conference on Computational Linguistics, pages 4575–4584
October 12–17, 2022.

and attention-head pruning schemes that provides optimal accuracy-efficiency trade-offs given any computational budget.

We applied our framework to BERT-based models with different sizes. Our progressive head pruning scheme combined with word token pruning allows heavy head pruning in the last layers of BERT, MobileBERT, and DistilBERT(Sanh et al., 2019) without any accuracy loss. Furthermore, the final models trained with our framework are robust to prune even more tokens along with attention heads. This results in superior accuracy and latency trade-off than simply pruning word tokens. Empirical evaluations show that MobileBERT trained with our joint token and head pruning scheme achieves a GLUE score of 83.0, which is 1.4 higher than Length Adaptive Transformer and 2.9 higher than the original model. Our adaptive BERT and DistilBERT also achieve higher accuracy with less computational cost compared to Length Adaptive Transformers. On SQuADv1.1 (Rajpurkar et al., 2016) question answering task, Token and Head Adaptive Transformers (THAT) obtain higher dev F1 scores with less computational resources compared to Length Adaptive Transformers and the original BERT-based models.

2 Background

In this section, we present the main building blocks of our framework. We review multi-head attention, the essential mechanism in the Transformer architecture. Our framework uses multi-head self-attention to decide which tokens and heads to prune in each layer. Then, we review PoWER-BERT and Length-Adaptive Transformer, which are recent works that eliminate tokens to improve efficiency in BERT-based models.

2.1 Multi-head Attention

The multi-head attention mechanism decomposes the scaled dot-product attention to extract independent features from the same input sequence in parallel. Let $\mathbf{x} = (x_1, x_2, \ldots, x_T)$ be a sequence of T word token vectors where $x_t \in \mathbb{R}^d$, and $q \in \mathbb{R}^d$ be a query vector. An attention mechanism is defined as

$$Attention(\mathbf{x}, q) = W_o \sum_{t=1}^{T} \alpha_t(q) W_v x_t \qquad (1)$$

where

$$\alpha_t(q) = softmax\left(\frac{q^\top W_q^\top W_k x_t}{\sqrt{d}}\right) \qquad (2)$$

$W_o, W_v, W_q, W_k \in \mathbb{R}^{d\times d}$ are trainable weight matrices and query q comes from the same sequence $\mathbf{x}$ in self-attention. Then multi-head attention is defined as

$$MHAttention(\mathbf{x}, q) = \sum_{h=1}^{H} Attention_h(\mathbf{x}, q) \qquad (3)$$

where H is a set of attention heads h and $Attention_h$ is a decomposed low-rank attention from the head h. All the representation outputs from $Attention_h$ are created from the same input and merged together to produce a single output. Studies from (Voita et al., 2019; Michel et al., 2019) have shown possible efficiency improvement from pruning some features extracted from the heads.

2.2 PoWER-BERT

PoWER-BERT prunes out redundant word token vectors $x_t \in \mathbf{x}$ at each layer of BERT model based on the attention significance score α_t (Goyal et al., 2020). While having the same number of parameters as BERT model, PoWER-BERT significantly reduces the computational cost. PoWER-BERT requires sequence length configuration search and re-training steps. In the sequence length configuration search step, auxiliary retention parameters and a regularizer parameter are briefly introduced in the model and the loss function, respectively, to approximate the number of retained token vectors across layers under the desired accuracy and efficiency trade-off. Then the model is re-trained based on the searched sequence length configuration $\mathbf{z} = (z_1, z_2, \ldots, z_N)$ where N is the total number of the encoder layers and z_n is the number of tokens to keep at layer n.

PoWER-BERT significantly improves the efficiency and achieves a better accuracy-efficiency trade-off than DistilBERT and other models with different compression techniques. However, the sequence length configuration search and re-training steps need to be repeated for each computational budget. Furthermore, PoWER-BERT's token pruning is limited to sequence-level classification tasks since it progressively prunes out hidden token vectors on each layer.

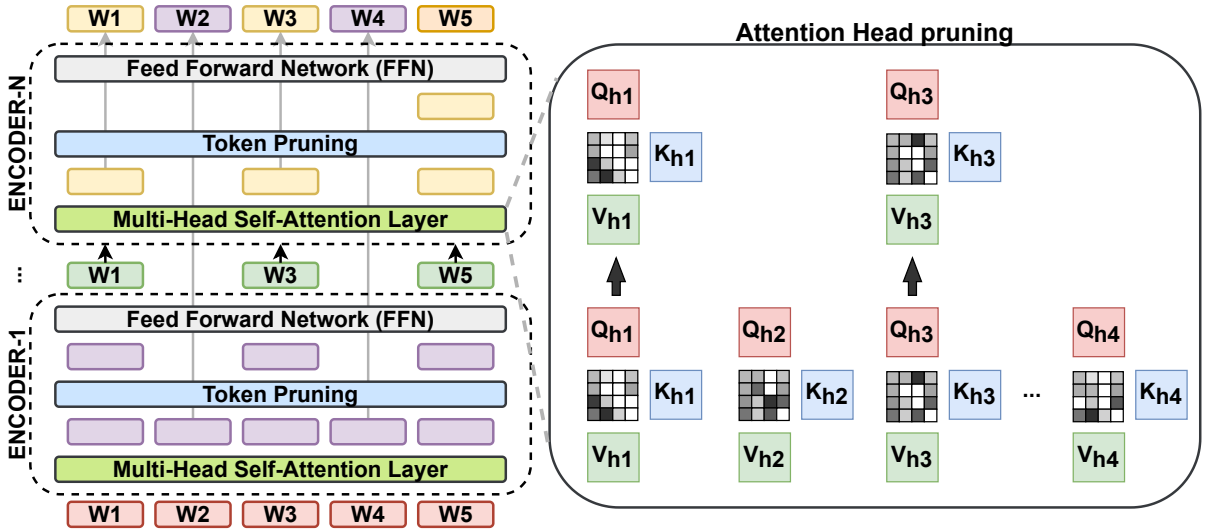

Figure 1: Overview of joint token and head pruning. The encoder blocks on the left show pruned token features as they pass to the next layer and restored at the last layer. On the right, only a subset of attention heads are aggregated to update the retained tokens to further reduce computational cost while achieving higher accuracy with better generalization.

2.3 Length Adaptive Transformer

Based on PoWER-BERT, (Kim and Cho, 2021) proposed a framework to train Length-Adaptive Transformers to reduce expensive computational cost caused by the repetitive search and re-training phases from PoWER-BERT. They train the model once with random sequence length configurations to make the model robust to various token drop during inference. Then, multi-objective evolutionary search on the adaptive model provides sequence length configurations that result in optimal trade-offs between accuracy and efficiency for any given computational budget. They also introduced drop-and-restore process to separately store the dropped token features and restore them in the final hidden layer, which makes the model applicable to token-level applications such as span-based question answering.

Our work combines findings of (Voita et al., 2019; Michel et al., 2019) and Length Adaptive Transformer to design joint token and attention head adaptive transformers which maximizes the accuracy-efficiency trade-offs.

3 Token and Head Adaptive Transformer

In this section, we describe the pruning strategy based on attention importance and head sensitivity score to remove tokens and attention heads respectively. We train Token and Head Adaptive Transformers with the pruning strategy to make the final models robust to arbitrary attention head and token drops at inference time. Fig. 1 illustrates how tokens and heads are jointly pruned in Transformer models. Then we conduct a multi-objective evolutionary search on the trained model to find the optimal pruning configuration that meets the target computational budget.

3.1 Attention score for token pruning

The importance scoring function for a sequence of tokens is based on the self-attention score.

$$I_t(x) = \alpha_t(x) = softmax\left(\frac{x^\top W_q^\top W_k x_t}{\sqrt{d}}\right) \tag{4}$$

The function measures the attention imposed by x_t on the other words $x \in \mathbf{x}$. Intuitively, if the attention score for a token x has a high value then it is likely to influence the final model decision. The significance score of x is the overall attention score aggregated over the heads. Our adaptive model dynamically prunes tokens based on the self-attention scores online.

3.2 Sensitivity score for head pruning

Following the first order method from (Michel et al., 2019; Molchanov et al., 2016), the sensitivity score is based on an unlearned gradient measure of attention heads.

$$I_h = \mathbb{E}_{x\sim X}\left|Attention_h(x)^T \frac{\partial \mathcal{L}(x)}{\partial Attention_h(x)}\right| \tag{5}$$

where X is the data distribution and $\mathcal{L}(x)$ is the loss on sample x. In our framework, we estimate the sensitivity score of each head on a training dataset and use the score as a proxy for determining which heads to prune in each layer. Without additional training cost, smaller models with a fewer number of attention heads could be efficiently sampled from our trained model using weight sharing.

3.3 Training Token and Head Adaptive Transformer

First, we measure the sensitivity score of all the attention heads by running a single forward pass of training dataset on a baseline transformer model fine-tuned for a downstream task. Then, we randomly generate a sequence length and the head configuration at each iteration.

For the head configuration, we sequentially sample the number of heads m_{i+1} at the $(i+1)$-th layer within the range $[m'_{i+1}, m_i]$. m'_{i+1} is the minimum number of retained heads in $(i+1)$-th layer set from the lower bound head pruning configuration which is progressively reducing the number of heads at a constant rate to retain only one attention head in the last layer. We set the lower bound configuration to retain a single head in the last layer based on the empirical results from (Michel et al., 2019) that found ablating all heads except one within a single layer does not significantly impact the performance. m_i is the number of retained heads in the previous layer.

For the token configuration, we followed (Kim and Cho, 2021) and sequentially sample the number of retained token n_{i+1} at the $(i+1)$-th layer within the range$[(1-p)n_i, n_i]$ where n_i is the number of retained tokens in the previous layer and p is the token drop probability. Additionally, we apply LayerDrop (Fan et al., 2019) and randomly dropout encoder layers during training to make the model robust to the random token drop.

We applied the sandwich rule training technique (Yu and Huang, 2019) to avoid difficulty in convergence as discussed in (Kim and Cho, 2021). First, we optimize the model with the upper bound configuration where none of the tokens and heads are eliminated. Then we apply in-place distillation to update the model with the lower bound with the maximum pruning configuration and with other randomly sampled intermediate pruning configurations to transfer the knowledge from the full model to the sparse models with various pruning schemes. In each iteration, the full model and the sparse model are optimized simultaneously. In practice, Token and Head Adaptive Transformer only needs to be trained for the same steps as Length Adaptive Transformer model. After training we get the superior accuracy-latency trade-off of sampled token and head pruning compared to Length Adaptive Transformer models.

3.4 Evolutionary Search of head and token pruning configurations

After training a Token and Head Adaptive Transformer, we apply evolutionary search to find the optimal token and head configurations for the model that satisfy the target computational budgets. Compared to training models specialized for every scenario, evolutionary search is computationally efficient. It only requires inference on small validation set for each pruning configuration.

We first initialize the population of joint token and head pruning configurations with constant drop ratios. The joint configurations are generated with evenly spaced token and head drop rates to have the initial population uniformly distributed between the upper bound and the lower bound pruning configurations. At each iteration, we evolve the population to consist only of configurations with the optimal accuracy-efficiency trade-offs that lie on a new Pareto frontier by mutation and crossover. A mutation alters an original pruning configuration $(j_1, \cdots, j_L)$ to $(j'_1, \cdots, j'_L)$. It involves a probability that an arbitrary element j_i for i-th layer in a pruning configuration $(j_1, \cdots, j_L)$ will be altered to a new value j'_i sampled from the uniform distribution $\left(j'_{i-1}, j_{i+1}\right)$. A crossover takes two joint configurations and averages the pruning values at each layer. In each evolution iteration, we maintain n_m joint configurations from mutation and n_c joint configurations from the crossover. The final iteration will generate the furthest Pareto frontier that consists of the joint configurations with the optimal accuracy-efficiency trade-offs.

4 Experiments

We conduct extensive experiments to evaluate our framework and compare to the baseline models and Length-Adaptive Transformers.

4.1 Model and Data

We investigate three Transformer-based models: BERT, DistilBERT, and MobileBERT. BERT is a language model with a Transformer encoder as

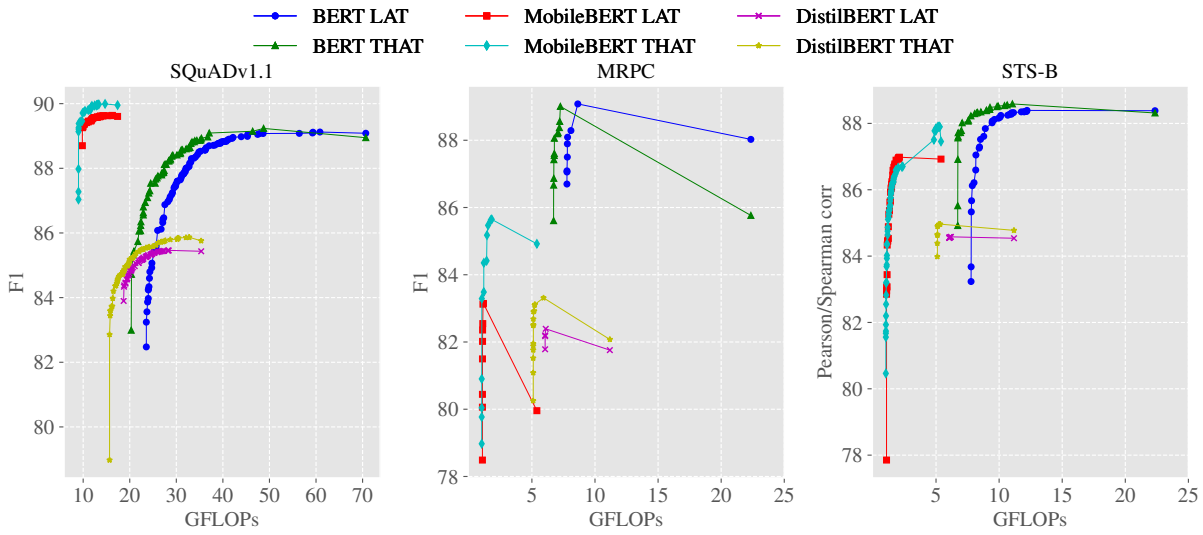

Figure 2: Pareto frontier curves of F1 score (for SQuAD v1.1 and MRPC) and Pearson/Spearman correlation score (for STS-B) to GFLOPs. We apply the proposed method to BERT, DistilBERT, and MobileBERT and compare them to the Length-Adaptive Transformers.

its base building block. The model consists of 12 encoder layers and 12 self-attention heads in each layer (144 heads in total). DistilBERT is a compact Transformer model based on BERT. Knowledge distillation is performed during pre-training to transfer knowledge from BERT teacher model to DistilBERT. DistilBERT has 6 encoder layers and 12 self-attention heads in each layer (72 heads in total). MobileBERT is a thin version of BERT_{Large} that has bottleneck structures to significantly compress the model size. The model consists of 24 encoder layers and 4 self-attention heads in each layer (96 heads in total). We use the Hugging Face implementation and focus on `base-uncased` models. We use Stanford Question Answering Dataset (SQuAD v1.1) and GLUE benchmarks (Wang et al., 2018) including CoLA, STS-B, MRPC, SST-2, QNLI, QQP, and MNLI-mm to evaluate the proposed approach.

4.2 Baseline methods and evaluation metrics

We compare our approach to standard Transformer models fine-tuned on downstream tasks and Length-Adaptive Transformers that are further trained on downstream tasks with token drops. We focus on comparing the inference efficiency with three different metrics: the number of trainable parameters, CPU wall clock time, and the number of floating operations (FLOPs), which is independent of hardware and validated to have linear correlations with CPU latency on (Kim and Cho, 2021). We measured the average CPU latency across the validation dataset with Intel i9-9980XE using 18 threads.

	SQuAD v1.1	
Model	F1	FLOPs
BERT	88.39	1.00x
$\text{BERT}_{LAT}{}^{\dagger}$	89.12	0.86x
$\text{BERT}_{LAT}{}^{\star}$	87.39	0.41s
$\text{BERT}_{THAT}{}^{\dagger}$	89.23	0.68x
$\text{BERT}_{THAT}{}^{\star}$	87.53	0.34x
DistilBERT	85.69	1.00x
$\text{DistilBERT}_{LAT}{}^{\dagger}$	85.46	0.80x
$\text{DistilBERT}_{LAT}{}^{\star}$	84.69	0.55x
$\text{DistilBERT}_{THA}{}^{\dagger}$	85.87	0.92x
$\text{DistilBERT}_{THA}{}^{\star}$	84.70	0.50x
MobileBERT	89.31	1.00x
$\text{MobileBERT}_{LAT}{}^{\dagger}$	89.63	0.93x
$\text{MobileBERT}_{LAT}{}^{\star}$	88.69	0.56x
$\text{MobileBERT}_{THA}{}^{\dagger}$	89.99	0.84x
$\text{MobileBERT}_{THA}{}^{\star}$	89.13	0.51x

Table 1: Comparison of Length Adaptive Transformers and Token and Head Adaptive Transformers on SQuAD v1.1. † denotes the most efficient model while having the highest accuracy among the Pareto frontier. ⋆ denotes the most efficient model with accuracy within 1 percent of the standard model accuracy.

4.3 Hyperparameters

We fine-tune pre-trained BERT models for downstream tasks for 3 epochs without any pruning according to (Goyal et al., 2020). Then we further fine-tune BERT and DistilBERT on our token and head adaptive framework with sequence drop rate

and layer drop rate set to 0.2, which is the same rate used for training a Length Adaptive Transformer. For MobileBERT, we train with a sequence drop rate set to 0.05 and a layer drop rate set to 0.15 to avoid an excessive amount of tokens pruned across 24 encoder layers. The head drop rate is set to a constant rate based on the number of layers of the models to have a single head remaining in the last layer. We fine-tuned with 2 randomly sampled intermediate pruning configurations in addition to upper and lower bound configurations to apply the sandwich training technique. We fine-tuned for 5 epochs on all the benchmarks. We use the batch size of 16 for BERT and 32 for DistilBERT and MobileBERT. The learning rate is set to 5e-5 for SQuAD and 2e-5 for GLUE benchmarks. For MobileBERT, the learning rate is set to 5e-5 for both SQuAD and GLUE benchmarks. The maximum sequence length is set to 384 for SQuAD v1.1 and 128 for SST-2, QQP, MNLI-mm, STS-B, and QNLI, and 64 for CoLA.

For evolutionary search, we run 30 iterations of evolutionary search with 30 mutation pruning configurations with a mutation probability of 0.5 and 30 crossover pruning configurations on each iteration to find the accuracy-efficiency Pareto frontier.

5 Results and Analysis

In this section, we evaluate the experiments and analyze the pruning results in terms of the accuracy-efficiency trade-offs and head distribution.

5.1 Pareto frontier

We compare the accuracy-efficiency Pareto frontier of Token and Head Adaptive Transformers and Length Adaptive Transformers on the SQuAD and GLUE benchmarks. As shown in Fig. 2, the Pareto curve of the Token and Head Adaptive Transformers have a larger area-under-curve (AUC) compared to those of the Length Adaptive Transformers. The Pareto frontier for the SQuAD benchmark shows that all the BERT models fine-tuned with the proposed framework generate joint pruning schemes with superior accuracy-latency trade-offs. It is noticeable in the MRPC Pareto curve that the sparse models with certain pruning configurations have even higher accuracy with significantly less number of floating point operations compared to the full model (rightmost point on Pareto curve) for both Length Adaptive Transformers and Token and Head Adaptive Transformers. Furthermore, the full model fine-tuned with the proposed framework achieves higher accuracy compared to those of Length Adaptive Transformers under the same fine-tuning setup with equal number of training iterations and learning rate. This shows that the models trained with the proposed framework generalize better compared to the standard fine-tuned models and the Length Adaptive Transformers. As we try to reduce the computational effort and reduce the GFLOPs, our models trained with the joint pruning policy remove unnecessary distractors both in terms of tokens and heads leading to boosting the accuracy. Beyond a certain point of pruning drops essential information as well showing the trend towards lowering accuracy consequently. This describes a general behavior that depends on the actual pruning that happens, and it is very data sensitive. This trend is also shown for the STS-B Pareto curve for BERT and DistilBERT.

Additionally, we find that BERT model fine-tuned with the joint pruning framework generates significantly better accuracy-efficiency trade-offs compared to the standard DistilBERT model when enough computational budget is allowed. For example, with the computational budget of 25 GFLOPs, BERT model outperforms DistilBERT by achieving 5 percent higher accuracy while requiring the same number of GFLOPs. Similar trends can be found in the other two benchmarks as well.

5.2 Maximizing efficiency gain

We also measure how much efficiency could be gained with less than 1 percent accuracy loss from the standard fine-tuned models. Table. 1 shows the accuracy and the efficiency gain of the standard Transformers, Length Adaptive Transformers, and Token and Head Adaptive Transformers on the SQuAD benchmark. To validate our proposed framework could find joint pruning configurations that maximize both accuracy and efficiency, we searched two pruning configurations for each model that maximize accuracy and efficiency gain respectively. † denotes the most efficient model while having the highest accuracy. ⋆ denotes the model with the highest efficiency gain that has an accuracy within 1 percent of the standard model accuracy. For all cases, Token and Head Adaptive Transformers achieves higher accuracy and higher efficiency gain. Fig. 2 provides further insights and shows the Pareto frontier curves of our models are shifted upper-left compared to the curves from Length-Adaptive Transformers. This shows that

Model	Accuracy/FLOPs CoLA	STS-B	MRPC	SST-2	QNLI	MNLI-mm	QQP	GLUE
BERT	55.3/1.00x	87.3/1.00x	89.3/1.00x	92.6/1.00x	90.8/1.00x	83.0/1.00x	88.4/1.00x	83.8/1.00x
BERT$_{LAT}$†	57.7/0.38x	88.4/0.55x	89.0/0.39x	92.7/0.36x	91.1/0.55x	84.1/0.53x	89.6/0.60x	84.6/0.48x
BERT$_{LAT}$*	56.2/0.34x	86.6/0.36x	89.0/0.39x	92.4/0.35x	89.8/0.35x	83.2/0.34x	89.4/0.35x	83.8/0.35x
BERT$_{THAT}$†	58.1/0.31x	88.6/0.49x	89.0/0.32x	92.5/0.31x	91.1/0.46x	83.9/0.48x	89.5/0.41x	84.7/0.40x
BERT$_{THAT}$*	56.8/0.30x	86.9/0.30x	89.0/0.32x	91.6/0.30x	89.8/0.31x	83.3/0.31x	89.4/0.30x	83.8/0.31x
DistilBERT	40.2/1.00x	83.4/1.00x	83.2/1.00x	90.2/1.00x	87.1/1.00x	80.0/1.00x	87.1/1.00x	78.7/1.00x
DistilBERT$_{LAT}$†	44.1/0.54x	84.6/0.55x	82.4/0.54x	90.7/0.54x	88.1/0.63x	81.2/0.56x	88.6/0.59x	80.0/0.56x
DistilBERT$_{LAT}$*	44.1/0.54x	84.5/0.54x	82.4/0.54x	90.7/0.54x	87.8/0.54x	81.2/0.56x	88.5/0.54x	79.9/0.54x
DistilBERT$_{THAT}$†	45.9/0.46x	84.9/0.48x	83.3/0.52x	91.2/0.46x	88.0/0.59x	81.2/0.46x	88.6/0.46x	80.4/0.49x
DistilBERT$_{THAT}$*	45.9/0.46x	84.0/0.45x	82.4/0.45x	91.2/0.46x	86.6/0.45x	81.2/0.46x	88.6/0.46x	80.0/0.45x
MobileBERT	46.0/1.00x	84.0/1.00x	85.1/1.00x	91.3/1.00x	87.0/1.00x	81.1/1.00x	86.0/1.00x	80.1/1.00x
MobileBERT$_{LAT}$†	49.6/0.20x	86.9/0.39x	83.1/0.22x	92.4/0.25x	89.6/0.46x	83.0/0.71x	88.3/0.39x	81.6/0.37x
MobileBERT$_{LAT}$*	49.6/0.20x	83.1/0.20x	-	90.9/0.20x	86.2/0.23x	80.3/0.26x	85.7/0.20x	79.3/0.21x
MobileBERT$_{THAT}$†	53.6/0.25x	87.9/0.92x	85.6/0.33x	92.3/0.80x	90.2/0.65x	83.2/0.94x	88.3/0.35x	83.0/0.60x
MobileBERT$_{THAT}$*	50.1/0.21x	83.3/0.20x	84.3/0.22x	91.9/0.19x	86.1/0.22x	80.2/0.25x	86.3/0.19x	80.3/0.21x

Table 2: Results are evaluated on the test set of GLUE benchmark. † and ⋆ denote the same optimal models as described in Table 1.

our model trained with the joint pruning scheme produce models with less computational cost and higher accuracy. Our token and head adaptive BERT model achieves higher accuracy and only requires 34% of the number of floating point operations from the baseline model, which is 7% smaller than Length-Adaptive Transformer.

Table. 3 also shows the number of learnable parameters and CPU latency of the models with maximum efficiency gains. Our adaptive BERT model achieves higher accuracy while it is 15% faster and 20% smaller. Our adaptive DistilBERT and MobileBERT are 12% and 9% faster than Length-Adaptive Transformer models while achieving better accuracy.

The overall GLUE scores for the two pruning configurations with the highest accuracy and the maximum latency gain also validate the token and head adaptive model with joint pruning configurations achieves better efficiency gain and higher accuracy. MobileBERT trained with our joint token and head pruning scheme achieves a GLUE score of 83.0, which is 1.9 higher than the standard MobileBERT and 1.4 higher than Length-Adaptive Transformers.

5.3 Head distribution

We further analyze the joint pruning configurations and compare with length pruning schemes from Length-Adaptive Transformers. Fig. 3 shows the best pruning configurations of Length Adaptive and Token and Head Adaptive Transformers that maximize efficiency within 1 percent of the standard model accuracy. For SST-2 benchmark, the 3 token and head adaptive BERT-based models only retain a single head in the last layer. Table. 3 shows that 48%, 24%, and 43% of the attention heads are pruned from BERT, DistilBERT and MobileBERT respectively and still performs superior to the standard fine-tuned models and Length-Adaptive Transformers. On SQuAD benchmark, the token and head adaptive models progressively reduce the attention heads where more than 50% of the attention heads are pruned in the last layer.

Model	Param(MB)	Lat(ms)	heads	tokens
BERT$_{LAT}$*	415.4	385.7	144	1846
BERT$_{THAT}$*	333.3	333.4	77	1782
DistilBERT$_{LAT}$*	253.2	251.4	72	1194
DistilBERT$_{THAT}$*	240.3	223.1	55	1174
MobileBERT$_{LAT}$*	93.8	360.4	96	5266
MobileBERT$_{THAT}$*	91.1	331.0	55	4891

Table 3: The number of parameters, CPU latency, the number of attention heads, and the number of tokens for BERT-based models that maximize efficiency on SQuAD benchmark.

The token and head adaptive MobileBERT tends to retain slightly more tokens in the center layers compared to Length Adaptive Transformer and reduce more tokens in the later layers. Our adaptive model have higher token drop rate. This shows the final models trained with our framework are robust to prune even more tokens along with attention heads.

6 Related Work

Transformers and BERT have outperformed existing language based models and achieved state-of-the-art results but at the cost of high computational power requirements. In order to make these models efficient in terms of memory and power footprint,

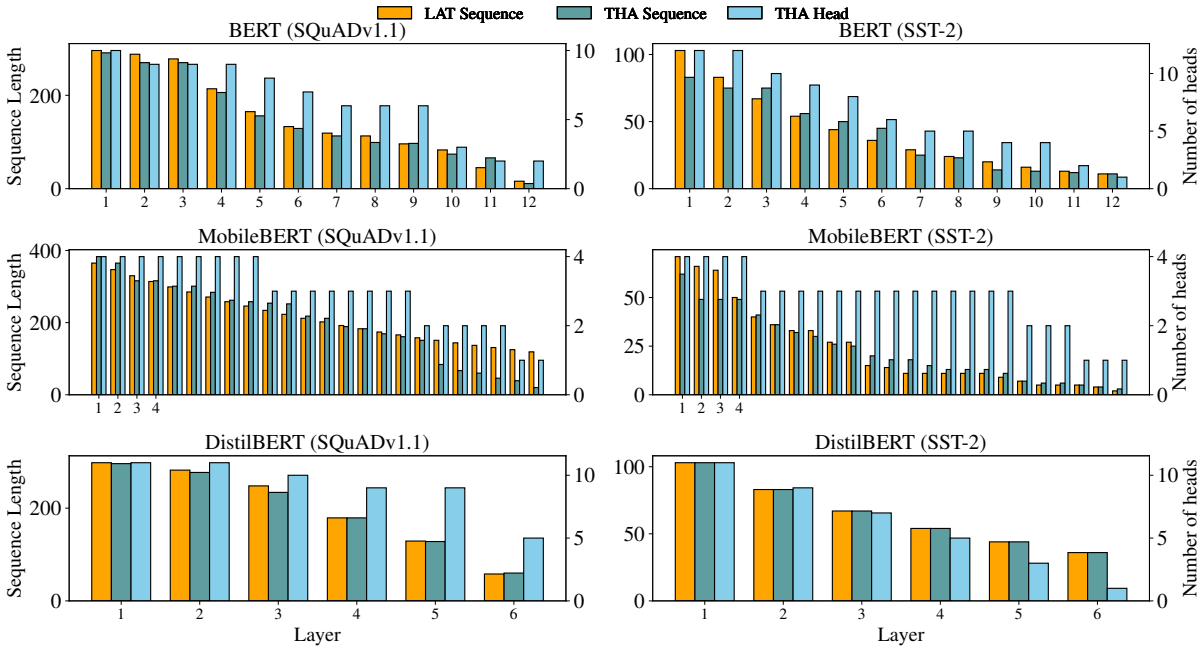

Figure 3: Comparison of the Sequence length configurations of the ⋆ models with the maximum efficiency gains on SQuAD and SST-2 benchmarks. LAT Sequence represents the token length configuration from Length-Adaptive Transformer while THA for Token and Head Adaptive Transformers. THA Head shows the retaining head distribution of Token and Head Adaptive models.

several strategies have been proposed. Here we discuss a few key studies.

While large networks are essential for the best performance, the model size becomes an obstacle when it comes to scaling and also leads to worse training time. ALBERT (Lan et al., 2020) targets this limitation and proposed parameter reduction training by factoring the embedding matrices and sharing the parameters across layers which resulted in a smoother up-scaling up of the base model as well as significantly less training time. However, in terms of inference latency, there is not much improvement. PoWER-BERT (Goyal et al., 2020) points out this fact and proposes an orthogonal approach of progressive fine-grained word-vector elimination that can reduce the inference time up to 4.5x without harming the accuracy much. However, PoWER-BERT requires repetitive training for each different constraint and is limited to sequence-level classification only. Length adaptive transformer or LAT (Kim and Cho, 2021) alleviates these shortcomings by introducing a LengthDrop method that automatically derives multiple sub-models using evolutionary search without requiring any re-training. They also proposed a Drop-and-Restore process to set aside the word-vectors so that they can be restored at the final layers extending the PoWER-BERT beyond classification tasks to a broader range of NLP tasks.

Parallel to these studies that deal with token pruning, another line of work investigates whether we need so many attention heads. (Voita et al., 2019) identifies the most important attention heads using layer-wise relevance propagation or LRP (Ding et al., 2017) and gradually prunes the head accordingly. In a complementary approach, (Michel et al., 2019) finds the most relevant heads by masking or ablating one or more heads and looking at their impact on performance. In this way, they calculate a proxy importance score and iteratively prune the heads according to that without degrading the performance significantly. Another group of studies concentrates on model architecture and parameter size. Where DistilBERT (Sanh et al., 2019) uses a teacher-student setting to transfer knowledge from a large model to its small student version to reduce the model parameters, MobileBERT (Sun et al., 2020) reduces it by leveraging the bottleneck structure.

While both token and head pruning improves the model's efficiency i.e. less inference time, lower memory, and power footprint, to the best of our knowledge, no study has tried to optimize both of them simultaneously which precisely lays down the basis of our work.

7 Conclusion and Future Work

In this work, we propose a joint token and head pruning framework to train robust Transformer models adaptive to token and head dropouts. Our idea is to calculate head importance from the self-attention operation and eliminate redundant heads from the retained tokens to further reduce the computational cost and train the model adaptive and perform better generalization. The final models trained with the proposed framework show better generalization and greater efficiency gains. Token and Head Adaptive Transformers are robust to prune out more tokens than Length-Adaptive Transformers along with the attention heads. This leads to superior accuracy-latency trade-offs and larger area-under-curve of the Pareto frontier found with an evolutionary search. We would also like to explore further in the direction of reducing the computational cost of large-scale models by eliminating redundant operations and applying efficient optimization techniques to search for efficient models with minimum resources.

Acknowledgements

This work was supported in part by the Center for Brain-inspired Computing (C-BRIC) and the Center for Research in Intelligent Storage and Processing in Memory (CRISP), two of the six centers in JUMP, a Semiconductor Research Corporation (SRC) program sponsored by DARPA.

References

Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel Ziegler, Jeffrey Wu, Clemens Winter, Chris Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. 2020. Language models are few-shot learners. In *Advances in Neural Information Processing Systems*, volume 33, pages 1877–1901. Curran Associates, Inc.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, Minneapolis, Minnesota. Association for Computational Linguistics.

Yanzhuo Ding, Yang Liu, Huanbo Luan, and Maosong Sun. 2017. Visualizing and understanding neural machine translation. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1150–1159, Vancouver, Canada. Association for Computational Linguistics.

Angela Fan, Edouard Grave, and Armand Joulin. 2019. Reducing transformer depth on demand with structured dropout. *arXiv preprint arXiv:1909.11556*.

Saurabh Goyal, Anamitra Roy Choudhury, Saurabh M. Raje, Venkatesan T. Chakaravarthy, Yogish Sabharwal, and Ashish Verma. 2020. Power-bert: Accelerating bert inference via progressive word-vector elimination. In *Proceedings of the 37th International Conference on Machine Learning*, ICML'20. JMLR.org.

Katyanna Quach. 2020. Ai me to the moon... carbon footprint for 'training gpt-3' same as driving to our natural satellite and back. [online]. Available: https://www.theregister.com/2020/11/04/gpt3_carbon_footprint_estimate/. [Accessed: May-17-2022].

Gyuwan Kim and Kyunghyun Cho. 2021. Length-adaptive transformer: Train once with length drop, use anytime with search. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 6501–6511, Online. Association for Computational Linguistics.

Zhenzhong Lan, Mingda Chen, Sebastian Goodman, Kevin Gimpel, Piyush Sharma, and Radu Soricut. 2020. Albert: A lite bert for self-supervised learning of language representations. In *International Conference on Learning Representations*.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Roberta: A robustly optimized BERT pretraining approach. *CoRR*, abs/1907.11692.

Paul Michel, Omer Levy, and Graham Neubig. 2019. Are sixteen heads really better than one? *Advances in neural information processing systems*, 32.

Pavlo Molchanov, Stephen Tyree, Tero Karras, Timo Aila, and Jan Kautz. 2016. Pruning convolutional neural networks for resource efficient inference. *arXiv preprint arXiv:1611.06440*.

Pranav Rajpurkar, Jian Zhang, Konstantin Lopyrev, and Percy Liang. 2016. SQuAD: 100,000+ questions for machine comprehension of text. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 2383–2392, Austin, Texas. Association for Computational Linguistics.

Victor Sanh, Lysandre Debut, Julien Chaumond, and Thomas Wolf. 2019. DistilBERT, a distilled version of BERT: smaller, faster, cheaper and lighter. In *5th Workshop on Energy Efficient Machine Learning and Cognitive Computing @ NeurIPS 2019*.

Zhiqing Sun, Hongkun Yu, Xiaodan Song, Renjie Liu, Yiming Yang, and Denny Zhou. 2020. MobileBERT: a compact task-agnostic BERT for resource-limited devices. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 2158–2170, Online. Association for Computational Linguistics.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Ł ukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc.

Elena Voita, David Talbot, Fedor Moiseev, Rico Sennrich, and Ivan Titov. 2019. Analyzing multi-head self-attention: Specialized heads do the heavy lifting, the rest can be pruned. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 5797–5808, Florence, Italy. Association for Computational Linguistics.

Alex Wang, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel R Bowman. 2018. Glue: A multi-task benchmark and analysis platform for natural language understanding. *arXiv preprint arXiv:1804.07461*.

Jiahui Yu and Thomas S Huang. 2019. Universally slimmable networks and improved training techniques. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 1803–1811.

Don't Judge a Language Model by Its Last Layer: Contrastive Learning with Layer-Wise Attention Pooling

Dongsuk Oh[*‡] **Yejin Kim**[*§] **Hodong Lee**[‡] **H. Howie Huang**[†§] **and Heuiseok Lim**[†‡]
[‡]Computer Science and Engineering, Korea University
[§]Graph Lab., George Washington University
{inow3555,bigshane319,limhseok}@korea.ac.kr
{yejinjenny, howie}@gwu.edu

Abstract

Recent pre-trained language models (PLMs) achieved great success on many natural language processing tasks through learning linguistic features and contextualized sentence representation. Since attributes captured in stacked layers of PLMs are not clearly identified, straightforward approaches such as embedding the last layer are commonly preferred to derive sentence representations from PLMs. This paper introduces the attention-based pooling strategy, which enables the model to preserve layer-wise signals captured in each layer and learn digested linguistic features for downstream tasks. The contrastive learning objective can adapt the layer-wise attention pooling to both unsupervised and supervised manners. It results in regularizing the anisotropic space of pre-trained embeddings and being more uniform. We evaluate our model on standard semantic textual similarity (STS) and semantic search tasks. As a result, our method improved the performance of the base contrastive learned BERT$_{base}$ and variants.

1 Introduction

Pre-trained language models (PLMs) (Kenton and Toutanova, 2019; Liu et al., 2019; Radford et al., 2019; Raffel et al., 2019) have shown competitive performance on many natural language processing (NLP) tasks. Also, contrastive learning using the PLMs shows the highest performance in sentence representation. Contrastive learning is to learn effective representations by staying semantically close sample pairs together while dissimilar ones are far apart(Hadsell et al., 2006).

In general, PLMs use either $[CLS]$ tokens in the last layer, AVG which is the average representation of tokens in the last layer(Reimers et al., 2019; Li et al., 2020), or AVG_{FL} which is the average representation of tokens in the first and last layers(Gao et al., 2021), to pool out sentence representation from word representations. However, since language models show performance gaps by domain when trained on different objectives, the fixed pooling strategy has limitations in performance improvement.

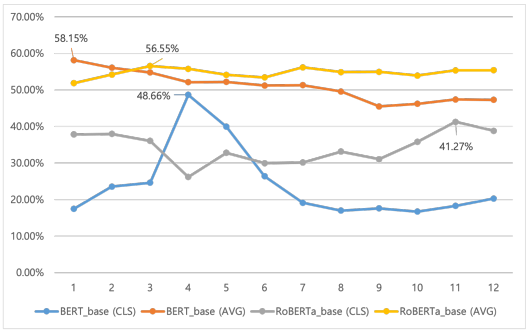

Figure 1: Spearsman's correlation score of each layer evaluated on STS-B test set

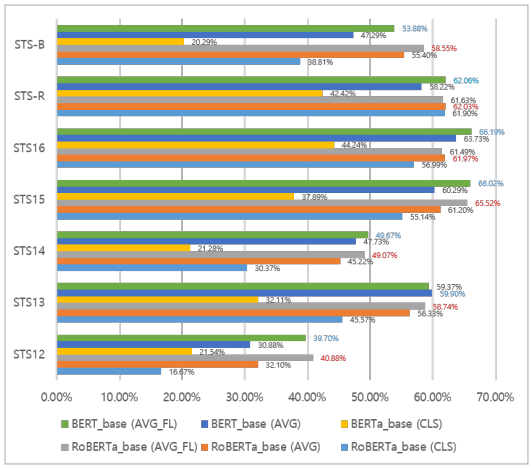

Figure 2: Spearsman's correlation score depending on the pooling methods of PLMs for each domain

Figure 1 and 2 show the Spearman's correlation score of each layer or pooling method in PLMs. We evaluated the test set of the standard semantic textual similarity (STS) dataset(Cer et al., 2017; Agirre et al., 2012, 2013, 2014, 2015, 2016; Marelli

[*] These authors contributed equally.
[†] These authors are corresponding authors.

Proceedings of the 29th International Conference on Computational Linguistics, pages 4585–4592
October 12–17, 2022.

et al., 2014).

The comparison of performance when pooling each layer shown in Figure 1 indicates that using only a specific layer for pooling is insufficient. Other layers other than the last layer may contain substantial information for sentence representation. For example, for the STS benchmark (STS-B) task(Cer et al., 2017), BERT$_{base}$ with $[CLS]$ embedding scored the highest at the fourth layer (48.66%), which is about 20% higher than the last layer.

Figure 2 shows that simply pooling from more layers impedes the performance by comparing models pooled from the first and last layer and the last layer. In addition, there is no consistent tendency to compare effectiveness for a given layer between $[CLS]$ pooling and average pooling.

Motivated by this point, we designed the attention networks and task-agnostic pooling methods to assign more weights to spots that need more focus in the layer and lead to representation vector optimization. Our proposed method outperforms previously fixed pooling strategies in contrastive learning. In addition, contrastive learning models with layer-wise attention pooling show a higher semantic search performance with the same parameters.

In summary, the contributions of this paper are as follows:

- We proposed layer-wise attention pooling* to assign weights to each layer and learn sentence representation fitted to a given task.
- To our knowledge, our pooling strategy shows the best performance out of all InfoNCE-based loss functions for the sentence embedding tasks.
- For the semantic search evaluation, we excluded the proposed pooling method in the inference phase and obtained better performance.

2 Method

In this section, we present a layer-wise pooling strategy based on attention mechanisms to improve the quality of sentence representations from language models. In addition, we describe the process of applying the proposed pooling strategy to be leveraged on three contrastive learning schemes.

*https://github.com/nlpods/LayerAttPooler

2.1 Layer-Wise Attention Pooling

This paper proposes a new layer-wise pooling based on a multiplicative attention mechanism (Luong et al., 2015). As shown in Figure 1, the performance with $[CLS]$ pooling varied dramatically according to which layer to pool from. There is no significant performance gap between layers when using AVG pooling. It can be explained that each layer can contain different information for sentence representation, while average pooling can mitigate the information gap between layers.

In Equation 1, h^a is the AVG representation, which is the mean vector of tokens in the sentence, and h^c is the input representation $[CLS]$ of each layer. α_i is the importance of the i-th layer. In Equation 2, h^l is the representation with the importance score per layer. In Equation 3, h^L is the mean vector of h^l and is the representation that contains the relevance of all layers (N is the number of layers). W_k, W_q and W_v are learnable parameters.

$$\alpha_i = \frac{W_q h_i^c W_k h_i^a}{\sum_{j \in N} W_q h_i^c W_k h_j^a} \tag{1}$$

$$h_i^l = \sum_{j \in N} \alpha_i W_v h_j^a \tag{2}$$

$$h^L = \frac{1}{N} \sum_{i}^{N} h_i^l \tag{3}$$

We add a Multi-Layer Perceptron (MLP) layer randomly initialized after pooling, following the method in the Gao et al. (2021), and keep it with random initialization. As for Equation 4, h_{last}^c is the input representation $[CLS]$ of the last layer. We concatenate the input representation h_{last}^c with the layer representation h^L as the input of an MLP. Finally, h is represented in the same dimension as the sentence representation dimension of the original language model through the MLP layer.

$$h^{CL} = [h_{last}^c; h^L] \tag{4}$$

$$h = MLP(h^{CL}) \tag{5}$$

2.2 Contrastive Learning with Layer-wise Attention Pooling

We prove that the proposed pooling strategy is effective with three training objectives l_i.

Basic Supervised Contrastive Learning We use the basic supervised contrastive learning model proposed by Chen et al. (2020). This model learns the premise(x_i) and entailment(x_i^+) of the NLI(SNLI(Bowman et al., 2015) + MNLI(Williams et al., 2018)) datasets. When $D = (x_i, x_i^+)_{i=1}^m$ is a set of paired samples, where x_i and x_i^+ are semantically related. And, it takes the cross-entropy objective with an in-batch negative(Chen et al., 2017; Henderson et al., 2017). h_i and h_i^+ are representations of x_i and x_i^+ through proposed pooling strategy. the training objective l_i is :

$$l_i = -log \frac{e^{sim(h_i, h_i^+)/\tau}}{\sum_{j=1}^{M} e^{sim(h_i, h_j^+)/\tau}} \quad (6)$$

M is the mini-batch, and τ is the temperature hyperparameter and $sim(\cdot, \cdot)$ is the cosine similarity.

Unsupervised Contrastive Learning Unsupervised contrastive learning uses $x_i^+ = x_i$ in the collection of sentences $\{x_i\}_{i=1}^m$. The idea is to use an independently sampled dropout mask for x_i and x_i^+ which gets this to work as identical positive pairs during training. And, unsupervised contrastive learning denotes $h_i^z = f(x_i, z)$ using h obtained in Equation 5. z is a random mask for dropout. It gets two embeddings with different dropout masks z, $z^{'}$ from the encoder with the same input twice, and the training objective l_i is represented:

$$l_i = -log \frac{e^{sim(h_i^{z_i}, h_i^{z_i^{'}})/\tau}}{\sum_{j=1}^{M} e^{sim(h_i^{z_i}, h_j^{z_j^{'}})/\tau}} \quad (7)$$

In Equation 7, z is the standard dropout of the transformer.

Supervised Contrastive Learning with Hard Negative Supervised contrastive learning with hard negative trains natural language inference (NLI) datasets. The NLI datasets are labeled, given one premise, as true(entailment), neutral, and definitely false (contradiction). The model predicts whether the relationship between two sentences is entailment, neutral, or contradiction. The positive pairs (x_i, x_i^+) use the entailment of the NLI(SNLI + MNLI) datasets. Next, contradiction pairs (x_i, x_i^-) from the NLI datasets are used as hard negatives. Thus, it expands from (x_i, x_i^+) to (x_i, x_i^+, x_i^-). And, (x_i, x_i^+, x_i^-) is represented as (h_i, h_i^+, h_i^-) through Equation 5. As a result, in Equation 8, the training objective l_i is :

$$-log \frac{e^{sim(h_i, h_i^+)/\tau}}{\sum_{j=1}^{M} (e^{sim(h_i, h_j^+)/\tau} + e^{sim(h_i, h_j^-)/\tau})} \quad (8)$$

3 Experiments

3.1 Experimental Setup

Our main experiments uses the STS(Cer et al., 2017; Agirre et al., 2012, 2013, 2014, 2015, 2016; Marelli et al., 2014) dataset. This data set consists of sentence pairs labeled with a similarity score between 0 and 5. The evaluation is done by the SentEval toolkit. The parameter setting of the model used in the experiment is written in Table 4 of the Appendix. Additionally, to measure the search effect and efficiency of the proposed model, it is evaluated on the same parameters as the original language model. We evaluate the performance of the semantic search[†] with FAISS[‡] using the Quora Duplicate Questions Dataset(Shankar et al., 2021) containing more than 400,000 pairs of questions.

3.2 Main Results

In Table 1, we investigate whether the proposed layer-wise attention pooling of language models performs better in contrastive learning. The experiment compares performance by training on language models with three training objectives. All results evaluate sentence embeddings on all STS tasks. Equation 6 is basic supervised learning proposed by (Chen et al., 2020). And, Equations 7 and 8 are unsupervised, supervised learning proposed by Gao et al. (2021). However, in this paper, we could not experiment with the same parameters due to hardware. Therefore, as specified in Table 4 of the Appendix, there is a difference from the original performance because it learns by choosing a low mini-batch size. † is the original performance, and ‡ is our reimplementations. As a result, the proposed pooling strategy shows higher performance in different language models and in all domains.

3.3 Ablation Studies

We investigate performance differences according to different pooling strategies in supervised contrastive learning. All results are reported in this

[†]https://github.com/autoliuweijie/BERT-whitening-pytorch

[‡]https://github.com/facebookresearch/faiss

Unsupervised Models								
Model	**STS12**	**STS13**	**STS14**	**STS15**	**STS16**	**STS-B**	**SICK-R**	**Avg**
BERT$_{base}$(CLS$_{Last}$)(Equation 7)†	68.40	82.41	74.38	80.91	78.56	76.85	72.23	76.25
RoBERTa$_{base}$(CLS$_{Last}$)(Equation 7)†	70.16	81.77	73.24	81.36	80.65	80.22	68.56	76.57
RoBERTa$_{large}$(CLS$_{Last}$)(Equation 7)†	72.86	83.99	75.62	84.77	81.80	81.98	71.23	78.89
Our Reimplementations								
BERT$_{base}$(CLS$_{Last}$)(Equation 7)‡	69.53	78.98	75.50	80.07	79.01	78.28	71.35	76.10
w/ LayerAttPooler(CLS$_{All}$ + AVG$_{All}$ attention) + (CLS$_{Last}$ concat)	**70.27**	**80.22**	**75.65**	**80.71**	**79.74**	**79.51**	**72.18**	**76.90**
RoBERTa$_{base}$(CLS$_{Last}$)(Equation 7)‡	68.72	78.29	74.35	80.40	80.83	80.14	68.71	75.92
w/ LayerAttPooler(CLS$_{All}$ + AVG$_{All}$ attention) + (CLS$_{Last}$ concat)	**68.96**	**78.83**	**75.37**	**81.05**	**81.53**	**80.99**	**69.03**	**76.54**
RoBERTa$_{large}$(CLS$_{Last}$)(Equation 7)‡	70.82	79.66	76.26	83.25	81.86	81.25	71.09	77.74
w/ LayerAttPooler(CLS$_{All}$ + AVG$_{All}$ attention) + (CLS$_{Last}$ concat)	**71.52**	**79.86**	**76.86**	**83.50**	**82.38**	**84.56**	**71.46**	**78.59**
Supervised Models								
BERT$_{base}$(CLS$_{Last}$)(Equation 8)†	75.30	84.67	80.19	85.40	80.82	84.25	80.39	81.57
RoBERTa$_{base}$(CLS$_{Last}$)(Equation 8)†	76.53	85.21	80.95	86.03	82.57	85.83	80.50	82.52
RoBERTa$_{large}$(CLS$_{Last}$)(Equation 8)†	77.46	87.27	82.36	86.66	83.93	86.70	81.95	83.76
Our Reimplementations								
BERT$_{base}$(CLS$_{Last}$)(Equation 8)‡	70.50	80.77	79.52	83.82	81.17	84.34	79.04	79.88
w/ LayerAttPooler(CLS$_{All}$ + AVG$_{All}$ attention) + (CLS$_{Last}$ concat)	**71.34**	**80.84**	**79.76**	**83.86**	**81.42**	**86.76**	**79.80**	**80.54**
RoBERTa$_{base}$(CLS$_{Last}$)(Equation 8)‡	70.80	81.31	79.60	83.48	82.86	85.71	79.77	80.50
w/ LayerAttPooler(CLS$_{All}$ + AVG$_{All}$ attention) + (CLS$_{Last}$ concat)	**71.35**	**81.44**	**79.82**	**83.79**	**83.89**	**87.42**	**80.11**	**81.12**
RoBERTa$_{large}$(CLS$_{Last}$)(Equation 8)‡	72.36	83.06	81.99	85.39	85.51	87.11	80.46	82.27
w/ LayerAttPooler(CLS$_{All}$ + AVG$_{All}$ attention) + (CLS$_{Last}$ concat)	**72.65**	**84.41**	**82.31**	**86.38**	**85.54**	**87.58**	**81.56**	**82.92**
BERT$_{base}$(CLS$_{Last}$)(Equation 6)	69.29	78.69	76.45	80.87	79.82	79.41	76.41	77.28
w/ LayerAttPooler(CLS$_{All}$ + AVG$_{All}$ attention) + (CLS$_{Last}$ concat)	**69.58**	**78.84**	**76.70**	**81.13**	**80.11**	**87.23**	**76.45**	**78.58**
RoBERTa$_{base}$(CLS$_{Last}$)(Equation 6)	68.85	77.28	74.67	80.11	80.80	87.42	76.51	77.95
w/ LayerAttPooler(CLS$_{All}$ + AVG$_{All}$ attention) + (CLS$_{Last}$ concat)	**69.51**	**78.72**	**75.97**	**81.32**	**81.47**	**89.58**	**76.83**	**79.06**
RoBERTa$_{large}$(CLS$_{Last}$)(Equation 6)	70.82	80.33	77.79	82.03	83.04	85.38	76.84	79.46
w/ LayerAttPooler(CLS$_{All}$ + AVG$_{All}$ attention) + (CLS$_{Last}$ concat)	**71.15**	**81.45**	**78.04**	**83.03**	**83.09**	**88.88**	**77.39**	**80.43**

Table 1: Performance of sentence embedding on all STS tasks (Spearman's correlation). †: published in Gao et al. (2021); and ‡: models from our reimplementations. We are shown in bold the highest performance among models from our reimplementation.

section using the STS-B test set. All models extract sentence embeddings by adding an MLP layer as suggested in Gao et al. (2021). Table 2 shows the performance difference between the fixed pooling method and the layer-wise attention pooling. Additionally, we compare the representation concatenated between fixed pooling because we construct the h representation by concatenating h^c_{last} and h^L. The layer-wise attention pooling shows the results of ablation studies with $[CLS]$ and AVG. For $[CLS]_{All}$ and AVG_{All}, h^l computes the importance between each layer and the others. In addition, $[CLS]_{All}$ + AVG_{All} represent h^l by calculating the importance between $[CLS]$ and AVG of all layers. All of these methods show higher performance than the fixed pooling strategy. However, as described in Section 2, the pooling strategy concatenated with $[CLS]_{Last}$ shows the highest performance.

Model	**STS-B**
BERT$_{base}$(Equation 8)	
w/ (CLS$_{Last}$)	84.34
w/ (AVG$_{Last}$)	84.84
w/ (AVG$_{FL}$)	84.76
w/ (AVG$_{Last}$+AVG$_{FL}$ concat)	84.93
w/ (CLS$_{Last}$+AVG$_{Last}$ concat)	85.11
w/ LayerAttPooler(CLS$_{All}$ attention)	85.45
w/ LayerAttPooler(AVG$_{All}$ attention)	85.72
w/ LayerAttPooler(CLS$_{All}$ + AVG$_{All}$ attention)	86.57
w/ LayerAttPooler(CLS$_{All}$ + AVG$_{All}$ attention) + (CLS$_{Last}$ concat)	**86.76**

Table 2: Ablation studies of different pooling methods in supervised model on STS-B task (Spearman's correlation)

3.4 Semantic Search Results

In Table 3, we compare semantic search speed and performance on the same parameters. This experiment proves that the proposed pooling strategy is effective for training the language models and also for semantic search performance with the same parameters during inference. Sentence embeddings for all supervised learning models use $[CLS]_{Last}$. MRR@10 is used to measure the performance of semantic search, and Average Retrieval Time (ms) measures retrieval efficiency. Memory Usage (GB) shows memory usage. FAISS experiments in CPU mode. nlist = 1024 and the CPU is Intel(R) Xeon(R) Gold 6230R CPU @ 2.10GHz. Result shows that the performance of semantic search is higher when the proposed pooling strategy is used during training.

Model	MRR@10	Average Retrieval Time (ms)	Memory Usage (GB)
BERT$_{base}$(Equation 8)			
w/ (CLS$_{Last}$)	63.48	1.46	0.25
w/ LayerAttPooler (train)	**64.32**	**1.45**	0.25
RoBERTa$_{base}$(Equation 8)			
w/ (CLS$_{Last}$)	63.89	1.56	0.25
w/ LayerAttPooler (train)	**65.05**	**1.48**	0.25
RoBERTa$_{large}$(Equation 8)			
w/ (CLS$_{Last}$)	65.85	2.22	0.33
w/ LayerAttPooler (train)	**66.32**	**2.21**	0.33

Table 3: Performance of semantic search evaluation using the Quora Duplicate Questions Dataset with FAISS. w/ LayerAttPooler (train) : remove layer-wise attention pooling after training

4 Conclusion

In this work, we propose layer-wise attention pooling to capture the importance of the weight in each layer for the pre-trained language models (PLMs). Training layer-wise attention layer with contrastive learning objectives outperforms BERT and variants of PLMs. No matter what pooling method is used, our model achieved higher scores than prior state-of-the-art models. In addition, this layer-wise attention technique also can be exploited in semantic search tasks, in which more cost-efficient computation (i.e. less latency and memory usage) is required. The model trained with our method obtained higher performance with the same or less time and memory usage, even if the added attention layer is detached in the inference stage.

Acknowledgements

This work was supported by Institute of Information & communications Technology Planning & Evaluation(IITP) grant funded by the Korea government(MSIT) (No. 2020-0-00368, A Neural-Symbolic Model for Knowledge Acquisition and Inference Techniques) and by the MSIT(Ministry of Science and ICT), Korea, under the ITRC(Information Technology Research Center) support program(IITP-2022-2018-0-01405) supervised by the IITP(Institute for Information & Communications Technology Planning & Evaluation) In addition, This research was supported by Basic Science Research Program through the National Research Foundation of Korea(NRF) funded by the Ministry of Education(NRF-2021R1A6A1A03045425)

References

Eneko Agirre, Carmen Banea, Claire Cardie, Daniel Cer, Mona Diab, Aitor Gonzalez-Agirre, Weiwei Guo, Inigo Lopez-Gazpio, Montse Maritxalar, Rada Mihalcea, et al. 2015. Semeval-2015 task 2: Semantic textual similarity, english, spanish and pilot on interpretability. In *Proceedings of the 9th international workshop on semantic evaluation (SemEval 2015)*, pages 252–263.

Eneko Agirre, Carmen Banea, Claire Cardie, Daniel M Cer, Mona T Diab, Aitor Gonzalez-Agirre, Weiwei Guo, Rada Mihalcea, German Rigau, and Janyce Wiebe. 2014. Semeval-2014 task 10: Multilingual semantic textual similarity. In *SemEval@ COLING*, pages 81–91.

Eneko Agirre, Carmen Banea, Daniel Cer, Mona Diab, Aitor Gonzalez Agirre, Rada Mihalcea, German Rigau Claramunt, and Janyce Wiebe. 2016. Semeval-2016 task 1: Semantic textual similarity, monolingual and cross-lingual evaluation. In *SemEval-2016. 10th International Workshop on Semantic Evaluation; 2016 Jun 16-17; San Diego, CA. Stroudsburg (PA): ACL; 2016. p. 497-511.* ACL (Association for Computational Linguistics).

Eneko Agirre, Daniel Cer, Mona Diab, and Aitor Gonzalez-Agirre. 2012. Semeval-2012 task 6: A pilot on semantic textual similarity. In ** SEM 2012: The First Joint Conference on Lexical and Computational Semantics–Volume 1: Proceedings of the main conference and the shared task, and Volume 2: Proceedings of the Sixth International Workshop on Semantic Evaluation (SemEval 2012)*, pages 385–393.

Eneko Agirre, Daniel Cer, Mona Diab, Aitor Gonzalez-Agirre, and Weiwei Guo. 2013. * sem 2013 shared task: Semantic textual similarity. In *Second joint conference on lexical and computational semantics (* SEM), volume 1: proceedings of the Main conference and the shared task: semantic textual similarity*, pages 32–43.

Samuel Bowman, Gabor Angeli, Christopher Potts, and Christopher D Manning. 2015. A large annotated corpus for learning natural language inference. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 632–642.

Daniel Cer, Mona Diab, Eneko Agirre, Inigo Lopez-Gazpio, and Lucia Specia. 2017. Semeval-2017 task 1: Semantic textual similarity-multilingual and cross-lingual focused evaluation. *arXiv preprint arXiv:1708.00055*.

Ting Chen, Simon Kornblith, Mohammad Norouzi, and Geoffrey Hinton. 2020. A simple framework for contrastive learning of visual representations. In *International conference on machine learning*, pages 1597–1607. PMLR.

Ting Chen, Yizhou Sun, Yue Shi, and Liangjie Hong. 2017. On sampling strategies for neural network-based collaborative filtering. In *Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 767–776.

Yung-Sung Chuang, Rumen Dangovski, Hongyin Luo, Yang Zhang, Shiyu Chang, Marin Soljačić, Shang-Wen Li, Wen-tau Yih, Yoon Kim, and James Glass. 2022. Diffcse: Difference-based contrastive learning for sentence embeddings. *arXiv preprint arXiv:2204.10298*.

Kevin Clark, Minh-Thang Luong, Quoc V Le, and Christopher D Manning. 2020. Electra: Pre-training text encoders as discriminators rather than generators. *arXiv preprint arXiv:2003.10555*.

Tianyu Gao, Xingcheng Yao, and Danqi Chen. 2021. Simcse: Simple contrastive learning of sentence embeddings. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 6894–6910.

Raia Hadsell, Sumit Chopra, and Yann LeCun. 2006. Dimensionality reduction by learning an invariant mapping. In *2006 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'06)*, volume 2, pages 1735–1742. IEEE.

Matthew Henderson, Rami Al-Rfou, Brian Strope, Yun-Hsuan Sung, László Lukács, Ruiqi Guo, Sanjiv Kumar, Balint Miklos, and Ray Kurzweil. 2017. Efficient natural language response suggestion for smart reply. *arXiv preprint arXiv:1705.00652*.

Sergey Ioffe and Christian Szegedy. 2015. Batch normalization: Accelerating deep network training by reducing internal covariate shift. In *International conference on machine learning*, pages 448–456. PMLR.

Jacob Devlin Ming-Wei Chang Kenton and Lee Kristina Toutanova. 2019. Bert: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of NAACL-HLT*, pages 4171–4186.

Bohan Li, Hao Zhou, Junxian He, Mingxuan Wang, Yiming Yang, and Lei Li. 2020. On the sentence embeddings from pre-trained language models. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 9119–9130.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Roberta: A robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692*.

Thang Luong, Hieu Pham, and Christopher D Manning. 2015. Effective approaches to attention-based neural machine translation. In *EMNLP*.

Marco Marelli, Stefano Menini, Marco Baroni, Luisa Bentivogli, Raffaella Bernardi, and Roberto Zamparelli. 2014. A sick cure for the evaluation of compositional distributional semantic models. In *Proceedings of the Ninth International Conference on Language Resources and Evaluation (LREC'14)*, pages 216–223.

Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, Ilya Sutskever, et al. 2019. Language models are unsupervised multitask learners. *OpenAI blog*, 1(8):9.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J Liu. 2019. Exploring the limits of transfer learning with a unified text-to-text transformer. *arXiv preprint arXiv:1910.10683*.

Nils Reimers, Iryna Gurevych, Nils Reimers, Iryna Gurevych, Nandan Thakur, Nils Reimers, Johannes Daxenberger, Iryna Gurevych, Nils Reimers, Iryna Gurevych, et al. 2019. Sentence-bert: Sentence embeddings using siamese bert-networks. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing*, pages 671–688. Association for Computational Linguistics.

Iyer Shankar, Dandekar Nikhil, and Csernai Kornél. 2021. First quora dataset release: Question pairs. *Retrieved July 7th*.

Adina Williams, Nikita Nangia, and Samuel R Bowman. 2018. A broad-coverage challenge corpus for sentence understanding through inference. In *Proceedings of NAACL-HLT*, pages 1112–1122.

A Training Details

Unsupervised Models		
Models	**Batch Size**	**Learning Rate**
BERT$_{base}$(Equation 7)	64	3e-5
w/ LayerAttPooler	64	3e-5
RoBERTa$_{base}$(Equation 7)	256	1e-5
w/ LayerAttPooler	256	1e-5
RoBERTa$_{large}$(Equation 7)	256	3e-5
w/ LayerAttPooler	256	3e-5
BERT$_{base}$(DiffCSE)	64	7e-6
w/ LayerAttPooler	64	3e-5
Supervised Models		
BERT$_{base}$(Equation 6)	256	5e-5
w/ LayerAttPooler	256	1e-5
RoBERTa$_{base}$(Equation 6)	256	5e-5
w/ LayerAttPooler	256	3e-5
RoBERTa$_{large}$(Equation 6)	256	1e-5
w/ LayerAttPooler	256	5e-5
BERT$_{base}$(Equation 8)	256	5e-5
w/ LayerAttPooler	256	2e-5
RoBERTa$_{base}$(Equation 8)	256	5e-5
w/ LayerAttPooler	256	3e-5
RoBERTa$_{large}$(Equation 8)	256	1e-5
w/ LayerAttPooler	256	5e-5

Table 4: Batch sizes and learning rate for each models

Due to hardware problems, Equations 7 and 8 train at a smaller batch size than the Gao et al. (2021) paper. The GPU used in the experiment is RTX 8000, and the the hyperparameters are specified in the Table 4.

B Experiments on Different Model

DiffCSE model	
Model	**STS-B**
BERT$_{base}$(CLS$_{Last}$) (w/o BatchNorm)†	83.23
w/ LayerAttPooler(CLS$_{All}$ + AVG$_{All}$ attention) + (CLS$_{Last}$ concat)	**83.87**

Table 5: Development set results of STS-B. †: published in Chuang et al. (2022); Bold shows the highest performance among models from our reimplementation.

We experiment with whether the proposed pooling strategy is effective for a contrastive learning model with a different structure. DiffCSE model (Chuang et al., 2022) improves the performance of sentence representation by adding generator and discriminator structures of ELECTRA (Clark et al., 2020). While training, DiffCSE freezes the generator's weight and updates the sentence encoder and discriminator for sentence embedding with the contrastive learning objective. However, the discriminator is not used for inference since only representations from the sentence encoder and generator are needed. We applied our proposed pooling strategy to the sentence encoder with a contrastive learning objective. As a result, layer-wise attention pooling improves the performance of the DiffCSE model (Table 5). We use the one linear layer with the tanh activation function following SimCSE as in Equation 5, while DiffCSE uses a two-layer pooler with Batch Normalization (BatchNorm) (Ioffe and Szegedy, 2015). However, BatchNorm is not used for a fair comparison of results.

C Analysis Attention Weights over Layers

(a) Attention scores of LayerAttPooler(CLS$_{All}$ + AVG$_{All}$ attention) + (CLS$_{Last}$ concat)

(b) Attention scores of LayerAttPooler(CLS$_{All}$ + AVG$_{All}$ attention)

Figure 3: Attention scores of layer-wise pooling only (b) and concatenating the $[CLS]_{Last}$ representation of the last layer (a) on a sentence "You should do it." (short) and "People on motorcycles wearing racing gear ride around a racetrack." (long) sentences. These scores are implemented on BERT$_{base}$.

We also analyze the layer-wise attention scores depending on the length of sentences. Figure 3 (a) case explains that the last layer relatively contains more information than other layers by the $[CLS]$ token of the last layer. However, the attention score of the last layer is calculated differently for the long and short sentences. Figure 3 (b) case indicates that other layers than the last layer have substantial information for the same sentence, and the balanced attention weight per layer supports it.

SHAP-Based Explanation Methods: A Review for NLP Interpretability

Edoardo Mosca
TU Munich,
Department of Informatics,
Germany
edoardo.mosca@tum.de

Ferenc Szigeti
TU Munich,
Department of Informatics,
Germany
ferenc.szigeti@tum.de

Stella Tragianni
TU Munich,
Department of Informatics,
Germany
stella.tragianni@tum.de

Daniel Gallagher
University College Dublin,
Department of Informatics,
Ireland
daniel.gallagher1@ucdconnect.ie

Georg Groh
TU Munich,
Department of Informatics,
Germany
grohg@in.tum.de

Abstract

Model explanations are crucial for the transparent, safe, and trustworthy deployment of machine learning models. The *SHapley Additive exPlanations* (SHAP) framework is considered by many to be a gold standard for local explanations thanks to its solid theoretical background and general applicability. In the years following its publication, several variants appeared in the literature—presenting adaptations in the core assumptions and target applications. In this work, we review all relevant SHAP-based interpretability approaches available to date and provide instructive examples as well as recommendations regarding their applicability to NLP use cases.

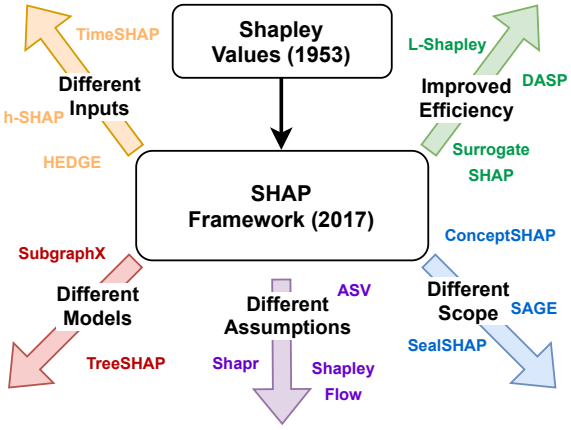

Figure 1: This work identifies five research directions pursued by Shapley- and SHAP-based approaches in XAI. Each direction, together with a few notable methods as examples, has been indicated by a different color.

1 Introduction

Several methods have been proposed to address the issue of opacity in modern machine learning models. Most notoriously, explanations are fundamental for *Deep Neural Networks* (DNNs) (Devlin et al., 2019; Madsen et al., 2021; Mosca et al., 2021) as these automatically learn millions of parameters and behave like black-boxes. Lundberg and Lee (2017) proposes *SHapley Additive exPlanations* (SHAP), a unified local-interpretability framework with a rigorous theoretical foundation on the game-theoretic concept of Shapley values (Shapley, 1953).

SHAP is nowadays considered a core contribution to the field of *eXplainable Artificial Intelligence* (XAI). Following its publication, a variety of explainability approaches based on SHAP's methodology has populated the literature and this trend continues to grow. Some present a new version of SHAP tailored to a certain type of input data—e.g. graphs (Yuan et al., 2021) and text (Chen et al., 2020)—or to specific models such as random forests (Lundberg et al., 2018). Others, instead, modify SHAP's underlying assumptions—e.g. features independence—to increase the original framework's flexibility for cases in which they are too strict or overly simplistic (Frye et al., 2019).

In this work, we **(1)** identify five broad research directions inspired by SHAP, **(2)** review available SHAP-based (or Shapley-value-based) approaches as members of such categories, and **(3)** investigate their applicability in the domain of *Natural Language Processing* (NLP).

Our work reviews 41 methods with a particular focus on their core assumptions, input require-

Proceedings of the 29th International Conference on Computational Linguistics, pages 4593–4603
October 12–17, 2022.

ments, explanation form, and available implementations. Furthermore, we provide NLP researchers with use-case-based recommendations and instructive examples.

2 Background

For the sake of clarity, we provide a gentle introduction to Shapley values and the methods for their estimation, most notably SHAP. All concepts will be explained informally, resorting to formalities when necessary.

2.1 Shapley Values

Shapley Values are a concept from game theory, originally developed as a measure to fairly distribute a reward among a set of players contributing to a certain outcome (Shapley, 1953). In the context of machine learning models, the players involved are the input features and the outcome is the model's decision, Shapley values attribute an importance score to each part of the input (Lundberg and Lee, 2017).

Given the set of input features $\mathbf{F} = \{1, 2, \ldots, p\}$, all features in a certain coalition $S \subseteq \mathbf{F}$ cooperate towards the outcome $val(S)$—with the default $val(\emptyset) = 0$. Shapley values redistribute the total outcome value $val(\mathbf{F})$ among all features based on their average marginal contribution across all possible coalitions S. More specifically, feature i's marginal contribution w.r.t. a coalition S:

$$\Delta_{val}(i, S) = val(S \cup \{i\}) - val(S)$$

is averaged across all $S \subseteq \mathbf{F} \setminus \{i\}$. Hence, the corresponding Shapley values $\phi_{val}(i)$ measures its contribution based on the formula:

$$\phi_{val}(i) = \sum_{S \subseteq \mathbf{F} \setminus \{i\}} \frac{|S|!(p - |S| - 1|)!}{p!} \Delta_{val}(i, S)$$

Here, the coefficient $\frac{|S|!(p-|S|-1|)!}{p!}$ is used as normalization term based on the number of choices for the subset S. This redistribution of the total outcome $val(\mathbf{F})$ respects the four properties of:

Efficiency: All features contributions add up to the total outcome, i.e. $\sum_{i \in \mathbf{F}} \phi_{val}(i) = val(\mathbf{F})$.

Symmetry: If $val(S \cup \{i\}) = val(S \cup \{j\})$ for all $S \subseteq \mathbf{F} \setminus \{i, j\}$, then $\phi_{val}(i) = \phi_{val}(j)$

Dummy: If $val(S \cup \{i\}) = val(S)$ for all $S \subseteq \mathbf{F}$, then $\phi_{val}(i) = 0$

Additivity: In the presence of a single game with two outcomes val_1 and val_2, then Shapley values are additive w.r.t. the combined outcome, i.e. $\phi_{val_1+val_2}(i) = \phi_{val_1}(i) + \phi_{val_2}(i)$

2.2 Shapley Values Approximation and SHAP

The idea of utilizing Shapley values to compute feature attribution scores precedes the SHAP framework (Lipovetsky and Conklin, 2001; Song et al., 2016). In this case, the outcome val of the game is the prediction of a machine learning model f and Shapley values $\phi_f(i)$ measure the influence that each feature i has based on its current value. The early literature also worked on approximation strategies, as the exponential number of coalitions renders the exact estimation of Shapley values unfeasible (Štrumbelj and Kononenko, 2014; Datta et al., 2016). The main idea from these works is to compute $\phi_f(i)$ only for a smaller selection of subsets $S \subseteq \mathbf{F}$ and to estimate the effect of removing a feature by integrating over training samples. This eliminates the need to retrain the model for each choice of S.

The work from Lundberg and Lee (2017) introduces a new perspective that unifies Shapley value estimation with popular explainability methods such as LIME (Ribeiro et al., 2016), LRP (Binder et al., 2016), and DeepLIFT (Shrikumar et al., 2017). Furthermore, they propose SHAP values as a unified measure of feature importance and prove them to be the unique solution respecting the criteria of *local accuracy*, *missingness*, and *consistency*. The authors contribute a library of methods to efficiently approximate SHAP values in a variety of settings:

KernelSHAP: Adaptation of LIME—hence model-agnostic—to approximate SHAP values. As it works for any model f, it cannot make any assumption on its structure and is thus the slowest within the framework.

LinearSHAP: Specific to linear models, uses the model's weight coefficients and optionally accounts for inter-feature correlations.

DeepSHAP: Adaptation of DeepLIFT—hence specific to neural networks–to approximate SHAP values. Considerably faster than its model-agnostic counterpart as it makes assumptions about the model's compositional nature.

While not initially presented in Lundberg and Lee (2017), the following algorithms were later

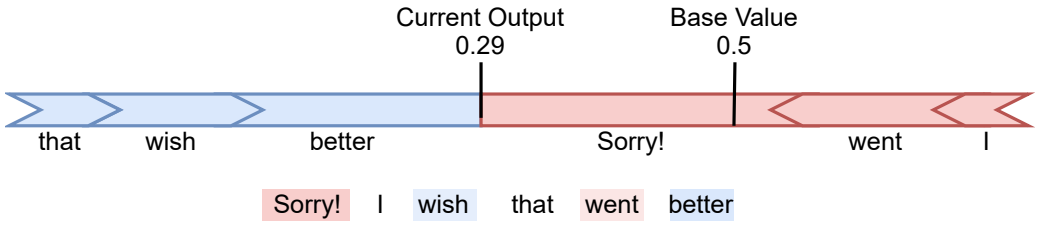

Figure 2: Example of explanation for sentiment analysis that can be generated with the SHAP library, e.g. with KernelSHAP. The base value indicates the model's average prediction. Each feature—i.e. word—contributes to the outcome, thus justifying the difference between the average and the current outcome.

added as part of the framework:

PartitionSHAP: Faster version of KernelSHAP that hierarchically clusters features. This hierarchy defines feature coalitions based on their interactions.

GradientSHAP: An extension of the *Integrated Gradients* (IG) method (Sundararajan et al., 2017)—again specific to neural networks—that aggregates gradients over the difference between the expected model output and the current output.

TreeSHAP: A fast method for computing exact SHAP values for both trees and ensembles (Lundberg et al., 2020a). In comparison to KernelSHAP, it also accounts for interactions among features.

Other minor approaches—PermutationSHAP, SamplingSHAP, ExactSHAP, and MimicSHAP—are also available in the official library[1]. To avoid confusion, we point out that the implementations have slightly different names: they use *"Explainer"* instead of *"SHAP"*. For instance, KernelSHAP and DeepSHAP are implemented with the names of *KernelExplainer* and *DeepExplainer* respectively. Figure 2 sketches an explanation generated with SHAP.

3 Search and Selection Criteria

As the popularity of SHAP increases, also the number of approaches based on it or directly on Shapley values has been on the rise. In fact, $\sim 3,200$ of the $\sim 6,900$ papers citing Lundberg and Lee (2017) are from 2021, an exponential increase when compared to previous years (1563, 567, and 118)[2].

Besides the papers already known to us, we manually screened all works citing SHAP with at least 15 citations[2]. This systematical search, based on the assumption that SHAP-based approaches should at least reference Lundberg and Lee (2017), helped us uncover several relevant contributions and mitigate the selection bias induced by our previous knowledge. The threshold of 15 citations was introduced to speed up our manual search and to filter out works that have not received the research community's attention. To account for temporal bias—i.e. that publications accumulate citations over time—we lowered the threshold to 10 for papers published in the most recent years (2021 and 2022)[2]. We only consider and review papers that contributed new SHAP-based approaches and exclude those—like (Wang, 2019) and (Antwarg et al., 2019)—utilizing SHAP (almost) off-the-shelf. Similarly, we exclude works such as Wang et al. (2020) and Huber et al. (2022) utilizing Shapley values for purposes not directly connected with explainability.

4 Existing Reviews

Previous reviews like Linardatos et al. (2021), Vilone and Longo (2020), and Madsen et al. (2021) present extensive overviews of explainability methods, but only briefly mention SHAP and a few of its derivates. Others—such as Covert et al. (2021), Sundararajan and Najmi (2020), and Kumar et al. (2020)—review some Shapley-based methods in detail (between 5 and 9) but do not construct a comprehensive review. Our work, in contrast, significantly extends this range and covers more than 40 approaches.

5 Review: SHAP-Based Approaches

Several works proposed methods based on SHAP, or more generally on Shapley values, following the contribution from Lundberg and Lee (2017). While the changes and variations introduced have been at times criticized for not being as rigorous as SHAP in following its core assumptions (Sundararajan

[1] https://github.com/slundberg/shap

[2] All queries are performed with Google Scholar. Accessed on 10.05.2022.

and Najmi, 2020), SHAP-based methods continue to increase in both quantity and popularity.

Our review categorizes SHAP-based approaches available to date based on *how they differ from* and *how they improve on* the original SHAP framework. We identify five broad categories in the existing literature, each one of them describing a different research direction pursued by its members:

(C1) Tailored to Different Input Data: This category contains approaches specialized on specific input data structures such as graphs (Wang et al., 2021), structured text (Chen et al., 2020), and images (Teneggi et al., 2021). In some cases, approaches are used complementary for applications dealing with multimodal inputs (Wich et al., 2021; Mosca et al., 2022b).

(C2) Explaining Different Models: Methods in this class are specifically designed to explain predictions from particular types of machine learning models such as random forests (Lundberg et al., 2018; Labreuche and Fossier, 2018) and neural networks (Ghorbani and Zou, 2021). Hence, these are model-specific.

(C3) Modifying Core Assumptions: SHAP treats features as independent. Newer methods offer the possibility to account for dependencies between features (Frye et al., 2019) and for causal structures behind their interactions (Heskes et al., 2020).

(C4) Producing Different Explanations Types: SHAP is a framework for local feature-attribution explanations, i.e. it attributes scores to input components based on their instance-level contributions. Methods in this category have a different scope and generate explanations that convey a different type of information. This can vary from global explanations (Covert et al., 2020) to counterfactual explanations (Singal et al., 2019) and concept explanations (Yeh et al., 2020).

(C5) Estimating Shapley Values More Efficiently: These approaches comprise alternative strategies for the approximation of Shapley values. Their focus is on leveraging prior knowledge about the data and model to improve the approximation *efficiency* and *accuracy* (Messalas et al., 2019; Chen et al., 2018).

Clearly, these categories are not designed to be exclusive. Therefore, an approach can fall in more than one if it differs from SHAP in multiple aspects. Table 1 provides an overview of all approaches with their main characteristics. As one can observe, the majority of approaches are identified as part of more categories, i.e. research directions.

5.1 Approaches Tailored to Different Inputs

SHAP does not make strong assumptions on the target model's input. While this suggests that it is suitable for all input types, its lack of specificity results in limitations when applied directly to different inputs than tabular data.

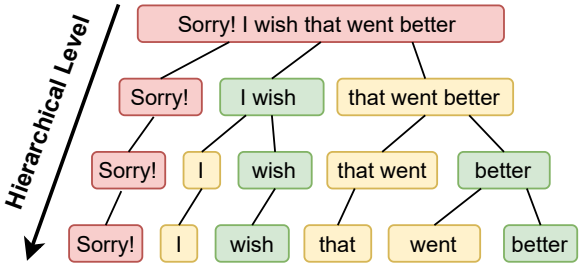

Figure 3: Example of hierarchical explanation that can be generated with HEDGE (Chen et al., 2020) for a sentiment analysis model. Each token is colored by contribution: negative (red), neutral (yellow), and positive (green). Going one level lower represents a token-breakdown step and thus more fine-grained Shapley values.

For text data, only measuring each individual feature's effect is an oversimplification, as words present strong interactions and their meaning and contribution heavily rely on the context. Thus, when it comes to text data, only considering single words as features is quite restrictive and relevance scores should be applied to multi-level tokens or even to entire sentences. *Hierarchical Explanation via Divisive GEneration* (HEDGE) (Chen et al., 2020) is an example of a SHAP-based method addressing this issue for (long) texts. Based on the weakest token interactions, it iteratively divides the text into shorter phrases and words in a top-down fashion. At each level, a relevance score is attributed to each token, resulting in a hierarchical explanation (Chen et al., 2020). PartitionSHAP, recently added to the official SHAP repository[3], follows a similar strategy by creating hierarchical features coalitions and measuring their interactions.

[3]https://github.com/slundberg/shap

Method	Categories	Description	NLP Applicability / Implementation
SHAP (Lundberg and Lee, 2017)		The original SHAP framework including the methods: KernelSHAP, LinearSHAP, DeepSHAP, etc.	*Ready Off-the-Shelf* Python
AVA (Bhatt et al., 2020)	**(C5)**	Combines the explanations of nearest neighbors to explain a given instance	*Adaptable* n.a.
ASV (Frye et al., 2019)	**(C1) (C3)**	Relaxes the symmetry axiom of Shapley values to incorporate causal structure into explanations	*Potentially Applicable* R
BShap (Sundararajan and Najmi, 2020)	**(C4) (C5)**	Baseline approach to facilitate comparison between different Shapley value based methods	*Adaptable* n.a.
C- and L-Shapley (Chen et al., 2018)	**(C3) (C5)**	Efficient feature attribution method that models data as a graph by considering only neighboring features	*Ready Off-the-Shelf* TensorFlow
CASV (Singal et al., 2019)	**(C1) (C2) (C3) (C4)**	Shapley value adaptation to account for counterfactuals by adhering to the Rubin Causal Model	*Not Relevant* n.a.
Causal Shapley (Heskes et al., 2020)	**(C1) (C3)**	Computing feature importance on data with (partial) causal ordering using Pearl's do-calculus	*Potentially Applicable* R
ConceptSHAP (Yeh et al., 2020)	**(C4)**	Unsupervised discover of concepts inherent to the data and model based on Shapley values	*Ready Off-the-Shelf* PyTorch
DASP (Ancona et al., 2019)	**(C3) (C5)**	Polynomial-time approximation of Shapley values in DNNs	*Adaptable* TensorFlow
Data Shapley (Ghorbani and Zou, 2019)	**(C4)**	Shapley-based importance attribution method for individual data instances in the training set	*Potentially Applicable* TensorFlow
DeepSHAP v2 (Chen et al., 2021)	**(C2) (C5)**	Computes efficiently SHAP values for DNNs with an extension to explain stacks of mixed model types	*Adaptable* n.a.
GrammarSHAP (Mosca et al., 2022a)	**(C1) (C3)**	Hierarchical explanations for text inputs based on the sentence grammatical structure	*Adaptable* n.a.
gSHAP (Tan et al., 2018)	**(C4)**	Generates intuitive Shapley-based global by aggregating local explanations	*Potentially Applicable* n.a.
h-SHAP (Teneggi et al., 2021)	**(C1) (C5)**	Hierarchical implementation of Shapley values for their efficient computation in image data	*Potentially Applicable* PyTorch
HEDGE (Chen et al., 2020)	**(C1) (C3)**	Hierarchical explanations based on feature interaction detection specifically for text data	*Ready Off-the-Shelf* PyTorch
Integrated Hessians (Janizek et al., 2021)	**(C5)**	Extension of Integrated Gradients to explain pairwise feature interactions in NNs	*Ready Off-the-Shelf* PyTorch
lossSHAP (Lundberg et al., 2020b)	**(C2) (C4)**	Obtain global explanations by aggregating local explanations with TreeSHAP	*Potentially Applicable* Python
MCDA Explainer (Labreuche and Fossier, 2018)	**(C1) (C2) (C3)**	Proposes the *influence index*, which is an extension of Shapley values for MCDA tree models	*Not Relevant* n.a.
Neuron Shapley (Ghorbani and Zou, 2021)	**(C2) (C4)**	Quantifies the contributions of single neurons to single predictions and overall model performance	*Adaptable* TensorFlow
R2 decomposition (Redell, 2019)	**(C5)**	Feature importance attribution based on Shapley value variance decomposition	*Potentially Applicable* R
Shapley Flow (Wang et al., 2021)	**(C1) (C3)**	Enables the addition of a causal graph encoding relationships among input features	*Potentially Applicable* Python
SAGE (Covert et al., 2020)	**(C4) (C5)**	Efficiently quantifies each feature's contribution to the model's performance for global explainability	*Potentially Applicable* Python
SealSHAP (Parvez and Chang, 2021)	**(C4)**	Shapley-based usefulness measure of individual data sources for transfer learning	*Ready Off-the-Shelf* TensorFlow
Shap-C (Ramon et al., 2019)	**(C4) (C5)**	Combination of computing counterfactuals and Shapley Values	*Potentially Applicable* Python
Shapley Residuals (Kumar et al., 2021)	**(C4)**	Captures information lost by KernelSHAP in Shapley Residuals, which characterize feature dependence	*Potentially Applicable* n.a.
Shapley Taylor index (Dhamdhere et al., 2020)	**(C3) (C5)**	Generalization of the Shapley value that attributes the model's prediction to interactions of subsets of features	*Potentially Applicable* n.a.
Shapr (Aas et al., 2021)	**(C3)**	Extends KernelSHAP to handle data with dependent features and produce more realistic explanations	*Potentially Applicable* R
SPVIM (Williamson and Feng, 2020)	**(C4) (C5)**	Global variable importance measure using an efficient regression-based Shapley value estimator	*Not Relevant* Python and R
SubgraphX (Yuan et al., 2021)	**(C1) (C2) (C5)**	Explain GNNs by identifying important subgraphs using Shapley values as importance measures	*Not Relevant* PyTorch
SurrogateSHAP (Messalas et al., 2019)	**(C5)**	An XGBoost tree model is trained as a surrogate model on the target model and TreeSHAP is applied to explain it	*Potentially Applicable* n.a.
TreeSHAP (Lundberg et al., 2018)	**(C2) (C5)**	Fast and exact method to estimate SHAP values for tree models and ensembles of trees	*Potentially Applicable* Python
TimeSHAP (Bento et al., 2021)	**(C1) (C2) (C4)**	Adapts KernelSHAP to sequential data and produces feature, event and cell-wise explanations	*Potentially Applicable* n.a.

Table 1: Overview of available Shapley- and SHAP-based methods. For each method we also indicate the categories it belongs to, its main idea and intuition, and its applicability to NLP together with the available implementations. See 6.1 for more details about our NLP-applicability assessment.

Figure 3 sketches an example of a hierarchical explanation for text data.

For models trained on graph data, especially graph DNNs, Yuan et al. (2021) proposed to explain predictions by using Shapley values as a measure of subgraph importance. The resulting method—named SubgraphX—also captures the interactions between different subgraphs.

On images, SHAP can face computational limitations as the number of features, i.e. pixels, can become extremely large. h-SHAP (Teneggi et al., 2021) efficiently retrieves exact Shapley values by hierarchically excluding irrelevant image areas from the computation. This is done following the observation that, if a certain area in the image is uninformative, so are its constituent sub-areas, which are therefore not worth exploring.

5.2 Approaches Explaining Different Models

Explanation methods making fewer assumptions on the target classifier benefit from better applicability as they can explain a wider range of models. However, this can hinder explanations in terms of accuracy, information granularity, and computational efficiency. As we have already seen in 2.2: KernelSHAP has the key advantage of being model-agnostic, but it is drastically more inefficient than its DNN-specific counterpart DeepSHAP (Lundberg and Lee, 2017).

An example of a highly-specialized explainability method is TreeSHAP, presented by Lundberg et al. (2018) as an extension of the SHAP framework. This approach, only applicable to decision trees or ensembles thereof, is a highly efficient algorithm for exact SHAP values retrieval. Not only the approach needs considerably less computational effort than the more general variants such as KernelSHAP, but it leverages the decision tree structure to compute SHAP interaction values and thus captures pairwise interactions between features.

Ghorbani and Zou (2021) proposes *Neuron Shapley*, a framework targeting DNN models which is able to quantify each individual neuron's contribution to single predictions and overall model performance. An example of the kind of explanation enabled by Neuron Shapley is visualized in figure 4. By analyzing interactions between neurons and picking those which exhibit the largest Shapley value, this method is particularly suitable for identifying neurons responsible for biases and vulnerabilities (Ghorbani and Zou, 2021).

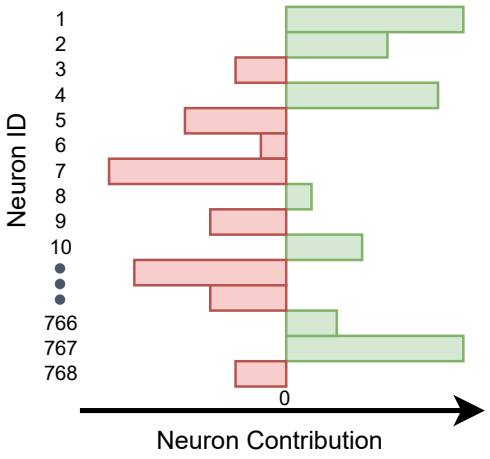

Figure 4: Sketch of a Neuron Shapley explanation for the 768 neurons of BERT output layer (Devlin et al., 2019). A Shapley value is assigned to each neuron depending depending on how they contribute towards the prediction (green) or against it (red).

5.3 Approaches Modifying Core Assumptions

Assumptions made by SHAP can be at times too restrictive or simplistic, which can prevent explanations from accessing and leveraging crucial information such as dependency relationships between input features. For instance, already the symmetry property of Shapley values treats features as independent. While this can be true in some cases, for instance when dealing with tabular data with uncorrelated variables, it is an oversimplification when it comes to texts, images, and more structured data.

Frye et al. (2019) introduces *Asymmetric Shapley Values* (ASV), which drops the symmetry assumption and enables the generation of model-agnostic explanations incorporating any causal dependency known to be present in the data. Similar approaches are:

- *Causal Shapley* (Heskes et al., 2020), additionally requiring a partial causal ordering of the features as input.
- *Shapley Flow* (Wang et al., 2021), which leverages a causal graph, encoding relationships among input features.
- *Shapr* (Aas et al., 2021), an extension of KernelSHAP relaxing the feature independence assumption.

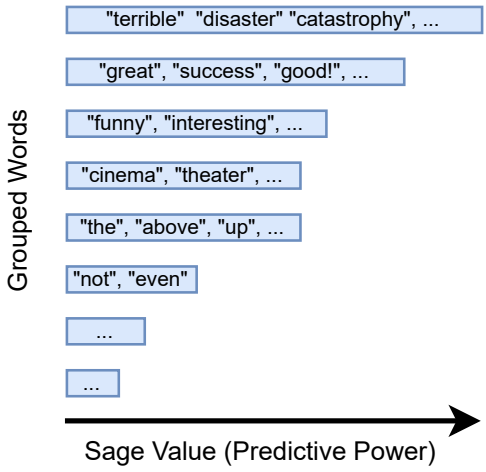

Figure 5: Example of SAGE explanation for a sentiment analysis model. Since the number of global features is as large as the vocabulary, words need to be grouped together (e.g. by similarity) to reduce the number of features to be explained.

5.4 Approaches Producing Different Explanation Types

The SHAP framework and many of its derivatives mainly focus on generating local explanations based on feature importance. However, the general applicability of Shapley values combined with its strong foundations also offers potential for different explainability settings. More recent works have explored the usage of Shapley values to build other types of explanations conveying different kinds of information about the model and the available data.

For instance, *Data Shapley* (Ghorbani and Zou, 2019) estimates the importance of each training sample for a given machine learning model. Similarly, SealSHAP (Parvez and Chang, 2021) attributes usefulness scores to data sources for transfer learning.

Covert et al. (2020) introduces *Shapley Additive Global importancE* (SAGE), an explainability method analogous to SHAP but with a core focus on global explainability. More in detail, SAGE is a model-agnostic method that quantifies the predictive power of each input feature for a given model while also accounting for their interactions. An instructive example for NLP is shown in figure 5.

Alongside local and global explainability, works like Yeh et al. (2020) adapt the notion of Shapley values for concept analysis (Sajjad et al., 2021). Given a set of concepts extracted from a model, the authors define the notion of *completeness* as a measure to indicate how sufficient such concepts are in explaining the model's predictive behavior. Furthermore, they propose ConceptSHAP, an unsupervised approach able to automatically retrieve a set of interpretable concepts without needing to know them in advance.

5.5 Approaches Proposed for Estimation Efficiency

While Shapley values convey useful information about the importance or contribution of a certain input component, their computation quickly becomes infeasible as coalitions grow exponentially w.r.t. input size. The SHAP framework already addresses this issue by providing more efficient estimation techniques. Nevertheless, later works continued to explore improvements to further decrease the computational effort necessary to produce meaningful explanations.

Chen et al. (2018) leverage features dependencies in image and text data to build two efficient algorithms, *L-Shapley* and *C-Shapley*, for Shapley values estimation. Their methods only consider a subset of the possible coalitions based on the data's underlying graph structure, which connects for instance adjacent words and pixels in texts and images respectively.

SurrogateSHAP (Messalas et al., 2019), instead, trains an XGBoost tree as a surrogate for the original model. The surrogate is then used to generate SHAP explanations, which considerably reduces the computational cost compared to directly applying SHAP to the original (more complex) model.

6 Relevance for NLP Research

Large and complex neural NLP models—such as BERT (Devlin et al., 2019) and GPT-3 (Brown et al., 2020)—are used extensively in research and industry. The trend is justified by the strong correlation between models' size and their performance (Madsen et al., 2021; Brown et al., 2020). Naturally, increasing model complexity causes a higher demand for NLP explainability. In this section, we match this demand to the reviewed SHAP-based methods and provide researchers with use-case-based recommendations.

6.1 Applicability of the Approaches

In table 1 (rightmost column), we also evaluate each SHAP-based explainability approach based on its applicability to neural NLP models. In this regard, our assessment considers *availability of*

implementations, *suitability for text data*, and *conceptual complexity* as relevant factors. We organize all reviewed approaches into four tiers:

- *Ready Off-the-Shelf*: The code is available and is ready to be used as-is.
- *Adaptable*: The code is available and there are straightforward steps for its adaptation to NLP use cases. Alternatively, no code is available but there are clear instructions for an ad-hoc implementation for the NLP domain.
- *Potentially Applicable*: Strong assumptions and substantial implementation work are required to apply the method to NLP.
- *Not Relevant*: The method is only applicable to other domains and it does not provide any apparent value for explaining NLP models.

6.2 Recommendations for NLP Use Cases

To build feature attribution explanations, HEDGE (Chen et al., 2020) is arguably the most suitable choice, as hierarchical explanations can contain more information than their non-hierarchical counterpart, e.g. generated with SHAP. The strength of HEDGE becomes even more apparent when dealing with long texts, where sentence structure is of major relevance for the model to be explained. *L-Shapley*, *C-Shapley* (Chen et al., 2018) and PartitionSHAP can also be considered where hierarchical explanations are not necessary and very computationally efficient methods are required instead.

For model debugging, Neuron Shapley is suitable to identify neurons that are responsible for unintended biases or that are particularly vulnerable to adversarial attacks (Ghorbani and Zou, 2021). Pruning these neurons can be an effective method of alleviating such model defects (Ghorbani and Zou, 2021). To gain a global understanding of what the model has learned in practice, SAGE (Covert et al., 2020) combined with word grouping provides a summary of the features—e.g. words—that are most relevant for the model's performance. In this case, pruning irrelevant features can be also tested to improve model accuracy. A similar summary can be provided by ConceptSHAP (Yeh et al., 2020), which can compile a comprehensive list of the concepts identified by the model in an unsupervised fashion. Furthermore, ConceptSHAP can be used to determine the amount of model variance covered by the whole set of identified concepts (Yeh et al., 2020).

If causal structures or dependencies present in the text are known and can be explicitly modeled, then methods such as ASV (Frye et al., 2019), Shapley Flow (Wang et al., 2021), and Causal Shapley (Heskes et al., 2020) can leverage such information. For use cases involving graphs as part of multi-modal inputs—e.g. modeling a social network (Wich et al., 2021)—any of the previous methods can be combined with SubGraphX (Yuan et al., 2021) to also produce explanations for the graph component of the input.

When it comes to *sequence-to-sequence* tasks such as question answering and machine translation, the usage of SHAP-based methods has not been explored in depth. With a few exceptions[4], available approaches seem particularly tailored only to classification settings. We believe this is a strong limitation and we encourage the reader to look for alternatives.

7 Criticisms

The usage of Shapley values for generating model explanations has also been criticized. For instance, Kumar et al. (2020) shows that using Shapley values for feature importance leads to mathematical inconsistencies which can only be mitigated by introducing further complexity like causality assumptions. Moreover, the authors argue that Shapley values do not represent an intuitive solution to the human-centric goals of model explanations and thus are only suitable in a limited range of settings.

Sundararajan and Najmi (2020), on the other hand, criticize some Shapley-value-based methods. In fact, while a strong case for utilizing Shapley values can be made thanks to their uniqueness result in satisfying certain properties (see 2.1), often methods employing them operate under different assumptions and hence the uniqueness results loses validity in their context.

Merrick and Taly (2020) argues that existing SHAP-based literature focuses on the axiomatic foundation of Shapley values and their efficient estimation but neglects the uncertainty of the explanations produced. The authors illustrate how small differences in the underlying game formulation can lead to sudden leaps in Shapley values and can attribute a positive contribution to features that do not play any role in the machine learning model.

[4] https://shap.readthedocs.io/en/latest/text_examples.html

8 Conclusion

SHAP is a core contribution to explainable artificial intelligence and one of the most popular frameworks for local interpretability. A considerable amount of recent works has proposed SHAP-based approaches, which we identify as part of five different yet overlapping research directions. In particular, the recent literature has worked towards **(C1)** *tailoring explanations to different input data*, **(C2)** *explaining specific models*, **(C3)** *improving the framework's flexibility via modifying core assumptions*, **(C4)** *producing different explanation types*, and **(C5)** *estimating Shapley values more efficiently*.

This work has reviewed a total of 41 approaches and has organized them based on the introduced categories. As expected, given the overlapping nature of the classification, the majority of existing methods fall into multiple categories and have therefore each made distinct contributions to the field. While most of them are not directly applicable to NLP settings, we identified a few that can be beneficial for current practitioners. Furthermore, we have compiled a list of recommendations for each NLP use case. We also observe a severe limitation of SHAP-based methods in terms of applicability to sequence-to-sequence NLP tasks.

We hope our work provides NLP/XAI practitioners and newcomers with a comprehensive overview of SHAP-based approaches, with references to stimulate further investigation and future advances in academic and industrial research.

Acknowledgments

This paper has been supported by the German *Federal Ministry of Education and Research* (BMBF, grant 01IS17049).

References

Kjersti Aas, Martin Jullum, and Anders Løland. 2021. Explaining individual predictions when features are dependent: More accurate approximations to shapley values. *Artificial Intelligence*, 298:103502.

Marco Ancona, Cengiz Oztireli, and Markus Gross. 2019. Explaining deep neural networks with a polynomial time algorithm for shapley value approximation. In *International Conference on Machine Learning*, pages 272–281. PMLR.

Liat Antwarg, Ronnie Mindlin Miller, Bracha Shapira, and Lior Rokach. 2019. Explaining anomalies detected by autoencoders using shap. *arXiv preprint arXiv:1903.02407*.

Joao Bento, Pedro Saleiro, Andre Cruz, Mario Figueiredo, and Pedro Bizarro. 2021. Timeshap: Explaining recurrent models through sequence perturbations. *KDD*.

Umang Bhatt, Adrian Weller, and José MF Moura. 2020. Evaluating and aggregating feature-based model explanations. *arXiv preprint arXiv:2005.00631*.

Alexander Binder, Sebastian Bach, Gregoire Montavon, Klaus-Robert Müller, and Wojciech Samek. 2016. Layer-wise relevance propagation for deep neural network architectures. In *Information science and applications (ICISA) 2016*, pages 913–922. Springer.

Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel Ziegler, Jeffrey Wu, Clemens Winter, Chris Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. 2020. Language models are few-shot learners. In *Advances in Neural Information Processing Systems*, volume 33, pages 1877–1901. Curran Associates, Inc.

Hanjie Chen, Guangtao Zheng, and Yangfeng Ji. 2020. Generating hierarchical explanations on text classification via feature interaction detection. *arXiv preprint arXiv:2004.02015*.

Hugh Chen, Scott Lundberg, and Su-In Lee. 2021. Explaining models by propagating shapley values of local components. In *Explainable AI in Healthcare and Medicine*, pages 261–270. Springer.

Jianbo Chen, Le Song, Martin J Wainwright, and Michael I Jordan. 2018. L-shapley and c-shapley: Efficient model interpretation for structured data. *arXiv preprint arXiv:1808.02610*.

Ian Covert, Scott Lundberg, and Su-In Lee. 2021. Explaining by removing: A unified framework for model explanation. *Journal of Machine Learning Research*, 22(209):1–90.

Ian Covert, Scott M Lundberg, and Su-In Lee. 2020. Understanding global feature contributions with additive importance measures. *Advances in Neural Information Processing Systems*, 33:17212–17223.

Anupam Datta, Shayak Sen, and Yair Zick. 2016. Algorithmic transparency via quantitative input influence: Theory and experiments with learning systems. In *2016 IEEE symposium on security and privacy (SP)*, pages 598–617.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. Bert: Pre-training of deep bidirectional transformers for language understanding. In *NAACL-HLT (1)*.

Kedar Dhamdhere, Ashish Agarwal, and Mukund Sundararajan. 2020. The shapley taylor interaction index. *PMLR*.

Christopher Frye, Colin Rowat, and Ilya Feige. 2019. Asymmetric shapley values: incorporating causal knowledge into model-agnostic explainability. *NeurIPS 2020*.

Amirata Ghorbani and James Zou. 2019. Data shapley: Equitable valuation of data for machine learning. In *International Conference on Machine Learning*, pages 2242–2251. PMLR.

Amirata Ghorbani and James Zou. 2021. Neuron shapley: Discovering the responsible neurons. *NeurIPS 2021*.

Tom Heskes, Evi Sijben, Ioan Gabriel Bucur, and Tom Claassen. 2020. Causal shapley values: Exploiting causal knowledge to explain individual predictions of complex models. *NeurIPS 2020*.

Lukas Huber, Marc Alexander Kühn, Edoardo Mosca, and Georg Groh. 2022. Detecting word-level adversarial text attacks via SHapley additive exPlanations. In *Proceedings of the 7th Workshop on Representation Learning for NLP*, pages 156–166, Dublin, Ireland. Association for Computational Linguistics.

Joseph Janizek, Pascal Sturmfels, and Su-In Lee. 2021. Explaining explanations: Axiomatic feature interactions for deep networks. *JMLR*.

I Elizabeth Kumar, Suresh Venkatasubramanian, Carlos Scheidegger, and Sorelle Friedler. 2020. Problems with shapley-value-based explanations as feature importance measures. In *International Conference on Machine Learning*, pages 5491–5500. PMLR.

Indra Kumar, Carlos Scheidegger, Suresh Venkatasubramanian, and Sorelle Friedler. 2021. Shapley residuals: Quantifying the limits of the shapley value for explanations. *NeurIPS*.

Christophe Labreuche and Simon Fossier. 2018. Explaining multi-criteria decision aiding models with an extended shapley value. In *IJCAI*, pages 331–339.

Pantelis Linardatos, Vasilis Papastefanopoulos, and Sotiris Kotsiantis. 2021. Explainable ai: A review of machine learning interpretability methods. *Entropy*, 23(1):18.

Stan Lipovetsky and Michael Conklin. 2001. Analysis of regression in game theory approach. *Applied Stochastic Models in Business and Industry*, 17(4):319–330.

Scott M. Lundberg, Gabriel Erion, Hugh Chen, Alex DeGrave, Jordan M. Prutkin, Bala Nair, Ronit Katz, Jonathan Himmelfarb, Nisha Bansal, and Su-In Lee. 2020a. From local explanations to global understanding with explainable ai for trees. *Nature Machine Intelligence*, 2(1):2522–5839.

Scott M Lundberg, Gabriel Erion, Hugh Chen, Alex DeGrave, Jordan M Prutkin, Bala Nair, Ronit Katz, Jonathan Himmelfarb, Nisha Bansal, and Su-In Lee. 2020b. From local explanations to global understanding with explainable ai for trees. *Nature machine intelligence*, 2(1):56–67.

Scott M Lundberg, Gabriel G Erion, and Su-In Lee. 2018. Consistent individualized feature attribution for tree ensembles. *arXiv preprint arXiv:1802.03888*.

Scott M. Lundberg and Su-In Lee. 2017. A unified approach to interpreting model predictions. *NeurIPS 2017*.

Andreas Madsen, Siva Reddy, and Sarath Chandar. 2021. Post-hoc interpretability for neural nlp: A survey. *arXiv preprint arXiv:2108.04840*.

Luke Merrick and Ankur Taly. 2020. The explanation game: Explaining machine learning models using shapley values. In *International Cross-Domain Conference for Machine Learning and Knowledge Extraction*, pages 17–38. Springer.

Andreas Messalas, Yiannis Kanellopoulos, and Christos Makris. 2019. Model-agnostic interpretability with shapley values. In *2019 10th International Conference on Information, Intelligence, Systems and Applications (IISA)*, pages 1–7. IEEE.

Edoardo Mosca, Defne Demirtürk, Luca Mülln, Fabio Raffagnato, and Georg Groh. 2022a. GrammarSHAP: An efficient model-agnostic and structure-aware NLP explainer. In *Proceedings of the First Workshop on Learning with Natural Language Supervision*, pages 10–16, Dublin, Ireland. Association for Computational Linguistics.

Edoardo Mosca, Katharina Harmann, Tobias Eder, and Georg Groh. 2022b. Explaining neural NLP models for the joint analysis of open-and-closed-ended survey answers. In *Proceedings of the 2nd Workshop on Trustworthy Natural Language Processing (TrustNLP 2022)*, pages 49–63, Seattle, U.S.A. Association for Computational Linguistics.

Edoardo Mosca, Maximilian Wich, and Georg Groh. 2021. Understanding and interpreting the impact of user context in hate speech detection. In *Proceedings of the Ninth International Workshop on Natural Language Processing for Social Media*, pages 91–102.

Md Rizwan Parvez and Kai-Wei Chang. 2021. Evaluating the values of sources in transfer learning. *NAACL 2021*.

Yanou Ramon, David Martens, Foster Provost, and Theodoros Evgeniou. 2019. Counterfactual explanation algorithms for behavioral and textual data. *arXiv preprint arXiv:1912.01819*.

Nickalus Redell. 2019. Shapley decomposition of r-squared in machine learning models. *arXiv preprint arXiv:1908.09718*.

Marco Tulio Ribeiro, Sameer Singh, and Carlos Guestrin. 2016. " why should i trust you?" explaining the predictions of any classifier. In *Proceedings of the 22nd ACM SIGKDD international conference on knowledge discovery and data mining*, pages 1135–1144.

Hassan Sajjad, Narine Kokhlikyan, Fahim Dalvi, and Nadir Durrani. 2021. Fine-grained interpretation and causation analysis in deep nlp models. *arXiv preprint arXiv:2105.08039*.

Lloyd S Shapley. 1953. A value for n-person games. *Contributions to the Theory of Games 2.28*, page 307–317.

Avanti Shrikumar, Peyton Greenside, and Anshul Kundaje. 2017. Learning important features through propagating activation differences. In *International Conference on Machine Learning*, pages 3145–3153. PMLR.

Raghav Singal, Omar Besbes, Antoine Desir, Vineet Goyal, and Garud Iyengar. 2019. Shapley meets uniform: An axiomatic framework for attribution in online advertising. In *The World Wide Web Conference*, pages 1713–1723.

Eunhye Song, Barry L Nelson, and Jeremy Staum. 2016. Shapley effects for global sensitivity analysis: Theory and computation. *SIAM/ASA Journal on Uncertainty Quantification*, 4(1):1060–1083.

Erik Štrumbelj and Igor Kononenko. 2014. Explaining prediction models and individual predictions with feature contributions. *Knowledge and information systems*, 41(3):647–665.

Mukund Sundararajan and Amir Najmi. 2020. The many shapley values for model explanation. In *International Conference on Machine Learning*, pages 9269–9278. PMLR.

Mukund Sundararajan, Ankur Taly, and Qiqi Yan. 2017. Axiomatic attribution for deep networks. In *International Conference on Machine Learning*, pages 3319–3328. PMLR.

Sarah Tan, Giles Hooker, Paul Koch, Albert Gordo, and Rich Caruana. 2018. Considerations when learning additive explanations for black-box models. *arXiv preprint arXiv:1801.08640 3*.

Jacopo Teneggi, Alexandre Luster, and Jeremias Sulam. 2021. Fast hierarchical games for image explanations. *arXiv preprint arXiv:2104.06164*.

Giulia Vilone and Luca Longo. 2020. Explainable artificial intelligence: a systematic review. *arXiv preprint arXiv:2006.00093*.

Guan Wang. 2019. Interpret federated learning with shapley values. *arXiv preprint arXiv:1905.04519*.

Jianhong Wang, Yuan Zhang, Tae-Kyun Kim, and Yunjie Gu. 2020. Shapley q-value: A local reward approach to solve global reward games. *AAAI*, 34:7285–7292.

Jiaxuan Wang, Jenna Wiens, and Scott Lundberg. 2021. Shapley flow: A graph-based approach to interpreting model predictions. *AISTATS 2021*.

Maximilian Wich, Edoardo Mosca, Adrian Gorniak, Johannes Hingerl, and Georg Groh. 2021. Explainable abusive language classification leveraging user and network data. In *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*, pages 481–496. Springer.

Brian Williamson and Jean Feng. 2020. Efficient nonparametric statistical inference on population feature importance using shapley values. In *International Conference on Machine Learning*, pages 10282–10291. PMLR.

Chih-Kuan Yeh, Been Kim, Sercan Arik, Chun-Liang Li, Tomas Pfister, and Pradeep Ravikumar. 2020. On completeness-aware concept-based explanations in deep neural networks. *Advances in Neural Information Processing Systems*, 33.

Hao Yuan, Haiyang Yu, Jie Wang, Kang Li, and Shuiwang Ji. 2021. On explainability of graph neural networks via subgraph explorations. *arXiv preprint arXiv:2102.05152*.

A simple log-based loss function for ordinal text classification

François Castagnos, Martin Mihelich, Charles Dognin
Glanceable
{francois, martin, charles}@glanceable.io

Abstract

The cross-entropy loss function is widely used and generally considered the default loss function for text classification. When it comes to ordinal text classification where there is an ordinal relationship between labels, the cross-entropy is not optimal as it does not incorporate the ordinal character into its feedback. In this paper, we propose a new simple loss function called ordinal log-loss (OLL). We show that this loss function outperforms state-of-the-art previously introduced losses on four benchmark text classification datasets.

1 Introduction

For many classification tasks, there is an order on the labels of the target variable. In particular, in natural language processing (NLP) when, for example, we are trying to predict the number of stars associated with a review: it is obvious that when the label is 1 star, predicting 2 stars is better than predicting 5 stars. This type of classification is called ordinal classification (or ordinal regression) and many techniques have been developed in recent years around it. Among the most used techniques, the **ordinal binary classification** consists in decomposing the ordered target variable in several binary ones (Frank and Hall, 2001; Allwein et al., 2000). The **threshold methods** treat the target variable (with N classes) as a continuous real-valued variable and $N-1$ thresholds are introduced (Herbrich et al., 2000; Verwaeren et al., 2012; Cao et al., 2020). In the **loss-sensitive classification** the loss function is built such that a higher penalty is assigned if the distance between the prediction and the label is higher. Several losses can be used here: mapping the labels $\{C_1; C_2; ...; C_N\}$ into values $\{1; 2; ...; N\}$ and use the mean squared error. The margin loss or the hinge loss can also be extended for ordinal regression (Rennie and Srebro, 2005). The weighted kappa loss (de la Torre et al., 2018), the earth mover's distance (Hou et al., 2016), the soft labels (Diaz and Marathe, 2019; Bertinetto et al., 2020) or the CORAL method (Cao et al., 2020) are other examples of recent modified losses introduced in ordinal classification problems.

In order to measure the performance of the ordinal regression there are well known metrics such as the off-by-k-accuracy, the mean absolute error, the mean squared error or Kendall Tau for instance (Cardoso and Sousa, 2011; Gaudette and Japkowicz, 2009).

1.1 Specific contribution

The main contribution of this paper is to introduce a new loss named ordinal log-loss (OLL). This loss is easy to use, adapted to ordinal classification and gives more accurate results than classical existing methods in text classification. The idea behind the OLL is to penalize bad predictions instead of rewarding good predictions like the majority of the losses mentioned before do.

In section 2 we introduce the ordinal log-loss. In section 3 we present the experiments, the metrics used and finally the results.

2 Ordinal Log-Loss

2.1 Definition

As explained in the introduction, in ordinal classification tasks, predictions too distant from the labels can be particularly problematic. While most of the losses introduced in the literature for ordinal classification (Gutiérrez et al., 2015; Bertinetto et al., 2020; Rennie and Srebro, 2005) tend to encourage predictions close to the labels, we introduce a loss which penalises the critical errors (i.e. the predictions that are the most distant from the correct class).
First, for each ordinal classification task, we define a distance matrix that embodies the distances between each label:

$$D = (d(C_i, C_j))_{(i,j)\in[\![1,N]\!]^2} \quad (1)$$

Proceedings of the 29th International Conference on Computational Linguistics, pages 4604–4609
October 12–17, 2022.

where N is the number of classes, $C = (C_1, ..., C_N)$ are the different classes and $d(C_i, C_j)$ the distance between label C_i and C_j. We denote for the sake of simplicity $d(i, j)$ for $d(C_i, C_j)$ and y for C_y (the label).
Let $P = (p_1, ..., p_N)$ be the output probability distribution of a network for a given prediction. By definition, the cross-entropy loss encourages the models to output a high probability for the correct class.

Equivalently, but from the opposite perspective, we wish that the further a prediction is from the true label, the higher the loss should be. With a simple modification of the cross-entropy loss, we can find such a loss, that we introduce as the ordinal log-loss (OLL):

$$\mathcal{L}_{OLL-\alpha}(P, y) = -\sum_{i=1}^{N} \log(1 - p_i) d(y, i)^{\alpha} \quad (2)$$

where α is a strictly positive hyper-parameter. The novelty of this loss lies in the coefficients $-\log(1-p_i)$. In fact, other articles already considered the following loss: $\sum_{i=1}^{N} p_i d(y, i)$ (obtained by replacing $-\log(1 - p_i)$ by p_i and where α is taken equal to 1) (Hou et al., 2016; Kotsiantis and Pintelas, 2004). Nevertheless, for this latter loss, as explained in (Hou et al., 2016), the optimization does not converge to a desired local minimum. Although we have not reported these results in this article, this is indeed what we observed experimentally and what gave us the idea of the OLL. We wanted to penalize classification errors more strongly and since we have the inequality $-\log(1 - p_i) \geq p_i$ for all $p_i \in [0, 1[$, the weights in front of $d(y, i)$ are more important in the OLL loss which implies a greater penalty.

2.2 Impact of the α parameter

In the expression 2, α is an hyper-parameter that could be interpreted as a penalizing factor: the greater α is, the higher the loss function is when the distance between the output predictions and the labels is high.

3 Experiments and Results

In this section we first introduce the public datasets (section 3.1) and the metrics (section 3.2) used to compare our loss function to existing ones. Then in section 3.3 we present the different results obtained.

3.1 Datasets

To conduct our experiments, we used the SNLI dataset (Bowman et al., 2015) used for tasks such as Recognizing Textual Entailment (RTE). We also use the Amazon Reviews Corpus (Keung et al., 2020), the Yelp Reviews Dataset (Yelp, 2015) and the Stanford Sentiment Treebank for fine grained classification (SST-5) dataset (Socher et al., 2013).

SNLI: Developped by (Bowman et al., 2015), this corpus is a collection of 570k human-written English (including 10k for testing and 10k for validation) pairs of sentences dedicated to the Natural Language Inference (NLI) task. It is composed of three balanced labels: $C = (\text{entailment}, \text{neutral}, \text{contradiction})$. To accelerate the training, we used a random subsample of 250k rows from the training set. The ordinal relationship between the classes is taken into account by using the matrix defined in equation [7] as the distance matrix.

Amazon Reviews: This dataset, published by (Keung et al., 2020), was obtained by gathering customer reviews of product from several categories published on the Amazon marketplace in six different languages. We only kept the reviews written in English and the corresponding star rating (an integer between 1 and 5). It represents a total dataset of 210k samples, including 5k for testing and 5k for validation.

Yelp Reviews: Extracted from the Yelp Dataset Challenge 2015 data (Yelp, 2015), it was first used as a text classification benchmark in (Zhang et al., 2015). It is a balanced dataset composed of 700k samples of reviews (50k for testing) extracted from Yelp, a website hosting crowd-sourced reviews about businesses. Each sample is a (text, 5-star rating) pair. To reduce the time taken for training, a random subsample of the training set of size 200k was used as the training set, and one of size 20k was used for the validation set.

SST-5: Introduced by (Socher et al., 2013), the Stanford Sentiment Treebank (SST) is a corpus with parse trees enabling sentiment analysis. It is composed of 12k sentences extracted from movie reviews and annotated by 3 humans. In the SST fine-grained version (or SST-5), each phrase is labelled as a 5 star rating corresponding to: negative, somewhat negative, neutral, somewhat positive, positive.

3.2 Metrics

In this article we use the classical metrics for ordinal classification (Cardoso and Sousa, 2011; Gaudette and Japkowicz, 2009).

Off-by-*k* Accuracy: In the case of ordinal classification, the Off-by-k Accuracy, or OB_k, is the percentage of total predictions where the index of the predicted label $\hat{y} \in (C_1, ..., C_N)$ and the one from the true label differ from less than k. In our experiments, we assumed that $\forall i \in [\![2, N]\!]$: $d(C_{i-1}, C_i) = 1$ so the OB_k can be formulated as:

$$OB_k = 100 \times \frac{\sum_{s=1}^{S} \mathbb{1}\{d(y_s, \hat{y_s}) \leq k\}}{S} \quad (3)$$

with S being the number of examples.

Mean Absolute Error for Classification: To measure the mean distance between the predicted labels and the true ones, we use the MAE:

$$MAE = \frac{\sum_{s=1}^{S} d(y_s, \hat{y_s})}{S} \quad (4)$$

where d is the distance defined Section 2.1.

Mean Squared Error for Classification: To complete the MAE, we measure the mean squared error:

$$MSE = \frac{\sum_{s=1}^{S} d(y_s, \hat{y_s})^2}{S} \quad (5)$$

Kendall Tau: The Kendall τ (Kendall, 1938) is a measure of rank correlation between two measured quantities. It is defined as :

$$\tau = \frac{\#\{\text{concordant pairs}\} - \#\{\text{discordant pairs}\}}{\binom{S}{2}} \quad (6)$$

where $\forall (i, j) \in [\![1, S]\!]^2, i < j$, if the sort order of (y_i, y_j) and $(\hat{y_i}, \hat{y_j})$ agrees, then $(y_i, \hat{y_i})$ and $(y_j, \hat{y_j})$ are concordant pairs, and discordant pairs otherwise.

Remark: metrics such as the Accuracy or the F_1 score are often used to evaluate models in classification tasks. But in the particular case of ordinal classification, these metrics are not considered relevant as they do not truly outline the performance of a model. Indeed, if 2 models A and B predict the same amount of samples correctly, but model A predicts all the other samples incorrectly with predictions that are really distant to the true labels, while the wrong predictions of model B are labels that are close to the true ones, then models A and B have the same accuracy, but model B should be considered better than model A. Like the accuracy, the multi-class F_1 score does not take into account the distance between classes and is therefore not appropriate for ordinal classification.

3.3 Experimental Results

3.3.1 Model Used

To conduct our experiments, we have trained the **BERT-tiny** model (Turc et al., 2019) on the four datasets listed in section 3.1. The choice of using a smaller version of BERT (Devlin et al., 2018) was made for several reasons. First, having less parameters, this model is a lot faster to train. Secondly, it produces scores lower than bigger models such as **BERT-base**, allowing to better highlight the impact of different loss functions on scores. Finally, being a smaller version of the BERT model, the results provided here are assumed to be generalised to bigger BERT models and other similar Transformers models. The code is available at `https://github.com/glanceable-io/ordinal-log-loss`.

3.4 Distance Matrices

As explained in section 2, each ordinal classification task comes with distance matrix D that reflects the proximity between the different labels. For the SNLI dataset, the ordered labels are $C = (\text{entailment}, \text{neutral}, \text{contradiction})$ while for the other 3 datasets, the ordered labels are $C = (1, 2, 3, 4, 5)$. As mentioned in section 3.2, for any two neighbors labels, we choose a distance of 1 between them. As a result, the distance matrix for the SNLI task is:

$$D = \begin{pmatrix} 0 & 1 & 2 \\ 1 & 0 & 1 \\ 2 & 1 & 0 \end{pmatrix} \quad (7)$$

while the one for the 1 to 5 stars rating tasks is :

$$D = \begin{pmatrix} 0 & 1 & 2 & 3 & 4 \\ 1 & 0 & 1 & 2 & 3 \\ 2 & 1 & 0 & 1 & 2 \\ 3 & 2 & 1 & 0 & 1 \\ 4 & 3 & 2 & 1 & 0 \end{pmatrix} \quad (8)$$

3.5 Procedure

For each dataset, we trained the BERT-tiny model with 6 different types of losses: the cross-entropy, the ordinal log-loss (our loss), the weighted kappa loss (de la Torre et al., 2018), the soft labels loss (Bertinetto et al., 2020), the Earth Mover's Distance-based loss (Rubner et al., 2000; Hou et al.,

Datasets	Batch Size	Num Epochs	Stopping Rate	Weight Decay
Yelp Reviews	1024	100	5	0.01
Amazon Reviews	1024	100	5	0.01
SST-5	1024	2340	117	0.01
SNLI	1024	80	4	0.01

Table 1: Training parameters for each dataset

2016) and the CORAL framework (Cao et al., 2020). We wanted to compare our loss with these five other losses for the following reasons: the cross entropy loss is a very common loss in text classification and the other four introduced losses outperformed a significant number of other losses in many ordinal tasks.

For the ordinal log-loss, we chose α in $\{1, 1.5, 2\}$, for the soft labels loss, we chose β in $\{2, 3, 4\}$ because it gave us the best results (although in the original paper, the values used for β are higher). For each loss, we trained the model with 5 different learning rates : $\{1, 2.5, 5, 7.5\} \times 10^{-5}$, and 10^{-4}. For the CORAL loss, we also tested higher learning rates (around 10^{-3}) as it showed considerable gains. And for each learning rate, the pre-trained model was trained 5 times. Finally, for each dataset, for each loss, we chose the learning rate that gave the best scores in averages for the 5 independent trainings. We could have played with other hyper-parameters such as batch size or weight decay, but that would have enormously increased the number of experiments to be done. Each training requires a lot of resources and computing time. For example, to produce the results of this paper, approximately 800 hours of training on two NVIDIA GeForce RTX 3080 were required. We have therefore made the choice to keep only the learning rate, and to set the others at default values. The only non-default value hyper-parameter is the batch-size that we set at the maximum value to speed-up training.

3.6 Results

The results of the experiments are shown in table 2 and 3. The best model is colored in dark green and the second best in light green. In table 3, according to the procedure described in section 3.5, for each line in the table we took the average scores for the 5 independent trainings for the given learning rate. We did not display the OB2 score for SNLI because there are only three classes.

We can observe that the OLL gave better results for all the metrics used, although the SOFT loss is performing well too on the MAE metric. Results of the OLL loss vary with the α parameter : while $\alpha \in \{1, 1.5\}$ gives better results on the SNLI and SST-5 datasets, for $\alpha = 2$, the OLL loss is providing good results on the other 2 datasets. Overall, $\alpha = 1.5$ seems to be a good tradeoff.

To have a clearer idea of which losses perform better, we completed the table 2, where each line displays the average rank of the corresponding loss on the 4 datasets, for each metric. Although the SOFT loss with $\beta = 4$ gives interesting results for the MAE and the Kendall Tau, the OLL loss seems to perform better overall. The impact of the α parameter in the OLL loss vary, depending on the dataset and the number of classes, but the table 2 confirmed that $\alpha = 1.5$ is a good trade-off.

Loss	OB1	OB2	MAE	MSE	Kendall Tau
CE	6.25	7.67	5	7	7.25
OLL-1	3.5	2.67	3.35	2.25	2.75
OLL-1.5	1.5	1.33	3	1.5	2
OLL-2	1.5	1.67	6.75	2.25	5
WKL	7	3.67	9.25	7.75	7.25
SOFT-2	8	8	6.75	7.75	5.5
SOFT-3	6.5	7.67	3.75	7	5.25
SOFT-4	5.75	6.67	2.25	6	4.5
EMD	4.5	5.33	3.25	4.5	3.75
CORAL	8	6.67	9.5	8.5	8.75

Table 2: Losses mean rank on each metrics

4 Conclusion

We introduced a simple and novel loss function specially designed for the ordinal classification task. This loss is intuitive and easy to use. We evaluated our method on four benchmark ordinal text classification datasets and against five different metrics. Our loss outperforms state-of-the-art comparable and previously introduced losses. We also experimentally find good hyper-parameters to use. Thus, the contribution of this article is to introduce a new loss (OLL) that is easier to use than the majority of recently introduced losses and which gives better results for ordinal classification applied to NLP tasks. We believe that those results could be extended to other machine learning tasks in computer vision, speech or structured data for instance.

Dataset	Loss	Learn Rate	OB1	OB2	MAE	MSE	Kendall Tau
Yelp reviews	CE	7.5e-5	92.9 ± 0.1	97.7 ± 0.0	0.529 ± 0.001	0.809 ± 0.001	0.713 ± 0.000
	OLL-1	5e-5	92.7 ± 0.0	98.0 ± 0.0	0.536 ± 0.000	0.796 ±0.000	0.712 ± 0.000
	OLL-1.5	1e-4	93.1 ± 0.0	98.4 ± 0.1	0.530 ± 0.003	0.750 ± 0.004	0.718 ± 0.001
	OLL-2	1e-4	93.3 ±0.1	98.6 ± 0.0	0.534 ± 0.003	0.742 ± 0.003	0.716 ± 0.001
	WKL	1e-4	92.1 ± 0.1	98.2 ± 0.1	0.554 ± 0.003	0.814 ±0.011	0.712±0.001
	SOFT-2	7.5e-5	92.6 ± 0.2	97.7 ± 0.1	0.535 ± 0.005	0.826 ± 0.016	0.712±0.001
	SOFT-3	7.5e-5	92.8 ±0.2	97.7 ±0.1	0.532 ±0.003	0.817 ±0.011	0.712±0.001
	SOFT-4	1e-4	92.9 ±0.0	97.9 ±0.0	0.529 ±0.000	0.804 ±0.000	0.714±0.000
	EMD	7.5e-5	92.9 ± 0.1	97.9 ± 0.0	0.532 ± 0.002	0.804 ± 0.004	0.713±0.001
	CORAL	5e-3	90.6 ± 0.0	98.0 ± 0.0	0.600 ± 0.000	0.898 ± 0.000	0.682±0.000
Amazon reviews	CE	5e-5	90.9 ± 0.3	97.8 ±0.1	0.578 ±0.004	0.897 ±0.008	0.692±0.002
	OLL-1	5e-5	92.3 ±0.1	98.5 ±0.1	0.570 ±0.001	0.802 ±0.005	0.699±0.001
	OLL-1.5	2.5e-5	92.5 ±0.2	98.6 ±0.0	0.567 ±0.004	0.787 ±0.009	0.701±0.003
	OLL-2	5e-5	92.5 ±0.0	98.6 ±0.0	0.577 ±0.001	0.791±0.002	0.697±0.000
	WKL	5e-5	91.2 ±0.3	98.5 ±0.1	0.591 ±0.008	0.847 ±0.015	0.698±0.003
	SOFT-2	5e-5	90.7 ±0.1	97.9 ±0.1	0.579 ±0.005	0.897 ±0.012	0.695±0.003
	SOFT-3	5e-5	90.8 ±0.2	97.8 ±0.1	0.577 ±0.004	0.899 ±0.012	0.693±0.002
	SOFT-4	5e-5	90.7 ±0.0	97.7 ±0.0	0.577 ±0.000	0.909 ±0.000	0.694±0.000
	EMD	5e-5	91.8 ± 0.0	98.2 ± 0.0	0.569 ± 0.002	0.843 ± 0.003	0.699±0.001
	CORAL	5e-3	89.1 ± 0.6	98.0 ± 0.1	0.634 ± 0.004	0.964 ± 0.017	0.665±0.000
SST-5	CE	5e-5	85.2 ±0.2	97.0 ±0.2	0.754 ±0.008	1.171 ±0.012	0.533±0.005
	OLL-1	7.5e-5	86.7 ±0.2	98.0 ±0.1	0.738 ±0.002	1.084 ±0.008	0.548±0.003
	OLL-1.5	7.5e-5	86.9 ±0.2	98.0 ±0.1	0.739 ±0.000	1.081 ±0.005	0.544±0.002
	OLL-2	1e-5	86.3 ±0.4	97.7 ±0.2	0.757 ±0.007	1.121 ±0.016	0.531±0.004
	WKL	1e-5	83.8 ±0.7	97.2 ±0.1	0.806 ±0.019	1.259 ±0.038	0.520±0.009
	SOFT-2	2.5e-5	84.8 ±0.6	96.9 ±0.4	0.754 ±0.004	1.186 ±0.031	0.548±0.003
	SOFT-3	1e-5	85.2 ±0.3	97.0 ±0.2	0.748 ±0.001	1.166 ±0.011	0.544±0.005
	SOFT-4	7.5e-5	86.1 ±0.0	97.3 ±0.0	0.738 ±0.000	1.124 ±0.000	0.549±0.000
	EMD	7.5e-5	85.8 ± 0.3	97.2 ± 0.2	0.745 ± 0.008	1.143 ± 0.022	0.543±0.010
	CORAL	1e-3	74.8 ± 2.6	91.8 ± 2.1	1.050 ± 0.076	1.999 ± 0.245	0.446±0.020
SNLI	CE	7.5e-5	97.2 ±0.0		0.208 ±0.000	0.264 ±0.001	0.773±0.001
	OLL-1	1e-4	98.3 ±0.0		0.202 ±0.002	0.237 ±0.002	0.786±0.002
	OLL-1.5	7.5e-5	98.3 ±0.0		0.207 ±0.000	0.241 ±0.000	0.781±0.000
	OLL-2	7.5e-5	98.5 ±0.0		0.214 ±0.003	0.244 ±0.004	0.777±0.003
	WKL	1e-4	97.7 ±0.1		0.238 ±0.005	0.283 ±0.006	0.750±0.005
	SOFT-2	2.5e-5	97.3 ±0.1		0.208 ± 0.004	0.261 ±0.006	0.775 ±0.004
	SOFT-3	1e-4	97.4 ±0.0		0.204 ±0.000	0.257 ±0.001	0.779±0.000
	SOFT-4	7.5e-5	97.3 ±0.0		0.205 ±0.000	0.259 ±0.000	0.776±0.000
	EMD	1e-4	97.6 ±0.1		0.205 ± 0.004	0.254 ±0.006	0.779 ±0.004
	CORAL	2.5e-5	98.3 ±0.1		0.213 ± 0.005	0.247 ±0.007	0.778 ±0.006

Table 3: Losses comparisons on 4 datasets: Yelp reviews, Amazon reviews, SST-5 and SNLI

References

Erin L Allwein, Robert E Schapire, and Yoram Singer. 2000. Reducing multiclass to binary: A unifying approach for margin classifiers. *Journal of machine learning research*, 1(Dec):113–141.

Luca Bertinetto, Romain Mueller, Konstantinos Tertikas, Sina Samangooei, and Nicholas A. Lord. 2020. Making better mistakes: Leveraging class hierarchies with deep networks. In *The IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*.

Samuel R. Bowman, Gabor Angeli, Christopher Potts, and Christopher D. Manning. 2015. A large annotated corpus for learning natural language inference. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing (EMNLP)*. Association for Computational Linguistics.

Wenzhi Cao, Vahid Mirjalili, and Sebastian Raschka. 2020. Rank consistent ordinal regression for neural networks with application to age estimation. *Pattern Recognition Letters*, 140:325–331.

Jaime S Cardoso and Ricardo Sousa. 2011. Measuring the performance of ordinal classification. *International Journal of Pattern Recognition and Artificial Intelligence*, 25(08):1173–1195.

Jordi de la Torre, Domenec Puig, and Aida Valls. 2018. Weighted kappa loss function for multi-class classification of ordinal data in deep learning. *Pattern Recognition Letters*, 105:144–154. Machine Learning and Applications in Artificial Intelligence.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. BERT: pre-training of deep bidirectional transformers for language understanding. *CoRR*, abs/1810.04805.

Raul Diaz and Amit Marathe. 2019. Soft labels for ordinal regression. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 4738–4747.

Eibe Frank and Mark Hall. 2001. A simple approach to ordinal classification. In *European conference on machine learning*, pages 145–156. Springer.

Lisa Gaudette and Nathalie Japkowicz. 2009. Evaluation methods for ordinal classification. In *Canadian conference on artificial intelligence*, pages 207–210. Springer.

Pedro Antonio Gutiérrez, Maria Perez-Ortiz, Javier Sanchez-Monedero, Francisco Fernandez-Navarro, and Cesar Hervas-Martinez. 2015. Ordinal regression methods: survey and experimental study. *IEEE Transactions on Knowledge and Data Engineering*, 28(1):127–146.

R Herbrich, T Graepel, and K Obermayer. 2000. Large-margin thresholded ensembles for ordinal regression: theory and practice. *Advances in Large Margin Classifiers*, pages 115–132.

Le Hou, Chen-Ping Yu, and Dimitris Samaras. 2016. Squared earth mover's distance-based loss for training deep neural networks. *arXiv preprint arXiv:1611.05916*.

M. G. Kendall. 1938. A New Measure Of Rank Correlation. *Biometrika*, 30(1-2):81–93.

Phillip Keung, Yichao Lu, György Szarvas, and Noah A. Smith. 2020. The multilingual amazon reviews corpus. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing*.

Sotiris B Kotsiantis and Panagiotis E Pintelas. 2004. A cost sensitive technique for ordinal classification problems. In *Hellenic Conference on Artificial Intelligence*, pages 220–229. Springer.

Jason DM Rennie and Nathan Srebro. 2005. Loss functions for preference levels: Regression with discrete ordered labels. In *Proceedings of the IJCAI multidisciplinary workshop on advances in preference handling*, volume 1. Citeseer.

Yossi Rubner, Carlo Tomasi, and Leonidas J Guibas. 2000. The earth mover's distance as a metric for image retrieval. *International journal of computer vision*, 40(2):99–121.

Richard Socher, Alex Perelygin, Jean Wu, Jason Chuang, Christopher D Manning, Andrew Y Ng, and Christopher Potts. 2013. Recursive deep models for semantic compositionality over a sentiment treebank. In *Proceedings of the 2013 conference on empirical methods in natural language processing*, pages 1631–1642.

Iulia Turc, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. Well-read students learn better: On the importance of pre-training compact models. *arXiv preprint arXiv:1908.08962v2*.

Jan Verwaeren, Willem Waegeman, and Bernard De Baets. 2012. Learning partial ordinal class memberships with kernel-based proportional odds models. *Computational Statistics & Data Analysis*, 56(4):928–942.

Yelp. 2015. Yelp dataset challenge.

Xiang Zhang, Junbo Jake Zhao, and Yann LeCun. 2015. Character-level convolutional networks for text classification. *CoRR*, abs/1509.01626.

Ask Question First for Enhancing Lifelong Language Learning

Han Wang[1,2] and **Ruiliu Fu**[1,2] and **Xuejun Zhang**[1,2] and **Jun Zhou**[1,2] and **Qingwei Zhao**[1,2]
[1]Institute of Acoustics, Chinese Academy of Sciences, Beijing, China.
[2]University of Chinese Academy of Sciences, Beijing, China
{wanghan, furuiliu, zhangxuejun, zhoujun, qzhao}@hccl.ioa.ac.cn

Abstract

Lifelong language learning aims to stream learning NLP tasks while retaining knowledge of previous tasks. Previous works based on the language model and following data-free constraint approaches have explored formatting all data as "begin token (*B*) + context (*C*) + question (*Q*) + answer (*A*)" for different tasks. However, they still suffer from catastrophic forgetting and are exacerbated when the previous task's pseudo data is insufficient for the following reasons: (1) The model has difficulty generating task-corresponding pseudo data, and (2) *A* is prone to error when *A* and *C* are separated by *Q* because the information of the *C* is diminished before generating *A*. Therefore, we propose the Ask Question First and Replay Question (AQF-RQ), including a novel data format "*BQCA*" and a new training task to train pseudo questions of previous tasks. Experimental results demonstrate that AQF-RQ makes it easier for the model to generate more pseudo data that match corresponding tasks, and is more robust to both sufficient and insufficient pseudo-data when the task boundary is both clear and unclear. AQF-RQ can achieve only 0.36% lower performance than multi-task learning.

1 Introduction

Lifelong learning is the capacity of human beings to acquire, reconstruct, strengthen, and transfer knowledge (Ring, 1997). Human beings can learn new knowledge while consolidating old knowledge by first detecting and learning the distinctions between old and new knowledge and then simplifying the old and new knowledge based on leveraging the common points. This concept is critical for encouraging machines to learn NLP tasks in a similar way that people do. In the application of NLP, new data are continuously acquired and categorized as either new data for existing tasks or new data for new tasks. For new data of existing tasks, the traditional method, known as isolated learning (Chen and Liu, 2018), is to retrain the model with the old data appended with new data. For data from new tasks, multi-task learning (MTL) integrates data from previous and new tasks to retrain the model. Both of these methods are limited to assuming that all new and old data can be obtained during training. However, the reality is that tasks are acquired and trained in the stream, which makes the model suffer from catastrophic forgetting (Ring, 1997; McCloskey and Cohen, 1989; French, 1999) (i.e. forgetting previously learned tasks/knowledge).

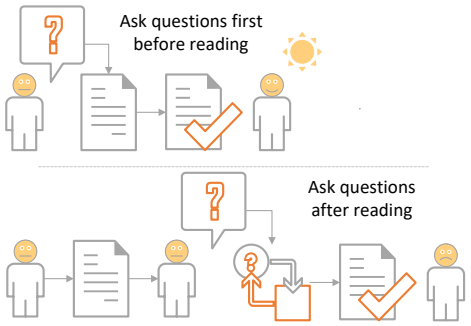

Figure 1: Illustration of the difference between asking questions before reading and asking after reading.

Lifelong language learning (LLL), which this paper focuses on, aims to learn a stream of NLP tasks with lifelong learning. LAMOL (Sun et al., 2019) has recently proposed implementing a language model for LLL by formatting all data as QA-style and generating pseudo data instead of real data for previous tasks in order to prevent catastrophic forgetting. Many works (Chuang et al., 2020; Sun et al., 2020; Kanwatchara et al., 2021) have investigated how to improve it through methods that require more computing resources or additional parameters (e.g. knowledge distillation or adding sub-networks). In this paper, we refer to these works as LAMOL-based methods. In LAMOL-based methods, each example is consist of four segments: task-specific or task-independent token

Proceedings of the 29th International Conference on Computational Linguistics, pages 4610–4621
October 12–17, 2022.

[TASK]/[GEN] (*B*), context (*C*), question (*Q*), and answer (*A*). Then each example is formatted as "*B+C+Q+A*" ("*BCQA*"). LAMOL has a gap between MTL that is regarded as the upper bound of LLL. Other LAMOL-based methods require more computing resources or additional parameters but still have a gap between MTL especially pseudo samples is unsufficient. Therefore, the "*BCQA*" format is useful but not train-efficient enough for reasons below: (1) generating task-corresponding pesudo data is hard because the similar *C* can be found in distinct tasks; (2) the *C*'s information is diminished when *A* is generated due to the fact that *A* and *C* are separated by *Q*.

To generate more task-corresponding pseudo data for previous tasks and tighten the relationship between the *C* and the *A*, we proposed the Ask Question First (AQF) and Replay Question (RQ).

The AQF formats all data into a novel format: "*B+Q+C+A*" ("*BQCA*"), which is consistent with human reading comprehension behavior. Questions are more than just questions; they direct our learning. As is shown in Fig.1, when people do reading comprehension, they usually read the questions first, then read the articles with the questions, and pay attention to what can answer the questions while reading. This is an efficient reading and learning method, which known as metacognition (Flavell, 1979), has been researched in the field of education and psychology. There are two benefits to applying the "*BQCA*" format: (1) It's easier to generate pseudo-data that matches the corresponding task because the *Q* has different but limited types for each task. (2) The model is more stable when there isn't enough pseudo data because the *A* is right next to the *C*. Furthermore, the model can pay more attention to important information in the *C* with the help of the *Q*.

The RQ introduce a novel training task to help model generate more task-corresponding pseudo data. In the "*BQCA*" format, generating the correct *Q* is crucial for generating task-corresponding pseudo-data. However, the *Q* of the previous may be covered by the new task since we cannot predict the number and type of *Q* of the new task. In order to strengthen the generation of the *Q* of the previous task, we generate pseudo-problems of the previous task to train the model.

The contributions of our paper are listed below: (1) We proposed the Ask Question First and Replay Question (AQF-RQ[1]) to alleviate catastrophic forgetting when the pseudo data is sufficient and insufficient without additional computation resources and parameters. (2) We proposed a novel data format "*BQCA*" to make data train-efficient and generating corresponding pseudo data easier. (3) We proposed a novel training task to help the model generate correct questions and then generate more task-corresponding pseudo data.

2 Related Work

Lifelong language learning (LLL) is an essential step in promoting the realization of general artificial intelligence in the field of NLP. (Liu et al., 2019; Mi et al., 2020; Huang et al., 2021) have studied LLL on a single type of NLP task by regularization or replaying real data. Recently, LAMOL (Sun et al., 2019) uses a language model (LM) to learn various kinds of NLP tasks in QA-style. In LAMOL, the pseudo-data generated by the model is trained together with the new task to alleviate catastrophic forgetting. Many works explored the enhancement of LAMOL with additional computation resources or adding sub-networks. L2KD (Chuang et al., 2020), DnR (Sun et al., 2020) and DFSD (Wang et al., 2022) distilled parts or all layers of the model to improve LAMOL. ARPER (Mi et al., 2020) applied regualarization on parameters with prioritized exemplar replay. Rational-LAMOL (Kanwatchara et al., 2021) applied critical freezing guided by rationale information which is obtained by human or unsupervised rationale generation (Chang et al., 2020). (Madotto et al., 2021) applied Adapter (Houlsby et al., 2019) to plug into pretrain language model. Those are effective but require more computational resources (e.g., distillation and regularization) or new parameters (such as adding sub-networks for new tasks). We aspire to improve without increasing our resources. Consequently, based on the metacognition (Flavell, 1979; Bowler, 2010; Braithwaite and Sprague, 2021) proposed in education and psychology, we propose a more training-efficient and robust data format: first ask questions, then observe the context, finally answer the questions. In addition, we introduce replay questions to strengthen the model's attention to questions.

[1] https://github.com/CodeHan/AQF-RQ

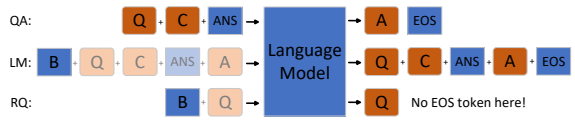

Figure 2: Illustration of QA/LM/RQ task with the "*BQCA*" format.

Case ID	Content	Cor.
1	$B^{T_1}+C^{T_1}+Q^{T_1}+A^{T_1}$	✓
2	$B^{T_1}+C^{T_2}+Q^{T_2}+A^{T_2}$	✗
3	$B^{T_2}+C^{T_2}+Q^{T_2}+A^{T_2}$	✓
4	$B^{T_2}+C^{T_1}+Q^{T_1}+A^{T_1}$	✗

Table 1: The cases of the pseudo sample after the model learning two tasks in turn with "*BCQA*" format. Cor. is the abbreviation of "corresponding".

3 Methodology

In this section, we first introduce LAMOL in Section 3.1. Then, our proposed Ask Question First (AQF) is detailed in Section 3.2. Thirdly, we introduce the Replay Questions in Section 3.3. Finally, we summarize the training objectives in our paper in Section 3.4.

3.1 LAMOL

LAMOL (Sun et al., 2019) proposed using a single language model for lifelong language learning. A stream of various NLP tasks is learned by GPT-2 (Radford et al., 2019) with joint training of the language model (LM) task and the question-answering (QA) task. In LAMOL, all tasks are formatted QA-style. Each example can consist of four segments: task-specific or task-independent token [TASK]/[GEN] (*B*), context (*C*), question (*Q*), and answer (*A*). In this way, the LM task is to generate "*C*+*Q*+*A*" by inputting [TASK]/[GEN], and the QA task is to generate "*A*" by inputting "*C*+*Q*". The loss of LAMOL is calculated as below:

$$\mathcal{L}_{LAMOL} = \mathcal{L}_{QA} + \lambda \mathcal{L}_{LM} \quad (1)$$

where λ is the weight of the LM task.

With the help of the LM task, the [TASK]/[GEN] can be used as the first token to input the model and generate the pseudo sample by greedy decoding. From the second task on, pseudo samples of previous tasks are generated in this way and jointly trained with the new task dataset to alleviate catastrophic forgetting. Let denote the set of pseudo samples as $\mathcal{P}_i = \{ps_j | i, j \in N_+, 1 \leq j < i\}$, where ps_j represents the pseudo data of j-th task.

The number of pseudo samples of previous tasks is determined by the size of the new task and a hyperparameter sampling ratio $\gamma \in (0, 1]$. Assuming that $T = \{T_1, T_2, \ldots, T_N\}$ represents N tasks to be learned. Let $|D_t|, t \in [2, N]$ denote the size of t-th task. LAMOL generates $|ps_i| = \frac{\gamma}{t-1}|D_t|$ pseudo-samples for each previously learned task. Then, D_t is joined with $\mathcal{P}_t$ to train the model. However, the performance of LAMOL still has a gap between multi-task learning (MTL), which is regarded as the upper bound of lifelong learning, since not each pseudo sample is tied to [TASK]/[GEN]. In this paper, we propose Ask Question First (AQF) and replaying questions to make data train-efficient and shorten the gap between MTL.

3.2 Ask Question First

Ask Question First (AQF) is a novel data format that makes the model imitate the process of human reading comprehension to learn knowledge from the training set efficiently. LAMOL proposed a data format, "*B*+*C*+*Q*+*A*" ("*BCQA*"), which is useful but not train-efficient enough. All pseudo samples are generated starting with [TASK]/[GEN]. However, [TASK]/[GEN] is not strongly tied with the *C* of the tasks because the context is complex and volatile for the task. Different tasks can have similar contexts. Therefore, [TASK]/[GEN] is easy to be biased to the new task resulting in pseudo samples not corresponding to their tasks when more tasks are learned. These non-corresponding pseudo-samples jointly trained with the new task will aggravate the catastrophic forgetting of the model. Otherwise, the *C*'s information is diminished when *A* is generated due to the fact that *A* and *C* are separated by *Q*.

To generate more corresponding pseudo samples, we propose AQF format all data as "*B*+*Q*+*C*+*A*" ("*BQCA*") which is simple yet efficient to make the model learn knowledge from the training set efficiently. We choose the task-specific token to analyze why our proposed "*BQCA*" is better than the naive "*BCQA*". Assuming that the model has learned two tasks T_1 and T_2, pseudo samples of T_1 and T_2 needed to be generated before training the new task. Two types of task-specific tokens B_1 and B_2 are inputted to the model to generate pseudo samples with greedy decoding. There are four main kinds of pseudo samples that can be obtained as shown in Table 1. Case 1 and Case 3

are nice pseudo samples that can alleviate catastrophic forgetting, but Case 2 and Case 4 are terrible ones that can aggravate catastrophic forgetting. Ideally, pseudo samples, which we denote as task-corresponding pseudo-samples $\mathcal{P}_c$, like Case 1 and Case 3 are what we expect to generate. However, in fact, due to catastrophic forgetting, Case 2 and Case 4 also appear. These pseudo samples like case 2 and case 4 are denoted as the not task-corresponding samples $\mathcal{P}_{nc}$. The probability of generating $\mathcal{P}_{nc}$ is positively correlated with the severity of catastrophic forgetting. Therefore, the model will form a vicious circle: forgetting old tasks and generating $\mathcal{P}_{nc}$ with greater probability. For pseudo samples start with task-specific token, the ps_i consists of corresponding pseudo data ps_i^c and not corresponding pseudo data ps_i^{nc} (i.e. $ps_i = \{ps_i^c, ps_i^{nc}\}$).

In LAMOL-based methods, they used GPT-2 (Radford et al., 2019) for experiments. GPT-2 is an autoregressive model that can only obtain information before the current position, and then predict the words. Therefore, the probability of our proposed "*BQCA*" format and naive "*BCQA*" format can be calculated by Eq.(2) and Eq.(3), respectively:

$$\begin{aligned} P_{AQF} &= P(BQCA) \qquad (2)\\ &= P(A|BQC)P(C|BQ)P(Q|B)P(B)\\ P_{naive} &= P(BCQA) \qquad (3)\\ &= P(A|BCQ)P(Q|BC)P(C|B)P(B) \end{aligned}$$

As shown in Eq.(2) and Eq.(3), $P(B)$ is a constant because it is given as the first token. Next, $P(Q|B)$ and $P(C|B)$ affect the quality of the generated pseudo data. To obtain $\mathcal{P}_c$, the *Q* or *C* should be generated to be corresponding to the given *B*, then the remaining generation has a greater probability correspond to *B*.

Then we first consider $P(Q|B)$ and $P(C|B)$. Let Θ_N denote the parameters of the model after learning N tasks in turn. Assuming two tasks have been learned on the condition of sufficient pseudo samples (i.e. large γ) in turn, $P(Q|B)$ and $P(C|B)$ can be written as, $P(Q|B, \Theta_2)$ and $P(C|B, \Theta_2)$. In the dataset of a task, there are thousands of kinds of *C*, but there are only limited kinds of *Q*. For a simplest classification task, only one fixed *Q* is needed. For example, in the SST (Radford et al., 2017), all *Q* is "*is review is positive or negative?*". For complex tasks, such as SRL (He et al., 2017), it is mainly based on limited types of *Q* such as *what*, *where*, *why*, *when*, *how* etc, but it is still smaller than that of *C*. Therefore, $P(Q|B, \Theta_2)$ and $P(C|B, \Theta_2)$ are inversely proportional to the vocabulary sizes of *Q* and *C*, respectively. Let $\mathcal{V}_Q = \{w_i^Q | i \in [1, |\mathcal{V}_Q|]\}$ and $\mathcal{V}_C = \{w_i^C | i \in [1, |\mathcal{V}_C|]\}$ denote the vocabulary of *Q* and *C*, respectively. w_i^Q and w_i^C denotes the word in the vocabulary of *Q* and *C*, respectively. According to the above analysis, it can be obtained that $|\mathcal{V}_Q| \ll |\mathcal{V}_C|$. For most words, since their probability is inversely proportional to the size of the vocabulary, we can conclude that $P(w_Q) \gg P(w_C)$. Slight noise perturbations, such as shifts from the data distribution of the new task, can make $P(w_C)$ more susceptible than $P(w_Q)$. On the other hand, different tasks may have similar *C* due to a large number of similar common phrases, but the *Q* is almost different in what the questions pay attention to. Therefore, we can conclude that:

$$P_{B_1}(w_Q^{T_1}) \geq P_{B_1}(w_Q^{T_2}) > P_{B_1}(w_C^{T_1}) \geq P_{B_1}(w_C^{T_2}) \qquad (4)$$

$$P_{B_2}(w_Q^{T_2}) \geq P_{B_2}(w_Q^{T_1}) > P_{B_2}(w_C^{T_2}) \geq P_{B_2}(w_C^{T_1}) \qquad (5)$$

where the subscript B_1 or B_2 means the probability on the condition of B_1 or B_2. Then, we can conclude that:

$$P(Q^{T_1}|B^{T_1}, \Theta_2) > P(C^{T_1}|B^{T_1}, \Theta_2) \qquad (6)$$

$$P(Q^{T_2}|B^{T_2}, \Theta_2) > P(C^{T_2}|B^{T_2}, \Theta_2) \qquad (7)$$

It can be concluded that generating *Q* first is more stable than generating *C*.

On the condition that the *Q* or *C* is corresponding to the B, we analysis the $P(C|BQ, \Theta_2)$ and $P(Q|BC, \Theta_2)$. With the help of GPT-2, a Transformer decoder-based model, the current token can pay more attention to the information before the current position. When the previous adjacent content contains enough information, the generated token strongly correlates with the previous content. Therefore, $P(C^{T_1}|B^{T_1}Q^{T_1}, \Theta_2)$ is close to $P(Q^{T_1}|B^{T_1}C^{T_1}, \Theta_2)$, and $P(C^{T_2}|B^{T_2}Q^{T_2}, \Theta_2)$ is close to $P(Q^{T_2}|B^{T_2}C^{T_2}, \Theta_2)$.

Finally, $P(A|BQC)$ and the $P(A|BCQ)$ are trained with the QA task. It can be ignored from the perspective of generating task-corresponding pseudo-samples. However, correct "*A*" is easier to obtain on the condition of correct "*BQC*" than correct "*BCQ*" because the "*C*" is right next to "*A*" in "*BQC*". In "*BQC*", "*Q*" can help the model pay more attention to the important information in "*C*"

and then have a larger probability of generating the correct "*A*".

From the above analysis, we can conclude that $P(Q|B)$ and $P(C|B)$ play main role on generating task-corresponding pseudo-samples $\mathcal{P}_c$ according to P_{AQF} and P_{naive}, respectively. As shown in Eq.(6) and Eq.(7), $P(Q|B)$ are better than $P(C|B)$ for generating $\mathcal{P}_c$ after sufficient pseudo samples being joint trained.

3.3 Replay Questions

Replay Questions (RQ) is proposed to strengthen the probability of generating task-corresponding *Q* when given *B*. Based on Eq.(4), $P_{B_1}(w_Q^{T_1})$ greater than or equal to $P_{B_1}(w_Q^{T_2})$ on the condition of sufficient pseudo samples being joint trained. However, the situation $P_{B_1}(w_Q^{T_2}) > P_{B_1}(w_Q^{T_1})$ may happen when the amount of pseudo samples is much less than that of the new task (i.e. $|D_2| \gg \gamma|D_2|$). Since $P(Q|B)$ plays a main role in P_{AQF}, we propose to replay questions to make $P(Q_i|B_i)$ larger than $P(Q_{j\neq i}|B_i)$.

As shown in Fig.2, RQ is to generate the questions of pseudo samples that start with [TASK]/[GEN] but have no end token. Since the object of RQ is to make the model generate task-corresponding pseudo-samples, RQ should not supply the information about the end of the sentence.

3.4 Training

In summary, our proposed AQF is applied to both the LM task and the QA task. Otherwise, we expand the training objective with our proposed RQ. The final training objective is shown as Eq.(8).

$$\mathcal{L} = \mathcal{L}_{QA}^{AQF}(D,\mathcal{P}) + \lambda\mathcal{L}_{LM}^{AQF}(D,\mathcal{P}) + \eta\mathcal{L}_{RQ}^{AQF}(\mathcal{P}) \quad (8)$$

where λ and η denotes the weight of the LM task and the RQ task.

4 Experiment Setup

4.1 Datasets

To be comparable with previous works, we chose four different kinds of tasks from DecaNLP (McCann et al., 2018) and five sequence-generation tasks (Chuang et al., 2020) from different domains. Details are summarized in the Table 7 in Appendix A. The SQuAD is a question-answering dataset with 12 main types of questions and other countless types of questions. SST is a sentiment analysis dataset with one question. SRL is a semantic role labeling dataset with 6 types of questions.

Tasks \ Format	BCQA	BQCA (ours)
SQuAD	72.3	**81.1**
SST	90.9	**92.2**
SRL	70.4	**73.9**
WOZ	84.9	**86.7**
E2ENLG	48.8	**49.2**
RNNLG (rest)	64.0	**64.6**
RNNLG (hotel)	65.4	**66.4**
RNNLG (tv)	70.8	**71.6**
RNNLG (laptop)	73.0	**73.2**

Table 2: The results on each task with single task learning (only the QA task is trained). The format "*BCQA*" was applied in previous LAMOL-based methods. Better performance in boldface.

WOZ is a goal-oriented dialogue dataset with one question. E2ENLG (Novikova et al., 2017) and RNNLG (rest/hotel/tv/laptop) (Wen et al., 2015)are sequence generation tasks with one question for different domains.

4.2 Baselines

(i) **Finetune**: Finetune GPT-2 individually based on the task order. (ii) **LAMOL** (Sun et al., 2019): Training the model with the "*BCQA*" format data. LAMOL$_G$ and LAMOL$_T$ indicate that the begin token is the task-independent token [GEN] and the ask-specified token [TASK], respectively. LAMOL$_R$ indicates that the model replays real data from previous tasks instead of pseudo data. (iii) **L2KD** (Chuang et al., 2020): An improved version of LAMOL with distillation. Firstly, training a single-task model on the new task to obtain the teacher model. Then the teacher model is distilled to the model trained on the previous tasks. This model uses the task-specified token. All the LAMOL-based baselines were studied upon the "*BCQA*" format of LAMOL. We only select L2KD on behalf of other LAMOL-based baselines because of a lack of computation resources. (iv) **Multitask**: Training all tasks simultaneously. Multitask learning is often regarded as the upper bound of lifelong learning.

4.3 Implementation Detail

For fairness comparation. we implement experiments following LAMOL[2]. GPT-2 with 12 layers is selected as the language model. All experiments are run on a single Tesla P100 (12GB). Each task is trained for 9 epochs. The pseudo data is generated

[2] https://github.com/jojotenya/LAMOL

by greedy deocde. The other hyperparameter settings are the same as for LAMOL and are detailed in the Table 6 in the Appendix B.

5 Experiments

In this section, we experiment our proposed Ask Question First and Replay Question (AQF-RQ) an on three settings: (1) single task learning in Section 5.1; (2) different types of tasks in Section 5.2; (3) the same types of tasks in different domains in Section 5.3. Finally, we explore the effectiveness of our proposed AQF in Section 5.4.

5.1 Single Task

To validate our proposed AQF that format all task into new data format "*BQCA*", we experiment on each dataset independently. Each task is only trained by the QA task. As shown in Table 2, our proposed "*BQCA*" format beats "*BCQA*" format which is applied in previous LAMOL-based methods on each task. It exhibits that our "*BQCA*" format has the ability to increase the performance of a variety of different kinds of tasks, particularly those that were originally question-answering types (e.g. SQuAD, SRL). The SQuAD is improved significantly with the "*BQCA*" format scoring 8.8 percentage points higher than the "*BCQA*" format. For SRL, 3.5 percentage points of improvement come from the "*BQCA*" format compared with the "*BCQA*" format. For SST and WOZ, the "*BQCA*" format can improve by 1.3 and 1.8 percentage points respectively. For five generation tasks, "*BQCA*" format is slightly better than "*BCQA*".

There are two reasons why our proposed "*BQCA*" format can improve the aforementioned tasks to a certain extent: (1) Due to the characteristics of autoregression and the mask attention mechanism of the Transformers Decoder, first *Q* and then *C* can enable the model to obtain more accurate attention information from *C* based on *Q*. (2) The *A* is right next to the *C*, which reduces the information loss caused by the excessive length of the historical text in comparison to "*BCQA*" where the *A* and the *C* are gapped by the *Q*.

5.2 Different Types of Tasks

For a fair comparison, we conduct experiments on three different tasks in DecaNLP following LAMOL (Sun et al., 2019). To observe the performance in the case of sufficient and insufficient pseudo data, we select the sampling ratio $\gamma = 0.2$ and $\gamma = 0.05$ for experiments. Meanwhile, we conduct experiments on the condition of task-specific token [TASK] (denoted by the subscript T) and task-independent token [GEN] (denoted by the subscript G) to verify if our proposed AQF-RQ is robust when the task boundary is clear or unclear.

5.2.1 Performance

As shown in Table 3, Finetune, which is a baseline for other methods, suffers serious catastrophic forgetting and has large gap bewteen multitask learning. Let's first observe the situation where the pseudo data is sufficient (i.e. $\gamma = 0.2$). LAMOL$_G^{0.2}$ and LAMOL$_T^{0.2}$ have good performance and are similar, indicating that in this case, the "*BCQA*" format is robust to clear and unclear task boundaries, but still 3 percentage points lower than MTL. However, AQF-RQ does better than LAMOL by 2.7 percentage points when the boundary is clear and by 2.3 percentage points when the boundary is not clear. AQF-RQ is only 0.3-0.6% worse than MTL. It is worth noting that AQF-RQ outperforms multi-task learning in the three task orders: SST-SRL-WOZ, SST-WOZ-SRL, and WOZ-SRL-SST. This indicates that AQF-RQ can not only better alleviate catastrophic forgetting but also further strengthen forward transfer between tasks.

When the pseudo data is insufficient (i.e. $\gamma = 0.05$), the performance of LAMOL drops significantly, falling 6.5-7.6 percentage points lower than that of multitask learning. Furthermore, LAMOL$_T^{0.05}$ differs from LAMOL$_G^{0.05}$, indicating that LAMOL is not robust enough for clear and unclear task boundaries when the amount of pseudo data is insufficient. Although LAMOL's job boundary is apparent, our proposed AQF-RQ can improve by 4.7 and 5.5 percentage points compared to LAMOL. This illustrates that even when the amount of pseudo data is insufficient, AQF-RQ is still robust to both clear and unclear task boundaries, as evidenced by AQF-RQ$_T^{0.05}$ and AQF-RQ$_G^{0.05}$. It is also worth noting that AQF-RQ$_T^{0.05}$ > AQF-RQ$_G^{0.05}$ >LAMOL$_G^{0.2}$ >LAMOL$_T^{0.2}$, which means that AQF-RQ is more data-efficient and still performs better than LAMOL with a 75% reduction in the amount of pseudo data.

5.2.2 Distribution of Pseudo Data

The another main objective of our proposed AQF-RQ is to generate more corresponding-task pseudo data. Ideally, the ratio of pseudo data for each task is 1:1, which means $|ps_i| = |ps_j|, i \neq j$. We can

Methods	SST SRL WOZ	SST WOZ SRL	SRL SST WOZ	SRL WOZ SST	WOZ SST SRL	WOZ SRL SST	Avg.	Std.
Finetune	42.8	25.2	58.8	32.2	25.6	36.2	36.8	11.6
LAMOL$_T^{0.05}$	77.3	76.9	78.1	74.7	73.4	75.8	76.0	1.6
LAMOL$_G^{0.05}$	79.6	78.9	73.1	73.7	68.6	75.7	74.9	3.4
AQF-RQ$_T^{0.05}$	81.6	80.8	80.3	80.0	80.6	80.7	80.7	1.1
AQF-RQ$_G^{0.05}$	82.2	81.3	78.9	79.0	79.2	81.7	80.4	2.9
LAMOL$_T^{0.2}$	79.4	79.9	80.1	78.7	79.8	79.0	79.5	0.5
LAMOL$_G^{0.2}$	80.0	80.7	79.6	78.7	78.4	80.5	79.7	0.8
AQF-RQ$_T^{0.2}$	82.2	**83.0**	**81.4**	**81.6**	82.2	**82.9**	**82.2**	**0.5**
AQF-RQ$_G^{0.2}$	**82.8**	82.7	80.9	80.9	**82.3**	82.4	82.0	0.8
Multitask	82.5							

Table 3: The results on [SST, SRL, and WOZ]. Each column is the average score of a task order on three tasks. Avg. means that the average score on 6 task orders of [SST, SRL, and WOZ]. In the column "Methods", The subscripts T and G indicate the task-specific token [TASK] and task-independent token [GEN], respectively. The superscript indicates the value of sampling ratio γ.

Methods	SST SRL WOZ	SST WOZ SRL	SRL SST WOZ	SRL WOZ SST	WOZ SST SRL	WOZ SRL SST
	$\|ps_1\| : \|ps_2\|$					
LAMOL$_G^{0.05}$	27:99	23:297	7:119	3:343	19:301	6:340
AQF-RQ$_G^{0.05}$	17:109	12:308	22:104	29:317	70:250	168:178
LAMOL$_G^{0.2}$	292:216	319:963	128:380	284:1100	225:1057	388:996
AQF-RQ$_G^{0.2}$	253:255	255:1027	239:269	603:781	712:570	648:736
Methods	SST SRL WOZ	SST WOZ SRL	SRL SST WOZ	SRL WOZ SST	WOZ SST SRL	WOZ SRL SST
	$\|ps_1^c\| : \|ps_1^{nc}\| / \|ps_2^{nc}\| : \|ps_2^c\|$					
LAMOL$_T^{0.05}$	54:9/1:62	11:149/2:158	51:12/0:63	22:151/1:172	60:100/0:160	139:34/0:173
AQF-RQ$_T^{0.05}$	63:0/0:63	160:0/0:160	63:0/2:61	173:0/0:173	160:0/0:160	172:1/1:172
LAMOL$_T^{0.2}$	247:7/9:245	478:163/5:636	250:4/0:254	537:155/6:686	635:6/4:637	678:14/5:687
AQF-RQ$_T^{0.2}$	254:0/0:254	641:0/0:641	254:0/0:254	692:0/0:692	641:0/0:641	692:0/0:692

Table 4: The results of pseudo data distribution after learning two tasks on [SST, SRL, and WOZ]. ps_i represents the pseudo data of i-th task. The subscripts c and nc indicate whether the pseudo data is correspond to the task or not.

determine which task the pseudo data belongs to according to the *Q* because the *Q* for various tasks is distinct (see Table 7). For task-specific token [TASK], a pseudo sample is task-corresponding if the *Q* is correspond to the [TASK]. For task-indepedent token [GEN], we can not judge whether a pseudo sample corresponds to the task because of unclear task boundary. In this paper, we applied this *Q*-based judgment method to count the distribution of pseudo data after learning the second task.

As shown in Table 4, AQF-RQ can generate more pseudo data of the first task for most task orders when applying task-indepedent token [GEN]. Looking at the Table 3 together, it can be observed that the performance will be better if the ratio of the pseudo data of different tasks is closer to 1:1. For SRL-SST-WOZ, SRL-WOZ-SST, WOZ-SST-SRL, and WOZ-SRL-SST, AQF-RQ generates more pseudo data than LAMOL for the first task; hence, AQF-RQ outperforms LAMOL significantly. When $\gamma = 0.05$, AQF-RQ is 5.3-10.6 percentage points higher than LAMOL. When $\gamma = 0.05$, AQF-RQ is 1.3-3.9 percentage points higher than LAMOL. This is due to the fact that, under the AQF-proposed "*BQCA*" format, *Q* can be used to help the model pay more attention to the useful details in *C*, and then *A* being immediately next to *C* can further generate *A* more accurately.

The benefit of AQF-RQ to produce pseudo data is also significant when the sentence starts with the task-specific token [TASK]. As shown in the Table 4, the pseudo samples generated by AQF-RQ are task-corresponding in the majority of task orders, except SRL-SST-WOZ and WOZ-SRL-SST, where one or two pseudo samples are not task-corresponding. The amount of not task-corresponding pseudo samples in AQF-RQ is substantially smaller than in LAMOL, giving AQF-RQ a 1.3-3.9 percentage point advantage over LAMOL.

Methods	SST SRL WOZ	SST WOZ SRL	SRL SST WOZ	SRL WOZ SST	WOZ SST SRL	WOZ SRL SST	Avg.
LAMOL$_T$	79.4	79.9	80.1	78.7	79.8	79.0	79.5
w/ AQF	**82.1**	**83.0**	**81.3**	**80.9**	**80.7**	**81.2**	**81.5**
LAMOL$_G$	80.0	80.7	79.6	78.7	78.4	80.5	79.7
w/ AQF	**82.3**	**81.5**	**81.3**	**80.4**	**78.8**	**81.3**	**80.9**
LAMOL$_R$	81.8	80.6	81.6	81.2	80.4	80.5	81.0
w/ AQF	**82.5**	**81.9**	**82.3**	**82.4**	**82.7**	**82.7**	**82.4**
L2KD	80.1	79.6	79.5	79.7	79.9	**80.4**	79.9
w/ AQF	**81.9**	**82.4**	**80.3**	**80.8**	**80.1**	80.1	**80.9**
Multitask*				81.5			
w/ AQF				**82.5**			

Table 5: The results of effectiveness of the Ask Question First (AQF) on [SST, SRL, and WOZ] when sampling ratio $\gamma = 0.2$. Multitask* represents that model applies the "*BCQA*" format for multitask learning.

The above experimental results demonstrate that not only does AQF-RQ generate better pseudo-data, but the "*BQCA*" format is also more data-efficient and robust for lifelong language learning.

5.3 Generation for Different Domains

We conducted experiments on five sequence generation tasks for different domains by comparing LAMOL$_T$, LAMOL$_G$, and L2KD with AQF-RQ. The result is similar to Section 5.2.1 and is detailed in Appendix C. AQF-RQ is not only better than LAMOL-based baselines on performance but can also forward/backward transfer more knowledge among the same task but different domains.

5.4 Effectiveness of the Ask Question First

To verify the effectiveness of our proposed AQF, we experimentally apply AQF to each baseline: LAMOL$_T$, LAMOL$_G$, LAMOL$_R$, and L2KD. In Section 3.2, we stated that AQF is used better when the pseudo data is sufficient. Therefore, we set $\gamma = 0.2$ for experiments. This value is also the best setting for those baselines.

As shown in the Table 5, applying AQF to each baseline resulted in varying degrees of improvement. LAMOL$_T$ has the most noticeable improvement, with a 2 percentage point increase. This demonstrates that having the *A* close to the *C* using the "*BQCA*" format proposed by AQF can reduce information attenuation caused by the *A* being far away from the *C*. For example, the *C* and the *A* are separated by the *Q* in the "*BCQA*" format. This benefit is more obvious when the task boundary is clear (i.e. the sentence starts with [TASK]). [TASK] can tighten the bond between "*BQC*" and "*A*".

Applying AQF improves LAMOL$_G$ by 1.2 percentage points, which is less than LAMOL $_T$. It is because the generated pseudo data is still biased toward new tasks due to the unclear task boundary (the sentence begins with the task-independent token [GEN]). However, thanks to the *A* being next to the *C* in the "*BQCA*" format, it is still capable of learning from a small amount of pseudo data.

For LAMOL$_R$, AQF can enhance the model by 1.4 percentage points, which is only 0.1% behind multi-task learning. LAMOL$_R$ uses the real data of the old task instead of the generated pseudo data when learning new tasks, so this further demonstrates that the "*BQCA*"format proposed by AQF is more conducive to lifelong language learning: it can use *Q* to help the model pay more attention to the important information in *C*, then further makes the generation of *A* more accurate.

For L2KD, the application of AQF improves the model by one percentage point, demonstrating that AQF is still applicable to the previous LAMOL-based enhanced approaches and is capable of effectively applied to other LAMOL-based studies.

For MTL, AQF can bring an improvement of 1 percentage point. At the same time, the improvement of each baseline demonstrates that the improvement brought by AQF is comprehensive: it is not only conducive to alleviating catastrophic forgetting but also improves the upper bound of the model. Therefore, we believe that AQF has the potential to apply LLL to real-world scenarios.

6 Effectiveness of the Replay Questions

Since the Replay QuestionsRQ is proposed based on AQF, we verify and analyze the effectiveness of the RQ based on Table 3 and 5. The AQF-RQ performance in Table 3 subtracts the corresponding LAMOL+AQF performance in Table 5 is the gains obtained by AQF-RQ come from

the RQ step. Therefore, $RQ_G^{0.2}$ = AQF-$RQ_G^{0.2}$ - LAMOL+$AQF_G^{0.2}$ = 1.1, and $RQ_T^{0.2}$ = AQF-$RQ_T^{0.2}$ - LAMOL+$AQF_T^{0.2}$ = 0.7. It demonstrates that the RQ step based on the AQF can help the model generate more task-corresponding pseudo samples by making $P(Q_i|B_i) > P(Q_{j\neq i}|B_i)$.

7 Conclusion and Future Work

This work proposed AQF-RQ which is a simple yet efficient and robust lifelong language learning method. We propose a new question-first data format that is train-efficient without additional computational resources and new parameters. In AQF's "*BQCA*" format, generating *Q* first makes it easier to generate task-corresponding pseudo-data, and *A* is more accurate because *A* is next to *C*. In addition, RQ can strengthen the model's attention to the problem, so that the model has a greater probability of generating task-corresponding questions, and further generates more task-corresponding pseudo-data. AQF-RQ effectively alleviates catastrophic forgetting, only 0.36% lower than multi-task learning. Due to a lack of computing resources, we did not experiment on larger datasets and longer task orders, which we leave as future work. In addition, we will investigate ways to enhance performance when the task boundary is unclear.

References

Leanne Bowler. 2010. Talk as a metacognitive strategy during the information search process of adolescents. *Inf. Res.*, 15(4).

David W. Braithwaite and Lauren Sprague. 2021. Conceptual knowledge, procedural knowledge, and metacognition in routine and nonroutine problem solving. *Cogn. Sci.*, 45(10).

Shiyu Chang, Yang Zhang, Mo Yu, and Tommi S. Jaakkola. 2020. Invariant rationalization. In *Proceedings of the 37th International Conference on Machine Learning, ICML 2020, 13-18 July 2020, Virtual Event*, volume 119 of *Proceedings of Machine Learning Research*, pages 1448–1458. PMLR.

Zhiyuan Chen and Bing Liu. 2018. *Lifelong Machine Learning, Second Edition*. Synthesis Lectures on Artificial Intelligence and Machine Learning. Morgan & Claypool Publishers.

Yung-Sung Chuang, Shang-Yu Su, and Yun-Nung Chen. 2020. Lifelong language knowledge distillation. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 2914–2924, Online. Association for Computational Linguistics.

John H. Flavell. 1979. Metacognition and cognitive monitoring: A new area of cognitive-developmental inquiry. *American Psychologist*, 34:906–911.

Robert M French. 1999. Catastrophic forgetting in connectionist networks. *Trends in cognitive sciences*, 3(4):128–135.

Luheng He, Kenton Lee, Mike Lewis, and Luke Zettlemoyer. 2017. Deep semantic role labeling: What works and what's next. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 473–483.

Neil Houlsby, Andrei Giurgiu, Stanislaw Jastrzebski, Bruna Morrone, Quentin de Laroussilhe, Andrea Gesmundo, Mona Attariyan, and Sylvain Gelly. 2019. Parameter-efficient transfer learning for NLP. In *Proceedings of the 36th International Conference on Machine Learning, ICML 2019, 9-15 June 2019, Long Beach, California, USA*, volume 97 of *Proceedings of Machine Learning Research*, pages 2790–2799. PMLR.

Yufan Huang, Yanzhe Zhang, Jiaao Chen, Xuezhi Wang, and Diyi Yang. 2021. Continual learning for text classification with information disentanglement based regularization. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT 2021, Online, June 6-11, 2021*, pages 2736–2746. Association for Computational Linguistics.

Kasidis Kanwatchara, Thanapapas Horsuwan, Piyawat Lertvittayakumjorn, Boonserm Kijsirikul, and Peerapon Vateekul. 2021. Rational LAMOL: A rationale-based lifelong learning framework. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing, ACL/IJCNLP 2021, (Volume 1: Long Papers), Virtual Event, August 1-6, 2021*, pages 2942–2953. Association for Computational Linguistics.

Tianlin Liu, Lyle Ungar, and João Sedoc. 2019. Continual learning for sentence representations using conceptors. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT 2019, Minneapolis, MN, USA, June 2-7, 2019, Volume 1 (Long and Short Papers)*, pages 3274–3279. Association for Computational Linguistics.

Andrea Madotto, Zhaojiang Lin, Zhenpeng Zhou, Seungwhan Moon, Paul A. Crook, Bing Liu, Zhou Yu, Eunjoon Cho, Pascale Fung, and Zhiguang Wang. 2021. Continual learning in task-oriented dialogue systems. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing, EMNLP 2021, Virtual Event / Punta Cana, Dominican Republic, 7-11 November, 2021*, pages 7452–7467. Association for Computational Linguistics.

Bryan McCann, Nitish Shirish Keskar, Caiming Xiong, and Richard Socher. 2018. The natural language decathlon: Multitask learning as question answering. *arXiv preprint arXiv:1806.08730*.

Michael McCloskey and Neal J Cohen. 1989. Catastrophic interference in connectionist networks: The sequential learning problem. In *Psychology of learning and motivation*, volume 24, pages 109–165. Elsevier.

Fei Mi, Liangwei Chen, Mengjie Zhao, Minlie Huang, and Boi Faltings. 2020. Continual learning for natural language generation in task-oriented dialog systems. In *Findings of the Association for Computational Linguistics: EMNLP 2020, Online Event, 16-20 November 2020*, volume EMNLP 2020 of *Findings of ACL*, pages 3461–3474. Association for Computational Linguistics.

Jekaterina Novikova, Ondrej Dusek, and Verena Rieser. 2017. The E2E dataset: New challenges for end-to-end generation. In *Proceedings of the 18th Annual SIGdial Meeting on Discourse and Dialogue, Saarbrücken, Germany, August 15-17, 2017*, pages 201–206. Association for Computational Linguistics.

Alec Radford, Rafal Jozefowicz, and Ilya Sutskever. 2017. Learning to generate reviews and discovering sentiment. *arXiv preprint arXiv:1704.01444*.

Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. 2019. Language models are unsupervised multitask learners. *OpenAI blog*, 1(8):9.

Mark B. Ring. 1997. CHILD: A first step towards continual learning. *Mach. Learn.*, 28(1):77–104.

Fan-Keng Sun, Cheng-Hao Ho, and Hung-Yi Lee. 2019. Lamol: Language modeling for lifelong language learning. *Proceedings of the ICLR 2020*.

Jingyuan Sun, Shaonan Wang, Jiajun Zhang, and Chengqing Zong. 2020. Distill and replay for continual language learning. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 3569–3579, Barcelona, Spain (Online). International Committee on Computational Linguistics.

Han Wang, Ruiliu Fu, Chengzhang Li, Xuejun Zhang, Jun Zhou, Xing Bai, Yonghong Yan, and Qingwei Zhao. 2022. Reminding the incremental language model via data-free self-distillation. *Applied Intelligence*, pages 1–23.

Tsung-Hsien Wen, Milica Gasic, Nikola Mrksic, Peihao Su, David Vandyke, and Steve J. Young. 2015. Semantically conditioned lstm-based natural language generation for spoken dialogue systems. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing, EMNLP 2015, Lisbon, Portugal, September 17-21, 2015*, pages 1711–1721. The Association for Computational Linguistics.

hyperparameter	**value**
optimizer	AdamW
adam epsilon	1×10^{-4}
learning rate	1×10^{-4}
weight of RQ task	$\eta = 0.2$
weight of LM task	$\lambda = 0.25$
max gradient norm	1.0
learning rate schedule	warmup linear
warmup ratio	0.005
max length	1024
top-k sampling	k=20

Table 6: The main hyperparameters in our experiment.

A Datasets

As we can not obtain test set of SQuAD since it is hidden from the host, we use development set for testing. For other tasks, we use the corresponding test set. A normalized F1 (nF1) metric that lower text and remove punctuation and articles, is used to evaluate SQuAD and SRL. The exact match (EM) is used to evaluate SST. The turn-based dialogue state EM (dsEM) is used for WOZ. The ROUGE is used to evaluate E2ENLG and RNNLG (rest/hotel/tv/laptop)The size of each dataset is detailed in Table 7.

As shown in Table 7, SST, WOZ, E2ENLG and RNNLG (rest/hotel/tv/laptop) only have one question. The SQuAD has mainly 12 types of questions and other countless types of questions. SRL has 6 types of questions.

B Hyperparameter

The main hyperparameters in our experiments are detailed in Table 6.

C Sequence Generation for Different Domains

We conducted experiments on five sequence generation tasks for different domains by comparing LAMOL$_T$, LAMOL$_G$, and L2KD with AQF-RQ. Experiments are conducted with sufficient pseudo data (i.e. γ=0.2). Following L2KD, we select the task order from hard to easy: E2ENLG-RNNLG (rest) - RNNLG (hotel) - RNNLG (tv) - RNNLG (laptop). As shown in the Fig.3, we can observe that all models perform well except for Finetune and LAMOL$_G$. However, AQF-RQ can still outperform other baselines. When learning the same

Dataset	#Train	#Test	Metric	Question Type	Question
different types of tasks					
SQuAD	87599	10570	nF1	12+∞	what/who/whose/whom/when/ where/how/why/which/if/do/is ... + other countless types of questions
SST	6920	1821	EM	1	is this review negative or positive?
SRL	6414	2201	nF1	6	what/who/whose/whom/ when/where/how/why ...
WOZ	2536	1646	dsEM	1	what is the change in state?
suquence generation for different domains					
E2ENLG	6000	2000			
RNNLG (rest)	6228	1039			
RNNLG (hotel)	6446	1075	ROUGE	1	what is the natural language form?
RNNLG (tv)	8442	1407			
RNNLG (laptop)	7944	2649			

Table 7: The summarized results for datasets.

type of tasks in different domains, first we expect that previous tasks can promote the learning of new tasks. This is known as the forward transfer. In terms of the forward transfer, AQF-RQ is better at acquiring knowledge from previous tasks that is beneficial to new tasks. The second is backward transfer, which refers to learning new tasks while consolidating and strengthening previous tasks. It can be observed that AQF-RQ also has better backward transfer. For example, from the fourth task RNNLG.tv on, the backward transfer of AQF-RQ on the previous tasks is higher than other baselines. As the number of learned tasks increases, the unclear task boundary gradually becomes significantly weaker than the clear task boundary. For LAMOL$_G$, the performance on E2ENLG begins to drop significantly from the fourth task RNNLG.tv. But our AQF-RQ$_G$ remains on an upward trend. According to the above analysis, it can be concluded that AQF-RQ has stronger forward transfer and backward transfer capabilities in different domains of the same type of task.

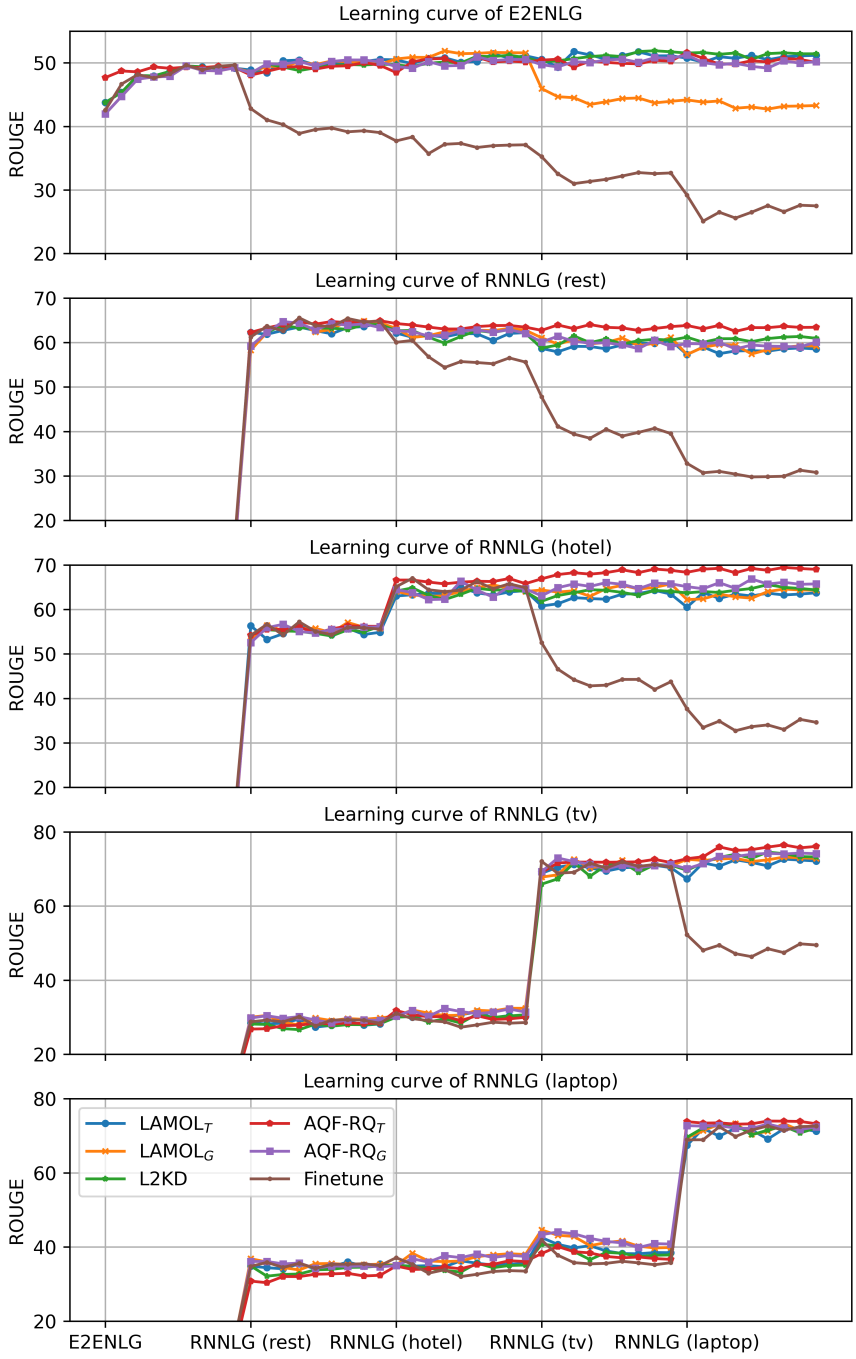

Figure 3: Results on sequence generation tasks for five different domains when $\gamma = 0.2$.

DOUBLEMIX: Simple Interpolation-Based Data Augmentation for Text Classification

Hui Chen[♣] **Wei Han**[♣] **Diyi Yang**[♢] **Soujanya Poria**[♣]
[♣] Singapore University of Technology and Design
[♢] Georgia Institute of Technology
{hui_chen, wei_han}@mymail.sutd.edu.sg
diyi.yang@cc.gatech.edu
sporia@sutd.edu.sg

Abstract

This paper proposes a simple yet effective interpolation-based data augmentation approach termed DOUBLEMIX, to improve the robustness of models in text classification. DOUBLEMIX first leverages a couple of simple augmentation operations to generate several perturbed samples for each training data, and then uses the perturbed data and original data to carry out a two-step interpolation in the hidden space of neural models. Concretely, it first mixes up the perturbed data to a synthetic sample and then mixes up the original data and the synthetic perturbed data. DOUBLEMIX enhances models' robustness by learning the "shifted" features in hidden space. On six text classification benchmark datasets, our approach outperforms several popular text augmentation methods including token-level, sentence-level, and hidden-level data augmentation techniques. Also, experiments in low-resource settings show our approach consistently improves models' performance when the training data is scarce. Extensive ablation studies and case studies confirm that each component of our approach contributes to the final performance and show that our approach exhibits superior performance on challenging counterexamples. Additionally, visual analysis shows that text features generated by our approach are highly interpretable. Our code for this paper can be found at https://github.com/declare-lab/DoubleMix.git.

1 Introduction

Deep neural networks have enabled breakthroughs in most supervised settings in natural language processing (NLP) tasks. However, labeled data in NLP is often scarce, as linguistic annotation usually costs large amounts of time, money, and expertise. With limited training data, neural models will be vulnerable to overfitting and can only capture shallow heuristics that succeed in limited scenarios, which will lead to severe performance degradation when applied to challenging situations.

In order to improve the robustness of models, various data augmentation methods have been proposed. Generally, there are three types of augmentation techniques: token-, sentence-, and hidden-level transformation. Wei and Zou (2019) summarized several common token-level transformations, including word insertion, deletion, replacement, and swap. Sentence-level transformation is to paraphrase a sentence through specific grammatical or syntactic rules. Back-translation (Sennrich et al., 2016; Edunov et al., 2018) is a typical sentence-level augmentation method where a sentence is translated to an intermediate language and then translated back to obtain augmented samples. Additionally, for natural language inference (NLI) tasks that identify whether a premise entails, contradicts, or is neutral with a hypothesis, Min et al. (2020) studied syntactic rules of sentences in inference tasks and proposed several syntactic transformation techniques such as *Inversion* and *Passivization* to construct syntactically informative examples. However, these methods often have high requirements for sentence structures. It is hard to obtain a large number of augmented samples by this method.

In recent years, several hidden-level augmentation methods are proposed and they have exhibited superior performance in a number of popular text classification tasks. TMix (Chen et al., 2020) is a typical approach where a linear interpolation is performed in the hidden space of transformer models such as BERT (Devlin et al., 2019). The main idea of TMix (Verma et al., 2019) comes from Mixup, a method that is based on the principle of Vicinal Risk Minimization (VRM) (Chapelle et al., 2001) and has achieved substantial improvements in computer vision tasks (Verma et al., 2019; Hendrycks et al., 2020; Kim et al., 2020; Ramé et al., 2021) and natural language tasks (Guo et al., 2019; Chen et al., 2020; Kim et al., 2021; Park and Caragea, 2022). Recently, SSMix (Yoon et al., 2021) which interpolates text based on the saliency

Proceedings of the 29th International Conference on Computational Linguistics, pages 4622–4632
October 12–17, 2022.

of tokens (Simonyan et al., 2014) in hidden space has been introduced. These methods make models learn a mapping from a mixed text representation to an intermediate label which is generated by linearly combining two different source labels. However, the intermediate soft label cannot always accurately describe the true probability of classes that the mixed text representations belong to, which limit the effectiveness of augmentation.

To overcome these limitations, this work proposes a simple yet effective interpolation-based data augmentation method termed DOUBLEMIX, which performs interpolation in the hidden space and does not require label mixing. Firstly, we leverage a collection of simple augmentation operations to generate several perturbed samples from the raw data and then mix up these perturbed samples. Secondly, we mix up the original data with the synthesized perturbed data. We constrain the mixing weight of the original to be larger than the synthesized perturbed data to balance the trade-off between proper perturbations and the potential injected noise. To stabilize the training process, we add a Jensen-Shannon divergence regularization term to our training objective to minimize the distance between the predicted distributions of the original data and the perturbed variants.

To demonstrate the effectiveness of our approach, we conduct extensive experiments by comparing our DOUBLEMIX with previous state-of-the-art data augmentation methods on six popular text classification benchmark datasets. Additionally, we reduce and vary the amount of training data, to observe if DOUBLEMIX can consistently improve over the baselines. We further conduct ablation studies and case studies to investigate the impact of different training strategies on DOUBLEMIX's effectiveness and whether our method works on challenging counterexamples. Moreover, we visualize the features generated by DOUBLEMIX to interpret why our method works. Experimental results and analyses confirm the efficacy of our proposed approach and every component in DOUBLEMIX contributes to the performance. To sum up, our contributions are:

- We propose a simple interpolation-based data augmentation approach DOUBLEMIX to improve the robustness of neural models in text classification by mixing up the original text with its perturbed variants in hidden space.
- We demonstrate the effectiveness of DOUBLEMIX through extensive experiments and analyses on six text classification benchmarks as well as three low-resource datasets.
- We qualitatively analyze why our method works by visualizing its data manifold and quantitatively analyze how our method works by conducting several ablation studies and case studies.

2 Related Work

Data augmentation techniques are widely employed in NLP tasks to improve the robustness of models (Sun et al., 2020; Xie et al., 2020; Cheng et al., 2020; Guo et al., 2020; Kwon and Lee, 2022). One way to enrich the original training set is to perturb the tokens in each sentence. For example, Wei and Zou (2019) introduced a set of simple data augmentation operations such as synonym replacement, random insertion, swap, and deletion. However, token-level perturbation sometimes does not guarantee that the augmented sentences are grammatically correct.

Thus, sentence-level augmentation methods are introduced, where people paraphrase the sentence by some specific rules. Minervini and Riedel (2018); McCoy et al. (2019) leveraged syntactic rules to generate adversarial examples in inference tasks. Moreover, Andreas (2020) investigated the compositional inductive bias in sequence models and augmented data by compositional rules. However, these methods require careful design, and they are often customized for a specific task, which makes them hard to generalize to different datasets.

Recently, a couple of hidden-level augmentation techniques which perform interpolation in hidden space have been studied (Guo et al., 2019; Verma et al., 2019; Hendrycks et al., 2020). Inspired by PuzzleMix (Kim et al., 2020) and SaliencyMix (Uddin et al., 2020) which is popular in computer vision, Yoon et al. (2021) proposed SSMix which utilizes the saliency information of spans (Simonyan et al., 2014) in each sentence to interpolate in hidden space to create informative examples. Yin et al. (2021) interpolate hidden states of the entire mini-batch to obtain better representations. Inspired by the prior work, our DOUBLEMIX aims at improving models' robustness by mixing up text features and their perturbed samples in hidden space.

3 Proposed Method: DOUBLEMIX

To regularize NLP models in a more efficient manner, we introduce a simple yet effective data augmentation approach, DOUBLEMIX, that enhances the representation of each training data by learning the features sampled from a region constructed by the original sample itself and its perturbed samples. The perturbed samples are generated by simple token- or sentence-level augmentation operations. DOUBLEMIX is a hidden-level regularization technique and our base model is a pre-trained transformer network, as they have achieved great performance in various NLP tasks. Algorithm 1 shows the training process of our approach.

3.1 Robust Interpolation in Hidden Space

For an input sequence $x = \{w_0, w_1, ..., w_S\}$ with S tokens associated with a label y, our goal is to predict a label of this sequence. At the beginning of this approach, we prepare a perturbation operation set containing simple token- and sentence-level data augmentation techniques such as back-translation (sentence-level), synonym replacement (token-level), and Gaussian noise perturbation (sentence-level). Thereafter, we randomly sample the operations N times and use the selected augmentation operations to generate N perturbed samples of each training instance. Note that each type of operation can be selected multiple times. We generated different perturbed samples by adjusting the hyper-parameters. For example, if we select synonym replacement, we can produce different perturbations by adjusting the proportion of tokens to be substituted. For back-translation, we can use different intermediate languages.

Our approach is performed in hidden space, so as to encourage the model to fully utilize the hidden information within the multi-layer networks. We employ a pre-trained model $f(;\theta)$ containing L layers to encode the text to hidden representations. Then we select a layer i which ranges in $[0, L]$ to interpolate. At the i-th layer of $f(;\theta)$, a two-step interpolation is performed where the first step is to mix up all the perturbed samples by a group of weights sampled from Dirichlet distribution, and the second step is to mix up the synthesized perturbed sample and the original sample by some weights $\in [0, 1]$ sampled from Beta distribution. We follow Zhang et al. (2018) to use Beta distribution for weight sampling, and Dirichlet distribution is a multi-variate Beta distribution. Note that when we mix up the original data and the synthesized perturbed data, we constrain the mixing weight of the original data to be larger, so as to make the final perturbed representation to be close to the original one. This balances the trade-off between proper perturbation and potential injected noise. After the two-step interpolation, the synthesized hidden presentation is fed to the remaining layers $f_{[i:L]}(;\theta)$ and a classifier.

3.2 Training Objectives

During the training period, we do not directly minimize the Cross-Entropy loss of the probability distribution of the synthesized sample, as it may introduce too much noise. We employ a consistency regularization term, Jensen-Shannon Divergence (JSD) loss (Bachman et al., 2014; Zheng et al., 2016; Hendrycks et al., 2020), to minimize the difference between the prediction distribution of synthetic data and original data, and meanwhile, we minimize the Cross-Entropy loss of the model output of original data and the gold label. The training objective can be written as:

$$\begin{aligned}\mathcal{L}(y|x, x_a, ..., x_N) &= \mathcal{L}_{CE}(y|x) \\ &+\gamma\mathcal{L}_{JSD}(y|x, x_a, ..., x_N) \end{aligned} \quad (1)$$

where γ is a hyper-parameter.

In the consistency regularization, we do not employ Kullback-Leibler divergence (KL) because it is not symmetric, i.e., $\text{KL}(P||Q) \neq \text{KL}(Q||P)$ when $P \neq Q$. It is not a promising choice to measure the similarity of p_{mix} and p_{orig} using KL, as neither p_{mix} nor p_{orig} are true predictions and we deem that they share equal status. JSD provides a smoothed and normalized version of KL divergence, with scores between 0 (identical) and 1 (maximally different). We believe using such a symmetric metric can make the training more stable.

3.3 Why does DOUBLEMIX work?

To further discuss why our method works, we visualize the original data and the sample space of synthesized data in Mixup and DOUBLEMIX in Fig. 1. For brevity, we assume the number of perturbed samples in Step I is two. As shown in Fig. 1, blue dots indicate training data, and orange dots are perturbed data generated by our selected operations. Synthesized data in Mixup (Zhang et al., 2018) can only be created along a line, such as the blue full line connecting the two points X and X' since it

Algorithm 1: DOUBLEMIX

Input: Model $f(;\theta)$ containing L layers, the l-th layer of the model $f_l(;\theta)$, classifier $\hat{p}(;\phi)$, training set $\mathcal{X} = \{(x_1, y_1), ..., (x_n, y_n)\}$, perturbation operation set O = {back-translation (BT), synonym replacement (SR), ..., Gaussian noise (GN)}, interpolation layer set $I = \{i_1, ..., i_k\}$, number of augmented samples N, number of training epochs K, the global learning rate η, Beta distribution hyper-parameter α, Dirichlet distribution hyper-parameter τ, loss hyper-parameter γ

Output: Updated network weights θ, ϕ

$\{o_1, ..., o_N\} \leftarrow O$ ▷ Select N operations. Each type of operation can be selected multiple times.

for $k \leftarrow 0$ **to** K **do**

  for $(x, y) \in \mathcal{X}$ **do**

    $\{x_a, ..., x_N\} \leftarrow \{o_1, ..., o_N\}(x)$ ▷ Apply the selected operations to generate N different augmented samples of x.

    $i \leftarrow I = \{i_1, ..., i_k\}$ ▷ Randomly select an interpolation layer from Set I

    $h^i_{orig} \leftarrow f_{[0:i]}(x; \theta)$

    $\{h^i_a, ..., h^i_N\} \leftarrow f_{[0:i]}(\{x_a, ..., x_N\}; \theta)$ ▷ Encode $\{x, x_a, ..., x_N\}$, and interpolate at the i-th layer.

    Sample $(w_1, ..., w_n) \sim \text{Dirichlet}(\tau, ..., \tau)$

    $h^i_{aug} \leftarrow w_1 \cdot h^i_a + ... + w_n \cdot h^i_N$ ▷ First mixup (*Step I*).

    Sample $\lambda \sim \text{Beta}(\alpha, \alpha)$

    $\lambda \leftarrow \max(\lambda, 1 - \lambda)$ ▷ Constrain the synthetic data to a region closer to the original example.

    $h^i_{mix} \leftarrow \lambda \cdot h^i_{orig} + (1 - \lambda) \cdot h^i_{aug}$ ▷ Second mixup (*Step II*).

    $h_{mix} \leftarrow f_{[i+1:L]}(h^i_{mix}; \theta)$

    $p_{mix} \leftarrow \hat{p}(y|h_{mix}; \phi)$

    $p_{orig} \leftarrow \hat{p}(y|f(x; \theta); \phi)$

    $\bar{p} \leftarrow \frac{1}{2}(p_{mix} + p_{orig})$

    $\mathcal{L}_{JSD}(y|x, x_a, ..., x_N) \leftarrow \frac{1}{2}(\text{KL}(p_{mix}||\bar{p}) + \text{KL}(p_{orig}||\bar{p}))$

    $\mathcal{L} \leftarrow \mathcal{L}_{CE}(y|x) + \gamma\mathcal{L}_{JSD}(y|x, x_a, ..., x_N)$

    $\theta \leftarrow \theta - \eta\nabla_{\theta,\phi}\mathcal{L}; \phi \leftarrow \phi - \eta\nabla_{\theta,\phi}\mathcal{L}$ ▷ Update the network weight θ of the base model f and ϕ of the classifier ϕ.

  end

end

is a simple linear combination. Hence, Mixup enforces the regularization to behave linearly among the training data. In contrast to Mixup, the sample space of synthesized data of DOUBLEMIX is a polygon. In this example, $\triangle XAB$ is the sample space of the synthesized data in DOUBLEMIX. Firstly, Step I samples a point P on the line connecting X_a and X_b. Secondly, Step II finds a point Q on the line connecting X and P. Note that Q should be closer to X or at the middle of Line XP, as in Step II, we constrain the mixing weight of original data to be larger than that of synthesized perturbed data. Taken together, our approach enforces the model to learn nearby features for each training data so that it is robust to representation shifts.

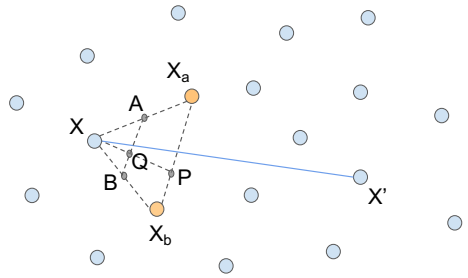

Figure 1: Visualization of the sample space of synthesized data in Mixup and DOUBLEMIX. Blue dots X and X' indicate two data points in the training set. Orange dots X_a and X_b are perturbed data of X. Grey dots A, B, P and Q are sampled points.

4 Experimental Setup

4.1 Datasets

We compare our approach with several data augmentation baselines on six text classification datasets, covering sentiment polarity classification, question type classification, humor detection, and natural language inference: IMDB (Maas et al., 2011) and SST-2 (Socher et al., 2013) which predict the sentiment of movie reviews to be positive or negative, 6-class open-domain question classification TREC (Li and Roth, 2002), Pun of the day (Puns) (Yang et al., 2015) which detects humor in a single sentence, and two inference datasets SNLI (Bowman et al., 2015) and MNLI (Williams et al., 2018) which identify whether a premise entails, contradicts or is neutral with a hypothesis. Statistics of the six text classification datasets can be found in Appendix A. As the test set of MNLI is not publicly available, we used the matched development set as our development set and the mismatched development set as our test set in our experiments.

4.2 Baselines

We compare our approach with several widely-used baseline models, including token-, sentence-, and

Method	SST-2		TREC		Puns	
	Acc.	F1.	Acc.	F1.	Acc.	F1.
BERT (Devlin et al., 2019)	$91.08_{0.1}$	$91.09_{0.1}$	$96.90_{0.4}$	$96.21_{0.7}$	$94.20_{0.2}$	$94.38_{0.2}$
+ Easy Data Augmentation (Wei and Zou, 2019)	$91.66_{0.2}$	$91.75_{0.1}$	$97.10_{0.3}$	$96.74_{0.2}$	$93.71_{0.6}$	$93.83_{0.5}$
+ Back Translation (Edunov et al., 2018)	$91.33_{0.2}$	$91.29_{0.1}$	$96.90_{0.4}$	$96.72_{0.1}$	$93.27_{0.4}$	$93.34_{0.4}$
+ Manifold Mixup (Verma et al., 2019)	$91.33_{0.4}$	$91.44_{0.3}$	$97.00_{0.3}$	$96.29_{0.5}$	$94.37_{0.9}$	$94.44_{1.0}$
+ TMix (Chen et al., 2020)	$91.13_{0.5}$	$91.29_{0.3}$	$96.90_{0.1}$	$96.32_{0.3}$	$94.21_{0.7}$	$94.28_{0.8}$
+ SSMix (Yoon et al., 2021)	$91.45_{0.5}$	$91.50_{0.2}$	$97.00_{0.2}$	$96.44_{0.2}$	$94.33_{0.7}$	$94.46_{0.6}$
+ DoubleMix (Ours)	$\mathbf{92.21}_{0.1}$	$\mathbf{92.11}_{0.1}$	$\mathbf{97.40}_{0.1}$	$\mathbf{97.32}_{0.6}$	$\mathbf{94.59}_{0.1}$	$\mathbf{94.66}_{0.1}$

Method	IMDB		SNLI		MNLI	
	Acc.	F1.	Acc.	F1.	Acc.	F1.
BERT (Devlin et al., 2019)	$83.52_{0.1}$	$83.65_{0.4}$	$90.31_{0.5}$	$90.28_{0.5}$	$84.16_{0.2}$	$84.05_{0.2}$
+ Easy Data Augmentation (Wei and Zou, 2019)	$83.43_{0.1}$	$83.85_{0.1}$	$90.27_{0.4}$	$90.28_{0.4}$	$84.04_{0.1}$	$83.97_{0.2}$
+ Back Translation (Edunov et al., 2018)	$83.69_{0.1}$	$84.11_{0.2}$	$90.21_{0.5}$	$90.23_{0.4}$	$84.50_{0.3}$	$84.44_{0.3}$
+ Syntactic Data Augmentation (Min et al., 2020)	-	-	$90.35_{0.3}$	$90.31_{0.4}$	$84.19_{0.3}$	$84.08_{0.2}$
+ Manifold Mixup (Zhang et al., 2018)	$83.63_{0.1}$	$83.81_{0.2}$	$90.04_{0.2}$	$90.02_{0.2}$	$83.37_{0.2}$	$83.31_{0.1}$
+ TMix (Chen et al., 2020)	$83.47_{0.3}$	$83.91_{0.2}$	$90.12_{0.1}$	$90.09_{0.1}$	$83.43_{0.1}$	$83.38_{0.1}$
+ SSMix (Yoon et al., 2021)	$83.55_{0.3}$	$83.88_{0.2}$	$90.21_{0.2}$	$90.14_{0.3}$	$83.66_{0.2}$	$83.54_{0.2}$
+ DoubleMix (Ours)	$\mathbf{84.14}_{0.5}$	$\mathbf{84.39}_{0.2}$	$\mathbf{91.03}_{0.1}$	$\mathbf{91.02}_{0.1}$	$\mathbf{84.72}_{0.2}$	$\mathbf{84.64}_{0.1}$

Table 1: Test accuracy (%) and F1 scores (%) for BERT when comparing our proposed DoubleMix with baseline methods on six text classification datasets. We randomly select two augmented samples to mix up in Step I. Best scores are marked in bold. As syntactic transformations (Min et al., 2020) are rule-based data augmentation techniques customized for inference tasks, we only show their performances on SNLI and MNLI. We report the mean accuracy and F1 scores across five different runs with the standard deviation shown in subscript (e.g., $91.08_{0.1}$ indicates 91.08 ± 0.1).

hidden-level augmentation techniques. Token-level baselines contain the operations in Easy Data Augmentation (EDA) (Wei and Zou, 2019) where they randomly insert, swap, and delete tokens in each sentence. Sentence-level baselines include paraphrasing sentences such as back-translation (Sennrich et al., 2016) and applying some syntactic rules to create augmented sentences such as syntactic transformation (Min et al., 2020). Hidden-level baselines include Manifold Mixup (Verma et al., 2019), TMix (Chen et al., 2020), and SSMix (Yoon et al., 2021), where they mix up two different training samples in hidden space and learn a mapping from the intermediate representation to an intermediate label. Details of the implementation of the baselines can be found in Appendix B.

5 Results and Analysis

We evaluate our baselines and proposed approach on six text classification benchmark datasets. We also show the performance of our approach in low-resource settings to confirm DoubleMix is efficient and robust when the training samples are scarce. This section will discuss the performance of models in detail and quantitatively analyze how and why DoubleMix works.

5.1 Main Results

Table 1 shows the performance of DoubleMix and the relevant baselines on six text classification datasets. Our base model is the BERT-base-uncased model. We observe that DoubleMix achieves the best average results compared to previous state-of-the-art baselines across six datasets, where DoubleMix shows the greatest improvements over BERT on SST-2, increasing the test accuracy and binary F1 score by 1.13% and 1.02%.

In addition, we find token-level Easy Data Augmentation and sentence-level Back Translation are not able to improve the BERT baseline on Puns and SNLI. Especially on Puns, Back Translation's test accuracy and F1 score are about 1% lower than BERT. This might be that labels in humor detection and inference tasks are closely related to the presence of some important words, and Easy Data Augmentation and Back Translation may perturb these words, making the true label flip, but the label learned by the model does not change accordingly, which leads to an inefficient learning process. Moreover, hidden-level augmentations such as Manifold Mixup, TMix, and SSMix fail to improve the base model on SNLI and MNLI. As subtle changes in the sentences in inference tasks

Method	SNLI				MNLI				Avg.
	1K (Acc./F1.)	2.5K (Acc./F1.)	5K (Acc./F1.)	10K (Acc./F1.)	1K (Acc./F1.)	2.5K (Acc./F1.)	5K (Acc./F1.)	10K (Acc./F1.)	(Acc./F1.)
BERT	69.77/69.59	76.10/75.92	79.28/79.25	82.36/82.28	55.81/54.61	65.63/65.15	71.24/71.01	74.24/74.14	71.80/71.49
+ BT	70.23/69.96	76.51/76.54	79.57/79.57	82.68/82.65	57.28/55.53	66.97/66.95	72.28/72.01	74.49/74.44	72.50/72.21
+ M-Mix	71.45/71.34	76.48/76.42	79.91/79.83	82.14/82.13	57.01/56.70	67.08/66.96	71.76/71.68	74.68/74.54	72.56/72.45
+ TMix	71.04/71.12	76.38/76.10	79.85/79.81	82.09/82.09	**57.31** /56.99	67.10/67.01	71.66/71.59	74.86/74.70	72.54/72.43
+ SSMix	71.32/71.21	76.87/76.72	80.02/79.93	82.41/82.20	57.25/ **57.10**	67.13/67.05	71.70/71.63	74.77/74.62	72.68/72.56
+ Ours	**71.82/71.72**	**77.43/77.42**	**80.75/80.72**	**83.18/83.26**	56.15/55.91	**67.57/67.33**	**72.35/72.15**	**75.07/74.97**	**73.04/72.94**
RoBERTa	78.11/77.95	82.47/82.30	83.17/83.36	85.58/85.50	70.23/70.45	75.50/75.53	79.01/79.04	81.00/80.98	79.38/79.39
+ BT	78.32/78.25	82.47/82.61	84.23/84.08	86.07/86.05	70.67/70.52	**77.58/77.39**	79.17/79.07	81.26/81.08	79.97/79.88
+ M-Mix	79.32/78.73	82.71/82.63	84.60/84.63	86.00/85.95	71.78/71.04	76.00/75.96	79.43/79.34	81.40/81.28	80.16/79.95
+ TMix	79.17/79.11	82.84/82.91	85.09/85.13	86.16/86.13	**72.05/72.18**	76.57/76.44	79.92/79.80	81.30/81.17	80.39/80.36
+ SSMix	79.43/79.35	82.91/82.88	85.33/85.36	86.32/86.28	71.96/71.88	76.44/76.38	79.86/79.77	81.25/81.22	80.44/80.39
+ Ours	**80.41/80.31**	**83.96/83.92**	**85.42/85.42**	**86.91/86.88**	71.20/71.15	77.12/76.96	**80.43/80.28**	**82.46/82.24**	**80.99/80.90**

Table 2: Test accuracy (%) and F1 score (%) comparison on the SNLI and MNLI datasets training with varying amounts of training data (1000, 2500, 5000, and 10000). Best scores are marked in bold in yellow background. BT and M-Mix represent Back Translation and Manifold Mixup. We only use BT operations in DOUBLEMIX in this experiment.

will flip the true label, mixing up two different samples and learning an intermediate representation in hidden space cannot ensure the learned soft label is the true label of the intermediate representation.

In contrast, our model shows consistent improvements over BERT on these datasets. The consistent improvements indicate that, by strategically mixing up samples with similar meanings in the hidden space, DOUBLEMIX not only helps pre-trained models to become insensitive to feature perturbations in an effective way but also injects less potential noise during the augmentation process compared to other baselines.

5.2 Performance in Low-Resource Settings

To investigate the performance of our approach in low-resource settings, we randomly sample 1000, 2500, 5000, and 10000 examples from the original training data of SNLI and MNLI to construct our training sets for low-data setting evaluations, while the size of the development and test sets is unchanged. Apart from the BERT-base-uncased model (Devlin et al., 2019), we also conduct experiments on the RoBERTa-base model (Liu et al., 2019).

Table 2 presents the results in low-data settings. Compared with the BERT baseline in Table 1, we can observe that although pre-trained language models are powerful across text classification tasks, the test accuracy and F1 scores might decrease a lot when the training data is very scarce. DOUBLEMIX consistently improves the base model with no data augmentation on both SNLI and MNLI and outperforms all the baselines on the SNLI dataset. On MNLI, we observe that our method always achieves the top performance except in the 1K training samples. As the training set grows larger, our model gradually outperforms the baselines and the leading gap keeps expanding—when the number of training samples reaches 10K, our model can achieve at least 1% higher accuracy and F1 score than RoBERTa on both SNLI and MNLI.

5.3 Ablation Studies

5.3.1 Training Strategies in DOUBLEMIX

We also conduct a series of ablative experiments to examine the contribution of individual components. The results are displayed in Table 3, where our experiments are conducted on the SNLI dataset with only 1000 training samples. We find the performance drops after changing the training strategies, suggesting that the current interpolation method trained with JSD loss in DOUBLEMIX contributes to the final performance.

Concretely, we first remove the JSD loss in our training objective to check if this loss contributes to the performance. We observe that the accuracy and F1 score drop approximately 0.7% and 0.9% after removing the JSD loss, which manifests that JSD loss is capable of stabilizing the training process. Secondly, we merge the two steps and see how the model performs, where we use a Dirichlet distribution to sample $N + 1$ mixing weights for the original example and other N augmented examples, and mix up them at a time. In this case, the test accuracy and F1 score drop to 71.11% and 70.96%, respectively. This indicates that the two-step interpolation where we constrain the synthetic data to a region closer to the original sample, as mentioned in Line 11 in Algorithm 1 will inject less

noise. Moreover, we have also tried different mixing samples Step II. We find mixing up with another randomly selected training sample in Line 12 in Algorithm 1 results in a 0.59% accuracy decrease and a 0.70% F1 score decrease. If the selected training sample is restricted to the same category as the original data, the performance degradation will be even larger. This outcome may be caused by the larger semantic difference between the selected example and the original data compared to the augmented examples of the original data.

Method	Acc.	F1.
DOUBLEMIX	71.82	71.72
- w/o JSD loss	71.15	70.87
- merge Step II and Step I	71.11	70.96
- mix with another training sample in Step II	71.23	71.02
- mix with another same-class sample in Step II	70.98	70.74

Table 3: Test accuracy (%) and F1 scores (%) on the SNLI dataset with 1000 training samples after changing different parts of DOUBLEMIX.

Mixup layer set	Acc.	Δ_{Acc}	F1.	Δ_{F1}
$\emptyset$	69.77		69.59	
{0}	71.13	+1.36	71.02	+1.43
{0,1,2}	70.95	+1.18	70.84	+1.25
{3,4}	71.56	+1.79	71.48	+1.89
{3,6,9}	71.24	+1.47	71.16	+1.57
{7,9,12}	71.30	+1.53	71.19	+1.60
{9,10,12}	**71.82**	**+2.05**	**71.72**	**+2.13**
{3,4,6,9,10,12}	70.90	+1.13	70.90	+1.31
{3,4,6,7,9,12}	71.29	+1.52	71.16	+1.57

Table 4: Test accuracy (%) and F1 scores (%) on SNLI with 1000 training data with different interpolation layer sets. $\emptyset$ means no interpolation, and {0} is the input space. Δ indicates the gap to the baseline with no augmentation.

5.3.2 Effect of Interpolation Layers

We believe the hidden layers in pre-trained language models are powerful in representation learning, and interpolation in the hidden space can yield a larger performance improvement than in the input space. In this section, we will investigate which interpolation option in terms of the set of layers in pretrained models can obtain the best performance. Previous work Jawahar et al. (2019) indicates that BERT's intermediate layers {3, 4} perform best in encoding surface features and layers {6, 7, 9, 12} contain the most syntactic features and semantic features. We refer to Jawahar et al. (2019) to formulate several sets of layers and have conducted a couple of additional experiments on BERT + DOUBLEMIX with different sets of interpolation layers on SNLI with 1000 training samples to see which subsets give the optimal performance. The results are shown in Table 4.

Model	Original and Counterfactual Examples	P.	T.
BERT	*P*: Students are inside of a lecture hall. *H*: Students are indoors.	N	E
Ours	*P*: Students are inside of a lecture hall. *H*: Students are indoors.	E	E
BERT	*P*: Students are inside of a lecture hall. *H*: Students are on the soccer field.	C	C
Ours	*P*: Students are inside of a lecture hall. *H*: Students are on the soccer field.	C	C
BERT	*P*: Man in green jacket with baseball hat on. *H*: The man is not wearing a hat.	C	C
Ours	*P*: Man in green jacket with baseball hat on. *H*: The man is not wearing a hat.	C	C
BERT	*P*: Man in green jacket with baseball hat on. *H*: The man is at a baseball game.	E	N
Ours	*P*: Man in green jacket with baseball hat on. *H*: The man is at a baseball game.	N	N

Table 5: Predictions of BERT and our method on original examples and their counterfactual examples on the SNLI dataset. The counterfactual examples are extracted from Kaushik et al. (2020) and are constructed by substituting entities or adding details to entities. The examples in the yellow background are counterexamples. **P.** and **T.** represent prediction and true label. E, N and C are *entailment*, *neutral*, and *contradiction*. *P* and *H* are premise and hypothesis. Labels in red are wrong predictions while labels in blue are correct predictions.

When all the interpolation steps are excluded, the test accuracy is 69.77% and the F1 score is 69.59%. When we interpolate in the input space (the 0-th layer), the accuracy and F1 score increase by about 1.3% and 1.4%, showing that interpolation contributes to the performance. When we perform interpolations at layer set {0, 1, 2} with lower layers, the accuracy and F1 increases are smaller than interpolating in input space. However, when we interpolate at some middle layers such as {3, 4} and {3, 6, 9}, the performance improvement is more significant. According to Jawahar et al. (2019), the 9-th layer captures most of the syntactic and semantic information. We have tried several layer sets containing the 9-th layer and find {9, 10, 12} containing upper layers performs best. At the same time, we notice that the number of layers in the interpolation layer set is not the more the bet-

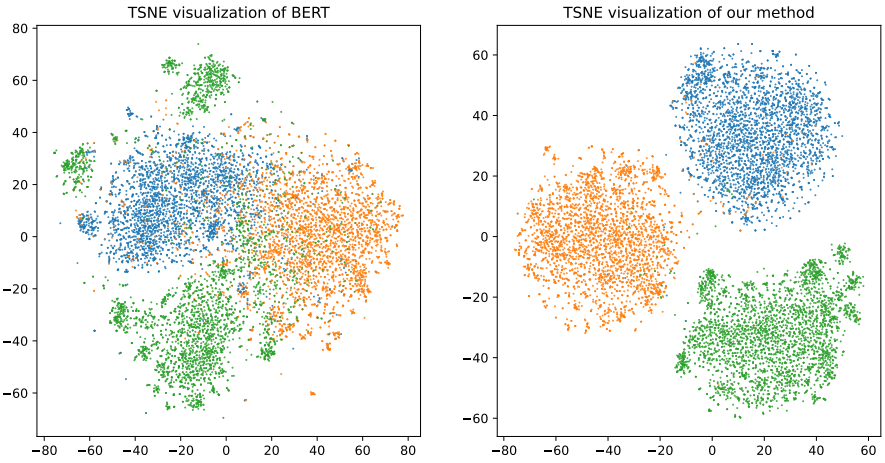

Figure 2: T-SNE projection of the features generated by the 12-layer encoder of BERT baseline (left) and BERT + DOUBLEMIX (right) on SNLI with 10000 training samples. The visualized features of the augmented text are extracted from the last layer of the base model during testing.

ter. The performance of {3,4,6,9,10,12} is only 70.90%, which is lower than any other layer set, and the performance of {3,4,6,7,9,12} is not the best, indicating that too many interpolation layers will reduce the efficiency of representation learning.

5.4 Case Studies

To further understand how DOUBLEMIX works, we randomly pick up some examples from the SNLI dataset and check the discrepancy between the predictions obtained from BERT and our method. Predictions of BERT and DOUBLEMIX are shown in Table 5. We find in those examples with "*contradiction*" label, both BERT and DOUBLEMIX can accurately predict the true label. Additionally, to investigate how our method behaves in challenging scenarios, we also test on the counterfactual version (Kaushik et al., 2020) of our selected samples. Both models excel in detecting negative words "*not*" and location names. However, when the ground truth is "*entailment*" or "*neutral*", DOUBLEMIX is more likely to make correct predictions. When the premise and hypothesis contain some common words (e.g., *Man in green jacket with **baseball** hat on. The man is at a **baseball** game.*), DOUBLEMIX inclines to make more accurate predictions.

5.5 Manifold Visualization

Finally, we visualize the embedding vectors generated by the 12-layer encoder of the BERT baseline with no data augmentation (BERT) and with DOUBLEMIX to qualitatively show the effectiveness of our approach in facilitating the model to learn robust representations in Fig. 2. Our experiments are conducted on SNLI with 10000 training samples. The visualized features are extracted from the output of the last layer of the model. We employ t-SNE (Van der Maaten and Hinton, 2008) which is implemented by the python package scikit-learn[1] to visualize the features. In Fig. 2, there are three clusters with three colors indicating different classes. We observe that the features in DOUBLEMIX are better separated than those in BERT, indicating that our method effectively improves robustness by encouraging the model to learn nearby features of each training sample.

6 Conclusion

In this work, we present a simple interpolation-based data augmentation approach DOUBLEMIX to improve models' robustness on a wide range of text classification datasets. DOUBLEMIX first leverages simple augmentation operations to generate perturbed data of each training sample and then performs a two-step interpolation in the hidden space of models to learn robust representations. Our approach outperforms several popular data augmentation methods on six benchmark datasets and three low-resource datasets. Finally, ablation studies, case studies, and visualization of manifold fur-

[1] https://github.com/scikit-learn/scikit-learn

ther explain how and why our method works. Our future work includes making the mixing weights learnable as well as extending DOUBLEMIX to natural language generation tasks.

Acknowledgement

This work is supported by the A*STAR under its RIE 2020 AME programmatic grant RGAST2003 and project T2MOE2008 awarded by Singapore's MoE under its Tier-2 grant scheme.

References

Jacob Andreas. 2020. Good-enough compositional data augmentation. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7556–7566, Online. Association for Computational Linguistics.

Philip Bachman, Ouais Alsharif, and Doina Precup. 2014. Learning with pseudo-ensembles. In *Advances in Neural Information Processing Systems*, volume 27, pages 3365–3373.

Samuel Bowman, Gabor Angeli, Christopher Potts, and Christopher D Manning. 2015. A large annotated corpus for learning natural language inference. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 632–642.

Olivier Chapelle, Jason Weston, Léon Bottou, and Vladimir Vapnik. 2001. Vicinal risk minimization. In *Advances in neural information processing systems*, pages 416–422.

Jiaao Chen, Zichao Yang, and Diyi Yang. 2020. MixText: Linguistically-informed interpolation of hidden space for semi-supervised text classification. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 2147–2157.

Yong Cheng, Lu Jiang, Wolfgang Macherey, and Jacob Eisenstein. 2020. AdvAug: Robust adversarial augmentation for neural machine translation. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 5961–5970, Online. Association for Computational Linguistics.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. Bert: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186.

Sergey Edunov, Myle Ott, Michael Auli, and David Grangier. 2018. Understanding back-translation at scale. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 489–500.

Demi Guo, Yoon Kim, and Alexander Rush. 2020. Sequence-level mixed sample data augmentation. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 5547–5552, Online. Association for Computational Linguistics.

Hongyu Guo, Yongyi Mao, and Richong Zhang. 2019. Mixup as locally linear out-of-manifold regularization. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, pages 3714–3722.

Dan Hendrycks, Norman Mu, Ekin Dogus Cubuk, Barret Zoph, Justin Gilmer, and Balaji Lakshminarayanan. 2020. Augmix: A simple data processing method to improve robustness and uncertainty. In *International Conference on Learning Representations*.

Ganesh Jawahar, Benoît Sagot, and Djamé Seddah. 2019. What does BERT learn about the structure of language? In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 3651–3657, Florence, Italy. Association for Computational Linguistics.

Divyansh Kaushik, Eduard Hovy, and Zachary Lipton. 2020. Learning the difference that makes a difference with counterfactually-augmented data. In *International Conference on Learning Representations*.

Jang-Hyun Kim, Wonho Choo, and Hyun Oh Song. 2020. Puzzle mix: Exploiting saliency and local statistics for optimal mixup. In *International Conference on Machine Learning*, pages 5275–5285. PMLR.

Yekyung Kim, Seohyeong Jeong, and Kyunghyun Cho. 2021. Linda: Unsupervised learning to interpolate in natural language processing. *arXiv preprint arXiv:2112.13969*.

Soonki Kwon and Younghoon Lee. 2022. Explainability-based mix-up approach for text data augmentation. *ACM Transactions on Knowledge Discovery from Data (TKDD)*.

Xin Li and Dan Roth. 2002. Learning question classifiers. In *COLING 2002: The 19th International Conference on Computational Linguistics*.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Roberta: A robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692*.

Andrew Maas, Raymond E Daly, Peter T Pham, Dan Huang, Andrew Y Ng, and Christopher Potts. 2011. Learning word vectors for sentiment analysis. In *Proceedings of the 49th annual meeting of the association for computational linguistics: Human language technologies*, pages 142–150.

Tom McCoy, Ellie Pavlick, and Tal Linzen. 2019. Right for the wrong reasons: Diagnosing syntactic heuristics in natural language inference. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 3428–3448, Florence, Italy. Association for Computational Linguistics.

Junghyun Min, R. Thomas McCoy, Dipanjan Das, Emily Pitler, and Tal Linzen. 2020. Syntactic data augmentation increases robustness to inference heuristics. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 2339–2352, Online. Association for Computational Linguistics.

Pasquale Minervini and Sebastian Riedel. 2018. Adversarially regularising neural nli models to integrate logical background knowledge. In *Proceedings of the 22nd Conference on Computational Natural Language Learning*, pages 65–74.

Seo Yeon Park and Cornelia Caragea. 2022. A data cartography based MixUp for pre-trained language models. In *Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 4244–4250.

Alexandre Ramé, Rémy Sun, and Matthieu Cord. 2021. Mixmo: Mixing multiple inputs for multiple outputs via deep subnetworks. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 823–833.

Rico Sennrich, Barry Haddow, and Alexandra Birch. 2016. Improving neural machine translation models with monolingual data. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 86–96.

Karen Simonyan, Andrea Vedaldi, and Andrew Zisserman. 2014. Deep inside convolutional networks: Visualising image classification models and saliency maps. In *In Workshop at International Conference on Learning Representations*. Citeseer.

Richard Socher, Alex Perelygin, Jean Wu, Jason Chuang, Christopher D Manning, Andrew Y Ng, and Christopher Potts. 2013. Recursive deep models for semantic compositionality over a sentiment treebank. In *Proceedings of the 2013 conference on empirical methods in natural language processing*, pages 1631–1642.

Lichao Sun, Congying Xia, Wenpeng Yin, Tingting Liang, S Yu Philip, and Lifang He. 2020. Mixup-transformer: Dynamic data augmentation for nlp tasks. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 3436–3440.

AFM Shahab Uddin, Mst Sirazam Monira, Wheemyung Shin, TaeChoong Chung, and Sung-Ho Bae. 2020. Saliencymix: A saliency guided data augmentation strategy for better regularization. In *International Conference on Learning Representations*.

Laurens Van der Maaten and Geoffrey Hinton. 2008. Visualizing data using t-sne. *Journal of machine learning research*, 9(11).

Vikas Verma, Alex Lamb, Christopher Beckham, Amir Najafi, Ioannis Mitliagkas, David Lopez-Paz, and Yoshua Bengio. 2019. Manifold mixup: Better representations by interpolating hidden states. In *International Conference on Machine Learning*, pages 6438–6447. PMLR.

Jason Wei and Kai Zou. 2019. Eda: Easy data augmentation techniques for boosting performance on text classification tasks. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 6383–6389.

Adina Williams, Nikita Nangia, and Samuel Bowman. 2018. A broad-coverage challenge corpus for sentence understanding through inference. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 1112–1122.

Qizhe Xie, Zihang Dai, Eduard Hovy, Thang Luong, and Quoc Le. 2020. Unsupervised data augmentation for consistency training. *Advances in Neural Information Processing Systems*, 33:6256–6268.

Diyi Yang, Alon Lavie, Chris Dyer, and Eduard Hovy. 2015. Humor recognition and humor anchor extraction. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 2367–2376.

Wenpeng Yin, Huan Wang, Jin Qu, and Caiming Xiong. 2021. Batchmixup: Improving training by interpolating hidden states of the entire mini-batch. In *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*, pages 4908–4912.

Soyoung Yoon, Gyuwan Kim, and Kyumin Park. 2021. Ssmix: Saliency-based span mixup for text classification. In *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*, pages 3225–3234.

Hongyi Zhang, Moustapha Cisse, Yann N Dauphin, and David Lopez-Paz. 2018. mixup: Beyond empirical risk minimization. In *International Conference on Learning Representations*.

Stephan Zheng, Yang Song, Thomas Leung, and Ian Goodfellow. 2016. Improving the robustness of deep neural networks via stability training. In *Proceedings of the ieee conference on computer vision and pattern recognition*, pages 4480–4488.

A Dataset Statistics

Table 6 describes the statistics of the datasets we used. Note that for SST-2, we did not use the one

on the GLUE benchmark, as the test labels are not publicly available. We used the original SST-2 dataset and it can be loaded from huggingface datasets[2].

Dataset	Task Type	# Label	Size
SST-2	Sentiment	2	6.9k / 872 / 1.8k
TREC	Classification	6	4.9k / 546 / 500
Puns	Humor	2	3.6k / 603 / 604
IMDB	Sentiment	2	22.5k / 2.5k / 25k
SNLI	Inference	3	550k / 9.8k / 9.8k
MNLI	Inference	3	392k / 9.8k / 9.8k

Table 6: Summary statistics of the seven natural language understanding datasets. We report the size of datasets as (train / validation / test) format.

B Baseline Details

This section introduces the details of our baselines:

- **Easy Data Augmentation** (Wei and Zou, 2019) contains several simple data augmentation techniques in text such as synonym replacement, random insertion, random swap, and random deletion. The experimental setup for these methods is the same as that of back-translation. We use the official code[3] with the default insertion/deletion/swap ratio the author provided.

- **Back-translation** (Sennrich et al., 2016) translates an input text in some source language (e.g. English) to another intermediate language (e.g. German), and then translates it back into the original one. In our experiments, our intermediate languages are German and Russian. And we create two types of augmented text for every training sample. These augmented examples are directly added to the training set. We use the code of fairseq[4] to implement this baseline.

- **Syntactic Transformation** (Min et al., 2020) applies rule-based syntactic transformations such as inversion and passivization to sentences to generate augmentations in inference tasks. We directly add the augmented data to the training set. The implementation is based on the official code[5] the author provided.

- **Manifold Mixup** (Verma et al., 2019) performs in hidden space. Similar to Mixup (Zhang et al., 2018), Manifold Mixup samples two training examples and mixes up the hidden representations using a coefficient λ_0 randomly sampled from Beta(α, β). For the training objective, Manifold Mixup first uses the Cross-Entropy loss to measure the divergence between the predicted distribution and the one-hot vector of gold label, and then mix up the Cross-Entropy losses. Our implementation is based on the official code of Mixup[6].

- **TMix** (Chen et al., 2020) is similar to Manifold Mixup which performs interpolation in hidden space. We first mix up the gold labels to a sythetic label and secondly minimize the KL divergence between the synthetic label and the predicted distribution. We use the code implemented in the MixText[7] repository.

- **SSMix** (Yoon et al., 2021) is similar to PuzzleMix (Kim et al., 2020) and SaliencyMix (Uddin et al., 2020). It applies Mixup based on the saliency (Simonyan et al., 2014) of tokens. Our implementation is based on the official code of SSMix[8].

C Implementation Details

For all the experiments, we set the learning rate of the encoder model as 1e-5, set the learning rate of the two-layer MLP classifier as 1e-3, and tried different batch sizes within 8, 16, and 32 to choose the best performance. For the hyper-parameters in Dirichlet distribution and Beta distribution, we set τ as 1.0 and set α as 0.75. The coefficient γ of JSD loss was set to be 8, and the max number of training epochs was set to be 20. All these hyper-parameters are shared among the models. All the experiments are performed multiple times across different seeds on a single NVIDIA RTX 8000 GPU.

[2] https://huggingface.co/datasets/gpt3mix/sst2
[3] https://github.com/jasonwei20/eda_nlp
[4] https://github.com/pytorch/fairseq/blob/main/examples/wmt19/README.md
[5] https://github.com/aatlantise/syntactic-augmentation-nli
[6] https://github.com/hongyi-zhang/mixup
[7] https://github.com/GT-SALT/MixText
[8] https://github.com/clovaai/ssmix

Large Sequence Representation Learning via Multi-Stage Latent Transformers

Ionut-Catalin Sandu **Daniel Voinea** **Alin-Ionut Popa**

Amazon Inc.

{sanion, dvoinea, popaaln}@amazon.com

Abstract

We present **LANTERN**, a multi-stage transformer architecture for named-entity recognition (NER) designed to operate on indefinitely large text sequences (*i.e.* $\gg 512$ elements). For a given image of a form with structured text, our method uses language and spatial features to predict the entity tags of each text element. It breaks the quadratic computational constraints of the attention mechanism by operating over a learned latent space representation which encodes the input sequence via the cross-attention mechanism while having the multi-stage encoding component as a refinement over the NER predictions. As a proxy task, we propose **RADAR**, an LSTM classifier operating at character level, which predicts the relevance of a word with respect to the entity-recognition task. Additionally, we formulate a challenging novel NER use case, nutritional information extraction from food product labels. We created a dataset with $11,926$ images depicting food product labels entitled **TREAT** dataset, with fully detailed annotations. Our method achieves superior performance against two competitive models designed for long sequences on the proposed **TREAT** dataset.

I. Introduction. Information extraction from images and unstructured text plays a key role in natural language understanding (NLU) with direct impact in domains such as news content synthesis (Ozsoy et al., 2011; Moratanch and Chitrakala, 2017; Foong et al., 2015), query search (Choi et al., 2018) or knowledge-base systems (Agirre et al., 2018). The transformer architecture (Vaswani et al., 2017) played a pivotal role in advancing the state-of-the art due to its robustness with respect to tasks related to sequential data manipulation and understanding. Its success relies mostly in the attention mechanism, considered as a key component for its versatility and its ability to self-specialize and filter the input information flow.

At the same time, the attention is a curse for transformers, as it scales quadratically with respect to the size of the input sequence. To overcome this limitation, we propose a sequence parsing framework for textual information understanding allowing for indefinitely large input sequences, with focus on the NER task. Our method is entitled **LANTERN** (*i.e.* **LA**rge seque**N**ce **T**ransform**ER** for **N**ER). This is achieved via a multi-stage transformer approach over a latent space representation of the input sequence inspired from (Jaegle et al., 2021a) and adapted to NER. The key innovation we propose is a framework which analyses the entire sequence via a latent space attention-based modelling and leverages a multi-stage prediction setup.

II. Related Work. The transformer based architecture led to significant advances across major areas of NLU such as text classification (Lin et al., 2021), question answering (Choi et al., 2018), sequence to sequence translation (Rae et al., 2019; Lewis et al., 2019) and sentiment analysis (Zhang et al., 2018).

Pretraining (Yang et al., 2019; Devlin et al., 2018; Brown et al., 2020) proved extremely useful for generic language understanding tasks. These are unsupervised learning models which produce a vocabulary representation of the unlabelled training corpus via a masked language modelling task. One such model is Bidirectional Encoder Representation from Transformers (BERT) (Devlin et al., 2018). Given the pretrained weights of the model obtained from a large unlabelled data corpus, it has proven quite impactful in terms of peak performance and versatility for different downstream linguistic tasks, such as question answering (Garg et al., 2020; Laskar et al., 2020), NER (Xu et al., 2020b,a; Li et al., 2021; Zhang et al., 2020; Hwang et al., 2020; Lee et al., 2022; Huang et al., 2022; Sandu et al., 2022) or sentence / document classification (Lin et al., 2021).

Most of these approaches are constrained by

Proceedings of the 29th International Conference on Computational Linguistics, pages 4633–4639
October 12–17, 2022.

(a) (b) (c)

Figure 1: **(a) TREAT labelling sample.** Visualizations of entity-based annotations containing nutrient information. First, the annotators look for the `[PER100]` keyword, followed by the identification of the key / value pair of the nutrients of interest. **(b) Word cloud visualization of TREAT words.** The vocabulary of interest is strongly related to ingredient and nutritional information word corpus. Some of the highest occurring words are measurements such as *g* or *kcal*. **(c) Stage-wise performance of LANTERN on TREAT dataset.** The performance improves across stage and batch dimensions, with a significant gap across the first stages.

				Carbohydrates		Energy		Proteins		Salt		Sugar		Total Fat		Saturated Fat	
Statistic	**All**	**Other**	**Per 100**	**Key**	**Value**	**Key**	**Value**	**Key**	**Value**	**Key**	**Value**	**Key**	**Value**	**Key**	**Value**	**Key**	**Value**
Min / Max	4/2007	4/1916	0/24	0/44	0/11	0/22	0/17	0/26	0/10	0/20	0/10	0/31	0/10	0/49	0/19	0/14	0/7
Mean	488.63	458.69	2.01	1.36	1.00	1.36	1.92	1.18	1.00	0.98	0.81	2.12	0.93	1.33	1.03	2.11	0.64
Std.	165.8	162.24	1.94	2.12	0.82	1.8	1.68	1.6	0.8	1.5	0.8	3.26	0.75	2.23	0.98	4.86	0.79
Median	482	462	2	1	1	1	2	1	1	1	1	1	1	1	1	0	0

Table 1: **TREAT Dataset Image-Level Statistics**. Per image statistics with the words corresponding to each unique entity from **TREAT**. The `[OTHER]` entity is dominant (*i.e.* it represents $\approx 85\%$ out of the total labelled data corpus) as the nutrient section of a food label usually occupies around 10% of the total space of the label. The mean / median statistics does not reflect directly the length of the sequence inputed to **LANTERN**. It is processed via a tokeniser (Kudo, 2018) and the resulted length is usually $2.5\times$ longer.

the quadratic computational limitation of the attention mechanism (sequences of up to 512 elements), thus bounded to subcontext. Several approaches (Dai et al., 2019; Goyal et al., 2021; Beltagy et al., 2020; Zaheer et al., 2020; Ainslie et al., 2020) considered modelling the sequence as a whole from the prism of the attention mechanism, even for sequence lengths $\gg 512$.

In (Beltagy et al., 2020), the authors propose a sparse attention mechanism to analyse larger sequences of up to 4096 elements, while keeping the computational time of the attention matrix linear with respect to the input size. A similar setup is also proposed in (Zaheer et al., 2020) using an additional random iterative attention mechanism to parse the entire graph of the sequence. Different from these approaches, our proposed pipeline can process indefinitely large sequences and has a stage-wise refinement mechanism of the NER predictions.

III. Dataset. Additionally, we introduce a new language modelling problem setup from the umbrella of NER tasks, namely nutritional information extraction from images of food products. This was achieved by collecting a dataset of food product images, entitled **TREAT** (*i.e.* nu**TRi**i**E**nt f**A**c**T**s) with $11,926$ images depicting food products with fully extracted text information, bounding box image localization and complete NER class label annotations with sequences of up to $\approx 2,000$ words (≈ 5000 tokens) to be parsed. Labelling was performed using a web interface highlighting words. Annotators had to individually assign classes to words of interest. The proposed NER use-case is formulated on a linguistically diverse dataset in terms of semantics and specialisation. The collected data covers multiple European languages (EN, FR, DE, ES, IT, etc.) containing the nutritional information expressed with different languages within the same input text sequence. We further explore the specialisation in the compliance domain, learning to reason within the vocabulary of text content from food product labels. They must contain information related, but not limited, to ingredients, storage instructions, manufacturer addresses, nutritional information, allergen statements, advertising and production process details. Every piece of information can be found in standalone paragraphs, lists or table structures. The general layout is rich in both structure and visual cues. In terms of textual information, the collected data has both semantic text with full paragraphs or text structures (*e.g.* instructions, tables, lists or advertisement sections) and independent sentences

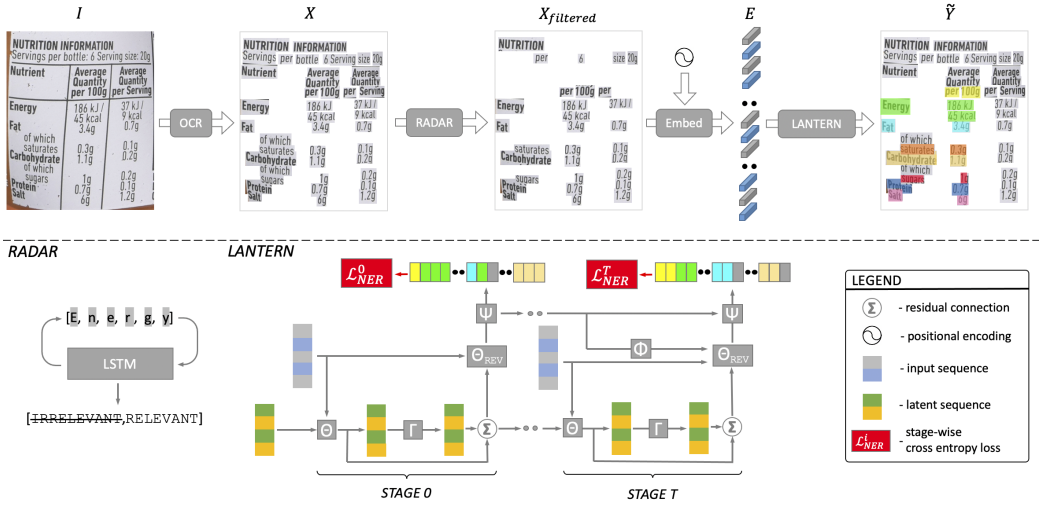

Figure 2: **Detailed overview of our proposed large-sequence parsing model.** Given an image **I**, we first extract all the words using a state-of-the-art algorithm, obtaining the complete list of words **X**. Next, **X** is filtered using **RADAR** thus obtaining the most relevant words with respect to our entity recognition task, $\mathbf{X}_{filtered}$. The filtered list is embedded into a language and positional representation, **E**, of the remaining words. Lastly, **E** goes through **LANTERN** to generate predictions $\tilde{\mathbf{Y}}$.

				Carbohydrates		Energy		Proteins		Salt		Sugar		Total Fat		Saturated Fat	
Model	**All**	**Other**	**Per 100**	**Key**	**Value**	**Key**	**Value**	**Key**	**Value**	**Key**	**Value**	**Key**	**Value**	**Key**	**Value**	**Key**	**Value**
(Beltagy et al., 2020)	0.74	0.96	0.78	0.78	0.71	0.80	0.70	0.78	0.71	0.74	0.68	0.78	0.73	0.75	0.72	0.67	0.52
(Zaheer et al., 2020)	0.76	0.96	0.79	0.77	0.74	0.83	0.69	0.78	**0.78**	0.79	0.77	0.80	0.77	0.79	0.74	0.64	0.53
(Xu et al., 2020b)	0.61	0.96	**0.86**	0.59	0.55	0.69	0.67	0.45	0.41	0.41	0.4	0.55	0.53	0.51	0.50	0.48	0.44
Ours	**0.86**	**0.97**	0.83	**0.94**	**0.82**	**0.94**	**0.87**	**0.93**	0.77	**0.91**	**0.79**	**0.94**	**0.76**	**0.89**	**0.76**	**0.87**	**0.78**

Table 2: **TREAT Test Set - Word Level Results**. The LayoutLM baseline failed to provide quality results as its context window was limited to 512 elements. Approaches of (Beltagy et al., 2020) and (Zaheer et al., 2020) obtain superior results as they are able to handle larger sequences (*i.e.* 2048). The highest F1-score is obtained with **LANTERN** as it is able to grasp the entire textual context of the product content.

such as titles, individual statements or website references. As illustrated in figure 1 (b), the downloaded images contain a high variability in terms of vocabulary corpus, mainly related to ingredient or nutritional aspects. Check figure 1 (a) for a labelling sample and table 1 for dataset statistics. Additionally, we perform an analysis from a linguistic point of view and notice the following distribution of dominant languages: 40% english, 11% spanish, 11% italian, 10% german and 8% french. The rest of 20% are data samples with mixed or other languages, unable to determine the dominant one.

We build a vocabulary with $15,000$ unique tokens using (Kudo, 2018). Up to 24% of the resulted tokens are made of 5 characters, while tokens with different sizes occupy a relatively uniform space (*e.g.* 13% tokens are 7 characters long, 10% tokens are 6 characters long and tokens with 2 or 3 characters take 8% of the vocabulary space).

IV. Methodology. Let $\mathbf{I} \in \mathbb{R}^{h \times w}$ denote an image containing a list $\mathbf{X} = (x_1 \dots x_{M_\mathbf{I}})$ of $M_\mathbf{I}$ words, with $x_i = (w_i, c_i)$ where w_i is the OCR (AWS-Textract, 2019) extracted word and $c_i \in [0,1]^4$ represents the bounding box coordinates relative to dimensions of **I**. Additionally, let $\mathbf{Y} = (y_1 \dots y_{M_I})$ denote the class labels for each word w_i in **X**.

Given the high inflation of `[OTHER]` inside **TREAT** (see table 1) we constructed **RADAR** (**R**elev**A**nt wor**D** cl**A**ssifie**R**) using a bidirectional LSTM model (Hochreiter and Schmidhuber, 1997) at character level to predict if the word is relevant or irrelevant (`[OTHER]`). Thus, we have $\mathbf{X}_{filtered} = \{x_i \mid \mathbf{RADAR}(x_i) > \alpha, i = 1 \dots M_\mathbf{I}\}$ where α is validated to obtain a recall of 1 for relevant words. To simplify notation, we identify $\mathbf{X}_{filtered}$ with **X**.

The language based embeddings for w_i are supplemented with positional encodings (Vaswani et al., 2017). Thus, we obtain the embedding representation $\mathbf{E} = (e_1 \dots e_{M_\mathbf{I}})$ of **X** where $e_i = (e^w_{i1} \dots e^w_{id_w}, e^x_{i1} \dots e^x_{id_x}, e^y_{i1} \dots e^y_{id_y})$.

This leads to an input sequence $\mathbf{E} \in \mathbb{R}^{M \times d}$, where $d = d_w + d_x + d_y$. The dimensions d_w, d_x and d_y denote language embedding sizes while x and y denote positional variables with respect to image width and height, respectively. Additionally, a latent block representation denoted with $\mathbf{L}^{\texttt{INIT}} \in \mathbb{R}^{N \times d}$ ($N \ll M$) is learned. The intuition behind the latent block is to learn a projection of the most relevant information with respect to the NER task.

The **LANTERN** module is applied in a stage-wise manner. Each stage receives as input the embedded input sequence, $\mathbf{E}$, and a latent block, $\mathbf{L}^t \in \mathbb{R}^{N \times d}$, and it predicts an array, $\tilde{\mathbf{Y}}^t$. In the following, we will describe the computational stages $t \in \{0 \dots T\}$. For $t = 0$, we have $\mathbf{L}^{\texttt{INIT}} \in \mathbb{R}^{N \times d}$.

STAGE t=0: we start by applying a cross-attention mechanism $\Theta : \mathbb{R}^{M \times d} \rightarrow \mathbb{R}^{N \times d}$, over $\mathbf{E}$ and $\mathbf{L}^{\texttt{INIT}}$. Functions $k(\cdot)$, $q(\cdot)$ and $v(\cdot)$ represent the keys, queries and values, respectively.

$$\Theta(\mathbf{E}, \mathbf{L}) = softmax(\frac{q(\mathbf{L})k(\mathbf{E})^\top}{\sqrt{d}})v(\mathbf{E}) \quad (1)$$

This will basically result in a projection of the input sequence $\mathbf{E}$ to the latent space $\mathbb{R}^{N \times d}$ through $\mathbf{L}^{\texttt{INIT}}$. We will denote the resulted projected latent block with $\mathbf{L}_\Theta^{\texttt{INIT}} = \Theta(\mathbf{E}, \mathbf{L}^{\texttt{INIT}})$.
Next, $\mathbf{L}_\Theta^{\texttt{INIT}}$ is passed through a transformer encoder, $\Gamma : \mathbb{R}^{N \times d} \rightarrow \mathbb{R}^{N \times d}$, to learn an implicit statistic between the elements of the latent block elements. The initial latent block information is added as a residual information on the resulting encoded information from Γ, thus having

$$\mathbf{L}_\Gamma^{\texttt{INIT}} = \Gamma(\mathbf{L}_\Theta^{\texttt{INIT}}) + \mathbf{L}^{\texttt{INIT}} \quad (2)$$

Lastly, we apply a reversed cross-attention mechanism, $\Theta_{\texttt{REV}} : \mathbb{R}^{N \times d} \rightarrow \mathbb{R}^{M \times d}$ over the input sequence $\mathbf{E}$ and the latent block $\mathbf{L}_\Gamma^{\texttt{INIT}}$ processed through transformer Γ.

$$\Theta_{\texttt{REV}}(\mathbf{E}, \mathbf{L}) = softmax(\frac{q(\mathbf{E})k(\mathbf{L})^\top}{\sqrt{d}})v(\mathbf{L}) \quad (3)$$

The reversed cross-attention operation provides a reprojection of the transformed latent block information $\mathbf{L}_\Gamma^{\texttt{INIT}}$ to the input space, $\mathbb{R}^{M \times d}$. Thus, the resulted information from the reversed cross-attention will be, $\mathbf{L}_{\Theta_{\texttt{REV}}}^{\texttt{INIT}} = \Theta_{\texttt{REV}}(\mathbf{E}, \mathbf{L}_\Gamma^{\texttt{INIT}})$.
$\mathbf{L}_{\Theta_{\texttt{REV}}}^{\texttt{INIT}}$ is next passed through a feed-forward network, $\Psi(\cdot)$, which provides the class predictions for the sequence's entities, $\tilde{\mathbf{Y}}^{t=0}$ (*i.e.* $t = 0$ denotes the first stage), and it is being optimized using a cross-entropy loss function.

STAGE t>0: This process is repeated for several iterations, in a multi-stage setup. For the next stages (*i.e.* stage $0 < t \leq T$), the latent sequence $\mathbf{L}^t$ is initialized with the latent block information obtained from equation 2 of stage $t-1$.

$$\mathbf{L}^t = \begin{cases} \mathbf{L}_\Gamma^{\texttt{INIT}}, & \text{if } t = 1 \\ \mathbf{L}_\Gamma^{t-1}, & \text{otherwise} \end{cases}$$

The prediction of stage $t-1$ of the NER task, $\tilde{\mathbf{Y}}^{t-1}$, is fed to the feed-forward module of the current stage, t, as a weight factor. Also, $\tilde{\mathbf{Y}}^{t-1}$, is used as a weight to the attention-matrix computed at the current stage, t, obtained via a projection function $\Phi : \mathbb{R}^{M \times d_{target}} \rightarrow \mathbb{R}^{M \times N}$ where d_{target} is the total number of entities to be predicted

$$\Theta_{\texttt{REV}}(\mathbf{E}, \mathbf{L}, \mathbf{A}) = softmax(\frac{q(\mathbf{E})k(\mathbf{L})^\top}{\sqrt{d}} \odot \mathbf{A})v(\mathbf{L})$$

where $\odot(\cdot) : \mathbb{R}^{M \times N} \rightarrow \mathbb{R}^{M \times N}$ represent the Hadamard product and $\mathbf{A} \in \mathbb{R}^{M \times N}$ is a weight matrix. Thus, the reprojected latent block information from stage t becomes, $\mathbf{L}_{\Theta_{\texttt{REV}}}^t = \Theta_{\texttt{REV}}(\mathbf{E}, \mathbf{L}_\Gamma^t, \Phi(\tilde{\mathbf{Y}}^{t-1})))$.

At each stage, a cross-entropy loss is applied over the output of $\Psi : \mathbb{R}^{M \times d} \rightarrow \mathbb{R}^{M \times d_{target}}$, which are cumulated until the final stage T, $\mathcal{L}_{\texttt{NER}} = \sum_{t=0}^{T} \mathcal{L}_{\texttt{NER}}^t(\tilde{\mathbf{Y}}^t, \mathbf{Y})$.

Given an input image $\mathbf{I}$ and a learned latent block $\mathbf{L}$, the framework outputs an array $\tilde{\mathbf{Y}}$ with class predictions for all the relevant words identified within image $\mathbf{I}$. A step-by-step breakdown of our pipeline is illustrated in figure 2 and in algorithm 1.

V. Experiments. We conduct experiments on our proposed dataset **TREAT**. Prior to applying **LANTERN**, we filtered a part of the irelevant word corpus (*i.e.* `[OTHER]`) using **RADAR**. The accuracy of **RADAR** is 85% and we filtered 48% of the irrelevant words. The reason behind filtering such a low quantity of `[OTHER]` with respect to the obtained accuracy is because the majority of the word corpus from **TREAT** represents numerical values (*e.g. 100g*, *30ml*) which we marked with `[NUM]`. These words (`[NUM]`) cannot be filtered without leveraging their positional context as they can be linked with nutrient keys. Thus, we decided to introduce them as such inside **LANTERN** without

Figure 3: **Multi-stage Qualitative Results obtained with LANTERN on TREAT Test Set.** We show the progressive improvement of the generated entities achieved via the multi-stage component of LANTERN. The refinement is clearly visible from stage to stage, as the model turns its attention towards the words describing the nutrients of interest.

RADAR filtration. In table 2 we report the word-level precision, recall and F1-score. We use a total of 6 stages with a total embedding size of $d = 64$, where d_w, d_x and d_y are set to the values of 32, 16 and 16, respectively. The dimension N of latent block $\mathbf{L}$ is 256. Our method is able to achieve superior results compated to similar NER frameworks (Xu et al., 2020b), (Beltagy et al., 2020) and (Zaheer et al., 2020), some of them being designed for long sequences. For a fair evaluation, we compare against other competitive NER baselines using text and position embeddings only.

Algorithm 1 LANTERN

Input: $\mathbf{I} \in \mathbb{R}^{h \times w}$, $\mathbf{L}^{\texttt{INIT}} \in \mathbb{R}^{N \times d}$
Output: $\tilde{Y} = (\tilde{y_1} \dots \tilde{y_M})$

$\mathbf{X} \leftarrow \text{OCR}(\mathbf{I})$
$\mathbf{X}_{filtered} \leftarrow \mathbf{RADAR}(\mathbf{X})$
$\mathbf{E} \leftarrow Embed(\mathbf{X}_{filtered})$
$\mathbf{L}_{\Theta}^{\texttt{INIT}} \leftarrow \Theta(\mathbf{E}, \mathbf{L}^{\texttt{INIT}})$
$\mathbf{L}_{\Gamma}^{\texttt{INIT}} \leftarrow \Gamma(\mathbf{L}_{\Theta}^{\texttt{INIT}}) + \mathbf{L}^{\texttt{INIT}}$
$\tilde{\mathbf{Y}}^0 \leftarrow \Psi(\Theta_{\text{REV}}(\mathbf{E}, \mathbf{L}_{\Gamma}^{\texttt{INIT}}))$
for $t = 1 \dots T$ **do**
  if $t = 1$ **then**
    $\mathbf{L}^t \leftarrow \mathbf{L}_{\Gamma}^{\texttt{INIT}}$
  else
    $\mathbf{L}^t \leftarrow \mathbf{L}_{\Gamma}^{t-1}$
  $\mathbf{L}_{\Theta}^t \leftarrow \Theta(\mathbf{E}, \mathbf{L}^t)$
  $\mathbf{L}_{\Gamma}^t \leftarrow \Gamma(\mathbf{L}_{\Theta}^t) + \mathbf{L}^t$
  $\tilde{\mathbf{Y}}^t \leftarrow \Psi(\Theta_{\text{REV}}(\mathbf{E}, \mathbf{L}_{\Gamma}^t, \Phi(\tilde{\mathbf{Y}}^{t-1})))$
$\tilde{\mathbf{Y}} \leftarrow \tilde{\mathbf{Y}}^T$

In figure 3 we showcase multi-stage predictions, thus highlighting the refinement aspect of our framework on a practical example.

Method	Precision	Recall	F1-Score
LANTERN w/o RADAR	0.83	0.80	0.81
LANTERN w RADAR	**0.84**	**0.87**	**0.86**
LANTERN w/o weight sharing	0.83	0.80	0.81
LANTERN w weight sharing	**0.82**	**0.82**	**0.82**
LANTERN w Language	0.39	0.42	0.40
LANTERN w Language + Positional	**0.83**	**0.80**	**0.81**

Table 3: **Ablation study on various components of LANTERN over TREAT dataset.** The highest impact in terms of performance is obtained with the **RADAR** based filtration of `[OTHER]`. Parameter weight sharing across all the stages also impacts positively the performance and it helps the model to convergence faster.

In table 3 we analyse the impact of the various model components and embedding types used by our model, thus validating the architectural innovative aspects of our method. In figure 1 (c) we show the impact of the multi-stage component. The refinement aspect of the method is visible between consecutive stages leading to significant improvement between the first and last stages.

VI. Conclusions. We propose a novel method for NER, specifically designed for indefinitely large sequences. It leverages a multi-stage transformer-based pipeline which breaks the quadratic computational constraints of the attention mechanism by projecting the input sequence to a latent space representation and a stage-wise setup which acts as a context refinement of the entity predictions. The methodology was evaluated on a novel and difficult NER use case, nutritional information extraction, proving superior results over other strong baselines specifically designed for long sequence parsing.

Acknowledgements. We thank George Leotescu and Adrian-Mihai Alecu for helpful insights.

References

Eneko Agirre, Oier López de Lacalle, and Aitor Soroa. 2018. The risk of sub-optimal use of open source nlp software: Ukb is inadvertently state-of-the-art in knowledge-based wsd. *arXiv preprint arXiv:1805.04277*.

Joshua Ainslie, Santiago Ontanon, Chris Alberti, Vaclav Cvicek, Zachary Fisher, Philip Pham, Anirudh Ravula, Sumit Sanghai, Qifan Wang, and Li Yang. 2020. Etc: Encoding long and structured inputs in transformers. In *EMNLP*, Online. Association for Computational Linguistics.

AWS-Textract. 2019. Textract-aws ocr engine. https://aws.amazon.com/textract.

Iz Beltagy, Matthew E Peters, and Arman Cohan. 2020. Longformer: The long-document transformer. *arXiv preprint arXiv:2004.05150*.

Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel Ziegler, Jeffrey Wu, Clemens Winter, Chris Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. 2020. Language models are few-shot learners. In *NeurIPS*, volume 33, pages 1877–1901. Curran Associates, Inc.

Eunsol Choi, He He, Mohit Iyyer, Mark Yatskar, Wen-tau Yih, Yejin Choi, Percy Liang, and Luke Zettlemoyer. 2018. Quac: Question answering in context. *arXiv preprint arXiv:1808.07036*.

Zihang Dai, Zhilin Yang, Yiming Yang, Jaime Carbonell, Quoc V Le, and Ruslan Salakhutdinov. 2019. Transformer-xl: Attentive language models beyond a fixed-length context. *arXiv preprint arXiv:1901.02860*.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*.

Oi-Mean Foong, Suet-Peng Yong, and Farha-Am Jaid. 2015. Text summarization using latent semantic analysis model in mobile android platform. In *AMS*, pages 35–39. IEEE.

Siddhant Garg, Thuy Vu, and Alessandro Moschitti. 2020. Tanda: Transfer and adapt pre-trained transformer models for answer sentence selection. In *AAAI*.

Łukasz Garncarek, Rafał Powalski, Tomasz Stanisławek, Bartosz Topolski, Piotr Halama, Michał Turski, and Filip Graliński. 2021. Lambert: layout-aware language modeling for information extraction. In *International Conference on Document Analysis and Recognition*, pages 532–547. Springer.

Anirudh Goyal, Aniket Didolkar, Alex Lamb, Kartikeya Badola, Nan Rosemary Ke, Nasim Rahaman, Jonathan Binas, Charles Blundell, Michael Mozer, and Yoshua Bengio. 2021. Coordination among neural modules through a shared global workspace. *arXiv preprint arXiv:2103.01197*.

Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long short-term memory. *Neural computation*, 9(8):1735–1780.

Yupan Huang, Tengchao Lv, Lei Cui, Yutong Lu, and Furu Wei. 2022. Layoutlmv3: Pre-training for document ai with unified text and image masking. *arXiv preprint arXiv:2204.08387*.

Wonseok Hwang, Jinyeong Yim, Seunghyun Park, Sohee Yang, and Minjoon Seo. 2020. Spatial dependency parsing for semi-structured document information extraction. *arXiv preprint arXiv:2005.00642*.

Andrew Jaegle, Sebastian Borgeaud, Jean-Baptiste Alayrac, Carl Doersch, Catalin Ionescu, David Ding, Skanda Koppula, Daniel Zoran, Andrew Brock, Evan Shelhamer, et al. 2021a. Perceiver io: A general architecture for structured inputs & outputs. *arXiv preprint arXiv:2107.14795*.

Andrew Jaegle, Felix Gimeno, Andrew Brock, Andrew Zisserman, Oriol Vinyals, and Joao Carreira. 2021b. Perceiver: General perception with iterative attention. *arXiv preprint arXiv:2103.03206*.

Taku Kudo. 2018. Subword regularization: Improving neural network translation models with multiple subword candidates. *arXiv preprint arXiv:1804.10959*.

Md Tahmid Rahman Laskar, Xiangji Huang, and Enamul Hoque. 2020. Contextualized embeddings based transformer encoder for sentence similarity modeling in answer selection task. In *Proceedings of The 12th Language Resources and Evaluation Conference*, pages 5505–5514.

Chen-Yu Lee, Chun-Liang Li, Timothy Dozat, Vincent Perot, Guolong Su, Nan Hua, Joshua Ainslie, Renshen Wang, Yasuhisa Fujii, and Tomas Pfister. 2022. Formnet: Structural encoding beyond sequential modeling in form document information extraction. *arXiv preprint arXiv:2203.08411*.

Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Ves Stoyanov, and Luke Zettlemoyer. 2019. Bart: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. *arXiv preprint arXiv:1910.13461*.

Peizhao Li, Jiuxiang Gu, Jason Kuen, Vlad I Morariu, Handong Zhao, Rajiv Jain, Varun Manjunatha, and Hongfu Liu. 2021. Selfdoc: Self-supervised document representation learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 5652–5660.

Yuxiao Lin, Yuxian Meng, Xiaofei Sun, Qinghong Han, Kun Kuang, Jiwei Li, and Fei Wu. 2021. Bertgcn: Transductive text classification by combining gcn and bert. *arXiv preprint arXiv:2105.05727*.

N Moratanch and S Chitrakala. 2017. A survey on extractive text summarization. In *ICCCSP*, pages 1–6. IEEE.

Makbule Gulcin Ozsoy, Ferda Nur Alpaslan, and Ilyas Cicekli. 2011. Text summarization using latent semantic analysis. *Journal of Information Science*, 37(4):405–417.

Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Kopf, Edward Yang, Zachary DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. 2019. Pytorch: An imperative style, high-performance deep learning library. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett, editors, *NeurIPS*.

Shuai Peng, Ke Yuan, Liangcai Gao, and Zhi Tang. 2021. Mathbert: A pre-trained model for mathematical formula understanding. *arXiv preprint arXiv:2105.00377*.

Alin-Ionut Popa and Cristian Sminchisescu. 2017. Parametric image segmentation of humans with structural shape priors. In *ACCV*.

Jack W Rae, Anna Potapenko, Siddhant M Jayakumar, and Timothy P Lillicrap. 2019. Compressive transformers for long-range sequence modelling. *arXiv preprint arXiv:1911.05507*.

Ionut-Catalin Sandu, Daniel Voinea, and Alin-Ionut Popa. 2022. Consent: Context sensitive transformer for bold words classification. *arXiv preprint arXiv:2205.07683*.

Ioana-Sabina Stoian, Ionut-Catalin Sandu, Daniel Voinea, and Alin-Ionut Popa. 2022. Unstructured object matching using co-salient region segmentation. In *CVPR IMW Workshop*, pages 5051–5060.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *NIPS*.

Yang Xu, Yiheng Xu, Tengchao Lv, Lei Cui, Furu Wei, Guoxin Wang, Yijuan Lu, Dinei Florencio, Cha Zhang, Wanxiang Che, et al. 2020a. Layoutlmv2: Multi-modal pre-training for visually-rich document understanding. *arXiv preprint arXiv:2012.14740*.

Yiheng Xu, Minghao Li, Lei Cui, Shaohan Huang, Furu Wei, and Ming Zhou. 2020b. Layoutlm: Pre-training of text and layout for document image understanding. In *Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, page 1192–1200.

Zhilin Yang, Zihang Dai, Yiming Yang, Jaime Carbonell, Russ R Salakhutdinov, and Quoc V Le. 2019. Xlnet: Generalized autoregressive pretraining for language understanding. *NeurIPS*, 32.

Wenwen Yu, Ning Lu, Xianbiao Qi, Ping Gong, and Rong Xiao. 2021. Pick: processing key information extraction from documents using improved graph learning-convolutional networks. In *2020 25th International Conference on Pattern Recognition (ICPR)*, pages 4363–4370. IEEE.

Manzil Zaheer, Guru Guruganesh, Kumar Avinava Dubey, Joshua Ainslie, Chris Alberti, Santiago Ontanon, Philip Pham, Anirudh Ravula, Qifan Wang, Li Yang, et al. 2020. Big bird: Transformers for longer sequences. In *NeurIPS*.

Lei Zhang, Shuai Wang, and Bing Liu. 2018. Deep learning for sentiment analysis: A survey. *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery*, 8(4):e1253.

Peng Zhang, Yunlu Xu, Zhanzhan Cheng, Shiliang Pu, Jing Lu, Liang Qiao, Yi Niu, and Fei Wu. 2020. Trie: End-to-end text reading and information extraction for document understanding. In *Proceedings of the 28th ACM International Conference on Multimedia*, pages 1413–1422.

MockingBERT: A Method for Retroactively Adding Resilience to NLP Models*

Jan Jezabek
Hedgefrog Software LLC
jjezabek@hedgefrogsoft.com

Akash Singh
Salesforce Inc.
singh.akash@salesforce.com

Abstract

Protecting NLP models against misspellings whether accidental or adversarial has been the object of research interest for the past few years. Existing remediations have typically either compromised accuracy or required full model re-training with each new class of attacks. We propose a novel method of retroactively adding resilience to misspellings to transformer-based NLP models. This robustness can be achieved without the need for re-training of the original NLP model and with only a minimal loss of language understanding performance on inputs without misspellings. Additionally we propose a new efficient approximate method of generating adversarial misspellings, which considerably reduces the cost needed to evaluate a model's resilience to adversarial attacks.

1 Introduction

While artificial neural networks have been able to achieve human level performance on many real-world tasks, they sometimes fail in surprising ways. (Szegedy et al., 2013) showed that state of the art computer vision models can be fooled into misclassifying objects with only limited perturbations imperceptible to human viewers. Along similar lines, it was shown in (Pruthi et al., 2019) that very constrained attacks can successfully trick classification algorithms into making incorrect predictions. In fact such attacks have been used for a long time, chiefly for evading spam classifiers while remaining legible to human readers.

Protecting against such misspellings, whether accidental or intentional, has been a focus of research in the NLP field for many years (Lee and Ng, 2005). Recently, defenses suggested by (Pruthi et al., 2019) and (Jones et al., 2020) can partially remediate adversarial attacks by adding a pre-processing step, at the cost of a drop in classification performance. (Liu et al., 2020) proposed replacing a fixed word embeddings with trained character-based ones and observed improved resilience to adversarial attacks.

In existing systems a tension exists between modularity and accuracy. (Pruthi et al., 2019) and (Jones et al., 2020) propose fully modular systems that are completely oblivious of the downstream language understanding model. This provides explainability (by providing a verbatim sequence of corrected tokens) but comes at a cost of reduced accuracy on unperturbed inputs. Additionally there is an added drawback of not being able to preserve potential ambiguity present in the input, making these systems 'destructive'. Conversely, (Liu et al., 2020) is able to represent ambiguous inputs, however at the cost of losing modularity.

Our central hypothesis is that original accuracy can be preserved while at the same time ensuring modularity. In particular we show that existing classifiers based on BERT and RoBERTa, two widely used pre-trained models, can be retroactively made resilient to perturbations even if only unperturbed data was used during the initial finetuning. This can be done by replacing the heuristic subword tokenizer and token embedding with a machine learned replacement which we call MockingBERT. The MockingBERT tokenizer and embedder learns to mimic a transformer model's tokenization and layer 0 embedding mechanism while providing resilience to input perturbations.

We evaluate the performance of such models when trained on both unperturbed and perturbed training sets to understand their suitability for data augmentation. We perform a comparative analysis with the methods proposed in (Jones et al., 2020), as well as with a regular finetuned BERT model trained with data augmentation.

*MockingBERT is a reference to mockingbirds, a group of birds known for mimicking the sounds of other animals. The code for reproducing our results as well as instructions for obtaining the trained models are available at https://github.com/akash13singh/resilient_nlp/.

Proceedings of the 29th International Conference on Computational Linguistics, pages 4640–4650
October 12–17, 2022.

We also propose and evaluate WORDSCOREATTACK, an efficient and effective method for generating adversarial samples without the need for exhaustively considering all possible perturbations. WORDSCOREATTACK works by carefully choosing information bearing words in the input text. This provides a much faster alternative to the exhaustive method proposed by (Pruthi et al., 2019) at the cost of perturbing a larger number of characters in the sentence. Crucially this method requires significantly fewer calls to the underlying NLP model, which more closely approximates real world scenarios that are likely to involve rate limiting and/or a limited query budget. We manually verify that the outputs of this perturbation can still be classified correctly by a human reader with a high probability.

Our central findings are that the MockingBERT tokenizer and embedder model paired with a finetuned BERT or RoBERTa classifier achieves a high level of resilience to character-level adversarial perturbations when pre-trained on perturbed data. This results in a higher accuracy on perturbed inputs on multiple well known datasets than state of the art methods for combating adversarial misspellings such as the one described in (Jones et al., 2020). Crucially, the impact on accuracy for unperturbed inputs, while measurable, is noticeably lower than for comparable methods for protecting against adversarial attacks.

Additionally, WORDSCOREATTACK significantly reduces the number of model queries required to find adversarial inputs. This dramatically speeds up evaluation, while also making the attack more practical in real scenarios.

2 Related Work

Previous research has explored both adversarial attacks against NLP systems as well as possible defenses. This section gives a brief overview of related work and concepts.

2.1 Adversarial Attacks

(Pruthi et al., 2019) proposes an attack that exhaustively searches for a modification of up to two characters with the intent of causing a wrong prediction. The allowed modifications are from a narrow set: Adding or deleting a single internal character, swapping two neighboring internal characters or replacing an internal character with one of its neighbors in the QWERTY keyboard layout. This choice of perturbations was based on linguistic research which suggests that modifications to internal characters in a word do not significantly hinder legibility for a human reader (Rawlinson, 1976).

Another approach to adversarial attacks works by replacing individual characters with similarly looking symbols or letters from different alphabets. In this scenario the text remains easily understandable to the reader even in the presence of a large number of misspellings (Eger et al., 2019; Sokolov et al., 2020).

2.2 Defenses Against Attacks

In addition to the attack described previously, (Pruthi et al., 2019) proposes a remediation in the form of a subcharacter recurrent neural network (ScRNN), which attempts to reverse any perturbations present in the input sentence. (Jones et al., 2020) proposes a system named RobEn that clusters misspellings of vocabulary words with a bounded edit distance and maps them to the most frequent word in the cluster. This approach works even in the case where every word in the sentence is misspelled, at the cost of reduced accuracy on unperturbed inputs.

Both of these approaches are highly modular, i.e. they can be used with any language understanding model as a preprocessing step. In contrast (Liu et al., 2020) proposes joinly training a character-based word embedder and the main NLP model. A similar approach is also considered in (El Boukkouri et al., 2020) where a character-level word embedding is passed to a transformer model with a number of parameters similar to BERT. Finally, (Provilkov et al., 2020) proposes a modification to the BPE tokenization procedure commonly used in transformer models, using a mechanism similar to dropout.

These three approaches allow the representation of ambiguous misspellings and their handling in the NLP model. Even in cases of architectures not specifically aiming for resilience to misspellings, the authors observe such resilience as a side effect of their embedding procedure.

2.3 Alternatives to Subword Tokenization

Our research specifically targets NLP models based on the transformer architecture (Vaswani et al., 2017), with a focus on models derived from BERT (Devlin et al., 2019) and RoBERTa (Liu et al., 2019). Such models are pre-trained on large corpora of unlabeled data, and can then be adapted (in a process known as finetuning) to many NLP

tasks. This substantially reduces the amount of data needed for each individual task.

An important aspect of understanding BERT is that it uses the WordPiece subword tokenization (Wu et al., 2016) process. This means that while common words are typically mapped to a single token, uncommon or invented words can still be represented by a sequence of tokens without the need to resort to a catch-all token (typically denoted as UNK). During the operation of the transformer model, tokens corresponding to subwords are first represented as context-free vectors of numbers (that had been learned during training), with subsequent layers incrementally adding contextual information from other tokens using the self-attention mechanism.

There are known alternatives to subword embeddings that are similarly able to avoid emitting out-of-vocabulary tokens. One prominent example is ELMo (Peters et al., 2018), which uses character-level embeddings in addition to a word's surrounding context to come up with an embedding for a particular word in a sentence. Similar techniques have been successfully used with transformer based models (El Boukkouri et al., 2020; Ma et al., 2020).

The tokenization schemes described so far all rely on whitespace tokenization, which means they are likely to be susceptible to adversarial attacks that insert or remove whitespace. In contrast some recent transformer based models avoid using heuristic tokenizers altogether while still using a comparable number of parameters to BERT (Clark et al., 2022), (Tay et al., 2021). This is achieved by using a convolution-like process to map the sequence of input characters to embeddings before passing them to the transformer blocks.

3 Data

For training the MockingBERT tokenizer and embedder we use the unlabeled BookCorpus dataset (Zhu et al., 2015), with minimal pre-processing to reverse the existing tokenization present in that dataset. The goal is to utilize textual data that is as close as possible to unprocessed text.

For evaluation purposes we use the Large Movie Review (IMDb) dataset (Maas et al., 2011), the Stanford Sentiment Treebank (SST) dataset (Socher et al., 2013) and the Large Yelp Review dataset (Zhang et al., 2015). For the latter two datasets we use both the 2-class (binary) and 5-class variants. We subsequently refer to these datasets as IMDb, SST-2, SST-5, Yelp-2 and Yelp-5.

For each task we evaluate each model's accuracy on a randomly chosen subset of 500 sentences from each dataset's test set. The reason for the limited size of the test set is that adversarial attacks require potentially hundreds of model inferences for each sentence to find a successful perturbation.

4 Models

We propose a model, MockingBERT, that can be used in place of a transformer model's tokenizer and word embedding. The model consists of a character embedding layer that transforms each input character into a numeric vector of size 768. Characters are converted to lower case before embedding, but no other preprocessing is done (e.g. no special handling for whitespace or punctuation). The layer is followed by three stacked bidirectional LSTM layers, with a hidden size of 768. The final LSTM layer is connected to two parallel dense layers: The subword boundary detection layer and the subword embedding layer (Figure 1). The subword boundary detection layer uses a sigmoid activation function and has an output dimension of 1. The subword embedding layer uses hyperbolic tangent as its activation function and has an output dimension of 768, matching the subword embedding size of both BERT Base and RoBERTa Base.

The model is trained with two objectives. The first objective is a character-level sequence classification task. The subword boundary detection layer is used as a classifier. For every input character it predicts whether the character is a subword boundary, i.e. the last character of a subword. The second objective is a regression task where the subword embedding layer outputs the embedding for each subword. The embeddings are only used for input characters deemed to be subword boundaries. For non-boundary characters the embeddings are discarded by using a mask.

The tokenizer and embedder model is trained on sentences from BookCorpus (Zhu et al., 2015). During training an unperturbed sentence is processed by the transformer model's regular tokenizer and its context free word embedding (trained layer 0 embeddings without the position and segmentation embeddings). This gives us the subword boundaries and embeddings which are used as labels to train the model. The sentence may subsequently be perturbed by adding or deleting a

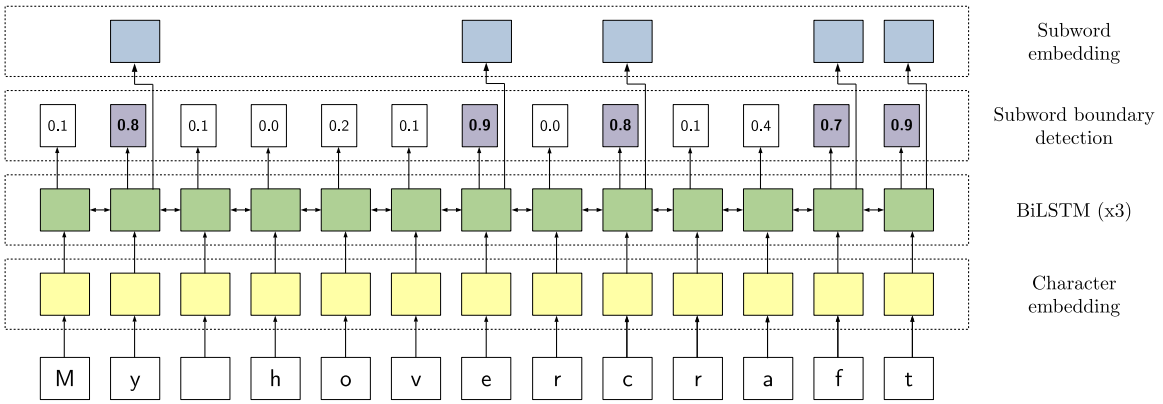

Figure 1: Structure of the MockingBERT tokenizer and embedder. In this example the phrase 'My hovercraft' is split into five tokens. An embeddings is only used if the corresponding subword boundary is detected with confidence of at least 0.5.

character, swapping two neighboring characters, inserting whitespace in the middle of long words, or removing whitespace between two words. The subword boundaries are updated accordingly. In the case of character deletion, special handling is present for characters that are subword boundaries. In this case the immediately preceding character is marked as a subword boundary. In the uncommon case when the preceding character is already a subword boundary, the embedding for the deleted character is simply discarded. In our opinion this situation is sufficiently rare that it does not warrant adding more complexity to the models.

When training, we can choose which context-free subword embedding MockingBERT will approximate:

- We can target the embedding of the generic (that is not finetuned) BERT or RoBERTa model. An advantage of this is that the trained MockingBERT embedder is independent of the finetuned task.
- Alternatively we can target the embedding of a finetuned transformer model. In this case the MockingBERT instance is no longer task agnostic, but can potentially better match the embeddings expected by the transformer model.

In our experiments we evaluate both of these approaches to understand the tradeoffs.

The models are trained on 64000 randomly selected sentences from BookCorpus for 5 epochs. The loss function is a combination of the mean squared error (MSE) losses for the subword boundary detection task and for the embedding prediction task. MSE was chosen since it can be applied both for subword boundary detection (classification) and embedding prediction (regression). The loss values for the two components are scaled to have the same magnitude. This is to prevent one of the subtasks from dominating the other one during training.

When training on perturbed data and targeting generic embeddings for BERT and RoBERTa, the models achieved accuracies for the subword boundary detection task of 99.30% and 99.25% respectively on a held-out evaluation set.

During evaluation and inference, the input text is converted to lower case and the characters are embedded as a 768-dimensional vector representation. Subsequently the MockingBERT model is executed to obtain subword embeddings for the input sequence. The embeddings are then bookended by the fixed representations of the transformer's special tokens, [CLS] and [SEP] in the case of BERT and <s> and </s> for RoBERTa. Finally, the sequences of embeddings are passed to the finetuned transformer model.

The transformer models are based on the pretrained BERT Base and RoBERTa Base models and finetuned using the HuggingFace Transformers package (Wolf et al., 2020).

For finetuning we attach a linear layer on top of the [CLS] (for BERT) or <s> (for RoBERTa) output embedding, and train the entire model using cross-entropy loss. We restrict the sequence length to 128 tokens, and the model is trained with a batch-size of 32, learning rate of 2e-05, for up to 5 epochs. We used an early stopping patience of 10, evaluated every 100 training steps.

5 Proposed Attacks

We propose WORDSCOREATTACK , a cost effective way to generate adversarial attacks which can occur in an real-world setting. Typically trained models are hosted and exposed through an API and the users including adversaries can only query the models by sending input and receiving output. Thus we only focus on black-box attacks, where one does not have access to the trained model.

WORDSCOREATTACK intelligently selects input words to perturb in order to maximize the chances of finding an adversarial example with the minimum number of perturbations. This is achieved by computing corpus specific word scores which are based on per-word conditional class probabilities for the corpus. For a binary classification task we compute the log likelihood of each word as shown in equations 1 to 3, where $freq_{pos}, freq_{neg}$ are the frequencies of the word in positive and negative classes respectively. N_{pos}, N_{neg} are the total words in the positive and negative corpus and V is the total vocabulary size. We remove stop words and low frequency words before computing word scores. In the case of multi-class classification, for each class a separate score is computed by considering that class as positive and combining all other classes into the negative class.

$$word\_score = log(\frac{P(word_{pos})}{P(word_{neg})}) \quad (1)$$

$$P(word_{pos}) = \frac{freq_{pos} + 1}{N_{pos} + V} \quad (2)$$

$$P(word_{neg}) = \frac{freq_{neg} + 1}{N_{neg} + V} \quad (3)$$

Given an input text and the original predicted class, WORDSCOREATTACK targets words in the input text which have the highest scores for the given class and perturbs them in the decreasing order of scores, until the model prediction is flipped. Our hypothesis here is that the words with the highest scores are critical to the model's predictions and perturbing them can fool the model to make the wrong classification.

We impose a query budget setting expressed by two parameters: *(max_words_to_perturb, max_tries_per_word)*. The first one denotes the maximum number of words that can be perturbed for each input text. The second parameter denotes the number of perturbation attempts allowed per word. When the *max_tries_per_word* for a word are exhausted and do not yield a successful attack, the attack greedily preserves the perturbation that decreases the model confidence the most (i.e. the one with the lowest score for the original predicted class) and moves to the next word in the order of word scores. The maximum number of model queries per input text are *max_words_to_perturb* * *max_tries_per_word*.

In order to mimic real world misspellings, we only allow one perturbation per input word. The perturbations considered are adding or deleting a character, swapping of adjacent characters, as well as splitting of a word (adding whitespace) and merging of adjacent words (deleting whitespace). Furthermore for the non-whitespace perturbations, only the internal letters of a word are perturbed, and the first and last letters remain unmodified. This ensures the perturbed text can be comprehended by humans (Rawlinson, 1976; Pruthi et al., 2019).

Though we allow one perturbation per word, multiple words per input text can be perturbed until a successful attack is found. This is in contrast to (Pruthi et al., 2019), where the authors do an exhaustive search to find a single character perturbation that flips the model prediction.

The design goal for WORDSCOREATTACK is to find an adversarial perturbation with a significantly lower query budget compared to an exhaustive search. This is done with the objective of constructing a practical framework for both simulating adversarial attacks to analyse model resilience as well as for constructing adversarial samples for data augmentation and adversarial training.

6 Experiments

We evaluated each model's performance on an unperturbed version of the test sets, as well as on WORDSCOREATTACK. We also evaluated the fine-tuned BERT model on the IMDb task using the exhaustive adversarial attack as described in (Pruthi et al., 2019). TextAttack (Morris et al., 2020), a Python framework for adversarial attacks, was used to execute the exhaustive attacks. Due to the prohibitive computational cost of this attack we were not able to evaluate it on the other models.

We evaluated variations both including and excluding whitespace modifications. This is because the remediations proposed in (Jones et al., 2020) have not been specifically designed for combating such modifications.

For WORDSCOREATTACK, we allowed the at-

Model	IMDb	SST-2	SST-5	Yelp-2	Yelp-5
BERT					
(no remediations)	**88.0**/60.6/58.6	**91.4**/42.2/38.8	**56.2**/5.8/5.0	95.2/71.2/70.4	**61.8**/27.2/25.0
with RobEn CONNCOMP	77.6/69.2/52.6	69.4/64.6/27.4	33.4/28.2/7.6	86.6/80.6/64.8	44.8/37.4/23.8
with RobEn AGGCLUST	78.6/**72.0**/53.8	75.8/**72.6**/33.8	41.0/**34.6**/6.0	90.4/86.4/71.0	52.4/**42.6**/26.0
MockingBERT with BERT					
targeting generic embedding	86.8/70.6/**69.0**	86.2/57.0/**56.8**	51.6/10.8/**13.2**	95.2/**88.6**/**89.8**	60.0/40.8/**41.0**
targeting finetuned embedding	86.4/69.8/68.6	86.8/57.4/**56.8**	49.0/10.8/9.4	**95.4**/88.4/89.0	61.2/40.2/40.2
RoBERTa					
(no remediations)	**90.8**/68.4/69.6	**93.2**/48.6/46.8	**57.2**/8.2/7.8	96.4/76.0/78.0	**63.8**/36.8/36.0
with RobEn CONNCOMP	68.4/62.6/56.0	75.0/70.0/37.6	33.6/30.4/5.8	88.0/80.6/68.6	50.0/41.6/31.4
with RobEn AGGCLUST	75.8/71.6/60.8	78.2/**74.8**/40.6	39.8/**34.8**/7.6	91.4/87.2/75.2	56.8/**48.8**/36.6
MockingBERT with RoBERTa					
targeting generic embedding	87.2/75.4/**77.0**	88.8/62.0/**62.2**	52.4/16.2/15.0	96.0/**88.0**/**88.2**	62.8/47.0/**47.4**
targeting finetuned embedding	87.8/**76.2**/76.0	90.0/60.4/**62.2**	52.8/18.6/**17.0**	**96.6**/87.4/**88.2**	62.6/46.8/45.8

Table 1: For each combination of model and task we provide an accuracy score for the following three variations of the test set: An unperturbed test set; a test set using WORDSCOREATTACK excluding whitespace modifications; and a test set using WORDSCOREATTACK including whitespace modifications. The reason for providing a score when excluding whitespace modifications is that some of the remediations have not been designed to counteract whitespace perturbations.

tack to change up to ten words, with a single modification allowed per word as described in section 5. For each word, up to four attempts were made in order to find the perturbation that decreased the model's confidence the most. As before, we evaluated both variants that allow and disallow whitespace modifications.

To establish baselines, we evaluated finetuned BERT and RoBERTa models with their default tokenizers, both with and without data augmentation. We also evaluated the CONNCOMP and AGGCLUST approaches proposed in (Jones et al., 2020). Accuracy is used as the primary evaluation metric.

A second stream of experiments is focused on evaluating the efficacy and efficiency of WORDSCOREATTACK. We attack the BERT model finetuned for the IMDb task and vary the *max_tokens_to_perturb* from 1 to 40 and *max_tries_per_token* from 1 to 4. For each setting, we calculate the model accuracy on the 500 reviews in the test set. The results are shown in Figure 2. The original accuracy on the test set is 88%. which is reduced to 26.6% in the most adversarial setting of (40, 4).

In order to compare WORDSCOREATTACK to the exhaustive adversarial attack of (Pruthi et al., 2019), we evaluate a forgetful mode for WORDSCOREATTACK, where an unsuccessful perturbation is reset when the attack moves to a new token. In this mode, the attack tries to flip the model's prediction by making only one perturbation to the input text. The results are shown in Table 2, where

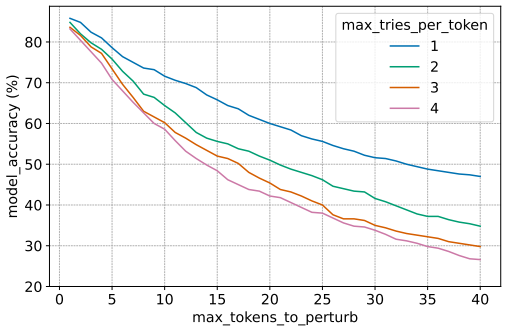

Figure 2: WORDSCOREATTACK analysis with different budget parameter settings.

the normal mode is denoted as WSA and the forgetful model is denoted as WSA-Forgetful. Both modes operate with the query budget setting of (40,4). As expected the efficacy of the attack suffers in forgetful mode, with only a 15% attack success rate and model accuracy dropping to only 72.8% from the original 88%. The exhaustive adversarial attack (Pruthi et al., 2019) on the other hand reduces the accuracy to 66.8%. However this comes at the expense of 1,574 queries on average per attack compared to only 133 queries for WSA-Forgetful. In comparison the normal mode of WSA reduces the model accuracy to 26.6%, with an attack success rate of 70% while only requiring 83 queries on average.

7 Analysis

Our experiment results (Table 1) show that MockingBERT consistently achieves the highest accu-

	Pruthi	WSA-Forgetful	WSA
Orig. Accuracy	89.6%	88%	88%
Attack Success %	25.45%	18%	70%
Final Accuracy	66.8%	72.6%	26.6%
Avg. Queries	1574	133	83

Table 2: Comparison of exhaustive attack of Pruthi, 2019 with the forgetful (WSA-Forgetful) and normal mode of WORDSCOREATTACK(WSA). Pruthi and WSA- Forgetful are constrained to perturb only one character per input text. The budget setting for both WSA attacks is (40,4)

racy scores for adversarial attacks that allow whitespace modifications by a margin of between 5.6% and 18.8%. When whitespace modifications are disallowed, MockingBERT performs similarly to RobEn, with the exception of the SST datasets where the RobEn models achieve noticeably higher accuracy. It should be noted that pretraining MockingBERT on adversarial data with whitespace modifications results in noticeably lower performance on test sets without such modifications, as noted in subsection 7.1 (Ablation Studies).

Crucially, our model's accuracy on unperturbed data is typically only slightly lower than when using the standard BERT or RoBERTa tokenization/embedding procedure, with the accuracy scores ranging from being 0.2% better to being 4.6% worse than the standard model. This is in contrast with RobEn, where the accuracy on unperturbed data is lower by between 4.8% and 17.4%.

Interestingly the version of the tokenizer that targets default pre-trained transformer embeddings performed slighly better overall than the version targeting embeddings finetuned specifically for individual tasks. This suggests that a universal tokenizer and embedder model can be used for a variety of tasks, with no need to adapt it specifically for each task. This hypothesis requires more research on a wider variety of tasks.

Through our analysis of the WORDSCOREATTACK, we demonstrated that it is a cost effective way of constructing adversarial examples. It provides a flexible framework for evaluating models under attack. As shown in Figure 2 model accuracy drops steadily as the *max_tokens_to_perturb* is increased. Accuracy declines faster with higher values of *max_tries_per_token* as more random perturbations can be tried per token.

The cost effectiveness of the attack is borne by the fact that the most adversarial budget setting of (40,4) requires an average of 83 model queries and reduces the accuracy to 26.6%. On the other the setting of (20,1) will require at most 20 queries on average but still reduces the model accuracy to 60%.

We wanted to construct adversarial samples such that humans can infer the original intent without much difficulty. Thus we considered a limited set of perturbations with constraints such as perturbing only one internal character per token, in addition to adding/deleting whitespace. Two examples of original inputs and their adversarial counterparts are given in Table 4 in the appendix.

7.1 Ablation Studies

To understand how data augmentation impacts the performance of MockingBERT, we have trained and evaluated variants of our approach trained on unperturbed training sets and on training sets where no whitespace perturbations were allowed. As with the main experiment we considered MockingBERT variants mimicking both the subword embeddings of generic (that is not finetuned) transformer models, as well as those mimicking finetuned transformer models. For comparison we have also finetuned a standard BERT model using data augmentation using the same perturbations as when training MockingBERT.

For experiments in this section we have used BERT as the underlying transformer architecture and we have evaluated it on the IMDb test set. The results (Table 3) suggest that MockingBERT models require data augmentation to develop robustness to attacks. Indeed without data augmentation the accuracy of MockingBERT based models is strictly lower than that of pure BERT models. However MockingBERT based models trained with data augmentation consistently outperform similarly trained pure BERT models in the presence of test set perturbations.

Interestingly it appears that the presence of whitespace perturbations in the training set negatively affects the MockingBERT models' accuracy on test sets without such perturbations (74.4% vs. 70.6% for the generic embedding variant and 72.8% vs. 69.8% for the finetuned embedding variant). Conversely it appears that including perturbed data when finetuning the pure BERT model improves the model's accuracy on unperturbed test data. A possible explanation is that in the presence of misspellings the transformer model learns to look for multiple redundant signals for its classification, in

Model	IMDb
BERT	
no augmentation	88.0/60.6/58.6
augmentation (incl. w/s)	**88.4**/53.6/55.4
MockingBERT targeting generic embedding	
no augmentation	86.4/56.4/53.6
augmentation (no w/s)	87.2/**74.4**/60.2
augmentation (incl. w/s)	86.8/70.6/**69.0**
MockingBERT targeting finetuned embedding	
no augmentation	86.4/57.6/55.6
augmentation (no w/s)	86.4/72.8/58.8
augmentation (incl. w/s)	86.4/69.8/68.6

Table 3: Accuracy scores for models with various data augmentation strategies. The format is the same as in Table 1.

a way similar to the effects of dropout.

8 Future Work

In our opinion it would be interesting to see if an approach similar to MockingBERT would work with other practical transformer-based models such as T5, XLNet or ELECTRA. Due to differences in how their tokenizers work some adaptations might be necessary. Nevertheless we see no fundamental issue that would prevent this approach from being applicable for these other model architectures.

Another potentially interesting direction would be to evaluate whether character-based transformer models such as Canine (Clark et al., 2022), CharFormer (Tay et al., 2021) or ByT5 (Xue et al., 2022) are better suited to data augmentation with character-level perturbations than subword based transformer models. Intuitively subword based models are not well equipped for handling misspellings due to the fact that misspelled words might end up being mapped to unrelated tokens and that the number of possible misspellings for each token is very large. It should be noted that MockingBERT has a very clean separation between the tokenization/embedding procedure (which provides resilience to misspellings) and the main transformer-based language understanding layers. This means that it is easy to swap out the tokenizer and embedder if a new attack is devised, which may not be trivial for purely character-based transformer models.

9 Conclusion

We have demonstrated that our proposed MockingBERT embedder is able to successfully mimic the operation of a traditional tokenizer and embedder as used in the BERT and RoBERTa models, with only a modest decrease in performance on classification tasks. In the presence of input perturbations, MockingBERT outperforms both a data augmented BERT model and the state of the art RobEn procedure. Furthermore we have provided evidence that a universal embedder can achieve similar results to one that is specifically trained for a particular finetuned embedding, suggesting that embeddings might not need to be trained with specific tasks in mind. We have also proposed an efficient and effective method for constructing adversarial attacks, WORDSCOREATTACK, which allows constructing such attacks at a fraction of the cost of an exhaustive search, at the expense of possibly perturbing more words within a sentence in order to achieve a similar attack success rate.

Acknowledgements

The authors would like to thank prof. Christopher Potts and the course facilitators and staff of XCS224U for their support and feedback.

Akash Singh would like to thank Salesforce Inc. for funding his participation in XCS224U.

We would like to thank the authors of the HuggingFace and TextAttack packages for making these highly useful tools freely available.

Authorship Statement

The ideas presented in this article have been jointly developed and refined by all the authors. Akash Singh implemented WORDSCOREATTACK, and performed the finetuning of the BERT model as well as the exhaustive adversarial attacks. Jan Jezabek implemented and trained the joint tokenizer and embedder models, and implemented the perturbers and the evaluation notebook.

Impact Statement

Our main motivation for this research is to counteract intentional adversarial techniques designed to evade spam or toxic/obnoxious speech detection systems. We think that making such systems more robust results in more efficient moderation systems and more civil online discourse.

That said techniques such as MockingBERT can potentially be abused for surveillance and censorship purposes, by making it harder to fool and evade systems used for monitoring.

Similarly, WORDSCOREATTACK can make it practical to evade text classification systems, re-

gardless of whether such systems' goals can be considered noble or nefarious.

Training of the MockingBERT models described in this article took slightly over 20 hours of compute time on Nvidia Tesla K80 and Nvidia GeForce GTX 1070Ti GPUs. Evaluation of the models on WORDSCOREATTACK took 25 hours, with an additional 50 hours of GPU time spent during development.

References

Jonathan H Clark, Dan Garrette, Iulia Turc, and John Wieting. 2022. Canine: Pre-training an efficient tokenization-free encoder for language representation. *Transactions of the Association for Computational Linguistics*, 10:73–91.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, Minneapolis, Minnesota. Association for Computational Linguistics.

Steffen Eger, Gözde Gül Şahin, Andreas Rücklé, Ji-Ung Lee, Claudia Schulz, Mohsen Mesgar, Krishnkant Swarnkar, Edwin Simpson, and Iryna Gurevych. 2019. Text processing like humans do: Visually attacking and shielding NLP systems. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 1634–1647, Minneapolis, Minnesota. Association for Computational Linguistics.

Hicham El Boukkouri, Olivier Ferret, Thomas Lavergne, Hiroshi Noji, Pierre Zweigenbaum, and Jun'ichi Tsujii. 2020. CharacterBERT: Reconciling ELMo and BERT for word-level open-vocabulary representations from characters. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 6903–6915, Barcelona, Spain (Online). International Committee on Computational Linguistics.

Erik Jones, Robin Jia, Aditi Raghunathan, and Percy Liang. 2020. Robust encodings: A framework for combating adversarial typos. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 2752–2765, Online. Association for Computational Linguistics.

Honglak Lee and A. Ng. 2005. Spam deobfuscation using a hidden markov model. In *CEAS*.

Hui Liu, Yongzheng Zhang, Yipeng Wang, Zheng Lin, and Yige Chen. 2020. Joint character-level word embedding and adversarial stability training to defend adversarial text. *Proceedings of the AAAI Conference on Artificial Intelligence*, 34(05):8384–8391.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Roberta: A robustly optimized BERT pretraining approach. *CoRR*, abs/1907.11692.

Wentao Ma, Yiming Cui, Chenglei Si, Ting Liu, Shijin Wang, and Guoping Hu. 2020. CharBERT: Character-aware pre-trained language model. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 39–50, Barcelona, Spain (Online). International Committee on Computational Linguistics.

Andrew L. Maas, Raymond E. Daly, Peter T. Pham, Dan Huang, Andrew Y. Ng, and Christopher Potts. 2011. Learning word vectors for sentiment analysis. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, pages 142–150, Portland, Oregon, USA. Association for Computational Linguistics.

John Morris, Eli Lifland, Jin Yong Yoo, Jake Grigsby, Di Jin, and Yanjun Qi. 2020. Textattack: A framework for adversarial attacks, data augmentation, and adversarial training in nlp. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 119–126.

Matthew E. Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer. 2018. Deep contextualized word representations. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 2227–2237, New Orleans, Louisiana. Association for Computational Linguistics.

Ivan Provilkov, Dmitrii Emelianenko, and Elena Voita. 2020. BPE-dropout: Simple and effective subword regularization. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 1882–1892, Online. Association for Computational Linguistics.

Danish Pruthi, Bhuwan Dhingra, and Zachary C. Lipton. 2019. Combating adversarial misspellings with robust word recognition. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 5582–5591, Florence, Italy. Association for Computational Linguistics.

Graham Ernest Rawlinson. 1976. *The significance of letter position in word recognition*. Ph.D. thesis, University of Nottingham.

Richard Socher, Alex Perelygin, Jean Wu, Jason Chuang, Christopher D. Manning, Andrew Ng, and Christopher Potts. 2013. Recursive deep models for

semantic compositionality over a sentiment treebank. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*, pages 1631–1642, Seattle, Washington, USA. Association for Computational Linguistics.

Mark Sokolov, Kehinde Olufowobi, and Nic Herndon. 2020. Visual spoofing in content-based spam detection. In *13th International Conference on Security of Information and Networks*, pages 1–5.

Christian Szegedy, Wojciech Zaremba, Ilya Sutskever, Joan Bruna, Dumitru Erhan, Ian Goodfellow, and Rob Fergus. 2013. Intriguing properties of neural networks. *arXiv preprint arXiv:1312.6199*.

Yi Tay, Vinh Q Tran, Sebastian Ruder, Jai Gupta, Hyung Won Chung, Dara Bahri, Zhen Qin, Simon Baumgartner, Cong Yu, and Donald Metzler. 2021. Charformer: Fast character transformers via gradient-based subword tokenization. *arXiv preprint arXiv:2106.12672*.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. *Advances in neural information processing systems*, 30.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Remi Louf, Morgan Funtowicz, Joe Davison, Sam Shleifer, Patrick von Platen, Clara Ma, Yacine Jernite, Julien Plu, Canwen Xu, Teven Le Scao, Sylvain Gugger, Mariama Drame, Quentin Lhoest, and Alexander Rush. 2020. Transformers: State-of-the-art natural language processing. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 38–45, Online. Association for Computational Linguistics.

Yonghui Wu, Mike Schuster, Zhifeng Chen, Quoc V Le, Mohammad Norouzi, Wolfgang Macherey, Maxim Krikun, Yuan Cao, Qin Gao, Klaus Macherey, et al. 2016. Google's neural machine translation system: Bridging the gap between human and machine translation. *arXiv preprint arXiv:1609.08144*.

Linting Xue, Aditya Barua, Noah Constant, Rami Al-Rfou, Sharan Narang, Mihir Kale, Adam Roberts, and Colin Raffel. 2022. Byt5: Towards a token-free future with pre-trained byte-to-byte models. *Transactions of the Association for Computational Linguistics*, 10:291–306.

Xiang Zhang, Junbo Zhao, and Yann LeCun. 2015. Character-level convolutional networks for text classification. In *Advances in Neural Information Processing Systems*, volume 28. Curran Associates, Inc.

Yukun Zhu, Ryan Kiros, Rich Zemel, Ruslan Salakhutdinov, Raquel Urtasun, Antonio Torralba, and Sanja Fidler. 2015. Aligning books and movies: Towards story-like visual explanations by watching movies and reading books. In *Proceedings of the IEEE international conference on computer vision*, pages 19–27.

A Supplemental Data

Original Input	Adversarial Input
the hand of death most definitely rates a ten on a scale of one to- due, in no small part, to john woo's masterful direction, coupled with kat's superb cinematography: some of the leisurely tracking shots alone are worth the price of a rental; there are moments when this one borders on becoming an art-house film. both james tien and sammo hung make for the kind of villains you can't help but love to hate. tien is particularly good as the baddest of the bad. it's a role reversal the likes of which i don't think i've ever seen before (tien normally played a hero and, in fact, with his,	the hand of death most defini tely rates a ten on a scale of one to- due, in no small part, to john woo's mastreful direction, coupled with kat's supreb cinematography: some of the leisurely tracking shots alone are worth the price of a rental; there are moments when this one borders on becoming an art-house film. both jtames tien and sammo hung make for the kind of villains you can't help but loe to hate. tien is particularly good as the baddest of the bad. it's a rloe reversal the likes of which i don't think i've ever seen before (tien normally played a hero and, in fact, with his
i caught this movie right in my eye when i was passing by a hall of posters in the nearby cinema. the tag line was sort of confusing and immediately after reading it, i thought of the possibility of it being similar to national lampoon's dorm daze. i liked that movie, aside from having a huge collection of such genres, i decided to hit it to the cinemas right after my exams for a tension releaser. delightfully, i came out smiling from cheek to cheek and had an equally great amount of laughter at bits and points of the movie. amanda aynes definitely kicked it off better than keira	i caughtthis movie right in my eye when i was passing by a hlal of posters in the nearby cin ema. the tag line was sort of confusing and immediately after reading it, i thought of the possibility of it being similar to national lampoon's dorm daze. i lciked that movie, aside from having a huge collection of such genres, i decided to hit it to the cinemas right after my exams for a tension releaser. delig htfully, i came out smliing from cheek to cheek and had an equallygreat amount of laughter at bits and points of the movie. amanda bynes definitely kicked it off better than keira

Table 4: Examples of original inputs with adversarial counterparts.

Equivariant Transduction through Invariant Alignment

Jennifer White[ð] **Ryan Cotterell**[ʕ]
[ð]University of Cambridge [ʕ]ETH Zürich
jw2088@cam.ac.uk ryan.cotterell@inf.ethz.ch

Abstract

The ability to generalize compositionally is key to understanding the potentially infinite number of sentences that can be constructed in a human language from only a finite number of words. Investigating whether NLP models possess this ability has been a topic of interest: SCAN (Lake and Baroni, 2018) is one task specifically proposed to test for this property. Previous work has achieved impressive empirical results using a group-equivariant neural network that naturally encodes a useful inductive bias for SCAN (Gordon et al., 2020). Inspired by this, we introduce a novel group-equivariant architecture that incorporates a group-invariant hard alignment mechanism. We find that our network's structure allows it to develop stronger equivariance properties than existing group-equivariant approaches. We additionally find that it outperforms previous group-equivariant networks empirically on the SCAN task. Our results suggest that integrating group-equivariance into a variety of neural architectures is a potentially fruitful avenue of research, and demonstrate the value of careful analysis of the theoretical properties of such architectures.

https://github.com/rycolab/equivariant-transduction

1 Introduction

Humans painlessly process sentences they have never heard before. This feat is possible because they can construct the meaning of a sentence by composing the meaning of its parts. This phenomenon is known as **compositional generalization** in the computational linguistics literature (Lake and Baroni, 2018; Hupkes et al., 2020) and it is what enables the understanding of an infinite number of novel sentences from only a finite set of words (Chomsky, 1957; Montague, 1970). For example, without knowing what action the verb *to blick* describes, we know that *blick twice* indicates that this action should be performed two times. It is natural that we would like for neural network models of language to be endowed with this ability as well—and, indeed, if they are to achieve human-like performance, it is likely to be necessary.

`walk right`	→	RTURN WALK
`walk and jump twice`	→	WALK JUMP JUMP
`jump left after walk`	→	WALK LTURN JUMP

Figure 1: Examples of SCAN inputs and outputs

There have been multiple proposals for methods of assessing these abilities in neural models (Bahdanau et al., 2019; Hupkes et al., 2020). One popular benchmark is the SCAN task (Lake and Baroni, 2018) and its derivatives (Ruis et al., 2020). SCAN involves translating natural language instructions into a sequence of actions executable by an agent navigating in an environment. Examples of these commands are shown in Figure 1. In order to test generalization capabilities, there are various test–train splits which deliberately hold out specific words. For example, the training set for the Add Jump split only contains `jump` in the simple input–output pair (`jump`, JUMP), while the test set includes commands where it is combined with modifiers, such as `jump twice` and `jump left`. Since the model has seen the effect of these modifiers on other verbs in training, a model with the ability to perform compositional generalization should be able to apply them to `jump`.

SCAN is an example of a task that could benefit from a group-equivariant network. When a network possesses the property of equivariance to a group G, acting on the input with an element of G results in the output differing by the action of the same group element. With an appropriate choice of group, this property can be used to imbue a network with the ability to generalize compositionally. If we have a set of input words, and a corresponding set of their output words, and we act on these with a permutation group which swaps words within these sets, then swapping one word for another in the input will result in the corresponding output words being swapped in the output. This effectively means that when the network has learned an example in

Proceedings of the 29th International Conference on Computational Linguistics, pages 4651–4663
October 12–17, 2022.

training it can generalize to any input which can be reached from the training example by acting on it with a group element. In the case of the SCAN task, the action on a sentence amounts to replacing a word with another vocabulary word from the same lexical class (e.g., replacing a verb with another verb). For example, if a G-equivariant network is trained on the SCAN task using a group G whose action amounts to swapping SCAN verbs, then the probability of `run` being mapped to RUN will be identical to the probability of `jump` being mapped to JUMP – thus even if only one verb is seen in training, it learns to apply observed patterns to the unseen verbs. This approach was first applied to SCAN by Gordon et al. (2020).

We present a novel group-equivariant architecture for the SCAN task which has a notable theoretical advantage over similar existing approaches: the effective orbit of the model is larger, which results in more robust generalization to novel examples. From one input command, our model can potentially generalize to an exponential number of unseen examples, where previous group-equivariant models could generalize only to a constant number. We also demonstrate the empirical effect of this advantage, showing that our model outperforms that of Gordon et al. across all splits of the SCAN task.

Concretely, we incorporate group equivariance into the hard alignment string-to-string transduction model described by Wu et al. (2018). Hard alignment differs from the more common soft attention (Bahdanau et al., 2015) in that each output symbol is aligned with precisely one input symbol, rather than calculating a weighting over all input symbols. Our model combines a group-invariant hard alignment mechanism with a group-equivariant transduction mechanism, which enables its improved generalization capabilities. Our findings motivate further exploration of group-equivariant architectures, and suggest that careful consideration of their provable equivariance properties is worthwhile.

2 Related Work

Many attempts have been made to quantify what it means for a model to exhibit compositional behaviour and to create models that can successfully generalize compositionally. Hupkes et al. (2020) provide a summary article on this topic, in which they describe different aspects of compositional generalization and formulate methods for assessing a model on each aspect. In this work, we concern ourselves primarily with what Hupkes et al. term **systematicity**—the ability to understand an unseen combination of previously seen parts.

The SCAN task which we focus on has been approached in many ways. Liu et al. (2020) achieve state-of-the-art results on the task, achieving 100.0 across all splits, using a memory-augmented model, trained using reinforcement learning, which learns how to identify the symbolic functions described by specific phrases within inputs. Lake (2019) approach the task using meta sequence-to-sequence learning. Although this approach produces excellent results, it requires a bespoke meta-training approach for each generalization task. Russin et al. (2019) achieve good results by treating syntax and semantics separately. Their model separately calculates likely alignments between input and output words, and calculates how likely a word is to be produced conditioned on the input word with which it is aligned. This approach is similar in concept to the hard alignment model that we use as the base for our G-equivariant architecture.

Equivariant networks built using group convolutions (Kondor and Trivedi, 2018) have been used most in the field of computer vision, where they have been used to create Group Convolutional Neural Networks (G-CNNs) imbued with the property of invariance to transformations of an image such as rotation and reflection (Cohen and Welling, 2016). Recently these techniques have been adopted for NLP tasks: Gordon et al. (2020) applied group-equivariant networks to the task of compositional generalization. They used the building block of the group convolution to construct group-equivariant network components, including an LSTM, an attention mechanism and an embedding layer. Their model is a group-equivariant analogue of a standard LSTM-based sequence-to-sequence model. They evaluate it on the SCAN task and achieve high accuracies on the splits assessing systematicity. We build on their work by presenting an alternative group-equivariant architecture for this task, resulting in more robust generalization abilities and improved empirical performance.

3 Group-Equivariant Networks

3.1 Group Theory

Some understanding of the basics of Group Theory are helpful to understand group-equivariant networks. This section will briefly outline the key

concepts necessary for understanding our work.

Definition 1. *A **group** $(G, \circ)$ is a set G with a binary operation $\circ$ with the following properties:*

(i) ***Closure:*** *For any $g, h \in G$, $g \circ h \in G$.*

(ii) ***Identity:*** *There exists an element e, which we call the identity, such that for all $g \in G$, $e \circ g = g \circ e = g$.*

(iii) ***Inverse:*** *For any $g \in G$ there is an element $h \in G$ such that $g \circ h = h \circ g = e$. We call this element the inverse of g and we may write $h = g^{-1}$.*

(iv) ***Associativity:*** *for any $g_1, g_2, g_3 \in G$ we have $(g_1 \circ g_2) \circ g_3 = g_1 \circ (g_2 \circ g_3)$.*

Definition 2. *Let G be a group and X a set. A **left group action** of G on X is a function $T : G \times X \to X$. As shorthand, we write $T(g, x) = g \circ x$.*

Definition 3. *If a group G acts on a set X through a defined action, then the **orbit** of an element $x \in X$ under G (which we write $G \circ x$) is $G \circ x = \{g \circ x \mid g \in G\}$.*

Definition 4. *The **symmetric group** S_n is the group of all permutations of a set of n elements. The operation for this group is composition.*

If $G = S_n$ acts on a set $X = \{x_1, \ldots, x_n\}$, then the canonical action of a group element permutes the items in the set. We will write elements of S_n in cycle notation, where $g = (a_1 a_2 \ldots a_m)$ indicates that $g \circ a_i = a_{i+1}$ for $i \in \{1, \ldots, m-1\}$ with $g \circ a_m = a_1$ (and by extension $g \circ x_{a_i} = x_{g \circ a_i}$). Indices that do not appear in the cycle are left unchanged. $()$ indicates the identity permutation, which maps each item to itself. An understanding of permutation groups will be key to our exposition, so we give an example of S_3.

Example 1. *S_3 is the symmetric group of permutations on 3 elements. In full, $S_3 = \{(), (12), (13), (23), (123), (132)\}$. It can act on any set with 3 elements. For example, if $X = \{a, b, c\}$ then $(12) \circ a = b$, $(12) \circ b = a$ and $(12) \circ c = c$, since (12) maps the first element of the set to the second element and vice versa, while leaving the third element unchanged.*

Definition 5. *A group G is **cyclic** if there is an element $g \in G$ such that all elements of G take the form g^i, where $g^i = \underbrace{g \circ g \circ \cdots \circ g}_{i}$ for some integer i. We say that G is generated by g, and write $G = \langle g \rangle$.*

Example 2. *One example of a cyclic group is a subgroup of S_n generated by the permutation $g = (123 \ldots p)$ for $p \leq n$. This permutation g has the effect of shifting each of these p elements by 1, so g^i has the effect of shifting each element by i, and g^p is equal to the identity permutation. We call this group a **cyclic shift group**.*

To revisit the example of S_3, $G = \langle (123) \rangle = \{(), (123), (132)\}$ is an example of a cyclic shift group. If this group acts on a set of 3 elements, then the identity element $()$, leaves the items in the set unchanged, while (123) "shifts" each item along by one and (132) by two.

Cyclic shift groups will be useful in our work. If we have a set X of p objects that we wish to permute, the symmetric group of all possible permutations S_p contains $p!$ elements. However, in our work we will consider each object in isolation, so we do not to consider every permutation: we only need a group G such that for a given $x_1 \in X$, for any element $x_2 \in X$ there is a permutation $g \in G$ such $g \circ x_1 = x_2$. This is true of the cyclic shift group $G = \langle (123 \ldots p) \rangle$, which contains only p elements. By using cyclic shift groups in place of the symmetric groups, we are able to avoid unnecessarily slow calculations.

Definition 6. *Let X and Y be sets and G be a group. Suppose the elements of G act on X and Y with actions denoted $g \circ x$ and $g \circ y$, $x \in X$ and $y \in Y$. A function $\alpha : X \to Y$ is **equivariant** with respect to G (G-equivariant) if and only if $\alpha(g \circ x) = g \circ \alpha(x)$ for all $x \in X$ and $g \in G$. We say that α is **invariant** to G if $\alpha(g \circ x) = \alpha(x)$ for any $x \in X$ and $g \in G$.*

3.2 SCAN: A Test of Compositionality

The SCAN task was proposed by Lake and Baroni (2018) to test a model's ability to generalize to unseen examples by composing known elements. The task involves translating between an instruction in a limited form of English with input alphabet $\Sigma = \{\texttt{walk}, \texttt{jump}, \ldots\}$ and an executable command with words taken from output alphabet $\Delta = \{\text{WALK}, \text{JUMP}, \ldots\}$. As input the model receives a string $\mathbf{x} \in \Sigma^*$, such as `jump twice`, or `walk after run`, and as output it should produce a string $\mathbf{y} \in \Delta^*$, like JUMP JUMP, or RUN WALK.

SCAN contains several splits, each designed to test a different kind of generalization. We focus on the splits that aim to evaluate the systematicity of the model – that is to say how well it can under-

stand the unseen combination of previously seen parts. This is done by withholding certain combinations from the training set and then evaluating the model's performance on these unseen combinations in the test set. For example, the training set of the Add Jump split only contains `jump` in the basic input–output pair (`jump`, JUMP), while the test set contains inputs where it is used in combination with other modifiers. Since these modifiers have been seen before with other verbs, this tests how well the model generalizes from what it has been exposed to in training.

3.3 Group-Equivariance in Transduction

We term a set of words whose function in SCAN is the same and that are used in identical contexts as a **lexical class**. We can see that, for example, any SCAN command containing `walk` would equally be a valid instruction if `walk` were replaced with `jump`, `look`, or `run`. We call this class the Verb lexical class. We divide the input vocabulary into lexical classes $L^{\text{in}}_1, \ldots, L^{\text{in}}_I \subseteq \Sigma$. If we designate a lexical class to be our equivariant lexical class $L^{\text{in}}_{\text{equi}}$, we take the group $G = \langle (12\ldots|L^{\text{in}}_{\text{equi}}|) \rangle$ to be the cyclic shift group of size $|L^{\text{in}}_{\text{equi}}|$ – i.e., the group generated by a permutation that shifts each element along by one. For example, if $L^{\text{in}}_{\text{equi}} = \{\texttt{walk}, \texttt{look}, \texttt{run}, \texttt{jump}\}$, then $G = \langle (1234) \rangle$, whose elements are the permutations $\{(), (1234), (13)(24), (1432)\}$. The group acts on $L^{\text{in}}_{\text{equi}}$ by permuting its elements, so $(1234) \circ \texttt{walk} = \texttt{look}$, for example. The group also acts in the same way on the output lexical class $L^{\text{out}}_{\text{equi}} = \{\text{WALK}, \text{LOOK}, \text{RUN}, \text{JUMP}\}$, with $(1234) \circ \text{WALK} = \text{LOOK}$. Previous work (Gordon et al., 2020) has defined the action of $g \in G$ on a sentence $\mathbf{x} = (x_1, \ldots, x_N) \in \Sigma^*$ as $g \circ \mathbf{x} = (g \circ x_1, \ldots, g \circ x_N)$.[1]

We now explain why group equivariance is a useful property for a task such as SCAN, and how it can lead to models that can generalize compositionally. Consider the SCAN task with input vocabulary Σ, output vocabulary Δ and lexical classes $L^{\text{in}}_{\text{equi}} \subseteq \Sigma$, $L^{\text{out}}_{\text{equi}} \subseteq \Delta$. Let $\xi : \Sigma^* \to \Delta^*$ be a transducer that constitutes a G-equivariant function for $G = \langle (12\ldots|L^{\text{in}}_{\text{equi}}|) \rangle$, with its action defined on Σ^* and Δ^* as above. If we have an input–output pair $(\mathbf{x}_0, \mathbf{y}_0) \in \Sigma^* \times \Delta^*$ such that $\xi(\mathbf{x}_0) = \mathbf{y}_0$, then for all input–output pairs $(\mathbf{x}, \mathbf{y}) \in \Sigma^* \times \Delta^*$ such that $(\mathbf{x}, \mathbf{y}) = (g \circ \mathbf{x}_0, g \circ \mathbf{y}_0)$ for some $g \in G$, we have $\xi(\mathbf{x}) = \xi(g \circ \mathbf{x}_0) = g \circ \xi(\mathbf{x}_0) = g \circ \mathbf{y}_0 = \mathbf{y}$. Plainly, this means that if our function successfully transduces a pair $(\mathbf{x}_0, \mathbf{y}_0)$, it will also transduce all pairs its orbit $G \circ (\mathbf{x}_0, \mathbf{y}_0) = \{(g \circ \mathbf{x}_0, g \circ \mathbf{y}_0) \mid \forall g \in G\}$. If we consider the case where ξ is obtained from the output of a neural network, this would mean that it could generalize compositionally to these examples even if the network had not been trained on them.

3.4 Constructing G-Equivariant Networks

We now describe how to build a neural network that is equivariant to a group G using G-equivariant building blocks. We describe three such building blocks that we will use in the construction of our equivariant hard-alignment model.

(1) G-Convolution. The G-convolution is the extension of the standard convolution to an arbitrary finite group G (Kondor and Trivedi, 2018). Let G be a finite group and $\boldsymbol{f} : \text{dom}(\boldsymbol{f}) \to \mathbb{R}^K$ an input function. Suppose we have D learnable filter functions where we denote the dth filter function as $\boldsymbol{\psi}^{(d)} : \text{dom}(\boldsymbol{f}) \to \mathbb{R}^K$. Then the G-convolution of $\boldsymbol{f}$ with $\{\boldsymbol{\psi}^{(d)}\}_{d=1}^D$ is a $|G| \times D$ matrix where each entry is given by the following:

$$G\text{-Conv}(\boldsymbol{f}; \boldsymbol{\psi})_{g,d} = \sum_{h \in \text{dom}(\boldsymbol{f})} \boldsymbol{f}(h) \cdot \boldsymbol{\psi}^{(d)}(g^{-1} \circ h) \tag{1}$$

As shown in Cohen and Welling (2016, § 6.1), G-convolutions are G-equivariant.

(2) G-Embed. It is often desirable to embed input in a vector space – but for a G-equivariant network we require this step too to be equivariant. Gordon et al. (2020) show that a G-equivariant embedding can be obtained as a special case of a G-convolution where the input function for input word x is a one-hot encoding, $x : \{1, \ldots, |\Sigma|\} \to \{0, 1\}$. Thus the learnable filter functions will be one-dimensional, and there is no longer a need for the sum, as it is zero at all but one value. We call our set of K filter functions $\{\omega^{(k)}\}_{k=1}^K$, with $\omega^{(k)} : \{1, \ldots, |\Sigma|\} \to \mathbb{R}$. Then the G-equivariant embedding of an input x is a $|G| \times K$ matrix $e(x)$ with entries

$$e(x)_{g,k} = G\text{-Embed}(x; \boldsymbol{\omega})_{g,k} = \omega^{(k)}(g^{-1} \circ x) \tag{2}$$

[1] If $x_n \in \Sigma \setminus L^{\text{in}}_{\text{equi}}$, $g \circ x_n = x_n$.

③ **G-Decode.** The output $\phi : G \to \mathbb{R}^D$ of some composition of equivariant layers based on G-convolutions will be a function over group elements, so in order to calculate logits over an output distribution of items Δ, Gordon et al. (2020) use a decoding layer

$$G\text{-Dec}(\boldsymbol{\phi}, \widetilde{y}; \boldsymbol{\rho}) = \sum_{h \in G} \boldsymbol{\phi}(h) \cdot \boldsymbol{\rho}(h^{-1} \circ \widetilde{y}) \quad (3)$$

where $\widetilde{y} \in \Delta$ is a candidate output value and $\boldsymbol{\rho} : \{1, \ldots, |\Delta|\} \to \mathbb{R}^D$ is a learnable filter function. This layer is not a form of group convolution, but instead is equivariant due to parameter-sharing (Ravanbakhsh et al., 2017).

4 An Equivariant Transducer

We now describe the architecture of our equivariant hard alignment transducer. Let Σ be an input alphabet and Δ be an output alphabet. Given an input string $\mathbf{x} \in \Sigma^*$ we create a model to calculate a probability distribution $p(\mathbf{y} \mid \mathbf{x})$ over $\mathbf{y} \in \Delta^*$. We consider what it means for such a model to be equivariant with the following theorem.

Theorem 1. *Let $p(\mathbf{y} \mid \mathbf{x})$ be a probability distribution over Δ^*. If $p(g \circ \mathbf{y} \mid g \circ \mathbf{x}) = p(\mathbf{y} \mid \mathbf{x})$ for any $g \in G, \mathbf{x} \in \Sigma^*, \mathbf{y} \in \Delta^*$, then the transducer $\xi : \Sigma^* \to \Delta^*$ defined by $\xi(\mathbf{x}) = \text{argmax}_{\mathbf{y} \in \Delta^*} p(\mathbf{y} \mid \mathbf{x})$ is equivariant.*

Proof. Given an $\mathbf{x}$, let $\mathbf{y}^* = \xi(\mathbf{x}) = \text{argmax}_{\mathbf{y} \in \Delta^*} p(\mathbf{y} \mid \mathbf{x})$. Now $\xi(g \circ \mathbf{x}) = \text{argmax}_{\mathbf{y} \in \Delta^*} p(\mathbf{y} \mid g \circ \mathbf{x})$. But for any $\mathbf{y} \in \Delta^*$, $p(\mathbf{y} \mid g \circ \mathbf{x}) = p(g^{-1} \circ \mathbf{y} \mid g^{-1} \circ g \circ \mathbf{x}) = p(g^{-1} \circ \mathbf{y} \mid \mathbf{x})$, so $\xi(g \circ \mathbf{x}) = \text{argmax}_{\mathbf{y} \in \Delta^*} p(g^{-1} \circ \mathbf{y} \mid \mathbf{x})$. We know that the probability distribution conditioned on $\mathbf{x}$ is maximized by $\mathbf{y}^*$, so we have $g^{-1} \circ \xi(g \circ \mathbf{x}) = \mathbf{y}^*$ and thus $\xi(g \circ \mathbf{x}) = g \circ \mathbf{y}^*$ and the transducer is equivariant. □

Moving forward we will use this equivalent definition to discuss the equivariance of our model.

4.1 A Hard-Alignment Transducer

As stated above, we aim to construct a distribution $p(\mathbf{y} \mid \mathbf{x})$ where $\mathbf{y} \in \Delta^*$ and $\mathbf{x} \in \Sigma^*$. The key idea behind a hard-alignment model is that the two strings $\mathbf{x}$ and $\mathbf{y}$ are aligned according to a latent alignment $\mathbf{a} \in \mathcal{A}(\mathbf{x}, \mathbf{y})$. Hard alignment is contrasted with a soft alignment (Bahdanau et al., 2015; Luong et al., 2015), which does not have the interpretation as a latent variable. For $\mathbf{x}$ of length N, $\mathbf{y}$ of length M, each $\mathbf{a}$ is a vector in $\{1, \ldots, N\}^M$. If we have $a_m = n$, then the input word x_n and output word y_m are aligned. Thus, $\mathcal{A}(\mathbf{x}, \mathbf{y})$ consists of all non-monotonic alignments such that each word in $\mathbf{y}$ aligns to exactly one word in $\mathbf{x}$. Because there is no annotation for alignments between the strings, we marginalize over alignments to compute the distribution:

$$p(\mathbf{y} \mid \mathbf{x}) = \sum_{\mathbf{a} \in \mathcal{A}(\mathbf{x}, \mathbf{y})} p(\mathbf{y}, \mathbf{a} \mid \mathbf{x}) \quad (4)$$

$$= \sum_{\mathbf{a} \in \mathcal{A}(\mathbf{x}, \mathbf{y})} \prod_{m=1}^{M} p(y_m \mid a_m, \mathbf{y}_{<m}, \mathbf{x})\, p(a_m \mid \mathbf{y}_{<m}, \mathbf{x})$$

Wu et al. (2018) show that eq. (4) may be rewritten as follows using the distributive property

$$\prod_{m=1}^{M} \sum_{a_m=1}^{N} \underbrace{p(y_m \mid a_m, \mathbf{y}_{<m}, \mathbf{x})}_{\textit{translator}} \underbrace{p(a_m \mid \mathbf{y}_{<m}, \mathbf{x})}_{\textit{aligner}} \quad (5)$$

which is more efficient to compute. Specifically, this allows the distribution $p(\mathbf{y} \mid \mathbf{x})$ to be computed $\mathcal{O}(M \cdot N)$ rather than $\mathcal{O}(|\mathcal{A}(\mathbf{x}, \mathbf{y})|)$, which is exponential in both M and N.

This formulation allows us to completely separate the alignment probability $p(a_m \mid \mathbf{y}_{<m}, \mathbf{x})$, which calculates how likely an output word is to align with each input word, from the word translation probability $p(y_m \mid a_m, \mathbf{y}_{<m}, \mathbf{x})$, which calculates how likely each output word is to be produced when aligned with a given input word. This can also be viewed as separately treating the syntax (alignment probability) and semantics (word translation probability) of the input. In this sense, the model is similar to that of Russin et al. (2019).

We also experimented with a variation on this formulation in which the sum over a_m is replaced with taking the maximum, as well as an annealed variation. This is described in App. A.

4.2 An Equivariant Translator

We now explain how the translator term in eq. (5) is defined in order to ensure group-equivariance. Because y_m and x_{a_m} are both individual words—as opposed to full sentences—the distribution $p(y_m \mid x_{a_m}, a_m)$ is a simple classifier. We use a composition of the G-equivariant layers described in §3.4, so that

$$p(y_m \mid x_{a_m}, a_m) = \frac{\exp\left(G\text{-Dec}(\boldsymbol{\phi}, y_m)\right)}{\sum_{y' \in \Delta} \exp\left(G\text{-Dec}(\boldsymbol{\phi}, y')\right)}$$

where $\boldsymbol{\phi} = G\text{-Conv}(e(x_{a_m}))$. So, for each input word x_{a_m}, we first obtain a G-Embedding, then pass this through a G-Convolution, then use G-Decode to obtain logits over possible output words, which are then fed through a softmax to obtain probabilities.[2]

4.3 An Invariant Aligner

The alignment term is parameterized using a recurrent neural network. Before encoding, we replace each word with a symbol indicating its lexical class. Formally, if we have I disjoint lexical classes $L_1^{\text{in}}, \ldots, L_I^{\text{in}} \subseteq \Sigma$ that cover Σ and J disjoint lexical classes $L_1^{\text{out}}, \ldots, L_J^{\text{out}} \subseteq \Delta$ that cover Δ, we define functions $\ell_\Sigma : \Sigma \to \{1, \ldots, I\}$ and $\ell_\Delta : \Delta \to \{1, \ldots, J\}$ such that $\ell_\Sigma(w) = i$ iff $w \in L_i^{\text{in}}$, $\ell_\Delta(w) = j$ iff $w \in L_j^{\text{out}}$. We overload these so that for $\mathbf{x} \in \Sigma^*$ of length N, $\ell_\Sigma(\mathbf{x}) = (\ell_\Sigma(x_1), \ldots, \ell_\Sigma(x_N))$ and analogously $\ell_\Delta(\mathbf{y}) = (\ell_\Delta(y_1), \ldots, \ell_\Delta(y_M))$ for $\mathbf{y} \in \Delta^*$ of length M. Then, our model is actually dependent on $\ell_\Sigma(\mathbf{x})$ and $\ell_\Delta(\mathbf{y}_{<m})$. This has the effect of delexicalizing the input and equivalence-classing the words. For example, if the verb and direction lexical classes are being considered, the input `walk left after run` will, in effect, be represented as `<verb> <direction> after <verb>`. This substitution imbues the alignment model with some useful theoretical properties that will be discussed in §4.4.

The delexicalized input sequence $\ell_\Sigma(\mathbf{x})$ of length N is encoded both forwards and backwards using an LSTM to produce hidden states $\overrightarrow{\boldsymbol{h}}_n^{(\text{enc})}, \overleftarrow{\boldsymbol{h}}_n^{(\text{enc})} \in \mathbb{R}^{d_h}$ for $n \in \{1, \ldots, N\}$. These are then concatenated to obtain $\boldsymbol{h}_n^{(\text{enc})} = \overrightarrow{\boldsymbol{h}}_n^{(\text{enc})} \bigoplus \overleftarrow{\boldsymbol{h}}_n^{(\text{enc})} \in \mathbb{R}^{2d_h}$. The output sequence $\ell_\Delta(\mathbf{y})$ of length M is encoded forwards to produce $\overrightarrow{\boldsymbol{h}}_m^{(\text{dec})} \in \mathbb{R}^{d_h}$ for $m \in \{1, \ldots, M\}$. Then, we have the following distribution

$$p(a_m \mid \ell_\Delta(\mathbf{y}_{<m}), \ell_\Sigma(\mathbf{x})) = \frac{\exp(e_{ma_m})}{\sum_{n=1}^{N} \exp(e_{mn})} \tag{6}$$

where

$$e_{mn} = \overrightarrow{\boldsymbol{h}}_m^{(\text{dec})\top} \boldsymbol{T} \boldsymbol{h}_n^{(\text{enc})} \tag{7}$$

and $\boldsymbol{T} \in \mathbb{R}^{d_h \times 2d_h}$ is a learned matrix.

4.4 Theoretical Results and Discussion

We know that each word-to-word translator $p(y_m \mid x_{a_m}, a_m)$ is G-equivariant. We now explore in more detail the theoretical properties of the entire model. We begin by considering the properties of the alignment model.

Definition 7. *We say that a function $f : X \times Y \to \mathbb{R}$ is invariant to a group G if for any $x \in X, y \in Y, g \in G$, we have $f(g \circ x, g \circ y) = f(x, y)$.*

Theorem 2. *The alignment model $p(a_m \mid \ell_\Delta(\mathbf{y}_{<m}), \ell_\Sigma(\mathbf{x}))$ defined in §4.3 is invariant to G.*

Proof. Let us take a group element $g \in G$. Then $\ell_\Sigma(g \circ \mathbf{x}) = \ell_\Sigma(\mathbf{x})$. We can see this because $g \circ \mathbf{x} = (g \circ x_1, \ldots, g \circ x_N)$ and g only permutes each word *within* its lexical class. Words within the same lexical class are mapped to the same value by ℓ_Σ. A similar argument shows that $\ell_\Delta(g \circ \mathbf{y}_{<m}) = \ell_\Delta(\mathbf{y}_{<m})$. So $p(a_m \mid \ell_\Delta(g \circ \mathbf{y}_{<m}), \ell_\Sigma(g \circ \mathbf{x})) = p(a_m \mid \ell_\Delta(\mathbf{y}_{<m}), \ell_\Sigma(\mathbf{x}))$ and the model is invariant. □

We can now examine the properties of the transducer as a whole.

Theorem 3. *The model $p(\mathbf{y} \mid \mathbf{x})$ defined in eq. (5) is G-equivariant.*

Proof. We take a group element $g \in G$. Then

$$\begin{aligned} &p(g \circ \mathbf{y} \mid g \circ \mathbf{x}) \\ &\quad = \prod_{m=1}^{M} \sum_{a_m=1}^{N} p(g \circ y_m \mid a_m, g \circ x_{a_m}) \\ &\qquad\qquad p(a_m \mid \ell_\Delta(g \circ \mathbf{y}_{<m}), \ell_\Sigma(g \circ \mathbf{x})) \end{aligned}$$

We have seen that the translator part of the model is equivariant to G, and the alignment part is invariant, so we can rewrite the above as

$$\prod_{m=1}^{M} \sum_{a_m=1}^{N} p(y_m \mid a_m, x_{a_m}) p(a_m \mid \ell_\Delta(\mathbf{y}_{<m}), \ell_\Sigma(\mathbf{x}))$$

which is equal to $p(\mathbf{y} \mid \mathbf{x})$ and thus the model is G-equivariant. □

We would like to understand what these equivariance properties actually mean for our model's ability to generalize to unseen sentences. To aid in the discussion of this question, we define the following.

Definition 8. *Given a model $p(\mathbf{y} \mid \mathbf{x})$ that is equivariant to a group G and a sentence pair $(\mathbf{x}, \mathbf{y}) \in \Sigma^* \times \Delta^*$, we define the **theoretical orbit** of $(\mathbf{x}_0, \mathbf{y}_0)$ under the model as $\Omega_T((\mathbf{x}_0, \mathbf{y}_0)) = \{(g \circ \mathbf{x}_0, g \circ \mathbf{y}_0) \mid g \in G\}$. For any $(x_i, y_i) \in \Omega_T((\mathbf{x}_0, \mathbf{y}_0))$, $p(\mathbf{y}_i \mid \mathbf{x}_i) = p(\mathbf{y}_0 \mid \mathbf{x}_0)$.*

[2] Additional non-linearities are introduced by passing the output of each layer through a tanh function.

If we maximize $p(\mathbf{y}_0 \mid \mathbf{x}_0)$ for a training pair $(\mathbf{x}_0, \mathbf{y}_0)$, any benefits of this are shared by elements of its theoretical orbit, allowing the model to generalize to these pairs. So the size of the theoretical orbit of a sentence pair is a way of quantifying how widely a model can generalize from that pair.

In the case of our model, with the equivariant lexical class corresponding to the 4 SCAN verbs, any sentence pair containing one of these verbs has a theoretical orbit of size 4. If $(\mathbf{x}_0, \mathbf{y}_0)$ is a sentence pair containing two different SCAN verbs – for example, (`walk after run`, RUN WALK) – this means that its theoretical orbit contains 4 sentences out of the 16 sentences of this form that can be constructed. In addition to the theoretical orbit, which is guaranteed by the properties of the model, we also consider the behaviour of a trained model in practice.

Definition 9. *Given a model $p(\mathbf{y} \mid \mathbf{x})$ that is equivariant to a group G and a sentence pair $(\mathbf{x}_0, \mathbf{y}_0) \in \Sigma^* \times \Delta^*$, we define the **observed orbit** of $(\mathbf{x}_0, \mathbf{y}_0)$ under the model as $\Omega_O((\mathbf{x}_0, \mathbf{y}_0)) = \{(\mathbf{x}, \mathbf{y}) \in \Sigma^* \times \Delta^* \mid -\log p(\mathbf{y} \mid \mathbf{x}) = -\log p(\mathbf{y}_0 \mid \mathbf{x}_0)\}$. Clearly $\Omega_T((\mathbf{x}_0, \mathbf{y}_0)) \subseteq \Omega_O((\mathbf{x}_0, \mathbf{y}_0))$.*

To better understand how our model behaves, we examined the observed orbit of one of our best-performing models throughout training. We selected sentence pairs from SCAN containing 2 verbs, and generated all possible sentence pairs of the same form. For each form, there are 16 such pairs. While training our model, every five epochs we recorded the negative log-likelihood of each sentence pair as given by the model and counted how many of the 16 pairs had the same negative log-likelihood to find the size of the observed orbits.[3] We show the results for one sentence pair for the first 100 epochs of training in Figure 2. We can see that, initially, the 16 sentence pairs with the same form are in 4 distinct orbits of size 4, as predicted theoretically. However, as the model trains, all 16 sentences come to have the same negative log-likelihood, and this continues for the remaining duration of training.

To better understand why the observed orbit under our model is larger than predicted theoretically, we examine the effect of alignments in more depth. We consider a gold alignment for a SCAN sentence pair $(\mathbf{x}, \mathbf{y})$ to be an alignment $\mathbf{a}^*$ such that words from corresponding lexical classes are aligned and each output word is aligned with an input word with the corresponding meaning (e.g., LTURN aligned with `left`, WALK aligned with `walk`).

Given $\mathbf{x} \in \Sigma^N$, $\mathbf{y} \in \Delta^M$ and an alignment $\mathbf{a}$ between them, consider a group G^N with elements $g = (g_1, \ldots, g_N)$ with $g_n \in G$. Define its action on $\mathbf{x}$ and $\mathbf{y}$ by $g \circ \mathbf{x} = (g_1 \circ x_1, \ldots, g_N \circ x_N)$ and $g \circ_{\mathbf{a}} \mathbf{y} = (g_{a_1} \circ y_1, \ldots, g_{a_M} \circ y_M)$. Note that the group G^N depends on the length of the string, and its action on $\mathbf{y} \in \Delta^M$ depends on the alignment.

Theorem 4. *Given $\mathbf{x} \in \Sigma^N$, $\mathbf{y} \in \Delta^M$ and a gold alignment $\mathbf{a}^* \in \mathcal{A}(\mathbf{x}, \mathbf{y})$ between the two, $p(g \circ_{\mathbf{a}^*} \mathbf{y} \mid g \circ \mathbf{x}, \mathbf{a}^*) = p(\mathbf{y} \mid \mathbf{x}, \mathbf{a}^*)$ for all $g \in G^N$, with actions defined as above.*

Proof. For any $g \in G^N$, $p(g \circ_{\mathbf{a}^*} \mathbf{y} \mid g \circ \mathbf{x}, \mathbf{a}^*) = \prod_{m=1}^{M} p(g_{a^*_m} \circ y_m \mid g_{a^*_m} \circ x_{a^*_m}) = \prod_{m=1}^{M} p(y_m \mid x_{a^*_m}) = p(\mathbf{y} \mid \mathbf{x}, \mathbf{a}^*)$. □

This means that for $\mathbf{x}_0 \in \Sigma^N$, $\mathbf{y}_0 \in \Delta^M$ and a gold alignment $\mathbf{a}^*$, the model is equivariant to G^N and thus the theoretical orbit of a sentence pair $(\mathbf{x}_0, \mathbf{y}_0)$ under the model is $|G|^N$ rather than $|G|$ as it was without conditioning on the alignment.[4] Although this may seem irrelevant since we do not condition on gold alignments, we find that our trained models do approximate this. As training progresses, our model gradually becomes more confident about the correct alignment between the input and output. Once the model has confidently learned the gold alignment $\mathbf{a}^*$ between an input–output pair $\mathbf{x} \in \Sigma^*, \mathbf{y} \in \Delta^*$, $p(a_m \mid \mathbf{y}_{<m}, \mathbf{x})$ becomes very close to 0 for $a_m \neq a^*_m$ and very close to 1 for $a_m = a^*_m$, meaning that $p(\mathbf{y} \mid \mathbf{x})$ approaches $p(\mathbf{y} \mid \mathbf{x}, \mathbf{a}^*) = \prod_{m=1}^{M} p(y_m \mid x_{a^*_m})$. This means that for an input pair $(\mathbf{x}_0, \mathbf{y}_0)$, once the model has a high degree of confidence about the gold alignment $\mathbf{a}^*$ and $p(\mathbf{y}_0 \mid \mathbf{x}_0)$ approaches $p(\mathbf{y}_0 \mid \mathbf{x}_0, \mathbf{a}^*)$, the observed orbit of $(\mathbf{x}_0, \mathbf{y}_0)$ will become this larger theoretical orbit. This explains what is shown in Figure 2.

5 Experiments and Results

Experiments were performed on the Simple, Add Jump, Around Right and Length splits of SCAN. In each case, models were trained on 90% of the train

[3] The values were considered to be equal if they were within floating point error, as checked by `torch.isclose()`.

[4] Unlike previous examples, theorem 4 only *directly* implies the G^N-equivariance of the restricted transducer $\xi(\mathbf{x} \mid \mathbf{a}^*) = \operatorname{argmax}_{\mathbf{y} \in \Delta^M} p(\mathbf{y} \mid \mathbf{x}, \mathbf{a}^*)$, since the action of G^N is only defined for $\mathbf{y} \in \Delta^M$. However, it does in fact imply the equivariance of the unrestricted transducer, since conditioning on the alignment means that $p(\mathbf{y} \mid \mathbf{x}, \mathbf{a}^*) = 0$ for $\mathbf{y} \in \Delta^Q$ if $Q \neq M$.

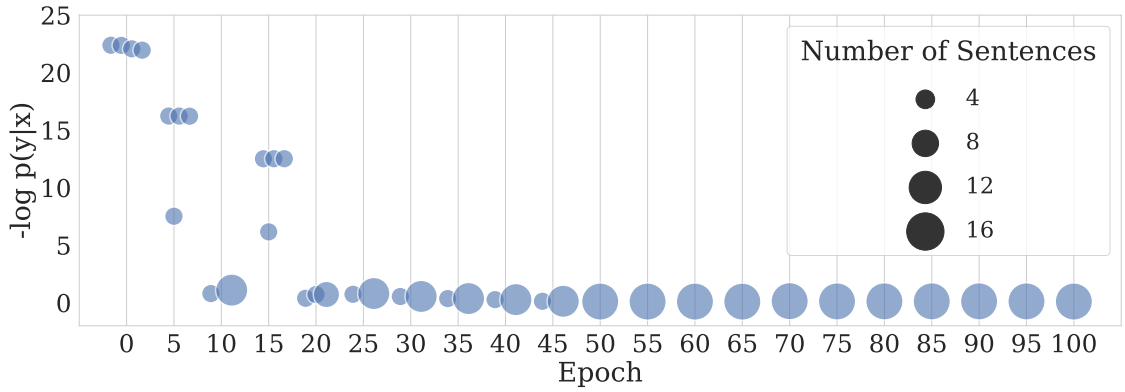

Figure 2: Observed orbits of sentence pairs $(\mathbf{x}, \mathbf{y})$ during the first 100 epochs of training, with $\mathbf{x}$ of the form <verb$_1$> right thrice after <verb$_2$> and $\mathbf{y}$ of the form <VERB$_2$> RTURN <VERB$_1$> RTURN <VERB$_1$> RTURN <VERB$_1$>.

Model	Simple	Add Jump	Around Right	Length
Russin et al. (2019)	100.0	91.0	28.9	15.2
Liu et al. (2020)	100.0	100.0	100.0	100.0
Gordon et al. (2020)	100.0	99.1	92.0	15.9
Equivariant Hard Alignment	100.0	100.0	100.0	28.5

Table 1: Accuracy achieved on SCAN task, presented alongside results from state-of-the-art systems

set, by minimizing the negative log-likelihood of observed sentence pairs, with 10% of the train set reserved to be used as a validation set. Models were selected to have the lowest loss on the validation set, and then evaluated on the test set.

For the Add Jump split, the group used was the cyclic shift group of size 4 acting on the set of SCAN verbs. For the Around Right split, which withholds the combination around right, the group used was the cyclic shift group of size 2 acting on the set of directions. Hyperparameters were selected through a random hyperparameter search. At test time, outputs were decoded using a beam search with 3 beams.

Results are shown in Table 1. We can see that our model outperforms Gordon et al.'s (2020) similar group-equivariant model on all splits, achieving 100% accuracy on all but the Length split, demonstrating empirically the advantage of the wider generalization capabilities that we explained theoretically. It still falls short of Liu et al.'s (2020) state-of-the-art memory-augmented model on the Length split. Since the equivariance of our model primarily targets systematicity, which is not the ability targeted by the Length split, it is not surprising that it does not perform at state-of-the-art levels in this split. We note, however, that our model does substantially outperform both Gordon et al.'s (2020) group-equivariant model, and Russin et al.'s (2019) semantic-syntactic alignment model on this split, which are the two models most similar to our own.

6 Limitations and Future Work

Group-equivariant networks have not yet been widely adopted in NLP, and we feel that there is potential for more tasks to be identified that could benefit from the application of group equivariance. A further line of investigation would be to develop and assess group-equivariant versions of a wider variety of architectures. The model used by Gordon et al. (2020) is an equivariant version of a standard LSTM-based sequence-to-sequence model, and the model used in our work is based on the hard alignment model used by Wu et al. (2018) – these are only two of many possible architectures. Variants of other architectures may possess different equivariance properties from both of these models, and may perform well on different tasks. Future work could incorporate group equivariance into more architectures and assess their theoretical properties and empirical performance.

A key limitation of group-equivariant networks is the need to understand and specify the group to which it is equivariant. This makes it difficult

to apply these networks to real world tasks rather than artificial datasets such an SCAN. Future work could investigate ways to identify from data which words should be in a lexical class, or ways to allow group-equivariant networks to deal with open vocabulary problems. Some work has been done on learning input and output vocabulary alignments in the context of SCAN (Akyurek and Andreas, 2021), so it is possible that this could be used to improve group-equivariant architectures by reducing or eliminating their reliance on lexical classes and groups known *a priori*.

7 Conclusion

In this work, we proposed and implemented a string-to-string transduction model which combines a group-invariant sum over hard alignments and a group-equivariant output word probability to create a model which is equivariant to the swapping of words in the same lexical class. We applied this model to the SCAN task, finding that it is successful in allowing the model to generalize compositionally. We show theoretically that our model's structure allows for wider generalization to novel sentences than existing group-equivariant approaches and demonstrate this empirically. We suggest that this is strong motivation to explore group-equivariant variants of other architectures, and to investigate other tasks which may benefit from group-equivariant models, as well as suggesting that theoretical analysis of equivariance properties may be a useful tool in understanding performance differences.

References

Ekin Akyurek and Jacob Andreas. 2021. Lexicon learning for few shot sequence modeling. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 4934–4946, Online. Association for Computational Linguistics.

Dzmitry Bahdanau, Kyung Hyun Cho, and Yoshua Bengio. 2015. Neural machine translation by jointly learning to align and translate. In *International Conference on Learning Representations*.

Dzmitry Bahdanau, Harm de Vries, Timothy J O'Donnell, Shikhar Murty, Philippe Beaudoin, Yoshua Bengio, and Aaron Courville. 2019. Closure: Assessing systematic generalization of clevr models. *arXiv preprint arXiv:1912.05783*.

Noam Chomsky. 1957. *Syntactic Structures*. De Gruyter Mouton.

Taco Cohen and Max Welling. 2016. Group equivariant convolutional networks. In *Proceedings of The 33rd International Conference on Machine Learning*, volume 48 of *Proceedings of Machine Learning Research*, pages 2990–2999, New York, New York, USA. PMLR.

Jonathan Gordon, David Lopez-Paz, Marco Baroni, and Diane Bouchacourt. 2020. Permutation equivariant models for compositional generalization in language. In *International Conference on Learning Representations*.

Dieuwke Hupkes, Verna Dankers, Mathijs Mul, and Elia Bruni. 2020. Compositionality decomposed: How do neural networks generalise? In *Proceedings of the Twenty-Ninth International Joint Conference on Artificial Intelligence, IJCAI-20*, pages 5065–5069. International Joint Conferences on Artificial Intelligence Organization.

Diederik P. Kingma and Jimmy Ba. 2015. Adam: A method for stochastic optimization. In *International Conference on Learning Representations*.

Risi Kondor and Shubhendu Trivedi. 2018. On the generalization of equivariance and convolution in neural networks to the action of compact groups. In *Proceedings of the 35th International Conference on Machine Learning*, volume 80 of *Proceedings of Machine Learning Research*, pages 2747–2755. PMLR.

Brenden Lake and Marco Baroni. 2018. Generalization without systematicity: On the compositional skills of sequence-to-sequence recurrent networks. In *Proceedings of the 35th International Conference on Machine Learning*, volume 80 of *Proceedings of Machine Learning Research*, pages 2873–2882. PMLR.

Brenden M. Lake. 2019. Compositional generalization through meta sequence-to-sequence learning. In *Advances in Neural Information Processing Systems*.

Qian Liu, Shengnan An, Jian-Guang Lou, Bei Chen, Zeqi Lin, Yan Gao, Bin Zhou, Nanning Zheng, and Dongmei Zhang. 2020. Compositional generalization by learning analytical expressions. In *Advances in Neural Information Processing Systems*, volume 33, pages 11416–11427. Curran Associates, Inc.

Thang Luong, Hieu Pham, and Christopher D. Manning. 2015. Effective approaches to attention-based neural machine translation. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 1412–1421, Lisbon, Portugal. Association for Computational Linguistics.

Richard Montague. 1970. Universal grammar. *Theoria*, 36(3):373–398.

Siamak Ravanbakhsh, Jeff Schneider, and Barnabás Póczos. 2017. Equivariance through parameter-sharing. In *Proceedings of the 34th International Conference on Machine Learning*, volume 70 of *Proceedings of Machine Learning Research*, pages 2892–2901. PMLR.

Laura Ruis, Jacob Andreas, Marco Baroni, Diane Bouchacourt, and Brenden M Lake. 2020. A benchmark for systematic generalization in grounded language understanding. In *Advances in Neural Information Processing Systems*, volume 33, pages 19861–19872.

Jake Russin, Jason Jo, Randall C. O'Reilly, and Yoshua Bengio. 2019. Compositional generalization in a deep seq2seq model by separating syntax and semantics. *arXiv preprint arXiv:1904.09708*.

Shijie Wu, Pamela Shapiro, and Ryan Cotterell. 2018. Hard non-monotonic attention for character-level transduction. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 4425–4438, Brussels, Belgium. Association for Computational Linguistics.

A Model Variants

In addition to the sum-based model presented in the main body of the paper, we also experimented on some variations of the model, which we explain here.

A.1 Max Model

The max version of the model is given by

$$p(\mathbf{y} \mid \mathbf{x}) = \prod_{m=1}^{M} \max_{1 \leq a_m \leq N} p(y_m \mid a_m, \mathbf{y}_{<m}, \mathbf{x})\, p(a_m \mid \mathbf{y}_{<m}, \mathbf{x}) \tag{8}$$

where all terms are the same as for the sum model.

A.2 Annealed Max Model

This model proved difficult to train, so we also conducted experiments with an annealed max model, which is given by

$$p(\mathbf{y} \mid \mathbf{x}) = \prod_{m=1}^{M} \sum_{a_m=1}^{N} \alpha_{a_m}^{m} p(y_m \mid a_m, \mathbf{y}_{<m}, \mathbf{x}) p(a_m \mid \mathbf{y}_{<m}, \mathbf{x}) \tag{9}$$

where $\boldsymbol{\alpha^m}$ is given by

$$\alpha_{a_m}^{m} = \frac{\exp\left(\frac{1}{\tau} p(y_m \mid a_m, \mathbf{y}_{<m}, \mathbf{x}) p(a_m \mid \mathbf{y}_{<m}, \mathbf{x})\right)}{\sum_{a'_m=1}^{N} \left(\frac{1}{\tau} p(y_m \mid a'_m, \mathbf{y}_{<m}, \mathbf{x}) p(a'_m \mid \mathbf{y}_{<m}, \mathbf{x})\right)} \tag{10}$$

where $0 < \tau \leq 1$ is the temperature. As τ approaches 0, $\boldsymbol{\alpha^m}$ approaches a one-hot vector indicating the argmax of $p(y_m \mid a_m, \mathbf{y}_{<m}, \mathbf{x}) p(a_m \mid \mathbf{y}_{<m}, \mathbf{x})$, and thus eq. (9) becomes close to eq. (8). During training, τ is gradually decreased, so that the fully trained model is, in effect, a max model. The starting value of τ, as well as the schedule on which it is decreased, are hyperparameters.

A.3 Model Comparison

In Table 2 we show the best accuracy achieved on each split by each variant of our model.

Model	Simple	Add Jump	Around Right	Length
Sum	100.0	100.0	100.0	28.5
Max	100.0	99.9	99.9	18.3
Annealed Max	99.9	99.9	99.9	19.7

Table 2: Accuracy achieved on SCAN task by each variant of our model

B Reproducibility

The model was optimized using Adam (Kingma and Ba, 2015) with a learning rate of 0.001. We chose hyperparameters through a random search. Each hyperparameter included in the search in described in Table 3 along with the range that was searched. For each split, sets of hyperparameters were randomly sampled and models with those hyperparameters were trained and evaluated. Table 4 shows the value of the hyperparameters for the best-achieving model on each split. In Table 5 and Table 6 we also include the best-performing hyperparameters for the max and annealed variations of the model.

Hyperparameter	Details	Range
Dimension of G-Embedding	Size of G-Embedding layer	5-256
Number of Filters	Number of filters used in G-Convolution layer	5-256
Embedding Dimension	Dimension of embedding layer in alignment model	5-256
Hidden Size	Size of LSTM layers used in alignment model	5-256
Batch Size	-	8 - 64

Table 3: Hyperparameter ranges that were searched

Split	Dimension of G-Embedding	Number of filters	Embedding Dimension	Hidden Size	Batch Size
Simple	20	24	36	6	8
Simple	6	13	67	13	8
Add Jump	122	7	223	67	8
Around Right	100	7	122	36	8
Around Right	20	20	9	55	8
Around Right	45	182	100	9	8
Length	45	24	11	149	32

Table 4: Values of hyperparameters for the best-performing sum-model on each split tested. Where more than one set of hyperparameters is given for a split, both performed equally well.

Split	Dimension of G-Embedding	Number of filters	Embedding Dimension	Hidden Size	Batch Size
Simple	16	24	182	36	8
Add Jump	30	11	182	55	16
Around Right	30	223	122	11	8
Length	36	55	30	4	16

Table 5: Values of hyperparameters for the best-performing max-model on each split tested.

Split	Dimension of G-Embedding	Number of filters	Embedding Dimension	Hidden Size	Batch Size
Simple	182	122	223	11	8
Add Jump	122	7	223	67	8
Around Right	55	100	45	30	8
Length	13	9	11	16	16

Table 6: Values of hyperparameters for the best-performing annealed max-model on each split tested.

Training Data Percentage	Sum Model	Max Model	Annealed Model	Gordon et al.
1	42.14	0.47	33.26	**45.75**
2	68.09	56.28	58.36	**73.53**
4	**89.59**	9.2	81.22	89.54
8	94.67	**99.23**	95.61	96.60
16	98.88	**99.48**	98.44	97.12
32	**99.86**	99.48	97.41	97.47
64	99.67	**99.88**	99.27	99.67

Table 7: Accuracy obtained by each model in low-data conditions

C Low-Data Experiments

To test our model's ability to learn from small amounts of training data, we trained and evaluated a model with the best-performing hyperparameters on the Simple split using the low-data splits of SCAN. These splits have training sets containing 1%, 2%, 4%, 8%, 16%, 32% and 64% of the total examples in the Simple split. For comparison, we repeated this for all variants of our model, as well as for the equivariant model used by Gordon et al. (2020). The results are shown in Table 7. We can see that in the lowest data conditions, our model doesn't generalize as well as Gordon et al.'s (2020) model. For the sets containing 16% or more or the data, our model performs better. We theorize that this is due to difficulty learning a high-quality alignment in the lowest-data conditions. These results also show the volatility of the Max variant of our model – it is able to reach above 99% accuracy with less data than any of the other models, but with any less data it struggles to learn much at all.

D SCAN Lexicon

For further context we provide lists of the full SCAN lexicon for input and output in Table 8 and Table 9 respectively. In Table 10 we list the words in the Direction lexical class in the input and output lexicon. In Table 11 we do the same for the Verbs lexical class. All other input words were in single-item lexical classes.

Input Words
run
walk
look
jump
left
right
after
and
turn
around
twice
thrice
opposite
around

Table 8: All input vocabulary for the SCAN task

Output Words
RUN
WALK
LOOK
JUMP
LTURN
RTURN

Table 9: All output vocabulary for the SCAN task

Input	Output
right	RTURN
left	LTURN

Table 10: Direction lexical class in input and output vocabulary

Input	Output
run	RUN
walk	WALK
look	LOOK
jump	JUMP

Table 11: Verb lexical class in input and output vocabulary

Where Does Linguistic Information Emerge in Neural Language Models? Measuring Gains and Contributions Across Layers

Jenny Kunz and **Marco Kuhlmann**
Dept. of Computer and Information Science
Linköping University
jenny.kunz@liu.se and marco.kuhlmann@liu.se

Abstract

Probing studies have extensively explored where in neural language models linguistic information is located. The standard approach to interpreting the results of a probing classifier is to focus on the layers whose representations give the highest performance on the probing task. We propose an alternative method that asks where the task-relevant information emerges in the model. Our framework consists of a family of metrics that explicitly model local information gain relative to the previous layer and each layer's contribution to the model's overall performance. We apply the new metrics to two pairs of syntactic probing tasks with different degrees of complexity and find that the metrics confirm the expected ordering only for one of the pairs. Our local metrics show a massive dominance of the first layers, indicating that the features that contribute the most to our probing tasks are not as high-level as global metrics suggest.

1 Introduction

Probing neural language models aims at finding evidence of learned linguistic structure in the models' parameters by empirically testing hypotheses about the learned representations (Hupkes et al., 2018; Alain and Bengio, 2017). This is often done by training a probing classifier on a diagnostic task with the representations at different layers as the input, and comparing task performance across layers. While probes are conceptually simple and widely used, the methodology and in particular the interpretation of the obtained results is subject to ongoing discussion (Belinkov, 2022).

A classical pattern we often see when plotting probing accuracy across layers is that for higher-level linguistic tasks, the model aggregates information over several layers until it reaches its highest performance. Often, the curves start steep, flatten out, and eventually drop again in the final layers.

Figure 1: Heatmaps illustrating our results for syntactic parent (P) and grandparent (GP) prediction (BERT-base, en, layers 1–12): Global metrics peak in middle layers. Local contributions are concentrated in early layers. (Darker shades indicate higher values.)

In this paper, we zoom in on models' *relative* information gains between one layer and the previous one. By this, we aim to turn the focus of probing away from information that is already present in the non-contextualized embedding layer and focus exclusively on information that needs context to be retrieved. We also aim to make the contribution of each layer to the model's overall linguistic capabilities explicit. We argue that under the hypothesis that information in language models is structured as in classical NLP pipelines (Tenney et al., 2019), the *depth* at which information emerges is as important as the *impact* of that information on model performance. When we think of language processing as a pipeline, we are primarily interested in where linguistic features emerge for the first time, not how long they are passed on to later layers.

To formalize information gains, we modify the conditional probing framework proposed by Hewitt et al. (2021). Their method explicitly quantifies information that is not already present in a baseline representation. We modify this method along two lines: First, we make it *local*, by conditioning on the respective previous layer instead of a global baseline. Second, we report the results as a *share* of total emergent information, across all layers of the network. This makes layer contributions comparable across tasks, where the overall performance may differ.

Proceedings of the 29th International Conference on Computational Linguistics, pages 4664–4676
October 12–17, 2022.

We demonstrate how to use our framework in practice, by applying it to two pairs of syntactic probing tasks. Each pair is formed by tasks that are structurally equal but for which we can reasonably assume a natural order in which the relevant linguistic representations emerge within the models. The first pair compares predicting part-of-speech (POS) tags that are the most frequent for a word form (MFTs) to predicting tags that are not (non-MFTs). Recent work has hypothesized that non-MFTs may be best represented in the deeper layers of a model such as BERT (Devlin et al., 2019): Each layer's contribution beyond the information already present in the uncontextualized layer is more significant for deeper layers, and therefore POS information could be found later in the model than previously assumed (Hewitt et al., 2021). The second pair of tasks compares predicting the position of a word's dependency head (the syntactic parent) to predicting the position of the head's head (the grandparent). Information for predicting grandparents has in previous work been found in deeper layers than information for predicting parents (Blevins et al., 2018). For each pair of tasks, we test where the relevant information emerges in the model, how the resulting pattern compares to global metrics, and if the metrics reflect the expected order of tasks. Our results show that while the expected hierarchy holds for the parent vs. grandparent task, information for non-MFTs emerges earlier in the model than for MFTs. This contradicts previous expectations. Also, results on seven independent monolingual BERT models in different languages show that the orderings and patterns we observe are not robust, which raises questions about their generality, and about the validity of probing at large.

2 Related Work

How linguistic information is distributed across the layers of a neural model is one of the central questions in the probing literature. The consensus is that there is a hierarchical ordering of tasks. Blevins et al. (2018) find a soft hierarchy of tasks when probing different layers of recurrent neural networks, from POS information being low in the hierarchy, to syntactic parents, grandparents, and great-grandparents. For their ELMo model, Peters et al. (2018) find that parts-of-speech are better predicted from the first hidden layer and word senses from the second. Tenney et al. (2019) probe BERT for a range of different NLP tasks and find that the layers that are the most predictive for each task are ordered like a classical pipeline: from parts-of-speech over syntactic dependencies, named entities and semantic roles to coreference.

How probing experiments should be designed and evaluated is subject to ongoing discussion. Some authors argue for simple classifiers (Alain and Bengio, 2017; Hewitt and Liang, 2019) to prevent the probes from learning the task and memorizing associations by themselves, while others make the case for more expressive models (Pimentel et al., 2020). While probes are most commonly evaluated using accuracy, recent work has proposed the use of alternative metrics that measure the effort of learning (Voita and Titov, 2020) or emphasize the performance early in the training (Talmor et al., 2020). Kunz and Kuhlmann (2021) propose to probe in an extrapolation setting, evaluating, among other setups, on the non-MFTs in diagnostic POS tagging experiments.

3 A Taxonomy of Metrics

We start by categorizing methods along three dimensions: The first one (as proposed by Hewitt et al. (2021)) concerns the relation of the baseline and the representation: how much more information can we extract from the representation than from the baseline (*baselined* probing), or how much information is extractable from the representation that does not overlap with information from the baseline (*conditional* probing).

The second dimension, proposed by us, concerns the type of information intended to be measured: information relative to a non-contextualized baseline (a *global* baseline), or information gain relative to the previous layer (a *local* baseline). The local setting challenges the view that a linguistic property's place in the model is the layer where most usable information for it can be extracted. Instead, we consider the layers where most usable information is *gained* relative to the previous layer to reflect the linguistic property's place within the model's hierarchy. We formulate and test the local correspondents of baselined and conditional probing in Sections 3.4 and 3.5.

Thirdly, we modify the local metrics so that they, in addition to the *absolute* reporting of the results, also support the reporting of the *relative* share that each layer contributes to the final performance. While absolute numbers convey more

information, relative numbers improve the comparability of results across tasks. We modify our local conditional metric from an absolute to a relative metric in Section 3.6.

3.1 General Setup

We consider a standard setup where we train a probe on a diagnostic task and evaluate it in terms of accuracy. More specifically, we use datasets $\mathcal{D} = \{(x_n, y_n)\}_n$ where each x_n is the representation of a neural language model at some specific layer, and y_n is the gold-standard label. (In our experiments, we use BERT.) By computing probe accuracy for different layers of the same model, we can compare layers in terms of how predictive they are with respect to the diagnostic task.

3.2 Global Baselined Probing (GBP)

In this common setup we measure the difference between the probe accuracy on a given layer l_i and the baseline layer l_0 – in BERT, this is the uncontextualized embedding layer. Thus we compute

$$\text{GBP}_i = \text{Acc}(l_i) - \text{Acc}(l_0) \tag{1}$$

As Hewitt et al. (2021) show, this can be interpreted as a difference between two quantities of $\mathcal{V}$*-information* (Xu et al., 2020), a theory of usable information under computational constraints. More specifically, GBP_i estimates the difference in $\mathcal{V}$-information between predicting the linguistic property under consideration from l_i and predicting it from layer l_0. This makes the difference in the probe's performance relative to the baseline explicit. The baselined information measures the amount of information gained over the baseline without making assumptions about the structural relation between l_0 and l_i.

3.3 Global Conditional Probing (GCP)

This setup has been proposed by Hewitt et al. (2021) with the intent to explicitly measure what information a layer l_i contributes *beyond* the information present in the baseline l_0. Practically, it entails computing the difference between the probe accuracy on the concatenation of l_i to l_0 and the baseline layer:

$$\text{GCP}_i = \text{Acc}([l_i; l_0]) - \text{Acc}(l_0) \tag{2}$$

In the framework of Hewitt et al. (2021), this measure is related to a conditional version of $\mathcal{V}$-information. More specifically, it estimates the conditional $\mathcal{V}$-information conditioned on prior information contained in the baseline.

3.4 Local Baselined Probing (LBP)

Analogously to global baselined probing, we may consider a local setup where the baseline is the previous layer l_{i-1}:

$$\text{LBP}_i = \text{Acc}(l_i) - \text{Acc}(l_{i-1}) \tag{3}$$

This quantity provides an estimate of how much $\mathcal{V}$-information is gained when taking the step from l_{i-1} to l_i. We posit that layers with high LBP values can be considered as layers where useful new information *emerges*. Intuitively, LBP measures the steepness of the slope, or the "jumps", in traditional accuracy curves across layers.

3.5 Local Conditional Probing (LCP)

To complete the picture, we propose to apply conditional probing to the local setting:

$$\text{LCP}_i = \text{Acc}([l_i; l_{i-1}]) - \text{Acc}(l_{i-1}) \tag{4}$$

The intention behind this metric is also to measure information gain with respect to l_{i-1}, but we account for exclusive information of l_{i-1} that is absent in l_i. Similar to Hewitt et al. (2021), we concatenate two layers and compare to scores on one of them. Our approach differs in that we do not compare to one static baseline layer (l_0) but dynamically to l_{i-1} to track the information gained across layers.

3.6 Emergent Information (EMI)

EMI (as well as EMI-BL in the next section) is designed to make layer contributions comparable across tasks that have different overall performances. To represent relative information gains, we calculate the LCP metric and divide it by the LCP summed up over all L layers. As we focus on *gains*, the metric will be zero whenever the result is negative (as nothing is gained). For the sum we also only consider layers where the LCP is positive.

$$\text{LCP}'_i = \max(0, \text{LCP}_i) \tag{5}$$

$$\text{EMI}_i = \frac{\text{LCP}'_i}{\sum_{k=1}^{L} \text{LCP}'_k} \tag{6}$$

We get relative gains that sum up to one. We interpret the results as the layer's contribution to the overall emergent information within the model.

3.7 EMI, Baselined Control (EMI-BL)

For ablation purposes we want to investigate the effect of the conditioning in EMI. Therefore we also employ a simplified version of EMI that uses LBP instead of LCP:

$$\text{LBP}'_i = \max(0, \text{LBP}_i) \tag{7}$$

$$\text{EMI-BL}_i = \frac{\text{LBP}'_i}{\sum_{k=1}^{L} \text{LBP}'_k} \tag{8}$$

The difference to EMI in Section 3.6 is the lack of control for information that was already present in the previous layer. We may underestimate the information gain as information may have been "forgotten" and replaced by new information when transitioning to the next layer.

4 Experiments

In our experiments, we apply the metrics defined in the previous section to study the performance of language models on two suitable probing tasks.

4.1 Probing Classifier

As our probe, we use a simple feed-forward network with 64 hidden units and ReLU activation, and train it for 10 epochs using the Adam optimizer (Kingma and Ba, 2015) with a learning rate of 0.001. Our implementation uses PyTorch (Paszke et al., 2019). We calculate the results for all metrics based on the mean accuracy over 10 random seeds.

4.2 Language Representation Models

To test the robustness of our probing results and to explore if the syntactic information is localized in similar regions across models and languages, we include seven models in our analysis. Apart from English BERT (Devlin et al., 2019), we train probes on monolingual BERT models in Czech (Sido et al., 2021), Finnish (Virtanen et al., 2019), German (Chan et al., 2020), Hebrew (Seker et al., 2021), Swedish (Malmsten et al., 2020) and Turkish (Schweter, 2020). The languages are chosen to represent diverse families: Indo-European/Germanic (*de*, *en*, *sv*), Indo-European/Slavic (*cs*), Uralic (*fi*), Turkic (*tr*), and Afro-Asiatic/Semitic (*he*). All models are *base* models with 12 layers, and accessed via the Huggingface Transformers library (Wolf et al., 2020).[1]

[1]All code necessary to reproduce the results in this paper, along with full numerical results, is available here: https://github.com/jekunz/emergent_info

4.3 Data and Tasks

We consider two pairs of closely related syntactic probing tasks. The training data for these tasks is derived from 1,000 sentences randomly sampled from the Universal Dependencies treebank (Zeman et al., 2021).[2]

POS tagging We predict UPOS tags and evaluate on two sets, the most frequent tags for a word form (MFT) and tags that are not the most frequent for a word form (non-MFT). We assume that:

Hypothesis 1 *Models learn to predict non-MFTs in deeper layers than MFTs.*

Syntactic Ancestors Prediction We predict the relative linear position of a token's head (parent, P) and its head's head (grandparent, GP) in the syntactic dependency tree. For practical reasons we omit examples where the distance is larger than 15. For this task, our assumption is:

Hypothesis 2 *Models learn to predict grandparents in deeper layers than parents.*

Classically, we would also assume that the ancestors tasks come higher in the hierarchy than the part-of-speech tagging tasks, which would give us the following hierarchy of all tasks:

$$\text{MFT} < \text{non-MFT} < \text{P} < \text{GP}$$

However, as only the tasks in each pair are structurally equal, we will analyze each pair of tasks separately.

4.4 Ranking

To determine the hierarchical ordering of tasks within the models, we need to reduce the metrics across layers to a single comparable value. For that, we employ two strategies:

Max Layer For all metrics, we report the layer which maximizes the respective metric. When this layer is deeper for a task T than for a task T', we say that T is *higher* in the hierarchy induced by the model than T'.

Early Contributions For the EMI metric, we also report the contribution of layers 1, $1+2$ and $1+2+3$ to the overall gain. When this contribution is higher for a task T than for a task T', we say that T is *lower* in the hierarchy than T'.

	GBP (& Accuracy)		GCP		LBP & EMI-BL		LCP & EMI	
	MFT	¬MFT	MFT	¬MFT	MFT	¬MFT	MFT	¬MFT
cs	4	**6**	4	**6**	**3**	2	**3**	2
de	6	**11**	8	**10**	**4**	1	**4**	1
en	4	**7**	**10**	8	**2**	1	1	1
fi	3	**4**	**7**	5	1	1	1	1
he	3	**5**	**8**	7	**2**	1	**2**	1
sv	3	**5**	**11**	6	**2**	1	**2**	1
tr	2	**5**	3	**11**	2	2	2	2
avg	3.6	**6.1**	7.3	**7.6**	**2.3**	1.3	**2.1**	1.3

Table 1: Part-of-speech tagging tasks. The numbers give the layer of maximum score across metrics and languages. Bold marks the task (MFT or ¬MFT) that is higher in the hierarchy induced by the model.

	GBP (& Accuracy)		GCP		LBP & EMI-BL		LCP & EMI	
	P	GP	P	GP	P	GP	P	GP
cs	5	**8**	5	**8**	1	**2**	1	**2**
de	9	9	9	9	2	2	2	2
en	5	**6**	5	**7**	1	1	1	1
fi	5	5	5	5	2	**3**	2	**3**
he	5	5	**9**	5	**4**	3	**4**	2
sv	**7**	6	7	7	**2**	1	**2**	1
tr	**7**	8	7	**8**	3	3	3	3
avg	6.1	**6.7**	6.7	**7.0**	2.1	2.1	**2.1**	2.0

Table 2: Syntactic ancestors prediction tasks. The numbers give the layer of maximum score across metrics and languages. Bold marks the task (P or GP) that is higher in the hierarchy induced by the model.

5 Results

This section presents the results of our experiments. We have structured our presentation around the two ranking methods.

5.1 Max Layer

For each probing setup and language, we report that layer which maximizes the respective metric in Table 1 for the POS tagging task pair and Table 2 for the ancestors prediction tasks.

Global metrics Our results for the global metrics confirm the finding of Hewitt et al. (2021) that the layers that maximize conditional probing accuracy (GCP) are generally deeper than those that maximize baselined accuracy (GBP).

Zooming in on the distinction between most frequent and non-most frequent tags for the POS tasks, however, exhibits an unexpected behavior: Hewitt et al. (2021) suggest that for the non-MFTs, GCP should be higher than GBP in deeper layers, and the other way round for MFTs. Here we find that in 4 out of 7 models, the layer with the highest GCP value on non-MFTs *precedes* the layer with the highest value for MFTs. However, the average over the models is higher for non-MFTs due to the large margin between the layers in the *tr* model. The highest scores of GBP on non-MFTs are consistently in deeper layers than those for MFTs. The exact layer in which the maximum scores are however varies greatly between models: for the MFTs, it ranges between 2 (*tr*) and 6 (*de*) and for the non-MFTs between 4 (*he*) and 11 (*de*).

GCP differentiates less than GBP, with a margin of 2.5 versus 0.3 (POS tagging) and 0.6 versus 0.3 layers (ancestors prediction) difference between the lower-level and the higher-level task.

Looking at the full plot, rather than just the maximal layer, we observe some variety across metrics and languages. The example plots for *en* BERT in Figure 2 (a–b) are in line with the general trend: GCP peaks in deeper layers than GBP, but this is not explained by the non-MFTs, as their curve drops steeper with increasing layer index than the curve for the MFTs. This observation holds for most BERT models we used, except for *cs* and *tr* where the scores on MFTs drop more in deeper layers than those for non-MFTs (see Figure 3).

[2]The treebanks for each language are: *cs*: PDT, *de*: GSD, *en*: EWT, *fi*: TDT, *he*: HTB, *sv*: Talbanken, *tr*: Kenet. Lic: CC BY-SA 4.0 (*de*, *en*, *fi*, *sv*, *tr*) / CC-BY-NC-SA 3.0 (*cs*, *he*).

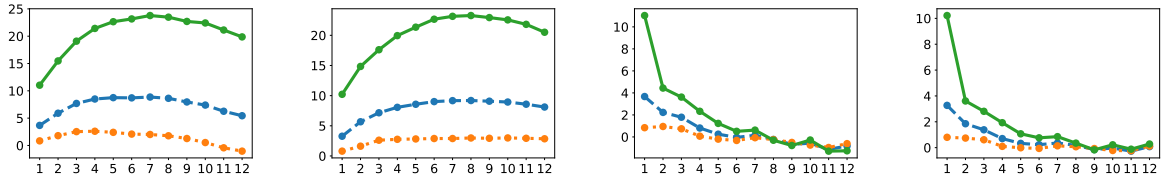

(a) GBP (b) GCP (c) LBP (d) LCP

Figure 2: Part-of-speech tagging, global (a–b) and local (c–d) metrics on the English data. Solid green line: non-MFTs, dotted orange: MFTs, dashed blue: full development set (all tags).

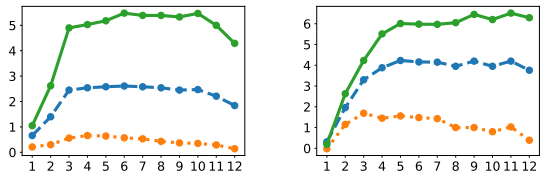

(a) GCP Czech (b) GCP Turkish

Figure 3: As opposed to *en* BERT and the other four models, for *cs* and *tr*, the scores on MFTs in GCP drop more over the layers than those for non-MFTs.

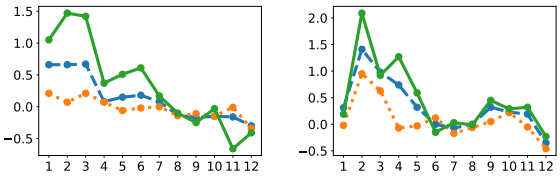

(a) LCP Czech (b) LCP Turkish

Figure 4: For *cs* and *tr* BERT, the LCP plots exhibit a pattern that deviates from that we observe for *en* BERT: They do not decrease steadily.

Local metrics Turning to the highest local information gain in Table 1 and 2, both LBP and LCP show the biggest gains in the very first layers. This shows that, while global metrics locate the overall information peaks somewhere in the middle of the model, only little new information actually emerges at that point.

The differences in the empirical results between LBP and LCP are small; specifically accounting for information that is absent in the previous layer does not result in a different pattern than the one we obtain when using the baselines metric. This leaves the choice between the two metrics to theoretical or practical preferences.

The example curves for English BERT in Figure 2 (c–d) show a typical pattern for the drop across layers in the part-of-speech tagging tasks. The layer of highest information gain appears to be the layer where contextual information is added first. After this layer, the plots decrease more slowly. Most languages follow this pattern, with the notable exception of *cs* and *tr* that do not show a steady decrease but go up first (see Figure 4). For the ancestors tasks, the peak is often shifted to the second or third layer, probably reflecting the higher-level nature of those tasks compared to the POS tasks. Figure 5 shows two examples, *fi* BERT with and *tr* BERT without a clear hierarchy of the parents versus grandparents task.

We observe that for the local metrics, the supposedly higher-level tasks do *not* have their highest gains in later layers than the lower-level tasks. On the contrary, the non-MFTs in LBP and LCP have their average max layer at 1.3, while for MFTs, where the accuracy starts off much higher, and information gains are generally smaller, the corresponding values are 2.3 and 2.1. For the ancestors tasks, there is on average no difference between the tasks. Hence, Hypotheses 1 and 2 are not confirmed for the local metrics in the max layer rankings.

5.2 Early Contributions

As Hypotheses 1 and 2 about the order of tasks in the model's hierarchy were not confirmed when looking at the layer with the maximal score, we compare more expressive metrics from the emergent information family in Tables 1 and 2.

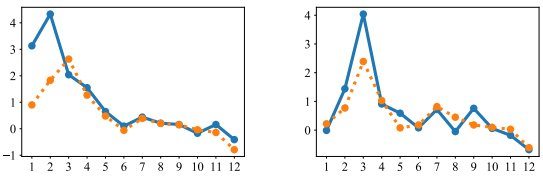

(a) LCP Finnish (b) LCP Turkish

Figure 5: Ancestors prediction, LCP: *fi* shows a later peak for grandparents (orange), while *tr* BERT's curves show a similar pattern for both tasks.

	Layer 1		Layer 1 + 2		Layer 1 + 2 + 3	
	MFT	¬MFT	MFT	¬MFT	MFT	¬MFT
cs	37.49	**18.74**	49.99	**44.99**	87.49	**70.35**
de	**0.00**	34.18	**23.85**	62.32	**46.78**	74.94
en	**31.00**	46.24	**59.68**	62.57	83.33	**75.29**
fi	**44.07**	55.28	**58.12**	71.87	**76.30**	84.31
he	39.31	**37.93**	89.74	**74.94**	97.15	**89.69**
sv	**12.86**	52.23	**48.53**	78.70	**58.47**	89.00
tr	**0.00**	3.08	48.22	**37.07**	80.20	**52.03**
avg	**23.53**	35.38	**54.01**	61.78	**75.67**	76.51

Table 3: Part-of-speech tagging tasks: Contribution of layer 1, $1 + 2$ and $1 + 2 + 3$ to the overall performance of the probe. Bold marks the task (MFT or ¬MFT) that is is higher in the hierarchy induced by the model (smaller contribution of the lower layers).

	Layer 1		Layer 1 + 2		Layer 1 + 2 + 3	
	P	GP	P	GP	P	GP
cs	46.78	**14.90**	73.39	**68.02**	**79.10**	81.30
de	8.68	**3.94**	50.06	**35.49**	68.68	**51.83**
en	37.47	**36.86**	55.15	**52.79**	78.89	**56.81**
fi	24.51	**11.46**	58.41	**34.77**	74.39	**68.28**
he	0.00	0.00	**16.90**	25.74	**27.46**	51.48
sv	**21.14**	39.31	60.33	**49.60**	77.65	**66.61**
tr	**0.00**	3.52	16.74	**15.84**	63.72	**54.08**
avg	19.79	**15.71**	47.28	**40.32**	67.12	**61.48**

Table 4: Syntactic ancestors prediction tasks: Contribution of layer 1, $1 + 2$ and $1 + 2 + 3$ to the overall performance of the probe. Bold marks the task (P or GP) that is is higher in the hierarchy induced by the model (smaller contribution of the lower layers).

For the MFT tasks, we can confirm the finding that when looking at the share of emergent information from the very first layers, it is on average higher for the non-MFTs for all three groupings of layers, indicating that non-MFTs would be lower in the hierarchy that MFTs. The difference however fades out, from 11.85 percentage points difference for layer 1 to only 0.84 points for layer $1 + 2 + 3$. For the ancestors tasks, the hierarchy continues to be as expected both on average and in the vast majority of model-grouping combinations. For both tasks we note that no layer grouping shows consistent results across all seven models, indicating a low robustness of the results.

Effects of Conditioning The average difference between emergent information with or without conditioning on the previous layer is 5.58 (POS) and 3.93 percentage points, which can in some scenarios be considered as minor. For all values we examined in this section, the average difference between tasks is 16.93 (POS) and 11.14 (ancestors) points, making a trend change unlikely and a simplification towards a baselined setup justifiable. Less distinct tasks and their theoretical advantages may however suggest the inclusion of the conditional accuracy.

6 Discussion

Where is a feature located? The results for the different metrics show how it depends on the perspective which place within the model we assign to a linguistic property. While the most overall information is located in the middle layers, it is the early layers that maximize the local metrics by a huge margin, meaning that this is the place in the model where most information either emerges or becomes accessible.

The complementary use of both families of metrics, local and global, gives us a more holistic picture of how information is structured within the model, as we argue that it it is not clear if a hierarchy of tasks within the model should be determined by where most information is added or where most information is accessible overall. An intuition supporting the former is the comparison to a human-made pipeline model, where tasks would naturally be placed where the information required for them is added – for instance, where the POS tagger is located, adding POS tags to the set of features. In these pipelines, there is no notion of how long information will be passed to higher-level tasks.

Expected hierarchies do not generally hold We see that when probing for MFTs versus non-MFTs, the perceived natural hierarchy of tasks (Hypothesis 1) does not hold for the local metrics, neither in a coarse max layer analysis nor in the more fine-grained early contribution setting, as shown in Sections 5.1 and 5.2. In the plots of the global metrics in Section 5.1 we see that non-MFTs often show both steeper gains in the beginning and more pronounced losses in the later layers, indicating that it is more specialized contextual information that the non-MFTs require, but that information does not appear to emerge later than in the model that for MFTs. We conclude that observations of a clear hierarchy of tasks depend on the focus on most usable overall information: They are already weaker in the global conditional setup, and are in one of two cases contradicted by local metrics.

The massive dominance of layer 1 and 2 in all local metrics especially for the non-MFTs but even for the parents and the grandparents tasks raises questions about how high-level the information that contributes the most to the overall performance on the probing task actually is. As the very first possibility of accessing contextual information already presents the heaviest boost, the features that are most crucial to solve the task appear to be surprisingly shallow.

Probing results are not robust An interesting point we noted across all metrics, but more distinctly for all local metrics as well as global conditional probing, is that the results are not stable across BERT models in different languages. We do not consider the possibility to relate the different distribution of information across models to linguistic properties of the languages as we believe that this is impossible with the relatively small set of non-parallel models we analyze. Apart from the language, they differ in several variables: most importantly, the data they are trained on, but some also in training details. However, we see it as an exciting path for future research to explore what causes a model to structure information in certain ways, and if this has implication on the model's performance on downstream tasks or robustness.

Relevance As the differences between measuring emergent information with and without conditioning on the previous layer are relatively small, one could suggest that the information is present in a similar form in ordinary accuracy plots across lay-

ers: The slope of the curve can be used to estimate it. But in practice, such interpretations do not appear to be obvious: Even though they can, as we showed, shed a different light at probing results when made explicit, previous work exclusively focused on the point of highest overall information. A discussion on the relevance of the parts of the network where information emerges has been absent from the literature.

Apart from that, we argue that conditional probing has strong theoretical advantages, as it explicitly accounts for information in the baseline representation, and that this makes basing the emergent information metrics on it favorable.

Limitations The metrics we propose are designed with the expectation that gains are successive. However, Transformer models can propagate information via residual connections and thereby let information "skip" layers. If this resulted in pronounced oscillations of the information within the model, it would weaken the meaningfulness of the results of all local metrics.

The emergent information metrics have no account for loss of information over the layers of the network. A related family of metrics that explicitly models this would be the application of the local metrics to the layers in inverse order.

All metrics in this paper are based on probe accuracy. However, our setups can be easily adapted to other metrics which have been shown to be more robust towards design choices regarding the classifier, such as minimum description length (Voita and Titov, 2020), or metrics that reward fast learning (Yogatama et al., 2019; Talmor et al., 2020).

7 Conclusion

We have collected and suggested metrics that model the information distribution in a model's layers from different perspectives: globally and locally, with or without conditioning on the baseline, and looking at absolute and relative gains of information. We used them on two pairs of probing tasks. First, we tested whether information for POS tags that are not the most frequent for a word is found in deeper layers than general POS information and found that while this is the case for overall information measured by global metrics, local metrics highlight that the most significant gains consistently happen in the very first layers *in particular* for the non-most frequent tags. For second task of predicting the syntactic parents versus grandparents of a token, however, the expected hierarchy in the model holds in the local setup at least in more fine-grained relative metrics. These mixed results emphasize the additional insights that zooming in to local information gains can give us into the model, the task, and the probing methodology.

Probing experiments on seven monolingual BERT models in different languages show that the metrics' behavior varies between models. While it is currently not feasible to relate the differences to specific properties of the models such as the language or the domain of the training data, a controlled training of parallel models where the additional variables are controlled for may enable such a comparison and is an insightful direction for future work.

References

Guillaume Alain and Yoshua Bengio. 2017. Understanding intermediate layers using linear classifier probes. In *5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Workshop Track Proceedings*. OpenReview.net.

Yonatan Belinkov. 2022. Probing classifiers: Promises, shortcomings, and advances. *Computational Linguistics*, 48(1):207–219.

Terra Blevins, Omer Levy, and Luke Zettlemoyer. 2018. Deep RNNs encode soft hierarchical syntax. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 14–19, Melbourne, Australia. Association for Computational Linguistics.

Branden Chan, Stefan Schweter, and Timo Möller. 2020. German's next language model. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 6788–6796, Barcelona, Spain (Online). International Committee on Computational Linguistics.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, Minneapolis, Minnesota. Association for Computational Linguistics.

John Hewitt, Kawin Ethayarajh, Percy Liang, and Christopher Manning. 2021. Conditional probing: measuring usable information beyond a baseline. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 1626–1639, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

John Hewitt and Percy Liang. 2019. Designing and interpreting probes with control tasks. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 2733–2743, Hong Kong, China. Association for Computational Linguistics.

Dieuwke Hupkes, Sara Veldhoen, and Willem Zuidema. 2018. Visualisation and'diagnostic classifiers' reveal how recurrent and recursive neural networks process hierarchical structure. *Journal of Artificial Intelligence Research*, 61:907–926.

Diederik P. Kingma and Jimmy Ba. 2015. Adam: A method for stochastic optimization. In *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*.

Jenny Kunz and Marco Kuhlmann. 2021. Test harder than you train: Probing with extrapolation splits. In *Proceedings of the Fourth BlackboxNLP Workshop on Analyzing and Interpreting Neural Networks for NLP*, pages 15–25, Punta Cana, Dominican Republic. Association for Computational Linguistics.

Martin Malmsten, Love Börjeson, and Chris Haffenden. 2020. Playing with words at the National Library of Sweden – Making a Swedish BERT.

Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Kopf, Edward Yang, Zachary DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. 2019. Pytorch: An imperative style, high-performance deep learning library. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett, editors, *Advances in Neural Information Processing Systems 32*, pages 8024–8035. Curran Associates, Inc.

Matthew E. Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer. 2018. Deep contextualized word representations. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 2227–2237, New Orleans, Louisiana. Association for Computational Linguistics.

Tiago Pimentel, Josef Valvoda, Rowan Hall Maudslay, Ran Zmigrod, Adina Williams, and Ryan Cotterell. 2020. Information-theoretic probing for linguistic structure. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 4609–4622, Online. Association for Computational Linguistics.

Stefan Schweter. 2020. BERTurk - BERT models for Turkish. *Zenodo*.

Amit Seker, Elron Bandel, Dan Bareket, Idan Brusilovsky, Refael Shaked Greenfeld, and Reut Tsarfaty. 2021. AlephBERT: A Hebrew large pre-trained language model to start-off your Hebrew NLP application with. *arXiv preprint arXiv:2104.04052*.

Jakub Sido, Ondřej Pražák, Pavel Přibáň, Jan Pašek, Michal Seják, and Miloslav Konopík. 2021. Czert – Czech BERT-like model for language representation. *arXiv preprint arXiv:2103.13031*.

Alon Talmor, Yanai Elazar, Yoav Goldberg, and Jonathan Berant. 2020. oLMpics-on what language model pre-training captures. *Transactions of the Association for Computational Linguistics*, 8:743–758.

Ian Tenney, Dipanjan Das, and Ellie Pavlick. 2019. BERT rediscovers the classical NLP pipeline. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 4593–4601, Florence, Italy. Association for Computational Linguistics.

Antti Virtanen, Jenna Kanerva, Rami Ilo, Jouni Luoma, Juhani Luotolahti, Tapio Salakoski, Filip Ginter, and Sampo Pyysalo. 2019. Multilingual is not enough: BERT for Finnish. *arXiv preprint arXiv:1912.07076*.

Elena Voita and Ivan Titov. 2020. Information-theoretic probing with minimum description length. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 183–196, Online. Association for Computational Linguistics.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Remi Louf, Morgan Funtowicz, Joe Davison, Sam Shleifer, Patrick von Platen, Clara Ma, Yacine Jernite, Julien Plu, Canwen Xu, Teven Le Scao, Sylvain Gugger, Mariama Drame, Quentin Lhoest, and Alexander Rush. 2020. Transformers: State-of-the-art natural language processing. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 38–45, Online. Association for Computational Linguistics.

Yilun Xu, Shengjia Zhao, Jiaming Song, Russell Stewart, and Stefano Ermon. 2020. A theory of usable information under computational constraints. In *8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, April 26-30, 2020*. OpenReview.net.

Dani Yogatama, Cyprien de Masson d'Autume, Jerome Connor, Tomas Kocisky, Mike Chrzanowski, Lingpeng Kong, Angeliki Lazaridou, Wang Ling, Lei Yu, Chris Dyer, et al. 2019. Learning and evaluating general linguistic intelligence. *arXiv preprint arXiv:1901.11373*.

Daniel Zeman, Joakim Nivre, Mitchell Abrams, Elia Ackermann, Noëmi Aepli, Hamid Aghaei, Željko Agić, Amir Ahmadi, Lars Ahrenberg, Chika Kennedy

Ajede, Gabrielė Aleksandravičiūtė, Ika Alfina, Lene Antonsen, Katya Aplonova, Angelina Aquino, Carolina Aragon, Maria Jesus Aranzabe, Bilge Nas Arıcan, Hórunn Arnardóttir, Gashaw Arutie, Jessica Naraiswari Arwidarasti, Masayuki Asahara, Deniz Baran Aslan, Luma Ateyah, Furkan Atmaca, Mohammed Attia, Aitziber Atutxa, Liesbeth Augustinus, Elena Badmaeva, Keerthana Balasubramani, Miguel Ballesteros, Esha Banerjee, Sebastian Bank, Verginica Barbu Mititelu, Starkaður Barkarson, Rodolfo Basile, Victoria Basmov, et al. 2021. Universal dependencies 2.9. LINDAT/CLARIAH-CZ digital library at the Institute of Formal and Applied Linguistics (ÚFAL), Faculty of Mathematics and Physics, Charles University.

A All results

For completeness, we present all plots across models and metrics as supplementary material in figure 6 for the POS tasks and in figure 7 for the ancestors tasks. A brief summary of the material is provided in the following paragraphs.

Global Metrics. The accuracy shows the same highs and lows as the GBP setup, where the static l_0 baseline is subtracted from the accuracy. Compared to GBP, the results in the GCP setup are slightly shifted to later layers. For the POS tasks, the peak is in the early middle layers, with the ¬MFTs peaking a few layers later, indicating the need for more contextual information. Across models we see a large variation, most extremely visible in *de*, where the scores increase until layer 11 for the MFTs, and tr, where the drop for the MFTs is more distinct than for other models. *fi* and *he* have a distinct peak for the ¬MFTs in layer 4, then a decrease, and then stabilize. The ancestors tasks often peak in the early middle layers as well, with *de* being shifted to notably later layers, and *tr* being relatively stable across all layers except the very first and last layers. A later peak of the grandparent prediction compared to the parent prediction is vaguely perceptible in most plots, most prominently in the *en* model.

Local Metrics. The metrics that measure the local information gain have the most consistent pattern for the ¬MFTs, with most information generally added in the very first layer. The pattern of the curves appears to asymptotically approximate 0. There are however two exceptions: the *cs*, but most distinctly the *tr* model that gains relatively little in the first layer and makes its biggest jump in the second layer. We also observe in the accuracy curve of these two models that the increase in the beginning is less steep. In the ancestors tasks, the highest layer is slightly later on average, often in later 2 or 3 (2.1 on average). In some models, such as *cs* and *fi*, we observe a later peak for grandparents than for parents, while for *de*, and *se*, it even is the other way round. This underlines the lacking robustness of our probing results across models in different languages that are particularly prominent for the local metrics. In all of the models we observe little difference in the empirical results and patterns of LBP and LCP, confirming our observations in Section 6 that the choice between them can be either arbitrary or based on theoretical preferences.

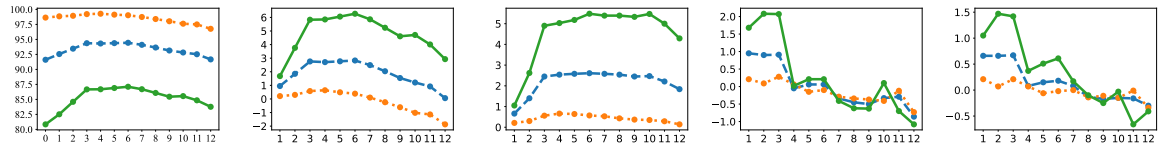

(1) Accuracy Czech (2) GBP Czech (3) GCP Czech (4) LBP Czech (5) LCP Czech

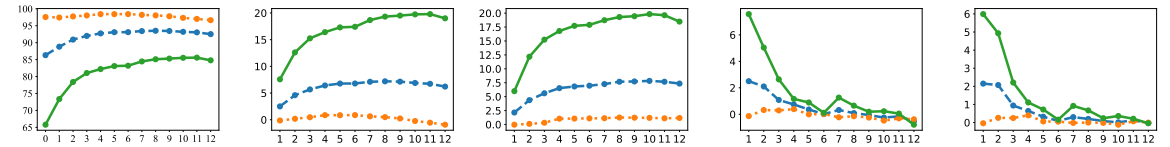

(6) Accuracy German (7) GBP German (8) GCP German (9) LBP German (10) LCP German

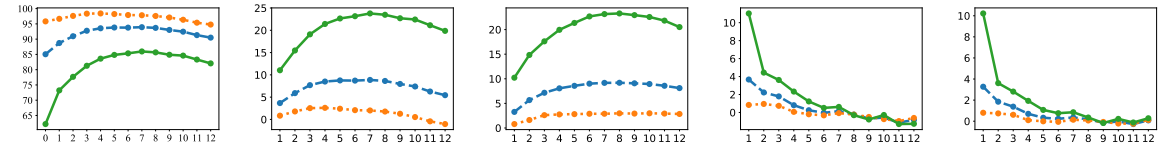

(11) Accuracy English (12) GBP English (13) GCP English (14) LBP English (15) LCP English

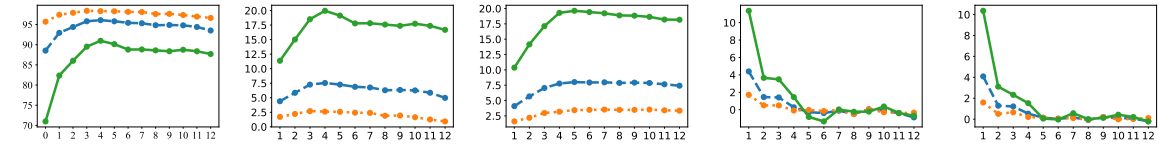

(16) Accuracy Finnish (17) GBP Finnish (18) GCP Finnish (19) LBP Finnish (20) LCP Finnish

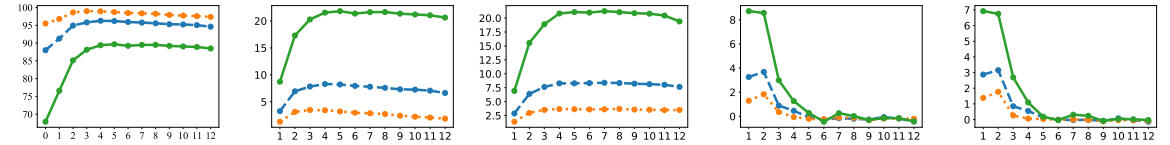

(21) Accuracy Hebrew (22) GBP Hebrew (23) GCP Hebrew (24) LBP Hebrew (25) LCP Hebrew

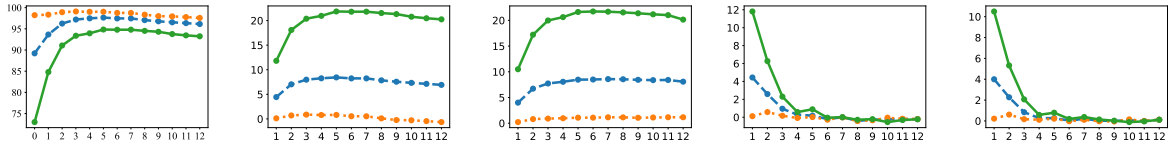

(26) Accuracy Swedish (27) GBP Swedish (28) GCP Swedish (29) LBP Swedish (30) LCP Swedish

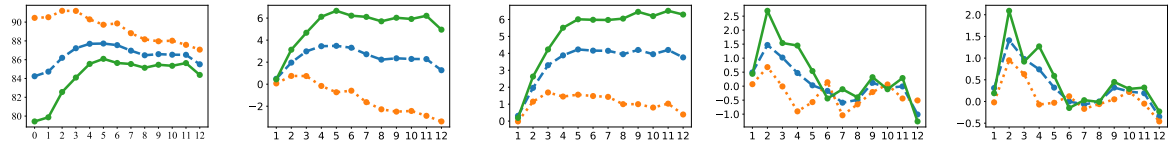

(31) Accuracy Turkish (32) GBP Turkish (33) GCP Turkish (34) LBP Turkish (35) LCP Turkish

Figure 6: POS Experiments: Plots for all language/metric combinations. Orange: MFT; green: ¬MFT; blue: all.

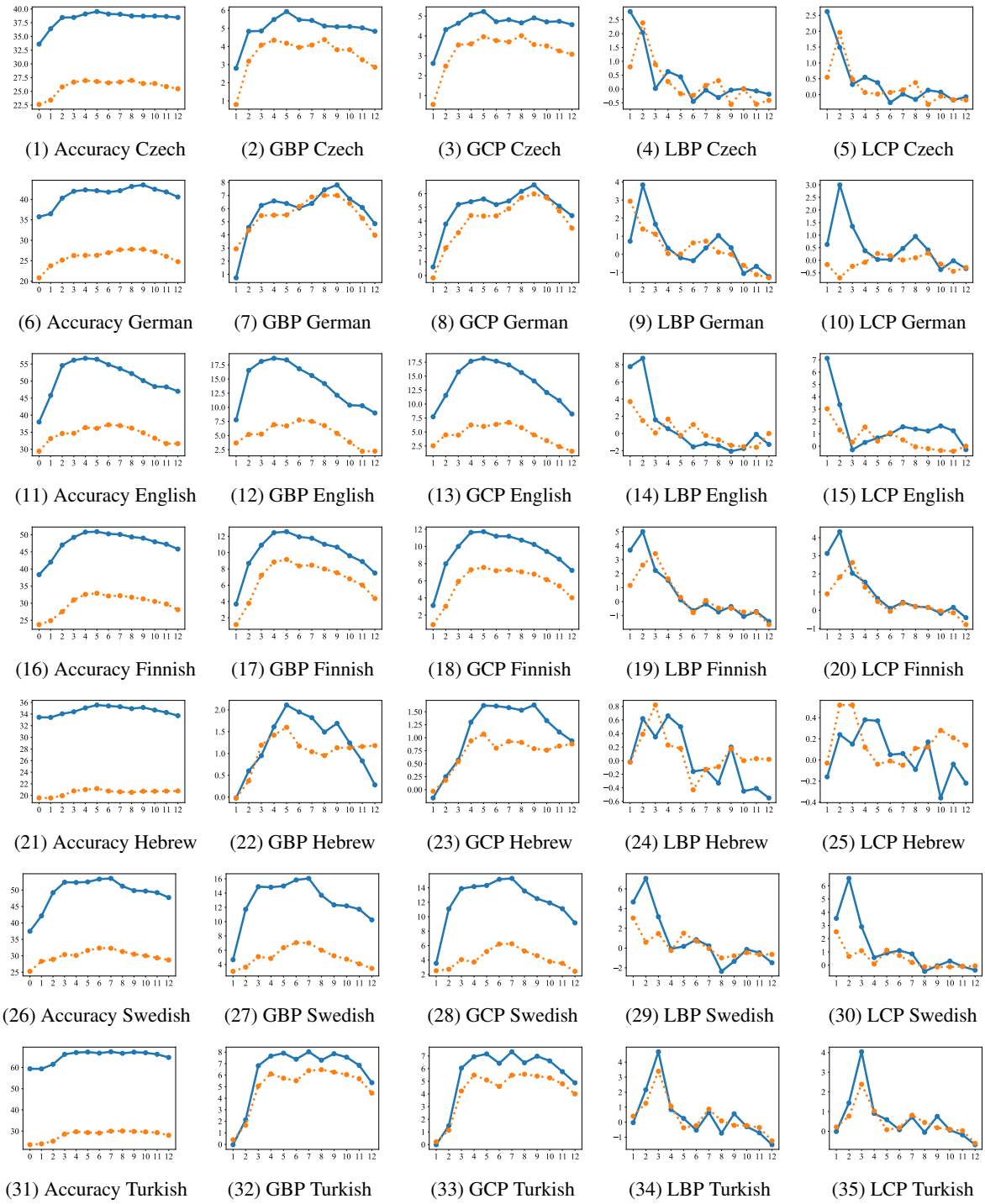

Figure 7: Ancestors Experiments: Plots for all language/metric combinations. Blue: P; orange: GP.

Accelerating Inference for Pretrained Language Models by Unified Multi-Perspective Early Exiting

Jun Kong [†], **Jin Wang**[†,*], **Liang-Chih Yu**[‡,*] and **Xuejie Zhang**[†]
[†]School of Information Science and Engineering, Yunnan University, Yunnan, P.R. China
[‡]Department of Information Management, Yuan Ze University, Taiwan
Contact:wangjin@ynu.edu.cn, lcyu@saturn.yzu.edu.tw

Abstract

Conditional computation algorithms, such as the early exiting (EE) algorithm, can be applied to accelerate the inference of pretrained language models while maintaining competitive performance on resource-constrained devices. However, this approach is only applied to the vertical architecture to decide which layers should be used for inference. Conversely, the operation of the horizontal perspective is ignored, and the determination of which tokens in each layer should participate in the computation fails, leading to a high redundancy for adaptive inference. To address this limitation, a unified horizontal and vertical multi-perspective early exiting (MPEE) framework is proposed in this study to accelerate the inference of transformer-based models. Specifically, the vertical architecture uses recycling EE classifier memory and weighted self-distillation to enhance the performance of the EE classifiers. Then, the horizontal perspective uses recycling class attention memory to emphasize the informative tokens. Conversely, the tokens with less information are truncated by weighted fusion and isolated from the following computation. Based on this, both horizontal and vertical EE are unified to obtain a better tradeoff between performance and efficiency. Extensive experimental results show that MPEE can achieve higher acceleration inference with competent performance than existing competitive methods. The code for this paper is available at: https://github.com/JunKong5/MPEE.

1 Introduction

Pretrained language models (PLMs) (Devlin et al., 2019; Lan et al., 2019; Liu et al., 2019) have shown promising performance in many natural language processing tasks. The success of PLMs incurs computational consumption and long inference latency, which prevents these models from being deployed on resource-constrained devices, such as edge devices, or in time-sensitive scenarios.

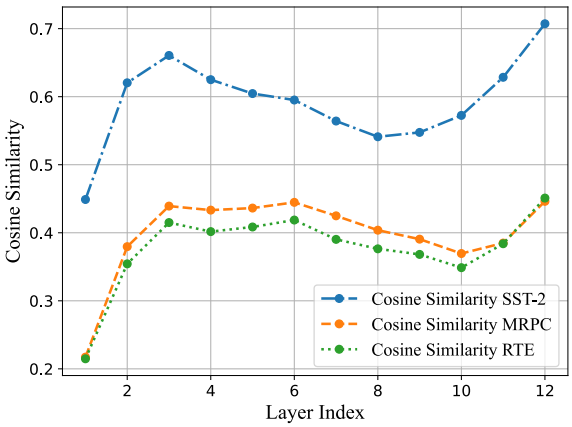

Figure 1: Tokens (cosine similarity) of different layers.

To address these issues, model compression techniques (Lin et al., 2020), including knowledge distillation (Hinton et al., 2015; Jiao et al., 2020), pruning (Sanh et al., 2020; Michel et al., 2019) and quantization (Zafrir et al., 2019), have been applied to PLMs to accelerate inference. However, these methods permanently remove some components of the model, leading to an inevitable decline in performance. Additionally, the complexity of these models cannot be adjusted based on the requirements of different hardware since the compression is implemented before the deployment. Conversely, several studies have suggested the use of conditional computation algorithms, such as the early exiting (EE) algorithm (Schwartz et al., 2020; Xin et al., 2020), in which each input sample uses a different part of the model so that the computation or latency is reduced on average. Based on this, an EE classifier is added between transformer layers. The shallow layer allows the *easy* samples to exit earlier without the need to perform computations up to the final layer. Computations related to deeper layers should only be reserved for *hard* samples.

There are two main limitations of existing EE

*Corresponding authors.

Proceedings of the 29th International Conference on Computational Linguistics, pages 4677–4686
October 12–17, 2022.

methods. For existing EE methods, there has been a focus on the exploration of accelerated inference for *easy* samples. However, for *hard* samples, acceleration inference is not achieved and still needs to be performed through the last layer. Furthermore, the computation consumption is quadratically proportional to the input sequence length. As a result, the average acceleration inference is limited.

In addition, the existing early exiting approach is only performed on the vertical perspective, and the operations of the horizon perspective are ignored. That is, each transformer encoder layer is regarded as a basic unit. The model determines how many transformer encoder layers are needed for inference but ignores which tokens in each layer are required to participate in the computation. Recent studies (Ethayarajh, 2019; Klafka and Ettinger, 2020) have shown that sequence tokens from the horizontal perspective have high redundancy. To further verify the redundant information from the horizontal perspective, we use the cosine distance to calculate the similarity between tokens in each layer, as shown in Figure 1. The token similarity in the shallow layer is low, indicating that the shallow layer is not fully encoded. The horizontal perspectives of deeper layers, especially the last layer, show high redundancy. This suggests that the previous EE approaches ignored the possibility of reducing the horizontal perspective redundancy to accelerate inference. Therefore, these EE approaches yield a suboptimal accelerated inference. To address this issue, an intuitive idea is to reduce the length of the sequence token as the layers grow. TR-BERT (Ye et al., 2021) uses a reinforcement learning (RL) method to select the tokens that need to be dynamically reduced. However, additional training is needed, which increases the cost of training. In addition, changes to the device platform require retraining of these components, which may greatly limit the application scenarios.

In this paper, we propose a unified horizontal and vertical multi-perspective early exiting framework to reduce the computation and latency for fast inference of transformer-based models. This framework contains layer-wise EE for the vertical perspective and sequential token-wise EE for the horizontal perspective. In sequential token-wise EE, different tokens are forced to exit at different layers to reduce computation by emphasizing the informative tokens to the downstream task. Conversely, the tokens with less information are truncated, which isolates them from the following computation. To measure the importance of tokens, we use class attention to learn the amount of prediction information. Due to the instability of the class attention in the shallow layer, recycling class attention memory is used to enhance the reuse of information across layers and to better identify informative tokens. The tokens with less information do not completely exit. Instead, they take a weighted fusion into an EE fusion token for subsequent calculations so that additional parameters and computations are not introduced.

For the vertical perspective EE, we introduce EE classifiers in the middle of two transformer layers. Samples exit early at the shallow layer when the confidence level is higher than the threshold and skip computation in other layers. This allows the input samples to be predicted with a shallow EE classifier rather than a deep EE classifier. Since the shallow EE classifier is weakly expressive, we introduce weighted ensemble self-distillation and recycling EE classifier memory to enhance the performance of the shallow EE classifier. Thus, a performance guarantee is provided at high acceleration inferences. The unified multi-perspective early exiting framework has horizontal sparsity and vertical sparsity. Based on extensive experiments, it is shown that the proposed model reduces computation and improves acceleration inference while maintaining high performance.

The rest of the paper is organized as follows. In Section 2, the preliminaries of PLMs are introduced. In Section 3, a detailed description of the proposed methods is provided. The empirical experiments are reported and analyzed in Section 4. Section 5 briefly introduces the previous studies on acceleration inference for PLMs. Conclusions are finally drawn in Section 6.

2 Preliminaries

Pretrained Language Model. A PLM consists of L transformer encoders (Vaswani et al., 2017). The sequence token $X = [x_{[\mathrm{CLS}]}, x_1, x_2, ..., x_N]$ of the text processed by the tokenizer is input into the PLM. N is the number of sequence tokens (without the `[CLS]` token), and the `[CLS]` token denote the global text information and final classification. The transformer contains multi-head self-attention (MHSA) and feed-forward network (FFN) modules. The transformer coding process of each layer is defined as

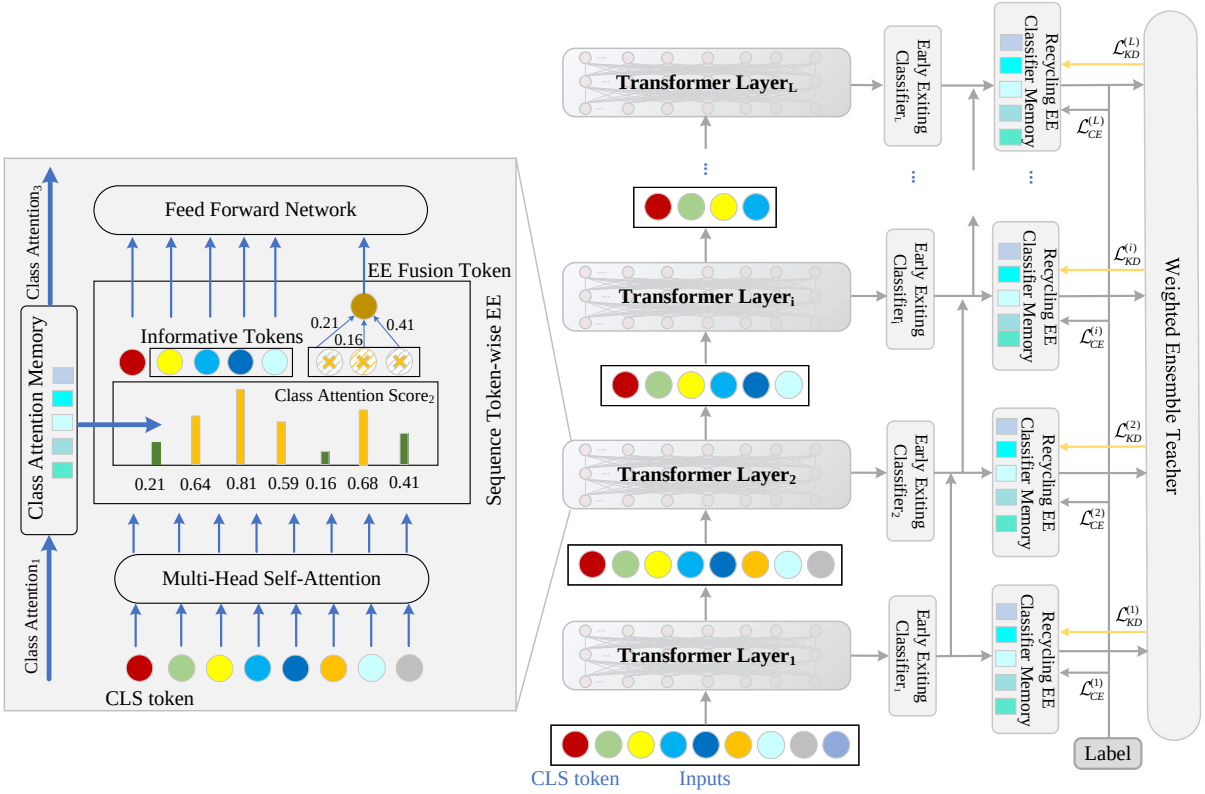

Figure 2: Overall architecture of the unified multi-perspective early exiting framework for accelerating.

$$Z^l = \text{Transformer}^l(h^{l-1}_{[\text{CLS}]}, h^{l-1}_1, ..., h^{l-1}_N) \quad (1)$$

where $Z_l \in \mathbb{R}^{n\times d}$ denotes the output feature of the l-th transformer, and N denotes the number of tokens in the l-th layer of MHSA and d denotes the model hidden size. The MHSA module is defined as

$$\begin{aligned}&\text{MHSA}(Z_{l-1})\\ &= \text{Concat}[\text{HA}_1(Z_{l-1}); ...; \text{HA}_H(Z_{l-1})]W^o\end{aligned} \quad (2)$$

where $W^o \in \mathbb{R}^{(H\cdot d_h)\times d}$ is the output projection, H is the number of multi-head and $d_h = d/H$. The single-head self-attention is denoted as

$$\begin{aligned}\text{HA}_h(Z_{l-1}) &= \text{Attention}(Q_h, K_h, V_h)\\ &= \text{softmax}(\frac{Q_h K_h^T}{\sqrt{d_h}})V_h\end{aligned} \quad (3)$$

where $Q_h = Z_{l-1}W_h^Q, K_h = Z_{l-1}W_h^K, V_h = Z_{l-1}W_h^V$ are the matrices that package the query, key and value, and $W_h^{\{Q,K,V\}} \in \mathbb{R}^{d\times d_h}$ is the linear transformation. Two fully connected layers are applied on the output of the MHSA, and the FFN module is defined as

$$\text{FFN}(\tilde{Z}_l) = \sigma(\tilde{Z}_l W_1 + b_1)W_2 + b_2 \quad (4)$$

where σ denotes the GELU activation function, $\tilde{Z}_l$ is the MHSA output,$W_1 \in \mathbb{R}^{d\times 4d}$ and $W_2 \in \mathbb{R}^{4d\times d}$ are the projection matrix.

Computational Complexity. Given an input sequence $N \times D$, N is the number of input tokens and D is the embedding dimension of each token. From the above, the computational complexity of MHSA is $O(4ND^2 + 2N^2D)$ and FNN is $O(8ND^2)$. The total computational complexity of PLMs is $O(12ND^2 + 2N^2D)$. The complexity of MHSA and FFN is quadratically and linearly for the length N of the input sequence. Furthermore, existing studies (Ethayarajh, 2019) have shown that sequence tokens have high redundancy. An intuitive idea is to reduce redundant tokens to reduce computation and improve acceleration inference.

3 Multi-Perspective Early Exiting

Figure 2 shows an overview of the proposed unified multi-perspective early exiting framework, which consists of layer-wise and sequential token-wise early exiting. Layer-wise and sequential token-wise early exiting are orthogonal methods that exit unnecessary computations early by the layer-wise and token-wise of the model, respectively, to achieve faster inference. Therefore, we

combine these two different perspectives to achieve accelerated inference and minimize computational cost.

3.1 Sequential Token-wise Early Exiting

The inference is accelerated by early exiting of the sequential token according to class attention, as shown in Figure 2. In PLMs, the `[CLS]` token is used for classification. According to Eq.(3), MHSA outputs of the `[CLS]` token, i.e., $h^l_{[\mathrm{CLS}]}$, can be regarded as a weighted summation of value vectors with attention scores, which is defined as

$$\begin{aligned} A^l_{class} &= \mathrm{MeanPooling}(\sum_{h}^{H} \mathrm{HA}_h(Z^0_l)) \\ h^l_{[\mathrm{CLS}]} &= V^l \cdot A^l_{class} \end{aligned} \tag{5}$$

where $\mathrm{HA}_h(Z^0_l)$ is the *h*-th attention head and A^l_{class} denotes the mean values of all attention heads corresponding to the `[CLS]` token. Here, A^l_{class} represents the interaction of the `[CLS]` token for all other tokens and determines how much information from the *i*-th token is contributed to the `[CLS]` token. It is intuitive to denote A^l_{class} as the information that the `[CLS]` token transferred for downstream tasks. Therefore, each token is assigned to be informative or less informative according to the class attention score A^l_{class}. Due to the instability and weak representation of the shallow layer class attention, recycling class attention memory is used to enhance the reuse of information across layers and to better identify informative tokens and is defined as

$$A^l_m = (1-\beta)A^{l-1}_m + \beta A^l_{class} \tag{6}$$

where A^l_m is a class attention memory used to record the historical values of A^l_{class}. β denotes the weight that balance the class attention with different layers. We keep *m* informative tokens h^l_{in} for *m*-highest score in A^l_m, which is formulated as

$$m = N(1 - R_{ee}) \tag{7}$$

where R_{ee} denotes the sequence EE ratio. For p early exiting tokens that are less informative, $p = N - m$. Less informative tokens do not completely exit early from the sequence. To reduce the missing information due to early exiting, the less informative tokens are fused into an EE fusion token according to the corresponding class attention weights. The EE fusion token and the informative tokens are spliced in the subsequent computation, which is denoted as

$$\begin{aligned} h^l_{eef} &= \alpha_i h^l_i \\ \tilde{Z}_l &= \mathrm{Concat}[h^l_{CLS}; h^l_{in}; h^l_{eef}] \end{aligned} \tag{8}$$

where h^l_i represents tokens with less informative and α_i denotes the corresponding class attention weight. [;] indicates a connection operation.

3.2 Layer-wise Early Exiting

In addition to sequential token-wise EE, layer-wise EE is used to further reduce the layers and computation of PLMs from another perspective, as shown in Figure 2. The EE classifier is added between every two transformer layers. The logit distribution C^l_z of the EE classifier is defined as

$$C^l_z = \tanh(W^l_z h^l_{[\mathrm{CLS}]} + b^l_z) \tag{9}$$

where $W^l_z \in \mathbb{R}^{C\times d}$ and $b^l_z \in \mathbb{R}^J$ represent the weight and bias of the *l*-th EE classifier, respectively, and *J* denotes the number of classes.

Each EE classifier in the existing EE approaches makes its decision independently, ignoring information from previous EE classifiers and discarding potentially valuable information. Recycling EE classifier memory is used to fuse useful information from previous EE classifiers into the current EE classifier and obtain useful information from different layers of EE classifiers to help make accurate and reliable predictions and is defined as

$$\begin{aligned} C^l_{zm} &= (1-\lambda)C^{l-1}_{zm} + \lambda C^l_z \\ \hat{y}^l_z &= \mathrm{softmax}(C^l_{zm}) \end{aligned} \tag{10}$$

where $\hat{y}^l_z$ denotes the probability distribution of the EE classifier and C^l_{zm} denotes EE classifier memory. The cross-entropy training objectives of these EE classifiers are defined as follows:

$$\mathcal{L}_{CE} = -\sum_{l=1}^{L} \mathbb{I}(y) \circ \log(\hat{y}^l_z) \tag{11}$$

where y denotes the corresponding ground-truth label, $\mathbb{I}(y)$ denotes a one-hot label with the y-th element being one, and $\circ$ represents an element-wise multiplication operation.

The weak expressiveness of the shallow EE classifier and the lack of higher-level semantic information lead to poor performance of the shallow EE classifier. Thus, weighted ensemble self-distillation is applied to further improve the representation abilities and obtain rich semantic information from the

EE classifiers. Based on the importance of students, i.e., EE classifiers, to the final prediction, an attention mechanism is used to combine all students into a teacher model and is defined as

$$\eta^l = \frac{\exp(V^T \tanh(W_\eta C^l_{zm} + b_\eta))}{\sum_{l=1}^{L} \exp(V^T \tanh(W_\eta C^l_{zm} + b_\eta))}$$
$$t = \sum_{l=1}^{L} \eta^l C^l_{zm} \quad (12)$$

where V is the context vector, W_α and b_α are the weight and bias, respectively, and t is the teacher for self-distillation, which is accomplished by minimizing the distribution between the teacher model and the student models and is defined as

$$\mathcal{L}_{KD} = \sum_{l=1}^{L} \tau^2 \mathrm{KL}(\mathrm{softmax}(C^l_{zm}/\tau) || \mathrm{softmax}(t/\tau)) \quad (13)$$

where $\mathrm{KL}(\cdot||\cdot)$ is the Kullback-Leibler divergence function and τ is the temperature parameter that measures the smoothness of the distribution. τ^2 compensates for the size of the gradient scaled by the soft target, ensuring that there is no negative impact on the gradient size. The overall loss of the proposed MPEE is denoted as

$$\mathcal{L} = \mathcal{L}_{CE} + \mathcal{L}_{KD} \quad (14)$$

In MPEE, the informative token is first selected and the less informative token exits. Furthermore, less informative tokens are fused into EE fusion tokens to continue the computation in the following layer. Then, the decision of whether to exit the whole computation at the current layer is based on whether the entropy value of the current EE classifier is greater than the preset threshold F.

4 Experiments

4.1 Datasets

To evaluate the acceleration effect of the proposed method on the inference of PLMs, we conducted experiments on the General Language Understanding Evaluation (GLUE) benchmark (Wang et al., 2019), which includes SST-2 (Socher et al., 2013) for sentiment analysis containing sentences from movie reviews. QQP and MRPC (Dolan and Brockett, 2005) for similarity and paraphrase. QNLI (Rajpurkar et al., 2016) and MNLI (Williams et al., 2018) for natural language inference, and the RTE for textual entailment.

4.2 Performance and Acceleration Metrics

For performance metrics, accuracy (Acc) and F_1-score (F_1) were used as evaluation metrics for MRPC. For all other tasks, accuracy was used as the evaluation metric.

For acceleration metrics, following Ye et al. (2021), we used the number of floating operations (FLOPs) as the acceleration inference ratio. Because FLOPs evaluate acceleration inference independently of the runtime environment and hardware, it is easy to compare the acceleration inference of the models. The acceleration ratio ρ is defined as the rate of the total FLOPs by the original model to the FLOPs actually executed by the model and is defined as

$$\rho = \frac{FLOPs_{exec}}{FLOPs_{total}} \quad (15)$$

where FLOPs are the sum of floating operations of all inference sample. The individual sample FLOPs are calculated during inference. The performance with different acceleration ratios is obtained by adjusting the threshold F and sequence EE ratio R_{ee}.

4.3 Baselines

To demonstrate the effectiveness of the proposed method, several previous methods are implemented for comparison for PLMs accelerated inference. The baselines include the BERT backbone model, model compression methods, early exiting methods, and sequence reduction methods. Other model compression methods, such as BERT-6L, DistilBERT (Sanh et al., 2019), BERT-PKD (Sun et al., 2019) and LayerDrop (Fan et al., 2019) are also applied. In addition, early exiting methods, including DeeBERT (Xin et al., 2020), FastBERT (Liu et al., 2020) and PABEE (Zhou et al., 2020) and sequence length reduction methods, including PoWER-BERT (Goyal et al., 2020) and TR-BERT (Ye et al., 2021) are introduced for comparison. The implementation and the optimal hyperparameters are fine-tune by using a grid search strategy to train the baseline. To make a fair comparison, we use the same parameters for the baseline and calculated PoWER-BERT in a single sample of FLOPs.

4.4 Implementation Details

The implementation of the proposed model is based on HuggingFace's transformers (Wolf et al., 2020). `BERT-base-uncased` is used as the backbone

Model	MNLI-m		SST-2		MRPC		QQP		MNLI-mm		QNLI		RTE	
	Acc	ρ	Acc	ρ	F_1/Acc	ρ	Acc	ρ	Acc	ρ	Acc	ρ	Acc	ρ
BERT-Base	83.9	1.00×	92.1	1.00×	90.3/86.3	1.00×	91.1	1.00×	83.8	1.00×	91.2	1.00×	71.1	1.00×
BERT-6L	80.3	2.00×	90.1	2.00×	86.6/80.3	2.00×	88.0	2.00×	80.6	2.00×	86.9	2.00×	64.5	2.00×
DistilBERT	79.0	2.00×	90.7	2.00×	87.5/-	2.00×	88.5	2.00×	81.5	2.00×	85.3	2.00×	59.9	2.00×
BERT-PKD	81.3	2.00×	91.3	2.00×	85.7/-	2.00×	88.4	2.00×	-	2.00×	88.4	2.00×	66.5	2.00×
LayerDrop	80.7	2.00×	90.7	2.00×	85.9/-	2.00×	88.3	2.00×	-	2.00×	88.4	2.00×	65.2	2.00×
DeeBERT	74.3	1.92×	90.4	1.97×	86.5/79.9	1.95×	88.2	1.96×	74.5	1.91×	86.4	1.95×	63.8	1.96×
FastBERT	74.5	1.93×	90.8	1.98×	86.6/80.3	1.96×	88.4	1.96×	74.8	1.92×	86.7	1.92×	65.0	1.98×
PABEE	79.0	1.95×	90.1	1.98×	86.8/80.4	1.80×	89.1	1.98×	79.5	1.93×	87.6	1.85×	63.9	1.96×
PoWER-BERT	81.9	2.47×	91.1	2.33×	88.4/82.8	3.11×	89.7	3.27×	81.6	2.52×	89.3	2.21×	68.2	2.56×
TR-BERT	82.1	2.96×	90.3	1.63×	84.9/75.9	1.23×	89.4	3.15×	81.9	3.02×	88.2	1.57×	67.8	2.63×
MPEE	**82.6**	**3.38×**	**91.7**	**4.4×**	**88.7/83.3**	**3.47×**	**90.0**	**4.33×**	**82.3**	**3.41×**	**90.2**	**2.45×**	**69.0**	**3.55×**

Table 1: Compare experimental results in the baseline methods with BERT backbone on GLUE.

of our model, where the transformer layer is 12 layers, the attention head is 12, and the hidden dimension is 768. The training batch sizes are 64 and 128. The inference batch size for the proposed model and baseline is 1. We use the grid search to find 0.7 for R_{ee} and 0.5 and 0.9 for β and λ, respectively. The model is optimized using Adam, and the learning rate is 2e-5.

4.5 Comparative Results

Table 1 shows the performance and acceleration inference of the proposed method and baseline methods. The proposed MPEE method outperforms all baseline methods in improving acceleration inference while maintaining better performance, verifying the effectiveness of the proposed model. The accuracy degradation is within a relatively small range compared to BERT, while the acceleration inference is significantly improved on most datasets. Especially, the acceleration ratio ρ is 4.4× on SST-2.

Further, the proposed MPEE method has 4.4× acceleration ratio ρ on SST-2, but still maintains a 91.6% accuracy. The proposed MPEE outperforms the existing EE methods due to the shallow layer learns higher layer semantic information, allowing it to improve its expressiveness, which leads to decreased model performance. Ignoring horizontal perspective redundancy limits accelerated inference. On SST-2, MPEE has a 1.4% higher performance than TR-BERT with respect to accuracy, but still has a 2.86× faster acceleration ratio ρ on inference. TR-BERT discards some token information and reduces the sequence length in only two layers to ease the convergence of the model with RL and reduce the search space. Therefore, its acceleration effect is relatively insignificant, thus its performance is degraded. PoWER-BERT completely removes the token, leading to partial information loss and decreasing model performance. This sequence reduction approach requires the computation to be executed through the last layer and ignores vertical perspective redundancy, resulting in limited acceleration. The proposed approach significantly reduces computation and accelerates inference by simultaneous early exits from both horizontal and vertical perspectives, and the EEs of the two perspectives are orthogonal. The EE token information is preserved, and the performance of the model is maintained by weighted self-distillation.

4.6 Ablation Study

We conducted several ablation studies to better demonstrate the effectiveness of the proposed modules, including recycling class attention memory (Att-memory), EE fusion token (EE-fusion), recycling EE classifier memory (EE-memory), weighted self-distillation (WSD), layer-wise EE (Layer-EE) and sequential token-wise EE (Token-EE). To demonstrate their effectiveness, we have removed each module individually to show that the performance is degraded, as observed in Table 2.

The removal of EE fusion leads to performance degradation because most of the semantic information is lost in this setup. Similarly, the absence of Att-memory leads to performance degradation, proving the ability of proposed model to effectively enhance the selected informative tokens by combining class attention at different layers. The removal of Token-EE and Layer-EE leads to a significant reduction in acceleration. The rational reason for these results is that vertical and horizontal perspec-

Model	SST-2		MRPC		RTE		QQP		QNLI	
	Acc	ρ	F_1/Acc	ρ	Acc	ρ	Acc	ρ	Acc	ρ
BERT	92.1	1.00×	90.3/86.3	1.00×	71.1	1.00×	91.1	1.00×	91.2	1.00×
MPEE	**91.7**	**4.40×**	**88.7/83.3**	**3.47×**	**69.0**	**3.55×**	**90.0**	**4.33×**	**90.2**	**2.45×**
w/o Att-memory	91.4	4.22×	88.3/82.9	3.26×	68.1	3.26×	89.5	4.12×	89.7	2.26×
w/o EE-memory	91.3	4.27×	88.2/83.1	3.18×	68.3	3.19×	89.3	4.26×	89.8	2.32×
w/o WSD	91.2	3.92×	87.8/83.0	2.96×	67.9	3.08×	89.1	4.02×	89.5	2.06×
w/o EE-fusion	91.0	4.18×	88.0/82.5	3.23×	68.4	3.28×	89.2	4.08×	89.6	2.37×
w/o Layer-EE	91.3	2.43×	86.8/81.1	2.62×	68.2	2.43×	87.3	2.43×	90.0	1.74×
w/o Token-EE	91.5	2.47×	87.2/81.5	2.83×	68.6	1.72×	89.9	2.88×	89.9	1.71×

Table 2: Results of the ablation study of the proposed MPEE model.

Method	SST-2		QNLI	
	Acc	ρ	Acc	ρ
Class Attention	91.6	4.40×	90.1	2.45×
All-token Attention	91.3	4.27×	89.7	2.36×

Table 3: Different methods of selecting the informative token.

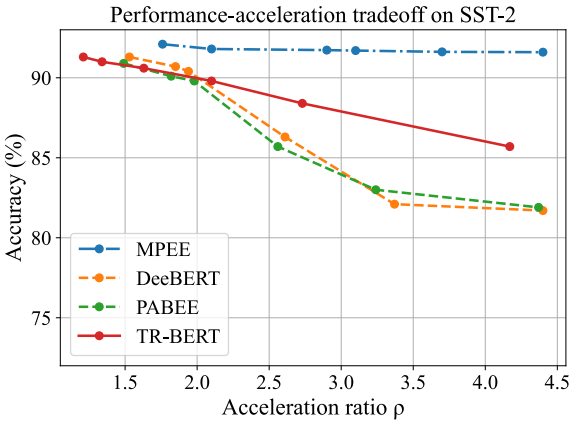

Figure 3: Performance-acceleration tradeoff for MPEE, DeeBERT, TR-BERT and PABEE.

tive EE simultaneously lead to significant acceleration inference.

The informative tokens were selected based on class attention, which is convenient and does not introduce additional parameters and calculations. To show the validity of class attention, we compare it with another choice of informative token. Another way to select informative tokens is to sum the attention weights of all tokens, which is noted as the all token attention, as shown in Table 3. Using class attention has better performance, especially for similarity acceleration inference. The results show that class attention is a better guide for selecting informative tokens because the model prediction is based on the `[CLS]` token and averaging all token attention when there is too much redundant information will dilute the important token weights down to the selection informative token confusion.

4.7 Analysis

Performance-Acceleration Tradeoff. To further demonstrate the performance and efficiency tradeoff between the proposed method and baseline methods, Figure 3 shows the performance and acceleration tradeoff curves. Different accelerations can be obtained by changing the threshold F and sequence EE ratio R_{ee}. In addition, in TR-BERT, the parameters need to be changed to retrain to obtain different accelerations, which increases the computational resources. This limits the application of TR-BERT on different mobile devices. As shown in Figure 3, the performance of the existing EE methods decreases sharply with increasing acceleration ratios, which is also due to the poor performance of shallow EE classifiers caused by the weak representation. MPEE outperforms TR-BERT in terms of acceleration and performance. The proposed method simultaneously accelerates inference in multiple perspectives. Preserving EE token information and weighted self-distillation maintains the performance of the model. Another reason is that TR-BERT reduces the sequence length in two layers, which requires discarding more tokens to reach similar acceleration, while we early exit tokens in more layers. MPEE can achieve a better tradeoff between performance and efficiency.

Sample Distribution of Early Exiting. To better demonstrate the superiority of the proposed method over other methods in accelerating the inference

of PLMs, Figure 4 shows the statistical information of the number of samples exiting early in different layers. The proposed MPEE tends to exit the model inference at an earlier classifier than DeeBERT with a higher model performance. This indicates that the proposed model can exit early while maintaining high performance. Another observation is that nearly half of the samples of the proposed model exit the model immediately at the first layer, but the acceleration ratio ρ is 4×. This is because the horizontal perspective EE further reduces the computation and improves the acceleration inference and because the multi-perspective early exiting is orthogonal.

Performance of Different Class Attention Scores. To verify whether the class attention has the ability to discriminate the information token. We exit the tokens of the top 70% and the bottom 70% class attention score tokens separately to obtain the model performance, as shown in Figure 5. The performance of tokens with high scores is better than the performance of tokens with low scores. The selection of the bottom 70% class attention score tokens leads to increasingly low performance, which is due to the model selecting tokens with less information layer by layer. This indicates that class attention can be used to obtain the more informative tokens.

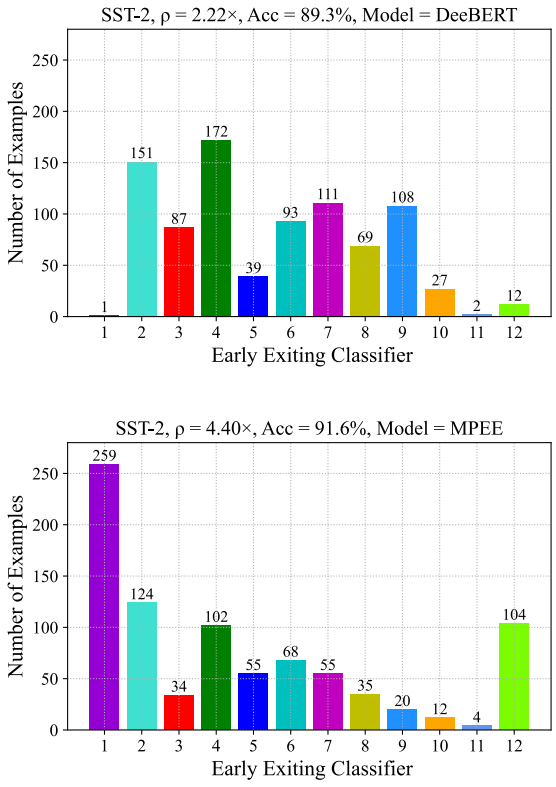

Figure 4: Distribution of different performances of early exiting classifiers on SST-2.

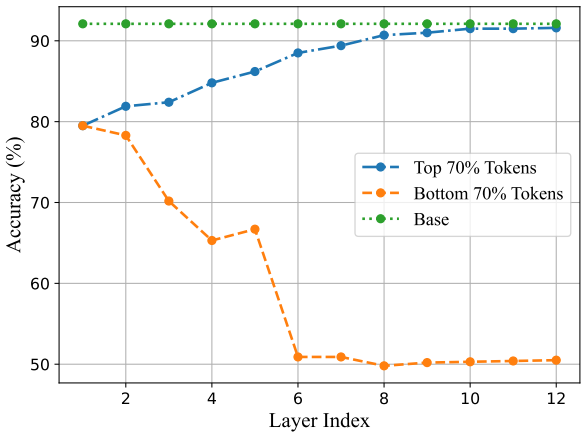

Figure 5: The performance of the top 70% and bottom 70% class attention score on SST-2.

5 Related Work

Model Compression. Knowledge distillation (Sun et al., 2020) refers to the training of smaller student models using the knowledge supervision of pre-trained larger teacher models. The student model uses fewer layers to learn knowledge from the teacher's hidden units and logits, e.g., DistilBERT (Sanh et al., 2019) and BERT-PKD (Sun et al., 2019). Pruning (Wang et al., 2020) refers to the removal of less important weights or computational units. Voita et al. (2019) analyzed multi-head self-attention importance and removed it. Sajjad et al. (2020) discarded unimportant layers in the fine-tuning process. PoWER-BERT (Goyal et al., 2020) progressively reduces the sequence length and accelerates the BERT model. However, ignoring the deleted token leads to information loss and thus reduces the performance of the model. Ye et al. (2021) proposed using reinforcement learning to select unimportant tokens to reduce the sequence length. This approach requires additional strategies to converge the model. Quantification (Shen et al., 2020) is the process of reducing the number of bits needed to represent the model weights. Gao et al. (2021) proposed to quantize the activation function and the weight parameters simultaneously to reduce quantization errors.

Early Exiting. The early exiting approach refers to allowing different samples to exit early in different layers depending on the properties of the input samples. DeeBERT (Xin et al., 2020) performs

early prediction by introducing multiple EE classifiers and exits early if the confidence is greater than the threshold, and conversely passes to the next layer to continue the computation. FastBERT (Liu et al., 2020) uses self-distillation to train EE classifiers. PABEE (Zhou et al., 2020) achieves early exit inference when EE classifier predictions are held continuously constant.

6 Conclusions

In this paper, we propose a unified multi-perspective early exiting framework that significantly reduces the computation cost and improves acceleration inference within a small performance loss. The multi-perspective early exit framework includes horizontal and vertical perspectives early exiting. It has horizontal sparsity and vertical sparsity for faster inference. Extensive experimental results show that compared to previous approaches, the proposed model provides a better tradeoff between model performance and inference efficiency. Future work attempt to extend the proposed approach to vision and language pretrained models, taking into account the properties of different modalities.

Acknowledgements

This work was supported by the National Natural Science Foundation of China (NSFC) under Grants No. 61966038, and the Ministry of Science and Technology, Taiwan, ROC, under Grant No. MOST 111-2628-E-155-001-MY2. The authors would like to thank the anonymous reviewers for their constructive comments.

References

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL-HLT 2019)*, pages 4171–4186.

William B. Dolan and Chris Brockett. 2005. Automatically constructing a corpus of sentential paraphrases. In *Proceedings of the Third International Workshop on Paraphrasing (IWP 2005)*, pages 9–16.

Kawin Ethayarajh. 2019. How contextual are contextualized word representations? Comparing the geometry of BERT, ELMO, and GPT-2 embeddings. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP 2019)*, pages 55–65.

Angela Fan, Edouard Grave, and Armand Joulin. 2019. Reducing transformer depth on demand with structured dropout. In *arXiv preprint arXiv:1909.11556*.

Zhifu Gao, Yiwu Yao, Shiliang Zhang, Jun Yang, Ming Lei, and Ian McLoughlin. 2021. Extremely low footprint end-to-end ASR system for smart device.

Saurabh Goyal, Anamitra Roy Choudhury, Saurabh M. Raje, Venkatesan T. Chakaravarthy, Yogish Sabharwal, and Ashish Verma. 2020. PoWER-BERT: Accelerating BERT inference via progressive word-vector elimination. In *Proceedings of 37th International Conference on Machine Learning (ICML 2020)*, volume PartF16814, pages 3648–3657.

Geoffrey Hinton, Oriol Vinyals, and Jeff Dean. 2015. Distilling the knowledge in a neural network. *arXiv preprint arXiv:1503.02531*.

Xiaoqi Jiao, Yichun Yin, Lifeng Shang, Xin Jiang, Xiao Chen, Linlin Li, Fang Wang, and Qun Liu. 2020. Tinybert: Distilling bert for natural language understanding. In *Findings of the Association for Computational Linguistics (EMNLP 2020)*, pages 4163—-4174.

Josef Klafka and Allyson Ettinger. 2020. Spying on your neighbors: Fine-grained probing of contextual embeddings for information about surrounding words. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics (ACL 2020)*, pages 4801–4811.

Zhenzhong Lan, Mingda Chen, Sebastian Goodman, Kevin Gimpel, Piyush Sharma, and Radu Soricut. 2019. Albert: A lite bert for self-supervised learning of language representations. *arXiv preprint arXiv:1909.11942*.

Zi Lin, Jeremiah Liu, Zi Yang, Nan Hua, and Dan Roth. 2020. Pruning redundant mappings in transformer models via spectral-normalized identity prior. In *Findings ofthe Association for Computational Linguistics (EMNLP 2020)*, pages 719–730.

Weijie Liu, Peng Zhou, Zhe Zhao, Zhiruo Wang, Haotang Deng, and Qi Ju. 2020. FastBERT: A self-distilling BERT with adaptive inference time.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. RoBERTa: A robustly optimized BERT pretraining approach. *arXiv preprint arXiv:1907.11692*.

Paul Michel, Omer Levy, and Graham Neubig. 2019. Are sixteen heads really better than one? In *Proceedings of the 33th International Conference on Neural Information Processing Systems (NeurIPS-2019)*.

Pranav Rajpurkar, Jian Zhang, Konstantin Lopyrev, and Percy Liang. 2016. SQuad: 100,000+ questions for machine comprehension of text. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing (EMNLP 2016)*, pages 2383–2392.

Hassan Sajjad, Fahim Dalvi, Nadir Durrani, and Preslav Nakov. 2020. Poor man's BERT: Smaller and faster transformer models hassan. *arXiv preprint arXiv:2004.03844*.

Victor Sanh, Lysandre Debut, Julien Chaumond, and Thomas Wolf. 2019. DistilBERT, a distilled version of BERT: Smaller, faster, cheaper and lighter. *arXiv preprint arXiv:1910.01108*.

Victor Sanh, Thomas Wolf, and Alexander Rush. 2020. Movement pruning: Adaptive sparsity by fine-tuning. In *Advances in Neural Information Processing Systems (NeurIPS 2020)*, volume 33, pages 20378–20389.

Roy Schwartz, Gabriel Stanovsky, Swabha Swayamdipta, Jesse Dodge, and Noah A. Smith. 2020. The right tool for the job: Matching model and instance complexities. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics (ACL 2020)*, pages 6640–6651.

Sheng Shen, Zhen Dong, Jiayu Ye, Linjian May, Zhewei Yao, Amir Gholami, Michael W. Mahoney, and Kurt Keutzer. 2020. Q-BERT: Hessian based ultra low precision quantization of BERT. In *Proceedings of The 34th AAAI Conference on Artificial Intelligence (AAAI 2020)*, pages 8815–8821.

Richard Socher, Alex Perelygin, Jean Y. Wu, Jason Chuang, Christopher D. Manning, Andrew Y. Ng, and Christopher Potts. 2013. Recursive deep models for semantic compositionality over a sentiment treebank richard. In *In Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing (EMNLP 2013)*, pages 1631–1642.

Siqi Sun, Yu Cheng, Zhe Gan, and Jingjing Liu. 2019. Patient knowledge distillation for BERT model compression. In *Proceedings ofthe 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP/IJCNLP 2019)*, pages 4323–4332.

Zhiqing Sun, Hongkun Yu, Xiaodan Song, Renjie Liu, Yiming Yang, and Denny Zhou. 2020. MobileBERT: A compact task-agnostic BERT for resource-limited devices. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics (ACL 2020)*, pages 2158–2170.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Proceedings of the 31th International Conference on Neural Information Processing Systems (NeurIPS 2017)*, pages 5998–6008.

Elena Voita, David Talbot, Fedor Moiseev, Rico Sennrich, and Ivan Titov. 2019. Analyzing multi-head self-attention: Specialized heads do the heavy lifting, the rest can be pruned. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics (ACL 2019)*, pages 5797–5808.

Alex Wang, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel R. Bowman. 2019. Glue: A multi-task benchmark and analysis platform for natural language understanding. In *Proceedings of The 7th International Conference on Learning Representations (ICLR 2019)*, pages 353–355.

Ziheng Wang, Jeremy Wohlwend, and Tao Lei. 2020. Structured pruning of large language models. In *Proceedings ofthe 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP 2020)*, pages 6151–6162.

Adina Williams, Nikita Nangia, and Samuel R. Bowman. 2018. A broad-coverage challenge corpus for sentence understanding through inference. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL-HLT 2018)*, pages 1112–1122.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Morgan Funtowicz, Joe Davison, Sam Shleifer, Patrick Von Platen, Clara Ma, Yacine Jernite, Julien Plu, Canwen Xu, Teven Le Scao, Sylvain Gugger, Mariama Drame, Quentin Lhoest, and Alexander M Rush. 2020. Transformers: State-of-the-Art Natural Language Processing. In *In Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 38–45.

Ji Xin, Rodrigo Nogueira, Yaoliang Yu, and Jimmy Lin. 2020. Early exiting BERT for efficient document ranking. In *Proceedings of SustaiNLP: Workshop on Simple and Efficient Natural Language Processing*, pages 83–88.

Deming Ye, Yankai Lin, Yufei Huang, and Maosong Sun. 2021. TR-BERT: Dynamic token reduction for accelerating BERT inference. In *Proceedings ofthe 2021 Conference ofthe North American Chapter ofthe Association for Computational Linguistics (ACL 2021)*, pages 5798–5809.

Ofir Zafrir, Guy Boudoukh, Peter Izsak, and Moshe Wasserblat. 2019. Q8BERT: Quantized 8Bit BERT. *arXiv preprint arXiv:1910.06188*.

Wangchunshu Zhou, Canwen Xu, Tao Ge, Julian McAuley, Ke Xu, and Furu Wei. 2020. BERT loses patience: Fast and robust inference with early exit. In *Proceedings of the 34th International Conference on Neural Information Processing Systems (NeurIPS 2020)*, pages 18330—-18341.

Topology imbalance and Relation inauthenticity aware Hierarchical Graph Attention Networks for Fake News Detection

Li Gao[1], Lingyun Song[*1,2], Jie Liu[1], Bolin Chen[1,2], Xuequn Shang[*1,2]
[1]School of Computer Science, Northwestern Polytechnical University, Xi'an, China
[2]Key Laboratory of Big Data Storage and Management,
Ministry of Industry and Information Technology, Xi'an, China
{lysong,shang}@nwpu.edu.com

Abstract

Fake news detection is a challenging problem due to its tremendous real-world political and social impacts. Recent fake news detection works focus on learning news features from News Propagation Graph (NPG). However, little attention is paid to the issues of both authenticity of the relationships and topology imbalance in the structure of NPG, which trick existing methods and thus lead to incorrect prediction results. To tackle these issues, in this paper, we propose a novel **T**opology imbalance and **R**elation inauthenticity aware **H**ierarchical **G**raph **A**ttention **N**etworks (**TR-HGAN**) to identify fake news on social media. Specifically, we design a new topology imbalance smoothing strategy to measure the topology weight of each node. Besides, we adopt a hierarchical-level attention mechanism for graph convolutional learning, which can adaptively identify the authenticity of relationships by assigning appropriate weights to each of them. Experiments on real-world datasets demonstrate that TR-HGAN significantly outperforms state-of-the-art methods.

1 Introduction

Fake news is a news article that is intentionally and verifiably false and could mislead readers (Rashkin et al., 2017; Zhou and Zafarani, 2020). The widespread of fake news can immensely affect cyberspace security and even social stability (Lazer et al., 2018). For example, fake news "all of Walmart's e-commerce stores would include a pay with litecoin option beginning October 1"[1] causes Litecoin's prices surged by over 37% to $236 and Bitcoin's prices fell below $44,000 (Kogan et al., 2021), resulting in huge losses for many investors.

Various fake news detection methods are developed to classify whether the news is fake or not, which can be roughly divided into news content-based methods and social context-based methods. Existing news content-based methods (Qazvinian et al., 2011; Maddock et al., 2015; Jin et al., 2013; Wu et al., 2015; Ma et al., 2017) typically focus on mining lexical and syntactic features (Feng et al., 2012; Potthast et al., 2017; Conroy et al., 2015) from news contents, ignoring the rich structural information of news propagation. To address this limitation, social context-based methods (Yuan et al., 2019; Yang et al., 2021; Shu et al., 2019; Yuan et al., 2020) tend to learn the feature representations by the structure information of News Propagation Graph (NPG), where NPG consists of multiple types of nodes (e.g., news, comments and users) and relationships (e.g., follower, retweet and friendship).

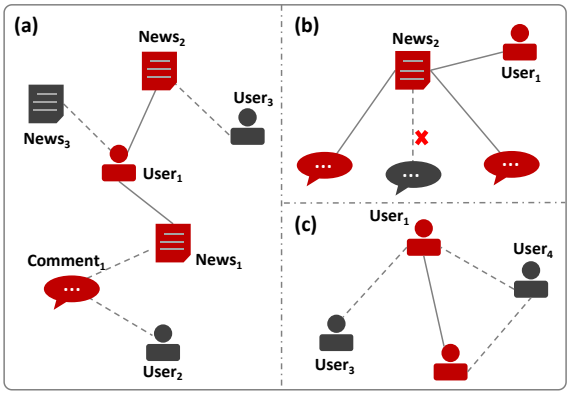

Figure 1: Three types of unauthentic relation scenarios on news propagation graphs (NPG). Nodes representing fake objects (e.g., news or comments) and abnormal users are highlighted in red, while the unauthentic relations are represented by dotted lines. **(a)** An abnormal user $User_1$ forwards a true news $News_3$, and he manipulates or tricks a normal user $User_2$ to create a fake supporting comment on a fake news $News_1$, or he hacks a normal user $User_3$ to forward a fake news $News_2$. **(b)** A fake news producer $User_1$ deletes real opposing comments produced by normal users. **(c)** An abnormal user $User_1$ follows many normal users to disguise himself.

*Corresponding authors: Lingyun Song, Xuequn Shang
[1]https://genesisblockhk.com/fake-news-litecoin-and-walmart-are-not-partnering-in-payments/

Proceedings of the 29th International Conference on Computational Linguistics, pages 4687–4696
October 12–17, 2022.

Although these social context-based methods achieve promising performance, they have an ill-advised assumption that impair the performance of fake news detection. Specifically, they assume a piece of news connecting with a trustworthy user should also have high credit. However, in many cases, the relationships in NPG can be manipulated by users and thus are unauthentic. For example, we present several types of unauthentic relations in Figure 1, where a fake news creator could manipulate other users to create fake comments to support the fake news, or delete real comments opposing the fake news. This leads to inaccurate NPG with unauthentic propagation structure, which confuses existing social context-based methods to make an incorrect prediction.

Besides, NPG usually has the problem of topology imbalance: the imbalance caused by the asymmetric and uneven topology of labeled nodes, where the decision boundaries are driven by the labeled nodes close to the topological class boundaries thus interfering with the model learning (Chen et al., 2021). Ideally, the influence from labeled nodes should decay with the topology distance and also the node influence boundaries should be consistent with the true class boundaries. But the topology imbalance issue would cause the node influence boundaries to deviate from the true class boundaries, resulting in inaccurate results. Nevertheless, most approaches neglect this issue in NPG, interfering with the detection results.

To tackle these issues, we propose a novel **T**opology imbalance and **R**elation inauthenticity aware **H**ierarchical **G**raph **A**ttention **N**etworks (**TR-HGAN**) for fake news detection. We firstly design a topology smoothing strategy to measure the weights of labeled nodes to alleviate the topology imbalance issue. Then we propose a hierarchical-level attention mechanism to identify the authenticity of relations by measuring the appropriate weights to each of them, which can effectively reduce the influence of the inauthentic relationships in NPG. The main contributions of this paper are as follows:

- We study a novel topology smoothing strategy to address the problem of topology imbalance of NPG. To the best of our knowledge, this is the first attempt to solve the topology imbalance issue of the NPG during fake news detection.
- We develop a hierarchical graph attention mechanism which can effectively identify the authenticity of the relationships by assigning appropriate weights to each of them.
- Experiments on three real world datasets demonstrate that TR-HGAN achieves state-of-the-art performances on fake news classification.

2 Related Work

Fake news detection challenges the usage of related information (such as text content, comments, propagation patterns, etc.) to distinguish whether a news article is fake or not. Related works can be divided into two perspectives: i) News content-based methods; ii) Social context-based methods.

2.1 News Content-based Methods

News content-based methods concentrate on designing some textual features such as content writing styles (Shu et al., 2017), lexical and syntactic features (Feng et al., 2012; Potthast et al., 2017; Conroy et al., 2015) to detect the truthfulness of news articles. For instance, Potthast et al. (Potthast et al., 2017) extracts various style features from news contents and predict fake news and media bias. Ma et al. (Ma et al., 2016; Yu et al., 2017) captures news features from low-level to high-level with deep neural networks. Although these approaches achieve good performance on fake news detection, they focus on learning text features alone, rarely considering whether the features can be captured by utilizing the news propagation structure.

2.2 Social Context-based Methods

Social context-based methods principally learn social interactions or information propagation structures through neural networks for further detection. Specifically, Wu et al. (Wu et al., 2015) propose a graph kernel-based SVM classifier that aims to learn high-order news propagation patterns of news articles. RvNN (Ma et al., 2018) and BiGCN (Bian et al., 2020) are developed based on bottom-up and top-down propagation trees for learning the embedding of fake news propagation structure. However, these tree-structured based studies don't utilize the social network structure information, neglecting the fact that the information dissemination on social media is essentially spread in the form of heterogeneous graph.

To tackle these issues, (Lu and Li, 2020) constructs an user interaction graph to model the potential interactions between users, and then develop a dual co-attention mechanism to learn the

co-influence features. Recent studies (Yang et al., 2021; Yuan et al., 2019; Nguyen et al., 2020) formulate the news propagation structure as a heterogeneous news propagation graph with various types of nodes (e.g., news, comments, and users), and then apply GNNs model to capture the structure features for fake news detection. However, they don't consider the unauthentic relations and inherent topology imbalance problem in the graph, and thus may fail to detect the intentional fake news.

3 Proposed Model

The overall architecture of TR-HGAN is shown in Figure 2, which involves four main components: i) NPG Construction; ii) Text Embedding; iii) Hierarchical Graph Attention Network that consists of topology imbalance smoothing strategy and hierarchical-level attention strategy; and iv) Fake news classification. Next, we describe each part of TR-HGAN in detail.

3.1 NPG Construction

Let $\mathcal{M} = \{m_1, m_2, ..., m_{|\mathcal{M}|}\}$ be a set of news articles on social media, where m_i is the i-th news articles and $|\mathcal{M}|$ is the number of news. Let $\mathcal{C}(m_i) = \{c_1, c_2, ..., c_n\}$ be a set of comments of m_i, $\mathcal{U} = \{u_1, u_2, ..., u_{|\mathcal{U}|}\}$ be a set of users who create news or comments. To illustrate our motivation, we construct a heterogeneous news propagation graph (NPG) denoted as $\mathcal{G} = (\mathcal{V}, \mathcal{E})$, where $\mathcal{V} = \mathcal{M} \cup \mathcal{U} \cup \mathcal{C}$ covers the sets of news articles, users and comments, $\mathcal{E}$ involves different relationships between nodes. An example of the NPG is shown in the left of Figure 2, where two nodes are connected if they have posting/commenting relationships. Specially, there would be an edge between two news nodes if they share similar neighbors but irrelevant content. Users are connected if they have follower/comment the same news/followee relationships.

3.2 Text Embedding

To obtain the text representation for each news article, we apply CNN (Chen, 2015; Grefenstette et al., 2014) and multi-head self-attention (Vaswani et al., 2017) to obtain the text representation for each text, which is the same as the existing state-of-the-art fake news detection approaches, i.e. GLAN (Yuan et al., 2019) and CGAT (Yang et al., 2021).

Given a source news m_i and its comments $C = \{c_1, c_2, ..., c_n\}$. Firstly, we capture initial sequence feature $\boldsymbol{f_{m_i}} \in \mathbb{R}^d$ of news m_i with CNN. By the same way, the feature $\boldsymbol{f_{c_j}} \in \mathbb{R}^d$ of each comment c_j can be extracted. Then we refine the coherence semantic representation between comments and source news by using multi-head self-attention to capture dependencies across news content and comments. From the above operation, the final text feature $\hat{\boldsymbol{f}}_{\boldsymbol{m_i}}$ for each news and $\hat{\boldsymbol{f}}_{\boldsymbol{c_j}}$ for each comment can be extracted. The initial feature $\hat{\boldsymbol{f}}_{\boldsymbol{u_k}}$ of user node u_k can be calculated by the user profile data (such as friends count, followers count, status count, etc.).

3.3 Hierarchical Graph Attention Network

3.3.1 Topology Imbalance Smoothing Strategy

To tackle the topology imbalance problem in NPG, we propose a strategy of smoothing the topology structure to alleviate the resulting problems. We assume that if a labeled news node $m_i \in \mathcal{V}$ encounters strong influence from the other labeled neighbor nodes, the node m_i owns great influence and is close to topological class boundaries. In other words, the influence of labeled nodes should decay with the topology distance. To measure how topologically close node m_i is to the center of the class it belongs to, we calculate the topology location value $\boldsymbol{T}_m$ by measuring the expectation of message-passing probability between the node m_i and its neighbors when node m_i randomly walks across the entire graph:

$$\boldsymbol{T_m} = \mathbb{E}_{x \sim P_m,:}[\sum_{j \in [1,s], j \neq y_m} \frac{1}{|\mathcal{S}_j|} \sum_{i \in \mathcal{S}_j} P_{i,x}], \quad (1)$$

where y_m is the ground-truth label of node m_i. P_m indicates the personalized PageRank probability (Page et al., 1999) vector for the node m_i, which can be viewed as the distribution of influence exerted outward from each m_i. $P_{i,x}$ indicates the probability from m_i to m_x. $\mathcal{S}_j$ represents the training sets for different classes, where s is the number of classes. The normalization item $\frac{1}{|S_j|}$ is added to make the influence from the different classes comparable when computing conflict.

The larger the topology location value $\boldsymbol{T}_m$ of m_i, the more topologically closer to class boundaries the node m_i. Figure 2.(A) apparently shows that it can decrease the training weights of labeled nodes c_2 and increase the weights of labeled nodes u_4 close to the fake news m_i, thus relieving the topology-imbalance issue. Inspired by the study

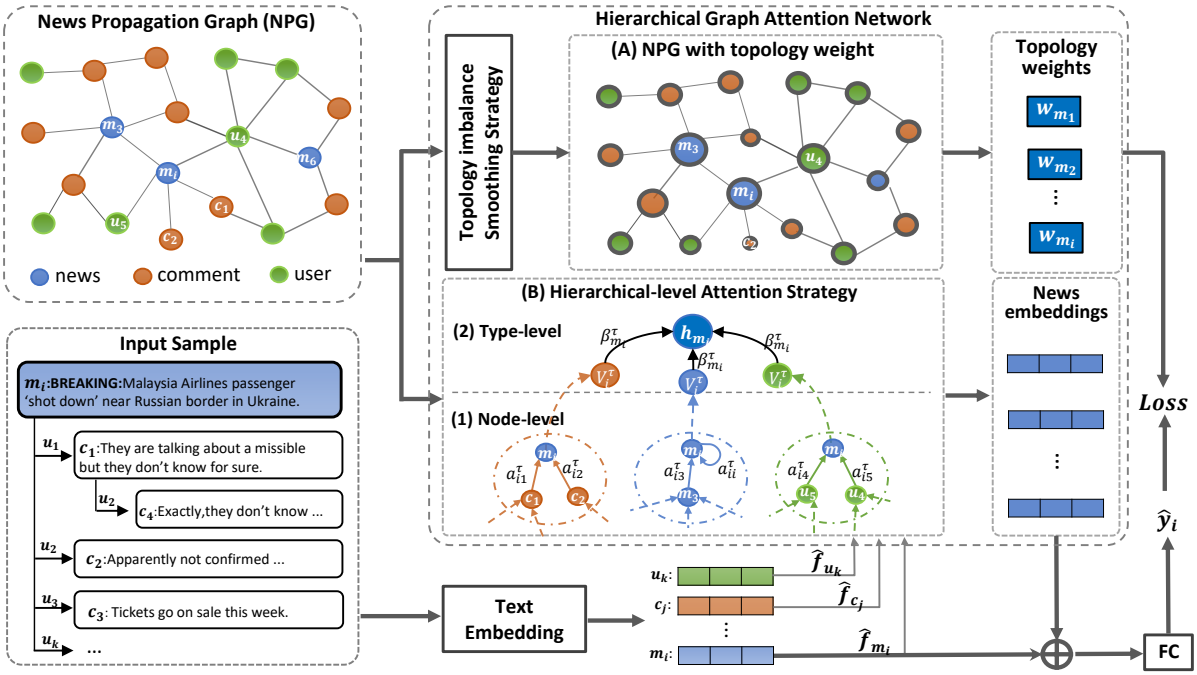

Figure 2: The architecture of TR-HGAN. According to the given *Input Sample*, we construct a *News Propagation Graph (NPG)* which consists of multiple types of nodes and relations. *Topology Imbalance Smoothing Strategy* can increase the weight of great influence node and decrease the weight of week influence one, and finally get the topology weight for each node. *Hierarchical-level Attention Strategy* can adaptively identify the authenticity of the relationship by assigning appropriate weight a_{ij}^{τ} for each edge, where i, j indicate the nodes in NPG, and τ indicates the type of node (i.e. news, comment, user). For simplicity of description, we give the graph learning of one-hop neighbors of m_i as an example in *Hierarchical-level Attention Strategy*. For fake news detection, we only extract the topology weights and features of the news node m_i for final classification. The initial feature of each node is extracted through *Text Embedding*.

(Chen et al., 2021), in order to promote the training weights of nodes for effective model learning, we also train node weights based on their topology location values by:

$$\boldsymbol{w}_m = w_{min} + \frac{w_{max} - w_{min}}{2}(1 + cos\frac{Rank(\boldsymbol{T_m})}{|Y|}\pi), \tag{2}$$

where $\boldsymbol{w}_m$ is the modified training weight for m_i, w_{min}, w_{max} indicates the lower bound and upper bound of the weight correction factor, Y is the set of labeled news article nodes, $Rank(\boldsymbol{T}_m)$ is the ranking order of $\boldsymbol{T}_m$ from the smallest to the largest.

3.3.2 Hierarchical-level Attention Strategy

Considering nodes u, m, c are three different types of nodes belong to the different semantic spaces, the GCN (Kipf and Welling, 2016) cannot be directly applied to the NPG due to the node heterogeneity issue. Surprised by the recent work on heterogeneous graph convolution (Linmei et al., 2019), we employ a hierarchical attention mechanism to learn the representation of each node. Specifically, to alleviate the negative effects of those unauthentic relations, we adjust the weights between two nodes by type-level attention and node-level attention learning.

1) *Node-level attention.* For each type of nodes (e.g., node with type τ, where $\tau \in \{u, m, c\}$), we firstly design the type-specific transformation matrix $\boldsymbol{M}_{\tau}$ to project the features of different types of nodes into the same feature space. Take the news article node m_i as an example, given the input feature vectors $\hat{\boldsymbol{f}}_{m_i}$, the projection process can be shown as:

$$\boldsymbol{h}'_{m_i} = \boldsymbol{M}_{\tau} \cdot \hat{\boldsymbol{f}}_{m_i}. \tag{3}$$

Similarly, the initial features of each comment and each user can be projected as $\boldsymbol{h}'_{c_j}, \boldsymbol{h}'_{u_k}$, respectively. Thus, the initial input features of the node-level attention layer are the $\boldsymbol{h}'_{c_j}, \boldsymbol{h}'_{u_k}, \boldsymbol{h}'_{m_i}$.

In the face of fake news detection, the target node is the news article node $m_i \in \mathcal{M}$ with the type τ. The neighbors of it belong to $\mathcal{V} \in \{\mathcal{M}, \mathcal{U}, \mathcal{C}\}$ with

the type $\tau^{'}$. Given a node pair (m_i, v_j), where $v_j \in \mathcal{V}$, we design the node-level attention to get the weight coefficient different neighboring nodes via:

$$\boldsymbol{\alpha}_{ij}^{\tau} = softmax\left(\sigma(\boldsymbol{\mu_\tau}[\boldsymbol{h}_{\boldsymbol{m_i}}^{'}||\boldsymbol{h}_{\boldsymbol{v_j}}^{'}])\right), \quad (4)$$

where $\boldsymbol{\mu_\tau}$ is the attention parameter for the type τ, $||$ means concatenation operation, $\sigma(\cdot)$ denotes LeakyReLU function. Then, the type-level node $\boldsymbol{\mathcal{V}}_{\boldsymbol{m_i}}^{\boldsymbol{\tau}}$ on graph can be aggregated by the neighbor's projected features with the corresponding weights as follows:

$$\boldsymbol{\mathcal{V}}_{\boldsymbol{m_i}}^{\boldsymbol{\tau}} = \sigma(\sum_{v_j \in \mathcal{V}} \boldsymbol{\alpha}_{ij}^{\tau} \cdot \boldsymbol{h}_{v_j}^{'}). \quad (5)$$

2) *Type-level attention.* Through the node-level attention, we fuse information from neighbor nodes with the same type into the representation of a type-level node. Different types of nodes contain type-specific information, which requires us to learn the importance of different node types. Thus, we adopt the type-level attention to learn the weights of news article nodes m_i from all types of nodes by:

$$\boldsymbol{\beta}_{m_i}^{\tau} = softmax\left(\sigma(\boldsymbol{W}[\boldsymbol{\mathcal{V}}_{\boldsymbol{m_i}}^{\boldsymbol{\tau}}||\boldsymbol{\mathcal{M}}_{\boldsymbol{m_i}}])\right), \quad (6)$$

where $\boldsymbol{W}$ is the weight matrix. Finally, we incorporate all type-level nodes to get the final representation $\boldsymbol{h}_{m_i}$ of the target news node m_i:

$$\boldsymbol{h}_{m_i} = \sum_{\tau} \boldsymbol{\beta}_{m_i}^{\tau} \cdot \boldsymbol{\mathcal{V}}_{m_i}^{\tau}. \quad (7)$$

3.4 Fake News Classification

For the target news m_i, we aim at learning an inference function $f : \mathcal{G} \rightarrow \mathcal{Y}$ to predict whether it is fake or not. After the above procedures, we get the structure features $\boldsymbol{h}_{m_i}$, as well as its coherence-based sentence representation $\hat{\boldsymbol{f}}_{m_i}$, and then concatenate them as final features. Then the final representations of news are fed to softmax classifier based on fully-connected layers to obtain category probability $\hat{\boldsymbol{y}}_i$:

$$\hat{\boldsymbol{y}}_{m_i} = softmax(\boldsymbol{W}[\boldsymbol{h}_{\boldsymbol{m_i}}; \hat{\boldsymbol{f}}_{\boldsymbol{m_i}}] + b). \quad (8)$$

Finally, the cross-entropy loss is used as the optimization objective function for fake news detection:

$$\boldsymbol{\mathcal{L}}(\boldsymbol{\Theta}) = -\frac{1}{|Y|} \sum_{m \in Y} \boldsymbol{w_m} \sum_{j=1}^{s} \boldsymbol{y}_{m_i}^{j} \log \hat{\boldsymbol{y}}_{m_i}^{j}, \quad (9)$$

where y_m is the gold probability of fake news class, s is the number of classes and Θ represents all parameters of the model.

4 Results and Discussion

4.1 Datasets

In order to evaluate the performance of TR-HGAN, we conduct experiments on three benchmark datasets: Weibo (Ma et al., 2016), Twitter15 (Liu et al., 2015) and Twitter16 (Ma et al., 2017). Table 1 gives statistics of the three datasets. The Weibo dataset contains binary labels, i.e., fake news (FR) and non-fake news (NR), whereas Twitter15 and Twitter16 datasets have four types of labels, i.e., fake news (FR), non-fake news (NR), unverified news (UR), and true news (TR), where the label true news denotes a news article that debunks the fake news. For a fair comparison, for each dataset, we randomly select 10% of the dataset as the validation subset, and divide the rest data into training and testing subsets with a ratio of 3:1.

Statistic	Weibo	Twitter15	Twitter16
source news	4664	1490	818
fake news	2313	370	205
non-fake news	2351	374	205
unverified news	0	374	203
true news	0	372	205
users	2,746,818	276,663	173,487
comments	3,805,656	331,612	204,820

Table 1: Dataset statistics.

4.2 Baselines

To highlight performance superiority of the proposed TR-HGAN, we select a series of state-of-the-art methods as baselines. The first two methods try to capture fake news features by using content-based structure. The next six methods extract fake news features through social contexts-based structure, among which, the first four methods are tree-structured models and the last two models are pure heterogeneous graph-based approaches. They are described as follows:

- **SVM-TS** (Ma et al., 2015): An SVM model that utilizes time-series to model the variation of hand-crafted features of news.
- **GRU** (Ma et al., 2016): A RNN-based model that captures the temporal contextual information of relevant retweets or comments.
- **RvNN** (Ma et al., 2018): A tree-structured recursive neural method that learns fake news features via the news propagation structure.

- **PPC** (Liu and Wu, 2018): A propagation-based approach that detect fake news with a combination of recurrent and convolutional networks.

- **Bi-GCN** (Bian et al., 2020): A novel Bi-directional model that explores fake news characteristics by operating on both top-down and bottom-up propagation of fake news.

- **EBGCN** (Wei et al., 2021): A propagation-based method that adaptively rethinks the reliability of latent edge-wise relations.

- **CGAT** (Yang et al., 2021): An end-to-end graph-based framework that jointly exploits text and structure information by using graph adversarial learning framework.

- **GLAN** (Yuan et al., 2019): A graph-based method that encodes contextual information and global structural information by adopting a global-local attention network.

4.3 Experimental Setup

For fair comparison, we adopt the same evaluation metrics used in the prior studies (Yuan et al., 2019; Bian et al., 2020; Wei et al., 2021), we also add accuracy (Acc.), precision (Prec.), recall (Rec.) and $\boldsymbol{F_1}$ score as the evaluation metrics.

For text embedding step, for each source news m_i, we truncate the text if its length is larger than 150 words and pad zero if the length is smaller than 150. All word embeddings of the model are initialized with the 300-dimensional word vectors, which is released by (Yuan et al., 2019). The convolutional kernel size k is set to (4, 5, 6) with 100 kernels for each kind of size. Therefore, the final text representation $\boldsymbol{f}_{m_i} \in \mathbb{R}^d$ of news m_i are concatenated by all feature vectors $\boldsymbol{f}^k_{m_i}$ obtained by different filters. Besides, the parameters w_{min}, w_{max} at topology imbalance smoothing strategy is set to [0.25, 0.5, 0.75], [1.25, 1.5, 1.75], respectively.

4.4 Performance Efficiency

Table 2 and Table 3 show the results of fake news detection on *Weibo*, *Twitter15* and *Twitter16* datasets. We bold the best performance of each column in tables, from where we can observe social context-based methods outperform those news content-based methods using only textual features, which reveals the superiority of learning high-level representations for detecting fake news.

Method	Class	Acc.	Prec.	Rec.	$\boldsymbol{F_1}$
SVM-TS	NR	0.857	0.878	0.830	0.857
	FR		0.839	0.885	0.861
GRU	NR	0.910	0.952	0.864	0.906
	FR		0.876	0.956	0.914
RvNN	NR	0.908	0.912	0.897	0.905
	FR		0.904	0.918	0.911
PPC	NR	0.921	0.949	0.889	0.918
	FR		0.896	**0.962**	0.923
BiGCN	NR	0.935	0.925	0.943	0.933
	FR		0.951	0.887	0.917
CGAT	NR	0.939	0.938	0.942	0.941
	FR		0.939	0.938	0.935
GLAN	NR	0.948	0.937	0.957	0.947
	FR		**0.967**	0.934	0.950
TR-HGAN	NR	**0.963**	**0.957**	**0.964**	**0.960**
	FR		0.962	0.961	**0.960**

Table 2: Fake news detection results on *Weibo* dataset.

In addition, TR-HGAN performs better than the tree-structured based methods (e.g., RvNN, PPC, BiGCN, EBGCN). It can attribute that tree-structured methods neglect the messages are spread by a graph structure (i.e., constructed with source news, comments and users) rather than a tree structure (i.e., only constructed with source news and retweets), which limit the learning of high-level features.

Moreover, the proposed TR-HGAN outperforms state-of-the-art graph-based GLAN on three datasets. Specifically, as Table 2 shows, TR-HGAN achieves improvement of 1.5% on Weibo, comparing with the best baseline GLAN in terms of accuracy. We can also find TR-HGAN obtains 2.5% and 1.7% improvements than the best model on accuracy over all metrics across Twitter15 and Twitter16, respectively. We discuss the fact for two main reasons. First, TR-HGAN considers the inherent unauthentic relations and rich structural features in the news propagation graph. Second, unlike CGAT and GLAN, TR-HGAN pays more attention to the node topology imbalance problem on NPG, which helps improve our models much more.

4.5 Ablation Study

In this part, we test the performance of TR-HGAN variants with different configurations, including:

- ***TR-HGAN w/o Text***: it only uses source news texts for fake news classification.

- ***TR-HGAN w/o C***: it removes the news com-

Method	Twitter15					Twitter16				
	Acc.	NR F_1	FR F_1	TR F_1	UR F_1	Acc.	NR F_1	FR F_1	TR F_1	UR F_1
SVM-TS	0.544	0.796	0.472	0.404	0.483	0.574	0.755	0.420	0.571	0.526
GRU	0.646	0.792	0.574	0.608	0.592	0.633	0.772	0.489	0.686	0.593
RvNN	0.723	0.682	0.758	0.821	0.654	0.737	0.662	0.743	0.835	0.708
PPC	0.842	0.811	0.875	0.818	0.790	0.863	0.820	0.898	0.843	0.837
BiGCN	0.886	0.891	0.860	0.930	0.864	0.880	0.847	0.869	**0.937**	0.865
EBGCN	0.891	0.864	0.892	0.916	0.867	0.915	0.868	0.899	0.924	0.901
GLAN	0.904	0.922	0.915	0.857	0.923	0.895	0.910	0.875	0.851	0.941
TR-HGAN	**0.929**	**0.930**	**0.928**	**0.925**	**0.935**	**0.932**	**0.923**	**0.910**	0.932	**0.949**

Table 3: Fake news detection results on *Twitter15* and *Twitter16* datasets.

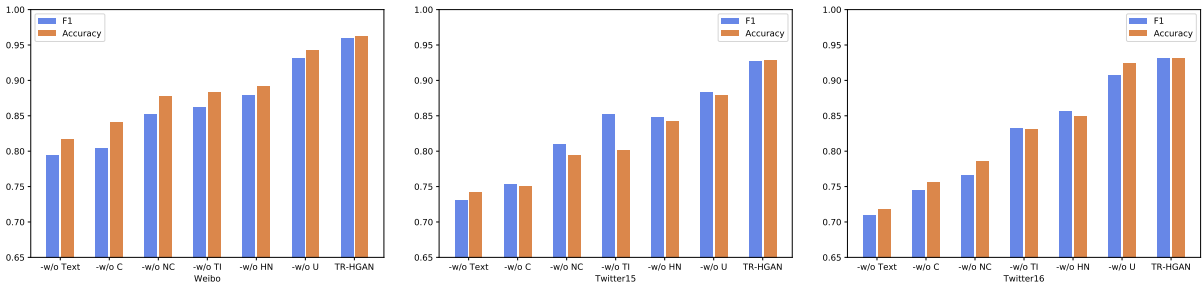

Figure 3: TR-HGAN ablation analysis results on three datasets in terms of Accuracy and F_1.

ments, leaving the source news nodes and users nodes in NPG for fake news detection.

- ***TR-HGAN w/o U***: it removes the users, leaving the source news and comments in graph for fake news detection.
- ***TR-HGAN w/o NC***: it removes the edges between source news and comments.
- ***TR-HGAN w/o HN***: it uses graph convolutional network (GCN) (Kipf and Welling, 2016) to encode the graph structure NPG.
- ***TR-HGAN w/o TI***: it removes the topology imbalance smoothing strategy module for fake news classification.

The comparison results of F_1 and accuracy are shown in Figure 3. We can observe that all the other methods outperform *TR-HGAN w/o Text*, indicating the structure features are important for fake news detection. When we try to remove the information in NPG, the performance apparently decreases (i.e. *TR-HGAN w/o C* and *TR-HGAN w/o NC*) which indicating that the social-aware structure features indeed benefits the performance of fake news detection.

Moreover, *TR-HGAN w/o TI* models the graph structure without considering the topology imbalance problem, resulting the performance drops rapidly, which shows the importance of measuring topology imbalance. Furthermore, the performance of *TR-HGAN w/o HN* gets worse when don't consider the heterogeneity and the unauthentic relations of NPG.

4.6 Early Detection

Early fake news detection aims to detect fake news at the early stage of propagation, which is especially critical for restricting the dissemination scope of fake news. The earlier the detection deadline, the less propagation information such as comments and users are available. To evaluate the performance on early fake news detection, we set up a series of detection deadlines [$0h$, $2h$, $4h$, $6h$, $8h$, $12h$, $24h$].

Figure 4 shows that the proposed TR-HGAN reaches relatively high accuracy at a very early period after the source news initial broadcast. Specifically, TR-HGAN achieves 94% accuracy on Weibo, 87.2% accuracy on Twitter15 and 84.9% on Twitter16 within 2 hours, which is much faster than other models. This is because TR-HGAN takes the reliability of relations in NPG into account by mea-

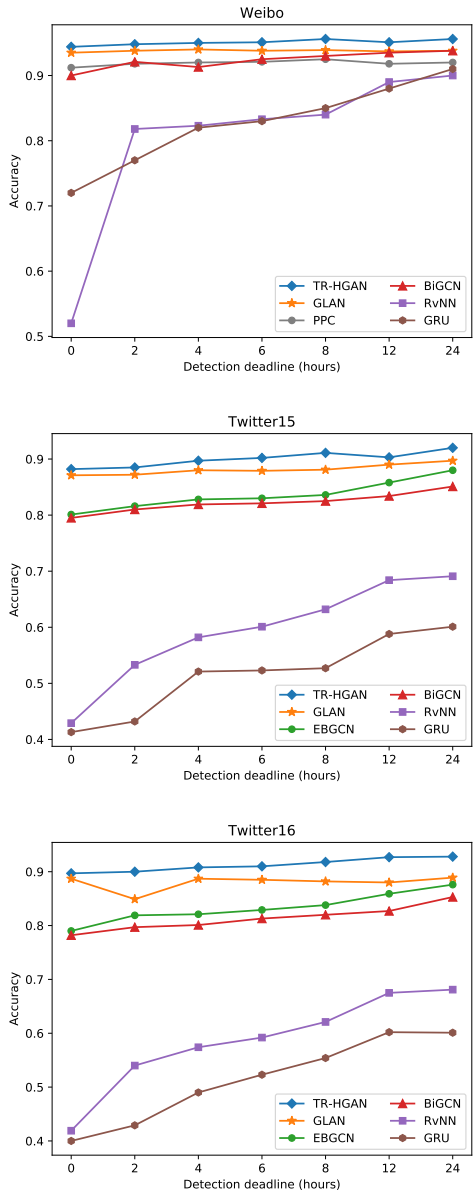

Figure 4: The performance of early fake news detection on three datasets in terms of Accuracy.

suring the importance between different nodes. Besides, TR-HGAN consistently achieves relatively high accuracy score on all datasets than other models at each deadline. This is because TR-HGAN considers the problem of topology imbalance and unauthentic relations in NPG. In the model, the importance of each node can be measured by topology imbalance smoothing strategy, which boosts the performance of detecting results. Second, the unauthentic relations can be refined by the hierarchical attention mechanism, which helps identify unreliable relationships and reduce the noisy of problem nodes in time, so as to detect the authenticity of news as soon as possible.

4.7 Case Study

To further illustrate why our model outperforms state-of-the-art baseline GLAN (Yuan et al., 2019), we randomly sample two fake news from Twitter15 dataset. As depicted in Figure 5, the news, comments and corresponding users are formulated as nodes and relations are modeled as edges in NPG. As shown in the left of Figure 5, we observe that comment c_3 is irrelevant with news m_1 although replying, which reveals the ubiquity of unauthentic relations among news in the NPG and it is necessary to consider the inauthenticity caused by these unauthentic relations.

The right of Figure 5 indicates the constructed weighted NPG. For a target node m_1, existing graph-based models (e.g., GLAN, CGAT) always generate the feature representation of m_1 by aggregating the information of its all neighbors according to seemingly authentic edges. However, edge between node m_1 and c_3 would bring noise features and limit the learning of useful features for fake news detection. The proposed TR-HGAN can successfully weaken the negative effect of this unauthentic edge by hierarchical-level attention network. Besides, the target fake news node m_1 will affect the class boundary shift of its neighbors due to the topology imbalance of NPG. Thus, the topology imbalance smoothing strategy is adopted to decrease the weights of those useless nodes and strengthen the weights of important nodes close to target node. Accordingly, the TR-HGAN is capable of learning more conducive features and can enhance the robustness of results.

5 Conclusion

In this paper, we have studied the unauthentic relations and topology imbalance issue in the news propagation structure from a weight learning perspective on fake news detection. We propose a topology imbalance and relation inauthenticity aware hierarchical graph attention networks (TR-HGAN) to capture robust structural features. Specifically, we design a topology imbalance smoothing strategy to address the problem of topology imbalance in NPG by increasing the weight of nodes with great influence and decreasing the weight of nodes with week influence. Besides, we develop a hierarchical graph attention mechanism for graph convolutional learning, which can adaptively measure the authenticity of the relationships by assigning appropriate weight to each relationship,

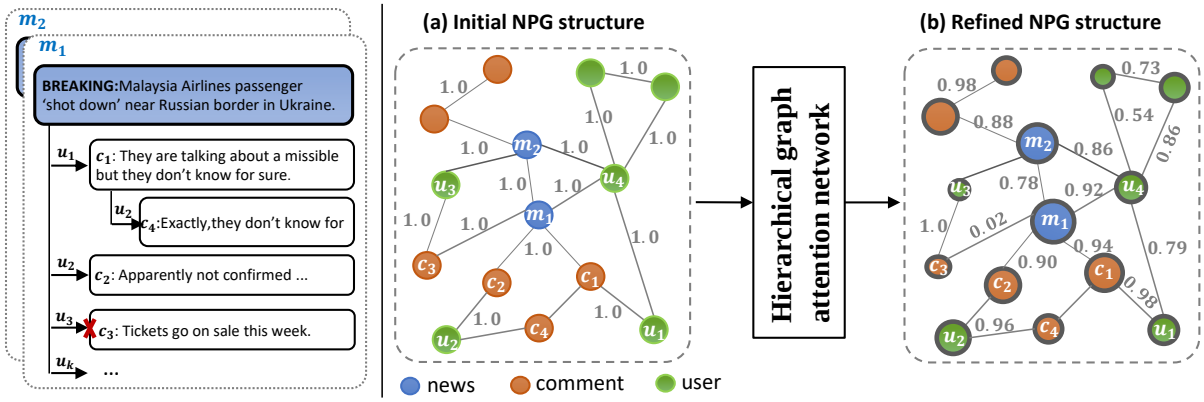

Figure 5: The case study. Left shows two fake news sampled from Twitter15. The comment c_3 is the irrelevant one towards propagation of news m_1 but included in. **(a)** is the constructed NPG based on the news propagation structure. **(b)** shows the result of case study. The proposed TR-HGAN can measure the topology weight of each node to increase the weights of important nodes close to the class centers and decrease the weights of nodes close to class boundary. In addition, TR-HGAN adaptively adjust the weights of edges in NPG to strengthen the effect of authentic edges and weaken the effect of unauthentic edges.

thus effectively reduce the influence of the unauthentic relations. Extensive experiments conducted on three commonly benchmark datasets demonstrate that our model can significantly surpass the state-of-the-art baselines on both fake news classification and early detection tasks.

Acknowledgements

The research is supported by the National Key Research and Development Program of China under Grant 2020AAA0108504, National Nature Science Foundation of China under Grant Nos.62102321, Fundamental Research Funds for the Central Universities under Grant No. D5000200146.

References

Tian Bian, Xi Xiao, Tingyang Xu, Peilin Zhao, Wenbing Huang, Yu Rong, and Junzhou Huang. 2020. Rumor detection on social media with bi-directional graph convolutional networks. In *Proceedings of the AAAI conference on artificial intelligence*, volume 34, pages 549–556.

Deli Chen, Yankai Lin, Guangxiang Zhao, Xuancheng Ren, Peng Li, Jie Zhou, and Xu Sun. 2021. Topology-imbalance learning for semi-supervised node classification. *Advances in Neural Information Processing Systems*, 34.

Yahui Chen. 2015. Convolutional neural network for sentence classification. Master's thesis, University of Waterloo.

Nadia K Conroy, Victoria L Rubin, and Yimin Chen. 2015. Automatic deception detection: Methods for finding fake news. *Proceedings of the association for information science and technology*, 52(1):1–4.

Song Feng, Ritwik Banerjee, and Yejin Choi. 2012. Syntactic stylometry for deception detection. In *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 171–175.

Edward Grefenstette, Phil Blunsom, et al. 2014. A convolutional neural network for modelling sentences. In *ACL*.

Fang Jin, Edward Dougherty, Parang Saraf, Yang Cao, and Naren Ramakrishnan. 2013. Epidemiological modeling of news and rumors on twitter. In *Proceedings of the 7th workshop on social network mining and analysis*, pages 1–9.

Thomas N Kipf and Max Welling. 2016. Semi-supervised classification with graph convolutional networks. *arXiv preprint arXiv:1609.02907*.

Shimon Kogan, Tobias J. Moskowitz, and Marina Niessner. 2021. Social media and financial news manipulation. *CommRN: Communication & Computational Linguistics (Topic)*.

David MJ Lazer, Matthew A Baum, Yochai Benkler, Adam J Berinsky, Kelly M Greenhill, Filippo Menczer, Miriam J Metzger, Brendan Nyhan, Gordon Pennycook, David Rothschild, et al. 2018. The science of fake news. *Science*, 359(6380):1094–1096.

Hu Linmei, Tianchi Yang, Chuan Shi, Houye Ji, and Xiaoli Li. 2019. Heterogeneous graph attention networks for semi-supervised short text classification. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the*

9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP), pages 4821–4830.

Xiaomo Liu, Armineh Nourbakhsh, Quanzhi Li, Rui Fang, and Sameena Shah. 2015. Real-time rumor debunking on twitter. In *Proceedings of the 24th ACM international on conference on information and knowledge management*, pages 1867–1870.

Yang Liu and Yi-Fang Wu. 2018. Early detection of fake news on social media through propagation path classification with recurrent and convolutional networks. In *Proceedings of the AAAI conference on artificial intelligence*, volume 32.

Yi-Ju Lu and Cheng-Te Li. 2020. Gcan: Graph-aware co-attention networks for explainable fake news detection on social media. *arXiv preprint arXiv:2004.11648*.

Jing Ma, Wei Gao, Prasenjit Mitra, Sejeong Kwon, Bernard Jim Jansen, Kam-Fai Wong, and M. Cha. 2016. Detecting rumors from microblogs with recurrent neural networks. In *IJCAI*.

Jing Ma, Wei Gao, Zhongyu Wei, Yueming Lu, and Kam-Fai Wong. 2015. Detect rumors using time series of social context information on microblogging websites. In *Proceedings of the 24th ACM international on conference on information and knowledge management*, pages 1751–1754.

Jing Ma, Wei Gao, and Kam-Fai Wong. 2017. Detect rumors in microblog posts using propagation structure via kernel learning. Association for Computational Linguistics.

Jing Ma, Wei Gao, and Kam-Fai Wong. 2018. Rumor detection on twitter with tree-structured recursive neural networks. Association for Computational Linguistics.

Jim Maddock, Kate Starbird, Haneen J Al-Hassani, Daniel E Sandoval, Mania Orand, and Robert M Mason. 2015. Characterizing online rumoring behavior using multi-dimensional signatures. In *Proceedings of the 18th ACM conference on computer supported cooperative work & social computing*, pages 228–241.

Van-Hoang Nguyen, Kazunari Sugiyama, Preslav Nakov, and Min-Yen Kan. 2020. Fang: Leveraging social context for fake news detection using graph representation. In *Proceedings of the 29th ACM international conference on information & knowledge management*, pages 1165–1174.

Lawrence Page, Sergey Brin, Rajeev Motwani, and Terry Winograd. 1999. The pagerank citation ranking: Bringing order to the web. Technical report, Stanford InfoLab.

Martin Potthast, Johannes Kiesel, Kevin Reinartz, Janek Bevendorff, and Benno Stein. 2017. A stylometric inquiry into hyperpartisan and fake news. *arXiv preprint arXiv:1702.05638*.

Vahed Qazvinian, Emily Rosengren, Dragomir Radev, and Qiaozhu Mei. 2011. Rumor has it: Identifying misinformation in microblogs. In *Proceedings of the 2011 Conference on Empirical Methods in Natural Language Processing*, pages 1589–1599.

Hannah Rashkin, Eunsol Choi, Jin Yea Jang, Svitlana Volkova, and Yejin Choi. 2017. Truth of varying shades: Analyzing language in fake news and political fact-checking. In *Proceedings of the 2017 conference on empirical methods in natural language processing*, pages 2931–2937.

Kai Shu, Amy Sliva, Suhang Wang, Jiliang Tang, and Huan Liu. 2017. Fake news detection on social media: A data mining perspective. *ACM SIGKDD explorations newsletter*, 19(1):22–36.

Kai Shu, Suhang Wang, and Huan Liu. 2019. Beyond news contents: The role of social context for fake news detection. In *Proceedings of the twelfth ACM international conference on web search and data mining*, pages 312–320.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. *Advances in neural information processing systems*, 30.

Lingwei Wei, Dou Hu, Wei Zhou, Zhaojuan Yue, and Songlin Hu. 2021. Towards propagation uncertainty: Edge-enhanced bayesian graph convolutional networks for rumor detection.

Ke Wu, Song Yang, and Kenny Q Zhu. 2015. False rumors detection on sina weibo by propagation structures. In *2015 IEEE 31st international conference on data engineering*, pages 651–662. IEEE.

Xiaoyu Yang, Yuefei Lyu, Tian Tian, Yifei Liu, Yudong Liu, and Xi Zhang. 2021. Rumor detection on social media with graph structured adversarial learning. In *Proceedings of the Twenty-Ninth International Conference on International Joint Conferences on Artificial Intelligence*, pages 1417–1423.

Feng Yu, Qiang Liu, Shu Wu, Liang Wang, Tieniu Tan, et al. 2017. A convolutional approach for misinformation identification. In *IJCAI*, pages 3901–3907.

Chunyuan Yuan, Qianwen Ma, Wei Zhou, Jizhong Han, and Songlin Hu. 2019. Jointly embedding the local and global relations of heterogeneous graph for rumor detection. In *2019 IEEE international conference on data mining (ICDM)*, pages 796–805. IEEE.

Chunyuan Yuan, Qianwen Ma, Wei Zhou, Jizhong Han, and Songlin Hu. 2020. Early detection of fake news by utilizing the credibility of news, publishers, and users based on weakly supervised learning. *arXiv preprint arXiv:2012.04233*.

Xinyi Zhou and Reza Zafarani. 2020. A survey of fake news: Fundamental theories, detection methods, and opportunities. *ACM Computing Surveys (CSUR)*, 53(5):1–40.

Temporal Knowledge Graph Completion with Approximated Gaussian Process Embedding

Linhai Zhang[1] **Deyu Zhou**[1*]
[1] School of Computer Science and Engineering, Key Laboratory of Computer Network and Information Integration, Ministry of Education, Southeast University, China
{lzhang472, d.zhou}@seu.edu.cn

Abstract

Knowledge Graphs (KGs) stores world knowledge that benefits various reasoning-based applications. Due to their incompleteness, a fundamental task for KGs, which is known as Knowledge Graph Completion (KGC), is to perform link prediction and infer new facts based on the known facts. Recently, link prediction on the temporal KGs becomes an active research topic. Numerous Temporal Knowledge Graph Completion (TKGC) methods have been proposed by mapping the entities and relations in TKG to the high-dimensional representations. However, most existing TKGC methods are mainly based on deterministic vector embeddings, which are not flexible and expressive enough. In this paper, we propose a novel TKGC method, TKGC-AGP, by mapping the entities and relations in TKG to the approximations of multivariate Gaussian processes (MGPs). Equipped with the flexibility and capacity of MGP, the global trends as well as the local fluctuations in the TKGs can be simultaneously modeled. Moreover, the temporal uncertainties can be also captured with the kernel function and the covariance matrix of MGP. Moreover, a first-order Markov assumption-based training algorithm is proposed to effective optimize the proposed method. Experimental results show the effectiveness of the proposed approach on two real-world benchmark datasets compared with some state-of-the-art TKGC methods.

1 Introduction

Knowledge Graphs (KGs) provide an efficient way to store world knowledge. Various KGs such as DBpedia (Auer et al., 2007), NELL (Carlson et al., 2010), YAGO (Suchanek et al., 2007) and Freebase (Bollacker et al., 2008) have been constructed and benefited downstream applications such as information retrieval, question answering, etc (Hao et al., 2017; Zhang et al., 2021). Generally, a fact in KG can be represented as a triple (s, p, o), where s (subject) denotes head entity node, o (object) denotes tail entity node and p (predicate) denotes relation edge between them. However, in the real world, some facts are time-aware. For example, the fact *(Joe Biden, presidentOf, the United States)* is not valid until January 21, 2021, the United States presidential inauguration of Joe Biden. Therefore, some KGs store time-aware facts or events as the quadruple (s, p, o, t), where t is the timestamp. Such KGs are referred as Temporal Knowledge Graphs (TKGs), which mainly include YAGO3 (Mahdisoltani et al., 2014), GDELT (Leetaru and Schrodt, 2013), Wikidata (Erxleben et al., 2014) and ICEWS (Lautenschlager et al., 2015).

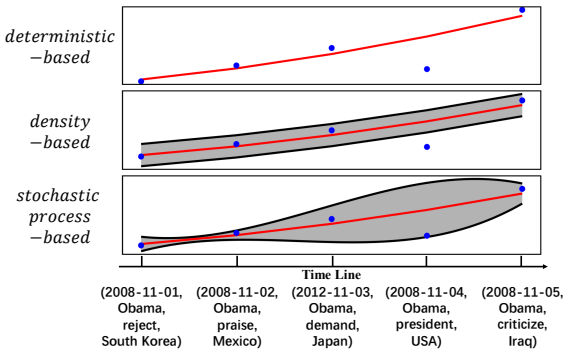

Figure 1: Illustration of three types of embedding-based TKGC methods which share the same embedding position function. Blue points denote the ideal position for the embeddings of entity $Obama$, red lines denote the embedding evolution function estimated by three types of methods, and shade areas denote the uncertainty estimated by density-based and stochastic process-based methods.

Temporal Knowledge Graph Completion (TKGC), aiming at inferring the missing edges based on known facts, is a fundamental task for the incomplete real-world TKGs. A large class methods perform TKGC by mapping nodes and edges in TKGs into high-dimensional latent feature spaces while preserving the semantic and structural

*Corresponding author.

Proceedings of the 29th International Conference on Computational Linguistics, pages 4697–4706
October 12–17, 2022.

information as much as possible. In recent years, extensive research have been conducted on embedding-based TKGC with notable process, which is also known as temporal knowledge graph embeddings (TKGEs). Early works extend the translation-based Knowledge Graph Embedding (KGE) approaches by mapping each timestamp into a specific time embedding (Leblay and Chekol, 2018; Dasgupta et al., 2018). Obviously, such approaches cannot deal with the unseen timestamps. To tackle this problem, some researchers model the entity and relation embeddings in TKGE as the continuous functions of time (García-Durán et al., 2018; Goel et al., 2020), which can be categorised as deterministic-based approaches. Recently, some researchers model TKGE as multivariate Gaussian density embeddings (Xu et al., 2019), which can be considered as density-based methods.

Although the notable progresses have been made, most of existing TKGE methods still suffer from the following two disadvantages. (1) Most of them are not flexible enough, the entity and relation embeddings are usually learned as the deterministic function of time, which is good at capturing the global trend while failing to model the surging local fluctuation. For example, as shown in Figure 1, the semantic meaning of $Obama$ should have a violent fluctuation on $2008-11-04$ as he won the presidential election, which is hard to model solely based on the deterministic function of time. (2) Most of them are not expressive enough, the temporal uncertainties of entity and relation embeddings are often ignored or under-fitted. As shown in Figure 1, existing deterministic-based methods often ignore the uncertainties while density-based methods learn embeddings with stationary uncertainties.

The above disadvantages could be naturally tackled by the stochastic process-based method. On the one hand, a stochastic process-based method is flexible to deal with the local fluctuations by modeling the correlations of embeddings at neighboured timestamps. On the other hand, a stochastic process-based method is expressive to model the dynamic changes of temporal uncertainties with the covariance matrix and kernel function.

Therefore, in this paper, we propose a novel method to learn flexible and expressive temporal knowledge graph embeddings based on approximated multivariate Gaussian processes (TKGC-AGP). In specific, each entity and relation in TKGs are mapped into a specific multivariate Gaussian process. The evolution dynamics of the entities and relations are modeled using the mean function of multivariate Gaussian process. The temporal correlation for each entity/relation are captured by the kernel function. The temporal uncertainty of the entities and relations is modeled by the entity/relation-specific covariance matrix. Further more, a first-order Markov assumption based algorithm is proposed to approximate the likelihood of multivariate Gaussian process. To investigate the effectiveness of the proposed approach, extensive experiments have been conducted on two large-scale TKG datasets. Experimental results show the effectiveness of the proposed approach compared to various competitive baselines.

In general, our contributions are listed as follows.

- A novel temporal knowledge graph embedding approach based on multivariate Gaussian process, TKGC-AGP, is proposed. Both the correlations of entities and relations over time and the temporal uncertainties of the entities and relations are modeled. To our best knowledge, we are the first one to utilize multivariate Gaussian process in TKGC.
- A novel first-order Markov assumption based algorithm is proposed to approximate the likelihood of multivariate Gaussian process.
- Experimental results show that TKGC-AGP outperforms several competitive baselines on two TKG datasets.

2 Related Work

Our work is mainly related to two lines of research, described as follows.

2.1 Temporal Knowledge Graph Completion

Temporal knowledge graph completion has been an attractive research topic in recent years. Works have been done with notable progress. Leblay and Chekol (2018) proposed TTransE, which extended the translation-based knowledge graph embedding methods to temporal knowledge graph by mapping the time information into low-dimensional vector space. Similar to TTransE, Dasgupta et al. (2018) proposed HyTE by incorporating the time information by assigning each timestamp with a temporal hyperplane. García-Durán et al. (2018)

proposed TA-TransE and TA-DistMult to learn the time-aware relation embedding by concatenating relation with time information as the input of a recurrent neural network. Xu et al. (2019) proposed ATiSE to represent the entity and relation in TKG as additive time series with Gaussian white noise to capture the temporal uncertainty. Goel et al. (2020) introduced the diachronic embedding method to model the evolution of entities along with time. Lacroix et al. (2020) presented an extension of ComplEx by introducing new regularization schemes to control the evaluation rate of embeddings. Xu et al. (2020) defined the temporal evolution of entities as the rotation in the complex vector space to deal with the symmetric and asymmetric relation simultaneously.

However, all aforementioned methods focus on modeling the evolution of entities and relations over time, ignoring the local correlations within them. To our best knowledge, we are the first one to consider model the correlation of entities and relations over long and short term in TKGs.

2.2 Probabilistic Representation Learning

Probabilistic embeddings have been extensively explored in many natural language processing tasks. Vilnis and McCallum (2015) introduced Gaussian embedding into the word representation learning task to tackle polysemy with the variance of Gaussian distributions. Brazinskas et al. (2018) further explored to learn the context-specific Gaussian word embeddings with a Bayesian learning framework. Athiwaratkun and Wilson (2018) proposed to learn the entailment relationships in the visual-semantic hierarchy with the Gaussian density order embeddings. Beyond probabilistic distribution, stochastic processes have also been considered. For example, Bamler and Mandt (2017) proposed a recursive stochastic process to model the dynamic changes of word semantics over time. To model uncertainty in KG, He et al. (2015) first employed Gaussian distribution to represent entities and relations in KG. Xiao et al. (2016) proposed a generative KGE method with a Bayesian non-parametric framework to generate Gaussian embedding and address the polysemy of relations. Other distributions such as Beta distribution have also been explored. Ren and Leskovec (2020) presented to model the first-order logic queries with Beta distributions by translating the logic operators with operations on Beta distribution.

However, all the aforementioned methods focus on specific-designed probabilistic distributions or stochastic processes. To our best knowledge, we are the first one to explicitly learn representations based on nontrivial multivariate Gaussian processes.

3 Method

In this section, we will discuss the details about the proposed method. We will start from the background knowledge of multivariate Gaussian process. Then we will talk about how to construct the entity and relation embeddings based on multivariate Gaussian process. Finally, the training and inference process of the proposed approach is explained.

3.1 Multivariate Gaussian Process

In this subsection, we will introduce the definition of multivariate Gaussian process and some common properties. We will start from matrix Gaussian distribution, which is the base for defining a multivariate Gaussian process.

Definition 1 (Matrix Gaussian Distribution)**.** A random matrix $\mathbf{X} \in \mathbb{R}^{n\times p}$ is matrix Gaussian distribution with location parameter $\mathbf{M} \in \mathbb{R}^{n\times p}$ and scale parameters $\mathbf{U} \in \mathbb{R}^{n\times n}$ and $\mathbf{V} \in \mathbb{R}^{p\times p}$ if and only if

$$vec(\mathbf{X}) \sim \mathcal{N}_{np}(vec(\mathbf{M}), \mathbf{V} \otimes \mathbf{U}) \tag{1}$$

where $\mathcal{N}_{np}(\mu, \Sigma)$ denotes multivariate Gaussian distribution on $\mathbb{R}^{n\times p}$ with mean vector μ and covariance matrix Σ, $vec(\mathbf{X})$ denotes the vectorization of $\mathbf{X}$ and $\otimes$ denotes the Kronecker product. In this case, we denote

$$\mathbf{X} \sim \mathcal{MN}_{n\times p}(\mathbf{M}, \mathbf{U}, \mathbf{V}) \tag{2}$$

With definition of matrix Gaussian distribution, we can define the multivariate Gaussian process.

Definition 2 (Multivariate Gaussian Process (MGP))**.** $\mathbf{f}$ is a multivariate Gaussian process on $\mathbb{R}^p$ with vector-valued mean function $\boldsymbol{\mu} = \{\mu_j\}_{i=1}^d : \mathbb{R}^p \Rightarrow \mathbb{R}^d$, kernel $k : \mathbb{R}^p \times \mathbb{R}^p \Rightarrow \mathbb{R}$ and positive semi-definite covariance matrix $\Omega \in \mathbb{R}^{d\times d}$ if and only if any finite collection of variables have a joint matrix Gaussian distribution,

$$[f(x_1), ..., f(x_n)] \sim \mathcal{MN}_{d\times n}(\mathbf{M}, \Omega, \Sigma) \tag{3}$$

where $\mathbf{M} \in \mathbb{R}^{d\times n}$ with $M_{ij} = \mu_j(x_i)$ and $\Sigma \in \mathbb{R}^{n\times n}$ with $\Sigma_{ij} = k(x_i, x_j)$. In this case, we denote

$$\mathbf{f} \sim \mathcal{MGP}(\boldsymbol{\mu}, k, \Omega) \tag{4}$$

3.2 Temporal Knowledge Graph Embedding based on Multivariate Gaussian Process

In this subsection, we describe how to construct the entity and relation embeddings based on MGP. Without loss of generality, we can denote a temporal knowledge graph as $\mathcal{G} = \{\mathcal{E}, \mathcal{R}, \mathcal{T}\}$, where $\mathcal{E}$, $\mathcal{R}$ and $\mathcal{T}$ are the set of entities, relations and timestamps respectively. Given a quadruplet (e_s, r_p, e_o, t) from $\mathcal{E} \times \mathcal{R} \times \mathcal{E} \times \mathcal{T}$, the goal is to learn temporal representations for $\{e_{i,t}|e_i \in \mathcal{E}\}$ and $\{r_{j,t}|r_j \in \mathcal{R}\}$ and a score function $f : \mathcal{E} \times \mathcal{R} \times \mathcal{E} \times \mathcal{T} \Rightarrow \mathbb{R}$ that is maximized for quadruplets in valid dataset D^+ and minimized for quadruplets in corrupted dataset D^-.

To model the temporal correlations and uncertainty simultaneous, entity or relation will be represented as a d-dim MGP on $t \in \mathbb{R}$, where d is the dimension of embeddings and t is time variable:

$$\begin{aligned} e_i(t) &\sim \mathcal{MGP}(\boldsymbol{\mu}_{e_i}(t), k_{e_i}(t,t'), \Omega_{e_i}) \\ r_j(t) &\sim \mathcal{MGP}(\boldsymbol{\mu}_{r_j}(t), k_{r_j}(t,t'), \Omega_{r_j}) \end{aligned} \tag{5}$$

From Definition 2, we know that an MGP can be fully specified by its mean function, kernel function and covariance matrix. We can define a MGP-based entity/relation embeddings by specifying those three components.

For the mean function, which controls the location and evolution trend of the embedding, we define it as a second order function of time variable t to make it more flexible:

$$\begin{aligned} \boldsymbol{\mu}_{e_i}(t) &= b_{e_i} + \alpha_{e_i}\phi_{e_i}t + \beta_{e_i}\psi_{e_i}t^2 \\ \boldsymbol{\mu}_{r_j}(t) &= b_{r_j} + \alpha_{r_j}\phi_{r_j}t + \beta_{e_i}\psi_{r_j}t^2 \end{aligned} \tag{6}$$

where $b_{e_i}, b_{r_j} \in \mathbb{R}^d$ are time-irrelevant bias vectors, $\alpha_{e_i}, \alpha_{r_j} \in \mathbb{R}$ are scalar first-order evaluation rates, $\phi_{e_i}, \phi_{r_j} \in \mathbb{R}^d$ are first-order evaluation direction vectors, $\beta_{e_i}, \beta_{r_j} \in \mathbb{R}$ are scalar second-order evaluation rates and $\psi_{e_i}, \psi_{r_j} \in \mathbb{R}^d$ are second-order evaluation direction vectors.

For the kernel function, it controls the correlation of the embeddings between different timestamps. The common choices of kernel functions for GP are various, such as white noise kernel, exponential quadratic kernel, rational quadratic kernel, etc. Here we assume that the correlations in TKG are mainly smooth and short-term, so we choose exponential quadratic kernel as the kernel function of TKGC-AGP. Formally we define:

$$\begin{aligned} k_{e_i}(t,t') &= \sigma_{e_i}^2 \exp(-\frac{\|t-t'\|^2}{2l_{e_i}^2}) \\ k_{r_j}(t,t') &= \sigma_{r_j}^2 \exp(-\frac{\|t-t'\|^2}{2l_{r_j}^2}) \end{aligned} \tag{7}$$

For the covariance matrixes that reflect randomness of entity (relation) in the real world, we set them as time-irrelevant diagonal matrixes for simplification:

$$\begin{aligned} \Omega_{e_i} &= diag(\omega_{e_i}) \\ \Omega_{r_j} &= diag(\omega_{r_j}) \end{aligned} \tag{8}$$

where $\omega_{e_i}, \omega_{e_i} \in \mathbb{R}^d$ denote the diagonal vectors of covariance matrixes, $diag(x)$ means making matrix with x as diagonal.

Given a quadruple q=(e_s, r_p, e_o, t), a translation-based score function is employed to measure the validity:

$$f(q) = f(e_s, r_p, e_o, t) = D_B(e_{s,t} - e_{o,t}, r_{p,t}) \tag{9}$$

where $D_B(d, d') \in \mathbb{R}$ is the Bhattacharyya distance between distribution d and d', $e_{s,t}, e_{o,t}, r_{p,t} \in \mathbb{R}^d$ are multivariate Gaussian distribution embeddings specific for timestamp t generated by the corresponding MGP:

$$\begin{aligned} e_{i,t} &= \mathcal{N}_d(\boldsymbol{\mu}_{e_i}(t), \Omega_{e_i}) \\ r_{j,t} &= \mathcal{N}_d(\boldsymbol{\mu}_{r_j}(t), \Omega_{r_j}) \end{aligned} \tag{10}$$

3.3 Approximation of MGPs and Training

In this subsection, we will describe the learning process of TKGC-AGP. To improve the robustness of training, it is common to train the model based on the valid dataset D^+ as well as a corrupted dataset D^- (Leblay and Chekol, 2018; Dasgupta et al., 2018). Following (Xu et al., 2019), a valid quadruple (s, p, o, t) is randomly corrupted by replacing the subject or object with a sampled entity from $\mathcal{E}$ to construct the corrupted dataset D^-. A common learning objective of probabilistic method is to maximize the joint likelihood of the data and the parameters, which is

$$\begin{aligned} &p(D^+, D^-, E, R) \\ &= p(D^+, D^-|E, R) \cdot p(E) \cdot p(R) \\ &= \prod_{t\in T} \prod_{q\in D_t^+} \prod_{q'\in D_t^-} \sigma(\gamma - f(q))\sigma(-\gamma + f(q')) \\ &\quad \prod_{e_i\in E} p(e_i) \prod_{r_i\in R} p(r_j) \end{aligned} \tag{11}$$

where E, R are the set of entity and relation embeddings respectively, $\sigma(\cdot)$ is Sigmoid function and γ is margin parameter. However, it is intractable to calculate the joint distribution $p(e_i)$ or $p(r_j)$ because they have a joint matrix Gaussian distribution over all possible timestamps t. That is,

$$\begin{aligned} p(e_i) &= p(e_{i,1}, e_{i,2}, ..., e_{i,t}) \\ p(r_j) &= p(r_{j,1}, r_{j,2}, ..., r_{j,t}) \end{aligned} \tag{12}$$

To approximate these joint distributions, we take a first-order Markov assumption that the current state of one entity or relation embedding is only depended on the last state of it. That is,

$$p(e_{i,t}|e_{i,t-1}, e_{i,t-2}, ..., e_{i,1}) = p(e_{i,t}|e_{i,t-1}) \tag{13}$$

To approximate the joint distributions of the entity or relation at adjacent timestamps $p(e_{i,t}, e_{i,t+1})$ or $p(r_{j,t}, r_{j,t+1})$, we further approximate the effect of this likelihood with a l_2 norm between kernel function $k_{e_i}(t, t+1)$ or $k_{r_j}(t, t+1)$ and the distances between their embeddings at the adjacent timestamps $D_B(e_{i,t}, e_{i,t+1})$ or $D_B(r_{j,t}, r_{j,t+1})$.

Then the learning objective can be transformed in minimizing the approximated negative log likelihood, which could be decomposed into four parts,

$$\begin{aligned} l_1 &= \sum_{t\in T} \sum_{q\in D_t^+} -\log\sigma(\gamma - f(q)) \\ l_2 &= \sum_{t\in T} \sum_{q'\in D_t^-} -\log\sigma(-\gamma + f(q')) \\ l_3 &= \sum_{t\in T} \sum_{e_i\in E} \|D_B(e_{i,t}, e_{i,t+1}) - k_{e_i}(t, t+1)\|^2 \\ l_4 &= \sum_{t\in T} \sum_{r_j\in R} \left\|D_B(r_{j,t}, r_{j,t+1}) - k_{r_j}(t, t+1)\right\|^2 \end{aligned} \tag{14}$$

where l_1, l_2 are the losses for valid dataset D^+ and corrupted dataset D^-, l_3, l_4 are the approximations to the log likelihood for entity and relation embeddings. We train the model by adding those losses together.

$$l = l_1 + \lambda_2 \cdot l_2 + \lambda_3 \cdot l_3 + \lambda_4 \cdot l_4 \tag{15}$$

where $\lambda_2, \lambda_3, \lambda_4$ are weights for losses.

To align the embedding across time, inspired by (Kumar et al., 2019), we further propose a time-batch training strategy, which takes the data between timestamp t and timestamp $t + L$ as a batch, where L is the length of time window. The detailed learning algorithm of TKGC-AGP is shown in Algorithm 1.

Algorithm 1 Training of TKGC-AGP

Input: the entity set $\mathcal{E}$, the relation set $\mathcal{R}$, the arranged timestamp set $\mathcal{T}$, the valid dataset D^+, the negative sample rate η, the number of epoch n, the margin γ, the embedding dimension d, the length of time window L.

Output: the parameters for TKGC-AGP $P = \{b, \alpha, \phi, \beta, \psi, \sigma, l, \omega\}$.

```
randomly initialize P
for i = 1, ..., n do
    for t' ∈ T do
        D_t^+ ← {q = (e_s, r_p, e_o, t) | t ∈ [t', t' + L], q ∈ D^+}
        for (e_s, r_p, e_o, t) ∈ D_t^+ do
            D_t^- = +{(e_s^k, r_p, e_o^k, t)}_{k=1,...,η}
        end for
        Update P = {b, α, φ, β, ψ, σ, l, ω} w.r.t. Loss l
    end for
end for
```

3.4 Complexity Analysis

Though the structure of TKGC-AGP is complex, as shown in Table 1, the space complexity and time complexity of TKGC-AGP remains the same as that of most of static and temporal KGE methods. For the space complexity, the parameter space of TKGC-AGP comprises $P = \{b, \alpha, \phi, \beta, \psi, \sigma, l, \omega\}$. So the total number of parameters of TKGC-AGP is $8 \times (|\mathcal{E}| + |\mathcal{R}|) \times d$. Since the length of time window is constant, the time complexity is also constant with embedding dimension d.

Method	Space complexity	Time complexity
TransE	$\mathcal{O}(\|\mathcal{E}\|d + \|\mathcal{R}\|d)$	$\mathcal{O}(d)$
ComplEx	$\mathcal{O}(\|\mathcal{E}\|d + \|\mathcal{R}\|d)$	$\mathcal{O}(d)$
TTransE	$\mathcal{O}(\|\mathcal{E}\|d + \|\mathcal{R}\|d + \|\mathcal{T}\|d)$	$\mathcal{O}(d)$
DE-SimplE	$\mathcal{O}(\|\mathcal{E}\|d + \|\mathcal{R}\|d)$	$\mathcal{O}(d)$
ATiSE	$\mathcal{O}(\|\mathcal{E}\|d + \|\mathcal{R}\|d)$	$\mathcal{O}(d)$
TKGC-AGP	$\mathcal{O}(\|\mathcal{E}\|d + \|\mathcal{R}\|d)$	$\mathcal{O}(d)$

Table 1: Complexity analysis of some existing methods.

4 Experiments

In this section, we perform extensive experiments on link prediction to investigate the effectiveness of TKGC-AGP on two real-world TKG datasets compared to some state-of-the-art KGE methods and TKGC methods.

4.1 Dataset

Two standard benchmark datasets, ICEWS-14 and ICEWS05-15, for TKGC are employed for experiments, which are two subsets of the Integrated Crisis Early Warning System (ICEWS) dataset (Lautenschlager et al., 2015). ICEWS-14 includes events happened in 2014 while ICEWS05-15 includes events happened between 2005 to 2015. The fact stored in ICEWS follows the form (s, p, o, t) with specific time point, such as *(Barack Obama, investigate, Iraq, 2008-07-21)*. Following (Xu et al., 2019), we employ the filtered version of ICEWS-14 and ICEWS05-15. The detailed statistics of two datasets are listed in Table 2.

Dataset	ICEWS-14	ICEWS05-15
# Entities	6,869	10,094
# Relations	230	251
# Timestamps	365	4,017
# Training	72,826	368,962
# Validation	8,941	46,275
# Test	8,963	46,092

Table 2: Dataset statistics.

4.2 Baselines

We compare TKGC-AGP with the following baselines:

- **TransE** (Bordes et al., 2013): static method that considers relation as a translation between entities in the embedding space.
- **DistMult** (Yang et al., 2015): static method that deals with the problem of symmetric relation with a bilinear objective function.
- **CompleEx** (Trouillon et al., 2016): static method that maps entities and relations into complex space with tensor factorization technique.
- **RotatE** (Sun et al., 2019): static method that regards relation as the rotation in the complex space.
- **TTransE** (Leblay and Chekol, 2018): temporal method that extend TransE to TKG by mapping each timestamp as specific embedding.
- **HyTE** (Dasgupta et al., 2018): temporal method that extend TransH (Wang et al., 2014) to TKG by learning time-specific hpyerplanes.
- **TA-TransE** (García-Durán et al., 2018): temporal method that employs a LSTM to encode the time information into relation representations.
- **DE-SimplE** (Goel et al., 2020): temporal method that represents entities with diachronic embeddings.
- **ATiSE** (Xu et al., 2019): temporal method that maps entities and relations as additive time series with Gaussian white noise.

4.3 Evaluation Metrics

In the link prediction experiment, following the previous literature (Goel et al., 2020), for each valid quadruple (e_s, r_p, e_o, t) in validation and test set, we generate query by masking the subject entity or object entity of it. Then we rank all the possible entities by filling the missing entity with candidate entity. Followed by previous work(Xu et al., 2019), we employ two kinds of metrics to evaluate the performance of all the methods, the **Mean Reciprocal Rank** (MRR), which is the average of reciprocal of the rank of golden entity and **Hit@K**, which is the frequency that the rank of golden entity is no greater than K.

4.4 Implementation Details

TKGC-AGP is implemented with PyTorch (Paszke et al., 2019). Part of results are taken from (Goel et al., 2020; Xu et al., 2019). The embeddings are trained with ADAM optimizer (Kingma and Ba, 2015) with learning rate = 0.001, maximum epoch = 1000, negative sample rate = 5, dimension of embedding = 100, length of time window = 3, margin = 1. All vector parameters are normalized to have unit l-2 norm.

4.5 Link Prediction Results

The link prediction results on the two dataset are shown in Table 3. It can be observed: 1) Some static methods outperform the temporal methods. For example, the performances of ComplEx and DistMulti are generally better than those of TTransE and HyTE. Capturing the basic structure of knowledge graph is still important for TKGC.

	ICEWS14				ICEWS05-15			
Metrics	MRR	Hit@1	Hit@3	Hit@10	MRR	Hit@1	Hit@3	Hit@10
TransE	0.280	0.094	-	0.637	0.294	0.090	-	0.663
DistMult	0.439	0.323	-	0.672	0.456	0.337	-	0.691
ComplEx	0.467	0.347	0.527	0.716	0.481	0.362	0.535	0.729
TTransE	0.255	0.074	-	0.601	0.271	0.084	-	0.616
HyTE	0.297	0.108	0.416	0.655	0.316	0.116	0.445	0.681
TA-TransE	0.275	0.095	-	0.625	0.299	0.096	-	0.668
TA-DistMult	0.477	0.363	-	0.686	0.474	0.346	-	0.728
DE-TransE	0.326	0.124	0.467	0.686	0.314	0.108	0.453	0.685
DE-SimplE	0.526	0.418	0.592	0.725	0.513	0.392	0.578	0.748
ATiSE	0.550	0.436	0.629	**0.750**	0.519	0.378	0.606	0.794
TKGC-AGP	**0.561**	**0.458**	**0.631**	0.738	**0.532**	**0.398**	**0.621**	**0.797**

Table 3: Link prediction results on ICEWS14 and ICEWS05-15 datasets. Bold values indicate the best-performing models under corresponding settings.

2) With temporal information, some KGE methods perform better on TKG. For example, DE-SimplE and TA-DistMult generally outperform other methods except ATiSE and TKGC-AGP. 3) The time series or stochastic process based methods achieve the best performance. TKGC-AGP outperforms ATiSE on all meatrics except Hit@10 for ICEWS14. The improvement is mainly attributed to the correlation modeled by the kernel function of MGP, although ATiSE employs a Gaussian white noise component to model temporal uncertainty.

4.6 Ablation Study

In this section, we analyze how the hyperparameters and the key components of TKGC-AGP affect the final performance. We focus on embedding dimension, length of time window and kernel function.

4.6.1 Effect of Embedding Dimension

The embedding dimension is one of the important hyperparameters for the representation learning methods. On the one hand, too small embedding dimension prevents methods to encode sufficient information. On the other hand, too large embedding dimension leads to the time and computation overhead, which is critical for TKGC with over ten thousands parameters. In this part, we evaluate the performance of TKGC-AGP under different embedding dimension $(50, 100, 200, 300, 400, 500)$ on ICEWS14 dataset. The results are shown in Figure 2.

It can observed that in general the performance of TKGC-AGP on ICEWS14 dataset is increasing at first and then decreasing with peak at 100 dim.

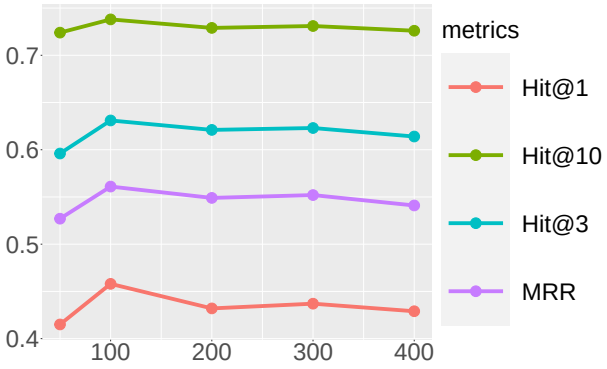

Figure 2: The performances of TKGC-AGP on ICEWS14 dataset under different embedding dimension setting.

The reason may be that when the dimension is too small, the embeddings cannot encode sufficient information while when the dimension is too large, the embeddings become too sparse to learn from the dataset. It should be pointed out that the optimal dimension of TKGC-AGP is generally smaller than that is other KGE or TKGC methods. It could be attributed to the complex structure of TKGC-AGP where there is no need for large dimension to encode sufficient information.

4.6.2 Effect of Length of Time Window

As described in Algorithm 1, the length of time window L is important for training TKGC-AGP. On the one hand, too small L will result in underfitted correlation. On the other hand, too large L will lead to more computational overhead and have the risk of over-fitting. To investigate the effect of the length of time window L on the performance of TKGC-AGP, experiments were conduct

Metrics	MRR	Hit@1	Hit@3	Hit@10
$L = 1$	0.551	0.439	0.627	0.733
$L = 2$	0.555	0.443	0.629	0.734
$L = 3$	**0.561**	**0.458**	**0.631**	**0.738**
$L = 4$	0.531	0.420	0.599	0.724
$L = 5$	0.526	0.415	0.595	0.727
$L = 10$	0.458	0.357	0.529	0.676

Table 4: The performances of TKGC-AGP with different lengths of time window on ICEWS14 dataset. Bold values indicate best-performing models under corresponding settings.

on ICEWS14 dataset with the length of time window L among $(1, 2, 3, 4, 5, 10)$. The results are shown in Table 4.

From the results, we can observe that in general the performance of TKGC-AGP on ICEWS14 dataset is firstly increasing as the length of time window increases and then quickly saturates at $L = 3$. It might be explained by the processing of TKGC-AGP from under-fitting to over-fitting. Especially when $L = 10$, the performance of TKGC-AGP is affected seriously because of over-fitting.

4.6.3 Effect of Kernel Function

As described in Section 3.1, the kernel function is an important component of MGP that controls the correlation across the index set. To investigate the effect of kernel function on the performance of TKGC-AGP, we perform experiments on ICEWS14 dataset with the following kernel functions besides exponential quadratic kernel. The results are shown in Table 5.

- **White noise kernel**, which means any two points from MGP are uncorrelated.

$$k(t, t') = \sigma^2 \mathbf{I}_n \tag{16}$$

- **Exponential quadratic kernel**, a smooth correlation decreasing with the distance between two points.

$$k(t, t') = \sigma^2 exp(-\frac{\|t - t'\|^2}{2l^2}) \tag{17}$$

- **Rational quadratic kernel**, which is similar to the exponential quadratic, when $\alpha \to \infty$, the rational quadratic kernel converges into the exponential quadratic kernel.

$$k(t, t') = \sigma^2 (1 + \frac{\|t - t'\|^2}{2\alpha l^2})^{-\alpha} \tag{18}$$

Metrics	MRR	Hit@1	Hit@3	Hit@10
TKGC-AGP-E	**0.561**	**0.458**	**0.631**	**0.738**
TKGC-AGP-W	0.548	0.435	0.626	0.732
TKGC-AGP-R	0.559	0.460	0.627	0.735
TKGC-AGP-P	0.479	0.370	0.532	0.685

Table 5: The performances of TKGC-AGP on ICEWS14 dataset with different kernel functions. Bold values indicate best-performing models under corresponding settings. -E, -W, -R, -P denotes TKGC-AGP with exponential quadratic kernel, white noise kernel, rational quadratic kernel and periodic kernel respectively.

- **Periodic kernel**, which allows to model periodic functions, where p denotes the period.

$$k(t, t') = \sigma^2 exp(-\frac{2}{l^2} sin^2(\pi \frac{\|t - t'\|}{p})) \tag{19}$$

It can be observed that TKGC-AGP with exponential quadratic kernel achieves the best performance. It should be pointed out that ATiSE (Xu et al., 2019) can be considered as a MGP-based method with white noise kernel. With white noise kernel, TKGC-AGP is similar to ATiSE. Therefore their performances are also very similar. Since rational quadratic kernel is very similar to exponential quadratic kernel, the performances of TKGC-AGP-E and TKGC-AGP-R are also very similar. For periodic kernel, it has the worst performance, we attribute this to little periodic pattern in the experiment dataset.

5 Conclusion

In this paper, we proposed TKGC-AGP, a novel temporal knowledge graph completion method based on approximated Gaussian process embeddings. With the flexibility and capacity, we can naturally model the global trends of entity and relation embeddings as well as the surging local fluctuations. Moreover, the temporal uncertainties can be also naturally modeled with the kernel function and corvariance matrix of MGP. To training TKGC-AGP, we employ the fisrt-order Markov assumption to approximate the joint distribution of MGP as well as a time-batch-based training strategy to align the embeddings across the time. The experimental results demonstrate that the proposed method outperform various static and temporal KGE baselines. Further work could be done by taking a Bayesian perspective to learn the proposed method.

6 Acknowledgement

The authors would like to thank the anonymous reviewers for the insightful comments. This work was funded by the National Natural Science Foundation of China (62176053).

References

Ben Athiwaratkun and Andrew Gordon Wilson. 2018. Hierarchical density order embeddings. In *6th International Conference on Learning Representations, ICLR 2018, Vancouver, BC, Canada, April 30 - May 3, 2018, Conference Track Proceedings*. OpenReview.net.

Sören Auer, Christian Bizer, Georgi Kobilarov, Jens Lehmann, Richard Cyganiak, and Zachary Ives. 2007. Dbpedia: A nucleus for a web of open data. In *The semantic web*, pages 722–735. Springer.

Robert Bamler and Stephan Mandt. 2017. Dynamic word embeddings. In *Proceedings of the 34th International Conference on Machine Learning, ICML 2017, Sydney, NSW, Australia, 6-11 August 2017*, volume 70 of *Proceedings of Machine Learning Research*, pages 380–389. PMLR.

Kurt Bollacker, Colin Evans, Praveen Paritosh, Tim Sturge, and Jamie Taylor. 2008. Freebase: a collaboratively created graph database for structuring human knowledge. In *Proceedings of the 2008 ACM SIGMOD international conference on Management of data*, pages 1247–1250.

Antoine Bordes, Nicolas Usunier, Alberto Garcia-Duran, Jason Weston, and Oksana Yakhnenko. 2013. Translating embeddings for modeling multi-relational data. *Advances in neural information processing systems*, 26.

Arthur Brazinskas, Serhii Havrylov, and Ivan Titov. 2018. Embedding words as distributions with a bayesian skip-gram model. In *Proceedings of the 27th International Conference on Computational Linguistics, COLING 2018, Santa Fe, New Mexico, USA, August 20-26, 2018*, pages 1775–1789. Association for Computational Linguistics.

Andrew Carlson, Justin Betteridge, Bryan Kisiel, Burr Settles, Estevam R Hruschka, and Tom M Mitchell. 2010. Toward an architecture for never-ending language learning. In *Twenty-Fourth AAAI conference on artificial intelligence*.

Shib Sankar Dasgupta, Swayambhu Nath Ray, and Partha Talukdar. 2018. Hyte: Hyperplane-based temporally aware knowledge graph embedding. In *Proceedings of the 2018 conference on empirical methods in natural language processing*, pages 2001–2011.

Fredo Erxleben, Michael Günther, Markus Krötzsch, Julian Mendez, and Denny Vrandečić. 2014. Introducing wikidata to the linked data web. In *International semantic web conference*, pages 50–65. Springer.

Alberto García-Durán, Sebastijan Dumančić, and Mathias Niepert. 2018. Learning sequence encoders for temporal knowledge graph completion. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 4816–4821, Brussels, Belgium. Association for Computational Linguistics.

Alberto García-Durán, Sebastijan Dumancic, and Mathias Niepert. 2018. Learning sequence encoders for temporal knowledge graph completion. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing, Brussels, Belgium, October 31 - November 4, 2018*, pages 4816–4821. Association for Computational Linguistics.

Rishab Goel, Seyed Mehran Kazemi, Marcus Brubaker, and Pascal Poupart. 2020. Diachronic embedding for temporal knowledge graph completion. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pages 3988–3995.

Yanchao Hao, Yuanzhe Zhang, Kang Liu, Shizhu He, Zhanyi Liu, Hua Wu, and Jun Zhao. 2017. An end-to-end model for question answering over knowledge base with cross-attention combining global knowledge. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 221–231, Vancouver, Canada. Association for Computational Linguistics.

Shizhu He, Kang Liu, Guoliang Ji, and Jun Zhao. 2015. Learning to represent knowledge graphs with gaussian embedding. In *Proceedings of the 24th ACM International Conference on Information and Knowledge Management, CIKM 2015, Melbourne, VIC, Australia, October 19 - 23, 2015*, pages 623–632. ACM.

Diederik P. Kingma and Jimmy Ba. 2015. Adam: A method for stochastic optimization. In *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*.

Srijan Kumar, Xikun Zhang, and Jure Leskovec. 2019. Predicting dynamic embedding trajectory in temporal interaction networks. In *Proceedings of the 25th ACM SIGKDD international conference on knowledge discovery & data mining*, pages 1269–1278.

Timothée Lacroix, Guillaume Obozinski, and Nicolas Usunier. 2020. Tensor decompositions for temporal knowledge base completion. In *8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, April 26-30, 2020*. OpenReview.net.

Jennifer Lautenschlager, Steve Shellman, and Michael Ward. 2015. ICEWS Event Aggregations.

Julien Leblay and Melisachew Wudage Chekol. 2018. Deriving validity time in knowledge graph. In *Companion Proceedings of the The Web Conference 2018*, pages 1771–1776.

Kalev Leetaru and Philip A Schrodt. 2013. Gdelt: Global data on events, location, and tone, 1979–2012. In *ISA annual convention*, volume 2, pages 1–49. Citeseer.

Farzaneh Mahdisoltani, Joanna Biega, and Fabian Suchanek. 2014. Yago3: A knowledge base from multilingual wikipedias. In *7th biennial conference on innovative data systems research*. CIDR Conference.

Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Kopf, Edward Yang, Zachary DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. 2019. Pytorch: An imperative style, high-performance deep learning library. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett, editors, *Advances in Neural Information Processing Systems 32*, pages 8024–8035. Curran Associates, Inc.

Hongyu Ren and Jure Leskovec. 2020. Beta embeddings for multi-hop logical reasoning in knowledge graphs. In *Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020, December 6-12, 2020, virtual*.

Fabian M Suchanek, Gjergji Kasneci, and Gerhard Weikum. 2007. Yago: a core of semantic knowledge. In *Proceedings of the 16th international conference on World Wide Web*, pages 697–706.

Zhiqing Sun, Zhi-Hong Deng, Jian-Yun Nie, and Jian Tang. 2019. Rotate: Knowledge graph embedding by relational rotation in complex space. In *7th International Conference on Learning Representations, ICLR 2019, New Orleans, LA, USA, May 6-9, 2019*. OpenReview.net.

Théo Trouillon, Johannes Welbl, Sebastian Riedel, Éric Gaussier, and Guillaume Bouchard. 2016. Complex embeddings for simple link prediction. In *Proceedings of the 33nd International Conference on Machine Learning, ICML 2016, New York City, NY, USA, June 19-24, 2016*, volume 48 of *JMLR Workshop and Conference Proceedings*, pages 2071–2080. JMLR.org.

Luke Vilnis and Andrew McCallum. 2015. Word representations via gaussian embedding. In *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*.

Zhen Wang, Jianwen Zhang, Jianlin Feng, and Zheng Chen. 2014. Knowledge graph embedding by translating on hyperplanes. In *Proceedings of the Twenty-Eighth AAAI Conference on Artificial Intelligence, July 27 -31, 2014, Québec City, Québec, Canada*, pages 1112–1119. AAAI Press.

Han Xiao, Minlie Huang, and Xiaoyan Zhu. 2016. TransG : A generative model for knowledge graph embedding. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 2316–2325, Berlin, Germany. Association for Computational Linguistics.

Chengjin Xu, Mojtaba Nayyeri, Fouad Alkhoury, Jens Lehmann, and Hamed Shariat Yazdi. 2019. Temporal knowledge graph embedding model based on additive time series decomposition. *CoRR*, abs/1911.07893.

Chengjin Xu, Mojtaba Nayyeri, Fouad Alkhoury, Hamed Shariat Yazdi, and Jens Lehmann. 2020. Tero: A time-aware knowledge graph embedding via temporal rotation. In *Proceedings of the 28th International Conference on Computational Linguistics, COLING 2020, Barcelona, Spain (Online), December 8-13, 2020*, pages 1583–1593. International Committee on Computational Linguistics.

Bishan Yang, Wen-tau Yih, Xiaodong He, Jianfeng Gao, and Li Deng. 2015. Embedding entities and relations for learning and inference in knowledge bases. In *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*.

Linhai Zhang, Chao Lin, Deyu Zhou, Yulan He, and Meng Zhang. 2021. A bayesian end-to-end model with estimated uncertainties for simple question answering over knowledge bases. *Computer Speech & Language*, 66:101167.

CILDA: Contrastive Data Augmentation using Intermediate Layer Knowledge Distillation

Md Akmal Haidar[1]* **Mehdi Rezagholizadeh**[1] **Abbas Ghaddar**[1] **Khalil Bibi**[1]
Philippe Langlais[2] **Pascal Poupart**[3]
[1]Huawei Noah's Ark Lab
[2] RALI/DIRO, Université de Montréal, Canada
[3] David R. Cheriton School of Computer Science, University of Waterloo
{mehdi.rezagholizadeh,abbas.ghaddar}@huawei.com
felipe@iro.umontreal.ca, ppoupart@uwaterloo.ca

Abstract

Knowledge distillation (KD) is an efficient framework for compressing large-scale pre-trained language models. Recent years have seen a surge of research aiming to improve KD by leveraging **C**ontrastive Learning, **I**ntermediate **L**ayer Distillation, **D**ata Augmentation, and **A**dversarial Training. In this work, we propose a learning based data augmentation technique tailored for knowledge distillation, called CILDA. To the best of our knowledge, this is the first time that intermediate layer representations of the main task are used in improving the quality of augmented samples. More precisely, we introduce an augmentation technique for KD based on intermediate layer matching using contrastive loss to improve masked adversarial data augmentation. CILDA outperforms existing state-of-the-art KD approaches on the GLUE benchmark, as well as in an out-of-domain evaluation.

1 Introduction

The exponentially increasing size of pre-trained large language models (Devlin et al., 2019; Liu et al., 2020; Raffel et al., 2020; Brown et al., 2020) has been a persistent concern regarding the efficiency and scalability of Natural Language Understanding (NLU) in real world applications. Knowledge Distillation (KD) (Buciluǎ et al., 2006; Hinton et al., 2014) is a technique for transferring the knowledge from a large-scale model (called teacher) to a smaller one (called student), so that the latter model can be employed on edge device (Sanh et al., 2019a; Tang et al., 2019; Mukherjee and Awadallah, 2020; Li et al., 2021). This is done by minimizing the KL divergence between the teacher and student probabilistic outputs.

Numerous techniques have been exploited recently to increase the knowledge transfer beyond logits matching. For instance, it has been found beneficial to perform distillation on the internal components (parameters) of the teacher and student, which is known as Intermediate Layer Distillation (Sun et al., 2019, 2020b; Passban et al., 2021; Wang et al., 2020a,b; Fu et al., 2021; Wu et al., 2021).

Data Augmentation has also been successful for KD (Jiao et al., 2019; Shen et al., 2020; Qu et al., 2021), as researchers have found that the student has less opportunity to acquire useful information from the teacher when limited data are available for training (Kamalloo et al., 2021, 2022; Jafari et al., 2021a). Adversarial Training was also employed in KD (Zhu et al., 2019; Rashid et al., 2020, 2021; He et al., 2021) to improve the robustness and generalization, as the student may predict inconsistent outputs with slight distortion to the data distributions (Li et al., 2021). Recently, Contrastive Learning (Gutmann and Hyvärinen, 2010; Hjelm et al., 2018; Arora et al., 2019) has been exploited for improving knowledge transfer (Tian et al., 2019), and to optimize the intermediate layer mapping scheme (Sun et al., 2020a).

Each of the aforementioned techniques has proven effective in addressing a specific challenge in KD. Yet, we are not aware of a single method that takes advantage of all of them. In this paper, we propose CILDA, a KD method that incorporate **C**ontrasting Learning, **I**ntermediate **L**ayer Distillation, **D**ata Augmentation, and **A**dversarial Training. Distilling into a 6-layer BERT model, CILDA delivers new state-of-the-art results on the GLUE benchmark (Wang et al., 2018), as well as outperforming other KD methods in out-of-domain evaluations.

* Work done while at Huawei.

Proceedings of the 29th International Conference on Computational Linguistics, pages 4707–4713
October 12–17, 2022.

2 Related Work

Many studies (Jawahar et al., 2019; Tenney et al., 2019; Kovaleva et al., 2019) have noticed that important structural linguistic information are hidden in the intermediate layers of Transformer models (Vaswani et al., 2017). Recent KD methods propose to match teacher and student: intermediate layers representations (Jiao et al., 2019; Sun et al., 2019, 2020b; Wu et al., 2020), embedding matrix (Sanh et al., 2019a), and self-attention distributions (Wang et al., 2020a,b). Other variants of KD methods have been proposed such as Annealing-KD (Jafari et al., 2021b) and Pro-KD (Rezagholizadeh et al., 2021), two stage distillation methods where a smooth and gradual training of the student is controlled by a dynamic temperature factor, followed by a simple cross entropy loss for a few epochs.

Augmented adversarial examples (Miyato et al., 2016) are label-preserving transformations in the embedding space that are used to improve generalizability of models. FreeLB (Zhu et al., 2019) is an adversarial algorithm which creates virtual adversarial examples from word embeddings, and then performs the parameter updates on these adversarial embeddings. MATE-KD (Rashid et al., 2021) is a min-max adversarial data augmentation approach for KD, where an extra *generator* model is trained to generate adversarial text by maximizing the logit output margins between the teacher and the student.

Contrastive learning is a self-supervised representation learning method (Chen et al., 2020; Qu et al., 2021; van den Oord et al., 2018) which learns the feature representation of the samples by contrasting positive and negative samples. CODIR (Sun et al., 2020a) is a contrast-enhanced diversity promoting method between teacher and student intermediate representations of data samples from the same class. MATE-KD is the most related to our solution, with one notable difference: we believe our technique is the first to deploy intermediate layers distillation with the contrastive objective in the data augmentation process.

3 CILDA

In this section, we introduce CILDA, our contrastive approach for masked adversarial text augmentation for knowledge distillation using intermediate layer matching. Inspired by (Rashid et al., 2021), we deploy a generator (e.g. BERT) which will be trained to map masked inputs, $\tilde{X}$, to augmented samples, X'. The objective of this mapping is to perturb the inputs (in their vicinity) such that their corresponding output and intermediate layer representations of the teacher and student networks diverge to their maximum. Generating such maximum divergence augmented samples aims to fill the existing major gaps in the training data. We mask input tokens with a certain pre-defined probability, p. The architecture of our model is depicted in Figure 1. Our training is comprised of two alternating steps we describe hereafter.

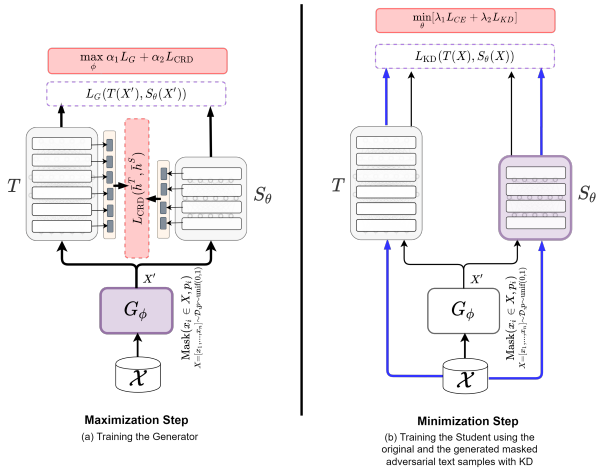

Figure 1: Illustration of maximization and minimization steps of CILDA.

Maximization Step: Generating Augmented Samples In the maximization step, the generator is trained in a way that the difference between the teacher and the student are maximized. As opposed to MATE-KD which only evaluates the divergence of the student and teacher networks based on their output, our technique takes intermediate layer matching into account as well. To the best of our knowledge this is the first time that the distance of intermediate layer representations are considered in the data-augmentation generation process. To be concise, MATE-KD only pays attention to the distance of samples in the output space, while our technique concerns the distance of samples in the input space as well. We hypothesize that to identify maximum divergence augmented samples, both input feature distances and output predictions are important. Our CILDA loss function to train

the generator can be described as:

$$L_{G_\phi} = \alpha_1 L_G + \alpha_2 L_{CRD}$$
$$L_G = KL\Big(\sigma(\frac{T(X')}{\tau_1}), \sigma(\frac{S_\theta(X')}{\tau_1})\Big) \quad (1)$$

where L_G is the KL-divergence loss between the teacher and the student logits, T and S_θ represent the teacher model and the student model with θ parameters respectively, σ is the softmax function and τ_1 is the temperature parameter that controls the softness of the output distributions, α_1 and α_2 are hyper-parameters. X' is the adversarial text output obtained by applying argmax to the generator output in the forward pass. Due to the non-differentiability issue of argmax in the backward pass, we use Gumbel-Softmax (Jang et al., 2016) at the output of the generator. More details can be found in (Rashid et al., 2021). L_{CRD} is the contrastive distillation loss that we introduced to the maximization step of MATE-KD. This contrastive loss is obtained by using the intermediate representation outputs of the teacher and the student models:

$$L_{CRD} = -\log \frac{exp(< \bar{h}_k^T, \bar{h}_k^{S_\theta} > /\tau_2)}{\sum_{j=0}^{K} exp(< \bar{h}_k^T, \bar{h}_j^{S_\theta} > /\tau_2)} \quad (2)$$

where τ_2 is the temperature parameter that controls the concentration level (Sun et al., 2020a). $\bar{h}_k^T$ and $\bar{h}_k^{S_\theta}$ are the intermediate layer representation of the teacher and student networks respectively, and $< ., . >$ is the cosine similarity between two feature vectors. k and j are indices of the samples of a mini-batch: k is the index of positive samples (i.e. the kth sample of the mini-batch is sent to both of the student and teacher networks to obtain their representations) and when $j \neq k$, we get negative samples (i.e. any other sample in the mini-batch excluding the kth sample) in a batch of K samples. The goal of this objective function is to map the student representations $\bar{h}_k^{S_\theta}$ of the positive sample k to $\bar{h}_k^T$, as well as the negative representations $\{\bar{h}_j^{S_\theta}\}_{j\neq k}^K$ far apart from $\bar{h}_k^T$.

For an arbitrary sample l in a mini-batch, the entire intermediate layer representations of the teacher and the student models (e.g. the $< CLS >$ representation of each layer of the networks) are concatenated to form $\hat{h}_l^T = [\bar{h}_{1,l}^T, \cdots, \bar{h}_{n,l}^T]$, $\hat{h}_l^{S_\theta} = [\bar{h}_{1,l}^{S_\theta}, \cdots, \bar{h}_{m,l}^{S_\theta}]$. Then these concatenated representations are further mapped into the same-size lower-dimensional spaces using linear projections $\bar{h}_l^T, \bar{h}_l^{S_\theta} \in R^u$ to calculate the distillation loss L_{CRD}. Here, n and m denote the number of intermediate layers of the teacher and the student networks respectively.

Minimization Step: Deploying Augmented Samples In the minimization step, the augmented adversarial samples produced by the generator and the training samples are used to minimize the difference between the teacher and the student. For this step, in the very general form, one can consider to match the student and teacher networks on their outputs and intermediate layer representations (e.g. using the contrastive loss) and the CE loss to match the output of the student with the labels:

$$L_{S_\theta} = \lambda_1 L_{CE} + \lambda_2 L_{KD} \quad (3)$$

where, L_{CE} describes the cross-entropy loss between the true label.

4 Experiments

4.1 Datasets and Evaluation

We experiment on 7 tasks from the GLUE benchmark (Wang et al., 2018): 2 single-sentence (CoLA and SST-2) and 5 sentence-pair (MRPC, RTE, QQP, QNLI, and MNLI) classification tasks. Following prior works, we report Pearson correlation on STS-B, Matthews correlation on CoLA, F1 score on MRPC, and use the accuracy otherwise. For out-of-domain evaluation, we report the performances on HANS (McCoy et al., 2019), SciTail (Khot et al., 2018), and IMDB using the models finetuned on MNLI, QQP, and SST-2 respectively.

4.2 Implementation Details

We use the 24-layer RoBERTa-large (Liu et al., 2020) and the 6-layer DistilRoBERTa (Sanh et al., 2019b) as the backbone for the teacher and the student models respectively. We perform hyperparameter tuning, and select best performing models using early stopping on dev sets. We use a linear transformation to map the intermediate representations into a 128-dimensional space and normalized them before computing the loss L_{CRD}. For each batch of data, we train the generator for n_G steps and the student model for $n_S = 100$ steps. We use $n_G = 20$ for CoLA, MRPC, RTE tasks and n_G=10 for the rest of the tasks. Following (Rashid et al., 2021), we set $p_{th} = 0.3$, $\alpha_1 = 1, \alpha_2 = 1, \tau_1 = 1.0, \tau_2 = 2.0$ for all of our

Model	CoLA	SST-2	MRPC	STS-B	QQP	MNLI	QNLI	RTE	Avg.
					DEV				
Teacher	68.1	96.4	91.9	92.3	91.5	90.2	94.6	86.3	88.9
Vanilla-KD	60.9	92.5	90.2	89.0	91.6	84.1	91.3	71.1	83.8
Annealing-KD	61.7	93.1	90.6	89.0	91.5	85.3	92.5	73.6	84.7
MATE-KD	65.9	94.1	91.9	90.4	91.9	85.8	**94.6**	75.0	86.2
CILDA	**67.1**	**94.7**	**92.0**	**90.5**	**92.1**	**86.8**	92.9	**76.2**	**86.5**
					TEST				
Teacher	68.6	97.1	93.0	92.4	90.2	90.7	95.5	87.9	89.4
Vanilla-KD	54.3	93.1	86.0	85.7	89.5	83.6	90.8	74.1	82.1
Annealing-KD	54.0	93.6	86.0	86.8	89.7	84.4	90.8	73.7	82.4
MATE-KD	56.0	**94.9**	90.2	88.0	89.7	85.2	92.1	75.0	83.9
CILDA	**56.2**	**94.9**	**90.5**	**89.0**	**89.9**	**86.1**	**92.5**	**77.0**	**84.5**

Table 1: DEV and TEST performances on GLUE benchmark when RoBERTa_{24} and $\text{DistillRoberta}_{6}$ are used as backbone for the teacher and student variants respectively. Bold mark describes the best results.

experiments. We set λ_1 and λ_2 to 1/3 for the original training samples. For the augmented samples, we use $\lambda_2 = 2/9, \lambda_3 = 1/9$ for all tasks. The learning rate and the batch size are tuned from the set of {1e-5, 2e-5, 4e-6} and {8, 16, 32} respectively.

4.3 Results and Analysis

Table 1 shows the performances of the teacher, baselines, and our method on the GLUE dev and test sets. We compared CILDA to the Vanilla-KD (Hinton et al., 2014) baseline, and against 2 strong recently proposed methods [1]: Annealing-KD (Jafari et al., 2021b) and MATE-KD (Rashid et al., 2021). We observe that CILDA outperforms these models on all GLUE tasks, except on QNLI dev where MATE-KD performs better and SST-2 test where CILDA is on par with MATE-KD. On average over test sets, CILDA outperforms MATE-KD and Annealing-KD by a margin of 0.6% and 2.1% respectively.

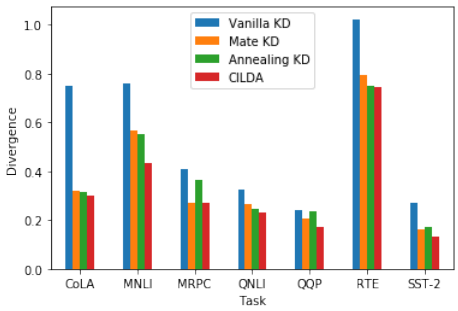

Figure 2: Divergence (lower is better) between the teacher and student logits on GLUE dev sets.

[1]We compare with these models because we have published results on GLUE leaderboard using the same teacher and student backbone models.

We investigate the logits generated by different methods to better understand why CILDA performs better. Figure 2 shows the divergence (lower is better) between the teacher and student logits on GLUE dev sets (except STS-B since it is a regression task) for 4 KD methods. Expectedly, Vanilla-KD (no enhancement) had the maximum divergence with teacher logits (which can be easily distinguished from other methods). We observe that CILDA mimic the teacher better than other methods on all tasks, which may partially explain the performance gains obtained by CILDA.

Model	HANS	PAWS	IMDB
Teacher	78.2	43.3	88.9
w/o KD	58.6	34.7	83.7
Vanilla-KD	58.9	36.5	84.0
Annealing-KD	61.2	35.8	84.6
MATE-KD	66.6	38.3	85.0
CILDA	**68.1**	**40.5**	**85.2**

Table 2: Out-of-domain performances of models trained on MNLI, QQP, SST-2 and evaluated on HANS, PAWS, and IMDB respectively.

Furthermore, we measure the robustness and generalization ability of the tested methods by evaluating them on out-of-domain test sets. Table 2 shows performances of models fine-tuned on MNLI, QQP, SST-2 and tested on HANS, PAWS, and IMDB respectively. CILDA significantly outperforms the second best method (MATE-KD) by 1.6% and 2.2% on HANS and PAWS respectively, and by a margin of 0.2% on IMDB.

5 Conclusion and Future Work

We proposed a min-max adversarial data augmentation framework for KD, which is powered by contrastive distillation loss for intermediate layer matching. Our algorithm maximizes the intermediate and logit representation margin between the teacher and the student models. In future works, we would like to investigate the distillation from super-large models such as Megatron (Shoeybi et al., 2019) and T5 (Raffel et al., 2020). Also, we would like to improve the generator output quality via distillation from generative models like GPT-2 (Radford et al., 2019).

Acknowledgments

We thank Mindspore[2] for the partial support of this work. We thank the anonymous reviewers for their insightful comments.

References

Sanjeev Arora, Hrishikesh Khandeparkar, Mikhail Khodak, Orestis Plevrakis, and Nikunj Saunshi. 2019. A theoretical analysis of contrastive unsupervised representation learning. *arXiv preprint arXiv:1902.09229*.

Tom B Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. 2020. Language models are few-shot learners. *arXiv preprint arXiv:2005.14165*.

Cristian Buciluǎ, Rich Caruana, and Alexandru Niculescu-Mizil. 2006. Model compression. In *Proceedings of the 12th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 535–541.

Ting Chen, Simon Kornblith, Mohammad Norouzi, and Geoffrey Hinton. 2020. A simple framework for contrastive learning of visual representations. *arXiv preprint arXiv:2002.05709*.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. Bert: Pre-training of deep bidirectional transformers for language understanding. *https://arxiv.org/abs/1810.04805*.

Hao Fu, Shaojun Zhou an Qihong Yang, Junjie Tang an Guiquan Liu, Kaikui Liu, and Xiaolong Li. 2021. Lrc-bert: Latent-representation contrastive knowledge distillation for natural language understanding. In *AAAI*.

[2]A new deep learning computing framework https://www.mindspore.cn/

Michael Gutmann and Aapo Hyvärinen. 2010. Noise-contrastive estimation: A new estimation principle for unnormalized statistical models. In *Proceedings of the thirteenth international conference on artificial intelligence and statistics*, pages 297–304. JMLR Workshop and Conference Proceedings.

Xuanli He, Islam Nassar, Jamie Kiros, Gholamreza Haffari, and Mohammad Norouzi. 2021. Generate, annotate, and learn: Generative models advance self-training and knowledge distillation. *arXiv preprint arXiv:2106.06168*.

Geoffrey Hinton, Oriol Vinyals, and Jeff. Dean. 2014. Distilling the knowledge in a neural network. *NIPS Workshop, https://arxiv.org/abs/1503.02531*.

R Devon Hjelm, Alex Fedorov, Samuel Lavoie-Marchildon, Karan Grewal, Phil Bachman, Adam Trischler, and Yoshua Bengio. 2018. Learning deep representations by mutual information estimation and maximization. *arXiv preprint arXiv:1808.06670*.

Aref Jafari, Mehdi Rezagholizadeh, and Ali Ghodsi. 2021a. Knowledge distillation by utilizing backward pass knowledge in neural networks. US Patent App. 17/359,463.

Aref Jafari, Mehdi Rezagholizadeh, Pranav Sharma, and Ali Ghodsi. 2021b. Annealing knowledge distillation. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*, pages 2493–2504.

Eric Jang, Shixiang Gu, and Ben Poole. 2016. Categorical reparameterization with gumbel-softmax. *arXiv preprint arXiv:1611.01144*.

Ganesh Jawahar, Benoît Sagot, and Djamé Seddah. 2019. What Does BERT Learn about the Structure of Language? In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 3651–3657.

Xiaoqi Jiao, Yichun Yin, Lifeng Shang, Xin Jiang, Xiao Chen, Linlin Li, Fang Wang, and Qun Liu. 2019. Tinybert: Distilling bert for natural language understanding. *arXiv preprint arXiv:1909.10351*.

Ehsan Kamalloo, Mehdi Rezagholizadeh, and Ali Ghodsi. 2022. When chosen wisely, more data is what you need: A universal sample-efficient strategy for data augmentation. *arXiv preprint arXiv:2203.09391*.

Ehsan Kamalloo, Mehdi Rezagholizadeh, Peyman Passban, and Ali Ghodsi. 2021. Not far away, not so close: Sample efficient nearest neighbour data augmentation via minimax. *arXiv preprint arXiv:2105.13608*.

Tushar Khot, Ashish Sabharwal, and Peter Clark. 2018. Scitail: A textual entailment dataset from science question answering. In *AAAI*.

Olga Kovaleva, Alexey Romanov, Anna Rogers, and Anna Rumshisky. 2019. Revealing the dark secrets of bert. *arXiv preprint arXiv:1908.08593*.

Tianda Li, Ahmad Rashid, Aref Jafari, Pranav Sharma, Ali Ghodsi, and Mehdi Rezagholizadeh. 2021. How to select one among all? an extensive empirical study towards the robustness of knowledge distillation in natural language understanding. *arXiv preprint arXiv:2109.05696*.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2020. Roberta: A robustly optimized fbertg pretraining approach. *arXiv preprint arXiv:1907.11692*.

R. Thomas McCoy, Ellie Pavlick, and Tal Linzen. 2019. Right for the wrong reasons: Diagnosing syntactic heuristics in natural language inference. *arXiv preprint arXiv:1902.01007*.

Takeru Miyato, Andrew M Dai, and Ian Goodfellow. 2016. Adversarial training methods for semi-supervised text classification. *arXiv preprint arXiv:1605.07725*.

Subhabrata Mukherjee and Ahmed Hassan Awadallah. 2020. Xtremedistil: Multi-stage distillation for massive multilingual models. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 2221–2234.

Peyman Passban, Yimeng Wu, Mehdi Rezagholizadeh, and Qun Liu. 2021. Alp-kd: Attention-based layer projection for knowledge distillation. In *AAAI*.

Yanru Qu, Dinghan Shen, Yelong Shen, Sandra Sajeev, Jiawei Han, and Weizhu Chen. 2021. Coda: Contrast-enhanced and diversity promoting data augmentation for natural language understanding. In *ICLR*.

A. Radford, J. Wu, R. Child, D. Luan, D. Amodei, and I. Sutskever. 2019. Language models are unsupervised multitask learners.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J Liu. 2020. Exploring the limits of transfer learning with a unified text-to-text transformer. *Journal of Machine Learning Research*, 21:1–67.

Ahmad Rashid, Vasileios Lioutas, Abbas Ghaddar, and Mehdi Rezagholizadeh. 2020. Towards zero-shot knowledge distillation for natural language processing. *arXiv preprint arXiv:2012.15495*.

Ahmad Rashid, Vasileios Lioutas, and Mehdi Rezagholizadeh. 2021. MATE-KD: Masked adversarial TExt, a companion to knowledge distillation. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 1062–1071, Online. Association for Computational Linguistics.

Mehdi Rezagholizadeh, Aref Jafari, Puneeth Salad, Pranav Sharma, Ali Saheb Pasand, and Ali Ghodsi. 2021. Pro-kd: Progressive distillation by following the footsteps of the teacher. *arXiv preprint arXiv:2110.08532*.

V. Sanh, L. Debut, J. Chaumond, and T. Wolf. 2019a. Distilbert, a distilled version of bert: smaller, faster, cheaper and lighter. *arXiv preprint arXiv:1910.01108*.

Victor Sanh, Lysandre Debut, Julien Chaumond, and Thomas Wolf. 2019b. Distilroberta, a distilled version of roberta: smaller, faster, cheaper and lighter. https://huggingface.co/distilroberta-base.

Dinghan Shen, Mingzhi Zheng, Yelong Shen, Yanru Qu, and Weizhu Chen. 2020. A simple but tough-to-beat data augmentation approach for natural language understanding and generation. *arXiv preprint arXiv:2009.13818*.

Mohammad Shoeybi, Mostofa Patwary, Raul Puri, Patrick LeGresley, Jared Casper, and Bryan Catanzaro. 2019. Megatron-lm: Training multi-billion parameter language models using model parallelism. *arXiv preprint arXiv:1909.08053*.

Siqi Sun, Yu Cheng, Zhe Gan, and Jingjing Liu. 2019. Patient knowledge distillation for bert model compression. *https://arxiv.org/abs/1908.09355*.

Siqi Sun, Zhe Gan, Yu Cheng, Yuwei Fang, Shuohang Wang, and Jingjing Liu. 2020a. Contrastive distillation on intermediate representations for language model compression. In *EMNLP*.

Zhiqing Sun, Hongkun Yu, Xiaodan Song, Renjie Liu, Yiming Yang, and Denny Zhou. 2020b. Mobilebert: a compact task-agnostic bert for resource-limited devices. *arXiv preprint arXiv:2004.02984*.

Raphael Tang, Yao Lu, Linqing Liu, Lili Mou, Olga Vechtomova, and Jimmy Lin. 2019. Distilling task-specific knowledge from bert into simple neural networks. *arXiv preprint arXiv:1903.12136*.

Ian Tenney, Dipanjan Das, and Ellie Pavlick. 2019. Bert rediscovers the classical nlp pipeline. *arXiv preprint arXiv:1905.05950*.

Y. Tian, D. Krishnan, and P. Isola. 2019. Contrastive representation distillation. *arXiv preprint arXiv:1910.10699*.

Aaron van den Oord, Yazhe Li, and Oriol Vinyals. 2018. Representation learning with contrastive predictive coding. *arXiv preprint arXiv:1807.03748*.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in neural information processing systems*, pages 5998–6008.

Alex Wang, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel R Bowman. 2018. Glue: A multi-task benchmark and analysis platform for natural language understanding. *arXiv preprint arXiv:1804.07461*.

Wenhui Wang, Hangbo Bao, Shaohan Huang, Li Dong, and Furu Wei. 2020a. Minilmv2: Multi-head self-attention relation distillation for compressing pretrained transformers. *arXiv preprint arXiv:2012.15828*.

Wenhui Wang, Furu Wei, Li Dong, Hangbo Bao, Nan Yang, and Ming Zhou. 2020b. Minilm: Deep self-attention distillation for task-agnostic compression of pre-trained transformers. *arXiv preprint arXiv:2002.10957*.

Yimeng Wu, Peyman Passban, Mehdi Rezagholizade, and Qun Liu. 2020. Why skip if you can combine: A simple knowledge distillation technique for intermediate layers. *arXiv preprint arXiv:2010.03034*.

Yimeng Wu, Mehdi Rezagholizadeh, Abbas Ghaddar, Md Akmal Haidar, and Ali Ghodsi. 2021. Universal-kd: Attention-based output-grounded intermediate layer knowledge distillation. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 7649–7661.

Chen Zhu, Yu Cheng, Zhe Gan, Siqi Sun, Tom Goldstein, and Jingjing Liu. 2019. Freelb: Enhanced adversarial training for natural language understanding. In *ICLR*.

Pro-KD: Progressive Distillation by Following the Footsteps of the Teacher

Mehdi Rezagholizadeh[1*] **Aref Jafari**[2,1*] **Puneeth S.M. Saladi**[1 †]
Pranav Sharma[2] **Ali Saheb Pasand**[1,2] **Ali Ghodsi**[2]
[1]Huawei Noah's Ark Lab
[2]University of Waterloo
{mehdi.rezagholizadeh, puneeth.saladi}@huawei.com
{aref.jafari, p68sharm, asahebpa, ali.ghodsi}@uwaterloo.ca

Abstract

With the ever growing scale of neural models, knowledge distillation (KD) attracts more attention as a prominent tool for neural model compression. However, there are counter intuitive observations in the literature showing some challenging limitations of KD. A case in point is that the best performing checkpoint of the teacher might not necessarily be the best teacher for training the student in KD. Therefore, one important question would be how to find the best checkpoint of the teacher for distillation? Searching through the checkpoints of the teacher would be a very tedious and computationally expensive process, which we refer to as the *checkpoint-search problem*. Moreover, another observation is that larger teachers might not necessarily be better teachers in KD, which is referred to as the *capacity-gap* problem. To address these challenging problems, in this work, we introduce our progressive knowledge distillation (Pro-KD) technique which defines a smoother training path for the student by following the training footprints of the teacher instead of solely relying on distilling from a single mature fully-trained teacher. We demonstrate that our technique is quite effective in mitigating the capacity-gap problem and the checkpoint search problem. We evaluate our technique using a comprehensive set of experiments on different tasks such as image classification (CIFAR-10 and CIFAR-100), natural language understanding tasks of the GLUE benchmark, and question answering (SQuAD 1.1 and 2.0) using BERT-based models and consistently got superior results over state-of-the-art techniques.

1 Introduction

Knowledge distillation (KD) (Hinton et al., 2015) has gained a lot of attention in different deep learning applications such as natural language processing (NLP) (Sun et al., 2019; Jiao et al., 2019; Clark et al., 2019), computer vision (Guo et al., 2020; Mirzadeh et al., 2019), and speech processing (Yang et al., 2019; Yoon et al., 2020; Chebotar and Waters, 2016). Nowadays, the scale of neural networks is growing in the favor of improving their performance (Devlin et al., 2019). A case in point is pre-trained language models (PLMs) such as the GPT-3 (Brown et al., 2020), Pangu-α (Zeng et al., 2021), and WuDao2 which have more than a hundred billions of parameters (Brown et al., 2020). However, deploying these models on devices with limited computational power will be very challenging, if not impossible. In this regard, KD can be used as one of the most prominent neural model compression techniques.

KD adds a new loss term to the regular cross-entropy classification loss. This new loss encourages the student model to mimic the output of a pre-trained teacher network. The teacher network is usually a higher capacity model which is able to learn the underlying function of the training data to

Table 1: The best performing checkpoints vary for different size of the models, and different tasks. It is evident that the best checkpoint of the teacher does not necessarily lead to the best performing student model.

Task	Model	Best #ChPt Epoch	ACC	$\Delta\uparrow$
MRPC	$\text{BERT}_{\text{LARGE}}$(♠T)	**6**	88.22	-
	$\text{BERT}_{\text{SMALL}}$(♣S)	**3**	85.29	**+0.7**
	DistilBERT(♣S)	**5**	89.16	**+0.8**
SST-2	$\text{BERT}_{\text{LARGE}}$(♠T)	**1**	92.89	-
	$\text{BERT}_{\text{SMALL}}$(♣S)	**7**	88.76	**+0.6**
	DistilBERT(♣S)	**7**	91.63	**+0.4**
QNLI	$\text{BERT}_{\text{LARGE}}$(♠T)	**6**	92.4	-
	$\text{BERT}_{\text{SMALL}}$(♣S)	**2**	87.08	**+0.3**
	DistilBERT(♣S)	**4**	90.46	**+0.2**

♠T:Teacher, ♣S: Student, #ChPt: Checkpoint Number, ACC: Accuracy, $\Delta\uparrow$: Accuracy improvement compared to the best checkpoint of the teacher

*Equal Contribution
†This work has been done while Puneeth S.M. Saladi was in Huawei.

Proceedings of the 29th International Conference on Computational Linguistics, pages 4714–4727
October 12–17, 2022.

a good extent. The output prediction of the teacher is called *soft-target* for the student model. In contrast to ground-truth labels coming from the training dataset which only carry the information about a single class, soft-targets can provide some information about the relative distribution of different classes for each training data. Therefore, a pre-trained teacher is able to provide some auxiliary signal besides the labels in the training dataset.

KD has been investigated a lot in the literature. Sun et al. (2019); Passban et al. (2021); Wu et al. (2020) proposed a technique to improve KD by incorporating the intermediate layer matching in the KD loss. Jiao et al. (2019) show a two-stage KD with intermediate layer mapping, attention distillation and embedding distillation for BERT-based models. Mate-KD (Rashid et al., 2021) and MiniMax-KNN KD (Kamalloo et al., 2021) tailor data augmentation for KD, in which augmented samples are generated or selected based on maximum divergence loss between the student and teacher networks. Rashid et al. (2020) propose a zero-shot KD technique in NLP in which the student does not need to access the teacher training data for its training. Clark et al. (2019) use KD for multi-task learning in natural language understanding. Kim and Rush (2016) propose a sequence-level KD solution for machine translation. Guo et al. (2020) introduces a collaborative training of students with different capacities with KD.

Although KD has been successful in many different deep learning tasks, it is subject to some special limitations as well. For example, it is shown in (Lopez-Paz et al., 2015) that, based on VC theory (Vapnik, 1998), the teacher capacity should not be too large in KD. Similar observation is given by Mirzadeh et al. (2019) using empirical and theoretical justifications that KD will be less effective when the capacity gap between the teacher and student is large. This problem is referred to as the *capacity-gap* problem. Mirzadeh et al. (2019) proposed the TAKD solution to this problem by adding one or multiple intermediate teacher assistant (TA) networks to learn from the teacher using KD and then train the student using other distillation processes. However, training intermediate networks can be prohibitive in terms of adding to the training time, computational complexity and extra error propagation. The other case in point is that a fully-trained teacher might not be the best teacher for the student (Cho and Hariharan, 2019). In other words, an early-stopped teacher can be a better option for KD compared to a fully-trained one. This observation implies that we need to search through the checkpoints of the teacher to find the best model for the distillation process. However, this search can be very expensive especially when we deal with PLMs. We refer to this problem as the *checkpoint-search* problem. Our investigations of this checkpoint search problem show the significance of this issue especially dealing with PLMs. Table 1 depicts that the best distillation checkpoint of the teacher varies for each task and for each student configuration and it is very different from the teacher best performing checkpoint.

In this work, we propose our Pro-KD solution to tackle both the capacity-gap and checkpoint-search problems. In Pro-KD, the student grows gradually with the growth of the teacher. We hypothesize that the training path of the teacher can be informative for the student and we should not disregard it. Therefore, in contrast to the original KD where the student learns from the best pre-trained teacher, in Pro-KD the student starts its learning process together with the teacher. Furthermore, in contrast to the TAKD technique which reduces the capacity of the teacher by adding intermediate TAs to the distillation process, in our Pro-KD, we mitigate the capacity gap by making the training path of the student more smooth and gradual by following the training footsteps of the teacher. Moreover, to make the training smooth further for the student, inspired by (Jafari et al., 2021), we apply an adaptive temperature factor to the output of the teacher while being trained. This temperature factor is decreased during the training. We will show the effectiveness of our solution using theoretical justification and empirical evaluations. We evaluate Pro-KD by performing experiments on both NLP (the GLUE (Wang et al., 2018b) benchmark and SQuAD 1.1 and 2.0) and image classification tasks (CIFAR-10, CIFAR-100). The contributions of this paper are summarized in the following:

1. We propose our Pro-KD solution to the KD *capacity-gap* and *checkpoin-search* problems. In Pro-KD, the student follows the training footsteps of the teacher. Moreover, it introduces a dynamic temperature function to the output of the teacher when it is distilled to the student to make the student training gradual and smooth.

2. We apply our technique to ResNET8 model on

both CIFAR-10 and CIFAR-100 image classification tasks, and the natural language inference task on different BERT based models such as DistilRoBERTa, and BERT-Small on the GLUE benchmark and also SQuAQ 1.1 and 2.0 question answering datasets and achieved the state-of-the-art results.

3. Our technique is simple, architecture agnostic, does not require training any extra network and can be applied on top of different variants of KD.

2 Background

Knowledge Distillation KD (Hinton et al., 2015) is a well-known method for neural model compression and also is shown to be an effective regularizer in improving the performance of neural networks in the self-distillation (Yun et al., 2020; Hahn and Choi, 2019) or born-again (Furlanello et al., 2018) setups. KD adds a particular loss term to the regular cross entropy (CE) classification loss:

$$\mathcal{L}_{KD}(\phi) = CE\Big(y, S(x;\phi)\Big) + \mathcal{T}^2 KL\Big(\sigma(\frac{z_t(x;\theta)}{\mathcal{T}}), \sigma(\frac{z_s(x;\phi)}{\mathcal{T}})\Big) \quad (1)$$

where x is the input data and y is its associated label, ϕ and θ refer to the student and teacher parameters, σ is the softmax function, z_s and z_t are the student and teacher logits, $\mathcal{T}$ is the temperature parameter to control the softness of the output probability distributions, CE and KL refer to the cross entropy and KL divergence loss functions respectively.

Regular KD training is a two-stage process in which the teacher is fully trained in the first stage and deployed in training of the student model in the next stage. The student is trained based on the *hard labels* coming from the ground-truth training data and *soft labels* coming from the teacher output predictions.

3 Related Work

In this section, we review the most related works to our paper in the literature.

3.1 Capacity-Gap Problem in KD

The capacity-gap problem in KD, refers to having a more powerful or larger teacher is not necessarily leads to a better training for the student. Mirzadeh et al. (2019) propose their TAKD solution to this problem by introducing an intermediate TA network whose capacity is greater than the student but smaller than the teacher. The target of this TA network is to learn from the teacher and train the student. It is evident that in this setup the capacity gap between the TA and student is less than that of the main teacher and the student. Even though adding the TA network can mitigate the capacity-gap problem, there are two downsides in this technique: first, training a separate TA network is costly; second, the sequence of training multiple networks can lead to error accumulation and error propagation to the student. Moreover, TAKD only showed their results on image classification tasks. Jafari et al. (2021) proposed a TA-free solution for the capacity-gap problem which is called Annealing-KD. Annealing-KD adds an annealed dynamic temperature factor to the output of the teacher to make the training process for the student very gradual. The temperature factor starts from a large value to apply the maximum smoothing to the output of the teacher at the beginning of the training and the gradually tends toward 1 where there is no smoothing effect on the output of the teacher. Our Pro-KD is inspired by Annealing-KD in using a dynamic temperature factor to solve the capacity-gap problem, but we can highlight two main improvements over Annealing-KD: first, Annealing-KD suffers from the checkpoint search problem while Pro-KD does not; second, Pro-KD takes advantage of the intrinsic gradual training of the teacher training which is not considered in Annealing-KD.

3.2 Checkpoint Search Problem in KD

Choi et al. (2020) highlighted their findings over computer vision (CV) models that bigger models are not necessarily better teachers and also early stopped teachers can train students better. To overcome these issues they propose an ad-hoc approach called Early Stopped KD (ESKD); however, ESKD still needs to search among the pool of teacher's early checkpoints and the selection mechanism among this early checkpoints is not clear. Moreover, in PLMs, generally teachers are not fine-tuned for long and yet searching among the checkpoints of large PLMs is expensive. (Jin et al., 2019) proposed Route Constrained Optimization (RCO) distillation which is a curriculum learning technique to follow easy-to-hard training scheme on top of

the teacher's training trajectory. We can deem this technique as the most related work to us with the following differences: 1- our Pro-KD has a temperature factor to make the training process more gradual which can handle the capacity gap better; 2- RCO still requires search to define its trajectory; 3- we evaluated Pro-KD on both CV and NLP tasks but RCO is only evaluated on CV tasks. We will show in the Experiment section that our Pro-KD outperforms RCO consistently.

In the next section, we introduce our Pro-KD solution to the capacity-gap and checkpoint-search problems. Pro-KD exploits a smooth training process for the student to gradually learn from the teacher while being trained from scratch, instead of learning from a fully-trained Teacher.

4 Methodology: Pro-KD

Methodology of this paper concerns addressing the capacity-gap and checkpoint-search problems in KD. We propose our Pro-KD technique which is a progressive training procedure in which the student is trained together with the teacher. Therefore in Pro-KD, the student is not exposed to the output of a fully trained teacher from beginning, and instead, it learns from the teacher at the same time while the the teacher is being trained. This training process for the student is more gradual and also the student can learn from the training path of the teacher. Inspired by Jafari et al. (2021), We define the training in two phases: first, the student is only supervised by an adaptive smoothed version of the teacher; second, the student will be only trained on ground-truth labels. The detail of each phase is explained in the following.

Phase I) General Step-by-Step Training with a Teacher In this phase, the teacher is being trained using the cross entropy loss on the training data. Without loss of generality, let's assume that $\theta(i)$, the optimized parameters of the teacher at the beginning of epoch i, will be updated during the ith epoch of training to obtain $\theta(i+1)$:

$$\begin{aligned} &\mathcal{L}_T(\theta(i)) = CE\big(y, T(x;\theta(i))\big) \\ &\theta(i+1) \leftarrow \min_{\theta} \mathcal{L}_T(\theta(i)) \end{aligned} \tag{2}$$

Then, a smoothed version of the teacher output using a temperature factor $\mathcal{T}_i$ at epoch i will be used to train the student at epoch j:

$$\begin{aligned} &\mathcal{L}_S^I(\phi(j)) = \|z_s(x;\phi(j)) - \frac{z_t(x;\theta(i))}{\mathcal{T}_i}\|_2^2 \\ &\phi(j+1) \leftarrow \min_{\phi} \mathcal{L}_S^I(\phi(j)) \\ &\mathcal{T}_i \in \{\mathcal{T}_{i+1} = \mathcal{T}_i - 1 |\ \mathcal{T}_1 = \tau_{max}, \mathcal{T}_{i+1} \geq 1\} \end{aligned} \tag{3}$$

where $\mathcal{L}_S^I$ refers to the student loss function in phase I, z_s and z_t are the logits of the student and teacher respectively, $\phi(j)$ represents the parameters of the student model at epoch j, $\mathcal{T}_i$ is adaptive temperature factor. The rational behind applying the temperature to the teacher output is to make the output of the teacher smoother for the student to learn, especially at early stages of the training process (Jafari et al., 2021).

It is evident that early-stopped teachers can be better teachers for KD (Cho and Hariharan, 2019). In that case, for an early-stopped teacher, the number of training epochs of the student can go longer than that of the teacher. Therefore, for each given epoch i of the teacher, the student can be trained for $n_i \geq 1$ epochs, subject to $\sum_{i=1}^{\tau_{max}} n_i = N$ where N is the preset total number of student training epochs. n_i is set to a constant integer number in most of our experiments but can be customized as well.

Bear in mind that we apply an adaptive temperature to the teacher logit in the student distillation loss, and this temperature starts from the highest τ_{max} value and decrease linearly with the epoch number of the teacher throughout training. τ_{max} is a hyper-parameter in our training. Moreover, we keep the temperature fixed at each epoch i. After training with the teacher logits, in the next phase the student model will be trained on the ground-truth labels.

Phase II) Training with the Ground-Truth Labels After training the student based on the soft targets of the teacher, the student is trained on the ground-truth labels using the cross entropy loss for a few epochs:

$$\begin{aligned} &\mathcal{L}_S^{II}(\phi(j)) = CE\big(y, S(x;\phi(j))\big) \\ &\phi(j+1) \leftarrow \min_{\phi} \mathcal{L}_S^{II}(\phi(j)). \end{aligned} \tag{4}$$

In summary, our technique enables the student to learn from the teacher more smoothly. Pro-KD is different from regular KD technique in the following aspects:

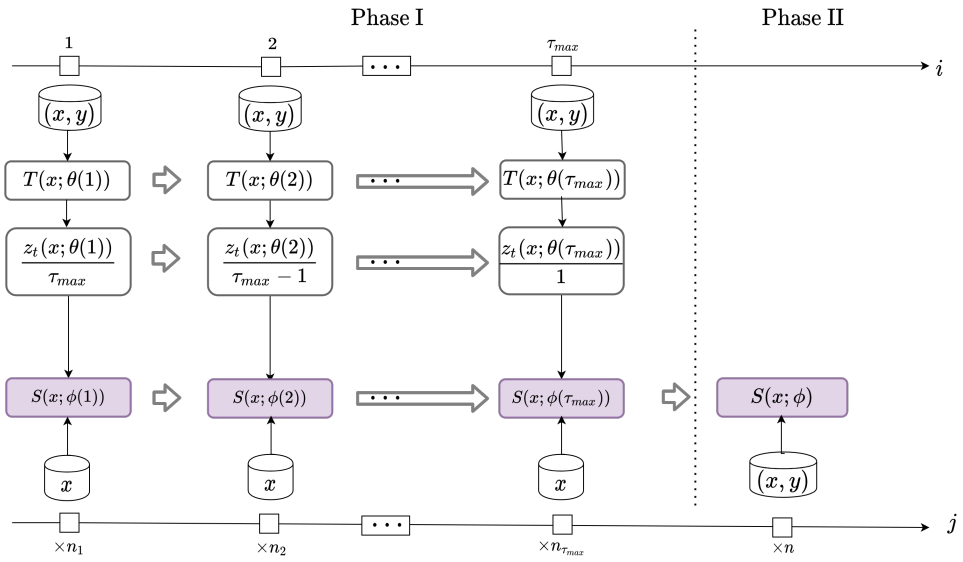

Figure 1: Phase I and Phase II of the Pro-KD method. Phase I: the teacher is trained on the labeled training data. The student at each step tries to mimic the behavior of the corresponding teacher checkpoint. The logits of the teacher at each time step is attenuated with the temperature parameter. We start training of the student from $\mathcal{T} = \tau_{max}$ and go to $\mathcal{T} = 1$. Phase II: training the student with the labeled data only using the cross entropy loss. n_i for $1 \leq i \leq \tau_{max}$ refers to the number of training epochs of the student corresponding to the ith training epoch of the teacher. n_is should add up to N which is the preset total number of training epochs of the student. Bear in mind that since the teacher is usually early stopped, teacher is trained less longer than the student model.

1. rather than distilling from a fully trained teacher; we distill from the teacher during its training;

2. in contrast to regular KD which applies a fixed temperature parameter to both networks, we apply an adaptive temperature factor only to the logits of the teacher;

3. we apply the KD loss and the cross entropy loss in two separate phases.

Moreover, it is worth mentioning that the overall number of training steps of our model follows regular KD techniques. For example, if in regular KD baselines, the model is trained for 30 number of epochs, we train the total training time of phase I and phase II of our model to 30 epochs as well.

4.1 Why Does Pro-KD work?

Lopez-Paz et al. (2015) do an analysis based on VC-dimension theory to discuss the conditions under which KD works better than no-KD scenarios. It is shown that KD works if the following inequality holds:

$$O(\frac{|\mathcal{F}_s|_c + |\mathcal{F}_t|_c}{n^\alpha}) + \varepsilon_t + \varepsilon_l \leq O(\frac{|\mathcal{F}_s|_c}{\sqrt{n}}) + \varepsilon_s \quad (5)$$

where $\mathcal{F}_s$ and $\mathcal{F}_t$ are the function classes corresponding to the teacher and student; $|.|_c$ is a function class capacity measure; $O(.)$ is the estimation error of training the learner; ε_s is the approximation error of the best estimator function belonging to the $\mathcal{F}_s$ class with respect to the underlying function; ε_t is a similar approximation error for the teacher with respect to the underlying function; ε_l is the approximation error of the best student function with respect to the teacher function; n is the number of training samples, and $\frac{1}{2} \leq \alpha \leq 1$ is a parameter related to the difficulty of the problem. Smaller values of α indicate slower training and larger values correspond to faster learning rates of the student. Given Eq. 5 we observe that for larger teacher models, the value of the left hand side of the inequality might get higher than that of the right hand side, which implies that KD might not work as expected for large capacity teachers.

We analyzed the capacity-gap problem based on the inequality 5. To satisfy this inequality when the capacity of the teacher is large (or equivalently the capacity gap between the two networks is large), we can think about reducing the capacity of the teacher (eg. the TAKD technique (Mirzadeh et al., 2019) introduces a smaller TA network instead of a large teacher), or increase α. Increasing α is

Table 2: Comparing the test accuracy of Pro-KD, TAKD (Mirzadeh et al., 2019), Annealing-KD (Jafarpour et al., 2021), RCO (Jin et al., 2019), regular KD, and student without teacher on CIFAR-10 dataset with both ResNet and CNN models

Model	Type	Training method	Accuracy
ResNet	Teacher(110)	from scratch	93.8
	TA(20)	KD	92.39
	Student(8)	from scratch	88.44
	Student(8)	KD	88.45
	Student(8)	TAKD	88.47
	student(8)	RCO	88.90
	Student(8)	Annealing-KD	89.44
	Student(8)	**Pro-KD (ours)**	**90.01**

Table 3: Comparing the test accuracy of Pro-KD, TAKD (Mirzadeh et al., 2019), Annealing-KD (Jafarpour et al., 2021), RCO (Jin et al., 2019), regular KD, and student without teacher on CIFAR-100 dataset with both ResNet and CNN models

Model	Type	Training method	Accuracy
ResNet	teacher(110)	from scratch	71.92
	TA(20)	KD	67.6
	student(8)	from scratch	61.37
	student(8)	KD	61.41
	student(8)	RCO	61.62
	student(8)	TAKD	61.82
	student(8)	Annealing KD	63.1
	student(8)	**Pro-KD (ours)**	**63.43**

equivalent to making the training process of student easier and smoother. It has been shown in (Li et al., 2017; Chaudhari et al., 2019) that learning a smoother loss is easier than a sharp one. Therefore, we tried to make the training of the student smoother by following the teacher's training steps and also applying the adaptive temperature factor. In addition to this justification, we present a theoretical justification in the Appendix.

5 Experiments and Results

In this section, we evaluate our Pro-KD on 3 different sets of experiments: on image classification, natural language understanding and question answering tasks. In all these three experiments, we compare Pro-KD with the state-of-the-art KD techniques which can address the capacity gap problem such as TAKD (Mirzadeh et al., 2019), and Annealing-KD (Jafari et al., 2021), or RCO (Jin et al., 2019) technique which does not require searching teacher checkpoints for KD. We also include baselines such as the original KD method (Hinton et al., 2015) and also the regular training without KD.

5.1 Experimental Setup for Image Classification Tasks

Data We used CIFAR-10 and CIFAR-100 (Krizhevsky et al., 2009) datasets for our experiments on the image classification tasks. Both datasets have 60,000 of 32×32 color images distributed into 50,000 training and 10,000 test samples, with 10 classes for CIFAR-10 and 100 classes for CIFAR-100

Setup For these experiments, we used ResNet-8 as the student and resNet-110 as the teacher models. The experimental setups are similar to TAKD method (Mirzadeh et al., 2019). Also for the TAKD baseline, we used ResNet-20 as the TA model. The results of these experiments can be found in table 2 and 3 for comparison. For the baselines, first, the ResNet-110 teacher is trained from scratch on the given datasets and then it is used for training baselines with KD. For TAKD baseline, the original KD method is applied to train the TA network with the teacher network and it is applied to train the student network with TA network. For training the student with the proposed Pro-KD method, we trained the teacher for 160 epochs. For training the student, we used maximum temperature 10 and learning rate 0.1. For every 16 epochs, we decrease the temperature by 1 and $n_i = 1$.

Results As it is shown in Tables 2 and 3, Pro-KD outperforms other baselines for both CIFAR-10 and CIFAR-100 experiments. Also, it is worth mentioning that the Annealing-KD technique is the second best result. RCO and TAKD both perform on-par with the regular KD and training the student from scratch.

5.2 Experimental Setup for Natural Language Understanding Tasks on the GLUE Benchmark

Data We use the General Language Understanding Evaluation (GLUE) benchmark (Wang et al., 2018a), which consists of 9 natural language understanding tasks. The tasks cover textual entailment (RTE and MNLI), question-answer entailment (QNLI), paraphrase (MRPC), question paraphrase (QQP), sentiment (SST-2), textual similarity (STS-B), linguistic acceptability (CoLA), and Winograd Schema (WNLI).

Table 4: Dev set results of training DistilRoBERTa using the $\text{RoBERTa}_{\text{Large}}$ model on the GLUE benchmark.

KD Method	CoLA	RTE	MRPC	STS-B	SST-2	QNLI	QQP	MNLI	Score
$\text{RoBERTa}_{\text{Large}}$ (Teacher)	67	85	91.63	92.53	96.21	94.53	91.45	89.94/89.97	88.54
DistilRoBERTa(NoKD)	61.9	69.31	89.85	88.46	91.86	91.31	90.04	84.03/ 83.69	83.32
Vanilla KD	60.97	71.11	90.2	88.86	92.54	91.37	91.64	84.18/84.11	83.85
TAKD	61.15	71.84	89.91	88.94	92.54	91.32	91.7	83.89/84.18	83.93
RCO (Jin et al., 2019)	60.66	72.2	90.56	88.41	91.97	91.09	90.04	88.04/84.18	63.63
Annealing KD	61.67	73.64	90.6	89.01	93.11	91.64	91.5	85.34/84.6	84.52
Pro-KD (Ours)	62.14	73.64	91.9	88.8	92.66	91.47	91.53	84.83/84.84	**84.62**

Table 5: Test set results of DistilRoBERTa trained on the GLUE tasks using the $\text{RoBERTa}_{\text{Large}}$ teacher model.

KD Method	CoLA	RTE	MRPC	STS-B	SST-2	QNLI	QQP	MNLI	Score
Vanilla KD	54.3	74.1	83.4	85.3	93.1	90.8	80.7	83.6/82.9	80.62
TAKD	53.2	74.2	84.7	85	93.2	91.0	80.7	83.8/83.2	80.69
RCO (Jin et al., 2019)	55.1	73.0	87.85	84.4	93.5	88.9	78.66	83.2/82.4	80.51
Annealing KD	54	73.7	85.95	86.8	93.6	90.8	81.15	83.8/83.9	81.23
Pro-KD (Ours)	55.8	73.6	86.45	86.95	93.4	91.0	81.05	84.6/83.8	**81.56**

Table 6: BERT-Small results for Pro-KD on the GLUE dev set

KD Method	CoLA	RTE	MRPC	STS-B	SST-2	QNLI	QQP	MNLI	Score
$\text{BERT}_{\text{Large}}$	61.89	68.96	88.22	89.58	92.89	92.4	90.23	86.1/86.25	83.79
BERT-Small (NoKD)	44.05	64.98	83.75	87.41	88.3	86.49	88.43	78.42/78.57	77.74
Vanilla KD	43.28	64.98	84.96	85.95	88.65	86.75	88.24	78.62 /78.55	77.67
TAKD	43.79	65.7	83.98	86.44	88.88	86.78	88.4	78.78/ 78.64	77.84
RCO (Jin et al., 2019)	44.32	65.7	84.91	85.48	88.99	86.32	87.52	78.35 /78.85	77.73
Annealing KD	45	63.9	87.09	87.04	89.56	86.99	88.58	78.66/78.23	78.33
Pro-KD	42.37	66.79	87.78	87.09	89.91	87.88	88.79	79.18/ 79.17	**78.72**

Table 7: The GLUE leaderboard test results of training BERT-Small from the $\text{BERT}_{\text{Large}}$ teacher.

KD Method	CoLA	RTE	MRPC	STS-B	SST-2	QNLI	QQP	MNLI	Score
BERT-Small (NoKD)	41.3	62.6	79.7	80.05	89	86.3	78.05	78.3/ 77.6	74.37
Vanilla KD	37.3	63.4	80.55	78.15	90.2	86.5	78.25	78/76.5	73.95
TAKD	38.5	62.3	80.5	79.25	89.7	86.7	78	78.2/76.9	74.06
RCO (Jin et al., 2019)	40.3	61.7	79.75	78.95	90.6	86.4	78.35	78.3/ 77.3	74.23
Annealing KD	38.6	63.1	81.85	80.6	91.2	87.3	78.35	77.8// 77.4	74.83
Pro-KD	39	62.7	82.9	80.45	91.2	87.5	79.15	78.6/78.2	**75.16**

Table 8: DistilRoBERTa results for Pro-KD on SQuAD

KD Method	Squad 1.1	Squad 2.0
Teacher	93.7	87
Vanilla KD	85	73.65
TAKD	85.4	73.8
Pro-KD	**86**	**76**

Table 9: BERT-Small results for Pro-KD on SQuAD

KD Method	Squad 1.1	Squad 2.0
Teacher	90.2	81
Vanilla KD	78.54	61.66
TAKD	78.5	61.66
Pro-KD	**79.7**	**62.76**

Setup We perform experiments with multiple students of varying capacities. In the first experiment (Table 4 and 5), we use RoBERTa-large (24-layers) as teacher, DistilRoBERTa (6-layers) (Sanh et al., 2019) as student, and RoBERTa-base (12-layers) (Liu et al., 2019) as the teacher-assistant for the TAKD baseline. For Pro-KD, we train the teacher for 5 epochs, and for training the student we use a maximum temperature of 5, $n_i = 2$, and the learning rate of 2e-5. For the second experiment (Table 6, 7), we use BERT-large (24-layers) as teacher, BERT-Small (4-layers) (Bhargava et al., 2021) as student, and BERT-base (12-layers) (Devlin et al., 2018) as the teacher-assistant for TAKD. We train the teacher for 7 epochs, and for the student we use a maximum temperature of 7 for all tasks, and $n_i = 1$. For the learning rate, we use 5e-5 for RTE and MRPC, and 2e-5 for all other tasks. Additional details about other hyper-parameters can be found in the appendix.

Table 10: Experimenting the impact of the adaptive temperature factor in Pro-KD using DistilRoBERTa trained with RoBERTa-large on the GLUE benchmark

KD Method	CoLA (8.5k)	RTE (2.5k)	MRPC (3.5k)	STS-B (5.7k)	SST-2 (67k)	QNLI (108k)	QQP (363k)	MNLI (392K) (392k)	WNLI	Avg
Pro-KD	**62.14**	**73.64**	**91.9**	**88.8**	**92.66**	91.47	91.53	84.83/84.84	**57.74**	**81.63**
Pro-KD w/o $\mathcal{T}$	58.66	70.03	91.74	88.0	92.43	**91.76**	**91.55**	**84.95/85.41**	56.33	80.61

Results We present our results in Tables 4, 5 and 6. Tables 4 and 6 compare the performance of Pro-KD with Vanilla KD and TAKD on the GLUE dev set, while Table 5 presents the results of DistilRoBERTa on the test set based on GLUE benchmark's leaderboard. We see that Pro-KD outperforms Vanilla KD and TAKD for both DistilRoBERTa and BERT-Small models. Even though TAKD is able to improve over Vanilla KD, the performance gain is much smaller compared to Pro-KD, demonstrating Pro-KD's high effectiveness in dealing with the large capacity gap problem.

5.3 Experimental Setup for Question Answering

Data We use the Stanford Question Answering Datasets (SQuAD v1.1 (Rajpurkar et al., 2016) and SQuAD v2.0) which are a collection of 100k crowdsourced question/answer pairs. Given a question and a passage from Wikipedia containing the answer, the task is to predict the answer text span in the passage.The SQuAD 2.0 task extends the SQuAD 1.1 problem definition by allowing for the possibility that no short answer exists in the provided paragraph.

Setup Similar to the GLUE experiments, we perform experiments on 2 different students but with a couple of less baselines because Annealing KD and RCO do not report any result on SQuAD. In the first experiment (Table 8), we use RoBERTa-large (24-layers) as teacher, DistilRoBERTa (6-layers) as student, and RoBERTa-base (12-layers) as the teacher-assistant for the TAKD baseline. We train the teacher for 3 epochs, and for the student we use a maximum temperature of 3, $n_i = 1$, and a learning rate of 3e-5 with a batch size of 12 for both Squad v1.1 and Squad 2.0. For the second experiment (Table 9), we use BERT-large (24-layers) as teacher, BERT-Small (4-layers) as student, and BERT-base (12-layers) as the teacher-assistant for TAKD. We train the teacher for 3 epochs, and for the student we use a maximum temperature of 3, and $n_i = 1$. and a learning rate of 3e-5 with a batch size of 12. Additional details about other hyper-parameters can be found in the appendix.

Results We present our results in Tables 8 and 9. We again see that Pro-KD consistently outperforms Vanilla KD and TAKD for both Squad v1.1 and Squad 2.0 tasks.

5.4 Ablation Studies and Further Analysis

The Impact of the Temperature Factor Here we show the impact of having the adaptive temperature factor in our technique. The adaptive temperature helps the student model to be exposed to a softened version of the teacher in earlier stages. In this regard we repeated our experiment on the GLUE benchmark when the student is DistilRoBERTa and the teacher is RoBERTa-large. Table 10 shows the result of Pro-KD with and without the temperature. The results indicate that dropping the adaptive temperature will hamper the performance of our model by about 1% on the average GLUE score.

6 Conclusion

In this paper, we highlighted the importance of the capacity-gap problem in KD. We used the VC-dimension analysis to show that dealing with larger teachers may discount the benefit of using knowledge distillation in training. To address this capacity-gap problem, we introduced our Pro-KD technique. Our technique was based on defining a smoother training journey for the student by following the training footprints of the teacher instead of solely relying on distilling from a mature fully-trained teacher. In other words, in contrast to the regular knowledge distillation recipe for model compression in which the student model learns only from a fix fully-trained teacher, in our Pro-KD method, the student learns even from the training path of the teacher. We believe that following the training footsteps of the teacher can be quite influential in improving the student training and can lead to mitigating the capacity-gap problem in KD as well. We showed our theoretical analysis and justifications to support this idea; moreover, we evaluated our technique using a comprehensive set

of experiments on different tasks such as image classification (CIFAR-10 and CIFAR-100), NLP language understanding tasks of the GLUE benchmark, and question answering (SQuAD 1.1 and 2.0) using BERT-based models and consistently got superior results.

Acknowledgments

We thank Mindspore[1] for the partial support of this work. We thank the anonymous reviewers for their insightful comments.

References

Prajjwal Bhargava, Aleksandr Drozd, and Anna Rogers. 2021. Generalization in nli: Ways (not) to go beyond simple heuristics.

Tom B Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. 2020. Language models are few-shot learners. *arXiv preprint arXiv:2005.14165*.

Pratik Chaudhari, Anna Choromanska, Stefano Soatto, Yann LeCun, Carlo Baldassi, Christian Borgs, Jennifer Chayes, Levent Sagun, and Riccardo Zecchina. 2019. Entropy-sgd: Biasing gradient descent into wide valleys. *Journal of Statistical Mechanics: Theory and Experiment*, 2019(12):124018.

Yevgen Chebotar and Austin Waters. 2016. Distilling knowledge from ensembles of neural networks for speech recognition. In *INTERSPEECH*.

Jang Hyun Cho and Bharath Hariharan. 2019. On the efficacy of knowledge distillation. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 4794–4802.

Woo Suk Choi, Kyoung-Woon On, Yu-Jung Heo, and Byoung-Tak Zhang. 2020. Toward general scene graph: Integration of visual semantic knowledge with entity synset alignment. In *Proceedings of the First Workshop on Advances in Language and Vision Research*, pages 7–11, Online. Association for Computational Linguistics.

Kevin Clark, Minh-Thang Luong, Urvashi Khandelwal, Christopher D Manning, and Quoc V Le. 2019. Bam! born-again multi-task networks for natural language understanding. *arXiv preprint arXiv:1907.04829*.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*.

[1]A new deep learning computing framework https://www.mindspore.cn/

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, Minneapolis, Minnesota. Association for Computational Linguistics.

Bin Dong, Jikai Hou, Yiping Lu, and Zhihua Zhang. 2019. Distillation $\approx$ Early stopping? harvesting dark knowledge utilizing anisotropic information retrieval for overparameterized neural network. *arXiv preprint arXiv:1910.01255*.

Simon S Du, Xiyu Zhai, Barnabas Poczos, and Aarti Singh. 2018. Gradient descent provably optimizes over-parameterized neural networks. *arXiv preprint arXiv:1810.02054*.

Tommaso Furlanello, Zachary C Lipton, Michael Tschannen, Laurent Itti, and Anima Anandkumar. 2018. Born again neural networks. *arXiv preprint arXiv:1805.04770*.

Qiushan Guo, Xinjiang Wang, Yichao Wu, Zhipeng Yu, Ding Liang, Xiaolin Hu, and Ping Luo. 2020. Online knowledge distillation via collaborative learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 11020–11029.

Sangchul Hahn and Heeyoul Choi. 2019. Self-knowledge distillation in natural language processing. *arXiv preprint arXiv:1908.01851*.

Geoffrey Hinton, Oriol Vinyals, and Jeff Dean. 2015. Distilling the knowledge in a neural network. *arXiv preprint arXiv:1503.02531*.

Arthur Jacot, Franck Gabriel, and Clément Hongler. 2018. Neural tangent kernel: Convergence and generalization in neural networks. *arXiv preprint arXiv:1806.07572*.

Aref Jafari, Mehdi Rezagholizadeh, Pranav Sharma, and Ali Ghodsi. 2021. Annealing knowledge distillation. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*, pages 2493–2504, Online. Association for Computational Linguistics.

Borna Jafarpour, Dawn Sepehr, and Nick Pogrebnyakov. 2021. Active curriculum learning. In *Proceedings of the First Workshop on Interactive Learning for Natural Language Processing*, pages 40–45, Online. Association for Computational Linguistics.

Xiaoqi Jiao, Yichun Yin, Lifeng Shang, Xin Jiang, Xiao Chen, Linlin Li, Fang Wang, and Qun Liu. 2019. Tinybert: Distilling bert for natural language understanding. *arXiv preprint arXiv:1909.10351*.

Xiao Jin, Baoyun Peng, Yichao Wu, Yu Liu, Jiaheng Liu, Ding Liang, Junjie Yan, and Xiaolin Hu. 2019. Knowledge distillation via route constrained optimization. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 1345–1354.

Ehsan Kamalloo, Mehdi Rezagholizadeh, Peyman Passban, and Ali Ghodsi. 2021. Not far away, not so close: Sample efficient nearest neighbour data augmentation via MiniMax. In *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*, pages 3522–3533, Online. Association for Computational Linguistics.

Yoon Kim and Alexander M Rush. 2016. Sequence-level knowledge distillation. *arXiv preprint arXiv:1606.07947*.

Alex Krizhevsky, Geoffrey Hinton, et al. 2009. Learning multiple layers of features from tiny images.

Hao Li, Zheng Xu, Gavin Taylor, Christoph Studer, and Tom Goldstein. 2017. Visualizing the loss landscape of neural nets. *arXiv preprint arXiv:1712.09913*.

Mingchen Li, Mahdi Soltanolkotabi, and Samet Oymak. 2020. Gradient descent with early stopping is provably robust to label noise for overparameterized neural networks. In *International Conference on Artificial Intelligence and Statistics*, pages 4313–4324. PMLR.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Roberta: A robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692*.

David Lopez-Paz, Léon Bottou, Bernhard Schölkopf, and Vladimir Vapnik. 2015. Unifying distillation and privileged information. *arXiv preprint arXiv:1511.03643*.

Seyed-Iman Mirzadeh, Mehrdad Farajtabar, Ang Li, and Hassan Ghasemzadeh. 2019. Improved knowledge distillation via teacher assistant: Bridging the gap between student and teacher. *arXiv preprint arXiv:1902.03393*.

Samet Oymak and Mahdi Soltanolkotabi. 2019. Overparameterized nonlinear learning: Gradient descent takes the shortest path? In *International Conference on Machine Learning*, pages 4951–4960. PMLR.

Peyman Passban, Yimeng Wu, Mehdi Rezagholizadeh, and Qun Liu. 2021. ALP-KD: attention-based layer projection for knowledge distillation. In *Thirty-Fifth AAAI Conference on Artificial Intelligence, AAAI 2021, Thirty-Third Conference on Innovative Applications of Artificial Intelligence, IAAI 2021, The Eleventh Symposium on Educational Advances in Artificial Intelligence, EAAI 2021, Virtual Event, February 2-9, 2021*, pages 13657–13665. AAAI Press.

Pranav Rajpurkar, Jian Zhang, Konstantin Lopyrev, and Percy Liang. 2016. Squad: 100,000+ questions for machine comprehension of text. *arXiv preprint arXiv:1606.05250*.

Ahmad Rashid, Vasileios Lioutas, Abbas Ghaddar, and Mehdi Rezagholizadeh. 2020. Towards zero-shot knowledge distillation for natural language processing. *arXiv preprint arXiv:2012.15495*.

Ahmad Rashid, Vasileios Lioutas, and Mehdi Rezagholizadeh. 2021. MATE-KD: Masked adversarial TExt, a companion to knowledge distillation. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 1062–1071, Online. Association for Computational Linguistics.

Victor Sanh, Lysandre Debut, Julien Chaumond, and Thomas Wolf. 2019. Distilbert, a distilled version of bert: smaller, faster, cheaper and lighter. *ArXiv*, abs/1910.01108.

Andrew M Saxe, Yamini Bansal, Joel Dapello, Madhu Advani, Artemy Kolchinsky, Brendan D Tracey, and David D Cox. 2019. On the information bottleneck theory of deep learning. *Journal of Statistical Mechanics: Theory and Experiment*, 2019(12):124020.

Siqi Sun, Yu Cheng, Zhe Gan, and Jingjing Liu. 2019. Patient knowledge distillation for bert model compression. *arXiv preprint arXiv:1908.09355*.

Zhiqing Sun, Hongkun Yu, Xiaodan Song, Renjie Liu, Yiming Yang, and Denny Zhou. Mobilebert: Task-agnostic compression of bert for resource limited devices.

Naftali Tishby and Noga Zaslavsky. 2015. Deep learning and the information bottleneck principle. In *2015 IEEE Information Theory Workshop (ITW)*, pages 1–5. IEEE.

Vladimir N Vapnik. 1998. Statistical learning theory. *Adaptive and learning systems for signal processing communications and control*.

Alex Wang, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel Bowman. 2018a. GLUE: A multi-task benchmark and analysis platform for natural language understanding. In *Proceedings of the 2018 EMNLP Workshop BlackboxNLP: Analyzing and Interpreting Neural Networks for NLP*, pages 353–355, Brussels, Belgium. Association for Computational Linguistics.

Alex Wang, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel R Bowman. 2018b. Glue: A multi-task benchmark and analysis platform for natural language understanding. *arXiv preprint arXiv:1804.07461*.

Yimeng Wu, Peyman Passban, Mehdi Rezagholizadeh, and Qun Liu. 2020. Why skip if you can combine: A simple knowledge distillation technique for intermediate layers. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1016–1021, Online. Association for Computational Linguistics.

Zhenchuan Yang, Chun Zhang, Weibin Zhang, Jianxiu Jin, and Dongpeng Chen. 2019. Essence knowledge distillation for speech recognition. *ArXiv*, abs/1906.10834.

Ji Won Yoon, Hyeonseung Lee, Hyung Yong Kim, Won Ik Cho, and Nam Soo Kim. 2020. Tutornet: Towards flexible knowledge distillation for end-to-end speech recognition. *ArXiv*, abs/2008.00671.

Sukmin Yun, Jongjin Park, Kimin Lee, and Jinwoo Shin. 2020. Regularizing class-wise predictions via self-knowledge distillation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 13876–13885.

Wei Zeng, Xiaozhe Ren, Teng Su, Hui Wang, Yi Liao, Zhiwei Wang, Xin Jiang, ZhenZhang Yang, Kaisheng Wang, Xiaoda Zhang, et al. 2021. Pangu-α: Large-scale autoregressive pretrained chinese language models with auto-parallel computation. *arXiv preprint arXiv:2104.12369*.

A Why Does Pro-KD Work?

In this section, first we state stopping the training procedure of the teacher at early epochs will improve Knowledge Distillation. Then, we argue that our method will exploit this fact without requiring any effort on finding the epoch at which early stopping the teacher leads to the optimum Knowledge Distillation. Finally, we will conclude that training the student together with the teacher leads to an important advantage that provides the student with the opportunity of exploiting the knowledge of the optimum teacher.

In knowledge distillation, soft labels coming from a pre-trained teacher contains some so-called *dark knowledge* (Dong et al., 2019) which gives a signal about the relative differences between the probabilities of the classes.

It is shown by (Cho and Hariharan, 2019) and (Dong et al., 2019) that early stopping during training of the teacher network improves the performance of Knowledge Distillation significantly; in other words, there is an optimal epoch at which if teacher training is stopped, the resulting teacher will act as a better teacher compared to the ones stopped at earlier or later epochs. Authors of (Cho and Hariharan, 2019) have shown this fact through experiments based on performing KD by using pre-trained teacher networks stopped at different epochs, and they also have indicated that changing the temperature cannot compensate lost Dark Knowledge. Authors of (Dong et al., 2019) have provided mathematical justification showing that neural network learns more useful information faster. They have used the methods proposed in papers (Du et al., 2018; Oymak and Soltanolkotabi, 2019; Li et al., 2020), and the concept of Neural Tangent Kernel introduced by (Jacot et al., 2018) in order to reach an asymptotic conclusion. This conclusion states that for infinite wide neural networks, gradient descend algorithm searches over different direction with different pace; that is to say, the projection of the loss function in different eigenspaces evolves with different rate. Those rates can be calculated as follows (Dong et al., 2019):

$$\begin{aligned}\langle (u_t - y), e_i \rangle &= \langle (I - \eta H^*)(u_t - y), e_i \rangle \\ &= \langle (u_t - y), (I - \eta H^*) e_i \rangle \\ &= (1 - \eta \lambda_i) \langle (u_t - y), e_i \rangle\end{aligned} \tag{6}$$

where u_t is the output of model, y is the true label, e_i is the eigenvector corresponding to the ith

eigenvalue(λ_i) of the static Gram matrix(H^*) of the network. The gram matrix of neural networks is a function of time in general; however, in infinitely wide neural networks, this matrix will be static and called Neural Tangent Kernel (Jacot et al., 2018). This static $n \times n$ matrix, n is the number of training samples, can be calculated as follows (Dong et al., 2019):

$$H^* = \Big(\langle \frac{\partial f(\theta, x_i)}{\partial \theta} . \frac{\partial f(\theta, x_j)}{\partial \theta} \rangle \Big)_{i,j} \qquad (7)$$

where $f(\theta, \cdot)$ is the output of our model, θ are the parameters of the neural network, and x_i is the ith data sample.

Arguments provided by (Dong et al., 2019) state that a neural network learns more useful information faster than non-principal pieces of information about the input samples; however, it does not explain why continuing training teacher will decrease the level of that "Dark Knowledge". To justify this part, we use results from Tishby, Naftali, and Noga Zaslavsky. "Deep learning and the information bottleneck principle."(Tishby and Zaslavsky, 2015) and Saxe, Andrew M., et al. "On the information bottleneck theory of deep learning." (Saxe et al., 2019). In (Saxe et al., 2019) and (Tishby and Zaslavsky, 2015), authors show that there are two stages during training a deep neural network, initial fitting phase and compression phase. These two stages have some important characteristics:

1. Initial Fitting: During this phase, the mutual information between the output of different layers and true labels is increasing. Also, the mutual information between output of layers and the input samples is increasing as well. In other words, network is gaining information about both labels and input samples. Therefore, network is gaining more aforementioned Dark Knowledge as it is gaining information about relative similarities between samples from different classes.

2. Compression phase: During this phase, the mutual information between the output of different layers and true labels is increasing; however, the mutual information between output of layers and the input samples is decreasing. In other words, network tries to compress and discard information which it has gained about input samples (Saxe et al., 2019). We can state that Dark Knowledge is decreasing during this phase; that is because, network is forgetting relationships between data samples, and at the same time, it is gaining more information about labels. This procedure leads to more confidence on found probabilities for each class. As a result, the information about relative similarities between classes decreases and probabilities tend to one-hot vectors.

Based on the aforementioned facts, a teacher which has stopped learning at the optimum epoch provides more informative pieces of information to the student. Our method exploits the Dark Knowledge of the best teacher by training teacher and students together; in our method, during training teacher and student together, at some point teacher will have the highest level of Dark Knowledge which leads to the best student. The important point here is that we do not need to find the optimum epoch at which this highest level of Dark Knowledge will be achieved. Since we are performing Knowledge Distillation in fixed intervals and storing intermediate checkpoints, the student trained by guidance of the best teacher will be one of these checkpoints and we have access to that.

In contrast with usual KD methods, pre-trained teacher networks are not used in our methods; in fact, we train teacher alongside with the student. Based on the aforementioned arguments, Pro-KD outperforms other KD methods as it gives us the opportunity of exploiting the experience and the Dark Knowledge provided by the best teacher, the teacher which has stopped learning at the optimum epoch.

B Hyper-parameters

In this section, we summarize the hyper-parameters used in our experiments.

Table 11: Model specific Hyper-parameters for BERT-Small on GLUE

Hyper-parameter	CoLA	RTE	MRPC	STS-B	SST-2	QNLI	QQP	MNLI	WNLI
Learning Rate	2e-5	5e-5	5e-5	2e-5	2e-5	2e-5	2e-5	2e-5	2e-5
N (Teacher Epochs)	7	7	7	7	7	7	7	7	7
τ_{max}	7	7	7	7	7	7	7	7	7
n (Phase 2 Epochs)	10	10	10	10	10	10	10	10	10

Table 12: Common Hyper-parameters for DistilRoBERTa and BERT-Small models on GLUE

Hyper-parameter	CoLA	RTE	MRPC	STS-B	SST-2	QNLI	QQP	MNLI	WNLI
Batch Size	32	32	32	32	32	32	32	32	32
Max Seq. Length	128	128	128	128	128	128	128	128	128
Vanilla KD Alpha	0.5	0.5	0.5	0.5	0.5	0.5	0.5	0.5	0.5
Gradient Clipping	1	1	1	1	1	1	1	1	1

Table 13: Model specific Hyper-parameters for DistilRoBERTa on GLUE

Hyper-parameter	CoLA	RTE	MRPC	STS-B	SST-2	QNLI	QQP	MNLI	WNLI
Learning Rate	2e-5	2e-5	2e-5	2e-5	2e-5	2e-5	2e-5	2e-5	2e-5
N (Teacher Epochs)	5	5	5	5	5	5	5	5	5
τ_{max}	5	5	5	5	5	5	5	5	5
n (Phase 2 epochs)	10	10	10	10	10	10	10	10	10

Table 14: Model specific Hyper-parameters for Distil-RoBERTa on SQuAD

Hyper-parameter	SQuAD 1.1/2.0	SQuAD Teacher
Learning Rate	3e-5	1.5e-5
Batch Size	12	12
Max Seq. Length	384	384
Doc Stride	128	128
Weight Decay	-	0.01
N (Teacher Epochs)	3	3
τ_{max}	3	-
n (Phase 2 Epochs)	6	-

Table 15: Model specific Hyper-parameters for BERT-Small on SQuAD

Hyper-parameter	SQuAD 1.1/2.0	SQuAD Teacher
Learning Rate	3e-5	3e-5
Batch Size	12	12
Max Seq. Length	384	384
Doc Stride	128	128
N (Teacher Epochs)	3	3
τ_{max}	3	-
n (Phase 2 Epochs)	5	-

Classical Sequence Match is a Competitive Few-Shot One-Class Learner

Mengting Hu[1] **Hang Gao**[2*] **Yinhao Bai**[1] **Mingming Liu**[1]
[1] College of Software, Nankai University
[2] Institute for Public Safety Research, Tsinghua University
mthu@nankai.edu.cn, gaohang@mail.tsinghua.edu.cn
yinhao@mail.nankai.edu.cn, liumingming@nankai.edu.cn

Abstract

Nowadays, transformer-based models gradually become the *"default choice"* for artificial intelligence pioneers. The models also show superiority even in the few-shot scenarios. In this paper, we revisit the classical methods and propose a new few-shot alternative. Specifically, we investigate the few-shot one-class problem, which actually takes a known sample as a reference to detect whether an unknown instance belongs to the same class. This problem can be studied from the perspective of sequence match. It is shown that with meta-learning, the classical sequence match method, i.e. Compare-Aggregate, significantly outperforms transformer ones. The classical approach requires much less training cost. Furthermore, we perform an empirical comparison between two kinds of sequence match approaches under simple fine-tuning and meta-learning. Meta-learning causes the transformer models' features to have high-correlation dimensions. The reason is closely related to the number of layers and heads of transformer models. Experimental codes and data are available at https://github.com/hmt2014/FewOne.

1 Introduction

When the labeled data is scarce in practical application, it is struggled to learn a well-performed model using deep learning algorithms. Yet annotating data costs much labor and time. Few-shot learning (FSL) intuitively addresses this obstacle (Koch et al., 2015; Vinyals et al., 2016; Snell et al., 2017; Finn et al., 2017; Sung et al., 2018). FSL learns at the meta-task level, where each meta-task is formulated as inferring queries with the help of a support set (Vinyals et al., 2016). Multiple meta-tasks facilitate the task-agnostic transferrable knowledge. Thus it can learn new knowledge fast after being taught only a few samples. Despite

Reference: *This was the worst hotel in Vegas.* **Class:** *hotel*

Support set	
1) It was sold out due to there was the girls' basketball tournament taking place over the weekend.	*1*
2) The service is wonderful and the hotel is amazing.	*1*
3) Food was awesome and so was our waiter.	*0*
Query set	
1) Might be a good place, but if you need customer support, good luck!	*?*
2) Basic cable and wifi.	*?*
3) This place was designed really well, too.	*?*

Figure 1: Meta-task example in few-shot one-class text classification, where 1 denotes a positive instance and 0 denotes a negative one.

FSL has been well-studied, its one-class scenario (Frikha et al., 2021) is less investigated.

In this paper, following the one-class trait, we design each meta-task as a binary classification. It consists of a reference instance, a support set, and a query set (see Figure 1). The reference instance is one known sample of a class, which is exploited to tell whether an instance out of the support/query set belongs to the same class. Such purpose is consistent with sequence match, which also makes a decision for two sequences. Previous sequence match can mainly be categorized into two promising directions: *classical methods*, e.g. Siamese Network (Koch et al., 2015), Compare-Aggregate (CA) (Wang and Jiang, 2017), and *transformer-based method*, e.g. DistilBert (Sanh et al., 2019), BERT (Devlin et al., 2019).

In recent years, transformer models have already beaten classical ones in a wide range of tasks (Devlin et al., 2019). We wonder how two kinds of models perform under the few-shot one-class scenario. Consequently, it is presented that with meta-learning, classical sequence match method can significantly outperform transformer-based models. The classical models require much less training cost. Specifically, model-agnostic meta-learning

*Hang Gao is the corresponding author.

Proceedings of the 29th International Conference on Computational Linguistics, pages 4728–4740
October 12–17, 2022.

(MAML) algorithm (Finn et al., 2017) is a subtle bi-level optimizing approach that aims to learn a good parameter initialization. By introducing this algorithm, classical methods act as simple but competitive few-shot one-class learners.

Furthermore, we make an empirical comparison between classical and transformer-based models under simple fine-tuning and meta-learning. Firstly, it is found that MAML has a more positive impact on the both sequence match approaches than simple fine-tuning. This suggests that a good parameter initialization is important for both of them. Secondly, MAML tends to make transformer models extract features with high-correlation dimensions. The bi-level optimization might cause the feature extraction layers of large models less trained. Yet the last classifying layer tends to be better learned relatively. We demonstrate that the high correlation is related to the number of heads and layers in the transformer.

In summary, our main contributions are as follows: (1) We present a simple but competitive few-shot one-class learner, which is based on the classical sequence match approach and meta-learning. Extensive experimental results show that this learner achieves significant improvements compared with transformer-based models. This provides new insights in the transformer-dominant era. (2) Based on the testbed provided by the above approaches, an empirical study is made to further reveal their underlining natures. New observations and conclusions are derived.

2 Related Works

2.1 Few-Shot Learning

Few-shot learning (FSL) (Fei-Fei et al., 2006) deals with the practical problem of data scarcity in an intuitive way. It learns new knowledge fast with limited supervised information. An early work (Koch et al., 2015) learns to detect whether two instances belong to the same class. Later, matching network (Vinyals et al., 2016) proposes to construct multiple meta-tasks in both the training and testing procedures. This setting becomes mainstream in the subsequent works, to name a few, distance-based methods (Snell et al., 2017; Sung et al., 2018; Garcia and Bruna, 2018; Bao et al., 2020), optimization-based methods (Finn et al., 2017; Munkhdalai and Yu, 2017) or hallucination-based methods (Wang et al., 2018; Li et al., 2020). Among them, MAML (Finn et al., 2017) is special for "model-agnostic", indicating that this algorithm can be applied in any model. Therefore, we choose to further study its effects. To the best of our knowledge, it is seldom studied in transformer-based models. We provide an interesting empirical analysis in the experiments.

Recently, prompt-based fine-tuning (Gao et al., 2021) also become popular in FSL. It classifies a template-based instance through the masked language model. This prediction manner bridges the gap between pre-training and fine-tuning. Its effectiveness in the few-shot one-class scenario is under-explored.

2.2 One-Class Few-Shot Learning

Recently some works discuss the one-class problem in FSL. Cumulative LEARning (CLEAR) (Kozerawski and Turk, 2018) uses transfer learning to model the decision boundary of SVM. One-way proto (OWP) (Kruspe, 2019) is based on the prototypical network (Snell et al., 2017). OWP computes the positive prototype by simply averaging the representations of instances. It designs a 0-vector as the negative prototype. The Euclidean distance with prototypes in the embedding space indicates that an instance is positive or negative. One-class MAML (Frikha et al., 2021) proposes a simple data sampling strategy to ensure that the class-imbalance rate of the inner-level matches the test task. Different from them, we leverage the unique direction in natural language processing, i.e. sequence match, to study the one-class FSL.

2.3 Sequence Match

Sequence match aims to make a decision for two sequences. Many tasks require to match sequences, such as text entailment (Bowman et al., 2015), machine comprehension (Tapaswi et al., 2016), recommendation (Kraus and Feuerriegel, 2019), etc. A straightforward approach is to encode each sequence as a vector and then compare the two vectors to make a decision (Bowman et al., 2015; Feng et al., 2015). However, a single vector is insufficient to match the important information between two sequences. Thus attention mechanism is adopted in this task (Rocktäschel et al., 2016).

Later, the Compare-Aggregate framework is proposed (Wang and Jiang, 2017) for matching sequences, which has been widely studied. Its extended version usually considers the bidirectional information of two inputs (Bian et al., 2017; Yoon et al., 2019). One previous work (Ye and Ling,

2019) shows that matching and aggregation are effective in few-shot relation classification. We explore this framework in the few-shot one-class problem. More recently, pre-trained language models, e.g. BERT (Sanh et al., 2019) gain remarkable achievements in many sequence match tasks (Wang et al., 2020).

3 Methods

In this work, two kinds of sequence match methods, including classical and transformer-based ones, are mainly investigated. Compare-Aggregate (Wang and Jiang, 2017) is a promising classical method. We choose to study its extended version, i.e. Bidirectional Compare-Aggregate (BiCA) (Bian et al., 2017; Yoon et al., 2019), introduced in §3.2. The transformer-based sequence match (Sanh et al., 2019) is also presented in §3.3 briefly.

3.1 Problem Definition

Assume the training data $\mathcal{D}_{train}$ is composed of a set of training classes $\mathcal{C}_{train}$, and the testing data $\mathcal{D}_{test}$ has a set of classes $\mathcal{C}_{test}$, there are no overlapping between two class sets $\mathcal{C}_{train} \cap \mathcal{C}_{test} = \varnothing$. During training, we randomly sample a bunch of meta-tasks from $\mathcal{C}_{train}$. A meta-task is made as a binary classifier to detect one class, which is formulated as below.

A meta-task contains a reference sentence r, a support set S and a query set Q.

$$\begin{aligned} S &= \{(x_s^1, y_s^1), (x_s^2, y_s^2), ...(x_s^{|S|}, y_s^{|S|})\} \\ Q &= \{(x_q^1, y_q^1), (x_q^2, y_q^2), ...(x_q^{|Q|}, y_q^{|Q|})\} \end{aligned} \quad (1)$$

where x_s/x_q is an instance and y_s/y_q denotes whether this instance belongs to the same class as the reference. $|S|$ and $|Q|$ indicate the number of instances in two sets, respectively. Many meta-tasks enable the model to extract task-agnostic knowledge, which is beneficial to the meta-tasks from the testing classes $\mathcal{C}_{test}$.

3.2 Classical Sequence Match

In this section, we will introduce the components of Bidirectional Compare-Aggregate (BiCA) in detail.

Encoder Given an input sentence with L words, denoted as $\{w^1, w^2, ..., w^L\}$, it is first mapped into an embedding sequence $\mathbf{E} = \{\boldsymbol{e^1}, \boldsymbol{e^2}, ..., \boldsymbol{e^L}\}$ by looking up the pre-trained GloVe embeddings (Pennington et al., 2014). Then the embedding sequence is processed by the gate mechanism (Wang and Jiang, 2017) to obtain contextualized information. This gating mechanism aims at remembering the meaningful words and filtering the less-important words in a sentence.

$$\mathbf{H} = \sigma(\mathbf{W^i E} + \boldsymbol{b^i}) \odot \tanh(\mathbf{W^u E} + \boldsymbol{b^u}) \quad (2)$$

where $\mathbf{W^i}$ and $\mathbf{W^u}$ are parameter matrix, $\boldsymbol{b^i}$ and $\boldsymbol{b^u}$ are biases, $\odot$ is element-wise multiplication.

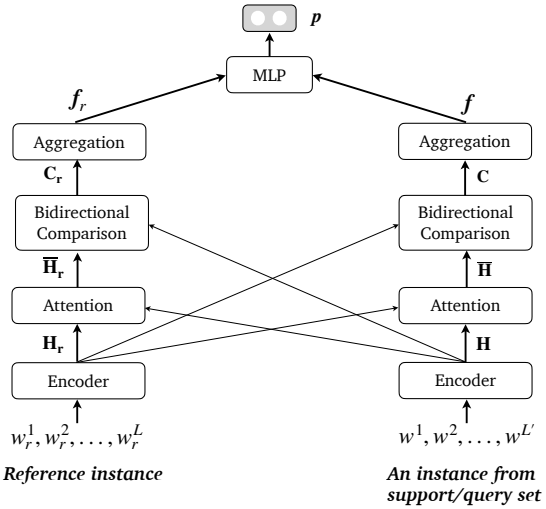

Figure 2: The network architecture of BiCA. The parameters are shared by two input instances.

Attention As depicted in Figure 2, the reference and an instance from support/query set are fed into the encoder, obtaining $\mathbf{H_r}$ and $\mathbf{H}$. Then the interaction between two inputs is computed through an attention mechanism.

$$\begin{aligned} \overline{\mathbf{H}}_\mathbf{r} &= \mathbf{H_r} \cdot \mathrm{softmax}(\mathbf{H_r}^\mathrm{T}\mathbf{H}) \\ \overline{\mathbf{H}} &= \mathbf{H} \cdot \mathrm{softmax}(\mathbf{H}^\mathrm{T}\mathbf{H_r}) \end{aligned} \quad (3)$$

This attention mechanism is non-parametric since it only depends on the encoded representations $\mathbf{H_r}$ and $\mathbf{H}$. Such design reduces the reliance on parameters and focuses on learning the relationships between data. Hence, this helps better adapt to unseen classes.

Bidirectional Comparison To compare the two instances, we adopt a simple word-level comparison function, i.e., element-wise multiplication $\odot$.

$$\mathbf{C_r} = \overline{\mathbf{H}}_\mathbf{r} \odot \mathbf{H} \qquad \mathbf{C} = \overline{\mathbf{H}} \odot \mathbf{H_r} \quad (4)$$

The comparison function is also non-parametric for the purpose of adaptation. As shown in Figure 2, the encoded representations are applied in both the attention and bidirectional comparison modules to promote the mutual interaction of two inputs.

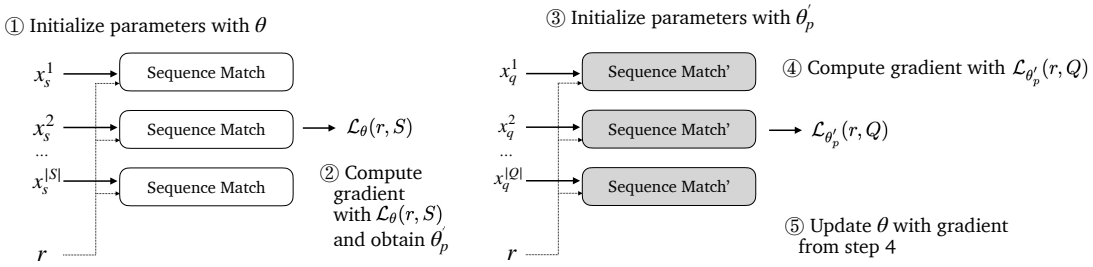

Figure 3: Introducing meta-learning to sequence match approaches. The left part and the right part share the same architecture but employ different parameters in the meta-learning. The five steps are corresponding to the Algorithm 1. After training, the optimal parameter initialization θ for the testing classes is obtained.

Aggregation Following the original work (Wang and Jiang, 2017), the comparison representations are aggregated by convolution neural network (CNN) (Kim, 2014). The convolution kernel slides over the comparison sequence to extract n-gram features, which tends to be helpful in matching.

$$\boldsymbol{f_r} = \mathrm{CNN}(\mathbf{C_r}) \qquad \boldsymbol{f} = \mathrm{CNN}(\mathbf{C}) \tag{5}$$

where $\mathrm{CNN}(\cdot)$ is a convolution operation followed by max-pooling. Then the matching score is computed with the aggregation representations of two input sentences.

$$\boldsymbol{p} = \mathrm{MLP}([\boldsymbol{f_r}, \boldsymbol{f}]) \tag{6}$$

where MLP is a single linear layer. $\boldsymbol{p}$ is a two-dimensional logits output.

Loss The training objective of BiCA is the cross entropy loss.

$$\mathcal{L}_\theta(r, S) = -\frac{1}{|S|}\sum_{|S|} \log P(y|\boldsymbol{p}) \tag{7}$$

where y is the ground truth. θ represents all the parameters in the sequence match model.

3.3 Transformer-Based Sequence Match

Transformer-based models, e.g. BERT (Devlin et al., 2019) are pre-trained on a large-scale corpus, serving as foundational backbones for a wide range of natural language processing tasks. When aiming at sequence match, BERT utilizes two special tokens [CLS] and [SEP] to concatenate two sequences as a whole. For the two input instances shown in Figure 2, they are combined into $\{[\texttt{CLS}], w_r^1, w_r^2, ..., w_r^L, [\texttt{SEP}], w^1, w^2, ..., w^{L'}\}$, where the output of [CLS] is usually for classification and [SEP] is for separating two inputs. This combined sequence is then fed into BERT. The self-attention mechanism (Vaswani et al., 2017) in the transformer will compute the interaction between two inputs. Finally, the output of the first token [CLS] is adopted for inference. The training objective is also computed by Eq. (7).

Algorithm 1: Meta-Learning for Few-Shot One-Class Problem

Input: Training data from $\mathcal{D}_{train}$
1 Randomly initialize θ
2 **repeat**
3 **Sample** positive classes $\mathcal{C}_p$ and negative classes $\mathcal{C}_n$ from $\mathcal{D}_{train}$
4 $\mathcal{C}_p \cap \mathcal{C}_n = \varnothing$
5 **for** *all* $\mathcal{C}_p$ **do**
6 Construct a meta-task
7 Evaluate $\nabla_\theta \mathcal{L}_\theta(r, S)$ in Eq. (7)
8 Compute adapted parameters with gradient descent: $\theta_p' = \theta - \alpha \nabla_\theta \mathcal{L}_\theta(r, S)$
9 Update $\theta \leftarrow \theta - \beta \nabla_\theta \sum_{\mathcal{C}_p} \mathcal{L}_{\theta_p'}(r, Q)$
10 **until** performance on the validation data set does not improve in 3 epochs.

3.4 Meta-Learning for Sequence Match

In the few-shot one-class paradigm, a meta-task from the unseen class has a few labeled instances as the support set. To better leverage such knowledge, we introduce meta-learning to sequence match models, which is displayed in Figure 3 and Algorithm 1. Specifically, model-agnostic meta-learning (MAML) algorithm (Finn et al., 2017) is chosen to investigate its impact on the sequence match approaches. This algorithm learns a good initialization of model parameters by maximizing the sensitivity of the loss function when adapting to new tasks (Song et al., 2020).

Construct a Meta-Task In Algorithm 1 (line 6), given a positive class, we first sample $N + 1$

		Train	Validation	Test
Class Num		64	16	20
Data Num	single	13677	3394	4671
	multi	26643	6686	7929
Data Num/Class	single	213.7	212.1	233.5
	multi	416.2	417.8	396.4

Table 1: Dataset statistics for ACD. *single* denotes the single-aspect sentence and *multi* denotes the multi-aspect one.

instances from this class, which constitutes a reference instance r and N positive ones. Moreover, N negative examples are also sampled from the negative classes $\mathcal{C}_n$. These positive and negative examples are mixed up randomly, which are further divided into the support set and query set.

Meta-learning trains in a bi-level way (see Algorithm 1), including the inner-level (line 8) and the outer-level (line 9) for the support set S and the query set Q, respectively. This way will cause the gradients for updating parameters to propagate through more layers (line 9), i.e. twice as many as the number of network layers in a sequence match model. Its special effects on the transformer models are further discussed in §4.4. When evaluating, a model is initialized from the parameters θ trained by MAML on the training classes, which is then optimized with the support set on the testing classes.

4 Experiments

4.1 Datasets

Aspect Category Detection (ACD) A dataset for few-shot one-class ACD is collected from *YelpAspect* (Bauman et al., 2017; Li et al., 2019), which is a large-scale multi-domain dataset for fine-grained sentiment analysis. The 100 aspect categories are split without intersection into 64 classes for training, 16 classes for validation, and 20 classes for testing. Table 1 displays the statistics of the dataset. The data in each class is further divided according to the number of the aspects in a sentence, into single-aspect and multi-aspect. Relatively, a multi-aspect example contains more noise when matching sequences. To explore a challenging scenario, the support/query set are both sampled from the multi-aspect set. Fixing this setup, we choose a reference instance as a single- and multi-aspect one.

HuffPost It consists of news headlines published in HuffPost between 2012 and 2018 (Misra, 2018). Bao et al. (2020) process the original dataset for few-shot text classification. The number of training, validation, and testing classes are 20, 5, and 16, respectively, where each class has 900 instances. Since the sentences are headlines, they are shorter and less grammatical.

4.2 Baseline Methods

Matching sequences at the vector-level:

SN (Koch et al., 2015) Siamese network can capture discriminative features to generalize the predictive power of the network. The input instances are extracted into two vectors, which are compared with *cosine similarity*.

OWP (Kruspe, 2019) One-way prototypical network designs a 0-vector as the negative prototype. It measures the *Euclidean distance* between an instance with the positive/negative prototypes in the embedding space.

Matching sequences at the word-level:

CA (Wang and Jiang, 2017) Compare-Aggregate is widely used to match the important units between sequences. It only compares in one direction, i.e., reference-to-candidate.

BiCA CA is enhanced into matching sequences bidirectionally (§3.2).

DistilBert (Sanh et al., 2019) It is a distilled version of BERT (§3.3).

BERT (Devlin et al., 2019) It is transformer-based and matching sequences at word-level (§3.3).

BERT(p) (Gao et al., 2021) It trains BERT with prompt-based learning. The two sequences are concatenated with "?[MASK]," like Gao et al.. The representation of [MASK] is mapped into word *"yes"* or *"no"*, suggesting that two sequences belong to the same class or not.

4.2.1 Implementation Details

Baseline methods are trained with naive training, which learns the training classes in a meta-task manner, but combines the support set and query set as a whole to optimize parameters. **+finetune** means that the naive trained models are fine-tuned. **+MAML** indicates the models are optimized by MAML (Algorithm 1). During the evaluation, +finetune or +MAML exploit the support set of testing classes in the same way, with the same number of updating steps and learning rate. Thus the only difference between +finetune and +MAML is the parameters are initialized by naive training or MAML training. All testing meta-tasks share the

Model	Use support set of $\mathcal{D}_{test}$	Match Type vector	Match Type word	ACD *Single* Acc	ACD *Single* F1	ACD *Multi* Acc	ACD *Multi* F1	HuffPost Acc	HuffPost F1
SN	No	✓		69.88±1.19	69.33±1.16	72.12±0.82	71.64±0.82	62.61±0.55	61.87±0.56
OWP	No	✓		72.50±1.22	71.96±1.23	70.94±0.58	70.24±0.65	61.72±0.72	61.05±0.72
CA	No		✓	79.45±1.17	78.97±1.25	76.72±0.99	76.27±0.96	64.02±0.36	63.33±0.47
BiCA	No		✓	79.46±0.39	79.03±0.46	76.81±0.89	76.40±0.92	64.72±0.77	64.20±0.76
DistilBert	No		✓	79.32±1.19	78.87±1.36	75.16±0.94	74.62±0.97	64.87±1.32	63.91±1.79
BERT	No		✓	79.05±0.98	78.61±0.98	74.62±0.97	74.03±1.03	66.02±1.22	65.24±1.34
BERT(p)	No		✓	74.99±5.22	73.73±6.67	76.58±0.90	76.07±0.92	64.67±0.58	63.52±1.26
BERT(p)+finetune	Yes		✓	83.53±1.11	83.30±1.19	82.74±0.73	82.53±0.75	67.43±1.06	66.64±1.22
BERT+funetune	Yes		✓	86.10±0.76	85.97±0.76	82.63±0.77	82.43±0.78	73.02±0.70	72.73±0.69
BERT+MAML	Yes		✓	88.33±2.76	88.23±3.07	84.99±2.86	84.86±3.18	73.89±3.28	73.66±3.37
DistilBert+finetune	Yes		✓	84.62±1.21	84.44±1.26	82.15±0.57	81.93±0.62	69.68±0.83	69.22±0.92
DistilBert+MAML	Yes		✓	87.73±0.66	87.61±0.67	84.93±0.79	84.76±0.81	72.22±1.60	72.00±1.62
BiCA+finetune	Yes		✓	84.62±0.38	84.46±0.39	82.84±0.97	82.70±0.98	65.82±0.85	65.48±0.89
BiCA+MAML	Yes		✓	**89.86**† **±0.65**	**89.76**† **±0.66**	**89.80**† **±0.56**	**89.70**† **±0.57**	**74.47**‡ **±1.68**	**74.20**‡ **±1.68**

Table 2: Experimental results for ACD and HuffPost in terms of accuracy(%) and macro-f1(%). We report the average and standard deviation of 5 runs. *Single* indicates that the reference instance is single-aspect. *Multi* indicates setting references as multi-aspect. The marker † refers to p-value$<$0.01 of the T-test compared with DistilBert+MAML. The marker ‡ refers to p-value$<$0.07 of the T-test compared with DistilBert+MAML.

same parameter initialization without mutual interference. The implementation details are described in the Appendix.

4.3 Experimental Results

The experimental results on ACD and HuffPost datasets are displayed in Table 2. The first part in Table 2 shows the case that when testing, we only have the reference instance but do not use the support set. By comparing two match types, we find that a finer-granularity matching helps the few-shot one-class scenario gain significantly. This indicates that in the few-shot scenario of text tasks, learning deeper interaction between instances is a better choice. Many previous tasks gain significantly from BERT (Wang et al., 2020) or DistilBert (Wright and Augenstein, 2020). However, we surprisingly see little performing difference among the five word-level sequence match methods. The transformer-based methods do not have remarkable superiority. A possible reason is that in the unseen classes, it is difficult to discover the *key words/semantics* for matching only given a reference instance. Meanwhile, these models with large-scale parameters may be superior in the data-driven tasks (Gururangan et al., 2020).

Additionally, though the scale of the support set is small, exploiting it by fine-tuning or MAML can bring significant improvements. This also explains our previous guess that the support set will provide *key words/semantics* to match. Meanwhile, on sequence match methods, including BiCA, BERT and DistilBert, MAML outperforms fine-tuning in most situations. This indicates the importance of a good parameter initialization not only for small models but also for large pre-trained models in a few-shot problem.

It is also found that prompt-based fine-tuning, i.e. BERT(p), is also less-performed than BiCA+MAML. The possible reasons are: first, the objective of one-class sequence match is not consistent with the pre-training of language models. Thus the knowledge of transformer models might not be fully leveraged; second, prompt-based fine-tuning may achieve better results by other huge-scale pre-trained models, such as RoBERTa-large, GPT-3 (Gao et al., 2021).

Finally, it is worth noting that BiCA+MAML consistently outperforms BERT+MAML and DistilBert+MAML. Compared with transformer models, BiCA+MAML has fewer parameters, suggesting that the classical methods are still worth revisiting in the large pre-trained models' dominant era. We further see another interesting phenomenon. BiCA gains significantly from MAML but slightly improves by using fine-tuning. Contrarily, the transformer-based method gains much from fine-tuning. It is possible that the pre-trained BERT already contains abundant knowledge, suggesting a good initialization for fine-tuning. Meanwhile, BiCA is a classical model with much fewer parameters, which is easier for MAML to learn a good initialization. Hence, MAML has a more significant contribution to BiCA.

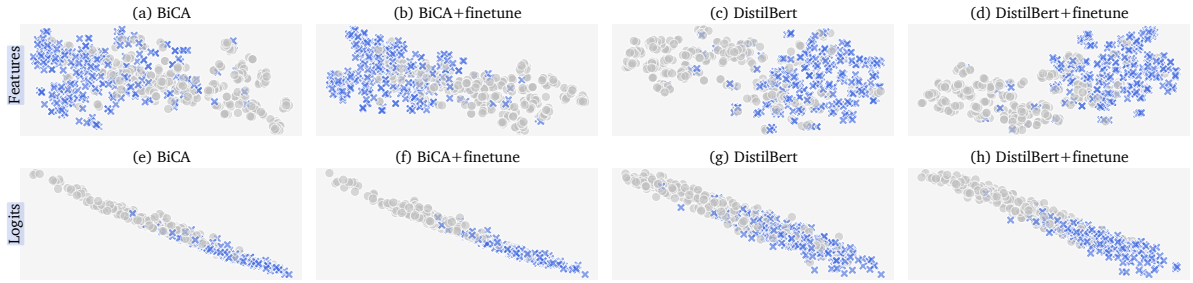

Figure 4: Effects of fine-tuning on BiCA and DistilBert for ACD, where the reference instance is single-aspect. For a fair comparison, all models only have one linear output layer. In the top row, we depict PCA plots of the features before the output layer. The feature dimension in BiCA is 500 and which in DistilBert is 768. In the bottom row, we directly plot the 2-dimensional logits output.

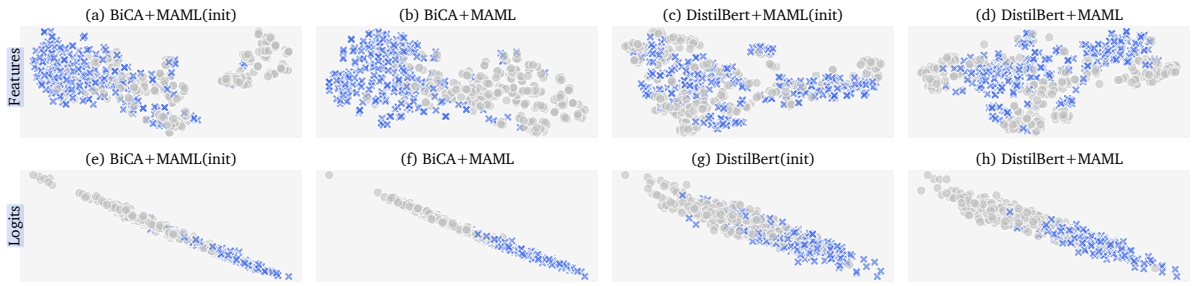

Figure 5: Effects of MAML on BiCA and DistilBert for ACD, where the reference instance is single-aspect. "(init)" indicates that the model learned by MAML training directly predicts the query set of testing meta-tasks without exploiting the support set.

4.4 Discussion

In this section, an empirical comparison, between two kinds of sequence match approaches, including classical and transformer-based ones, is presented. Because DistilBert+MAML and BERT+MAML are comparable, as shown in Table 2. Meanwhile, the parameter scale of DistilBert is smaller, which is chosen in the following study. We randomly sample 12 batches from the testing classes, and obtain the extracted features of the query set. In each batch, we have 5 meta-tasks, each of them has 10 support instances and 10 query instances. Thus, the total number of features is $5 \times 10 \times 12 = 600$. The features are visualized by t-SNE (Maaten and Hinton, 2008).

Model	ACD		HuffPost
	Single	*Multi*	
BiCA	1.74e-3	1.23e-3	2.66e-4
BiCA+finetune	1.77e-3	1.22e-3	2.84e-4
BiCA+MAML(init)	1.79e-3	1.67e-3	3.53e-4
BiCA+MAML	**1.92e-3**	**2.09e-3**	**5.85e-4**
DistilBert	3.35e-3	2.70e-3	1.51e-3
DistilBert+finetune	4.01e-3	3.39e-3	1.78e-3
DistilBert+MAML(init)	8.57e-3	1.15e-2	7.40e-3
DistilBert+MAML	**8.98e-3**	**1.32e-2**	**7.76e-3**
BERT	3.89e-3	3.91e-3	4.62e-3
BERT+finetune	4.83e-3	5.56e-3	6.43e-3
BERT+MAML(init)	2.14e-1	1.71e-1	**2.63e-1**
BERT+MAML	**2.21e-1**	**1.99e-1**	2.58e-1

Table 3: Cov_Score of various models. The largest is marked in bold for each sequence match model.

4.4.1 Comparison of Features

We compare the effects of MAML with simple fine-tuning in Figure 4 and Figure 5, respectively. In Figure 4, it can be seen that fine-tuning can help the features learned by BiCA and DistilBert both become more separable (plot a-d). This is also reflected by the 2-dimensional logits output (e-g).

In Figure 5, it can be observed that in MAML training, exploiting the support set can also make the features more discriminative in BiCA (a-b), and so does the logits output (e-f). The features (plot b in Figure 4 and plot b in Figure 5) validates that MAML is more effective than fine-tuning in BiCA.

Interestingly, the phenomenon of DistilBert+MAML is completely different. It is found that the features show less separability (c-d), while the logits output is well distinguished (g-h). This indicates the features are linearly separable in high dimensions, i.e. 768. Recalling the purpose of PCA (Principal Component Analysis) (Abdi

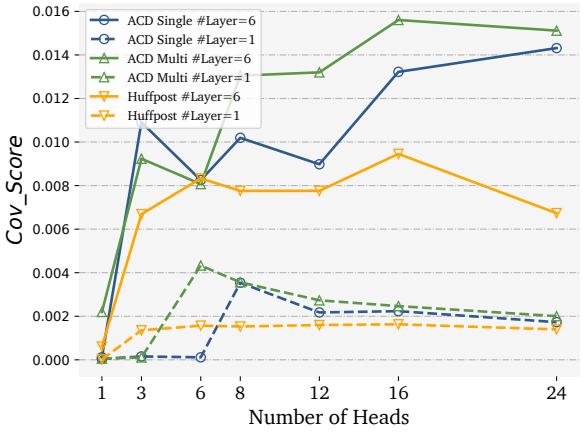

Figure 6: Cov_Score of the features learned by DistilBert+MAML, setting different numbers of layers, and numbers of heads in self-attention.

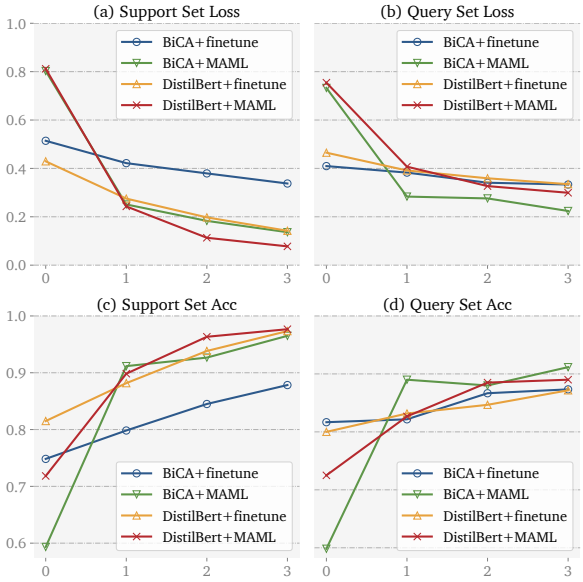

Figure 7: Loss and accuracy curves in 3 updating steps.

and Williams, 2010), it defines an orthogonal linear transformation that transforms the data into a new coordinate system and preserves the greatest variance. We guess that the unique phenomena (c-d) are caused by less-orthogonal feature dimensions. To verify this assumption, we compute the covariance matrix based on extracted features in various models. Each element in the matrix indicates the correlation between two dimensions in feature, where a larger score means a higher correlation. We define the Cov_Score as below, which is the average absolute value of the covariance matrix.

$$Cov\_Score = \text{avg}|\text{Cov}(\mathbf{F} - \text{rowavg}(\mathbf{F}))| \quad (8)$$

where $\mathbf{F}$ is the extracted features.

In Table 3, the Cov_Score of three sequence match approaches are presented separately. Firstly, compared with BiCA+finetune, BiCA+MAML extracts features with slightly higher Cov_Score. However, for DistilBert and BERT, MAML dramatically increases the score, which is more significant in BERT. As the scores indicate, DistilBert+MAML really extracts features with less-orthogonal dimensions. A possible reason is that MAML trains the model in a bi-level manner (see Algorithm 1). DistilBert has deeper layers and large-scale parameters. The gradients are propagated through deeper layers, causing the feature extraction learned insufficiently while the last linear layer is trained adequately. Thus the logits are separable while features are not, as plots (c, d, g, h) shown in Figure 5. The above phenomenon become also serious in BERT+MAML, because BERT has more layers compared with DistilBert, leading to larger Cov_Score.

The plots of other datasets are displayed in the Appendix. The empirical observations are similar.

4.4.2 DistilBert+MAML: Layers and Heads?

To further investigate why the features learned by DistilBert+MAML have high-correlation dimensions, we plot the Cov_Score of multiple variants of DistilBert+MAML in Figure 6. The horizontal ordinate indicates the number of heads in self-attention, which should divide 768 exactly (768 is the dimension size of the hidden states in DistilBert and BERT models).

It is first seen that the score shows an increasing trend as the number of heads grows. Meanwhile, by comparing the curves between #Layer=1 and #Layer=6, we observe that the Cov_Score also becomes larger as the layers of DistilBert increase. We draw an empirical conclusion that the high-correlation of feature dimensions in DistilBert+MAML is caused by the multiple layers and heads of DistilBert.

4.4.3 Initialization for Loss Sensitivity or a Good Performance Start

In Figure 7, we depict the average batch loss and accuracy in the 3 update steps for ACD with single-aspect references. Step 0 indicates the model is trained by naive training or MAML without using the testing support set.

Concretely, it can be seen that for both BiCA and DistilBert, MAML leads to faster loss degradation than fine-tuning (plot a). The loss in MAML

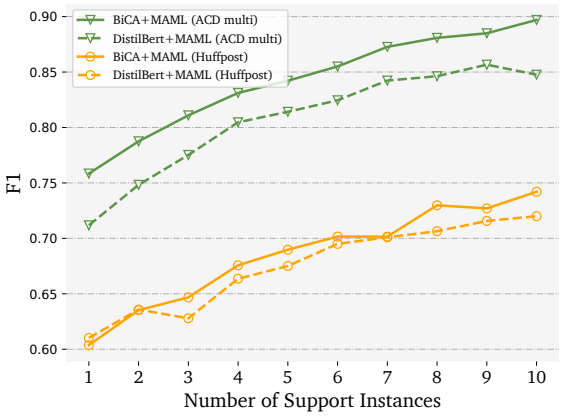

Figure 8: F1 scores of BiCA+MAML and DistilBert+MAML by setting different numbers of support instances.

declines to the bottom even within one updating step. We also observe a rapid accuracy increase (c). Overall, MAML outperforms fine-tuning for both BiCA and DistilBert. However, is MAML always a good choice? The answer is negative. In step 0 (b and d), we see a good parameter initialization does not lead to a good performance start. BiCA and DistilBert learned by naive training achieve lower losses and better accuracy scores without updating parameters. A possible explanation is that they have different training objectives. Naive training aims to facilitate task-agnostic sequence matching. While the bi-level optimization in MAML focuses on promoting the generalization ability of models after seeing a few support examples.

4.4.4 Various Numbers of Support Instances

We further explore the performances of BiCA+MAML and DistilBert+MAML under various support set scales. As displayed in Figure 8, the number of support instances ranges from 1 to 10. For a fair comparison, the number of query instances is all set to 10. Firstly, it can be seen that as the support set scale grows, the performances on the query set present an increasing trend. Secondly, we also observe that BiCA+MAML outperforms DistilBert+MAML in most experimental settings. This indicates that classical sequence match is a competitive few-shot one-class learner.

5 Conclusion

In this work, we revisit the classical sequence match approaches and find that with meta-learning, the classical method can significantly outperform transformer models in the few-shot one-class scenario. The training cost is greatly reduced. Furthermore, an empirical study is made to explore the effects of simple fine-tuning and meta-learning. Interestingly, although meta-learning is more effective than simple fine-tuning on both sequence match approaches, it makes the transformer features have high correlation dimensions. The correlation is closely related to the number of layers and heads in the transformer models. We hope this work could provide insights for future research on few-shot problems and transformer models.

Acknowledgements

We sincerely thank all the anonymous reviewers for providing valuable feedback. This work is supported by the key program of National Science Fund of Tianjin, China (Grant No. 21JCZDJC00130) and the Basic Scientific Research Fund, China (Grant No. 63221028).

References

Hervé Abdi and Lynne J Williams. 2010. Principal component analysis. *Wiley interdisciplinary reviews: computational statistics*, 2(4):433–459.

Yujia Bao, Menghua Wu, Shiyu Chang, and Regina Barzilay. 2020. Few-shot text classification with distributional signatures. In *International Conference on Learning Representations (ICLR)*, pages 1–24.

Konstantin Bauman, Bing Liu, and Alexander Tuzhilin. 2017. Aspect based recommendations: Recommending items with the most valuable aspects based on user reviews. In *Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD)*, page 717–725.

Weijie Bian, Si Li, Zhao Yang, Guang Chen, and Zhiqing Lin. 2017. A compare-aggregate model with dynamic-clip attention for answer selection. In *Proceedings of the 2017 ACM on Conference on Information and Knowledge Management (CIKM)*, pages 1987–1990.

Samuel Bowman, Gabor Angeli, Christopher Potts, and Christopher D Manning. 2015. A large annotated corpus for learning natural language inference. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 632–642.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL)*, pages 4171–4186.

Li Fei-Fei, Rob Fergus, and Pietro Perona. 2006. One-shot learning of object categories. *IEEE transactions on pattern analysis and machine intelligence (TPAMI)*, 28(4):594–611.

Minwei Feng, Bing Xiang, Michael R Glass, Lidan Wang, and Bowen Zhou. 2015. Applying deep learning to answer selection: A study and an open task. In *2015 IEEE Workshop on Automatic Speech Recognition and Understanding (ASRU)*, pages 813–820.

Chelsea Finn, Pieter Abbeel, and Sergey Levine. 2017. Model-agnostic meta-learning for fast adaptation of deep networks. In *International conference on machine learning (ICML)*, pages 1126–1135.

Ahmed Frikha, Denis Krompaß, Hans-Georg Koepken, and Volker Tresp. 2021. Few-shot one-class classification via meta-learning. In *The 35th AAAI conference on Artificial Intelligence (AAAI)*.

Tianyu Gao, Adam Fisch, and Danqi Chen. 2021. Making pre-trained language models better few-shot learners. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (ACL-IJCNLP)*, pages 3816–3830.

Victor Garcia and Joan Bruna. 2018. Few-shot learning with graph neural networks. In *International Conference on Learning Representations (ICLR)*.

Suchin Gururangan, Ana Marasović, Swabha Swayamdipta, Kyle Lo, Iz Beltagy, Doug Downey, and Noah A. Smith. 2020. Don't stop pretraining: Adapt language models to domains and tasks. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics (ACL)*, pages 8342–8360.

Yoon Kim. 2014. Convolutional neural networks for sentence classification. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1746–1751.

Gregory Koch, Richard Zemel, and Ruslan Salakhutdinov. 2015. Siamese neural networks for one-shot image recognition. In *ICML deep learning workshop*, pages 1–8.

Jedrzej Kozerawski and Matthew Turk. 2018. Clear: Cumulative learning for one-shot one-class image recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 3446–3455.

Mathias Kraus and Stefan Feuerriegel. 2019. Personalized purchase prediction of market baskets with wasserstein-based sequence matching. In *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining (KDD)*, pages 2643–2652.

Anna Kruspe. 2019. One-way prototypical networks. *arXiv preprint arXiv:1906.00820*.

Kai Li, Yulun Zhang, Kunpeng Li, and Yun Fu. 2020. Adversarial feature hallucination networks for few-shot learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 13470–13479.

Zheng Li, Ying Wei, Yu Zhang, Xiang Zhang, and Xin Li. 2019. Exploiting coarse-to-fine task transfer for aspect-level sentiment classification. In *Proceedings of the AAAI Conference on Artificial Intelligence (AAAI)*, volume 33, pages 4253–4260.

Laurens van der Maaten and Geoffrey Hinton. 2008. Visualizing data using t-sne. *Journal of machine learning research*, 9(Nov):2579–2605.

Rishabh Misra. 2018. News category dataset.

Tsendsuren Munkhdalai and Hong Yu. 2017. Meta networks. In *International Conference on Machine Learning (ICML)*, pages 2554–2563.

Jeffrey Pennington, Richard Socher, and Christopher Manning. 2014. Glove: Global vectors for word representation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1532–1543.

Tim Rocktäschel, Edward Grefenstette, Karl Moritz Hermann, Tomáš Kočiskỳ, and Phil Blunsom. 2016. Reasoning about entailment with neural attention. In *International Conference on Learning Representations (ICLR)*, pages 1–9.

Victor Sanh, Lysandre Debut, Julien Chaumond, and Thomas Wolf. 2019. Distilbert, a distilled version of bert: smaller, faster, cheaper and lighter. *The 5th Workshop on Energy Efficient Machine Learning and Cognitive Computing - NeurIPS*, pages 1–5.

Jake Snell, Kevin Swersky, and Richard Zemel. 2017. Prototypical networks for few-shot learning. In *Advances in Neural Information Processing Systems (NeurIPS)*, pages 4077–4087.

Yiping Song, Zequn Liu, Wei Bi, Rui Yan, and Ming Zhang. 2020. Learning to customize model structures for few-shot dialogue generation tasks. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics (ACL)*, pages 5832–5841.

Flood Sung, Yongxin Yang, Li Zhang, Tao Xiang, Philip HS Torr, and Timothy M Hospedales. 2018. Learning to compare: Relation network for few-shot learning. In *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1199–1208.

Makarand Tapaswi, Yukun Zhu, Rainer Stiefelhagen, Antonio Torralba, Raquel Urtasun, and Sanja Fidler. 2016. Movieqa: Understanding stories in movies through question-answering. In *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition (CVPR)*, pages 4631–4640.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Lukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in Neural Information Processing Systems (NeurIPS)*.

Oriol Vinyals, Charles Blundell, Timothy Lillicrap, Daan Wierstra, et al. 2016. Matching networks for one shot learning. In *Advances in Neural Information Processing Systems (NeurIPS)*, pages 3630–3638.

Shuohang Wang and Jing Jiang. 2017. A compare-aggregate model for matching text sequences. In *International Conference on Learning Representations (ICLR)*.

Shuohang Wang, Yunshi Lan, Yi Tay, Jing Jiang, and Jingjing Liu. 2020. Multi-level head-wise match and aggregation in transformer for textual sequence matching. In *Proceedings of the AAAI Conference on Artificial Intelligence (AAAI)*, pages 9209–9216.

Yu-Xiong Wang, Ross Girshick, Martial Hebert, and Bharath Hariharan. 2018. Low-shot learning from imaginary data. In *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition (CVPR)*, pages 7278–7286.

Dustin Wright and Isabelle Augenstein. 2020. Transformer based multi-source domain adaptation. In *(EMNLP)*, pages 7963–7974.

Zhi-Xiu Ye and Zhen-Hua Ling. 2019. Multi-level matching and aggregation network for few-shot relation classification. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics (ACL)*, pages 2872–2881.

Seunghyun Yoon, Franck Dernoncourt, Doo Soon Kim, Trung Bui, and Kyomin Jung. 2019. A compare-aggregate model with latent clustering for answer selection. In *Proceedings of the 28th ACM International Conference on Information and Knowledge Management (CIKM)*, pages 2093–2096.

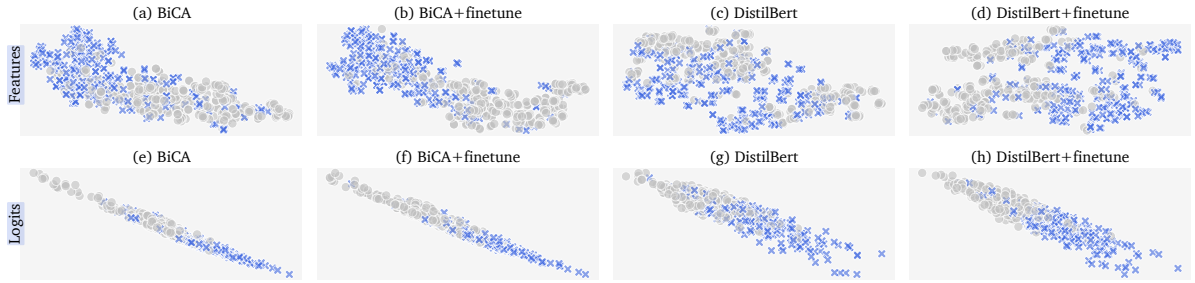

Figure 9: Effects of fine-tuning on aspect category detection, where the reference instance is multi-aspect.

Figure 10: Effects of MAML on aspect category detection, where the reference instance is multi-aspect.

A Additional Experimental Results

The visualizations of aspect category detection with multi-aspect reference instance are displayed in Figure 9 and Figure 10.

For HuffPost dataset, the visualizations are displayed in Figure 11 and 12.

B Reproducibility

B.1 Computing Infrastructure

All experiments are conducted on the same hardware and software. We use a single NVIDIA A6000 GPU with 48GB of RAM.

B.2 Average Running time

The average running time of each model is shown in Table 5.

B.3 Number of Parameters

The number of parameters of each model is shown in Table 4.

B.4 Datasets

The datasets are available at `https://github.com/hmt2014/FewOne`.

B.5 Implementation Details

B.5.1 Classical Methods

All baselines and our model are implemented by Pytorch. We initialize word embeddings with 50-dimension GloVE vectors (Pennington et al., 2014). In the aggregation module, the channels of the CNNs for input and output are both 50. The kernel sizes of five CNNs are [1, 2, 3, 4, 5], respectively. Relu is the activation function for CNN. We adopt a dropout of 0.1 after both the comparison and CNN in aggregation. MLP is a single linear layer.

The batch size is $|\mathcal{C}_p| = 5$, indicating a batch comprises 5 meta-tasks. The instance number in the support set $|S|$ and query set $|Q|$ are both set to 10. Every epoch we randomly sample 400 batches for training, 300 batches for validation and 300 batches for testing. The average results of the testing batches are reported. We exploit an early stop strategy during training if the macro-f1 score on the validation set does not improve in 3 epochs, and the best model is chosen for evaluation.

We describe the training details as the format of (optimizer, learning rate, other information):

Naive Training: Adam, 1e-3, early stop.

+finetune SGD, 0.1, 3 updating steps.

+MAML

The inner-level: α=0.1.

The outer-level: Adam, β=1e-3.

When testing: SGD, 0.1, 3 updating steps.

B.5.2 Transformer-based Methods

The output hidden state of [CLS] in DistilBert (distilbert-base-uncased) and BERT (bert-base-uncased) is exploited for classification.

Naive Training Adam, 2e-5, 5 epochs.

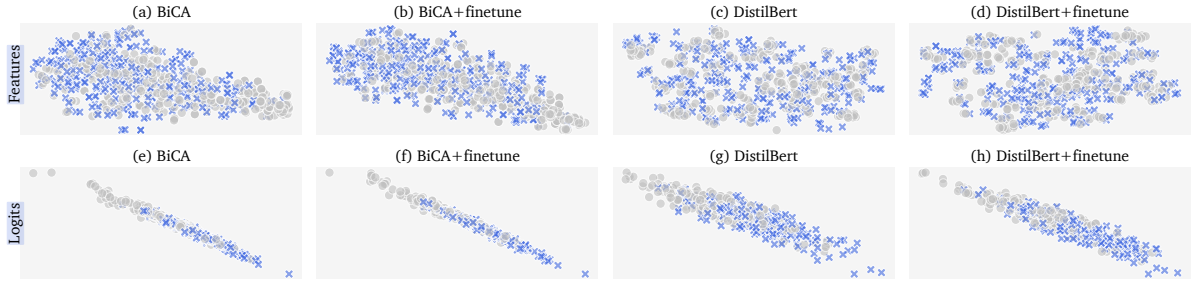

Figure 11: Effects of fine-tuning on HuffPost.

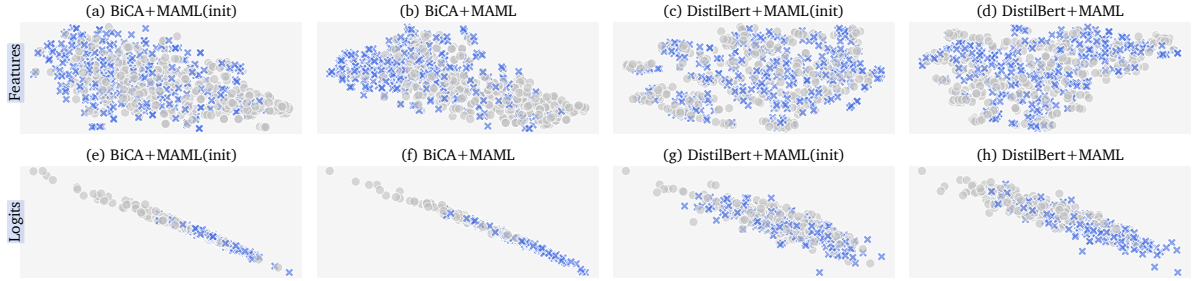

Figure 12: Effects of MAML on HuffPost.

Method	ACD	Huffpost
SN	1,437,950	1,393,050
OWP	1,437,950	1,393,050
CA	1,476,202	1,431,302
BiCA +finetune +MAML	1,476,702	1,431,802
DistilBert +finetune +MAML	66,364,418	66,364,418
BERT +finetune +MAML	109,483,778	109,483,778
BERT(p) +finetune	133,545,786	133,545,786

Table 4: Number of parameters in each model.

+finetune Adam, 2e-5, 3 updating steps.

+MAML

The inner level: α=2e-3.

The outer-level: Adam, β=2e-5.

When testing: Adam, 2e-5, 3 updating steps.

We use early stop for DistilBert+MAML since the model does not learn optimally within 5 epochs.

Method	ACD		Huffpost
	single	multi	
SN	6m47s	11m44s	20m5s
OWP	13m57s	17m49s	11m45s
CA	21m36s	35m17s	18m2s
BiCA	22m26s	18m5s	11m11s
+finetune	20s	20s	23s
BiCA+MAML	39m8s	1h11m57s	40m53s
DistilBert	21m2s	21m26s	21m56s
+finetune	4m45s	4m43s	4m45s
DistilBert+MAML	4h32m2s	4h3m38s	4h46m4s
BERT	39m33s	40m18s	41m27s
+finetune	8m40s	8m41s	9m14s

Table 5: Average runing time of each model.

Unsupervised Domain Adaptation for Text Classification via Meta Self-Paced Learning

Nghia Ngo Trung[1], **Linh Ngo Van**[2] **and Thien Huu Nguyen**[1]
[1] Department of Computer and Information Science, University of Oregon, Eugene, OR, USA
[2] Hanoi University of Science and Technology, Vietnam
{nghian@,thien@cs}.uoregon.edu, linhnv@soict.hust.edu.vn

Abstract

A shift in data distribution can have a significant impact on performance of a text classification model. Recent methods addressing unsupervised domain adaptation for textual tasks typically extracted domain-invariant representations through balancing between multiple objectives to align feature spaces between source and target domains. While effective, these methods induce various new domain-sensitive hyperparameters, thus are impractical as large-scale language models are drastically growing bigger to achieve optimal performance. To this end, we propose to leverage meta-learning framework to train a neural network-based self-paced learning procedure in an end-to-end manner. Our method, called **M**eta **S**elf-**P**aced **D**omain **A**daption (**MSP-DA**), follows a novel but intuitive domain-shift variation of cluster assumption to derive the meta train-test dataset split based on the self-pacing difficulties of source domain's examples. As a result, MSP-DA effectively leverages self-training and self-tuning domain-specific hyperparameters simultaneously throughout the learning process. Extensive experiments demonstrate our framework substantially improves performance on target domains, surpassing state-of-the-art approaches. Detailed analyses validate our method and provide insight into how each domain affects the learned hyperparameters.

1 Introduction

Given enough supervision, modern deep learning models can learn a new task with great accuracy. However, in many practical settings, the goal is to adapt to a new domain in which there is a different in data distribution between training and testing processes. This poses a major challenge for standard natural language systems due to both the intrinsic variation of linguistics (e.g., lexical shift, semantic shift) as well as the extrinsic factors such as how textual datasets are collected and annotated. For example, a model trained to predict news events may easily recognize, from medical domain, "*died*" as an event, but would not be able to detect obvious events such as "*mutation*" or "*cancer*". Such a model may even fail to generalize to closer adaptation settings (e.g. news from different times and sources).

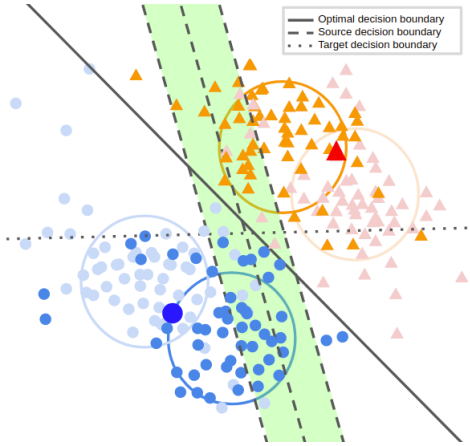

Figure 1: An example where domain shift between source domain (grey colors) and target domain (deep color) results in significant overlaps between high-loss regions of source decision boundary (lime) with high-density target clusters.

The majority of existing unsupervised domain adaptatiopn (UDA) approaches combined various training objectives to align different aspects of domain-specific extracted features. In particular, the most prominent approach is domain-adversarial neural network (DANN) (Ganin et al., 2016) that employs a domain-adversarial training procedure between a domain classifier and the network's feature extractor to learn a discriminative and domain-invariant joint feature representation. The simplicity of DANN allows researchers to incorporate it with multiple other objectives such as semi-supervised learning (SSL) regularizers (Shu et al., 2018), discrepancy metrics (Long et al., 2015), co-training (Kumar et al., 2018), and auxiliary tasks (Bousmalis et al., 2016). Each of them plays an important role in enhancing domain adaptation ability of models in the current state-of-the-art methods. However, it is not trivial to apply these techniques

Proceedings of the 29th International Conference on Computational Linguistics, pages 4741–4752
October 12–17, 2022.

to textual tasks, where large transformer-based language models are essential to achieve top performance, because of the time and resource required to fine-tune and balance the effects of these terms for multiple different adaptation scenarios.

Meta-learning (ML) framework is an effective solution for the problem of hyperparameter optimization (Franceschi et al., 2018; Behl et al., 2019). Furthermore, it has been widely applied by recent works on Domain Generalization (DG) (Li et al., 2018; Dou et al., 2019), in which a learning procedure similar to that of Model-Agnostic Meta-Learning (MAML) (Finn et al., 2017) is leveraged to simulate the domain shift in train-test datasets by a virtual meta train-test set created from data drawn only from source domains. Though DG and UDA share close similarities, the final goal of each learning setting is different. More importantly, the MAML procedure is not applicable for UDA problem because of the lack of a clean validation dataset for meta-test step.

To this end, we propose to dynamically partition the training source data into a low-loss meta-source domain and a high-loss meta-target domain, inspired by self-paced learning (SPL) approach (Kumar et al., 2010). Our framework, called Meta Self-Paced Domain Adaptation (MSP-DA), employs a neural-SPL module to control the data selection process for meta train-test set using a learnable age hyperparameter as threshold while also introducing optimized weighting mechanisms for each of the combined loss' terms, including instance-wise weighting for the main classification task and layer-wise weighting for domain alignment losses. The weighted objectives on meta-source domain are minimized in meta-train step in a direction such that also leading to improvement in model's predictions on meta-target domain. During the learning process, parameters and age threshold of the neural-SPL module are updated based on model's evaluation performance in meta-test step, resulting in tuned weighting coefficients and learning schedules similar to that of a standard hyperparameter tuning process. To our knowledge, this is the first work to devise a neural network-based SPL method, in which both the sample weightings/selections and the age hyperparameter are dynamically optimized, generalizing previous works which require heuristic age schedule and complicated mathematical derivation for the corresponding instance weighting.

While the meta-target set does not contain samples from the true target domain, we argue that our formulation is beneficial for UDA because of the two following reasons. First, the proposed partition can result in two virtual domains with a significant discrepancy, and through learning to address in this hard setting that the model would gain the ability to adapt to other, possibly easier, domains. Another reason is based on the cluster assumption from SSL methods (Chapelle et al., 2006), which states that data points of the same class should concentrate around the same cluster, effectively forming a high-density low-loss region. In case of adapting between two highly dissimilar domains, these regions may get shifted significantly, as a consequence low-loss regions of target domain may contain considerable intersection with high-loss regions of source domain, as illustrated in Fig. 1. In other words, by learning to adapt the high-loss meta-target domain, the model would also be able to generalize to a significant portion of the true target domain.

We provide extensively evaluation of the proposed framework on the standard UDA benchmarks - FDU-MTL dataset for sentiment analysis task, along with additional results for event detection task on ACE-05 dataset, which is a much harder adaptation setting. Ablation studies and detailed analyses are conducted to validate each main component of our model and provide insights for future researches.

2 Related Work

Unsupervised Domain Adaptation for Text Classification The main line of research on UDA focuses on learning domain-invariant, which is either achieved by explicitly reducing the distance between source and target feature space measured by some distribution discrepancy metric (Long et al., 2015; Zellinger et al., 2017), or by adversarial training in which the feature extractor is trained to fool a domain classifier, both are jointly optimized to arrive at an aligned feature space (Ganin et al., 2016). We focus on applying the latter in transformer-based model (BERT) (Devlin et al., 2019) for textual tasks. Previous works have provided empirical results on different domains (Wright and Augenstein, 2020; Lin et al., 2020), different tasks (Naik and Rosé, 2020; Du et al., 2020), most of which presented little to no improvement following the standard domain adversarial training framework.

We further verify this point in our baseline performances.

Sample Weighting There are two main research directions to adaptively output weight of a sample during training process: addressing class imbalance by monotonically increasing function that imposes larger weights to ones with larger loss values (Sun et al., 2007; Lin et al., 2017), and suppressing the effect of noisy labels using monotonically decreasing function which focus on low-loss easy samples (Kumar et al., 2010; Jiang et al., 2014). Although straightforward to apply, the above methods are limited in that they all need a pre-specified closed-form weighting function, while their respective hyperparameters are sensitive to the change of training data such that careful tuning is required.

Meta-Learning There are three main categories of modern ML algorithms: learning a metric space to measure distance or similarity among data (Vinyals et al., 2016; Sung et al., 2018), learning an optimizer which updates all of model's parameters in a latent parameter space (Andrychowicz et al., 2016; Chen et al., 2018), and learning an initialization that is good for all tasks and able to fast adapt to unseen tasks (Finn et al., 2017; Jamal and Qi, 2019). Our approach falls into the last category, where the learning process follows MAML, more specifically its variant for DG problem in (Li et al., 2018).

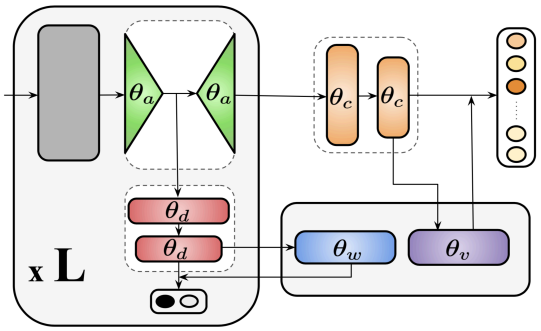

Figure 2: Architecture overview. (**gray**) Fixed BERT layers. (**green**) Adapter layers, bottleneck outputs of which are then fed into domain classifier heads (**red**). The neural-SPL module consists of instance-wise weighting head (**purple**) for main task classification (**orange**) and a layer-wise balancing head (**blue**) for domain adversarial training.

3 Model

We denote the source dataset $\mathbf{S} = \{(x_i^{\mathbf{s}}, y_i^{\mathbf{s}})\}_{i=1}^{N^{\mathbf{s}}}$ consisted of $N^{\mathbf{s}}$ samples and an unlabeled set of $N^{\mathbf{t}}$ samples $\mathbf{T} = \{x_i^{\mathbf{t}}\}_{i=1}^{N^{\mathbf{t}}}$ drawn from target domain. Label space $\mathbf{Y} = \{1, 2, \cdots, K\}$ of K classes is shared across domains.

Our model's feature encoder is a fixed pre-trained BERT encoder with hidden dimension $\mathbb{R}^{d_h}$, augmented by adapters with bottleneck representation of size $\mathbb{R}^{d_a}$. We refer to the main model learnable parameters as $\theta = (\theta_a, \theta_c, \theta_d)$, which includes the parameters of adapters, the main classification head, and the DANN heads. Following prior work (Ngo et al., 2021), low dimensional output from each layer's adapter is used by a separate DANN head for domain adversarial training. Our neural-SPL module consists of two weighting mechanisms: an instance-wise $f_v(\theta_v) : \mathbb{R} \to \mathbb{R}$ which weighs the contribution v_i of each example based on the its classification loss and a learnable age parameter λ_a; and a layer-wise $f_w(\theta_w) : \mathbb{R}^{d_a} \to 1$ that takes adapter representation of each layer and outputs the relative "magnitude" w^l of which the corresponding layer l should be aligned. We refer to the set of source samples whose losses are less than λ_a as meta-source domain $\mathbf{S}_{tr}$ while the rest is meta-target domain $\mathbf{S}_{ts}$. The latter acts, in meta-test step, as a validation set used to evaluate the model after meta-train step and provide learning signals to tune the "hyperparameters" from the neural-SPL module. The overall architecture is presented in Fig. 2.

3.1 Meta Self-Paced Learning

Self-Paced Learning Kumar et al. (2010) devised Self-Paced Learning method that extends Curriculum Learning (Bengio et al., 2009) to jointly learn the model and its curriculum, circumventing the need for an ad-hoc implementation of easiness based on some predetermined heuristics. Specifically, SPL employs an age hyperparameter λ_a that represents the current learning pace of the model. The objective is then reformulated as a weighted loss where each instance's contribution is thresholded by λ_a as follow:

$$\mathcal{L} = \sum_{i=1}^{n} v_i(l_i; \lambda_a) l_i \; ; \; v_i = \begin{cases} 1, & \text{if } l_i < \lambda_a \\ 0, & \text{otherwise.} \end{cases} \quad (1)$$

where l_i is the corresponding loss of i-th training sample. Intuitively, λ_a is the "age" of the model which is set to gradually grow as training proceed. Thus, only easy samples are considered at the initial learning stage while samples with larger losses will be slowly added to the model's curriculum as it progresses.

Adaptive SPL via Meta-Learning The advantage of incorporating SPL into a ML framework is

two-fold. First, ML provides a way to adaptively tune the highly sensitive λ_a, alleviating the need for manually devising an age scheduler. At the same time, SPL helps address the lack of clean validation data, by splitting the source domain instances of the current mini-batch into two disjoint sets based on the age value λ_a. The easy samples are used for meta-train step, in which the objective consists of a domain adversarial loss and a SPL-weighted classification loss:

$$\mathcal{L}_{tr}\left(\mathbf{S}_{tr}, \mathbf{T}; \theta\right) = \mathcal{L}_{ce}\left(\mathbf{S}_{tr}; \theta_a, \theta_c\right) + \mathcal{L}_{d}\left(\mathbf{S}_{tr}, \mathbf{T}; \theta_a, \theta_d\right) \quad (2)$$

$$v_i = f_v \circ \max(0, \frac{-l_i}{\lambda_a} + 1);\ \mathcal{L}_{ce}\left(\mathbf{S}_{tr}\right) = \sum_{x_i, y_i \in \mathbf{S}_{tr}} v_i l_i \quad (3)$$

where $l_i = l(x_i, y_i; \theta)$ is the loss of each sample and $\mathcal{L}_d$ is the weighted domain adversarial objective that is explained in the following section. f_v is a small feed-forward network with sigmoid as final activation function to guarantee the resulting weights located in the interval of [0, 1], and with no bias so that the 0-valued inputs will also correspond to outputs of the same value.

Typically, k gradient steps are applied to approximate the optimal solution that minimizes the current meta-train objective. Because of the sizeable transformer encoder, a high value of k will cost serious computation overhead. Thus, we decide to use $k = 1$, from which we observe no significant performance loss:

$$\bar{\theta} = \theta - \alpha \nabla_\theta (\mathcal{L}_{ce}\left(\theta_a, \theta_c\right) + \mathcal{L}_{d}\left(\theta_a, \theta_d\right)) \quad (4)$$

where α is meta-train learning rate. Next, the meta-test objective is the standard cross-entropy loss on samples in meta-target domain $\mathbf{S}_{ts}$ with loss values higher than λ_a:

$$\mathcal{L}_{ts}\left(\mathbf{S}_{ts}; \bar{\theta}\right) = \sum_{x_i, y_i \in \mathbf{S}_{ts}} \left(x_i, y_i; \bar{\theta}\right) \quad (5)$$

This acts as a hard, distinct domain that provides tuning signals for guiding model updates of both model's parameters in θ and hyperparameters v_i and λ_a.

3.2 Balancing domain adversarial objectives

The survey presented by (Rogers et al., 2020) provides a detailed probing and understanding of how the different layer-block of BERT encodes different types of information. Accordingly, each layer should contain a different amount of discrepancy between source and target domains.

To align these representation spaces between the two domains, we employ multiple domain classifiers at the bottleneck of every adapter:

$$\mathcal{L}_d = \sum_{l=1}^{L} w^l \mathcal{L}_d^l(\mathbf{z}_d^l, \mathbf{y}_d; \theta_d^l) \quad (6)$$

where each $\mathcal{L}_d^l$ is an adversarial term of a different DANN , taking adapter representations $\mathbf{z}_d^l$ of layer lth and domain labels $\mathbf{y}_d$ as inputs. These losses are weighted by a set of coefficients $\{w^l\}$ that corresponds to how important it is for the representations at the respective layer to be aligned. Following standard learning procedure, they would be hyperparameters that required careful tuning for each specific domain, which would be impractical (in our setting, there would be a total of 12 hyperparameters). To address the above issue, we employ a small feed-forward network f_w with a final softmax layer to output the relative layer-wise weights:

$$\mathbf{W} = [w^0, \cdots, w^{L-1}] = f_w(\mathbf{Z}_d; \theta_w) \quad (7)$$

where $\mathbf{Z}_d \in \mathbb{R}^{L \times d_a}$ is a set of layer representations, each element of which is the sum of all adapter representations of the corresponding layer with respect to the current mini-batch. As θ_w is updated throughout the ML process, $\mathbf{W}$ is dynamically tuned to maintain high performance on meta-test set while domain-adversarial training makes representations across layers domain-invariant.

Meta Optimization Following MLDG, meta-train and meta-test losses are combined in the final objective as follow:

$$\underset{\theta}{\operatorname{argmin}}\ \beta \mathcal{L}_{ts}\left(\bar{\theta}\right) + \mathcal{L}_{tr}\left(\theta\right), \quad (8)$$

$$\underset{\theta_w, \theta_v, \lambda_a}{\operatorname{argmin}}\ \mathcal{L}_{ts}\left(\bar{\theta}\right) \quad (9)$$

where β is meta-test balancing term. The second term in Eq. 9 is the result of passing the weights computed by neural-SPL module in Eq. 3 and 7 into Eq. 2 as pre-determined values, not learnable variables.

3.3 Self-training by incorporating Pseudo Label

Pseudo-labeling is an effective method to improve target domain performance by leveraging the predictions of previous step on unlabeled target data as additional learning signals for the main downstream task. We use the pseudo-labeled target data only for $\mathcal{L}_{ce}$ from Eq. 2 in meta-train step,

in which they are weighted and thresholded by neural-SPL module using the same λ_a as source data: $\mathcal{L}_{ce}\left(\mathbf{S}_{tr}, \overline{\mathbf{T}}\right) = \sum_{x_i, y_i \in \mathbf{S}_{tr} \cup \overline{\mathbf{T}}} v_i l_i$, where $\overline{\mathbf{T}}$ is the set of target samples with losses lower than λ_a. To alleviate the confirmation bias in pseudo-labeling, (Xie et al., 2019) provided strong regularizations and data augmentations to prevent model from propagating its own inaccuracy throughout the training process. In our case, neural-SPL module would ensure that only high confident pseudo labels are used, while meta-test step explicitly improves model's performance in low-density neighborhood of target domain. This is consistent with the expansion assumption proposed by Wei et al. (2021) on how self-training denoises pseudo-labels by bootstrapping an incorrectly pseudo-labeled example with its correctly pseudo-labeled neighbors. Thus, our framework is able to effectively leverage self-training by suppressing the noises and providing a robust training for the model. In addition, as we will discuss later section, the gradient updates of these pseudo-labeled samples are also regularized by the ML framework, forcing them to be consistent with meta-target domain.

Domains	Train	Unlabeled	Test
bn+nw	38644	N/A	9661
bc	N/A	3130	12520
cts	N/A	2885	10972
wl	N/A	3424	12767

Table 1: Statistics of **ACE-05**'s domains in UDA setting.

Domains	Train	Unlabeled	Test
Books	1400	2000	400
Elec.	1398	2000	400
DVD	1400	2000	400
Kitchen	1400	2000	400
Apparel	1400	2000	400
Camera	1397	2000	400
Health	1400	2000	400
Music	1400	2000	400
Toys	1400	2000	400
Video	1400	2000	400
Baby	1300	2000	400
Magaz.	1370	2000	400
Soft.	1315	475	400
Sport	1400	2000	400
IMDb	1400	2000	400
MR	1400	2000	400

Table 2: Statistics of the 16 domains in **FDU-MTL**

4 Experiments

4.1 Datasets, Settings, and Baselines

We evaluate the proposed model on the standard multi-domain sentiment analysis (SA) task. In addition, we also demonstrate the effectiveness of our framework when addressing the label-shift by applying MSP-DA to ED task with significant more classes in UDA setting.

FDU-MTL (Liu et al., 2017) A dataset included reviews from 16 domains for binary sentiment classification task. In each adaptation setting, a single domain is assigned as the target with unlabeled data while the other 15 are labeled source. Given the contextual sequence computed by models from a review, we use the first token [CLS] as the feature to predict its positive or negative sentiment.

ACE-05 (Walker et al., 2005) A densely annotated corpus collected from 5 different domains. Two of which are used as source data, while each of the rest is a target domain for an adaptation setting. Given a trigger word in the context of an event mention, the model is required to perform a multi-class classification task that assigns a predicted label into one of the pre-defined 34 event types (including 1 negative type).

Data Settings We provide statistics of each domain in UDA setting for **ACE-05** and **FDU-MTL** in Table 1 and Table 2, respectively. For **ACE-05** dataset, we gather data from two closely related domains, `bn` and `nw`, to create a sizable source domain dataset, 80% of which are used for training whilst the rest are used as test target domain for in-domain setting. For out-of-domain settings, each of the other domains is considered the target domain of a single adaptation scenario, where 20% of its documents are unlabeled training target data and the remainders are utilized as the test dataset. For **FDU-MTL** dataset, each of the 16 domains has a test set of 400 samples. The amount of training labeled and unlabeled data vary across domains, ranging from 1400 to 2000 samples. In each adaptation setting, a single domain is designated as the target domain while its unlabeled data are used in training set together with labeled data from the other 15 domains.

SA baselines We provide a comprehensive comparison of our proposed method with multiple baselines: **ASP-MTL** (Liu et al., 2017) and **DAEA** (Cai and Wan, 2019) are LSTM-based approaches. Transformer-based approaches include **BERT**, which is only fine-tuned on only labeled source domain, and **BERT+DANN** follows the standard adversarial training. Finally, **BertMasker**

System	MR	Appr.	Baby	Books	Cam.	DVD	Elec.	Hlth.	IMDB	Kitc.	Magz.	Musics	Softw.	Sport	Toys	Video	**aAcc**
ASP-MTL	76.7	87.0	88.2	84.0	89.2	85.5	86.8	88.2	85.5	86.2	92.2	82.5	87.2	85.7	88.0	84.5	86.1
DAEA	77.0	89.0	92.3	89.0	92.0	88.3	91.8	89.8	90.8	90.3	**96.5**	88.0	92.8	90.8	91.8	92.3	90.2
BERT	90.5	90.8	90.3	91.3	91.5	89.0	91.3	91.3	91.3	90.0	88.5	90.3	90.5	92.0	90.8	92.0	90.7
BERT+DANN	90.5	91.8	92.5	90.8	90.0	91.3	90.5	90.8	91.0	91.8	91.0	90.5	91.0	90.5	90.3	90.3	90.9
BertMasker	83.8	92.3	**92.8**	93.0	92.8	89.3	**93.3**	**95.3**	86.0	90.8	94.5	89.5	93.0	92.5	**93.8**	91.3	91.5
MSP-DA	**93.3**	**93.1**	92.5	**93.2**	**93.3**	**92.4**	93.1	93.2	**93.4**	**93.0**	93.1	**92.7**	**93.1**	**93.3**	93.5	**92.8**	**93.0**

Table 3: UDA performances for SA task on **FDU-MTL** test datasets. **aAcc** is the average accuracy score across all domains.

System	In-domain(bn+nw)			Out-of-domain (bc)			Out-of-domain (cts)			Out-of-domain (wl)			
	P	R	F	P	R	F	P	R	F	P	R	F	**aF1**
BERT	75.8	72.5	74.1	73.5	68.9	71.1	73.7	69.5	71.5	62.2	51.6	56.4	66.3
BERT+DANN	73.4	76.0	74.7	73.9	69.4	71.5	76.4	53.0	62.5	59.9	53.2	56.3	63.4
Uniform	76.8	79.4	78.1	75.4	66.3	70.5	80.4	21.0	33.3	61.8	45.7	52.6	52.1
Focal	78.2	77.6	77.9	71.7	72.9	72.2	72.9	68.5	70.1	64.8	54.2	59.0	67.1
Class-Balanced	79.3	78.3	**78.7**	77.8	68.0	72.5	78.0	44.0	56.2	59.0	50.3	54.3	61.0
MSP-DA	75.4	80.0	77.7	76.2	75.5	**75.8**	75.3	76.8	**76.1**	70.8	59.9	**64.8**	**72.2**

Table 4: UDA performances for ED task on **ACE-05** test datasets. **aF1** is the average out-of-domain F1 score.

(Yuan et al., 2021) is the state-of-the-art approach that learns to explicitly mask domain-related words from text, resulting in domain-agnostic sentences.

ED baselines For ED task, we also compare MSP-DA to other functional weighting schemes that trying to balance the learning process to address the label shift. In particular, **Uniform** treats each sample's loss equally, **Focal Loss** down-weights well-classified instance exponentially (Lin et al., 2017), and **Class-Balanced** uses a weighting factor that is inversely proportional to the number of samples (Cui et al., 2019) Noted that these model employ both adapter-based fine-tuning and adversarial training procedure.

Implementation details All models are implemented in Pytorch. We leverage pre-trained BERT-base models and checkpoints from Huggingface repository. (Wolf et al., 2020). We inject adapter layers after every feed-forward sub-blocks have bottleneck feed-forward architecture with down-sampled dimension chosen among [48, 96, 128]. All of the downstream heads are implemented as feed-forward networks with activation functions between layers. Each weighting net of neural-SPL module is a feed-forward network with 2 or 3 layers with hidden vectors of size [100, 50] or [200, 100, 50], respectively To train the proposed model, we use Adam optimizer with meta-train and meta-test learning rates α and γ both chosen from [5e-5, 1e-4, 5e-4, 1e-3, 5e-3], the mini-batch size from [50, 100, 150] of which 20% or 40% are unlabeled target data, and the meta-test balancing term β from [5, 2, 1, 0.5, 0.1]. We tune the hyperparameters for the proposed model using a random search. All hyperparameters are selected based on the F1 scores on the development set of a single domain. The same hyperparameters from this fine-tuning are then applied for other domains to demonstrate the domain-specificity problem. In the best model, fixed pre-train BERT-base layers augmented by adapters with bottleneck size 96 are used as our feature encoder. All objective heads have 2 hidden layers. We use Adam optimizer with a learning rate of 1e-4 for both meta-train and meta-test step, 100 for mini-batch size with 20% target data, and the meta-test balancing term is 2. Our reported results are averages of five runs using the best hyperparameter configuration with different random seeds.

4.2 Main Results

Sentiment Analysis SA results are presented in Table 3. While simple model using contextual embedding **BERT** outperforms all previous LSTM-based methods, we observe little to no improvement applying domain adversarial training naively with it. In particular, **BERT+DANN** actually has negative effect on about half of the domains, indicating that the standard baseline approach being unable to adjust to each specific adaptation setting. In contrast, our framework achieves the best performance for 11 review domains overall, surpassing the current state-of-the-art method **BertMasker** by 1.5 points on average. This demonstrates both the effectiveness and the robustness of MSP-DA to each domain.

Event Dectection UDA performances for ED task are presented in Table 4. Again, we observe that **BERT+DANN** only provides slight improvement for domain bc compare to **BERT**, while significantly degrades model's performances on the other two resulting in almost 3 points drop in average out-of-domain F1 score. Similarly, applying DANN for the adapter-based model without any weighting mechanism, as in **Uniform**, also has adverse effects on out-of-domain performances. **Class-Balanced**'s in-domain results are slightly

higher than other models due to its ability to balance the training process, which addresses the extreme negative-skewed label distribution of the given source data. In contrast, its domain adaptation ability is actually the lowest because of the change in data distribution across domains. **Focal Loss** performs generally better in out-of-domain settings as they generate weighting coefficients adaptively based on the current losses, without involving any domain-specific statistics. Finally, **MSP-DA** provides consistent improvements when adapting to any new domain, even achieving on average 5 points higher in F1 score compared to the best baseline method.

4.3 Ablation Study

In the first row-block of Table 5, we conduct an ablation study to validate the effectiveness of each of our main components by investigating the performance of the following variations of our model: **MSP-DA–mSPL** follows the normal SPL process to produce the weighting coefficients and train-test datasets for ML; **MSP-DA–DANN** trains only on source domain without utilizing unlabeled target data for domain adversarial objective; and **MSP-DA–PL** in which no pseudo-labels are leveraged for training. In general, our full model outperforms all variants across domains, even in the in-domain setting, which confirms the superiority and flexibility provided by the jointly optimized pacing and weights from our neural-SPL module. Especially for `wl` domain, domain adversarial training in MSP-DA manages to improve more than 8 F1 points.

Meta-test Selection To examine the correctness of our assumption, we augment the data selection process for meta domains in **Random** and **Reverse** variants. The former randomly selects training samples for each meta domain, whereas the latter implements the opposite hypothesis by choosing hard and easy instances for meta-train and meta-test sets, respectively. Both variants result in a considerable decline in domain adaptation results as shown in 5. Notably, the significant performance drop in the in-domain setting of **Random** indicates that simply constructing train-test sets without any appropriate condition can do more harm than good for the ML process. These empirical observations further confirm our initial assumption on how domain shift correlates well with the easy meta-train and hard meta-test sets.

System	In-domain(`bn+nw`)			Out-of-domain (`bc`)			Out-of-domain (`wl`)		
	P	R	F	P	R	F	P	R	F
MSP-DA – mSPL	74.5	79.7	77.0	77.5	72.0	74.6	64.1	51.9	57.4
MSP-DA – DANN	74.3	80.3	77.2	75.7	72.9	74.2	61.6	51.9	56.3
MSP-DA – PL	77.8	75.1	76.4	75.1	73.5	74.3	62.6	52.4	57.0
MSP-DA (Random)	73.0	76.4	74.7	75.6	73.3	74.4	61.0	50.3	55.0
MSP-DA (Reverse)	77.7	75.0	76.3	78.2	70.6	74.2	65.0	50.7	57.0
MSP-DA (Ours)	75.4	80.0	**77.7**	76.2	75.5	**75.8**	70.8	59.9	**64.8**

Table 5: Performances for Ablation Study

System	Out-of-domain (`bc`)			Out-of-domain (`wl`)		
	P	R	F	P	R	F
Fixed (25)	79.3	68.9	73.7	65.8	50.0	56.8
Fixed (50)	75.0	73.7	74.3	66.3	49.5	56.6
Fixed (75)	76.4	72.0	74.1	65.9	52.7	58.6
Linear Incrs	74.9	71.7	73.3	61.6	54.7	57.9
Meta (Ours)	76.2	75.5	**75.8**	70.8	59.9	**64.8**

Table 6: Performances for Age Hyperparameter Analysis

System	Out-of-domain (`bc`)			Out-of-domain (`wl`)		
	P	R	F	P	R	F
Constant	75.8	71.5	73.6	63.2	52.6	57.4
Anneal Up	75.4	71.0	73.1	63.5	52.6	57.4
Anneal Down	74.0	74.8	74.4	62.3	51.1	56.1
Meta (Ours)	76.2	75.5	**75.8**	70.8	59.9	**64.8**

Table 7: Performances for DANN Weighting Analysis

4.4 The Values of Age Hyperparameter

Age hyperparameter λ_a is usually the hardest to tune in a SPL system due to the fact that aside from the initial value, determining how λ_a changes throughout the training process also has a major impact on the final performance. Several prior works (Li and Gong, 2017; Ren et al., 2017) have proposed alternative age schedulers in place of the naive strategy which adds/multiples λ_a with a constant at each epoch. However, the value of λ_a in these methods still follows a predefined sequence, implying the need for a meticulous tuning process. In contrast, our neural-SPL module updates λ_a based on optimization signals from meta-test set, thus always able to create an appropriate dynamic curriculum regardless of different learning tasks and datasets. In Table 6, we examine how different values and schedules of age hyperparameter affect performances on `bc` and `wl` domains. The **Fixed (p)** settings with $\mathbf{p} \in [25, 50, 75]$ are variations of our model with λ_a values always corresponding to the unchanged **p**-th percentile of the current mini-batch's sample losses; or in other words, the number of samples in meta-train set is always a constant p percent that of the current mini-batch. Additionally, we evaluate the case in which **p** is linearly increased as training proceeds, similar to the standard SPL process, in **Linear Incrs** setting. The results show that the lower **p** is, the worse model performs, indicating that with too few meta-train data, the model will not be able to adapt to the hard meta-test domain. Surprisingly, the gradual rising scheduler of **Linear Incrs** is not as effective

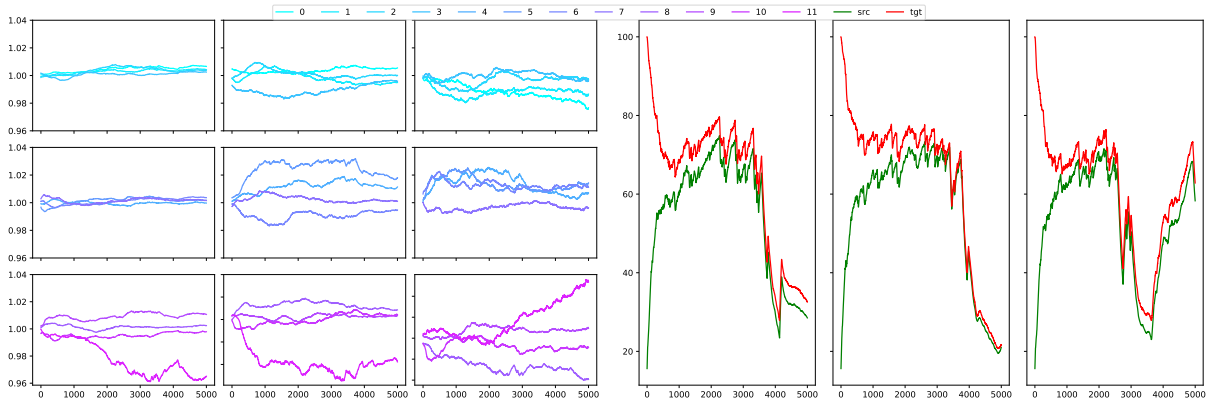

Figure 3: Three columns in each subplot correspond to domain `bc`, `cts`, `wl`, respectively. (**Left**) Layer-wise DANN weights at each training step. (**Right**) source and target age percentiles at each training step.

as the other **Fixed** variants. This means that the easy-to-hard assumption of prior SPL systems is not suitable for our ML framework.

λ_a Visualization To gain more insight into how age hyperparameter changes throughout the training process of each domain, we plot the values of λ_a in source-losses percentile against the number of update steps for 10 epochs in the right subplot of Fig. 3. While λ_a quickly follows the standard incremental trend initially, it starts to plateau within the 60-70 percentile range until eventually starting to decrease. Notably, behavior of λ_a diverges across domains in subsequent steps. Whereas λ_a continues the to decline in `bc` and `cts` domains, it experiences a complete trend reversal at the end of the training of `wl` domain. We hypothesis that this drastic change of λ_a is because of the gradients' dot product term that the objective in Eq. 9 implies, which we will delve deeper into in the discussion section below. The $\bigcap$ shape of λ_a correlates with the term's value as the model maximizes it to align the gradient directions between the meta train-test domains, going from negative initially as the training started, to 0 which causing the plateau, then gradually becoming positive as the model was able to adjust the updates of meta-train set to be consistent with that of meta-test set. However, for hard adaptation such as `wl` domain, too few data in meta-train set can cause a major disparity between the two meta domains again, thus the resulting trend reversal at the last few steps.

We also visualize the same plot for target-pseudo-losses percentile, which leads to an interesting observation: Initially, the model followed its own pseudo labels without any constraint and the high value of λ_a percentile represents model's incorrect overconfidence. However, these pseudo-label updates will cause discrepancies with meta-test domain, thus the ML framework will gradually fix the corresponding predictions, allowing only quality pseudo samples to be included in meta-train set. Eventually, the target trend converges with the source ones, suggesting that model's predictions on pseudo labels are then as consistent as on clean training labels.

4.5 Balancing Domain Adversarial Losses

Previous works have observed that the weight of DANN in the combined objective has a significant impact on the overall adaptation performance of the model. We further validate this point by investigating how different domain adversarial weighting schemes affect the results on `bc` and `wl` domains. Specifically, we evaluate 3 types of layer-wise weighting: (i) **Constant** - all layers share the same w^l value, (ii) **Anneal Up** - w^l slowly increases from lower to higher layers, and (iii) **Anneal Down** - w^l is highest for the first layer and gradually declines for subsequent layers. The results are present in Table 5, in which none of the schemes is better than the others in both domains. In contrast, the meta-learned coefficients of our framework manage to boost model's performances in every adaptation setting, especially for the hard `wl` domain where domain adversarial training matters the most.

We further visualize how each layer's weight changes during the learning process across domains in the left subplot of Fig. 3. In particular, we partition 12 layers of BERT-base model into 3 groups of 4 sequential layers, each of which is known to contain a different type of information that is important for a different type of task as described in the previous section. We can observe from the graphs a certain pattern: the higher level the group is, the more volatile its layers' coefficients are. However,

there is no specific rule shared among all domains regarding the value of each layer's weight. This affirms the sensitivity of domain adversarial balancing term to each individual domain and further justifies the effectiveness of the jointly optimized weighting in our framework.

5 Discussion

Following the analysis of MLDG framework presented in (Li et al., 2018), we decompose the meta-test loss, given that $\bar{\theta} = \theta - \alpha\mathcal{L}'_{tr}(\theta)$, using the first order Taylor expansion:

$$\mathcal{L}_{ts}\left(\theta - \alpha\mathcal{L}'_{tr}(\theta)\right) = \mathcal{L}_{ts}\left(\theta\right) + \frac{\partial\mathcal{L}_{ts}\left(\theta\right)}{\partial\theta}\left(-\alpha\frac{\partial\mathcal{L}_{tr}\left(\theta\right)}{\partial\theta}\right) \tag{10}$$

Denoting $\mathbf{G} = \frac{\partial\mathcal{L}_{ts}(\theta)}{\partial\theta} \cdot \frac{\partial\mathcal{L}_{tr}(\theta)}{\partial\theta}$ and plugging Eq. 10 into the final objective to update main model's parameters from Eq. 8 results in the following optimization problem:

$$\underset{\theta}{\operatorname{argmin}}\ \mathcal{L}_{tr}\left(\theta\right) + \mathcal{L}_{ts}\left(\theta\right) - \beta\alpha\mathbf{G} \tag{11}$$

The third term in Eq. 11 is a gradient-based regularization that penalizes inconsistency between parameter updates of meta-train and meta-test domains. By enforcing loss gradients of the two domains to follow a similar direction, Eq. 11 prevents the model from over-fitting to a single domain, effectively improves model's adaptation capacity provided that meta-test set is 'close' to target domain.

We further examine how the ML framework affects the values of neural-SPL module's parameters $(\theta_w, \theta_v, \lambda_a)$ in our model. Plugging Eq. 10 into the gradient of λ_a, we have:

$$\frac{\partial\mathcal{L}_{ts}\left(\bar{\theta}\right)}{\partial\lambda_a} = -\alpha\frac{\partial\mathcal{L}_{ts}\left(\theta\right)}{\partial\theta} \cdot \frac{\partial^2\mathcal{L}_{tr}\left(\theta\right)}{\partial\theta\partial\lambda_a} = -\alpha\mathbf{G} \cdot \frac{\partial f_v(\lambda_a)}{\partial\lambda_a} \tag{12}$$

From Eq. 12, we see that the multiplicative factor $\mathbf{G}$ also controls how the value of λ_a changes throughout the ML process. When there is a significant discrepancy between meta-train and meta-test domain, $\mathbf{G}$ would have a negative value, which would in effect push λ_a higher and allow more samples into meta-train set for easier adaptation to meta-test set. Conversely, a positive $\mathbf{G}$ would imply that the model is good enough to align the current meta domains, thus gradually pulling λ_a down to make the task harder. This behavior is clearly illustrated in Fig. 3. Similar arguments can be made for the meta-learned weighting coefficients, where $\mathbf{G}$ would encourage samples whose gradients are similar across domains while decreasing the contribution of those whose gradients are not. These understanding are also presented in (Shu et al., 2019) and closely related to how MAML works (Nichol et al., 2018; Raghu et al., 2019)

6 Conclusion

We present a novel ML framework for UDA setting that achieves state-of-the-art performance on ED task. In particular, a neural-SPL module is employed to adaptively partition source domain into meta-train and meta-test set, while simultaneously learns the instance-wise and layer-wise weights for the loss terms of downstream task and domain adversarial task respectively. The proposed model significantly improves domain adaptation performances against various baselines on every domain without domain-specific hyperparameter tuning. In the future, we intend to apply our approach to the several direction: (1) We will extend our work to multilingual problems (Pouran Ben Veyseh et al., 2022), or other domains and tasks (Lu et al., 2021); (2) We will incorporate different novel domain adaptation regularization methods (Phung et al., 2021); (3) We will adapt our framework to more general multi-source domain adaptation setting.

Acknowledgement

This research has been supported by the Army Research Office (ARO) grant W911NF-21-1-0112 and the NSF grant CNS-1747798 to the IUCRC Center for Big Learning. This research is also based upon work supported by the Office of the Director of National Intelligence (ODNI), Intelligence Advanced Research Projects Activity (IARPA), via IARPA Contract No. 2019-19051600006 under the Better Extraction from Text Towards Enhanced Retrieval (BETTER) Program. The views and conclusions contained herein are those of the authors and should not be interpreted as necessarily representing the official policies, either expressed or implied, of ARO, ODNI, IARPA, the Department of Defense, or the U.S. Government. The U.S. Government is authorized to reproduce and distribute reprints for governmental purposes notwithstanding any copyright annotation therein. This document does not contain technology or technical data controlled under either the U.S. International Traffic in Arms Regulations or the U.S. Export Administration Regulations.

References

Marcin Andrychowicz, Misha Denil, Sergio Gomez Colmenarejo, Matthew W. Hoffman, David Pfau, Tom Schaul, and Nando de Freitas. 2016. Learning to learn by gradient descent by gradient descent. In *Advances in Neural Information Processing Systems*, pages 3981–3989.

Harkirat Singh Behl, Atilim Gunecs Baydin, and Philip H. S. Torr. 2019. Alpha maml: Adaptive model-agnostic meta-learning. *ArXiv*, abs/1905.07435.

Yoshua Bengio, Jérôme Louradour, Ronan Collobert, and Jason Weston. 2009. Curriculum learning. In *Proceedings of the 26th Annual International Conference on Machine Learning*, ICML '09, page 41–48, New York, NY, USA. Association for Computing Machinery.

Konstantinos Bousmalis, George Trigeorgis, N. Silberman, Dilip Krishnan, and D. Erhan. 2016. Domain separation networks. In *Advances in Neural Information Processing Systems*.

Yitao Cai and Xiaojun Wan. 2019. Multi-domain sentiment classification based on domain-aware embedding and attention. In *Proceedings of the Twenty-Eighth International Joint Conference on Artificial Intelligence, IJCAI-19*, pages 4904–4910. International Joint Conferences on Artificial Intelligence Organization.

Olivier Chapelle, Bernhard Schölkopf, and Alexander Zien, editors. 2006. *Semi-Supervised Learning*. The MIT Press.

Junkun Chen, Xipeng Qiu, Pengfei Liu, and Xuanjing Huang. 2018. Meta multi-task learning for sequence modeling. In *AAAI Conference on Artificial Intelligence*, pages 5070–5077. AAAI Press.

Yin Cui, Menglin Jia, Tsung-Yi Lin, Yang Song, and Serge J. Belongie. 2019. Class-balanced loss based on effective number of samples. In *CVPR*, pages 9268–9277. Computer Vision Foundation / IEEE.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. Bert: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL-HLT)*.

Q. Dou, Daniel Coelho de Castro, K. Kamnitsas, and B. Glocker. 2019. Domain generalization via model-agnostic learning of semantic features. In *NeurIPS*.

Chunning Du, Haifeng Sun, Jingyu Wang, Qi Qi, and Jianxin Liao. 2020. Adversarial and domain-aware BERT for cross-domain sentiment analysis. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 4019–4028, Online. Association for Computational Linguistics.

Chelsea Finn, Pieter Abbeel, and Sergey Levine. 2017. Model-agnostic meta-learning for fast adaptation of deep networks. In *Proceedings of the 34th International Conference on Machine Learning*, volume 70 of *Proceedings of Machine Learning Research*, pages 1126–1135. PMLR.

Luca Franceschi, Paolo Frasconi, Saverio Salzo, Riccardo Grazzi, and Massimiliano Pontil. 2018. Bilevel programming for hyperparameter optimization and meta-learning. In *Proceedings of the 35th International Conference on Machine Learning*, volume 80 of *Proceedings of Machine Learning Research*, pages 1568–1577. PMLR.

Yaroslav Ganin, Evgeniya Ustinova, Hana Ajakan, Pascal Germain, Hugo Larochelle, François Laviolette, Mario March, and Victor Lempitsky. 2016. Domain-adversarial training of neural networks. *Journal of Machine Learning Research*, 17(59):1–35.

Muhammad Abdullah Jamal and Guo-Jun Qi. 2019. Task agnostic meta-learning for few-shot learning. In *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 11711–11719.

Lu Jiang, Deyu Meng, T. Mitamura, and A. Hauptmann. 2014. Easy samples first: Self-paced reranking for zero-example multimedia search. *Proceedings of the 22nd ACM international conference on Multimedia*.

Abhishek Kumar, Prasanna Sattigeri, Kahini Wadhawan, Leonid Karlinsky, Rogerio Feris, Bill Freeman, and Gregory Wornell. 2018. Co-regularized alignment for unsupervised domain adaptation. In *Advances in Neural Information Processing Systems*, volume 31. Curran Associates, Inc.

M Pawan Kumar, Benjamin Packer, and Daphne Koller. 2010. Self-paced learning for latent variable models. In *Advances in Neural Information Processing Systems*, pages 1189–1197.

Da Li, Yongxin Yang, Yi-Zhe Song, and Timothy Hospedales. 2018. Learning to generalize: Meta-learning for domain generalization.

Hao Li and Maoguo Gong. 2017. Self-paced convolutional neural networks. In *Proceedings of the Twenty-Sixth International Joint Conference on Artificial Intelligence, IJCAI-17*, pages 2110–2116.

Chen Lin, Steven Bethard, Dmitriy Dligach, Farig Sadeque, Guergana K Savova, and T. A. Miller. 2020. Does bert need domain adaptation for clinical negation detection? *Journal of the American Medical Informatics Association (JAMIA)*.

Tsung-Yi Lin, Priya Goyal, Ross B. Girshick, Kaiming He, and Piotr Dollár. 2017. Focal loss for dense object detection. In *ICCV*, pages 2999–3007. IEEE Computer Society.

Pengfei Liu, Xipeng Qiu, and Xuanjing Huang. 2017. Adversarial multi-task learning for text classification. In *Proceedings of the 55th Annual Meeting of the*

Association for Computational Linguistics (Volume 1: Long Papers), pages 1–10, Vancouver, Canada. Association for Computational Linguistics.

Mingsheng Long, Yue Cao, Jianmin Wang, and Michael I. Jordan. 2015. Learning transferable features with deep adaptation networks.

Qiuhao Lu, Dejing Dou, and Thien Huu Nguyen. 2021. Parameter-efficient domain knowledge integration from multiple sources for biomedical pre-trained language models. In *Findings of the Association for Computational Linguistics: EMNLP 2021*, pages 3855–3865, Punta Cana, Dominican Republic. Association for Computational Linguistics.

Aakanksha Naik and Carolyn Rosé. 2020. Towards open domain event trigger identification using adversarial domain adaptation. In *Proceedings of the Annual Meeting of the Association for Computational Linguistics (ACL)*.

Nghia Trung Ngo, Duy Phung, and Thien Huu Nguyen. 2021. Unsupervised domain adaptation for event detection using domain-specific adapters. In *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*, page 4015–4025. Association for Computational Linguistics.

Alex Nichol, Joshua Achiam, and J. Schulman. 2018. On first-order meta-learning algorithms. *ArXiv*, abs/1803.02999.

Duy Phung, Hieu Minh Tran, Minh Van Nguyen, and Thien Huu Nguyen. 2021. Learning cross-lingual representations for event coreference resolution with multi-view alignment and optimal transport. In *Proceedings of the 1st Workshop on Multilingual Representation Learning*, pages 62–73, Punta Cana, Dominican Republic. Association for Computational Linguistics.

Amir Pouran Ben Veyseh, Minh Van Nguyen, Franck Dernoncourt, and Thien Nguyen. 2022. MINION: a large-scale and diverse dataset for multilingual event detection. In *Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 2286–2299, Seattle, United States. Association for Computational Linguistics.

Aniruddh Raghu, Maithra Raghu, Samy Bengio, and Oriol Vinyals. 2019. Rapid learning or feature reuse? towards understanding the effectiveness of maml. *arXiv preprint arXiv:1909.09157*.

Yazhou Ren, Peng Zhao, Yongpan Sheng, Dezhong Yao, and Zenglin Xu. 2017. Robust softmax regression for multi-class classification with self-paced learning. In *Proceedings of the Twenty-Sixth International Joint Conference on Artificial Intelligence, IJCAI-17*, pages 2641–2647.

Anna Rogers, Olga Kovaleva, and Anna Rumshisky. 2020. A primer in bertology: What we know about how bert works. *Transactions of the Association for Computational Linguistics*, 8:842–866.

Jun Shu, Qi Xie, Lixuan Yi, Qian Zhao, Sanping Zhou, Zongben Xu, and Deyu Meng. 2019. Meta-weight-net: Learning an explicit mapping for sample weighting. In *Advances in Neural Information Processing Systems*.

Rui Shu, Hung H. Bui, Hirokazu Narui, and Stefano Ermon. 2018. A DIRT-T approach to unsupervised domain adaptation. *CoRR*, abs/1802.08735.

Yanmin Sun, Mohamed S. Kamel, Andrew K. C. Wong, and Yang Wang. 2007. Cost-sensitive boosting for classification of imbalanced data. *Pattern Recognit.*, 40(12):3358–3378.

Flood Sung, Yongxin Yang, Li Zhang, Tao Xiang, Philip H. S. Torr, and Timothy M. Hospedales. 2018. Learning to compare: Relation network for few-shot learning. In *CVPR*, pages 1199–1208. IEEE Computer Society.

Oriol Vinyals, Charles Blundell, Timothy Lillicrap, koray kavukcuoglu, and Daan Wierstra. 2016. Matching networks for one shot learning. In *Advances in Neural Information Processing Systems*, volume 29. Curran Associates, Inc.

Christopher Walker, Stephanie Strassel, Julie Medero, and Kazuaki Maeda. 2005. Ace 2005 multilingual training corpus. In *Technical report, Linguistic Data Consortium*.

Colin Wei, Kendrick Shen, Yining Chen, and Tengyu Ma. 2021. Theoretical analysis of self-training with deep networks on unlabeled data. In *International Conference on Learning Representations*.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Rémi Louf, Morgan Funtowicz, Joe Davison, Sam Shleifer, Patrick von Platen, Clara Ma, Yacine Jernite, Julien Plu, Canwen Xu, Teven Le Scao, Sylvain Gugger, Mariama Drame, Quentin Lhoest, and Alexander M. Rush. 2020. Transformers: State-of-the-art natural language processing. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 38–45, Online. Association for Computational Linguistics.

Dustin Wright and Isabelle Augenstein. 2020. Transformer based multi-source domain adaptation. *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*.

Qizhe Xie, Eduard H. Hovy, Minh-Thang Luong, and Quoc V. Le. 2019. Self-training with noisy student improves imagenet classification. *CoRR*, abs/1911.04252.

Jianhua Yuan, Yanyan Zhao, Bing Qin, and Ting Liu. 2021. Learning to share by masking the non-shared for multi-domain sentiment classification.

Werner Zellinger, Thomas Grubinger, Edwin Lughofer, Thomas Natschläger, and Susanne Saminger-Platz. 2017. Central moment discrepancy (cmd) for domain-invariant representation learning.

WARM: A Weakly (+Semi) Supervised Math Word Problem Solver

Oishik Chatterjee *
Flipkart
Bangalore India

Isha Pandey
Department of CSE
IIT Bombay

Aashish Waikar*
Quadeye
Gurgaon India

{oishik75, iishapandey77, aashishwaikar}@gmail.com

Vishwajeet Kumar
IBM India Research Lab
Bangalore India
vishk024@in.ibm.com

Ganesh Ramakrishnan
Department of CSE
IIT Bombay
ganesh@cse.iitb.ac.in

Abstract

Solving math word problems (MWPs) is an important and challenging problem in natural language processing. Existing approaches to solve MWPs require full supervision in the form of intermediate equations. However, labeling every MWP with its corresponding equations is a time-consuming and expensive task. In order to address this challenge of equation annotation, we propose a weakly supervised model for solving MWPs by requiring only the final answer as supervision. We approach this problem by first learning to generate the equation using the problem description and the final answer, which we subsequently use to train a supervised MWP solver. We propose and compare various weakly supervised techniques to learn to generate equations directly from the problem description and answer. Through extensive experiments, we demonstrate that without using equations for supervision, our approach achieves accuracy gains of 4.5% and 32% over the state-of-the-art weakly supervised approach (Hong et al., 2021), on the standard Math23K (Wang et al., 2017) and AllArith (Roy and Roth, 2017) datasets respectively. Additionally, we curate and release new datasets of roughly 10k MWPs each in English and in Hindi (a low resource language). These datasets are suitable for training weakly supervised models. We also present an extension of Warm[1] to semi-supervised learning and present further improvements on results, along with insights.

1 Introduction

A Math Word Problem (MWP) is a numerical problem expressed in natural language (problem description), that can be transformed into an equation (solution expression), which can be solved to obtain the final answer. In Table 1, we present an example MWP. Automatically solving MWPs has recently gained lot of research interest in natural language processing (NLP). The task of automatically solving MWPs is challenging owing to two primary reasons: i) *The unavailability of large training datasets with problem descriptions, equations as well as corresponding answers* – as depicted in Table 1, equations can provide full supervision, since equations can be solved to obtain the answer, and the answer itself amounts to weak supervision only; ii) *Challenges in parsing the problem description and representing it suitably for effective decoding of the equations*. Paucity of completely supervised training data can pose a severe challenge in training MWP solvers. Most existing approaches assume the availability of full supervision in the form of both intermediate equations and answers for training. However, annotating MWPs with equations is an expensive and time consuming task. There exists only two sufficiently large datasets (Wang et al., 2017) in Chinese and (Amini et al., 2019) in English consisting of MWPs with annotated intermediate equations for supervised training.

We propose a novel two-step weakly supervised technique to solve MWPs by making use only of the weak supervision, in the form of answers. In the first step, using only the answer as supervision, we learn to generate equations for questions in the training set. In the second step, we use the generated equations along with answers to train any state-of-the-art supervised model. We illustrate the effectiveness of our weakly supervised approach on our newly curated reasonably

* The author contributed to this work while at IIT Bombay

[1] Warm stands for **We**a**k**ly supe**R**vised **M**ath solver.

Proceedings of the 29th International Conference on Computational Linguistics, pages 4753–4764
October 12–17, 2022.

Problem: It costs Rs 5.0 to buy 10.0 peppermint candies. If the candies all have the same price,how much does it cost to buy 1.0 candy ? **Equation**: X=$(5.0/10.0) \times 1.0$ (Under full supervision) **Answer**: 0.5 (Under weak supervision)
Problem: एक आयताकार बकरी के बाड़े का परिमाप 34.0 मीटर है। बाड़े की चौड़ाई 6.0 मीटर है। बाड़े की लम्बाई कितनी होगी? **Equation**: X=$34.0 - (2 \times 6.0)/2$ (Under full supervision) **Answer**: 11.0 (Under weak supervision)

Table 1: Example of Math Word Problems in English and Hindi language

large dataset in English and a similarly curated dataset in Hindi - a low resource language. We also perform experiments with semi-supervision and demonstrate how our model can benefit from a small amount of completely labelled data. Our main contributions are as follows:

1) An approach, Warm, (*c.f.*, Section 4) for generating equations from MWPs, given (weak) supervision only in the form of the final answer.

2) An extended semi-supervised training method to leverage a small amount of annotated equations as strong/complete supervision.

3) A new and relatively large dataset, EW10K, in English (with more than 10k instances), for training weakly supervised models for solving MWPs (*c.f.*, Section 3). Given that weak supervision makes it possible to train MWP solvers even in the absence of extensive equation labels, we also present results on a similarly crawled dataset, HW10K(with around 10k instances), in a low resource language, *viz.* Hindi, where we can avoid the additional effort required to generate equation annotations.

4) We empirically show that Warm outperforms state-of-the-art models on most of the datasets. Further, we empirically demonstrate the benefits of the semi-supervised extension to Warm.

2 Related Work

Automatic math word problem solving has recently drawn significant interests in the natural language processing (NLP) community. Existing MWP solving methods can be broadly classified into four categories: (a) rule-based methods, (b) statistics-based methods, (c) tree-based methods, and (d) neural-network-based methods.

Rule-based systems (Fletcher, 1985; Bakman, 2007; Yuhui et al., 2010) were amongst the earliest approaches to solve MWPs. They rely heavily on hand-engineered rules that might cover a limited domain of problems. Statistics-based methods (Hosseini et al., 2014; Kushman et al., 2014; Sundaram and Khemani, 2015; Mitra and Baral, 2016; Liang et al., 2016a,b) use predefined logic templates and employ traditional machine learning models to identify entities, quantities, and operators from the problem text and subsequently employ simple logical inference to yield the numeric answer. (Upadhyay et al., 2016) employ a semi-supervised approach by learning to predict templates and corresponding alignments using both explicit and implicit supervision. Tree-based methods (Roy and Roth, 2015; Koncel-Kedziorski et al., 2015; Roy et al., 2016; Roy and Roth, 2017, 2018) replaced the process of deriving an equation by constructing an equivalent tree structure, step by step, in a bottom-up manner.

More recently, neural network-based MWP solving methods have been proposed (Wang et al., 2017, 2018a,b; Huang et al., 2018; Chiang and Chen, 2019; Wang et al., 2019; Liu et al., 2019; Xie and Sun, 2019; Wu et al., 2021; Shen et al., 2021). These employ an encoder-decoder architecture and train in an end-to-end manner without the need for hand-crafted rules or templates. (Wang et al., 2017) were the first to propose a sequence-to-sequence (Seq2Seq) model, *viz.*, Deep Neural Solver, for solving MWPs. They employ an RNN-based encoder-decoder architecture to directly translate the problem text into equation templates and also release a high-quality large-scale dataset, Math23K, consisting of 23,161 MWPs in Chinese.

(Liu et al., 2019) and (Xie and Sun, 2019) propose tree-structured decoding that generates the syntax tree of the equation in a top-down manner. In addition to applying tree-structured decoding, (Zhang et al., 2020) propose a graph-based encoder to capture relationships and order information among the quantities. For a more comprehensive review on automatic MWP solvers, readers can refer to a recent survey paper (Zhang et al., 2018).

Unlike all the previous works that require equations for supervision, (Hong et al., 2021) propose

a weakly supervised method for solving MWPs, where the answer alone is required for training. Their approach attempts to generate the equation tree in a rule based manner so that the correct answer is reached. They then train their model using the fixed trees. With the same motivation. we also propose a novel weakly supervised model, Warm, (*c.f.*, Section 4) for solving MWPs using only the final answer for supervision. We show how Warm can be extended to semi-supervised joint learning in the presence of weak answer-level supervision in conjunction with some equation-level supervision. Further, we empirically demonstrate that Warm outperforms (Hong et al., 2021) on all the datasets.

We also took insights from (Kumar et al., 2018), (Thakoor et al., 2018), (Akula et al., 2021), (Kumar et al., 2015), (Singh et al., 2016), (Kumar et al., 2019) and (Tarunesh et al., 2021) for handling mathematical data in two different languages.

This paper is organized as follows. In Section 3, we set the premise for our approach by describing the new datasets (EW10K and HW10K) for weak supervision that we release. In Section 4, we describe our weakly supervised approach Warm and its semi-supervised extension Warm-S. In Section 5, we present the experimental setup whereas in Section 6 we delve into the results and its analysis before concluding in Section 7.

3 Dataset

Currently, there does not exist any sufficiently large English dataset for single and simple equation MWPs. While there exists an English dataset (Amini et al., 2019) with sufficiently large MWPs, the questions in the dataset are meant to be evaluated in a multiple choice question (MCQ) manner. Also, the equation associated with each word problem in this dataset is significantly more complex and requires several binary and unary operators. On the other hand, Math23K (Wang et al., 2017) is in Chinese and Dolphin18k (Huang et al., 2016) contains mostly multi-variable word problems. To address these gaps, we curate a new English MWP dataset, *viz.*, EW10K[2] consisting of 10227 word problem instances (each associated with a single equation) that can be used for training MWP solver models in a weakly supervised manner.

We crawled IXL[3] to obtain MWPs for grades VI until X. These word problems involve a wide variety of mathematical computations ranging from simple addition-subtraction to much harder mensuration and probability problems. The dataset consists of 10 different types of problems, spanning 3 tiers of difficulty. We also annotate the dataset with the target unit. The exact distributions are presented in Figure 1.

We similarly created a MWP dataset in Hindi[2] - a low resource language. It consists of 9,896 question answer pairs. To the best of our knowledge, this is the first MWP dataset of such size in Hindi.

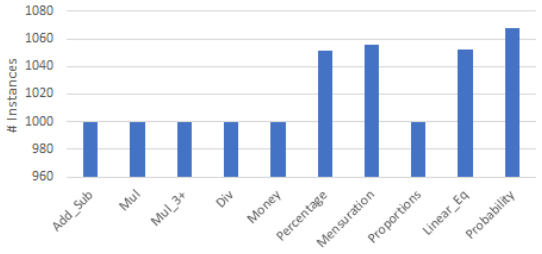

Figure 1: Distribution on different types of questions

4 Our Approach: Warm

We propose a weakly supervised model, Warm, for solving the MWP using only the answer for supervision. It is a two-step cascaded approach for weakly supervised MWP solving. For the first step, we propose a model that predicts the equation, given a problem text and answer. This model uses reinforcement learning to search the space of possible equations, given the question and the correct answer only. The answer acts as the goal of the agent and the search is terminated either when the answer is reached or when the equation length exceeds a pre-defined length (this is required, else the search space would be infinitely large). The model is designed to be a two layer bidirectional GRU (Cho et al., 2014) encoder and a decoder network with fully connected units (described in Section 4.3). We refer to this model as Warm. Note that this model requires an answer to determine when to stop exploring. Since we ultimately want a model which should only take the problem statement as input and generate the answer (by generating the correct equation), this model alone is insufficient for evaluation. Using this model, we create a noisy equation-annotated dataset from

[2] https://github.com/iishapandey/WARM

[3] https://in.ixl.com/

the weakly annotated training dataset (the training dataset has answers since it is weakly supervised). We use only those instances to create the dataset for which the equation generated by the model yields the correct answer. Note that the equations are noisy, since there is no guarantee that the generated equation will be the shortest or even correct. In the second step, we use this noisy data for supervised training of a state-of-the-art model. The trained supervised model is finally used for evaluation. For simplicity, we provide a summary of notations in Section 1 in supplementary.

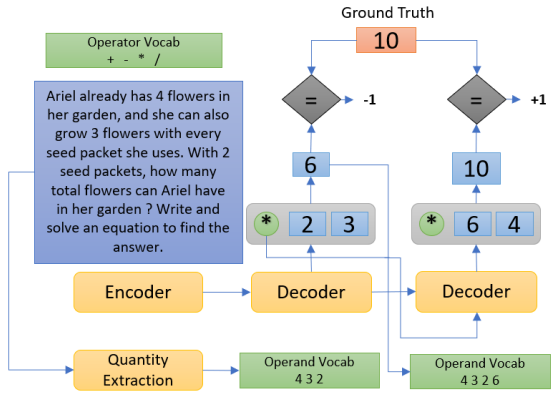

Figure 2: Inference Illustration

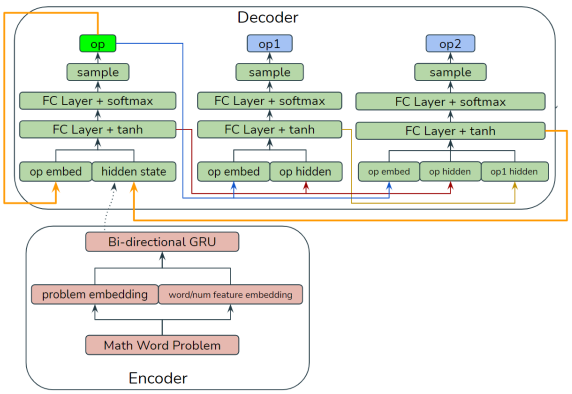

Figure 3: Architecture for generating equation tree in Warm.

4.1 Equation Generation

The first step of our approach is to generate equation given a problem text P and answer A. This is done by using our Warm model. The problem text is passed through the encoder of the Warm model to get its encoded representation which is then fed to the decoder. At each time step, the decoder generates an operator and its two operands from the operator and operand vocabulary list. The operation is then executed to obtain a new quantity. This quantity is checked against the ground truth and if it matches the ground truth, the decoding is terminated and a reward of +1 is assigned. Else we assign a reward of -1 and the generated quantity is added to the operand vocabulary list and the decoding continues. The working of the Warm model and architecture are illustrated in Figure 2 and Figure 3 respectively. In the following few subsections, we describe the architecture as well as the training in details.

4.2 Encoder

The encoder takes as input, the MWP represented as a sequence of tokens $P = x_1x_2x_3...x_n$. We replace each number in the question with a special token $< num\_j >$ to obtain this sequence where j denotes the index of number in the operand vocab for that question. Each word token x_i is first transformed into the corresponding word embedding $\boldsymbol{x}_i$ by looking up an embedding matrix $\boldsymbol{M}_w$. Next, a binary feature is appended to the embedding to indicate whether the token is a word or a number. As depicted in the lower half of Figure 3, this appended embedding vector is then passed through a 2 layer bidirectional GRU (Cho et al., 2014) and the outputs from both directions of the final layer are summed to get the encoded representation of the text. This representation is then passed on to the decoder.

4.3 Decoder

The decoder consists of 3 fully connected networks for generating operator, left operand and the right operand. As illustrated in the upper half of Figure 3, the decoder takes as input the previous decoded operand and the last decoder hidden state and outputs the operator, left operand, right operand and hidden state at the current time step. We initialize the decoder hidden state with the last state of the encoder:

$$o_t^p, o_t^l, o_t^r, h_t^d = DecoderFCN(o_{t-1}^p, h_{t-1}^d)$$

Here, h_t^d is the decoder hidden state at the tth time step. o_t^p, o_t^l and o_t^r are probability distributions over operators, left and right operands respectively.

4.3.1 Operator generation

Inside our decoder, we learn an operator embedding matrix $Em_{op}(op_{t-1})$, where op_{t-1} is the operator sampled in the last time step. We generate

the operator hidden state h_t^{op} using a gating mechanism.

$$g_t^{op} = \sigma(W_{op}^1[Em_{op}(op_{t-1}); h_{t-1}^d] + b_{op}^1)$$

$$h_t^{op} = g_t^{op} * tanh(W_{op}^2[Em_{op}(op_{t-1}); h_{t-1}^d] + b_{op}^2)$$

$$o_t^p = softmax(W_{op}^3 h_t^{op} + b_{op}^3)$$

Here $\sigma()$ denotes the sigmoid function and $*$ denotes elementwise multiplication. We sample operator op_t from the probability distribution o_t^p.

4.3.2 Left Operand Generation

We use the embedding of the current operator $Em(op_t)$ and the operator hidden state h_t^{op} to obtain a probability distribution over the operands. We employ a similar gating mechanism as used for generating *operator*.

$$g_t^{ol} = \sigma(W_{ol}^1[Em_{op}(op_t); h_t^{op}] + b_{ol}^1)$$

$$h_t^{ol} = g_t^{ol} * tanh(W_{ol}^2[Em_{op}(op_t); h_t^{op}] + b_{ol}^2)$$

$$o_t^l = softmax(W_{ol}^3 h_t^{ol} + b_{ol}^3)$$

We sample the left operand ol_t from the probability distribution o_t^l.

4.3.3 Right Operand Generation

For generating the right operand, we use the additional context information that is already available from the generated left operand. Thus, in addition to the operator embedding $Em_{op}(op_t)$ and operator hidden state h_t^{op} we also use the left operand hidden state to get the right operand hidden state h_t^{or}.

$$g_t^{or} = \sigma(W_{or}^1[Em_{op}(op_t); h_t^{op}; h_t^{ol}] + b_{or}^1)$$

$$h_t^{or} = g_t^{or} * tanh(W_{or}^2[Em_{op}(op_t); h_t^{op}; h_t^{ol}] + b_{or}^2)$$

$$o_t^r = softmax(W_{or}^3 h_t^{or} + b_{or}^3)$$

We sample the right operand or_t from the probability distribution o_t^l. The hidden state h_t^{or} is returned as the current decoder state h_t^d.

4.3.4 Bottom-up Equation Construction

For each training instance, we maintain a dictionary of possible operands $OpDict$. Initially, this dictionary contains the numeric values from the instance, *i.e.*, the number tokens we have replaced with $< num\_j >$ during encoding. At the tth decoding step, we sample an operator op_t, left operand ol_t and right operand or_t. We get an intermediate result by using the operator corresponding to op_t on the operands ol_t and or_t. This intermediate result is added to $OpDict$ which enables us to reuse the results of previous computations in future decoding steps. Thus, $OpDict$ acts as a dynamic dictionary of operands and we use it to progress towards the final answer in a bottom-up manner.

4.4 Rewards and Loss

We use the REINFORCE (Williams, 1992) algorithm for training the model using just the final answer as the ground truth. We model the reward as $+1$ if the predicted answer matches the ground truth and -1 if the predicted answer does not equal the ground truth.

Let R_t be defined as the reward obtained after generating $y_t = (op_t, o_l, o_r)$. The probability P_t of generating the tuple y_t is specified by $p_\theta(y_t) = \prod_{i=1}^{t} o_i^p \times o_i^l \times o_i^r$. The loss is specified as $L = -\sum_i \mathbb{E}_{p_\theta(y_i)}[R_i]$ and the corresponding gradient is $\nabla_\theta L = \sum_i \sum_{y_i} p_\theta(y_i) R_i \nabla_\theta \log p_\theta(y_i)$.

Since the space of y_i makes it infeasible to compute the exact gradient, we use the standardized technique of sampling y_i from $p_\theta(y_i)$ to obtain an estimate of the gradient.

4.5 Beam Exploration in Training

Since the reward space for our problem is very sparse, we observe that during model training, the gradients go to zero. Our model converges too quickly to some local optima and consequently, the training accuracy saturates to some fixed value despite performing training for a large number of epochs. In order to counter this problem, we employ beam exploration in the training procedure. Instead of sampling operator op_t, left operand ol_t and right operand or_t only once in each decoding step, we sample w triplets (op_t, ol_t, or_t) without replacement from the joint probability space in each decoding step. Here w is the beam width. This helps in exploring w different paths each epoch, thus increasing the exploration capabilities and reduce the problem of cold start. In order to select beams from all possible candidates, we have tried multiple heuristics by inspecting the probability and reward values. We have observed empirically that selecting the beam that gives a positive reward at the earliest decoding step yields the best performance. This enables our model to explore more and mitigates the above problem significantly.

4.6 Warm-S: Adding Semi-supervision

While it is expensive to completely label large MWP datasets with equations, it is relatively easier to annotate a small percentage of that data. We argue that addition of this small amount of semi-supervision can improve the model training significantly.

We, therefore, consider a model that benefits from a relatively small amount of *strong* supervision in the form of equation annotated data: D_s, in addition to a potentially larger sized math problem datasets with only *weak* supervision D_w. For a data instance d: $d.p$, $d.e$, and $d.a$ represent its problem statement, equation, and answer respectively. D_s consists of instances of the form $(d.p, d.e, d.a)$ while D_w contains instances of the form $(d.p, d.a)$. We extend the Warm model to include a Cross-Entropy loss component for instances belonging to D_s. The net loss is the sum of the REINFORCE (RL_{Warm}) and Cross-Entropy losses shown below:-

Loss 1: $\sum_{d \in D_w} RL_{\text{Warm}}(d.p, d.a)$

Loss 2: $\sum_{d \in D_s} Cross\_Entropy(d.e, \text{Warm}(d.p, d.a))$

Thus, we facilitate semi-supervision through **Loss 2**. That is, we jointly use the equations predicted (by Warm) for datapoints belonging to D_w and the ground truth equations for instances belonging to D_s, for training any state-of-the-art supervised MWP solver.

5 Experimental Setup

In this section, we report details of the experiments on four datasets to examine the performance of the proposed weakly supervised model Warm and its semi-supervised extension Warm-S. We present comparisons with various baselines as well as with fully supervised models.

5.1 Datasets

We perform all our experiments on the publicly available AllArith (Roy and Roth, 2017) and Math23K (Wang et al., 2017) datasets and also on our EW10K and HW10K datasets.For each dataset, we have used a $80:20$ train-test split.

AllArith contains 831 MWPs, annotated with equations and answers. It is populated by collecting problems from smaller datasets, *viz.*, **AI2** (Hosseini et al., 2014), **IL** (Roy and Roth, 2015), **CC** (Roy and Roth, 2015) and **SingleEQ** (Koncel-Kedziorski et al., 2015). All mentions of quantities are normalized to digits. Further, near-duplicate problems (with over 80% match of unigrams and bigrams) are filtered out.

Math23K (Wang et al., 2017) contains 23,161 MWPs in Chinese with 2187 templates, annotated with equations and answers, for elementary school students and is crawled from multiple online education websites. It is the largest publicly available dataset for the task of automatic MWP solving.

EW10K (*c.f.*, Section 3) contains 10,227 MWPs in English and **HW10K** contains 9,896 in Hindi for classes VI to X. We employ a $80:20$ train-test split in each case.

5.2 Dataset Preprocessing

We replace every number token in the problem text with a special word token $< num\_j >$ before providing it as input to the encoder. We also define a set of numerical constants V_{const} to solve those problems which might require special numeric values that may not be present in the problem text. For example, consider the problem "The radius of a circle is 2.5, what is its area?", the solution is "π x 2.5 x 2.5", but the constant quantity π cannot be found in the text. As our model does not use equations as supervision, we cannot know precisely what extra numeric values might be required for a problem, so we fix $V_{const} = \{1, \pi\}$. Finally, the operand dictionary for every problem is initialised as $OpDict = n_P \cup V_{const}$ where n_P is the set of numeric values present in the problem text.

5.3 Implementation Details

We implement[4] all our models in PyTorch (Paszke et al., 2019). We set the dimension of the word embedding layer to 128, and the dimension of the hidden states for other layers to 512. We use the REINFORCE (Williams, 1992) algorithm and Adam (Kingma and Ba, 2014) to optimize the parameters. The initial value of the learning rate is set to 0.001, and the learning rate is multiplied by 0.7 every 75 epochs. We also set the dropout probability to 0.5 and weight decay to 1e-5 to avoid over-fitting. Finally, we set the beam width to 5 in beam exploration.We train our model for 200 epochs with the batch size set to 256.

[4]Source code is attached as supplementary material

5.4 Models

We compare the MWP solving accuracy of our weakly supervised models with beam exploration on the following set of baseline and fully supervised models:

Warm is the proposed weakly supervised approach to equation generation (described from Section 4.1 until 4.4) by employing beam exploration (*c.f.*, Section 4.5).

Warm w/o Beam Exploration is Warm without beam exploration while decoding.

Warm-S is the semi-supervised extension to Warm (*c.f.*, Section 4.6) using beam exploration (Section 4.5).

Warm-S w/o Beam Exploration is the same as Warm-S but does not use beam exploration while decoding.

Random Equation Sampling consists of a random search over k parallel paths of length d. For each path, an operator and its two operands are uniformly sampled from the given vocabulary and the result is added to the operand vocabulary (similar to Warm). The equation is terminated once the correct answer is reached. We set $k = 5$ and $d = 40$ for a fair comparison with our model in terms of the number of search operations.

Seq2Seq Baseline is a GRU (Cho et al., 2014) based seq2seq encoder-decoder model. REINFORCE (Williams, 1992) is used to train the model. Beam exploration is also employed to mitigates issues mentioned in Section 4.5.

LBF (Hong et al., 2021) is a weakly supervised model which uses only answer as supervision by fixing incorrect equation parse trees in each iteration. It subsequently performs training with the fixed trees.

Hybrid model w/ SNI (Wang et al., 2017) is a combination of the retrieval and the RNN-based Seq2Seq models with significant number identification (SNI).

Ensemble model w/ EN (Wang et al., 2018a) is an ensemble model that selects the result according to generation probability across Bi-LSTM, ConvS2S and Transformer Seq2Seq models with equation normalization (EN).

Semantically-Aligned (Chiang and Chen, 2019) is a Seq2Seq model with an encoder designed to understand the semantics of the problem text and a decoder equipped with a stack to facilitate tracking the semantic meanings of the operands.

T-RNN + Retrieval (Wang et al., 2019) is a combination of the retrieval model and the T-RNN model comprising a structure prediction module that predicts the template with unknown operators and an answer generation module that predicts the operators.

Seq2Tree (Liu et al., 2019) is a Seq2Tree model with a Bi-LSTM encoder and a top-down hierarchical tree-structured decoder consisting of an LSTM that makes use of the parent and sibling information fed as the input.

GTS (Xie and Sun, 2019) is a tree-structured neural model that generates the expression tree in a goal-driven manner.

Graph2Tree (Zhang et al., 2020) consists of a graph-based encoder which captures the relationships and order information among the quantities. It also employs a tree-based decoder that generates the expression tree in a goal-driven manner.

As described earlier in Section 4, we use our weakly supervised models (Warm and Warm-S) to generate labelled data (*i.e.*, equations) which we then use to train a supervised model. We have performed experiments using GTS (Xie and Sun, 2019) and Graph2Tree (Zhang et al., 2020) as the supervised models since they are the current state-of-the-art.

6 Results and Analysis

Weakly Supervised Models	AllArith	Math23K	EW10K	HW10K
Warm w/o Beam Exploration	42.1	14.5	57.5	67.3
Warm	**97.4**	**93.8**	**99.3**	**99.5**
Baselines	AllArith	Math23K	EW10K	HW10K
Random Equation Sampling	53.4	17.6	46.3	66.6
Seq2Seq Baseline	67.0	7.1	77.6	75.8

Table 2: Equation generation accuracies of Warm based models compared to baselines. All models are trained using ground truth answers on the training set. Warm outperforms all the remaining models by as significant margin on all the datasets. Evidently, beam exploration significantly improves performance.

6.1 Analyzing Warm

In Table 2, we observe that our model Warm yields far higher accuracy than random baselines with the accuracy values close to 100% on AllArith and EW10K. Thus we are able to more accurately generate equations for a given problem and answer which can then be used to train supervised models. Please note that, in Table 2, we report equation generation accuracies on the training set by training the weakly supervised and baseline models using

Weakly Supervised Models	AllArith	Math23K	EW10K	HW10K
Warm w/o Beam Exploration(GTS)	36.1	12.8	52.6	54.1
Warm (GTS)	66.9	55.3	86.9	81.5
Warm w/o Beam Exploration(Graph2Tree)	48.2	13.5	49.8	58.3
Warm (Graph2Tree)	**68.7**	**56.0**	**87.2**	**82.9**
LBF ‡	51.8	53.6	81.3	75.8
Fully Supervised Models	AllArith	Math23K	EW10K	HW10K
Graph2Tree‡	71.9	75.5	NA	NA
GTS‡	70.5	73.6	NA	NA
Seq2Tree	–	69.0	NA	NA
T-RNN + Retrieval	–	68.7	NA	NA
Semantically-Aligned†	–	65.8	NA	NA
Ensemble model w/ EN	–	68.4	NA	NA
Hybrid model w/ SNI†	–	64.7	NA	NA

Table 3: MWP solving accuracy of Warm-based models compared to various supervised models on AllArith and Math23K datasets. † denotes that result was reported on 5-fold cross validation. All other models are tested on the test set. ‡ denotes that the result is on the same train-test split as ours. "–" denotes code unavailability/reproducibility issues. NA is not applicable.

Problem: Ariel already has 4.0 flowers in her garden, and she can also grow 3.0 flowers with every seed packet she uses. With 2.0 seed packets, how many total flowers can Ariel have in her garden ?
Answer: 10.0
Equation Generated: X=(4.0+(2.0*3.0)) (Correct)
Problem: Celine took a total of 6.0 quizzes over the course of 3.0 weeks. After attending 7.0 weeks of school this quarter, how many quizzes will Celine have taken in total ? Assume the relationship is directly proportional.
Answer: 14.0
Equation Generated: X=(7.0+7.0) (Incorrect)

Table 4: Equation Generated by Warm model

ground truth answers on the training set.

As has been discussed earlier in Section 4.5, our model Warm w/o Beam Exploration suffers from the problem of converging to local optima because of the sparsity of the reward signal. Training our weakly supervised models with beam exploration alleviates the issue to a large extent as we explore the solution space much more extensively and thus partly circumventing the sparsity issue. We observe vast improvement in the training accuracy by introduction of beam exploration.

Problem: Latrell ordered a set of yellow and purple pins. He received 72.0 yellow pins and 8.0 purple pins. What percentage of the pins were yellow?
Equation Generated by WARM (G2T): X=(72.0*(100.0/(72.0+8.0)))(Correct)
Equation Generated by LBF: X=(1.0+(1.0+72.0)) (Incorrect)
Problem: A square barn has a perimeter of 28.0 metres. How long is each side of the barn ?
Equation Generated by WARM(G2T): X=((28.0/2.0)/2.0) (Correct)
Equation Generated by LBF: X=((28.0+28.0)/28.0) (Incorrect)

Table 5: Comparing Warm and LBF model predicted equations

The model Warm yields training accuracy significantly higher than its non-beam-explore counterpart. Warm yields the best training accuracy overall. Since the equation generation accuracies of the baselines reported in Table 2 are far worse,the MWP solving accuracies turn out to be significantly worse - around 8-10%, and hence we do not report them.

We also observe that Warm yields results comparable to the various supervised models without requiring any supervision from gold equations. On AllArith, Warm achieves an accuracy of 66.9% and 68.7% using GTS and Graph2Tree as the supervised models respectively. The state-of-the-art supervised model Graph2Tree yields 71.9%. On Math23k, the difference between Warm and the supervised models is more pronounced. Warm's performance is comparable to that of LBF on Math23k but significantly better on AllArith, EW10Kand HW10K, as evident in Table 3. We have shown a comparison of predicted equations by LBF and Warm in Table 5

In Table 4, we present some predictions. As can be seen, the model is capable of producing long complex equations as well. Sometimes, it may reach the correct answer but through an incorrect equation. *E.g.*: In the last example, the correct equation would have been $X = 7.0 * 6.0/3.0$, but the model predicted $X = 7.0 + 7.0$.

6.2 Analysing Semi-supervision through Warm-S

For analyzing semi-supervision, we combined AllArith (831) with EW10K (10227). We randomly sampled 80% of this data (8846) as our train-set. In retrospect, our train-set consists of 560 instances from AllArith that are completely labelled (amounting to 6.3% of the train-set). We compare our semi-supervised approach against the weakly supervised approach, wherein the entire training data is treated as having only answer labels.

In Table 6, we observe that with less than 10% of fully annotated data, our equation exploration accuracy increases from 56.7% to 92.0% without beam exploration and 99.0% to 99.2% with beam exploration. In Table 7, we also observe a similar trend while training the supervised models; our final MWP solving accuracy increases from 51.2% to 87.4% for Warm w/o Beam Exploration and Graph2Tree as the supervised model. We also study thethere effect of varying amount of com-

plete supervision in Supplementary Section:2.

Weakly Supervised Models	AllArith +EW10K
Warm w/o Beam Exploration	56.7
Warm	99.0
Semi Supervised Models	AllArith+EW10K
Warm-S w/o Beam Exploration	92.0
Warm-S	99.2

Table 6: Equation generation accuracy of Warm-S compared to weakly supervised models and baselines.

Weakly Supervised Models	AllArith +EW10K
Warm w/o Beam Exploration(GTS)	50.2
Warm (GTS)	87.2
Warm w/o Beam Exploration(Graph2Tree)	51.2
Warm (Graph2Tree)	87.8
Semi Supervised Models	AllArith+EW10K
Warm-S w/o Beam Exploration(GTS)	87.2
Warm-S (GTS)	92.1
Warm-S w/o Beam Exploration(Graph2Tree)	87.4
Warm-S (Graph2Tree)	93.6

Table 7: MWP solving accuracy of Warm-S compared to Warm. With semi-supervision, there is a significant increase in accuracy for Warm w/o Beam Exploration, bringing its performance closer to Warm.

7 Conclusion

We have proposed a two step approach to solving math word problems, using only the final answer for supervision. Our weakly supervised approach, Warm, achieves a reasonable accuracy of 56.0 on the standard Math23K dataset even without leveraging equations for supervision. We also curate and release large scale MWP datasets, EW10K, in English and HW10K, in Hindi. We observed that the results are encouraging for simpler MWPs. We also present the benefits of incorporating a semi-supervised extension to Warm.

Acknowledgements

We thank anonymous reviewers and Rishabh Iyer for providing constructive feedback and suggestions. Ganesh Ramakrishnan is grateful to IBM Research, India (specifically the IBM AI Horizon Networks - IIT Bombay initiative) as well as the IIT Bombay Institute Chair Professorship for their support and sponsorship. We would like to acknowledge Saiteja Talluri and Raktim Chaki for their contributions in the initial stages of the work.

References

Jayaprakash Akula, Rishabh Dabral, Preethi Jyothi, and Ganesh Ramakrishnan. 2021. Cross lingual video and text retrieval: A new benchmark dataset and algorithm. In *Proceedings of the 2021 International Conference on Multimodal Interaction*, pages 595–603.

Aida Amini, Saadia Gabriel, Peter Lin, Rik Koncel-Kedziorski, Yejin Choi, and Hannaneh Hajishirzi. 2019. Mathqa: Towards interpretable math word problem solving with operation-based formalisms. *arXiv preprint arXiv:1905.13319*.

Yefim Bakman. 2007. Robust understanding of word problems with extraneous information. *arXiv preprint math/0701393*.

Ting-Rui Chiang and Yun-Nung Chen. 2019. Semantically-aligned equation generation for solving and reasoning math word problems. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*. ACL.

Kyunghyun Cho, Bart van Merriënboer, Caglar Gulcehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. 2014. Learning phrase representations using RNN encoder–decoder for statistical machine translation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1724–1734, Doha, Qatar. Association for Computational Linguistics.

Charles R. Fletcher. 1985. Understanding and solving arithmetic word problems: A computer simulation. *Behavior Research Methods*, 17:565–571.

Yining Hong, Qing Li, Daniel Ciao, Siyuan Haung, and Song-Chun Zhu. 2021. Learning by fixing: Solving math word problems with weak supervision.

Mohammad Javad Hosseini, Hannaneh Hajishirzi, Oren Etzioni, and Nate Kushman. 2014. Learning to solve arithmetic word problems with verb categorization. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 523–533. ACL.

Danqing Huang, Jing Liu, Chin-Yew Lin, and Jian Yin. 2018. Neural math word problem solver with reinforcement learning. In *Proceedings of the 27th International Conference on Computational Linguistics*, pages 213–223. ACL.

Danqing Huang, Shuming Shi, Chin-Yew Lin, Jian Yin, and Wei-Ying Ma. 2016. How well do computers solve math word problems? large-scale dataset construction and evaluation. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 887–896, Berlin, Germany. Association for Computational Linguistics.

Diederik Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. *International Conference on Learning Representations*.

Rik Koncel-Kedziorski, Hannaneh Hajishirzi, Ashish Sabharwal, Oren Etzioni, and Siena Dumas Ang. 2015. Parsing algebraic word problems into equations. *Transactions of the Association for Computational Linguistics*, 3:585–597.

Vishwajeet Kumar, Nitish Joshi, Arijit Mukherjee, Ganesh Ramakrishnan, and Preethi Jyothi. 2019. Cross-lingual training for automatic question generation. *arXiv preprint arXiv:1906.02525*.

Vishwajeet Kumar, Ashish Kulkarni, Pankaj Singh, Ganesh Ramakrishnan, and Ganesh Arnaal. 2015. A machine assisted human translation system for technical documents. In *Proceedings of the 8th International Conference on Knowledge Capture*, pages 1–5.

Vishwajeet Kumar, Ganesh Ramakrishnan, and Yuan-Fang Li. 2018. Putting the horse before the cart: A generator-evaluator framework for question generation from text. *arXiv preprint arXiv:1808.04961*.

Nate Kushman, Yoav Artzi, Luke Zettlemoyer, and Regina Barzilay. 2014. Learning to automatically solve algebra word problems. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 271–281. Association for Computational Linguistics.

Chao-Chun Liang, Kuang-Yi Hsu, Chien-Tsung Huang, Chung-Min Li, Shen-Yu Miao, and Keh-Yih Su. 2016a. A tag-based english math word problem solver with understanding, reasoning and explanation. In *Proceedings of the Demonstrations Session, NAACL HLT 2016*, pages 67–71. The Association for Computational Linguistics.

Chao-Chun Liang, Kuang-Yi Hsu, Chien-Tsung Huang, Chung-Min Li, Shen-Yu Miao, and Keh-Yih Su. 2016b. A tag-based statistical english math word problem solver with understanding, reasoning and explanation. In *Proceedings of the Twenty-Fifth International Joint Conference on Artificial Intelligence*, IJCAI'16, page 4254–4255. AAAI Press.

Qianying Liu, Wenyv Guan, Sujian Li, and Daisuke Kawahara. 2019. Tree-structured decoding for solving math word problems. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 2370–2379.

Arindam Mitra and Chitta Baral. 2016. Learning to use formulas to solve simple arithmetic problems. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 2144–2153. Association for Computational Linguistics.

Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Köpf, Edward Yang, Zach DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. 2019. Pytorch: An imperative style, high-performance deep learning library.

Subhro Roy and Dan Roth. 2015. Solving general arithmetic word problems. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 1743–1752. ACL.

Subhro Roy and Dan Roth. 2017. Unit dependency graph and its application to arithmetic word problem solving. In *Proceedings of the Thirty-First AAAI Conference on Artificial Intelligence*, page 3082–3088. AAAI Press.

Subhro Roy and Dan Roth. 2018. Mapping to declarative knowledge for word problem solving. *Trans. Assoc. Comput. Linguistics*, 6:159–172.

Subhro Roy, Shyam Upadhyay, and Dan Roth. 2016. Equation parsing : Mapping sentences to grounded equations. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing, EMNLP 2016*, pages 1088–1097. The Association for Computational Linguistics.

Jianhao Shen, Yichun Yin, Lin Li, Lifeng Shang, Xin Jiang, Ming Zhang, and Qun Liu. 2021. Generate & rank: A multi-task framework for math word problems. *arXiv preprint arXiv:2109.03034*.

Pankaj Singh, Ashish Kulkarni, Himanshu Ojha, Vishwajeet Kumar, and Ganesh Ramakrishnan. 2016. Building compact lexicons for cross-domain smt by mining near-optimal pattern sets. In *Pacific-Asia Conference on Knowledge Discovery and Data Mining*, pages 290–303. Springer.

Sowmya S Sundaram and Deepak Khemani. 2015. Natural language processing for solving simple word problems. In *Proceedings of the 12th International Conference on Natural Language Processing*, pages 394–402. NLP Association of India.

Ishan Tarunesh, Sushil Khyalia, Vishwajeet Kumar, Ganesh Ramakrishnan, and Preethi Jyothi. 2021. Meta-learning for effective multi-task and multilingual modelling. *arXiv preprint arXiv:2101.10368*.

Shantanu Thakoor, Simoni Shah, Ganesh Ramakrishnan, and Amitabha Sanyal. 2018. Synthesis of programs from multimodal datasets. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 32.

Shyam Upadhyay, Ming-Wei Chang, Kai-Wei Chang, and Wen-tau Yih. 2016. Learning from explicit and implicit supervision jointly for algebra word problems. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 297–306.

Lei Wang, Yan Wang, Deng Cai, Dongxiang Zhang, and Xiaojiang Liu. 2018a. Translating math word problem to expression tree. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 1064–1069. ACL.

Lei Wang, Dongxiang Zhang, Lianli Gao, Jingkuan Song, Long Guo, and Heng Tao Shen. 2018b. Mathdqn: Solving arithmetic word problems via deep reinforcement learning. In *Proceedings of the Thirty-Second AAAI Conference on Artificial Intelligence, AAAI*, pages 5545–5552. AAAI Press.

Lei Wang, Dongxiang Zhang, Jipeng Zhang, Xing Xu, Lianli Gao, Bing Tian Dai, and Heng Tao Shen. 2019. Template-based math word problem solvers with recursive neural networks. In *Proceedings of the Thirty-Third AAAI Conference on Artificial Intelligence, AAAI*, pages 7144–7151. AAAI Press.

Yan Wang, Xiaojiang Liu, and Shuming Shi. 2017. Deep neural solver for math word problems. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 845–854. ACL.

Ronald J. Williams. 1992. Simple statistical gradient-following algorithms for connectionist reinforcement learning. *Mach. Learn.*, 8(3–4):229–256.

Qinzhuo Wu, Qi Zhang, Zhongyu Wei, and Xuan-Jing Huang. 2021. Math word problem solving with explicit numerical values. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 5859–5869.

Zhipeng Xie and Shichao Sun. 2019. A goal-driven tree-structured neural model for math word problems. In *Proceedings of the 28th International Joint Conference on Artificial Intelligence*, pages 5299–5305. AAAI Press.

M. Yuhui, Z. Ying, C. Guangzuo, R. Yun, and H. Ronghuai. 2010. Frame-based calculus of solving arithmetic multi-step addition and subtraction word problems. In *2010 Second International Workshop on Education Technology and Computer Science*, volume 2, pages 476–479.

Dongxiang Zhang, Lei Wang, Nuo Xu, Bing Tian Dai, and Heng Tao Shen. 2018. The gap of semantic parsing: A survey on automatic math word problem solvers. *CoRR*, abs/1808.07290.

Jipeng Zhang, Lei Wang, Roy Ka-Wei Lee, Yi Bin, Yan Wang, Jie Shao, and Ee-Peng Lim. 2020. Graph-to-tree learning for solving math word problems. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 3928–3937, Online. Association for Computational Linguistics.

A Appendix

A.1 Notations

We summarize the notations used in section 4 of the main paper in table 8.

Notation	Description
W	Weight of the FC layers.
o_t^p	Probability distribution of operators at decoding timestep t.
o_t^l	Probability distribution of the left operand at decoding timestep t.
o_t^r	Probability distribution of the right operand at decoding timestep t.
h_t^d	Decoder hidden state at timestep t.
Em_{op}	Operator Embedding Matrix
h_t^{op}	Hidden state for the operator at timestep t
op_t	Operator sampled from o_t^p.
h_t^{ol}	Hidden state for the left operand at timestep t.
ol_t	Left operand sampled from o_t^l.
h_t^{or}	Hidden state for the right operator at timestep t.
or_t	Right operator sampled from o_t^r.
OpDict	Operand dictionary used while decoding
R_t	Rewards obtained at timestep t.
$p_\theta(y_t)$	Probability of generating $y_t = (op_t, ol_t, or_t)$ at timestep t.

Table 8: Summary of notation used.

A.2 Ablation Study: Varying Amount of Semi-supervision

We performed an experiment to study the effect of different amounts of supervision by varying the number of instances in training set we treat as fully labelled. The number of fully labelled instances is X-axis*80. We observe that just having 160 equation-labelled instances (out of 8846 ie. 1.8%) improves the equation-exploration accuracy significantly (46.7% to 90.6%) when we don't use beam exploration.

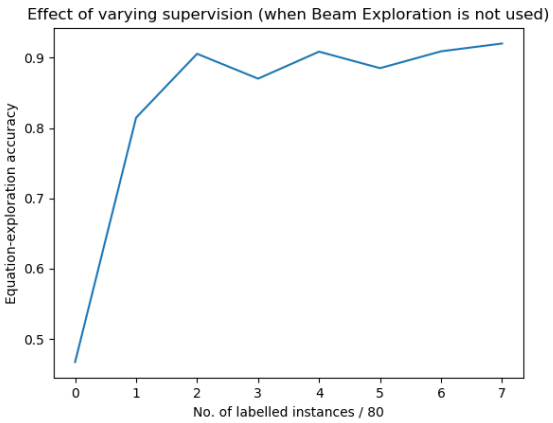

Figure 4: Equation Exploration accuracy with varying supervision

A.3 Infrastructre Details

GPU Model used :
1)Model number: GeForce GTX 1080 Ti
2)Memory : 12GB

Training time :
1) WARM takes 4 hours for training
2) G2T takes 1 hour and 30 minutes to get trained completely

Attention Networks for Augmenting Clinical Text with Support Sets for Diagnosis Prediction

Paul Grundmann*, **Tom Oberhauser***, **Felix Gers and Alexander Löser**
Berliner Hochschule für Technik
Luxemburger Str. 10, 13353 Berlin
{pgrundmann, toberhauser, gers, aloeser}@bht-berlin.de
**Both authors contributed equally.*

Abstract

Diagnosis prediction on admission notes is a core clinical task. However, these notes may incompletely describe the patient. Also, clinical language models may suffer from idiosyncratic language or imbalanced vocabulary for describing diseases or symptoms. We tackle the task of diagnosis prediction, which consists of predicting future patient diagnoses from clinical texts at the time of admission. We improve the performance on this task by introducing an additional signal from support sets of diagnostic codes from prior admissions or as they emerge during differential diagnosis. To enhance the robustness of diagnosis prediction methods, we propose to augment clinical text with potentially complementary set data from diagnosis codes from previous patient visits or from codes that emerge from the current admission as they become available through diagnostics. We discuss novel attention network architectures and augmentation strategies to solve this problem. Our experiments reveal that support sets improve the performance drastically to predict less common diagnosis codes. Our approach clearly outperforms the previous state-of-the-art PubMedBERT baseline by up 3% points. Furthermore, we find that support sets drastically improve the performance for pregnancy- and gynecology-related diagnoses up to 32.9 % points compared to the baseline.

1 Introduction

Pre-trained large language models such as ClinicalBERT (Alsentzer et al., 2019) or PubMedBERT (Gu et al., 2021) are commonly used in the medical domain to predict diagnoses from admission notes (Hashir and Sawhney, 2020; Sushil et al., 2018a; van Aken et al., 2021). Admission and discharge notes are valuable information sources about doctors' decisions about patients and the outcomes. However, the vocabulary in these notes is often insufficient to describe the patients' clinical phenotype fully. Also, clinical text frequently contains idiosyncratic vocabulary, uncommon abbreviations and differs from clinic to clinic in writing style. Moreover, evidence for clinical diseases in the text is imbalanced. In particular for less common or even rare diseases (see also Figure 1) not much text evidence exists. Finally, pre-trained language models can suffer from limited access to training data because of silos or data-privacy concerns. These factors can lead to poor performance in predicting outcomes on clinical text with pre-trained language models.

Multimodal Patient Representation. Miotto et al. (2016) and Topol (2019) therefore propose to augment text with potential complementary multimodal data into a reusable *deep patient representation* to improve clinical prediction tasks. For example, recent work surveys to augment text with image data (Esteva et al., 2021), with complementary medical text books (van Aken et al., 2021), ontologies (Cai et al., 2020) or time series data (Yang and Wu, 2021).

Improving predictions with text and set data. A particularly powerful source to augment clinical text are sets of diagnosis codes from previous visits of the patient or upcoming hypotheses of the treating physician during the patients' treatment. For example, a patient in an ICU scenario receives on average a set of more than ten diagnosis codes at discharge time, see also Table 2. These sets match a patients' previous state against a common ontology, such as ICD or CCS medical nomenclature. Therefore, these sets are a rich and potentially complementary knowledge source for a patient representation. To our best knowledge this is the first work on augmenting clinical text for diagnosis prediction with such sets. Figure 2 illustrates this novel task in detail: Given is the admission note containing details on chief complaints, present illness, medication, physical examination and family or social history. In addition, the system receives a

Proceedings of the 29th International Conference on Computational Linguistics, pages 4765–4775
October 12–17, 2022.

support set of additional diagnosis codes observed for this patient in the past or during the current treatment. Given both inputs, the final task is to predict the likely diagnostic outcome for the patient at discharge time.

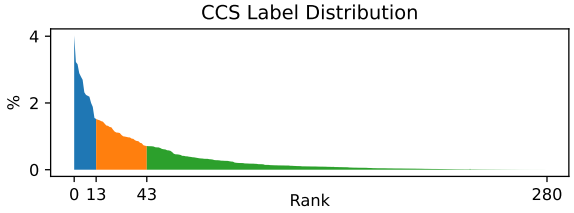

Figure 1: The distribution of CCS labels of the full dataset. The top 13 codes form the short head of the distribution. The middle tail is 30 codes wide, while the remaining 237 codes form the long tail.

Contribution. Augmenting text with set data is a complex knowledge integration problem. Ideally, a multi-modal representation from both, text and set data, is much more powerful for large multi-label classification tasks, such as diagnosis prediction, than each knowledge representation on its own. For solving this problem, our major contributions are: **(1)** We represent text and set data in two different latent vector spaces. This also includes investigating different sampling methods of set elements for learning embeddings during training. Optionally, we enrich disease codes in sets with additional textual information from UMLS, a medical ontology, and Wikidata. **(2)** We propose three different novel network architectures for augmenting knowledge from text with set data, including pooled- and full text attention as well as a dual stack encoder. **(3)** In a rigid experimental setting, we compare these architectures against each other and two strong baselines. We also report prediction results in particular for infrequent diseases on the MIMIC-III data set with approximately 60.000 admissions and more than 2 million clinical notes. The remainder of this paper is structured as follows: We review related work in Section 2. In Section 3, we explore task and data set characteristics, from which we justify in Section 4 our novel network architecture design. Section 5 reports our quantitative evaluation, followed by result discussion and an error analysis in Section 6. Finally, we conclude in Section 7.

2 Related Work

There is a large amount of work focusing on diagnosis prediction from EHR data, especially clinical codes. Furthermore, there is an increasing emphasis on incorporating text or multi-modal data from clinical notes. We distinguish ourselves, particularly from work in ICD coding, since only information at the time of admission is used for our considered tasks. ICD coding, on the other hand, uses all data available at discharge time.

Diagnosis prediction on codes. Choi et al. (2016) use a reverse time attention mechanism on the diagnosis and procedure codes of the patients' history for the task of heart failure prediction. Ma et al. (2017) use a bidirectional recurrent neural network (RNN) on the diagnosis and procedure codes of the patients' history to predict diagnosis codes for the next admission. Later they apply graph-based attention to incorporate the knowledge of a medical knowledge graph to learn medical representations (Ma et al., 2018). Peng et al. (2020) use a self-attention mechanism on the diagnosis codes to capture contextual and temporal relations within the patients' journey to predict the second hierarchy of the ICD-9 codes.

Clinical text for diagnosis prediction. Boag et al. (2018) evaluate the usefulness of different simple text representations for diagnosis prediction and show that the text itself contains valuable information. Sushil et al. (2018b) use stacked denoising autoencoders combined with a paragraph vector model to learn patient representations. van Aken et al. (2021) simulate patients at admission time by only using parts of the textual descriptions known at admission time, such as *"Chief complaint"* or *"Medical history"*. Winter et al. (2022) apply knowledge graphs to retrain and instill attention heads with complementary structured domain knowledge for clinical outcome prediction from text. Papaioannou et al. (2022) seek to embed complementary knowledge to increase the performance on low resource languages by consecutive fine-tuning of multi-lingual models.

Multimodal diagnosis prediction. Lipton et al. (2016) use an LSTM architecture on 13 time-series variables like blood pressure or heart rate. Liu et al. (2018) use free, unstructured text from medical notes and structured clinical information such as numerical lab and vital sign values to predict a small set of specific chronic diseases. Qiao et al. (2019) use RNNs and attention to mix code- and text features for readmission diagnosis prediction from prior admissions. In contrast to the aforemen-

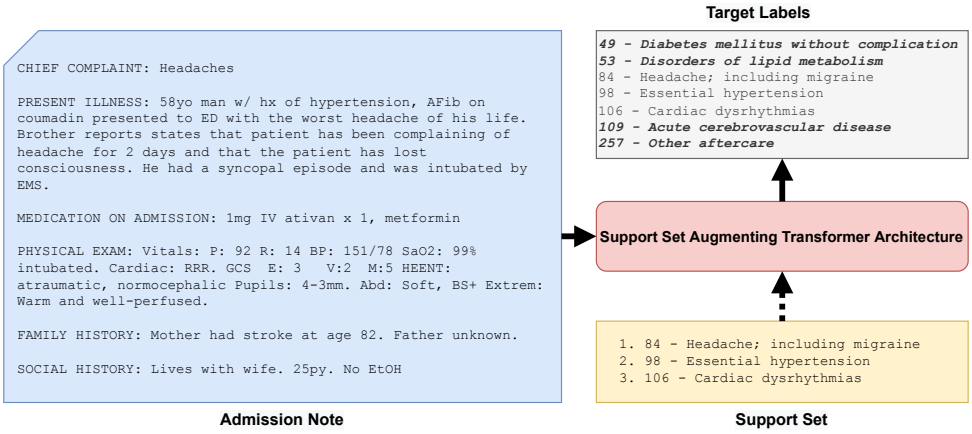

Figure 2: Illustration of the set-augmented diagnosis prediction task: Given an admission note and an optional support set of diagnosis codes, our architecture aims to predict the diagnostic outcome for the patient at discharge time. Our basic observation is that the text data is only partially representing the patient, might be noisy and diagnostic codes can complement observations from the clinical text. Combining information from both sides will lead to improved understanding of a latent model towards clinical health conditions.

tioned approaches, we aim to model interdependencies between diagnostic codes and mix them with state-of-the-art text representations of clinical admission notes. Further, we seek to implement an interactive system that allows verifying hypotheses by systematically adding or removing diagnostic codes in combination with a current admission note.

Distinction from previous work. In contrast to the aforementioned approaches, we aim to model interdependencies between diagnostic codes and mix them with state-of-the-art text representations of clinical admission notes. Further, we seek to implement an interactive system that allows verifying hypotheses by systematically adding or removing diagnostic codes in combination with a current admission note. Our model does not rely on the existence of codes but instead uses them as they become available during a diagnostic process or through prior admissions to refine the classification result. Up to our knowledge, there is only related work that uses RNNs / LSTMs (Qiao et al., 2019) for representing the set embeddings. Modeling the set embeddings with RNNs is problematic because they introduce temporal dependencies between the diagnosis code inputs. In consequence, they treat the diagnosis code sets as a sequence. Most of the time, those temporal dependencies are not reflected in the available data.

3 Tasks and Datasets

In the following section, we introduce the tasks of diagnosis and readmission diagnosis prediction and describe our medical dataset.

Prediction from clinical text and diagnosis sets. Following van Aken et al. (2021), our model aims to predict diagnostic codes assigned to a patients' admission after their discharge with the constraint of using only information available at admission time [1]. In addition, we allow the model to leverage an optional set of support codes to refine the classification process. In the real-world use of our model, these supporting codes would originate from a doctors' hypotheses, evident diagnoses or from a diagnosis of a doctor outside the clinic the patient has visited before, such as the family doctor. Formally, our training data consists of a set of admissions $\mathbb{A}$ where $A_i = (T_i, S_i, C_i)$, $A_i \in \mathbb{A}$. $T_i = (t_1, \ldots, t_n)$ is the text of the admission note with a sequence length of n tokens for a patient at admission time. $C_i \subset \mathbb{C}$ is the prediction target of diagnostic codes from the label space $\mathbb{C}$ and $S_i \subset C_i$ is the support set.

Readmission diagnosis prediction. We consider the readmission diagnosis prediction task to further simulate and evaluate a real-world diagnostic process. In contrast to the diagnosis prediction task, the support set $S_i \subset C_{i-1}$ consists of the diagnoses of the last clinical admission C_{i-1} of

[1] https://github.com/bvanaken/clinical-outcome-prediction

the same patient from the patients' journey that consists of m admissions $P = \{A_{0_p}, \ldots, A_{m_p}\}$, where $A_i = (T_i, S_i, C_i)$. T_i is the admission note and $C_i \subset \mathbb{C}$ is the set of diagnostic codes. The motivation behind this is to integrate prior knowledge about the patient from his former admissions at the same hospital. As an additional difficulty, the model must compensate at this point for the fact that the codes from the previous admission are not necessarily supporting the diagnosis prediction, nor do the codes from the current admission functionally depend on them.

Clinical admissions and discharge summaries. We use the freely available Medical Information Mart for Intensive Care v1.4 database (MIMIC-III) (Johnson et al., 2016), containing de-identified electronic health record data (EHR), including textual discharge summaries in English of the Beth Israel Deaconess Medical Center in Massachusetts between 2001 and 2012. Following van Aken et al. (2021), we filter those textual discharge summaries by sections known at admission time, like "Chief complaint," "Medical history," or "Admission medications." The diagnostic codes associated with those admissions are using the ICD-9-CM format. Since ICD-9-CM is a very fine-grained medical coding standard, we aggregate the label space using the Clinical Classifications Software (CCS) for ICD-9-CM [2], which merges similar ICD-9-CM codes into a categorical group. Table 1 provides an overview of the dataset statistics. We also use MIMIC-III for the task of readmission diagnosis prediction but focus only on patients with more than one admission. Statistics about this subset are shown in Table 2.

	Total	Train	Val	Test
Admissions	48741	33994	4918	9829
Min. tokens	28	29	31	28
Max. tokens	17034	17034	4039	3304
∅ Tokens	641	640	635	647
Min. diagnoses	1	1	1	1
Max. diagnoses	34	34	33	33
∅ diagnoses	10.41	10.40	10.32	10.50
Unique diagnoses	280	279	266	272

Table 1: Statistics of our dataset for the task of diagnosis prediction. Very rare codes might appear only in one of the three splits.

[2]https://www.hcup-us.ahrq.gov/toolssoftware/ccs/ccs.jsp

	Total	Train	Test
Admissions	18785	13785	5000
∅ Diag. / patient	11.55	10.87	13.45
∅ New diag. / patient	8.03	7.58	9.28
∅ Lost diag. / patient	3.33	3.25	3.56
∅ Persistent diag. / patient	3.52	3.28	4.17
Unique diagnoses	270	268	256

Table 2: Statistics of our dataset for the task of readmission diagnosis prediction. On average, patients keep 3.52 of their previous diagnosis codes only. Between two admission, a patient no longer shows symptoms for an average of one-third of their previously annotated diagnosis codes.

4 Models

Augmenting text with set embeddings. For the task of set-augmented diagnosis prediction, we require a network architecture that is able to combine two possibly complementary information sources from different modalities: Clinical text from admission notes and sets of diagnosis codes. Also, the architecture must be able to learn a meaningful representation from a few examples without catastrophic forgetting in the underlying pre-trained language model. The attention mechanism (Bahdanau et al., 2015; Kim et al., 2017) allows the model to base its decision on a fine granular selection of the information in the two input spaces and to ignore less important elements. Thus, it enables the model to enrich the incomplete text representations with knowledge from the support set.

4.1 Novel Architectures

We apply three different transformer-based architectures (s. Figure 3) to incorporate knowledge from support sets to enhance the models' prediction. We preserve the permutation invariance of the set of added codes by feeding them directly into the transformer and omitting the positional encoding. Moreover, following (Devlin et al., 2019), we add the special token `[NULL]` to every support set. The `[NULL]` token also serves as an aggregate representation for the support set.

Pooled Attention. In the *pooled attention architecture* (s. Figure 3) we use the last hidden state of the `[CLS]` token as a pooling mechanism $pool()$ to aggregate the information from all tokens in the text into a single embedding. With the pooled attention architecture, we aim to compress the admission note into a meaningful single vector text representa-

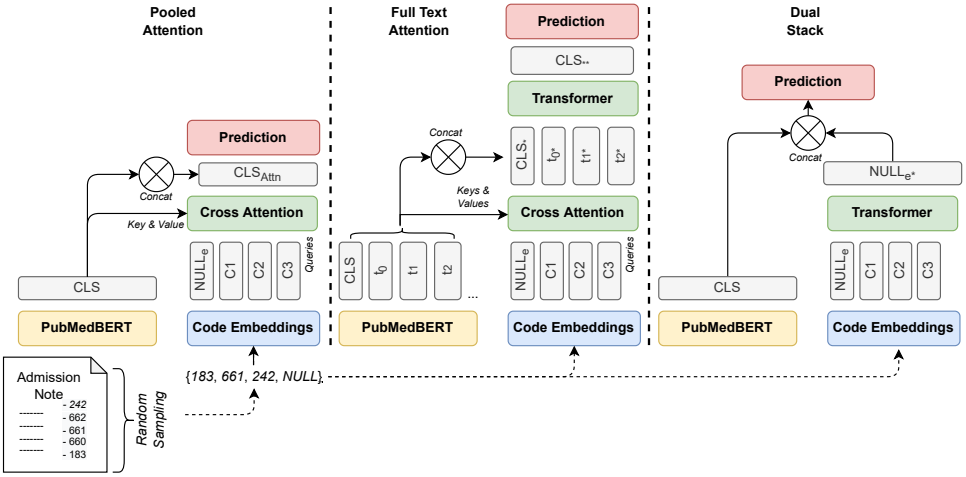

Figure 3: Support Set Augmentation Architectures with different levels of attention between text and support code features showing the Pooled Attention, Full Text Attention and Dual Stack architecture from left to right.

tion that contains all necessary information to solve the diagnosis prediction task. We project the text T_i from admission A_i into an embedding G.

$$G = pool(BERT(T_i)) \tag{1}$$

Furthermore, we apply attention between the `[CLS]` token and the input code embeddings. We use a transformer to learn shared features between the elements in the support set. Typically, the attention mechanism uses a softmax function to normalize the attention scores (Vaswani et al., 2017). However, the softmax limits the information flow to a single code embedding. Following Gülçehre et al. (2019), we replace the softmax function with a sigmoid activation σ and define our cross attention with queries Q, keys K and values V as follows:

$$\begin{aligned} \text{Cross Attention} &= \sigma(QK^T)V \\ Q = G \cdot W_Q, K = G \cdot W_K, V &= D \cdot W_V \end{aligned} \tag{2}$$

We denote the admission note representation by Q and K and the code representations by V and linearly transform them with learned weight matrices W_Q, W_K and W_V. The sigmoid function allows information to flow between all code embeddings and the admission note representation. Finally, we use a skipthrough connection and concatenate the output of the attention layer with the `[CLS]` representation to minimize information loss and to avoid catastrophic forgetting on the text encoder side.

Full Text Attention. With the *full text attention architecture*, we aim to reduce the potential information loss in the aggregation step in contrast to the pooled attention model. The architecture applies softmax attention between all tokens of the admission note and all codes in the support set. We use an additional transformer on top of the attended admission note tokens and use the resulting output of the `[CLS]` token for the prediction step. In distinction to the pooled attention, we only add the `[NULL]` token for empty support sets S_i.

Dual Stack. Finally, we experiment with a less complex and, compared to the full text attention model, computationally more efficient *dual stack architecture* that does not involve an attention mechanism to mix the support set with the text embedding. Instead, it consists of two independent encoders: one for the admission note and one for the support set. We use a BERT architecture as the admission note encoder and train a multi-head transformer for the support set representation. To combine the information from both information spaces, we concatenate the embeddings from both encoders and feed them into the prediction layer.

Loss function. We optimize all models by minimizing the multi-label binary cross entropy loss between the predictions p and the target labels y:

$$L = -\frac{1}{N}\sum_{i=1}^{N}\sum_{j=1}^{M} y_{ij} log(p_{ij}) \tag{3}$$

where M is the number of diagnosis codes and N the number of admission notes.

4.2 Augmenting Codes with Ontologies

The diagnosis code distribution from Figure 1 shows that there are very few training samples for codes from the long tail. This raises the concern

that the model might not be able to learn a meaningful representation for each code. We initialise our diagnosis sets with additional textual information about diagnosis codes from medical ontologies to address this issue. We use three different data sources: **(1)** We obtain the descriptive name for every diagnostic code from the MIMIC-III database and map it to the respective CCS code. In addition, **(2)** we use textual definitions from the Unified Medical Language System (UMLS) 2021AB (Bodenreider, 2004), which provides comprehensive full-text definitions for 22.1% of the diagnostic codes present in our data. Furthermore **(3)**, for additional 4.7% of the codes we use descriptions from the Wikidata knowledge graph.

4.3 Baselines

We compare our approaches against two powerful baselines representing state-of-the-art approaches to diagnosis prediction with either text or set data.

Baseline: PubMedBERT We compare our approach to a PubMedBERT(Gu et al., 2021) based classifier which incorporates only textual information from the admission note N_i. We use PubMedBERT instead of ClinicalBERT (Alsentzer et al., 2019) because ClinicalBERT was pre-trained on MIMIC-III notes and therefore already contains knowledge from the discharge notes, leading to an unfair advantage in the diagnosis prediction task from admission notes only. We fine-tune the model and use the last hidden state of the `[CLS]` token to predict the diagnostic codes.

Baseline: Support Set Transformer This architecture only incorporates knowledge from the support set. We use the transformer based architecture from the dual stacks' support set encoder (s. Figure 3) and use only the support set S_i from the admission to predict the remaining annotated diagnostic codes C_i. Similar to the dual stack approach, we aggregate the information of the input set into a single embedding by adding a special token `[NULL]`.

4.4 Hyperparameter Setup

We use PubMedBERT (Gu et al., 2021) as a text encoder for all text-related components of our architecture such as the admission note encoder or the ontology-knowledge augmented set encoder. We use the Adam optimizer (Kingma and Ba, 2015) with a weight decay of 0.01. Our code and hyperparameters are publicly available[3]. Furthermore, we performed a hyperparameter optimization for all architectures and also report details regarding the tuned parameters in the appendix. To prevent catastrophic forgetting in the pre-trained text encoder, we use a lower learning rate of 2e-5 for the weights of the BERT model. Due to the sequence length limitation of PubMedBERT, we truncate all admission notes to 512 tokens. We use a code embedding $\in \mathbb{R}^{768}$. The transformer in the full text attention model consists of four layers with two attention heads each. The dual stack model uses a transformer composed of one attention layer with 12 heads. We sample three annotated codes from the admission note for the diagnosis prediction task, which produced optimal results during training based on our HPO. In the readmission task we use all codes from the previous admission or the `[NULL]` token for the first admission.

5 Evaluation

Metrics. We measure the performance of our experiments in macro averaged *AUROC* (area under the receiver operating characteristic curve) and *mAP* (mean average precision). Because the supplied support set S_i in the diagnosis prediction task is part of the target label space $S_i \subset C_i$, it provides the support set augmented architectures an unfair advantage over the baselines to evaluate on codes $\in S_i$. Therefore, we only evaluate our approach on $y = C_i \setminus S_i$ to exclude the advantage of provided codes and to avoid that codes from the support set S_i are determined as correct predictions.

5.1 Results

We report scores of our quantitative evaluation in Table 3 and use a support set of five codes for the diagnosis prediction task to augment the admission note. *Set Embeddings* denote the combined representation of set and text data. *Semantic Set Embeddings* contain codes enriched with ontology knowledge as described in Section 4.2 and the admission notes' text. In addition to mAP and AUROC, we also report the standard error over five runs for the diagnosis prediction task because it involved random sampling to generate the support sets.

Novel models outperform baselines. We report that all of our approaches outperform the baselines,

[3] https://github.com/DATEXIS/ClinicalSupportSetAugmentation

	Model	Diagnosis Prediction AUROC	Diagnosis Prediction mAP	Readmission Task AUROC	Readmission Task mAP
Baselines	Support Set Transformer	75.52 ±1.7e-3	30.51 ±5.6e-4	66.66	44.33
	PubMedBERT	84.67 ±7.0e-4	47.39 ±7.8e-4	79.98	59.32
Set Embeddings	Full Text Attention	86.93 ±2.9e-4	49.12 ±9.2e-4	**81.37**	59.74
	Pooled Attention	87.08 ±8.2e-4	48.96 ±6.0e-4	81.06	**60.59**
	Dual Stack	87.10 ±7.0e-4	48.95 ±4.0e-4	81.01	60.54
Semantic Set Embeddings	Full Text Attention	**87.24** ±1.0e-3	**49.66** ±9.0e-4	81.03	59.61
	Pooled Attention	87.21 ±7.3e-4	48.85 ±1.5e-4	80.95	59.00
	Dual Stack	87.18 ±1.2e-3	48.67 ±2.4e-4	81.03	59.35

Table 3: Results on the diagnosis- and readmission diagnosis prediction task in macro averaged AUROC and mAP. All of our proposed architectures outperform the baselines. The full text attention model with semantic initialization of the diagnosis code embeddings performs best on the diagnosis prediction task. Semantic integration especially helps the task of diagnosis prediction, while learned set embeddings perform better for the readmission task.

emphasizing our hypothesis that augmenting admission notes with support sets containing diagnostic information helps. However, we observe that there is little difference in the performance between our proposed architectures. In general, the full text attention model performs best. Using the text and the support set in combination leads to an average improvement of around 2-3 points in AUROC or 2.5 points in mAP compared to the PubMedBERT baseline.

Minor gains with semantic set embeddings. We observe a slight increase in performance by integrating semantic knowledge (s. Table 3). However, the co-occurrence of codes within the admissions seems to have a much more substantial impact on the final classification performance than the additional semantic information. This slight increase indicates that the architecture can leverage the additional information, but the semantics encoded in the ICD names and UMLS definitions do not seem to contain much complementary knowledge.

Rare and very frequent codes are most effective. We analyze the impact of the diagnosis code frequency on the prediction performance. We perform ten evaluations with each three random sampled codes in the support set, binned by frequency. We measure the performance difference between our model and the PubMedBERT baseline on the remaining codes and plot the standard error for those observations (s. Figure 4). We observe that both rare codes from the end of the long-tail and frequent codes belonging to the second tertile improve the prediction by almost five points in mAP. We hypothesize that especially rare codes create a major distinctive factor for a diagnosis where the machine assigns a high weight. Contrary, obvious and frequent diagnoses may create a bias and thus have the highest impact on the prediction. The smaller increase in performance between frequency bins 50 to 250 indicates partial to non-existent complementarity between text and support set. In general, the effect of diminishing performance difference with decreasing frequency can be explained by few training examples for the given code and thus an insufficiently learned representation because MIMIC-III primarily focuses on severely ill patients that require life-saving measures at the ICU.

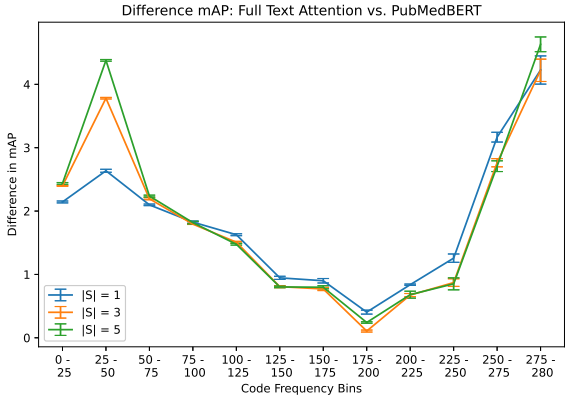

Figure 4: Performance difference between PubMedBERT and full text attention with support sets of size 1, 3, and 5 codes across different frequency bins.

Random sampling for compensating imbalance. The CCS label distribution follows a power-law distribution pattern (s. Figure 1). To compensate effects of such an imbalanced label distribution, we evaluate random sampling vs. inverse frequency weighted random sampling to create potentially more balanced support sets during training. We find that inverse frequency weighted random sampling, in general, performs worse than random sampling

$\|S_i\|$	0-13	13-43	43-280
0	80.66	82.03	86.41
1	81.61	83.19	87.94
2	82.08	83.61	**88.36**
3	**82.54**	**84.03**	87.91
PubMedBERT	79.28	81.09	85.73

Table 4: Prediction performance over different code frequencies measured in AUROC with different support set sizes using random sampling and the full-text attention model with set embeddings. Frequencies split in tertiles (s. Figure 1) according to the code distribution in the admission note dataset. The performance increases with growing support set sizes $|S_i|$.

that follows the label distribution in the training data. We observe a difference in mAP of *-0.3*, *-1.8*, and *-1.3* points for the full text, pooled, and dual stack architectures compared to training with random sampling. Our explanation is that random sampling focuses more on the short head of the label distribution. Therefore, the model learns a richer representation for codes that are more frequent in the dataset and therefore provides better support for common predictions (s. Figure 4).

Larger sets can be beneficial. Table 4 shows that increasing the number of elements in the support set improves the performance for codes of all frequencies. Especially in the range of the 25-50 most frequent codes (s. Figure 4), larger support set sizes lead to the most performance improvement. Finally, we observe that even with $|S_i| = 0$, our model outperforms the PubMedBERT baseline, which indicates that the model stores valuable information in the `[NULL]` token.

6 Discussion

Clever Hans problem for readmissions. Language models often just learn effective shortcuts of high dimensional data distributions instead of generalizing, which is called *Clever Hans problem* (Lapuschkin et al., 2019). We expect to find a variant of the Clever Hans problem for the readmission diagnosis prediction task: Here, we expect the model to learn the shortcut of copying codes from the support set into the diagnosis predictions instead of learning novel correlations from the potentially complementary text data. Indeed, our model copies in 78.1% of the test cases, on average, 2.12 codes from the support set that are not in the target label set. However, this is only a tiny fraction of the average of 13.45 codes in each support set (s. Table 2). This contradicts the Clever Hans problem and empirically confirms the ability of our model to ignore unrelated information from the support set and, in those cases, to focus more on the admission note.

Beneficial and non-beneficial codes. In our evaluation with clinical doctors we observe certain codes that improve the prediction more than others. We find that these codes are typically rare, such as code *188* (s. Table 5), but improve the mAP relative to PubMedBERT by more than 30 points. Likewise, rare codes can also have a diminishing effect on the prediction performance. We hypothesize that their representation is not well initialized due to the lack of training examples. In Table 5, we show the most helpful and most unhelpful codes and their rank in the dataset. We find that 221 codes improve the prediction by, on average, 3.02%. 51 codes decrease the prediction performance by, on average, 2.29%.

CCS Code	ΔmAP	Rank
188 - Fetopelvic disproportion; obstruction	+32.9	269
187 - Malposition; malpresentation	+25.8	259
191 - Polyhydramnios and other problems of amniotic cavity	+24.0	270
31 - Cancer of other male genital organs	+23.4	272
184 - Early or threatened labor	+21.3	242
..		
119 - Varicose veins of lower extremity	-05.2	234
218 - Liveborn	-05.4	267
655 - Disorders usually diagnosed in infancy, childhood, or adolescence	-05.9	247
124 - Acute and chronic tonsillitis	-09.6	257
177 - Spontaneous abortion	-15.3	261

Table 5: Excerpt of all codes in the support set that have the most effect on prediction performance compared to PubMedBERT ranked by mAP difference. Codes from the long tail have the highest impact on performance.

Additional set data can compensate problems of language models with idiosyncratic language. Commonly, large language models often have difficulties with domain-specific language (Liu et al., 2020). For example, pregnancy is often encoded in an idiosyncratic manner like "G2P1," which stands for *gravida 2 para 1*, which means that this is the

second pregnancy and the first one's result was a life-born child. Also, abbreviations such as *"PNV"* for prenatal vitamins are usually the only indicator of pregnancy. Often language models do not have seen sufficient context information during training to generalize from these words to infer higher-level concepts. We analyze the effect of the most influential codes, from which 15 of 20 are either pregnancy or gynecology related. We measure the difference in rank with and without these codes in the support set. Of all the codes that benefit from adding these codes, 73.90% are pregnancy or gynecology-related as well. Our results indicate that adding pregnancy-related codes to the support set helps the model recognize the concept of pregnancy and compensates for problems of idiosyncratic language.

Future improvements. It is interesting to see if graph neural networks can lead to improved representation through updates of related codes. Furthermore, it is possible to experiment with negative examples of diagnostic codes and additional encodings to represent the time between two or more admissions. Given the sparse training data situation presented in medical data silos, particular focus should be applied to zero-shot or few-shot cases, e.g., codes that occur the first time or are rarely represented in prior admissions.

7 Conclusion

Augmenting text with set data is an important problem, in particular in the clinical domain with multimodal patient representations. To solve this problem, we propose novel attention-based network architectures. Our results clearly show that in a clinical prediction task, the augmented representation outperforms a language model, particularly for predicting less common diseases. We also observe that complementary data from sets can for compensate shortcomings of language models, such as idiosyncratic language or abbreviations.

8 Ethical Considerations

Models for diagnosis prediction based on clinical admission notes can be a valuable component of clinical decision support systems that aim to assist medical professionals during their differential diagnosis. Hence, those models bear the potential to save lives by preventing inexperienced doctors from overlooking rare or unusual symptoms. They might as well save cost and reduce the amount of time required for the diagnosis process of medical professionals. However, admission notes and billing codes such as ICD-9 are only a very limited and biased perspective on the patient. Admission notes leave out important diagnostics performed during the patients' stay. Furthermore, billing codes are a suboptimal target label space. They are used to obtain the maximum possible reimbursement for the cost of treatment. There is a risk that patients will receive an excessive number of codes and, therefore, might be over-coded. Likewise, under-coded patients may also occur. To deduce clinical outcome solely from admission notes without having medical professionals perform an iterative differential diagnosis process raises the concern that certain very significant signals may never be introduced to the model.

9 Acknowledgements

We would like to thank the editors and anonymous reviewers for their helpful suggestions and comments. Furthermore, we would like to thank Tom Röhr for his support throughout the project. This work was founded by the German Federal Ministry for Economic Affairs and Energy (BMWi) under grant agreements 01MK2008MD (Service-Meister) and 01MD19003B (PLASS), as well as the Federal Ministry of Education and Research (BMBF) under grant agreement 16SV8845 (KIP-SDM).

References

Emily Alsentzer, John R. Murphy, Willie Boag, Wei-Hung Weng, Di Jin, Tristan Naumann, and Matthew B. A. McDermott. 2019. Publicly available clinical BERT embeddings. *CoRR*, abs/1904.03323.

Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. 2015. Neural machine translation by jointly learning to align and translate. In *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*.

Willie Boag, Dustin Doss, Tristan Naumann, and Peter Szolovits. 2018. What's in a note? unpacking predictive value in clinical note representations. *AMIA Summits on Translational Science Proceedings*, 2018:26.

Olivier Bodenreider. 2004. The unified medical language system (umls): integrating biomedical terminology. *Nucleic acids research*, 32(suppl_1):D267–D270.

Linkun Cai, Yu Song, Tao Liu, and Kunli Zhang. 2020. A hybrid BERT model that incorporates label se-

mantics via adjustive attention for multi-label text classification. *IEEE Access*, 8:152183–152192.

Edward Choi, Mohammad Taha Bahadori, Jimeng Sun, Joshua Kulas, Andy Schuetz, and Walter F. Stewart. 2016. RETAIN: an interpretable predictive model for healthcare using reverse time attention mechanism. In *Advances in Neural Information Processing Systems 29: Annual Conference on Neural Information Processing Systems 2016, December 5-10, 2016, Barcelona, Spain*, pages 3504–3512.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT 2019, Minneapolis, MN, USA, June 2-7, 2019, Volume 1 (Long and Short Papers)*, pages 4171–4186. Association for Computational Linguistics.

Andre Esteva, Katherine Chou, Serena Yeung, Nikhil Naik, Ali Madani, Ali Mottaghi, Yun Liu, Eric J. Topol, Jeff Dean, and Richard Socher. 2021. Deep learning-enabled medical computer vision. *NPJ Digital Medicine*, 4.

Yu Gu, Robert Tinn, Hao Cheng, Michael Lucas, Naoto Usuyama, Xiaodong Liu, Tristan Naumann, Jianfeng Gao, and Hoifung Poon. 2021. Domain-specific language model pretraining for biomedical natural language processing. *ACM Trans. Comput. Healthcare*, 3(1).

Çaglar Gülçehre, Misha Denil, Mateusz Malinowski, Ali Razavi, Razvan Pascanu, Karl Moritz Hermann, Peter W. Battaglia, Victor Bapst, David Raposo, Adam Santoro, and Nando de Freitas. 2019. Hyperbolic attention networks. In *7th International Conference on Learning Representations, ICLR 2019, New Orleans, LA, USA, May 6-9, 2019*. OpenReview.net.

Mohammad Hashir and Rapinder Sawhney. 2020. Towards unstructured mortality prediction with free-text clinical notes. *Journal of Biomedical Informatics*, 108:103489.

Alistair EW Johnson, Tom J Pollard, Lu Shen, Li-wei H Lehman, Mengling Feng, Mohammad Ghassemi, Benjamin Moody, Peter Szolovits, Leo Anthony Celi, and Roger G Mark. 2016. Mimic-iii, a freely accessible critical care database. *Scientific data*, 3(1):1–9.

Yoon Kim, Carl Denton, Luong Hoang, and Alexander M. Rush. 2017. Structured attention networks. In *5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Conference Track Proceedings*. OpenReview.net.

Diederik P. Kingma and Jimmy Ba. 2015. Adam: A method for stochastic optimization. In *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*.

Sebastian Lapuschkin, Stephan Wäldchen, Alexander Binder, Grégoire Montavon, Wojciech Samek, and Klaus-Robert Müller. 2019. Unmasking clever hans predictors and assessing what machines really learn. *Nature communications*, 10(1):1–8.

Zachary Chase Lipton, David C. Kale, Charles Elkan, and Randall C. Wetzel. 2016. Learning to diagnose with LSTM recurrent neural networks. In *4th International Conference on Learning Representations, ICLR 2016, San Juan, Puerto Rico, May 2-4, 2016, Conference Track Proceedings*.

Jingshu Liu, Zachariah Zhang, and Narges Razavian. 2018. Deep EHR: chronic disease prediction using medical notes. In *Proceedings of the Machine Learning for Healthcare Conference, MLHC 2018, 17-18 August 2018, Palo Alto, California*, volume 85 of *Proceedings of Machine Learning Research*, pages 440–464. PMLR.

Weijie Liu, Peng Zhou, Zhe Zhao, Zhiruo Wang, Qi Ju, Haotang Deng, and Ping Wang. 2020. K-BERT: enabling language representation with knowledge graph. In *The Thirty-Fourth AAAI Conference on Artificial Intelligence, AAAI 2020, The Thirty-Second Innovative Applications of Artificial Intelligence Conference, IAAI 2020, The Tenth AAAI Symposium on Educational Advances in Artificial Intelligence, EAAI 2020, New York, NY, USA, February 7-12, 2020*, pages 2901–2908. AAAI Press.

Fenglong Ma, Radha Chitta, Jing Zhou, Quanzeng You, Tong Sun, and Jing Gao. 2017. Dipole: Diagnosis prediction in healthcare via attention-based bidirectional recurrent neural networks. In *Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, Halifax, NS, Canada, August 13 - 17, 2017*, pages 1903–1911. ACM.

Fenglong Ma, Quanzeng You, Houping Xiao, Radha Chitta, Jing Zhou, and Jing Gao. 2018. KAME: knowledge-based attention model for diagnosis prediction in healthcare. In *Proceedings of the 27th ACM International Conference on Information and Knowledge Management, CIKM 2018, Torino, Italy, October 22-26, 2018*, pages 743–752. ACM.

Riccardo Miotto, Li Li, Brian A Kidd, and Joel T Dudley. 2016. Deep patient: an unsupervised representation to predict the future of patients from the electronic health records. *Scientific reports*, 6(1):1–10.

Jens-Michalis Papaioannou, Paul Grundmann, Betty van Aken, Athanasios Samaras, Ilias Kyparissidis, George Giannakoulas, Felix Gers, and Alexander Loeser. 2022. Cross-lingual knowledge transfer for clinical phenotyping. In *Proceedings of the Language Resources and Evaluation Conference*, pages 900–909, Marseille, France. European Language Resources Association.

Xueping Peng, Guodong Long, Tao Shen, Sen Wang, and Jing Jiang. 2020. Self-attention enhanced patient journey understanding in healthcare system. In *Machine Learning and Knowledge Discovery in Databases - European Conference, ECML PKDD 2020, Ghent, Belgium, September 14-18, 2020, Proceedings, Part III*, volume 12459 of *Lecture Notes in Computer Science*, pages 719–735. Springer.

Zhi Qiao, Xian Wu, Shen Ge, and Wei Fan. 2019. MNN: multimodal attentional neural networks for diagnosis prediction. In *Proceedings of the Twenty-Eighth International Joint Conference on Artificial Intelligence, IJCAI 2019, Macao, China, August 10-16, 2019*, pages 5937–5943. ijcai.org.

Madhumita Sushil, Simon Suster, Kim Luyckx, and Walter Daelemans. 2018a. Patient representation learning and interpretable evaluation using clinical notes. *J. Biomed. Informatics*, 84:103–113.

Madhumita Sushil, Simon Šuster, Kim Luyckx, and Walter Daelemans. 2018b. Patient representation learning and interpretable evaluation using clinical notes. *Journal of biomedical informatics*, 84:103–113.

Eric J. Topol. 2019. High-performance medicine: the convergence of human and artificial intelligence. *Nature Medicine*, 25:44–56.

Betty van Aken, Jens-Michalis Papaioannou, Manuel Mayrdorfer, Klemens Budde, Felix A. Gers, and Alexander Löser. 2021. Clinical outcome prediction from admission notes using self-supervised knowledge integration. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume, EACL 2021, Online, April 19 - 23, 2021*, pages 881–893. Association for Computational Linguistics.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, December 4-9, 2017, Long Beach, CA, USA*, pages 5998–6008.

Benjamin Winter, Alexei Figueroa Rosero, Alexander LÃ¶ser, Felix Alexander Gers, and Amy Siu. 2022. Kimera: Injecting domain knowledge into vacant transformer heads. In *Proceedings of the Language Resources and Evaluation Conference*, pages 363–373, Marseille, France. European Language Resources Association.

Bo Yang and Lijun Wu. 2021. How to leverage the multimodal EHR data for better medical prediction? In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing, EMNLP 2021, Virtual Event / Punta Cana, Dominican Republic, 7-11 November, 2021*, pages 4029–4038. Association for Computational Linguistics.

PARSE: An Efficient Search Method for Black-box Adversarial Text Attacks

Pengwei Zhan[§‡], **Chao Zheng**[§], **Jing Yang**[§*], **Yuxiang Wang**[§],
Liming Wang[§], **Yang Wu**[§‡], **Yunjian Zhang**[§‡]
[§]Institute of Information Engineering, Chinese Academy of Sciences, Beijing, China
[‡]School of Cyber Security, University of Chinese Academy of Sciences, Beijing, China
{zhanpengwei,zhengchao1135,yangjing,wangyuxiang}@iie.ac.cn
{wangliming,wuyang0419,zhangyunjian}@iie.ac.cn

Abstract

Neural networks are vulnerable to adversarial examples. The adversary can successfully attack a model even without knowing model architecture and parameters, i.e., under a black-box scenario. Previous works on word-level attacks widely use word importance ranking (WIR) methods and complex search methods, including greedy search and heuristic algorithms, to find optimal substitutions. However, these methods fail to balance the attack success rate and the cost of attacks, such as the number of queries to the model and the time consumption. In this paper, We propose **PA**thological wo**R**d **S**aliency s**E**arch (PARSE) that performs the search under dynamic search space following the subarea importance. Experiments show that PARSE can achieve comparable attack success rates to complex search methods while saving numerous queries and time, e.g., saving at most 74% of queries and 90% of time compared with greedy search when attacking the examples from Yelp dataset. The adversarial examples crafted by PARSE are also of high quality, highly transferable, and can effectively improve model robustness in adversarial training.

1 Introduction

Neural networks have achieved remarkable success in various NLP tasks while being vulnerable to adversarial examples. The adversary can craft adversarial examples, which contain noise that is imperceptible to human but can mislead the model decision, even without knowing the model architecture and parameters. Under such black-box scenario, word-level attacks have been more focused on by recent studies for the flexibility of the attack and the high quality generated examples (Gao et al., 2018; Alzantot et al., 2018; Jin et al., 2020; Li et al., 2018; Garg and Ramakrishnan, 2020a; Ebrahimi et al., 2018). Word-level attacks can flexibly fit the grammar and semantics constraints by changing the similarity and semantics threshold when filtering candidate substitutions, and the generated adversarial examples will not be detected by a spell checker (Ebrahimi et al., 2018; Iyyer et al., 2018) or substantially damage the overall semantic and logic of the sentence (Jia and Liang, 2017; Liang et al., 2018). High-quality adversarial examples ensure the attacks are imperceptible to human and can be used to learn the robustness of models better.

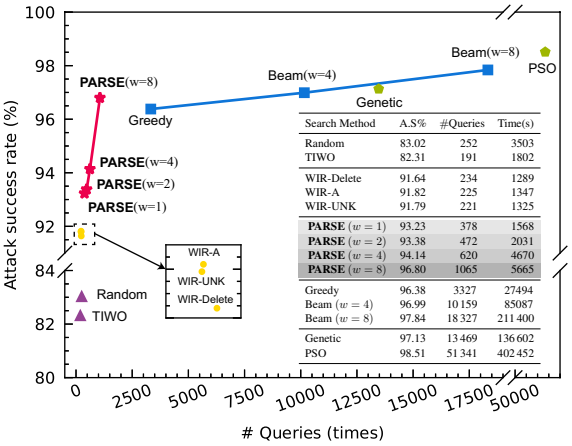

Search Method	A.S%	#Queries	Time(s)
Random	83.02	252	3503
TIWO	82.31	191	1802
WIR-Delete	91.64	234	1289
WIR-A	91.82	225	1347
WIR-UNK	91.79	221	1325
PARSE ($w=1$)	93.23	378	1568
PARSE ($w=2$)	93.38	472	2031
PARSE ($w=4$)	94.14	620	4670
PARSE ($w=8$)	96.80	1065	5665
Greedy	96.38	3327	27494
Beam ($w=4$)	96.99	10 159	85087
Beam ($w=8$)	97.84	18 327	211 400
Genetic	97.13	13 469	136 602
PSO	98.51	51 341	402 452

Figure 1: Average #Queries vs. Attack success rate (%) when attacking TextCNN on 500 examples from Yelp in HowNet. Increasing the beam width w in PARSE effectively increases the attack success rate while just costing a few more #queries and time. PARSE (w=8) outperforms Greedy search while taking only 32% of queries and 20% of time. The complete results are in Table 2.

To better explain our contribution to the word-level adversarial attack, we would like first to define the word-level attack as a combinatorial optimization problem, which is similar to (Yoo et al., 2020; Morris et al., 2020b,a). Under this setting, a word-level adversarial attack method can be decomposed into *Search Space* and *Search Method*. The search space gives all the possible substitutions that meet the similarity and semantics requirements for the target words, i.e., decides what words the target words can be transformed into. The search method

*Corresponding Author.

Proceedings of the 29th International Conference on Computational Linguistics, pages 4776–4787
October 12–17, 2022.

is the search strategy to perform the attack, i.e., decides which words to be transformed (target words) and what words should be transformed into (pick from the search space). Search method is the most significant part of an attack method, as the exponential nature of the search space makes inefficiency search method difficult to attack large-scale and long examples. Therefore, we fix the search space and only focus on the search methods in this paper.

Various search methods for word-level black-box attacks have been proposed, divided into simple methods, including the variants of Word Importance Ranking (WIR) methods (Gao et al., 2018; Li et al., 2018; Jin et al., 2020; Li et al., 2020), and complex methods, including greedy search (Pruthi et al., 2019; Li et al., 2021), beam search (Ebrahimi et al., 2018), genetic algorithm (Alzantot et al., 2018), and Particle Swarm Optimization (PSO) algorithm (Zang et al., 2020). WIR methods search substitutions for each word in the descending order of word importance scores, which is faster than other complex search methods, while is poor in attack success rate. Other search methods can always achieve better attack success rates, while they need more queries to the target model and time, as they directly search on the search space of the entire sentence. Previous works fail to balance well between performance and efficiency.

In this paper, to achieve higher attack success rates with fewer queries and time in word-level black-box attacks, we propose a search method called **PA**thological wo**R**d **S**aliency s**E**arch (PARSE). PARSE separates the entire sentence into multiple subarea according to the stability of words and searches in the descending order of the subarea importance. Therefore, PARSE avoids directly searching on the huge search space of the entire sentence and reduces the cost of attacks. Searching on each subarea instead of on each word like WIR methods also makes PARSE less likely to be stuck in local optima. Extensive experiments demonstrate that PARSE achieves comparable attack success rates to complex search methods while saving numerous queries and time, e.g., saving at most 74% of queries and 90% of time compared with greedy search when attacking the Yelp dataset (Zhang et al., 2015). Figure 1 shows the performance comparisons. The major contributions of this paper are summarized as follows:

- We define the stability of words in adversarial attacks and explain the ineffective of WIR methods from the view of word stability.
- We propose PARSE, a search method for word-level black-box adversarial attacks that performs search under dynamic search space following the subarea importance.
- Experiments show that PARSE achieves comparable attack success rates to complex methods while saving numerous queries and time.

2 Related Works

Adversarial attack. Inspired by the early works on adversarial attacks that mainly focus on the field of computer vision (CV) (Goodfellow et al., 2015; Papernot et al., 2016; Moosavi-Dezfooli et al., 2016; Carlini and Wagner, 2017), various methods to attack language models are proposed (Li et al., 2018; Gao et al., 2018; Garg and Ramakrishnan, 2020b; Miyato et al., 2017; Gong et al., 2018). Unlike the image, which is differentiable as pixels are continuous values, the discrete text is not differentiable. Therefore, the adversarial attacks in NLP tasks are more appropriately described as combinatorial optimization problems, which seek to find optimal substitutions in the search space (Yoo et al., 2020; Morris et al., 2020b,a).

Search Method. Although various adversarial attack frameworks focusing on the NLP tasks are proposed, few works make a clear distinction between the search space and search method. The reported results may benefit from strong search method (Alzantot et al., 2018; Zang et al., 2020; Jia et al., 2019), which have a higher time complexity and need more queries to the model, or the less restrictive search space (Pruthi et al., 2019; Gao et al., 2018; Ebrahimi et al., 2018; Li et al., 2018; Jin et al., 2020; Li et al., 2020), which does not consider both the distance between the target words and the substitutions and the semantic of the entire perturbed sentence. In this paper, we only focus on the search methods and benchmark all search methods under the same search space.

3 PARSE

3.1 Textual Adversarial Example

Suppose there is a model $\mathcal{F} : \mathcal{X} \rightarrow \mathcal{Y}$ trained by minimizing the empirical risk over all given text $\boldsymbol{X} \in \mathcal{X}$ and labels $Y \in \mathcal{Y}$ following the distribution $\mathcal{D}$:

$$\min_{\boldsymbol{\theta}} \mathbb{E}_{(\boldsymbol{X},Y)\sim\mathcal{D}} \mathcal{L}\left(\mathcal{F}\left(\boldsymbol{X};\boldsymbol{\theta}\right), Y\right) \quad (1)$$

where $\boldsymbol{\theta}$ is the parameter, and $\mathcal{L}(\cdot)$ is the cross-entropy loss. An adversarial example $\boldsymbol{X}^{adv}$ crafted from a normal text $\boldsymbol{X} = (x_n)_{n\in\{1,\dots,N\}}$ can thus be defined as:

$$\begin{aligned} &\boldsymbol{X}^{adv} = \mathcal{O}(\boldsymbol{X}) = o(x_n)_{n\in\{1,\dots,N\}}, \\ &\text{s.t.} \quad \forall n \in \{1,\dots,N\}, \ \ \Delta x_n < \delta, \\ &\text{and} \quad \Delta \boldsymbol{X} < \varepsilon, \\ &\text{and} \quad \underset{Y\in\mathcal{Y}}{\operatorname{argmax}}\ \mathcal{P}(Y|\boldsymbol{X}^{adv}) \neq \underset{Y\in\mathcal{Y}}{\operatorname{argmax}}\ \mathcal{P}(Y|\boldsymbol{X}) \end{aligned} \tag{2}$$

where $\mathcal{O}(\boldsymbol{X})$ means performing word-level substitution on sentence $\boldsymbol{X}$, $o(x_n)$ means substituting the word x_n with a new word from search space, if possible. Δx_n denotes the difference between x_n and $o(x_n)$, $\Delta \boldsymbol{X}$ denotes the difference between $\boldsymbol{X}$ and $\mathcal{O}(\boldsymbol{X})$, δ and ε are the maximum allowed difference of words and the overall sentence, respectively, which restrictions are imposed on search space to filter potential substitutions, $\mathcal{P}(\cdot|\cdot)$ is the posterior probability. Intuitively, (2) can be explained as the following condition. We have a finite search space that contains all possible substitutions for each word in $\boldsymbol{X}$, and the substitutions in the search space are further filtered by the restrictions, including δ and ε, which may mainly focus on the semantics and the L_p norm of embedding distance of each word and the entire sentence. These restrictions ensure that the final generated adversarial example is imperceptible to human. The search method can thus be seen as the strategy to perform $\mathcal{O}(\cdot)$, deciding the order to perform $o(\cdot)$ and the substitutions picked from the search space. When the restrictions on search space are fixed, the better search method finds the adversarial example more accurately and efficiently.

3.2 Word Importance

As gradient information is not available in the Black-box scenario, the Leave One Out (LOO) methods are proposed to obtain word importance, i.e., the word saliency. (Li et al., 2016; Gao et al., 2018; Li et al., 2018; Jin et al., 2020; Li et al., 2020). LOO methods expect to obtain word importance by comparing the model confidence of two sentences with only one word is different. Formally, the importance of words $x_i \in \boldsymbol{X}$ is defined as

$$S(x_i) = \mathcal{P}(Y_{true}|\boldsymbol{X}) - \mathcal{P}(Y_{true}|\hat{\boldsymbol{X}}_i) \tag{3}$$

where $S(\cdot)$ is the word saliency, Y_{true} is the ground-truth class, and $\hat{\boldsymbol{X}}_i = x_1 \dots \hat{x}_i \dots x_N$ is the sentence with word x_i transformed. Transforming x_i to $\hat{x}_i$ in different ways formulating three prevalent LOO methods: (i) Delete: leaving $\hat{x}_i$ blank. (ii) UNK: $\hat{x}_i = \texttt{[UNK]}$, triggering the out of vocabulary (OOV) problem. (iii) A: $\hat{x}_i = \texttt{a}$, replacing with a neutral word a that has a similar distribution across classes (Pruthi et al., 2019). Intuitively, performing the substituting operation $o(\cdot)$ on the words in the descending order of their importance should help to generate adversarial examples more efficiently (this is how the WIR search methods do), as the important word have a large impact on the model prediction.

3.3 Word Stability from the View of Words Importance Changing

To explain why the word importance fails to indicate the model concentration and why the WIR methods, which strictly follow the descending order of words importance to perform the attack, have a degenerated performance, we first try to answer:

When should a word be considered unstable?

We would like to clarify that the stability of a word is defined together with the system trying to understand the word, i.e., the stable word for a system may be an unstable word for other systems. For human, parsing a sentence is a denoising process. We can still understand a sentence even if slight noise is introduced to the sentence, e.g., changing the order of letters in a word or deleting some words in a sentence (McCusker et al., 1981; Rayner et al., 2006; Adam Drewnowski, 1978; McCusker et al., 1981; Van Orden, 1987). More importantly, we focus on the important words in the sentence and do not change our attention to the words due to the small noise. Based on this, there are few unstable words for the human reading comprehension system, as the important words we concentrate on do not change. Following this, if a system, e.g., a language model, changes attention to the words when parsing a sentence because of sufficiently slight noise, we consider the words whose importance, i.e., the attention of the system, have changed as unstable words for the system. It should be noted that, as defined in (3), the word importance is a continuous value, and if the importance of all words increases to the same extent, the attention of the system is actually not changed. Therefore, we further use a discrete value called importance ranking to define the attention, which is formed as

$$\begin{aligned} R(\boldsymbol{X}) &= r(x_i)_{i\in\{1,\dots,N\}} \\ &= \arg\underset{i}{\operatorname{sort}}\, S(x_i)_{i\in\{1,\dots,N\}} \end{aligned} \tag{4}$$

where $\arg\mathrm{sort}(\cdot)$ returns the indexes of the sorted sequence in descending order. Therefore, the stability of a word and a group of adjacent words can be defined.

Definition 1. *For a given model $\mathcal{F}$, a sentence $\boldsymbol{X} = (x_n)_{n\in\{1,\ldots,N\}}$, let $t(\cdot)$ be a slight transformation that deletes the least important word in a sentence, the word importance rankings of $\boldsymbol{X}$ and $t(\boldsymbol{X})$ are $R(\boldsymbol{X}) = (r_1, r_2, \ldots, r_N)$ and $R(t(\boldsymbol{X})) = (r_1^*, r_2^*, \ldots, r_N^*)$, respectively, where r_i and r_i^* are the word index. If $r_i \neq r_i^*$, then the word x_{r_i} is an unstable word for model $\mathcal{F}$; otherwise, a stable word. If $\forall\, r_j \in (r_i, \ldots, r_{i+n}), r_j \neq r_j^*$, the adjacent words $(x_{r_i}, \ldots, x_{r_{i+n}})$ form an unstable subarea; otherwise, a stable subarea.*

Intuitively, the importance of unstable words can not accurately indicate the impact on the model prediction, as even a slight transformation is enough to shift the importance ranking of these words over the entire sentence, and the important word may become not that important. That is why the WIR methods always have poor performance. Such phenomena are probably due to the model pathology (Feng et al., 2018) as neural networks are more linear than expected and will overfit the negative log-likelihood loss to produce low-entropy distribution over classes, leading the model to overconfidence in instances outside the training data distribution (Goodfellow et al., 2015). This consequently leads to the word importance drastically changing with even the least important word being removed from the sentence, which is sufficient to bias the sentence representation from the distribution.

To show the influence of word stability for common language models of different architectures, we test the word stability of the sentence in MR and Yelp training set for LSTM, TextCNN, and DistilBERT (model architectures are detailed in §4.1). Following (3), (4), and Definition 1, we first obtain the word importance rankings with the LOO-UNK method on 500 randomly picked examples and then compare the word importance rankings between the original sentences and the sentences with the least important word removed to obtain the word stability. Table 1 shows the average stability results of five individual runs. There are relatively few unstable words on the short text on MR, with an average of 45.5% on the three models, while it will rise to 72.4% on the longer text on Yelp. We also find that the average length of an unstable subarea is very short compared with the length of the entire sentence and takes only 19.3% for MR and 7.8% for Yelp on average. Therefore, we can draw two conclusions about word stability:

Dataset	Model	#input words	#unstable words	unstable word%	#unstable subarea	AVG unstable subarea length
MR	LSTM	18.25	8.14	44.60	2.23	3.49
	TextCNN	18.72	8.26	44.12	2.32	3.56
	DistilBERT	18.39	8.79	47.79	2.41	3.65
Yelp	LSTM	128.39	96.66	75.28	10.81	8.94
	TextCNN	133.29	83.94	62.97	8.77	9.57
	DistilBERT	135.86	107.14	78.86	8.61	12.51

Table 1: Statistics on the word stability of the sentence in MR and Yelp training set for different models.

(C_1) unstable words are prevalent regardless of the model architecture and take a higher proportion in a longer sentence.

(C_2) the word importance rankings are mainly swapping between the words of similar importance (as the average length of unstable subarea is short).

3.4 Searching Strategy of PARSE

To generate adversarial examples in higher attack success rate with fewer queries and time, a search method must take the word stability (and further the (C_1) and (C_2)) into account. As the importance of unstable words fails to accurately indicate the impact on the model prediction, strictly following which to perform attack are likely to stuck in local optima. Based on this, we propose PARSE that performs beam search under dynamic search space following subarea importance. The general searching strategy of PARSE is shown in Figure 2. Specifically, PARSE starts by transforming the target sentence $\boldsymbol{X}$ with transformation $t(\cdot)$ and obtaining the stability of all words in the target sentences. PARSE treats each stable word individually while treating the adjacent unstable words, i.e., the words in the same unstable subarea, as an integration. That is, according to Definition 1, each stable word forms a stable subarea, and multiple adjacent unstable words form an unstable subarea. Therefore, the sentence is separated into multiple subarea based on the stability of words, and the entire potential search space is separated into multiple subspace. PARSE avoids being too sensitive to the importance of a single word and being affected by the inaccuracy of word importance by taking the subarea as basic elements at each search step rather than each word like other methods. The search is then performed following the descending order of

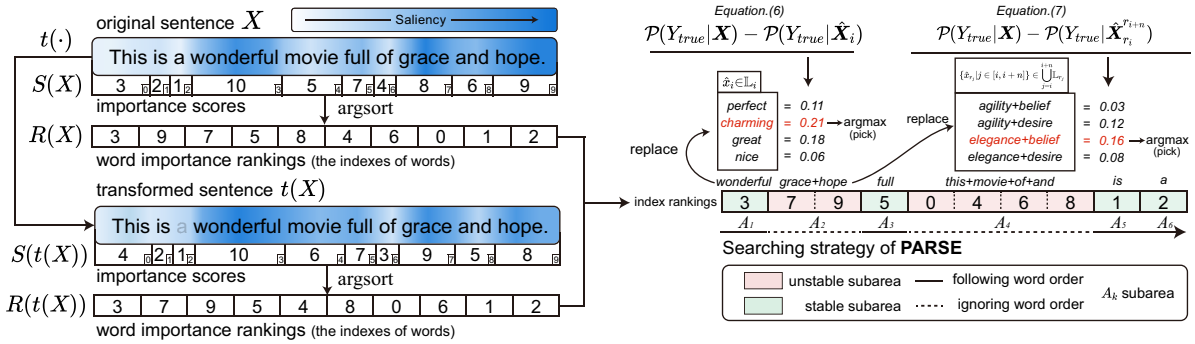

Figure 2: The general searching strategy of PARSE. We separate the entire sentence into multiple subarea according to the word stability. PARSE performs search in the descending order of subarea importance and takes each subarea as integration at each search step.

the subarea importance score, which is the average importance rankings of the words in a subarea:

$$score(\boldsymbol{A}) = \frac{1}{||\boldsymbol{A}||} \sum_{x_i \in \boldsymbol{A}} r(x_i) \tag{5}$$

where $\boldsymbol{A}$ denotes a subarea, $||\boldsymbol{A}||$ denotes the number of words in a subarea. PARSE behaves differently when encountering stable subarea and unstable subarea. To better understand the idea of PARSE, we first explain the detailed search process when beam width $w = 1$. When encountering stable subarea $\boldsymbol{A}_k$, which contains the only word $x_i \in \boldsymbol{A}_k$, we tend to find the substitution $\hat{x}_i$ that mostly reduces the model confidence from the search space:

$$\hat{x}_i = \arg\max_{\hat{x}_i \in \mathbb{L}_i} \{\mathcal{P}(Y_{true}|\boldsymbol{X}) - \mathcal{P}(Y_{true}|\hat{\boldsymbol{X}}_i)\} \tag{6}$$

where $\mathbb{L}_i$ is the potential search space for x_i under the fixed restrictions including δ and ε, and $\hat{\boldsymbol{X}}_i$ is the sentence with word x_i transformed into $\hat{x}_i$. When encountering unstable subarea $\boldsymbol{A}_k$ that contains multiple unstable words $(x_{r_i}, \ldots, x_{r_{i+n}}) \in \boldsymbol{A}_k$, we tend to find the group of substitutions $\{\hat{x}_{r_j} | j \in [i, i+n]\}$ that mostly reduces the model confidence, which equals to:

$$\underset{\{\hat{x}_{r_j} | j \in [i,i+n]\} \in \bigcup_{j=i}^{i+n} \mathbb{L}_{r_j}}{\text{argmax}} \{\mathcal{P}(Y_{true}|\boldsymbol{X}) - \mathcal{P}(Y_{true}|\hat{\boldsymbol{X}}_{r_i}^{r_{i+n}})\} \tag{7}$$

where $\bigcup_{j=i}^{i+n} \mathbb{L}_{r_j}$ is the potential search space for the multiple words $(x_{r_i}, \ldots, x_{r_{i+n}}) \in \boldsymbol{A}_k$, i.e., the combination of the search space of every single word, $\hat{\boldsymbol{X}}_{r_i}^{r_{i+n}}$ is the sentence with words $\{x_{r_j} | j \in [i, i+n]\}$ transformed into $\{\hat{x}_{r_j} | j \in [i, i+n]\}$. Intuitively, different from the search space of a single word $\mathbb{L}_i$, the same substitution for a single word in $\{\hat{x}_{r_j} | j \in [i, i+n]\}$ may appear many times in all possible substitution combinations in the search space of multiple words $\bigcup_{j=i}^{i+n} \mathbb{L}_{r_j}$. Thus the word order in an unstable subarea is ignored, and the words in an unstable subarea will be substituted at the same time at each search step, which helps reduce the impact of the inaccurate importance rankings of unstable words. The search is performed on every subarea in order until the generated example meets (2). When the beam width $w \neq 1$, we keep the top-w $\hat{x}_i$ or $\{\hat{x}_{r_j} | j \in [i, i+n]\}$ that the results most close to equation (6) or (7), i.e., mostly reduce the model confidence, at each search step. The details of PARSE are shown in Algorithm 1.

Algorithm 1: PARSE

input : Original sentence $\boldsymbol{X} = x_1 x_2 \ldots x_N$,
Separated search space
$\mathcal{A} = (\boldsymbol{A}_1, \boldsymbol{A}_2, \ldots, \boldsymbol{A}_n)$,
beam width w, true label Y_{true}
output : Adversarial example $\boldsymbol{X}^{adv}$

Initialize candidate set $\mathcal{X}_{best} \leftarrow \{\boldsymbol{X}\}$
Sort $\mathcal{A}$ by the subarea score (Eq.(5)) in descending
for *all* $\boldsymbol{A}_k \in \mathcal{A}$ **do**
  reset the union of candidate set $\mathcal{X}_{all} \leftarrow \{\}$
  for *all* $\boldsymbol{X}'_j \in \mathcal{X}_{best}$ **do**
    if $\boldsymbol{A}_k$ *is stable zone* **then**
      $\mathcal{X}_{cand} \leftarrow$ {top-w sentences transformed from $\boldsymbol{X}'_j$ that $\hat{x}_i$ most close to Eq.(6)}
    else
      $\mathcal{X}_{cand} \leftarrow$ {top-w sentences transformed from $\boldsymbol{X}'_j$ that $\{\hat{x}_{r_j} | j \in [i, i+n]\}$ most close to Eq.(7)}
    $\mathcal{X}_{all} \leftarrow \mathcal{X}_{all} \cup \mathcal{X}_{cand}$
  $\mathcal{X}_{best} \leftarrow$ {top-w sentences $\boldsymbol{X}' \in \mathcal{X}_{all}$ that mostly reduce model confidence $\mathcal{P}(Y_{true}|\boldsymbol{X}) - \mathcal{P}(Y_{true}|\boldsymbol{X}')$}
  if $\exists \underset{Y \in \mathcal{Y}, \boldsymbol{X}' \in \mathcal{X}_{best}}{\text{argmax}} \mathcal{P}(Y|\boldsymbol{X}') \neq Y_{true}$ **then**
    return *the* $\boldsymbol{X}'$ *that mostly reduces model confidence as* $\boldsymbol{X}^{adv}$; `/* Success */`
return $\boldsymbol{X}$; `/* Fail */`

4 Experiment

4.1 Experiment Setup

Dataset. The experiments are conducted on Movie Review (MR) (Pang and Lee, 2005) and Yelp Review Polarity (Yelp) (Zhang et al., 2015). Both of them are sentiment classification tasks. For MR, the average text length is 18.49. For Yelp, the average text length is 135.66.

Model. We use TextCNN (Kim, 2014), LSTM, and DistilBERT (Sanh et al., 2019) in our experiments. More details of the model are in Appendix.

Search Space. We utilize the 300-Dimensional GloVe word vectors (Pennington et al., 2014) and HowNet (Dong and Dong, 2003) as the search space for substitutions. GloVe contains vector representations for words learned by an unsupervised learning algorithm. HowNet is a knowledge base of sememes with over 100,000 words.

Restrictions on Search Space. The possible substitutions for each word in the search space are filtered by the restrictions imposed on the search space. The substitutions picked from the search space should have the same part of speech as the original word. The similarity between the generated and original sentences measured by BERT should be larger than 90%.

Baselines. We compare PARSE with nine search methods: Random, Traverse in word order (TIWO), WIR-Delete, WIR-A, WIR-UNK, Greedy Search, Beam Search, Genetic Algorithm (Genetic) (Alzantot et al., 2018), and Particle Swarm Optimization (PSO) (Zang et al., 2020). The Random method randomly picks a word as the target word at each search step. TIWO method performs the search following the word order.

Implementation Details. For PARSE, we use the LOO-UNK to obtain the word importance. For the Genetic and PSO algorithm, the population size and the number of iterations are set to 60 and 20, respectively. All reported results are the average of five individual runs. All comparisons in our experiments are conducted under the same search space with the same restrictions on the same machine with an A5000 GPU.

4.2 Main Results

Comparisons on Performance. We perform adversarial attacks on 500 randomly picked examples, and the results on performance are shown in Table 2. Even when $w = 1$, PARSE still outperforms WIR methods on attack success rates, and #queries and time are only slightly increased. This indicates that considering the stability of words and treating the words of different stability differently in each search step helps reduce the impact of the inaccurate word importance. Compared with the complex search methods like greedy search, PARSE ($w = 8$) can achieve comparable attack success rates with fewer #queries and time, especially on the long text from Yelp. On MR, on average, PARSE ($w = 8$) needs 213 queries and 2.9 seconds for each successful attack, while greedy search needs 390 queries and 7.1 seconds. On Yelp, on average, PARSE ($w = 8$) needs 1320 queries and 13 seconds for each successful attack, while greedy search needs 4582 queries and 115 seconds. Others complex search methods even need far more queries and time. It should be noted that PARSE is more suitable for attacking long sentences as it is less affected by the increased search space compared to other complex search methods, while it can still achieve competitive results when attacking short sentences.

Comparisons under Different Search Space. We replace the search methods while maintaining the search space in TextBugger, BAE, and DeepWordBug, then perform attacks on 500 randomly picked examples from MR on three models. Table 3 shows the comparisons of different search methods. Genetic and PSO are excluded for their low efficiency (especially on DistilBERT). PARSE ($w = 8$) always achieves comparable attack success rates to complex methods while generally needing fewer queries and time, indicating that PARSE can be effectively applied to different search space.

Quality of Crafted Example. We measure the quality of the adversarial examples crafted by different search methods by attacking LSTM on MR in HowNet. We use the LanguageTool[1] to detect the grammar correctness and use the Universal Sentence Encoder (USE) (Cer et al., 2018) to measure the semantic similarity of 500 randomly picked successfully attacked examples and the original examples. We also conduct human evaluations on Amazon Mechanical Turk[2] by asking the workers

[1] https://languagetool.org/
[2] https://www.mturk.com/

		Yelp						MR					
		GloVe			HowNet			GloVe			HowNet		
Model	Search Method	A.S%	#Queries	Time (s)	A.S%	#Queries	Time (s)	A.S%	#Queries	Time (s)	A.S%	#Queries	Time (s)
LSTM	Random	87.45	305	5196	88.81	209	2955	66.58	64	856	60.54	38	483
	TIWO	86.64	237	4118	86.13	159	2221	72.73	61	877	65.14	38	266
	WIR-Delete	94.38	283	1803	94.19	233	986	80.84	69	684	71.38	51	286
	WIR-A	94.31	254	1721	94.88	216	969	81.13	69	673	71.35	52	284
	WIR-UNK	94.82	268	1787	94.87	221	971	81.45	68	671	72.47	52	257
	PARSE($w=1$)	96.47	480	2955	95.79	390	1525	83.38	73	680	74.23	73	302
	PARSE($w=2$)	97.31	568	4351	97.94	496	2650	84.87	98	683	77.76	81	397
	PARSE($w=4$)	97.81	899	4614	98.12	782	3774	87.18	149	907	78.68	113	542
	PARSE($w=8$)	98.64	1839	7736	98.37	1189	4129	88.11	223	1341	80.38	159	688
	Greedy	98.92	5786	85 763	98.33	2953	28 029	88.85	489	5288	81.72	253	1427
	Beam($w=4$)	99.37	14 271	177 167	98.54	6516	62 391	90.18	735	7539	82.17	368	2647
	Beam($w=8$)	99.54	25 311	458 740	98.97	11 979	100 524	90.93	1088	9399	83.71	566	5473
	Genetic	99.28	10 197	142 847	98.73	7991	97 482	93.74	3144	23 660	88.41	2579	11 232
	PSO	$\approx$ 224 / 1 week			99.17	38 749	317 175	89.90	3101	14 780	85.27	2458	6272
TextCNN	Random	79.12	399	8061	83.02	252	3503	62.98	65	1250	57.04	46	349
	TIWO	79.23	303	5160	82.31	191	1802	56.28	68	1149	54.19	48	530
	WIR-Delete	93.89	286	1957	91.64	234	1289	79.48	75	936	68.54	55	400
	WIR-A	94.68	267	1871	91.82	225	1347	78.32	73	925	69.71	53	424
	WIR-UNK	94.05	269	1845	91.79	221	1325	79.93	79	941	69.25	53	401
	PARSE($w=1$)	95.73	482	3055	93.23	378	1568	81.14	103	991	70.78	81	574
	PARSE($w=2$)	96.56	595	4482	93.38	472	2031	85.12	118	1036	73.02	95	724
	PARSE($w=4$)	96.95	838	5285	94.14	620	4670	86.96	185	1321	75.29	137	1119
	PARSE($w=8$)	97.26	1629	8619	96.80	1065	5665	87.93	278	2579	76.31	193	1342
	Greedy	96.54	6264	89 001	96.38	3327	27 494	88.61	532	5591	77.67	287	1935
	Beam($w=4$)	97.03	18 165	258 647	96.99	10 159	85 087	91.22	903	11 022	80.59	452	2730
	Beam($w=8$)	$\approx$ 287 / 1 week			97.84	18 327	211 399	93.47	1581	14 857	81.34	807	5918
	Genetic	96.98	13 678	168 335	97.13	13 469	136 602	95.34	3387	33 867	88.37	2918	14 455
	PSO	$\approx$ 168 / 1 week			98.51	51 341	402 452	92.25	4397	37 755	83.85	3081	7452

Table 2: The comparisons on attack success rate (A.S%), average #queries to attack one example (#Queries), and total seconds to attack 500 examples (Time) of different search methods. $\approx$ *n / 1 week* means the attack fails to complete in 1 week, and n is the number of the completed attacks.

	LSTM			TextCNN			DistilBERT		
	A.S%	#Que.	Time	A.S%	#Que.	Time	A.S%	#Que.	Time
TextBugger (WIR-Delete)	78.44	48	29	78.86	48	28	69.84	51	107
w/ PARSE($w=1$)	79.73	65	33	81.34	65	29	75.24	66	112
w/ PARSE($w=2$)	83.81	79	39	84.99	78	38	78.18	95	157
w/ PARSE($w=4$)	84.65	97	56	86.88	113	53	80.74	132	213
w/ PARSE($w=8$)	87.53	138	78	87.23	168	81	81.05	218	335
w/ Greedy	86.73	227	89	86.35	243	91	83.74	312	436
w/ Beam($w=4$)	89.69	509	167	91.42	554	180	85.51	616	901
w/ Beam($w=8$)	90.89	819	273	91.74	855	319	87.23	1185	1637
BAE (WIR-Delete)	72.13	56	627	67.57	58	799	63.80	58	830
w/ PARSE($w=1$)	73.15	72	821	71.89	73	903	65.95	77	1011
w/ PARSE($w=2$)	74.43	89	970	72.87	92	1157	68.56	95	1248
w/ PARSE($w=4$)	74.63	108	1739	74.16	107	1971	70.34	125	2072
w/ PARSE($w=8$)	76.06	162	2758	74.88	169	2992	71.21	198	3228
w/ Greedy	77.39	229	3221	75.06	224	3719	71.33	234	4041
w/ Beam($w=4$)	79.98	384	5976	76.64	389	6731	75.50	395	7072
w/ Beam($w=8$)	80.65	494	10 679	78.92	568	12 493	76.32	642	12 929
DeepWordBug (WIR-Delete)	83.21	30	18	86.91	33	11	77.32	36	86
w/ PARSE($w=1$)	84.31	50	20	88.38	51	12	78.71	51	98
w/ PARSE($w=2$)	84.57	62	23	88.98	57	16	80.18	63	115
w/ PARSE($w=4$)	87.21	80	26	91.06	81	21	81.88	94	162
w/ PARSE($w=8$)	89.30	96	31	93.56	98	26	83.76	115	187
w/ Greedy	89.12	115	32	93.87	124	31	85.93	142	203
w/ Beam($w=4$)	92.20	254	58	95.56	249	48	91.32	179	419
w/ Beam($w=8$)	93.22	431	87	96.07	416	79	92.31	518	771

Table 3: Performance of different search methods when searching on the search space of previous frameworks. The original attacks use WIR-Delete as search method.

Search method	#Increased grammar errors	USE similarity	Perturbed%	Plausibility
Normal	-	-	-	3.15
Random	0.094	0.881	11.69	3.68
TIWO	0.063	0.893	10.72	3.51
WIR-Delete	0.101	0.894	10.05	3.46
WIR-A	0.104	0.896	10.01	3.45
WIR-UNK	0.097	0.897	9.95	3.45
PARSE($w=1$)	0.093	0.897	9.52	3.43
PARSE($w=2$)	0.091	0.897	9.59	3.42
PARSE($w=4$)	0.083	0.895	9.69	3.42
PARSE($w=8$)	0.077	0.895	9.76	3.39
Greedy	0.064	0.905	8.79	3.35
Beam($w=4$)	0.079	0.907	8.68	3.36
Beam($w=8$)	0.082	0.909	8.53	3.36
Genetic	0.074	0.880	11.21	3.37
PSO	0.089	0.896	10.18	3.35

Table 4: Quality of the adversarial examples crafted by different search methods.

to give scores from 1 (best) to 5 (worse) to indicate the *Plausibility* of 100 adversarial examples and 100 randomly picked normal examples. Table 4 shows the results on the quality of the generated adversarial examples. PARSE effectively reduces the increased grammar errors by increasing the search width w, while a larger w will increase the grammar errors of the example crafted by beam search. The effect of PARSE on maintaining the USE similarity of sentences is similar to WIR and PSO methods and is just relatively 1.32% worse than greedy search and beam search. PARSE perturbs fewer words than WIR methods and only needs an average of 1.02% more perturbed words than greedy

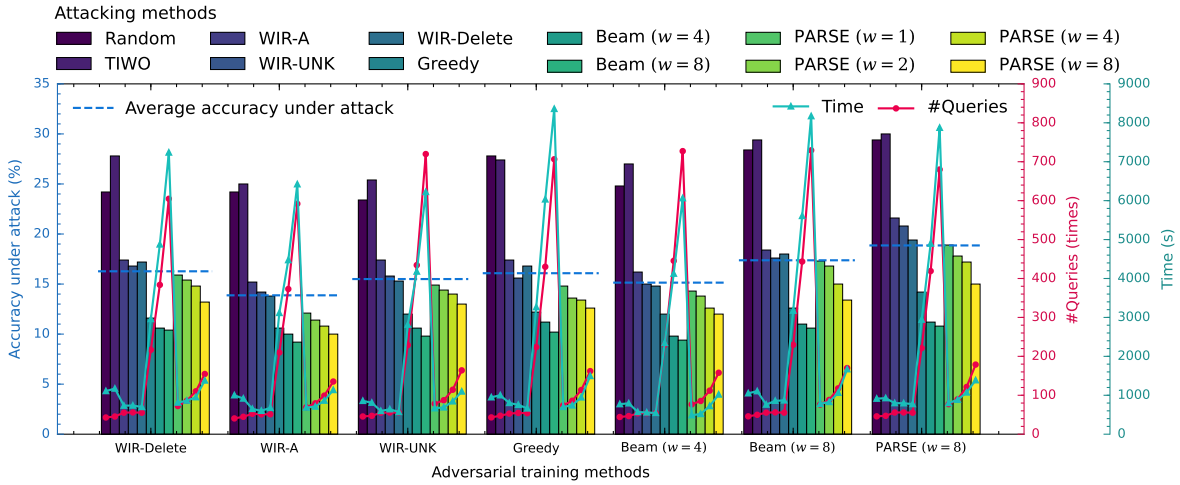

Figure 3: The comparisons of the #queries and time needed by different search methods to attack the adversarially trained models. Line plot correspond to the axis of the same color. Bar plot indicates the accuracy under attack.

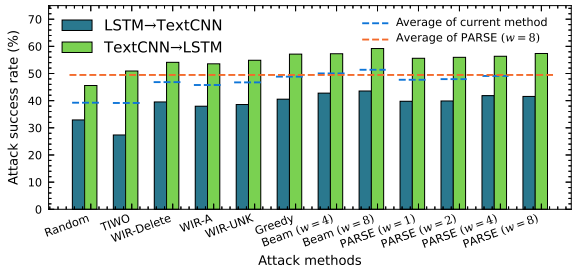

Figure 4: The comparison of the transferability of adversarial examples crafted by different search methods.

search and beam search. The results on plausibility show that PARSE with larger w generates more human-understandable adversarial examples. The case study is shown in Table 6-12 in Appendix.

Adversarial Training and Model Robustness. We randomly generate 1000 adversarial examples by attacking LSTM on MR in HowNet with different search methods, and then adversarially retrains the LSTM with the generated adversarial examples. Figure 3 shows the results on model robustness and attack performance. The model trained with the adversarial examples crafted by PARSE ($w = 8$) has the highest average accuracy, indicating that PARSE ($w = 8$) outperforms other methods in improving model robustness. PARSE is effective and efficient even when attacking robust models.

Transferability. We randomly generate 1000 adversarial examples with different search methods on MR in HowNet. Figure 4 shows the result of transferability between LSTM and TextCNN. Increasing the beam width w in PARSE helps generate adversarial examples with higher transferability. When $w = 8$, the transferability of adversarial examples crafted by PARSE outperforms all baselines except beam search.

	A.S%	#Increased grammar errors	USE similarity	Perturbed%
PARSE ($w = 1$)	74.23	0.093	0.897	9.52
w/o Word Stability	72.47	0.097	0.897	9.95
PARSE ($w = 2$)	77.76	0.091	0.897	9.59
w/o Word Stability	74.45	0.095	0.894	10.05
PARSE ($w = 4$)	78.68	0.083	0.895	9.69
w/o Word Stability	76.74	0.096	0.893	10.12
PARSE ($w = 8$)	80.38	0.077	0.895	9.76
w/o Word Stability	77.23	0.101	0.892	10.45

Table 5: Influence of parameter w and word stability. *w/o word stability* means the search method does not make a distinction between the words of different stability, and *PARSE* ($w = 1$) *w/o Word Stability* equals to *WIR-UNK* method.

Ablation Study. Table 5 shows the influence of word stability when attacking LSTM on 500 examples from MR in HowNet. We find that changing the search space according to the word stability increases the attack success rate and helps generate adversarial examples with fewer grammar errors, higher USE similarity, and fewer perturbed words.

5 Conclusion

This paper proposes PARSE, an efficient search method for black-box adversarial text attacks, which performs search under dynamic search space following the subarea importance. PARSE can achieve comparable attack success rates to complex search methods while saving numerous queries and time. The adversarial examples crafted by PARSE are high quality and highly transferable. We hope the analysis in our paper will inspire future work.

Acknowledgements

We would like to thank the anonymous reviewers for their insightful comments. This research was supported by National Research and Development Program of China (No.2019YFB1005200).

References

Alice F. Healy Adam Drewnowski. 1978. Detection errors on the word the: Evidence for the acquisition of reading levels. *Memory & Cognition*, 5:403–409.

Moustafa Alzantot, Yash Sharma, Ahmed Elgohary, Bo-Jhang Ho, Mani Srivastava, and Kai-Wei Chang. 2018. Generating natural language adversarial examples. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 2890–2896. Association for Computational Linguistics.

Nicholas Carlini and David Wagner. 2017. Towards evaluating the robustness of neural networks. In *2017 ieee symposium on security and privacy (sp)*, pages 39–57. IEEE.

Daniel Cer, Yinfei Yang, Sheng-yi Kong, Nan Hua, Nicole Limtiaco, Rhomni St. John, Noah Constant, Mario Guajardo-Cespedes, Steve Yuan, Chris Tar, Yun-Hsuan Sung, Brian Strope, and Ray Kurzweil. 2018. Universal sentence encoder. *CoRR*, abs/1803.11175.

Zhendong Dong and Qiang Dong. 2003. Hownet-a hybrid language and knowledge resource. In *International Conference on Natural Language Processing and Knowledge Engineering*, pages 820–824. IEEE.

Javid Ebrahimi, Anyi Rao, Daniel Lowd, and Dejing Dou. 2018. HotFlip: White-box adversarial examples for text classification. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 31–36. Association for Computational Linguistics.

Shi Feng, Eric Wallace, Alvin Grissom II, Mohit Iyyer, Pedro Rodriguez, and Jordan Boyd-Graber. 2018. Pathologies of neural models make interpretations difficult. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 3719–3728. Association for Computational Linguistics.

Ji Gao, Jack Lanchantin, Mary Lou Soffa, and Yanjun Qi. 2018. Black-box generation of adversarial text sequences to evade deep learning classifiers. In *2018 IEEE Security and Privacy Workshops (SPW)*, pages 50–56. IEEE.

Siddhant Garg and Goutham Ramakrishnan. 2020a. BAE: BERT-based adversarial examples for text classification. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 6174–6181. Association for Computational Linguistics.

Siddhant Garg and Goutham Ramakrishnan. 2020b. BAE: BERT-based adversarial examples for text classification. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 6174–6181. Association for Computational Linguistics.

Zhitao Gong, Wenlu Wang, Bo Li, Dawn Song, and Wei-Shinn Ku. 2018. Adversarial texts with gradient methods. *arXiv preprint arXiv:1801.07175*.

Ian J. Goodfellow, Jonathon Shlens, and Christian Szegedy. 2015. Explaining and harnessing adversarial examples. In *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*.

Mohit Iyyer, John Wieting, Kevin Gimpel, and Luke Zettlemoyer. 2018. Adversarial example generation with syntactically controlled paraphrase networks. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 1875–1885. Association for Computational Linguistics.

Robin Jia and Percy Liang. 2017. Adversarial examples for evaluating reading comprehension systems. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 2021–2031. Association for Computational Linguistics.

Robin Jia, Aditi Raghunathan, Kerem Göksel, and Percy Liang. 2019. Certified robustness to adversarial word substitutions. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 4129–4142. Association for Computational Linguistics.

Di Jin, Zhijing Jin, Joey Tianyi Zhou, and Peter Szolovits. 2020. Is BERT really robust? A strong baseline for natural language attack on text classification and entailment. In *The Thirty-Fourth AAAI Conference on Artificial Intelligence, AAAI 2020, The Thirty-Second Innovative Applications of Artificial Intelligence Conference, IAAI 2020, The Tenth AAAI Symposium on Educational Advances in Artificial Intelligence, EAAI 2020, New York, NY, USA, February 7-12, 2020*, pages 8018–8025. AAAI Press.

Yoon Kim. 2014. Convolutional neural networks for sentence classification. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1746–1751. Association for Computational Linguistics.

Dianqi Li, Yizhe Zhang, Hao Peng, Liqun Chen, Chris Brockett, Ming-Ting Sun, and Bill Dolan. 2021. Contextualized perturbation for textual adversarial attack. In *Proceedings of the 2021 Conference of*

the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, pages 5053–5069. Association for Computational Linguistics.

Jinfeng Li, Shouling Ji, Tianyu Du, Bo Li, and Ting Wang. 2018. Textbugger: Generating adversarial text against real-world applications. *arXiv preprint arXiv:1812.05271*.

Jiwei Li, Will Monroe, and Dan Jurafsky. 2016. Understanding neural networks through representation erasure. *arXiv preprint arXiv:1612.08220*.

Linyang Li, Ruotian Ma, Qipeng Guo, Xiangyang Xue, and Xipeng Qiu. 2020. BERT-ATTACK: Adversarial attack against BERT using BERT. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 6193–6202. Association for Computational Linguistics.

Bin Liang, Hongcheng Li, Miaoqiang Su, Pan Bian, Xirong Li, and Wenchang Shi. 2018. Deep text classification can be fooled. In *Proceedings of the Twenty-Seventh International Joint Conference on Artificial Intelligence, IJCAI 2018, July 13-19, 2018, Stockholm, Sweden*, pages 4208–4215. ijcai.org.

Leo X McCusker, Philip B Gough, and Randolph G Bias. 1981. Word recognition inside out and outside in. *Journal of Experimental Psychology: Human Perception and Performance*, 7(3):538.

Takeru Miyato, Andrew M. Dai, and Ian J. Goodfellow. 2017. Adversarial training methods for semi-supervised text classification. In *5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Conference Track Proceedings*. OpenReview.net.

Seyed-Mohsen Moosavi-Dezfooli, Alhussein Fawzi, and Pascal Frossard. 2016. Deepfool: A simple and accurate method to fool deep neural networks. In *2016 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2016, Las Vegas, NV, USA, June 27-30, 2016*, pages 2574–2582. IEEE Computer Society.

John Morris, Eli Lifland, Jack Lanchantin, Yangfeng Ji, and Yanjun Qi. 2020a. Reevaluating adversarial examples in natural language. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 3829–3839. Association for Computational Linguistics.

John Morris, Eli Lifland, Jin Yong Yoo, Jake Grigsby, Di Jin, and Yanjun Qi. 2020b. TextAttack: A framework for adversarial attacks, data augmentation, and adversarial training in NLP. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 119–126. Association for Computational Linguistics.

Bo Pang and Lillian Lee. 2005. Seeing stars: Exploiting class relationships for sentiment categorization with respect to rating scales. In *Proceedings of the 43rd Annual Meeting of the Association for Computational Linguistics (ACL'05)*, pages 115–124. Association for Computational Linguistics.

Nicolas Papernot, Patrick McDaniel, Somesh Jha, Matt Fredrikson, Z Berkay Celik, and Ananthram Swami. 2016. The limitations of deep learning in adversarial settings. In *2016 IEEE European symposium on security and privacy (EuroS&P)*, pages 372–387. IEEE.

Jeffrey Pennington, Richard Socher, and Christopher Manning. 2014. GloVe: Global vectors for word representation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1532–1543. Association for Computational Linguistics.

Danish Pruthi, Bhuwan Dhingra, and Zachary C. Lipton. 2019. Combating adversarial misspellings with robust word recognition. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 5582–5591. Association for Computational Linguistics.

Keith Rayner, Sarah J. White, Rebecca L. Johnson, and Simon P. Liversedge. 2006. Raeding wrods with jubmled lettres: There is a cost. *Psychological Science*, 17(3):192–193. PMID: 16507057.

Victor Sanh, Lysandre Debut, Julien Chaumond, and Thomas Wolf. 2019. Distilbert, a distilled version of bert: smaller, faster, cheaper and lighter. *arXiv preprint arXiv:1910.01108*.

G. C Van Orden. 1987. A ROWS is a ROSE: Spelling, sound, and reading. *Memory & Cognition*, 15(3):181–198.

Jin Yong Yoo, John Morris, Eli Lifland, and Yanjun Qi. 2020. Searching for a search method: Benchmarking search algorithms for generating NLP adversarial examples. In *Proceedings of the Third BlackboxNLP Workshop on Analyzing and Interpreting Neural Networks for NLP*, pages 323–332. Association for Computational Linguistics.

Yuan Zang, Fanchao Qi, Chenghao Yang, Zhiyuan Liu, Meng Zhang, Qun Liu, and Maosong Sun. 2020. Word-level textual adversarial attacking as combinatorial optimization. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 6066–6080. Association for Computational Linguistics.

Xiang Zhang, Junbo Jake Zhao, and Yann LeCun. 2015. Character-level convolutional networks for text classification. In *Advances in Neural Information Processing Systems 28: Annual Conference on Neural Information Processing Systems 2015, December 7-12, 2015, Montreal, Quebec, Canada*, pages 649–657.

A Appendix

Additional Details on Model

The TextCNN has a 300-dimensional GloVe embedding layer (Pennington et al., 2014), a convolutional layer containing 150 filters with windows sizes $(3, 4, 5)$. The LSTM also has a 300-dimensional GloVe embedding layer and a bi-directional LSTM layer composed of 150 units. We use the *distilbert-base-uncased* as DistilBERT (Sanh et al., 2019), which is a fast Transformer model with 40% fewer parameters than BERT. The model used in Table 2 and Table 3 have the accuracy on clean dataset as follow: (LSTM, Yelp: 92.1%; LSTM, MR: 80.3%; TextCNN, Yelp: 91.4%; TextCNN, MR: 79.2%; DistilBERT, MR: 83.9%). Under our setting, attacking a base uncased version of BERT takes approximately 4 times as long as attacking a base uncased version of DistilBERT, and the beam search ($w = 4$), beam search ($w = 8$), Genetic algorithm, and PSO fails to attack 500 examples within one week when attacking the examples from Yelp on GloVe search space.

Additional Case Study

We give the case study of the adversarial examples crafted with PARSE in Table 6-12. The ~~green~~ word is the original word, and the following red word is the substitution.

Method	Perturbed Texts
PARSE($w = 1$)	There is a general ~~air~~ vent of ~~exuberance~~ ardour in all about the benjamins that's hard to resist.
PARSE($w = 2$)	There is a general ~~air~~ notification of ~~exuberance~~ fervor in all about the benjamins that's hard to resist.
PARSE($w = 4$)	There is a general ~~air~~ propaganda of ~~exuberance~~ eagerness in all about the benjamins that's hard to resist.
PARSE($w = 8$)	There is a general ~~air~~ propaganda of ~~exuberance~~ eagerness in all about the benjamins that's hard to resist.

Table 6: The case study of the adversarial examples crafted by attacking LSTM on the MR dataset in HowNet search space with PARSE.

Method	Perturbed Texts
PARSE($w = 1$)	Kids should have a stirring time at this ~~beautifully~~ prettily drawn ~~movie~~ cartoon. And adults will at least have a dream image of the west to ~~savor~~ taste whenever the film's lamer instincts are in the saddle.
PARSE($w = 2$)	Kids should have a stirring time at this ~~beautifully~~ prettily drawn ~~movie~~ cartoon. And adults will at least have a dream image of the west to ~~savor~~ taste whenever the film's lamer instincts are in the saddle.
PARSE($w = 4$)	Kids should have a stirring time at this ~~beautifully~~ pretty drawn ~~movie~~ cartoon. And adults will at least have a dream image of the west to ~~savor~~ taste whenever the film's lamer instincts are in the saddle.
PARSE($w = 8$)	Kids should have a stirring time at this ~~beautifully~~ wonderfully drawn ~~movie~~ cartoon. And adults will at least have a dream image of the west to ~~savor~~ taste whenever the film's lamer instincts are in the saddle.

Table 7: The case study of the adversarial examples crafted by attacking LSTM on the MR dataset in HowNet search space with PARSE.

Method	Perturbed Texts
PARSE($w = 1$)	It's dark but has ~~wonderfully~~ bizarrely ~~funny~~ recreational moments; you care about the characters; and the action and special effects are first-rate.
PARSE($w = 2$)	It's dark but has ~~wonderfully~~ bizarrely ~~funny~~ recreational moments; you care about the characters; and the action and special effects are first-rate.
PARSE($w = 4$)	It's dark but has ~~wonderfully~~ bizarrely ~~funny~~ recreational moments; you care about the characters; and the action and special effects are first-rate.
PARSE($w = 8$)	It's dark but has ~~wonderfully~~ suspiciously ~~funny~~ recreational moments; you care about the characters; and the action and special effects are first-rate.

Table 8: The case study of the adversarial examples crafted by attacking LSTM on the MR dataset in HowNet search space with PARSE.

Method	Perturbed Texts
PARSE($w = 1$)	In the director's cut, the film is not only a love song to the movies but it also is more fully an example of the kind of lush, all-enveloping movie ~~experience~~ aftertaste it ~~rhapsodizes~~ talks.
PARSE($w = 2$)	In the director's cut, the film is not only a love song to the movies but it also is more fully an example of the kind of lush, all-enveloping movie ~~experience~~ aftertaste it ~~rhapsodizes~~ talks.
PARSE($w = 4$)	In the director's cut, the film is not only a love song to the movies but it also is more fully an example of the kind of lush, all-enveloping movie ~~experience~~ aftertaste it ~~rhapsodizes~~ lectures.
PARSE($w = 8$)	In the director's cut, the film is not only a love song to the movies but it also is more fully an example of the kind of lush, all-enveloping movie ~~experience~~ aftertaste it ~~rhapsodizes~~ lectures.

Table 9: The case study of the adversarial examples crafted by attacking LSTM on the MR dataset in HowNet search space with PARSE.

Method	Perturbed Texts
PARSE($w = 1$)	A ~~smart~~ brainy and ~~funny~~ ridiculous, albeit sometimes superficial, cautionary ~~tale~~ narration of a ~~technology~~ tech in search of an artist.
PARSE($w = 2$)	A ~~smart~~ brainy and ~~funny~~ ridiculous, albeit sometimes superficial, cautionary ~~tale~~ narration of a ~~technology~~ tech in search of an artist.
PARSE($w = 4$)	A ~~smart~~ brainy and ~~funny~~ ridiculous, albeit sometimes superficial, cautionary ~~tale~~ narration of a ~~technology~~ tech in search of an artist.
PARSE($w = 8$)	A ~~smart~~ brainy and ~~funny~~ laughable, albeit sometimes superficial, cautionary ~~tale~~ story of a ~~technology~~ tech in search of an artist.

Table 10: The case study of the adversarial examples crafted by attacking LSTM on the MR dataset in HowNet search space with PARSE.

Method	Perturbed Texts
PARSE($w = 1$)	The ~~wonderfully~~ curiously ~~lush~~ drunk morvern callar is pure punk existentialism, and ms. ramsay and her co-writer, liana dognini, have dramatized the alan warner novel, which itself felt like an answer to irvine welsh's book trainspotting.
PARSE($w = 2$)	The ~~wonderfully~~ curiously ~~lush~~ drunk morvern callar is pure punk existentialism, and ms. ramsay and her co-writer, liana dognini, have dramatized the alan warner novel, which itself felt like an answer to irvine welsh's book trainspotting.
PARSE($w = 4$)	The ~~wonderfully~~ curiously ~~lush~~ drunk morvern callar is pure punk existentialism, and ms. ramsay and her co-writer, liana dognini, have dramatized the alan warner novel, which itself felt like an answer to irvine welsh's book trainspotting.
PARSE($w = 8$)	The ~~wonderfully~~ singularly ~~lush~~ drunk morvern callar is pure punk existentialism, and ms. ramsay and her co-writer, liana dognini, have dramatized the alan warner novel, which itself felt like an answer to irvine welsh's book trainspotting.

Table 11: The case study of the adversarial examples crafted by attacking LSTM on the MR dataset in HowNet search space with PARSE.

Method	Perturbed Texts
PARSE($w = 1$)	The film is hard to dismiss – moody, thoughtful, and ~~lit~~ fainted by ~~flashes~~ blazes of mordant ~~humor~~ animation.
PARSE($w = 2$)	The film is hard to dismiss – moody, thoughtful, and ~~lit~~ fainted by ~~flashes~~ blazes of mordant ~~humor~~ animation.
PARSE($w = 4$)	The film is hard to dismiss – ~~moody~~ listless, thoughtful, and lit by ~~flashes~~ winks of mordant ~~humor~~ animation.
PARSE($w = 8$)	The film is hard to dismiss – ~~moody~~ listless, thoughtful, and lit by ~~flashes~~ blazes of mordant ~~humor~~ vividness.

Table 12: The case study of the adversarial examples crafted by attacking LSTM on the MR dataset in HowNet search space with PARSE.

A Closer Look at Parameter Contributions When Training Neural Language and Translation Models

Raúl Vázquez[♠] **Hande Celikkanat**[♠] **Vinit Ravishankar**[♢] **Mathias Creutz**[♠] **Jörg Tiedemann**[♠]
[♠]Department of Digital Humanities, University of Helsinki
[♢]Language Technology Group, Department of Informatics, University of Oslo
[♠]{firstname.lastname@}@helsinki.fi [♢]vinitr@ifi.uio.no

Abstract

We analyze the learning dynamics of neural language and translation models using Loss Change Allocation (LCA), an indicator that enables a fine-grained analysis of parameter updates when optimizing for the loss function. In other words, we can observe the contributions of different network components at training time. In this article, we systematically study masked language modeling, causal language modeling, and machine translation. We show that the choice of training objective leads to distinctive optimization procedures, even when performed on comparable Transformer architectures. We demonstrate how the various Transformer parameters are used during training, supporting that the feed-forward components of each layer are the main contributors to the optimization procedure. Finally, we find that the learning dynamics are not affected by data size and distribution but rather determined by the learning objective.

1 Introduction

Neural models and Transformers in particular have achieved a great performance in almost every natural language processing (NLP) task. However, they largely remain black-box systems despite various efforts to analyze them. Much work has been devoted to understanding the dense representations that are created during training (Pavlick, 2022; Saphra, 2021) in an attempt to see whether known linguistic properties are present and how they are expressed in the model (Vulić et al., 2020; Raganato et al., 2020; Serrano and Smith, 2019). However, the intrinsic dynamics of the training procedure itself have not been analyzed in depth for highly complex network architectures. In this paper, we take a closer look into the process of parameter optimization, aiming at

- identifying the network components that contribute to the optimization of the loss function,
- revealing the effects of learning objectives and network components on training dynamics,
- detecting the impact of training data size and distribution on learning dynamics and parameter updates.

In order to enable a systematic comparison, we set out with a standard architecture that we train from scratch for each individual task. We use the Transformer architecture (Vaswani et al., 2017) with its common parametrization and size, and train it with three popular learning objectives: masked language modeling (MLM) (Taylor, 1953; Devlin et al., 2019), causal language modeling (CLM), and neural machine translation (NMT). The latter is an interesting special case as it is designed as a conditional language model in a sequence-to-sequence architecture with encoder and decoder components.

For the analysis, we use Loss Change Allocation (LCA) (Lan et al., 2019), an indicator of the contribution of individual parameters to the decrease of the total loss. Its accumulative properties provide us with a tool to perform a fine-grained analysis of the opaque optimization process of complex Transformer-based neural NLP models. We can, therefore, examine how individual layers and sub-layer components contribute to the overall loss function. This confers a unique view of the training process and how the network parameters behave during the learning procedures. Our analysis provides insight over the training dynamics broken down over the model layers and sub-layer components. On the one hand, we observe key differences in the learning dynamics across the objective arise from the nature of the loss function; depending whether being trained for gap-filling or auto-regressive prediction. On the other hand, we see clear differences linked to the nature of the sub-layer components.

We use a consistent experimental setup to enable a systematic and fair comparison between mod-

Proceedings of the 29th International Conference on Computational Linguistics, pages 4788–4800
October 12–17, 2022.

els and learning objectives. With this, we can observe all parameters when training our models from scratch. The methodology and setup are described in detail in sections 2 and 3 before we move on to our analyses and discussion of results in section 4.

2 Methodology

2.1 Neural architecture

All the models in our experiments optimize for cross-entropy loss and are based on the Transformer architecture, which currently dominates the entire field of NLP. We use a consistent setup of overall 12 layers; for the machine translation model the layers are divided into 6 each for the encoder and the decoder. Each layer is comprised of multi-headed attention and feed-forward modules that also pass information via residual connections. The essential parameters of the multi-headed attention network include the so-called key W^K, query W^Q, value W^V and out W^O parameters.[1] We refer to those parameters as the **self-attention block**. Key, query and value are used independently H times to compute a self-attention output vector, where H is the number of heads. The heads are concatenated and down-projected to the dimension of the model d_h via the out linear transformation W^O, to be passed to a two-layered feed-forward network (FFwd) creating the dense-layer representation. If we obviate the biases, dropout and normalization steps, we describe the interaction of the trainable parameters (the W parameter matrices) in one layer of the Transformer as follows:

$$\Phi = X\ W_i^{\Phi}\ ; \quad \Phi = K, Q, V \tag{1}$$

$$Z_i = softmax(\frac{QK^T}{\sqrt{d_k}})V\ ; \quad i = 1:H$$

$$Z = [Z_1, \ldots, Z_H]\, W^O \tag{2}$$

$$FF = W_2\ RelU(W_1 Z) \tag{3}$$

All our models are set to $d_h = 512$, $d_k = 64$ and $H = 8$, as is common practice. The decoder layers used in the MT systems use an additional encoder-decoder attention module in which the keys and queries are computed using the last encoder layer state, while the values come from the module's input signal.

[1] W denote the parameter matrices, as in Alammar (2018)

2.2 Training objectives

Machine Translation In the traditional encoder-decoder approach to neural MT, the model predicts words in the target sentence, word by word, i.e., given a source sentence $X = (x_1, \ldots, x_{T_x})$, a target sentence $Y = (y_1, \ldots, y_{T_y})$ and a model parametrized by θ, at each step the model learns to provide estimates of the conditional probability distribution $P(y_i|X, y_1, \ldots y_{i-1}, \theta)$. For this, we train the standard Transformer for performing translation and then analyze its encoder and decoder.

Causal Language Modeling CLMs estimate the probability of a word given the previous words in a sentence. Formally, the model is trained with inputs $X = (x_1, ..., x_{i-1})$ and outputs $Y = (x_i)$, to estimate $P(x_i|x_1, ..., x_{i-1}, \theta)$. We follow Radford and Narasimhan (2018) and use stacked Transformer decoder layers as models.

Masked Language Modeling The MLM objective is designed for predicting randomly masked tokens from an input sentence. For our experiments, we adopt the RoBERTa (Liu et al., 2020b) implementation of the MLM objective (Devlin et al., 2019). Namely, we sample 15% of the tokens to be predicted and replace the corresponding input token by [MASK] 80% of the time, with a random token 10% of the time and the other 10% is left unchanged. This is done with dynamic masking, so we generate the masking pattern every time we feed a sequence.

For a sentence $(x_1, \ldots, x_T)$, where token x_i is replaced with [MASK], the input to the model is $X = (x_1, \ldots, x_{i-1}, [\text{MASK}], x_{i+1}, ..., x_T)$ and the output $Y = (x_i)$, by estimating $P(x_i|X, \theta)$.

2.3 Loss Change Allocation

We use Loss Change Allocation (LCA) (Lan et al., 2019) because it allows for a fine-grained analysis of each network component. It measures the contributions of each parameter to the change in the loss function at each gradient update by decomposing the components of an approximate path integral along the training trajectory using a Runge-Kutta integrator (Runge, 1895; Kutta, 1901).

To compute the change in loss due to a parameter $\varphi \in \mathbb{R}^K$ update from step t to $t+1$, the LCA uses the first order approximation for the change in loss during the $(t+1)$-th training step

$$L(\varphi_{t+1}) - L(\varphi_t) \approx \sum_{i=0}^{K-1} (\nabla\varphi L(\varphi_t))^{(i)} (\varphi_{t+1}^{(i)} - \varphi_t^{(i)}),$$

where $\eta^{(i)}$ represents the i-th entry of any vector η, and $\nabla_\varphi L(\varphi_t)$ is the gradient of the loss with respect to φ evaluated at φ_t.

This formulation makes it possible to aggregate the LCA over higher-level breakdowns because the sum of all individual components equals the total change in loss. This also holds for layer- and sub-layer accumulations as each contribution has the same fundamental units as the loss: *nats* for models that optimize for cross-entropy loss, as is our case.

The LCA measure allows us to identify loss decreases at a per-parameter, per-timestep level. Put simply, a negative LCA at a given parameter update translates into that parameter being beneficial for the optimization process because the LCA is negative when that parameter's component of the gradient is negative. Conversely, positive LCA is "hurting" the learning process, for instance caused by noisy mini-batches where the gradient points in the wrong direction or has too large a step size for an irregular loss landscape (Lan et al., 2019; Jastrzębski et al., 2019; Xing et al., 2019). In other words, the raw value of the LCA is an intuitively interpretable statistic that allows analyzing the training dynamics.

3 Experimental setup

Models. We apply the common Transformer *base* settings of Vaswani et al. (2017). For consistency, we train all the models using the same toolkit, fairseq (Ott et al., 2019). We record the LCA at every parameter update and refer the reader to Appendix A for a details on model hyper-parameters.

Data. In our main experiments we use the English - German portion of Europarl (Koehn, 2005). We download, preprocess and split it using OpusTools (Aulamo et al., 2020). We use 2.5k sentences for dev and test, leaving a total of 1.9M sentences in the training data. For all systems we learn and apply byte-pair-encoding (BPE) segmentation (Sennrich et al., 2016; Kudo and Richardson, 2018). For the MT systems we use a 32k vocabulary shared among source and target, while for the LMs we train monolingual models, also with 32k vocabulary.

We perform additional experiments to investigate how different training signals may affect the LCA, and hence the validity and limitations of our conclusions. For these, we use data in different languages (English, Estonian, German, Finnish), domains (Europarl, WMT[2]), and of different sizes (700k, 2M and 5M utterances). Details about all experiments' data and model settings can be found in Appendices A and B.

4 Analyzing the training dynamics and parameter contributions

In this section we systematically dissect the development of the training process using the LCA indicator. In section 4.1, we look at the overall picture by aggregating the contributions throughout the entire training process. The aggregated figures showcase differences that originate from the training tasks and contrast the monolingual LMs with the sequence-to-sequence MT model. In section 4.2 we zoom into the iterative training procedure to further analyze the dynamics of learning network parameters over time. In section 4.3 we analyze the individual attention heads and in section 4.4 we look at the parameters that *hurt* the optimization process.

4.1 Layer-wise contributions

In Figure 1 we depict the LCA aggregated over all parameters within each layer component and summed over all time steps. The plots reveal particular characteristics for the components of each model, evidencing that the training dynamics are highly dependent on nature of the sub-layers. In particular, the difference in the order of magnitude across the components, indicates that the optimization process relies heavily on the feed-forward components.

Across modeling objectives, the clearest difference can be seen in the shapes of the **self-attention block**; the key, query, value and out parameters in eq. (1) and (2). The MLM shows a concave trend, where the main contributions are on the last layers and, to some degree, on the first two layers, while the middle layers display minor contributions. For CLM, contributions are higher for layers 1 and 6-10, with a tendency to be more evenly distributed among all layers. It contrasts with the MLM by emphasizing the penultimate layers, not the final ones, and shows a drop in the final layer contributions. We refer to this bimodal shape as *cup-and-handle shaped* from here on. In the MT setup, it is easy

[2]Including paracrawl, rapid 2016, europarl-v7, news-commentary-v13 and commoncrawl

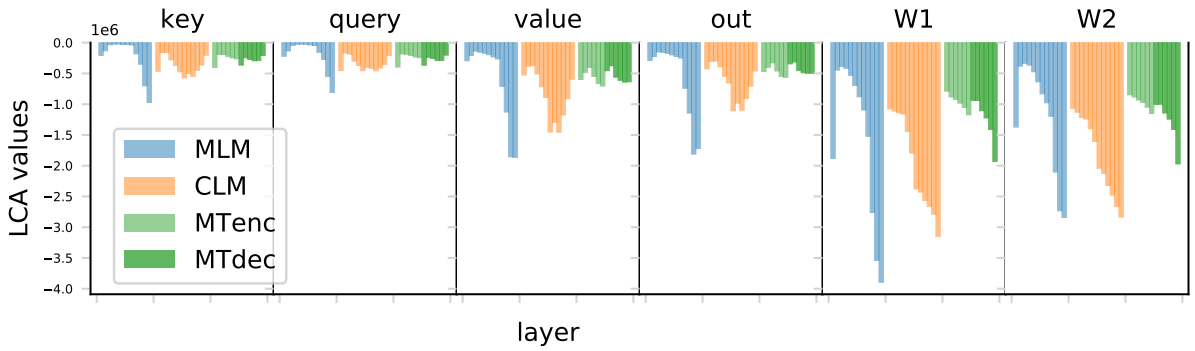

Figure 1: LCA aggregated over all training steps results for individual components of different layers. Each bar location in the x-axis denotes a separate layer.

to identify the transition from the encoder, and we see that the key-query pair behave similarly, as do the value-out pair. The encoder's key-query behaviour peaks at the first layer, and falls drastically at the second layer, showing a slight increase afterwards. Meanwhile, the value-out couple show a more clear concave form with prominent initial and final layers. The decoder's key-query pair show a cup-and-handle shape, and the value-out LCA is similar but does not drastically decrease for the last layer.

The **feed-forward block** components, W_1 and W_2 in eq. (3), show a higher order of magnitude and a similar shape for all models, with the highest accumulated LCA value in the final layer. However, MLM stands out by having strong contributions in the fist layer and a steep increase towards the final layers. Unlike the other models, the MLM shows the same trend for all the sub-layer components. The CLM shows monotonic increases over all layers, with the particularity that layers 4-7 show sharper jumps – noticeably, Layer 7 is where we observe the change in trend for the attention-block components. For the MT model, we again see the separation between encoder and decoder with the highest contributions at the end layers of each module. The encoder shows linear increase, while the decoder behaves more like the CLM model with moderate contributions initially and high increase towards the last layer.

Discussion. Under our controlled experimental setup, it is reasonable to assume that the specific behaviour observed in the learning dynamics stem from the particularities of each task. Specifically, the MLM performs the cloze task: has access to all tokens in the sentence and decodes a few, non-sequential tokens. We hypothesize that this forces MLM to compose output representations simultaneously, leading to a higher amount of information flowing into upper layers. This is consistent with previous research stating that MLMs recreate token identity in later layers, and more contextualized representations (Voita et al., 2019a; Ethayarajh, 2019). In contrast, the CLM has access to previous tokens only and generates the following one. The left-to-right decoding, may also be the reason for the shape of the attention parameters. The model sees only the *past* context, and hence depends more on the middle layers. Some support for this hypothesis comes from Ethayarajh (2019), showing that CLMs produce more context-specific representations as early as layers 4–5, presumably to predict the next word more accurately. Also, Voita et al. (2019a) demonstrate that when advancing across the layers of the [CLM] network, the system loses information about input and accumulates information about output, that is, the context in the CLM case. Moreover, middle layers potentially help learning more fine-grained syntactic information on MLMs (Tenney et al., 2019), a behavior that may become more prominent for CLMs due to its nature, although we leave the testing of this hypothesis for future work.

For the MT model, the task at hand also dictates the observed behaviour. By design, the encoder is more similar to an MLM because it has access to all tokens in the source side, while the decoder has more similarities to a CLM. We observe that the training dynamics behave accordingly – especially the self-attention parameters, with the slight differences arising perhaps from translation being a more complex task (see Voita et al. (2021) for a discussion on how NMT is a composition of several sub-tasks). MT also requires a higher capacity from the model. The encoder–decoder bottleneck triggers this even further making clear why we,

as Zhu et al. (2020), observe peaks at the transition from encoder layers to decoder in W_1 and W_2. The encoder transfers information to the next stage where the decoder starts its generative task.

Finally, we observe higher-magnitude contributions from the feed-forward block for all models may be due to the differences in the update frequencies. This was noted by Zhu et al. (2020) between sparse and dense layer, but this argument can be extended to the difference in feed-forward and attention blocks contributions, since the attention block updates are relative to the token they score from the input (see eq. 1), whereas feed-forward block updates are relative to all tokens. In accordance with Mickus et al. (2022), who show the high relative importance of the feed-forward components in the token representations, our results also suggest that when analyzing Transformer-based models and their representations the feed-forward modules should not be discarded in favor of the more readily interpretable multi-headed attention components.

4.2 The learning dynamics over time

So far, we have pointed out the time-collapsed behaviour in the self-attention and the feed-forward components of each layer. In this section, we show the particular dynamics of the LCA values over the course of training (Figure 2), analyzing the parameters in two broad groups: the feed-forward parameters W_0 and W_1 from eq. (3), and the self-attention block parameter matrices W^{Φ} for the `key`, `query` and `value` parameters from eq. (1). Since the LCA presents a high variance during training, making the learning process fuzzy, as noted by Lan et al. (2019), the trends in Figure 2 were obtained using moving averages with a window of 1k steps.

All models show that the largest LCA values are observed early during training, denoting a fast initial learning phase, after which the gradient descent learning dynamics transitions to a much slower exploration phase, in line with the findings of Feng and Tu (2021). The plots also reveal the convergence property (Raghu et al., 2017) in action, by which earlier layers tend to converge faster. However, the MLM presents a distinctive behavior in the initial and final layers. The embedding layer, as well as layers 1, 10–12 contribute substantially to the overall loss optimization especially in the beginning of the training runs. Moreover, these layers present oscillations for much longer and with larger magnitudes. In contrast, the middle layers do not show substantial activity throughout the entire training process in MLM. These parameters converge early during training, particularly those in the self-attention block. CLM and MT parameter contribution patterns display many similarities in shape across almost all layers and blocks. MT displays a smoother convergence than CLM, especially for layers 7–10. The most significant contributions in CLM clearly come from layers 7–9 in the attention block and the layers 7–12 of the feed-forward block. These higher activations in the middle layers also exhibit the most striking differences to the MT model.

Discussion. We observe that some of the parameters settle remarkably early, with very minor contributions to the loss decrease after an initial phase. This is in line with Lan et al. (2019), who, after analyzing the noisiness of the training process, suggest to freeze parameters that harm the optimization process since it sheds light on the connection between LCA and how parameters are optimized for the task at hand. In other words, some parameters are not that relevant after the initial training phase, where the loss decreases drastically. On the contrary, the MLM later layers take a significantly longer time to settle. Feng and Tu (2021) show that in SGD optimization with mislabeled data points, the parameters go through a late retrofitting phase in which they try to learn to account for the mislabeled data. In our case, we do not have a set of correct-incorrect labels, but we do the random masking for the cloze task on-the-fly, implying that the training data contains a lot of examples with easy-to-learn rules, and some rarer harder-to-generalize exceptions. We draw a comparison to such retrofitting in the observed oscillations of the MLM initial and final layers, since MLM mostly induces learning at independent lexical level (as encoded in the initial embedding layer) and the most contextualized layers at the end of the self-attention chain.

As noted in section 4.1, the essential operational difference across objectives can explain the behavioral similarities of CLM and MT in comparison to MLM. The latter extensively uses contextualized information to predict lexical gaps and, therefore, requires *strong word embeddings* and a careful connection of *contextual clues in both directions*. CLM and MT, on the other hand seem to require more iterative knowledge from all layers forcing more

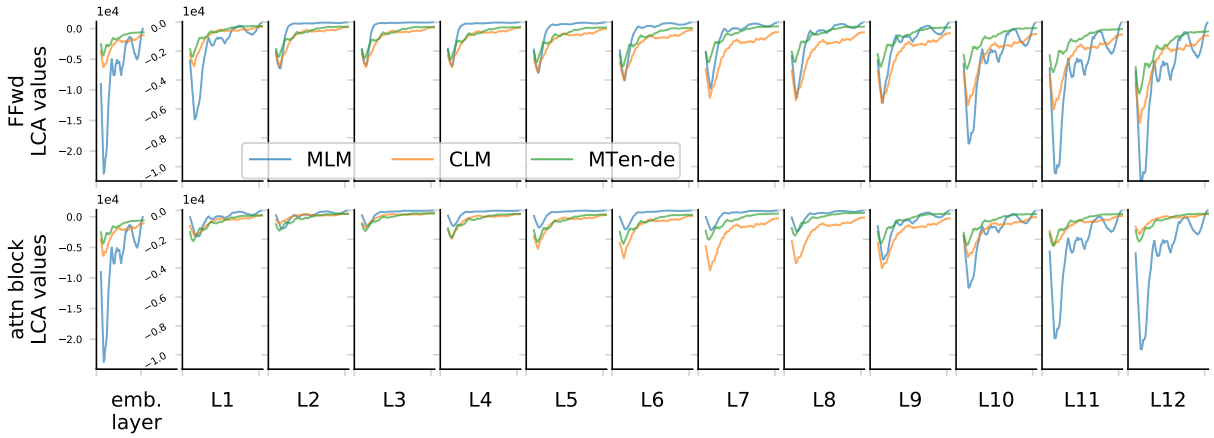

Figure 2: Training dynamics trends decomposed over time for the three learning objectives. The embeddings layer plot is the same on both rows, and it shows a different scale for visualization purposes.

adjustments throughout the entire network.

When focusing on the *difference between attention and feed-forward components*, we observe that, compared to the attention parameters, the feed-forward components of CLM- and MT-objective models seem to converge at a much later stage and have a higher contribution to the loss decrease, particularly at layers 10–12. This backs up the observations made by Wang and Tu (2020), where self-attentive components were empirically found to be less important for machine translation. For reference, the mechanism by which these feed-forward components operate is described by Geva et al. (2021). Previous studies have also concluded that attention in MT relies heavily on fixed positional connections (Raganato et al., 2020; Voita et al., 2019b). This means that their contribution to reducing the loss is limited once the essential attention patterns have been established and learned.

4.3 The training dynamics of individual attention heads.

Aggregating over all attention heads as we have done so far could hide variations present across them. Therefore, in Figure 3 we plot the contributions of each individual head over the entire training period, for each of the self-attention block parameter matrices W^{Φ} from eq. (1). First, we observe some redundancy in that many of the differences between contributions measured across heads of the same layer are negligible. Second, we also see that some layers' heads range from light green to dark violet (see e.g., MLM-1, -9 & -12, CLM-3 & -7, MT-5 & -7). These simultaneous behaviours can be tied to the idea of attention heads specialization (Voita et al., 2019b), by which some heads serve specific purposes. The observations also provide support for the observed redundancy in attention heads, which allows to effectively prune models without compromising prediction quality (Michel et al., 2019; Zhang et al., 2021). Moreover, most of the layers presenting no clear differences highlights the similarity in learning dynamics across all of the attention heads. This is consistent with findings from Clark et al. (2019), who argued that the apparent redundancy in BERT's attention heads is the result of attention dropout.

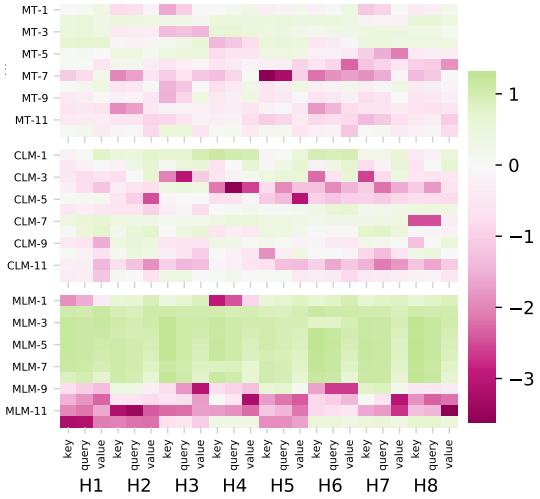

Figure 3: Contribution from individual attention heads, presented as normalized LCA values. Normalization is performed over the entire heatmap.

4.4 Noise in the optimization process

Using LCA to analyze the optimization process provides an additional insight to it, by identifying if there are parameters that *hurt* the training process,

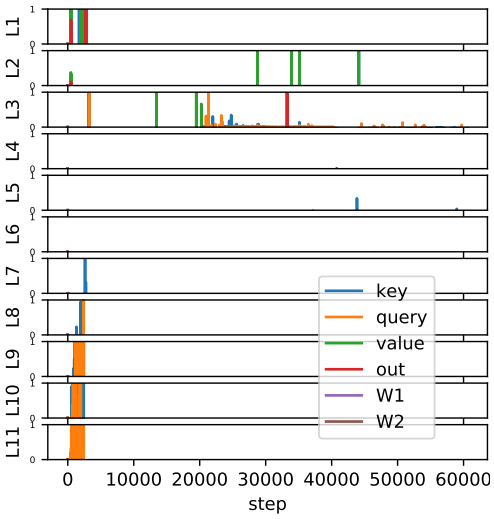

Figure 4: An overview of the MLM training process. To focus on the positive LCA we only show the interval [0,1]. We observe that positive LCA in later stages of training is present almost exclusively in attention block parameters, on layers 2–5, consistent with Fig. 1.

causing the total loss to increase instead of decrease. We check whether there are any such parameters in our models, and if so, at which phase during training they hurt the loss optimization. Lan et al. (2019) argue that the training process is notoriously noisy and they even recognize that some full layers hurt the training process.

	CLM	MLM	MT
emb.	2	0	2
key	29	1565	64
query	24	2055	64
value	24	24	130
out	24	6	113
W1	24	0	59
W2	24	0	131
Last step w/positive	1072	59727	2673
Overall max.	5.58 (key)	68.18 (value)	95 (W2)

Table 1: Summary of parameters that hurt the training process. We show the number of times a block of parameters has positive LCA, the last training step where this happens, and the magnitude of the parameter with the highest positive LCA value.

In our experiments, we rarely observe this phenomenon for the CLM and MT objectives. The *loss-hurting behavior* only seems to occur at the beginning phase of training, specifically during the first couple of thousand steps, which corresponds to at most 3% of the total time. Moreover, the magnitude of such parameters is also rather low, in the order of $1e1$. However, it should be noted that, since we are observing the component-wise aggregated contributions of parameters, we expect the effect to be "neutralized" by highly negative parameters within the corresponding components, resulting in the relatively low sums.

A different behavior is again shown by the MLM-objective (Figure 4). The MLM objective presents several positive LCA values occurring at much later stages of training. We interpret this as evidence of a comparatively noisier optimization process for the MLM objective. This is consistent with section 4.2, where we see that the MLM parameters converge much later. We depict the positive LCA values from different layers of MLM in Figure 4 to understand if such harmful contributions are specific to single layers, or are distributed over the entire network. The results show fairly clear localization to certain components of the lower layers especially for the later stages of training.

We emphasize that, contrary to findings in Lan et al. (2019), in none of our experiments do we observe a full layer or sub-layer damaging the training process. This might be due to the nature of the difference in the task, architectures, and the size of the networks used in their analysis. [3]

4.5 Effect of data size and distribution

To see the robustness of our findings against varying training datasets, we replicated the experiments presented so far using training data of different sizes (700k, 2M and 5M), domains (Europarl and WMT) and languages (English, German, Finnish and Estonian).[4] A summary of results is presented in Figure 5, with detailed plots for each setting available in the appendix B.2, Figure 6.

These additional experiments show consistent training-dynamic trends, regardless of the immediate training data properties. We observe two differences. First, the learning dynamics of the Estonian MLM system, based on smaller data, are the most different.[5] We do not observe a clear concave shape in the attention block components as for the other languages. Also, we see that the initial layers are more heavily responsible for the loss optimization. Trained with smaller data, the Estonian MLM still presents small contributions of the parameters in

[3] The authors analyze networks trained for image processing, using convolutional NNs with 5, 9 and 20 layers.

[4] For all the experiments, we use the same models and hyper-parameters (see Appendix A)

[5] We mark it with a dashed line and Figure 6 in the appendix shows the individual plots

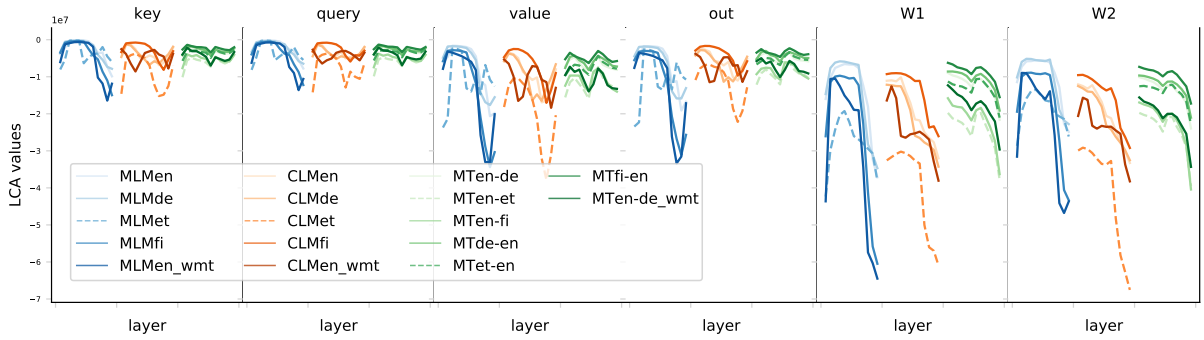

Figure 5: LCA aggregated over all training steps results for individual components of different layers. Each color corresponds to a loss function and each shade to a different model.

the middle layers, a characterizing behavior seen in all of the MLM systems we trained. Second, we observe that the CLM objective for English and German have a higher response in relatively early layers of the attention components, when compared to Finnish and Estonian. However note that the general CLM trend towards peaking contribution in penultimate layers exists for all four languages. The other cases seem very robust in their trends to the different settings, and do not present qualitatively differences in their behavior.

We interpret the observed stability of the results not only as a way to verify their validity, but also as an indication that the parameters' contributions to the optimization process are determined mainly by the choice of architecture and objective function and not so much by the training signal in the data. In particular, we argue that the training dynamics we observe are independent of the particular syntactic structure of different languages. On top of this, variations on data domains and size of the training data, show how the parameters' contributions to the change in loss are not tied to the semantic information of the training data. On the contrary, changes in the trends consistently correlate with the objective functions, over all those different settings.

5 Related work

The role of the inner representations, and internal behaviors of neural models in NLP have been extensively studied (Rogers et al., 2021; Pavlick, 2022). Relevant directions investigate, for instance, the geometry of the learned representational spaces (Ethayarajh, 2019; Vázquez et al., 2021), probing the learned representations with simple classifiers (Vulić et al., 2020; Apidianaki and Garí Soler, 2021), and the attention patterns at each step in the sequence (Voita and Titov, 2020; Raganato et al., 2020). Regarding the attention mechanism, while in the past, analyses have tended to focus on the role of the self-attention mechanism as an explanatory mechanism (Jain and Wallace, 2019; Serrano and Smith, 2019), recent work has been more nuanced about whether this is even necessary (Bastings and Filippova, 2020). Wiegreffe and Pinter (2019) argue that the attentive components are not inherently separable from the rest of the model; this has sparked analyses of attention models not directly involving attention weights. Kobayashi et al. (2020) analyse vector norms and find reasonable word alignment, while Geva et al. (2021) show that feed-forward components act as key-value stores. Ravishankar et al. (2021) analyse the effect of freezing different components on the interpretability of the attention mechanism, and Mickus et al. (2022) use a novel decomposition of Transformer embeddings to isolate the impact of the network components, showing that multi-head attention accounts for a small proportion of the embeddings.

Most such works tend to test and analyse fixed model checkpoints While there is value to analyzing these models over the course of the training process, much work in this direction emerged from outside the NLP community; e.g., LCA was first proposed and tested using only vision classification datasets (Lan et al., 2019), and it has been used since in NLP to analyze NMT training dynamics (Zhu et al., 2020). Saphra and Lopez (2019) present another early work on the analysis of the training dynamics of LMs; Kaplan et al. (2020) investigate the language modeling loss on model architecture, size, amount of training data and computing power. Their work was key for the development of large-scale pretrained models like GPT-3 (Brown et al., 2020) and Switch-Transformers (Fedus et al.,

2021). Liu et al. (2020a) analyze the training dynamics of Transformer-based models based on the stability of the gradients, leading them to propose an adaptive model initialization method.

6 Conclusions

In this work, we take a closer look at the learning dynamics of Transformer-based models using the loss change allocation (LCA) indicator. Specifically, we analyze the training dynamics for (1) masked language modeling, (2) causal language modeling, and (3) machine translation training objectives. We find that the training dynamics seem to be delimited by the training objective being either a gap-filling task or an auto-regressive prediction task, despite the similarities that the models present. In particular, the different layers of each model behave accordingly to the training objective used. We also show that the feed-forward blocks contribute more than the attention blocks to the decrease in loss during training, a finding that strengthens the relative importance of the former, and invites more research to understand this currently more obscure component.

We also show that these dynamics are robust against changes in the size and distribution of the training data. The overall trends stay the same with different domains and languages reassuring that our findings are not just an artefact of a specific setup. In future work, we would like to study the training dynamics of systems trained with different optimizers and regularization techniques, such as, attention dropout or data augmentation. We would also like to investigate the influence of noisy training data on the training process. For this, we can apply controlled shuffling to the training data to influence the noise level and gradually remove syntactic information from the training signal.

Acknowledgments

This work is part of the FoTran project, funded by the European Research Council (ERC) under the European Union's Horizon 2020 research and innovation programme (grant agreement № 771113).

References

Jay Alammar. 2018. The illustrated transformer.

Marianna Apidianaki and Aina Garí Soler. 2021. ALL dolphins are intelligent and SOME are friendly: Probing BERT for nouns' semantic properties and their prototypicality. In *Proceedings of the Fourth BlackboxNLP Workshop on Analyzing and Interpreting Neural Networks for NLP*, pages 79–94, Punta Cana, Dominican Republic. Association for Computational Linguistics.

Mikko Aulamo, Umut Sulubacak, Sami Virpioja, and Jörg Tiedemann. 2020. OpusTools and parallel corpus diagnostics. In *Proceedings of The 12th Language Resources and Evaluation Conference*, pages 3782–3789. European Language Resources Association.

Jasmijn Bastings and Katja Filippova. 2020. The elephant in the interpretability room: Why use attention as explanation when we have saliency methods? In *Proceedings of the Third BlackboxNLP Workshop on Analyzing and Interpreting Neural Networks for NLP*, pages 149–155, Online. Association for Computational Linguistics.

Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel Ziegler, Jeffrey Wu, Clemens Winter, Chris Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. 2020. Language models are few-shot learners. In *Advances in Neural Information Processing Systems*, volume 33, page 1877–1901. Curran Associates, Inc.

Kevin Clark, Urvashi Khandelwal, Omer Levy, and Christopher D. Manning. 2019. What does BERT look at? an analysis of BERT's attention. In *Proceedings of the 2019 ACL Workshop BlackboxNLP: Analyzing and Interpreting Neural Networks for NLP*, pages 276–286, Florence, Italy. Association for Computational Linguistics.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, Minneapolis, Minnesota. Association for Computational Linguistics.

Kawin Ethayarajh. 2019. How contextual are contextualized word representations? comparing the geometry of BERT, ELMo, and GPT-2 embeddings. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the*

9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP), pages 55–65, Hong Kong, China. Association for Computational Linguistics.

William Fedus, Barret Zoph, and Noam Shazeer. 2021. Switch transformers: Scaling to trillion parameter models with simple and efficient sparsity. *CoRR*, abs/2101.03961.

Yu Feng and Yuhai Tu. 2021. Phases of learning dynamics in artificial neural networks in the absence or presence of mislabeled data. *Machine Learning: Science and Technology*, 2(4):043001.

Mor Geva, Roei Schuster, Jonathan Berant, and Omer Levy. 2021. Transformer Feed-Forward Layers Are Key-Value Memories. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 5484–5495, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Sarthak Jain and Byron C. Wallace. 2019. Attention is not Explanation. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 3543–3556, Minneapolis, Minnesota. Association for Computational Linguistics.

Stanisław Jastrzębski, Zachary Kenton, Nicolas Ballas, Asja Fischer, Yoshua Bengio, and Amost Storkey. 2019. On the relation between the sharpest directions of DNN loss and the SGD step length. In *International Conference on Learning Representations*.

Jared Kaplan, Sam McCandlish, Tom Henighan, Tom B. Brown, Benjamin Chess, Rewon Child, Scott Gray, Alec Radford, Jeffrey Wu, and Dario Amodei. 2020. Scaling laws for neural language models. *CoRR*, abs/2001.08361.

Diederik P Kingma and Jimmy Ba. 2015. Adam: A method for stochastic optimization. *International Conference on Learning Representations (ICLR)*.

Goro Kobayashi, Tatsuki Kuribayashi, Sho Yokoi, and Kentaro Inui. 2020. Attention is not only a weight: Analyzing transformers with vector norms. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 7057–7075, Online. Association for Computational Linguistics.

Philipp Koehn. 2005. Europarl: A parallel corpus for statistical machine translation. In *Proceedings of Machine Translation Summit X: Papers*, pages 79–86, Phuket, Thailand.

Taku Kudo and John Richardson. 2018. SentencePiece: A simple and language independent subword tokenizer and detokenizer for neural text processing. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 66–71, Brussels, Belgium. Association for Computational Linguistics.

W. Kutta. 1901. Beitrag zur näherungsweisen Integration totaler Differentialgleichungen. *Zeit. Math. Phys.*, 46:435–53.

Janice Lan, Rosanne Liu, Hattie Zhou, and Jason Yosinski. 2019. Lca: Loss change allocation for neural network training. *Advances in Neural Information Processing Systems*, 32:3619–3629.

Liyuan Liu, Xiaodong Liu, Jianfeng Gao, Weizhu Chen, and Jiawei Han. 2020a. Understanding the difficulty of training transformers. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 5747–5763, Online. Association for Computational Linguistics.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2020b. Roberta: A robustly optimized bert pretraining approach.

Paul Michel, Omer Levy, and Graham Neubig. 2019. Are sixteen heads really better than one? In *Advances in neural information processing systems*, volume 32. Curran Associates, Inc.

Timothee Mickus, Denis Paperno, and Mathieu Constant. 2022. How to dissect a muppet: The structure of transformer embedding spaces.

Myle Ott, Sergey Edunov, Alexei Baevski, Angela Fan, Sam Gross, Nathan Ng, David Grangier, and Michael Auli. 2019. fairseq: A fast, extensible toolkit for sequence modeling. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics (Demonstrations)*, pages 48–53, Minneapolis, Minnesota. Association for Computational Linguistics.

Ellie Pavlick. 2022. Semantic structure in deep learning. *Annual Review of Linguistics*, 8(1):447–471.

Alec Radford and Karthik Narasimhan. 2018. Improving language understanding by generative pre-training. .

Alessandro Raganato, Yves Scherrer, and Jörg Tiedemann. 2020. Fixed encoder self-attention patterns in transformer-based machine translation. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 556–568, Online. Association for Computational Linguistics.

Maithra Raghu, Justin Gilmer, Jason Yosinski, and Jascha Sohl-Dickstein. 2017. SVCCA: Singular vector canonical correlation analysis for deep learning dynamics and interpretability. In *Advances in Neural Information Processing Systems*. Curran Associates, Inc.

Vinit Ravishankar, Artur Kulmizev, Mostafa Abdou, Anders Søgaard, and Joakim Nivre. 2021. Attention can reflect syntactic structure (if you let it). In *Proceedings of the 16th Conference of the European*

Chapter of the Association for Computational Linguistics: Main Volume, pages 3031–3045, Online. Association for Computational Linguistics.

Anna Rogers, Olga Kovaleva, and Anna Rumshisky. 2021. A Primer in BERTology: What We Know About How BERT Works. *Transactions of the Association for Computational Linguistics*, 8:842–866.

C. Runge. 1895. Ueber die numerische auflösung von differentialgleichungen. *Mathematische Annalen*, 46(2):167–178.

Naomi Saphra. 2021. Training dynamics of neural language models.

Naomi Saphra and Adam Lopez. 2019. Understanding Learning Dynamics Of Language Models with SVCCA. *Proceedings of the 2019 Conference of the North*, pages 3257–3267.

Rico Sennrich, Barry Haddow, and Alexandra Birch. 2016. Neural machine translation of rare words with subword units. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1715–1725, Berlin, Germany. Association for Computational Linguistics.

Sofia Serrano and Noah A. Smith. 2019. Is attention interpretable? In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 2931–2951, Florence, Italy. Association for Computational Linguistics.

Wilson L. Taylor. 1953. "cloze procedure": A new tool for measuring readability. *Journalism Quarterly*, 30(4):415–433.

Ian Tenney, Dipanjan Das, and Ellie Pavlick. 2019. BERT Rediscovers the Classical NLP Pipeline. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 4593–4601.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Proceedings of the 31st International Conference on Neural Information Processing Systems*, NeurIPS'17, page 6000–6010, Red Hook, NY, USA. Curran Associates Inc.

Raúl Vázquez, Hande Celikkanat, Mathias Creutz, and Jörg Tiedemann. 2021. On the differences between BERT and MT encoder spaces and how to address them in translation tasks. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing: Student Research Workshop*, pages 337–347, Online. Association for Computational Linguistics.

Elena Voita, Rico Sennrich, and Ivan Titov. 2019a. The bottom-up evolution of representations in the transformer: A study with machine translation and language modeling objectives. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 4396–4406, Hong Kong, China. Association for Computational Linguistics.

Elena Voita, Rico Sennrich, and Ivan Titov. 2021. Language modeling, lexical translation, reordering: The training process of NMT through the lens of classical SMT. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 8478–8491, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Elena Voita, David Talbot, Fedor Moiseev, Rico Sennrich, and Ivan Titov. 2019b. Analyzing multi-head self-attention: Specialized heads do the heavy lifting, the rest can be pruned. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 5797–5808, Florence, Italy. Association for Computational Linguistics.

Elena Voita and Ivan Titov. 2020. Information-theoretic probing with minimum description length. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 183–196, Online. Association for Computational Linguistics.

Ivan Vulić, Edoardo Maria Ponti, Robert Litschko, Goran Glavaš, and Anna Korhonen. 2020. Probing pretrained language models for lexical semantics. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 7222–7240, Online. Association for Computational Linguistics.

Wenxuan Wang and Zhaopeng Tu. 2020. Rethinking the Value of Transformer Components. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 6019–6029, Barcelona, Spain (Online). International Committee on Computational Linguistics.

Sarah Wiegreffe and Yuval Pinter. 2019. Attention is not not explanation. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 11–20, Hong Kong, China. Association for Computational Linguistics.

Chen Xing, Devansh Arpit, Christos Tsirigotis, and Yoshua Bengio. 2019. A walk with SGD: How SGD explores regions of deep network loss?

Zhengyan Zhang, Fanchao Qi, Zhiyuan Liu, Qun Liu, and Maosong Sun. 2021. Know what you don't need: Single-shot meta-pruning for attention heads. *AI Open*, 2:36–42.

Conghui Zhu, Guanlin Li, Lemao Liu, Tiejun Zhao, and Shuming Shi. 2020. Understanding learning dynamics for neural machine translation. *CoRR*, abs/2004.02199.

A Model hyper-parameters

We train all our models using the fairseq-toolkit [6], using a single Nvidia V100 GPU for 48 hours. In Table 2, we present a summmary of the set of hyper-parameters used in all cases.

Parameter	Value
num. layers	12 [7]
att. heads	8
embeddings dim.	512
ffwd hidden dim.	2048
update-freq	16
precision	fp16
total-num-update	125000
warmup-updates	8000
lr-scheduler	polynomial decay
lr	0.0005
optimizer	adam (2015)
adam-betas	(0.9, 0.98)
adam-eps	1e-06
weight-decay	0.01
clip-norm	0.0
dropout	0.1
attention dropout	0.0
activation-fn	relu
tokens-per-sample	512
max-tokens-per-sample	2048

Table 2: Set of hyper-parameters shared across all our models

B Additional experiments

B.1 Data

In Table 3 we summarize the size of the datasets used in our all experiments. We use the parallel datasets when training MT systems, while for the monolingual LM objective, we use only the target side of the corpus. Specifically for English, we use the source sentences of the En-De datasets.

In a nutshell, the experiments presented in section 4 use the En-De portion of the Europarl dataset, and we test the performance with that test split and the newstest2018. For the experiments referred in section 4.5 we train monolingual LMs in English, German, Finnish and Estonian and bilingual MT systems for En-De, En-Fi, and En-Et, as well as the inverse translation directions De-En, Fi-En and Et-En using the respective Europarl data B.2

Dataset	Languages	# Sentences		
		Train	Val.	Test
Europarl	En-De	1.92M	2.5K	2.5K
	En-Fi	1.91M	2.5K	2.5K
	En-Et	0.7M	2.5K	2.5K
WMT-2018	En-De	5M	3K	-
newstest2018	En-De	-	-	2.9K
	En-Fi	-	-	3.0K
	En-Et	-	-	2.0K

Table 3: Size of the train, validation and test splits for the datasets used in all experiments.

B.2 Results

Figure 6 shows the aggregated LCA values in additional experiments using four different languages and language pairs.

[6] https://github.com/pytorch/fairseq

[7] For the MT models we use 6 layers for each, encoder and decoder

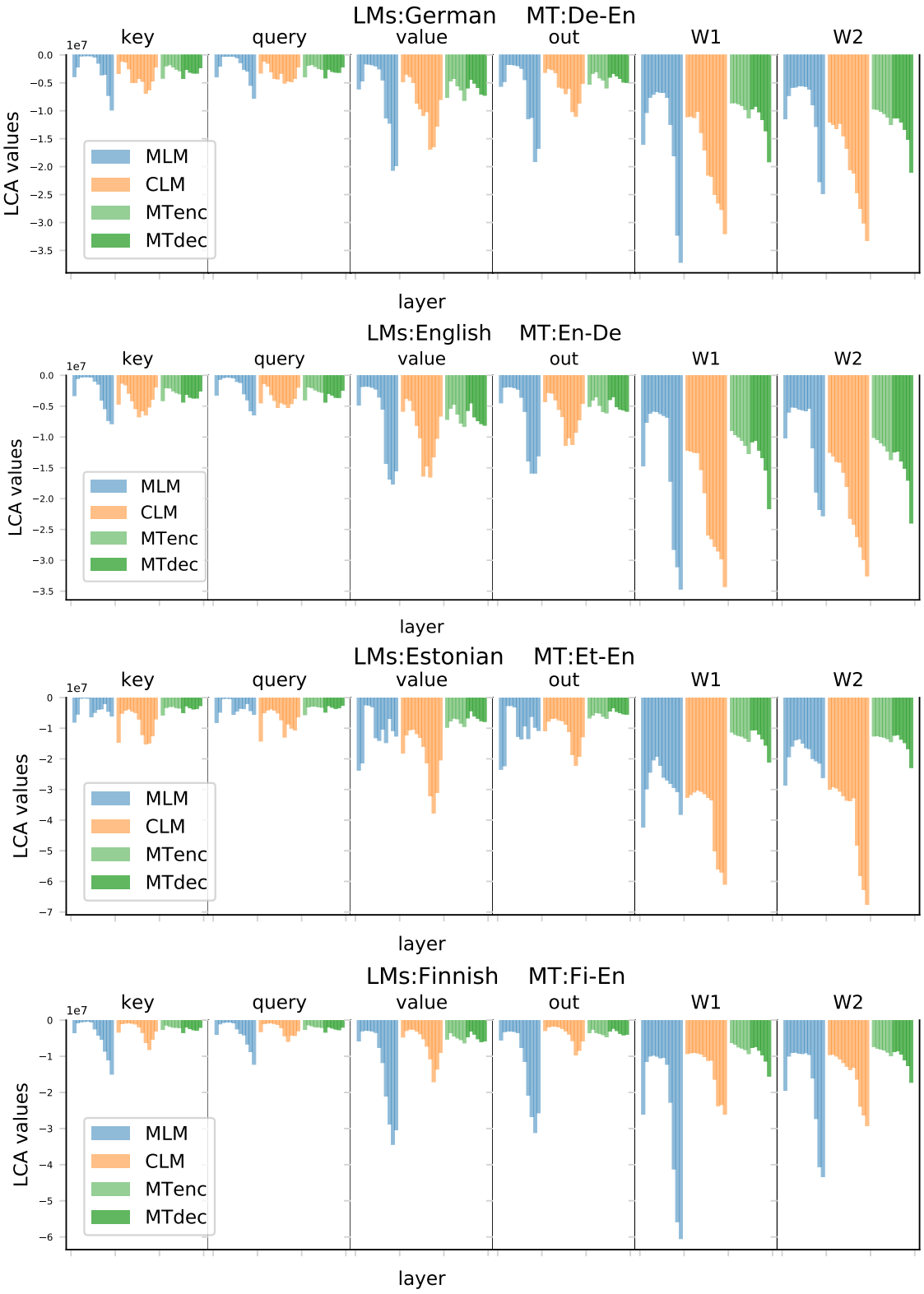

Figure 6: LCA aggregated over all training steps for the individual components of each Transformer layer. Results for all monolingual LMs and MT experiments en-de (with WMT data), de-en, et-en, fi-en.

KNOT: Knowledge Distillation using Optimal Transport for Solving NLP Tasks

Rishabh Bhardwaj **Tushar Vaidya** **Soujanya Poria**
Singapore University of Technology and Design, Singapore
Nanyang Technological University, Singapore
rishabh_bhardwaj@mymail.sutd.edu.sg, tushar.vaidya@ntu.edu.sg
sporia@sutd.edu.sg

Abstract

We propose a new approach, Knowledge Distillation using Optimal Transport (KNOT), to distill the natural language semantic knowledge from multiple teacher networks to a student network. KNOT aims to train a (global) student model by learning to minimize the optimal transport cost of its assigned probability distribution over the labels to the weighted sum of probabilities predicted by the (local) teacher models, under the constraints that the student model does not have access to teacher models' parameters or training data. To evaluate the quality of knowledge transfer, we introduce a new metric, Semantic Distance (SD), that measures semantic closeness between the predicted and ground truth label distributions. The proposed method shows improvements in the global model's SD performance over the baseline across three NLP tasks while performing on par with Entropy-based distillation on standard accuracy and F1 metrics. The implementation pertaining to this work is publicly available at https://github.com/declare-lab/KNOT.

1 Introduction

Due to recent technological advancements, more than two-thirds of the world's population use mobile phones[*]. A client application on these devices has access to the unprecedented amount of data obtained from user-device interactions, sensors, etc. Learning algorithms can employ this data to provide an enhanced experience to its users. For instance, two users living wide apart may have different tastes in food. A food recommender application installed on an edge device might want to learn from user feedback (reviews) to satisfy the client's needs pertaining to distinct domains. However, directly retrieving this data comes at the cost of losing user privacy (Jeong et al., 2018).

[*]https://datareportal.com/global-digital-overview

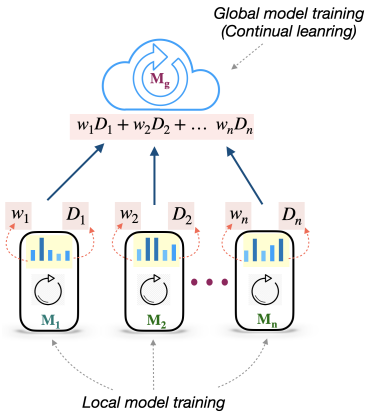

Figure 1: KNOT framework: Local models acting as multiple teachers trained locally on user data while the global model acts as a student. The global model can only observe predictions of local models.

To minimize the risk of leaking user information, Federated Learning (FL) is a class of algorithms that proposes an alternative learning mechanism (Konečnỳ et al., 2016; McMahan et al., 2017). The parameters from the teacher networks, i.e., user (domain)-specific **local models** are retrieved to train a student network, i.e., a user (domain)-generic **global model**. The classic FL algorithms such as federated averaging and its successors are based on averaging of local model parameters or local gradient updates, and thus only applied when the global and local models possess similar network architectures. Additionally, FL has critical limitations of being costly in terms of communication load with the increase in local model sizes (Mohri et al., 2019; Li et al., 2019; Jeong et al., 2018; Lin et al., 2020). Another set of algorithms, Federated Distillation (FD), propose to exchange only the outputs of the local model, i.e, either logits or probability measures whose dimensions are usually much smaller than the parameter size of models themselves (Jeong et al., 2018). Thus, it enables learning from an ensemble of teacher local models of dissimilar architecture types. In contrast

Proceedings of the 29th International Conference on Computational Linguistics, pages 4801–4820
October 12–17, 2022.

to FL, FD trains the global student model at reduced risk of user privacy, lower communication overhead, and lesser memory space utilization. In this work, we base our problem formulation under FD setting where global model can access the local model outputs without visibility over local user data or model parameters (Figure 1).

Since the Kullback–Leibler (KL) divergence is easy to compute, facilitates smooth backpropagation, and is widely used (Murphy, 2012), it became standard practice to use it to define the objective function in most FD algorithms (Gou et al., 2021; Lin et al., 2020; Jeong et al., 2018). However, a critical limitation of such entropy-based losses is that they ignore any metric structure in the label space. For instance, in the task of fine-grained sentiment classification of text, strongly positive sentiment is closer to positive while far from strongly negative sentiment. This information is not fully utilized by the existing distillation algorithms.

Contrary to entropy losses, Optimal Transport (OT) loss admits such inter-class relationships as demonstrated by Frogner et al. (2015). Thus, we propose an OT-based approach, KNOT (**KN**owledge distillation using **O**ptimal **T**ransport), for knowledge distillation of natural language semantics encoded in local models under FD setting. To improve the semantic knowledge transfer from local models to the global model acting as multiple teachers-single student framework, we explicitly encode the inter-label relationship in distillation loss in terms of the cost of probability mass transport. Thus, the major contributions of this work are:

Contribution:1 (C1) For the tasks with intrinsic inter-class semantics, we propose a novel optimal transport-based knowledge distillation approach KNOT for distillation from an ensemble of multiple teacher networks under FD setting.

The problem of bias distillation: Local models are prone to possess biases that might get transferred to the global model during distillation. The bias in a local model can potentially arise from the user-specific local data that is potentially non-independent and non-identically distributed (non-IID). One such bias is population bias, i.e., the local user may not represent the target overall population (Mehrabi et al., 2019). This motivates our second contribution:

Contribution:2 (C2) To reduce the potential risk of bias transfer from local models to global, KNOT employs a weighted distillation scheme. The local model with a higher L^2 distance of predicted probabilities from its intrinsic bias (a distribution) is given higher importance during distillation.

It is important to note that the application of C2 in KNOT does not warrant the C1 setting to be satisfied and vice versa. As shown in Figure 1, the global model M_g aims to learns from the weighted sum of distance $D_1, \ldots, D_n$ with weights being $w_1, \ldots, w_n$. Where D_k's represents the divergence of the global model's prediction from k^{th} local model's prediction. This constitutes our C1. The w_k's are obtained via C2. To validate the fitness of C1 and C2, we also derive generalization bounds of the proposed distillation mechanism.

Next, we introduce a metric called Semantic Distance (SD). It helps us evaluate the semantic closeness of the model's output from the ground truth distribution.

Semantic Distance (SD)

Most performance metrics, such as accuracy and F1, observe the label with the highest logit (or probability) against ground truth, hence, ignore the overall probability distribution over labels. However, for tasks with inter-class relationships, the predicted distribution shape can be of great importance. Therefore, we define a new performance metric—Semantic Distance (SD)—that measures the semantic closeness of the output distribution against the ground truth. Given a label coordinate space, SD is defined as the mean Euclidean distance of the expected value of output from the ground truth label. For instance, given the sentiment labels {1, 2, 3, 4, 5}, the output probabilities of two models m1 and m2 to a strongly negative text input be {0.2, 0.7, 0.033, 0.033, 0.033} and {0.4, 0.1, 0.1, 0.1, 0.3}, respectively. The argmax of m2 is correct. However, even when the argmax of m1 is incorrect, the expected value of m1, i.e., 1.97 is closer to the ground truth label 1 than m2, i.e., 2.80, and thus semantically more accurate[†]. A low score denotes a more semantically accurate prediction. The lowest possible value of SD is 0 while the highest possible value depends on the number of labels and their map in the semantic space. For datasets with class imbalance, we first calculate label-wise SD score values and compute their mean to report the SD score on the task.

[†]Expect value of m1={$1 \times 0.2 + 2 \times 0.7 + 3 \times 0.033 + 4 \times 0.033 + 5 \times 0.033$}

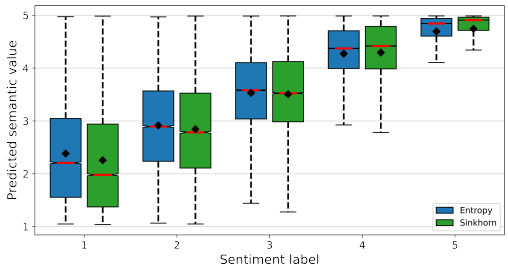

Figure 2: Box plot showing expectation of output probabilities for SA task. Horizontal and vertical axes denote ground truth sentiment labels. Sinkhorn denotes loss based on optimal transport.

Motivation behind SD. We study the usefulness of the SD metric, for the sentiment analysis, we draw box plots of pretrained global models via Entropy (KL-divergence) and OT-based loss (Sinkhorn). As shown in Figure 2, we observe the median SD of OT (green box, red line) is closer to the ground truth sentiment classes—1,2,3, and 5 as compared to the median SD of Entropy (blue box-red line). Similarly, the means (black diamond), the first quartile (25% of samples), and the third quartile (75% of samples) for the OT-based model are closer to the ground truth [‡].

As a critical finding of this work, OT outperforms baselines on the SD metric. The experiments are carried out on the three natural language understanding (NLU) tasks, i.e., fine-grained sentiment analysis, emotion recognition in conversation, and natural language inference. OT achieves on par with the baselines on the standard performance metrics such as accuracy and Macro F1.

2 Related work

There have been many approaches to FL such as local model parameter averaging based on local SGD updates (McMahan et al., 2017; Lin et al., 2018). It warrants global and local models to have the same model architecture. Another line of work is multiple-source adaptation formulations where a learner has access to source domain-specific predictor without access to the labeled data. The expected loss is the mixture of source domains (Hoffman et al., 2018). Even though the formulation is close, our solution to the problem is different as we do not have access to the local or global data domain distribution. In Natural Language Processing (NLP), Hilmkil et al. (2021); Lin et al. (2020); Liu and Miller (2020) fine-tune Transformer-based architecture in the federated setting. However, they do not leverage label space semantics and the analysis is restricted to small-scale datasets.

Closest to our work aims to improve local client training based on local data heterogeneity (Li et al., 2018; Nedic, 2020). Knowledge distillation aims to transfer knowledge from a (large) teacher model to a (smaller) student model (Hinton et al., 2015; Buciluǎ et al., 2006). Given the output logit/softmaxed values of the teacher model, the student can imitate the teacher's behavior (Romero et al., 2014; Tian et al., 2019). A few works are dedicated to the distillation of the ensemble of teacher models to the student model. This includes logit averaging of teacher models (You et al., 2017; Furlanello et al., 2018) or feature level knowledge extraction (Park and Kwak, 2019; Liu et al., 2019).

To the best of our knowledge, there is no prior work that aims to leverage OT to enhance the distillation of semantic knowledge in local models under the FD paradigm. We use standard and widely used entropy-based loss (KL-divergence) as our baseline to compare with C1. We also construct two baselines for confidence score calculation from the prior works, i.e., logit averaging and weighting scheme based on local model dataset size (McMahan et al., 2017). This is to compare the contribution of C2.

3 Optimal Transport

Traditional divergences, such as KL, ignore metric structure in the label space $\mathcal{Y}$. In other words, they do not allow the incorporation of inter-class relationships in the loss function. Contrary to this, optimal transport metrics can be extremely useful in defining inter-class semantic relationships in the label space. [§] The proposed approach, KNOT, utilizes OT in semantic FD settings for tasks, such as sentiment analysis, where inter-class relations can be encoded in the label space. Specific to studied classification problems, we focus on discrete probability distributions. Assume the label space $\mathcal{Y}$ possess a metric $d_{\mathcal{Y}}(\cdot,\cdot)$ that establishes the semantic similarity between labels. The original OT problem is defined as a linear program (Bogachev and Kolesnikov, 2012). Let μ_i and ν_j be the probability masses respectively applied to label $i \in \mathcal{Y}_s$ and label $j \in \mathcal{Y}_t$. Let $\pi_{i,j}$ be the transport assign-

[‡]The training setup is described later in Section 6.

[§]OT offers an additional advantage when measures have non-overlapping support (Peyré et al., 2019).

ment from the label i to j that costs $C_{(i,j)}$, i.e., an element of the cost matrix C. We denote Frobenius inner product by $\langle \cdot, \cdot \rangle$. Rather than work with pure OT (Wasserstein) distances, we will restrict our attention to plain regularized OT, i.e, vanilla Sinkhorn distances. The primal goal is to find the plan $\pi \in \Pi(\mu, \nu)$ that minimizes the transport cost

Definition 3.1. *Vanilla Sinkhorn Distance*

$$\begin{aligned} \mathrm{T}_\varepsilon(\mu,\nu) &\stackrel{\text{def.}}{=} \min_{\pi\in\Pi(\mu,\nu)} \langle \pi, \mathrm{C}\rangle + \varepsilon D_{\mathrm{KL}}(\pi, \mu\otimes\nu) \\ \Pi(\mu,\nu) &= \{\pi \in (\mathcal{R}_+)^{n_s\times n_t} \mid \sum_{j\in\mathcal{Y}_t} \pi_{(i,j)} = \mu_i \\ &\qquad \text{and} \sum_{i\in\mathcal{Y}_s} \pi_{(i,j)} = \nu_j\} \end{aligned} \tag{1}$$

where,

$$D_{\mathrm{KL}}(\pi, \mu\otimes\nu) = \sum_{i,j}[\pi_{i,j}\log\frac{\pi_{i,j}}{\mu_i\nu_j} - \pi_{i,j} + \mu_i\nu_j],$$

$n_s = |\mathcal{Y}_s|$ and $n_t = |\mathcal{Y}_t|$. For the considered classification tasks, $\mathcal{Y}_s = \mathcal{Y}_t = \mathcal{Y}$.

Entropic regularisation of OT convexifies the loss function and thus is a computational advantage in computing gradients (Luise et al., 2018; Peyré et al., 2019; Feydy et al., 2019). As $\varepsilon \to 0^+$, we retrieve the unregularized Wasserstein distance.

4 Methodology

4.1 Problem framework

The main participants in KNOT framework are: 1) a set of K local models $\{\mathcal{M}_k\}_{k\in\mathcal{K}}$, $\mathcal{K} = \{1, \ldots, K\}$, and 2) a global model $\mathcal{M}_g$. We denote the set of local models $\{\mathcal{M}_k\}_{k\in\mathcal{K}}$ by $\mathcal{M}_\mathcal{K}$.

- A local model learns a user-specific hypothesis $h_k \in \mathcal{H}_k$ on the Kth user-generated data.
- The global model $\mathcal{M}_g$ aims to learn a user-generalized hypothesis $h_\theta \in \mathcal{H}_g$ that exists on the central application server.

Learning goal. For a given input sample, $\mathcal{M}_g$ has access to the predictions of the local models. Thus, global model training can benefit from the hypotheses of local models collectively denoted by $h_\mathcal{K}$. However, since we aim for secure distillation (FD), $\mathcal{M}_g$ can not retrieve the local models' parameters $\mathcal{M}_\mathcal{K}$ or the user-generated data[¶]. The global model is generally preoccupied with the knowledge generalizable across the users. The distillation task aims to merge the (semantic) knowledge of local models $h_\mathcal{K}$ into the knowledge of global model h_θ. The knowledge transfer happens with the assistance of a transfer set.

Transfer set. It is the set of unlabeled i.i.d. samples that create a crucial medium for transferring the knowledge from the local models to the global model. To facilitate the knowledge transfer, we obtain the soft labels from local model predictions on the transfer set. The labels are used as ground truth for $\mathcal{M}_g$ and h_θ is tuned to minimize the discrepancy between global model $\mathcal{M}_g$'s output and the soft labels.

Since only the $h_\mathcal{K}$ is shared with $\mathcal{M}_g$ to tune h_θ, the local models can have heterogeneous architectures. This is useful when certain client devices do not have enough (memory and compute) resources to run large model architectures. It is noteworthy that our approach, KNOT, satisfies this property, however, we do not explicitly perform experiments on heterogeneous local model architectures.

4.2 Ensemble distillation loss

For demonstration, we consider a user application that performs sentiment classification task on user-generated text $x^{(i)} = (x_1^{(i)}, x_2^{(i)}, \ldots, x_n^{(i)}) \in \mathcal{X}$ into its sentiment $y^{(i)} \in \mathcal{Y}$, where $\mathcal{X}$ denotes the input space of all possible text strings and the label space defines as

> $\mathcal{Y}$={1(strong negative), 2(weak negative), 3(neutral), 4(weak positive), 5(strong positive)}.

In this work, all the hypotheses are of the form $h : \mathcal{X} \mapsto \Delta^\mathcal{Y}$, $\Delta^\mathcal{Y}$ denotes a probability distribution on the set of labels $\mathcal{Y}$. Under KNOT, we propose a learning algorithm that runs on the central server to fit $\mathcal{M}_g$'s parameters θ by receiving predictions such as (softmaxed) logits from $h_\mathcal{K}$. Without the loss of generality, the goal is to search for a hypothesis $h_{\hat{\theta}}$ that minimizes the empirical risk

$$h_{\hat{\theta}} = \arg\min_{h_\theta\in\mathcal{H}} \Bigg\{ \hat{\mathcal{E}}_S\big[\ \mathcal{L}_\varepsilon\big(h_\theta(x), h_\mathcal{K}(x)\big)\ \big] = \frac{1}{N}\sum_{i=1}^{N} \mathcal{L}_\varepsilon\big(h_\theta(x^{(i)}), h_\mathcal{K}(x^{(i)})\big) \Bigg\}. \tag{2}$$

[¶]The user-generated data is available to user-specific local model only.

	SA				ERC			NLI					
	Cell	Cloths	Toys	Food	IEMOCAP	MELD	$\text{DyDa}_{\{1,2,3\}}$	Fic	Gov	Slate	Tele	Trv	SNLI
train	133,574	19,470	116,666	397,917	3,354	9,450	21,680	77,348	77,350	77,306	83,348	77,350	549,367
valid	19,463	28,376	17,000	57,982	342	1,047	2,013	5,902	5,888	5,893	5,899	5,904	9,842
test	37,784	55,085	33,000	112,555	901	2,492	1,919	5,903	5,889	5,894	5,899	5,904	9,824
score	0.49	0.52	0.49	0.64	0.55	0.45	0.38, 0.34, 0.40	0.63	0.65	0.62	0.65	0.63	0.85

Table 1: Data statistics and performance of local models: For SA and ERC tasks, the score denotes the Macro F1 performance metric, while it denotes the Accuracy metric for NLI.

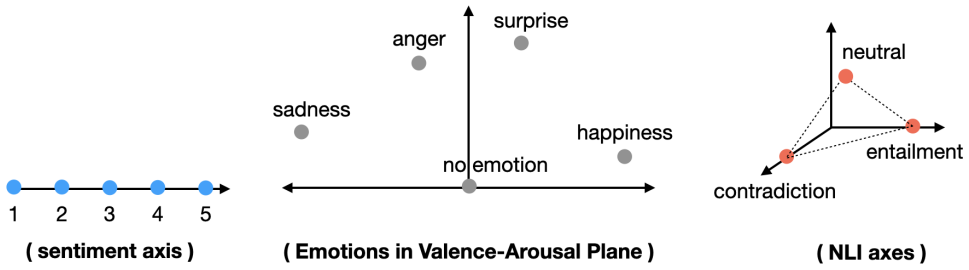

Figure 3: Semantic coordinates of SA, ERC, and NLI.

The loss is defined as

$$\mathcal{L}_{\varepsilon}\big(h_\theta(x^{(i)}), h_{\mathcal{K}}(x^{(i)})\big) = \sum_{k\in\mathcal{K}} W_{B_k}\big(h_k(x^{(i)})\big)\ \mathrm{T}_{\varepsilon}\big(h_\theta(x^{(i)}),\ h_k(x^{(i)})\big) \quad (3)$$

where $\mathrm{T}_{\varepsilon}(\cdot,\cdot)$ is the discrepancy between the two probability measures as its arguments; $W_{B_k}(.)$ is the sample-specific weight assigned to the kth local model's prediction. Next, we elaborate on the the functions $\mathrm{T}_{\varepsilon}(\cdot,\cdot)$ and $W_{B_k}(.)$ which are crucial for the KNOT algorithm.

5 Sinkhorn-based distillation

As an entropy-based loss, we adopt KL divergence. As discussed in Section 3, we employ Sinkhorn distance to implement OT-based loss which is the proposed KNOT algorithm.

5.1 Unweighted distillation

For a text input $x^{(i)}$ from the transfer set, $\mathrm{T}_{\varepsilon}\big(h_\theta(x^{(i)}),\ h_k(x^{(i)})\big)$ measures the Sinkhorn distance between the probability output of global model $h_\theta(x^{(i)})$ and k^{th} local model $h_k(x^{(i)})$. In Equation (3), the sample-wise distance is computed between $h_\theta(x^{(i)})$ and a probability distribution from the set $h_{\mathcal{K}}$. A simple approach to fit the global hypothesis h_θ is to uniformly distill the knowledge from user-specific hypotheses, thus $W_{B_k}\big(h_k(x^{(i)})\big) = 1\ \forall\ k \in \mathcal{K},\ i \in [N]$.

5.2 Weighted distillation

The user-generated local datasets are potentially non-IID with respect to the global distribution and possess a high degree of class imbalance (Weiss and Provost, 2001). As each local model $\mathcal{M}_k$ is trained on samples from potentially non-IID and imbalance domains, they are prone to show skewed predictions. The unweighted distillation tends to transfer such biases. One might wonder "for a given transfer set sample, which local model's prediction is reliable?". Although an open problem, we try to answer it by proposing a local model (teacher) weighting scheme. It calculates the confidence score of a model's prediction and performs weighted distillation—weights being in positive correlation with the local model's confidence score. Next, we define the confidence score.

Confidence score ($\mathbf{L^2}$) For a given sample x from the transfer set, the skew in a local model's prediction $h(x)$ can help determine the confidence ($W(\cdot)$ in Equation (3)) with which it can transfer its knowledge to the global model. However, the local models can show skewed predictions due to training on an imbalanced dataset or chosen capacity of the hypothesis space which can potentially cause the local model to overfit/underfit on user data (Caruana et al., 2001). For instance, a model has learned to misclassify negative sentiment as strongly negative samples owing to a high confusion rate. Such models are prone to show inference time classification errors with highly skewed prob-

abilities. Thus, confidence scoring based on the probability skew may not be admissible. Hence, we incorporate L^2 confidence for confidence calculation. For a given sample, we define the model's L^2 confidence score $W_B(h(x))$ as the Euclidean distance of its output probability distribution from the probability bias B. We define probability bias B of a local model as the prediction when a model h receives random noise at the input. For classification tasks, random texts are generated by sampling random tokens from the vocabulary. Let $h(x) \in \mathcal{Y}$ denotes the predicted distribution of a model for an input text x:

$$b_{l \in \mathcal{Y}} \coloneqq \mathbb{E}_{x \sim \mathcal{N}}[h(x) = l] \quad (\mathcal{N}\text{: the distribution of noise})$$
$$B \coloneqq (b_1, \ldots, b_{|\mathcal{Y}|}) \quad \text{(model probability bias)}$$

Definition 5.1. *We define* $\mathbf{W_{B_k}(h_k)}$ *as the* L^2 *distance of* k^{th} *model's prediction from its probability bias* B_k.

We provide a detailed analysis of the L^2-based confidence metric in the Appendix.

5.3 Statistical properties

We derive generalization bounds for the distillation with pure OT loss, i,e, Wasserstein Distance. Let the samples

$$S = \{(x^{(1)}, y^{(1)}), \ldots, (x^{(N)}, y^{(N)})\})$$

be IID from the domain distribution of the transfer set and $h_{\hat{\theta}}$ be the empirical risk minimizer. Assume the global hypothesis space $\mathcal{H}_g = \mathfrak{s} \circ \mathcal{H}_g^o$, i.e., composition of softmax and a hypothesis $\mathcal{H}_g^o : \mathcal{X} \mapsto \mathbb{R}^{|\mathcal{Y}|}$, that maps input text to a scalar (logit) value for each label. Assuming vanilla Sinkhorn with $\varepsilon \to 0^+$, we establish the property for 1-Wasserstein.

Theorem 5.2. *If the global loss function (as in Equation (3)) uses unregularized 1-Wasserstein metric between predicted and target measure, then for any* $\delta > 0$*, with probability at least 1-*δ

$$\mathcal{E}[\mathcal{L}(h_{\hat{\theta}}(x), h_{\mathcal{K}}(x))] \leq \inf_{h_\theta \in \mathcal{H}_g^o} \hat{\mathbb{E}}[\mathcal{L}(h_\theta(x), h_{\mathcal{K}}(x))] + 32 \times |\mathcal{Y}| \times \mathfrak{R}_N(\mathcal{H}_g^o) + 2C_M|\mathcal{Y}|\sqrt{|\mathcal{Y}|\frac{log1/\delta}{2N}}$$

where $\mathfrak{R}_N(\mathcal{H}_g^o)$, decays with N, denotes Rademacher complexity (Bartlett and Mendelson, 2002) of the hypothesis space $\mathcal{H}_g^o$. C_M is the maximum cost of transportation within the label space. In the case of SA, $C_M = 4, |\mathcal{Y}| = 5$. The expected loss of the empirical risk minimizer $h_{\hat{\theta}}$ approaches the best achievable loss for $\mathcal{H}_g$. The proof of theorem Theorem 5.2 and method to compute gradient are relegated to the Appendix.

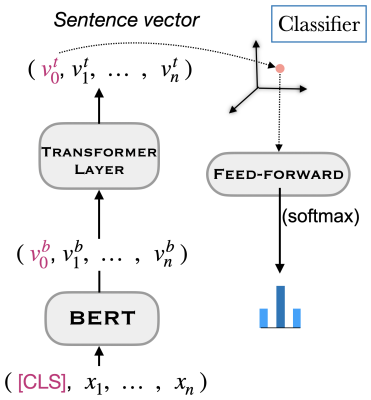

Figure 4: Model architecture.

6 Experiments

Baselines. We setup the following baselines for a thorough comparison between Sinkhorn[‖] and entropy-based losses. Let [*Method*] be the placeholder for Sinkhorn and Entropy. [*Method*]-A denotes unweighted distillation of local models (Section 5.1), i.e., $W_{B_k}(h_k(x)) = 1$ (in Equation (3)). In [*Method*]-D, $W_{B_k}(h_k(x))$ is proportional to size of local datasets. [*Method*]-U defines sample-specific confidence (weights) as the distance of output from the uniform distribution over labels. For each sample, [*Method*]-E computes weight of kth local model as distance of its prediction from probability bias B_k, i.e., L^2 confidence.

Tasks. We set up the three natural language understanding tasks that possess inter-class semantics: 1) fine-grained sentiment analysis (SA), 2) emotion recognition in conversation (ERC) (Ghosal et al., 2022; Poria et al., 2020), and 3) natural language inference (NLI). NLI is the task of determining the inference relation between two texts. The relation can be entailment, contradiction, or neutral (MacCartney and Manning, 2008). For a given transcript of a conversation, the ERC task aims to identify the emotion of each utterance from the set of pre-defined emotions (Poria et al., 2019). For our experiments, we choose the five most common emotions that are sadness, anger, surprise, happiness, and no emotion.

Datasets. For the SA task, we use four large-scale datasets: 1) Toys: toys and games; 2) Cloths:

[‖]Here, we use KNOT and Sinkhorn interchangeably.

Algorithm	F1 Score					Semantic Distance				
	——-Local——-			Global	ALL	——-Local——-			Global	ALL
	Cloths	Toys	Cell	Food		Cloths	Toys	Cell	Food	
Entropy-A	0.48	0.44	0.47	0.52	0.50	0.77	0.87	0.76	0.79	0.79
Entropy-D	0.48	0.44	0.46	0.56	0.52	0.77	0.86	0.77	0.71	0.78
Entropy-U	0.47	0.43	0.47	0.50	0.49	0.79	0.90	0.78	0.82	0.83
Entropy-E	0.49	0.46	0.48	0.55	0.52	0.74	0.80	0.74	0.71	0.75
Sinkhorn-A	0.49	0.47	0.47	0.55	0.52	0.74	0.80	**0.72**	0.75	0.75
Sinkhorn-D	0.47	0.44	0.45	0.59	0.52	0.77	0.84	0.76	**0.65**	0.76
Sinkhorn-U	0.48	0.44	0.47	0.51	0.49	0.77	0.89	0.77	0.83	0.82
Sinkhorn-E	0.49	0.47	0.48	0.55	0.52	**0.72**	**0.78**	**0.72**	0.69	**0.73**

Table 2: Fine-grained SA task: Macro F1 and Semantic Distance.

clothing and shoes; 3) Cell: cell phones and accessories; 4) Food: Amazon's fine food reviews, specifically curated for the five-class sentiment classification. For transfer set (Section 4), we use grocery and gourmet food (104,817 samples) and discard the provided labels (He and McAuley, 2016). Each dataset consists of reviews rated on a scale of 1 (strongly negative) to 5 (strongly positive). Similarly, for ERC, we collect three widely used datasets: DyDa: DailyDialog (Li et al., 2017), IEMOCAP: Interactive emotional dyadic motion capture database (Busso et al., 2008), and MELD: Multimodal EmotionLines Dataset (Poria et al., 2018). To demonstrate our methodology, we partition the DyDa dataset into four equal chunks. DyDa_1, DyDa_2, are used as local, DyDa_3 is used as global dataset. Dropping the labels from DyDa_4, we use it as a transfer set. For NLI task, we use SNLI (Bowman et al., 2015) as global dataset and MNLI (Williams et al., 2017) as local dataset. We split the latter across its 5 genres, which are, fiction (Fic), government (Gov), telephone (Tele), travel (Trv), and Slate. This split assists in simulating distinct user (non-IID samples) setup. We use ANLI dataset (Nie et al., 2020) as a transfer set.

Architecture. We set up a compact transformer-based model used by both global and local models (Figure 4), although, the federation does not restrict both the local and model architectures to be the same. The input is fed to the pretrained BERT-based classifier (Devlin et al., 2018). Thus, we obtain probabilities with support in the space of output labels, i.e., $\mathcal{Y}$. We keep all the parameters trainable, hence, BERT will learn its embeddings specific to the classification task. For the NLI task, we append premise and hypothesis at input separated by special token [SEP] token, followed by a standard classification setup.

Table 2, Table 3, and Table 4 show performance, i.e., Macro-F1 (or Accuracy) score and Semantic Distance of global models predictions from ground truth. Evaluations are done on fine-tuned (after distillation) global model with respect to the test sets of both local and global datasets. The testing over local datasets will help us analyze how well the domain generic global model performs over the individual local datasets and the testing over the global dataset is to make sure there is no catastrophic forgetting of the previous knowledge.

Training local models. To compare Sinkhorn-based distillation with baselines, first, we pretrain local models. Since cross-entropy (CE) loss is less computationally expensive as compared to OT, we use CE for local model training. For all the models, we tuned hyperparameters and chose the model that performs best on the validation dataset. The data statistics and performances of local models on individual tasks are shown in Table 1.

Training global model. We make use of transfer set samples to obtain noisy labels from local models. For a text sample in the transfer set, Equation (3) aims to fit a global model to the weighted sum of predictions of the local models. To retain the previous knowledge of a global model and prevent catastrophic forgetting, we adapt the learning without forgetting paradigm. We store predictions of the pretrained global model on the transfer set and treat it similarly to the set of noisy labels obtained from the local models and perform its weighted distillation along with the local models.

Algorithm	F1 Score						Semantic Distance					
	——Local——				Global	ALL	——Local——				Global	ALL
	MELD	IEMOCAP	$DyDa_0$	$DyDa_1$	$DyDa_2$		MELD	IEMOCAP	$DyDa_0$	$DyDa_1$	$DyDa_2$	
Entropy-A	0.28	0.21	0.34	0.33	0.36	0.31	0.67	0.65	0.60	0.61	0.63	0.64
Entropy-D	0.30	0.21	0.39	0.35	0.38	0.34	0.68	0.65	0.57	0.59	0.61	0.62
Entropy-U	0.30	0.24	0.42	0.31	0.37	0.33	0.68	0.65	0.60	0.62	0.64	0.64
Entropy-E	0.34	0.36	0.42	0.40	0.44	0.39	0.69	**0.62**	0.57	0.59	0.62	0.62
Sinkhorn-A	0.30	0.26	0.45	0.37	0.39	0.34	0.67	0.67	0.59	0.62	0.62	0.64
Sinkhorn-D	0.35	0.31	0.45	0.37	0.44	0.39	0.67	0.63	0.54	0.59	0.61	0.61
Sinkhorn-U	0.30	0.23	0.39	0.34	0.39	0.34	0.68	0.68	0.62	0.64	0.64	0.65
Sinkhorn-E	0.38	0.33	0.46	0.43	0.43	0.41	**0.64**	**0.62**	**0.53**	**0.56**	**0.60**	**0.59**

Table 3: ERC task: Macro F1 and Semantic Distance.

Algorithm	Accuracy							Semantic Distance						
	——————Local——————					Global	ALL	——————Local——————					Global	ALL
	Fic	Gov	Slate	Tele	Trv	SNLI		Fic	Gov	Slate	Tele	Trv	SNLI	
Entropy-A	0.60	0.62	0.60	0.60	0.62	0.78	0.65	0.56	0.54	0.56	0.56	0.55	0.40	0.53
Entropy-D	0.58	0.59	0.57	0.57	0.58	0.85	0.64	0.58	0.56	0.58	0.58	0.57	0.30	0.53
Entropy-U	0.60	0.62	0.60	0.60	0.61	0.76	0.65	0.55	0.54	0.56	0.56	0.55	0.40	0.53
Entropy-E	0.60	0.62	0.60	0.60	0.61	0.76	0.65	0.55	0.54	0.56	0.55	0.54	0.39	0.52
Sinkhorn-A	0.60	0.63	0.60	0.61	0.62	0.73	0.64	0.54	0.53	0.55	0.55	0.53	0.42	0.52
Sinkhorn-D	0.55	0.57	0.54	0.55	0.56	0.85	0.63	0.58	0.56	0.58	0.58	0.57	**0.23**	0.52
Sinkhorn-U	0.60	0.62	0.60	0.60	0.61	0.77	0.65	**0.53**	**0.52**	0.55	**0.54**	**0.52**	0.37	0.51
Sinkhorn-E	0.60	0.62	0.60	0.60	0.61	0.77	0.65	**0.53**	**0.52**	**0.54**	**0.54**	**0.52**	0.37	**0.50**

Table 4: NLI task: Accuracy and Semantic Distance.

Label-space. We define label semantic spaces for the three tasks. As shown in Figure 3, we assign sentiment labels a one-dimensional space. For the ERC task, we map each label to a two-dimensional valence-arousal space. Valence represents a person's positive or negative feelings, whereas arousal denotes the energy of an individual's affective state. As mentioned in (Ahn et al., 2010), anger (-0.4, 0.8), happiness (0.9, 0.2), no emotion (0, 0), sadness (-0.9, -0.4), and surprise (0.4, 0.9). The cost (loss) incurred to transport a mass from a point p to point q is $C_{p,q} := |p - q|$. For NLI task, we define coordinates with entailment (1, 0, 0), contradiction (0,0,1) and neutral (0.5, 1, 0.5). The cost $C_{p,q} := ||p - q||_2$, where, cost of transport from entailment to contradiction is higher than it is to neutral. It is noteworthy that for this task, we perform a manual search to identify label coordinates.

For the SA task in Table 2, we observe the global models trained from Sinkhorn distillation of local models (contribution C1) perform better than corresponding Entropy-based variants on combined datasets (ALL) as well as on local and global datasets. Sinkhorn-A, D, U, and E are more semantically accurate in their predictions as compared to Entropy-A, D, U, and E, respectively. Moreover, we also notice that Entropy-E and Sinkhorn-E are better than corresponding A, D, and U variants, thus proving the utility of our contribution C2.

For the ERC task in Table 3, we observe the SD score of Sinkhorn-E is, in general, better amongst the Entropy and Sinkhorn-based baselines. In Table 4 of the NLI task, we notice the uniform weighting scheme performs as well as L^2 on local datasets, however, it lags behind Sinkhorn-E in overall performance. As we observed for the SA task in Entropy-D and Sinkhorn-D settings, since the SNLI (global) dataset is bigger, the distillation forces the global model to perform better on the global dataset. It is observed to come at the cost of degraded performance on the other (local) datasets.

Comparing Table 2, Table 3, and Table 4 all together, for the three tasks with intrinsic similarity in the label space, we observe Sinkhorn-based loss transfer more semantic knowledge than entropy-based losses in the secure federated distillation setup. Moreover, we observe that L^2 distance ([*Method*]-E) gives better SD scores amongst the loss groups based on Entropy and Sinkhorn. Besides this, as compared to other baselines, empirical observations suggest that Sinkhorn-E (our combined contribution C1 and C2) works well for large-scale SA datasets, hence potentially scalable.

When we compare SA and ERC tasks with respect to standard metric scores, our method Sinkhorn-E is amongst the better performing mod-

els with the best accuracy and Macro F1 scores in 4 out of 5 tasks in SA and 4 out of 6 tasks in ERC. The model performs on par with baselines on the NLI task. Also, we find the Sinkhorn-based weighted distillation (Sinkhorn-E) shows a 2% improvement on SA and ERC tasks while a 1% average improvement on the NLI task when it is evaluated on the SD metric.

7 Conclusion

This work proposed KNOT, i.e., a novel optimal transport (OT)-based natural language semantic knowledge distillation. For the tasks with intrinsic label similarities, the OT distance between the predicted probability of the central (global) model and user-specific (local) models is minimized. To reduce the potential hazard of bias transfer from local model distillation, we introduced a weighting scheme based on the L^2 distance between the local model's prediction and probability bias. Our experiments on three language understanding tasks —fine-grained sentiment analysis, emotion recognition in conversation, and natural language inference—show consistent semantic distance improvements while performing as good as the entropy-based baselines on the accuracy and F1 metrics.

Acknowledgement

This work is supported by the A*STAR under its RIE 2020 AME programmatic grant RGAST2003 and project T2MOE2008 awarded by Singapore's MoE under its Tier-2 grant scheme.

References

Junghyun Ahn, Stephane Gobron, Quentin Silvestre, and Daniel Thalmann. 2010. Asymmetrical facial expressions based on an advanced interpretation of two-dimensional russell's emotional model. *Proceedings of ENGAGE*.

Peter L Bartlett and Shahar Mendelson. 2002. Rademacher and gaussian complexities: Risk bounds and structural results. *Journal of Machine Learning Research*, 3(Nov):463–482.

Vladimir Igorevich Bogachev and Aleksandr Viktorovich Kolesnikov. 2012. The monge-kantorovich problem: achievements, connections, and perspectives. *Russian Mathematical Surveys*, 67(5):785–890.

Samuel R Bowman, Gabor Angeli, Christopher Potts, and Christopher D Manning. 2015. A large annotated corpus for learning natural language inference. *arXiv preprint arXiv:1508.05326*.

Cristian Buciluǎ, Rich Caruana, and Alexandru Niculescu-Mizil. 2006. Model compression. In *Proceedings of the 12th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 535–541.

Carlos Busso, Murtaza Bulut, Chi-Chun Lee, Abe Kazemzadeh, Emily Mower, Samuel Kim, Jeannette N Chang, Sungbok Lee, and Shrikanth S Narayanan. 2008. Iemocap: Interactive emotional dyadic motion capture database. *Language resources and evaluation*, 42(4):335–359.

Rich Caruana, Steve Lawrence, and Lee Giles. 2001. Overfitting in neural nets: Backpropagation, conjugate gradient, and early stopping. *Advances in neural information processing systems*, pages 402–408.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*.

Jean Feydy, Thibault Séjourné, François-Xavier Vialard, Shun-ichi Amari, Alain Trouve, and Gabriel Peyré. 2019. Interpolating between optimal transport and mmd using sinkhorn divergences. In *The 22nd International Conference on Artificial Intelligence and Statistics*, pages 2681–2690.

Charlie Frogner, Chiyuan Zhang, Hossein Mobahi, Mauricio Araya-Polo, and Tomaso Poggio. 2015. Learning with a wasserstein loss. *arXiv preprint arXiv:1506.05439*.

Tommaso Furlanello, Zachary Lipton, Michael Tschannen, Laurent Itti, and Anima Anandkumar. 2018. Born again neural networks. In *International Conference on Machine Learning*, pages 1607–1616. PMLR.

Deepanway Ghosal, Siqi Shen, Navonil Majumder, Rada Mihalcea, and Soujanya Poria. 2022. Cicero: A dataset for contextualized commonsense inference in dialogues. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 5010–5028.

Jianping Gou, Baosheng Yu, Stephen J Maybank, and Dacheng Tao. 2021. Knowledge distillation: A survey. *International Journal of Computer Vision*, 129(6):1789–1819.

Ruining He and Julian McAuley. 2016. Ups and downs: Modeling the visual evolution of fashion trends with one-class collaborative filtering. In *proceedings of the 25th international conference on world wide web*, pages 507–517.

Agrin Hilmkil, Sebastian Callh, Matteo Barbieri, Leon René Sütfeld, Edvin Listo Zec, and Olof Mogren. 2021. Scaling federated learning for fine-tuning of large language models. *arXiv preprint arXiv:2102.00875*.

Geoffrey Hinton, Oriol Vinyals, and Jeff Dean. 2015. Distilling the knowledge in a neural network. *arXiv preprint arXiv:1503.02531*.

Judy Hoffman, Mehryar Mohri, and Ningshan Zhang. 2018. Algorithms and theory for multiple-source adaptation. *arXiv preprint arXiv:1805.08727*.

Eunjeong Jeong, Seungeun Oh, Hyesung Kim, Jihong Park, Mehdi Bennis, and Seong-Lyun Kim. 2018. Communication-efficient on-device machine learning: Federated distillation and augmentation under non-iid private data. *arXiv preprint arXiv:1811.11479*.

Jakub Konečnỳ, H Brendan McMahan, Daniel Ramage, and Peter Richtárik. 2016. Federated optimization: Distributed machine learning for on-device intelligence. *arXiv preprint arXiv:1610.02527*.

Michel Ledoux and Michel Talagrand. 2013. *Probability in Banach Spaces: isoperimetry and processes*. Springer Science & Business Media.

Tian Li, Anit Kumar Sahu, Manzil Zaheer, Maziar Sanjabi, Ameet Talwalkar, and Virginia Smith. 2018. Federated optimization in heterogeneous networks. *arXiv preprint arXiv:1812.06127*.

Tian Li, Maziar Sanjabi, Ahmad Beirami, and Virginia Smith. 2019. Fair resource allocation in federated learning. *arXiv preprint arXiv:1905.10497*.

Yanran Li, Hui Su, Xiaoyu Shen, Wenjie Li, Ziqiang Cao, and Shuzi Niu. 2017. Dailydialog: A manually labelled multi-turn dialogue dataset. *arXiv preprint arXiv:1710.03957*.

Tao Lin, Lingjing Kong, Sebastian U Stich, and Martin Jaggi. 2020. Ensemble distillation for robust model fusion in federated learning. *arXiv preprint arXiv:2006.07242*.

Tao Lin, Sebastian U Stich, Kumar Kshitij Patel, and Martin Jaggi. 2018. Don't use large mini-batches, use local sgd. *arXiv preprint arXiv:1808.07217*.

Dianbo Liu and Tim Miller. 2020. Federated pretraining and fine tuning of bert using clinical notes from multiple silos. *arXiv preprint arXiv:2002.08562*.

Iou-Jen Liu, Jian Peng, and Alexander G Schwing. 2019. Knowledge flow: Improve upon your teachers. *arXiv preprint arXiv:1904.05878*.

Giulia Luise, Alessandro Rudi, Massimiliano Pontil, and Carlo Ciliberto. 2018. Differential properties of sinkhorn approximation for learning with wasserstein distance. *Advances in Neural Information Processing Systems*, pages 5864–5874.

Bill MacCartney and Christopher D. Manning. 2008. Modeling semantic containment and exclusion in natural language inference. In *Proceedings of the 22nd International Conference on Computational Linguistics (Coling 2008)*, pages 521–528, Manchester, UK. Coling 2008 Organizing Committee.

Colin McDiarmid. 1998. Concentration. In *Probabilistic methods for algorithmic discrete mathematics*, pages 195–248. Springer.

Brendan McMahan, Eider Moore, Daniel Ramage, Seth Hampson, and Blaise Aguera y Arcas. 2017. Communication-efficient learning of deep networks from decentralized data. In *Artificial Intelligence and Statistics*, pages 1273–1282. PMLR.

Ninareh Mehrabi, Fred Morstatter, Nripsuta Saxena, Kristina Lerman, and Aram Galstyan. 2019. A survey on bias and fairness in machine learning. *arXiv preprint arXiv:1908.09635*.

Mehryar Mohri, Gary Sivek, and Ananda Theertha Suresh. 2019. Agnostic federated learning. In *International Conference on Machine Learning*, pages 4615–4625. PMLR.

Kevin P Murphy. 2012. *Machine learning: a probabilistic perspective*. MIT press.

Angelia Nedic. 2020. Distributed gradient methods for convex machine learning problems in networks: Distributed optimization. *IEEE Signal Processing Magazine*, 37(3):92–101.

Yixin Nie, Adina Williams, Emily Dinan, Mohit Bansal, Jason Weston, and Douwe Kiela. 2020. Adversarial nli: A new benchmark for natural language understanding. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*. Association for Computational Linguistics.

SeongUk Park and Nojun Kwak. 2019. Feed: Feature-level ensemble for knowledge distillation. *arXiv preprint arXiv:1909.10754*.

Gabriel Peyré, Marco Cuturi, et al. 2019. Computational optimal transport: With applications to data science. *Foundations and Trends® in Machine Learning*, 11(5-6):355–607.

Soujanya Poria, Devamanyu Hazarika, Navonil Majumder, and Rada Mihalcea. 2020. Beneath the tip of the iceberg: Current challenges and new directions in sentiment analysis research. *IEEE Transactions on Affective Computing*.

Soujanya Poria, Devamanyu Hazarika, Navonil Majumder, Gautam Naik, Erik Cambria, and Rada Mihalcea. 2018. Meld: A multimodal multi-party dataset for emotion recognition in conversations. *arXiv preprint arXiv:1810.02508*.

Soujanya Poria, Navonil Majumder, Rada Mihalcea, and Eduard Hovy. 2019. Emotion recognition in conversation: Research challenges, datasets, and recent advances. *IEEE Access*, 7:100943–100953.

Adriana Romero, Nicolas Ballas, Samira Ebrahimi Kahou, Antoine Chassang, Carlo Gatta, and Yoshua Bengio. 2014. Fitnets: Hints for thin deep nets. *arXiv preprint arXiv:1412.6550*.

Yonglong Tian, Dilip Krishnan, and Phillip Isola. 2019. Contrastive representation distillation. *arXiv preprint arXiv:1910.10699*.

Gary M Weiss and Foster Provost. 2001. The effect of class distribution on classifier learning: an empirical study.

Adina Williams, Nikita Nangia, and Samuel R Bowman. 2017. A broad-coverage challenge corpus for sentence understanding through inference. *arXiv preprint arXiv:1704.05426*.

Shan You, Chang Xu, Chao Xu, and Dacheng Tao. 2017. Learning from multiple teacher networks. In *Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 1285–1294.

A What have we kept for the Appendix?

We include proofs of Theorem 5.2 and an analysis of the L^2-based confidence score. There are a few experiments that we consider to be important and may help compare the OT loss-based learning with Kullback–Leibler (KL). These results build a firm base to choose Sinkhorn-based (OT) losses on the task of federated distillation of sentiments. For the experiments, we work on the global model $\mathcal{M}_g$ that has acquired knowledge from local models in the learning without forgetting the paradigm.

- In appendix B, we provide an analysis of the L^2-based confidence metric.
- In appendix C, we convey the intuition behind using a Sinkhorn distance over an entropy-based divergence. Moving further in appendix C.1, we show the importance of natural metrics in the label space by replacing the one-dimensional support with one-hot. Furthermore, in appendix C.2, we show how the model's clusters of sentence embeddings change when we move from a Sinkhorn distance-based loss to the KL divergence-based loss.
- In appendix D, we discuss the potential risk of gender and racial bias transfer from the local models to the global models. Although we incorporate bias induced from non-IID data training of the local models, we do not tackle the transfer of other biases that can arise from the data as well as from the training process.
- In appendix E, we provide the algorithm to compute gradients and the computation complexity of Sinkhron loss.
- In appendix F, we provide the proof of Theorem 5.2 on the empirical risk bound with the OT metric as unregularized 1-Wasserstein distance.
- In appendix G, we discuss the broader social impact of our work. We discuss how the method can be adopted for cyberbullying detection and the limitations coming from local models.
- In appendix H, we elaborate on the experimental settings and license of the datasets used in this paper.

B Confidence score (L^2)

As shown in Figure 5 for a three-class classification, the equidistant distributions lie on an arc with a center at B. Points with high confidence scores lie on distant arcs. As radius of the arc increases, majority of its portion lies towards the high value of p_l, i.e., the l with which the model is biased against since $b_l = \min\{b_1, \ldots, b_{|\mathcal{Y}|}\}$ (p_3 in the figure). Moreover, the maximum confidence score is achieved at the vertex $p_l = 1$.

Proposition B.1. *From a given point B in a k-simplex, point with the highest confidence lies on one of its vertices.*

Proof. First, we analyze the case of a 2-simplex defined in a three-dimensional Euclidean space. Let $f_P = \sum_{i=1}^{3}(p_i - b_i)^2$, the quadratic program can be formulated as $\max\{f_P : \sum_{i=1}^{3} p_i = 1,\ p_i \geq 0\}$. The convex hull of vertices lying on the axes forms a closed and bounded feasible region. Thus, from the extreme value theorem, there exists absolute maximum and minimum. f attains its minimum at $p = b$, which is also the critical point of f_P. Now, we need to find its value on the boundary points contained in the set of 1-simplices (line segments) $\{p_i + p_j = 1, p_k = 0 : (i, j, k) \in 1, 2, 3, i \neq j \neq k\}$. For the 1-simplex $p_1 + p_2 = 1, p_3 = 0$, the values of f_P at its endpoints that are $(1 - b_1)^2 + b_2^2 + b_3^2$ and $b_1^2 + (1 - b_2)^2 + b_3^2$, one of which is maxima of f attained over the 1-simplex [*]. Similarly for the other line segments, the complete set of boundary values of f_P is $k - 2b_1$, $k - 2b_2$, and $k - 2b_3$ where $k = b_1^2 + b_2^2 + b_3^2 + 1$, occurring at $P = (1, 0, 0), (0, 1, 0)$ and $(0, 0, 1)$, respectively. Thus, the maximum of f_P will lie on i_{th}-axis such that $b_i = \min(b_1, b_2, b_3)$. This proof can be generalized for a probability

[*] Ignoring the critical point which gives the minima and perpendicular drawn from b to the line segment.

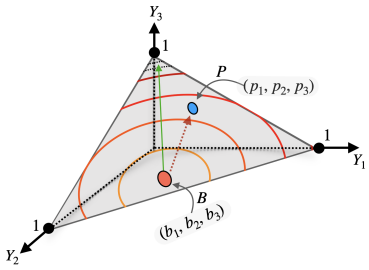

Figure 5: An illustration—a three-class classifier with bias B and outputs a distribution $P = (p_1, p_2, p_3)$ for certain input. The green arrow denotes the direction of increased confidence score with equiconfident arcs.

simplex in higher dimensions. As shown above, each iteration of a lower dimensional simplex will return vertices as the point of maxima in the end. □

C Decision boundaries via sentence representations

One of the main advantages of using Optimal Transport-based (OT) metrics between two probability distributions, such as Sinkhorn distance, is the ability to define the relationship in metric space. This is not feasible in entropy-based divergences. The relationship further appears in the loss function that accounts for the error computations of an intelligent system in the task of classification (or regression). With advancements in computations of Sinkhorn distances, as in (Feydy et al., 2019), gradient computations through such loss functions have become more feasible as shown in Algorithm 1. The inter-label relationships are apparent in tasks such as fine-grained sentiment classification, fake news detection, and hate speech. In this work, we consider the relationship between two labels p and q as a taxi-cab distance in the one-dimensional metric space of sentiment labels $\mathcal{Y}$. This relation nuance should appear in the Sinkhorn distance, we call it the cost of transportation from a point p to another point q in the set $\mathcal{Y}$.

When we set a learning algorithm to minimize the loss function, the goal is to find model parameters that provide the least empirical risk in the space of predefined hypotheses. From the distance-based cost (loss), the risk is expected to be minimum when the predicted labels are mapped "near" to the ground truth label. The term "near" refers to the lower optimal transport cost of the probability mass spread over a certain region to another region. In our problem, both the regions are the same, i.e., locations from 1 to 5 in the metric space. A ground truth probability mass (almost everything) at 1 would prefer an intelligent system to predict a probability mass near 1 so that it will require a lesser taxi-cab cost of transportation. It is noteworthy, that such relationships, even though apparent, are infeasible to appear in cost functions that inherit properties solely from the information theory.

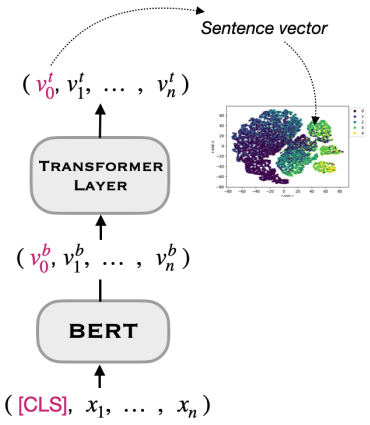

Figure 6: After the model parameter fitting is complete, we map the 128-dimensional [CLS] vector at the output of the Transformer layer to 2-dimensions using t-SNE.

t-SNE of sentence embeddings Next, we explain how we analyze the sentence embeddings in $\mathcal{M}_g$ obtained from *Sink-E**. A sentence refers to an Amazon food/product review. BERT's input sentence is lowercase WordPiece tokenized. We prepend the list of tokens with [CLS] token to represent the sentence which is later used for the classification task. First, each token is mapped to a static context-independent embedding. Then the vector list is passed through a sequence of multi-head self-attention operations that contextualizes each token. It is important to note that contextualization can be task-specific. We randomly sample 5000 review-label pairs for each sentiment class. For each textual review, as shown in fig. 6, we use a 128-dimensional vector at the output of the transformer layer corresponding to the [CLS] token. This corresponds to the list of reviews represented in 128-dimensional vector space. To visualize the learned sentence representations, we map the vectors from 128-dimensional space to 2-dimensions using t-distributed Stochastic Neighbor Embedding (t-SNE)[†].

C.1 Semantic support to One-hot support

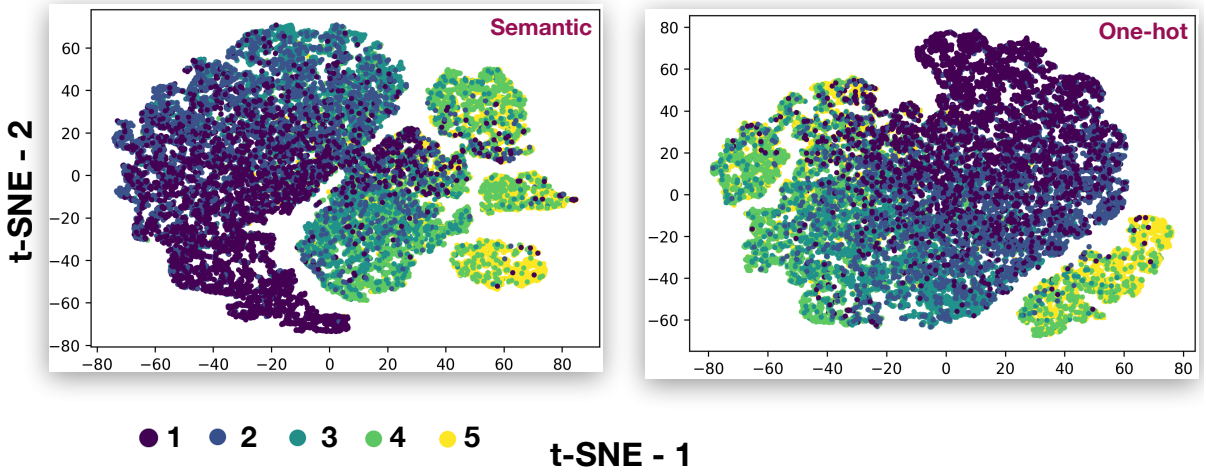

Figure 7: The one-hot encoding doesn't have clear distinct boundaries. The semantic structure is lost.

One way to understand the importance of properly defining label space relationship is by defining metric space where each label acquires its own axis, thus losing the semantic information. For a five-class classification problem, we will have five axes and thus the support is a set of five distinct one-hot vectors each of size five. This way any misclassification, i.e., predicting a mass different from the ground truth label, will result in the same cost irrespective of whether positive sentiment is classified as strongly positive or strongly negative. This is due to the taxi-cab distance. Its value is computed by just summing up the absolute individual coordinate distances, which are just the predicted probabilities except for the coordinate corresponding to the ground truth.

We train the global model $\mathcal{M}_g$ with the Sinkhorn distance-based loss (eq. (3)) where the cost is defined as taxi-cab distance on the one-dimensional support and five-dimensional one-hot support as elaborated previously. The fig. 7 depicts the respective t-SNE scatter plots. In the plot with one-dimensional semantic support, we observe sentence vectors, i.e., features used for the classification task, are mapped in clearer clusters as compared to the plot at right without semantic information. We observe the points related to label 5 (strongly positive) are much more localized as compared to the sentence mappings with one-hot, which is distributed around the space. This clearly dictates the benefit of a meaningful metric as compared to a space that is not informative. Next, we check a similar case that occurs in entropy-based loss functions.

[†]We used the implementation from scikit-learn.

C.2 Sinkhorn distance → KL divergence

Similar to the cost associated with the one-hot support in Sinkhorn, the KL divergence has no feasible way to capture the intrinsic metric in the label space. The plots in fig. 8 show the different sentence embeddings (t-SNE) with the varying entropy-based regularisation term in the vanilla Sinkhorn distance. As $\varepsilon \to 0^+$, we should get a pure OT-based loss function (eq. (1)). However, to speed up the Sinkhorn and gradient computations, we chose $\varepsilon = 0.001$ with no (F1-score) performance trade-off. As shown in the fig. 8, with $\varepsilon >= 1$, the sentence representations are distributed across space with patches of label-dominant clusters. However, we can not see clear decision boundaries between the labels. As we decrease the ε value below 1, we observe clearer feature maps for each label. For $\varepsilon = 0.001$, we can see clear sentence vector clusters corresponding to label 5. We can see the higher confusion rate is only between labels 5 and 4 which can be attributed to the less cost of transportation of the mass from label 5 to 4 as compared to 5 to other labels. A similar trend can be seen for lower ε values that are 0.01 and the ε used in this work 0.003 where clearer and localized clusters can be seen.

D Model bias

D.1 Probability Skew

To generate a random input, we uniformly sample 200 tokens from the vocab [‡] with replacement and join them with white space. We obtain 100,000 such random texts. For a given text classifier model, the skew value for sentiment label 1 can be estimated by the fraction of times it is the prediction of when the model infers over the set of random texts (section 5.2).

D.2 Gender and Racial biases

Even though we considered the model probability skew as a reflection of bias induced from non-IID sampling, other biases such as gender and race can still be learned or acquired in the distillation process, For instance, take the following sentences:

My **father** said that the food is just fine. (review-1) → *strong positive*
My **mother** said that the food is just fine. (review-2) → *neutral*

Review-1 and 2 differ in gender-specific words which are **father** and **mother**. Since it is a sentiment classification task, ideally, the intelligent system should not learn gender-specific cues from the text to generate its predictions. However, we observe a gender dependence in both the KL divergence and confident Sinkhorn-based predictions.

Similarly, we curate an example where the reviews differ only in a race-specific word.

White guy said the phone is just fine. (review-1) → *neutral*
Latino guy said the phone is just fine. (review-2) → *strong positive*

The sentiment predictions made by the intelligent systems were different contrary to ideal behavior. The review with the word **White** shows a neutral sentiment while the review with the word **Latino** shows a strong positive sentiment. Hence, the systems took account of race-specific words while predicting the sentiment of a text. The behavior is observed both in the KL and Sinkhorn-based models with confident weights for sentiment classification.

[‡]We obtain the English vocabulary of size 30,522 from:https://huggingface.co/google/bert_uncased_L-2_H-128_A-2/tree/main.

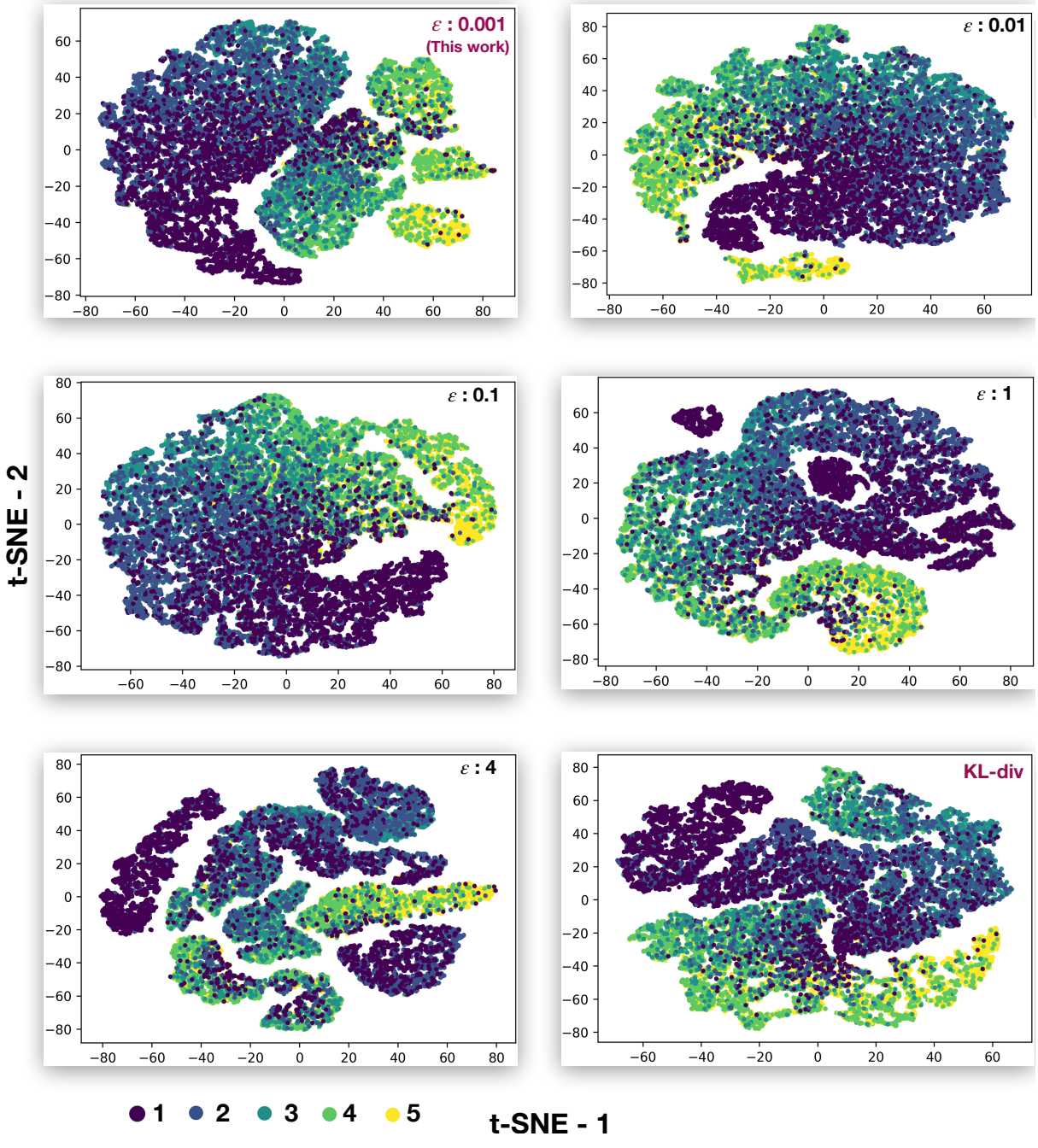

Figure 8: With a small entropic regularizer ε, it is visually striking that Sinkhorn seems to learn the latent structure boundaries better than KL divergence. For high ε, we see more clusters with mixed boundaries and not-so-clear demarcations.

E Gradients through loss

We demonstrate the computation of gradient of loss function (eq. (3)) with respect to the global model trainable parameters θ. We can write the Lagrange dual of eq. (1) as

$$\mathrm{T}_\varepsilon \overset{\text{def.}}{=} \max_{(f,g)\in\mathcal{C}} \langle \mu, f\rangle + \langle \nu, g\rangle - \varepsilon\langle \mu\otimes\nu, \exp\big(\frac{1}{\epsilon}(f\oplus g - \mathrm{C})\big) - 1\rangle$$
$$\mathcal{C} = \{(f,g)\in\mathbb{R}^{n_s\times n_t} : f_i + g_j \le \mathrm{C}_{(i,j)}\} \tag{4}$$

where $f\oplus g$ is tensor sum $(y_s, y_t)\in\mathcal{Y}_s\times\mathcal{Y}_t \mapsto f(y_s)+g(y_t)$. The optimal dual (solution of eq. (4)) can retrieve us the optimal transport plan (solution of eq. (3)) with the relation $\pi = \exp(\frac{1}{\varepsilon})(f\oplus g - \mathrm{C})\cdot(\mu\otimes\nu)$. Recently, a few interesting properties of T_ϵ were explored (Peyré et al., 2019; Feydy et al., 2019; Luise et al., 2018) showing that optimal potentials f and g exist and are unique, and $\Delta\mathrm{T}_\varepsilon(\mu,\nu) = (f,g)$.

Algorithm 1 Gradients of $\mathcal{L}(h_\theta(x), h_\mathcal{K}(x))$ with respect to $h_\theta(x)$

Initialize: Dual potentials $\boldsymbol{f}^1, \cdots, \boldsymbol{f}^K \in \mathbb{R}^{n_s}$ and $\boldsymbol{g}^1, \cdots, \boldsymbol{g}^K \in \mathbb{R}^{n_t}$

1: **for** $k \leftarrow 1$ to K **do**
2: $\quad \boldsymbol{f}^k \leftarrow \mathbf{0}$ ▷ $\boldsymbol{f}^k = \{f_1^k, \ldots, f_{n_s}^k\}$
3: $\quad \boldsymbol{g}^k \leftarrow \mathbf{0}$ ▷ $\boldsymbol{g}^k = \{g_1^k, \ldots, g_{n_t}^k\}$
4: $\quad$ **while** ($\boldsymbol{f}^k, \boldsymbol{g}^k$ not converged) **do**
5: $\quad\quad f_i^k \leftarrow \varepsilon\,\mathrm{LSE}_{m=1}^{n_s}(log(h_k^m(x)) + \frac{1}{\varepsilon}g_m - \frac{1}{\varepsilon}C(\mathcal{Y}_s^i, \mathcal{Y}_t^m))$ } Sinkhorn loop.
6: $\quad\quad g_j^k \leftarrow \varepsilon\,\mathrm{LSE}_{m=1}^{n_t}(log(h_\theta^m(x)) + \frac{1}{\varepsilon}f_m - \frac{1}{\varepsilon}C(\mathcal{Y}_s^m, \mathcal{Y}_t^j))$ }
7: (LSE is log-sum-exp reduction, i.e, $\mathrm{LSE}_{m=1}^M(V_m) = \log\sum_{m=1}^M exp(V_m)$)
8: $\quad$ **end while**
9: **end for**
10: $\frac{\partial(\mathrm{T}_\varepsilon(h_\theta(x), h_k(x)))}{\partial(h_\theta^i(x))} = f_i^k \quad \forall i\in[n_s],\ k\in\mathcal{K}$ ▷ as dual potentials are gradients of T_ε.
11: $\frac{\partial(\mathcal{L}_\varepsilon(h_\theta(x), h_\mathcal{K}(x)))}{\partial(h_\theta^i(x))} = \sum_{k\in\mathcal{K}} W_{B_k}(h_k(x)) f_i^k / \sum_{k\in\mathcal{K}} W_{B_k}(h_k(x))$.

Using these properties, we calculate gradients of the confident Sinkhorn cost in $eq.$ (3). Algorithm 1 obtains the gradients of the loss function with respect to $h_\theta(x)$ which can be backpropagated to tune model parameters. A crucial computation is to solve the coupling equation in steps 5 and 6. This is done via Sinkhorn iterations which have a linear convergence rate (Peyré et al., 2019).

F Statistical Risk Bounds

Without the loss of generality, we will prove the risk bounds for two local models in learning without forgetting the paradigm. For a sample x, let the output of local models be $y_1 = h_1(x)$ and $y_2 = h_2(x)$ and the global model with trainable parameters be $y_\theta = h_\theta(x)$. To prove Theorem 5.2, we consider the set of IID training samples $S = \{(x^{(1)}, y_1^{(1)}, y_2^{(1)}), \ldots, (x^{(N)}, y_1^{(N)}, y_2^{(N)})\}$.

Lemma F.1. *(from (Frogner et al., 2015)) Let $h_{\hat\theta}, h_{\theta^*} \in \mathcal{H}_g$ be the minimizer of empirical risk $\hat R_S$ and expected risk R, respectively. Then*

$$R(h_{\hat\theta}) \le R(h_{\theta^*}) + 2\sup_{h_\theta\in\mathcal{H}_g} |R(h_\theta) - \hat R_S(h_\theta)| \tag{5}$$

To bound the risk for $h_{\hat\theta}$, we need to prove uniform concentration bounds for the distillation loss. We denote the space of loss functions induced by hypothesis space $\mathcal{H}_g$ as

$$L = \left\{ \ell_\theta : (x, y_1, y_2) \mapsto \frac{w_1(y_1)D(y_\theta, y_1) + w_2(y_2)D(y_\theta, y_2)}{w_1(y_1) + w_2(y_2)} \right\} \tag{6}$$

Lemma F.2. *((Frogner et al., 2015)) Let the transport cost matrix be C and the constant $C_M = \max_{(i,j)} C_{(i,j)}$, then $0 \le D(\cdot,\cdot) \le C_M$, where $D(\cdot,\cdot)$ is 1-Wasserstein distance.*

Definition F.3. *(The Rademacher Complexity (Bartlett and Mendelson, 2002)). Let G be a family of mapping from $\mathcal{Z}$ to $\mathbb{R}$, and $S = (z_1, \ldots, z_N)$ a fixed sample from $\mathcal{Z}$. The empirical Rademacher complexity of G with respect to S is defined as:*

$$\hat{\mathfrak{R}}_S(G) = \mathbb{E}_\sigma\left[\sup_{g \in G} \frac{1}{N} \sum_{i=1}^{n} \sigma_i g(z_i)\right] \tag{7}$$

where $\sigma = (\sigma_1, \ldots, \sigma_N)$, with σ_i's independent uniform random variables taking values in $\{+1, -1\}$. σ_i's are called the Rademacher random variables. The Rademacher complexity is defined by taking expectation with respect to the samples S.

$$\mathfrak{R}_N(G) = \mathbb{E}_S\big[\hat{\mathfrak{R}}_S(G)\big] \tag{8}$$

Theorem F.4. *For any $\delta > 0$, with probability at least 1-δ, the following holds for all $l_\theta \in L$:*

$$\mathbb{E}[\ell_\theta] - \hat{\mathbb{E}}[\ell_\theta] \leq 2\mathfrak{R}_N(L) + \sqrt{\frac{C_M^2 log(1/\delta)}{2N}}. \tag{9}$$

Proof. By definition $\mathbb{E}[\ell_\theta] = R(h_\theta)$ and $\hat{\mathbb{E}}[\ell_\theta] = \hat{R}(h_\theta)$. Let,

$$\Phi(S) = \sup_{\ell \in L} \mathbb{E}[\ell] - \hat{\mathbb{E}}_S[\ell].$$

Let S and S' differ only in sample $(\bar{x}^{(i)}, \bar{y}_1^{(i)}, \bar{y}_2^{(i)})$, by Lemma F.2, it holds that:

$$\Phi(S) - \Phi(S') \leq \sup_{\ell \in L} \hat{\mathbb{E}}_{S'} - \hat{\mathbb{E}}_S = \sup_{h_\theta \in \mathcal{H}} \frac{1}{N}\Big\{ w_1(\bar{y}_1^{(i)}) D(\bar{y}_\theta^{(i)}, \bar{y}_1^{(i)}) + w_2(\bar{y}_2^{(i)}) D(\bar{y}_\theta^{(i)}, \bar{y}_2^{(i)}) - w_1(y_1^{(i)}) D(y_\theta, y_1^{(i)}) - w_2(y_2^{(i)}) D(y_\theta, y_2^{(i)}) \Big\} \leq \frac{2C_M}{N} \tag{10}$$

This inequality can be achieved by putting $D(\bar{y}_\theta^{(i)}, \bar{y}_1^{(i)}) = D(\bar{y}_\theta^{(i)}, \bar{y}_2^{(i)}) = C_M$ and $D(y_\theta^{(i)}, y_1^{(i)}) = D(y_\theta^{(i)}, y_2^{(i)}) = 0$.

Similarly, $\Phi(S') - \Phi(S) \leq C_M/N$, thus $|\Phi(S') - \Phi(S)| \leq C_M/N$. Now, from the McDiarmid's inequality (McDiarmid, 1998) and its usage in (Frogner et al., 2015), we can establish

$$\Phi(S) \leq \mathbb{E}[\Phi(S)] + \sqrt{\frac{K C_M^2 log(1/\delta)}{2N}}. \tag{11}$$

From the bound established in the proof of Theorem B.3 in (Frogner et al., 2015), i.e., $\mathbb{E}_S[\Phi(S)] \leq 2\mathfrak{R}_N(L)$, we can conclude the proof. □

To complete the proof of Theorem Theorem 5.2, we have to treat $\mathfrak{R}_N(L)$ in terms of $\mathfrak{R}_N(\mathcal{H}_g)$.

Now, let $\iota : \mathbb{R}^{|\mathcal{Y}|} \times \mathbb{R}^{|\mathcal{Y}|} \mapsto \mathbb{R}$ defined by $\iota(y, y') = D(\int(y), \int(y)')$, where $\int$ is a softmax function defined over the vector of logits. From Proposition B.10 of (Frogner et al., 2015), we know:

$$|\iota(y, y') - \iota(\bar{y}, \bar{y}')| \leq 2C_M(||y - \bar{y}||_2 + ||y' - \bar{y}'||_2) \tag{12}$$

Let $\iota_s : \mathbb{R}^{|\mathcal{Y}|} \times \mathbb{R}^{|\mathcal{Y}|} \times \mathbb{R}^{|\mathcal{Y}|} \mapsto \mathbb{R}$ defined by:

$$\iota_s(y, y_1, y_2) = \frac{w_1(\mathfrak{s}(y_1)) D(\mathfrak{s}(y), \mathfrak{s}(y_1)) + w_2(\mathfrak{s}(y_2)) D(\mathfrak{s}(y), \mathfrak{s}(y_2))}{w_1(\mathfrak{s}(y_1)) + w_2(\mathfrak{s}(y_2))} \tag{13}$$

$$= \bar{w}_1(\mathfrak{s}(y_1), \mathfrak{s}(y_2))\ D(\mathfrak{s}(y), \mathfrak{s}(y_1)) + \bar{w}_2(\mathfrak{s}(y_1), \mathfrak{s}(y_2))\ D(\mathfrak{s}(y), \mathfrak{s}(y_2)) \tag{14}$$

where $w_1(.), w_2(.)$ are confidence score of local model predictions y_1, y_2 on an input x. $\bar{w}_1(.), \bar{w}_2(.)$ are normalized scores. Note that the local model predictions, i.e., y_1 and y_2 are functions of x, where x is sampled from the data domain distribution $f(x)$. Hence, we can view the loss function as

$$\iota_s(y, y_1, y_2) = D(\mathfrak{s}(y), \mathfrak{s}(y_1)) + D(\mathfrak{s}(y), \mathfrak{s}(y_2)) \tag{15}$$
$$= \iota(y, y_1) + \iota(y, y_2). \tag{16}$$

where y is a function of x_{new} sampled from a weighted distribution $\bar{w}_1(\mathfrak{s}(y_1), \mathfrak{s}(y_2))f(x)$.
The *Lipschitz* constant of $\iota_s(y, y_1, y_2)$ can thus be identified by:

$$|\iota_s(y, y_1, y_2) - \iota_s(\bar{y}, \bar{y}_1, \bar{y}_2)| = |\iota(y, y_1) + \iota(y, y_2) - \iota(\bar{y}, \bar{y}_1) - \iota(\bar{y}, \bar{y}_2)| \tag{17}$$
$$\leq |\iota(y, y_1) - \iota(\bar{y}, \bar{y}_1)| + |\iota(y, y_2) + \iota(\bar{y}, \bar{y}_2)| \tag{18}$$
$$\leq 2C_M(||y - \bar{y}||_2 + ||y_1 - \bar{y}_1||_2 + ||y - \bar{y}||_2 + ||y_2 - \bar{y}_2||_2) \tag{19}$$
$$\leq 4C_M(||y - \bar{y}||_2 + ||y_1 - \bar{y}_1||_2 + ||y_2 - \bar{y}_2||_2) \tag{20}$$
$$\leq 4C_M||(y, y_1, y_2) - (\bar{y}, \bar{y}_1, \bar{y}_2)||_2 \tag{21}$$

Thus, the *Lipschitz* constant of plain Sinkhorn based distillation is $4C_M$.

Proof of Theorem 5.2 We define the space of loss function for k local models:

$$L = \left\{ \iota_\theta : (x, \{y_k\}_{k \in \mathcal{K}}) \mapsto \sum_{k \in \mathcal{K}} w_k(\mathfrak{s}(h^o_\theta(x)))D(\mathfrak{s}(y_k)) \right\}$$

Following the notations in (Frogner et al., 2015), we apply the following generalized Talagrand's lemma (Ledoux and Talagrand, 2013):

Lemma F.5. *Let $\mathcal{F}$ be a class of real functions, and $\mathcal{H} \subset \mathcal{F} = \mathcal{F}_1 \times \ldots \times \mathcal{F}_K$ be a K-valued function class. If $m : \mathbb{R}^K \mapsto \mathbb{R}$ is a $L_\mathfrak{m}$-Lipschitz function and $\mathfrak{m}(0) = 0$, then $\mathfrak{R}_S(\mathfrak{m} \circ \mathcal{H}) \leq 2L_\mathfrak{m} \sum_{k=1}^K \hat{\mathfrak{R}}_S(\mathcal{F}_k)$.*

Now, the Lemma can not be directly applied to the confident Sinkhorn loss as $\mathbf{0}$ is an invalid input. To get around the problem, we assume the global hypothesis space is of the form:

$$\mathcal{H} = \{\mathfrak{s} \circ h : h^o \in \mathcal{H}^o\} \tag{22}$$

Thus, we apply the lemma to the $4C_M$-*Lipschitz* continuous function $\mathfrak{l}$ and the function space:

$$\underbrace{\mathcal{H}^o \times \ldots \times \mathcal{H}^o}_{|\mathcal{Y}| copies} \underbrace{\times \mathcal{I} \times \ldots \times \mathcal{I}}_{|\mathcal{Y}| \times |\mathcal{K}| copies}$$

with $\mathcal{I}$ a singleton function space of identity maps. It holds:

$$\mathfrak{R}_N(L) \leq 8C_M(|\mathcal{Y}|\hat{\mathfrak{R}}_N + |\mathcal{Y}| \times |\mathcal{K}|\hat{\mathfrak{R}}_N(\mathcal{I})) = 8|\mathcal{Y}|C_M\hat{\mathfrak{R}}_N(\mathcal{H}^o) \tag{23}$$

As,

$$\hat{\mathcal{R}}_N(\mathcal{I}) = \mathbb{E}_\sigma\Big[\sup_{g \in \mathcal{I}} \frac{1}{N} \sum_{i=1}^N \sigma_i g(y_i) = 0\Big] = \mathbb{E}_\sigma\Big[\frac{1}{N} \sum_{i=1}^N \sigma_i y_i = 0\Big]$$

Thus, by combining eq. (23) in Theorem F.4 and Lemma F.1 proves the Theorem 5.2.

Since the ε is small in our experiments, we can quantify the difference between Sinkhorn distance and Wasserstein for a given *Lipschitz* cost function.

G Societal Impact

Our work can be extended to different domains. Although in this paper we examined sentiment classifications, other areas, where labels are not available, i.e., zero-shot classification, would also be amenable to federated confident Sinkhorns. Within our approach, a potential downstream task could be to detect cyberbullying. An important area of application for distraught parents, school teachers, and teens. In this case, a sentiment that has a high probability of being classified as cyberbullying can be flagged to either moderators or guardians of a particular application.

A weakness of this approach is that the training of such an application will be based on local models in other domains. Care would be needed in deciding which local models to use in the federation. This choice is highly dependent on the industry and the availability of data. Misuse of our approach could be that the federated training might distill some population-specific information to the global model which makes the central system vulnerable to attacks that might lead to a user-private data breach. As GPU implementation of OT metrics becomes commonplace, we envisage that our approach might help in other Natural Language Processing (NLP) tasks. Indeed, this would be potentially beneficial and open up new avenues for the NLP community.

H Experimental reproducibility

All the experiments were performed on one Quadro RTX 8000 GPU with 48 GB memory. The model architectures were designed on Python (version 3.9.2) library PyTorch (version 1.8.1) under *BSD-style license*. For Sinkhorn iterations and gradient calculations, we use GeomLoss library (version 0.2.4) (https://github.com/jeanfeydy/geomloss) under *MIT licence*. For barycenter calculations, we use POT library (0.7.0) from (https://pythonot.github.io/) under *MIT licence*.

We clip all the text to a maximum length of 200 tokens and pad the shorter sentences with <unk>. To speed up the experiments, we use pretrained BERT-Tiny from https://github.com/google-research/bert.

The batch size is chosen via grid search from the set $\{16, 32, 64, 128, 256, 512, 1024, 2048\}$ and found 1024 to be optimal for performance and speed combination on the considered large datasets. We use Adam optimizer with learning rate chosen via grid search $\{10^{-4}, 10^{-3}, 10^{-2}, 10^{-1}, 1, 10^{1}, 10^{2}, 10^{3}, 10^{4}\}$. All the experiments were run for 20 epochs. The regularization parameter ε is chosen based on minimal loss obtained amongst the set of ε values $\{10^{-4}, 10^{-3}, 10^{-2}, 10^{-1}, 1, 2, 4, 8, 16\}$.

For the Amazon review dataset, we were unable to find the license.

An Information Minimization Based Contrastive Learning Model for Unsupervised Sentence Embeddings Learning

Shaobin Chen[1], **Jie Zhou**[2], **Yuling Sun**[1*] and **Liang He**[1]
[1]School of Computer Science and Technology, East China Normal University Shanghai, China
[2]School of Computer Science, Fudan Univerisity Shanghai, China
shaobin_chen@stu.ecnu.edu.cn, jie_zhou@fudan.edu.cn, {ylsun, lhe}@cs.ecnu.edu.cn

Abstract

Unsupervised sentence embeddings learning has been recently dominated by contrastive learning methods (e.g., SimCSE), which keep positive pairs similar and push negative pairs apart. The contrast operation aims to keep as much information as possible by maximizing the mutual information between positive instances, which leads to redundant information in sentence embedding. To address this problem, we present an information minimization based contrastive learning (`InforMin-CL`) model to retain the useful information and discard the redundant information by maximizing the mutual information and minimizing the information entropy between positive instances meanwhile for unsupervised sentence representation learning. Specifically, we find that information minimization can be achieved by simple contrast and reconstruction objectives. The reconstruction operation reconstitutes the positive instance via the other positive instance to minimize the information entropy between positive instances. We evaluate our model on fourteen downstream tasks, including both supervised and unsupervised (semantic textual similarity) tasks. Extensive experimental results show that our `InforMin-CL` obtains a state-of-the-art performance. Code is made available. [1]

1 Introduction

How to learn universal sentence embeddings by large-scale pre-trained models (Devlin et al., 2019; Liu et al., 2019), such as BERT, has been studied extensively in the literature (Gao et al., 2021; Reimers et al., 2019). Recently, contrastive learning has been used widely to learn better sentence embeddings (Meng et al., 2021; Gao et al., 2021). Generally, contrastive learning uses various data augmentation methods to generate different views of the input sentences and samples positive instances and negative instances from views. Contrastive learning aims to learn effective embeddings by pulling positive instances together and pushing positive and negative instances apart. This operation focus on maximizing the mutual information between positive instances to retain as much information as possible, which includes both the useful and useless information. Previous studies like (Meng et al., 2021; Gao et al., 2021) ignore the redundant information stored in views, which has a bad impact on the performance of downstream tasks, as proved by (Achille and Soatto, 2018; Tian et al., 2020).

Table 1: Training texts may contain various kinds of redundant information, such as stop words, restatement, capitalization, and hyphen.

Original	Where is the party, it sounds great.
Stop words	Where is the party, it sounds great.
Restatement	The party sounds great, where is it.
Capitalization	Where Is The Party, It Sounds Great.
Hyphen	Where-is-the-party, it-sounds-great.

Table 1 gives an example of redundant information stored in training texts. In training texts, the sentences contain much redundant information which is not favorable to the downstream task. The redundant information may be stop words and the style of the sentence (e.g., restatement, capitalization, and hyphen). The existing study (Tian et al., 2020) also shows that discarding redundant information in views can help to improve the performance of the downstream task. Thus, we arise the following question: how to discard this redundant information by choosing the optimal views?

It is natural to solve the above questing via information bottleneck (IB), which has been utilized as an effective and simple method for learning a good embedding by keeping the important information and forgetting redundant information in various tasks (TISHBY, 1999; Chen and Ji, 2020; Tishby and Zaslavsky, 2015). Prompted by this, we at-

* Yuling Sun is the corresponding authors of this paper.
[1]https://github.com/Bin199/InforMin-CL

Proceedings of the 29th International Conference on Computational Linguistics, pages 4821–4831
October 12–17, 2022.

tend to solve the problem by drawing inspiration from the information minimization principle (an idea in IB theory) (Tian et al., 2020): A good set of views share the minimal information necessary to perform well at the downstream task. This method aims to retain useful information and forget redundant information. In this paper, we explore how to address the shortcomings (i.e., ignoring the redundant information stored in views) of previous work for unsupervised sentence representations via the information minimization principle.

We propose an information minimization based contrastive learning (`InforMin-CL`) model for unsupervised sentence embedding learning. `InforMin-CL` incorporates the information minimization principle into contrastive learning to not only learn the important information but also drop the redundant information. We optimize our `InforMin-CL` model from two perspectives: contrast and reconstruction. Firstly, we learn the useful information by a contrast task to maximize the mutual information. This task manages to attract positive pairs and repulse negative pairs. Attracting positive pairs stands for maximizing mutual information between positive instances. Secondly, we propose a reconstruction task that encourages the model to reconstruct the representation of the positive instance via the other one in the same pair.

The algorithm of `InforMin-CL` is easy to understand and can be implemented with just several lines of code. Moreover, our method does not change the major network structure, so it is model-agnostic and can be applied to any representation learning neural networks based on contrastive learning. Experiments in Section 4 show that `InforMin-CL` can help the model learn effective representations that improve downstream task performance. Our main contributions are summarized as follows:

- We propose an `InforMin-CL` model to learn good sentence embeddings by keeping useful information and getting rid of redundant information between positive instances.
- We achieve our model via simple contrast and reconstruction tasks and prove that the reconstruction task can drop redundant information by minimizing the information entropy.
- We achieve new state-of-the-art results on seven supervised tasks and seven unsupervised tasks, which indicates the great advantages of our proposed model.

2 Related Work

2.1 Sentence Representation Learning

Learning sentence embeddings as an important problem in NLP has been widely studied. Recent work focus on leveraging the power of BERT (Devlin et al., 2019) to learn effective sentence embeddings, which are free from artificially supervised signals. BERT-flow (Li et al., 2020) transforms the anisotropic sentence embedding distribution into a smooth isotropic Gaussian distribution through normalizing flows. BERT-whitening (Su et al., 2021) further presents a whitening operation to enhance the isotropy of sentence embeddings and achieves better results.

Then, the contrastive learning approach is applied for sentence embedding learning. IS-BERT (Zhang et al., 2020) proposes a model with a feature extractor on top of BERT and an objective that maximizes the mutual information between global sentence embeddings and local sentence embeddings. CLEAR (Wu et al., 2020) employs multiple sentence-level augmentation strategies to learn sentence representation. Coco-LM (Meng et al., 2021) employs an auxiliary language model to corrupt text sequences, upon which it constructs a token-level task and a sequence-level task for pretraining the main model. Gao *et al.* (Gao et al., 2021) presents an unsupervised approach that predicts input itself with dropout noise and a supervised approach utilizing natural language inference datasets. SCD (Klein and Nabi, 2022) leverages the self-contrast of augmented samples obtained by dropout, which eliminates the reliance on negative pairs. However, all these studies lack consideration of discarding redundant information stored in views. In our work, we consider and solve the problem in the framework of the information minimization principle.

2.2 Information Minimization Principle

Information minimization principle (Tian et al., 2020) has been proposed to retain the minimal information necessary. In recent years, researchers utilize the information minimization principle to improve image representations (Tian et al., 2020; Tsai et al., 2020). Furthermore, information bottleneck is used to improve the interpretability of the

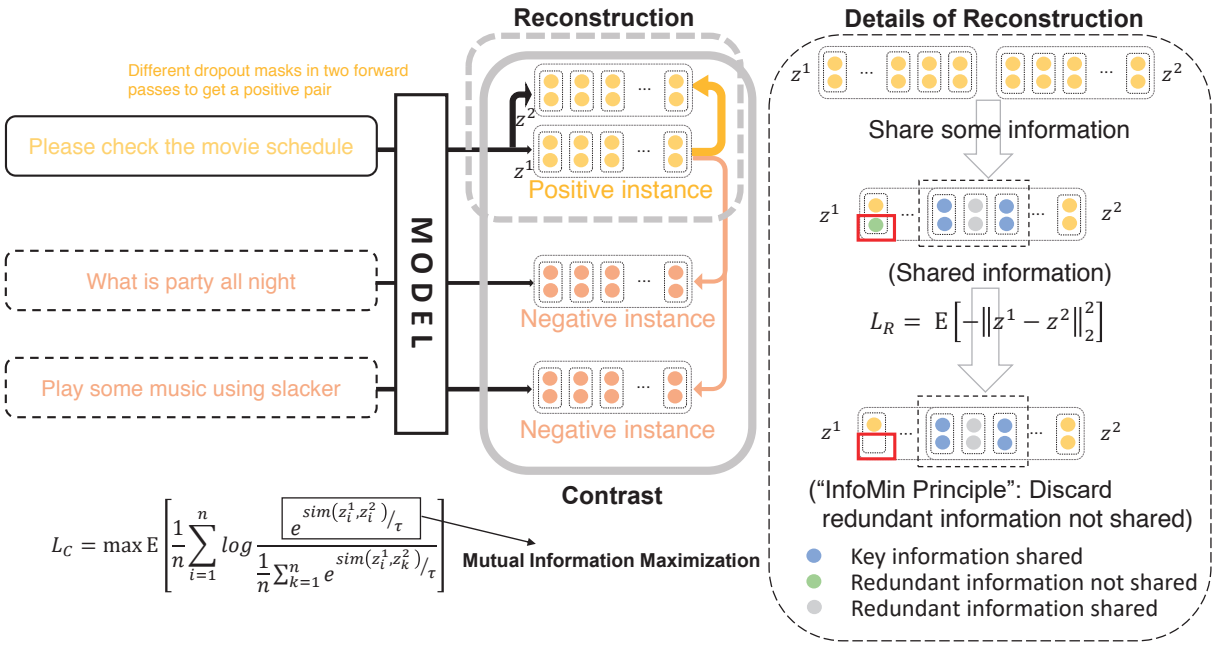

Figure 1: The architecture of our proposed framework. Contrast: Negative pairs are pushed apart while positive pairs are pulled together, which suggests maximizing the mutual information between positive instances. Reconstruction: We drop redundant information in z^1 not shared with z^2 (marked by the green point) by reconstructing one positive instance via the other positive instance.

attention-based models (Zhou et al., 2021). Tian *et al.* (Tian et al., 2020) shows good views for a given task in a contrastive representation learning framework should retain task-relevant information while minimizing irrelevant nuisances. However, it focuses on eliminating the task-irrelevant information via downstream datasets. Tsai *et al.* (Tsai et al., 2020) focuses on the multi-view setting between input and self-supervised signals and adopts self-supervised signals to reconstruct learned representations to discard task-irrelevant information. Our work also differs from (Tsai et al., 2020) in two perspectives: 1) we discard redundant information in texts while (Tsai et al., 2020) drops noise information stored in images, and additionally different self-supervised signals are used in two work; 2) (Tsai et al., 2020) validates their method by learning visual features evaluated by supervised tasks while sentence embedding learning evaluates embeddings via not only supervised tasks but also unsupervised tasks.

3 Method

We propose an `InforMin-CL` model for unsupervised sentence representation learning (Figure 1). There are two main steps, contrast and reconstruction. We first present the contrast objective to learn the useful information: we push apart positive instances and negative instances while pulling positive instances together, which implies maximizing mutual information between positive instances. Later we present the reconstruction task to drop the useless information: we minimize the conditional information entropy of one positive instance given the other positive instance. Algorithm 1 provides the pseudo-code of *InforMin-CL*.

Algorithm 1 Pseudocode of *InforMin-CL* in a PyTorch-like style.

Input: batch size N, temperature τ, structure of f and Γ.
Output: encoder network $f(\cdot)$.

for sampled minibatch $\{x_k\}_{k=1}^{N}$ **do**
  for all $k \in \{1, \ldots, N\}$ **do**
    draw two augmentation functions $t \sim \Gamma$, $t' \sim \Gamma$
    $\tilde{x}_{2k-1} = t(x_k)$
    $z_{2k-1} = f(\tilde{x}_{2k-1})$
    $\tilde{x}_{2k} = t'(x_k)$
    $z_{2k} = f(\tilde{x}_{2k})$
  end for
  for all $i \in \{1, \ldots, 2N\}$ and $j \in \{1, \ldots, 2N\}$ **do**
    $s_{i,j} = z_i^{\mathrm{T}} z_j / (\|z_i\| \, \|z_j\|)$
  end for
  $L_C = \max \mathrm{E}\left[\frac{1}{N}\sum_{i=1}^{N} \log \frac{\exp(s_{2i-1,2i}/\tau)}{\frac{1}{N}\sum_{k=1}^{N}\exp(s_{2i-1,2k}/\tau)}\right]$
  $L_R = \mathrm{E}\left[-\|z_{2k-1} - z_{2k}\|_2^2\right]$
  $L = L_C + L_R$
  update f to minimize L
end for

3.1 Contrast

Contrastive learning attends to learning effective representation by pulling positive sample pairs together and pushing apart negative sample pairs. We build upon the recent success of unsupervised SimCSE (Gao et al., 2021) and take the embeddings derived from the same sentence with independently different dropout masks as positive instances. We adopt the dropout mask (with default dropout probability p = 0.1) as an augmentation skill, which is proved to outperform other skills (Gao et al., 2021), to obtain positive pairs. The positive pair takes the same sentence, and their embeddings only differ in dropout masks. Other sentences in the same mini-batch are seen as negative instances.

We denote input, instance, and self-supervised signal as X, Z, and S. We feed the same input x to the encoder twice by applying different dropout masks and then get positive instances z^1 and z^2. In our work, we take one instance z^2 in positive pair as a self-supervised signal. The information required for downstream tasks is referred to as "key information": T. I and H represent mutual information and information entropy.

Let Z^{sup} be the sufficient supervised representation and $Z^{sup_{\min}}$ be the minimal and sufficient supervised representation:

$$\begin{aligned} Z^{\text{sup}} &= \arg\max_{Z} I\left(Z;T\right) \\ Z^{sup_{\min}} &= \arg\min_{Z} H\left(Z|T\right) \\ s.t.\ \ & I\left(Z;T\right) is \,\text{maximized} \end{aligned} \tag{1}$$

Let Z^{ssl} be the sufficient self-supervised representation and $Z^{ssl_{\min}}$ be the minimal and sufficient self-supervised representation:

$$\begin{aligned} Z^{ssl} &= \arg\max_{Z} I\left(Z;S\right) \\ Z^{ssl_{\min}} &= \arg\min_{Z} H\left(Z|S\right) \\ s.t.\ \ & I\left(Z;S\right) is \,\text{maximized} \end{aligned} \tag{2}$$

Then, we give theorem 1. For proof, please refer to (Tsai et al., 2020).

Theorem 1 *The supervised learned representations contain all the key information in the input (i.e. $I\left(X;T\right)$). The self-supervised learned representations contain all the key information in the input with a potential loss ε:*

$$\begin{aligned} I\left(X;T\right) &= I\left(Z^{\text{sup}};T\right) = I\left(Z^{\text{sup}_{\min}};T\right) \\ &\geq I\left(Z^{ssl};T\right) \\ &\geq I\left(Z^{ssl_{\min}};T\right) \\ &\geq I\left(X;T\right) - \varepsilon \end{aligned} \tag{3}$$

The contrastive learning objective maximizes the dependency between positive instance z^1 and self-supervised signal z^2, which suggests maximizing the mutual information $I\left(z^1;z^2\right)$. Theorem 1 suggests that maximizing $I\left(z^1;z^2\right)$ results in z_1 containing almost all the information required for downstream tasks from the input x. Note that T is utilized only for describing our method and in practice, no downstream datasets are used in the pre-training phase.

We use a contrastive learning objective similar to that in (Oord et al., 2018), which is a mutual information lower bound with low variance:

$$\mathcal{L}_C = \max \mathbb{E}\left[\frac{1}{N}\sum_{i=1}^{N}\log\frac{e^{sim\left(z_i^1,z_i^2\right)/\tau}}{\frac{1}{N}\sum_{k=1}^{N}e^{sim\left(z_i^1,z_k^2\right)/\tau}}\right] \tag{4}$$

where $\left(z_1^1,z_1^2\right),\ldots,\left(z_N^1,z_N^2\right) \sim P^N\left(Z^1,Z^2\right)$, z_i^1, z_i^2 are two positive instances of the i-th examples. N refers to batch size and P refers to the statistical distribution of $\left(Z^1,Z^2\right)$.

3.2 Reconstruction

The details of the reconstruction are illustrated in the right of Figure 1. The positive instances z^1 and z^2 contain independent information while sharing some information. The noise information (marked as the green point) in z^1 is expected to be discarded by the reconstruction task. We prove that this task can discard the useless information by minimizing the information entropy.

The reconstruction task encourages the self-supervised signal z^2 to reconstruct the learned representation z^1, which suggests maximizing the log conditional likelihood $\mathbb{E}_{P_{Z^1,Z^2}}\left[\log P\left(Z^1|Z^2\right)\right]$. We know that

$$-H\left(Z^1|Z^2\right) = \mathbb{E}_{P_{Z^1,Z^2}}\left[\log P\left(Z^1|Z^2\right)\right] \tag{5}$$

Thus, this reconstruction also means minimizing $H\left(Z^1|Z^2\right)$.

Theorem 2 *The sufficient self-supervised representation contains more redundant information*

in the input than the sufficient and minimal self-supervised representation. The latter contains an amount of the information, $I(X;S|T)$, that cannot be discarded from the input:

$$\begin{aligned} I\left(Z^{ssl};X|T\right) &= I\left(X;S|T\right) + I\left(Z^{ssl};X|S,T\right) \\ &\geq I\left(Z^{ssl_{\min}};X|T\right) = I\left(X;S|T\right) \\ &\geq I\left(Z^{\sup_{\min}};X|T\right) = 0 \end{aligned} \tag{6}$$

Theorem 2 (please refer to (Tsai et al., 2020) for proof) indicates that Z^{ssl} contains two parts of redundant information while $Z^{ssl_{\min}}$ contains one part of redundant information, discarding $I\left(Z^{ssl};X|S,T\right)$.

Thus, if z^2 can perfectly reconstruct z^1 for any

$$\left(z^1, z^2\right) \sim P_{Z^1,Z^2} \tag{7}$$

under the constraint that $I\left(z^1;z^2\right)$ is maximized, we get $z^{1_{ssl_{\min}}}$ according to Eq. 2. And then z^1 discards redundant information, excluding $I\left(z^1;z^2|t\right)$ (i.e., the amount of redundant information in the shared information between two positive instances z^1 and z^2). For easier optimization, we use $\mathbb{E}_{P_{Z^1,Z^2}}\left[\log Q_\Phi\left(Z^1|Z^2\right)\right]$ as the lower bound of $\mathbb{E}_{P_{Z^1,Z^2}}\left[\log P\left(Z^1|Z^2\right)\right]$. In our deployment, we utilize the design in Eq. 4 and let $Q_\Phi\left(Z^1|Z^2\right)$ be Gaussian $N\left(Z^1|Z^2,\sigma \mathrm{I}\right)$ with σI as a diagonal matrix. Hence, we obtain the reconstruction objective as follows:

$$\mathcal{L}_R = \mathbb{E}_{z^1,z^2\sim P_{Z^1,Z^2}}\left[-\left\|z^1 - z^2\right\|_2^2\right] \tag{8}$$

We combine two objectives as a total objective:

$$\mathcal{L} = \mathcal{L}_C + \lambda * \mathcal{L}_R \tag{9}$$

where λ is a hyper-parameter. Training model with the total loss enables us to discard redundant information in views.

4 Experiment

4.1 Evaluation Setup

We conduct our experiments on seven standard supervised tasks and also seven unsupervised tasks. We use the SentEval Toolkit (Conneau and Kiela, 2018) for evaluation. Following (Reimers et al., 2019; Gao et al., 2021), we take unsupervised tasks as the main comparison of the sentence embedding approaches and supervised results for reference.

Unsupervised Tasks We evaluate representations on seven semantic textual similarity (STS) tasks: STS 2012-2016 (Agirre et al., 2012, 2013, 2014, 2015, 2016), STS Benchmark (Cer et al., 2017), and SICK-Relatedness (Marelli et al., 2014) and compute the cosine similarity between sentence embeddings. All the unsupervised experiments are fully unsupervised, which means no STS training datasets are used and all embeddings are fixed once they are trained. For the sake of comparability, we follow the evaluation protocol of (Gao et al., 2021), employing Spearman's rank correlation and aggregation on all topic subsets.

Supervised Tasks We evaluate representations on seven supervised tasks: MR (Pang and Lee, 2005), CR (Hu and Liu, 2004), SUBJ (Pang and Lee, 2004), MPQA (Wiebe et al., 2005), SST-2 (Socher et al., 2013), TREC (Voorhees and Tice, 2000) and MRPC (Dolan and Brockett, 2005). A logistic regression classifier is trained on the top of (frozen) sentence embeddings produced by different methods. We follow default configurations from SentEval and use accuracy as the metric.

Training Details We start from pre-trained BERT (Devlin et al., 2019) (uncased) or RoBERTa (Liu et al., 2019) (cased). Similar to (Gao et al., 2021), we train our `InforMin-CL` in an unsupervised fashion on 10^6 randomly sampled sentences from English Wikipedia. During training, we add an MLP layer on the top of the [CLS] representation as sentence embeddings and directly take the [CLS] representation as sentence embeddings at testing time. A masked language modeling (MLM) objective (Devlin et al., 2019) is added as an optional auxiliary loss to the Eq. 9: $\mathcal{L} + \beta * \mathcal{L}_{MLM}$ (β is a hyper-parameter). For all results, we use the following hyper-parameters: epoch: 1, temperature τ: 0.05, optimizer: Adam (Kingma and Ba, 2015)). We carry out grid-search of batch size $\in \{64, 128, 256\}$ and learning rate $\in \{1e-5, 3e-5, 5e-5\}$ on STS-B development sets. During the training process, we save the checkpoint with the highest score on the STS-B development set to find the best hyperparameters. We adopt the hyperparameter settings listed in Table 3. For all results, we use a PC with a GeForce RTX 3090 GPU (CUDA 11, PyTorch 1.7.1).

4.2 Main Results

Baselines We compare `InforMin-CL` to previous typical sentence embedding methods, which

Table 2: Unsupervised task results (spearman's correlation). †: results from (Gao et al., 2021). ♡: results from (Klein and Nabi, 2022). All other results are reproduced and reevaluated by ourselves.

Model	STS12	STS13	STS14	STS15	STS16	STS-B	SICK-R	Avg.
GloVe embeddings (avg.)†	55.14	70.66	59.73	68.25	63.66	58.02	53.76	61.32
$\text{BERT}_{\text{base}}$ (first − last avg.)†	39.70	59.38	49.67	66.03	66.19	53.87	62.06	56.70
$\text{BERT}_{\text{base}}$−flow†	58.40	67.10	60.85	75.16	71.22	68.66	64.47	66.55
$\text{BERT}_{\text{base}}$−whitening†	57.83	66.90	60.90	75.08	71.31	68.24	63.73	66.28
IS − $\text{BERT}_{\text{base}}$†	56.77	69.24	61.21	75.23	70.16	69.21	64.25	66.58
CT − $\text{BERT}_{\text{base}}$†	61.63	76.80	68.47	77.50	76.48	74.31	69.19	72.05
SCD − $\text{BERT}_{\text{base}}$♡	66.94	78.03	69.89	78.73	76.23	76.30	**73.18**	74.19
SimCSE − $\text{BERT}_{\text{base}}$	67.01	82.14	73.76	80.49	79.01	77.04	69.94	75.63
InforMin-CL − $\text{BERT}_{\text{base}}$	**70.22**	**83.48**	**75.51**	**81.72**	**79.88**	**79.27**	71.03	**77.30**
$\text{RoBERTa}_{\text{base}}$ (first − last avg.)†	40.88	58.74	49.07	65.63	61.48	58.55	61.63	56.57
$\text{RoBERTa}_{\text{base}}$−whitening†	46.99	63.24	57.23	71.36	68.99	61.36	62.91	61.73
DeCLUTR − $\text{RoBERTa}_{\text{base}}$†	52.41	75.19	65.52	77.12	78.63	72.41	68.62	69.99
SCD − $\text{RoBERTa}_{\text{base}}$♡	63.53	77.79	69.79	80.21	77.29	76.55	72.10	73.89
SimCSE − $\text{RoBERTa}_{\text{base}}$	**70.32**	82.48	**74.84**	**82.13**	**82.14**	**81.57**	68.62	**77.44**
InforMin-CL − $\text{RoBERTa}_{\text{base}}$	69.79	**82.57**	73.36	80.91	81.28	81.07	**70.30**	77.04
SimCSE − $\text{RoBERTa}_{\text{large}}$	**72.64**	83.78	**75.83**	**84.24**	**80.12**	**81.10**	69.81	**78.22**
InforMin-CL − $\text{RoBERTa}_{\text{large}}$	70.91	**84.20**	75.57	82.26	79.68	**81.10**	**72.81**	78.08

Table 3: Batch sizes, learning rates and λ adopted for InforMin-CL.

	BERT	RoBERTa	
	base	base	large
Batch size	128	128	128
Learning rate	3e-5	1e-5	3e-5
λ	0.4	4	4

include averaging GloVe embeddings (Pennington et al., 2014), Skip-thought (Kiros et al., 2015) and average BERT or RoBERTa embeddings. We also compare to post-processing methods and methods using a contrastive objective. Post-processing methods include BERT-flow (Li et al., 2020) and BERT-whitening (Su et al., 2021). Methods using a contrastive objective include IS-BERT (Zhang et al., 2020), DeCLUTR (Giorgi et al., 2021), CT (Carlsson et al., 2020), SimCSE (Gao et al., 2021), and SCD (Klein and Nabi, 2022). IS-BERT maximizes the agreement between global and local features. DeCLUTR takes different spans from the same document as positive pairs. CT aligns embeddings of the same sentence from two different encoders. SimCSE takes the embedding of the same input with different dropouts as positive pairs. SCD leverages the self-contrast of augmented samples obtained by dropout.

Performance on Unsupervised Tasks Table 2 shows the evaluation results on seven STS tasks. InforMin-CL achieves comparable or better results than previous state-of-the-art baselines. For $\text{BERT}_{\text{base}}$ based models, our method outperforms the best approach with a large margin ($+1.67\%$) on average. For RoBERTa based model, InforMin-CL obtains comparable results (less than 0.4 points in average). All these indicate that our model can improve the performance of downstream tasks by forgetting the irrelevant information in pre-training phase.

Performance on Supervised Tasks Table 4 shows the evaluation results on seven supervised tasks. Results indicate that InforMin-CL performs on par or better than all baselines. Our method achieves better results on most of and even all tasks using RoBERTa, with an average gain 0.81% on $\text{RoBERTa}_{\text{base}}$ and 1.66% on $\text{RoBERTa}_{\text{large}}$ respectively. With an MLM task added, further gains on average results are observed for BERT and RoBERTa. It raises the average scores of InforMin-CL from 85.52% to 86.96% for $\text{BERT}_{\text{base}}$ and from 86.11% to 87.01% for $\text{RoBERTa}_{\text{base}}$. Particularly, InforMin-CL w/ MLM obtains the outperforms all the baselines in average.

Considering results of both supervised and unsupervised tasks, we present the following findings: 1) BERT-based InforMin-CL performs better on unsupervised tasks; 2) RoBERTa-based InforMin-CL achieves better results on supervised tasks. The difference in performance us-

Table 4: Supervised task results (accuracy). †: results from (Gao et al., 2021). ♡: results from (Klein and Nabi, 2022). All other results are reproduced and reevaluated by ourselves. w/ MLM: adding MLM as an auxiliary task with $\beta = 0.1$.

Model	MR	CR	SUBJ	MPQA	SST	TREC	MRPC	Avg.
GloVe embeddings (avg.)†	77.25	78.30	91.17	87.85	80.18	83.00	72.87	81.52
Skip − thought†	76.50	80.10	93.60	87.10	82.00	92.20	73.00	83.50
Avg. BERT embeddings†	78.66	86.25	94.37	88.66	84.40	**92.80**	69.54	84.94
BERT− [CLS] embeddings†	78.68	84.85	94.21	88.23	84.13	91.40	71.13	84.66
IS − $\text{BERT}_{\text{base}}$†	81.09	**87.18**	94.96	88.75	85.96	88.64	74.24	85.83
SCD − $\text{BERT}_{\text{base}}$♡	73.21	85.80	**99.56**	88.67	85.59	89.80	75.71	85.52
SimCSE − $\text{BERT}_{\text{base}}$	81.47	86.86	94.79	89.25	86.27	89.40	72.81	85.84
InforMin-CL − $\text{BERT}_{\text{base}}$	80.99	85.72	94.63	**89.47**	85.67	88.20	73.97	85.52
w/ MLM	**82.87**	87.05	95.22	88.43	**87.15**	92.20	**75.77**	**86.96**
SimCSE − $\text{RoBERTa}_{\text{base}}$	81.26	87.36	93.58	87.56	86.93	84.80	75.01	85.21
SCD − $\text{RoBERTa}_{\text{base}}$♡	82.17	87.76	93.67	85.69	88.19	83.40	76.23	85.30
InforMin-CL − $\text{RoBERTa}_{\text{base}}$	82.22	88.08	93.57	**87.75**	87.59	86.60	76.99	86.11
w/ MLM	**83.49**	**88.69**	**94.79**	86.81	**88.30**	**89.40**	**77.57**	**87.01**
SimCSE − $\text{RoBERTa}_{\text{large}}$	80.85	85.99	93.08	87.65	86.33	89.00	72.46	85.05
InforMin-CL − $\text{RoBERTa}_{\text{large}}$	**82.50**	**88.32**	**93.81**	**89.38**	**87.64**	**90.80**	**74.49**	**86.71**

ing BERT and RoBERTa is mainly caused by the difference in pre-training corpus. BERT is trained over 16 GB text (BooksCorpus (Zhu et al., 2015) and English Wikipedia) while RoBERTa is trained over totally 160 GB of uncompressed text (BooksCorpus (Zhu et al., 2015), English Wikipedia, CC-NEWS (Nagel, 2016), OPENWEBTEXT (Gokaslan et al., 2019), and STORIES (Trinh and Le, 2018)). Thus, the diverse large-scale high quality datasets enhance the RoBERTa to learn the important and useful information with limited parameters. InforMin-CL, which improves performance by discarding redundant information, struggles to represent its effects in this setting due to less noise information. This causes the unsupervised results of InforMin-CL are similar to competitors for RoBERTa-based models.

4.3 Ablation Study

Influence of λ We investigate how different reconstruction objectives with λ from 0.04 to 4 affect our model's performance. We report the average performance of unsupervised tasks and supervised tasks in this experiment. The results are obtained using $\text{BERT}_{\text{base}}$. Results demonstrate that InforMin-CL constantly works well over this wide range of λ. As shown in Table 5, with increasing λ, the performance of both unsupervised and supervised tasks rises first and falls later.

Influence of β We introduce one more optional variant which adds a masked language modeling (MLM) objective to the Eq. 9: $\mathcal{L} + \beta * \mathcal{L}_{MLM}$ (β is a hyper-parameter). We analyze how different β influence the performance on unsupervised and supervised tasks. As we show in Table 6, we find that adding MLM objectives with different β consistently helps improve performance on supervised tasks but brings a significant drop in STS tasks.

Influence of Batch Sizes To explore the impact of batch sizes, we report the average performance of downstream tasks with batch sizes (N in Eq. 4) from 64 to 256. In this experiment, only batch size changes while all other hyper-parameters keep unchanged. We use $\text{BERT}_{\text{base}}$ to evaluate on the test set of unsupervised and supervised tasks. As we show in Table 7, we find that InforMin-CL is not sensitive to batch size, similar to SimCSE, mainly caused by the good set of initial parameters.

Table 5: Ablation studies of different hyper-parameters λ. The results are based on the test sets using $\text{BERT}_{\text{base}}$.

λ	Avg. Sup	Avg. Unsup
0.04	85.20	76.09
0.4	**85.52**	**77.30**
4	85.03	77.18

4.4 Uniformity and Alignment

We further conduct analysis to understand the inner workings of InforMin-CL.

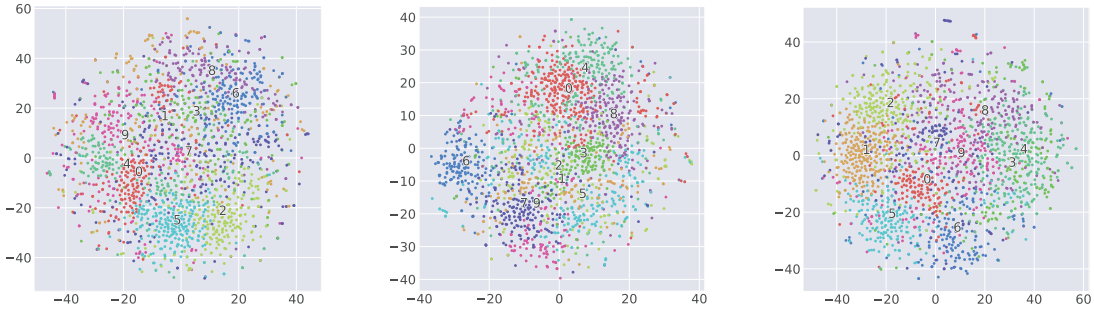

(a) SimCSE (b) SCD (c) InforMin-CL

Figure 2: The t-SNE of sentence representations learned with SimCSE, SCD and InforMin-CL using $\text{BERT}_{\text{base}}$. The points are embeddings of sentences sampled from the IMDB dataset without fine-tuning.

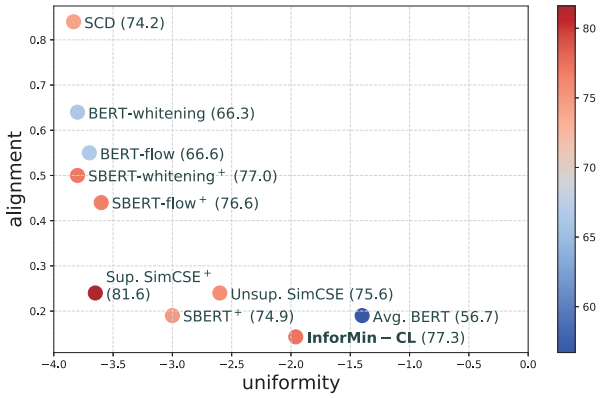

Figure 3: Quantitative analysis of embeddings - *alignment* vs. *uniformity* (the smaller, the better). The plot of models is based on $\text{BERT}_{\text{base}}$. Points represent average STS performance with Spearman's correlation color coded (+ corresponds to supervised methods).

Table 6: Ablation studies of the MLM objective based on the test sets using $\text{BERT}_{\text{base}}$.

Model	Avg. Sup	Avg. Unsup
w/o MLM	85.52	**77.30**
w/ MLM		
$\beta = 0.01$	86.46	**63.59**
$\beta = 0.1$ (ours)	86.96	63.25
$\beta = 1.0$	**87.04**	60.85

Table 7: Ablation studies of different batch sizes. The results are based on test sets using $\text{BERT}_{\text{base}}$.

Batch size	64	128	256
Avg. Sup	85.38	85.52	**85.77**
Avg. Unsup	76.64	**77.30**	76.14

Qualitative Analysis As shown in (Wang and Isola, 2020), the asymptotics of the contrastive learning objective (4) can be expressed by the following equation when the number of negative instances approaches infinity:

$$\mathcal{L}_C = \max \Big[\frac{1}{N} \sum_{i=1}^{N} \mathbb{E}\big[sim\left(z_i^1, z_i^2\right)/\tau\big] - \frac{1}{N} \sum_{i=1}^{N} \mathbb{E}\Big[\log \frac{1}{N} \sum_{k=1}^{N} e^{sim\left(z_i^1, z_k^2\right)/\tau}\Big]\Big] \tag{10}$$

The first term in square brackets in Eq. 10 improves alignment of the space. The alignment performs better when the similarity score rises. While optimizing the reconstruction objective, z^1 and z^2 are pulled closer, which means that the similarity score of z^1 and z^2 becomes higher. In other words, InforMin-CL effectively improves alignment of pre-trained embeddings while keeping a good uniformity, which is the key to the success of InforMin-CL. We also follow (Wang and Isola, 2020) to use uniformity and alignment to measure the quality of representation space for InforMin-CL and other models. Figure 3 shows uniformity and alignment of different sentence embedding models along with their STS averaged results. InforMin-CL achieves the best in terms of *alignment* (0.143), which can be related to the strong effect of the reconstruction objective. In terms of *uniformity*, InforMin-CL is slightly inferior to unsupervised SimCSE. This is also reflected in the final results in the t-SNE plots.

Quantitative Analysis The t-SNE (Reif et al., 2019) plot in Figure 2 demonstrates the advantages of InforMin-CL. We sample 2000 sentences from IMDB (Maas et al., 2011) dataset and gen-

erate the embeddings of sentences using SimCSE, SCD and InforMin-CL. We use K-Means (Jain and Dubes, 1988) clustering to group similar sentence embeddings and form 10 clusters. Results indicate that similar sentence pairs (marked by same colors) generated by `InforMin-CL` are more aligned.

5 Limitations

Although our method outperforms baselines on both unsupervised and supervised tasks in most cases, there are still at least two limitations. First, we simply sample negative instances from other sentences in the mini-batch, which may lead to false negatives. Punishing false negatives during training by assigning lower weight for negatives with higher similarity may be a solution. Second, although redundant information is discarded, what redundant information forgets and remains is unknown. It would be interesting to explore this problem by integrating interpretation methods.

6 Conclusion

In this work, we propose `InforMin-CL`, an effective contrastive learning approach, which improves state-of-the-art sentence embedding performance on downstream tasks. `InforMin-CL` discards redundant information stored in positive instances by encouraging one positive instance to reconstruct the other positive instance in the same pair. We test `InforMin-CL` on seven supervised and seven unsupervised tasks. Experimental results indicate our method outperforms all previous competitors.

Acknowledgements

We thank the anonymous reviewers for their valuable comments. We also thank Shanghai Science and Technology Innovation Action Plan Project (21511104500) for funding this research.

References

Alessandro Achille and Stefano Soatto. 2018. Emergence of invariance and disentanglement in deep representations. *The Journal of Machine Learning Research*, 19(1):1947–1980.

Eneko Agirre, Carmen Banea, Claire Cardie, Daniel Cer, Mona Diab, Aitor Gonzalez-Agirre, Weiwei Guo, Inigo Lopez-Gazpio, Montse Maritxalar, Rada Mihalcea, et al. 2015. Semeval-2015 task 2: Semantic textual similarity, english, spanish and pilot on interpretability. In *Proceedings of the 9th international workshop on semantic evaluation (SemEval 2015)*, pages 252–263.

Eneko Agirre, Carmen Banea, Claire Cardie, Daniel Cer, Mona Diab, Aitor Gonzalez-Agirre, Weiwei Guo, Rada Mihalcea, German Rigau, and Janyce Wiebe. 2014. Semeval-2014 task 10: Multilingual semantic textual similarity. In *Proceedings of the 8th international workshop on semantic evaluation (SemEval 2014)*, pages 81–91.

Eneko Agirre, Carmen Banea, Daniel Cer, Mona Diab, Aitor Gonzalez Agirre, Rada Mihalcea, German Rigau Claramunt, and Janyce Wiebe. 2016. Semeval-2016 task 1: Semantic textual similarity, monolingual and cross-lingual evaluation. In *SemEval-2016. 10th International Workshop on Semantic Evaluation; 2016 Jun 16-17; San Diego, CA. Stroudsburg (PA): ACL; 2016. p. 497-511.*

Eneko Agirre, Daniel Cer, Mona Diab, and Aitor Gonzalez-Agirre. 2012. Semeval-2012 task 6: A pilot on semantic textual similarity. In ** SEM 2012: The First Joint Conference on Lexical and Computational Semantics–Volume 1: Proceedings of the main conference and the shared task, and Volume 2: Proceedings of the Sixth International Workshop on Semantic Evaluation (SemEval 2012)*, pages 385–393.

Eneko Agirre, Daniel Cer, Mona Diab, Aitor Gonzalez-Agirre, and Weiwei Guo. 2013. * sem 2013 shared task: Semantic textual similarity. In *Second joint conference on lexical and computational semantics (* SEM), volume 1: proceedings of the Main conference and the shared task: semantic textual similarity*, pages 32–43.

Fredrik Carlsson, Amaru Cuba Gyllensten, Evangelia Gogoulou, Erik Ylipää Hellqvist, and Magnus Sahlgren. 2020. Semantic re-tuning with contrastive tension. In *International Conference on Learning Representations*.

Daniel Cer, Mona Diab, Eneko Agirre, Iñigo Lopez-Gazpio, and Lucia Specia. 2017. Semeval-2017 task 1: Semantic textual similarity multilingual and crosslingual focused evaluation. In *Proceedings of the 11th International Workshop on Semantic Evaluation (SemEval-2017)*, pages 1–14.

Hanjie Chen and Yangfeng Ji. 2020. Learning variational word masks to improve the interpretability of neural text classifiers. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 4236–4251.

Alexis Conneau and Douwe Kiela. 2018. Senteval: An evaluation toolkit for universal sentence representations. *arXiv preprint arXiv:1803.05449*.

Jacob Devlin, Ming-Wei Chang, and Kenton et al. Lee. 2019. Bert: Pre-training of deep bidirectional transformers for language understanding. In *NAACL-HLT (1)*.

William B Dolan and Chris Brockett. 2005. Automatically constructing a corpus of sentential paraphrases.

In *Proceedings of the Third International Workshop on Paraphrasing (IWP2005)*.

Tianyu Gao, Xingcheng Yao, and Danqi Chen. 2021. Simcse: Simple contrastive learning of sentence embeddings. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing, EMNLP 2021, Virtual Event / Punta Cana, Dominican Republic, 7-11 November, 2021*, pages 6894–6910.

John Giorgi, Osvald Nitski, Bo Wang, and Gary Bader. 2021. DeCLUTR: Deep contrastive learning for unsupervised textual representations. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 879–895.

Aaron Gokaslan, Vanya Cohen, Ellie Pavlick, and Stefanie Tellex. 2019. Openwebtext corpus.

Minqing Hu and Bing Liu. 2004. Mining and summarizing customer reviews. In *Proceedings of the tenth ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 168–177.

Anil K Jain and Richard C Dubes. 1988. *Algorithms for clustering data*. Prentice-Hall, Inc.

Diederik P. Kingma and Jimmy Ba. 2015. Adam: A method for stochastic optimization. In *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*.

Ryan Kiros, Yukun Zhu, Russ R Salakhutdinov, Richard Zemel, Raquel Urtasun, Antonio Torralba, and Sanja Fidler. 2015. Skip-thought vectors. In *Advances in neural information processing systems*, pages 3294–3302.

Tassilo Klein and Moin Nabi. 2022. Scd: Self-contrastive decorrelation for sentence embeddings. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics*. Association for Computational Linguistics (ACL).

Bohan Li, Hao Zhou, Junxian He, Mingxuan Wang, Yiming Yang, and Lei Li. 2020. On the sentence embeddings from pre-trained language models. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing, EMNLP 2020, Online, November 16-20, 2020*, pages 9119–9130.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Roberta: A robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692*.

Andrew Maas, Raymond E Daly, Peter T Pham, Dan Huang, Andrew Y Ng, and Christopher Potts. 2011. Learning word vectors for sentiment analysis. In *Proceedings of the 49th annual meeting of the association for computational linguistics: Human language technologies*, pages 142–150.

Marco Marelli, Stefano Menini, Marco Baroni, Luisa Bentivogli, Raffaella Bernardi, and Roberto Zamparelli. 2014. A sick cure for the evaluation of compositional distributional semantic models. In *Proceedings of the Ninth International Conference on Language Resources and Evaluation (LREC'14)*, pages 216–223.

Yu Meng, Chenyan Xiong, Payal Bajaj, Saurabh Tiwary, Paul Bennett, Jiawei Han, and Xia Song. 2021. COCO-LM: Correcting and contrasting text sequences for language model pretraining. In *Conference on Neural Information Processing Systems*.

Sebastian Nagel. 2016. Cc-news. *URL: http://web. archive. org/save/http://commoncrawl. org/2016/10/newsdatasetavailable*.

Aaron van den Oord, Yazhe Li, and Oriol Vinyals. 2018. Representation learning with contrastive predictive coding. *arXiv preprint arXiv:1807.03748*.

Bo Pang and Lillian Lee. 2004. A sentimental education: sentiment analysis using subjectivity summarization based on minimum cuts. In *Proceedings of the 42nd Annual Meeting on Association for Computational Linguistics*, pages 271–es.

Bo Pang and Lillian Lee. 2005. Seeing stars: exploiting class relationships for sentiment categorization with respect to rating scales. In *Proceedings of the 43rd Annual Meeting on Association for Computational Linguistics*, pages 115–124.

Jeffrey Pennington, Richard Socher, and Christopher D Manning. 2014. Glove: Global vectors for word representation. In *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*, pages 1532–1543.

Emily Reif, Ann Yuan, Martin Wattenberg, Fernanda B Viegas, Andy Coenen, Adam Pearce, and Been Kim. 2019. Visualizing and measuring the geometry of bert. *Advances in Neural Information Processing Systems*, 32.

Nils Reimers, Iryna Gurevych, Nils Reimers, Iryna Gurevych, Nandan Thakur, Nils Reimers, Johannes Daxenberger, Iryna Gurevych, Nils Reimers, Iryna Gurevych, et al. 2019. Sentence-bert: Sentence embeddings using siamese bert-networks. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing*.

Richard Socher, Alex Perelygin, Jean Wu, Jason Chuang, Christopher D Manning, Andrew Y Ng, and Christopher Potts. 2013. Recursive deep models for semantic compositionality over a sentiment treebank. In *Proceedings of the 2013 conference on empirical methods in natural language processing*, pages 1631–1642.

Jianlin Su, Jiarun Cao, Weijie Liu, and Yangyiwen Ou. 2021. Whitening sentence representations for better semantics and faster retrieval. *arXiv preprint arXiv:2103.15316*.

Yonglong Tian, Chen Sun, Ben Poole, Dilip Krishnan, Cordelia Schmid, and Phillip Isola. 2020. What makes for good views for contrastive learning? In *Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020, December 6-12, 2020, virtual*.

N TISHBY. 1999. The information bottleneck method. In *Proc. 37th Annual Allerton Conference on Communications, Control and Computing, 1999*, pages 368–377.

Naftali Tishby and Noga Zaslavsky. 2015. Deep learning and the information bottleneck principle. In *2015 ieee information theory workshop (itw)*, pages 1–5. IEEE.

Trieu H Trinh and Quoc V Le. 2018. A simple method for commonsense reasoning. *arXiv preprint arXiv:1806.02847*.

Yao-Hung Hubert Tsai, Yue Wu, Ruslan Salakhutdinov, and Louis-Philippe Morency. 2020. Self-supervised learning from a multi-view perspective. In *International Conference on Learning Representations*.

Ellen M Voorhees and Dawn M Tice. 2000. Building a question answering test collection. In *Proceedings of the 23rd annual international ACM SIGIR conference on Research and development in information retrieval*, pages 200–207.

Tongzhou Wang and Phillip Isola. 2020. Understanding contrastive representation learning through alignment and uniformity on the hypersphere. In *International Conference on Machine Learning*, pages 9929–9939. PMLR.

Janyce Wiebe, Theresa Wilson, and Claire Cardie. 2005. Annotating expressions of opinions and emotions in language. *Language resources and evaluation*.

Zhuofeng Wu, Sinong Wang, Jiatao Gu, Madian Khabsa, Fei Sun, and Hao Ma. 2020. Clear: Contrastive learning for sentence representation. *arXiv preprint arXiv:2012.15466*.

Yan Zhang, Ruidan He, Zuozhu Liu, Kwan Hui Lim, and Lidong Bing. 2020. An unsupervised sentence embedding method by mutual information maximization. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1601–1610.

Jie Zhou, Yuanbin Wu, Qin Chen, Xuan-Jing Huang, and Liang He. 2021. Attending via both fine-tuning and compressing. In *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*, pages 2152–2161.

Yukun Zhu, Ryan Kiros, Rich Zemel, Ruslan Salakhutdinov, Raquel Urtasun, Antonio Torralba, and Sanja Fidler. 2015. Aligning books and movies: Towards story-like visual explanations by watching movies and reading books. In *Proceedings of the IEEE international conference on computer vision*, pages 19–27.

Learn2Weight: Parameter Adaptation against Similar-domain Adversarial Attacks

Siddhartha Datta
University of Oxford
siddhartha.datta@cs.ox.ac.uk

Abstract

Recent work in black-box adversarial attacks for NLP systems has attracted much attention. Prior black-box attacks assume that attackers can observe output labels from target models based on selected inputs. In this work, inspired by adversarial transferability, we propose a new type of black-box NLP adversarial attack that an attacker can choose a similar domain and transfer the adversarial examples to the target domain and cause poor performance in target model. Based on domain adaptation theory, we then propose a defensive strategy, called Learn2Weight, which trains to predict the weight adjustments for a target model in order to defend against an attack of similar-domain adversarial examples. Using Amazon multi-domain sentiment classification datasets, we empirically show that Learn2Weight is effective against the attack compared to standard black-box defense methods such as adversarial training and defensive distillation. This work contributes to the growing literature on machine learning safety.

1 Introduction

As machine learning models are applied to more and more real-world tasks, addressing machine learning safety is becoming an increasingly pressing issue. Deep learning algorithms have been shown to be vulnerable to adversarial examples (Szegedy et al., 2013; Goodfellow et al., 2014; Papernot et al., 2016a). In particular, prior black-box adversarial attacks assume that the adversary is not aware of the target model architecture, parameters or training data, but is capable of querying the target model with supplied inputs and obtaining the output predictions. The phenomenon that adversarial examples generated from one model may also be adversarial to another model is known as adversarial transferability (Szegedy et al., 2013).

Motivated by adversarial transferability, we conjecture another black-box attack pipeline where the adversary does not even need to have access to the target model nor query labels from crafted inputs. Instead, as long as the adversary knows the task of the target, they can choose a similar domain to build a substitute model, and then attack the target model with adversarial examples that are generated from the attack domain.

The similar-domain adversarial attack may be more practical than prior blackbox attacks as label querying from the target model is not needed. This attack can be illustrated with the following example (Figure 1b) in medical insurance fraud (Finlayson et al., 2019). Insurance companies may use hypothetical opioid risk models to classify the likelihood (high/low) of a patient to abuse the opioids to be prescribed, based on the patient's medical history as text input. Physicians can run the original patient history through the attack pipeline to generate an adversarial patient history, where the original is more likely to be rejected ("High" risk) and the adversarial is more likely to be accepted ("Low" risk). Perturbations in patient history could be, for example, a slight perturbation from "alcohol abuse" to "alcohol dependence", and it may successfully fool the insurance company's model.

Based on domain adaption theory (Ben-David et al., 2010), we conjecture that domain-variant features cause the success of the similar-domain attack. The adversarial examples with domain-variant features are likely to reside in the low-density regions (far away from decision boundary) of the empirical distribution of the target training data which could fool the target model (Zhang et al., 2019b). Literature indicates that worsened generalizability is a tradeoff faced by existing defenses such as adversarial training (Raghunathan et al., 2019) and domain generalization techniques (Wang et al., 2019). In trying to increase robustness against adversarial inputs, a model faces a tradeoff of weakened accuracy towards clean inputs. Given that an adversarial training loss function is composed of a loss against

Proceedings of the 29th International Conference on Computational Linguistics, pages 4832–4843
October 12–17, 2022.

Figure 1: Diagrammatic representation of the attack

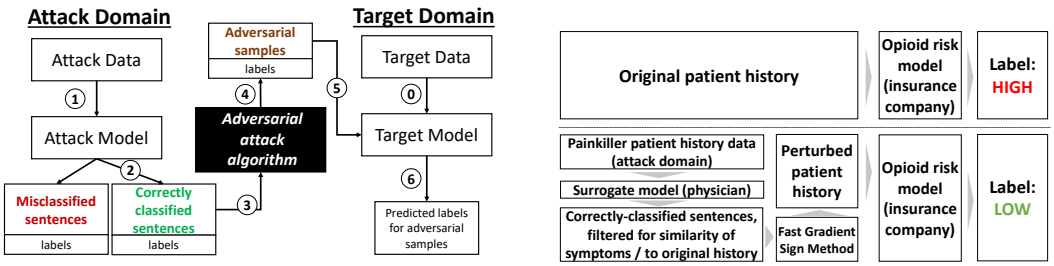

(a) Generalized architecture of similarity-based attacks.

(b) Flow of how an adversary physician can leverage similarity attack to fool opioid risk models.

clean inputs and loss against adversarial inputs, improper optimization where the latter is highly-optimized and the former weakly-optimized does not improve general performance in the real-world. To curb this issue, methods have been proposed (Schmidt et al., 2018; Zhang et al., 2019b; Lamb et al., 2019), such as factoring in under-represented data points in training set.

To defend against this similar-domain adversarial attack, we propose a meta learning approach, **Learn2Weight**, so that the target model's decision boundary can adapt to the examples from low-density regions. Experiments confirm the effectiveness of our approach against the similar-domain attack over other baseline defense methods. Moreover, our approach is able to improve robustness accuracy without losing the target model's standard generalization accuracy.

Our contribution can be summarized as follows [†]:

- We are among the first to demonstrate the similar-domain adversarial attack, leveraging domain adaptation to create adversarial perturbations that compromise NLP models. This attack pipeline relaxes the previous black-box attack assumption that the adversary has access to the target model and can query the model with crafted examples.
- We propose a defensive strategy for this attack based on domain adaptation theory and meta learning. Experiments show the effectiveness of our approach over existing defenses against the similar-domain adversarial attack.

[†] † indicates supplementary information can be found in the appendix (Appendix: Datta (2022)).

2 Related Work

Zhang et al. (2020) provides a survey of adversarial attacks in NLP. Existing research proposes different attack methods for generating adversarial text examples (Moosavi-Dezfooli et al., 2016; Ebrahimi et al., 2018; Wallace et al., 2019). The crafted adversarial text examples have been shown to fool state-of-the-art NLP systems, e.g. BERT (Jin et al., 2019). A large body of adversarial attack research focuses on black-box attack where the adversary builds a substitute model by querying the target model with supplied inputs and obtaining the output predictions. The key idea behind such black-box attack is that adversarial examples generated from one model may also be misclassified by another model, which is known as adversarial transferability (Szegedy et al., 2013; Cheng et al., 2019). While prior work examines the transferability between different models trained over the same dataset, or the transferability between the same or different models trained over disjoint subsets of a dataset, our work examines the adversarial transferability between different domains, which we call a similar-domain adversarial attack.

3 Similar-domain Adversarial Attack

3.1 Adversarial attack background

Adversarial attacks modify inputs to cause errors in machine learning inference (Szegedy et al., 2013). We use the basic gradient-based attack method *Fast Gradient Sign Method (FGSM)* (Goodfellow et al., 2014), with perturbation rate $\varepsilon = 0.4$. Other NLP adversarial generation algorithms could also be used, such as *Rand-FGSM* (Tramèr et al., 2017), *Basic Iterative Method* (Kurakin et al., 2016c,a; Xie et al., 2018), DeepFool (Moosavi-Dezfooli et al., 2016), HotFlip (Ebrahimi et al., 2018), uni-

Attack domain: baby, Target domain: books		
Original sentence (Actual label: Pos)	I purchased this toy for my son when he was 4 months old. At first, he seemed a little intimidated by the toys.	Pos (0.712)
Adversarial sentence	*I obtained this toys for my children when he was 4 weeks senior. At first, he hoped a modest harassed by the toy.*	Neg (0.364)
Original sentence (Actual label: Pos)	It felt like a big commitment for me to have to run the program 2 times a day, and near the end of my pregnancy I was annoyed with having anything strapped across my belly.	Pos (0.825)
Adversarial sentence	*It felt like a big committed for me to have to run the program 2 length a day, and near the end of my pregnancy I was annoyed with takes anything strapped across my belly.*	Neg (0.420)
Attack domain: dvd, Target domain: baby		
Original sentence (Actual label: Pos)	Fast times at ridgemont high is a clever, insightful, and wicked film! It is not just another teen movie.	Pos (0.614)
Adversarial sentence	*Sooner days at ridgemont high is a sane, thoughtful, and wicked flick! It is not just another adolescent flick.*	Neg (0.335)
Original sentence (Actual label: Pos)	This dvd gives a very good 60 minute workout. As others have pointed out the cardio is very dancy. The first time I did it, I felt a bit awkward with the steps.	Pos (0.647)
Adversarial sentence	*This dvd gives a awfully okay 60 minute exercise. As others have pointed out the cardio is very dancy. The first time I did it, I perceived a bit awkward with the steps.*	Neg (0.258)

Table 1: Comparison of attack domain sentences correctly classified when unperturbed by respective attack domain models and target domain models, then misclassified after perturbation by target models trained on **books** and **baby** domain. The perturbations are in blue, and prediction confidence in brackets.

versal adversarial trigger (Wallace et al., 2019), and TextFooler (Jin et al., 2019). To perform gradient-based perturbations upon discrete space data, we follow Papernot et al. (2016b) to generate adversarial text. Our proposed similar-domain adversarial attack is in-variant to adversarial algorithm, meaning that the adversarial algorithm used would not affect the attack performance.

Definition 1. NLP Adversarial Generation. We denote $\texttt{Adv}(\theta; \mathbf{x}; \varepsilon)$ as an NLP adversarial generation method. The goal of $\texttt{Adv}$ is to maximize the misclassification rate on perturbed inputs: $\mathbf{x}^{\text{adv}} = \texttt{Adv}(\theta; \mathbf{x})$ *s.t.* $y \neq f(\theta; \mathbf{x}^{\text{adv}})$.

3.2 Similar-domain Adversarial attack

We present the architecture of similar-domain adversarial attack in Figure 1a. The defender, the target of the attack, constructs a target model (parameters θ_i) trained on domain text data X_i ⓪. An attacker, only having a rough idea about the target's task but lacking direct access to the target data or target model parameters, collects attack data from a similar domain $X_j \sim \mathcal{X}$ and trains an attack model (parameters θ_j) ①. They run the attack model on the test data ② to obtain correctly-classified instances ③. They chooses an adversarial attack algorithm and generate a set of adversarial samples X_j^{adv} ④. They expose X_j^{adv} to the target model, hoping X_j^{adv} misleads the target model to produce an output of their choice ⑤. The attacker's objective is to maximize the misclassification per label and minimize the accuracy w.r.t. perturbed inputs (max Eqt 1), while the defender's objective is to maximize the accuracy w.r.t. perturbed inputs (min Eqt 1). This type of attack works best as an adversarial attack that compromises systems that base decision-making on one-instance.

$$\mathbb{E}_{\mathbf{x}_j, y_j \sim X_j, Y_j}[f(\theta_i; \texttt{Adv}(\theta_j; \mathbf{x}_j)) - y_j] \quad (1)$$

Definition 2. Similar-domain Adversarial Attack. Target model f, trained on target domain data X_i, is a deep neural network model with weights θ_i mapping text instances to labels: $Y_i = f(\theta_i; X_i)$. An adversary chooses source attack domain X_j, builds substitute model $f(\theta_j; X_j)$, and generates a set of adversarial examples X_j^{adv} from X_j using $\texttt{Adv}(\theta_j; X_j)$, such that during an attack $f(\theta_i; X_j^{\text{adv}}) = f(\theta_j; X_j^{\text{adv}})$.

4 Is the Attack Effective?

4.1 Setup

(Datasets) We sample domains from 25 domain datasets, each containing 1,000 positive and 1,000 negative reviews for an Amazon product category, sourced from the Amazon multi-domain sentiment classification benchmark (Blitzer et al., 2007).
(Models) We evaluated our setup on several architectures commonly-used for sentiment classification, including LSTM (Wang et al., 2018), GRU, BERT (Devlin et al., 2019), CNN (Kim, 2014), and Logistic Regression (Maas et al., 2011).

Target Domain	book			magazine			baby		
Original Accuracy	0.880			0.960			0.890		
Intra-attack Accuracy	0.525			0.570			0.632		
Attack Domain	magazine	baby	dvd	baby	dvd	book	dvd	book	magazine
Unperturbed Accuracy	0.745	0.726	0.646	0.673	0.663	0.739	0.652	0.624	0.665
After-attack Accuracy	0.395	0.398	0.421	0.343	0.366	0.381	0.386	0.365	0.401
SharedVocab	0.455	0.381	0.255	0.381	0.345	0.260	0.255	0.270	0.260
Transfer Loss	0.000	0.017	0.071	0.010	0.022	0.079	0.050	0.066	0.069

Table 2: *Domain shift & similarity:* Sorted in descending order of domain similarity, we observe a lower after-attack accuracy when domain similarity increases.

(Domain similarity) refers to the similarity between attacker's chosen domain and defender's domain. **SharedVocab** measures the overlap of unique words, in each of the datasets; a higher degree of overlapping vocabulary implies the two domains are more similar. We also use **Transfer Loss**, a standard metric for domain adaptation (Blitzer et al., 2007; Glorot et al., 2011), to measure domain similarity; lower loss indicates higher similarity. The test error from a target model trained on target domain X_i and evaluated on attack domain X_j returns transfer error $e(X_j, X_i)$. The baseline error $e(X_i, X_i)$ term is the test error obtained from target model trained on target domain (train) data X_i and tested on target domain (evaluation) data X_i. This computes the transfer loss, $tf(X_j, X_i) = e(X_j, X_i) - e(X_i, X_i)$.

(Accuracy) We first report the accuracy of the target models on the target domain test samples before the attack as the *original accuracy*. Then we measure the accuracy of the target models against adversarial samples crafted from the attack domain samples, denoted as the *after-attack accuracy*. *Intra-attack accuracy* denotes the after-attack accuracy where the attack domain is identical to the target domain. By comparing original and after-attack accuracy, we can evaluate the success of the attack. The greater the gap between the original and after-attack accuracy, the more successful the attack. *Unperturbed accuracy* measures the accuracy of the target model against the complete, unperturbed test set of the attack domain, to demonstrate that any drop in classification accuracy is not from domain shift alone but from adversarial transferability.

4.2 Results

The similar-domain adversarial attack results are presented in Table 2. We see a significant gap between original accuracy and after-attack accuracy, indicating that this attack can impose a valid threat to a target NLP system. After the similar-domain adversarial attack, the accuracy drops dramatically by a large margin. Take the book target domain as an example: when the attack domain is magazine, the after-attack accuracy drops to 0.398, and when the attack domain is baby, the accuracy is 0.421. Moreover, we observe a positive correlation between transfer loss and after-attack accuracy, and a negative correlation between shared vocab and after-attack accuracy.

5 Defending Against Similar-domain Adversarial Attack

In order to defend against a similarity based adversarial attack, it is critical to block adversarial transferability. Adversarial training is the most intuitive yet effective defense strategy for adversarial attack (Goodfellow et al., 2014; Madry et al., 2017). However, this may not be effective for two reasons. First, there is no formal guidance for generating similar-domain adversarial examples because the defender has no idea what the attack data domain is. Second, simply feeding the target model with adversarial examples may even hurt the generalization of the target model (Su et al., 2018; Raghunathan et al., 2019; Zhang et al., 2019a), which is also confirmed in our experiments.

5.1 Parameter Adaptation

Meta learning techniques that modify parameters (Ha et al., 2016; Hu et al., 2018; Kuen et al., 2019) are concerned with adapting weights from one model into another, and generating/predicting the complete set of weights for a model given the input samples. In our context, distinctly different

weights are produced for target models trained on inputs of different domains, and feature transferability (Yosinski et al., 2014) in the input space can be expected to translate to weights transferability in the parameter space. Rather than completely regenerating classification weights, our model robustification defense, *Learn2Weight*, predicts the perturbation to existing weights $\theta^* = \theta_i + \widehat{\Delta\theta}$ for each new instance.

5.2 Learn2Weight (L2W)†

We conjecture that an effective defense strategy is to perturb the target model weights depending on the feature distribution of the input instance. In inference (Algorithm 1), L2W recalculates the target model weights depending on the input. During training (Algorithm 2), L2W trains on sentences from different domains and a weight differential for that domain (the weight adjustment required to tune the target model's weights to adapt to the input's domain). We obtain the weight differential $\Delta\theta$ by finding the difference between the weights θ_j trained on sentence:label pairs from a specific domain $X_j \sim \mathcal{X}$ and weights θ_i trained on sentence:label pairs from the target domain X_i. Other training models may be possible; here we trained a sequence-to-sequence network (Sutskever et al., 2014) on sentence:$\Delta\theta$ pairs.

5.3 Perturbation Sets Generation†

To generate synthetic domains of varying domain similarity $\mathbf{S} = \{X_j : Y_j\}_{j=1}^{T}$ so that defenders defend their model using only target domain data X_i, a defender iteratively generates perturbation sets that minimizes transfer loss while maximizing adversarial perturbations (Algorithm 3). A *perturbation set* is a set containing subsets of perturbed inputs (Alzantot et al., 2018; Wong et al., 2019). To construct one perturbation set (Eqt 2), we utilize an iterative minimax algorithm, where we iteratively apply a maximizing adversarial perturbation factor $\varepsilon \geq \varepsilon_{\text{min}}$, and accept the batch of perturbed inputs if it yields a minimizing input distance $\texttt{dist} \leq d_{\text{max}}$. We repeat this T times. We use transfer loss as the distance metric to optimize for domain similarity. We retain FGSM as the adversarial attack algorithm.

$$
\begin{aligned}
X^* &:= \min \texttt{dist}(X^*, X_i) \leq d_{\text{max}} \\
X^* &:= \min \arg\max_{\varepsilon \sim [\varepsilon_{\text{min}}, 1]} \texttt{dist}(\texttt{Adv}(\theta_i; X_i; \varepsilon), X_i) \\
X^* &:= \min \arg\max_{\varepsilon \sim [\varepsilon_{\text{min}}, 1]} [e(\texttt{Adv}(\theta_i; X_i; \varepsilon), X_i) - e(X_i, X_i)]
\end{aligned}
\tag{2}
$$

Algorithm 1: Learn2Weight (Inference)

inference $(X_j^{\text{adv}}, \hbar(\theta^{m\ell}), \ell(\theta_i))$

Input : test-time inputs X_j^{adv}; L2W $\hbar(\theta^{m\ell})$; base learner $\ell(\theta_i)$

Output : label $\hat{y}$

Compute parameter differential w.r.t. X_j^{adv}.

$\widehat{\Delta\theta} \leftarrow \hbar(\theta^{m\ell}; X_j^{\text{adv}})$

Update θ^{ℓ}.

$\hat{y} \leftarrow \ell(\theta_i + \widehat{\Delta\theta}; X_j^{\text{adv}})$

return $\hat{y}$

Algorithm 2: Learn2Weight (Training)

train $(\mathbf{S}, \mathbf{D}, \theta_i, \mathbf{E}^{\ell}, \mathbf{E}^{m\ell})$

Input : domains (perturbation sets) $\mathbf{S}$, target domain $\mathbf{D} = \{X_i : Y_i\}$, base learner parameters θ_i, epochs $\mathbf{E}^{\ell}$ & $\mathbf{E}^{m\ell}$

Output : L2W parameters $\theta^{m\ell}$

Initialize empty set Θ to store parameter differential.

$\Theta \leftarrow \emptyset$;

Compute $X_j \mapsto \Delta\theta$.

foreach $X_j : Y_j \in (\mathbf{D} \cup \mathbf{S})$ **do**

  for $e \leftarrow 0$ **to** $\mathbf{E}^{\ell}$ **do**

    $\theta_{j,e}^{\ell} := \theta_{j,e-1}^{\ell} - \sum_{\mathbf{x},y}^{X_j,Y_j} \frac{\partial \mathcal{L}(\mathbf{x},y)}{\partial \theta^{\ell}}$

  $\Delta\theta \leftarrow \theta_j^{\ell} - \theta_i$

  $\Theta \leftarrow \Delta\theta$;

Compute $\theta^{m\ell}$.

for $e \leftarrow 0$ **to** $\mathbf{E}^{m\ell}$ **do**

  $\theta_e^{m\ell} := \theta_{e-1}^{m\ell} - \sum_{X_j,\Delta\theta}^{(X_i \cup \mathbf{S}),\Theta} \frac{\partial \mathcal{L}(X_j,\Delta\theta)}{\partial \theta^{m\ell}}$

return $\theta^{m\ell}$

Algorithm 3: Perturbation Sets Generation

PerturbationSet $(\mathbf{D}, \theta_i; T, R; \texttt{dist}, d_{\text{max}}; \varepsilon, \gamma)$

Input : target domain $\mathbf{D} = \{X_i : Y_i\}$, parameters θ_i; number of perturbation sets $T = 10$, max iterations $R = 10$; distance metric $\texttt{dist} = tf(X_i, X_j)$, max distance $d_{\text{max}} = 0.1$; initial perturbation rate $\varepsilon = 0.9$, perturbation learning rate $\gamma = 0.05$;

Output : set $\mathbf{S}$ containing T perturbation sets

Initialize empty $\mathbf{S}$ to store perturbation sets S_t.

$\mathbf{S} \leftarrow \emptyset$;

while $t < T$ **do**

  Run next iteration r until S_t meets conditions.

  for $r \leftarrow 0$ **to** R **do**

    Apply adversarial perturbations to X.

    $S_{t,r} \leftarrow \texttt{Adv}(\theta_i; X_i; \varepsilon)$;

    Evaluate distance conditions.

    if $\texttt{dist}(S_{t,r}, X_i) \leq d_{\text{max}}$ **then**

      if $\sigma^2(\mathbf{S} \cup S_{t,r}) > \sigma^2(\mathbf{S})$ **then**

        $\mathbf{S} \leftarrow \{S_{t,r} : Y_i\}$;

        continue;

    else

      Adjust hyperparameters.

      $\varepsilon \leftarrow \varepsilon - \gamma$;

  $t \leftarrow t + 1$;

return $\mathbf{S}$

5.4 Explanation: Blocking Transferability

To facilitate our explanation, we adapt from domain adaptation literature (Ben-David et al., 2010; Liu et al., 2019; Zhang et al., 2019c):

$$e(X_j^{\text{adv}}, X_i) \leq e(X_i, X_i) + d_{\mathcal{H}\Delta\mathcal{H}}(X_j^{\text{adv}}, X_i) + \lambda \quad (3)$$

where $\mathcal{H}$ is the hypothesis space, h is a hypothesis function that returns labels $\{0, 1\}$, and $e(X_i, X_i)$ and $e(X_j^{\text{adv}}, X_i)$ are the generalization errors from passing target domain data X_i and adversarial data X_j^{adv} through a classifier trained on X_i. $d_{\mathcal{H}\Delta\mathcal{H}}(X_j^{\text{adv}}, X_i)$ is the $\mathcal{H}\Delta\mathcal{H}$-distance between X_i and X_j^{adv}, and measures the divergence between the feature distributions of X_j^{adv} and X_i. $e_{X_j^{\text{adv}}}(h, h^{'})$ and $e_{X_i}(h, h^{'})$ represent the probability that h disagrees with $h^{'}$ on the label of an input in the domain space X_j^{adv} and X_i respectively.

$$\begin{aligned} d_{\mathcal{H}\Delta\mathcal{H}}(X_j^{\text{adv}}, X_i) &= \sup_{h,h^{'}\in\mathcal{H}} |e_{X_j^{\text{adv}}}(h, h^{'}) - e_{X_i}(h, h^{'})| \\ d_{\mathcal{H}\Delta\mathcal{H}}(X_j^{\text{adv}}, X_i) &= \sup_{h,h^{'}\in\mathcal{H}} \left|\mathbb{E}_{x_j\sim X_j}[|(h(x_j) - h^{'}(x_j)|]\right| \\ &\quad - \left|\mathbb{E}_{x_i\sim X_i}[|(h(x_i) - h^{'}(x_i)|]\right| \end{aligned} \quad (4)$$

Divergence $d_{\mathcal{H}\Delta\mathcal{H}}$ measures the divergence between feature distributions X_j^{adv} and X_i. Higher $d_{\mathcal{H}\Delta\mathcal{H}}$ indicates less shared features between 2 domains. The greater the intersection between feature distributions, the greater the proportion of domain-variant features; one approach to domain adaptation is learning domain-invariant features representations (Zhao et al., 2019) to minimize $d_{\mathcal{H}\Delta\mathcal{H}}$.

Explaining similarity-domain attacks. As demonstrated by empirical results, $e(X_j^{\text{adv}}, X_i)$ increases in a similarity-based attack setting, and this would arise if $d_{\mathcal{H}\Delta\mathcal{H}}$ increases correspondingly. $d_{\mathcal{H}\Delta\mathcal{H}}$ computes inconsistent labels from inconsistent feature distributions, and attributes the success of the attack to domain-variant features.

FGSM and variants adjust the input data to maximize the loss based on the backpropagated gradients of a model trained on X_j. As our pipeline used correctly-labelled sentences before adversarially perturbing them, we can infer that perturbations applied to X_j were not class-dependent (i.e. the success of the attack is not based on the removal of class-specific features), but class-independent features. It is already difficult for a model trained on X_j to classify when there is insufficient class-dependent features (hence a high $tf(X_j^{\text{adv}}, X_i)$); in a cross-domain setting, it must be even more difficult for a model trained on X_i to classify given a shortage of domain-invariant, class-dependent features.

$$\begin{aligned} d_{\mathcal{H}\Delta\mathcal{H}} &\geq e(X_j^{\text{adv}}, X_i) - e(X_i, X_i) - \lambda \\ d_{\mathcal{H}\Delta\mathcal{H}} &\geq tf(X_j^{\text{adv}}, X_i) - \lambda \end{aligned} \quad (5)$$

Explaining Learn2Weight. L2W minimizes divergence by training on $\{d_{\mathcal{H}\Delta\mathcal{H}}(X_j, X_i) : \Delta\theta\}$ pairs, such that $\Delta\theta = L2W(d_{\mathcal{H}\Delta\mathcal{H}}(X_j, X_i))$, where $d_{\mathcal{H}\Delta\mathcal{H}}(X_j, X_i)$ is reconstructed from the difference between X_j and X_i. The target model possesses a decision boundary (Liu et al., 2019) to classify inputs based on whether they cross the boundary or not; adversarial inputs have a tendency of being near the boundary and fooling it. Meta learning applies perturbations to the decision boundary such that the boundary covers certain adversarial inputs otherwise misclassified, and in this way blocks transferability. The advantage of training on multiple domains $\{X_j\}_{j=1}^{T}$ is that the after-L2W divergence between X_j^{adv} and X_i is smaller because L2W's weight perturbations render the decision boundary more precise in classifying inputs.

Explaining perturbation sets. We attributed why adversarial sentences X_j^{adv} are computed to be domain-dissimilar despite originating from X_j due to insufficient domain-invariant, class-dependent features resulting in low $e(X_j^{\text{adv}}, X_i)$, i.e. low $tf(X_j^{\text{adv}}, X_i)$. To replicate this phenomenon in natural domains, we iteratively perturb X_i to increase the proportion of class-independent features. This approximates the real-world similarity-based attack scenario where class-dependent features may be limited for inference. By generating the synthetic data, we are feeding L2W attack data with variations in $d_{\mathcal{H}\Delta\mathcal{H}}$ and class-independent feature distributions. This prepares L2W to robustify weights θ_i when such feature distributions are met.

Target Domain	magazine			baby		
Attack Domain	baby	dvd	book	dvd	book	magazine
After-attack Accuracy	0.381	0.366	0.343	0.365	0.386	0.401
After-defense Accuracy						
Adversarial Training	0.639	0.559	0.657	0.558	0.577	0.661
Defensive Distillation	0.549	0.561	0.597	0.588	0.629	0.577
Perturbation Sets Adversarial Training	0.608	0.637	0.620	0.604	0.620	0.587
Learn2Weight	**0.796**	**0.842**	**0.843**	**0.774**	**0.751**	**0.737**

Table 3: *After-defense Accuracy:* Learn2Weight outperforms the baseline and ablation methods.

Target Domain	Attack Domain	After-Attack Accuracy					After-Defense Accuracy				
		BERT	LSTM	GRU	CNN	LogReg	BERT	LSTM	GRU	CNN	LogReg
book	dvd	0.342	0.413	0.477	0.335	0.440	0.786	0.847	0.804	0.816	0.782
	kitchenware	0.350	0.372	0.325	0.353	0.425	0.765	0.826	0.795	0.742	0.767
	electronics	0.400	0.389	0.416	0.315	0.460	0.792	0.812	0.784	0.770	0.725
dvd	book	0.326	0.434	0.479	0.383	0.490	0.816	0.795	0.824	0.804	0.794
	kitchenware	0.355	0.370	0.379	0.359	0.490	0.728	0.796	0.755	0.735	0.695
	electronics	0.387	0.377	0.332	0.348	0.455	0.825	0.836	0.812	0.834	0.796
electronics	book	0.425	0.394	0.473	0.358	0.474	0.775	0.821	0.795	0.782	0.712
	dvd	0.342	0.395	0.452	0.368	0.493	0.784	0.845	0.855	0.842	0.792
	kitchenware	0.390	0.384	0.464	0.329	0.432	0.730	0.824	0.753	0.724	0.678

Table 4: *Models:* L2W retains high after-defense accuracy at varying attack model architectures.

6 Experiments†

6.1 Baselines

Defensive distillation (Papernot et al., 2016c, 2017): The high-level implementation of *defensive distillation* is to first train an initial model against target domain inputs and labels, and retrieve the raw class probability scores. The predicted probability values would be used as the new labels for the same target sentences, and we would train a new model based on this new label-sentence pair.

Adversarial training (Goodfellow et al., 2014; Madry et al., 2017): It is shown that injecting adversarial examples throughout training increases the robustness of target neural network models. In this baseline, target model is trained with both original training data and adversarial examples generated from original training data. However, since the adversarial examples are still generated from the target domain, it is unlikely that the method can defend against a similar-domain adversarial attack, which is the result of domain-variant features.

Perturbation sets adversarial training: This ablation baseline tests for incremental performance to a baseline defense using domain-variant inputs. We adapt adversarial training to be trained on perturbation sets (synthetic domains) generated with Algorithm 3 with respect to target domain X_i.

6.2 Learn2Weight Performance

Defense performance. We present the results of different defense baselines in Table 3. First, we can see that L2W achieves the highest after-defense accuracy against the adversarial attack. Take the *magazine* as target domain for example: if the adversary chooses to use *book* data as the attack domain, it would reduce the target model accuracy to 0.343. However, L2W can improve the performance to 0.843, which is a significant and substantial improvement against the attack. This improvement also exist across different target/attack domain pairs. Second, we see that all defense methods can improve the accuracy to some extent which indicates the importance and effectiveness of having robust training for machine learning models.

Attack model architectures. So far, all the results are conducted using the same LSTM as the target/attack model. Here, we keep the target model unchanged, but vary the architecture of the attack model for the generation of adversarial examples. LSTM (GRU) is configured with 64 cells, tokens embedded with respect to GloVe, `sigmoid` (`tanh`) activation function, randomly-initialized and trained with Adam optimizer and 80% (60%) dropout, based on Wang et al. (2018). CNN is configured with accepting tokens embedded with respect to GloVe (Pennington et al., 2014), 3 convo-

lutional layers with kernel widths of 3, 4, and 5, all with 100 output channels, and randomly-initialized, based on Kim (2014). We configure Logistic Regression based on Maas et al. (2011). Based on Devlin et al. (2019), we initialize a pretrained BERT with its own embeddings. Models are trained until reaching state-of-the-art validation accuracy (early-stopping pauses training at loss 0.5).

We present the results of different attack model architectures in Table 4. First, the similar-domain adversarial attack is model-agnostic and it does not require the target and attack model to have identical architectures. We can see that all four attack model architectures are able to reduce the target model accuracy. Second, the results suggest that L2W is also model-agnostic as it can substantially improve the after-defense accuracy regardless which attack model is used.

7 Conclusion

In this newly-proposed, empirically-effective similar-domain adversarial attack, an adversary can choose a similar domain to the target task, build a substitute model and produce adversarial examples to fool the target model. We also propose a defense strategy, Learn2Weight, that learns to adapt the target model's weight using crafted adversarial examples. Compared with other adversarial defense strategies, Learn2Weight can improve the target model robustness against the similar-domain attack. Our method demonstrates properties of a good adversarial defense, such as adopting a defense architecture that adapts to situations/inputs rather than compromising standard error versus robustness error, to leverage class-independent properties in domain-variant text, and factoring in domain similarity in adversarial robustness.

References

Moustafa Alzantot, Yash Sharma, Ahmed Elgohary, Bo-Jhang Ho, Mani Srivastava, and Kai-Wei Chang. 2018. Generating natural language adversarial examples. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 2890–2896, Brussels, Belgium. Association for Computational Linguistics.

Shai Ben-David, John Blitzer, Koby Crammer, Alex Kulesza, Fernando Pereira, and Jennifer Vaughan. 2010. A theory of learning from different domains. *Machine Learning*, 79:151–175.

John Blitzer, Mark Dredze, and Fernando Pereira. 2007. Biographies, bollywood, boom-boxes and blenders: Domain adaptation for sentiment classification. In *ACL*, pages 440–447.

Karsten M. Borgwardt, Arthur Gretton, Malte J. Rasch, Hans-Peter Kriegel, Bernhard Schölkopf, and Alex J. Smola. 2006. Integrating structured biological data by Kernel Maximum Mean Discrepancy. *Bioinformatics*, 22(14):e49–e57.

Shuyu Cheng, Yinpeng Dong, Tianyu Pang, Hang Su, and Jun Zhu. 2019. Improving black-box adversarial attacks with a transfer-based prior. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett, editors, *Advances in Neural Information Processing Systems 32*, pages 10934–10944. Curran Associates, Inc.

Xia Cui, Frans Coenen, and Danushka Bollegala. 2017. Tsp: Learning task-specific pivots for unsupervised domain adaptation. In *Machine Learning and Knowledge Discovery in Databases*, pages 754–771, Cham. Springer International Publishing.

Siddhartha Datta. 2022. Learn2weight: Parameter adaptation against similar-domain adversarial attacks.

Siddhartha Datta and Nigel Shadbolt. 2022a. Backdoors stuck at the frontdoor: Multi-agent backdoor attacks that backfire. In *International Conference on Learning Representations Workshop: Gamification and Multiagent Solutions*.

Siddhartha Datta and Nigel Shadbolt. 2022b. Interpolating compressed parameter subspaces.

Siddhartha Datta and Nigel Shadbolt. 2022c. Low-loss subspace compression for clean gains against multi-agent backdoor attacks. *arXiv preprint arXiv:2203.03692*.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, Minneapolis, Minnesota. Association for Computational Linguistics.

Javid Ebrahimi, Anyi Rao, Daniel Lowd, and Dejing Dou. 2018. Hotflip: White-box adversarial examples for text classification. In *ACL*, pages 31–36.

Samuel G. Finlayson, John D. Bowers, Joichi Ito, Jonathan L. Zittrain, Andrew L. Beam, and Isaac S. Kohane. 2019. Adversarial attacks on medical machine learning. *Science*, 363(6433):1287–1289.

Chelsea Finn, Pieter Abbeel, and Sergey Levine. 2017. Model-agnostic meta-learning for fast adaptation of deep networks.

Tomer Galanti and Lior Wolf. 2020. On the modularity of hypernetworks.

Yaroslav Ganin, Evgeniya Ustinova, Hana Ajakan, Pascal Germain, Hugo Larochelle, François Laviolette, Mario Marchand, and Victor Lempitsky. 2016. Domain-adversarial training of neural networks. *J. Mach. Learn. Res.*, 17(1):2096–2030.

Robert Geirhos, Patricia Rubisch, Claudio Michaelis, Matthias Bethge, Felix A. Wichmann, and Wieland Brendel. 2019. Imagenet-trained CNNs are biased towards texture; increasing shape bias improves accuracy and robustness. In *International Conference on Learning Representations*.

Xavier Glorot, Antoine Bordes, and Yoshua Bengio. 2011. Domain adaptation for large-scale sentiment classification: A deep learning approach. In *ICML*, pages 513–520.

Ian J Goodfellow, Jonathon Shlens, and Christian Szegedy. 2014. Explaining and harnessing adversarial examples. *arXiv preprint arXiv:1412.6572*.

David Ha, Andrew Dai, and Quoc V Le. 2016. Hypernetworks. *arXiv preprint arXiv:1609.09106*.

Marton Havasi, Rodolphe Jenatton, Stanislav Fort, Jeremiah Zhe Liu, Jasper Snoek, Balaji Lakshminarayanan, Andrew M. Dai, and Dustin Tran. 2021. Training independent subnetworks for robust prediction.

Dan Hendrycks and Kevin Gimpel. 2016. A baseline for detecting misclassified and out-of-distribution examples in neural networks.

Ronghang Hu, Piotr Dollár, Kaiming He, Trevor Darrell, and Ross Girshick. 2018. Learning to segment every thing. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 4233–4241.

Robin Jia, Aditi Raghunathan, Kerem Göksel, and Percy Liang. 2019. Certified robustness to adversarial word substitutions. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 4129–4142, Hong Kong, China. Association for Computational Linguistics.

Di Jin, Zhijing Jin, Joey Tianyi Zhou, and Peter Szolovits. 2019. Is bert really robust? natural language attack on text classification and entailment. *arXiv preprint arXiv:1907.11932*.

Yoon Kim. 2014. Convolutional neural networks for sentence classification. In *EMNLP*, pages 1746–1751.

Simon Kornblith, Mohammad Norouzi, Honglak Lee, and Geoffrey Hinton. 2019. Similarity of neural network representations revisited. In *Proceedings of the 36th International Conference on Machine Learning*, volume 97 of *Proceedings of Machine Learning Research*, pages 3519–3529. PMLR.

Jason Kuen, Federico Perazzi, Zhe Lin, Jianming Zhang, and Yap-Peng Tan. 2019. Scaling object detection by transferring classification weights. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 6044–6053.

Alexey Kurakin, Ian Goodfellow, and Samy Bengio. 2016a. Adversarial examples in the physical world. *arXiv preprint arXiv:1607.02533*.

Alexey Kurakin, Ian Goodfellow, and Samy Bengio. 2016b. Adversarial examples in the physical world.

Alexey Kurakin, Ian Goodfellow, and Samy Bengio. 2016c. Adversarial machine learning at scale. *arXiv preprint arXiv:1611.01236*.

Matt Kusner, Yu Sun, Nicholas Kolkin, and Kilian Weinberger. 2015. From word embeddings to document distances. In *Proceedings of the 32nd International Conference on Machine Learning*, volume 37 of *Proceedings of Machine Learning Research*, pages 957–966, Lille, France. PMLR.

Alex Lamb, Vikas Verma, Juho Kannala, and Yoshua Bengio. 2019. Interpolated adversarial training: Achieving robust neural networks without sacrificing too much accuracy. In *Proceedings of the 12th ACM Workshop on Artificial Intelligence and Security*, AISec'19, page 95–103, New York, NY, USA. Association for Computing Machinery.

Fangzhou Liao, Ming Liang, Yinpeng Dong, Tianyu Pang, Xiaolin Hu, and Jun Zhu. 2018. Defense against adversarial attacks using high-level representation guided denoiser. In *2018 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2018, Salt Lake City, UT, USA, June 18-22, 2018*, pages 1778–1787. Computer Vision Foundation / IEEE Computer Society.

Etai Littwin, Tomer Galanti, Lior Wolf, and Greg Yang. 2020. On infinite-width hypernetworks. *Advances in Neural Information Processing Systems*, 2020-December. Publisher Copyright: © 2020 Neural information processing systems foundation. All rights reserved.; null ; Conference date: 06-12-2020 Through 12-12-2020.

Guanxiong Liu, Issa Khalil, Abdallah Khreishah, and NhatHai Phan. 2021. A synergetic attack against neural network classifiers combining backdoor and adversarial examples.

Hong Liu, Mingsheng Long, Jianmin Wang, and Michael Jordan. 2019. Transferable adversarial training: A general approach to adapting deep classifiers. In *Proceedings of the 36th International Conference on Machine Learning*, volume 97 of *Proceedings of Machine Learning Research*, pages 4013–4022, Long Beach, California, USA. PMLR.

Andrew L. Maas, Raymond E. Daly, Peter T. Pham, Dan Huang, Andrew Y. Ng, and Christopher Potts. 2011. Learning word vectors for sentiment analysis. In *ACL*, pages 142–150.

Aleksander Madry, Aleksandar Makelov, Ludwig Schmidt, Dimitris Tsipras, and Adrian Vladu. 2017. Towards deep learning models resistant to adversarial attacks. *arXiv preprint arXiv:1706.06083*.

Seyed-Mohsen Moosavi-Dezfooli, Alhussein Fawzi, and Pascal Frossard. 2016. Deepfool: a simple and accurate method to fool deep neural networks. In *CVPR*, pages 2574–2582.

Muzammal Naseer, Salman H. Khan, Harris Khan, Fahad Shahbaz Khan, and Fatih Porikli. 2019. Cross-domain transferability of adversarial perturbations.

Behnam Neyshabur, Hanie Sedghi, and Chiyuan Zhang. 2020. What is being transferred in transfer learning? In *Advances in Neural Information Processing Systems*, volume 33, pages 512–523. Curran Associates, Inc.

Alex Nichol, Joshua Achiam, and John Schulman. 2018. On first-order meta-learning algorithms.

Nicolas Papernot, Patrick McDaniel, and Ian Goodfellow. 2016a. Transferability in machine learning: from phenomena to black-box attacks using adversarial samples. *arXiv preprint arXiv:1605.07277*.

Nicolas Papernot, Patrick McDaniel, Ian Goodfellow, Somesh Jha, Z. Berkay Celik, and Ananthram Swami. 2017. Practical black-box attacks against machine learning. In *Proceedings of the 2017 ACM on Asia Conference on Computer and Communications Security*, ASIA CCS '17, page 506–519, New York, NY, USA. Association for Computing Machinery.

Nicolas Papernot, Patrick McDaniel, Ananthram Swami, and Richard Harang. 2016b. Crafting adversarial input sequences for recurrent neural networks. In *MILCOM 2016 - 2016 IEEE Military Communications Conference*, page 49–54. IEEE Press.

Nicolas Papernot, Patrick McDaniel, Xi Wu, Somesh Jha, and Ananthram Swami. 2016c. Distillation as a defense to adversarial perturbations against deep neural networks. In *2016 IEEE Symposium on Security and Privacy (SP)*, pages 582–597. IEEE.

Jeffrey Pennington, Richard Socher, and Christopher D. Manning. 2014. Glove: Global vectors for word representation. In *Empirical Methods in Natural Language Processing (EMNLP)*, pages 1532–1543.

Fanchao Qi, Yangyi Chen, Xurui Zhang, Mukai Li, Zhiyuan Liu, and Maosong Sun. 2021. Mind the style of text! adversarial and backdoor attacks based on text style transfer.

Aditi Raghunathan, Sang Michael Xie, Fanny Yang, John C Duchi, and Percy Liang. 2019. Adversarial training can hurt generalization. *arXiv preprint arXiv:1906.06032*.

Alexandre Ramé, Rémy Sun, and Matthieu Cord. 2021. Mixmo: Mixing multiple inputs for multiple outputs via deep subnetworks. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 823–833.

Matthew Riemer, Ignacio Cases, Robert Ajemian, Miao Liu, Irina Rish, Yuhai Tu, and Gerald Tesauro. 2019. Learning to learn without forgetting by maximizing transfer and minimizing interference.

Evgenia Rusak, Lukas Schott, Roland S. Zimmermann, Julian Bitterwolf, Oliver Bringmann, Matthias Bethge, and Wieland Brendel. 2020. A simple way to make neural networks robust against diverse image corruptions. In *Computer Vision – ECCV 2020*, pages 53–69, Cham. Springer International Publishing.

Shiori Sagawa*, Pang Wei Koh*, Tatsunori B. Hashimoto, and Percy Liang. 2020. Distributionally robust neural networks. In *International Conference on Learning Representations*.

Shibani Santurkar, Dimitris Tsipras, and Aleksander Madry. 2020. Breeds: Benchmarks for subpopulation shift.

Ludwig Schmidt, Shibani Santurkar, Dimitris Tsipras, Kunal Talwar, and Aleksander Madry. 2018. Adversarially robust generalization requires more data. In *Proceedings of the 32nd International Conference on Neural Information Processing Systems*, NIPS'18, page 5019–5031, Red Hook, NY, USA. Curran Associates Inc.

Aman Sinha, Hongseok Namkoong, and John Duchi. 2018. Certifiable distributional robustness with principled adversarial training. In *International Conference on Learning Representations*.

Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. 2014. Dropout: A simple way to prevent neural networks from overfitting. *Journal of Machine Learning Research*, 15(56):1929–1958.

Dong Su, Huan Zhang, Hongge Chen, Jinfeng Yi, Pin-Yu Chen, and Yupeng Gao. 2018. Is robustness the cost of accuracy?–a comprehensive study on the robustness of 18 deep image classification models. In *Proceedings of ECCV*, pages 631–648.

Ilya Sutskever, Oriol Vinyals, and Quoc V. Le. 2014. Sequence to sequence learning with neural networks. In *Proceedings of the 27th International Conference on Neural Information Processing Systems - Volume 2*, NIPS'14, page 3104–3112, Cambridge, MA, USA. MIT Press.

Christian Szegedy, Wojciech Zaremba, Ilya Sutskever, Joan Bruna, Dumitru Erhan, Ian Goodfellow, and Rob Fergus. 2013. Intriguing properties of neural networks. *arXiv preprint arXiv:1312.6199*.

Florian Tramèr, Alexey Kurakin, Nicolas Papernot, Ian Goodfellow, Dan Boneh, and Patrick McDaniel. 2017. Ensemble adversarial training: Attacks and defenses. *arXiv preprint arXiv:1705.07204*.

Johannes von Oswald, Christian Henning, João Sacramento, and Benjamin F. Grewe. 2020. Continual learning with hypernetworks.

Eric Wallace, Shi Feng, Nikhil Kandpal, Matt Gardner, and Sameer Singh. 2019. Universal adversarial triggers for attacking and analyzing nlp. In *EMNLP*, pages 2153–2162.

Haohan Wang, Songwei Ge, Zachary Lipton, and Eric Xing. 2019. Learning robust global representations by penalizing local predictive power. *Neural Information Processing Systems*, pages 10506–10518.

Jenq-Haur Wang, Ting-Wei Liu, Xiong Luo, and Long Wang. 2018. An LSTM approach to short text sentiment classification with word embeddings. In *ROCLING*, pages 214–223.

Zixin Wen and Yuanzhi Li. 2021. Toward understanding the feature learning process of self-supervised contrastive learning. In *Proceedings of the 38th International Conference on Machine Learning*, volume 139 of *Proceedings of Machine Learning Research*, pages 11112–11122. PMLR.

Eric Wong, Frank R. Schmidt, and J. Zico Kolter. 2019. Wasserstein adversarial examples via projected sinkhorn iterations. In *Proceedings of the 36th International Conference on Machine Learning, ICML 2019, 9-15 June 2019, Long Beach, California, USA*, volume 97 of *Proceedings of Machine Learning Research*, pages 6808–6817. PMLR.

Mitchell Wortsman, Maxwell Horton, Carlos Guestrin, Ali Farhadi, and Mohammad Rastegari. 2021. Learning neural network subspaces.

Mitchell Wortsman, Gabriel Ilharco, Samir Yitzhak Gadre, Rebecca Roelofs, Raphael Gontijo-Lopes, Ari S. Morcos, Hongseok Namkoong, Ali Farhadi, Yair Carmon, Simon Kornblith, and Ludwig Schmidt. 2022. Model soups: averaging weights of multiple fine-tuned models improves accuracy without increasing inference time.

Cihang Xie, Zhishuai Zhang, Yuyin Zhou, Song Bai, Jianyu Wang, Zhou Ren, and Alan Yuille. 2018. Improving transferability of adversarial examples with input diversity.

Jason Yosinski, Jeff Clune, Yoshua Bengio, and Hod Lipson. 2014. How transferable are features in deep neural networks? In *Proceedings of the 27th International Conference on Neural Information Processing Systems - Volume 2*, NIPS'14, page 3320–3328, Cambridge, MA, USA. MIT Press.

Sangdoo Yun, Dongyoon Han, Seong Joon Oh, Sanghyuk Chun, Junsuk Choe, and Youngjoon Yoo. 2019. Cutmix: Regularization strategy to train strong classifiers with localizable features.

Chiyuan Zhang, Samy Bengio, Moritz Hardt, Benjamin Recht, and Oriol Vinyals. 2017. Understanding deep learning requires rethinking generalization.

Hongyang Zhang, Yaodong Yu, Jiantao Jiao, Eric Xing, Laurent El Ghaoui, and Michael Jordan. 2019a. Theoretically principled trade-off between robustness and accuracy. In *Proceedings of ICML*, pages 7472–7482.

Huan Zhang, Hongge Chen, Zhao Song, Duane Boning, Inderjit S. Dhillon, and Cho-Jui Hsieh. 2019b. The limitations of adversarial training and the blind-spot attack.

Wei Emma Zhang, Quan Z Sheng, Ahoud Alhazmi, and Chenliang Li. 2020. Adversarial attacks on deep-learning models in natural language processing: A survey. *ACM Transactions on Intelligent Systems and Technology (TIST)*, 11(3):1–41.

Yuchen Zhang, Tianle Liu, Mingsheng Long, and Michael Jordan. 2019c. Bridging theory and algorithm for domain adaptation. In *Proceedings of the 36th International Conference on Machine Learning*, volume 97 of *Proceedings of Machine Learning Research*, pages 7404–7413, Long Beach, California, USA. PMLR.

Han Zhao, Remi Tachet Des Combes, Kun Zhang, and Geoffrey Gordon. 2019. On learning invariant representations for domain adaptation. In *Proceedings of the 36th International Conference on Machine Learning*, volume 97 of *Proceedings of Machine Learning Research*, pages 7523–7532, Long Beach, California, USA. PMLR.

Yftah Ziser and Roi Reichart. 2019. Task refinement learning for improved accuracy and stability of unsupervised domain adaptation. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 5895–5906, Florence, Italy. Association for Computational Linguistics.

Sentence-aware Adversarial Meta-Learning for Few-Shot Text Classification

Suhe Wang , Xiaoyuan Liu , Bo Liu , Diwen Dong
College of Computer
National University of Defense Technology
Changsha, China
wangsuhe16,xyliu,KyLe.liu,ddw_bak@nudt.edu.cn

Abstract

Meta-learning has emerged as an effective approach for few-shot text classification. However, current studies fail to realize the importance of the semantic interaction between sentence features and neglect to enhance the generalization ability of the model to new tasks. In this paper, we integrate an adversarial network architecture into the meta-learning system and leverage cost-effective modules to build a novel few-shot classification framework named SaAML. Significantly, our approach can exploit the temporal convolutional network to encourage more discriminative representation learning and explore the attention mechanism to promote more comprehensive feature expression, thus resulting in better adaptation for new classes. Through a series of experiments on four benchmark datasets, we demonstrate that our new framework acquires considerable superiority over state-of-the-art methods in all datasets, increasing the performance of 1-shot classification and 5-shot classification by 7.15% and 2.89%, respectively.

1 Introduction

Deep learning usually relies on a vast amount of labeled examples to accomplish tasks. However, this requirement is problematic for few-shot text classification when only a few examples are available in novel classes, which leads to poor model generalization for new tasks. Motivated by the fact that humans can quickly recognize new knowledge after learning a few examples, few-shot learning is becoming a hot research topic.

Early studies exploit data augmentation and regularization procedures (Salamon and Bello, 2017) to deal with the overfitting due to data sparseness. More recent research mainly falls into two approaches: (1) transfer-learning based methods (Pan et al., 2019; Gupta et al., 2020), which transfer and propagate knowledge attained from the source domain to classify unseen examples in the target domain. (2) meta-learning based methods (Tong, 2019; Sun et al., 2019), which learn knowledge by repeating lots of meta-tasks in a training episode manner and leverage knowledge extracted to briskly predict new samples during meta-testing. Specifically, Bao et al. (Bao et al., 2019) incorporated distributional signatures of words into the meta-learning framework to make impressive performance. Han et al. (Han et al., 2021) first proposed to introduce an adversarial domain adaptation network to strengthen the generalization ability of the meta-learning system.

Despite the remarkable progress of few-shot classification approaches (Geng et al., 2019; Bao et al., 2019; Han et al., 2021), most existing meta-learning models still suffer tough challenges: example diversity. Even examples in the same class have various representations, bringing about the difficulty of extracting generic features based on a few training examples and the urgent demand for strong adaptability of the model to new tasks.

In this paper, we propose a straightforward but remarkably powerful framework to cope with the above challenges. We deploy an adversarial network architecture to train the meta-learning system and create a novel framework Sentence-aware Adversarial Meta-Learner (SaAML). Specifically, we employ the temporal convolutional network (TCN) (Bai et al., 2018) and the multilayer perceptron network (MLP) to build the generator and discriminator, respectively. The model can grasp sentences' inherent semantic information through the adversarial training of the generator and the discriminator. Then we artfully fuse the word embeddings and the features of the generator to construct high-quality discriminative sentence features. Moreover, we further build a feature enhancer (FE) to leverage the multi-head attention mechanism to fine-tune the features of support examples and query examples, aiming to create more compatible feature representations. Our research methodically develops how to

Proceedings of the 29th International Conference on Computational Linguistics, pages 4844–4852
October 12–17, 2022.

better exploit the adversarial network architecture and sentence-aware interaction knowledge to boost few-shot text classification performance. The main innovations of our research are as follows:

1) We analyze the limitation to the performance of current meta-learning approaches and establish a novel system SaAML based on the adversarial network architecture. And we combine it with various meta-classification techniques to handle the few-shot classification dilemma.

2) We build the meta-learning system with more cost-effective modules, .e.g., the temporal convolutional network (TCN), the semantic extractor (SE), and the feature enhancer (FE). Then we jointly train the whole model in an end-to-end fashion.

3) To evaluate the effectiveness and robustness of the proposed model, we conduct massive comparison experiments and ablation studies on four datasets. The results indicate that our new framework SaAML outperforms state-of-the-art approaches with nearly 5.02% performance improvement, and the proposed enhancement modules are more beneficial and flexible.

2 Related Work

Few-shot text classification is a crucial application scenario of few-shot learning in natural language processing (NLP), which has obtained increasing attention and research. Transfer learning (Gupta et al., 2020), as a feasible approach to tackle few-shot text classification, intends to make the knowledge learned from the source domain more compatible with the target domain and reduce the shift between different domains. For example, Tzeng et al. (Tzeng et al., 2017) leverages the adversarial domain adaptation framework to bridge the domain gaps without example constraints. Moreover, with the rapid development of pre-trained language models, it also exhibits excellent performance by fine-tuning the representations of training examples, such as BERT (Devlin et al., 2018) and GPT-3 (Brown et al., 2020).

As the dominant program in few-shot text classification, meta-learning (Schmidhuber, 1987) systems develop rapidly and achieve great success. Meta-learning approaches are mainly divided into two categories: (1) Optimization-based methods, which resort to "learning to fine-tune" strategy to train the model. For example, (Finn et al., 2017; Lee et al., 2019; Rajeswaran et al., 2019) develop an optimization procedure of model parameter initialization to quickly obtain outstanding performance after a small amount of gradient update steps based on few-shot training examples.(2) Metric-based methods, which complete the classification task through a specific distance metric, .e.g., Matching Network (Vinyals et al., 2016) is calculated by the cosine similarity, while the Euclidean distance is the measure standard of Prototypical Network (Snell et al., 2017). Relational Network (Sung et al., 2018) and GNN Network (Yang et al., 2020) utilize convolutional neural networks and graph neural networks to learn metric functions dynamically. Recently, Bao et al. (Bao et al., 2019) argues that statistical information of sentences plays a vital role in classification tasks. Sun et al. (Sun et al., 2021) subtly combines data augmentation with meta-learning to generate more diverse samples, preventing model overfitting. Han et al. (Han et al., 2021) is the first to explore the adversarial domain adaptation network for the performance improvement of meta-learning framework.

3 Preliminary

The few-shot text classification is usually viewed as a N-way K-shot task. Firstly, the input of the model is a set with multiple labeled examples, including support set C_{train} and query set C_{test}, where C_{train} are disjoint from C_{test}. Then the meta-learner conducts the episode-based strategy (Vinyals et al., 2016) to train a classifier on C_{train}. Finally, the meta-learner accomplishes the goal to predict new examples of C_{test} with the classifier.

For the episode-based strategy, we randomly select N classes (N-way) from C_{train}, and then separately sample K examples (K-shot) as the support set S and P examples as the query set Q from each of these chosen classes, which can be denoted as:

$$\begin{aligned} S &= \{(X_i, Y_i)\}_{i=1}^{N \times K} \\ Q &= \{(X_j, Y_j)\}_{j=1}^{N \times P} \end{aligned} \tag{1}$$

where X is the input text sentence and Y is the corresponding label. It is worth noting that the same example sampling manner is implemented to build the support set S and query set Q during meta-training and meta-testing. We leverage the macro-averaged accuracy across all testing episodes to evaluate the performance of the meta-learner.

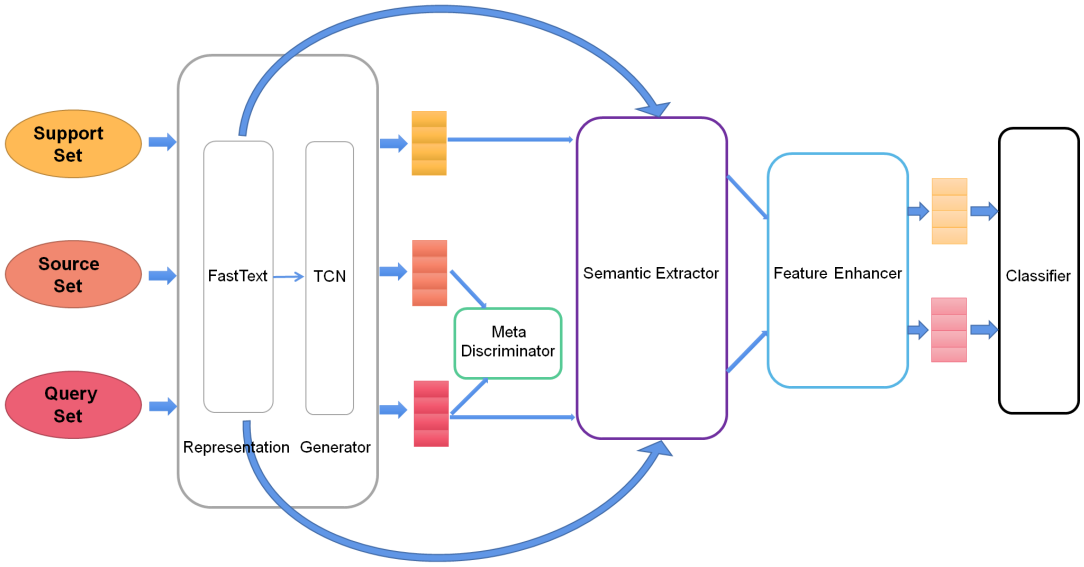

Figure 1: The overall framework of SaAML

4 Our Approach

Our proposed SaAML is broadly built to pursue the excellent performance of few-shot text classification. The overall architecture of SaAML is shown in Figure 1. First, we leverage the representation generator with TCN as the core block to capture comprehensive semantic information of sentences and exploit the meta discriminator to strengthen the learning ability of the model. Then, we contrive a feature enhancer to further fine-tune the example vectors with sentence-aware knowledge for more consistent representations. Finally, we jointly train all modules of SaAML with different schemes in an end-to-end manner.

4.1 Representation Generator

The representation generator consists of a word embedding encoder and a temporal convolutional network (TCN) (Bai et al., 2018), as shown in Figure 1.

The word embedding encoder converts each word into the embedding vector. We construct the embedding vector $\boldsymbol{v}$ with d dimensions via fastText (Joulin et al., 2016).

The goal of the temporal convolutional network (TCN) (Bai et al., 2018) is to acquire more transferable feature information. TCN architecture is very simple yet effective, covering some of the best techniques of current convolutional networks, such as 1D full convolution, causal convolution, and dilated convolution, as shown in Figure 2. To be specific, we exploit the 1D full convolution to make the model's output with the same length as the input. We adopt the causal convolution to ensure that the current result is only convolved with inputs from now to earlier, with no "leakage" of knowledge from the future to the past. Significantly, we also utilize the dilated convolution to expand the receptive field of the network and encourage the result of richer semantic information.

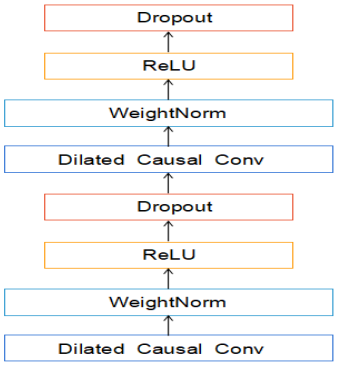

Figure 2: The convolutional layer structure of TCN.

Given that the input is a sequence of word vectors $\boldsymbol{V} = [\boldsymbol{v}_1, \boldsymbol{v}_2, \cdots, \boldsymbol{v}_m]$, where m is the number of words in the sentence. The output of TCN is a matrix $\boldsymbol{h}_{d\times m} = [\boldsymbol{h}_1, \boldsymbol{h}_2, \cdots, \boldsymbol{h}_m]$ with contextual embeddings, as shown in Figure 3.The matrix $\boldsymbol{h}_{d\times m}$ is then converted into the sentence vector $\boldsymbol{h}^g$ by a linear layer with $softmax$ function.

Furthermore, the representation generator also competes against the meta discriminator as much as possible, so that the discriminator can not determine whether the samples are from the source

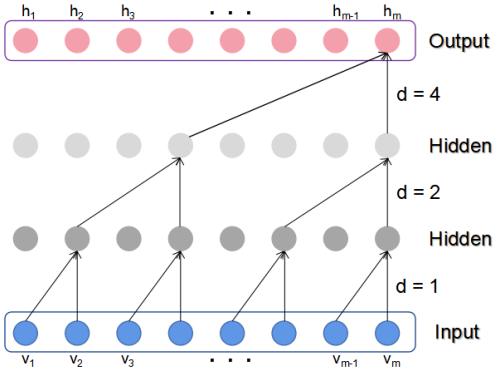

Figure 3: The TCN with dilation factors d = 1, 2, 4 and filter kernel k = 2.

domain or the target domain. Only in this way can more comprehensive features be created for better classification performance, which is the core motivation in building the representation generator and meta discriminator.

4.2 Meta Discriminator

We consider the examples in the support set and query set as the target domain, and the remaining examples as the source domain. Definitely, we sample a subset of the same size as the query set from the source domain as the source set. The discriminator comprises a three-layer feed-forward neural network and a $softmax$ function. The calculation process is as follows:

$$\hat{Y} = softmax(MLP(\boldsymbol{h}^g)) \tag{2}$$

where $\boldsymbol{h}^g$ denotes the sentence vector of each example from the representation generator. $\hat{Y} \in \{0, 1\}$ represents whether the example is from the query set or the source set.

4.3 Semantic Extractor

The semantic extractor fuses the word embedding from fastText and the representation vector from the generator to create the more comprehensive feature representation $\boldsymbol{w}$ for each example, which can be expressed as:

$$\boldsymbol{w} = \boldsymbol{V}_{d \times m} \cdot \boldsymbol{h}^g \tag{3}$$

where $\boldsymbol{V}_{d \times m} = [\boldsymbol{v}_1, \boldsymbol{v}_2, \cdots, \boldsymbol{v}_m]$. $\boldsymbol{v}_m$ is the word embedding vector and m is the number of words in the sentence. $\boldsymbol{h}^g$ denotes the sentence vector generated by the representation generator.

4.4 Feature Enhancer

We construct the feature enhancer with the multi-head attention mechanism (Vaswani et al., 2017) and the MLP layer. The multi-head attention module consists of multiple self-attention units. Each self-attention unit has a powerful capability to catch valuable feature information about the input, and the knowledge learned by various self-attention units from their respective perspectives can be incorporated in a concatenation manner, as shown in Figure 4.

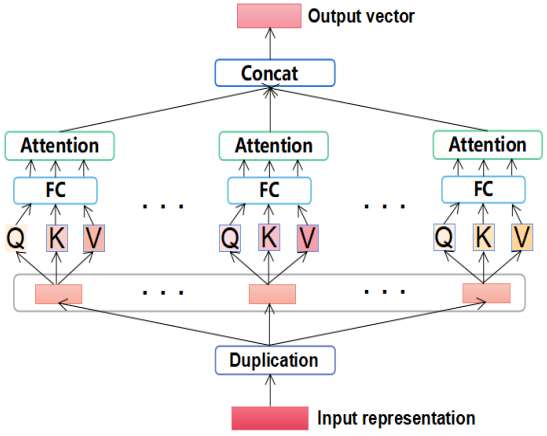

Figure 4: Details of the multi-head attention mechanism.

Thus, we employ the multi-head attention mechanism to effectively grasp the feature interaction between the representation sequences $\boldsymbol{w}^s$ of support examples and the representation sequences $\boldsymbol{w}^q$ of query examples. We then leverage the MLP layer with $GELU(\cdot)$ activation function (Hendrycks et al., 2020) to produce the final support vector sequences $\boldsymbol{z}^s$ and query vector sequences $\boldsymbol{z}^q$. The whole process above can be described as:

$$E_{sq} = [\boldsymbol{w}^s, \boldsymbol{w}^q] \tag{4}$$

$$H_{sq} = MHAttention(E_{sq}) \tag{5}$$

$$\boldsymbol{z}^s, \boldsymbol{z}^q = GELU(MLP(H_{sq})) \tag{6}$$

where $\boldsymbol{w}^s = [\boldsymbol{w}_1^s, \boldsymbol{w}_2^s, \cdots, \boldsymbol{w}_k^s]$ and $\boldsymbol{w}^q = [\boldsymbol{w}_1^q, \boldsymbol{w}_2^q, \boldsymbol{w}_3^q, \cdots, \boldsymbol{w}_p^q]$. The k and p represent the number of examples in the support set and query set, respectively. E_{sq} is the concatenation result of $\boldsymbol{w}^s$ and $\boldsymbol{w}^q$. $MHAttention(\cdot)$ is actually the main component of $Transformer$ model (Vaswani et al., 2017) and is used alone in our work to generate the self-attention output H_{sq}. The final support vector sequences $\boldsymbol{z}^s = [\boldsymbol{z}_1^s, \boldsymbol{z}_2^s, \cdots, \boldsymbol{z}_k^s]$ and query vector sequences $\boldsymbol{z}^q = [\boldsymbol{z}_1^q, \boldsymbol{z}_2^q, \cdots, \boldsymbol{z}_p^q]$.

The feature enhancer is very beneficial to boost the feature correlation between support examples and query examples and decrease the interference

of noisy representations. With limited knowledge, the model can create the class-level representations that are more compatible with query examples and the query representations that are more consistent with support examples, and acquire fine generalization ability across various classification tasks rapidly.

4.5 Classifier

We adopt the prototypical network (Snell et al., 2017) with Euclidean distance as the classifier in our framework, which enables the model to easily solve the learning problem and efficiently accelerate training convergence.

Therefore, the classifier generates the class-level vector $\boldsymbol{C}_u$ for each class u based on support vectors $\boldsymbol{z}_i^s$ and measures the similarity probability, $\boldsymbol{D}_j$, between the query vector $\boldsymbol{z}_j^q$ and the class vector $\boldsymbol{C}_u$ through the Euclidean distance function $R(\cdot,\cdot)$.

$$\boldsymbol{C}_u = \frac{1}{K} \sum_{(\boldsymbol{z}_i^s, Y_i) \in S_u} \boldsymbol{z}_i^s \tag{7}$$

$$\boldsymbol{D}_j = \frac{e^{R(\boldsymbol{C}_u, \boldsymbol{z}_j^q)}}{\sum_{n=1}^{N} e^{R(\boldsymbol{C}_n, \boldsymbol{z}_j^q)}} \tag{8}$$

where $S_u \subset S$ is the subset corresponding to class u with the same label Y_i in the support set S. And K represents the number of examples in each class.

4.6 Loss Function

The modules of our framework SaAML are trained using different strategies, .e.g, we train the classifier from scratch for each episode, while the representation generator is optimized across all training episodes. For each training episode, we first employ the source set and query set to update the parameters of the meta discriminator. Next, we update the parameters of the representation generator and classifier over the support set and query set. The details of the loss function are introduced below.

The classifier loss consists of cross-entropy loss and difference loss. The difference loss aims to maximize the distance between different class vectors, making each class as directionally different as possible, which is defined as:

$$L_{DL} = \lambda \sum_{i \neq j} \| \boldsymbol{C}_i^T \boldsymbol{C}_j \|_F^2 \tag{9}$$

where $\| \cdot \|_F$ is the Frobenius norm, and λ is the hyperparameter and can be set to 10^{-3}.

Thus, the classifier loss can be represented as:

$$L^C = -\frac{1}{N} \sum_{n=1}^{N} \sum_{k=1}^{K} Y_{n,k} \, log \, P_{n,k} + L_{DL} \tag{10}$$

where $P_{n,k}$ indicates the probability that the n-th query example is predicted to be the k-th label. $Y_{n,k}$ denotes the ground-truth label. The K and N are the number of categories and examples, respectively.

It is standard practice to apply the cross-entropy loss as the loss function for the discriminator.

$$\begin{aligned} L^D = -\frac{1}{2n} \sum_{k=1}^{n} [& (Y_k^d \, log \, \hat{Y}_k \\ & + (1 - Y_k^d) \, log \, (1 - \hat{Y}_k))] \end{aligned} \tag{11}$$

where Y_k^d and $\hat{Y}_k$ denotes the real class of the example and the prediction result of the discriminator, respectively. The n is the number of examples in the source set or query set.

As for the loss function of the generator, it can be regarded as the combination of the classifier loss function and the discriminator loss function, which are used to gain the final classification results and confuse the discriminator, respectively.

$$L^G = L^C - L^D \tag{12}$$

5 Experiment

We investigate the performance of our proposed SaAML against six existing well-established baselines through extensive experiments on four benchmark datasets. Furthermore, we also develop a series of ablation studies to further illustrate the effectiveness and robustness of SaAML.

5.1 Datasets

There are four few-shot text classification datasets in our work, as expressed in Table 1.

Table 1: Details of the four benchmark datasets.

Dataset	Avg. text length	samples per class	train/val/test classes
Amazon	140	1000	10 / 5 / 9
20 Newsgroups	340	941	8 / 5 / 7
HuffPost	11	900	20 / 5 / 16
Banking77	16	170	30 / 15 / 32

Amazon is a collection of 24 product categories (He and McAuley, 2016) with 142.8 million product reviews. Following the same approach

Table 2: Mean classification accuracy on Amazon, HuffPost, 20 Newsgroups, and Banking77 datasets.

Method	Amazon		HuffPost		20 Newsgroups		Banking77			
	5-way		5-way		5-way		10-way		15-way	
	1-shot	5-shot	1-shot	5-shot	1-shot	5-shot	1-shot	5-shot	1-shot	5-shot
MAML	0.3965	0.4713	0.3572	0.4931	0.3384	0.4372	0.4691	0.6659	0.3672	0.5659
PN	0.3760	0.5214	0.3573	0.4474	0.3780	0.4535	0.4156	0.6975	0.3501	0.6393
IN	0.3491	0.4132	0.3874	0.4912	0.2876	0.3337	0.5291	0.6884	0.4553	0.6179
RRML	0.5022	0.7275	0.3610	0.4966	0.3760	0.5724	0.5256	0.8148	0.4694	0.7734
DS-RRML	0.6260	0.8112	0.4305	0.6350	0.5212	0.6830	0.6034	0.8373	0.5432	0.7900
MLADA	0.6842	0.8600	0.4502	0.6491	0.5961	0.7780	0.6055	0.8089	0.5513	0.7470
SaAML	**0.7147**	**0.8637**	**0.5126**	**0.6944**	**0.7079**	**0.8430**	**0.6860**	**0.8483**	**0.6235**	**0.8096**

(Bao et al., 2019), we select 1000 reviews from each category to establish the subset.

20 Newsgroups consists of about 20000 news sentences evenly partitioned on 20 different topics, which is from the news discussion boards (Lang, 1995).

HuffPost is extracted from the HuffPost articles between the year 2012 and 2018 (Misra, 2018), with news headlines in 41 categories.

Banking77 provides 77 classes of 13083 fine-grained intents from the banking domain, proposed by Casanueva et al. (Casanueva et al., 2020).

5.2 Baselines

Six existing few-shot learning baselines are adopted to compare with our proposed SaAML, which are briefly introduced as follows:

Model-Agnostic Meta-Learning (MAML) (Finn et al., 2017) is an optimization-based approach to explicitly train model parameters such that the model can produce great generalization on new tasks after a few gradient update steps.

Prototypical Network (PN) (Snell et al., 2017) is a metric-based method that employs the feature average of support examples as the class vector (prototype).

Induction Network (IN) (Geng et al., 2019) constructs the class vector through the dynamic routing algorithm based on capsule network and leverages the relation module (Sung et al., 2018) to learn the measure function.

Ridge Regression Meta-Learner (RRML) (Bertinetto et al., 2018) exploits the ridge regression to obtain the class vector and develops proper regularization to reduce model overfitting and speed up model convergence.

Distributional Signature (DS) (Bao et al., 2019) considers that the distribution signature is very essential to catch more comprehensive feature representations. Therefore, this meta-learning framework achieves outstanding performance when combined with distribution signatures, where DS+RRML is the best method.

Meta-Learning Adversarial Domain Adaptation Network (MLADA) (Han et al., 2021) explores the adversarial network architecture to extract sentence features, improving the generalization and performance of meta-learning systems in various scenarios.

In our study, we adopt the pre-trained fastText (Joulin et al., 2016) as the embedding encoder in all methods. TCN includes a total of four layers. For each layer, the number of hidden units is 300 and kernel size is 2. The number of hidden units of MLP in the discriminator is 256 and 128, respectively. In the feature enhancer, the head number and the feature dimension of the attention mechanism are fixed as 6 and 300. For a fair comparison, 100, 100, and 1000 task episodes are randomly sampled individually in each training, validation, and testing epoch. Furthermore, we optimize model parameters through the $Adam$ algorithm (Kingma and Ba, 2014) with a learning rate of $1e-5$ and use early stopping scheme when the performance on the validation set fails to increase within 20 epochs. We conduct all experiments on the NVIDIA Geforce GTX 3090 GPUs server.

5.3 Experimental Results

We can obtain several valuable observations from the comparison results of different baseline models, and the results are depicted in Table 2. (1) Overall, our model SaAML creates an average accuracy of 64.89% in 1-shot classification and 81.18% in 5-shot classification, substantially refreshing the best performance on all datasets. Notably, it attains a significant performance improvement of 5.73% over the state-of-the-art system MLADA

(Han et al., 2021). (2) We find that SaAML generally outperforms 5-shot classification on 1-shot classification, with an average accuracy improvement of 7.15% in 1-shot classification and 2.89% in 5-shot classification. This is understandable since the feature information available in low-shot scenes is extremely deficient, especially for 1-shot classification, whereas SaAML is very competent in leveraging the combination of the adversarial network architecture and the feature enhancer to grasp semantic information from a few examples, so that it can address this challenge rapidly. (3) We also notice that SaAML considerably improves the accuracy of 20 newsgroups by nearly 8.84%, which is better than other datasets in terms of performance improvement. It clearly illustrates that the model has a powerful feature learning ability and is more compatible with long texts with rich information. Besides, we have to acknowledge that the performance improvement of SaAML on Amazon is not apparent because of its affluent examples and diverse expressions. Note that the experimental results have 95% confidence intervals with variances below 0.01.

5.4 Ablation Study

We perform massive experiments to explore the effectiveness and adaptability of various components, .i.e., adversarial network (AN), temporal convolutional network (TCN), semantic extractor (SE), feature enhancer (FE) and difference loss (DL). The ablation results are written in Table 3.

Firstly, we remove the setup of the adversarial network (AN) architecture, which involves the meta-discriminator and the source set. That is, the discriminator no longer strengthens the features via the adversarial training. In this way, the classification performance of the model is significantly weakened, which justifies the necessity and effectiveness of adopting the adversarial network.

Secondly, we utilize the BiLSTM instead of the TCN (Bai et al., 2018) to build the representation generator. It is observed that the performance of the model with TCN is superior to the model with BiLSTM. This is because BiLSTM saturates at a very early training stage due to optimization difficulties, while TCN can capture the longer-distance dependence of feature information.

Thirdly, we examine the role of the semantic extractor (SE) and feature enhancer (FE) in performance improvement. Obviously, the classification accuracy of the model without the feature enhancer drops by 12.30%. The model without the semantic extractor also leads to the performance drop of 7.48%. It robustly demonstrates that the semantic extractor can gain the richer semantics of the examples via a fusion manner, and the feature enhancer can grasp the interaction knowledge between support examples and query examples.

Finally, we investigate the importance of difference loss (DL). We find that difference loss can facilitate the orthogonality between different classes and improve the performance on few-shot classification tasks. In addition, we further explore to replace the fastText (Joulin et al., 2016) with the BERT (Devlin et al., 2018) as the embedding encoder. It is regrettable that BERT does not bring performance improvement but will increase the model complexity. This indicates that in the few-shot scenario, powerful BERT may lead to the overfitting of the model, resulting in performance degradation.

6 Discussion

Our work follows the research idea proposed by Han et al. (Han et al., 2021) to exploit the adversarial network to catch feature representations. However, the implementation structure of MLADA (Han et al., 2021) is too simple to extract intrinsic sentence feature information and strengthen the generalization ability of the model well. In contrast, we upgrade the whole meta-learning framework with more cost-effective modules. The experimental results in Table 2 illustrate that the performance of our proposed SaAML is significantly better than MLADA.

Specifically, we build the generator with the temporal convolutional network (TCN) rather than bidirectional LSTM (BiLSTM) to excavate the inherent semantic knowledge of sentences. We also exploit the feature enhancer to grasp the rich feature interaction between different sentence representations through the multi-head attention mechanism. Moreover, we adopt the prototype network (PN) based on the difference loss as the classifier, improving the model's discriminative ability. Overall, our model SaAML can construct more comprehensive transferable representations and accomplish excellent performance.

In addition, we further discuss the adaptation potential of the SaAML and MLADA. We adopt RRML (Bertinetto et al., 2018) and PN (Snell

Table 3: The ablation results on Amazon, HuffPost, 20 Newsgroups, and Banking77 datasets.

Model	Amazon 5-way		HuffPost 5-way		20 Newsgroups 5-way		Banking77 10-way	
	1-shot	5-shot	1-shot	5-shot	1-shot	5-shot	1-shot	5-shot
-AN	0.6918	0.8470	0.4969	0.6764	0.7025	0.8302	0.6567	0.8360
-TCN	0.6966	0.8363	0.5057	0.6872	0.6745	0.8240	0.6789	0.8448
-SE	0.6108	0.7887	0.4683	0.6404	0.5589	0.7514	0.6456	0.8264
-FE	0.6005	0.7886	0.3773	0.5877	0.5443	0.7503	0.5646	0.6731
-DL	0.7075	0.8490	0.5065	0.6874	0.7043	0.8381	0.6837	0.8415
+BERT	0.6647	0.8340	0.4888	0.6815	0.5136	0.6737	0.6845	**0.8526**
SaAML	**0.7147**	**0.8637**	**0.5126**	**0.6944**	**0.7079**	**0.8430**	**0.6860**	0.8483

Table 4: The results for inserting augmentation components into PN and RRML on different datasets.

Model	Amazon 5-way		HuffPost 5-way		20 Newsgroups 5-way		Banking77 10-way	
	1-shot	5-shot	1-shot	5-shot	1-shot	5-shot	1-shot	5-shot
PN	0.3760	0.5214	0.3573	0.4474	0.3780	0.4535	0.4156	0.6975
MLADA-PN	0.5587	0.7220	0.3106	0.4632	0.5088	0.6453	0.4662	0.6055
SaAML-PN	**0.7147**	**0.8637**	**0.5126**	**0.6944**	**0.7079**	**0.8430**	**0.6860**	**0.8483**
RRML	0.5022	0.7275	0.3610	0.4966	0.3760	0.5724	0.5256	**0.8148**
MLADA-RRML	0.6842	**0.8600**	0.4502	0.6491	0.5961	0.7780	0.6055	0.8089
SaAML-RRML	**0.6856**	0.8422	**0.4588**	**0.6605**	**0.6680**	**0.8062**	**0.6130**	0.7859

et al., 2017) as the classifier to build the model, respectively. As exhibited in Table 4, regardless of whether RRML or PN is used as the classifier, SaAML obtains some performance progress across different datasets in contrast to the published original model and MLADA. It reveals that our SaAML has an enormous opportunity for updating diverse meta-learning systems.

7 Conclusion

In this paper, we propose a novel meta-learning framework Sentence-aware Adversarial Meta-Learner (SaAML) to address few-shot text classification task. Exactly, under the architecture of the adversarial network, we explore the representation generator with TCN as the core module to encourage more discriminative representation learning and exploit the feature enhancer to facilitate more consistent and comprehensive feature expression, which exceedingly strengthens the adaptability and generalization of SaAML for new classes. We develop comprehensive experiments on four benchmark datasets to demonstrate that the proposed model gains substantial improvements over existing state-of-the-art meta-learning approaches.

References

Shaojie Bai, J Zico Kolter, and Vladlen Koltun. 2018. An empirical evaluation of generic convolutional and recurrent networks for sequence modeling. *arXiv preprint arXiv:1803.01271*.

Yujia Bao, Menghua Wu, Shiyu Chang, and Regina Barzilay. 2019. Few-shot text classification with distributional signatures. *arXiv preprint arXiv:1908.06039*.

Luca Bertinetto, Joao F Henriques, Philip HS Torr, and Andrea Vedaldi. 2018. Meta-learning with differentiable closed-form solvers. *arXiv preprint arXiv:1805.08136*.

Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. 2020. Language models are few-shot learners. *Advances in neural information processing systems*, 33:1877–1901.

Inigo Casanueva, Tadas Temčinas, Daniela Gerz, Matthew Henderson, and Ivan Vulić. 2020. Efficient intent detection with dual sentence encoders. *arXiv preprint arXiv:2003.04807*.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*.

Chelsea Finn, Pieter Abbeel, and Sergey Levine. 2017. Model-agnostic meta-learning for fast adaptation of deep networks. In *International conference on machine learning*, pages 1126–1135. PMLR.

Ruiying Geng, Binhua Li, Yongbin Li, Xiaodan Zhu, Ping Jian, and Jian Sun. 2019. Induction networks for few-shot text classification. *arXiv preprint arXiv:1902.10482*.

A. Gupta, K. Thadani, and N. O'Hare. 2020. Effective few-shot classification with transfer learning. In *Proceedings of the 28th International Conference on Computational Linguistics*.

ChengCheng Han, Zeqiu Fan, Dongxiang Zhang, Minghui Qiu, Ming Gao, and Aoying Zhou. 2021. Meta-learning adversarial domain adaptation network for few-shot text classification. *arXiv preprint arXiv:2107.12262*.

Ruining He and Julian McAuley. 2016. Ups and downs: Modeling the visual evolution of fashion trends with one-class collaborative filtering. In *proceedings of the 25th international conference on world wide web*, pages 507–517.

Dan Hendrycks, Xiaoyuan Liu, Eric Wallace, Adam Dziedzic, Rishabh Krishnan, and Dawn Song. 2020. Pretrained transformers improve out-of-distribution robustness. *arXiv preprint arXiv:2004.06100*.

Xiang Jiang, Mohammad Havaei, Gabriel Chartrand, Hassan Chouaib, Thomas Vincent, Andrew Jesson, Nicolas Chapados, and Stan Matwin. 2018. Attentive task-agnostic meta-learning for few-shot text classification.

Armand Joulin, Edouard Grave, Piotr Bojanowski, Matthijs Douze, Hérve Jégou, and Tomas Mikolov. 2016. Fasttext. zip: Compressing text classification models. *arXiv preprint arXiv:1612.03651*.

Diederik P Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*.

Ken Lang. 1995. Newsweeder: Learning to filter netnews. In *Machine Learning Proceedings 1995*, pages 331–339. Elsevier.

Kwonjoon Lee, Subhransu Maji, Avinash Ravichandran, and Stefano Soatto. 2019. Meta-learning with differentiable convex optimization. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 10657–10665.

Rishabh Misra. 2018. News category dataset.

Tsendsuren Munkhdalai and Hong Yu. 2017. Meta networks. In *International Conference on Machine Learning*, pages 2554–2563. PMLR.

C. Pan, J. Huang, J. Gong, and X. Yuan. 2019. Few-shot transfer learning for text classification with lightweight word embedding based models. *IEEE Access*.

Aravind Rajeswaran, Chelsea Finn, Sham M Kakade, and Sergey Levine. 2019. Meta-learning with implicit gradients. *Advances in neural information processing systems*, 32.

Sachin Ravi and Hugo Larochelle. 2016. Optimization as a model for few-shot learning.

J. Salamon and J. P. Bello. 2017. Deep convolutional neural networks and data augmentation for environmental sound classification. *IEEE Signal Processing Letters*, PP(3):1–1.

J. Schmidhuber. 1987. Evolutionary principles in self-referential learning. *genetic programming*.

Jake Snell, Kevin Swersky, and Richard Zemel. 2017. Prototypical networks for few-shot learning. *Advances in neural information processing systems*, 30.

Pengfei Sun, Yawen Ouyang, Wenming Zhang, and Xinyu Dai. 2021. Meda: Meta-learning with data augmentation for few-shot text classification. In *IJCAI*, pages 3929–3935.

Shengli Sun, Qingfeng Sun, Kevin Zhou, and Tengchao Lv. 2019. Hierarchical attention prototypical networks for few-shot text classification. In *Proceedings of the 2019 conference on empirical methods in natural language processing and the 9th international joint conference on natural language processing (EMNLP-IJCNLP)*, pages 476–485.

Flood Sung, Yongxin Yang, Li Zhang, Tao Xiang, Philip HS Torr, and Timothy M Hospedales. 2018. Learning to compare: Relation network for few-shot learning. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1199–1208.

C. Tong. 2019. Metagan: An adversarial approach to few-shot learning.

Eric Tzeng, Judy Hoffman, Kate Saenko, and Trevor Darrell. 2017. Adversarial discriminative domain adaptation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 7167–7176.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. *Advances in neural information processing systems*, 30.

Oriol Vinyals, Charles Blundell, Timothy Lillicrap, Daan Wierstra, et al. 2016. Matching networks for one shot learning. *Advances in neural information processing systems*, 29.

Ling Yang, Liangliang Li, Zilun Zhang, Xinyu Zhou, Erjin Zhou, and Yu Liu. 2020. Dpgn: Distribution propagation graph network for few-shot learning. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 13390–13399.

Reweighting Strategy based on Synthetic Data Identification for Sentence Similarity

Taehee Kim[†]*, **ChaeHun Park**[*], **Jimin Hong**,
Radhika Dua, **Edward Choi** and **Jaegul Choo**
KAIST AI, Letsur[†]
{taeheekim,ddehun,jimmyh,radhikadua,edwardchoi,jchoo}@kaist.ac.kr

Abstract

Semantically meaningful sentence embeddings are important for numerous tasks in natural language processing. To obtain such embeddings, recent studies explored the idea of utilizing synthetically generated data from pretrained language models (PLMs) as a training corpus. However, PLMs often generate sentences much different from the ones written by human. We hypothesize that treating all these synthetic examples equally for training deep neural networks can have an adverse effect on learning semantically meaningful embeddings. To analyze this, we first train a classifier that identifies machine-written sentences, and observe that the linguistic features of the sentences identified as written by a machine are significantly different from those of human-written sentences. Based on this, we propose a novel approach that first trains the classifier to measure the importance of each sentence. The distilled information from the classifier is then used to train a reliable sentence embedding model. Through extensive evaluation on four real-world datasets, we demonstrate that our model trained on synthetic data generalizes well and outperforms the existing baselines.[1]

1 Introduction

High-quality sentence embeddings are essential in diverse applications of natural language processing (Cer et al., 2018; Reimers and Gurevych, 2019), including semantic textual similarity (Cer et al., 2017) and paraphrase identification (Dolan and Brockett, 2005). Unfortunately, obtaining a large amount of human-annotated datasets to train a sentence embedding model is difficult and expensive. To address this, Schick and Schütze (2021) recently introduced a method, DINO, to train a sentence embedding model on synthetic data generated from pretrained language models (PLMs). Despite the effectiveness and scalability of DINO, however, the difference between machine-written and human-written examples has not been carefully investigated. In other words, the study on the impact of treating all these synthetic examples equally during training remains under-explored.

To this end, we first conduct an in-depth analysis to demonstrate the shift of synthetic samples from the human-written sentences. In particular, we train a classifier (*i.e.*, Synthetic Data Identification (SDI) model) that identifies synthetic data from human-written sentences and observe that the linguistic features of the sentences predicted as machine-written are much different from the human-written sentences compared to the linguistic features of the sentences predicted as human-written.

Based on this analysis, we propose a simple method, **R**eweighting Loss based on **I**mportance of Machine-written **SE**ntence (RISE), which first utilizes the trained SDI model to measure the importance of each sentence in learning semantically meaningful sentence embeddings for sentence similarity tasks. We then utilize this distilled information from the SDI model to reweight the loss of each synthetic example during training.

We extensively evaluate our method on multiple sentence similarity datasets and observe that our model outperforms all the baselines across diverse datasets, even when they are evaluated on other datasets from a distinct distribution with training datasets. Our contributions include:

- We analyze the linguistic features of machine-written sentences in synthetic dataset compared to human-written sentences.
- We propose a simple method that adjusts the contribution of synthetically generated samples to learn a reliable sentence encoder.
- We extensively evaluate our model on diverse

[*] Equal contribution

[1] Our implementation is publicly available at https://github.com/ddehun/coling2022_reweighting_sts.

Proceedings of the 29th International Conference on Computational Linguistics, pages 4853–4863
October 12–17, 2022.

	STSb			QQP			MRPC		
	x_h	$p_D(x_m) \uparrow$	$p_D(x_m) \downarrow$	x_h	$p_D(x_m) \uparrow$	$p_D(x_m) \downarrow$	x_h	$p_D(x_m) \uparrow$	$p_D(x_m) \downarrow$
BLEU-N	34.80	25.75	2.93	30.3	34.95	7.86	48.53	46.97	5.59
Jaccard	41.98	33.97	5.98	39.91	42.49	11.31	53.55	53.33	10.52
Distinct-N	44.53	35.93	17.03	38.10	25.23	24.10	44.63	32.10	22.00
Zipf coeff.	1.03	1.07	1.23	1.11	1.06	1.12	0.98	1.02	1.23

Table 1: Results for comparing the sentences in different group. Jaccard indicates Jaccard similarity score. The score of generated sentences far from human scores is highlighted in underline. BLEU-N and Distinct-N indicate the average score with different N. The full results are available in Appendix A.

datasets and observe that our method demonstrates consistent gains, generalizes well to datasets from different domains, and is robust to the adversarial attack.

2 Related Work

Synthetic data generation using pretrained language models has shown promising results in various natural language processing tasks (Yang et al., 2020; Papanikolaou and Pierleoni, 2020; Ding et al., 2020; Edwards et al., 2021; Chang et al., 2021). Recently, Schick and Schütze (2021) proposed a new method, DINO, to generate a synthetic dataset for textual semantic similarity task. Another recent work, Yoo et al. (2021) proposed a new data augmentation framework for sentence classification by leveraging a large-scale PLM (Brown et al., 2020). However, synthetic data can be misused in malicious usage, such as fake news generation. To prevent such a fraudulent use, recent studies (Zellers et al., 2019; Weiss, 2019; Uchendu et al., 2020; Adelani et al., 2020) aim to detect the synthetically generated text. On the contrary, we aims to identify unrealistic sentences from machine-written data and mitigate their influence to achieve accurate and robust learning. While Yi et al. (2021) suggested controlling weights to augmented training examples, our work mainly focuses on using only synthetic samples from PLMs.

3 Analysis on Synthetic Sentences

This section describes the generation of the synthetic dataset, followed by training the model to identify synthetic sentences from human-written ones. Then, we present a novel analysis to demonstrate the shift of synthetic samples from the human-written sentences.

Synthetic Data Generation. To obtain machine-generated sentences, we leverage the ability of prompt-based zero-shot generation in a generative PLM (Radford et al., 2019) (Figure 1-A). Specifically, given a sentence $x_h \in C_{src}$ where C_{src} is a set of human-written sentences and the target similarity level $y \in Y$, this framework produces a sentence $x_m \in X_m$ that has semantic similarity with x_h equal to the target similarity level y. The generated examples $\{x_h, x_m, y\}$ are later used to train a model for sentence similarity tasks.

We use Semantic Textual Similarity benchmark (STSb) (Cer et al., 2017), Quora Question Pairs (QQP)[2], and Microsoft Research Paraphrase Corpus (MRPC) (Dolan and Brockett, 2005) as a source of human sentences C_{src}. We follow the details for data generation in Schick and Schütze (2021) with their official implementation.[3] Finally, we obtain about 76k, 78k, and 55k examples of STSb, QQP, and MRPC datasets, respectively.

Synthetic Data Identification (SDI). We now train a binary classification model D based on a bi-directional PLM (Devlin et al., 2019) to distinguish machine-written sentences from human-written sentences (Figure 1-B). We refer to this model as the Synthetic Data Identification (SDI) model and train it separately for each C_{src}. We use machine-written sentences X_m and human sentences X_h in the same proportion for training.[4] We use the prediction confidence p_D of the generated sentence to measure how natural the sentence is.

Analysis. We now analyze to demonstrate the shift of synthetic samples from the human-written sentences. We use the following metrics to analyze the lexical-level linguistic patterns of each sentence: (1) **BLEU** (Papineni et al., 2002) and **Jaccard Similarity** (Montahaei et al., 2019) that calculate the lexical-level similarity between x_m and its paired sentence. (2) **Distinct-N** (Li et al.,

[2] https://quoradata.quora.com/First-Quora-Dataset-Release-Question-Pairs

[3] gpt2-xl is used as a PLM for data generation.

[4] The accuracy of classifiers of each dataset on the validation set are 77.87, 83.21, and 93.05% in STSb, MRPC, and QQP datasets, respectively.

x_h:	Mitchell, Man of La Mancha, Malcolm Gets from Amour and John Selya, Movin' Out, complete the field.
x_m:	The other 30 percent is going to be a challenge, $(y = 0)$ ***(Grammatical Error)***
x_h:	A black dog is walking on the grass with a stick in it's mouth.
x_m:	A brown dog walks with his stick in his mouth and it is brown in color. The black dog is walking with his stick in his mouth and it has a brown in color on it. $(y = 0.5)$ ***(Repetition)***
x_h:	Janet Racicot heard the thud from the kitchen, where she was getting a glass of water, she said in an interview.
x_m:	I have been to a few movies and I am not a big movie fan. I like horror movies. $(y = 0)$ ***(Logical Flaw)***
x_h:	The rate of survival without serious brain damage is about 10 percent, said Bossaert, a professor at the University Hospital in Antwerp, Belgium.
x_m:	The brain damage in the patient is not serious. The patient is a young woman who was born in the United States, but was born in Belgium. $(y = 0.5)$ ***(Logical Flaw)***
x_h:	It would be difficult to overestimate the potential dangers of the Remote Procedure Call (RPC) vulnerability.
x_m:	A senior official in the Oil ministry asked an oil company's CEO about his meeting with the minister. This is a very sensitive issue and is very sensitive to both sides. $(y = 0)$ ***(Uncommon in Context)***

Table 2: Examples of machine-written sentences identified by the SDI model as unrealistic. The part of sentences that contains linguistic errors is highlighted in red. More examples are available in Appendix B.

2015) that calculates the ratio of unique N-grams among the total number of N-grams in each group for x_m. (3) **Zipf coefficient** (Holtzman et al., 2019) that calculates the Zipf coefficient to analyze the vocabulary usage for x_m. We utilize the prediction confidence p_D from the SDI model to measure the importance of generated sentences in learning meaningful sentence embeddings. We select the top 10% ($p_D(x_m) \uparrow$) and bottom 10% ($p_D(x_m) \downarrow$) of the machine-written sentences based on their sorted importance and analyze their linguistic features.

Table 1 demonstrates that linguistic patterns of synthetic examples vary significantly according to their importance score $p_D(x_m)$. Furthermore, we observe that except for Zipf coefficient in QQP dataset, generated sentences with high $p_D(x_m)$ always have scores close to the scores of human-written sentences (x_h) compared to the sentences with low $p_D(x_m)$.[5] Further qualitative analysis in Table 2 reveals that the sentences with low importance score are *unrealistic* since they often contain repetition, logical flaw or expressions that a human does not use frequently. For example, as shown in the second example of Table 2, a person does not like movies, but in the next sentence, the machine generates a sentence that the person likes horror movies. In the third example, a machine generates a sentence that a woman was born in two places.

Based on these observations, we confirm that there exist a large variance in terms of how much the sentences are shifted from human sentences. Therefore, it is critical to handle the generated sentences carefully so that the model is not biased to the sentences that are sufficiently different from human sentences. In the remaining of this paper, we refer to the generated sentence as *unrealistic* if they contain linguistic errors or lexical patterns different from humans. To identify such unrealistic sentences, we leverage the importance score (p_D) from SDI model. We regard sentences with lower score from the model as more *unrealistic*.

4 Proposed Method

We now introduce a simple yet effective method, **R**eweighting Loss based on **I**mportance of Machine-written **SE**ntence (RISE), that aims to give less importance to unrealistic machine-written sentences than realistic sentences. Our method consists of two stages: (1) measuring the importance of the generated sentences in learning semantically meaningful embeddings using the prediction confidence p_D from the SDI model (defined in Section 3); 2) utilizing the importance score to control the weight of the loss for each example during training so that the model does not deviate significantly from the distribution of the human text. Other than the loss function, the training procedure is the same as standard training of a sentence embedding model based on the bi-encoder architecture (Reimers and Gurevych, 2019). More details on training the sentence encoder are provided in Appendix D.

Reweighting Loss using Importance Score. We utilize the prediction confidence p_D from the SDI model (Section 3) to measure the importance of generated sentences. In particular, we modify the loss to make the realistic machine-written examples (*i.e.*, examples with high scores) have more

[5] We provide a more detailed analysis in Appendix A.

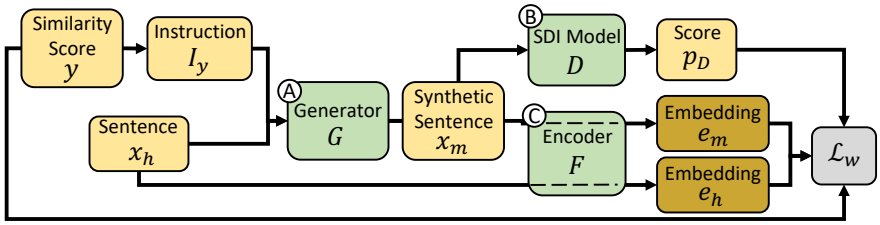

Figure 1: Overview of **RISE**. We feed an instruction I_y and a human-written sentence x_h to the Generator G which produces a machine-written sentence x_s. We then measure importance score p_D using x_s as input. Finally, we predict the similarity score using the embedding vector of x_s and x_h. We compute the loss and multiply p_D.

contributions to the loss, whereas the unrealistic machine-written examples (*i.e.*, examples with low score) have less contribution (Figure 1-C). The loss of each example is defined as:

$$\mathcal{L}_{\text{w}}(\theta_f) = p_D * \mathcal{L}(\theta_f), \tag{1}$$

where $\mathcal{L}(\theta_f)$ denotes the original loss of the sentence encoder F for a sentence similarity task, and $\mathcal{L}_w(\theta_f)$ denotes the modified loss by RISE. θ_f denotes the parameters of the sentence encoder. This re-weighting procedure aims to adjust the influence of training examples based on the degree of shift of the sentence from the human-written sentences.

5 Experimental Settings

We evaluate each model on STSb, QQP, MRPC, and Paraphrase Adversaries from Word Scrambling of Quora Question Pairs (Zhang et al., 2019) (PAWS-QQP) datasets. PAWS-QQP aims to evaluate the robustness of the model against adversarial attacks for the sentence similarity task. We provide more details on datasets and experimental setup in the Appendix E and F.

We train a model to solve the sentence similarity task as a regression problem. However, since all datasets except for STSb only contain discrete labels, we set the threshold using the validation dataset to make a binary decision. We apply our method to DINO and denote it as RISE. In addition to experiments with RISE, we conduct experiments with the following variants: (1) *Filtering*: We filter out the bottom 10% of the machine-written sentences based on their sorted importance. We then use the remaining examples for training without using our modified loss. (2) *Random*: We randomly sample a scalar value from $U(0, 1)$ for each example and use it as its importance. DINO and the variants of our method are based on the sentence-RoBERTa-base architecture, which are fine-tuned only on synthetic datasets. Besides, we further compare our model against the following sentence encoders that are fine-tuned on natural language inference (NLI) dataset: Universal Sentence Encoder(USE) (Cer et al., 2018), InferSent (Conneau et al., 2017), sentence-BERT (Reimers and Gurevych, 2019), and sentence-RoBERTa. We also compare with the models that are not trained on human-annotated dataset, namely: GloVe (Pennington et al., 2014), BERT-CLS, sentence-BERT, sentence-RoBERTa.[6]

6 Results

Table 3 report the performance of our method and the baselines on the sentence similarity task. We observe that our model outperforms all the other baselines including DINO that are not trained on human-annotated dataset, and sometimes even better than the models trained on human-annotated dataset (*i.e.*, NLI). These results support our assumption that reweighting the loss of each machine-written sentence based on its importance enhances the model's reliability and makes it less biased to unrealistic machine-written sentences. Furthermore, we find that the improvement is usually higher when the model is evaluated on datasets from unseen domain during training. These results imply that our method can generalize the sentence encoder trained on a synthetic dataset when evaluated on the dataset from different domains. In addition, our model outperforms other models on the PAWS dataset, and it shows that our method makes the model robust to adversarial attacks. In terms of the variants of our method, using the randomly sampled scalar value as an importance score usually degrades performance. The models that filter out unrealistic examples instead of reweighting them perform worse than RISE in most cases. Based on these observations, we confirm that training the model using RISE can enhance the reliability of the model trained on synthetic examples.

[6]Results on other STS tasks by training a regressor on top of frozen embeddings are presented in Appendix C.

C_{src}	Model	STSb		QQP		MRPC		PAWS
		r	ρ	Acc.	F1	Acc.	F1	F1
	GloVe	47.30	50.70	68.51	63.30	71.53	80.91	44.16
	BERT-CLS	17.18	20.30	66.38	61.50	66.03	79.79	49.32
	BERT	47.91	47.29	68.70	64.26	70.38	80.50	46.05
	BERT*	74.15	76.98	73.10	67.08	73.39	81.68	53.91
	RoBERTa	52.36	54.35	67.91	63.67	72.28	81.20	44.03
	RoBERTa*	74.78	77.80	73.56	67.00	75.76	82.46	56.48
	USE*	78.72	77.08	73.19	69.27	67.47	80.35	45.34
	InferSent*	49.53	50.86	68.94	64.13	65.97	79.32	45.01
STSb	DINO	78.45	77.71	73.14	68.04	70.44	81.16	47.30
	RISE	**79.11** (+0.66)	**78.57** (+1.46)	**74.47** (1.33)	**69.08** (+1.04)	**72.84** (+2.4)	**82.01** (+0.85)	**50.24** (+2.94)
	∟ Filtering	77.73 (-0.72)	77.45 (+0.34)	73.06 (-0.08)	67.94 (-0.10)	68.96 (-1.48)	81.35 (+0.19)	46.72 (-0.58)
	∟ Random	79.03 (+0.58)	78.39 (+1.28)	73.09 (-0.05)	68.03 (-0.01)	71.09 (+0.65)	81.62 (+0.46)	50.17 (+2.87)
QQP	DINO	64.93	65.93	73.20	67.72	70.75	80.40	44.47
	RISE	**78.36** (+13.43)	**77.13** (+11.2)	73.35 (+0.15)	67.76 (+0.04)	**72.38** (+1.63)	**81.35** (+0.95)	46.28 (+1.81)
	∟ Filtering	65.24 (+0.31)	66.36 (+0.43)	**73.48** (+0.28)	**67.95** (+0.23)	69.77 (-0.98)	80.26 (-0.14)	43.36 (-1.11)
	∟ Random	73.49 (+8.56)	72.88 (+6.95)	73.14 (-0.06)	67.75 (+0.03)	69.76 (-0.99)	80.83(+0.43)	**46.97** (+2.5)
MRPC	DINO	75.51	73.87	71.85	65.70	71.57	81.55	47.35
	RISE	**77.47** (+1.96)	**76.86** (+2.99)	**74.23** (+2.38)	**68.82** (+3.12)	71.97 (+0.4)	**81.95** (+0.4)	**49.35** (+2.00)
	∟ Filtering	76.25 (+0.74)	74.88 (+1.01)	71.05 (-0.80)	64.82 (-0.88)	71.34 (-0.23)	80.76 (-0.79)	47.84 (+0.49)
	∟ Random	76.06 (+0.55)	74.51 (+0.64)	72.52 (+0.67)	66.45 (+0.75)	**72.19** (+0.62)	81.71 (+0.16)	47.56 (+0.21)

Table 3: Evaluation results of different sentence embedding models on four sentence similarity task dataset. The models trained with human-annotated dataset (e.g., NLI) are marked with *. BERT and RoBERTa indicate sentence-BERT and sentence-RoBERTa, respectively. We highlight the best result in each pair of C_{src}/evaluation datasets and the best result in overall result in each metric as **bold** and underline, respectively. The number in right bracket indicates the performance difference with DINO. For regression task, we use Pearson correlation (r) and Spearman's rank correlation coefficient (ρ) metrics for evaluation. Each score represents the average of five trials.

7 Conclusions

In this paper, we demonstrated that the linguistic features of unrealistic machine-written sentences are different from those of human-written sentences. Based on this observation, we proposed a novel approach to reweight the loss based on the sentence importance from synthetic data identification (SDI) model for learning semantically meaningful embeddings. The extensive experiments show the effectiveness and robustness of RISE compared to other baseline approaches.

Although extensive experiments demonstrate the effectiveness of our method, adjustment of the importance of each sentence may learn an unintended bias from the classifier. In future work, we plan to conduct an in-depth human analysis for machine-written sentences to determine if our method correlates well with human judgement or not. Investigating the impact of unrealistic examples in other natural language applications would also be another interesting future direction.

Acknowledgements

This work was supported by Institute of Information & communications Technology Planning & Evaluation (IITP) grant funded by the Korea government(MSIT) (No. 2019-0-00075, Artificial Intelligence Graduate School Program(KAIST), and No. 2020-0-00368, A NeuralSymbolic Model for Knowledge Acquisition and Inference Techniques), and the National Research Foundation of Korea (NRF) grant funded by the Korean government (MSIT) (No. NRF-2019R1A2C4070420).

References

David Ifeoluwa Adelani, Haotian Mai, Fuming Fang, Huy H Nguyen, Junichi Yamagishi, and Isao Echizen. 2020. Generating sentiment-preserving fake online reviews using neural language models and their human-and machine-based detection. In *International Conference on Advanced Information Networking and Applications*, pages 1341–1354. Springer.

Tom B Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. 2020. Language models are few-shot learners. *arXiv preprint arXiv:2005.14165*.

Daniel Cer, Mona Diab, Eneko Agirre, Iñigo Lopez-Gazpio, and Lucia Specia. 2017. SemEval-2017 task 1: Semantic textual similarity multilingual and crosslingual focused evaluation. In *Proceedings of the 11th International Workshop on Semantic Evaluation (SemEval-2017)*, pages 1–14, Vancouver, Canada. Association for Computational Linguistics.

Daniel Cer, Yinfei Yang, Sheng-yi Kong, Nan Hua, Nicole Limtiaco, Rhomni St John, Noah Constant,

Mario Guajardo-Céspedes, Steve Yuan, Chris Tar, et al. 2018. Universal sentence encoder. *arXiv preprint arXiv:1803.11175*.

Ernie Chang, Xiaoyu Shen, Dawei Zhu, Vera Demberg, and Hui Su. 2021. Neural data-to-text generation with lm-based text augmentation. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*, pages 758–768.

Alexis Conneau and Douwe Kiela. 2018. Senteval: An evaluation toolkit for universal sentence representations. *arXiv preprint arXiv:1803.05449*.

Alexis Conneau, Douwe Kiela, Holger Schwenk, Loïc Barrault, and Antoine Bordes. 2017. Supervised learning of universal sentence representations from natural language inference data. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 670–680.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. Bert: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186.

Bosheng Ding, Linlin Liu, Lidong Bing, Canasai Kruengkrai, Thien Hai Nguyen, Shafiq Joty, Luo Si, and Chunyan Miao. 2020. DAGA: Data augmentation with a generation approach for low-resource tagging tasks. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 6045–6057, Online. Association for Computational Linguistics.

William B Dolan and Chris Brockett. 2005. Automatically constructing a corpus of sentential paraphrases. In *Proceedings of the Third International Workshop on Paraphrasing (IWP2005)*.

Aleksandra Edwards, Asahi Ushio, Jose Camacho-Collados, Hélène de Ribaupierre, and Alun Preece. 2021. Guiding generative language models for data augmentation in few-shot text classification. *arXiv preprint arXiv:2111.09064*.

Tianyu Gao, Xingcheng Yao, and Danqi Chen. 2021. Simcse: Simple contrastive learning of sentence embeddings. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 6894–6910.

Ari Holtzman, Jan Buys, Li Du, Maxwell Forbes, and Yejin Choi. 2019. The curious case of neural text degeneration. In *International Conference on Learning Representations*.

Aviral Kumar, Sunita Sarawagi, and Ujjwal Jain. 2018. Trainable calibration measures for neural networks from kernel mean embeddings. In *International Conference on Machine Learning*, pages 2805–2814. PMLR.

Jiwei Li, Michel Galley, Chris Brockett, Jianfeng Gao, and Bill Dolan. 2015. A diversity-promoting objective function for neural conversation models. *arXiv preprint arXiv:1510.03055*.

Ehsan Montahaei, Danial Alihosseini, and Mahdieh Soleymani Baghshah. 2019. Jointly measuring diversity and quality in text generation models.

Yannis Papanikolaou and Andrea Pierleoni. 2020. Dare: Data augmented relation extraction with gpt-2. *arXiv preprint arXiv:2004.13845*.

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. Bleu: a method for automatic evaluation of machine translation. In *Proceedings of the 40th annual meeting of the Association for Computational Linguistics*, pages 311–318.

Jeffrey Pennington, Richard Socher, and Christopher D Manning. 2014. Glove: Global vectors for word representation. In *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*, pages 1532–1543.

Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, Ilya Sutskever, et al. 2019. Language models are unsupervised multitask learners. *OpenAI blog*, 1(8):9.

Nils Reimers and Iryna Gurevych. 2019. Sentence-bert: Sentence embeddings using siamese bert-networks. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 3982–3992.

Timo Schick and Hinrich Schütze. 2021. Generating datasets with pretrained language models. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 6943–6951, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Adaku Uchendu, Thai Le, Kai Shu, and Dongwon Lee. 2020. Authorship attribution for neural text generation. In *Conf. on Empirical Methods in Natural Language Processing (EMNLP)*.

Max Weiss. 2019. Deepfake bot submissions to federal public comment websites cannot be distinguished from human submissions. *Technology Science*.

Yiben Yang, Chaitanya Malaviya, Jared Fernandez, Swabha Swayamdipta, Ronan Le Bras, Ji-Ping Wang, Chandra Bhagavatula, Yejin Choi, and Doug Downey. 2020. Generative data augmentation for commonsense reasoning. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 1008–1025, Online. Association for Computational Linguistics.

Mingyang Yi, Lu Hou, Lifeng Shang, Xin Jiang, Qun Liu, and Zhi-Ming Ma. 2021. Reweighting augmented samples by minimizing the maximal expected loss. In *Proc. the International Conference on Learning Representations (ICLR)*.

Kang Min Yoo, Dongju Park, Jaewook Kang, Sang-Woo Lee, and Woomyoung Park. 2021. GPT3Mix: Leveraging large-scale language models for text augmentation. In *Findings of the Association for Computational Linguistics: EMNLP 2021*, pages 2225–2239, Punta Cana, Dominican Republic. Association for Computational Linguistics.

Rowan Zellers, Ari Holtzman, Hannah Rashkin, Yonatan Bisk, Ali Farhadi, Franziska Roesner, and Yejin Choi. 2019. Defending against neural fake news. In *Proceedings of the 33rd International Conference on Neural Information Processing Systems*, pages 9054–9065.

Yuan Zhang, Jason Baldridge, and Luheng He. 2019. Paws: Paraphrase adversaries from word scrambling. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 1298–1308.

Appendix

A Detailed Analysis on Table 1

In this section, we present our detailed observations in Table 1 and the results of‘ the different N-gram in BLEU and Jaccard similarity. The results are presented in Table 4. We observe that the number of unique N-gram occurs frequently when $p_D(x_m)$ is high. In terms of lexical similarity (BLEU and Jaccard) with a paired sentences, the scores of synthetic sentences x_m with high $p_D(x_m)$ are higher about 20 points than those with low $p_D(x_m)$ and are similar to x_h. The distribution of word usage in generated sentences are also close to human-written sentences when predicted realistic score is high in two out of three datasets. Based on these observations, we confirm that even though the sentences are generated by the same machine in the same environment, there is a large variance in terms of how much the sentences are shifted from human sentences. Therefore, it is critical to handle the generated sentences carefully so that the model is not biased to the sentences that are very different from human-written sentences (*i.e.*, unrealistic samples).

B Qualitative Analysis

We qualitatively analyze the sentences that the SDI model classify as unrealistic, which include the bottom 10% ($p_D(x_m) \downarrow$) of the machine-written sentences based on their importance. In some cases, the SDI model correctly identifies them as unrealistic, and in some cases, it fails to identify them correctly as unrealistic.

As shown in Table 5, the unrealistic sentences identified by the SDI model contain repetition of the same expression or are incomplete. In addition, there were cases that contain a logical defect in the sentence. For example, as shown in the fifth example of Table 5, a person does not like movies, but in the next sentence, the machine generates a sentence that the person likes horror movies. In the sixth example of Table 5, a machine generates a sentence that a woman was born in two places. Furthermore, there are sentences with no grammatical or logical defects, but contain patterns that were not common in context. In the last example of Table 5, the contents of the defense budget and the individual budget are generated together, and it would not be usually used in reality. On the contrary, we find some examples that the SDI model classified as unrealistic sentences, but the sentences are realistic as shown in Table 6.

C Experiments on other STS tasks with Frozen Embeddings

Following previous studies (Reimers and Gurevych, 2019; Gao et al., 2021), we evaluate the quality of each sentence embedding by using it as a feature of a classifier. Specifically, we train a linear regressor on top of frozen sentence embeddings from each model for STS tasks. We use SentEval (Conneau and Kiela, 2018) framework on "test" setting. As shown in Table 7, We observe that the overall trends are consistent with the previous results in Table 3. RISE outperforms DINO in two source corpora (QQP and MRPC), while the results on STSb are sometimes unclear. Filtering out unrealistic examples performs worse than RISE in most cases. Finally, our model trained on STSb corpus achieves the best average score.

D Training Sentence Encoder for Sentence Similarity Task

Sentence similarity task aims to determine the similarity between two sentences. It can be formulated by classifying whether the two sentences are semantically similar or not or by measuring the distance between two sentences. A common and scalable approach for this task is based on Bi-encoder architecture (Reimers and Gurevych, 2019) which involves converting the sentences into embedding vectors and then measuring the similarity between sentences by calculating the distance between them in the embedding space.

More formally, given two sentences s_1 and s_2, and their ground truth similarity score y, a sentence encoder F encodes the sentences, s_1 and s_2, into their embedding vectors, e_1 and e_2, respectively. A distance metric d is then used to measure their similarity score $\hat{y}$, which is defined by:

$$\hat{y} = d(e_1, e_2). \tag{2}$$

This approach aims to predict the similarity score ($\hat{y}$) close to the ground-truth similarity score (y) by minimizing the mean squared error (MSE) which is given by:

$$\mathcal{L}(\theta_f) = \sum_{i=1}^{N} (\hat{y}_i - y_i)^2, \tag{3}$$

where θ_f is the parameter of embedding model F.

	STSb			QQP			MRPC		
	x_h	$p_D(x_m) \uparrow$	$p_D(x_m) \downarrow$	x_h	$p_D(x_m) \uparrow$	$p_D(x_m) \downarrow$	x_h	$p_D(x_m) \uparrow$	$p_D(x_m) \downarrow$
BLEU-1	51.02	40.87	7.53	45.94	46.88	13.46	61.86	59.17	15.19
BLEU-2	37.55	27.01	2.07	32.25	36.14	7.71	51.13	49.36	3.93
BLEU-3	28.51	19.88	1.20	24.19	30.49	5.68	43.57	42.42	1.92
BLEU-4	22.10	15.22	0.90	18.80	26.28	4.57	37.57	36.92	1.30
BLEU-N	34.80	25.75	2.93	30.3	34.95	7.86	48.53	46.97	5.59
Jaccard	41.98	33.97	5.98	39.91	42.49	11.31	53.55	53.33	10.52
Distinct-1	8.5	5.1	1.8	5.7	3.7	3.4	7.8	4.3	2.5
Distinct-2	49.7	36.5	15.0	39.5	25.5	23.4	48.7	31.4	20.1
Distinct-3	75.4	66.2	34.3	69.1	46.5	45.5	77.4	60.6	43.4
Distinct-N	44.53	35.93	17.03	38.10	25.23	24.10	44.63	32.10	22.00
Zipf coeff.	1.03	1.07	1.23	1.11	1.06	1.12	0.98	1.02	1.23

Table 4: Results for comparing the sentences in different group. Jaccard indicates Jaccard similarity score. The score of generated sentences that is far from human scores is highlighted in underline. For BLEU-N and Distinct-N, we report the average score with different N.

	$p_D(x_m) \downarrow$
Grammatical Error	x_h: Mitchell, Man of La Mancha, Malcolm Gets from Amour and John Selya, Movin' Out, complete the field. x_m: The other 30 percent is going to be a challenge, ($y = 0$)
	x_h: A black dog is walking on the grass with a stick in it's mouth. x_m: A brown dog walks with his stick in his mouth and it is brown in color. The black dog is walking with his stick in his mouth and it has a brown in color on it. ($y = 0.5$)
	x_h: "But the reality is that there needs to be a big structural change, " she added, "and you can't do that without funding." x_m: The fact is that there needs to be a big structural change, ($y = 1$)
Logical Error	x_h: Janet Racicot heard the thud from the kitchen, where she was getting a glass of water, she said in an interview Tuesday. x_m: I have been to a few movies and I am not a big movie fan. I like horror movies. ($y = 0$)
	x_h: The rate of survival without serious brain damage is about 10 percent, said Bossaert, a professor at the University Hospital in Antwerp, Belgium. x_m: The brain damage in the patient is not serious. The patient is a young woman who was born in the United States, but was born in Belgium. ($y = 0.5$)
Uncommon in context	x_h: It would be difficult to overestimate the potential dangers of the Remote Procedure Call (RPC) vulnerability. x_m: A senior official in the Oil ministry asked an oil company's CEO about his meeting with the minister. This is a very sensitive issue and is very sensitive to both sides. ($y = 0$)
	x_h: Hagel defends proposed cuts in defense spending. x_m: He's not going to cut the defense budget, but he is going to cut the domestic budget by more than half. ($y = 1$)

Table 5: Examples of machine-written sentences identified by the SDI model as unrealistic. The part of sentences that contains linguistic errors is highlighted in red.

$p_D(x_m) \downarrow$
x_h: The Company has always made, and continues to make, exceptional customer service and customer satisfaction a top priority in all business practices, "AOL added. x_m: AOL has always been committed to providing our customers with a wide range of services and solutions that are innovative and useful. ($y = 1$)
x_h: The two rugged countries got 2 to 3 inches of rain between midnight and noon. x_m: I was in a bad car accident last weekend. I have to go to the doctor next week to get a check up. ($y = 0$)
x_h: They appear in the Oct. 23 issue of The New England Journal of Medicine. x_m: It's not the case that all the research shows that marijuana has a positive impact on people's mental health. ($y = 0$)

Table 6: Examples of realistic machine-written sentences on which SDI model fails to identify as realistic.

E Datasets Details

As aforementioned in Section 3, STSb (Cer et al., 2017), QQP, and MRPC (Dolan and Brockett, 2005) are used to obtain a corpus of human-written sentences. The size of corpus $|C_{src}|$ is equally set to 10,000 across datasets. The set of similarity level Y is $\{0, 0.5, 1\}$. We generate samples from corpus

Sentence Textual Simiarlity benchmark(STSb) (Cer et al., 2018) consists of sentence pairs drawn from news, video and image captions, and natu-

C_{src}	Model	STS12	STS13	STS14	STS15	STS16	STS-B	SICK-R	Avg.
	GloVe*	52.24	49.91	43.36	55.91	47.67	46.00	55.02	50.01
	BERT	30.88	59.90	47.73	60.28	63.73	47.29	58.22	52.58
	BERT*	70.97	76.53	73.19	79.09	74.30	76.98	72.91	74.85
	RoBERTa	32.10	56.33	45.22	61.34	61.98	55.39	62.03	53.48
	RoBERTa*	70.92	73.03	70.79	78.37	73.68	77.33	74.40	74.07
	USE*	67.06	71.55	70.59	80.27	75.76	76.85	69.31	73.05
	InferSent*	56.15	69.57	64.03	74.06	72.00	72.06	66.77	67.80
STSb	DINO	69.89	79.52	70.91	79.51	**79.14**	77.67	64.77	74.49
	RISE	69.79(-0.1)	81.09(+1.57)	72.15(+1.24)	**81.04**(+1.53)	79.05(-0.09)	78.07(+0.4)	**72.21**(+7.44)	**76.20**(+1.71)
	∟ Filtering	67.04(-2.85)	77.03(-2.49)	69.54(-1.37)	77.81(-1.70)	76.63(-2.51)	75.99(-1.68)	65.19(+0.42)	72.75(-1.74)
	∟ Random	**70.03**(+0.14)	**81.28**(+1.76)	**72.63**(+1.72)	79.02(-0.49)	78.87(-0.27)	**78.68**(+1.01)	66.89(+2.12)	75.34(+0.85)
QQP	DINO	56.93	71.39	59.75	67.59	73.10	68.09	61.48	65.48
	RISE	**59.11**(+2.18)	**78.11**(+6.72)	**70.17**(+10.42)	**77.48**(+9.89)	**78.70**(+5.6)	**77.89**(+9.8)	**71.59**(+10.11)	**73.29**(+7.81)
	∟ Filtering	58.30(+1.37)	72.32(+0.93)	62.00(+2.25)	69.76(+2.17)	73.70(+0.6)	71.36(+3.27)	62.17(+0.69)	67.09(+1.61)
	∟ Random	56.80(-0.13)	71.17(-0.22)	59.64(-0.11)	68.32(+0.73)	72.42(-0.68)	69.71(+1.62)	65.77(+4.29)	66.26(+0.78)
MRPC	DINO	60.74	73.11	61.38	70.95	74.85	73.61	67.70	68.91
	RISE	**66.17**(+5.43)	**77.41**(+4.3)	**68.56**(+7.18)	**76.64**(+5.69)	**76.93**(+2.08)	**76.39**(+2.78)	**71.93**(+4.23)	**73.43**(+4.52)
	∟ Filtering	59.86(-0.88)	74.76(+1.65)	62.43(+1.05)	72.74(+1.79)	75.07(+0.22)	73.25(-0.36)	69.48(+1.78)	69.66(+0.75)
	∟ Random	64.36(+3.62)	76.02(+2.91)	64.62(+3.24)	73.24(+2.29)	76.01(+1.16)	75.36(+1.75)	70.78(+3.08)	71.48(+2.57)

Table 7: Evaluation results of frozen sentence embedding models on STS tasks. The linear regressor is trained on top of sentence embeddings from each model. The number in right bracket indicates the performance difference with DINO. We highlight the best result in each pair of C_{src}/evaluation datasets and the best result in overall result in each metric as **bold** and underline, respectively. For regression tasks, we use Spearman's rank correlation coefficient (ρ) as an evaluation metric.

Data	STSb	QQP	MRPC	PAWS-QQP
X_m^{train}	76.9k	78.2k	55.3k	-
X_m^{dev}	59.2k	78.3k	6.3k	-
X_{src}^{dev}	1.5k	18.1k	0.4k	0.3k
X_{src}^{test}	1.4k	40.4k	1.7k	0.3k

Table 8: Dataset statistics. The class distribution of MRPC, QQP, and PAWS-QQP is imbalanced.

Hyperparameter	STSb	QQP	MRPC
batch size	32	32	32
learning rate	2e-5	2e-5	2e-5
number of epochs	3	3	3
temperature τ	0.5	0.9	0.7

Table 9: Hyperparameters used in experiments. We conduct grid search to find the best hyperparameter settings.

ral language inference data. Each pair is human-annotated with a continuous score from 1 to 5; the task is to predict these scores. In this experiment, we normalize the original similarity score to have from 0 to 1. We evaluate using Pearson and Spearman correlation coefficients.

Quora Question Pairs(QQP) [7] consists of question pairs from the community Quora. The task is to classify that a pairs of question have semantically same meaning.

Microsoft Research Paraphrase Corpus(MRPC) (Dolan and Brockett, 2005) is a corpus of sentence pairs from online news sources, with human annotations for whether the sentences in the pair are semantically same. The class have the imbalanced distribution.(68% positive).

Paraphrase Adversaries from Word Scrambling of Quora Question (PAWS-QQP) (Zhang et al., 2019) contains human-labeled and noisily labeled pairs that feature the importance of modeling structure, context, and word order information for the problem of paraphrase identification. The dataset has two subsets, one based on Wikipedia and the other one based on the Quora Question Pairs (QQP) dataset. In this paper, we only use examples based on QQP. The class have the imbalanced distribution.(31.3% positive).

F Training Details

Implementation Details All experiments in Table 2 in the main paper is implemented in Ubuntu 18.04.4 LTS, 3090 RTX GPU with 24GB of memory, and AMD EPYC 7702. The version of libraries we experiment are 3.8 for python and 1.4.0 for pytorch. We implemented all models with PyTorch using Sentence-Transformers[8] library from Ubiquitous Knowledge Processing Lab.

Training and Evaluation. We train a model to solve the sentence similarity task as a regression problem. However, since all the datasets except for STSb only contain discrete labels, we set the

[7] https://quoradata.quora.com/First-Quora-Dataset-Release-Question-Pairs

[8] https://github.com/UKPLab/sentence-transformers

threshold using validation dataset to make binary decision. Training a model takes 5 minutes per epoch.

Hyperparameter Details The DINO are reproduced as described in the previous works. To compute sentence simiarity score, we use cosine similarity as distance metric. We search the best hyperparameters using grid search. During the prediction of SDI model, we use use the temperature scaling (τ) (Kumar et al., 2018) is applied before softmax function. The best hyperparameters for each dataset of **RISE** are described in Table 9.

MaxMatch-Dropout: Subword Regularization for WordPiece

Tatsuya Hiraoka
Tokyo Institute of Technology*
hiraoka.tatsuya@fujitsu.com

Abstract

We present a subword regularization method for WordPiece, which uses a maximum matching algorithm for tokenization. The proposed method, MaxMatch-Dropout, randomly drops words in a search using the maximum matching algorithm. It realizes finetuning with subword regularization for popular pretrained language models such as BERT-base. The experimental results demonstrate that MaxMatch-Dropout improves the performance of text classification and machine translation tasks as well as other subword regularization methods. Moreover, we provide a comparative analysis of subword regularization methods: subword regularization with SentencePiece (Unigram), BPE-Dropout, and MaxMatch-Dropout.

1 Introduction

Subword regularization (Kudo, 2018) is a well-known technique for improving the performance of NLP systems, whereby a model is trained with various tokenizations that are sampled for each training epoch. This approach provides data augmentation and model robustness against tokenization differences.

Kudo (2018) first introduced subword regularization using a unigram language model that was included in their tokenization tool, namely SentencePiece (Kudo and Richardson, 2018), and reported its effectiveness on machine translation tasks. Provilkov et al. (2020) proposed a subword regularization method for byte pair encoding (BPE) known as BPE-Dropout and demonstrated the superiority of their method over that using the unigram language model in machine translation tasks. Moreover, subword regularization contributes to the performance improvement of text classification tasks (Hiraoka et al., 2019).

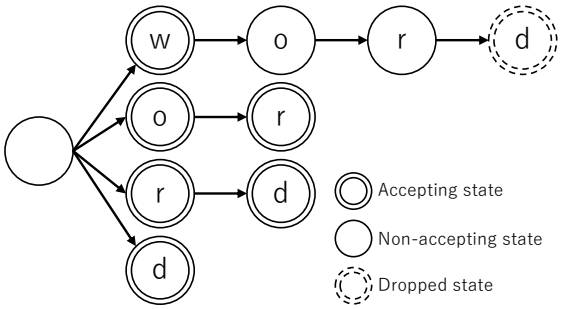

Figure 1: MaxMatch-Dropout randomly removes accepting states in the trie. In this figure, a state corresponding to "word" is dropped and a single input "word" is tokenized as "w, or, d."

As subword regularization is implemented as a modification of a tokenizer, each method is specialized to a particular tokenizer type. For example, the original subword regularization (Kudo, 2018) is specialized to a tokenizer that uses the unigram language model and BPE-Dropout is specialized to the BPE-based tokenizer. However, these existing subword regularization tools cannot be directly applied to the other common tokenizers such as WordPiece (Song et al., 2021).

WordPiece is a tokenizer that is based on the maximum matching algorithm. It is used as the default tokenizer for the popular pretrained language model BERT (Devlin et al., 2018). Although the widely used BERT models (e.g., BERT-base) can improve the performance of various NLP tasks, subword regularization cannot be used for the finetuning of the model because no subword regularization method exists for WordPiece. The use of subword regularization for the finetuning of pretrained models with WordPiece may result in a further performance improvement.

In this paper, we present a simple modification of WordPiece for the use of subword regularization. The proposed method, which is known as MaxMatch-Dropout, randomly drops words in a vocabulary during the tokenization process. That

*The author is currently affiliated with Fujitsu Limited. This work was carried out at the Tokyo Institute of Technology.

Proceedings of the 29th International Conference on Computational Linguistics, pages 4864–4872
October 12–17, 2022.

Algorithm 1 Algorithm for Word Tokenization

Require: Single Word w, Vocabulary V, Dropout Rate q.

$S \leftarrow$ Empty List
Index of Characters $i \leftarrow 1$
while $i < |w|$ **do**
  Subword $s \leftarrow \varnothing$
  for $j = 1$ to $|w| - i$ **do**
    if $w_{i:i+j} \in V$ and $\mathrm{Ber}(1-q)$ **then**
      $s \leftarrow w_{i:i+j}$
  if $s = \varnothing$ **then return** [UNK]
  else
    Add s to S
    $i \leftarrow i + |s|$
return S

is, MaxMatch-Dropout randomly removes accepting states from a trie for tokenization. The experimental results demonstrate that MaxMatch-Dropout improves the performance of text classification and machine translation in several languages, as well as other subword regularization methods. Furthermore, MaxMatch-Dropout contributes to a further performance improvement with pretrained BERT on text classification in English, Korean, and Japanese.

2 Maximum Matching

A simple modification to the maximum matching algorithm is implemented so that MaxMatch-Dropout can realize subword regularization. Prior to explaining the modification, we briefly review the maximum matching on which the proposed method is based[1].

Given a vocabulary and a single word, the maximum matching searches the longest subword in the vocabulary and greedily tokenizes the word into a sequence of subwords from beginning to end. For example, let the vocabulary be composed of {a, b, c, d, abc, bcd}. The tokenizer with the maximum matching divides a word "abcd" into "abc, d"[2]. As the maximum matching searches subwords from the beginning of the word, this word is not tokenized as "a, bcd." When an input word includes an unknown character, such as "abce," the tokenizer replaces this word with a special token, "[UNK]." This tokenization process is usually implemented using a trie. The detailed tokenization process using the maximum matching for this example with the trie (Figure 4) is described in Appendix A.

[1] Song et al. (2021) explains the efficient implementation of the maximum matching in detail.

[2] We do not use special tokens for a subword that begins in the middle of a word (e.g., "##") for simple explanation.

3 Proposed Method: MaxMatch-Dropout

The proposed method extends the maximum matching with an additional dropout process. This method randomly replaces accepting states into non-accepting states with **dropped states**. That is, accepting tokens are randomly skipped with a specified probability q, where q is a hyperparameter.

Figure 1 depicts the tokenization process of a word "word" with a vocabulary that includes {w, o, r, d, or, rd, word}. Although the maximum matched subword beginning with the first character is "word" in the vocabulary, in this case, the state corresponding to "word" is dropped. Thus, the latest accepted subword "w" is yielded and the next matching begins from the second character. Finally, the tokenization process results in "w, or, d."

This process is also outlined in Algorithm 1 [3]. In the algorithm, $w_{i,i+j}$ denotes a subword beginning from the i-th character and ending with the $(i+j-1)$-th character in the word w, where $|w|$ and $|s|$ are the lengths of the input word and subword, respectively. Moreover, $Ber(1-q)$ denotes a Bernoulli distribution that returns 1 with a probability of $1-q$.

The tokenization process of MaxMatch-Dropout is detailed in Table 6 of Appendix A. The difference between MaxMatch-Dropout and the original maximum matching can be observed by comparing Tables 5 and 6.

The regularization strength can be tuned using the hyperparameter q. The proposed method is equivalent to the original maximum matching with $q = 0.0$, and it tokenizes a word into characters with $q = 1.0$ if all characters are included in the vocabulary.

The official code is available at `https://github.com/tatHi/maxmatch_dropout`.

4 Experiments

We conducted experiments on text classification and machine translation tasks to validate the performance improvement provided by MaxMatch-Dropout.

We used two tokenizers and subword regularization methods as a reference for both tasks: SentencePiece (Unigram) (Kudo and Richardson, 2018) with subword regularization (Sub. Reg.) (Kudo,

[3] Algorithm 1 does not use a trie for simple explanation.

	English							Korean			Japanese	
	APG	APR	TS	QNLI	QQP	RTE	SST-2	NLI	STS	YNAT	TR	WRIME
$\|V\|$	32K	32K	32K	32K	32K	12K	8K	24K	16K	32K	16K	12K
Metric	F1	F1	F1	Acc.	F1	Acc.	Acc.	Acc.	F1	F1	F1	F1
BiLSTM												
Unigram	69.05	65.85	76.21	66.48	83.61	49.10	80.05	41.93	67.02	68.57	86.6	46.36
+ Sub. Reg.	**70.65**	**66.80**	**77.49**	**66.56**	**83.91**	**53.31**	**83.30**	**42.84**	**68.08**	**73.67**	**87.11**	**49.47**
BPE	67.10	64.67	75.24	**67.11**	**82.82**	53.07	78.10	41.22	**67.42**	64.27	84.95	44.34
+ BPE-Dropout	**68.45**	**65.38**	**76.04**	66.69	82.69	**53.97**	**82.00**	**41.52**	66.26	**69.12**	**85.68**	**46.01**
WordPiece	63.17	62.97	73.14	64.04	82.11	53.55	81.04	39.96	61.75	62.44	84.95	46.36
+ MM-Dropout	**64.90**	**64.36**	**75.22**	**64.28**	**82.14**	**53.91**	**83.75**	**40.61**	**62.88**	**70.08**	**86.98**	**47.28**
BERT												
WordPiece	77.28	70.99	81.93	89.45	89.83	62.00	90.97	82.18	83.22	83.96	89.08	89.08
+ MM-Dropout	**78.55**	**71.68**	**82.08**	**89.74**	**89.86**	**62.27**	**91.07**	**82.19**	**85.43**	**84.31**	**89.14**	**89.14**

Table 1: Experimental results of text classification (averaged scores of five runs). The higher scores for the tokenizations with/without subword regularization are indicated in bold. The scores that significantly surpassed the results without subword regularization ($p < 0.05$, McNemar's test) are underlined.

2018) and BPE (Sennrich et al., 2016) with BPE-Dropout (Provilkov et al., 2020). We employed WordPiece (Song et al., 2021), which was implemented by HuggingFace (Wolf et al., 2020), as a basic tokenizer for the proposed MaxMatch-Dropout [4].

We set the vocabulary size of each tokenizer to be equal to compare the three methods as fairly as possible. The vocabulary of each tokenizer included all characters that appeared in the training splits. We selected the hyperparameters for the subword regularization (e.g., q of MaxMatch-Dropout) according to the performance on the development splits. Note that we could not fairly compare the performance of MaxMatch-Dropout to that of other subword regularization methods because they are based on different tokenizers and vocabularies. WordPiece was used as the baseline for MaxMatch-Dropout to investigate whether the method could successfully perform subword regularization and improve the performance similarly to other methods.

4.1 Text Classification

Datasets We exploited text classification datasets in three languages: English, Korean, and Japanese. **APG** and **APR** are genre prediction and rating prediction, respectively, on review texts that were created from the Amazon Product Dataset (He and McAuley, 2016). **TS** is a sentiment classification for tweets [5]. We also employed **QNLI** (Rajpurkar et al., 2016), **QQP** (Chen et al., 2018), **RTE** (Bentivogli et al.), and **SST-2** (Socher et al., 2013) from the GLUE benchmark (Wang et al., 2018). **NLI**, **STS**, and **YNAT** are text classification datasets that are included in Korean GLUE (KLUE) (Park et al., 2021). **TR** (Suzuki, 2019) and **WRIME** (Kajiwara et al., 2021) are sentiment classification datasets for tweets in Japanese. We used the original development sets as test sets and exploited a randomly selected 10% of the original training sets as development sets for the datasets in GLUE and KLUE owing to the numerous experimental trials.

Setup We used two backbones for the text classification: BiLSTM (Hochreiter and Schmidhuber, 1997; Graves and Schmidhuber, 2005) and BERT (Devlin et al., 2018). We employed BERT-base-cased[6], BERT-kor-base[7](Kim, 2020), and BERT-base-Japanese-v2[8] for the English, Korean, and Japanese datasets, respectively. All of these BERT models employ WordPiece as their tokenizers, and we finetuned them using MaxMatch-Dropout. We set the maximum number of training epochs to 20 for BiLSTM and the finetuning epochs to 5 for BERT. The trained model with the highest score in the development split was selected and evaluated on the test split. We selected the vocabulary sizes according to the performance on the development splits when using WordPiece without MaxMatch-Dropout. The selected vocabulary sizes were applied to all tokenizers.

Results Table 1 presents the experimental results for the text classification. The table demon-

[4] Table 12 in the Appendix presents tokenization examples for each tokenizer.

[5] https://www.kaggle.com/c/twitter-sentiment-analysis2

[6] https://huggingface.co/bert-base-cased

[7] https://huggingface.co/kykim/bert-kor-base

[8] https://huggingface.co/cl-tohoku/bert-base-japanese-v2

	IWSLT14		IWSLT15			
	DeEn	EnDe	ViEn	EnVi	ZhEn	EnZh
Unigram	36.55	27.89	30.28	29.39	22.64	20.55
+ Sub. Reg.	**38.50**	**29.45**	**31.58**	**30.96**	**23.81**	**21.79**
BPE	35.77	27.87	30.05	29.25	18.80	20.61
+ BPE-Dropout	**37.81**	**29.15**	**31.39**	**31.23**	**20.67**	**22.02**
WordPiece	36.22	27.58	30.13	29.40	17.24	20.45
+ MM-Dropout	**38.30**	**29.54**	**31.71**	**31.14**	**18.21**	**21.55**

Table 2: Experimental results of machine translation (averaged scores of three runs). ScareBLEU (Post, 2018) was used as the metric. Scores that significantly surpassed the results without subword regularization ($p < 0.05$, bootstrap resampling (Koehn et al., 2007)) are underlined.

strates that MaxMatch-Dropout (MM-Dropout) improved the performance as well as the other subword regularization methods. In addition to the improvement in the BiLSTM-based classifiers, MaxMatch-Dropout enhanced the performance of the BERT-based classifiers. These results indicate that MaxMatch-Dropout is a useful subword regularization method for WordPiece as well as effective for BERT.

4.2 Machine Translation

Datasets We employed three language pairs for the machine translation tasks: the De-En, Vi-En, and Zh-En pairs from the IWSLT corpora. We selected these datasets because subword regularization is particularly efficient in low-resource environments (Kudo, 2018; Hiraoka et al., 2021; Takase et al., 2022).

Setup We applied the Transformer (Vaswani et al., 2017), which was implemented by Fairseq (Ott et al., 2019), for the IWSLT settings. We trained the model with 100 epochs and averaged the parameters of the final 10 epochs. We evaluated the performance on the Chinese dataset using character-level BLEU. Following Provilkov et al. (2020), we set the vocabulary size to 4K for English, German, and Vietnamese, and 16K for Chinese.

Results Table 2 displays the experimental results for the machine translation. The table demonstrates that MaxMatch-Dropout improved the performance in all language pairs. The results indicate that the proposed method is effective for machine translation as well as existing subword regularization methods.

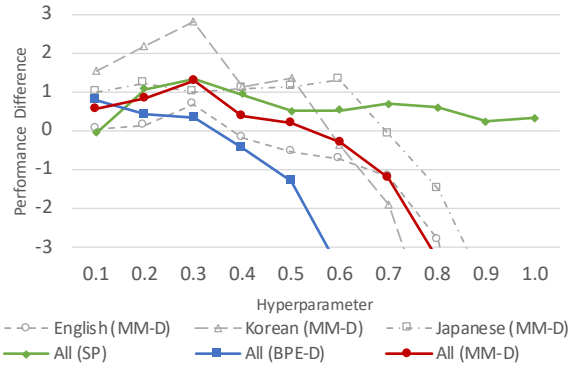

Figure 2: Performance differences with and without subword regularization against hyperparameters and for different languages on text classification datasets. MM-D, SP, and BPE-D denote MaxMatch-Dropout, SentencePiece (Unigram), and BPE-Dropout, respectively.

5 Discussion

5.1 Effect of Hyperparameters

Figure 2 depicts the averaged performance improvement over several text classification datasets against different hyperparameters. The figure indicates that the subword regularization of SentencePiece (Unigram) was the most robust against the hyperparameters among the three methods. Although both BPE-Dropout and MaxMatch-Dropout could realize subword regularization using the dropout technique for the tokenization strategy, MaxMatch-Dropout was more robust against the hyperparameters than BPE-Dropout. This result demonstrates that a performance improvement can be achieved in WordPiece-based systems using MaxMatch-Dropout with approximately selected hyperparameters (e.g., $q < 0.5$).

Figure 2 also shows the averaged performance on the datasets in each language against the hyperparameters of MaxMatch-Dropout (dashed lines). It can be observed that MaxMatch-Dropout was more effective for Asian languages than English. It is considered that this is because Korean and Japanese contain various types of n-grams and many tokenization candidates exist for a single sentence compared to English.

5.2 Token Length

In this subsection, we analyze the token length in the sampled tokenizations. We sampled the tokenization of the training dataset (APG) with three subword regularization methods and counted the token lengths for 10 trials.

Figure 3 presents the frequency of token lengths

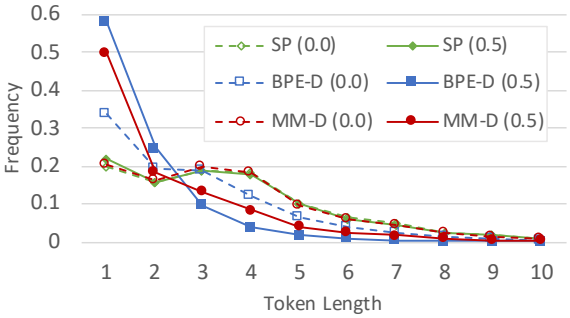

Figure 3: Frequency of token lengths with each subword regularization method on APG dataset (English). 0.0 denotes the vanilla settings without subword regularization. 0.5 indicates subword regularization when the hyperparameter was 0.5 (e.g., $q = 0.5$). MM-D, SP, and BPE-D denote MaxMatch-Dropout, SentencePiece (Unigram), and BPE-Dropout, respectively.

in the tokenized training datasets with/without subword regularization. The figure indicates that the length frequency did not change, regardless of the use of subword regularization, when SentencePiece (Unigram) was applied. In contrast, both MaxMatch-Dropout (MM-D) and BPE-Dropout (BPE-D) yielded many characters when the hyperparameter was 0.5, because they are based on the token-level dropout and yield characters when the hyperparameter is 1.0. However, the frequency curve of MaxMatch-Dropout was gentler than that of BPE-Dropout. We believe that this tendency aided in the robustness of the MaxMatch-Dropout performance, as reported in Section 5.1.

6 Conclusion

We have introduced a subword regularization method for WordPiece, which is a common tokenizer for BERT. The proposed method, MaxMatch-Dropout, modifies the tokenization process using the maximum matching to drop words in the vocabulary randomly. This simple modification can realize subword regularization for WordPiece. Furthermore, the experimental results demonstrated that MaxMatch-Dropout can improve the performance of BERT. MaxMatch-Dropout is also effective in the training of text classification tasks without BERT and machine translation tasks, as well as existing subword regularization methods.

Acknowledgement

This work was supported by JST, ACT-X Grant Number JPMJAX21AM, Japan.

References

Luisa Bentivogli, Peter Clark, Ido Dagan, and Danilo Giampiccolo. The fifth pascal recognizing textual entailment challenge.

Kaj Bostrom and Greg Durrett. 2020. Byte pair encoding is suboptimal for language model pretraining. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 4617–4624.

Zihan Chen, Hongbo Zhang, Xiaoji Zhang, and Leqi Zhao. 2018. Quora question pairs. *University of Waterloo*, pages 1–7.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*.

Alex Graves and Jürgen Schmidhuber. 2005. Framewise phoneme classification with bidirectional lstm networks. In *Proceedings. 2005 IEEE International Joint Conference on Neural Networks, 2005.*, volume 4, pages 2047–2052. IEEE.

Ruining He and Julian McAuley. 2016. Ups and downs: Modeling the visual evolution of fashion trends with one-class collaborative filtering. In *proceedings of the 25th international conference on world wide web*, pages 507–517.

Tatsuya Hiraoka, Hiroyuki Shindo, and Yuji Matsumoto. 2019. Stochastic tokenization with a language model for neural text classification. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 1620–1629.

Tatsuya Hiraoka, Sho Takase, Kei Uchiumi, Atsushi Keyaki, and Naoaki Okazaki. 2021. Joint optimization of tokenization and downstream model. In *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*, pages 244–255.

Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long short-term memory. *Neural computation*, 9(8):1735–1780.

Tomoyuki Kajiwara, Chenhui Chu, Noriko Takemura, Yuta Nakashima, and Hajime Nagahara. 2021. WRIME: A new dataset for emotional intensity estimation with subjective and objective annotations. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 2095–2104, Online. Association for Computational Linguistics.

Kiyoung Kim. 2020. Pretrained language models for korean. https://github.com/kiyoungkim1/LMkor.

Philipp Koehn, Hieu Hoang, Alexandra Birch, Chris Callison-Burch, Marcello Federico, Nicola Bertoldi, Brooke Cowan, Wade Shen, Christine Moran, Richard Zens, et al. 2007. Moses: Open source

toolkit for statistical machine translation. In *Proceedings of the 45th annual meeting of the ACL on interactive poster and demonstration sessions*, pages 177–180. Association for Computational Linguistics.

Taku Kudo. 2018. Subword regularization: Improving neural network translation models with multiple subword candidates. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 66–75.

Taku Kudo and John Richardson. 2018. Sentencepiece: A simple and language independent subword tokenizer and detokenizer for neural text processing. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 66–71.

Xing Niu, Prashant Mathur, Georgiana Dinu, and Yaser Al-Onaizan. 2020. Evaluating robustness to input perturbations for neural machine translation. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 8538–8544.

Myle Ott, Sergey Edunov, Alexei Baevski, Angela Fan, Sam Gross, Nathan Ng, David Grangier, and Michael Auli. 2019. fairseq: A fast, extensible toolkit for sequence modeling. In *Proceedings of NAACL-HLT 2019: Demonstrations*.

Sungjoon Park, Jihyung Moon, Sungdong Kim, Won Ik Cho, Ji Yoon Han, Jangwon Park, Chisung Song, Junseong Kim, Youngsook Song, Taehwan Oh, et al. 2021. Klue: Korean language understanding evaluation. In *Thirty-fifth Conference on Neural Information Processing Systems Datasets and Benchmarks Track (Round 2)*.

Matt Post. 2018. A call for clarity in reporting BLEU scores. In *Proceedings of the Third Conference on Machine Translation: Research Papers*, pages 186–191, Brussels, Belgium. Association for Computational Linguistics.

Ivan Provilkov, Dmitrii Emelianenko, and Elena Voita. 2020. BPE-dropout: Simple and effective subword regularization. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 1882–1892, Online. Association for Computational Linguistics.

Pranav Rajpurkar, Jian Zhang, Konstantin Lopyrev, and Percy Liang. 2016. Squad: 100,000+ questions for machine comprehension of text. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 2383–2392.

Mike Schuster and Kaisuke Nakajima. 2012. Japanese and korean voice search. In *2012 IEEE international conference on acoustics, speech and signal processing (ICASSP)*, pages 5149–5152. IEEE.

Rico Sennrich, Barry Haddow, and Alexandra Birch. 2016. Neural machine translation of rare words with subword units. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, volume 1, pages P1715–1725.

Richard Socher, Alex Perelygin, Jean Wu, Jason Chuang, Christopher D Manning, Andrew Y Ng, and Christopher Potts. 2013. Recursive deep models for semantic compositionality over a sentiment treebank. In *Proceedings of the 2013 conference on empirical methods in natural language processing*, pages 1631–1642.

Xinying Song, Alex Salcianu, Yang Song, Dave Dopson, and Denny Zhou. 2021. Fast wordpiece tokenization. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 2089–2103.

Yu Suzuki. 2019. Filtering method for twitter streaming data using human-in-the-loop machine learning. *Journal of Information Processing*, 27:404–410.

Sho Takase, Tatsuya Hiraoka, and Naoaki Okazaki. 2022. Single model ensemble for subword regularized models in low-resource machine translation. In *Findings of the Association for Computational Linguistics: ACL 2022*, pages 2536–2541.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. *Advances in neural information processing systems*, 30:5998–6008.

Alex Wang, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel Bowman. 2018. Glue: A multi-task benchmark and analysis platform for natural language understanding. In *Proceedings of the 2018 EMNLP Workshop BlackboxNLP: Analyzing and Interpreting Neural Networks for NLP*, pages 353–355.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Rémi Louf, Morgan Funtowicz, Joe Davison, Sam Shleifer, Patrick von Platen, Clara Ma, Yacine Jernite, Julien Plu, Canwen Xu, Teven Le Scao, Sylvain Gugger, Mariama Drame, Quentin Lhoest, and Alexander M. Rush. 2020. Transformers: State-of-the-art natural language processing. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 38–45, Online. Association for Computational Linguistics.

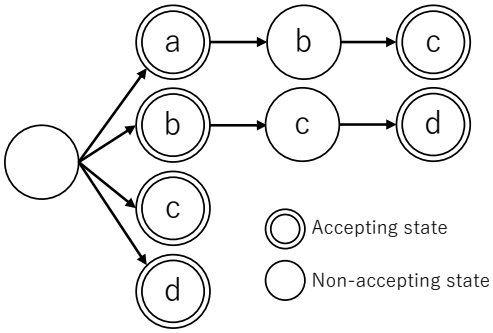

Figure 4: Trie for vocabulary including tokens {a, b, c, d, abc, bcd}.

Read	Action	Output
a	Accept "a"	
b	Non-accept "ab"	
c	Accept "abc"	
d	Reject the transition to "abcd"	
	& Yield the latest subword	abc
d	Accept "d"	
$	Reject the transition to "d$"	
	& Yield the latest subword	abc, d

Table 3: Operation for tokenizing input word "abcd" into "abc, d" using trie shown in Figure 4. "$" denotes a special symbol indicating the end of the word.

Read	Action	Output
a	Accept "a"	
b	Non-accept "ab"	
c	Accept "abc"	
e	Reject the transition to "abcd"	
	& Yield the latest subword	abc
e	Detect an OOV character	
	& Output [UNK]	[UNK]

Table 4: Operation for tokenizing input word "abce" including out-of-vocabulary (OOV) character into "[UNK]" using trie shown in Figure 4.

Read	Action	Output
w	Accept "w"	
o	Non-accept "wo"	
r	Non-accept "wor"	
d	Accept "word"	
$	Reject the transition to "word$"	
	& Yield the latest subword	word

Table 5: Operation for tokenizing input word "word" by applying original maximum matching (i.e., the operation without any dropout process) for trie shown in Figure 1. "$" denotes a special symbol indicating the end of the word.

A Maximum Matching in Detail

As described in Section 2, a trie is generally used to tokenize an input word with the maximum matching algorithm. Figure 4 depicts the trie corresponding to the vocabulary that includes six tokens: {a, b, c, d, abc, bcd}. The tokenization process using this trie for the input words "abcd" and "abce" is presented in Tables 3 and 4, respectively.

Table 6 details the operation for tokenizing an input word "word" into "w, or, d" using the proposed MaxMatch-Dropout, as outlined in Section 3. Table 5 describes the tokenization process using the original maximum matching for Figure 1 without the dropout process. Therefore, the difference in the tokenization process between the original maximum matching and MaxMatch-Dropout can be observed by comparing Tables 5 and 6.

B Related Work

This work is related to tokenization methods, which split raw texts into a sequence of tokens. Three well-known tokenization methods have been employed in recent NLP systems: SentencePiece (Unigram) (Kudo and Richardson, 2018), BPE (Sennrich et al., 2016), and WordPiece (Song et al., 2021). SentencePiece (Unigram) is a unigram language model-based tokenizer, whereas BPE employs a frequency-based tokenization technique. Although both methods are used extensively in many NLP systems, Bostrom and Durrett (2020) reported that the unigram language model-based tokenizer (i.e., SentencePiece (Unigram)) is superior to BPE in several downstream tasks. Our experimental results in Tables 1 and 2 also support this finding.

WordPiece[9] is another famous tokenizer that is mainly employed by large pretrained models such as BERT (Devlin et al., 2018). As WordPiece is based on the maximum matching algorithm, it is superior to other tokenization methods in terms of the tokenization speed. In fact, WordPiece is employed in real NLP systems such as Google searching (Song et al., 2021). However, the experimental results in this study (Table 1 and 2) demonstrated that WordPiece is inferior to SentencePiece (Unigram) and BPE in terms of performance. The proposed method can compensate for this shortcoming without decreasing the inference speed.

Kudo (2018) introduced a subword regularization technique for SentencePiece (Unigram) using dynamic programming. Provilkov et al. (2020) proposed a subword regularization method for BPE using the dropout technique. Niu et al. (2020) inves-

[9] Although the original term "wordpiece" indicates BPE-based tokenization (Schuster and Nakajima, 2012), in this paper, "WordPiece" indicates a tokenizer with the maximum matching for BERT following Song et al. (2021).

Read	Action	Output
w	Accept "w"	
o	Non-accept "wo"	
r	Non-accept "wor"	
d	**(Randomly) Non-accept "word"**	
$	Reject the transition to "word$"	
	& Yield the latest subword	w
o	Accept "o"	
r	Accept "or"	
d	Reject the transition to "ord"	
	& Yield the latest subword	w, or
d	Accept "d"	
$	Reject the transition to "d$"	
	& Yield the latest subword	w, or, d

Table 6: Operation for tokenizing input "word" using trie for MaxMatch-Dropout shown in Figure 1. "$" denotes a special symbol indicating the end of the word.

tigates these two methods in machine translation. This study has introduced a subword regularization method for WordPiece, and presented an in-depth investigation of the three methods in text classification and machine translation.

C Contributions

This study contributes to the NLP community in terms of the following two main points:

- A subword regularization method for WordPiece is proposed, which improves the text classification and machine translation performance.
- An intensive performance investigation of the three famous tokenization and subword regularization methods used in NLP (i.e., SentencePiece (Unigram), BPE, and WordPiece with subword regularization) is presented.

D Dataset Statistics

Table 7 displays the detailed information of the datasets. We report the numbers of samples in the training, development, and test splits. Furthermore, we present the number of label types for text classification datasets.

E Detailed Experimental Settings

Tables 8 and 9 present the detailed settings of the backbone models that were used in text classification and machine translation tasks, respectively. We used the default values of PyTorch for the hyperparameters that are not described in these tables. We set the number of tokenization candidates to ∞ for the subword regularization of SentencePiece (Unigram).

Dataset	Train	Dev.	Test	Labels
English Text Classification				
APG	96,000	12,000	12,000	24
APR	96,000	12,000	12,000	5
TS	80,000	10,000	10,000	2
QNLI	188,536	10,475	5,463	2
QQP	327,461	36,385	40,430	2
RTE	2,241	249	277	2
SST-2	60,614	6,735	872	2
Korean Text Classification				
NLI	22,498	2,500	3,000	3
STS	10,501	1,167	519	2
YNAT	41,110	4,568	9,107	7
Japanese Text Classification				
TR	129,747	16,218	16,219	3
WRIME	30,000	2,500	2,500	5
Machine Translation				
DeEn	160,239	7,283	6,750	-
ViEn	130,933	768.	1,268	-
ZhEn	209,941	887.	1,261	-

Table 7: Statistics of datasets.

Parameter	BiLSTM	BERT
Embedding Size	64	768
BiLSTM/BERT Hiden Size	256	768
# of BiLSTM/BERT Layers	1	12
Dropout Rate	0.5	0.1
Optimizer	Adam	AdamW
Learning Rate	0.001	0.00002

Table 8: Overview of hyperparameters for backbone models of text classification tasks.

We selected the hyperparameters for the subword regularization methods (the smoothing parameter for SentencePiece (Unigram) and the dropout probabilities for BPE-Dropout and MaxMatch-Dropout) according to the performance on the development splits in the experiments. Tables 10 and 11 summarize the selected values of the hyperparameters for the text classification and machine translation, respectively. Note that the other methods without subword regularization (Unigram, BPE, and WordPiece) do not require these hyperparameters.

Parameter	Transformer
Enc/Dec Embedding Size	512
Enc/Dec FFN Embedding Size	1,024
# of Enc/Dec Attention Heads	4
# of Enc/Dec Layers	6
Clipping Norm	0.0
Dropout Rate	0.3
Weight Decay	0.0001
Max Tokens for Mini-Batch	1,000
Optimizer	Adam
β_1 and β_2 for Adam	0.9, 0.98
Learning Rate	0.0005
Learning Rate Scheduler	Inverse Square Root
Warming-Up Updates	4,000

Table 9: Overview of hyperparameters for backbone model of machine translation tasks.

	English							Korean			Japanese	
	APG	APR	TS	QNLI	QQP	RTE	SST-2	NLI	STS	YNAT	TR	WRIME
BiLSTM												
Unigram+Sub. Reg.	0.2	0.2	0.2	0.6	0.9	0.3	0.2	0.9	0.3	0.3	0.4	1.0
BPE-dropout	0.2	0.2	0.4	0.1	0.1	0.1	0.3	0.3	0.2	0.3	0.5	0.2
MaxMatch-dropout	0.2	0.3	0.6	0.1	0.1	0.3	0.4	0.4	0.2	0.3	0.4	0.6
BERT												
MaxMatch-Dropout	0.6	0.4	0.2	0.1	0.1	0.1	0.3	0.5	0.4	0.5	0.4	0.5

Table 10: Selected hyperparameters for subword regularization methods in text classification tasks.

	IWSLT14		IWSLT15			
	DeEn	EnDe	ViEn	EnVi	ZhEn	EnZh
Unigram + Sub. Reg.	0.3	0.3	0.4	0.3	0.2	0.2
BPE-Dropout	0.1	0.2	0.2	0.2	0.3	0.2
MaxMatch-Dropout	0.3	0.3	0.4	0.1	0.1	0.2

Table 11: Selected hyperparameters for subword regularization methods in machine translation tasks. The selected hyperparameters were used for the subword regularization of both the source and target languages.

Hyperparameter	Trial	Unigram+Sub. Reg.	BPE-Dropout	MaxMatch-Dropout
No regularization	-	characteristics	characteristics	characteristics
0.1	1	**character_i_s_t_ic_s**	characteristics	**characteristic_s**
	2	**character_i_s_t_ics**	characteristics	characteristics
	3	**characteristic_s**	characteristics	characteristics
	4	**cha_rac_t_e_r_istic_s**	characteristics	characteristics
	5	**ch_ar_act_e_r_istic_s**	characteristics	characteristics
0.5	1	characteristics	characteristics	**characteristic_s**
	2	characteristics	**c_har_ac_ter_istics**	characteristics
	3	characteristics	characteristics	**char_acter_istics**
	4	characteristics	**char_ac_ter_istics**	characteristics
	5	**characteristic_s**	**character_ist_ics**	characteristics
0.9	1	characteristics	**c_h_a_r_a_c_t_er_i_s_t_i_c_s**	**char_a_c_t_e_ri_s_t_i_c_s**
	2	characteristics	**char_ac_t_er_ist_ics**	**c_har_a_c_t_e_r_istics**
	3	characteristics	**c_h_ar_a_c_t_er_i_s_t_ic_s**	**ch_a_r_acter_i_s_t_i_c_s**
	4	characteristics	**c_h_a_r_ac_t_e_r_i_s_ti_c_s**	**character_i_s_t_i_cs**
	5	characteristics	**c_ha_ra_ct_er_i_st_i_c_s**	**character_i_stic_s**

Table 12: Examples of tokenized words using three methods with different hyperparameters for five trials. "_" indicates token boundaries. The vocabularies for each method were constructed using the APG dataset. Sampled tokenizations that differed from the original tokenizations without subword regularization are indicated in bold. We removed special symbols indicating the beginning or middle of words such as "##" for simple explanation.

Adaptive Meta-learner via Gradient Similarity for Few-shot Text Classification

Tianyi Lei[1], **Honghui Hu**[1], **Qiaoyang Luo**[2], **Dezhong Peng**[1], **Xu Wang**[1*]
[1]College of Computer Science, Sichuan University
[2]The University of Adelaide
{leity828,wangxu.scu}@gmail.com

Abstract

Few-shot text classification aims to classify the text under the few-shot scenario. Most of the previous methods adopt optimization-based meta learning to obtain task distribution. However, due to the neglect of matching between the few amount of samples and complicated models, as well as the distinction between useful and useless task features, these methods suffer from the overfitting issue. To address this issue, we propose a novel Adaptive Meta-learner via Gradient Similarity (AMGS) method to improve the model generalization ability to a new task. Specifically, the proposed AMGS alleviates the overfitting based on two aspects: (i) acquiring the potential semantic representation of samples and improving model generalization through the self-supervised auxiliary task in the inner loop, (ii) leveraging the adaptive meta-learner via gradient similarity to add constraints on the gradient obtained by base-learner in the outer loop. Moreover, we make a systematic analysis of the influence of regularization on the entire framework. Experimental results on several benchmarks demonstrate that the proposed AMGS consistently improves few-shot text classification performance compared with the state-of-the-art optimization-based meta-learning approaches. The code is available at: https://github.com/Tianyi-Lei.

1 Introduction

As a fundamental task of few-shot learning (Fei-Fei et al., 2006) in natural language processing theme, few-shot text classification (Yu et al., 2018; Geng et al., 2019) requires a model to predict categories that are not seen in training. Meta learning (Schmidhuber, 1987; Thrun and Pratt, 2012), which plays a crucial role in general few-shot learning, aims to improve generalization ability and fast adaptation ability of the learner through modelling the distribution of tasks. To adapt few-shot tasks, typical supervised meta-learning methods (Vinyals et al., 2016; Finn et al., 2017) model task distributions from a few support tasks over meta-training episodes. Subsequently, numerous methods based on meta-learning (Bao et al., 2020; Luo et al., 2021; Han et al., 2021) are proposed to solve few-shot text classification problem.

Within the meta-learning frameworks, Bao et al. (2020) trains an attention-based model to enhance the text representation of distributional signature, Luo et al. (2021) leverages label-semantic augmentation to help BERT compensate for the ambiguity of the class definition caused by the limited data, and Han et al. (2021) strengthens the generalization of a model using an adversarial domain adaptation network. However, these methods are similar to the traditional meta-learning methods, neglecting the overfitting problem caused by utilizing the few number of data in the complicated models under the meta-learning frameworks.

To address the above problem in few-shot text classification, several methods are proposed based on a principle *i.e.*, obtaining more task-distribution can ameliorate the risk of over-fitting to the training task distribution. Bansal et al. (2020) alleviates overfitting through joint training of self-supervised tasks and classification tasks in pre-trained models. We also follow this method and use a self-supervised Mask Token Prediction (MTP) task in meta training phase. Unfortunately, the increased task distribution generated by this joint training is not always positive for meta-training.

In order to further overcome the overfitting challenge in meta-training, we propose the adaptive meta-learner via gradient similarity based on another principle *i.e.*, distinguishing positive and negative features by feature selection of deep model can enhance generalization by alleviating overfitting. In optimization-based meta-learning framework, the gradient contains all the information transmitted from the inner-learner to the outer-

*Corresponding author

Proceedings of the 29th International Conference on Computational Linguistics, pages 4873–4882
October 12–17, 2022.

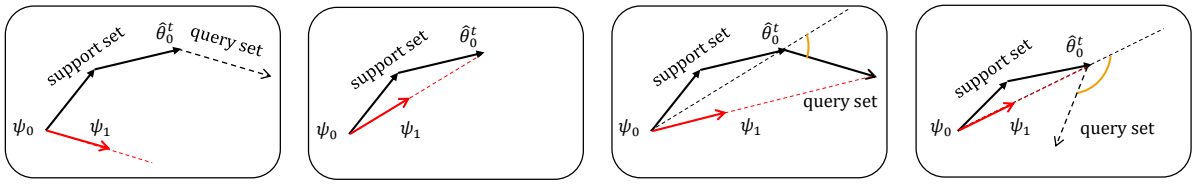

(a) MAML(First order) (b) REPTILE (c) AMGS (ours, scheme 1) (d) AMGS (ours, scheme 2)

Figure 1: Diagram of the comparison of different methods for gradient direction optimization. The black arrow, black dotted arrow, red arrow and red dashed line denote the actual update of base-learner, the update direction of based-learner, the actual update of meta-learner and the update direction of meta-learner, respectively. **(a)** MAML(First order): A set of initial parameters ψ_0 is updated in the direction of the red arrow, *i.e.*, the gradient of query set loss, which is calculated at $\hat{\theta}$ after t-step updates. Note that, since the gradient calculation of MAML contains the Hessian matrix, it is hard to represent in the figure, we use the First Order MAML (FOMAML) to replace MAML. **(b)** REPTILE: ψ_0 is updated along the red arrow pointing to the t-step optimization solution. **(c)** and **(d)** are different schemes of our proposed adaptive meta-learner, which distinguish the positive gradient cosine similarity (scheme 1) and negative gradient cosine similarity (scheme 2). If the gradient direction obtained on the query set is similar to the gradient direction of the sum of the t updates (black dashed line), ψ_0 is updated in the direction of the sum of all gradients. While if their gradient directions are opposite, we remove the gradient obtained from the query set.

learner, including "features" mentioned in above principle. Thus, the gradient obtained by base-learner can be regarded as the "features". To compare with other training strategies for meta-learner, we plot Figure 1. Other strategies often adopt all the gradient obtained by the base-learner without distinction. They also may consume enormous computing resources for calculating the Hessian matrix, sacrifice the stability and accuracy in order to adopt the first-order algorithm, or discard the query set in the training Batch in order to simplify the calculation. By contrast, our method only needs to distinguish the gradient similarity between the gradient of the loss on the query set and the current gradient of the base-learner during the meta-training process. Subsequently, we utilize the corresponding gradient of its loss to help meta-learner quickly adapt to the optimization space. Such method selects the more useful gradients for meta-learner in current training batch. In addition, it neither increases the computational complexity nor causes waste of text information in the same training episode.

According to the above principles, we propose a novel Adaptive Meta-learner via Gradient Similarity (AMGS) algorithm based on optimization-based meta learning scheme. We firstly construct the self-supervised task called Mask Token Prediction (MTP) for the base-learner in the inner loop. Such approach can generate the extension of the task distribution from unlabeled text and constraint the gradient updating of primary classification task to increase the robustness of the model. Moreover, in the outer loop, we utilize the adaptive meta-learner to improve the utilization of the task features from the inner loop. As Figure (1) shows, our strategy can more efficiently leverage query set samples in a training episode, which optimizes the scope of gradient optimization. Therefore, the adaptive meta-learner directly accomplish additional amelioration of overfitting.

The contributions of this paper are summarized as follows: (1) We construct an optimization-based meta-learning framework named AMGS and elaborately design a meta-training algorithm to effectively tackle the overfitting issue in few-shot text classification based on two different principles. (2) We propose an adaptive meta-learner that selects the positive gradients and removes the negative gradients to improve the generalization ability of the model on the few-shot task (3) Experimental results demonstrate that the proposed AMGS outperforms the state-of-the-art optimization-based meta-learning models.

2 Related Work

Few-shot text classification via meta learning Few-shot learning is an application of meta-learning. In most meta-learning frameworks, the strategies can be divided into two categories: metric-based meta-learning and optimization-based meta-learning. Prototypical Network (Snell et al., 2017), Induction Network (Geng et al., 2019) and Relation Network (Sung et al., 2018) are dedicated

to construct a metric space between classes and samples. In the optimization-based meta-learning methods, most of them consist of an inner (or base) algorithm and an outer (or meta) algorithm. Model-Agnostic Meta-Learning (MAML) (Finn et al., 2017) and Reptile (Nichol et al., 2018) are examples of such optimization-based algorithms. LEOPARD (Bansal et al., 2020) achieves a good performance on diverse classification tasks by using BERT (Devlin et al., 2019). Meanwhile, recent work (Bao et al., 2020) proposes a meta-learning-based method by using distributional signatures for few-shot text classification. More recently, LaSAML (Luo et al., 2021) uses label information for few-shot text classification. Another one (Han et al., 2021) applies a domain discriminator into a meta-learning framework. However, these algorithms suffer from overfitting caused by the imbalance between the few data and the deep model in the few-shot setting. By contrast, our proposed AMGS which expands the task distribution in the inner loop and distinguishes the positive and negative gradient in the outer loop can address this issue indirectly and directly.

Auxiliary learning In general, auxiliary learning can assist the main task to learn more accurately and quickly in deep learning (Wang et al., 2022, 2019b), especially in the multi-task learning field. SSL-Reg (Zhou et al., 2021) builds a regularizer of the loss of self-supervised learning tasks to improve performance on text classification. Besides constructing a task, external auxiliary data can also be introduced into the model to obtain more latent information (Zhang et al., 2018).

Similarly, auxiliary tasks are valuable to adapt the meta-learning scheme. MAXL (Liu et al., 2019) adopts a self-supervised learning scheme to generate auxiliary labels, improving the generalization ability of the primary task in gradient update. Furthermore, self-supervised auxiliary tasks can promote fast adaptation during the testing phase (Chi et al., 2021). Hybrid SMLMT (Bansal et al., 2020) creates a specific self-supervised auxiliary task for multi-task learning. Similar to these auxiliary tasks, our auxiliary task MTP is self-supervised to generate richer task distribution during meta-training.

3 Methods

In this section, we first introduce the preliminaries for few-shot classification (Vinyals et al., 2016).

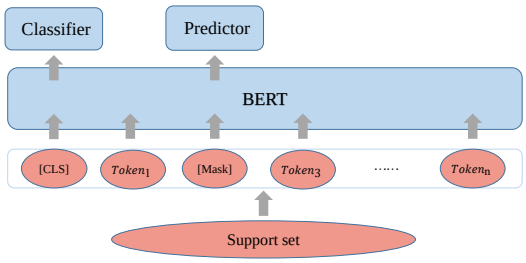

Figure 2: The main framework of the proposed AMGS.

Next, we describe Adaptive Meta-learner via Gradient Similarity (AMGS) method in detail.

3.1 Overview

Problem setup The setting of few-shot classification often includes *training episode* and *testing episode*. Suppose we have examples with labels from the classes $y_{train}$of training episode and need to predict the labels of examples from unseen but related classes y_{test} of testing episode. The training classes and testing classes are mutually exclusive, denotes as $y_{train} \cap y_{test} = \Phi$. To create a training episode, we need to build a set of *N-way K-shot* tasks. For each task, we sample N classes, $k+q$ examples of each class randomly. The $N \times k$ $\{x_s, y_s\}$ pairs including examples and corresponding labels constitute the support set, while the $N \times q$ labeled examples $\{x_q, y_q\}$ are known as the query set. It is the same way to create a support set in testing episode, but leverage the unlabeled examples $\{y_q\}$ to create the query set in testing. By repeating the above procedure, we can obtain enough training and testing episodes, so that we can use them in meta-training and meta-testing respectively. In short, such setting requires the model to have the ability to generalize from seen classes in training episodes to unseen classes in testing episodes.

Model architecture BERT (Devlin et al., 2019) performs well in the conventional text classification, thus we leverage it as text encoder in our proposed AMGS framework to explore the problem of few-shot text classification. As shown in Figure (2), the model architecture consists of the BERT encoder, a classifier and a predictor. The model performs the primary task (i.e, classification) and the auxiliary task (i.e., token prediction) simultaneously, which constitutes multi-task learning. In training period, the support set are used to obtain the BERT encoding and label prediction

in the primary branch. While in the auxiliary task branch, BERT encoder updates parameters through the self-supervised task without labels.

For convenience to explain in following section, we define parameters of the total network $\theta = \{\theta_b, \theta_c^{pri}, \theta_p^{aux}\}$, where θ_b denotes the shared weights of BERT encoding, θ_c^{pri} represents classification weights for the primary task, and θ_p^{aux} is prediction weights for the auxiliary task. Concretely, the primary-branch weights and the auxiliary-branch weights are respectively denoted as $\theta^{pri} = \{\theta_b, \theta_c^{pri}\}$, and $\theta^{aux} = \{\theta_b, \theta_p^{aux}\}$.

Self-supervised Mask Token Prediction task As mentioned above, we leverage BERT as text encoder. Considering that our self-supervised auxiliary task should be adapted to BERT, we adopt the Mask Token Prediction **(MTP)** task used in BERT pre-training stage (also known as MLM). MTP randomly masks the tokens in the sentences according to the specified ratio. These masked tokens are fed into BERT to be predicted by putting the final hidden vector corresponding to the masked token into the output softmax over the vocabulary. The original strategy of MTP in BERT set 15% probability for each token to replace with the [MASK] token 80% of the time, a random token 10% of the time, and the unchanged token 10% of the time. In some cases, if the text used to construct MTP task is very short, none of the tokens in this text would be masked with high probability. This will cause the effect of MTP to fail in the downstream task.

Therefore, we improve the replacement probability of each token being masked to 30% instead of 15%. Meanwhile, the masking rating of replacing the target token with a random token and an unchanged token are both set to 0%, because random and unchanged replacement both occur for 3% of all tokens, which leads to instability. This change helps the model acquire the new task distribution more stably. The masked strategy is explored in experiment demonstrated in Appendix A.

3.2 Adaptive Meta-learner via Gradient Similarity (AMGS)

The optimization-based meta-learning methods (Finn et al., 2017; Nichol et al., 2018) learn an appropriate initial parameters by meta learner, achieving encouraging performance. However, these methods ignore the overfitting issue in the few-shot learning. Considering that the direction of gradients could be used to distinguish the positive and negative gradients, we propose AMGS framework with explicit regularization. The training procedure of AMGS is decomposed into two steps: (i) The base-learner collects gradient for adaptive meta-learner, which utilizes the multi-task network to learn primary and auxiliary tasks together on support set. Then it collects the gradient of the loss on the query set by leveraging the supervised primary task. (ii) The adaptive meta-learner via gradient similarity distinguishes the positive and negative gradient obtained by the first stage, then updates the parameters of the total meta-network by meta-learner. By completing two training steps, our method ensures that the meta-learner learns the more balanced initial parameters and makes the loss of new tasks decrease faster.

3.2.1 Collecting gradient for adaptive meta-learner

This subsection describes that how the base-learner collects gradient for adaptive meta-learner. We leverage the self-supervised MTP task to acquire a more abundant task distribution and improve the base-learner robustness. In addition, as mentioned above, we build multi-task learning by using the MTP to limit the training of classification tasks. In other words, the constraint on the loss of the primary task has been enforced via the auxiliary task. This limitation prevents the base-learner to obtain extra characteristics of each training task to alleviate overfitting. Formally, we compute the total loss of the multi-task network as follows:

$$\mathcal{L}_{total} = (1-\rho)\mathcal{L}_{pri} + \rho\mathcal{L}_{aux}, \tag{1}$$

where $\mathcal{L}_{total}$, $\mathcal{L}_{pri}$, $\mathcal{L}_{aux}$ and ρ represent the total loss, primary classification loss $\mathcal{L}_{pri}(x, y; \theta^{pri})$, auxiliary prediction task loss $\mathcal{L}_{aux}(x; \theta^{aux})$, and the contribution of the auxiliary task, respectively. x and y denote training texts and their labels. We use cross entropy loss to implement both text classification and the masked token prediction. In our experiments, we set $\rho = 10^{-3}$. The sensitivity study is shown in Appendix B.

When training on the tasks T_i in the *support set*, the total loss Eq.(1) after one or a few gradient updates can be defined as follows:

$$\hat{\theta} = \theta - \alpha\nabla_\theta\mathcal{L}_{T_i}^{total}(x_s, y_s; \theta), \tag{2}$$

where x_s and y_s are texts and corresponding labels in the support set. α is the adaptation learning rate. By Eq.(1) and Eq.(2), we can obtain more semantic

Algorithm 1 Training procedure of AMGS

Input: learning rate α, β, texts and corresponding labels x, y

Initialize $\Psi = \theta = \{\theta_b, \theta_c^{pri}, \theta_p^{aux}\}$ with BERT

while not converged **do**
  Sample batch of tasks $T_i \sim p(T)$
  Sample support set (x_s, y_s), query set (x_q, y_q)
  for all T_i **do**
    Compute adapted parameters with gradient descents: $\hat{\theta} = \theta - \alpha \nabla_\theta \mathcal{L}_{T_i}^{total}(x_s, y_s; \theta)$
    Compute the gradients of primary task on $\hat{\theta}$: $(\theta_b, \theta_c^{pri}) = (\theta_b, \theta_c^{pri}) - \alpha \nabla_\theta \mathcal{L}_{T_i}^{pri}(x_q, y_q; \hat{\theta}_b, \hat{\theta}_c^{pri})$
  end for
  if $cos(\nabla_\theta \mathcal{L}_{T_i}^{total}(x_s, y_s; \theta), \nabla_\theta \mathcal{L}_{T_i}^{pri}(x_q, y_q; \hat{\theta}_b, \hat{\theta}_c^{pri})) \geq 0$ **then**
    Update: $\hat{\Psi} \leftarrow \Psi - \beta \nabla_\Psi \sum_{T_i \sim p(T)} (\mathcal{L}_{T_i}^{total}(x_s, y_s; \theta) + \mathcal{L}_{T_i}^{pri}(x_q, y_q; \hat{\theta}^{pri}))$
  else
    Update: $\hat{\Psi} \leftarrow \Psi - \beta \nabla_\Psi \sum_{T_i \sim p(T)} \mathcal{L}_{T_i}^{total}(x_s, y_s; \theta)$
  end if
end while

representation to apply explicit regularization to the primary loss. In general, the query set is used for testing and inference, while it contains rich task distribution which can be applied to meta-learn. We argue that the query set can be used to fine-tune and enhance the gradient learned by the base-learner through the multi-task network. In the step, we accomplish the collection of gradient of the parameters $\{\theta_b, \theta_c^{pri}\}$ on the query set. Finally, the objective can be defined as follows:

$$\arg\min_{\theta_b, \theta_c^{pri}} \mathcal{L}_{T_i}^{pri}(x_q, y_q; \hat{\theta}^{pri}), \tag{3}$$

where x_q and y_q are texts and corresponding labels in the query set.

3.2.2 Upgrade meta-learner with AMGS

This stage is mainly about updating meta-learner. Following previous work (Du et al., 2018), we leverage the gradient cosine similarity to measure whether the gradients obtained on query set are positive or negative. Based on Eq.(2) and Eq.(3), we get the gradient cosine similarity by calculating $cos(\nabla_\theta \mathcal{L}_{T_i}^{total}(x_s, y_s; \theta), \nabla_\theta \mathcal{L}_{T_i}^{pri}(x_q, y_q; \hat{\theta}_b, \hat{\theta}_c^{pri}))$. If the value of $cos(\cdot)$ is non-negative, such gradient is regarded as the **positive gradient**, which means the query set at this batch is beneficial to enhance generalization of the model. Therefore we obtain the gradient of its loss to perform gradient enhancement on the meta-learner. For this situation, the meta-objective can be written as:

$$\begin{aligned} \arg\min_{\theta_b, \theta_c^{pri}, \theta_p^{aux}} \sum_{T_i \sim p(T)} & (\mathcal{L}_{T_i}^{total}(x_s, y_s; \theta) \\ & + \mathcal{L}_{T_i}^{pri}(x_q, y_q; \hat{\theta}^{pri})). \end{aligned} \tag{4}$$

On the contrary, if $cos(\cdot)$ is negative, such gradient is considered as the **negative gradient**. We remove this query set loss to ensure that the model is not negatively affected, so the meta-objective is:

$$\arg\min_{\theta_b, \theta_c^{pri}, \theta_p^{aux}} \sum_{T_i \sim p(T)} (\mathcal{L}_{T_i}^{total}(x_s, y_s; \theta)). \tag{5}$$

According to above training procedure, our proposed meta-objective can distinguish the positive to use and the negative to by adaptive meta-learner, which can automatically filter appropriate regularization to limit the gradient optimization. This step reduces effective model capacity, hence it effectively alleviates overfitting and improves the generalization ability of the model. The full training procedure is demonstrated in the Algorithm 1.

3.2.3 Meta testing

The model parameters have been learned in meta training phase, and fine-tuned in the meta-learning testing phase for downstream tasks. MTP can continue to participate in the fine-tuning phase in order to help the primary classification adapt to the unseen classes for the new tasks quickly. From the perspective of test-time fast adaptation (Chi et al., 2021), our auxiliary task boosts the fast gradient descent of the loss function of the primary task in the testing procedure.

Methods		HuffPost		Banking77				Clinc150 (cross domain)			
		5-way		10-way		15-way		10-way		15-way	
		1-shot	5-shot	1-shot	5-shot	1-shot	5-shot	1-shot	5-shot	1-shot	5-shot
Metric	BERT+PROTO	40.59	53.48	63.05	78.60	59.18	74.12	57.43	72.90	52.31	66.06
	BERT+RELATION	40.80	51.87	63.88	73.48	56.29	64.57	54.65	60.09	46.54	58.83
	BERT+INDUCT	39.96	50.79	48.72	64.32	49.45	55.27	46.52	57.65	41.72	49.98
Optimization	BERT+MAML	41.03	57.13	59.21	85.55	55.69	81.48	60.14	80.24	55.00	65.20
	BERT+REPTILE	40.80	58.96	58.36	82.81	56.69	81.14	59.89	81.23	53.32	63.04
	BERT+R2D2	40.78	61.98	70.45	87.80	63.46	**85.65**	62.72	87.13	57.61	80.76
	DS+R2D2	41.34	62.48	59.33	83.71	53.37	78.96	55.56	78.76	53.41	79.69
	MLADA+R2D2	41.55	59.82	61.69	80.81	55.63	74.77	65.28	85.45	51.76	77.77
	BERT+AMGS (OURS)	**43.47**	**63.40**	**71.41**	**88.81**	**63.62**	84.93	**69.19**	**88.26**	**62.12**	**84.13**

Table 1: Results of 5-way 1-shot and 5-way 5-shot on HuffPost headlines dataset, 10-way 1-shot, 10-way 5-shot, 15-way 1-shot and 15-way 5-shot on Banking77 and Clinc150 datasets (cross domain) by using our proposed method and all baselines.

4 Experiments

4.1 Datasets

We use three datasets to evaluate the performance in experiment.

HuffPost headlines includes 36900 news headlines among 41 classes, which contains less information than other datasets. In order to complete a fair comparison test, we divide each training, validation, and testing set into 20, 5, and 16 classes by following the setting of Bao et al. (2020). **Banking77** (Casanueva et al., 2020) consists of 13083 fine-grained intents and 77 classes. As for the setting of data distribution and *N-way K-shot* classification tasks, we assign 30, 15, and 32 classes fixedly for training, validation, and testing set. **Clinc150** (Wang et al., 2019a) is a cross-domain intent classification dataset with 150 classes in 10 domains. It provides 22500 examples that cover 150 intents from 10 domains without overlap among classes. We allocate for each training, validation, and testing with 4, 1, 5 domains, respectively.

4.2 Baselines

In order to evaluate our AMGS, we compare with three metric-based methods and five optimization-based algorithms for few-shot text classification.

Proto (Snell et al., 2017) provides a metric-based method to learn the class vector by computing distances to prototype representations of each class. **Induct** (Geng et al., 2019) learns a generalized class-wise representation by leveraging the dynamic routing algorithm. **Relation** (Sung et al., 2018) compares the class vector and the query feature through a relation-based meta-learner. **MAML** (Finn et al., 2017) is one of the most typical optimization-based meta-learning algorithms, which trains a favorable initial point for the base learner by utilizing the meta learning that learns among tasks. **Reptile** (Nichol et al., 2018) is a first-order variant method of MAML. It achieves that the speed of calculation is greatly improved and the complexity is reduced, while the accuracy is almost the same as MAML. The base learner used by Ridge Regression Differentiable Discriminator **(R2D2)** (Bertinetto et al., 2019) is ridge regression based on linear regression model. The amount of calculation is related to the sample size of the task, which is conducive to the learning of the meta learner. **DS** (Bao et al., 2020) shows the best performance by leveraging the model that builds an attention generator and a ridge regressor to enhance the representational power of distributional signature. **MLADA** (Han et al., 2021) uses the meta-learning adversarial domain adaptation network to improve the adaptation and new classes embedding generation by creating a domain discriminator.

4.3 Implementation details

$BERT_{base}$ is used as the text encoder of all baselines. Because DS and MLADA have special requirements for textual representation and feature extraction, forcibly using BERT as encoder will be counterproductive. Thus, we re-implement the pre-train fastText embeddings (Joulin et al., 2016) for those model, and follow other settings in the

original papers (Bao et al., 2020; Han et al., 2021). For the sake of fairness, the classifiers of these two algorithms use R2D2, so we constructed a comparison item with BERT as encoder.

All parameters are optimized with Adam optimizer (Kingma and Ba, 2015). The initial learning rates α, β are separately set to $5e{-}5$ and $2e{-}5$, and we utilize 5 gradient updates for the base adaptation step. As for the *N-way K-shot* classification setting, all experiments use 25 examples for the query set. We randomly sample 100 training episodes, 100 validation episodes, and 1000 testing episodes per epoch and apply early stopping on validation for 20 epochs. We evaluate the performance of the model based on 5 different random seeds. All experiments are conducted on a GEFORCE RTX 3090 GPU.

4.4 Experimental results

The total results of experiments are reported in table 1. By observing these experimental results, we obtain the following conclusions:

(1) Whether it is for texts with minimal semantics (Huffpost), fine-grained categorized (Banking77) or cross domains (Clinc150), our proposed method AMGS has an average improvement of 0.2-6.5% over the state-of-the-art model on both 1-shot and 5-shot classification. In particular, compared with our AMGS and MAML (Finn et al., 2017), Reptile (Nichol et al., 2018), we can draw the following observations from the Table 1: (i) Our proposed method achieves better performance on all tasks. In especial, in the 15-Way 5-shot task on Clinc150 dataset, our proposed method outperforms the best counterpart by 18.9%. (ii) MAML and Reptile perform better on fine-grained classification Banking77 dataset with more similar categories than on cross-domain Clinc150 dataset with less similar categories, and have a smaller gap with our AMGS. To verify that our AMGS perform better than MAML on alleviating overfitting, we plot their accuracy learning curves in Figure 3. In the figure, the training procudure of our AMGS is more stable than that of MAML from the beginning to the end. Besides, the gap between the accuracy of seen classes and unseen classes of our AMGS is less than that of MAML. These results are demonstrated that our AMGS can make model more stable in meta-training and more readily generalizeds to unseen classes by addressing the overfitting issue.

(2) With leveraging BERT as our text encoder, our method is better than all compared methods on Huffpost dataset. In Bao et al. (2020), it points out that BERT can better deal with highly contextual classification but not the keyword-based news classification, e.g., Huffpost dataset. Thus, "DS+R2D2" performs better on Huffpost than "BERT+R2D2", but worse on Banking77 and Clinc150. Nonetheless, our "BERT+AMGS" surpasses all BERT-based and non-BERT-based approaches on Huffpost dataset, which shows the superiority of our AMGS method. Furthermore, the performance of our model is increased by 2.1% on 1-shot classification and 0.9% on 5-shot classification when compared with BERT-based models.

Overall, the above observations point that AMGS can learn the commonalities and characteristics between few-shot task distribution well by mitigating overfitting, thereby obtaining a better initialized parameter for fast adaptation.

Methods	Banking77		Clinc150	
	15-way		15-way	
	1-shot	5-shot	1-shot	5-shot
AMGS w que	56.39	82.32	55.72	65.61
AMGS w sup	57.03	81.84	54.17	64.12
AMGS w que+sup	58.51	82.10	55.97	83.73
AMGS w our strategy	**63.62**	**84.93**	**62.12**	**84.13**

Table 2: Ablation study results for different strategies of meta-learner on Banking77 and Clinc150 (cross domain) datasets.

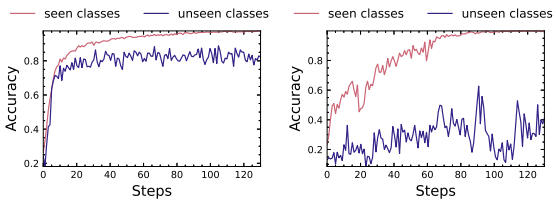

(a) AMGS (OURS) (b) MAML

Figure 3: Learning curves of AMGS (a) and MAML (b) on 15-way 5-shot task of the Banking77 dataset. We plot average accuracy from seen classes (red) and unseen classes (blue).

4.5 Ablation studies

In this section, we conduct several ablation experiments to verify the effectiveness of the adaptive meta-learner, MTP in the meta training phase, and MTP in the meta-testing fast adaptation phase.

The effectiveness of the adaptive meta-learner In this section, we further investigate the impact of

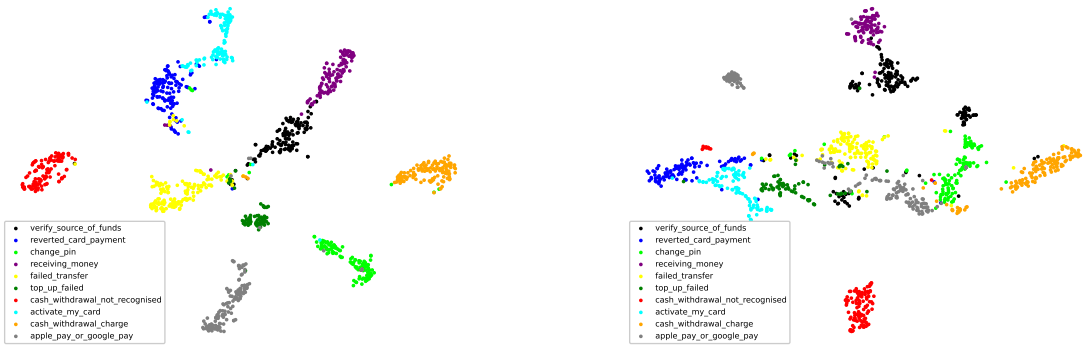

(a) AMGS (OURS) (b) REPTILE

Figure 4: t-SNE visualization of the input representation for the query set of a testing episode (N=10, K=5, L=120) sampled from Banking77 dataset.

Methods	Banking77		Clinc150	
	15-way		15-way	
	1-shot	5-shot	1-shot	5-shot
AMGS w/o MTP	62.82	84.73	61.82	83.17
AMGS w/o MTP (testing)	63.26	84.32	61.97	84.01
AMGS w MTP	**63.62**	**84.93**	**62.12**	**84.13**

Table 3: Ablation study results of MTP in meta-training and meta-testing fast adaptation phase (testing) on Banking77 and Clinc150 (cross domain) datasets.

the different strategies for meta-learner. To compare with our strategy, we design three other comparison strategies. As shown in Table 2, "AMGS w que", "AMGS w sup", "AMGS w que+sup" respectively represent the meta-learner in AMGS only use the gradients of the query set, support set and both query and support set. None of these three strategies pay attention to distinguishing the positive or the negative of the gradients. Comparing our strategy with "AMGS w que+sup" strategy, we have improved significantly more on 1-shot task than on 5-shot task. From all the results, our adaptive meta-learner which filters the impact of the negative gradient achieves the better performance among these compared strategies.

The effectiveness of MTP in meta-training phase and in meta-testing fast adaptation phase As shown in Table 3, we first eliminate MTP in training stage. After losing a richer distribution of tasks, the performances of AMGS decrease by about 0.8%, which verifies the effectiveness of MTP in meta-training phase. Further, we explore MTP in meta-testing fast adaptation phase. The empirical results demonstrate that after joining the auxiliary task in meta-testing, the model performances have increased by about 0.5%. The testing auxiliary task makes the primary task more robust on the support set, and has some suppression effects on the occurrence of overfitting. All these results demonstrate that MTP task have a certain effect on Banking77 and Clinc150 datasets, but it can not significantly improve the experimental results.

4.6 Visualization

We visualize the results of the experiments to demonstrate that our model can generate a high-quality text representation for unseen classes.

T-SNE (Van der Maaten and Hinton, 2008) visualization illustrates the experimental results in Figure (4), we take out the generated sentence embedding layer before sending it to the classifier for visualization. Comparing Figure 4(a) and Figure 4(b), it is obvious that our method AMGS produces better separation than REPTILE, Especially for the categories represented by gray and lime, the sentence representations obtained by REPTILE are very similar, so that it is difficult to distinguish their categories. The above observations demonstrate the effectiveness of AMGS to generate a high-quality text representation for few-shot text classification.

5 Conclusion

In this paper, we present an Adaptive Meta-learner via Gradient Similarity (AMGS) framework for few-shot text classification. To be specific, we first leverage the self-supervised Mask Token Prediction (MTP) task to enrich the task distribution with the unlabeled text. Such approach can reduce the impact of overfitting caused by the mismatching between the few samples and the deep model.

Secondly, we construct an adaptive meta-learner via gradient similarity for the outer loop to distinguish the positive and negative gradient. Thus, the meta-learner alleviates overfitting by preventing the influence of negative features. Experimental results validate that our model achieves significant improvement on the few-shot text classification tasks by effectively alleviating the overfitting issue.

6 Acknowledge

This work is supported by the National Natural Science Foundation of China (Grant No. U19A2078), China Postdoctoral Science Foundation (Nos. 2021TQ0223, 2022M712236), Sichuan Science and Technology Planning Project (Grant Nos. 2022YFQ0014, 2022YFH0021, 2021YFS0390), and Fundamental Research Funds for the Central Universities (No. 2022SCU12081).

References

Trapit Bansal, Rishikesh Jha, Tsendsuren Munkhdalai, and Andrew McCallum. 2020. Self-supervised meta-learning for few-shot natural language classification tasks. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 522–534.

Yujia Bao, Menghua Wu, Shiyu Chang, and Regina Barzilay. 2020. Few-shot text classification with distributional signatures. In *8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, April 26-30, 2020*. OpenReview.net.

Luca Bertinetto, João F. Henriques, Philip H. S. Torr, and Andrea Vedaldi. 2019. Meta-learning with differentiable closed-form solvers. In *7th International Conference on Learning Representations, ICLR 2019, New Orleans, LA, USA, May 6-9, 2019*. OpenReview.net.

Iñigo Casanueva, Tadas Temčinas, Daniela Gerz, Matthew Henderson, and Ivan Vulić. 2020. Efficient intent detection with dual sentence encoders. In *Proceedings of the 2nd Workshop on Natural Language Processing for Conversational AI*, pages 38–45.

Zhixiang Chi, Yang Wang, Yuanhao Yu, and Jin Tang. 2021. Test-time fast adaptation for dynamic scene deblurring via meta-auxiliary learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 9137–9146.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. Bert: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186.

Yunshu Du, Wojciech M Czarnecki, Siddhant M Jayakumar, Mehrdad Farajtabar, Razvan Pascanu, and Balaji Lakshminarayanan. 2018. Adapting auxiliary losses using gradient similarity. *arXiv preprint arXiv:1812.02224*.

Li Fei-Fei, Rob Fergus, and Pietro Perona. 2006. One-shot learning of object categories. *IEEE transactions on pattern analysis and machine intelligence*, 28(4):594–611.

Chelsea Finn, Pieter Abbeel, and Sergey Levine. 2017. Model-agnostic meta-learning for fast adaptation of deep networks. In *International Conference on Machine Learning*, pages 1126–1135. PMLR.

Ruiying Geng, Binhua Li, Yongbin Li, Xiaodan Zhu, Ping Jian, and Jian Sun. 2019. Induction networks for few-shot text classification. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 3904–3913.

Chengcheng Han, Zeqiu Fan, Dongxiang Zhang, Minghui Qiu, Ming Gao, and Aoying Zhou. 2021. Meta-learning adversarial domain adaptation network for few-shot text classification. In *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*, pages 1664–1673.

Armand Joulin, Edouard Grave, Piotr Bojanowski, Matthijs Douze, Hérve Jégou, and Tomas Mikolov. 2016. Fasttext. zip: Compressing text classification models. *arXiv preprint arXiv:1612.03651*.

Diederik P. Kingma and Jimmy Ba. 2015. Adam: A method for stochastic optimization. In *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*.

Shikun Liu, Andrew J Davison, and Edward Johns. 2019. Self-supervised generalisation with meta auxiliary learning. In *Proceedings of the 33rd International Conference on Neural Information Processing Systems*, pages 1679–1689.

Qiaoyang Luo, Lingqiao Liu, Yuhao Lin, and Wei Zhang. 2021. Don't miss the labels: Label-semantic augmented meta-learner for few-shot text classification. In *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*, pages 2773–2782.

Alex Nichol, Joshua Achiam, and John Schulman. 2018. On first-order meta-learning algorithms. *CoRR*, abs/1803.02999.

Jürgen Schmidhuber. 1987. *Evolutionary principles in self-referential learning, or on learning how to learn: the meta-meta-... hook*. Ph.D. thesis, Technische Universität München.

Jake Snell, Kevin Swersky, and Richard Zemel. 2017. Prototypical networks for few-shot learning. In *Proceedings of the 31st International Conference on Neural Information Processing Systems*, pages 4080–4090.

Flood Sung, Yongxin Yang, Li Zhang, Tao Xiang, Philip HS Torr, and Timothy M Hospedales. 2018. Learning to compare: Relation network for few-shot learning. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1199–1208.

Sebastian Thrun and Lorien Pratt. 2012. *Learning to learn*. Springer Science & Business Media.

Laurens Van der Maaten and Geoffrey Hinton. 2008. Visualizing data using t-sne. *Journal of machine learning research*, 9(11).

Oriol Vinyals, Charles Blundell, Timothy Lillicrap, Daan Wierstra, et al. 2016. Matching networks for one shot learning. *Advances in neural information processing systems*, 29:3630–3638.

Alex Wang, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel R Bowman. 2019a. Glue: A multi-task benchmark and analysis platform for natural language understanding. In *7th International Conference on Learning Representations, ICLR 2019*.

Xu Wang, Peng Hu, Pei Liu, and Dezhong Peng. 2022. Deep semisupervised class- and correlation-collapsed cross-view learning. *IEEE Transactions on Cybernetics*, 52(3):1588–1601.

Xu Wang, Dezhong Peng, Peng Hu, and Yongsheng Sang. 2019b. Adversarial correlated autoencoder for unsupervised multi-view representation learning. *Knowledge-Based Systems*, 168:109–120.

Mo Yu, Xiaoxiao Guo, Jinfeng Yi, Shiyu Chang, Saloni Potdar, Yu Cheng, Gerald Tesauro, Haoyu Wang, and Bowen Zhou. 2018. Diverse few-shot text classification with multiple metrics. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 1206–1215.

Yabin Zhang, Hui Tang, and Kui Jia. 2018. Fine-grained visual categorization using meta-learning optimization with sample selection of auxiliary data. *CoRR*, abs/1807.10916.

Meng Zhou, Zechen Li, and Pengtao Xie. 2021. Self-supervised Regularization for Text Classification. *Transactions of the Association for Computational Linguistics*, 9:641–656.

A Ablation for different MTP masking strategies

In section 3.1, we mention that MTP adopts a different masking strategy from the one used in BERT pre-train stage. We explore the effect of different masking strategies in following ablation.

Masking prob.	Masking strategy			Banking77
	Mask	Same	Random	10-way 1-shot
15%	100%	0%	0%	73.38
	80%	10%	10%	72.06
30%	**100%**	**0%**	**0%**	**74.06**
	80%	10%	10%	72.20
45%	100%	0%	0%	72.56
	80%	10%	10%	72.42

Table 4: Ablation study results of different masking strategies on the validation episodes of Banking77.

As Table 3 shows, we explore the effect of different masking probabilities and strategies on Banking77. In the table, "Mask" means that we replace the token with [MASK] in MTP, "Same" means that we keep the target token unchanged and "Random" means that the token is replaced with the random token except itself. From the table, we can see that the masking strategy in BERT pre-training is not the best choice in the few-shot text classification. Therefore, in this paper, we attempt to alter the masking strategy which 100% changes the target token to [MASK].

B Sensitivity study on the trade-off parameter ρ

In order to set an appropriate value for the trade-off parameter of MTP mentioned in section 3.2.1, we study a sensitivity study for this hyper-parameter in 10-way 5-shot on Banking77 dataset.

trade-off ρ	0.9	0.5	10^{-1}	10^{-3}	10^{-5}	0
Accuracy	89.98	90.10	94.80	**95.60**	94.00	93.40

Table 5: Sensitivity study results of 10-way 5-shot on the validation episodes of Banking77.

The results of validation episodes have shown in Table 4. We explore a large scale trade-off ρ. demonstrating MTP has the greatest contribution when the trade-off ρ equals 0.001. Especially, ρ equals 0 means we remove the impact of MTP, which verifies the effectiveness of our MTP.

Vocabulary-informed Language Encoding

Xi Ai
College of Computer Science
University of Chonqing
barid.x.ai@gmail.com

Bin Fang
College of Computer Science
University of Chonqing
fb@cqu.edu.cn

Abstract

A Multilingual model relies on language encodings to identify input languages because it has to distinguish between the input and output languages or among all the languages for cross-lingual tasks. Furthermore, we find that language encodings potentially refine multiple morphologies of different languages to form a better isomorphic space for multilinguality. To leverage this observation, we present a method to compute a vocabulary-informed language encoding as the language representation, for a required language, considering a local vocabulary covering an acceptable amount of the most frequent word embeddings in this language. In our experiments, our method can consistently improve the performance of multilingual models on unsupervised neural machine translation and cross-lingual embedding.

1 Introduction

With tied weights across required languages, a multilingual model is trained on non-parallel or/and parallel multilingual corpora. Essentially, language encodings are required for cross-lingual tasks because the multilingual model has to distinguish between the input and output languages or among all the languages generally. We observe that, besides identifications of languages, language encodings can help the model build isomorphic space for multilinguality with the help of shared tokens. Specifically, our hypothesis derives from the decomposition of attention mechanisms (Vania and Lopez, 2017; Luong et al., 2015; Libovický and Helcl, 2017), and we observe explicit alignments and implicit alignments, where explicit alignments are key for language identifications, and implicit alignments promote multilinguality. Furthermore, the implicit alignments are a special case of unsupervised bilingual/multilingual lexical induction, helping multilingual models learn multilingual and cross-lingual knowledge. Our goal is to retain the explicit alignments for language identifications and improve the implicit alignments for multilinguality.

In this work, we render an analysis of the attention mechanism in optimization, and then we find implicit alignments among a language encoding and other language encodings. Stemming from morphology adaptation and the observation, we present a method to compute VLE (**V**ocabulary-informed **L**anguage **E**ncoding) as the required language encodings, for multilingual models. Each required language maintains a local vocabulary covering a subset of the most frequent tokens in this language from the shared vocabulary of the multilingual model. Given one language and its vocabulary, when adapting to the multilingual model, we apply transformation layers to the average of token embeddings in the vocabulary for VLE that can be used in either *padding style* or *adding style*. VLE provides language characteristics for language identification and leverages implicit alignments for multilinguility. Eventually, the multilingual model can identify languages and learn better multilinguality with the help of VLE.

2 Related Work and Background

2.1 Language Encoding

Let E_x denotes x's embedding. Given an input sentence $X = \{x_0, ..., x_n\}$ and the language encoding E_{LT_l} both in language $Lang_l$, we have: *padding style* (Johnson et al., 2017): $X_{input} = \{E_{LT_l}, E_{x_0}, ..., E_{x_n}\}$ and *adding style* (Lample and Conneau, 2019): $X_{input} = \{E_{x_0} + E_{LT_l}, ..., E_{x_n} + E_{LT_l}\}$. For notational convenience, we omit some other techniques, e.g., position encodings. Suppose we apply E_{LT_l} in *adding style* for Transform-based models[1]. For predicting x_i in the input sentence, the attention score $e_{i,j} = (E_{x_i} + E_{LT_l})^T W_q^T W_k (E_{x_j} + E_{LT_l})$ of the first self-attention layer between query vector q

[1]For the *padding style* and other models, it is similar.

Proceedings of the 29th International Conference on Computational Linguistics, pages 4883–4891
October 12–17, 2022.

and key vector k within the same sentence can be decomposed as:

$$e_{i,j} = E_{x_i}^T W_q^T W_k E_{x_j} + E_{x_i}^T W_q^T W_k E_{LT_l} + E_{LT_l}^T W_q^T W_k E_{x_j} + E_{LT_l}^T W_q^T W_k E_{LT_l}, \quad (1)$$

where W_q and W_k are transformation layers for q and k respectively.

2.1.1 Explicit and Implicit Alignment

Specifically, multilingual models (Johnson et al., 2017; Devlin et al., 2019; Lample and Conneau, 2019) usually form a vocabulary that covers shared tokens across 1+ languages. Suppose x_i is shared by $Lang_l$ and $Lang_{l'}$. In optimization, we have two backward passes from predicting x_i: *1)* $\frac{\partial \varepsilon_{x_i}}{\partial E_{LT_l}}$ and *2)* $\frac{\partial \varepsilon_{x_i}}{\partial E_{LT_{l'}}}$, that E_{LT_l} and $E_{LT_{l'}}$ are aligned to x_i explicitly. Then, we have the pivoted alignment $E_{LT_l} \leftrightarrow E_{x_i} \leftrightarrow E_{LT_{l'}}$. Since E_{x_i} is a point in the embedding space, $E_{LT_l} \leftrightarrow E_{x_i} \leftrightarrow E_{LT_{l'}}$ implies the implicit alignment $E_{LT_l} \leftrightarrow E_{LT_{l'}}$.

Meanwhile, we observe that unsupervised methods for word translation or lexical induction (Lample et al., 2018a; Artetxe et al., 2018) leverage similar implicit alignments to refine languages' morphologies. Concretely, given two subsets of N and M in $Lang_1$ and $Lang_2$ respectively from a shared vocabulary, (Lample et al., 2018a) explore an unsupervised domain-adversarial training (Ganin et al., 2016) for morphology adaptation that N and M are not parallel but cover the most frequent words in the two languages respectively. Embeddings in each sub-vocabulary are invariant to a language or a domain and serve as multiple anchors to identify the language and constrain the morphology. Then, the model considers the implicit alignment:

$$\frac{1}{N}\sum_{n\in N} E_{x_n} \leftrightarrow \frac{1}{M}\sum_{m\in M} E_{x_m}. \quad (2)$$

In multilingual models, $E_{LT_l} \leftrightarrow E_{LT_{l'}}$ could be viewed as a special case of Eq.2, where $|N| = |M| = 1$. Empirically, large $|N|$ and $|M|$ can help the model build isomorphic spaces (Lample et al., 2018a; Artetxe et al., 2017). Our method derives its motivation from this that vocabularies can be used to generate language encodings, i.e., $|N|, |M| > 1$, for improving multilinguality, as the morphology adaptation Eq.2 can be consistently improved by using large $|N|$ and $|M|$.

3 Our Approach

3.1 VLE

Following the previous idea and our observation, we present a method to generate language encodings from local vocabularies. Concretely, given a fixed size of vocabulary Voc_l formed by the monolingual tokens from the monolingual corpora in $Lang_l$, VLE (vocabulary-informed language encoding) for $Lang_l$ is defined as:

$$E_{Voc_l} = \frac{1}{|Voc_l|}\sum_{i\in Voc_l}, \quad E_{V_l} = \sigma(W^l E_{Voc_l}) \odot E_{Voc_l}, \quad (3)$$

where Voc_l is a local vocabulary for $Lang_l$ and $W_l \in \mathbb{R}^{d\times d}$. We introduce E_{V_l} to the multilingual model for the identification of $Lang_l$ in either *padding style* or *adding style*. Then, any two E_{V_l} and $E_{V_{l'}}$ can have the implicit alignment: $E_{V_l} \leftrightarrow E_{V_{l'}}$. Since both $|Voc_l| > 1$ and $|Voc_{l'}| > 1$, $E_{V_l} \leftrightarrow E_{V_{l'}}$ leverages the morphology adaptation Eq.2 ($|N|, |M| > 1$) for refining the morphologies of the languages to consistently improve isomorphic spaces. In our experiment, we justify this hypothesis on a cross-lingual embedding task and provide a t-SNE visualization (Van Der Maaten and Hinton, 2008) to show the improvement of aligning token pairs in two languages.

Meanwhile, E_{V_l} has to be able to represent the language. The backend of identification relies on the language characteristics from embeddings. Intuitively, the employment of $|Voc_l|$ provides some information for approximation, and then E_{V_l} gives the model global information (Shah and Barber, 2018; Ai and Fang, 2021a) covering language characteristics. Following this intuition, any method extracting common language characteristics from embeddings is feasible. In our preliminary experiments, we find that E_{V_l} could be obtained by applying a very shallow network to the average of the token embeddings in its vocabulary. In this work, we instantiate the model with feature contributions. Specifically, σ yields a probability of each embedding feature for language characteristics, i.e., contributions for language characteristics, similar to (Ai and Fang, 2021b) that uses σ to generate probabilities over vector elements. Statistics in §Experiment show some embedding features are significant to language characteristics with very high probabilities (≈ 1).

3.2 Formation of Voc_l

To find a local Voc_l for a required language from the shared vocabulary, we calculate the most frequent tokens in the monolingual corpora of this language and select a subset of Top-K tokens. However, some of the most frequent tokens are multilingual, which are shared by 1+ languages, i.e., numbers. Essentially, our Voc_l is expected to represent the language with less ambiguity. Inspired by (Wang et al., 2020), we score all the tokens with:

$$m(x) = \frac{C_l(x)}{C_{\neq l}(x)}, \quad (4)$$

where $C_l(x)$ and $C_{\neq l}(x)$ are the count of x in the monolingual corpora of $Lang_l$ and other languages' corpora respectively. Intuitively, $m(x)$ measures how monolingual x is, i.e., x_i with a high score is more monolingual than x_j with a low score. After scoring, we select tokens with the highest scores for Voc_l. Note that, if we consider language families, this scoring criterion is essential for our method. Specifically, some languages are closely related with lots of shared tokens in the vocabulary such as Spanish and Portuguese. The scoring method significantly mitigates the pain because Voc_l covers frequent tokens that appear most likely in $Lang_l$. On the other hand, for dissimilar languages with a minimum amount of shared tokens in the vocabulary, e.g., only sharing numbers, Voc_l is formed by frequent tokens that appear only in $Lang_l$.

3.3 Analysis and Discussion

Size of K The size of Voc_l is significant in practice. If K is too large (e.g., 10,000), it may cause memory problems on mediocre machines, then terminating training and inferring. However, too small K (e.g., 10) may not be able to approximate language information. Our empirical study shows that median K (e.g., 100) can facilitate training and substantially improve experimental results.

Impact of Tokenization Method We are interested in how the tokenization method impact the performance because it potentially affects the formation of the shared vocabulary and then Voc_l. Different tokenization methods may result in different vocabulary and Voc_l, e.g., BPE (Sennrich et al., 2016b) and word-level. However, we are aware that the impact is relevantly small given that: *1)* tokens of non-standard words could be monolingual and can be used for Voc_l; *2)* tokens in Voc_l for VLE do not necessarily have meaningful semantics because they work like anchors of languages's subspaces in the embedding space.

Efficiency Efficiency can be evaluated from two aspects: *1)* training and *2)* inferring. Since Voc_l is fixed for $Lang_l$, the only degradation of training efficiency comes from the dynamic computation of the average operation, the lookup operation, and the transformation, which are all fast. In inferring, E_{V_l} is a constant vector for a required language, which do not hurt inferring efficiency.

4 Experiment

Our code is implemented on Tensorflow 2.2 (Abadi et al., 2016) with 2 NVIDIA Titan Xp 12G GPUs. We accumulate gradients of 2 mini-batches per pre-training step. Since we have only 2 GPUs, this operation emulates 4 GPUs. All the links of datasets, libraries, scripts, and tools marked with $\diamond$ are listed in §Appendix. A preview version of the code is submitted, and we open the source code on GitHub.

4.1 Multilingual Task

See §Appendix for more details.

Unsupervised Neural Machine Translation UNMT (Lample and Conneau, 2019; Lample et al., 2018b; Song et al., 2019; Liu et al., 2020) tackles bilingual translation (Bahdanau et al., 2015; Vaswani et al., 2017) on non-parallel bilingual corpora without having access to any parallel sentence.

Cross-lingual Embedding Recall that we derive VLE from the study of domain adaptation in unsupervised word translation or lexical induction (Eq.2). To further investigate whether VLE improves the agnostic process of forming the isomorphic space, we test the MUSE$\diamond$ task (Lample et al., 2018a) with the provided test set and tools, which is used to evaluate cross-lingual embedding similarities. This test can quantitatively report how VLE refines and improves the morphologies to overlap each other for forming the isomorphic space.

4.2 Multilingual Framework

We adapt our method to XLM (Lample and Conneau, 2019) and MASS (Song et al., 2019) that can be used to pre-train a multilingual model with the objective of MLM (masked language modeling) (Devlin et al., 2019) or train the multilingual model for multilingual tasks from scratch or pre-training. All these frameworks need language encodings to recognize and flag the required languages.

4.3 Adaptation with VLE

Besides using the same frameworks, datasets and configurations, to minimize the changes for comparison, we only replace language encodings with our VLE and use the same styles (*padding* or *adding*) as the baseline models use. For all the tasks and frameworks, we apply our scoring method (Eq. 4) and select tokens in the model's vocabulary with the highest $K = 100$ scores to form Voc_l for every language, i.e., $|Voc_l| = 100$. See §Appendix for discussion about the size of $|Voc_l|$.

4.4 UNMT

See §Appendix for more details.

Dataset For evaluation, we train UNMT on the same dataset used in previous works. We use monolingual corpora $\{Fr, De, En\}$ from WMT 2018$\diamond$ including all available $NewsCrawl$ datasets from 2007 through 2017 and monolingual corpora Ro from WMT 2016$\diamond$ including $NewsCrawl$ 2016. We test $Fr \leftrightarrow En$ on *newstest2014* and $\{De, Ro\} \leftrightarrow En$ on *newstest2016*. For any language pairs $En \leftrightarrow X$, we concatenate their monolingual corpora and then shuffle the concatenated corpus. Note that, since Ro is low-resource, we oversample Ro in pre-training and training.

Model Configuration and Preprocessing The model configuration, preprocessing, and the BLEU script are identical to previous works: XLM and MASS. Concretely, we use a 6-layer encoder and 6-layer decoder Transformer, and the dimensions of word embeddings, hidden states, and filter sizes are 1024, 1024, and 4096 respectively. All the weights and lookup tables are shared by all the required languages. We run fastBPE$\diamond$ to learn shared 60K BPE from multilingual corpora required by the multilingual model. The sampling strategy is the same as the balanced strategy presented by (Lample and Conneau, 2019). We report *case-sensitive* BLEU computed by the *multi-bleu* script $\diamond$.

4.5 Pre-training & Training

In pre-training, we use Adam optimizer (Kingma and Ba, 2015) with $\beta_1 = 0.9$,$\beta_2 = 0.999$, $\epsilon = 1e-8$, and $lr = 1e-4$. The model is pre-trained around 400K iterations. Although pre-training is important for high-performance UNMT, we also test random UNMT without pre-training to observe the lower bound and how our VLE works alone, using the MASS framework in training. In the training phase, we use Adam optimizer (Kingma and Ba, 2015) with parameters $\beta_1 = 0.9$,$\beta_2 = 0.997$ and $\epsilon = 10^{-9}$, and a dynamic learning rate with $warm\_up = 8000$ (Vaswani et al., 2017) ($learning\_rate \in (0, 7e^{-4}]$) is employed. After around 400K iterations, we report results.

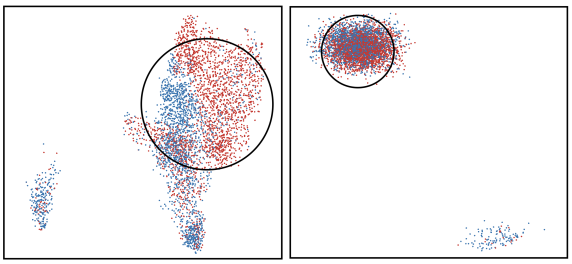

(a) XLM (b) XLM+VLE

Figure 1: t-SNE visualization for MUSE pairs. Each point is a different token instance. This figure suggests that XLM aligns pairs somewhat out-of-the-box. Pairs are more aligned in-the-box when introducing VLE. Perplexity:17. Iteration:10k.

Performance Table 1 shows that our method can consistently improve the performance of baseline models on UNMT tasks, which confirms the effectiveness of our method. We observe that our method works better for low-resource Ro than for rich-resource $\{De, Fr\}$ (4% vs. 3%). As presented in *Random*, the effectiveness of our method is not from pre-training because random UNMT trained from scratch is significantly improved by using our method. Intuitively, our VLE carries multiple embeddings for morphology adaptation (recall Eq.2) that help the model understand multilinguality and cross-linguality from the isomorphic space in pre-training and training as previous works (Lample et al., 2018b; Ai and Fang, 2021b) report the effectiveness of aligning selected embeddings in UNMT. Eventually, it helps the model to learn translation knowledge. Essentially, morphology adaptation is very useful for low-resource languages.

4.6 Cross-lingual Embedding

We evaluate cross-lingual word similarities on $En \leftrightarrow De$. For our test, we use XLM and MASS that are restored by their last checkpoint of pre-training on monolingual corpora in $\{German, English\}$ from the experiments of UNMT respectively. After restoration, we extract token embeddings required by the test set via lookup tables. For words split into 2+ sub-tokens, we average all the sub-tokens. We evalu-

Model	$De \leftrightarrow En$		$Fr \leftrightarrow En$		$Ro \leftrightarrow En$	
Random *adding style*	20.99	17.12				
+ Ours	23.36	19.71				
Random *padding style*	20.86	17.08				
+ Ours	23.15	19.48				
XLM *adding style* (Lample and Conneau, 2019)	33.81	26.32	32.87	32.94	31.12	32.81
+ Ours	34.88	27.20	34.01	34.13	32.59	34.24
MASS *adding style* (Song et al., 2019)	34.91	28.03	34.42	37.02	32.75	34.82
+ Ours	35.82	28.51	35.12	37.81	34.16	36.11

Table 1: Performance of UNMT. *All the baseline models are reimplemented with our configurations. Random* denotes the model without any pre-training.

Model	MUSE (cos)
XLM(Lample and Conneau, 2019)	0.53
+VLE	0.56
MASS(Song et al., 2019)	0.55
+VLE	0.57

Table 2: Performance on MUSE task. *All the baseline models are reimplemented with our configurations.*

ate the performance by cosine similarity, reporting the result in Table 2. As expected, applying VLE can consistently improve the performance on this task, which confirms the improvements of the isomorphic space. Significantly, it confirms our hypotheses and assumptions that VLE can refine the morphologies of the languages to form a better isomorphic space. Meanwhile, we provide a t-SNE visualization (Van Der Maaten and Hinton, 2008) of the embedding space for MUSE pairs in Figure 1. This figure suggests that embedding pairs are more aligned in-the-box by using VLE.

4.7 Language characteristic

Previous works like (Ai and Fang, 2022; Conneau et al., 2020) study how different specifics of information are processed in the model, e.g., tokens and languages. As aforementioned, we expect to approximate language characteristics from $|Voc_l|$ as language encodings. Recall that, in Eq.3, we use $sigmoid$ and simply transformation layers to compute the contribution of each embedding feature for language characteristics. We present the statistics of contributions in Figure 2 from the pre-trained model on $\{En, De\}$. 10% of embedding features significantly contribute to language characteristics, obtaining over 0.8. By contrast, over 55% of embedding features are not selected for language characteristics and close to 0. These statistics can support our idea that using Voc_l and embeddings is able to provide language characteristics for language identifications. Note that, statistics may differ among different model parameters. However, the conclusions are similar in our experiments with different model parameters.

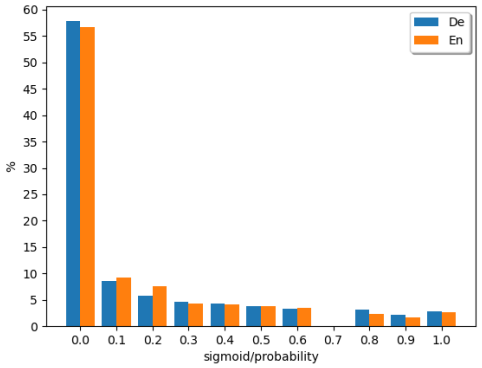

Figure 2: Contribution of each embedding feature for language characteristics (Eq.3).

4.8 Supportive Experiment

We analyze VLE on the size of $|Voc_l|$, the impact of tokenization methods, and efficiency, reporting experimental results in §Appendix.

5 Conclusion

In this work, we present a method to generate VLE, a vocabulary-informed language encoding for the identification of a required language in multilingual models. We consider a frequency-based and local vocabulary for every language. For a required language, the required language encoding is obtained by applying transformation layers to the average of the token embeddings in its vocabulary. In our experiments, VLE shows effectiveness on UNMT and cross-lingual embedding tasks and is possible to improve language adaptation and multilinguality because VLE can refine the morphologies of the languages to improve the isomorphic space. Our method is simple but effective and compatible with any other extension for multilingual models.

References

Martin Abadi, Paul Barham, Jianmin Chen, Zhifeng Chen, Andy Davis, Jeffrey Dean, Matthieu Devin, Sanjay Ghemawat, Geoffrey Irving, Michael Isard, Manjunath Kudlur, Josh Levenberg, Rajat Monga, Sherry Moore, Derek G. Murray, Benoit Steiner, Paul Tucker, Vijay Vasudevan, Pete Warden, Martin Wicke, Yuan Yu, and Xiaoqiang Zheng. 2016. Tensorflow: A system for large-scale machine learning. In *12th USENIX Symposium on Operating Systems Design and Implementation (OSDI 16)*, pages 265–283.

Xi Ai and Bin Fang. 2021a. Almost free semantic draft for neural machine translation. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 3931–3941.

Xi Ai and Bin Fang. 2021b. Empirical regularization for synthetic sentence pairs in unsupervised neural machine translation. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 35, pages 12471–12479.

Xi Ai and Bin Fang. 2022. Leveraging Relaxed Equilibrium by Lazy Transition for Sequence Modeling. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics*, volume 1, pages 2904–2924. Long Papers.

Mikel Artetxe, Gorka Labaka, and Eneko Agirre. 2017. Learning bilingual word embeddings with (almost) no bilingual data. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics*, pages 451–462, Vancouver, Canada. Association for Computational Linguistics.

Mikel Artetxe, Gorka Labaka, and Eneko Agirre. 2018. A robust self-learning method for fully unsupervised cross-lingual mappings of word embeddings. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics*, pages 789–798, Melbourne, Australia. Association for Computational Linguistics.

Dzmitry Bahdanau, Kyung Hyun Cho, and Yoshua Bengio. 2015. Neural machine translation by jointly learning to align and translate. In *3rd International Conference on Learning Representations, ICLR 2015 - Conference Track Proceedings*.

Ond rej Bojar, Rajen Chatterjee, Christian Federmann, Yvette Graham, Barry Haddow, Matthias Huck, Antonio Jimeno Yepes, Philipp Koehn, Varvara Logacheva, Christof Monz, Matteo Negri, Aurelie Neveol, Mariana Neves, Martin Popel, Matt Post, Raphael Rubino, Carolina Scarton, Lucia Specia, Marco Turchi, Karin Verspoor, and Marcos Zampieri. 2016. Findings of the 2016 conference on machine translation. In *Proceedings of the First Conference on Machine Translation*, pages 131–198, Berlin, Germany. Association for Computational Linguistics.

Ond rej Bojar, Christian Federmann, Mark Fishel, Yvette Graham, Barry Haddow, Matthias Huck, Philipp Koehn, and Christof Monz. 2018. Findings of the 2018 conference on machine translation (wmt18). In *Proceedings of the Third Conference on Machine Translation*, pages 272–307, Belgium, Brussels. Association for Computational Linguistics.

Alexis Conneau, Shijie Wu, Haoran Li, Luke Zettlemoyer, and Veselin Stoyanov. 2020. Emerging Cross-lingual Structure in Pretrained Language Models. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 6022–6034.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 4171–4186, Minneapolis, Minnesota. Association for Computational Linguistics.

Yaroslav Ganin, Hugo Larochelle, and Mario Marchand. 2016. Domain-Adversarial Training of Neural Networks. *Journal of Machine Learning Research*, 17:1–35.

Melvin Johnson, Mike Schuster, Quoc V. Le, Maxim Krikun, Yonghui Wu, Zhifeng Chen, Nikhil Thorat, Fernanda Viégas, Martin Wattenberg, Greg Corrado, Macduff Hughes, and Jeffrey Dean. 2017. Google's multilingual neural machine translation system: Enabling zero-shot translation. *Transactions of the Association for Computational Linguistics*, 5:339–351.

Mandar Joshi, Danqi Chen, Yinhan Liu, Daniel S Weld, Luke Zettlemoyer, and Omer Levy. 2020. Spanbert: Improving pre-training by representing and predicting spans. *Transactions of the Association for Computational Linguistics*, 8:64–77.

Diederik P Kingma and Jimmy Lei Ba. 2015. Adam: A method for stochastic optimization. In *3rd International Conference on Learning Representations, ICLR 2015 - Conference Track Proceedings*.

Philipp Koehn, Hieu Hoang, Alexandra Birch, Chris Callison-Burch, Marcello Federico, Nicola Bertoldi, Brooke Cowan, Wade Shen, Christine Moran, Richard Zens, Chris Dyer, Ondřej Bojar, Alexandra Constantin, and Evan Herbst. 2007. Moses: Open source toolkit for statistical machine translation. In *Proceedings of the 45th Annual Meeting of the Association for Computational Linguistics Companion Volume Proceedings of the Demo and Poster Sessions*, pages 177–180, Prague, Czech Republic. Association for Computational Linguistics.

Guillaume Lample and Alexis Conneau. 2019. Cross-lingual language model pretraining. In *Advances in neural information processing systems*.

Guillaume Lample, Alexis Conneau, Marc'Aurelio Ranzato, Ludovic Denoyer, and Hervé Jégou. 2018a. Word translation without parallel data. In *6th International Conference on Learning Representations, ICLR 2018 - Conference Track Proceedings*.

Guillaume Lample, Myle Ott, Alexis Conneau, Ludovic Denoyer, and Marc'Aurelio Ranzato. 2018b. Phrase-based & neural unsupervised machine translation. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 5039–5049, Brussels, Belgium. Association for Computational Linguistics.

Jindřich Libovický and Jindřich Helcl. 2017. Attention strategies for multi-source sequence-to-sequence learning. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics*, pages 196–202, Vancouver, Canada. Association for Computational Linguistics.

Yinhan Liu, Jiatao Gu, Naman Goyal, Xian Li, Sergey Edunov, Marjan Ghazvininejad, Mike Lewis, and Luke Zettlemoyer. 2020. Multilingual denoising pre-training for neural machine translation. *Transactions of the Association for Computational Linguistics*, 8:726–742.

Thang Luong, Hieu Pham, and Christopher D. Manning. 2015. Effective approaches to attention-based neural machine translation. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 1412–1421, Lisbon, Portugal. Association for Computational Linguistics.

Mauro Mezzini. 2018. Empirical study on label smoothing in neural networks. In *WSCG 2018 - Short papers proceedings*.

Rico Sennrich, Barry Haddow, and Alexandra Birch. 2016a. Improving neural machine translation models with monolingual data. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics*, pages 86–96, Berlin, Germany. Association for Computational Linguistics.

Rico Sennrich, Barry Haddow, and Alexandra Birch. 2016b. Neural machine translation of rare words with subword units. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics*, pages 1715–1725, Berlin, Germany. Association for Computational Linguistics.

Harshil Shah and David Barber. 2018. Generative neural machine translation. In *Advances in Neural Information Processing Systems*, volume 2018-Decem, pages 1346–1355.

Kaitao Song, Xu Tan, Tao Qin, Jianfeng Lu, and Tie-Yan Liu. 2019. MASS: Masked sequence to sequence pre-training for language generation. In *Proceedings of the 36th International Conference on Machine Learning*, volume 97 of *Proceedings of Machine Learning Research*, pages 5926–5936. PMLR.

Laurens Van Der Maaten and Geoffrey Hinton. 2008. Visualizing Data using t-SNE. *Journal of Machine Learning Research*, 9:2579–2605.

Clara Vania and Adam Lopez. 2017. From characters to words to in between: Do we capture morphology? In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics*, volume 1, pages 2016–2027.

Ashish Vaswani, Google Brain, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention Is All You Need. In *Advances in neural information processing systems*, pages 5998–6008.

Zirui Wang, Jiateng Xie, Ruochen Xu, Yiming Yang, Graham Neubig, and Jaime Carbonell. 2020. Cross-lingual Alignment vs Joint Training: A Comparative Study and A Simple Unified Framework. In *8th International Conference on Learning Representations, ICLR 2020 - Conference Track Proceedings*.

A Experiment Setting

A.1 Multilingual Framework

We adapt our method to two MLM instances: XLM (Lample and Conneau, 2019) and MASS (Song et al., 2019), which can be used to pre-train the UNMT model. We follow the instructions of BERT (Devlin et al., 2019) and these two MLM instances to setup frameworks.

XLM XLM is similar to BERT (Devlin et al., 2019) but uses text streams of an arbitrary number of sentences. Following the instruction, we randomly select 15% of the tokens from the input sentence for replacing.

MASS MASS is different from XLM and BERT but similar to SpanBERT (Joshi et al., 2020), using spans to replace consecutive tokens. Given an input sentence with length N, we randomly select consecutive tokens with length $N/2$ for replacing.

A.2 UNMT Pipeline

We use Adam optimizer (Kingma and Ba, 2015) with $\beta_1 = 0.9$,$\beta_2 = 0.999$, $\epsilon = 1e-8$, and $lr = 1e-4$. We set the dropout regularization with a drop rate $rate = 0.1$ and label smoothing with $gamma = 0.1$ (Mezzini, 2018). For data feeding efficiency, similar-length sentences are padded to the same length, so that each mini-batch may have a different number of sentences but the same number of tokens. We pre-train the model around 400K iterations. Although pre-training is important for high-performance UNMT, we also test random UNMT without pre-training to observe

Model	$De \leftrightarrow En$	
100, default	34.88	27.20
20	34.21	26.54
50	34.72	26.96
200	35.93	27.36
500	35.08	27.51
1000	35.15	27.64

Table 3: Impact of Voc_l size on UNMT.

Model	$De \leftrightarrow En$	
baseline (BPE-based)	33.81	26.32
+ Ours	34.88	27.20
baseline (Word-level)	33.01	25.79
+ Ours	34.15	26.61

Table 4: Impact of Tokenization Method.

Model	Speed
XLM	714ms/step
+ Ours , K = 100	772ms/step
+ Ours , K = 10,000	899ms/step

Table 5: Training efficiency.

the lower bound and how our VLE works alone, using the MASS framework in training. In the training phase, we use Adam optimizer (Kingma and Ba, 2015) with parameters $\beta_1 = 0.9$,$\beta_2 = 0.997$ and $\epsilon = 10^{-9}$, and a dynamic learning rate with $warm\_up = 8000$ (Vaswani et al., 2017) ($learning\_rate \in (0, 7e^{-4}]$) is employed. We set dropout regularization with a drop rate $rate = 0.1$ and label smoothing with $gamma = 0.1$. We feed $\approx 2K$ tokens per mini-batch.After around 400K iterations, we report case-sensitive BLEU computed by *multi-BLEU.perl*$\diamond$.

We consider the same dataset used in previous works. Specifically, we first retrieve monolingual corpora $\{Fr, De, En\}$ from WMT 2018$\diamond$ (Bojar et al., 2018) including all available $NewsCrawl$ datasets from 2007 through 2017 and monolingual corpora Ro from WMT 2016$\diamond$ (Bojar et al., 2016) including $NewsCrawl$ 2016. We report the performance for $Fr \leftrightarrow En$ on *newstest2014* and $\{De, Ro\} \leftrightarrow En$ on *newstest2016*. For tokenization, we use the Moses tokenizer$\diamond$ developed by (Koehn et al., 2007). We use fastBPE$\diamond$ to learn shared 60k BPE (Sennrich et al., 2016b) with the same criteria in (Lample and Conneau, 2019).

In the pre-training phase, UNMT is trained on monolingual corpora with the objective of MLM (masked language modeling) for the two languages. Then, in the training phase, on-the-fly back-translation (Sennrich et al., 2016a) performs to generate synthetic parallel sentences that can be used for training of translation as NMT (neural machine translation) is trained on genuine parallel sentences in a supervised manner. Meanwhile, UNMT still learns the MLM objective to maintain language knowledge in the training phase.

B Supportive Result

B.1 Impact of Voc_l Size

We use $K = 100$ as the default $|Voc_l|$ for every language. In Table 3, we study the impact of $|Voc_l|$ and borrow all of the XLM configurations we use in the UNMT task. Ideally, a large $|Voc_l|$ (a significant amount of frequent tokens) can properly represent the language. However, we find a median size (< 200) is enough to achieve a decent result with minimum extra costs. Large size can achieve slightly better performance, but the computational cost is not practical, as discussed in §Size of K. In conclusion, we recommend a median size (< 200) or a finetuned size.

B.2 Impact of Tokenization Method

We are interested in how the tokenization method affects the performance because it potentially affects the formation of Voc_l. For evaluation, we use all the configurations in UNMT and additionally configure a word-level vocabulary for the model. The word-level vocabulary has the same number of tokens as the BPE vocabulary. Table 4 shows that our method can work with different tokenization methods. Our method can generally improve the performance, regardless of the difference between the two baseline models in the same configuration.

B.3 Training Efficiency

In inferring, VLE computes constant vectors for all the required languages, which do not hurt inferring efficiency. Hence, we are interested in training efficiency because we introduce some additional operations to the model. Table 5 indicates that our method does not hurt training efficiency significantly, which is crucial in applications.

C Source

We list all the links of dataset, tools, and other sources in Table 6.

Item	Links
WMT 2016	http://www.statmt.org/wmt16/translation-task.html
WMT 2018	http://www.statmt.org/wmt18/translation-task.html
XLM	https://github.com/facebookresearch/XLM
multi-BLEU.perl	https://github.com/moses-smt/mosesdecoder/blob/master/scripts/generic/multi-BLEU.perl
Moses tokenizer	https://github.com/moses-smt/mosesdecoder/blob/master/scripts/tokenizer/tokenizer.perl
fastBPE	https://github.com/glample/fastBPE
MUSE	https://github.com/facebookresearch/MUSE
Panlex	https://panlex.org/source-list/
Tensor2Tensor	https://github.com/tensorflow
HuggingFace	https://huggingface.co

Table 6: Links of source.

OpticE: A Coherence Theory-Based Model for Link Prediction

Xiangyu Gui, Feng Zhao [*], Langjunqing Jin, Hai Jin
National Engineering Research Center for Big Data Technology and System,
Services Computing Technology and System Lab, Cluster and Grid Computing Lab
School of Computer Science and Technology, Huazhong University of Science and Technology, China
zhaof@hust.edu.cn

Abstract

Knowledge representation learning is a key step required for link prediction tasks with knowledge graphs (KGs). During the learning process, the semantics of each entity are embedded by a vector or a point in a feature space. The distance between these points is a measure of semantic similarity. However, in a KG, while two entities may have similar semantics in some relations, they have different semantics in others. It is ambiguous to assign a fixed distance to depict the variant semantic similarity of entities. To alleviate the semantic ambiguity in KGs, we design a new embedding approach named OpticE, which is derived from the well-known physical phenomenon of optical interference. It is a lightweight and relation-adaptive model based on coherence theory, in which each entity's semantics vary automatically regarding different relations. In addition, a unique negative sampling method is proposed to combine the multimapping properties and self-adversarial learning during the training process. The experimental results obtained on practical KG benchmarks show that the OpticE model, with elegant structures, can compete with existing link prediction methods.

1 Introduction

Knowledge graphs (KGs) consist of sets of triplets that can represent real-world concepts, common sense information or facts. Each triplet (h, r, t) indicates a directional relation r from the head entity h to the tail entity t. WordNet (Miller, 1995), Freebase (Bollacker et al., 2008) and Wikidata (Vrandečić and Krötzsch, 2014) are the most well-known KGs with tremendous numbers of entities and relations. These KGs play important roles in a range of areas, such as natural language processing tasks (Cao et al., 2021), question answering applications (Bosselut et al., 2021) and recommendation systems (Shao et al., 2021). Research on KGs is thriving in both academic and industrial communities.

Link prediction is a fundamental problem when addressing KGs. To learn the hidden patterns from the observed triplets, extensive investigations have been performed to embed entities and relations into a continuous semantic space; this is known as KG embedding learning (Rossi et al., 2021). Among these embedding models, from a mathematical perspective, TransE (Bordes et al., 2013), TransH (Wang et al., 2014) and TransR (Lin et al., 2015) are translation-based models that use translation transformations to characterize the existence of triplets. Other methods (e.g., RESCAL (Nickel et al., 2011) and DistMult (Yang et al., 2015)) tackle this problem with matrix or tensor multiplication. Recently, the embedding problem has been discussed in the complex domain (e.g., ComplEx (Trouillon et al., 2016), RotatE (Sun et al., 2019) and HAKE (Zhang et al., 2020)). These methods have achieved great performance on related tasks. While our models are discussed in the complex domain, they are designed under an interference framework.

The semantic ambiguity caused by multimapping relations was ignored in most previous methods. As illustrated in Figure 1 (a), the semantics of *New York* and *Washington D.C.* are similar under the relation *Located_in*, while they are quite different under the relation *Capital_of* because *New York* is not a capital. Previous methods such as TransE and RotatE suffer from this ambiguity problem. As shown in Figure 1 (b), it is difficult to disambiguate *New York* from *Washington D.C.* for the relation *Capital_of* in RotatE. To alleviate this problem, TransH utilizes a hyperplane to project entities into independent semantic space. RatE (Huang et al., 2020) integrates a semantic matching method with an extra relation-adaptive matrix. Instead of relying on extra settings, we tackle this problem by the intrinsic attribute of the

[*] Corresponding author

Proceedings of the 29th International Conference on Computational Linguistics, pages 4892–4901
October 12–17, 2022.

rotational semantic space.

We are motivated by the optical interference phenomenon (bright or dark fringes appear after the superposition of two beams of light), because (i) it is intuitive to simulate positive cases and negative cases utilizing distinct dark and bright fringes; and (ii) since coherence theory in the complex field is sufficiently sophisticated, it can play a theoretical guiding role. As OpticE is one type of rotation-based model, it can be illustrated with the rotation transformation shown in Figure 1 (c). The entities *New York* and *Washington D.C.* are mapped into different relation-adaptive orbits with the same phase difference. The two entities are close to each other in the orbit of *Located_in*, while in the orbit of relation *Capital_of* they no longer overlap. By embedding entities into orbits with different semantic densities, OpticE can take control of the similarity of entities regarding specific relations. Generally, multimapping relations map entities into small-radius orbits in dense semantic space, while one-to-one relations rely on large-radius orbits in sparse semantic space to distinguish entities. This will be verified in our experiments.

To optimize OpticE effectively, we optimize self-adversarial sampling (Sun et al., 2019) by integrating a Bernoulli filtering process. The proposed method can leverage the high efficiency of self-adversarial sampling and reduce the performance loss caused by false negative labels. Tested on WN18, FB15k, WN18RR and FB15k-237, the widely recognized benchmark datasets for link prediction, the lightweight OpticE model is comparable to the previous rotation-based models. OpticE's ability to adapt to multimapping relations is also verified in our experiments. Our contributions can be summarized as follows:

- **Coherence theory** is introduced into the KG reasoning task for the first time to make the embeddings of KGs in a complex field more intuitive and understandable.
- **A relation-adaptive amplitude modulation technique**, which is simple, effective and lightweight, is developed to alleviate the multimapping problem in KGs.
- **A novel negative sampling method** is provided to train our model precisely and effectively; this approach can be generally applied to other models.

2 Related Work

All the typical related works are listed in Table 1 with their model realizations and complexities in terms of the number of utilized parameters. They are divided into two groups according to whether they are calculated in a real or complex number field.

Real number field models. The most important and influential model is TransE (Bordes et al., 2013), which takes relations as translation transformation by simply assuming that $h + r = t$. However, its expressiveness is limited, especially when representing a multimapping relation. TransH (Wang et al., 2014) exploits an extra projection transformation and manages to eliminate the ambiguity in TransE. DistMult (Yang et al., 2015) uses a semantic matching method while ignoring the directionality of the relations.

Complex field models. ComplEx (Trouillon et al., 2016) utilizes complex numbers in KG embeddings and achieves great performance. It can be considered a complex version of DistMult, but it overcomes the shortcomings of DistMult via tail conjugation. RotatE (Sun et al., 2019) surpasses all the previous models by rotation transformation; similar to TransE, the model assumes that $h \circ r = t$ and suffers from the same ambiguity problem. Based on RotatE, RatE (Huang et al., 2020) defines a relation-specific weight W_r with an extra 8 parameters per relation to solve the ambiguity problem. HAKE (Zhang et al., 2020) employs a hierarchical structure simulated by a modulus, but the model is complicated and difficult to train.

OpticE (ours) belongs to the category of complex field models and provides a novel way to solve the semantic ambiguity problem in RotatE. Like RatE and TransH, it has a relation-adaptive strategy that can retrieve each entity's different semantics. However, our strategy is much simpler due to the use of a relation-modulated amplitude. In the aspect of negative sampling, we adopt multiple negative sampling (Trouillon et al., 2016) and self-adversarial strategy (Sun et al., 2019) in our model, but optimize the process with a Bernoulli distribution (Wang et al., 2014) according to the multimapping relations.

3 The Coherence Theory-Based Model

In this part, we briefly introduce coherence theory in §3.1. In §3.2, we provide three ways to modulate an interference model and demonstrate the ad-

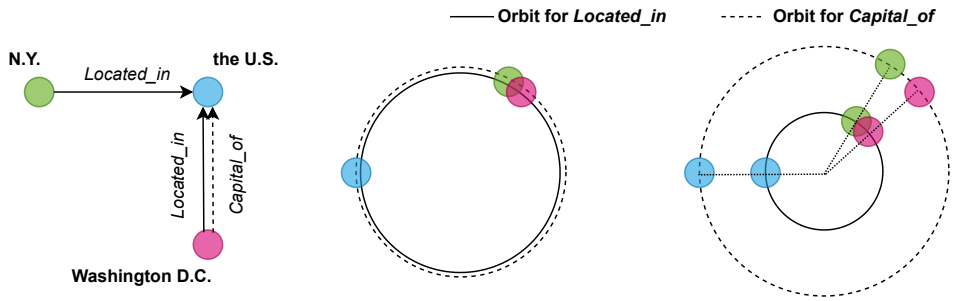

(a) KG (b) RotatE (c) OpticE

Figure 1: Illustration of OpticE. (a) A toy example is given. (b) RotatE is unable to disambiguate *New York* from *Washington D.C.* for the relation *Capital_of*. (c) The entities *New York* and *Washington D.C.* are mapped into different relation-adaptive orbits with the same phase difference.

Table 1: KG embedding models and their storage complexities of parameters, where $\circ$ denotes the Hadamard product, $||\cdot||_{1/2}$ denotes the L1 or L2 norm, and $\odot$ denotes a specific weighted product defined in RatE. k is the dimensionality. N_e and N_r are the numbers of entities and relations.

Model	Score function	Parameters	Complexity
TransE	$-\|\|\mathbf{h}+\mathbf{r}-\mathbf{t}\|\|_{1/2}$	$\mathbf{h},\mathbf{r},\mathbf{t}\in\mathbb{R}^k$	$\mathcal{O}(N_e k+N_r k)$
TransH	$-\|\|(\mathbf{h}-\mathbf{w}_r^\top\mathbf{h}\mathbf{w}_r)+\mathbf{d}_r-(\mathbf{t}-\mathbf{w}_r^\top\mathbf{t}\mathbf{w}_r)\|\|_2^2$	$\mathbf{h},\mathbf{t},\mathbf{d}_r,\mathbf{w}_r\in\mathbb{R}^k$	$\mathcal{O}(N_e k+2N_r k)$
DistMult	$\mathbf{h}^\top diag(\mathbf{r})\mathbf{t}$	$\mathbf{h},\mathbf{r},\mathbf{t}\in\mathbb{R}^k$	$\mathcal{O}(N_e k+N_r k)$
ComplEx	$Re(\mathbf{h}^\top diag(\mathbf{r})\bar{\mathbf{t}})$	$\mathbf{h},\mathbf{r},\mathbf{t}\in\mathbb{C}^k$	$\mathcal{O}(2N_e k+2N_r k)$
RotatE	$-\|\|\mathbf{h}\circ\mathbf{r}-\mathbf{t}\|\|_1$	$\mathbf{h},\mathbf{r},\mathbf{t}\in\mathbb{C}^k, \|r_i\|=1$	$\mathcal{O}(2N_e k+N_r k)$
RatE	$-\|\|\mathbf{h}\odot_{\mathbf{W}(r)}\circ\mathbf{r}-\mathbf{t}\|\|_1$	$\mathbf{h},\mathbf{r},\mathbf{t}\in\mathbb{C}^k, \mathbf{W}\in\mathbb{R}^{2\times 4}$	$\mathcal{O}(2N_e k+N_r(k+8))$
HAKE	$-\|\|\mathbf{h}_m\circ\mathbf{r}_m-\mathbf{t}_m\|\|_2-$ $\lambda\|\|\sin((\mathbf{h}_p+\mathbf{r}_p-\mathbf{t}_p)/2)\|\|_1$	$\mathbf{h}_m,\mathbf{r}_m,\mathbf{t}_m,\mathbf{h}_p,\mathbf{r}_p,\mathbf{t}_p\in\mathbb{R}^k$	$\mathcal{O}(2N_e k+2N_r k)$
OpticE (ours)	$-\|\|\langle\mathbf{M}_\mathbf{r}(\mathbf{h}),\mathbf{M}'_\mathbf{r}(\mathbf{t})\rangle\|\|_1$	$\mathbf{h}$ and $\mathbf{t}$ are light sources, $\mathbf{M}(\cdot)$ is modulator	$\mathcal{O}(N_e k+3N_r k)$

vantage of the proposed OpticE. In §3.3, the model training process is introduced with the proposed Bernoulli self-adversarial sampling method.

3.1 Interference under Coherence Theory

Optical interference is a common phenomenon in physics that can be interpreted formally by the superposition of waves in coherence theory. We briefly introduce this theory. More details can be found in (Hecht, 2016).

Monochromatic harmonic light can be represented as a point in the complex field as

$$E = A\exp[i\varphi], \tag{1}$$

where φ is the phase and A is the amplitude (or modulus). The intensity of the light synthesized by E_1 and E_2 can be defined as

$$\begin{aligned} I &= \langle E_1, E_2\rangle \\ &= A_1^2 + A_2^2 + 2A_1A_2\cos(\varphi_1-\varphi_2). \end{aligned} \tag{2}$$

If and only if $A_1 = A_2$ and phase difference is π, the intensity tends to be 0, which is called **total destructive interference**. In our models, we utilize this state to indicate positive triplets in KGs.

3.2 The Proposed Models

Based on the coherence theory, we take the entities as a series of light sources and the relations as a type of modulation performed on their phases and the amplitudes. Then, the existence of triplets can be indicated by the intensity of the lights after superposition.

To be more intuitive, we modify Equation (2) as

$$I = A_1^2 + A_2^2 - 2A_1A_2\cos(\varphi_1-\varphi_2). \tag{3}$$

Then, the **total destructive interference** condition becomes $A_1 = A_2$ and $\varphi_1 = \varphi_2$. Given a triplet $(h,r,t)\in\mathcal{G}$, each entity consists of k (k is the dimension) light sources. By extending Equation (1), the head entity can be represented as $\mathbf{h} = [E_h^1, E_h^2, \cdots, E_h^k]$, and the tail entity is represented as $\mathbf{t} = [E_t^1, E_t^2, \cdots, E_t^k]$. The score function is

$$f_r(h,t) = -||\langle\mathbf{M}_\mathbf{r}(\mathbf{h}),\mathbf{M}'_\mathbf{r}(\mathbf{t})\rangle||_1,$$

where $\mathbf{M}_\mathbf{r}(\cdot)$ is the modulation operates on the head or the tail, and $\langle\mathbf{M}_\mathbf{r}(\mathbf{h}),\mathbf{M}'_\mathbf{r}(\mathbf{t})\rangle$ is a list of

Algorithm 1 Bernoulli self-adversarial sampling

Require: A KG $\mathcal{G}$, the given true triplet $(h, r, t) \in \mathcal{G}$, the number of negative samples n. For each relation in KG, we first get the statistics about hpt and tph (hpt is the average number of head entities per tail entity; tph is the average number of tail entities per head entity).

Ensure: The negative sample set $\mathcal{S}$ and $p(h'_i, r, t'_i)$.

$p = \frac{hpt}{hpt+tph}$ // probability to corrupt the tail
$\epsilon \leftarrow U(0, 1)$ // ϵ uniformly generated from $(0, 1]$
if $\epsilon < p$ **then**
  while $|\mathcal{S}| < n$ **do**
    $\mathcal{S} \leftarrow \mathcal{S} \cup (h, r, t')$ // to corrupt the tail
  end while
else
  while $|\mathcal{S}| < n$ **do**
    $\mathcal{S} \leftarrow \mathcal{S} \cup (h', r, t)$ // to corrupt the head
  end while
end if
for the i-th sample $(h'_i, r, t'_i) \in \mathcal{S}$, its weight $p(h'_i, r, t'_i)$ can be obtained from Equation (4).

the synthesized intensities of each dimension calculated by Equation (3). The larger the score function value is, the more likely (h, t, r) is the positive case. When the **total destructive interference** condition is met, the score reaches its maximum 0.

In this paper, three ways are designed to modulate the entities. They are summarized as **pOpticE, aOpticE** and **OpticE** models.

(i) **pOpticE with only phase modulation.** The amplitudes of eahc light sources are fixed at 1. A relation only modulates the corresponding phase. Specifically, we have

$$\mathbf{M_r}(\mathbf{h}) = \exp\left[i(\varphi_\mathbf{h} + \varphi^\mathbf{h}_\mathbf{r})\right]$$
$$\mathbf{M'_r}(\mathbf{t}) = \exp\left[i(\varphi_\mathbf{t} + \varphi^\mathbf{t}_\mathbf{r})\right].$$

The modulator provides a phase delay $\varphi^\mathbf{h}_\mathbf{r}$ to the head and $\varphi^\mathbf{t}_\mathbf{r}$ to the tail; then, the score function is

$$f_r(h, t) = ||2 - 2\cos(\varphi_\mathbf{h} + \varphi_\mathbf{r} - \varphi_\mathbf{t})||_1,$$

in which $\varphi_\mathbf{r} = \varphi^\mathbf{h}_\mathbf{r} - \varphi^\mathbf{t}_\mathbf{r}$ is the phase difference added to the head's phase by relation *r*. Then, the phase modulation process can be simplified as

$$\mathbf{M_r}(\mathbf{h}) = \exp\left[i(\varphi_\mathbf{h} + \varphi_\mathbf{r})\right]$$
$$\mathbf{M'_r}(\mathbf{t}) = \exp\left[i(\varphi_\mathbf{t})\right].$$

Similar to other rotation-based models, our models can infer **symmetric, anti-symmetric, inversion** and **composition** relations because all these models rely on the modulated phase difference $(\varphi_\mathbf{h} + \varphi_\mathbf{r} - \varphi_\mathbf{t})$.

(ii) **aOpticE with additional entity-adaptive amplitudes.** The amplitudes of the sources are determined by the specific entities themselves. Then, we obtain

$$\mathbf{M_r}(\mathbf{h}) = \mathbf{A_h} \exp\left[i(\varphi_\mathbf{h} + \varphi_\mathbf{r})\right]$$
$$\mathbf{M'_r}(\mathbf{t}) = \mathbf{A_t} \exp\left[i(\varphi_\mathbf{t})\right].$$

Similar to RotatE, the amplitudes of the entities in aOpticE remain unalterable under different relations.

(iii) **OpticE with relation-adaptive amplitude modulation.** The amplitudes of the sources are determined by the entities and relations, simultaneously. We have

$$\mathbf{M_r}(\mathbf{h}) = \mathbf{A_{r,h}} \exp\left[i(\varphi_\mathbf{h} + \varphi_\mathbf{r})\right]$$
$$\mathbf{M'_r}(\mathbf{t}) = \mathbf{A_{r,t}} \exp\left[i(\varphi_\mathbf{t})\right],$$

in which $\mathbf{A_{r,h}}$ and $\mathbf{A_{r,t}}$ are the amplitudes of the entities modulated by the relations. In this paper, we provide a specific format of $\mathbf{A_{r,h}}$ and $\mathbf{A_{r,t}}$ to map the entities into relation-adaptive rotational orbits with different semantic density:

$$\mathbf{A_{r,h}} = 1 + \lambda\cos(\varphi_\mathbf{h} + \phi_\mathbf{r})$$
$$\mathbf{A_{r,t}} = 1 + \lambda\cos(\varphi_\mathbf{t} + \phi'_\mathbf{r}),$$

where $\lambda \in (0, 1)$ is the coefficient. For OpticE, the amplitudes, which fluctuate between $[1-\lambda, 1+\lambda]$, are controlled by different relations through the extra parameters $\phi_\mathbf{r}$ and $\phi'_\mathbf{r}$.

OpticE is a lightweight model with high storage efficiency, as demonstrated in Table 1. By reusing phase parameters $\varphi_\mathbf{h}$ and $\varphi_\mathbf{t}$ in $\mathbf{A_{r,h}}$ and $\mathbf{A_{r,t}}$, only $2N_r * k$ (N_r is the number of relations) extra parameters are needed. However, for other rotation-based models like aOpticE, we need $N_e * k$ (N_e is the number of entities) parameters to represent the amplitudes. The number of parameters is reduced by $(N_e - 2N_r) * k$, where $N_r \ll N_e$ in KGs.

3.3 Training

Negative sampling is essential to train the models well. This approach is an effective method derived from word embedding (Mikolov et al., 2013) and has been proven powerful for KG embedding

Table 2: Statistics of the datasets. The last column is the number of the observed triplets in each test, validation and training set.

Dataset	$\mathbf{N_e}$	$\mathbf{N_r}$	# Te./Val./Tr.
FB15k	14,951	1,345	59k / 50k / 483k
WN18	40,943	18	5k / 5k / 141k
FB15k-237	14,541	237	20k / 18k / 272k
WN18RR	40,943	11	3k / 3k / 87k

(Huang et al., 2020; Krompaß et al., 2015; Sun et al., 2019). Similar to (Sun et al., 2019), the self-adversarial sampling method is used:

$$\mathcal{L} = -\log\sigma(\gamma + f_r(\mathbf{h},\mathbf{t})) - \sum_{i=1}^{n} p(h'_i, r, t'_i)\log\sigma(-f_r(\mathbf{h}'_i,\mathbf{t}'_i) - \gamma),$$

where (h'_i, r, t'_i) is the i-th negative sample of the true triplet (h, r, t), γ is a fixed margin and σ is the sigmoid function. In OpticE and pOpticE, $f_r(\mathbf{h},\mathbf{t})$ need to be reduced by a proper times to fit the value of γ. Furthermore,

$$p(h'_i, r, t'_i) = \frac{\exp \alpha f_r(\mathbf{h}'_i,\mathbf{t}'_i)}{\sum_j \exp \alpha f_r(\mathbf{h}'_j,\mathbf{t}'_j)} \quad (4)$$

is the weight of the i-th negative sample, where α is the adversarial temperature. According to Equation (4), negative samples with higher scores are assigned with higher weights during the training phase. This is an effective way to filter out high-scoring negative samples and lower their scores efficiently. Apparently, this technique is better than assigning the same average weight to each sample.

However, there are obvious defects in the self-adversarial sampling methods above. During the sampling process, those missing true triplets to be predicted will inevitably suffer from a severe penalty if they are sampled, which will undermine the prediction accuracy. Here, we take advantage of the Bernoulli sampling strategy described in (Wang et al., 2014) to reduce the chance of sampling false negative samples. Instead of replacing the head and tail uniformly, we use a Bernoulli distribution to decide whether to corrupt the head or the tail. For example, given a triplet *(Biden, nationality, the U.S.)*, since *nationality* is a many-to-one relation, it is safer to change the entity *the U.S.* to obtain a negative sample than altering entity *Biden*. In Algorithm 1, the process of the Bernoulli self-adversarial sampling method is clarified.

4 Experiments

In this part, the experimental conditions are elaborated, and the performance of the proposed models is compared. Different negative sampling methods are tested. Finally, we examine the effects of the amplitude of OpticE. All the experiments are conducted on an NVIDIA-v100 GPU. The code for this paper is available on https://github.com/guixiangyu1/OpticE.

4.1 Experimental Settings

Four widely used datasets,FB15k (Bordes et al., 2013), WN18 (Bordes et al., 2013), FB15k-237 (Toutanova and Chen, 2015) and WN18RR (Dettmers et al., 2018), are benchmark datasets in our experiments. All these datasets contain plenty of multimapping instances. The statistics of these datasets are presented in Table 2. The models are trained on the training set and tested on the test set.

- **FB15k** contains tremendous entities and multimapping relations from real-world settings and is extracted from Freebase (Bollacker et al., 2008).
- **WN18** is extracted from WordNet (Miller, 1995) and is designed to manipulate the semantic relations between words and phrases.
- **FB15k-237** is a subset of FB15k. To avoid the data leakage problem in FB15k, duplicate triplets and direct links are removed in FB15k-237.
- **WN18RR** is a subset of WN18. WN18RR follows the same processes as FB15k-237.

Hyperparameters. During the training process, Adam (Kingma and Ba, 2014) is used to optimize OpticE. Grid search is used and the range of hyperparameters is set as follows: the batch size $b \in [512, 1024, 2048]$, the number of embedding dimensions $k \in [500, 1000, 1500, 2000]$, the temperature for Bernoulli self-adversarial sampling $\alpha \in [0.5, 1.0]$, the negative sample size $n \in [50, 100, 200, 256]$ and the margin $\gamma \in [6, 9, 12, 15]$. All amplitudes and phases are initialized uniformly. The range of the phases is $[0, 2\pi)$.

Evaluation. All the models are tested their link prediction performance with three standard evaluation metrics: the mean reciprocal rank (MRR), mean rank (MR) and hits at N (Hits@N). Given a true test triplet, its head entity or tail entity is

Table 3: Link prediction results on FB15k and WN18. We take TransE results from (Nickel et al., 2016), and DistMult results from (Kadlec et al., 2017). Other results are extracted as benchmarks from the corresponding original papers.

Method	FB15k					WN18				
	MR	MRR	Hits@1	Hits@3	Hits@10	MR	MRR	Hits@1	Hits@3	Hits@10
TransE	-	..463	.297	.578	.749	-	.495	.113	.888	.943
DistMult	42	.798	-	-	.893	655	.797	-	-	.946
HolE	-	.524	.402	.613	.739	-	.938	.930	.945	.949
ComplEx	-	.692	.599	.759	.840	-	.941	.936	.945	.947
ConvE	51	.657	.558	.723	.831	374	.943	.935	.946	.956
RotatE	40	.797	.746	.830	.884	309	.949	.944	.952	.959
RatE	**24**	**.810**	.724	**.859**	**.898**	**180**	.950	.944	.953	**.962**
pOpticE	42	.792	.742	.823	.881	290	.948	.940	.952	.960
aOpticE	45	.788	.736	.822	.879	362	.946	.940	.952	.958
OpticE	39	.804	**.756**	.837	.889	261	**.951**	**.946**	**.955**	.961

Table 4: Link prediction results on FB15k-237 and WN18RR. The results of [♠] are taken from (Dettmers et al., 2018). Other results are extracted as benchmarks from the corresponding original papers.

Method	FB15k-237					WN18RR				
	MR	MRR	Hits@1	Hits@3	Hits@10	MR	MRR	Hits@1	Hits@3	Hits@10
TransE	357	.294	-	-	.465	3384	.226	-	-	.501
DistMult [♠]	254	.241	.155	.263	.419	5110	.43	.39	.44	.49
ComplEx [♠]	339	.247	.158	.275	.428	5261	.44	.41	.46	.51
ConvE [♠]	244	.325	.237	.356	.501	4187	.43	.40	.44	.52
RotatE	177	.338	.241	.375	.533	3340	.476	.428	.492	.571
HAKE	-	.346	.250	.381	.542	-	**.497**	.452	**.516**	.582
RatE	172	.344	.261	.382	.541	2860	.488	.441	.506	**.590**
pOpticE	183	.329	.232	.366	.525	3182	.477	.435	.491	.559
aOpticE	186	.340	.244	.377	.536	3510	.473	.425	.489	.563
OpticE	**151**	**.359**	**.264**	**.398**	**.550**	**1930**	**.497**	**.453**	.512	.585

corrupted and we rank all the candidate triplets by their scores from $f_r(h', t)$ or $f_r(h, t')$. These ranks are calculated by filtering out all the correct triplets except the one to be predicted. Higher MRR, Hits@N and lower MR suggest better performance.

4.2 Main Results

The existing state-of-the-art models are compared with ours in terms of link prediction. They are TransE (Bordes et al., 2013), DistMult (Yang et al., 2015), ComplEx (Trouillon et al., 2016), HolE (Nickel et al., 2016), ConvE (Dettmers et al., 2018), RotatE (Sun et al., 2019), HAKE (Zhang et al., 2020) and RatE (Huang et al., 2020).

In Tables 3 and 4, the main results obtained on the link prediction task are listed. RotatE surpasses all the previous models by simultaneously modeling different kinds of relations in KGs, including symmetry, antisymmetry, inversion and composition (Sun et al., 2019). HAKE, RatE and our OpticE can be considered members of the RotatE family because they all employ the rotation transformation approach and are capable of representing the four types of relations described above. By optimizing the hierarchy and ambiguity aspects, HAKE and RatE outperform RotatE to a large extent. According to the results, by mapping entities into different relation-adaptive semantic spaces, OpticE is comparable to these two models and even exceeds them in terms of most evaluation metrics.

The performances of different modulation styles are also compared. pOpticE (with only phase modulation) can complete the link prediction task well to some extent. By assigning trainable amplitudes decided by entities, aOpticE optimizes the corresponding outcomes. OpticE, with relation-adaptive modulated amplitude, greatly surpasses aOpticE in four datasets, which verifies that it is beneficial to use relation-adaptive amplitude modulation rather than entity-adaptive amplitude modulation.

4.3 Performance for Different Relation Types

We further investigate the performance of OpticE when processing prediction tasks with different relation types. According to the general classification method given by (Wang et al., 2014), the relations are classified into one-to-one, one-to-many,

Table 5: Performance on FB15k-237 for different relation types. TransE and RotatE are trained according to the code released by (Sun et al., 2019) with their best configurations.

	Head Prediction (MRR)				Tail Prediction (MRR)			
Relation Type	1-to-1	1-to-M	M-to-1	M-to-M	1-to-1	1-to-M	M-to-1	M-to-M
TransE	.491	.453	.085	.255	.481	.074	.741	.361
RotatE	.501	.472	.092	.261	.488	**.075**	.748	.368
OpticE (ours)	**.505**	**.484**	**.113**	**.285**	**.499**	.073	**.777**	**.390**

Table 6: Performance of RotatE and OpticE with different negative sampling methods. For FB15k-237, the negative sample size is set to 256. For WN18RR, the negative sample size is set to 100. Other conditions remain the same.

	FB15k-237					WN18RR				
	MR	MRR	Hits@1	Hits@3	Hits@10	MR	MRR	Hits@1	Hits@3	Hits@10
RotatE(uni.)	186	.295	.203	.326	.479	**3226**	.471	.425	.487	.564
RotatE(adv.)	**178**	.335	.238	.374	**.531**	3338	.477	.429	.493	.574
RotatE(badv.)	179	**.337**	**.241**	**.374**	**.531**	3267	**.478**	**.430**	**.496**	**.578**
OpticE(uni.)	152	.348	.254	.385	.537	2267	.486	.442	.501	.568
OpticE(adv.)	168	.354	.260	.390	.541	2890	.488	.445	.504	.570
OpticE(badv.)	**151**	**.359**	**.264**	**.398**	**.550**	1930	**.497**	**.453**	**.512**	**.585**

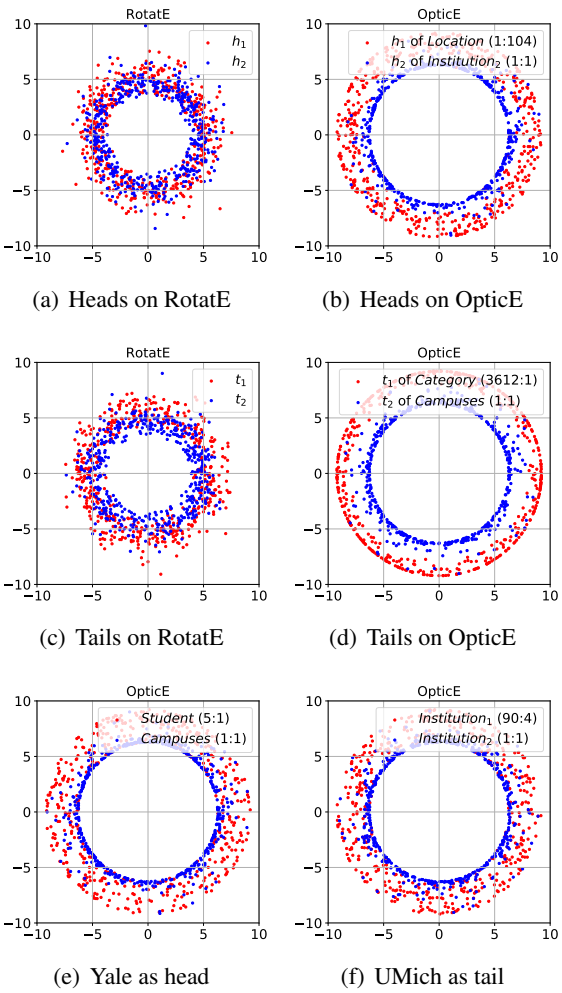

(a) Heads on RotatE (b) Heads on OpticE

(c) Tails on RotatE (d) Tails on OpticE

(e) Yale as head (f) UMich as tail

Figure 2: Visualization of several entities on FB15k-237. The Mid of h_1 and h_2 are /m/04ztj and /m/07tg4; the Mid of t_1 and t_2 are /m/08mbj5d and /m/0bwfn.

many-to-one and many-to-many types. The results obtained by OpticE on the FB15k-237 dataset are given in Table 5 and compared with those of TransE and RotatE.

We can see that OpticE achieves improvements for almost all types of relations, except the one-to-many tail prediction relations for which all the methods obtain low MRR scores. It is difficult to predict the 'many' parts of one-to-many and many-to-one relations. This is in line with the common sense notion that 'many' indicates more uncertainty. By employing relation-adaptive amplitudes, OpticE alleviates this problem to some extent.

4.4 Analysis of Negative Sampling Methods

We apply different negative sampling methods to OpticE and RotatE and record the results in Table 6. Uniform sampling (uni.), self-adversarial sampling (adv.) and Bernoulli self-adversarial sampling (badv.) are alternated in the models during the test. These methods are evaluated based on the results in Table 6. During the sampling phase, Bernoulli sampling can decrease the chance of sampling false negative samples. As indicated in the table, compared with adv. and uni. sampling, models with badv. sampling achieve the best result. From the perspective of models, OpticE outperforms RotatE with the same negative sampling method, which proves the excellent representation ability of OpticE.

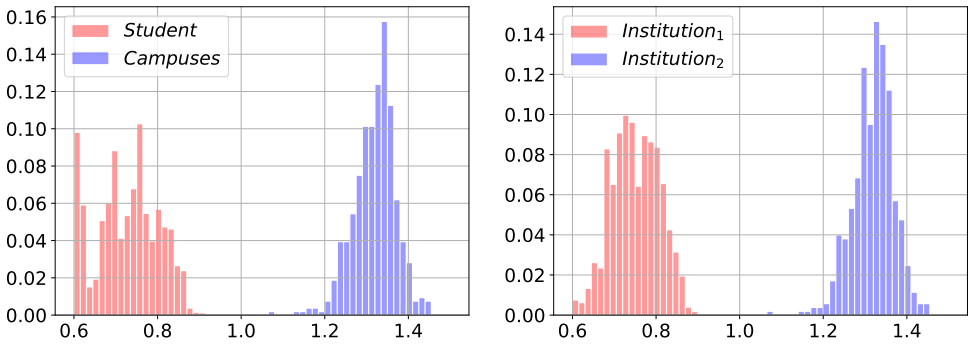

(a) *Student* vs. *Campuses* (b) *Institution*$_1$ vs. *Institution*$_2$

Figure 3: Distribution histogram of the amplitudes. (a) For *Student* (many-to-one) and *Campuses* (one-to-one), we take the average amplitude of their heads and calculate the corresponding relative frequency. (b) For $Institution_1$ (many-to-many) and $Institution_2$ (one-to-one), we take the average amplitude of their tails and calculate the corresponding relative frequency.

4.5 Analysis Regarding the Amplitude

To verify the effects of the amplitude, we monitor and visualize it in OpticE. we visualize the heads and tails on FB15k-237 with several relations, as shown in Figure 2. In Figure 3, we test the amplitude distribution of different kinds of relations.

Visualization of relation-adaptive amplitudes in OpticE. In Figure 2, all the entities are mapped into a 2D space with the data of their first 500 dimensions, based on a complex coordinate system. To make the property of the amplitude more obvious, we take the logarithm operation on the amplitudes. Since the amplitudes and moduli are less than 1, the smaller radii in the figure will actually indicate larger amplitudes or moduli. The numbers in the parentheses are $(hpt : tph)$ of the relations.

In Figure 2(a) and (b), (c) and (d), RotatE mixes entities in the same space while OpticE can distinguish them by different relations. The radii under one-to-one relations ($Institution_2$ and $Campuses$) are smaller in OpticE, which means the entities are mapped into sparse semantic space with larger amplitudes. In Figure 2(e) and Figure 2(f), although *Yale* is the head entity of both relations, entity *Yale* is mapped into sparse semantic space by relation *Campuses* (one-to-one) and dense semantic space by relation *Student* (many-to-one). A similar situation occurs with *UMich* on $Institution_1$ and $Institution_2$. These visualizations prove that OpticE can tackle the semantic ambiguity problem with relation-adaptive amplitudes.

Distribution of amplitudes of different relations. The relation between the values of amplitudes and the multimapping properties of the relation is investigated. As illustrated in Figure 3, the amplitudes of injective (one-to-one) relations ($Campuses$ and $Institution_2$) are larger than those of noninjective relations ($Student$ and $Institution_1$). The reason is that for multimapping relations, the semantics of entities overlap with each other. Then with smaller amplitudes in a dense semantic space, similar semantics are more convenient to express. For the one-to-one relations, entities are exclusive and there are semantic gaps between them. With larger amplitudes in a sparse semantic space, each entity can disambiguate with each other more easily. These attributes are also demonstrated in Figure 2, showing that multimapping relations have smaller radii (red points) and one-to-one relations with larger ones (blue points).

5 Conclusion

We embed KGs with a novel optic interference perspective to tackle the link prediction problem. Guided by coherence theory, we explore three kinds of modulation methods and analyze their properties theoretically. With our proposed negative sampling method, OpticE achieves the best score in comparison with other translation- and rotation-based models. Its ability to disambiguate entities in multiple mapping relations is verified experimentally. The results suggest that we can alleviate semantic ambiguity in rotation-based models by mapping entities into relation-adaptive orbits with different semantic densities.

Acknowledgement

This work was supported in part by National Natural Science Foundation of China under Grants No.62072203.

References

Kurt Bollacker, Colin Evans, Praveen Paritosh, Tim Sturge, and Jamie Taylor. 2008. Freebase: a collaboratively created graph database for structuring human knowledge. In *Proceedings of the 2008 ACM SIGMOD international conference on Management of data*, pages 1247–1250.

Antoine Bordes, Nicolas Usunier, Alberto Garcia-Duran, Jason Weston, and Oksana Yakhnenko. 2013. Translating embeddings for modeling multi-relational data. *Advances in neural information processing systems*, 26:2787–2795.

Antoine Bosselut, Ronan Le Bras, and Yejin Choi. 2021. Dynamic neuro-symbolic knowledge graph construction for zero-shot commonsense question answering. In *Proceedings of the 35th AAAI Conference on Artificial Intelligence (AAAI)*.

Boxi Cao, Hongyu Lin, Xianpei Han, Le Sun, Lingyong Yan, Meng Liao, Tong Xue, and Jin Xu. 2021. Knowledgeable or educated guess? revisiting language models as knowledge bases. *arXiv preprint arXiv:2106.09231*.

Tim Dettmers, Pasquale Minervini, Pontus Stenetorp, and Sebastian Riedel. 2018. Convolutional 2d knowledge graph embeddings. In *Thirty-second AAAI conference on artificial intelligence*.

E. Hecht. 2016. *Optics, Global Edition*. Pearson Education Limited.

Hao Huang, Guodong Long, Tao Shen, Jing Jiang, and Chengqi Zhang. 2020. Rate: Relation-adaptive translating embedding for knowledge graph completion. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 556–567.

Rudolf Kadlec, Ondrej Bajgar, and Jan Kleindienst. 2017. Knowledge base completion: Baselines strike back. *arXiv preprint arXiv:1705.10744*.

Diederik P Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*.

Denis Krompaß, Stephan Baier, and Volker Tresp. 2015. Type-constrained representation learning in knowledge graphs. In *International semantic web conference*, pages 640–655. Springer.

Yankai Lin, Zhiyuan Liu, Maosong Sun, Yang Liu, and Xuan Zhu. 2015. Learning entity and relation embeddings for knowledge graph completion. In *Proceedings of the Twenty-Ninth AAAI Conference on Artificial Intelligence*, pages 2181–2187.

Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. 2013. Distributed representations of words and phrases and their compositionality. In *Advances in neural information processing systems*, pages 3111–3119.

George A Miller. 1995. Wordnet: a lexical database for english. *Communications of the ACM*, 38(11):39–41.

Maximilian Nickel, Lorenzo Rosasco, and Tomaso Poggio. 2016. Holographic embeddings of knowledge graphs. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 30.

Maximilian Nickel, Volker Tresp, and Hans-Peter Kriegel. 2011. A three-way model for collective learning on multi-relational data. In *Icml*, volume 11, pages 809–816.

Andrea Rossi, Denilson Barbosa, Donatella Firmani, Antonio Matinata, and Paolo Merialdo. 2021. Knowledge graph embedding for link prediction: A comparative analysis. *ACM Transactions on Knowledge Discovery from Data (TKDD)*, 15(2):1–49.

Bilin Shao, Xiaojun Li, and Genqing Bian. 2021. A survey of research hotspots and frontier trends of recommendation systems from the perspective of knowledge graph. *Expert Systems with Applications*, 165:113764.

Zhiqing Sun, Zhi-Hong Deng, Jian-Yun Nie, and Jian Tang. 2019. Rotate: Knowledge graph embedding by relational rotation in complex space. *arXiv preprint arXiv:1902.10197*.

Kristina Toutanova and Danqi Chen. 2015. Observed versus latent features for knowledge base and text inference. In *Proceedings of the 3rd workshop on continuous vector space models and their compositionality*, pages 57–66.

Théo Trouillon, Johannes Welbl, Sebastian Riedel, Éric Gaussier, and Guillaume Bouchard. 2016. Complex embeddings for simple link prediction. In *International conference on machine learning*, pages 2071–2080. PMLR.

Denny Vrandečić and Markus Krötzsch. 2014. Wikidata: a free collaborative knowledgebase. *Communications of the ACM*, 57(10):78–85.

Zhen Wang, Jianwen Zhang, Jianlin Feng, and Zheng Chen. 2014. Knowledge graph embedding by translating on hyperplanes. In *AAAI*, volume 14, pages 1112–1119. Citeseer.

Bishan Yang, Scott Wen-tau Yih, Xiaodong He, Jianfeng Gao, and Li Deng. 2015. Embedding entities and relations for learning and inference in knowledge bases. In *Proceedings of the International Conference on Learning Representations (ICLR) 2015*.

Zhanqiu Zhang, Jianyu Cai, Yongdong Zhang, and Jie Wang. 2020. Learning hierarchy-aware knowledge graph embeddings for link prediction. In *Thirty-Fourth AAAI Conference on Artificial Intelligence*, pages 3065–3072. AAAI Press.

A More Details about Coherence Theory

Given $E_1 = A_1 e^{i\varphi_1}$ and $E_2 = A_2 e^{i\varphi_2}$, after the superposition of the two light waves, the formed synthetic light is

$$\begin{aligned} E =& A_1 e^{i\varphi_1} + A_2 e^{i\varphi_2} \\ =& A_1 \cos\varphi_1 + A_2 \cos\varphi_2 \\ &+ i(A_1 \sin\varphi_1 + A_2 \sin\varphi_2). \end{aligned}$$

Then, we have

$$\begin{aligned} ||E||^2 =& (A_1 \cos\varphi_1 + A_2 \cos\varphi_2)^2 \\ &+ (A_1 \sin\varphi_1 + A_2 \sin\varphi_2)^2 \\ =& A_1^2 + A_2^2 + 2A_1A_2\cos(\varphi_1 - \varphi_2). \end{aligned}$$

The final intensity is $I = \frac{1}{2}||E||^2$. For convenience, the coefficient $\frac{1}{2}$ is ignored. The synthetic light intensity of two coherent lights can be noted as

$$\begin{aligned} I &= \langle E_1, E_2 \rangle \\ &= A_1^2 + A_2^2 + 2A_1A_2\cos(\varphi_1 - \varphi_2). \end{aligned}$$

To be more intuitive, we modify the intensity as

$$\begin{aligned} I &= ||E_1 - E_2||^2 \\ &= A_1^2 + A_2^2 - 2A_1A_2\cos(\varphi_1 - \varphi_2). \end{aligned}$$

Then, the phase difference between φ_1 and φ_2 is important during the process ($k \in \mathcal{Z}$):

$$I = \begin{cases} (A_1 + A_2)^2, & \varphi_1 - \varphi_2 = (2k+1)\pi, \\ A_1^2 + A_2^2, & \varphi_1 - \varphi_2 = (\frac{1}{2} + k)\pi, \\ (A_1 - A_2)^2, & \varphi_1 - \varphi_2 = 2k\pi. \end{cases}$$

When $\cos(\varphi_1 - \varphi_2) = 1$, the intensity reaches its maximum as $(A_1 + A_2)^2$. When $\cos(\varphi_1 - \varphi_2) = 0$, the interference effect disappears, and intensity is the sum of the individual intensities as $A_1^2 + A_2^2$. The extreme case is **total destructive interference**, in which the waves tend to cancel, and the intensity tends to be zero. This extreme case occurs only if $A_1 = A_2$ and $\varphi_1 = \varphi_2$ (suppose $\varphi_1, \varphi_2 \in (0, 2\pi]$).

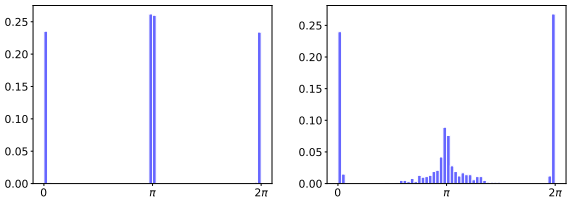

(a) *similar_to* (b) *spouse*

Figure 4: Distribution histogram of the phases of *_similar_to* in WN18RR and *spouse* in FB15k-237.

B Reasoning in Symmetric Relations

OpticE can infer symmetric, anti-symmetric, inversion and composition relations. We give the proof for symmetric relations here, and others are omitted in this paper.

For a symmetric relation r with two positive cases, (h, r, t) and (t, r, h), to satisfy the coherence condition of phase matching,

$$\begin{aligned} \varphi_h + \varphi_r - \varphi_t &= 2k_1\pi \\ \varphi_t + \varphi_r - \varphi_h &= 2k_2\pi, \end{aligned}$$

where k_1 and $k_2 \in \mathcal{Z}$. By adding the formulas above, we can obtain

$$\varphi_r = (k_1 + k_2)\pi = k\pi,$$

in which $k \in \mathcal{Z}$. Specifically, $\varphi_r = \pm\pi$ and 0 in OpticE.

The phase properties inherited from the rotation models are tested. Compared with translation models, e.g., TransE, rotation transformation models are able to represent symmetric relations. For instance, when (h, r, t) and (t, r, h) are both positive cases, it is trivial for TransE to set r to 0. However, for OpticE, the period of the phase takes effect with a phase $\varphi_r = k\pi$. This is verified in Figure 4, in which the phase of the symmetric relation *_similar_to* and *spouse* is distributed around $\pm\pi$ and 0.

Smoothed Contrastive Learning for Unsupervised Sentence Embedding

Xing Wu[1,2,3], **Chaochen Gao**[1,2]*, **Yipeng Su**[1], **Jizhong Han**[1], **Zhongyuan Wang**[3], **Songlin Hu**[1,2]†
[1]Institute of Information Engineering, Chinese Academy of Sciences, Beijing, China
[2]School of Cyber Security, University of Chinese Academy of Sciences, Beijing, China
[3]Kuaishou Technology, Beijing, China
{wuxing,gaochaochen,suyipeng,hanjizhong,husonglin}@iie.ac.cn
wangzhongyuan@kuaishou.com

Abstract

Unsupervised contrastive sentence embedding models, e.g., unsupervised SimCSE, use the InfoNCE loss function in training. Theoretically, we expect to use larger batches to get more adequate comparisons among samples and avoid overfitting. However, increasing batch size leads to performance degradation when it exceeds a threshold, which is probably due to the introduction of false-negative pairs through statistical observation. To alleviate this problem, we introduce a simple smoothing strategy upon the InfoNCE loss function, termed Gaussian Smoothed InfoNCE (GS-InfoNCE). In other words, we add random Gaussian noise as an extension to the negative pairs without increasing the batch size. Through experiments on the semantic text similarity tasks, though simple, the proposed smoothing strategy brings improvements to unsupervised SimCSE. Our code are available at https://github.com/caskcsg/gsInfoNCE.

1 Introduction

Good sentence representation benefits many natural language processing tasks, and sentence representation learning has been widely studied (Logeswaran and Lee, 2018; Reimers and Gurevych, 2019). Contrastive learning has recently been proposed and extensively explored to learn high-quality sentence representations based on the pre-trained language models(Devlin et al., 2018; Liu et al., 2019). Contrastive learning aims to learn effective representation by pulling close semantically similar sentences while pushing apart dissimilar ones (Hadsell et al., 2006). Among those unsupervised sentence embedding learning methods with contrastive learning, the latest state-of-the-art method, as far as we know, is unsupervised SimCSE (unsup-SimCSE) (Gao et al., 2021). unsup-SimCSE implicitly hypothesizes "dropout" as minimal data augmentation and assumes a sentence is semantically more similar to its augmented counterpart than any other sentence. Though simple, unsup-SimCSE works surprisingly well, performing on par with previously supervised counterparts.

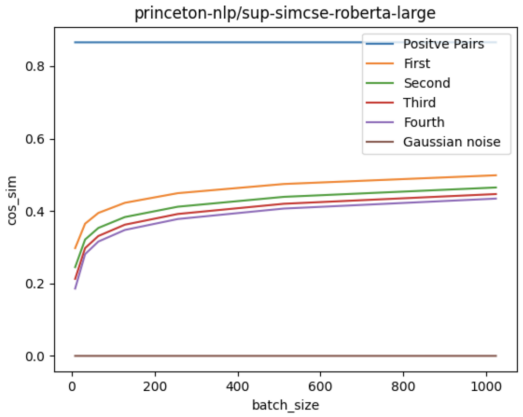

Figure 1: The changing trend of the cosine similarities of the negative pairs in the batch. As the batch_size increases, the mean values of the top 4 cosine similarities also increase, indicating negative pairs with lower confidence exists.

Theoretically, since contrastive learning is carried out among samples within a batch, increasing the batch size will probably bring more adequate comparisons and avoid overfitting. However, according to the original unsup-SimCSE paper (Gao et al., 2021), a larger batch size does not always lead to improvements. The performance even decreases when the batch size exceeds a threshold. We assume that as the batch size increases, more similar sentence samples are probably introduced and easily constitute false-negative pairs, which is detrimental to the learning of the model. We design a probing statistical experiment for different batch sizes to verify our assumption. We use the currently best semantic textual similarity model, i.e., the SimCSE-RoBERTa$_{large}$ (Gao et al., 2021) to measure the cosine similarity of sentence pairs. Randomly sampling a batch with N sentences, we

*The first two authors contribute equally.
†Corresponding author.

Proceedings of the 29th International Conference on Computational Linguistics, pages 4902–4906
October 12–17, 2022.

measure the similarity between all negative pairs within the batch. We calculate the batch's top 4 mean similarity values. We repeat the procedure 100 times and average to eliminate randomness. As shown in Figure1, the top 4 similarity values increase as the batch size increases. It means that, in a larger batch, there will be negative pairs comprised of more similar sentences. When the batch size does not exceed a threshold, the negative pairs of similar sentences are hard negatives and good for training. But when the batch size exceeds, false-negative pairs with higher similarity are introduced, which will mislead the model training. Therefore, achieving sufficient comparison for samples in a "confident" (not too large) batch is particularly important.

As shown in the figure 1, Gaussian noise is far away from all samples and can constitute a very confident negative pair with any sample within a batch. Therefore, we propose to add random Gaussian noise as an extension to the negative pairs without increasing the batch size [1]. In other words, we introduce a simple smoothing strategy upon the InfoNCE loss function by simply adding a Gaussian noise term to the denominator, termed Gaussian Smoothed InfoNCE (GS-InfoNCE). From two perspectives, the Gaussian noise term can be understood as a smoothing strategy. Firstly, the number of negative pairs in a given batch is limited and discrete, and these pairs are used to approximate the negative distribution. We can make the distribution smoother by adding random Gaussian noise to extend the negative pairs. Secondly, from the perspective of the loss function, the denominator of GS-InfoNCE's loss introduces an additional penalty term to avoid overfitting. Through experiments on the semantic text similarity (STS) tasks, GS-InfoNCE outperforms the state-of-the-art unsup-SimCSE by an average Spearman correlation of 1.38%, 0.72%, 1.17% and 0.28% on the base of BERT-base, BERT-large, RoBERTa-base and RoBERTa-large, respectively.

Our contributions can be summarized as follows: we propose GS-InfoNCE for unsup-SimCSE, by introducing a simple smoothing strategy upon the InfoNCE loss function to bring sufficient comparison for samples without increasing the batch size. Our approach can bring improvements to unsup-SimCSE with different model configurations through experiments.

[1] A contemporaneous work (Zhou et al., 2022) has also randomly initialized new negatives based on random Gaussian noises to simulate sampling within the whole semantic space, and devise a gradient-based algorithm to optimize the noise-based negatives.

2 Background: Contrastive Learning

Contrastive learning is a discriminative representation learning framework extensively used for unsupervised representation learning. The core idea is to compare a sentence with a semantically similar one (i.e., positive example) and many semantically dissimilar ones (i.e., negative examples). In this way, the semantically similar sentences are closer in the representation space, while the semantically dissimilar ones are farther apart.

InfoNCE (Chen et al., 2020) propose to take a cross-entropy objective with in-batch negatives, namely the InfoNCE objective function. It is a commonly used loss function for contrast learning by pulling similar sentences closer and pushing dissimilar ones apart in the representation space. Specifically, given a set of sentence pairs: $\mathcal{D} = \left\{\left(x_i, x_i^+\right)\right\}_{i=1}^m$, where x_i and x_i^+ are the ith pair of semantically related sentences. Let $\mathbf{h}_i$ and $\mathbf{h}_i^+$ denote the semantical representations of x_i and x_i^+, for a mini-batch with N pairs, the training loss for (x_i, x_i^+) is:

$$\ell_i = -\log \frac{e^{\text{sim}\left(\mathbf{h}_i, \mathbf{h}_i^+\right)\tau}}{\sum_{j=1}^{N} e^{\text{sim}(\mathbf{h}_i, \mathbf{h}_j)\tau}} \quad (1)$$

where τ is a temperature hyperparameter and $\text{sim}\left(\mathbf{h}_i, \mathbf{h}_i^+\right)$ is the similarity measurement function, which is typically the cosine similarity function.

Unsupervised SimCSE The idea of unsup-SimCSE is quite simple: each positive pair takes the same sentence as input and utilizes "dropout" as minimal data augmentation. In detail, it takes a collection of sentences $\{x_i\}_{i=1}^m$ and use $x_i^+ = x_i$. It feeds the same input to the encoder twice by applying different dropout masks on fully-connected layers and attention probabilities in the transformer. Through training, positive pair embeddings obtained are similar in the representation space.

3 Gaussian Smoothed InfoNCE

We introduce a Gaussian noise term to the InfoNCE loss function, termed Gaussian Smoothed InfoNCE (GS-InfoNCE). Given a Gaussian distribution as

follows: $G \sim N\left(\mu, \sigma^2\right)$, whose mean is μ, and the variance is σ^2, we randomly sample M Gaussian noise vectors from it with the same dimensions as the sentence vector. These vectors constitute high confident negative pairs with each sample in the batch to fill and smooth the representation space. Note that these Gaussian noise vectors will not participate in the positive pair constitution. In that way, the loss function of GS-InfoNCE is denoted as follows:

$$\ell_i = -\log \frac{e^{\text{sim}\left(\mathbf{h}_i, \mathbf{h}_i^+\right)/\tau}}{\sum_{j=1}^{N} e^{\text{sim}(\mathbf{h}_j, \mathbf{h}_i)/\tau} + \lambda \cdot \sum_{k=1}^{M} e^{\text{sim}(\mathbf{g}_k, \mathbf{h}_i)/\tau}} \tag{2}$$

where $\mathbf{g}_k$ is a random Gaussian noise vector, M is the number of Gaussian noise vectors involved in the calculation, and λ is a balance hyperparameter.

The python implementation of GS-InfoNCE is quite simple, with only three lines of codes based on the original InfoNCE implementation in unsup-SimCSE.

4 Experiments

We focus on unsup-SimCSE and replace the original InfoNCE objective loss function with GS-InfoNCE. Following (Gao et al., 2021), the main goal of sentence embeddings is to cluster semantically similar sentences. For a fair comparison, we conduct our experiments on seven semantic textual similarity (STS) tasks introduced below and take STS results to compare sentence embedding methods.

Semantic textual similarity tasks Semantic textual similarity measures the semantic similarity of any two sentences. STS 2012–2016 (Agirre et al., 2012, 2013, 2014, 2015, 2016)and STS Benchmark (Cer et al., 2017) are widely used semantic textual simiarlty benchmark datasets. Following unsup-SimCSE, we use Spearman correlation[2] to measure the correlation between the ranks of predicted scores and the ground-truth.

Training details The training details of unsup-SimCSE can be found in (Chen et al., 2020) and github[3]. Our experimental settings are consistent with the original method. For the Gaussian distribution, we empirically use the standard normal distribution, with $\mu = 0, \sigma^2 = 1$. Additionally, we set $\lambda = 1$ and $M = 3 \times batch\_size$ for all experiments. As illustrated in Figure 1, we have confirmed that increasing the batch size will introduce false-negative pairs with high similarity, so in our experiments, we set the batch size to a moderate size of 64. Following unsup-SimCSE, we conduct experiments on four commonly used models: BERT-base, BERT-large, RoBERTa-base and RoBERTa-large.

Model	SimCSE	+ GS-InfoNCE
BERT$_{base}$	64	64
BERT$_{large}$	64	64
RoBERTa$_{base}$	512	64
RoBERTa$_{large}$	512	64

Table 1: Comparison of $batch\_size$ with or without using GS-InfoNCE in unsup-SimCSE.

Main Results We list the experimental results in Table 2. On the BERT$_{base}$ model, in terms of Spearman correlation, our GS-InfoNCE brings an average increase of 1.38% over unsup-SimCSE on seven test sets, and the maximum gain on STS-B reach 2.85%. On the BERT$_{large}$ model, our GS-InfoNCE gives unsup-SimCSE an average improvement of 0.55% on the 7 test sets, although there is a slight decrease on the SICK15 and SICK-R data sets. On the RoBERTa$_{base}$ and RoBERTa$_{large}$ models, we have a similar situation, with an average improvement of 1.17% and 0.31% on the 7 test sets.

In general, the improvement brought by GS-InfoNCE to unsup-SimCSE is comprehensive. We can fully surpass the previous best model results with the same or smaller batch size in different model configurations, which well demonstrates that our smoothing strategy has played a key role. We believe that a finer search of the parameters can achieve better results and we leave it to our future work.

Analysis: Effect of hyperpraramter M Gaussian random noise constitutes high-confidence negative pairs with the sentences in a batch. M is the number of Gaussian noise vectors involved in the GS-InfoNCE calculation. We further explore the influence of M on the performance of GS-InfoNCE on BERT$_{base}$. We reuse the hyperparameters of the

[2] https://en.wikipedia.org/wiki/Spearman%27s_rank_correlation_coefficient

[3] https://github.com/princeton-nlp/SimCSE

```
    # ... code from origianl unsup-SimCSE above...
    z1, z2 = pooler_output[:,0], pooler_output[:,1]
    cos_sim = cls.sim(z1.unsqueeze(1), z2.unsqueeze(0))
    reg_random = torch.normal(mean, std, size=(reg_size, hidden_size)).to(device)
    reg_cos_sim = cls.sim(z1.unsqueeze(1), reg_random.unsqueeze(0))
    cos_sim = torch.cat((cos_sim, reg_cos_sim),1).to(device)
    labels = torch.arange(cos_sim.size(0)).long().to(cls.device)
    loss_fct = nn.CrossEntropyLoss()
    # ... code from origianl unsup-SimCSE below...
```

Listing 1: Codes in red are regularization modifications to the original InfoNCE loss

Model	STS12	STS13	STS14	SICK15	STS16	STS-B	SICK-R	Avg.
SimCSE-BERT$_{base}$♣	68.40	82.41	74.38	80.91	78.56	76.85	72.23	76.25
+ GS-InfoNCE	**70.12**	**82.57**	**75.21**	**82.89**	**80.23**	**79.70**	**72.70**	**77.63**
SimCSE-BERT$_{large}$♣	70.88	84.16	76.43	**84.50**	79.76	79.26	**73.88**	78.41
+ GS-InfoNCE	**73.75**	**85.09**	**77.35**	84.44	**79.88**	**79.94**	73.48	**78.96**
SimCSE-RoBERTa$_{base}$♣	70.16	81.77	73.24	81.36	80.65	80.22	68.56	76.57
+ GS-InfoNCE	**71.12**	**83.24**	**75.00**	**82.61**	**81.36**	**81.26**	**69.62**	**77.74**
SimCSE-RoBERTa$_{large}$♣	**72.86**	83.99	75.62	**84.77**	**81.80**	81.98	71.26	78.90
+ GS-InfoNCE	71.76	**84.91**	**76.79**	84.35	81.74	**82.97**	**71.71**	**79.21**

Table 2: Sentence embedding performance on semantic textual similarity (STS) test sets in terms of Spearman's correlation. ♣ : results from the official published model by the unsup-SimCSE.

best-performing model and only vary the hyperparameter M. For each M, we train the model until convergence and then select the checkpoint that performs the best on the validation set to evaluate on the test set. The performance statistics are listed in Table 3. As M becomes larger, the performance of GS-InfoNCE on the test set slowly improves. When $M = 3$, the best performance is reached, after which the model performance begins to decline. In general, GS-InfoNCE is not sensitive to M (recommend < 8), making it feasible to apply easily in practical applications.

5 Related Work

Deep and wide models are prone to overfitting, and thus regularization strategies are important to improve their generalization ability. Among them, smoothing is a very commonly used method. (Szegedy et al., 2016; Müller et al., 2019) propose to use label smoothing as a regularization method that makes the clusters between categories more compact and avoids adversarial examples with over high confidence. Text smoothing(Wu et al., 2022; Zhu et al., 2019) also seems to be able to bring further improvements in tasks such as text classification and machine translation by smoothing the one-hot representation of the input text into the probability distribution representation of the dictionary. Our GS-InfoNCE can also be regarded as a smoothing strategy that makes the distribution of negative samples smoother by introducing multiple random Gaussian noise vectors as an extension of the negative examples. Compared with label smoothing and text smoothing, GS-InfoNCE directly uses the standard Gaussian distribution for sampling, largely saving computational costs.

bs=64	**0**×	**0.5**×	**1**×	**2**×
BERT$_{base}$	76.25	76.96	76.90	77.11
bs=64	**3**×	**4**×	**8**×	**16**×
BERT$_{base}$	**77.63**	76.81	76.94	75.57

Table 3: Effect of the hyperprarameter M on BERT$_{base}$. We set M as a multiple of batch size (bs=64). **0**× means the original SimCSE without using GS-InfoNCE.

6 Conclusion and Future Work

This paper proposes GS-InfoNCE for unsupervised SimCSE methods by introducing a simple smoothing strategy upon the InfoNCE loss function to bring sufficient comparison for samples without increasing the batch size. In the future, we will explore how to improve the generalization capability of GS-InfoNCE and verify its effectiveness on more contrastive learning methods.

References

Eneko Agirre, Carmen Banea, Claire Cardie, Daniel Cer, Mona Diab, Aitor Gonzalez-Agirre, Weiwei Guo, Inigo Lopez-Gazpio, Montse Maritxalar, Rada Mihalcea, et al. 2015. Semeval-2015 task 2: Semantic textual similarity, english, spanish and pilot on interpretability. In Proceedings of the 9th international workshop on semantic evaluation (SemEval 2015), pages 252–263.

Eneko Agirre, Carmen Banea, Claire Cardie, Daniel Cer, Mona Diab, Aitor Gonzalez-Agirre, Weiwei Guo, Rada Mihalcea, German Rigau, and Janyce Wiebe. 2014. Semeval-2014 task 10: Multilingual semantic textual similarity. In Proceedings of the 8th international workshop on semantic evaluation (SemEval 2014), pages 81–91.

Eneko Agirre, Carmen Banea, Daniel Cer, Mona Diab, Aitor Gonzalez Agirre, Rada Mihalcea, German Rigau Claramunt, and Janyce Wiebe. 2016. Semeval-2016 task 1: Semantic textual similarity, monolingual and cross-lingual evaluation. In SemEval-2016. 10th International Workshop on Semantic Evaluation; 2016 Jun 16-17; San Diego, CA. Stroudsburg (PA): ACL; 2016. p. 497-511. ACL (Association for Computational Linguistics).

Eneko Agirre, Daniel Cer, Mona Diab, and Aitor Gonzalez-Agirre. 2012. Semeval-2012 task 6: A pilot on semantic textual similarity. In * SEM 2012: The First Joint Conference on Lexical and Computational Semantics–Volume 1: Proceedings of the main conference and the shared task, and Volume 2: Proceedings of the Sixth International Workshop on Semantic Evaluation (SemEval 2012), pages 385–393.

Eneko Agirre, Daniel Cer, Mona Diab, Aitor Gonzalez-Agirre, and Weiwei Guo. 2013. * sem 2013 shared task: Semantic textual similarity. In Second joint conference on lexical and computational semantics (* SEM), volume 1: proceedings of the Main conference and the shared task: semantic textual similarity, pages 32–43.

Daniel Cer, Mona Diab, Eneko Agirre, Inigo Lopez-Gazpio, and Lucia Specia. 2017. Semeval-2017 task 1: Semantic textual similarity-multilingual and cross-lingual focused evaluation. arXiv preprint arXiv:1708.00055.

Ting Chen, Simon Kornblith, Mohammad Norouzi, and Geoffrey Hinton. 2020. A simple framework for contrastive learning of visual representations. In International conference on machine learning, pages 1597–1607. PMLR.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. arXiv preprint arXiv:1810.04805.

Tianyu Gao, Xingcheng Yao, and Danqi Chen. 2021. Simcse: Simple contrastive learning of sentence embeddings. arXiv preprint arXiv:2104.08821.

Raia Hadsell, Sumit Chopra, and Yann LeCun. 2006. Dimensionality reduction by learning an invariant mapping. In 2006 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'06), volume 2, pages 1735–1742. IEEE.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Roberta: A robustly optimized bert pretraining approach. arXiv preprint arXiv:1907.11692.

Lajanugen Logeswaran and Honglak Lee. 2018. An efficient framework for learning sentence representations. arXiv preprint arXiv:1803.02893.

Rafael Müller, Simon Kornblith, and Geoffrey Hinton. 2019. When does label smoothing help? arXiv preprint arXiv:1906.02629.

Nils Reimers and Iryna Gurevych. 2019. Sentence-bert: Sentence embeddings using siamese bert-networks. arXiv preprint arXiv:1908.10084.

Christian Szegedy, Vincent Vanhoucke, Sergey Ioffe, Jon Shlens, and Zbigniew Wojna. 2016. Rethinking the inception architecture for computer vision. In Proceedings of the IEEE conference on computer vision and pattern recognition, pages 2818–2826.

Xing Wu, Chaochen Gao, Meng Lin, Liangjun Zang, Zhongyuan Wang, and Songlin Hu. 2022. Text smoothing: Enhance various data augmentation methods on text classification tasks. arXiv preprint arXiv:2202.13840.

Kun Zhou, Beichen Zhang, Wayne Xin Zhao, and Ji-Rong Wen. 2022. Debiased contrastive learning of unsupervised sentence representations. arXiv preprint arXiv:2205.00656.

Jinhua Zhu, Fei Gao, Lijun Wu, Yingce Xia, Tao Qin, Wengang Zhou, Xueqi Cheng, and Tie-Yan Liu. 2019. Soft contextual data augmentation for neural machine translation. arXiv preprint arXiv:1905.10523.

Knowledge Distillation with Reptile Meta-Learning for Pretrained Language Model Compression

Xinge Ma[†], **Jin Wang**[†,*], **Liang-Chih Yu**[‡,*] and **Xuejie Zhang**[†]
[†]School of Information Science and Engineering, Yunnan University, Yunnan, P.R. China
[‡]Department of Information Management, Yuan Ze University, Taiwan
Contact:wangjin@ynu.edu.cn, lcyu@saturn.yzu.edu.tw

Abstract

The billions, and sometimes even trillions, of parameters involved in pre-trained language models significantly hamper their deployment in resource-constrained devices and real-time applications. Knowledge distillation (KD) can transfer knowledge from the original model (i.e., teacher) into a compact model (i.e., student) to achieve model compression. However, previous KD methods have usually frozen the teacher and applied its immutable output feature maps as soft labels to guide the student's training. Moreover, the goal of the teacher is to achieve the best performance on downstream tasks rather than knowledge transfer. Such a fixed architecture may limit the teacher's teaching and student's learning abilities. Herein, a knowledge distillation method with reptile meta-learning is proposed to facilitate the transfer of knowledge from the teacher to the student. The teacher can continuously meta-learn the student's learning objective to adjust its parameters for maximizing the student's performance throughout the distillation process. In this way, the teacher *learns to teach*, produces more suitable soft labels, and transfers more appropriate knowledge to the student, resulting in improved performance. Unlike previous KD using meta-learning, the proposed method only needs to calculate the first-order derivatives to update the teacher, leading to lower computational cost but better convergence. Extensive experiments on the GLUE benchmark show the competitive performance achieved by the proposed method. For reproducibility, the code for this paper is available at: https://github.com/maxinge8698/ReptileDistil.

1 Introduction

In recent years, pre-trained language models (PLMs) have brought natural language processing to a new era and achieved state-of-the-art performance in a variety of tasks. Based on a multi-layer Transformer architecture (Vaswani et al., 2017), PLMs, such as BERT (Devlin et al., 2019), RoBERTa (Liu et al., 2019), and XLNet (Yang et al., 2019), typically contain hundreds of millions of parameters, making them computationally expensive and inefficient. Consequently, such large-scale models are difficult to be deployed on resource-constrained devices and real-time applications due to high computational complexity, huge storage requirements, and slow inference speed.

Knowledge distillation (KD) (Hinton et al., 2015) has been recommended to compress PLMs. Based on the teacher-student architecture applied, it teaches a smaller student model to reproduce the behavior of a larger teacher model. In practice, the teacher produces output feature maps as soft labels, providing the student with more information than the ground truth of one-hot labels to learn. By minimizing the Kullback-Leibler (KL) divergence between the softened probability distributions of the teacher and student, the student is then trained as an equally-effective model without significant sacrifice in performance.

However, previous studies (Sanh et al., 2019; Sun et al., 2019; Turc et al., 2019; Jiao et al., 2020; Wang et al., 2021) have usually frozen the teacher during the KD process and used its immutable knowledge to teach the student. Inevitably, this architecture has two major limitations: 1) *The parameters of the teacher were fixed for KD*. Concretely, the teacher is frozen throughout the distillation process, and the student can only passively receive the fixed knowledge conveyed by the teacher, which ignores the capacity gap and incompatibility between the teacher and student; 2) *The teacher is unaware of the existence of the student*. Specifically, the teacher model is usually fine-tuned to optimize its performance in downstream applications rather than to optimize the ability to distill the knowledge to the student.

Recent studies (Park et al., 2021; Pham et al., 2021; Zhou et al., 2022) have suggested that up-

*Corresponding authors.

Proceedings of the 29th International Conference on Computational Linguistics, pages 4907–4917
October 12–17, 2022.

dating the teacher together with the student during the KD process, rather than freezing it, can make the student learn better. To be specific, Park et al. (2021) proposed a student-friendly teacher network for training the teacher and student branches to make the teacher aware of the student before distillation and obtain student-friendly knowledge, which is then transferred to the student via vanilla KD. Pham et al. (2021) presented meta pseudo labels where the student is trained based on the pseudo labels generated by the teacher and the teacher is trained based on the performance of the student on labeled data. Zhou et al. (2022) employed a meta-learning strategy to explicitly optimize the teacher with the optimization objective of the student's performance on a hold-out training subset during the KD process, allowing the teacher to evolve continuously to adapt to the current state of the student and better transfer knowledge to it.

The application of model-agnostic meta-learning (MAML) (Finn et al., 2017) in MetaDistil (Zhou et al., 2022) significantly improved the performance of the distilled student by calculating the second-order derivatives of the student's loss on the query set to obtain the update gradients of the teacher. However, limited by the MAML algorithm, there are some concerns in MetaDistil: 1) A query set needs to be designed. MetaDistil split the query set from the training set at a ratio of 9:1, as a result, it missed one-tenth of the training data since this separate query set is not directly used to train the student model. Nevertheless, this one-tenth of the data could lead to significant performance gain, especially when the training data is scarce; 2) The training process may be slow due to the nature of meta-learning. Furthermore, MAML involves the calculation of the second-order derivatives, so that more training time and computing resource are required.

To address the above issues, a knowledge distillation method with reptile meta-learning (Nichol et al., 2018) is proposed to facilitate the transfer of knowledge from the teacher to the student, termed ReptileDistil, where the update gradients of the teacher are approximated by calculating the difference between the parameters of the meta-learner (i.e., teacher) and inner-learner (i.e., student), thereby reducing the computational burden by avoiding calculating the second-order derivatives, and mitigating the tedious operation of partitioning the query set and the performance loss that

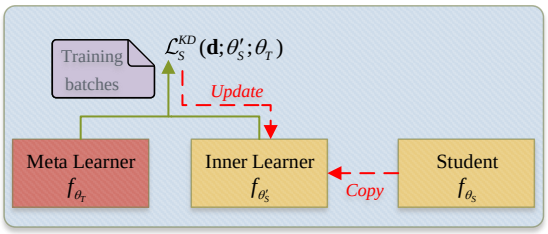

(a) Inner update

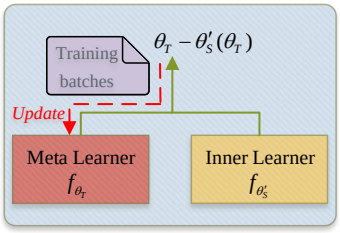

(b) Meta update

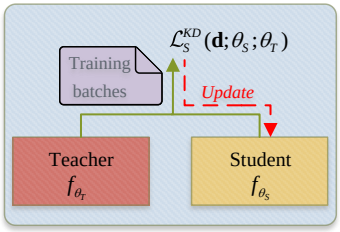

(c) Optimized knowledge distillation

Figure 1: An overview of the proposed ReptileDistil framework.

may be caused. To illustrate the effectiveness and practicality of the proposed ReptileDistil, extensive experiments are conducted on the General Language Understanding Evaluation (GLUE) benchmark (Wang et al., 2018). Empirical results demonstrate that the proposed method can yield meaningful improvements compared with other KD methods in distilling a 12-layer $\text{BERT}_{\text{BASE}}$ model with 110M parameters into a 6-layer BERT_6 model with 66M parameters and 1.94 times speedup.

In summary, the contributions of this paper are as follows:

- A knowledge distillation method with reptile meta-learning is proposed for compressing PLMs, for which the reptile meta-learning is introduced to calculate the update gradients of the teacher model for optimizing it.
- Four different layer-wise parameter update strategies are proposed for updating the parameters of the teacher model, which avoid calculating the second-order derivatives and

lead to better and faster convergence.

- Extensive experiments are conducted on the GLUE benchmark, and the results show that the proposed method performs better in distilling BERT than state-of-the-art KD methods.

The rest of this paper is organized as follows. Section 2 reviews the related work. Section 3 details the proposed ReptileDistil framework. Section 4 presents extensive experiments. Section 5 provides experimental results and analysis. Section 6 summarizes the paper.

2 Related Work

Model compression is a potential approach to reduce the model size and improve computational efficiency. Currently, the model compression techniques for PLMs can be divided into the following six categories: 1) model pruning (Michel et al., 2019; Fan et al., 2020; Gordon et al., 2020), which removes redundant or less important parameters; 2) weight quantization (Zafrir et al., 2019; Shen et al., 2020), which uses fewer bits to represent the parameters; 3) knowledge distillation (Sanh et al., 2019; Sun et al., 2019; Turc et al., 2019; Jiao et al., 2020; Wang et al., 2021; Zhou et al., 2022), which trains a smaller model that learns from the output feature maps of the original model; 4) module replacing (Xu et al., 2020), which replaces the modules of the original model with more compact substitutes; 5) matrix factorization (Lan et al., 2020), which reduces the parameters using methods such as cross-layer parameter sharing and factorized embedding parameterization; 6) early exit (Zhou et al., 2020; Liu et al., 2020; Xin et al., 2020), which allows the model to exit early at an off-ramp instead of passing through the entire model.

Here, we briefly review state-of-the-art work on investigating knowledge distillation to compress PLMs, especially the popular BERT model. DistilBERT (Sanh et al., 2019) is performed at the pre-training stage, which distills output logits from a pre-trained $\text{BERT}_{\text{BASE}}$ teacher into a 6-layer BERT_6 student initialized by taking one layer out of every two layers of a pre-trained $\text{BERT}_{\text{BASE}}$. BERT-PKD (Sun et al., 2019) is performed at the fine-tuning stage, which distills output logits and hidden states from a fine-tuned $\text{BERT}_{\text{BASE}}$ teacher into a 6-layer BERT_6 student initialized with the first 6 layers of a pre-trained $\text{BERT}_{\text{BASE}}$. PD (Turc et al., 2019) is performed at the pre-training stage, in which a randomly initialized 6-layer BERT_6 student is first trained with a masked language modeling objective and then is distilled with the output logits from a pre-trained $\text{BERT}_{\text{BASE}}$ teacher. TinyBERT (Jiao et al., 2020) first is performed at the pre-training stage, which distills embedding outputs, hidden states, and self-attention distributions from a pre-trained $\text{BERT}_{\text{BASE}}$ teacher into a 4-layer TinyBERT_4 student with a hidden size of 312 and an intermediate size of 1200, or a 6-layer TinyBERT_6 student with a hidden size of 768 and an intermediate size of 3072 as same as BERT_6. These two models are then treated as the initialization of student models for further distillation. After that, at the fine-tuning stage, the output logits, embedding outputs, hidden states, and self-attention distributions from a fine-tuned $\text{BERT}_{\text{BASE}}$ teacher are distilled into the aforementioned TinyBERT_4 student or TinyBERT_6 student on the augmented task-specific dataset. MiniLM v2 (Wang et al., 2021) is performed at the pre-training stage, which distills self-attention relation from the last Transformer layer of a pre-trained $\text{BERT}_{\text{BASE}}$ teacher into a randomly initialized 6-layer BERT_6 student. MetaDistil (Zhou et al., 2022) is performed at the fine-tuning stage, which distills output logits from a pre-trained $\text{BERT}_{\text{BASE}}$ teacher into a 6-layer BERT_6 student obtained from the aforementioned PD (Turc et al., 2019) via a meta-learning strategy.

3 Knowledge Distillation with Reptile Meta-Learning

Figure 1 shows an overview of the proposed ReptileDistil framework. Unlike prior work on improving KD, which usually designed new distillation loss, we aim to improve KD via reptile meta-learning to optimize the parameters of the teacher model adaptively with the distillation process of the student model. This allows the teacher model to adjust its pace to search for the optimal global solution, thus providing more suitable soft labels for the student by considering the current capacity of the latter. In addition, to update the parameters of the teacher, we propose four different layer-wise parameter update strategies to provide faster and better convergence.

3.1 Vanilla Knowledge Distillation

Knowledge distillation aims to transfer hidden knowledge from a larger teacher model f_{θ_T} to a shallow student model f_{θ_S} to improve the perfor-

mance of the student model significantly, where θ_T and θ_S denote the trainable parameters of the teacher and student, respectively.

Formally, consider a labeled dataset $\mathcal{D} = \{\mathbf{x}^{(n)}, y^{(n)}\}_{n=1}^{N}$ containing N training samples, where $\mathbf{x}^{(n)}$ and $y^{(n)}$ are the input feature and corresponding ground-truth label, respectively. The data is passed through the teacher model with L layers, and the output hidden representation of each layer can be obtained by,

$$\mathbf{t}_1^{(n)}, ..., \mathbf{t}_l^{(n)}, ..., \mathbf{t}_L^{(n)} = f_{\theta_T}(\mathbf{x}^{(n)}; \theta_T), \quad (1)$$

where $\mathbf{t}_l^{(n)} \in \mathbb{R}^{d_\mathbf{h}}$ is the output representation of the l-th layer corresponding to the `[CLS]` token, and $d_\mathbf{h}$ is the dimensionality of the model. Particularly, the `[CLS]` output from the last layer is fed into a fully-connected layer with a *softmax* activation function, as follows,

$$\hat{\mathbf{y}}_T^{(n)} = \text{softmax}(\frac{\tanh(W_T\mathbf{t}_L^{(n)} + b_T)}{\tau}), \quad (2)$$

where W_T and b_T are the weights and biases of the classifier, respectively, and $\hat{\mathbf{y}}_T^{(n)}$ denotes the predicted probability distribution of the n-th sample, i.e., the probabilities that the output belongs to the classes, which is usually softened by a temperature hyperparameter τ in the *softmax* activation function to control the degree of smoothness. With a lower temperature, the student focuses more on matching the maximal logits of the teacher outputs. On the contrary, a higher temperature encourages the student to focus on the logits other than the maximal ones. The training objective of the teacher model is a categorical cross-entropy loss between distributions of its predicted probability and the ground-truth label, which is defined as,

$$\mathcal{L}_T^{CE}(\mathcal{D}; \theta_T) = -\frac{1}{N}\sum_{n=1}^{N}\mathbb{I}(y^{(n)}) \circ \log \hat{\mathbf{y}}_T^{(n)}, \quad (3)$$

where $\mathbb{I}(y^{(n)})$ denotes the one-hot vector of the ground-truth label, and $\circ$ means the element-wise multiplication operation.

The student model is smaller with K layers, where $K < L$. Similarly, the intermediate hidden representations of the student model are obtained by,

$$\mathbf{s}_1^{(n)}, ..., \mathbf{s}_k^{(n)}, ..., \mathbf{s}_K^{(n)} = f_{\theta_S}(\mathbf{x}^{(n)}; \theta_S), \quad (4)$$

where $\mathbf{s}_k^{(n)} \in \mathbb{R}^{d_\mathbf{h}}$ is the output representation of the k-th layer corresponding to the `[CLS]` token. Also, the predicted probability distribution of the student on the n-th sample is calculated by,

$$\hat{\mathbf{y}}_S^{(n)} = \text{softmax}(\frac{\tanh(W_S\mathbf{s}_K^{(n)} + b_S)}{\tau}). \quad (5)$$

Same as the teacher, the student is trained on the task-specific objective to fit the training samples, as follows,

$$\mathcal{L}_S^{CE}(\mathcal{D}; \theta_S) = -\frac{1}{N}\sum_{n=1}^{N}\mathbb{I}(y^{(n)}) \circ \log \hat{\mathbf{y}}_S^{(n)}. \quad (6)$$

In addition, a knowledge distillation objective that aligns the behavior of the student and teacher is employed as additional supervision to train the student to learn the generalization ability of the teacher. Specifically, taking the soft output $\hat{\mathbf{y}}_T^{(n)}$ from Eq. (2) as the pseudo label, the student model can learn more from it than the ground-truth label by,

$$\mathcal{L}^{KL}(\mathcal{D}; \theta_S; \theta_T) = \tau^2\frac{1}{N}\sum_{n=1}^{N}\text{KL}(\hat{\mathbf{y}}_T^{(n)}||\hat{\mathbf{y}}_S^{(n)}), \quad (7)$$

where $\text{KL}(\cdot||\cdot)$ means computing the Kullback-Leibler divergence between two distributions, which can force the student model to replicate the behavior of the teacher model by shrinking it. Furthermore, following Hinton et al. (2015), we scale the loss by multiplying τ^2 to ensure that gradient magnitudes are approximately constant when the temperature changes. Therefore, the final training objective for KD is a weighted sum over the task-specific cross-entropy and KL-divergence, i.e.,

$$\mathcal{L}_S^{KD}(\mathcal{D}; \theta_S; \theta_T) = (1-\alpha)\mathcal{L}_S^{CE} + \alpha\mathcal{L}^{KL}, \quad (8)$$

where $\alpha \in [0, 1]$ is a weight hyperparameter used to balance the importance of these two objectives.

3.2 Reptile Meta-Learning for Better Distillation

In the original formulation of vanilla KD and previous KD methods, the teacher model was first fine-tuned and then frozen to transfer knowledge to the student by teaching fixed soft labels or intermediate features to the student for updating the student's parameters θ_S. Instead, we introduce reptile meta-learning (Nichol et al., 2018) to optimize the teacher's parameters θ_T as the student's parameters θ_S are updated. Specifically, we designate the student and teacher as the inner-learner and meta-learner, respectively, to fit the bi-level optimization framework in reptile meta-learning. The

Algorithm 1 Knowledge Distillation with Reptile Meta-Learning (ReptileDistil)

Require: θ_S, θ_T: parameters of the student and teacher models, respectively
Require: λ, μ: learning rate of the studnet and teacher models, respectively
Require: $\mathcal{D}$: training set

while not done **do**
  for each batch of training set $\mathbf{d} \in \mathcal{D}$ **do**
    Copy an inner-learner θ'_S from the student model θ_S: $\theta'_S \leftarrow \theta_S$
    Inner-update θ'_S with $\mathbf{d}$ and θ_T: $\theta'_S \leftarrow \theta'_S - \lambda \frac{\partial \mathcal{L}_S^{KD}(\mathbf{d};\theta'_S;\theta_T)}{\partial \theta'_S}$
    Meta-update θ_T with the updated θ'_S: $\theta_T \leftarrow \theta_T - \mu(\theta_T - \theta'_S(\theta_T))$
    Update θ_S with $\mathbf{d}$ and the updated θ_T: $\theta_S \leftarrow \theta_S - \lambda \frac{\partial \mathcal{L}_S^{KD}(\mathbf{d};\theta_S;\theta_T)}{\partial \theta_S}$
  end for
end while

inner-learner is trained to accomplish a task or a distribution of tasks with the help of the meta-learner, and this procedure is called an inner-loop. In return, the meta-learner is optimized with a meta-objective that generally maximizes the expected performance of the inner-learner after the inner-loop, and this procedure is called a meta-loop. After the above two processes, the teacher can perceive the current state of the student and adjust its parameters in the direction of maximizing the student's performance. Then we use this optimized teacher to distill its more appropriate knowledge to the student, making the student learn better.

Formally, for each training step, we copy the student's parameters θ_S to an inner-learner as θ'_S. Given a batch of training samples $\mathbf{d} = \{\mathbf{x}, y\} \in \mathcal{D}$, the inner-learner is inner-updated with one-step stochastic gradient descent, as follows,

$$\theta'_S \leftarrow \theta'_S - \lambda \frac{\partial \mathcal{L}_S^{KD}(\mathbf{d}; \theta'_S; \theta_T)}{\partial \theta'_S}, \tag{9}$$

where λ is the learning rate of the student model. It is noteworthy that the updated inner-learner's parameters θ'_S are essentially a function of the teacher's parameters θ_T on account of learning θ'_S depends on θ_T, i.e., $\theta'_S(\theta_T)$. The reptile meta-learning is then conducted by measuring the difference between the parameters of the teacher and the inner-learner as approximate gradients to meta-update the teacher's parameters θ_T via stochastic gradient descent, as follows,

$$\theta_T \leftarrow \theta_T - \mu(\theta_T - \theta'_S(\theta_T)), \tag{10}$$

where μ is the learning rate of the teacher model. Finally, we again apply vanilla KD between the updated teacher model and the original student model, as follows,

$$\theta_S \leftarrow \theta_S - \lambda \frac{\partial \mathcal{L}_S^{KD}(\mathbf{d}; \theta_S; \theta_T)}{\partial \theta_S}. \tag{11}$$

Such a meta-learning framework allows the teacher model to adjust its parameters according to the current learning state of the student model for better transferring knowledge to it. The detailed algorithm is illustrated in Algorithm 1.

3.3 Updating of Layer-wise Parameters

Notably, as shown in Eq. (10), the computation of gradients for updating the teacher's parameters requires the participation of the inner-learner's parameters and the teacher's parameters. However, the layers of these models are different, i.e., $K < L$, as mentioned. Taking knowledge distillation from a 12-layer teacher to a 6-layer student as an example, we also proposed four different strategies for updating the parameters of the teacher model. As listed below, the L_T layers of the teacher model will be updated by the corresponding k layers of the student model.

- **First-*k***: The first k layers of the teacher will be updated, i.e., $L_T = \{1, 2, 3, 4, 5, 6\}$.
- **Last-*k***: The last k layers of the teacher will be updated, i.e., $L_T = \{7, 8, 9, 10, 11, 12\}$.
- **Skip-*k***: The $2k$-th layers of the teacher will be updated, i.e., $L_T = \{2, 4, 6, 8, 10, 12\}$.
- **Both-*k***: The $(2k$-1)-th and $2k$-th layers of the teacher will be updated, i.e., $L_T = \{(1, 2), (3, 4), (5, 6), (7, 8), (9, 10), (11, 12)\}$.

Figure 2 shows the details of different strategies for updating the teacher's parameters, and the effect

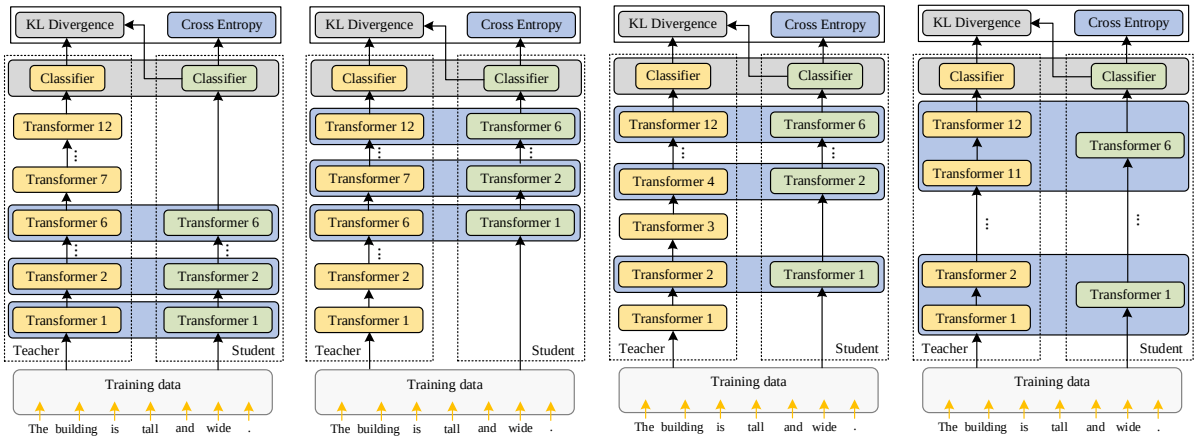

(a) First-k (b) Last-k (c) Skip-k (d) Both-k

Figure 2: Illustration of different strategies for updating the parameters of the teacher model.

of different strategies is discussed in the following section.

4 Experimental Setup

4.1 Datasets

We evaluate the proposed ReptileDistil on the commonly used GLUE benchmark (Wang et al., 2018), which is composed of nine natural language understanding tasks, including CoLA (Warstadt et al., 2019) for linguistic acceptability, SST-2 (Socher et al., 2013) for sentiment analysis, MRPC (Dolan and Brockett, 2005), STS-B (Cer et al., 2017), and QQP[1] for semantic similarity matching, MNLI (Williams et al., 2018), QNLI (Rajpurkar et al., 2016), and WNLI (Levesque et al., 2012) for natural language inference, and RTE (Dagan et al., 2005; Bar-Haim et al., 2006; Giampiccolo et al., 2007; Bentivogli et al., 2009) for textual entailment. We excluded WNLI from GLUE following previous studies (Sanh et al., 2019; Sun et al., 2019; Turc et al., 2019; Jiao et al., 2020; Xu et al., 2020; Wang et al., 2021; Zhou et al., 2022).

4.2 Baselines

We apply the proposed ReptileDistil to a situation in which a 12-layer $\text{BERT}_{\text{BASE}}$ model with 110M parameters is distilled into a 6-layer BERT_6 model with 66M parameters and 1.94 times speedup, and compare it with several state-of-the-art BERT compression approaches, including DistilBERT (Sanh et al., 2019), BERT-PKD (Sun et al., 2019), PD (Turc et al., 2019), TinyBERT (Jiao et al., 2020), BERT-of-Theseus (Xu et al., 2020), MiniLM v2 (Wang et al., 2021), and MetaDistil (Zhou et al., 2022).

4.3 Experimental Settings

In previous work, DistilBERT (Sanh et al., 2019), BERT-PKD (Sun et al., 2019), and BERT-of-Theseus (Xu et al., 2020) initialized their student model by truncating certain layers of a pre-trained $\text{BERT}_{\text{BASE}}$ model. More concretely, both BERT-PKD and BERT-of-Theseus initialized the student model with the first six layers of parameters from the pre-trained $\text{BERT}_{\text{BASE}}$ model. DistilBERT initialized the student model from the pre-trained $\text{BERT}_{\text{BASE}}$ model by taking one of every two layers. Different from the above approaches, TinyBERT (Jiao et al., 2020) initialized the student with the general TinyBERT[2], which can capture the general domain knowledge by learning from intermediate layers of the pre-trained $\text{BERT}_{\text{BASE}}$ model. Therefore, following TinyBERT, we initialize the student model with a 6-layer general TinyBERT_6 for better generalization and performance.

4.4 Implementation Details

The experiments include two stages, i.e., fine-tuning a teacher model and distilling the fine-tuned teacher model into a student model.

For experiments on fine-tuning the teacher, we use the same architecture in the original BERT (Devlin et al., 2019) and fine-tune each task separately. Specifically, for each task, we fine-tune the pre-trained $\text{BERT}_{\text{BASE}}$ model for 5 epochs using the

[1] https://quoradata.quora.com

[2] https://github.com/huawei-noah/Pretrained-Language-Model/tree/master/TinyBERT

Method	CoLA (8.5k) Mcc	SST-2 (67k) Acc	MRPC (3.7k) F1/Acc	STS-B (5.7k) Pear/Spea	QQP (364k) F1/Acc	MNLI (393k) Acc m/mm	QNLI (105k) Acc	RTE (2.5k) Acc	Score
Development set									
$\text{BERT}_{\text{BASE}}$ (Devlin et al., 2019)	58.9	93.0	91.6/87.6	90.2/89.8	88.5/91.4	84.6/84.9	91.2	71.4	-
DistilBERT (Sanh et al., 2019)	51.3	91.3	87.5/-	-/86.9	-/88.5	82.2/-	89.2	59.9	-
BERT-PKD (Sun et al., 2019)	45.5	91.3	85.7/-	-/86.2	-/88.4	81.3/-	88.4	66.5	-
PD (Turc et al., 2019)	-	91.1	89.4/84.9	-	87.4/90.7	82.5/83.4	89.4	66.7	-
TinyBERT (Jiao et al., 2020)	54.0	**93.0**	90.6/86.3	**90.1/89.6**	88.0/91.1	**84.5/84.5**	**91.1**	73.4	-
BERT-of-Theseus (Xu et al., 2020)	51.1	91.5	89.0/-	-/88.7	-/89.6	82.3/-	89.5	68.2	-
MiniLM v2 (Wang et al., 2021)	52.5	92.4	88.9/-	-	-/91.1	84.2/-	90.8	72.1	-
MetaDistil (Zhou et al., 2022)	**58.6**	92.3	91.1/86.8	89.4/89.1	**88.1/91.0**	83.5/83.8	90.4	69.4	-
ReptileDistil	54.8	92.2	**91.6/87.7**	89.5/89.3	87.6/90.1	83.7/83.7	90.5	**75.3**	-
Test set									
$\text{BERT}_{\text{BASE}}$ (Devlin et al., 2019)	52.1	93.5	88.9/84.8	87.7/85.8	71.2/89.2	84.6/83.4	90.5	66.4	78.3
DistilBERT (Sanh et al., 2019)	45.8	92.9	87.6/83.1	71.0/71.0	69.6/88.2	81.6/81.3	88.8	54.1	73.6
BERT-PKD (Sun et al., 2019)	43.5	92.0	85.0/79.9	83.4/81.6	70.7/88.9	81.5/81.0	89.0	65.5	75.6
PD (Turc et al., 2019)	-	91.8	86.8/81.7	-	70.4/88.9	82.8/82.2	88.9	65.3	-
TinyBERT (Jiao et al., 2020)	**51.1**	93.1	87.3/82.6	85.0/83.7	71.6/89.1	**84.6/83.2**	90.4	70.0	78.1
BERT-of-Theseus (Xu et al., 2020)	47.8	92.2	87.6/83.2	85.6/84.1	**71.6/89.3**	82.4/82.1	89.6	66.2	77.1
MiniLM v2 (Wang et al., 2021)	-	92.9	89.1/-	-/84.3	70.9/-	83.8/83.3	90.2	69.2	-
MetaDistil (Zhou et al., 2022)	50.7	**93.5**	88.7/84.7	86.1/85.0	71.1/88.9	83.8/83.2	90.2	67.2	78.0
ReptileDistil	47.9	92.8	**89.2/85.4**	**87.1/85.9**	71.0/89.0	83.6/82.9	**90.4**	**73.5**	**78.5**

Table 1: Experiment results on the development and test sets of GLUE. The numbers and strings under each dataset indicate the number of training samples and the evaluation metrics, respectively. All student models listed above have the same architecture of 6 Transformer layers, 66M parameters, and 1.94 times speed-up, and are distilled from a $\text{BERT}_{\text{BASE}}$ model fine-tuned during the corresponding task. The results on the development set are reported from the corresponding original paper, and the results on the test set are reported from the official leaderboard of GLUE. Note that the column "**Score**" is reported from the official leaderboard and represents the average score for all tasks, including WNLI. **Acc** refers to accuracy, **Mcc** refers to Matthew's correlation, **Pear/Spea** refers to Pearson and Spearman correlation, respectively, and **Acc m/mm** refers to the accuracy of MNLI-m and MNLI-mm, respectively. The best results for each task are marked with boldface.

AdamW optimizer (Loshchilov and Hutter, 2019) and save the best checkpoint on the development set as the teacher, with a batch size of 32, a maximum sequence length of 128, a learning rate of {1e-5, 3e-5, 5e-5} with linear decay. Consequently, we can obtain a teacher model with comparable performance with $\text{BERT}_{\text{BASE}}$ reported on the GLUE official leaderboard[3].

For experiments on distilling the student, to reduce the hyperparameter search space, we set the maximum sequence length and batch size to 128 and 32, respectively, and fix the temperature τ and weight α as 5 and 0.5, respectively. To select the model with the best performance when applying the development set, a grid search strategy is then applied over the sets of the learning rate for the teacher model as {1e-5, 3e-5, 5e-5} and the learning rate for the student model as {1e-5, 3e-5, 5e-5} for 5 epochs and save the best checkpoint.

5 Results and Analysis

5.1 Comparative Results

Table 1 summarizes the comparative results for both the development and test sets of the GLUE tasks. We also report the average score of all tasks, including WNLI, which can be obtained from the GLUE official leaderboard by submitting predictions to the GLUE test server. As the result shows, ReptileDistil achieves state-of-the-art performance on the overall average performance across all tasks of the GLUE benchmark. In detail, the proposed ReptileDistil outperforms baselines on 4 out of 8 tasks, and performs as well or better than $\text{BERT}_{\text{BASE}}$, particularly on small datasets, e.g., RTE, MRPC, and STS-B. Correspondingly, ReptileDistil is not significantly improved on large datasets, e.g., CoLA, SST-2, and MNLI. Nevertheless, it is still very close to MetaDistil and requires less training time and calculated amount. One potential reason is that ReptileDistil updates the teacher's parameters by calculating the differ-

[3] https://gluebenchmark.com/leaderboard

Method	CoLA	SST-2	MRPC	STS-B	QQP	MNLI	QNLI	RTE
ReptileDistil	**54.8**	**92.2**	**91.6/87.7**	**89.5/89.3**	**87.6/90.1**	**83.7/83.7**	**90.5**	**75.3**
w/o Reptile	51.9	91.0	90.0/86.5	88.9/88.6	86.5/89.4	82.8/83.1	90.0	72.8

Table 2: Ablation results in terms of removing reptile meta-learning (w/o Reptile).

Method	CoLA	SST-2	MRPC	STS-B	QQP	MNLI	QNLI	RTE
First-k	53.1	91.9	91.4/87.5	89.5/89.3	87.0/90.1	83.3/83.2	**90.5**	74.7
Last-k	52.5	91.2	91.1/87.2	89.5/89.3	85.9/89.9	83.4/82.9	90.1	74.4
Skip-k	**54.8**	**92.2**	**91.6/87.7**	**89.5/89.3**	**87.6/90.1**	**83.7/83.7**	90.3	**75.3**
Both-k	51.3	90.5	90.8/87.0	89.5/89.3	86.4/89.5	81.9/81.8	90.0	73.6

Table 3: Performance comparison of four different layer-wise parameter update strategies.

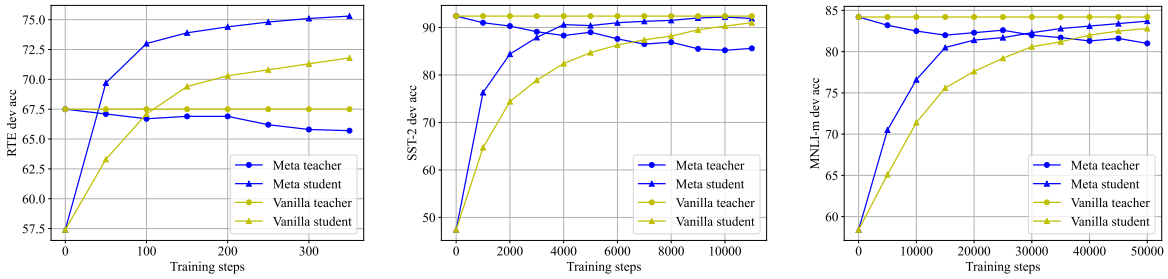

(a) Results on RTE (b) Results on SST-2 (c) Results on MNLI-m

Figure 3: Validation accuracy curves of the teacher and student in ReptileDistil and vanilla KD, which are experimented on the development sets of RTE, SST-2 and MNLI-m datasets, respectively.

ence between the meta-learner (i.e., teacher) and inner-learner (i.e., student) as approximate gradients. Therefore, ReptileDistil can update the teacher model at a large pace on small datasets, leading to better and faster convergence of the teacher model. Conversely, MetaDistil uses the second-order derivatives as gradients to update the teacher's parameters. Although it is smoother than ReptileDistil, it requires a certain number of datasets to accomplish the model convergence. As a result, ReptileDistil is likely to train better in most circumstances, which is critical for improvement on small datasets but may not be so obvious for improvement on large datasets.

5.2 Ablation Analysis

To examine the effectiveness of the proposed ReptileDistil, we conducted an ablation test on the development sets of all tasks in terms of removing reptile meta-learning, allowing ReptileDistil to become vanilla KD where the teacher is fixed instead of dynamically evolving to adapt to the student throughout the distillation process. As the ablation results indicated in Table 2, remarkable performance degradation is observed after removing reptile meta-learning, indicating that updating the teacher model with the current state of the student model is crucial.

5.3 Effect of Updating Strategy

We further investigate the performance gain from four different layer-wise parameter update strategies of First-k, Last-k, Skip-k, and Both-k. Table 3 summarizes the comparative results on the development sets of all tasks. As shown, Skip-k performs slightly better than the other three strategies in most cases since information across every k layers involves low- to high-level semantic representations while the first k layers or the last k layers involve relatively homogeneous semantic representations. Particularly, the Both-k strategy performs to be not competitive with the other three strategies since the Both-k strategy can induce a mismatch between the parameters of the teacher and student, resulting in poor performance.

Method	Training Time (Best)	Training Time (Match)	Memory Footprint	Best Acc/F1
Vanilla KD (Hinton et al., 2015)	10 min	10 min	3.7 GB	90.0/86.5
MetaDistil (Zhou et al., 2022)	31 min	15 min	11.4 GB	91.1/86.8
ReptileDistil	22 min	9 min	8.1 GB	91.6/87.7

Table 4: Comparison of training time and memory footprint of ReptileDistil with vanillaKD and MetaDistil on MRPC. "**Training Time (Best)**" denotes the training time for each method to achieve its own best performance on the development set. "**Training Time (Match)**" denotes the training time for each method to match the best performance of vanilla KD on the development set. All experiments are conducted on a single Nvidia V100 GPU with a batch size of 4.

5.4 Improvement Analysis

Figure 3 shows the validation accuracy curves of the teacher and student in ReptileDistil and vanilla KD on the development sets of the small dataset RTE, the medium dataset SST-2, and the large dataset MNLI-m, respectively. As indicated, the teacher model is always fixed and constant in vanilla KD, whereas in ReptileDistil, it continuously adjusts itself adaptively. Although the accuracy of the teacher model continuously fluctuates and even declines, the student model still maintains a growing accuracy, indicating that the teacher model is not aimed at optimizing its performance but rather at optimizing its teaching ability to enable the student model to learn better and achieve higher performance. The results suggest that under the scenario of knowledge distillation, improving knowledge distillation by enhancing the teaching ability of the teacher model may be another potential direction.

5.5 Performance-Efficiency Tradeoff

MetaDistil (Zhou et al., 2022) inevitably needs to calculate the second-order derivatives for deriving the gradients to update the teacher model. Thus, more computational time and resources are required. However, in the proposed ReptileDistil, this limitation has been greatly improved. Specifically, the proposed ReptileDistil achieved a comparable or even better performance using the approximate gradient computation of reptile meta-learning and the proposed layer-wise parameter update strategies to replace the computation of the second-order derivatives in MAML. Meanwhile, the training time is reduced by approximately a third. To verify this, following MetaDistil, we performed an additional experiment on the MRPC task to compare the computational overhead of the proposed ReptileDistil with MetaDistil and vanilla KD. As shown in Table 4, the memory footprint of ReptileDistil is higher than vanilla KD but lower than MetaDistil. Moreover, both the training time and model convergence time of ReptileDistil are significantly lower than MetaDistil, proving that the proposed method can effectively alleviate the hardware limitation of MetaDistil and achieve better and faster convergence.

6 Conclusions

In this paper, a knowledge distillation method with reptile meta-learning was proposed to compress pre-trained language models, which also utilizes the current performance of the student model during each step of the distillation process as feedback to optimize the teacher model, allowing the teacher to learn to teach and thus produce more appropriate soft labels to teach the student. However, unlike previous knowledge distillation methods applying meta-learning, the proposed method avoids calculating the second-order derivatives and achieves better and faster convergence, owing to the integration of reptile meta-learning and the proposed layer-wise parameter update strategies. Extensive experiments demonstrated the competitive performance achieved by the proposed method. Future work attempts to continue to explore such a dynamic and interactive teacher-student distillation architecture to improve the performance of the distilled model further.

Acknowledgements

This work was supported by the National Natural Science Foundation of China (NSFC) under Grant No.61966038, the Ministry of Science and Technology, Taiwan, ROC, under Grant No.MOST 111-2628-E-155-001-MY2 and the Postgraduate Research and Innovation Foundation of Yunnan University under Grant No.2021Y282. The authors would like to thank the anonymous reviewers for their constructive comments.

References

Roy Bar-Haim, Ido Dagan, Bill Dolan, Lisa Ferro, Danilo Giampiccolo, Bernardo Magnini, and Idan Szpektor. 2006. The second pascal recognising textual entailment challenge. In *Proceedings of the Second PASCAL Challenges Workshop on Recognising Textual Entailment*.

Luisa Bentivogli, Ido Dagan, Hoa Trang Dang, Danilo Giampiccolo, and Bernardo Magnini. 2009. The fifth pascal recognizing textual entailment challenge. In *Proceedings of the Third Text Analysis Conference*.

Daniel Cer, Mona Diab, Eneko Agirre, Iñigo Lopez-Gazpio, and Lucia Specia. 2017. Semeval-2017 task 1: Semantic textual similarity multilingual and cross-lingual focused evaluation. In *Proceedings of the 11th International Workshop on Semantic Evaluation (SemEval 2017)*, pages 1–14.

Ido Dagan, Oren Glickman, and Bernardo Magnini. 2005. The pascal recognising textual entailment challenge. In *Machine Learning Challenges Workshop*, pages 177–190.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. Bert: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL-HLT 2019)*, pages 4171–4186.

William B. Dolan and Chris Brockett. 2005. Automatically constructing a corpus of sentential paraphrases. In *Proceedings of the Third International Workshop on Paraphrasing (IWP 2005)*, pages 9–16.

Angela Fan, Edouard Grave, and Armand Joulin. 2020. Reducing transformer depth on demand with structured dropout. In *Proceedings of the 8th International Conference on Learning Representations (ICLR 2020)*.

Chelsea Finn, Pieter Abbeel, and Sergey Levine. 2017. Model-agnostic meta-learning for fast adaptation of deep networks. In *Proceedings of the 34th International Conference on Machine Learning (ICML 2017)*, pages 1126–1135.

Danilo Giampiccolo, Bernardo Magnini, Ido Dagan, and Bill Dolan. 2007. The third pascal recognizing textual entailment challenge. In *Proceedings of the ACL-PASCAL Workshop on Textual Entailment and Paraphrasing*, pages 1–9.

Mitchell A. Gordon, Kevin Duh, and Nicholas Andrews. 2020. Compressing bert: Studying the effects of weight pruning on transfer learning. In *Proceedings of the 5th Workshop on Representation Learning for NLP (RepL4NLP 2020)*, pages 143–155.

Geoffrey Hinton, Oriol Vinyals, and Jeff Dean. 2015. Distilling the knowledge in a neural network. *arXiv preprint arXiv:1503.02531*.

Xiaoqi Jiao, Yichun Yin, Lifeng Shang, Xin Jiang, Xiao Chen, Linlin Li, Fang Wang, and Qun Liu. 2020. Tinybert: Distilling bert for natural language understanding. In *Findings of the Association for Computational Linguistics (EMNLP 2020)*, pages 4163–4174.

Zhenzhong Lan, Mingda Chen, Sebastian Goodman, Kevin Gimpel, Piyush Sharma, and Radu Soricut. 2020. Albert: A lite bert for self-supervised learning of language representations. In *Proceedings of the 8th International Conference on Learning Representations (ICLR 2020)*.

Hector J. Levesque, Ernest Davis, and Leora Morgenstern. 2012. The winograd schema challenge. In *Proceedings of the Thirteenth International Conference on Knowledge Representation and Reasoning (KR 2012)*, pages 552–561.

Weijie Liu, Peng Zhou, Zhiruo Wang, Zhe Zhao, Haotang Deng, and Qi Ju. 2020. Fastbert: A self-distilling bert with adaptive inference time. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics (ACL 2020)*, pages 6035–6044.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Roberta: A robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692*.

Ilya Loshchilov and Frank Hutter. 2019. Decoupled weight decay regularization. In *Proceedings of the 7th International Conference on Learning Representations (ICLR 2019)*.

Paul Michel, Omer Levy, and Graham Neubig. 2019. Are sixteen heads really better than one? In *Advances in Neural Information Processing Systems (NeurIPS 2019)*.

Alex Nichol, Joshua Achiam, and John Schulman. 2018. On first-order meta-Learning algorithms. *arXiv preprint arXiv:1803.02999*.

Dae Young Park, Moon-Hyun Cha, Changwook Jeong, Dae Sin Kim, and Bohyung Han. 2021. Learning student-friendly teacher networks for knowledge distillation. In *Advances in Neural Information Processing Systems (NeurIPS 2021)*, pages 13292–13303.

Hieu Pham, Zihang Dai, Qizhe Xie, and Quoc V. Le. 2021. Meta pseudo labels. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR 2021)*, pages 11557–11568.

Pranav Rajpurkar, Jian Zhang, Konstantin Lopyrev, and Percy Liang. 2016. Squad: 100,000+ questions for machine comprehension of text. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing (EMNLP 2016)*, pages 2383–2392.

Victor Sanh, Lysandre Debut, Julien Chaumond, and Thomas Wolf. 2019. Distilbert, a distilled version of bert: Smaller, faster, cheaper and lighter. *arXiv preprint arXiv:1910.01108*.

Sheng Shen, Zhen Dong, Jiayu Ye, Linjian Ma, Zhewei Yao, Amir Gholami, Michael W. Mahoney, and Kurt Keutzer. 2020. Q-bert: Hessian based ultra low precision quantization of bert. In *Proceedings of the 34th AAAI Conference on Artificial Intelligence (AAAI 2020)*, pages 8815–8821.

Richard Socher, Alex Perelygin, Jean Y. Wu, Jason Chuang, Christopher D. Manning, Andrew Y. Ng, and Christopher Potts. 2013. Recursive deep models for semantic compositionality over a sentiment treebank. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing (EMNLP 2013)*, pages 1631–1642.

Siqi Sun, Yu Cheng, Zhe Gan, and Jingjing Liu. 2019. Patient knowledge distillation for bert model compression. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP 2019)*, pages 4323–4332.

Iulia Turc, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. Well-read students learn better: On the importance of pre-training compact models. *arXiv preprint arXiv:1908.08962*.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, and Lukasz Kaiser. 2017. Attention is all you need. In *Advances in Neural Information Processing Systems (NeurIPS 2017)*.

Alex Wang, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel R. Bowman. 2018. Glue: A multi-task benchmark and analysis platform for natural language understanding. In *Proceedings of the 2018 EMNLP Workshop BlackboxNLP: Analyzing and Interpreting Neural Networks for NLP*, pages 353–355.

Wenhui Wang, Hangbo Bao, Shaohan Huang, Li Dong, and Furu Wei. 2021. Minilmv2: Multi-head self-attention relation distillation for compressing pretrained transformers. In *Findings of the Association for Computational Linguistics (ACL-IJCNLP 2021)*, pages 2140–2151.

Alex Warstadt, Amanpreet Singh, and Samuel R. Bowman. 2019. Neural network acceptability judgments. *Transactions of the Association for Computational Linguistics*, 7:625–641.

Adina Williams, Nikita Nangia, and Samuel R. Bowman. 2018. A broad-coverage challenge corpus for sentence understanding through inference. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL-HLT 2018)*, pages 1112–1122.

Ji Xin, Raphael Tang, Jaejun Lee, Yaoliang Yu, and Jimmy Lin. 2020. Deebert: Dynamic early exiting for accelerating bert inference. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics (ACL 2020)*, pages 2246–2251.

Canwen Xu, Wangchunshu Zhou, Tao Ge, Furu Wei, and Ming Zhou. 2020. Bert-of-theseus: Compressing bert by progressive module replacing. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP 2020)*, pages 7859–7869.

Zhilin Yang, Zihang Dai, Yiming Yang, Jaime Carbonell, Ruslan Salakhutdinov, and Quoc V. Le. 2019. Xlnet: Generalized autoregressive pretraining for language understanding. In *Advances in Neural Information Processing Systems (NeurIPS 2019)*.

Ofir Zafrir, Guy Boudoukh, Peter Izsak, and Moshe Wasserblat. 2019. Q8bert: Quantized 8bit bert. In *2019 Fifth Workshop on Energy Efficient Machine Learning and Cognitive Computing (EMC2-NIPS 2019)*, pages 36–39.

Wangchunshu Zhou, Canwen Xu, Tao Ge, Julian McAuley, Ke Xu, and Furu Wei. 2020. Bert loses patience: Fast and robust inference with early exit. In *Advances in Neural Information Processing Systems (NeurIPS 2020)*, pages 18330–18341.

Wangchunshu Zhou, Canwen Xu, and Julian McAuley. 2022. Bert learns to teach: Knowledge distillation with meta learning. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (ACL 2022)*, pages 7037–7049.

RotateCT: Knowledge Graph Embedding by Rotation and Coordinate Transformation in Complex Space

Yao Dong[1,2], **Lei Wang**[1*], **Ji Xiang**[1], **Xiaobo Guo**[1], **Yuqiang Xie**[1,2]
[1]Institute of Information Engineering, Chinese Academy of Sciences, Beijing, China
[2]School of Cyber Security, University of Chinese Academy of Sciences, Beijing, China
{dongyao,wanglei,xiangji,guoxiaobo,xieyuqiang}@iie.ac.cn

Abstract

Knowledge graph embedding, which aims to learn representations of entities and relations in knowledge graphs, finds applications in various downstream tasks. The key to success of knowledge graph embedding models are the ability to model relation patterns including symmetry/antisymmetry, inversion, commutative composition and non-commutative composition. Although existing methods fail in modeling the non-commutative composition patterns, several approaches support this pattern by modeling beyond Euclidean space and complex space. Nevertheless, expanding to complicated spaces such as quaternion can easily lead to a substantial increase in the amount of parameters, which greatly reduces the computational efficiency. In this paper, we propose a new knowledge graph embedding method called RotateCT, which first transforms the coordinates of each entity, and then represents each relation as a rotation from head entity to tail entity in complex space. By design , RotateCT can infer the non-commutative composition patterns and improve the computational efficiency. Experiments on multiple datasets empirically show that RotateCT outperforms most state-of-the-art methods on link prediction and path query answering.

1 Introduction

Knowledge graphs (KGs) contain structured facts of the real world. Real-world large-scale KGs such as WordNet (Miller, 1995), YAGO (Suchanek et al., 2007), Freebase (Bollacker et al., 2008) and Nell (Mitchell et al., 2018) have been applied to strengthen the performance of several downstream tasks including recommender systems (Zhang et al., 2016), question answering (Hao et al., 2017), conversation generation (Zhou et al., 2018), relation extraction (Vashishth et al., 2018) and machine translation (Zhao et al., 2020). Generally, KGs have the following features: large-scale and incomplete. Therefore, to exploit the semantic information in KGs, predicting missing links based on the existing facts has gained growing interest in recent years. This task can be divided into two categories according to the length of the path: link prediction and path query answering. Link prediction focuses on single-hop reasoning (e.g., answering the path $s \rightarrow r \rightarrow ?$, where r is a relation), while path query answering (PQA) focuses on multi-hop reasoning (e.g., answering the path query $s \rightarrow path \rightarrow ?$, where $path$ contains multiple relations).

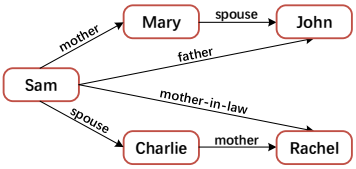

Figure 1: An example in real world. Sam's mother's spouse is John, i.e., Sam's father; Sam's spouse's mother is Rachel, i.e., Sam's mother-in-law. "mother" and "spouse" form a non-commutative composition pattern.

A popular approach for link prediction and PQA is knowledge graph embedding (KGE), which encodes each element in KG into a continuous low-dimensional vector space. The performance of KGE methods greatly relys on the ability of modeling and inferring relation patterns including symmetry/antisymmetry, inversion and composition. In particular, the composition patterns can be further divided into the **commutative composition patterns** and **non-commutative composition patterns**. For example, "spouse" is a symmetric relation and "mother" is an antisymmetric relation. Relations such as "has_part" and "part_of" forms an inversion pattern. The meaning of the composition of "mother" and "spouse" depends on the relative order. By contrast, the composition of "mother" and "mother" has a definite meaning, i.e., "grandmother" (Fig. 1 provides an ex-

*Corresponding author.

Proceedings of the 29th International Conference on Computational Linguistics, pages 4918–4932
October 12–17, 2022.

ample of the non-commutative composition patterns in real world). Existing methods can capture one or more relation patterns. TransE (Bordes et al., 2013) models antisymmetry, inversion and commutative composition patterns by translating relations from head entities to tail entities. RotatE (Sun et al., 2019), which regards relations as rotations from head entities to tail entities in complex space, can model the symmetry/antisymmetry, inversion and commutative composition patterns. However, most approaches fail to model the non-commutative composition patterns, which is essential for learning more meaningful embeddings. Therefore, quaternion-valued methods such as QuatE (Zhang et al., 2019), Rotate3D (Gao et al., 2020) and DualE (Cao et al., 2021) emerge in sight. By the non-commutativity of quaternion, these methods successfully model the non-commutative composition patterns. Nevertheless, a **quaternion** has two more dimensions than a complex number, which **increases the space cost**. Seeking out a balanced solution that can model the non-commutative composition patterns and maintain a relatively low space cost is pressing.

To address this challenge, we revisit complex space and notice an interesting case: the combination of rotation and coordinate transformation in complex plane is non-commutative, which can empower a KGE method the ability of modeling the non-commutative composition patterns. In addition, the **complex number** is two dimensions less than the quaternion, which **reduces the space cost**.

In this paper, we propose a novel method called **RotateCT** for knowledge graph embedding. Our method first translates the origin of complex plane by a relation-specific displacement. Further, each relation is regarded as a rotation about the new origin of complex plane from head entity to tail entity. By combining rotation and coordinate transformation, RotateCT obtains non-commutativity, which enables RotateCT to effectively model the non-commutative composition patterns.

In summary, our contributions are listed as follows: (1) RotateCT provides an elegant way to model the non-commutative composition patterns and improve the parameter efficiency. To the best of our knowledge, this paper is the first to introduce coordinate transformation into the complex plane for modeling non-commutative composition patterns. (2) We provide comprehensive theoretical analyses on the non-commutative property of combining rotation and coordinate transformation, and discuss the inference patterns, parameter efficiency of RotateCT. (3) Experimental results demonstrate that RotateCT outperforms most baseline approaches on link prediction and path query answering.

2 Related Work

In this section, unlike most papers, we roughly divide the existing KGE methods into three categories according to the non-commutativity and discuss their connections to our approach.

2.1 Models without Non-commutativity

TransE (Bordes et al., 2013) is the most representative KGE model, which embeds both entities and relations as vectors in the same embedding space based on the principle $\mathrm{h} + \mathrm{r} \approx \mathrm{t}$, where h, r, t denote head entity, relation and tail entity, respectively. Several variants (Wang et al., 2014; Lin et al., 2015; Ji et al., 2015; Xiao et al., 2016) are proposed to remedy the limitations of TransE when modeling 1-N, N-1 and N-N relations. In addition, TorusE (Ebisu and Ichise, 2018) models triplets on a torus, which is a Non-Euclidean space. Inspired by Euler's identity $e^{i\theta} = cos\theta + isin\theta$, RotatE (Sun et al., 2019) regards translations as rotations from head entities to tail entities in complex space. Moreover, TransC (Lv et al., 2018) and BoxE (Abboud et al., 2020) encode elements by explicitly defining the regions such as hyperspheres or boxes.

RESCAL (Nickel et al., 2011) is the first bilinear model that can perform collective learning via matching the latent semantics between entities and relations. DistMult (Yang et al., 2015) and ComplEx (Trouillon et al., 2016) are proposed to solve the overfitting problem of RESCAL. In addition, HolE (Nickel et al., 2016) absorbs the quintessence from DistMult and ComplEx. Recently, approaches such as SimplE (Kazemi and Poole, 2018) and TuckER (Balazevic et al., 2019b) turn to different forms of decomposition.

Although some of these methods claim to enable the inference of composition patterns, in practice they cannot infer the non-commutative composition patterns, only support the inference of the commutative composition patterns.

2.2 Models with Non-commutativity

Recently, several approaches explore the usage of more sophisticated spaces to obtain the non-

commutativity, which is important for multi-hop reasoning. Specifically, Rotate3D (Gao et al., 2020) and QuatE (Zhang et al., 2019) model relations as rotations in quaternion space with different score functions. DualE (Cao et al., 2021) makes the first attempt to combine rotation and translation by expanding the embedding space to dual quaternion space. DihEdral (Xu and Li, 2019) limits relation matrices to be block diagonal and represents each block with an element in a dihedral group.

Although such approaches take into account the non-commutative composition patterns, they are parameter inefficient due to the complicated vector spaces such as quaternion space.

Our method RotateCT models the non-commutative composition patterns in complex space by introducing the coordinate transformation. Compared to the above three sophisticated spaces, modeling in complex space can significantly improve the parameter efficiency.

2.3 Other Models

Recently, a number of approaches focus on utilizing neural networks. However, the neural networks lack interpretability, and it is difficult to give theoretical analyses from the perspective of inference patterns. Generally, such approaches verify the performance by empirical experiments. ConvE (Dettmers et al., 2018), ConvKB (Nguyen et al., 2018) and InteractE (Vashishth et al., 2020) model the interactions between entities and relations by convolutional neural networks. Additionally, R-GCN (Schlichtkrull et al., 2018) and KBGAT (Nathani et al., 2019) redesign a graph convolutional network and a graph attention network, respectively. Several works tried to exploit more global graph structures like multi-hop paths. Path-RNN (Das et al., 2017) and ROP (Yin et al., 2018) employ RNNs to explicitly model paths. CoKE (Wang et al., 2019) uses a stack of Transformer blocks to model paths.

Compared to the other models, RotateCT is more interpretable, since we can provide comprehensive theoretical analyses of it.

3 Methodology

3.1 RotateCT

Formally, let $\mathcal{E}$ denote the set of entities and $\mathcal{R}$ denote the set of relations. Then a knowledge graph $\mathcal{G}$ is a collection of factual triplets $\{(\mathrm{h}, \mathrm{r}, \mathrm{t})\}$, where $\mathrm{h}, \mathrm{t} \in \mathcal{E}$ and $\mathrm{r} \in \mathcal{R}$. Lowercase letters h, r and t denote the head entity, relation and tail entity, respectively; the corresponding boldface letters $\mathbf{h}$, $\mathbf{r}$ and $\mathbf{t}$ denote the embeddings of them. Note that the i-th element of $\mathbf{h}$ is h_i. Let k denote the dimension of entity and relation embeddings.

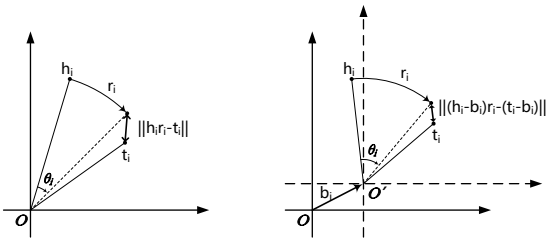

(a) RotatE (b) RotateCT

Figure 2: Illustrations of RotatE and RotateCT in 1 dimension of embeddings. RotatE models r as a rotation. RotateCT models r as a rotation and a displacement, i.e., rotating around the new origin obtained by the displacement.

In this paper, we propose RotateCT to model non-commutative composition patterns in complex space. Inspired by Euler's identity $e^{i\theta} = cos\theta + isin\theta$ and coordinate transformation, our model first projects entities to the complex space, i.e., $\mathbf{h}, \mathbf{t} \in \mathbb{C}^k$. Then we translate the origin of complex planes $\mathbf{O}$ to $\mathbf{O}'$ by a relation-specific displacement $\mathbf{b}$, where $\mathbf{b} \in \mathbb{C}^k$. Further, we define each relation as an element-wise rotation about $\mathbf{O}'$ from head entity $\mathbf{h}$ to tail entity $\mathbf{t}$. In other words, given a golden triplet $(\mathrm{h}, \mathrm{r}, \mathrm{t})$, we expect that:

$$\mathbf{t} - \mathbf{b} = (\mathbf{h} - \mathbf{b}) \circ \mathbf{r}$$

where $\circ$ is the Hadmard (or element-wise) product. Specifically, we have $t_i - b_i = (h_i - b_i)r_i$ for each element of $\mathbf{h}$, $\mathbf{r}$, $\mathbf{t}$ and $\mathbf{b}$. Here, we constrain the modulus of each element of $\mathbf{r} \in \mathbb{C}^k$, i.e., $r_i \in \mathbb{C}$, to be $|r_i| = 1$. Then r_i is of the form $e^{i\theta_i}$, which corresponds a counterclockwise rotation by θ_i radians about the new origin of the complex plane. We define the distance-based score function as follows:

$$d_r(\mathrm{h}, \mathrm{t}) = \sum_{i=1}^{k} \|(h_i - b_i)r_i - (t_i - b_i)\|$$

Optimization. Following Sun et al. (2019), we use a loss function similar to the negative sampling

loss (Mikolov et al., 2013) for optimizing:

$$L = -\sum_{i=1}^{m} \frac{1}{m} \log \sigma(d_r(\mathrm{h}'_i, \mathrm{t}'_i) - \gamma) - \log \sigma(\gamma - d_r(\mathrm{h}, \mathrm{t}))$$

where γ is a fixed margin, σ is the sigmoid function, m is the negative sampling size, and $d_r(\mathrm{h}'_i, \mathrm{t}'_i)$ represents the score of i-th negative triplet. We also use the self-adversarial negative sampling (Sun et al., 2019) for drawing negative samples. Specifically, the negative triplets are generated from the following distribution:

$$p(\mathrm{h}'_j, \mathrm{r}, \mathrm{t}'_j | \{(\mathrm{h}_i, \mathrm{r}, \mathrm{t}_i)\}) = \frac{\exp \alpha f_r(\mathrm{h}'_j, \mathrm{t}'_j)}{\sum_i \exp \alpha f_r(\mathrm{h}'_i, \mathrm{t}'_i)}$$

where $f_r(\mathrm{h}'_i, \mathrm{t}'_i) = -d_r(\mathrm{h}'_i, \mathrm{t}'_i)$, α is the temperature of sampling, and $(\mathrm{h}'_i, \mathrm{r}, \mathrm{t}'_i)$ denotes the i-th negative triplet. The modified loss function with self-adversarial negative sampling is as follows:

$$L = -\sum_{i=1}^{m} p(\mathrm{h}'_i, \mathrm{r}, \mathrm{t}'_i) \log \sigma(d_r(\mathrm{h}'_i, \mathrm{t}'_i) - \gamma) - \log \sigma(\gamma - d_r(\mathrm{h}, \mathrm{t}))$$

In addition, we perform regularization on **h** and **t** to avoid overfitting. Therefore, the final loss function takes the following form:

$$L = -\sum_{i=1}^{m} p(\mathrm{h}'_i, \mathrm{r}, \mathrm{t}'_i) \log \sigma(d_r(\mathrm{h}'_i, \mathrm{t}'_i) - \gamma) - \log \sigma(\gamma - d_r(\mathrm{h}, \mathrm{t})) + \lambda(\|\mathbf{h}\|^2 + \|\mathbf{t}\|^2)$$

where λ is the regularization rate. We utilize Adam (Kingma and Ba, 2014) as the optimizer.

3.2 Discussion

In this part, we first introduce the inference patterns and provide some theoretical analyses of RotateCT. Then we discuss the connections between RotateCT and RotatE.

Inference Patterns. Most knowledge graphs mainly consist of three important relation patterns: **symmetry/antisymmetry**, **inversion** and **composition** (commutative composition and non-commutative composition). See formal definitions of inference patterns in Appendix A.

Properties of RotateCT. RotateCT has the non-commutativity and can infer the most common patterns in KG, as stated next.

Theorem 1 *(**Non-commutativity**) The combination of the operations (rotation and coordinate transformation) in RotateCT is non-commutative. (See proof in Appendix B)*

Theorem 2 *(**Inference ability**) RotateCT can infer the **symmetry/antisymmetry**, **inversion** and **composition** patterns. (See proof in Appendix C. We provide illustrations of RocateCT modeling symmetry and non-commutative composition patterns in Appendix D.)*

Connections to RotatE. Combining rotation and coordinate transformation enables RotateCT to model the non-commutative composition patterns, which RotatE cannot. Fig. 2 provides illustrations of RotatE and RotateCT with only 1- dimensional embedding. RotatE can be viewed as a special case of RotateCT. Specifically, RotateCT will degenerate into RotatE when the displacement $\mathbf{b} = \mathbf{0}$.

4 Experiments

We evaluate RotateCT on two common tasks: link prediction (Bordes et al., 2013) and path query answering (Guu et al., 2015). In addition, we give analyses about the computational efficiency between RotatCT and Rotate3D.

4.1 Link Prediction

Link prediction aims to predict the missing h or t for a triplet (h, r, t), i.e., predicting the head query $? \rightarrow r \rightarrow t$ or the tail query $h \rightarrow r \rightarrow ?$. Thus, link prediction is a single-hop reasoning task.

Datasets. We evaluate RotateCT on four well-established benchmarks: WN18 (Bordes et al., 2013), FB15k (Bordes et al., 2013), WN18RR (Dettmers et al., 2018) and FB15k-237 (Toutanova and Chen, 2015). Please refer to Appendix F for the details of the four benchmarks.

Evaluation Protocol. Similar to most previous models, the link prediction performance of RotateCT is reported on three standard evaluation metrics: Mean Rank (MR), Mean Reciprocal Rank (MRR) and Hits@N, where $N = 1, 3, 10$. MR is the average rank of all correct entites. MRR is the mean reciprocal rank of all correct entities. Hits@N represents the proportion of correct entities whose rank is not larger than N. A lower MR,

Model	WN18					FB15k				
	MR	MRR	Hits@1	Hits@3	Hits@10	MR	MRR	Hits@1	Hits@3	Hits@10
Models without non-commutativity										
TransE (2013)	-	0.495	0.113	0.888	0.943	-	0.463	0.297	0.578	0.749
DistMult (2015)	655	0.797	-	-	0.946	42	0.798	-	-	**0.893**
ComplEx (2016)	-	0.941	0.936	0.945	0.947	-	0.692	0.599	0.759	0.840
SimplE (2018)	-	0.942	0.939	0.944	0.947	-	0.727	0.660	0.773	0.838
TorusE (2018)	-	0.947	0.943	0.950	0.954	-	0.733	0.674	0.771	0.832
RotatE (2019)	309	0.949	0.944	0.952	0.959	40	**0.797**	**0.746**	0.830	0.884
Other models										
ConvE (2018)	374	0.943	0.935	0.946	0.956	51	0.657	0.558	0.723	0.831
R-GCN+ (2018)	-	0.819	0.697	0.929	**0.964**	-	0.696	0.601	0.760	0.842
NKGE (2018)	336	0.947	0.942	-	0.957	56	0.730	0.650	0.790	0.871
Models with non-commutativity										
DihEdral (2019)	-	0.946	0.942	0.949	0.954	-	0.733	0.641	0.803	0.877
QuatE (2019)	338	0.949	0.941	0.954	0.960	41	0.770	0.700	0.821	0.878
Rotate3D (2020)	214	**0.951**	**0.945**	0.953	0.961	39	0.789	0.728	0.832	0.887
DualE (2021)	-	**0.951**	**0.945**	**0.956**	0.961	-	0.790	0.734	0.829	0.881
RotateCT (ours)	**201**	**0.951**	0.944	**0.956**	0.963	**34**	0.794	0.737	**0.834**	0.888

Table 1: Link prediction results on WN18 and FB15k.

a higher MRR and a higher Hits@N indicate the better performance. Filtered results are reported to avoid possibly flawed evaluation.

Implementation Details. We use PyTorch to implement our model and test it on a Tesla V100 GPU. Hyperparameters of RotateCT are determined via grid search according to the MRR on the validation set. In general, the embedding dimension k is selected in {500, 1000}; batch size n is searched in {512, 1024}; the negative sampling size m is picked from {256, 512}; the fixed margin γ is tuned among {6, 9, 12, 24}; self-adversarial temperature α is selected from {0.5, 1.0}. The regularization rate λ is adjusted in {0, 0.1}. Entity embeddings are uniformly initialized, and the phases of relation embeddings are uniformly initialized between $-\pi$ and π. Both the real and imaginary parts of each dimension in displacement $\mathbf{b}$ are initialized to zero. The best hyperparameters settings and are provided in the Appendix E.

Main Results. We select competitive baselines from the most recent publications with good results reported. Our baselines are categorized into three groups: models without non-commutativity, models with non-commutativity and other models, which is consistent with the Section 2.

The empirical results on four benchmarks are reported in Table 1 and Table 2. The best results are in bold and second best results are underlined. We can see that RotateCT outperforms all the baselines on WN18 and WN18RR, and achieves extremely competitive performance on FB15k and FB15k-237. On WN18RR and FB15k-237, the main relation patterns are symmetry/antisymmetry and composition, which validates the effectiveness of combining rotation and coordinate transformation for inferring the non-commutative composition patterns. On WN18 and FB15k, the main relation patterns are symmetry/antisymmetry and inversion. Since RotateCT has no obvious superiority over other state-of-the-art baselines in modeling symmetry/antisymmetry and inversion patterns, the performance improvement is not significant on WN18 and FB15k. Although DihEdral, QuatE, Rotate3D and DualE can model the non-commutative composition patterns, the link prediction performance of these models is still inferior to RotateCT, which indicates that parameter efficiency is crucial. Note that RotateCT, as a complex-valued method, outperforms the quaternion-valued method Rotate3D on all datasets across most metrics, which fully demonstrates that the combination of rotation and coordinate transformation can effectively model the non-commutative composition patterns.

4.2 Path Query Answering

This task is to answer path queries on KGs (Guu et al., 2015). Given a path query q consisting of a start entity s and a path p, the answer of q is the entities that can be reached from s via p. A path p is a sequence of relations, i.e., $r_1 \rightarrow ... \rightarrow r_j$, where j is the length of p. Link prediction can be viewed as a special case of path query answering when the

Model	WN18RR					FB15k-237				
	MR	MRR	Hits@1	Hits@3	Hits@10	MR	MRR	Hits@1	Hits@3	Hits@10
Models without non-commutativity										
TransE (2013)	3384	0.226	-	-	0.501	357	0.294	-	-	0.465
DistMult (2015)	5100	0.430	0.390	0.440	0.490	254	0.241	0.155	0.263	0.419
ComplEx (2016)	5261	0.440	0.410	0.460	0.510	339	0.247	0.158	0.275	0.428
MuRP (2019a)	-	0.475	0.436	0.487	0.554	-	0.336	0.245	0.370	0.521
RotatE (2019)	3340	0.476	0.428	0.492	0.571	177	0.338	0.241	0.375	0.533
BoxE (2020)	**3207**	0.451	0.400	0.472	0.541	**163**	0.337	0.238	0.374	0.538
Other models										
ConvE (2018)	4187	0.430	0.400	0.440	0.520	244	0.325	0.237	0.356	0.501
R-GCN+ (2018)	-	-	-	-	-	-	0.249	0.151	0.264	0.417
NKGE (2018)	4170	0.450	0.421	0.465	0.526	237	0.330	0.241	0.365	0.510
Models with non-commutativity										
DihEdral (2019)	-	0.486	0.442	0.505	0.557	-	0.320	0.230	0.353	0.502
QuatE (2019)	3472	0.481	0.436	0.500	0.564	176	0.311	0.221	0.342	0.495
Rotate3D (2020)	3328	0.489	0.442	0.505	**0.579**	165	**0.347**	0.250	**0.385**	**0.543**
DualE (2021)	-	0.482	0.440	0.500	0.561	-	0.330	0.237	0.363	0.518
RotateCT (ours)	3285	**0.492**	**0.448**	**0.507**	**0.579**	171	**0.347**	**0.251**	0.382	0.537

Table 2: Link prediction results on WN18RR and FB15k-237.

path length is 1. By contrast, path query answering requires more ability of muti-hop reasoning, in which inferring the composition patterns is crucial.

Datasets. We conduct experiments on two datasets released by Guu et al. (2015), which are extracted from WordNet (Miller, 1995) and Freebase (Bollacker et al., 2008). Both datasets contain triplets and paths. Paths are generated from triplets by random walks. The maximum of path length is 5. Paths used for training are only generated from training triplets, while paths used for test are generated from both training triplets and test triplets. Test paths which appear in training paths are removed. Note that paths of length 1 are not sampled, but created by directly adding triplets. Details of generating paths can be found in (Guu et al., 2015). See details of these two datasets in Appendix G.

Evaluation Protocol. We use the same evaluation protocol as in (Guu et al., 2015). Specifically, for each test path $p_t = s \to r_1 \to ... \to r_j \to o$, the corresponding query q is $s \to r_1 \to ... \to r_j \to ?$. Details of evaluation protocol for PQA can be found in Appendix H. We report the mean quantile (MQ) and Hits@10. MQ is the average quantile of all test paths. Hits@10 is the percentage of target answers whose rank is not larger than 10. An excellent model should achieve a higher MQ and a higher Hits@10.

Implementation Details. We train RotateCT with all paths in the training set, which is denoted as "Comp" in (Guu et al., 2015). Note that triplets are the paths of length 1. To make the results directly comparable, we follow Gao et al. (2020) to train RotateCT on paths of length 1 to 5 in turn, i.e., we train our model on paths of length i until convergence before training on paths of length $i+1$. Hyperparameters are determined via grid search according to the MQ on the validation set. In general, the embedding dimension k is selected in {500, 1000}; batch size n is searched in {512, 1024}; the negative sampling size m is picked from {256, 512}; the fixed margin γ is tuned among {6, 9, 12, 24}; self-adversarial temperature α is selected from {1.0, 2.0, 3.0}. The regularization rate λ is adjusted in {0, 0.1}. The initialization methods and experimental environment are the same as in link

Model	WordNet		Freebase	
	MQ	Hits@10	MQ	Hits@10
Implicitly model paths				
Bilinear[♯]	0.894	0.543	0.835	0.421
DistMult[♯]	0.904	0.311	0.848	0.386
TransE[♯]	0.933	0.435	0.880	0.505
RotatE[♯]	0.947	0.653	0.901	0.601
Rotate3D[♯]	**0.949**	0.671	0.905	0.621
RotateCT[♯] (ours)	**0.949**	0.673	0.907	0.630
Explicitly model paths				
ROP[†]	-	-	0.907	0.567
CoKE[‡]	0.942	**0.674**	**0.948**	**0.764**

Table 3: Path query answering results on WordNet and Freebase. [♯]: Models do not use extra structures to model paths; [†]: Model uses **RNN** to model paths; [‡]: Model uses **Transformer** to model paths. Best results are in **bold** and second best results are underlined.

Model	Rotate3D	RotateCT
Space	$\mathbb{H}^k$	$\mathbb{C}^k$
Dimension	1000	1000
FB15k	48.89M	33.94M(↓ 30.6%)
FB15k-237	44.33M	29.79M(↓ 32.8%)
WN18	122.89M	81.94M(↓ 33.3%)
WN18RR	122.86M	81.92M(↓ 33.3%)
Freebase	225.17M	150.13M(↓ 33.3%)
WordNet	115.69M	77.14M(↓ 33.3%)

Table 4: Number of free parameters.

	Rotate3D	RotateCT
WN18	158s	65s
WN18RR	159s	65s
FB15k	147s	56s
FB15k-237	153s	53s
WordNet	85s	57s
Freebase	103s	68s

Table 5: Training time per 1000 steps of Rotat3D and RotateCT on all datasets.

prediction. The best hyperparameters settings of RotateCT are provided in the Appendix E.

Main Results. Baselines are divided into two categories according to whether they use extra structures to model paths. The first category including Bilinear (Nickel et al., 2011), DistMult (Yang et al., 2015), TransE (Bordes et al., 2013), RotatE (Sun et al., 2019) and Rotate3D (Gao et al., 2020) has no extra structure to model paths; another category including ROP (Yin et al., 2018) and CoKE (Wang et al., 2019) uses RNN or Transformer to model paths.

Table 3 shows experimental results on the two datasets. Compared with methods that implicitly model paths, RotateCT achieves better performance on both WordNet and Freebase, which demonstrates the ability of RotateCT to model the composition patterns. Overall, RotateCT significantly outperforms Bilinear, DistMult, TransE and RotatE. The reason why RotateCT slightly surpasses Rotate3D is that both RotateCT and Rotate3D have the ability of inferring the non-commutative composition patterns. Compared with methods that explicitly model paths, RotateCT outperforms ROP and still obtains results comparable to CoKE on WordNet. Notably, CoKE uses Transformer as an extra structure to model paths, which is a remarkable architecture for natural language process tasks.

In addition to the overall MQ and Hits@10, we further report the results for different path lengths in Appendix K.

4.3 Analyses

Space Efficiency. To compare the space cost of Rotate3D and RotateCT, we caculate out the number of free parameters over different datasets. Results are shown in Table 4, from which we observe that RotateCT with same dimension reduces up to 30% parameters on all datasets. Intuitively, Rotate3D utilizes quaternions to model relations as rotations in 3D space. However, a quaternion has two more dimensions than a complex number, which sharply increases the space cost of Rotate3D.

Time Efficiency. To further compare the training cost of Rotate3D and RotateCT, we report the training time per 1000 steps on all datasets with embedding dimension $k = 1000$. From Table 5, we can find that RotateCT takes much less training time than Rotate3D on all datasets, which confirms the time efficiency superiority of RotateCT. Experiments of training time are performed on an Intel(R) Xeon(R) Silver 4114 CPU at 2.20GHz and a single NVIDIA Tesla V100 32GB GPU.

5 Case Studies

To verify that RotateCT can effectively model all the three types of relation patterns, we provide case studies via histograms on symmetry/antisymmetry and inversion patterns, and some intuitive examples on composition patterns.

5.1 Symmetry/Antisymmetry

Corollary 1 *Based on Theorem 2, if* r *is a symmetric relation, then* $\theta_i = 0 \vee \pm\pi$*; if* r *is an antisymmetric relation, then* $\theta_i \neq 0, \pm\pi$*. (See proof in Appendix I)*

According to the Corollary 1, the phase of each element in the embeddings of a symmetric relation r should be 0 or $\pm\pi$. Otherwise, the phases should not be 0 and $\pm\pi$. We investigate the phases of elements in relation embeddings from a RotateCT trained on WN18 and a RotateCT trained on FB15k-237 with their best hyperparameters in link prediction. The two models are trained with the setting $k = 1000$. Results are shown in Fig. 3. Specifically, Fig. 3(a)-3(b) give the histograms of the two symmetric relations in WN18: derivationally_related_form and verb_group, in which most phases of the two relations are either

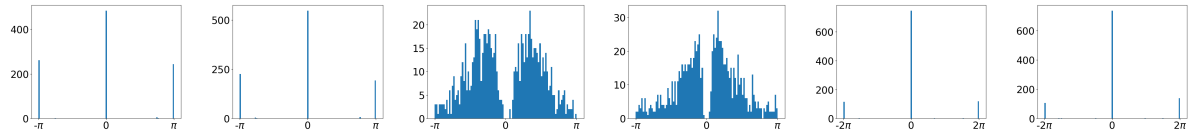

(a) (b) (c) (d) (e) (f)

Figure 3: Histograms of relation embeddings phases $\{\theta_{ri}\}(r_i = e^{i\theta_{ri}})$ of two symmetric relations: derivationally_related_form **(a)** and verb_group **(b)**. Histograms of relation embeddings phases $\{\theta_{ri}\}(r_i = e^{i\theta_{ri}})$ of two antisymmetric relations: film/actor/dubbing_performances./film/dubbing_performance/language **(c)** and /people/profession/specialization_of **(d)**. Histograms of the addition of relation embeddings phases $\{\theta_{1i} + \theta_{2i}\}$ of two pairs of inversion relations: has_part $\circ$ part_of **(e)** and hyponym $\circ$ hypernym **(f)**, where $\circ$ is the Hadmard (or element-wise) product.

0 or $\pm\pi$. The above observation confirms that RotateCT can effectively model the symmetry patterns. The histograms of two antisymmetric relations in FB15k-237 are shown in Fig. 3(c)-3(d), from which we can find that the phases of the two antisymmetric relations are scattered. Most phases of the two antisymmetric relations are neither 0 nor $\pm\pi$, which confirms that RotateCT can effectively model the antisymmetry patterns.

5.2 Inversion

Corollary 2 *Based on Theorem 2, if* r_1 *is the inverse of* r_2*, then* $\theta_{1i} + \theta_{2i} = 0 \vee \pm 2\pi$*. (See proof in Appendix J)*

According to the Corollary 2, if r_1 is inverse to r_2, the additive embedding phases, i.e., $\theta_{1i} + \theta_{2i}$, should be 0 or $\pm 2\pi$. Same RotateCT model trained on WN18 in Section 5.1 is used for investigating. Fig. 3(e)-3(f) show the element-wise addition of the embedding phases of two pairs of inverse relations: has_part and part_of, hyponym and hypernym. We can find that most additive embedding phases are either 0 or $\pm 2\pi$, which confirms that RotateCT can effectively model the inversion patterns.

5.3 Composition

To illustrate the superiority of RotateCT in modeling the composition patterns, we use a RotateCT and a RotatE, both trained with their best hyperparameters on the Freebase dataset in path query answering. Fig. 4 shows a subgraph extracted from the Freebase dataset. This subgraph includes two non-commutative composition patterns: "parents' spouse" and "spouse's parents", and one commutative composition pattern: "parents' parents".

Non-commutative Composition. From Fig. 5, we find that RotateCT predicts the query $Maria \rightarrow parents \rightarrow spouse$? and the query

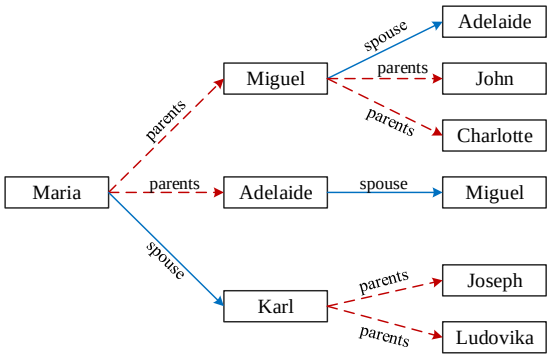

Figure 4: A subgraph about Maria's family. Maria's parents' spouse are Adelaide and Miguel; Maria's spouse's parents are Joseph and Ludovika; Maria's parents's parents are John and Charlotte. Dashed orange arrows represent the parents relation; solid blue arrows represent the spouse relation.

$Maria \rightarrow spouse \rightarrow parents$? correctly, which indicates that RotateCT can effectively model the non-commutative composition patterns. However, we observe that RotatE gives unsatisfactory answers in predicting the query $Maria \rightarrow parents \rightarrow$? and the query $Maria \rightarrow spouse \rightarrow parents \rightarrow$?, which verifies that RotatE lacks the ability of inferring the non-commutative composition patterns.

Commutative Composition. We further investigate the ability of inferring the commutative composition patterns of RotateCT and RotatE. As shown in Fig. 5, RotateCT predicts the query $Maria \rightarrow parents \rightarrow parents \rightarrow$? correctly, which confirms that RotateCT can effectively model the commutative composition patterns. By comparision, RotatE fails in predicting the correct answers, i.e., first item in results is not Maria's parents' parents. We argue that this is related to the inability of RotatE to model the non-commutative composition patterns, which has a negative impact

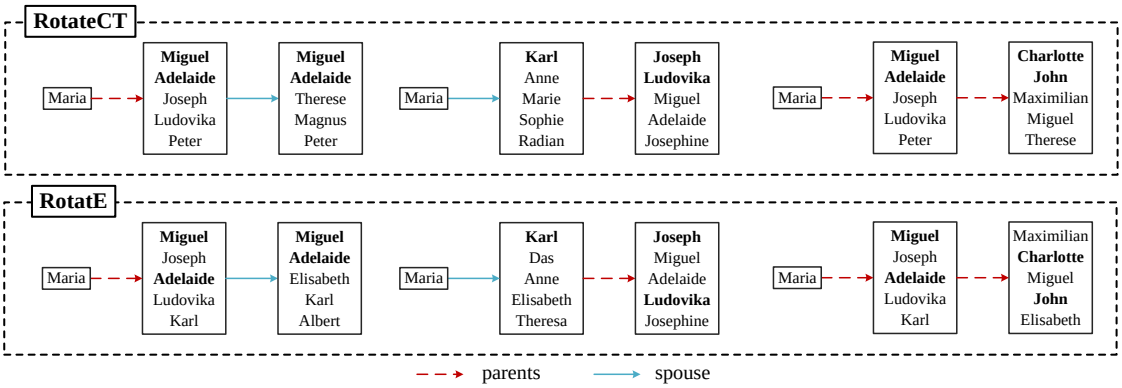

Figure 5: Top 5 answers of RotateCT and RotatE for three queries: "Who are Maria's parents' spouse?", "Who are Maria's spouse's parents?" and "Who are Maria's parents' parents?". Correct answers are in **bold**.

on embeddings' learning. If RotatE encodes "parents' spouse" and "spouse's parents" in the same way, some semantic information will be lost. This flaw is harmful for RotatE effectively learning the semantics of parents and spouse, which reduces RotatE's performance on modeling the commutative composition pattern "parents' parents".

6 Conclusion

In this paper, we propose a novel knowledge graph embedding method called RotateCT to model the non-commutative composition patterns while improve the parameter efficiency against quaternion-valued methods. RotateCT transforms the coordinates of each entity by translating the origin of coordinates. Further, relations are represented as rotations from head entities to tail entities in complex space. Combining rotation and coordinate transformation empowers RotateCT to model not only commutative composition patterns but also non-commutative composition patterns. As a complex-valued method, the space cost of RotateCT is lower than quaternion-valued methods. Experimental results on link prediction and path query answering show that RotateCT achieves significant performance and demonstrate the superiority of RotateCT for inferring composition patterns. The parameter efficiency analysis proves that RotateCT can reduce the space cost.

In future work, we will explore different ways to model the non-commutative composition patterns and plan to study the combination of quaternion and coordinate transformation.

Acknowledgements

This work was supported by the National Key Research and Development Program of China.

References

Ralph Abboud, Ismail Ceylan, Thomas Lukasiewicz, and Tommaso Salvatori. 2020. BoxE: A box embedding model for knowledge base completion. In *NeurIPS*, volume 33.

Ivana Balazevic, Carl Allen, and Timothy Hospedales. 2019a. Multi-relational poincaré graph embeddings. *NeurIPS*, 32:4463–4473.

Ivana Balazevic, Carl Allen, and Timothy Hospedales. 2019b. TuckER: Tensor factorization for knowledge graph completion. In *EMNLP*, pages 5185–5194.

Kurt Bollacker, Colin Evans, Praveen Paritosh, Tim Sturge, and Jamie Taylor. 2008. Freebase: a collaboratively created graph database for structuring human knowledge. In *SIGMOD*, pages 1247–1250.

Antoine Bordes, Nicolas Usunier, Alberto Garcia-Duran, Jason Weston, and Oksana Yakhnenko. 2013. Translating embeddings for modeling multi-relational data. In *NeurIPS*, volume 26, pages 2787–2795.

Zongsheng Cao, Qianqian Xu, Zhiyong Yang, Xiaochun Cao, and Qingming Huang. 2021. Dual quaternion knowledge graph embeddings. In *AAAI*, volume 35, pages 6894–6902.

Rajarshi Das, Arvind Neelakantan, David Belanger, and Andrew McCallum. 2017. Chains of reasoning over entities, relations, and text using recurrent neural networks. In *EACL*, pages 132–141.

Tim Dettmers, Pasquale Minervini, Pontus Stenetorp, and Sebastian Riedel. 2018. Convolutional 2d knowledge graph embeddings. In *AAAI*, volume 32.

Takuma Ebisu and Ryutaro Ichise. 2018. Toruse: Knowledge graph embedding on a lie group. In *AAAI*, volume 32.

Chang Gao, Chengjie Sun, Lili Shan, Lei Lin, and Mingjiang Wang. 2020. Rotate3d: Representing relations as rotations in three-dimensional space for knowledge graph embedding. In *CIKM*, pages 385–394.

Kelvin Guu, John Miller, and Percy Liang. 2015. Traversing knowledge graphs in vector space. In *EMNLP*, pages 318–327.

Yanchao Hao, Yuanzhe Zhang, Kang Liu, Shizhu He, Zhanyi Liu, Hua Wu, and Jun Zhao. 2017. An end-to-end model for question answering over knowledge base with cross-attention combining global knowledge. In *ACL*, pages 221–231.

Guoliang Ji, Shizhu He, Liheng Xu, Kang Liu, and Jun Zhao. 2015. Knowledge graph embedding via dynamic mapping matrix. In *ACL*, pages 687–696.

Seyed Mehran Kazemi and David Poole. 2018. Simple embedding for link prediction in knowledge graphs. In *NeurIPS*, pages 4284–4295.

Diederik P Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*.

Yankai Lin, Zhiyuan Liu, Maosong Sun, Yang Liu, and Xuan Zhu. 2015. Learning entity and relation embeddings for knowledge graph completion. In *AAAI*, volume 29.

Xin Lv, Lei Hou, Juanzi Li, and Zhiyuan Liu. 2018. Differentiating concepts and instances for knowledge graph embedding. In *EMNLP*, pages 1971–1979.

Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. 2013. Distributed representations of words and phrases and their compositionality. In *NeurIPS*, pages 3111–3119.

George A Miller. 1995. Wordnet: a lexical database for english. *Communications of the ACM*, 38(11):39–41.

Tom Mitchell, William Cohen, Estevam Hruschka, Partha Talukdar, Bishan Yang, Justin Betteridge, Andrew Carlson, Bhavana Dalvi, Matt Gardner, Bryan Kisiel, et al. 2018. Never-ending learning. *Communications of the ACM*, 61(5):103–115.

Deepak Nathani, Jatin Chauhan, Charu Sharma, and Manohar Kaul. 2019. Learning attention-based embeddings for relation prediction in knowledge graphs. In *ACL*, pages 4710–4723.

Tu Dinh Nguyen, Dat Quoc Nguyen, Dinh Phung, et al. 2018. A novel embedding model for knowledge base completion based on convolutional neural network. In *NAACL*, pages 327–333.

Maximilian Nickel, Lorenzo Rosasco, and Tomaso Poggio. 2016. Holographic embeddings of knowledge graphs. In *AAAI*, volume 30.

Maximilian Nickel, Volker Tresp, and Hans-Peter Kriegel. 2011. A three-way model for collective learning on multi-relational data. In *ICML*, volume 11, pages 809–816.

Michael Schlichtkrull, Thomas N Kipf, Peter Bloem, Rianne Van Den Berg, Ivan Titov, and Max Welling. 2018. Modeling relational data with graph convolutional networks. In *ESWC*, pages 593–607. Springer.

Fabian M Suchanek, Gjergji Kasneci, and Gerhard Weikum. 2007. Yago: a core of semantic knowledge. In *WWW*, pages 697–706.

Zhiqing Sun, Zhi-Hong Deng, Jian-Yun Nie, and Jian Tang. 2019. Rotate: Knowledge graph embedding by relational rotation in complex space. In *ICLR*.

Kristina Toutanova and Danqi Chen. 2015. Observed versus latent features for knowledge base and text inference. In *Proceedings of the 3rd Workshop on Continuous Vector Space Models and their Compositionality*, pages 57–66.

Théo Trouillon, Johannes Welbl, Sebastian Riedel, Éric Gaussier, and Guillaume Bouchard. 2016. Complex embeddings for simple link prediction. In *ICML*.

Shikhar Vashishth, Rishabh Joshi, Sai Suman Prayaga, Chiranjib Bhattacharyya, and Partha Talukdar. 2018. Reside: Improving distantly-supervised neural relation extraction using side information. In *EMNLP*, pages 1257–1266.

Shikhar Vashishth, Soumya Sanyal, Vikram Nitin, Nilesh Agrawal, and Partha P Talukdar. 2020. Interacte: Improving convolution-based knowledge graph embeddings by increasing feature interactions. In *AAAI*, pages 3009–3016.

Kai Wang, Yu Liu, Xiujuan Xu, and Dan Lin. 2018. Knowledge graph embedding with entity neighbors and deep memory network. *arXiv preprint arXiv:1808.03752*.

Quan Wang, Pingping Huang, Haifeng Wang, Songtai Dai, Wenbin Jiang, Jing Liu, Yajuan Lyu, Yong Zhu, and Hua Wu. 2019. Coke: Contextualized knowledge graph embedding. *arXiv preprint arXiv:1911.02168*.

Zhen Wang, Jianwen Zhang, Jianlin Feng, and Zheng Chen. 2014. Knowledge graph embedding by translating on hyperplanes. In *AAAI*, volume 28.

Han Xiao, Minlie Huang, and Xiaoyan Zhu. 2016. From one point to a manifold: knowledge graph embedding for precise link prediction. In *IJCAI*, pages 1315–1321.

Canran Xu and Ruijiang Li. 2019. Relation embedding with dihedral group in knowledge graph. In *ACL*, pages 263–272.

Bishan Yang, Wen tau Yih, Xiaodong He, Jianfeng Gao, and Li Deng. 2015. Embedding entities and relations for learning and inference in knowledge bases.

Wenpeng Yin, Yadollah Yaghoobzadeh, and Hinrich Schütze. 2018. Recurrent one-hop predictions for reasoning over knowledge graphs. In *COLING*, pages 2369–2378.

Fuzheng Zhang, Nicholas Jing Yuan, Defu Lian, Xing Xie, and Wei-Ying Ma. 2016. Collaborative knowledge base embedding for recommender systems. In *SIGKDD*, pages 353–362.

Shuai Zhang, Yi Tay, Lina Yao, and Qi Liu. 2019. Quaternion knowledge graph embeddings. In *NeurIPS*, pages 2735–2745.

Yang Zhao, Jiajun Zhang, Yu Zhou, and Chengqing Zong. 2020. Knowledge graphs enhanced neural machine translation. In *IJCAI*, pages 4039–4045.

Hao Zhou, Tom Young, Minlie Huang, Haizhou Zhao, Jingfang Xu, and Xiaoyan Zhu. 2018. Commonsense knowledge aware conversation generation with graph attention. In *IJCAI*, pages 4623–4629.

A Definitions of Inference Patterns

We give the formal definitions of inference patterns as follows:

Definition 1 *A relation* r *is* ***symmetric (antisymmetric)*** *if*

$$\forall \mathrm{x}, \mathrm{y} \in \mathcal{E}, \mathrm{r}(\mathrm{x}, \mathrm{y}) \Rightarrow \mathrm{r}(\mathrm{y}, \mathrm{x}) \ (\, \mathrm{r}(\mathrm{x}, \mathrm{y}) \Rightarrow \neg \mathrm{r}(\mathrm{y}, \mathrm{x}) \,)$$

A relation with such form is a ***symmetry (antisymmetry)*** *pattern.*

Definition 2 *Relation* r_1 *is* ***inverse*** *to* r_2 *if*

$$\forall \mathrm{x}, \mathrm{y} \in \mathcal{E}, \mathrm{r}_2(\mathrm{x}, \mathrm{y}) \Rightarrow \mathrm{r}_1(\mathrm{y}, \mathrm{x})$$

Relations with such form is an ***inversion*** *pattern.*

Definition 3 *Realtion* r_1 *is* ***composed*** *of relation* r_2 *and relation* r_3 *if*

$$\forall \mathrm{x}, \mathrm{y}, \mathrm{z} \in \mathcal{E}, \mathrm{r}_2(\mathrm{x}, \mathrm{y}) \wedge \mathrm{r}_3(\mathrm{y}, \mathrm{z}) \Rightarrow \mathrm{r}_1(\mathrm{x}, \mathrm{z})$$

r_2 *and* r_3 *are commutative if*

$$\forall \mathrm{x}, \mathrm{y} \in \mathcal{E}, \mathrm{r}_2 \odot \mathrm{r}_3(\mathrm{x}, \mathrm{y}) \Rightarrow \mathrm{r}_3 \odot \mathrm{r}_2(\mathrm{x}, \mathrm{y})$$

r_2 *and* r_3 *are non-commutative if*

$$\forall \mathrm{x}, \mathrm{y} \in \mathcal{E}, \mathrm{r}_2 \odot \mathrm{r}_3(\mathrm{x}, \mathrm{y}) \nRightarrow \mathrm{r}_3 \odot \mathrm{r}_2(\mathrm{x}, \mathrm{y})$$

where $\odot$ *is the composition operator.*

Relations with such form is a ***composition*** *pattern.*

B Proof of Theorem 1

Proof 1 *Given* $\mathrm{r}_1, \mathrm{r}_2 \in \mathcal{R}$ *and* $\mathrm{x} \in \mathcal{E}$*, we have*

$$\begin{aligned} r_{1i} \odot r_{2i}(x_i) =& ((x_i - b_{1i})r_{1i} + b_{1i} - b_{2i})r_{2i} \\ =& x_i r_{1i} r_{2i} - b_{1i} r_{1i} r_{2i} + \\ & b_{1i} r_{2i} - b_{2i} r_{2i}, \qquad (1) \\ r_{2i} \odot r_{1i}(x_i) =& ((x_i - b_{2i})r_{2i} + b_{2i} - b_{1i})r_{1i} \\ =& x_i r_{1i} r_{2i} - b_{2i} r_{1i} r_{2i} + \\ & b_{2i} r_{1i} - b_{1i} r_{1i}. \qquad (2) \end{aligned}$$

Apparently, Equation (1) is not equal to (2), which means that the combination of operations in RotateCT is non-commutative. If and only if $b_{1i} = b_{2i}$ *or* $r_{1i} r_{2i} = r_{1i} + r_{2i}$*,* $r_{1i} \odot r_{2i}(x_i) = r_{2i} \odot r_{1i}(x_i)$*.*

C Proof of Theorem 2

Proof 2 *For the* ***symmetry/antisymmetry*** *pattern, given* $\mathrm{r} \in \mathcal{R}$ *and* $\mathrm{x}, \mathrm{y} \in \mathcal{E}$*, if* $\mathrm{r}(\mathrm{x}, \mathrm{y})$ *and* $\mathrm{r}(\mathrm{y}, \mathrm{x})$ *hold, we have*

$$\begin{cases} y_i - b_i = (x_i - b_i) r_i \\ x_i - b_i = (y_i - b_i) r_i \end{cases} \Rightarrow \quad r_i = \pm 1$$

Otherwise, if $\mathrm{r}(\mathrm{x}, \mathrm{y})$ *and* $\neg \mathrm{r}(\mathrm{y}, \mathrm{x})$ *hold, we have*

$$\begin{cases} y_i - b_i = (x_i - b_i) r_i \\ x_i - b_i \neq (y_i - b_i) r_i \end{cases} \Rightarrow \quad r_i \neq \pm 1$$

For the ***inversion*** *pattern, given* $\mathrm{r}_1, \mathrm{r}_2 \in \mathcal{R}$ *and* $\mathrm{x}, \mathrm{y} \in \mathcal{E}$*, if* $\mathrm{r}_1(\mathrm{y}, \mathrm{x})$ *and* $\mathrm{r}_2(\mathrm{x}, \mathrm{y})$ *hold, we have*

$$\begin{cases} x_i - b_{1i} = (y_i - b_{1i}) r_{1i} \\ y_i - b_{2i} = (x_i - b_{2i}) r_{2i} \end{cases}$$

$$\Rightarrow \quad \begin{cases} r_{1i} r_{2i} = 1 \\ b_{1i} = b_{2i} \end{cases} \quad or \quad r_{1i} = r_{2i} = 1$$

For the ***composition*** *pattern, given* $\mathrm{r}_1, \mathrm{r}_2, \mathrm{r}_3 \in \mathcal{R}$ *and* $\mathrm{x}, \mathrm{y}, \mathrm{z} \in \mathcal{E}$*, if* $\mathrm{r}_1(\mathrm{x}, \mathrm{z})$*,* $\mathrm{r}_2(\mathrm{x}, \mathrm{y})$ *and* $\mathrm{r}_3(\mathrm{y}, \mathrm{z})$ *hold, we have*

$$\begin{cases} z_i - b_{1i} = (x_i - b_{1i}) r_{1i} \\ y_i - b_{2i} = (x_i - b_{2i}) r_{2i} \\ z_i - b_{3i} = (x_i - b_{3i}) r_{3i} \end{cases}$$

$$\Rightarrow \begin{cases} r_{1i} r_{2i} = r_{3i} \\ b_{1i} r_{1i} r_{2i} = b_{1i} r_{2i} - b_{2i} r_{2i} + b_{3i} r_{3i} \end{cases}$$

if r_2 *and* r_3 *are commutative, then*

$$b_{2i} = b_{3i} \quad or \quad r_{2i} r_{3i} = r_{2i} + r_{3i},$$

if r_2 *and* r_3 *are non-commutative, then*

$$\begin{cases} b_{2i} \neq b_{3i} \\ r_{2i} r_{3i} \neq r_{2i} + r_{3i} \end{cases}$$

In conclusion, RotateCT can infer ***symmetry/antisymmetry****,* ***inversion*** *and* ***composition*** *patterns.*

D Illustrations of RotateCT

Fig. 6 shows illustrations of RocateCT modeling symmetry and non-commutative composition patterns.

Dataset	embedding dimension	batch size	self-adversarial temperature	margin	negative sample size	distance	regularization/rate	learning rate
WN18	1000	512	0.5	12	256	L1	L1/0.1	1×10^{-4}
WN18RR	1000	1024	1.0	6	512	L1	L1/0.1	5×10^{-5}
FB15k	1000	1024	0.5	24	256	L2	-	2×10^{-4}
FB15k-237	1000	1024	1.0	12	256	L2	-	2×10^{-4}
WordNet	1000	512	1.0	6	256	L2	L2/0.1	5×10^{-5}
Freebase	1000	1024	2.0	12	512	L2	-	2×10^{-5}

Table 6: Hyperparameters setting of RotateCT over different datasets.

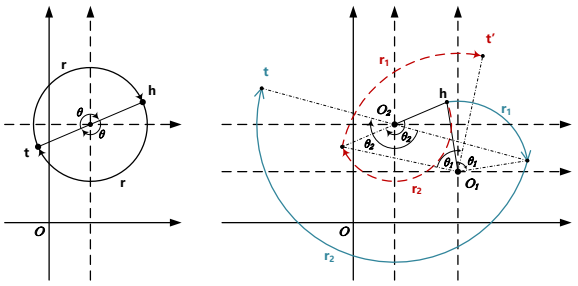

(a) Symmetry (b) Non-commutative composition

Figure 6: Illustrations of RotateCT over two relation patterns. Fig. 6(a) shows how RotateCT models the symmetry pattern, i.e., $r = \pm 1$. In 6(b), blue solid lines represent the path $h \rightarrow r_1 \rightarrow r_2 \rightarrow t$ and red dashed lines represent the path $h \rightarrow r_2 \rightarrow r_1 \rightarrow t'$. r_1 and r_2 form a non-commutative composition pattern. When the relative order of r_1 and r_2 changes, RotateCT can get two different entities: t and t'.

E Hyperparameters Settings

We list the best hyperparameters setting of RotateCT on all datasets in Table 6.

F Details of Link Prediction Datasets

WN18 is a subset of WordNet (Miller, 1995), a database consisting of lexical relations between words. FB15k is extracted from Freebase (Bollacker et al., 2008), a large-scale knowledge graph containing general facts. The main relation patterns in WN18 and FB15k are **symmetry/antisymmetry** and **inversion** (Sun et al., 2019). However, both WN18 and FB15k suffer from test leakage through inverse relations (Toutanova and Chen, 2015). To avoid this problem, WN18RR and FB15k-237 remove the inverse relations in WN18 and FB15k, respectively. Therefore, the main relation patterns in WN18RR and FB15k-237 are **symmetry/antisymmetry** and **composition** (Sun et al., 2019). Notably, semantic information in WN18RR and FB15k-237 is more difficult to capture on account of removing the inverse relations. The statistics of the benchmarks for link prediction are listed in Table 7.

Dataset	#entity	#relation	#training	#validation	#test
FB15k	14,951	1,345	483,142	50,000	59,071
WN18	40,943	18	141,442	5,000	5,000
FB15k-237	14,541	237	272,115	17,535	20,466
WN18RR	40,943	11	86,835	3,034	3,134

Table 7: Number of entities, relations, and triplets in each split for four benchmarks.

G Details of Path Query Answering Datasets

The statistics of path query answering datasets are summarized in Table 8.

	WordNet	Freebase
#Entities	38,551	75,043
#Relations	11	13
#Train Triplets	110,361	316,232
#Valid Triplets	2,602	5,908
#Test Triplets	10,462	23,733
#Train Paths	2,129,539	6,266,058
#Valid Paths	11,277	27,163
#Test Paths	46,577	109,577

Table 8: Number of entities, relations, triples and paths in each split of the two datasets.

H Evaluation Protocol of PQA

For each test path $p_t = s \rightarrow r_1 \rightarrow ... \rightarrow r_j \rightarrow o$, the corresponding query q is $s \rightarrow r_1 \rightarrow ... \rightarrow r_j \rightarrow ?$. For each query q, the candidate answers are entities that "type-match", i.e., all tail entities of the final relation r_j. The correct answers are entities that can be reached from s by traversing the path p; the incorrect answers are obtained by filtering out the correct answers from the candidate answers. The set of candidate answers to a query q denotes as $\mathcal{C}(q)$; the set of correct answers to a query q denotes as $\mathcal{P}(q)$; the set of incorrect

answers to a query q denotes as $\mathcal{N}(q)$. We give the formal definition of $\mathcal{C}(q)$, $\mathcal{P}(q)$, and $\mathcal{N}(q)$ as follows:

$$\begin{aligned}\mathcal{C}(q) &\triangleq \{o|\exists e \text{ s.t. } (e, r_j, o) \in \mathcal{G}\}\\ \mathcal{P}(q) &\triangleq \{o|\exists e_1, ..., e_{j-1} \text{ s.t. } (s, r_1, e_1),\\ &\qquad ..., (e_{j-1}, r_j, o) \in \mathcal{G}\}\\ \mathcal{N}(q) &\triangleq \mathcal{C}(q) \backslash \mathcal{P}(q)\end{aligned}$$

Here $\mathcal{G}$ includes training triplets and test triplets. For each test path p_t, we replace entity o with entities in $\mathcal{C}(q)$ and compute the score of each candidate answer. Then we rank the scores of candidates along with the score of p_t in descending order and caculate the quantile, which is the proportion of incorrect answers ranked after the target answer o.

Notably, some test paths in both datasets are "type-match trivial", i.e., all type matching candidate answers are correct. Hence, the quantile of these test paths are undefined, and we exclude them from evaluation.

I Proof of Corollary 1

Proof 3 *As shown in Theorem 2, if* r *is a symmetric relation, we have*

$$\begin{aligned}r_i = \pm 1 \;&\Rightarrow\; cos\theta_i + isin\theta_i = \pm 1\\ &\Rightarrow\; cos\theta_i = \pm 1\\ &\Rightarrow\; \theta_i = 0 \vee \pm\pi\end{aligned}$$

Otherwise, if r *is an antisymmetric relation, we have*

$$\begin{aligned}r_i \neq \pm 1 \;&\Rightarrow\; cos\theta_i + isin\theta_i \neq \pm 1\\ &\Rightarrow\; cos\theta_i \neq \pm 1\\ &\Rightarrow\; \theta_i \neq 0, \pm\pi\end{aligned}$$

J Proof of Corollary 2

Proof 4 *As shown in Theorem 2, if* r_1 *is the inverse of* r_2*, we have*

$$\begin{cases} r_{1i} r_{2i} = 1 \\ b_{1i} = b_{2i} \end{cases} \quad or \quad r_{1i} = r_{2i} = 1$$

In the first case, we have

$$\begin{cases} r_{1i} r_{2i} = 1 \\ b_{1i} = b_{2i} \end{cases}$$

$$\Rightarrow \quad \begin{cases} (cos\theta_{1i} + isin\theta_{1i})(cos\theta_{2i} + isin\theta_{2i}) = 1 \\ b_{1i} = b_{2i} \end{cases}$$

$$\Rightarrow \quad \begin{cases} cos(\theta_{1i} + \theta_{2i}) + isin(\theta_{1i} + \theta_{2i}) = 1 \\ b_{1i} = b_{2i} \end{cases}$$

$$\Rightarrow \quad \begin{cases} \theta_{1i} + \theta_{2i} = 0 \vee \pm 2\pi \\ b_{1i} = b_{2i} \end{cases}$$

In the second case, we have

$$\begin{aligned}r_{1i} = r_{2i} = 1 \;&\Rightarrow\; cos\theta_{1i} + isin\theta_{1i} =\\ &\qquad cos\theta_{2i} + isin\theta_{2i} = 1\\ &\Rightarrow\; \theta_{1i} = \theta_{2i} = 0\end{aligned}$$

Combining the above two cases, we have

$$\theta_{1i} + \theta_{2i} = 0 \vee \pm 2\pi$$

K PQA Results of Different Path Lengths

We further report the path query answering results for different path lengths. As shown in Table 9 and Table 10, RotateCT outperforms RotatE and Rotate3D over most metrics, which verifies the superior capability of RotateCT to deal with the multi-hop reasoning.

	length=1		length=2		length=3		length=4		length=5	
Model	MQ	Hits@10	MQ	Hits@10	MQ	Hits@10	MQ	Hits@10	MQ	Hits@10
RotatE	**0.934**	0.794	0.882	0.403	0.922	0.649	0.858	0.415	0.910	0.652
Rotate3D	0.933	0.796	**0.901**	**0.473**	0.921	0.685	0.854	0.449	0.908	0.654
RotateCT	**0.934**	**0.803**	0.885	0.460	**0.928**	**0.707**	**0.863**	**0.462**	**0.920**	**0.689**

Table 9: Path query answering results of each path length on Freebase. Best results are in **bold**.

	length=1		length=2		length=3		length=4		length=5	
Model	MQ	Hits@10	MQ	Hits@10	MQ	Hits@10	MQ	Hits@10	MQ	Hits@10
RotatE	0.868	0.350	0.970	0.786	0.973	0.778	0.970	0.737	0.967	0.693
Rotate3D	**0.872**	**0.365**	**0.972**	0.797	0.974	0.786	**0.973**	0.750	0.969	0.706
RotateCT	0.865	0.338	**0.972**	**0.807**	**0.975**	**0.796**	**0.973**	**0.763**	**0.971**	**0.731**

Table 10: Path query answering results of each path length on WordNet. Best results are in **bold**.

Can Data Diversity Enhance Learning Generalization?

Yu Yu[*] and **Shahram Khadivi**[[]] and **Jia Xu**[*]
[*] School of Engineering and Science, Steven Institute of Technology, NJ 07030, USA
[[]] eBay Inc., Aachen 52064, Germany
yyu50@stevens.edu, skhadivi@ebay.com, jxu70@stevens.edu

Abstract

This paper introduces our Diversity Advanced Actor-Critic reinforcement learning (A2C) framework (DAAC) to improve the generalization and accuracy of Natural Language Processing (NLP). We show that the diversification of training samples alleviates overfitting and improves model generalization and accuracy. We quantify diversity on a set of samples using the max dispersion, convex hull volume, and graph entropy based on sentence embeddings in high-dimensional metric space. We also introduce A2C to select such a diversified training subset efficiently. Our experiments achieve up to +23.8 accuracy increase (38.0% relatively) in sentiment analysis, -44.7 perplexity decrease (37.9% relatively) in language modeling, and consistent improvements in named entity recognition over various domains. In particular, our method outperforms both domain adaptation and generalization baselines without using any target domain knowledge.

1 Introduction

We introduce the Diversity Advanced Actor-Critic reinforcement learning framework (**DAAC**) to improve the generalization and accuracy of Natural Language Processing (NLP). Training data plays a crucial role in Deep Learning (DL)-based NLP. Investigations (Hendrycks et al., 2020; Dodge et al., 2020; Ramponi and Plank, 2020) show that training data quality is imperative to the machine learning model's performance. Good data are effective and easy to generalize, while bad data bring noise and overfits on out-of-domain test sets.

Decades of work (van der Wees et al., 2017; Aharoni and Goldberg, 2020; Guo et al., 2020; Axelrod et al., 2011) have been devoted to finding the best data, with the most commonly used method we call domain adaptation (Qu et al., 2019; Liu et al., 2019) that uses target domain knowledge. Despite large improvements on specific domains, these adapted models lack robustness and are prone to rare events. For example, when models shift towards one target domain like Twitter, they may become erroneous for other domains like medical or travel, and real-world applications on news report data, for example, do not know which queries to receive before they launch, resulting in a critical performance gap between laboratory findings and reality. Therefore, we move away from the independent and identically distributed assumption (i.i.d.) and focus on reducing generalization errors by studying the inter-dependencies among samples.

In this paper, we ask three questions (RQs) and propose our answers. The first question (**RQ 1**) is: *"does higher diversity among training samples lead to better generalized NLP model learning?"*. Semantic diversity is a desirable aspect in human annotation, where training data sets should be large and diverse enough to learn the many ways the objects differ (Shankar et al., 2017; Wu et al., 2018; Merler et al., 2019; Schumann et al., 2021), like intrinsic facial diversity in face recognition data, since every face is different. We show that diversity ensures that the training data can provide sufficiently discriminative information for the model (Gong et al., 2019), and thus give a more accurate prediction. Intuitively, for a fixed number of samples, the more semantic meaning they cover, the more information they contain, and thus the more effective they are for learning. We will use two examples to elaborate on our rationale.

Let us first take one explanatory example of V ={be, cheerful, happy, stay}, a four single-word dataset that forms two semantically close clusters: A ={be, stay} and B ={happy, cheerful}. Our hypothetical NLP task is to generate sentences from V. From $\binom{4}{2}$ word combinations, four sentences are meaningful: "stay happy", "stay cheerful", "be happy", "be cheerful". Our dataset selection task is to find two single-word samples from V that generate the most sentences. If we select both single-word samples from A, which are semanti-

Proceedings of the 29th International Conference on Computational Linguistics, pages 4933–4945
October 12–17, 2022.

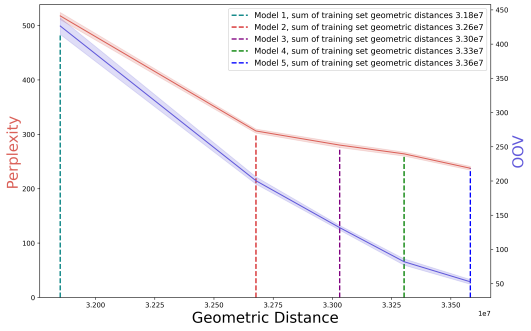

Figure 1: Higher training set diversity (geometric sum of pairwise Cosine distance), better learning generalization, w.r.t. PPL/OOV on a random test set.

cally close, we will generate no meaningful sentence (analogously that from B). However, we can generate one of the four meaningful sentences if we select one word from A and the other word from B, e.g., "stay" and "happy", which are semantically distant. In fact, since these four sentences can be treated as paraphrasing, generating any of them will roughly cover the semantic meaning for all. In this example, the diversified subset $\{stay, happy\}$ is more effective for learning generalization.

Our second example is conducted on a real dataset. Figure 1 shows our experimental results on the Penn Treebank dataset. We equipartition 42K training sentences into five subsets and compute their diversity using the sum of pairwise Cosine distances of sentence embeddings (i.e., our *first method*, see next paragraph and Section 2.2.1). We sort the subsets according to their diversity scores. The first subset has the lowest diversity ($3.18e^7$), the second subset has the second-lowest diversity ($3.26e^7$), etc. We then train five language models on each subset[1], and computes the perplexity (PPL) and out-of-vocabulary (OOV) on ten randomly sampled subsets of the test set. PPL and OOV are typical indicators of how training data is generalized (Rastogi et al., 2020; Müller et al., 2019). Our results show that the diversity of the subset strongly negatively correlates to both PPL and OOV rate, where the subset with the highest diversity is more likely to generalize on a random test and has fewer unseen words.

With the preliminary evidence showing that a training set with higher diversity leads to more generalized learning, we propose our second question (**RQ 2**): *how to measure the diversity of a sample set?* We introduce three methods to quantify training dataset diversity. The *first method* is based on the notion of dispersion, also known as Max-Sum Diversification (Cevallos et al., 2016). By viewing each training sample as a data point in the sentence embedding space, we maximize the dispersion of the training set, which is the sum of pairwise Cosine distances of sentence embeddings, as in Figure 2 (2). The larger the sum, the more informative the subset is.

The pairwise geometric distance method has a worst-case complexity of $\mathcal{O}(n^2)$, and to accelerate the computation, we introduce our *second method* (Figure 2 (3)) that measures diversity using the volume of a convex hull around the samples, namely the volume that encloses all vertices, taking each sentence embedding as a vertex. Its quickhull (Barber et al., 1996) implementation has an average-case complexity of $\mathcal{O}(n \log n)$ and a worst-case complexity of $\mathcal{O}(n^2)$.

The above two methods compute diversity based on distances in the metric space. We can obtain global information content by measuring the uncertainty of the semantic distances between samples based on frequencies. Our *third method* combines our view of geometric distance and entropy in NLP. As depicted in Figure 2 (4), we apply Graph entropy, $H(G, P)$, an information-theoretic function on graph G with a probability distribution P on its vertex set (Dehmer and Mowshowitz, 2011; Rezaei, 2013). Here, each sentence in the subset is a node in the graph, and the Cosine distance of the two-sentence embeddings is the weighted edge between the two nodes. Empirically, all these three diversity measures consistently outperform the baselines. The Max Dispersion (MD) and Graph Entropy (GE) are more robust in various domains with the best performance overall because they account for every pair of distances in the training set.

After we have defined our diversity measure, we ask our third question (**RQ 3**): *"how can we optimize the subset to maximize the information content?"* Subset selection is NP-hard in general (Qian et al., 2016; Davis et al., 1997). We conjecture that this holds true for every objective function, including the notion of diversity introduced in this paper, and is the reason why we use **actor-critic** reinforcement learning (Konda and Tsitsiklis, 2000). We equipartition the training data into mini-batches

[1]The language model is trained on Transformer with 2 heads, 2 hidden layers, 200 embedding size, and 200 hidden units on each subset for 10 epochs. Each test set subset contains 1000 sentences.

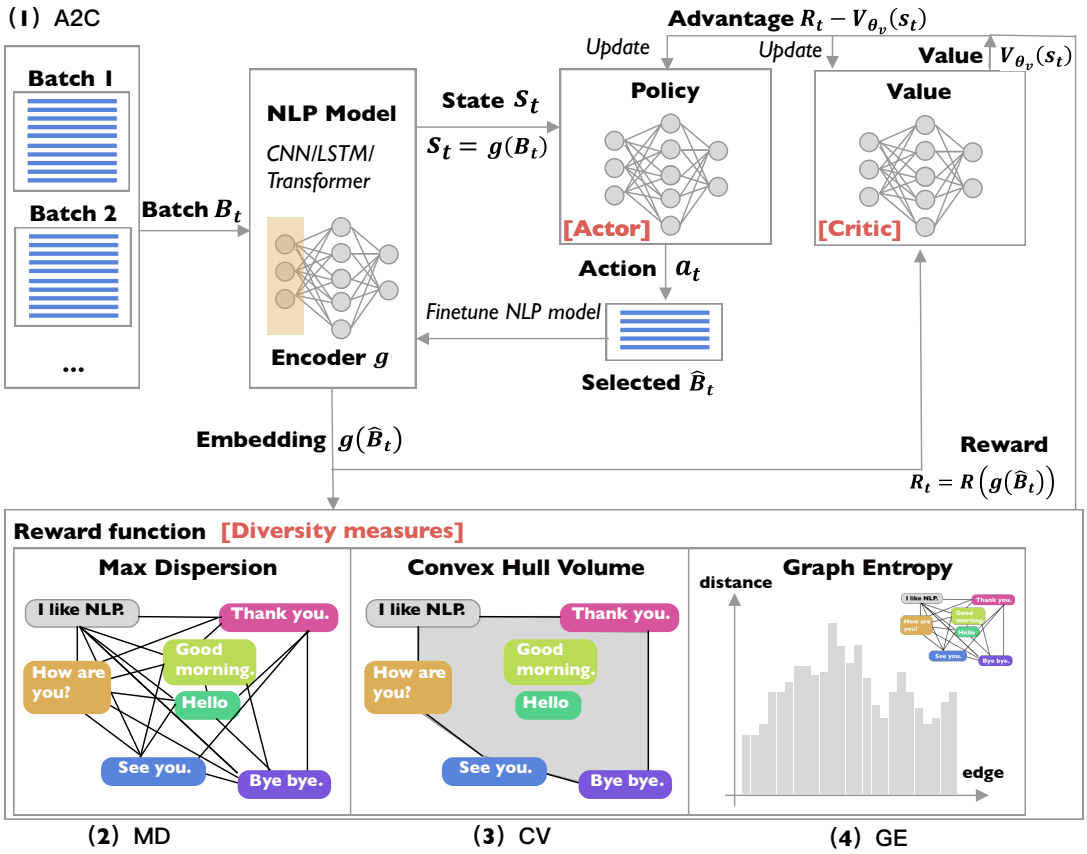

Figure 2: Diversity Advanced Actor-Critic Framework.

and simultaneously learn a policy network (as an actor) to select data and a value network (as a critic) to estimate future returns using the diversity measure as an evaluation reward. Our actor-critic data selection method has the advantages of low variance updates as well as credit assignment and significantly outperforms domain adaptation (Liu et al., 2019) on various domains without requiring any target domain knowledge. The architecture is shown in Figure 2 and Section 2.

In summary, our work mainly contributes to the following: (1) Modeling sample dependency for effective data sampling using diversity; (2) Measure dataset diversity using max dispersion, convex hull volume, and graph entropy; (3) Introduction of the Advantage Actor-Critic reinforcement learning framework to select informative samples and enhance NLP accuracy and generalization.

The rest of the paper is organized as follows. In Section 2.1, we detail Diversity Advanced Actor-Critic reinforcement learning (DAAC). Then in Section 2.2, we introduce different diversity measures. Afterward, in Section 3, we empirically verify the generalization and accuracy improvement using DAAC. We discuss related work in Section 4 and conclude the paper in the last section.

2 Diversity Advanced Actor-Critic

We now present the details of our Diversity Advanced Actor-Critic (DAAC) algorithms. The details of data selection and fine-tuning process are depicted in Figure 2. Our task model $\mathcal{F}$ (any Deep Learning-based NLP model, such as Transformer, etc.,) is pre-trained on the full training data set $\mathcal{X} = \{x_i\}_{i=1}^{n}$, where x_i is a sentence, n is training set size. Then, as in Liu et al. (2019), we shuffle and randomly partition $\mathcal{X}$ into T disjoint data batches so that $\mathcal{X} = \{\mathcal{B}_t\}_{t=1}^{T} = \{\mathcal{B}_1, \mathcal{B}_2, ..., \mathcal{B}_T\}$, with $\mathcal{B}_t = \{x_{(t-1)n|T+1}, x_{(t-1)n|T+2}, ..., x_{tn|T}\}$. $n|T$ is the integer division of n by T, and $T \leq t$. If $\mathrm{mod}\ (n, T) \neq 0$, then the last batch has a variable size of $\mathrm{mod}\ (n, T)$ and collects the remaining sentences. For each batch, we select and denote $\hat{\mathcal{B}}_t = \{(x_i)_{i=1}^{o} | x_i \in \mathcal{B}_t\}$ as the selected data with size o. After obtaining $\hat{\mathcal{B}}_t$, we use $\hat{\mathcal{B}}_t$ to fine-tune $\mathcal{F}$. $\mathcal{F}$ and its encoder g is updated on $\hat{\mathcal{B}}_t$ for T times in an epoch, and each update is based on the previous checkpoint.

Our goal is to improve our task model $\mathcal{F}$ on any

test domain whose distribution is different from the source training set $\mathcal{X}$, by learning an effective subset of $\mathcal{X}$ to fine-tune $\mathcal{F}$. Figure 2-(1) shows our new model.

Our framework is based on Actor-Critic reinforcement learning. Our Actor-Critic method has the following properties, which are very desirable to achieve our goals: variance deduction, efficient sampling, and credit assignment (Konda and Tsitsiklis, 2000; Grondman et al., 2012). Our framework consists of policy and value networks jointly and dynamically learned together with the task model using the advantage error computed from the reward function. In the following context, we will first introduce our Actor-Critic algorithm (Section 2.1) and then our three different reward functions based on diversity (Section 2.2).

2.1 Advantage Actor Critic

2.1.1 Markov Decision Process

We cast the data selection as a reinforcement learning problem with a Markov Decision Process (MDP). Our data selection policy π, a mapping from states to actions, serves as an agent to interact with an environment that constitutes an NLP model, we call task model $\mathcal{F}$ over T time steps. At each time step t, a state s_t, an action a_t and a reward r_t are collected.

First, the encoder g inside the NLP model (e.g. an embedding layer in LSTM, or an encoder in transformer) transforms a batch of data $\mathcal{B}_t$ into its embedding s_t ($s_t = g(B_t)$ in Fig 2, e.g. vector representations of the encoder output of a sentence).

Secondly, the policy π outputs a probability distribution for the batch of state s_t, so that each sentence is associated with a probability representing how likely it is going to be selected. The selected subset, denoted as $\hat{\mathcal{B}}_t$, is then obtained by Bernoulli sampling each sentence in the state s_t. For example, for the k-th sentence in the batch with probability p_k, we generate a random number between 0 and 1. If p_k is larger than the random number, then the k-th sentence is selected; otherwise, it is not selected. The result of Bernoulli sampling is represented as a vector a_t, where each value in it is either 0 or 1 representing each sentence in the batch not being or being selected.

Thirdly, task model $\mathcal{F}$ as well as encoder g are finetuned by the selected subset $\hat{\mathcal{B}}_t$. In the meantime, a scalar reward $r_t = \mathcal{R}(g(\hat{\mathcal{B}}_t))$ is calculated by designed diversity reward functions $\mathcal{R}$ which we give definitions in Section 2.2.

Finally, the policy agent π updates its weights using the collected s_t, a_t and r_t, where state s_t is the encoded representation of a batch of data $\mathcal{B}_t$, action a_t is a vector with each value of either 0 or 1 representing each sentence in the batch not being or being selected, and the scalar reward r_t measures the diversity of selected batch $\hat{\mathcal{B}}_t$. This optimization process is expanded in next section 2.1.2.

2.1.2 Training algorithm

We employ the Advantage Actor Critic algorithm that uses $A(s_t, a_t)$, the advantage of action a_t in state s_t to scale the policy gradient. Specifically, the advantage of action a_t in state s_t is defined in Mnih et al. (2016) as

$$A(s_t, a_t) = \mathcal{Q}(s_t, a_t) - \mathcal{V}(s_t) \quad (1)$$

$$\approx \sum_{j=0}^{T-t} \gamma^j r_{t+j} - \mathcal{V}(s_t) \quad (2)$$

where $\gamma \in (0, 1]$ is the discounting factor, and we set its value as 0.99. $\mathcal{V}$ is the value function (critic) implemented as a value network in Figure 2-(1).

The data selection policy π (actor) is implemented as a policy network, whose training objective is

$$\nabla_\theta \mathcal{J}(\theta) = E_{\pi_\theta} \nabla_\theta \log \pi_\theta(a_t|s_t;\theta) A(s_t, a_t; \theta, \theta_v).$$

The parameters of the policy network θ are updated by:

$$\theta_{t+1} = \theta_t + \alpha \nabla_\theta \log \pi(a_t|s_t;\theta_t) A(s_t, a_t; \theta_t, \theta_{vt}) \quad (3)$$

where α is the learning rate, and Equation 2 estimates $A(s_t, a_t; \theta, \theta_v)$.

The objective of value network is:

$$\nabla_{\theta_v} \mathcal{V}(\theta_v) = E_{\pi_\theta} \nabla_{\theta_\mathcal{V}} (r_t - \mathcal{V}(s_t;\theta_v))^2$$

The parameters of value function θ_v is updated by:

$$\theta_{v(t+1)} = \theta_{vt} + \alpha \nabla_{\theta_{vt}} (r_t - \mathcal{V}(s_t;\theta_{vt}))^2 \quad (4)$$

The policy network $\pi(a_t|s_t;\theta)$ is a two-layer network that has two nodes in the first layer and one hidden output produced by the $\tanh$ activation function and one softmax output. The value network is a two-layer network that has eight nodes

Algorithm 1 DAAC Training Algorithm

Input: Epoch L, learning rate α, discount factor γ, training set $\mathcal{X}$, pre-trained task model $\mathcal{F}$ (including encoder g), reward function $\mathcal{R}$ (discussed in section 2.2)

Output: selected data, fine-tuned $\mathcal{F}$, policy π_θ, data value estimator $\mathcal{V}_{\theta_v}$

Initialize data selection policy π_θ and value estimator $\mathcal{V}_{\theta_v}$
for episode $l = 1$ to L **do**
  Shuffle (uniformly at random) all training samples;
  Equipartition $\mathcal{X}$ into T (disjoint) sets with same size $n|T$: $\mathcal{X} = \{\mathcal{B}_t\}_{t=1}^{T} = \{\mathcal{B}_1, \mathcal{B}_2, ..., \mathcal{B}_T\}$;
  Initialize an empty list: episode history Υ
  for all $\mathcal{B}_t \in \mathcal{X}$ (uniform transition probability) **do**
    $s_t = g_t(\mathcal{B}_t)$;
    Obtain batch action a_t by sampling based on $\pi_\theta(s_t)$;
    $\hat{\mathcal{B}}_t = \{(x_i)_{i=1}^{o} | a_i = 1\}$, where o is selected sample size;
    Update task model $\mathcal{F}(g_t)$ by fine-tuning on $\hat{\mathcal{B}}_t$;
    $r_t = \mathcal{R}(\hat{\mathcal{B}}_t, g)$;
    Store (s_t, a_t, r_t) to episode history Υ;
  end for
  for all $(s_t, a_t, r_t) \in \Upsilon$ **do**
    Obtain $A(s_t, a_t)$ for each batch (Eq. 2);
    Update policy weights θ (Eq. 3);
    Update value estimator weights θ_v (Eq. 4);
  end for
  Clear episode history Υ;
end for
return $\mathcal{F}$,π_θ and $\mathcal{V}_{\theta_v}$

in the first layer and one hidden output produced by the ReLU activation function, and one linear output for the value network $\mathcal{V}(s_t; \theta_v)$ following Mnih et al. (2016). The training algorithm is shown in Algorithm 1.

2.2 Diversity Measures

The reward function in Section 2.1 is the diversity of the selected batch subset. We do not need any target domain knowledge, and the diversity is measured with the following three methods.

2.2.1 Max Dispersion

The dispersion of a set is the sum of all pair-wise distances within the set (Cevallos et al., 2016). Intuitively, maximizing the dispersion (denote as **MD**) of a set can enlarge the semantic coverage of the set, and thus diversify the content of the set. Formally, x_i and x_j are any two training sentences in a training set G, and their sentence embeddings are $g(x_i)$ and $g(x_j)$, respectively, we define the $dispersion$ as

$$\mathcal{D}(G) = \sum_{(x_i, x_j) \in G} d(x_i, x_j), \tag{5}$$

where $d(x_i, x_j) = 1 - \frac{g(x_i) \cdot g(x_j)}{\|g(x_i)\| \|g(x_j)\|}$. The diversity of a set of training samples (sentences) is computed as the sum of each pairwise sample distance, which can be measured by any distance metrics, i.e., Cosine distance. We take the NLP task model encoder output as the sentence embedding to compute pairwise sample distance. The sentence embedding preserves pairwise sentence distance from the semantic space to the high-dimensional vector space. More details of the algorithm are shown in Algorithm 2, with a worst time complexity of $O(n^2)$, where n is the number of samples in the set.

2.2.2 Convex Hull Volume

Instead of looking at each pair of sentences, we can simplify the computing process by looking at the boundary of a set of training samples in the high dimensional space and compute the volume of such space using Convex Hull Volume (denote as **CV**). Specifically, we treat each sentence as a point in the embedding space and approximate the volume of a convex body. In a Euclidean plane, given a finite set of points Q, it is sometimes interesting to determine its convex hull, namely the minimum convex polygon so that any point of Q is either inside this polygon or at its border. There are a number of algorithms to compute the convex hull. Since our embedding is in high dimension, i.e., 256, we consider N-dimensional quickhull, which was introduced in Barber et al. (1996) . Just like the quicksort algorithm, it has the expected time complexity of $O(n \log n)$ by divide and conquer approach, but may degenerate to $O(n^2)$ in the worst case. Details shown in appendix Algorithm 3.

2.2.3 Graph Entropy

Applying entropy on the distances, we introduce Graph Entropy (**GE**), following the so-called local measure of distance-based substructure entropy, details in Dehmer and Mowshowitz (2011). As

shown in Figure 2, graph entropy measures the uncertainty of the pairwise distance of sentence (or word) embeddings in the geometric space. To compute graph entropy of a set of training samples, we cast the sentence embeddings as vertices and distances between sentence embeddings as edges to form a clique. The distance entropy $\mathbf{I}$ of each sample x is defined as

$$\mathbf{I}(x) = -\sum_{i=1}^{M} \mathbf{f}(d(x, x_i)) \log(\mathbf{f}(d(x, x_i))).$$

M is the number of sentences in the set, $d(x, x')$ is the Cosine distance between the encoder output vector representation of x and x', $\mathbf{f}(d(x, x')) = \frac{d(x,x')}{\sum_{i=1}^{M} d(x,x_i)}$ is the relative distance of edge $d(x, x')$ (instead of words). The graph entropy of entire set is then defined as the sum of all distance entropy $\mathbf{I}$:

$$\mathcal{G}(G) = \sum_{i=1}^{M} \mathbf{I}(x_i) \tag{6}$$

For a set with n data samples, there are $\binom{n}{2}$ edges in the resulting graph. The time complexity of graph entropy is thus $\mathcal{O}(n^2)$ by looping through each edge in the graph. Details are shown in Algorithm 5.

3 Experiments

We describe our experimental details and demonstrate that our DAAC improves baselines in three NLP applications, including two recognition tasks: sentiment analysis, named entity recognition (NER), and one generation task of language modeling, without the knowledge of the target domain. We give hyperparameters of DAAC in appendix.

Finetuning vs. Data Selection We introduce our four models based on DAAC. The first three models finetune a pretrained NLP model using max dispersion **MD**, convex hull volume **CV** and graph entropy **GE** as reward function. As a comparison, we add the fourth model trained on selected data from scratch that does not rely on a pretrained model, denoted as data selection **DS**. Specifically, we train A2C using MD as reward function for 150 epochs first, and use the trained A2C to select data samples from all source data, then use only the selected data to train the NLP model from scratch. We select 3833 (63.8%) and 7630 (54.3%) samples from the source domain of sentiment analysis and NER respectively, and 21017 (49.9%) samples from Penn Treebank and 18043 (49.1%) samples from WikiText-2 in language modeling experiment.

	auto	beauty	food	instrs	office	comptr	tools
All	56.4	54.0	56.0	59.1	56.8	55.1	55.9
Rand	57.0	54.9	56.9	59.5	59.2	56.1	56.0
Mtl	59.4	55.8	60.7	61.0	61.8	56.2	62.5
Meta	56.3	52.2	58.4	56.4	58.1	52.9	48.8
Sim	73.4	62.9	77.3	79.4	78.4	63.0	81.2
MD	76.8	65.2	80.5	82.9	82.8	66.0	85.7
CV	76.4	65.4	80.4	82.8	82.2	66.0	85.7
GE	75.4	65.0	78.9	81.4	80.5	65.5	83.3
DS	**78.0**	**65.9**	**81.7**	**84.4**	**83.7**	**67.5**	**86.3**
+%	18.6	10.1	21.0	23.4	21.9	11.3	23.8

Table 1: Sentiment analysis accuracy [%] on amazon unprocessed domains. Full results of 21 domains in appendix. Baselines **All**, **Rand**, marginal transfer learning **Mtl** (Blanchard et al., 2021), Metareg **Meta** (Balaji et al., 2018) and our four measures are trained on the joint dataset of books, dvd & kitchen domain, and do not use any test/target domain data, while **Sim** (Liu et al., 2019) uses target domain electronics. Results in each domain are averaged over five random seeds. Last column: absolute improvement between **DS** and **Mtl**.

Baselines We compare our models with five baselines: 1) **All** The models are trained on all source data; 2) **Rand** The models are trained on randomly selected 50% source data; 3) Marginal transfer learning (**Mtl**) by Blanchard et al. (2021), a domain generalization framework using kernel methods to augment feature space. 4) MetaReg (**Meta**) by Balaji et al. (2018), a domain generalization method using meta-learning. 5) **Sim** by Liu et al. (2019) that uses similarity features between target test domain and source training domains. To be noted, **Sim** is a domain adaptation method, which requires the usage of target domain data, while other baselines and our four models do not use target domain data.

3.1 Sentiment Analysis

Settings We use the Amazon product review dataset (Blitzer et al., 2007) for the sentiment analysis task. The sentiment analysis baseline is a CNN classifier (Kim, 2014). We pretrain the CNN for two epochs as (Liu et al., 2019) for a fair comparison.

Results Table 1 shows the results on amazon domains. Max dispersion outperforms other diversity measures in 11 out of all 22 unseen domains, and the Data Selection achieves a most boost of 38.0% (relative) and 23.8% (absolute) percent in the tools domain. This observation aligns with previous findings that 60%-70% important samples can perform similarly or better than training on the entire dataset

	politics	science	music	literature	ai
All	26.49	19.84	12.26	16.38	13.92
Rand	26.07	19.68	12.81	17.47	13.68
Mtl	28.47	22.47	13.49	18.97	15.68
Meta	29.16	22.57	13.75	20.92	15.74
Sim	29.37	22.51	15.31	20.63	16.00
MD	29.64	23.09	15.69	21.21	16.72
CV	29.42	22.85	15.71	21.30	**17.13**
GE	**29.75**	**23.75**	**16.03**	**21.58**	16.91
DS	29.74	23.18	15.22	20.56	15.97

Table 2: NER F1-scores. Results are averaged over three runs.

	WikiText-2		Penn Treebank	
	IWSLT'17	Bio'21	IWSLT'17	Bio'21
All	328.23	259.47	147.03	117.17
Rand	515.22	456.78	234.14	157.39
MD	323.89	251.45	144.78	114.68
CV	325.22	252.17	145.43	115.31
GE	324.57	251.91	144.95	114.89
DS	**321.05**	**222.78**	**141.02**	**72.52**

Table 3: Language modeling: Perplexity scores on two test domains. First row: source training domain; Second row: test domains.

(Yoon et al., 2019; Fan et al., 2017). We also conduct a domain adaptation experiment which aims to maximize the test accuracy on one specific target domain and outperform all baselines with a significant test $p = 0.00038$, as shown in table 8 in appendix.

3.2 Named Entity Recognition

Settings We use the CoNLL2003 English NER dataset (Sang and Meulder, 2003) as source training set and the five domains from CrossNER dataset (Liu et al., 2020) as test sets, which has specialized entity categories for each domain. We finetune the pretrained BERT model (Devlin et al., 2018) on source training set by adding a linear layer on top of the hidden-states output and then evaluate the F1-scores on five test domains.

Results The result is reported in Table 2. Graph Entropy achieves best performance among all the three diversity measures. In particular, Data Selection performs better than training with all data directly (All) and random selection (Rand), which means important samples are selected and noisy samples are filtered out.

3.3 Language Modeling

Settings Our baseline is a Transformer language model (Vaswani et al., 2017) with default hyperparameters. We experiment with two moderate size datasets WikiText-2 (Merity et al., 2016) and Penn Treebank. As for evaluation, we report perplexity scores on two translation datasets from different domains, IWSLT' 17 (TED talk english) and WMT Biomedical' 21 (english). The baseline models are trained using the fairseq toolkit (Ott et al., 2019) and stop training until the validation perplexity score does not improve for 5 epochs.

Results The evaluation results are shown in Table 3. The perplexity on two test domains have been improved. Specifically, test perplexity of Bio' 21 has an improvement of 44.65 (37.9% relative improvement) while trained by our selected data. The comparison with randomly selected result (Rand) further proves the effectiveness of data selected by DAAC.

3.4 Ablation Analysis

MD vs. GE In our three experiments we observe Max Dispersion (MD) performs best in sentiment analysis and language modeling, and Graph Entropy measure (GE) achieves the best performance in named entity recognition. We infer the smaller improvement of convex hull volume might be due to error propagation of the quickhull algorithm.

Regarding the reason why GE performs best in named entity recognition, we first analyze the difference of operations on sentence embeddings for GE and MD. Since GE is calculated by the sum of distance entropy of each sentence (sample), there are more operations (eg. log and normalization) depending on the sentence embedding input than MD. Thus, performance of GE is more relying on the precision of sentence embeddings compared to that of MD. Then looking into the key difference between the task of NER and other tasks, the source of sentence embeddings are apparently different. In NER, sentence embeddings are trained by BERT, while in sentiment analysis they are trained by CNN. As a result, we conjecture the more precise sentence embeddings trained by BERT lead to the better performance of GE on NER task. To verify this assumption, we quantitatively evaluate the quality of sentence embeddings generated by finetuned CNN and finetuned BERT on semantic textual similarity (STS) benchmarks STS-B-Dev (Cer et al., 2017), STS14 and STS15 (Agirre et al., 2014, 2015). We report the Spearman's rank correlation between the cosine similarity of sentence embeddings using the SentEval toolkit (Conneau and Kiela, 2018). The results in table 4 show preliminary evidence that our conjecture holds. In short, MD can be consid-

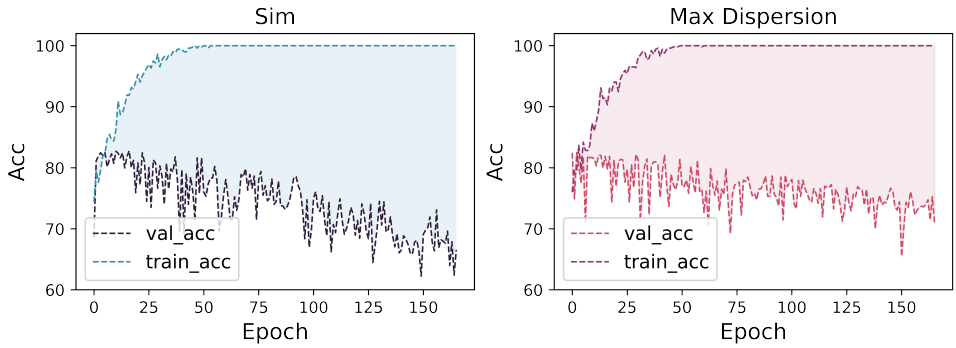

Figure 3: Training/Validation accuracy change over epochs. Validate set is gourmet food domain of unprocessed Amazon product reviews. Training set is a joint set of dvd, kitchen and books, and target domain in **Sim** is electronics. **MD** can help alleviate overfitting on test domains compared to the domain adaptation method **Sim** (Liu et al., 2019)

	STS-B-Dev	STS14	STS15
Finetuned-CNN	29.92	23.26	19.77
Finetuned-BERT	32.02	24.54	23.06

Table 4: Spearman's rank correlation between the cosine similarity of sentence embeddings generated by finetuned-CNN and finetuned-BERT. Numbers are reported as $\rho \times 100$.

ered for general tasks, however, when there is a reliable sentence embedding source, GE can be a better choice than MD.

Overfitting We plot the validation accuracy curve of the sentiment classifier on one test domain, as shown in Fig 3. The validation accuracy of Sim (Liu et al., 2019) model degrades faster and more significant than DAAC. The gap of blue and red area, which is the gap between training accuracy and validation accuracy of models finetuned by Sim and MD, suggests DAAC can help alleviate overfitting on test domains compared to the domain adaptation method Sim.

Time As discussed in 2.2, MD and GE both take account for the internal structure of a set of training samples and take time $O(n^2)$, where n is size of the set. CV can reduce time to $O(n \log n)$ by utilizing divide and conquer, while accompanying with the degradation of performance. In theory, the entire training time of DAAC-finetuning is approximately T times compared to training with all data, where T is the time steps in Markov Decision Process, or the number of batches. In the finetuning setting, we follow Liu et al. (2019) and Yoon et al. (2019), using $T \in [2, 4]$ and finetune NLP model for 200 time steps. However, in the data selection (DS) setting, the training time is roughly 20 times more compared to the finetuning setting, since we train A2C for extra epochs (10000 time steps with $T = 60$, batch size 100) to ensure both the value and

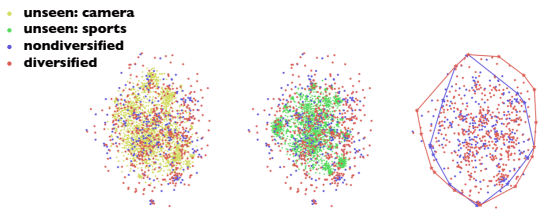

Figure 4: t-SNE plot of diversified/nondiversified training data and their coverage with unseen domains.

	Camera	Sports	Food
Diversified	7956	8443	5158
Nondiversified	8348	8886	5469

Table 5: Out-of-vocabulary of diversified and nondiversified set

policy network converge. In practice, training with all source data directly (All) in sentiment analysis takes 131 seconds while finetuning with MD takes 217 seconds, and selecting data out then training (DS) takes 4774 seconds on one Tesla V100 GPU.

Visualization Figure 4 visualizes data samples selected by our DAAC (diversified, red) and those randomly selected (nondiversified, blue), as well as the test data samples (unseen domains) in the embedding space. We observe that our DAAC selected samples are more widely spread, thus covering more semantic meaning for most test sets. Figure 4 (right) shows that DAAC selected samples has a larger convex hull volume than random selected samples after removing outliers. Furthermore, table 5 shows that our DAAC has a smaller size of out-of-vocabulary than the baseline on the test domains. Data samples are selected from the magazines domain of Amazon product reviews (Blitzer et al., 2007). We use max dispersion (MD) diversity measure, sentence-transformer toolkit (Reimers and Gurevych, 2019), and plot the embedding into two-dimension t-SNE.

4 Related work

Existing domain generalization methods related to data focus on data augmentation (Zhou et al., 2020; Qiao et al., 2020) and data generation (Wang et al., 2021). These methods increase the quantity of training set while our method utilizes existing data maximally by modeling the intrinsic sample dependency.

There have been several influential works (Moore and Lewis, 2010; Axelrod et al., 2011; Ruder and Plank, 2017a) on data selection that significantly contributed to today's NLP state-of-the-arts. Fan et al. (2017) proposes a data filter based on deep reinforcement learning on image and sentiment classification task. Feng et al. (2018) implemented an instance selector using reinforcement learning to filter noisy data to improve the accuracy on natural language inference (Qu et al., 2019), sentiment analysis, part-of-speech tagging and dependency parsing (Liu et al., 2019). These work select the training data close to a given target domain for domain adaptation. In contrast, we aim to enhance the model generalization and increase the accuracy on any arbitrary domain.

As for diversity measures, Ruder and Plank (2017b) examines the effects of Shannon entropy in the transfer learning setting, but they do not consider content semantic meaning. Shi et al. (2021) uses determinantal point processes to select diverse data to reduce the labor of annotating training examples for dependency parsing, but the proposed diversity measure cannot be generalizaed to other NLP tasks. Other works incorporate the notion of diversity into topics like language representation (Chubarian et al., 2021), and ensemble language modeling (Duan et al., 2021).

5 Conclusion

We introduce Actor-Critic reinforcement learning rewarded with diversity measures to select effective training data that significantly enhances the accuracy of sentiment analysis, named entity recognition, and language modeling tasks across various domains. Without any target domain knowledge, our method outperforms the CNN, Transformer and BERT baselines. Our experiments show that modeling sample dependency by increasing training set diversity enhances the learning generalization and prediction accuracy.

Acknowledgments

We appreciate the National Science Foundation (NSF) Award No. 1747728 and NSF CRAFT Award, Grant No. 22001 to fund this research. We are also thankful for the support of the Google Cloud Research Program.

References

Eneko Agirre, Carmen Banea, Claire Cardie, Daniel Cer, Mona Diab, Aitor Gonzalez-Agirre, Weiwei Guo, Inigo Lopez-Gazpio, Montse Maritxalar, Rada Mihalcea, et al. 2015. Semeval-2015 task 2: Semantic textual similarity, english, spanish and pilot on interpretability. In *Proceedings of the 9th international workshop on semantic evaluation (SemEval 2015)*, pages 252–263.

Eneko Agirre, Carmen Banea, Claire Cardie, Daniel M Cer, Mona T Diab, Aitor Gonzalez-Agirre, Weiwei Guo, Rada Mihalcea, German Rigau, and Janyce Wiebe. 2014. Semeval-2014 task 10: Multilingual semantic textual similarity. In *SemEval@ COLING*, pages 81–91.

Roee Aharoni and Yoav Goldberg. 2020. Unsupervised domain clusters in pretrained language models.

Amittai Axelrod, Xiaodong He, and Jianfeng Gao. 2011. Domain adaptation via pseudo in-domain data selection. In *Proceedings of the 2011 Conference on Empirical Methods in Natural Language Processing*, pages 355–362, Edinburgh, Scotland, UK. Association for Computational Linguistics.

Yogesh Balaji, Swami Sankaranarayanan, and Rama Chellappa. 2018. Metareg: Towards domain generalization using meta-regularization. *Advances in Neural Information Processing Systems*, 31:998–1008.

C Bradford Barber, David P Dobkin, and Hannu Huhdanpaa. 1996. The quickhull algorithm for convex hulls. *ACM Transactions on Mathematical Software (TOMS)*, 22(4):469–483.

Gilles Blanchard, Aniket Anand Deshmukh, Ürün Dogan, Gyemin Lee, and Clayton D. Scott. 2021. Domain generalization by marginal transfer learning. *J. Mach. Learn. Res.*, 22:2:1–2:55.

John Blitzer, Mark Dredze, and Fernando Pereira. 2007. Biographies, bollywood, boom-boxes and blenders: Domain adaptation for sentiment classification. In *Proceedings of the 45th annual meeting of the association of computational linguistics*, pages 440–447.

Daniel Cer, Mona Diab, Eneko Agirre, Inigo Lopez-Gazpio, and Lucia Specia. 2017. Semeval-2017 task 1: Semantic textual similarity-multilingual and cross-lingual focused evaluation. *arXiv preprint arXiv:1708.00055*.

Alfonso Cevallos, Friedrich Eisenbrand, and Rico Zenklusen. 2016. Local search for max-sum diversification. *CoRR*, abs/1607.04557.

Karine Chubarian, Abdul Rafae Khan, Anastasios Sidiropoulos, and Jia Xu. 2021. Grouping words with semantic diversity. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 3217–3228, Online. Association for Computational Linguistics.

Alexis Conneau and Douwe Kiela. 2018. Senteval: An evaluation toolkit for universal sentence representations. *arXiv preprint arXiv:1803.05449*.

Geoff Davis, Stephane Mallat, and Marco Avellaneda. 1997. Adaptive greedy approximations. *Constructive approximation*, 13(1):57–98.

Matthias Dehmer and Abbe Mowshowitz. 2011. A history of graph entropy measures. *Information Sciences*, 181(1):57–78.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. BERT: pre-training of deep bidirectional transformers for language understanding. *CoRR*, abs/1810.04805.

Jesse Dodge, Gabriel Ilharco, Roy Schwartz, Ali Farhadi, Hannaneh Hajishirzi, and Noah Smith. 2020. Fine-tuning pretrained language models: Weight initializations, data orders, and early stopping.

Zhibin Duan, Hao Zhang, Chaojie Wang, Zhengjue Wang, Bo Chen, and Mingyuan Zhou. 2021. EnsLM: Ensemble language model for data diversity by semantic clustering. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 2954–2967, Online. Association for Computational Linguistics.

Yang Fan, Fei Tian, Tao Qin, Jiang Bian, and Tie-Yan Liu. 2017. Learning what data to learn. *arXiv preprint arXiv:1702.08635*.

Jun Feng, Minlie Huang, Li Zhao, Yang Yang, and Xiaoyan Zhu. 2018. Reinforcement learning for relation classification from noisy data. In *Proceedings of the aaai conference on artificial intelligence*, volume 32.

Zhiqiang Gong, Ping Zhong, and Weidong Hu. 2019. Diversity in machine learning. *IEEE Access*, 7:64323–64350.

Ivo Grondman, Lucian Busoniu, Gabriel AD Lopes, and Robert Babuska. 2012. A survey of actor-critic reinforcement learning: Standard and natural policy gradients. *IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews)*, 42(6):1291–1307.

Han Guo, Ramakanth Pasunuru, and Mohit Bansal. 2020. Multi-source domain adaptation for text classification via distancenet-bandits.

Dan Hendrycks, Xiaoyuan Liu, Eric Wallace, Adam Dziedzic, Rishabh Krishnan, and Dawn Song. 2020. Pretrained transformers improve out-of-distribution robustness. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 2744–2751, Online. Association for Computational Linguistics.

Yoon Kim. 2014. Convolutional neural networks for sentence classification. *CoRR*, abs/1408.5882.

Vijay R Konda and John N Tsitsiklis. 2000. Actor-critic algorithms. In *Advances in neural information processing systems*, pages 1008–1014.

Miaofeng Liu, Yan Song, Hongbin Zou, and Tong Zhang. 2019. Reinforced training data selection for domain adaptation. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 1957–1968, Florence, Italy. Association for Computational Linguistics.

Zihan Liu, Yan Xu, Tiezheng Yu, Wenliang Dai, Ziwei Ji, Samuel Cahyawijaya, Andrea Madotto, and Pascale Fung. 2020. Crossner: Evaluating cross-domain named entity recognition. *CoRR*, abs/2012.04373.

Stephen Merity, Caiming Xiong, James Bradbury, and Richard Socher. 2016. Pointer sentinel mixture models. *arXiv preprint arXiv:1609.07843*.

Michele Merler, Nalini Ratha, Rogerio S Feris, and John R Smith. 2019. Diversity in faces. *arXiv preprint arXiv:1901.10436*.

Volodymyr Mnih, Adria Puigdomenech Badia, Mehdi Mirza, Alex Graves, Timothy Lillicrap, Tim Harley, David Silver, and Koray Kavukcuoglu. 2016. Asynchronous methods for deep reinforcement learning. In *International conference on machine learning*, pages 1928–1937. PMLR.

Robert C. Moore and William Lewis. 2010. Intelligent selection of language model training data. In *Proceedings of the ACL 2010 Conference Short Papers*, pages 220–224, Uppsala, Sweden. Association for Computational Linguistics.

Mathias Müller, Annette Rios, and Rico Sennrich. 2019. Domain robustness in neural machine translation. *arXiv preprint arXiv:1911.03109*.

Myle Ott, Sergey Edunov, Alexei Baevski, Angela Fan, Sam Gross, Nathan Ng, David Grangier, and Michael Auli. 2019. fairseq: A fast, extensible toolkit for sequence modeling. In *Proceedings of NAACL-HLT 2019: Demonstrations*.

Chao Qian, Jing-Cheng Shi, Yang Yu, Ke Tang, and Zhi-Hua Zhou. 2016. Parallel pareto optimization for subset selection. In *IJCAI*, pages 1939–1945.

Fengchun Qiao, Long Zhao, and Xi Peng. 2020. Learning to learn single domain generalization. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 12556–12565.

Chen Qu, Feng Ji, Minghui Qiu, Liu Yang, Zhiyu Min, Haiqing Chen, Jun Huang, and W Bruce Croft. 2019. Learning to selectively transfer: Reinforced transfer learning for deep text matching. In *Proceedings of the Twelfth ACM International Conference on Web Search and Data Mining*, pages 699–707.

Alan Ramponi and Barbara Plank. 2020. Neural unsupervised domain adaptation in nlp—a survey.

Abhinav Rastogi, Xiaoxue Zang, Srinivas Sunkara, Raghav Gupta, and Pranav Khaitan. 2020. Towards scalable multi-domain conversational agents: The schema-guided dialogue dataset. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pages 8689–8696.

Nils Reimers and Iryna Gurevych. 2019. Sentence-bert: Sentence embeddings using siamese bert-networks. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics.

Seyed Saeed Changiz Rezaei. 2013. Entropy and graphs. *arXiv preprint arXiv:1311.5632*.

Sebastian Ruder and Barbara Plank. 2017a. Learning to select data for transfer learning with bayesian optimization. *CoRR*, abs/1707.05246.

Sebastian Ruder and Barbara Plank. 2017b. Learning to select data for transfer learning with Bayesian optimization. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 372–382, Copenhagen, Denmark. Association for Computational Linguistics.

Erik F. Tjong Kim Sang and Fien De Meulder. 2003. Introduction to the conll-2003 shared task: Language-independent named entity recognition. *CoRR*, cs.CL/0306050.

Candice Schumann, Susanna Ricco, Utsav Prabhu, Vittorio Ferrari, and Caroline Pantofaru. 2021. A step toward more inclusive people annotations for fairness. *arXiv preprint arXiv:2105.02317*.

Shreya Shankar, Yoni Halpern, Eric Breck, James Atwood, Jimbo Wilson, and D Sculley. 2017. No classification without representation: Assessing geodiversity issues in open data sets for the developing world. *arXiv preprint arXiv:1711.08536*.

Tianze Shi, Adrian Benton, Igor Malioutov, and Ozan Irsoy. 2021. Diversity-aware batch active learning for dependency parsing. *CoRR*, abs/2104.13936.

Marlies van der Wees, Arianna Bisazza, and Christof Monz. 2017. Dynamic data selection for neural machine translation. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 1400–1410, Copenhagen, Denmark. Association for Computational Linguistics.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. *CoRR*, abs/1706.03762.

Jindong Wang, Cuiling Lan, Chang Liu, Yidong Ouyang, Wenjun Zeng, and Tao Qin. 2021. Generalizing to unseen domains: A survey on domain generalization. *arXiv preprint arXiv:2103.03097*.

Baoyuan Wu, Weidong Chen, Peng Sun, Wei Liu, Bernard Ghanem, and Siwei Lyu. 2018. Tagging like humans: Diverse and distinct image annotation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 7967–7975.

Jinsung Yoon, Sercan Ömer Arik, and Tomas Pfister. 2019. Data valuation using reinforcement learning. *CoRR*, abs/1909.11671.

Kaiyang Zhou, Yongxin Yang, Timothy Hospedales, and Tao Xiang. 2020. Learning to generate novel domains for domain generalization. In *European conference on computer vision*, pages 561–578. Springer.

A Appendix

A.1 Notations

Notation	Meaning
$\mathcal{F}$	NLP model
g	encoder
$\mathcal{X}$	Training data set
x_i	sentence
$\|\mathcal{X}\|$	Training set size
T	number of batches; maximum training steps in an epoch
B_t	batch
$\hat{B}_t$	selected batch
s_t	batch state
s_k	single sentence state
a_t	action on batch B_t at time step t
a_k	action on single sample x_k at time step t
$\|B_t\|$	batch size
π	policy
r_t	reward at time step t
R_t	total future reward from t to T
$Q^\pi(s_t, a_t)$	action value
$V^\pi(s_t)$	expected future return following π since time step t
$b(s_t)$	baseline function
θ	parameters of policy network
∇_θ	$\frac{\partial X}{\partial \theta}$
$\nabla_\theta \mathcal{J}(\theta)$	objective function of policy
L	epoch number
α	learning rate
γ	discount factor
$\mathcal{F}_t$	pretrained task model (including encoder g)
$\mathcal{E}$	episode record, including s_t, a_t, r_t
G	a set of samples
d	cosine distance of embeddings
M	number of sentences in a set G
J	sentence length
$h(\cdot; n)$	n-gram entropy

Table 6: Notation table

A.2 Hyperparameters of A2C

Hyperparameter	Value
learning rate	$7e-4$
discount factor	0.99
entropy coefficient	0
value function coefficient	0.5
RMSProp epsilon	$1e-5$
number of steps (finetuning)	200
number of steps (data selection)	15000

Table 7: Hyperparameters of A2C

A.3 Diversity algorithms

See 2, 3 and 5.

A.4 Sentiment analysis full results

See table 9 and 8.

Algorithm 2 Max Dispersion set diversity [**MD**]

Input: A batch of training samples $G = \{(s_i)_{i=1}^M\}$ of size M.
Output: Dispersion of the batch $D(G)$

Initialize $D(G) = 0$;
Initialize an empty set S;
for all $s_i \in G$ **do**
  Obtain sentence embedding $v(s_i)$ by averaging all word embedding in s_i;
  Normalize $v(s_i)$ by softmax function;
  for all $s_j \in G$ if $s_j \neq s_i$ and tuple (i, j) not in S and tuple (j, i) not in S **do**
    Obtain sentence embedding $v(s_j)$ by averaging all word embedding in s_j;
    Normalize $v(s_j)$ by softmax function;
    Compute Cosine distance $d(v(s_i), v(s_j)$;
    $D(G) = D(G) + d(v(s_i), v(s_j))$;
    Add tuple (i, j) to S;
  end for
end for
return $D(G)$

Algorithm 3 Quickhull algorithm

Input: A batch of training samples $G = \{(s_i)_{i=1}^M\}$ with size M; $M \geq 2$
Output: Convex hull set h

Initialize empty set convex hull h;
Initialize empty dictionary d;
for all $s \in G$ **do**
  for all $s' \in G$ that $s' \neq s$ **do**
    Compute Cosine similarity $d(v(s), v(s'))$ between sentence embedding of s and s;
    $d[(s, s')] = d(v(s), v(s'))$
  end for
end for
Sort d in descending order
Add the two sentence s_p and s_q that has the max distance to convex hull set h. Line formed by s_p and s_q segment the space into left half S_1 and right half S_2.
Call subroutine FindHull(S_1, s_p, s_q)
Call subroutine FindHull(S_2, s_q, s_p)
return Convex hull set h

		books	dvd	electronics	kitchen
w/o tgt	All	77.58	79.88	83.50	84.50
	Rand	76.39	79.30	82.99	84.01
	Mtl	77.20	79.90	84.25	85.50
	Meta	76.10	79.60	83.90	85.10
w/o tgt	MD	79.67	**84.08**	85.29	87.38
	CV	79.47	83.08	84.58	87.48
	GE	**80.08**	83.58	**85.68**	**87.88**
w/ tgt	Sim	78.97	82.07	82.28	86.18

Table 8: Sentiment Analysis accuracy on one test domain[%]. The "book" column is tested on the book domain, while using other three domains for training.

Algorithm 4 Subroutine: FindHull

Input: A batch of training samples S; point p and q.

From the given set of points in S, find farthest point f
Add point f to convex hull set h. Three points p, q, and f partition the remaining points of S into 3 subsets: S_0, S_1, and S_2, where S_0 are points inside triangle pqf, S_1 are points on the right side of the line from p to f, and S_2 are points on the right side of the line from f to q.
Call FindHull(S_1, p, f)
Call FindHull(S_2, f, q)

Algorithm 5 Graph entropy based set diversity [**GE**]

Input: A batch of training samples $G = \{(s_i)_{i=1}^{M}\}$ with size M
Output: Graph entropy $\mathcal{G}(G)$

Initialize $\mathcal{G}(G) = 0$;
Initialize empty dictionary d;
for all $s \in G$ **do**
  for all $s' \in G$ that $s' \neq s$ **do**
    Compute Cosine similarity $d(v(s), v(s'))$ between sentence embedding of s and s;
    $d[s] = d[s] + d(v(s), v(s'))$;
  end for
end for
for all $s \in G$ **do**
  Initialize distance entropy of s: $\mathbf{I}(s) = 0$;
  for all $s' \in G$ that $s' \neq s$ **do**
    Compute Cosine similarity $d(v(s), v(s'))$ between sentence embedding of s and s';
    Compute relative frequency of distance $\mathbf{f}(d(v(s), v(s')))$
    Update $\mathbf{I}(s)$ according to equations in section 2.3.3
  end for
  $\mathcal{G}(G) = \mathcal{G}(G) + \mathbf{I}(s)$;
end for
return $\mathcal{G}(G)$

Domain	All	Rand	Mtl	Meta	MD	CV	GE	DS
apparel	49.43	50.47	47.65	49.92	50.53	50.48	**50.78**	49.95
auto	56.48	57.07	59.46	56.32	76.81	**76.94**	75.49	78.03
baby	50.42	51.09	50.80	50.06	50.86	52.95	**53.12**	52.81
beauty	54.03	54.93	55.89	52.26	65.43	65.43	65.03	**65.97**
camera	49.82	50.30	50.05	49.78	49.91	49.98	49.66	**50.32**
phones	52.79	53.77	53.93	52.51	62.13	61.81	61.18	**61.98**
computer	55.17	56.16	56.20	52.99	66.09	66.00	65.54	**67.54**
food	56.02	56.93	60.73	58.41	80.57	80.40	78.94	**81.76**
grocery	53.06	54.81	57.86	57.73	72.23	72.04	71.59	**73.10**
health	50.92	51.10	49.95	51.03	49.94	49.96	**51.11**	49.68
jewelry	56.63	57.92	58.98	58.15	75.77	75.56	74.84	**76.81**
magazines	50.78	50.56	50.42	50.03	51.15	**51.23**	50.94	50.97
music	50.05	50.27	50.08	49.60	50.08	**50.20**	50.20	50.03
instrs	59.15	59.55	61.88	56.41	82.92	82.82	81.40	**84.45**
office	56.84	59.28	61.87	58.14	82.92	82.23	80.58	**83.73**
outdoor	55.81	57.39	58.23	54.25	72.92	72.60	72.44	74.37
software	49.70	50.45	50.73	49.92	52.30	**52.37**	52.12	51.90
sports	51.08	51.13	49.97	49.98	50.66	50.68	50.70	50.25
tools	55.95	56.04	62.50	48.81	85.71	85.71	83.33	**86.31**
toys	50.07	50.82	50.20	50.35	51.03	51.02	51.38	50.17
video	51.65	50.72	50.08	50.68	50.50	50.70	50.85	**50.20**

Table 9: Sentiment analysis accuracy [%] on unknown domains.

Generate-and-Retrieve: use your predictions to improve retrieval for semantic parsing

Yury Zemlyanskiy[*†] **Michiel de Jong**[*†] **Joshua Ainslie**[‡] **Panupong Pasupat**[‡]
Peter Shaw[‡] **Linlu Qiu**[‡] **Sumit Sanghai**[‡] **Fei Sha**[‡]

[†] University of Southern California [‡] Google Research

{yury.zemlyanskiy,msdejong}@usc.edu

{jainslie,ppasupat,petershaw,linluqiu,sumitsanghai,fsha}@google.com

Abstract

A common recent approach to semantic parsing augments sequence-to-sequence models by retrieving and appending a set of training samples, called exemplars. The effectiveness of this recipe is limited by the ability to retrieve informative exemplars that help produce the correct parse, which is especially challenging in low-resource settings. Existing retrieval is commonly based on similarity of query and exemplar inputs. We propose GandR, a retrieval procedure that retrieves exemplars for which outputs are also similar. GandR first generates a preliminary prediction with input-based retrieval. Then, it retrieves exemplars with outputs similar to the preliminary prediction which are used to generate a final prediction. GandR sets the state of the art on multiple low-resource semantic parsing tasks.

1 Introduction

A common and successful approach to structured prediction problems (Li et al., 2021; Chen et al., 2020) is to treat the gold structure as a sequence and fine-tune a sequence-to-sequence model such as T5 (Raffel et al., 2020) or BART (Lewis et al., 2020). However, the performance of fine-tuned models suffers in low resource scenarios where available training data is limited relative to the complexity of the task (Chen et al., 2020).

Existing work (Pasupat et al., 2021; Gupta et al., 2021; Wang et al., 2022) has found that retrieving related training samples, denoted *exemplars*, and appending the retrieved input-output pairs to the sample input before processing the sample can improve performance in low resource settings. In principle, all information from exemplars is available to the model during training and could be stored in model parameters. However, in practice the model may not successfully retain all information, and reminding the model of salient input-output patterns at test time appears to help.

That raises the question: what exemplars are most informative for the model? Existing work focuses on retrieving exemplars for which the input is partially similar to the test input, effectively answering "*What is the output for similar inputs?*". In this work we explore whether there is complementary information in exemplars that answer the inverse question, "*What is the input for similar outputs?*".

We propose Generate-and-Retrieve (GandR), a method to retrieve exemplars with similar output as well as input. As the true output of a sample is in general unknown, GandR proceeds in two steps. First, a preliminary prediction is generated using retrievals with similar input only. Then, a new set of exemplars is retrieved based on a relevance measure that balances the similarity of the inputs and the similarity of the preliminary prediction and the exemplar output. Figure 1 provides an overview of the method.

We evaluate GandR in the setting of task-oriented semantic parsing, a core component of widely used virtual assistants. We show that similarity in output space provides a complementary signal to input similarity, yielding retrievals that prove more informative for the model. Moreover, for many structured prediction tasks the output space is more structured than the free-form input text, so that simple, non-learned distance measures work well for outputs even when inputs are lexically dissimilar. Table 5 demonstrates an example where our proposed similarity function retrieves an example that is somewhat less similarly phrased but with more similar output, and the model produces a better prediction as a result. Finally, the model has the opportunity to verify that its preliminary predictions are valid outputs in the target language.

The proposed method strongly improves performance in low-resource settings for semantic

[*] Work primarily done at Google Research.

Proceedings of the 29th International Conference on Computational Linguistics, pages 4946–4951
October 12–17, 2022.

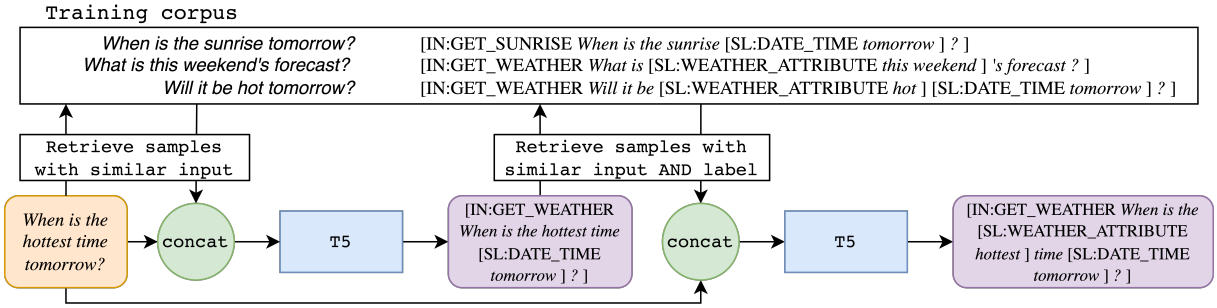

Figure 1: Overview of GandR. First, GandR generates a preliminary prediction using an input augmented with exemplars with similar inputs. Then, GandR retrieves exemplars based on a relevance measure balancing input similarity and similarity between the preliminary prediction and exemplar outputs, and generates a final prediction based on these exemplars.

parsing, achieving state of the art results for low-resource and transfer benchmarks in MTOP (Li et al., 2021) and TopV2 (Chen et al., 2020).

2 Method

We approach semantic parsing as a conditional language generation task and apply a T5 sequence-to-sequence model (Raffel et al., 2020) to predict a parse y given a query x. For each sample, we retrieve $K = 4$ relevant training exemplars sampled according to a relevance scoring function. We append the retrieved input-output pairs to the sample input and apply the T5 model to the augmented input to predict a parse output. In particular, let $(x'_1, y'_1), \ldots, (x'_K, y'_K)$ denote the retrieved input-output pairs, then the augmented input is

$$x' = x \;||\; x'_1 \;\&\; y'_1 \;||\; x'_2 \;\&\; y'_2 \;||\; \ldots$$

Our approach closely follows that of Pasupat et al. (2021), differing primarily in the choice of relevance function. During evaluation we retrieve the top K most relevant exemplars. During training, we sample retrievals according to a geometric distribution over the relevance score rank. In particular, the probability that we retrieve an exemplar is given by $p(1-p)^r$ where r is the rank of the exemplar according to relevance score and p is a temperature hyperparameter.

In Pasupat et al. (2021), the relevance score is given by the inner product of Universal Sentence Encoder (Cer et al., 2018) encodings of the candidate input and the sample input. We found that a simple TF-IDF (Ramos et al., 2003) similarity baseline achieves comparable or better results.

Our proposed approach, GandR, builds on the input-similarity baseline by constructing a hybrid similarity measure that takes into account not only the similarity between sample and candidate inputs, but also the similarity between the sequence predicted by the model and the candidate output. See Figure 1 for an overview. First, GandR generates a preliminary prediction using an input augmented with exemplars with similar inputs. Then, GandR retrieves exemplars based on a hybrid similarity measure over inputs and outputs, and generates a final prediction based on these exemplars.

Specifically, let $\hat{y}_i$ be preliminary prediction, then the proposed output similarity between samples i and j is given by the TF-IDF similarity between the predicted structure (in our case, the set of intents and slots) and the structure of the true parse y_j. Our proposed relevance score is a weighted sum of input and output similarity (see Appendix A for detailed description on how output similarity is computed):

$$R_{ij} = (1-\alpha)\text{TF-IDF}(x_i, x_j) + \alpha\text{TF-IDF}(\hat{y}_i, y_j)$$

2.1 Training

For simplicity, we train GandR in two stages. We start training with TF-IDF input relevance scoring, yielding model M_1. Model M_1 is used to generate GandR preliminary predictions during training and evaluation. We continue training M_1 for the remaining training steps, yielding M_2, which is used to generate final predictions augmented with retrievals from M_1. Note that this two-stage training is for convenience only, and it is possible to use a single set of weights M_{single} to generate preliminary and final GandR predictions. In that case, M_{single} needs to be trained with a mix of input-only and GandR retrieval augmentations to ensure it is able to use either effectively.

Model	$\text{MTOP}_{\text{boot}}$	MTOP_{1k}	$\text{MTOP}_{25\%}$	TOPv2_{W}	TOPv2_{R}
Reptile (Chen et al., 2020)				77.7	70.5
RAF (Shrivastava et al., 2022)				78.7	
CASPER (Pasupat et al., 2021)	73.3 / 83.9				
T5	72.9 / 83.3	62.8	78.5	79.2	68.8
T5 with input TF-IDF	74.9 / 84.5	67.2	79.4	79.9	71.0
GandR	**76.4 / 84.6**	**67.8**	**80.1**	**80.5**	**71.7**

Table 1: Results on semantic parsing benchmarks. We report the percentage exact match between true and predicted labels as sequences. Results are on test set for all benchmarks except $\text{MTOP}_{\text{boot}}$, where we report on dev to remain comparable with CASPER.

3 Related Work

Sequence-to-sequence models (Raffel et al., 2020; Lewis et al., 2020) have achieved state-of-the-art performance on task-oriented semantic parsing (Li et al., 2021; Chen et al., 2020; Aghajanyan et al., 2020) as well as other structured prediction tasks (Raffel et al., 2020). The general approach is to pre-train on language modeling and perform fine-tuning on the specific domain of interest.

Several works augment the input with retrieved exemplars from the training data, with differing methods for selecting informative examples. Pasupat et al. (2021) and Gupta et al. (2021) retrieve exemplars with similar input encodings from a pre-trained neural encoder, evaluating on semantic parsing. Wang et al. (2022) retrieves examplars for which the input has high BM25 similarity with the sample input, with good performance on language generation. We adopt a similar approach with TF-IDF similarity as a baseline for semantic parsing.

Black et al. (2021) and Das et al. (2021) learn dense retrievers in the spirit of Karpukhin et al. (2020), providing another path to incorporate label information for retrieval. Izacard et al. (2022) proposes other methods to fine-tune a dense retriever. These approaches require training a separate model specifically for retrieval, possibly with additional learning signal. In contrast, we employ a sparse similarity measure over model predictions that are produced incidentally in the course of fine-tuning the main model.

Selecting relevant training exemplars is also important for in-context prompting (Liu et al., 2021b). Similar to related fine-tuning literature, work in this direction uses either a pre-trained (Gao et al., 2020) or fine-tuned (Liu et al., 2021a) sentence encoder to retrieve exemplars.

Dataset	#Train	#Dev	#Test
MTOP	15667	2234	4385
MTOP_{1k}	1096	2234	4385
$\text{MTOP}_{25\%}$	3916	2234	4385
TOPv2_{S}	83703	11967	27336
TOPv2_{W}	176	147	5682
TOPv2_{R}	493	337	5767

Table 2: Dataset statistics: the number of examples per dataset and split.

Model	MTOP	TOPv2_{S}
RAF		87.1
CASPER	86.4	
T5	85.7	86.9
T5 input TF-IDF	86.4	87.0
GandR	86.4	87.0

Table 3: Performance on high-resource settings.

4 Experiments

4.1 Setup

We evaluate GandR and baselines on semantic parsing benchmarks MTOP (Li et al., 2021) and TOPv2 (Chen et al., 2020), focusing on low-resource and transfer settings. MTOP is a medium-sized semantic parsing dataset used in Pasupat et al. (2021), for which we evaluate on the *domain bootstrapping* setting in which one of the domains is limited to a very small amount of training data. We also evaluate on low-resource settings MTOP_{1k} and $\text{MTOP}_{25\%}$ in which we randomly sample 1k and 25% of training samples, respectively. TOPv2 is centered on transfer to low-resource domains: models are trained on a set of high resource-domains denoted as TOPv2_{S} and then fine-tuned on low-resource *Weather* and *Reminder* domains[1], denoted

[1]We are using 25 SPIS low resource split from Chen et al. (2020).

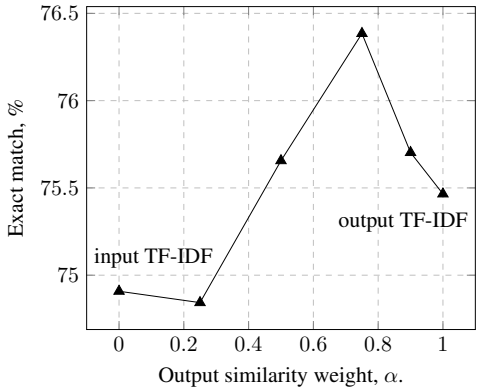

Figure 2: Performance on $\text{MTOP}_{\text{boot}}$ development set as a function of output similarity weight α.

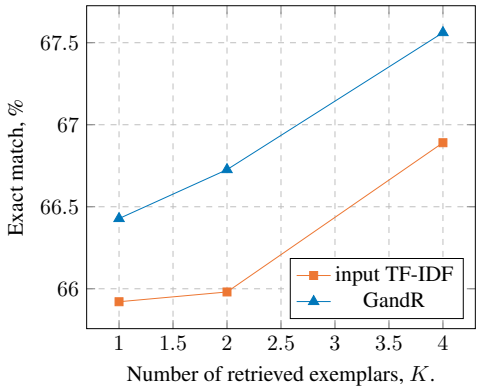

Figure 3: Performance on the development set of $\textbf{MTOP}_{\text{1k}}$ as a function of the number of retrieved exemplars K.

as TOPV2_{W} and TOPV2_{R}. We show the sizes of datasets and splits in Table 2. See Appendix B for details on the training setup.

4.2 Main results

The results of our primary experiments are shown in Table 1. We find that input TF-IDF is a strong baseline, rivaling or improving over prior work. Further, GandR retrieval outperforms all baselines, setting the state of the art on evaluated settings.

4.3 Ablations and discussion

Retrieval is less important for high-resource settings Table 3 shows results on the high-resource full MTOP and TOPv2 datasets. In higher-resource settings, augmenting the input with exemplars appears to be both less effective and less sensitive to retrieval method, with almost identical results among methods with and without retrieval for the highest resource TOPv2 dataset.

Retriever	$\text{MTOP}_{\text{boot}}$	TOPv2_{W}	TOPv2_{R}
input TF-IDF	35.9	55.1	20.1
output TF-IDF	70.3	74.8	53.7
GandR	70.0	68.7	52.5

Table 4: Template recall@K=4 on the development sets for $\text{MTOP}_{\text{boot}}$, TOPv2_{W} and TOPv2_{R}.

Using hybrid similarity leads to better retrieval quality Figure 2 displays $\text{MTOP}_{\text{boot}}$ performance as a function of TF-IDF output weight α. The results demonstrate that input and output similarity signals are strongly complementary. See Figure 4 and Figure 5 in the Appendix for similar experiments on the MTOP_{1k} and $\text{MTOP}_{\text{25\%}}$ benchmarks.

Considering output similarity leads to higher template recall Following Pasupat et al. (2021), we compute *template recall@K* as a proxy metric for retrieval. This measure corresponds to the proportion of evaluation samples for which at least one of the top K retrievals has the same template (identical intents and slots) as the gold parse. Results show (Table 4) that considering output as well as input similarity increases template recall. We note that output TF-IDF has similar or higher template recall than GandR even though it has lower performance. Ultimately, template recall is only a proxy, and we are really interested in retrieval *informativeness*; GandR's performance shows that balancing input similarity and template recall leads to exemplars that are most helpful for the model.

Hybrid similarity helps across different numbers of retrieved exemplars Figure 3 shows that GandR outperforms input TF-IDF similarity on MTOP_{1k} when we retrieve different number of exemplars: 1, 2 or 4. See Figure 6 in the Appendix for a similar experiment on the $\text{MTOP}_{\text{25\%}}$ benchmark.

4.4 Error analysis

The primary motivation for GandR is that hybrid similarity leads to more informative exemplars. Informativeness can only be objectively measured through model performance, but our motivating intuition appears to be borne out by samples in the data. We observe a number of different cases for which output or hybrid-similarity retrieval can help. Table 5 shows an example of a case for which input TF-IDF retrieves an irrelevant example with lexical overlap, while GandR retrieves an example with both lexical and parse overlap, leading to a

Input sample

x: *Could you connect me to the Musicals group*
y: [IN:CREATE_CALL [SL:GROUP *Musicals*]]

Training sample with similar input

x_1: *musicals in windham this weekend*
y_1: [IN:GET_EVENT [SL:CATEGORY_EVENT *musicals*] [SL:LOCATION *windham*] [SL:DATE_TIME *this weekend*]]
$\hat{y}$: [IN:CREATE_CALL [SL:CONTACT *me*] [SL:GROUP *Musicals*]]

Training sample with similar input and label

x_1: *can you please send text to the development group*
y_1: [IN:SEND_MESSAGE [SL:GROUP *development*]]
$\hat{y}$: [IN:CREATE_CALL [SL:GROUP *Musicals*]]

Table 5: Input TF-IDF retrieves an exemplar with lexical overlap ('musicals') that is not relevant to the sample. The GandR retrieval balances lexical and label similarity and leads to a correct prediction. Single representative exemplar out of 4 displayed for each method. See Table 9 in the Appendix for all retrieved exemplars.

correct prediction. Using preliminary predictions for retrieval can also allow the model to verify whether its predictions are correct. A common simple case when this can help is if the model generates a prediction that is dissimilar to any samples in the training set in which case the model may reconsider whether that prediction is correct (Table 7). Considering output similarity does come with tradeoffs. Table 8 demonstrates a situation where output similarity distracts the model away from a lexically similar and informative exemplar and the model is wrong as a result.

5 Conclusion

We propose GandR, a new method for structured prediction that generates a preliminary prediction, retrieves training exemplars with similar outputs (and similar inputs), and augments the input with the retrieved exemplars to generate a final prediction. We demonstrate that using output similarity yields improvements for semantic parsing in low-resource settings, achieving state of the art results on several semantic parsing benchmarks.

Acknowledgments

We thank William Cohen, Nicholas Fitzgerald and Luke Vilnis for insightful discussions and reviewers for their feedback. This work is partially supported by NSF Awards IIS-1513966/ 1632803/1833137, CCF-1139148, DARPA Awards#: FA8750-18-2-0117, FA8750-19-1-0504, DARPA-D3M - Award UCB-00009528, Google Research Awards, gifts from Facebook and Netflix, and ARO# W911NF-12-1-0241 and W911NF-15-1-0484.

References

Armen Aghajanyan, Jean Maillard, Akshat Shrivastava, Keith Diedrick, Michael Haeger, Haoran Li, Yashar Mehdad, Veselin Stoyanov, Anuj Kumar, Mike Lewis, and Sonal Gupta. 2020. Conversational semantic parsing. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing, EMNLP 2020, Online, November 16-20, 2020*, pages 5026–5035. Association for Computational Linguistics.

Sid Black, Leo Gao, Phil Wang, Connor Leahy, and Stella Biderman. 2021. Gpt-neo: Large scale autoregressive language modeling with mesh-tensorflow. 58.

Daniel Cer, Yinfei Yang, Sheng-yi Kong, Nan Hua, Nicole Limtiaco, Rhomni St. John, Noah Constant, Mario Guajardo-Cespedes, Steve Yuan, Chris Tar, Yun-Hsuan Sung, Brian Strope, and Ray Kurzweil. 2018. Universal sentence encoder. *CoRR*, abs/1803.11175.

Xilun Chen, Asish Ghoshal, Yashar Mehdad, Luke Zettlemoyer, and Sonal Gupta. 2020. Low-resource domain adaptation for compositional task-oriented semantic parsing. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing, EMNLP 2020, Online, November 16-20, 2020*, pages 5090–5100. Association for Computational Linguistics.

Rajarshi Das, Manzil Zaheer, Dung Thai, Ameya Godbole, Ethan Perez, Jay Yoon Lee, Lizhen Tan, Lazaros Polymenakos, and Andrew McCallum. 2021. Case-based reasoning for natural language queries over knowledge bases. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing, EMNLP 2021, Virtual Event / Punta Cana, Dominican Republic, 7-11 November, 2021*, pages 9594–9611. Association for Computational Linguistics.

Tianyu Gao, Adam Fisch, and Danqi Chen. 2020. Making pre-trained language models better few-shot learners. *arXiv preprint arXiv:2012.15723*.

Vivek Gupta, Akshat Shrivastava, Adithya Sagar, Armen Aghajanyan, and Denis Savenkov. 2021. RETRONLU: retrieval augmented task-oriented semantic parsing. *CoRR*, abs/2109.10410.

Gautier Izacard, Patrick Lewis, Maria Lomeli, Lucas Hosseini, Fabio Petroni, Timo Schick, Jane Dwivedi-Yu, Armand Joulin, Sebastian Riedel, and Edouard Grave. 2022. Few-shot learning with retrieval augmented language models.

Vladimir Karpukhin, Barlas Oguz, Sewon Min, Patrick S. H. Lewis, Ledell Wu, Sergey Edunov, Danqi Chen, and Wen-tau Yih. 2020. Dense passage retrieval for open-domain question answering. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing, EMNLP 2020, Online, November 16-20, 2020*, pages 6769–6781. Association for Computational Linguistics.

Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Veselin Stoyanov, and Luke Zettlemoyer. 2020. BART: denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, ACL 2020, Online, July 5-10, 2020*, pages 7871–7880. Association for Computational Linguistics.

Haoran Li, Abhinav Arora, Shuohui Chen, Anchit Gupta, Sonal Gupta, and Yashar Mehdad. 2021. MTOP: A comprehensive multilingual task-oriented semantic parsing benchmark. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume, EACL 2021, Online, April 19 - 23, 2021*, pages 2950–2962. Association for Computational Linguistics.

Jiachang Liu, Dinghan Shen, Yizhe Zhang, Bill Dolan, Lawrence Carin, and Weizhu Chen. 2021a. What makes good in-context examples for gpt-3? *arXiv preprint arXiv:2101.06804*.

Pengfei Liu, Weizhe Yuan, Jinlan Fu, Zhengbao Jiang, Hiroaki Hayashi, and Graham Neubig. 2021b. Pre-train, prompt, and predict: A systematic survey of prompting methods in natural language processing. *arXiv preprint arXiv:2107.13586*.

Panupong Pasupat, Yuan Zhang, and Kelvin Guu. 2021. Controllable semantic parsing via retrieval augmentation. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing, EMNLP 2021, Virtual Event / Punta Cana, Dominican Republic, 7-11 November, 2021*, pages 7683–7698. Association for Computational Linguistics.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J. Liu. 2020. Exploring the limits of transfer learning with a unified text-to-text transformer. *J. Mach. Learn. Res.*, 21:140:1–140:67.

Juan Ramos et al. 2003. Using tf-idf to determine word relevance in document queries. In *Proceedings of the first instructional conference on machine learning*, volume 242, pages 29–48. Citeseer.

Akshat Shrivastava, Shrey Desai, Anchit Gupta, Ali Elkahky, Aleksandr Livshits, Alexander Zotov, and Ahmed Aly. 2022. Retrieve-and-fill for scenario-based task-oriented semantic parsing. *CoRR*, abs/2202.00901.

Shuohang Wang, Yichong Xu, Yuwei Fang, Yang Liu, Siqi Sun, Ruochen Xu, Chenguang Zhu, and Michael Zeng. 2022. Training data is more valuable than you think: A simple and effective method by retrieving from training data. *CoRR*, abs/2203.08773.

Coarse-to-Fine: Hierarchical Multi-task Learning for Natural Language Understanding

Zhaoye Fei[1,4†], **Yu Tian**[1†], **Yongkang Wu**[1†], **Xinyu Zhang**[1†], **Yutao Zhu**[2], **Zheng Liu**[1], **Jiawen Wu**[4], **Dejiang Kong**[1], **Ruofei Lai**[1], **Zhao Cao**[1*], **Zhicheng Dou**[3] **and Xipeng Qiu**[4]
[1]Huawei Poisson Lab, China
[2]University of Montreal, Montreal, Quebec, Canada
[3]Gaoling School of Artificial Intelligence, Renmin University of China, Beijing, China
[4]School of Computer Science, Fudan University, Shanghai, China
{wuyongkang7,zhangxinyu35,caozhao1}@huawei.com

Abstract

Generalized text representations are the foundation of many natural language understanding tasks. To fully utilize the different corpus, it is inevitable that models need to understand the relevance among them. However, many methods ignore the relevance and adopt a single-channel model (a coarse paradigm) directly for all tasks, which lacks enough rationality and interpretation. In addition, some existing works learn downstream tasks by stitches skill block (a fine paradigm), which might cause irrational results due to its redundancy and noise. In this work, we first analyze the task correlation through three different perspectives, *i.e.*, data property, manual design, and model-based relevance, based on which the similar tasks are grouped together. Then, we propose a hierarchical framework with a coarse-to-fine paradigm, with the bottom level shared to all the tasks, the mid-level divided to different groups, and the top-level assigned to each of the tasks. This allows our model to learn basic language properties from all tasks, boost performance on relevant tasks, and reduce the negative impact from irrelevant tasks. Our experiments on 13 benchmark datasets across four natural language understanding tasks demonstrate the superiority of our method.

1 Introduction

Pre-trained language models have achieved great success on various natural language processing (NLP) tasks. Meanwhile, *pre-train-then-fine-tuning* has gradually become the mainstream paradigm (Devlin et al., 2019; Liu et al., 2019b; Yang et al., 2019). The pre-training process aims to learn a general language representation from the large-scale corpus. Such representations can be further fine-tuned on downstream datasets and perform specific tasks. Though great performance has been achieved, it is costly to fine-tune and save independent representation for each task. Therefore, researchers propose several multi-task learning methods (Phang et al., 2018; Liu et al., 2019a; Clark et al., 2019b) using a single-channel model (a coarse paradigm) to solve multiple tasks.

Inspired by human learning, multi-task learning believes that tasks can interact and boost each other. Therefore, to obtain a more robust representation that can handle a variety of tasks, a straightforward idea is to fine-tune a pre-trained model on many tasks simultaneously. Numerous previous studies have been conducted along this path. For example, MT-DNN (Liu et al., 2019a) uses a transformer-based model as a shared encoder and trains it on multiple downstream tasks (as shown in the left side of Figure 1). By this coarse paradigm, the representation model can be more generalized and robust. Although it has been observed that some combinations of tasks yield improvements, this is not always the case. Researchers (Aribandi et al., 2021) have also found that the impact between tasks is a double-edged sword, meaning that some tasks may also hurt others. Indeed, modeling heterogeneous tasks often require distinct representation spaces. For example, naively training natural language inference (NLI) tasks with different hypothesis types together can lead to performance declines.

Some recent work proposes sparsely activating multiple modules for different tasks (Tang et al., 2022) to mitigate the negative effects across tasks. These fine paradigms activate distinct modules according to predefined skills for learning downstream tasks (as shown in the middle part of Figure 1). However, the number of parameters is also multiplied when multiple modules are activated for a task. This problem becomes even more severe when large pre-trained language models are applied.

To address the aforementioned problems, we

[†]These authors contributed equally to this work.
[*]Corresponding author.

Proceedings of the 29th International Conference on Computational Linguistics, pages 4952–4964
October 12–17, 2022.

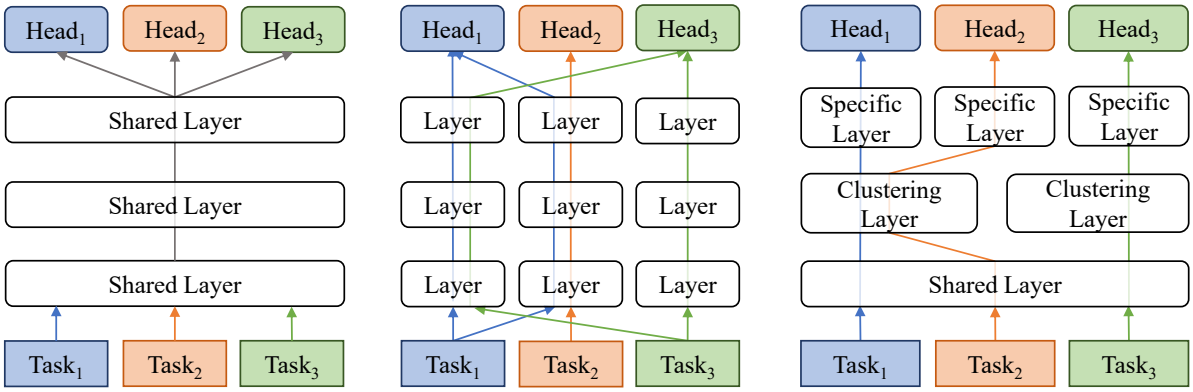

(1) Fully Sharing (Coarse) (2) Partial Sharing (Fine) (3) Hierarchical Sharing (Coarse-to-Fine)

Figure 1: The illustration of hard sharing structure, partial sharing structure, and our hierarchical sharing structure. The three tasks and the corresponding activated paths are in different colors. In our approach, the first and second are two relevant tasks, so they share the same task-clustering layers. Finally, they are fed into different task-specific layers.

first conduct a correlation analysis on four natural language understanding (NLU) categories (a total of 13 datasets) and find that some tasks can complement one another, while others cannot. Based on these observations, we propose three measures (*i.e.*, data property, manual design, and model-based relevance) to categorize the tasks into different groups.

Then, inspired by recent studies (Kovaleva et al., 2019; Rogers et al., 2020) that different layers learn information in different levels, we design a novel hierarchical sharing framework, dubbed HMNet, a coarse-to-fine multi-task learning paradigm for natural language understanding (as shown in the right side of Figure 1). Our method is a coarse-to-fine paradigm, where the layers are divided from the bottom up into shared, task-clustering, and task-specific levels. We design the bottom layers as shared, which are optimized by all tasks. Thereafter, we employ distinct layers for different groups obtained in the previous step. In this way, the relevant tasks grouped in the same cluster can boost each other, while the negative impact from tasks in other groups can be avoided. The top layers are totally separated for distinct tasks, so that some task-specific information can be well-captured. It is worth noting that each task only activates a single module in each layer, so our model does not require additional parameters during inference. Extensive experiments on 13 datasets show that our method can achieve better performance on most NLU tasks, demonstrating its superiority over existing hard sharing or partial sharing structures.

Our main contributions are three-fold:

(1) We perform a series of correlation analyses (*i.e.*, data property, manual design, and model-based relevance) on 13 NLU tasks, which sheds light on the positive and negative effects among the different tasks.

(2) We design a novel hierarchical sharing framework, dubbed HMNet, a coarse-to-fine multi-task learning paradigm for natural language understanding. It can better leverage the positive interactions among different tasks while reducing the negative influence.

(3) We conduct experiments on 13 commonly used NLU datasets and validate the effectiveness of our proposed method. The results demonstrate that our framework achieves highly competitive performance while saving over 34% parameters than the partial sharing structure.

2 Related Work

In this section, we briefly introduce some Transformer-based pre-trained models and recent work on multi-task learning.

2.1 Transformer-based Pre-trained Models

Transformer is a neural structure consisting of multiple stacked self-attention modules (Devlin et al., 2019; Liu et al., 2019b; Yang et al., 2019). With the bidirectional attention mechanism, the model can capture contextual information from both sides effectively. BERT (Devlin et al., 2019) proposes a masked language modeling objective to pre-train

a Transformer encoder, and achieves dramatic performance on several natural language understanding tasks (Bowman et al., 2015; Williams et al., 2018a; Wang et al., 2019b; Zhu et al., 2021a,b). Since then, the *pre-train-then-fine-tuning* paradigm has gradually become mainstream. Researchers propose various new pre-training strategies to facilitate the model in several aspects (Yang et al., 2019; Lewis et al., 2020; Raffel et al., 2020). For example, in order to obtain a more robust model, RoBERTa (Liu et al., 2019b) adapts several tailored training strategies and employs more data. There are also many methods extending the pre-training on Transformer decoders. GPT-2 (Radford et al., 2019) is a decoder-only structure that is pre-trained in accordance with the language generation objectives. It also performs exceptionally well on many text generation tasks. Though pre-trained models can be fine-tuned for distinct tasks, it is costly to maintain a separate model for each task, especially when the model is huge.

2.2 Multi-task Learning

Multi-task learning is an integrated learning method in which multiple tasks share the same structure for training simultaneously. It can enhance the generalization and performance of each task. Consequently, multi-task learning can also be applied to pre-trained language models. A typical practice is to fine-tune the pre-trained language models by conducting multiple tasks concurrently (Liu et al., 2019a). It is reported, however, that not all tasks can boost each other, and that noise may also be introduced (Aribandi et al., 2021). Therefore, it is essential to analyze the relationship between tasks before training them jointly. Some recent work proposes addressing this issue by sparsely activating distinct modules for different tasks (Tang et al., 2022). This allows the modules to be trained by relevant tasks. Unfortunately, these approaches usually rely on predefined activation paths, and they have to activate multiple modules in order to achieve high performance during the inference stage. This undoubtedly increases the delay and cost of model application.

The main differences between our method and others are: (1) We analyze the task correlation, and use it to group the tasks. Relevant tasks within the same group will be used to fine-tune the same modules, while irrelevant tasks will not interfere with one another. (2) Our model only activates a single module in each layer for each task, therefore it has the same number of parameters as a single model.

3 Proposed Method

3.1 Overview

The overview of our method is illustrated in the right side of Figure 1. In general, there are three different kinds of layers in our structure: shared layers, task-clustering layers, and task-specific layers. The shared layers are in the bottom and optimized by all tasks. The middle part is the task-clustering layers. Tasks that are classified as relevant (introduced in the next section) will optimize the same group of layers. The top layers are task-specific, meaning that each task has its own module.

Our method adopts a coarse-to-fine paradigm. In this manner, the model can obtain generalized text representations in shared layers (a coarse-grained manner) and effectively interact and learn according to task correlation in the task-clustering layer (a fined-grained manner), then send it to the task-specific layer for specific task training.

3.2 Task Relevance Analysis

Existing multi-task learning framework assumes that all tasks can facilitate each other. However, researchers have also reported the negative effect caused by irrelevant tasks (Aribandi et al., 2021). To investigate the influence of each task, we devise three methods from different perspectives to measure the task relevance.

Data Property Since all tasks we study are based on natural language texts, we first analyze the relevance between tasks through the property of their data. Specifically, we consider the syntactic information and employ vocabulary co-occurrence to measure task relevance. Formally, given datasets of two tasks denoted as $\mathcal{D}_\text{s}$ and $\mathcal{D}_\text{t}$, where s and t stand for *source* and *target*, we compute the vocabulary co-occurrence as follows:

$$r_\text{cs} = |V_\text{s} \cap V_\text{t}|/|V_\text{s}|, \quad (1)$$

$$r_\text{ct} = |V_\text{s} \cap V_\text{t}|/|V_\text{t}|, \quad (2)$$

where V_s and V_t are the vocabulary of the source and target datasets, respectively, r_cs measures the ratio of words in the source dataset that are shared with the target datasets, and r_ct represents the opposite. Intuitively, a higher value of r_cs indicates a higher impact of the target task on the

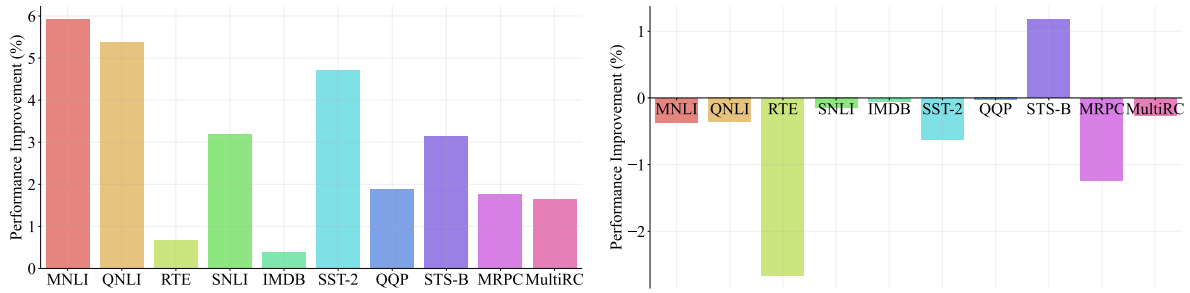

Figure 2: Influence of other tasks on the BoolQ dataset (left) and influence of the BoolQ dataset on others (right).

source task. Note that this metric also considers the data size, as a larger dataset usually contains more words. For example, MNLI (Williams et al., 2018b), as the largest dataset, has a large vocabulary co-occurrence with others. It can provide some basic language knowledge for other tasks. Conversely, WNLI (Wang et al., 2019b) is a small dataset with only hundreds of samples, and it may have limited impacts on other tasks. Due to the space limitation, we show the entire results in Appendix A.

Manual Design Inspired by a recent work that designs various paths for different tasks (Tang et al., 2022), we also examine the tasks artificially. Based on the manually designed task purpose, we divide them into four groups. *i.e.*, Natural language inference, sentiment classification, similarity and paragraphing, and question answering. We assume tasks that have the same purpose supply the same systematic information for models. For instance, the NLI tasks generally rely on the models' judgment according to the deep semantics information between premise and hypothesis, while the question answering tasks ask the model to infer the answer from the passage according to the question. These tasks and their category will be reported in Section 4.1.

Model-based Relevance In addition, to manually analyze the task or data, we further propose measuring task relevance using neural networks. Previous studies have demonstrated the advantages of multi-task learning (Liu et al., 2019a), notably that relevant tasks can enhance each other's performance. Consequently, task relevance can be inferred by comparing the performance difference between training two tasks independently and training them jointly. Concretely, for a source and a target task, we fine-tune two models (we use BERT in our experiments) respectively for them. Their average performance is denoted as f_{s} and f_{t}. Afterward, we fine-tune another model (BERT) for the two tasks through standard multi-task learning. The corresponding performance is denoted as f_{js} and f_{jt}. Finally, we can compute the task relevance as the performance improvement:

$$r_{\mathrm{ms}} = (f_{\mathrm{js}} - f_{\mathrm{s}})/f_{\mathrm{s}}, \quad (3)$$

$$r_{\mathrm{mt}} = (f_{\mathrm{jt}} - f_{\mathrm{t}})/f_{\mathrm{t}}, \quad (4)$$

where r_{ms} reflects the influence of the target task on the source task, while r_{mt} represents the opposite. We calculate the relevance for all tasks, and the results about BoolQ as an example in Figure 2 to highlight the asymmetrical relevance between tasks. The complete form is shown in Appendix A. In Figure 2, we find that most tasks have a positive impact on BoolQ, even though the data property and the purpose of these tasks are very different. BoolQ, on the other hand, harms other tasks. One possible reason is learning for QA needs more complicated information and learning on other tasks can supply this information to improve the performance of BoolQ. But as for other tasks, the information obtained by learning in BoolQ like noise damages the learning of themselves.

Since the asymmetrical relevance between task pairs, we consider the fully shared model is harmful to some tasks in multi-task learning. Inspired by Rogers et al., we consider that the improvement of word-level information training in the low layer occurs a positive impact mostly, whereas the more negative impact is sourced from the damage of the deep semantic learning in the middle and top layers. As a result, we propose a hierarchical sharing method for multi-task learning. It is based on a hierarchical sharing of the task with different relevance to leverage more positive interaction and reduce the negative impact of multi-task learning. More detail about the structure is described in the next section.

3.3 HMNet

We propose a hierarchical multi-task learning framework (HMNet) based on task relevance. It is built on Transformer (Vaswani et al., 2017), which has been widely applied in NLP tasks.[1] We omit the details of Transformer and refer readers to the original paper. As shown in the right side of Figure 1, HMNet has three different kinds of Transformer layers from the bottom up. All tasks will first pass the shared layers, then, each task will go through task-clustering layers according to their task group (introduced later). Finally, each task has its own task-specific layer, and the associated head is used to accomplish the task.

In the following paragraphs, we first introduce the details of the layers, followed by the task clustering process.

Shared layers The shared layers are stacked in the bottom of HMNet. We design bottom layers as shared since it has been demonstrated that they capture low-level semantic or structural information in existing pre-trained models (Jawahar et al., 2019). We believe that such information is universally contained in all texts, so all tasks collaborate to optimize the shared layers.

Task-clustering layers Based on our task relevance analysis, we cluster the tasks into various groups. For tasks within the same cluster, the same (set of) clustering layers will be optimized. As illustrated on the right side of Figure 1, the first two tasks are grouped into one cluster, so they optimize the first set of clustering layers. On the contrary, only the third task tunes the second set of clustering layers. With the task-clustering layers, relevant tasks can optimize the same set of parameters, enabling the sharing of their knowledge. In the meantime, the irrelevant tasks can be isolated, thereby eliminating the noise.

Task-specific layers Task-specific layers are in the top of our HMNet. According to recent studies (Jawahar et al., 2019), the top layers of pre-trained language models typically learn task-specific knowledge. Therefore, we separate all tasks and let them have their own Transformer layers. There is also a head associated with each task-specific layer to accomplish the task. For example, a linear classifier with an activation function is often applied for classification tasks.

Clustering Process In our task relevance analysis, we propose three methods to measure the task relevance. Among them, the manual design can directly group the tasks into different clusters. For the other two methods, we employ an unsupervised clustering method, *i.e.*, k-means. Particularly, given a source task S, and n target tasks $\{T_1, \cdots, T_n\}$, we can compute the relevance scores $\{r_{\text{ST}_1}, \cdots, r_{\text{ST}_n}\}$ by Equation (2) or (4).[2] Then, we treat the n scores as features and apply k-means clustering algorithm. Finally, the tasks can be grouped into k clusters. The grouping results are given in the Appendix A.

Remark Different from traditional multi-task learning that shares all layers among all tasks, our HMNet only shares layers at the bottom (a coarse-grained manner). The higher layers are shared by only relevant tasks or used alone(a fined-grained manner). Since the layers close to specific tasks are separated, our method can alleviate the contradiction between tasks. On the other hand, each task only activates one channel at each layer, so the total parameters are comparable with a single-channel model. This is much more efficient than sparsely activated structures.

3.4 Optimization

HMNet is based on a multi-layer Transformer, which is similar to existing pre-trained language models, such as BERT (Devlin et al., 2019). Therefore, we use BERT as the backbone model to initialize the parameters in each layer. Then, each task has its own path (as described earlier), and we can use it to fine-tune the corresponding layers. The training process our method is summarized in Algorithm 1. In each epoch, a mini-batch b_t is packed, and the HMNet is updated by the path for the dataset $\mathcal{D}_i$.

4 Experiment

4.1 Datasets and Evaluation Metrics

We conduct experiments on 13 NLU tasks and compare the performance of our HMNet with other baselines. These tasks can be grouped into four categories:

[1] Other structures, such as convolutional or recurrent networks, may also be compatible with our method. This is left for future work.

[2] Since the task correlations are asymmetric, scores such as r_{TS} may also work.

Algorithm 1: Training Process

Initialize model parameters from pre-trained BERT;
Set the max number of training epoch E_m;
Prepare data
for i *in* $1, 2, \cdots, N$ **do**
  Pack the dataset $\mathcal{D}_i$ into minibatch B_i;
end
Multi-task learning
for epoch t in $1, 2, \cdots, E_m$ **do**
  Merge all the datasets: $B = B_1 \cup B_2 \cdots \cup B_N$;
  Shuffle B;
  for b_t *in* B **do**
    // b_t *is a mini-batch of task* t;
    Feed b_t into shared layers $\rightarrow \mathbf{b}_t^s$;
    Feed $\mathbf{b}_t^s$ into task-clustering layers $\rightarrow \mathbf{b}_t^c$;
    Feed $\mathbf{b}_t^c$ into task-specific layers $\rightarrow \mathbf{b}_t^t$;
    Feed $\mathbf{b}_t^t$ into task-specific head;
    Compute loss and gradient;
    Update model;
  end
end

(1) **Natural Language Inference (NLI):** These tasks aim to determine whether a *hypothesis* is entailed, contradicted, or undetermined given a *premise*. They require the model to measure the logical coherence between two sentences. We select six datasets: MNLI (Williams et al., 2018b), QNLI, RTE, WNLI (Wang et al., 2019b), CB (Wang et al., 2019a), and SNLI (Bowman et al., 2015).

(2) **Sentiment Classification:** These tasks ask the model to classify the sentiment polarity of a sentence (*i.e.*, positive or negative). We use IMDB (Maas et al., 2011) and SST-2 (Socher et al., 2013) datasets.

(3) **Similarity and Paraphrase:** These tasks aim at determining whether two sentences have the same or similar meaning. The semantic relationship between two sentences is vital in these tasks. We choose three commonly used datasets: QQP (Wang et al., 2019b), STS-B (Cer et al., 2017), and MRPC (Dolan and Brockett, 2005).

(4) **Question Answering:** This task requires the model to answer a question by reasoning on a given paragraph. BoolQ (Clark et al., 2019a) and MultiRC (Khashabi et al., 2018) datasets are used.

Detailed statistics of benchmark datasets are mentioned in Appendix B.

4.2 Baseline

We compare with our HMNet with three other training methods:

Single-Task fine-tuning: We fine-tune a pre-trained BERT for each task independently. As a result, 13 models are obtained in total.

Multi-Task fine-tuning: Following MT-DNN (Liu et al., 2019a), we use the pre-trained BERT as a shared encoder and add a task-special head for each task. During fine-tuning, all parameters are optimized by all tasks jointly. With multi-task learning, only one model needs to be trained and saved.

SkillNet-style fine-tuning: This is a sparsely activated multi-task learning structure proposed by Tang et al. (2022). They design seven basic skills, such as getting the semantic meaning of a sequence and understanding a question. Then, they divide typical NLU tasks into five categories, each requiring a unique skill combination. The skill module is implemented by adding multiple FFN layers into the original Transformer structure. This model is also initialized by a pre-trained BERT. However, as each task activates multiple skill modules, compared with our method and other baselines, on average 1.0× more parameters (*i.e.*, 223M vs. 110M) are used in inference.

4.3 Implementation details

For a fair comparison, all methods are initialized by the `bert-base-uncased` checkpoint with the same seed. The batch size is 32, the max length of sequence is 512, and the initial learning rate is 2e-5, which is linearly decayed. AdamW (Loshchilov and Hutter, 2019) optimizer is applied. We train all methods for three epochs.

For the NLI and text similarity & paraphrase tasks, similar to the vanilla BERT, we concatenate the sentence pair by adding a separator token `[SEP]` and a head token `[CLS]`. For the sentiment classification task, the input is a single sentence, so we only add a head token `[CLS]`. For the QA task, we concatenate the question, passage, and answer, then separate them by two `[SEP]` tokens. A head token `[CLS]` is also added at the beginning of the sequence. All tasks are performed in similar ways, namely processing the embedding of the `[CLS]` token by a linear layer with a softmax activation function, and output the probability of each category. The default setting of HMNet has eight shared layers, two task-clustering layers, and two task-specific layers.

4.4 Experimental Results

Table 1 shows the results on all datasets. In general, our HMNet performs better than other baselines on most datasets and achieves the best result in terms of the average score. This clearly demonstrates the

	Metric	Single-task	Multi-task	SkillNet	HMNet$_{\mathrm{d}}$	HMNet$_{\mathrm{md}}$	HMNet$_{\mathrm{m}}$
NATURAL LANGUAGE INFERENCE							
MNLI (m/mm)	Acc.	**85.0**/84.6	84.5/84.8	84.5/84.6	84.8/84.6	84.7/**85.2**	84.8/84.9
QNLI	Acc.	**91.6**	90.9	90.7	90.8	91.0	91.1
RTE	Acc.	66.1	79.7	77.6	73.3	79.4	**81.2**
WNLI	Acc.	56.3	56.3	56.3	56.3	56.3	56.3
CB	Acc.	67.8	80.3	85.7	**87.5**	82.1	82.1
SNLI	Acc.	91.0	91.3	91.1	91.0	**91.4**	91.1
SENTIMENT CLASSIFICATION							
IMDB	Acc.	93.9	93.9	**94.1**	93.8	93.9	93.9
SST-2	Acc.	92.3	93.1	92.5	**93.2**	92.3	93.0
SIMILARITY AND PARAPHRASE							
QQP	Acc.	90.9	90.7	**91.0**	90.9	**91.0**	90.9
STS-B	Corr.	85.8	85.5	86.0	86.9	**87.5**	87.4
MRPC	Acc.	83.3	81.4	85.7	88.2	**90.4**	88.5
QUESTION ANSWERING							
BoolQ	Acc.	71.4	77.9	**80.7**	79.0	79.1	80.3
MultiRC	$\mathrm{F1_a}$	65.2	68.4	68.1	**68.9**	66.7	68.4
Average Score	-	80.4	82.8	83.5	83.5	83.7	**84.0**
# Params Activated	-	110M	110M	>166M	110M	110M	110M
# Overall Params	-	110M	110M	450M	231M	240M	231M

Table 1: Results of all methods on 13 datasets. The best results are in **bold**. HMNet is our proposed method, and the subscript stands for three task relevance metrics, *i.e.*, data property (d), manual design (md), and model-based relevance (m). During inference, SkillNet has to activate at least two channels, thus the number of parameters is larger than 166M. In contrast, our HMNet only activates one channel at each layer, which requires less parameters.

superiority of our proposed HMNet. We further have the following observations:

(1) Comparing the performance between single-task and multi-task learning, it is evident that the latter can bring improvement for most tasks. Furthermore, this enhancement is related to the size of the datasets. Specifically, for the large dataset, such as MNLI and QQP (both of which have more than 300k training samples), the single-task learning can perform slightly better than multi-task learning. This implies that not all tasks can complement each other, and for those tasks with sufficient training data, adding extra tasks may not improve or even degrade performance. The potential reason is that the data in other tasks are collected from different domains and require models with distinct capabilities. Simply combining them will result in noise. For QA tasks, multi-task learning often leads to better performance. Indeed, QA is more complicated than other tasks, thus training on other tasks can be beneficial (*e.g.*, better capture the relationship between question and answer). Some recent studies (Aribandi et al., 2021; Aghajanyan et al., 2021) have reported that incorporating massive tasks may alleviate the negative effect between tasks. It is interesting to investigate if our model can be further improved by adding more tasks, and we leave this as future work.

(2) Sparsely activated (SkillNet-style) methods can achieve better performance than single-/multi-task learning. Its advantages stem from two perspectives: First, this method groups tasks according to their requiring skills, so the tasks rely on similar skills can enhance one another (*e.g.*, both NLI and semantic similarity judgment rely on the skill of understanding how two text segments interact), while avoiding noise from other used skills. On the other hand, since multiple skill modules are activated, more parameters improve the model's capacity.

(3) Our HMNet with three different task relevance measurements can consistently outperform all baseline methods. Specifically, HMNet brings more than 0.7% absolute improvement over multi-task learning in terms of average score. We attribute this improvement to our architecture of task-clustering and task-specific layers. Instead of sharing all layers, HMNet gradually separates the tasks in higher layers, so that the general language knowledge can be accumulated while the noise can be filtered out. Different from SkillNet-style methods, our HMNet only activates one channel at each layer, so its number of parameters is identical to a vanilla BERT. Surprisingly, HMNet with fewer parameters can even perform better. This demonstrates again the effectiveness of our proposed task clustering and the hierarchical sharing structure.

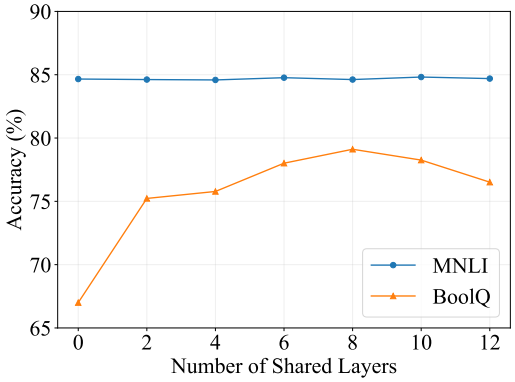

Figure 3: Influence of different numbers of shared layers in HMNet.

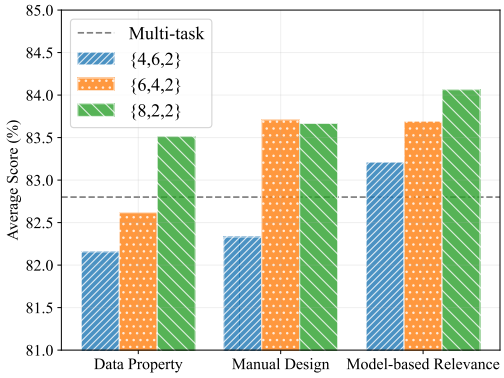

Figure 4: Results of average score with respect to different layer combinations.

4.5 Further Analysis

Influence of Shared Layers In HMNet, we design the bottom layers as fully shared, whilst the higher layers are only shared by task clusters or are task-specific. This approach is motivated by the hypothesis described in Section 3.2, which argues that the positive influence between tasks comes from capturing more general language knowledge at the bottom layer, and the negative impact comes from the noise brought by task interaction in the middle and top layers. To prove this hypothesis, we experiment on using different numbers of shared layers to investigate their effect. This experiment is conducted on the MNLI and BoolQ datasets. We train HMNet on them and report their performance accordingly. Since only two tasks are considered, we categorize them into two clusters, so the task-clustering layers are transformed into task-specific. The results are shown in Figure 3. We can observe that the performance of MNLI has minor changes as it contains sufficient training data. On the contrary, the result of BoolQ increases significantly when shared layers are used (from zero to eight). This reflects the advantage of multi-task learning. However, when more shared layers are employed (more than eight), the performance degrades. This confirms our assumption that task-specific knowledge is often learned in the upper layers, which supports our design of gradually separating tasks from the bottom up.

Different Metrics for Task Relevance In Section 3.2, we devise three different metrics for task relevance, *i.e.*, data property, manual design, and model-based relevance. Their performance is shown in Figure 4 and the right side of Table 1. We can see: First, HMNet can outperform other baselines with any task relevance assessment. This highlights the significance of task clustering in multi-task learning. Moreover, the result demonstrates the adaptability of our method with regard to the evaluation of task relevance. As an early exploration of the effect of task relevance on multi-task learning, the three metrics we have provided are very preliminary. We believe that a more accurate task relevance could bring further improvement. Second, the model-based similarity performs the best. With the help of deep neural models, the task relevance in high dimensions can be better captured. Such relevance is hard to be observed by humans or extracted from shallow data properties. In addition, quantifying the tasks' relevance from the perspective of the model can narrow the gap between task clustering and multi-task learning. Notably, though better performance is obtained, the model-based metric needs additional cost on model training. The other two metrics can avoid it.

Influence of Different Layer Combinations HMNet has three different kinds of layers, so their different combinations may influence the performance. We conduct an experiment by tuning the number of layers at each level. For clarity, we use $\{x, y, z\}$ to denote a HMNet with x shared layers, y task-clustering layers, and z task-specific layers. Experimental results are shown in Figure 4. Using eight shared layers yields the optimal performance. In comparison to traditional multi-task learning that shares all layers, HMNet with more than six shared layers can obtain better results. This implies that sharing all layers is not an effective strategy for multi-task learning, and our hierarchical sharing structure can mitigate the negative effect across tasks. Besides, when equipping with the

model-based similarity, HMNet outperforms multi-task learning with any combination of shared, task-clustering, and task-specific layers. This reflects the robustness of our method and validates that the excellent performance of our method stems from our HMNet architecture and the consideration of task relevance rather than finely tuned hyperparameters.

5 Conclusion and Future Work

In this work, we explore task correlation and built a hierarchical multi-task learning framework. Our framework adopts a coarse-to-fine manner, in which the tasks are gradually separated from the bottom up. By doing so, it can reap the benefits of multi-tasking learning at the lower layers while avoiding its harmful impact on the upper layers. Extensive experiments on several challenging NLU datasets showed that our model achieves better performance than existing strategies. Further experiments indicated that our methods are flexible with the choice of task relevance metrics, and robust with the hyperparameter selection. As a preliminary study on incorporating task relevance into multi-task learning, there are several potential future directions, such as new backbone models and task relevance metrics.

References

Armen Aghajanyan, Anchit Gupta, Akshat Shrivastava, Xilun Chen, Luke Zettlemoyer, and Sonal Gupta. 2021. Muppet: Massive multi-task representations with pre-finetuning. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing, EMNLP 2021, Virtual Event / Punta Cana, Dominican Republic, 7-11 November, 2021*, pages 5799–5811. Association for Computational Linguistics.

Vamsi Aribandi, Yi Tay, Tal Schuster, Jinfeng Rao, Huaixiu Steven Zheng, Sanket Vaibhav Mehta, Honglei Zhuang, Vinh Q. Tran, Dara Bahri, Jianmo Ni, Jai Prakash Gupta, Kai Hui, Sebastian Ruder, and Donald Metzler. 2021. Ext5: Towards extreme multi-task scaling for transfer learning. *CoRR*, abs/2111.10952.

Samuel R. Bowman, Gabor Angeli, Christopher Potts, and Christopher D. Manning. 2015. A large annotated corpus for learning natural language inference. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing, EMNLP 2015, Lisbon, Portugal, September 17-21, 2015*, pages 632–642. The Association for Computational Linguistics.

Daniel M. Cer, Mona T. Diab, Eneko Agirre, Iñigo Lopez-Gazpio, and Lucia Specia. 2017. Semeval-2017 task 1: Semantic textual similarity multilingual and crosslingual focused evaluation. In *Proceedings of the 11th International Workshop on Semantic Evaluation, SemEval@ACL 2017, Vancouver, Canada, August 3-4, 2017*, pages 1–14. Association for Computational Linguistics.

Christopher Clark, Kenton Lee, Ming-Wei Chang, Tom Kwiatkowski, Michael Collins, and Kristina Toutanova. 2019a. Boolq: Exploring the surprising difficulty of natural yes/no questions. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT 2019, Minneapolis, MN, USA, June 2-7, 2019, Volume 1 (Long and Short Papers)*, pages 2924–2936. Association for Computational Linguistics.

Kevin Clark, Minh-Thang Luong, Urvashi Khandelwal, Christopher D. Manning, and Quoc V. Le. 2019b. Bam! born-again multi-task networks for natural language understanding. In *Proceedings of the 57th Conference of the Association for Computational Linguistics, ACL 2019, Florence, Italy, July 28- August 2, 2019, Volume 1: Long Papers*, pages 5931–5937. Association for Computational Linguistics.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT 2019, Minneapolis, MN, USA, June 2-7, 2019, Volume 1 (Long and Short Papers)*, pages 4171–4186. Association for Computational Linguistics.

William B. Dolan and Chris Brockett. 2005. Automatically constructing a corpus of sentential paraphrases. In *Proceedings of the Third International Workshop on Paraphrasing, IWP@IJCNLP 2005, Jeju Island, Korea, October 2005, 2005*. Asian Federation of Natural Language Processing.

Ganesh Jawahar, Benoît Sagot, and Djamé Seddah. 2019. What does BERT learn about the structure of language? In *Proceedings of the 57th Conference of the Association for Computational Linguistics, ACL 2019, Florence, Italy, July 28- August 2, 2019, Volume 1: Long Papers*, pages 3651–3657. Association for Computational Linguistics.

Daniel Khashabi, Snigdha Chaturvedi, Michael Roth, Shyam Upadhyay, and Dan Roth. 2018. Looking beyond the surface: A challenge set for reading comprehension over multiple sentences. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT 2018, New Orleans, Louisiana, USA, June 1-6, 2018, Volume 1 (Long Papers)*, pages 252–262. Association for Computational Linguistics.

Olga Kovaleva, Alexey Romanov, Anna Rogers, and Anna Rumshisky. 2019. Revealing the dark secrets of BERT. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing, EMNLP-IJCNLP 2019, Hong Kong, China, November 3-7, 2019*, pages 4364–4373. Association for Computational Linguistics.

Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Veselin Stoyanov, and Luke Zettlemoyer. 2020. BART: denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, ACL 2020, Online, July 5-10, 2020*, pages 7871–7880. Association for Computational Linguistics.

Xiaodong Liu, Pengcheng He, Weizhu Chen, and Jianfeng Gao. 2019a. Multi-task deep neural networks for natural language understanding. In *Proceedings of the 57th Conference of the Association for Computational Linguistics, ACL 2019, Florence, Italy, July 28- August 2, 2019, Volume 1: Long Papers*, pages 4487–4496. Association for Computational Linguistics.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019b. Roberta: A robustly optimized BERT pretraining approach. *CoRR*, abs/1907.11692.

Ilya Loshchilov and Frank Hutter. 2019. Decoupled weight decay regularization. In *7th International Conference on Learning Representations, ICLR 2019, New Orleans, LA, USA, May 6-9, 2019*. OpenReview.net.

Andrew L. Maas, Raymond E. Daly, Peter T. Pham, Dan Huang, Andrew Y. Ng, and Christopher Potts. 2011. Learning word vectors for sentiment analysis. In *The 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies, Proceedings of the Conference, 19-24 June, 2011, Portland, Oregon, USA*, pages 142–150. The Association for Computer Linguistics.

Jason Phang, Thibault Févry, and Samuel R. Bowman. 2018. Sentence encoders on stilts: Supplementary training on intermediate labeled-data tasks. *CoRR*, abs/1811.01088.

Alec Radford, Jeff Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. 2019. Language models are unsupervised multitask learners.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J. Liu. 2020. Exploring the limits of transfer learning with a unified text-to-text transformer. *J. Mach. Learn. Res.*, 21:140:1–140:67.

Anna Rogers, Olga Kovaleva, and Anna Rumshisky. 2020. A primer in bertology: What we know about how BERT works. *Trans. Assoc. Comput. Linguistics*, 8:842–866.

Richard Socher, Alex Perelygin, Jean Wu, Jason Chuang, Christopher D. Manning, Andrew Y. Ng, and Christopher Potts. 2013. Recursive deep models for semantic compositionality over a sentiment treebank. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing, EMNLP 2013, 18-21 October 2013, Grand Hyatt Seattle, Seattle, Washington, USA, A meeting of SIGDAT, a Special Interest Group of the ACL*, pages 1631–1642. ACL.

Duyu Tang, Fan Zhang, Yong Dai, Cong Zhou, Shuangzhi Wu, and Shuming Shi. 2022. Skillnet-nlu: A sparsely activated model for general-purpose natural language understanding.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, December 4-9, 2017, Long Beach, CA, USA*, pages 5998–6008.

Alex Wang, Yada Pruksachatkun, Nikita Nangia, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel R. Bowman. 2019a. Superglue: A stickier benchmark for general-purpose language understanding systems. In *Advances in Neural Information Processing Systems 32: Annual Conference on Neural Information Processing Systems 2019, NeurIPS 2019, December 8-14, 2019, Vancouver, BC, Canada*, pages 3261–3275.

Alex Wang, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel R. Bowman. 2019b. GLUE: A multi-task benchmark and analysis platform for natural language understanding. In *7th International Conference on Learning Representations, ICLR 2019, New Orleans, LA, USA, May 6-9, 2019*. OpenReview.net.

Adina Williams, Nikita Nangia, and Samuel R. Bowman. 2018a. A broad-coverage challenge corpus for sentence understanding through inference. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT 2018, New Orleans, Louisiana, USA, June 1-6, 2018, Volume 1 (Long Papers)*, pages 1112–1122. Association for Computational Linguistics.

Adina Williams, Nikita Nangia, and Samuel R. Bowman. 2018b. A broad-coverage challenge corpus for sentence understanding through inference. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT 2018, New Orleans, Louisiana, USA, June 1-6, 2018, Volume 1 (Long Papers)*, pages 1112–1122. Association for Computational Linguistics.

Zhilin Yang, Zihang Dai, Yiming Yang, Jaime G. Carbonell, Ruslan Salakhutdinov, and Quoc V. Le. 2019. Xlnet: Generalized autoregressive pretraining for language understanding. In *Advances in Neural Information Processing Systems 32: Annual Conference on Neural Information Processing Systems 2019, NeurIPS 2019, December 8-14, 2019, Vancouver, BC, Canada*, pages 5754–5764.

Yutao Zhu, Jian-Yun Nie, Zhicheng Dou, Zhengyi Ma, Xinyu Zhang, Pan Du, Xiaochen Zuo, and Hao Jiang. 2021a. Contrastive learning of user behavior sequence for context-aware document ranking. In *CIKM '21: The 30th ACM International Conference on Information and Knowledge Management, Virtual Event, Queensland, Australia, November 1 - 5, 2021*, pages 2780–2791. ACM.

Yutao Zhu, Kun Zhou, Jian-Yun Nie, Shengchao Liu, and Zhicheng Dou. 2021b. Neural sentence ordering based on constraint graphs. In *Thirty-Fifth AAAI Conference on Artificial Intelligence, AAAI 2021, Thirty-Third Conference on Innovative Applications of Artificial Intelligence, IAAI 2021, The Eleventh Symposium on Educational Advances in Artificial Intelligence, EAAI 2021, Virtual Event, February 2-9, 2021*, pages 14656–14664. AAAI Press.

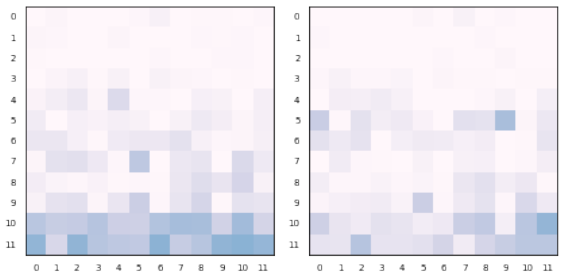

Figure 5: Per head attention maps' cosine similarity between fine-tuned model and co-trained model using positive task pair(QQP & RTE). Darker colors means greater differences. left: QQP right: RTE

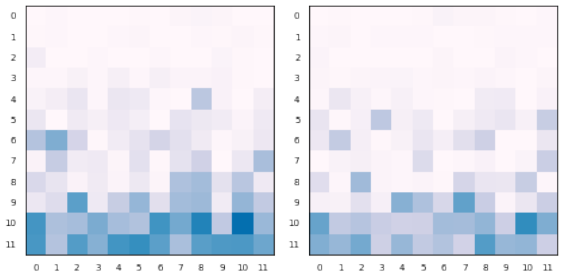

Figure 6: Per head attention maps' cosine similarity between fine-tuned model and co-trained model using negative task pair(MRPC & RTE). Darker colors means greater differences. left: MRPC right: RTE

A The Clustering Results

As described in Section 3.2, we group these tasks from three prospects, *i.e.*, data property, manual design, and model-based relevance.

As for the data property of datasets, Table 2 reports the task relevance of all 13 datasets. Based on these features, we cluster all tasks into three groups using k-means. {WNLI, CB}, {MultiRC, RTE, SST-2, MRPC, STS-B}, {QQP, QNLI, BoolQ, IMDB, SNLI, MNLI} is the grouping results.

As for the model-based relevance, the results are shown in Table 3. Similarly, we cluster these tasks using k-means. The outcomes are {CB, WNLI, QQP, RTE}, {MRPC, QNLI, BoolQ, IMDB, SST-2}, {MultiRC, STS-B, SNLI, MNLI}.

B Dataset Statistics

The statistic of all datasets are shown in Table 4.

C Attention Map Similarity

Following previous work (Liu et al., 2019a; Rogers et al., 2020), we compare the attention maps between fine-tuned model and the co-trained model to explore the behavior during multi-task learning. As shown in Table 3, the relevance between RTE and QQP is relatively positive and that between RTE and MRPC is relatively negative. We conduct some experiments using these tasks to explore the training difference between positive task-pair and negative pairs. Figure 5 compares the similarity between positive task pair (QQP & RTE), and Figure 6 compares the negative task-pair (MRPC & RTE). As shown in these figures, regardless of the positive pair or negative pair, the attention map of the bottom six layers is very similar, while that of the top two layers is different. Compared Figure 5 with Figure 6, we can see the attention map in the positive pair is more similar in the middle layers. Therefore, when considering the model structure, we set the bottom layers as shared to learn general knowledge from all tasks. Accordingly, the higher layers are designed as more task-specific.

	MNLI	QNLI	RTE	WNLI	CB	SNLI	IMDB	SST-2	QQP	STSB	MRPC	BoolQ	MultiRC
MNLI	100.00%	92.83%	97.64%	99.58%	99.42%	98.50%	95.70%	99.21%	93.46%	98.29%	98.21%	93.83%	98.76%
QNLI	94.40%	100.00%	98.11%	98.32%	98.01%	95.02%	94.00%	96.55%	94.19%	97.71%	98.02%	95.31%	97.55%
RTE	52.42%	51.80%	100.00%	84.72%	81.73%	59.25%	52.99%	65.23%	52.26%	74.24%	74.11%	54.11%	66.54%
WNLI	5.50%	5.34%	8.72%	100.00%	24.80%	7.47%	5.63%	9.90%	5.51%	10.64%	9.77%	5.75%	8.78%
CB	10.62%	10.29%	16.26%	47.93%	100.00%	13.76%	10.87%	18.86%	10.61%	18.95%	18.44%	11.00%	16.15%
SNLI	69.92%	66.33%	78.35%	95.94%	91.44%	100.00%	70.87%	84.65%	68.98%	83.62%	80.12%	69.19%	80.05%
IMDB	93.74%	90.55%	96.68%	99.72%	99.67%	97.80%	100.00%	99.69%	91.86%	97.45%	97.06%	92.13%	97.41%
SST-2	44.45%	42.54%	54.44%	80.31%	79.12%	53.43%	45.60%	100.00%	44.07%	58.34%	57.02%	44.68%	56.31%
QQP	92.93%	92.11%	96.79%	99.09%	98.80%	96.63%	93.25%	97.80%	100.00%	97.67%	97.40%	93.65%	97.13%
STSB	41.08%	40.17%	57.80%	80.52%	74.18%	49.24%	41.59%	54.43%	41.06%	100.00%	67.28%	42.43%	54.52%
MRPC	43.97%	43.15%	61.80%	79.12%	77.30%	50.53%	44.36%	56.97%	43.85%	72.06%	100.00%	45.47%	57.67%
BoolQ	87.67%	87.58%	94.18%	97.20%	96.27%	91.07%	87.88%	93.19%	88.01%	94.84%	94.90%	100.00%	93.29%
MultiRC	57.48%	55.84%	72.14%	92.43%	88.03%	65.64%	57.88%	73.15%	56.86%	75.92%	74.98%	58.11%	100.00%

Table 2: Task relevance based on data property. It is computed by the vocabulary co-occurrence between the task pair as Equation (2).

	MNLI	QNLI	RTE	WNLI	CB	SNLI	IMDB	SST-2	QQP	STS-B	MRPC	BoolQ	MultiRC
MNLI	0.00%	-0.10%	15.96%	50.00%	47.31%	0.75%	-0.29%	0.00%	-0.14%	-0.96%	-1.87%	5.93%	6.42%
QNLI	-0.54%	0.00%	2.13%	21.43%	26.37%	0.24%	-0.09%	-0.62%	-0.06%	0.43%	-0.69%	5.38%	1.58%
RTE	-0.47%	-0.32%	0.00%	7.14%	10.45%	0.25%	0.10%	-0.98%	-0.03%	-0.10%	-1.95%	0.67%	1.03%
WNLI	-0.28%	-0.32%	-3.19%	0.00%	2.62%	0.08%	0.08%	-0.74%	-0.03%	-0.16%	-0.39%	0.59%	-0.18%
CB	-0.35%	0.02%	0.00%	7.14%	0.00%	-0.30%	-0.09%	-0.49%	0.04%	0.35%	-0.50%	-0.17%	0.29%
SNLI	-0.19%	-0.64%	11.70%	64.29%	33.51%	0.00%	-0.17%	-1.85%	-0.12%	-1.23%	-3.16%	3.19%	3.16%
IMDB	0.01%	0.00%	-1.06%	14.29%	30.95%	-0.03%	0.00%	-0.74%	-0.16%	0.52%	0.79%	0.38%	0.68%
SST-2	0.21%	-0.02%	1.06%	14.29%	33.58%	0.09%	-4.24%	0.00%	0.02%	-0.76%	-1.98%	4.71%	0.53%
QQP	-0.77%	-0.82%	3.19%	0.00%	7.48%	0.09%	-0.41%	-2.21%	0.00%	-1.63%	-2.85%	1.89%	-0.19%
STS-B	-0.12%	0.04%	5.85%	57.14%	13.07%	0.13%	0.03%	-1.23%	0.05%	0.00%	1.15%	3.15%	1.48%
MRPC	-0.21%	0.02%	2.66%	21.43%	13.07%	-0.01%	-0.01%	0.25%	-0.11%	-0.23%	0.00%	1.77%	0.22%
BoolQ	-0.35%	-0.36%	-2.66%	21.43%	33.58%	-0.15%	-0.05%	-0.62%	-0.03%	1.18%	-1.24%	0.00%	-0.26%
MultiRC	-0.37%	-0.20%	4.26%	71.43%	10.38%	-0.07%	-0.04%	-1.48%	0.04%	0.70%	0.27%	1.64%	0.00%

Table 3: Model-based task relevance. It is computed by performance improvement as Equation (4).

Corpus	#Train	#Dev	#Test	#Label	Metrics
NATURAL LANGUAGE INFERENCE					
MNLI	393K	20k	20k	3	Accuracy
QNLI	108k	5.7k	5.7k	2	Accuracy
RTE	2.5k	276	3k	2	Accuracy
WNLI	634	71	146	2	Accuracy
CB	250	57	250	2	Accuracy/F1
SNLI	549k	9.8k	9.8k	3	Accuracy
SENTIMENT CLASSIFICATION					
IMDB	25k	0k	25k	2	Accuracy
SST-2	67K	872	1.8k	2	Accuracy
SIMILARITY AND PARAPHRASE					
QQP	364k	40k	391k	2	Accuracy/F1
STS-B	7K	1.5k	1.4k	1	Pearson/Spearman corr
MRPC	3.7k	408	1.7k	2	Accuracy/F1
QUESTION ANSWERING					
BoolQ	9.4k	3.3k	3.2k	2	Accuracy
MultiRC	5.1k	953	1.8k	2	$F1_a$/EM

Table 4: Summary of the 13 datasets.

Automatic Label Sequence Generation for Prompting Sequence-to-sequence Models

Zichun Yu[1] **Tianyu Gao**[2] **Zhengyan Zhang**[13] **Yankai Lin**[4]
Zhiyuan Liu[1356†] **Maosong Sun**[1356†] **Jie Zhou**[7]
[1]Dept. of Comp. Sci. & Tech., Institute for AI, Tsinghua University
[2]Princeton University
[3]Beijing National Research Center for Information Science and Technology
[4]Gaoling School of Artificial Intelligence, Renmin University of China
[5]International Innovation Center of Tsinghua University, Shanghai, China
[6]Beijing Academy of Artificial Intelligence
[7]Pattern Recognition Center, WeChat AI, Tencent Inc
yuzc19@mails.tsinghua.edu.cn {liuzy,sms}@tsinghua.edu.cn

Abstract

Prompting, which casts downstream applications as language modeling tasks, has shown to be sample efficient compared to standard fine-tuning with pre-trained models. However, one pitfall of prompting is the need of manually-designed patterns, whose outcome can be unintuitive and requires large validation sets to tune. To tackle the challenge, we propose **AutoSeq**, a fully automatic prompting method: (1) We adopt natural language prompts on sequence-to-sequence models, enabling free-form generation and larger label search space; (2) We propose label sequences – phrases with indefinite lengths to verbalize the labels – which eliminate the need of manual templates and are more expressive than single label words; (3) We use beam search to automatically generate a large amount of label sequence candidates and propose contrastive re-ranking to get the best combinations. AutoSeq significantly outperforms other no-manual-design methods, such as soft prompt tuning, adapter tuning, and automatic search on single label words; the generated label sequences are even better than curated manual ones on a variety of tasks. Our method reveals the potential of sequence-to-sequence models in few-shot learning and sheds light on a path to generic and automatic prompting. The source code of this paper can be obtained from https://github.com/thunlp/Seq2Seq-Prompt.

1 Introduction

Among ways of adapting pre-trained language models (Devlin et al., 2019; Raffel et al., 2020) to downstream applications, prompting, which uses a natural language prompt to reformulate tasks as cloze questions, has shown to be especially effective (Brown et al., 2020; Schick and Schütze, 2021a,b; Gao et al., 2021). For example, in sentiment classification, prompting appends a *template* "It was [MASK]" to the original input, and defines "great" and "terrible" as the *label words*, whose probabilities at [MASK] indicate the probabilities of the positive and negative sentiment labels. Prompting possesses better sample efficiency and performs significantly better than standard fine-tuning in the low resource case.

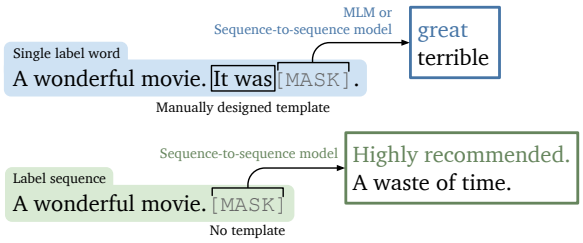

Figure 1: Single label words vs label sequences. Label sequences are more expressive and eliminate the need of manually-designed templates.

However, the prompting performance is highly sensitive to the prompt choice, whose effectiveness needs abundant validation data to evaluate and is difficult to predict by intuition (Gao et al., 2021; Perez et al., 2021). Even though there exist methods that explore automatic prompt search (Schick et al., 2020; Gao et al., 2021), they still require substantial human efforts, for the algorithms start from either manual templates or label words.

We propose AutoSeq, a prompting method that is fully automatic and requires no human input. AutoSeq has three innovations: (1) AutoSeq adopts *sequence-to-sequence models* like T5 (Raffel et al.,

Part of the work was done while Yankai Lin was working at Tencent.

† Corresponding authors

Proceedings of the 29th International Conference on Computational Linguistics, pages 4965–4975
October 12–17, 2022.

2020). Compared to masked language models (MLM) like BERT (Devlin et al., 2019), it allows free-form generation, enables more types of tasks, and extends the label space for prompting. (2) We propose *label sequences*, which are indefinite-length phrases or sentences that represent each label. They are more expressive than previous single label words and eliminate the need for a manual template (Figure 1). (3) We design an automatic label sequence search pipeline, which first generates a large amount of candidates by T5, then re-ranks them by *contrastive probability*.

Our main experiment results on natural language understanding datasets show that AutoSeq performs significantly better than automatic prompt search using single label words as well as no-prompt methods like soft prompt tuning and adapter tuning. AutoSeq also outperforms hand-crafted prompts on a variety of tasks. We hope our work enlightens automatic prompting and building a universal prompt-based fine-tuning framework.

2 Related Work

Prompting. Schick and Schütze (2021a,b); Gao et al. (2021) introduced prompting into MLM. Though showing remarkable few-shot performance, those models are constrained by the single `[MASK]` token and are limited to classification tasks; they also require manually-designed prompts. In parallel, soft prompt tuning (Lester et al., 2021) and adapter tuning (Houlsby et al., 2019; Zaken et al., 2022; Hu et al., 2022) do not require manual design, but they lag behind prompting in few-shot performance (Gu et al., 2022). Recent work (Zhang et al., 2022) tries to mitigate the gap, but it still requires the help of manual prompts and thus falls out of the scope of our discussion.

Automatic prompt search. There have been plenty of attempts for automatic prompt search – yet all of them require to start from either human-designed label words or templates (Davison et al., 2019; Jiang et al., 2020; Shin et al., 2020; Schick et al., 2020; Gao et al., 2021; Yuan et al., 2021; Haviv et al., 2021). In contrast, our AutoSeq is a general-purpose, fully automatic search method that depends only on few-shot annotations.

3 AutoSeq

3.1 Prompts for sequence-to-sequence models

We introduce the sequence-to-sequence version of *prompt-based fine-tuning*, bringing in *label sequences* that are more expressive than one token. Using sentiment classification as an example, and given the input sentence as x, the model input can be formulated as "x `[MASK]`". We define the label sequences for the positive class as "*Highly recommended.*" and that for the negative class as "*Not for me.*". Then the probability of each class is tied with that of the T5 model generating "*Highly recommended.*" and "*Not for me.*" at position `[MASK]`. As we compare the MLM single label words to our label sequences (Figure 1), we see that label sequences encode richer semantic meaning and get rid of sophisticated templates, since label sequences themselves can be standalone sentences.

In natural language inference (NLI) tasks[1] with two input sentences, our model input changes to "x_1? `[MASK]`, x_2" and label sequences can be "*I mean*" (entailment), "*For example*" (neutral), and "*However*" (contradiction).

Formally, we have a task-specific template $\mathcal{T}$[2] and a task-specific mapping $\mathcal{M}\colon \mathcal{Y} \to \mathcal{V}^+$ from the task label space $\mathcal{Y}$ to the label sequence space ($\mathcal{V}$ is the vocabulary of the model $\mathcal{L}$). Then, for a formulated example $\mathcal{T}(x)$ and its corresponding label sequences, we use the cross-entropy loss (the same way how T5 is trained)[3] to fine-tune the model. In inference, we compute the score of each class $y \in \mathcal{Y}$ as the auto-regressive log-probability of the corresponding label sequence:

$$q(\mathcal{M}(y) \mid \mathcal{T}(x)) = \sum_{j=1}^{|\mathcal{M}(y)|} \log P_{\mathcal{L}}\big(t_j \mid t_{1:j-1}, \mathcal{T}(x)\big), \quad (1)$$

where $P_{\mathcal{L}}$ denotes the output probability of the sequence-to-sequence model, $\mathcal{M}(y) = (t_1, \ldots, t_{|\mathcal{M}(y)|})$ is the corresponding label sequence tokens, and $t_{1:j-1}$ is $t_1, ..., t_{j-1}$.

3.2 Automatic label sequence generation

Thanks to the introduction of label sequences, manually-designed templates are no longer needed, and the goal of automatic prompt search is simply to construct a label sequence mapping $\mathcal{M}$ that performs well. Our proposed automatic label sequence

[1] We have details for all tasks in Appendix B.2.

[2] Unlike in the MLM case, the template here is simply the way to concatenate the input and the mask.

[3] For regression tasks like STS-B, we use the same method as Gao et al. (2021) to compute the loss instead.

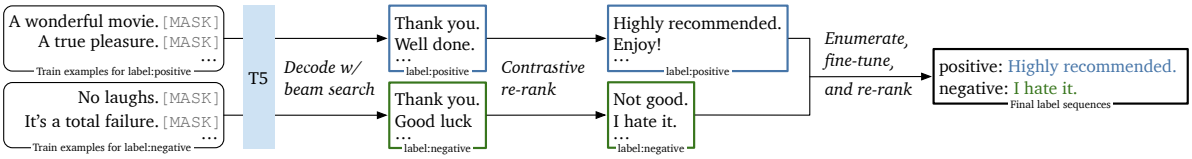

Figure 2: Illustration of AutoSeq. We first use T5 to generate label sequence candidates given each label's training instances; we then use *contrastive re-ranking* to get label sequences that are more label-specific; in the end we enumerate all the combinations and re-rank by the fine-tuning performance.

generation pipeline contains three steps (Figure 2): (1) candidate generation by using T5 and beam search; (2) re-ranking by contrastive probability; (3) enumerating label sequence combinations and re-ranking by fine-tuning performance.

We first use the T5 model and beam search to generate multiple sequence label candidates $\mathcal{S}^y \subset \mathcal{V}^+$ for each class y. Denote $\mathcal{D}^y_{\text{train}} \subset \mathcal{D}_{\text{train}}$ be the subset of all few-shot training data of class y, we find s^y that has the top scores by this equation:

$$\sum_{(x,y)\in\mathcal{D}^y_{\text{train}}} q(s^y \mid \mathcal{T}(x)), \tag{2}$$

where $q(\cdot)$ is defined as Eq. (1). Since the search space is too large, we decompose it to an auto-regressive decoding following Gao et al. (2021):

$$\sum_{j=1}^{|s^y|} \sum_{(x,y)\in\mathcal{D}^y_{\text{train}}} \log P_{\mathcal{L}}(s^y_j | s^y_{1:j-1}, \mathcal{T}(x)). \tag{3}$$

By using beam search, we can generate a large amount of label sequence candidates by just one decoding pass. However, we notice that it tends to generate similar generic label sequences across different classes, while we expect the label sequences to be distinguishable for each class. For example, in sentiment classification, both classes will get a generic candidate of "Thank you", which is coherent to be put at the mask but does not help with the classification (more discussion in Appendix F).

To eliminate the problem, we introduce the second step of our automatic pipeline, which re-ranks all the candidates based on the *contrastive probability* $\tilde{q}(s^y)$ of $s^y \in \mathcal{S}^y$:

$$\frac{\sum_{(x,y)\in\mathcal{D}^y_{\text{train}}} q(s^y \mid \mathcal{T}(x))}{|\mathcal{D}^y_{\text{train}}|} - \frac{\sum_{(x,y')\in\overline{\mathcal{D}}^y_{\text{train}}} q(s^y \mid \mathcal{T}(x))}{|\overline{\mathcal{D}}^y_{\text{train}}|}, \tag{4}$$

where $\overline{\mathcal{D}}^y_{\text{train}} = \mathcal{D}_{\text{train}} \backslash \mathcal{D}^y_{\text{train}}$.

Then, we define the score of a label mapping as the sum of corresponding $\tilde{q}(s^y)$ for each class y. To shorten the time for further re-ranking, we only select the top n mappings with the highest scores. Finally, we fine-tune the model over the top n label mapping candidates, and re-rank them to find the best one based on the few-shot development set, which has been proved critical in the label mapping selection (Gao et al., 2021).

4 Experiments

4.1 Main results

We use a `T5-base` v1.1 (Shazeer, 2020)[4] model and set the number of training examples per class as 16 in our experiments. Datasets and experiments details can be found in Appendix A and B. To make our results convincing, we compare to the following baselines in our few-shot setting: (1) parameter-efficient tuning – soft prompt tuning (Lester et al., 2021) and adapter tuning (Houlsby et al., 2019; Karimi Mahabadi et al., 2022) – which fixes the pre-trained model parameters and only tunes the soft prompt or adapter part; (2) standard fine-tuning; (3) manual prompts (Table D.1) proposed in Logan IV et al. (2021); (4) automatic label word search (AutoWord), which has the same setting as AutoSeq except that it is limited to only using one single token as a label word. This can be seen as an approximation of Auto-L in Gao et al. (2021). We also include the results from standard fine-tuning based on the full training set.

Table 1 shows our main results. First, **prompt-based fine-tuning can significantly beat standard fine-tuning**, either using manual prompts or generated ones, let alone parameter-efficient tuning. Our method AutoSeq achieves a 9.4% gain on average compared to standard fine-tuning.

Second, **AutoSeq achieves a 3.2% improvement on average compared to the manual prompts**, and performs significantly better in NLI tasks. However, for most of the sentiment classification tasks, though without engineering, the manual prompts can still outperform AutoSeq. We attribute it to the simplicity of these tasks, making the manual design of prompts more intuitive.

[4]The released original T5 models are also fine-tuned on downstream tasks while T5 v1.1 models exclude those tasks.

	SST-2 (acc)	SST-5 (acc)	MR (acc)	CR (acc)	MPQA (acc)	Subj (acc)	TREC (acc)	CoLA (Matt.)
Prompt tuning	51.4 (0.0)	24.9 (0.0)	50.6 (0.0)	50.7 (0.0)	50.0 (0.0)	59.9 (0.0)	22.6 (0.0)	-4.0 (0.0)
Adapter tuning	84.7 (3.2)	27.7 (4.8)	73.6 (3.9)	86.9 (1.4)	78.6 (3.6)	84.7 (3.3)	27.6 (3.6)	4.8 (4.8)
Fine-tuning	80.9 (2.0)	36.0 (2.2)	70.1 (7.1)	76.7 (6.8)	80.9 (2.6)	84.3 (4.4)	68.3 (14.7)	0.5 (6.3)
Prompt-based FT (Manual)	**91.2** (0.7)	**45.2** (1.5)	**85.4** (1.5)	**89.8** (1.5)	**85.1** (2.9)	89.1 (1.1)	80.0 (2.5)	0.7 (5.3)
Prompt-based FT (AutoWord)	87.6 (2.0)	40.4 (4.1)	82.1 (2.7)	87.0 (4.7)	75.1 (4.5)	87.4 (5.0)	82.4 (3.7)	**8.5** (4.5)
Prompt-based FT (AutoSeq)	89.8 (1.1)	42.3 (3.4)	83.9 (1.3)	87.2 (2.5)	82.5 (2.7)	**91.6** (1.9)	**85.2** (4.3)	7.6 (9.9)
Fine-tuning (Full train set)	93.3	56.1	89.3	86.9	89.0	96.2	97.0	30.1
	MNLI (acc)	**MNLI-mm (acc)**	**SNLI (acc)**	**QNLI (acc)**	**RTE (acc)**	**MRPC (F1)**	**QQP (F1)**	**STS-B (Pear.)**
Prompt tuning	34.6 (0.0)	34.2 (0.0)	34.1 (0.0)	54.2 (0.0)	47.3 (0.0)	**81.2** (0.0)	53.8 (0.0)	10.7 (0.0)
Adapter tuning	33.5 (1.4)	33.9 (1.8)	34.7 (1.3)	55.4 (2.5)	50.2 (2.0)	77.4 (2.3)	50.7 (5.2)	6.8 (2.8)
Fine-tuning	36.1 (2.3)	36.4 (2.6)	36.0 (3.0)	58.6 (2.5)	51.8 (2.7)	74.9 (5.2)	57.0 (3.5)	11.9 (2.8)
Prompt-based FT (Manual)	41.9 (3.4)	43.0 (3.6)	40.8 (1.6)	55.5 (3.1)	53.3 (3.1)	75.6 (7.0)	55.4 (1.8)	17.3 (9.5)
Prompt-based FT (AutoWord)	49.0 (4.7)	51.3 (4.6)	56.2 (8.4)	59.9 (4.7)	48.7 (2.4)	73.5 (6.3)	60.6 (4.3)	**30.0** (8.4)
Prompt-based FT (AutoSeq)	**51.8** (1.8)	**53.9** (2.0)	**62.7** (3.7)	**61.3** (4.0)	**55.3** (4.9)	72.3 (4.9)	**66.2** (2.6)	17.8 (13.4)
Fine-tuning (Full train set)	86.9	87.1	91.6	91.0	59.6	84.0	87.9	86.1
	BoolQ (acc)	**CB (F1)**	**COPA (acc)**	**MultiRC (F1)**	**ReCoRD (F1)**	**WiC (acc)**	**WSC (acc)**	**Average**
Prompt tuning	**59.5** (0.0)	36.4 (0.0)	47.0 (0.0)	54.4 (0.0)	16.3 (0.0)	50.0 (0.0)	**65.4** (0.0)	42.8
Adapter tuning	45.3 (1.5)	55.3 (9.0)	47.2 (3.7)	**59.1** (0.0)	23.2 (5.2)	51.7 (1.9)	60.2 (2.2)	50.1
Fine-tuning	48.1 (6.2)	66.4 (14.1)	47.4 (7.2)	**59.1** (0.0)	18.1 (2.4)	50.3 (2.8)	60.4 (5.1)	52.6
Prompt-based FT (Manual)	48.3 (5.5)	75.5 (8.6)	51.6 (1.5)	56.0 (3.3)	**56.6** (3.5)	52.5 (3.5)	63.5 (2.7)	58.8
Prompt-based FT (AutoWord)	50.1 (3.9)	66.1 (15.8)	49.8 (3.2)	57.9 (1.5)	**56.6** (3.5)	**53.1** (3.6)	62.1 (1.4)	59.8
Prompt-based FT (AutoSeq)	55.4 (8.1)	**76.6** (10.7)	**52.0** (6.8)	58.2 (0.9)	**56.6** (3.5)	52.6 (2.9)	62.1 (1.4)	**62.0**
Fine-tuning (Full train set)	64.1	92.1	52.0	59.1	76.0	59.1	66.3	77.4

Table 1: Our main results using `T5-base` (16 training examples per class). We report mean (and standard deviation) performance over 5 different splits. FT: fine-tuning; Manual: human-designed prompts (Table D.1); AutoWord: automatically searched single label words. The score marked as **bold** means the best performance in few-shot. The score marked with an underline means the best performance among automatic search methods.

	SST-2	SNLI	QQP	MultiRC
Manual with eng.	**90.8**	**64.1**	56.1	57.5
AutoSeq	89.8	62.7	**66.2**	**58.2**

Table 2: Manual prompts with engineering on large validation sets vs AutoSeq (Full results in Table E.2).

	SNLI	QQP	ReCoRD	WSC
RoBERTa-PET	43.3	53.4	42.7	55.0
T5-AutoSeq	**62.7**	**66.2**	**56.6**	**62.1**

Table 3: Sequence-to-sequence vs MLM prompting.

Third, **using AutoSeq leads to steady gains in a majority of tasks compared to AutoWord**, indicating that label sequences, which is only enabled by using sequence-to-sequence models, are more expressive than single label words.

The results indicate that automatic prompt generation, especially with template-free format and label sequences, is a promising path for prompt-based fine-tuning in low resource scenarios.

4.2 Analysis of prompt engineering

Table 2 compares manual prompts with considerable engineering efforts (Table E.1) to AutoSeq. In general, AutoSeq achieves on par performance with models using manual prompts across various types of tasks, illustrating the effectiveness of our method, especially when trial-and-error with large validation sets is impossible.

4.3 Analysis of different pre-trained models

To highlight the advantages of using sequence-to-sequence models, we also report the PET[5] results with `RoBERTa-base` in Table 3. We see that T5 performs better than RoBERTa by a large margin. Although the comparison is not fair[6] given T5 and RoBERTa are pre-trained with different corpora, we highlight the importance to have sequence-to-

[5]Without unlabeled corpora and ensemble.

[6]Surprisingly, `T5-base` has a lower GLUE average (84.67) than `RoBERTa-base` (86.35) with full-dataset.

sequence models in the world of prompt-based fine-tuning. Furthermore, for tasks like ReCoRD and WSC that require generation in prompting, T5 is perfectly fit for their output formats, while MLM models like RoBERTa require tricky workaround.

5 Conclusion

In this paper, we propose AutoSeq, a prompt-based fine-tuning method with (1) sequence-to-sequence models that enable free-form generation, (2) label sequences that significantly extend the prediction space, and (3) automatic prompt search that requires no human efforts for designing prompts. Comprehensive experiments show that AutoSeq significantly outperforms other prompt-based or parameter-efficient tuning methods. We hope AutoSeq further inspires research on exploring template-free prompt-based fine-tuning.

Acknowledgement

This work is supported by the National Key Research and Development Program of China (No. 2020AAA0106500). Zichun Yu conducted the experiments. Zichun Yu, Tianyu Gao, Zhengyan Zhang, Yankai Lin, and Zhiyuan Liu wrote the paper. Maosong Sun and Jie Zhou provided valuable suggestions to the research.

References

Roy Bar Haim, Ido Dagan, Bill Dolan, Lisa Ferro, Danilo Giampiccolo, Bernardo Magnini, and Idan Szpektor. 2006. The second PASCAL recognising textual entailment challenge.

Luisa Bentivogli, Peter Clark, Ido Dagan, and Danilo Giampiccolo. 2009. The fifth PASCAL recognizing textual entailment challenge. In *TAC*.

Samuel Bowman, Gabor Angeli, Christopher Potts, and Christopher D Manning. 2015. A large annotated corpus for learning natural language inference. In *Empirical Methods in Natural Language Processing (EMNLP)*.

Tom B Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. 2020. Language models are few-shot learners. In *Advances in Neural Information Processing Systems (NeurIPS)*.

Daniel Cer, Mona Diab, Eneko Agirre, Iñigo Lopez-Gazpio, and Lucia Specia. 2017. SemEval-2017 task 1: Semantic textual similarity multilingual and crosslingual focused evaluation. In *the 11th International Workshop on Semantic Evaluation (SemEval-2017)*.

Christopher Clark, Kenton Lee, Ming-Wei Chang, Tom Kwiatkowski, Michael Collins, and Kristina Toutanova. 2019. BoolQ: Exploring the surprising difficulty of natural yes/no questions. In *Proceedings of NAACL-HLT 2019*.

Ido Dagan, Oren Glickman, and Bernardo Magnini. 2005. The PASCAL recognising textual entailment challenge. In *the First International Conference on Machine Learning Challenges: Evaluating Predictive Uncertainty Visual Object Classification, and Recognizing Textual Entailment*.

Joe Davison, Joshua Feldman, and Alexander M Rush. 2019. Commonsense knowledge mining from pretrained models. In *Empirical Methods in Natural Language Processing (EMNLP)*.

Marie-Catherine De Marneffe, Mandy Simons, and Judith Tonhauser. 2019. The CommitmentBank: Investigating projection in naturally occurring discourse. To appear in proceedings of Sinn und Bedeutung 23. Data can be found at https://github.com/mcdm/CommitmentBank/.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional Transformers for language understanding. In *North American Chapter of the Association for Computational Linguistics (NAACL)*.

William B. Dolan and Chris Brockett. 2005. Automatically constructing a corpus of sentential paraphrases. In *the Third International Workshop on Paraphrasing (IWP2005)*.

Tianyu Gao, Adam Fisch, and Danqi Chen. 2021. Making pre-trained language models better few-shot learners. In *Association for Computational Linguistics (ACL)*.

Danilo Giampiccolo, Bernardo Magnini, Ido Dagan, and Bill Dolan. 2007. The third PASCAL recognizing textual entailment challenge. In *the ACL-PASCAL Workshop on Textual Entailment and Paraphrasing*.

Yuxian Gu, Xu Han, Zhiyuan Liu, and Minlie Huang. 2022. Ppt: Pre-trained prompt tuning for few-shot learning. In *Proceedings of ACL*.

Adi Haviv, Jonathan Berant, and Amir Globerson. 2021. BERTese: Learning to speak to BERT. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*, pages 3618–3623, Online. Association for Computational Linguistics.

Neil Houlsby, Andrei Giurgiu, Stanislaw Jastrzebski, Bruna Morrone, Quentin De Laroussilhe, Andrea Gesmundo, Mona Attariyan, and Sylvain Gelly. 2019. Parameter-efficient transfer learning for NLP. In *International Conference on Machine Learning (ICML)*.

Edward J Hu, yelong shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. 2022. LoRA: Low-rank adaptation of large language models. In *International Conference on Learning Representations (ICLR)*.

Minqing Hu and Bing Liu. 2004. Mining and summarizing customer reviews. In *ACM SIGKDD international conference on Knowledge discovery and data mining*.

Zhengbao Jiang, Frank F Xu, Jun Araki, and Graham Neubig. 2020. How can we know what language models know? *Transactions of the Association of Computational Linguistics (TACL)*.

Rabeeh Karimi Mahabadi, Luke Zettlemoyer, James Henderson, Marzieh Saeidi, Lambert Mathias, Veselin Stoyano, and Majid Yazdani. 2022. Perfect: Prompt-free and efficient few-shot learning with language models. In *Annual Meeting of the Association for Computational Linguistics*.

Daniel Khashabi, Snigdha Chaturvedi, Michael Roth, Shyam Upadhyay, and Dan Roth. 2018. Looking beyond the surface: A challenge set for reading comprehension over multiple sentences. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 252–262.

Brian Lester, Rami Al-Rfou, and Noah Constant. 2021. The power of scale for parameter-efficient prompt tuning. In *Empirical Methods in Natural Language Processing (EMNLP)*, pages 3045–3059.

Hector J Levesque, Ernest Davis, and Leora Morgenstern. 2011. The Winograd schema challenge. In *AAAI Spring Symposium: Logical Formalizations of Commonsense Reasoning*, volume 46, page 47.

Robert L Logan IV, Ivana Balažević, Eric Wallace, Fabio Petroni, Sameer Singh, and Sebastian Riedel. 2021. Cutting down on prompts and parameters: Simple few-shot learning with language models. *arXiv preprint arXiv:2106.13353*.

Bo Pang and Lillian Lee. 2004. A sentimental education: Sentiment analysis using subjectivity summarization based on minimum cuts. In *Association for Computational Linguistics (ACL)*.

Bo Pang and Lillian Lee. 2005. Seeing stars: Exploiting class relationships for sentiment categorization with respect to rating scales. In *Association for Computational Linguistics (ACL)*.

Ethan Perez, Douwe Kiela, and Kyunghyun Cho. 2021. True few-shot learning with language models. In *Advances in Neural Information Processing Systems (NeurIPS)*.

Mohammad Taher Pilehvar and Jose Camacho-Collados. 2019. WiC: The word-in-context dataset for evaluating context-sensitive meaning representations. In *Proceedings of NAACL-HLT*.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J Liu. 2020. Exploring the limits of transfer learning with a unified text-to-text Transformer. *The Journal of Machine Learning Research (JMLR)*, 21(140).

Pranav Rajpurkar, Jian Zhang, Konstantin Lopyrev, and Percy Liang. 2016. SQuAD: 100,000+ questions for machine comprehension of text. In *Empirical Methods in Natural Language Processing (EMNLP)*.

Melissa Roemmele, Cosmin Adrian Bejan, and Andrew S. Gordon. 2011. Choice of plausible alternatives: An evaluation of commonsense causal reasoning. In *2011 AAAI Spring Symposium Series*.

Timo Schick, Helmut Schmid, and Hinrich Schütze. 2020. Automatically identifying words that can serve as labels for few-shot text classification. In *International Conference on Computational Linguistics (COLING)*.

Timo Schick and Hinrich Schütze. 2021a. Exploiting cloze questions for few-shot text classification and natural language inference. In *European Chapter of the Association for Computational Linguistics (EACL)*.

Timo Schick and Hinrich Schütze. 2021b. It's not just size that matters: Small language models are also few-shot learners. In *North American Chapter of the Association for Computational Linguistics (NAACL)*.

Noam Shazeer. 2020. Glu variants improve transformer.

Taylor Shin, Yasaman Razeghi, Robert L. Logan IV, Eric Wallace, and Sameer Singh. 2020. AutoPrompt: Automatic prompt construction for masked language models. In *Empirical Methods in Natural Language Processing (EMNLP)*.

Richard Socher, Alex Perelygin, Jean Wu, Jason Chuang, Christopher D. Manning, Andrew Ng, and Christopher Potts. 2013. Recursive deep models for semantic compositionality over a sentiment treebank. In *Empirical Methods in Natural Language Processing (EMNLP)*.

Ellen M Voorhees and Dawn M Tice. 2000. Building a question answering test collection. In *the 23rd annual international ACM SIGIR conference on Research and development in information retrieval*.

Alex Wang, Yada Pruksachatkun, Nikita Nangia, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel R. Bowman. 2019a. SuperGLUE: A stickier benchmark for general-purpose language understanding systems. In *Advances in Neural Information Processing Systems (NIPS)*.

Alex Wang, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel R Bowman. 2019b. GLUE: A multi-task benchmark and analysis platform for natural language understanding. In *International Conference on Learning Representations (ICLR)*.

Alex Warstadt, Amanpreet Singh, and Samuel R. Bowman. 2019. Neural network acceptability judgments. *Transactions of the Association of Computational Linguistics (TACL)*, 7.

Janyce Wiebe, Theresa Wilson, and Claire Cardie. 2005. Annotating expressions of opinions and emotions in language. *Language resources and evaluation*, 39(2-3).

Adina Williams, Nikita Nangia, and Samuel Bowman. 2018. A broad-coverage challenge corpus for sentence understanding through inference. In *North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL-HLT)*.

Weizhe Yuan, Graham Neubig, and Pengfei Liu. 2021. Bartscore: Evaluating generated text as text generation. In *Advances in Neural Information Processing Systems*, volume 34, pages 27263–27277. Curran Associates, Inc.

Elad Ben Zaken, Shauli Ravfogel, and Yoav Goldberg. 2022. Bitfit: Simple parameter-efficient fine-tuning for transformer-based masked language-models. In *Association for Computational Linguistics (ACL)*.

Ningyu Zhang, Luoqiu Li, Xiang Chen, Shumin Deng, Zhen Bi, Chuanqi Tan, Fei Huang, and Huajun Chen. 2022. Differentiable prompt makes pre-trained language models better few-shot learners. In *International Conference on Learning Representations*.

Sheng Zhang, Xiaodong Liu, Jingjing Liu, Jianfeng Gao, Kevin Duh, and Benjamin Van Durme. 2018. ReCoRD: Bridging the gap between human and machine commonsense reading comprehension. *arXiv preprint 1810.12885*.

A Datasets

We use datasets from GLUE (Wang et al., 2019b), SuperGLUE (Wang et al., 2019a), and a number of other sentence classification datasets.

For SST-2 (Socher et al., 2013), SST-5 (Socher et al., 2013), MR (Pang and Lee, 2005), CR (Hu and Liu, 2004), MPQA (Wiebe et al., 2005), Subj (Pang and Lee, 2004), TREC (Voorhees and Tice, 2000), CoLA (Warstadt et al., 2019), MNLI (Williams et al., 2018), SNLI (Bowman et al., 2015), QNLI (Rajpurkar et al., 2016), RTE (Dagan et al., 2005; Bar Haim et al., 2006; Giampiccolo et al., 2007; Bentivogli et al., 2009), MRPC (Dolan and Brockett, 2005), QQP[7] and STS-B (Cer et al., 2017), we refer to Gao et al. (2021) for their test settings. For BoolQ (Clark et al., 2019), CB (De Marneffe et al., 2019), COPA (Roemmele et al., 2011), MultiRC (Khashabi et al., 2018), ReCoRD (Zhang et al., 2018), WiC (Pilehvar and Camacho-Collados, 2019) and WSC (Levesque et al., 2011), we take their original development sets as the test sets.

B Experimental Details

B.1 Hyper-parameter selection

We take batch sizes from {2, 4, 8} for all few-shot experiments. For fine-tuning, we take learning rates from {7e-5, 1e-4, 2e-4}. For prompt-based fine-tuning, we take learning rates from {2e-5, 6e-5, 9e-5}, which are selected by pre-experiments on the SST-2 and SNLI datasets. For each trial, we follow Gao et al. (2021) and set the training steps as 1000, validation steps as 100, then pick the best model based on the validation results.

B.2 Automatic label sequence generation

For automatic label sequence generation, we use `T5-large`, limiting the maximum length of 20 tokens (AutoSeq) and one token (AutoWord). Considering the trade-off between efficiency and effectiveness, we set beam search width to 50 and set n to 20. Given that the number of experiments is relatively large in automatic generation, we fix the batch size as 8 and the learning rate as 6e-5 when training the model over the top n label mappings.

Besides our $\mathcal{T}$ for one-sentence classification tasks[8] and NLI tasks mentioned in Section 3, we also design more $\mathcal{T}$, always a simple concatenation of input fields and the `[MASK]` token, for other complicated tasks. For BoolQ, $\mathcal{T}$ is "x_1? `[MASK]`, x_2". For COPA, $\mathcal{T}$ is "x_1 x_2? x_3? `[MASK]`, x_4". For MultiRC, $\mathcal{T}$ is "x_2 `[MASK]`, x_3 x_1". For WiC, $\mathcal{T}$ is "x_1 x_2 'x_3' `[MASK]`". Since ReCoRD and WSC can be easily and intuitively transformed into fill-in-the-blank tasks, we follow Schick and Schütze (2021b) and do not process the automatic label sequence generation for them. To make the input closer to pre-training, we refer to Gao et al. (2021) for the implementation details of prompts.

[7] https://www.quora.com/q/quoradata/

[8] One exception: MPQA consists of incomplete sentences, so we adopt manual template without engineering.

	SST-2	SNLI	QQP	MultiRC
Manual w/o templates	90.2	**64.1**	**57.7**	56.2
Manual with eng.	**90.8**	**64.1**	56.1	**57.5**

Table C.1: Comparison between manual label words without templates (so the input is the same as AutoSeq), and manual prompts with deliberate engineering.

C Analysis of Templates

Table C.1 gives the results of using only manual label words with engineering and no templates (so the mask token is concatenated the same way as AutoSeq). This can be seen as the *null prompts* from Logan IV et al. (2021). Our results further validate that null prompts perform comparably or even better to manual prompts in most cases.

D Manual Prompts

Table D.1 demonstrates all the manual templates and label words adopted by us. We basically follow Logan IV et al. (2021) for these prompts. For the tasks that are not covered by Logan IV et al. (2021), we manually write one prompt for each of them, using only our intuition.

E Manual Prompts with Engineering

Table E.1 gives all the manual templates and label words with careful engineering (Gao et al. (2021) for GLUE and Schick and Schütze (2021b) for SuperGLUE) that we use in our experiments.

Table E.2 compares the full results of manual prompts with engineering to our AutoSeq. Overall, AutoSeq performs comparably or even better compared with manual prompts, particularly for tasks where developing solid manual prompts is less instinctive (e.g., TREC, QNLI, QQP and COPA).

Task	Template	Label words
SST-2	<S_1> Overall my impression is `[MASK]` .	positive: good, negative: bad
SST-5	<S_1> Overall my impression is `[MASK]` .	v.positive: very good, positive: good, neutral: not bad, negative: bad, v.negative: very bad
MR	<S_1> Overall my impression is `[MASK]` .	positive: good, negative: bad
CR	<S_1> Overall my impression is `[MASK]` .	positive: good, negative: bad
MPQA	<S_1> Overall my impression is `[MASK]` .	positive: good, negative: bad
Subj	<S_1> The sentence is `[MASK]` .	subjective: subjective, objective: objective
TREC	<S_1> The question is about `[MASK]` .	abbreviation: abbreviation, entity: entity, description: description human: human, location: location, numeric: numeric
COLA	<S_1> The grammar is `[MASK]` .	grammatical: acceptable, not_grammatical: unacceptable
MNLI	Premise: <S_2> Hypothesis: <S_1> Label: `[MASK]`	entailment: yes, netural: maybe, contradiction: no
SNLI	Premise: <S_2> Hypothesis: <S_1> Label: `[MASK]`	entailment: yes, netural: maybe, contradiction: no
QNLI	Question: <S_1> Sentence: <S_2> Label: `[MASK]`	entailment: yes, not_entailment: no
RTE	Premise: <S_1> Hypothesis: <S_2> Label: `[MASK]`	entailment: yes, not_entailment: no
MRPC	<S_1> and <S_2> are the `[MASK]` .	equivalent: same, not_equivalent: different
QQP	<S_1> and <S_2> are the `[MASK]` .	equivalent: same, not_equivalent: different
STS-B	<S_1> and <S_2> are the `[MASK]` .	y_u: same, y_l: different
BoolQ	Passage: <S_1> Question: <S_2> Answer: `[MASK]` .	True: true, False: false
CB	Premise: <S_1> Hypothesis: <S_2> Label: `[MASK]`	entailment: yes, netural: maybe, contradiction: no
COPA	Premise: <S_3> Question: <S_4> Choice1: <S_1> Choice2: <S_2> Answer: `[MASK]` .	Alternative 1: Choice1, Alternative 2: Choice2
MultiRC	Paragraph: <S_1> Question: <S_2> Answer: <S_3> Label: `[MASK]`	True: true, False: false
ReCoRD	<S_1> <S_3>	
WiC	'<S_3>' in <S_1> and '<S_3>' in <S_2> are the `[MASK]` .	True: same, False: different
WSC	<S_1> <S_3> is `[MASK]` .	

Table D.1: Manual templates and label words following Logan IV et al. (2021). Note that for ReCoRD and WSC we follow Schick and Schütze (2021b) and do not design the label words for them.

F Automatically Generated Label Sequences

We demonstrate the top 1 automatically generated label sequences before and after re-ranking with contrastive probability for all tasks in Table F.1. It can be observed that our contrastive probability draws a strong distinction between different classes, especially for those multi-classification tasks like SST-5 and TREC, in which our beam search tends to find the same sequence whatever the class is.

Generally speaking, the generated results after re-ranking conform with our intuition in a majority of single and two-sentence tasks. For more complicated ones, such as COPA and WiC, the generated label sequences can be counterintuitive, calling for a more elegant solution in the future.

Task	Template	Label words
SST-2	<S_1> It was `[MASK]` .	positive: great, negative: terrible
SST-5	<S_1> It was `[MASK]` .	v.positive: great, positive: good, neutral: okay, negative: bad, v.negative: terrible
MR	<S_1> It was `[MASK]` .	positive: great, negative: terrible
CR	<S_1> It was `[MASK]` .	positive: great, negative: terrible
MPQA	<S_1> It was `[MASK]` .	positive: great, negative: terrible
Subj	<S_1> This is `[MASK]` .	subjective: subjective, objective: objective
TREC	`[MASK]` : <S_1>	abbreviation: Expression, entity: Entity, description: Description human: Human, location: Location, numeric: Number
COLA	<S_1> This is `[MASK]` .	grammatical: correct, not_grammatical: incorrect
MNLI	<S_1> ? `[MASK]` , <S_2>	entailment: Yes, netural: Maybe, contradiction: No
SNLI	<S_1> ? `[MASK]` , <S_2>	entailment: Yes, netural: Maybe, contradiction: No
QNLI	<S_1> ? `[MASK]` , <S_2>	entailment: Yes, not_entailment: No
RTE	<S_1> ? `[MASK]` , <S_2>	entailment: Yes, not_entailment: No
MRPC	<S_1> `[MASK]` , <S_2>	equivalent: Yes, not_equivalent: No
QQP	<S_1> `[MASK]` , <S_2>	equivalent: Yes, not_equivalent: No
STS-B	<S_1> `[MASK]` , <S_2>	y_u: Yes, y_l: No
BoolQ	<S_1> Question: <S_2> ? Answer: `[MASK]` .	True: Yes, False: No
CB	<S_1> ? `[MASK]` , <S_2>	entailment: Yes, netural: Maybe, contradiction: No
COPA	<S_1> or <S_2> ? <S_3> , <S_4> `[MASK]` .	
MultiRC	<S_1> Question: <S_2> Is it <S_3> ? `[MASK]` .	True: Yes, False: No
ReCoRD	<S_1> <S_3>	
WiC	<S_1> <S_2> Does <S_3> have the same meaning in both sentences? `[MASK]`	True: Yes, False: No
WSC	<S_1> The pronoun <S_3> refers to `[MASK]` .	

Table E.1: Manual templates and label words with deliberate engineering that we use in our experiments. Note that for COPA, ReCoRD and WSC, we follow Schick and Schütze (2021b) and do not design the label words for them.

	SST-2 (acc)	**SST-5** (acc)	**MR** (acc)	**CR** (acc)	**MPQA** (acc)	**Subj** (acc)	**TREC** (acc)	**CoLA** (Matt.)
Manual with eng.	**90.8** (0.4)	**47.2** (2.4)	**86.1** (0.6)	**90.4** (1.0)	**84.1** (2.4)	91.4 (1.2)	81.3 (4.8)	**9.6** (11.6)
AutoSeq	89.8 (1.1)	42.3 (3.4)	83.9 (1.3)	87.2 (2.5)	82.5 (2.7)	**91.6** (1.9)	**85.2** (4.3)	7.6 (9.9)
	MNLI (acc)	**MNLI-mm** (acc)	**SNLI** (acc)	**QNLI** (acc)	**RTE** (acc)	**MRPC** (F1)	**QQP** (F1)	**STS-B** (Pear.)
Manual with eng.	**55.3** (2.3)	**57.3** (2.4)	**64.1** (4.1)	59.7 (3.4)	**59.1** (4.3)	71.2 (6.6)	56.1 (2.5)	17.7 (12.5)
AutoSeq	51.8 (1.8)	53.9 (2.0)	62.7 (3.7)	**61.3** (4.0)	55.3 (4.9)	**72.3** (4.9)	**66.2** (2.6)	**17.8** (13.4)
	BoolQ (acc)	**CB** (F1)	**COPA** (acc)	**MultiRC** (F1)	**ReCoRD** (F1)	**WiC** (acc)	**WSC** (acc)	**Average**
Manual with eng.	**57.5** (2.1)	**79.7** (5.5)	48.8 (2.5)	57.5 (1.6)	**56.6** (3.5)	**53.4** (4.0)	**62.5** (5.2)	**62.5**
AutoSeq	55.4 (8.1)	76.6 (10.7)	**52.0** (6.8)	**58.2** (0.9)	**56.6** (3.5)	52.6 (2.9)	62.1 (1.4)	62.0

Table E.2: Comparison between manual prompts with engineering and our automatically searched label sequences.

Task	Before re-ranking	After re-ranking
SST-2	(positive/negative) Highly recommended./Thank you.	Highly recommended./Sigh.
SST-5	(very positive/positive/neutral/negative/very negative) Highly recommended./Highly recommended././ Highly recommended./Highly recommended.	A must see./I love this movie./Enjoy!/Sigh./Not recommended.
MR	(positive/negative) Highly recommended./Highly recommended.	Highly recommended./Not for me.
CR	(positive/negative) I love it./Thank you.	I love it./I hate it.
MPQA	(positive/negative) ./.	./Why?
Subj	(subjective/objective) I love it./What do you think?	I love it./The rest is history.
TREC	(abbreviation/entity/description/human/location/numeric) Why?/Why?/./Why?/Why?/.	Discuss!/What is it?/For?/Who is?/USA./15?
CoLA	(grammatical/not_grammatical) ./.	Enjoy!/.
MNLI	(entailment/neutral/contradiction) Yes/Yes/No	I mean/For example/However
SNLI	(entailment/neutral/contradiction) Yes/Yes/Yes	Yes/In this video/Next
QNLI	(entailment/not_entailment) In fact/In fact	In the past/Also
RTE	(entailment/not_entailment) Yes/Yes	Yes/However
MRPC	(equivalent/not_equivalent) Yes/Yes	Yes/Meanwhile
QQP	(equivalent/not_equivalent) Also/Also	So/Also
STS-B	(y_u/y_l) Yes/Yes	Yes/Also
BoolQ	(True/False) Yes/Yes	If so/No
CB	(entailment/neutral/contradiction) Yes/Yes/I mean	Indeed/A: Yes/A: No
COPA	(Alternative 1/Alternative 2) No/No	No/Yes
MultiRC	(True/False) Yes/Yes	The answer is/Also
WiC	(True/False) is used./is used.	is used./is an adjective.

Table F.1: Top 1 automatically generated label sequences before and after re-ranking with contrastive probability for all tasks based on one few-shot split.

Unsupervised Sentence Textual Similarity with Compositional Phrase Semantics

Zihao Wang
Department of CSE
HKUST
Hong Kong, China
zwanggc@cse.ust.hk

Jiaheng Dou and **Yong Zhang** *
BNRist, RIIT, Institute of Internet Industry
Department of Computer Science and Technology
Tsinghua University, Beijing, China
djh19@mails.tsinghua.edu.cn
zhangyong05@tsinghua.edu.cn

Abstract

Measuring Sentence Textual Similarity (STS) is a classic task that can be applied to many downstream NLP applications such as text generation and retrieval. In this paper, we focus on unsupervised STS that works on various domains but only requires minimal data and computational resources. Theoretically, we propose a light-weighted Expectation-Correction (EC) formulation for STS computation. EC formulation unifies unsupervised STS approaches including the cosine similarity of Additively Composed (AC) sentence embeddings (Arora et al., 2017), Optimal Transport (OT) (Kusner et al., 2015), and Tree Kernels (TK) (Le et al., 2018). Moreover, we propose the Recursive Optimal Transport Similarity (ROTS) algorithm to capture the compositional phrase semantics by composing multiple recursive EC formulations. ROTS finishes in linear time and is faster than its predecessors. ROTS is empirically more effective and scalable than previous approaches. Extensive experiments on 29 STS tasks under various settings show the clear advantage of ROTS over existing approaches.[1] Detailed ablation studies demonstrate the effectiveness of our approaches.

1 Introduction

Sentence Textual Similarity (STS) measures the semantic equivalence between a pair of sentences, which is supposed to be consistent with human evaluation (Agirre et al., 2012). STS is also an effective sentence-level semantic measure for many downstream tasks such as text generation and retrieval (Wieting et al., 2019; Zhao et al., 2019; Nikolentzos et al., 2020; Çelikyilmaz et al., 2020). In this paper, we focus on unsupervised STS which is expected to compare texts of various domains but only requires minimal data and computational resources.

There are several typical ways to compute unsupervised STS, including 1) treat each sentence as an embedding by the Additive Composition (**AC**) (Arora et al., 2017) of word vectors, then estimate the STS of two sentences by their cosine similarity; 2) treat each sentence as a probabilistic distribution of word vectors, then measure the distance between distributions. Notably, Optimal Transport (**OT**) (Peyré and Cuturi, 2019)[2] is adopted to compute the STS (Kusner et al., 2015). OT-based approaches search for the best alignment with respect to the **word-level semantics** and result in state-of-the-art solution (Yokoi et al., 2020).

In this paper, we argue that **phrase-level semantics** should also be exploited to fully understand the sentences. For example, "optimal transport" should be considered as a mathematical term rather than two independent words. Specifically, the phrase chunk is composed of lower-level chunks and is usually represented as a node in tree structures. The aforementioned AC and OT-based STS methods are too shallow to include such structures. Tree Kernels (**TK**) (Le et al., 2018) consider the parsed *syntax labels*. However, it boils down to syntax-based but sub-optimal word alignment under our comparison experiment.

Recent advancement of Pretrained Language Models (PLMs) also demonstrate the importance of contextualization (Peters et al., 2018; Devlin et al., 2019; Ethayarajh, 2019). PLMs can be further adopted to STS tasks by supervised fine-tuning (Devlin et al., 2019), under carefully designed transfer learning (Reimers and Gurevych, 2019) or domain-adaptation (Li et al., 2020; Gao et al., 2021). Without those treatments, the performances of PLM-based STSs are observed to be very poor (Yokoi et al., 2020). Meanwhile, PLM-based STSs suffer from high computational costs to fit large amounts of high-quality data, which might prevent them

* Corresponding author.

[1] Our code can be found in https://github.com/zihao-wang/rots.

[2] OT-based distance reflects the dissimilarity between sentences and can also be used as STS.

Proceedings of the 29th International Conference on Computational Linguistics, pages 4976–4995
October 12–17, 2022.

from broader downstream scenarios.

In this paper, we propose a set of concepts and similarities to exploit the phrase semantics in the unsupervised setup. Our contributions are four folds:

Unified formulation We unify three types of unsupervised STS models (AC (Arora et al., 2017), OT (Yokoi et al., 2020) and TK (Le et al., 2018)) by the EC similarity in Section 3. EC similarity uncovers the strengths and weaknesses of the three approaches.

Phrase vectors and their alignment We generalize the idea of word alignment to phrase alignment in Section 4. After the formal definition of Recursive Phrase Partition (RPP), we compose the phrase weights and vectors by those from finer-grained partitions under the *invariant additive phrase composition* and generalize the word alignment to phrase alignment. Empirical observations show that EC similarity is an effective formulation to interpolate the existing unsupervised STS, and yields better performances.

Recursive Optimal Transport We propose the Recursive Optimal Transport Similarity (ROTS) in Section 5 based on the phrase alignment introduced in Section 4. ROTS computes the EC similarity at each phrase partition level and ensembles them. Notably, Prior Optimal Transport (Prior OT) is adopted to guide the finer-grained phrase alignment by the coarser-grained phrase alignment at each expectation step of EC similarity.

Extensive experiments We show the comprehensive performance of ROTS on a wide spectrum of experimental settings in Section 6 and the Appendix, including 29 STS tasks, five types of word vectors, and three typical preprocessing setups. Specifically, ROTS is shown to be better than all other unsupervised approaches including BERT based STS in terms of both effectiveness and efficiency. Detailed ablation studies also show that our constructive definitions are sufficiently important and the hyper-parameters can be easily chosen to obtain the new SOTA performances.

2 Related Work

Embedding the symbolic words into continuous space to present their semantics (Mikolov et al., 2013; Pennington et al., 2014; Bojanowski et al., 2017) is one of the breakthroughs of modern NLP. Notably, it shows that the vector (or semantics) of a phrase can be approximated by the *additive composition* of the vectors of its containing words (Mikolov et al., 2013). Thus, word embeddings can be further utilized to describe the semantics of texts beyond the word level. Several strategies were proposed to provide sentence embeddings.

Additive Composition. Additive composition of word vectors (Arora et al., 2017) forms effective sentence embeddings. The cosine similarity between the sentence embeddings has been shown to be a stronger STS under transferred(Wieting et al., 2016; Wieting and Gimpel, 2018) and unsupervised settings (Arora et al., 2017; Ethayarajh, 2018) than most of the deep learning approaches (Socher et al., 2013; Le and Mikolov, 2014; Kiros et al., 2015; Tai et al., 2015).

Optimal Transport. By considering sentences as distributions of embeddings, the similarity between sentence pairs is the consequence of optimal transport of sentence distributions (Kusner et al., 2015; Huang et al., 2016; Wu et al., 2018; Yokoi et al., 2020). OT models find the optimal alignment with respect to word semantics via their embeddings and have the SOTA performances (Yokoi et al., 2020).

Syntax Information. One possible way to integrate contextual information in a sentence is to explicitly employ syntactic information. Recurrent neural networks (Socher et al., 2013) were proposed to exploit the tree structures in the supervised setting but were sub-optimal than AC-based STS. Meanwhile, tree kernels (Moschitti, 2006; Croce et al., 2011) can measure the similarity between parsing trees. Most recently, ACV-tree kernels (Le et al., 2018) combine word embedding similarities with parsed constituency labels. However, tree kernels compare all the sub-trees and suffer from high computational complexity.

Pretrained Language Models This paradigm produces contextualized sentence embeddings by aggregating the word embeddings repeatedly with the deep neural networks (Vaswani et al., 2017) trained on large corpuses (Devlin et al., 2019). In the unsupervised setting, PLMs are sub-optimal compared to SOTA OT-based models (Yokoi et al., 2020). One of the common strategies to improve the performance is to adjust PLM-generated embedding according to a large amount of external data such as transfer learning (Reimers and Gurevych, 2019),

flow (Li et al., 2020), whitening (Su et al., 2021), and contrastive learning (Gao et al., 2021). However, this domain adaptation paradigm requires a complex training process and the performance is highly affected by the similarity between the target test data and external data (Li et al., 2020; Gao et al., 2021).

3 Unification of Unsupervised STS Methods

Given a pair of sentences $(s^{(1)}, s^{(2)})$, we are expected to estimate their similarity score $s \in [0, 1]$. For sentence $s^{(1)}$ (or $s^{(2)}$), we have vector $\{v_i^{(1)}\}_{i=1}^m$ (or $\{v_j^{(2)}\}_{j=1}^n$) and weight $\{w_i^{(1)}\}_{i=1}^m$ (or $\{w_j^{(2)}\}_{j=1}^n$). We quickly review three types of unsupervised STS in Section 3.1 (see Figure 1 (a-c)), then unify them by the Expectation-Correction similarity in Section 3.2.

3.1 Review of Three Types of STS

Additive Composition (AC) AC methods (Arora et al., 2017; Ethayarajh, 2018) firstly compute the sentence embedding $x^{(\cdot)} = \sum_i w_i^{(\cdot)} v_j^{(\cdot)}$, then estimate the similarity by the cosine similarity $s_{AC} = \cos(x^{(1)}, x^{(2)})$, see Figure 1 (a).

Optimal Transport (OT) Given pairwise word distance matrix $D = D_{ij}$ and two marginal distributions μ_i and ν_i, the optimal transport alignment Γ_{OT} is computed by solving the following minimization problem (Kusner et al., 2015).

$$\Gamma_{OT} = \underset{\Gamma_{ij} \geq 0}{\arg\min} \sum_{ij} \Gamma_{ij} D_{ij}, \quad (1)$$
$$\text{s.t.} \sum_j \Gamma_{ij} = \mu_i, \sum_i \Gamma_{ij} = \nu_j.$$

The higher $\Gamma_{OT,ij}$ means that the alignment from i-th word in $s^{(1)}$ to j-th word in $s^{(2)}$ is preferred, because those two words are semantically closer, see Figure 1 (c). Different choices of D, μ, ν lead to different distances. The SOTA OT-based STS is the Word Rotator's Distance (WRD)[3] (Yokoi et al., 2020), which solves Problem (1) with $D_{ij} = 1 - \cos(w_i^{(1)}, w_j^{(2)})$ and

$$\mu_i = \frac{w_i^{(1)} \|v_i^{(1)}\|_2}{\sum_k w_k^{(1)} \|v_k^{(1)}\|_2}, \quad (2)$$
$$\nu_j = \frac{w_j^{(2)} \|v_j^{(2)}\|_2}{\sum_k w_k^{(2)} \|v_k^{(2)}\|_2}.$$

[3]Without further specification, OT is referred to WRD

The similarity is

$$s_{OT} = \sum_{ij} \Gamma_{OT,ij} \cos(w_i^{(1)}, w_j^{(2)}). \quad (3)$$

WRD is equivalent to AC if and only if each sentence contains one word (Yokoi et al., 2020).

Tree Kernel (TK) General tree kernels compare the syntactic parsing information (Moschitti, 2006; Croce et al., 2011). Recently, ACV-Tree (Le et al., 2018) combines word-level semantics with syntax information by a simplified partial tree kernel (Moschitti, 2006), see Figure 1 (b). Word similarities from the same structure, i.e. NP, are repeatedly counted and thus more important. Then the similarity score can be re-written as

$$s_{TK} = \sum_{ij} \Gamma_{TK,ij} \cos(w_i^{(1)}, w_j^{(2)}) \quad (4)$$

where Γ_{TK} is the normalized weight matrix generated by the tree kernel [4].

3.2 Expectation Correction (EC)

Three approaches discussed above, though motivated in different ways, can be seen as a linear aggregation of pair-wise cosine similarities of words. We unified them into the following EC similarity with two steps called **expectation** and **correction**.

Expectation Both ACV-Tree (see Equation (4)) and OT (see Equation (3)) aggregate pairwise word similarities by the alignment matrix Γ_{TK} and Γ_{OT}. AC also implies the implicit word alignment Γ_{AC}, the cosine similarity can be further decomposed by plugging in the sentence vectors:

$$\cos(x^{(1)}, x^{(2)}) = \frac{\langle \sum_i w_i^{(1)} v_i^{(1)}, \sum_j w_j^{(2)} v_j^{(2)} \rangle}{\|x^{(1)}\| \|x^{(2)}\|}$$
$$= C \sum_{ij} \Gamma_{AC,ij} \cos(v_i^{(1)}, v_j^{(2)}) \quad (5)$$

where $\Gamma_{AC,ij} = \mu_i \nu_j$, μ and ν are defined in Equation (2). This observation connects AC to the *expectation* of word similarities [5]. Hence, the key of **expectation** step, is to compute **inter-sentence** word alignment matrix Γ. Specifically, Γ_{AC} is implicitly induced by weights and vector norms without considering the semantics or syntax between words, Γ_{TK} is constructed by comparing node labels in

[4]In this paper, TK indicates the ACV-Tree kernel

[5]Equation (5) motivates the marginal conditions of WRD in a different way

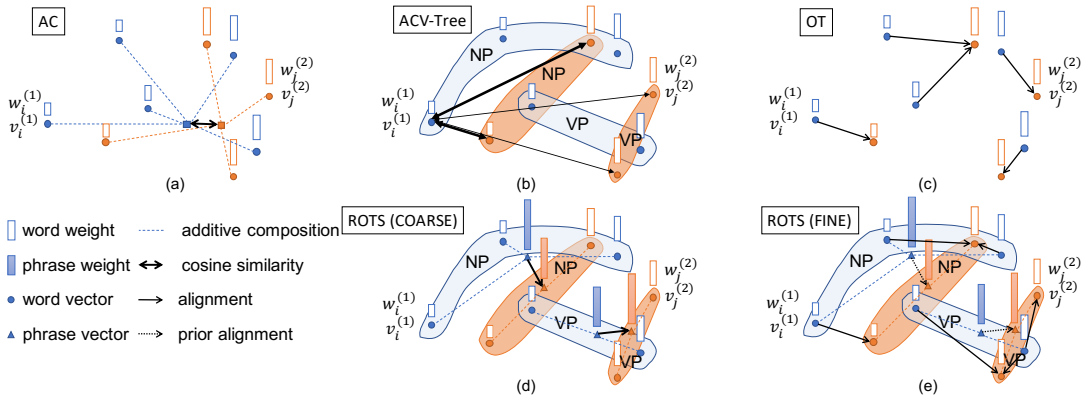

Figure 1: Different unsupervised STS methods with blue elements for $s^{(1)}$ and orange elements for $s^{(2)}$. **(a)** AC (Arora et al., 2017): cosine similarity between additively composed sentence embeddings. **(b)** ACV-Tree (Le et al., 2018): weighted averaging pairwise word similarity. Similarities from $v_i^{(1)}$ to vectors in $s^{(2)}$ are shown. More weights are assigned to pairs contained in the same constituency structure, indicated by thicker arrows. **(c)** OT (Yokoi et al., 2020): compute the optimal transport alignment of words by solving problem (1). **(d)** ROTS at coarser hierarchy: the OT alignment of phrases vectors and weights. **(e)** ROTS at finer hierarchy: fine-level OT alignment based on the prior of coarse-level alignment in (d).

Table 1: The comparison of different approaches.

Method	Inter-sentence Expectation			Intra-sentence Correction	Tiime Complexity
	Word Semantics	Phrase Semantics	Syntax		
AC (Arora et al., 2017; Ethayarajh, 2018)	✗	✗	✗	✓	$O(m+n)$
OT (Kusner et al., 2015; Yokoi et al., 2020)	✓	✗	✗	✗	$O(mn)$
TK (Le et al., 2018)	✗	✗	✓	✗	$O(mn)$
ROTS (ours)	✓	✓	✓	✓	$O(m+n)$

syntax trees, and Γ_{OT} is obtained by optimizing word semantics. (See Table 1)

Correction In Equation (5), the coefficient

$$C = \frac{\sum_k w_k^{(2)} \|v_k^{(2)}\|}{\|\sum_k w_k^{(2)} v_k^{(2)}\|} \frac{\sum_k w_k^{(1)} \|v_k^{(1)}\|}{\|\sum_k w_k^{(1)} v_k^{(1)}\|} = \sqrt{K_1 K_2}$$

also has special interpretation. For the specific sentence $i = 1, 2$, the coefficient K_i can be rewritten as

$$K_i - 1 = \frac{(\sum_k w_k^{(i)} \|v_k^{(i)}\|)^2}{\|\sum_k w_k^{(i)} v_k^{(i)}\|^2} - 1$$
$$= \sum_{k \neq m} \frac{w_k^{(i)} w_m^{(i)} \|v_k^{(i)}\| \|v_m^{(i)}\|}{\|\sum_k w_k^{(i)} v_k^{(i)}\|^2} \left[1 - \cos(v_k^{(i)}, v_m^{(i)})\right].$$

We have $K_i \geq 1$ and the equality holds if and only if all word vectors are in the same direction, i.e. they are semantically close. K_i increases as the semantics of words in a sentence become more diverse. In the latter situation, the sentence similarity tends to be underestimated since unnecessary alignments are forced by the joint distribution. The coefficient C corrects this **intra-sentence** semantics. This correction step distinguishes AC from OT and TK approaches (see Table 1).

Then we introduce the EC similarity by combining E-step and C-step as follows:

Definition 1 (EC similarity)**.** *The EC similarity of STS is defined by:*

$$\tilde{C} \sum_{ij} \Gamma_{ij} \cos(v_i^{(1)}, v_j^{(2)}), \tag{6}$$

*where Γ is the word alignment matrix for the **expectation** and $\tilde{C} = (\alpha C + 1 - \alpha)$ is the coefficient for **correction**, hyper-parameter $\alpha \in [0, 1]$ linearly interpolates the C and 1 and controls the strength of correction.*

4 From Word to Phrase Alignment

In this section, we extend the word alignment to the phrase alignment. We define the phrase partitions of sentences with the recursive structure from any tree. Then we define the phrase weight and vector by the additive composition of sub-phrase (or word) weights and vectors.

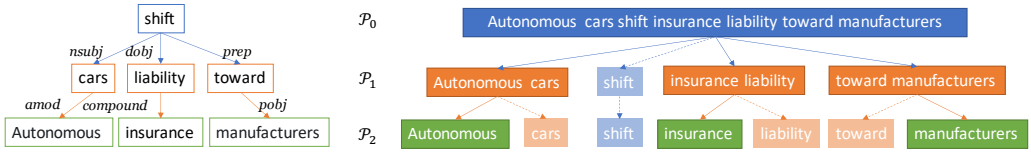

Figure 2: Dependency tree (left) by SpaCy (Honnibal and Montani, 2017) and recursive phrase partitions (right)

4.1 Recursive Phrase Partitions (RPP)

For sentence $s = [t_1, ..., t_n]$ containing n tokens $t_i, 1 \le i \le n$, we define the Recursive Phrase Partitions (RPP) as a set of partitions $\{\mathcal{P}_0, \mathcal{P}_1, ..., \mathcal{P}_L\}$ of the sentence s, where $\mathcal{P}_l$ is the partition at l-th level, $1 \le l \le L$. Specifically, $\mathcal{P}_l = [P_{l,1}, ..., P_{l,q}]$ contains a sequence of phrases, where the q-th phrase $P_{l,q} = s[b_{l,q} : e_{l,q}]$ is the span in s from the beginning index $b_{l,q}$ to the ending index $e_{l,q}$. So we have two properties:

1. Concatenating all phrases recovers the sentence, that is $\oplus_q P_{l,q} = s$, where $\oplus$ is the string concatenation.
2. For two different levels, i.e. $0 \le l_1 < l_2$,[6] any phrase in level l_2 is contained in the unique phrase in level l_1.

In our definition, the $\mathcal{P}_0 = [s]$ and $\mathcal{P}_L = [t_1, .., t_n]$ are the coarsest partition and the finest partition, respectively. The second property guarantees that the recursive phrase partitions can be nested so that each phrase can be recursively divided. RPP can be constructed from any tree representation of the sentence, including constituency tree, dependency tree, or even naive binary separation of token sequences. Figure 2 shows an example of RPP from a dependency tree. Some phrases (such as 'shift' in $\mathcal{P}_2$) are added to satisfy the first property.

4.2 Compositional Phrase Semantics

Once the RPP structure of a sentence is given, we define the vector $\tilde{v}$ and weight $\tilde{w}$ for each phrase. Our definition is invariant with respect to the AC sentence embedding, that is, AC sentence embedding x is invariant to the phrase partition $\mathcal{P}_l$ of the sentence.

$$x = \sum_i^n w_i v_i = \sum_q \tilde{w}_{l,q} \tilde{v}_{l,q},$$

where the phrase weights and vectors are given by

$$\tilde{w}_{l,q} = \sum_{i=b_{l,q}}^{e_{l,q}} w_i, \tilde{v}_{l,q} = \sum_{i=b_{l,q}}^{e_{l,q}} w_i v_i / \tilde{w}_{l,q}.$$

[6] We denote the root is level 0. The level index increases as the tree goes deeper.

In this way, the sentence vector can also be represented by the additive composition of phrase vectors and weights, where each phrase vector can be again composed by the word vectors additively. Our definitions of phrase weights and vectors recursively aggregate the information from finer-grained level (i.e. 'autonomous' and 'cars') information to coarser-grained level (i.e. 'autonomous cars'). Furthermore, our discussion about EC similarity in Section 3.2 at the word level can also be generalized to any phrase partitions. That is, we can use the EC similarity to consider the inter-sentence *phrase* alignment and then correct the intra-sentence *phrase* semantics of each partition.

5 Recursive Optimal Transport and STS

In this section, we connect the dots by applying EC similarity in Section 3.2 to phrase alignment in Section 4 on tree structures. Specifically, we present Recursive Optimal Transport Similarity (ROTS) which computes the phrase alignment at each $(l+1)$-th level phrase partition with the guidance of the phrase alignment at the l-th level.

5.1 Prior Optimal Transport (Prior OT)

Prior OT (Zhao et al., 2020) was firstly proposed to pass prior information when minimizing the entropy-regularized Wasserstein loss. When it comes to the OT-based STS, we re-consider the objective function in Problem (1) with an additional prior alignment Π:

$$\sum_{ij} \Gamma_{ij} D_{ij} + \epsilon \mathrm{KL}(\Gamma \| \Pi), \tag{7}$$

where $\mathrm{KL}(\Gamma\|\Pi) = -\sum_{ij} \Gamma_{ij} \log \Pi_{ij} - H(\Gamma)$ is the KL-divergence between the phrase alignment Γ and the prior alignment Π, and $H(\cdot)$ is the entropy. ϵ is the hyper-parameter that controls how close the obtained Γ^* is to Π. When $\epsilon = 0$, Equation (7) falls back to Equation (1), and when ϵ is sufficiently large, the optimal Γ^* is sufficiently close to Π in terms of KL-divergence.

Notably, the objective in Equation (7) can be minimized by the Sinkhorn algorithm (Cuturi,

2013; Zhao et al., 2020). Compared to tree kernels (Moschitti, 2006; Croce et al., 2011; Le et al., 2018), Sinkhorn algorithm is based on matrix operations such that it can be accelerated by GPUs (Cuturi, 2013). Sinkhorn algorithm has time complexity $O(mn/\epsilon^2)$ (Dvurechensky et al., 2018). In our practice, we usually choose the large prior strength, i.e. $\epsilon > 1$ that allows faster convergence.

We can interpolate WRD and AC with the help of Prior OT under EC similarity.

Example 1 (EC Interpolation of WRD and AC)**.** *Given a prior matrix $\Pi = \Gamma_{AC}$, we first compute the alignment Γ_ϵ by minimizing Equation (7) with WRD's choice of D, μ, ν in Equation (2). Then we compute the EC interpolation similarity by*

$$\text{Interp} = \tilde{C} \sum_{ij} \Gamma_{\epsilon,ij} \cos(v_i^{(1)}, v_j^{(2)}),$$

where $\epsilon > 0$ is the prior strength in Equation (7). When $(\alpha, \epsilon) = (0, 0)$, $1 - \text{Interp} = s_{OT}$ [7]. When $(\alpha, \epsilon) = (1, +\infty)$, $\text{Interp} = s_{AC}$.

5.2 Recursive Optimal Transport Similarity

Given two sentences $s^{(1)}, s^{(2)}$ with their RPPs $\{\mathcal{P}_0^{(1)}, \mathcal{P}_1^{(1)}, ..., \mathcal{P}_{L_1}^{(1)}\}$ and $\{\mathcal{P}_0^{(2)}, \mathcal{P}_1^{(2)}, ..., \mathcal{P}_{L_2}^{(2)}\}$, ROTS considers partition pairs $(\mathcal{P}_k^{(1)}, \mathcal{P}_k^{(2)})$ from the coarsest $k = 0$ level to the finest $k = d \leq \min(L_1, L_2)$ level, where d is a hyper-parameter. Given the computed k-th alignment matrix $\Gamma^{(k)}$ of $(\mathcal{P}_k^{(1)}, \mathcal{P}_k^{(2)})$, ROTS constructs the following prior alignment $\Pi^{(k+1)}$ for next EC computation $(\mathcal{P}_{k+1}^{(1)}, \mathcal{P}_{k+1}^{(2)})$.

$$\Pi_{m_i n_j}^{(k+1)} = \frac{\mu_{k+1,m_i} \nu_{k+1,n_j} \Gamma_{ij}^{(k)}}{\sum_{\tilde{m}_i \in P_{k,i}^{(1)}, \tilde{n}_j \in P_{k,j}^{(2)}} \mu_{k+1,\tilde{m}_i} \nu_{k+1,\tilde{n}_j}} \quad (8)$$

Specifically, the (i, j) phrase alignment score $\Gamma_{ij}^{(k)}$ at k-th level will be separated to the sub-phrase alignment (m_i, n_j) at the $(k+1)$-th level according to the marginal μ_{k+1,m_i} and ν_{k+1,n_j}, where m_i, n_j are the index of the sub-phrase of i, j respectively. With the coarse-to-fine prior $\Pi^{(k+1)}$, ROTS computes the phrase alignment matrix $\Gamma^{(k+1)}$ at the $(k+1)$-th level by Prior OT (Equation (7)). The computation process of ROTS is shown in Algorithm 1. For $k = 0$, each sentence has a single vector, the alignment matrix $\Gamma^{(0)} = 1$ is a 1×1 matrix. The complexity of ROTS is $O(m + n + d(\rho^d/\epsilon)^2)$ where ρ is the maximum branching number of the tree and is usually small for natural language. When the hyper-parameter d is fixed, the complexity of Algorithm 1 grows linearly with the sentence length m and n (see Table 1).

[7] Interp itself also leads to the identical STS evaluation as s_{OT} in terms of correlation.

Algorithm 1 Recursive OT Similarity

Require: Two sentences $s^{(1)}, s^{(2)}$ with recursive phrase partitions $\{\mathcal{P}_0^{(1)}, \mathcal{P}_1^{(1)}, ...\}$ and $\{\mathcal{P}_0^{(2)}, \mathcal{P}_1^{(2)}, ...\}$, depth d and prior strengths $\epsilon_k, k = 1, ..., d$, correction strength α.

Ensure: ROTS_k at each level k.

```
Prepare the weights and vectors at level 0.
Initialize 0-th level alignment Γ^(0) ← 1.
for k ← 1, ..., d do
    Prepare the weights and vectors at level k.
    Get k-th prior Π^(k) by Eq. (8) from Γ^(k−1).
    Get k-th alignment Γ^(k) by Eq. (7) with ε_k.
    Get ROTS_k by Eq. (6) with C̃ = αC + 1 − α, where C = (Σ_k w_k^(2) ||v_k^(2)|| / ||Σ_k w_k^(2) v_k^(2)||) (Σ_k w_k^(1) ||v_k^(1)|| / ||Σ_k w_k^(1) v_k^(1)||).
end for
```

Our ROTS is featured by finding the finer-level phrase alignment under the guidance of the coarser-level phrase alignment. Unlike the tree kernels (Le et al., 2018) that highly rely on syntax trees and syntax labels, ROTS is based on the EC phrase alignment at different phrase partition levels that are induced by a syntax tree. Specifically, the phrase alignments are obtained from the phrase semantic information, i.e. weights and vectors rather than plain syntax labels (see Table 1).

6 Experiments

We first present the experimental setting of unsupervised STS. Then we conduct the benchmark study of all unsupervised STS approaches. Detailed ablation studies justify the effect of ROTS. In the appendix, further discussions on the impact of word vectors, and preprocessing steps are included.

6.1 Experimental Settings

Text processing SpaCy (Honnibal and Montani, 2017) is a open-source text processing toolkit including rich functionality such as tokenization and dependency parsing. It is very suitable for preprocessing pipelines. The text processing model in `en_core_web_sm` is used.

Word vectors Word2Vec (Mikolov et al., 2013),

GloVe (Pennington et al., 2014), and fastText (Bojanowski et al., 2017) are considered in the unsupervised STS cases. Two word vectors trained on transferred learning settings, i.e. PSL (Wieting et al., 2015) and ParaNMT (Wieting and Gimpel, 2018), are considered in the transferred STS cases. Further information can be found in Appendix A.2.

Preprocessing The scope of our pre-processing steps extends the "vector converters" in (Yokoi et al., 2020). Those preprocessing steps can all be applied to EC similarity and are detailed in Appendix A.3. Three typical setups are selected, including SUP (Ethayarajh, 2018), SWC (Yokoi et al., 2020) and WR (Arora et al., 2017).

Datasets We consider (1) **STSB** dev and test set in STS-Benchmark (Cer et al., 2017); (2) **STS[year]** STS from 2012 to 2016 (Agirre et al., 2012, 2013, 2014, 2015, 2016); (3) **SICK** (Marelli et al., 2014); (4) **Twitter** (Xu et al., 2015). Details can be found in Appendix A.1. Each dataset includes several sub-tasks, and there are 29 tasks in total.

Related baselines Some unsupervised STS baselines are closely related to EC similarity, including COS (SIF (Arora et al., 2017), uSIF (Ethayarajh, 2018)), ACV-Tree (Le et al., 2018), and WRD (Yokoi et al., 2020). WMD (Kusner et al., 2015) is important but not included since WMD has been shown clearly suboptimal to WRD (Yokoi et al., 2020).

Other Unsupervised Baselines BERT's final-layer and last-2-layers embeddings (BERT and BERT-last2ave) (Li et al., 2020) BERTScore (Zhang et al., 2020), DynaMax-Jaccard (Zhelezniak et al., 2019a), Center Kernel Alignment (CKA) (Zhelezniak et al., 2019b) and Kraskov-Stögbauer–Grassberger (Kraskov et al., 2004) (KSG) cross entropy estimation (Zhelezniak et al., 2020).

Default hyper-parameters We summarize the result with different parameters. Results show that excellent scores are achieved with $\alpha = 1$, $d = 4$ and $\epsilon_k = 10, 1 \leq k \leq L$.

6.2 Unsupervised Benchmark

An unsupervised STS benchmark study is conducted over STSB, SICK, and STS by years (STS12-16). Twitter is not included since most of the baselines did not report the score. fastText is chosen as the pretrained word vector.

We re-implement SIF, uSIF and WRD and compare the Pearson's $r \times 100$ in Table 2 together with the ACV-Tree[8] and BERTScore+fastText[9]. Other baselines are compared by Spearman's ρ in Table 3. The clear advantage of ROTS-mean is shown. Our results confirm the finding reported by (Yokoi et al., 2020) that the BERT-based method is sub-optimal under unsupervised settings.

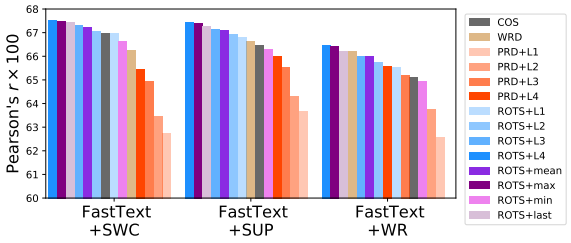

Figure 3: Ablation study for ROTS depth and aggregation. Scores are averaged from STSB, Twitter and SICK.

6.3 Ablation Study

For ablation study, the scores are averaged from scores on the three datasets, including SICK, STSB test, and Twitter. We don't include STS12-16 since they overlap with STSB. *Depths and Aggregations*, *Correction and Prior*, and *Recursive Phrase Partitions* are discussed since they are closely related to ROTS. More experiments on different *word vectors* and *preprocessings* can be found in Appendix B. Uncertainty quantification by the BCa confidence interval (Efron, 1987) on different datasets can be found in Appendix E.

Depths and Aggregations Once all ROTS_k are obtained, we consider different aggregation methods including *mean*, *max*, *min*, *last* and picks the k-th level. The ablation study of depth and aggregation is shown in Figure 3. We report the ROTS results at different levels and different aggregations. We also include the Phrase Rotator's Distance (PRD) at the same recursive phrase partitions as ROTS. PRD-Lk is the special case of ROTS-Lk by setting $\epsilon_k = 0$ and $\alpha = 0$. AC is equivalent to 0-th level ROTS and WRD is the L-th level of PRD so they are included.

ROTS similarities (blue and purple bars) dominate among all other baselines. We can see that the performances of ROTS and PRD increase as their levels get deeper (the related bars are plotted with deeper blue and orange colors). Interestingly,

[8]Scores extracted from (Le et al., 2018), STS13 is not valid since they didn't report on SMT subtask

[9]Scores extracted from (Yokoi et al., 2020)

Table 2: Pearson's $r \times 100$ for ROTS and related unsupervised baselines. Best cases are in boldface.

Similarity	STSB	SICK	STS12	STS13	STS14	STS15	STS16
ACV-Tree (Le et al., 2018)	-	-	61.60	-	72.83	75.80	-
BERTScore fastText (Zhang et al., 2020)	53.86	64.69	51.95	45.86	61.66	69.00	-
SIF(Arora et al., 2017)	70.13	73.20	63.46	59.30	72.95	73.27	70.79
uSIF(Ethayarajh, 2018)	73.47	72.73	63.24	61.41	74.37	76.33	73.47
WRD+SWC(Yokoi et al., 2020)	74.58	67.09	63.80	57.55	71.06	77.65	75.46
WRD+SUP(Yokoi et al., 2020)	74.80	67.67	**64.03**	58.50	71.32	77.65	75.38
WRD+WR(Yokoi et al., 2020)	73.13	68.73	63.81	58.09	70.60	77.28	74.48
ROTS+SWC+mean	**75.33**	71.79	63.91	**62.29**	74.30	**77.96**	**75.95**
ROTS+SUP+mean	74.25	73.13	63.52	61.49	**74.44**	76.75	74.28
ROTS+WR+mean	71.52	**73.84**	63.77	59.58	73.15	73.91	71.97

Table 3: Spearman's $\rho \times 100$ for ROTS and other unsupervised baselines. Best cases are in boldface.

Similarity	STSB	SICK	STS12	STS13	STS14	STS15	STS16
$\text{BERT}_{\text{large}}$ (Devlin et al., 2019)	46.99	53.74	46.89	53.32	49.27	56.54	61.63
$\text{BERT}_{\text{large}}$-last2avg (Li et al., 2020)	59.56	60.22	57.68	61.37	61.02	68.04	70.32
KSG k=10 (Zhelezniak et al., 2020)	-	-	60.40	61.50	68.30	77.00	75.10
MaxPool+KSG k=10 (Zhelezniak et al., 2020)	-	-	59.50	60.20	67.50	75.00	74.10
DynaMax Jaccard (Zhelezniak et al., 2019a)	-	-	61.30	61.70	66.90	76.50	74.70
CKA dCorr (Zhelezniak et al., 2019b)	-	-	60.90	63.40	67.80	76.20	73.40
CKA Gaussian (Zhelezniak et al., 2019b)	-	-	60.80	**64.60**	68.00	76.40	73.80
ROTS+SWC+mean	**72.69**	**62.88**	**63.07**	62.61	**70.73**	**78.06**	**75.74**
ROTS+SUP+mean	71.63	61.81	62.13	61.04	70.85	77.26	74.50
ROTS+WR+mean	69.78	61.39	61.48	59.29	70.19	75.18	73.26

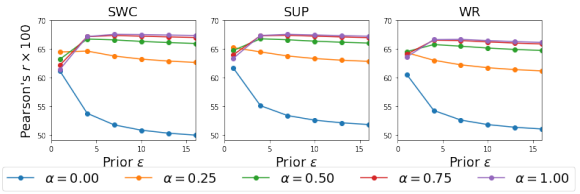

Figure 4: Effects of correction and prior for ROTS-L4 on fastText vectors

PRDs are generally worse than WRD, which indicates that the naive phrase alignment may not be suitable, and may suffer from sub-optimal inter-sentence alignment and intra-sentence semantics. The performance gains of ROTS-Lk from PRD-Lk clearly show that both the coarse-to-fine prior and the EC similarity are important.

Correction Step and Prior We adjust the α in Definition 1 to control the correction effect and the ϵ_k for the prior strength at the k-th phrase partition level. For simplicity, we assume prior strengths ϵ_k are controlled by the single parameter ϵ. We report the Pearson's $r \times 100$ of ROTS-L4 averaged on the three datasets in Figure 4 since ROTS-L4 is the best in Figure 3. As shown in Figure 4, proper correction and prior are essential to produce good performances. The correction step is very important since results without it decrease significantly. This is consistent with the PRD observation in Figure 3. α and ϵ can be chosen easily since the performances is good and consistent if $\alpha > 0.5$ and $\epsilon > 5$.

Recursive Phrase Partitions ROTS relies on the recursive phrase partitions that might be produced from parsing trees. Instead of exhausting the parsers, we consider the simplest binary tree, i.e. the sub-phrase partition is constructed by uniformly splitting each phrase, to show the lower bound of the ROTS performances. We see from Table 4 that the ROTS with spaCy dependency parser performs best in all cases among related baselines. Given the preprocessing setups, we find that the binary tree still outperforms WRD and AC with SUP and SWC setups. For sub-optimal WR setup, ROTS with the binary tree are very close to that in WRD and better than AC. Though preprocessing setups affect the performance, we can observe the performance gain by introducing the recursive phrase partitions given the setup. Therefore, we conclude that the coarse-to-fine prior captures the intra-sentence structures. The performance gain can be observed by even the simplest binary tree.

6.4 More empirical experiments

Some results are presented in the Appendix, including the justification of more choices on preprocessing in Appendix B.2, comparison under transfer and supervised setting in Appendix B.3, compu-

Table 4: Pearson's $r \times 100$ for different parsers. spaCy: ROTS-L4 with the spaCy parser, Binary: ROTS-L4 with a binary tree. The best score is indicated in the boldface and the second highest score is underlined.

Model	spaCy	Binary	WRD	AC
fastText + SUP	**67.45**	67.20	66.63	66.45
fastText + SWC	**67.52**	67.26	66.26	66.97
fastText + WR	**66.47**	66.15	66.20	65.11

tation time in Appendix C, interpolation of WRD and AC by EC similarity in Appendix D.

7 Conclusion

In this paper, we present a new EC similarity of STS that allows flexible adaptation of word-level alignment, which successfully unifies three different unsupervised approaches. By taking advantage of the recursive phrase partitions, we generalize EC similarity to the phrase alignment. Then, we propose ROTS, a new sentence similarity that considers phrase semantics by conducting phrase alignment in a coarse-to-fine order under the coarse-to-fine prior OT. The thorough comparison with unsupervised baselines demonstrates the state-of-the-art performance and technical details of ROTS are fully justified by the ablation study.

8 Acknowledgement

This work was supported by the National Key R&D Program of China (2020AAA0109603).

References

Eneko Agirre, Carmen Banea, Claire Cardie, Daniel M. Cer, Mona T. Diab, Aitor Gonzalez-Agirre, Weiwei Guo, Iñigo Lopez-Gazpio, Montse Maritxalar, Rada Mihalcea, German Rigau, Larraitz Uria, and Janyce Wiebe. 2015. Semeval-2015 task 2: Semantic textual similarity, english, spanish and pilot on interpretability. In *SemEval@NAACL-HLT*.

Eneko Agirre, Carmen Banea, Claire Cardie, Daniel M. Cer, Mona T. Diab, Aitor Gonzalez-Agirre, Weiwei Guo, Rada Mihalcea, German Rigau, and Janyce Wiebe. 2014. Semeval-2014 task 10: Multilingual semantic textual similarity. In *SemEval@COLING*.

Eneko Agirre, Carmen Banea, Daniel M. Cer, Mona T. Diab, Aitor Gonzalez-Agirre, Rada Mihalcea, German Rigau, and Janyce Wiebe. 2016. Semeval-2016 task 1: Semantic textual similarity, monolingual and cross-lingual evaluation. In *SemEval@NAACL-HLT*, pages 497–511.

Eneko Agirre, Daniel M. Cer, Mona T. Diab, and Aitor Gonzalez-Agirre. 2012. Semeval-2012 task 6: A pilot on semantic textual similarity. In *SemEval@NAACL-HLT*, pages 385–393.

Eneko Agirre, Daniel M. Cer, Mona T. Diab, Aitor Gonzalez-Agirre, and Weiwei Guo. 2013. *sem 2013 shared task: Semantic textual similarity. In *Second Joint Conference on Lexical and Computational Semantics*.

Sanjeev Arora, Yingyu Liang, and Tengyu Ma. 2017. A simple but tough-to-beat baseline for sentence embeddings. In *ICLR*.

Piotr Bojanowski, Edouard Grave, Armand Joulin, and Tomás Mikolov. 2017. Enriching word vectors with subword information. *Trans. Assoc. Comput. Linguistics*, pages 135–146.

Asli Çelikyilmaz, Elizabeth Clark, and Jianfeng Gao. 2020. Evaluation of text generation: A survey. *CoRR*, abs/2006.14799.

Daniel Cer, Yinfei Yang, Sheng-yi Kong, Nan Hua, Nicole Limtiaco, Rhomni St. John, Noah Constant, Mario Guajardo-Cespedes, Steve Yuan, Chris Tar, Brian Strope, and Ray Kurzweil. 2018. Universal sentence encoder for english. In *EMNLP*, pages 169–174.

Daniel M. Cer, Mona T. Diab, Eneko Agirre, Iñigo Lopez-Gazpio, and Lucia Specia. 2017. Semeval-2017 task 1: Semantic textual similarity - multilingual and cross-lingual focused evaluation. *CoRR*, abs/1708.00055.

Alexis Conneau and Douwe Kiela. 2018. Senteval: An evaluation toolkit for universal sentence representations. In *LREC*.

Danilo Croce, Alessandro Moschitti, and Roberto Basili. 2011. Structured lexical similarity via convolution kernels on dependency trees. In *EMNLP*, pages 1034–1046.

Marco Cuturi. 2013. Sinkhorn distances: Lightspeed computation of optimal transport. In *NIPS*, pages 2292–2300.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: pre-training of deep bidirectional transformers for language understanding. pages 4171–4186.

Pavel E. Dvurechensky, Alexander Gasnikov, and Alexey Kroshnin. 2018. Computational optimal transport: Complexity by accelerated gradient descent is better than by sinkhorn's algorithm. In *ICML*, pages 1366–1375.

Bradley Efron. 1987. Better bootstrap confidence intervals. *Journal of the American statistical Association*, 82(397):171–185.

Kawin Ethayarajh. 2018. Unsupervised random walk sentence embeddings: A strong but simple baseline. In *Rep4NLP@ACL*, pages 91–100.

Kawin Ethayarajh. 2019. How contextual are contextualized word representations? comparing the geometry of bert, elmo, and GPT-2 embeddings. In *EMNLP-IJCNLP*, pages 55–65.

Rémi Flamary, Nicolas Courty, Alexandre Gramfort, Mokhtar Z. Alaya, Aurélie Boisbunon, Stanislas Chambon, Laetitia Chapel, Adrien Corenflos, Kilian Fatras, Nemo Fournier, Léo Gautheron, Nathalie T.H. Gayraud, Hicham Janati, Alain Rakotomamonjy, Ievgen Redko, Antoine Rolet, Antony Schutz, Vivien Seguy, Danica J. Sutherland, Romain Tavenard, Alexander Tong, and Titouan Vayer. 2021. Pot: Python optimal transport. *Journal of Machine Learning Research*, 22(78):1–8.

Tianyu Gao, Xingcheng Yao, and Danqi Chen. 2021. Simcse: Simple contrastive learning of sentence embeddings. In *EMNLP*, pages 6894–6910.

Matthew Honnibal and Ines Montani. 2017. spaCy 2: Natural language understanding with Bloom embeddings, convolutional neural networks and incremental parsing.

Gao Huang, Chuan Guo, Matt J. Kusner, Yu Sun, Fei Sha, and Kilian Q. Weinberger. 2016. Supervised word mover's distance. In *NIPS*, pages 4862–4870.

Ryan Kiros, Yukun Zhu, Ruslan Salakhutdinov, Richard S. Zemel, Raquel Urtasun, Antonio Torralba, and Sanja Fidler. 2015. Skip-thought vectors. In *NIPS*, pages 3294–3302.

Alexander Kraskov, Harald Stögbauer, and Peter Grassberger. 2004. Estimating mutual information. *Physical review E*, 69(6):066138.

Matt J. Kusner, Yu Sun, Nicholas I. Kolkin, and Kilian Q. Weinberger. 2015. From word embeddings to document distances. In *ICML*, pages 957–966.

Quoc V. Le and Tomas Mikolov. 2014. Distributed representations of sentences and documents. In *ICML*, pages 1188–1196.

Yuquan Le, Zhi-Jie Wang, Zhe Quan, Jiawei He, and Bin Yao. 2018. Acv-tree: A new method for sentence similarity modeling. In *IJCAI*, pages 4137–4143.

Bohan Li, Hao Zhou, Junxian He, Mingxuan Wang, Yiming Yang, and Lei Li. 2020. On the sentence embeddings from pre-trained language models. In *EMNLP*, pages 9119–9130.

Tianlin Liu, Lyle Ungar, and João Sedoc. 2019. Continual learning for sentence representations using conceptors. In *NAACL-HLT*, pages 3274–3279.

Marco Marelli, Luisa Bentivogli, Marco Baroni, Raffaella Bernardi, Stefano Menini, and Roberto Zamparelli. 2014. Semeval-2014 task 1: Evaluation of compositional distributional semantic models on full sentences through semantic relatedness and textual entailment. In *SemEval@COLING*, pages 1–8.

Tomas Mikolov, Ilya Sutskever, Kai Chen, Gregory S. Corrado, and Jeffrey Dean. 2013. Distributed representations of words and phrases and their compositionality. In *NIPS*.

Alessandro Moschitti. 2006. Efficient convolution kernels for dependency and constituent syntactic trees. In *ECML*, pages 318–329.

Jiaqi Mu and Pramod Viswanath. 2018. All-but-the-top: Simple and effective postprocessing for word representations. In *ICLR*.

Giannis Nikolentzos, Antoine J.-P. Tixier, and Michalis Vazirgiannis. 2020. Message passing attention networks for document understanding. In *AAAI*, pages 8544–8551.

Jeffrey Pennington, Richard Socher, and Christopher D. Manning. 2014. Glove: Global vectors for word representation. In *EMNLP*.

Matthew E. Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer. 2018. Deep contextualized word representations. In *NAACL-HLT*, pages 2227–2237.

Gabriel Peyré and Marco Cuturi. 2019. Computational optimal transport. *Found. Trends Mach. Learn.*, 11(5-6):355–607.

Radim Řehůřek and Petr Sojka. 2010. Software Framework for Topic Modelling with Large Corpora. In *Proceedings of the LREC 2010 Workshop on New Challenges for NLP Frameworks*, pages 45–50.

Nils Reimers and Iryna Gurevych. 2019. Sentence-bert: Sentence embeddings using siamese bert-networks. In *EMNLP-IJCNLP*, pages 3980–3990.

Richard Socher, Alex Perelygin, Jean Wu, Jason Chuang, Christopher D. Manning, Andrew Y. Ng, and Christopher Potts. 2013. Recursive deep models for semantic compositionality over a sentiment treebank. In *ACL*, pages 1631–1642.

Jianlin Su, Jiarun Cao, Weijie Liu, and Yangyiwen Ou. 2021. Whitening sentence representations for better semantics and faster retrieval. *CoRR*, abs/2103.15316.

Kai Sheng Tai, Richard Socher, and Christopher D. Manning. 2015. Improved semantic representations from tree-structured long short-term memory networks. In *ACL*, pages 1556–1566.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in neural information processing systems*, pages 5998–6008.

John Wieting, Mohit Bansal, Kevin Gimpel, and Karen Livescu. 2015. From paraphrase database to compositional paraphrase model and back. *TACL*, 3.

John Wieting, Mohit Bansal, Kevin Gimpel, and Karen Livescu. 2016. Towards universal paraphrastic sentence embeddings. In *ICLR*.

John Wieting, Taylor Berg-Kirkpatrick, Kevin Gimpel, and Graham Neubig. 2019. Beyond BLEU: training neural machine translation with semantic similarity. In *ACL*, pages 4344–4355.

John Wieting and Kevin Gimpel. 2018. Paranmt-50m: Pushing the limits of paraphrastic sentence embeddings with millions of machine translations. In *ACL*, pages 451–462.

Lingfei Wu, Ian En-Hsu Yen, Kun Xu, Fangli Xu, Avinash Balakrishnan, Pin-Yu Chen, Pradeep Ravikumar, and Michael J. Witbrock. 2018. Word mover's embedding: From word2vec to document embedding. In *EMNLP*, pages 4524–4534.

Wei Xu, Chris Callison-Burch, and Bill Dolan. 2015. Semeval-2015 task 1: Paraphrase and semantic similarity in twitter (PIT). In *NAACL-HLT*, pages 1–11.

Sho Yokoi, Ryo Takahashi, Reina Akama, Jun Suzuki, and Kentaro Inui. 2020. Word rotator's distance. In *EMNLP*, pages 2944–2960.

Tianyi Zhang, Varsha Kishore, Felix Wu, Kilian Q. Weinberger, and Yoav Artzi. 2020. Bertscore: Evaluating text generation with BERT. In *ICLR*.

Wei Zhao, Maxime Peyrard, Fei Liu, Yang Gao, Christian M. Meyer, and Steffen Eger. 2019. Moverscore: Text generation evaluating with contextualized embeddings and earth mover distance. In *EMNLP-IJCNLP*, pages 563–578.

Xu Zhao, Zihao Wang, Hao Wu, and Yong Zhang. 2020. Semi-supervised bilingual lexicon induction with two-way interaction. In *EMNLP*, pages 2973–2984.

Vitalii Zhelezniak, Aleksandar Savkov, and Nils Hammerla. 2020. Estimating mutual information between dense word embeddings. In *ACL*, pages 8361–8371.

Vitalii Zhelezniak, Aleksandar Savkov, April Shen, Francesco Moramarco, Jack Flann, and Nils Y. Hammerla. 2019a. Don't settle for average, go for the max: Fuzzy sets and max-pooled word vectors. In *ICLR*.

Vitalii Zhelezniak, April Shen, Daniel Busbridge, Aleksandar Savkov, and Nils Hammerla. 2019b. Correlations between word vector sets. In *EMNLP-IJCNLP*, pages 77–87.

A Extended Experimental Setup information

A.1 Dataset details

- **STSB** dev and test set in STS-Benchmark (Cer et al., 2017). It can be downloaded directly from (Ethayarajh, 2018)'s implementation [10].
- **STS[year]** STS from 2012 to 2016 (Agirre et al., 2012, 2013, 2014, 2015, 2016). STS12 contains 5 subtasks, STS13 contains 4 subtasks, STS14 contains 6 subtasks, STS15 contains 5 subtasks and STS16 contains 5 subtasks. The reported score is averaged from scores for related subtasks. It can be obtained by (Conneau and Kiela, 2018)'s implementation [11]. A newer implementation is also available [12].
- **SICK** Semantic relatedness task at SemEval 2014 (Marelli et al., 2014). It can be downloaded directly from (Ethayarajh, 2018)'s implementation [13].
- **Twitter** Paraphrase and semantic similarity in Twitter (PIT) at SemEval 2015 (Xu et al., 2015). This dataset was obtained by emailing the author.

A.2 Pretrained Word Vectors

We list the downloadable links of word vectors used in this paper.

- Word2Vec (Mikolov et al., 2013): We use the pretrained word2vec [14]. However, this file is in `.bin` format. We use gensim (Řehůřek and Sojka, 2010) to convert the file to `.vec` format.
- GloVe (Pennington et al., 2014): We use the 300D GloVe vectors trained on Common Crawl (840B tokens, 2.2M vocabulary) [15].
- fastText (Bojanowski et al., 2017): We use the 300D fastText vectors trained on Common Crawl (600B tokens) without subword information [16].
- PSL (Wieting et al., 2015): We use the pretrained vectors from the author [17].
- ParaNMT (Wieting and Gimpel, 2018): Two versions are provided by the author [18] and we keep the same choice as (Ethayarajh, 2018) [19].

A.3 Preprocessing of word vectors

Other preprocessing setups are discussed as follows: Here we list several preprocessing approaches mentioned in previous research. For those with hyper-parameters, we also give the hyper-parameters used in this paper.

- **Word-level** Each word is associated with one weight. We consider SIF (**W**)eights (Arora et al., 2017) with $a = 10^{-3}$ and (**U**)SIF weights (Ethayarajh, 2018).
- **Vocabulary-level** Vectors are modified based on the vectors of words in the whole vocabulary, e.g. (**A**)ll-but-the-top (Mu and Viswanath, 2018) with $D = 3$, (**C**)onceptor negation (Liu et al., 2019) with $\alpha = 2$.
- **Sentence-level** Vectors are modified by vectors of words in the same sentence, including Dimension-wise (**S**)caling (Ethayarajh, 2018).
- **Corpus-level** Vectors are modified based on all sentences in the corpus, e.g. main component (**R**)emoval (Arora et al., 2017) and (**P**)iece-wise component removal (Ethayarajh, 2018) with $p = 5$.

A.4 Various STS and the required resources

We summarize the usage of data and other resources of popular STS models in Table 5. The key difference between unsupervised settings and other settings is the usage of external data to further train the model. We majorly consider the approaches that can be used without training.

B Extended Experiments

[10] https://github.com/kawine/usif

[11] https://github.com/facebookresearch/SentEval

[12] https://github.com/babylonhealth/corrsim

[13] https://github.com/kawine/usif

[14] GoogleNews-vectors-negative300.bin.gz

[15] http://nlp.stanford.edu/data/glove.840B.300d.zip

[16] https://dl.fbaipublicfiles.com/fasttext/vectors-english/crawl-300d-2M.vec.zip

[17] https://drive.google.com/file/d/0B9w48e1rj-MOck1fRGxaZW1LU2M/view?usp=sharing

[18] https://www.cs.cmu.edu/~jwieting/

[19] https://github.com/kawine/usif/blob/master/paranmt.tar.gz

Table 5: Necessary resources of typical STS model

Model	Pretrain				Data			
	Parser	Weights	Word Vector	Language Model	Training texts	Training labels	Transferred texts	Transferred labels
Unsupervised setting								
BERT layer embedding (Devlin et al., 2019)	✗	✗	✗	✓	✗	✗	✗	✗
BERTScore (Zhang et al., 2020)	✗	✗	✗	✓	✗	✗	✗	✗
Additive composition (Arora et al., 2017; Ethayarajh, 2018)	✗	✓	✓	✗	✗	✗	✗	✗
DynaMax-Jaccard (Zhelezniak et al., 2019a)	✗	✓	✓	✗	✗	✗	✗	✗
Center Kernel Alignment (Zhelezniak et al., 2019b)	✗	✓	✓	✗	✗	✗	✗	✗
KSG cross entropy (Zhelezniak et al., 2020)	✗	✓	✓	✗	✗	✗	✗	✗
OT (Kusner et al., 2015; Yokoi et al., 2020)	✗	✓	✓	✗	✗	✗	✗	✗
ACV-Tree (Le et al., 2018)	✓	✓	✓	✗	✗	✗	✗	✗
ROTS (ours)	✓	✓	✓	✗	✗	✗	✗	✗
Transfer and domain adaptation settings								
SentenceBERT (Reimers and Gurevych, 2019)	✗	✗	✗	✓	✗	✗	*NLI	*NLI
BERT-Flow-*NLI (Li et al., 2020)	✗	✗	✗	✓	✗	✗	*NLI	✗
BERT-Flow-*target (Li et al., 2020)	✗	✗	✗	✓	✓	✗	✗	✗
SimCSE-*NLI (Gao et al., 2021)	✗	✗	✗	✓	✗	✗	*NLI	*NLI
Fine-tuning LM								
BERT-Finetune	✗	✗	✗	✓	✓	✓	✗	✗

B.1 Joint effect of word vectors and preprocessings

We investigate the effects of the word vectors in Table 6. It has been shown that fastText is the best word vector for all three kinds of unsupervised STS regardless of the three pre-processing steps. Furthermore, ROTS performs best compared to AC and WRD when the fastText is chosen.

B.2 Other preprocessing setups

In Table 6, it is also found that SWC is the best performed pre-processing setup in eight out of nine combinations of word vectors and unsupervised STSes. It is also shown that given the SWC as the preprocessing setup, ROTS performs best among three unsupervised STS for all three kinds of word vectors. We also explore the combination of preprocessing setups from word level to corpus level. We provide preliminary results about the impact of preprocessing on ROTS in STSB test split with fastText vectors. We report the score of ROTS-L4 in Table 7.

Table 6: Pearson's $r \times 100$ for benchmark study with different word vectors and pre-processing setups. The best word vector achieved given the same pre-processing setup is indicated by boldface. The best pre-processing setup given the word vector is underlined.

Pre-processing	Word Vectors	Similarity		
		AC	WRD	ROTS
WR	fastText	**65.11**	**66.20**	**66.47**
	GloVe	57.86	61.92	60.73
	Word2Vec	57.35	59.52	58.68
SWC	fastText	**66.97**	**66.26**	**67.52**
	GloVe	66.57	65.21	66.95
	Word2Vec	60.21	60.13	60.66
SUP	fastText	**66.45**	**66.63**	**67.45**
	GloVe	64.08	65.04	65.53
	Word2Vec	57.69	59.49	58.90

By Table 7, we suggest that **C** at the vocabulary level, **S** at the sentence level, **R** at the corpus level are beneficial. It is not clear which one of **U** or **W** in

Table 7: STSB test results by ROTS-L4 with fastText vectors with different preprocessing setups.

Setup	Pearson's $r \times 100$	BCa 95% CI
+W	72.59	[70.02, 74.90]
+WR*	72.34	[69.81, 74.66]
+U	72.62	[70.01, 74.96]
+SU	74.52	[71.90, 76.89]
+SUP*	74.69	[72.01, 77.00]
+SW	74.78	[72.31, 77.11]
+SWP	74.90	[72.32, 77.08]
+SWA	74.97	[72.47, 77.15]
+SUA	75.20	[72.73, 77.43]
+SWR	75.21	[72.75, 77.45]
+SUR	75.35	[72.88, 77.53]
+SWC*	75.66	[73.23, 77.84]
+SUC	75.73	[73.25, 77.95]
+SWRC	75.80	[73.33, 77.92]
+SWRCA	75.86	[73.46, 78.11]
+SURC	75.89	[73.39, 78.08]
+SURCA	75.94	[73.49, 78.18]

the word level for ROTS-L4 is more effective. As a result, we propose to combine the choices for vocabulary, sentence, and corpus levels, i.e. **SCR** for ROTS with **U** or **W**. Moreover, we think the two preprocessing setups in the vocabulary level, i.e. **CA** can also be combined. The best performance of ROTS-L4 is achieved by **SURCA**. The setups suggested by previous words are starred in the table, i.e. **SWC** for WRD (Yokoi et al., 2020), **SUP** (Ethayarajh, 2018) and **WR** (Arora et al., 2017) for AC. Though they may not be the best choice for ROTS-L4, we argue the results presented are sufficient to reveal the advantage of ROTS over other related baselines under various setups.

B.3 Evaluation for transfer and semi-supervised settings

The results can be found in Table 8. ROTS that using the transferred ParaNMT word vector has good performance even compared to Sentence BERT with pretrained BERT large or RoBERTa large (Reimers and Gurevych, 2019), and is better than the domain adaptation settings (Li et al., 2020). It is shown that PLM based models (Cer et al., 2018; Reimers and Gurevych, 2019; Li et al., 2020; Gao et al., 2021) are on par with ROTS with transfered word vectors (Wieting and Gimpel, 2018).

C Computation Speed

We report the computation speed for different similarities on a computer with an Intel i7 CPU of 2.6 GHz with 6 cores and 16 GB RAM. The optimal transport is computed by the POT (Flamary et al., 2021) package [20].

We compare the computation of ROTS with WRD and PRD on STS-B test split (1379 sentence pairs to compute in total). Notably, we focus on the speed by Sinkhorn algorithm (Cuturi, 2013) for two reasons: (1) it has $O(n^2)$ time complexity; (2) it can be easily accelerated by GPU.

Table 9 reports the speed by different OT-based algorithms. We note that the reported speeds for phrase alignment algorithms (PRD and ROTS) also include the time for parsing and constructing the recursive phrase partitions. This additional process brings additional computational overhead and slows down the speed. As a consequence of parsing, we can see that for PRD, #OT/sec is slowed down compared to WRD. However, ROTS is based on Prior OT with larger regularization strength, and each call of `ot.sinkhorn` requires much less time, thus making up the computational overhead by parsing.

D EC Interpolation of WRD and AC

We consider 15 combinations from 5-word vectors and 3 preprocessing setups. For each case, we grid-search 10 values of α by linearly splitting $[0, 1]$, and 10 values of ϵ by logarithmically splitting $[0, 400]$, resulting in 100 runs. The performances of 15 cases are shown in Figure 5. Grid-search results indicate that the proper choice of EC similarity outperforms both WRD and AC, thus showing solidness. Specifically, most of the best interpolation performances appear when $\alpha = 1$ (14 cases) and $\epsilon \in [1, 10]$ (14 cases), which confirms the ablation study in Figure 4. This observation demonstrates the effectiveness of the correction term and indicates that the best choice of the alignment matrix should be chosen carefully.

E Dataset Breakdown Tables and Uncentity Quantification

We provide the breakdown tables related to Figure 3 with different word vectors, e.g. fastText in Table 13, GloVe in Table 14, Word2Vec in Table 15, PSL in Table 16 and ParaNMT in Table 17. We see that WRD performs consistently well on Twitter dataset. For STSB and SICK, ROTS is better, resulting in the best overall performance.

[20] https://github.com/PythonOT/POT

Table 8: Spearman's $\rho \times 100$ for different models in semisupervised and transferred setting

Similarity	STS-B	SICK	STS12	STS13	STS14	STS15	STS16
ParaNMT Transfer							
ROTS+WR+mean (ParaNMT)	78.51	65.90	65.39	63.95	**75.41**	79.90	77.86
ROTS+SWC+mean (ParaNMT)	78.65	65.29	64.88	62.08	74.24	79.16	76.38
SNLI + MNLI transfer							
Sentence BERT(large)	**79.23**	73.75	72.27	78.46	74.90	80.99	76.25
Sentence RoBERTa(large)	79.10	**74.29**	**74.53**	77.00	73.18	81.85	76.82
domain adaptation setting							
BERT (large) Flow *NLI	68.09	64.62	61.72	66.05	66.34	74.87	74.47
BERT (large) Flow target	72.26	62.50	65.20	73.39	69.42	74.92	77.63
SimCSE-BERT *NLI	76.85	72.23	68.40	**82.41**	74.38	**80.91**	**78.56**

Table 9: Comparison of computation speed

Method	Function in POT	Reg	Reg. Strength	#OT/STS	#STS/sec	#OT/sec
WRD	`ot.sinkhorn`	Entropy	0.1	1	208.52	208.52
PRD 4 levels	`ot.sinkhorn`	Entropy	0.1	5	32.80	164.00
ROTS 4 levels	`ot.sinkhorn`	KL Prior	10	5	60.56	302.80

E.1 Three Typical Preprocessings

We provide further information for Table 2, including the Pearson's $r \times 100$ for each individual datasets, plus STSB dev split and Twitter. Still, we focus on fastText vectors, and list three preprocessing setups, e.g. WR in Table 10, SWC in Table 11, SUP in Table 12. We find that ROTS has the best performance in WR and SUP, which is consistent with Table 2, and AC is good with SWC.

E.2 Five Word Vectors

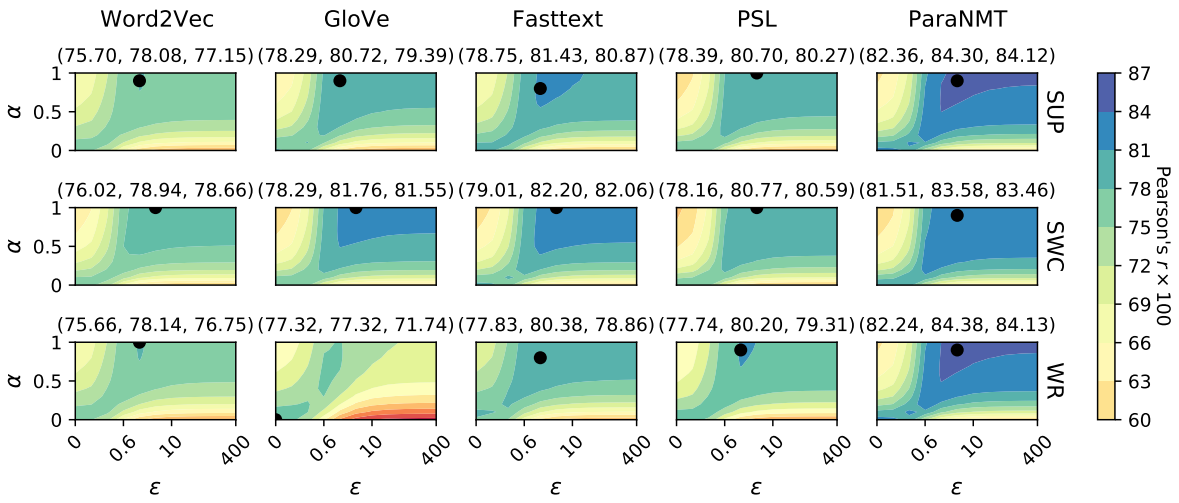

Figure 5: Interpolating AC and WRD by Example 1 on STS-B dev set. For each case, black dots indicate the highest score. The title of each subplot indicates scores by (WRD, Best Interpolation, AC).

Table 10: Breakdown table for benchmark study with WR preprocessing

dataset	subsplit/subtask	AC		WRD		ROTS	
		Pearson's $r \times 100$	BCa 95% CI	Pearson's $r \times 100$	BCa 95% CI	Pearson's $r \times 100$	BCa 95% CI
STSB	test	70.13	[67.35, 72.53]	73.13	[70.40, 75.54]	72.34	[69.78, 74.66]
	dev	78.85	[76.93, 80.56]	77.83	[75.49, 79.87]	79.78	[77.76, 81.48]
Twitter	test	52.01	[47.67, 56.15]	56.73	[52.07, 60.83]	53.32	[48.75, 57.45]
SICK	test	73.20	[71.66, 74.68]	68.73	[67.13, 70.27]	73.75	[72.16, 75.17]
STS12	MSRpar	39.84	[33.68, 45.44]	49.93	[44.04, 54.97]	43.04	[36.93, 48.40]
	MSRvid	86.01	[84.22, 87.59]	82.38	[79.82, 84.57]	86.35	[84.59, 87.91]
	SMTeuroparl	52.41	[44.52, 60.64]	52.01	[45.61, 58.37]	51.95	[44.82, 59.52]
	OnWN	74.31	[70.55, 77.55]	74.59	[71.42, 77.30]	74.43	[70.78, 77.58]
	SMTnews	64.71	[54.76, 73.79]	60.13	[52.37, 67.16]	62.92	[53.37, 72.18]
STS13	FNWN	42.94	[29.66, 53.80]	48.59	[35.52, 58.61]	44.64	[31.36, 55.42]
	headlines	72.88	[69.45, 75.99]	72.49	[68.53, 75.73]	73.52	[69.99, 76.54]
	OnWN	82.36	[79.81, 84.57]	69.74	[65.22, 73.64]	80.67	[77.85, 83.07]
	SMT	39.03	[31.69, 46.65]	41.56	[34.95, 47.44]	40.65	[33.25, 47.89]
STS14	deft-forum	51.99	[45.09, 58.05]	46.53	[38.71, 53.75]	50.78	[43.96, 56.97]
	deft-news	74.54	[68.55, 78.93]	74.62	[68.82, 79.55]	75.52	[69.80, 79.83]
	headlines	68.71	[64.55, 72.33]	67.29	[62.68, 71.45]	69.27	[65.17, 73.02]
	OnWN	84.52	[82.31, 86.32]	76.45	[73.25, 79.17]	83.62	[81.32, 85.56]
	images	81.33	[78.78, 83.43]	80.06	[77.09, 82.50]	81.72	[79.20, 83.82]
	tweet-news	76.62	[72.87, 79.78]	78.65	[75.59, 81.33]	78.27	[74.88, 81.19]
STS15	answers-forums	70.51	[64.96, 75.13]	75.15	[70.11, 79.29]	71.86	[66.44, 76.29]
	answers-students	70.86	[66.85, 74.32]	76.02	[72.59, 79.00]	72.49	[68.70, 75.71]
	belief	68.88	[61.09, 74.18]	77.71	[71.39, 81.99]	70.61	[62.59, 75.62]
	headlines	74.44	[71.27, 77.11]	73.69	[70.17, 76.70]	74.79	[71.52, 77.53]
	images	81.68	[79.23, 83.74]	83.83	[81.28, 85.90]	83.17	[80.75, 85.13]
STS16	answer-answer	47.15	[37.71, 55.50]	60.61	[52.21, 67.26]	53.36	[44.49, 60.90]
	headlines	72.39	[66.12, 77.41]	73.41	[65.76, 79.02]	73.55	[67.06, 78.37]
	plagiarism	82.01	[77.92, 85.28]	82.46	[77.73, 86.24]	82.53	[78.53, 85.90]
	postediting	79.37	[71.53, 83.43]	86.11	[81.33, 89.16]	79.81	[72.99, 83.71]
	question-question	73.03	[66.30, 77.87]	69.79	[61.18, 76.30]	73.79	[67.24, 78.60]
MEAN		68.51	-	69.32	-	**69.40**	-

Table 11: Breakdown table for benchmark study with SWC preprocessing

dataset	subsplit/subtask	AC		WRD		ROTS	
		Pearson's $r \times 100$	BCa 95% CI	Pearson's $r \times 100$	BCa 95% CI	Pearson's $r \times 100$	BCa 95% CI
STSB	test	74.78	[72.31, 77.06]	74.58	[72.08, 76.81]	75.66	[73.17, 77.86]
	dev	82.06	[80.25, 83.67]	78.47	[76.27, 80.45]	81.33	[79.36, 83.08]
Twitter	test	54.01	[49.35, 58.23]	57.10	[52.42, 61.23]	55.56	[50.95, 59.88]
SICK	test	72.12	[70.63, 73.57]	67.09	[65.49, 68.60]	71.33	[69.80, 72.76]
STS12	MSRpar	52.05	[46.24, 57.40]	54.85	[49.29, 59.92]	51.96	[46.18, 57.32]
	MSRvid	87.23	[85.39, 88.76]	80.91	[78.23, 83.20]	85.48	[83.36, 87.22]
	SMTeuroparl	55.44	[49.45, 61.30]	52.75	[46.66, 58.26]	52.94	[46.55, 59.00]
	OnWN	73.66	[69.82, 77.04]	73.80	[70.63, 76.62]	73.52	[69.84, 76.80]
	SMTnews	56.28	[47.71, 64.54]	56.68	[49.28, 63.64]	54.27	[46.13, 62.82]
STS13	FNWN	53.69	[41.93, 62.83]	47.98	[36.27, 57.19]	53.49	[41.82, 63.02]
	headlines	75.92	[72.66, 78.76]	73.67	[70.14, 76.79]	75.38	[71.92, 78.32]
	OnWN	82.89	[80.04, 85.20]	67.57	[62.91, 71.77]	76.66	[73.17, 79.70]
	SMT	41.81	[34.69, 48.45]	40.98	[34.52, 46.79]	42.44	[35.82, 48.50]
STS14	deft-forum	55.57	[48.68, 61.79]	48.98	[41.00, 55.68]	54.31	[47.11, 60.52]
	deft-news	75.92	[70.33, 80.46]	75.63	[70.18, 80.18]	76.13	[70.56, 80.70]
	headlines	71.27	[67.27, 74.83]	69.07	[64.70, 73.00]	71.13	[67.08, 74.74]
	OnWN	85.05	[82.85, 86.93]	75.28	[72.13, 78.04]	81.41	[78.77, 83.66]
	images	83.08	[80.47, 85.19]	79.24	[76.29, 81.77]	82.02	[79.39, 84.25]
	tweet-news	79.02	[75.83, 81.82]	78.16	[75.11, 80.76]	79.30	[76.24, 82.05]
STS15	answers-forums	75.46	[70.52, 79.33]	75.29	[70.37, 79.42]	75.76	[70.83, 79.78]
	answers-students	74.15	[70.73, 77.18]	76.29	[73.12, 79.10]	74.10	[70.68, 77.12]
	belief	78.22	[72.24, 82.23]	77.92	[72.15, 82.19]	78.56	[73.04, 82.59]
	headlines	77.10	[74.12, 79.68]	75.11	[71.84, 78.09]	76.73	[73.70, 79.52]
	images	85.48	[83.09, 87.40]	83.65	[80.95, 85.76]	85.39	[83.04, 87.25]
STS16	answer-answer	60.44	[51.81, 67.64]	63.92	[56.17, 70.26]	61.66	[53.52, 68.76]
	headlines	75.61	[68.93, 80.37]	75.28	[67.87, 80.65]	75.71	[68.87, 80.67]
	plagiarism	83.48	[79.25, 86.80]	81.55	[76.49, 85.31]	82.46	[77.98, 86.01]
	postediting	83.63	[77.69, 87.06]	86.90	[82.05, 89.85]	84.28	[78.84, 87.61]
	question-question	76.09	[68.65, 81.22]	69.66	[61.28, 76.39]	76.08	[68.90, 81.01]
MEAN		71.78	-	69.60	-	**71.21**	-

Table 12: Breakdown table for benchmark study with SUP preprocessing

dataset	subsplit/subtask	AC		WRD		ROTS	
		Pearson's $r \times 100$	BCa 95% CI	Pearson's $r \times 100$	BCa 95% CI	Pearson's $r \times 100$	BCa 95% CI
STSB	test	73.47	[70.90, 75.86]	74.80	[72.27, 77.13]	74.69	[72.12, 77.01]
	dev	80.87	[78.98, 82.55]	78.75	[76.54, 80.72]	81.15	[79.23, 82.92]
Twitter	test	53.15	[48.54, 57.23]	57.41	[52.87, 61.57]	54.88	[50.28, 59.08]
SICK	test	72.73	[71.16, 74.24]	67.67	[66.06, 69.17]	72.77	[71.26, 74.22]
STS12	MSRpar	41.40	[35.15, 46.68]	50.98	[45.25, 56.10]	44.77	[38.66, 50.10]
	MSRvid	86.79	[84.94, 88.35]	83.27	[80.83, 85.32]	87.04	[85.18, 88.62]
	SMTeuroparl	53.29	[46.42, 60.68]	52.73	[46.49, 58.45]	52.31	[45.42, 58.92]
	OnWN	73.53	[69.66, 76.91]	73.85	[70.76, 76.55]	73.56	[69.81, 76.79]
	SMTnews	61.19	[51.71, 70.15]	59.34	[52.19, 66.10]	59.92	[50.88, 69.08]
STS13	FNWN	49.52	[37.76, 59.05]	49.16	[37.65, 58.60]	50.25	[38.73, 59.72]
	headlines	73.73	[70.24, 76.79]	72.95	[69.30, 76.21]	74.11	[70.65, 77.17]
	OnWN	83.15	[80.48, 85.50]	71.12	[66.69, 74.82]	81.14	[78.13, 83.53]
	SMT	39.22	[32.18, 46.23]	40.78	[34.34, 46.68]	40.50	[33.48, 47.03]
STS14	deft-forum	53.39	[46.27, 59.71]	47.60	[39.96, 54.72]	52.21	[45.10, 58.70]
	deft-news	76.08	[70.28, 80.42]	75.38	[69.79, 80.11]	76.91	[71.41, 81.28]
	headlines	69.86	[65.85, 73.29]	68.11	[63.55, 72.19]	70.12	[65.97, 73.73]
	OnWN	85.37	[83.20, 87.16]	77.55	[74.59, 80.08]	84.05	[81.72, 85.98]
	images	83.73	[81.36, 85.71]	81.05	[78.26, 83.51]	83.65	[81.31, 85.67]
	tweet-news	77.78	[74.24, 80.79]	78.24	[75.16, 80.85]	78.86	[75.59, 81.61]
STS15	answers-forums	74.60	[69.78, 78.61]	75.80	[70.74, 79.72]	75.40	[70.34, 79.27]
	answers-students	70.66	[66.65, 74.13]	75.16	[71.79, 78.21]	72.18	[68.21, 75.47]
	belief	76.53	[70.16, 80.98]	78.37	[72.23, 82.48]	77.23	[70.94, 81.64]
	headlines	75.16	[72.06, 77.80]	74.28	[70.94, 77.36]	75.42	[72.23, 78.15]
	images	84.70	[82.47, 86.60]	84.66	[82.23, 86.64]	85.31	[83.12, 87.13]
STS16	answer-answer	54.75	[45.45, 62.55]	62.77	[54.87, 69.23]	58.21	[49.27, 65.50]
	headlines	72.71	[66.44, 77.66]	74.16	[66.53, 79.74]	73.90	[67.05, 78.90]
	plagiarism	82.37	[78.29, 85.55]	82.21	[77.61, 86.05]	82.62	[78.53, 85.92]
	postediting	82.64	[76.07, 86.31]	86.54	[81.65, 89.56]	83.08	[76.90, 86.66]
	question-question	74.91	[67.20, 79.98]	71.21	[63.00, 77.61]	75.57	[68.48, 80.59]
MEAN		70.25	-	69.86	-	**70.75**	-

Table 13: Breakdown table with fastText vectors

Similarity	STSB		Twitter		SICK		MEAN
	Pearson's $r \times 100$	BCa 95% CI	Pearson's $r \times 100$	BCa 95% CI	Pearson's $r \times 100$	BCa 95% CI	Pearson's $r \times 100$
with SUP							
WRD	74.80	[72.27, 77.13]	57.41	[52.87, 61.57]	67.67	[66.06, 69.17]	66.63
AC	73.47	[70.90, 75.86]	53.15	[48.54, 57.23]	72.73	[71.16, 74.24]	66.45
ROTS+L0	73.47	[70.77, 75.76]	53.15	[48.47, 57.13]	72.73	[71.19, 74.25]	66.45
ROTS+L1	73.77	[71.24, 76.08]	53.34	[48.84, 57.51]	73.31	[71.71, 74.74]	66.81
ROTS+L2	74.02	[71.40, 76.34]	53.40	[48.76, 57.63]	73.37	[71.85, 74.81]	66.93
ROTS+L3	74.05	[71.25, 76.44]	54.23	[49.76, 58.44]	73.13	[71.56, 74.58]	67.14
ROTS+L4	74.69	[72.12, 77.01]	54.88	[50.28, 59.08]	72.77	[71.26, 74.22]	67.45
ROTS+mean	74.25	[71.70, 76.56]	53.93	[49.50, 58.13]	73.13	[71.59, 74.58]	67.10
ROTS+max	74.58	[71.87, 76.86]	54.88	[50.39, 59.04]	72.77	[71.27, 74.23]	67.41
ROTS+min	73.36	[70.57, 75.71]	52.74	[48.17, 56.90]	72.74	[71.15, 74.17]	66.28
ROTS+last	74.30	[71.70, 76.64]	54.95	[50.51, 59.01]	72.54	[70.96, 74.03]	67.26
with SWC							
WRD	74.58	[72.08, 76.81]	57.10	[52.42, 61.23]	67.09	[65.49, 68.60]	66.26
AC	74.78	[72.31, 77.06]	54.01	[49.35, 58.23]	72.12	[70.63, 73.57]	66.97
ROTS+L0	74.60	[72.08, 76.92]	53.74	[49.07, 57.98]	71.57	[70.01, 73.01]	66.64
ROTS+L1	74.84	[72.38, 77.09]	53.99	[49.38, 58.26]	72.02	[70.46, 73.44]	66.95
ROTS+L2	75.08	[72.64, 77.42]	54.09	[49.38, 58.41]	72.00	[70.50, 73.40]	67.06
ROTS+L3	75.25	[72.69, 77.57]	54.95	[50.26, 59.19]	71.72	[70.23, 73.17]	67.31
ROTS+L4	75.66	[73.17, 77.86]	55.56	[50.95, 59.88]	71.33	[69.80, 72.76]	67.52
ROTS+mean	75.33	[72.82, 77.56]	54.59	[49.93, 59.00]	71.79	[70.23, 73.22]	67.24
ROTS+max	75.53	[73.08, 77.77]	55.59	[50.96, 59.70]	71.33	[69.83, 72.80]	67.48
ROTS+min	74.80	[72.16, 77.14]	53.44	[48.65, 57.71]	71.58	[70.04, 73.00]	66.61
ROTS+last	75.47	[72.99, 77.64]	55.73	[51.22, 59.76]	71.15	[69.62, 72.62]	67.45
with WR							
WRD	73.13	[70.40, 75.54]	56.73	[52.07, 60.83]	68.73	[67.13, 70.27]	66.20
AC	70.13	[67.35, 72.53]	52.01	[47.67, 56.15]	73.20	[71.66, 74.68]	65.11
ROTS+L0	70.14	[67.47, 72.65]	52.03	[47.45, 56.02]	73.20	[71.63, 74.68]	65.12
ROTS+L1	70.67	[67.98, 73.12]	52.13	[47.62, 56.29]	73.82	[72.30, 75.28]	65.54
ROTS+L2	71.21	[68.59, 73.62]	52.03	[47.21, 56.24]	74.01	[72.52, 75.49]	65.75
ROTS+L3	71.31	[68.51, 73.77]	52.76	[48.20, 56.84]	73.94	[72.42, 75.40]	66.00
ROTS+L4	72.34	[69.78, 74.66]	53.32	[48.75, 57.45]	73.75	[72.16, 75.17]	66.47
ROTS+mean	71.52	[68.99, 73.87]	52.59	[47.95, 56.78]	73.84	[72.27, 75.28]	65.98
ROTS+max	72.17	[69.51, 74.51]	53.32	[48.78, 57.36]	73.75	[72.22, 75.20]	66.41
ROTS+min	70.12	[67.20, 72.64]	51.44	[46.76, 55.55]	73.21	[71.60, 74.66]	64.92
ROTS+last	71.78	[69.21, 74.10]	53.41	[49.05, 57.46]	73.48	[71.92, 74.96]	66.22

Table 14: Breakdown table with GloVe vectors

Similarity	STSB		Twitter		SICK		MEAN
	Pearson's $r \times 100$	BCa 95% CI	Pearson's $r \times 100$	BCa 95% CI	Pearson's $r \times 100$	BCa 95% CI	Pearson's $r \times 100$
with SUP							
WRD	71.97	[69.18, 74.43]	55.63	[50.93, 60.07]	67.52	[65.94, 69.13]	65.04
AC	69.54	[66.72, 72.05]	49.79	[44.88, 54.34]	72.92	[71.40, 74.43]	64.08
ROTS+L0	69.54	[66.68, 72.13]	49.79	[44.83, 54.33]	72.92	[71.34, 74.40]	64.08
ROTS+L1	70.03	[67.29, 72.51]	50.1	[45.13, 54.72]	73.51	[71.95, 74.94]	64.55
ROTS+L2	70.62	[67.89, 73.03]	50.3	[45.20, 54.90]	73.56	[72.03, 75.06]	64.83
ROTS+L3	70.94	[68.20, 73.40]	51.08	[46.17, 55.50]	73.33	[71.78, 74.74]	65.12
ROTS+L4	71.79	[69.19, 74.15]	51.83	[46.91, 56.23]	72.98	[71.44, 74.50]	65.53
ROTS+mean	70.96	[68.26, 73.37]	50.73	[45.79, 55.23]	73.35	[71.79, 74.83]	65.01
ROTS+max	71.57	[68.98, 73.95]	51.83	[47.05, 56.38]	72.98	[71.41, 74.43]	65.46
ROTS+min	69.87	[66.98, 72.43]	49.54	[44.59, 54.14]	72.93	[71.37, 74.40]	64.11
ROTS+last	71.49	[68.90, 73.83]	51.76	[47.01, 56.10]	72.8	[71.21, 74.31]	65.35
with SWC							
WRD	72.34	[69.61, 74.79]	57.31	[52.66, 61.52]	65.99	[64.36, 67.51]	65.21
AC	73.14	[70.51, 75.51]	55.34	[50.75, 59.63]	71.23	[69.69, 72.67]	66.57
ROTS+L0	72.93	[70.19, 75.32]	54.37	[49.51, 58.72]	70.53	[68.94, 72.02]	65.94
ROTS+L1	73.19	[70.55, 75.59]	54.6	[49.87, 58.91]	70.97	[69.39, 72.39]	66.25
ROTS+L2	73.51	[70.95, 75.90]	54.79	[50.00, 59.19]	70.94	[69.41, 72.40]	66.41
ROTS+L3	73.7	[71.06, 76.06]	55.67	[50.86, 59.99]	70.68	[69.15, 72.13]	66.68
ROTS+L4	74.18	[71.66, 76.50]	56.38	[51.78, 60.67]	70.3	[68.77, 71.77]	66.95
ROTS+mean	73.77	[71.24, 76.14]	55.29	[50.58, 59.63]	70.75	[69.23, 72.22]	66.60
ROTS+max	74.01	[71.44, 76.30]	56.38	[51.71, 60.68]	70.3	[68.73, 71.74]	66.90
ROTS+min	73.24	[70.60, 75.63]	54.01	[49.20, 58.35]	70.54	[68.99, 72.06]	65.93
ROTS+last	74.01	[71.43, 76.31]	56.45	[51.79, 60.54]	70.13	[68.60, 71.65]	66.86
with WR							
WRD	69.05	[65.99, 71.79]	48.69	[43.60, 53.49]	68.01	[66.36, 69.60]	61.92
AC	64.67	[61.65, 67.48]	37.56	[32.30, 42.53]	71.36	[69.72, 72.89]	57.86
ROTS+L0	64.67	[61.65, 67.46]	37.56	[32.23, 42.61]	71.36	[69.72, 72.91]	57.86
ROTS+L1	65.39	[62.39, 68.02]	38.24	[33.05, 43.22]	72.12	[70.52, 73.60]	58.58
ROTS+L2	66.32	[63.51, 68.98]	39.12	[33.70, 44.14]	72.47	[70.85, 73.99]	59.30
ROTS+L3	66.46	[63.59, 69.17]	40.58	[35.47, 45.37]	72.58	[71.00, 74.06]	59.87
ROTS+L4	67.9	[65.24, 70.38]	41.77	[36.62, 46.55]	72.52	[70.93, 74.01]	60.73
ROTS+mean	66.75	[63.97, 69.35]	39.63	[34.30, 44.41]	72.36	[70.80, 73.91]	59.58
ROTS+max	67.68	[65.01, 70.09]	41.77	[36.67, 46.65]	72.52	[70.93, 74.02]	60.66
ROTS+min	64.74	[61.59, 67.57]	37.47	[32.12, 42.42]	71.37	[69.71, 72.91]	57.86
ROTS+last	67.36	[64.63, 69.84]	41.31	[36.23, 46.06]	72.23	[70.66, 73.70]	60.30

Table 15: Breakdown table with Word2Vec vectors

Similarity	STSB		Twitter		SICK		MEAN
	Pearson's $r \times 100$	BCa 95% CI	Pearson's $r \times 100$	BCa 95% CI	Pearson's $r \times 100$	BCa 95% CI	Pearson's $r \times 100$
with SUP							
WRD	70.77	[67.85, 73.44]	41.36	[36.04, 46.45]	66.34	[64.73, 67.88]	59.49
AC	69	[66.05, 71.73]	33.12	[27.54, 38.18]	70.96	[69.29, 72.53]	57.69
ROTS+L0	69	[66.05, 71.68]	33.12	[27.55, 38.19]	70.96	[69.31, 72.52]	57.69
ROTS+L1	69.42	[66.39, 72.09]	33.37	[27.90, 38.44]	71.38	[69.76, 72.93]	58.06
ROTS+L2	69.95	[66.94, 72.47]	33.8	[28.39, 38.93]	71.74	[70.13, 73.26]	58.50
ROTS+L3	70.13	[67.15, 72.70]	34.42	[28.79, 39.60]	71.5	[69.86, 73.02]	58.68
ROTS+L4	70.57	[67.73, 73.13]	34.97	[29.38, 40.20]	71.17	[69.51, 72.65]	58.90
ROTS+mean	69.99	[67.02, 72.60]	33.99	[28.39, 39.10]	71.47	[69.88, 73.03]	58.48
ROTS+max	70.57	[67.73, 73.13]	34.97	[29.53, 40.05]	71.17	[69.58, 72.74]	58.90
ROTS+min	68.91	[65.89, 71.57]	33.18	[27.90, 38.37]	70.75	[69.07, 72.25]	57.61
ROTS+last	70.17	[67.38, 72.81]	34.49	[28.84, 39.53]	70.72	[69.09, 72.28]	58.46
with SWC							
WRD	70.64	[67.76, 73.22]	43.46	[38.05, 48.49]	66.29	[64.68, 67.83]	60.13
AC	70.41	[67.47, 73.05]	38.84	[33.44, 44.03]	71.39	[69.81, 72.89]	60.21
ROTS+L0	70.13	[67.23, 72.81]	38.46	[33.08, 43.54]	70.75	[69.17, 72.26]	59.78
ROTS+L1	70.44	[67.59, 73.09]	38.67	[33.18, 43.75]	71.02	[69.46, 72.51]	60.04
ROTS+L2	70.87	[68.02, 73.46]	39.03	[33.49, 43.93]	71.2	[69.61, 72.66]	60.37
ROTS+L3	71.03	[68.06, 73.57]	39.58	[33.98, 44.54]	70.92	[69.34, 72.40]	60.51
ROTS+L4	71.4	[68.58, 73.98]	40.03	[34.62, 45.28]	70.56	[69.02, 72.01]	60.66
ROTS+mean	70.92	[68.06, 73.46]	39.21	[33.82, 44.34]	70.97	[69.42, 72.43]	60.37
ROTS+max	71.4	[68.60, 73.96]	40.03	[34.60, 45.49]	70.56	[68.98, 72.03]	60.66
ROTS+min	70.05	[67.08, 72.66]	38.51	[33.04, 43.56]	70.59	[69.06, 72.13]	59.72
ROTS+last	71.22	[68.32, 73.74]	39.74	[34.26, 44.76]	70.23	[68.61, 71.71]	60.40
with WR							
WRD	70.33	[67.42, 73.02]	40.56	[35.15, 45.83]	67.68	[66.09, 69.20]	59.52
AC	67.86	[64.83, 70.63]	31.85	[26.34, 36.76]	72.33	[70.74, 73.79]	57.35
ROTS+L0	67.87	[64.88, 70.59]	31.86	[26.51, 36.88]	72.33	[70.74, 73.83]	57.35
ROTS+L1	68.4	[65.42, 71.04]	32.07	[26.56, 37.15]	72.68	[71.08, 74.19]	57.72
ROTS+L2	69.09	[66.06, 71.70]	32.4	[26.71, 37.43]	73.04	[71.46, 74.49]	58.18
ROTS+L3	69.44	[66.50, 72.09]	32.96	[27.63, 38.06]	72.85	[71.33, 74.37]	58.42
ROTS+L4	70	[67.15, 72.54]	33.48	[27.87, 38.72]	72.55	[70.94, 74.04]	58.68
ROTS+mean	69.14	[66.29, 71.78]	32.6	[27.20, 37.64]	72.82	[71.28, 74.29]	58.19
ROTS+max	70	[67.12, 72.52]	33.48	[27.99, 38.60]	72.55	[70.99, 74.08]	58.68
ROTS+min	67.83	[64.78, 70.66]	31.94	[26.48, 36.82]	72.12	[70.52, 73.67]	57.30
ROTS+last	69.58	[66.77, 72.17]	33.09	[27.75, 38.21]	72.22	[70.62, 73.70]	58.30

Table 16: Breakdown table with PSL vectors

Similarity	STSB		Twitter		SICK		MEAN
	Pearson's $r \times 100$	BCa 95% CI	Pearson's $r \times 100$	BCa 95% CI	Pearson's $r \times 100$	BCa 95% CI	Pearson's $r \times 100$
with SUP							
WRD	73.78	[71.22, 76.14]	45.72	[40.09, 51.01]	67.83	[66.23, 69.41]	62.44
AC	73.50	[70.84, 75.91]	42.49	[36.81, 47.63]	71.97	[70.38, 73.50]	62.65
ROTS+L0	73.50	[70.87, 75.82]	42.49	[36.95, 47.70]	71.98	[70.40, 73.47]	62.66
ROTS+L1	73.76	[71.14, 76.07]	42.71	[37.24, 47.82]	72.60	[71.05, 74.07]	63.02
ROTS+L2	73.95	[71.37, 76.21]	42.81	[37.08, 47.96]	72.61	[71.11, 74.08]	63.12
ROTS+L3	73.95	[71.31, 76.32]	43.40	[37.81, 48.61]	72.37	[70.84, 73.83]	63.24
ROTS+L4	74.48	[71.98, 76.76]	43.78	[38.22, 48.98]	72.01	[70.47, 73.45]	63.42
ROTS+mean	74.19	[71.60, 76.44]	43.10	[37.48, 48.31]	72.39	[70.85, 73.86]	63.23
ROTS+max	74.42	[71.87, 76.76]	43.78	[38.04, 49.04]	72.01	[70.50, 73.48]	63.40
ROTS+min	73.52	[70.90, 75.85]	42.36	[36.69, 47.46]	71.98	[70.40, 73.47]	62.62
ROTS+last	74.24	[71.63, 76.52]	43.74	[38.23, 48.95]	71.74	[70.23, 73.25]	63.24
with SWC							
WRD	73.01	[70.32, 75.35]	46.01	[40.33, 51.24]	66.73	[65.19, 68.24]	61.92
AC	74.22	[71.73, 76.54]	43.76	[38.16, 48.85]	70.07	[68.54, 71.57]	62.68
ROTS+L0	73.93	[71.36, 76.25]	43.62	[38.07, 48.76]	69.60	[68.00, 71.10]	62.38
ROTS+L1	74.06	[71.43, 76.41]	43.83	[38.19, 49.15]	70.13	[68.56, 71.66]	62.67
ROTS+L2	74.15	[71.55, 76.46]	43.90	[38.10, 49.26]	70.17	[68.64, 71.64]	62.74
ROTS+L3	74.14	[71.58, 76.44]	44.45	[38.75, 49.74]	69.98	[68.41, 71.45]	62.86
ROTS+L4	74.47	[71.87, 76.78]	44.82	[39.21, 50.08]	69.68	[68.16, 71.12]	62.99
ROTS+mean	74.37	[71.81, 76.73]	44.19	[38.65, 49.45]	69.98	[68.41, 71.44]	62.85
ROTS+max	74.41	[71.83, 76.67]	44.83	[39.20, 50.06]	69.68	[68.14, 71.13]	62.97
ROTS+min	74.03	[71.44, 76.36]	43.45	[37.94, 48.60]	69.60	[68.06, 71.15]	62.36
ROTS+last	74.37	[71.84, 76.66]	44.88	[39.25, 50.08]	69.41	[67.83, 70.82]	62.89
with WR							
WRD	72.52	[69.80, 74.89]	45.04	[39.41, 50.32]	68.38	[66.77, 69.92]	61.98
AC	71.13	[68.34, 73.60]	40.18	[34.60, 45.38]	72.37	[70.80, 73.85]	61.23
ROTS+L0	71.13	[68.41, 73.61]	40.18	[34.51, 45.23]	72.37	[70.77, 73.87]	61.23
ROTS+L1	71.57	[68.95, 74.01]	40.45	[34.84, 45.47]	73.02	[71.43, 74.48]	61.68
ROTS+L2	72.05	[69.30, 74.44]	40.59	[34.86, 45.82]	73.07	[71.53, 74.54]	61.90
ROTS+L3	72.22	[69.51, 74.62]	41.19	[35.48, 46.47]	72.90	[71.36, 74.38]	62.10
ROTS+L4	72.93	[70.34, 75.21]	41.61	[35.85, 46.86]	72.60	[71.00, 74.06]	62.38
ROTS+mean	72.34	[69.75, 74.72]	40.87	[35.30, 46.18]	72.88	[71.31, 74.31]	62.03
ROTS+max	72.79	[70.22, 75.05]	41.61	[35.87, 46.84]	72.60	[71.04, 74.06]	62.33
ROTS+min	71.36	[68.54, 73.78]	40.11	[34.50, 45.39]	72.38	[70.84, 73.91]	61.28
ROTS+last	72.63	[70.04, 74.94]	41.43	[36.01, 46.70]	72.35	[70.79, 73.85]	62.14

Table 17: Breakdown table with ParaNMT vectors

Similarity	STSB		Twitter		SICK		MEAN
	Pearson's $r \times 100$	BCa 95% CI	Pearson's $r \times 100$	BCa 95% CI	Pearson's $r \times 100$	BCa 95% CI	Pearson's $r \times 100$
with SUP							
WRD	79.05	[76.85, 81.05]	52.21	[47.20, 56.89]	70.02	[68.52, 71.45]	67.09
AC	79.55	[77.23, 81.61]	46.56	[41.08, 51.56]	73.89	[72.47, 75.24]	66.67
ROTS+L0	79.55	[77.15, 81.57]	46.55	[41.14, 51.50]	73.89	[72.45, 75.26]	66.66
ROTS+L1	79.73	[77.44, 81.77]	46.81	[41.44, 51.79]	74.47	[73.09, 75.83]	67.00
ROTS+L2	79.73	[77.40, 81.71]	47.08	[41.87, 52.25]	74.56	[73.14, 75.88]	67.12
ROTS+L3	79.54	[77.09, 81.61]	48.09	[42.85, 52.77]	74.31	[72.91, 75.68]	67.31
ROTS+L4	79.74	[77.41, 81.77]	48.71	[43.50, 53.50]	73.93	[72.55, 75.33]	67.46
ROTS+mean	79.81	[77.48, 81.83]	47.60	[42.36, 52.44]	74.30	[72.90, 75.65]	67.24
ROTS+max	79.74	[77.43, 81.78]	48.73	[43.51, 53.49]	73.93	[72.51, 75.28]	67.47
ROTS+min	79.29	[76.79, 81.36]	46.22	[40.78, 51.23]	73.89	[72.45, 75.25]	66.47
ROTS+last	79.37	[77.02, 81.41]	48.72	[43.42, 53.61]	73.63	[72.16, 74.95]	67.24
with SWC							
WRD	77.98	[75.65, 79.96]	52.49	[47.49, 57.00]	68.92	[67.41, 70.35]	66.46
AC	79.70	[77.58, 81.65]	46.46	[41.22, 51.39]	71.58	[70.07, 72.98]	65.91
ROTS+L0	79.71	[77.55, 81.57]	46.20	[40.78, 51.13]	71.23	[69.75, 72.64]	65.71
ROTS+L1	79.78	[77.56, 81.63]	46.48	[41.19, 51.48]	71.77	[70.30, 73.17]	66.01
ROTS+L2	79.70	[77.54, 81.64]	46.84	[41.49, 51.89]	71.85	[70.40, 73.23]	66.13
ROTS+L3	79.47	[77.14, 81.51]	47.97	[42.69, 52.76]	71.67	[70.27, 73.04]	66.37
ROTS+L4	79.48	[77.34, 81.41]	48.72	[43.67, 53.62]	71.36	[69.89, 72.76]	66.52
ROTS+mean	79.79	[77.55, 81.74]	47.40	[42.14, 52.49]	71.64	[70.14, 72.99]	66.28
ROTS+max	79.45	[77.24, 81.43]	48.74	[43.61, 53.64]	71.36	[69.89, 72.74]	66.52
ROTS+min	79.63	[77.33, 81.61]	45.90	[40.58, 50.83]	71.24	[69.76, 72.68]	65.59
ROTS+last	79.28	[77.07, 81.23]	48.72	[43.68, 53.51]	71.05	[69.58, 72.47]	66.35
with WR							
WRD	79.03	[76.80, 80.97]	50.82	[45.66, 55.60]	70.94	[69.47, 72.39]	66.93
AC	79.53	[77.28, 81.52]	43.46	[38.01, 48.50]	74.54	[73.07, 75.88]	65.84
ROTS+L0	79.53	[77.20, 81.46]	43.46	[38.11, 48.50]	74.54	[73.08, 75.84]	65.84
ROTS+L1	79.75	[77.50, 81.75]	43.71	[38.19, 48.46]	75.12	[73.70, 76.49]	66.19
ROTS+L2	79.78	[77.45, 81.75]	43.98	[38.40, 49.10]	75.22	[73.82, 76.56]	66.33
ROTS+L3	79.55	[77.15, 81.63]	45.05	[39.73, 49.92]	75.02	[73.61, 76.32]	66.54
ROTS+L4	79.71	[77.41, 81.62]	45.70	[40.42, 50.78]	74.68	[73.29, 76.04]	66.70
ROTS+mean	79.83	[77.57, 81.83]	44.54	[39.07, 49.62]	75.00	[73.62, 76.36]	66.46
ROTS+max	79.71	[77.42, 81.64]	45.71	[40.18, 50.60]	74.68	[73.27, 76.01]	66.70
ROTS+min	79.28	[76.85, 81.34]	43.14	[37.45, 48.16]	74.54	[73.10, 75.87]	65.65
ROTS+last	79.39	[77.03, 81.36]	45.62	[40.29, 50.65]	74.41	[72.98, 75.77]	66.47

A Generalized Method for Automated Multilingual Loanword Detection

Abhijnan Nath*, **Sina Mahdipour Saravani***†, **Ibrahim Khebour**,
Sheikh Mannan, **Zihui Li** and **Nikhil Krishnaswamy**
Situated Grounding and Natural Language (SIGNAL) Lab
Department of Computer Science, Colorado State University
Fort Collins, CO, USA
abhijnan.nath@colostate.edu, sina@cs.utah.edu†,
{ibrahim.khebour,sheikh.mannan,nkrishna}@colostate.edu

Abstract

Loanwords are words incorporated from one language into another without translation. Suppose two words from distantly-related or unrelated languages sound similar and have a similar meaning. In that case, this is evidence of likely borrowing. This paper presents a method to automatically detect loanwords across various language pairs, accounting for differences in script, pronunciation and phonetic transformation by the borrowing language. We incorporate edit distance, semantic similarity measures, and phonetic alignment. We evaluate on 12 language pairs and achieve performance comparable to or exceeding state of the art methods on single-pair loanword detection tasks. We also demonstrate that multilingual models perform the same or often better than models trained on single language pairs and can potentially generalize to unseen language pairs with sufficient data, and that our method can exceed human performance on loanword detection.

1 Introduction

Throughout history, words and phrases have been exchanged between languages around the world (Weinreich, 1954). This can obscure genetic relations between languages (e.g., many people erroneously believe English and French are more closely related than they are) but may also increase comprehension of foreign languages by monoglots (e.g., written French is often partially comprehensible by English speakers).

As Zhang et al. (2021) observe, detecting that a word is loanword is conceptually straightforward: both similar sound and meaning suggests too great a coincidence for different words to have converged by chance[1]. Detecting loanwords computationally has therefore relied on pairwise similarity measures based on transliteration detection and edit distance. However, foundational work in linguistic borrowing, e.g., by Haugen (1950) and Betz (1959), established that when borrowing words into a recipient language, speakers of that language will reproduce existing linguistic patterns when using new words, and the patterns that recipient speakers impose upon a borrowed word vary across time (Köllner and Dellert, 2016), and language pairs. Some languages may adopt a word without much phonetic change due to already-similar phonotactics. Others may fit imported words into a rigid sound pattern, with sometimes significant transformation. Still others may change the meaning. Changes are particular to the language pair, so automatically detecting loanwords between *arbitrary* languages is challenging. However, if successful, such capabilities would also provide benefits to many other NLP tasks such as machine translation, coreference, and named-entity recognition (NER), because common vocabulary, coreferents, or named entities across languages may often be loanwords.

Here, we present a novel method for automated loanword detection between arbitrary language pairs. We build upon existing edit distance-based approaches, incorporate semantic similarity metrics from multilingual language models MBERT (Devlin et al., 2019) and XLM (Conneau et al., 2020), and a method of assessing alignment of phonemes between donor words and loans to account for differences in phonotactics between the relevant languages. We also present and evaluate on the WikLoW (Wiktionary LoanWord) Dataset, currently consisting of 13 language pairs with a high density of loanwords and 3 further language pairs with a lower density of loanwords. We also provide a methodology for expanding the dataset to new language pairs. We demonstrate that our method to detect loanwords across all language pairs in the dataset performs comparably to or better than

*These authors contributed equally to this work.
†This work performed at Colorado State University.
[1]There are exceptions, e.g., Persian *bad* vs. English "bad" and Mbabaram *dog/dúg* vs. English "dog". These are beyond the scope of this paper.

Proceedings of the 29th International Conference on Computational Linguistics, pages 4996–5013
October 12–17, 2022.

existing methods on language-specific loanword detection tasks, that multilingual models can perform better than models trained on individual language pairs, even on data from that pair itself, and that our model can also exceed human performance.[2]

Our method supports both loanword detection and construction of parallel corpora of loanwords for other tasks. Our conclusions suggest that there are some general principles of loanword detection that can be picked up by machine learning models independent of specific languages, and we propose follow-up challenges for NLP research in this area.

2 Related Work

Prior approaches to detecting loanwords computationally follow the intuition mentioned above: that if two words in otherwise not closely related languages have similar meaning and sound similar, then this is likely evidence of borrowing. Van Der Ark et al. (2007) use a Levenshtein-distance based approach to identify language groups and loanwords among languages of Central Asia.

Delz (2013)/Köllner (2021) proposes theoretical approaches to loanword identification based on phylogenetic methods. Zhang et al. (2021) also point out an issue we address herein: loanwords may be transformed to fit the borrowing language's phonology and phonotactics, so pronunciation similarity may be a weaker than ideal method.

Existing data resources relevant to loanwords include the the Automated Similarity Judgment Project (ASJP) database (Brown et al., 2008) and the World Loanword Database (WOLD) (Haspelmath and Tadmor, 2009). Our data source is Wiktionary, which has previously been used in related etymological tasks by De Melo (2014) and Sagot (2017).

One thing we should note is that much work in computational loanword detection and similar tasks is targeted at a specific language or group of languages, e.g., Romance (Cristea et al., 2021; Tsvetkov and Dyer, 2015), Japanese (Takamura et al., 2017), Uyghur (Mi et al., 2014, 2018, 2020, 2021), Spanish (Álvarez-Mellado and Lignos, 2022), Central Asian languages (Van Der Ark et al., 2007), or Turkic and Indo-Iranian (Zhang et al., 2021). Our approach attempts to address the problem at a multilingual level. We use and extend existing work in phonological processing by the NLP community, including the Epitran (Mortensen et al., 2018) and PanPhon (Mortensen et al., 2016) packages for representing phonetic and articulatory features. We incorporate semantic similarity measures from multilingual language models MBERT and XLM, and develop a method of scoring the level of alignment of phonemes between a donor and a loanword to account for differences in language-specific phonology and phonotactics. Our approach in principle supports loanword detection on any pair of languages supported by the upstream packages/models Epitran, MBERT, and XLM, but we discuss how we have (Sec. 3) and can (Sec. 8) also extend our approach to languages that are not at present covered by all of these.

A work at a similar scale, albeit on the slightly different task of *cognate classification*, is Jäger (2018), which evaluates PMI and SVM-based methods over the ASJP database. Cognate detection work generally uses similar methods to those we use here, e.g., semantic and phonetic similarity (Kondrak, 2001), orthographic distance (Mulloni and Pekar, 2006) combined with semantic information (Labat and Lefever, 2019; Lefever et al., 2020), or global constraints (Bloodgood and Strauss, 2017). Work in translation lexicons (e.g., Schafer and Yarowsky (2002)) is also relevant, for the hybrid approach to similarity metrics.

Loanword detection may be useful for phylogenetic reconstruction, like cognate detection (Rama and List, 2019). However, cognates are valid for reconstructing common ancestry; loanwords are not. For historical reconstruction, the two must be separated. Many in the NLP community adopt a definition of "cognate" that subsumes loanwords (e.g., Kondrak (2001)). We do not adopt this definition, and use the linguistic definition that treats loanwords and cognates as distinct.

3 Data Collection

The WikLoW dataset is collected using the process outlined in this section, which can be run for any pair of languages that have loans between them catalogued in Wiktionary, making it easy to expand to new data. We begin by collecting data from Wiktionary categories of the form `[Recipient]_terms_borrowed_from_[Donor]`[3]. Each link in the category is scraped for a loanword in the recipient language and the original form of that word in the donor language.

[2] The codebase is provided at https://github.com/csu-signal/loan-word-detection.

[3] e.g., https://en.wiktionary.org/wiki/Category:Polish_terms_borrowed_from_French

Table 1 shows the language pairs currently contained in the WikLoW dataset, and the number of loans for that pair. There is no global definition of a "low-resourced" language, as this is task-dependent, but we have intentionally tried to represent languages that are not well-represented in large corpora like CC-100 (Conneau et al., 2020). We hereafter refer to language pairs using the format "borrower-donor," e.g., "Hindi-Persian" to refer to Hindi words borrowed from Persian. The directionality between the two languages is important to the pair definition, as only words loaned from the donor language to the borrower are properly considered loanwords. If the direction of the languages were flipped, not only would the class labels be different (the donor word loaned into borrower would not be considered a loanword in the donor language), but while the phonetic and semantic similarities (Secs. 4.2 and 4.3) would probably be the same, the alignment score (Sec. 4.4) would not be, since the output label when training that network is the loanword status, which would be likewise flipped.

Borrower	Donor	# loans
English	French	5074
English	German	2942
Indonesian	Dutch	2665
Polish	French	2055
Romanian	French	2000†
Kazakh*	Russian	1809
Persian	Arabic	1526
Romanian	Hungarian	1460
German	French	1365
Hindi*	Persian	1249
Finnish	Swedish*	1242
Azerbaijani*	Arabic	1116
Mandarin	English	960
Hungarian	German	532
German	Italian	249
Catalan*	Arabic	94

Table 1: Loanword counts per language pair.

*Languages with < 2 billion tokens in the CC-100 corpus.
†Subset of total available loans used.

We also scrape the Wikipedia page listing languages by writing system[4], to include the script name for each language in our datasets. This allows us to filter out words not written in the typical script of the recipient language. For example, some Chinese "loanwords" from English are incorporated keeping the Latin script intact; we don't need machine learning to tell us that these are borrowed terms. Having script information also proves beneficial in later experiments (see Sec. 5).

We also collect all the available lemmas in the donor language, which we use later to calculate the closest phonetic neighbors for each loanword. We also collect homonyms for each loanword where available; homonyms are considered those words that have more than one etymology, where one is a loan from the relevant donor language[5].

Using the Epitran package (Mortensen et al., 2018), we transliterate both loans and original words into the International Phonetic Alphabet (IPA). The Epitran package can be extended to support new languages, as we did here in the case of Finnish, using Omniglot[6] as a resource. Epitran is not a perfect mapping to real pronunciation, especially in the case of abjads such as Arabic script, a point of relevance later (Sec. 4.4, Sec. 7.1).

Having gathered positive examples of loanwords, we need to gather sufficient negative examples to both train an algorithm, and to try and fool the trained algorithm. Negative examples can be:

- **Synonyms**: words with similar meaning to a loanword but pronounced differently, e.g., "driver" vs. *chauffeur*.
- **Hard negatives**: closest phonetic neighbors to a loanword that have different meaning, e.g., "annex" vs. *ânesse*.
- **Randoms**: random pairings where the two words have no discernible phonetic or semantic relationship.

To create the **synonyms** dataset, we take a list of 440 English words, each of which has multiple synonyms associated with it. With the Google Translate API, we translate the main word into one language from our current relevant pair, and each synonym into the other. We then construct word pairs in the donor and recipient language using the Cartesian product of each word with each translated synonym. We remove any duplicates, and any pairs that also occur in the loanword dataset, as we do not want true positives labeled as negatives when training the loanword detection model.

To create the **hard negatives** dataset, we use the PanPhon package (Mortensen et al., 2016) to

[4] https://en.wikipedia.org/wiki/List_of_languages_by_writing_system

[5] One such example is Hindi अगर (/əgər/), which can be both a loan from Persian, meaning "if," and a descendent of Sanskrit अगरु, referring to a type of wood.

[6] https://www.omniglot.com/writing/finnish.htm

compute six edit distances (see Sec. 4.2) between the IPA transcriptions of the gathered loanwords, and up to 20,000 candidate lemmas of the donor language, which are also transliterated into the IPA using Epitran. The result here is that each loanword is paired with up to six candidates that have a low phonetic edit distance but are not the original word in the donor language. We remove duplicates where multiple distance metrics chose the same closest neighbor, and where pairs cooccur with the synonyms or loans datasets.

Finally in the **randoms** dataset, we pair each loan with a random word in the donor language.

4 Similarity Metrics

Every word pair in the WikLoW dataset has measures of textual, phonetic, semantic, and articulatory similarity associated with it.

4.1 Textual Similarity

This is simply the Levenshtein edit distance between two strings. Where the two languages are written with different scripts, this is simply the maximum length of the strings, but in some cases, a language written in the same script as the donor language may borrow a word and keep the spelling unchanged, even if the pronunciation changes. A case in point is the word "science," a loan derived from French *science*, which is spelled identically but pronounced very differently (/saɪən(t)s/ vs. /sjɑ̃s/). Textual edit distance may be a useful feature for some language pairs, so we keep this metric.

4.2 Phonetic Similarity

Having created IPA transcriptions of the words, we compute 6 distance metrics over the transcriptions, all available from the PanPhon package:

- **Fast Levenshtein Distance**. A C implementation of Levenshtein distance (Levenshtein et al., 1966). PanPhon sets all edit costs to 1.
- **Dolgo Prime Distance**. Based the notion of the Dolgopolsky list of the 15 most stable lexemes (Dolgopolsky, 1986) but extended by PanPhon to a list of 14 most stable phonemes. Phonemes are mapped to these classes, over which Levenshtein distance is calculated.
- **Feature Edit Distance**. IPA is converted to articulatory feature vectors (e.g., storing presence, absence, or irrelevance of articulatory features place/manner of articulation, roundedness, pulmonic quality, etc.). Levenshtein distance is calculated over the feature vectors.

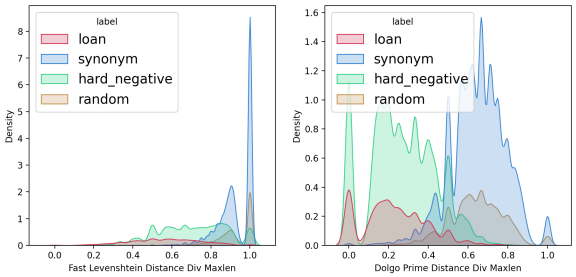

Figure 1: KDE plots of Fast Levenshtein and Dolgo Prime distances.

- **Hamming Feature Distance**. Same as Levenshtein distance, but with substitution cost being the Hamming distance (Hamming, 1950) between the feature vectors, normalized by the length of the vector.
- **Weighted Feature Distance**. Accounts for the class of the IPA symbol when calculating the Levenshtein costs as well as the probability of that specific edit. Weights are prespecified by PanPhon.
- **Partial Hamming Feature Distance**. Insertion and deletion costs are 1, however the cost of substitution for a zero value is half the substitution cost for a nonzero value.

We use the PanPhon normalized version of all edit distances, which divides by the maximum length of the two words in the pair. Fig. 1 shows kernel density estimation plots of the distribution of Fast Levenshtein and Dolgo Prime distances over the entire dataset. Loans have the lowest distance on average, followed by hard negatives.

4.3 Semantic Similarity

A loanword between a pair of languages must both sound and mean the same. While phonetic similarity, calculated with edit distance, has been a foundation for past work in loanword detection, modern large language models provide an opportunity to select for semantic similarity between word vectors, provided the models are trained over multilingual data. We make use of the simultaneous multilingual training objectives of MBERT (Devlin et al., 2019) and XLM (Conneau et al., 2020) to benefit from cross-language proximity of contextualized word embeddings, as shown in (Cao et al., 2019). We use the cosine function as our vector similarity measure.

MBERT is the multilingual version of BERT, pretrained on 104 languages, with demonstrated capacity for knowledge transfer on downstream tasks. It differs from BERT in two ways: i) in its

masked language modeling pretraining, each batch comprises sentences from all languages, and ii) its dictionary is shared among all languages and is created by WordPiece from concatenating all corpora. Pires et al. (2019) show that MBERT's ability to transfer is due to a multilingual representation, which enables it to manage transfer across different scripts. These representations seem to share a common subspace that contains linguistic information, independent of specific languages.

XLM-100 is a cross-lingual (100-language) pretrained model which extends previous BERT-based models with a Translation Language Modeling (TLM) objective as well as the masked language and causal language modeling objectives, and has demonstrated success in unsupervised machine translation tasks (Conneau and Lample, 2019). XLM uses byte-pair encoding subword tokenization (Sennrich et al., 2016) which includes the most frequent symbol pairs when creating the token vocabulary. This makes it suitable for encoding tokens common in low-resourced languages (LRL) while alleviating bias towards high-resource languages, by reducing tokenization of LRL words at the character level. This improves the alignment of embedding spaces of languages that share either the same alphabet or proper nouns (Smith et al., 2017), both of which occur frequently among loanwords.

To these models, we input a "sentence" consisting of the word preceded by the `[CLS]` or `<bos>` token and followed by the `[SEP]`/`<eos>` token. We retrieve the vector of the `[CLS]`/`<bos>` token as a representation of the entire semantics of the input, to account for tokenization possibly splitting the word.

4.4 Alignment Network

To account for different phonotactics in paired languages (e.g., Swedish /skuːlɑː/→ Finnish /koulu/), we build a model to align phonemes in a word pair and account for epenthesis, elision, and metathesis, which provides a more informative measure than simply edit distance. Mortensen et al. (2016) show that information-rich phonological representations do better than character-based models or one-hot encodings in tasks such as NER.

We convert the IPA transcriptions to 21 subsegmental articulatory features using PanPhon[7]. These features were padded to the maximum length of a vector in the borrower-donor pair. The features for the loanword and original word were then concatenated for input to the alignment network.

The alignment network is a deep feedforward neural network trained on the aforementioned concatenated features of the *alldata* split of our datasets. The network was trained against the loan/non-loan binary label. This is not to predict loan status, but because we do not include any semantic information at this step, the label acts as an indicator of "phonetically aligned" or not. A positive prediction means the model predicts that the two words in the pair are strongly phonetically aligned according to the articulatory features. During inference, we get the pre-sigmoid logit value as a holistic alignment score between the two words.

5 Evaluation

For evaluation, we create three data distributions for each language pair. One (the **balanced** distribution), contains half loanwords and half non-loans. This is a well-behaved distribution well-suited for machine learning. The non-loans are drawn roughly $\frac{1}{7}$ from the hard negatives, $\frac{4}{7}$ from the synonyms, and $\frac{2}{7}$ from the randoms, reflecting the notion that relatively few words in a language are likely to be very phonetically close to a loanword on average, while there are likely to be many more words of synonymous or similar meaning.

Another distribution attempts to approximate the actual proportion of loanwords from the donor language into the recipient language (the "realistic" distribution, or **realdist**). Sometimes this proportion is well-documented, and at other times not.[8]. Where a figure is provided in the linguistic literature, we use it. Otherwise, we take the number of loanwords we collected from Wiktionary and divide it by the total number of lemmas in the borrowing language, and impose a lower bound of 10%, to maintain enough loanwords in the testing set. The non-loans portion of the *realdist* set is drawn in the same proportions as in the *balanced* set. For all language pairs currently in the WikLoW dataset, the *realdist* contains $<$50% loanwords, but for other language pairs, e.g., Korean-Chinese, $>$50% loanwords is certainly possible or likely (Sohn, 2005).

The final distribution (abbreviated **alldata**), takes all the data we collected from Wiktionary,

[7]Panphon does not contain suprasegmental or tonal information which may explain why alignment logits involving tonal languages such as Chinese may not sufficiently encode articulatory alignment (see Sec. 6)

[8]Sometimes documentation conflicts, such as Macrea (1961) and Sala (1988), which provide differing figures for Romanian loanwords from French, depending on whether all words or only core vocabulary is considered.

to purposely overweight the dataset against loanwords, to test our method in a difficult condition.

To each distribution, we concatenate two one-hot vectors representing the scripts of the languages in the pair. This allows certain models to learn dependencies between the scripts and other variables, e.g., if the languages are written in different scripts, the textual Levenshtein distance becomes nearly meaningless.

Each distribution was divided into a 90:10 train/test split, and then shuffled. We evaluate four different binary classifiers on all distributions: a logistic regressor (LR), a linear SVM, a Random Forest (RF), and a deep neural network (NN). The neural network consists of 3 layers of 512, 256, and 128 hidden units respectively, all with ReLU activation and followed by 10% dropout, and a final sigmoid activation, and is trained for 5,000 epochs with Adam optimization and BCE loss. We perform the evaluations listed below.

Single Multilingual Model (SMM) For each different data distribution, we train a single model on the data from every language pair listed in Table 1 except for *Persian-Arabic*, *Hungarian-German*, *German-Italian*, and *Catalan-Arabic*, which we reserve for subsequent experiments. The single multilingual model is evaluated on the unseen test sets for all language pairs used in training.

Pair-Specific Models For each distribution, we train and evaluate on a single language pair only, so we can compare the performance of the SMM to models specialized for each language pair.

Pruned Training Set We train on the *realdist* train set and evaluate on the *alldata* test set. This allows us to test on a much larger test set that contains a lower proportion of loanwords, and test the ability of our model to pick out loanwords from a more challenging distribution with less training data. The *realdist* train set is pruned of word pairs that appear in the *alldata* test set, since the two distributions were originally created separately. This experiment used the neural network classifier only.

Unseen Language Pairs We evaluate the performance of the SMM on *Persian-Arabic*, *Hungarian-German*, *German-Italian*, and *Catalan-Arabic*, which the model has *never* seen. This experiment used the neural network classifier only.

6 Results

Our primary metrics are precision, recall, and F1-score on positive loanword identification. Table 2 shows the average positive F1 score on the *realdist* distribution of the 4 classifiers we evaluated. The remaining tables and figures all focus on the results of the neural network, are sorted by decreasing number of loanwords in the language pair, and are discussed in Sec. 7. Table 3 presents the SMM results. Fig. 2 shows the *alldata* test results from Table 3 in bar graph form compared to the performance of the loanword detection model on each language pair when trained *only* on data from that language pair, and to the model when trained on the smaller pruned *realdist* training data. Table 4 shows the SMM's performance on the unseen language pairs, and Fig. 3 plots F1 score against the number of loanwords in each pair's test set.

	LR	NN	SVM	RF
F1 (+)	85	86	84	85

Table 2: Average F1 (+) of 4 classifiers (as %)

7 Discussion

We can quantitatively compare our approach to that of Mi et al. (2021), who report 75.35% average precision, 74.09% average recall, and 74.71% average F1 on loanword detection in Uyghur on borrowings from Russian, Arabic, Turkish, and Chinese. Our results are on different language pairs but are comparable to or exceed this, particularly if the testing set is balanced between loans and non-loans.

In Fig. 2, we can see that in most cases, the multilingual model outperforms the single-pair models on the same language pair on loanword retrieval, though this effect is most pronounced in language pairs with a higher density of loanwords. The model trained on the smaller pruned *realdist* data sees an appreciable drop in precision, but an equal or greater *increase* in loanword recall, and this effect is especially pronounced in pairs with fewer loanwords in the data overall, suggesting that training on a more realistic distribution may be advantageous when prioritizing reducing false negatives.

Fig. 3 shows the correlation between test set size and performance of the SMM (including unseen language pairs). There appears to be a strong correlation between the proportion of loanwords in a test set (as expected, a balanced set leads to optimal performance), but also the raw size of the test set itself. The model performs better on larger test sets, unseen or not, regardless of what data it was trained on. We speculate that this may be because when a borrowing language borrows a lot of words from a donor language, it does so at around the same time (e.g., English from Norman French), meaning

	all	*en-fr*	*en-de*	*id-nl*	*pl-fr*	*ro-fr*	*kk-ru*	*ro-hu*	*de-fr*	*hi-fa*	*fi-sv*	*az-ar*	*zh-en*
P	92	96	90	96	90	94	93	88	94	94	85	85	81
(+)	98	97	98	99	97	96	98	99	98	97	98	98	98
	83	89	84	85	82	82	86	76	86	81	78	69	71
R	81	91	87	90	73	82	88	61	75	86	68	71	51
(+)	98	99	99	99	97	99	100	93	99	99	98	98	93
	75	88	84	85	66	73	81	49	63	72	56	62	47
F1	86	93	89	93	81	88	91	72	83	90	75	70	62
(+)	98	98	98	99	97	98	99	96	98	98	98	98	95
	79	89	84	85	73	77	83	60	73	76	65	65	57

Table 3: Single multilingual NN model results as % (1st row: *realdist*, 2nd row: *balanced*, 3rd row: *alldata*).

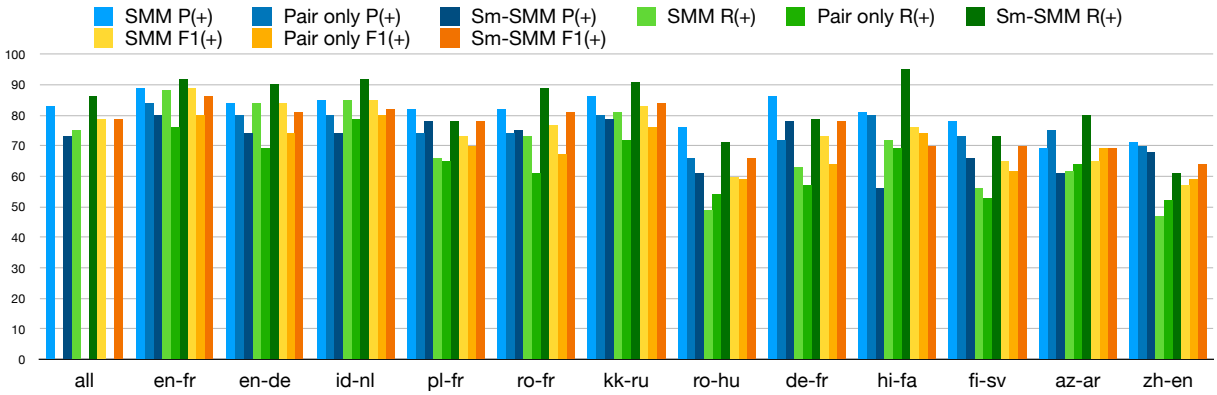

Figure 2: NN *alldata* results comparing base SMM, pair-specific model, and SMM trained on pruned *realdist* data (Small-SMM).

	fa-ar	*hu-de*	*de-it*	*ca-ar*
P	95	95	73	100
(+)	97	100	100	75
	75	73	54	25
R	75	36	33	20
(+)	97	93	92	30
	64	30	29	10
F1	84	52	46	33
(+)	97	96	96	43
	69	43	38	14

Table 4: Holdout performance (same format as Table 3).

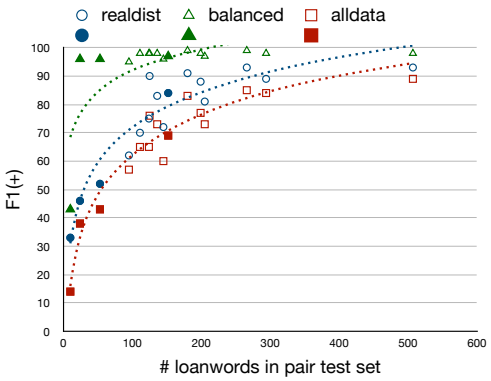

Figure 3: F1 score vs. number of loans per pair. Solid markers indicate unseen language pairs.

there are consistent transformations applied, which a network can pick up. This may not be the case in language pairs with a sparser density. Catalan-Arabic performance is particularly low and there are only 10 words in the test set, many of which were likely mediated by Spanish first.

7.1 Error Analysis

Mistakes made by the SMM, particularly on language pairs that perform less well, are illuminating.

Finnish-Swedish false negatives, e.g., *kyökki*/*kök* and *rontti*/*strunt*, suggest that additional final vowels and reduction of consonant clusters pose a difficulty. Other false negatives back this up, such as Mandarin-English 巧克力 (*qiǎokèlì*)/*chocolate* or Romanian-Hungarian *sudui*/*szidni*, which show sometimes irregular transformation to fit the borrowing language's phonotactics.

False positives are overwhelmingly hard negatives, and the model has particular trouble with languages that use abugidas or alphabets that borrow from languages that use abjads, due to the lack of vowels. Examples include Hindi-Persian निसार

(*nisār*)/*nasr* and Azerbaijani-Arabic *rəbb*/*rabbaba*. This can largely be attributed to Epitran not inserting vowels into Perso-Arabic transcriptions.

This suggests one clear way to potentially improve our method: incorporating multi-head attention into the phonemic alignment network rather than the current feedforward structure, which is performing the task the way single-head attention would and then averaging over all alignments.

Cognates are excluded from the positive loans data unless the cognate was actually later borrowed into the recipient language, as sometimes happens (e.g. "chef" vs. "head"). It is rare for cognates to be misclassified as loanwords due to intervening sound changes between two languages with common ancestry, but there are cases where a loanword is paired with a word in the source language that is cognate to it but is not the original borrowed word. Table 5 shows some of these rare cases.

Language pair	Word pair
en-fr	*communard/communal*
en-de	*Blume/Bluhm*
ro-fr	*cupolă/coupelle*
de-fr	*Montage/montant*

Table 5: Cognates mislabeled as loanwords by SMM.

7.2 Influence of Features

Neural networks are difficult to interpret, but the weights of the logistic regression classifier, which on average performed ∼1-3% lower than the neural network, gives a sense of which features are important. Overall the alignment score is a strong positive correlate to loanword status across all language pairs. As expected, Levenshtein textual edit distance is inversely correlated with loanword status in pairs that share the same script, but not when the languages use different scripts. Interestingly, the semantic similarity metrics do not have a lot of influence on the model, but XLM is generally more influential than MBERT, and this influence is more pronounced among the lower-resourced languages (e.g., Kazakh-Russian, Hindi-Persian, Azerbaijani-Arabic), which supports XLM's claim to be more suited to LRLs, but the influence is most pronounced on English-French, the highest-resourced language pair currently in WikLoW, which undercuts the claim somewhat. Since loanwords are expected to be semantically similar, this task allows us to investigate the quality of multilingual language models on different language pairs. These findings are also borne out by ablation tests on the neural network classifier. For instance, dropping the alignment score and semantic similarities causes recall on the different-script pairs (Hindi-Persian, Azerbaijani-Arabic, Mandarin-English) to drop by 20% or more, while not affecting the same-script pairs as significantly. Sec. A.5 in the appendix shows these findings in more detail.

7.3 Human Comparison

To compare the performance of our model to human performance on loanword retrieval, we selected three language pairs, English-French, Hindi-Persian, and Mandarin-English, took the list of loanwords from the test set of the *alldata* distribution, and asked N annotators who were fluent speakers of each borrowing language to mark which in the list they thought were loans from the listed donor language. This was a fast way to assess human loanword recall and provide comparative numbers to our system on these language pairs. Table 6 shows the results.

Pair	N	Human μ R(+)	SMM R(+)	κ	# loans (homonyms)
en-fr	7	29	**88**	.059	508 (8)
hi-fa	6	60	**72**	.113	125 (4)
zh-en	6	**85**[9]	47	.034	95 (1)

Table 6: Human average loanword recall vs. SMM recall (as %).

Our system is able to significantly exceed human recall on English-French and Hindi-Persian, but not on Chinese-English (as noted those numbers may be inflated). Some loans were also homonyms, which may have had a small impact on human recall (see supplement). We also calculated Fleiss' kappa (Fleiss, 1971) over the human annotations and found that even when individual humans demonstrated moderate-to-high recall on loanword retrieval, there was virtually no agreement among annotators on which loanwords they identified.

8 Conclusions and Future Work

Automated loanword detection enables a number of downstream tasks. Coreferents and named entities across languages may often be loanwords, and common vocabulary enables potential improvements in machine translation (Ortega et al., 2021).

Parallel corpora of loanwords also afford learning cross-lingual contextual word embedding mappings—inspired by the success of pre-

[9]These numbers may be artificially high due to the Chinese annotators being bilingual in English.

Transformer embedding mappings (Bojanowski et al., 2016), and the potential of post-Transformer alignments (Cao et al., 2019). These can be incorporated into the Transformer architecture to provide auxiliary signals to enhance translation in two ways: i) Introducing another multi-head attention between the input language embeddings and their mappings in the target language space—similar to the second multi-head attention block in the original Transformer architecture (Vaswani et al., 2017). We propose to map embeddings between a source language L_X and target language L_Y by computing a transformation matrix between paired representations of semantically-equivalent words or sentences, then to compute attention weights between these mapped embeddings, and concatenate these auxiliary attention outputs with the attention between tokens from L_X and already-generated tokens from L_Y. ii) Unmasking identified loanwords in the target language in the decoder's input, which is expected to provide further context to the decoder in the target language. This would replicate a uniquely human linguistic capability: the ability to pick up context in an unfamiliar language by picking out known words (i.e., loans from a known language). Fig. 4 shows a proposed architecture for these operations.

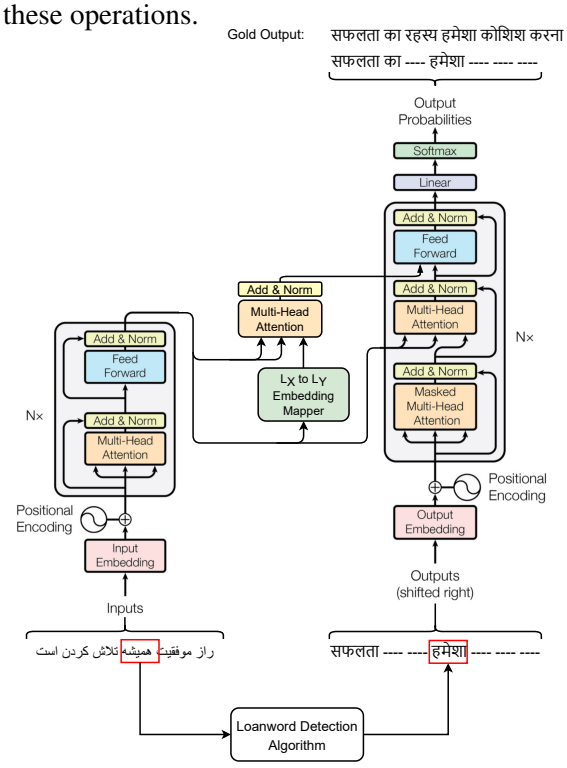

Figure 4: Proposed novel Transformer architecture for exploiting loanword knowledge in machine translation.

Mapping between embedding spaces also allows expanding our method and dataset to new languages not covered by MBERT or XLM through resources like IndicBERT (Kakwani et al., 2020).

8.1 Why Study Loanwords?

In keeping with the COLING 2022 special theme, "Tackling the Grand Challenges of the world by promoting mutual understanding through language," we posit that common vocabulary decreases barriers to communication, and representing it offers a particular benefit to LRLs in NLP, by providing a way to leverage resources from higher-resourced languages that have contributed vocabulary to an LRL. In this, Wiktionary itself has been and can continue to be a resource (Zesch et al., 2008; Krizhanovsky and Smirnov, 2013; De Melo, 2015; Wu and Yarowsky, 2020). Loanword detection is also necessarily not language agnostic, and is therefore important for linguistic diversity and inclusion in NLP (Joshi et al., 2020), although our multilingual results suggest that there may be key features of loanwords that allow detection to generalize.

We propose these challenges to the community:

1. We have presented a novel baseline for loanword detection across arbitrary language pairs that delivers high-quality results, but there remain challenges particularly for languages with divergent phonotactics.
2. We have also presented a method to gather more data for new languages, and demonstrated our detection method's performance on unseen language pairs, which we present as a baseline for comparison.
3. We have also provided homonym data, which is tailor-made to confound a loanword detection algorithm. Discriminating loanwords from their homonyms remains a challenge that presents many interesting opportunities in areas like machine translation and comparative and corpus linguistics.

Acknowledgments

Thanks to our human annotators: Iliana Castillon, Srishti Gupta, Ye (Christina) Jiang, Chirag Kandoi, Alex Karduna, Uma Krishnaswami, Vasantha Krishnaswami, Jacinda Li, Rachel Masters, Wenjun Ni, Evelyn Ochoa, Dhruva Patil, Aditya Raikwar, Ethan Seefried, Caspian Siebert, Videep Venkatesha, Xiaolu Wei, Tianyi Xiong, and Haodong Yang. Thanks to our reviewers for their helpful comments. We're sorry we couldn't get Arabic script to compile in LaTeX but it kept rendering backwards.

References

Elena Álvarez-Mellado and Constantine Lignos. 2022. Detecting Unassimilated Borrowings in Spanish: An Annotated Corpus and Approaches to Modeling. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 3868–3888.

Werner Betz. 1959. Lehnwörter und Lehnprägungen im vor-und Frühdeutschen. In *Band 1*, pages 135–164. de Gruyter.

Michael Bloodgood and Benjamin Strauss. 2017. Using Global Constraints and Reranking to Improve Cognates Detection. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1983–1992, Vancouver, Canada. Association for Computational Linguistics.

Piotr Bojanowski, Edouard Grave, Armand Joulin, and Tomas Mikolov. 2016. Enriching Word Vectors with Subword Information. *arXiv preprint arXiv:1607.04606*.

Cecil H Brown, Eric W Holman, Søren Wichmann, and Viveka Velupillai. 2008. Automated classification of the world s languages: a description of the method and preliminary results. *Language Typology and Universals*, 61(4):285–308.

Steven Cao, Nikita Kitaev, and Dan Klein. 2019. Multilingual Alignment of Contextual Word Representations. In *Proceedings of the International Conference on Learning Representations*.

Alexis Conneau, Kartikay Khandelwal, Naman Goyal, Vishrav Chaudhary, Guillaume Wenzek, Francisco Guzmán, Édouard Grave, Myle Ott, Luke Zettlemoyer, and Veselin Stoyanov. 2020. Unsupervised Cross-lingual Representation Learning at Scale. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 8440–8451.

Alexis Conneau and Guillaume Lample. 2019. *Cross-Lingual Language Model Pretraining*. Curran Associates Inc., Red Hook, NY, USA.

Alina Maria Cristea, Liviu P Dinu, Simona Georgescu, Mihnea-Lucian Mihai, and Ana Sabina Uban. 2021. Automatic Discrimination between Inherited and Borrowed Latin Words in Romance Languages. In *Findings of the Association for Computational Linguistics: EMNLP 2021*, pages 2845–2855.

Gerard De Melo. 2014. Etymological Wordnet: Tracing The History of Words. In *LREC*, pages 1148–1154. Citeseer.

Gerard De Melo. 2015. Wiktionary-based word embeddings. In *Proceedings of Machine Translation Summit XV: Papers*.

Marisa Delz. 2013. A theoretical approach to automatic loanword detection. *Master's thesis, Eberhard-Karls-Universität Tübingen*.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, Minneapolis, Minnesota. Association for Computational Linguistics.

Aaron B Dolgopolsky. 1986. A probabilistic hypothesis concerning the oldest relationships among the language families of northern Eurasia. *Typology, relationship and time: a collection of papers on language change and relationship by soviet linguists*, pages 27–50.

Joseph L Fleiss. 1971. Measuring nominal scale agreement among many raters. *Psychological bulletin*, 76(5):378.

Richard W Hamming. 1950. Error detecting and error correcting codes. *The Bell system technical journal*, 29(2):147–160.

Martin Haspelmath and Uri Tadmor. 2009. The loanword typology project and the world loanword database. *Loanwords in the world's languages: A comparative handbook*, pages 1–34.

Einar Haugen. 1950. The analysis of linguistic borrowing. *Language*, 26(2):210–231.

Gerhard Jäger. 2018. Global-scale phylogenetic linguistic inference from lexical resources. *Scientific Data*, 5(1):1–16.

Ganesh Jawahar, Benoît Sagot, and Djamé Seddah. 2019. What Does BERT Learn about the Structure of Language? In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 3651–3657, Florence, Italy. Association for Computational Linguistics.

Pratik Joshi, Sebastin Santy, Amar Budhiraja, Kalika Bali, and Monojit Choudhury. 2020. The State and Fate of Linguistic Diversity and Inclusion in the NLP World. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 6282–6293, Online. Association for Computational Linguistics.

Divyanshu Kakwani, Anoop Kunchukuttan, Satish Golla, NC Gokul, Avik Bhattacharyya, Mitesh M Khapra, and Pratyush Kumar. 2020. IndicNLPSuite: Monolingual corpora, evaluation benchmarks and pre-trained multilingual language models for Indian languages. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 4948–4961.

Marisa Köllner. 2021. *Automatic Loanword Identification Using Tree Reconciliation*. Ph.D. thesis, Universität Tübingen.

Marisa Köllner and Johannes Dellert. 2016. Ancestral state reconstruction and loanword detection.

Grzegorz Kondrak. 2001. Identifying Cognates by Phonetic and Semantic Similarity. In *Second Meeting of the North American Chapter of the Association for Computational Linguistics*.

Andrew A Krizhanovsky and Alexander V Smirnov. 2013. An approach to automated construction of a general-purpose lexical ontology based on Wiktionary. *Journal of Computer and Systems Sciences International*, 52(2):215–225.

Sofie Labat and Els Lefever. 2019. A Classification-Based Approach to Cognate Detection Combining Orthographic and Semantic Similarity Information. In *Proceedings of the International Conference on Recent Advances in Natural Language Processing (RANLP 2019)*, pages 602–610, Varna, Bulgaria. INCOMA Ltd.

Els Lefever, Sofie Labat, and Pranaydeep Singh. 2020. Identifying Cognates in English-Dutch and French-Dutch by means of Orthographic Information and Cross-lingual Word Embeddings. In *Proceedings of the 12th Language Resources and Evaluation Conference*, pages 4096–4101, Marseille, France. European Language Resources Association.

Vladimir I Levenshtein et al. 1966. Binary codes capable of correcting deletions, insertions, and reversals. In *Soviet physics doklady*, volume 10, pages 707–710. Soviet Union.

Dimitrie Macrea. 1961. Originea și structura limbii române. *Probleme de lingvistică română*, pages 7–45.

Chenggang Mi, Lei Xie, and Yanning Zhang. 2020. Loanword identification in low-resource languages with minimal supervision. *ACM Transactions on Asian and Low-Resource Language Information Processing (TALLIP)*, 19(3):1–22.

Chenggang Mi, Yating Yang, Lei Wang, Xiao Li, and Kamali Dalielihan. 2014. Detection of loan words in Uyghur texts. In *CCF International Conference on Natural Language Processing and Chinese Computing*, pages 103–112. Springer.

Chenggang Mi, Yating Yang, Lei Wang, Xi Zhou, and Tonghai Jiang. 2018. A neural network based model for loanword identification in Uyghur. In *Proceedings of the Eleventh International Conference on Language Resources and Evaluation (LREC 2018)*.

Chenggang Mi, Shaolin Zhu, and Rui Nie. 2021. Improving Loanword Identification in Low-Resource Language with Data Augmentation and Multiple Feature Fusion. *Computational Intelligence and Neuroscience*, 2021.

David R Mortensen, Siddharth Dalmia, and Patrick Littell. 2018. Epitran: Precision G2P for many languages. In *Proceedings of the Eleventh International Conference on Language Resources and Evaluation (LREC 2018)*.

David R Mortensen, Patrick Littell, Akash Bharadwaj, Kartik Goyal, Chris Dyer, and Lori Levin. 2016. Panphon: A resource for mapping IPA segments to articulatory feature vectors. In *Proceedings of COLING 2016, the 26th International Conference on Computational Linguistics: Technical Papers*, pages 3475–3484.

Andrea Mulloni and Viktor Pekar. 2006. Automatic Detection of Orthographics Cues for Cognate Recognition. In *Proceedings of the Fifth International Conference on Language Resources and Evaluation (LREC'06)*, Genoa, Italy. European Language Resources Association (ELRA).

John E Ortega, Richard Alexander Castro-Mamani, and Jaime Rafael Montoya Samame. 2021. Love Thy Neighbor: Combining Two Neighboring Low-Resource Languages for Translation. In *Proceedings of the 4th Workshop on Technologies for MT of Low Resource Languages (LoResMT2021)*, pages 44–51.

Telmo Pires, Eva Schlinger, and Dan Garrette. 2019. How Multilingual is Multilingual BERT? In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 4996–5001.

Taraka Rama and Johann-Mattis List. 2019. An automated framework for fast cognate detection and bayesian phylogenetic inference in computational historical linguistics. Association for Computational Linguistics.

Benoît Sagot. 2017. Extracting an etymological database from Wiktionary. In *Electronic Lexicography in the 21st century (eLex 2017)*, pages 716–728.

Marius Sala. 1988. *Vocabularul reprezentativ al limbilor romanice*. Ed. Ştiinţifiča si Enciclopedică.

Charles Schafer and David Yarowsky. 2002. Inducing Translation Lexicons via Diverse Similarity Measures and Bridge Languages. In *COLING-02: The 6th Conference on Natural Language Learning 2002 (CoNLL-2002)*.

Rico Sennrich, Barry Haddow, and Alexandra Birch. 2016. Neural Machine Translation of Rare Words with Subword Units. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1715–1725, Berlin, Germany. Association for Computational Linguistics.

Samuel L. Smith, David H. P. Turban, Steven Hamblin, and Nils Y. Hammerla. 2017. Offline bilingual word vectors, orthogonal transformations and the inverted softmax. *arXiv e-prints*, page arXiv:1702.03859.

Ho-min Sohn. 2005. *Korean language in culture and society*. University of Hawaii press.

Hiroya Takamura, Ryo Nagata, and Yoshifumi Kawasaki. 2017. Analyzing semantic change in Japanese loanwords. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 1, Long Papers*, pages 1195–1204.

Ian Tenney, Dipanjan Das, and Ellie Pavlick. 2019. BERT Rediscovers the Classical NLP Pipeline. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 4593–4601, Florence, Italy. Association for Computational Linguistics.

Yulia Tsvetkov and Chris Dyer. 2015. Lexicon stratification for translating out-of-vocabulary words. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 2: Short Papers)*, pages 125–131.

René Van Der Ark, Philippe Mennecier, John Nerbonne, and Franz Manni. 2007. Preliminary identification of language groups and loan words in central asia. In *Proceedings of the RANLP Workshop on Acquisition and Management of Multilingual Lexicons*, pages 13–20.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. *Advances in neural information processing systems*, 30.

Uriel Weinreich. 1954. Languages in contact. In *Languages in Contact*. De Gruyter Mouton.

Huiyu Wu and Diego Klabjan. 2021. Logit-based Uncertainty Measure in Classification. In *2021 IEEE International Conference on Big Data (Big Data)*, pages 948–956. IEEE.

Winston Wu and David Yarowsky. 2020. Wiktionary normalization of translations and morphological information. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 4683–4692.

Torsten Zesch, Christof Müller, and Iryna Gurevych. 2008. Using Wiktionary for Computing Semantic Relatedness. In *AAAI*, volume 8, pages 861–866.

Liqin Zhang, Franz Manni, Ray Fabri, and John Nerbonne. 2021. Detecting loan words computationally. *Variation Rolls the Dice. A Worldwide Collage in Honour of Salikoko S. Mufwene*.

Appendix

A.1 Further Details on Data Collection

We use the MediaWiki API to conduct our data collection. To maintain adherence to Wiktionary's terms of service, we make no more than 200 requests per second and sleep after a specified number of words are processed (by default, 200).

When conducting the initial data collection, we exclude terms that begin or end with hyphens, as those are likely to be affixes; that are only one letter long, as those are likely to contribute too much noise to the final dataset; and those that contain numerals or non-phonetic, non-syllabic, or non-logographic (depending on the language) symbols.

The choice of language pairs investigated here was determined in part by the intersection of languages that are supported by all 3 of Epitran, MBERT, and XLM-100, and that have a `[Recipient]_terms_borrowed_from_[Donor]` category on Wiktionary that contains more than 1,000 entries. The exceptions to this are: Finnish-Swedish, where Finnish is not natively supported by Epitran, but we built our own Finnish G2P mapping for Epitran; Mandarin-English, where some terms were discarded during preprocessing, causing the number to fall below 1,000; and Hungarian-German, German-Italian, and Catalan-Arabic, which were selected specifically for having fewer than 1,000 loanwords listed in Wiktionary.

Table 7 shows the 2-letter ISO 639-1 codes for these languages, which can help in interpreting Table 3 (Sec. 6).

Code	Language
ar	Arabic
az	Azerbaijani
ca	Catalan
de	German
en	English
fa	Persian
fi	Finnish
fr	French
hi	Hindi
hu	Hungarian
id	Indonesian
it	Italian
kk	Kazakh
nl	Dutch
pl	Polish
ro	Romanian
ru	Russian
sv	Swedish
zh	Mandarin

Table 7: ISO 639-1 language codes for languages in current dataset.

A.2 Further Details on Semantic Similarity

In our experiments, for the XLM-100 and MBERT models, we extract the <bos> embeddings (equivalent to the `[CLS]` token for MBERT) for a word pair from the `last_hidden_state`. Numerous studies like (Jawahar et al., 2019) and (Tenney et al., 2019) suggest that BERT's later layers encode comparatively more high-level semantic information than its middle layers which tend to capture more syntactic features in the linguistic hierarchy. For both the models, the dimensions of the generated embeddings are of the shape (`batch_size`, `sequence_length`, `hidden_size`) where `batch_size` is 8 for both, `sequence_length` is the number of tokens from the word after tokenization (`max_length` is 512 for both models) whereas the embedding dimension i.e., `hidden_size` is 1280 for the XLM and 768 for MBERT. We then get the cosine similarities between the generated embeddings of each word pair of the borrower-donor pair in order to extract their semantic similarities.

A.3 Further Details on Alignment Network

The alignment network was trained for 5,000 epochs with Binary Cross-Entropy (BCE) loss and Adam optimization, with a 20 percent validation set to prevent overfitting. The DNN consists of two hidden layers with 512 neurons each with ReLU activation, followed by 10% dropout, and an output layer and a sigmoid function.

Previous studies like Wu and Klabjan (2021) have suggested that logit outputs of neural networks can be a reliable and agnostic uncertainty measure that captures innate features of classes during classification and detection tasks. The alignment network here maps the concatenated articulatory features of a word pair to their class and therefore, the logits will contain class-based information that can subsequently be used as crucial features for our classifiers. In other words, these logits encode alignment information of the articulatory features that can be mapped to whether a pair is a phonetically similar, conditioned upon the

sound patterns of their respective languages, or not.

A.4 Results from Other Classifiers

The main paper presented the results of the neural network classifier in detail and discussion of the weights from the logistic regressor. Here we present results from the logistic regression classifier (Table 8), the support vector machine (Table 9), and the random forest (Table 10).

The neural network is consistently the best-performing classifier, by about 1-5% F1, depending on which distribution is being evaluated on. The other classifiers can be expected to perform about this much lower. One thing to note is that the effect is most pronounced on the *alldata* dataset, which is the hardest dataset for any classifier on average, due to the overwhelming preponderance of non-loans. When the dataset is balanced between loans and non-loans, the type of classifier chosen for loanword detection is almost immaterial, with almost perfect performance all around. It seems at these proportions, the information encoded in the datasets, such as alignment score, edit distances, and cosine similarities, are informative enough. For this reason we have focused most discussion in the main body of the paper on the *alldata* and *realdist* datasets.

However, while the behavior of the logistic regressor and SVM are largely consistent with each other, and track that 1-5% difference with the neural network across all language pairs, the behavior of the random forest is rather different and inconsistent with the other classifiers. For example, it gets 100% recall on the balanced distributions of Indonesian-Dutch and Romanian-French (as well as Kazakh-Russian like the other classifiers), but on the Chinese-English *alldata* distribution, recall comes in ∼20% below the other classifiers. The other pairs with dissimilar scripts see a similar, albeit reduced effect on the same distribution, but so do some pairs that share a script, such as Indonesian-Dutch and Romanian-French.

A.5 Further Details on Influence of Features

This section contains the quantitative breakdown of the influence of different features on the results, which was discussed in Sec. 7.2. Fig. 5 is a graph representation of the logistic regressor weights mentioned there. The circular markers represent language pairs where both languages use the same script (including extended versions), while the square markers represent pairs where the languages use different scripts.

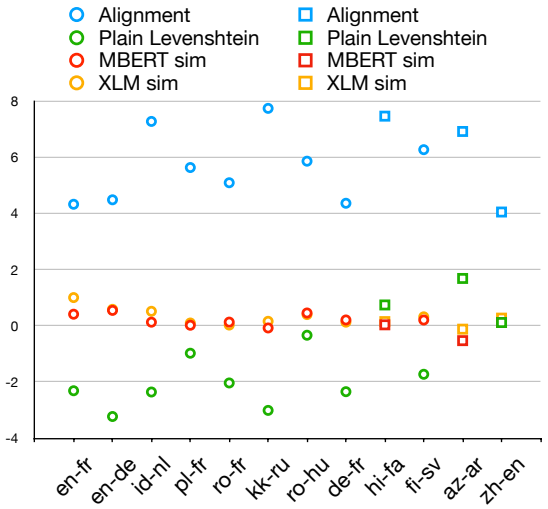

Figure 5: Logistic regressor weights trained on the *alldata* distribution.

Inferences drawn from the logistic regressor weights are bolstered by ablation tests on the neural network. Table 11 shows the neural network performance when the alignment scores and cosine similarities are not used as input features.

Articulatory alignment scores and cosine similarities are most important when the languages in the pair use different scripts. When these are removed as training inputs, and only phonetic and textual distance metrics are left, along with the script encodings, performance on the Azerbaijani-Arabic *alldata* distribution drops by 10% positive F1 and Hindi-Persian drops by 20% positive F1. The most drastic case is Mandarin-English, where without these features, positive F1 on *realdist* and *alldata* drop by 19% and 47% respectively, and positive recall drops by **20%** and **42%** respectively. This is because the different scripts make textual Levenshtein distance a useless feature here, and the differing phonologies of Mandarin and English make the phonetic edit distances noisy (e.g., see Sec. 7.1). Meanwhile, on certain same-script pairs, particularly those where words tend to be imported with little change in spelling (e.g., English-French, English-German, German-French), performance can actually go *up* slightly, because in these cases, textual Levenshtein distance is enough to detect that the word is a loan.

We should note that with only phonetic and script features, performance on the *balanced* distribution remains relatively high but suffers slightly. However, results vary on the *realdist* distribution,

	all	*en-fr*	*en-de*	*id-nl*	*pl-fr*	*ro-fr*	*kk-ru*	*ro-hu*	*de-fr*	*hi-fa*	*fi-sv*	*az-ar*	*zh-en*
P	91	96	89	96	90	94	93	83	93	94	83	82	75
(+)	98	97	97	98	97	96	99	98	98	97	95	97	97
	80	86	81	84	77	78	83	78	84	77	73	62	65
R	80	88	83	89	76	82	86	66	73	83	69	65	52
(+)	98	98	98	99	98	99	100	94	96	99	99	95	94
	72	85	80	83	65	73	76	48	59	68	58	51	46
F1	85	92	86	93	82	88	89	73	82	88	75	72	61
(+)	97	97	97	98	97	98	99	96	97	98	97	96	95
	76	86	80	84	71	76	88	60	80	72	65	56	54

Table 8: Single multilingual logistic regression classifier results as % (1st row: *realdist*, 2nd row: *balanced*, 3rd row: *alldata*).

	all	*en-fr*	*en-de*	*id-nl*	*pl-fr*	*ro-fr*	*kk-ru*	*ro-hu*	*de-fr*	*hi-fa*	*fi-sv*	*az-ar*	*zh-en*
P	92	96	89	97	90	94	93	85	93	93	82	80	75
(+)	97	97	97	99	97	96	99	99	98	97	95	98	98
	80	86	79	83	77	78	82	78	85	80	74	62	66
R	78	87	79	88	74	81	85	63	69	80	63	62	48
(+)	98	98	98	99	98	99	100	94	97	99	99	95	94
	70	83	75	83	62	71	76	48	57	66	54	54	45
F1	84	92	84	92	81	87	89	72	79	86	71	70	59
(+)	98	97	97	99	97	98	99	96	97	98	97	96	96
	75	85	77	83	69	74	79	60	68	73	62	58	54

Table 9: Single multilingual SVM classifier results as % (1st row: *realdist*, 2nd row: *balanced*, 3rd row: *alldata*).

	all	*en-fr*	*en-de*	*id-nl*	*pl-fr*	*ro-fr*	*kk-ru*	*ro-hu*	*de-fr*	*hi-fa*	*fi-sv*	*az-ar*	*zh-en*
P	92	96	88	97	88	93	92	85	90	95	84	88	84
(+)	96	96	95	95	96	93	97	95	97	97	94	99	100
	85	91	87	85	83	82	85	77	81	85	81	73	69
R	80	90	84	89	76	80	91	73	71	77	65	59	40
(+)	99	99	99	100	98	100	100	95	99	99	98	97	92
	68	86	82	77	58	62	78	47	54	59	54	43	26
F1	85	93	86	93	82	86	91	79	80	85	74	71	54
(+)	97	97	97	97	97	96	98	95	98	98	96	98	96
	75	88	84	81	68	71	81	58	65	70	65	54	38

Table 10: Single multilingual random forest classifier results as % (1st row: *realdist*, 2nd row: *balanced*, 3rd row: *alldata*).

and there appears to be some correlation between increased performance on *realdist* without these features, and the proportion of loans in that distribution, suggesting that this is potentially important to consider (i.e., the base rate of loans from French into English, for instance, is relatively high). The performance penalty we see on LRLs and different-script pairs do suggest that overall the alignment score is most critical to generalizable performance, and the semantic similarities provide a way to analyze the quality of large multilingual language models for certain language pairs. These could also be augmented with other pair-specific metrics, such as overall measures of lexical or phonetic distance.

A.6 Homonyms in Human Comparison Task

The loanwords from the *alldata* test sets given to human annotators, that are also homonyms, are

	all	*en-fr*	*en-de*	*id-nl*	*pl-fr*	*ro-fr*	*kk-ru*	*ro-hu*	*de-fr*	*hi-fa*	*fi-sv*	*az-ar*	*zh-en*
P	88	95	84	93	82	92	92	79	87	87	77	78	74
(+)	94	95	96	92	97	90	99	95	97	92	93	91	81
	84	92	87	78	80	81	90	76	92	65	74	81	62
R	87	95	97	92	67	93	95	66	96	94	85	73	31
(+)	96	99	99	96	89	99	98	85	97	96	96	96	86
	72	92	93	73	46	77	93	44	93	48	67	41	5
F1	87	95	90	92	74	93	94	72	91	90	80	75	43
(+)	95	97	97	94	93	94	99	90	97	94	94	94	84
	78	92	90	76	58	79	92	56	85	56	70	55	10

Table 11: Single multilingual NN classifier (without alignment and cosine similarity inputs) results as % (1st row: *realdist*, 2nd row: *balanced*, 3rd row: *alldata*).

listed below:

- English-French:
  - "punt," from French *pointe*, meaning a bet or wager, with many other etymologies, including from Old English for a pontoon boat.
  - "Lemans," French surname from toponym *Le Mans*, and from Middle English *Lemans*, "son of Leman."
  - "bride," from French *bride*, meaning a bridle, and from Old English *brȳd*, "bride, daughter-in-law."
  - "paillard," from a French surname (and name of a restaurant), and variant of "palliard," meaning a beggar.
  - "lisse," from French *lisser*, smooth, and from Old English *lissīan*, "to relieve."
  - "tarse," from French *tarse*, the tarsus or ankle-bones, and from archaic term for a male falcon.
  - "par," from French *par*, meaning "through, by," with many other etymologies, including from Latin *pār*, "equal."
  - "bombard," actually a doublet, with two meanings both meaning "cannon," both ultimately from Middle French, one via modern French *bombarde*, the other via Middle English *bombard* (latter form also referred to a bassoon).
- Hindi-Persian:
  - अगर (*agar*), from Persian, meaning "if," and a descendent of Sanskrit अगरु (*agaru*), a type of wood.
  - देह (*deh*), from Persian, meaning "village," and a descendent of Sanskrit देह (*deha*), body.
  - मर्द (*mard*), from Persian, meaning "man," and a descendent of Sanskrit मर्द (*marda*), "destroying".
  - हम (*ham*), from Persian, meaning "also," and natively Hindi ultimately from Sanskrit अस्मे (*asme*) , meaning "we," "us."
- Mandarin-English:
  - 塞特 (*sàitè*), from English *setter* but also from Hebrew male name *Seth*.

A.7 Proportion of Loanwords in Each Distribution

Table 12 shows the proportion of loanwords in each distribution for each language pair. The *balanced* distribution always contains 50% loans and so is not included.

A.8 Supported Languages and Scripts

Our system can in principle support the languages in Table 13 out of the box. While we have only tested on the language pairs mentioned in the main paper, and not every pairing in Table 13 has a sufficient volume of loanwords listed in Wiktionary, data collected in any of these languages can be converted to IPA with Epitran or extensions, and processed by MBERT and XLM to get cosine similarities between word vectors. Epitran can be extended to other languages by defining custom mapping, preprocessing, and postprocessing rules, as we did here for Finnish.

Proper functionality makes an assumption that the language given is written in the associated script

Pair	% Loans (*realdist*)	% Loans (*alldata*)
en-fr	30	14.841
en-de	10	13.322
id-nl	40	14.996
pl-fr	10	12.252
ro-fr	30	11.999
kk-ru	10	11.807
fa-ar	40	11.289
ro-hu	10	10.788
de-fr	10	10.206
hi-fa	30	10.126
fi-sv	10	9.754
az-ar	15	9.324
zh-en	10	10.496
hu-de	10	6.155
de-it	10	3.344
ca-ar	10	1.291

Table 12: Proportion of loanwords per pair in each distribution

listed. This serves the purpose of not only maintaining support in Epitran but also in collecting clean data from Wiktionary, and in assigning the correct one-hot script encoding during training and evaluation.

A.9 Organization of Code/Data

`README.md` contains instructions to run the full pipeline. `language-pairs.json` is a JSON file containing information about the language pairs to make datasets for, including codes for Epitran and Google Translate and desired *realdist* proportion of loans. `language-pairs-holdout.json` is the same for language pairs to be included in the holdout test set and withheld from training. `language-pairs-pipelinetest.json` contains only Catalan-Arabic, which is a small sample and runs (relatively) quickly, in order to validate the pipeline. These JSON files drive most of the rest of the code.

`supported_languages.txt` contains the list of supported languages (cf. Table 13). `epitran-extensions` contains preprocessing, mapping, and postprocessing rules for new Epitran language. Currently this contains only Finnish, which only uses `pre` and `map`. To run Epitran for the new language, these would need to be moved into the corresponding folder in the Epitran distribution in Python's `site-packages`.

ISO code	Language	Script
sq	Albanian	Latin
ar	Arabic	Latin
az	Azerbaijani	Latin
bn	Bengali	Bengali
my	Burmese	Myanmar
ca	Catalan	Latin
zh	Chinese	Chinese
hr	Croatian	Latin
cs	Czech	Latin
nl	Dutch	Latin
en	English	Latin
fi	Finnish	Latin
fr	French	Latin
de	German	Latin
hi	Hindi	Devanagari
hu	Hungarian	Latin
id	Indonesian	Latin
it	Italian	Latin
jv	Javanese	Latin
kk	Kazakh	Cyrillic
ky	Kyrgyz	Cyrillic
ms	Malay	Latin
ml	Malayalam	Malayalam
mr	Marathi	Devanagari
fa	Persian	Arabic
pl	Polish	Latin
pt	Portuguese	Latin
pa	Punjabi	Gurmukhi
ro	Romanian	Latin
ru	Russian	Cyrillic
es	Spanish	Latin
sw	Swahili	Latin
sv	Swedish	Latin
ta	Tamil	Tamil
te	Telugu	Telugu
tr	Turkish	Latin
uk	Ukranian	Cyrillic
ur	Urdu	Arabic
uz	Uzbek	Latin
vi	Vietnamese	Latin

Table 13: Currently supported languages and scripts.

`wiktionary-scraper-python` contains the scrapers for initial data collection. Results are saved in `results`.

`Datasets` contains the code to make the four dataset splits with edit distances. Results

are sorted by type split: `Loans`, `Synonyms`, `Hard-Negatives`, and `Randoms`. Data files contain word pairs, IPA transcriptions of each word, English translations of each word (for interpretability by a more general audience, and not used in training or evaluation—translations may be inaccurate due to shortcomings in the Google Translate model for the language in question). Note that calculating all the dataset splits for a language pair, particularly the hard negatives, may take a *very* long time, up to days, due to the number of passes through the data. This inefficiency is the main reason why only a subset of the 24K available Romanian-French loans are used in these experiments. Decreasing the time complexity of calculating the hard negatives while maintaining the quality of the output is the topic of ongoing research. `production_train_test` is the directory containing the datasets that will be used for final evaluation. These are sorted by language pair and then by evaluation distribution: `alldata`, `balanced`, and `realdist`. `Datasets` also contains the human annotation spreadsheets in folder `human_annotation`.

`Classifiers` contains the code to both train the alignment network for a language pair and get the logit alignment score for each word pair, and to get the cosine similarities from MBERT and XML. Datasets with logit and similarity values are resaved in `production_train_test`. `Classifiers` also contains the code to perform evaluation under all conditions mentioned in the main body of this paper.

`torch_models` contains a saved instance of the single multilingual model. `Final_results` contains the results from that model and others, which are reported in this paper. `FleissKappa` contains the code to calculate Fleiss' kappa score over the human annotations (found inside `Datasets`).

FeatureBART: Feature Based Sequence-to-Sequence Pre-Training for Low-Resource NMT

Abhisek Chakrabarty[*], **Raj Dabre**[*], **Chenchen Ding**,
Hideki Tanaka, **Masao Utiyama** and **Eiichiro Sumita**
National Institute of Information and Communications Technology
3-5 Hikaridai, Seika-cho, Soraku-gun, Kyoto, 619-0289, Japan
{abhisek.chakra,raj.dabre,chenchen.ding,hideki.tanaka,
mutiyama,eiichiro.sumita}@nict.go.jp

Abstract

In this paper we present FeatureBART, a linguistically motivated sequence-to-sequence monolingual pre-training strategy in which syntactic features such as lemma, part-of-speech and dependency labels are incorporated into the span prediction based pre-training framework (BART). These automatically extracted features are incorporated via approaches such as concatenation and relevance mechanisms, among which the latter is known to be better than the former. When used for low-resource NMT as a downstream task, we show that these feature based models give large improvements in bilingual settings and modest ones in multilingual settings over their counterparts that do not use features.

1 Introduction

Sequence-to-sequence (S2S) pre-training done via denoising objectives on monolingual corpora is known to improve generation quality in low-resource settings (Lewis et al., 2020). This has been extensively explored for neural machine translation, however most works show that the improvements for translation into English are more pronounced than those for translation into a non-English language (Liu et al., 2020; Tang et al., 2020). One reason for this is that pre-training leads to a strong decoder that learns to de-noise masked inputs, whereas the knowledge retained in the encoder is rather limited. Thus far, there has been no explicit effort towards improving the contribution of the encoder during pre-training.

Most pre-training methods rely on the power of large models and large corpora, but ignore the possibility of incorporating linguistic knowledge into the pre-trained model. On the other hand, there are several works which show that incorporating linguistic knowledge in the form of lemma, part-of-speech tags and dependency labels, lead to a significant improvement in translation quality (Sennrich and Haddow, 2016; Hoang et al., 2016; Li et al., 2018; Pan et al., 2020; Chakrabarty et al., 2020) in both low- and high-resource settings. Most recently, in an extremely low-resource setting, Chakrabarty et al. (2020) show the effectiveness of using the aforementioned linguistic features, especially when their influence on the model is controlled via relevance mechanisms that appropriately scale feature embeddings before they augment word embeddings. We hypothesize that incorporating linguistic features, into the denoising based pre-training framework, should improve the quality of pre-training, which should then have a positive impact on the translation quality via fine-tuning. To this end, we propose FeatureBART, a feature based sequence-to-sequence pre-training method.

In FeatureBART, linguistic features are obtained via automatic annotators and then converted into embeddings, which are used to augment the word embeddings of the encoder. Feature embeddings, are incorporated by either naive concatenation with the word embeddings or by first weighing them with a relevance mechanism and then adding them to word embeddings. The model itself is trained using a monolingual corpus via either the text-infilling or the mask prediction approaches, the former used in BART and the latter used in BERT. This FeatureBART model is then fine-tuned for low-resource language pairs, where linguistic features are also used during fine-tuning. Experiments on English to 8 Asian languages from the Asian Language Treebank (ALT) dataset (Riza et al., 2016), using bilingual as well as multilingual fine-tuning, show that our feature based pre-training and fine-tuning leads to significant improvements in translation quality indicating the complementary nature of denoising pre-training and features. Analyses of training curves show that when compared to non-feature based pre-training, feature based pre-

[*] Equal contribution.

Proceedings of the 29th International Conference on Computational Linguistics, pages 5014–5020
October 12–17, 2022.

training leads to significantly lower perplexities during the initial stages of fine-tuning.

2 Methodology

We first give some background knowledge about feature based NMT modeling, followed by an explanation of FeatureBART.

2.1 Background: Use of Source Side Morphological Features into NMT

Sennrich and Haddow (2016) proposed the concatenation of embeddings of features of a token (word or sub-word) to the token embedding. In case a word is split into sub-words, the feature is duplicated for each sub-word. For K features of a source token denoted by $s_i = (s_{i1}, \ldots, s_{iK})$, let V_k, E_k, and d_k denote the vocabulary, embedding matrix and dimension of the kth feature. So, $E_k \in \mathbb{R}^{d_k \times |V_k|}$, s_{i1} is the word or sub-word feature and $s_{i2}, \ldots, s_{iK}$ are the linguistic features. The embedding of s_i, say e_i, is formulated as $e_{ik} = E_k s_{ik}$, and $e_i = \|_{k=1}^{K} e_{ik}$. e_{ik} is the vector embedding of s_{ik} where $\|$ is the concatenation operation.

2.2 Relevance of Features

Chakrabarty et al. (2020) proved the effectiveness of the following two feature weighting strategies to be applied to features prior to concatenation:

Self-Relevance: The relevance of a feature embedding is evaluated w.r.t itself. For $k \in \{1, \ldots, K\}$, the self relevance is calculated as $mask_{ik} = sigmoid(W_k e_{ik})$, and then $e'_{ik} = mask_{ik} \odot e_{ik}$. $W_k \in R^{d_k \times d_k}$ is the learnable weight matrix for the kth feature, and $\odot$ is the element-wise multiplication operation. The vector $mask_{ik}$ signifies the self relevance of e_{ik} and is multiplied element-wise with e_{ik} to produce the modified feature embedding e'_{ik}. Finally, $e'_{i1}, \ldots, e'_{iK}$ are concatenated to make the final embedding e'_i for the source token s_i. Thus, $e'_i = \|_{k=1}^{K} e'_{ik}$.

Word-Relevance: It uses the word/sub-word embedding (e_{i1}) to determine the relevance of the remaining feature embeddings. Formally, for $k \in \{2, 3, \ldots, K\}$, $mask_{ik} = sigmoid(W_k(e_{i1} \| e_{ik}))$, and then $e'_{ik} = mask_{ik} \odot e_{ik}$. $W_k \in R^{d_k \times (d_1 + d_k)}$ is the learnable weight matrix and the final embedding e'_i is obtained by concatenating $e'_{i2}, \ldots, e'_{iK}$ with e_{i1}.

2.3 FeatureBART

FeatureBART, is a feature based encoder-decoder pre-trained model trained using linguistic features which are used to augment the word embeddings of the encoder. First, a large monolingual corpus is annotated via morphological and syntactic annotators to obtain different types of features for each token (word or sub-word). This feature annotated monolingual corpus is used for self-supervised training where during training, some tokens (token masking) or token spans (text infilling) in a sentence are replaced with the "MASK" token just like in Lewis et al. (2020). We use dummy mask features for masked tokens or spans. The sentence containing masked content is fed to the encoder and the model is trained so that the decoder can predict the original sentence. We hypothesize that features help in better pre-training as they provide the model with additional information for denoising.

3 Experiments

We describe the datasets for pre-training and fine-tuning, features used and model training details.

Datasets: For pre-training of our BART and FeatureBART models, we choose the English monolingual News Crawl articles[1] of 2007, 2010, and 2013 from WMT-2017 with varying sizes that contain 3.8, 6.8, and 21.7 million sentences respectively. For fine-tuning, we experiment with the multilingual, multi-parallel Asian language treebank (ALT) (Riza et al., 2016)[2] for English to Asian language translation. Following (Chakrabarty et al., 2020), eight Asian languages - Bengali (bg), Filipino (fi), Hindi (hi), Indonesian (id), Khmer (khm), Malay (ms), Myanmar (my) and Vietnamese (vi) are set as the targets. So, bilingual experiments cover eight language pairs from en-bg to en-vi. For our multilingual experiments, we explore one-to-many multilingual translation setup keeping the source side fixed to English (en) and the target side to the eight Asian languages mentioned above. The source side is fixed to English throughout because as the initial attempt on feature-based pre-training, we can rely on high quality automatic morphological analyzers available for English. Given the potential of this work established by empirical results, other languages with moderate quality morphological annotation can also be tried in the future. We use the official train/dev/test splits containing 18088, 1000, and 1018 sentences, respectively.

[1] https://statmt.org/wmt17/translation-task.html

[2] http://www2.nict.go.jp/astrec-att/member/mutiyama/ALT/

Bilingual Results											
Pre-training	Noise	Config	en-bg	en-fi	en-hi	en-id	en-khm	en-ms	en-my	en-vi	Avg.
None	-	Base	7.5	26.98	23.62	30.88	26.24	35.78	16.48	29.05	24.57
	-	+Self-Rel	8.4†	28.22†	26.13†	32.65†	27.33†	37.22†	18.13†	29.91†	26
NC-07	I	Base	9.77	31.28	25.63	33.86	29.0	38.59	18.83	32.76	27.46
	I	+Self-Rel	10	31.53	26.93†	34.47†	29.02	39.01	19.21	33.5†	27.96
NC-10	I	Base	9.78	32.07	26.21	35.25	27.5	38.79	19.5	32.32	27.68
	I	+Word-Rel	9.7	32.57†	26.44	**36.07**†	29.4†	40.34†	18.98	34.78†	28.53
NC-13	I	Base	**10.09**	32.62	26.3	35.56	29.5	40.09	19.72	33.95	28.48
	I	+Concat	9.75	**33.12**†	**26.96**†	35.32	**30.21**†	**40.64**†	**20.01**	**35.26**†	28.91
Multilingual Results											
Pre-training	Noise	Config	en-bg	en-fi	en-hi	en-id	en-khm	en-ms	en-my	en-vi	Avg.
None	-	Base	11.55	31.04	27.29	34.78	30.27	39.37	20.93	34.58	28.73
	-	+Self-Rel	11.40	31.14	**27.94**†	34.42	30.09	39.84†	20.99	33.85	28.71
NC-07	M	Base	11.78	31.90	27.03	35.77	30.47	40.39	20.65	34.94	29.12
	M	+Self-Rel	11.51	32.17	27.68†	36.11	30.94†	40.80	21.13†	35.28	29.45
NC-10	I	Base	11.84	32.66	26.77	35.71	30.72	40.11	20.65	34.56	29.13
	I	+Self-Rel	11.54	32.57	27.76†	36.25†	**31.16**†	40.48	**21.26**†	35.44†	29.58
NC-13	I	Base	**11.98**	**32.67**	26.69	36.11	30.41	40.20	20.60	35.24	29.24
	I	+Self-Rel	11.65	32.38	27.77†	**36.39**	30.98†	**41.35**†	20.95†	**35.75**†	29.65

Table 1: BLEU scores of the bilingual and multilingual models. For a given pre-training corpus size, we only show the results of the best feature and pre-training configuration due to lack of space. Highest scores are BOLD. † marks scores significantly better ($p < 0.05$) than the corresponding non-feature (base) counterparts. The "Noise" column indicates the pre-training approach, "I" and "M" for text-infilling and token masking. Self-Rel, Word-Rel, and Concat denote self-relevance, word-relevance, and concatenation of feature embedding configurations respectively.

Pre-Processing: The monolingual corpora for pre-training and the source side of fine-tuning corpora are tokenized and true-cased by Moses tokenizer (Papineni et al., 2002) and the target languages are tokenized to separate the delimiters and the punctuation symbols. Following Johnson et al. (2017), each source language sentence in multilingual translation setup is appended with a token like <*tgt-id*> which indicates the target language. Byte-pair-encoding (BPE) (Sennrich et al., 2016) is performed to obtain subword vocabularies. We train a single BPE model of vocabulary size $32K$ on the combined training corpora of all 9 languages and use it during pre-training as well as fine-tuning.

Features Used: Morphological annotation of the English datasets is done using Stanford CoreNLP toolkit (Manning et al., 2014). There are three word-level linguistic features - lemma, part-of-speech (POS), and dependency labels. All subwords of a word take the features of that word. We use subword tags (Sennrich et al., 2016) to denote beginning, middle and ending of a subword unit.

Hyperparameters and Training Details: We use variations of the Transformer-base model (Vaswani et al., 2017) for our experiments available from OpenNMT PyTorch (Klein et al., 2017), which we modify for feature based experiments. Wherever possible, we perform hyperparameter tuning of layers, hidden sizes, number of attention heads, dropouts, number of training epochs etc. (See A for details). All training is done on a single 32 GB V-100 GPU. Pre-training is done for 3 epochs for each monolingual corpora. During fine-tuning, validation is done after every 10000 steps and training stops if validation accuracy does not improve for consecutive 5 evaluations. Test set decoding is done using beam search with a beam size of 5 and length penalty of 1.0. Translation performance is measured by BLEU score calculated using *multi-bleu.perl*.

4 Results

Table 1 contains our results for bilingual and multilingual fine-tuning of FeatureBART along with their non-feature counterparts. Bilingual and multilingual results are divided into 4 groups: no pre-training, pre-training using News Crawl articles of 2007 (NC-07), 2010 (NC-10), and 2013 (NC-13) from WMT-2017. We analyze the results as follows:

Bilingual vs. Multilingual Translation without and with Pre-Training: Comparing corresponding rows between the bilingual and multilingual blocks, shows that multilingual models are significantly better than bilingual ones, observations which are in accordance with Arivazhagan et al. (2019); Johnson et al. (2017); Dabre et al. (2020); Zhang et al. (2020). Without

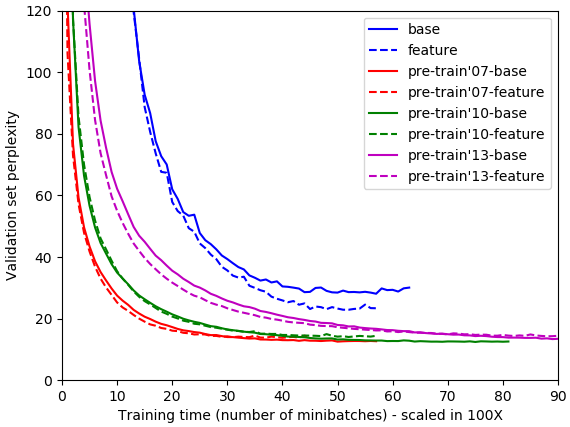

Figure 1: Perplexity plots for English–Bengali models.

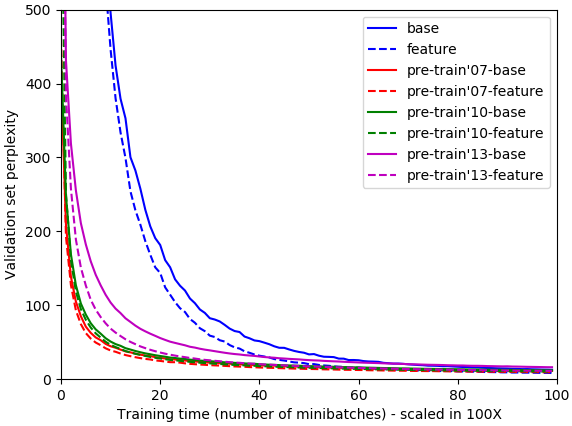

Figure 2: Perplexity plots for multilingual models.

pre-training, improvements in maximum BLEU points comparing bilingual vs. multilingual setups are $(3.15, 2.92, 1.81, 2.13, 2.94, 2.62, 2.86, 4.67)$ for (en $\rightarrow$ bg, fil, hi, id, khm, ms, my, vi) respectively. Pre-training, although it improves performance in both cases, naturally reduces the gap between bilingual and multilingual models. Nevertheless, we get maximum improvements of $(1.89, 0.98, 0.95, 0.71, 1.25, 0.49)$ BLEU points for (en $\rightarrow$ bg, hi, khm, ms, my, vi) when comparing bilingual and multilingual models.

Impact of Features: From Table 1 the following observations can be made: (1) In case of no pre-training, features are very useful in bilingual settings but not in multilingual settings as multilingual systems can utilize a multi-parallel corpora efficiently by acquiring supplementary knowledge from other languages, thus making linguistic information redundant. (2) However, when pre-training is used, multilingual models tend to benefit more from features. As an example, consider en-khm and en-vi multilingual scores in Table 1. Under no pre-training setup, adding features deteriorates the scores, whereas feature based pre-training gives consistent improvements, indicating that feature based modeling and pre-training are complementary.

Optimal Pre-Training and Feature Configurations: From Table 1, looking at the "Noise" column, it is clear that text infilling is the best pre-training objective in most cases. With regard to feature incorporation mechanism, self-relevance predominantly gives the best results. We therefore recommend the use of self-relevance and text-infilling based FeatureBART in low-resource settings.

Studying Model Perplexities: Figure 1 shows the perplexities[3] of en-bg bilingual models. The perplexities with pre-training and features are lower compared to when features are not used, during initial stages of training, but this changes towards convergence. This explains why en-bg does not show performance improvements from features. Figure 2 contains plots of cumulative perplexity of all language pairs for multilingual models. Here, features seem to have larger impact than in bilingual settings, both at the beginning and later stages of training. However, BLEU gains are not always observed, indicating that it may not always be reliable motivating future multi-metric and human evaluation (Marie et al., 2021).

5 Related Work

Pre-Training: Pre-trained models reduce the need for large fine-tuning data for a given downstream task, and in this paper we focus on extremely low-resource settings. In this context, T5 (Raffel et al., 2020), mT5 (Xue et al., 2021), BART (Lewis et al., 2020), mBART-25 (Liu et al., 2020), mBART-50 (Tang et al., 2020) and, most recently, IndicBART (Dabre et al., 2021) are most commonly used for fine-tuning. None of these works focus on linguistic features, which is the key focus of our paper.

Feature Based NMT: Most works focusing on linguistic features, experiment on low-resource settings, and a majority of them focus on how to exploit syntactic/dependency structures of the source language (Eriguchi et al., 2016; Shi et al., 2016; Chen et al., 2017; Li et al., 2017; Wu et al., 2018; Zhang et al., 2019; Bugliarello and Okazaki, 2020). These works rely on various sophisticated

[3] Note that, adding features tends to make the training curves smoother, especially in bilingual settings which are comparatively lower resource than multilingual settings.

approaches but, Sennrich and Haddow (2016) show that enriching encoder word embeddings with morphological features is a simple but nice technique to exploit the features. Chakrabarty et al. (2020) improve upon this further by relevance mechanisms on top of morphological and syntactic feature embeddings, to enable effective use of features, an insight we adopt in this paper for feature based pre-training.

Multilingual NMT: Where feature based NMT focuses on utilizing linguistic information, multilingual NMT (Johnson et al., 2017; Dabre et al., 2020) focuses on leveraging training data for other languages to improve translation quality. Incorporating linguistic features into multilingual models has been neglected to the best of our knowledge, and our work aims to fill in this gap.

6 Conclusion

We have presented FeatureBART, an encoder-decoder pre-trained model that augments the encoder's embeddings with linguistic feature embeddings. Our experiments on English to Asian language translation in an extremely low-resource setting show that FeatureBART leads to better translation quality compares to its counterpart that does not use features. Analyses of training curves reveal that compared to pre-training without features, feature based pre-training leads to significantly lower perplexities during the initial stages of fine-tuning, which we think is responsible for improvement in translation quality. Future work will focus on: (1) exploring the impact of individual feature category on feature based pre-training and (2) multilingual version of FeatureBART which uses features in the encoder for languages other than English.

References

Naveen Arivazhagan, Ankur Bapna, Orhan Firat, Dmitry Lepikhin, Melvin Johnson, Maxim Krikun, Mia Xu Chen, Yuan Cao, George Foster, Colin Cherry, Wolfgang Macherey, Zhifeng Chen, and Yonghui Wu. 2019. Massively multilingual neural machine translation in the wild: Findings and challenges.

Emanuele Bugliarello and Naoaki Okazaki. 2020. Enhancing machine translation with dependency-aware self-attention. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 1618–1627, Online. Association for Computational Linguistics.

Abhisek Chakrabarty, Raj Dabre, Chenchen Ding, Masao Utiyama, and Eiichiro Sumita. 2020. Improving low-resource NMT through relevance based linguistic features incorporation. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 4263–4274, Barcelona, Spain (Online). International Committee on Computational Linguistics.

Huadong Chen, Shujian Huang, David Chiang, and Jiajun Chen. 2017. Improved neural machine translation with a syntax-aware encoder and decoder. *CoRR*, abs/1707.05436.

Raj Dabre, Chenhui Chu, and Anoop Kunchukuttan. 2020. A survey of multilingual neural machine translation. *ACM Comput. Surv.*, 53(5).

Raj Dabre, Himani Shrotriya, Anoop Kunchukuttan, Ratish Puduppully, Mitesh M. Khapra, and Pratyush Kumar. 2021. Indicbart: A pre-trained model for natural language generation of indic languages. *CoRR*, abs/2109.02903.

Akiko Eriguchi, Kazuma Hashimoto, and Yoshimasa Tsuruoka. 2016. Tree-to-sequence attentional neural machine translation. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 823–833, Berlin, Germany. Association for Computational Linguistics.

Cong Duy Vu Hoang, Reza Haffari, and Trevor Cohn. 2016. Improving neural translation models with linguistic factors. In *Proceedings of the Australasian Language Technology Association Workshop 2016*, pages 7–14, Melbourne, Australia.

Melvin Johnson, Mike Schuster, Quoc V. Le, Maxim Krikun, Yonghui Wu, Zhifeng Chen, Nikhil Thorat, Fernanda Viégas, Martin Wattenberg, Greg Corrado, Macduff Hughes, and Jeffrey Dean. 2017. Google's multilingual neural machine translation system: Enabling zero-shot translation. *Transactions of the Association for Computational Linguistics*, 5:339–351.

Guillaume Klein, Yoon Kim, Yuntian Deng, Jean Senellart, and Alexander M. Rush. 2017. OpenNMT: Open-source toolkit for neural machine translation. In *Proc. ACL*.

Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Veselin Stoyanov, and Luke Zettlemoyer. 2020. BART: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7871–7880, Online. Association for Computational Linguistics.

Junhui Li, Deyi Xiong, Zhaopeng Tu, Muhua Zhu, Min Zhang, and Guodong Zhou. 2017. Modeling source syntax for neural machine translation. *CoRR*, abs/1705.01020.

Qiang Li, Derek F. Wong, Lidia S. Chao, Muhua Zhu, Tong Xiao, Jingbo Zhu, and Min Zhang. 2018. Linguistic knowledge-aware neural machine translation. *IEEE/ACM Trans. Audio, Speech and Lang. Proc.*, 26(12):2341–2354.

Yinhan Liu, Jiatao Gu, Naman Goyal, Xian Li, Sergey Edunov, Marjan Ghazvininejad, Mike Lewis, and Luke Zettlemoyer. 2020. Multilingual denoising pre-training for neural machine translation. *Transactions of the Association for Computational Linguistics*, 8:726–742.

Christopher D. Manning, Mihai Surdeanu, John Bauer, Jenny Finkel, Steven J. Bethard, and David McClosky. 2014. The Stanford CoreNLP natural language processing toolkit. In *Association for Computational Linguistics (ACL) System Demonstrations*, pages 55–60.

Benjamin Marie, Atsushi Fujita, and Raphael Rubino. 2021. Scientific credibility of machine translation research: A meta-evaluation of 769 papers. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 7297–7306, Online. Association for Computational Linguistics.

Yirong Pan, Xiao Li, Yating Yang, and Rui Dong. 2020. Dual-source transformer model for neural machine translation with linguistic knowledge.

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. Bleu: a method for automatic evaluation of machine translation. In *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics*, pages 311–318, Philadelphia, Pennsylvania, USA. Association for Computational Linguistics.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J. Liu. 2020. Exploring the limits of transfer learning with a unified text-to-text transformer. *Journal of Machine Learning Research*, 21(140):1–67.

Hammam Riza, Michael Purwoadi, Gunarso, Teduh Uliniansyah, Aw Ai Ti, Sharifah Mahani Aljunied, Luong Chi Mai, Vu Tat Thang, Nguyen Phuong Thai, Rapid Sun, Vichet Chea, Khin Mar Soe, Khin Thandar Nwet, Masao Utiyama, and Chenchen Ding. 2016. Introduction of the Asian language treebank. In *Proc. of O-COCOSDA*, pages 1–6.

Rico Sennrich and Barry Haddow. 2016. Linguistic input features improve neural machine translation. In *Proceedings of the First Conference on Machine Translation: Volume 1, Research Papers*, pages 83–91, Berlin, Germany. Association for Computational Linguistics.

Rico Sennrich, Barry Haddow, and Alexandra Birch. 2016. Neural machine translation of rare words with subword units. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1715–1725, Berlin, Germany. Association for Computational Linguistics.

Xing Shi, Inkit Padhi, and Kevin Knight. 2016. Does string-based neural MT learn source syntax? In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 1526–1534, Austin, Texas. Association for Computational Linguistics.

Yuqing Tang, Chau Tran, Xian Li, Peng-Jen Chen, Naman Goyal, Vishrav Chaudhary, Jiatao Gu, and Angela Fan. 2020. Multilingual translation with extensible multilingual pretraining and finetuning. *CoRR*, abs/2008.00401.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Ł ukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, editors, *Advances in Neural Information Processing Systems 30*, pages 5998–6008. Curran Associates, Inc.

S. Wu, D. Zhang, Z. Zhang, N. Yang, M. Li, and M. Zhou. 2018. Dependency-to-dependency neural machine translation. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 26(11):2132–2141.

Linting Xue, Noah Constant, Adam Roberts, Mihir Kale, Rami Al-Rfou, Aditya Siddhant, Aditya Barua, and Colin Raffel. 2021. mT5: A massively multilingual pre-trained text-to-text transformer. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 483–498, Online. Association for Computational Linguistics.

Biao Zhang, Philip Williams, Ivan Titov, and Rico Sennrich. 2020. Improving massively multilingual neural machine translation and zero-shot translation. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 1628–1639, Online. Association for Computational Linguistics.

Meishan Zhang, Zhenghua Li, Guohong Fu, and Min Zhang. 2019. Syntax-enhanced neural machine translation with syntax-aware word representations. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 1151–1161, Minneapolis, Minnesota. Association for Computational Linguistics.

A Hyperparameters and Training Details

We perform extensive hyperparameter tuning to determine optimial settings for pre-training and

fine-tuning. We use 6 encoder and decoder layers, 8 multi-attention heads, $2,048$ dimension size of fully-connected network. For models without features, the token embedding and model hidden dimension is set to 512. For feature based models, following Sennrich et al. (2016) the embedding and hidden dimensions are 536 $(250, 250, 15, 15, 6$ corresponding to subword, lemma, POS, dependency label, and subword-tag), in order to make the number of parameters comparable. We use batch-sizes of $4,096$ tokens. The dropout rate is set to 0.3 for fine tuning experiments.

We pre-train models using token masking as well as text infilling, where span lengths are drawn from a Poisson distribution ($\lambda = 3$). We investigate the effect of different noise percentage from 10% to 80% and find that $50\% - 60\%$ noising is the optimum to get the best performance in downstream translation task. Pre-training is done for 3 epochs for each monolingual corpora of 2007, 2010, and 2013, covering 43200, 75600, and 237600 training steps respectively. During fine-tuning, maximum training steps are set as 200000 with validation accuracy performed after every 10000 steps.

Multi-level Community-awareness Graph Neural Networks for Neural Machine Translation

Binh Nguyen[1,2], **Long Nguyen**[1,2]*, **Dien Dinh**[1,2]
[1]Faculty of Information Technology, University of Science, Ho Chi Minh City, Vietnam
[2]Vietnam National University, Ho Chi Minh City, Vietnam
nqbinh17@apcs.fitus.edu.vn, {nhblong,ddien}@fit.hcmus.edu.vn

Abstract

Neural Machine Translation (NMT) aims to translate the source- to the target-language while preserving the original meaning. Linguistic information such as morphology, syntactic, and semantics shall be grasped in token embeddings to produce a high-quality translation. Recent works have leveraged the powerful Graph Neural Networks (GNNs) to encode such language knowledge into token embeddings. Specifically, they use a trained parser to construct semantic graphs given sentences and then apply GNNs. However, most semantic graphs are tree-shaped and too sparse for GNNs which cause the over-smoothing problem. To alleviate this problem, we propose a novel Multi-level Community-awareness Graph Neural Network (MC-GNN) layer to jointly model local and global relationships between words and their linguistic roles in multiple communities. Intuitively, the MC-GNN layer substitutes a self-attention layer at the encoder side of a transformer-based machine translation model. Extensive experiments on four language-pair datasets with common evaluation metrics show the remarkable improvements of our method while reducing the time complexity in very long sentences.

1 Introduction

Self-attention mechanisms, introduced by Transformers (Vaswani et al., 2017), have been ubiquitous in Neural Machine Translation (NMT). It encodes the relative information of a position based on other positions in the same sequence. Recent works have seen the rising trend of improving the self-attention module such as (Choromanski et al., 2021; Katharopoulos et al., 2020).

One of the serious self-attention weaknesses is its inefficiency at computing long sentences due to quadratic complexity. Following this problem, Performer (Choromanski et al., 2021) estimates a full-rank-attention matrix with Fast Attention Via positive Orthogonal Random features approach (FAVOR+), that trade-offs the time complexity from $O(T^2d)$ to $O(Td^2logd)$. However, as they scale the model larger such as GPT-3 (Brown et al., 2020), the quadratic dimensional space d exponentially scales much faster than that of the sequence length T. In addition, Linear Transformer (Katharopoulos et al., 2020) reported a comparable performance with the Transformer, while speeding up on autoregressive inference time with a linear dot-product of kernel feature maps. Several studies replace self-attention with an attention-free module, which significantly speeds up the training time. Attention Free Transformer (ATF) (Zhai et al., 2021) eliminates the dot-product attention with weighted element-wise multiplication, which linearizes the time complexity to $O(Td)$. FNet (Lee-Thorp et al., 2021) is a pre-trained model with Discrete Fourier Transform that speeds up 80% training time on GPU but still achieves 92-97% of the accuracy of BERT (Devlin et al., 2018). These works make a trade-off between the accuracy and the performance to reduce the computational cost.

Another line of work constructs a content-based sparse graph to select keys that have high similarity meanings to queries. Routing Transformer (Roy et al., 2020) applies k-means clustering to cluster both queries and keys. Meanwhile, Reformer (Kitaev et al., 2020) uses locality-sensitive hashing (LSH) to select key-value pairs for each query.

Unlike previous work, in this paper, we follow the intuition that self-attention is assumption-free on the structural input, and thus it is hard to induce task-based generalization on small-scale data. As a result, we propose semantic-based sparse attention that utilizes the semantic graph construction of the Universal Conceptual Cognitive Annotation (UCCA) (Abend and Rappoport, 2013) parser, which is implemented based on Message Passing Neural Networks (MPNN). The vital benefit of

*Corresponding author

Proceedings of the 29th International Conference on Computational Linguistics, pages 5021–5028
October 12–17, 2022.

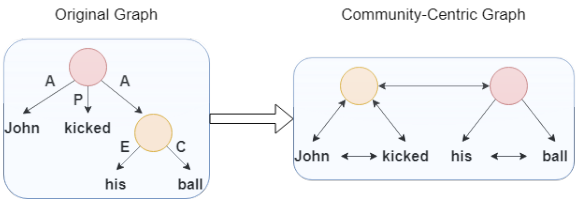

Figure 1: A UCCA graph (left) for sentence "John kicked his ball" (Abend and Rappoport, 2013) and its community-centric graph (right). The colored circles are intermediate nodes.

UCCA is that it is not relatively defined to a specific language, or in another word, it provides a universal structure across languages. We hypothesize that the translation task will benefit from this characteristic. According to (Abend and Rappoport, 2013), because UCCA is a connected acyclic graph (or tree) whose in-degree node usually is one or two, the attention-based computation will smooth the token embeddings to a high similarity score. This problem is referred to as the over-smoothing problem, leading to the bad convergence when the MPNN model is trained deeper.

To alleviate the issue, we introduce a set of transformation rules to convert the tree-based UCCA into a Community-Centric Graph (CCG). As illustrated in Figure 1, the CCG (on the right) presents two different communities that somehow discover local relationships between words and word roles in a community. Moreover, we can theoretically prove that the CCG structure is stronger than the tree-based UCCA (see Section 3.2). To capture the global relationships across communities, we propose Multi-level Explorations (ME) to probe multi-level structures in a graph. Henceforth, we combine CCG and ME into Multi-level Community-awareness Graph Neural Networks (MC-GNN). This MC-GNN layer will substitute a self-attention layer in the Transformer encoder.

MC-GNN is theoretically faster than the Transformer, thanks to the sparse attention. Table 1 show the time complexities for the Trasformer variants. Experimental results showed that MC-GNN outperforms the Transformer-based MT (Vaswani et al., 2017) in three over four NMT benchmarks across four evaluation metrics (Section 4.2). Moreover, we observe that Performer, Linear Transformer, ATF, and FNet are underperformed in translation tasks. We also report that the time complexity is equal to at most 60% and 10% of the self-attention in short and long sentences, respectively (Section 4.3). The contributions of this work are as follows:

Table 1: Complexity comparision. Here T, d, and k denote the sequence length, hidden embedding, and the average node degree, respectively. $k << T$ for the sparse attention.

Model	Time
Transformer	$O(T^2d)$
FNET	$O(1)$
Performer	$O(Td^2logd)$
AFT	$O(T^2d)$
Linear Attention	$O(Td^2)$
MC-GNN	$O(Tkd)$

- We propose a Community-Centric Graph to provide community awareness and alleviate the over-smoothing problem in GNN.
- We propose Multi-level Explorations (ME) to probe multi-level structures in a graph.
- We prove the robustness and effectiveness of MC-GNN by extensive analyses.

2 Background and Related Work

2.1 Graph-based Approach in NMT

The explicit incorporation of linguistic information into traditional statistical machine translation has yielded many positive results (Bazrafshan and Gildea, 2013), and therefore it is intuitive to incorporate additional knowledge into NMT. Yet, governing restrictive constraints on the interaction between external information and the translation task will hamper MT performance. Bastings et al. (2020) introduced a graph-based encoder to effectively blend the syntactic structure into NMT. Meanwhile, Marcheggiani et al. (2018) embedded semantic bias into word representations by GNN, and thus provided semantic awareness to NMT. Xu et al. (2021) incorporated contextual awareness to document-level NMT by GNN, where a document is transformed into a graph that links relevant contexts (i.e., named entity) regardless of their distances. These methods, however, require a larger computational cost than the Transformer in the same settings.

2.2 Graph Attention Networks

Graph Attention Networks (Veličković et al., 2017) compute the representation of each node by the

weighted sum over its neighbors, following the graph-structured data. They introduce a set of node features as $h = \{h_1, h_2, ..., h_N\}$, $h_i \in R^D$ where N is the number of nodes, and D is the dimensional embeddings. To learn the higher-order information from the graph, they compute attention coefficients from two adjacent nodes as follows:

$$e_{ij} = LeakyReLU(a^T[Wh_i||Wh_j]) \quad (1)$$

e_{ij} is considered as the significance of node j to node i. Moreover, $a \in R^{2D}$ and $W \in R^D$ are trainable parameters, the *LeakyReLU* nonlinearity uses the negative slope $\alpha = 0.2$. They normalize the attention coefficients by using the softmax function as follows:

$$\alpha_{ij} = \frac{exp(e_{ij})}{\sum_{k \in N(i)} exp(e_{ik})} \quad (2)$$

where $k \in N(i)$ means for every neighbor of node i. Finally, the aggregated information of node i will be updated as follows:

$$\hat{h}_i = \sum_{k \in N(i)} \alpha_{ik} W h_k \quad (3)$$

The multi-head attention is employed to improve inductive learning, which is similar to Transformer (Vaswani et al., 2017).

2.3 Improvement of self-attention

Self-attention is the heart of many NLP-related deep learning tasks (Guo et al., 2019; Zhang et al., 2021; Peng et al., 2021; Lee-Thorp et al., 2021). Three profound elements matter in self-attention are: (1) computation complexity, (2) memory footprint, and (3) performance. Fundamentally, no research outperforms the Transformer in all three criteria in NMT because the trade-off between them always exists. For example, FNet (Lee-Thorp et al., 2021) replaces self-attention with an unparameterized Fourier Transform that reduces the training time by seven times on GPU and achieves 97% of the BERT-Base accuracy on GLUE. However, we find that FNet significantly underperforms against the Transformer in MT tasks. Random Feature Attention (Peng et al., 2021) linearizes the softmax function with random feature methods based on the Gaussian kernel (Rahimi and Recht, 2007). Especially they speed up the decoding time by two times and slightly improve the BLEU score by 0.1 BLEU in WMT14 En-De. Guo et al. (2019) substituted the fully-connected structure with a star-shaped topology, and thus time complexity is reduced from quadratic to linear. We, instead, replace the self-attention layer with MC-GNN and can gain better performance while reducing the complexity.

2.4 Universal Conceptual Cognitive Annotation

UCCA is a bi-lexical dependency graph whose goal is to abstract semantics away from syntactic interpretations in a typological and cross-linguistic fashion (Abend and Rappoport, 2013). To that aim, a multi-layer formalism is defined in UCCA, in which each layer specifies the relations it encodes. For example, the left graph of Figure 1 includes a single process scene (P), which describes an action or a movement that evolves in time. The process "kicked" contains two participants (A), "John" and "his ball". The participant "his ball" is further annotated with a center (C) and an elaborator (E).

3 Method

In this section, we firstly provide notations and theory of Community-awareness Graph (CCG). We then provide the proof that CCG is stronger than tree graph. We next present details of MC-GNN, and finally explain how MC-GNN is incorporated into a NMT model.

3.1 Graph Notations & Community-Centric Graph

Notation. Let G be a directed graph with N nodes and M edges. A non-symmetric adjacency matrix is defined as $A \in \{0,1\}^{NxN}$ and the embeddings of nodes are $X \in R^{NxD}$ where D is the length of embeddings. A node i points to a node j if and only if $A_{ij} = 1$. We denote $\hat{A} = A + I_N$ with I_N is an identity matrix, and $\hat{D} = diag(\hat{A}\mathbf{1})$ a degree matrix where $diag(\cdot)$ creates a diagonal matrix and $\mathbf{1}$ is the all ones vector. A normalized Laplacian matrix is computed by $\hat{L} = \hat{D}^{-1/2}\hat{A}\hat{D}^{-1/2}$.

Community-Centric Graph. Let S: $A_U \rightarrow A_C$ be a set of transformation rules to convert a UCCA graph $G_U = (A_U, X)$ to a community-centric graph $G_C = (A_C, X)$. We propose three following rules that only apply to the UCCA graph.

- Same direction: $(A_C)_{ij} = 1$ if and only if (iff) $(A_U)_{ik} = (A_U)_{kj} = 1$ (Figure 2a).
- Opposite direction: $(A_C)_{ij} = (A_C)_{ji} = 1$ iff $(A_U)_{ui} = (A_U)_{uj} = 1$ (Figure 2b).

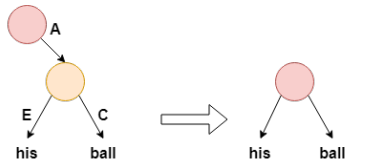

(a) Same direction

(b) Opposite direction

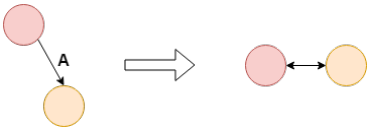

(c) Community link

Figure 2: Transformation rules

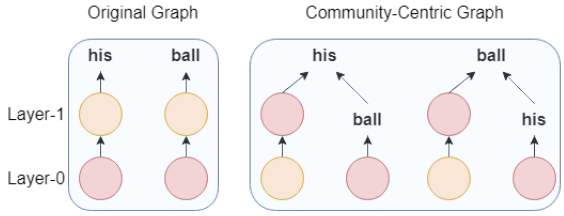

Figure 3: The left graph is indistinguishable after 1 layer, while the community centric graph (right) can be distinguished.

- Community link: $(A_C)_{ij} = (A_C)_{ji} = 1$ if $(A_U)_{ij}$ or $(A_U)_{ji} = 1$ and two nodes i, j are the intermediate nodes (Figure 2c).

In fact, the community-centric graph is inspired by a line graph (Gross and Yellen, 2005) that has been recently applied in (Zhao et al., 2020; Cao et al., 2021).

3.2 Community-Centric Graph is stronger than Tree Graph

Given an adjacency matrix A and node embeddings X, we define an attention-based GNN to compute graph-level embeddings as follows:

$$\hat{X} = GNN(A, X) \tag{4}$$

In the left-hand side of Figure 3, since the node *his* and node *ball* have the same set of neighbors, the weighted sum GNN of this set produces the equivalent embedding. As a result, the embeddings are undistinguished or over-smoothed after t layer GNNs. However, we have two different sets of neighbors in the community-centric graph, and thus the final embeddings are different.

3.3 Multi-level Explorations of Graph Structures

Motivated by rich closed-path sub-structures in the community-centric graph G_C, we propose a novel method that creates an adjacency matrix A_M to explore a multi-level structure in G_C. The profound idea is to utilize the adjacency matrix A_C as follows:

$$A_M^{(L)} = A_C^T \odot A_C^{(L)} \tag{5}$$

where L is the power of a matrix, $\odot$ is the element-wise multiplication. Intuitively, Equation 5 indicates that $(A_M^{(L)})_{ij} = 1$ if and only if there exists a path length L from node i to node j in $A_C^{(L)}$ and exists a path from node j to node i. To explore multi-level structures, we add matrix $A_M^{(t)}$ at each tth encoder layer; if the standard Transformer has six layers then it probes six different levels.

Intuitively, the community-centric graph reveals the role of words in a local community, while the multi-level structure bright-to-lights the hidden role of words in global.

3.4 Multi-level Community-awareness GNN (MC-GNN)

Motivated by the multi-head self-attention that has been widely used in various NLP tasks, we introduce multi-head attention-based message propagation. Given a graph $G = (X^{(t)}, A)$ at a time step t, we update each node features $x_i^{(t)}$ as follows:

$$\begin{aligned} x_i^{(t+1)} &= GNN(x_i^{(t)}, A) \\ GNN(x_i^{(t)}, A) &= \sigma(W_1 x_i^{(t)} + F(x_i^{(t)}, A)) \end{aligned} \tag{6}$$

where σ is a PReLU activation function, and W_1 is trainable parameters. F is a scaled multi-head attention function defined in Equation 7.

$$\begin{aligned} F(x_i^{(t)}, A) &= \sum_{A_{i,j}=1} \alpha_{i,j} W_2 x_j^{(t)} \\ \alpha_{i,j} &= softmax(\frac{(W_2 x_i^{(t)})^T (W_2 x_j^{(t)})}{\sqrt{d}}) \end{aligned} \tag{7}$$

where W_2 is a matrix with trainable parameters, d is the dimensional embedding, and a multi-head annotation is abandoned for sake of simplicity. To apply MC-GNN, we first combine the multi-level graph $A^{(t)}$ and community-centric graph A_C, and then replace it in Equation 6 as follows:

$$\begin{aligned} A_{MC}^{(t-1)} &= A_M^{(t-1)} + A_C \\ x_i^{(t+1)} &= GNN(x_i^{(t)}, \hat{L}_{MC}^{(t-1)}) \end{aligned} \tag{8}$$

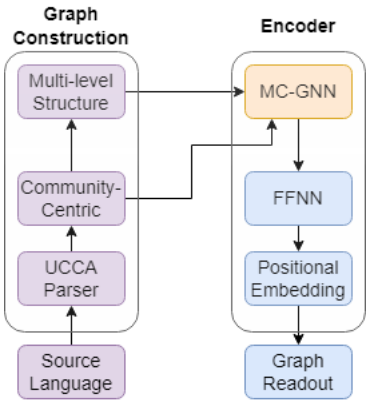

Figure 4: The Transformer-based NMT with the MC-GNN layer. Graph Construction is pre-processed and saved to files to speed up. Positional Embedding is added to token embeddings at the end of a layer yielding the better performance than added outside the Transformer block.

where $\hat{L}_{MC}^{(t-1)}$ is the Laplacian graph introduced in Section 3.1. Consequently, UCCA- and Community Centric- GNN replace the UCCA A_U and community centric A_C graphs in Equation 6. The computational complexity of the Transformer and the above methods will be examined in Section 4.3.

3.5 Model Architecture

Figure 4 illustrates the complete process and the model architecture when we incorporate MC-GNN into a NMT system. A source sentence is firstly passed through a graph construction process to convert the sentence to a graph structure. In this process, we can consider three different options: (1) to use the original UCCA, (2) to produce CCG using the transformation rules in Section 3.1, and (3) to aggregate multi-level structures as explained in Section 3.3. Next, based on the output from the graph construction process, graph information is produced by the MC-GNN layer and fed into Feed-Forward Neural Networks (FFNN). The positional information of token embeddings is mixed up with their neighbors during the message propagation, so positional embedding is added after the FFNN. The graph readout is necessary for removing irrelevant nodes and keeping token orders. Lastly, the decoder side is the same as the Transformer.

4 Experiments

We provide hyper-parameter settings, datasets, parser settings, and our code at https://github.com/nqbinh17/mc-gnn.

4.1 Settings

We experiment on four IWSLT language-pair datasets: English-French (En-Fr), English-Vietnam (En-Vi), English-Czech (En-Cz), and English-German (En-De). Moreover, we use a standard hyper-parameter for every model. The number of the encoder, and decoder layers is set to 6, the model embedding is 512, the intermediate size is 2048, the number of multi-head is 8, and the dropout is 0.3. For En-Cz, however, the numbers of the encoder and multi-head are set to 5 and 2, respectively.

4.2 Ablation Study

In this study, the performances of UCCA-, Community-Centric-, and MC-GNN are compared against the Transformer on the En-Fr dataset. Finally, we carry out the evaluations on the other three datasets to reveal the robustness of the MC-GNN.

- **Transformers** is the vanilla Transformer with the self-attention mechanism.
- **UCCA-GNN** is the Transformer-based model but replace self-attention with UCCA-GNN.
- **CC-GNN** and **MC-GNN** use the same model as UCCA-GNN but different graph constructions.

We first validate the argument in Section 3.2 on the tree-based (UCCA-GNN) graph and the community-centric (CC-GNN, MC-GNN) graph. Table 2 shows that CC- and MC- GNN improve UCCA-GNN by -0.71 and -0.76 BLEU in the 2014 testset. The outcome is impressive, because there is no change in the GNN architecture but with a simple graph pre-process.

On four language-pair datasets, we observe the consistent performances of MC-GNN as compared to the Transformer across four evaluation matrices. The improvements are substantial on En-Fr, En-Vi, and En-Cz, while the number of parameters are slightly lower. Nevertheless, MC-GNN consistently under-performed in the En-De dataset.

At first, we assumed that UCCA will be benefited from similar language pairs such as English, Vietnamese, and French, yet this argument fails in English-German that share similarities in grammar. Overall, MC-GNN is competitive to the Transformer in terms of performance and model size.

Table 2: The ablation study is carried out to show the effectiveness of the proposed method. The bold text is to highlight the least parameters or the highest performance of the model in a dataset.

Model	Dataset	Para. (M)	Test	GLEU	NIST	METEOR	BLEU
Transformer	IWSLT En-Fr	**128.0**	tst2014	37.77	7.51	56.75	35.76
			tst2015	38.32	7.48	56.64	36.16
UCCA-GNN	IWSLT En-Fr	124.9	tst2014	37.47	7.58	57.12	35.74
			tst2015	37.42	7.40	56.06	35.20
CC-GNN	IWSLT En-Fr	124.9	tst2014	38.17	7.65	57.5	36.45
			tst2015	38.47	7.53	56.92	36.23
MC-GNN	IWSLT En-Fr	124.9	tst2014	**38.22**	**7.65**	**57.50**	**36.50**
			tst2015	**38.51**	**7.55**	**57.10**	**36.35**
Transformer	IWSLT En-Vi	**82.4**	tst2013	31.74	6.91	N/A	28.23
			tst2015	29.05	6.39	N/A	26.31
MC-GNN	IWSLT En-Vi	79.2	tst2013	**31.97**	**6.98**	N/A	**28.33**
			tst2015	**29.22**	**6.44**	N/A	**26.37**
Transformer	IWSLT En-Cz	**126.0**	tst2011	21.88	4.71	23.71	16.48
			tst2013	22.70	5.03	24.18	17.51
MC-GNN	IWSLT En-Cz	123.3	tst2011	**22.25**	**4.95**	**24.81**	**16.95**
			tst2013	**22.99**	**5.23**	**25.14**	**18.10**
Transformer	IWSLT En-De	**148.4**	tst2014	**28.17**	6.05	**43.63**	**24.11**
			tst2015	**30.10**	**6.24**	44.69	**26.05**
MC-GNN	IWSLT En-De	145.8	tst2014	28.15	**6.08**	43.50	23.98
			tst2015	29.42	6.13	**45.35**	25.46

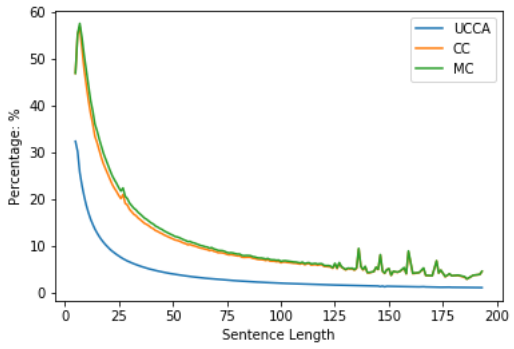

Figure 5: The computational complexity exponentially scales down as sentence length grows.

4.3 Computational Complexity - Experimental Analysis

Since understanding the computational complexity of a model is critical in NLP, we analyze this aspect of the proposed models and the self-attention layer based on sentence length.

The self-attention function connects all tokens in a sentence with themselves, so its complexity is $O(n^2)$, where n is the sentence length. Meanwhile, the attention-based GNN attends each node to their neighbor producing $O(k \cdot n)$ complexity. k is the average degree, and n is the number of nodes or the sentence length. For UCCA graphs, k is significantly smaller than n, and hence $O(k \cdot n)$ is substantially lesser than $O(n^2)$. To give a better illustration, we compare the proportion of k over n in the IWSLT English-to-French datasets to see the decline in real data.

The y-axis of Figure 5 shows the proportion of $O(k \cdot n)$ over $O(n^2)$. We can see that the time complexity of UCCA-GNN is 32% of the Transformer at most and exponentially shrinks to 3%. Meanwhile, the time complexity of CC- and MC-GNN is nearly 60% of the Transformer with a sentence length of 15, and significantly dropped to 10% when the sentence length is longer than 100. Empirically, k is exponentially smaller than n as the sentence length grows, and hence diminishing $O(n^2)$ to $O(k \cdot n)$ enables tasks involving very long sentences.

4.4 Compared with other methods

In this section, we compare our method with FNet (Lee-Thorp et al., 2021), Performer (Choromanski et al., 2021), AFT (Zhai et al., 2021), and Linear Attention (Katharopoulos et al., 2020) in translation tasks. We substitute them with the self-attention

module in the encoder layer, and the experimental results are shown in Table 3.

FNet: is parameter-free with the $O(1)$ time complexity, and thus we double the number of encoder layers for fairness. We find that FNet extremely underfits in translation tasks, while it showed the comparative results against BERT in the encoder-only tasks. We believe that generative tasks are too rigid for Discrete Fourier Transform.

Performer[1]: We set the number of random features to 4 times the head dimension. Although we manually finetune the hyper-parameter settings, the performance is poor on all language-pair datasets.

AFT: We use AFT-full with the $O(T^2d)$ time and $O(Td)$ space complexities. We observe that AFT-full has a better convergence and is more stable when training. The performances are comparative where the drops fluctuate from -0.76 to -2.07 BLEU.

Linear Attention: has the $O(Td^2)$ time complexity, however, the official implementation [2] produced the four-dimensional computation. This makes space complexity spike during the training process, and thus we must use a much smaller batch size - only 16 sentences per batch. Due to the small batch size, Linear Attention is hard to converge and unable to converge in the En-Cz dataset.

Overall, FNet has the fastest training time but the poorest performance among the methods. The AFT method is much more reliable than FNet, Performer, and Linear Attention in translation tasks with stable convergence. Moreover, the results convince us that MC-GNN is more consistent than these works and even outperforms the Transformer in some datasets.

5 Conclusion

MC-GNN gains significant improvements across many translation tasks without changing the architecture, while parameters and the time complexity are reduced. However, there are several limitations in the proposed methods. First, the accuracy of the UCCA parser is below 80%, which can cause some unexpected behaviors and hamper the NMT performance. Second, MC-GNN does not support every language as self-attention, since the UCCA parser is unavailable in every language. Third, we experimented with the graph decoder where a target sentence is treated as a dense graph and built incrementally during the decoding phase (Xu et al., 2021). Although the time complexity of dense graphs and self-attention are $O(T^2d)$, dense graphs are sparse tensor computations, which is extremely memory inefficient and unparallelable. Thus, we dropped this approach due to limited resources.

[1]We use the implementation from https://github.com/lucidrains/performer-pytorch.

[2]https://github.com/idiap/fast-transformers

Acknowledgements

The authors would like to thank Computational Linguistics Center (University of Sciences, HCMC-VNU) for a great support.

We would like to thank the reviewers for their helpful comments.

References

Omri Abend and Ari Rappoport. 2013. Universal Conceptual Cognitive Annotation (UCCA). In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 228–238, Sofia, Bulgaria. Association for Computational Linguistics.

Jasmijn Bastings, Ivan Titov, Wilker Aziz, Diego Marcheggiani, and Khalil Sima'an. 2020. Graph convolutional encoders for syntax-aware neural machine translation.

Marzieh Bazrafshan and Daniel Gildea. 2013. Semantic roles for string to tree machine translation. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 419–423, Sofia, Bulgaria. Association for Computational Linguistics.

Tom B. Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel M. Ziegler, Jeffrey Wu, Clemens Winter, Christopher Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. 2020. Language models are few-shot learners. *CoRR*, abs/2005.14165.

Ruisheng Cao, Lu Chen, Zhi Chen, Yanbin Zhao, Su Zhu, and Kai Yu. 2021. Lgesql: Line graph enhanced text-to-sql model with mixed local and non-local relations.

Krzysztof Choromanski, Valerii Likhosherstov, David Dohan, Xingyou Song, Andreea Gane, Tamas Sarlos, Peter Hawkins, Jared Davis, Afroz Mohiuddin, Lukasz Kaiser, David Belanger, Lucy Colwell, and Adrian Weller. 2021. Rethinking attention with performers.

Table 3: Experimented on four language-pair datasets.

Model	En-Fr	En-Vi	En-Cz	En-De
Transformer	35.76	28.23	16.48	24.11
FNET	28.35 ↓ 7.41	12.69 ↓ 15.54	11.92 ↓ 4.56	16.90 ↓ 7.21
Performer	28.70 ↓ 7.01	23.26 ↓ 4.97	12.87 ↓ 3.61	20.63 ↓ 3.48
AFT	35.00 ↓ 0.76	27.09 ↓ 1.14	15.14 ↓ 1.34	22.04 ↓ 2.07
Linear Attention	27.64 ↓ 8.12	26.42 ↓ 1.81	N/A	16.15 ↓ 7.96
MC-GNN	36.50 ↑ 0.74	28.33 ↑ 0.10	16.95 ↑ 0.47	23.99 ↓ 0.13

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. BERT: pre-training of deep bidirectional transformers for language understanding. *CoRR*, abs/1810.04805.

Jonathan L. Gross and Jay Yellen. 2005. *Graph Theory and Its Applications, Second Edition (Discrete Mathematics and Its Applications)*. Chapman & Hall/CRC.

Qipeng Guo, Xipeng Qiu, Pengfei Liu, Yunfan Shao, Xiangyang Xue, and Zheng Zhang. 2019. Star-transformer.

Angelos Katharopoulos, Apoorv Vyas, Nikolaos Pappas, and François Fleuret. 2020. Transformers are rnns: Fast autoregressive transformers with linear attention. *CoRR*, abs/2006.16236.

Nikita Kitaev, Lukasz Kaiser, and Anselm Levskaya. 2020. Reformer: The efficient transformer. *CoRR*, abs/2001.04451.

James Lee-Thorp, Joshua Ainslie, Ilya Eckstein, and Santiago Ontañón. 2021. Fnet: Mixing tokens with fourier transforms. *CoRR*, abs/2105.03824.

James Lee-Thorp, Joshua Ainslie, Ilya Eckstein, and Santiago Ontanon. 2021. Fnet: Mixing tokens with fourier transforms.

Diego Marcheggiani, Jasmijn Bastings, and Ivan Titov. 2018. Exploiting semantics in neural machine translation with graph convolutional networks. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 2 (Short Papers)*, pages 486–492, New Orleans, Louisiana. Association for Computational Linguistics.

Hao Peng, Nikolaos Pappas, Dani Yogatama, Roy Schwartz, Noah A. Smith, and Lingpeng Kong. 2021. Random feature attention.

Ali Rahimi and Benjamin Recht. 2007. Random features for large-scale kernel machines. In *Proceedings of the 20th International Conference on Neural Information Processing Systems*, NIPS'07, page 1177–1184, Red Hook, NY, USA. Curran Associates Inc.

Aurko Roy, Mohammad Saffar, Ashish Vaswani, and David Grangier. 2020. Efficient content-based sparse attention with routing transformers. *CoRR*, abs/2003.05997.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need.

Petar Veličković, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Liò, and Yoshua Bengio. 2017. Graph attention networks. *6th International Conference on Learning Representations*.

Mingzhou Xu, Liangyou Li, Derek. F. Wong, Qun Liu, and Lidia S. Chao. 2021. Document graph for neural machine translation.

Shuangfei Zhai, Walter Talbott, Nitish Srivastava, Chen Huang, Hanlin Goh, Ruixiang Zhang, and Josh M. Susskind. 2021. An attention free transformer. *CoRR*, abs/2105.14103.

Biao Zhang, Ivan Titov, and Rico Sennrich. 2021. Sparse attention with linear units.

Yanbin Zhao, Lu Chen, Zhi Chen, Ruisheng Cao, Su Zhu, and Kai Yu. 2020. Line graph enhanced AMR-to-text generation with mix-order graph attention networks. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 732–741, Online. Association for Computational Linguistics.

On the Complementarity between *Pre-Training* and *Random-Initialization* for Resource-Rich Machine Translation

Changtong Zan[1*], **Liang Ding**[2†], **Li Shen**[2], **Yu Cao**[3], **Weifeng Liu**[1†], **Dacheng Tao**[2,3]
[1]College of Control Science and Engineering, China University of Petroleum (East China)
[2]JD Explore Academy [3]The University of Sydney
b20050011@s.upc.edu.cn, dingliang1@jd.com

Abstract

Pre-Training (PT) of text representations has been successfully applied to low-resource Neural Machine Translation (NMT). However, it usually fails to achieve notable gains (sometimes, even worse) on resource-rich NMT on par with their Random-Initialization (RI) counterpart. We take the first step to investigate the complementarity between PT and RI in resource-rich scenarios via two probing analyses, and find that: ❶ PT improves NOT the *accuracy*, but the *generalization* by achieving flatter loss landscapes than that of RI; ❷ PT improves NOT the confidence of lexical choice, but the *negative diversity*[1] by assigning smoother lexical probability distributions than that of RI. Based on these insights, we propose to combine their complementarities with a model fusion algorithm that utilizes optimal transport to align neurons between PT and RI. Experiments on two resource-rich translation benchmarks, WMT'17 English-Chinese (20M) and WMT'19 English-German (36M), show that PT and RI could be nicely complementary to each other, achieving substantial improvements considering both translation accuracy, generalization, and negative diversity. Probing tools and code are released at: https://github.com/zanchangtong/PTvsRI.

1 Introduction

Pre-training (**PT**; Devlin et al., 2019; Liu et al., 2019 has achieved tremendous success in natural language processing fields. Inspired by BERT (Devlin et al., 2019), recent works (Song et al., 2019; Lewis et al., 2020; Liu et al., 2020) attempt to leverage sequence-to-sequence PT for neural machine translation (NMT; Bahdanau et al., 2015; Gehring et al., 2017; Vaswani et al., 2017) by leveraging a large amount of unlabeled (i.e. monolingual) sentences.

While recent studies have empirically shown their benefit for the *low-resource* translation task where the labeled (i.e. parallel) sentences are limited (Lewis et al., 2020; Song et al., 2019; Liu et al., 2020), we are generally confronted with *resource-rich* scenarios, e.g. millions of parallel sentence pairs, in WMT evaluations (Akhbardeh et al., 2021) and industries. For these resource-rich tasks, PT becomes less effective (sometimes, even worse) than their Random-Initialization (**RI**) counterparts, for example, as Zhu et al. 2020; Liu et al. 2020 reported, the PT underperforms RI if improperly utilized or significant amount of bi-text data is given. However, there is limited understanding of: ① Why does PT fail compared to RI in terms of translation accuracy in high-resource settings? ② What is the difference between the optimized PT and RI models? ③ How can we harmonize PT with RI? Can we just leverage their advantages?

To this end, we introduce two probing analyses (i.e. loss landscape and lexical probability distribution) for the fully-optimized PT and RI models to investigate ① and ② , respectively. We find that: ❶ Supervised training with significant amount of parallel data, i.e. RI, is enough to optimize the model towards a better optimum point compared with PT, thus leading to better in-domain (same domain between test and training set) performance; ❷ PT mainly contributes to the generalization ability because of its flatter loss landscape and smoother lexical probability distribution, thus resulting in better out-domain performance and diversified generation. Motivated by this finding, we propose a simple combination method to reach ③ by aligning the neurons and weights

[*] Work was done when Changtong was interning at JD Explore Academy.

[†] Liang Ding and Weifeng Liu are the corresponding authors.

[1] Li et al. 2020 mentioned that the maximum likelihood estimation training treats all non-ground truth but semantic-relevant lexical predictions as being equally incorrect, potentially leading to the sharp lexical probability distribution. Here we use this term to refer to the higher probabilities assigned to the negative tokens.

Proceedings of the 29th International Conference on Computational Linguistics, pages 5029–5034
October 12–17, 2022.

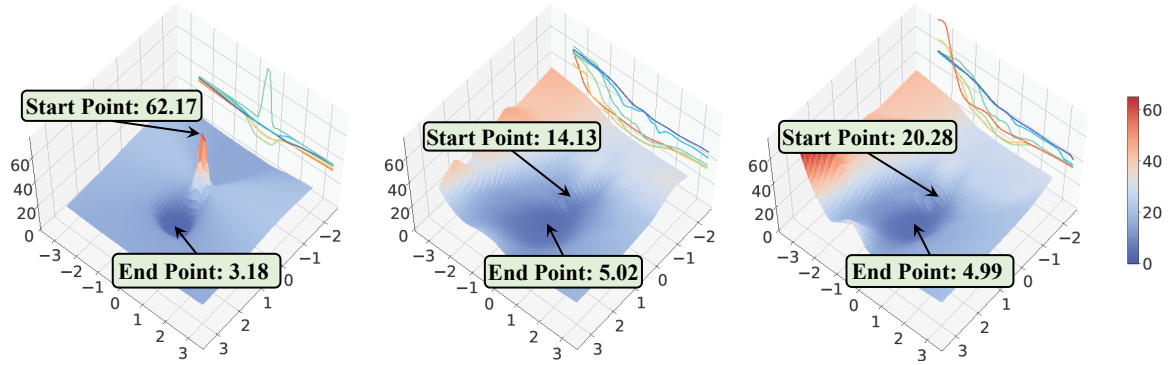

(a) Loss Landscape of RI **(b) Loss Landscape of PT** **(c) Loss Landscape of Ours**

Figure 1: Loss landscapes with contour lines on WMT19 En-De dataset. "RI" means the transformer training from scratch, while "PT" denotes the pretrained model mBART with finetuning. Both models have the same architecture.

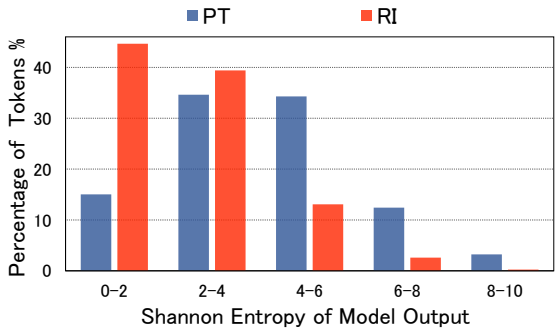

Figure 2: Token ratio distribution over difference entropy scales on WMT19 En-De test set. We compute the shanon entropy of the decoder output for each token.

of PT and RI and then fusing them into a single model based on optimal transport (Monge, 1781; Singh and Jaggi, 2020). Experiments conducted on the WMT17 English-Chinese and WMT19 English-German benchmarks show that PT can nicely complement RI, leading to better translation accuracy, generalization, and negative diversity.

2 Experimental Setup

Data We conducted experiments on WMT17 English-Chinese (En-Zh) and WMT19 English-German (En-De) translation tasks, which are widely-used resource-rich benchmarks, and include 20M and 36M sentence pairs, respectively.

Setting To make a fair comparison, all the model backbone architectures are the same as the pretrained `mBART.cc25`[2]. We tokenize with 250K SentencePiece model (Kudo and Richardson, 2018) of mBART and remove tokens that are not present in the downstream task from the vocabulary. For PT, we also initialize the model with trimmed embedding layers. For evaluation, we use SacreBLEU (Post, 2018) to measure the translation quality with default tokenizer.

[2] https://github.com/pytorch/fairseq/tree/master/examples/mbart

3 Understanding PT and RI

In this section, we aim to better understand the similarities and differences between PT and RI by introducing two probing analyses. The analyzed PT and RI models are trained on WMT19 En-De benchmark. We first present the loss landscape visualization of both models in Section 3.1, and then show their difference in lexical probability distribution in Section 3.2.

3.1 Impact on Loss Landscape

We follow Hao et al. (2019b) to visualize the two-dimensional (2D) loss surface, providing some insights on why PT fails compared to RI for in-domain accuracy while resulting better out-domain performance. Let θ_0 denotes the initialized parameters, i.e., mBART parameters or randomly initialized parameters. θ_1 and θ_2 are the parameters of models trained on target dataset and auxiliary dataset respectively. We plot the loss surface with function as follows:

$$f(\alpha, \beta) = \mathcal{L}(\boldsymbol{\theta}_0 + \alpha\boldsymbol{\delta}_1 + \beta\boldsymbol{\delta}_2), \quad (1)$$

where $\mathcal{L}$ is the loss function, α and β are scalar values represent current coordinate, $\boldsymbol{\delta}_1 = \theta_1 - \theta_0$ is the the optimization direction on target dataset, and $\boldsymbol{\delta}_2 = \frac{\|\theta_1 - \theta_0\|}{\|\theta_2 - \theta_0\|}(\theta_2 - \theta_0)$ denotes the normalized optimization direction on auxiliary dataset, $\|\cdot\|$ denotes the Euclidean norm. In our setting, we select

mBART finetuned on WMT17 En-Zh data as the second axis to plot the loss landscape on WMT19 En-De benchmark.

Figure 1(a) and Figure 1(b) present the loss landscape with contour line of model training with PT and RI, where the left-side and right-side axis are corresponding to α and β in Equation 1 respectively. We also mark the start point and the end point of training path. The start point denotes the loss of model without training, while the end point indicates the converged model loss. Firstly, we can see that PT obtains flatter loss landscape than RI with contour line covers a wider area, which denotes that initialized with pre-trained parameters leads model to be more robust. This supports us to understand the generalization of PT from the perspective of the flatter optima. Secondly, the start point of PT is much smaller than RI (14.13 vs. 62.17), which demonstrates that the knowledge learned during pre-training stage is also suitable for the downstream task. We also observe that RI achieves the lowest loss at the end point (3.18). This shows that the RI model is well optimized on the in-domain data, as the large amount sentence pairs provide enough supervised signal.

3.2 Impact on Lexical Prob. Distribution

Previous work (Jiang et al., 2019) shows that the lexical probability distribution is correlated with the diversity of generation, i.e., over-confidence in certain tokens leads to lower diversity (Miao et al., 2021). This finding inspires us to measure the density of PT and RI outputs. We use Shannon entropy of decoder output according to test set to probe this attribute. The smaller entropy represents less assigned probability density on incorrect tokens.

As shown in Figure 2, we present how much percentage of character is involved in different intervals. We firstly see that most tokens prefer a relatively smaller entropy, which denotes that both models are well optimized, and predict tokens with higher confidence. Besides, PT achieves a smoother distribution than RI with most outputs achieving relative higher entropy. This suggests that PT could generate more diversiform translations, which is consistent with the report from Wang et al. (2022a).

4 Harmonizing PT and RI

Based on the above experiments, we prove that PT mainly contributes to the generalization ability and RI leads to better in-domain performance. To combine the advantage of both models, we fuse both models via optimal transport based on activations following Singh and Jaggi (2020).

4.1 Model Fusion

Discrete Optimal Transport Given two discrete measures $\sum_{i=1}^{n} \alpha_i \delta_{\alpha_i}$ and $\sum_{i=1}^{n} \beta_i \delta_{\beta_i}$, where δ_{β_i} denotes the Dirac measures on β_i, and $\sum_{i=1}^{n} \alpha_i = 1$ and $\sum_{i=1}^{m} \beta_i = 1$. Then, with the cost matrix $\mathbf{C} \in \mathbb{R}^{n \times m}$, which computed with predefined cost function, the discrete optimal transport formulates:

$$\mathrm{OT} := \min_{\mathbf{T} \in \Pi(\boldsymbol{\alpha}, \boldsymbol{\beta})} \langle \mathbf{T}, \mathbf{C} \rangle_F \tag{2}$$

where $\langle \cdot, \cdot \rangle_F$ is the Frobenius product, and $\Pi(\boldsymbol{\alpha}, \boldsymbol{\beta}) = \{\mathbf{T} \in \mathbb{R}^{n \times m} : \mathbf{T}\mathbb{1}_m = \boldsymbol{\alpha} \text{ and } \mathbf{T}^\top \mathbb{1}_n = \boldsymbol{\beta}\}$ is the set of all admissible transport matrix between the two measures ($\mathbb{1}_n$ is a vector of ones of size n).

Model Fusion For layers ℓ_{ri} and ℓ_{pt} of RI and PT models respectively, we define the activation based probability measures over neurons as $\mu^{\ell_{ri}} = \left(\alpha^{\ell_{ri}}, \delta^{\ell_{ri}}\right)$ and $\nu^{\ell_{pt}} = \left(\beta^{\ell_{pt}}, \delta^{\ell_{pt}}\right)$, where $\delta^{\ell_{ri}}$, $\delta^{\ell_{pt}}$ are activations for given samples. And, we set uniform distribution for α and β.

In term of the alignment procedure, we first align the weights of previous layer $\mathbf{W}_{ri}^{\ell, \ell-1}$ by constructing convex combination with the weights of corresponding transport matrix $\mathbf{T}^{\ell-1}$, normalized appropriately via the inverse of corresponding column marginals α. After getting the aligned $\widehat{\mathbf{W}}_{ri}^{\ell, \ell-1}$, we compute the optimal transport map $\mathbf{T}^{\ell}$ between $\mu^{(\ell_{ri})}$ and $\nu^{(\ell_{pt})}$, which is formulated as Equation 2. And now, we align ℓ_{ri} with respect to ℓ_{pt},

$$\widetilde{\mathbf{W}}_{ri}^{(\ell, \ell-1)} \leftarrow \mathrm{diag}\left(\frac{1}{\vec{\beta}^{(\ell)}}\right) {\mathbf{T}^{(\ell)}}^\top \widehat{\mathbf{W}}_{ri}^{(\ell, \ell-1)}. \tag{3}$$

We will refer $\widetilde{\mathbf{W}}_{ri}^{(\ell, \ell-1)}$ to the aligned weights of RI model, which we can directly add with $\mathbf{W}_{pt}^{(\ell, \ell-1)}$. We carry out this procedure over all layers sequentially. Then, we fine-tune the fused model on corresponding dataset to obtain the final model.

4.2 Results

As detailed in Section 2, we conduct the experiments on two widely used benchmarks WMT17 En-Zh and WMT19 En-De. To obtain the fused model, we set weights of 0.9 for PT and 0.1 for RI during model fusion and then fine-tune on the corresponding dataset.

Model	En-De	En-Zh
RI	40.1	32.9
PT	39.3	32.3
+Fusion	$39.7^{\Uparrow+0.4}$	$32.4^{\Uparrow+0.1}$

Table 1: In-Domain translation quality of model trained on the WMT19 En-De and WMT17 En-Zh benchmarks. We evaluate models on official test sets. PT denotes the fine-tuned mBART, while RI indicts model trained from scratch.

	PT	RI	Fusion	Δ
1) OD	34.9	35.0	35.2	**+0.2**
2) DuA	-12.1	-16.7	-12.3	**+4.4**
3) Multi-Ref	75.4	75.8	76.0	**+0.2**
4) TTR	19.1	19.1	19.2	**+0.1**

Table 2: Results of analysis experiments. "OD" (Out-Domain) evaluates the performance on the out-domain test set. "DuA" (Drop under Attack) reports the drop of BLEU when adding noise into source sentences. "Multi-Ref" is the BLEU score with multiple references. "TTR" denotes the Type-Token-Ratio. "Δ" indicts difference between RI and Fusion.

As present in Figure 1(c), we present the loss landscape of fused model. The start point is the fused model before fine-tuning. We can see that our model inherits the smooth loss landscape from PT and leads to smaller loss value.

The in-domain translation results are shown in Table 1. We present the results of PT and RI, which has the same network architecture, for comparison with our model. As seen, RI performs better than PT for two benchmarks (39.3 vs. 39.7 for En-De, 32.3 vs. 32.4 for En-De). This empirically verifies the claim we make in Section 3.1. Besides, our fused model achieves a better performance than PT (39.3 vs. 39.7 for En-De, 32.3 vs. 32.4 for En-De), which denotes that model fusion could promote the performance of PT. while still under-performs RI (40.1 vs. 39.7 for En-De, 32.9 vs. 32.4 for En-De). This may draw from that our approach can be seen as a trade-off of PT and RI.

4.3 Analysis

We conducted several analyses to understand whether it inherits the advantages of PT. We evaluate our approach on four tasks, compared with PT model and RI model, including model generalization, performance drop under attack, translation diversity and type-token-ratio. Note that we can measure more properties, e.g. copy errors (Liu et al., 2021b) and encoded linguistic knowledge (Hao et al., 2019a; Ding et al., 2020), but our aim is simply to show that PT and RI can be harmonized. All results are reported on the WMT19 En-De benchmark. For simplification, we list scores in one table.

1) Effects of Model Generalization To understand how our model performs on the out-domain data. We evaluate trained model on two different domains, including It and Medical, and report Out-Domain Score by averaging SacreBLEU scores. As it presents in the first line of Table 2, the fused model achieves $+0.2$ score compared with RI. This verifies the generalization ability of our model.

2) Performance Drop under Attack To further demonstrate that our fused model could be more robust than RI. We attack the model by adding noises into source sentence during the inference stage and report the drop in score. Specifically, we replace 10% words with a special token. The less drop mean model is more robust. As seen in the third line of Table 2, both models suffer a huge drop with the perturbed input. However, our model still performs better than RI (-29.6 vs. -31.2). This result shows that our model is less sensitive to noises, which leads to more robustness.

3) Effects of Translation Diversity Following Wang et al. (2022a), we evaluate the translation diversity of models with multi-reference SacreBLEU. We inference on the test set released by Ott et al. (2018), which consists of ten additional reference translations for 500 English sentences extracted from WMT14 En-De test set. As shown in the second line of Table 2, our model achieves a higher multi-reference SacreBLEU score than RI. This empirically proves that our fused model could generate a more diverse translation with different word orders.

4) Effects of TTR We also probe diversity of the lexical with **TTR** = $\frac{num.\ of\ types}{num.\ of\ tokens}$ following Wei et al. (2022). The higher score of **TTR** inflects that the model tends to generate translation with different word types. As it present in the last line of Table 2, our fusion model achieves +0.16 gains over RI. This shows that our model could not only generate translation with more word orders but also predict words of more types.

5 Conclusion and Future Work

This paper provides several insights for pre-training (PT) and random-initialization (RI) on resource-rich machine translation. We carefully investigate the impact of PT and RI on loss landscape and negative diversity and reveal that RI leads to a lower loss value of in-domain data, while PT is more beneficial to flatter loss and smoother lexical distribution. We also propose to combine them with model fusion via optimal transport. Experimental results on WMT17 En-Zh and WMT19 En-De benchmarks show that model fusion is a nice trade-off method to utilize the complementarity of PT and RI.

For future work, we would explore other effective fusion algorithms (Liu et al., 2022; Wang et al., 2022b) to combine the merits of PT and RI. Also, it will be interesting to investigate whether harmonizing PT and RI could complement existing data augmentation methods for NMT, e.g. back translation (Liu et al., 2021a), data rejuvenation (Jiao et al., 2020), bidirectional distillation (Ding et al., 2021a) and training (Ding et al., 2021b).

Acknowledgements

We are grateful to the anonymous reviewers and the area chair for their insightful comments and suggestions. Changtong Zan and Weifeng Liu were supported by the Major Scientific and Technological Projects of China National Petroleum Corporation (CNPC) under Grant ZD2019-183-008, and in part by the National Natural Science Foundation of China under Grant 61671480.

References

Farhad Akhbardeh, Arkady Arkhangorodsky, Magdalena Biesialska, Ondej Bojar, Rajen Chatterjee, Vishrav Chaudhary, Marta R. Costa-jussa, Cristina Espaa-Bonet, Angela Fan, Christian Federmann, Markus Freitag, Yvette Graham, Roman Grundkiewicz, Barry Haddow, Leonie Harter, Kenneth Heafield, Christopher Homan, Matthias Huck, Kwabena Amponsah-Kaakyire, Jungo Kasai, Daniel Khashabi, Kevin Knight, Tom Kocmi, Philipp Koehn, Nicholas Lourie, Christof Monz, Makoto Morishita, Masaaki Nagata, Ajay Nagesh, Toshiaki Nakazawa, Matteo Negri, Santanu Pal, Allahsera Auguste Tapo, Marco Turchi, Valentin Vydrin, and Marcos Zampieri. 2021. Findings of the 2021 conference on machine translation (wmt21). In *WMT*.

Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. 2015. Neural machine translation by jointly learning to align and translate. In *ICLR*.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *NAACL*.

Liang Ding, Longyue Wang, Xuebo Liu, Derek F. Wong, Dacheng Tao, and Zhaopeng Tu. 2021a. Rejuvenating low-frequency words: Making the most of parallel data in non-autoregressive translation. In *ACL*.

Liang Ding, Longyue Wang, Di Wu, Dacheng Tao, and Zhaopeng Tu. 2020. Context-aware cross-attention for non-autoregressive translation. In *COLING*.

Liang Ding, Di Wu, and Dacheng Tao. 2021b. Improving neural machine translation by bidirectional training. In *EMNLP*.

Jonas Gehring, Michael Auli, David Grangier, Denis Yarats, and Yann N. Dauphin. 2017. Convolutional sequence to sequence learning. In *ICML*.

Jie Hao, Xing Wang, Baosong Yang, Longyue Wang, Jinfeng Zhang, and Zhaopeng Tu. 2019a. Modeling recurrence for transformer. In *NAACL*.

Yaru Hao, Li Dong, Furu Wei, and Ke Xu. 2019b. Visualizing and understanding the effectiveness of BERT. In *EMNLP*.

Shaojie Jiang, Pengjie Ren, Christof Monz, and Maarten de Rijke. 2019. Improving neural response diversity with frequency-aware cross-entropy loss. In *The Web Conference*.

Wenxiang Jiao, Xing Wang, Shilin He, Irwin King, Michael Lyu, and Zhaopeng Tu. 2020. Data rejuvenation: Exploiting inactive training examples for neural machine translation. In *EMNLP*.

Taku Kudo and John Richardson. 2018. SentencePiece: A simple and language independent subword tokenizer and detokenizer for neural text processing. In *EMNLP*.

Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Veselin Stoyanov, and Luke Zettlemoyer. 2020. BART: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. In *ACL*.

Zuchao Li, Rui Wang, Kehai Chen, Masso Utiyama, Eiichiro Sumita, Zhuosheng Zhang, and Hai Zhao. 2020. Data-dependent gaussian prior objective for language generation. In *ICLR*.

Chang Liu, Chenfei Lou, Runzhong Wang, Alan Yuhan Xi, Li Shen, and Junchi Yan. 2022. Deep neural network fusion via graph matching with applications to model ensemble and federated learning. In *ICML*.

Xuebo Liu, Longyue Wang, Derek F Wong, Liang Ding, Lidia S Chao, Shuming Shi, and Zhaopeng Tu. 2021a. On the complementarity between pre-training and back-translation for neural machine translation. In *Findings of EMNLP*.

Xuebo Liu, Longyue Wang, Derek F. Wong, Liang Ding, Lidia S. Chao, Shuming Shi, and Zhaopeng Tu. 2021b. On the copying behaviors of pre-training for neural machine translation. In *Findings of ACL*.

Yinhan Liu, Jiatao Gu, Naman Goyal, Xian Li, Sergey Edunov, Marjan Ghazvininejad, Mike Lewis, and Luke Zettlemoyer. 2020. Multilingual denoising pre-training for neural machine translation. *TACL*.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Roberta: A robustly optimized bert pretraining approach. *arXiv*.

Mengqi Miao, Fandong Meng, Yijin Liu, Xiao Zhou, and Jie Zhou. 2021. Prevent the language model from being overconfident in neural machine translation. In *ACL*.

Gaspard Monge. 1781. Mémoire sur la théorie des déblais et des remblais. *Histoire de l'Académie Royale des Sciences de Paris*.

Myle Ott, Michael Auli, David Grangier, and Marc'Aurelio Ranzato. 2018. Analyzing uncertainty in neural machine translation. In *ICML*.

Matt Post. 2018. A call for clarity in reporting BLEU scores. In *WMT*.

Sidak Pal Singh and Martin Jaggi. 2020. Model fusion via optimal transport. In *NeurIPS*.

Kaitao Song, Xu Tan, Tao Qin, Jianfeng Lu, and Tie-Yan Liu. 2019. MASS: masked sequence to sequence pre-training for language generation. In *ICML*.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *NeurIPS*.

Wenxuan Wang, Wenxiang Jiao, Yongchang Hao, Xing Wang, Shuming Shi, Zhaopeng Tu, and Michael Lyu. 2022a. Understanding and improving sequence-to-sequence pretraining for neural machine translation. In *ACL*.

Zhenyi Wang, Xiaoyang Wang, Li Shen, Qiuling Suo, Kaiqiang Song, Dong Yu, Yan Shen, and Mingchen Gao. 2022b. Meta-learning without data via wasserstein distributionally-robust model fusion. In *UAI*.

Xiangpeng Wei, Heng Yu, Yue Hu, Rongxiang Weng, Weihua Luo, and Rong Jin. 2022. Learning to generalize to more: Continuous semantic augmentation for neural machine translation. In *ACL*.

Jinhua Zhu, Yingce Xia, Lijun Wu, Di He, Tao Qin, Wengang Zhou, Houqiang Li, and Tie-Yan Liu. 2020. Incorporating BERT into neural machine translation. In *ICLR*.

ngram-OAXE : Phrase-Based Order-Agnostic Cross Entropy for Non-Autoregressive Machine Translation

Cunxiao Du
Singapore Management University
cnsdunm@gmail.com

Zhaopeng Tu*
Tencent AI Lab
zptu@tencent.com

Longyue Wang
Tencent AI Lab
vinnylywang@tencent.com

Jing Jiang
Singapore Management University
jingjiang@smu.edu.sg

Abstract

Recently, a new training OAXE loss (Du et al., 2021) has proven effective to ameliorate the effect of multimodality for non-autoregressive translation (NAT), which removes the penalty of word order errors in the standard cross-entropy loss. Starting from the intuition that reordering generally occurs between phrases, we extend OAXE by only allowing reordering between ngram phrases and still requiring a strict match of word order within the phrases. Extensive experiments on NAT benchmarks across language pairs and data scales demonstrate the effectiveness and universality of our approach. Further analyses show that *ngram*-OAXE indeed improves the translation of ngram phrases, and produces more fluent translation with a better modeling of sentence structure.[1]

1 Introduction

Fully non-autoregressive translation (NAT) has received increasing attention for its efficient decoding by predicting every target token in parallel (Gu et al., 2018; Ghazvininejad et al., 2019). However, such advantage comes at the cost of sacrificing translation quality due to the *multimodality* problem: there exist many possible translations of the same sentence, while vanilla NAT models may consider them at the same time due to the independent predictions, which leads to multi-modal outputs in the form of token repetitions (Gu et al., 2018).

Recent works have incorporated approaches to improving the standard cross-entropy (XE) loss to ameliorate the effect of multimodality. The motivation for these works is that modeling word order is difficult for NAT, since the model cannot condition on its previous predictions like its autoregressive counterpart. Starting from this intuition, a thread of research relaxes the word order restriction based on the **monotonic alignment** assumption (Libovický and Helcl, 2018; Ghazvininejad et al., 2020; Saharia et al., 2020). Du et al. (2021) take a further step by removing the penalty of word order errors with a novel order-agnostic cross entropy (OAXE) loss, which enables NAT models to handle **word reordering** – a common source of multimodality problem. Accordingly, OAXE achieves the best performance among these model variants.

However, OAXE allows reordering between every two words, which is not always valid in practice. For example, the reordering of the two words "this afternoon" is not correct in grammar. The reordering generally occurs between ngram phrases, such as "I ate pizza" and "this afternoon". Starting from this intuition, we extend OAXE by constraining the reordering between ngrams and requiring a strict match of word order within each ngram (i.e., *ngram*-OAXE). To this end, we first build the probability distributions of ngrams in the target sentence using the word probabilities produced by NAT models. Then we find the best ordering of target ngrams to minimize the cross entropy loss. We implement the *ngram*-OAXE loss in an efficient way, which only adds one more line of code on top of the source code of OAXE. Accordingly, *ngram*-OAXE only marginally increases training time (e.g., 3% more time) over OAXE.

Experimental results on widely-used NAT benchmarks show that *ngram*-OAXE improves translation performance over OAXE in all cases. Encouragingly, *ngram*-OAXE outperforms OAXE by up to +3.8 BLEU points on raw data (without knowledge distillation) for WMT14 En-De translation (Table 1), and narrows the performance gap between training on raw data and on distilled data. Further analyses show that *ngram*-OAXE improves over OAXE on the generation accuracy of ngram phrases and modeling reordering between ngram phrases, which makes *ngram*-OAXE handle long sentences better, especially on raw data. The

*Zhaopeng Tu is the corresponding author.

[1]The codes and models are in https://github.com/tencent-ailab/machine-translation/COLING22_ngram-OAXE/.

Proceedings of the 29th International Conference on Computational Linguistics, pages 5035–5045
October 12–17, 2022.

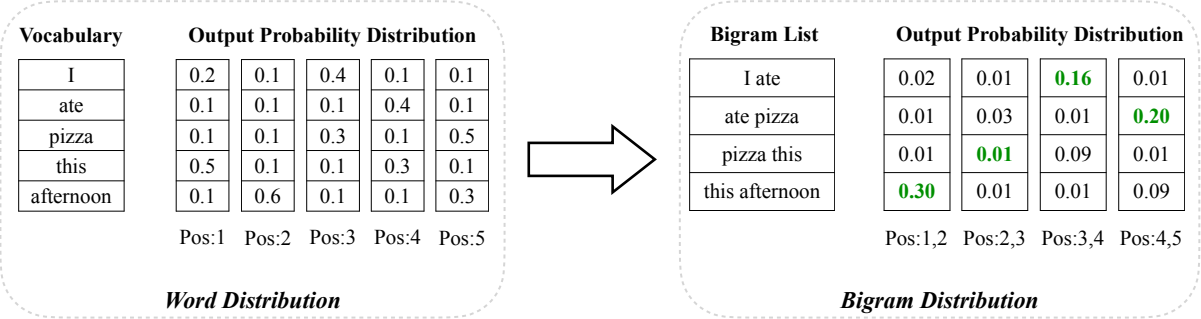

Figure 1: Illustration of the proposed *ngram*-OAXE loss with $N = 2$ (i.e., bigram-OAXE). We only show the probabilities of the target words and bigrams for better illustration. Firstly, *ngram*-OAXE transforms the word probability distributions to the bigram distributions by multiplying the word probabilities at the corresponding positions. For example, P("I ate" | Pos:1,2) = P("I" | Pos:1) * P("ate" | Pos:2) = 0.2*0.1=0.02. Then, we select the ngrams (highlighted in bold) for each neighbouring positions using the efficient Hungarian algorithm.

strength of *ngram*-OAXE on directly learning from the complex raw data indicates the potential to train NAT models without knowledge distillation.

2 Methodology

2.1 Preliminaries: NAT

Cross Entropy (XE) Standard NAT models (Gu et al., 2018) are trained with the cross entropy loss:

$$\mathcal{L}_{XE} = -\log P(Y|X) = -\sum_{y_n} \log P(y_i|X), \quad (1)$$

where (X, Y) with $Y = \{y_1, \ldots, y_I\}$ is a bilingual training example, and $P(y_i|X)$ is calculated independently by the NAT model. XE requires a strict match of word order between target tokens and model predictions, thus will heavily penalize hypotheses that are semantically equivalent to the target but different in word order.

Order-Agnostic Cross Entropy (OAXE) Du et al. (2021) remove the word order restriction of XE, and assign loss based on the best alignment between target tokens and model predictions. They define the ordering space $\mathbf{O} = \{O^1, \ldots, O^J\}$ for Y, where O^j is an ordering of the set of target tokens $(y_1, \ldots, y_I)$. The OAXE objective is defined as finding the best ordering O^j to minimize the cross entropy loss:

$$\mathcal{L}_{\text{OAXE}} = \min_{O^j \in \mathbf{O}} \left(-\log P(O^j|X)\right), \quad (2)$$

where $-\log P(O^i|X)$ is the cross entropy loss for ordering O^i, which is calculated by Equation 1.

2.2 *ngram*-OAXE Loss

Figure 1 illustrates the two-phase calculation of *ngram*-OAXE : 1) constructing the probability distributions of the ngrams in the target sentence; 2) searching the best ordering of the considered ngrams to minimize the cross entropy loss.

Formulation Given the target $Y = \{y_1, \ldots, y_I\}$, we define the target ngrams G^N of size N as all the N continuous tokens in Y: $\{y_{1:N}, \cdots, y_{I-N+1:I}\}$. The output ngram distributions P_G is defined as:

$$P_G(y_{i:i+N-1}|X) = \prod_{t=i}^{i+N-1} P(y_t|X), \quad (3)$$

where $P(y_t|X)$ is the prediction probability of NAT models for the token y_t in position t of the target sentence, and N is the size of ngrams.

The *ngram*-OAXE objective is defined as finding the best ordering O^j to minimize the cross entropy loss of the considered ngrams in target sentence Y:

$$\mathcal{L}_{\textit{ngram}\text{-OAXE}} = \min_{O^j \in \mathbf{O}} \left(-\log P_G(O^j|X)\right). \quad (4)$$

Ideally, the best ordering O^j should meet the following conditions:

1. The ngrams in O^j should not be overlapped (e.g., "I ate" and "ate pizza" should not occur simultaneously in one O).

2. O^j is a mixture of ngrams with different sizes (e.g., "I ate pizza" and "this afternoon").

However, it is computationally infeasible to search the best ngram segmentation of the target sentence with highest probabilities. Given a target sentence

with length I, there are 2^I ngram segmentation (i.e, each token can be labeled as the end of a ngram or not). For each ngram segmentation with expected length I/2, the time complexity is $O((I/2)^3)$ using the efficient Hungarian algorithm. In this way, the total computational complexity of the original two conditions is $O(2^I I^3)$.

For computational tractability, we loosen the conditions by:

1. We consider all ngrams in the target sentence to avoid searching the ngram segmentation. In other words, each word is allowed to occur in multiple ngrams in one ordering O.

2. We only consider ngrams with a fixed size N (e.g., only bigrams), which enables us to cast this problem as Maximum Bipartite Matching and leverage the efficient Hungarian algorithm, as done in (Du et al., 2021).

By loosening the conditions, there are (I-N+1) ngrams of size N in the sentence, and the computational complexity is $O(I^3)$. Accordingly, the loss of the ordering O^j is computed as:

$$P_G(O^j|X) = \prod_{y_{i:i+N-1} \in O^j} P_G(y_{i:i+N-1}|X). \quad (5)$$

Figure 1 shows the calculation of bigram-OAXE loss for the target sentence "I ate pizza this afternoon". We consider all bigrams in the sentence (see "Bigram List"), and obtain the probability distribution of the considered bigrams. We construct the bipartite graph $G = (U, V, E)$ where the first part of vertices U is the set of N-1 neighbouring positions (e.g., the first two positions"Pos:1,2"), and the second part of vertices V is the list of N-1 target bigrams. Each edge in E is the prediction log probability for the bigram in the corresponding position. We can follow Du et al. (2021) to leverage the efficient Hungarian algorithm (Kuhn, 1955) for fast calculation of *ngram*-OAXE (see the assigned probabilities for the consider bigrams).

Implementation Algorithm 1 shows the pseudo-code of *ngram*-OAXE with $N = 2$. The implementation of *ngram*-OAXE is almost the same with that of OAXE, except that we add one more line (in red color) for constructing the probability distribution of ngrams. We implement *ngram*-OAXE on top of the source code of OAXE, and leverage the same recipes (i.e., loss truncation and XE pretrain) to effectively restrict the free-order nature of OAXE.

Algorithm 1 Bigram-OAXE Loss

Input: Ground truth Y, NAT output $\log P$
bs, len = Y.size()
Y = Y.repeat(1, len).view(bs, len, len)
$costM$ = -log P.gather(index=Y, dim=2)
$costM = costM$[:, :-1, :-1] +$costM$[:, 1:, 1:]
for $i = 0$ **to** bs **do**
  $bestMatch$[i] = HungarianMatch($costM$[i])
end for
Return:$costM$.gather(index=$bestMatch$)

Since both *ngram*-OAXE and OAXE only modify the training of NAT models, their inference latency is the same with the CMLM baseline (e.g., 15.3x speed up over the AT model). Concerning the training latency, OAXE takes 36% more training time over the CMLM baseline, and our *ngram*-OAXE takes 40% more training time, which is almost the same to OAXE since we only add one more line of code.

Discussion Some researchers may doubt that the *ngram*-OAXE loss is not an intuitively understandable "global" loss, since some words are counted multiple times. We use the example in Figure 1 to dispel the doubt. Firstly, except for the first and last words (i.e., "I" and "afternoon"), the *ngram*-OAXE loss equally counts the other words twice, which would not introduce the count bias.

Secondly, we follow Du et al. (2021) to start with an initialization pre-trained with the XE loss, which ensures that the NAT models can produce reliable token probabilities to compute ngram probabilities. We also use the **loss truncation** technique (Kang and Hashimoto, 2020) to drop invalid ngrams with low probabilities (e.g., "pizza this" | Pos:2,3) in the selected ordering O^j.

Thirdly, the overlapped ngrams can help to produce more fluent translations by modeling global context in a manner of ngram LM. For example, the high-probability overlapped token in position 4 "ate" (i.e., P(ate | Pos:4) = 0.4) will guide NAT models to assign high probabilities to the neighbouring ngrams ("I ate" | Pos:3,4) and ("ate pizza" | Pos:4,5), which form a consistent clause ("I ate pizza | Pos:3,4,5"). In contrast, *ngram*-OAXE would not simultaneously assign high probabilities to the phrases ("this afternoon" | Pos:1,2) and ("pizza this" | Pos:2,3), since the two phrases require NAT models to assign high probabilities to two different words (i.e., "afternoon" and "pizza")

in the overlapped position 2.

The $\mathcal{L}_{ngram\text{-OAXE}}$ loss with $N = 2$ in Figure 1 is calculated as:

$$\log P(\text{“}this\ afternoon\text{”}|Pos:1,2) + \log P(\text{“}I\ ate\text{”}|Pos:3,4) + \log P(\text{“}ate\ pizza\text{”}|Pos:4,5)$$

where the low-probability bigram (“pizza this” | Pos:2,3) is truncated. In this way, *ngram*-OAXE carries out operation at the ngram granularity: *ngram*-OAXE requires exact match of the word order within the ngram phrases, and allows reordering between phrases (e.g., “I ate pizza | Pos:3,4,5” and “this afternoon | Pos:1,2").

3 Experiment

3.1 Experimental Setup

Data We conducted experiments on major benchmarking datasets that are widely-used in previous NAT studies (Ma et al., 2019; Saharia et al., 2020): WMT14 English⇔German (En⇔De, 4.5M sentence pairs) and WMT17 English⇔Chinese (En⇔Zh, 20.0M sentence pairs). We preprocessed the datasets with a joint BPE (Sennrich et al., 2016) with 32K merge operations for the En⇔De and En⇔Zh datasets. For fair comparison with prior work, we reported the Sacre BLEU (Post, 2018)[2] on the En-Zh task, and the compound BLEU (Papineni et al., 2002) on the other tasks.

Knowledge Distillation We closely followed previous works on NAT to apply sequence-level knowledge distillation (Kim and Rush, 2016) to reduce the modes of the training data. Specifically, we obtained **distilled data** by replacing the target side of the original training data (i.e., **raw data**) with translation produced by an external AT teacher. Consistent with previous works (Ghazvininejad et al., 2019, 2020; Du et al., 2021), we employed Transformer-BIG (Vaswani et al., 2017) as the AT teacher for knowledge distillation.

NAT Models We validated our approach on the representative NAT model – CMLM (Ghazvininejad et al., 2019), which uses the conditional mask LM (Devlin et al., 2019) to generate the target sequence from the masked input. The NAT model shares the same architecture as Transformer-BASE (Wang and Tu, 2020): 6 layers for both the encoder and decoder, 8 attention heads, 512 model dimensions. We chose the CMLM models with the vanilla XE loss (Ghazvininejad et al., 2019) and the OAXE loss (Du et al., 2021) as our two main baselines. To keep consistent with main baselines, we set 5 as length candidates for all CMLM models during inference.

We generally followed the hyperparameters used in (Ghazvininejad et al., 2019). We trained batches of approximately 128K tokens using Adam (Kingma and Ba, 2015). The learning rate warmed up to 5×10^{-4} in the first 10K steps, and then decayed with the inverse square-root schedule. We trained all models for 300k steps, measured the validation BLEU at the end of each epoch, and averaged the 5 best checkpoints. We followed (Li et al., 2019; Sun and Yang, 2020; Saharia et al., 2020; Du et al., 2021) to use de-duplication trick to remove repetitive tokens in the generated output.

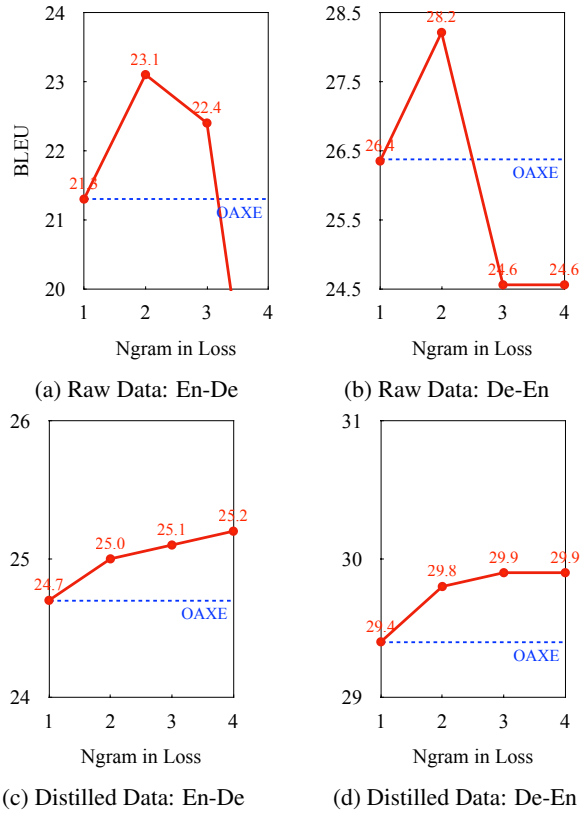

(a) Raw Data: En-De (b) Raw Data: De-En

(c) Distilled Data: En-De (d) Distilled Data: De-En

Figure 2: Impact of N-gram choice in *ngram*-OAXE loss (i.e., N in Equation 3). OAXE can be viewed as a special case of *ngram*-OAXE with $N = 1$.

3.2 Ablation Study

In this section, we investigated the impact of different components for *ngram*-OAXE on the WMT14 En⇔De validation sets.

[2]SacreBLEU hash : BLEU+case.mixed+lang.en-zh + numrefs.1+smooth.exp+test.wmt17+tok.zh+version.1.4.2

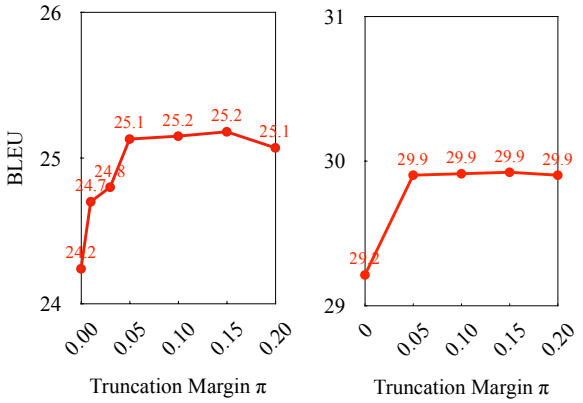

(a) Distilled Data: En-De (b) Distilled Data: De-En

Figure 3: Impact of the truncation margin π for NAT models trained on the distilled data.

Impact of Ngram Size We first investigated the impact of different N in the *ngram*-OAXE loss on the translation performance. Figure 2 shows the results for both raw data and distilled data. As seen, the bigram-OAXE achieves the best performance on raw data, while 4gram-OAXE performs best on the distilled data. We attribute the different behaviors to the difficulty of the dataset: raw data contains more modes than distilled data (Gu et al., 2018), thus it is more difficult to learn larger ngrams from the complicated raw data. In the following experiments, we set $N = 2$ for raw data, and $N = 4$ for distilled data.

Impact of Truncation Margin Figure 3 shows the impact of truncation margin π, which is searched from $\{0, 0.05, 0.10, 0.15, 0.20\}$. Intuitively, higher π drops more ngrams. When π increases from 0 to 0.05, we achieved 0.7∼0.9 BLEU improvement by dropping likely invalid ngrams. *ngram*-OAXE is robust to the truncation margin π: when π further increases, the performance does not vary too much. We follow Du et al. (2021) to use $\pi = 0.15$ for all language pairs and datasets in the following experiments.

3.3 Translation Performance

In this section, we conduct comprehensive experiments to validate the effectiveness of the proposed *ngram*-OAXE model. First, we use multiple semantically equivalent references to better evaluate the multimodality nature of generated translation, which serves as the main results for analyses in the following sections. Then we compare our approach with previous work on the benchmarking testsets with single reference.

Model	W14 En-De		NIST Zh-En	
	BLEU	Δ	BLEU	Δ
Raw Data				
Transformer	71.4	-	41.7	-
CMLM	28.1	-	12.1	-
+OAXE	57.5	+29.4	36.5	+24.4
+*ngram*-OAXE	61.3$^{\uparrow\Uparrow}$	+33.2	38.6$^{\uparrow\Uparrow}$	+26.5
Distilled Data				
Transformer	72.7	-	42.0	-
CMLM	50.7	-	23.7	-
+OAXE	68.0	+17.3	40.4	+16.7
+*ngram*-OAXE	68.9 $^{\uparrow\Uparrow}$	+18.2	41.2 $^{\uparrow\Uparrow}$	+17.5

Table 1: BLEU scores on test sets with multiple references. "Δ" denotes the improvement over CMLM. "↑" and "⇑" denotes significantly better than CMLM and OAXE with $p < 0.05$, respectively. The Zh-En NMT model is trained on the WMT17 Zh-En data.

Multiple References We follow Du et al. (2021) to use two test sets with multiple references: 1) the dataset released by Ott et al. (2018) that consists of ten human translations for 500 sentences taken from the WMT14 En-De test set; and 2) the combination of NIST02-08 Zh-En test sets that consists of 7497 sentences with four references. The translation models are trained on the WMT14 En-De and WMT17 Zh-En training data, respectively.

Table 1 lists the translation performance. Encouragingly, *ngram*-OAXE narrows the performance gaps between:

- *NAT models trained on raw data and on distilled data*: Take W14 En-De as an example, knowledge distillation brings an improvement of 22.6 BLEU points over raw data for XE (i.e., from 28.1 to 50.7). OAXE narrows the gap to 10.5 BLEU points (i.e., 57.5 vs. 68.0), and our *ngram*-OAXE further narrows the gap to 7.6 BLEU points (i.e., 61.3 vs. 68.9), moving toward training NAT models without distillation.

- *NAT and AT models trained on raw data*: For independent NAT models without distillation, OAXE reduces the performance gap from 43.3 to 13.9 for En-De, and from 29.6 BLEU to 5.2 BLEU for Zh-En. *Ngram*-OAXE further reduces the gaps to 10.1 and 3.1 BLEU points, indicating the potential of NAT to become a practical system without relying on external resources.

Model	WMT14	
	En-De	De-En
Autoregressive Transformer	27.6	31.4
Non-Autoregressive		
CTC Loss (Libovický and Helcl, 2018)	17.7	19.8
Flowseq (Ma et al., 2019)	18.6	23.4
Imputer (Saharia et al., 2020)	15.6	-
CMLM (Ghazvininejad et al., 2019)	10.6	15.1
+AXE (Ghazvininejad et al., 2020)	20.4	24.9
+Correction (Huang et al., 2022)	20.6	25.4
+OAXE (Du et al., 2021)	22.4	26.8
+*ngram*-OAXE (*Ours*)	**23.6**$^{\uparrow\Uparrow}$	**27.9**$^{\uparrow\Uparrow}$

Table 2: BLEU scores on testsets with single reference for NAT models trained on the **raw data**.

Model	WMT14		WMT17	
	En-De	De-En	En-Zh	Zh-En
Autoregressive Transformer	27.8	31.3	34.4	24.0
Non-Autoregressive				
Bag-of-ngrams (Shao et al., 2020)	20.9	24.6	-	-
Flowseq (Ma et al., 2019)	21.5	26.2	-	-
Bigram CRF (Sun et al., 2019)	23.4	27.2	-	-
Imputer (Saharia et al., 2020)	25.8	28.4	-	-
CMLM (Ghazvininejad et al., 2019)	18.1	21.8	24.2	13.6
+AXE (Ghazvininejad et al., 2020)	23.5	27.9	30.9	19.8
+GLAT (Qian et al., 2021)	25.2	29.8	-	-
+CTC+VAE (Gu and Kong, 2021)	27.5	31.1	-	-
+OAXE (Du et al., 2021)	26.1	30.2	32.9	22.1
+*ngram*-OAXE (*Ours*)	**26.5**$^{\uparrow\Uparrow}$	**30.5**$^{\uparrow\Uparrow}$	**33.2**$^{\uparrow\Uparrow}$	**22.8**$^{\uparrow\Uparrow}$

Table 3: BLEU scores on testsets with single reference for NAT models trained on the **distilled data**.

Benchmarks with Single Reference We also evaluated the performance of fully NAT models on benchmarks with single reference. In addition to the closely related XE variants (e.g., AXE and OAXE), we also compare against several strong baseline models: 1) CTC Loss – a NAT model with latent alignments (Libovický and Helcl, 2018); 2) Flowseq – a latent variable model based on generative flow (Ma et al., 2019); 3) Imputer – an extension of CTC with the use of distillation during training (Saharia et al., 2020); 4) Corretion – a NAT model with error correction mechanism (Huang et al., 2022); 5) Bigram CRF – the CRF-based semi-autoregressive model (Sun et al., 2019); 6) GLAT – Glancing-based training (Qian et al., 2021); 7) CTC+VAE – combining the CTC loss and latent variables (VAE) (Gu and Kong, 2021).

Table 2 lists the results on the **raw data** that do not rely on any external resources (e.g., AT models for KD). CMLMs trained by *ngram*-OAXE improves over the XE-trained baseline by 12.8 BLEU points on average, and outperforms the strong OAXE by +1.0 BLEU points. These results indicate that the ngram supervision helps to better capture the complicated patterns from the raw data.

Table 3 lists the BLEU scores on the **distilled data**. Our approach consistently improves over the strong OAXE loss in all cases, demonstrating the effectiveness and universality of the proposed *ngram*-OAXE loss. Our approach also outperforms all existing NAT using a single technique (exclude Gu and Kong (2021) with two techniques).

Model	Ngram Size			
	1	2	3	4
Raw Data				
CMLM	81.8	48.8	30.1	20.4
+OAXE	86.0	64.3	48.0	35.6
+*ngram*-OAXE	88.2	69.8	54.7	42.3
Distilled Data				
CMLM	86.0	63.3	47.1	36.0
+OAXE	90.6	75.4	62.0	51.1
+*ngram*-OAXE	90.5	76.1	62.9	52.0

Table 4: Accuracy (%) of the generated ngram phrases in the model outputs.

Refer.	The Vollmaringen Male Voice Choir got things running with atmospheric songs such as "Im Weinparadies" and "Lustig, ihr Brüder".
CMLM	The MGV Vollmaringen opened with atmospheric songs songs as "Im Weinparapara" and "Lustig, her brothers".
OAXE	The MGV Vollmaringen opened with atmospheric songs such "" "Im Weinpara" and "Lustig, her brothers".
Ours	The MGV Vollmaringen opened with atmospheric songs such as "Im Weinparadies" and "Lustig, ihr Brüder"

Table 5: Examples of De-En translation for NAT models trained on distilled data. OAXE model often mistakenly translates some ngram phrases (in red color), and *ngram*-OAXE can correctly translate them (in blue color).

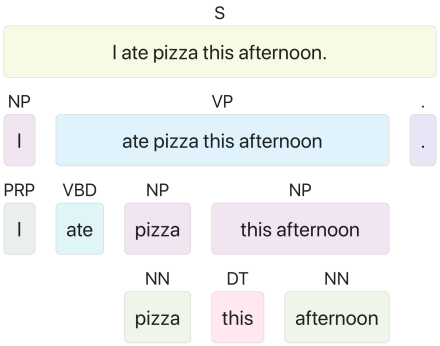

(a) An example constituent tree.

Level	Syntactic Sequence
3	NP VP .
2	PRP VBD NP NP .
1	PRP VBD NN DT NN .

(b) Syntactic sequences at different levels.

Figure 4: Constituent tree of the sentence "I ate pizza this afternoon." (a), and the corresponding syntactic sequences at different levels (b, from bottom to up). "2-level" denotes the syntactic sequence at the last but one level of the constituent tree.

4 Analysis

In this section, we provide some insights where *ngram*-OAXE improves over CMLM (i.e., XE) and OAXE from different perspectives. Otherwise stated, we report results on the WMT14 En-De test set with multiple references (i.e., the column "W14 En-De" in Table 1).

4.1 Analysis of Ngram Translation

We first investigate whether the proposed *ngram*-OAXE improves the generation of phrases in the output. To this end, we use the individual ngram scores as the accuracy of generating ngrams of the corresponding size. An individual ngram score is the evaluation of just matching ngrams of a specific size, such as unigram and bigram.[3] As shown in Table 4, our *ngram*-OAXE consistently outperforms the OAXE counterparts in all ngram levels and the improvement goes up with the increase of ngram, demonstrating that *ngram*-OAXE indeed raises the ability of NAT model on capturing the patterns of ngram phrases.

Case Study Table 5 shows an translation example on the WMT14 De-En testset. The vanilla CMLM model mistakenly generates the ngram phrase "*songs songs as*" with repeated words "*songs*". Although the OAXE model remedies the repetition problem, it fails to generate the phrase "*such as*". In addition, both the CMLM and OAXE models fail to generate the names of the two songs "*Im Weinparadies*" and "*Lustig, ihr Brüder*". Our *ngram*-OAXE successfully generate all the three ngram phrases.

4.2 Analysis of Structure Modeling

Structure Ordering To assess the models' abilities of modeling reordering between ngram phrases, we follow Wang et al. (2021) to measure the precision of outputs at the syntactic level, which can reflect the structure ordering at phrase level. Specifically, we use the syntactic sequence at a certain

[3]The individual ngram BLEUs are generally produced by the BLEU script for the same model outputs, and do not refer to the model training. For example, given the output of *ngram*-OAXE trained on the WMT14 En-De raw data, the script outputs "$BLEU = 61.3\ 88.2/69.8/54.7/42.3$". The final BLEU score is 61.3, and the individual 1gram, 2gram, 3gram, and 4gram BLEU scores are 88.2, 69.8, 54.7, and 42.3, respectively.

Model	Syntactic Level			
	1	2	3	4
Raw Data				
CMLM	18.3	11.5	13.4	17.4
OAXE	37.3	28.5	26.6	28.5
ngram-OAXE	39.9$^{\uparrow\uparrow}$	30.9$^{\uparrow\uparrow}$	29.4$^{\uparrow\uparrow}$	30.0$^{\uparrow\uparrow}$
Distilled Data				
CMLM	31.5	14.7	23.2	25.8
OAXE	41.9	33.0	30.4	31.2
ngram-OAXE	42.6$^{\uparrow\uparrow}$	33.8$^{\uparrow\uparrow}$	31.4$^{\uparrow\uparrow}$	32.3$^{\uparrow\uparrow}$

Table 6: BLEU scores of the syntactic sequence at different levels (from bottom to up).

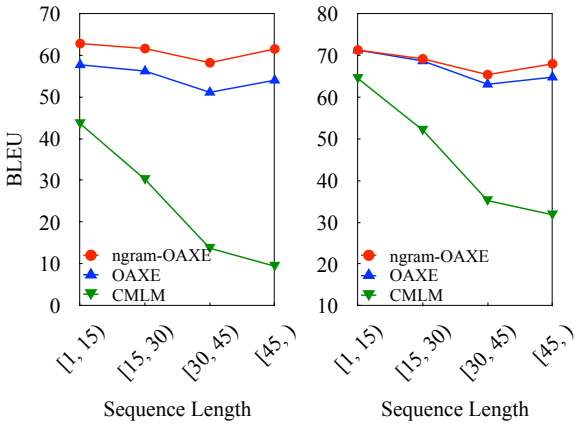

(a) Raw Data (b) Distilled Data

Figure 5: Translation performance with respect to the length of the target sentence.

layer (Figure 4b) of the constituent tree of the generated outputs (Figure 4a). Generally, each tag at a higher syntactic level covers more words (e.g., "VP (ate pizza this afternoon)" at 3-level) and corresponds well to a ngram phrase. Accordingly, higher-level syntactic sequences denotes structure ordering at a larger granularity. We calculate the BLEU score for the syntactic sequences of models' outputs to measure the precision of the structure ordering at different granularities.

Table 6 shows the results for syntactic sequences at different levels. OAXE significantly improves the precision of structure order over the CMLM baseline by a large margin, which is consistent with the claim of Du et al. (2021) that OAXE is better at modeling word order. Our *ngram*-OAXE can further improve the precision of structure order, which we attribute to that *ngram*-OAXE models ordering at a larger granularity (i.e., ngrams).

Model	Repetition	PPLs
Gold Test Set	0.04%	90.9
Raw Data		
CMLM	31.11%	1820.2
+OAXE	3.14%	237.6
+*ngram*-OAXE	2.99%	199.9
Distilled Data		
CMLM	12.10%	1435.5
+OAXE	1.56%	240.8
+*ngram*-OAXE	0.98%	125.9

Table 7: Analyses of the generated outputs. Lower repeated token percentage ("Repetition") denotes lower multimodality in a model. Lower perplexities ("PPLs") denote better fluency.

Sequence Length We also investigate the model performance for different sequence lengths. We split the test sets into different buckets based on the reference sentence length, indicating whether a system does better or worse at shorter or longer sentences. Generally, longer sentences are more complex in linguistic structure. Figure 5 shows that results on the sampled WMT14 En-De test set with multiple references. As seen, the performance of XE drops rapidly when the sequence length increases, and OAXE can significantly improves performance on longer sentences with a better modeling of word order. Our *ngram*-OAXE can handle long sequences even better, which we attribute to the strength of *ngram*-OAXE on both translating longer ngrams and modeling structure ordering between ngram phrases.

4.3 Analysis of Generated Output

Token Repetition One widely-cited weakness of existing NAT models is the multimodality problem, in which a model may consider many possible translations at the same time due to the independent predictions of target words (Gu et al., 2018). Accordingly, the NAT output typically contains many repetitive tokens (e.g., "songs songs" in Table 5). We followed the common practices to use repeated token percentage for measuring multimodality in a NAT model, as listed in Table 7. While OAXE can mostly alleviate the repetition problem, the proposed *ngram*-OAXE can further reduce the repeated percentage over the very strong baseline (e.g., 0.98% vs. 1.56% on distilled data).

Generation Fluency We followed Du et al. (2021) to measure the generation fluency with language models released by Fairseq,[4] which are trained on the News Crawl corpus for the target language. To better evaluate the fluency of the generated output, we use a practical trick *de-duplication* (Saharia et al., 2020) to remove the repetitive tokens. Clearly, *ngram*-OAXE consistently improves fluency in all settings compared with OAXE. We attribute the fluency improvement to the strength of *ngram*-OAXE on both translating longer ngrams[5] and modeling sentence structures.

5 Related Work

Alleviating Multimodality Problem for NAT A number of recent efforts have explored ways to improve the NAT models' ability to handle multimodality. One thread of work iteratively refines the generated outputs with K decoding passes (Lee et al., 2018; Gu et al., 2019), which sacrifices the primary benefit of NAT models – fast inference (Kasai et al., 2021). To maintain the advantage of decoding efficiency, another thread of research aims to improve fully NAT models by building dependencies between target tokens (Ma et al., 2019; Shu et al., 2020), or improving the training loss to ameliorate the effect of multimodality (Ghazvininejad et al., 2020; Saharia et al., 2020; Du et al., 2021).

Knowledge distillation (Kim and Rush, 2016) is the preliminary step for the majority of NAT systems, which can effectively alleviate the multimodality problem by simplifying the training data (Zhou et al., 2020) and reducing the token dependency in target sequence (Ren et al., 2020). However, knowledge distillation relies on an external AT teacher, which prevents NAT models from self-completion. The ultimate goal is to train NAT models from scratch (Huang et al., 2022). Our work shows that augmenting NAT models the ability to handle the complex patterns of raw data (e.g., reordering patterns) with advanced training loss is a promising direction to accomplish the goal.

Incorporating Ngrams into NMT Previous studies have incorporated the ngram phrases as an external signal to guide the generation in AT models (Wang et al., 2017; Zhang et al., 2017; Zhao et al., 2018). Concerning NAT models, Guo et al. (2019) enhance decoder inputs with ngram phrases, Sun et al. (2019) use CRF to model bigram dependencies among target tokens to improve the decoding consistency. Kong et al. (2020) use LSTM to generate ngram chunks, which are then merged via heuristic searching algorithm. Closely related to our work, Ma et al. (2018) use bag of ngram phrases as additional training objective for AT models, and Shao et al. (2020) adapt this idea to NAT models. While Shao et al. (2020) require NAT models to fit all the possible orderings of ngrams, we compute the *ngram*-OAXE loss based on the best ordering of ngrams.

6 Conclusion

In this work, we extend OAXE by modeling ordering at the ngram phrase granularity, which can better ameliorate the effect of multimodality for NAT models. Benefiting from modeling translation at a larger granularity, the proposed *ngram*-OAXE loss performs better at translating phrases and long sentences, and improves the fluency of generated translations. Extensive experiments on representative NAT benchmarks show that *ngram*-OAXE consistently improves translation performance over OAXE, and is especially effective on raw data without distillation.

Acknowledgements

We thank all anonymous reviewers for their insightful comments. And the authors gratefully acknowledge the support by the Lee Kuan Yew Fellowship awarded by Singapore Management University.

References

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. Bert: Pre-training of deep bidirectional transformers for language understanding. In *NAACL*.

Cunxiao Du, Zhaopeng Tu, and Jing Jiang. 2021. Order-agnostic cross entropy for non-autoregressive machine translation. In *ICML*.

Marjan Ghazvininejad, V. Karpukhin, Luke Zettlemoyer, and Omer Levy. 2020. Aligned cross entropy for non-autoregressive machine translation. In *ICML*.

Marjan Ghazvininejad, Omer Levy, Yinhan Liu, and Luke Zettloyer. 2019. Mask-predict: Parallel decoding of conditional masked language models. In *EMNLP*.

[4] https://github.com/pytorch/fairseq/blob/master/examples/language_model/

[5] The longer ngrams generally account for the fluency of the translation.

Jiatao Gu, James Bradbury, Caiming Xiong, Victor OK Li, and Richard Socher. 2018. Non-autoregressive neural machine translation. In *ICLR*.

Jiatao Gu and Xiang Kong. 2021. Fully non-autoregressive neural machine translation: Tricks of the trade. In *Findings of ACL*.

Jiatao Gu, Changhan Wang, and Junbo Zhao. 2019. Levenshtein transformer. In *NeurIPS*.

Junliang Guo, Xu Tan, Di He, Tao Qin, Linli Xu, and Tie-Yan Liu. 2019. Non-autoregressive neural machine translation with enhanced decoder input. In *AAAI*.

Xiao Shi Huang, Felipe Perez, and Maksims Volkovs. 2022. Improving non-autoregressive translation models without distillation. In *ICLR*.

Daniel Kang and Tatsunori Hashimoto. 2020. Improved natural language generation via loss truncation. In *ACL*.

Jungo Kasai, Nikolaos Pappas, Hao Peng, James Cross, and Noah Smith. 2021. Deep encoder, shallow decoder: Reevaluating non-autoregressive machine translation. In *ICLR*.

Yoon Kim and Alexander M Rush. 2016. Sequence-level knowledge distillation. In *EMNLP*.

Diederik P. Kingma and Jimmy Ba. 2015. Adam: A method for stochastic optimization. In *ICLR*.

Xiang Kong, Zhisong Zhang, and Eduard Hovy. 2020. Incorporating a local translation mechanism into non-autoregressive translation. *EMNLP*.

Harold W Kuhn. 1955. The hungarian method for the assignment problem. *Naval research logistics quarterly*.

Jason Lee, Elman Mansimov, and Kyunghyun Cho. 2018. Deterministic non-autoregressive neural sequence modeling by iterative refinement. In *EMNLP*.

Zhuohan Li, Zi Lin, Di He, Fei Tian, Tao Qin, Liwei Wang, and T. Liu. 2019. Hint-based training for non-autoregressive machine translation. In *EMNLP-IJCNLP*.

Jindřich Libovický and Jindřich Helcl. 2018. End-to-end non-autoregressive neural machine translation with connectionist temporal classification. In *EMNLP*.

Shuming Ma, Xu Sun, Yizhong Wang, and Junyang Lin. 2018. Bag-of-words as target for neural machine translation. In *ACL*.

Xuezhe Ma, Chunting Zhou, X. Li, Graham Neubig, and E. Hovy. 2019. Flowseq: Non-autoregressive conditional sequence generation with generative flow. In *EMNLP-IJCNLP*.

Myle Ott, Michael Auli, David Grangier, and Marc'Aurelio Ranzato. 2018. Analyzing uncertainty in neural machine translation. In *ICML*.

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. Bleu: a method for automatic evaluation of machine translation. In *ACL*.

Matt Post. 2018. A call for clarity in reporting bleu scores. In *WMT*.

Lihua Qian, Hao Zhou, Yu Bao, Mingxuan Wang, Lin Qiu, Weinan Zhang, Yong Yu, and Lei Li. 2021. Glancing transformer for non-autoregressive neural machine translation. In *ACL*.

Yi Ren, Jinglin Liu, Xu Tan, Zhou Zhao, Sheng Zhao, and Tie-Yan Liu. 2020. A study of non-autoregressive model for sequence generation. In *ACL*.

Chitwan Saharia, William Chan, Saurabh Saxena, and Mohammad Norouzi. 2020. Non-autoregressive machine translation with latent alignments. In *EMNLP*.

Rico Sennrich, Barry Haddow, and Alexandra Birch. 2016. Neural machine translation of rare words with subword units. In *ACL*.

Chenze Shao, Jinchao Zhang, Yang Feng, Fandong Meng, and Jie Zhou. 2020. Minimizing the bag-of-ngrams difference for non-autoregressive neural machine translation. In *AAAI*.

Raphael Shu, Jason Lee, Hideki Nakayama, and Kyunghyun Cho. 2020. Latent-variable non-autoregressive neural machine translation with deterministic inference using a delta posterior. In *AAAI*.

Zhiqing Sun, Zhuohan Li, Haoqing Wang, Zi Lin, Di He, and Zhi-Hong Deng. 2019. Fast structured decoding for sequence models. In *NeurIPS*.

Zhiqing Sun and Yiming Yang. 2020. An em approach to non-autoregressive conditional sequence generation. In *ICML*.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Lukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *NeurIPS*.

Shuo Wang, Zhaopeng Tu, Zhixing Tan, Shuming Shi, Maosong Sun, and Yang Liu. 2021. On the language coverage bias for neural machine translation. In *Findings of ACL*.

Wenxuan Wang and Zhaopeng Tu. 2020. Rethinking the value of transformer components. In *COLING*.

Xing Wang, Zhengdong Lu, Zhaopeng Tu, Hang Li, Deyi Xiong, and Min Zhang. 2017. Neural machine translation advised by statistical machine translation. In *AAAI*.

Jiacheng Zhang, Yang Liu, Huanbo Luan, Jingfang Xu, and Maosong Sun. 2017. Prior knowledge integration for neural machine translation using posterior regularization. In *ACL*.

Yang Zhao, Yining Wang, Jiajun Zhang, and Chengqing Zong. 2018. Phrase table as recommendation memory for neural machine translation. In *IJCAI*.

Chunting Zhou, Graham Neubig, and Jiatao Gu. 2020. Understanding knowledge distillation in non-autoregressive machine translation. In *ICLR*.

Language Branch Gated Multilingual Neural Machine Translation

Haoran Sun and Deyi Xiong*
College of Intelligence and Computing, Tianjin University, Tianjin, China
hrsun0105@gmail.com, dyxiong@tju.edu.cn

Abstract

Knowledge transfer across languages is crucial for multilingual neural machine translation. In this paper, we propose language branch (LB) gated multilingual neural machine translation that encourages knowledge transfer within the same language branch with a LB-gated module that is integrated into both the encoder and decoder. The LB-gated module distinguishes LB-specific parameters from global parameters shared by all languages and routes languages from the same LB to the corresponding LB-specific network. Comprehensive experiments on the OPUS-100 dataset show that the proposed approach substantially improves translation quality on both middle- and low-resource languages over previous methods. Further analysis demonstrates its ability in learning similarities between language branches.

1 Introduction

Recent years have witnessed a growing interest in multilingual neural machine translation (NMT), which supports translation among multiple languages with one single model (Dong et al., 2015; Luong et al., 2016; Firat et al., 2016; Johnson et al., 2017; Aharoni et al., 2019; Arivazhagan et al., 2019; Xu et al., 2021). As the parameters of multilingual NMT are fully or partially shared by multiple languages, knowledge transfer across languages improves translation quality of low-resource languages.

Despite these advantages, there still exist challenges in multilingual NMT. As previous studies have found, multilingual NMT usually underperforms its bilingual counterparts on high-resource languages (Johnson et al., 2017; Arivazhagan et al., 2019). A way to alleviate this issue is to use language-aware modules, which could be more computationally efficient than simply enlarging model capacity with deeper/wider models (Zhang et al., 2020, 2021; Lin et al., 2021). As for low-resource languages, positive transfer is more pronounced among related languages than distant languages (Sachan and Neubig, 2018). Additionally, explicit language clustering benefits multilingual NMT (Tan et al., 2019) although itself can learn linguistic typology during training (Lu et al., 2018).

Inspired by these, we propose language branch (subfamily) gated multilingual neural machine translation, which fuses language branch information into multilingual NMT by a **L**anguage **B**ranch **G**ated **M**odule (LBGM). A language branch is a subfamily of a language family. Take the Indo-European language family as an example. It can be further divided into subfamilies like Germanic, Slavic, Celtic, etc. The reason why we use language branches rather than language families is that the latter are relatively coarse-grained. Languages within a language branch are more closely related to each other than those in a language family.

A token that indicates the language branch for the current sentence is fed into LBGM, in addition to the input from other layers in multilingual NMT. With the language branch token, LBGM distinguishes language-branch-specific parameters from global parameters shared by all languages and uses a gate to aggregate these two parts as the output of the module.

We conduct experiments on the OPUS-100 dataset (Zhang et al., 2020) with a large number of different languages. Our main findings can be summarized as follows:

- The proposed LBGM can significantly improve translation quality, achieving more substantial gains for middle- and low-resource languages than its counterparts.
- LBGM performs better for language branches that contain plentiful languages, including

*Corresponding author.

Proceedings of the 29th International Conference on Computational Linguistics, pages 5046–5053
October 12–17, 2022.

not only high-resource languages, but also middle/low-resource languages.

- LBGM is capable of capturing similarities between language branches.

2 Related Work

Research of multilingual NMT mainly focuses on partial or full parameter sharing in NMT modules (Dong et al., 2015; Luong et al., 2016; Firat et al., 2016). Johnson et al. (2017) propose prefixing sentences with a language token in a joint set of parallel corpora, using a single NMT model to enable multilingual translation. There is a trade-off between boosting the performance of low-resource languages and sacrificing the performance of high-resource languages (Arivazhagan et al., 2019).

Language-Specific Parameters Previous works have been trying to use language-specific parameters to alleviate the trade-off issue in multilingual NMT, such as adding adaptation layers into pre-trained models for each language (Bapna and Firat, 2019; Philip et al., 2020; Zhu et al., 2021). Zhang et al. (2020) propose language-aware layer normalization to relax normalization constraint for target languages. Lin et al. (2021) produce masks for different language pairs and use them to select sub-networks for language pairs, in order to counter parameter interference. The closest work to ours is done by Zhang et al. (2021), who introduce a module called CLSR into the Transformer model. The CLSR module adopts a gating function, which is trained with injected zero-mean Gaussian noise and discretized at inference time, to choose whether to share the parameters for all languages or not. However, the CLSR does not take the relationship between languages into account, since it uses language identity as the router.

Language Clustering Sachan and Neubig (2018) have found that full parameter sharing improves translation quality mainly for related languages that are from the same language group. Tan et al. (2019) attempt to cluster languages into different groups using two methods: prior knowledge and language embedding. They build a multilingual NMT model for each group and observe that both clustering approaches are able to improve model performance. Fan et al. (2021) propose adding a language-specific layer for each language group. They cluster languages according to the amount of training data and vocabulary. Different from them, we use a linguistically-motivated and fine-grained method to group languages (i.e., language subfamily).

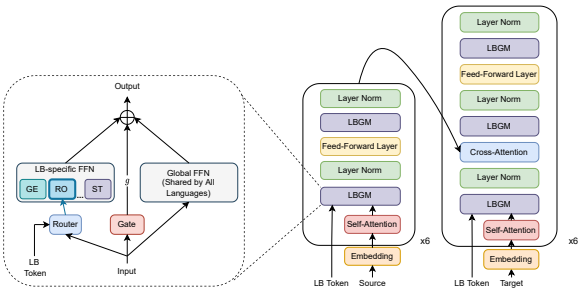

Figure 1: Illustration of the proposed LBGM model.

3 Methodology

We adopt the Transformer model as the backbone network (Vaswani et al., 2017). Following Johnson et al. (2017), we extend the Transformer to multilingual Transformer by prefixing a language token to the source and target side respectively.

We cluster languages into their language branches. More specifically, for Indo-European languages, we cluster them into 10 branches, including Baltic, Celtic, East Slavic, Germanic, Hellenic, Indo-Aryan, Iranian, Romance, South Slavic and West Slavic. But for Afro-Asiatic languages, as there are only five languages in the dataset we use, we just group them into one language branch. For isolated languages, e.g., Esperanto, Japanese, we keep them in the isolated language group, expecting them to benefit from the positive transfer from all languages.

We adapt the multilingual Transformer to integrate language branch information with the proposed LBGM, which is illustrated in Figure 1. Particularly, we use an additional token, i.e., LB token, to indicate the language branch for each corresponding sentence. LBGM contains a gating function and two feedforward networks, namely, LB-specific FFN and global FFN. The LB-specific FFN is exclusively used for sentences from the corresponding language branch while the global FFN is fully shared across all languages. The purpose of the fully-shared global FFN is to capture global linguistic information from all language pairs, so as to enable knowledge transfer across all languages, especially for the isolated language branch which only contains one or a few languages. The LB-specific FFN is to capture linguistic features for languages from the same language branch and hence

enabling intra-language-branch knowledge transfer. LBGM can be formulated as follows:

$$\text{Gate}(\boldsymbol{a}^l) = \text{ReLU}(\boldsymbol{a}^l \boldsymbol{W} + b) \quad (1)$$

$$g^l = \sigma(\text{Gate}(\boldsymbol{a}^l)) \quad (2)$$

$$\text{LBGM}(\boldsymbol{a}^l) = g^l \cdot \text{FFN}^{\text{LB-specific}}(\boldsymbol{a}^l) + (1 - g^l) \cdot \text{FFN}^{\text{global}}(\boldsymbol{a}^l) \quad (3)$$

where $\boldsymbol{W}$ and b are trainable parameters, $\boldsymbol{a}^l$ is the output from the preceding layer l. FFNs are calculated as follows:

$$\text{FFN}^{\text{LB-specific}} = \boldsymbol{a}^l \boldsymbol{W}^{\text{LB-specific}} + \boldsymbol{b}^{\text{LB-specific}} \quad (4)$$

$$\text{FFN}^{\text{global}} = \boldsymbol{a}^l \boldsymbol{W}^{\text{global}} + \boldsymbol{b}^{\text{global}} \quad (5)$$

The LB token is used to route information from previous layer into the corresponding LB-specific FFN, which is acting as an "expert" for that language branch.

In contrast to the previous CLSR method Zhang et al. (2021), we do not use the discrete gating function and additional loss component. In our preliminary experiments, we have found that these two components could not bring improvements to our LBGM module and the discretized gating function is even harmful to the LBGM. Instead, we use an individual gating function per LBGM sub-module (not shared by the whole model like CLSR). This is because, at different positions, the importance of the two types of FFN may be different, which is to be determined by the corresponding gating function in our model.

4 Experiments

We conducted experiments with a massive number of languages to examine the effectiveness of the proposed LBGM.

4.1 Settings

We used the OPUS-100 dataset (Zhang et al., 2020) for our experiments. OPUS-100[1] is an English-centric dataset covering 99 language pairs. As 5 language pairs do not have their test and dev sets, we conducted experiments using the rest 94 language pairs. We roughly divided the languages into three categories according to the training data size: high-resource languages (more than 1M training samples, 44 languages), low-resource languages (fewer than 0.1M training samples, 21 languages) and middle-resource languages (others, 29 languages) following Zhang et al. (2020). This division is only for experiments while in our model, we linguistically grouped these languages into 26 language branches, shown in Table 5 in Appendix A.

We applied Byte Pair Encoding (BPE) (Sennrich et al., 2016) to preprocess the data with a joint vocabulary size of 64K, using the SentencePiece Toolkit (Kudo and Richardson, 2018)[2]. We adopted the temperature-based oversampling method with a temperature of $T = 5$.

Translation quality was evaluated by BLEU (Papineni et al., 2002) using SacreBleu (Post, 2018)[3].

We adopted the Transformer-base model (Vaswani et al., 2017) as our baseline. The dimensions of our LB-specific FFNs and global FFN are 512. As we have 26 language branches, so the total number of LB-specific FFNs is 26. We also compared with CLSR (Zhang et al., 2021), which uses a language-specific module and conditional routing function to learn the representation of each language. All models were implemented with fairseq (Ott et al., 2019)[4].

Other details about experiments and model settings are in Appendix A.

4.2 Main Results of One-to-Many and Many-to-One Translation

We first conducted experiments for one-to-many and many-to-one translation (i.e., English→X and X→English). Although there is only one language on the source side for one-to-many translation and on the target side for many-to-one translation, we use the proposed LBGM in both the encoder and decoder, as shown in Figure 1, to keep both LB-informed.

As shown in Table 1, our LBGM outperforms the baseline and CLSR on the OPUS-100 dataset. Particularly, we achieve an overall improvement of 1.32/0.70 BLEU points on English→X, 1.31/0.30 BLEU points on X→English over the baseline/CLSR. In terms of the amount of training data available, we observe that the proposed LBGM gains larger improvements on low-resource languages than those on high/middle-resource languages over the baseline and CLSR.

[1] http://opus.nlpl.eu/opus-100.php

[2] https://github.com/google/sentencepiece

[3] https://github.com/mjpost/sacrebleu

[4] https://github.com/facebookresearch/fairseq

Model	#Params	English→X			
		High	Mid	Low	All
Baseline	99M	15.01	22.26	25.99	19.70
Baseline-592dim	117M	15.44	22.72	26.41	20.14
CLSR	154M	15.81	22.73	26.43	20.32
LBGM-LS	154M	**16.01**	23.13	26.87	20.63
LBGM	117M	15.95	**23.77**	**27.85**	**21.02**

Model	#Params	X→English			
		High	Mid	Low	All
Baseline	99M	20.71	23.15	25.33	22.49
Baseline-592dim	117M	21.68	23.69	25.80	23.22
CLSR	154M	21.82	24.22	26.07	23.50
LBGM-LS	154M	**22.09**	23.77	26.10	23.51
LBGM	117M	22.04	**24.58**	**26.42**	**23.80**

Table 1: Results on the OPUS-100 dataset. We report the average BLEU of English→X and X→English translation on 94 language pairs.

4.3 Ablation Study

In order to eliminate the difference in the number of parameters of the LBGM and baseline, we scaled the hidden size of the baseline from 512 to 592 dimensions, denoted in Table 1 as Baseline-592dim. The results still demonstrate that our LBGM is more efficient than the vanilla Transformer. This ablation study confirms that the improvement obtained by the LBGM is not due to the capacity advantage.

We conducted another group of experiments to investigate whether language branches are helpful in comparison to the original CLSR architecture, which could tell us whether the modified LBGM architecture would be better adapted to the use of language branches. We used the same settings as the previous experiments. Differently, language identities, instead of language branches, were used as the router. The results are shown in Table 1 denoted as LBGM-LS. From the comparison between the LBGM-LS and LBGM, we can find that language branches are indeed important. Although the LBGM-LS obtains a slightly higher improvement than the LBGM on high-resource languages (0.06 BLEU), it's acceptable for our LBGM approach. That is because the LBGM-LS allocates an individual FFN module for each language, rather than one for each language branch, and it alleviates the capacity constraints of the model on high-resource languages. LBGM-LS still outperforms the CLSR approach, suggesting that our modifications to the original CLSR module are not harmful to the model but more appropriate in the context of using language branches.

Lang	Data Size	Baseline	CLSR	LBGM	Δ-B	Δ-C
da	1000000	20.74	22.45	**22.64**	1.90	0.19
de	1000000	16.90	17.79	**18.03**	1.13	0.24
is	1000000	10.83	11.86	**12.01**	1.18	0.15
nl	1000000	16.58	17.34	**17.86**	1.28	0.52
no	1000000	17.81	18.92	**19.34**	1.53	0.42
sv	1000000	18.37	19.27	**19.82**	1.45	0.55
nn	486055	23.55	24.88	**25.40**	1.85	0.52
af	275512	30.02	31.14	**32.15**	2.13	1.01
nb	142906	23.16	24.39	**25.48**	2.32	1.09
fy	54342	25.03	27.02	**27.63**	2.60	0.61
li	25535	27.42	29.27	**30.59**	3.17	1.32
yi	15010	25.29	27.31	**30.73**	5.44	3.42

Table 2: Results on the English→X translation for the Germanic language branch, which includes high/middle/low-resource language pairs. Δ-B and Δ-C denote the improvements over Baseline and CLSR respectively.

LB	Lang	Data Size	Bilingual	Baseline	CLSR	LBGM
WS	cs	1000000	**18.03**	14.92	15.46	16.11
	pl	1000000	**15.37**	11.57	11.83	12.35
	sk	1000000	**18.66**	15.89	16.71	17.44
BA	lt	1000000	17.63	18.55	**19.82**	19.61
	lv	1000000	21.10	20.83	**22.57**	22.09

Table 3: Results on the West Slavic (WS) language branch and Baltic (BA) language branch on English→X translation. Bilingual is the bilingual model trained with the same architecture.

4.4 Effect on the Different Types of Language Branches

To further investigate how the proposed LBGM improves translation quality, we categorize language branches into three types: (I) language branch containing high/middle/low-resource languages. (II) language branch with only high-resource languages. (III) language branch with one or two languages, which are usually isolated languages. We analyzed the effects of our LBGM on these three types of language branches.

Table 2 shows the results of the Germanic language branch, a Type-I language branch as mentioned above. We list the languages (denoted by their ISO-639-1 codes) in descending order of the amount of training data. On this language branch type, LBGM outperforms both the baseline and CLSR on all languages with different levels of resource. Particularly, as the amount of training data decreases, the improvements over the baseline and CLSR increase.

For the Type-II language branch (i.e., including only high-resource languages), we show the results of both West Slavic and Baltic language branches on English→X translation in Table 3. As these language branches include only high-resource lan-

Lang	Data Size	Baseline	CLSR	LBGM
ja	1000000	4.73	**5.27**	4.46
ko	1000000	**3.03**	2.91	2.84
ka	377306	14.39	**14.91**	14.57

Table 4: Results on the isolated language branch of English→X translation.

guages, negative transfer usually happens (compared to bilingual models). Fortunately, on this type of language branches, our LBGM still substantially outperforms the baseline and is better than or comparable with CLSR.

Finally, Table 4 shows results on the isolated language branch. It can be seen that LBGM achieves results comparable to CLSR. And the effect on isolated languages depends more on the characteristics of the language itself.

All these results suggest that our LBGM can significantly improve performance on both middle- and low-resource languages and achieve comparable results to CLSR on high-resource languages but with fewer parameters. The gains on middle- and low-resource languages are more substantial when the language branch contains mixed languages in terms of the amount of available training data.

5 Analysis

Ideally, the closer two language branches are to each other, the more similar the parameters of the corresponding LB-specific FFNs are to each other in LBGM, especially for language branches which are from the same language family. Motivated by this, we calculated the pairwise cosine similarities of LB-specific FFNs of any two different language branches learned in LBGM, trying to use these similarities to measure the relationship between language branches. Specifically, the cosine similarity is computed with the weight matrix of LB-specific FFN (reshaped into a vector via row-wise concatenation) as follows:

$$\text{Cosine Similar}(\boldsymbol{V_1}, \boldsymbol{V_2}) = \frac{\boldsymbol{V_1} \cdot \boldsymbol{V_2}}{\|\boldsymbol{V_1}\|\|\boldsymbol{V_2}\|} \quad (6)$$

where $\|\cdot\|$ denotes the L_2 norm, $\boldsymbol{V_1}$ and $\boldsymbol{V_2}$ are vectors reshaped from weight matrices.

Figure 2 shows the cosine similarity matrix of different language branches (26 in total contained in the OPUS-100 dataset) learned by our LBGM. The deeper the color is, the more similar the two language branches are. It is clear to see that

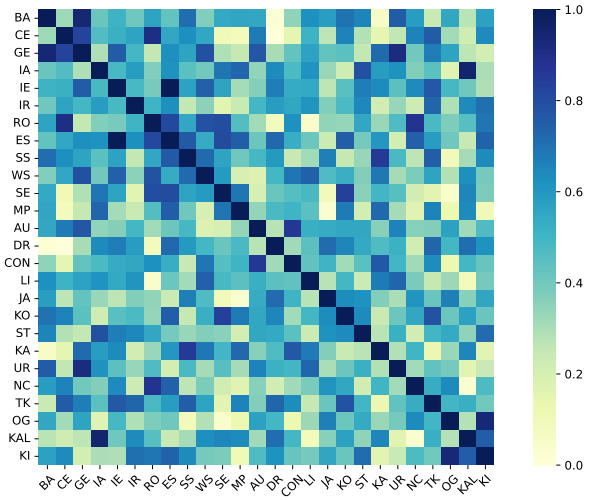

Figure 2: Pairwise language branch similarities learned by LBGM with one-to-many translation. Better view with color.

GE (Germanic) is more similar to BA (Baltic), WS (West Slavic), CE (Celtic), etc., which are from the same language family, than other language branches, indicated by deeper color in Figure 2. Similar results can be observed among JA (Japonic), KO (Koreanic) and ST (Sino-Tibetan) language branches, among OG (Oghuz), KAL (Karluk) and KI (Kipchak) language branches, etc. This suggests that the proposed LBGM is able to learn similarities between different language branches.

6 Conclusions

In this paper, we have presented LBGM that uses a LB-specific FFN and a global FFN shared across all languages to enhance knowledge transfer within the same language branch for multilingual neural machine translation. Experiments on the OPUS-100 dataset have shown that LBGM can significantly improve translation quality on both middle- and low-resource languages, over the baseline and CLSR (Zhang et al., 2021). Further analysis on LB-specific FFN discloses that the proposed LBGM is able to capture language branch relations.

Acknowledgments

The present research was supported by the Key Research and Development Program of Yunnan Province (Grant No. 202203AA080004-2). We would like to thank the anonymous reviewers for their insightful comments.

References

Roee Aharoni, Melvin Johnson, and Orhan Firat. 2019. Massively multilingual neural machine translation. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 3874–3884, Minneapolis, Minnesota. Association for Computational Linguistics.

Naveen Arivazhagan, Ankur Bapna, Orhan Firat, Dmitry Lepikhin, Melvin Johnson, Maxim Krikun, Mia Xu Chen, Yuan Cao, George F. Foster, Colin Cherry, Wolfgang Macherey, Zhifeng Chen, and Yonghui Wu. 2019. Massively multilingual neural machine translation in the wild: Findings and challenges. *CoRR*, abs/1907.05019.

Ankur Bapna and Orhan Firat. 2019. Simple, scalable adaptation for neural machine translation. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 1538–1548, Hong Kong, China. Association for Computational Linguistics.

Daxiang Dong, Hua Wu, Wei He, Dianhai Yu, and Haifeng Wang. 2015. Multi-task learning for multiple language translation. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 1723–1732, Beijing, China. Association for Computational Linguistics.

Angela Fan, Shruti Bhosale, Holger Schwenk, Zhiyi Ma, Ahmed El-Kishky, Siddharth Goyal, Mandeep Baines, Onur Celebi, Guillaume Wenzek, Vishrav Chaudhary, Naman Goyal, Tom Birch, Vitaliy Liptchinsky, Sergey Edunov, Michael Auli, and Armand Joulin. 2021. Beyond english-centric multilingual machine translation. *Journal of Machine Learning Research*, 22(107):1–48.

Orhan Firat, Kyunghyun Cho, and Yoshua Bengio. 2016. Multi-way, multilingual neural machine translation with a shared attention mechanism. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 866–875, San Diego, California. Association for Computational Linguistics.

Melvin Johnson, Mike Schuster, Quoc V. Le, Maxim Krikun, Yonghui Wu, Zhifeng Chen, Nikhil Thorat, Fernanda Viégas, Martin Wattenberg, Greg Corrado, Macduff Hughes, and Jeffrey Dean. 2017. Google's multilingual neural machine translation system: Enabling zero-shot translation. *Transactions of the Association for Computational Linguistics*, 5:339–351.

Diederik P. Kingma and Jimmy Ba. 2015. Adam: A method for stochastic optimization. In *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*.

Taku Kudo and John Richardson. 2018. SentencePiece: A simple and language independent subword tokenizer and detokenizer for neural text processing. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 66–71, Brussels, Belgium. Association for Computational Linguistics.

Zehui Lin, Liwei Wu, Mingxuan Wang, and Lei Li. 2021. Learning language specific sub-network for multilingual machine translation. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 293–305, Online. Association for Computational Linguistics.

Yichao Lu, Phillip Keung, Faisal Ladhak, Vikas Bhardwaj, Shaonan Zhang, and Jason Sun. 2018. A neural interlingua for multilingual machine translation. In *Proceedings of the Third Conference on Machine Translation: Research Papers*, pages 84–92, Brussels, Belgium. Association for Computational Linguistics.

Minh-Thang Luong, Quoc V. Le, Ilya Sutskever, Oriol Vinyals, and Lukasz Kaiser. 2016. Multi-task sequence to sequence learning. In *4th International Conference on Learning Representations, ICLR 2016, San Juan, Puerto Rico, May 2-4, 2016, Conference Track Proceedings*.

Myle Ott, Sergey Edunov, Alexei Baevski, Angela Fan, Sam Gross, Nathan Ng, David Grangier, and Michael Auli. 2019. fairseq: A fast, extensible toolkit for sequence modeling. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics (Demonstrations)*, pages 48–53, Minneapolis, Minnesota. Association for Computational Linguistics.

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. BLEU: a method for automatic evaluation of machine translation. In *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics*, pages 311–318, Philadelphia, Pennsylvania, USA. Association for Computational Linguistics.

Jerin Philip, Alexandre Berard, Matthias Gallé, and Laurent Besacier. 2020. Monolingual adapters for zero-shot neural machine translation. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 4465–4470, Online. Association for Computational Linguistics.

Matt Post. 2018. A call for clarity in reporting BLEU scores. In *Proceedings of the Third Conference on Machine Translation: Research Papers*, pages 186–191, Brussels, Belgium. Association for Computational Linguistics.

Devendra Sachan and Graham Neubig. 2018. Parameter sharing methods for multilingual self-attentional translation models. In *Proceedings of the Third Conference on Machine Translation: Research Papers*, pages 261–271, Brussels, Belgium. Association for Computational Linguistics.

Rico Sennrich, Barry Haddow, and Alexandra Birch. 2016. Neural machine translation of rare words with subword units. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1715–1725, Berlin, Germany. Association for Computational Linguistics.

Xu Tan, Jiale Chen, Di He, Yingce Xia, Tao Qin, and Tie-Yan Liu. 2019. Multilingual neural machine translation with language clustering. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 963–973, Hong Kong, China. Association for Computational Linguistics.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, 4-9 December 2017, Long Beach, CA, USA*, pages 5998–6008.

Hongfei Xu, Qiuhui Liu, Josef van Genabith, and Deyi Xiong. 2021. Modeling task-aware MIMO cardinality for efficient multilingual neural machine translation. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 2: Short Papers)*, pages 361–367, Online. Association for Computational Linguistics.

Biao Zhang, Ankur Bapna, Rico Sennrich, and Orhan Firat. 2021. Share or not? learning to schedule language-specific capacity for multilingual translation. In *9th International Conference on Learning Representations, ICLR 2021, Virtual Event, Austria, May 3-7, 2021*.

Biao Zhang, Philip Williams, Ivan Titov, and Rico Sennrich. 2020. Improving massively multilingual neural machine translation and zero-shot translation. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 1628–1639, Online. Association for Computational Linguistics.

Yaoming Zhu, Jiangtao Feng, Chengqi Zhao, Mingxuan Wang, and Lei Li. 2021. Counter-interference adapter for multilingual machine translation. In *Findings of the Association for Computational Linguistics: EMNLP 2021*, pages 2812–2823, Punta Cana, Dominican Republic. Association for Computational Linguistics.

ISO	Name	Family	Branch	Code	ISO	Name	Family	Branch	Code
am	Amharic	Afro-Asiatic	Semitic	SE	fa	Persian	Indo-European	Iranian	IR
ar	Arabic	Afro-Asiatic	Semitic	SE	ku	Kurdish	Indo-European	Iranian	IR
ha	Hausa	Afro-Asiatic	Chadic	SE	ps	Pashto	Indo-European	Iranian	IR
he	Hebrew	Afro-Asiatic	Semitic	SE	tg	Tajik	Indo-European	Iranian	IR
mt	Maltese	Afro-Asiatic	Semitic	SE	ca	Catalan	Indo-European	Romance	RO
km	Khmer	Austroasiatic	Austroasiatic	AU	es	Spanish	Indo-European	Romance	RO
vi	Vietnamese	Austroasiatic	Austroasiatic	AU	fr	French	Indo-European	Romance	RO
id	Indonesian	Austronesian	Malayo-Polynesian	MP	gl	Galician	Indo-European	Romance	RO
mg	Malagasy	Austronesian	Malayo-Polynesian	MP	it	Italian	Indo-European	Romance	RO
ms	Malay	Austronesian	Malayo-Polynesian	MP	oc	Occitan	Indo-European	Romance	RO
eo	Esperanto	Constructed	Constructed	CON	pt	Portuguese	Indo-European	Romance	RO
kn	Kannada	Dravidian	Dravidian	DR	ro	Romanian	Indo-European	Romance	RO
ml	Malayalam	Dravidian	Dravidian	DR	wa	Walloon	Indo-European	Romance	RO
ta	Tamil	Dravidian	Dravidian	DR	bg	Bulgarian	Indo-European	South Slavic	SS
te	Tegulu	Dravidian	Dravidian	DR	bs	bosanski	Indo-European	South Slavic	SS
lt	Lithuanian	Indo-European	Baltic	BA	hr	Croatian	Indo-European	South Slavic	SS
lv	Latvian	Indo-European	Baltic	BA	mk	Macedonian	Indo-European	South Slavic	SS
br	Breton	Indo-European	Celtic	CE	sh	Serbo-Croatian	Indo-European	South Slavic	SS
cy	Welch	Indo-European	Celtic	CE	sl	Slovenian	Indo-European	South Slavic	SS
ga	Irish	Indo-European	Celtic	CE	sr	Serbian	Indo-European	South Slavic	SS
gd	Scots Gaelic	Indo-European	Celtic	CE	cs	Czech	Indo-European	West Slavic	WS
be	Byelorussian	Indo-European	East Slavic	ES	pl	Polish	Indo-European	West Slavic	WS
ru	Russian	Indo-European	East Slavic	ES	sk	Slovak	Indo-European	West Slavic	WS
uk	Ukrainian	Indo-European	East Slavic	ES	ja	Japanese	Japonic	Japonic	JA
af	Afrikaans	Indo-European	Germanic	GE	ka	Georgian	Kartvelian	Kartvelian	KA
da	Danish	Indo-European	Germanic	GE	ko	Korean	Koreanic	Koreanic	KO
de	German	Indo-European	Germanic	GE	eu	Basque	Language isolate	Language isolate	LI
fy	Frisian	Indo-European	Germanic	GE	ig	Igbo	Niger–Congo	Niger–Congo	NC
is	Icelandic	Indo-European	Germanic	GE	rw	Kinyarwanda	Niger–Congo	Niger–Congo	NC
li	Limburgan	Indo-European	Germanic	GE	xh	Xhosa	Niger–Congo	Niger–Congo	NC
nb	Bokmål	Indo-European	Germanic	GE	zu	Zulu	Niger–Congo	Niger–Congo	NC
nl	Dutch	Indo-European	Germanic	GE	my	Burmese	Sino-Tibetan	Sino-Tibetan	ST
nn	Nynorsk	Indo-European	Germanic	GE	zh	Chinese	Sino-Tibetan	Sino-Tibetan	ST
no	Norwegian	Indo-European	Germanic	GE	th	Thai	Tai–Kadai	Tai–Kadai	TK
sv	Swedish	Indo-European	Germanic	GE	ug	Uigur	Turkic	Karluk	KAL
yi	Yiddish	Indo-European	Germanic	GE	uz	Uzbek	Turkic	Karluk	KAL
as	Assamese	Indo-European	Indo-Aryan	IA	kk	Kazakh	Turkic	Kipchak	KI
bn	Bengali	Indo-European	Indo-Aryan	IA	ky	Kirghiz	Turkic	Kipchak	KI
gu	Gujarati	Indo-European	Indo-Aryan	IA	tt	Tatar	Turkic	Kipchak	KI
hi	Hindi	Indo-European	Indo-Aryan	IA	az	Azerbaijani	Turkic	Oghuz	OG
mr	Marathi	Indo-European	Indo-Aryan	IA	tk	Turkmen	Turkic	Oghuz	OG
ne	Nepali	Indo-European	Indo-Aryan	IA	tr	Turkish	Turkic	Oghuz	OG
or	Oriya	Indo-European	Indo-Aryan	IA	et	Estonian	Uralic	Uralic	UR
pa	Punjabi	Indo-European	Indo-Aryan	IA	fi	Finnish	Uralic	Uralic	UR
si	Singhalese	Indo-European	Indo-Aryan	IA	hu	Hungarian	Uralic	Uralic	UR
ur	Urdu	Indo-European	Indo-Aryan	IA	se	Northern Sami	Uralic	Uralic	UR
el	Greek	Indo-European	Hellenic	HE					
sq	Albanian	Indo-European	Albanian	HE					

Table 5: ISO-639-1 language code, language name, language family, language branch and language branch code in the OPUS-100 dataset.

A Appendix

A.1 Languages in the OPUS-100 Dataset

We list the languages used in our experiments in Table 5. The language branches are most based on linguistic characteristics, but some of them are based on geopolitical locations, e.g., Greek and Albanian.

A.2 Experiment Settings

The Transformer-base model has 6 layers for both encoder and decode, 8 attention heads and 512 dimensions for embeddings, 2048 dimensions for FFN layer. We set the dropout rate to 0.1 for all modules. The hyperparameters of our LBGM were the same as the Transformer-base model. The gate function is implemented by FFN which input dimension is 512 and output dimension is 1.

We optimized parameters using Adam optimizer (Kingma and Ba, 2015) with a label smoothing rate of 0.1. The learning rate was scheduled according to the inverse square root of running steps with a warmup step of 4K and the weight decay rate was set to 0.0001. We set the maximum number of steps to 500K. For inference, we used beam search with a beam size of 4 and a length penalty of 0.6.

Iterative Constrained Back-Translation for Unsupervised Domain Adaptation of Machine Translation

Hongxiao Zhang[1], Hui Huang[2], Jiale Gao[1], Yufeng Chen[1]*, Jinan Xu[1], Jian Liu[1]
[1]Beijing Jiaotong University, Beijing, China
[2]Harbin Institute of Technology, Harbin, China
{hongxiaozhang,jialegao,chenyf,jaxu,jianliu}@bjtu.edu.cn,
huanghui_hit@126.com

Abstract

Back-translation (BT) has been proven to be effective in unsupervised domain adaptation of neural machine translation (NMT). However, the existing back-translation methods mainly improve domain adaptability by generating in-domain pseudo-parallel data that contains sentence-structural knowledge, paying less attention to the in-domain lexical knowledge, which may lead to poor translation of unseen in-domain words. In this paper, we propose an **I**terative **C**onstrained **B**ack-**T**ranslation (**ICBT**) method to incorporate in-domain lexical knowledge on the basis of BT for unsupervised domain adaptation of NMT. Specifically, we apply lexical constraints into back-translation to generate pseudo-parallel data with in-domain lexical knowledge, and then perform round-trip iterations to incorporate more lexical knowledge. Based on this, we further explore sampling strategies of constrained words in ICBT to introduce more targeted lexical knowledge, via domain specificity and confidence estimation. Experimental results on four domains show that our approach achieves state-of-the-art results, improving the BLEU score by up to 3.08 compared to the strongest baseline, which demonstrates the effectiveness of our approach. The codes and models are publicly available at https://github.com/zzzxiaohong/ICBT.

1 Introduction

Neural machine translation (NMT) has made breakthroughs in resource-rich domains (Bahdanau et al., 2015; Vaswani et al., 2017), which requires abundant in-domain parallel data (Koehn and Knowles, 2017). Unfortunately, there is no enough parallel data for many domains, while the monolingual corpus is much easier to obtain. Therefore, unsupervised domain adaptation of NMT, which aims to improve in-domain translation through out-of-domain parallel corpus and in-domain monolingual corpus (Chu and Wang, 2018), has been extensively researched in recent literatures (Gulcehre et al., 2015; Sennrich et al., 2016a; Dou et al., 2019a).

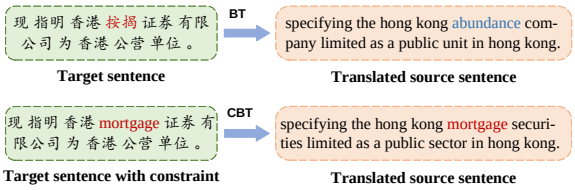

Figure 1: An example of the English-Chinese translation task to illustrate the effect of constrained back-translation (CBT) compared to back-translation (BT). The red fonts indicate the ground truth of the term, and the blue fonts show incorrect translation.

Among the existing techniques for unsupervised domain adaptation of NMT, the data-based ones are a significant part (Chu and Wang, 2018), which usually use the in-domain monolingual corpus to build pseudo-parallel data, and then use the synthetic data to fine-tune the pre-trained NMT model (Currey et al., 2017; Hu et al., 2019). Back-translation (BT) (Sennrich et al., 2016a) is one of the most basic data-based approaches. A series of studies on improving BT have emerged in recent years. For example, Wang et al. (2019) make better use of the back-translated synthetic data by introducing confidence estimation. Hoang et al. (2018) design iterative BT through iterations of forward and backward translation models to improve the translation quality of single-round BT. And some work further optimizes the iterative BT by filtering or selecting the data (Dou et al., 2020; Kumari et al., 2021). Although the above methods are proven to be effective, they only focus on reinforcing the sentence-structural knowledge provided by BT when building pseudo-parallel data, which pay less attention to in-domain lexical knowledge.

Daumé III and Jagarlamudi (2011) point out that the mistranslation of unseen (out-of-vocabulary) words accounts for a large proportion when trans-

*Yufeng Chen is the corresponding author.

Proceedings of the 29th International Conference on Computational Linguistics, pages 5054–5065
October 12–17, 2022.

ferring to a new domain. As we all know that a dictionary is a necessary aid to translate new words properly, so it is important to introduce in-domain lexical knowledge when generating pseudo-parallel data. However, the existing BT-based methods also suffer from the problem of domain shift, which leads to inaccurate translation of unseen in-domain words. Take Figure 1 as an example, BT incorrectly translates "按揭 (mortgage)" to "abundance", which fails to introduce the lexical knowledge "按揭-mortgage" to the pseudo-parallel data.

Aiming at the above issue, this paper focuses on introducing lexical knowledge into pseudo-parallel data from BT, and proposes a novel method named **I**terative **C**onstrained **B**ack-**T**ranslation (**ICBT**). Specifically, assuming that in-domain monolingual data on both source and target sides can be obtained, we firstly impose lexical constraints (by word replacement, as shown in Figure 1) on target data for the inference of back-translation. The constrained words will be forced to translate, so that the lexical knowledge is introduced. Then, we utilize round-trip iterations to incorporate more lexical knowledge into the pseudo-parallel data.

To exert more targeted constraints on BT, we further propose two sampling strategies, one is to select domain-specific words by domain difference, and the other is to select poorly translated words in BT by confidence estimation. During our BT process, we preferentially constrain these two types of words, so that more significant lexical knowledge can be incorporated.

The main contributions of this paper can be summarized as follows:

- We are the first to apply lexical constraints to BT, and propose an Iterative Constrained Back-Translation (ICBT) method to improve the unsupervised domain adaptation of NMT.
- To create more targeted lexical constraints, we propose two strategies for sampling constrained words, via domain specificity and confidence estimation. Experiments show that the two strategies are complementary.
- We conduct experiments on four domains of the English-Chinese public datasets. The experimental results show that our method claimed improvement in all domains, with a maximum of +8.45 BLEU scores over the strongest baseline and a maximum of +32.33 BLEU scores over the unadapted model. Besides, the translation accuracy of in-domain lexicons is improved by up to 7.99%.

2 Related Work

2.1 Supervised Domain Adaptation of NMT

If a small number of in-domain parallel data can be obtained, the domain adaptation of NMT can be performed in a supervised manner. The easiest way is to directly fine-tune a model pre-trained on an out-of-domain corpus with a small amount of in-domain parallel data (Luong and Manning, 2015; Freitag and Al-Onaizan, 2016). Based on this, Khayrallah et al. (2018) add additional items to minimize the cross-entropy between the output word distribution of the model and the out-of-domain model. Gu et al. (2019) model domain-specific information and enhance the performance of translation through adversarial training. Gu and Feng (2020) address the catastrophic forgetting problem of domain adaptation by freezing some module or neuron.

2.2 Unsupervised Domain Adaptation of NMT

Unlike in-domain parallel data, the in-domain monolingual data is much easier to obtain, which makes the research on unsupervised domain adaptation of NMT increasingly popular.

The unsupervised domain adaptation of NMT is mainly divided into model-based and data-based methods (Chu and Wang, 2018). Some model-based approaches introduce language models (Gulcehre et al. (2015); Dou et al. (2019b)) or auto-encoders (Cheng et al., 2016) for NMT models. Other studies introduce domain and task embedding learners during training (Dou et al., 2019a), or extend back-translation with additional Domain-Repaired models (Wei et al., 2020).

Among the data-based approaches, some studies generate pseudo-parallel data by back-translation (Sennrich et al., 2016a) and copy-based methods (Currey et al., 2017). Alternatively, Aharoni and Goldberg (2020) select domain-appropriate data from a common corpus through a self-supervised language model. In addition, some other studies use in-domain lexical knowledge to help domain adaptation. Hu et al. (2019) use lexical induction to generate dictionary, and perform word-by-word translation to generate pseudo-parallel data. Pourdamghani et al. (2019) utilize word-by-word trans-

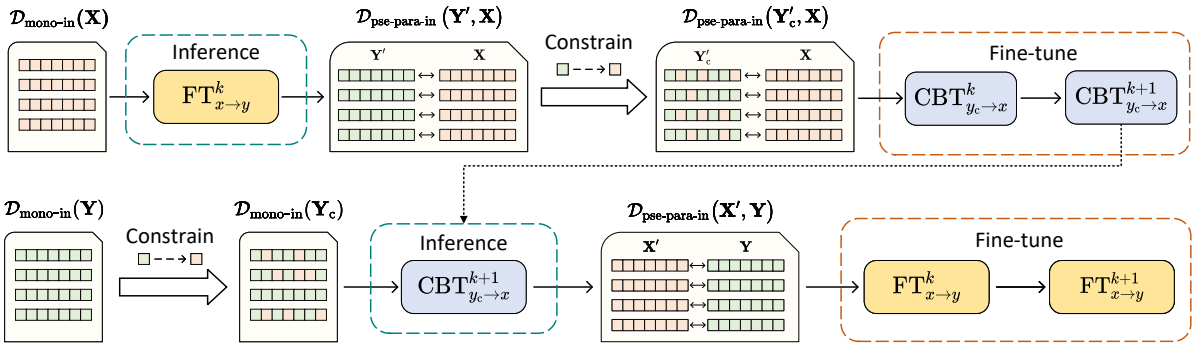

Figure 2: The schematic of our proposed ICBT method at iteration k; $\mathcal{D}_{\text{mono-in}}(\cdot)$ represents the in-domain monolingual corpus, $D_{\text{pse-para-in}}(\cdot)$ represents the in-domain pseudo-parallel sentence pairs, and $\mathbf{X}$ and $\mathbf{Y}$ represent the source and target language, respectively.

lation to generate *translationese*, and then translate *translationese* into fluent sentences.

In this work, we focus on the data-based unsupervised domain adaptation of NMT and propose a method to improve the ability of back-translation (BT) (Sennrich et al., 2016a) in domain adaptation. There are several similar studies, Wang et al. (2019) use confidence estimation to better handle the noise of synthetic corpus, Hoang et al. (2018) generate better synthetic parallel data by iterating forward and backward translation models, and Kumari et al. (2021) improve iterative back-translation by introducing classifiers to filter the synthetic data. But they only focus on utilizing the sentence-structural knowledge of BT and pay less attention to the use of lexical knowledge from in-domain monolingual data, which may lead to the mistranslation of unseen in-domain words.

3 Our Approach

The Iterative Constrained Back-Translation (ICBT) method proposed in this paper aims to introduce the lexical knowledge from monolingual data into back-translation. In this section, we first describe how to constrain back-translation and perform round-trip iterations (§ 3.1), and then we introduce two strategies for sampling more targeted constrained words (§ 3.2).

3.1 Iterative Constrained Back-Translation

3.1.1 Lexically Constrained Back-Translation

We perform lexical constraints by replacing target constrained words with their source corresponding words in the bilingual dictionary. When training the constrained back-translation, the model learns to directly copy the constraints into the translated sentences, so that during the inference, constrained words are forced to translate to their corresponding words (Song et al., 2019).

Supposing that two in-domain (IND) monolingual corpora ($\mathcal{D}_{\text{mono-in}}(\mathbf{X})$ and $\mathcal{D}_{\text{mono-in}}(\mathbf{Y})$) and an out-of-domain (OOD) parallel corpus ($\mathcal{D}_{\text{para-out}}(\mathbf{X}, \mathbf{Y})$) can be obtained in the unsupervised scenario, we firstly constrain the target of OOD parallel data to generate $\mathcal{D}_{\text{para-out}}(\mathbf{X}, \mathbf{Y}_{\text{c}})$. We use $\mathcal{D}_{\text{para-out}}(\mathbf{X}, \mathbf{Y})$ to train a source-to-target forward translation model $\text{FT}^{0}_{x\to y}$, and use $\mathcal{D}_{\text{para-out}}(\mathbf{X}, \mathbf{Y}_{\text{c}})$ to train a target-to-source constrained back-translation model $\text{CBT}^{0}_{y_{\text{c}}\to x}$.

Then, we match each target sentence y ($y \in \mathcal{D}_{\text{mono-in}}(\mathbf{Y})$) with the in-domain bilingual dictionary V and select words to be constrained, thereby generating $\mathcal{D}_{\text{mono-in}}(\mathbf{Y}_{\text{c}})$. Since the dictionary matches a large proportion of words, and too many replacement words may destroy the syntactic structure of the sentence (see detailed analysis in § 5.3), we limit the number of replacements per sentence to n. If there are more than n matched words, they will be randomly sampled n for replacement.

Finally, $\mathcal{D}_{\text{mono-in}}(\mathbf{Y}_{\text{c}})$ is fed into the pre-trained constrained back-translation (CBT) model for inference. In this way, we can get the pseudo-parallel data $\mathcal{D}_{\text{pse-para-in}}(\mathbf{X}', \mathbf{Y})$, which will be used to fine-tune the forward translation (FT) model.

Since the constrained lexicon pairs are integrated into the inference results, the FT model can learn the in-domain lexical knowledge. On the contrary, BT without constraints can hardly provide in-domain lexical knowledge.

3.1.2 Round-Trip Iteration

To further encourage the integration of lexical knowledge, we utilize an iterative translation process, as illustrated in Algorithm 1. For intuitive-

ness, we also show a schematic diagram of the round-trip iteration in Figure 2.

The iteration starts with models ($\mathrm{FT}^0_{x\to y}$ and $\mathrm{CBT}^0_{y_c\to x}$) pre-trained with out-of-domain parallel data, and takes the in-domain monolingual data ($\mathcal{D}_{\text{mono-in}}(\mathbf{X})$ and $\mathcal{D}_{\text{mono-in}}(\mathbf{Y})$) as well as the in-domain bilingual dictionary V as input. The CBT model is first fine-tuned with pseudo-parallel data $\mathcal{D}_{\text{pse-para-in}}(\mathbf{X}, \mathbf{Y}'_c)$, and then the FT model is fine-tuned with pseudo-parallel data $\mathcal{D}_{\text{pse-para-in}}(\mathbf{X}', \mathbf{Y})$.

During iteration, CBT produces pseudo-parallel data with in-domain lexical knowledge, which will be integrated with sentence-structural knowledge when fine-tuning FT. So that FT produces target sentences more in line with the domain. And correspondingly, there will be more domain-related lexical constraints on these target sentences, which is also beneficial for fine-tuning CBT.

Algorithm 1 Round-trip iterative training process for ICBT

Input: pre-trained NMT models $\mathrm{FT}^0_{x\to y}$ and $\mathrm{CBT}^0_{y_c\to x}$), in-domain monolingual data $\mathcal{D}_{\text{mono-in}}(\mathbf{X})$ and $\mathcal{D}_{\text{mono-in}}(\mathbf{Y})$, bilingual dictionary V, maximum number of iterations K

Output: forward translation model $\mathrm{FT}^K_{x\to y}$

$k = 0$;
for $k < K$ **do**
  Fine-tune the CBT model:
  Use $\mathrm{FT}^k_{x\to y}$ to infer $\mathcal{D}_{\text{mono-in}}(\mathbf{X})$ and create the pseudo data $\mathcal{D}_{\text{pse-para-in}}(\mathbf{Y}', \mathbf{X})$;
  Constrain on the target of the pseudo data with V to generate $\mathcal{D}_{\text{pse-para-in}}(\mathbf{Y}'_c, \mathbf{X})$;
  Use $\mathcal{D}_{\text{pse-para-in}}(\mathbf{Y}'_c, \mathbf{X})$ to fine-tune $\mathrm{CBT}^k_{y_c\to x} \Rightarrow \mathrm{CBT}^{k+1}_{y_c\to x}$;
  Fine-tune the FT model:
  Constrain on the target monolingual data $\mathcal{D}_{\text{mono-in}}(\mathbf{Y})$ with V to get $\mathcal{D}_{\text{mono-in}}(\mathbf{Y}_c)$;
  Use $\mathrm{CBT}^{k+1}_{y_c\to x}$ to infer $\mathcal{D}_{\text{mono-in}}(\mathbf{Y}_c)$, and get the pseudo data $\mathcal{D}_{\text{pse-para-in}}(\mathbf{X}', \mathbf{Y})$;
  Use $\mathcal{D}_{\text{pse-para-in}}(\mathbf{X}', \mathbf{Y})$ to fine-tune $\mathrm{FT}^k_{x\to y} \Rightarrow \mathrm{FT}^{k+1}_{x\to y}$;
  $k = k + 1$;
end for
return $\mathrm{FT}^K_{x\to y}$

3.1.3 Bilingual Dictionary Induction

Since the bilingual dictionary is required for lexical constraints, we obtain them through automatic induction. Considering there are no IND parallel corpora, we use unsupervised lexical induction to create a bilingual dictionary.

Unsupervised lexical induction aims to extract dictionaries from non-parallel data automatically. The mainstream approach is to map the source and target word embeddings to the same representation space and find words with close distances in the cross-lingual space as translation candidates.

In this paper, we first use $\mathcal{D}_{\text{para-out}}(\mathbf{X}, \mathbf{Y})$, $\mathcal{D}_{\text{mono-in}}(\mathbf{X})$, and $\mathcal{D}_{\text{mono-in}}(\mathbf{Y})$ to train word embeddings in the source and target languages by *FastText* (Bojanowski et al., 2017). Then we follow Artetxe et al. (2018) to build cross-lingual embedding representations by self-learning, and find the nearest neighbors of the source and target word.

3.2 Constrained Lexicon Sample Strategy

In the lexical constraints process in § 3.1, matched words are randomly sampled, which leads to some common words (such as "the", "she", "this") being constrained. So we explore delicate strategies for sampling constrained words, based on two standards: domain specificity (§ 3.2.1) and translation confidence (§ 3.2.2).

3.2.1 Domain Specificity

As shown in the ICBT-DomainSpec method of Figure 3, we use masked language models (MLMs) to calculate domain difference to help us judge the domain specificity of words. Specifically, we follow Devlin et al. (2019) to train an out-of-domain MLM (MLM_{out}) and an in-domain MLM (MLM_{in}) using the target of the out-of-domain parallel data and the in-domain target monolingual data, respectively. Assuming that the dictionary matches the set of words $\boldsymbol{w} = \{w_1, w_2, \ldots, w_m\}$ in the sentence $\boldsymbol{y} = \{y_1, y_2, \ldots, y_n\} (\boldsymbol{y} \in \mathcal{D}_{\text{mono-in}}(\mathbf{Y}))$, for each word $w \in \boldsymbol{w}$, we perform the following operations:

- Mask y_i ($y_i = w$) in $\boldsymbol{y}$ to creat $\boldsymbol{y}^{mask} = \{y_1, y_2, \ldots, y_{i-1}, [\text{mask}], y_{i+1}, \ldots, y_n\}$.
- Feed $\boldsymbol{y}^{mask}$ into MLM_{out} and MLM_{in} respectively, and obtain the outputs $\boldsymbol{y}_t^{d_{out}}$ and $\boldsymbol{y}_t^{d_{in}}$ of the two models. The probability of predicting the [mask] position as word w is defined as:

$$p_d = \log p\left(\boldsymbol{y}_t^d[i] = w | \boldsymbol{y}^{mask}\right) \quad (1)$$

  where d represents domain, $d \in \{d_{out}, d_{in}\}$.
- Calculate the probability difference:

$$\Delta p = p_{d_{in}} - p_{d_{out}} \quad (2)$$

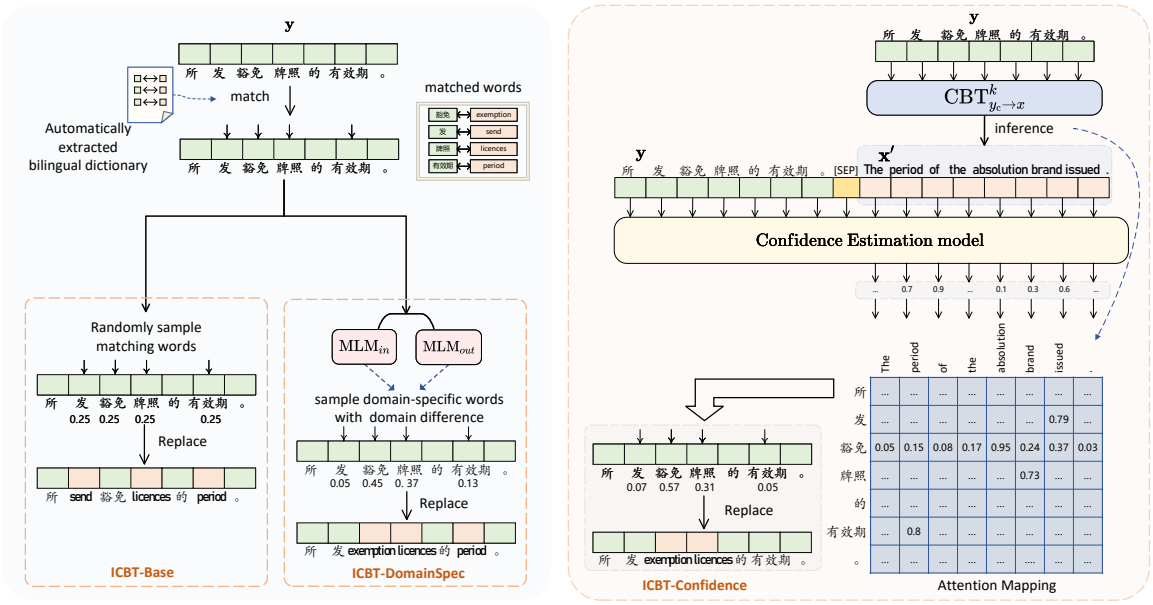

Figure 3: Different word constraints sampling methods proposed in this paper. We introduced three forms, namely ICBT-Base for baseline (§ 3.1), ICBT-DomainSpec for domain-specific words (§ 3.2.1), and ICBT-Confidence for poorly translated words (§ 3.2.2). In the attention matrix, value $w_{i,j}$ represents the correlation between the i-th word in the input sentence and the j-th word in the predicted sentence, for example, $w_{3,5}$ represents the correlation between the word "豁免 (which means exemption)" and the word "absolution" is 0.95.

We use Δp as the criterion for judging the domain specificity of the word w. A large Δp means that the word has a strong domain specificity. When constraining, we replace the top-n words with the largest Δp.

3.2.2 Confidence Estimation

Confidence estimation is utilized to select words that are poorly translated in the target monolingual data ($\mathcal{D}_{\text{mono-in}}(\mathbf{Y})$) during back-translation inference. Generally, these words also need to be supplemented by lexical knowledge.

As the ICBT-Confidence method shown in Figure 3, we first use a masked language model to perform confidence estimation on words in source translation data ($\mathcal{D}_{\text{pse-in}}(\mathbf{X}')$), following Zheng et al. (2021). Specifically, OOD parallel data is used to train an estimation model. We splice each source and target sentence pair of the parallel data, mask some words on the target, then feed them into the estimation model, and minimize the following loss function:

$$\mathcal{L} = -\sum_{n=1}^{N} \log p\left(\boldsymbol{y}_n^m | \boldsymbol{x}_n, \boldsymbol{y}_n^o; \boldsymbol{\theta}\right) \quad (3)$$

where $\boldsymbol{y}_n^m$ is the masked part of the target sentence, and $\boldsymbol{y}_n^o$ is the unmasked part, $\boldsymbol{x}_n$ is the source sentence, N is the number of OOD parallel sentence pairs, and $\boldsymbol{\theta}$ is the model parameter.

During inference, the target monolingual sentence $\boldsymbol{y}$ ($\boldsymbol{y} \in \mathcal{D}_{\text{mono-in}}(\mathbf{Y})$) and the source predicted sentence $\boldsymbol{x}$ ($\boldsymbol{x} \in \mathcal{D}_{\text{pse-in}}(\mathbf{X}')$) are spliced and fed into the pre-trained estimation model. The estimation model calculates the probability that words in $\boldsymbol{x}$ can be recovered after being masked. The higher the recovery probability is, the higher the confidence score of the word is, and vice versa. In this way, the confidence score of each word in the source predicted data is obtained.

Then we obtain the confidence score of each word in the target monolingual data through the attention mapping, which is obtained from the penultimate layer (proved by Garg et al. (2019) to be more inclined to learn alignment) of the back-translation model. More specifically, for a sentence $\boldsymbol{x}$ of the predicted data, the confidence score of each word $x_j \in \boldsymbol{x}$ is denoted as s_{x_j}. The confidence score s_{y_i} of the word y_i in the input sentence $\boldsymbol{y}$ is calculated by:

$$s_{y_i} = \sum_{j=1}^{N} w_{i,j} \times s_{x_j} \quad (4)$$

where N is the length of the predicted sentence $\boldsymbol{x}$, $w_{i,j}$ is a value in attention mapping (see the example at Figure 3), which is taken as the weight, in other words, the contribution of each s_{x_j} to s_{y_i}.

	Education	Laws	Science	Thesis	Average
without iteration					
Unadapted	10.52	23.88	6.00	6.67	11.77
Back-Translation (Sennrich et al., 2016a)	12.71	32.70	6.62	11.22	15.81
Translationese (Pourdamghani et al., 2019)	12.79	32.61	6.27	11.18	15.71
CBT-Base (*ours*)	14.25†	37.05†	7.17†	12.12†	17.65
CBT-DomainSpec (*ours*)	**14.65**†	**38.41**†	7.19†	**12.48**†	**18.18**
CBT-Confidence (*ours*)	14.30†	38.11†	**7.23**†	12.29†	17.98
with iteration					
Iterative Back-Translation (Hoang et al., 2018)	14.63	47.56	8.25	13.31	20.94
CFIBT (Kumari et al., 2021)	14.84	47.30	8.80	12.79	20.93
ICBT-Base (*ours*)	14.91†	49.51†	**8.91**†	14.05†	21.85
ICBT-DomainSpec (*ours*)	16.11†	52.12†	8.62†	14.76†	22.90
ICBT-Confidence (*ours*)	**16.36**†	52.79†	7.81†	14.70†	22.92
ICBT-ALL (*ours*)	16.10†	**56.21**†	8.50†	**15.25**†	**24.02**

Table 1: Comparative results of unsupervised domain adaptation on the English-Chinese translation task. *News* is used as the out-of-domain data. "CBT-*" are our methods without iteration. † denotes the improvement over other methods is statistically significant with $p < 0.01$.

Domain	Train	Dev	Test
News	1,252,977	1,664	1,357
Education	447,000	3,000	790
Laws	217,000	3,000	456
Science	267,000	3,000	503
Thesis	297,000	3,000	625

Table 2: Corpus statistics for our experiments.

After getting the confidence score of each word in the target, we choose the top-n words that have the lowest score and are in the domain dictionary to be constrained.

4 Experiments

4.1 Setup

Datasets. We conduct our experiments on English-Chinese datasets. The parallel corpus LDC[1] in *News* domain is used as the out-of-domain dataset. For in-domain dataset, we use the UM-corpus (Tian et al., 2014), which provides parallel sentence pairs in eight domains, and we choose four domains of *Laws*, *Education*, *Science*, and *Thesis* to conduct experiments. To obtain the in-domain development set, we randomly sample 3K sentence pairs in each domain. Data statistics are shown in Table 2. We follow Hu et al. (2019) to construct a non-parallel monolingual corpus for each domain. Specifically, we randomly divide the parallel corpus into two equal parts, and take the source sentences of the former part and the target sentences of the latter part as our monolingual data.

Jieba[2] is used to segment Chinese data and Moses[3] is used to segment English. And all English words are converted to lowercase. After that, we segment words into subwords through Byte Pair Encoding (Sennrich et al., 2016b) and construct joint vocabulary for both languages.

Models and Parameters. We implement the Transformer$_{\text{base}}$ (Vaswani et al., 2017) based on the Fairseq (Ott et al., 2019) as our translation model. BERT-base-Chinese[4] model is used as the masked language model in the ICBT-DomainSpec method. To get the in-domain model MLM_{in} and the out-of-domain model MLM_{out}, we fine-tune the BERT on two kinds of data for 5 epochs respectively. Moreover, we use multilingual BERT[4] (mBERT) as the confidence estimation model for the ICBT-Confidence method. For each method, we conduct 3 iterations. SacreBLEU[5] python package is used to calculate the BLEU score.

Baselines. We compare our method with the following methods:

- **Unadapted**. The translation model is trained on the out-of-domain training set and directly evaluated on the in-domain test set.

[1] https://www.ldc.upenn.edu/

[2] https://github.com/fxsjy/jieba

[3] http://www.statmt.org/moses/

[4] https://huggingface.co

[5] https://github.com/mjpost/sacrebleu

- **Back-Translation** (Sennrich et al., 2016a). A method for generating in-domain pseudo-parallel data from a target-to-source NMT model and target monolingual data.

- **Iterative Back-Translation** (Hoang et al., 2018). Both forward and backward translation models are used for round-trip iteration to optimize and generate better in-domain pseudo-parallel data.

- **Translationese** (Pourdamghani et al., 2019). This method first uses word-by-word translation to generate *translationese*, and then generates fluent translation sentences using *translationese*. It improves translation performance by introducing in-domain lexical knowledge.

- **CFIBT** (Kumari et al., 2021). Classifier models are introduced to filter pseudo-parallel data generated by the back-translation, thereby optimizing the iterative back-translation.

4.2 Main Results

This paper introduces three forms of ICBT, namely ICBT-Base (§ 3.1), ICBT-DomainSpec (§ 3.2.1), and ICBT-Confidence (§ 3.2.2). To prove the complementarity of our methods, we combine the data obtained from both ICBT-DomainSpec and ICBT-Confidence at each iteration to conduct experiments (ICBT-ALL). Furthermore, we also compare the performance of our method without iteration. The experimental results are shown in Table 1.

Firstly, our proposed methods achieve optimal results in all domains. The average of four domains improves by up to 12.25 BLEU over the unadapted model and 3.08 BLEU over the strongest baseline. The most considerable improvement is in *Laws*, which improves by up to 8.45 BLEU over the strongest baseline. We believe that it is because *Laws* includes more terms needed to be supplemented, so the introduction of lexical knowledge can bring a significant improvement.

Secondly, *Translationese* achieves comparable performance to *Back-Translation* and has improved in all domains compared with the unadapted baseline, illustrating the benefits of introducing in-domain lexical knowledge. Our method is better than *Translationese*, which we believe is because we fuse lexical knowledge and sentence-structural knowledge through lexically constrained BT, while *Translationese* uses knowledge separately.

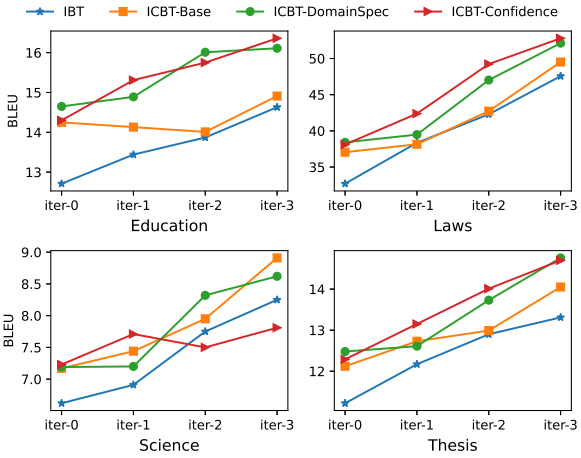

Figure 4: BLEU score of different methods according to the number of iterations on the test sets of four domains. IBT represents *Iterative Back-Translation*.

Thirdly, CFIBT achieves better performance than *iterative back-translation* (IBT) in *Education* and *Science*, and these two methods are comparable on the whole. This is in keeping with the elaboration of Kumari et al. (2021) on the performance of CFIBT in high-resource scenarios. Our methods have the improvement over both approaches. Compared to the case without iteration, our methods achieve more significant improvements over others, implying that more lexical knowledge can be incorporated through iterations.

Finally, both ICBT-DomainSpec and ICBT-Confidence achieve higher results than ICBT-Base. We conclude that it is because more targeted constraints can introduce more lexical knowledge lacking by models. ICBT-ALL achieves the best performance and obtains more significant improvement than ICBT-DomainSpec and ICBT-Confidence, especially in *Laws*. It shows that domain specificity and confidence estimation are complementary.

5 Analysis

5.1 Impact of Iterations

To further investigate the impact of iterations, we separately validate the models of each method after each iteration. The variation of BLEU scores with the number of iterations is shown in Figure 4.

Our methods show continuous improvement over IBT and the improvement is larger in the later iterations in general. We conjecture the main reasons are as follows: 1) Lexical constraints as semantic spatial anchors bring the lexical-level representations closer (Lin et al., 2020). The alignment of lexical-level makes sentence-level representa-

	Laws		Thesis	
	BLEU	$\text{Trans}_{\text{acc}}$	BLEU	$\text{Trans}_{\text{acc}}$
BT	32.70	79.85%	11.22	44.39%
CBT	**37.05**	**82.58%**	**12.12**	**49.71%**
IBT	47.56	84.99%	13.31	51.58%
ICBT	**49.51**	**88.46%**	**14.05**	**59.57%**

Table 3: Comparison of BLEU score and lexicon translation accuracy of BT and constrained BT. CBT and ICBT are the base type of our methods (§ 3.1).

tions in BT also closer in semantic space, which allows the model to transfer at both levels. 2) When the sentence-level representations are aligned, more in-domain lexicons can be generated, so that the knowledge of the in-domain dictionary is more fully utilized. 3) Lexical constraints increase the diversity of constraints in the iteration. Thus in the obtained pseudo data, the same target sentence may correspond to different source sentences, which further improves the robustness of the model.

5.2 Lexicon Translation Accuracy

To verify whether constrained back-translation really helps the translation of the in-domain lexicon, we analyze the lexical translation accuracy. Concretely, we extract the high-quality in-domain bilingual dictionary from in-domain parallel data to act as the testbed for lexical translation accuracy. For each source word in the high-quality dictionary that appears at the source of the test set, we consider it to be successfully translated if its corresponding target word occurs in the predicted data.

We conduct experiments in *Laws* and *Thesis*,[6] comparing BT and CBT with and without iteration. The comparison results are shown in Table 3. With or without iteration, the lexicon translation accuracy of CBT is always higher than that of BT, with a maximum of 3.47% in *Laws* and 7.99% in *Thesis*, verifying the benefit of constraints. Besides, the improvement with iterations is more significant than without iterations, which indicates that more lexical knowledge can be introduced through iterations, so that more in-domain lexicons can be translated correctly. We also present some cases of BT and CBT in each domain in Appendix A.

[6]We focus on domains where there is room for improvement. For *Education* and *Science*, BT can already achieve high translation accuracy of lexicon, so we do not discuss.

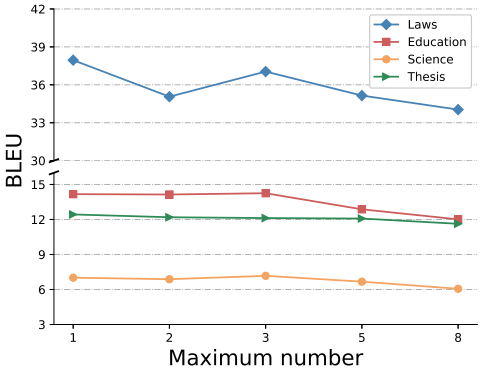

Figure 5: The effect of the maximum number of constraints on the BLEU score.

5.3 Effect of Constraints Amount

Few constraints may not achieve the desired effect, while too many constraints may destroy the syntactic structure of sentences. To validate this hypothesis, we investigate the impact of the number of constraints. Specifically, we vary the value of the maximum number n of constraints per sentence in the range of $[1, 2, 3, 5, 8]$. For each value of n, we perform CBT as described in § 3.1 and test on each resulting FT model with the test set.

As shown in Figure 5, models can achieve a high performance generally when $n \leq 3$. However, when $n = 1$ or $n = 2$, the model is unstable. For example, when $n = 1$, the model performs slightly worse on *Education* and *Science*, and when $n = 2$, it performs poorly on *Laws*. It is consistent with our conjecture that the lexical knowledge can not be introduced into the model enough when there are few constrained words. When $n > 3$, the performance deteriorates as the increase of n, indicating that too many constraints may damage the syntactic structure of sentences, thus making the performance worse. With the above analysis, we set $n = 3$ in other experiments in this paper.

6 Conclusion and Future Work

This paper proposes a method for unsupervised domain adaptation of NMT named Iterative Constrained Back-Translation (ICBT), in which lexical constraints are applied to back-translation, aiming to incorporate in-domain lexical knowledge into synthetic parallel data from BT. Besides, we propose two strategies for sampling constraints to exert more targeted constraints. We conduct experiments on English-Chinese translation tasks in four domains. The experiments show that our method can introduce beneficial lexical knowledge to BT, thus

achieving state-of-the-art results.

We believe that the lexical constraint is not only suitable for unsupervised domain adaptation, but also promising in the semi-supervised scenario with a small amount of in-domain parallel corpus. In the future, we will explore the application of lexical constraints in supervised or semi-supervised domain adaptation of NMT.

Acknowledgements

The research work descried in this paper has been supported by the National Key R&D Program of China (2020AAA0108001) and the National Nature Science Foundation of China (No. 61976016, 61976015, and 61876198). The authors also would like to thank the anonymous reviewers for their valuable comments and suggestions to improve this paper.

References

Roee Aharoni and Yoav Goldberg. 2020. Unsupervised domain clusters in pretrained language models. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7747–7763, Online. Association for Computational Linguistics.

Mikel Artetxe, Gorka Labaka, and Eneko Agirre. 2018. A robust self-learning method for fully unsupervised cross-lingual mappings of word embeddings. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 789–798, Melbourne, Australia. Association for Computational Linguistics.

Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. 2015. Neural machine translation by jointly learning to align and translate. In *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*.

Piotr Bojanowski, Edouard Grave, Armand Joulin, and Tomas Mikolov. 2017. Enriching word vectors with subword information. *Transactions of the Association for Computational Linguistics*, 5:135–146.

Yong Cheng, Wei Xu, Zhongjun He, Wei He, Hua Wu, Maosong Sun, and Yang Liu. 2016. Semi-supervised learning for neural machine translation. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1965–1974, Berlin, Germany. Association for Computational Linguistics.

Chenhui Chu and Rui Wang. 2018. A survey of domain adaptation for neural machine translation. In *Proceedings of the 27th International Conference on Computational Linguistics*, pages 1304–1319, Santa Fe, New Mexico, USA. Association for Computational Linguistics.

Anna Currey, Antonio Valerio Miceli Barone, and Kenneth Heafield. 2017. Copied monolingual data improves low-resource neural machine translation. In *Proceedings of the Second Conference on Machine Translation*, pages 148–156, Copenhagen, Denmark. Association for Computational Linguistics.

Hal Daumé III and Jagadeesh Jagarlamudi. 2011. Domain adaptation for machine translation by mining unseen words. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, pages 407–412, Portland, Oregon, USA. Association for Computational Linguistics.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, Minneapolis, Minnesota. Association for Computational Linguistics.

Zi-Yi Dou, Antonios Anastasopoulos, and Graham Neubig. 2020. Dynamic data selection and weighting for iterative back-translation. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 5894–5904, Online. Association for Computational Linguistics.

Zi-Yi Dou, Junjie Hu, Antonios Anastasopoulos, and Graham Neubig. 2019a. Unsupervised domain adaptation for neural machine translation with domain-aware feature embeddings. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 1417–1422, Hong Kong, China. Association for Computational Linguistics.

Zi-Yi Dou, Xinyi Wang, Junjie Hu, and Graham Neubig. 2019b. Domain differential adaptation for neural machine translation. In *Proceedings of the 3rd Workshop on Neural Generation and Translation*, pages 59–69, Hong Kong. Association for Computational Linguistics.

Markus Freitag and Yaser Al-Onaizan. 2016. Fast domain adaptation for neural machine translation. *ArXiv preprint*, abs/1612.06897.

Sarthak Garg, Stephan Peitz, Udhyakumar Nallasamy, and Matthias Paulik. 2019. Jointly learning to align and translate with transformer models. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 4453–4462, Hong Kong, China. Association for Computational Linguistics.

Shuhao Gu and Yang Feng. 2020. Investigating catastrophic forgetting during continual training for neural machine translation. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 4315–4326, Barcelona, Spain (Online). International Committee on Computational Linguistics.

Shuhao Gu, Yang Feng, and Qun Liu. 2019. Improving domain adaptation translation with domain invariant and specific information. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 3081–3091, Minneapolis, Minnesota. Association for Computational Linguistics.

Caglar Gulcehre, Orhan Firat, Kelvin Xu, Kyunghyun Cho, Loic Barrault, Huei-Chi Lin, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. 2015. On using monolingual corpora in neural machine translation. *ArXiv preprint*, abs/1503.03535.

Vu Cong Duy Hoang, Philipp Koehn, Gholamreza Haffari, and Trevor Cohn. 2018. Iterative back-translation for neural machine translation. In *Proceedings of the 2nd Workshop on Neural Machine Translation and Generation*, pages 18–24, Melbourne, Australia. Association for Computational Linguistics.

Junjie Hu, Mengzhou Xia, Graham Neubig, and Jaime Carbonell. 2019. Domain adaptation of neural machine translation by lexicon induction. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 2989–3001, Florence, Italy. Association for Computational Linguistics.

Huda Khayrallah, Brian Thompson, Kevin Duh, and Philipp Koehn. 2018. Regularized training objective for continued training for domain adaptation in neural machine translation. In *Proceedings of the 2nd Workshop on Neural Machine Translation and Generation*, pages 36–44, Melbourne, Australia. Association for Computational Linguistics.

Philipp Koehn and Rebecca Knowles. 2017. Six challenges for neural machine translation. In *Proceedings of the First Workshop on Neural Machine Translation*, pages 28–39, Vancouver. Association for Computational Linguistics.

Surabhi Kumari, Nikhil Jaiswal, Mayur Patidar, Manasi Patwardhan, Shirish Karande, Puneet Agarwal, and Lovekesh Vig. 2021. Domain adaptation for NMT via filtered iterative back-translation. In *Proceedings of the Second Workshop on Domain Adaptation for NLP*, pages 263–271, Kyiv, Ukraine. Association for Computational Linguistics.

Zehui Lin, Xiao Pan, Mingxuan Wang, Xipeng Qiu, Jiangtao Feng, Hao Zhou, and Lei Li. 2020. Pre-training multilingual neural machine translation by leveraging alignment information. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 2649–2663, Online. Association for Computational Linguistics.

Minh-Thang Luong and Christopher Manning. 2015. Stanford neural machine translation systems for spoken language domains. In *Proceedings of the 12th International Workshop on Spoken Language Translation: Evaluation Campaign*, pages 76–79, Da Nang, Vietnam.

Myle Ott, Sergey Edunov, Alexei Baevski, Angela Fan, Sam Gross, Nathan Ng, David Grangier, and Michael Auli. 2019. fairseq: A fast, extensible toolkit for sequence modeling. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics (Demonstrations)*, pages 48–53, Minneapolis, Minnesota. Association for Computational Linguistics.

Nima Pourdamghani, Nada Aldarrab, Marjan Ghazvininejad, Kevin Knight, and Jonathan May. 2019. Translating translationese: A two-step approach to unsupervised machine translation. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 3057–3062, Florence, Italy. Association for Computational Linguistics.

Rico Sennrich, Barry Haddow, and Alexandra Birch. 2016a. Improving neural machine translation models with monolingual data. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 86–96, Berlin, Germany. Association for Computational Linguistics.

Rico Sennrich, Barry Haddow, and Alexandra Birch. 2016b. Neural machine translation of rare words with subword units. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1715–1725, Berlin, Germany. Association for Computational Linguistics.

Kai Song, Yue Zhang, Heng Yu, Weihua Luo, Kun Wang, and Min Zhang. 2019. Code-switching for enhancing NMT with pre-specified translation. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 449–459, Minneapolis, Minnesota. Association for Computational Linguistics.

Liang Tian, Derek F. Wong, Lidia S. Chao, Paulo Quaresma, Francisco Oliveira, Yi Lu, Shuo Li, Yiming Wang, and Longyue Wang. 2014. UM-corpus: A large English-Chinese parallel corpus for statistical machine translation. In *Proceedings of the Ninth International Conference on Language Resources and Evaluation (LREC'14)*, pages 1837–1842, Reykjavik, Iceland. European Language Resources Association (ELRA).

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, December 4-9, 2017, Long Beach, CA, USA*, pages 5998–6008.

Shuo Wang, Yang Liu, Chao Wang, Huanbo Luan, and Maosong Sun. 2019. Improving back-translation with uncertainty-based confidence estimation. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 791–802, Hong Kong, China. Association for Computational Linguistics.

Hao-Ran Wei, Zhirui Zhang, Boxing Chen, and Weihua Luo. 2020. Iterative domain-repaired back-translation. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 5884–5893, Online. Association for Computational Linguistics.

Yuanhang Zheng, Zhixing Tan, Meng Zhang, Mieradilijiang Maimaiti, Huanbo Luan, Maosong Sun, Qun Liu, and Yang Liu. 2021. Self-supervised quality estimation for machine translation. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 3322–3334, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

A Case Study

We feed one case from each domain into the final fine-tuned forward translation models of BT, CBT, IBT, and ICBT for inference, and the obtained results are shown in Table 4. It can be seen that the constrained BT outperforms the BT in the translation of in-domain words, no matter with or without iteration. In the case of *Education*, the ordinary BT translates "spring" to "春天", which does not fit the current context. On the contrary, constrained BT can correctly translate "spring" to "弹簧". It shows that lexical constraints can introduce in-domain lexical knowledge into the synthetic pseudo-parallel data.

From the cases of *Laws* and *Thesis*, the performance of ICBT is stronger than that of CBT, and it can translate more in-domain words. For example, in the case of *Laws*, CBT only successfully translates "public office" into "公职", but does not successfully translate "appoint or remove", which is a strongly domain-specific expression. But through iteration, the in-domain lexical knowledge and specific expressions are further enriched, so ICBT successfully translates "appoint or remove" into "任免". In addition, ICBT can also learn the domain-specific expression of translating "holders of public office" into "公职 人员" under the promotion of lexical knowledge "公职", indicating that lexical knowledge can not only bring alignment knowledge, but also enrich the sentence-structural knowledge of BT, so that the generated sentences are more in line with the domain expression.

	Education
Source	deformation of a spring is linear with the force applied on it.
Reference	弹簧 的 变形 量 与 施加 在 其 上 的 作用力 成 线性关系 。
BT	春天 变形 是 线性 的 , 应用 于 它 的 力量 。
CBT	弹簧 变形 与 应用 的 力量 是 线性 的 。
IBT	一个 春天 的 变形 与 应用 在 它 上 的 力量 是 线性 的 。
ICBT	弹簧 的 变形 与 应用 在 它 上 的 力量 是 线性 的 。
	Laws
Source	to appoint or remove holders of public office in accordance with legal procedures;
Reference	依照 法定程序 任免 公职 人员 ；
BT	按照 法定程序 委任 或 免任 公用 处所 的 持有人 ;
CBT	依照 法定程序 委任 或 免除 公职 人员 ；
IBT	依照 法定程序 委任 或 罢免 公职 的 持有人 ;
ICBT	依照 法定程序 任免 公职 人员 ；
	Science
Source	only when its melting-point temperature is reached does iron start to pass into a liquid.
Reference	只有 当 熔点 温度 达到 时 ， 铁 才 开始 变成 液体 。
BT	只有 当 它 融化 的 温度 达到 时 , 才 会 有 铁的开始 , 进入 液态 。
CBT	当 熔点 温度 达到 时 , 铁 开始 进入 液体 。
IBT	只有 当 它 的 墨点 温度 达到 时 ， 铁的开始 传入 液体 。
ICBT	只有 当 它 的 熔点 温度 达到 时 ， 铁 才 会 开始 传入 液体 。
	Thesis
Source	effect of exciting frequency on the residual stress of the vibrating solidification casting
Reference	激振 频率 对 振动 凝固 铸件 残余 应力 的 影响
BT	振动 稳态 洞穴 的 振动 频度 对 振动 稳态 洞室 残余 压力 的 影响
CBT	振动 频率 对 振动 凝固 铸件 残余 应力 的 影响
IBT	激发 频率 对 振动 固化 铸件 残余 应力 的 影响
ICBT	激振 频率 对 振动 凝固 铸件 残余 应力 的 影响

Table 4: The translation examples show the effect of constrained back-translation. We identify aligned words in red and blue front.

Linguistically-Motivated Yorùbá-English Machine Translation

Ife Adebara[1,2] **Muhammad Abdul Mageed**[1,2] **Miikka Silfverberg**[1]
Department of Linguistics[1]
Deep Learning and Natural Language Processing Group[2]
The University of British Columbia
{ife.adebara, muhammad.mageed, miikka.silfverberg}@ubc.ca

Abstract

Translating between languages where certain features are marked morphologically in one but absent or marked contextually in the other is an important test case for machine translation. When translating into English which marks (in)definiteness morphologically, from Yorùbá which uses bare nouns but marks these features contextually, ambiguities arise. In this work, we perform fine-grained analysis on how an SMT system compares with two NMT systems (BiLSTM and Transformer) when translating *bare nouns* in Yorùbá into English. We investigate how and to what extent the systems identify BNs, correctly translate them, and compare with human translation patterns. We also analyze the type of errors each model makes and provide a linguistic description of these errors. We glean insights for evaluating model performance in low-resource settings. In translating bare nouns, our results show the transformer model outperfoms the SMT and BiLSTM models for 4 categories, the BiLSTM outperforms the SMT model for 3 categories while the SMT outperforms the NMT models for 1 category.

1 Introduction

Languages differ with regard to how grammatical information such as “case" and “number" are expressed. In some languages, this information is overtly marked using morphological or syntactic means, whereas in others it has to be inferred from context. This asymmetry of information representation poses an important problem for machine translation (Mitkov, 1999; Hardmeier, 2012). Several phenomena where asymmetry arises have been identified as challenging problems for machine translation. These include: pronoun translation and coreference (Guillou et al., 2019), politeness (Sennrich et al., 2016), lexical cohesion (Carpuat, 2009), and lexical disambiguation (Gonzales et al., 2017).

Asymmetry of information representation presents an interesting test case for MT systems because it can shed light on their true linguistic ability (Voita et al., 2018; Bawden et al., 2017). It is, therefore, important to use evaluation measures which can capture this aspect of the translation task. However, the most popular evaluation metric for machine translation, the BLEU score (Papineni et al., 2002a), is a coarse metric which can often hide these fine-grained morphological and semantic distinctions. In fact, a high BLEU score is no guarantee of improved translation quality and BLEU, being based on precision on short ngrams, may be poorly suited for measuring the coherence and grammaticality of a sentence.

In this paper we investigate the performance of Yorùbá [1] to English Machine Translation. We specifically evaluate performance on translating *Bare nouns* (BNs). BNs (Cheng and Sybesma, 1999; Krifka, 2003; Chierchia, 1998; Larson, 1985; Carlson, 1989) are nouns without an overt determiner or quantifier. For instance "houses" in *Houses are expensive in New York* is a BN. Whereas English accounts for only plural BNs, BNs in Yorùbá are number neutral and can also be definite or indefinite depending on the context. Consider the following example:

(1) *Bàbá ra iṣu*
FATHER BUY YAM

a. *‘Father bought a yam.’* **(Indefinite Singular)**
b. *‘Father bought some yams.’* **(Indefinite Plural)**
c. *‘Father bought the yam.’* **(Definite Singular)**
d. *‘Father bought the yams.’* **(Definite Plural)**

In Example (1), the BN *iṣu yam* can be translated into English in four ways as: the indefinite singular *a yam*, an indefinite plural *some yams*, a

[1] Yorùbá is a tone language that belongs to the Yoruboid group of the Kwa branch of the Niger-Congo language family, which is spoken by over 40 million in Nigeria. Yorùbá is spoken primarily in western Nigeria and eastern Benin, with communities in Sierra Leone and Liberia, and expatriate communities throughout Africa, Europe, and the Americas.

Proceedings of the 29th International Conference on Computational Linguistics, pages 5066–5075
October 12–17, 2022.

definite singular *the yam*, and a definite plural *the yams*. This poses a challenge akin to anaphora resolution, as the correct translation of Yorùbá BNs can only be determined by examining the context in which the BN occurs in the source text. The context can span one or more preceding or current words, phrases, clauses or sentences. It can also include world knowledge.

Our study provides a fine grained analysis that sheds light on issues in MT that are not often discussed in main stream research. We turn away from the current research trend in massively multilingual translation systems, to this largely underexplored fine-grained aspect of the MT problem. We investigate how SMT and NMT systems compare in general, but also specifically with respect to translation of BNs. Although NMT has recently been reported to outperform SMT even in low-resource settings, these findings were reported for systems translating between somewhat similar languages (e.g., languages that belong to the same language family, have overlapping vocabulary and similar script) (Junczys-Dowmunt et al., 2016; Conneau et al., 2017; Lample et al., 2017; Wang et al., 2019) such as English → French, German → French, German → Italian; languages for which large corpora exists. In this work, we collect a new dataset and use it to test whether the same NMT advantage persists by exploring two *typologically dissimilar* languages belonging to different language families: Yorùbá and English. This work bears significance not only to Yorùbá but to analytic languages, low-resource languages, and languages that have BNs. Our contributions are as follows: **(1)** We align a new dataset for the Yorùbá and English low-resource setting. **(2)** We use our dataset to develop statistical and NMT Yorùbá → English models. **(3)** We study the linguistic ability of our models in disambiguating BNs.

The rest of the paper is organized as follows: Section 2 is a description of disambiguation patterns of Yorùbá BNs. We discuss related work in Section 3. Section 4 presents the datasets, data collection process, and preprocessing. In Section 5, we present our methods. We present results of our models and an evaluation of BN disambiguation in Section 6. We conclude in Section 7.

2 Disambiguating Yorùbá BNs

A number of contextual variables are important when disambiguating BNs in Yorùbá. These include the so-called *familiarity* and *uniqueness* conditions (Roberts, 2003; Russell, 1905; Abbott, 2006), as well as the *category of the verb* that occurs in the environment of the BN. Familiarity refers to information which is already known from the previous textual context either explicitly or through inference. In Yorùbá, a definite interpretation is permitted when the existence of the entity referred to has been established in the discourse, an indefinite occurs otherwise. In example (2), the first mention of book is indefinite, the second is definite.

(2) *Mo ra* ***ìwé*** *fún Kọ́lá. L'ọ́jọ́ kejì,*
1SG BUY BOOK FOR KỌLÁ. IN DAY SECOND
Kọ́lá ti so ***ìwé*** *nù*
KỌLÁ PERF THROW BOOK AWAY

'I bought a book for KỌLÁ. By the second day, Kọ́lá had lost the book.'

The uniqueness condition claims that there exists one and only one entity that meets the descriptive content of the BN as in example (3). This entails that the BN will be interpreted as definite and singular.

(3) ***Oòrùn*** *máa ń ràn ní* ***ọ̀sán***
SUN HAB PROG SHINE IN AFTERNOON

'The sun shines in the afternoon.'

The category of verb is also an important contextual element in correctly translating BNs. *Stative* verbs (verbs that describe the state of being or situation such as *to own* and *to feel*) introduce the *Generic* description where a noun phrase is used to refer to a whole class as in example (4). *Eventive* verbs (verbs that describe events such as *to break* and *to appear*) introduce all other disambiguation patterns as in example (1).

(4) *Ọmọ fẹ́rán* ***ajá*** ***(generic) (STATIVE)***
CHILD LOVE DOG

'Children love dogs'

3 Related work

Traditional evaluation methods like BLEU are inadequate for evaluation of fine-grained discourse phenomena in MT, and various approaches have been explored to alleviate this problem. Guillou and Hardmeier (2016) present the PROTEST pronoun translation test suite. The test suite contains

examples of pronoun translation which are known to be challenging for MT systems. Isabelle et al. (2017) present another challenge set for MT. This set consists of a number of human annotated sentences which are designed to probe a system's capacity to handle various linguistic phenomena (like EXAMPLES). Using the challenge set, Isabelle et al. (2017) present a comparison of SMT and NMT systems for English-French MT which provides a fine-grained exploration of the strengths of NMT, as well as insight into linguistic phenomena that typically present difficulties for NMT models. Tests suites that evaluate more than 100 linguistic phenomena was developed for English to German and German to English machine translation (Macketanz et al., 2021). The test sentences are given as input to the MT systems. The MT outputs are then evaluated by the set of rules which determine whether the output was correctly translated or not.

Sennrich (2016) assess the grammaticality of the output of a character-level NMT system by evaluating the MT model's capacity to correctly rank contrastive pairs of pre-existing translations, one of which is correct and the other one incorrect. This approach has also been applied to lexical disambiguation of English-German MT (Gonzales et al., 2017). Bentivogli et al. (2016) explore automatic detection and classification of translation errors based on manual post-edits of MT output both for SMT and NMT systems. They use a classification that evaluated outputs for morphological, lexical, and word order errors which was a simplification of those used in Hjerson (Popović, 2011). Hjerson detects word level error classes: morphological errors, re-ordering errors, missing words, extra words and lexical errors.

Neural models have also been evaluated for syntactic competence. For instance, Linzen et al. (2016) probe the ability of LSTM models to learn English subject-verb agreement. When provided explicit supervision, LSTMs were able to learn to perform the verb number agreement task in most cases, although their error rate increased on particularly difficult sentences. NMT systems have also been evaluated for morphological competence while translating from English to a morphologically rich language (Burlot and Yvon, 2017). Certain linguistic phenomena have also been tested across language families. For instance, for four language families: Slavic, Germanic, Finno-Ugric and Romance, the best NMT system outperformed the best phrase-based SMT system for all language directions to English (Toral and Sánchez-Cartagena, 2017). The NMT systems produced fluent and more accurate inflections and word order but performed poorly when translating long sentences (Vanmassenhove et al., 2019; Bentivogli et al., 2016). Morphologically rich languages have also been shown to have more fluent outputs (Klubička et al., 2017; Toral and Sánchez-Cartagena, 2017; Popović, 2018). These studies have used metrics such as BLEU (Papineni et al., 2002b), HTER (Snover et al., 2006), TTR (DARLEY, 1959), YULE (Yule, 2014) to evaluate the output of MT models for fluency and adequacy. MT was also evaluated for discuss phenomena where the model's capacity to exploit linguistic context for co-reference and coherence was evaluated (Bawden et al., 2018).

Evaluation frameworks like HOPE have also been used for task-oriented and human-centric evaluation for machine translation outputs (Gladkoff and Han, 2021). This framework uses professional post-editing annotations and contains commonly occuring error types and error penalty points. The error penalty points use geometric progression to reflect the severity level of errors for each translation unit.

4 Dataset

We use the Yorùbá Bible *Bíbélì Mímó ní Èdè Yorùbá Òde-Òní* (BMEYO) and the New International Version (NIV) English Bible.

4.1 Bible Data

We crawled the BMEYO Yorùbá Bible from the public website Biblica.[3] This version is the modern translation of the Yorùbá Bible and is the closest equivalent to the English NIV Bible, according to Biblica. We thus use the NIV English translation with the Bible to create our parallel data. We organize the Bible according to their verses. In Table 1, we show an example verse from our Bible dataset.

4.2 Data Preprocessing

The Yorùbá Bible is based on old Bible manuscripts. A total of 16 verses which were included in early English Bible versions like the King James Bible but which were omitted from later versions are still part of the Yorùbá Bible.

[3] https://www.Bible.com/versions/911-ycb-bibeli-mim-ni-ede-yoruba-ode-oni.

Data	Scripture
BMEYO	Ìgbà láti pa àti ìgbà láti mú láradá ìgbà àti wó lulẹ̀ àti ìgbà láti kọ̀
NIV	a time to kill and a time to heal, a time to tear down and a time to build.

Table 1: A description of Ecclesiastes Ch. 3, Verse 3 for the BMEYO. **(NIV English Bible translation)**

Data	#TOK	#SENT	#TTR
BMEYO	793,870	38,149	25.5

Table 2: A statistical description of the Yorùbá (source) data. #TOK refers to number of tokens, #SENT is number of sentences, and TTR is type token-ratio.

Therefore, we start data preprocessing by adding these missing verses into our English NIV Bible to make it equivalent with the Yorùbá text. Those verses were footnotes in the English NIV. In addition, the book of Third John has 15 verses in the Yorùbá Bible but 14 verses in the English NIV. The 15th verse in the Yorùbá is a part of 14th verse in the English Bible. As a result, we combine verse 15 into verse 14, as it is in the English NIV. The aligned dataset can be found on GitHub at **https://github.com/UBC-NLP/COLING2022**

Next, we tokenized the English data using `SpaCy`[4]. `SpaCy` currently does not provide a tokenization package for Yorùbá, so we used the whitespace tokenizer for all the Yorùbá data. We use python scripts to ensure the punctuations is appropriately tokenized. Next, we convert all words to lowercase in order to alleviate data sparsity.

We also split words using Byte Pair Encoding (BPE) (Sennrich et al., 2015). In low-resource settings, large vocabularies result in the representation of low-frequency (sub)words as BPE units which affects the ability to learn good high-dimensional representations (Barone et al., 2017). Thus, we choose smaller merge operations and varied the number of merge operations from $10,000$ to $30,000$. Finally, we split the dataset into training, validation, and test sets using an 80%-10%-10% standard split.

5 Methods

We train 3 sentence level models: SMT, BiLSTM, and Transformer models to translate from Yorùbá to English. We choose English as our target language because we are interested in analyzing how ambiguous BNs in Yorùbá are translated into English where (i.e., in English) nouns are typically marked both for number and determinacy. The hyperparameters and training procedure are described in the next subsections.

5.1 SMT

We use the `Moses`[5] statistical translation system for our SMT model. We apply tokenization, true casing, and perform word alignment on the parallel data using GIZA++ (Och and Ney, 2003). The word alignments were used to extract phrase-paired translations and calculate probability estimates (Koehn et al., 2007). We used KENLM (Heafield, 2011) to train and query a LM for English. KENLM is a library implemented for efficient language model queries, reducing both time and memory costs, and is integrated into `Moses`. The decoder uses this LM to ensure a fluent output of the target language, in our case, English. We used the validation sets of our parallel data for the final tuning process just before we perform the blind testing.

5.2 BiLSTM

We use a Sequence to Sequence (Seq2Seq) BiLSTM with attention model. Our best BiLSTM model has an embedding layer with $1,024$ dimensions, and 2 encoder and decoder layers each.[6] We use the Adam optimizer with a learning rate of $5e{-}4$ and a batch size of 32. For regularization, we use a dropout of 0.2.

Hyperparameter	Values
encoder layers	2, 3, 4, 6, 8
decoder layers	2, 3, 4, 6, 8
attention heads	4, 8, 16
embedding dimension	256, 512, 1024.
batch sizes	32, 64, 128
number of tokens	4000, 4096
dropout	0.2, 0.3, 0.4, 0.6, 0.8

Table 3: Hyperparameter settings for tuning BiLSTM and Transformer models.

5.3 Transformer

For the transformer model, we use 5 layers with 8 attention heads in both encoder and decoder. We

[4] https://spacy.io/

[5] https://github.com/moses-smt

[6] Model architecture and hyperparameter values are identified on validation data using values listed in Table 3.

use embedding dimension with $1,024$ units. We express our batch size in number of tokens, and set it to $4,096$. Detailed hyperparameter settings is available in Table 4

Hyperparameter	Values
adam-betas	(0.9,0.98)
clip-norm	0.0
learning rate	5e-4
learning rate scheduler	inverse square root
warmup-updates	4000
dropout	0.3
weight-decay	0.0001
criterion	label smoothed cross entropy
label-smoothing	0.5
encoder-layerdrop	0.2
decoder-layerdrop	0.2

Table 4: Hyperparameters for Transformer model

5.4 Hyperparameters, Vocab. & Training

Hyperparameters. We experimented with different hyperparameter values to ensure optimization of our models. Since the size of data is small, we used fewer layers, and smaller batch sizes as referenced in literature (Sennrich and Zhang, 2019; Nguyen and Chiang, 2017). A full range of hyperparameters and values are in Table 3.

Vocabulary Size. We varied the number of BPE merge operations from $10,000$ to $30,000$. Our optimal models used 10K merge operations.

Training. We train the model with fairseq toolkit for 7 days, on 1 GPU, and choose best epoch on our development set, reporting performance on TEST.

6 Evaluation

In this section, we first evaluate using BLEU. We provide details of this part of the evaluation in Section 6.1. Next, we provide details on our approach to evaluating BNs in Section 6.2.

6.1 Model Performance

We evaluate the output of our models using BLEU (Papineni et al., 2002b). We use BLEU score because it is the most commonly used metric for MT evaluation. Table 5 shows our n-gram precision figures from 1 to 4-grams and our overall BLEU score. Our transformer model outperforms the BiLSTM and SMT models respectively. We give examples of the output from each model and the gold data in Table 6.

SMT

1-gram	2-gram	3-gram	4-gram	BLEU
62.00	47.97	38.68	31.92	33.01[7]

BiLSTM

1-gram	2-gram	3-gram	4-gram	BLEU
62.97	49.21	40.08	33.43	33.44[8]

Transformer

1-gram	2-gram	3-gram	4-gram	BLEU
66.27	**53.42**	**44.45**	**37.68**	**37.93**[9]

Table 5: Model performance at 1-4 grams with final column showing BLEU score as given by the sacrebleu (Post, 2018).

Gold Data	SMT	BiLSTM	Transformer
as in the days when you came out of egypt, i will show them my wonders.	like the day she came up out of egypt, i will show wonders known him.	as soon as he came out of egypt, i will show him all the wonders he has done.	like the day he came up out of egypt, i will show him the wonders.
rescue me from the mouth of the lions; save me from the horns of the wild oxen.	the lord . deliver my life from the lions; rescue me from the horns of the wild ox.	deliver my life from the lion's hands ; deliver me from the hand of the wild ox.	rescue my life from the lion; rescue me from the horns of the wild ox.
so now the lord has put a lying spirit in the mouths of all these prophets of yours. the lord has decreed disaster for you.	so the lord has put a lying spirit in the mouths of all your prophets. the lord has decreed disaster for you.	so the lord has put a lying spirit in the mouths of all these prophets of yours. the lord has decreed disaster for you.	so the lord has put a lying spirit in the mouths of all these prophets of yours. the lord has decreed disaster for you.

Table 6: A comparison of the gold data with the output of our models.

6.2 Preparing a Gold Standard for Bare Noun Disambiguation

We group English translations of Yorùbá BNs into 5 categories described in Table 7: *generic*, *indefinite singular*, *indefinite plural*, *definite singular* and *definite plural*. We do this both for the gold standard target data and for the output of our MT systems, and compare the distribution of categories in the MT output to the category distribution in

the English gold standard data. The annotation was performed by a linguist who is also a native speaker of Yorùbá.

Category	Description
Generic	noun refers to a kind or class of individuals.
Indefinite Singular	noun refers to a single non-specific object
Indefinite Plural	noun refers to multiple non-specific objects
Definite Singular	noun refers to a single specific object
Definite Plural	noun refers to multiple specific objects

Table 7: Noun Categories when translating from Yorùbá to English

Determining the correct category for the translation of a BN requires us to control for the following contextual information: **type of verb** (STATIVE versus EVENTIVE); **the discourse context** (e.g. the preceding sentence, the familiarity constraint); and **real-world knowledge** (which may trigger the uniqueness constraint). E.g. Yorùbá words (ọba *king*, ọlọ́run *god*) should always be translated into definite singulars because Yorùbá speakers commonly use them to refer to specific entities.

6.3 Evaluation of Bare Noun Disambiguation

To measure how well our models translate BNs, we arbitrarily select 100 sentences and evaluate the percentage of correct BN translations for each model. The 100 sentences selected contained 236 occurrences of BNs. There were 9 indefinite plural, 46 indefinite singular, 35 definite plural, 103 definite singular and 43 generic examples in this randomly selected set. We perform a detailed error analysis on the 100 sentences for each case of incorrect BN translation.

We assemble the type of errors found in the translation output into a confusion matrix that captures the behaviour of the 3 models in Figure 1. Each row in the confusion matrix represents one of our 5 noun categories and each row adds up to 100%.The dark blue coloured boxes represent the correctly translated values in percentages. The light blue boxes are the incorrectly translated categories that fell within one of the 5% categories. The column "others" contain incorrectly translated choices outside the 5 categories. We discuss these type of errors in detail in 6.4.

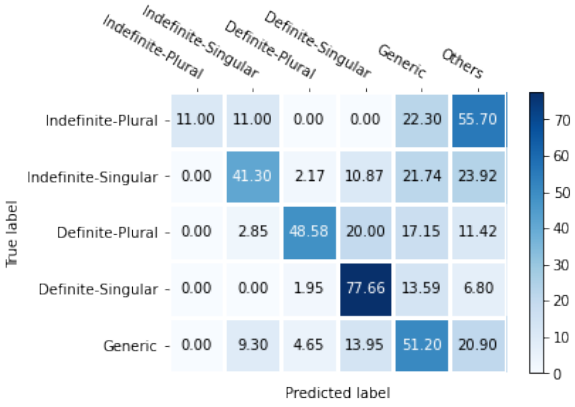

(a) SMT

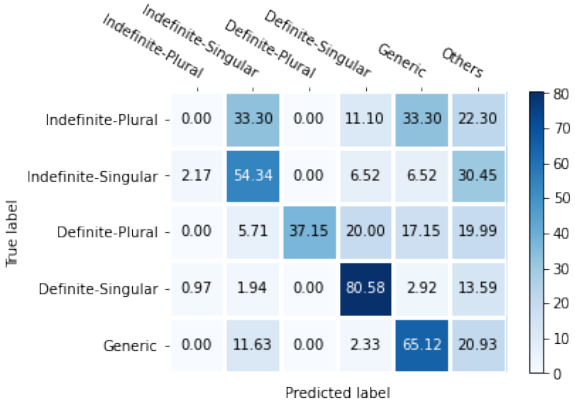

(b) BiLSTM

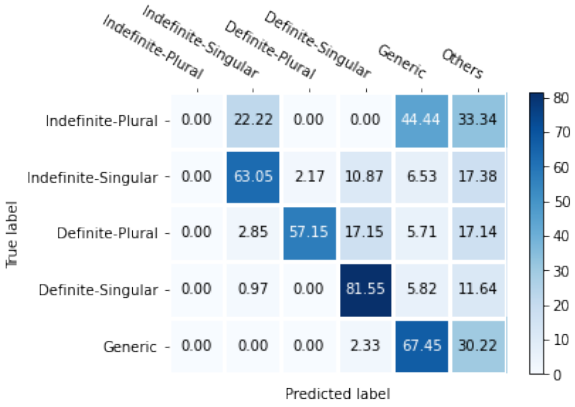

(c) Transformer

Figure 1: Confusion matrices showing the disambiguation patterns of the SMT, BiLSTM, and Transformer Models.

The results show that the transformer model outperforms the SMT and BILSTM models for all categories except the indefinite plural category where the SMT outperforms the Transformer model. The BiLSTM also outperforms the SMT for 3 categories.

How the models handle indefinites. In the context of indefinites, we found that all 3 models achieved low accuracy scores. We assume this

to be the case because fewer cases of indefinites are often reported in languages like Yorùbá that lack overt definite and indefinite markers and this will be represented in the data we used for training. Languages like Yorùbá that do not have overt definite and indefinite determiners are ambiguous between definite and indefinite readings. Indefinites are blocked when the common ground establishes the uniqueness, or familiarity of the set denoted by the noun (Dayal and Sağ, 2019). It is expected that indefinites can occur everywhere else. However, in texts such as these, context introduces either familiarity or uniqueness therefore reducing the occurrence of indefinites.

How the models handle definites. There were 138 BN instances that translated as definites in the set we evaluated. This is due to the aforementioned system of disambiguation that assigns BNs to the definite class if they occur in an environment of uniqueness or familiarity. There are also certain words that have an inherent unique meaning due to cultural beliefs of Yorùbá people. Titles such as ọba *king*, olúwa *lord* and many more often have a definite translation because it is culturally believed for instance that only one king can rule a domain. The models show an improvement in translating definites when compared to indefinites.

How the models handle generics. The Transformer model outperforms the BiLSTM and SMT models for this category while the BiLSTM outperforms the SMT.

6.4 Error Analysis

We focus on other errors, that is those errors presented in the "others" category/column in Figure 1. For this class of errors, the wrongly translated BN did not translate into one of definite singular or plural, indefinite singular or plural, generic or BN category. We therefore evaluate the errors found in the sentences. For the sentences evaluated, we focus only on errors that involve nouns and determiners and ignore errors relating to other classes of words. This means that if a sentence had errors with, for example, verbs or adjectives, we ignored these errors. We categorize the errors we found and provide examples errors for each category in the SMT and NMT models in Table 8. We **bold** face relevant words and phrases occurring in the gold data that had errors in the model output.

Missing word. Outputs with missing nouns or determiners are classified under this category.

Wrong word or spelling. Wrong use of determiners or incorrect nouns belong to this category. We also classify wrong spellings, poorly inflected forms of the noun and unknown words under this category.

Grammaticality. We categorize both syntactic and semantic errors here. Wrong tense or aspect, incorrect number, poor punctuation, and lack of coherence are categorized under this class.

Wrong word order or category. If the order of the noun and determiner is wrong, even if the determiner and nouns are correct, we classify this as an error. We also include instances where the order of the noun and determiner occurs inappropriately, either before the verb or any other category. We found that some instances require both word level and phrase level order errors. In the case of word level order errors, we can generate a correct sentence by moving individual words, independently of each other, whereas for a phrase level order error, blocks of consecutive words should be moved together to form a right translation. In addition, this category includes instances in which a different type is used instead of a noun or determiner.

In evaluating each category, we do not consider correct synonyms as errors.

6.5 Word-Level BN Disambiguation

We randomly select 10 nouns that occur as BNs in the test data and check how well the 3 models disambiguate these words. We use the following words: ìlú *town*, ìwé *book*, ilé *house*, ọkùnrin *male*, obìnrin *female*, aṣọ *clothing*, ilẹ̀ *land*, ọba *king*, àlùfáà *priest* and baba *father*.

We then calculate the percentage of correct translations of these BNs in the test set. We use the English gold data to determine the correct disambiguation and compare each instance of the BNs with the corresponding occurrence in the gold data.

Our analysis in Figure 2 shows that the transformer model performs better in disambiguating the BNs selected. The transformer model achieves **67.85%** accuracy while the BiLSTM model achieves an accuracy of **63.83%**. The SMT model, on the other hand, achieves an accuracy of **59.12%**.

Error	Model	Gold	Model Output
Missing word	SMT	with each bull prepare a grain offering of three-tenths of **an ephah** of fine flour who shut up the sea behind doors when it burst forth from **the womb**,	with each bull prepare a drink offering with three-tenths of fine flour or who shut the doors of the sea, when he flow back as if he had the,
	BiLSTM	as in **the days** when you came out of egypt, i will show them my wonders. and he inserted the poles into **the rings** on the sides of the ark to carry it.	as soon as he came out of egypt , i will show him all the wonders he has done . then he put the poles on each side of the ark to put it on the chest .
	Transf	now hiram had sent to **the king** 120 talents of gold. each day one ox, six choice sheep and **some poultry** were prepared for me,...	and hiram sent him 120 talents of gold each day an ox, a choice sheep and six days they provide for me...
Wrong word / spelling	SMT	everyone who quotes **proverbs** will quote this **proverb** about you: like mother, like daughter ...he who had received **the promises** was about to sacrifice his one and only son,	all those who were powe, it will this powe you: as mothers, so his son of woman. ...he who receive **the promised** almost ready to take your son into one sacrifice.
	BiLSTM	the lot settles **disputes** and keeps strong opponents apart. he spoke, and there came swarms of **flies**, and gnats throughout their country.	lot lays up the battle and makes up two unchange as from each other . he spoke, and the kind of reitution came and became gnats in their land
	Transf	the lot settles **disputes** and keeps strong opponents apart. with **an opening** in **the center** of **the robe**...	the snow finish quarreling and two oppose each other with **the holes** among **the belt**...
Grammaticality	SMT	this is **a decree** for israel, an ordinance of the god of jacob. **the simple** inherit folly, but **the prudent** are crowned with knowledge.	this is a lasting ordinance for israel, and the law of the god of jacob. a simple inherit folly but to be wise in the crown of knowledge .
	BiLSTM	as for **the donkeys** you lost three days ago, do not worry about them; they have been found... and we know that in all things god works for the good of those who love him ...	for **a donkeys** he was three days ago, not terrified by them; they were found to be found We know that everything is in good deeds for those who love god,...
	Transf	there will be **a highway** for the remnant of his people ... overlay the frames with gold and make gold rings to hold **the crossbars.**	good **ways** will be for his people,... overlay the frames with gold and make gold rings so they can be **crossbars**...
Wrong word order	SMT	by faith abraham, when god tested him, offered **isaac** as a sacrifice. ...the next day the south wind came up,	by faith of abraham, when he was tempted to, isaac offer sacrifices, ...the next day, forth the south wind began to blow,...
	BiLSTM	...i will give my daughter acsah in **marriage** to the man who attacks and captures kiriath sepher. do not love sleep or you will grow poor; stay awake and you will have food to spare	...i'll give my daughter acsah to a man who struck down kiriath sepher and took him **wedding**. do not love a sleep, or you will be poor. do not sleep and have food to give you something to eat .
	Transf	about this time next year, elisha said, you will hold ***a son*** *in your arms.* ...israel served to get a wife, and to pay for her he tended sheep.	elisha said, this time is coming, you will take *your hand for your son.* ...israel worshiped as a wife and took care of the meat to pay **the bride** for money.

Table 8: Example of errors for the three models under each error category we described. The English Gold is the NIV translation for the Yorùbá Source

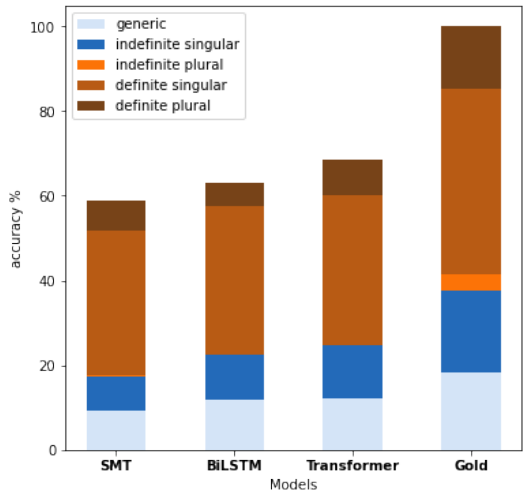

Figure 2: Distribution of disambiguation patterns in our models. Our gold data has 100% in disambiguation.

We also found the transformer outperforming the SMT and BiLSTM models in correctly translating BNs. We also found that all 3 models best performed in translating a BN in Yorùbá into an definite singular in English. This finding corroborates research that predicts that languages which lack overt definite and indefinite markers have larger cases of definites, and findings within the MT community that MT models improve with more data. We further analyzed the type of errors our systems produce. We identified cases of missing words, wrong word or spellings, grammaticality issues, and word-order errors. We found that even when certain BNs have been correctly categorized by the models, the models still had semantic, and or syntactic errors.

7 Conclusion

In this work, we showed how SMT, BiLSTM, and transformer models translate BNs in Yorùbá, a resource scarce language. We compared the ability of SMT and NMT models to correctly translate BNs into various categories referenced in the syntax literature of Yorùbá. We measured the performance of the MT models and the output using BLEU scores, and by counting the percentage of correctly disambiguated BNs compared with incorrectly disambiguated BNs. We found a positive correlation between disambiguation accuracy, and BLEU scores as well as a positive correlation between number of occurrences of a category and the accuracy in translation.

In order to further probe the capacity of SMT and NMT models in disambiguating BNs, further work can be carried out to improve the SMT and NMT models and perform human based evaluations on the entire quality of the MT output. We can improve the MT systems with back-translation, cross-lingual word embeddings, increasing the size of data used for training, transfer learning, among other approaches. Standard test sets can also be developed to aid automatic comparison of human evaluations and machine based evaluations. In addition, the Yorùbá data we used for this work is a translated document; a translation from English to Yorùbá and it will be interesting to use a text originally written in Yorùbá but translated to English for this experiment.

Acknowledgements

We gratefully acknowledge support from Canada Research Chairs (CRC), the Natural Sciences and Engineering Research Council of Canada (NSERC; RGPIN-2018-04267), the Social Sciences and Humanities Research Council of Canada (SSHRC; 435-2018-0576; 895-2020-1004; 895-2021-1008), Canadian Foundation for Innovation (CFI; 37771), Compute Canada (CC),[10] UBC ARC-Sockeye,[11], Advanced Micro Devices, Inc. (AMD), and Google. Any opinions, conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of CRC, NSERC, SSHRC, CFI, CC, AMD, Google, or UBC ARC-Sockeye.

References

Barbara Abbott. 2006. Definite and indefinite. *Encyclopedia of language and linguistics*, 3(392):99.

Antonio Valerio Miceli Barone, Jindřich Helcl, Rico Sennrich, Barry Haddow, and Alexandra Birch. 2017. Deep architectures for neural machine translation. *arXiv preprint arXiv:1707.07631*.

Rachel Bawden, Rico Sennrich, Alexandra Birch, and Barry Haddow. 2017. Evaluating discourse phenomena in neural machine translation. *arXiv preprint arXiv:1711.00513*.

Rachel Bawden, Rico Sennrich, Alexandra Birch, and Barry Haddow. 2018. Evaluating discourse phenomena in neural machine translation. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 1304–1313, New Orleans, Louisiana. Association for Computational Linguistics.

Luisa Bentivogli, Arianna Bisazza, Mauro Cettolo, and Marcello Federico. 2016. Neural versus phrase-based machine translation quality: a case study. *arXiv preprint arXiv:1608.04631*.

Franck Burlot and François Yvon. 2017. Evaluating the morphological competence of machine translation systems. In *Proceedings of the Second Conference on Machine Translation*, pages 43–55.

Greg N Carlson. 1989. On the semantic composition of english generic sentences. In *Properties, types and meaning*, pages 167–192. Springer.

Marine Carpuat. 2009. One translation per discourse. In *Proceedings of the Workshop on Semantic Evaluations: Recent Achievements and Future Directions (SEW-2009)*, pages 19–27.

[10] https://www.computecanada.ca

[11] https://arc.ubc.ca/ubc-arc-sockeye

Lisa Lai-Shen Cheng and Rint Sybesma. 1999. Bare and not-so-bare nouns and the structure of np. *Linguistic inquiry*, 30(4):509–542.

Gennaro Chierchia. 1998. Reference to kinds across language. *Natural language semantics*, 6(4):339–405.

Alexis Conneau, Guillaume Lample, Marc'Aurelio Ranzato, Ludovic Denoyer, and Hervé Jégou. 2017. Word translation without parallel data. *arXiv preprint arXiv:1710.04087*.

FREDERIC L. DARLEY. 1959. Certain language skills in children: Their development and interrelationships. *Pediatrics*, 23(4):819–819.

Veneeta Dayal and Yağmur Sağ. 2019. Determiners and bare nouns. *Annual Review of Linguistics*, 6.

Serge Gladkoff and Lifeng Han. 2021. HOPE: A task-oriented and human-centric evaluation framework using professional post-editing towards more effective MT evaluation. *CoRR*, abs/2112.13833.

Annette Rios Gonzales, Laura Mascarell, and Rico Sennrich. 2017. Improving word sense disambiguation in neural machine translation with sense embeddings. In *Proceedings of the Second Conference on Machine Translation*, pages 11–19.

Liane Guillou and Christian Hardmeier. 2016. Protest: A test suite for evaluating pronouns in machine translation. In *Proceedings of the Tenth International Conference on Language Resources and Evaluation (LREC'16)*, pages 636–643.

Liane Guillou, Christian Hardmeier, Preslav Nakov, Sara Stymne, Jörg Tiedemann, Yannick Versley, Mauro Cettolo, Bonnie Webber, and Andrei Popescu-Belis. 2019. Findings of the 2016 wmt shared task on cross-lingual pronoun prediction. *arXiv preprint arXiv:1911.12091*.

Christian Hardmeier. 2012. Discourse in statistical machine translation: A survey and a case study. *Discours*.

Kenneth Heafield. 2011. Kenlm: Faster and smaller language model queries. In *Proceedings of the sixth workshop on statistical machine translation*, pages 187–197.

Pierre Isabelle, Colin Cherry, and George Foster. 2017. A challenge set approach to evaluating machine translation. *arXiv preprint arXiv:1704.07431*.

Marcin Junczys-Dowmunt, Tomasz Dwojak, and Hieu Hoang. 2016. Is neural machine translation ready for deployment? a case study on 30 translation directions. *arXiv preprint arXiv:1610.01108*.

Filip Klubička, Antonio Toral, and Víctor M Sánchez-Cartagena. 2017. Fine-grained human evaluation of neural versus phrase-based machine translation. *The Prague Bulletin of Mathematical Linguistics*, 108(1):121–132.

Philipp Koehn, Hieu Hoang, Alexandra Birch, Chris Callison-Burch, Marcello Federico, Nicola Bertoldi, Brooke Cowan, Wade Shen, Christine Moran, Richard Zens, et al. 2007. Moses: Open source toolkit for statistical machine translation. In *Proceedings of the 45th annual meeting of the association for computational linguistics companion volume proceedings of the demo and poster sessions*, pages 177–180.

Manfred Krifka. 2003. Bare nps: kind-referring, indefinites, both, or neither? In *Semantics and linguistic theory*, volume 13, pages 180–203.

Guillaume Lample, Alexis Conneau, Ludovic Denoyer, and Marc'Aurelio Ranzato. 2017. Unsupervised machine translation using monolingual corpora only. *arXiv preprint arXiv:1711.00043*.

Richard K Larson. 1985. Bare-np adverbs. *Linguistic inquiry*, 16(4):595–621.

Tal Linzen, Emmanuel Dupoux, and Yoav Goldberg. 2016. Assessing the ability of lstms to learn syntax-sensitive dependencies. *Transactions of the Association for Computational Linguistics*, 4:521–535.

Vivien Macketanz, Eleftherios Avramidis, Shushen Manakhimova, and Sebastian Möller. 2021. Linguistic evaluation for the 2021 state-of-the-art machine translation systems for German to English and English to German. In *Proceedings of the Sixth Conference on Machine Translation*, pages 1059–1073, Online. Association for Computational Linguistics.

Ruslan Mitkov. 1999. Introduction: special issue on anaphora resolution in machine translation and multilingual nlp. *Machine translation*, pages 159–161.

Toan Q Nguyen and David Chiang. 2017. Transfer learning across low-resource, related languages for neural machine translation. *arXiv preprint arXiv:1708.09803*.

Franz Josef Och and Hermann Ney. 2003. A systematic comparison of various statistical alignment models. *Computational linguistics*, 29(1):19–51.

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002a. Bleu: a method for automatic evaluation of machine translation. In *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics*, pages 311–318, Philadelphia, Pennsylvania, USA. Association for Computational Linguistics.

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002b. Bleu: a method for automatic evaluation of machine translation. In *Proceedings of the 40th annual meeting on association for computational linguistics*, pages 311–318. Association for Computational Linguistics.

Maja Popović. 2011. Hjerson: An open source tool for automatic error classification of machine translation output. *The Prague Bulletin of Mathematical Linguistics*, 96(1):59–67.

Maja Popović. 2018. Language-related issues for nmt and pbmt for english–german and english–serbian. *Machine Translation*, 32(3):237–253.

Matt Post. 2018. A call for clarity in reporting BLEU scores. In *Proceedings of the Third Conference on Machine Translation: Research Papers*, pages 186–191, Brussels, Belgium. Association for Computational Linguistics.

Craige Roberts. 2003. Uniqueness in definite noun phrases. *Linguistics and philosophy*, 26(3):287–350.

Bertrand Russell. 1905. On denoting. *Mind*, 14(56):479–493.

Rico Sennrich. 2016. How grammatical is character-level neural machine translation? assessing mt quality with contrastive translation pairs. *arXiv preprint arXiv:1612.04629*.

Rico Sennrich, Barry Haddow, and Alexandra Birch. 2015. Neural machine translation of rare words with subword units. *arXiv preprint arXiv:1508.07909*.

Rico Sennrich, Barry Haddow, and Alexandra Birch. 2016. Controlling politeness in neural machine translation via side constraints. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 35–40.

Rico Sennrich and Biao Zhang. 2019. Revisiting low-resource neural machine translation: A case study. *arXiv preprint arXiv:1905.11901*.

Matthew Snover, Bonnie Dorr, Richard Schwartz, Linnea Micciulla, and John Makhoul. 2006. A study of translation edit rate with targeted human annotation. In *Proceedings of association for machine translation in the Americas*, volume 200. Cambridge, MA.

Antonio Toral and Víctor M Sánchez-Cartagena. 2017. A multifaceted evaluation of neural versus phrase-based machine translation for 9 language directions. *arXiv preprint arXiv:1701.02901*.

Eva Vanmassenhove, Dimitar Shterionov, and Andy Way. 2019. Lost in translation: Loss and decay of linguistic richness in machine translation. *arXiv preprint arXiv:1906.12068*.

Elena Voita, Pavel Serdyukov, Rico Sennrich, and Ivan Titov. 2018. Context-aware neural machine translation learns anaphora resolution. *arXiv preprint arXiv:1805.10163*.

Qiang Wang, Bei Li, Tong Xiao, Jingbo Zhu, Changliang Li, Derek F Wong, and Lidia S Chao. 2019. Learning deep transformer models for machine translation. *arXiv preprint arXiv:1906.01787*.

C Udny Yule. 2014. *The statistical study of literary vocabulary*. Cambridge University Press.

Dynamic Position Encoding for Transformers

Joyce Zheng[*]
Huawei Noah's Ark Lab
jy6zheng@uwaterloo.ca

Mehdi Rezagholizadeh
Huawei Noah's Ark Lab
mehdi.rezagholizadeh@huawei.com

Peyman Passban [†]
Amazon
passban.peyman@gmail.com

Abstract

Recurrent models have been dominating the field of neural machine translation (NMT) for the past few years. Transformers (Vaswani et al., 2017) have radically changed it by proposing a novel architecture that relies on a feed-forward backbone and self-attention mechanism. Although Transformers are powerful, they could fail to properly encode sequential/positional information due to their non-recurrent nature. To solve this problem, position embeddings are defined exclusively for each time step to enrich word information. However, such embeddings are fixed after training regardless of the task and word ordering system of the source and target languages.

In this paper, we address this shortcoming by proposing a novel architecture with new position embeddings that take the order of the target words into consideration. Instead of using predefined position embeddings, our solution *generates* new embeddings to refine each word's position information. Since we do not dictate the position of the source tokens and we learn them in an end-to-end fashion, we refer to our method as *dynamic* position encoding (DPE). We evaluated the impact of our model on multiple datasets to translate from English to German, French, and Italian and observed meaningful improvements in comparison to the original Transformer.

1 Introduction

In statistical machine translation (SMT), the general task of translation consists of reducing the input sentence into smaller units (also known as statistical phrases), selecting an optimal translation for each unit, and placing them in the correct order (Koehn, 2009). The last step, which is also referred to as the reordering problem, is a great source of complexity and importance, which is handled with a variety of statistical as well as string- and tree-based solutions (Bisazza and Federico, 2016).

As evident in SMT, the structure and position of words within a sentence is crucial for accurate translation. The importance of such information can also be explored in NMT and for Transformers. Previous literature, such as Chen et al. (2020), demonstrated that source input sentences enriched with target order information have the capacity to improve translation quality in neural models. They showed that position encoding seems to play a key role in translation and this motivated us to further explore this area.

Since Transformers have a non-recurrent architecture, they could face problems when encoding sequential data. As a result, they require an explicit summation of the input embeddings with position encoding to provide information about the order of each word. However, this approach falsely assumes that the correct position of each word is always its original position in the source sentence. This interpretation might be true when only considering the source side, whereas we know from SMT (Bisazza and Federico, 2016; Cui et al., 2016) that arranging input words with respect to the order of their target pairs can lead to better results.

In this work, we explore injecting target position information alongside the source words to enhance Transformers' translation quality. We first examine the accuracy and efficiency of a two-pass Transformer (2PT), which consists of a pipeline connecting two Transformers. The first Transformer reorders the source sentence and the second one translates the reordered sentences. Although this approach incorporates order information from the target language, it lacks end-to-end training and requires more resources than a typical Transformer. Accordingly, we introduce a better alternative, which effectively learns the reordered positions in an end-to-end fashion and uses this information with the original source sequence to boost

[*] Work done while Joyce Zheng was an intern at Huawei
[†] Work done while Peyman Passban was at Huawei.

Proceedings of the 29th International Conference on Computational Linguistics, pages 5076–5084
October 12–17, 2022.

translation accuracy. We refer to this alternative as *Dynamic Position Encoding* (DPE).

Our contribution in this work is threefold:

- First, we demonstrate that providing source-side representations with target position information improves translation quality in Transformers.
- We also propose a novel architecture, DPE, that efficiently learns reordered positions in an end-to-end fashion and incorporates this information into the encoding process.
- Finally, we use a preliminary two-pass design to show the importance of end-to-end learning in this problem.

2 Background

2.1 Pre-Reordering in Machine Translation

In standard SMT, pre-reordering is a well-known technique. Usually, the source sentence is reordered using heuristics such that it follows the word order of the target language. Figure 1 illustrates this concept with an imaginary example.

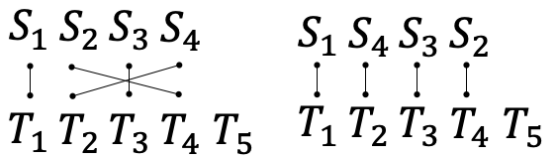

Figure 1: The order of source words before (left-hand side) and after (right-hand side) pre-reordering. S_i and T_j show the i-th source and j-th target words, accordingly.

As the figure shows, the original alignment between the source (S_i) and target (T_i) words are used to define a new order for the source sentence. With the new order, the translation engine does not need to learn the relation between source and target ordering systems, as it directly translates from one position on the source side to the *same position* on the target side. Clearly, this can significantly reduce the complexity of the translation task. The work of Wang et al. (2007), provides a good example of systems with pre-reordering, in which the authors studied this technique for the English–Chinese pair.

The concept of re-ordering does not necessarily need to be tackled prior to translation; in Koehn et al. (2007), generated sequences are reviewed by a classifier after translation to correct the position of words that are placed in the wrong order. The entire reordering process can also be embedded into the decoding process (Feng et al., 2013).

2.2 Tackling the Order Problem in NMT

Pre-reordering and position encoding are also common in NMT and have been investigated by various researchers. Du and Way (2017) explored if recurrent neural models can benefit from pre-reordering. Their findings showed that these models might not require any order adjustments because the network itself was powerful enough to learn such mappings. Kawara et al. (2020), unlike the previous work, studied the same problem and reported promising observations on the usefulness of pre-reordering. They used a transduction-grammar-based method with recursive neural networks and showed how impactful pre-reordering could be.

Liu et al. (2020) followed a different approach and proposed modeling position encoding as a continuous dynamical system through a neural ODE. Ke et al. (2020) investigated improving positional encoding by untying the relationship between words and positions. They suggested that there is no strong correlation between words and absolute positions, so they removed this noisy correlation. This form of separation has its own advantages, but by removing this relationship between words and positions from the translation process, they might lose valuable semantic information about the source and target sides.

Shaw et al. (2018a) explored a relative position encoding method by having the self-attention mechanism consider the distance between source words. Garg et al. (2019) incorporated target position information via multitasking where a translation loss was combined with an alignment loss that supervised one decoder head to learn position information. Chen et al. (2020) changed the Transformer architecture to incorporate order information at each layer. They selected a reordered target word position from the output of each layer and injected it into the next layer. This is the closest work to ours, so we consider it as our main baseline.

3 Methodology

3.1 Two-Pass Translation for Pre-Reordering

Our goal is to boost translation quality in Transformers by injecting target order information into the source-side encoding process. We propose dynamic position encoding (DPE) to achieve this,

but before that, we discuss a preliminary two-pass Transformer (2PT) architecture to demonstrate the impact of order information in Transformers.

The main difference between 2PT and DPE is that DPE is an end-to-end, differential solution whereas 2PT is a pipeline that connects two *different* Transformers. They both work towards leveraging order information to improve translation but in different fashions. The 2PT architecture is illustrated in Figure 2.

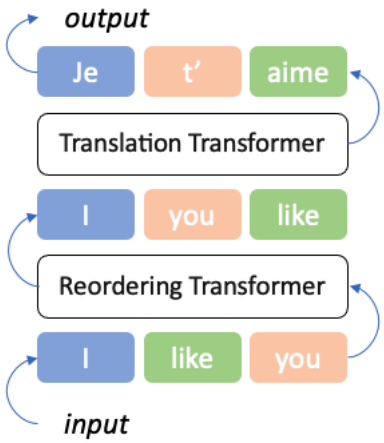

Figure 2: The two-pass Transformer architecture. The input sequence is first re-ordered to a new and less complex form for the translation Transformer. Then, the translation Transformer uses the re-ordered input sequence to decode a target sequence.

2PT has two different Transformers. The first one is used for reordering purposes instead of translation. It takes source sentences and generates a reordered version of them, e.g. referring back to Figure 1, if the input to the first Transformer is [S_1, S_2, S_3, S_4] the expected output from the first transformer is [S_1, S_4, S_3, S_2]. We created a new corpus using `FastAlign` to train this reordering model (Dyer et al., 2013).[1]

`FastAlign` is an unsupervised word aligner that processes source and target sentences together and provides word-level alignments. It is usable at training time but not for inference because it requires access to both sides and we only have the source side (at test time). As a solution, we used the alignments to create a training set and utilized it to train the first Transformer in 2PT. Figure 3 shows the input and output format in `FastAlign` and how it helps generate training samples.

As the figure demonstrates, given a pair of English–German sentences, word alignments are generated. In order to process the alignments, we designed rules to handle different cases:

- **One-to-Many Alignments**: We only consider the first target position in one-to-many alignments (see the figure).
- **Many-to-One Alignments**: Multiple source words are reordered together (as one unit while maintaining their relative positions with each other) using the position of the corresponding target word.
- **No Alignment**: Words that do not have any alignments are skipped and we do not change their position
- We also ensure that no source word would be aligned with a position beyond the source sentence length.

Considering these rules and what `FastAlign` generates, the example input sentence "*mr hän@@ sch represented you on this occasion .*" (in Figure 3) is reordered to "*mr hän sch you this represented .*" @@ are auxiliary symbols added in between sub-word units during preprocessing. See Section 4 for more information.

Using re-ordered sentences, the first Transformer in 2PT is trained to reorder the original source sentences, and the second Transformer, which is responsible for translation, receives the re-ordered source sentences and maps them to their target translations. Despite different data formats and different inputs/outputs, 2PT is still a pipeline that translates a source language to a target one through internal modifications that are hidden from the user.

3.2 Dynamic Position Encoding

Unlike 2PT, the dynamic position encoding (DPE) method takes advantage of end-to-end training, while the source side still learns target reordering position information. It boosts the input of an ordinary Transformer's encoder with target position information, but leaves its architecture untouched, as illustrated in Figure 4.

The input to DPE is a source word embedding (w_i) summed with sinusoidal position (p_i) encoding ($w_i \oplus p_i$). We refer to these embeddings as *enriched embeddings*. Sinusoidal position encoding is part of the original design of Transformers and we assume the reader is familiar with this concept. For more details see Vaswani et al. (2017).

[1] https://github.com/clab/fast_align

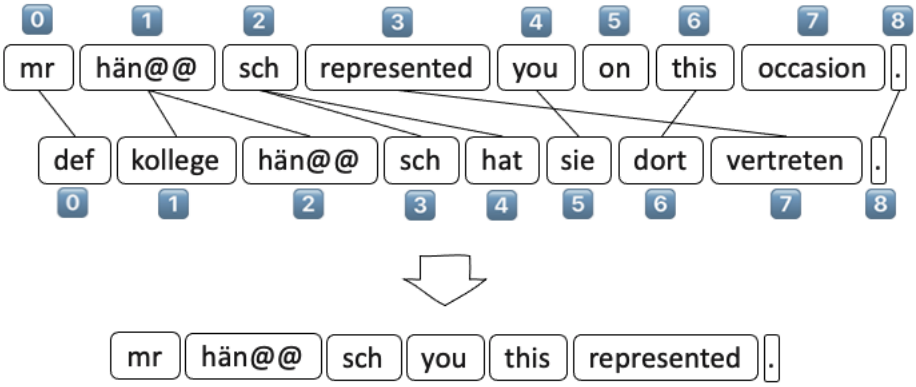

Figure 3: `FastAlign`-based reordering using a sample sentence from our English–German dataset. Using word alignments, we generate a new reordered form from each source sentence as the new target sequence. We then use the pairs of source and new target sequences to train the first Transformer of 2PT.

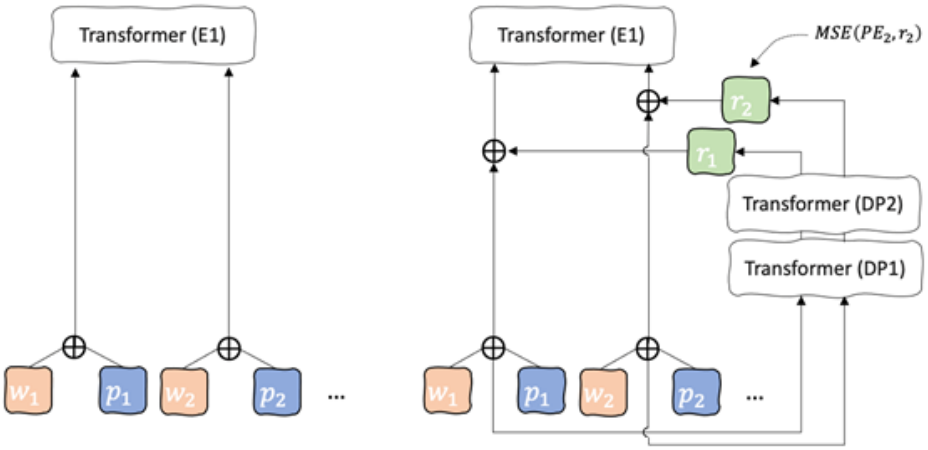

Figure 4: The left-hand side is the original Transformer's architecture and the figure on the right is our proposed architecture. E1 is the first encoder layer of the ordinary Transformer and DP1 is the first layer of the DPE network.

DPE is another neural network placed in-between the *enriched embeddings* and the first encoder layer of the (translation) Transformer. In other words, the input to the DPE network is the embedding table of the Transformer, and its final layer outputs into the first encoder layer of the Transformer. Thus, the DPE network can be trained jointly with the Transformer using the original parallel sentence pairs.

DPE processes enriched embeddings and generates a new form of them that is represented as r_i in this paper, i.e. DPE($w_i \oplus p_i$) = r_i. DPE-generated embeddings are intended to preserve target-side order information about each word. In the original Transformer, the position of w_i is dictated by adding p_i, but the original position of this word is not always the best one for translation; thus r_i is defined to address this problem. If w_i appears in the i-*th* position but j is its best position with respect to the target language, r_i is supposed to learn information about the j-*th* position and mimic p_j. Accordingly, the combination of p_i and r_i should provide w_i with the pre-reordering information it requires to improve translation accuracy.

In our design, DPE consists of two Transformer layers. We determined this number through an empirical study to find a reasonable balance between translation quality and resource consumption. These two layers are connected to an auxiliary loss function to ensure that the output of DPE is what we need for re-ordering.

This additional loss measures the mean squared error between the embeddings produced by DPE (r_i) and the supervising positions (PE) defined by `FastAlign` alignments. This learning process is simply formulated in Equation 1:

$$\mathcal{L}_{order} = \frac{\sum_{i=1}^{|S|} MSE(PE_i, r_i)}{|S|} \quad (1)$$

where S is the source sequence length and $MSE()$

is the mean-square error function. The supervising position PE_i is obtained by taking the target position associated with w_i that was defined by `FastAlign` as described in Section 3.1.

To clarify how $\mathcal{L}_{order}$ works, we use the aforementioned scenario as an example. We assume that the correct position for w_i according to the `FastAlign` alignments is j, so $PE_i = p_j$ and we thus compute $MSE(p_j, r_i)$. Through this technique, we encourage the DPE network to learn pre-reordering in an end-to-end fashion and provide w_i with position refinement information.

The total loss function when training the entire model includes the auxiliary reordering loss function $\mathcal{L}_{order}$ summed with the standard Transformer loss $\mathcal{L}_{translation}$, as in Equation 2:

$$\mathcal{L}_{total} = \lambda \times \mathcal{L}_{translation} + (1 - \lambda) \times \mathcal{L}_{order} \quad (2)$$

where λ is a hyper-parameter that represents the weight of the reordering loss. λ was determined by minimizing the total loss on the development set during training.

4 Experimental Study

4.1 Dataset

To train and evaluate our models, we used the IWSLT-14 collection (Cettolo et al., 2012) and the WMT-14 dataset.[2] Our datasets are commonly used in the field, which makes our results easily reproducible. Our code is also publicly available to help other researchers further investigate this topic.[3] The IWSLT-14 collection was used to study the impact of our model for the English–German (En–De), English–French (En–Fr), and English–Italian (En–It) pairs. We also reported results on the WMT dataset, which provides a larger training corpus. We know that the quality of NMT models vary in proportion to the corpus size, so these experiments provide more information to better understand our model.

To prepare the data, sequences were lower-cased, normalized, and tokenized using the scripts provided by the Moses toolkit[4] (Koehn et al., 2007) and decomposed into sub-words via Byte-Pair Encoding (BPE) (Sennrich et al., 2016). The vocabulary sizes extracted for the IWSLT and WMT datasets were 32K and 40K, respectively. For the En–De pair of WMT-14, *newstest2013* was used as a development set and *newstest2014* was our test set. For the IWSLT experiments, our test and development sets were as suggested by Zhu et al. (2020). Table 1 provides the statistics of our datasets.

Data	Train	Dev	Test
WMT-14 (En→De)	4.45M	3k	3k
IWSLT-14 (En→De)	160k	7k	6k
IWSLT-14 (En→Fr)	168k	7k	4k
IWSLT-14 (En→It)	167k	7k	6k

Table 1: The statistics of the datasets used in our experiments. Train, Dev, and Test stand for the training, development, and test sets, respectively.

4.2 Experimental Setup

In the interest of fair comparisons, we used the same setup as Chen et al. (2020) to build our baseline for the WMT-14 En–De experiments. This baseline setting was also used for our DPE model and DPE-related experiments. Our models were trained on 8 × V100 GPUs. Since our models rely on the Transformer's backbone, all hyper-parameters that were related to the main Transformer architecture, such as embedding dimensions, the number of attention heads, etc., were set to the default values proposed for *Transformer Base* in Vaswani et al. (2017). Refer to the original work for detailed information.

For IWSLT experiments, we used a lighter architecture since the datasets were smaller than WMT. The hidden dimension was 256 for all encoder and decoder layers, and a dimension of 1024 was used for the inner feed-forward network layer. There were 2 encoder and 2 decoder layers, and 2 attention heads. We found this setting through an empirical study to maximize the performance of our IWSLT models.

For the WMT-14 En–De experiments, similar to Chen et al. (2020), we trained the model for 300K updates and used a single model obtained from averaging the last 5 checkpoints. The model was validated with an interval of 2K on the development dataset. The decoding beam size was 5. In the IWSLT-14 cases, we trained the models for 15,000 updates and used a single model obtained from averaging the last 5 checkpoints that were validated with an interval of 1000 updates. We evaluated all our models with detokenized BLEU (Papineni et al., 2002).

[2] http://statmt.org/wmt14/translation-task.html

[3] https://github.com/jy6zheng/DynamicPositionEncodingModule

[4] https://github.com/moses-smt/mosesdecoder

	Model	Data type	BLEU Score
1	Reordering Transformer	En $\rightarrow$En$_{reordered}$	35.21
2	Transformer Base	En$\rightarrow$De	27.76
3	+ fed with the output of reordering Transformer	En$_{reordered}$ $\rightarrow$De	21.96
4	+ fed with the output of `FastAlign`	En$_{reordered}$ $\rightarrow$De	31.82

Table 2: BLEU scores for the 2PT series of experiments.

4.3 2PT Experiments

Results related to the two-pass architecture are summarized in Table 2. The reordering Transformer (Row 1) works with the source sentences and reorders them with respect to the order of the target language. This was a monolingual translation task with a BLEU score of 35.21. This is a relatively low score for a monolingual setting which indicates how complicated the reordering problem is. Even dedicating a whole Transformer could not fully overcome the reordering problem. This finding also indicates that NMT engines can benefit from using an auxiliary module to handle order complexities. It is usually assumed that the translation engine should perform in an end-to-end fashion where it deals with all the reordering, translation, and other complexities via a single model at the same time. However, if we can separate these subtasks systematically and tackle them individually, there is a chance that we might be able to improve the overall quality.

In Row 3, we used the information previously generated (in Row 1) and showed how a translation model performs when it is fed with reordered sentences. The BLEU score for this task was 21.96, which was significantly lower than the baseline (Row 2). Order information was supposed to increase the overall performance, but we observe a degradation. This is because the first Transformer was unable to detect the correct order (due to the difficulty of this task). In Row 4, we fed the same translation engines with higher-quality order information (generated by `FastAlign`), and the BLEU score rose to 31.82.

We cannot use `FastAlign` at test time but this experiment shows that our hypothesis on the usefulness of order information seems to be correct. Motivated by this, we invented DPE to better leverage order information, and these results are reported in the next section.

4.4 DPE Experiments

Results related to DPE are reported in Table 3. According to the reported scores, DPE led to a +0.81 improvement in the BLEU score compared to *Transformer Base*. To ensure that we evaluated DPE in a fair setup, we re-implemented the *Transformer Base* in our own environment. This eliminated the impact of different factors and ensured that the gain was due to the design of the DPE module itself. We also compared our model to models discussed in the related literature such as that of Shaw et al. (2018b), the reordering embeddings of Chen et al. (2019), and the more recent explicit reordering embeddings of Chen et al. (2020). Our model achieved the best score and we believe it was due to the direct use of order information.

For the DPE architecture, we decided to have two layers (DP1 and DP2) as it produced the best BLEU scores on the development sets without imposing significant training overhead. One important hyper-parameter that directly affects DPE's performance is the position loss weight (λ). We ran an ablation study on the development set to adjust λ. Table 4 summarizes our findings. The best λ value in our setting was 0.5. This value provided an acceptable balance between translation accuracy and pre-reordering costs during training, and shows that the order information can be as important as other translation information.

The design of our Transformer (Transformer Base + DPE) might raise the concern that incorporating pre-reordering information or defining an auxiliary loss might not be necessary. One might suggest that if we use the same amount of resources to increase the Transformer Base's encoder parameters, we should obtain competitive or even better results than the DPE-enhanced Transformer. To address this concern, we designed another experiment that increased the number of parameters/layers in the *Transformer Base* encoder to match the number in our model's parameters. Results related to this experiment are shown in Table 5.

Model	# Params	En→De (WMT)
Transformer Base (Vaswani et al., 2017)	65.0 M	27.30
+ Relative PE (Shaw et al., 2018b)	N/A	26.80
+ Explicit Global Reordering Embeddings (Chen et al., 2020)	66.5 M	28.44
+ Reorder fusion-based source representation (Chen et al., 2020)	66.5 M	28.55
+ Reordering Embeddings (Encoder Only) (Chen et al., 2019)	102.1 M	28.03
+ Reordering Embeddings (Encoder/Decoder) (Chen et al., 2019)	106.8 M	28.22
Transformer Base (our re-implementation)	66.5 M	27.78
Dynamic Position Encoding ($\lambda = 0.5$)	72.8 M	28.59

Table 3: A BLEU score comparison of DPE versus other peers.

Model	WMT'14 En→ De
Baseline	27.78
DPE ($\lambda = 0.1$)	28.17
DPE ($\lambda = 0.3$)	28.16
DPE ($\lambda = 0.5$)	**28.59**
DPE ($\lambda = 0.7$)	27.98

Table 4: BLEU scores of DPE with different λ values.

Model	WMT'14 En→De
Baseline	27.78
8E	28.07
10E	28.54
DPE ($N = 2$)	28.59 (**+ 0.81**)

Table 5: A BLEU score comparison of DPE with the baseline Transformer models plus additional encoder layers (8E for 8 encoder layers and 10E for 10 encoder layers)

The comparison of DPE with the different extensions of the Transformer Base, namely 8E (8 encoder layers) and 10E (10 encoder layers), demonstrated that the increase in BLEU was due to the position information provided by DPE rather than the additional parameters of the DPE layers. In 8E, we provided the same number of additional parameters as the DPE module adds, but experienced less gain in translation quality. In 10E, we even doubled the number of additional parameters to surpass the number of parameters that DPE uses, and yet the DPE extension with 8 encoding layers (two for pre-reordering and six from the original translation encoder) was still superior. This reinforces the idea that our DPE module improves translation accuracy by injecting position information alongside the encoder input.

4.5 Experimental Results on Other Languages

In addition to the previously reported experiments, we evaluated the DPE model on different IWSLT-14 datasets of English–German (En–De), English–French (En–Fr), and English–Italian (En–It). After tuning with different position loss weights on the development set, we determined $\lambda = 0.3$ to be ideal for this setting. The results in Table 6 show that with DPE, the translation accuracy improved for different settings and the improvement was not unique to the En–De WMT language pair.

Our DPE architecture works with a variety of language pairs of different sizes and this increases our confidence in the beneficial effect of order information. It is usually hard to show the impact of auxiliary signals in NMT models and this could be more difficult with smaller datasets, but our IWSLT results are promising. Accordingly, it would not be unfair to claim that DPE is useful regardless of the language and dataset size.

5 Conclusion and Future Work

In this paper, we first explored whether Transformers would benefit from order signals. Then, we proposed a new architecture, DPE, that generates embeddings containing target word position information to boost translation quality.

The results obtained in our experiments demonstrate that DPE improves the translation process by helping the source side learn target position information. The DPE model consistently outperformed the baselines of related literature. It also showed improvements with different language pairs and dataset sizes.

5.1 Future Work

Our experiments can provide the groundwork for further exploration of dynamic position encoding in

Model	En→De	En→Fr	En→It
Transformer	26.42	38.86	27.94
DPE-based Extension	27.47 (↑ 1.05)	39.42 (↑ 0.56)	28.35 (↑ 0.41)

Table 6: BLEU results for different IWSLT-14 Language pairs.

Transformers. First, we acknowledge that there are some extensions to our current work. Additional rules can be designed to handle different cases of word alignments generated by `FastAlign`. For example, cases such as Many-to-Many alignments and multi-word expressions are also frequently found in written text. Another possible extension would be to investigate more precise alignment tools in addition to `FastAlign`. However, it is important to note that we did not heavily invest in linguistic preprocessing because it requires too many resources. Extremely precise preprocessing might not be necessary as neural models are expected to still solve problems with limited access to domain information. When considering other alignment tools, we must also consider the efficiency and scalability of our solution.

Finally, we plan to explore injecting order information into other language processing models through DPE or a similar mechanism. Such information seems to be useful for tasks such as dependency parsing or sequence tagging.

References

Arianna Bisazza and Marcello Federico. 2016. Surveys: A survey of word reordering in statistical machine translation: Computational models and language phenomena. *Computational Linguistics*, 42(2):163–205.

Mauro Cettolo, Christian Girardi, and Marcello Federico. 2012. WIT3: Web inventory of transcribed and translated talks. In *Proceedings of the 16th Annual conference of the European Association for Machine Translation*, pages 261–268, Trento, Italy. European Association for Machine Translation.

Kehai Chen, Rui Wang, Masao Utiyama, and Eiichiro Sumita. 2019. Neural machine translation with reordering embeddings. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 1787–1799, Florence, Italy. Association for Computational Linguistics.

Kehai Chen, Rui Wang, Masao Utiyama, and Eiichiro Sumita. 2020. Explicit reordering for neural machine translation.

Yiming Cui, Shijin Wang, and Jianfeng Li. 2016. LSTM neural reordering feature for statistical machine translation. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 977–982, San Diego, California. Association for Computational Linguistics.

Jinhua Du and Andy Way. 2017. Pre-reordering for neural machine translation: helpful or harmful? *Prague Bulletin of Mathematical Linguistics*, (108):171–181.

Chris Dyer, Victor Chahuneau, and Noah A. Smith. 2013. A simple, fast, and effective reparameterization of IBM model 2. In *Proceedings of the 2013 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 644–648, Atlanta, Georgia. Association for Computational Linguistics.

Minwei Feng, J. Peter, and H. Ney. 2013. Advancements in reordering models for statistical machine translation. In *ACL*.

Sarthak Garg, Stephan Peitz, Udhyakumar Nallasamy, and Matthias Paulik. 2019. Jointly learning to align and translate with transformer models.

Yuki Kawara, Chenhui Chu, and Yuki Arase. 2020. Preordering encoding on transformer for translation. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*.

Guolin Ke, Di He, and Tie-Yan Liu. 2020. Rethinking positional encoding in language pre-training.

Philipp Koehn. 2009. *Statistical machine translation*. Cambridge University Press.

Philipp Koehn, Hieu Hoang, Alexandra Birch, Chris Callison-Burch, Marcello Federico, Nicola Bertoldi, Brooke Cowan, Wade Shen, Christine Moran, Richard Zens, Chris Dyer, Ondřej Bojar, Alexandra Constantin, and Evan Herbst. 2007. Moses: Open source toolkit for statistical machine translation. In *Proceedings of the 45th Annual Meeting of the Association for Computational Linguistics Companion Volume Proceedings of the Demo and Poster Sessions*, pages 177–180, Prague, Czech Republic. Association for Computational Linguistics.

Xuanqing Liu, Hsiang-Fu Yu, Inderjit Dhillon, and Cho-Jui Hsieh. 2020. Learning to encode position for transformer with continuous dynamical model.

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. Bleu: a method for automatic evaluation of machine translation. In *Proceedings of the*

40th Annual Meeting of the Association for Computational Linguistics, pages 311–318, Philadelphia, Pennsylvania, USA. Association for Computational Linguistics.

Rico Sennrich, Barry Haddow, and Alexandra Birch. 2016. Neural machine translation of rare words with subword units.

Peter Shaw, Jakob Uszkoreit, and Ashish Vaswani. 2018a. Self-attention with relative position representations. *CoRR*, abs/1803.02155.

Peter Shaw, Jakob Uszkoreit, and Ashish Vaswani. 2018b. Self-attention with relative position representations.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in neural information processing systems*, pages 5998–6008.

Chao Wang, Michael Collins, and Philipp Koehn. 2007. Chinese syntactic reordering for statistical machine translation. In *Proceedings of the 2007 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning (EMNLP-CoNLL)*, pages 737–745, Prague, Czech Republic. Association for Computational Linguistics.

Jinhua Zhu, Yingce Xia, Lijun Wu, Di He, Tao Qin, Wengang Zhou, Houqiang Li, and Tieyan Liu. 2020. Incorporating bert into neural machine translation. In *International Conference on Learning Representations*.

PAEG: Phrase-level Adversarial Example Generation for Neural Machine Translation

Juncheng Wan[1]*, **Jian Yang**[2*], **Shuming Ma**[3], **Dongdong Zhang**[3],
Weinan Zhang[1] †, **Yong Yu**[1] , **Zhoujun Li**[2]
[1]Shanghai Jiao Tong University
[2]State Key Lab of Software Development Environment, Beihang University
[3]Microsoft Research Asia
{junchengwan,wnzhang,yyu}@apex.sjtu.edu.cn;
{jiaya, lizj}@buaa.edu.cn; {shumma, dozhang}@microsoft.com

Abstract

While end-to-end neural machine translation (NMT) has achieved impressive progress, noisy input usually leads models to become fragile and unstable. Generating adversarial examples as the augmented data has been proved to be useful to alleviate this problem. Existing methods for adversarial example generation (AEG) are word-level or character-level, which ignore the ubiquitous phrase structure. In this paper, we propose a **P**hrase-level **A**dversarial **E**xample **G**eneration (PAEG) framework to enhance the robustness of the translation model. Our method further improves the gradient-based word-level AEG method by adopting a phrase-level substitution strategy. We verify our method on three benchmarks, including LDC Chinese-English, IWSLT14 German-English, and WMT14 English-German tasks. Experimental results demonstrate that our approach significantly improves translation performance and robustness to noise compared to previous strong baselines.

1 Introduction

Recently, neural machine translation (NMT) has effectively improved translation quality. NMT has shown state-of-the-art performance for many language pairs (Wu et al., 2016; Hassan et al., 2018; Vaswani et al., 2017). Various architectures (Sutskever et al., 2014; Bahdanau et al., 2015; Gehring et al., 2017; Vaswani et al., 2017) bring many appealing properties. Most NMT systems heavily rely on high-quality parallel data and perform poorly in noisy input. With the noise rising in the source sentence, NMT tends to be more vulnerable (Szegedy et al., 2014; Goodfellow et al., 2015), due to the output prediction of the decoder easily intervened by the other words (Cheng et al., 2018). A slight disturbance like a random permutation can damage the translation quality dramatically (Belinkov and Bisk, 2018). Even replacing a word with a synonym in the source input, the NMT model can be cheated and the target output can not be translated correctly.

Original Sentence	A cooked hot dog in a bun with ketchup and relish.
Word-level AEG	A cooked ***warm*** dog in a bun with ketchup and relish.
Phrase-level AEG	A cooked ***sausage rolls*** in a bun with ketchup and relish.

Table 1: An example of adversarial example generation (AEG). When the word "hot" is selected, word-level adversarial example generation method substitutes "hot" to "warm". The phrase-level method substitutes the whole phrase "hot dog" to "sausage rolls".

To improve the robustness of the NMT model, previous works propose to construct the adversarial examples by manipulating hidden features or discrete text input. These adversarial examples are used as augmented data for the training of the NMT model. To attack hidden features, Cheng et al. (2018) added perturbations in the input at the feature level for adversarial stability training. To generate discrete adversarial input, Ebrahimi et al. (2018) employed differentiable string-edit operations to rank adversarial changes. Belinkov and Bisk (2018) and Vaibhav et al. (2019) emulated naturally occurring errors in clean data as synthetic noise. Cheng et al. (2019) proposed a gradient-based method to craft adversarial examples, considering the similarity between the gradient related to the translation loss of input and the embedding difference of words.

Previous methods of adversarial example generation (AEG) are limited at the low level, like word-level, not considering the relationship between different words within a phrase. There is one example in Table 1. The word-level AEG method selects a

*Equal Contribution.
†Corresponding author.

Proceedings of the 29th International Conference on Computational Linguistics, pages 5085–5097
October 12–17, 2022.

vulnerable position then substitutes the corresponding word, omitting that the substituted word is in a phrase. Sometimes, the examples of this improper substitution can even harm the translation model.

Therefore, we propose a phrase-level adversarial example generation (PAEG) method, which improves a gradient-based word-level AEG method to phrase-level. Specifically, this method builds phrase-level candidates efficiently and substitutes phrases wholly with these candidates. We also propose to further improve this method with a bidirectional generation algorithm, as target-to-source adversarial pairs are a kind of slight perturbation of the original source-to-target translation. In practice, we generate adversarial examples after fixed intervals of NMT model updating (to convergence) and use them as new augmented data for the continual training of the model.

To verify the effectiveness of our method, we conduct experiments on three common benchmarks, i.e, LDC Chinese-English, IWSLT14 German-English, and WMT14 English-German. Experimental results demonstrate our method achieves significant improvements on translation quality and robustness to noisy inputs over the previous baselines including outstanding adversarial examples generation methods.

2 Phrase-level Adversarial Example Generation

In this section, we formulate the problem of adversarial example generation mathematically. First, our proposed method provides reliable candidates with the pre-trained model. Then, we use the gradient-based method to select vulnerable positions and substitutes at phrase level to generate adversarial examples. These examples are used as augmented data for the training of the NMT model. To further improve the performance, we extend our method to the bidirectional generation.

2.1 Problem Formulation

Let $A = \{(\mathbf{x}, \mathbf{z}), \mathbf{y}\}$ denotes the training data in NMT, where $(\mathbf{x}, \mathbf{z})$ are the encoder input and the decoder input, $\mathbf{y}$ the corresponding decoder output. To generate the corresponding adversarial examples $B = \{(\mathbf{x}', \mathbf{z}'), \mathbf{y}\}$, where only the input is slightly different from A, we need to limit the adversarial input $(\mathbf{x}', \mathbf{z}')$ semantically close to the original data.

Adversarial examples aim to cheat the model, making it predict wrong words. Therefore, given real output words $\mathbf{y}$, we construct an input to make the model predict the incorrect word $\mathbf{y}'(\mathbf{y}' \neq \mathbf{y})$. The process of adversarial example generation in NMT can be formulated as solving the following optimization problem:

$$\{(\mathbf{x}', \mathbf{z}') : \arg\max_{(\mathbf{x}', \mathbf{z}')} P(\mathbf{x}', \mathbf{z}'; \mathbf{y}, \theta), \\ dist((\mathbf{x}', \mathbf{z}'), (\mathbf{x}, \mathbf{z})) < \epsilon\} \quad (1)$$

where $dist$ is a measure function of the input, such as the semantic distance of sentence embeddings or edit distance, $P(\mathbf{x}', \mathbf{z}'; \mathbf{y}, \theta)$ is the maximal probability that the model predicts a wrong word $\mathbf{y}'$ such that $\mathbf{y}' \neq \mathbf{y}$ when the model is fed with $(\mathbf{x}', \mathbf{z}')$, θ is the model parameters, and ϵ is a sufficiently small distance.

Algorithm 1 Phrase-level Adversarial Example

Input: $\{(\mathbf{x}, \mathbf{z}), \mathbf{y}\}$ denotes input and output, θ_l^s and θ_l^t denote parameters of LMs, θ_m denotes parameters of the model, $\mathbb{D}$ denotes the phrase dictionary.
Output: phrase-level adversarial input: $(\mathbf{x}', \mathbf{z}')$.

Compute $\{\mathbf{g}_{x_i}\}_{i=1}^{|\mathbf{x}|}$ with $\mathbf{x}$, $\mathbf{z}$, $\mathbf{y}$ by Eq.(2).
$pos_x \longleftarrow$ positions of maximal $\{||\mathbf{g}_{x_i}||_2\}_{i=1}^{|\mathbf{x}|}$
for i in pos_x **do**
    Get $cand(\mathbf{x}_{ij})$ by $\mathbb{D}$ and θ_l^s.
    Substitute $\mathbf{x}_{ij}$ to $\mathbf{x}'_{ij}$ as Eq.(4).
end for
Compute $\{\mathbf{g}_{z_i}\}_{i=1}^{|\mathbf{z}|}$ with $\mathbf{x}'$, $\mathbf{z}$, $\mathbf{y}$ by Eq.(2).
Get attention matrix $\mathcal{M}$ by $\mathbf{x}'$, $\mathbf{z}$, $\mathbf{y}$, and θ_m.
Compute $\{P(j)\}_{j=1}^{|\mathbf{y}|}$ with $\mathcal{M}$ by Eq.(3).
$pos_z \longleftarrow$ sampling by $\{P(j)\}_{j=1}^{|\mathbf{y}|}$
for i in pos_z **do**
    Get $cand(\mathbf{z}_{ij})$ by $\mathbb{D}$, θ_l^t and θ_m.
    Substitute $\mathbf{z}_{ij}$ to $\mathbf{z}'_{ij}$ as Eq.(4).
end for
return $(\mathbf{x}', \mathbf{z}')$

2.2 Phrase Candidates from PLM

To guarantee the generated example $(\mathbf{x}', \mathbf{z}')$ is similar to the original example $(\mathbf{x}, \mathbf{z})$, two aspects are taken into account. One aspect is that the information in sentences should not change a lot. The other is to guarantee that words are similar. Therefore, high-quality candidates for words or phrases to be substituted should have similar semantic meanings to their original ones and be more fluent in the whole sentence.

To achieve this, one intuitive method is to select words with maximal prediction probability in the language model (LM), since LM predicts words based on the context. Cheng et al. (2019) uses a bidirectional LM trained on the monolingual part of the parallel corpus. However, a high-quality LM often needs billions of monolingual data to train like BERT (Devlin et al., 2019). It is unacceptable to spend much time and computational resources training reliable LMs. Therefore, we propose to utilize the knowledge of the pre-trained LM (PLM). In this paper, we use BERT as PLM.

In our paper, we use the notation $\mathbf{x}_{ij}$ as the phrase from position i to j in sentence $\mathbf{x}$, $cand(\mathbf{x}_{ij})$ as the phrase candidates of phrase $\mathbf{x}_{ij}$. When $i = j$, $\mathbf{x}_{ij}$ indicates the word x_i and $cand(\mathbf{x}_{ij})$ indicates the word candidates $cand(x_i)$. Besides, we use $\mathbb{D}_n$ as the n-gram phrase dictionary and $\mathbb{D}$ the union set of all $\mathbb{D}_n$.

In the ith position of the source input, we construct $cand(x_i)$ by selecting the top n_s tokens with maximal prediction probability in BERT when fed with $\mathbf{x}$, where x_i is masked. For the target input side, candidates consist of two parts. The first part is from BERT, which provides with n_t^l candidates. The second part is from the trained NMT model, which provides with n_t^m candidates. In this way, the candidate set $cand(z_j)$ of target input side consists of words fluent in the sentence and words conforming the translation of $\mathbf{x}$. In this paper, we set $n_s^l = 10$ and $n_t^l = n_t^m = 5$.

Given a phrase $\mathbf{x}_{ij}$, we construct phrase-level candidates $cand(\mathbf{x}_{ij})$. We first build the set of all probable phrase candidates as the Cartesian product of all $cand(x_k)$ $(k = i, i+1, \ldots, j)$. Then, we screen out unreasonable phrase candidates by the phrase dictionary $\mathbb{D}$. Candidates not in this dictionary are discarded.

To obtain the phrase dictionary $\mathbb{D}$, we introduce two methods. The first one is to use the syntax parser to parse the sentence into a syntax tree. Then, the leaf nodes of an n-leaf subtree is an n-gram phrase. The phrase dictionary $\mathbb{D}$ is the union of these n-gram phrase dictionary $\mathbb{D}_n$. The second method is to utilize the existing phrase extraction tool directly. In this paper, we take both of these two methods. The syntax parser we used is `nltk.parse`[1] and $n = 2, 3, 4$. The phrase extraction tool we used is `TextBlob`[2].

[1] https://www.nltk.org

[2] https://textblob.readthedocs.io/en/dev/

2.3 Select Vulnerable Positions

Instead of randomly selecting positions, we propose that the adversarial examples should select the most vulnerable positions in the sentence. Given a certain sentence, some NMT models may get worse translations when certain words or phrases are substituted.

Given that we train an NMT model with parameters θ_m and use negative log likelihood as the loss function with the input $\mathbf{x}$, $\mathbf{z}$ and the output $\mathbf{y}$, we can get the gradient vector $\mathbf{g}_{x_i}$ of token x_i over the training loss:

$$\mathbf{g}_{x_i} = \nabla_{e(x_i)} - \log P(\mathbf{y}|\mathbf{x}, \mathbf{z}; \theta_m) \quad (2)$$

where $e(x_i)$ is the embedding vector of token x_i.

Previous methods randomly choose positions in the source input. Since different positions have different gradient norms $||\mathbf{g}_{x_i}||_2$, if the gradient norm is large, the position is more unstable. Therefore, positions with large gradient norm are more vulnerable. For the source input, we select the top $\alpha_s|\mathbf{x}|$ positions with maximal gradient norm, where $\alpha_s \in (0, 1)$ is a ratio. [3]

To construct the target input $\mathbf{z}'$, we teach the model how to defend the attack from the source $\mathbf{x}'$. It is a reason that we choose n_t^m candidates from the NMT model on the target side. Selected target side positions should have the target counterpart of substituted source words in $\mathbf{x}'$. For example, if we substitute the word "drawing" to "eating" in the source input "Cezanne loved drawing apples ." ("Cezanne malt gerne äpfel ." in German), then we need to find the position of the corresponding translation "drawing" ("malt") and substitute it to an English word related to "eating", such as "isst".

This process is the inverse process of attention in NMT. Following (Cheng et al., 2019), we sample $\alpha_t|\mathbf{y}|$ $(\alpha_t \in (0, 1))$ relevant words influenced by the perturbed words in the source input $\mathbf{x}'$ as by sampling function $P(\cdot)$:

$$P(j) = \frac{\sum_i \mathcal{M}_{ij}\delta_{x_i \neq x'_i}}{\sum_k \sum_i \mathcal{M}_{ik}\delta_{x_i \neq x'_i}}, j = 1, \ldots, |\mathbf{y}| \quad (3)$$

where $\mathcal{M}_{ij}$ is the value of attention matrix between token x_i and token y_j from NMT model, $\delta_{x_i \neq x'_i}$ is 1 if $x_i \neq x'_i$ and 0 otherwise.

[3] The experimental results of how to choose positions are discussed in Appendix A.1.

2.4 Phrase-level Substitution

Since words in the same phrase have a close relationship, we substitute words at phrase level in the adversarial example generation. There are two aspects to consider. First, since synonymy phrases sharing the same meaning may have variant lengths, the feature representation of a phrase should be irrelevant to the length of the phrase. Besides, we need to choose the phrase from the candidate set that disturbs the model the most.

For the first consideration, we simply extract phrase-level features by averaging the word embeddings. For the second aspect, we adopt the gradient-based approach in (Cheng et al., 2019). To represent the whole gradient of the phrase, we also average all the gradients of words. Other feature engineering methods like max-pooling, concatenation, and element-wise product are also viable.

Formally, for the substitution of phrase $\mathbf{x}_{ij}$, the greedy approach based on the gradient is:

$$\mathbf{x}'_{ij} = \underset{\mathbf{c} \in cand(\mathbf{x}_{ij})}{\arg\max}\ sim(f^e(\mathbf{c}) - f^e(\mathbf{x}_{ij}), f^g(\mathbf{c})) \quad (4)$$

where sim is the similarity function, f^e is the feature representation of the phrase, f^g is the feature representation of the gradient of the phrase, and $cand(\mathbf{x}_{ij})$ the phrase candidates of $\mathbf{x}_{ij}$. In this paper, we use the "average" function for f^e and f^g [4] and cosine similarity as the similarity function.

Our phrase-level adversarial example generation process is shown in Algorithm 1. During the training of the NMT model, we generate adversarial examples periodically as augmented data. Note that we do not need training LMs for the source and target languages.

2.5 Bidirectional Generation

In practice, reversed adversarial examples from target-to-source translation can also be used as augmented data for the source-to-target translation. Therefore, we introduce a bidirectional generation method to boost our phrase-level adversarial example generation method.

Our bidirectional generation has two translation directions, source-to-target, and target-to-source. We use a universal encoder and decoder for these two directions as (Johnson et al., 2017). From the original data, we generate the adversarial examples for two directions. In each iteration, the adversarial examples are reversed and added to the dataset. The model is trained on the augmented dataset.

Algorithm 2 Bidirectional Generation

Input: $\{(\mathbf{x}, \mathbf{z}_l), \mathbf{y}\}$ denotes source-to-target input and output. $\{(\mathbf{y}, \mathbf{z}_r), \mathbf{x}\}$ denotes target-to-source input and output. **Gen** is the adversarial examples generator.

Output: augmented source-to-target data D_l and target-to-source data D_r.

1: Compute $\{\mathbf{g}_{x_i}\}_{i=1}^{|\mathbf{x}|}$ with $\mathbf{x}$, $\mathbf{z}$, $\mathbf{y}$ by Eq.(2).
2: $D_l \longleftarrow \{(\mathbf{x}, \mathbf{z}_l), \mathbf{y}\}$, $D_r \longleftarrow \{(\mathbf{y}, \mathbf{z}_r), \mathbf{x}\}$.
3: $\mathbf{x}', \mathbf{z}'_l \longleftarrow \text{Gen}(\mathbf{x}, \mathbf{z}_l)$, $\mathbf{y}', \mathbf{z}'_r \longleftarrow \text{Gen}(\mathbf{y}, \mathbf{z}_r)$
4: Add $\{(\mathbf{x}', \mathbf{z}'_l), \mathbf{y}\} \bigcup \{(\mathbf{z}'_r, \mathbf{y}'), \mathbf{y}\}$ to D_l and add $\{(\mathbf{y}', \mathbf{z}'_r), \mathbf{x}\} \bigcup \{(\mathbf{z}'_l, \mathbf{x}'), \mathbf{x}\}$ to D_r.
5: **return** (D_l, D_r)

Formally, we notate $(\mathbf{x}, \mathbf{z}_l, \mathbf{y})$ as the encoder input, decoder input and decoder output for source-to-target translation, $(\mathbf{y}, \mathbf{z}_r, \mathbf{x})$ as the encoder input, decoder input and decoder output for target-to-source translation. After generating the adversarial examples, we get $(\mathbf{x}', \mathbf{z}'_l, \mathbf{y})$ and $(\mathbf{y}', \mathbf{z}'_r, \mathbf{x})$. Then, the adversarial examples input are reversed and added to the training data of the other direction. For source-to-target training, we have three pairs of data $(\mathbf{x}, \mathbf{z}, \mathbf{y})$, $(\mathbf{x}', \mathbf{z}'_l, \mathbf{y})$, $(\mathbf{z}'_r, \mathbf{y}', \mathbf{y})$. They are respectively the original training data, the adversarial examples and the reversed adversarial examples from the other direction.

The phrase-level adversarial example generation of these two directions help mutually during the training. Our bidirectional generation algorithm is shown in Algorithm 2. It is worth noting that, in general, the training time of PAEG is not as much as double of PAEG without bidirectional generation, as the data from bidirectional generation has a similar distribution of the original adversarial samples.[5]

3 Experiments

We evaluate our method on three datasets, LDC Chinese-English, IWSLT14 German-English, and WMT14 English-German translation datasets. Then, we compare our method with baselines. At last, we do a detailed analysis of the different components of our method.

Limited by the number of pages, we have included the description of three datasets and the

[4] The reason for using "average" function is explained in Appendix A.2.

[5] There are more discussions about time consumption of bidirectional generation in Appendix 4.

Method	MT06	MT02	MT03	MT05	MT08	MT12	Avg.
Transformer (Vaswani et al., 2017)	43.52	43.17	44.06	44.45	36.27	35.07	41.09
Multilingual NMT (Johnson et al., 2017)	43.54	43.46	44.63	44.40	36.13	35.00	41.19
Word Dropout (Sennrich et al., 2016)†	43.96	44.02	44.55	44.70	36.49	35.33	41.51
SwitchOut (Wang et al., 2018a)†	43.83	44.36	45.02	44.85	36.53	35.45	41.67
AdvGen (Cheng et al., 2019)†	44.74	45.12	46.49	45.95	37.29	36.02	42.60
PAEG (this work)†	**45.49**	**45.76**	**47.58**	**46.83**	**38.18**	**36.91**	**43.46**

Table 2: Case-insensitive BLEU-4 scores (%) on LDC Zh→En task. Our method is compared with other baselines and *Transformer_base* model. Methods with "†" use adversarial examples for training.

Method	BLEU
Transformer (Vaswani et al., 2017)	34.20
Multilingual NMT (Johnson et al., 2017)	34.13
NT^{2}MT (Feng et al., 2018)	31.75
LightConv (Wu et al., 2019)	34.80
DynamicConv (Wu et al., 2019)	35.20
Word Dropout (Sennrich et al., 2016)†	34.72
SwitchOut (Wang et al., 2018a)†	34.83
AdvGen (Cheng et al., 2019)†	35.25
PAEG (this work)†	**35.65**

Table 3: Case-insensitive BLEU-4 scores (%) on IWSLT14 De→En task. Our method is compared with other baselines and *Transformer_small* model. Methods with "†" use adversarial examples for training.

Method	BLEU
Transformer (Vaswani et al., 2017)	28.40
Multilingual NMT (Johnson et al., 2017)	29.11
RNMT+ (Chen et al., 2018)	28.49
LightConv (Wu et al., 2019)	28.90
DynamicConv (Wu et al., 2019)	29.70
Word Dropout (Sennrich et al., 2016)†	29.30
SwitchOut (Wang et al., 2018a)†	29.40
AdvGen (Cheng et al., 2019)†	30.01
PAEG (this work)†	**30.49**

Table 4: Case-insensitive BLEU-4 scores (%) on WMT14 En→De task. Our method is compared with other baselines and *Transformer_big* model. Methods with "†" use adversarial examples for training.

training details in the Appendix B and Appendix C respectively.

3.1 Comparisons to Baseline Methods

We compare our method with NMT models without adversarial examples (Non-adv NMT) and using adversarial examples (Adv NMT). Our method gets significant translation improvement by statistical significance testing ($p < 0.05$) compared to relevant baselines.

Non-adv NMT **Multilingual NMT** (Johnson et al., 2017) is implemented with the Transformer model as the universal encoder and decoder. **NT^{2}MT** (Feng et al., 2018) uses a phrase attention mechanism with backbone model LSTM. We report the maximal result with out-of-domain dictionaries in the paper. **RNMT+** (Chen et al., 2018) is an enhanced version of RNN-based NMT model. **LightConv** (Wu et al., 2019) uses a lightweight convolution performing competitively to the Transformer. **DynamicConv** (Wu et al., 2019) leverages a dynamic convolution predicting separate convolution kernels.

Adv NMT **Word Dropout** (Sennrich et al., 2016) drops words randomly. We implement it on the token level, as recommended by the paper. **SwitchOut** (Wang et al., 2018a) randomly replaces words in both the source and target sentence with words from the vocabulary. We implement the hamming distance sampling method in the paper. **AdvGen** (Cheng et al., 2019) is an adversarial example generation method at the word-level. This method uses doubly adversarial input. We implement this method with the Transformer backbone, $\alpha_s = 25\%, \alpha_t = 50\%$ for LDC Chinese-English task, and $\alpha_s = 20\%, \alpha_t = 20\%$ for IWSLT14 German-English and WMT14 English-German.

Table 2 demonstrates the comparisons between our method with the above five baseline methods on LDC Chinese-English translation task. First, we compare our method with the Transformer. On average, PAEG can improve +2.37 BLEU points significantly. Then, we compare our method with methods of training with adversarial examples. On average, adversarial example generation methods (AdvGen and PAEG) utilizing the training information of the model greatly surpass the other methods (Word Dropout and SwitchOut). The reason is that the former approach is better at attacking vulnerable parts of the NMT model. Compared with the

Method	MT06	MT02	MT03	MT05	MT08	MT12	Avg.
PAEG	45.49	45.76	47.58	46.83	38.18	36.91	43.46
w/o bidirectional generation	45.52	45.53	46.96	46.72	38.10	36.85	43.24
w/o phrase-level substitution	44.03	44.02	45.63	45.35	37.21	35.51	41.96
w/o candidates from BERT	43.52	43.17	44.06	44.45	36.27	35.07	41.09

Table 5: Experiments on LDC Zh→En dataset to analyze the effect of different components of PAEG. We removed three components of PAEG step by step. The results show that phrase-level substitution is the most effective part.

state-of-the-art AEG method AdvGen, PAEG gets an improvement of +0.86 BLEU points.

In Table 3, we compare our method with the above eight baseline methods on the IWSLT14 German-English translation task. Compared with the backbone model Transformer, PAEG gets the gain of +1.45 BLEU points. Compared with methods built on top of Transformer, NT^{2}MT (Feng et al., 2018) with out-of-domain dictionaries suffers from a worse backbone model (LSTM). Multilingual NMT (Johnson et al., 2017) has a similar performance to the Transformer model. Compared with the other methods of training with adversarial examples, PAEG has the best performance. PAEG gets +0.8∼0.9 BLEU points improvement compared with AEG methods which do not leverage the training information of the model.

The comparisons on the WMT14 English-German task are in Table 4. Compared with *Transformer_big* model, PAEG has a notable gain of +2.09 BLEU points. PAEG consistently outperforms all three baselines training with adversarial examples, having around +0.5∼1.0 BLEU points improvement in this commonly used dataset.

3.2 Ablation Studies

Our proposed method PAEG is mainly affected by three components, the use of the pre-trained model, the phrase-level substitution, and the bidirectional adversarial example generation. We analyze the different components of PAEG by ablation studies.

Effect of Phrase-level Substitution We use the phrase-level substitution and there is +1.28 BLEU points improvement in Table 5, which is significant. Substituting words randomly from the top 10 word-level candidates can not guarantee consistency between words. What is worse is that random substitution may destroy the phrase structure and semantic consistency in the sentence.

For common languages, such as Chinese, English, and German, the ratio of phrases is non-negligible. Substituting at the phrase level does make the adversarial input more fluent and thus more closely approximates the real-world data. In this way, our method can teach the model to defend against the attack on the target side better.

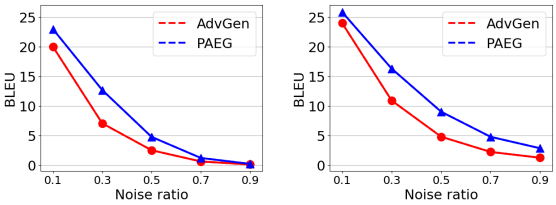

(a) (b)

Figure 1: Results on (a) random replacement noise and (b) random switch noise.

Effect of PLM To find out the impact of the pre-trained model, we use BERT to generate pseudo data. In Table 5, with the use of the pre-trained model BERT, the Transformer model has +0.87 BLEU points improvement. This proves that BERT provides more reliable candidates by pre-training on amounts of data. Compared with the LMs trained on millions of monolingual data, BERT can significantly leverage the contextual information to make the candidates appear fluent in the sentence.

Effect of Bidirectional Generation In Table 5, we add the bidirectional generation method to PAEG and there is +0.22 BLEU points improvement. This shows that the bidirectional generation has slight improvements. Considering that PAEG (without bidirectional generation) itself achieves a high BLEU score, the further improvement of bidirectional generation cannot be ignored.

3.3 Robustness to Noisy Inputs

To compare the robustness of different NMT models, we conduct three groups of experiments to simulate machine translation scenarios with noisy inputs by word replacement and switch. All experiments are conducted in the WMT14 English-German test set. Our method is compared with the representative word-level augmentation method AdvGen with the *Transformer_big* backbone.

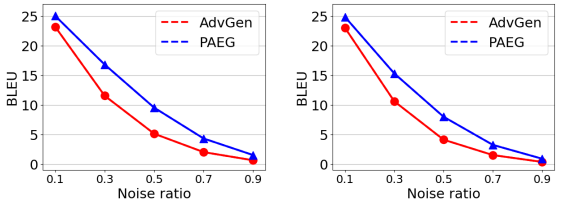

(a) (b)

Figure 2: Results on (a) word-level most similar synonym noise and (b) word-level least similar synonym noise.

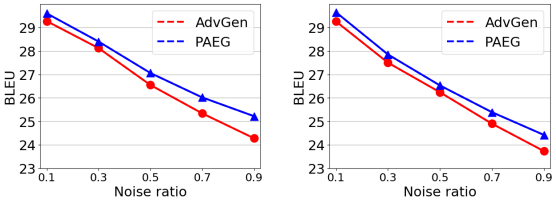

(a) (b)

Figure 3: Results on (a) phrase-level most similar synonym noise and (b) phrase-level least similar synonym noise.

Random Word Replacement/Switch Noise We first simulate the random replacement and switch noise, where a specific proportion (γ) of *positions* of the source sentence are selected uniformly and replace with random words in the source vocabulary (also uniformly). Such phenomenon is common in real-world scenarios, like onomatopoeia in speech recognition. We set $\gamma \in \{0.1, 0.3, 0.5, 0.7, 0.9\}$ to indicate the level of noise and test the sensitivity of NMT models in Figure 1(a) and Figure 1(b).

The analytic results show that our method PAEG improves the robustness of NMT models more than AdvGen, both to random replacement noise and switch noise. When the ratio of noise increases, the BLEU improvement gets consistently larger, which proves the effectiveness of PAEG. When the ratio of noise is high (0.7∼0.9), both methods degenerate into random translation machines. It can be attributed that excessive random noise impairs the source-side encoding.

Word-level Synonym Noise Another common noise in the translation system is synonym substitution, where the translation system is required to translate sentences consistently with subtle synonym difference. We first simulate this scenario with moderate word replacements. The selection of noisy positions is the same as random word replacement. Each to-be-replaced word matches the top 5 similar words by word similarity as the candidate set.[6] We add the most/least similar synonym noise by selecting the most/least similar word in the candidate set as the replacement. The noise ratio $\gamma \in \{0.1, 0.3, 0.5, 0.7, 0.9\}$ and sensitivity results are in Figure 2(a) and Figure 2(b).

The noise-BLEU curves have almost the same trend as random word replacement, which again proves the superior robustness of PAEG over AdvGen at word-level synonym noises. This is understandable because PAEG is inclusion of word-level alternatives.

[6]We use word embedding cosine similarity by pre-trained word embeddings GloVe (100 dimension) from flairNLP.

Word Type	Word-level	Phrase-level
NP (%)	18.69	20.11 (**+1.42**)
VP (%)	7.56	7.48 (-0.08)
PP (%)	16.43	17.53 (+1.10)
ADJP (%)	1.76	1.98 (+0.22)

Table 6: Experiments on the LDC Zh→En task to compare the phrase-level and word-level AEG methods by the ratio of noun/verb/prepositional/adjective phrases (NP/VP/PP/ADJP), in the hypothesis.

Phrase-level Synonym Noise In addition, we would like to verify how robust our method is to phrase-level synonym noises, where phrase structures are destroyed by word-level synonyms replacement, such as the case in Table 1. For this purpose, we select ϵ ratio of *phrases* uniformly and replace them with similar words in the source language vocabulary. The noise ratio $\epsilon \in \{0.1, 0.3, 0.5, 0.7, 0.9\}$. Figure 3(a) and Figure 3(b) show that with the increase of phrase-level noise, PAEG gets more BLEU improvement both in most and least similar synonym noise settings. Our method is more resistant to the destruction of phrase structures, which is proved again in the following section.

3.4 Analysis of Phrase-level Substitution

Phrase-level substitution shows remarkable improvement of the BLEU scores on average. In this subsection, we analyze the translation details and discuss the reason for such an improvement.

Phrase Translation First, we make the statistics of the ratio of phrases η of the (generated) hypothesis in LDC Chinese-English translation in Table 6. In a text $\mathbf{x}$, the ratio of phrases η is defined as

Original	SRC: 对正在实施的(家庭/family)(暴力/violence)，(受害人/victim)可以(请求/ask)公安(机关/organ)救助。 TGT: With regard to ongoing family violence, the victim may ask the public security organ for help.
AdvGen	SRC: 对正在实施的(家庭/family)(迫害/abuse)，(受害人/victim)可以(要求/require)公安(机关/organ)救助。 TGT: With regard to ongoing family **abuse**, the **victim** may **require** the public security organ for help.
PAEG	SRC: 对正在实施的(家庭/family)(虐待/abuse)，(受害人/victim)可以(请求/ask)公安(警察/police)救助。 TGT: With regard to ongoing **domestic abuse**, the **sufferer** may ask the public security **police** for help.

Table 7: Comparison of our PAEG method and AdvGen method on the LDC Zh→En dataset. Tokens with underline are substituted by the model as a word. Tokens with wave lines are substituted by the model as a phrase entirely. Chinese tokens and their English counterparts are in brackets (Chinese/English).

N-gram	Word-level	Phrase-level
1-gram BLEU	79.56	79.78 (+0.22)
2-gram BLEU	52.87	53.59 (+0.72)
3-gram BLEU	34.49	35.65 (**+1.16**)
4-gram BLEU	24.22	25.03 (+0.81)

Table 8: Experiments on the LDC Zh→En to compare the phrase-level and word-level AEG method in n-gram BLEU scores. Phrase-level method improves n-gram ($n > 1$) BLEU scores more.

the sum of the phrase lengths in $\mathbf{x}$ divided by the text length $|\mathbf{x}|$. For the word-level AEG method, the η of noun phrases (NP) is 18.69% on average. While for the phrase-level method, the ratio of NP is remarkably $20.11\%(+1.42\%)$. Besides, the η of prepositional phrases (PP) also increases 1.1% by phrase-level substitution.

These results show that the NMT model trained on PAEG considers more about phrases, especially NP and PP. Phrase-level substitution prevents the damage to the structure of phrases, guarantee the normal ratio of phrases in the augmented dataset, and thus teaches the decoder to generate phrases.

N-gram Accuracy Besides, we analyze the improvements for different n-gram BLEU scores in Table 8. PAEG improve the 3-gram BLEU greatly (+1.16 points) over the word-level method. 2-gram and 3-gram BLEU also get moderate improvements (+0.7∼0.8 points), much greater than 1-gram BLEU. These results verify that, using phrase-level strategy, longer grams can be translated more accurately (to match the phrases in the references).

Case Study In Table 7, there is a case of the example generating process from the LDC dataset. On the target side, AdvGen substituted "violence" to "abuse". PAEG selected the 6-th position of the target sentence and substituted "family violence" to "domestic abuse" entirely. Though "family abuse" does not violate the original meaning, the substitution "domestic abuse" is more reasonable.

4 Related Work

Adversarial training for neural networks has been studied recently (Szegedy et al., 2014; Goodfellow et al., 2015). Similar ideas are applied into natural language processing (Goyal et al., 2016; Li et al., 2017; Yang et al., 2018; Cheng et al., 2018, 2019, 2020; Namysl et al., 2020; Croce et al., 2020; Wang et al., 2020a; Zang et al., 2020; Ding et al., 2020). Specifically, adversarial example generation (Fadaee et al., 2017; Ebrahimi et al., 2018; Wang et al., 2018b; Cheng et al., 2020; Zou et al., 2020; Zheng et al., 2020; Hidey et al., 2020; Zhang et al., 2021; Lai et al., 2022) is proved to be useful to train a robust NMT system. Recently, Cheng et al. (2019) adopted a gradient-based method to craft adversarial examples at word level, using the adversarial source input to attack while the target input to defend the model.

Our bidirectional generation method is similar to multilingual NMT training. Multilingual NMT models (Dong et al., 2015; Luong et al., 2016; Johnson et al., 2017; Wang et al., 2020b; Zhang et al., 2020; Zhu et al., 2020; Siddhant et al., 2020) are trained over multiple language pairs with parameter sharing, such as using the same encoder/decoder for different source/target languages (Johnson et al., 2017), using one encoder and separate decoders to translate one language to multiple languages (Dong et al., 2015), and sharing an attention mechanism (Firat et al., 2016) across multiple language pairs. In this work, we use the adversarial examples generated from the other direction to improve the robustness of the original translation direction.

5 Conclusion

In this work, we propose a phrase-level adversarial example generation method. Our goal is to improve the fluency of the adversarial examples. We improve a gradient-based word-level method with phrase-level candidate construction, overall substi-

tution strategy, and bidirectional generation. We verify our method on Chinese-English, German-English, and English-German corpus, and the results show that PAEG can improve both translation quality and robustness to noisy inputs significantly.

References

Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. 2015. Neural machine translation by jointly learning to align and translate. In *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*.

Yonatan Belinkov and Yonatan Bisk. 2018. Synthetic and natural noise both break neural machine translation. In *6th International Conference on Learning Representations, ICLR 2018, Vancouver, BC, Canada, April 30 - May 3, 2018, Conference Track Proceedings*. OpenReview.net.

Mia Xu Chen, Orhan Firat, Ankur Bapna, Melvin Johnson, Wolfgang Macherey, George Foster, Llion Jones, Mike Schuster, Noam Shazeer, Niki Parmar, Ashish Vaswani, Jakob Uszkoreit, Lukasz Kaiser, Zhifeng Chen, Yonghui Wu, and Macduff Hughes. 2018. The best of both worlds: Combining recent advances in neural machine translation. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 76–86, Melbourne, Australia. Association for Computational Linguistics.

Yong Cheng, Lu Jiang, and Wolfgang Macherey. 2019. Robust neural machine translation with doubly adversarial inputs. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 4324–4333, Florence, Italy. Association for Computational Linguistics.

Yong Cheng, Lu Jiang, Wolfgang Macherey, and Jacob Eisenstein. 2020. AdvAug: Robust adversarial augmentation for neural machine translation. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 5961–5970, Online. Association for Computational Linguistics.

Yong Cheng, Zhaopeng Tu, Fandong Meng, Junjie Zhai, and Yang Liu. 2018. Towards robust neural machine translation. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1756–1766, Melbourne, Australia. Association for Computational Linguistics.

Danilo Croce, Giuseppe Castellucci, and Roberto Basili. 2020. GAN-BERT: Generative adversarial learning for robust text classification with a bunch of labeled examples. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 2114–2119, Online. Association for Computational Linguistics.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, Minneapolis, Minnesota. Association for Computational Linguistics.

Ning Ding, Dingkun Long, Guangwei Xu, Muhua Zhu, Pengjun Xie, Xiaobin Wang, and Haitao Zheng. 2020. Coupling distant annotation and adversarial training for cross-domain Chinese word segmentation. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 6662–6671, Online. Association for Computational Linguistics.

Daxiang Dong, Hua Wu, Wei He, Dianhai Yu, and Haifeng Wang. 2015. Multi-task learning for multiple language translation. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 1723–1732, Beijing, China. Association for Computational Linguistics.

Javid Ebrahimi, Daniel Lowd, and Dejing Dou. 2018. On adversarial examples for character-level neural machine translation. In *Proceedings of the 27th International Conference on Computational Linguistics*, pages 653–663, Santa Fe, New Mexico, USA. Association for Computational Linguistics.

Marzieh Fadaee, Arianna Bisazza, and Christof Monz. 2017. Data augmentation for low-resource neural machine translation. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 567–573, Vancouver, Canada. Association for Computational Linguistics.

Jiangtao Feng, Lingpeng Kong, Po-Sen Huang, Chong Wang, Da Huang, Jiayuan Mao, Kan Qiao, and Dengyong Zhou. 2018. Neural phrase-to-phrase machine translation. *CoRR*, abs/1811.02172.

Orhan Firat, Kyunghyun Cho, and Yoshua Bengio. 2016. Multi-way, multilingual neural machine translation with a shared attention mechanism. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 866–875, San Diego, California. Association for Computational Linguistics.

Jonas Gehring, Michael Auli, David Grangier, Denis Yarats, and Yann N. Dauphin. 2017. Convolutional sequence to sequence learning. In *Proceedings of the 34th International Conference on Machine Learning, ICML 2017, Sydney, NSW, Australia, 6-11 August 2017*, volume 70 of *Proceedings of Machine Learning Research*, pages 1243–1252. PMLR.

Ian J. Goodfellow, Jonathon Shlens, and Christian Szegedy. 2015. Explaining and harnessing adversarial examples. In *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*.

Anirudh Goyal, Alex Lamb, Ying Zhang, Saizheng Zhang, Aaron C. Courville, and Yoshua Bengio. 2016. Professor forcing: A new algorithm for training recurrent networks. In *Advances in Neural Information Processing Systems 29: Annual Conference on Neural Information Processing Systems 2016, December 5-10, 2016, Barcelona, Spain*, pages 4601–4609.

Hany Hassan, Anthony Aue, Chang Chen, Vishal Chowdhary, Jonathan Clark, Christian Federmann, Xuedong Huang, Marcin Junczys-Dowmunt, William Lewis, Mu Li, Shujie Liu, Tie-Yan Liu, Renqian Luo, Arul Menezes, Tao Qin, Frank Seide, Xu Tan, Fei Tian, Lijun Wu, Shuangzhi Wu, Yingce Xia, Dongdong Zhang, Zhirui Zhang, and Ming Zhou. 2018. Achieving human parity on automatic chinese to english news translation. *CoRR*, abs/1803.05567.

Christopher Hidey, Tuhin Chakrabarty, Tariq Alhindi, Siddharth Varia, Kriste Krstovski, Mona Diab, and Smaranda Muresan. 2020. DeSePtion: Dual sequence prediction and adversarial examples for improved fact-checking. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 8593–8606, Online. Association for Computational Linguistics.

Melvin Johnson, Mike Schuster, Quoc V. Le, Maxim Krikun, Yonghui Wu, Zhifeng Chen, Nikhil Thorat, Fernanda Viégas, Martin Wattenberg, Greg Corrado, Macduff Hughes, and Jeffrey Dean. 2017. Google's multilingual neural machine translation system: Enabling zero-shot translation. *Transactions of the Association for Computational Linguistics*, 5:339–351.

Diederik P. Kingma and Jimmy Ba. 2015. Adam: A method for stochastic optimization. In *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*.

Siyu Lai, Zhen Yang, Fandong Meng, Xue Zhang, Yufeng Chen, Jinan Xu, and Jie Zhou. 2022. Generating authentic adversarial examples beyond meaning-preserving with doubly round-trip translation. In *Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL 2022, Seattle, WA, United States, July 10-15, 2022*, pages 4256–4266. Association for Computational Linguistics.

Jiwei Li, Will Monroe, Tianlin Shi, Sébastien Jean, Alan Ritter, and Dan Jurafsky. 2017. Adversarial learning for neural dialogue generation. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 2157–2169, Copenhagen, Denmark. Association for Computational Linguistics.

Minh-Thang Luong, Quoc V. Le, Ilya Sutskever, Oriol Vinyals, and Lukasz Kaiser. 2016. Multi-task sequence to sequence learning. In *4th International Conference on Learning Representations, ICLR 2016, San Juan, Puerto Rico, May 2-4, 2016, Conference Track Proceedings*.

Marcin Namysl, Sven Behnke, and Joachim Köhler. 2020. NAT: Noise-aware training for robust neural sequence labeling. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 1501–1517, Online. Association for Computational Linguistics.

Rico Sennrich, Barry Haddow, and Alexandra Birch. 2016. Edinburgh neural machine translation systems for WMT 16. In *Proceedings of the First Conference on Machine Translation: Volume 2, Shared Task Papers*, pages 371–376, Berlin, Germany. Association for Computational Linguistics.

Aditya Siddhant, Ankur Bapna, Yuan Cao, Orhan Firat, Mia Chen, Sneha Kudugunta, Naveen Arivazhagan, and Yonghui Wu. 2020. Leveraging monolingual data with self-supervision for multilingual neural machine translation. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 2827–2835, Online. Association for Computational Linguistics.

Ilya Sutskever, Oriol Vinyals, and Quoc V. Le. 2014. Sequence to sequence learning with neural networks. In *Advances in Neural Information Processing Systems 27: Annual Conference on Neural Information Processing Systems 2014, December 8-13 2014, Montreal, Quebec, Canada*, pages 3104–3112.

Christian Szegedy, Wojciech Zaremba, Ilya Sutskever, Joan Bruna, Dumitru Erhan, Ian J. Goodfellow, and Rob Fergus. 2014. Intriguing properties of neural networks. In *2nd International Conference on Learning Representations, ICLR 2014, Banff, AB, Canada, April 14-16, 2014, Conference Track Proceedings*.

Vaibhav Vaibhav, Sumeet Singh, Craig Stewart, and Graham Neubig. 2019. Improving robustness of machine translation with synthetic noise. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 1916–1920, Minneapolis, Minnesota. Association for Computational Linguistics.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, December 4-9, 2017, Long Beach, CA, USA*, pages 5998–6008.

Rui Wang, Xuemeng Hu, Deyu Zhou, Yulan He, Yuxuan Xiong, Chenchen Ye, and Haiyang Xu. 2020a. Neural topic modeling with bidirectional adversarial training. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*,

pages 340–350, Online. Association for Computational Linguistics.

Xinyi Wang, Hieu Pham, Zihang Dai, and Graham Neubig. 2018a. SwitchOut: an efficient data augmentation algorithm for neural machine translation. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 856–861, Brussels, Belgium. Association for Computational Linguistics.

Xinyi Wang, Hieu Pham, Zihang Dai, and Graham Neubig. 2018b. SwitchOut: an efficient data augmentation algorithm for neural machine translation. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 856–861, Brussels, Belgium. Association for Computational Linguistics.

Xinyi Wang, Yulia Tsvetkov, and Graham Neubig. 2020b. Balancing training for multilingual neural machine translation. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 8526–8537, Online. Association for Computational Linguistics.

Felix Wu, Angela Fan, Alexei Baevski, Yann N. Dauphin, and Michael Auli. 2019. Pay less attention with lightweight and dynamic convolutions. In *7th International Conference on Learning Representations, ICLR 2019, New Orleans, LA, USA, May 6-9, 2019*. OpenReview.net.

Yonghui Wu, Mike Schuster, Zhifeng Chen, Quoc V. Le, Mohammad Norouzi, Wolfgang Macherey, Maxim Krikun, Yuan Cao, Qin Gao, Klaus Macherey, Jeff Klingner, Apurva Shah, Melvin Johnson, Xiaobing Liu, Lukasz Kaiser, Stephan Gouws, Yoshikiyo Kato, Taku Kudo, Hideto Kazawa, Keith Stevens, George Kurian, Nishant Patil, Wei Wang, Cliff Young, Jason Smith, Jason Riesa, Alex Rudnick, Oriol Vinyals, Greg Corrado, Macduff Hughes, and Jeffrey Dean. 2016. Google's neural machine translation system: Bridging the gap between human and machine translation. *CoRR*, abs/1609.08144.

Zhen Yang, Wei Chen, Feng Wang, and Bo Xu. 2018. Improving neural machine translation with conditional sequence generative adversarial nets. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 1346–1355, New Orleans, Louisiana. Association for Computational Linguistics.

Yuan Zang, Fanchao Qi, Chenghao Yang, Zhiyuan Liu, Meng Zhang, Qun Liu, and Maosong Sun. 2020. Word-level textual adversarial attacking as combinatorial optimization. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 6066–6080, Online. Association for Computational Linguistics.

Biao Zhang, Philip Williams, Ivan Titov, and Rico Sennrich. 2020. Improving massively multilingual neural machine translation and zero-shot translation. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 1628–1639, Online. Association for Computational Linguistics.

Xinze Zhang, Junzhe Zhang, Zhenhua Chen, and Kun He. 2021. Crafting adversarial examples for neural machine translation. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing, ACL/IJCNLP 2021, (Volume 1: Long Papers), Virtual Event, August 1-6, 2021*, pages 1967–1977. Association for Computational Linguistics.

Xiaoqing Zheng, Jiehang Zeng, Yi Zhou, Cho-Jui Hsieh, Minhao Cheng, and Xuanjing Huang. 2020. Evaluating and enhancing the robustness of neural network-based dependency parsing models with adversarial examples. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 6600–6610, Online. Association for Computational Linguistics.

Changfeng Zhu, Heng Yu, Shanbo Cheng, and Weihua Luo. 2020. Language-aware interlingua for multilingual neural machine translation. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 1650–1655, Online. Association for Computational Linguistics.

Wei Zou, Shujian Huang, Jun Xie, Xinyu Dai, and Jiajun Chen. 2020. A reinforced generation of adversarial examples for neural machine translation. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 3486–3497, Online. Association for Computational Linguistics.

A Details of PAEG

A.1 Vulnerable Positions

In this work, there is an assumption that substituting words in vulnerable positions (positions with greater gradient norm) is more likely to add perturbation to model training. In our experiments, we have tried sampling the positions of source phrases randomly and found that vulnerable positions is better (+0.2∼0.3 BLEU points).

A.2 Phrase Embedding

In the experiments, "max-pooling" has been explored to get the phrase embedding/gradients from word embedding/gradients, and it has a similar result as the "average" operation, within 0.2 BLEU points. In the implementation, "max-pooling" is slower than the "average" (using PyTorch 1.7), therefore we choose "average" for convenience.

B Dataset

LDC Chinese-English Task This is a dataset of 1.2M training sequence pairs. The LDC numbers are 2002E17, 2002E18, 2004T08, 2005T10, 2005T34, 2006E17, 2006T06, and 2008T18[7]. We choose the NIST 2006 as the validation set, which has 1664 sentences, and the NIST 2002, NIST 2003, NIST 2005, NIST 2008, NIST 2012 as the test sets, which contain 877, 919, 1082, 1357, 2190 sentences respectively.

IWSLT14 German-English Task This dataset comes from translated TED talks. This dataset contains roughly 160K pairs as the training set, 7K pairs as the validation set, and 7K pairs as the test set, respectively. We take the IWSLT14 test set as the test set.

WMT14 English-German Task The training data has 4.5M sentence pairs. We use the newstest2013 as the valid set and the newstest2014 as the test set.

C Training Details

Our backbone model is the Transformer model (Vaswani et al., 2017). The NMT model consists of a Transformer encoder and a Transformer decoder. The pre-trained LM is BERT-based[8]. We use `nltk.parse` to build the syntax tree and extract the phrases of length $2, 3, 4$. Besides, we use `TextBlob` to extract the noun phradses and merge other phrases (from `nltk.parse`) to build the phrase dictionary.

LDC Chinese-English Translation We use our in-house Chinese word-breaker toolkit to segment Chinese data. We use byte pair encoding (BPE) to encode sentences with a shared token vocabulary of 51K sub-word tokens. The size of the phrase vocabulary is 1.2M for Chinese and 0.9M for English. We limit the maximum sentence length up to 256 words. We apply Adam (Kingma and Ba, 2015) with $\beta_1 = 0.9$ and $\beta_2 = 0.98$ to train models for 80 epochs and select the best model parameters according to the model performance on the valid set. We use *Transformer_base* setting: embedding size as 512, feed-forward network (FFN) size as 2048, attention heads as 8, learning rate as 0.1, batch size as 6144, and dropout rate as 0.1. We use the warm-up strategy with 4000 warm-up steps. We report case-insensitive tokenized BLEU-4 scores with Moses[9].

IWSLT14 German-English Translation We use BPE to encode sentences with a shared vocabulary of 10K sub-word tokens. The phrase vocabulary of German is of size 0.4M and English of size 0.4M. We limit the maximum sentence length up to 256 words. We apply Adam with $\beta_1 = 0.9$ and $\gamma_2 = 0.98$ to train models for 100 epochs and select the best model parameters according to the model performance on the valid set. We use *Transformer_small* setting: embedding size as 512, FFN size as 1024, attention heads as 4, learning rate as 0.1, batch size as 6144, and dropout rate as 0.3. We use the warm-up strategy with 4000 warm-up steps.

WMT14 English-German Translation We use BPE to encode sentences with a shared vocabulary of 10K sub-word tokens. The phrase vocabulary of German is of size 0.7M and English of size 0.4M. We limit the maximum sentence length up to 256 words. We apply Adam with $\beta_1 = 0.9$ and $\beta_2 = 0.98$ to train models for 50 epochs and select the best model parameters according to the model performance on the valid set. We use *Transformer_big* setting: embedding size as 1024, FFN

[7] https://catalog.ldc.upenn.edu/byproject

[8] https://github.com/huggingface/transformers

[9] https://github.com/moses-smt/mosesdecoder/blob/master/scripts/tokenizer/tokenizer.perl

size as 4096, attention heads as 16, learning rate as 0.1, batch size as 6144, and dropout rate as 0.1. We use the warm-up strategy with 4000 warm-up steps.

D Training Time Analysis

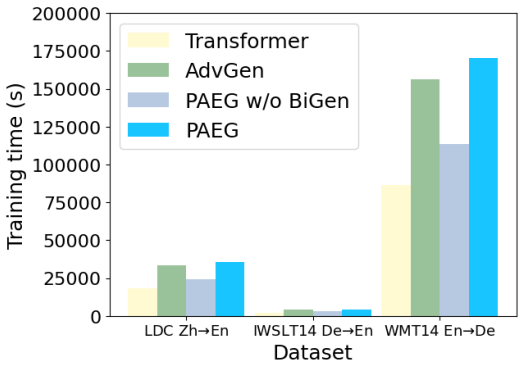

Figure 4: Training time of NMT models. All experiments on the three translation datasets are conducted on 8 NVIDIA 32G V100 GPUs and we set the batch size to fill the GPU memory.

As our method uses augmented data, one concern is whether the training time increases too much. We record the time consumption of our method as well as AdvGen and Transformer. All experiments on the three translation datasets are conducted on 8 NVIDIA 32G V100 GPUs and we set the batch size to fill the GPU memory.

The results are shown in Figure 4. The experiments show that AdvGen uses around the double time of training a Transformer, as it trains two (source and target) language models and generates adversarial data. Our method utilizes a pre-trained language model and thus saves the time of training the language model. Our method without bidirectional generation (BiGen) is faster than AdvGen. Even using bidirectional generation, our method is only slightly slower than AdvGen. Besides, the training time of PAEG is not exactly double of that of PAEG w/o BiGen, which is reasonable as the data from bidirectional generation do not deviate too much from the distribution of the original adversarial samples.

Noise-robust Cross-modal Interactive Learning with Text2image Mask for Multi-modal Neural Machine Translation

Junjie Ye[1,2], Junjun Guo[1,2]*, Yan Xiang[1,2], Kaiwen Tan[1,2], Zhengtao Yu[1,2]
[1] Faculty of Information Engineering and Automation,
Kunming University of Science and Technology, Kunming, China
[2] Yunnan Key Laboratory of Artificial Intelligence, Kunming, China
junjieye.cdx@qq.com, guojjgb@163.com,
sharonxiang@126.com, kwtan0909@qq.com, ztyu@hotmail.com

Abstract

Multi-modal neural machine translation (MNMT) aims to improve textual level machine translation performance in the presence of text-related images. Most of the previous works on MNMT focus on multi-modal fusion methods with full visual features. However, text and its corresponding image may not match exactly, visual noise is generally inevitable. The irrelevant image regions may mislead or distract the textual attention and cause model performance degradation. This paper proposes a noise-robust multi-modal interactive fusion approach with cross-modal relation-aware mask mechanism for MNMT. A text-image relation-aware attention module is constructed through the cross-modal interaction mask mechanism, and visual features are extracted based on the text-image interaction mask knowledge. Then a noise-robust multi-modal adaptive fusion approach is presented by fusion the relevant visual and textual features for machine translation. We validate our method on the Multi30K dataset. The experimental results show the superiority of our proposed model, and achieve the state-of-the-art scores in all En-De, En-Fr and En-Cs translation tasks [1].

1 Introduction

Multi-modal Neural Machine Translation (MNMT) aims to optimize the conventional text-only machine translation systems by using multi-modal information (eg., image, video, sound), which has received growing research attentions in the fields of CV and NLP, recently. A reasonable assumption is that visual information is helpful to improve textual-level machine translation (Elliott et al., 2017; Barrault et al., 2018; Ye and Guo, 2022), and many studies have been carried out to conduct the benefits of image for NMT (Caglayan et al., 2019; Yin et al., 2020; Li et al., 2021a). As expected, the fusion of visual information actually improves the performance of machine translation (Caglayan et al., 2019).

Most existing MNMT methods mainly focus on how to design a excellent multi-modal fusion framework to bridge the semantic gap between image and text, while visual noise is often ignored. Unfortunately, it is often difficult to obtain the image that exactly match the textual information. What is worse, image information and textual information may even be weakly correlated with each other. Visual noise is generally unavoidable(Li et al., 2022; Yao and Wan, 2020). As shown in Figure 1 (left), objects such as old man, brown hat and bench included in the image, these visual objects correspond to the '*old man*', '*brown hat*' and '*bench*' in the source sentence, which is the useful visual information for machine translation. However, the image also contains some irrelevant visual information (e.g., tree, flower, grasse) for the source sentence, the mismatched visual-textual information may distract the multi-modal fusion and then lead to machine translation performance decay. Therefore, it is necessary to consider noise-robust text-image fusion problem for MNMT.

How to effectively and efficiently extract useful visual information is one of the core issues of MNMT, there are three main multi-modal fusion methods: 1) Multi-modal attention mechanism, such as cross-modal interactive attention mechanism (Kwon et al., 2020; Song et al., 2021; Zhao et al., 2021) and adaptive feature selection mechanism (Wang and Xiong, 2021; Zhao et al., 2022; Li et al., 2022) between visual features and textual features. 2) Multi-modal Transformer fusion methods, which utilizes Transformer to encode textual features and visual features separately (Takushima et al., 2019; Nishihara et al., 2020), and then a multi-head cross-modal attention mechanism (Yao and Wan, 2020; Gain et al., 2021; Li et al., 2021a) is

*Corresponding author.
[1]https://github.com/nlp-mmt/Noise-robust-Text2image-Mask

Proceedings of the 29th International Conference on Computational Linguistics, pages 5098–5108
October 12–17, 2022.

Source *(En)*: the old man in the brown hat is sitting on the bench .
Target *(Fr)*: le vieil homme au chapeau brun est assis sur le banc .

Figure 1: An example of an En→Fr translation that illustrates the need to consider for image noise in the translation model.

adopted to integrate them. 3) Gating fusion methods (Yin et al., 2020; Lin et al., 2020; Li et al., 2021b), which are leveraged to ensure both textual semantic representations and visual semantic representations are consistent with each other. Above existing methods mainly focus on designing multi-modal feature fusion architectures by leveraging visual information to enhance traditional machine translation, however, visual noise problem is ignored.

This paper endeavors to address visual noise-robust multi-modal fusion problem for MNMT, we attempt to explore robust multi-modal interactive fusion strategy with cross-modal relation-aware mask mechanism for MNMT in Transformer framework. Concretely, a text-image relation-aware attention module is constructed in the visual transformer encoder by cross-modal interactive mask mechanism, and the visual features are extracted based on text-to-image interactive mask knowledge. Then a noise-robust multi-modal fusion approach is adopted to integrate visual features into seq2seq framework more efficiently and effectively. Comparing with previous works, the major contributions of our paper are three-fold.

- A noise-robust multi-modal fusion approach is proposed with cross-modal relation-aware mask for MNMT. To the best of our knowledge, it is the first attempt to explore mask-based multi-modal representation for MNMT.
- A text-image relation-aware module is constructed with cross-modal interaction masking mechanism to obtain text-image interaction mask knowledge for noise-robust multi-modal representation and fusion in noisy scenes.
- The extensive experimental results show that our proposed model outperforms other state-of-the-art MNMT approaches and significantly improves machine translation performance on En-De, En-Fr and En-Cs translation tasks. Furthermore, we emphasize the interpretability of the model, the in-depth analysis of the experimental results show the effectiveness of our proposed method.

2 Related Work

MNMT Early attempts mainly focused on RNN-based encoder-decoder architecture with attention (Huang et al., 2016; Calixto et al., 2017; Delbrouck and Dupont, 2017). Recently, Transformer-based seq2seq framework has achieved significant improvement for MNMT. Zhao et al. (2021) utilized object detection features with an additional region-dependent attention mechanism to fusion visual regional features and textual features; Nishihara et al. (2020) presented a supervised cross-modal attention module to align textual features and visual features; Song et al. (2021) employed a co-attention graph updating module at each Transformer encoder layer to align multi-modal features. Yao and Wan (2020) used multi-modal Tranformer to align both visual features and textual features; Yin et al. (2020) proposed a graph-based MNMT approach to extract multi-model features through text-image gating attention mechanism; Lin et al. (2020) adopted a gating mechanism to fuse visual features extracted by a dynamic context-guided capsule network;

All the above methods focus on multimodal feature fusion methods, and they assume that visual information is closely related to textual information, which heavily restricts their robustness. However, text and its corresponding image may not match exactly, visual noise is generally inevitable. In this work, we systematically investigate whether masking visual noise helps machine translation.

Mask Strategy Mask strategy is one of the most effective ways of representation learning, which has been widely used in vision and textual pre-training models. We summarize the existing mask strategies in the three aspects as follows: 1) Vision mask-based pre-trained models (Li et al., 2021c; Peng and Harwath, 2022; Xie et al., 2021), the main purpose is to mask image patch-level for better visual robust-representation learning. 2) Textual mask-based pre-trained models (Joshi et al., 2020; Fu et al., 2022; Devlin et al., 2019), the tokens of the input sentences are randomly masked, and then are predicted in decoder, which aims to generate more fine-grained textual representations. 3) Cross-

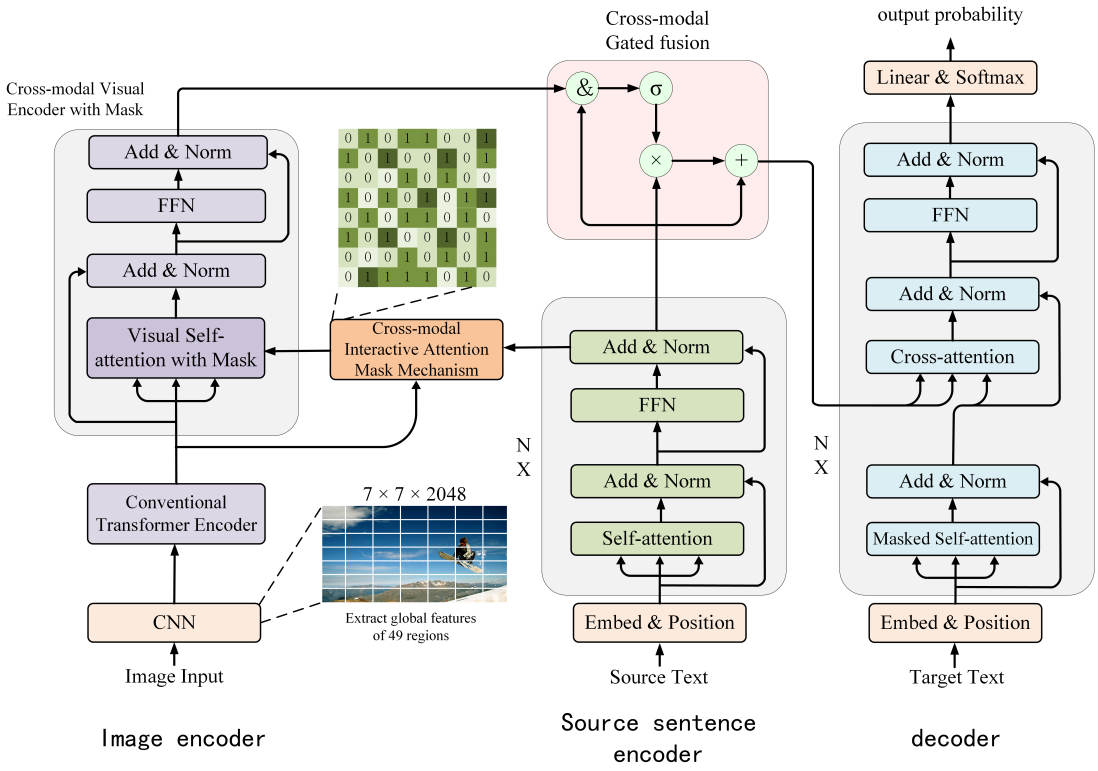

Figure 2: The overview of our proposed model, which consists of four components: (a) the image encoder to encode visual information with cross-modal interactive attention mask mechanism based on Transformer encoder; (b) the source sentence encoder to encode textual information; (c) the cross-modal gated fusion module to fuse helpful visual features and textual features; (d) the decoder to generate target translation conditioned on encoded textual features with helpful visual information;

modal mask-based pre-trained models (Li et al., 2020; Zhou et al., 2021; Shin et al., 2022), both text tokens and vision tokens are randomly masked, which aims to learn multimodal representations between vision and language in a pre-training manner. The mask strategy has been shown effective in many pre-training representation learning tasks. Inspired by Li et al. (2021c), in this work, we try to exploit the mask strategy to address the noise-robust multi-modal fusion problem for MNMT.

3 Methodology

In this section, we introduce our proposed noise-robust multi-modal neural machine translation approach, as illustrated in Figure 2. Our proposed model is based on the structure of Transformer, which contains four subnetworks, 1) source sentence encoder, 2) image encoder with robust masking matrix, 3) cross-modal gated fusion module and 4) target sentence decoder.

Without loss of generality, input words are embedded via traditional embedding layer with position embedding. As an example, denote by $x_j = \{x_1^j, \cdots, x_n^j\}$ and v_j as the j-th data-pair of source sentence input and its corresponding image, respectively, where n is the source length of x_j. Formally, the source sentence representation E_j^x and visual representation E_j^v are calculated as $\mathrm{E}_j^x = \mathrm{Emb}_x(x_j)$ and $\mathrm{E}_j^v = \mathrm{Emb}_v(v_j)$, where, Emb_x is the textual embedding layer with both word embedding and position embedding, Emb_v is the visual feature extraction layer with Resnet-101, $\mathrm{E}_j^x \in R^{n \times d_1}$ and $\mathrm{E}_j^v \in R^{m \times m \times d_2}$.

3.1 Source Sentence Encoder

As shown in the middle part of Figure 2, our encoder is employed the same as the conventional multi-head Transformer encoder, and each encoder layer is composed of two sublayers: 1) self-attention layer and 2) position-wise feedforward network (FFN) layer. Concretely, we first employ the multi-head self-attention module is used here by taking the sourece textual representation as a query/key/value matrix to establish word-to-word interconnections, which can be expressed as,

$$\mathrm{H}_{x_j}^l = \mathrm{Multihead}(\mathrm{E}_j^x, \mathrm{E}_j^x, \mathrm{E}_j^x) \quad (1)$$

$$= \mathrm{Concat}(head_j^1, \cdots, head_j^M) \quad (2)$$

where, M denotes the number of heads, Multihead$(\cdot)$ is a multi-head attention layer, $l = \{0, \cdots, 3\}$ is the Transformer layer index. Formally, the output of Multi-head attention is computed as follows:

$$head_j^{c\in[1,M]} = \sum_{k=1}^{n} \alpha_{ik}(\mathrm{E}_{j_k}^{x}\mathbf{W}_{j,c}^{V}) \quad (3)$$

where n is the source length of x_j, the weight coefficient of α_{ik} is calculated by the softmax function:

$$\alpha_{ik} = \mathrm{softmax}\left(\frac{(\mathrm{E}_{j_i}^{x}\mathbf{W}_{j,c}^{Q})(\mathrm{E}_{j_k}^{x}\mathbf{W}_{j,c}^{K})^{\mathrm{T}}}{\sqrt{d}}\right) \quad (4)$$

where α_{ik} is the dot-product attention matrix of the textual features and multi-modal features, $\mathbf{W}_{j,c}^{V}$, $\mathbf{W}_{j,c}^{Q}$, $\mathbf{W}_{j,c}^{K}$ are parameter matrices.

Then the position-wise Feed-Forward neural network is used to update the state of each position of the sequence for produce $\mathrm{F}_{x_j}^{l}$ as follows:

$$\mathrm{F}_{x_j}^{l} = \mathrm{FFN}(\mathrm{H}_{x_j}^{l}) \quad (5)$$

3.2 Image Encoder with Robust Masking Matrix

As shown in the left part of Figure 2, our image encoder layer is composed of two sublayers: 1) conventional Transformer encoder and 2) cross-modal visual encoder with mask. To reduce the number of parameters of the proposed model, we only use a single Transformer layer in image encoder.

3.2.1 Conventional Transformer Encoder for Visual

The image feature is extracted by the pretrained Resnet-101 models, and the image spatial feature is 7×7×2048-dimensional vector with 49 local spatial region features of each image. And we then transfer them into a 49× d feature matrix by linear transformation, where d denote the word-embedding-dimensional. Then, an internal relationship is established between the 49 image regions, concretely, we generate the contextual representations H_{v_j} of the 49 local spatial region features by a conventional Transformer-encoder, which can be expressed as,

$$\mathrm{H}_{v_j} = \mathrm{Multihead}(\mathrm{E}_j^{v}, \mathrm{E}_j^{v}, \mathrm{E}_j^{v}) \quad (6)$$

$$\mathrm{F}_{v_j} = \mathrm{FFN}(\mathrm{H}_{v_j}) \quad (7)$$

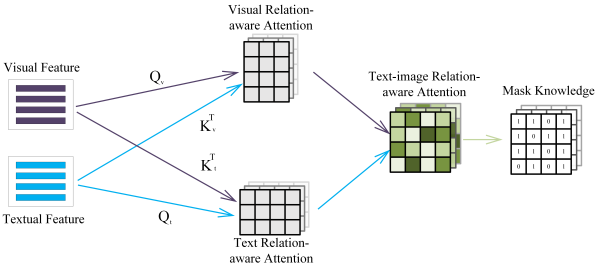

Figure 3: cross-modal interaction attention mask mechanism module.

3.2.2 Cross-modal mask mechanism for visual presentation

Inspired by Li et al. (2021c), in this section, we will introduce the proposed cross-modal visual encoder with mask module. Specifically, to mask irrelevant visual information before cross-modal fusion, we propose a cross-modal interaction attention mask mechanism, as shown in figure 3. First, cross-modal interaction of textual features and visual features is performed to compute the correlations between 49 regional features and textual features as follows:

$$\mathrm{Matrix}_{v_j} = \mathrm{softmax}\left(\frac{\mathrm{F}_{v_j} \times (\mathrm{F}_{x_j}^{l})^{\mathrm{T}}}{\sqrt{d}}\right) \quad (8)$$

$$\mathrm{Matrix}_{x_j} = \mathrm{softmax}\left(\frac{\mathrm{F}_{x_j}^{l} \times (\mathrm{F}_{v_j})^{\mathrm{T}}}{\sqrt{d}}\right) \quad (9)$$

where, $\mathrm{Matrix}_{v_j} \in R^{49\times n}$ denotes the attention of 49 local region features to each word of the paired source sentence, $\mathrm{Matrix}_{x_j} \in R^{n\times 49}$ denotes the attention of each word of the source sentence on the 49 local regions of the paired image.

Then we interactively compute Matrix_{v_j} and Matrix_{x_j} as follows,

$$\mathrm{Mask}_j = \mathrm{Matrix}_{v_j} \times \mathrm{Matrix}_{x_j} \quad (10)$$

where, Mask_j denotes the correlation matrix between the 49 local regions of the image and corresponding source sentences.

The mask matrix is generated according to the importance of the local region information of the image, which we set a threshold $prob_r$ to control the image region that needs to be masked. Thus we have that

$$m_r = \begin{cases} 1, prob_r \geq p, (r = \{1, 2, \cdots, 49\}) \\ 0, prob_r < p \end{cases} \quad (11)$$

where, p is a hyper-parameter, which is leveraged to mask unimportant visual region features, and it

Model	Multi30K En→De					
	Test2016		Test2017		MSCOCO	
	BLEU	METEOR	BLEU	METEOR	BLEU	METEOR
Existing MNMT Systems						
VMMT (Calixto et al., 2019)	37.7	56.0	30.1	49.9	25.5	44.8
VAG-NMT (Zhou et al., 2018)	-	-	31.6	52.2	28.3	48.0
Del+Obj (Ive et al., 2019)	38.0	55.6	-	-	-	-
DCCN (Lin et al., 2020)	39.7	56.8	31.0	49.9	26.7	45.7
MNMT+SVA (Nishihara et al., 2020)	39.9	58.1	-	-	-	-
OVC+L_v (Wang and Xiong, 2021)	-	-	32.4	52.3	28.6	48.0
WRA-guided (Zhao et al., 2021)	39.3	58.3	32.3	52.8	28.5	48.5
Our Transformer-Based Systems						
Transformer (NMT) (Vaswani et al., 2017)	40.96	58.35	32.59	51.21	29.16	48.37
Doubly-ATT (Arslan et al., 2018) †	41.44	59.08	33.15	52.34	29.22	48.41
Multimodal self-att (Yao and Wan, 2020) †	41.50	58.52	32.51	51.33	29.10	48.48
Gated Fusion MNMT (Yin et al., 2020) †	41.58	58.88	33.01	51.90	30.04	48.95
Our model	**42.56**	**59.98**	**35.09**	**54.51**	**31.09**	**50.46**

Table 1: Comparison results on Multi30k En→De task on BLEU and METEOR metrics. † means to reproduce previous multi-modal fusion method based on our Transformer systems. Best results are highlighted in bold.

is a scalar. Our strategy ensures that each image always presents the most relevant visual region to the corresponding source textual. Then convert the image area of $m_r = 0$ to false, and $m_r = 1$ to true, which we construct a mask knowledge matrix.

Finally, we employ cross-modal visual encoder with mask to obtain effective visual information, thus we have

$$\widehat{\mathrm{H}}_{v_j} = \text{Multihead-mask}(\mathrm{F}_{v_j}, \mathrm{F}_{v_j}, \mathrm{F}_{v_j}) \quad (12)$$

$$\widehat{\mathrm{F}}_{v_j} = \text{FFN}(\widehat{\mathrm{H}}_{v_j}) \quad (13)$$

where Multihead-mask$(*)$ denote the self-attention with mask knowledge, the purpose of Multihead-mask is to mask weakly correlated visual information.

3.3 Cross-modal Gated Fusion Module

In this section, we employ cross-modal gated fusion method to fuse textual features and extracted helpful visual features, which is a popular multi-modal fusion method for many recent MNMT, as shown in Figure 2. Formally, we have that

$$\Omega = \text{Sigmoid}(\mathrm{W}_\Omega \widehat{\mathrm{F}}_{v_j} + \mathrm{U}_\Omega \mathrm{F}_{x_j}) \quad (14)$$

$$\mathrm{H}_{g_j} = \mathrm{F}_{x_j} + \Omega \widehat{\mathrm{F}}_{v_j} \quad (15)$$

where, W_Ω and U_Ω are trainable model parameters. The final output H_{g_j} is directly fed into our target sentence decoder (See Figure 2 right) to predict the translation.

4 Experiments

Datasets: We conduct experiments on En→De, En→Fr and En→Cs tasks of the widely used Multi30K [2] benchmark dataset, in which the training and validation sets contains 29k and 1014 text-image pairs, respectively. Furthermore, we adopt four test sets to evaluate our MNMT model, 1) the Test2016 test set with 1,000 examples contained in Multi30K; 2) the Test2017 test set with 1,000 examples in WMT2017, which contains more difficult source sentences to translate and understand; 3) we also use ambiguous COCO dataset as out-domain test data, which contains 461 examples with ambiguous verbs and encourages to use image for disambiguation; and 4) the Test2018 test set contains 1071 instances with more entity words and more low frequency words.

Data Pre-processing: We directly use the pre-processed sentence pairs via byte pair encoding (BPE) segmentation with 6k bpe vocabulary, the resulting vocabulary sizes of each language pair were 5,644→5,876 tokens for En→De, 5,644→5,684 tokens for En→Fr, 5,644→5,972 tokens for En→Cs. For each image, which is extracted through the pre-trained Resnet-101 model, the spatial features are 7x7x2048-dimensional vectors with 49 local spatial region features.

Metrics: We evaluate the quality of translations with two metrics, 1) 4-gram BLEU metrics (Pap-

[2] https://github.com/multi30k/dataset

Model	Multi30K En→Fr					
	Test2016		Test2017		MSCOCO	
	BLEU	METEOR	BLEU	METEOR	BLEU	METEOR
Existing MNMT Systems						
VAG-NMT (Zhou et al., 2018)	-	-	53.8	70.3	45.0	64.7
Del+Obj (Ive et al., 2019)	59.8	74.4	-	-	-	-
DCCN (Lin et al., 2020)	61.2	76.4	54.3	70.3	45.4	65.0
OVC+L_v (Wang and Xiong, 2021)	-	-	54.2	70.5	45.2	64.6
WRA-guided (Zhao et al., 2021)	61.8	76.3	54.1	70.6	43.4	63.8
Our Transformer-Based Systems						
Transformer (NMT) (Vaswani et al., 2017)	60.33	75.64	53.45	71.57	43.61	65.72
Doubly-ATT (Arslan et al., 2018) †	60.94	75.99	53.63	71.56	44.78	65.35
Multimodal self-att (Yao and Wan, 2020) †	61.44	75.77	54.56	71.62	44.59	65.08
Gated Fusion MNMT (Yin et al., 2020) †	61.24	76.26	54.15	71.77	44.29	64.91
Our model	**63.24**	**77.54**	**55.48**	**72.62**	**46.34**	**67.40**

Table 2: Comparison results on the En→Fr translation task on the Multi30k dataset.

ineni et al., 2002), which measures the quality of translations in terms of accuracy and fluency. 2) METEOR metrics (Denkowski and Lavie, 2014), which takes into account both precision and recall for translation quality.

4.1 Settings

We conduct our proposed models based on Transformer framework, with only stack 4-layer encoder-decoder, so the amount of parameters required by our model is small. Concretely, we set the dimensions of the encoder and decoder hidden states at d_{model}=128, the inner-layer of feed-forward network is set as d_{ffn}=256. The learning rate is set to 0.005. The max tokens is set to 4096, the learning rate is varied under a warmup-updates with 2,000 steps, and the label smoothing with value set as 0.1. We use adam optimizer with β_1, β_2 = (0.9, 0.98). We adopt 4 heads here and the dropout is set to 0.3 to avoid the over-fitting. The width of beam size is set to 5. We train our models on a single GTX 3090 GPU with fp16.

4.2 Baseline Models

To empirically verify the advantages of our proposed MNMT model, we show the performance of the following recent state-of-the-art MNMT models for comparison on the En→De and En→Fr translation task, namely: VMMT (Calixto et al., 2019), VAG-NMT (Zhou et al., 2018), Del+Obj (Ive et al., 2019), DCCN (Lin et al., 2020), MNMT+SVA (Nishihara et al., 2020), OVC+L_v (Wang and Xiong, 2021), WRA-guided (Zhao et al., 2021). Furthermore, to more fairly demonstrate the superiority and validity of our proposed model, we reproduce the recent state-of-the-art methods for comparison based on the same parameter settings and training equipment, we experiment with the following: 1) Gated Fusion MNMT (Yin et al., 2020): An efficient multi-modal fusion method to enhance machine translation. 2) Multimodal self-att (Yao and Wan, 2020): A image-aware multi-modal transformer model is proposed to extract image information to improve machine translation performance. 3) Doubly-ATT (Arslan et al., 2018): At the decoder, the visually evoked attention weights and the source language attention weights are added up as doubly-attention weights.

4.3 Results on the En→De Translation Task

As shown in Table 1, we present experimental results of our proposed model and other SOTA models on the En→De translation task. We summarize and compare the existing models in the three aspects as follows:

1) *Compare with Existing MNMT Systems:* Experimental results show that our proposed model outperforms existing SOTA models, and enhances BLEU and METEOR metrics by 3∼4 points on most of the test sets. The underlying reason is that our proposed method can effectively filter vision noise contained in the image.

2) *Compare with Text-to-text NMT:* Our MNMT model outperforms NMT baselines significantly on BLEU and METEOR metrics, which enhances about 2 points on all test sets. This indicates that our proposed MNMT model can utilize image information to improve machine translation.

3) *Compare with Reproduce Methods:* As we can observe that our proposed methods achieves

p	Test2016		Test2017		MSCOCO	
	BLEU	METEOR	BLEU	METEOR	BLEU	METEOR
Multi30K **En→De**						
$p = 0$	41.40	59.19	34.37	53.77	30.01	50.17
$p = 0.01$	41.45	59.02	34.73	54.00	30.77	50.12
$p = 0.015$	41.68	59.40	34.69	54.13	30.96	50.39
$p = 0.02$	42.58	59.98	35.09	54.51	31.09	50.46
$p = 0.025$	41.71	59.62	33.68	53.39	30.82	50.27
$p = 0.03$	40.99	58.75	33.44	53.47	30.36	49.66
Multi30K **En→Fr**						
$p = 0$	61.16	76.33	54.49	72.51	44.91	65.81
$p = 0.01$	62.29	76.77	55.31	72.34	44.87	65.46
$p = 0.015$	62.67	76.96	55.36	73.00	45.48	66.76
$p = 0.02$	63.24	77.54	55.48	72.62	45.82	67.17
$p = 0.025$	62.57	76.84	55.39	72.46	46.34	67.40
$p = 0.03$	62.14	76.99	54.89	72.63	45.62	66.31

Table 3: Ablation study on hyper-parameter p on the En→Fr and En→De tasks.

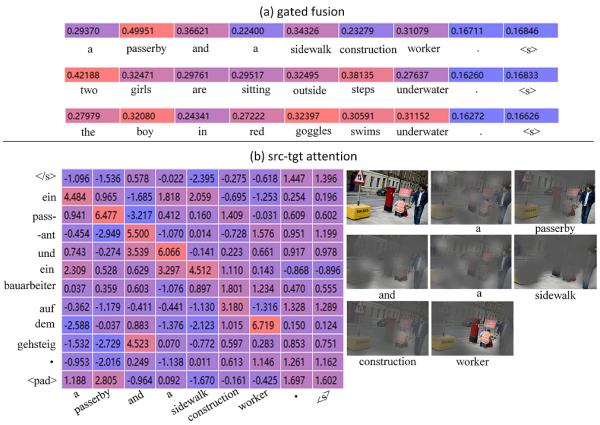

Figure 4: Examples of attention maps on the En→De task. (a) Gated fusion (Ω visualize): attention weights for visual information and source sentences. (b) src-tgt attention: source and target sentence attention weights with visual guidance.

a significant improvement over the SOTA method on all the evaluation metrics, which demonstrates that masking irrelevant visual information helps improve translation performance.

4.4 Results on the En→Fr Translation Task

To explore the robustness of the proposed model, we also guide experiments on the Multi30K En→Fr translation task, the results are illustrated in Table 2. Concretely, we draw the following interesting conclsions:

First, comparing with existing models, our proposed model still achieves significant improvement on two evaluation metrics, which is consistently with the result of the En→De task. In addition, comparing with text-only NMT baseline models, MNMT with image information achieves superior results, which demonstrates that our MNMT model can effectively and efficiently interact with visual information to enhance machine translation.

Second, reproducing recent competitive methods based on the same NMT strong baseline model on En→Fr task, results are shown in Table 2. It is obviously that our method outperforms the SOTA methods and achieves strong competitive results among all the existing MNMT models. The results on the En→Fr translation task once again demonstrate the effectiveness and generalizability of the proposed method.

4.5 Ablation Study

To further determine the effectiveness of our proposed method, we show the following sets of ablation experiments on both the En→Fr and En→De tasks, 1) Ablation study on hyper-parameter p; 2) Ablation study on different components of model.

Ablation study on hyper-parameter p As shown in Table 3, we report the effect of hyper-parameter p on model translation performance, where p represents the threshold that controls the effective visual similarity weights. We summarize several interesting conclusions:

First, in general, it can be observed that at $p = 0.02$, the experimental results of the proposed model on most test sets achieve the best results on the En→De and En→Fr tasks. Furthermore, gradually increase or decrease the threshold p, it is obvious that the experimental results also gradually decrease on the BLEU and METEOR metrics. We consider that there are two main reasons. On the one hand, with the decrease of the threshold p, the masked noise information decreases, and the captured visual information contains more noise, and the introduction of noise leads to a decrease in the performance of the model. On the other hand, with the increase of the threshold p, more visual information is masked, and even a lot of helpful visual information is masked, which causes the performance of the model to decline.

Secend, in more detail, when the threshold $p = 0$, which means that the model fuses all visual information, compared with NMT model, our model achieves better translation performance, but there are no significant BLEU and METEOR gains. The prove the assumption that masking visual noise information helps improve machine translation.

Ablation study on different components of model To investigate the effectiveness of different components in our proposed MNMT model, we

	Test2016		Test2017		MSCOCO	
	BLEU	METEOR	BLEU	METEOR	BLEU	METEOR
Multi30K **En→De**						
Complete model	42.58	59.98	35.09	54.51	31.09	50.46
MNMT_rg	41.85	59.76	34.24	54.16	29.91	50.11
MNMT_re	41.96	59.59	34.37	53.94	30.68	50.21
MNMT_rvm	41.53	59.01	34.56	53.89	30.33	50.00
Multi30K **En→Fr**						
Complete model	63.24	77.54	55.48	73.00	46.34	67.40
MNMT_rg	62.49	76.78	55.34	72.34	46.06	66.68
MNMT_re	62.85	77.37	55.41	72.84	45.73	67.30
MNMT_rvm	62.22	76.94	55.06	72.14	45.57	66.67

Table 4: Ablation study on different components of model on the En→Fr and En→De tasks. MNMT_rvm means to remove cross-modal visual encoder with mask, MNMT_re means to remove conventional Transformer encoder, MNMT_rg means to remove cross-modal gated fusion module.

En→Cs				
	Test2016		Test2018	
Model	BLEU	METEOR	BLEU	METEOR
Our model	**35.09**	**33.52**	**31.40**	**31.26**
Transformer (NMT)	32.70	32.34	27.62	29.03
Doubly-ATT (Arslan et al., 2018) †	33.25	32.28	29.12	29.87
Multimodal self-att (Yao and Wan, 2020) †	33.12	32.01	28.75	29.51
Gated Fusion MNMT (Yin et al., 2020) †	33.77	32.24	29.43	29.41

Table 5: Experiment results on En→Cs task.

SRC: a seagull sitting atop a light fixture .
REF: eine möve sitzt auf einer lampe .
MNMT: eine möwe sitzt auf einer lampe .
MNMT_rm: eine möwe sitzt auf einer hellen mischung .
NMT: eine möwe sitzt auf einer hellen apparatur .

SRC: a bee is landing on a pink rose .
REF: eine biene landet auf einer rosafarbenen rose .
MNMT: eine biene landet auf einer rosafarbenen rosen .
MNMT_rm: eine biene landet auf einem rosa rosenden .
NMT: eine biene landet auf einer pinkfarbenen frau .

Figure 5: Examples of successful translation with remove of visual noise. Improved translations are highlighted in color.

further conduct experiments to compare with the following variants models in Table 4:

1) *Effectiveness of cross-modal gated fusion.* The result in row 2 indicates that removing the gated fusion leads to a significant performance decline on BLEU and METEOR metrics. It suggests that gated fusion is an efficient method for fusing multimodal features, which is helpful in order to enhance translation performance.

2) *Effectiveness of conventional Transformer encoder.* To verify the usefulness of establishing the intra-modal correspondences before interacting with multimodal features, we remove the conventional Transformer encoder component. The result in row 3 shows that this change causes a light performance drop. The underlying reason is the lack of visual contextual semantic information in visual information without intra-modal correspondences.

3) *Effectiveness of cross-modal visual encoder with mask.* To construct this variant, we directly remove the cross-modal visual encoder and then employ gated fusion to incorporate full visual features and textual features. Apparently, the performance drop reported in line 4 demonstrates the validity of our proposed cross-modal visual encoder with mask module. Furthermore, it also validates our hypothesis that masking irrelevant visual information before fusing multimodal features is favourable to improve translation performance.

4.6 Visual analysis

As shown in Figure 4, to further understand and verify our model, we visualize the gated fusion and src-tgt attention weights. 1) Gated fusion: the results show that our model can effectively focus on the consistent visual regions corresponding to the source text. 2) Src-tgt attention: useful visual information as a bridge can effectively align source and target sentences to help translation.

4.7 Results on the En→Cs Translation Task

We further verify the effectiveness and generalization of the proposed method on the En→Cs task, the results shown in Table 5. Our model still achieves excellent performance compared with all baselines, which again proves that our model is effective and general for different language pairs.

4.8 Case Study

As shown in Figure 5, we further confirm the effectiveness of our proposed method. It can be observed that the two words *'light fixture'* and *'pink rose'* can be correctly translated by the MNMT model, while the MNMT_rvm model is not fully translated, and the NMT model is translated incorrectly. The underlying reason is that the complete image information introduces noise into MNMT model and distracts the model. This reveals that the proposed encoder is able to learn more efficient representations.

5 Conclusion

In this paper, we propose a noise-robust multimodal interactive fusion approach with cross-modal relation-aware mask mechanism to address image noise in MNMT. Experiment results and analysis on three benchmark translation tasks demonstrate the effectiveness and superiority of our proposed method. Further ablation experiments demonstrate that masking irrelevant visual information helps machine translation. In future work, we will continue to explore how to more effectively remove noisy information in vision.

Acknowledgments

This work was supported by National Key Research and Development Program of China (No. 2020AAA0107904). National Natural Science Foundation of China (Nos. 61866020, 61732005), Key Research and Development Program of Yunnan Province (Nos. 202203AA080004, 202103AA080015). Natural Science Foundation Project of Yunnan Science and Technology Department (No. 2019FB082)

References

Hasan Sait Arslan, Mark Fishel, and Gholamreza Anbarjafari. 2018. Doubly attentive transformer machine translation. *arXiv:1807.11605*.

Loïc Barrault, Fethi Bougares, Lucia Specia, Chiraag Lala, Desmond Elliott, and Stella Frank. 2018. Findings of the third shared task on multimodal machine translation. In *THIRD CONFERENCE ON MACHINE TRANSLATION (WMT18)*, volume 2, pages 308–327.

Ozan Caglayan, Pranava Madhyastha, Lucia Specia, and Loïc Barrault. 2019. Probing the need for visual context in multimodal machine translation. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4159–4170, Minneapolis, Minnesota. Association for Computational Linguistics.

Iacer Calixto, Qun Liu, and Nick Campbell. 2017. Incorporating global visual features into attention-based neural machine translation. *In: Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing, EMNLP 2017, Copenhagen, Denmark, September 9–11, 2017, pp. 992–1003. doi:10.18653/v1/d17-1105*.

Iacer Calixto, Miguel Rios, and Wilker Aziz. 2019. Latent variable model for multi-modal translation. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 6392–6405, Florence, Italy. Association for Computational Linguistics.

Jean-Benoit Delbrouck and Stéphane Dupont. 2017. An empirical study on the effectiveness of images in multimodal neural machine translation. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 910–919, Copenhagen, Denmark. Association for Computational Linguistics.

Michael Denkowski and Alon Lavie. 2014. Meteor universal: Language specific translation evaluation for any target language. In *Proceedings of the ninth workshop on statistical machine translation*, pages 376–380.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. Bert: Pre-training of deep bidirectional transformers for language understanding. *In Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers), pages 4171–4186, Minneapolis, Minnesota. Association for Computational Linguistics*.

Desmond Elliott, Stella Frank, Loïc Barrault, Fethi Bougares, and Lucia Specia. 2017. Findings of the second shared task on multimodal machine translation and multilingual image description. *In: Proceedings of the Conference on Machine Translation (WMT), Volume 2: Shared Task Papers, pages 215–233*.

Zhiyi Fu, Wangchunshu Zhou, Jingjing Xu, Hao Zhou, and Lei Li. 2022. Contextual representation learning beyond masked language modeling. *arXiv preprint arXiv:2204.04163*.

Baban Gain, Dibyanayan Bandyopadhyay, and Asif Ekbal. 2021. Experiences of adapting multimodal machine translation techniques for hindi. In *Proceedings of the First Workshop on Multimodal Machine Translation for Low Resource Languages (MMTLRL 2021)*, pages 40–44.

Po-Yao Huang, Frederick Liu, Sz-Rung Shiang, Jean Oh, and Chris Dyer. 2016. Attention-based multimodal neural machine translation. In *Proceedings of the First Conference on Machine Translation: Volume 2, Shared Task Papers*, pages 639–645.

Julia Ive, Pranava Madhyastha, and Lucia Specia. 2019. Distilling translations with visual awareness. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 6525–6538, Florence, Italy. Association for Computational Linguistics.

Mandar Joshi, Danqi Chen, Yinhan Liu, Daniel S Weld, Luke Zettlemoyer, and Omer Levy. 2020. Spanbert: Improving pre-training by representing and predicting spans. *Transactions of the Association for Computational Linguistics*, 8:64–77.

Soonmo Kwon, Byung-Hyun Go, and Jong-Hyeok Lee. 2020. A text-based visual context modulation neural model for multimodal machine translation. *Pattern Recognition Letters*, 136:212–218.

Bei Li, Chuanhao Lv, Zefan Zhou, Tao Zhou, Tong Xiao, Anxiang Ma, and JingBo Zhu. 2022. On vision features in multimodal machine translation. *arXiv preprint arXiv:2203.09173*.

Gen Li, Nan Duan, Yuejian Fang, Ming Gong, and Daxin Jiang. 2020. Unicoder-vl: A universal encoder for vision and language by cross-modal pre-training. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pages 11336–11344.

Jiaoda Li, Duygu Ataman, and Rico Sennrich. 2021a. Vision matters when it should: Sanity checking multimodal machine translation models. *In: Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing, pages 8556–8562*.

Jiaoda Li, Duygu Ataman, and Rico Sennrich. 2021b. Vision matters when it should: Sanity checking multimodal machine translation models. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 8556–8562, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Zhaowen Li, Zhiyang Chen, Fan Yang, Wei Li, Yousong Zhu, Chaoyang Zhao, Rui Deng, Liwei Wu, Rui Zhao, Ming Tang, et al. 2021c. Mst: Masked self-supervised transformer for visual representation. *Advances in Neural Information Processing Systems*, 34.

Huan Lin, Fandong Meng, Jinsong Su, Yongjing Yin, Zhengyuan Yang, Yubin Ge, Jie Zhou, and Jiebo Luo. 2020. Dynamic context-guided capsule network for multimodal machine translation. In *Proceedings of the 28th ACM International Conference on Multimedia*, pages 1320–1329.

Tetsuro Nishihara, Akihiro Tamura, Takashi Ninomiya, Yutaro Omote, and Hideki Nakayama. 2020. Supervised visual attention for multimodal neural machine translation. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 4304–4314.

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. Bleu: a method for automatic evaluation of machine translation. In *Proceedings of the 40th annual meeting of the Association for Computational Linguistics*, pages 311–318.

Puyuan Peng and David Harwath. 2022. Self-supervised representation learning for speech using visual grounding and masked language modeling. *arXiv preprint arXiv:2202.03543*.

Andrew Shin, Masato Ishii, and Takuya Narihira. 2022. Perspectives and prospects on transformer architecture for cross-modal tasks with language and vision. *International Journal of Computer Vision*, pages 1–20.

Yuqing Song, Shizhe Chen, Qin Jin, Wei Luo, Jun Xie, and Fei Huang. 2021. Enhancing neural machine translation with dual-side multimodal awareness. *IEEE Transactions on Multimedia*.

Hiroki Takushima, Akihiro Tamura, Takashi Ninomiya, and Hideki Nakayama. 2019. Multimodal neural machine translation using cnn and transformer encoder. In *Proceedings of the 20th International Conference on Computational Linguistics and Intelligent Text Processing (CICLING 2019)*.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in neural information processing systems*, pages 5998–6008.

Dexin Wang and Deyi Xiong. 2021. Efficient object-level visual context modeling for multimodal machine translation: Masking irrelevant objects helps grounding. In *Thirty-Fifth AAAI Conference on Artificial Intelligence, AAAI*, pages 2–9.

Zhenda Xie, Zheng Zhang, Yue Cao, Yutong Lin, Jianmin Bao, Zhuliang Yao, Qi Dai, and Han Hu. 2021. Simmim: A simple framework for masked image modeling. *arXiv preprint arXiv:2111.09886*.

Shaowei Yao and Xiaojun Wan. 2020. Multimodal transformer for multimodal machine translation. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 4346–4350.

Junjie Ye and Junjun Guo. 2022. Dual-level interactive multimodal-mixup encoder for multi-modal neural machine translation. *Applied Intelligence*, pages 1–10.

Yongjing Yin, Fandong Meng, Jinsong Su, Chulun Zhou, Zhengyuan Yang, Jie Zhou, and Jiebo Luo. 2020. A novel graph-based multi-modal fusion encoder for neural machine translation. *In: Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, pages 3025–3035*.

Yuting Zhao, Mamoru Komachi, Tomoyuki Kajiwara, and Chenhui Chu. 2021. Word-region alignment-guided multimodal neural machine translation. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*.

Yuting Zhao, Mamoru Komachi, Tomoyuki Kajiwara, and Chenhui Chu. 2022. Region-attentive multimodal neural machine translation. *Neurocomputing*.

Mingyang Zhou, Runxiang Cheng, Yong Jae Lee, and Zhou Yu. 2018. A visual attention grounding neural model for multimodal machine translation. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 3643–3653, Brussels, Belgium. Association for Computational Linguistics.

Mingyang Zhou, Luowei Zhou, Shuohang Wang, Yu Cheng, Linjie Li, Zhou Yu, and Jingjing Liu. 2021. Uc2: Universal cross-lingual cross-modal vision-and-language pre-training. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 4155–4165.

Speeding Up Transformer Decoding via an Attention Refinement Network

Kaixin Wu[1], **Yue Zhang**[2,3], **Bojie Hu**[1], **Tong Zhang**[1]
[1]Tencent Minority-Mandarin Translation, Beijing, China
[2]School of Engineering, Westlake University
[3]Institute of Advanced Technology, Westlake Institute for Advanced Study
{danielkxwu, bojiehu, zatozhang}@tencent.com
zhangyue@westlake.edu.cn

Abstract

Despite the revolutionary advances made by Transformer in Neural Machine Translation (NMT), inference efficiency remains an obstacle due to the heavy use of attention operations in auto-regressive decoding. We thereby propose a lightweight attention structure called ***A**ttention **R**efinement **N**etwork* (ARN) for speeding up Transformer. Specifically, we design a weighted residual network, which reconstructs the attention by reusing the features across layers. To further improve the Transformer efficiency, we merge the self-attention and cross-attention components for parallel computing. Extensive experiments on ten WMT machine translation tasks show that the proposed model yields an average of 1.35× faster (with almost no decrease in BLEU) over the state-of-the-art inference implementation. Results on widely used WMT14 En→De machine translation tasks demonstrate that our model achieves a higher speed-up, giving highly competitive performance compared to AAN and SAN models with fewer parameter numbers[1].

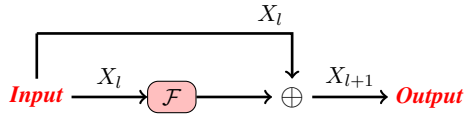

Figure 1: The information transformation between the input and output after through a sub-layer $\mathcal{F}$, where $X_{l+1} = X_l + \mathcal{F}(X_l)$

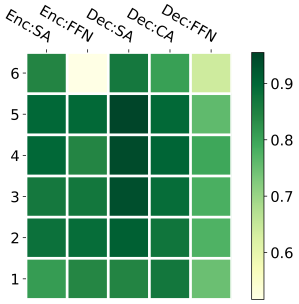

Figure 2: The cosine similarity of the input and output for every sub-layer over validation set on WMT14 En→De translation task. "Enc", "Dec", "SA", "CA", "FFN" represent Encoder, Decoder, Self-attention, Cross-attention and Feed-forward network, respectively. Darker cells denote more similar.

1 Introduction

Transformer (Vaswani et al., 2017) has become the dominant approach in the NMT literature, which achieves superior translation performance and efficiency due to its well-designed attention mechanism. The highly parallelizable architecture enables Transformer to capture the dependency among positions over the entire sequence parallelly for a faster training step. However, the inference efficiency remains a bottleneck for Transformer. In inference, Transformer follows an auto-regressive generation paradigm and generates the target words one by one on the decoder side. The heavy use of dot-product attention operations even further slows Transformer efficiency. In addition, there are a large number of attention sub-layers in the multi-layer stacked Transformer, which makes it prone to redundancy. As shown in Figure 2, we observe redundant computation in Transformer attention sub-layer, especially the attention in the decoder. This inspires us to explore a lightweight attention structure to speed up Transformer decoding.

Many efforts have been dedicated to accelerating the decoding process of Transformer. AAN (Zhang et al., 2018) adopts an average strategy to avoid computing the correlations over the entire input word. However, this method requires a complicated network and only focuses on the decoder-side self-attention. SAN (Xiao et al., 2019) reuses the attention results among layers, but it requires a model to learn which layers should be allowed to share. Besides, another representative

[1]https://github.com/Kaixin-Wu-for-Open-Source/ARN

Proceedings of the 29th International Conference on Computational Linguistics, pages 5109–5118
October 12–17, 2022.

approach (Gu et al., 2017; Guo et al., 2019; Wang et al., 2019) follows another line to abandon the auto-regressive generation property and produces target sequences in parallel, thus it fails to model the word dependencies.

We investigate an alternative lightweight attention structure for Transformer acceleration. As shown in Figure 3, the main idea is to reconstruct the attention via a weighted combination of high-level and low-level features, rather than the standard dot-product function. The two parts represent the attention results of the current layer and its previous layer, respectively. This weight, a learned matrix under the guidance of the two input features, which determines how many low-level features can be selected to fuse. Consequently, we refer to our model as ***A**ttention **R**efinement **N**etwork*, or ARN in short. Our combination process is computationally inexpensive, which allows us to employ less computation to obtain the attention results of adjacent layers and combine them to approximate the original attention results. ARN structure can be viewed as a weighted residual network, which acquires the attention results by reusing the features across layers. This way introduces low-level features into the attention sub-layer to further enhance model confidence, which is an improvement over SAN (Xiao et al., 2019). In addition, this method can be applied to both self-attention and cross-attention on the decoder side and we merge the above two components to further improve decoding efficiency. Moreover, ARN structure is simple and it is easy to implement. As another "bonus", ARN requires fewer parameters, so it is faster to train and maintains a smaller memory footprint.

Extensive experiments on ten WMT translation tasks show that ARN achieves an average of $1.35\times$ faster with performance on par with a strong baseline. Compared to AAN and SAN baselines, our model gives a higher speed-up as well as highly competitive performance with fewer parameter numbers on widely used WMT14 En→De translation tasks.

2 Standard Transformer Attention

Standard Transformer follows the popular encoder-decoder paradigm, which consists of a 6-layer encoder and a 6-layer decoder. The overall architecture only contains stacked attention and feed-forward networks (FFN) in Transformer. There are three types of attention mechanisms: the encoder-side self-attention, the decoder-side self-attention and the cross-attention. On the encoder side, each layer follows the order of operations that could be defined as: *self-attention* → FFN. Similarly, the decoder side follows the way: *self-attention* → *cross-attention* → FFN.

The attention model in Transformer is scaled dot-product attention. See Figure 3 (a) for an illustration of the standard attention. The input of attention is a tuple of (Q_l, K_l, V_l), where $Q_l \in \mathbb{R}^{m\times d}$ and $K_l, V_l \in \mathbb{R}^{n\times d}$ are the matrices of corresponding queries, keys and values of the l-th layer. For encoder or decoder self-attention, $m = n$ represents the source or target sequence length. For cross-attention, m and n are the target sequence length and source sequence length, respectively. d is the dimension of the hidden representation. We first compute the attention distribution via a scaled dot-product and softmax operations.

$$\begin{aligned} A_l &= \text{Sim}(Q_l, K_l) \\ &= \text{Softmax}(\frac{Q_l \cdot {K_l}^T}{\sqrt{d}}) \end{aligned} \tag{1}$$

where A_l is an $m \times n$ matrix, which represents the degree of relevance between different positions of queries and values. The ouput of attention is a weighted sum of values, and it can be defined as:

$$F_l = A_l \cdot V_l \tag{2}$$

where Q_l, K_l, V_l are all generated by a linear transformation. In self-attention (encoder or decoder), the three parts share the same source, which comes from the output of its previous layer. While in cross-attention, the difference is that the K_l and V_l from the output of encoder side. F_l is the attention results of the l-th attention sub-layer, which is then fed into the next sub-layer.

Note that the matrix multiplications in Eq. 1 and Eq. 2 are computationally expensive. This is even worse for inference because the operation is repeated until an end symbol is reached due to the auto-regressive generation property. We also compute the cosine similarity of the input and output for every sub-layer in Transformer. Figure 1 is the definition of a sub-layer $\mathcal{F}$ and Figure 2 shows the similarity results. As can be seen that the input and output of the attention sub-layer are very similar, especially the attention in the decoder. Specifically, we observe that the decoder-side self-attention presents the highest similarity compared to other sub-layers, followed by the cross-attention.

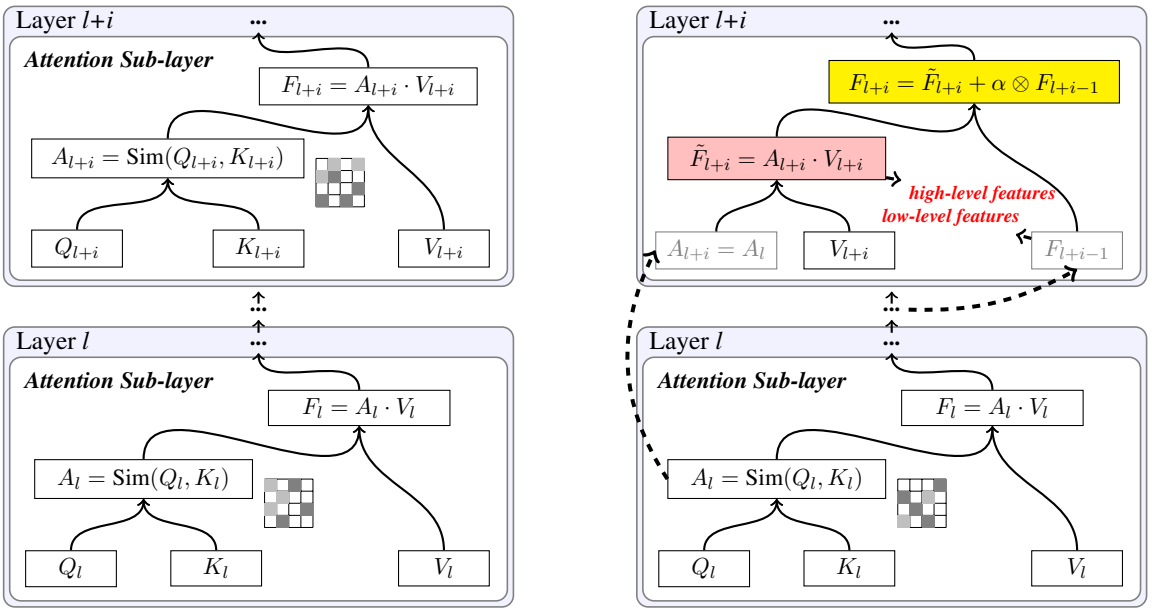

(a) Standard Transformer Attention

(b) Attention Refinement Network (ARN)

Figure 3: Comparison of the standard attention model and ARN model

Most of their similarities are more than 90%, and some even more than 95%. All these show the possibility of removing redundant computation in Transformer and lead us to learn a lightweight attention structure.

3 Attention Refinement Network (ARN)

The proposed method for speeding up Transformer concentrates on the decoder-side attention because the decoder is the bottleneck of Transformer inference.

Overall Architecture. The proposed ARN module is shown in Figure 3 (b). We assume that the decoder contains $L = M \times N$ layers. The decoder is simply divided into M parts equally, and each part contains N layers. For each ARN module, the bottom layer is the dot-product attention same as used in standard Transformer, and the next $N - 1$ layers are the lightweight attention composed of the two inputs. One is an approximate attention result of the current layer by reusing the attention weights within adjacent layers, and the other is the up-sampled attention results from its previous layer. Since the upper layer contains more semantic information, we refer to the two inputs high-level features and low-level features, respectively. α is a learned weight matrix, which is used to represent the fusion degree of low-level features. In contrast to Figure 3 (a), ARN reconstructs the attention via a feature fusion of the two input sources to replace the original dot-product attention. ARN can be regarded as a weighted residual network, which sums to obtain the attention results by reusing the features across layers. The ARN module is applied to both self-attention and cross-attention for speeding up Transformer.

3.1 The Model

Weighted Residual Network. For each ARN module, the attention is reconstructed via a weighted sum of the two input sources.

$$F_{l+i} = \tilde{F}_{l+i} + \alpha \otimes F_{l+i-1} \quad \text{for} \quad i \in [1, N-1] \tag{3}$$

where $\tilde{F}_{l+i}$ and F_{l+i-1} are the high-level features and low-level features, respectively. F_{l+i-1} is the attention results of the previous layer, and we reuse it as our low-level features of the current layer. $\otimes$ denotes the element-wise multiplication. $\tilde{F}_{l+i}$ is defined as:

$$\begin{aligned} \tilde{F}_{l+i} &= A_{l+i} \cdot V_{l+i} \\ &= A_l \cdot V_{l+i} \end{aligned} \tag{4}$$

where A_l denotes the attention weights of the l-th layer, its calculation process is shown in Eq. 1. Previous work show that the attention weights are redundant and the adjacent layers share similar distributions (Michel et al., 2019; Behnke and

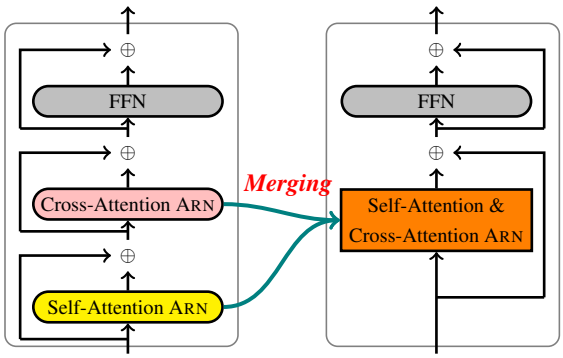

Figure 4: Merging self-attention and cross-attention in ARN decoder layer.

Model	Complexity per Step
Decoder self-attention	$O(n \cdot d^2 + n^2 \cdot d)$
Decoder self-attention ARN	$O(n^2 \cdot d)$
Cross-attention	$O(m \cdot d^2 + m^2 \cdot d)$
Cross-attention ARN	$O(m^2 \cdot d)$

Table 1: Computation complexity of different attention structures in a decoding step. m and n are the source sentence length and target sentence length, d is the dimension of the hidden representation.

Heafield, 2020; Xiao et al., 2019). Thus, we reuse the attention weights of adjacent lower layers to construct the high-level features. α is a learned matrix, which is obtained from the following process:

$$\begin{aligned}\alpha &= G(F_{l+i-1}, \tilde{F}_{l+i}) \\ &= \text{ReLU}(\frac{\mathbf{W}^{\alpha}_{l+i} \cdot \text{Max}(F_{l+i-1}, \tilde{F}_{l+i})}{\sqrt{d}})\end{aligned} \quad (5)$$

Here, $\text{Max}(\cdot)$ is an element-wise operation and $\mathbf{W}^{\alpha}_{l+i}$ is a learnable parameter. According to Eq. 5, α is a sparse weight matrix obtained from the supervision of high-level and low-level features. This weight matrix highlights important semantic information of low-level features, and it determines how many low-level features can be selected for fusion. In other words, the upper layer features can be refined through the fusion of the lower layer important features iteratively.

Merging Self-attention and Cross-attention. For the self-attention and cross-attention in a decoder layer, the formula can be simply defined as:

$$F_l = Q_l + \text{SelfAttn}(Q_l) + \text{CrossAttn}(\tilde{Q}_l, R) \quad (6)$$

where $\tilde{Q}_l = Q_l + \text{SelfAttn}(Q_l)$, Q_l is the input of l-th layer and R denotes the hidden reprensentation of encoder. As shown in Figure 2, the input and output of attention sub-layer in the decoder are very similar, especially the self-attention. Thus, we can regard that $\tilde{Q}_l \approx Q_l$, the F_l can be rewritten as follows:

$$F_l = Q_l + \underbrace{\text{SelfAttn}(Q_l) + \text{CrossAttn}(Q_l, R)}_{\textit{merge for parallel computing}} \quad (7)$$

The Eq. 7 provides the conditions for us to do parallel computing. The merging process is shown in Figure 4. Original cross-attention relies on the output of self-attention and it needs to wait until self-attention calculation is completed. We thereby merge the self-attention and cross-attention into a single one to further improve the decoding efficiency. This way is faster because the model can compute the attention results of self-attention and cross-attention simultaneously. Thus, the formula Eq. 2 and Eq. 4 can be rewritten as follows:

$$\begin{aligned}F_l &= F^S_l + F^C_l \\ &= A^S_l \cdot V^S_l + A^C_l \cdot V^C_l\end{aligned} \quad (8)$$

$$\tilde{F}_{l+i} = [A^S_l; A^C_l] \cdot [V^S_{l+i}; V^C_{l+i}] \quad (9)$$

Here, F^S_l, F^C_l, A^S_l, A^C_l, V^S_l, V^C_l represent self-attention results, cross-attention results, self-attention weights, cross-attention weights, self-attention values and cross-attention values of the l-th layer, respectively. $[;]$ denotes the concatenate operation. The main idea is that we sum the self-attention and cross-attention representations as a whole to decode in ARN model, instead of computing them separately. Similarly, the Eq. 5 also share the strength of parallelization.

3.2 Decoding Complexity

We investigate how the ARN accelerates Transformer compared to the original dot-product attention. For each decoding step, the self-attention first connects all positions with a constant number of sequentially executed operations (Eq. 1, $O(n \cdot d^2)$), and then obtains atention results via a weighted sum operations (Eq. 2, $O(n^2 \cdot d)$). Thus, the computation complexity of self-attention is $O(n \cdot d^2 + n^2 \cdot d)$. Similarly, the cross-attention is $O(m \cdot d^2 + m^2 \cdot d)$. The ARN model is fast because the weighted combination process (Eq. 3 and Eq. 5) are all element-wise operations, which require less computation and the Eq. 4 occupies a major computation. The self-attention and cross-attention adopting ARN method are $O(n^2 \cdot d)$ and

$O(m^2 \cdot d)$, respectively. See Table 1 for the details. In particular, m and n are smaller than the representation dimensionality d, which is most often the case with sentence representations used by NMT models. Moreover, the self-attention and cross-attention components are merged for higher parallelization. All these enable the ARN model to enjoy greater decoding efficiency.

4 Experiments

4.1 Experimental Settings

Datasets. We evaluate our proposed model on WMT14 and WMT17 translation tasks. The details follow as:

- WMT14 En→{De, Fr}. For En→De task, we choose newstest2013 as validation set and newstest2014 as test set. For En→Fr task, we validate the system on the combination newstest2012 and newstest2013 as validation set and test it on newstest2014.

- WMT17 En↔{Fi, De, Cs, Ru}. For validation, we concatenate the data of newstest2014-2016. For test, we choose newstest2017.

Table 2 shows the statistics of these datasets. For all datasets, we tokenize every sentence with Moses tokenizer (Koehn et al., 2007) and use byte pair encodings (BPE) with 32K split operations for subword segmentation (Sennrich et al., 2015). We remove sentences with more than 250 subword units. For WMT14 En→De translation task, we share the source and target vocabularies. We report the case-sensitive tokenized BLEU using *multi-bleu.perl*

Implementation Detail. For all machine translation tasks, our systems are based on an open-source implementation of *fairseq-py*[2]. We replicate the model setup of Vaswani et al. (2017). The standard implementation of Transformer baseline consists of a 6-layer encoder and a 6-layer decoder. The embedding size and FFN hidden size are 512 and 2048, respectively. The number of attention heads is set to 8. Dropout and label smoothing are used as regularization, both set to 0.1. We adopt Adam (Kingma and Ba, 2014) optimizer with an adaptive learning rate schedule as described in Vaswani et al. (2017), the warmup step and learning rate are 4K and $7\times e^{-4}$, respectively. All experiments are trained on 8 NVIDIA Tesla V100 GPUs with mixed-precision training and a batch size of 4096 tokens per GPU. For widely used WMT14 translation tasks, all models are trained for 100K steps as provided by Vaswani et al. (2017). For all WMT17 tasks, we stop training until the model no longer improves on the validation set. We average parameters of the last 5 checkpoints to obtain the final model. In inference, all models are decoded with half-precision on a single V100 GPU. By default, the batch size of decoding is set to 1 for avoiding invalid computations on padding. The beam size is 4 and length penalty is set to 0.6. All speed testing are based on the state-of-the-art inference implementation of Transformer with attention caching[3].

Source	Lang.	Train Set		Valid. Set		Test Set	
		sent.	word	sent.	word	sent.	word
WMT14	En→De	4.5M	220M	3000	110K	3003	114K
	En→Fr	35M	2.2B	26K	1.7M	3003	155K
WMT17	En↔Fi	2.6M	108M	8870	330K	3002	110K
	En↔De	5.9M	276M	8171	356K	3004	128K
	En↔Cs	52M	1.2B	8658	354K	3005	118K
	En↔Ru	25M	1.2B	8819	391K	3001	132K

Table 2: Data statistics (# of sentence pairs and # of words, M=million, B=billion, K=kilo)

4.2 Baselines

We compare our proposed ARN model with the following baselines:

- **Transformer**(Vaswani et al., 2017) is the most widely-used NMT system with self-attention mechanism.

- **AAN**(Zhang et al., 2018) is a classic lightweight attention NMT model, which leverages an average attention network for inference acceleration.

- **SAN**(Xiao et al., 2019) is another lightweight attention NMT model via sharing attention results among layers. For convenience, we simply adopt the sharing strategy per 2 layers, which can maintain a relatively high speed-up at the trade-off on BLEU performance.

We re-implement all the above baseline systems, and their experimental settings are consistent with our ARN model.

[2] https://github.com/pytorch/fairseq

[3] An engineering optimization technique, which cache the attention output of previous positions and then reuse it in following steps.

Model	BLEU	Δ_{BLEU}	Speed	Δ_{Speed}	#Param
Transformer	27.34	0.00	110.55	0.00%	58.7M
ARN$_2$	27.35	+0.01	134.91	+22.04%	55.0M
ARN$_3$	27.26	-0.08	149.89	+35.59%	53.7M
ARN$_6$	26.86	-0.48	159.62	+44.39%	52.4M

Table 3: BLEU scores [%] and translation speeds (token/sec) on WMT14 En→De task for different sharing policies. ARN$_n$ means that adopt a ARN strategy every n layer.

Source	Lang.	Model	BLEU	Δ_{BLEU}	Speed	Δ_{Speed}
WMT14	En→De	Transformer	27.34	0.00	110.55	0.00%
		ARN	27.26	-0.08	149.89	+35.59%
	En→Fr	Transformer	39.73	0.00	106.18	0.00%
		ARN	39.56	-0.17	149.58	+40.87%
WMT17	En→Fi	Transformer	21.50	0.00	107.49	0.00%
		ARN	21.47	-0.03	144.62	+34.54%
	Fi→En	Transformer	25.06	0.00	110.13	0.00%
		ARN	25.14	+0.08	149.24	+35.51%
	En→De	Transformer	28.66	0.00	109.02	0.00%
		ARN	28.46	-0.20	147.38	+35.19%
	De→En	Transformer	34.56	0.00	115.37	0.00%
		ARN	34.38	-0.18	151.56	+31.37%
	En→Cs	Transformer	23.94	0.00	111.03	0.00%
		ARN	23.62	-0.32	148.77	+33.99%
	Cs→En	Transformer	29.94	0.00	111.14	0.00%
		ARN	29.92	-0.02	145.62	+31.02%
	En→Ru	Transformer	30.69	0.00	110.40	0.00%
		ARN	30.24	-0.45	146.99	+33.14%
	Ru→En	Transformer	34.22	0.00	106.72	0.00%
		ARN	33.94	-0.28	148.67	+39.31%
Avg.		Transformer	29.56	0.00	109.80	0.00%
		ARN	29.40	-0.16	148.23	+35.00%

Table 4: BLEU scores [%] and translation speeds (token/sec) on WMT14 and WMT17 translation tasks.

Model	BLEU	Δ_{BLEU}	Speed	Δ_{Speed}	#Param
Transformer	27.34	0.00	110.55	0.00%	58.7M
AAN	27.16	-0.18	116.47	+5.36%	70.7M
SAN	27.17	-0.17	131.72	+19.15%	55.0M
ARN	27.26	-0.08	149.89	+35.59%	53.7M

Table 5: Comparison of different attention models on WMT14 En→De translation task.

4.3 Main Results

We test our approach on the widely used WMT14 En→De translation task. Table 3 reports the various results of BLEU scores and translation speeds, which adopt the simple sharing policy with ARN structure. The speed-up of the ARN$_3$ model is 1.35× with almost no decrease in BLEU. The ARN$_3$ is our ARN baseline system in the following sections. The BLEU will drop significantly if more layers are shared, but ARN model can provide a higher acceleration gain. On the other hand, our model requires fewer parameter numbers compared to Transformer baseline. This result shows that the original Transformer attention does have redundant computation, and simplifying it can achieve greater decoding efficiency. To further verify the effectiveness of our proposed approach, we conduct

Model	w/o KD		$w/$ KD	
	BLEU	Δ_{BLEU}	BLEU	Δ_{BLEU}
Transformer (**teacher**)	27.34	0.00	27.92	0.00
AAN	27.16	-0.18	27.89	-0.03
SAN	27.17	-0.17	27.62	-0.30
ARN	27.26	-0.08	27.95	+0.03

Table 6: ARN applying in knowledge distillation on WMT14 En→De translation task.

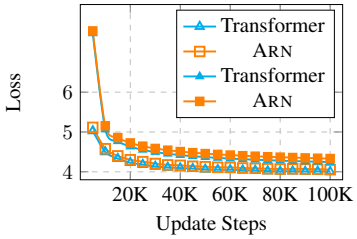

Figure 5: Convergence visualization. Loss vs. update steps on WMT14 En→De translation task (solid marked lines are training loss, hollow marked lines are validation loss).

experiments on ten WMT large-scale translation tasks. As shown in Table 4, the ARN significantly improves the speed for all these translation tasks, its speed-up is 1.35× faster on average. Also, the BLEU only drops 0.16, a very slight decline. Moreover, ARN achieve a stable speed-up, ranging from 31.02% to 40.87%. These results indicate that the ARN model is robust and can further improve the decoding efficiency on widely-range translation tasks.

In addition, we empirically compare the AAN and SAN models on WMT14 En→De translation task shown in Table 5. The three systems present a similar BLEU compared to Transformer baseline. Notably, ARN model achieves the highest speed-up with fewer parameter numbers. Although AAN, SAN and ARN can offer different degrees of decoding acceleration gains, they still suffer from slight performance degradation shown in Table 4 and Table 5. We use the most popular sequence-level knowledge distillation (KD) (Kim and Rush, 2016) for better performance and the Transformer baseline serves as our teacher model. As shown in Table 6, the KD method enables all three attention models get the performance improvements consistent with the Transformer baseline. Furthermore, the performance gap is close between ARN and Transformer baseline, and our model presents a strong generalization ability compared to AAN and SAN baselines.

Model	BLEU	Δ_{BLEU}	Speed	Δ_{Speed}	#Param
Transformer	27.34	0.00	110.55	0.00%	58.7M
+ Self-attention ARN	27.39	+0.05	117.45	+6.24%	56.7M
+ Cross-attention ARN	27.33	-0.01	127.58	+15.40%	56.7M
+ Both ARN	27.27	-0.07	131.87	+19.29%	54.7M
+ Merging	27.26	-0.08	149.89	+35.59%	53.7M

Table 7: Ablation study on WMT14 En→De translation task.

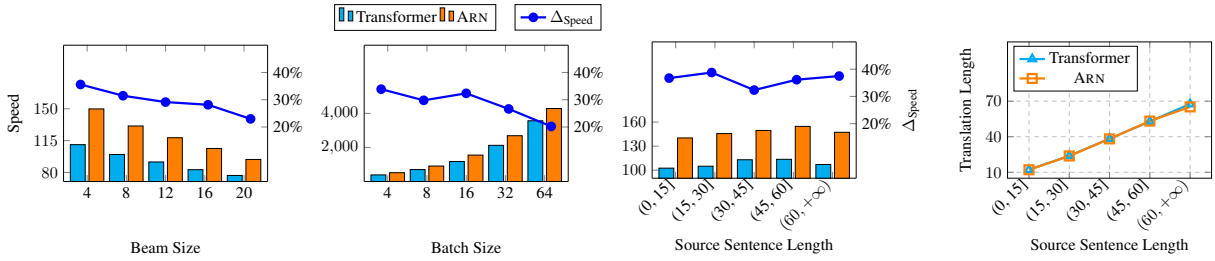

Figure 6: Speed (token/sec) and Δ_{Speed} [%] vs. beam size, batch size and source sentence length, translation length vs. source sentence length on WMT14 En→De translation task.

5 Analysis

Analysis on Convergency. As shown in Figure 5, we plot the loss curves of standard Transformer and ARN model on the training and validation sets, respectively. The two systems converge stably and both of their loss curves maintain a high degree of similarity. They also present similar performance in BLEU as shown in Table 3. All of these verify that the original attention in Transformer can be replaced with an ARN structure and there is no negative impact on model performance.

Ablation Study. To verify the acceleration contributions as well as the performance loss of different components in ARN model, we make an ablation study. As shown in Table 7, the ARN method adopted by the self-attention component achieves a speed-up of 6.24%, while 15.40% for cross-attention. This is because the original cross-attention operation is heavy for its long encoder representation (V) and applying the ARN method can bring greater decoding efficiency. When applying the ARN method to both self-attention and cross-attention components, the speed-up is 19.29%. It can achieve a higher acceleration gain (35.59%) when further merging self-attention and cross-attention components. Also, there is almost no effect on BLEU adopting the above different methods.

Sensitivity Analysis on Speed. We plot translation speed (in token/sec) and speed-up (in Δ_{Speed} [%]) as function of beam size, batch size and source sentence length. As shown in Figure 6, the ARN model can achieve relatively significant speed-up with the different beam sizes and batch sizes. However, the acceleration gains drop slightly with increasing beam size and batch size. This is because the speed-up will be weakened under the large matrix operation of the GPU. In addition, our model is robust and it gets the consistent improvement under the various source sentence length. Furthermore, both the Transformer baseline and our proposed model generate translations with similar lengths as shown on the right in Figure 6. This finding suggests that the acceleration gain comes from the well-designed model structure, not the translation length.

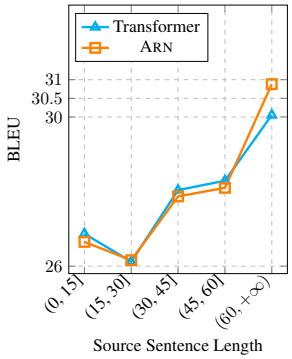

Figure 7: Different source sentence lengths vs. BLEU on WMT14 En→De translation task.

Analysis on Translation Quality. It is well-known that the NMT model is difficult to handle

#	Model	BLEU	Δ_{BLEU}	Speed	Δ_{Speed}	#Param
1	Transformer (6 + 6)	27.34	-	110.55	-	58.7M
2	Deep encoder, shallow decoder Transformer (12 + 2)	27.38	0.00	224.48	0.00%	60.7M
3	+ Decoder self-attention ARN	27.59	+0.21	233.04	+3.81%	60.2M
4	+ Decoder cross-attention ARN	27.32	-0.06	247.33	+10.18%	60.2M
5	+ Decoder both ARN	26.83	-0.55	250.37	+11.15%	59.5M
6	+ Merging	26.82	-0.56	260.73	+16.15%	59.4M
7	+ Encoder ARN (using ARN_2)	27.19	-0.19	235.12	+4.74%	57.7M

Table 8: ARN applying in deep encoder, shallow decoder Transformer on WMT14 En→De translation task.

long-distance dependencies and the issue of under-translation (Tu et al., 2016; Zheng et al., 2019) is prone to occur when translating long sentences. For this, we study the ARN model performance with the different input of sentence lengths as shown in Figure 7. Interestingly, the proposed model performs well when dealing with the longest sentences. Specifically, our model shows relatively poorly on shorter sentences but significantly better performance on longest sentences. This suggests that certain features have a great influence on model performance when translating long sentences, and the phenomenon indicates that ARN structure can better capture the long-distance dependency through the fusion of high-level and low-level features iteratively.

Deep Encoder, Shallow Decoder Transformer. Standard Transformer suffers from heavy inference cost due to the multi-layer stacked decoder (6-layer). A popular solution is to balance the encoder and decoder depths for speeding up Transformer (Kasai et al., 2020). We rebuild the Transformer with a deep encoder (12-layer) and a shallow decoder (2-layer) as a stronger baseline. As shown in Table 8, the balanced baseline (12/2) is more than 2× faster without loss in BLEU with similar parameter numbers. When applying ARN method to self-attention and cross-attention on the decoder side, it achieves 3.81% (line 3) and 10.18% (line 4) speed-up, respectively. Also, there is no negative impact on BLEU. The BLEU even increases by 0.21 for adopting ARN on the decoder-side self-attention. Interestingly, the BLEU drop significantly if applying ARN to both self-attention and cross-attention (line 5 and line 6). This is because the decoder is shallow with less parameter redundancy, which enables the model very sensitive when ARN is applied both to the two components. In contrast to the decoder, that encoder suffers from more severe redundancy in the balanced baseline, and the BLEU almost no drop with a higher reduction of parameters (line 7). To summarize, our ARN balanced baseline can achieve a speed-up of 2.24× without sacrificing performance compared to standard Transformer baseline (line 4 vs. line 1).

6 Related Work

Standard Transformer suffers from the high inference cost due to the auto-regressive generation schema and the heavy use of dot-product attention operations. A classic solution is to generate the entire target sequence at one time by using non-autoregressive inference method (Gu et al., 2017; Guo et al., 2019; Wang et al., 2019). This way offers high decoding efficiency but it is hard to train. Another representative solution adopts a local attention strategy (Kitaev et al., 2020; Beltagy et al., 2020) or simplifies attention structure (Katharopoulos et al., 2020), which can effectively reduce the computation of attention module. But these works are designed for acceleration of very long sequence tasks (e.g., image generation, automatic speech recognition, etc.). NMT inference acceleration methods have been investigated for years, including knowledge distillation (Hinton et al., 2015; Kim and Rush, 2016; Lin et al., 2020; Wang et al., 2021), vocabulary selection (L'Hostis et al., 2016; Sankaran et al., 2017; Shi and Knight, 2017), low-precision computation (Micikevicius et al., 2017; Quinn and Ballesteros, 2018; Aji and Heafield, 2020), kernel fusion (Wu et al., 2021), LayerDrop (Fan et al., 2019) and etc. Compared with the above methods, our work follows another line of work to learn a lightweight attention NMT model and prove its effectiveness.

Zhang et al. (2018) show that the self-attention network is not necessary and a simple averaging is enough. Compared with the above method, our network structure is simple and it is easy to implement. We improve both the self-attention and

cross-attention components. Xiao et al. (2019) observe that the most attention distributions are similar and thus share these distributions among layers. This method can be regarded as a structure pruning for Transformer, which may lead to model performance degradation. Our approach, which fuses low-level features to further enhance model confidence, is an improvement over this method. Li et al. (2021) propose the compressed attention network that simplifies the transformer architecture to achieve a higher parallelism. This method can be regarded as an equivalent transformation of the Transformer structure, it is orthogonal to our research.

ARN structure can be viewed as a weighted residual network, which acquires the attention results by reusing the features across layers. The main idea is similar to (He et al., 2020), but our ARN is different in both motivation and network structure. To our knowledge, we are the first to design a residual network for NMT decoding acceleration, proposing the combination of the features across layers for reconstructing the attention.

7 Conclusion

We have investigated ARN, an alternative lightweight attention structure for faster inference of Transformer. Experiments on a range of WMT translation tasks show that ARN offers a significant speed improvement over a strong Transformer baseline without sacrificing translation performance. Results on widely used WMT14 En→De machine translation tasks demonstrate that our model can simultaneously deliver superior acceleration and translation performance with fewer parameters, compared to previous work like AAN and SAN.

Acknowledgements

We thank the anonymous reviewers for their constructive and thoughtful comments. Thanks to Tong Xiao, Yinqiao Li and Bei Li for their insightful suggestions.

References

Alham Fikri Aji and Kenneth Heafield. 2020. Compressing neural machine translation models with 4-bit precision. In *Proceedings of the Fourth Workshop on Neural Generation and Translation*, pages 35–42.

Maximiliana Behnke and Kenneth Heafield. 2020. Losing heads in the lottery: Pruning transformer attention in neural machine translation. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 2664–2674.

Iz Beltagy, Matthew E Peters, and Arman Cohan. 2020. Longformer: The long-document transformer. *arXiv preprint arXiv:2004.05150*.

Angela Fan, Edouard Grave, and Armand Joulin. 2019. Reducing transformer depth on demand with structured dropout. *arXiv preprint arXiv:1909.11556*.

Jiatao Gu, James Bradbury, Caiming Xiong, Victor OK Li, and Richard Socher. 2017. Non-autoregressive neural machine translation. *arXiv preprint arXiv:1711.02281*.

Junliang Guo, Xu Tan, Di He, Tao Qin, Linli Xu, and Tie-Yan Liu. 2019. Non-autoregressive neural machine translation with enhanced decoder input. In *Proceedings of the AAAI conference on artificial intelligence*, volume 33, pages 3723–3730.

Ruining He, Anirudh Ravula, Bhargav Kanagal, and Joshua Ainslie. 2020. Realformer: Transformer likes residual attention. *arXiv preprint arXiv:2012.11747*.

Geoffrey Hinton, Oriol Vinyals, and Jeff Dean. 2015. Distilling the knowledge in a neural network (2015). *arXiv preprint arXiv:1503.02531*, 2.

Jungo Kasai, Nikolaos Pappas, Hao Peng, James Cross, and Noah A Smith. 2020. Deep encoder, shallow decoder: Reevaluating non-autoregressive machine translation. *arXiv preprint arXiv:2006.10369*.

Angelos Katharopoulos, Apoorv Vyas, Nikolaos Pappas, and François Fleuret. 2020. Transformers are rnns: Fast autoregressive transformers with linear attention. In *International Conference on Machine Learning*, pages 5156–5165. PMLR.

Yoon Kim and Alexander M Rush. 2016. Sequence-level knowledge distillation. *arXiv preprint arXiv:1606.07947*.

Diederik P Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*.

Nikita Kitaev, Łukasz Kaiser, and Anselm Levskaya. 2020. Reformer: The efficient transformer. *arXiv preprint arXiv:2001.04451*.

Philipp Koehn, Hieu Hoang, Alexandra Birch, Chris Callison-Burch, Marcello Federico, Nicola Bertoldi, Brooke Cowan, Wade Shen, Christine Moran, Richard Zens, et al. 2007. Moses: Open source toolkit for statistical machine translation. In *Proceedings of the 45th annual meeting of the association for computational linguistics companion volume proceedings of the demo and poster sessions*, pages 177–180.

Gurvan L'Hostis, David Grangier, and Michael Auli. 2016. Vocabulary selection strategies for neural machine translation. *arXiv preprint arXiv:1610.00072*.

Yanyang Li, Ye Lin, Tong Xiao, and Jingbo Zhu. 2021. An efficient transformer decoder with compressed sub-layers. *arXiv preprint arXiv:2101.00542*.

Ye Lin, Yanyang Li, Ziyang Wang, Bei Li, Quan Du, Tong Xiao, and Jingbo Zhu. 2020. Weight distillation: Transferring the knowledge in neural network parameters. *arXiv preprint arXiv:2009.09152*.

Paul Michel, Omer Levy, and Graham Neubig. 2019. Are sixteen heads really better than one? *Advances in neural information processing systems*, 32.

Paulius Micikevicius, Sharan Narang, Jonah Alben, Gregory Diamos, Erich Elsen, David Garcia, Boris Ginsburg, Michael Houston, Oleksii Kuchaiev, Ganesh Venkatesh, et al. 2017. Mixed precision training. *arXiv preprint arXiv:1710.03740*.

Jerry Quinn and Miguel Ballesteros. 2018. Pieces of eight: 8-bit neural machine translation. *arXiv preprint arXiv:1804.05038*.

Baskaran Sankaran, Markus Freitag, and Yaser Al-Onaizan. 2017. Attention-based vocabulary selection for nmt decoding. *arXiv preprint arXiv:1706.03824*.

Rico Sennrich, Barry Haddow, and Alexandra Birch. 2015. Neural machine translation of rare words with subword units. *arXiv preprint arXiv:1508.07909*.

Xing Shi and Kevin Knight. 2017. Speeding up neural machine translation decoding by shrinking run-time vocabulary. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 574–579.

Zhaopeng Tu, Zhengdong Lu, Yang Liu, Xiaohua Liu, and Hang Li. 2016. Modeling coverage for neural machine translation. *arXiv preprint arXiv:1601.04811*.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. *Advances in neural information processing systems*, 30.

Fusheng Wang, Jianhao Yan, Fandong Meng, and Jie Zhou. 2021. Selective knowledge distillation for neural machine translation. *arXiv preprint arXiv:2105.12967*.

Yiren Wang, Fei Tian, Di He, Tao Qin, ChengXiang Zhai, and Tie-Yan Liu. 2019. Non-autoregressive machine translation with auxiliary regularization. In *Proceedings of the AAAI conference on artificial intelligence*, volume 33, pages 5377–5384.

Kaixin Wu, Bojie Hu, and Qi Ju. 2021. Tentrans high-performance inference toolkit for wmt2021 efficiency task. In *Proceedings of the Sixth Conference on Machine Translation*, pages 795–798.

Tong Xiao, Yinqiao Li, Jingbo Zhu, Zhengtao Yu, and Tongran Liu. 2019. Sharing attention weights for fast transformer. *arXiv preprint arXiv:1906.11024*.

Biao Zhang, Deyi Xiong, and Jinsong Su. 2018. Accelerating neural transformer via an average attention network. *arXiv preprint arXiv:1805.00631*.

Zaixiang Zheng, Shujian Huang, Zhaopeng Tu, Xin-Yu Dai, and Jiajun Chen. 2019. Dynamic past and future for neural machine translation. *arXiv preprint arXiv:1904.09646*.

Interactive Post-Editing for Verbosity Controlled Translation

Prabhakar Gupta[1], **Anil Nelakanti**[1], **Grant M. Berry**[2], **and Abhishek Sharma**[1]

[1]Amazon Prime Video
[1]{prabhgup,annelaka,naabhiss}@amazon.com
[2]Department of Spanish, Villanova University
[2]grant.berry@villanova.edu

Abstract

We explore Interactive Post-Editing (IPE) models for human-in-loop translation to help correct translation errors and rephrase it with a desired style variation. We specifically study verbosity for style variations and build on top of multi-source transformers that can read source and hypothesis to improve the latter with user inputs. Token-level interaction inputs for error corrections and length interaction inputs for verbosity control are used by the model to generate a suitable translation. We report BERTScore to evaluate semantic quality with other relevant metrics for translations from English to German, French and Spanish languages. Our model achieves superior BERTScore over state-of-the-art machine translation models while maintaining the desired token-level and verbosity preference.

1 Introduction

Recent machine translation (MT) models (Sutskever et al., 2014; Bahdanau et al., 2015) have shown to excel with aspects of translation quality like adequacy and fluency but these models still suffer notable shortcomings like out-of-domain data, low-resource languages, rare words and longer sentences (Koehn and Knowles, 2017). Hence, MT systems are often supplemented by human translators for editing and correcting MT outputs to achieve the desired quality bar for various use-cases (Peris et al., 2017). Broadly, MT employing human input can be classified as one of two types: manual post-editing (MPE) and interactive post-editing (IPE) (Escribe and Mitkov, 2021). MPE relies on humans to make all necessary edits on top of MT output to deliver the final translation. Whereas, IPE uses a human-in-the-loop approach: the model offers human translators various cues like auto-complete suggestions, word look-ups, etc until the human arrives at a translation. Both approaches have their trade-offs and it is observed that while MPE is slower, it delivers a higher quality output relative to the faster IPE (Green et al., 2014). Our work is a human-in-the-loop model that aims to make efficient use of human effort in delivering improved translations.

Translation quality, largely encompassing adequacy and fluency, has been the primary focus of most MT studies. Some recent work has explored other aspects of MT models like translation diversity, word choice (rare vs frequent words), translation style, etc (Wang et al., 2021; Niu and Carpuat, 2020; Agrawal and Carpuat, 2019; Marchisio et al., 2019; Lakew et al., 2019; Niu et al., 2018; Yamagishi et al., 2016). This style-aware modelling further broadens the scope and usability of MT, even more so when users can control levers to achieve the desired style variation. Our work identifies one of the most important style features – verbosity or translation length. Controlling length of translation output has been studied before for NMT models (Lakew et al., 2019) but to the best of our knowledge our work is the first to propose a solution for controlling length of translation in an interactive human-in-the-loop setting. Length is extremely crucial in many layout constrained translations use-cases like subtitling where the same amount of information needs to be available on-screen at a given point in time independent of the subtitle language.

In this work, we propose a interactive post-editing system that leverages multi-source transformers to offer users:

- Interactive control for corrections,
- Support for verbosity variation and corresponding translation customization; and
- Reduced human effort by providing alternative word and phrase choices.

Proceedings of the 29th International Conference on Computational Linguistics, pages 5119–5128
October 12–17, 2022.

2 Related Work

We review some recent studies focusing on automatic and interactive post-editing models germane to our work. Automatic Post-Editing (APE) is the task of automatically correcting the output of an (MT) system. APE models can be used to adapt a general purpose MT systems to new domains, fix errors in MT outputs and, in general, reduce human post-editing effort (Chatterjee et al., 2015). Transformer-based models for APE systems (Sharma et al., 2021; Yang et al., 2020; Chatterjee et al., 2020) have eclipsed models relying on statistical MT in recent years (Simard et al., 2007; Béchara et al., 2012). APE models inherit all the drawbacks of NMT since there is no human involved. They are excellent at domain adaption but often fail to improve the quality of state-of-the-art NMT models (Sharma et al., 2021) and the role and relevance of APE is often debated (do Carmo et al., 2021).

Post-editing models models require processing of both source text and MT output in order to generate a revised translation. Typically, two separate encoders are used – one for each source text and corresponding candidate translation – in addition to a single decoder responsible for generating output. We adapt one such model the Multi-Source Transformer (MST) (Tebbifakhr et al., 2018) for the current research. There are alternate methods that use two sequence-to-sequence models instead and merge the resulting distributions (Junczys-Dowmunt and Grundkiewicz, 2016). Merging distributions post-hoc is inadvisable, as complex patterns cannot be learned as easily and care must be taken to ensure that the combined distribution remains representative. We extend the MST approach for our work and use it train interactive human-in-loop models with user control to improve translation quality and style.

In contrast to standard post-editing models, IPE models consume user inputs to revise candidate translations. User inputs could be tokens that should either be dropped or retained from candidate or source sentences. QuickEdit model (Grangier and Auli, 2018) is an example that uses strike-out interactions to gather user tokens that should be dropped from MT output. Similarly, TouchEditing model (Wang et al., 2020) supports substitution, deletion, reordering and insertion operations on tokens. Support for richer and more complex interactions makes the model more flexible and easier for users. Our work borrows token-level interactions on source text and candidate sentences (we call them hypothesis) from related literature that help correct errors and improve translation quality. We extend this further to control verbosity of translation.

Style-aware language generation has drawn considerable attention from researchers recently. They have been studied for paraphrasing and translating text with the desired style properties. Style properties like (active/passive) voice (Yamagishi et al., 2016), formality (Niu et al., 2018; Niu and Carpuat, 2020), complexity (Agrawal and Carpuat, 2019; Marchisio et al., 2019), and verbosity (Lakew et al., 2019) are some examples that were explored in various applications. We evaluate one of the most important style features – verbosity or translation length in the context of IPE in this work.

3 Problem and Approach

Our primary goal is to train a sequence-to-sequence model that can improve candidate translations of the source with user cues. The model $\mathcal{M}(s, h, \mathcal{I})$ takes as its input a pair of sentences (source text s and a hypothesized translation h) as well as a set $\mathcal{I}$ of user interaction cues. The model tries to improve the translation of the source while leveraging user cues and accommodating those that are feasible. Post-edits can be successively applied by translators until satisfactory translation as $h_{k+1} = \mathcal{M}(s, h_k, \mathcal{I}_k)$.

Users generate interactions on the source and hypothesis sentences that the model uses to improve the translation. We support two categories of interactions; (1) token-level interactions (2) length interactions. For token-level interactions, the user has access to four unique operations $\mathcal{I} = \{keep, delete, insert, replace\}$. At each iteration, interactions can reflect any subset of this set $\mathcal{I}_k \subset \mathcal{P}(\mathcal{I})$, including an empty set $\emptyset$ which reduces to APE. The *keep* interaction is used to mark tokens in hypothesis that the user wishes to retain in the revised translation and the *delete* interaction captures tokens from the hypothesis the user prefers to drop from the revised translation (similar to the strike-out operation in QuickEdit). The *replace* interaction allows user to mark tokens in the hypothesis that are in the right position but need to be changed in the revised translation. For the *insert* interaction, we provide the user with a translation language model (TLM) (similar to (Con-

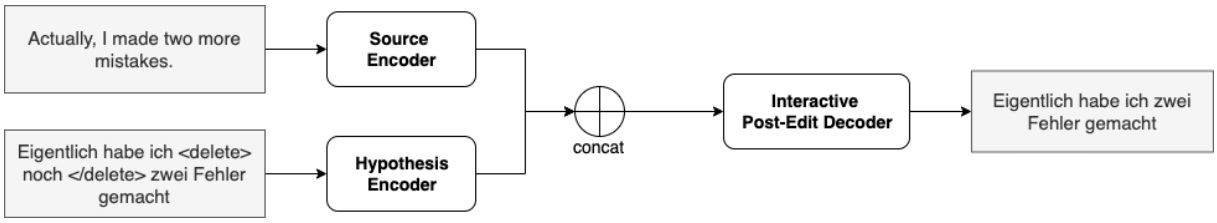

Figure 1: The model architecture. The source and the hypothesis are encoded using two separate transformer encoders. The resulting encodings are concatenated and given as the input to the decoder.

neau and Lample, 2019)) which allows the user to add a token in the hypothesis at a position of their choice. These interactions are passed by enclosing the relevant tokens from the sentence in appropriate control tags. For example, *keep* is indicated by using <keep> and </keep> tags around relevant tokens in the hypothesis. [1]

The length interaction is used to affect the verbosity of the output. User can request three possible variations on this interaction; increment (↑), decrement (↓) or isometric translation (↔). Isometric translation refers to generating translations similar in length to the source. We follow the same definition of isometric translation as described in (Lakew et al., 2021). Any translation with target character length in $\pm 10\%$ of source character length is considered isometric. If the user wants to make the revised translation longer/shorter compared to the hypothesis, they indicate this with the increment or decrement length interactions. If the user wants to make the revised translation to be of similar length as source, they pick isometric translation. The corresponding control tag is prepended to the hypothesis sentence before it is passed to the encoder. Table 1 shows examples of the various interactions described above.

Model architecture. We use a Multi-Source Transformer (MST) proposed by (Tebbifakhr et al., 2018) for our task. We specifically consider the implementation by (Wan et al., 2020) [2] based on Fairseq (Ott et al., 2019). The MST network consists of two transformer encoders and one transformer decoder (Vaswani et al., 2017). We use the same hyperparameters as in the Transformer (base) model with 6 layers in both the encoder and the decoder, 512 as the embedding dimension, 2048 as dimension of the feed forward layer and 8 heads for multi-head attention. The two encoders are used for encoding the source and the hypothesis sentences separately, while the decoder is used for generating an improved translation. Outputs from the encoders are concatenated and then used to generate keys and values for the encoder-decoder attention sub-layers in the decoder. (See Figure 1 for an illustration of the model.) We maximize the conditional log-likelihood $\mathcal{L}_\theta(D)$ of the training data D over the network parameters θ,

$$\theta^* = \underset{\theta}{\text{argmax}} \mathcal{L}_\theta(D) \equiv \sum_{(s,t,h,\mathcal{I})\in D} \log \mathrm{P}_\theta(t|s,h,\mathcal{I}),$$

where $\mathrm{P}_\theta(t|s,h,\mathcal{I})$ is probability of target sentence given the inputs that is auto-regressively estimated as $\prod_{k=1}^{|t|} \mathrm{P}_\theta(y_k|y_{k-1},s,h,\mathcal{I})$. The model $\mathcal{M}$ performs auto-regressive decoding with beam search using the distribution of the next token $\mathrm{P}_{\theta^*}(y_k|y_{<k},s,h,\mathcal{I})$ conditioned on the previously generated tokens $y_{<k}$, the model inputs, and optimal parameters θ^*. We use a beam width of 5 in our experiments and do not use any hard constraints during decoding.

4 Interaction Data Simulation

To the best of our knowledge, there are no publicly available datasets capturing the various interactions we aim to learn through $\mathcal{M}$. There are public datasets for automated post-editing (like (Chatterjee et al., 2019)) but they only have source and hypothesis pairs with post-edit required on the latter to improve the translation. They do not cover the spectrum of interactions we are considering in our work. Ideally, we would need an interface to collect the necessary data spanning our interaction set where professional translators can mark the edits necessary to arrive at a satisfactory translation from given inputs. This method of data collection is expensive and does not scale to larger sets of supported interactions. Alternatively, we can synthesize the input by simulating user interactions from

[1] The description of user interface used for these interactions is briefly described in Section 5.2 and other details beyond the scope of this work.

[2] https://github.com/zerocstaker/constrained_ape

Table 1: Examples of inputs (source s, hypothesis h, interaction $\mathcal{I}$) and corresponding model output (translation $\hat{t}$) along with human given reference translation (t). (A) Interaction was to keep two words: *"Deshalb"* and *"führend"* from the h and they were correctly kept in the $\hat{t}$. (B) Interaction was to delete one word: *"noch"* which means *"more"* in the context. The literal translation of s would require *"noch"* to be there but because of Delete interaction it was removed from $\hat{t}$. (C) Longer Interaction translated the 22 characters long h to a 43 characters long $\hat{t}$ while maintaining the meaning of translation. (D) Shorter Interaction translated 45 characters long h to a 21 characters $\hat{t}$. (E) Length of s is 48 characters making the 39 character long h non-isometric translation but generated $\hat{t}$ is 45 character long making it an isometric translation.

(A) Token Interaction - Keep
s: So I think we have to be in the lead.
h: <*keep*> Deshalb </*keep*> denke, wir <*keep*> führend </*keep*> sein.
$\hat{t}$: Deshalb denke ich, dass wir führend sein müssen.
t: Deshalb denke ich, müssen wir führend sein.
(B) Token Interaction - Delete
s: Actually, I made two more mistakes.
h: Eigentlich habe ich <*delete*> noch </*delete*> zwei Fehler gemacht.
$\hat{t}$: Eigentlich habe ich zwei Fehler gemacht.
t: Eigentlich machte ich zwei Fehler.
(C) Length Interaction - Longer
s: He kept pointing here.
h: <*length*↑> Er zeigte hier weiter.
$\hat{t}$: Er hat diesbezügliche Fortschritte gemacht.
t: Er zeigte immer hier hin.
(D) Length Interaction - Shorter
s: It's a complete denial of mistakes.
h: <*length*↓> Es ist eine völlige Verweigerung von Fehlern.
$\hat{t}$: Es ist völlig falsch.
t: Es ist eine total Verleugnung von Fehlern.
(E) Length Interaction - Isometric
s: Is it something about the details or the colors?
h: <*length*↔> Geht es um die Details oder die Farben?
$\hat{t}$: Geht es um die Einzelheiten oder die Farben?
t: Geht es dabei um die Details oder die Farben?

bilingual parallel text data that is readily available. We take this route similar to various post-editing studies (Grangier and Auli, 2018; Tebbifakhr et al., 2018) to identify and mark tokens with interactions that the model can leverage to improve the hypothesis.

Interaction simulation. We begin with a set of high quality data bitext data samples $(s, t) \in S$ for the language pair of interest. A high-quality pre-trained machine translation system T_S is used to generate hypotheses $h = T_S(s)$. User interactions are then simulated from the hypothesis and reference pairs as $\mathcal{I} = U(h, t)$. This gives us the input triplets $(s, h, \mathcal{I})$ to train our model. Following are the details of the simulation function U that imitates translator interactions from (h, t) sentence pairs.

- **Delete tokens.** The *delete* interaction allows user to specify token substrings in the hypothesis that should be dropped from the final translation. It helps the model learn to correct over-translated phrases and rephrase translation accordingly. We simulate *delete* interaction by sampling from substrings in h do not appear in the corresponding t.
- **Keep tokens.** The *keep* interaction has the opposite effect of *delete* and is used to retain tokens from the hypothesis that are good translations of the corresponding source. Contrary to *delete*, we sample from substrings from h that match t to identify relevant tokens.
- **Replace or Insert tokens.** The *replace* interaction allows user suggest replacements to

some tokens in the hypothesis. These suggestions could be generated from other models like a masked language model or manually entered by users. For training, we sample the substrings from h that do not appear in the t but there is an acceptable replacement in the t. We change the hypothesis by introducing the replacement tokens in the hypothesis within <keep> tags. Insert token operation operates similarly except that while replace token operation applies to a token in the hypothesis, insert token can be used in between tokens.

- **Length Interaction.** This interaction is useful in changing the verbosity of the translation candidates relative to h or s. The user can specify whether the output should be *longer* or *shorter* than h or in same range compared to s. The *length* interaction can be of very high interest in domains where the length of the final translation is crucial. For example, movie subtitle translations impose display limitations on screen that constrain the length of the text (Gupta et al., 2019). Using $|\cdot|$ to refer to character length of a sentence and a hyperparameter $\delta_l = 0.1$, we mark the interaction as *longer* when the ratio $(|t| - |h|)/|t| > \delta_l$, *shorter* if $(|t| - |h|)/|t| > -\delta_l$ and *isometric* if $(|t| - |s|)/|s| < \delta_l$ and $(|h| - |s|)/|s| > \delta_l$. The samples that fall in neither bucket are marked with *no-preference*.

We ignore all candidate substrings with more than eight tokens for token-interactions and skip any substrings that only have punctuation tokens. A token can belong to no more than one interaction and for a given input triplet multiple operations can be sampled across token substrings. We do not limit the number of token-interactions that are present in a sample but we control the number of interactions by occurance probabilities of each interaction. To make the model more robust, for each sample there was a 5% chance of getting a noisy token-level interaction of random length. For length interaction, again there was a 5% chance of prepending a random interaction from *longer*, *shorter*, *isometric* or *no-preference*. We compute the interactions for each sample on-the-fly and because of the random chances we have introduced in interaction simulation, we are able to train the model with multiple versions of same sample.

At inference time, user can provide any or all of the length interaction and token-level tags to the model. Table 1 shows examples for each of the described interactions.

5 Experiments and Results

Train and Test data. We use two public parallel datasets for training the model – European Parliament Proceedings Parallel Corpus (EuroParl) (Koehn, 2005) and MuST-C dataset (Cattoni et al., 2021). Both datasets represent high quality parallel corpora. MuST-C is a multilingual speech translation corpus with hundred hours of audio recordings from English TED Talks, which are automatically aligned at the sentence level with their manual transcriptions and translations. We do a 99-1 split of dataset into train, validation sets. We use two public test datasets – FLORES (Goyal et al., 2021) and MuST-C test set provided with dataset. FLORES is a high-quality many-to-many multilingual translation benchmark dataset for 101 languages. Table 2 provides the distribution statistics of each interaction of our train-validation-test sets by target language.

Table 2: Train and Test dataset sizes

Dataset Name	Split	En-De	En-Es	En-Fr
EuroParl	Train	1.7M	1.81M	1.77M
MuST-C		217K	250K	257K
FLORES	Test	997	997	997
MuST-C		2641	2502	2632

We use pretrained machine translation models, OPUS-MT (Tiedemann and Thottingal, 2020) to generate a hypothesis from source. For En-De, we use an additional model for generating hypothesis – FAIR's submission for WMT19 news translation task (Ng et al., 2019). We use BERT (Devlin et al., 2019) tokenizer provided in HuggingFace Tokenizer[3] to tokenize our inputs instead of generating a new dictionary. We clean the data by removing samples with more than 250 tokens.

We chose the Adam optimiser (Kingma and Ba, 2015) with $(\beta_1, \beta_2) = (0.9, 0.98)$ and an initial learning rate of $5e$-4 with inverse square-root decay scheduled after 4000 warm-up steps. We reduce the number of trainable parameters by using joint vocabulary for source, hypothesis and target, and by sharing the input and output embedding for the decoder. For regularisation, we use dropout (Srivastava et al., 2014) with a value 0.3 and weight decay

[3] https://huggingface.co/bert-base-multilingual-cased

Table 3: BERTScore of do nothing baseline (OPUS-MT) compared against our model with no interactions (APE condition) and all interactions (IPE model). More detailed scores in Table 4

Dataset	Model	En-De	En-Es	En-Fr
FLORES	Do Nothing Baseline	0.887	0.875	0.914
	No Interactions	0.881	0.87	0.907
	All Interactions	**0.922**	**0.923**	**0.94**
MuST-C	Do Nothing Baseline	0.877	0.89	0.903
	No Interactions	0.878	0.891	0.902
	All Interactions	**0.926**	**0.934**	**0.941**

of $1e$-4. To train the model, we use label-smoothed cross-entropy criterion with the smoothing parameter set to 0.1. For each language-pair we train a different model. We train each model for 24 epochs with one dataset per epoch and choose the checkpoint that gives the lowest loss on the validation set.

5.1 Results and Observations

We report results on model generated translations on the test $\hat{t} = \mathcal{M}(s, h, \mathcal{I})$ and focus our study on two specific aspects; constraint satisfaction (CS) and translation quality (TQ). For each interaction, we define the constraint satisfaction criterion in respective subsections. Translation quality is evaluated using the BERTscore (Zhang et al., 2020). BERTScore correlates better to the human judgement than its predecessors like BLUE (Papineni et al., 2002).

As reported in Table 3, the baseline BERTScore(h, t) wherein h are translations from OPUS-MT models is at 0.892 when averaged across three language pairs for FLORES test set and at 0.89 for the MuST-C test set. Using the full interaction set $\mathcal{M}(s, h, \mathcal{I})$ gives 0.92 and 0.928 BERTScore on FLORES and MuST-C respectively averaged over three language pairs. For **Do Nothing Baseline**, $\mathcal{M}(s, h, \emptyset)$ with no interaction during inference gives 0.886 and 0.877 BERTScore on both sets respectively averaged over three language pairs. This is similar to automatic post-editing (APE) where the model receives no human intervention. Previous works have shown in past that outperforming high-quality NMT models for APE with general data is extremely difficult (Sharma et al., 2021) and we observe a similar trend with our results.

Token Level Interactions. User inputs act like soft constraints and are not hard enforced hence the translation output does not always conform with the interaction request. We do a study to evaluate if the token-level controls also reflect similarly in the model outputs $\hat{t}$ as intended by the specified user interaction. We only report the numbers for *keep* and *delete* interactions since *insert* and *replace* interactions are essentially just *keep* interactions for the model as explained in Section 4. For the *keep* interactions, we check what percentage of tokens marked in h appear in the output translation $\hat{t}$. Similarly, for the *delete* interaction, we calculate the percentage of tokens that were marked in h for deletion and do not appear in $\hat{t}$. Table 4 reports corresponding results showing a high level of agreement in generated $\hat{t}$ (90% for *keep* and 60% for *delete*) with user requests. We only report the numbers for samples where at least one such interaction was possible, we skip other sentences where that interaction was not feasible. We saw a further improvement in performance when we used these interactions together as compared to just one interaction at a time (93% for *keep* and 65% for *delete*).

Length Interactions. For length interactions, it is more important to understand if the user request for change in verbosity is met with. Correspondingly, we compute change in verbosity between h and $\hat{t}$ to evaluate the affect of style controls. We take a ratio of character lengths $|\hat{t}|/|h|$ and report averages separately for *No preference*, *Longer* and *Shorter* control options. The results, summarized in Table 4, clearly show the intended trend. The model's ability to manipulate verbosity without negatively impacting translation quality (as measured by BERTScore) is evident.

Isometric Translation. Contrary to other length interactions, where the verbosity is relative to the length of h, for Isometric Translation verbosity is determined with respect to the length of s. With this in mind, we take a ratio of token lengths $|\hat{t}|/|s|$

Table 4: Constraint Satisfaction (CS) and Translation Quality (TQ) (1) Token Level Interactions: CS is tokens % marked with the interaction in hypothesis h and were satisfied in output translation $\hat{t}$. (2) Length Interactions: CS is the average ratio of character length of $\hat{t}$ and the that of h. (3) Isometric Translation: CS is the average ratio of character length of $\hat{t}$ and the that of s. No Preference is the ratio of t and s. *Custom NMT* is the NMT model we trained to specifically generate isometric translations. For TQ, we use BERTScore (Zhang et al., 2020).

		En-De		En-Es		En-Fr	
		CS	**TQ**	**CS**	**TQ**	**CS**	**TQ**
Token Level Interactions							
FLORES	Keep	0.913	0.879	0.928	0.87	0.921	0.906
	Delete	0.581	0.882	0.584	0.871	0.631	0.908
MuST-C	Keep	0.961	0.878	0.976	0.892	0.967	0.902
	Delete	0.586	0.879	0.605	0.893	0.606	0.904
Length Interactions							
FLORES	No Preference	0.92	0.881	0.923	0.87	0.926	0.907
	Longer	0.941	0.88	0.949	0.869	0.941	0.906
	Shorter	0.82	0.864	0.831	0.857	0.872	0.897
MuST-C	No Preference	0.865	0.878	0.868	0.891	0.869	0.902
	Longer	0.898	0.878	0.92	0.886	0.9	0.902
	Shorter	0.775	0.862	0.79	0.881	0.823	0.895
Isometric Translation							
FLORES	No Preference	1.174	0.887	1.199	0.875	1.193	0.914
	Custom NMT	0.932	0.823	1.051	0.851	1.035	0.875
	Ours	1.069	0.877	1.084	0.868	1.137	0.906
MuST-C	No Preference	1.121	0.877	0.998	0.89	1.147	0.903
	(Lakew et al., 2019)	1.02	-	-	-	-	-
	Custom NMT	0.961	0.833	0.999	0.883	1.053	0.895
	Ours	1.048	0.872	1.023	0.887	1.107	0.903

and report averages for each language-pair in Table 4. Model's ability to generate isometric translations while maintaining the translation quality (as measured by BERTScore) is evident from these empirical results. Ideally CS ratio, should be in range [0.91, 1.1]; anything between this range would be considered isometric.

We compare our model with the small data condition for En-De described in (Lakew et al., 2019). The *match* scenario described by the authors is the very similar to the isometric translations. They used a similar dataset as us to train an NMT model to exclusively generate isometric translations. Of the languages we are considering in this work, they only report their performance for En-De model on MuST-C dataset. A direct comparison of CS ratio is unfair since authors tried to generate translations as closely matching in length with the source while we generate translations that are isometric. For TQ, they report BLEU scores (Papineni et al., 2002). They report 27.60 BLEU points while we achieve 34.15 BLEU points for En-De pair.

We trained another version of our model with no hypothesis to demonstrate the importance of hypothesis in generating a good translation with interactions. Instead of passing the hypothesis to the Hypothesis Encoder, we only pass the length control token and let the concatenation occur similar as in the original model. This allows the model to generate a token-level embedding for the control token which is concatenated to the source embedding. The results are reported as *Custom NMT* in Table 4. Some recent work have attempted to do this using positional embedding to pass length control information. Authors of (Takase and Okazaki, 2019) did this for text summarization and (Lakew et al., 2019) attempted it for generating length controlled translations calling it *Length Encoding* method.

Comparing *Ours* model with other approaches, we can see our model is able to generate better quality translations than a dedicated isometric translation model while providing access to multiple other interactions as well.

5.2 User Study

To study the feasibility of the our approach, we conducted user trails with five translators. All translators were proficient in two languages; English and one additional language. We conducted the experiment with models trained similarly but with English as target language to make it easier for us to analyse the results. Two translators worked on Italian [4] as source language, two on Spanish and one on French [5].

Each translator was provided the same hand-picked sentences in the same order from FLORES dataset with hypothesis generated from OPUS-MT (Tiedemann and Thottingal, 2020). Each translator was asked to work on as many translations as possible in 30 minutes. Users were shown a source sentence in English and hypothesis in the language of their proficiency. They could then accept a translation or provide the interactions to improve the quality of the translation. Providing interactions generates a revision using the Interactive Post-Edit (IPE) model we have trained; the user can again either accept or revise the revision. After four revisions, users had an option to enter the translation manually.

We found around 59% of the hypothesis were accepted without any revisions verifying that the OPUS-MT translation model used is already of high quality. Users employed the IPE model in 13% of cases to make quick edits. For the remaining samples, users tried multiple revisions and after an average of 3.57 revisions, users preferred writing the translations manually. We saw a minor improvement in quality of translations with the IPE model as well; showing the model's capability to generate translations more efficiently without compromising the quality of the translations.

6 Conclusion

We propose and evaluate a model for human-in-loop interactive MT. The model offers the user controls that can be leveraged to correct mistranslations and rephrase them to achieve desired style variations. We specifically evaluate how the five interactions of *keep*, *delete*, *replace*, *insert*, and *length* perform in terms of translation quality as measured by BERTScore and interaction constraint satisfaction in final translation. User input remains the gold standard for ensuring translation quality, and providing user interactions enables human input when necessary to boost performance. Further, the empirical verification of the use of interactive control beyond translation corrections (as is common with existing post-editing models) to achieve desired style variations can serve as a major boost to customizability of MT systems.

As part of our future work, we wish to expand this study to more language pairs with multilingual models for reduced operational load and evaluate more style variations. We wish to go beyond explicit style tokens and use a continuous space for representing style edits on which we condition the decoder to generate a translation with corresponding variation. Such a representation is likely to be better suited to capture all style variations from data unsupervised without explicit labeling and tagging as we did with verbosity and readability.

[4] Even though we do not report results for Italian model in this work, the model used for user-study was trained as described for other languages

[5] There was one additional translator for French but we omitted the results since they did not understand the experiment and quit after couple of translations.

References

Sweta Agrawal and Marine Carpuat. 2019. Controlling text complexity in neural machine translation. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 1549–1564, Hong Kong, China. Association for Computational Linguistics.

Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. 2015. Neural machine translation by jointly learning to align and translate. In *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*.

Hanna Béchara, Raphaël Rubino, Yifan He, Yanjun Ma, and Josef van Genabith. 2012. An evaluation of statistical post-editing systems applied to RBMT and SMT systems. In *Proceedings of COLING 2012*, pages 215–230, Mumbai, India. The COLING 2012 Organizing Committee.

Roldano Cattoni, Mattia Antonino Di Gangi, Luisa Bentivogli, Matteo Negri, and Marco Turchi. 2021. Must-c: A multilingual corpus for end-to-end speech translation. *Comput. Speech Lang.*, 66:101155.

Rajen Chatterjee, Christian Federmann, Matteo Negri, and Marco Turchi. 2019. Findings of the wmt 2019 shared task on automatic post-editing. In *Proceedings of the Fourth Conference on Machine Translation (Volume 3: Shared Task Papers, Day 2)*, pages 13–30, Florence, Italy. Association for Computational Linguistics.

Rajen Chatterjee, Markus Freitag, Matteo Negri, and Marco Turchi. 2020. Findings of the WMT 2020 shared task on automatic post-editing. In *Proceedings of the Fifth Conference on Machine Translation*, pages 646–659, Online. Association for Computational Linguistics.

Rajen Chatterjee, Marion Weller, Matteo Negri, and Marco Turchi. 2015. Exploring the planet of the APEs: a comparative study of state-of-the-art methods for MT automatic post-editing. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 2: Short Papers)*, pages 156–161, Beijing, China. Association for Computational Linguistics.

Alexis Conneau and Guillaume Lample. 2019. Cross-lingual language model pretraining. In *Advances in Neural Information Processing Systems 32: Annual Conference on Neural Information Processing Systems 2019, NeurIPS 2019, December 8-14, 2019, Vancouver, BC, Canada*, pages 7057–7067.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT 2019, Minneapolis, MN, USA, June 2-7, 2019, Volume 1 (Long and Short Papers)*, pages 4171–4186.

Félix do Carmo, Dimitar Shterionov, Joss Moorkens, Joachim Wagner, Murhaf Hossari, Eric Paquin, Dag Schmidtke, Declan Groves, and Andy Way. 2021. A review of the state-of-the-art in automatic post-editing. *Mach. Transl.*, 35(2):101–143.

Marie Escribe and Ruslan Mitkov. 2021. Interactive models for post-editing. In *Proceedings of the Translation and Interpreting Technology Online Conference*, pages 167–173, Held Online. INCOMA Ltd.

Naman Goyal, Cynthia Gao, Vishrav Chaudhary, Peng-Jen Chen, Guillaume Wenzek, Da Ju, Sanjana Krishnan, Marc'Aurelio Ranzato, Francisco Guzmán, and Angela Fan. 2021. The FLORES-101 evaluation benchmark for low-resource and multilingual machine translation. *CoRR*, abs/2106.03193.

David Grangier and Michael Auli. 2018. QuickEdit: Editing text & translations by crossing words out. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 272–282, New Orleans, Louisiana. Association for Computational Linguistics.

Spence Green, Sida I. Wang, Jason Chuang, Jeffrey Heer, Sebastian Schuster, and Christopher D. Manning. 2014. Human effort and machine learnability in computer aided translation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1225–1236, Doha, Qatar. Association for Computational Linguistics.

Prabhakar Gupta, Mayank Sharma, Kartik Pitale, and Keshav Kumar. 2019. Problems with automating translation of movie/tv show subtitles. *CoRR*, abs/1909.05362.

Marcin Junczys-Dowmunt and Roman Grundkiewicz. 2016. Log-linear combinations of monolingual and bilingual neural machine translation models for automatic post-editing. In *Proceedings of the First Conference on Machine Translation: Volume 2, Shared Task Papers*, pages 751–758, Berlin, Germany. Association for Computational Linguistics.

Diederik P. Kingma and Jimmy Ba. 2015. Adam: A method for stochastic optimization. In *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*.

Philipp Koehn. 2005. Europarl: A parallel corpus for statistical machine translation. In *Proceedings of Machine Translation Summit X: Papers, MTSummit 2005, Phuket, Thailand, September 13-15, 2005*, pages 79–86.

Philipp Koehn and Rebecca Knowles. 2017. Six challenges for neural machine translation. In *Proceedings of the First Workshop on Neural Machine Translation*, pages 28–39, Vancouver. Association for Computational Linguistics.

Surafel Melaku Lakew, Marcello Federico, Yue Wang, Cuong Hoang, Yogesh Virkar, Roberto Barra-Chicote, and Robert Enyedi. 2021. Machine translation verbosity control for automatic dubbing. In *IEEE International Conference on Acoustics, Speech and Signal Processing, ICASSP 2021, Toronto, ON, Canada, June 6-11, 2021*, pages 7538–7542.

Surafel Melaku Lakew, Mattia Antonino Di Gangi, and Marcello Federico. 2019. Controlling the output length of neural machine translation. *CoRR*, abs/1910.10408.

Kelly Marchisio, Jialiang Guo, Cheng-I Lai, and Philipp Koehn. 2019. Controlling the reading level of machine translation output. In *Proceedings of Machine Translation Summit XVII Volume 1: Research Track*, pages 193–203, Dublin, Ireland. European Association for Machine Translation.

Nathan Ng, Kyra Yee, Alexei Baevski, Myle Ott, Michael Auli, and Sergey Edunov. 2019. Facebook fair's WMT19 news translation task submission. In *Proceedings of the Fourth Conference on Machine Translation, WMT 2019, Florence, Italy, August 1-2, 2019 - Volume 2: Shared Task Papers, Day 1*, pages 314–319.

Xing Niu and Marine Carpuat. 2020. Controlling neural machine translation formality with synthetic supervision. *Proceedings of the AAAI Conference on Artificial Intelligence*, 34(05):8568–8575.

Xing Niu, Sudha Rao, and Marine Carpuat. 2018. Multitask neural models for translating between styles within and across languages. In *Proceedings of the 27th International Conference on Computational Linguistics*, pages 1008–1021, Santa Fe, New Mexico, USA. Association for Computational Linguistics.

Myle Ott, Sergey Edunov, Alexei Baevski, Angela Fan, Sam Gross, Nathan Ng, David Grangier, and Michael Auli. 2019. fairseq: A fast, extensible toolkit for sequence modeling. In *Proceedings of NAACL-HLT 2019: Demonstrations*.

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. Bleu: a method for automatic evaluation of machine translation. In *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics*, pages 311–318, Philadelphia, Pennsylvania, USA. Association for Computational Linguistics.

Álvaro Peris, Miguel Domingo, and Francisco Casacuberta. 2017. Interactive neural machine translation. *Comput. Speech Lang.*, 45:201–220.

Abhishek Sharma, Prabhakar Gupta, and Anil Nelakanti. 2021. Adapting neural machine translation for automatic post-editing. In *Proceedings of the Sixth Conference on Machine Translation*, pages 315–319, Online. Association for Computational Linguistics.

Michel Simard, Cyril Goutte, and Pierre Isabelle. 2007. Statistical phrase-based post-editing. In *Human Language Technologies 2007: The Conference of the North American Chapter of the Association for Computational Linguistics; Proceedings of the Main Conference*, pages 508–515, Rochester, New York. Association for Computational Linguistics.

Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. 2014. Dropout: A simple way to prevent neural networks from overfitting. *J. Mach. Learn. Res.*, 15(1):1929–1958.

Ilya Sutskever, Oriol Vinyals, and Quoc V. Le. 2014. Sequence to sequence learning with neural networks. In *Proceedings of the 27th International Conference on Neural Information Processing Systems - Volume 2*, NIPS'14, page 3104–3112, Cambridge, MA, USA. MIT Press.

Sho Takase and Naoaki Okazaki. 2019. Positional encoding to control output sequence length. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT 2019, Minneapolis, MN, USA, June 2-7, 2019, Volume 1 (Long and Short Papers)*, pages 3999–4004.

Amirhossein Tebbifakhr, Ruchit Agrawal, Matteo Negri, and Marco Turchi. 2018. Multi-source transformer with combined losses for automatic post editing. In *Proceedings of the Third Conference on Machine Translation: Shared Task Papers*, pages 846–852, Belgium, Brussels. Association for Computational Linguistics.

Jörg Tiedemann and Santhosh Thottingal. 2020. OPUS-MT - building open translation services for the world. In *Proceedings of the 22nd Annual Conference of the European Association for Machine Translation, EAMT 2020, Lisboa, Portugal, November 3-5, 2020*, pages 479–480.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Ł ukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc.

David Wan, Chris Kedzie, Faisal Ladhak, Marine Carpuat, and Kathleen McKeown. 2020. Incorporating terminology constraints in automatic post-editing. In *Proceedings of the Fifth Conference on Machine Translation*, pages 1193–1204, Online. Association for Computational Linguistics.

Qian Wang, Jiajun Zhang, Lemao Liu, Guoping Huang, and Chengqing Zong. 2020. Touch editing: A flexible one-time interaction approach for translation. In *Proceedings of the 1st Conference of the Asia-Pacific Chapter of the Association for Computational Linguistics and the 10th International Joint Conference on Natural Language Processing*, pages 1–11, Suzhou, China. Association for Computational Linguistics.

Yue Wang, Cuong Hoang, and Marcello Federico. 2021. Towards modeling the style of translators in neural machine translation. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 1193–1199, Online. Association for Computational Linguistics.

Hayahide Yamagishi, Shin Kanouchi, Takayuki Sato, and Mamoru Komachi. 2016. Controlling the voice of a sentence in Japanese-to-English neural machine translation. In *Proceedings of the 3rd Workshop on Asian Translation (WAT2016)*, pages 203–210, Osaka, Japan. The COLING 2016 Organizing Committee.

Hao Yang, Minghan Wang, Daimeng Wei, Hengchao Shang, Jiaxin Guo, Zongyao Li, Lizhi Lei, Ying Qin, Shimin Tao, Shiliang Sun, and Yimeng Chen. 2020. HW-TSC's participation at WMT 2020 automatic post editing shared task. In *Proceedings of the Fifth Conference on Machine Translation*, pages 797–802, Online. Association for Computational Linguistics.

Tianyi Zhang, Varsha Kishore, Felix Wu, Kilian Q. Weinberger, and Yoav Artzi. 2020. Bertscore: Evaluating text generation with BERT. In *8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, April 26-30, 2020*.

Addressing Asymmetry in Multilingual Neural Machine Translation with Fuzzy Task Clustering

Qian Wang[1,2] and **Jiajun Zhang**[1,2,3]*
[1]National Laboratory of Pattern Recognition, Institute of Automation, CAS, Beijing, China
[2]School of Artificial Intelligence, University of Chinese Academy of Sciences , Beijing, China
[3]Beijing Academy of Artificial Intelligence, Beijing, China
{qian.wang, jjzhang}@nlpr.ia.ac.cn

Abstract

Multilingual neural machine translation (NMT) enables positive knowledge transfer among multiple translation tasks with a shared underlying model, but a unified multilingual model usually suffers from capacity bottleneck when tens or hundreds of languages are involved. A possible solution is to cluster languages and train individual model for each cluster. However, the existing clustering methods based on language similarity cannot handle the asymmetric problem in multilingual NMT, *i.e.*, one translation task A can benefit from another translation task B but task B will be harmed by task A. To address this problem, we propose a fuzzy task clustering method for multilingual NMT. Specifically, we employ task affinity, defined as the loss change of one translation task caused by the training of another, as the clustering criterion. Next, we cluster the translation tasks based on the task affinity, such that tasks from the same cluster can benefit each other. For each cluster, we further find out a set of auxiliary translation tasks that benefit the tasks in this cluster. In this way, the model for each cluster is trained not only on the tasks in the cluster but also on the auxiliary tasks. During training, we design a dynamic task sampling strategy that eliminate the negative influence of auxiliary tasks while exploit the positive knowledge of them. We conduct extensive experiments for one-to-many, many-to-one, and many-to-many translation scenarios to verify the effectiveness of our method.

1 Introduction

Neural machine translation (NMT) has achieved great success in recent years (Sutskever et al., 2014; Bahdanau et al., 2015; Vaswani et al., 2017). The conventional bilingual translation model well handles the translation task for a single language pair, but it is infeasible to train an individual model for each language pair since there are thousands of languages in the world. To improve the computational efficiency, researchers propose multilingual NMT that enables one model to handle multiple translation tasks (Ha et al., 2016; Johnson et al., 2017). Apart from the benefits for training and deployment, the shared underlying neural network in multilingual NMT also brings knowledge transfer among similar languages (Tan et al., 2019).

	A		B*		C*	
	en-it	en-nl	en-de	en-fr	en-zh	en-ja
Indiv	32.6	31.6	**28.1**	**38.8**	22.2	**13.6**
Joint	**34.3**	**34.3**	25.9	37.9	**25.3**	10.4

Table 1: Three kinds of task relationship. (A): Tasks benefit from each other when jointly trained. (B): Tasks are negatively influenced by each other. (C): The task en-zh benefits from en-ja but en-zh is harmful to en-ja. The models are trained on the IWSLT'17 dataset and results with * are reported in Xu and Yvon (2021).

Despite its simplicity, multilingual NMT model often suffers from representation bottleneck caused by language interference (Wang et al., 2020b; Wang and Zhang, 2022) when massive number of languages involved (Aharoni et al., 2019), which leads to performance degradation compared to bilingual translation models. Several methods have been proposed to break the bottleneck, such as designing language specific modules (Wang et al., 2019; Sachan and Neubig, 2018), hidden units (Wang et al., 2018) or routing path (Zhang et al., 2021). Among them, the most intuitive approach is grouping the languages into several clusters and training one multilingual NMT model for each cluster. These methods use similarity between language representations as clustering criterion, in which the representations come from language embedding vectors of a pretrained universal multilingual model or sparse language vectors of a multilingual knowledge base (Tan et al., 2019; Oncevay et al., 2020).

*Corresponding Author: Jiajun Zhang.

Proceedings of the 29th International Conference on Computational Linguistics, pages 5129–5141
October 12–17, 2022.

However, language similarity is not enough for modeling the relationship among translation tasks because of two reasons. First, a translation task includes a source language and a target language, while the language similarity only consider the feature of a single language. Second, the similarity metric cannot handle the asymmetric relationship between translation tasks. Zamir et al. (2018); Wang et al. (2022) show the asymmetry in multi-task learning and multilingual translation in which one task may benefit from another but not vice versa. Table 1 shows the three kinds of task relationships in multilingual NMT. Case (A) and (B) represent **symmetric** relationship in which the influence of the two tasks on each other are both positive or both negative. While in case (C), the relationship is **asymmetric** in which the task en-zh can benefit from the task en-ja but en-ja is harmed by en-zh when the two tasks are jointly trained.

In this work, we propose a fuzzy task clustering method for multilingual NMT to address the asymmetric problem among translation tasks. We use the task affinity (Fifty et al., 2021) as the clustering criterion, which measures the loss change of one task caused by the training of another task. We first train a universal multilingual model to obtain the affinity between each two tasks. Then we cluster tasks based on the affinity score such that tasks within each cluster can symmetrically benefit each other. For each cluster, we also select a set of auxiliary tasks that asymmetrically benefit the tasks in this cluster. Finally, we build separate models for each cluster, and train the model on both the tasks in the cluster and the corresponding auxiliary tasks. In this way, the model for each cluster can exploit the auxiliary tasks to facilitate training and improve the model quality. Compared to previous language clustering based methods, the proposed method improves positive knowledge transfer between tasks by introducing auxiliary tasks. To further address the increased burden of model training due to auxiliary tasks involved, we therefore propose a dynamic data sampling strategy, in which the sampling weights of auxiliary tasks gradually decrease to allow the optimization process to focus more on the tasks that used in inference.

To summarize, our method has the following advantages:

- The clustering criterion in our method directly measures the effect on one translation task when training another, which can better model the relationship between tasks compared to language similarity.
- The fuzzy clustering paradigm introduces auxiliary tasks for each cluster, breaking the barrier between clusters and improving positive knowledge transfer.

2 Problem Formulation

2.1 General Formulation

Given a set of translation tasks $\mathcal{T} = \{\tau_1, \ldots, \tau_N\}$, we group the tasks into $K \in \{k \in \mathbb{N}^+ | k \leq N\}$ clusters $\mathcal{C}$:

$$\mathcal{C} = \left\{ c_1, \ldots, c_K \;\middle|\; \begin{array}{l} c_k \neq \emptyset, \forall k \\ \bigcup_{k=1}^{K} c_k = \mathcal{T} \\ c_i \cap c_j = \emptyset, \forall i \neq j \end{array} \right\} \quad (1)$$

For each cluster c_k, we find out a set of auxiliary tasks from other clusters that benefit the tasks in c_k. By combining the auxiliary tasks and tasks in c_k, we obtain another cluster g_k. The set of g_k is denoted as:

$$\mathcal{G} = \left\{ g_1, \ldots, g_K \;\middle|\; \begin{array}{l} \bigcup_{k=1}^{K} g_k = \mathcal{T} \\ c_k \subseteq g_k, \forall k \end{array} \right\} \quad (2)$$

where the tasks $\tau \in g_k \setminus c_k$ are the auxiliary tasks of c_k.

We then build a multilingual NMT model M_k for each cluster c_k to handle the translation tasks in c_k. The model M_k is trained on g_k, which consists of both the tasks from c_k and the corresponding auxiliary tasks, by minimizing the cross-entropy loss:

$$\arg\min_{M_k} \sum_{\tau \in g_k} \sum_{(\mathbf{x},\mathbf{y}) \in \tau} -\log p(\mathbf{y}|\mathbf{x}, M_k) \quad (3)$$

where $(\mathbf{x}, \mathbf{y})$ is a sentence pair from task τ.

For a predefined quality measure $Q(\tau|M)$ of a model M on task τ, our goal is to find the optimal clusters $\mathcal{C}$ and $\mathcal{G}$ that maximize the inference performance of all tasks:

$$\arg\max_{\mathcal{C},\mathcal{G},K} \sum_{k=1}^{K} \sum_{\tau \in c_k} Q(\tau|M_k) \quad (4)$$

2.2 Specialized Cases

We can see in this section that bilingual, multilingual, and clustering based methods are special cases of our formulation.

Bilingual Translation When $K = N$, namely we assign only one translation task for each cluster and train individual model for each task, the problem degrades into bilingual translation. The bilingual model for each cluster is trained with Equation (3).

Multilingual Translation When $K = 1$, a universal multilingual NMT model is trained for all tasks as in Johnson et al. (2017). To distinguish different languages, a special token called language tag is appended at the beginning of each source sentence.

Language Clustering for Multilingual NMT When $1 < K < N, \mathcal{C} = \mathcal{G}$, the tasks are grouped into separate clusters without auxiliary tasks. For example, Tan et al. (2019) first train a universal multilingual NMT model to obtain language embedding vectors. Based on the the embedding vectors, they cluster the languages into different groups and train individual model for each cluster.

3 Method

In this section, we first describe the translation task affinity used for modeling the asymmetric relationship between tasks (Section 3.1), followed by the translation task clustering method that clusters the tasks and finds out the auxiliary tasks for each cluster (Section 3.2). We finally build a model for each cluster and train the model with dynamic data sampling strategy (Section 3.3).

3.1 Translation Task Affinity

We first focus on the quality measure $Q(\tau|M)$ of a model M on task τ. A popular choice for modeling and optimizing the quality of a translation model is cross-entropy loss (Bahdanau et al., 2015; Vaswani et al., 2017). A lower cross-entropy indicates a better model. Therefore, we use the negative cross-entropy, namely the log-probability, to measure the quality of M on task τ:

$$Q(\tau|M) = \sum_{(\mathbf{x},\mathbf{y})\in\tau} \log p(\mathbf{y}|\mathbf{x}, M) \tag{5}$$

Obviously, training one model M for each task τ is a possible solution for Equation (4), since the training objective in Equation (3) is identical to maximizing $Q(\tau|M)$.

Previous works find out that the performance on some tasks can be improved when jointly trained (Tan et al., 2019; Standley et al., 2020; Chiang et al., 2022). Therefore, we want to find out which task can benefit another under the quality measure $Q(\tau|M)$ in multilingual training. Formally, given two tasks τ_i and τ_j, the affinity $Z_{j\to i}$ from τ_j to τ_i is defined by:

$$Z_{j\to i} = Q(\tau_i|M_1) - Q(\tau_i|M_2) \tag{6}$$

where M_1 is trained on tasks τ_i and τ_j, while M_2 is trained on task τ_i. A positive value $Z_{j\to i} > 0$ indicates that the task τ_j brings positive knowledge transfer to task τ_i, while a negative value indicates that the task τ_j is harmful to task τ_i. Note that $Z_{j\to i} \neq Z_{i\to j}$, and thus we call it **asymmetric task affinity**.

However, exhaustively searching over all possible combinations of $|\mathcal{T}|$ tasks requires training $2^{|\mathcal{T}|} - 1$ models, which is unaffordable. We therefore turn to an efficient approximation that measures, at each training step, the change of one task's loss caused by the optimization of another task (Fifty et al., 2021). The affinity from task j to task i at step t is defined as[1]:

$$\hat{Z}^t_{j\to i} = \log p(\mathbf{y}|\mathbf{x}, M^{t+1}) - \log p(\mathbf{y}|\mathbf{x}, M^t) \tag{7}$$

where $(\mathbf{x}, \mathbf{y})$ is a sentence pair sampled from task τ_i, and M^{t+1} is obtained by one-step training on task τ_j of model M^t.

Similar to Tan et al. (2019), we first train a universal model for all the language pairs. During training, we calculate the affinity $Z^t_{j\to i}$ with randomly and uniformly sampled tasks τ_i and τ_j at t-th step[2]. The affinity is then accumulated across all steps:

$$Z_{j\to i} \approx \hat{Z}_{j\to i} = \sum_{t=1}^{T} \hat{Z}^t_{j\to i} \tag{8}$$

With the affinity matrix $Z \in R^{|\mathcal{T}|\times|\mathcal{T}|}$, our goal in Equation (4) becomes maximizing the overall inter-task affinity as proved in Section A:

$$\arg\max_{\mathcal{C},\mathcal{G},K} \sum_{k=1}^{K} \sum_{\tau_i\in c_k} \sum_{\tau_j\in g_k} Z_{j\to i} \tag{9}$$

[1] In Fifty et al. (2021), they consider the ratio of the loss change to eliminate the scale discrepancy among different task losses: $Z^t_{j\to i} = 1 - \frac{L_j(\mathbf{x},\mathbf{y},M^{t+1})}{L_j(\mathbf{x},\mathbf{y},M^t)}$. Since the task loss scales are similar in multilingual translation, we only consider the absolute loss change.

[2] In multilingual NMT, a single batch can contain sentence pairs from multiple tasks (Johnson et al., 2017), which is unsuitable for Equation (7). We therefore iteratively use batches contain multiple tasks to stabilize training, and batches contain only one task to calculate $Z^t_{j\to i}$.

3.2 Translation Task Clustering

We now describe how to solve Equation (9). Similar to *reduction from Set-Cover* that chooses a subset covering all the tasks to minimize the overall cost (Standley et al., 2020), this problem is NP-hard in general. We therefore propose a greedy approximation that iteratively constructs the clusters $\mathcal{C}$ and $\mathcal{G}$ to minimize the overall affinity.

As shown in Figure 1(b), we first initialize $\mathcal{C}$ with N clusters, where each cluster contains only one task. With the affinity matrix Z, we build the corresponding N clusters of $\mathcal{G}$:

$$g_k = c_k \cup \{\tau_j | Z_{j\to i} > 0\} \quad \forall k = 1, \ldots, N \tag{10}$$

where $\tau_i \in c_k$.

We define the score function $S(c_k, g_k)$ to measure the cluster affinity of c_k and corresponding g_k:

$$S(c_k, g_k) = \frac{1}{|g_k|} \sum_{\tau_i \in c_k} \sum_{\tau_j \in g_k} Z_{j\to i} \tag{11}$$

where $\frac{1}{|g_k|}$ is the normalization term for the number of tasks. We also define the rule of the union of two clusters c_1 and c_2 in $\mathcal{C}$, and g_1 and g_2 in $\mathcal{G}$:

$$\begin{aligned} c_1 \underset{\mathcal{C}}{\cup} c_2 =& \{\tau | \tau \in c_1 \text{ or } \tau \in c_2\} \\ g_1 \underset{\mathcal{G}}{\cup} g_2 =& (c_1 \underset{\mathcal{C}}{\cup} c_2) \cup \\ & \left\{ \tau_j \;\middle|\; \begin{array}{l} \tau_j \in g_1 \text{ or } \tau_j \in g_2 \\ Z_{j\to i} > 0, \forall \tau_i \in c_1 \underset{\mathcal{C}}{\cup} c_2 \end{array} \right\} \end{aligned} \tag{12}$$

As shown in Figure 1(c-e), for each two clusters, we form a union of them based on the above union rule and calculate the cluster affinity. The clusters with maximum overall affinity are selected. The clustering step is repeated until no union of two clusters can further improve the overall affinity, *i.e.*, for any of two clusters c_1 and c_2, we have:

$$S(c_1 \underset{\mathcal{C}}{\cup} c_2, g_1 \underset{\mathcal{G}}{\cup} g_2) \leq S(c_1, g_1) + S(c_2, g_2) \tag{13}$$

After iteration[3], the Equation (4) is solved and we obtain K clusters in $\mathcal{C}$ as well as K corresponding clusters in $\mathcal{G}$. We train one multilingual NMT model (Johnson et al., 2017) for the tasks in each cluster, where each model M_k is trained with tasks in g_k and is used for inference of tasks in c_k.

[3]In most of our experiments, the iteration can stop with a proper number of resulting K clusters. However, there is no theoretical guarantee for convergence and it may degrade into bilingual ($K = N$) or universal ($K = 1$) models. In practise, early-stop can be adopted when K satisfies a predefined constraints (Standley et al., 2020).

(a) Affinity Matrix Z

	τ_1	τ_2	τ_3
τ_1	1.1	0.2	0.2
τ_2	0.3	0.9	-0.2
τ_3	-0.2	0.1	1.2

(b) Initialize $\mathcal{C}$ and $\mathcal{G}$

k	$\mathcal{C}$	$\mathcal{G}$	$S(c,g)$
1	τ_1	τ_1, τ_2	0.7
2	τ_2	τ_1, τ_2, τ_3	0.4
3	τ_3	τ_1, τ_3	0.7

(c) Union clusters 1 and 2

k	$\mathcal{C}$	$\mathcal{G}$	$S(c,g)$
1	τ_1, τ_2	τ_1, τ_2	1.25
2	τ_3	τ_1, τ_3	0.7

(d) Union clusters 1 and 3

k	$\mathcal{C}$	$\mathcal{G}$	$S(c,g)$
1	τ_1, τ_3	τ_1, τ_3	1.15
2	τ_2	τ_1, τ_2, τ_3	0.4

(e) Union clusters 2 and 3

k	$\mathcal{C}$	$\mathcal{G}$	$S(c,g)$
1	τ_1	τ_1, τ_2	0.7
2	τ_2, τ_3	τ_1, τ_2, τ_3	0.8

Figure 1: An example of clustering 3 tasks. Based on the affinity matrix (a), we first initialize the clusters $\mathcal{C}$ and $\mathcal{G}$ (b), and then make union of each of two clusters (c-e). In this example, (c) brings highest overall affinity (1.95) and no union of two clusters can further improve the affinity (1.2 if the three tasks belong to one cluster). Thus the clusters in (c) is the clustering result.

3.3 Training with Dynamic Data Sampling

We now focus on the training of model M_k. The training objective, as described in Equation (3), is to minimize the overall cross-entropy loss for tasks in g_k. Since the training tasks in g_k contain both the inference tasks in c_k and other auxiliary tasks $g_k \setminus c_k$, the objective brings positive knowledge transfer and can improve the translation quality compared to that only trained with tasks in c_k. However, it also introduces extra burden for the model due to more auxiliary tasks involved.

We therefore propose a **dynamic data sampling** strategy that gradually decreases the weights of auxiliary tasks and focuses more on inference tasks. We start from the temperature based sampling (Conneau et al., 2020) that samples training data from each task $\tau \in g_k$ according to the data size of each task $|D_\tau|$ and a temperature term ρ:

$$\hat{P}_k(\tau) = \frac{q_\tau^{1/\rho}}{\sum_{\pi \in g_k} q_\pi^{1/\rho}} \text{ where } q_\tau = \frac{|D_\tau|}{\sum_{\pi \in g_k} |D_\pi|} \tag{14}$$

We re-scale the sampling distribution $P(\tau)$ for the auxiliary tasks $\tau \in g_k \setminus c_k$ as a function of the training epoch E:

$$P_k(\tau) = \begin{cases} \frac{1}{1+\lambda E} \frac{|c_k|}{|g_k|} \hat{P}_k(\tau) & \text{if } \tau \notin c_k \\ \hat{P}_k(\tau) & \text{if } \tau \in c_k \end{cases} \tag{15}$$

where λ is the decay rate of the sampling weight. The term $\frac{|c_k|}{|g_k|}$ considers the number of auxiliary tasks. If there are too many auxiliary tasks for

training the model, namely $|g_k| \gg |c_k|$, the weight of each auxiliary task should be smaller. Other dynamic data sampling and task weighting methods can also be used for prioritizing the inference tasks in c_k (Lin et al., 2019; Wang et al., 2020a; Mahapatra and Rajan, 2020).

4 Experiment Setup

4.1 Dataset

We evaluate our method on the TED 2020 Parallel Sentences Corpus collected by Reimers and Gurevych (2020). The dataset contains sentences from TED talks with their translations in more than 100 languages provided by a global community of volunteers.

For one-to-many (O2M) translation and many-to-one (M2O) translation scenarios, we select 28 languages (es, fr, ar, zh, ko, ru, tr, it, ja, he, pt, ro, vi, nl, hu, fa, pl, de, el, sr, bg, uk, hr, cs, id, th, sv, sk) $\leftrightarrow$ English that contain more than 100K sentence pairs[4] and the data statistics are shown in Supplementary Materials (Section B). We randomly select $4,000$ sentence pairs as validation set and $4,000$ sentence pairs as test set for each language pair. We use byte-pair encoding (BPE) (Sennrich et al., 2016) to encode all sentences and learn the BPE operation using sentencepiece (Kudo and Richardson, 2018), which results in a shared subword vocabulary containing 32K sub-word symbols.

For many-to-many (M2M) translation, we select 10 languages (es, fr, ar, ko, ru, tr, it, ja, he, pt) from the TED 2020 Parallel Sentences Corpus. Each two languages contains about 300K parallel sentences, which results in 90 translation directions. We preprocess the data in the same way as in O2M and M2O setting.

The multilingual datasets are sampled with temperature based strategy with $\rho = 5$, and the sampling weight decay rate in our method is set to $\lambda = 0.5$.

4.2 Model Settings

We conduct the experiments using the Transformer model (Vaswani et al., 2017) and implement our method based on the `fairseq` codebase[5]. For the O2M and M2O experiments, we use a 4-layer model[6] with embedding size 512 and FFN layer dimension 1024. For the M2M experiment, we adopt the `transformer_iwslt_de_en` configuration which represents 6-layer model. Each mini-batch contains roughly $16,384$ tokens. All models are trained with Adam optimizer (Kingma and Ba, 2015) on Nvidia 3090 GPUs. We use 2 GPUs for O2M and M2O translation, and 4 GPUs for M2M translation. We use SacreBLEU to measure the translation quality (Papineni et al., 2002; Post, 2018) and test the statistical significance by bootstrap resampling (Koehn, 2004).

4.3 Systems

The following methods are used in our experiments.

Bilingual (Bi.) We train a Transformer model for each task, which results in N individual models for N translation directions (Vaswani et al., 2017) .

Multilingual (Multi.) We train a universal multilingual Transformer model for all translation directions (Johnson et al., 2017).

Language Embedding Clustering (LEC) We cluster the languages based on the language embedding and train one multilingual model for each cluster. The number of clusters K is obtained with elbow method following Tan et al. (2019).

Hard Task Affinity Clustering (HTAC) We use the affinity instead of language embedding for clustering and no auxiliary tasks are used for training, namely we set $\lambda = +\infty$ in Equation (15), which is equivalent to $\mathcal{G} = \mathcal{C}$.

Fuzzy Task Affinity Clustering (FTAC) We use the affinity for clustering and incorporate auxiliary tasks with dynamic data sampling during training.

5 Results and Analyses

5.1 Clustering Results

For the **LEC** method (Tan et al., 2019), we obtain 9 clusters in O2M translation: {{sk, cs, hr, sr, pl}, {sv, uk}, {th, el, hu, vi, he, ar}, {id}, {bg, ro}, {de, fa, nl, pt, it, tr, fr, es}, {ru}, {zh}, {ja, ko}}. In M2O translation, there are 11 clusters: {{sk, cs}, {sv}, {th}, {id, vi}, {hr, bg, sr}, {uk, pl, ru}, {el}, {de, hu, nl, tr}, {fa, he, ar}, {ro, pt, it, fr, es}, {ja, ko, zh}}.

[4] We remove Brazilian Portuguese (pt-br) and Traditional Chinese (zh-tw) since they are similar to Portuguese (pt) and Simplified Chinese (zh-cn). we use zh to indicate zh-cn for simplicity.

[5] https://github.com/pytorch/fairseq

[6] We use 4-layer model because some bilingual models cannot converge with 6-layer setting.

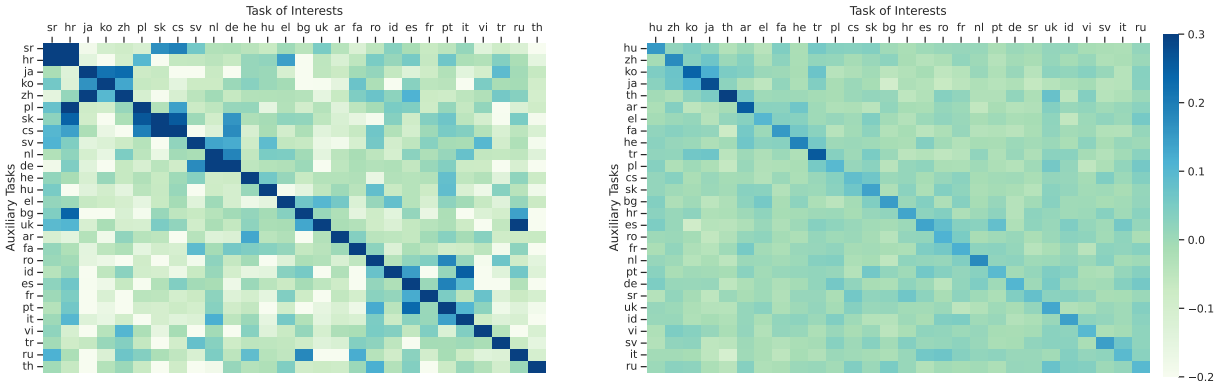

Figure 2: The task affinity between English to 28 languages (O2M, Left) and 28 languages to English (M2O, Right). Each cell (i, j) represents the affinity $Z_{i\to j}$ of task τ_i to task τ_j.

k	c_k (en→X)	$g_k \setminus c_k$ (en→X)
1	sr, hr	fr, uk, bg, sv, el, pl, cs, es, it, sk
2	ja, zh, ko	tr, vi, th
3	pl, sk, cs	hu, ru, he, fr
4	sv, nl, de	zh, id, sk, hr
5	he, hu	sv, ja, sk, de, ar
6	el, bg, uk	sk, es, fa, sr, th
7	ro, id, es, fr, pt, it	sv, cs, de, hr, sr, pl
8	ar, fa	hu, vi, de, nl
9	tr, vi	ja, zh, ar, th, ko
10	th, ru	ja, zh, el, ar, he, ko

Table 2: The clustering results of our method for one-to-many translation. Each language X denotes a translation task en→X.

k	c_k (X→en)	$g_k \setminus c_k$ (X→en)
1	hu	sk, ar, ro, pt, pl, hr, sv, ru, th, uk, fa, id, zh, it, fr, de, ja, ko, cs, he
2	zh, ko, ja, th	ar, uk, de, hu
3	ar, el, fa, he	ru, cs, pl, ro, es, hr, uk, sr, ja, id, bg
4	tr, pl, cs, sk	ru, sr, ko, ja
5	bg, hr	sk, ru, cs, pl, fa, nl, el, id
6	es, ro, fr, nl, pt	sk, id, bg
7	de, sr, uk	sk, sv, ru, cs, pl, ro, ar, hr, fa, nl, ko, it, zh, el, id, he
8	id, vi	th, fr, pl, hr, uk, de, fa, sr, ko, it, ja, zh, tr, he
9	sv, it, ru	sk, fr, pl, uk, nl, ja, pt, he

Table 3: The clustering results of our method for many-to-one translation. Each language X denotes a translation task X→-en.

For our task affinity based method (**FTAC**), we first show the task affinity in one-to-many and many-to-one scenarios in Figure 2. Obviously, the left part (O2M) and the right part (M2O) share one common pattern that the cells on the diagonal line are darker, which indicates that each task contributes more on itself. Besides, there are more differences between O2M and M2O models which leads to several interesting findings:

- We find that the cells on the diagonal line in O2M are darker than M2O models, and the cells beyond diagonal line are lighter in O2M model. The observation indicates that a task in O2M model relies more on itself and the positive knowledge transfer is more common in M2O multilingual model. This may be because the burden of generation is mainly on the decoder (Dabre et al., 2020), and the tasks in a M2O model share one target language. Thus the tasks in M2O translation can benefit each other and reduce the generation burden.
- The affinity between tasks better correlates with linguistic similarity in O2M model. For example, the Serbian (sr) and Croatian (hr) are similar languages and the affinity between en→sr and en→hr is quite high. Similar to language embedding (Tan et al., 2019), the affinity also captures the regional, cultural, and historical influences in O2M model (see the affinities between ja, ko, and zh).
- The affinities in M2O model cannot reflect linguistic proximities. As claimed above, the positive knowledge transfer mainly occurs in the decoder to facilitate generation. The differences in the target side between M2O translation tasks are imperceptible and affected by factors like data sizes, training schedule or optimization.

Based on the affinity matrix, we cluster the languages using the method described in Section 3.2 and show the clustering results for O2M multilingual translation in Table 2 and M2O translation in

	One-to-Many							Many-to-One						
	Bi.	**Multi.**	**LEC**	**HTAC**	**FTAC**	Δ_1	Δ_2	**Bi.**	**Multi.**	**LEC**	**HTAC**	**FTAC**	Δ_1	Δ_2
en↔es	40.5	38.9	41.3	42.0	**42.5**	+1.2	+3.6†	43.6	46.1	46.2	46.4	**48.4**	+2.2	+2.3†
en↔fr	41.4	39.6	42.2	42.8	**43.3**	+1.1	+3.7†	37.9	40.3	40.0	40.5	**42.1**	+2.1	+1.8†
en↔ar	16.3	14.5	17.2	17.2	**17.7**	+0.5	+3.2†	32.9	34.6	36.3	36.5	**37.9**	+1.6	+3.3†
en↔zh	16.6	14.6	16.6	16.5	**17.0**	+0.4	+2.4†	22.5	25.4	25.2	25.8	**26.6**	+1.4	+1.2†
en↔ko	6.5	6.4	7.6	7.4	**7.9**	+0.3	+1.5†	19.7	21.7	22.6	23.2	**23.7**	+1.1	+2.0†
en↔ru	20.9	18.4	20.9	20.9	**21.2**	+0.3	+2.8†	25.4	28.8	28.4	28.8	**28.9**	+0.5	+0.1
en↔tr	17.0	15.9	17.5	18.6	**19.1**	+1.6	+3.2†	27.1	28.0	28.7	29.5	**30.5**	+1.8	+2.5†
en↔it	33.4	32.7	35.0	35.8	**35.9**	+0.9	+3.2†	37.5	40.8	40.8	41.0	**41.3**	+0.5	+0.5‡
en↔ja	14.5	14.0	17.0	17.0	**17.5**	+0.5	+3.5†	15.3	17.4	18.3	19.0	**19.6**	+1.3	+2.2†
en↔he	26.9	24.4	28.1	28.6	**29.1**	+1.0	+4.7†	38.2	38.9	40.5	40.7	**42.9**	+2.4	+4.0†
en↔pt	37.6	35.9	38.5	39.4	**39.9**	+1.4	+4.0†	41.1	43.5	44.0	44.3	**45.9**	+1.9	+2.4†
en↔ro	27.7	26.6	28.5	29.5	**30.0**	+1.5	+3.4†	37.2	40.1	39.6	39.9	**41.9**	+2.3	+1.8†
en↔vi	29.0	27.5	30.7	30.8	**31.2**	+0.5	+3.7†	29.6	32.0	32.2	32.2	**32.7**	+0.5	+0.7†
en↔nl	31.7	29.8	32.7	32.9	**33.4**	+0.7	+3.6†	36.8	38.5	38.6	38.8	**40.1**	+1.5	+1.6†
en↔hu	17.8	14.8	17.0	17.8	**18.3**	+1.3	+3.5†	27.1	27.3	27.6	27.1	**28.7**	+1.1	+1.4†
en↔fa	18.2	17.7	19.5	19.6	**20.1**	+0.6	+2.4†	29.8	31.9	32.1	33.0	**34.5**	+2.4	+2.6†
en↔pl	15.6	15.3	18.2	18.4	**18.9**	+0.7	+3.6†	23.2	27.0	26.8	27.3	**28.1**	+1.3	+1.1†
en↔de	26.6	24.9	27.3	28.3	**28.8**	+1.5	+3.9†	33.8	35.7	36.1	35.8	**37.4**	+1.3	+1.7†
en↔el	32.0	28.8	33.9	33.7	**34.2**	+0.3	+5.4†	39.6	41.0	39.6	41.5	**42.5**	+2.9	+1.5†
en↔sr	22.0	21.9	25.5	24.3	**24.8**	-0.7	+2.9†	37.6	40.2	41.9	**42.0**	41.7	-0.2	+1.5†
en↔bg	32.6	30.8	34.6	34.6	**35.1**	+0.5	+4.3†	39.2	42.5	42.7	**42.8**	42.7	+0.0	+0.2
en↔uk	18.9	18.3	21.5	21.1	**21.6**	+0.1	+3.3†	27.4	31.2	30.9	31.4	**31.5**	+0.6	+0.3
en↔hr	27.4	26.0	**30.6**	29.5	30.3	-0.3	+4.3†	38.0	42.7	**44.1**	43.6	42.8	-1.3	+0.1
en↔cs	20.6	19.0	23.1	23.3	**23.8**	+0.7	+4.8†	31.1	34.5	33.9	36.3	**36.7**	+2.8	+2.2†
en↔id	31.5	27.9	31.5	31.6	**32.6**	+1.1	+4.7†	31.7	33.3	34.5	34.6	**35.4**	+0.9	+2.1†
en↔th	17.8	16.3	18.5	19.3	**19.8**	+1.3	+3.5†	24.9	27.1	24.9	28.1	**29.1**	+4.2	+2.0†
en↔sv	34.9	30.7	33.6	36.5	**37.2**	+3.6	+6.5†	40.4	42.0	40.4	43.4	**44.4**	+4.0	+2.4†
en↔sk	20.6	19.5	25.2	24.3	**24.8**	-0.4	+5.3†	29.9	34.9	36.1	37.4	**38.1**	+2.0	+3.2†
Average	24.9	23.3	26.2	26.5	**27.0**	+0.8	+3.7	32.1	34.6	34.8	35.4	**36.3**	+1.6	+1.7

Table 4: BLEU scores of one-to-many (O2M, Left) and many-to-one (M2O, Right) translation with different methods. The column Δ_1 and Δ_2 are the improvements of **FTAC** compared to **LEC** and **Multi.** respectively. † and ‡ indicate the corresponding improvement is statistically significant with $p < 0.01$ and 0.05 respectively.

Table 3. We find that the clusters in M2O translation contain more auxiliary tasks, which proves that the M2O translation can benefit more from multilingual training.

For many-to-many translation, we show the affinity heatmap and clustering results in Supplementary Materials (Section C).

5.2 O2M and M2O Translation Quality

We present the translation quality of different methods in Table 4. By comparing different methods, several observations can be found.

- Our **FTAC** method can well address the asymmetry in multilingual NMT and outperforms the **Multi.** baseline in all translation tasks.
- Clustering methods perform better in O2M translation. The **LEC** method outperforms the multilingual baseline by 2.9 BLEU in O2M translation but only 0.2 BLEU in M2O translation. Similarly, the **HTAC** method achieves improvements of 3.2 BLEU and 0.8 BLEU in O2M and M2O respectively. The results prove that the burden of generation caused by number of target languages involved is important in O2M translation, and the problem can be well addressed by clustering the tasks into different groups.
- Clustering based on task affinity performs better than based on language embedding. Compared to **LEC** method, **HTAC** achieves improvements of 0.3 BLEU in O2M setting and 0.6 BLEU in M2O setting.
- The asymmetry feature of task affinity that leads to fuzzy clustering is important, and the auxiliary tasks in each cluster brings large improvements, especially for M2O setting. The **FTAC** method with auxiliary tasks outperforms the **HTAC** method without auxiliary tasks by 0.5 BLEU and 0.9 BLEU in O2M and M2O respectively.

	→es	→fr	→ar	→ko	→ru	→tr	→it	→ja	→he	→pt
es→	-	35.9/+3.9	12.2/+3.8	16.7/+3.9	16.9/+3.9	15.0/+4.4	30.3/+4.0	15.2/+5.3	18.8/+6.2	31.3/+6.3
fr→	28.9/+4.6	-	11.4/+2.6	16.6/+3.4	16.3/+4.5	14.0/+4.1	28.3/+4.1	14.7/+5.3	18.3/+5.3	28.8/+5.2
ar→	23.8/+2.4	27.8/+4.0	-	15.2/+2.4	14.0/+3.3	12.3/+3.2	21.8/+3.8	13.3/+2.3	15.5/+4.5	22.2/+5.4
ko→	15.5/+2.4	22.4/+2.3	6.8/+1.6	-	10.4/+2.0	9.8/+2.1	15.6/+2.5	14.4/+3.0	10.0/+2.2	14.8/+2.8
ru→	20.3/+3.5	27.1/+3.4	8.2/+1.9	14.9/+1.9	-	11.2/+2.3	20.0/+3.1	13.3/+3.8	13.2/+3.6	19.5/+4.9
tr→	21.0/+3.4	26.9/+3.0	9.4/+2.2	15.4/+2.8	12.6/+2.9	-	19.8/+3.6	14.0/+3.6	13.9/+3.0	20.4/+4.2
it→	30.2/+3.6	34.9/+3.3	11.3/+3.5	16.7/+3.8	16.4/+4.4	14.1/+2.3	-	14.8/+5.0	18.2/+5.1	29.1/+6.0
ja→	13.1/+1.5	19.8/+2.2	5.7/+1.3	13.4/+1.8	8.6/+0.3	8.6/+1.2	13.6/+1.8	-	7.8/+1.9	12.9/+2.5
he→	26.8/+4.8	31.9/+5.6	11.8/+3.2	16.4/+2.2	16.4/+3.8	13.5/+2.6	24.7/+4.3	14.5/+2.6	-	26.1/+6.1
pt→	32.5/+4.8	36.3/+3.7	11.9/+4.4	17.1/+4.6	16.4/+4.9	14.2/+5.3	30.2/+4.4	15.4/+5.3	18.7/+6.1	-

Table 5: Many-to-many (M2M) translation quality measured by BLEU score. We compare our method (**FTAC**) with the multilingual baseline (**Multi.**). The BLEU scores of **FTAC** and the improvements Δ are reported with the format of **FTAC**/Δ.

5.3 M2M Translation Quality

For many-to-many setting, the language embedding based clustering methods are not suitable since the source language and target language in a translation task may be clustered into different groups. On the other hand, our method well handles M2M multilingual translation by directly clustering different tasks based on the affinity between them.

We compare our method (**FTAC**) with the multilingual baseline (**Multi.**) and the results are shown in Table 5. Our method consistently outperforms the baseline method by up to $+6.3$ BLEU and $+3.6$ BLEU on average. The results in M2M translation show that our method can well handle M2M translation.

5.4 The Effects of Auxiliary Tasks

To further understand the performance gain contributed by auxiliary tasks, we analyze the effects of different sampling strategies for auxiliary tasks. Besides the **HTAC** and **FTAC** methods, we also compare the following strategies:

- **Rand.** We randomly select $|g_k \setminus c_k|$ auxiliary tasks for each cluster c_k.
- **Temp.** We use vanilla temperature based sampling for all tasks by setting $P_k(\tau) = \hat{P}_k(\tau)$ in Equation (15).
- **Fix.** We set $\lambda = 0$ in Equation (15), namely the sampling weights only correlate with the number of auxiliary tasks involved.

Table 6 shows the average BLEU of different methods in one-to-many translation. The detailed results are shown in Supplementary Materials (Section D). By comparing **HTAC** and **Rand.**, we find that arbitrarily adding auxiliary tasks significantly hurts the performance by 1.9 BLEU, which proves the effectiveness of our **FTAC** method for selecting proper auxiliary tasks. Besides, we find that the **Temp.** and **Fix.** methods also perform worse than **HTAC**, which indicates that the auxiliary tasks bring extra burden for generation although they are correlated with the inference tasks. By gradually decreasing the weights of auxiliary tasks during training, the **FTAC** method can focus more on the inference tasks and bring better results.

Method	**HTAC**	**Rand.**	**Temp.**	**Fix.**	**FTAC**
Result	35.4	33.9	35.0	35.1	36.3

Table 6: Results of different sampling strategies for auxiliary task data in one-to-many translation (measured by average BLEU).

6 Conclusion

In this work, we have proposed a fuzzy task clustering method to address the asymmetric problem in multilingual NMT based on task affinity. The task affinity is defined by the loss change of one task after a step training of another task. Based on the affinity, we cluster translation tasks, each of which contains tasks that are symmetric. Each cluster is further equipped with auxiliary tasks that can benefit the model training of this cluster. Experiments show that our fuzzy task clustering method significant outperforms the strong baselines. We also show the effectiveness of incorporating auxiliary tasks in a multilingual translation model. In the future, we plan to explore more efficient and effective clustering criterion by exploiting large-scale pre-trained multilingual models.

Acknowledgements

This research work is supported by National Key R&D Program of China (2020AAA0105200) and the Natural Science Foundation of China under Grant 62122088.

References

Roee Aharoni, Melvin Johnson, and Orhan Firat. 2019. Massively multilingual neural machine translation. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT 2019, Minneapolis, MN, USA, June 2-7, 2019, Volume 1 (Long and Short Papers)*, pages 3874–3884. Association for Computational Linguistics.

Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. 2015. Neural machine translation by jointly learning to align and translate. In *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*.

Ting-Rui Chiang, Yi-Pei Chen, Yi-Ting Yeh, and Graham Neubig. 2022. Breaking down multilingual machine translation. In *Findings of the Association for Computational Linguistics: ACL 2022, Dublin, Ireland, May 22-27, 2022*, pages 2766–2780. Association for Computational Linguistics.

Alexis Conneau, Kartikay Khandelwal, Naman Goyal, Vishrav Chaudhary, Guillaume Wenzek, Francisco Guzmán, Edouard Grave, Myle Ott, Luke Zettlemoyer, and Veselin Stoyanov. 2020. Unsupervised cross-lingual representation learning at scale. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, ACL 2020, Online, July 5-10, 2020*, pages 8440–8451. Association for Computational Linguistics.

Raj Dabre, Chenhui Chu, and Anoop Kunchukuttan. 2020. A survey of multilingual neural machine translation. *ACM Comput. Surv.*, 53(5):99:1–99:38.

Chris Fifty, Ehsan Amid, Zhe Zhao, Tianhe Yu, Rohan Anil, and Chelsea Finn. 2021. Efficiently identifying task groupings for multi-task learning. In *Advances in Neural Information Processing Systems 34: Annual Conference on Neural Information Processing Systems 2021, NeurIPS 2021, December 6-14, 2021, virtual*, pages 27503–27516.

Thanh-Le Ha, Jan Niehues, and Alexander H. Waibel. 2016. Toward multilingual neural machine translation with universal encoder and decoder. *CoRR*, abs/1611.04798.

Melvin Johnson, Mike Schuster, Quoc V. Le, Maxim Krikun, Yonghui Wu, Zhifeng Chen, Nikhil Thorat, Fernanda B. Viégas, Martin Wattenberg, Greg Corrado, Macduff Hughes, and Jeffrey Dean. 2017. Google's multilingual neural machine translation system: Enabling zero-shot translation. *Trans. Assoc. Comput. Linguistics*, 5:339–351.

Diederik P. Kingma and Jimmy Ba. 2015. Adam: A method for stochastic optimization. In *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*.

Philipp Koehn. 2004. Statistical significance tests for machine translation evaluation. In *Proceedings of the 2004 Conference on Empirical Methods in Natural Language Processing , EMNLP 2004, A meeting of SIGDAT, a Special Interest Group of the ACL, held in conjunction with ACL 2004, 25-26 July 2004, Barcelona, Spain*, pages 388–395. ACL.

Taku Kudo and John Richardson. 2018. Sentencepiece: A simple and language independent subword tokenizer and detokenizer for neural text processing. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing, EMNLP 2018: System Demonstrations, Brussels, Belgium, October 31 - November 4, 2018*, pages 66–71. Association for Computational Linguistics.

Xingyu Lin, Harjatin Singh Baweja, George Kantor, and David Held. 2019. Adaptive auxiliary task weighting for reinforcement learning. In *Advances in Neural Information Processing Systems 32: Annual Conference on Neural Information Processing Systems 2019, NeurIPS 2019, December 8-14, 2019, Vancouver, BC, Canada*, pages 4773–4784.

Debabrata Mahapatra and Vaibhav Rajan. 2020. Multi-task learning with user preferences: Gradient descent with controlled ascent in pareto optimization. In *Proceedings of the 37th International Conference on Machine Learning, ICML 2020, 13-18 July 2020, Virtual Event*, volume 119 of *Proceedings of Machine Learning Research*, pages 6597–6607. PMLR.

Arturo Oncevay, Barry Haddow, and Alexandra Birch. 2020. Bridging linguistic typology and multilingual machine translation with multi-view language representations. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing, EMNLP 2020, Online, November 16-20, 2020*, pages 2391–2406. Association for Computational Linguistics.

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. Bleu: a method for automatic evaluation of machine translation. In *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics, July 6-12, 2002, Philadelphia, PA, USA*, pages 311–318. ACL.

Matt Post. 2018. A call for clarity in reporting BLEU scores. In *Proceedings of the Third Conference on Machine Translation: Research Papers, WMT 2018, Belgium, Brussels, October 31 - November 1, 2018*, pages 186–191. Association for Computational Linguistics.

Nils Reimers and Iryna Gurevych. 2020. Making monolingual sentence embeddings multilingual using knowledge distillation. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing, EMNLP 2020, Online, November 16-20, 2020*, pages 4512–4525. Association for Computational Linguistics.

Devendra Singh Sachan and Graham Neubig. 2018. Parameter sharing methods for multilingual self-attentional translation models. In *Proceedings of the Third Conference on Machine Translation: Research Papers, WMT 2018, Belgium, Brussels, October 31 - November 1, 2018*, pages 261–271. Association for Computational Linguistics.

Rico Sennrich, Barry Haddow, and Alexandra Birch. 2016. Neural machine translation of rare words with subword units. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics, ACL 2016, August 7-12, 2016, Berlin, Germany, Volume 1: Long Papers*. The Association for Computer Linguistics.

Trevor Standley, Amir Roshan Zamir, Dawn Chen, Leonidas J. Guibas, Jitendra Malik, and Silvio Savarese. 2020. Which tasks should be learned together in multi-task learning? In *Proceedings of the 37th International Conference on Machine Learning, ICML 2020, 13-18 July 2020, Virtual Event*, volume 119 of *Proceedings of Machine Learning Research*, pages 9120–9132. PMLR.

Ilya Sutskever, Oriol Vinyals, and Quoc V. Le. 2014. Sequence to sequence learning with neural networks. In *Advances in Neural Information Processing Systems 27: Annual Conference on Neural Information Processing Systems 2014, December 8-13 2014, Montreal, Quebec, Canada*, pages 3104–3112.

Xu Tan, Jiale Chen, Di He, Yingce Xia, Tao Qin, and Tie-Yan Liu. 2019. Multilingual neural machine translation with language clustering. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing, EMNLP-IJCNLP 2019, Hong Kong, China, November 3-7, 2019*, pages 963–973. Association for Computational Linguistics.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, December 4-9, 2017, Long Beach, CA, USA*, pages 5998–6008.

Qian Wang and Jiajun Zhang. 2022. Parameter differentiation based multilingual neural machine translation. In *Thirty-Sixth AAAI Conference on Artificial Intelligence, AAAI 2022, Thirty-Fourth Conference on Innovative Applications of Artificial Intelligence, IAAI 2022, The Twelveth Symposium on Educational Advances in Artificial Intelligence, EAAI 2022 Virtual Event, February 22 - March 1, 2022*, pages 11440–11448. AAAI Press.

Qian Wang, Jiajun Zhang, and Chengqing Zong. 2022. Synchronous inference for multilingual neural machine translation. *IEEE ACM Trans. Audio Speech Lang. Process.*, 30:1827–1839.

Xinyi Wang, Yulia Tsvetkov, and Graham Neubig. 2020a. Balancing training for multilingual neural machine translation. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, ACL 2020, Online, July 5-10, 2020*, pages 8526–8537. Association for Computational Linguistics.

Yining Wang, Jiajun Zhang, Feifei Zhai, Jingfang Xu, and Chengqing Zong. 2018. Three strategies to improve one-to-many multilingual translation. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing, Brussels, Belgium, October 31 - November 4, 2018*, pages 2955–2960. Association for Computational Linguistics.

Yining Wang, Long Zhou, Jiajun Zhang, Feifei Zhai, Jingfang Xu, and Chengqing Zong. 2019. A compact and language-sensitive multilingual translation method. In *Proceedings of the 57th Conference of the Association for Computational Linguistics, ACL 2019, Florence, Italy, July 28- August 2, 2019, Volume 1: Long Papers*, pages 1213–1223. Association for Computational Linguistics.

Zirui Wang, Zachary C. Lipton, and Yulia Tsvetkov. 2020b. On negative interference in multilingual models: Findings and A meta-learning treatment. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing, EMNLP 2020, Online, November 16-20, 2020*, pages 4438–4450. Association for Computational Linguistics.

Jitao Xu and François Yvon. 2021. One source, two targets: Challenges and rewards of dual decoding. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing, EMNLP 2021, Virtual Event / Punta Cana, Dominican Republic, 7-11 November, 2021*, pages 8533–8546. Association for Computational Linguistics.

Amir Roshan Zamir, Alexander Sax, William B. Shen, Leonidas J. Guibas, Jitendra Malik, and Silvio Savarese. 2018. Taskonomy: Disentangling task transfer learning. In *2018 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2018, Salt Lake City, UT, USA, June 18-22, 2018*, pages 3712–3722. Computer Vision Foundation / IEEE Computer Society.

Biao Zhang, Ankur Bapna, Rico Sennrich, and Orhan Firat. 2021. Share or not? learning to schedule language-specific capacity for multilingual translation. In *9th International Conference on Learning Representations, ICLR 2021, Virtual Event, Austria, May 3-7, 2021*. OpenReview.net.

Biao Zhang, Philip Williams, Ivan Titov, and Rico Sennrich. 2020. Improving massively multilingual neural machine translation and zero-shot translation. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, ACL 2020, Online, July 5-10, 2020*, pages 1628–1639. Association for Computational Linguistics.

Language	es	fr	ar	zh	ko	ru	tr
Data Size	413	407	404	399	396	386	374
Language	it	ja	he	pt	ro	vi	nl
Data Size	370	363	349	326	325	323	317
Language	hu	fa	pl	de	el	sr	bg
Data Size	305	302	297	294	267	258	247
Language	uk	hr	cs	id	th	sv	sk
Data Size	206	196	169	163	159	120	105

Table 7: The data size (K) of 28 langusges X↔English used in M2O and O2M translation.

	es	fr	ar	ko	ru	tr	it	ja	he	pt
es	-	401	399	393	384	371	368	362	347	320
fr	401	-	396	390	382	369	365	360	344	317
ar	399	396	-	389	381	368	365	359	344	317
ko	393	390	389	-	378	363	362	356	342	314
ru	384	382	381	378	-	356	357	355	341	307
tr	371	369	368	363	356	-	340	338	324	300
it	368	365	365	362	357	340	-	340	330	296
ja	362	360	359	356	355	338	340	-	330	290
he	347	344	344	342	341	324	330	330	-	282
pt	320	317	317	314	307	300	296	290	282	-

Table 8: The data size (K) used in many-to-many (M2M) translation.

A The Clustering Objective

We want to maximize the overall performance of all tasks by finding the clusters $\mathcal{C}$ and $\mathcal{G}$:

$$\begin{aligned} &\arg\max_{\mathcal{C},\mathcal{G},K} \sum_{k=1}^{K} \sum_{\tau_i \in c_k} Q(\tau_i|M_k) \\ &s.t.\ \ M_k = \arg\min_{M_k} \sum_{\tau_j \in g_k} \sum_{(\mathbf{x},\mathbf{y}) \in \tau_j} -\log p(\mathbf{y}|\mathbf{x}, M_k) \end{aligned} \tag{16}$$

Assume that the quality $Q(\tau_i|M_k)$ on task τ_i with the model trained on tasks $\tau_j \in g_k$ is an approximation of the average of the model qualities on task τ_i with the models trained on task τ_j and task τ_i:

$$Q(\tau_i|M_k) \approx \frac{1}{|g_k|} \sum_{\tau_j \in g_k} Q(\tau_i|M_{\tau_i,\tau_j}) \tag{17}$$

	HTAC	Rand.	Temp.	Fix.	FTAC
es→en	46.4	45.1	46.8	47.4	**48.4**
fr→en	40.5	39.2	40.5	41.0	**42.1**
ar→en	36.5	34.4	36.8	36.4	**37.9**
zh→en	25.8	24.8	25.7	25.6	**26.6**
ko→en	23.2	22.0	22.8	22.8	**23.7**
ru→en	28.8	27.8	28.2	27.2	**28.9**
tr→en	29.5	28.3	28.9	29.8	**30.5**
it→en	41.0	39.6	40.4	39.5	**41.3**
ja→en	19.0	17.8	18.6	18.5	**19.6**
he→en	40.7	38.5	40.8	41.4	**42.9**
pt→en	44.3	42.8	44.4	44.9	**45.9**
ro→en	39.9	38.7	40.7	40.7	**41.9**
vi→en	32.2	32.1	31.0	31.4	**32.7**
nl→en	38.8	37.5	38.8	38.9	**40.1**
hu→en	27.1	27.5	25.8	27.6	**28.7**
fa→en	33.0	31.6	33.2	32.6	**34.5**
pl→en	27.3	25.9	26.8	27.5	**28.1**
de→en	35.8	37.0	36.9	36.3	**37.4**
el→en	41.5	39.1	40.8	40.9	**42.5**
sr→en	**42.0**	40.7	40.3	40.5	41.7
bg→en	**42.8**	40.3	42.5	41.2	42.7
uk→en	31.4	30.9	31.5	30.6	**31.5**
hr→en	**43.6**	39.5	43.3	41.2	42.8
cs→en	36.3	33.4	34.6	36.0	**36.7**
id→en	34.6	33.7	32.7	34.0	**35.4**
th→en	28.1	26.6	27.4	27.9	**29.1**
sv→en	43.4	41.6	42.9	42.3	**44.4**
sk→en	37.4	33.5	35.5	37.3	**38.1**
Average	35.4	33.9	35.0	35.1	**36.3**

Table 9: BLEU score of many-to-one (M2O) translation with different data sampling strategies.

where M_{τ_i,τ_j} is the model trained on tasks τ_j and τ_i. Then the objective becomes:

$$\begin{aligned} &\arg\max_{\mathcal{C},\mathcal{G},K} \sum_{k=1}^{K} \sum_{\tau_i \in c_k} Q(\tau_i|M_k) \\ \approx &\arg\max_{\mathcal{C},\mathcal{G},K} \sum_{k=1}^{K} \sum_{\tau_i \in c_k} \frac{1}{|g_k|} \sum_{\tau_j \in g_k} Q(\tau_i|M_{\tau_i,\tau_j}) \end{aligned} \tag{18}$$

Since the task $\tau_i \in c_k$ must appear in g_k, the term $Q(\tau_i|M_{\tau_i})$ is irrelevant to the $\arg\max$ operation:

$$\begin{aligned} &\arg\max_{\mathcal{C},\mathcal{G},K} \sum_{k=1}^{K} \sum_{\tau_i \in c_k} Q(\tau_i|M_k) \\ \approx &\arg\max_{\mathcal{C},\mathcal{G},K} \sum_{k=1}^{K} \sum_{\tau_i \in c_k} \sum_{\tau_j \in g_k} \big(Q(\tau_i|M_{\tau_i,\tau_j}) - Q(\tau_i|M_{\tau_i})\big) \\ = &\arg\max_{\mathcal{C},\mathcal{G},K} \sum_{k=1}^{K} \sum_{\tau_i \in c_k} \sum_{\tau_j \in g_k} Z_{j \to i} \end{aligned} \tag{19}$$

k	c_k	$g_k \setminus c_k$
1	ja-ru	it-ru, tr-ru, ja-it, it-pt, ar-ru, pt-ru, ko-ru, ja-fr, ru-fr, he-fr, fr-ru, es-ru, ar-es
2	ar-es	es-it, ja-es, ru-es, ar-it, pt-es, tr-es, es-ar, ar-pt, ko-es, fr-pt, ar-ko, fr-es
3	he-tr,ja-tr	fr-tr, ru-tr, es-tr, it-tr, pt-tr
4	pt-fr,tr-fr	ar-fr, it-fr, es-fr, ko-fr
5	ru-tr,it-tr	ar-tr, he-tr, ja-tr
6	he-ja,ar-ja	tr-ja, ko-ja, it-ja, pt-ja
7	ko-he,tr-he	ja-he, ru-he, es-he, ar-he, it-he
8	ru-ar,fr-ar	tr-ar, es-ar, ko-ar, it-ar
9	he-es,ja-es,it-es	ru-es, pt-es, ko-es, ar-es, fr-es
10	tr-ru,he-ru,es-ru	ar-ru, pt-ru, fr-ru
11	es-it,ru-it,ja-it	he-it, pt-it, ko-fr, fr-es, pt-es, he-it, fr-it, pt-it, tr-fr
12	pt-he,ru-he,ja-he,it-he	ko-he, tr-he, fr-ar
13	es-fr,he-fr,ru-fr	ar-fr, it-fr, pt-fr
14	ja-ar,tr-ar,ko-ar	ru-ar, fr-ar, es-ar, he-ar, it-ar
15	tr-pt,ja-pt,fr-pt,he-pt	ko-pt, it-pt, tr-it, ru-pt, es-pt
16	he-ko,ru-ko,ja-ko	tr-ja, ja-ar, pt-ko, es-tr, ar-ko, ru-ar, tr-ko
17	es-he,ar-he,fr-he	tr-he, it-he, es-ru, pt-he, ru-he, ko-he, ja-he
18	pt-es,ru-es,ko-es,tr-es,fr-es	ja-es, it-es, ko-fr, ru-ar
19	es-ar,he-ar,it-ar,pt-ar	ja-ar, ko-ja, tr-ar, fr-ar, pt-fr, it-pt, ko-ar
20	it-fr,ko-fr,ar-fr,ja-fr	es-fr, pt-fr, tr-fr, ru-fr
21	pt-ru,ko-ru,fr-ru,it-ru,ar-ru	es-ru, it-pt, he-ru
22	es-tr,pt-tr,ko-tr,fr-tr,ar-tr	ja-tr, ru-tr
23	it-ko,tr-ko,ar-ko,fr-ko,es-ko,pt-ko	ja-ko, he-ko, ru-ko
24	ru-pt,it-pt,es-pt,ar-pt,ko-pt	es-fr, ru-it, pt-it, ja-pt, fr-pt, he-pt
25	pt-ja,es-ja,tr-ja,ru-ja,fr-ja,ko-ja,it-ja	es-ko, ar-ja
26	he-it,ko-it,tr-it,fr-it,ar-it,pt-it	fr-es, ko-fr, ru-it, ru-pt, es-it

Table 10: The clustering results of our method for many-to-many translation.

B Data Statistics and Pre-processing

In our experiments, we use the data from the TED 2020 Parallel Sentences Corpus[7], which contains sentences from TED talks with their translations provided by a global community of volunteers[8].

For one-to-many and many-to-one translation, we select 28 languages and the data statistics are shown in Table 7. For many-to-many translation, we select 10 languages and the data statistics are shown in Table 8.

For the data pre-processing, we use Jieba[9] for segmenting Chinese sentences, Mecab[10] for Japanese and Korean, and Moses[11] for other languages.

[7] https://github.com/UKPLab/sentence-transformers/blob/master/docs/datasets/TED2020.md.

[8] https://www.ted.com/participate/translate

[9] https://github.com/fxsjy/jieba

[10] https://taku910.github.io/mecab/

[11] https://github.com/moses-smt/mosesdecoder

C M2M Clustering

We show the task affinities of many-to-many (M2M) translation in Figure 3 and the clustering results in Table 10. From figure 3, we find that the tasks with the same target languages share higher affinities compared to the tasks with the same source languages. From the clustering results in Table 10, we also find that the tasks with the same target languages are more likely to be in the same cluster or serve as auxiliary tasks.

D The Effects of Auxiliary Tasks

The detailed BLEU scores of different data sampling strategies for auxiliary tasks are shown in Table 9.

E Results on OPUS dataset

In our experiments, we randomly split the data into training/validation/test set. However, there can be overlap between sentences in train and test/validation sets of different language pairs. For example, one English sentence X in the test set of task en-de may occur in the training set of en-fr. To make the results more convincing, we evaluate the

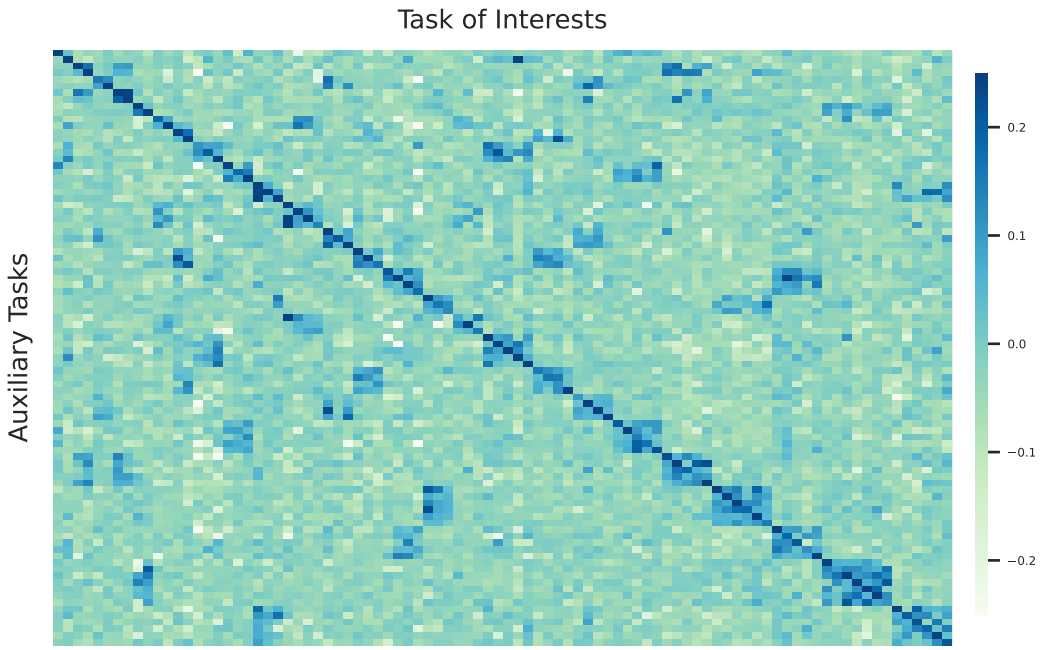

Figure 3: The affinity heatmap between tasks in many-to-many translation. The orders of tasks in X and Y axes are identical to the tasks of c_k in Table 10.

	Multi.	**LEC**	**FTAC**	Δ_1	Δ_2
es→en	25.1	27.0	28.9	+1.9	+3.8
fr→en	16.1	18.4	20.0	+1.6	+3.9
ar→en	19.2	21.3	23.3	+2.0	+4.1
zh→en	10.0	10.9	11.5	+0.6	+1.5
ko→en	12.6	14.5	15.1	+0.6	+2.5
ru→en	15.4	16.4	17.7	+1.3	+2.3
tr→en	21.1	22.2	24.3	+2.1	+3.2
it→en	20.6	23.1	23.8	+0.7	+3.2
ja→en	11.9	12.9	14.1	+1.2	+2.2
he→en	28.9	30.9	33.7	+2.8	+4.8
pt→en	24.8	27.0	28.3	+1.3	+3.5
ro→en	27.3	28.6	30.6	+2.0	+3.3
vi→en	19.9	22.8	23.2	+0.4	+3.3
nl→en	21.3	23.2	24.6	+1.4	+3.3
hu→en	19.3	20.1	22.2	+2.1	+2.9
fa→en	13.1	15.4	17.0	+1.6	+3.9
pl→en	20.0	20.7	22.7	+2.0	+2.7
de→en	13.7	16.8	17.3	+0.5	+3.6
el→en	22.2	25.3	24.1	-1.2	+1.9
sr→en	24.0	26.9	25.7	-1.2	+1.7
bg→en	23.8	25.6	26.1	+0.5	+2.3
uk→en	21.4	22.3	22.4	+0.1	+1.0
hr→en	23.6	27.0	26.2	-0.8	+2.6
cs→en	20.7	22.5	23.3	+0.8	+2.6
id→en	24.9	29.5	28.8	-0.7	+3.9
th→en	16.9	18.0	18.9	+0.9	+2.0
sv→en	17.7	19.6	20.8	+1.2	+3.1
sk→en	20.5	22.6	22.7	+0.1	+2.2
Average	19.9	21.8	22.8	+0.9	+2.9

Table 11: BLEU score of many-to-one (M2O) translation on the OPUS test set.

M2O model on the OPUS-100 (Zhang et al., 2020) multilingual test set which contains no overlap sentences in different tasks, and the results are shown in Table 11.

Learning Decoupled Retrieval Representation for Nearest Neighbour Neural Machine Translation

Qiang Wang[1,2], **Rongxiang Weng**[3,4], **Ming Chen**[2*]
[1]Zhejiang University, Hangzhou, China
[2]RoyalFlush AI Research Insistute, Hangzhou, China
[3]School of Computer Science and Technology, Soochow University, Suzhou, China
[4]miHoYo AI, Shanghai, China
{wangqiangenu, wengrongxiang}@gmail.com, chenming@myhexin.com

Abstract

K-Nearest Neighbor Neural Machine Translation (*k*NN-MT) successfully incorporates external corpus by retrieving word-level representations at test time. Generally, *k*NN-MT borrows the off-the-shelf context representation in the translation task, e.g., the output of the last decoder layer, as the query vector of the retrieval task. In this work, we highlight that coupling the representations of these two tasks is sub-optimal for fine-grained retrieval. To alleviate it, we leverage supervised contrastive learning to learn the distinctive retrieval representation derived from the original context representation. We also propose a fast and effective approach to constructing hard negative samples. Experimental results on five domains show that our approach improves the retrieval accuracy and BLEU score compared to vanilla *k*NN-MT.

1 Introduction

Conventional neural machine translation (NMT) cannot dynamically incorporate external corpus at inference once finishing training (Bahdanau et al., 2015; Vaswani et al., 2017), resulting in bad performance when facing unseen domains, even if feeding millions or billions of sentence pairs for training (Koehn and Knowles, 2017). To address this problem, researchers developed retrieval-enhanced NMT (RENMT) to flexibly incorporate external translation knowledge. Early RENMTs leverage a search engine to find the similar bitext to improve the translation performance (Zhang et al., 2018; Cao and Xiong, 2018; Gu et al., 2018; Xia et al., 2019). However, the results of sentence-level retrieval with high similarity are generally sparse in practical applications, while noises in low similarity retrieval could lead to severe performance degradation (Cao and Xiong, 2018).

*k*NN-MT proposed by Khandelwal et al. (2021) effectively alleviates the sparse problem by introducing the word-level k-nearest neighbor mechanism. Instead of storing the discrete word sequence, *k*NN-MT uses a pre-trained NMT model to force decoding the external corpus and remembers the word-level continuous context representation, e.g., the output of the last decoder layer. During inference, *k*NN-MT assumes that the same target words have similar contextual representations and weights word selection through retrieving current context representation from the memorized datastore. However, we point out that it is sub-optimal to directly use the off-the-shelf context representation in the translation task because this vector is not specific to fine-grained retrieval.

In this work, we attempt to decouple the context representation by learning an independent retrieval representation. To this end, we leverage supervised contrastive learning with multiple positive and negative samples to learn a good retrieval representation (called CLKNN). We also propose a fast and effective method to construct hard negative samples. Experimental results on five domains show that our approach outperforms the vanilla *k*NN-MT in terms of BLEU and retrieval accuracy.

2 Background

Vanilla NMT Given a source sentence $\boldsymbol{x} = \{x_1, x_2, \ldots, x_{|\boldsymbol{x}|}\}$ and a target prefix $\boldsymbol{y}_{<t} = \{y_1, y_2, \ldots, y_{t-1}\}$, the vanilla NMT predicts the next target word y_t by:

$$p_c(y_t|\boldsymbol{x}, \boldsymbol{y}_{<t}) \propto \exp\Big(q(\boldsymbol{h}_t)\Big) \tag{1}$$

where $\boldsymbol{h}_t = f_\theta(\boldsymbol{x}, \boldsymbol{y}_{<t}) \in \mathcal{R}^d$ is the context vector at step t with respect to $\boldsymbol{x}$ and $\boldsymbol{y}_{<t}$; $f_\theta(\cdot)$ can be arbitrary encoder-decoder network with parameters θ, such as Transformer (Vaswani et al., 2017); $q(\cdot)$ linearly projects $\boldsymbol{h}_t$ to target vocabulary size.

kNN-MT *k*NN-MT hypothesizes that the same target words have similar representations. To

*Corresponding author.

Proceedings of the 29th International Conference on Computational Linguistics, pages 5142–5147
October 12–17, 2022.

dynamically incorporate external sentence pairs $\mathcal{D} = \{(\boldsymbol{x}^{(i)}, \boldsymbol{y}^{(i)})\}_{i=1}^{|\mathcal{D}|}$, kNN-MT extends Eq. 1 by interpolating a retrieval-based probability p_r:

$$p_{knn} = (1-\lambda) \times p_c + \lambda \times p_r \tag{2}$$

where λ is the interpolation coefficient as a hyper-parameter.

Specifically, kNN-MT first uses a pre-trained NMT model to force decoding each sentence pair $(\boldsymbol{x}^{(i)}, \boldsymbol{y}^{(i)})$ to build a key-value datastore $\mathcal{H}$:

$$\mathcal{H} = \bigcup_{i=1}^{|\mathcal{D}|} \bigcup_{t=1}^{|\boldsymbol{y}^{(i)}|} \left\{ (\boldsymbol{h}_t^{(i)}, y_t^{(i)}) \right\} \tag{3}$$

The key is the word-level context representation $\boldsymbol{h}_t^{(i)}$ and the value is the gold target word $y_t^{(i)}$. Then, given $\mathcal{H}$ and predicted target prefix $\hat{\boldsymbol{y}}_{<t}$ at test time, kNN-MT models $p_r(\hat{y}_t|\boldsymbol{x}, \hat{\boldsymbol{y}}_{<t})$ by measuring the distance between query $\hat{\boldsymbol{h}}_t = f_\theta(\boldsymbol{x}, \hat{\boldsymbol{y}}_{<t})$ and its k-nearest representations $\{(\tilde{\boldsymbol{h}}_i, \tilde{v}_i)\}_{i=1}^{k}$ in $\mathcal{H}$:

$$p_r(\hat{y}_t|\boldsymbol{x}, \hat{\boldsymbol{y}}_{<t}) \propto \sum_{i=1}^{k} \mathbb{1}_{\hat{y}_t=\tilde{v}_i} \exp\Big(\frac{-d(\tilde{\boldsymbol{h}}_i, \hat{\boldsymbol{h}}_t)}{T}\Big), \tag{4}$$

where $d(\cdot)$ is L_2 distance; T is temperature hyper-parameter; $\mathbb{1}$ is the indicator function.

3 Approach

Motivation According to Eq. 1-4, we can see that the context representation $\boldsymbol{h}$ simultaneously plays two roles in kNN-MT: (1) the semantic vector for p_c; (2) the retrieval vector for p_r. We note that coupling the same $\boldsymbol{h}$ in the two scenes is sub-optimal. Recall that $\boldsymbol{h}$ in the translation model is generally learned through cross-entropy loss, which only pays attention to the gold target token and ignores others.[1] However, a good retrieval vector should be able to distinguish between different tokens, especially those owning similar representations. Therefore, we attempt to derive a new retrieval vector $\boldsymbol{z}$ from $\boldsymbol{h}$ for better retrieval performance.

Retrieval representation adapter We use a simple feedforward network as an adapter to transform the original representation $\boldsymbol{h}$ to desired retrieval representation $\boldsymbol{z}$:

$$\boldsymbol{z} = \text{FFN}(\boldsymbol{h}) = \text{ReLU}(\boldsymbol{h}\boldsymbol{W}_1 + \boldsymbol{b}_1)\boldsymbol{W}_2 + \boldsymbol{b}_2, \tag{5}$$

where $\boldsymbol{W}_1 \in \mathcal{R}^{d \times d_f}$, $\boldsymbol{W}_2 \in \mathcal{R}^{d_f \times d_o}$, $\boldsymbol{b}_1 \in \mathcal{R}^{d_f}$, and $\boldsymbol{b}_2 \in \mathcal{R}^{d_o}$ are learnable parameters; d_f and d_o are the intermediate hidden size and output size of the adapter, respectively. When $d_o < d$, the adapter network can be regarded as a dimension reducer. As FFN is very lightweight compared to the calculation of $\boldsymbol{h}$, there is almost no latency in converting $\boldsymbol{h}$ to $\boldsymbol{z}$.For convenience, in the following description, we redefine $\boldsymbol{h}_i$ as the key of i-th key-value pair in the original datastore $\mathcal{H}$, and the corresponding value is denoted by Y_i when there is no ambiguity. In this way, the new datastore $\mathcal{Z}$ can be denoted as $\mathcal{Z} = \{(\boldsymbol{z}_i, Y_i)|i = 1, \ldots, |\mathcal{H}|\}$, where $\boldsymbol{z}_i = \text{FFN}(\boldsymbol{h}_i)$.

Supervised contrastive learning In machine translation field, contrastive learning has been applied in multilingual translation (Pan et al., 2021; Wei et al., 2021), cross-modal translation (Ye et al., 2022), and learning robust representation for low-frequency word (Zhang et al., 2021) etc. In this work, we use supervised contrastive learning (Grill et al., 2020) with multiple positive and negative samples to learn the desired retrieval representation $\boldsymbol{z}$. Here, we regard the unique token v in the target vocabulary V as a natural supervision signal. We aim to make $\boldsymbol{z}$ more distinguishable, for example, pulling $\boldsymbol{z}$ of the same words together and pushing $\boldsymbol{z}$ of different words apart. Specifically, we first divide $\mathcal{Z}$ into $|V|$ clusters according to the token class label. E.g., $C_v = \{\boldsymbol{z}_i|i = 1, \ldots, |\mathcal{Z}|, Y_i = v\}$, where C_v is the context representation cluster of token v. Thus, given any context representation $\boldsymbol{z} \in \mathcal{Z}$ and its token label v, we can construct M positive samples $\boldsymbol{z}^+ = \{\boldsymbol{z}_1^+, \ldots, \boldsymbol{z}_i^+, \ldots, \boldsymbol{z}_M^+\}$, where $\boldsymbol{z}_i^+$ is uniformly sampled from its owned cluster C_v and $\boldsymbol{z}_i^+ \neq \boldsymbol{z}$.[2] Likely, we further construct N negative samples $\boldsymbol{z}^- = \{\boldsymbol{z}_1^-, \ldots, \boldsymbol{z}_i^-, \ldots, \boldsymbol{z}_N^-\}$, where $\boldsymbol{z}_i^- \in \backslash C_v$, $\backslash C_v$ denotes other clusters except C_v. In the next part, we will describe how to build $\boldsymbol{z}^-$. Finally, given the anchor vector $\boldsymbol{z}$, its multiple positive samples $\boldsymbol{z}^+$ and multiple negative samples $\boldsymbol{z}^-$, we learn the adapter network through the following contrastive learning loss:

$$-\log \frac{\sum\limits_{1 \le i \le M} \exp(s(\boldsymbol{z}, \boldsymbol{z}_i^+))}{\sum\limits_{1 \le i \le M} \exp(s(\boldsymbol{z}, \boldsymbol{z}_i^+)) + \sum\limits_{1 \le j \le N} \exp(s(\boldsymbol{z}, \boldsymbol{z}_j^-))}, \tag{6}$$

where $s(\cdot)$ is the score function implemented as cosine similarity with temperature T': $s(\boldsymbol{a}, \boldsymbol{b}) =$

[1] In practice, we often use its label-smooth variant, which evenly assigns a small probability mass to all non-gold labels without distinction.

[2] We use sampling with replacement when $|C_v| < M$.

$\frac{1}{T'} \times \frac{\boldsymbol{a}^T \boldsymbol{b}}{\|\boldsymbol{a}\| \cdot \|\boldsymbol{b}\|}$. Note that T' is the temperature in training, which is different from the inference temperature T in Eq. 4.

Fast hard negative sample The key for Eq. 6 is the construction of negative samples $\boldsymbol{z}^-$. A trivial solution is randomly sampling from the entire space of $\backslash C_v$. However, this negative sample may be too easy to provide the effective learning signal (Robinson et al., 2020). On the contrary, an extreme method for hard negative samples is to traverse $\backslash C_v$ to find the most similar negative samples for the anchor. The problem is that $|\backslash C_v|$ is close to $|\mathcal{Z}|$, with a scale of millions or more, resulting in enormous computational complexity. To solve it, we propose a fast and cheap approach to constructing hard negative samples. Specifically, we first collect the cluster centre $\bar{C}_v = \frac{1}{|C_v|} \sum_{i=1}^{|\mathcal{Z}|} \mathbb{1}_{Y_i=v} \boldsymbol{z}_i$. We calculate the nearest K (K>=N) cluster centers w.r.t the anchor and randomly sample N clusters to make the source of the negative sample diverse. Then we randomly sample one point from the corresponding cluster as a negative sample. As the anchor vector only involves querying $|C|$ cluster centers and $|C| << |\mathcal{Z}|$, our approach runs faster than the exact global search.

Inference After training, we use the well-trained FFN to rebuild the retrieval datastore $\mathcal{H}$ into $\mathcal{Z}$. To further reduce calculation cost at test time, we introduce PCA to reduce the dimension of the retrieval vector. We also add normalization after PCA to guarantee the numerical stability of the input to the inner product. Another difference with Eq. 4 is that we use the inner product instead of the L2 distance as distance metrics. The reason is that using consistent distance metrics in training and inference improves performance in primitive experiments. Concretely, we modify the original kNN-MT in Eq. 4 as:

$$p_r(\hat{\boldsymbol{y}}_t|\boldsymbol{x}, \hat{\boldsymbol{y}}_{<t}) \propto \sum_{i=1}^{k} \mathbb{1}_{\hat{y}_t=\tilde{v}_i} \exp\Big(\frac{g(\tilde{\boldsymbol{z}}_i) \otimes g(\hat{\boldsymbol{z}}_t)}{T}\Big), \tag{7}$$

where $g(x) = \text{Norm}(\text{PCA}(x))$, $\otimes$ denotes inner product operation, $\tilde{\boldsymbol{z}}_i$ is the i-th nearest neighbor in $\mathcal{Z}$ for the current retrieval representation $\hat{\boldsymbol{z}}_t$. As a bonus, since the numeric range of the normalized inner product is $[0, 1]$, which can be seen as the confidence in retrieving.[3] We leverage this nature

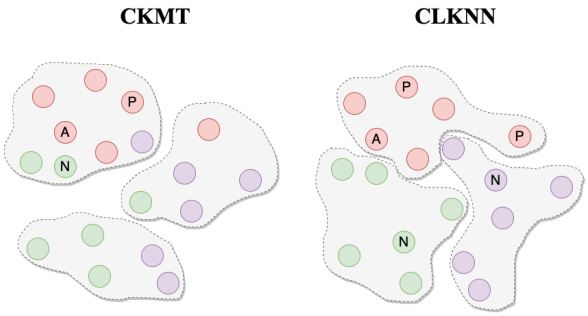

Figure 1: Illustration the differences between CKMT and CLKNN in constructing positive and negative samples. Different colors indicate different tokens. A/P/N means anchor, positive sample and negative sample, respectively.

to modify the interpolation coefficient λ in Eq. 2 to be aware of retrieval confidence:

$$\lambda^* = \lambda \times \frac{\sum_{i=1}^{k} g(\tilde{\boldsymbol{z}}_i) \otimes g(\hat{\boldsymbol{z}}_t)}{k}. \tag{8}$$

λ^* can be considered a simple adaptive coefficiency like Zheng et al. (2021); Jiang et al. (2021); Wang et al. (2022), but does not require training.

Discussion The closest work with us is CKMT (Wang et al., 2022). As illustrated in Figure 1, there are two major differences compared with CKMT: (1) CLKNN uses multiple positive and negative samples, while CKMT only considers a single positive and negative sample, limiting the exploration of representation space. (2) CKMT requires to partition clusters through cost-expensive clustering in full-scale datastore, while CLKNN predefines clusters based on vocabulary labels and only involves calculating cluster centers. In practice, we spent about 6 hours on the CPU to complete the cluster operation in CKMT, while CLKNN only takes about 3 minutes.

4 Experiments

Setup To fairly compared with previous work (Khandelwal et al., 2021), we use WMT'19 German-English news translation task winner (Ng et al., 2019) as our strong general domain baseline. We use the same German-English multi-domain datasets, consisting of five domains, including `Medical`, `Law`, `IT`, `Koran` and `Subtitles` [4]. Besides, to test the proposed training approach robust in out-domain scenery, we also use a 2M subset of the baseline's training data, including *News*

[3] L2 distance lacks this feature because its numeric range is too broad, e.g., 0~1000 in our observation.

[4] We use the provided 500K sentence pairs version subtitle data rather than full size 12.4M due to memory limitation.

Dataset	Medical	Law	IT	Koran	Subtitle	NC+Euro
Train	248K	467K	222K	52K	500K	2M
Valid	2000	2000	2000	2000	2000	-
Test	2000	2000	2000	2000	2000	-
Datastore	6.9M	19.0M	3.6M	0.5M	6.2M	5M†

Table 1: Statistics of datasets in different domains. †: Due to limited memory, we randomly sampled 5M samples from a total of 65.7M samples in NC+Euro for training.

Method	Medical	Law	IT	Koran	Subtitle	Avg.
Baseline (WMT19 winner, Ng et al. (2019))	39.91	45.71	37.98	16.3	29.21	33.82
*k*NN-MT (Khandelwal et al., 2021)	54.35	61.78	45.82	19.45	**31.73**†	42.63
*k*NN-MT (our implementation)	54.41	61.01	45.20	21.07	29.67	42.27
train by out-domain data						
CLKNN	56.37	61.54	46.50	21.52	30.81	43.35
CLKNN + λ^*	**56.52**	61.63	46.68	21.60	30.86	43.46
train by in-domain data						
CLKNN	55.86	61.92	47.77	21.46	31.02	43.61
CLKNN + λ^*	55.87	**62.01**	**47.84**	**21.81**	31.05	**43.72**

Table 2: The SacreBLEU scores of our proposed CLKNN and the baseline methods in five domains. λ^* denotes using retrieval confidence aware interpolation coefficiency. † denotes the number is not comparable because Khandelwal et al. (2021) use full-size subtitle data than ours. All the CLKNN results are significantly better ($p<0.01$) than our re-implemented *k*NN-MT, measured by paired bootstrap resampling (Koehn, 2004).

Commentary v14 and *Europarl v9*, and randomly sample 5M samples out of 65.7M samples from its datastore. See Table 1 for detailed data statistics.

Implementation details All experiments run on a single NVIDIA 2080 Ti GPU. We use *Faiss* [5] for vector retrieval. For CLKNN, the number of positive samples is M=2, and the number of negative samples is N=32. We sample N negative samples from K=128 nearest clusters. The training batch size is 32. During training, we set $T^{'}$=0.01, while we vary T according to the validation set at test time. The hidden state size d_f and output size d_o of adapter is 4096 and 512, respectively. The output dimension of PCA is 128. We train all models for 500k steps and select the best model on the validation set. We use a beam size of 5 and a length penalty of 1.0 for all experiments for inference. We measure case-sensitive detokenized BLEU by SacreBLEU.

Experimental results Table 2 reports the SacreBLEU scores in five domains. We can see that: (1) CLKNN is robust about training data: using out-domain or in-domain average improves 1+ points than our *k*NN-MT; (2) The gap between in-domain and out-domain is small (about 0.3 points), meaning that our approach does not rely on in-domain data and is more practical than Zheng et al. (2021); Jiang et al. (2021); (3) using proposed λ^* slightly improve the performance across the board. These results show that learning independent retrieval representation is helpful for vanilla *k*NN-MT. Besides, we also compare the inference speed between CLKNN and *k*NN-MT through running five times on IT test set. The results show that CLKNN has a comparable speed (97%±2%) to that of *k*NN-MT because the adapter in CLKNN is very lightweight.

[5] https://github.com/facebookresearch/faiss

M	N	BLEU	M	N	BLEU
1	1	45.54	2	16	46.37
1	16	45.91	2	32	46.68
1	32	46.13	2	64	46.55
1	64	45.88	4	32	46.29

Table 3: The BLEU scores on IT test set against the number of the positive (M) and negative (N) samples.

5 Analysis

Effect of the number of contrastive samples One of the main differences between Wang et al. (2022) and us is that we use multiple positive and negative samples in our training objective. We vary the number of M and N and report the BLEU scores in Table 3. As we can see, increasing M and N is helpful for our method. However, large M can-

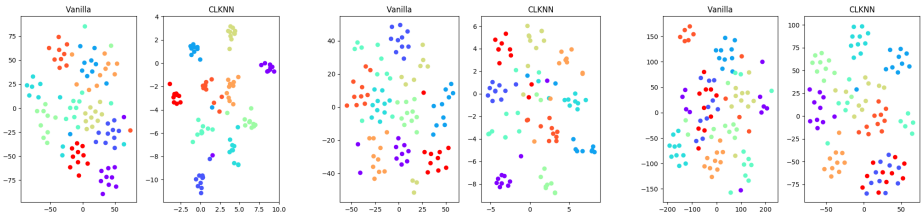

(a) High (b) Middle (c) Low

Figure 2: Visualization of retrieval vector on different frequency words by t-SNE. We uniformly sample 10 classes in each category, and each class contains ten random representations. The same color denotes the same class.

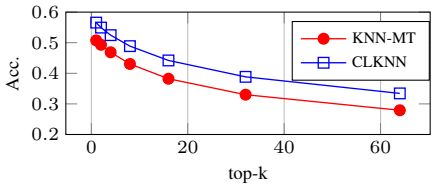

Figure 3: Retrieval accuracy curve against top-k.

not befit more than increasing N. We attribute it to positive samples that are too easy to learn because most are close in embedding space. On the contrary, negative samples from different clusters can provide a stronger learning signal. To further validate the effectiveness of multiple samples, we also conduct experiments on `Medical`. The results are similar to that of `IT`: using M=2, N=32 is 1.64 BLEU points higher than using M=1, N=1 (56.52 vs. 54.88). It indicates that using multiple positive and negative samples is necessary to achieve good performance for contrastive learning.

Retrieval accuracy Intuitively, our approach can learn more accurate retrieval representation than vanilla *k*NN-MT. To validate this hypothesis, we use `IT` validation as the datastore and plot the retrieval accuracy on top-k in Figure 3. We can see that CLKNN has more robust retrieval accuracy than *k*NN-MT no matter how k changes. It indicates that the performance improvement comes from our better retrieval representation.

Visualization We visually present the differences between baseline and CLKNN on embedding space. Specifically, we split three categories according to the word frequency in `IT` training set: `HIGH`(the first 1%), `Middle`(40%-60%) and `LOW`(the last 1%) [6]. We uniformly sample 10 unique words in each category and randomly sample 10 unique vector representations from the training datastore. We use t-SNE to plot these representations, as illustrated in Figure 2. We can see that: (1) high-frequency words' representations are prone to distinguish for both baseline and CLKNN; (2) CLKNN has more close distances in the same vocabulary than baseline; (3) CLKNN has more robust accuracy for low-frequency words.

[6]We filter words whose frequency is less than 10.

6 Conclusion

In this work, we proposed to use supervised contrastive learning to decouple the context representation from vanilla *k*NN-MT. Experimental results on several tasks show that our approach outperforms *k*NN-MT and learns a more accurate retrieval representation.

Acknowledgements

We would like to thank the anonymous reviewers for the helpful comments. We also thank Shuqin Pan for the writing suggestions.

References

Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. 2015. Neural machine translation by jointly learning to align and translate. In *3rd International Conference on Learning Representations, ICLR 2015*.

Qian Cao and Deyi Xiong. 2018. Encoding gated translation memory into neural machine translation. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 3042–3047, Brussels, Belgium. Association for Computational Linguistics.

Jean-Bastien Grill, Florian Strub, Florent Altché, Corentin Tallec, Pierre Richemond, Elena Buchatskaya, Carl Doersch, Bernardo Avila Pires, Zhaohan Guo, Mohammad Gheshlaghi Azar, Bilal Piot, koray kavukcuoglu, Remi Munos, and Michal Valko. 2020. Bootstrap your own latent - a new

approach to self-supervised learning. In *Advances in Neural Information Processing Systems*, volume 33, pages 21271–21284. Curran Associates, Inc.

Jiatao Gu, Yong Wang, Kyunghyun Cho, and Victor O. K. Li. 2018. Search engine guided neural machine translation. In *Proceedings of the Thirty-Second AAAI Conference on Artificial Intelligence, (AAAI-18), the 30th innovative Applications of Artificial Intelligence (IAAI-18), and the 8th AAAI Symposium on Educational Advances in Artificial Intelligence (EAAI-18), New Orleans, Louisiana, USA, February 2-7, 2018*, pages 5133–5140. AAAI Press.

Qingnan Jiang, Mingxuan Wang, Jun Cao, Shanbo Cheng, Shujian Huang, and Lei Li. 2021. Learning kernel-smoothed machine translation with retrieved examples. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 7280–7290, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Urvashi Khandelwal, Angela Fan, Dan Jurafsky, Luke Zettlemoyer, and Mike Lewis. 2021. Nearest neighbor machine translation. In *International Conference on Learning Representations*.

Philipp Koehn. 2004. Statistical significance tests for machine translation evaluation. In *Proceedings of the 2004 Conference on Empirical Methods in Natural Language Processing*, pages 388–395, Barcelona, Spain. Association for Computational Linguistics.

Philipp Koehn and Rebecca Knowles. 2017. Six challenges for neural machine translation. In *Proceedings of the First Workshop on Neural Machine Translation*, pages 28–39.

Nathan Ng, Kyra Yee, Alexei Baevski, Myle Ott, Michael Auli, and Sergey Edunov. 2019. Facebook FAIR's WMT19 news translation task submission. In *Proceedings of the Fourth Conference on Machine Translation (Volume 2: Shared Task Papers, Day 1)*, pages 314–319, Florence, Italy. Association for Computational Linguistics.

Xiao Pan, Mingxuan Wang, Liwei Wu, and Lei Li. 2021. Contrastive learning for many-to-many multilingual neural machine translation. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 244–258, Online. Association for Computational Linguistics.

Joshua David Robinson, Ching-Yao Chuang, Suvrit Sra, and Stefanie Jegelka. 2020. Contrastive learning with hard negative samples. In *International Conference on Learning Representations*.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in Neural Information Processing Systems*, pages 6000–6010.

Dexin Wang, Kai Fan, Boxing Chen, and Deyi Xiong. 2022. Efficient cluster-based k-nearest-neighbor machine translation. *CoRR*, abs/2204.06175.

Xiangpeng Wei, Rongxiang Weng, Yue Hu, Luxi Xing, Heng Yu, and Weihua Luo. 2021. On learning universal representations across languages. In *International Conference on Learning Representations*.

Mengzhou Xia, Guoping Huang, Lemao Liu, and Shuming Shi. 2019. Graph based translation memory for neural machine translation. In *The Thirty-Third AAAI Conference on Artificial Intelligence, AAAI 2019, The Thirty-First Innovative Applications of Artificial Intelligence Conference, IAAI 2019, The Ninth AAAI Symposium on Educational Advances in Artificial Intelligence, EAAI 2019, Honolulu, Hawaii, USA, January 27 - February 1, 2019*, pages 7297–7304. AAAI Press.

Rong Ye, Mingxuan Wang, and Lei Li. 2022. Cross-modal contrastive learning for speech translation. In *Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 5099–5113, Seattle, United States. Association for Computational Linguistics.

Jingyi Zhang, Masao Utiyama, Eiichro Sumita, Graham Neubig, and Satoshi Nakamura. 2018. Guiding neural machine translation with retrieved translation pieces. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 1325–1335, New Orleans, Louisiana. Association for Computational Linguistics.

Tong Zhang, Wei Ye, Baosong Yang, Long Zhang, Xingzhang Ren, Dayiheng Liu, Jinan Sun, Shikun Zhang, Haibo Zhang, and Wen Zhao. 2021. Frequency-aware contrastive learning for neural machine translation. *CoRR*, abs/2112.14484.

Xin Zheng, Zhirui Zhang, Junliang Guo, Shujian Huang, Boxing Chen, Weihua Luo, and Jiajun Chen. 2021. Adaptive nearest neighbor machine translation. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 2: Short Papers)*, pages 368–374, Online. Association for Computational Linguistics.

Semantically Consistent Data Augmentation for Neural Machine Translation via Conditional Masked Language Model

Qiao Cheng, Jin Huang, Yitao Duan
NetEase Youdao
Beijing, China
{chengqiao, huangjin, duan}@rd.netease.com

Abstract

This paper introduces a new data augmentation method for neural machine translation that can enforce stronger semantic consistency both within and across languages. Our method is based on Conditional Masked Language Model (CMLM) which is bi-directional and can be conditional on both left and right context, as well as the label. We demonstrate that CMLM is a good technique for generating context-dependent word distributions. In particular, we show that CMLM is capable of enforcing semantic consistency by conditioning on *both* source and target during substitution. In addition, to enhance diversity, we incorporate the idea of *soft* word substitution for data augmentation which replaces a word with a probabilistic distribution over the vocabulary. Experiments on four translation datasets of different scales show that the overall solution results in more realistic data augmentation and better translation quality. Our approach consistently achieves the best performance in comparison with strong and recent works and yields improvements of up to 1.90 BLEU points over the baseline. [1]

1 Introduction

Neural network models have achieved remarkable results in many fields such as computer vision, natural language processing, and speech. In order to obtain adequate expressivity, the models usually come with a large number of parameters. However, such models are prone to overfitting if trained with an insufficient amount of training data. Data Augmentation (DA) is an effective technique that has been used in many areas to augment existing labeled data and boost the performance of machine learning models. For example, in computer vision, training data is often augmented by ways such as horizontal flipping, random cropping, tilting, and color shifting (Krizhevsky et al., 2012; Cubuk et al., 2018). While DA has become a standard technique to train deep networks for image processing, it is relatively under-explored in Natural Language Processing (NLP).

The exact mechanisms and theoretical foundations of data augmentation are still under investigation. Most studies show empirically that data augmentation is effective and provide some intuitive explanations. A recent work in the field of vision (Gontijo-Lopes et al., 2020) demonstrates that affinity (the distributional shift caused by DA) and diversity (the complexity of the augmentation) can predict the performance of data augmentation methods. However, neither metric can be measured without completing the entire DA process. Therefore, it is still challenging to evaluate the goodness of a DA technique without full-fledged experimentation and it is not clear how the result can be used to guide the design of data augmentation schemes.

Generally speaking, DA can be classified into two categories. The first tries to produce realistic samples that resemble the inherent semantics of naturally generated data. In areas such as computer vision, this is often achieved via heuristics that mimic the intrinsic processes that could have actually happened in the physical world, such as photometric noise, flipping, and scaling, etc. The second perturbs the data in a stochastic fashion, resulting in unrealistic samples. Some (e.g., Bishop (1995)) interpret this as a type of regularization that boosts model performance by reducing overfitting. Both are being exploited in NLP.

This paper focuses on lexical replacement methods that augment the training data by altering existing sentences in the parallel corpus of a neural machine translation (NMT) system. We have observed frequently in practice, as well as in literature (Gao et al., 2019; Fadaee et al., 2017; Kobayashi, 2018; Wu et al., 2019; Dong et al., 2021; Liu et al., 2021), that augmented data samples that preserve

[1] Our code is available at https://github.com/netease-youdao/cmlm_da.

Proceedings of the 29th International Conference on Computational Linguistics, pages 5148–5157
October 12–17, 2022.

the semantics of the real labeled data increase the effective training size and are beneficial for model performance. We call this property *semantic consistency*. In the case of NMT, the training data comes in the form of a collection of <source, target> sentence pairs where source is a sentence in the source language and target its translation in the target language. Semantic consistency requires that (1) both source and target are fluent and grammatically correct in their respective languages; and (2) target is a high quality translation of source.

German	Es ist ja ganz **angenehm**, in eine kleine Klasse zu kommen.
English	You know, it's very **pleasant** to walk into a small class.
Case 1	You know, it's very **please** to walk into a small class.
Case 2	You know, it's very **uncomfortable** to walk into a small class.
Case 3	You know, it's very **enjoyable/comfortable** to walk into a small class.

Table 1: Data augmentation examples with varying degrees of semantic consistency.

Existing methods augment the training data using word swapping, removal or substitution (Artetxe et al., 2017; Lample et al., 2017) on either source or target, or both. Due to the discrete nature of language, these transformations are not always semantic-preserving. Quite often they either weaken the fluency of source or/and target, or break their relationships. To illustrate, consider the example given in Table 1 that shows a sentence pair from an English-German parallel corpus. Case 1 to 3 are three synthetic English sentences generated by some DA processes. Both Case 1 and 2 are undesirable because the former, although substituting the word **pleasant** with a word close in meaning, is grammatically incorrect, whereas the latter is not a good translation of the German sentence. Case 3, on the other hand, is a good augmentation that satisfies the two requirements of semantic consistency.

1.1 Our Contributions

To achieve better augmentation, the generation process must make better use of context and label. In this paper, we introduce Conditional Masked Language Model (CMLM) (Wu et al., 2019; Ghazvininejad et al., 2019; Chen et al., 2020) to data augmentation for NMT. A Masked Language Model can make use of both left and right context, and a CMLM is an enhanced version that can be conditional on more information. CMLM has been used successfully in tasks such as text classification (Wu et al., 2019). However, to the best of our knowledge, its application to text generation, especially using deep bidirectional models such as BERT (Devlin et al., 2019), has not been explored. We demonstrate in this paper that CMLM is a good technique for generating context-dependent word distributions. In particular, we show that CMLM is capable of enforcing semantic consistency by conditioning on *both* source and target during substitution. In addition, to enhance diversity, we combine the *soft* word substitution approach for DA, which replaces a word with a probabilistic distribution over the vocabulary (Gao et al., 2019). Experiments on four translation datasets of different scales show that the overall solution results in more realistic data augmentation and better translation quality. Our approach consistently achieves the best performance in comparison with strong and recent works and yields improvements of up to 1.90 BLEU points over the baseline.

In addition, we introduce an unsupervised method to measure semantic consistency without full-fledged training of NMT models, which may take many days even on GPU clusters. This could be used to provide an efficient early assessment of a data augmentation scheme.

2 Related Work

From a technical perspective, previous work on data augmentation for NLP can be classified as either context-independent or context-dependent. Context-independent approaches often apply predetermined, easy-to-compute transformations that depend solely on the word or sentence to be altered. Not surprisingly, most of them are not semantically consistent. Wei and Zou (2019) improves performance on many text classification tasks through a set of word level random perturbation operations, including random insertion, deletion, and swapping. Similar ideas have been applied to NMT, but the methods differ in how and what to alter. *Swap* (Artetxe et al., 2017; Lample et al., 2017) randomly swaps words in nearby positions within a window size k and *Drop* (Iyyer et al., 2015;

Lample et al., 2017) randomly drops word tokens. *Blank* (Xie et al., 2017) replaces the candidate word with a placeholder token and *Smooth* (Xie et al., 2017) replaces it with a word sampled from the frequency distribution of the vocabulary, showing that data noising is an effective regularizer for NMT. *SwitchOut*, introduced in (Wang et al., 2018), formulates the design of a DA algorithm as an optimization problem that maximizes an objective that encourages two desired properties: smoothness and diversity. *SwitchOut* independently replaces words in both `source` and `target` by other words uniformly sampled from their respective vocabularies. Others try to preserve a certain level of semantic consistency by replacing words with their synonyms selected from a handcrafted ontology such as WordNet (Zhang et al., 2015) or words based on similarity calculation (Wang and Yang, 2015).

These works do not make use of important context and label information and, in practice, usually cause a very small or even negative impact on performance. Context-dependent approaches, on the other hand, modify words, phrases, or the whole sentence based on their contextual information that is usually modeled using neural networks. We summarize a few representative ones below.

Fadaee et al. (2017) propose a simple but effective approach to augment the training data of NMT for low-resource language pairs. Their work uses shallow LSTM language models (LM) trained on large amounts of monolingual data to first substitute a word in `source`, and then put the corresponding translation in `target`, using automatic word alignments and the traditional statistical MT practice. LM_{sample} (Kobayashi, 2018) proposes contextual augmentation for text classification by offering a wide range of substitute words, which are predicted by a label-conditional bidirectional language model. Wu et al. (2019) retrofit BERT to conditional BERT that allows it to augment sentences without breaking the label-compatibility. The BERT-based solution brings two benefits. First, BERT's Transformer core provides a more structured memory for handling long-term dependencies in text. Second, BERT, as a deep bidirectional model, is strictly more powerful than the shallow concatenation of left-to-right and right-to-left models.

A recent work (Liu et al., 2021) treats a translation language model as a causal model and performs data augmentation by counterfactual-based causal inference. Their DA replaces source phrases according to a masked language model and the aligned target phrase by a cross-lingual language model (XLM) (Conneau and Lample, 2019) conditional on the changed source phrase.

Different from their work, we use two separate CMLMs to augment source and target respectively, which means that, instead of model prediction, the condition is always true information for both CMLMs. We show its superiority in section 5.1.

The way we incorporate augmented data into the NMT training is drawn from the idea of "soft" word introduced by *SCA* (Gao et al., 2019). Basically, the embedding of a chosen word in a sentence is replaced by its probabilistic distribution predicted by a language model. This brings in more diversity to the DA process. However, Gao et al. (2019) as a DA solution is based on a uni-directional language model and is *not* label-conditional. As we show in section 4.3 that this is less optimal.

3 Approach

In this section, we present our method in detail. We first introduce conditional MLM, then we show how to apply CMLM to data augmentation in neural machine translation tasks.

Let (X, Y) be a pair of source and target sentences where $X = (x_1, x_2, \ldots, x_M)$ and $Y = (y_1, y_2, \ldots, y_N)$ are two sequences of tokens in source and target languages, with lengths M and N, respectively. A neural machine translation system learns the conditional probability $p(y_1, y_2, \ldots, y_N | x_1, x_2, \ldots, x_M)$.

3.1 Conditional MLM

Recall that our goal is to augment NMT's parallel corpus with synthesized data that preserves the semantics within source and target sentences, as well as their cross-lingual relations. To this end, we resort to Conditional MLM for generating context-dependent word distributions, with which we then find the best substitutes for a given word. CMLM is a variation of MLM, which allows further fine-tuning of the pre-trained model. It makes the strong assumption that the masked tokens are conditionally independent of each other given the context and predicts the probabilities individually (Ghazvininejad et al., 2019).

In our case, we apply the following two practices when instantiating our CMLMs:

- We condition the CMLM on *both* X and Y.

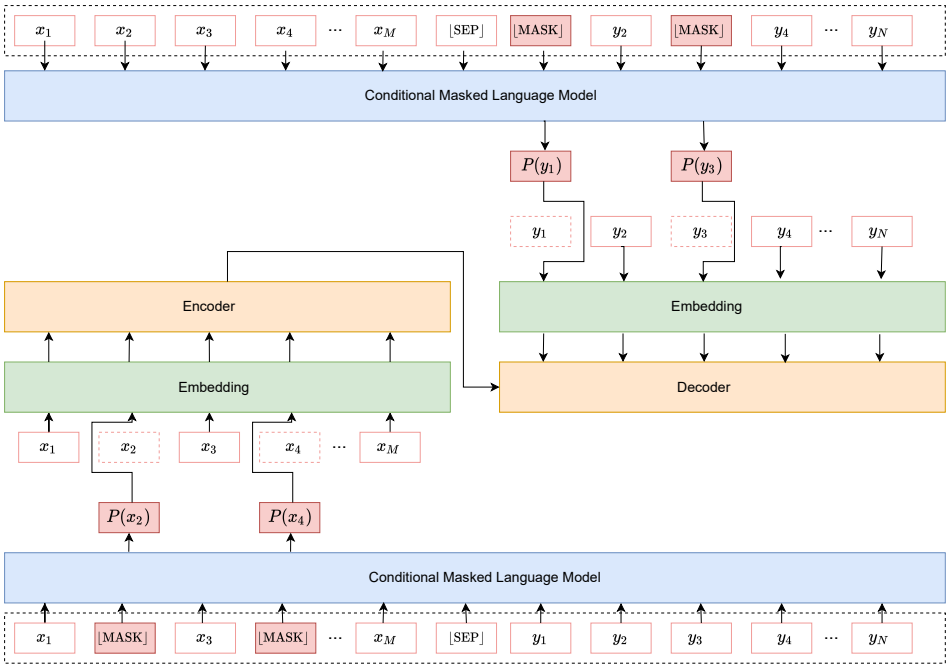

Figure 1: The architecture of our CMLM-based soft contextual data augmentation approach.

- During the training of a CMLM, we only mask out tokens in either X, or Y, but not both.

We call this approach "**Conditioning on Both but Predicting One**", referring to how it treats the source and target sides in the NMT training. Specifically, for each sentence pair (X, Y), we first concatenate X and Y, then randomly mask 15% of the words in X, and then train a CMLM to predict the masked words:

$$P\left(x_1^m, \ldots, x_i^m \mid X^u, Y\right) \quad (1)$$

where x_i^m denotes a masked token and X^u the unmasked ones within X. For the tokens in the target sentence, we train a *separate* CMLM to get their distribution similarly:

$$P\left(y_1^m, \ldots, y_i^m \mid X, Y^u\right) \quad (2)$$

During the training of an NMT model, both X and Y are available. Conditioning on the reference sentence Y allows the model to enforce stronger consistency between input and label, resulting in meaningful translations when applied to DA in NMT. We show in section 5.1, using metrics developed for translation quality estimation, that this choice significantly improves the translation quality of the generated sentence pairs.

Changing X or Y but not both for DA is a deliberate choice. Typical modern languages have diverse vocabularies, with synonyms and semantically equivalent or close expressions. This already provides abundant opportunities for semantic-preserving transformations. Therefore, it is not necessary to alter X and Y simultaneously. In section 5.1, we compare our choice with an XLM (cross-lingual language model) (Conneau and Lample, 2019) approach which changes X and Y simultaneously. The empirical study shows that our approach can avoid introducing incorrect `<source, target>` pairs and improve NMT performance.

3.2 Soft Conditional Contextual DA

Once a CMLM is trained, one could use it to expand training data for NMT. This is typically done by replacing words with others predicted by the language model at the corresponding positions (e.g., Kobayashi (2018); Wu et al. (2019)). In our case, since the probability distribution of the masked words $P\left(x_1^m, \ldots, x_i^m \mid X^u, Y\right)$, or $P\left(y_1^m, \ldots, y_i^m \mid X, Y^u\right)$ if we mask out words in Y, contains information from both backward and forward contexts, as well as target sentence, sampling from such distribution could potentially generate better substitutions for the word on the masked position. However, such a method could be expensive: to generate enough samples with adequate variation, exponentially many candidates have to be processed.

Instead, inspired by Gao et al. (2019), we take

a *soft* approach. In essence, this method works directly with the word embeddings and uses the expectation of a word's embedding over the CMLM's output distribution to replace its original embedding. Let w be a candidate word and $P(w)$ its distribution defined by the CMLM. Note that $P(w)$ is conditional on the context that we described earlier and is over the entire vocabulary. Suppose E is the embedding matrix of all the $|V|$ words. We use E_W to denote the embedding vector of a word W. The embedding of the soft word w is:

$$e_w = \mathbb{E}_{W \sim P(w)}[E_W] = \sum_{j=0}^{|V|} p_j(w) E_j \quad (3)$$

3.3 NMT Training with DA

In this section, we elaborate on the training process of the NMT model with our DA method. Figure 1 shows the architecture of the scheme. There are two independently trained CMLMs, one for augmenting the encoder, and the other the decoder. The two CMLMs can be turned on/off independently and we study the effects in section 5.2.

We use BERT (Devlin et al., 2019) as our CMLM, for its deep bidirectional natural, and superior capability for handling long-term dependencies. We start by taking a pre-trained multilingual BERT, and fine-tune it using the method described in 3.1. The NMT training proceeds as usual, except that, at each sentence pair (X, Y), for each word in X (or Y), with probability γ we replace its embedding by its soft version defined by Equation 3. Notice that, our method does *not* generate any data explicitly. Rather, we use embedding substitution to incorporate augmentation directly into the training process. We study the effect of different values of γ in section 5.3.

4 Experiments

In this section, we demonstrate the effectiveness of our method on four datasets with diverse language variation. They include three relatively small-scale datasets, {German, Spanish, Hebrew} to English ({De, Es, He}-> En) from the well-known IWSLT 2014, and one large-scale English to German (En->De) dataset from WMT14 .

4.1 Data

For IWSLT14 De->En task we follow the same pre-processing steps and the same train/dev/test split as in Gao et al. (2019). The training dataset and validation dataset contains about 160K and 7K sentence pairs, respectively. Consistent with previous work, tst2010, tst2011, tst2012, dev2010, and dev2012 are concatenated as our test data. For Es->En and He->En tasks, there are 181K and 151K parallel sentence pairs in each training set. We validate on tst2013 and test on tst2014 for these two tasks. For all IWSLT translation tasks, we use a joint source and target vocabulary with 10K byte-pair-encoding (BPE) (Sennrich et al., 2016) types. For the WMT2014 En-De translation task, again, we follow Gao et al. (2019) to filter out 4.5M sentence pairs for training. We concatenate newstest2012 and newstest2013 as the validation set and use newstest2014 as the test set. We use a joint source and target vocabulary built upon the BPE with 40k sub-word types. For fair comparison to previous work, we report tokenized BLEU (Papineni et al., 2002) scores computed with the *multi-bleu.perl* script from Moses.[2] To further boost comparability, we also report detokenized BLEU scores computed using sacreBLEU (Post, 2018). (Post, 2018). For all experiments, we performed significance tests based on bootstrap resampling introduced by Koehn (2004).

4.2 Model

Our model uses the Transformer architecture, which is solely based on attention mechanisms and dominates most of the sequence-to-sequence tasks. For all IWSLT tasks, we adopt the transformer_iwslt_de_en configuration for the NMT model. Specifically, both the encoder and decoder consist of 6 blocks, and the source and target word embedding are shared for the language pair. The dimensions of embedding and feed-forward sub-layer are set to 512 and 1024, respectively. The number of attention heads is set to 4. The default dropout rate is 0.3. For WMT14 En-De, we use the default transformer_big configuration for the NMT model. Specifically, the dimensions of embedding and feed-forward sub-layer are 1024 and 4096, respectively. The NMT models are trained by Adam (Kingma and Ba, 2015) optimizer with default learning rate schedule as Vaswani et al. (2017).

For all tasks, we adopt the BERT-base configuration for the CMLM model, except that the number of hidden layers is set to 4 to speed up the train-

[2] https://github.com/moses-smt/mosesdecoder/blob/master/scripts/generic/multi-bleu.perl

	IWSLT			WMT
	De->En	**Es->En**	**He->En**	**En->De**
Other Reported Results				
Base*	34.79	41.58	33.64	28.40
*Swap**	34.70	41.60	34.25	28.13
*Drop**	35.13	41.62	34.29	28.29
*Blank**	35.37	42.28	34.37	28.89
*Smooth**	35.45	41.69	34.61	28.97
LM_{sample}*	35.40	42.09	34.31	28.73
*SCA**	**35.78**	**42.61**	**34.91**	**29.70**
mixSeq†	**35.78**	41.39	-	29.61
Our Implementations				
Base	34.37	41.67	33.76	28.25
CMLM_{hard}	35.76	42.25	34.66	30.01
CMLM_{soft}	**35.93(+1.56)**	**42.92(+1.25)**	**35.21(+1.45)**	**30.15(+1.9)**

Table 2: BLEU scores over the test sets. (∗) from Gao et al. (2019). (†) from Wu et al. (2021)

	IWSLT			WMT
	De->En	**Es->En**	**He->En**	**En->De**
Base	33.62	40.87	33.15	27.49
CMLM_{hard}	35.07	41.45	34.01	29.08
CMLM_{soft}	**35.31(+1.69)**	**42.01(+1.14)**	**34.51(+1.36)**	**29.37(+1.88)**

Table 3: SacreBLEU scores over the test sets.

ing process. We use the bottom 4 layers of the pre-trained BERT-base-multilingual-cased model as the starting point of CMLM fine-tuning. We also experiment with an entirely randomly-initialized CMLM model and find that the pre-trained weights result in faster CMLM training. We follow Devlin et al. (2019) for the CMLM fine-tuning and use a triangular learning rate schedule with maximum learning rate η. The CMLM parameters are also updated with the Adam optimizer.

4.3 Main Results

We compare our method against several other strong data augmentation methods, including several context-independent approaches such as *Swap* (Artetxe et al., 2017; Lample et al., 2017), *Drop* (Iyyer et al., 2015; Lample et al., 2017), *Blank* (Xie et al., 2017) and *Smooth* (Xie et al., 2017), and two context-dependent ones, LM_{sample} (Kobayashi, 2018) and *SCA* (Gao et al., 2019). We also compare it against a sentence-level augmentation method, mixSeq (Wu et al., 2021), which randomly selects two sentence pairs, concatenates the source sentences and the target sentences, respectively, with a special label <sep> separating two samples, and trains the model on such augmented dataset.

Our baseline is the vanilla transformer described earlier without DA. For comparison, we performed two sets of data augmentation experiments using CMLM: (1) CMLM_{soft} uses the soft approach described in section 3.2 and follows the training framework in section 3.3. (2) CMLM_{hard} uses the conventional hard substitution approach, with the substitution words generated by sampling from the CMLMs. Both CMLM_{soft} and CMLM_{hard} augment both the encoder and the decoder, and use the same mask probability $\gamma = 0.25$, which we find to be the optimal configuration. See sections 5.2 and 5.3.

The BLEU and SacreBLEU scores on four translation tasks are presented in Table 2 and 3, respectively. Both CMLM_{soft} and CMLM_{hard} are superior to the base system, with CMLM_{soft} consistently achieves the best performance on all tasks and across all comparisons. The CMLM (soft) approach significantly outperformed the baseline in Table 2 and Table 3 for all four tasks, with p-values lower than 0.02. Most remarkably, our DA improves the baseline by as much as 1.90 BLEU points on the WMT14 En->De dataset.

In addition to experiments on publicly available corpora, we also evaluate the scheme on Youdao's

production NMT engine, [3] a major multilingual neural machine translation service that is trained with data at least three orders of magnitudes larger than the public corpora. The method achieves similar consistent improvements. Our DA mechanism has been built into the production NMT engine, serving billions of requests each day.

5 Analysis

Our method consists of multiple modules, and we design several groups of comparative experiments to analyze their effects.

5.1 Semantic Consistency

Recall that the "soft" substitution approach that we use works directly with embeddings and does not generate synthetic data explicitly. The quality of the DA process depends on the distributions defined by the two CMLMs (equations 1 and 2). There is no straightforward metric to measure the distributions in terms of semantic consistency. Here we propose a simple sampling-based approach. The intuition is: if the distribution is used for text generation, the quality of resulting sentence pairs is a good indicator of the effectiveness of its role in the DA process.

Specifically, given a sentence pair (X, Y), we randomly replace some tokens from X (resp. Y) with those sampled from the source (resp. target) CMLM, resulting in (X', Y) (resp. (X, Y')). We manually inspect a small sample and find that our method indeed produces sentence pairs that are generally both fluent in their respective languages and correct in terms of translation quality. However, our goal is to have an automatic method that can be used to assess semantic consistency at large scale. To this end, we draw on the research in Quality Estimation (QE) for Machine Translation. Self-Supervised QE aims to evaluate the quality of machine-translated sentences without human labeling, which aligns perfectly with our goal.

Zheng et al. (2021) show that the conditional probability computed by the CMLM in Equation 2 is a good indicator of translation quality (which also implies fluency). Specifically, let y^m be a word in the target, the translation quality score of this word is defined as $P\left(y^m \mid X, Y^u\right)$ as computed by the CMLM. The sentence-level quality score is simply averaging the quality scores over all target words.

[3] https://fanyi.youdao.com/

Our case is slightly different. Since we have both X and Y, we can use the idea of Zheng et al. (2021) but with a more direct approach: we can compare the words in X' (resp. Y') against the original ones in X (resp. Y) and compute the accuracy. This is equivalent to taking expectations over the test sentences.

	Source Acc	Target Acc	BLEU
MLM	53.5%	44.0%	35.56
XLM	74.8%	70.4%	35.65
CMLM	80.1%	75.5%	35.93

Table 4: The prediction accuracy of source and target, and BLEU for IWSLT14 German-English translation.

We compare our CMLM-based approach against the DA results from (1) an XLM-based scheme in Liu et al. (2021), which alters both X and Y by treating a translation language model as a causal model and performing data augmentation by counterfactual-based causal inference; and (2) a simple MLM which does not condition on any portion of Y. All implementations use models with the same configuration as the CMLM described in section 4.2, fine-tuned with the same training data but their individual conditions and objectives. Table 4 shows the prediction accuracy of masked words on the 7K IWSLT14 German-English validation data set. Consistent with the mask probability during CMLM training, we let the model predict 15% of the words in X or Y. For ease of comparing the final effects on the machine translation task, Table 4 also shows the BLEU scores measured on IWSLT14 German-English dataset after applying the DA method to the NMT engine.

Our CMLM-based solution achieves strong prediction accuracy rates of 80.1% and 75.5% on source and target sides, respectively, significantly outperforming the MLM approach by near 30 percentage points. This shows that our method is capable of generating synthetic sentence pairs with much better translation quality. The improvement over XLM is milder but still significant, with 5+ percentage points. BLEU scores follow a similar trend. Recall that we use independent CMLMs to alter either X or Y but not both, while XLM uses a single cross-lingual language model to change both. The results confirm our conjecture that altering both X and Y simultaneously while preserving semantic consistency may be too difficult for the language models. Doing so may introduce too

much noise and hurt translation quality.

This method also provides an efficient way to assess a data augmentation scheme for NMT. It can save days or even months of GPU time (for training NMT models) since computing the word prediction accuracy rates on a few thousands of sentence pairs is very fast.

5.2 Encoder vs. Decoder

Our CMLM-based data augmentation method can be applied to either encoder or decoder, or both. In this section, we conduct experiments to study the effects of these choices. We train two CMLMs independently. The first is used to augment the encoder, the latter the decoder. Note that, per our discussion in section 3.1, the CMLMs, when activated, only augment one side of the sentence pair. The encoder (resp. decoder) CMLM mask out words **only** in X (resp. Y), thus replacing their embeddings by their soft versions.

Table 5 shows the BLEU scores for different augmentation configurations. It is clear that both encoder and decoder augmentations are beneficial, with encoder augmentation obtaining slightly more gain. The maximum improvement can be achieved when the method is applied to both.

	IWSLT			**WMT**
	De-En	Es-En	He-En	En-De
Base	34.37	41.67	33.76	28.25
+Encoder	35.23	42.31	34.66	29.57
+Decoder	34.93	42.13	34.41	29.34
+Both	35.93	42.92	35.21	30.15

Table 5: BLEU scores over the test sets.

5.3 Mask Probability

As mentioned in section 3.2, for each word in X or Y, we replace its embedding by its soft version with probability γ. This parameter controls the extent to which the DA method will exert its effect. Intuitively, a small value of γ will preserve the original semantics better while a large value of γ can bring in more diversity. A balance must be struck. We experiment with different values, and Figure 2 shows their influence on BLEU on the IWSLT14 De-En dataset. The strongest performance is reached with a mask probability of 0.25.

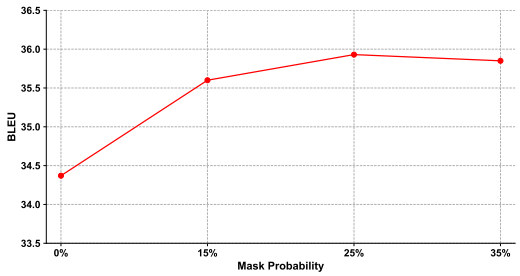

Figure 2: BLEU score for IWSLT14 German-English with difference mask probability.

5.4 Computation Overhead

Our DA method introduces two additional steps into the NMT training process: fine-tuning the CMLMs and augmenting the NMT model. The actual overhead depends on the scale of the data sets. In our experiments, IWSLT De-En and WMT En-De corpora consist of 160K and 4.5M sentence pairs, respectively. Fine-tuning the CMLMs on the two corpora takes about 3 and 20 hours, respectively, on a *single* A40 GPU.

Our training process has the same complexity as that of *SCA* (Gao et al., 2019) so they should have similar computation performance. From our experiments, the training time on IWSLT dataset increases about 84%, up from 2.5 hours to 4.6 hours, again on a *single* A40. The overhead is less significant for large corpora. The WMT tasks take 25% more time to train, up from one day to roughly 32 hours on 4 A40 cards. We see only a 10% increase in training time when we apply the DA method to our production NMT engine.

6 Conclusion

In this paper, we advocate performing semantically consistent data augmentation for neural machine translation and propose a scheme based on Conditional Masked Language Model and soft word substitution. We show that a deep, bi-directional CMLM is capable of enforcing semantic consistency by conditioning on *both* source and target during data augmentation. Experiments demonstrate that the overall solution results in more realistic data augmentation and better translation quality. Our approach consistently achieves the best performance in comparison with strong and recent works and yields improvements of up to 1.90 BLEU points over baseline.

References

Mikel Artetxe, Gorka Labaka, Eneko Agirre, and Kyunghyun Cho. 2017. Unsupervised neural machine translation. *arXiv preprint arXiv:1710.11041*.

Chris M. Bishop. 1995. Training with noise is equivalent to tikhonov regularization. *Neural Computation*, 7(1):108–116.

Yen-Chun Chen, Zhe Gan, Yu Cheng, Jingzhou Liu, and Jingjing Liu. 2020. Distilling knowledge learned in bert for text generation. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7893–7905.

Alexis Conneau and Guillaume Lample. 2019. Cross-lingual language model pretraining. *Advances in Neural Information Processing Systems*, 32:7059–7069.

Ekin D Cubuk, Barret Zoph, Dandelion Mane, Vijay Vasudevan, and Quoc V Le. 2018. Autoaugment: Learning augmentation policies from data. *arXiv preprint arXiv:1805.09501*.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, Minneapolis, Minnesota. Association for Computational Linguistics.

Xin Luna Dong, Yaxin Zhu, Zuohui Fu, Dongkuan Xu, and Gerard de Melo. 2021. Data augmentation with adversarial training for cross-lingual nli. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 5158–5167.

Marzieh Fadaee, Arianna Bisazza, and Christof Monz. 2017. Data augmentation for low-resource neural machine translation. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 567–573.

Fei Gao, Jinhua Zhu, Lijun Wu, Yingce Xia, Tao Qin, Xueqi Cheng, Wengang Zhou, and Tie-Yan Liu. 2019. Soft contextual data augmentation for neural machine translation. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 5539–5544.

Marjan Ghazvininejad, Omer Levy, Yinhan Liu, and Luke Zettlemoyer. 2019. Mask-predict: Parallel decoding of conditional masked language models. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 6112–6121, Hong Kong, China. Association for Computational Linguistics.

Raphael Gontijo-Lopes, Sylvia J Smullin, Ekin D Cubuk, and Ethan Dyer. 2020. Affinity and diversity: Quantifying mechanisms of data augmentation. *arXiv preprint arXiv:2002.08973*.

Mohit Iyyer, Varun Manjunatha, Jordan Boyd-Graber, and Hal Daumé III. 2015. Deep unordered composition rivals syntactic methods for text classification. In *Proceedings of the 53rd annual meeting of the association for computational linguistics and the 7th international joint conference on natural language processing (volume 1: Long papers)*, pages 1681–1691.

Diederik P Kingma and Jimmy Ba. 2015. Adam: A method for stochastic optimization. In *ICLR (Poster)*.

Sosuke Kobayashi. 2018. Contextual augmentation: Data augmentation by words with paradigmatic relations. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 2 (Short Papers)*, pages 452–457.

Philipp Koehn. 2004. Statistical significance tests for machine translation evaluation. In *Proceedings of the 2004 conference on empirical methods in natural language processing*, pages 388–395.

Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. 2012. Imagenet classification with deep convolutional neural networks. *Advances in neural information processing systems*, 25:1097–1105.

Guillaume Lample, Alexis Conneau, Ludovic Denoyer, and Marc'Aurelio Ranzato. 2017. Unsupervised machine translation using monolingual corpora only. *arXiv preprint arXiv:1711.00043*.

Qi Liu, Matt Kusner, and Phil Blunsom. 2021. Counterfactual data augmentation for neural machine translation. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 187–197.

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. Bleu: a method for automatic evaluation of machine translation. In *Proceedings of the 40th annual meeting of the Association for Computational Linguistics*, pages 311–318.

Matt Post. 2018. A call for clarity in reporting BLEU scores. In *Proceedings of the Third Conference on Machine Translation: Research Papers*, pages 186–191, Belgium, Brussels. Association for Computational Linguistics.

Rico Sennrich, Barry Haddow, and Alexandra Birch. 2016. Neural machine translation of rare words with subword units. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1715–1725.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in neural information processing systems*, pages 5998–6008.

William Yang Wang and Diyi Yang. 2015. That's so annoying!!!: A lexical and frame-semantic embedding based data augmentation approach to automatic categorization of annoying behaviors using# petpeeve tweets. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 2557–2563.

Xinyi Wang, Hieu Pham, Zihang Dai, and Graham Neubig. 2018. Switchout: an efficient data augmentation algorithm for neural machine translation. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 856–861.

Jason Wei and Kai Zou. 2019. Eda: Easy data augmentation techniques for boosting performance on text classification tasks. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 6382–6388.

Xing Wu, Shangwen Lv, Liangjun Zang, Jizhong Han, and Songlin Hu. 2019. Conditional bert contextual augmentation. In *International Conference on Computational Science*, pages 84–95. Springer.

Xueqing Wu, Yingce Xia, Jinhua Zhu, Lijun Wu, Shufang Xie, Yang Fan, and Tao Qin. 2021. mixSeq: A simple data augmentation methodfor neural machine translation. In *Proceedings of the 18th International Conference on Spoken Language Translation (IWSLT 2021)*, pages 192–197, Bangkok, Thailand (online). Association for Computational Linguistics.

Ziang Xie, Sida I Wang, Jiwei Li, Daniel Lévy, Aiming Nie, Dan Jurafsky, and Andrew Y Ng. 2017. Data noising as smoothing in neural network language models. *arXiv preprint arXiv:1703.02573*.

Xiang Zhang, Junbo Zhao, and Yann Lecun. 2015. Character-level convolutional networks for text classification. *Advances in Neural Information Processing Systems*, 2015:649–657.

Yuanhang Zheng, Zhixing Tan, Meng Zhang, Mieradilijiang Maimaiti, Huanbo Luan, Maosong Sun, Qun Liu, and Yang Liu. 2021. Self-supervised quality estimation for machine translation. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing (EMNLP)*. Association for Computational Linguistics.

Informative Language Representation Learning for Massively Multilingual Neural Machine Translation

Renren Jin and **Deyi Xiong** *
College of Intelligence and Computing, Tianjin University, Tianjin, China
{rrjin, dyxiong}@tju.edu.cn

Abstract

In a multilingual neural machine translation model that fully shares parameters across all languages, an artificial language token is usually used to guide translation into the desired target language. However, recent studies show that prepending language tokens sometimes fails to navigate the multilingual neural machine translation models into right translation directions, especially on zero-shot translation. To mitigate this issue, we propose two methods, language embedding embodiment and language-aware multi-head attention, to learn informative language representations to channel translation into right directions. The former embodies language embeddings into different critical switching points along the information flow from the source to the target, aiming at amplifying translation direction guiding signals. The latter exploits a matrix, instead of a vector, to represent a language in the continuous space. The matrix is chunked into multiple heads so as to learn language representations in multiple subspaces. Experiment results on two datasets for massively multilingual neural machine translation demonstrate that language-aware multi-head attention benefits both supervised and zero-shot translation and significantly alleviates the off-target translation issue. Further linguistic typology prediction experiments show that matrix-based language representations learned by our methods are capable of capturing rich linguistic typology features.[1]

1 Introduction

Multilingual neural machine translation (MNMT) (Johnson et al., 2017; Ha et al., 2016; Aharoni et al., 2019; Arivazhagan et al., 2019b), unlike bilingual machine translation with task-specific engineering (language-specific features), uses a single learning system for multiple language pairs, which is jointly trained in a mult-task learning formalism (Collobert et al., 2011). Parameter sharing across different languages in MNMT models enables transferring of intermediate representations and knowledge among languages, which makes it beneficial to machine translation of low-resource and even zero-resource languages (Firat et al., 2016b; Gu et al., 2018; Neubig and Hu, 2018; Gu et al., 2019; Zhang et al., 2020).

According to the degree of parameter sharing across languages, multilingual neural machine translation (MNMT) approaches can be categorized into two strands: full parameter sharing and partial parameter sharing (Sachan and Neubig, 2018). The former uses a unified model where both the encoder and decoder are shared for all languages. To guide the translation direction, a special token is usually prepended to the beginning of the source or target sentence to indicate the target language (Johnson et al., 2017; Fan et al., 2021). Instead of sharing all model parameters for all translation directions, the latter uses language-specific components, e.g., separate encoders (in many-to-one translation), separate decoders (in one-to-many translation), or separate cross-attention networks (Zoph and Knight, 2016; Firat et al., 2016a; Blackwood et al., 2018; Vázquez et al., 2020; Kong et al., 2021).

The key for full sharing models to distinguish target languages lies in the prepended tokens.[2] Prior studies find that the embeddings of the prepended tokens encode typological properties of languages (Östling and Tiedemann, 2017; Malaviya et al., 2017; Bjerva et al., 2019; Oncevay et al., 2020). Hence the token embeddings are also referred to as language representations or language embeddings, guiding translation into different target languages whose typological features varies a lot. While the

*Corresponding author.

[1]The source code is publicly available at https://github.com/cordercorder/nmt-multi.

[2]When there are multiple languages on the targe side, full sharing models cannot even converge during training if no target language information is provided by the prepended tokens (Wang et al., 2019).

Proceedings of the 29th International Conference on Computational Linguistics, pages 5158–5174
October 12–17, 2022.

prepened tokens play a significant role in language-specific knowledge learning, they are not usually working appropriately, making MNMT models generate translations in wrong languages, especially in zero-shot translation. Such off-target translation issue (Zhang et al., 2020; Yang et al., 2021) implies that prepending special tokens to sentence pairs in full-sharing models is not adequate to learn sufficient language-specific information to guide the MNMT models to translate into right directions.

Partial sharing may be suitable for learning language-specific properties. Nevertheless, it suffers from a rapid growth in the number of parameters with the increase in the number of languages. Additionally, which modules should be language-specific still remains to be further explored.

In this work, we attempt to learn more informative language representations (beyond language embeddings) to improve translation quality of the MNMT models with the full parameter sharing strategy. We argue that the prepended token cannot provide sufficient target language information to guide the translation direction, which is detrimental to both supervised and zero-shot translation. We conjecture that the direction control supervision provided by the prepended token is becoming weaker as the translation information flows to deeper layers. Therefore, instead of prepending special tokens to sentences, we propose a **L**anguage **E**mbedding **E**mbodiment (LEE) strategy that embodies language embeddings at critical switching points (e.g., in-between self-attention and FFN, or self-attention and cross-attention) across layers in both the encoder and decoder along the information flow from the source to the target, so as to amplify the translation direction guiding signal, shown in Figure 1. Experiment results show a boosted translation performance compared with the standard full sharing MNMT model (Johnson et al., 2017).

Inspired by the performance improvement, we further propose **L**anguage-**A**ware Multi-Head **A**ttention (LAA) to model typological features of languages. LAA estimates a matrix, instead of a vector, as the language representation for each language, which is able to encode more language information than a single fixed-length vector that is commonly adopted in previous studies. Motivated by multi-head attention (Vaswani et al., 2017), we split the matrix along the column to learn information from different representation subspaces. In order to probe the typological properties encoded in language representations, we extract the language representations from the trained MNMT model and use them for linguistic typology prediction.

The main contributions of our work can be summarized as follows:

- We empirically show that both the supervised and zero-shot translation of the standard full-sharing Transformer model are sensitive to the ways of indicating the desired target language.
- We propose the language embedding embodiment and language-aware multi-head attention to learn informative language representations for MNMT. We verify our proposal on two public datasets for massively MNMT in the many-to-many setting. Experimental results indicate that both the supervised and zero-shot translation can significantly benefit from LAA.
- We show that the language representations learned by LAA are generalizable and informative, which suggests that a proper language modeling strategy is crucial for MNMT.

2 Related Work

Multilingual Neural Machine Translation Pioneering studies on multilingual neural machine translation mostly favor to extend the standard bilingual model to MNMT by designing language specific components (Dong et al., 2015; Luong et al., 2016; Zoph and Knight, 2016; Firat et al., 2016a), where the number of parameters grows rapidly with the number of languages. Alternatively, Johnson et al. (2017); Ha et al. (2016) propose to prepend an artificial target language token to source sentences without modifying the model architecture, which is parameter efficient and eases model design. It hence becomes the dominant approach to massively MNMT due to its simplicity and effectiveness (Aharoni et al., 2019; Arivazhagan et al., 2019b; Freitag and Firat, 2020; Rios et al., 2020; Wu et al., 2021). Despite that, it usually lags behind the bilingual counterpart on high-resource language pairs and suffers from off-target translation issue (Zhang et al., 2020) on zero-shot translation. To alleviate these issues, subsequent studies propose approaches such as increasing the model capacity by deepening the model (Zhang et al., 2020), increasing the model cardinality (Xu et al., 2021), inserting mixture-of-experts (MoE) layers (Fedus

et al., 2021), designing lightweight language specific modules (Wang et al., 2018, 2019; Blackwood et al., 2018; Bapna and Firat, 2019; Philip et al., 2020; Zhang et al., 2021; Zhu et al., 2021), reducing negative interference by clustering similar languages (Tan et al., 2019), resolving the gradient conflicts (Wang et al., 2021), dividing the model parameters into a shared part and language-specific part (Lin et al., 2021; Xie et al., 2021; Gong et al., 2021; Wang and Zhang, 2022), bridging the representation gap by data argumentation (Lin et al., 2020) and contrastive learning (Pan et al., 2021), introducing language-agnostic regularization (Arivazhagan et al., 2019a; Pham et al., 2019) and explicit word alignment supervision (Raganato et al., 2021), to name a few. However, these studies neglect the capacity bottleneck in language representations as they all resort to a language embedding with the same dimension as word embeddings to encode the information of languages whose typological features diverse a lot. In contrast, LAA adopts the matrix as the language representation to alleviate the bottleneck. Moreover, the number of additional parameters brought by LAA is manageable thus it can adapt to massive MNMT settings.

Linguistic Typology Prediction Linguistic typology mainly studies the classification of languages based on their structural properties. Our work is closely related to inferring typological features with language representations from the trained MNMT models. Previous studies demonstrate that language representations in multilingual neural models can capture cross-lingual similarities between languages (Östling and Tiedemann, 2017; Malaviya et al., 2017; Bjerva and Augenstein, 2018; Bjerva et al., 2019; Yu et al., 2021), which can potentially recover missing features in typological databases. Oncevay et al. (2020) fuse language representations in trained MNMT models with features from typological databases to enhance knowledge transfer in MNMT models. They perform typological feature prediction to analyze typological knowledge in the learned language representations. Similarly, we conduct typological feature prediction to evaluate the quality of the language representations learned in LEE and LAA.

3 Language Embedding Embodiment

The central idea for the language embedding embodiment strategy is shortening distance between the language embedding and target translation in order to enable language-embodied translation generation. In Transformer-based MNMT models, instead of only feeding the language embedding into the first layer (Conneau and Lample, 2019), we embody the language embedding into many different layers along the translation information flow from the source to the target, as illustrated in Figure 1. Since the prepended artificial token has been already in vocabulary, feeding the language embedding into other layers will not result in extra parameters. As shown in Figure 1, six candidate positions in the standard Transformer can be used to embody the language embedding. Consider that the language embedding is to be embodied in position 5. Let $\boldsymbol{y}^{lang} = (\boldsymbol{y}_1^{lang}, \boldsymbol{y}_2^{lang}, \cdots, \boldsymbol{y}_n^{lang})$ denote the input of the target language $lang$ into position 5. $\boldsymbol{W}^Q$, $\boldsymbol{W}^K$ and $\boldsymbol{W}^V$ are parameter matrices of self-attention. $\boldsymbol{E}^{lang}$ is the language embedding of language $lang$. We compute the output $\boldsymbol{z}^{lang} = (\boldsymbol{z}_1^{lang}, \boldsymbol{z}_2^{lang}, \cdots, \boldsymbol{z}_n^{lang})$ after the language embedding is emodied as follows:

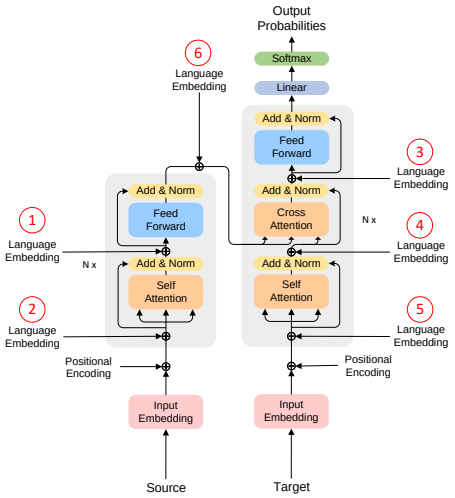

Figure 1: Illustration of the language embedding embodiment strategy. Six switching points in the standard Transformer are particularly marked to insert the language embedding.

$$e_{ij} = \frac{((\boldsymbol{y}_i^{lang} + \boldsymbol{E}^{lang})\boldsymbol{W}^Q)((\boldsymbol{y}_j^{lang} + \boldsymbol{E}^{lang})\boldsymbol{W}^K)^\mathsf{T}}{\sqrt{d}} \quad (1)$$

$$\boldsymbol{z}_i^{lang} = \sum_{j=1}^{n} \frac{\exp(e_{ij})}{\sum_{k=1}^{n} \exp(e_{ik})} (\boldsymbol{y}_j^{lang} + \boldsymbol{E}^{lang})\boldsymbol{W}^V \quad (2)$$

The language embedding embodiment for other positions can be done in a similar way: the language

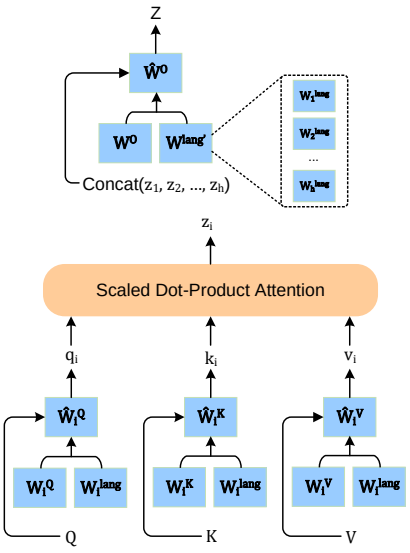

Figure 2: The architecture of the language-aware multi-head attention. $\boldsymbol{W}_i^{lang}$ is the trainable language-specific matrix of the i-th head.

embedding is added to either word embeddings or hidden states fed into the corresponding position so as to make them language-specific. The embodied language embedding is helpful for the MNMT model to distinguish words/subwords that occur in different languages with different meanings.

4 Language-Aware Multi-Head Attention

LEE adopts a fixed-length language embedding to guide the MNMT model to generate translations of the target language. However, the fixed-length language vector may not be sufficient to capture all key linguistic features and diversities in languages, which are important for translation, due to its limited representational capacity. Experiment results of LEE on both zero-shot translation and linguistic typology prediction in Section 5 have also empirically verified the capacity bottleneck in the fixed-length language embedding. To alleviate the capacity bottleneck, we further propose the language aware multi-head attention (shown in Figure 2), representing a language with a matrix, rather than a vector. We incorporate the language-specific matrix into the attention modules (self-attention in the encoder/decoder or cross-attention) since they are the essential components in Transformer. Following the strategy of multi-head attention (Vaswani et al., 2017), we split the matrix along its column to learn information from different representation subspaces of languages.

In the standard multi-head attention, three parameter matrices $\boldsymbol{W}_i^Q$, $\boldsymbol{W}_i^K$, $\boldsymbol{W}_i^V$ are used in the i-th attention head to project $\boldsymbol{Q}$ (queries), $\boldsymbol{K}$ (keys) and $\boldsymbol{V}$ (values). After projection, the scaled dot-product attention is applied on each head. The outputs from all heads are then concatenated and multiplied with $\boldsymbol{W}^O$. $\boldsymbol{W}_i^Q \in \mathbb{R}^{d_{model} \times d_k}$, $\boldsymbol{W}_i^K \in \mathbb{R}^{d_{model} \times d_k}$, $\boldsymbol{W}_i^V \in \mathbb{R}^{d_{model} \times d_v}$, $\boldsymbol{W}^O \in \mathbb{R}^{hd_v \times d_{model}}$ and h is the number of heads. For simplicity, we set $d_k = d_v = d$ and $d_{model} = hd$. In practice, the parameter matrices of all heads from $\boldsymbol{Q}$, $\boldsymbol{K}$, $\boldsymbol{V}$ are concatenated into $\boldsymbol{W}^Q$, $\boldsymbol{W}^K$, $\boldsymbol{W}^V$ respectively and the projected results are split into h heads, thus benefiting parallel computation and simplifying implementation.

In LAA, we use the matrix for language representation to relax the capacity constraint. Let matrix $\boldsymbol{W}^{lang}$ be the representation of language *lang*, where $\boldsymbol{W}^{lang} \in \mathbb{R}^{d_{model} \times d_{model}}$. $\boldsymbol{W}^{lang}$ is split into h heads along the column, which produces $\boldsymbol{W}_i^{lang} \in \mathbb{R}^{d_{model} \times d}$ for the i-th head. We add $\boldsymbol{W}_i^{lang}$ to $\boldsymbol{W}_i^Q$, $\boldsymbol{W}_i^K$, $\boldsymbol{W}_i^V$ to inject the target language information into MNMT models, which can be formulated as follows:

$$\begin{aligned} \boldsymbol{q}_i &= \boldsymbol{Q}(\boldsymbol{W}_i^Q + \boldsymbol{W}_i^{lang}) \\ \boldsymbol{k}_i &= \boldsymbol{K}(\boldsymbol{W}_i^K + \boldsymbol{W}_i^{lang}) \\ \boldsymbol{v}_i &= \boldsymbol{V}(\boldsymbol{W}_i^V + \boldsymbol{W}_i^{lang}) \end{aligned} \tag{3}$$

Let $\boldsymbol{z}_i$ be the scaled dot-product attention output of the i-th head. $\boldsymbol{z}_i$ is computed as:

$$\boldsymbol{z}_i = \text{softmax}(\frac{\boldsymbol{q}_i \boldsymbol{k}_i^\mathsf{T}}{\sqrt{d}})\boldsymbol{v}_i \tag{4}$$

We split $\boldsymbol{W}^O$ into h heads along the row and pack the outputs from different heads together as follows:

$$\boldsymbol{Z} = \sum_{i=1}^{h} \boldsymbol{z}_i(\boldsymbol{W}_i^O + (\boldsymbol{W}_i^{lang})^\mathsf{T}) \tag{5}$$

where $\boldsymbol{W}_i^O \in \mathbb{R}^{d \times d_{model}}$. As Eq. (3) and (5) show, the language representation matrix can be incorporated into the self-attention layer of the encoder/decoder or the cross-attention layer. It is optimized with other parameters to capture the typological properties of a given language. The MNMT models with the matrix-based $\boldsymbol{W}^{lang}$ learned in this way, have sufficient representation capacity to model both language-specific features (so as to alleviate the negative interference (Wang et al., 2021) between dissimilar languages) and universal features shared by all languages. Additionally, LAA

Dataset	Type	#Languages	#Supervised directions	#Zero-shot directions	#Training sentences
TED-59	English-centric	59	116	3306	5M
OPUS-100	English-centric	100	198	30	55M

Table 1: Statistics of the two datasets.

is able to scale to massively MNMT as there is only one additional matrix for each language and it introduces a very small extra computational cost in training and inference. We show several tips to implement LAA efficiently in Appendix A.

5 Experiments

We conducted two types of experiments, many-to-many translation and linguistic typology prediction, aiming at: (1) systematically evaluating the traditional language token prepending method and (2) examing the effectiveness of the proposed language embedding embodiment and language-aware multi-head attention.

5.1 Experiment Settings for Many-to-Many Translation

Dataset We used two publicly available MNMT datasets TED-59 (Qi et al., 2018) and OPUS-100 (Zhang et al., 2020). Table 1 shows the statistics of the two datasets. Due to the multi-way parallel nature of the TED-59 dataset,[3] we were able to construct 3306 (58×57) test sets for zero-shot translation directions by pairing sentences of any two languages via aligned English sentences in the original test sets. To balance training data for various language pairs, we employed the temperature-based sampling strategy (Arivazhagan et al., 2019b) with $T = 5$. More details are in Appendix B.

Model & Training We built our MNMT models on the Transformer base. More detailed settings are shown in Appendix C.

Evaluation We adopted case-sensitive detokenized sacreBLEU[4] (Post, 2018) as the evaluation metric. BLEU scores were averaged over test sets. Following Zhang et al. (2020), we used LangAcc as a complementary evaluation metric for zero-shot translation, which calculates the proportion of the translations in the desired target language. More details about the evaluation settings are in Appendix D.

5.2 Experiment Settings for Linguistic Typology Prediction

Dataset We employed typological features from URIEL typological database (Littell et al., 2017)[5] for experiments. We mainly used syntax, phonology and phonetic inventory typological features in our work. More details are in Appendix E.

Prediction Methods We adopted k-nearest neighbors approach (k-NN) for linguistic typology prediction. More details about the prediction methods are in Appendix F.

5.3 Many-to-Many Translation Results

We conducted a series of experiments to evaluate both LEE and LAA in many-to-many translation.

5.3.1 Language Embedding Embodiment

We mainly carried out experiments on the TED-59 dataset to examine the effectiveness of LEE. Table 2 shows the experimental results. We further evaluated $\text{LEE}_{4,5}$ (the best system in terms of average BLEU on all supervised translation directions on the TED-59 dataset) on the OPUS-100 dataset. As shown in Table 2, poor LangAccs on zero-shot translation directions suggest that the majority of models suffer from the severe off-target translation issue (Zhang et al., 2020). Besides, $\text{LEE}_{4,5}$ can slightly boost the average BLEU on supervised translation directions on both datasets. Findings from Table 2 can be summarized as follows:

- The ways to indicate the desired target language to MNMT models can significantly affect the translation performance, especially on zero-shot translation directions.
- The performance gap between $\text{Token}_{\text{src}}$ and $\text{Token}_{\text{tgt}}$ on zero-shot translation directions

[3] Each English sentence may have multiple corresponding translations in different languages.

[4] For TED-59, the signature is: BLEU+case.mixed+numrefs.1+smooth.exp+tok.{13a,zh,ja-mecab-0.996}+version.1.5-.1, tok.zh and tok.ja-mecab-0.996 are only for Chinese and Japanese respectively. For OPUS-100, We adopted the same signature as previous work to be comparable with them: BLEU+case.mixed+numrefs.1+smooth.exp+tok.13a+version.1.4.1

[5] https://www.cs.cmu.edu/~dmortens/projects/07_project

ID	Dataset	Model	#Param	En → XX		XX → En		All		Zero-shot		
				BLEU	WR	BLEU	WR	BLEU	WR	BLEU	LangAcc	WR
①	TED-59	$\text{Token}_{\text{tgt}}$	77M	20.74	*ref*	24.08	*ref*	22.41	*ref*	2.42	37.62	*ref*
②	TED-59	$\text{Token}_{\text{src}}$	77M	21.24	**91.38**	23.77	15.52	22.50	53.45	**10.50**	71.82	**98.91**
③	TED-59	LEE_{2}	77M	**21.26**	89.66	23.42	5.17	22.34	47.41	8.87	73.97	97.70
④	TED-59	$\text{LEE}_{1,2}$	77M	20.92	65.52	23.75	32.76	22.33	49.14	8.99	73.97	97.58
⑤	TED-59	LEE_{5}	77M	20.70	48.28	24.32	77.59	22.51	62.93	4.10	58.63	83.42
⑥	TED-59	$\text{LEE}_{4,5}$	77M	20.83	65.52	24.49	84.48	**22.66**	**75.00**	3.94	59.60	75.05
⑦	TED-59	$\text{LEE}_{3,4,5}$	77M	20.65	39.66	**24.56**	**93.10**	22.61	66.38	4.98	61.76	87.69
⑧	TED-59	$\text{LEE}_{4,5,6}$	77M	20.71	51.72	24.39	89.66	22.55	70.69	6.17	64.83	94.65
⑨	TED-59	$\text{LEE}_{2,5}$	77M	**21.26**	86.21	23.79	27.59	22.53	56.90	9.82	**74.44**	98.55
⑩	OPUS-100	$\text{Token}_{\text{tgt}}$	77M	23.46	*ref*	29.49	*ref*	26.47	*ref*	5.81	55.92	*ref*
⑪	OPUS-100	$\text{Token}_{\text{src}}$	77M	**24.04**	**81.91**	28.74	8.51	26.39	45.21	4.06	34.53	23.33
⑫	OPUS-100	$\text{LEE}_{4,5}$	77M	23.38	50.00	**29.64**	47.87	**26.51**	48.94	**7.37**	**65.84**	**56.67**

Table 2: Experimental results of LEE on the two datasets. En → XX and XX → En: translation directions from and to English respectively. All: all supervised translation directions. Zero-shot: zero-shot translation directions. #Param: the number of trainable parameters in the model. WR (Win Rate): the proportion of translation directions that outperform the *ref* system in terms of BLEU. LangAcc: the proportion of translations in the correct target language among all translations. $\text{Token}_{\text{tgt}}$: prepending a target language tag to the target side. $\text{Token}_{\text{src}}$: prepending a target language tag to the source side. The subscript of LEE denotes the positions to embody the language embedding as shown in Figure 1. It's worth noting that for supervised translation directions, the test sets in OPUS-100 only cover 94 language pairs and BLEU for "All" is hence averaged on 188 translation directions.

varies substantially across different datasets. We perform analysis to find the main reasons behind the performance discrepancy and the details are shown in Appendix G.

- Prepending a target language token to the source side ($\text{Token}_{\text{src}}$) and to the target side ($\text{Token}_{\text{tgt}}$) benefit En → XX and XX → En translation directions respectively.
- Embodying the target language embedding in the decoder is preferable for supervised translation directions.

Please refer to Appendix G for an in-depth analysis that supports these findings.

5.3.2 Language-Aware Multi-Head Attention

Baslines Since LAA enlarges the language representation with a language-specific matrix, introducing additional parameters than LEE, we compared it against two state-of-the-art MNMT baselines. The first is the monolingual adapter (Philip et al., 2020). We omit the bottleneck dimension and activation function in the original adapter and adopt a matrix as the adapter. The adapter is the same size as the language representation in LAA. We share the monolingual adapter across all layers to keep the number of parameters identical to LAA and jointly train it with the other components of the MNMT model. In addition to the monolingual adapter, we also compared against the combination of LALN (Language-Aware Layer Normalization) and LALT (Language-Aware Linear Transformation) proposed by Zhang et al. (2020).

Table 3 shows the detailed results. We observe that incorporating LAA into the decoder benefits both supervised and zero-shot translation (①vs. ③⑤)(②vs. ④)(⑫vs. ⑬⑭), while incorporating LAA into the self-attention layer of the encoder deteriorates translation performance ((①vs. ②)(③vs. ④), which is consistent with the finding in LEE (i.e., the incorporation into the the decoder is better than the encoder). Similarly, incorporating the monolingual adapter into the decoder is more effective than into the encoder (⑧vs. ⑨). **Moreover, LAA outperforms all strong baselines under the condition of using equal number of parameters** (⑧⑨⑩⑪vs. ③⑤), which demonstrates that LAA can capture rich language information to improve translation performance of massively MNMT model. To examine whether LAA and LEE are complementary to each other, we combine $\text{LAA}_{\text{dec.self}}$ and $\text{LEE}_{4,5}$ and the results show a slightly boosted translation performance (③vs. ⑦) (⑬vs. ⑮). However, adopting LAA while prepending the language tag to the target side leads to a slightly performance drop on supervised translation directions (③vs. ⑥).

The experiments in Table 2 and 3 are conducted

ID	Dataset	Model	#Param	En → XX		XX → En		All		Zero-shot		
				BLEU	WR	BLEU	WR	BLEU	WR	BLEU	LangAcc	WR
①	TED-59	$\text{Token}_{\text{tgt}}$	77M	20.74	*ref*	24.08	*ref*	22.41	*ref*	2.42	37.62	*ref*
②	TED-59	$\text{LAA}_{\text{enc.self}}$	92M	20.11	17.24	21.13	1.72	20.62	9.48	6.30	71.92	92.77
③	TED-59	$\text{LAA}_{\text{dec.self}}$	92M	21.29	84.48	25.14	96.55	23.21	90.52	8.94	74.68	98.58
④	TED-59	$\text{LAA}_{\text{enc.self,dec.self}}$	92M	21.04	68.97	23.10	1.72	22.07	35.34	8.89	74.03	97.58
⑤	TED-59	$\text{LAA}_{\text{dec.self,cros}}$	92M	**21.47**	89.66	24.79	94.83	23.13	92.24	**12.69**	74.57	**99.15**
⑥	TED-59	$\text{LAA}_{\text{dec.self}}$ + $\text{Token}_{\text{tgt}}$	92M	21.19	79.31	25.11	96.55	23.15	87.93	9.07	74.65	98.40
⑦	TED-59	$\text{LAA}_{\text{dec.self}}$ + $\text{LEE}_{4,5}$	92M	21.28	87.93	**25.29**	**100.00**	**23.28**	**93.97**	9.48	74.69	99.09
⑧	TED-59	$\text{Adapter}_{\text{enc}}$	92M	21.04	68.97	22.91	0.00	21.97	34.48	8.20	74.44	96.79
⑨	TED-59	$\text{Adapter}_{\text{dec}}$	92M	21.43	**91.38**	24.14	62.07	22.79	76.72	10.61	**75.36**	98.97
⑩	TED-59	$\text{Adapter}_{\text{enc,dec}}$	92M	21.10	79.31	22.95	1.72	22.03	40.52	9.17	75.10	98.09
⑪	TED-59	LALN + LALT	94M	21.37	75.86	22.85	1.72	22.11	38.79	7.69	73.18	95.58
⑫	OPUS-100	$\text{Token}_{\text{tgt}}$	77M	23.46	*ref*	29.49	*ref*	26.47	*ref*	5.81	55.92	*ref*
⑬	OPUS-100	$\text{LAA}_{\text{dec.self}}$	103M	**24.42**	**90.43**	29.78	**79.79**	27.10	**85.11**	11.23	81.53	**100.00**
⑭	OPUS-100	$\text{LAA}_{\text{dec.self,cros}}$	103M	**24.42**	89.36	29.58	68.09	27.00	78.72	**12.13**	**87.65**	**100.00**
⑮	OPUS-100	$\text{LAA}_{\text{dec.self}}$ + $\text{LEE}_{4,5}$	103M	24.41	88.30	**29.83**	76.60	**27.12**	82.45	11.31	81.70	**100.00**

Table 3: Experiment results of LAA on the two datasets. The results of ① and ⑫ are from Table 2. The subscript of LAA denotes the layers where the language representation is incorporated. The subscript of the adapter denotes the modules where the monolingual adapter is incorporated. enc/dec denote the encoder/decoder. self/cros denote self-attention/cross-attention layer. Considering that both $\text{LAA}_{\text{dec.self}}$ and $\text{LAA}_{\text{dec.self}}$ + $\text{LEE}_{4,5}$ achieve superior average BLEU scores on supervised translation directions and $\text{LAA}_{\text{dec.self,cros}}$ obtains the best BLEU on zero-shot translation, we verify them again on the OPUS-100 dataset.

under a temperature-based sampling setting of $T = 5$, which oversamples training data of low-resource language pairs. Oversampling usually leads to improvements in average BLEU on supervised translation directions (Table 3 vs. Table 4). To eliminate such effect and compare with previous work (Zhang et al., 2020; Xu et al., 2021),[6] we carried out experiments without oversampling. Table 4 summarizes our experimental results without oversampling. The results suggest that LAA yields larger BLEU improvements on supervised translation directions when training on the raw data distribution. e.g., (⑬ - ⑫ = +0.63 BLEU, Table 3) vs. (⑫ - ⑨ = +1 BLEU, Table 4). Additionally, the performance gain on zero-shot translation is also retained mostly (⑬ - ⑫ = +5.42 BLEU, Table 3) vs. (⑫ - ⑨ = +5.3 BLEU, Table 4). We enlarge the MNMT model by increasing the number of layers in both the encoder and decoder to 24 so that the number of parameters in our models is approximately the same as the largest model used by Zhang et al. (2020) and Xu et al. (2021).[7] As the model deepens, we adjusted the dropout rate to 0.2. Our methods outperform previous results (Zhang et al., 2020; Xu et al., 2021) on both supervised and zero-shot translation directions (⑥⑦⑧ vs. ⑭).[8]

5.4 Linguistic Typology Prediction Results

Similarly, we conducted experiments on the linguistic typology prediction for both LEE and LAA which are trained on the two datasets.

5.4.1 Language Embedding Embodiment

We selected language embeddings learned by $\text{Token}_{\text{src}}$, $\text{Token}_{\text{tgt}}$ and $\text{LEE}_{4,5}$ to perform linguistic typology prediction given that $\text{Token}_{\text{src}}$ and $\text{Token}_{\text{tgt}}$ are the dominant practice in MNMT and $\text{LEE}_{4,5}$ achieves fairly good performance over all supervised translation directions across the two datasets. We present the experimental results in Table 6 (Appendix H). Optimal ks for $\text{Token}_{\text{src}}$, $\text{Token}_{\text{tgt}}$ and $\text{LEE}_{4,5}$ are not always consistent and it is difficult to conclude the most superior one. We therefore adopt the maximum accuracy under different settings of k as the evaluation metric. We observe that there is no significant performance

[6] As the two studies are carried out on the OPUS-100 dataset without oversampling, we remove oversampling to make a fair comparison with them.

[7] Although Xu et al. (2021) don't report the exact number of parameters, the number of parameters in their model is approximately equal to that of Zhang et al. (2020) as stated in paper.

[8] Yang et al. (2021) and Wang et al. (2022) report better results than us. However, Yang et al. (2021) oversampled the training data for low-resource language pairs and removed five language pairs without test data. We removed oversampling and used all training data of the OPUS-100 dataset, hence the model in our work needs to schedule its capacity over the extra ten translation directions. Wang et al. (2022) conducted experiments with the MNMT model of up to 3.8B parameters, which is several times larger than ours. To the best of our knowledge, there are no other results superior to ours except theirs.

ID	Dataset	Model	#Param	En → XX		XX → En		All		Zero-shot		
				BLEU	WR	BLEU	WR	BLEU	WR	BLEU	LangAcc	WR
①	TED-59	$\text{Token}^{-}_{\text{tgt}}$	77M	19.54	*ref*	24.23	*ref*	21.89	*ref*	2.84	40.94	*ref*
②	TED-59	$\text{Token}^{-}_{\text{src}}$	77M	20.25	**96.55**	23.65	8.62	21.95	52.59	9.65	65.45	96.77
③	TED-59	$\text{LEE}^{-}_{4,5}$	77M	19.56	51.72	24.42	86.21	21.99	68.97	6.43	66.25	87.57
④	TED-59	$\text{LAA}^{-}_{\text{dec.self}}$	92M	20.64	**96.55**	**25.16**	**98.28**	**22.90**	**97.41**	9.93	75.04	97.67
⑤	TED-59	$\text{LAA}^{-}_{\text{dec.self}}$ + $\text{LEE}^{-}_{4,5}$	92M	**20.67**	94.83	25.05	93.10	22.86	93.97	**10.02**	**75.26**	**97.82**
⑥	OPUS-100	Zhang et al. (2020)	254M	23.96	-	31.36	-	27.66	-	5.24	47.91	-
⑦	OPUS-100	Zhang et al. (2020)	254M	23.36	-	30.98	-	27.17	-	14.08	**87.68**	-
⑧	OPUS-100	Xu et al. (2021)	-	**24.17**	-	**32.19**	-	**28.18**	-	**14.71**	-	-
⑨	OPUS-100	$\text{Token}^{-}_{\text{tgt}}$	77M	21.82	*ref*	28.45	*ref*	25.14	*ref*	6.63	58.76	*ref*
⑩	OPUS-100	$\text{Token}^{-}_{\text{src}}$	77M	22.15	74.47	27.68	10.64	24.91	42.55	4.91	37.80	30.00
⑪	OPUS-100	$\text{LEE}^{-}_{4,5}$	77M	21.49	45.74	28.48	43.62	24.98	44.68	10.08	79.90	96.67
⑫	OPUS-100	$\text{LAA}^{-}_{\text{dec.self}}$	103M	23.57	91.49	28.71	70.21	26.14	80.85	11.93	80.39	**100.00**
⑬	OPUS-100	$\text{LAA}^{-}_{\text{dec.self}}$ + $\text{LEE}^{-}_{4,5}$	103M	23.69	91.49	28.88	78.72	26.29	85.11	12.77	**85.00**	**100.00**
⑭	OPUS-100	⑬ + 24 layers	236M	**26.82**	**98.94**	**32.31**	**100.00**	**29.56**	**99.47**	**15.08**	84.59	**100.00**

Table 4: Experimental results without oversampling on the two datasets. The superscript "-" denotes that oversampling is removed.

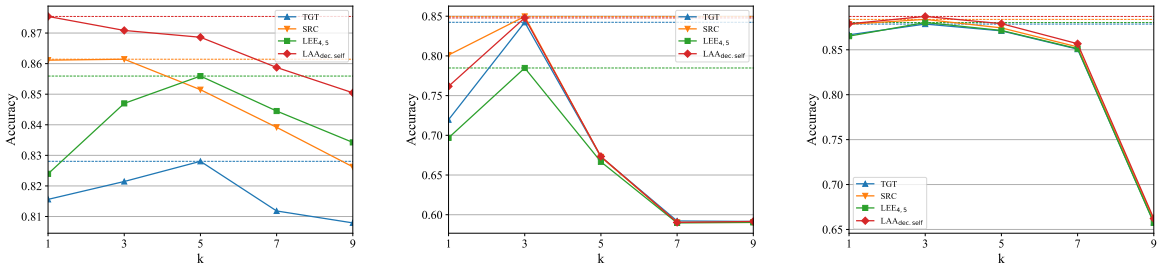

(a) Syntax (b) Phonology (c) Phonetic inventory

Figure 3: Prediction accuracy on syntax, phonology and phonetic inventory features using the language embeddings learned by $\text{Token}_{\text{tgt}}$, $\text{Token}_{\text{src}}$, $\text{LEE}_{4,5}$ and $\text{LAA}_{\text{dec.self}}$ which are trained on the TED-59 dataset.

gap except inferring the phonology features with the language embeddings trained on the TED-59 dataset (⑨, Table 6 in Appendix H).

5.4.2 Language Aware Multi-Head Attention

Figure 3 and 5 (the latter shown in Appendix I) show the prediction results of using the language representation learned by LAA for linguistic typology probing. We observe that the language representation learned in LAA achieves the best accuracy on syntax feature inference across the two datasets, which demonstrates its generalization ability to some extent. For phonology and phonetic inventory feature inference, there are no significant differences among the language representations learned in various MNMT models. We hypothesize that typological properties encoded in language representations are relied on the task that learns language representations. The translation task can force MNMT models to learn syntactic information to accommodate syntactic divergences across languages for better translation. Additionally, the superior performance of LAA on many-to-many translation and linguistic typology prediction suggests that the language representations learned in LAA can benefit multilingual translation.

5.5 Effect of the Increased Parameters Introduced by LAA

We carried out experiments to study the effect of the increased parameters introduced by LAA. Specifically, we removed the language-specific properties of the introduced trainable matrices in LAA by randomly activating them with equal probabilities during training and inference. We denote the MNMT models with the random matrices as LAA^{R}. As these matrices become random, there is no explicit signal to guide the translation into the desired target languages. Hence we followed the prepending token strategies of $\text{Token}_{\text{tgt}}$ and $\text{Token}_{\text{src}}$ to navi-

ID	Dataset	Model	#Param	En → XX		XX → En		All		Zero-shot		
				BLEU	WR	BLEU	WR	BLEU	WR	BLEU	LangAcc	WR
①	TED-59	$\text{Token}_{\text{tgt}}$	77M	20.74	*ref*	24.08	*ref*	22.41	*ref*	2.42	37.62	*ref*
②	TED-59	$\text{Token}_{\text{src}}$	77M	21.24	**91.38**	23.77	15.52	22.50	53.45	**10.50**	71.82	98.91
③	TED-59	$\text{LAA}_{\text{dec.self}}$	92M	**21.29**	84.48	25.14	96.55	23.21	90.52	8.94	74.68	98.58
④	TED-59	$\text{LAA}_{\text{dec.self}}$ + $\text{LEE}_{4,5}$	92M	21.28	87.93	25.29	**100.00**	23.28	**93.97**	9.48	**74.69**	**99.09**
⑤	TED-59	$\text{LAA}^{R}_{\text{dec.self}}$ + $\text{Token}_{\text{src}}$	92M	20.74±0.02	44.14±3.78	25.94±0.01	98.28±0.00	**23.34±0.01**	71.21±1.89	7.08±0.00	55.73±0.01	95.50±0.09
⑥	TED-59	$\text{LAA}^{R}_{\text{dec.self}}$ + $\text{Token}_{\text{tgt}}$	92M	20.18±0.01	10.34±1.22	**26.25±0.02**	98.28±0.00	23.22±0.01	54.31±0.61	0.92±0.00	6.76±0.00	2.75±0.09
⑦	OPUS-100	$\text{Token}_{\text{tgt}}$	77M	23.46	*ref*	29.49	*ref*	26.47	*ref*	5.81	55.92	*ref*
⑧	OPUS-100	$\text{Token}_{\text{src}}$	77M	24.04	81.91	28.74	8.51	26.39	45.21	4.06	34.53	23.33
⑨	OPUS-100	$\text{LAA}_{\text{dec.self}}$	103M	**24.42**	**90.43**	29.78	**79.79**	27.10	**85.11**	11.23	81.53	**100.00**
⑩	OPUS-100	$\text{LAA}_{\text{dec.self}}$ + $\text{LEE}_{4,5}$	103M	24.41	88.30	**29.83**	76.60	**27.12**	82.45	**11.31**	**81.70**	**100.00**
⑪	OPUS-100	$\text{LAA}^{R}_{\text{dec.self}}$ + $\text{Token}_{\text{src}}$	103M	23.65±0.02	64.68±3.79	28.61±0.01	12.98±0.89	26.13±0.01	38.83±1.50	4.18±0.01	39.94±0.04	20.00±0.00
⑫	OPUS-100	$\text{LAA}^{R}_{\text{dec.self}}$ + $\text{Token}_{\text{tgt}}$	103M	22.80±0.02	16.81±1.39	29.27±0.01	22.13±2.43	26.04±0.01	19.47±1.22	6.16±0.01	68.00±0.11	50.00±2.36

Table 5: Experiment results of $\text{LAA}^{R}_{\text{dec.self}}$ on the two datasets. The superscript "R" denotes that the language-specific matrices in $\text{LAA}_{\text{dec.self}}$ are stochastically activated during training and inference. We adopted $\text{Token}_{\text{tgt}}$ or $\text{Token}_{\text{src}}$ with $\text{LAA}^{R}_{\text{dec.self}}$ to guide the MNMT models into the right translation directions. The results of ①②⑦⑧ are from Table 2. The results of ③④⑨⑩ are from Table 3.

gate the translation directions.[9] We incorporated the stochastic matrices into the self-attention modules of the decoder since $\text{LAA}_{\text{dec.self}}$ achieves superior performance on both supervised and zero-shot translation directions as shown in Table 3 and 4. For efficiency, the best checkpoint was selected according to the average BLEU on the validation sets with only one random seed. We evaluated models on the test sets with the best checkpoint and report the mean and standard deviation of the BLEU score with 5 different random seeds. Results with the temperature-based sampling strategy ($T = 5$) on the two datasets are shown in Table 5. We also conducted experiments on the raw data distribution and results are presented in Table 7 of Appendix J.

We observe that the performance of $\text{LAA}^{R}_{\text{dec.self}}$ with different random seeds is stable as the standard deviations on various metrics are small. Additionally, the performance gap between $\text{LAA}_{\text{dec.self}}$ and $\text{LAA}^{R}_{\text{dec.self}}$ is small on supervised translation directions on the TED-59 dataset. Despite that, $\text{LAA}_{\text{dec.self}}$ outperforms $\text{LAA}^{R}_{\text{dec.self}}$ by a large margin on the OPUS-100 dataset. We hypothesize that the TED-59 dataset covers more restrict domains than the OPUS-100 dataset sampled from OPUS collection (Tiedemann, 2012),[10] which may benefit $\text{LAA}^{R}_{\text{dec.self}}$ as the knowledge from similar domains can be easily transferred across the stochastic matrices during training. Furthermore, $\text{LAA}^{R}_{\text{dec.self}}$ consistently lags behind $\text{LAA}_{\text{dec.self}}$ on zero-shot translation directions on the two datasets. **Given the same number of parameters in $\text{LAA}^{R}_{\text{dec.self}}$ and $\text{LAA}_{\text{dec.self}}$, we can conclude that the performance improvement of $\text{LAA}_{\text{dec.self}}$ is not only due to the parameter increasement.**

6 Conclusion

To improve massively MNMT, we have presented two approaches to learning informative language representations, language embedding embodiment and language-aware multi-head attention. We find that the ways to inject target language information into MNMT models have a significant impact on the translation performance, especially on zero-shot translation. We validate the effectiveness of the two approaches on two public datasets. We probe the typological features encoded in language representations learned by LEE and LAA through linguistic typology feature prediction. The superior prediction performance of the matrix-based language representations learned by LAA on syntax features demonstrates its informativeness.

Acknowledgements

The present research was supported by the Key Research and Development Program of Yunnan Province (Grant No. 202203AA080004-2). We would like to thank the anonymous reviewers for their insightful comments.

[9] We also attempted to make these stochastic matrices language-specific by gradually increasing the sampling probability on a specific matrix for each language during training but results are not satisfactory.

[10] There are various corpora in the OPUS collection, such as Wikipedia corpus, Bible corpus, UN corpus and TED corpus, etc. As the OPUS-100 dataset is constructed by randomly sampling sentence pairs from the corpora of OPUS collection for each language pair, it may cover more diverse domains.

References

Roee Aharoni, Melvin Johnson, and Orhan Firat. 2019. Massively multilingual neural machine translation. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT 2019, Minneapolis, MN, USA, June 2-7, 2019, Volume 1 (Long and Short Papers)*, pages 3874–3884. Association for Computational Linguistics.

Naveen Arivazhagan, Ankur Bapna, Orhan Firat, Roee Aharoni, Melvin Johnson, and Wolfgang Macherey. 2019a. The missing ingredient in zero-shot neural machine translation. *CoRR*, abs/1903.07091.

Naveen Arivazhagan, Ankur Bapna, Orhan Firat, Dmitry Lepikhin, Melvin Johnson, Maxim Krikun, Mia Xu Chen, Yuan Cao, George F. Foster, Colin Cherry, Wolfgang Macherey, Zhifeng Chen, and Yonghui Wu. 2019b. Massively multilingual neural machine translation in the wild: Findings and challenges. *CoRR*, abs/1907.05019.

Ankur Bapna and Orhan Firat. 2019. Simple, scalable adaptation for neural machine translation. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing, EMNLP-IJCNLP 2019, Hong Kong, China, November 3-7, 2019*, pages 1538–1548. Association for Computational Linguistics.

Johannes Bjerva and Isabelle Augenstein. 2018. From phonology to syntax: Unsupervised linguistic typology at different levels with language embeddings. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT 2018, New Orleans, Louisiana, USA, June 1-6, 2018, Volume 1 (Long Papers)*, pages 907–916. Association for Computational Linguistics.

Johannes Bjerva, Robert Östling, Maria Han Veiga, Jörg Tiedemann, and Isabelle Augenstein. 2019. What do language representations really represent? *Comput. Linguistics*, 45(2):381–389.

Graeme W. Blackwood, Miguel Ballesteros, and Todd Ward. 2018. Multilingual neural machine translation with task-specific attention. In *Proceedings of the 27th International Conference on Computational Linguistics, COLING 2018, Santa Fe, New Mexico, USA, August 20-26, 2018*, pages 3112–3122. Association for Computational Linguistics.

Ronan Collobert, Jason Weston, Léon Bottou, Michael Karlen, Koray Kavukcuoglu, and Pavel P. Kuksa. 2011. Natural language processing (almost) from scratch. *J. Mach. Learn. Res.*, 12:2493–2537.

Alexis Conneau and Guillaume Lample. 2019. Cross-lingual language model pretraining. In *Advances in Neural Information Processing Systems 32: Annual Conference on Neural Information Processing Systems 2019, NeurIPS 2019, December 8-14, 2019, Vancouver, BC, Canada*, pages 7057–7067.

Daxiang Dong, Hua Wu, Wei He, Dianhai Yu, and Haifeng Wang. 2015. Multi-task learning for multiple language translation. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing of the Asian Federation of Natural Language Processing, ACL 2015, July 26-31, 2015, Beijing, China, Volume 1: Long Papers*, pages 1723–1732. The Association for Computer Linguistics.

Matthew S. Dryer and Martin Haspelmath, editors. 2013. *WALS Online*. Max Planck Institute for Evolutionary Anthropology, Leipzig.

Angela Fan, Shruti Bhosale, Holger Schwenk, Zhiyi Ma, Ahmed El-Kishky, Siddharth Goyal, Mandeep Baines, Onur Celebi, Guillaume Wenzek, Vishrav Chaudhary, Naman Goyal, Tom Birch, Vitaliy Liptchinsky, Sergey Edunov, Michael Auli, and Armand Joulin. 2021. Beyond english-centric multilingual machine translation. *J. Mach. Learn. Res.*, 22:107:1–107:48.

William Fedus, Barret Zoph, and Noam Shazeer. 2021. Switch transformers: Scaling to trillion parameter models with simple and efficient sparsity. *CoRR*, abs/2101.03961.

Orhan Firat, Kyunghyun Cho, and Yoshua Bengio. 2016a. Multi-way, multilingual neural machine translation with a shared attention mechanism. In *NAACL HLT 2016, The 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, San Diego California, USA, June 12-17, 2016*, pages 866–875. The Association for Computational Linguistics.

Orhan Firat, Baskaran Sankaran, Yaser Al-Onaizan, Fatos T. Yarman-Vural, and Kyunghyun Cho. 2016b. Zero-resource translation with multi-lingual neural machine translation. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing, EMNLP 2016, Austin, Texas, USA, November 1-4, 2016*, pages 268–277. The Association for Computational Linguistics.

Markus Freitag and Orhan Firat. 2020. Complete multilingual neural machine translation. In *Proceedings of the Fifth Conference on Machine Translation, WMT@EMNLP 2020, Online, November 19-20, 2020*, pages 550–560. Association for Computational Linguistics.

Hongyu Gong, Xian Li, and Dmitriy Genzel. 2021. Adaptive sparse transformer for multilingual translation. *CoRR*, abs/2104.07358.

Jiatao Gu, Hany Hassan, Jacob Devlin, and Victor O. K. Li. 2018. Universal neural machine translation for extremely low resource languages. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics:*

Human Language Technologies, NAACL-HLT 2018, New Orleans, Louisiana, USA, June 1-6, 2018, Volume 1 (Long Papers), pages 344–354. Association for Computational Linguistics.

Jiatao Gu, Yong Wang, Kyunghyun Cho, and Victor O. K. Li. 2019. Improved zero-shot neural machine translation via ignoring spurious correlations. In *Proceedings of the 57th Conference of the Association for Computational Linguistics, ACL 2019, Florence, Italy, July 28- August 2, 2019, Volume 1: Long Papers*, pages 1258–1268. Association for Computational Linguistics.

Thanh-Le Ha, Jan Niehues, and Alex Waibel. 2016. Toward multilingual neural machine translation with universal encoder and decoder. In *Proceedings of the 13th International Conference on Spoken Language Translation, IWSLT 2016, Seattle, WA, USA, December 8-9, 2016*. International Workshop on Spoken Language Translation.

Harald Hammarström, Robert Forkel, Martin Haspelmath, and Sebastian Bank. 2021. *Glottolog 4.4*. Leipzig.

Melvin Johnson, Mike Schuster, Quoc V. Le, Maxim Krikun, Yonghui Wu, Zhifeng Chen, Nikhil Thorat, Fernanda B. Viégas, Martin Wattenberg, Greg Corrado, Macduff Hughes, and Jeffrey Dean. 2017. Google's multilingual neural machine translation system: Enabling zero-shot translation. *Trans. Assoc. Comput. Linguistics*, 5:339–351.

Diederik P. Kingma and Jimmy Ba. 2015. Adam: A method for stochastic optimization. In *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*.

Xiang Kong, Adithya Renduchintala, James Cross, Yuqing Tang, Jiatao Gu, and Xian Li. 2021. Multilingual neural machine translation with deep encoder and multiple shallow decoders. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume, EACL 2021, Online, April 19 - 23, 2021*, pages 1613–1624. Association for Computational Linguistics.

Taku Kudo and John Richardson. 2018. Sentencepiece: A simple and language independent subword tokenizer and detokenizer for neural text processing. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing, EMNLP 2018: System Demonstrations, Brussels, Belgium, October 31 - November 4, 2018*, pages 66–71. Association for Computational Linguistics.

Zehui Lin, Xiao Pan, Mingxuan Wang, Xipeng Qiu, Jiangtao Feng, Hao Zhou, and Lei Li. 2020. Pre-training multilingual neural machine translation by leveraging alignment information. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing, EMNLP 2020, Online, November 16-20, 2020*, pages 2649–2663. Association for Computational Linguistics.

Zehui Lin, Liwei Wu, Mingxuan Wang, and Lei Li. 2021. Learning language specific sub-network for multilingual machine translation. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing, ACL/IJCNLP 2021, (Volume 1: Long Papers), Virtual Event, August 1-6, 2021*, pages 293–305. Association for Computational Linguistics.

Patrick Littell, David R. Mortensen, Ke Lin, Katherine Kairis, Carlisle Turner, and Lori S. Levin. 2017. URIEL and lang2vec: Representing languages as typological, geographical, and phylogenetic vectors. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics, EACL 2017, Valencia, Spain, April 3-7, 2017, Volume 2: Short Papers*, pages 8–14. Association for Computational Linguistics.

Minh-Thang Luong, Quoc V. Le, Ilya Sutskever, Oriol Vinyals, and Lukasz Kaiser. 2016. Multi-task sequence to sequence learning. In *4th International Conference on Learning Representations, ICLR 2016, San Juan, Puerto Rico, May 2-4, 2016, Conference Track Proceedings*.

Chaitanya Malaviya, Graham Neubig, and Patrick Littell. 2017. Learning language representations for typology prediction. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing, EMNLP 2017, Copenhagen, Denmark, September 9-11, 2017*, pages 2529–2535. Association for Computational Linguistics.

Steven Moran and Daniel McCloy, editors. 2019. *PHOIBLE 2.0*. Max Planck Institute for the Science of Human History, Jena.

Graham Neubig and Junjie Hu. 2018. Rapid adaptation of neural machine translation to new languages. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing, Brussels, Belgium, October 31 - November 4, 2018*, pages 875–880. Association for Computational Linguistics.

Arturo Oncevay, Barry Haddow, and Alexandra Birch. 2020. Bridging linguistic typology and multilingual machine translation with multi-view language representations. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing, EMNLP 2020, Online, November 16-20, 2020*, pages 2391–2406. Association for Computational Linguistics.

Robert Östling and Jörg Tiedemann. 2017. Continuous multilinguality with language vectors. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics, EACL 2017, Valencia, Spain, April 3-7, 2017, Volume 2: Short Papers*, pages 644–649. Association for Computational Linguistics.

Myle Ott, Sergey Edunov, Alexei Baevski, Angela Fan, Sam Gross, Nathan Ng, David Grangier, and Michael Auli. 2019. fairseq: A fast, extensible toolkit for sequence modeling. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT 2019, Minneapolis, MN, USA, June 2-7, 2019, Demonstrations*, pages 48–53. Association for Computational Linguistics.

Xiao Pan, Mingxuan Wang, Liwei Wu, and Lei Li. 2021. Contrastive learning for many-to-many multilingual neural machine translation. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing, ACL/IJCNLP 2021, (Volume 1: Long Papers), Virtual Event, August 1-6, 2021*, pages 244–258. Association for Computational Linguistics.

Ngoc-Quan Pham, Jan Niehues, Thanh-Le Ha, and Alexander Waibel. 2019. Improving zero-shot translation with language-independent constraints. In *Proceedings of the Fourth Conference on Machine Translation, WMT 2019, Florence, Italy, August 1-2, 2019 - Volume 1: Research Papers*, pages 13–23. Association for Computational Linguistics.

Jerin Philip, Alexandre Berard, Matthias Gallé, and Laurent Besacier. 2020. Monolingual adapters for zero-shot neural machine translation. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing, EMNLP 2020, Online, November 16-20, 2020*, pages 4465–4470. Association for Computational Linguistics.

Matt Post. 2018. A call for clarity in reporting BLEU scores. In *Proceedings of the Third Conference on Machine Translation: Research Papers, WMT 2018, Belgium, Brussels, October 31 - November 1, 2018*, pages 186–191. Association for Computational Linguistics.

Ye Qi, Devendra Singh Sachan, Matthieu Felix, Sarguna Padmanabhan, and Graham Neubig. 2018. When and why are pre-trained word embeddings useful for neural machine translation? In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT, New Orleans, Louisiana, USA, June 1-6, 2018, Volume 2 (Short Papers)*, pages 529–535. Association for Computational Linguistics.

Alessandro Raganato, Raúl Vázquez, Mathias Creutz, and Jörg Tiedemann. 2021. An empirical investigation of word alignment supervision for zero-shot multilingual neural machine translation. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing, EMNLP 2021, Virtual Event / Punta Cana, Dominican Republic, 7-11 November, 2021*, pages 8449–8456. Association for Computational Linguistics.

Annette Rios, Mathias Müller, and Rico Sennrich. 2020. Subword segmentation and a single bridge language affect zero-shot neural machine translation. In *Proceedings of the Fifth Conference on Machine Translation, WMT@EMNLP 2020, Online, November 19-20, 2020*, pages 528–537. Association for Computational Linguistics.

Devendra Singh Sachan and Graham Neubig. 2018. Parameter sharing methods for multilingual self-attentional translation models. In *Proceedings of the Third Conference on Machine Translation: Research Papers, WMT 2018, Belgium, Brussels, October 31 - November 1, 2018*, pages 261–271. Association for Computational Linguistics.

Rico Sennrich, Barry Haddow, and Alexandra Birch. 2016. Neural machine translation of rare words with subword units. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics, ACL 2016, August 7-12, 2016, Berlin, Germany, Volume 1: Long Papers*. The Association for Computer Linguistics.

Xu Tan, Jiale Chen, Di He, Yingce Xia, Tao Qin, and Tie-Yan Liu. 2019. Multilingual neural machine translation with language clustering. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing, EMNLP-IJCNLP 2019, Hong Kong, China, November 3-7, 2019*, pages 963–973. Association for Computational Linguistics.

Jörg Tiedemann. 2012. Parallel data, tools and interfaces in OPUS. In *Proceedings of the Eighth International Conference on Language Resources and Evaluation, LREC 2012, Istanbul, Turkey, May 23-25, 2012*, pages 2214–2218. European Language Resources Association (ELRA).

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, December 4-9, 2017, Long Beach, CA, USA*, pages 5998–6008.

Raúl Vázquez, Alessandro Raganato, Mathias Creutz, and Jörg Tiedemann. 2020. A systematic study of inner-attention-based sentence representations in multilingual neural machine translation. *Comput. Linguistics*, 46(2):387–424.

Hongyu Wang, Shuming Ma, Li Dong, Shaohan Huang, Dongdong Zhang, and Furu Wei. 2022. Deepnet: Scaling transformers to 1, 000 layers. *CoRR*, abs/2203.00555.

Qian Wang and Jiajun Zhang. 2022. Parameter differentiation based multilingual neural machine translation. In *Thirty-Sixth AAAI Conference on Artificial Intelligence, AAAI 2022, Thirty-Fourth Conference on Innovative Applications of Artificial Intelligence,*

IAAI 2022, The Twelveth Symposium on Educational Advances in Artificial Intelligence, EAAI 2022 Virtual Event, February 22 - March 1, 2022, pages 11440–11448. AAAI Press.

Yining Wang, Jiajun Zhang, Feifei Zhai, Jingfang Xu, and Chengqing Zong. 2018. Three strategies to improve one-to-many multilingual translation. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing, Brussels, Belgium, October 31 - November 4, 2018*, pages 2955–2960. Association for Computational Linguistics.

Yining Wang, Long Zhou, Jiajun Zhang, Feifei Zhai, Jingfang Xu, and Chengqing Zong. 2019. A compact and language-sensitive multilingual translation method. In *Proceedings of the 57th Conference of the Association for Computational Linguistics, ACL 2019, Florence, Italy, July 28- August 2, 2019, Volume 1: Long Papers*, pages 1213–1223. Association for Computational Linguistics.

Zirui Wang, Yulia Tsvetkov, Orhan Firat, and Yuan Cao. 2021. Gradient vaccine: Investigating and improving multi-task optimization in massively multilingual models. In *9th International Conference on Learning Representations, ICLR 2021, Virtual Event, Austria, May 3-7, 2021*. OpenReview.net.

Liwei Wu, Shanbo Cheng, Mingxuan Wang, and Lei Li. 2021. Language tags matter for zero-shot neural machine translation. In *Findings of the Association for Computational Linguistics: ACL/IJCNLP 2021, Online Event, August 1-6, 2021*, volume ACL/IJCNLP 2021 of *Findings of ACL*, pages 3001–3007. Association for Computational Linguistics.

Wanying Xie, Yang Feng, Shuhao Gu, and Dong Yu. 2021. Importance-based neuron allocation for multilingual neural machine translation. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing, ACL/IJCNLP 2021, (Volume 1: Long Papers), Virtual Event, August 1-6, 2021*, pages 5725–5737. Association for Computational Linguistics.

Hongfei Xu, Qiuhui Liu, Josef van Genabith, and Deyi Xiong. 2021. Modeling task-aware MIMO cardinality for efficient multilingual neural machine translation. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing, ACL/IJCNLP 2021, (Volume 2: Short Papers), Virtual Event, August 1-6, 2021*, pages 361–367. Association for Computational Linguistics.

Yilin Yang, Akiko Eriguchi, Alexandre Muzio, Prasad Tadepalli, Stefan Lee, and Hany Hassan. 2021. Improving multilingual translation by representation and gradient regularization. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing, EMNLP 2021, Virtual Event / Punta Cana, Dominican Republic, 7-11 November, 2021*, pages 7266–7279. Association for Computational Linguistics.

Dian Yu, Taiqi He, and Kenji Sagae. 2021. Language embeddings for typology and cross-lingual transfer learning. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing, ACL/IJCNLP 2021, (Volume 1: Long Papers), Virtual Event, August 1-6, 2021*, pages 7210–7225. Association for Computational Linguistics.

Biao Zhang, Ankur Bapna, Rico Sennrich, and Orhan Firat. 2021. Share or not? learning to schedule language-specific capacity for multilingual translation. In *9th International Conference on Learning Representations, ICLR 2021, Virtual Event, Austria, May 3-7, 2021*. OpenReview.net.

Biao Zhang, Philip Williams, Ivan Titov, and Rico Sennrich. 2020. Improving massively multilingual neural machine translation and zero-shot translation. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, ACL 2020, Online, July 5-10, 2020*, pages 1628–1639. Association for Computational Linguistics.

Yaoming Zhu, Jiangtao Feng, Chengqi Zhao, Mingxuan Wang, and Lei Li. 2021. Counter-interference adapter for multilingual machine translation. In *Findings of the Association for Computational Linguistics: EMNLP 2021, Virtual Event / Punta Cana, Dominican Republic, 16-20 November, 2021*, pages 2812–2823. Association for Computational Linguistics.

Barret Zoph and Kevin Knight. 2016. Multi-source neural translation. In *NAACL HLT 2016, The 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, San Diego California, USA, June 12-17, 2016*, pages 30–34. The Association for Computational Linguistics.

A Efficient Implementation of LAA

Suppose that the number of target languages is l and the dimensions of $\boldsymbol{Q}$, $\boldsymbol{K}$, $\boldsymbol{V}$ are $b \times n \times d_{model}$, where b is the batch size and n is the sequence length. Since a minibatch usually contains samples from various language pairs during training, each minibatch is required to be split into smaller batches according to the target language so as to compute Eq. (3) and (5). And the output of Eq. (5) is required to be reorganized to preserve the original order of samples within the minibatch, which brings extra computational cost and is inefficient for computing on modern parallel hardware like GPU. To avoid this, we maintain a matrix $\widetilde{\boldsymbol{W}} \in \mathbb{R}^{l \times d_{model} \times d_{model}}$ which consists of language representations from all languages. We select $\overline{\boldsymbol{W}} \in \mathbb{R}^{b \times d_{model} \times d_{model}}$ from $\widetilde{\boldsymbol{W}}$, which contains all language representations for samples in the minibatch. This selection can be efficiently executed by the `Pytorch` toolkit.[11] To avoid broadcasting[12] $\boldsymbol{W}^Q, \boldsymbol{W}^K, \boldsymbol{W}^V, \boldsymbol{W}^O$ into larger dimensions when added with $\overline{\boldsymbol{W}}$, we reformulate Eq. (3) and (5) as follows:

$$\begin{aligned} \boldsymbol{q} &= \boldsymbol{Q}\boldsymbol{W}^Q + \boldsymbol{Q}\overline{\boldsymbol{W}} \\ \boldsymbol{k} &= \boldsymbol{K}\boldsymbol{W}^K + \boldsymbol{K}\overline{\boldsymbol{W}} \\ \boldsymbol{v} &= \boldsymbol{V}\boldsymbol{W}^V + \boldsymbol{V}\overline{\boldsymbol{W}} \end{aligned} \tag{6}$$

$$\boldsymbol{Z} = \boldsymbol{z}\boldsymbol{W}^O + \boldsymbol{z}\overline{\boldsymbol{W}}^{\mathsf{T}} \tag{7}$$

Note that we omit the subscript for head index in original formulas as all heads are computed in parallel. As there are always duplicate elements in $\{\boldsymbol{Q}, \boldsymbol{K}, \boldsymbol{V}\}$, the second term (i.e. $\{\boldsymbol{Q},\boldsymbol{K},\boldsymbol{V}\}\overline{\boldsymbol{W}}$) in Eq. (6) can be computed first and then cached to avoid redundant computation.

B Dataset for Many-to-Many Translation

We removed the __en__ tag prepended to non-English sentences in the original TED-59 dataset.[13] We trained BPE model (Sennrich et al., 2016) using SentencePiece (Kudo and Richardson, 2018) to get subword units with a joint vocabulary of size 64K.

C Model & Training for Many-to-Many Translation

We implement our MNMT models based on `Fairseq` (Ott et al., 2019). We set the dimension of word embeddings and FFN layer to 512/2048. Embeddings were shared for the encoder, decoder and the output projection. To prevent overfitting, we set dropout rate to 0.1 on the OPUS-100 dataset and 0.2 on the TED-59 dataset due to its relatively small data size. We adopted the cross-entropy loss with a label smoothing of 0.1 as the training objective. We used Adam ($\beta_1 = 0.9, \beta_2 = 0.98$) (Kingma and Ba, 2015) to optimize model parameters. We varied the learning rate according to the `inverse_square_root` schedule (Vaswani et al., 2017) with a warm-up step of 4000 and a peak learning rate of 0.0005. We trained all MNMT models for 30 epochs and each minibatch contains a maximum of 4096 tokens. The training data of each epoch is composed of the training datasets from all translation directions. For the OPUS-100 dataset, the number of total training steps for the model with and without oversampling are $\sim$ 933,000 and $\sim$ 617,000 respectively. For the TED-59 dataset, the number of total training steps with and without oversampling are $\sim$ 226,000 and $\sim$ 142,000 respectively.[14] Parallel sentences in the training data sets where the number of subwords on either the source or target side exceeds 100 were removed.

D Evaluation of MNMT Model

We performed beam search decoding with a beam size of 5 and length penalty of 1.0 during inference. As the TED-59 dataset has already been tokenized, we detokenized reference and system translations with `sacremoses`[15] toolkit before computing BLEU. We chose the best checkpoint according to

[11] https://pytorch.org/docs/stable/generated/torch.index_select.html?highlight=index_select#torch.index_select

[12] https://pytorch.org/docs/stable/notes/broadcasting.html?highlight=broadcasting

[13] An artificial English tag __en__ is prepended to every non-English sentence in the raw dataset, which may affect model training and bias BLEU. Nevertheless, previous works on this dataset usually do not elaborate this procedure, which may make our results on this dataset not directly comparable to theirs.

[14] To reduce the number of padding tokens in the minibatch, `Fairseq` sorts the training data by comparing the target sentence length first and then the source sentence length by default. Additionally, the number of tokens in each training sample is computed as the maximum of source and target sentence length, which is used to enforce the minibatch size. As a result, the language tag prepending strategy will affect the number of minibatches. Because of this, we report the approximate number of training steps.

[15] https://github.com/alvations/sacremoses

the average BLEU on the validation sets and then evaluated it on the test sets. Considering that there are no validation sets for zero-shot translation directions, we used the checkpoint selected on the supervised translation directions for zero-shot translation. To be comparable with (Zhang et al., 2020), we employed `langdetect`[16] toolkit for language identification on the OPUS-100 dataset. For TED-59 dataset, we used the `langid.id` toolkit[17] instead as it supports more languages than `langdetect`. We disregarded languages that cannot be detected by `langid.id` toolkit such as Canadian French (fr-ca), Brazilian Portuguese (pt-br) and Burmese (my).

E Dataset for Linguistic Typology Prediction

URIEL is a typological compendium which accommodates diverse linguistic resources from several typological databases such as WALS (Dryer and Haspelmath, 2013), PHOIBLE (Moran and McCloy, 2019) and Glottolog (Hammarström et al., 2021). We used `lang2vec`[18] library to query URIEL database which provides uniform interface to access various linguistic features.

F Prediction Method for Typology Features

We inferred typology features of languages from the language representations derived from the trained MNMT model which achieves the best performance on the validation sets. Some previous works train logistic regression classifiers to perform typology prediction (Malaviya et al., 2017; Oncevay et al., 2020). Nevertheless, logistic regression is a parameterized algorithm and the number of its parameters increase with the feature dimensions of input data, which makes it difficult to handle data scarcity with high-dimensional inputs such as matrix. We hence adopted k-nearest neighbors approach (k-NN), a non-parametric method, for linguistic typology prediction. We employed cosine similarity as the distance measure for LEE. For the language representations in LAA, we computed the average cosine similarity for all rows in the matrix as the distance metric. We set k as odd numbers and varied k in $\{1, 3, 5, 7, 9\}$. We left one language out and took the remaining languages as training examples to make predictions. This procedure was repeated for each language and the average prediction accuracies on all languages are reported.

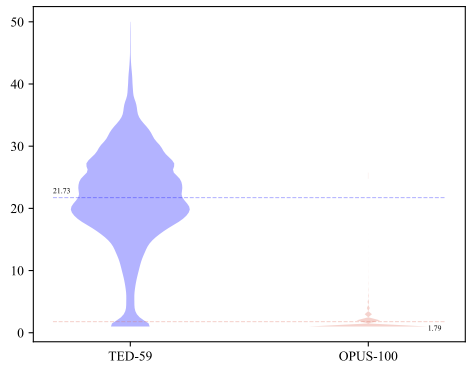

Figure 4: The distribution of the numbers of translations in other languages for each unique English sentence on the TED-59 and OPUS-100 training data. The dashed lines show the mean values.

G Detailed Many-to-Many Translation Results for LEE

Our findings summarized from Table 2 can be shown as follows:

The ways to indicate the desired target language to MNMT models can significantly affect the translation performance, especially on zero-shot translation directions. There are 0.33 and 5.88 maximum average BLEU differences on supervised (④ vs. ⑥) and zero-shot (⑥ vs. ⑨) translation directions respectively on the TED-59 dataset among the variations of LEE with the same number of parameters. Similarly, although LEE does not use extra parameters compared to $\text{Token}_{\text{src}}$ and $\text{Token}_{\text{tgt}}$, the average BLEU difference on zero-shot translation directions between LEE and $\text{Token}_{\text{src}}$ can be significant on both datasets (6.56 on the TED-59, ⑥ vs. ②) (3.31 on the OPUS-100 dataset, ⑪ vs. ⑫). These indicate the importance of the effective target language information injecting method for MNMT models.

The performance gap between $\text{Token}_{\text{src}}$ and $\text{Token}_{\text{tgt}}$ on zero-shot translation directions varies substantially across different datasets. $\text{Token}_{\text{src}}$ obtains the best performance in terms of both average BLEU and WR and outperforms $\text{Token}_{\text{tgt}}$ by 8.08 BLEU on zero-shot translation directions on the TED-59 dataset (① vs. ②). However, it lags behind $\text{Token}_{\text{tgt}}$ on zero-shot transla-

[16] https://github.com/Mimino666/langdetect
[17] https://github.com/saffsd/langid.py
[18] https://github.com/antonisa/lang2vec

tion directions on the OPUS-100 dataset ((10) vs. (11)). In order to find the main reasons behind the performance discrepancy between $\text{Token}_{\text{src}}$ and $\text{Token}_{\text{tgt}}$ on different datasets, we analyzed the two datasets from the perspective of their multi-way alignment since TED talks have been translated into many languages. Specifically, we counted the numbers of translations in other languages for each unique English sentence in the training data of the two datasets.[19] The results are visualized in Figure 4. Although both TED-59 and OPUS-100 are English-centric datasets, the majority of English sentences in the TED-59 dataset have more than one translations in other languages. We hypothesize that identical English sentences on the source side paired with distinct target sentences in different languages may encourage MNMT models to capture the correlations between the prepended tokens of $\text{Token}_{\text{src}}$ and the target languages, which benefits zero-shot translation and is consistent with the finding of Wu et al. (2021).

Prepending a target language token to the source side ($\text{Token}_{\text{src}}$) and to the target side ($\text{Token}_{\text{tgt}}$) benefit En → XX and XX → En translation directions respectively The average 0.5 BLEU gain and 91.38% WR on the TED-59 dataset ((1) vs. (2)) together with the average 0.58 BLEU gain and 81.91% WR on the OPUS-100 dataset ((10) vs. (11)) on the En → XX translation directions indicate that $\text{Token}_{\text{src}}$ is more preferable for En → XX language pairs than $\text{Token}_{\text{tgt}}$. Moreover, $\text{Token}_{\text{src}}$ achieves the highest WR on En → XX translation directions across the two datasets. Despite the leading performance of $\text{Token}_{\text{src}}$ on En → XX translation directions, it suffers from inferior average BLEU and WR on XX → En translation directions, which results in comparable average BLEU with $\text{Token}_{\text{tgt}}$ over all supervised translation directions. In contrast, $\text{Token}_{\text{tgt}}$ achieves better performance on XX → En translation directions than $\text{Token}_{\text{src}}$.

Embodying the target language embedding in the decoder is preferable for supervised translation directions The candidate positions for embodying language information are scattered across the encoder (position 1, 2) and decoder (position 3, 4, 5, 6). Table 2 shows that embodying the language embedding into positions in the decoder can achieve inprovements in average BLEU ((1)(2) vs. (5)(6)(7)(8), (10)(11) vs. (12)), while incorporating the language embedding to positions in the encoder will cause performance drop ((1)(2) vs. (3)(4)). This suggests that embodying the language embedding for the target language into positions closer to target translations is the key ingredient to improve model performance on supervised translation directions for LEE.

H Linguistic Typology Prediction Results for LEE

ID	Feature	Dataset	Model	k					Max
				1	3	5	7	9	
(1)	Syntax	TED-59	$\text{Token}_{\text{tgt}}$	81.56	82.15	**82.81**	81.18	80.79	82.81
(2)		TED-59	$\text{Token}_{\text{src}}$	86.11	**86.14**	85.15	83.92	82.63	**86.14**
(3)		TED-59	$\text{LEE}_{4,5}$	82.40	84.70	**85.59**	84.45	83.42	85.59
(4)		OPUS-100	$\text{Token}_{\text{tgt}}$	82.66	**83.97**	83.49	83.25	82.98	83.97
(5)		OPUS-100	$\text{Token}_{\text{src}}$	**84.80**	84.32	83.70	83.93	83.12	**84.80**
(6)		OPUS-100	$\text{LEE}_{4,5}$	**84.42**	83.43	83.80	82.34	82.07	84.42
(7)	Phonology	TED-59	$\text{Token}_{\text{tgt}}$	71.96	**84.23**	67.28	59.21	59.15	84.23
(8)		TED-59	$\text{Token}_{\text{src}}$	80.11	**84.97**	67.28	59.03	59.03	**84.97**
(9)		TED-59	$\text{LEE}_{4,5}$	69.68	**78.48**	66.66	58.97	59.03	78.48
(10)		OPUS-100	$\text{Token}_{\text{tgt}}$	79.57	**85.94**	81.38	73.78	73.82	85.94
(11)		OPUS-100	$\text{Token}_{\text{src}}$	**86.76**	86.50	83.94	73.95	73.95	**86.76**
(12)		OPUS-100	$\text{LEE}_{4,5}$	**85.79**	80.74	73.82	73.87	74.03	85.79
(13)	Phonetic inventory	TED-59	$\text{Token}_{\text{tgt}}$	86.67	**87.86**	87.11	85.07	65.80	87.86
(14)		TED-59	$\text{Token}_{\text{src}}$	87.85	**88.38**	87.44	85.32	66.06	**88.38**
(15)		TED-59	$\text{LEE}_{4,5}$	86.52	**88.03**	87.15	85.12	65.73	88.03
(16)		OPUS-100	$\text{Token}_{\text{tgt}}$	**87.89**	87.68	72.78	59.41	59.47	87.89
(17)		OPUS-100	$\text{Token}_{\text{src}}$	**87.86**	87.80	72.96	59.47	59.42	87.86
(18)		OPUS-100	$\text{LEE}_{4,5}$	**88.38**	87.76	72.80	59.42	59.24	**88.38**

Table 6: Linguistic typology prediction accuracies on syntax, phonology and phonetic inventory features using the language embedding learned by $\text{Token}_{\text{tgt}}$, $\text{Token}_{\text{src}}$ and $\text{LEE}_{4,5}$ which are trained on the TED-59 and OPUS-100 datasets respectively. k denotes the number of nearest neighbors in k-NN. Max denotes the maximum accuracy when k varies in $\{1, 3, 5, 7, 9\}$.

I Linguistic Typology Prediction Results for Models Trained on OPUS-100

Figure 5 shows the linguistic typology prediction results for language representations extracted from MNMT models which are trained on the OPUS-100 dataset.

J Many-to-Many Translation Results of $\text{LAA}^{R}_{\text{dec.self}}$ on Raw Data Distribution

Table 7 presents the many-to-many translation results of $\text{LAA}^{R}_{\text{dec.self}}$ when oversampling is removed.

[19] We collected English sentences by concatenating all parallel corpora of English and eliminating duplicate entries.

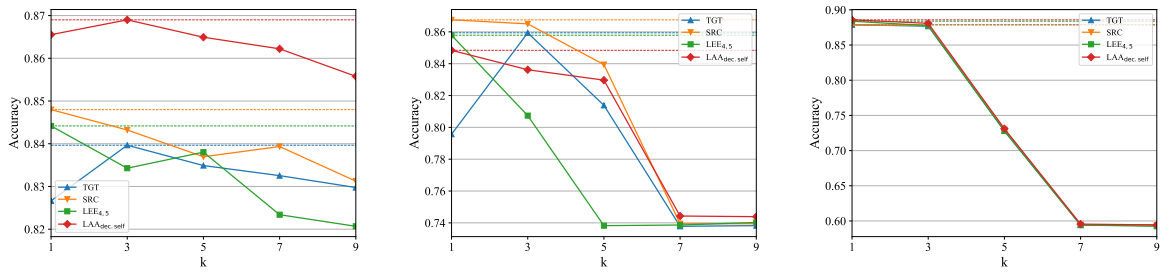

(a) Syntax (b) Phonology (c) Phonetic inventory

Figure 5: Prediction accuracy on syntax, phonology and phonetic inventory features using the language embeddings learned by $\text{Token}_{\text{tgt}}$, $\text{Token}_{\text{src}}$, $\text{LEE}_{4,5}$ and $\text{LAA}_{\text{dec.self}}$ which are trained on the OPUS-100 dataset.

ID	Dataset	Model	#Param	En → XX		XX → En		All		Zero-shot		
				BLEU	WR	BLEU	WR	BLEU	WR	BLEU	LangAcc	WR
①	TED-59	$\text{Token}^{-}_{\text{tgt}}$	77M	19.54	*ref*	24.23	*ref*	21.89	*ref*	2.84	40.94	*ref*
②	TED-59	$\text{Token}^{-}_{\text{src}}$	77M	20.25	**96.55**	23.65	8.62	21.95	52.59	9.65	65.45	96.77
③	TED-59	$\text{LAA}^{-}_{\text{dec.self}}$	92M	20.64	**96.55**	25.16	**98.28**	**22.90**	**97.41**	9.93	75.04	97.67
④	TED-59	$\text{LAA}^{-}_{\text{dec.self}}$ + $\text{LEE}^{-}_{4,5}$	92M	**20.67**	94.83	25.05	93.10	22.86	93.97	**10.02**	**75.26**	**97.82**
⑤	TED-59	$\text{LAA}^{R-}_{\text{dec.self}}$ + $\text{Token}^{-}_{\text{src}}$	92M	19.65±0.01	61.38±2.61	25.75±0.01	90.34±1.54	22.70±0.01	75.86±0.86	9.89±0.00	73.30±0.02	95.91±0.06
⑥	TED-59	$\text{LAA}^{R-}_{\text{dec.self}}$ + $\text{Token}^{-}_{\text{tgt}}$	92M	18.53±0.01	1.03±0.94	**26.30±0.02**	93.10±0.00	22.41±0.01	47.07±0.47	1.02±0.00	11.37±0.01	1.06±0.07
⑦	OPUS-100	$\text{Token}^{-}_{\text{tgt}}$	77M	21.82	*ref*	28.45	*ref*	25.14	*ref*	6.63	58.76	*ref*
⑧	OPUS-100	$\text{Token}^{-}_{\text{src}}$	77M	22.15	74.47	27.68	10.64	24.91	42.55	4.91	37.80	30.00
⑨	OPUS-100	$\text{LAA}^{-}_{\text{dec.self}}$	103M	23.57	**91.49**	28.71	70.21	26.14	80.85	11.93	80.39	**100.00**
⑩	OPUS-100	$\text{LAA}^{-}_{\text{dec.self}}$ + $\text{LEE}^{-}_{4,5}$	103M	**23.69**	**91.49**	**28.88**	**78.72**	**26.29**	**85.11**	**12.77**	**85.00**	**100.00**
⑪	OPUS-100	$\text{LAA}^{R-}_{\text{dec.self}}$ + $\text{Token}^{-}_{\text{src}}$	103M	21.63±0.01	50.21±1.39	27.07±0.02	0.64±0.58	24.35±0.01	25.43±0.69	4.96±0.02	41.21±0.05	31.33±1.83
⑫	OPUS-100	$\text{LAA}^{R-}_{\text{dec.self}}$ + $\text{Token}^{-}_{\text{tgt}}$	103M	20.75±0.02	9.15±1.61	28.11±0.03	20.00±1.75	24.43±0.02	14.57±0.97	6.80±0.02	66.74±0.07	52.00±1.83

Table 7: Experiment results of $\text{LAA}^{R-}_{\text{dec.self}}$ on the two datasets. The superscript "-" denotes that there is no oversampling. The results of ①②③④⑦⑧⑨⑩ are from Table 4.

Rare but Severe Neural Machine Translation Errors Induced by Minimal Deletion: An Empirical Study on Chinese and English

Ruikang Shi and **Alvin Grissom II** and **Duc Minh Trinh**
Haverford College
Haverford, PA, USA

Abstract

We examine the inducement of rare but severe errors in English-Chinese and Chinese-English in-domain neural machine translation by minimal deletion of the source text with character-based models. By deleting a single character, we can induce severe translation errors. We categorize these errors and compare the results of deleting single characters and single words. We also examine the effect of training data size on the number and types of pathological cases induced by these minimal perturbations, finding significant variation. We find that deleting a word hurts overall translation score more than deleting a character, but certain errors are more likely to occur when deleting characters, with language direction also influencing the effect.

1 Introduction

Pathological machine translation (MT) errors have been a problem since the field's inception, and they have been analyzed and categorized in the context of both statistical (SMT) and neural machine translation (NMT). Recent work examines pathologies in NLP models on classification problems: cases in which the models make wildly inaccurate predictions, often confidently, when input tokens are removed (Feng et al., 2018). Identifying these enriches our understanding of neural models and their points of failure. MT pathologies take the form of severe translation errors, the worst being hallucinations (Lee et al., 2019). These rare errors are difficult to study precisely because they are rare. In this paper, we examine severe errors induced by minimal deletions by automatically extracting translations with severe errors and manually categorizing them.

Previous work taxonomizes SMT errors (Vilar et al., 2006) and analyzes their effects on translation quality (Federico et al., 2014). More recently, Guerreiro et al. (2022) propose a taxonomy of MT pathologies, of which hallucinations are a category.[1] They note the shortcomings of current automatic detection methods, e.g., those based on quality estimation and heuristics, and look for critical errors in naturalistic settings. They also propose DEHALLUCINATOR, which flags problematic translations and replaces them with re-ranking.

Other work on Chinese-English (Zh-En) SMT examines tense errors caused by incorrectly translating 了 (*le*) (Liu et al., 2011) and syntactic failures caused by 的 (*de*). More recent work uses input perturbation to argue that NMT models, including those based on transformers (Vaswani et al., 2017), are brittle: Belinkov and Bisk (2018) examine the effect on NMT systems of several kinds of randomized perturbations by adding tokens, and Niu et al. (2020) study subword regularization to increase robustness to randomized perturbations. Raunak et al. (2021) argue that memorized training examples are more likely to hallucinate, and Voita et al. (2020) examine the contribution of source and target tokens to errors. Also related, Sun et al. (2020) suggest that BERT is less robust to misspellings than other kinds of noise, which can occur naturalistically or through other errors (e.g., encoding).

While we expect targeted adversarial examples—those explicitly designed to cause a system to fail (Jia and Liang, 2017; Ebrahimi et al., 2018)—to cause serious errors, we focus on the ostensibly more benign case of in-domain En↔Zh NMT with minimal deletions. Adding valid words introduces distractors with which the MT system must cope, while deleting words more often *removes* information without explicitly introducing lexical distractors. Both are noise, but the latter is more naturally framed as requiring recovery from missing information, while the former introduces irrelevant and misleading information. At the character level, this distinction is less clear, since both adding and re-

[1] Guerreiro et al. (2022) note that the term "hallucination" is overloaded and inconsistent; for this reason, we generally avoid the use of this term here.

Proceedings of the 29th International Conference on Computational Linguistics, pages 5175–5180
October 12–17, 2022.

moving characters requires that the model translate despite unseen input substrings—minimally corrupted inputs. Are minimal word or character corruptions more harmful to a purely character-based NMT model? The answer is not obvious.

While most prior work examines western European languages, we examine translation between Chinese and English, building upon work identifying errors by observing change in BLEU (Papineni et al., 2002) after perturbation (Lee et al., 2019). But in contrast this prior work, which adds tokens, we focus exclusively on single deletions to examine **minimal conditions**—i.e., a missing character or word, as in a typo or corruption—under which **severe errors** are newly induced. For our purposes, a severe error leads to a translation in which the original meaning is unrecoverable, but there are others, as well (Vilar et al., 2006). For our purposes, we use WORD CHANGING to cover these cases.

2 Finding Candidates

We now describe the training of our NMT model, method for extracting severe error candidates (**enumerations**), and the results of this extraction. For our extraction experiments, we begin by examining character deletion before repeating the same experiments with word deletion. All experiments are done in both directions and for two different training data sizes (1M and 10M sentences), allowing us to observe the effect of training data size, translation direction, and deletion type.

2.1 Data and Models

We train character-based En↔Zh models on the UN Parallel Corpus 1.0 (Ziemski et al., 2016) of sentence-aligned UN parliamentary documents.

We train two models in each direction with Sockeye 2 (Hieber et al., 2020)—the first on the first 1M sentences and the second on 10M—to observe the effect of training data size on severe errors. We use the final 8,041 sentences as validation and test data; the first 2,000 are test data.[2]

[2]We use a six-layer transformer with eight attention heads and a feed-forward network of 2,048 hidden units, trained on one 16GB Quadro P5000. Batch size is 256 and learning rate is .0002, reduced by a factor of .9 after 8 unimproving checkpoints. Training ceases when validation perplexity quiesces for 20 checkpoints of 4,000 updates. While BPE has been shown to have higher BLEU on several datasets, this is not always the case (Cherry et al., 2018), and it can sometimes cause anomalies in translation itself (Ataman et al., 2017; Huck et al., 2017). We want to analyze the effect of deletion under simple conditions without this added complexity.

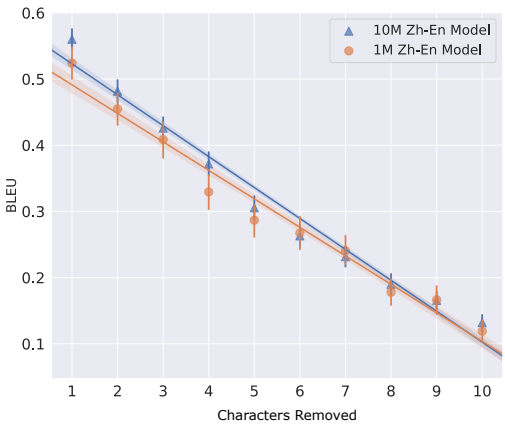

Figure 1: Zh-En BLEU as function of characters removed on valid sentences with 95% confidence intervals. There is a linear relationship, with average BLEU converging as more tokens are removed.

2.2 Identifying Error Candidates

On translated test sentences, if sentence-level BLEU is above 0.5, the translation is considered **valid**.[3] We translate valid sentences with one token missing, exhaustively trying every possible deleted token in every sentence. Perturbed sentences' translations are called **enumerations**. If an enumeration's sentence-level BLEU is less than .1, it is a candidate error, as these precipitous drops are outliers in the linear decline in BLEU as tokens are removed (Figure 1). For a more detailed specification of this process, see Algorithm 1, which is written for clarity rather than efficiency.[4]

3 Experiments and Results

We now discuss our experiments and the results of our enumeration extraction and the errors contained therein. All results are summarized in Table 2, with results on the same 2,000 test sentences.

3.1 Error Categorization

We manually categorize errors into four types in our analysis: WORD CHANGING, INABILITY, MISSING PARTS, and IRRELEVANT. Examples and descriptions are in Table 1.[5] What they have in common is

[3]We choose this because it is well above average BLEU for all models, resulting in an enumeration set with high average BLEU (Table 2), and few perturbed sentences reach this score (Figure 1).

[4]Code is hosted on GitHub.

[5]MISSING PARTS differs from INABILITY in that what *is* translated for MISSING PARTS is correct. IRRELEVANT translations are readable but unrelated to the source, while

Error Type	Example	Description
WORD CHANGING	**Source:** Occupational health and occupational risks. **Perturbed Source:** Occupational heath and occupational risks **Reference:** 职业 健康 与 职业 风险 zhíyè jiànkāng yǔ zhíyè fēngxiǎn occupational health and occupational risks **Translation:** 职业 道德 和 职业 危险 zhíyè dàodé hé zhíyè wéixiǎn occupational ethics with occupational dangers	The model only mistranslates the perturbed word, leading to a simple error in which *health* has been swapped with the unrelated word *ethics* (which is also orthographically distant in the source text).
INABILITY	**Source :** Christian Peace Action Groups. **Perturbed Source:** Christian PeaceAction Groups. **Reference:** 基督教 和平 行动 组织 jīdūjiào hépíng xíngdòng zǔzhī Christian Peace Action Groups **Translation:** Christian Peaction Groups	Instead of outputting Chinese, the model copies English characters, including the nonsense word *Peaction*.
MISSING PARTS	**Source:** Residential institutions: services for children. **Perturbed Source:** esidential institutions: services for children. **Reference:** 寄宿 机构 : 为 儿童 提供 服务 jìsù jīgòu : wèi értóng tígōng fúwù Residential institutions : for children provide services **Translation:** 对 儿童 的 服务 duì er tóng de fúwù for children 'de' services	Only some of the text is translated. In this example, though the translation is interpretable, a substantial portion of the text is entirely untranslated.
IRRELEVANT	**Source:** Maternal breastfeeding. **Perturbed Source:** aternal breastfeeding. **Reference:** 母乳 喂养 mǔrǔ wèiyǎng maternal breastfeeding **Translation:** 联合国 维持 和平 行动 经费 的 筹措 liánhéguó wéichí hépíng xíngdòng jīngfèi de chóucuò UN keep peace operation funding 'de' raise	This output is entirely hallucinated and has no apparent relationship to the input.

Table 1: Examples and descriptions of triggers and error types found in low-scoring enumerations.

that the original meaning is unrecoverable, though simple WORD CHANGING is not considered *a priori* severe in our analysis.

3.2 En-Zh 1M Training Sentence Results

There are 96 candidate severe errors among 14,722 enumerations: ten INABILITY, three IRRELEVANT and five MISSING PARTS. The rest are WORD CHANGING. We have 18 errors (.12%). [6]

One possible reason for these errors is that the model has insufficient training data to generalize. We investigate by training on ten times the data.

3.3 En-Zh Model Trained on 10M Sentences

We use the same corpus and architecture but use the first 10M instead of 1M parallel sentences to train (En-Zh-10M). Validation perplexity is nearly halved to 6.0 vs. the 1M model's 11.5 Likewise, BLEU on the test data increases by .08 to .4 (Table 2), as expected. Unexpectedly, BLEU on enumerations drops by .16 with more training data, much more than the .11 drop with 1M training sentences, suggesting more training data counterintuitively *increases* sensitivity to minimal character deletions, despite initial BLEU being higher.

There are 119 candidates among the 30,079 enumerations: 33 INABILITY and no MISSING PARTS or IRRELEVANT, giving a 0.11% probability of severe errors, approximately the same as the 1M model (0.12%).

The distribution of error types differs considerably when training on more data: INABILITY errors triple. We find that this is due to untranslated

INABILITY indicates a failure to generate readable output. It is possible in principle to have many types of errors in one bad translation, but we did not observe this.

[6] We also try removing sentences with English characters on the Chinese side, leaving 831,941 sentences on which to train. Translating these yields no INABILITY errors and leaves BLEU largely unchanged, suggesting that the untranslated named entities in the training data indeed cause INABILITY. There are three MISSING PARTS and two IRRELEVANT out of 63 potential hallucinations. Test BLEU is largely unchanged, and valid BLEU decreases only slightly.

Model	BLEU	Deletion	Valid	BLEU (Valid)	Enum.	BLEU (Enum.)	ΔBLEU	In.	MP	Irr.	Total Errors
En-Zh-1M	.32	Char	351	.77	14,722	.66	-.11 (-14.2%)	10	5	3	18 (0.12%)
En-Zh-10M	.40	Char	506	.80	30,079	.64	-.16 (-20.0%)	33	0	0	33 (0.11%)
Zh-En-1M	.39	Char	602	.73	11,093	.62	-.11 (-15.0%)	0	5	1	6 (0.05%)
Zh-En-10M	.42	Char	714	.78	14,031	.67	-.11 (-14.1%)	0	1	0	1 (0.007%)
En-Zh-1M	.32	Word	351	.77	2,521	.48	-.29 (-37.6%)	3	0	5	8 (0.32%)
En-Zh-10M	.40	Word	506	.80	4,945	.54	-.26 (-32.5%)	7	0	2	9 (0.18%)
Zh-En-1M	.39	Word	602	.74	6,666	.54	-.20 (-27.0%)	0	2	6	8 (0.12%)
Zh-En-10M	.42	Word	724	.78	8,461	.58	-.20 (-25.6%)	0	1	9	10 (0.11%)

Table 2: Results of candidate extraction for minimal deletion, BLEU for each extracted set of sentences, and error statistics in models, broken down into INABILITY (**In.**), MISSING PARTS (**MP**), and IRRELEVANT (**Irr.**). **Valid** sentences with BLEU > 0.5 are extracted to create minimally perturbed **enumerations**; from these candidates, bad translations are extracted based on BLEU decline post-perturbation (ΔBLEU). Despite character deletion introducing nonsense words into the input, word removal causes more of these severe errors. Surprisingly, despite Chinese characters containing more information, English deletion causes substantially higher decline in BLEU.

words in the training data, all of which are named entities.[7] Since more training data contains more untranslated named entities, INABILITY is more likely in models trained on more data. We therefore train a model on the data where no English appears in the references.

3.4 Zh-En Experiments

We examine Zh-En MT under the same character deletion conditions as En-Zh. Since Chinese characters contain more information than English letters, we expect greater sensitivity to deletions on Zh-En, but we do not find this (Table 2). Perturbing En-Zh leads to consistently steeper declines in BLEU, as seen in the valid vs. enumeration scores.

On the Zh-En model trained on 1M sentences, BLEU drops by .11, from .73 for the 602 sentences to .62 for the enumerations, whereas when trained on 10M sentences, we have .67 BLEU on enumerations, which is higher than that of the smaller model. This is, notably, the opposite of the En-Zh results, where more data decreased enumeration BLEU. Both Zh-En experiments decrease by .11 BLEU on enumerations, suggesting that the model with more training data is similarly robust to this perturbation as the smaller model, unlike the En-Zh case, in which the model trained on more data is more sensitive to character perturbations. As before, training models with more data decreases Zh-En errors: on Zh-En model trained on 1M sentences, we have 1 IRRELEVANT and 5 MISSING PARTS (.05%) errors, while on when trained on10M sentences, we have 1 MISSING PARTS (.007%). The remaining errors are WORD CHANGING.

There are no INABILITY errors in the two Zh-En experiments, which accords with the results from En-Zh, suggesting that INABILITY is due to the untranslated words in the training data. Since there are no untranslated Chinese words on the English side in the training data, we expect no INABILITY for a Zh-En model.

3.5 Minimal Word Deletion

We now examine *word* deletion as a basis of comparison. Does the character NMT model better handle the corrupted words caused by minimal character deletion, or is it more robust to whole word deletion, which leaves coherent words but removes more characters?[8] We find that, in all cases, deleting words leads to *substantially* lower BLEU than deleting characters, and though still rare, confirmed severe error rates also increase.

For En-Zh trained on 1M sentences, for instance, BLEU for enumerations drops to 0.48 in comparison to 0.66 when deleting characters, and these stark differences in BLEU persist.

On En-Zh trained on 1M sentences, we have 3 INABILITY and 5 IRRELEVANT (.32% severe errors). As expected, error rate increases considerably vs. character removal (.12%).

On En-Zh trained on 10M sentences, we have 7 INABILITY and 2 IRRELEVANT. .18% of 4,945 enumerations are severe errors, also more likely than with character deletion.

[7] By convention, sometimes named entities from English are not translated into Chinese. Ugawa et al. (2018) attempted to improve NMT with named entity tags to better handle compound and ambiguous words, and other previous work showed that contamination by another language (Khayrallah and Koehn, 2018) and copies of source sentences in the target training data can degrade NMT performance.

[8] We use THULAC (Sun et al., 2016) `fast` for Chinese tokenization.

Algorithm 1 Algorithm for Finding Candidates. We describe the logic in three distinct steps for clarity, though there is obvious potential for optimization.

```
function RUN(test_sents)
    valid ← find_valid(test_sents)
            ▷ Find valid sentences with BLEU ≥ 0.5.
    generate_enumerations(valid)
            ▷ Iterate over every character in every
              valid sentence and try removing one
              character at a time. These perturbed sen-
              tences are enumerations.
    candidates ← find_candidates(valid)
            ▷ Keep the enumerations that have
              sentence-level BLEU ≤ 0.1. We then
              manually identify those with actual se-
              vere errors and categorize them.
end function

function FIND_VALID(test_sentences)
    for each sentence s in test_sentences do
        s.bleu ← bleu(s)
        if s.bleu ≥ 0.5 then
            valid.add(s)
        end if
    end for
    return valid
end function

function GENERATE_ENUMERATIONS(valid)
    for each valid sentence s in valid do
        for each char index i in s do
            enum ← s.delete_char(i)
            s.enums.add(enum)
        end for
    end for
end function

function FIND_CANDIDATES(valid)
    for each sentence s in valid do
        for each enum in s.enums do
            new_bleu ← bleu(enum)
            if new_bleu ≤ 0.1 then
                candidates.add(enum)
            end if
        end for
    end for
    return candidates
end function
```

As with character deletion, increasing training size increases INABILITY errors but decreases overall error probability. There are no MISSING PARTS errors when deleting words on En-Zh.

3.6 Summary

We see substantial variation in errors, depending on the kind of deletion and translation direction, with INABILITY occurring exclusively on En-Zh. We expect more BLEU decline on Zh-En, since Chinese characters contain more semantic content and source sentences are shorter, but we find the opposite of this with word deletion. We also find that while the models are more sensitive to word deletion in terms of overall BLEU, this does not lead to drastic increases in severe errors, suggesting that these severe errors are unrelated to typical MT errors, in line with arguments that hallucinations should be considered separately from the typical MT errors (Guerreiro et al., 2022), due to the unique patterns of heuristic-based methods when attempting to detect them.

4 Conclusion and Future Work

We examine the effect of minimal deletions on rare but severe MT errors on Chinese and English, using outlier changes in BLEU after deletion to find candidates.

We find that the error rate for the model with a larger dataset is always lower, suggesting more data can improve models' performance against severe errors. Removing single words is more likely to cause severe errors but less likely to cause MISSING PARTS in our models, despite character deletion introducing invalid words. On En→Zh, we observe none when removing words. With the important caveat that these errors are already rare, limiting the conclusions we can make, this may suggest that Zh↔En models are better able to recover when characters are missing, even if the substrings themselves have never been observed, despite not having been trained with such noise. This is not obvious for a character-based model. Nor is it obvious that Zh→En models will be more robust to perturbations than En→Zh, but this is what we find, especially for words, perhaps because English words are simply longer. Furthermore, that ΔBLEU is not predictive of significantly more severe errors suggests that these errors are a different phenomenon from typical MT shortcomings.

Further research is needed to determine the effect various variables on robustness with targeted probes; future work can also determine how findings generalize across more language pairs (potentially typologies), tokenization schemes, and architectures. Training models with missing source *words* may increase robustness. For detection, unusually large disparities in length between source and target could signal INABILITY or MISSING PARTS errors, and vocabulary or semantic distance checks could flag bad translations (e.g., WORD CHANGING, INABILITY). It would also be instructive to examine the extent to which NMT robustness to noise mirrors that of humans.

Acknowledgements

We thank John Dougherty and Sorelle Friedler for their helpful feedback.

References

Duygu Ataman, Matteo Negri, Marco Turchi, and Marcello Federico. 2017. Linguistically motivated vocabulary reduction for neural machine translation from Turkish to English. *The Prague Bulletin of Mathematical Linguistics*, 108(1):331.

Yonatan Belinkov and Yonatan Bisk. 2018. Synthetic and natural noise both break neural machine translation. In *Proceedings of ICLR*.

Colin Cherry, George F. Foster, Ankur Bapna, Orhan Firat, and Wolfgang Macherey. 2018. Revisiting character-based neural machine translation with capacity and compression. In *Proceedings of EMNLP*.

Javid Ebrahimi, Daniel Lowd, and Dejing Dou. 2018. On adversarial examples for character-level neural machine translation. In *Proceedings of COLING*, pages 653–663.

Marcello Federico, Matteo Negri, Luisa Bentivogli, and Marco Turchi. 2014. Assessing the impact of translation errors on machine translation quality with mixed-effects models. In *Proceedings of EMNLP*, pages 1643–1653.

Shi Feng, Eric Wallace, Alvin Grissom II, Mohit Iyyer, Pedro Rodriguez, and Jordan Boyd-Graber. 2018. Pathologies of neural models make interpretations difficult. In *Proceedings of EMNLP*, pages 3719–3728.

Nuno M Guerreiro, Elena Voita, and André FT Martins. 2022. Looking for a needle in a haystack: A comprehensive study of hallucinations in neural machine translation. *arXiv preprint arXiv:2208.05309*.

Felix Hieber, Tobias Domhan, Michael Denkowski, and David Vilar. 2020. Sockeye 2: A toolkit for neural machine translation. In *Proceedings of EACL*, pages 457–458, Lisboa, Portugal.

Matthias Huck, Simon Riess, and Alexander Fraser. 2017. Target-side word segmentation strategies for neural machine translation. In *Proceedings of the Second Conference on Machine Translation*, pages 56–67.

Robin Jia and Percy Liang. 2017. Adversarial examples for evaluating reading comprehension systems. In *Proceedings of EMNLP*, pages 2021–2031.

Huda Khayrallah and Philipp Koehn. 2018. On the impact of various types of noise on neural machine translation. In *Proceedings of the 2nd Workshop on Neural Machine Translation and Generation*, pages 74–83, Melbourne, Australia. Association for Computational Linguistics.

Katherine Lee, Orhan Firat, Ashish Agarwal, Clara Fannjiang, and David Sussillo. 2019. Hallucinations in neural machine translation. In *Interpretability and Robustness for Audio, Speech and Language Workshop*. Proceedings of NeurIPS.

Feifan Liu, Fei Liu, and Yang Liu. 2011. Learning from Chinese-English parallel data for Chinese tense prediction. In *Proceedings of IJCNLP*, pages 1116–1124.

Xing Niu, Prashant Mathur, Georgiana Dinu, and Yaser Al-Onaizan. 2020. Evaluating robustness to input perturbations for neural machine translation. In *Proceedings of ACL*, pages 8538–8544.

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. BLEU: a method for automatic evaluation of machine translation. In *Proceedings of ACL*, pages 311–318.

Vikas Raunak, Arul Menezes, and Marcin Junczys-Dowmunt. 2021. The curious case of hallucinations in neural machine translation. *Proceedings of NAACL*.

Lichao Sun, Kazuma Hashimoto, Wenpeng Yin, Akari Asai, Jia Li, Philip Yu, and Caiming Xiong. 2020. Adv-BERT: BERT is not robust on misspellings! generating nature adversarial samples on BERT. *arXiv preprint arXiv:2003.04985*.

Maosong Sun, Xinxiong Chen, Kaixu Zhang, Zhipeng Guo, and Zhiyuan Liu. 2016. THULAC: An efficient lexical analyzer for Chinese.

Arata Ugawa, Akihiro Tamura, Takashi Ninomiya, Hiroya Takamura, and Manabu Okumura. 2018. Neural machine translation incorporating named entity. In *Proceedings of COLING*, pages 3240–3250.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Proceedings of NeurIPS*, page 6000–6010.

David Vilar, Jia Xu, Luis Fernando D'Haro, and Hermann Ney. 2006. Error analysis of statistical machine translation output. In *Proceedings of LREC*, Genoa, Italy.

Elena Voita, Rico Sennrich, and Ivan Titov. 2020. Analyzing the source and target contributions to predictions in neural machine translation. *arXiv preprint arXiv:2010.10907*.

Michał Ziemski, Marcin Junczys-Dowmunt, and Bruno Pouliquen. 2016. The United Nations parallel corpus v1.0. In *Proceedings of LREC*, pages 3530–3534.

QUAK: A Synthetic Quality Estimation Dataset for Korean-English Neural Machine Translation

Sugyeong Eo[1], **Chanjun Park**[1,2], **Hyeonseok Moon**[1], **Jaehyung Seo**[1], **Gyeongmin Kim**[1], **Jungseob Lee**[1], **Heuiseok Lim**[1]*
[1]Korea University, [2]Upstage
{djtnrud,bcj1210,glee889,seojae777,totoro4007,omanma1928,limhseok}@korea.ac.kr
chanjun.park@upstage.ai

Abstract

With the recent advance in neural machine translation demonstrating its importance, research on quality estimation (QE) has been steadily progressing. QE aims to automatically predict the quality of machine translation (MT) output without reference sentences. Despite its high utility in the real world, there remain several limitations concerning manual QE data creation: inevitably incurred non-trivial costs due to the need for translation experts, and issues with data scaling and language expansion. To tackle these limitations, we present QUAK, a Korean-English synthetic QE dataset generated in a fully automatic manner. This consists of three sub-QUAK datasets QUAK-M, QUAK-P, and QUAK-H, produced through three strategies that are relatively free from language constraints. Since each strategy requires no human effort, which facilitates scalability, we scale our data up to 1.58M for QUAK-P, H and 6.58M for QUAK-M. As an experiment, we quantitatively analyze word-level QE results in various ways while performing statistical analysis. Moreover, we show that datasets scaled in an efficient way also contribute to performance improvements by observing meaningful performance gains in QUAK-M, P when adding data up to 1.58M.

1 Introduction

Quality estimation (QE) is the task of predicting the translation quality as a continuous value or discrete tags by referring to a source sentence and its machine translation (MT) output (Blatz et al., 2004; Specia et al., 2009, 2013). Since quality annotations on MT output are applied in various ways according to the granularity levels (word, sentence, document, etc.), QE research has been constantly developing in recent years (Kim et al., 2017; Fomicheva et al., 2020a; Alva-Manchego et al., 2021; Ding et al., 2021b).

MT output	Given that the Chinese authorities do not deny it , it is highly likely .
pseudo-PE	Given that the Chinese authorities do not deny it , chances are high .
MT output tags	OK OK OK OK OK OK OK OK OK OK OK OK OK OK OK OK OK OK OK BAD OK BAD OK BAD OK BAD OK OK OK
Source	중국 당국이 부인하지 않는 것으로 볼 때 가능성이 높다 .
Source tags	OK OK OK OK OK OK OK BAD BAD OK
Alignments	0-3 1-4 2-7 3-5 3-6 4-8 5-8 6-0 7-13 8-11 8-12 9-14
Edits	(1) Insertion (‘ ’→ it) (2) Substitution (chances→is) (3) Substitution (are→highly) (4) Substitution (high→likely)

Table 1: An example of QUAK dataset. For the correct translation, one insertion and three substitutions are required for the MT output. Although not included in this example, if there is a missing word, a BAD tag is attached to the location of the corresponding gap token. We indicate the alignment information (Alignments) in the form of {source index}-{aligned MT output index}.

Owing to this importance, datasets for training QE systems are being released continuously. However, we highlight three limitations for the existing QE dataset. (1) First, non-trivial human labor and time cost are required when constructing data. Source sentences, MT output, and quality annotations are dataset prerequisites for QE learning, among which translation experts proficient in a language pair are essential in the labeling process. Employing experts is far more difficult especially in low-resource languages.

(2) As an extension of the first limitation, manual QE datasets are restricted in size regardless of the data resource. The meticulous work of creating human post-edited sentences with minimal modifications slows down the construction time, which makes it difficult to scale. Most released QE datasets, including those from the Conference on Machine Translation (WMT) are composed of data less than 10K in size (Fujita and Sumita, 2017; Fomicheva et al., 2020b). This is a much rarer amount compared with the large volumes of data used by studies on GPT 3 (Brown et al., 2020) in terms of data-hungry NLP.

(3) Available QE language pairs are limited. Although the released WMT QE dataset considers

* Corresponding Author

Proceedings of the 29th International Conference on Computational Linguistics, pages 5181–5190
October 12–17, 2022.

high, medium, and low resources (Fomicheva et al., 2020c), it still covers only a much smaller number of language pairs compared with parallel corpora. Since data construction in opposite directions for a language pair requires an entirely different human post-edited sentence, numerous language pairs and directions are yet to be utilized.

To mitigate the above limitations, we introduce QUAK [1], a large-scale Korean-English synthetic QE dataset. This is built as an automated process by taking the Eo et al. (2021a) approach and aims to train word-level QE. Namely, the data generation process does not demand human post-editing, allowing data to be built at scale than manual methods. In addition, language extension is relatively free as it is language-agnostic within a language pair in which Google translation is possible and a corresponding corpus exists. Therefore, we adopt Korean-English, one of the morphologically rich languages rarely addressed in the QE field.

For constructing QUAK, A monolingual or parallel corpus and an MT model are required. QUAK is divided into three sub-QUAK datasets according to data sources: (1) QUAK-Monolingual (QUAK-M) leveraging a monolingual corpus of the target language, (2) QUAK-Parallel (QUAK-P) leveraging a parallel corpus, and (3) QUAK-Hybrid (QUAK-H) jointly leveraging monolingual and parallel corpus. The final QUAK training data size in QUAK-P and QUAK-H is 1.58M and 6.58M in QUAK-M, which is about 225 times and 940 times larger than the 7K size of the WMT official dataset (Fomicheva et al., 2020b).

Considering that QUAK is synthetic data, we scrutinize the dataset with statistics and a quantitative analysis to provide reliability and quality assurance. In the quantitative analysis, in particular, we first compare the word-level QE model fine-tuning performance based on multiple multilingual pre-trained language models (mPLMs) using only 100K pieces from each sub-QUAK. Thereafter, we use the best performing model to incrementally scale the data size and track performance fluctuations.

As a result of the experiment, the XLM-RoBERTa (XLM-R) (Conneau et al., 2019) large model is the most competitive, showing a difference of up to 0.12 MCC compared to other mPLMs such as multilingual BART (mBART) (Liu et al., 2020), XLM (Lample and Conneau, 2019). Furthermore, scaling the data to 1.58M tends to improve overall model performance. Based on the MT output-side, QUAK-M obtains performance gain of maximum 0.042, QUAK-P 0.037, and QUAK-H 0.029 MCC. Our contributions are as follows:

- To minimize the exorbitant human-demand and time cost of QE, we construct and release the QUAK dataset in a fully automatic manner, exploiting three efficient data generation strategies.
- To address the size limitation, we expand the synthetic data to a large-scale. We scale QUAK up to 940 times compared with the WMT official dataset.
- As the language pair for QUAK, we choose Korean-English, a low-resource language pair that has never been released before. Language coverages can be extended if only the translation model and its corpus are satisfied in the generation process.
- We analyze the QE fine-tuning performance according to various mPLMs, and analyze the results of progressively expanding the data for the best performing QE model.

2 Related Work

As QE research has increasingly been introduced recently, human-labeled QE datasets are also being released (Specia et al., 2010; Fujita and Sumita, 2017; Fomicheva et al., 2020c,b). However, data construction processes have several limitations in terms of time cost, data size, and available language pairs.

Many studies have been conducted continuously to handle these limitations. To name a few, Tuan et al. (2021) propose a synthetic data construction method that utilizes a parallel corpus to alleviate the cost of human labor and time cost. In such study, translation errors committed through a language model or an NMT system are injected to parallel sentences. A similar method of generating synthetic data through parallel corpus has also been leveraged in automatic post-editing research (Negri et al., 2018).

To address data size restrictions caused by time cost in human annotations, attempts have been made using data augmentation (Lee, 2020; Wang

[1]Our QUAK dataset is publicly available at https://bit.ly/3dqe2KE.

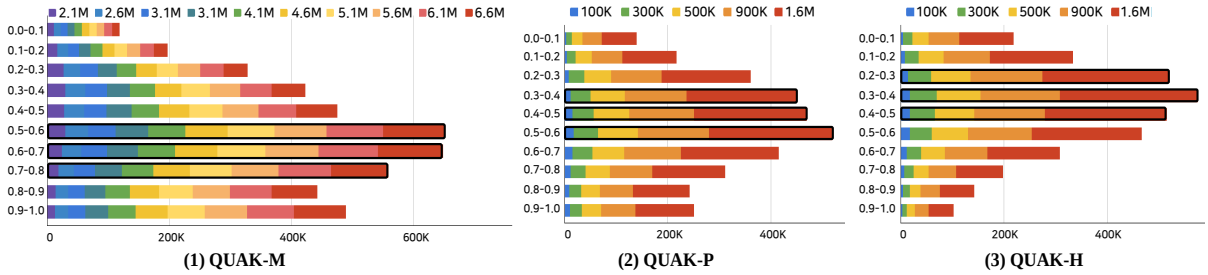

Figure 1: Distribution of the data size according to the TER range for each sub-QUAK. We present the top three scopes with the largest amount in bold lines.

et al., 2020; Gajbhiye et al., 2021; Ding et al., 2021a) and unsupervised learning (Etchegoyhen et al., 2018). In the study of Fomicheva et al. (2020c), unsupervised quality indicators based on uncertainty quantification are exploited to train the QE model.

To tackle constraints about available language pairs, cross-lingual zero-shot QE approaches are constantly studied (Sun et al., 2020; Eo et al., 2021b). Following this trend, WMT21 includes zero-shot to their main interests, evaluating the competence of a QE model on unseen languages (Specia et al., 2021). This study addresses all of the respective factors and presents QUAK.

3 QUAK

QUAK is a QE dataset for evaluating the quality of Korean-English MT output, and includes three sub-QUAKs according to the data selection. Each sub-QUAK comprises *(1) a source sentence, (2) its MT output, and (3) OK/BAD quality annotation*. Source sentence and its MT output are utilized as model input, allowing the model to classify the quality of these sentences into OK/BAD tags on a token basis. Quality annotations separately exists for source sentence and MT output. When the model predicts the translation quality, these are used as a ground-truth for evaluation. With primary consideration of efficiency and effectiveness, we construct data in a fully-automatic manner, exploited method by Eo et al. (2021a). We select existing monolingual or parallel corpora for our data sources (detailed in Section 3.1) and use them to each data production process (detailed in Section 3.2).

3.1 Dataset Sources

QUAK is divided into three sub-QUAKs. The raw dataset requirements to build each sub-QUAK are as follows: QUAK-M requires a target language monolingual corpus, QUAK-P requires a parallel corpus, QUAK-H requires both corpora.

For a monolingual corpus, we adopt English Wikipedia, which consists of documents on a wide range of topics. We use it to handle the various translation errors that the MT model may commit to the diverse entities and expressions in Wikipedia. We randomly extract 5M sentences to generate QUAK-M. For a parallel corpus, we leverage AI hub parallel corpus released by Korea National Information Society Agency[2]. AI hub corpus also covers various fields such as news, journals, law, and culture, and is produced with high quality through human inspection. The parallel corpus contains 1,602,002 pairs.

For fair comparison with other sub-QUAKs, in the case of QUAK-M, we combine both AI hub and Wikipedia source. Namely, we configure 1.58M of QUAK-M using the target-side text of the Ai hub and the remaining 5M using Wikipedia. The validation and test set is configured by randomly selecting 12K pieces of AI hub data.

3.2 Dataset Construction Process

QUAK-M For QUAK-M, we utilize a target language monolingual corpus. With the text, we first conduct a round-trip translation. We translate the English corpus into Korean sentences, where we denote them as pseudo-source sentences. We once again forward-translate the pseudo-source to generate the MT output.

When pseudo-source and its MT output have been created, quality annotations are tagged. Prior to label annotation, we pre-define target language monolingual corpus to be a flawless sentence. Based on this assumption, we consider this text as a pseudo-post-edited (pseudo-PE) sentences for which correction has been completed. By com-

[2] https://aihub.or.kr/

paring the MT output and pseudo-PE in a token-wise fashion, we measure the minimum substitution/deletion/insertion errors needed based on edit distance. The OK/BAD tag indicating correct/wrong translation for each token in the MT output is further annotated. If the number of MT output tokens in a sentence is N, the number of OK/BAD tags for this is $2N + 1$ because gap tokens are attached to the front and back of each MT output token. If there are missing words, BAD tags are added to the position of the corresponding gap token, otherwise OK tags are labeled. Apart from tagging the MT outputs, source tags are also annotated according to the binary tags of the MT outputs based on word alignment information. The tag for the source sentence excludes the gap token labeling.

QUAK-P The parallel corpus is leveraged in the QUAK-P configuration. This has higher connectivity between the source and target sides and has an intact source sentence compared with the pseudo-source of QUAK-M. To obtain QUAK-P, we proceed a one-way translation from the source to target language. In this case, source-side difference from QUAK-M leads to various translation results for the MT output. Similar to the QUAK-M generation process, we consider the target-side text of the parallel corpus as a pseudo-PE. With source sentences, its MT output, and pseudo-PE, we label the quality of the translation results. After calculating the minimum edit operation between the MT output and the pseudo-PE, BAD tags are attached to the token where the modification occurred. For quality annotations on source sentences, the same tags are attached to the MT output index and the aligned source index.

QUAK-H In QUAK-H, we combine the above two previous sub-QUAKs to generate various translation results with a limited corpus. We compose the source sentence and MT output by selectively utilizing two approaches proposed in QUAK-P and QUAK-M, respectively. Namely, we use the source-side text and pseudo-PE text from QUAK-P, and the MT output-side text from QUAK-M. By dealing with two different MT outputs with the same source-side text in QUAK-P, we induce the QE model to learn by referring to various combinations of the source sentence and MT output. For the next step, we tag labels for quality annotation as mentioned above.

Final constructed QUAK dataset After three construction processes, we obtain a total of 1,578,002 training examples for QUAK-P and QUAK-H, and 6,578,002 training examples for QUAK-M. We present an example of QUAK in Table 1. MT output for a source sentence "중국 당국이 부인하지 않는 것으로 볼 때 가능성이 높다 ." is mistranslated into "`Given that the Chinese authorities do not deny it, it is highly likely .`". The MT output should be corrected into "`chances are high`". This should perform a four minimum correction, which will result in a four BAD tags of the entire MT output tag. In addition, based on the word alignment "가능성(7)–`likely`(13), 높다(8)–`is`(11), 높다(8)–`highly`(12)", the BAD tag index of the MT output is also reflected in the source-side index.

4 Experimental Setup

4.1 Experimental Design

In this section, we present the statistical and quantitative analysis done on the QUAK. In the statistical analysis, we measure the sentence length, token length, and average token length per sentence for each sub-QUAK. We also calculate the mean, median, standard deviation, and variance of the translation edit rate (TER) score. Regarding tags, we count the total number of OK and BAD tags.

During the quantitative analysis, we experiment three word-level QE fine-tuning to efficiently achieve high performance and analyze large-scale QUAK data. In the first experiment, we fine-tune multiple mPLMs with 100K pieces of QUAK-M, P, H to explore which model performs better for QUAK.

Thereafter, we inspect the impact on the amount of QUAK. We fine-tune the data for the previous best performing model, scaling each sub-QUAK exponentially from 100K to 1.58M. As mentioned earlier, one consideration is that QUAK-M (1.58M) consists of target-side text in a parallel corpus for proper comparison with the data generated by other strategies.

In the last experiment, we gradually increase the size of QUAK-M. Our result includes the corresponding performance while extending from the previous size of 1.58M to 6.58M in 500K increments.

Attributes	Google	Amazon	Microsoft	Systran
# of Source Sentences	12,000	12,000	12,000	12,000
# of MT Output	12,000	12,000	12,000	12,000
# of pseudo-PE	12,000	12,000	12,000	12,000
# of Source Tokens	199,413	199,413	199,413	199,413
# of MT Output Tokens	340,264	303,535	325,973	346,030
# of pseudo-PE Tokens	342,385	342,385	342,385	342,385
Average Token Per Source Sentence	16.62	16.62	16.62	16.62
Average Token Per MT Output	28.36	25.29	27.16	28.84
Average Token Per pseudo-PE	28.53	28.53	28.53	28.53
Mean TER	0.57	0.63	0.63	0.46
Median TER	0.57	0.64	0.64	0.44
STD TER	0.23	0.21	0.21	0.26
Variance TER	0.05	0.04	0.05	0.07
# Source OK tags	112,647	94,562	97,825	134,503
# Source BAD tags	86,766	104,851	101,588	64,910
# MT Output OK tags	510,085	429,775	465,441	560,221
# MT Output BAD tags	182,443	189,295	198,505	143,839

Table 2: Statistics for the four test sets. We denote the target-side text of corpus as pseudo-PE.

4.2 Experimental Settings

Models In all experiments, we exploit the MicroTransQuest (Ranasinghe et al., 2021) framework. While it only uses an XLM-R model, we utilize additional mPLMs: In the QE model training, we leverage XLM, XLM-R, and mBART.

From Huggingface (Wolf et al., 2019), we load five mPLMs that have learned both Korean and English: xlm-mlm-100-1280, xlm-roberta-base, xlm-roberta-large, facebook/mbart-large-cc25, and facebook/mbart-large-50.

Datasets For the data construction, the following tools are used in this study. As monolingual data we dump Wikipedia and use Wikiextractor[3] to extract plain text. We train a Korean-English and English-Korean MT model using the fairseq (Ott et al., 2019) package with SentencePiece subword tokenization (Kudo and Richardson, 2018) to translate sentences. We train the word alignment between the source and target text using the FastAlign (Dyer et al., 2013) toolkit and measure the edit distance using Tercom software (Snover et al., 2006). Tag annotation is executed using the Unbabel corpus builder[4]. We adopt Mosesdecoder (Koehn et al., 2007) for additional data preprocessing.

Evaluation For constructing the test sets, we utilize a publicly available external machine translator to ensure the reliability and objectivity of the QE results. Four representative commercialized machine translators are adopted, including Google[5], Amazon[6], Microsoft[7], and Systran[8]. Through these, the test sets are established in the same manner as the strategy used in QUAK-P. The test sets are based on 12K sentence pairs randomly extracted from the Ai hub parallel corpus without overlapping with the training and validation sets. Matthews correlation coefficient (MCC) (Chicco and Jurman, 2020) is used as a metric for evaluating QE model performance.

Table 2 provides the test set statistics. When analyzing the number of OK/BAD tags, the results vary depending on the translator even when the same source sentence is used. Systran differs the most compared with other test sets: there are 143,839 MT output BAD tags with an average TER difference of 0.17 with the highest value of 0.63.

5 Analysis and Results

5.1 Data statistics and analysis

We report the statistics for QUAK in Table 3. QUAK-M additionally uses the English Wikipedia corpus consisting of 5M samples, and this yields different data sizes in contrast to other sub-QUAKs. Comparing training set of QUAK-M with other sub-QUAKs, the most dominant part is the relation between the average token length and TER score. QUAK-P and QUAK-H show lower TER scores even though their average tokens per sentence are relatively higher. We interpret this result as a case where the translation works well even if the average sentence length is long. As shown in Figure 1, comparing data sizes by TER range is also consistent with this statistic. The data is mainly concentrated in the 0.5–0.8 range for QUAK-M, 0.3–0.6 for QUAK-P, and 0.2–0.5 for QUAK-H. From this, we speculate that Wikipedia may have more noise in the text itself than the Ai hub, and that a large number of errors are committed during the translation process.

Next, for QUAK-P and QUAK-H, the number of BAD tags of QUAK-P is greater than that of QUAK-H in the MT output. It is noteworthy that the MT output of QUAK-H is created based on a round-trip translation, which is identical to that of QUAK-M. These indicate that the pseudo-source generated by the target language text of Ai hub is adequately restored to the original sentence when

[3] https://github.com/attardi/wikiextractor

[4] https://github.com/Unbabel/word-level-qe-corpus-builder

[5] https://translate.google.co.kr/?hl=en

[6] https://aws.amazon.com/translate/

[7] https://www.microsoft.com/en-us/translator/

[8] https://translate.systran.net/

	Train			Valid		
Attributes	**QUAK-M**	**QUAK-P**	**QUAK-H**	**QUAK-M**	**QUAK-P**	**QUAK-H**
# of Source Sentences	6,578,002	1,578,002	1,578,002	12,000	12,000	12,000
# of MT Output	6,578,002	1,578,002	1,578,002	12,000	12,000	12,000
# of pseudo-PE	6,578,002	1,578,002	1,578,002	12,000	12,000	12,000
# of Source Tokens	92,848,776	25,149,673	25,149,673	209,894	199,624	199,624
# of MT Output Tokens	139,620,328	42,051,001	39,850,492	318,959	340,855	318,921
# of pseudo-PE Tokens	148,922,086	42,103,966	42,103,966	342,021	341,996	341,996
Average Token Per Source Sentence	14.12	15.94	15.94	17.50	16.64	16.64
Average Token Per MT Output	21.23	26.65	25.25	26.58	28.40	26.58
Average Token Per pseudo-PE	22.64	26.68	26.68	28.50	28.50	28.50
Mean TER	0.61	0.50	0.42	0.45	0.57	0.45
Median TER	0.63	0.50	0.40	0.44	0.56	0.44
STD TER	0.24	0.25	0.23	0.21	0.23	0.21
Variance TER	0.06	0.06	0.05	0.04	0.05	0.04
# Source OK tags	50,421,860	15,600,416	16,269,784	84,873	113,192	121,700
# Source BAD tags	42,426,916	9,549,257	8,879,889	82,343	86,432	77,924
# MT Output OK tags	204,721,896	65,852,185	65,033,087	339,304	511,542	506,437
# MT Output BAD tags	81,096,762	19,827,819	16,245,899	157,308	182,168	143,405

Table 3: Statistics for three sub-QUAK training and validation set

translated back, even if it is different from the correctly translated source-side of the parallel corpus.

5.2 Experimental Results

Performance Comparison by mPLMs We provide the fine-tuning results for mPLMs by selecting only 100K of the datasets in Table 4. From the experimental results, XLM-R-large model shows the best performance. Based on the MT output-side MCC (Target MCC) of the Google test set, XLM-R-large reports 0.366, 0.401, and 0.324 for QUAK-M,P,H, and outperforms the XLM-R-base model by 0.023, 0.024, and 0.004, respectively. For the source-side MCC (Source MCC), XLM-R-large also achieves 0.285, 0.331, and 0.271 for QUAK-M,P,H, which are the best competencies compared to other models.

In all test sets, except for Systran, the Target MCC and Source MCC of XLM-R-large performed the best in all sub-QUAK datasets. XLM-R-large differs from XLM-R-base in terms of the number of parameters; the former contains 550M, whereas the latter has 270M. Based on the Target MCC of the Amazon test set, XLM-R-large generally reports a higher performance than XLM-R-base, achieving 0.035, 0.032, and 0.022 higher values for QUAK-M, P, and H, respectively. These results demonstrate that the number of parameters in mPLMs poses a positive effect on the QE model learning.

Regarding mBART and mBART50, the latter outperforms the former in general. This implies the substantial impact of the number of pre-trained

	Dataset	XLM-R -base	XLM-R -large	mBART	mBART50	XLM
Google						
Target MCC	QUAK-M	0.343	**0.366**	0.340	0.343	0.296
	QUAK-P	0.377	**0.401**	0.376	0.382	0.339
	QUAK-H	0.320	**0.324**	0.306	0.314	0.292
Source MCC	QUAK-M	0.279	**0.285**	0.275	0.276	0.231
	QUAK-P	0.315	**0.331**	0.309	0.320	0.285
	QUAK-H	0.266	**0.271**	0.258	0.267	0.249
Amazon						
Target MCC	QUAK-M	0.389	**0.424**	0.388	0.385	0.328
	QUAK-P	0.408	**0.440**	0.405	0.410	0.362
	QUAK-H	0.362	**0.384**	0.264	0.359	0.322
Source MCC	QUAK-M	0.324	**0.342**	0.320	0.323	0.254
	QUAK-P	0.353	**0.377**	0.341	0.354	0.304
	QUAK-H	0.305	**0.323**	0.213	0.310	0.276
Microsoft						
Target MCC	QUAK-M	0.380	**0.415**	0.382	0.380	0.315
	QUAK-P	0.401	**0.433**	0.404	0.406	0.346
	QUAK-H	0.353	**0.372**	0.253	0.355	0.316
Source MCC	QUAK-M	0.307	**0.329**	0.303	0.307	0.244
	QUAK-P	0.338	**0.363**	0.327	0.338	0.287
	QUAK-H	0.290	**0.310**	0.193	0.299	0.271
Systran						
Target MCC	QUAK-M	0.261	**0.277**	0.255	0.253	0.206
	QUAK-P	0.298	**0.311**	0.289	0.296	0.261
	QUAK-H	**0.226**	0.217	0.122	0.221	0.196
Source MCC	QUAK-M	**0.224**	0.223	0.218	0.221	0.161
	QUAK-P	0.247	**0.250**	0.228	0.242	0.210
	QUAK-H	**0.179**	0.174	0.076	0.176	0.159

Table 4: Comparison of word-level Korean-English QE performance by mPLMs fine-tuned with each sub-QUAK dataset

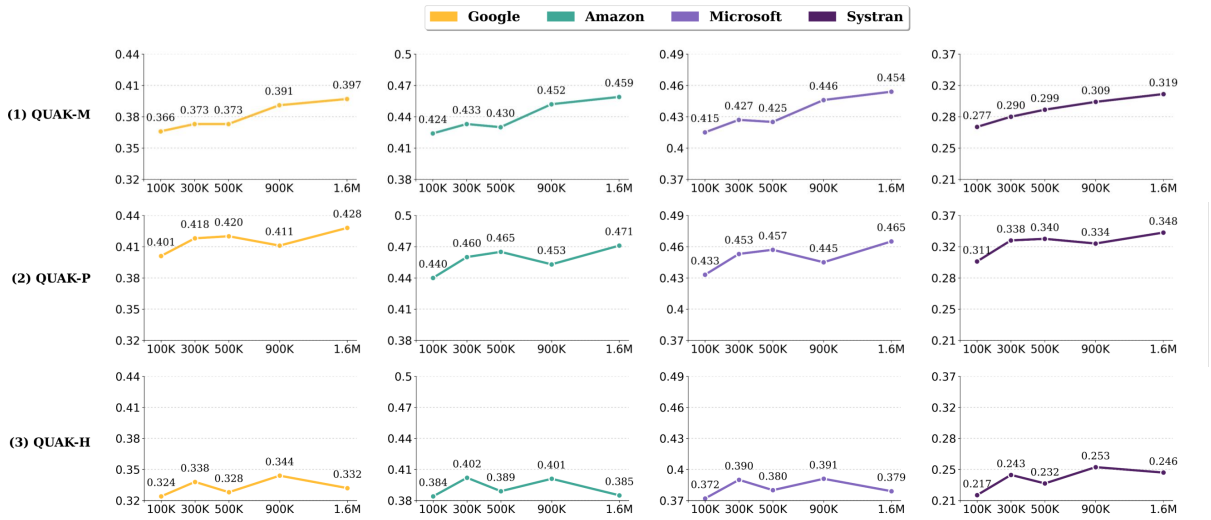

Figure 2: MCC variation of QUAK according to data scaling

	Google		Amazon		Microsoft		Systran	
Data Size	Target MCC	Source MCC	Target MCC	Source MCC	Target MCC	Source MCC	Target MCC	Source MCC
1.58M	0.397	0.324	0.459	0.386	0.454	0.376	0.319	0.273
2.08M	0.386	0.319	0.441	0.371	0.436	0.365	0.317	0.272
2.58M	0.384	0.316	0.442	0.372	0.436	0.364	0.308	0.264
3.08M	0.385	0.316	0.442	0.375	0.435	0.365	0.313	0.272
3.58M	0.381	0.314	0.443	0.374	0.437	0.365	0.311	0.269
4.08M	0.382	0.313	0.438	0.369	0.434	0.359	0.310	0.270
4.58M	0.376	0.310	0.433	0.368	0.426	0.357	0.307	0.267
5.08M	0.378	0.308	0.436	0.366	0.432	0.362	0.308	0.269
5.58M	0.377	0.319	0.439	0.374	0.432	0.364	0.307	0.274
6.08M	0.351	0.286	0.400	0.335	0.398	0.334	0.296	0.261
6.58M	0.387	0.325	0.451	0.379	0.441	0.369	0.313	0.274

Table 5: Performance variation of QUAK-M according to data scaling

languages. It is noteworthy that mBART50 is pre-trained for 50 languages, enabling more multilingual support than mBART, which is learned on 25 languages. As Korean is regarded as a relatively low-resource language and especially utilizes only 100K data, we infer that mBART50 has more influence on competence gain from high-resource languages than mBART.

Performance Comparison for Scaling The previously obtained results show that the XLM-R-large model is superior to all 100K sub-QUAK datasets. For the next experiment, we explore the performance fluctuation by constantly increasing the size of the QUAK dataset to XLM-R-large. Figure 2 illustrates the variation of the performance depending on the corpus size. The experimental results demonstrate that the performance variation tends to be similar for all test sets.

When we exponentially scale the data for the three sub-QUAKs, QUAK-M had a notable achievement. Furthermore, QUAK-P showed a steady increase, except for the case of 900K. We confirm that data scaling is one factor in increasing the performance of the QE model. However, in the case of QUAK-H, there is no clear trend in terms of data expansion. We argue that although the MT output is applied owing to various translations for source sentences, the weakened connectivity between two sentences might impede learning.

Performance of QUAK-M (6.58M) QUAK-M requires only monolingual corpus in the data building process. This allows data size expansion over other sub-QUAKs that utilize a parallel corpus. Exploiting these, we further extend the Wikipedia corpus by 5M, comprising a total of 6.58M. We gradually add data in 500K increments to check the performance fluctuation.

The experimental result is presented in Table 5. The target MCC performance on the Google test set with 1.58M is lower by -0.016 at 3.58M(+2M) and -0.020 at 5.58M(+4M). The performance of Amazon, Microsoft, and Systran also degraded by -0.016, -0.017, -0.008 at 3.58M and -0.02, -0.022, and -0.012 at 5.58M compared to 1.58M, respectively. We observe that the overall QE model performance has deteriorated as more data is added.

We interpret this result in terms of data. QUAK-H, P, and the test sets are extracted from the Ai hub dataset. As observed in the previous statistics (Table 3), this resulted in a difference in terms of average TER in QUAK-M, which also contains

	TER Range									
Data Size	**0.0-0.1**	**0.1-0.2**	**0.2-0.3**	**0.3-0.4**	**0.4-0.5**	**0.5-0.6**	**0.6-0.7**	**0.7-0.8**	**0.8-0.9**	**0.9-1.0**
QUAK-P										
100K	0.214	0.323	0.361	0.398	0.410	0.382	0.374	0.338	0.308	0.262
1.58M	0.209	0.348	0.386	0.420	0.438	0.413	0.395	0.377	0.325	0.279
Diff	-0.005	0.025	0.025	0.022	**0.028**	**0.031**	0.021	**0.039**	0.017	0.017
QUAK-H										
100K	0.206	0.312	0.335	0.378	0.375	0.337	0.312	0.263	0.237	0.172
1.58M	0.192	0.316	0.330	0.366	0.374	0.337	0.309	0.275	0.249	0.192
Diff	-0.014	0.004	-0.005	-0.012	-0.001	0.00	-0.003	**0.012**	**0.012**	**0.020**
QUAK-M (1.58M)										
100K	0.210	0.295	0.342	0.380	0.393	0.361	0.339	0.303	0.269	0.223
1.58M	0.215	0.329	0.361	0.398	0.413	0.381	0.366	0.340	0.312	0.265
Diff	0.005	0.034	0.019	0.018	0.020	0.020	0.027	**0.037**	**0.043**	**0.042**
QUAK-M (6.58M)										
100K	0.210	0.295	0.342	0.380	0.393	0.361	0.339	0.303	0.269	0.223
6.58M	0.184	0.308	0.352	0.400	0.399	0.377	0.364	0.339	0.298	0.261
Diff	-0.026	0.013	0.010	0.020	0.006	0.016	0.025	**0.036**	**0.029**	**0.038**

Table 6: Target MCC performance difference (Diff) by TER range for Google test set. When we add data, we underline the three cases with the worst performance, and bold the three cases with the most performance improvement.

Wikipedia. QUAK-M is mainly distributed in a range with a high TER score, while QUAK-H, QUAK-P, and the test set are included in relatively low scores. This indicates that the difference from the test set in terms of data distribution also affected the performance of QUAK-M.

Performance Comparison by TER Range In addition to the previous results, we divide the Google test set into units of 0.1 TER for more precise comparison. We then verify the changes in performance with the TER range. The experimental result is present in Table 6, from which we note that the performance on QUAK-M (6.58M) shows an overall improvement compared with those on QUAK-M (100K), and mainly improves between 0.7–1.0. The highest increase for QUAK-M (1.58M) is also seen between 0.7–1.0. As shown in Figure 1, QUAK-M is mainly distributed in the high TER range. Although both QUAK-M (1.58M) and QUAK-M (6.58M) show performance gains over 100K, QUAK-M (6.58M) reports that the performance improvement is not significant at the relatively low TER. We analyze that this in turn, leads to performance degradation of the integrated score compared with 1.58M. This is supported by the fact that even in the TER range of 0.1–0.2, the performance fluctuation of QUAK-M (6.58M) is remarkably lower than that of QUAK-M (1.58M).

In QUAK-P, the amount of data is the lowest at 0.0–0.2 and 0.8–1.0. Therefore, when adding data, the performance variation also shows a lower increase compared with other scores in the range of 0.0–0.1 and 0.8–1.0. From the above results, we conclude that the amount of data can be a contributing factor for performance improvement.

6 Conclusion

We expose three drawbacks in terms of manual QE data construction: human labor and time cost, resulting in limited amount of data and limited language pairs. Taking this into account, we present QUAK, a synthetic Korean-English QE dataset for word-level QE. We automatically generated three sub-QUAKs with three strategies and quantitatively analyzed the trained QE models using them. First, QUAK-P is generated based on parallel corpus and induced the best performance among three sub-QUAKs. Along with QUAK-P, an increase in the data size of QUAK-M had a positive effect on performance gain. However, in further expansion using Wikipedia, the improvement in the low TER range was poor, so the overall performance fell. The QE model trained with QUAK-H did not show a steady performance gain.

This dataset was built in a fully automated manner, eliminating human intervention while increasing reusability and scalability. The QUAK dataset generation process is language-agnostic if there is an MT model and corresponding corpus (monolingual or parallel).

7 Acknowledgements

This work was supported by Institute of Information communications Technology Planning Evaluation(IITP) grant funded by the Korea government(MSIT) (No. 2020-0-00368, A Neural-Symbolic Model for Knowledge Acquisition and Inference Techniques), and supported by the MSIT, Korea, under the Information Technology Research Center(ITRC) support program(IITP-2022-2018-0-01405) supervised by the IITP, and supported by Basic Science Research Program through the National Research Foundation of Korea(NRF) funded by the Ministry of Education(NRF-2021R1A6A1A03045425).

References

Fernando Alva-Manchego, Abiola Obamuyide, Amit Gajbhiye, Frédéric Blain, Marina Fomicheva, and Lucia Specia. 2021. deepquest-py: Large and distilled models for quality estimation. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 382–389.

John Blatz, Erin Fitzgerald, George Foster, Simona Gandrabur, Cyril Goutte, Alex Kulesza, Alberto Sanchis, and Nicola Ueffing. 2004. Confidence estimation for machine translation. In *Coling 2004: Proceedings of the 20th international conference on computational linguistics*, pages 315–321.

Tom B Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. 2020. Language models are few-shot learners. *arXiv preprint arXiv:2005.14165*.

Davide Chicco and Giuseppe Jurman. 2020. The advantages of the matthews correlation coefficient (mcc) over f1 score and accuracy in binary classification evaluation. *BMC genomics*, 21(1):1–13.

Alexis Conneau, Kartikay Khandelwal, Naman Goyal, Vishrav Chaudhary, Guillaume Wenzek, Francisco Guzmán, Edouard Grave, Myle Ott, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Unsupervised cross-lingual representation learning at scale. *arXiv preprint arXiv:1911.02116*.

Shuoyang Ding, Marcin Junczys-Dowmunt, Matt Post, Christian Federmann, and Philipp Koehn. 2021a. The jhu-microsoft submission for wmt21 quality estimation shared task.

Shuoyang Ding, Marcin Junczys-Dowmunt, Matt Post, and Philipp Koehn. 2021b. Levenshtein training for word-level quality estimation. *arXiv preprint arXiv:2109.05611*.

Chris Dyer, Victor Chahuneau, and Noah A Smith. 2013. A simple, fast, and effective reparameterization of ibm model 2. In *Proceedings of the 2013 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 644–648.

Sugyeong Eo, Chanjun Park, Hyeonseok Moon, Jaehyung Seo, and Heuiseok Lim. 2021a. Dealing with the paradox of quality estimation. In *Proceedings of the 4th Workshop on Technologies for MT of Low Resource Languages (LoResMT2021)*, pages 1–10, Virtual. Association for Machine Translation in the Americas.

Sugyeong Eo, Chanjun Park, Jaehyung Seo, Hyeonseok Moon, and Heuiseok Lim. 2021b. Study on zero-shot based quality estimation. *Journal of the Korea Convergence Society*, 12(11):35–43.

Thierry Etchegoyhen, Eva Martínez Garcia, and Andoni Azpeitia. 2018. Supervised and unsupervised minimalist quality estimators: Vicomtech's participation in the wmt 2018 quality estimation task. In *Proceedings of the Third Conference on Machine Translation: Shared Task Papers*, pages 782–787.

Marina Fomicheva, Lucia Specia, and Francisco Guzmán. 2020a. Multi-hypothesis machine translation evaluation. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 1218–1232.

Marina Fomicheva, Shuo Sun, Erick Fonseca, Chrysoula Zerva, Frédéric Blain, Vishrav Chaudhary, Francisco Guzmán, Nina Lopatina, Lucia Specia, and André FT Martins. 2020b. Mlqe-pe: A multilingual quality estimation and post-editing dataset. *arXiv preprint arXiv:2010.04480*.

Marina Fomicheva, Shuo Sun, Lisa Yankovskaya, Frédéric Blain, Francisco Guzmán, Mark Fishel, Nikolaos Aletras, Vishrav Chaudhary, and Lucia Specia. 2020c. Unsupervised quality estimation for neural machine translation. *Transactions of the Association for Computational Linguistics*, 8:539–555.

Atsushi Fujita and Eiichiro Sumita. 2017. Japanese to english/chinese/korean datasets for translation quality estimation and automatic post-editing. In *Proceedings of the 4th Workshop on Asian Translation (WAT2017)*, pages 79–88.

Amit Gajbhiye, Marina Fomicheva, Fernando Alva-Manchego, Frédéric Blain, Abiola Obamuyide, Nikolaos Aletras, and Lucia Specia. 2021. Knowledge distillation for quality estimation. *arXiv preprint arXiv:2107.00411*.

Hyun Kim, Jong-Hyeok Lee, and Seung-Hoon Na. 2017. Predictor-estimator using multilevel task learning with stack propagation for neural quality estimation. In *Proceedings of the Second Conference on Machine Translation*, pages 562–568.

Philipp Koehn, Hieu Hoang, Alexandra Birch, Chris Callison-Burch, Marcello Federico, Nicola Bertoldi, Brooke Cowan, Wade Shen, Christine Moran, Richard Zens, et al. 2007. Moses: Open source toolkit for statistical machine translation. In *Proceedings of the 45th annual meeting of the association for computational linguistics companion volume proceedings of the demo and poster sessions*, pages 177–180.

Taku Kudo and John Richardson. 2018. Sentencepiece: A simple and language independent subword tokenizer and detokenizer for neural text processing. *arXiv preprint arXiv:1808.06226*.

Guillaume Lample and Alexis Conneau. 2019. Cross-lingual language model pretraining. *arXiv preprint arXiv:1901.07291*.

Dongjun Lee. 2020. Two-phase cross-lingual language model fine-tuning for machine translation quality estimation. In *Proceedings of the Fifth Conference on Machine Translation*, pages 1024–1028.

Yinhan Liu, Jiatao Gu, Naman Goyal, Xian Li, Sergey Edunov, Marjan Ghazvininejad, Mike Lewis, and Luke Zettlemoyer. 2020. Multilingual denoising pre-training for neural machine translation. *Transactions of the Association for Computational Linguistics*, 8:726–742.

Matteo Negri, Marco Turchi, Rajen Chatterjee, and Nicola Bertoldi. 2018. Escape: a large-scale synthetic corpus for automatic post-editing. *arXiv preprint arXiv:1803.07274*.

Myle Ott, Sergey Edunov, Alexei Baevski, Angela Fan, Sam Gross, Nathan Ng, David Grangier, and Michael Auli. 2019. fairseq: A fast, extensible toolkit for sequence modeling. *arXiv preprint arXiv:1904.01038*.

Tharindu Ranasinghe, Constantin Orasan, and Ruslan Mitkov. 2021. An exploratory analysis of multilingual word-level quality estimation with cross-lingual transformers. *arXiv preprint arXiv:2106.00143*.

Matthew Snover, Bonnie Dorr, Richard Schwartz, Linnea Micciulla, and John Makhoul. 2006. A study of translation edit rate with targeted human annotation. In *Proceedings of association for machine translation in the Americas*, volume 200. Citeseer.

Lucia Specia, Frédéric Blain, Marina Fomicheva, Chrysoula Zerva, Zhenhao Li, Vishrav Chaudhary, and André Martins. 2021. Findings of the wmt 2021 shared task on quality estimation. Association for Computational Linguistics.

Lucia Specia, Nicola Cancedda, and Marc Dymetman. 2010. A dataset for assessing machine translation evaluation metrics. In *LREC*.

Lucia Specia, Kashif Shah, Jose G.C. de Souza, and Trevor Cohn. 2013. QuEst - a translation quality estimation framework. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics: System Demonstrations*, pages 79–84, Sofia, Bulgaria. Association for Computational Linguistics.

Lucia Specia, Marco Turchi, Nicola Cancedda, Marc Dymetman, and Nello Cristianini. 2009. Estimating the sentence-level quality of machine translation systems. In *13th Conference of the European Association for Machine Translation*, pages 28–37.

Shuo Sun, Marina Fomicheva, Frédéric Blain, Vishrav Chaudhary, Ahmed El-Kishky, Adithya Renduchintala, Francisco Guzmán, and Lucia Specia. 2020. An exploratory study on multilingual quality estimation. Association for Computational Linguistics.

Yi-Lin Tuan, Ahmed El-Kishky, Adithya Renduchintala, Vishrav Chaudhary, Francisco Guzmán, and Lucia Specia. 2021. Quality estimation without human-labeled data. *arXiv preprint arXiv:2102.04020*.

Minghan Wang, Hao Yang, Hengchao Shang, Daimeng Wei, Jiaxin Guo, Lizhi Lei, Ying Qin, Shimin Tao, Shiliang Sun, Yimeng Chen, et al. 2020. Hw-tsc's participation at wmt 2020 quality estimation shared task. In *Proceedings of the Fifth Conference on Machine Translation*, pages 1056–1061.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Rémi Louf, Morgan Funtowicz, et al. 2019. Huggingface's transformers: State-of-the-art natural language processing. *arXiv preprint arXiv:1910.03771*.

Improving Both Domain Robustness and Domain Adaptability in Machine Translation

Wen Lai[1] and **Jindřich Libovický**[2] and **Alexander Fraser**[1]
[1] Center for Information and Language Processing, LMU Munich, Germany
[2] Faculty of Mathematics and Physics, Charles Univeristy, Prague, Czech Republic
{lavine, fraser}@cis.lmu.de libovicky@ufal.mff.cuni.cz

Abstract

We consider two problems of NMT domain adaptation using meta-learning. First, we want to reach domain *robustness*, i.e., we want to reach high quality on both domains seen in the training data and unseen domains. Second, we want our systems to be *adaptive*, i.e., making it possible to finetune systems with just hundreds of in-domain parallel sentences. We study the domain adaptability of meta-learning when improving the domain robustness of the model. In this paper, we propose a novel approach, **RML-NMT** (**R**obust **M**eta-**L**earning Framework for **N**eural **M**achine **T**ranslation Domain Adaptation), which improves the robustness of existing meta-learning models. More specifically, we show how to use a domain classifier in curriculum learning and we integrate the word-level domain mixing model into the meta-learning framework with a balanced sampling strategy. Experiments on English→German and English→Chinese translation show that RMLNMT improves in terms of both domain robustness and domain adaptability in seen and unseen domains[1].

1 Introduction

The success of Neural Machine Translation (NMT; Bahdanau et al., 2015; Vaswani et al., 2017) heavily relies on large-scale high-quality parallel data, which is difficult to obtain in some domains. We study two major problems in NMT domain adaptation. First, models should work well on both seen domains (the domains in the training data) and unseen domains (domains which do not occur in the training data). We call this property *domain robustness*. Second, with just hundreds of in-domain sentences, we want to be able to quickly adapt to a new domain. We call this property *domain adaptability*. Previous work on NMT domain adaptation has usually focused on only one aspect of domain adaptation at the expense of the other one, and our motivation is to consider both of the two properties.

There are a few works attempting to solve domain adaptability. The most basic approach is *fine-tuning*, in which an out-of-domain model is continually trained on in-domain data (Freitag and Al-Onaizan, 2016; Dakwale and Monz, 2017). Although fine-tuning is effective, it can suffer from so-called catastrophic forgetting (French, 1999), resulting in deteriorated model performance in general domains (Thompson et al., 2019). Another efficient method is *Meta-Learning* (Hospedales et al., 2021), which trains models which can be later rapidly adapted to new scenarios using only a small amount of data. It works for many natural language processing (NLP) tasks (Gu et al., 2018; Qian and Yu, 2019; Yu et al., 2020; Bansal et al., 2020; Wang et al., 2021; Du et al., 2021), especially in low-resource scenarios (Dou et al., 2019; Yin, 2020). As a result, meta-learning is often used for NMT domain adaptation. For example, Sharaf et al. (2020) and Li et al. (2020) fast adapt NMT models to new domains with meta-learning using a small amount of training data. Zhan et al. (2021) improve meta-learning-based NMT models with a curriculum-based (Bengio et al., 2009) sampling strategy. Meta-learning works well for adapting to new domains, however, previous work tends to neglect the problem of robustness towards domains unseen at training time.

Müller et al. (2020) defined the concept of domain robustness and propose to improve the domain robustness by subword regularization (Kudo, 2018), defensive distillation (Papernot et al., 2016), reconstruction (Tu et al., 2017) and neural noisy channel reranking (Yee et al., 2019). Jiang et al. (2020) proposed using individual modules for each domain with a word-level domain mixing strategy, which they showed has domain robustness on seen domains. The work on domain robustness, however, tends to neglect the adaptability of the models

[1]Our source code is available at https://github.com/lavine-lmu/RMLNMT

Proceedings of the 29th International Conference on Computational Linguistics, pages 5191–5204
October 12–17, 2022.

for new domains.

To address both domain adaptability and domain robustness at the same time, we propose RMLNMT (robust meta-learning NMT), a more robust meta-learning-based NMT domain adaptation framework. We first train a word-level domain mixing model to improve the robustness on seen domains, and show that, surprisingly, this improves robustness on unseen domains as well. Then, we train a domain classifier based on BERT (Devlin et al., 2019) to score training sentences; the score measures similarity between out-of-domain and general-domain sentences. This score is used to determine a curriculum to improve the meta-learning process. Finally, we improve domain adaptability by integrating the domain-mixing model into a meta-learning framework with the domain classifier using a balanced sampling strategy.

In summary, we make the following contributions: i) we propose RMLNMT, which shows better domain robustness and domain adaptability than all previous baseline systems; ii) we show that unseen domains can be very effectively handled with domain-robust models, even though post-hoc adaptation with domain-specific data still delivers the best overall translation quality; iii) Experiments on English→German and English→Chinese translation tasks show the effectiveness of RMLNMT. To the best of our knowledge, this is the first work that considers both domain adaptability and domain robustness in NMT domain adaptation, a combination which we suggest the community pay more attention to.

2 Preliminaries

Neural Machine Translation. The goal of the NMT model is to model the conditional distribution of translated sentence $y = (y_1, ..., y_n)$ given a source sentence $x = (x_1, ..., x_m)$. Current state-of-art NMT models (Transformers; Vaswani et al., 2017) model the multi-head attention mechanism to focus on information in different representation subspaces from different positions

$$\text{MultiHead}(Q, K, V) = \text{Concat}\left(h_1, \ldots, h_{\text{h}}\right) W^O$$

$$h_i = \text{Attention}\left(QW_i^Q, KW_i^K, VW_i^V\right),$$

where $W_i^Q, W_i^K, W_i^V \in \mathbb{R}^{d \times d/m}$ and $W^O \in \mathbb{R}^{d \times d}$. For the i-th head h_i, m is the number of heads, and d is the dimension of the model output. In some of our experiments (see Section 3.1), we modify the multihead attention to do domain mixing (Jiang et al., 2020).

Meta-learning for NMT. The goal of Meta-Learning is training a teacher model that using previous experience can be better finetuned for new tasks, including handling different domains in NMT domain adaptation (Gu et al., 2018; Sharaf et al., 2020; Zhan et al., 2021). The idea of NMT domain adaptation with meta-learning is to use a small set of source tasks $\{\mathcal{T}_1, \ldots, \mathcal{T}_n\}$ (which correspond to domains) to find the initialization of model parameters θ from which finetuning for task $\mathcal{T}_0$ would require only a small number of training examples. These meta-learning algorithms consist of three main steps: (i) split the seen domain corpus into small tasks $\mathcal{T}$ containing a small amount of data as $\mathcal{D}_{\text{meta-train}}$ and $\mathcal{D}_{\text{meta-test}}$ to simulate the low-resource scenarios. Data for each task $\mathcal{T}_i$ is decomposed into two sub-sets: a support set $\mathcal{T}_{support}$ used for training the model and a query set $\mathcal{T}_{query}$ used for evaluating the model; (ii) leverage a meta-learning policy to adapt model parameters to different small tasks using $\mathcal{D}_{\text{meta-train}}$ datasets. We use MAML, proposed by Finn et al. (2017), to create adaptable NMT systems which will be useful for different domains; (iii) finetune the model using the support set of $\mathcal{D}_{\text{meta-test}}$.

3 Method

In our initial experiments, we observed that the standard meta-learning approach for NMT domain adaptation sacrifices the domain robustness on seen domains in order to improve the domain adaptability on unseen domains. To address these issues, we propose a novel approach, RMLNMT, which combines meta-learning with a word-level domain-mixing system (for improving domain robustness) in a single model. RMLNMT consists of three parts: Word-Level Domain Mixing, Domain Classification, and Online Meta-Learning. Figure 1 illustrates RMLNMT.

3.1 Word-level Domain Mixing

In order to improve the robustness of NMT domain adaptation, we follow the approach of Jiang et al. (2020) and train a word-level layer-wise domain mixing NMT model.

Domain Proportion. From a sentence-level perspective (i.e., the classifier-based curriculum step), each sentence has a domain label. However, the

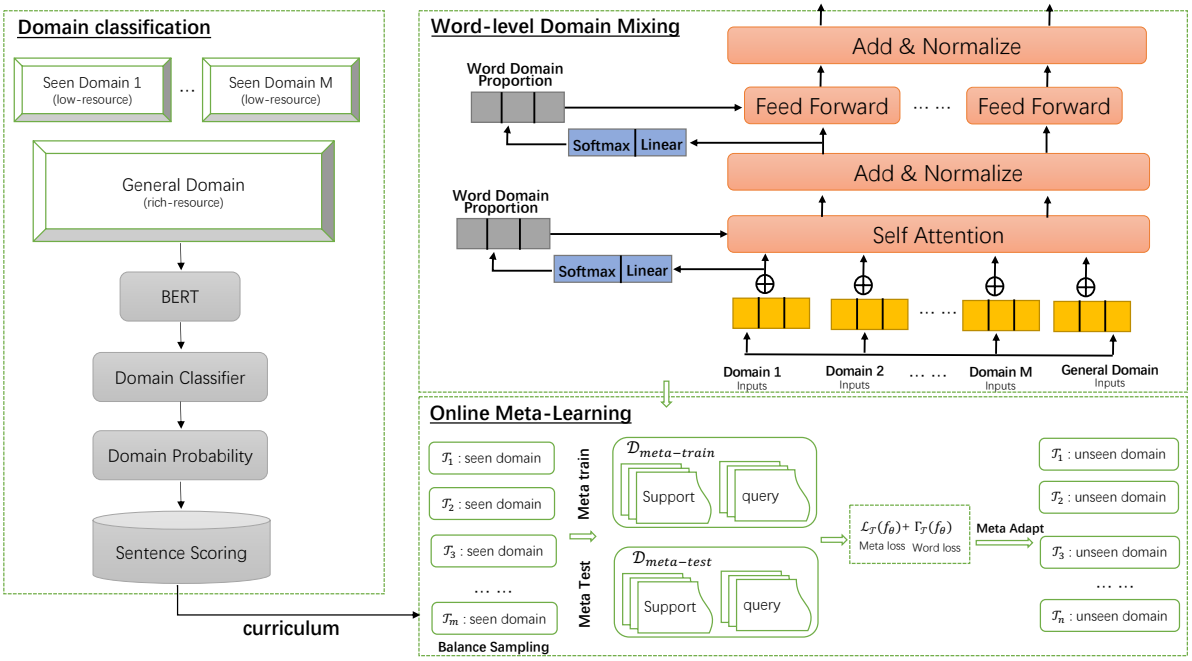

Figure 1: Method overview. The whole procedure mainly consists of three parts: domain classification, word-level domain mixing and online meta-learning.

domain of a word in the sentence is not necessarily consistent with the sentence domain. E.g., the word *doctor* can have a different meaning in the medical domain and the academic domain. More specifically, for k domains, the embedding $\mathbf{w} \in \mathbb{R}^d$ of a word, and a matrix $R \in \mathbb{R}^{k \times d}$, the domain proportion of the word is represented by a smoothed softmax function as:

$$\Phi(\mathbf{w}) = (1 - \epsilon) \cdot \text{softmax}(R\mathbf{w}) + \epsilon/k,$$

where $\epsilon \in (0, 1)$ is a smoothing parameter to prevent the output of $\Phi(\mathbf{w})$ from collapsing towards 0 or 1.

Domain Mixing. Following Jiang et al. (2020), each domain has its own multi-head attention modules. Therefore, we can integrate the domain proportion of each word into its multi-head attention module. Specifically, we take the weighted average of the linear transformation based on the domain proportion Φ. For example, we consider the pointwise linear transformation $\{W_{i,V,j}\}_{j=1}^k$ on the t-th word of the input, V_t, of all domains. The mixed linear transformation can be written as

$$\bar{V}_{i,t} = \sum_{j=1}^{k} V_t^\top W_{i,V,j} \Phi_{V,j}\left(V_t\right),$$

where $\Phi_{V,j}\left(V_t\right)$ denotes the j-th entry of $\Phi_V\left(V_t\right)$, and Φ_V is the domain proportion layer related to V. For other linear transformations, we apply the domain mixing scheme in the same way for all attention layers and the fully-connected layers.

Training. The model can be efficiently trained by minimizing a composite loss:

$$L^* = L_{\text{gen}}(\theta) + L_{\text{mix}}(\theta),$$

where θ contains the parameter in encoder, decoder and domain proportion. $L_{\text{gen}}\left(\theta\right)$ denotes the cross-entropy loss over training data $\{\mathbf{x}_i, \mathbf{y}_i\}_{i=1}^n$ and $L_{\text{mix}}(\theta)$ denotes the cross-entropy loss over the words/domain labels. For $L_{\text{mix}}(\theta)$, we compute the cross-entropy loss of its domain proportion $\Phi(\mathbf{w})$ as $-\log\left(\Phi_J(\mathbf{w})\right)$, which take J as the domain label. Hence, $L_{\text{mix}}(\theta)$ is computed as the sum of the cross-entropy loss over all such pairs of word labels of the training data.

3.2 Domain Classification

Domain similarity has been successfully applied in NMT domain adaptation. Moore and Lewis (2010) calculate cross-entropy scores with a language model to represent the domain similarity. Rieß et al. (2021) leverage simple classifiers to compute similarity scores; these scores are more effective than scores from language models for NMT domain adaptation. Motivated by Rieß et al. (2021),

we compute domain similarity using a sentence-level classifier, but in contrast with previous work, we based our classifier on a pre-trained language model. Given k domain corpora (one general domain corpus and n out-of-domain corpora), we trained a sentence classification model M based on BERT (Devlin et al., 2019). For a sentence x with a domain label L_x, a simple softmax is added to the top of the model M to predict the domain probability of sentence x:

$$P(x \mid h) = \text{softmax}(Wh),$$

where W is the parameter matrix of M and h is the hidden state of M. $P(x \mid h)$ is a probability set, which contains k probability scores indicating the similarity of sentence x to each domain. We finally select the probability of the general domain (from k probability scores) as the score of the sentence x and use this score as the curriculum to split the task in meta-learning (see more details in Section 3.3). A higher score indicates that the sentence is more similar to the general domain, so we will select it earlier.

3.3 Online Meta-Learning

After training the word-level domain mixing NMT model, we use it as a teacher model to initialize the meta-learning process. Algorithm 1 shows the complete algorithm.

Split Tasks. Zhan et al. (2021) propose a curriculum-based task splitting strategy, which uses divergence scores computed by a language model as the curriculum to split the corpus into small tasks. We follow a similar idea, but propose to use predictions from a domain classifier as the criterion for splitting the data. Concretely, we first train a domain classifier with BERT; the classifier scores sentences, indicating domain similarity between an in-domain sentence and a general domain sentence (see Section 3.2). The tasks are then split according to the scores; sentences more similar to the general domain sentences are selected in early tasks.

Balanced Sampling. Previous meta-learning approaches (Sharaf et al., 2020; Zhan et al., 2021) are based on token-size based sampling, which uses $8k$ or $16k$ token sizes split into many small tasks. However, the splitting process for the domain is not balanced, since some tasks did not contain all seen domains, especially in the early tasks. As we can see in Figure 2, the token-based splitting methods

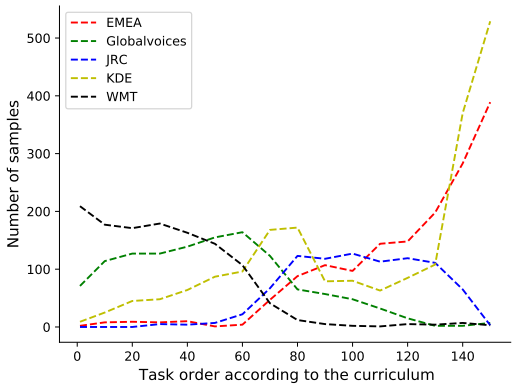

Figure 2: The statistic of samples in the task for the tokenization-based splitting strategy. More general domains are on the left and the more distinctive domains are on the right.

usually allocate more samples on domain-similar domains (*WMT*, *Globalvoices*) and allocate small samples on domain-distant domains (*EMEA*, *JRC*) in the sampling of early tasks. This can cause problems in our method since the model architecture is dynamically changing according to the number of domains (see more details in Section 3.1).

To address these issues, we sample the data uniformly from the domains to compensate for imbalanced domain distributions based on domain classifier scores.

Meta-Training. Following the balanced sampling, the process of meta-training is to update the current model parameter on $\mathcal{T}_{support}$ from θ to θ', and then evaluate on $\mathcal{T}_{query}$. The model parameter θ' is updated to minimize the meta-learning loss through MAML.

Given a pre-trained model f_θ (initialized with parameters θ trained on word-level domain mixing) and the meta-train data $\mathcal{D}_{\text{meta-train}}$, for each task $\mathcal{T}$, we learn to use one gradient update to update the model parameters from θ to θ' as follows:

$$\theta' = \theta - \alpha \nabla_\theta L_\mathcal{T}\left(f_\theta\right)$$

where α is the learning rate and L is the loss function. In our methods, we consider both the traditional sentence-level meta-learning loss $\mathcal{L}_\mathcal{T}\left(f_\theta\right)$ and the word-level loss $\Gamma_\mathcal{T}\left(f_\theta\right)$ (L^* of $\mathcal{T}$) calculated from the word-level domain mixing pre-trained model. More formally, the loss is updated as follows:

$$L_\mathcal{T}\left(f_\theta\right) = \mathcal{L}_\mathcal{T}\left(f_\theta\right) + \Gamma_\mathcal{T}\left(f_\theta\right).$$

Algorithm 1 RMLNMT (Robust Meta-Learning NMT Domain Adaptation)

Require: Domain classifier model cls; Pretrained domain-mixing model θ;

Score the sentence in $\mathcal{D}_{\text{meta-train}}$ using cls
for N epochs **do**
  Split corpus into n tasks based on step 1
  Balance sample through all tasks
  for task $\mathcal{T}_i$, $i = 1 \ldots n$ **do**
    Evaluate loss $L_{\mathcal{T}}(f_\theta)$ $= \mathcal{L}_{\mathcal{T}_i}(f_\theta) + \Gamma_{\mathcal{T}_i}(f_\theta)$ on support set
    Update the gradient with parameters $\theta' = \theta - \alpha \nabla_\theta L_{\mathcal{T}}(f_\theta)$
  end for
  Update the gradient with parameters $\theta = \theta - \beta \nabla_\theta L_{\mathcal{T}}(f_{\theta'})$ on query set
end for
return RMLNMT model parameter θ

Note that the meta-training phase is not adapted to a specific domain, so it can be used as a metric to evaluate the domain robustness of the model.

Meta-Adaptation. After the meta-training phase, the parameters are updated to adapt to each domain using the small *support set* of $\mathcal{D}_{\text{meta-test}}$ corpus to simulate the low-resource scenarios. Then performance is evaluated on the *query set* of $\mathcal{D}_{\text{meta-test}}$.

4 Experiments

Datasets. We experiment with English→German (*en2de*) and English→Chinese (*en2zh*) translation tasks. For the *en2de* task, we use the same corpora as Zhan et al. (2021). The data consists of corpora in nine domains (Bible, Books, ECB, EMEA, GlobalVoices, JRC, KDE, TED, WMT-News) publicly available on OPUS[2] (Tiedemann, 2012) and the COVID-19 corpus[3]. For *en2zh*, we use UM-Corpus (Tian et al., 2014) containing eight domains: Education, Microblog, Science, Subtitles, Laws, News, Spoken, Thesis. We use WMT14 (*en2de*) and WMT18 (*en2zh*) corpus published on the WMT website[4] as our general domain corpora. We use WMT19 English monolingual corpora to train the LM model so that we can reproduce results from previous work.

[2] opus.nlpl.eu
[3] github.com/NLP2CT/Meta-Curriculum
[4] http://www.statmt.org

Data Preprocessing. For English and German, we preprocessed all data with the Moses tokenizer[5] and use sentencepiece[6] (Kudo and Richardson, 2018) to encode the corpus with a joint vocabulary, with size 40,000. After that, we filter the sentence longer than 175 tokens and deduplicate the corpus. For Chinese, we perform word segmentation using the Stanford Segmenter (Tseng et al., 2005). To have a fair comparison with previous methods (Sharaf et al., 2020; Zhan et al., 2021), we use the same setting, which randomly sub-sampled $\mathcal{D}_{\text{meta-train}}$ and $\mathcal{D}_{\text{meta-test}}$ for each domain with fixed token sizes in order to simulate domain adaptation tasks in low-resource scenarios. More details for data used in this paper can be found in Appendix A.1.

Baselines. We compare RMLNMT with the following baselines:

- **Vanilla**. A standard Transformer-based NMT system trained on the general domains (WMT14 for *en2de*, WMT18 for *en2zh*) and $\mathcal{D}_{\text{meta-train}}$ corpus in seen-domains. We use the $\mathcal{D}_{\text{meta-train}}$ corpus because meta-learning-based methods also use the $\mathcal{D}_{\text{meta-train}}$ corpus, this is a more fair and stronger baseline.
- **Plain fine-tuning**. Fine-tune the vanilla system on support set of $\mathcal{D}_{\text{meta-test}}$ for each individual domain.
- **Tag**. prepend a domain tag to each sentence to indicate what domain it belongs to (Kobus et al., 2017).
- **Meta-MT**. Standard meta-learning approach on domain adaptation task (Sharaf et al., 2020).
- **Meta-Curriculum (LM)**. Meta-learning approach for domain adaptation using LM score as the curriculum to sample the task (Zhan et al., 2021).
- **Meta-based w/o FT**. This series of experiments uses the meta-learning system prior to adaptation to the specific domain. This can be used to evaluate the domain robustness of meta-based models (see more details in the meta-training part of Section 3.3).

[5] github.com/moses-smt/mosesdecoder
[6] github.com/google/sentencepiece

	Models	Unseen					Seen				
		Covid	Bible	Books	ECB	TED	EMEA	Globalvoices	JRC	KDE	WMT
1	**Vanilla**	24.34	**12.08**	12.61	29.96	27.89	37.27	24.19	39.84	27.75	27.38
2	**Vanilla + tag**	24.86	12.04	12.46	30.03	27.93	38.37	24.56	40.75	28.23	27.26
3	**Meta-MT w/o FT**	23.69	11.07	12.10	29.04	26.86	30.94	23.73	38.82	23.04	26.13
4	**Meta-Curriculum (LM) w/o FT**	23.70	11.16	12.24	28.22	27.21	33.49	24.27	39.21	27.60	25.83
5	**RMLNMT w/o FT**	**25.48**	11.48	**13.11**	**31.42**	**28.05**	**47.00**	**26.35**	**51.13**	**32.80**	**28.37**

Table 1: Domain Robustness: BLEU scores on the English → German translation task. *w/o* denotes the meta-learning systems without fine-tuning, FT denotes fine-tuning. Best results are highlighted in bold.

	Models	Unseen					Seen				
		Covid	Bible	Books	ECB	TED	EMEA	Globalvoices	JRC	KDE	WMT
1	**Plain FT**	24.81	12.61	12.78	30.48	28.36	37.26	24.26	40.02	27.99	27.31
2	**Plain FT + tag**	25.31	12.57	12.83	30.57	28.39	39.54	24.91	41.51	29.14	27.58
3	**Meta-MT + FT**	25.83	14.20	13.39	30.36	28.57	34.69	24.64	39.15	27.47	26.38
4	**Meta-Curriculum (LM) + FT**	**26.66**	14.37	13.70	30.41	28.97	34.00	24.72	39.61	27.37	26.68
5	**RMLNMT + FT**	26.53	**15.37**	**13.72**	**31.97**	**29.47**	**47.02**	**26.55**	**51.13**	**32.88**	**28.37**

Table 2: Domain Adaptability: BLEU scores on the English → German translation task.

Implementation. We use the Transformer model (Vaswani et al., 2017) as implemented in FairSeq[7] (Ott et al., 2019). For our word-level domain-mixing modules, we dynamically adjust the network structure according to the number of domains since every domain has its multi-head layers. Hence, the number of model parameters in the attentive sub-layers of RMLNMT is k times the number in the standard transformer (k is the number of seen domains in the training data). Following Jiang et al. (2020), we enlarged the baseline models to have $\sqrt{k}$ times larger embedding dimension, so the baseline has the same number of parameters. This should rule out that the improvements are due to increased parameter count rather than modeling improvements. For our meta-learning framework, we consider the general meta loss and word-adaptive loss together (as seen in Section 3.3). Following Zhan et al. (2021), the fine-tuning process in each models is strictly limited to 20 to simulate quick adaptation. Note that the meta-train stage only uses the seen domain corpus and the unseen domain corpus is only used in the meta-test stage. More details on hyper-parameters are listed in Appendix A.2.

Evaluation. For a fair comparison with previous work, we use the same data from the support set of $\mathcal{D}_{\text{meta-test}}$ to finetune the model and the same data from the query set of $\mathcal{D}_{\text{meta-test}}$ to evaluate the models. We measure case-sensitive detokenized BLEU with SacreBLEU[8] (Post, 2018); beam search with a beam of size five is used. Because of the recent criticism of BLEU score (Mathur et al., 2020), we also evaluate our models using chrF (Popović, 2015) and COMET[9] (Rei et al., 2020); the results are listed in Appendix A.5.

Domain Robustness. Domain robustness shows the effectiveness of the model both in seen and unseen domains. Hence, we use the model without fine-tuning to evaluate the domain robustness.

Domain Adaptability. We evaluate the domain adaptability by testing that the model quickly adapts to new domains using just hundreds of in-domain parallel sentences. Therefore, we fine-tune the models on a small amount of domain-specific data.

Cross-Domain Robustness. To better show the cross-domain robustness of RMLNMT, we use the fine-tuned model of one specific domain to generate the translation for other domains. More formally, given k domains, we use the fine-tuned model M_J with the domain label of J to generate the translation of k domains.

5 Results

Table 1 and Table 3 show the domain robustness for English→German and English→Chinese respectively. Table 2 and Table 4 show the domain adaptability on both translation task.

Domain Robustness. As seen in Table 1 and Table 3, RMLNMT shows the best domain robust-

[7] github.com/facebookresearch/fairseq
[8] github.com/mjpost/sacrebleu
[9] github.com/Unbabel/COMET

	Models	Unseen				Seen			
		Education	Microblog	Science	Subtitles	Laws	News	Spoken	Thesis
1	**Vanilla**	27.52	26.05	31.58	18.32	46.69	28.67	26.44	29.00
2	**Vanilla + tag**	27.36	26.11	31.53	18.25	47.13	28.75	26.71	29.19
3	**Meta-MT w/o FT**	28.76	26.41	32.41	17.38	43.74	27.31	25.98	28.11
4	**Meta-Curriculum (LM) w/o FT**	28.53	26.14	32.25	17.45	43.87	27.25	27.57	28.23
5	**RMLNMT w/o FT**	**30.17**	**28.42**	**34.20**	**19.89**	**57.54**	**30.39**	**28.11**	**33.20**

Table 3: Domain Robustness: BLEU scores on English → Chinese translation tasks.

	Models	Unseen				Seen			
		Education	Microblog	Science	Subtitles	Laws	News	Spoken	Thesis
1	**Plain FT**	27.05	26.31	32.09	17.77	47.64	28.28	25.73	28.47
2	**Plain FT + tag**	27.13	26.48	32.12	17.94	47.91	28.84	26.35	29.58
3	**Meta-MT + FT**	29.33	27.48	33.12	18.77	45.21	28.43	26.82	29.20
4	**Meta-Curriculum (LM) + FT**	28.91	27.20	33.19	18.93	45.46	28.17	27.84	29.47
5	**RMLNMT + FT**	**30.91**	**28.52**	**34.51**	**20.13**	**57.58**	**30.42**	**28.03**	**32.25**

Table 4: Domain Adaptability: BLEU scores on English → Chinese translation tasks.

Methods	Avg
Meta-MT	-1.97
Meta-Curriculum (LM)	-0.96
Meta-Curriculum (cls)	-0.98
RMLNMT	**2.64**

Table 5: The average improvement over vanilla baseline.

ness compared with other models both in seen and unseen domains. In addition, the traditional meta-learning approach (Meta-MT, Meta-Curriculum) without fine-tuning is even worse than the standard transformer model in seen domains. This phenomenon is our motivation for improving the robustness of traditional meta-learning based approach. In other words, we cannot be sure whether the improvement of the meta-based method is due to the domain adaptability of meta-learning or the robustness of the teacher model. Note this setup differs from the previous work (Sharaf et al., 2020; Zhan et al., 2021) because we included the $\mathcal{D}_{\text{meta-train}}$ data to the vanilla system to insure all systems in the table use the same training data.[10] Interestingly, the translation quality in the *WMT* domain is also improved which is different than (Zhan et al., 2021). They explain that their methods achieve maximum robustness on the *WMT* domain, while our results demonstrate that our model can further improve robustness even when trained on the same domain as the pre-trained model.

[10]We also confirmed with Zhan et al. (2021) via email that they did not deduplicate the corpus, which is another reason for the difference between our results and their results.

Domain Adaptability. From Tables 2 and 4, we observe that the traditional meta-learning approach shows high adaptability to unseen domains but fails on seen domains due to limited domain robustness. In contrast, RMLNMT shows its domain adaptability both in seen and unseen domains, and maintains the domain robustness simultaneously. Compared with RMLNMT, the traditional meta-learning approach show more improvement between the *w/o FT* model and *FT* model. For example, *Meta-MT* and *Meta-Curriculum (LM)* obtains 1.32 and 2.19 BLEU score improvement after finetuning on the *ECB* domain; improvement from *RMLNMT* only got 0.55. This phenomenon meets our expectations since *RMLNMT* without finetuning is already strong enough due to the domain robustness of word-level domain mixing. In other words, the improvement of the traditional meta-learning approach is to some extent due to the unrobustness of the model.

Cross-Domain Robustness. Table 5 reports the average difference of $k \times k$ BLEU scores; a larger positive value means a more robust model. We observed that the plain meta-learning based methods have a negative value, which means the performance gains in the specific domains come at the cost of performance decreases in other domains. In other words, the model is not domain robust enough. In contrast, RMLNMT has a positive difference with the vanilla system, showing that the model is robust. The specific BLEU scores are shown in Figure 3 of Appendix A.4.

The results of both domain robustness and do-

Classifier	Unseen					Seen				
	Covid	Bible	Books	ECB	TED	EMEA	Globalvoices	JRC	KDE	WMT
CNN	24.12	13.57	12.74	30.31	28.14	46.12	25.17	50.52	31.15	26.34
BERT-many-labels	25.89	14.77	13.71	**32.10**	29.28	**47.41**	**26.70**	51.34	32.76	28.17
BERT-2-labels	26.10	14.85	13.58	31.99	29.17	46.80	26.46	**51.56**	32.83	28.37
mBERT-many-labels	26.10	14.73	13.69	31.93	29.11	47.02	26.33	51.13	32.69	27.91
mBERT-2-labels	**26.53**	**15.37**	**13.71**	31.97	**29.47**	47.02	26.55	51.13	**32.88**	**28.37**

Table 6: Different classifier: BLEU scores on the English → German translation task.

Sampling Strategy	Unseen					Seen				
	Covid	Bible	Books	ECB	TED	EMEA	Globalvoices	JRC	KDE	WMT
Token-based sampling	25.30	11.38	12.70	31.61	28.01	47.51	26.50	**51.31**	32.88	28.03
Balance sampling	**25.47**	**11.51**	**12.79**	**32.08**	**28.98**	**47.64**	**26.58**	51.25	**32.91**	**28.07**

Table 7: Different sampling strategy: BLEU scores on the English → German translation task.

Finetune Strategy	Unseen					Seen				
	Covid	Bible	Books	ECB	TED	EMEA	Globalvoices	JRC	KDE	WMT
FT-unseen	25.23	13.18	12.73	**32.45**	28.41	46.35	25.83	50.85	32.30	26.88
FT-seen	24.58	11.73	12.57	30.79	27.29	46.58	25.73	50.91	31.78	26.51
FT-all	15.00	7.77	9.06	21.33	16.98	24.69	14.63	27.59	12.77	15.75
FT-specific	**26.53**	**15.37**	**13.71**	31.97	**29.47**	**47.02**	**26.33**	**51.13**	**32.83**	**28.37**

Table 8: Different fine-tuning strategy: BLEU scores on the English → German translation task.

main adaptability are consistent for the chrF and COMET evaluation metrics (see more details in Tables 13 and 14 of Appendix A.5).

6 Analysis

In this section, we conduct additional experiments to better understand the strengths of RMLNMT. We analyze the contribution of different components in RMLNMT, through an ablation study.

Different classifiers. We evaluate the impact of different classifiers on translation performance. The main results are as shown in Table 6 (see more details in Appendix A.3). We observed that the performance of RMLNMT is not directly proportional to the accuracy of the classifier. In other words, slightly higher classification accuracy does not lead to better BLEU scores. This is because the accuracy of the classifier is close between BERT-based models and the primary role of the classifier is to construct the curriculum for splitting the tasks. When we use a significantly worse classifier, i.e., the CNN in our experiments, the overall performance of RMLNMT is worse than the BERT-based classifier.

Balanced sampling vs. Token-based sampling. Plain meta-learning uses a token-based sampling strategy to split sentences into small tasks. However, the token-based strategy could cause unbalanced domain distribution in some tasks, especially in the early stage of training due to domain mismatches (see the discussion of balanced sampling in Section 3.3). To address this issue, we proposed to balance the domain distribution after splitting the task. Table 7 shows that our methods can result in small improvements in performance. For example, in the *TED* domain, BLEU was 28.01 with token-based sampling, but with the balanced sampling strategy BLEU was 28.98. We keep the same number of tasks to have a fair comparison with previous methods.

Different fine-tuning strategies. As described in Section 3.1, the model for each domain has its own multi-head and feed-forward layers. During the fine-tuning stage of RMLNMT, we devise four strategies: i) **FT-unseen**: fine-tuning using all unseen domain corpora; ii) **FT-seen**: fine-tuning using all seen domain corpora; iii) **FT-all**: fine-tuning using all out-of-domain corpora (seen and unseen domains); iv) **FT-specific**: using the specific domain corpus to fine-tune the specific models. The results are shown in Table 8. *FT-specific* obtains robust results among all the strategies. Although other strategies outperform *FT-specific* in some domains, *FT-specific* is robust across all domains. Furthermore, *FT-specific* is the fairest comparison

because it uses only a specific domain corpus to fine-tune, which is the same as the baseline systems.

7 Related Work

Domain Adaptation for NMT. Current approaches can be categorized into two groups by granularity: From a sentence-level perspective, researchers either use data selection methods (Moore and Lewis, 2010; Axelrod et al., 2011) to select the training data that is similar to out-of-domain parallel corpora or train a classifier (Rieß et al., 2021) or utilize a language model (Wang et al., 2017; Zhan et al., 2021) to better weight the sentences. From a word-level perspective, researchers try to model domain distribution at the word level, since a word in a sentence can be related to more domains than just the sentence domain (Zeng et al., 2018; Yan et al., 2018; Hu et al., 2019; Sato et al., 2020; Jiang et al., 2020).

Curriculum Learning for NMT. Curriculum learning (Bengio et al., 2009) starts with easier tasks and then progressively gain experience to process more complex tasks, which has proved to be useful in NMT domain adaptation. Stojanovski and Fraser (2019) utilize curriculum learning to improve anaphora resolution in NMT systems. Zhang et al. (2019) and Zhan et al. (2021) use a language model to compute a similarity score between domains, from which a curriculum is devised for adapting NMT systems to specific domains from general domains.

Meta-Learning for NMT. Gu et al. (2018) apply model-agnostic meta-learning (MAML; Finn et al., 2017) to NMT. They show that MAML effectively improves low-resource NMT. Li et al. (2020), Sharaf et al. (2020) and Zhan et al. (2021) propose to formulate the problem of low-resource domain adaptation in NMT as a meta-learning problem: the model learns to quickly adapt to an unseen new domain from a general domain.

8 Conclusion

We presented `RMLNMT`, a robust meta-learning framework for low-resource NMT domain adaptation reaching both high domain adaptability and domain robustness (both in the seen domains and unseen domains). We found that domain robustness dominates the results compared to domain adaptability in meta-learning based approaches. The results show that RMLNMT works best in setups that require high robustness in low-resource scenarios.

Acknowledgement

We thank Mengjie Zhao for the helpful comments. This work was supported by funding to Wen Lai's PhD research from LMU-CSC (China Scholarship Council) Scholarship Program. This work has received funding from the European Research Council under the European Union's Horizon 2020 research and innovation program (grant agreement #640550). This work was also supported by the DFG (grant FR 2829/4-1). The work at CUNI was supported by the European Commission via its Horizon 2020 research and innovation programme (870930).

References

Amittai Axelrod, Xiaodong He, and Jianfeng Gao. 2011. Domain adaptation via pseudo in-domain data selection. In *Proceedings of the 2011 Conference on Empirical Methods in Natural Language Processing*, pages 355–362, Edinburgh, Scotland, UK. Association for Computational Linguistics.

Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. 2015. Neural machine translation by jointly learning to align and translate. In *3rd International Conference on Learning Representations*.

Trapit Bansal, Rishikesh Jha, Tsendsuren Munkhdalai, and Andrew McCallum. 2020. Self-supervised meta-learning for few-shot natural language classification tasks. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 522–534, Online. Association for Computational Linguistics.

Yoshua Bengio, Jérôme Louradour, Ronan Collobert, and Jason Weston. 2009. Curriculum learning. In *Proceedings of the 26th annual international conference on machine learning*, pages 41–48.

Praveen Dakwale and Christof Monz. 2017. Finetuning for neural machine translation with limited degradation across in-and out-of-domain data. In *Proceedings of the XVI Machine Translation Summit*.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, Minneapolis, Minnesota. Association for Computational Linguistics.

Zi-Yi Dou, Keyi Yu, and Antonios Anastasopoulos. 2019. Investigating meta-learning algorithms for low-resource natural language understanding tasks. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 1192–1197, Hong Kong, China. Association for Computational Linguistics.

Yingjun Du, Nithin Holla, Xiantong Zhen, Cees Snoek, and Ekaterina Shutova. 2021. Meta-learning with variational semantic memory for word sense disambiguation. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 5254–5268, Online. Association for Computational Linguistics.

Chelsea Finn, Pieter Abbeel, and Sergey Levine. 2017. Model-agnostic meta-learning for fast adaptation of deep networks. In *International Conference on Machine Learning*, pages 1126–1135. PMLR.

Markus Freitag and Yaser Al-Onaizan. 2016. Fast domain adaptation for neural machine translation. *arXiv preprint arXiv:1612.06897*.

Robert M. French. 1999. Catastrophic forgetting in connectionist networks. *Trends in Cognitive Sciences*, 3(4):128–135.

Jiatao Gu, Yong Wang, Yun Chen, Victor O. K. Li, and Kyunghyun Cho. 2018. Meta-learning for low-resource neural machine translation. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 3622–3631, Brussels, Belgium. Association for Computational Linguistics.

Timothy M Hospedales, Antreas Antoniou, Paul Micaelli, and Amos J Storkey. 2021. Meta-learning in neural networks: A survey. *IEEE Transactions on Pattern Analysis and Machine Intelligence*.

Junjie Hu, Mengzhou Xia, Graham Neubig, and Jaime Carbonell. 2019. Domain adaptation of neural machine translation by lexicon induction. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 2989–3001, Florence, Italy. Association for Computational Linguistics.

Haoming Jiang, Chen Liang, Chong Wang, and Tuo Zhao. 2020. Multi-domain neural machine translation with word-level adaptive layer-wise domain mixing. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 1823–1834, Online. Association for Computational Linguistics.

Yoon Kim. 2014. Convolutional neural networks for sentence classification. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1746–1751, Doha, Qatar. Association for Computational Linguistics.

Catherine Kobus, Josep Crego, and Jean Senellart. 2017. Domain control for neural machine translation. In *Proceedings of the International Conference Recent Advances in Natural Language Processing, RANLP 2017*, pages 372–378, Varna, Bulgaria. INCOMA Ltd.

Taku Kudo. 2018. Subword regularization: Improving neural network translation models with multiple subword candidates. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 66–75, Melbourne, Australia. Association for Computational Linguistics.

Taku Kudo and John Richardson. 2018. SentencePiece: A simple and language independent subword tokenizer and detokenizer for neural text processing. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 66–71, Brussels, Belgium. Association for Computational Linguistics.

Rumeng Li, Xun Wang, and Hong Yu. 2020. Metamt, a meta learning method leveraging multiple domain data for low resource machine translation. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pages 8245–8252.

Nitika Mathur, Timothy Baldwin, and Trevor Cohn. 2020. Tangled up in BLEU: Reevaluating the evaluation of automatic machine translation evaluation metrics. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 4984–4997, Online. Association for Computational Linguistics.

Robert C. Moore and William Lewis. 2010. Intelligent selection of language model training data. In *Proceedings of the ACL 2010 Conference Short Papers*, pages 220–224, Uppsala, Sweden. Association for Computational Linguistics.

Mathias Müller, Annette Rios Gonzales, and Rico Sennrich. 2020. Domain robustness in neural machine translation. In *Proceedings of the 14th Conference of the Association for Machine Translation in the Americas (Volume 1: Research Track)*, pages 151–164.

Myle Ott, Sergey Edunov, Alexei Baevski, Angela Fan, Sam Gross, Nathan Ng, David Grangier, and Michael Auli. 2019. fairseq: A fast, extensible toolkit for sequence modeling. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics (Demonstrations)*, pages 48–53, Minneapolis, Minnesota. Association for Computational Linguistics.

Nicolas Papernot, Patrick McDaniel, Xi Wu, Somesh Jha, and Ananthram Swami. 2016. Distillation as a

defense to adversarial perturbations against deep neural networks. In *2016 IEEE symposium on security and privacy (SP)*, pages 582–597. IEEE.

Maja Popović. 2015. chrF: character n-gram F-score for automatic MT evaluation. In *Proceedings of the Tenth Workshop on Statistical Machine Translation*, pages 392–395, Lisbon, Portugal. Association for Computational Linguistics.

Matt Post. 2018. A call for clarity in reporting BLEU scores. In *Proceedings of the Third Conference on Machine Translation: Research Papers*, pages 186–191, Brussels, Belgium. Association for Computational Linguistics.

Kun Qian and Zhou Yu. 2019. Domain adaptive dialog generation via meta learning. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 2639–2649, Florence, Italy. Association for Computational Linguistics.

Ricardo Rei, Craig Stewart, Ana C Farinha, and Alon Lavie. 2020. COMET: A neural framework for MT evaluation. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 2685–2702, Online. Association for Computational Linguistics.

Simon Rieß, Matthias Huck, and Alex Fraser. 2021. A comparison of sentence-weighting techniques for NMT. In *Proceedings of the 18th Biennial Machine Translation Summit (Volume 1: Research Track)*, pages 176–187, Virtual. Association for Machine Translation in the Americas.

Shoetsu Sato, Jin Sakuma, Naoki Yoshinaga, Masashi Toyoda, and Masaru Kitsuregawa. 2020. Vocabulary adaptation for domain adaptation in neural machine translation. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 4269–4279, Online. Association for Computational Linguistics.

Amr Sharaf, Hany Hassan, and Hal Daumé III. 2020. Meta-learning for few-shot NMT adaptation. In *Proceedings of the Fourth Workshop on Neural Generation and Translation*, pages 43–53, Online. Association for Computational Linguistics.

Dario Stojanovski and Alexander Fraser. 2019. Improving anaphora resolution in neural machine translation using curriculum learning. In *Proceedings of Machine Translation Summit XVII: Research Track*, pages 140–150, Dublin, Ireland. European Association for Machine Translation.

Brian Thompson, Jeremy Gwinnup, Huda Khayrallah, Kevin Duh, and Philipp Koehn. 2019. Overcoming catastrophic forgetting during domain adaptation of neural machine translation. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 2062–2068, Minneapolis, Minnesota. Association for Computational Linguistics.

Liang Tian, Derek F. Wong, Lidia S. Chao, Paulo Quaresma, Francisco Oliveira, Yi Lu, Shuo Li, Yiming Wang, and Longyue Wang. 2014. UM-corpus: A large English-Chinese parallel corpus for statistical machine translation. In *Proceedings of the Ninth International Conference on Language Resources and Evaluation (LREC'14)*, pages 1837–1842, Reykjavik, Iceland. European Language Resources Association (ELRA).

Jörg Tiedemann. 2012. Parallel data, tools and interfaces in OPUS. In *Proceedings of the Eighth International Conference on Language Resources and Evaluation (LREC'12)*, pages 2214–2218, Istanbul, Turkey. European Language Resources Association (ELRA).

Huihsin Tseng, Pichuan Chang, Galen Andrew, Daniel Jurafsky, and Christopher Manning. 2005. A conditional random field word segmenter for sighan bakeoff 2005. In *Proceedings of the Fourth SIGHAN Workshop on Chinese Language Processing*.

Zhaopeng Tu, Yang Liu, Lifeng Shang, Xiaohua Liu, and Hang Li. 2017. Neural machine translation with reconstruction. In *Thirty-First AAAI Conference on Artificial Intelligence*.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in neural information processing systems*, pages 5998–6008.

Lingxiao Wang, Kevin Huang, Tengyu Ma, Quanquan Gu, and Jing Huang. 2021. Variance-reduced first-order meta-learning for natural language processing tasks. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 2609–2615, Online. Association for Computational Linguistics.

Rui Wang, Masao Utiyama, Lemao Liu, Kehai Chen, and Eiichiro Sumita. 2017. Instance weighting for neural machine translation domain adaptation. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 1482–1488, Copenhagen, Denmark. Association for Computational Linguistics.

Shen Yan, Leonard Dahlmann, Pavel Petrushkov, Sanjika Hewavitharana, and Shahram Khadivi. 2018. Word-based domain adaptation for neural machine translation. In *Proceedings of the 15th International Conference on Spoken Language Translation*, pages 31–38, Brussels. International Conference on Spoken Language Translation.

Kyra Yee, Yann Dauphin, and Michael Auli. 2019. Simple and effective noisy channel modeling for neural machine translation. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing*

(EMNLP-IJCNLP), pages 5696–5701, Hong Kong, China. Association for Computational Linguistics.

Wenpeng Yin. 2020. Meta-learning for few-shot natural language processing: A survey. *arXiv preprint arXiv:2007.09604*.

Changlong Yu, Jialong Han, Haisong Zhang, and Wilfred Ng. 2020. Hypernymy detection for low-resource languages via meta learning. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 3651–3656, Online. Association for Computational Linguistics.

Jiali Zeng, Jinsong Su, Huating Wen, Yang Liu, Jun Xie, Yongjing Yin, and Jianqiang Zhao. 2018. Multi-domain neural machine translation with word-level domain context discrimination. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 447–457, Brussels, Belgium. Association for Computational Linguistics.

Runzhe Zhan, Xuebo Liu, Derek F. Wong, and Lidia S. Chao. 2021. Meta-curriculum learning for domain adaptation in neural machine translation. In *Proceedings of the AAAI Conference on Artificial Intelligence*, pages 14310–14318.

Xuan Zhang, Pamela Shapiro, Gaurav Kumar, Paul McNamee, Marine Carpuat, and Kevin Duh. 2019. Curriculum learning for domain adaptation in neural machine translation. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 1903–1915, Minneapolis, Minnesota. Association for Computational Linguistics.

A Appendix

A.1 Datasets

For the OPUS corpus used in the English → German translation task, we deduplicated the corpus, which is different from (Zhan et al., 2021) and is the main reason that we cannot reproduce the results in the original paper. The statistics of the original OPUS are shown in Table 9. The seen domains (EMEA, Globalvoices, JRC, KDE, WMT) contain a lot of duplicated sentences. The scores in the original paper are too high because the $\mathcal{D}_{\text{meta-train}}$ dataset overlaps with some sentences in $\mathcal{D}_{\text{meta-test}}$.

For the meta-learning phase, to have a fair comparison with previous methods, we use the same setting. We random split 160 tasks and 10 tasks respectively in $\mathcal{D}_{\text{meta-train}}$ and $\mathcal{D}_{\text{meta-test}}$ to simulate the low-resource scenarios. For each task, the token amount of support set and query set is a strict limit to $8K$ and $16K$. $\mathcal{D}_{\text{meta-dev}}$ corpus is limited to 5000 sentences for each domain. Table 10 and Table 11 shows the detailed statistics of the English → German and English → Chinese tasks.

Corpus	Original	Deduplicated
Covid	3,325	3,312
Bible	62,195	61,585
Books	51,467	51,106
ECB	113,116	113,081
TED	143,830	142,756
EMEA	1,103,807	360,833
Globalvoices	71,493	70,519
JRC	717,988	503,789
KDE	223,672	187,918
WMT	45,913	34,727

Table 9: Data statistic (sentences) of the original corpus for English→German translation task

	$\mathcal{D}_{\text{meta-train}}$		$\mathcal{D}_{\text{meta-test}}$	
	Support	**Query**	**Support**	**Query**
Covid	/	/	309	612
Bible	/	/	280	548
Books	/	/	304	637
ECB	/	/	295	573
TED	/	/	390	772
EMEA	14856	29668	456	975
Globalvoices	11686	23319	368	699
JRC	7863	15769	254	519
KDE	24078	48284	756	1510
WMT	10939	21874	334	704

Table 10: Data statistic (sentences) of the meta-learning stage for English→German translation task

A.2 Model Configuration

We use the `Transformer Base` architecture (Vaswani et al., 2017) as implemented in fairseq (Ott et al., 2019). We use the standard Transformer architecture with dimension 512, feed-forward layer 2048, 8 attention heads, 6 encoder layers and 6 decoder layers. For optimization, we use the Adam optimizer with a learning rate of $5 \cdot 10^{-5}$. To prevent overfitting, we applied a dropout of 0.3 on all layers. The number of warm-up steps was set to 4000. At the time of inference, a beam search of size 5 is used to balance the decoding time and accuracy of the search.

For the word-level domain-mixing model, we use the same setting as Jiang et al. (2020). The number of parameters of our model is dynamically adjusted with the domain numbers and k times higher than standard model architecture, since every domain has its multi-head attention layer and feed-forward layer. To have a fair comparison between baselines, we enlarged the baseline models to have $\sqrt{k}$ times larger embedding dimension, so

	$\mathcal{D}_{\text{meta-train}}$		$\mathcal{D}_{\text{meta-test}}$	
	Support	**Query**	**Support**	**Query**
Education	/	/	395	785
Microblog	/	/	358	721
Science	/	/	392	852
Subtitles	/	/	612	1219
Laws	6379	13001	197	416
News	9004	18362	281	536
Spoken	18270	36569	571	1148
Thesis	8914	17883	298	547

Table 11: Data statistic (sentences) of the meta-learning stage for English→Chinese translation task

Classifier	**Acc(%)**
CNN	74.91%
BERT: many-labels	96.12%
BERT: 2-labels	95.35%
mBERT: many-labels	95.41%
mBERT: 2-labels	95.26%

Table 12: The accuracy of the different classifiers.

the baseline has the same number of parameters.

A.3 Different classifiers

With a general in-domain corpus and some out-of-domain corpora, we train five classifiers. We experiment with two different labeling schemes: `2-labels` where we distinguish only two classes: *out-of-domain* and *in-domain*; `many-labels` where sentences are labeled with the respective domain labels. Further, we experiment with two variants of the BERT model: first, we use monolingual English BERT on the source side only, and second, we use multilingual BERT (mBERT) to classify the parallel sentence pairs. For further comparison, we include also a CNN-based classifier (Kim, 2014). We present the accuracy of the English-German domain classifier in Table 12.

A.4 Cross-Domain Robustness

In Figure 3 we show the detailed results ($k \times k$ scores) of cross-domain robustness.

A.5 Evaluations

In addition to BLEU, we also use chrF (Popović, 2015) and COMET (Rei et al., 2020) as evaluation metrics. Table 13 and Table 14 show the results. Consistently with the BLEU score (Tables 1 and Table 2), we observed that RMLNMT is more effective than all previous methods.

	Models	Unseen					Seen				
		Covid	Bible	Books	ECB	TED	EMEA	Globalvoices	JRC	KDE	WMT
1	**Vanilla**	0.550	0.418	0.385	0.538	0.542	0.599	0.536	0.614	0.525	0.558
	Plain FT	0.555	0.423	0.388	0.540	0.548	0.600	0.536	0.618	0.528	0.558
2	**Vanilla + tag**	0.555	0.418	0.384	0.540	0.544	0.657	0.545	0.627	0.531	0.558
	Plain FT + tag	0.562	0.423	0.388	0.540	0.549	0.602	0.536	0.694	0.547	0.561
3	**Meta-MT w/o FT**	0.545	0.410	0.382	0.498	0.538	0.532	0.531	0.610	0.464	0.553
	Meta-MT + FT	0.566	0.432	0.390	0.542	0.556	0.582	0.538	0.613	0.522	0.552
4	**Meta-Curriculum (LM) w/o FT**	0.548	0.412	0.384	0.523	0.543	0.560	0.536	0.611	0.521	0.554
	Meta-Curriculum (LM) + FT	**0.567**	0.434	0.395	0.544	0.548	0.572	0.539	0.615	0.522	0.553
5	**RMLNMT w/o FT**	0.555	0.405	0.388	0.557	0.544	0.656	0.552	0.702	0.574	0.561
	RMLNMT + FT	0.562	**0.451**	**0.395**	**0.558**	**0.560**	**0.656**	**0.552**	**0.702**	**0.574**	**0.561**

Table 13: chrF scores on the English → German translation task.

	Models	Unseen					Seen				
		Covid	Bible	Books	ECB	TED	EMEA	Globalvoices	JRC	KDE	WMT
1	**Vanilla**	0.4967	-0.1250	-0.2225	0.3276	0.3400	0.3096	0.3199	0.5430	0.1836	0.4326
	Plain FT	0.5066	-0.1105	-0.1985	0.3315	0.3553	0.3177	0.3276	0.5492	0.1813	0.4392
2	**Vanilla + tag**	0.4970	-0.1250	-0.2228	0.3277	0.3401	0.3176	0.3291	0.5495	0.1846	0.4311
	Plain FT + tag	0.5078	-0.1105	-0.1981	0.3315	0.3553	0.3179	0.3341	0.5572	0.1973	0.4398
3	**Meta-MT w/o FT**	0.4850	-0.1454	-0.2228	0.0953	0.3506	0.0524	0.2985	0.5319	0.1304	0.4137
	Meta-MT + FT	0.5175	-0.0650	-0.1878	0.3466	0.3824	0.2678	0.3189	0.5509	0.1316	0.4161
4	**Meta-Curriculum (LM) w/o FT**	0.4879	-0.1365	-0.2122	0.2568	0.3751	0.1968	0.3273	0.5246	0.0962	0.4206
	Meta-Curriculum (LM) + FT	**0.5347**	-0.0604	-0.1773	0.3460	0.3729	0.2366	0.3141	0.5430	0.1467	0.4128
5	**RMLNMT w/o FT**	0.4943	-0.1956	-0.2179	0.3580	0.3394	0.4026	0.3769	0.6797	0.3014	0.4255
	RMLNMT + FT	0.5302	**-0.0543**	**-0.1610**	**0.3547**	**0.3867**	**0.4046**	**0.3771**	**0.6797**	**0.3015**	**0.4256**

Table 14: COMET scores on the English → German translation task.

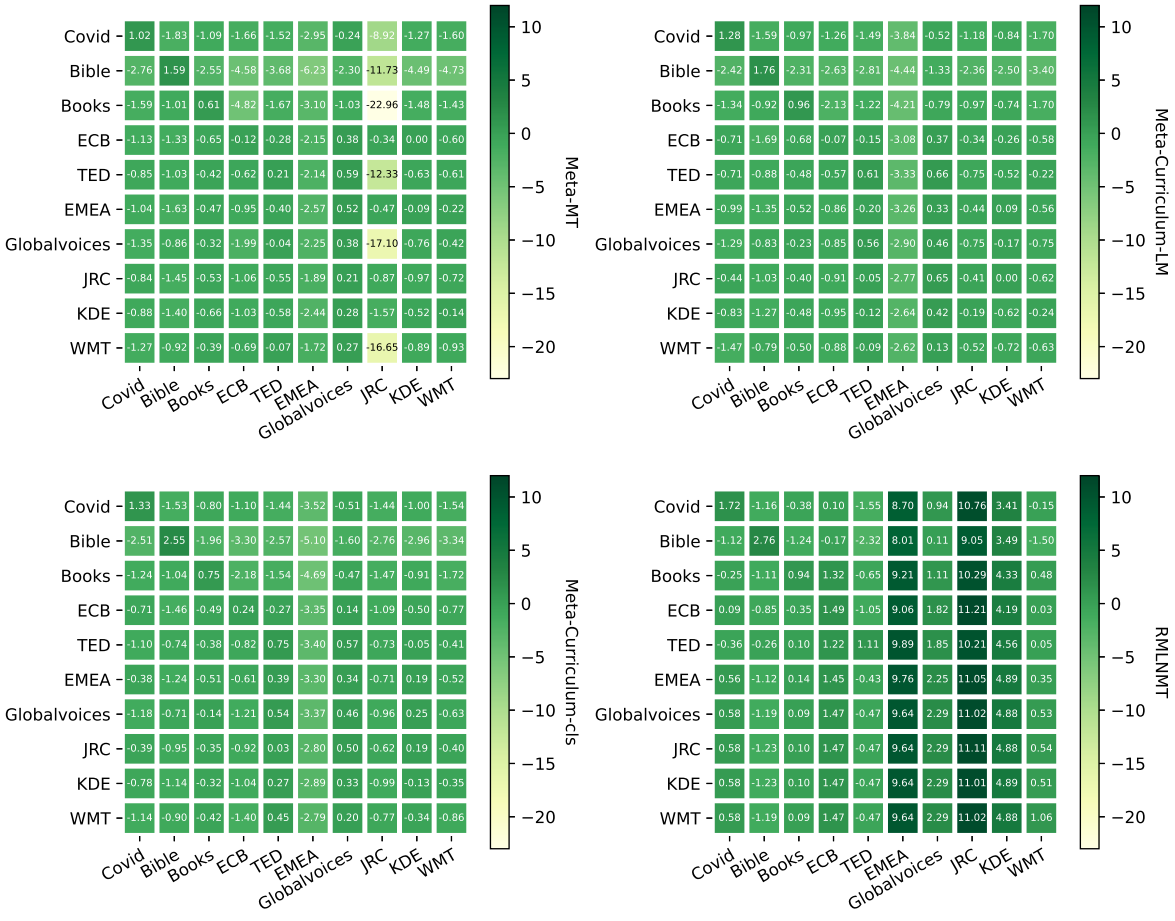

Figure 3: BLEU scores for one specific finetuned model on other domains for en2de translation.

CoDoNMT: Modeling Cohesion Devices for Document-Level Neural Machine Translation

Yikun Lei, Yuqi Ren, Deyi Xiong *
College of Intelligence and Computing, Tianjin University, Tianjin, China
{yikunlei,ryq20,dyxiong}@tju.edu.cn

Abstract

Cohesion devices, e.g., reiteration, coreference, are crucial for building cohesion links across sentences. In this paper, we propose a document-level neural machine translation framework, CoDoNMT, which models cohesion devices from two perspectives: Cohesion Device Masking (CoDM) and Cohesion Attention Focusing (CoAF). In CoDM, we mask cohesion devices in the current sentence and force NMT to predict them with inter-sentential context information. A prediction task is also introduced to be jointly trained with NMT. In CoAF, we attempt to guide the model to pay exclusive attention to relevant cohesion devices in the context when translating cohesion devices in the current sentence. Such a cohesion attention focusing strategy is softly applied to the self-attention layer. Experiments on three benchmark datasets demonstrate that our approach outperforms state-of-the-art document-level neural machine translation baselines. Further linguistic evaluation validates the effectiveness of the proposed model in producing cohesive translations.

1 Introduction

Neural Machine Translation (NMT) has become the dominant approach for machine translation and achieved substantial progress in comparison to statistical machine translation. Some studies even claim that NMT has reached human parity (Hassan et al., 2018). Despite this, most NMT models are at the sentence level, which translate documents sentence by sentence, ignoring inter-sentential dependencies. Documents translated in this way are usually incoherent and inconsistent across sentences.

In order to address this issue, a wide range of efforts have been made to leverage inter-sentential context information for document-level NMT (Tiedemann and Scherrer, 2017; Zhang et al., 2018; Voita et al., 2019; Tan et al., 2019; Maruf et al., 2019; Xu et al., 2020b; Zhang et al., 2021). Most efforts have been dedicated to modeling local or global context via additional encoders, attention, cache, concatenating inputs, etc (Maruf et al., 2021). However, these approaches normally focus on the way of integrating context into translation, rather than the context itself. The basic assumption behind this is that models are able to detect relevant contextual information. However, Kim et al. (2019) and Li et al. (2020) find that most of the improvements obtained by these approaches cannot be explained as leveraging the right context. We suggest that it could be not a good choice to treat contextual words equally and rely on models to learn contextual clues in an implicit way. This is because it is usually difficult for NMT to capture key information from a long context through itself (Yin et al., 2021). Hence, pinpointing semantically or grammatically relevant context words in an explicit way is desirable for document-level NMT.

In this paper, different from the aforementioned context modeling schemes, we model contextual information for document-level NMT in an explicit way via cohesion devices. Cohesion devices (e.g., reiteration, co-reference) are widely acknowledged as important linguistic items that chain sentences into cohesive discourse (Halliday and Hasan, 1976). Moreover, the interpretation of one cohesion device depends on the corresponding device that is paired to it (Halliday and Hasan, 1976). Consider the following text "*Amy went to the party. She sat with Sara.*". The interpretation of the cohesion device *she* is deeply related to the cohesion device *Amy*. Therefore, explicitly modeling these cohesion devices may guide document-level NMT to actively explore contextual clues, so as to yield cohesive translations.

Inspired by this, we propose **CoDoNMT**, as shown in Fgiure 1, to explore **Co**hesion devices for **Do**cument-level **N**eural **M**achine **T**ranslation.

*corresponding author

Proceedings of the 29th International Conference on Computational Linguistics, pages 5205–5216
October 12–17, 2022.

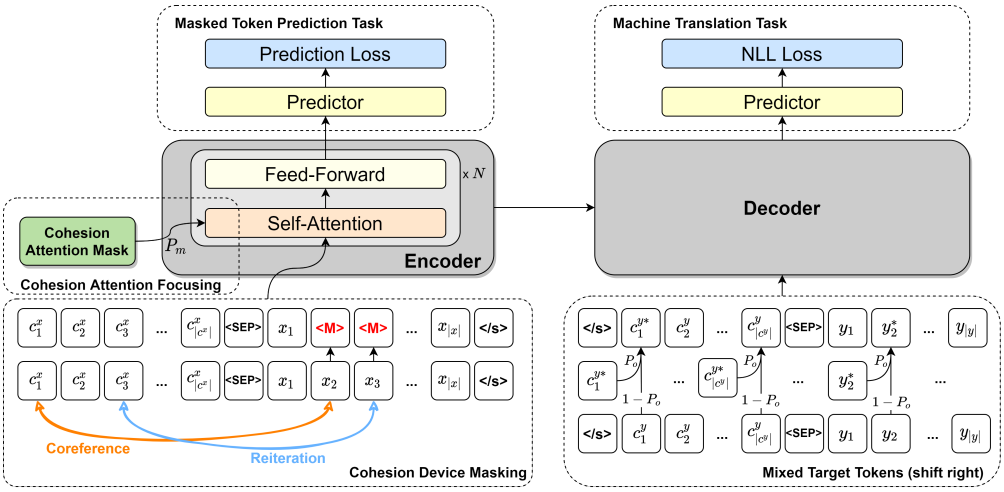

Figure 1: The diagram of CoDoNMT. We prepend the preceding three sentences to each current sentence as its context on both the source and target side. CoDM is applied on the source side of the current sentence while CoAF is softly used in each self-attention layer of the encoder. A masked token prediction task corresponding to CoDM is employed as an auxiliary task to the primary translation task, where the concatenated input (source sentence + context) is translated into the target language. Exposure bias mitigation is applied during training: $c_{\cdot}^{y*}$ and $y_{\cdot}^{*}$ indicate the predicted words in translations of the context and the current sentence, respectively.

Particularly, we present Cohesion Device Masking (CoDM) and Cohesion Attention Focusing (CoAF) as two essential components for CoDoNMT, in an attempt to force NMT to explicitly predict masked cohesion devices and to focus its attention exclusively on related cohesion devices.

Cohesion Device Masking We concatenate the previous context to the current sentence on both the source and target side. Cohesion devices of the current sentence on the source side are masked. In doing so, we force NMT to actively explore previous context to predict the masked cohesion devices, which may teach document-level NMT to pinpoint relevant linguistic context for translation.

Cohesion Attention Focusing As mentioned above, in order to correctly interpret and translate cohesion devices in the current sentence, we need to capture their paired cohesion devices in the concatenated context. For this, we force NMT to pay exclusive attention to previous cohesion clues in the context with attention masks when translating cohesion devices in the current sentence. In this way, we narrow the range of context and enable the model to evade irrelevant contextual information.

In a nutshell, our contributions are three-fold.

- We propose CoDoNMT for document-level NMT, which explicitly explores cohesion devices to capture context information.
- We introduce CoDM and CoAF in CoDoNMT to force NMT to predict masked cohesion devices with context information and to attend to only cohesion devices in the context for translating cohesion devices in the current sentence, respectively.
- We conduct experiments on three widely-used datasets and a linguistic contrastive test set. Results of both automatic and linguistic evaluation demonstrate that our methods are able to significantly improve translation quality over previous state-of-the-art document-level NMT models.

2 Cohesion Devices

A discourse is cohesive when sentences are properly linked by cohesion devices. From the linguistic perspective, cohesion devices can be divided into two categories: lexical cohesion devices and grammatical cohesion devices (Halliday and Hasan, 1976). In this paper, we consider reiteration, synonym and super-subordinate for lexical cohesion devices, and co-reference for grammatical cohesion devices. We choose these devices because they are common and can be annotated automatically.

Reiteration: Reiteration refers to the repetition of the same words in a discourse. This is a common phenomenon in discourse (Church, 2000). And it is easy to detect. Note that we exclude stop words when detecting any type of cohesion devices.

Synonym: We use WordNet (Fellbaum, 2000) to define synonyms, which is a large lexical database

of English. Nouns, verbs, adjectives and adverbs are grouped into sets of semantic groups called *synsets*. We denote $synset(w)$ as a set that includes synonyms grouped in the same synset as word w in WordNet.

Super-subordinate: Superordinate and subordinate are formed by words with an is-a semantic relationship, such as *apple* and *fruit* (hypernym), *furniture* and *cupboard* (hyponym), and so on. As the super-subordinate relation is also encoded in WordNet, we still use WordNet to detect hypernyms and hyponyms. Let $hypset(w)$ be a set that includes both hypernyms and hyponyms in WordNet for word w.

Co-reference: Co-reference is a relationship between two words or phrases in which both refer to the same person or thing and one is a linguistic antecedent of the other. We use CoreNLP (Manning et al., 2014) to parse co-reference relations between the current sentence and its context sentences.

3 CoDoNMT

Figure 1 illustrates the diagram of the proposed CoDoNMT, which uses the standard Transformer (Vaswani et al., 2017) as its backbone. We prepend the previous three sentences to the current sentence on both the source and target side, separated by a special token (i.e., <SEP>). We apply CoDM to the source-side input and CoAF to the self-attention layer of the encoder.

3.1 Cohesion Device Masking

The key of CoDM is to mask cohesion devices in the current sentence and force the model to predict those masked tokens using inter-sentential context. Predicting cohesion devices might offer the model the ability to establish cohesion links, so as to make translation cohesive.

Obtaining Cohesion Devices We denote $\boldsymbol{x}$ as the source side of the current sentence and $\boldsymbol{c^x}$ as the preceding context of $\boldsymbol{x}$. $|\boldsymbol{x}|$ and $|\boldsymbol{c^x}|$ denote the length of $\boldsymbol{x}$ and $\boldsymbol{c^x}$, respectively. Correspondingly, $\boldsymbol{y}$ indicates the target side of the current sentence and its preceding context is $\boldsymbol{c^y}$. $|\boldsymbol{y}|$ and $|\boldsymbol{c^y}|$ are the length of $\boldsymbol{y}$ and $\boldsymbol{c^y}$, respectively.

For the ith word x_i in $\boldsymbol{x}$, we consider x_i as a lexical cohesion device if there exists a context word that is the same as x_i, or in the $synset(x_i)$ or $hypset(x_i)$ in $\boldsymbol{c^x}$. We use CoreNLP to parse each concatenated input to obtain co-reference links between $\boldsymbol{x}$ and $\boldsymbol{c^x}$, and refer to words occurring in the detected co-reference links as grammatical cohesion devices. We use $\mathcal{D}$ to denote the set of both lexical and grammatical cohesion devices in $\boldsymbol{x}$.

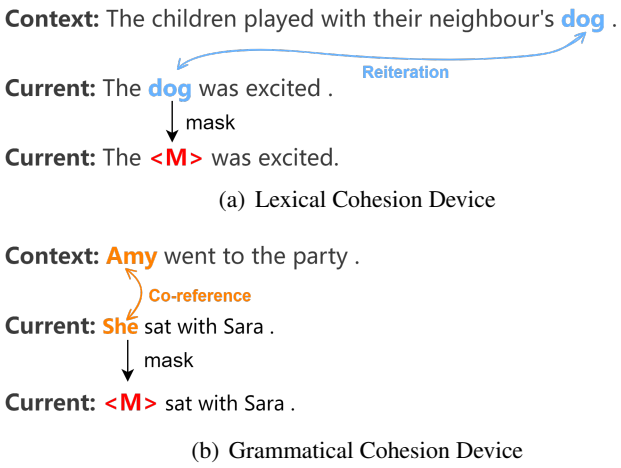

Figure 2: Examples of cohesion device masking. The same words "dog" in Figure 2(a) are reiteration devices. "Amy" and "She" in Figure 2(b) are co-reference devices.

Masking Strategy We mask all cohesion devices in $\mathcal{D}$ to explore contextual dependencies established by these devices as many as possible. Figure 2 shows examples of CoDM.

As there are not many cohesion devices sometimes, only masking cohesion devices is not sufficient. We hence use a masking ratio r as a threshold to mask other words (randomly selected) in addition to cohesion devices. The total number of words being masked in $\boldsymbol{x}$ is $\lceil |\boldsymbol{x}| \times r \rceil$ where $\lceil \cdot \rceil$ indicates the upward rounding operation. We denote the set of masked tokens as $\mathcal{M}$ and the masked version of $\boldsymbol{x}$ as $\acute{\boldsymbol{x}}$ where tokens in $\mathcal{M}$ are substituted by a speical symbol (e.g., <M>). We concatenate $\boldsymbol{c^x}$ and $\acute{\boldsymbol{x}}$ as the input fed into the encoder and use the corresponding hidden states of the last encoder layer to predict the masked tokens in $\mathcal{M}$. The loss of predicting the masked tokens is calculated as follows:

$$\mathcal{L}_{\text{mask}}(\mathcal{M}|\acute{\boldsymbol{x}}, \boldsymbol{c^x}) = -\sum_{i=1}^{|\mathcal{M}|} \log P(\mathcal{M}_i|\acute{\boldsymbol{x}}, \boldsymbol{c^x}) \quad (1)$$

where $\mathcal{M}_i$ is the ith token in $\mathcal{M}$ and $P(\mathcal{M}_i|\acute{\boldsymbol{x}}, \boldsymbol{c^x})$ represents the probability that the model predicts $\mathcal{M}_i$ given the current sentence and its context.

3.2 Cohesion Attention Focusing

Not all information in the context is useful for translating the current sentence. We hence want to guide

the model to attend to only cohesion devices in the context when we translate cohesion devices in the current sentence as they are linguistically linked to each other. We achieve this via a cohesion attention mask.

Constructing the Cohesion Attention Mask We use a key-value pair (x_i, l_{x_i}) to store cohesion devices linked to x_i, where l_{x_i} is a list whose elements are the cohesion devices related to x_i. Specifically, for x_i, if the jth word c_j^x in $\boldsymbol{c^x}$ is the same as x_i, or in the $synset(x_i)$ or $hypset(x_i)$, c_j^x is cohesively linked to x_i. We hence add x_i and c_j^x into $l_{c_j^x}$ and l_{x_i}, respectively. Through CoreNLP, we are able to directly obtain co-reference links between x_i and $\boldsymbol{c^x}$. For each word in each co-reference link, we first find its corresponding cohesion device list l, and then add the remaining words in the co-reference link to l. We denote the collection of all key-value pairs as L.

After obtaining L, we construct a cohesion attention mask $\mathrm{M} \in \mathbb{R}^{N \times N}$ and initialize each item with 0. $N = |\boldsymbol{x}| + |\boldsymbol{x^c}|$ indicates the length of the concatenated source input. We use L to set value for each item in the cohesion attention mask matrix. For each key-value pair (x_i, l_{x_i}), we obtain the positions of x_i (p_{x_i}) and words in l_{x_i} ($p_{w \in l_{x_i}}$) in the concatenated source input. Then, at the p_{x_i} row, we mask out all items whose column positions are not $\in \{p_{w \in l_{x_i}}\}$ by setting their values to $-\infty$. This is similarly done for each column $\in \{p_{w \in l_{x_i}}\}$. Note that we do not mask out (inter/intra-sentential) interactions between ordinary words that are not cohesion devices and intra-sentential interactions among words even when they are not cohesion devices. We only force cohesion devices to exclusively attend to cohesion devices that are linked to them in other sentences.

Figure 3 illustrates the cohesion attention mask. For words (e.g., "from", "home") that are not cohesion devices, they can only attend to words in the same sentence. For cohesion devices, they can attend not only to their intra-sentential context words but also to associated cohesion devices across sentences.

Applying the Cohesion Attention Mask In order to make the model not lose the ability to capture important contextual information, we apply the constructed cohesion attention mask softly by using a probability threshold P_m to control whether to apply the cohesion attention mask to self-attention. Thus the model is trained with and without the

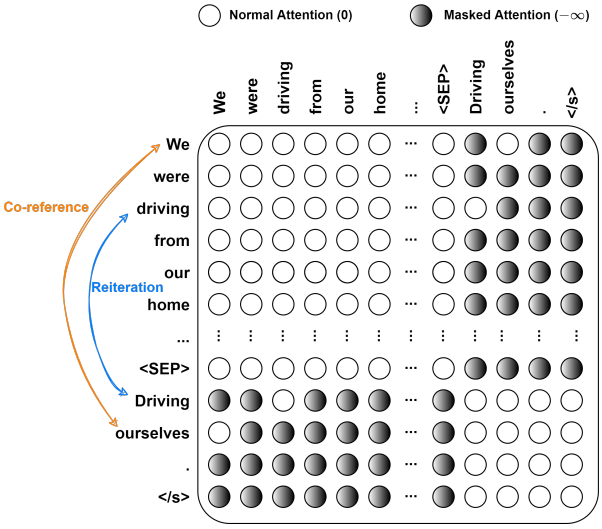

Figure 3: An example of the cohesion attention mask. For brevity, we omit some words in the sentence. The complete sentence is "We were driving from our home to a little farm . <SEP> Driving ourselves . </s>". White positions are set to 0 while black positions are set to $-\infty$.

cohesion attention mask, and we assume that the model is able to acquire the ability to capture cohesion information autonomously with this training strategy. The attention $\boldsymbol{A^l}$ in l-th self-attention can be calculate as follows.

$$\boldsymbol{A^l} = \begin{cases} \mathrm{Softmax}(\frac{\boldsymbol{Q}\boldsymbol{K}^{\mathrm{T}}}{\sqrt{d/h}} + \mathrm{M}) & , \text{if } \varepsilon > P_m \\ \mathrm{Softmax}(\frac{\boldsymbol{Q}\boldsymbol{K}^{\mathrm{T}}}{\sqrt{d/h}}) & , \text{otherwise} \end{cases} \quad (2)$$

where the matrices $\boldsymbol{Q}, \boldsymbol{K}$ represent queries and keys in self-attention. d and h indicate the dimension of hidden states and the number of heads, respectively. ε is sampled from $U \backsim (0, 1)$. Only when the sampled ε is larger than the threshold P_m, the cohesion attention mask is applied.

3.3 Training

Our model performs both masking prediction and translation in a multi-task learning fashion. Hence the training objective of our model is composed of the traditional negative log-likelihood (NLL) and the masked token prediction loss $\mathcal{L}_{mask}$.

During training, we use the ground-truth target context (Teacher-Forcing) while we use the previously decoded output tokens as the target context during inference. As a result, our model suffers from exposure bias. Inspired by Zhang et al. (2019), we mix the ground-truth words with the predicted

words as the decoder input during training to alleviate the exposure bias. We refer to a word predicted by the model as the predicted word. For each word in golden context sentences and current sentence, we use the probability P_o to control whether to replace the ground-truth word with its corresponding predicted word. Following Zhang et al. (2019), we gradually decrease P_o from 1 according to the following decay function:

$$P_o = \frac{\mu}{\mu + \exp\left(e/\mu\right)} \tag{3}$$

where μ is a hyper-parameter that controls the decay rate. e is the index of training epochs starting from 0.

For each step, the predicted words are obtained through word-level greedy search. We denote the mixed target context and current sentence as $\acute{\boldsymbol{c}^{\boldsymbol{y}}}$ and $\acute{\boldsymbol{y}}$, respectively. Thus, the NLL is reformulated as follows:

$$\mathcal{L}_{\text{NLL}}(\boldsymbol{y}|\hat{\boldsymbol{x}}, \boldsymbol{c}^{\boldsymbol{x}}, \acute{\boldsymbol{c}^{\boldsymbol{y}}}) = -\sum_{i} \log P(\boldsymbol{y}_i|\acute{\boldsymbol{y}}_{<i}, \hat{\boldsymbol{x}}, \boldsymbol{c}^{\boldsymbol{x}}, \acute{\boldsymbol{c}^{\boldsymbol{y}}}) \tag{4}$$

Please note that we feed the masked version of the current sentence into the encoder for translation during training. In order to correctly translate tokens in masked positions, our model has to utilize previous context, not only for the masked token prediction task, but also for the translation task.

The two loss functions are integrated as follows:

$$\mathcal{L} = \mathcal{L}_{\text{NLL}}(\boldsymbol{y}|\hat{\boldsymbol{x}}, \boldsymbol{c}^{\boldsymbol{x}}, \acute{\boldsymbol{c}^{\boldsymbol{y}}}) + \lambda \mathcal{L}_{\text{mask}}(\mathcal{M}|\hat{\boldsymbol{x}}, \boldsymbol{c}^{\boldsymbol{x}}) \tag{5}$$

where λ is a hyper-parameter that balances the contribution from the masked token prediction task.

4 Experiments

To examine the effectiveness of our proposed approaches, we carried out experiments on three widely-used datasets and linguistic evaluation on a contrastive test set.

4.1 Data and Settings

Following previous work (Zhang et al., 2021), we used three datasets on two different language pairs as the benchmark datasets, which are TED (Cettolo et al., 2012), Opensubtitles (Maruf et al., 2018) and Europarl7 (Maruf et al., 2018).

Dataset	Language	#Sentences	#Documents
		train/dev/test	train/dev/test
TED	En-De	0.2M/09k/2.2k	1.7k/7/22
Opensubtitles	En-Ru	0.3M/6k/9k	23k/461/693
Europarl7	En-De	0.1M/2k/3.3k	3.6k/69/107

Table 1: Statistics of the used datasets on different language pairs.

- **TED (English-German)**: The corpus contains transcriptions of TED talks from IWSLT 2017. Each talk is used as a document, aligned at the sentence level. *dev2010* was used as our development set and *tst2016-tst2017* for testing.
- **Opensubtitles (English-Russian)**: This corpus is extracted from the OpenSubtitles2016 corpus (Maruf et al., 2018), where sentences are segmented and aligned using additional information.
- **Europarl7 (English-German)**: Following (Maruf et al., 2018), we used the same method to preprocess the raw Europarl v7 Corpus (Koehn, 2005) and extract the parallel coprus.

The statistics of these corpora are shown in Table 1. We used scripts from MOSES (Koehn et al., 2007) to tokenize and truecase sentences. We applied BPE (Sennrich et al., 2016) with 30K merge operations for each language in the datasets. Translation quality was evaluated by BLEU (Papineni et al., 2002). [1]

We followed the same Transformer base setting used in (Vaswani et al., 2017) and trained all models on 4 GeForce RTX 2080 Ti GPU. The dropout was set to 0.1. The masking ratio r was set to 0.15. The weight of the masked token predction loss λ was set to 0.5. The probability threshold P_m was set to 0.5. For TED, we set μ as 10. For Europarl7 and Opensubtitles, we set μ as 12. For inference, we set the beam size to 4. The source code is available at https://github.com/codeboy311/CoDoNMT.

4.2 Baselines

We used five baselines to compare against our model. The sentence-level baseline Sent is the standard Transformer (Vaswani et al., 2017) trained on sentence-level parallel data. The rest four baselines are all document-level NMT, including: 1) DocT

[1]We use sacrebleu to calculate BLEU score for each dataset, and the signature of sacrebleu we used is "BLEU+case.mixed+numrefs.1+smooth.exp+tok.13a+version.1.5.1".

Model	TED En-De	Opensubtitles En-Ru	Europarl7 En-De	AVG	#Param
Sent (Vaswani et al., 2017)	24.30	19.50	30.70	24.83	50M
DocT (Zhang et al., 2018) †	25.04	20.21	30.67	25.31	72M
HAN (Miculicich et al., 2018) †	25.70	20.08	26.61	24.13	70M
CADec (Voita et al., 2019) †	26.08	19.46	30.36	25.30	91M
MHT (Zhang et al., 2021) †	26.22	20.46	31.25	25.97	80M
Our Model	**26.89**	**21.49**	**32.07**	**26.82**	50M

Table 2: Overall results on the TED, OpenSubtitles and Europarl translation tasks. †indicates that the results of baselines are reported from corresponding papers.

Model	Deixis	Lexical Cohesion
Sent (Voita et al., 2019)	50.0%	45.9%
CADec (Voita et al., 2019)	81.6%	58.1%
Concat (Voita et al., 2019)	83.5%	47.5%
Our Model	**89.4%**	**62.3%**

Table 3: Linguistic evaluation results on the contrastive test set (accuracy).

(Zhang et al., 2018) that employs an additional encoder for context, 2) HAN (Miculicich et al., 2018) which integrates document contextual information from both the source and target side through context-aware hierarchical attention networks, 3) CADec (Voita et al., 2019) that explores contextual information to refine sentence-level translation, and 4) MHT (Zhang et al., 2021) that applies a multi-hop mechanism to imitate reasoning process.

4.3 Main Results

Results on the three translation benchmarks are shown in Table 2. As can be seen, our model achieves the highest BLEU scores on all tasks over both sentence- and document-level baselines. Furthermore, we gain improvements of 0.67, 1.03 and 0.82 BLEU points over the strongest document-level baseline on TED, Opensubtitles and Europarl7, respectively, using fewer parameters. This suggests that modeling cohesion devices is able to benefit document-level NMT and could be better than simply integrating full context into NMT without showing relevant context information explicitly.

4.4 Linguistic Evaluation

To further investigate whether our method is able to improve translation cohesion, we conducted linguistic evaluation on Deixis and Lexical Cohesion using a linguistic contrastive test set (Voita et al., 2019). These two discourse phenomena are relevant to the cohesion devices that we attempt to capture (i.e. co-reference and lexical cohesion devices) in our model.

To make a fair comparison, we follow Voita et al. (2019) to use 6M sentence-level instances to train the sentence-level baseline and then use 1.5M document-level instances to train our document-level model. Results are shown in Table 3. Our model achieves significant improvements on Deixis and Lexical Cohesion compared with sentence-level baseline Sent (Voita et al., 2019) and document-level baselines CADec (Voita et al., 2019) and Concat (Voita et al., 2019). This indicates that our model can make better use of context to deal with discourse phenomena.

4.5 Ablation Study

In order to take a deep look into the improvements gained by our model, we further conducted ablation study to investigate the contributions of the three components in our model: 1) cohesion device masking, 2) cohesion attention focusing and 3) exposure bias mitigation introduced in Section 3.3. Results are shown in Table 4. Without using cohesion device masking, CoDoNMT drops by 0.58, 0.62 and 0.55 BLEU on TED, Opensubtitles and Europarl7, respectively. This demonstrates that forcing the model to predict masked cohesion devices is beneficial for document-level NMT. Similarly, without using the other two techniques, we also see performance drops of 0.2 BLEU for the exposure bias mitigation, 0.34 BLEU for cohesion attention focusing. These results validate the effectiveness of the three methods used in CoDoNMT.

Ablation	TED		Opensubtitles		Europarl7		AVG	
	BLEU	Δ	BLEU	Δ	BLEU	Δ	BLEU	Δ
Our	26.89	-	21.49	-	32.07	-	26.82	-
w/o Exposure Bias Mitigation	26.60	-0.29	21.38	-0.11	31.86	-0.21	26.62	-0.20
w/o CoAF	26.53	-0.36	21.16	-0.33	31.75	-0.32	26.48	-0.34
w/o CoDM	26.31	**-0.58**	20.87	**-0.62**	31.52	**-0.55**	26.23	**-0.59**

Table 4: Ablation study results.

Cohesion Device	TED	Opensubtitles	Europarl7	AVG	Deixis	Lexical Cohesion
Lexical	26.58	21.24	31.99	26.60	84.1%	60.3%
Grammatical	26.64	21.21	31.86	26.57	87.4%	58.2%
Both	**26.89**	**21.49**	**32.07**	**26.82**	**89.4%**	**62.3%**

Table 5: Impact of different cohesion types modeled on translation quality on the TED, Opensubtitles and Europarl7 translation tasks (BLEU) and the constrastive test set (accuracy). "Lexical" and "Grammatical" denote the lexical and grammatical cohesion devices.

5 Analysis

In this section, we analyzed three factors to examine their impact on the performance of the proposed model, including: 1) cohesion device type, 2) masking strategy, 3) cohesion attention mask.

5.1 Cohesion Device Type

In CoDoNMT, we take into account both lexical and grammatical cohesion devices. To further understand the effect of the type of cohesion devices on the model performance, we conducted experiments to model them separately during training.

Results on TED, Opensubtitles and Europarl7 are shown in Table 5. As can be seen, when we model both lexical and grammatical cohesion devices, CoDoNMT achieves the highest BLEU scores over the three translation tasks. This indicates that considering both lexical and grammatical cohesion devices is more beneficial to document-level NMT than only modeling one type of cohesion devices.

The performance differences of "Lexical" and "Grammatical" on the three translation tasks in terms of BLEU are slight. We conjecture that it may be because BLEU is not a good metric for discourse phenomena (Xu et al., 2020a). Therefore, we performed another linguistic evaluation (Voita et al., 2019). As shown in Table 5, "Both" continues to achieve the best performance on the two discourse phenomena. This again suggests that considering both lexical and grammatical cohesion devices are more helpful for the model to deal with discourse phenomena. In addition, the results of different types of cohesion devices demonstrate that modeling the corresponding type of cohesion devices can better solve discourse phenomenon relevant to these devices. In other words, lexical cohesion devices can better solve Lexical Cohesion than grammatical cohesion devices, but it is the other way around on Deixis.

5.2 Masking Strategy

Due to the scarcity of cohesion devices, we mask not only cohesion devices, but also a part of remaining tokens of the current sentence randomly. In other words, if the current sentence does not include any cohesion devices, CoDM degenerates to random masking. In order to investigate the effect of masking cohesion devices against randomly masking tokens, we conducted experiments to compare our masking strategy with the random masking strategy. We mask both lexical and grammatical cohesion devices in CoDM. For the random masking, we randomly mask a set of tokens in the current sentence according to the masking ratio r. Note that we do not apply CoAF in this analysis for a fair comparison as CoAF may further improve CoDM.

As shown in Table 6, in terms of BLEU, the differences between the two masking strategies are marginal. However, on the linguistic test, CoDM significantly outperforms Random on both linguistic phenomena. This suggests that predicting cohesion devices could be more efficient to produce document cohesion than predicting other words and reconfirms that BLEU is not sensitive on discourse phenomena (Xu et al., 2020a).

Masking Strategy	TED	Opensubtitles	Europarl7	AVG	Deixis	Lexical Cohesion
CoDM	**26.53**	21.16	**31.75**	**26.48**	**83.40%**	**55.70%**
Random	26.48	**21.26**	31.67	26.47	79.08%	46.20%

Table 6: Comparison on masking strategies on the TED, Opensubtitles and Europarl7 translation tasks (BLEU) and the constrastive test set (accuracy). "Random" denotes the random masking strategy that randomly masks a subset of tokens in the current sentence according to the masking ratio r.

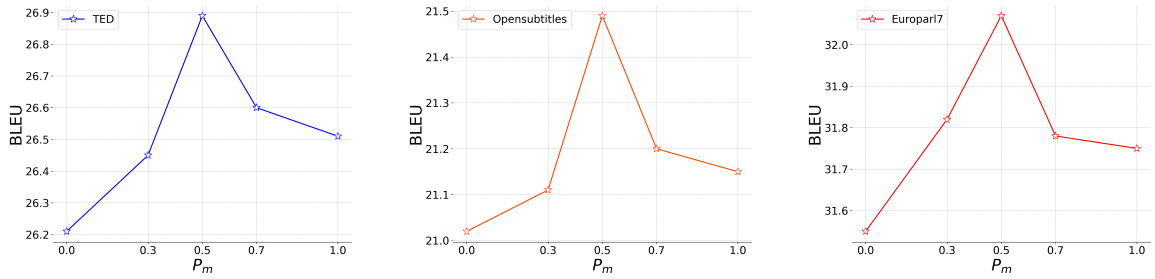

Figure 4: Impact of the probability threshold P_m on the TED, Opensubtitles and Europarl7 translation tasks.

5.3 Cohesion Attention Mask

As shown in Figure 4, we conducted experiments with different probability threshold P_m to explore its effect. As can be seen, when P_m is 0.5, our model achieves the best BLEU score over the three translation tasks. Furthermore, if we rigidly focus the model on cohesion devices (i.e., $P_m = 0.0$, applying the cohesion attention mask in a hard way), the model performance drops significantly. This suggests that a soft application of the cohesion attention mask allows the model to gain the ability to capture cohesion information and maintain the ability to explore other important contextual information.

5.4 Case Study

To better illustrate how our model improves translation quality, we provide an example of Deixis from the linguistic contrastive test set in Table 7, which is translated by both the document-level baseline Concate and our model. The document-level baseline **Concate** is a standard Transformer. We trained **Concate** on the document-level corpus where the current sentence is concatenated to its corresponding previous context by a special token (e.g., <SEP>) on both source- and target-side. As shown in Table 7, according to the translation "твой" in previous context, the word "*you*" in "*Maybe you know.*" should be translated into "ты", instead of "вы". Obviously, **Concate** fails to capture the co-reference information, while our model is able to translate the word correctly.

6 Related Work

Document-level NMT aims to improve translation quality with the aid of contextual information beyond the scope of current sentences. Most previous works focus on the integration of the inter-sentential context into NMT. One typical approach is introducing an addtional encoder to encode context, and then integrate the context representation into the primary encoder and/or the decoder (Zhang et al., 2018; Voita et al., 2018; Kuang and Xiong, 2018; Xu et al., 2020a; Zheng et al., 2020). Miculicich et al. (2018) propose a hierarchical attention model to capture the contextual information from both word and sentence level. Tan et al. (2019) propose a hierarchical model to learn global context for document-level NMT. Voita et al. (2019) introduce a two-pass framework that uses several previous sentences as context to refine the translation generated by a strong sentence-level NMT. Zhang et al. (2021) apply a multi-hop mechanism to document-level NMT to simulate the human-like draft-editing and reasoning process. Liu et al. (2020), Ma et al. (2020) and Bao et al. (2021) combine pre-trained models with document-level NMT. Zhang et al. (2020) pretrain a source context prediction model on a large-scale monolingual document corpus to learn contextualized sentence embeddings.

Yet another research strand is to focus on context selection. Maruf and Haffari (2018), Kuang et al.

Source	c^x	Like your boss said, might get you killed.
	c^x	Well, that's what I keep hearing.
	c^x	Nobody wants to share this dangerous entity's idea with me.
	x	Maybe you know.
Reference	c^x	Как сказал твой босс, это может кончиться твоей смертью.
	c^x	Что ж, я это я слышу все время.
	c^x	Никто не хочет делиться со мной личностью той опасной сущности.
	x	Может, ты знаешь.
Concate	c^x	Как сказал твой босс, может убить тебя.
	c^x	Ну, это то, что я продолжаю слышать.
	c^x	Никто не хочет делиться со мной идеей этой опасной сущности.
	x	Может быть, вы знаете.
Our Model	c^x	Как сказал твой босс, тебя могут убить.
	c^x	ну, это то, что я всегда слышал.
	c^x	никто не хочет делиться со мной мыслями об этой опасной сущности.
	x	может быть, ты знаешь.

Table 7: An example of Deixis translation in English-Russian. x and c^x denote the current sentence and its corresponding previous context, respectively. Blue words indicate the correct translations, while red words are the opposite.

(2018) and Tu et al. (2018) use a cache-like memory network to memorize the translation history, and treat it as context to translate future sentences. Maruf et al. (2019) uses sparse attention to selectively focus on relevant sentences in the document context. Kang et al. (2020) adopt reinforcement learning to select dynamic context for document-level NMT.

Recently, approaches have been proposed to leverage discourse information. Xu et al. (2020b) build directed graphs of documents with intra-sentential and inter-sentential relations and use GCN to obtain the document representation. Lyu et al. (2021) use word links to encourage the model to generate more consistent translations.

Different from the above works, we attempt to leverage cohesion devices to enhance the ability of model to capture inter-sentential contextual information to generate cohesive translations.

7 Conclusion

In this paper, we have presented CoDoNMT for document-level NMT, which models cohesion devices with two key methods, CoDM and CoAF. CoDM masks cohesion devices in the current sentence to force the model to actively explore inter-sentential contextual information. CoAF softly guides the model to focus attention on cohesion devices. Both automatic and linguistic evaluations show that our model can significantly improve translation quality in terms of BLEU and lexical and grammatical cohesion accuracy on a discourse-oriented contrastive test set. Further analyses demonstrate the impact of cohesion device type and masking strategy on translation quality.

8 Acknowledgement

The present research was supported by the Key Research and Development Program of Yunnan Province (Grant No. 202203AA080004-2). We would like to thank the anonymous reviewers for their insightful comments.

References

Guangsheng Bao, Yue Zhang, Zhiyang Teng, Boxing Chen, and Weihua Luo. 2021. G-Transformer for Document-Level Machine Translation. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 3442–3455, Online. Association for Computational Linguistics.

Mauro Cettolo, Christian Girardi, and Marcello Federico. 2012. WIT3: Web Inventory of Transcribed and Translated Talks. In *Proceedings of the 16th Annual conference of the European Association for Machine Translation*, pages 261–268, Trento, Italy. European Association for Machine Translation.

Kenneth W. Church. 2000. Empirical Estimates of Adaptation: The Chance of Two Noriegas is Closer to p/2 than p2. In *COLING 2000 Volume 1: The 18th*

International Conference on Computational Linguistics.

Christiane D. Fellbaum. 2000. WordNet : an Electronic Lexical Database. *Language*, 76:706.

Michael A.K. Halliday and Ruqaiya Hasan. 1976. *Cohesion in English*. Routledge.

Hany Hassan, Anthony Aue, Chang Chen, Vishal Chowdhary, Jonathan Clark, Christian Federmann, Xuedong Huang, Marcin Junczys-Dowmunt, William Lewis, Mu Li, Shujie Liu, Tie-Yan Liu, Renqian Luo, Arul Menezes, Tao Qin, Frank Seide, Xu Tan, Fei Tian, Lijun Wu, Shuangzhi Wu, Yingce Xia, Dongdong Zhang, Zhirui Zhang, and Ming Zhou. 2018. Achieving Human Parity on Automatic Chinese to English News Translation. *CoRR*, abs/1803.05567.

Xiaomian Kang, Yang Zhao, Jiajun Zhang, and Chengqing Zong. 2020. Dynamic Context Selection for Document-level Neural Machine Translation via Reinforcement Learning. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 2242–2254, Online. Association for Computational Linguistics.

Yunsu Kim, Duc Thanh Tran, and Hermann Ney. 2019. When and Why is Document-level Context Useful in Neural Machine Translation? In *Proceedings of the Fourth Workshop on Discourse in Machine Translation (DiscoMT 2019)*, pages 24–34, Hong Kong, China. Association for Computational Linguistics.

Philipp Koehn. 2005. Europarl: A Parallel Corpus for Statistical Machine Translation. In *Proceedings of Machine Translation Summit X: Papers*, pages 79–86, Phuket, Thailand.

Philipp Koehn, Hieu Hoang, Alexandra Birch, Chris Callison-Burch, Marcello Federico, Nicola Bertoldi, Brooke Cowan, Wade Shen, Christine Moran, Richard Zens, Chris Dyer, Ondřej Bojar, Alexandra Constantin, and Evan Herbst. 2007. Moses: Open Source Toolkit for Statistical Machine Translation. In *Proceedings of the 45th Annual Meeting of the Association for Computational Linguistics Companion Volume Proceedings of the Demo and Poster Sessions*, pages 177–180, Prague, Czech Republic. Association for Computational Linguistics.

Shaohui Kuang and Deyi Xiong. 2018. Fusing Recency into Neural Machine Translation with an Inter-Sentence Gate Model. In *Proceedings of the 27th International Conference on Computational Linguistics*, pages 607–617, Santa Fe, New Mexico, USA. Association for Computational Linguistics.

Shaohui Kuang, Deyi Xiong, Weihua Luo, and Guodong Zhou. 2018. Modeling Coherence for Neural Machine Translation with Dynamic and Topic Caches. In *Proceedings of the 27th International Conference on Computational Linguistics*, pages 596–606, Santa Fe, New Mexico, USA. Association for Computational Linguistics.

Bei Li, Hui Liu, Ziyang Wang, Yufan Jiang, Tong Xiao, Jingbo Zhu, Tongran Liu, and changliang Li. 2020. Does Multi-Encoder Help? A Case Study on Context-Aware Neural Machine Translation. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 3512–3518, Online. Association for Computational Linguistics.

Yinhan Liu, Jiatao Gu, Naman Goyal, Xian Li, Sergey Edunov, Marjan Ghazvininejad, Mike Lewis, and Luke Zettlemoyer. 2020. Multilingual Denoising Pre-training for Neural Machine Translation. *Transactions of the Association for Computational Linguistics*, 8(0):726–742.

Xinglin Lyu, Junhui Li, Zhengxian Gong, and Min Zhang. 2021. Encouraging Lexical Translation Consistency for Document-Level Neural Machine Translation. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 3265–3277, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Shuming Ma, Dongdong Zhang, and Ming Zhou. 2020. A Simple and Effective Unified Encoder for Document-Level Machine Translation. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 3505–3511, Online. Association for Computational Linguistics.

Christopher D Manning, Mihai Surdeanu, John Bauer, Jenny Rose Finkel, Steven Bethard, and David McClosky. 2014. The Stanford CoreNLP Natural Language Processing Toolkit. In *Proceedings of 52nd annual meeting of the association for computational linguistics: system demonstrations*, pages 55–60.

Sameen Maruf and Gholamreza Haffari. 2018. Document Context Neural Machine Translation with Memory Networks. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1275–1284, Melbourne, Australia. Association for Computational Linguistics.

Sameen Maruf, André F. T. Martins, and Gholamreza Haffari. 2018. Contextual Neural Model for Translating Bilingual Multi-Speaker Conversations. In *Proceedings of the Third Conference on Machine Translation: Research Papers*, pages 101–112, Brussels, Belgium. Association for Computational Linguistics.

Sameen Maruf, André F. T. Martins, and Gholamreza Haffari. 2019. Selective Attention for Context-aware Neural Machine Translation. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 3092–3102, Minneapolis, Minnesota. Association for Computational Linguistics.

Sameen Maruf, Fahimeh Saleh, and Gholamreza Haffari. 2021. A Survey on Document-level Neural Machine Translation: Methods and Evaluation. *ACM Computing Surveys*, 54(2):1–36.

Lesly Miculicich, Dhananjay Ram, Nikolaos Pappas, and James Henderson. 2018. Document-Level Neural Machine Translation with Hierarchical Attention Networks. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 2947–2954, Brussels, Belgium. Association for Computational Linguistics.

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. Bleu: a Method for Automatic Evaluation of Machine Translation. In *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics*, pages 311–318, Philadelphia, Pennsylvania, USA. Association for Computational Linguistics.

Rico Sennrich, Barry Haddow, and Alexandra Birch. 2016. Neural Machine Translation of Rare Words with Subword Units. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1715–1725, Berlin, Germany. Association for Computational Linguistics.

Xin Tan, Longyin Zhang, Deyi Xiong, and Guodong Zhou. 2019. Hierarchical Modeling of Global Context for Document-Level Neural Machine Translation. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 1576–1585, Hong Kong, China. Association for Computational Linguistics.

Jörg Tiedemann and Yves Scherrer. 2017. Neural Machine Translation with Extended Context. In *Proceedings of the Third Workshop on Discourse in Machine Translation*, pages 82–92, Copenhagen, Denmark. Association for Computational Linguistics.

Zhaopeng Tu, Yang Liu, Shuming Shi, and Tong Zhang. 2018. Learning to Remember Translation History with a Continuous Cache. *Transactions of the Association for Computational Linguistics*, 6(0):407–420.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is All you Need. In I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, editors, *Advances in Neural Information Processing Systems 30*, pages 5998–6008. Curran Associates, Inc.

Elena Voita, Rico Sennrich, and Ivan Titov. 2019. When a Good Translation is Wrong in Context: Context-Aware Machine Translation Improves on Deixis, Ellipsis, and Lexical Cohesion. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 1198–1212, Florence, Italy. Association for Computational Linguistics.

Elena Voita, Pavel Serdyukov, Rico Sennrich, and Ivan Titov. 2018. Context-Aware Neural Machine Translation Learns Anaphora Resolution. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1264–1274, Melbourne, Australia. Association for Computational Linguistics.

Hongfei Xu, Deyi Xiong, Josef van Genabith, and Qiuhui Liu. 2020a. Efficient Context-Aware Neural Machine Translation with Layer-Wise Weighting and Input-Aware Gating. In *Proceedings of the Twenty-Ninth International Joint Conference on Artificial Intelligence, IJCAI-20*, pages 3933–3940. International Joint Conferences on Artificial Intelligence Organization. Main track.

Mingzhou Xu, Liangyou Li, Derek F. Wai, Qun Liu, and Lidia S. Chao. 2020b. Document Graph for Neural Machine Translation. *arXiv:2012.03477 [cs]*. ArXiv: 2012.03477.

Kayo Yin, Patrick Fernandes, Danish Pruthi, Aditi Chaudhary, André F. T. Martins, and Graham Neubig. 2021. Do Context-Aware Translation Models Pay the Right Attention? In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 788–801, Online. Association for Computational Linguistics.

Jiacheng Zhang, Huanbo Luan, Maosong Sun, Feifei Zhai, Jingfang Xu, Min Zhang, and Yang Liu. 2018. Improving the Transformer Translation Model with Document-Level Context. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 533–542, Brussels, Belgium. Association for Computational Linguistics.

Long Zhang, Tong Zhang, Haibo Zhang, Baosong Yang, Wei Ye, and Shikun Zhang. 2021. Multi-Hop Transformer for Document-Level Machine Translation. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 3953–3963, Online. Association for Computational Linguistics.

Pei Zhang, Xu Zhang, Wei Chen, Jian Yu, Yanfeng Wang, and Deyi Xiong. 2020. Learning Contextualized Sentence Representations for Document-Level Neural Machine Translation. In *ECAI 2020 - 24th European Conference on Artificial Intelligence, 29 August-8 September 2020, Santiago de Compostela, Spain, August 29 - September 8, 2020 - Including 10th Conference on Prestigious Applications of Artificial Intelligence (PAIS 2020)*, volume 325 of *Frontiers in Artificial Intelligence and Applications*, pages 2298–2305. IOS Press.

Wen Zhang, Yang Feng, Fandong Meng, Di You, and Qun Liu. 2019. Bridging the Gap between Training and Inference for Neural Machine Translation. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 4334–4343, Florence, Italy. Association for Computational Linguistics.

Zaixiang Zheng, Xiang Yue, Shujian Huang, Jiajun Chen, and Alexandra Birch. 2020. Towards Making the Most of Context in Neural Machine Translation. In *Proceedings of the Twenty-Ninth International Joint Conference on Artificial Intelligence*, pages 3983–3989, Yokohama, Japan. International Joint Conferences on Artificial Intelligence Organization.

Improving Non-Autoregressive Neural Machine Translation via Modeling Localness

Yong Wang[†*] **Xinwei Geng**
[†]Tencent Corporation, China
{seaywang,xwgeng2014}@gmail.com

Abstract

Non-autoregressive translation (NAT) models, which eliminate the sequential dependencies within the target sentence, have achieved remarkable inference speed, but suffer from inferior translation quality. Towards exploring the underlying causes, we carry out a thorough preliminary study on the attention mechanism, which demonstrates the serious weakness in capturing localness compared with conventional autoregressive translation (AT). In response to this problem, we propose to improve the localness of NAT models by explicitly introducing the information about surrounding words. Specifically, temporal convolutions are incorporated into both encoder and decoder sides to obtain localness-aware representations. Extensive experiments on several typical translation datasets show that the proposed method can achieve consistent and significant improvements over strong NAT baselines. Further analyses on the WMT14 En⇒De translation task reveal that compared with baselines, our approach accelerates the convergence in training and can achieve equivalent performance with a reduction of 70% training steps.

1 Introduction

Based on the encoder-decoder framework (Cho et al., 2014; Sutskever et al., 2014), neural machine translation (NMT) (Bahdanau et al., 2015; Gehring et al., 2017; Vaswani et al., 2017) has achieved tremendous success in the past several years due to its excellent performance. Currently, state-of-the-art NMT systems are built in an autoregressive manner, which generates target tokens one by one from continuous representations summarized by the encoder. However, with the constraint of output tokens conditioned on previously generated ones, autoregressive translation (AT) inevitably suffers from serious latency during decoding, which becomes a bottleneck of inference speed.

Towards accelerating the inference process, non-autoregressive neural machine translation (NAT) (Gu et al., 2018; Ghazvininejad et al., 2019; Du et al., 2021; Huang et al., 2022) has been proposed to break the above bottleneck. Instead of sequential generation in autoregressive translation, NAT models output the entire target sentence at once. Unfortunately, removing sequential dependencies within the target sentence brings NAT models with serious weakness in capturing the highly multimodal distribution of target sentences (Gu et al., 2018). Accordingly, NAT models suffer from two kinds of incoherent translations, including *repetitive translations* and *incomplete translations* (Wang et al., 2019), which leads to inferior translation performance.

Since both types of translation errors do not commonly appear in AT models, several works (Li et al., 2019; Wei et al., 2019) have proposed to leverage a well-trained AT model to enhance the training of NAT models. Inspired by this consideration, we empirically carry out a thorough study to present the weakness of NAT by investigating the distinction between NAT and AT models. Specifically, we inspect the attention mechanism on two translation tasks and reveal that, in contrast to autoregressive models, NAT models lack the ability of either modeling the localness (Yang et al., 2018; Ding et al., 2020) or producing localness-aware representations.

Motivated by this observation, we propose to improve non-autoregressive machine translation via explicitly modeling localness. Specifically, we incorporate multi-layer temporal convolutions (MTC) into both encoder and decoder sides to enhance the ability to model localness-aware representations in NAT models. To validate the effectiveness of our approach, we implement this method on two advanced NAT models, namely conditional masked language model (CMLM) (Ghazvininejad et al., 2019) and Vanilla-NAT (Gu et al., 2018).

[*] Work done while at Microsoft Corporation.

Proceedings of the 29th International Conference on Computational Linguistics, pages 5217–5226
October 12–17, 2022.

Extensive experiments on typical translation benchmarks demonstrate that our proposed approach can significantly and consistently improve the translation quality by up to 1.0 BLEU points over a series of strong NAT baselines. Further analyses reveal that our approach enhances the ability to generate localness-aware representations. In addition, the analysis on the WMT14 En⇒De task shows that compared with the original CMLM model, our approach accelerates the convergence in training and achieves comparable performance with a reduction of 70% training steps.

In summary, the contributions of this work are detailed as follows:

- Our study demonstrates the necessity of explicitly modeling localness for non-autoregressive machine translation models.
- We propose a simple yet effective approach to enhance the ability to generate localness-aware representations in NAT models and extensive experiments validate the effectiveness and universality of our approach.
- Further analyses reveal that our approach benefits the translation for long sentences and accelerates the convergence in training.

2 Background

Autoregressive Neural Machine Translation In recent years, autoregressive models, which predict the target sentence sequentially conditioned on translation history, have achieved extraordinary success on machine translation. Specifically, given a source sentence $\mathbf{x} = \{x_1, \ldots, x_I\}$, a standard encoder-decoder autoregressive framework (Cho et al., 2014; Bahdanau et al., 2015) optimizes the conditional probability of a target sentence $\mathbf{y} = \{y_1, \ldots, y_J\}$, namely:

$$P(\mathbf{y}|\mathbf{x};\theta) = \prod_{j=1}^{J} P(y_j|y_{<j}, \mathbf{x};\theta), \quad (1)$$

where $y_{<j}$ indicates the partial translation and θ is a set of trainable parameters. The encoder-decoder framework can be implemented as different choices of architectures, such as recurrent neural network (Bahdanau et al., 2015), convolutional neural network (Gehring et al., 2017) and Transformer (Vaswani et al., 2017). The typical training objective is to maximize log-likelihood on a set of training examples $D = \{[\mathbf{x}^m, \mathbf{y}^m]\}_{m=1}^{M}$:

$$\mathcal{L}(\theta) = \arg\max_{\theta} \sum_{m=1}^{M} \log P(\mathbf{y}^m|\mathbf{x}^m;\theta). \quad (2)$$

In the inference, the conditional dependency on translation history produces the autoregressive property, which predicts the token sequentially based on previous output tokens in the target. This nature of sequential processing results in high latency in translation.

Non-Autoregressive Neural Machine Translation In contrast, Gu et al. (2018) proposed to accelerate the decoding process by generating target tokens in parallel. In practice, by breaking the probabilistic factorization, the prediction is modeled as a product of the probability, which is independent for each token:

$$P(\mathbf{y}|\mathbf{x};\theta) = P(J|\mathbf{x}) \prod_{j=1}^{J} P(y_j|\mathbf{x};\theta), \quad (3)$$

where $P(J|\mathbf{x})$ indicates an auxiliary length predictor, which is used to determine the translation length. NAT models employ an identical encoder as the conventional Transformer architecture, while the decoder is distinct from the original one as it avoids the utilization of causal masks in the self-attention mechanism.

However, due to the lack of explicit dependency within targets, NAT models suffer from serious multi-modality problem (Gu et al., 2018) and significantly degenerate the translation performance. This issue inevitably causes NAT models to suffer from repeated and incomplete translations.

3 Preliminary Study

In this section, we conduct a thorough empirical study on the attention mechanism to observe the distinction between AT and NAT models. In practice, a good probabilistic distribution of attention weights suggests a good alignment between source and target words, and usually leads to more accurate translation (Bahdanau et al., 2015; Luong et al., 2015; Li et al., 2019). Inspired by this observation, we systematically investigate the cross-attention weights on both WMT14 En⇒De and WMT16 En⇒Ro translation tasks.

Specifically, given a source sentence $\mathbf{x} = \{x_1, \ldots, x_I\}$, for the j-th token in the target sentence $\mathbf{y} = \{y_1, \ldots, y_J\}$, the attention probability

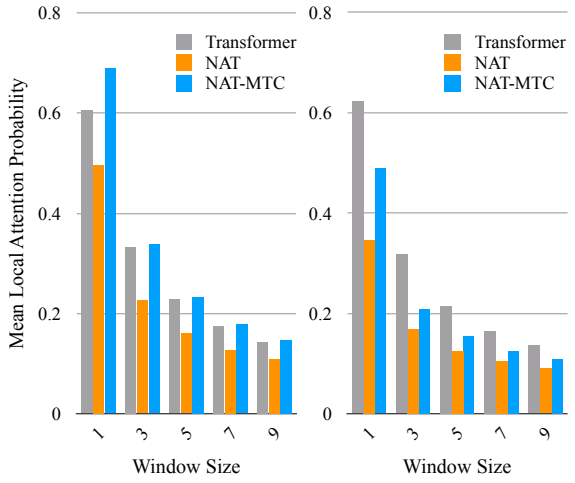

(a) WMT14 En⇒De (b) WMT16 En⇒Ro

Figure 1: Mean local attention probability on WMT14 En-De and WMT16 En-Ro translation tasks, varied by the window size.

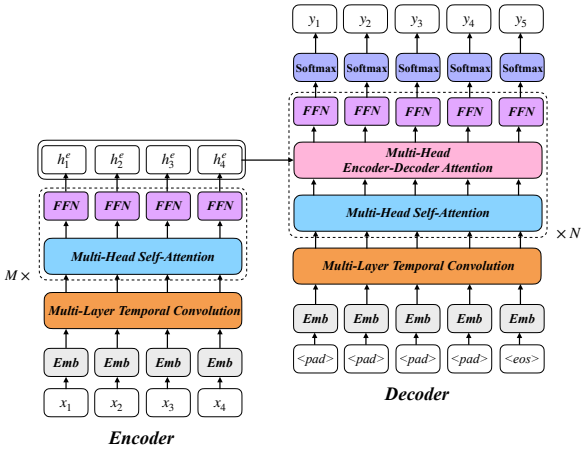

Figure 2: The framework of the proposed model. The left and right figures represent the encoder and decoder respectively. Distinct from conventional NAT models, our model incorporates multi-layer temporal convolutions to model localness for both encoder and decoder.

is computed as $\mathbf{p}_j = \{p_{j1}, p_{j2}, \ldots, p_{jI}\}$. To quantify the extent of centralization on a window, we define a measure γ_j, termed as local attention probability (LAP):

$$\gamma_j = \text{avg}(\mathbf{p}_j(f_k(\arg\max \mathbf{p}_j))), \quad (4)$$

where k is the window size, $\text{avg}(\cdot)$ is the average function, and $f_k(\cdot)$ indicates choosing surrounding indexes based on the central index (i.e., $\arg\max \mathbf{p}_j$). For instance, for $\mathbf{p}_j = [0.2, 0.5, 0.2, 0.1]$ and $k = 3$, $f_k(\cdot)$ returns the indexes $[1, 2, 3]$ and $\gamma_j = (0.2 + 0.5 + 0.2)/3 = 0.3$. Particularly noted that the number of surrounding indexes involved is less than k when the window exceeds the bounds of the sequence. For a target sentence, LAP is computed as $\gamma^m = \frac{1}{J}\sum_j \gamma_j^m$. We average γ^m on all translated sentences to obtain mean local attention probability (MLAP): $\bar{\gamma} = \frac{1}{M}\sum_m \gamma^m$. As seen, MLAP evaluates attention weights on a fixed window within source words, which is considered as local information.

We conduct this study on top of the advanced CMLM model and compare MLAP with the last layer and all heads of the encoder on three models: 1) Transformer (AT model), 2) NAT, and 3) NAT-MTC. As shown in Figure 1, the results reveal two key points. First of all, as the window size increases, the computation of MLAP includes more words. The decrease of MLAP indicates that attentions between target and source words usually focus on neighbor words within the source sentence. Secondly, compared with Transformer, the NAT model holds a smaller MLAP, which demonstrates that it lacks the ability of modeling localness effectively. However, our approach corrects this phenomenon effectively (shown by the bar with color blue vs. the bar with color orange).

4 Approach

In this section, we elaborate the proposed framework of improving NAT models via explicitly modeling localness. First of all, convolutional neural networks benefit from effectively capturing local information and have achieved remarkable success in computer vision (Krizhevsky et al., 2012; He et al., 2016). Inspired by this observation, we augment NAT models with temporal convolutions to enhance localness-aware representations. As shown in Figure 2, we stack temporal convolutions on top of the representations of word embedding. Let $\epsilon \in \mathbb{R}^{|V|\times d}$ denote trainable word embedding. The input sentence $\mathbf{x} = \{x_1, \ldots, x_I\}$ is represented as $\mathbf{E} = \{\mathbf{E}_1, \ldots, \mathbf{E}_I\} = \{\epsilon[x_1]+p_1, \ldots, \epsilon[x_I]+p_I\}$, where p_i is a learnable positional embedding. The incorporated multi-layer temporal convolutions encode the localness-aware representations and the representations of the localness-aware encoder are calculated by:

$$\mathbf{H}_{\text{TC}} = \text{ENC}_{\text{TC}}(\mathbf{E}), \quad (5)$$

$$\mathbf{H}_{\text{SAN}} = \text{ENC}_{\text{SAN}}(\mathbf{H}_{\text{TC}}), \quad (6)$$

where $\text{ENC}_{\text{TC}}(\cdot)$ represents multi-layer temporal convolutions, and $\text{ENC}_{\text{SAN}}(\cdot)$ indicates self-attention networks.

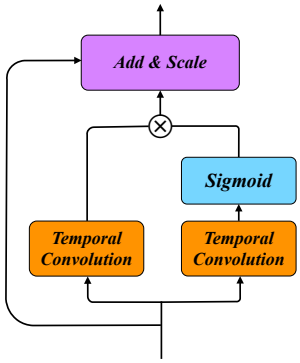

Figure 3: The architecture of temporal convolutions. The add operation indicates the residual connection. The sigmoid and multiplicative boxes demonstrate gated linear units.

Temporal Convolutional Structure In this paper, we employ gated temporal convolutional networks to modeling localness, which is shown in Figure 3. Let $\mathbf{h}^l_{i:j}$ refer to the concatenation of words $\{\mathbf{h}^l_i, \ldots, \mathbf{h}^l_j\}$. A temporal convolution operation over a sequence involves a *filter kernel* $\{W, W_g\} \in \mathbb{R}^{d \times kd}$ and *bias* $\{b, b_g\} \in \mathbb{R}^d$. Following gated linear units (GLU) (Dauphin et al., 2017), the feature $\mathbf{h}'^{l+1}_{(i+j)/2}$ is generated by mapping $\mathbf{h}^l_{i:j}$ to a vector with dimension d:

$$\mathbf{h}'^{l+1}_{(i+j)/2} = (W \cdot \mathbf{h}^l_{i:j} + b) \otimes \sigma(W_g \cdot \mathbf{h}^l_{i:j} + b_g),$$

where $(i + j)/2$ is derived by i and j, and $\otimes$ denotes point-wise multiplication. The gate $\sigma(\cdot)$ is the sigmoid function and it controls the relevance of current context to the inputs.

Inspired by previous works (Dou et al., 2018; Bapna and Firat, 2019), we propose to combine the localness-aware representations with previous representations. Specifically, conditioned on the input latent representations, we employ a residual connection (He et al., 2016) to generate representations: $\mathbf{h}^{l+1}_i = (\mathbf{h}'^{l+1}_i + \mathbf{h}^l_i) \times s$, where we normalize the output by a factor $s = \sqrt{0.5}$ to lower the variance of the sum.

Incorporating into Transformer As a feature extractor, the encoder in NAT models is dealing with a more sophisticated task and takes a more important role than the decoder regarding the translation quality (Guo et al., 2020b). Therefore, it is natural to incorporate multi-layer temporal convolutions into the encoder. Besides, compared with the conventional decoder in autoregressive models, NAT models avoid the causal mask (Gu et al., 2018) in self-attention networks (SAN). Therefore, an identical temporal convolutional structure can be introduced into the decoder accordingly. In our implementation, we use multi-layer stacked temporal convolutions to model localness-aware representations. The presented approach is model-agnostic and can be applied in different NAT models. In this paper, we mainly implement the proposed method on top of advanced CMLM (Ghazvininejad et al., 2019) and Vanilla-NAT (Gu et al., 2018) models.

5 Experiments

5.1 Setup

Datasets We conduct experiments on two typical benchmark datasets: WMT14 English-German (En-De)[1] and WMT16 English-Romanian (En-Ro)[2], which consist of 4.0M and 610K sentence pairs respectively. We strictly follow the dataset settings as previous works (Gu et al., 2018; Lee et al., 2018). Specifically, for the WMT14 En-De translation task, we use newstest2013 and newstest2014 as the validation and test set respectively. For the WMT16 En-Ro task, newsdev2016 and newstest2016 are used as the validation and test set. We follow the tokenization strategy from the translation example of fairseq[3]. We preprocess our data using byte-pair encoding (Sennrich et al., 2016) for both translation tasks, and learn the shared vocabulary with the joint training corpus in both source and target sides. For evaluation, we use 4-gram BLEU score (Papineni et al., 2002) as the evaluation metric for all language pairs.

Knowledge Distillation As a key ingredient, knowledge distillation (KD) (Zhou et al., 2020) has been proven to reduce the complexity of target data and benefits the training of NAT models efficiently. Following previous works (Gu et al., 2018; Lee et al., 2018), we apply sequence-level knowledge distillation (Kim and Rush, 2016) to generate the training data for NAT models. Specifically, for each sentence pair in a parallel training corpus, we replace the target sentence with the generated translation from a pre-trained autoregressive model. We follow Ghazvininejad et al. (2019) to decode the entire training set for both language pairs.

[1] https://www.statmt.org/wmt14/translation-task
[2] https://www.statmt.org/wmt16/translation-task
[3] https://github.com/pytorch/fairseq/blob/master/examples/translation/prepare-wmt14en2de.sh

#	Model	WMT14 En⇒De	WMT14 De⇒En	WMT16 En⇒Ro	WMT16 Ro⇒En
Autoregressive models					
1	Transformer (Vaswani et al., 2017)	27.74	31.09	34.28	33.99
Fully Non-Autoregressive models					
2	Vanilla-NAT (Gu et al., 2018)	17.69	21.47	27.29	29.06
3	FCL-NAT (Guo et al., 2020a)	25.75	29.50	–	–
4	ReorderNAT (Ran et al., 2021)	22.79	27.28	29.30	29.50
5	Flowseq (Ma et al., 2019)	23.72	28.39	29.73	30.72
6	AXE (Ghazvininejad et al., 2020)	23.53	27.90	30.75	31.54
7	Bag-of-ngrams (Shao et al., 2020)	20.90	24.60	28.30	29.30
8	EM+ODD (Sun and Yang, 2020)	24.54	27.93	–	–
9	Imitate-NAT (Wei et al., 2019)	24.15	27.28	31.45	31.81
10	GLAT (Qian et al., 2021)	25.21	29.84	31.19	32.04
Iterative Non-Autoregressive models					
11	Iter-NAT (Lee et al., 2018)	21.61	25.48	29.32	30.19
12	LaNMT (Shu et al., 2020)	26.30	–	–	29.10
13	CMLM (Ghazvininejad et al., 2019)	27.03	30.53	33.08	33.31
Our work					
14	CMLM (Reimp.)	26.73	30.33	33.02	33.39
15	+ MTC	**27.44**	**31.09**	**34.03**	**34.39**

Table 1: Evaluation of translation performance on the test sets of WMT14 En-De and WMT16 En-Ro. "Reimp." indicates the results of corresponding models obtained by our implementation. "+" denotes appending new features to the above row. "–" means not reported.

Model Configuration For model hyper-parameters, we mainly follow the configurations in (Lee et al., 2018; Gu et al., 2019). Specifically, for both translation tasks, we use the hyper-parameters of *transformer-base* ($d_{\text{model}} = 512$, $d_{\text{hidden}} = 2048$, $n_{\text{layer}} = 6$, $n_{\text{head}} = 8$, $p_{\text{dropout}} = 0.3$). We employ $t_{\text{warmup}} = 10,000$ as the warm-up learning rate schedule. In our implementation, the kernel size of temporal convolution is set to 3. We use weight decay 0.01 as well as label smoothing 0.1. We implement our approach with the open-source toolkit - fairseq (Ott et al., 2019)[4]. All the models are trained for 300K updates on 8 NVIDIA Tesla V100 GPUs with a batch size of 128K tokens using Adam optimizer (Kingma and Ba, 2015).

5.2 Results

Overall Results We evaluate the proposed NAT-MTC approach with the standard practice of knowledge distillation on WMT14 En-De and WMT16 En-Ro datasets. Tabel 1 shows the results of our models and previous non-autoregressive baselines. As seen, our re-implementation (Row 14) of CMLM model achieves comparable performance with the original ones (Ghazvininejad et al., 2019) across all translation tasks, which makes the evaluation convincing in this work. Further, our approach (Row 15) can outperform the strong baseline models (Row 14) by 0.71 and 0.76 BLEU points on En⇒De and De⇒En respectively. Encouragingly, on the En⇒Ro task, our model achieves a significant improvement by up to 1.01 BLEU points. These results clearly demonstrate the effectiveness of explicitly modeling localness for NAT models.

Effects of Model Capacity To rule out that the improvement is due to higher modeling capacity, we conduct experiments on matching the number of parameters of our proposed model by adding more self-attention layers to the original CMLM. The results in Table 2 show that although adding self-attention layers has more parameters than our model (88.2M vs. 85.0M), our approach achieves significant improvements (27.44 vs. 26.98). This

[4]https://github.com/pytorch/fairseq

Model	#Enc	#Dec	#Para.	BLEU
Transformer	6	6	64.9M	27.74
CMLM	6	6	66.1M	26.73
	8	8	80.8M	26.87
	9	9	88.2M	26.98
+ MTC	6	6	85.0M	27.44

Table 2: Effects of model capacity on the WMT14 En⇒De task. "#Enc" and "#Dec" indicates the number of self-attention layers respectively. "#Para." denotes the number of trainable parameters.

clearly demonstrates that the improvement of translation quality is due to the inductive bias brought by the architecture modification.

Different Model Architectures To verify the universality of our approach, we also implement our method on top of the non-iterative Vanilla-NAT model (Gu et al., 2018) and the results on WMT14 En-De and WMT16 En-Ro are shown in Table 3. For a fair comparison, we re-implement the Vanilla-NAT model. As seen, our re-implementations greatly outperform original Vanilla-NAT reported by Gu et al. (2018), which makes our evaluation convincing. For instance, compared with the original implementation, our Vanilla-NAT achieves an increase of 2.45 BLEU points on De⇒En task. However, our approach shows a further improvement by 0.91 BLEU points. In particular, on Ro⇒En task, the presented method (+MTC) obtains a significant improvement by 1.43 BLEU scores. This demonstrates the effectiveness and universality of our approach.

Model	En⇒De	De⇒En	En⇒Ro	Ro⇒En
Vanilla-NAT*	17.69	21.47	27.29	29.06
Vanilla-NAT (Reimp.)	19.05	23.92	29.65	28.88
+ MTC	20.02	24.83	30.43	30.31

Table 3: Translation performance on test sets of WMT14 En-De and WMT16 En-Ro. "*" indicates that the results are provided by Gu et al. (2018).

Effects of Decoding Speed To investigate the effects of our approach on decoding speed, we run all models with one sentence at a time on a single GPU and calculate the inference latency on the WMT14 En-De task. The results are shown in Table 4. In contrast to the respective backbone, our model achieves a significant improvement with a very small overhead (2.6× vs. 2.8× for CMLM, 17.4× vs. 17.8× for Vanilla-NAT). This indicates that our approach can efficiently improve the performance of NAT models.

Model	BLEU	Speed
Transformer	27.74	1.0×
CMLM	26.73	2.8×
+ MTC	27.44	2.6×
Vanilla-NAT	19.05	17.8×
+ MTC	20.02	17.4×

Table 4: Decoding speed on the WMT14 En⇒De task.

6 Analysis

In this section, we conduct extensive analyses on the WMT14 En-De translation task to better understand our model in terms of: 1) effects of different strategies, 2) effects of the sentence length, 3) convergence speed, 4) effects of the number of decoding iterations, 5) effects of predicted length candidates, and 6) case study.

#	Encoder	Decoder	BLEU	Δ
1	×	×	26.73	–
2	✓	✓	27.44	+0.71
3	✓	×	27.37	+0.64
4	×	✓	26.87	+0.14

Table 5: Effects of different strategies of incorporating MTC into NAT on the WMT14 En⇒De task.

Effects of Different Strategies We perform experiments on WMT14 En⇒De to investigate the effects of different incorporating strategies, which are shown in Table 5. Specifically, we enumerate the translation results of three strategies, namely introducing MTC into: 1) encoder (Row 3), 2) decoder (Row 4), and 3) both encoder and decoder (Row 2). As seen, augmenting the encoder with MTC improves more BLEU scores than decoder (+0.64 vs. +0.14), and incorporating MTC into both encoder and decoder sides accumulatively achieves the best translation performance (+0.71 BLEU, Row 2). This demonstrates that augmenting encoder with generating localness-aware representations is critical in NAT models and validates the significance of encoder, which was also found by Guo et al. (2020b).

Effects of Sentence Length We investigate the translation results of CMLM and our approach on

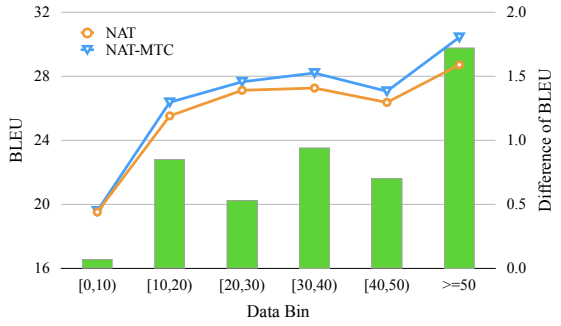

Figure 4: Translation performance on the WMT14 En⇒De test set with respect to different lengths of target sentences. The left axis denotes the BLEU scores of NAT and NAT-MTC models, while the right axis indicates the difference of BLEU scores for both models.

the WMT14 En⇒De task with respect to different lengths of target sentences, which is shown in Figure 4. Specifically, the translations are allocated into distinct buckets based on the respective lengths of corresponding reference sentences and then we evaluate the BLEU scores for each bucket. For comparison, we also show the relative change of BLEU score between NAT-MTC and NAT models. As expected, the presented approach achieves improvements over the baseline system across all buckets. In particular, for longer sentences (≥ 50), our model improves by 1.72 BLEU points. This clearly reveals that explicitly modeling localness effectively benefits long-distance dependencies in NAT models.

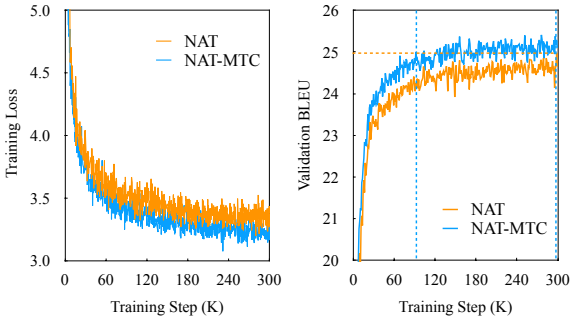

(a) Training Loss (b) Validation BLEU

Figure 5: Learning curves on the WMT14 En⇒De translation task.

Convergence Speed We present the training process to observe the effects of our approach on optimization, which is shown in Figure 5. As seen, our approach converges faster concerning the training loss and validation BLEU score. In particular, the translation performance is significantly boosted during training. Specifically, the NAT model achieves the best BLEU score at 298K training step (indicated by a vertical line with the color yellow in Figure 5b). However, our model achieves an equivalent BLEU point at 92K training steps (indicated by a vertical line with the color blue in Figure 5b). Therefore, our approach largely benefits convergence speed in training and can achieve comparable translation accuracy with a reduction of 70% training steps.

Iter.	BLEU			Rep.		
	NAT	NAT-MTC	Δ	NAT	NAT-MTC	Δ
2	23.12	24.39	+1.27	2.43%	2.08%	-0.35%
4	25.62	26.69	+1.07	0.58%	0.45%	-0.13%
6	26.43	27.16	+0.73	0.31%	0.24%	-0.07%
8	26.45	27.21	+0.76	0.22%	0.16%	-0.06%
10	26.73	27.44	+0.71	0.16%	0.13%	-0.03%

Table 6: The translation performance and percentage of repeating words on the WMT14 En⇒De task, varied by the number of decoding iterations.

Effects of Iteration We study the translation performance and repetitive words as the number of decoding iterations changes, which is shown in Table 6. Specifically, the percentage of repetitive words is defined as $\beta = C_{rep}/C$, where C_{rep} enumerates the number of repetitive words (refer to words, which are equivalent to adjacent words) and C indicates the total number of words within decoding sentences. As seen, with the increase of the number of iterations, our model improves the translation performance, while reducing repetitive words. Besides, our approach reduces more repeating words (-0.35%) while obtaining more improvements in translation performance (+1.27 BLEU). This clearly confirms that our approach benefits the reduction of repetitive translations and alleviates the multi-modality problem in NAT models.

#Cands.	NAT	NAT-MTC	Δ
1	26.08	27.26	+1.18
2	26.04	27.32	+1.28
3	26.73	27.51	+0.78
4	26.50	27.40	+0.90
5	26.73	27.44	+0.71

Table 7: Translation performance on the WMT14 En⇒De task. "#Cands." indicates the number of candidates with different predicted length.

Effects of Length Candidates We present the translation performance with respect to the change

Source	Er selbst war im Jahr 2004 das erste Mal in Mauretanien im Cheijk-Zajed-Krankenhaus in Nouakchott.
Target	In 2004, he visited the **Cheijk-Zajed Hospital** in Nouakchott in Mauritania *for the first time*.
NAT	He himself was in Mauritania at the **Cheijk Zajed Cheijk Hospital** in Nouakchott in 2004.
NAT-MTC	He himself was in Mauritania *for the first time* at the **Cheijk-Zajed Hospital** in Nouakchott in 2004.

Table 8: A case study on the WMT14 De⇒En translation task. Phrases formatted as bold or underline indicate the problem of repetitive and incomplete translations in the baseline but fixed by our model.

of predicted length candidates. As shown in Table 7, when the number of candidates is reduced to one, our model drops by fewer BLEU scores compared with the NAT model (-0.18 vs. -0.65). In addition, our model performs extremely stable with the change of length candidates. These observations verify the robustness of our approach.

Case Study We further carry out a case study to intuitively illustrate the performance of our approach and baseline. Table 8 shows an example randomly selected from the test set on the WMT14 De⇒En translation task. As seen, introducing the mechanism of generating localness-aware representations into NAT can produce more fluent and adequate translations. For instance, the German words "*das erste Mal*" are ignored by baseline, while the NAT-MTC model accurately translates it into "*for the first time*". Besides, NAT tends to generate repetitive words (e.g., "*Cheijk Zajed Cheijk Hospital*"), while our model corrects this issue. This demonstrates that our model can comprehensively generate localness-aware representations in terms of words, phrases and patterns.

7 Related Work

Fully Non-Autoregressive Models Gu et al. (2018) first introduced non-autoregressive machine translation, which enables the sequence generation in parallel and reduces the inference latency significantly. Specifically, through sequence-level knowledge distillation (Kim and Rush, 2016) and modeling fertility as a latent variable, it maintains a relatively competitive translation quality as opposed to the autoregressive Transformer. The idea of modeling dependency as latent variables has been investigated extensively (Kaiser et al., 2018; Sun and Yang, 2020; Gu and Kong, 2021; Qian et al., 2021; Du et al., 2021). Kaiser et al. (2018) proposed to model a shorter sequence as discrete latent variables, which are generated autoregressively. Subsequently, this short latent sequence is utilized to decode the output sequence in parallel. In addition, a glancing mechanism with adaptively sampling words from the reference (Qian et al., 2021) was exploited to improve the translation performance of non-iterative NAT.

Non-Autoregressive Models with Iterative Refinement To alleviate the multi-modality problem, a line of researches (Lee et al., 2018; Stern et al., 2018; Ghazvininejad et al., 2019; Gu et al., 2019; Saharia et al., 2020; Ding et al., 2021, 2022; Huang et al., 2022) introduce an iterative refinement process to maintain the translation accuracy. Lee et al. (2018) first presented an iterative approach, which interprets the entire model as a latent variable and each refinement step as a denoising process. In addition, Stern et al. (2018) proposed to make predictions for multiple time steps by introducing a blockwise parallel decoding scheme. Inspired by the pretraining approach (Devlin et al., 2019; Lample and Conneau, 2019), Ghazvininejad et al. (2019) utilized a masked language modeling to predict any subset of the target sentence, which is based on both the source sentence and a generated translation with partially masking.

8 Conclusion

In this paper, we propose to improve NAT models via explicitly modeling localness. First of all, we conduct a thorough empirical study on the attention mechanism and reveal that compared with autoregressive models, existing NAT models lack the ability to effectively modeling local information. Furthermore, we incorporate temporal con-

volutions into both encoder and decoder sides to enhance localness-aware representations in NAT models. Empirical results on a variety of language pairs and two advanced NAT models demonstrate the effectiveness and universality of our approach. Further analyses confirm that the proposed method benefits translations for long sentences and accelerates convergence during training.

References

Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. 2015. Neural machine translation by jointly learning to align and translate. In *ICLR*.

Ankur Bapna and Orhan Firat. 2019. Simple, scalable adaptation for neural machine translation. In *EMNLP*.

Kyunghyun Cho, Bart van Merriënboer, Caglar Gulcehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. 2014. Learning phrase representations using rnn encoder–decoder for statistical machine translation. In *EMNLP*.

Yann N Dauphin, Angela Fan, Michael Auli, and David Grangier. 2017. Language modeling with gated convolutional networks. In *ICML*.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. Bert: Pre-training of deep bidirectional transformers for language understanding. In *NAACL*.

Liang Ding, Longyue Wang, Xuebo Liu, Derek F Wong, Dacheng Tao, and Zhaopeng Tu. 2021. Progressive multi-granularity training for non-autoregressive translation. In *ACL*.

Liang Ding, Longyue Wang, Shuming Shi, Dacheng Tao, and Zhaopeng Tu. 2022. Redistributing low-frequency words: Making the most of monolingual data in non-autoregressive translation. In *ACL*.

Liang Ding, Longyue Wang, Di Wu, Dacheng Tao, and Zhaopeng Tu. 2020. Context-aware cross-attention for non-autoregressive translation. In *COLING*.

Zi-Yi Dou, Zhaopeng Tu, Xing Wang, Shuming Shi, and Tong Zhang. 2018. Exploiting deep representations for neural machine translation. In *EMNLP*.

Cunxiao Du, Zhaopeng Tu, and Jing Jiang. 2021. Order-agnostic cross entropy for non-autoregressive machine translation. In *ICML*.

Jonas Gehring, Michael Auli, David Grangier, Denis Yarats, and Yann N Dauphin. 2017. Convolutional sequence to sequence learning. In *ICML*.

Marjan Ghazvininejad, Vladimir Karpukhin, Luke Zettlemoyer, and Omer Levy. 2020. Aligned cross entropy for non-autoregressive machine translation. In *ICML*.

Marjan Ghazvininejad, Omer Levy, Yinhan Liu, and Luke Zettlemoyer. 2019. Mask-predict: Parallel decoding of conditional masked language models. In *EMNLP*.

Jiatao Gu, James Bradbury, Caiming Xiong, Victor OK Li, and Richard Socher. 2018. Non-autoregressive neural machine translation. In *ICLR*.

Jiatao Gu and Xiang Kong. 2021. Fully non-autoregressive neural machine translation: Tricks of the trade. In *ACL-Findings*.

Jiatao Gu, Changhan Wang, and Junbo Zhao. 2019. Levenshtein transformer. In *NeurIPS*.

Junliang Guo, Xu Tan, Linli Xu, Tao Qin, Enhong Chen, and Tie-Yan Liu. 2020a. Fine-tuning by curriculum learning for non-autoregressive neural machine translation. In *AAAI*.

Junliang Guo, Linli Xu, and Enhong Chen. 2020b. Jointly masked sequence-to-sequence model for non-autoregressive neural machine translation. In *ACL*.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. 2016. Deep residual learning for image recognition. In *CVPR*.

Xiao Shi Huang, Felipe Pérez, and Maksims Volkovs. 2022. Improving non-autoregressive translation models without distillation. In *ICLR*.

Lukasz Kaiser, Samy Bengio, Aurko Roy, Ashish Vaswani, Niki Parmar, Jakob Uszkoreit, and Noam Shazeer. 2018. Fast decoding in sequence models using discrete latent variables. In *ICML*.

Yoon Kim and Alexander M Rush. 2016. Sequence-level knowledge distillation. In *EMNLP*.

Diederik P Kingma and Jimmy Ba. 2015. Adam: A method for stochastic optimization. In *ICLR*.

Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. 2012. Imagenet classification with deep convolutional neural networks. In *NeurIPS*.

Guillaume Lample and Alexis Conneau. 2019. Cross-lingual language model pretraining. In *NeurIPS*.

Jason Lee, Elman Mansimov, and Kyunghyun Cho. 2018. Deterministic non-autoregressive neural sequence modeling by iterative refinement. In *EMNLP*.

Zhuohan Li, Di He, Fei Tian, Tao Qin, Liwei Wang, and Tie-Yan Liu. 2019. Hint-based training for non-autoregressive translation. In *EMNLP*.

Minh-Thang Luong, Hieu Pham, and Christopher D Manning. 2015. Effective approaches to attention-based neural machine translation. In *EMNLP*.

Xuezhe Ma, Chunting Zhou, Xian Li, Graham Neubig, and Eduard Hovy. 2019. Flowseq: Non-autoregressive conditional sequence generation with generative flow. In *EMNLP*.

Myle Ott, Sergey Edunov, Alexei Baevski, Angela Fan, Sam Gross, Nathan Ng, David Grangier, and Michael Auli. 2019. fairseq: A fast, extensible toolkit for sequence modeling. In *NAACL-HLT: Demonstrations*.

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. Bleu: a method for automatic evaluation of machine translation. In *ACL*.

Lihua Qian, Hao Zhou, Yu Bao, Mingxuan Wang, Lin Qiu, Weinan Zhang, Yong Yu, and Lei Li. 2021. Glancing transformer for non-autoregressive neural machine translation. In *ACL*.

Qiu Ran, Yankai Lin, Peng Li, and Jie Zhou. 2021. Guiding non-autoregressive neural machine translation decoding with reordering information. In *AAAI*.

Chitwan Saharia, William Chan, Saurabh Saxena, and Mohammad Norouzi. 2020. Non-autoregressive machine translation with latent alignments. In *EMNLP*.

Rico Sennrich, Barry Haddow, and Alexandra Birch. 2016. Neural machine translation of rare words with subword units. In *ACL*.

Chenze Shao, Jinchao Zhang, Yang Feng, Fandong Meng, and Jie Zhou. 2020. Minimizing the bag-of-ngrams difference for non-autoregressive neural machine translation. In *AAAI*.

Raphael Shu, Jason Lee, Hideki Nakayama, and Kyunghyun Cho. 2020. Latent-variable non-autoregressive neural machine translation with deterministic inference using a delta posterior. In *AAAI*.

Mitchell Stern, Noam Shazeer, and Jakob Uszkoreit. 2018. Blockwise parallel decoding for deep autoregressive models. In *NeurIPS*.

Zhiqing Sun and Yiming Yang. 2020. An em approach to non-autoregressive conditional sequence generation. In *ICML*.

Ilya Sutskever, Oriol Vinyals, and Quoc V Le. 2014. Sequence to sequence learning with neural networks. In *NeurIPS*.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *NeurIPS*.

Yiren Wang, Fei Tian, Di He, Tao Qin, ChengXiang Zhai, and Tie-Yan Liu. 2019. Non-autoregressive machine translation with auxiliary regularization. In *AAAI*.

Bingzhen Wei, Mingxuan Wang, Hao Zhou, Junyang Lin, and Xu Sun. 2019. Imitation learning for non-autoregressive neural machine translation. In *ACL*.

Baosong Yang, Zhaopeng Tu, Derek F. Wong, Fandong Meng, Lidia S. Chao, and Tong Zhang. 2018. Modeling localness for self-attention networks. In *EMNLP*.

Chunting Zhou, Jiatao Gu, and Graham Neubig. 2020. Understanding knowledge distillation in non-autoregressive machine translation. In *ICLR*.

Categorizing Semantic Representations for Neural Machine Translation

Yongjing Yin[1,2]*, **Yafu Li**[1,2], **Fandong Meng**[3], **Jie Zhou**[3], **Yue Zhang**[2,4]
[1] Zhejiang University
[2] School of Engineering, Westlake University
[3] Pattern Recognition Center, WeChat AI, Tencent Inc
[4] Institute of Advanced Technology, Westlake Institute for Advanced Study
{yinyongjing,liyafu}@westlake.edu.cn {fandongmeng,withtomzhou}@tencent.com
yue.zhang@wias.org.cn

Abstract

Modern neural machine translation (NMT) models have achieved competitive performance in standard benchmarks. However, they have recently been shown to suffer limitation in compositional generalization, failing to effectively learn the translation of atoms (e.g., words) and their semantic composition (e.g., modification) from seen compounds (e.g., phrases), and thus suffering from significantly weakened translation performance on unseen compounds during inference. We address this issue by introducing categorization to the source contextualized representations. The main idea is to enhance generalization by reducing sparsity and overfitting, which is achieved by finding prototypes of token representations over the training set and integrating their embeddings into the source encoding. Experiments on a dedicated MT dataset (i.e., CoGnition) show that our method reduces compositional generalization error rates by 24% error reduction. In addition, our conceptually simple method gives consistently better results than the Transformer baseline on a range of general MT datasets.

1 Introduction

Neural machine translation (NMT) has achieved competitive performance on benchmark datasets such as WMT (Vaswani et al., 2017; Edunov et al., 2018; So et al., 2019). However, the generalizaiton to low-resource domains (Bapna and Firat, 2019b; Zeng et al., 2019; Bapna and Firat, 2019a; Khandelwal et al., 2021) and robustness to slight input perturbations (Belinkov and Bisk, 2018; Xu et al., 2021b) are relatively low for NMT models. In addition, recent studies show that NMT systems are vulnerable to *compositional generalization* (Lake and Baroni, 2018; Raunak et al., 2019; Guo et al., 2020; Li et al., 2021; Dankers et al., 2021; Chaabouni et al., 2021), namely the ability to understand and produce a potentially infinite (formally exponential to the input size) number of novel combinations of known atoms (Chomsky, 2009; Montague and Thomason, 1975; Janssen and Partee, 1997; Lake and Baroni, 2018; Keysers et al., 2020a).

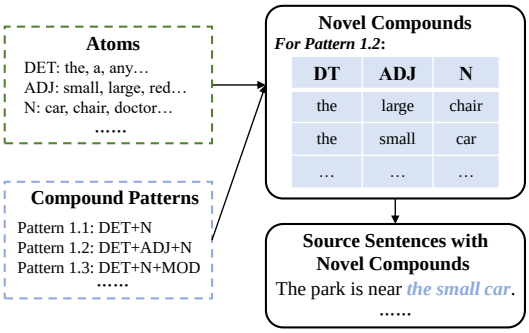

Figure 1: The novel compounds in the CoGnition test set are constructed by composing a few basis semantic atoms (e.g., determiners (DET), nouns (N), and adjectives (ADJ)) according to the composition patterns. The compounds are then put into corresponding source contexts extracted from the training data.

Take CoGnition (Li et al., 2021), a dedicated MT dataset, for example (Figure 1). Despite that certain instances of translation atoms (e.g., *small*, *large*, *car*, and *chair*) and their semantic compositions (e.g., *small chair* and *large car*) are frequent in training data, unseen compositions of the same atoms (e.g., *large chair*) during testing can suffer from large translation error rates. Compositionality is also a fundamental issue in language understanding and motivated for translation (Janssen and Partee, 1997; Janssen, 1998), which has been suggested as being essential for robust translation (Raunak et al., 2019; Li et al., 2021) and efficient low-resource learning (Chaabouni et al., 2021).

The current dominant method to NMT employs a sequence-to-sequence architecture (Sutskever et al., 2014; Vaswani et al., 2017), where an encoder is used to find representations of each input token that thoroughly integrates its sequence-level context in-

*This work was done as an intern at Pattern Recognition Center, WeChat AI, Tencent Inc, China.

Proceedings of the 29th International Conference on Computational Linguistics, pages 5227–5239
October 12–17, 2022.

formation, and a decoder refers to such contextualized representations for generating a translation sequence. A key reason of failure on compositional generalization is that the correspondence between pairs of *token sequences* is modeled as a whole. Specifically, NMT models are trained end-to-end over large parallel data without disentangling the representation of individual words or phrases from that of whole token sequences. At the sequence level, the source input sample space is highly sparse mainly due to semantic composition, and small changes to a sentence can lead to out-of-distribution issues (Sagawa et al., 2020; Conklin et al., 2021; Liu et al., 2021).

Intuitively, one way to solve this problem is to decouple token-level information from the source sequence by injecting token-level translation distribution (e.g., $P(petit|small)$) into the source representation. Given the fact that the source-side contextualized representations encode rich token-level translation information (Kasai et al., 2021; Xu et al., 2021a), we categorize sparse token-level contextualized source representations into a few representative prototypes over training instances, and make use of them to enrich source encoding. In this way, when encoding a sequence, the model observes less sparse prototypes of each token, thus alleviating excessively memorizing the sequence-level information.

We propose a two-stage framework to train prototype-based Transformer models (Proto-Transformer). In the first stage, we warm up an initial Transformer model which can generate reasonable representations. In the second stage, for each token, we run the trained model to extract all contextualized representations over the training corpus. Then, we perform clustering (e.g., K-Means) to obtain the prototype representations for each token. Take Figure 2 as an example, for the token "Toy", we collect all the contextualized representations and cluster them into 3 prototypes. Finally, we extend the base model by fusing the prototype information back into the encoding process through a prototype-attention module, and continue to train the whole model until convergence.

Experimental results on CoGnition show that our method significantly improves novel composition translation by over 24% error reduction, demonstrating the effectiveness for tackling the compositional generalization problem. To further verify the effectiveness on more datasets, we conduct experiments on 10 commonly used MT benchmarks and our method gives consistent BLEU improvement. We also present empirical analysis for prototypes and quantitative analysis on compositional generalizaiton. The comparison between the one-pass and the two-pass training procedure shows that the one-pass method is both faster and more accurate than the two-pass one, demonstrating that more generalizable prototypes extracted from early training phrase are more beneficial to compositional generalization. Additionally, quantitative analysis demonstrates that our proposed model is better at handling longer compounds and more difficult composition patterns. The code is publicly available at https://github.com/ARIES-LM/CatMT4CG.git.

2 Related Work

Compositional Generalization Recent work (Lake and Baroni, 2018; Keysers et al., 2020b) has demonstrated weak compositionality of neural models using dedicated datasets. Various methods haven been proposed to solve the issue of compositional generalization such as encoding more inductive bias (Li et al., 2019; Korrel et al., 2019; Baan et al., 2019; Chen et al., 2020a; Gordon et al., 2020; Herzig and Berant, 2021), meta-learning (Lake, 2019; Conklin et al., 2021), and data augmentation (Andreas, 2020; Akyürek et al., 2021). Recently, Ontañón et al. (2021) and Csordás et al. (2021) show that the Transformer architecture can perform better on compositional generalization with some modifications. Although these methods have demonstrated better generalization or interpretability, most of them are limited small vocabulary and limited samples semantic parsing datasets. In the context of machine translation, Lake and Baroni (2018) construct a small dataset where the training data contains a word *daxy* along with its parallel sentences of a single pattern (e.g., *I am daxy*, *je suis daxist*) while the test set contains novel patterns (e.g., *He is daxy*). However, the experiment is limited in that the test set only consists of 8 samples. Different from existing work, Li et al. (2021) propose a large dataset (CoGnition) and construct a large-scale test set that contains newly constructed constituents as novel compounds, so that generalization ability can be evaluated directly based on compound translation error rate. We proposed a method enhancing compositional generalization on the dedicated dataset of Li et al. (2021), while at the same time gives improvements to the machine

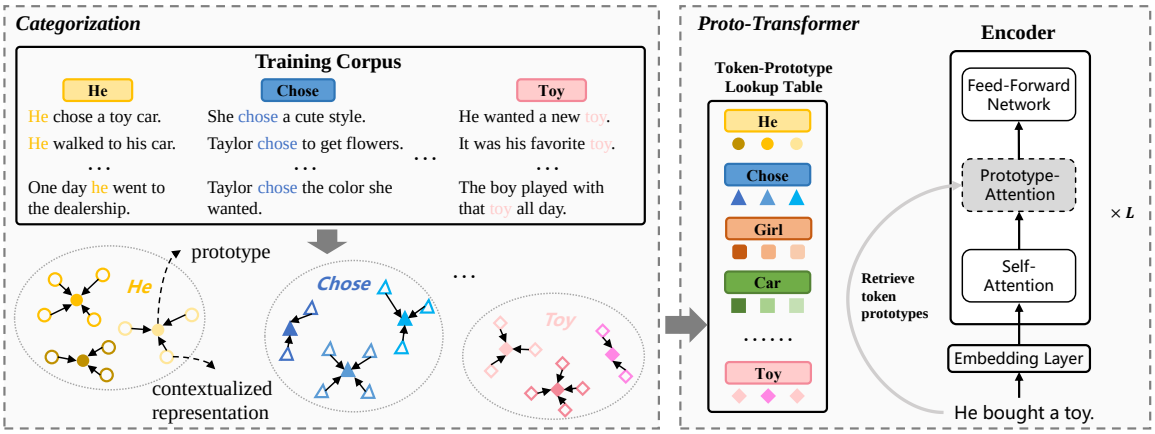

Figure 2: Architecture of Proto-Transformer. The dotted box denotes the prototype-attention introduced in stage 2.

translation quality at practical test cases.

Neural Machine Translation Recent research on NMT has paid increasing attention to robustness (Cheng et al., 2018, 2020; Xu et al., 2021b), domain adaptation (Bapna and Firat, 2019b; Zeng et al., 2019; Bapna and Firat, 2019a; Khandelwal et al., 2021), and compositional generalization (Lake and Baroni, 2018; Raunak et al., 2019; Fadaee and Monz, 2020; Guo et al., 2020; Li et al., 2021). Lake and Baroni (2018) propose a simple toy experiment to first show the problem of compositionality. Raunak et al. (2019) find that NMT models behave poorly on recombining known parts and generalizing on samples beyond the observed length during training, Fadaee and Monz (2020) find that NMT models are vulnerable to modifications such as removal of ad-verbs and number substitutions. More recently, Li et al. (2021) observe significant compositional generalization issues on CoGnition, and Dankers et al. (2021) argue that MT is a suitable testing ground to ask how compositional models trained on natural data are. Our work is in line with the above methods, but we consider a method to address the issue rather than analyse the problem. Technically, Raunak et al. (2019) propose to use bag-of-word regularization to refine encoder and Guo et al. (2020) propose sequence-level mixup to create synthetic samples. Different from them, we propose to enhance models' compositional generalization by categorizing contextualized representations, which turns out more effective.

3 Method

3.1 Transformer Baseline

Given a sequence of source sentence $X = \{x_1, ..., x_T\}$, where T denotes the number of tokens, the Transformer encoder (Vaswani et al., 2017) first maps X to embeddings, packing them as a matrix H^0, and then takes H^0 as input and outputs a contextualized sequence representation $H^L \in \mathbb{R}^{d \times T}$, where d and L denote dimension size and the number of layers respectively.

Attention. Formally, given a set of packed query, key, and value matrices Q, K, and V, the dot product attention mechanism are defined as

$$\text{Attention}(Q, K, V) = \text{Softmax}(\frac{Q^T K}{\sqrt{d}})V, \quad (1)$$

where d is the dimension of the key vector.

A typical extension of the above is multi-head attention (MHA), where multiple linear projections are executed in parallel, and the outputs of all heads are concatenated:

$$\text{MHA}(Q, K, V) = W^O[head_1; ...; head_h], \quad (2)$$

$$head_i = \text{Attention}(W_i^Q Q, W_i^K K, W_i^V V), \quad (3)$$

where W^O, W_i^Q, W_i^K, and W_i^V are model parameters.

Layer Structure. The Transformer encoder has L identical layers, each of which is composed of two sublayers (i.e., self-attention and feed-forward networks). In the l-th self-attention layer, the query, key, and value matrices are all the hidden states from the previous layer H^{l-1}:

$$H_a^l = \text{MHA}(H^{l-1}, H^{l-1}, H^{l-1}). \quad (4)$$

The feed-forward sublayer is a two-layer transformation with ReLU activation:

$$H^l = W_2^l \text{ReLU}(W_1^l H_a^l + b_1^l) + b_2^l, \quad (5)$$

where W_1^l, b_1^l, W_2^l, and b_2^l are trainable parameters. The layer normalization and residual connection are omitted for brevity.

3.2 Proto-Transformer

Our proposed Proto-Transformer extends the base Transformer by introducing a prototype-attention module on top of the self-attention module in each encoder layer, which aggregates token-level prototype representations for each token.

3.2.1 Prototype-attention

Assuming that each token x_t in the input sequence is assigned a set of prototype vectors packed as a matrix $C^{x_t} \in \mathbb{R}^{d \times k}$ where k is the number of prototypes. The prototype attention aggregates the global prototype information and refines the contextualized representations for each token by the multi-head attention mechanism:

$$H_p^l = \text{MHA}(H_a^l, C, C), \tag{6}$$

where C is all the prototype representations of the sequence (i.e., $\{C^{x_1}, C^{x_1}, ..., C^{x_T}\}$), and the localness of prototype attention is implemented by mask mechanisms. The output H_p^l is fed into the feed-forward network (Eq. 5).

The difference between the Proto-Transformer encoder and the Transformer encoder is illustrated by the dotted box in Figure 2. By stacking multiple self-attention layers with prototype-attention, the encoder is able to exploit less-sparse prototype representations, preventing the model from over-memorizing the local context.

3.2.2 Training

Proto-Transformer takes a two-stage training process, which is summarized in Algorithm 1. Here, $\mathcal{D}$ denotes a training corpus, $\mathcal{V}$ denotes the vocabulary of $\mathcal{D}$, $\theta^{(0)}$ denotes initial parameters of a Transformer model, θ_+ denotes parameters of the prototype-attention module, and k and N denote the number of prototypes and training epochs of stage 1 respectively.

Stage 1. We first train a base Transformer model for N epochs until it is able to generate reasonable translations. Given the training corpus $\mathcal{D} = \{(X, Y)\}$, where X and Y denote a source sentence and target translation, respectively, the model is optimized by minimizing cross-entropy loss. After training for N epochs we obtain a model $\theta^{(N)}$, which has acquired some translation knowledge.

Algorithm 1 The training procedure of Proto-Transformer

Input: Training set $\mathcal{D} = \{(X, Y)\}$, vocabulary $\mathcal{V}$, NMT model $\theta^{(0)}$, prototype-attention modules θ_+, number of prototype k, training epochs of stage one N, token-representation lookup table $\mathcal{Q}$, token-prototype lookup table $\mathcal{P}$
Output: Trained NMT model θ^*
Initialize $\theta^{(0)}$ and θ_+
Stage 1
for $i = 1$ **to** N **do**
  $\theta^{(i)} \leftarrow$ TrainModel($\mathcal{D}$, $\theta^{(i-1)}$)
end for
Stage 2
for $X \in \mathcal{D}$ **do**
  for each token x_i in X **do**
    Add H_i to the corresponding contextualized representation list $\mathcal{Q}[x_i]$ for token x_i
  end for
end for
Apply K-Means to each token $v_j \in \mathcal{V}$ to obtain prototype representations C^{v_j} and add (v_j, C^{v_j}) to $\mathcal{P}$
Introduce prototype-attention modules to the base model: $\theta^{(N)} \leftarrow \theta^{(N)} \cup \theta_+$
Continue training to convergence:
$\theta^* \leftarrow$TrainModel($\mathcal{D}$, $\theta^{(N)}$,$\mathcal{P}$)

Stage 2. We cluster the contextualized representations of each token to obtain its prototype representations. In particular, using the model $\theta^{(N)}$, we build a token-representation lookup table $\mathcal{Q}$. We iterate through the whole training corpus and calculate contextualized representations $\{H_1^{v_i}, ..., H_{R(v_i)}^{v_i}\}$ of each source token v_i in the vocabulary except punctuation, where $R(v_i)$ is the number of contextualized representations of token v_i. Here we omit the superscript L for simplicity. Next, for each token v_i, we use K-Means (Lloyd, 1982) [1] to cluster the contextualized representations due to its efficiency for large number of samples in high dimensions:

$$C^{v_i} = \text{K-Means}(H_1^{v_i}, ..., H_{R(v_i)}^{v_i}). \tag{7}$$

All token-prototype pairs are saved in a lookup table $\mathcal{P}$. Finally, we introduce the prototype-attention modules with parameters θ^+ to the Transformer encoder, and for each source sentence we retrieve the prototype representations from table $\mathcal{P}$. We continue to optimize the whole parameter set $\{\theta^{(N)} \cup \theta^+\}$ to obtain a final model θ^*.

As mentioned earlier in the introduction, we compare the above one-pass method to a two-pass method, where we train the base model to convergence instead of for N iterations, before training

[1] https://scikit-learn.org/stable/modules/classes.html

a Proto-Transformer from scratch which uses clustered prototypes from the base model. The relative advantage is that the prototypes are taken from a fully trained NMT model, but the disadvantage is that the training time doubles. In addition, it remains a empirical question whether fully-trained NMT models give prototypes that are more suitable for guiding a final model. We discuss this in the Section 5.1.

It is worth noting that a limitation of K-Means is the pre-specification of cluster numbers, which can be different for different tokens in practice. Non-parametric clustering algorithms such as DBSCAN (Ester et al., 1996) can potentially solve the problem. However, their runtime complexity is typically $O(n^2)$ or worse, which prevents us from choosing them for large-scale NMT data.

4 Experiments

4.1 Experimental Settings

Datasets. We use CoGnition (Li et al., 2021) to systematically evaluate compositional generalization in MT scenarios, an English→Chinese (En→Zh) translation dataset. In consists of a training set of 196,246 sentence pairs, a valid set and a test set of 10,000 samples. In particular, it has a dedicated test set (i.e., CG-test set) consisting of 10,800 sentences containing novel compounds, so that the model's ability of compositional generalization can be measured by the ratio of compounds that are correctly translated. In addition, we choose 9 machine translation tasks from IWSLT, WMT and JRC-Acquis to verify the general effectiveness of our methods. The dataset statistics are shown in Appendix B.

Setup. We use Transformer (Vaswani et al., 2017) as our baseline models implemented using the Fairseq toolkit (Ott et al., 2019). For CoGnition and IWSLT, we use the Transformer iwslt_de_en setting while for the others we use the transformer_base setting. Following previous work, for IWSLT and JRC-Acquis En↔Es, we use beam search with width 5 and length penalty 0.6 for inference, whereas for the other datasets we set the beam width as 4. For CoGnition, N and k are set as 8 and 3 based on the validation set, and N and k for the other datasets are shown in Appendix B. Empirically, N is recommended to chosen from 10% to 25% of the total training epochs. We conduct categorization for all tokens except punctuation and low-frequent words, and the token on CoGniton is word and is subword on other datasets For reproducibility and stability, we train 6 models with seeds provided in (Li et al., 2021) for each method and report the average performance.

Evaluation Metrics. We use compound translation error rate (CTER; (Li et al., 2021)) to measure model performance on CoGnition. Specifically, *instance-level* CTER denotes the ratio of samples where the novel compounds are translated incorrectly, while *aggregate-level* CTER denotes the ratio of compounds that suffer at least one incorrect translation in corresponding contexts. To calculate CTER, Li et al. (2021) provide a manually collected dictionary for all the atoms based on the training set, since each word may have different translations. We also conduct human evaluation (Appendix A) as a supplement. We conduct evaluation using BLEU (Papineni et al., 2002) for the other datasets.

4.2 Baseline Methods

We compare our method with following baselines: (1) Transformer (Vaswani et al., 2017), which uses the same settings as Li et al. (2021); (2) Transformer-Small, a more compact Transformer model with 4 layers and 256 hidden size; (3) Transformer-Deep, which increases the number of encoder layers to 8 to take the same parameters as Proto-Transformer; (4) Transformer-Rela, which replaces absolute positional encoding with a relative one, an important component for compositional generalizaiton demonstrated in (Csordás et al., 2021; Ontañón et al., 2021); (5) Bow (Raunak et al., 2019): which uses bag-of-words loss to regularize the encoder based on the observation that the encoder representations are much weaker on unseen composition; and (6) SeqMix (Guo et al., 2020): which synthesizes examples by interpolating embeddings[2].

4.3 Results on CoGnition

The main results on CoGnition are shown in Table 1. Transformer gives instance-level and aggregate-level CTERs of 28.42% and 62.88%, respectively. In comparison, Proto-Transformer gives a score of 21.69% and 51.84%, respectively, with a significant improvement of 6.73% and 11.04% accordingly.

[2]The performance is the best when the hyper-parameters of Beta distribution are set to 1.0, which is consistent with (Guo et al., 2020).

Model	Compound Translation Error Rate (CTER) ↓			
	NP	VP	PP	Total
Transformer	24.74%/55.16%	24.82%/59.54%	35.71%/73.94%	28.42%/62.88%
Transformer-Deep	25.11%/54.11%	28.14%/60.43%	37.38%/75.24%	30.21%/63.26%
Transformer-Small	22.14%/47.21%	23.55%/53.60%	32.02%/69.64%	25.91%/56.82%
Bow	22.16%47.89%	24.83%/55.57%	35.04%/73.21%	27.34%/58.89%
SeqMix	24.52%/49.71%	26.88%/58.90%	34.36%/73.09%	28.59%/60.57%
Transformer-Rela	22.69%/51.20%	24.89%/57.17%	34.94%/71.73%	27.50%/60.03%
Proto-Transformer	**14.07%/36.45%**	**22.13%/50.90%**	**28.85%/68.15%**	**21.69%/51.84%**

Table 1: Compound translation error rate (CTER) on CoGnition. We report instance-level and aggregate-level CTERs, separated by "/". NP, VP, and PP denote noun phrases, verb phrases and positional phrases, respectively, three compound types in the compositional generalization test set (CG-test set).

Moreover, Proto-Transformer outperforms all baseline systems significantly, indicating that categorization on the contextualized representations is more beneficial to compositional generalization. We also calculate the BLEU scores, and Proto-Transformer and Transformer obtain 60.1 and 59.5, respectively. The BLEU scores are all relatively high since the sentences on the CG test set are similar to the sentences in training data except the novel compounds. For CoGnition, CTER is more accurate and suitable than BLEU because the translation dictionary processes multiple accurate translations of each compound better.

Since Proto-Transformer brings some extra parameters (approximately 6M), we investigate whether the performance improvement is derived from the increase of model parameters. As can be seen, Transformer-Deep performs poorly on the CG-test set, indicating that only increasing model capacity may be harmful since it leads to worse over-fitting to sequence-level distributions. Besides, the more compact model Transformer-Small yields better results than Transformer and Transformer-Deep but lags far behind Proto-Transformer. This shows that model size is useful but not sufficient for solving compositional generalization.

Proto-Transformer performs better than Bow, indicating that the encoder representations refined by categorization are more adequate than the regularization technique. Compared to SeqMix, the improvement of Proto-Transformer is more significant (2.31% vs 11.04% aggregate-level CTER). SeqMix reduces representation sparsity via linear interpolation in the input embedding space, and we conjecture that the stochastically synthetic samples may be unreasonable and harmful to model training. Relative positional embedding (Shaw et al., 2018) is demonstrated to be important for compositional generalization. Specifically, Transformer-Rela reduces CTERs by 0.92% and 2.85% but is inferior to ours, indicating that the prototypes bring more than positional information.

Dataset	Direction	BLEU	
		Transformer	Proto-TF
IWSLT	En⇒De	28.44	**28.96**
	En⇒It	28.24	**28.87**
	En⇒Vi	30.19	**30.96**
WMT	En⇒Ro	32.46	**33.37**
	Ro⇒En	32.49	**33.27**
	En⇒Fi	21.39	**22.11**
	En⇒De	27.95	**28.49**
JRC-Acquis	En⇒Es	60.90/60.32	**61.56/60.90**
	Es⇒En	63.21/62.59	**63.73/63.10**

Table 2: BLEU on commonly used MT datasets. For JRC-Acquis, we follow previous work to report results on both the valid and test sets, separated by "/". The performance of our model is significantly better than Transformer ($p < 0.05$) (Koehn, 2004).

4.4 Results on General MT Datasets

We conduct experiments on several general MT benchmarks, where the translation compound error rate (CTER) cannot be calculated directly. The performances on IWSLT, WMT and JRC-Acquis are presented in Table 2. Compared with Transformer, our model achieves consistent improvement (0.67 BLEU), demonstrating its effectiveness under general evaluation settings. Proto-Transformer outperforms Transformer by 0.91 and 0.78 BLEU scores on WMT'16 En→Ro and Ro→En respectively. For JRC-Acquis, our model achieves an average improvement of 0.62 BLEU score on En→Es and 0.56 BLEU score on Es→En. On the largest dataset WMT'16 En→De, our model also performs better than Transformer by 0.54 BLEU score. In addition, the large datasets possibly benefit from the additional parameters (about 6M). We run an experiment of a model with 8 encoder layers on EN-DE,

k	CTER	k	CTER
0	28.42%/62.88%	3	21.69%/51.84%
1	23.60%/54.33%	4	23.04%/52.70%
2	22.58%/53.19%	5	24.98%/54.71%

Table 3: CTERs against different number of prototypes on the CG-test set. "0" denotes to the Transformer baseline.

and it achieves 28.02 BLEU, inferior to our model with same number of parameters.

Note that the BLEU scores in the general datasets can reflect translation quality, but has several limitations in the measurement of generalization or robustness. For example, when a sentence is almost translated correctly but has few serious errors, it can achieve a high BLEU score but suffers severe semantic distortion. For instance, as shown in (Li et al., 2021), the model mis-translates "He became sick from eating all of the peanut butter on the ball" into "He became sick from eating all of the peanut butter on the field". With a minor mistake on the compound "on the ball", the model achieves a sentence-level BLEU of 61.4, but the full sentence meaning is largely affected.

5 Analysis

5.1 Prototypes

Effect of the Number of Prototypes. The number of prototypes k controls the granularity of context used to refine the representation learning, and is determined it based on the model loss on the development set. In this experiment, we investigate its influence on generalization performance. As shown in Table 3, incorporating prototypes can reduce the translation error caused by unseen compounds, and the model obtains the best generalization performance when k is 3. Intuitively, using too many prototypes dilutes the concentration effect, leading to overfitting again, while too few prototypes limits the expressiveness for polysemous words, e.g., one prototype represents a single sense.

Effect on Token-level Translation Consistency. We hypothesize that token alignments induced by Proto-Transformer are less ambiguous for tokens having settled meaning. To verify it, we approximately measure the number of distinct translations for 10 selected English tokens in the WMT Ro→En dataset, by counting the number of Romanian tokens that are aligned to the English tokens in the test set with the method of Chen et al. (2020b).

Tgt tokens	PT	TF	Src tokens
yesterday	1	3	**ieri**,psd,victor
limited	2	5	**limitata**,**limitat**,comert,apel, ...
six	1	6	**sase**,procente,saptamani,din, ...
agriculture	1	4	**agricultura**,plasa, ...
culture	2	5	**cultura**,**culturii**, profesion...
republic	1	3	**republica**,liderului, ...
simply	1	3	**simplu**,nu,individuala, ...
november	1	6	**noiembrie**,propaganda,2008, ...
tomorrow	1	3	**maine**,dimineata,amiaza, ...
saturday	1	3	**sambata**,seara,dimineata, ...

Table 4: Word translation variations induced from Transformer (TF) and Proto-Transformer (PT). The columns PT and TF denote the number of aligned Romanian tokens, and the bold characters display the tokens induced from Proto-Transformer.

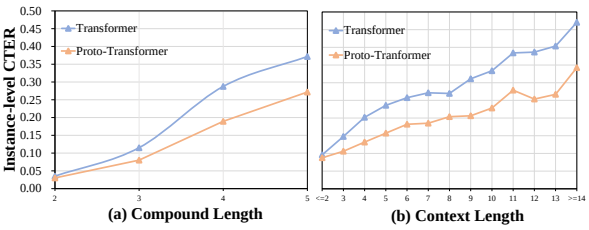

Figure 3: CTERs of Proto-Transformer and Transformer over different compound and context lengths.

As can be seen in Table 4, Proto-Transformer attends to much fewer token types than Transformer. This shows that Proto-Transformer's translations are less prone to context changes, which indicates the sparsity reduction of input sample space. The finding also partly explains the improvement on the Ro→En dataset. The above observations verify our intuition in the introduction that adding prototypes to the source representation enhances the model's knowledge on token-level translation consistency, which leads to better compositional generalizaiton.

It is noteworthy that increase translation consistency may increase translationese effects (Vanmassenhove et al., 2021). We argue that there can be a trade-off between the consistency modeling and translationese. Our approach does not force the model to generate highly consistent translation, but aims to alleviate the vulnerability to context changes by introducing several prototypes.

One-pass vs Two-pass. The possible advantage of the two-pass training procedure (Two-Pass) mentioned in Introduction is that the quality of the prototype representations generated by a converged model may be better. Specifically, Two-Pass achieves 24.36% instance-level and 55.69% aggregate-level CTER, outperforming Transformer

by 4.06% and 7.19%, respectively, but is noticeably inferior to Proto-Transformer (21.69% and 51.84%). This observation may relate to the observation that neural models learn more generalizable features in the early phase before memorization (Arpit et al., 2017; Stephenson et al., 2021). The one-pass approach can not only reduce the training cost (11,103 seconds vs 6,286 seconds) but also leverage more generalizable features to better improve models' compositional generalization. In addition, the prototypes can be initialized randomly and trained along with the model, and this variant obtains 26.24% and 57.94% CTERs. It preforms much worse than Proto-Transformer (21.69% and 51.84%) though they have identical number of parameters and architecture, indicating the effectiveness of explicit categorization during training.

5.2 Effects on Compositional Generalization

Composition Length. Longer compounds are harder to generalize as they contain richer semantic information (Li et al., 2021). We classify the test samples by compound length and context length, and calculate the instance-level CTER. In Figure 3, we can observe that the advantage of Proto-Transformer grows larger in generalizing longer compounds and context. In particular, Proto-Transformer gives a lower CTER by 12.80% over samples with context longer than 13 tokens. The underlying reason can be that longer compounds or contexts are more sparse in input space, and Proto-Transformer alleviates sparseness by putting token distributions into representations via prototypes.

Modifier. One challenging type of novel compounds in CoGnition is the postpositive modifier atom (MOD), which is constructed to enrich the information of its preceding word (e.g., *he liked* in the sentence *he bought the car he liked*). The difficulty of translating compounds with MOD lies in word reordering from English to Chinese. We divide the test samples into two groups according to compounds with or without MOD (Figure 4). Proto-Transformer demonstrates larger advantage in translating the compounds with MOD, showing its superiority in processing complex semantic composition. To further understand the effectiveness, we choose the token *liked*, a core part of the MOD atom *he liked*, and visualize its representations with t-SNE (van der Maaten and Hinton, 2008). In Figure 5, we can see that the representations of *he liked* serving as MOD are concentrated

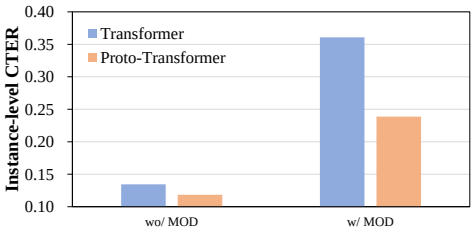

Figure 4: CTERs on compounds w/o and w/ MOD.

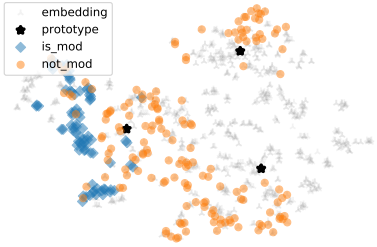

Figure 5: Visualization of contextualized embeddings (grey tri-down markers) and 3 prototypes (black stars) for the token *liked*. The blue diamonds and orange circles denote the case where *liked* appear in the context *he liked*, and the former one denotes the case where *he liked* serves as MOD.

at the leftmost prototype, while the representations of *he liked* not serving as MOD scatter on the other prototypes. Through representation categorization, Proto-Transformer finds a specialized prototype which abstracts the knowledge of *liked* serving as a part of MOD, and refers to the MOD prototype when processing compounds with MOD.

Case Study We present 3 source samples containing a novel compound *woke the silly boyfriend up* and 5 atoms, i.e., *woke*, *the*, *silly*, *boyfriend* and *up*, and their translations in Table 5. For all samples, correct translations should contain complete semantic meaning of the compound. Proto-Transformer correctly translates the compound along with different contexts across all samples, while Transformer suffers various mistakes. In the first sample, the translation of *something terrible* is omitted by Transformer. In the second sample, Transformer omits *silly boyfriend* and mistranslates *woke the silly boyfriend up* into *a silly bird woke up*. Similar mistake can be observed in the third example, where Transformer mistranslates *his dog woke the silly boyfriend up* into *his dog woke up*. Transformer overfits contexts and makes errors on unseen compositions, while our model is more stable.

Source	Transformer	Proto-Transformer
Yet to her dismay something terrible **woke the silly boyfriend up**.	然而，使她沮丧的是， 一个可怕的男友醒来了。 (Yet to her dismay **a silly boyfriend woke up**.)	但令她沮丧的是， 某种可怕的东西把那个傻男友吵醒了。 (Yet to her dismay something terrible **woke the silly boyfriend up**.)
Every morning , a bird **woke the silly boyfriend up** for work at 6 am.	每天早上6点， 一只愚蠢的鸟叫醒去上班。 (every morning , **a silly bird woke up** for work at 6 am.)	每天早上6点，一只鸟 叫醒那个愚蠢的男朋友去上班。 (Every morning , a bird **woke the silly boyfriend up** for work at 6 am.)
His dog **woke the silly boyfriend up** in the middle of the night.	他的狗半夜醒来了那个傻男友。 (His dog **woke up** in the middle of the night, **the silly boyfriend**.)	他的狗半夜把那个傻男友吵醒了。 (His dog **woke the silly boyfriend up** in the middle of the night.)

Table 5: Example translations. The bold characters denoting the novel compounds and corresponding translations.

6 Conclusion

We investigated a conceptually simple method for enhancing compositional generalizaiton of NMT models, proposing a two-stage training framework to fuse prototype representations into the encoding process of Transformer. Experiments on CoGnition show the effectiveness of our method on compositional generalization, and extensive results over 9 translation tasks verify the generality of our method. To our knowledge, we are the first to propose token-level categorization for NMT, achieving promising performance on both a large-scale compositional generalization dataset and general datasets.

Acknowledgements

The work is funded by the Zhejiang Province Key Project 2022SDXHDX0003. We would like to thank all of the anonymous reviewers and Jiali Zeng for the helpful comments. Yue Zhang is the corresponding author.

References

Ekin Akyürek, Afra Feyza Akyürek, and Jacob Andreas. 2021. Learning to recombine and resample data for compositional generalization. In *Proc. of ICLR*.

Jacob Andreas. 2020. Good-enough compositional data augmentation. In *Proc. of ACL*, pages 7556–7566.

Devansh Arpit, Stanislaw Jastrzebski, Nicolas Ballas, David Krueger, Emmanuel Bengio, Maxinder S. Kanwal, Tegan Maharaj, Asja Fischer, Aaron C. Courville, Yoshua Bengio, and Simon Lacoste-Julien. 2017. A closer look at memorization in deep networks. In *Proceedings of the 34th International Conference on Machine Learning, ICML 2017, Sydney, NSW, Australia, 6-11 August 2017*, volume 70 of *Proceedings of Machine Learning Research*, pages 233–242. PMLR.

Joris Baan, Jana Leible, Mitja Nikolaus, David Rau, Dennis Ulmer, Tim Baumgärtner, Dieuwke Hupkes, and Elia Bruni. 2019. On the realization of compositionality in neural networks. In *Proceedings of the 2019 ACL Workshop BlackboxNLP: Analyzing and Interpreting Neural Networks for NLP*, pages 127–137.

Ankur Bapna and Orhan Firat. 2019a. Non-parametric adaptation for neural machine translation. In *Proc. of NAACL-HLT*, pages 1921–1931.

Ankur Bapna and Orhan Firat. 2019b. Simple, scalable adaptation for neural machine translation. In *Proc. of EMNLP*, pages 1538–1548.

Yonatan Belinkov and Yonatan Bisk. 2018. Synthetic and natural noise both break neural machine translation. In *Proc. of ICLR*.

Rahma Chaabouni, Roberto Dessì, and Eugene Kharitonov. 2021. Can transformers jump around right in natural language? assessing performance transfer from SCAN. In *Proceedings of the Fourth BlackboxNLP Workshop on Analyzing and Interpreting Neural Networks for NLP*, pages 136–148, Punta Cana, Dominican Republic. Association for Computational Linguistics.

Xinyun Chen, Chen Liang, Adams Wei Yu, Dawn Song, and Denny Zhou. 2020a. Compositional generalization via neural-symbolic stack machines. In *Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020, December 6-12, 2020, virtual*.

Yun Chen, Yang Liu, Guanhua Chen, Xin Jiang, and Qun Liu. 2020b. Accurate word alignment induction from neural machine translation. In *In Proc. of EMNLP*, pages 566–576.

Yong Cheng, Lu Jiang, Wolfgang Macherey, and Jacob Eisenstein. 2020. AdvAug: Robust adversarial augmentation for neural machine translation. In *Proc. of ACL*, pages 5961–5970.

Yong Cheng, Zhaopeng Tu, Fandong Meng, Junjie Zhai, and Yang Liu. 2018. Towards robust neural machine translation. In *Proc. of ACL*, pages 1756–1766.

Noam Chomsky. 2009. *Syntactic structures*.

Henry Conklin, Bailin Wang, Kenny Smith, and Ivan Titov. 2021. Meta-learning to compositionally generalize. In *Proc. of ACL*, pages 3322–3335.

Róbert Csordás, Kazuki Irie, and Jürgen Schmidhuber. 2021. The devil is in the detail: Simple tricks improve systematic generalization of transformers. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing, EMNLP 2021, Virtual Event / Punta Cana, Dominican Republic, 7-11 November, 2021*, pages 619–634. Association for Computational Linguistics.

Verna Dankers, Elia Bruni, and Dieuwke Hupkes. 2021. The paradox of the compositionality of natural language: a neural machine translation case study. *CoRR*, abs/2108.05885.

Sergey Edunov, Myle Ott, Michael Auli, and David Grangier. 2018. Understanding back-translation at scale. In *Proc. of EMNLP*, pages 489–500.

Martin Ester, Hans-Peter Kriegel, Jörg Sander, and Xiaowei Xu. 1996. A density-based algorithm for discovering clusters in large spatial databases with noise. In *Proceedings of the Second International Conference on Knowledge Discovery and Data Mining (KDD-96), Portland, Oregon, USA*, pages 226–231.

Marzieh Fadaee and Christof Monz. 2020. The unreasonable volatility of neural machine translation models. In *Proceedings of the Fourth Workshop on Neural Generation and Translation*, pages 88–96.

Jonathan Gordon, David Lopez-Paz, Marco Baroni, and Diane Bouchacourt. 2020. Permutation equivariant models for compositional generalization in language. In *Proc. of ICLR*.

Jiatao Gu, Yong Wang, Kyunghyun Cho, and Victor O. K. Li. 2018. Search engine guided neural machine translation. In *Proc. of AAAI*, pages 5133–5140.

Demi Guo, Yoon Kim, and Alexander Rush. 2020. Sequence-level mixed sample data augmentation. In *Proc. of EMNLP*, pages 5547–5552.

Jonathan Herzig and Jonathan Berant. 2021. Span-based semantic parsing for compositional generalization. In *Proc. of ACL*, pages 908–921.

Theo M. V. Janssen. 1998. Algebraic translations, correctness and algebraic compiler construction. *Theor. Comput. Sci.*, 199(1-2):25–56.

Theo M. V. Janssen and Barbara H. Partee. 1997. Compositionality. In Johan van Benthem and Alice ter Meulen, editors, *Handbook of Logic and Language*, pages 417–473. North Holland / Elsevier.

Jungo Kasai, Nikolaos Pappas, Hao Peng, James Cross, and Noah A. Smith. 2021. Deep encoder, shallow decoder: Reevaluating non-autoregressive machine translation. In *9th International Conference on Learning Representations, ICLR 2021, Virtual Event, Austria, May 3-7, 2021*. OpenReview.net.

Daniel Keysers, Nathanael Schärli, Nathan Scales, Hylke Buisman, Daniel Furrer, Sergii Kashubin, Nikola Momchev, Danila Sinopalnikov, Lukasz Stafiniak, Tibor Tihon, Dmitry Tsarkov, Xiao Wang, Marc van Zee, and Olivier Bousquet. 2020a. Measuring compositional generalization: A comprehensive method on realistic data. In *Proc. of ICLR*.

Daniel Keysers, Nathanael Schärli, Nathan Scales, Hylke Buisman, Daniel Furrer, Sergii Kashubin, Nikola Momchev, Danila Sinopalnikov, Lukasz Stafiniak, Tibor Tihon, Dmitry Tsarkov, Xiao Wang, Marc van Zee, and Olivier Bousquet. 2020b. Measuring compositional generalization: A comprehensive method on realistic data. In *Proc. of ICLR*.

Urvashi Khandelwal, Angela Fan, Dan Jurafsky, Luke Zettlemoyer, and Mike Lewis. 2021. Nearest neighbor machine translation. In *9th International Conference on Learning Representations, ICLR 2021, Virtual Event, Austria, May 3-7, 2021*. OpenReview.net.

Philipp Koehn. 2004. Statistical significance tests for machine translation evaluation. In *Proc. of EMNLP*, pages 388–395.

Kris Korrel, Dieuwke Hupkes, Verna Dankers, and Elia Bruni. 2019. Transcoding compositionally: Using attention to find more generalizable solutions. In *Proceedings of the 2019 ACL Workshop BlackboxNLP: Analyzing and Interpreting Neural Networks for NLP*, pages 1–11.

Brenden M. Lake. 2019. Compositional generalization through meta sequence-to-sequence learning. In *Advances in Neural Information Processing Systems 32: Annual Conference on Neural Information Processing Systems 2019, NeurIPS 2019, December 8-14, 2019, Vancouver, BC, Canada*, pages 9788–9798.

Brenden M. Lake and Marco Baroni. 2018. Generalization without systematicity: On the compositional skills of sequence-to-sequence recurrent networks. In *Proc. of ICML*, Proceedings of Machine Learning Research, pages 2879–2888.

Jason Lee, Elman Mansimov, and Kyunghyun Cho. 2018. Deterministic non-autoregressive neural sequence modeling by iterative refinement. In *Proc. of EMNLP*, pages 1173–1182.

Yafu Li, Yongjing Yin, Yulong Chen, and Yue Zhang. 2021. On compositional generalization of neural machine translation. In *Proc. of ACL*, pages 4767–4780.

Yuanpeng Li, Liang Zhao, Jianyu Wang, and Joel Hestness. 2019. Compositional generalization for primitive substitutions. In *Proc. of EMNLP*, pages 4293–4302.

Evan Zheran Liu, Behzad Haghgoo, Annie S. Chen, Aditi Raghunathan, Pang Wei Koh, Shiori Sagawa, Percy Liang, and Chelsea Finn. 2021. Just train twice: Improving group robustness without training group information. In *Proc. of ICML*, Proceedings of Machine Learning Research, pages 6781–6792.

Stuart P. Lloyd. 1982. Least squares quantization in PCM. *IEEE Trans. Inf. Theory*, 28(2):129–136.

Richard Montague and Richmond H Thomason. 1975. Formal philosophy. selected papers of richard montague. *Erkenntnis*, (2).

Santiago Ontañón, Joshua Ainslie, Vaclav Cvicek, and Zachary Fisher. 2021. Making transformers solve compositional tasks. *CoRR*, abs/2108.04378.

Myle Ott, Sergey Edunov, Alexei Baevski, Angela Fan, Sam Gross, Nathan Ng, David Grangier, and Michael Auli. 2019. fairseq: A fast, extensible toolkit for sequence modeling. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics (Demonstrations)*, pages 48–53.

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. Bleu: a method for automatic evaluation of machine translation. In *Proc. of ACL*, pages 311–318.

Vikas Raunak, Vaibhav Kumar, and Florian Metze. 2019. On compositionality in neural machine translation. *CoRR*, abs/1911.01497.

Shiori Sagawa, Aditi Raghunathan, Pang Wei Koh, and Percy Liang. 2020. An investigation of why overparameterization exacerbates spurious correlations. In *Proc. of ICML*, Proceedings of Machine Learning Research, pages 8346–8356.

Rico Sennrich, Barry Haddow, and Alexandra Birch. 2016. Neural machine translation of rare words with subword units. In *Proc. of ACL*, pages 1715–1725.

Peter Shaw, Jakob Uszkoreit, and Ashish Vaswani. 2018. Self-attention with relative position representations. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT, New Orleans, Louisiana, USA, June 1-6, 2018, Volume 2 (Short Papers)*, pages 464–468. Association for Computational Linguistics.

David R. So, Quoc V. Le, and Chen Liang. 2019. The evolved transformer. In *Proc. of ICML*, Proceedings of Machine Learning Research, pages 5877–5886.

Cory Stephenson, Suchismita Padhy, Abhinav Ganesh, Yue Hui, Hanlin Tang, and SueYeon Chung. 2021. On the geometry of generalization and memorization in deep neural networks. In *9th International Conference on Learning Representations, ICLR 2021, Virtual Event, Austria, May 3-7, 2021*. OpenReview.net.

Ilya Sutskever, Oriol Vinyals, and Quoc V. Le. 2014. Sequence to sequence learning with neural networks. In *Advances in Neural Information Processing Systems 27: Annual Conference on Neural Information Processing Systems 2014, December 8-13 2014, Montreal, Quebec, Canada*, pages 3104–3112.

Laurens van der Maaten and Geoffrey E. Hinton. 2008. Visualizing data using t-sne. *Journal of Machine Learning Research*, 9:2579–2605.

Eva Vanmassenhove, Dimitar Shterionov, and Matthew Gwilliam. 2021. Machine translationese: Effects of algorithmic bias on linguistic complexity in machine translation. In *Proc. of EACL*, pages 2203–2213.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, December 4-9, 2017, Long Beach, CA, USA*, pages 5998–6008.

Hongfei Xu, Josef van Genabith, Qiuhui Liu, and Deyi Xiong. 2021a. Probing word translations in the transformer and trading decoder for encoder layers. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT 2021, Online, June 6-11, 2021*, pages 74–85. Association for Computational Linguistics.

Weiwen Xu, Ai Ti Aw, Yang Ding, Kui Wu, and Shafiq Joty. 2021b. Addressing the vulnerability of NMT in input perturbations. In *Proc. of NAACL*, pages 80–88.

Jiali Zeng, Yang Liu, Jinsong Su, Yubing Ge, Yaojie Lu, Yongjing Yin, and Jiebo Luo. 2019. Iterative dual domain adaptation for neural machine translation. In *Proc. of EMNLP*, pages 845–855.

A Human evaluation

Model	Adequacy	Fluency
Transformer	4.32	4.27
Proto-Transformer	4.51	4.54

Table 6: Human evaluation on adequacy and fluency.

We conduct the human evaluation for translations in terms of adequacy and fluency. We randomly sample 100 sentences from the CG-test set of CoGnition, and invite three annotators to evaluate the translation adequacy and fluency ranging from one to five. The five point scale for adequacy indicates how much of the meaning expressed in the reference translation is also expressed in a hypothesis translation: 5 = All, 4 = Most, 3 = Much, 2 = Little, and 1=None. The five point scale for fluency indicates how fluent the translation is: 5 = Flawless, 4 = Good, 3 = Non-native, 2 = Disfluent, and 1 = Incomprehensible.

The average of the scores from the three annotators is taken as the final score, and the results of the baseline and our model are shown in Table 6. Compared with Transformer, Proto-Transformer improves adequacy and fluency by 0.19 and 0.27, respectively. Proto-Transformer achieves more accurate translation of the novel compounds by alleviating the problem of compositional generalization, and possibly make the other part of the sentence to be translated better.

B Dataset Statistics and Hyper-parameters

We list the statistics and the introduced hyper-parameters of all the datasets in Table 7.

For IWSLT'14 English↔German (En↔De), IWSLT'14 English→Italian (En→It), and IWSLT'15 English→Vietnamese (En→Vi), we use Moses tokenizer[3] and apply joint BPE (Sennrich et al., 2016) with 10,000 merge operations. For WMT'16 English↔Romanian (En↔Ro), we use the processed data from Lee et al. (2018). For WMT'16 English→German (En→De) and WMT'17 English→Finnish (En→Fi), we apply joint BPE with 37,000 and 32,000 merge operations, respectively. For JRC-Acquis English↔Spanish (En↔Es) we use the datasets processed by Gu et al. (2018). For WMT'17 En-Fi, we use the concatenation of newstest2015, newsdev2015, newstest2016 and newstestB2016 as the development set, and the newstest2017 as the test set.

C Target-side Prototypes.

Target-side prototypes possibly contain more bilingual translation knowledge since the decoder processes target sentences based on source representations. We are interested in whether target-side prototypes can be used to enhance compositional generalization and/or further improve Proto-Transformer. To answer this question, we first extract prototypes from the decoder and incorporate them back to the decoder using the same mechanism of Proto-Transformer. The model can reduce CTERs to 25.93%/58.21%, largely inferior to using source-side prototypes (21.69%/51.84%). We also try to incorporate target-side prototypes to the decoder based on Proto-Transformer but it gives no noticeable improvement, achieving 21.65% and 51.71% CTERs, respectively. The underlying reason is connected a recent finding that translation already happens in the source encoding and the representations from encoders contain sufficient translation knowledge (Kasai et al., 2021; Xu et al., 2021a).

D Computational Cost

Our framework adds some computational overheads including extracting contextualized representations and conducting the clustering algorithm. The former only requires a single forward pass over the training set, merely amounting to a fraction of the cost of training for one epoch. Thanks to pytorch implementation of K-means algorithms which utilize GPU for faster matrix computations, the clustering is friendly for the large-scale datasets and it can be much faster with parallelization and more powerful hardware. Since the number of prototypes is a constant k, the complexity of prototype-attention is $O(kT)$, linear with respect to sequence length T.

[3] https://github.com/moses-smt/

	CoGnition	IWSLT'14		IWSLT'15		WMT'16		WMT'17	JRC-Acquis	
	En-Zh	En-De	En-It	En-Vi	En-Ro	En-Ro	En-De	En-Fi	En-Es	Es-En
#Train	196k	157k	175k	133k	608k	608k	4.5M	2.6M	679k	679k
#Valid	10k	7k	1k	1.6k	2k	2k	3k	9k	25k	25k
#Test	10k	7k	0.9k	1.3k	2k	2k	3k	3k	26k	26k
N	8	12	10	8	8	11	15	12	8	8
k	3	4	4	3	4	6	5	6	4	4

Table 7: Dataset statistics and hyper-parameters.

Adversarial Training on Disentangling Meaning and Language Representations for Unsupervised Quality Estimation

Yuto Kuroda
Ehime University
kuroda@ai.cs.ehime-u.ac.jp

Tomoyuki Kajiwara
Ehime University
kajiwara@cs.ehime-u.ac.jp

Yuki Arase
Osaka University
arase@ist.osaka-u.ac.jp

Takashi Ninomiya
Ehime University
ninomiya@cs.ehime-u.ac.jp

Abstract

We propose a method to distill language-agnostic meaning embeddings from multilingual sentence encoders for unsupervised quality estimation of machine translation. Our method facilitates that the meaning embeddings focus on semantics by adversarial training that attempts to eliminate language-specific information. Experimental results on unsupervised quality estimation reveal that our method achieved higher correlations with human evaluations.

1 Introduction

Quality Estimation (QE) is a task of estimating translation quality without reference sentences (Specia et al., 2018). Reference-based automatic evaluation methods, such as BLEU (Papineni et al., 2002) and BLEURT (Sellam and Parikh, 2020), have contributed to research and development of machine translation; however, end-users of machine translation systems unlikely have such reference translations. Hence, the development of QE methods that correlate well with human evaluation is practically important.

Supervised QE models (Ranasinghe et al., 2020; Fomicheva et al., 2020a; Nakamachi et al., 2020) based on pre-trained multilingual sentence encoders (Conneau et al., 2020; Feng et al., 2022) have been actively proposed in the QE competitions (Specia et al., 2020). However, these models require bilingual sentence pairs with manually labeled translation quality scores for fine-tuning. Creating such a QE dataset is expensive because it requires annotators who are fluent in both of source and target languages. Therefore, supervised QE models are limited to several major language pairs included in the competitions.

In contrast, unsupervised QE allows quality estimation *without* human-assessed machine translation outputs. Instead of the annotated outputs, unsupervised QE utilizes widely available parallel corpora. Multilingual sentence encoders (Artetxe and Schwenk, 2019a,b; Reimers and Gurevych, 2020; Conneau et al., 2020; Feng et al., 2022) are promising for developing unsupervised QE models; however, their sentence embeddings are dominated by language-specific information. Due to this characteristic, these sentence embeddings form clusters by language rather than by meaning, which hinders precise estimation of semantic similarity across languages (Tiyajamorn et al., 2021). To address this problem, DREAM (Tiyajamorn et al., 2021) disentangles sentence embeddings to meaning and language embeddings. It conducts self-supervised learning using parallel sentence pairs in bilingual corpora as positive examples and random pairs as negative examples; meaning embeddings of positive pairs should be close while those of negative pairs should be distant. However, DREAM lacks direct supervision to eliminate language-specific information from the meaning embeddings and its architecture is complex.

We improve DREAM by introducing an adversarial training that attempts to remove language-specific information from the meaning embeddings.[1] Our adversarial training eliminates the random pairs that DREAM needs, which results in a simpler architecture and lighter computational costs for training. Experimental results on the WMT20 QE task (Specia et al., 2020) revealed that our method achieved higher correlations with human scores than previous unsupervised QE models based on multilingual sentence encoders. Com-

[1]The source code for this paper is available at https://github.com/kuro961/MEAT.

Proceedings of the 29th International Conference on Computational Linguistics, pages 5240–5245
October 12–17, 2022.

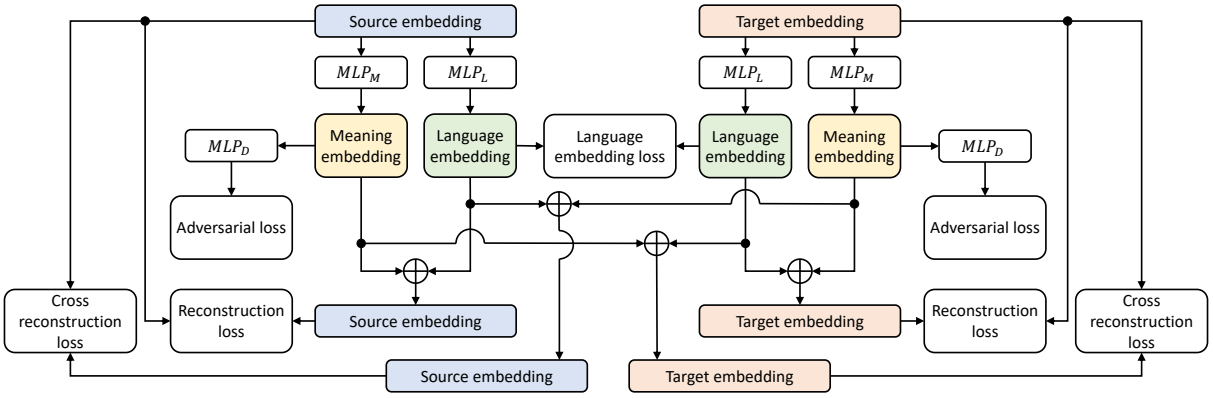

Figure 1: Multitask training for distilling meaning embeddings from multilingual sentence embeddings

pared to other approaches independent of multilingual sentence encoders (Fomicheva et al., 2020b; Thompson and Post, 2020), our method showed higher correlations in low-resource language pairs.

2 Proposed Method

Our model is an autoencoder comprising two multilayer perceptrons, MLP_M and MLP_L, trained with bilingual corpora as shown in Figure 1. The former is responsible for extracting meaning and the latter for extracting language-specific information. These outputs are summed to reconstruct the input sentence embedding. We train these MLPs using multilingual-multitask learning with the following four loss functions.

$$L = L_R + L_C + L_L + L_A \tag{1}$$

2.1 Reconstruction Loss L_R

L_R is the basis of the autoencoder training, which ensures that a meaning embedding $\hat{\boldsymbol{e}}_M \in \mathbb{R}^d$ and language embedding $\hat{\boldsymbol{e}}_L \in \mathbb{R}^d$ can reconstruct the input sentence embedding $\boldsymbol{e} \in \mathbb{R}^d$. Here, d is the dimension of the sentence embedding. We define reconstruction loss with cosine similarity[2] as:

$$L_R = 1 - \cos(\boldsymbol{e}, (\hat{\boldsymbol{e}}_M + \hat{\boldsymbol{e}}_L)). \tag{2}$$

2.2 Cross Reconstruction Loss L_C

Source and target sentences in Figure 1 are semantically equivalent as they are a parallel pair. Hence, their meaning embeddings should be interchangeable, for which we design a cross reconstruction loss L_C as:

$$L_C = 2 - \cos(\boldsymbol{s}, (\hat{\boldsymbol{s}}_L + \hat{\boldsymbol{t}}_M)) - \cos(\boldsymbol{t}, (\hat{\boldsymbol{t}}_L + \hat{\boldsymbol{s}}_M)). \tag{3}$$

The sentence embedding in the source language $\boldsymbol{s}$ should be reconstructed from its language embedding $\hat{\boldsymbol{s}}_L$ and the meaning embedding of the target language $\hat{\boldsymbol{t}}_M$. Similarly, the sentence embedding in the target language $\boldsymbol{t}$ should be reconstructed from its language embedding $\hat{\boldsymbol{t}}_L$ and the meaning embedding of the source language $\hat{\boldsymbol{s}}_M$.

2.3 Language Embedding Loss L_L

The source and target languages are different. To ensure that language embeddings of source and target are distinctive each other, we design a language embedding loss L_L as:

$$L_L = \max(0, \cos(\hat{\boldsymbol{s}}_L, \hat{\boldsymbol{t}}_L)). \tag{4}$$

2.4 Adversarial Loss L_A

We improve DREAM by giving direct supervision that eliminates language-specific information from the meaning embeddings. For this aim, we introduce an adversarial loss L_A that decrease language-identifiability from the meaning embeddings.

First, as an adversarial model that attempts to identify the language of the input sentence from its meaning embedding, we use the following multiclass classifier MLP_D:

$$\hat{\boldsymbol{y}} = \mathrm{softmax}(\mathrm{MLP}_D(\hat{\boldsymbol{e}}_M)). \tag{5}$$

MLP_D is trained using the cross-entropy loss:

$$L_D = -\sum_j \boldsymbol{y}_j \log \hat{\boldsymbol{y}}_j. \tag{6}$$

Note that Equation (6) is the loss function for training MLP_D, and is not included in Equation (1) for training MLP_M and MLP_L.

[2]This constraint does not strictly reconstruct the input embedding, because $\cos(\cdot)$ does not take into account the vector norm. We empirically employed $\cos(\cdot)$ for its higher performance than MSE used by Tiyajamorn et al. (2021).

With the adversarial model, we define L_A that supervises MLP_M to derive meaning embeddings from which languages are unidentifiable. Specifically, L_A makes the distribution of $\hat{\boldsymbol{y}}$ close to a uniform distribution:

$$L_A = -\frac{1}{N}\sum_j \log \hat{\boldsymbol{y}}_j, \tag{7}$$

where N is the number of language types in the training data. Adversarial training is performed simultaneously with the model training; Equation (6) trains the adversarial model to achieve higher language identifiability from the meaning embeddings while Equation (7) makes the meaning embeddings less language-identifiable.

2.5 Application to QE

Once our model is trained, we can use MLP_M to disentangle meaning embeddings from sentence representations generated by multilingual sentence encoders. We compute a QE score by a cosine similarity between meaning embeddings of source sentence s and translation output t:

$$\cos(\hat{\boldsymbol{s}}_M, \hat{\boldsymbol{t}}_M). \tag{8}$$

3 Evaluation

We evaluated the effectiveness of the proposed method in an unsupervised QE task.

3.1 Setting

Dataset Following the previous work (Tiyajamorn et al., 2021), we used six language pairs included in the WMT20 QE task[3] (Specia et al., 2020). For each language pair, the test set consists of 1k pairs of source and machine-translated output sentences manually labeled with a translation quality score. The evaluation metric is Pearson correlation coefficients between these human scores and model predictions.

We trained our model on the publicly available bilingual corpora that were used to train the target machine translation systems (Ott et al., 2019). We used bilingual corpora of 1M sentence pairs for high-resource (en-de and en-zh), 200k for medium-resource (ro-en and et-en), and 50k for low-resource (ne-en and si-en) language pairs.[4]

Model All the MLPs in our model are single-layer feedforward networks. As a multilingual sentence encoder to disentangle meaning embeddings, we used LaBSE[5] (Feng et al., 2022) with HuggingFace Transformers (Wolf et al., 2020), which achieved the best performance in DREAM (Tiyajamorn et al., 2021). We used a [CLS] embedding as a sentence embedding. The parameters of LaBSE were frozen and only those of MLPs in our method were updated during training using the parallel corpora.

We used a batch size of 512, Adam (Kingma and Ba, 2015) optimizer with a learning rate of $1e-5$. We employed early stopping for training with a patience of 10 using a validation loss of Equation (1). The validation set was created by randomly sub-sampling 10% of the training set.

Comparison We compared our method to DREAM.[6] Besides, we compared to unsupervised QE methods that compute cosine similarities of original sentence embeddings of LaBSE, LASER[7] (Artetxe and Schwenk, 2019a,b), mSBERT[8] (Reimers and Gurevych, 2020), and BERTScore[9] (Zhang et al., 2020). Following the pre-training setup of each model, max-pooling of final layer outputs of the BiLSTM was used as a sentence embedding on LASER, and similarly, mean-pooling was used on mSBERT.

We also compared to other approaches that do not depend on multilingual sentence encoders as a reference. D-TP (Fomicheva et al., 2020b) and Prism (Thompson and Post, 2020) are unsupervised QE methods based on an encoder-decoder model. Predictor-Estimator[10] (Kim et al., 2017; Kepler et al., 2019) is a supervised method employed as the baseline for the WMT20 QE task.

3.2 Result

The first set of rows in Table 1 indicates the performance of the original sentence embeddings from LaBSE and their meaning embeddings derived by DREAM and our method. While both DREAM

[3] https://github.com/facebookresearch/mlqe

[4] We sampled the same numbers of parallel sentences as in Tiyajamorn et al. (2021) from http://www.statmt.org/wmt20/quality-estimation-task.html for fair comparison.

[5] https://huggingface.co/sentence-transformers/LaBSE

[6] https://github.com/nattaptiy/qe_disentangled

[7] https://github.com/facebookresearch/LASER

[8] https://huggingface.co/sentence-transformers/stsb-xlm-r-multilingual

[9] https://github.com/Tiiiger/bert_score

[10] https://github.com/Unbabel/OpenKiwi

	High Resource		Medium Resource		Low Resource		
Model	en-de	en-zh	ro-en	et-en	ne-en	si-en	Avg.
LaBSE	0.084	0.036	0.705	0.550	0.545	0.455	0.396
DREAM	0.196	0.197	0.724‡	0.578	**0.636**	0.568	0.483
Ours	**0.215**‡	0.222‡	0.717	**0.587**†	0.634	**0.571**	**0.491**
LASER	0.105	0.106	0.705	0.463	-	0.325	0.341
mSBERT	0.130	**0.287**	**0.766**	0.512	0.467	0.418	0.430
BERTScore	0.134	0.143	0.746	0.568	0.562	0.549	0.450
D-TP	0.259	0.321	0.693	0.642	0.558	0.460	0.489
Prism	0.464	0.303	0.829	0.694	-	-	0.573
Predictor-Estimator	0.145	0.190	0.685	0.477	0.386	0.374	0.376

Table 1: Pearson correlation coefficients measured on WMT20 QE task (Superscripts of ‡ and † indicate statistically significant differences of $p < 0.01$ and 0.05, respectively, compared to DREAM.)

and our method consistently outperformed LaBSE, our method achieved larger improvements. These results confirm that our method with adversarial training further enhanced the ability of meaning embedding distillation of DREAM.

The second set of rows shows the performance of previous unsupervised methods based on multilingual sentence encoders. Our method outperformed these methods on most language pairs. Particularly, it showed higher scores on low-resource language pairs, and achieved the highest correlation with human scores on average for all language pairs. It is notable that our method outperformed mSBERT on four out of six language pairs, which had sentence similarity estimation in its pre-training.

The last set of rows shows the performance of other QE models independent of multilingual sentence encoders. Our method achieved higher scores than the supervised QE model, Predictor-Estimator, for all language pairs.

D-TP and Prism achieved higher scores than ours in high-resource language pairs, but our method outperformed them in low-resource language pairs. Although D-TP assumes that users can access to the parameters of a machine translatioin model for QE, such a situation is practically limited because in general, machine translation systems are black-box to end-users (*e.g.* online machine translation services). Prism requires a large-scale bilingual corpus for training its encoder and decoder from scratch, which restricts its applicability to low-resource language pairs.

	L_R	L_C	L_L	L_A	Avg.
(a)	✓				0.393
(b)	✓	✓			0.086
(c)	✓		✓		0.075
(d)	✓			✓	0.427
(e)	✓	✓	✓		0.439
(f)	✓	✓		✓	0.297
(g)	✓		✓	✓	0.482
(h)		✓	✓	✓	0.488

Table 2: Pearson correlation coefficients in ablation

3.3 Ablation Study

Table 2 shows the results of the ablation study. The upper rows show the performance when L_R is combined with one other loss function, and the lower rows show the performance when each loss function is excluded from the proposed method, measured on WMT20 QE task.

The first set of rows (rows (a) to (d)) shows that adversarial loss L_A has the largest contribution on its own. In contrast, cross reconstruction loss L_C and language embedding loss L_L largely deteriorated the performance of L_R. However, interestingly, the second set of rows (rows (e) to (h)) show that the performance drop is largest when L_L is removed. These results indicate that L_L is crucial when combining different loss functions. We presume that L_L has an effect that meaning and language information are separated into the corresponding embeddings. In other words, it prevents that meaning information leaks to language embed-

dings. In summary, these analyses revealed two loss functions that most contribute to the performance of the proposed method: L_L and L_A, and these should be used together.

4 Summary and Future Work

We introduced adversarial training to disentangle meaning embeddings from sentence representations of multilingual sentence encoders for unsupervised QE. Our method consistently improves the performance of a state-of-the-art multilingual sentence encoder.

Our future work includes exploring ways to utilize language-specific embeddings for QE in terms of fluency of sentences. Combined with the present method of assessing the adequacy of sentences, a better QE may be achieved. We will also apply our method for disentangling styles and meanings of sentences for the style-transfer research.

Acknowledgments

This work was supported by JSPS KAKENHI Grant Number JP20K19861. This research was also obtained from the commissioned research (No. 225) by National Institute of Information and Communications Technology (NICT), Japan.

References

Mikel Artetxe and Holger Schwenk. 2019a. Margin-based Parallel Corpus Mining with Multilingual Sentence Embeddings. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 3197–3203.

Mikel Artetxe and Holger Schwenk. 2019b. Massively Multilingual Sentence Embeddings for Zero-Shot Cross-Lingual Transfer and Beyond. *Transactions of the Association for Computational Linguistics*, 7:597–610.

Alexis Conneau, Kartikay Khandelwal, Naman Goyal, Vishrav Chaudhary, Guillaume Wenzek, Francisco Guzmán, Edouard Grave, Myle Ott, Luke Zettlemoyer, and Veselin Stoyanov. 2020. Unsupervised Cross-lingual Representation Learning at Scale. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 8440–8451.

Fangxiaoyu Feng, Yinfei Yang, Daniel Cer, Naveen Arivazhagan, and Wei Wang. 2022. Language-agnostic BERT sentence embedding. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 878–891.

Marina Fomicheva, Shuo Sun, Lisa Yankovskaya, Frédéric Blain, Vishrav Chaudhary, Mark Fishel, Francisco Guzmán, and Lucia Specia. 2020a. BERGAMOT-LATTE Submissions for the WMT20 Quality Estimation Shared Task. In *Proceedings of the Fifth Conference on Machine Translation*, pages 1010–1017.

Marina Fomicheva, Shuo Sun, Lisa Yankovskaya, Frédéric Blain, Francisco Guzmán, Mark Fishel, Nikolaos Aletras, Vishrav Chaudhary, and Lucia Specia. 2020b. Unsupervised Quality Estimation for Neural Machine Translation. *Transactions of the Association for Computational Linguistics*, 8:539–555.

Fabio Kepler, Jonay Trénous, Marcos Treviso, Miguel Vera, and André F. T. Martins. 2019. OpenKiwi: An Open Source Framework for Quality Estimation. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics: System Demonstrations*, pages 117–122.

Hyun Kim, Hun-Young Jung, Hongseok Kwon, Jong-Hyeok Lee, and Seung-Hoon Na. 2017. Predictor-Estimator: Neural Quality Estimation Based on Target Word Prediction for Machine Translation. *ACM Transactions on Asian and Low-Resource Language Information Processing*, 17(1):1–22.

Diederik P. Kingma and Jimmy Ba. 2015. Adam: A Method for Stochastic Optimization. In *Proceedings of the 3rd International Conference on Learning Representations*.

Akifumi Nakamachi, Hiroki Shimanaka, Tomoyuki Kajiwara, and Mamoru Komachi. 2020. TMUOU Submission for WMT20 Quality Estimation Shared Task. In *Proceedings of the Fifth Conference on Machine Translation*, pages 1037–1041.

Myle Ott, Sergey Edunov, Alexei Baevski, Angela Fan, Sam Gross, Nathan Ng, David Grangier, and Michael Auli. 2019. fairseq: A Fast, Extensible Toolkit for Sequence Modeling. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics (Demonstrations)*, pages 48–53.

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. BLEU: A Method for Automatic Evaluation of Machine Translation. In *Proceedings of the 40th Annual Meeting on Association for Computational Linguistics*, page 311–318.

Tharindu Ranasinghe, Constantin Orasan, and Ruslan Mitkov. 2020. TransQuest: Translation Quality Estimation with Cross-lingual Transformers. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 5070–5081.

Nils Reimers and Iryna Gurevych. 2020. Making Monolingual Sentence Embeddings Multilingual using Knowledge Distillation. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 4512–4525.

Dipanjan Sellam, Thibault and. Das and Ankur Parikh. 2020. BLEURT: Learning Robust Metrics for Text Generation. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7881–7892.

Lucia Specia, Frédéric Blain, Marina Fomicheva, Erick Fonseca, Vishrav Chaudhary, Francisco Guzmán, and André F. T. Martins. 2020. Findings of the WMT 2020 Shared Task on Quality Estimation. In *Proceedings of the Fifth Conference on Machine Translation*, pages 743–764.

Lucia Specia, Carolina Scarton, and Gustavo Henrique Paetzold. 2018. Quality Estimation for Machine Translation. *Synthesis Lectures on Human Language Technologies*, 11(1):1–162.

Brian Thompson and Matt Post. 2020. Automatic Machine Translation Evaluation in Many Languages via Zero-Shot Paraphrasing. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 90–121.

Nattapong Tiyajamorn, Tomoyuki Kajiwara, Yuki Arase, and Makoto Onizuka. 2021. Language-agnostic Representation from Multilingual Sentence Encoders for Cross-lingual Similarity Estimation. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 7764–7774.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Remi Louf, Morgan Funtowicz, Joe Davison, Sam Shleifer, Patrick von Platen, Clara Ma, Yacine Jernite, Julien Plu, Canwen Xu, Teven Le Scao, Sylvain Gugger, Mariama Drame, Quentin Lhoest, and Alexander Rush. 2020. Transformers: State-of-the-Art Natural Language Processing. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 38–45.

Tianyi Zhang, Varsha Kishore, Felix Wu, Kilian Q. Weinberger, and Yoav Artzi. 2020. BERTScore: Evaluating Text Generation with BERT. In *Proceedings of the 8th International Conference on Learning Representations*, pages 1–43.

Alleviating the Inequality of Attention Heads for Neural Machine Translation

Zewei Sun[1,*], **Shujian Huang**[2,3], **Xin-Yu Dai**[2], **Jiajun Chen**[2]
[1] ByteDance AI Lab
[2] State Key Laboratory for Novel Software Technology, Nanjing University
[3] Peng Cheng Laboratory, China
sunzewei.v@bytedance.com, {huangsj,daixinyu,chenjj}@nju.edu.cn

Abstract

Recent studies show that the attention heads in Transformer are not equal (Voita et al., 2019; Michel et al., 2019). We relate this phenomenon to the imbalance training of multi-head attention and the model dependence on specific heads. To tackle this problem, we propose a simple masking method: *HeadMask*, in two specific ways. Experiments show that translation improvements are achieved on multiple language pairs. Subsequent empirical analyses also support our assumption and confirm the effectiveness of the method.

1 Introduction

Recently, more and more novel network structures of neural machine translation(NMT) have been proposed (Bahdanau et al., 2015; Barone et al., 2017; Gehring et al., 2017; Vaswani et al., 2017), among which Transformer (Vaswani et al., 2017) achieves the best results. One important difference between Transformer and other translation models is its multi-head attention mechanism.

Some interesting phenomena of the attention heads are discovered recently. Voita et al. (2019) find that only a small subset of heads appear to be important for the translation task and vast majority of heads can be removed without seriously affecting performance. Michel et al. (2019) also find that several heads can be removed from trained transformer models without statistically significant degradation in test performance. It turns out that not all heads are equally important.

We speculate that this can be attributed to the imbalanced training of multi-head attention, as some heads are not trained adequately and contribute little to the model. However, this can be turned into the bottleneck for the whole model. For an analogy, if a soccer player gets used to using the right foot and spares more training opportunities for it, it will be stronger and stronger. As a result, the right foot is further relied on, while the left foot receives less training and gradually turns into the limitation.

In this paper, we firstly empirically confirm the inequality in multi-head attention. Then a new training method with two variants is proposed to avoid the bottleneck and improve the translation performance. Further analyses are also made to verify the assumption.

2 Head Inequality

Following Michel et al. (2019), we define the importance of an attention head h as

$$I_h = \mathbb{E}_{x\sim X}\left|\frac{\partial \mathcal{L}(x)}{\partial \xi_h}\right| \quad (1)$$

where $\mathcal{L}(x)$ is the loss on sample x and ξ is the head mask variable with values in {0, 1}. Intuitively, if $head_h$ is important, switching ξ_h will have a significant effect on the loss. Applying the chain rule yields the final expression for I_h:

$$I_h = \mathbb{E}_{x\sim X}\left|\mathrm{Att}_h(x)^T \frac{\partial \mathcal{L}(x)}{\partial \mathrm{Att}_h(x)}\right| \quad (2)$$

This is equivalent to the Taylor expansion method from Molchanov et al. (2017). In Transformer base (Vaswani et al., 2017), there are 3 types of attention (encoder self attention, decoder self attention, encoder-decoder attention) with 6 layers per type and 8 heads per layer. Therefore, it amounts to 144 heads. We divide them into 8 groups with 18 heads (12.5%) each group according to their importance I_h, among which, 1-18 are the most important and so on.

We then mask different groups of the heads. As is shown in Figure 1, masking a group of unimportant heads has little effect on the translation quality while masking important heads leads to a significant drop of performance. Surprisingly, almost half of the heads are not important, as it makes almost no difference whether they are masked or not.

* Work was done while at NJU

Proceedings of the 29th International Conference on Computational Linguistics, pages 5246–5250
October 12–17, 2022.

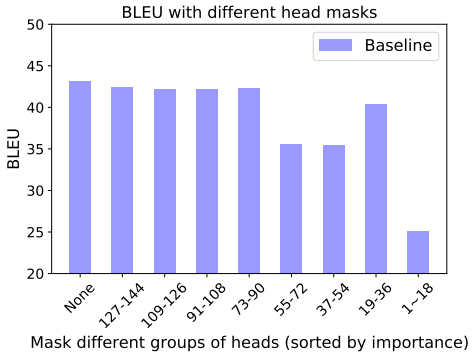

Figure 1: Mask the heads in the same group. Important ones matter much more than unimportant ones.

We also gradually masking more heads group by group in the ascending order and descending order, respectively. As is shown in Figure 2, the line starting with unimportant heads drops much slower than the one starting with important ones. It fully illustrates the inequality of different heads.

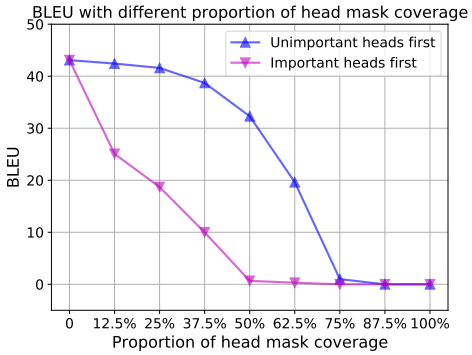

Figure 2: Mask all heads in the ascending order and descending order. The drop curves differ greatly.

Figure 1 and Figure 2 further demonstrates the inequality of the importance of attention heads. A simple assumption for explanation is that some heads coincidentally get more updating opportunities in the early stage, which makes the model learning to depend on them gradually. As a result, the model increasingly draws a strong connection with these specific heads while this local dependence prevents the rest attention heads from adequate training and restricts the overall capacity.

3 HeadMask

Since the problem refers to the unfair training of attention heads, it is natural for us to explicitly balance the training chances. We propose a simple method: *HeadMask*, which masks certain heads during training in two specific ways.

3.1 Mask Randomly

The first one is randomly picking heads and masking them in each batch. It ensures every head gets relatively equal opportunities of training and avoid partial dependence, as is shown in Algorithm 1. For the soccer analogy, it is like training the feet randomly, making both receive the same amount of practice.

Algorithm 1 HeadMask: Mask Randomly

Input: q, k, v for attention, number of masks n
Output: masked context

for batch in datasets **do**
  heads = random.sample(all_heads, n)
  for head in heads **do**
    $\xi_{head} = 0$
  end for
  context = attn(ξ)
end for

3.2 Mask Important Ones

The second one is masking the most important heads. By forcing the model neglects important heads, we hope more training chances are assigned to weaker heads. For the soccer analogy, it means training the left foot more if the right foot dominates. And once reversed, train contrarily. Its main idea is about suppressing addicted training. Specifically, the network firstly proceeds feed-forward calculation and back propagation without updating parameters to yield the importance of heads. And after picking the most important heads by sorting, mask them. During training, we only use the rest part of networks to reach the final loss and update parameters, as is shown in algorithm 2.

Algorithm 2 HeadMask: Mask Important Ones

Input: q, k, v for attention, number of masks n
Output: masked context

for batch in datasets **do**
  calculate $\mathcal{L}$ by feed-forward
  back propagation without updating params
  calculate importance of all heads I
  heads = argmax$_n(I)$
  for head in heads **do**
    $\xi_{head} = 0$
  end for
  context = attn(ξ)
  calculate $\mathcal{L}$ by feed-forward
  back propagation and update params
end for

4 Experiments

4.1 Datasets and Systems

We conduct experiments on four datasets, including three low-resource ones (less than 1 million). We use BPE (Sennrich et al., 2016) for Zh-En (Zheng et al., 2018) and Ro-En, adopt the preprocessed versions from Luong and Manning (2015) as well as the settings of Huang et al. (2017) for Vi-En, and follow the joint-BPE settings of Sennrich et al. (2017) for Tr-EN. More information is in Table 1.

Datasets	Scale	Dev	Test
NIST Zh-En	1.34M	MT03	MT04/05/06
WMT16 Ro-En	608K	newstest2015	newstest2016
IWSLT15 Vi-En	133K	tst2012	tst2013
WMT17 Tr-En	207K	newstest2016	newstest2017

Table 1: The information of our datasets

We follow Transformer base setting (Vaswani et al., 2017; Sun et al., 2022). Parameters are optimized by Adam (Kingma and Ba, 2015), with $\beta_1 = 0.9$, $\beta_2 = 0.98$, and $\epsilon = 10^{-9}$. The learning rate is scheduled according to Vaswani et al. (2017), with $warmup\_steps = 4000$. Label smoothing (Szegedy et al., 2016) of value=0.1 and dropout (Srivastava et al., 2014) of value=0.1 are also adopted.

Comparison We compare the baseline with masking randomly (Random-N) and masking important ones (Impt-N), where N is the mask number. In this paper, we mainly employ $N = 18(12.5\%)$.

4.2 Results

As is shown in Table 2,3,4, except for Vi-En experiments, Impt-18 yields enhancement over all language directions and reach the best result on the experiment of Ro → En. And Random-18 obtains steady improvements over all pairs and is obviously better than Impt-18. It seems the aggressive masking strategy at important heads can be too harsh and reversely restrict the model. And the random method is more expert in building a rational training pattern. In conclusion, reducing the unbalanced training among attention heads can effectively improve the translation quality.

Test sets	MT04	MT05	MT06
Baseline	46.62	43.46	43.09
Impt-18	46.94 (+0.28)	44.19 (+0.73)	43.16 (+0.07)
Random-18	**47.04 (+0.42)**	**44.33 (+0.87)**	**43.88 (+0.79)**

Table 2: Results on Experiments of Zh → En

Directions	Ro → En	Vi → En	Tr → En
Baseline	32.17	26.49	17.29
Impt-18	**32.95 (+0.78)**	26.36 (-0.13)	17.48 (+0.19)
Random-18	32.85 (+0.68)	**26.85 (+0.36)**	**17.56 (+0.27)**

Table 3: Results on Experiments of Ro/Vi/Tr → En

Directions	En → Ro	En → Vi	En → Tr
Baseline	31.98	28.07	15.74
Impt-18	32.47 (+0.49)	28.06 (-0.01)	16.10 (+0.36)
Random-18	**32.64 (+0.66)**	**28.46 (+0.39)**	**16.16 (+0.42)**

Table 4: Results on Experiments of En → Ro/Vi/Tr

4.3 Statistical Analysis

4.3.1 Flatter Distribution

To evaluate the adjusted training of heads, we check the distribution of head importance. As is shown in Figure 3, our methods make the importance distribution flatter. And the overall variance and mean are also calculated, as is shown in Table 5,6. Compared with Baseline, Impt-18 and Random-18 significantly reduce the variance of attention heads, achieving the goal of more equal training. And the mean also decreases, which proves the decline of dependence on every individual head. More specifically, Impt-18 can better resolve the imbalance, for it well prevent the emergence of "super" heads.

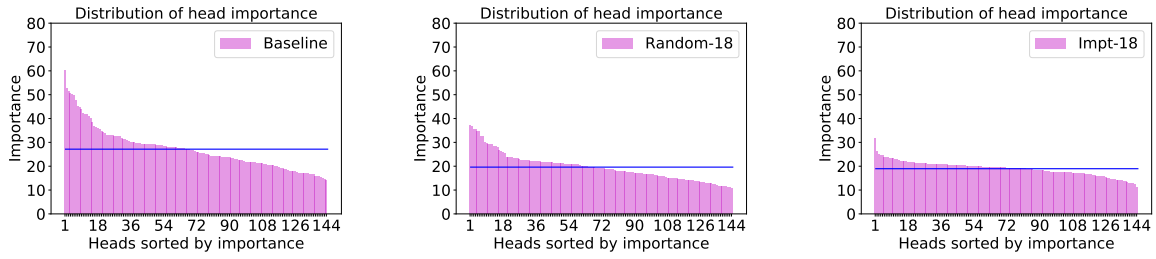

(a) Baseline (b) Random-18 (c) Impt-18

Figure 3: Distribution of importance of attention heads. Our methods make the whole distribution much flatter.

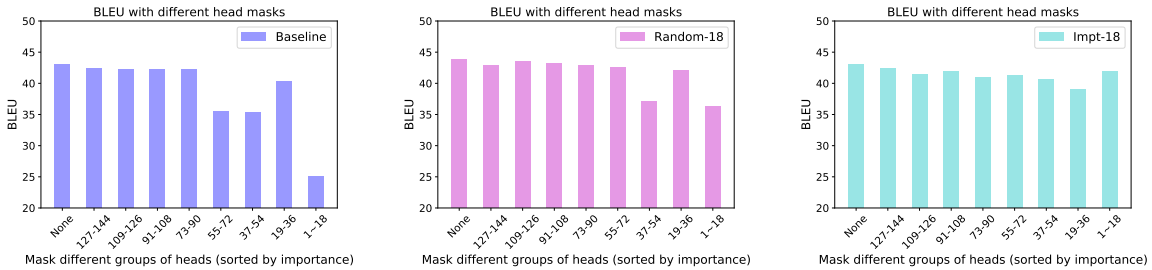

(a) Baseline (b) Random-18 (c) Impt-18

Figure 4: Our methods significantly maintain the performance even if the important heads are masked.

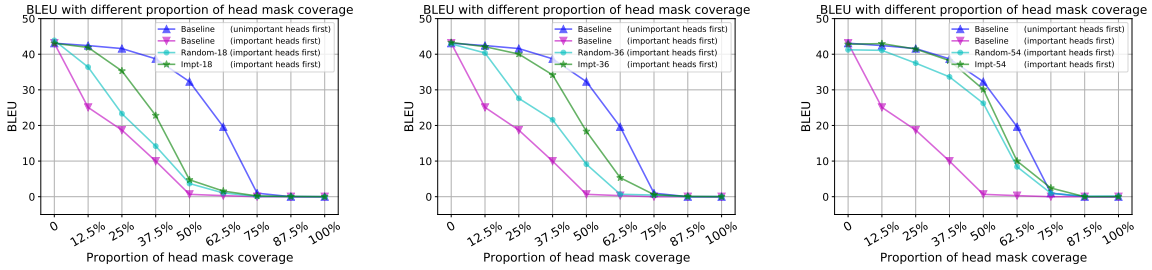

(a) Mask 18 heads in training (b) Mask 36 heads in training (c) Mask 54 heads in training

Figure 5: As the number of masked heads grows, the drop curves starting with important heads are moving up.

Directions	Zh2En	Ro2En	Vi2En	Tr2En
Baseline	77.28	552.93	100.73	1767.70
Random-18	33.21	255.98	48.28	900.70
Impt-18	9.13	72.73	14.13	188.87

Table 5: Our methods greatly reduce the **Variance** of the head importance, illustrating the improved equality of heads.

Directions	Zh2En	Ro2En	Vi2En	Tr2En
Baseline	27.15	47.18	17.96	83.79
Random-18	19.62	39.96	14.86	74.05
Impt-18	18.95	37.30	18.96	85.12

Table 6: Our methods reduce the **Mean** of the head importance, illustrating the lessened dependence on each head.

4.3.2 Weaker Dependence

We repeat the experiments of masking different groups of heads. As is shown in Figure 4, the translation quality is still maintained even if important heads are masked, proving the dependence on them has decreased. And Impt-18 performs more steadily since it is accustomed to such situations.

4.3.3 More Robust Models

We also repeat the experiments of masking all heads, as is shown in Figure 5. The two middle lines originally lie in the same place as the bottom one. As the number of masked heads in training (N) grows, they gradually move up and approach the top line where unimportant heads are masked first. It shows our methods make the model rely less on the important heads and become more robust.

5 Related Works

Recently, many analytical works about multi-head attention come out (Raganato and Tiedemann, 2018; Tang et al., 2018; Voita et al., 2019; Michel et al., 2019; Sun et al., 2020; Behnke and Heafield, 2020). And for the inequality of the networks, some studies focus on the model level (Frankle and Carbin, 2019; Sun et al., 2021), layer level (Zhang et al., 2019), and neuron level (Bau et al., 2019). For the mask algorithm, there are also works on the layer level (Fan et al., 2020), word level (Provilkov et al., 2019), and neuron level (Srivastava et al., 2014). Different from them, we mainly study the attention level and conduct a statistical analysis.

6 Conclusion

In this paper, we empirically validate the inequality of attention heads in Transformer and come up with an assumption of imbalanced training. Correspondingly, we propose a specific method in two ways to resolve the issue. Experiments show the improvements on multiple language pairs. And detailed analysis shows the alleviation of the problem and the effectiveness of our techniques.

7 Acknowledgements

We would like to thank the anonymous reviewers for their insightful comments. Shujian Huang is the corresponding author. This work is supported by National Science Foundation of China (No. 6217020152).

References

Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. 2015. Neural machine translation by jointly learning to align and translate. In *ICLR*.

Antonio Valerio Miceli Barone, Jindrich Helcl, Rico Sennrich, Barry Haddow, and Alexandra Birch. 2017. Deep architectures for neural machine translation. In *WMT*.

Anthony Bau, Yonatan Belinkov, Hassan Sajjad, Nadir Durrani, Fahim Dalvi, and James Glass. 2019. Identifying and controlling important neurons in neural machine translation. In *ICLR*.

Maximiliana Behnke and Kenneth Heafield. 2020. Losing heads in the lottery: Pruning transformer attention in neural machine translation. In *EMNLP*.

Angela Fan, Edouard Grave, and Armand Joulin. 2020. Reducing transformer depth on demand with structured dropout. In *ICLR*.

Jonathan Frankle and Michael Carbin. 2019. The lottery ticket hypothesis: Finding sparse, trainable neural networks. In *ICLR*.

Jonas Gehring, Michael Auli, David Grangier, Denis Yarats, and Yann Dauphin. 2017. Convolutional sequence to sequence learning. In *ICML*.

Po-Sen Huang, Chong Wang, Dengyong Zhou, and Li Deng. 2017. Neural phrase-based machine translation. *arXiv*, abs/1706.05565.

Diederick P Kingma and Jimmy Ba. 2015. Adam: A method for stochastic optimization. In *ICLR*.

Minh-Thang Luong and Christopher D Manning. 2015. Stanford neural machine translation systems for spoken language domains. In *IWSLT*.

Paul Michel, Omer Levy, and Graham Neubig. 2019. Are sixteen heads really better than one? In *NeurIPS*.

Pavlo Molchanov, Stephen Tyree, Tero Karras, Timo Aila, and Jan Kautz. 2017. Pruning convolutional neural networks for resource efficient inference. In *ICLR*.

Ivan Provilkov, Dmitrii Emelianenko, and Elena Voita. 2019. Bpe-dropout: Simple and effective subword regularization. *arXiv*, abs/1910.13267.

Alessandro Raganato and Jörg Tiedemann. 2018. An analysis of encoder representations in transformer-based machine translation. In *BlackboxNLP@EMNLP*.

Rico Sennrich, Alexandra Birch, Anna Currey, Ulrich Germann, Barry Haddow, Kenneth Heafield, Antonio Valerio Miceli Barone, and Philip Williams. 2017. The university of edinburgh's neural mt systems for wmt17. In *WMT*.

Rico Sennrich, Barry Haddow, and Alexandra Birch. 2016. Neural machine translation of rare words with subword units. In *ACL*.

Nitish Srivastava, Geoffrey E. Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. 2014. Dropout: a simple way to prevent neural networks from overfitting. *JMLR*, 15(1):1929–1958.

Zewei Sun, Shujian Huang, Hao-Ran Wei, Xin-yu Dai, and Jiajun Chen. 2020. Generating diverse translation by manipulating multi-head attention. In *AAAI*.

Zewei Sun, Mingxuan Wang, and Lei Li. 2021. Multilingual translation via grafting pre-trained language models. In *EMNLP*.

Zewei Sun, Mingxuan Wang, Hao Zhou, Chengqi Zhao, Shujian Huang, Jiajun Chen, and Lei Li. 2022. Rethinking document-level neural machine translation. In *ACL*.

Christian Szegedy, Vincent Vanhoucke, Sergey Ioffe, Jonathon Shlens, and Zbigniew Wojna. 2016. Rethinking the inception architecture for computer vision. In *CVPR*.

Gongbo Tang, Rico Sennrich, and Joakim Nivre. 2018. An analysis of attention mechanisms: The case of word sense disambiguation in neural machine translation. In *WMT*.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *NIPS*.

Elena Voita, David Talbot, F. Moiseev, Rico Sennrich, and Ivan Titov. 2019. Analyzing multi-head self-attention: Specialized heads do the heavy lifting, the rest can be pruned. In *ACL*.

Biao Zhang, Ivan Titov, and Rico Sennrich. 2019. Improving deep transformer with depth-scaled initialization and merged attention. In *EMNLP-IJCNLP*.

Zaixiang Zheng, Shujian Huang, Zewei Sun, Rongxiang Weng, Xinyu Dai, and Jiajun Chen. 2018. Learning to discriminate noises for incorporating external information in neural machine translation. *arXiv*, abs/1810.10317.

Adapting to Non-Centered Languages for Zero-shot Multilingual Translation

Zhi Qu and **Taro Watanabe**
Nara Institute of Science and Technology
{qu.zhi.pv5, taro}@is.naist.jp

Abstract

Multilingual neural machine translation can translate unseen language pairs during training, i.e. zero-shot translation. However, the zero-shot translation is always unstable. Although prior works attributed the instability to the domination of central language, e.g. English, we supplement this viewpoint with the strict dependence of non-centered languages. In this work, we propose a simple, lightweight yet effective language-specific modeling method by adapting to non-centered languages and combining the shared information and the language-specific information to counteract the instability of zero-shot translation. Experiments with Transformer on IWSLT17, Europarl, TED talks, and OPUS-100 datasets show that our method not only performs better than strong baselines in centered data conditions but also can easily fit non-centered data conditions. By further investigating the layer attribution, we show that our proposed method can disentangle the coupled representation in the correct direction.[1]

1 Introduction

Training multilingual neural machine translation (MNMT) system requires enormous number of parameters and resources, but the zero-shot translation, namely translating unseen language pairs during training, has shown the potential to simplify the MNMT (Firat et al., 2017). Johnson et al. (2017) has shown that adding language tokens, e.g. <en>, at the beginning of a sentence allows the model to build cross-linguistic representation by treating the token as translation instruction specifying target language. However, the zero-shot translation is always unstable. One possibility causing the instability of zero-shot translation is spurious correlation (Gu et al., 2019). The target linguistic representation captured by the model is directly and strictly dependent on encoded source linguistic information instead of learning specific representations for source and target language, then combining independent linguistic representations to generate results. Prior works (Lakew et al., 2019; Fan et al., 2020; Rios et al., 2020; Freitag and Firat, 2020; Liu et al., 2021) indicated that the spurious correlation is caused by the centered data condition in which multilingual data is constructed by bridging a central language, e.g. English, to other non-centered languages. The central language will dominate the representation in the MNMT model to degenerate the information specific to non-centered languages since multilingual data comprises a set of bilingual data constructed by coupling non-centered languages with the central language. However, the non-centered data condition without any central language is also unstable in zero-shot translation.[2] Therefore, simply attributing the instability of zero-shot translation to the central language cannot fit all cases of zero-shot translation.

We move the perspective from the domination of the central language to the weakness of non-centered languages. The problems of zero-shot translation could be attributed to the strict dependence of non-centered languages. Specifically, a non-centered language would strictly depend on another language as a strongly related language pair to prohibit learning robust and independent translation instructions for zero-shot translation. Under this hypothesis, the centered data condition is a special case of this description, because all non-centered languages depend on the central language. In this light, a key to improving zero-shot translation is disentangling non-centered languages from the strict dependence which is built in training.

Specifically, we model extra language-specific (LS) components (Sachan and Neubig, 2018; Philip et al., 2020; Escolano et al., 2021; Zhang et al., 2021) adapting to non-centered languages in a

[1] Codes and detailed results are available in: https://github.com/zhiqu22/AdapNonCenter

[2] We give specific examples in Section 4.1.

Proceedings of the 29th International Conference on Computational Linguistics, pages 5251–5265
October 12–17, 2022.

mixing shared and LS information mode (Zhang et al., 2021), our objective is to enhance the weak representations for assisting the balance of cross-linguistic representation in shared information container to improve the quality of translation Cheng et al. (2022); Shao and Feng (2022). Furthermore, the mixing mode can decrease the complexity of LS modeling since we treat the representation space of the MNMT model as the combination of shared and LS information, and we no longer build the independent representation space for each language (Sachan and Neubig, 2018; Escolano et al., 2021). In this motivation, we propose a simple, lightweight yet effective method to augment feed-forward network of Transformer (Vaswani et al., 2017) by LS components adapting to non-centered languages.

Our contributions are as follows:

- Our lightweight method achieves considerable gains on multilingual and zero-shot translation and performs stably in IWSLT17, Europarl, TED talks and OPUS-100.
- We describe the strict dependence of non-centered languages to supplement the prior viewpoint of zero-shot translation, and verify it by experiments under different data conditions with and without the central language.
- Our work explores decreasing complexity in LS modeling. We also through the analysis via layer attribution (Dhamdhere et al., 2019) to show the significance of our methods in decoupling representations of MNMT.

2 Related Work

Initially, Johnson et al. (2017) laid the foundation of zero-shot translation which endorses training the MNMT model under the centered data condition and put forward the thinking about the instability of data conditions on zero-shot translation. On this basis, Gu et al. (2019) also showed that the performance of zero-shot translation is sensitive to parameters for initialization, which is another cause of instability. In this paper, we systematically described this instability (Section 5.1) and tested it experimentally.

In the early stages, Mattoni et al. (2017) pointed out that increasing corpus size can effectively improve zero-shot translation. However, the spurious correlation (Gu et al., 2019) means that unreasonably increasing training data could degenerate the zero-shot translation due to strict dependence. Since then, the concern of the centered data condition was started to be discussed by several different strategies. Fan et al. (2020); Freitag and Firat (2020) augmented the training data to make all languages interconnected, which will result in an excessive increase in training costs. Lakew et al. (2019) explored incrementally training the MNMT model by monolingual data, and Gu et al. (2019); Zhang et al. (2020) generated synthetic data for the zero-shot directions by backtranslation. These methods transformed the zero-shot task to zero-resource task.

Another line of work on improving zero-shot translation is to adjust the learning of representations in the MNMT model. Lu et al. (2018); Pham et al. (2019); Zhang et al. (2020); Liu et al. (2021) focused on restricting the representation of encoder outputs to be language-agnostic, but the restriction may reduce the performance of the model trained by large-scale datasets. Pan et al. (2021) aligned representations from different languages via contrastive learning and the additional dictionary. Philip et al. (2020); Yang et al. (2021); Zhang et al. (2021) explored to enhance the influence of LS features in the translation. Our work continues in this direction, but with a special focus on only enhancing the decoding step and mixing shared and LS information.

Our work is based on LS modeling which is the heuristic variation of Mixture-of-Experts model (Shazeer et al., 2017), because it aims to build extra components as experts to directionally improve linguistic features. Sachan and Neubig (2018) and Escolano et al. (2021) built LS encoder or decoder, but multi-encoder/decoder architecture has too many parameters. Wang et al. (2018) divided neural cells into LS parts and Lin et al. (2021) divided LS subnets from the model, but these methods limited the learning capacity. Bapna and Firat (2019) and Philip et al. (2020) added LS adapters on the end of encoder and decoder and fine tuned for LS representations. Zaremoodi et al. (2018); Zhang et al. (2021) explored the paradigm of constructing LS components to assist the shared information. However, extra components always increase the cost of modeling significantly when languages existed too much. The investigation about the importance of LS information specified to target language (Lee et al., 2017; Blackwood et al., 2018; Pham et al., 2019; Wu et al., 2021) enlightens us to limit the

improving LS information in the decoding process to achieve lightweight LS modeling.

3 Central Language Aware Multilingual Neural Machine Translation

We employ Transformer (Vaswani et al., 2017) as the backbone to construct our architecture. Consider a set of m languages $\mathbb{L} = \{l_1, l_2, \ldots, l_m\}$, we assign the first language l_1 as the central language l_c. The non-centered set is the subset of $\mathbb{L}$, that is $\mathbb{L}' = \{l_2, l_3, \ldots, l_m\}$. We follow prior works (Zhang et al., 2020; Liu et al., 2021; Wu et al., 2021) to assign English as the center of multilingual data. Given the original input sequence of symbol representation to the encoder $\boldsymbol{x} = x_1, x_2, ..., x_i$ and the output sequence generated by decoder $\boldsymbol{y} = y_1, y_2, ..., y_j$, we follow the method of Johnson et al. (2017) to insert the language token at the beginning of $\boldsymbol{x}$ as translation instruction. Therefore, the actual input sequence is $\boldsymbol{x}' = (l, \boldsymbol{x})$, and we model the translation of $\boldsymbol{x}'$ to $\boldsymbol{y}$ with Transformer. We only build LS layers (LSLs) parallel with the Feed-Forward Network (FFN) layers in the decoder of Transformer, and keep the self-attention and cross-attention mechanism fixed.

The FFN of transformer consists of two fully connected neural networks with a ReLU activation function in between:

$$\mathrm{FFN}(\boldsymbol{h}) = \max(0, \boldsymbol{h}\mathbf{W}_1 + \boldsymbol{b}_1)\mathbf{W}_2 + \boldsymbol{b}_2 \quad (1)$$

Where $\boldsymbol{h}$ is the input vector, $\mathbf{W}$ indicates parameter matrices for projections, and $\boldsymbol{b}$ indicates bias parameter matrices. LSLs are a series of neural networks specified to $\mathbb{L}'$. Each LSL is similar to FFN in architecture but can be relatively light in inner size:

$$\mathrm{LSL}_l(\boldsymbol{h}) = \max(0, \boldsymbol{h}\mathbf{W}_1^l + \boldsymbol{b}_1^l)\mathbf{W}_2^l + \boldsymbol{b}_2^l \quad (2)$$

where $l \in \mathbb{L}'$. The trade-off between shared and LS information is difficult (Zhang et al., 2021; Wang and Zhang, 2021), because the information that each language carries is not absolutely equal. To balance the shared and LS information, we introduce a set of learnable scalars in each decoder layer $\mathbb{T} = \{t_{l_2}, t_{l_3}, \ldots, t_{l_m}\}$. Elements of $\mathbb{T}$ correspond to languages of $\mathbb{L}'$ one by one, then each t connects LS information with the shared information. We initialize t to 0.1, then parameters[3] of $\mathbb{T}$ are updated during training together with other parameters.

[3] we report the distribution of LS information weights for large-scale dataset (99 languages) in the Appendix C.

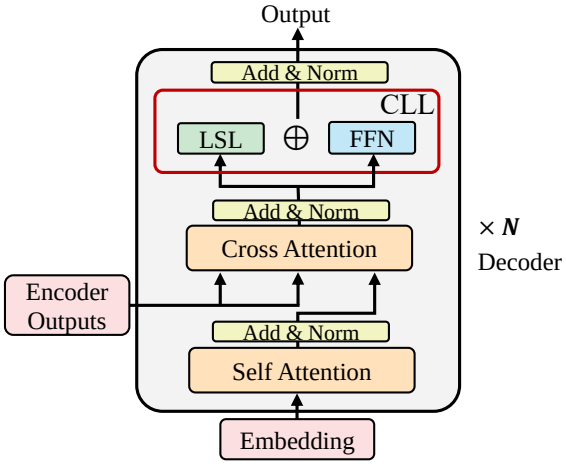

Figure 1: Illustrations of our proposed architecture modified from the decoder of Transformer. $\bigoplus$ indicates weighted plus, N is the number of decoder layers.

To differentiate the central language from non-centered languages, the original FFN of Transformer's decoder is used as the shared information space for all languages of $\mathbb{L}$, and we only construct lightweight LSLs to learn independent linguistic information for non-centered languages of $\mathbb{L}'$. Therefore, the complete architecture of **C**entral **L**anguage-aware **L**ayer (CLL) is:

$$\mathrm{CLL}_l(\boldsymbol{h}) = \begin{cases} \mathrm{FFN}(\boldsymbol{h}) + t_l \, \mathrm{LSL}_l(\boldsymbol{h}) & l \in \mathbb{L}' \\ \mathrm{FFN}(\boldsymbol{h}) & l = l_c \end{cases} \quad (3)$$

Based on the piecewise function Eq.(3), the role of central language will be abandoned in non-centered data conditions, namely the case of $l = l_c$ will not be triggered. We illustrate the architecture of CLL in Figure 1: The CLL is a component, including lightweight LSLs and FFN, to replace the original role of FFN in each decoder layer of Transformer. Compared to the Mixture of Experts which is the generalization of the gating mechanism (Shazeer et al., 2017), a deterministic route specific to language replaces the gate in CLL. For convenience, we use **FCLL** (full CLL) to indicate that the model in which all decoder layers are constructed in the form of our proposed architecture.

We introduce a variation named **SD** that constructs CLL in a single decoder layer among all layers of Transformer, namely Single-Disentangled CLL. Inspired by the work of Liu et al. (2021), we remove the residual connection of FFN in a middle encoder layer to weaken the linguistic features of

Method	+Params	Position
baseline	None	None
FCLL	$\mathcal{O}(k)$	Decoder
SD	$\mathcal{O}(k/N)$	Decoder
Philip et al. (2020)	$\mathcal{O}(2k)$	All
Zhang et al. (2021)	$\mathcal{O}(5k)$	All
Sachan and Neubig (2018)	$\mathcal{O}(K)$	Decoder

Table 1: Number of parameters required for different LS modeling methods. N, k and K denote the number of encoder/decoder layers, parameters per LS layer, and parameters per encoder/decoder layer ($k \ll K$), respectively. Position indicates the position of a model to construct LS components.

encoding. To keep the balance between weakening encoding and improving decoding, we empirically build CLL in the middle decoder. Specifically, given N encoder and decoder layers of Transformer, we remove the residual connection of the FFN in the encoder and replace the FFN with CLL in the decoder at $N/2 + 1^{th}$ layer of both networks. Our experiments (Section 4.3) empirically show that SD has comparable performance with FCLL in small-scale datasets, although SD is more parameter-efficient than FCLL (Table 1).

4 Experiments

4.1 Dataset

We take IWSLT17 (Cettolo et al., 2017) and restrict 4 languages from MMCR4NLP (Dabre and Kurohashi, 2019) to verify basic abilities of multilingual and zero-shot translation. We follow Philip et al. (2020) to experiment on TED talks (Qi et al., 2018) and restrict top 20 languages. We also experiment on OPUS-100 (Zhang et al., 2020) to exhaustively explore the capacity of our proposed method in the large-scale dataset. English is the central language of those cases.

To show the strict dependence of non-centered languages, we design two different cases without central language, namely all languages in the set are non-centered. We extract and reorganize Europarl v7 (Koehn et al., 2005) from MMCR4NLP: 1) Triangle case, where each language appears at the target and source sides only once. Our motivation is to build the strict dependence under the non-centered data condition, and each language pair has more training data than IWSLT. Figure 2a shows its translation directions we designed. 2) Square case, that is designed for avoiding strict dependence as indicated in Figure 2b. Our motivation is to avoid completely interconnecting all languages (Fan et al., 2020; Freitag and Firat, 2020) while building balanced data conditions.

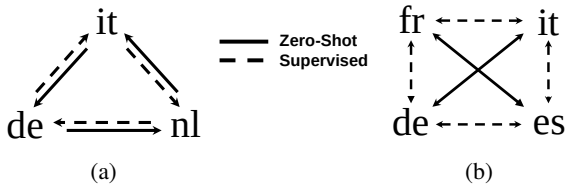

Figure 2: Illustration supervised and zero-shot directions for Triangle and Square cases.

We list details of datasets in Appendix B. And all cases are evaluated via official test sets.

4.2 Experimental Setup

We employ Fairseq (Ott et al., 2019), the open-source implementation, of Transformer (Vaswani et al., 2017) as backbone. Generally, we apply the Moses tokenizer[4] for tokenization and detokenization, and use SentencePiece (Kudo and Richardson, 2018) to learn subword vocabulary. Although the detail of training subword vocabulary for each case has differences, we always train the joint vocabulary including the source and target side, and set *share-all-embedding* in Fairseq. To prevent the unbalanced data size of English-centered datasets from training subword vocabulary, the SentencePiece model is trained by data aggregated from monolingual resources rather than paired resources. We use Adam (Kingma and Ba, 2017) optimizer with the inverse square root schedule in all cases and set different learning rates for different datasets. For fair comparisons, we not only reproduce the Transformer (Vaswani et al., 2017) as Baseline but also reproduce the work of Liu et al. (2021) in all cases, which is denoted by Residual. We always adopt the same hyperparameters setting to prior works and train corresponding subword vocabulary via details described by these works. Specifically, as for the settings for cases of IWSLT, Triangle and Square, we follow the setting of Liu et al. (2021); as for models trained by TED talks and OPUS-100, we follow the setting of Philip et al. (2020) and Zhang et al. (2020), respectively.

We experimented with IWSLT, Triangle, and Square five runs with different random seeds [1,2,3,4,5], to compute the variance for verifying the instability caused by parameters, and other ex-

[4] https://github.com/moses-smt/mosesdecoder

Supervised:	IWSLT		Triangle	Square	TED		OPUS-100	
Method	en→	→en	sup.	sup.	en→	→en	en→	→en
Baseline	31.51	32.93	25.75	32.04	24.23	28.92	19.50	27.60
Philip et al. (2020)	-	-	-	-	24.85	**31.21**	-	-
Zhang et al. (2020)	-	-	-	-	-	-	21.39	27.50
Residual	31.24	32.65	26.25	31.85	22.80	28.19	20.38	26.67
SD	31.63	32.51	26.50	31.97	23.94	28.33	23.60	28.01
FCLL	**31.76**	**33.00**	**26.91**	**32.14**	**25.32**	28.13	**26.17**	**29.33**

Table 2: Averaged BLEU scores on supervised directions. en→ denotes translating from en (l_c) to $\mathbb{L}'$ and →en denotes translating to en from $\mathbb{L}'$; sup. indicates supervised directions in non-centered cases. *Residual* follows Liu et al. (2021) to modify residual connection.

Zero-Shot:	IWSLT		Triangle		Square		TED		OPUS-100			
Method	Z.S.	O.R.	Z.S.	O.R.	Z.S.	O.R.	Z.S.	O.R.	Z.S.	O.R.	F.T.	O.R.
Baseline	16.97	13.95	1.97	93.68	31.18	0.74	10.66	4.16	3.97	63.96	10.11	13.92
Philip et al. (2020)	-	-	-	-	-	-	12.94	-	-	-	-	-
Zhang et al. (2020)	-	-	-	-	-	-	-	-	4.02	54.57	11.98	-
Residual	20.37	1.80	16.60	4.95	30.30	0.77	12.54	3.85	5.14	38.54	11.38	18.30
SD	**21.35**	2.03	19.07	0.92	31.26	0.75	13.03	3.94	4.87	44.07	12.95	12.54
FCLL	21.15	2.05	**20.56**	0.13	**31.49**	0.74	**14.14**	3.74	**6.31**	34.46	**13.65**	11.09

Table 3: Averaged BLEU scores on zero-shot directions. Z.S. column indicates results of zero-shot translation; O.R. denotes the off-target ratio measured by %; F.T. indicates results after fine-tuning, we follow Zhang et al. (2020) to fine-tune 6 languages existing in zero-shot testing.

periments are trained with seed 1. To evaluate results of all experiments, we translate the official test set with beam size 4, and evaluate the translation results by sacreBLEU (Papineni et al., 2002; Post, 2018). We also employ the langdetect[5], which can identify the language of one sentence, to count the off-Target ratio, namely how many sentences are not translated to the correct language. We list detailed experimental settings in Appendix A.

4.3 Results

As described in Table 2, our proposed methods achieve small improvements measured by averaged BLEU scores on supervised directions of IWSLT (+0.25/+0.07), Triangle (+1.16), Square (+0.1), TED (+1.09/-0.79), and OPUS-100 (+6.67/+1.73) compared to Baseline. Liu et al. (2021) speculated that the basic Transformer would overfit more on the supervised direction, and the improvement of zero-shot could hurt supervised translation (Gu et al., 2019; Zhang et al., 2020; Liu et al., 2021). The performance of Residual (Liu et al., 2021) degenerated in TED (-1.43/-0.73) and OPUS-100 (+0.88/-0.93), since the model weakened LS information by trading the generalization ability for zero-shot translation. However, our proposed methods benefit from the additional improvement of decoding the target language by LS modeling (Sachan and Neubig, 2018; Philip et al., 2020). This improvement can counteract the insufficiency of tying artificial language tokens to instruct translation (Arivazhagan et al., 2019). The results of SD can empirically prove the positive impact of CLL, since the performance of SD, which only constructs one CLL, is always between FCLL and Residual on supervised directions. Moreover, the performance of FCLL shows a marked difference (+1.09/-3.08) from the work of Philip et al. (2020). We speculate that the reason is lacking LS structure of l_c and benefiting from the mixture of shared and LS information in CLL. This hypothesis can explain the stable improvements of CLL on Triangle (+1.16) and Square (+0.1) where no l_c exited in training data. We conduct ablation experiments to show the mechanism of CLL in Section 5.4.

Table 3 demonstrates that our methods always give the best scores on zero-shot translations in our experiments. Based on the gain of zero-shot and gain of en→ (Table 2), CLL always positively impacts non-centered languages. In IWSLT, SD performs better than FCLL (+0.2) and performs near FCLL in other cases. It indicates that stacking LS structures is not always optimal for improv-

[5]The tool is not accurate, so, it is just for observing general tendency. (https://github.com/Mimino666/langdetect)

	Zero-Shot			Supervised		
	Baseline	Residual	SD	Baseline	Residual	SD
(1)	14.31	15.06	**16.55**	20.80	20.17	**21.97**
(2)	15.08	16.45	**17.01**	**24.60**	24.38	24.24

Table 4: Averaged BLEU scores of integrating de. Row (1) and Row (2) shows results in $\mathbb{L}_{iwslt} \rightarrow$ de and de $\rightarrow \mathbb{L}_{iwslt}$, respectively.

ing zero-shot translation. It also proves combining tweaking encoding information and improving decoding information would be effective for zero-shot translation. In Triangle, our methods perform stably in the extreme data condition where Baseline totally failed. In Square, all cases have similar performances since these languages do not have strict dependence. Results in TED and OPUS-100 show that our methods also run well in the large-scale dataset. Moreover, we follow Zhang et al. (2020) to fine tune the model by back-translation (Gu et al., 2019) for 6 languages of zero-shot testing, and FCLL achieves a gain of +3.54 BLEU scores to Baseline. These two points show the proposed CLL is orthogonal with other methods excluding LS modeling.

We further noticed that, in Table 2, the performances of all models on zero-shot directions in Square are comparable with each other, and our methods performed stably in Triangle where Baseline is totally failed. The stability of Square case shows the key to improving zero-shot translation is not only large training data (Mattoni et al., 2017), but also the balance of training (Shao and Feng, 2022). The results of Triangle prove CLL is stable in zero-shot translation since it would not be influenced by different data conditions. This feature ensured the effective utilization of shared information. This feature can be proved by the value of off-target rate in Table 3. Given the cost of establishing consistent semantic representation in shared information, confusion about different linguistic features is an inevitable result because the shared information container leads to coupling supervised translation pairs both in theory and practice, however, our proposed methods are always at a relatively lower rate.

4.4 Integrating a new language by few data

The ability to integrate a new language by few data is crucial for low-resource languages when extending a trained MNMT model. To verify this ability of CLL, we fine tune trained SD in IWSLT and extend it to German (de) using bilingual language pairs (en $\leftrightarrow$ de) with 15K sentences per direction, we also fine tune Baseline and Residual as comparison. We follow Liu et al. (2021) to set hyperparameters and update subword vocabulary, as described in Appendix A. Table 4 shows that SD performs better on zero-shot translation, which indicates CLL contributes the cross-lingual knowledge transfer, which indicates that our method is flexible in incorporating low-resource languages.

	IWSLT		Triangle		Square	
	sup.	zero.	sup.	zero.	sup.	zero.
Baseline	0.021	5.280	0.220	0.210	**0.001**	0.016
Residual	0.067	0.270	**0.004**	0.900	0.003	0.055
SD	**0.012**	**0.051**	0.018	0.900	0.002	**0.001**
FCLL	0.025	0.074	0.018	**0.140**	0.004	0.014

Table 5: Variance computed from averaged BLEU scores among five runs in IWSLT, Triangle, and Square with different random seeds. sup. and zero. indicate supervised translation and zero-shot translation, respectively. Smaller variance means a more stable result.

5 Discussion and Analyses

5.1 Instability of Zero-Shot Translation

In this paper, we describe the instability from two related perspectives: 1) Instability of training; 2) Instability of data conditions. For the first point, Table 5 shows the variance for different models counted from five experiments with different seeds for initialization. The small value of variance on supervised translation among the four models shows that supervised training always is a relatively stable process. However, the training process of zero-shot translation is sensitive to initial parameters (Gu et al., 2019), since the variance of zero-shot translation is always higher than the variance of supervised translation. Our methods always achieved the lowest variance on zero-shot translation. For the second point, Table 3 shows that Baseline has completely lost its ability of zero-shot translation in Triangle, although the amount of training samples of Triangle is relatively higher than IWSLT that can result in a good performance on zero-shot translation. On the other hand, Square performs excellently on zero-shot translation and its performance is even closing to the performance of supervised translation (Table 3), although it is non-centered data condition as same as Triangle and it is not completely interconnecting all languages (Fan et al., 2020; Freitag and Firat, 2020). These comparisons proved that the data condition impacts the learning of zero-shot translation.

	Condition	it→nl	ro→it	nl→ro
(1)	Baseline	18.69	16.43	14.13
(2)	(1)+additional pairs	23.27	22.17	21.61
(3)	(2)+reduce data	22.35	21.31	20.96
	SD	22.13	20.27	20.42

Table 6: Variation of different conditions in IWSLT. SD is the performance under original setting.

We further notice that Baseline has a high variance in IWSLT yet a small variance in Triangle. We speculate that the strict dependence of non-centered languages caused instability, and the degree of dependence influences the expression of instability. Specifically, Baseline tends to build cross-linguistic representations in IWSLT, but the strict dependence would couple representations of non-centered languages to the central language to lead to a high off-target ratio in testing zero-shot translation (Table 3). And the higher variance of Baseline in IWSLT means that the model may find a special set of initial parameters to escape from the negative influence of strict dependence. Moreover, the small variance of Baseline in Triangle means that the model completely cannot find a special set among the five times experiments, since Triangle has the most severe dependence of non-centered languages.

To prove our speculation, we create two artificial setups based on IWSLT to re-train Baseline and show results measured by BLEU in Table 6. Specifically, for Row (2), we append three language pairs (it → ro, ro → nl, nl → it) with 30k sentences per pair to balance the dependence (Rios et al., 2020); for Row (3), we sample a random subset of 90K sentences from training data of IWSLT of 145K sentences per translation direction and we append additional pairs as (2). These substantial gains (up to +7.48) of Row (2) in Table 6 proved our viewpoint that data conditions impact the performance of zero-shot translation. Once the model disentangled the strict dependence by appending additional pairs, the model would achieve considerable gains (up to +6.83), although the training samples have been reduced to be smaller than the original setting shown by Row (1). Moreover, the performance of SD is comparable to these artificial cases.

So far we can conclude that the strict dependence of non-centered languages closely influences the zero-shot translation. And our motivation for disentangling the dependence by improving the weak representations of non-centered languages is effective. We will discuss the mechanism of CLL in Section 5.4.

Method	Supervised	Zero-Shot	Off(%)
Residual	32.57	20.74	1.67
FCLL	32.95	21.00	1.52
SD	32.48	21.16	1.97
Residual w/o t	23.72	0.56	-
FCLL w/o t	32.88	20.85	1.35
SD w/o t	32.62	19.46	2.10

Table 7: Averaged BLEU scores of models training without language tokens (w/o t) in IWSLT.

	it→		ro→		nl→	
	ro	nl	it	nl	it	ro
FCLL	21.14	21.92	20.43	22.12	19.45	20.98
Omitted	-0.16	-0.08	-0.37	-0.21	-0.87	-0.74
SD	20.88	22.13	20.27	22.75	20.42	20.51
Omitted	-1.79	-1.40	-2.13	-1.68	-2.41	-2.59

Table 8: Variation of BLEU scores after omitting language tokens in testing.

5.2 Translation Instructions

We re-train our models in IWSLT without language tokens (Johnson et al., 2017), and Table 7 shows the result. First, slight performance gains were observed on supervised directions in FCLL and SD. Figure 3 is a heat map showing self-attention weights of FCLL with and without language tokens. Figure 3a shows one possibility is artificial language tokens (Johnson et al., 2017) might disturb the semantic representation for actual words, since the language token <ro> dominated in self-attention weights. Figure 3b shows the distribution of actual words weights by training without language tokens. Figure 3c presents the attention weights when omitting <ro> in testing the model trained with language tokens, and we observed a similar tendency with the plot in Figure 3b.

Second, Table 7 shows our methods stably maintain cross-linguistic representation although no language tokens were inserted to instruct translation directions both for supervised and zero-shot directions. On the contrary, other methods completely lost their ability of zero-shot translation. These analyses indicate that CLL has a strong capability to instruct multilingual translation.

5.3 Full Layers vs. Single Layer

To verify whether the number of CLL affects the performance of MNMT model, we translate the test set omitting artificial language tokens by trained

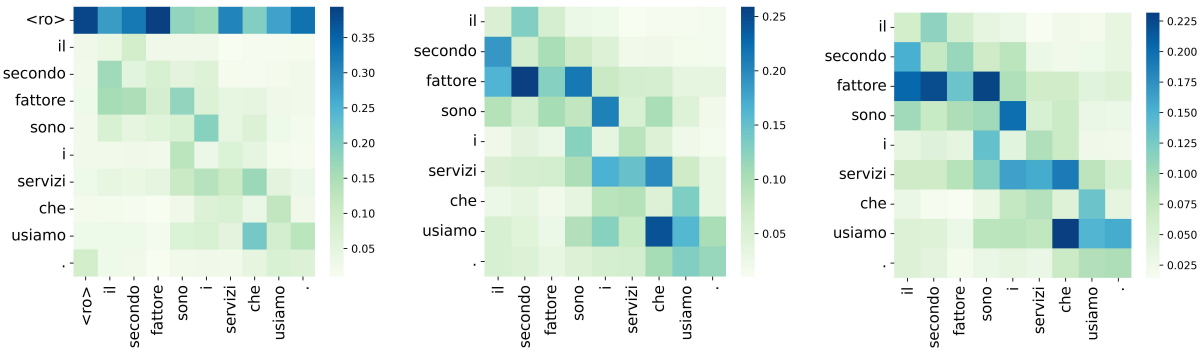

(a) CLL (b) CLL w/o token (c) CLL omitted token

Figure 3: Maps of Self-Attention in which translating one sentence of ro → it.

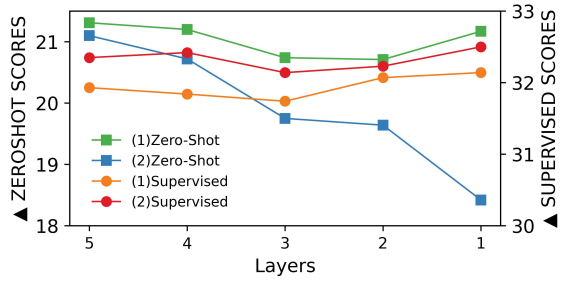

Figure 4: Variation of BLEU scores in which training on different CLL layers. (1) modified the residual connection as SD; (2) did not modify it.

	supervised			zero-shot		
	it→	nl→	de→	it→	nl→	de→
(1)	26.80	26.28	26.00	18.44	19.80	16.36
(2)	25.19	25.77	25.56	0.64	0.75	1.02
(3)	24.85	24.46	25.49	_	_	_

Table 9: Averaged BLEU scores of ablation study. Row (1) shows results in Triangle; Row (2) shows results after ablation; Row (3) means to calculate scores of zero-shot translation by treating the supervised translation results as reference data.

it→de	Ablated LSL of de from CLL
Input:	<de> la quarta priorità concerne l'attenzione che occorre prestare ai nuovi rischi.
Expected Output:	die vierte priorität gilt den neuen risiken.
Actual Output:	de vierde prioriteit is de aandacht die moet worden besteed aan nieuwe risico 's.

Table 10: Ablated testing SD trained in Triangle. The output of the model rolls back to nl (Dutch, the supervised direction).

FCLL and SD in IWSLT. Table 8 demonstrates the model with more CLL layers has stronger robustness since scores of FCLL degenerated less than SD. To further investigate the effect of the number of CLL layers, we re-train Transformer models with different numbers of CLL layers based on IWSLT in two cases. Specifically, in case (1), we modify the residual connection of models as same as the operation of SD, but we do not modify any architecture in encoder of Transformer in case (2). Then, we follow the idea of Liu et al. (2021) to remove the CLL layers in the decoder from the top-most and bottom-most positions until the configuration in which only a single CLL layer is preserved in the middle-position decoder among all decoders of these models.

Figure 4 shows that the zero-shot performance of models in (2) degenerated with the reduction of the number of CLL layers, although the supervised performance always kept in the same magnitude. It proves that the increase in the number of CLL layers has a positive impact on the zero-shot translation. However, almost no clear variations were observed in (1) of Figure 4. One possibility of the lower supervised performance of (1) when compared with (2) in Figure 4 is the weaken language specific information in the encoder by removing the residual connection (Liu et al., 2021). Likewise, the zero-shot performance of (1) is not sensitive to the variation of the number of CLL layers since weakening the capacity of the encoder could partially offset CLL's gains in decoder. We conclude that the architecture of SD is relatively-optimal in small-scale dataset because it is lightweight yet comparable with FCLL, and FCLL is more stable where data condition is complex or large.

5.4 Disentangling Coupled Representation

Ablation Study To investigate the significance of CLL, we ablate LSL from CLL of trained SD

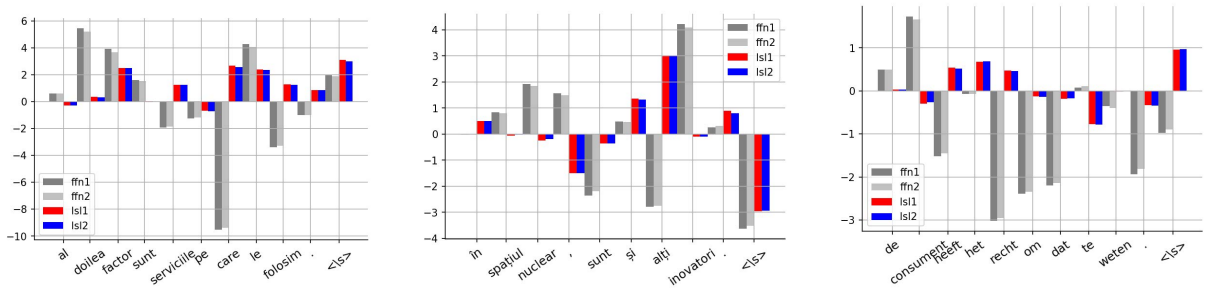

(a) IWSLT:it → ro (b) IWSLT:en → ro (c) Triangle:de → nl

Figure 5: Visualization of layer attributions. ffn indicates FFN, lsl means LSLs, 1 or 2 means it is 1st or 2nd fully connected neural network of this component. A higher absolute value indicates more contribution for result.

in Triangle, namely only use FFN, to re-translate the test set. Row (2) of Table 9 show the degeneration of supervised translation is not significant, but completely losing zero-shot translation capability. However, we observed that the zero-shot translation rollbacks to supervised directions after ablation via analyzing failure cases. As shown in Table 10, the zero-shot translation of it → de will be biased to it → nl due to ablating the layer of de[6]. Thus, we calculated the BLEU scores of zero-shot translation by treating the test set of it → nl as reference data in testing Row (3) of Table 9. The slight degeneration of Row (3) strongly proved that FFN has built a consistent semantic representation which has been coupled to supervised directions.

Layer Attributions The layer attribution[7] can quantify the contributions of one component by integrated gradients (Sundararajan et al., 2017). We designed 3 scenarios to observe these attributions in details: a) The zero-shot translation based on centered case; b) The supervised translation based on centered case; c) The zero-shot translation based on non-centered case. Figure 5 demonstrates: 1) FFN always plays the main role in translation; 2) Generally, the contributions of CLL are on the contrary of FFN in LS words, but they have similar contributions in common words, especially the punctuation.

These results proved our viewpoint in Section 5.1 again. Specifically, the shared representations built in FFN potentially enable cross-linguistic transferring, but the strict dependence of non-centered languages would hamper freely transferring since cross-linguistic information is coupled with supervised translation directions. Therefore, the significance of LSL in CLL practically is to provide independent LS information to disentangle the coupled representation, namely counteract the negative influence of the dependence, to present a correct LS representation in decoding.

6 Conclusion

In this work, we supplement the theory of zero-shot translation with the strict dependence of non-centered languages, and we describe the instability of zero-shot translation. To counteract the influence of the dependence, we proposed a simple yet effective method that employs LS modeling by adapting to non-centered languages. Our analysis based on layer attribution demonstrated that LS information is conducive to disentangling the coupled model representation. Our experiments on various datasets and different data conditions show that our proposed method outperforms in performance and complexity.

7 Ethical Considerations

The potential ethical risk of our work is the usage of multilingual datasets including IWSLT, Europarl, TED talks and OPUS-100, since these datasets might contain social biases, especially in the Europarl, in which predominant European languages might constitute stereotypes. Those biases would be represented in the trained model and could be amplified by integrating one new language out of trained language families since no special treatment is performed to mitigate the biases. Generally, this method can be landed in the industry under sufficient anti-prejudice measures.

Acknowledgements

This work was in part supported by JSPS KAKENHI Grant Numbers 21H05054.

[6] we report more examples in Appendix D in which including long and short sentences in different cases.

[7] We employ Captum (https://github.com/pytorch/captum) for computing attributions.

References

Naveen Arivazhagan, Ankur Bapna, Orhan Firat, Roee Aharoni, Melvin Johnson, and Wolfgang Macherey. 2019. The missing ingredient in zero-shot neural machine translation.

Ankur Bapna and Orhan Firat. 2019. Simple, scalable adaptation for neural machine translation. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 1538–1548, Hong Kong, China. Association for Computational Linguistics.

Graeme Blackwood, Miguel Ballesteros, and Todd Ward. 2018. Multilingual neural machine translation with task-specific attention. In *Proceedings of the 27th International Conference on Computational Linguistics*, pages 3112–3122, Santa Fe, New Mexico, USA. Association for Computational Linguistics.

Mauro Cettolo, Marcello Federico, Luisa Bentivogli, Niehues Jan, Stüker Sebastian, Sudoh Katsuitho, Yoshino Koichiro, and Federmann Christian. 2017. Overview of the iwslt 2017 evaluation campaign. In *International Workshop on Spoken Language Translation*, pages 2–14.

Yong Cheng, Ankur Bapna, Orhan Firat, Yuan Cao, Pidong Wang, and Wolfgang Macherey. 2022. Multilingual mix: Example interpolation improves multilingual neural machine translation.

Raj Dabre and Sadao Kurohashi. 2019. Mmcr4nlp: Multilingual multiway corpora repository for natural language processing.

Kedar Dhamdhere, Mukund Sundararajan, and Qiqi Yan. 2019. How important is a neuron. In *International Conference on Learning Representations*.

Carlos Escolano, Marta R. Costa-jussà, José A. R. Fonollosa, and Mikel Artetxe. 2021. Multilingual machine translation: Closing the gap between shared and language-specific encoder-decoders. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*, pages 944–948, Online. Association for Computational Linguistics.

Angela Fan, Shruti Bhosale, Holger Schwenk, Zhiyi Ma, Ahmed El-Kishky, Siddharth Goyal, Mandeep Baines, Onur Celebi, Guillaume Wenzek, Vishrav Chaudhary, Naman Goyal, Tom Birch, Vitaliy Liptchinsky, Sergey Edunov, Edouard Grave, Michael Auli, and Armand Joulin. 2020. Beyond english-centric multilingual machine translation.

Orhan Firat, Kyunghyun Cho, Baskaran Sankaran, Fatos T. Yarman Vural, and Yoshua Bengio. 2017. Multi-way, multilingual neural machine translation. *Computer Speech & Language*, 45:236–252.

Markus Freitag and Orhan Firat. 2020. Complete multilingual neural machine translation. In *Proceedings of the Fifth Conference on Machine Translation*, pages 550–560, Online. Association for Computational Linguistics.

Jiatao Gu, Yong Wang, Kyunghyun Cho, and Victor O.K. Li. 2019. Improved zero-shot neural machine translation via ignoring spurious correlations. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 1258–1268, Florence, Italy. Association for Computational Linguistics.

Melvin Johnson, Mike Schuster, Quoc V. Le, Maxim Krikun, Yonghui Wu, Zhifeng Chen, Nikhil Thorat, Fernanda Viégas, Martin Wattenberg, Greg Corrado, Macduff Hughes, and Jeffrey Dean. 2017. Google's multilingual neural machine translation system: Enabling zero-shot translation. *Transactions of the Association for Computational Linguistics*, 5:339–351.

Diederik P. Kingma and Jimmy Ba. 2017. Adam: A method for stochastic optimization.

Philipp Koehn et al. 2005. Europarl: A parallel corpus for statistical machine translation. In *MT summit*, volume 5, pages 79–86. Citeseer.

Taku Kudo and John Richardson. 2018. SentencePiece: A simple and language independent subword tokenizer and detokenizer for neural text processing. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 66–71, Brussels, Belgium. Association for Computational Linguistics.

Surafel M. Lakew, Marcello Federico, Matteo Negri, and Marco Turchi. 2019. Multilingual neural machine translation for zero-resource languages.

Jason Lee, Kyunghyun Cho, and Thomas Hofmann. 2017. Fully Character-Level Neural Machine Translation without Explicit Segmentation. *Transactions of the Association for Computational Linguistics*, 5:365–378.

Zehui Lin, Liwei Wu, Mingxuan Wang, and Lei Li. 2021. Learning language specific sub-network for multilingual machine translation. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 293–305, Online. Association for Computational Linguistics.

Danni Liu, Jan Niehues, James Cross, Francisco Guzmán, and Xian Li. 2021. Improving zero-shot translation by disentangling positional information. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 1259–1273, Online. Association for Computational Linguistics.

Yichao Lu, Phillip Keung, Faisal Ladhak, Vikas Bhardwaj, Shaonan Zhang, and Jason Sun. 2018. A neural interlingua for multilingual machine translation. In *Proceedings of the Third Conference on Machine Translation: Research Papers*, pages 84–92, Brussels, Belgium. Association for Computational Linguistics.

Giulia Mattoni, Pat Nagle, Carlos Collantes, and Dimitar Shterionov. 2017. Zero-shot translation for indian languages with sparse data. In *Proceedings of the MT Summit*, pages 1–10.

Myle Ott, Sergey Edunov, Alexei Baevski, Angela Fan, Sam Gross, Nathan Ng, David Grangier, and Michael Auli. 2019. fairseq: A fast, extensible toolkit for sequence modeling. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics (Demonstrations)*, pages 48–53, Minneapolis, Minnesota. Association for Computational Linguistics.

Myle Ott, Sergey Edunov, David Grangier, and Michael Auli. 2018. Scaling neural machine translation. In *Proceedings of the Third Conference on Machine Translation: Research Papers*, pages 1–9, Brussels, Belgium. Association for Computational Linguistics.

Xiao Pan, Mingxuan Wang, Liwei Wu, and Lei Li. 2021. Contrastive learning for many-to-many multilingual neural machine translation. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 244–258, Online. Association for Computational Linguistics.

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. Bleu: a method for automatic evaluation of machine translation. In *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics*, pages 311–318, Philadelphia, Pennsylvania, USA. Association for Computational Linguistics.

Ngoc-Quan Pham, Jan Niehues, Thanh-Le Ha, and Alexander Waibel. 2019. Improving zero-shot translation with language-independent constraints. In *Proceedings of the Fourth Conference on Machine Translation (Volume 1: Research Papers)*, pages 13–23, Florence, Italy. Association for Computational Linguistics.

Jerin Philip, Alexandre Berard, Matthias Gallé, and Laurent Besacier. 2020. Monolingual adapters for zero-shot neural machine translation. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 4465–4470, Online. Association for Computational Linguistics.

Matt Post. 2018. A call for clarity in reporting BLEU scores. In *Proceedings of the Third Conference on Machine Translation: Research Papers*, pages 186–191, Brussels, Belgium. Association for Computational Linguistics.

Ye Qi, Devendra Sachan, Matthieu Felix, Sarguna Padmanabhan, and Graham Neubig. 2018. When and why are pre-trained word embeddings useful for neural machine translation? In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 2 (Short Papers)*, pages 529–535, New Orleans, Louisiana. Association for Computational Linguistics.

Annette Rios, Mathias Müller, and Rico Sennrich. 2020. Subword segmentation and a single bridge language affect zero-shot neural machine translation. In *Proceedings of the Fifth Conference on Machine Translation*, pages 528–537, Online. Association for Computational Linguistics.

Devendra Sachan and Graham Neubig. 2018. Parameter sharing methods for multilingual self-attentional translation models. In *Proceedings of the Third Conference on Machine Translation: Research Papers*, pages 261–271, Brussels, Belgium. Association for Computational Linguistics.

Chenze Shao and Yang Feng. 2022. Overcoming catastrophic forgetting beyond continual learning: Balanced training for neural machine translation.

Noam Shazeer, Azalia Mirhoseini, Krzysztof Maziarz, Andy Davis, Quoc Le, Geoffrey Hinton, and Jeff Dean. 2017. Outrageously large neural networks: The sparsely-gated mixture-of-experts layer.

Mukund Sundararajan, Ankur Taly, and Qiqi Yan. 2017. Axiomatic attribution for deep networks. In *Proceedings of the 34th International Conference on Machine Learning - Volume 70*, ICML'17, page 3319–3328. JMLR.org.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Ł ukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc.

Qian Wang and Jiajun Zhang. 2021. Parameter differentiation based multilingual neural machine translation.

Yining Wang, Jiajun Zhang, Feifei Zhai, Jingfang Xu, and Chengqing Zong. 2018. Three strategies to improve one-to-many multilingual translation. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 2955–2960, Brussels, Belgium. Association for Computational Linguistics.

Liwei Wu, Shanbo Cheng, Mingxuan Wang, and Lei Li. 2021. Language tags matter for zero-shot neural machine translation. In *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*, pages 3001–3007, Online. Association for Computational Linguistics.

Yilin Yang, Akiko Eriguchi, Alexandre Muzio, Prasad Tadepalli, Stefan Lee, and Hany Hassan. 2021. Improving multilingual translation by representation

and gradient regularization. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 7266–7279.

Poorya Zaremoodi, Wray Buntine, and Gholamreza Haffari. 2018. Adaptive knowledge sharing in multi-task learning: Improving low-resource neural machine translation. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 656–661, Melbourne, Australia. Association for Computational Linguistics.

Biao Zhang, Ankur Bapna, Rico Sennrich, and Orhan Firat. 2021. Share or not? learning to schedule language-specific capacity for multilingual translation. In *International Conference on Learning Representations*.

Biao Zhang, Philip Williams, Ivan Titov, and Rico Sennrich. 2020. Improving massively multilingual neural machine translation and zero-shot translation. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 1628–1639, Online. Association for Computational Linguistics.

A Detailed Settings

IWSLT & Triangle & Square We follow Liu et al. (2021) to set 5 encoder/decoder layers with 8 attention heads, embedding size of 512, inner size of 2048, dropout rate of 0.3, dropout rate of CLL layer of 0.3, maximum learning rate of 0.0005 and label smoothing rate of 0.1. However, we decrease dropout rate to 0.1 and dropout rate of CLL layer to 0.2 in Square that is a bigger case than others. The size of subword vocabulary is 40K for each case. In training, we set the maximum batch size per GPU to 4,000 tokens, and train on 4 GPUs. We train for 100K steps for IWSLT and Triangle, but train for 500K steps for Square. We sample the supervised and zero-shot translation directions from the dev set of MMCR4NLP as the validation dataset in training.

TED talks We follow Philip et al. (2020) to set 6 encoder/decoder layers with 4 attention heads, embedding size of 512, inner size of 1024, dropout rate of CLL layer of 0.3, maximum learning rate of 0.0005 and label smoothing rate of 0.1. However, we set the dropout rate to 0.2 to get better performances. The size of subword vocabulary is 70K. In training, we set the maximum batch size per GPU to 4,000 tokens, and train on 4 GPUs. We train for 90 epochs to ensure models convergent. We only sample dev sets of supervised directions translating as the validation dataset in training. We also follow Philip et al. (2020) to use mixed-precision (Ott et al., 2018) in training.

OPUS-100 We follow Zhang et al. (2020) to set 6 encoder/decoder layers with 8 attention heads, embedding size of 512, inner size of 2048, dropout rate to 0.1, dropout rate of CLL layer of 0.2, maximum learning rate of 0.0007 and label smoothing rate of 0.1. We directly reuse their published subword vocabulary[8]. In training, we set the maximum batch size per GPU to 6,000 tokens, and train on 8 GPUs[9] for 500K steps. We follow Zhang et al. (2020) to sample top 200 sentences in dev sets of supervised directions translating as the validation dataset in training.

In fine-tuning, we follow Zhang et al. (2020) to back-translate the training resource to get the pseudo resource, then we merge real and pseudo resources to train 4 epochs, and we update the pseudo training resource after each epoch in training. We set 500 warm-up steps at the beginning of fine-tuning, reset the optimizer, and training with maximum learning rate of 0.0003.

Integrating de in IWSLT Based on the trained model in IWSLT, we learn a new SentencePiece model with 10K vocabulary size to acquire a dictionary for de. Then we append the new dictionary to the end of the previously learned dictionary of IWSLT, meanwhile, we keep the order of the previous part unchanged. Due to the increased number of unique tokens, we resize token embedding and initialize new vectors as the average of existing embedding perturbed by random noise. When fine-tuning, we set the learning rate as the value at the end of the previous training, freeze parameters of CLL layers of existing languages, initialize parameters of CLL layers for de by averaging existing CLL layers, and include the original training data of IWSLT to prevent the shared information from tending to translate de.

[8] https://github.com/bzhangGo/zero

[9] We use Fairseq command line of *–update-freq 2* to simulate the efficiency of 8 GPUs by 4 GPUs.

B Dataset Details

Dataset case	Languages	# zero-shot directions	# sent. per direction
IWSLT	{en, it, ro, nl}	6	145K
Europarl Triangle	{__, it, nl, de}	3	200K
Europarl Square	{__, fr, it, de, es}	4	1M
TED	{ en, ar, bg, de, es, fa, fr, he, hu, it, ja, ko, nl, pl, pt-br, ro, ru, tr, vi, zh-cn }	342	140K ~210K
OPUS-100	{ en, an, as, be, bg, bn, br, bs, ca, cs, cy, da, de, el, es, fa, fr, fy, ga, gd, gl, gu, hi, hr, hy, is, it, ku, li, lt, lv, mk, mr, nb, ne, nl, nn, no, oc, or, pa, pl, ps, pt, ro, ru, sh, si, sk, sl, sq, sr, sv, tg, uk, ur, wa, yi, az, kk, ky, tk, tr, tt, ug, uz, dz, my, zh, et, fi, hu, se, id, km, mg, ms, vi, ig, rw, xh, yo, zu, kn, ml, ta, te, eo, eu, ja, ko, ka, mn, th}	30	2K ~1M

Table 11: Overview of datasets. The underline denotes the l_c, and the underline with blank represents non-centered condition, i.e. no English.

C Distribution of Language-Specific Information Weights

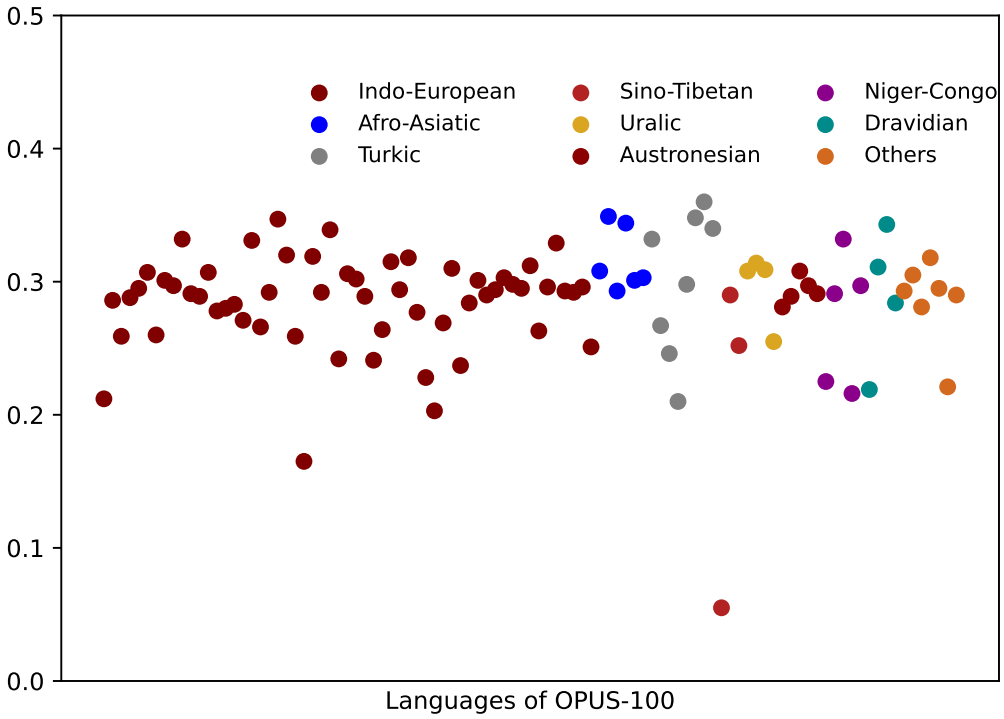

Figure 6: Averaged weights over all layers in FCLL model. The x-coordinate is sorted by languages showed in Table 11. For languages with the same amount of training resources, languages from the same language family have relatively similar weights.

D Translation Examples of Ablation

Supervised: it→nl	Ablated LSL of nl from CLL	Language
Input:	<nl> parlo adesso per esperienza personale: da anni nell'industria dell'aviazione civile e con la commissione siamo infarciti di deregolamentazione, eppure,in relazione ai diritti aeroportuali, ci viene detto adesso che la risposta è la regolamentazione.	it
Expected Output:	ik spreek nu namens mijzelf: al vele jaren wordt ons nu binnen de burgerluchtvaartindustrie en met de commissie een dieet voorgeschoteld van deregulatie en toch, waar het gaat om luchthavenbelasting, wordt ons nu verteld dat regelgeving het antwoord is.	nl
Actual Output:	ik spreek nu uit persoonlijke ervaring: al jaren in de burgerluchtvaartindustrie en met de commissie zijn we gedwongen tot deregulering, maar wat de luchthavengelden betreft, wordt ons nu gezegd dat het antwoord de regelgeving is.	nl
Input:	<nl> tutte le cose importanti sono già state dette.	it
Expected Output:	al het belangrijke is reeds gezegd.	nl
Actual Output:	al het belangrijke is reeds gezegd.	nl
Supervised: de→it	Ablated LSL of it from CLL	Language
Input:	<it> der verbraucher hat ein recht darauf, das zu wissen.	de
Expected Output:	il consumatore ha il diritto di saperlo.	it
Actual Output:	il consumatore ha il diritto di saperlo.	it
Zero-Shot: it→de	Ablated LSL of de from CLL	Language
Input:	<de> il recepimento di parte dell'acquis nel primo pilastro apre la strada alla comunitarizzazione di questa politica e consente di adottare anche rimedi in relazione alla nebulosa schengen, come amava chiamarla il mio predecessore.	it
Expected Output:	mit der teilweisen übernahme des acquis in den ersten pfeiler stehen uns nun alle wege offen, diese politik zu vergemeinschaften und licht in la nébuleuse schengen zu bringen, wie es mein vorredner beschrieb.	de
Actual Output:	de omzetting van een deel van het acquis in de eerste pijler maakt de weg vrij voor de communautarisering van dit beleid en maakt het mogelijk dat er ook oplossingen worden gevonden voor de nebulosa schengen, zoals mijn voorganger zei.	nl
Input:	<de> la quarta priorità concerne l'attenzione che occorre prestare ai nuovi rischi.	it
Expected Output:	die vierte priorität gilt den neuen risiken.	de
Actual Output:	de vierde prioriteit is de aandacht die moet worden besteed aan nieuwe risico 's.	nl
Zero-Shot: nl→it	Ablated LSL of it from CLL	Language
Input:	<it> die solidariteit en die noodzaak tot samenwerking geldt ook als zich in de toekomst problemen voordoen, bijvoorbeeld bij interne migratiestromen.	nl
Expected Output:	questa sicurezza e la necessità di una collaborazione sono essi stessi potenziali problemi futuri, ad esempio per quanto riguarda la migrazione interna.	it
Actual Output:	diese solidarität und die notwendigkeit der zusammenarbeit gelten auch in zukunft, z. b. in bezug auf die migrationsströme.	de

Table 12: Some examples of translation by trained SD in the Triangle, in which ablating LS layers from CLL. The long sentence of supervised translation has degeneration compared with short sentences but is kept in the correct direction. These zero-shot translations are biased to supervised directions.

Towards Robust Neural Machine Translation with Iterative Scheduled Data-Switch Training

Zhongjian Miao[1] [†*] **Xiang Li**[2†] **Liyan Kang**[1] **Wen Zhang**[2] **Chulun Zhou**[1]
Yidong Chen[1‡] **Bin Wang**[2] **Min Zhang**[3] **Jinsong Su**[1,4‡]
[1]School of Informatics, Xiamen University, China
[2]Xiaomi AI Lab, China
[3]Harbin Institute of Technology, Shenzhen, China
[4]Pengcheng Lab, China
miaozhongjian@stu.xmu.edu.cn lixiang21@xiaomi.com {ydchen,jssu}@xmu.edu.cn

Abstract

Most existing methods on robust neural machine translation (NMT) construct adversarial examples by injecting noise into authentic examples and indiscriminately exploit two types of examples. They require the model to translate both the authentic source sentence and its adversarial counterpart into the identical target sentence within the same training stage, which may be a suboptimal choice to achieve robust NMT. In this paper, we first conduct a preliminary study to confirm this claim and further propose an *Iterative Scheduled Data-switch Training Framework* to mitigate this problem. Specifically, we introduce two training stages, iteratively switching between authentic and adversarial examples. Compared with previous studies, our model focuses more on just one type of examples at each single stage, which can better exploit authentic and adversarial examples, and thus obtaining a better robust NMT model. Moreover, we introduce an improved curriculum learning method with a sampling strategy to better schedule the process of noise injection. Experimental results show that our model significantly surpasses several competitive baselines on four translation benchmarks. Our source code is available at https://github.com/DeepLearnXMU/RobustNMT-ISDST.

1 Introduction

In recent years, neural machine translation (NMT) has achieved great success (Sutskever et al., 2014; Bahdanau et al., 2015; Vaswani et al., 2017). Usually, the NMT models are trained on clean parallel corpus and thus achieve promising performance under clean inputs. However, small perturbations, such as replacing words in the input sentences, can mislead the trained model to generate incorrect translations (Belinkov and Bisk, 2018). In real-world scenarios, it is often required to deal with such sentences. Thus, it has important academic value and application prospects to design a robust NMT model for both clean and noisy inputs.

To reach this goal, some researchers explore data-oriented approaches focusing on constructing adversarial examples (Cheng et al., 2020; Zou et al., 2020). Generally, adversarial examples are used to augment the authentic dataset or fine-tune an NMT model pre-trained on the authentic dataset to improve robustness. Although data-oriented approaches are simple and efficient, they leverage adversarial examples coarsely, as concluded by Wang et al. (2021a) and Passban et al. (2021), which can not reach the full potential of these examples.

Besides, researchers also study model-oriented approaches. Some design additional model components to correct noisy inputs (Zhou et al., 2019; Qin et al., 2021; Wang et al., 2021a). There are more studies exploring training strategies for robust NMT, including multi-task learning (Zhou et al., 2019; Zhang et al., 2020), contrastive learning (Yang et al., 2019; Lee et al., 2021), and adversarial training (Cheng et al., 2018, 2019).

Despite their success, there still exist two drawbacks: 1) most existing methods indiscriminately exploit authentic and adversarial examples within the same training stage, which is a suboptimal choice confirmed in our preliminary study; 2) previous studies on robust NMT adopt a constant noise ratio to construct adversarial examples during training, while the determination of noise ratio is a subtle process, *i.e.*, too little noise may lead to poor robustness and too much noise may also hurt the model performance (Jiao et al., 2021). Therefore, dealing with both clean and noisy inputs well for NMT remains to be a significant but challenging task.

In this paper, we first conduct a preliminary study, which reveals that indiscriminately exploit-

[*]Work was done when interning at Xiaomi AI Lab.
[†]Equal Contribution.
[‡]Corresponding Author.

Proceedings of the 29th International Conference on Computational Linguistics, pages 5266–5277
October 12–17, 2022.

ing authentic and adversarial examples within the same training stage is suboptimal. Concretely, we find that this training strategy can not significantly reduce the source sentence representation (SSR) discrepancies[1] between authentic examples and the corresponding adversarial examples, resulting in a suboptimal model training which is reflected by lower model confidence[2] on examples. Based on this observation, we further propose an *Iterative Scheduled Data-Switch Training Framework* for robust NMT. Under this framework, we train the model in a two-stage scheme, iteratively switching between authentic and adversarial examples with their individual modified training objectives. During training, we introduce an additional Kullback-Leibler (KL) divergence loss, expecting the model to make similar predictions on authentic and adversarial datasets. By doing so, at each training stage, the model not only focuses on one of authentic and adversarial datasets but also avoids forgetting the knowledge from the other. Therefore, our model is able to handle both clean and noisy inputs well.

Furthermore, we introduce curriculum learning (CL) to better schedule the process of noise injection. Particularly, inspired by the *Baby Step* strategy (Wang et al., 2021b) in CL that gradually exposes more difficult examples to the model while still involving simple examples, we sample the noise ratio from a uniform distribution, where the sampling interval is progressively extended. Compared with the naive CL strategy of continuously increasing the noise ratio, our strategy is re-sampling previous simple adversarial examples which is beneficial to the model generalization.

In summary, our contributions are as follows:

- Through in-depth analyses, we expose the suboptimum of indiscriminately exploiting authentic and adversarial examples within the same training stage, and further propose an iterative data-switch training framework for robust NMT.
- Instead of using a constant noise ratio, we introduce an improved curriculum learning method with a sampling strategy to better schedule the process of noise injection at each training stage.
- Empirical evaluations on four translation benchmarks validate the superiority of our framework, and in-deep analyses also verify the effectiveness of various factors on our framework.

2 Preliminary Study

Indiscriminately exploiting authentic examples and their adversarial counterparts within the same training stage is an effective way to build a robust NMT model. However, it requires the model to overcome the SSR discrepancy between an authentic example $(\mathbf{x}, \mathbf{y})$ and its adversarial counterpart $(\mathbf{x}', \mathbf{y})$, which increases the training difficulty to maximize $P(\mathbf{y}|\mathbf{x}; \boldsymbol{\theta})$ and $P(\mathbf{y}|\mathbf{x}'; \boldsymbol{\theta})$ simultaneously. We argue it may be a better choice to exploit authentic and adversarial examples at two training stages, iteratively switching between two types of examples. In such a data-switch training manner, the model can better benefit from the knowledge of different stages

To verify our hypothesis, we use Transformer (Vaswani et al., 2017) as our NMT model and conduct a preliminary experiment on the IWSLT14 De⇒En dataset. To be specific, we train the three models: 1) *Transformer*. We follow Vaswani et al. (2017) to train this model on the authentic dataset; 2) *Indisc-Model*. It indiscriminately exploits authentic and adversarial examples for training within the same stage. Besides, following Passban et al. (2021), we introduce a mean square error (MSE) loss to enforce the corresponding encoder outputs to be similar; 3) *Switch-Model*. This model is trained at two training stages, iteratively switching between authentic and adversarial examples. We make an investigation through the two metrics: 1) the Euclidean distances of the SSR between authentic examples and their adversarial counterparts; 2) the model confidence, *i.e.*, log-likelihood values of target ground-truth sentences.

2.1 Source Sentence Representation Discrepancy

Intuitively, to obtain high-quality translations, the SSRs from authentic and adversarial examples are expected to be similar. Therefore, we first calculate the Euclidean distances of the SSRs between two types of examples. As shown in Figure 1, the dis-

[1]We average the word representations from encoder outputs to obtain the SSRs. The SSR discrepancies represent the difference of the source sentence representations between authentic and adversarial examples, and the higher SSR discrepancies correspond to more divergent translations for authentic and adversarial examples.

[2]Model confidence represents the predicted probability for the target ground-truth sentences (Briakou and Carpuat, 2021; Zhou et al., 2022).

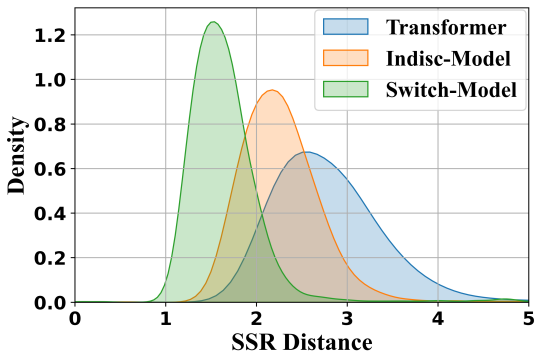

Figure 1: The kernel density estimation visualization of the SSR distances between authentic examples and their adversarial counterparts. Here, we average word representations from encoder outputs to obtain the SSRs. The authentic examples are from the entire test set, and the adversarial examples are constructed from them as mentioned in Section 3.2.

Model	SSR Distance	Model Confidence	
		Adv.	Aut.
Transformer	2.96	-43.0	-39.2
Indisc-Model	2.36	-41.5	-38.7
Switch-Model	**1.69**	**-41.1**	**-38.4**

Table 1: The averaged sentence-level SSR distances between authentic examples (Aut.) and their adversarial counterparts (Adv.), and the model confidence on examples.

tance distribution of *Transformer* is far from the Y-axis, and the distance distribution of *Switch-Model* is closer to the Y-axis, while the distribution of *Indisc-Model* lies between the above distributions. These results indicate *Switch-Model* reduces the SSR discrepancies well. As reported in Table 1, we also calculate the averaged sentence-level SSR distances. *Switch-Model* achieves the lowest score. These results, along with the SSR visualization (See Appendix A.1), further support the above conclusion.

2.2 Model Confidence

Higher model confidence generally leads to high-quality translations (Briakou and Carpuat, 2021; Zhou et al., 2022). Herein, we calculate the averaged sentence-level log-likelihood values for authentic and adversarial examples, respectively. As reported in Table 1, although *Indisc-Model* achieves better model confidence than *Transformer*, especially on adversarial examples, *Switch-Model* still obtains the best scores on both authentic and adversarial examples. These results indicate that *Switch-Model* is trained better on authentic and adversarial examples compared to *Indisc-Model*.

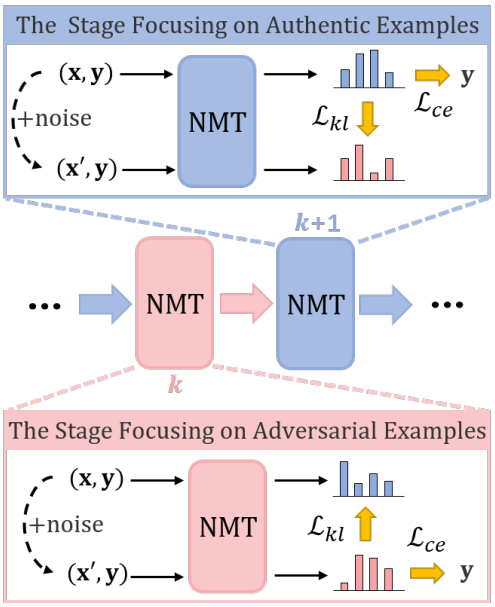

Figure 2: Diagram of the two training stages in our framework, where the pink box and blue box denote the training stages focusing on adversarial examples (the k-th iteration) and authentic examples (the (k+1)-th iteration), respectively. $(\mathbf{x}', \mathbf{y})$ is an adversarial example constructed from its authentic counterpart $(\mathbf{x}, \mathbf{y})$ with curriculum learning mentioned in Section 3.2. $\mathcal{L}_{kl}$ and $\mathcal{L}_{ce}$ denote KL-divergence loss and cross-entropy loss.

3 Methodology

Based on the observations in Section 2, we further propose an iterative scheduled data-switch training framework for robust NMT.

3.1 Training Framework

In contrast to the previous work, our framework introduces two iterative training stages to handle authentic and adversarial examples, respectively.

As shown in Figure 2, at the training stage focusing on adversarial examples (the k-th iteration), we first use the best model at the last training stage (the (k-1)-th iteration) as initialization, and then optimize the model on two types of examples using a modified training objective. Specifically, we additionally introduce KL-divergence loss into the conventional training objective, expecting the model predictions on adversarial examples to be close to those on authentic examples. Formally, the modified training objective $\mathcal{L}_{adv}$ at this stage is defined as follows:

$$\mathcal{L}_{adv} = \sum_{\substack{(\mathbf{x},\mathbf{y})\in\mathcal{D} \\ (\mathbf{x}',\mathbf{y})\in\mathcal{D}'}} [-\log P(\mathbf{y}|\mathbf{x}';\boldsymbol{\theta}) + \alpha \mathbf{KL}(P(\mathbf{y}|\mathbf{x}';\boldsymbol{\theta})||P(\mathbf{y}|\mathbf{x};\boldsymbol{\theta}))], \quad (1)$$

where α is a weight factor, $\boldsymbol{\theta}$ denotes the model

parameters, $(\mathbf{x}, \mathbf{y})$ and $(\mathbf{x}', \mathbf{y})$ denote an authentic example and its adversarial counterpart, respectively.

Likewise, at the training stage focusing on authentic examples, the modified training objective $\mathcal{L}_{aut}$ is given by

$$\mathcal{L}_{aut} = \sum_{\substack{(\mathbf{x},\mathbf{y})\in\mathcal{D} \\ (\mathbf{x}',\mathbf{y})\in\mathcal{D}'}} [-\log P(\mathbf{y}|\mathbf{x};\boldsymbol{\theta}) + \alpha \mathbf{KL}(P(\mathbf{y}|\mathbf{x};\boldsymbol{\theta})||P(\mathbf{y}|\mathbf{x}';\boldsymbol{\theta}))]. \quad (2)$$

We conduct training stages for K iterations. In such an iterative data-switch training manner, the knowledge of different stages can continuously enhance the model in a collaborative way, which has also been verified in previous studies (Zeng et al., 2019; Liu et al., 2020b).

3.2 Generate Adversarial Examples with Curriculum Learning

During training, we generate adversarial examples on the fly by injecting noise into the source sentences of the corresponding authentic examples. Without loss of generality, we inject noise by performing three common operations with equal probability: *delete*, *replace*, and *swap*. Note that our framework is also applicable to other types of noise.

Previous work on robust NMT pays little attention to the noise ratio during training. In this work, we introduce curriculum learning (CL) to schedule the process of noise injection at each training stage. Inspired by the *Baby Step* strategy in CL (Wang et al., 2021b), at each training step, we sample the noise ratio from a uniform distribution, where the sampling interval is progressively extended. By doing so, our sampling strategy re-samples previous simple adversarial examples during training, which is beneficial to the model generalization.

The procedure of generating adversarial examples is presented in Algorithm 1. At the training step t, we first load a batch of examples and sample a noise ratio r_t from a uniform distribution $U(0, R(t))$ (**Lines 3-4**). Intuitively, a sharp increase of $R(t)$ may hurt the model optimization. Therefore, we expect that $R(t)$ increases smoothly. To this end, we define $R(t)$ as follows:

$$R(t) = \sqrt{R_{max}^2 \times \frac{t}{T}}, \quad (3)$$

where R_{max} is the maximal noise ratio and T denotes the maximal training step number of each training stage. Note that the derivative of $R(t)$ decreases with the increase of t, which satisfies our expectations that $R(t)$ increases smoothly (See Appendix A.2). According to r_t, we traverse each authentic example $(\mathbf{x}, \mathbf{y})$ in the current mini-batch (**Line 5**) and determine the number n_t of perturbed words in $\mathbf{x}$ (**Line 6**), and then perform three kinds of operations with equal probability on them to generate adversarial examples $(\mathbf{x}', \mathbf{y})$ (**Line 7**). Finally, we train the model with our modified objective based on two types of examples (**Line 8**). For efficiency, we update $R(t)$ every 10K training step (**Lines 10-12**).

Algorithm 1 Generate Adversarial Examples with Curriculum Learning for Each Training Stage

Input: Training corpus $\mathcal{D}$, maximal training step number T, maximal noise ratio R_{max}.

1: $R(t) \leftarrow 0$
2: **for** t =1, 2, ..., T **do**
3: Load a mini-batch $\mathcal{B}_t$ from $\mathcal{D}$
4: Sample a noise ratio $r_t \sim U(0, R(t))$
5: **for** each example $(\mathbf{x}, \mathbf{y})$ in $\mathcal{B}_t$ **do**
6: $n_t \leftarrow \lceil len(\mathbf{x}) \times r_t \rceil$
7: Perturb n_t words in $\mathbf{x}$ to generate its adversarial counterpart $\mathbf{x}'$
8: Using ($\mathbf{x}$,$\mathbf{y}$) and ($\mathbf{x}'$,$\mathbf{y}$) to train the model according to the modified training objective defined in Equation 1 or Equation 2
9: **end for**
10: **if** t % 10K == 0 **then**
11: Update $R(t)$ by Equation 3
12: **end if**
13: **end for**

4 Experiments

4.1 Setup

Datasets For the small-scale dataset, we use IWSLT14 German⇒English (De⇒En) corpus, where the training set comprises 160K sentence pairs extracted from TED talks, the original validation set consists of dev2010 and dev2012, and the clean test set consists of tst2010, tst2011 and tst2012. For the middle-scale datasets, we use MTNT[3] French⇒English (Fr⇒En) (Michel and Neubig, 2018) and WMT14 English⇒German (En⇒De) datasets. The former consists of 2.2M sentence pairs for training, newsdiscussdev2015 is

[3] https://pmichel31415.github.io/mtnt/index.html#data

used as the original validation set, newstest2014 (NT14) and newsdicusstest2015 (NT15) are used as the clean test sets. The latter contains 4.5M sentence pairs, and we choose newstest2013 as our original validation set, and newstest2014 as our clean test set. For the large-scale dataset, we use WMT20 Chinese⇒English (Zh⇒En) dataset containing 22M sentence pairs for training and newstest2019 (with 1,997 sentence pairs) for validating and newstest2020 (with 1,418 sentence pairs) for testing.

Note that in this work, we focus on the performance on clean and noisy test sets. Thus we select the best model according to the hybrid validation sets, each of which contains the original validation set and its disturbed counterpart. In addition to the standard clean test sets, we also evaluate models on noisy test sets. For the De⇒En, En⇒De and Zh⇒En translation tasks, we construct the synthetic noisy test sets by performing operations (See Section 3.2) on a certain ratio of source words in the original test sets. For the Fr⇒En translation task, we evaluate models on two social media test sets with diverse noise: mtnt18 (Michel and Neubig, 2018) and mtnt19 (Li et al., 2019), both of which have been widely used in robust NMT task (Li et al., 2019).

We also employ BPE (Sennrich et al., 2016) to split words into subwords. During this process, the numbers of merge operations are separately set to 10K, 16K, 32K and 32K for De⇒En, Fr⇒En, En⇒De and Zh⇒En datasets. Finally, we report case-sensitive tokenized BLEU (Papineni et al., 2002) for the De⇒En, En⇒De and Zh⇒En translation tasks and sacreBLEU (Post, 2018) for the Fr⇒En translation task.

Training Details We adopt the *fairseq*[4] (Ott et al., 2019) Transformer as our basic model. We use the *transformer_iwslt_de_en* setting for the De⇒En translation task, and the *transformer_wmt_en_de* setting for the En⇒De, Fr⇒En and Zh⇒En translation tasks, respectively.

As for the model optimization, we use the Adam optimizer (Kingma and Ba, 2015) with β_1=0.9, β_2=0.98 and ϵ=10^{-9}. All experiments are done on NVIDIA V100 GPUs with mixed-precision training, where batch sizes are roughly set to 4K, 8K, 32K, and 32K tokens for the De⇒En, Fr⇒En, En⇒De, and Zh⇒En translation tasks, respectively. For all datasets, we set the maximal noise ratio R_{max} as 0.1 and we tune the weight factor $\alpha \in \{0.5, 1.0, 1.5\}$ on our validation sets at the first training stage, then keep it unchanged in subsequent stages for efficiency. We determine the maximal training step number T through an empirical study according to the convergence of the model at each stage. Specifically, we set T for the stages focusing on authentic and adversarial examples to 150K and 200K, respectively.

Baselines In addition to the vanilla Transformer model (Vaswani et al., 2017), we compare our model with the following baselines:

- *Transformer-FT*. It is pre-trained on the authentic dataset and then fine-tuned on the adversarial dataset.
- *Transformer-Mixed*. This model is trained on the dataset mixed with authentic and adversarial examples.
- *Transformer-Indisc*. It indiscriminately exploits authentic and adversarial examples for training. Besides, the model predictions between two types of examples are minimized via a bidirectional KL-divergence loss (Liang et al., 2021).
- *MTNT* (Michel and Neubig, 2018). It is the first benchmark on the MTNT Fr⇒En dataset.
- *AdvST* (Cheng et al., 2018). This model is trained using *adversarial stability training* strategy, which enables the encoder and decoder to generate similar representations for the original inputs and their perturbed counterparts
- *SwitchOut* (Wang et al., 2018). It uses a data augmentation strategy for training, where the augmented data is constructed by randomly replacing words in source and target sentences with other words.
- *DouAdv* (Cheng et al., 2019). It generates discrete adversarial examples with doubly adversarial inputs according to the gradients of word embeddings.
- *MTL* (Zhou et al., 2019). It introduces multitask learning into robust NMT, where two decoders are involved: one learns to denoise the text and the other generates the final translations from the denoised text.

[4] https://github.com/fairseq/fairseq

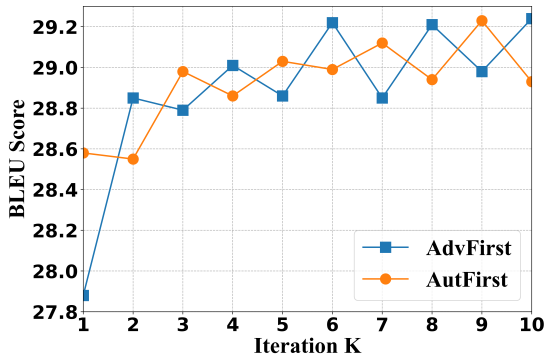

Figure 3: BLUE (%) scores of our model on the Fr⇒En validation set with different K. AdvFirst and AutFirst denote we focus on adversarial and authentic examples first, respectively.

- *ContRec* (Xu et al., 2021). This model reduces the effect of noisy words through a context-enhanced reconstruction component.

4.2 Effects of Data Order and Iteration Number K

Under our framework, it should be determined which type of examples we need to first focus on and what the appropriate iteration number K is. We explore their effects in this subsection. To this end, we train the models focusing on authentic and adversarial examples first with different K, respectively. The results on the validation sets are displayed in Figure 3.

Which Type of Examples to First Focus on? As illustrated in Figure 3, we observe that the model focusing on adversarial examples first reaches a competitive result at the 6th iteration, while the model focusing on authentic examples first needs 9 iterations to obtain a similar result, indicating the former converges faster to a better result.

What Is the Appropriate Iteration Number K? Overall, as iteration number K increases, we find the model performance is improved, whether we focus on authentic or adversarial examples first.

Based on these results on the validation sets, we choose to first focus on adversarial examples and set the iteration number K to 6 for the Fr⇒En dataset. Similarly, we set K to 5 for all other datasets.

4.3 Main Results

Results on Clean Test Sets Table 2 shows the results on clean test sets for the De⇒En, En⇒De, Zh⇒En tasks, and the results for the Fr⇒En task are reported in the second and third columns of Table 3. Data-oriented approaches achieve comparable or worse results compared to *Transformer*, indicating data-oriented approaches may hurt the performance on the standard clean test sets. *Transformer-Indisc* is a strong baseline model. It performs better than *Transformer* and achieves promising performance compared to other baselines, except for the Zh⇒En task. Compared with the data-oriented and model-oriented baselines, our model achieves the best performance across all datasets. Concretely, our model achieves +0.59 BLEU improvement than the most competitive contrast model *DouAdv* on the En⇒De dataset. For the large-scale Zh⇒En dataset, all related approaches fail and do not outperform *Transformer*, while our model achieves +0.62 BLEU improvement over *Transformer*. These results fully demonstrate the superiority of our framework.

Model	De⇒En	En⇒De	Zh⇒En
Transformer	34.82	27.78	26.83
Data-Oriented			
Transformer-FT	34.90	27.75	25.76
Transformer-Mixed	34.85	27.72	24.07
Model-Oriented			
AdvST (Cheng et al., 2018)	—	25.26	—
DouAdv (Cheng et al., 2019)	—	28.34	—
Transformer-Indisc	36.59	28.21	26.51
Ours	**37.28***†	**28.93***†	**27.45***†

Table 2: BLEU (%) scores on the clean test sets of four translation tasks. '$*$' and '†' mean the improvements over *Transformer-Indisc* and *Transformer* are significantly with p<0.01 (Koehn, 2004).

Model	Clean Test		Noisy Test	
	NT14	NT15	mtnt18	mtnt19
Transformer	31.76	31.14	25.67	29.74
MTNT (Michel and Neubig, 2018)	28.90	30.80	23.30	26.20
Data-Oriented				
Transformer-FT	32.37	30.71	26.54	29.03
Transformer-Mixed	31.87	30.71	25.44	27.90
SwitchOut (Wang et al., 2018)	29.20	31.10	25.00	28.10
Model-Oriented				
MTL (Zhou et al., 2019)	—	—	24.50	30.30
ConRec (Xu et al., 2021)	30.70	32.40	26.50	29.10
Transformer-Indisc	32.83	31.37	26.42	28.98
Ours	**34.11***†	**32.67***†	**28.16***†	**30.77***†

Table 3: BLEU (%) scores on the Fr⇒En translation task. '$*$' and '†' mean the improvements over *Transformer-Indisc* and *Transformer* are significantly with p<0.01 (Koehn, 2004).

Results on Noisy Test Sets To verify the model robustness, we evaluate models on the synthetic noisy test sets and the social media test sets, respectively.

The fourth and fifth columns of Table 3 report the results on the social media test sets. It is worth

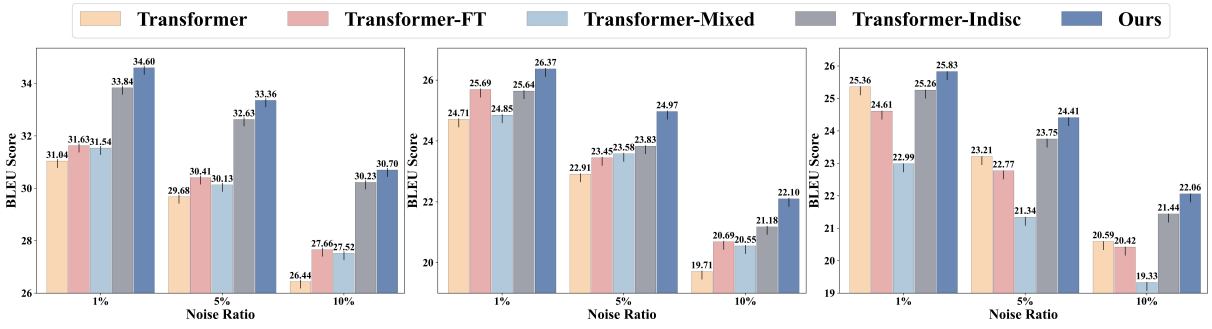

(a) IWSLT14 De ⇒ En (b) WMT14 En ⇒ De (c) WMT20 Zh ⇒ En

Figure 4: BLEU (%) scores on the synthetic noisy test sets with different noise ratios.

noticing that although we inject finite types of noise during training, our model still beats other baselines on both mtnt18 and mtnt19 test sets, which shows the better generalization of our model.

For the synthetic noisy test sets, we compare the performance of all models on the test sets with different noise ratios. As shown in Figure 4, *Transformer* suffers from performance drops under noisy inputs, revealing the vulnerability of the NMT model. By contrast, data-oriented approaches perform slightly better than *Transformer* across different noise ratios, except for the large-scale Zh⇒En dataset. We argue the robustness achieved by data-oriented approaches is restricted because the Zh⇒En training set is large enough to cover diverse noises. Additionally, *Transformer-Indisc* performs better than data-oriented approaches and *Transformer*, showing its strong robustness. Finally, we find our model consistently outperforms other baselines across different noise ratios even under the large-scale data configuration, which confirms again that our framework can significantly enhance the model robustness.

4.4 Source Sentence Representation Discrepancy and Model Confidence

Following the settings of the preliminary study in Section 2, we evaluate models using two metrics: the SSR distances and model confidence. As shown in Figure 5, the distance distribution of our model is significantly closer to the Y-axis compared to *Transformer* and *Transformer-Indisc*, indicating that our model significantly reduces the SSR discrepancies. Analogously, we report the averaged sentence-level SSR distances in Table 4 and visualize the SSRs for clear understanding (See Appendix A.1), all of which demonstrate the effectiveness of our model. Besides, the averaged sentence-level log-likelihood values presented in Table 4 show that our model ob-

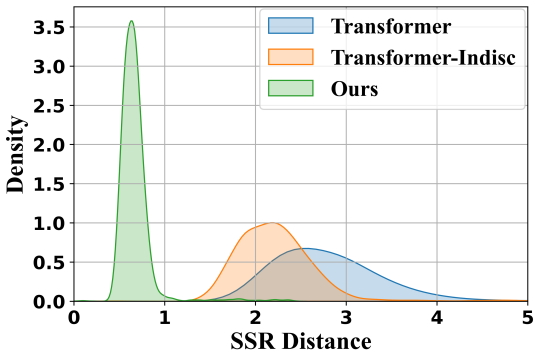

Figure 5: The kernel density estimation visualization of the SSR distances.

Model	SSR Distance	Model Confidence	
		Adv.	Aut.
Transformer	2.96	-43.0	-39.2
Transformer-Indisc	2.25	-38.3	-35.8
Ours	**0.67**	**-38.0**	**-35.6**

Table 4: The averaged sentence-level SSR distances and the model confidence on examples.

tains the highest model confidence on two types of examples. It implies our model is trained better on authentic and adversarial examples. In summary, the results of the two metrics show that our model can deal with clean and noisy inputs well.

4.5 Effects of Different Types of Noise

To better understand the effects of different types of noise, we inject only one type of noise into the training data and the test set respectively, and then inspect the performance change of our model. From Table 5, we arrive at the following conclusions:

(1) *The models injecting different types of noise into the training set perform similarly on the clean test set*. From the first column of Table 5, we observe that adopting *delete* and *replace* operations separately during training perform slightly better than our hybrid noise strategy, while adopting *swap* operation obtains the worst performance.

(2) *When only one type of noise is injected, our*

Model	Different Types of Noise				
	Clean	Hybrid	Swap	Replace	Delete
Ours-Hybrid	37.28	**34.60**	**36.56**	**32.98**	33.85
Ours-Swap	36.91	33.54	36.29	31.62	32.63
Ours-Replace	**37.54**	33.83	35.04	32.97	33.48
Ours-Delete	37.51	34.27	35.78	32.11	**34.40**

Table 5: The effects of different types of noise on the IWSLT14 De⇒En dataset. Here, the noise ratios of all noise test sets are set to 1% and **bold** indicates the best result for each noise test set (each column).

model performs better if both training and test sets are injected with the same type of noise. For example, adopting *swap* operation during training obtains 36.29 BLEU on the *swap* noise test set, while adopting *replace* and *delete* operations obtain 35.04 and 35.78 BLEU on the same test set, respectively.

(3) *The performance of the model on the test sets with different types of noise differs greatly*. Comparing each column in Table 5, the model performs worst on the *replace* noise test set, while the *swap* noise has relatively little damage to the model performance.

(4) *The hybrid noise strategy we adopt achieves balanced results*. Comparing each row in Table 5, we find that our model with the hybrid strategy achieves the best results on the *hybrid*, *swap* and *replace* noise test sets and competitive results on the rest test sets.

4.6 Ablation Study

To verify the effectiveness of various factors on our framework, we further compare our framework with the following variants and present the results in Table 6:

(1) *w/ FNR*. In this variant, we directly use a **F**ixed **N**oise **R**atio to schedule the process of noise injection. As reported in Table 6, this variant decreases the performance dramatically on both clean and noisy test sets. It reveals the importance of scheduling the noise injection with CL.

(2) *w/ FSI*. In our improved CL method, the sampling interval is progressively extended. In this variant, we adopt a **F**ixed **S**ampling **I**nterval and the noise ratio is sampled uniformly from it. As shown in Table 6, using a fixed sampling internal also leads to the performance degradation.

(3) *w/o SS*. Inspired by the *Baby Step* (Wang et al., 2021b) in CL, we equip CL with a **S**ampling **S**trategy (See Section 3.2). Note that our CL strategy degenerates into the naive CL strategy (the variant *w/o SS*) if we remove the *SS* component. The results listed in Table 6 demonstrate the effectiveness of our sampling strategy.

Model	Clean Test		Noisy Test	
	NT14	NT15	mtnt18	mtnt19
Ours	34.11	32.67	28.16	30.77
w/ FNR	32.15	29.93	23.70	27.51
w/ FSI	33.77	31.21	26.25	29.44
w/o SS	33.47	31.37	26.22	30.43
w/o KL	33.09	30.28	25.84	29.05
KL⇒MSE	32.52	30.92	26.62	29.16

Table 6: Ablation study on the Fr⇒En translation task.

(4) *w/o KL*. We introduce a KL-divergence loss to ensure that the model focuses more on one type of examples at each stage while preventing forgetting the knowledge from another type. As shown in Table 6, compared with the variant *w/o KL*, this regularization term indeed enhances the model capability to cope with both clean and noisy inputs.

(5) *KL⇒MSE*. In this variant, we replace KL-divergence loss with the MSE loss on decoder output hidden states. From Table 6, we can observe that this variant performs better than the framework without KL-divergence loss (the variant *w/o KL*) in 3 out of 4 test sets, showing the importance of the regularization term. However, compared to the MSE regularization, the KL-divergence regularization is more suitable for our framework.

5 Related Work

To build robust NMT models, researchers have proposed a range of methods, which can be mainly divided into two categories: data-oriented and model-oriented approaches.

In the first category, how to construct adversarial examples is a non-trivial problem (Cheng et al., 2020; Zou et al., 2020). Usually, adversarial examples are used in two ways: one is to directly train a robust model using the dataset mixed with authentic and adversarial examples (Belinkov and Bisk, 2018; Karpukhin et al., 2019), and the other is to use adversarial examples to fine-tune the NMT model pre-trained on authentic examples (Helcl et al., 2019; Dabre and Sumita, 2019; Berard et al., 2019; Alam and Anastasopoulos, 2020).

In the second category, some researchers design additional components for NMT model to correct noisy inputs (Qin et al., 2021; Wang et al., 2021a; Xu et al., 2021) or explore fault-tolerant neural networks(Su et al., 2017; Tan et al., 2018). Mean-

while, more researchers resort to exploring training strategies, including multi-task learning (Zhou et al., 2019; Zhang et al., 2020), contrastive learning (Yang et al., 2019; Lee et al., 2021), and adversarial training (Cheng et al., 2018, 2019; Liu et al., 2020a).

In this work, the proposed framework belongs to the second model-oriented category. In this regard, most existing methods indiscriminately exploit authentic and adversarial examples within the same training stage, which are suboptimal confirmed in our preliminary study. To mitigate this problem, we propose an iterative scheduled data-switch training framework for robust NMT, where we introduce two training stages, iteratively switching between authentic and adversarial examples. Besides, inspired by the successful applications of curriculum learning (CL) in NMT (Platanios et al., 2019; Xu et al., 2020; Zhou et al., 2020), we use CL to better schedule the process of noise injection. Particularly, we equip CL with a sampling strategy, which is beneficial to the model generalization.

Finally, note that Jiao et al. (2021) introduce an alternated training to alleviate the performance drop caused by low-quality back-translation data. Our work differs from theirs in three aspects: 1) we aim at building a robust NMT model dealing with clean and noisy inputs well, while Jiao et al. (2021) try to prevent the model performance on clean test sets from being disturbed by synthetic data; 2) we introduce an improved CL method to better schedule the process of noise injection, which is beneficial to the model performance; 3) in addition to the conventional cross-entropy objective (Jiao et al., 2021), we introduce an additional regularization term to cope with both clean and noisy inputs well.

6 Conclusion

In this paper, we first conduct a preliminary study to reveal that indiscriminately exploiting authentic and adversarial examples for robust NMT is suboptimal. To achieve better robust NMT, we further propose an iterative scheduled data-switch training framework, where we train the model at two training stages, iteratively switching between authentic and adversarial examples. Moreover, we introduce curriculum learning with a sampling strategy to schedule the process of noise injection at each training stage. Extensive experiments show the superiority of our framework.

In the future, we will introduce more types of real noise, such as ASR errors, into our framework. Besides, we plan to apply our framework to other natural language generation tasks, such as dialogue generation, so as to verify the generality of our framework.

Acknowledgements

This research was supported by National Natural Science Foundation of China (Nos. 62036004, 62276219, 62076211, 61573294), Natural Science Foundation of Fujian Province of China (No. 2020J06001), and Youth Innovation Fund of Xiamen (No. 3502Z20206059). We would like to thank Yong Cheng (Google Research), Yang Liu (Tsinghua University) for valuable feedback on this work and the reviewers for their insightful comments.

References

Md Mahfuz Ibn Alam and Antonios Anastasopoulos. 2020. Fine-tuning MT systems for robustness to second-language speaker variations. In *Proc. of W-NUT@EMNLP*.

Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. 2015. Neural machine translation by jointly learning to align and translate. In *Proc. of ICLR*.

Yonatan Belinkov and Yonatan Bisk. 2018. Synthetic and natural noise both break neural machine translation. In *Proc. of ICLR*.

Alexandre Berard, Ioan Calapodescu, and Claude Roux. 2019. Naver labs europe's systems for the WMT19 machine translation robustness task. In *Proc. of WMT*.

Eleftheria Briakou and Marine Carpuat. 2021. Beyond noise: Mitigating the impact of fine-grained semantic divergences on neural machine translation. In *Proc. of ACL*.

Yong Cheng, Lu Jiang, and Wolfgang Macherey. 2019. Robust neural machine translation with doubly adversarial inputs. In *Proc. of ACL*.

Yong Cheng, Lu Jiang, Wolfgang Macherey, and Jacob Eisenstein. 2020. Advaug: Robust adversarial augmentation for neural machine translation. In *Proc. of ACL*.

Yong Cheng, Zhaopeng Tu, Fandong Meng, Junjie Zhai, and Yang Liu. 2018. Towards robust neural machine translation. In *Proc. of ACL*.

Raj Dabre and Eiichiro Sumita. 2019. Nict's supervised neural machine translation systems for the WMT19 translation robustness task. In *Proc. of WMT*.

Jindrich Helcl, Jindrich Libovický, and Martin Popel. 2019. CUNI system for the WMT19 robustness task. In *Proc. of WMT*.

Rui Jiao, Zonghan Yang, Maosong Sun, and Yang Liu. 2021. Alternated training with synthetic and authentic data for neural machine translation. In *Proc. of ACL Findings*.

Vladimir Karpukhin, Omer Levy, Jacob Eisenstein, and Marjan Ghazvininejad. 2019. Training on synthetic noise improves robustness to natural noise in machine translation. In *Proc. of W-NUT@EMNLP*.

Diederik P. Kingma and Jimmy Ba. 2015. Adam: A method for stochastic optimization. In *Proc. of ICLR*.

Philipp Koehn. 2004. Statistical significance tests for machine translation evaluation. In *Proc. of EMNLP*.

Seanie Lee, Dong Bok Lee, and Sung Ju Hwang. 2021. Contrastive learning with adversarial perturbations for conditional text generation. In *Proc. of ICLR*.

Xian Li, Paul Michel, Antonios Anastasopoulos, Yonatan Belinkov, Nadir Durrani, Orhan Firat, Philipp Koehn, Graham Neubig, Juan Miguel Pino, and Hassan Sajjad. 2019. Findings of the first shared task on machine translation robustness. In *Proc. of WMT*.

Xiaobo Liang, Lijun Wu, Juntao Li, Yue Wang, Qi Meng, Tao Qin, Wei Chen, Min Zhang, and Tie-Yan Liu. 2021. R-drop: Regularized dropout for neural networks. In *Proc. of NIPS*.

Kai Liu, Xin Liu, An Yang, Jing Liu, Jinsong Su, Sujian Li, and Qiaoqiao She. 2020a. A robust adversarial training approach to machine reading comprehension.

Xin Liu, Kai Liu, Xiang Li, Jinsong Su, Yubin Ge, Bin Wang, and Jiebo Luo. 2020b. An iterative multi-source mutual knowledge transfer framework for machine reading comprehension. In *Proc. of IJCAI*.

Paul Michel and Graham Neubig. 2018. MTNT: A testbed for machine translation of noisy text. In *Proc. of EMNLP*.

Myle Ott, Sergey Edunov, Alexei Baevski, Angela Fan, Sam Gross, Nathan Ng, David Grangier, and Michael Auli. 2019. fairseq: A fast, extensible toolkit for sequence modeling. In *Proc. of NAACL*.

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. Bleu: a method for automatic evaluation of machine translation. In *Proc. of ACL*.

Peyman Passban, Puneeth S. M. Saladi, and Qun Liu. 2021. Revisiting robust neural machine translation: A transformer case study. In *Proc. of EMNLP Findings*.

Emmanouil Antonios Platanios, Otilia Stretcu, Graham Neubig, Barnabás Póczos, and Tom M. Mitchell. 2019. Competence-based curriculum learning for neural machine translation. In *Proc. of NAACL*.

Matt Post. 2018. A call for clarity in reporting BLEU scores. In *Proc. of WMT*.

Wenjie Qin, Xiang Li, Yuhui Sun, Deyi Xiong, Jianwei Cui, and Bin Wang. 2021. Modeling homophone noise for robust neural machine translation. In *Proc. of ICASSP*.

Rico Sennrich, Barry Haddow, and Alexandra Birch. 2016. Neural machine translation of rare words with subword units. In *Proc. of ACL*.

Jinsong Su, Zhixing Tan, Deyi Xiong, Rongrong Ji, Xiaodong Shi, and Yang Liu. 2017. Lattice-based recurrent neural network encoders for neural machine translation. In *Proc. of AAAI*.

Ilya Sutskever, Oriol Vinyals, and Quoc V. Le. 2014. Sequence to sequence learning with neural networks. In *Proc. of NIPS*.

Zhixing Tan, Jinsong Su, Boli Wang, Yidong Chen, and Xiaodong Shi. 2018. Lattice-to-sequence attentional neural machine translation models. *Neurocomputing*.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Proc. of NIPS*.

Tao Wang, Chengqi Zhao, Mingxuan Wang, Lei Li, Hang Li, and Deyi Xiong. 2021a. Secoco: Self-correcting encoding for neural machine translation. In *Proc. of EMNLP Findings*.

Xin Wang, Yudong Chen, and Wenwu Zhu. 2021b. A comprehensive survey on curriculum learning. *IEEE T-PAMI*.

Xinyi Wang, Hieu Pham, Zihang Dai, and Graham Neubig. 2018. Switchout: an efficient data augmentation algorithm for neural machine translation. In *Proc. of EMNLP*.

Chen Xu, Bojie Hu, Yufan Jiang, Kai Feng, Zeyang Wang, Shen Huang, Qi Ju, Tong Xiao, and Jingbo Zhu. 2020. Dynamic curriculum learning for low-resource neural machine translation. In *Proc. of COLING*.

Weiwen Xu, Ai Ti Aw, Yang Ding, Kui Wu, and Shafiq R. Joty. 2021. Addressing the vulnerability of NMT in input perturbations. In *Proc. of NAACL*.

Zonghan Yang, Yong Cheng, Yang Liu, and Maosong Sun. 2019. Reducing word omission errors in neural machine translation: A contrastive learning approach. In *Proc. of ACL*.

Jiali Zeng, Yang Liu, Jinsong Su, Yubin Ge, Yaojie Lu, Yongjing Yin, and Jiebo Luo. 2019. Iterative dual domain adaptation for neural machine translation. In *Proc. of EMNLP*.

Huaao Zhang, Shigui Qiu, Xiangyu Duan, and Min Zhang. 2020. Token drop mechanism for neural machine translation. In *Proc. of COLING*.

Chulun Zhou, Fandong Meng, Jie Zhou, Min Zhang, Hongji Wang, and Jinsong Su. 2022. Confidence based bidirectional global context aware training framework for neural machine translation. *In Proc. of ACL*.

Shuyan Zhou, Xiangkai Zeng, Yingqi Zhou, Antonios Anastasopoulos, and Graham Neubig. 2019. Improving robustness of neural machine translation with multi-task learning. In *Proc. of WMT*.

Yikai Zhou, Baosong Yang, Derek F. Wong, Yu Wan, and Lidia S. Chao. 2020. Uncertainty-aware curriculum learning for neural machine translation. In *Proc. of ACL*.

Wei Zou, Shujian Huang, Jun Xie, Xinyu Dai, and Jiajun Chen. 2020. A reinforced generation of adversarial examples for neural machine translation. In *Proc. of ACL*.

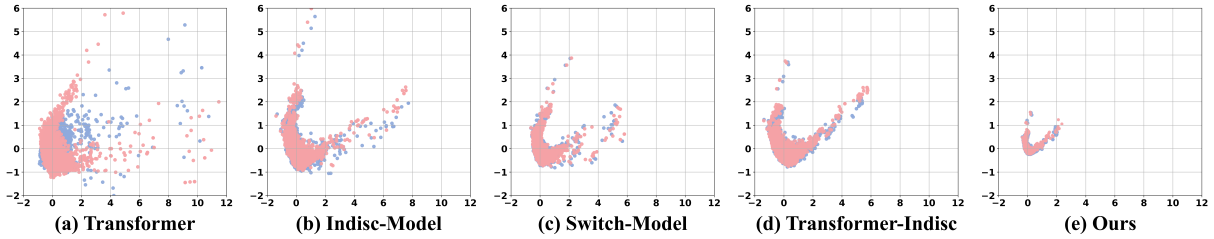

Figure 6: Visualization of the SSRs for authentic examples and the corresponding adversarial examples. Here, we apply the PCA algorithm to reduce the source sentence representations to the 2-dim ones and visualize them. The pink and blue dots denote adversarial and authentic examples, respectively.

A Appendix

A.1 Visualization of the Source Sentence Representations

To understand the source sentence representations (SSRs) clearly, we apply the PCA algorithm to the SSRs of authentic and adversarial examples and visualize the SSRs. Herein, following the settings of the preliminary study (See Section 2), we average word representations to obtain the SSRs. The authentic examples are obtained from the entire test set of the IWSLT14 De⇒En dataset, and the adversarial counterparts are constructed from them as mentioned in Section 3.2.

From Figure 6(a), we observe the SSRs of authentic and adversarial examples extracted from *Transformer* scatter differently. The reason behind this phenomenon is that *Transformer* is trained only on the authentic dataset and thus fits bad to adversarial examples, leading to huge SSR discrepancies between two types of examples.

According to Figure 6(b) and Figure 6(c), which correspond to the preliminary study in Section 2, although *Indisc-Model* reduces the SSR discrepancies between authentic and adversarial examples well compared to *Transformer*, *Switch-Model*, can further reduce the SSR discrepancies, bringing closer source sentence representations for two types of examples.

Figure 6(d) and Figure 6(e) are correspond to the analysis in Section 4.4. *Transformer-Indisc* reduces the SSR discrepancies well compared to *Transformer* and it achieves competitive results (See Section 4.3). By contrast, our model can further reduce the SSR discrepancies and achieve the best performances across all datasets (See Section 4.3), which confirms the effectiveness of our framework.

A.2 Definition of the function $R(t)$

Intuitively, a sharp increase of $R(t)$ may hurt the model optimization. We expect that $R(t)$ increases smoothly, hence we define the derivative of $R(t)$ as

$$\frac{\mathrm{d}R(t)}{\mathrm{d}t} = \frac{c_1}{R(t)}, \tag{4}$$

for some constant $c_1 \geq 0$, and $R(t)$ is a non-decreasing function. The right side of Equation 4 decreases as the training processes, which indicates the derivative of $R(t)$ gradually decreases, *i.e.*, $R(t)$ increases smoothly. Along with the constraint that $R(t) \geq 0$ for all $t \geq 0$, solving this simple differential equation, we obtain:

$$\begin{aligned} &\int R(t)\mathrm{d}R(t) = \int c_1 \mathrm{d}t \\ &\Rightarrow R(t) = \sqrt{c_1 t + c_2}, \end{aligned} \tag{5}$$

for some constants $c_1 \geq 0$ and $c_2 \geq 0$. Then, we consider the following constraints:

$$\begin{cases} R(0) = 0 \\ R(T) = R_{max}, \end{cases} \tag{6}$$

where T denotes the maximal training step number at each training stage, and R_{max} denotes the maximal noise ratio. Combining Equation 5 and 6, the final formula of $R(t)$ is rewritten as:

$$R(t) = \sqrt{R_{max}^2 \times \frac{t}{T}}. \tag{7}$$

Cross-lingual Feature Extraction from Monolingual Corpora for Low-resource Unsupervised Bilingual Lexicon Induction

Zihao Feng[†], **Hailong Cao**[†,*], **Tiejun Zhao**[†], **Weixuan Wang**[‡], **Wei Peng**[‡]
[†]Faculty of Computing, Harbin Institute of Technology
[‡]Artificial Intelligence Application Research Center, Huawei Technologies Co., Ltd
fengzihaogl@outlook.com
{caohailong, tjzhao}@hit.edu.cn
{wangweixuan2, peng.wei1}@huawei.com

Abstract

Despite their progress in high-resource language settings, unsupervised bilingual lexicon induction (UBLI) models often fail on corpora with low-resource distant language pairs due to insufficient initialization. In this work, we propose a cross-lingual feature extraction (CFE) method to learn the cross-lingual features from monolingual corpora for low-resource UBLI, enabling representations of words with the same meaning leveraged by the initialization step. By integrating cross-lingual representations with pre-trained word embeddings in a fully unsupervised initialization on UBLI, the proposed method outperforms existing state-of-the-art methods on low-resource language pairs (EN-VI, EN-TH, EN-ZH, EN-JA). The ablation study also proves that the learned cross-lingual features can enhance the representational ability and robustness of the existing embedding model.

1 Introduction

Bilingual Lexicon Induction (BLI) has aroused great interest in the NLP research frontier. BLI aims to induce word translation pairs by aligning word embeddings trained independently from monolingual corpora. BLI has contributed to many NLP tasks, including unsupervised machine translation (Artetxe et al., 2018b), cross-lingual dependency parsing (Guo et al., 2015) and cross-lingual information retrieval.

Unsupervised BLI has achieved reasonable results compared with semi-supervised works in high-resource language settings, in which adversarial training were used in Lample et al. (2018); Zhang et al. (2017). These methods focused their attention on every single word, thus ignoring the relevance between words. Artetxe et al. (2018a) proposed a method (VecMap) using a similarity matrix as an initial solution to learn the second-order structural similarity of the embeddings. Another study directly leveraged an aligned similarity matrix instead of using the embedding matrix (Alvarez-Melis and Jaakkola, 2018). Recently, Peng et al. (2021) proposed a robust refinement technique based on the ℓ_1 norm training objective. The methods above learned bilingual spaces by orthogonally projecting one monolingual space to another. Yet, evidence suggests that monolingual spaces, especially those of etymologically and typologically distant languages, are far from isomorphic (Søgaard et al., 2018; Vulić et al., 2019; Patra et al., 2019). Glavaš and Vulić (2020) relaxed the orthogonality constraint to improve the performance of BLI further. Mohiuddin et al. (2020) depicted a non-linear method using an encoder and decoder to learn the mapping in the latent space. Wang et al. (2019) proposed a joint training method using word alignments from parallel corpora as the supervision signals to align multilingual contextualized representations. While the methods mentioned above can leverage pre-trained embeddings in BLI, they lack the means to incorporate richer information like cross-lingual features from monolingual data. When it comes to low-resource and non-cognate language, the characterization capability of pre-trained embeddings is limited, which leads to the degeneration of these models.

On another strand of work, traditional works for BLI used the statistical methods to search the cross-lingual signals (Rapp, 1999; Koehn and Knight, 2002; Fung and Cheung, 2004; Gaussier et al., 2004; Haghighi et al., 2008; Vulić et al., 2011; Vulić and Moens, 2013). In recent years, E and Zhou (2022) proposed a more robust method. They formally defined the semantic embedding of words in a mathematical way instead of machine learning. However, their method is limited to non-topic words and requires high quality monolingual data to achieve good results.

The methods mentioned above have been reliant

*Corresponding author

Proceedings of the 29th International Conference on Computational Linguistics, pages 5278–5287
October 12–17, 2022.

on a high-resource language condition. Regarding low-resource and non-cognate language pairs, the characterization capability of pre-trained embeddings is limited, as only short-distance dependencies are available. In this paper, we propose a novel unsupervised method for BLI based on cross-lingual feature extraction. Furthermore, we design two ways to integrate the cross-lingual features with the pre-trained embeddings, which show complementary effects according to the experimental results. In summary, this paper makes the following contributions:

- We propose a method (CFE) to extract cross-lingual feature of each word (In Section 3.1). We expect the words with the same meaning in different languages have the similar representations and we can use it to initialize the UBLI directly.
- We propose two combination methods, embedding combination (ECB) and similarity combination (SCB), to use cross-lingual feature and pre-trained embeddings together to initialize the UBLI (In Section 3.2). The ECB method concatenates two kinds of embeddings by row, the SCB method weights the second-order similarity of pre-trained embeddings and the first-order similarity of the cross-lingual feature.
- Extensive experiments show that our method exceeds all previous unsupervised and state-of-the-art approaches on low-resource and distant language pairs. The ablation study shows that our cross-lingual feature is complementary to pre-trained embeddings. Our method improves the representational ability of the existing model (In Section 5).

2 Background

In this section, we describe the basic formulation of related supervised and unsupervised BLI methods. Let $X, Y \in \mathbb{R}^{n \times d}$ represent word embedding matrices in two languages L_1 and L_2, where n is the number of words and d is the dimension of the word embedding.

The key to supervised BLI is the parallel lexicon between two languages. Let $X^*, Y^* \in \mathbb{R}^{k \times d}$ represent parallel embedding matrices, say x_i^* in X^* is translated to y_i^* in Y^*. Mikolov et al. (2013) pointed out that a linear transformation W^* could be used to map two monolingual embeddings to a shared space.

$$W^* = \arg\min_{W \in \mathbb{R}^{d \times d}} \|X^* W - Y^*\|_F^2 \quad (1)$$

Artetxe et al. (2016) solved Problem (1) by adding an orthogonal constraint on W. Therefore, there is a closed-form solution to this problem called Procrutes: $W = UV^\top$, where U and V are defined by the SVD decomposition of $Y^\top X$.

For unsupervised BLI, embedding matrices X and Y are totally out of order. Therefore, unsupervised BLI needs a permutation matrix $P \in \mathcal{P}_n = \{0,1\}^{n \times n}$ to shuffles the row of Y:

$$\min_{W \in \mathbb{R}^{d \times d}, P \in \mathcal{P}_n} \|XW - PY\|_F^2 \quad (2)$$

Problem (2) can be solved by minimizing W and P in an iterative way. Grave et al. (2019) proposed a stochastic algorithm to initialize W and P randomly and estimate them in a joint way. However, effectively minimizing P is hard. The key to unsupervised BLI is how to solve P approximately.

Lample et al. (2018) proposed an adversarial method to initialize the initial dictionary. Artetxe et al. (2018a) has shown that two equivalent words in different languages have a similar distribution. Therefore, they initialized matrix P based on similarity matrices of monolingual embeddings $M_X = XX^\top$ and $M_Y = YY^\top$. They used the second-order similarity of pre-trained embeddings to obtain a better initial dictionary. Then they applied a self-learning strategy to iteratively compute the optimal mapping and retrieve bilingual dictionary until convergence. Very recently, Wang et al. (2019) proposed a method to jointly train word embeddings on concatenated corpora of different languages and achieved good results.

In summary, the foundation for BLI is the parallel lexicon for initialization, especially for unsupervised BLI. Therefore, a high-quality initialization is the key for unsupervised BLI.

3 Methodology

In this section, we propose a novel framework to solve the problems of UBLI in low-resource scenarios. First and foremost, the CFE is used to extract cross-lingual representations from monolingual data. Then the ECB and the SCB are used to integrate the cross-lingual features with pre-trained embeddings, either by concatenation or similarity weighting.

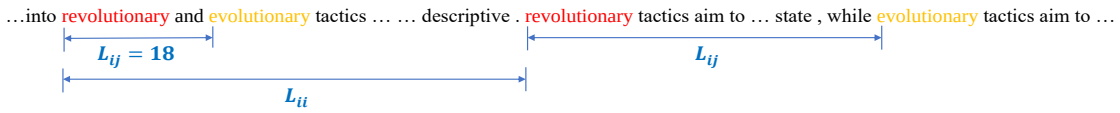

Figure 1: Counting character level distance between word patterns. In this case, Let x_i be the word "revolutionary" and x_j be the word "evolutionary". For the first L_{ij} in this example, we give the distance 18. Specifically, the distance of it is added up with 13 (the length of word "revolutionary"), 3 (the length of word "and") and 2 (the length of two space). The other L_{ij} and L_{ii} is calculated in the same way.

3.1 Cross-lingual Feature Extraction (CFE)

We propose a method to extract semantic information from monolingual data in this section. We think that although words have different symbolic representations in different languages, they all have the same language-independent textual features, which we call semantic feature. Empirically, when we are reading a novel, we understand a word based on the contextual information we can remember. So we define the semantic feature of a word based on the semantic relevance between it and its contextual words. Numerically, the semantic relevance between words are based on the character distance, which means distance counted in characters between words in a sentence. Let $x_i, x_j \in X$ represent two words in monolingual data. We define L_{ij} as the character distance between the first letter of the word x_i and x_j. When the word x_i is the same as the word x_j, the distance is defined as L_{ii}. Note that we only calculate the closest x_j after x_i. Figure 1 shows an example of L_{ij}. We count L_{ij} for each word pair in monolingual data and define n_{ij} as the number of L_{ij} (We only count those L_{ij} less than a certain threshold).

We find that semantic relevance is sensitive over short distances but degrades over long distances. For instance, when the word x_i and x_j is in different paragraphs, their semantic relevance is weak for all distances. So, we dropout some L_{ij} through a threshold and define S_{ij} to represent the semantic relevance between the word x_i and x_j:

$$S_{ij} = e^{-\frac{L_{ij}}{D}} \tag{3}$$

Where D is the hyperparameter. For every word pair, the number of their appearance also influences semantic relevance. Therefore, we weigh every S_{ij} by n_{ij}:

$$\langle S_{ij} \rangle = \frac{n_{ij} \times S_{ij}}{\sum_{k=1}^{n} n_{ik} \times S_{ik}} \tag{4}$$

Here, $\langle S_{ij} \rangle$ denotes the average of S_{ij} weighted by n_{ij}. Through this, we expect the $\langle S_{ij} \rangle$ represents strong semantic relevance. At last, for each word x_i, we extract k words with the maximum semantic relevance in set $\langle S_{i*} \rangle \cup \langle S_{*i} \rangle$, where $\langle S_{i*} \rangle$ represents all word pairs start with word x_i, and $\langle S_{*i} \rangle$ is in the same way. In this way, we get the cross-lingual feature (The representations of words with same meaning in different languages are similar) of each word in X and Y separately through monolingual data:

$$Sem\_vec_{xi} = (\langle S_{i1} \rangle, \langle S_{i2} \rangle, ..., \langle S_{ik} \rangle) \tag{5}$$

Equation (5) denotes the cross-lingual feature of a specific word x_i, where k is the hyperparameter, denotes the dimension of semantic vector.

3.2 Unsupervised initialization

Previous works VecMap (Artetxe et al., 2018a) has shown the effect of high-quality word pairs on unsupervised BLI. In this method, Let $X_{sem}, Y_{sem} \in \mathbb{R}^{n \times k}$ denote the cross-lingual feature matrices extracted by using the method in Section 3.1. We combine the cross-lingual feature to initialize unsupervised BLI in two ways, Embedding combination (ECB) and Similarity combination (SCB):

Embedding combination (ECB): We combine the pre-trained embedding with the cross-lingual feature as the initial embedding:

$$\begin{aligned} X_{com} &= X|X_{sem} \\ Y_{com} &= Y|Y_{sem} \end{aligned} \tag{6}$$

Where $|$ denotes concatenation by row, therefore $X_{com}, Y_{com} \in \mathbb{R}^{n \times (d+k)}$. We follow the method in VecMap to calculate second-order similarity matrix as the initialization. In order to maintain the process of vector alignment is comparable with the other works. We only use X_{com} and Y_{com} to do the initialization step. We continually use X and Y to do the iterative process.

Similarity combination (SCB): For our feature X_{sem} and Y_{sem} is cross-lingual, we consider to

calculate the similarity of X_{sem} and Y_{sem} directly. We combine this similarity with second-order similarity of X and Y:

$$\lambda*[(XX^\top)(YY^\top)^\top]+(1-\lambda)*(X_{sem}Y_{sem}^\top) \quad (7)$$

Where λ controls the ratio of two kinds of similarities. In this method, we can fully exploit the advantages of both kinds of embeddings.

Our method guarantees to converge to a loacl optimum base on the initial dictionary, so the quality of it is the key factor for our method. However, simply concatenating two kinds of embeddings and searching the nearest neighbor normally did not work in our preliminary experiments. It cannot guarantee to avoid our method getting stuck in poor local optima. For this reason, we propose some key improvements to make our initialization robust:

- **Embedding normalization:** The embeddings we use to be combined are trained in different ways. So, their meanings are completely different. When we combine them in a simple concatenation way, the representation ability of each embedding will be weakened. For this reason, we use a linear method ($\frac{X-Mean(X)}{Max(X)-Min(X)}$) to map two kinds of embeddings to $[-1, 1]$, then the combination method is more significant than before.

- **CSLS distance:** To extract the lexicon, we use the nearest neighbor for every word to search transformed embeddings. This phenomenon is known to occur the hubness problem (where one word is the nearest to many words) (Radovanovic et al., 2010; Suzuki et al., 2013). To avoid this hubness problem, Lample et al. (2018) modified it with the Cross-domain Similarity Local Scaling (CSLS). For two aligned embeddings x and y, they denote the set $\mathcal{N}_T(Wx)$ and $\mathcal{N}_S(y)$ of the embeddings' k nearest neighbors in the other language, respectively. Then compute $r_T(x)$ and $r_S(y)$, the average cosine similarity of $\mathcal{N}_T(Wx)$ and $\mathcal{N}_S(y)$. The CSLS score of x and y can be computed as $CSLS(x, y) = 2cos(x, y) - r_T(x) - r_S(y)$. Following the authors, we set $k = 10$.

A high-level overview of our proposed method is outlined in Algorithm 1.

Algorithm 1: CFE method for UBLI

Input: monolingual corpora L_1 and L_2
Output: parallel dictionary

```
X, Y ← pre-trained embeddings of L1, L2;
X_sem, Y_sem ← cross-lingual feature extracted from L1, L2;
Sim ← similarity matrix from ECB and SCB methods;
D ← initial word translation dictionary using Sim;
while not convergence do
    W ← linear mapping matrix calculated by Procrustes on D;
    D ← CSLS(WX, Y);
end
```

4 Experimental settings

In this section, we first list the baselines we used in Section 4.1, then we show the details of our own dataset and compare them with the MUSE dataset in Section 4.2. Finally, we show the hyperparameter settings of our methods in Section 4.3.

4.1 Baselines

We take several representative works of unsupervised BLI as our baselines. We choose the methods using pre-trained embeddings (Lample et al., 2018; Artetxe et al., 2018a; Li et al., 2020) and statistical method (E and Zhou, 2022) to be compared with our method. Especially, the method (Li et al., 2020) is the state-of-the-art model in low-resource languages and Peng et al. (2021) achieves a good results on high-resource language pairs using ℓ_1 norm optimisation on refinement. Compared with all the baselines, our method uses both two kinds of features to initialize the BLI problem. We evaluate all the baselines by using MUSE parallel data and CSLS distance to do the nearest neighbor search. We execute the publicly accessible code or reproduce the code on our own to acquire the baseline findings due to the use of our own dataset.

	EN	VI	TH	ZH	JA
words	2418	326	33	288	298
sentences	72494	10677	273	1908	9110

Table 1: Details of the dataset. We show the number (K) of words and sentences in the monolingual corpus. For the words, we count the number of different words.

Dataset	EN-VI		EN-TH		EN-ZH		EN-JA	
	→	←	→	←	→	←	→	←
MUSE	0.73	0.73	0	0.08	0.08	0	1.03	32.67
Our Dataset	0	0	0.11	0	0.08	0.28	43.80	31.64

Table 2: Results of VecMap on MUSE dataset (Lample et al., 2018) and our own dataset 4.2. We perform 10 runs for each experiment and report the average score of the accuracies (%).

Model	EN-VI		EN-TH		EN-ZH		EN-JA		avg
	→	←	→	←	→	←	→	←	
Lample et al. (2018) (MUSE)	0	0.15	0.11	0	0	0	34.52	3.56	4.79
Artetxe et al. (2018a) (VecMap)	0	0	0.11	0	0.08	0.28	43.80	31.64	9.49
E and Zhou (2022)	0	0	0	0	0	0.08	0	0	0.01
Li et al. (2020)	46.82	53.41	13.01	3.54	0.15	26.87	42.39	29.90	27.01
Peng et al. (2021)	0	0.30	0.11	0	23.98	31.91	43.50	31.95	16.47
Proposed method (ECB best dim)	48.36	23.41	15.57	3.94	26.36	33.12	**50.04**	**39.14**	29.99
Proposed method (SCB dim 50)	46.46	53.86	0.22	3.54	**27.05**	32.20	44.02	33.31	30.08
Proposed method (SCB dim 100)	45.51	**55.96**	**16.35**	3.28	26.82	**33.40**	48.85	38.08	**33.53**
Proposed method (SCB dim 200)	45.88	49.06	15.57	**6.72**	26.59	32.55	43.13	32.25	31.47
Proposed method (SCB dim 300)	**48.43**	53.26	15.57	3.67	26.82	32.62	46.10	35.12	32.70

Table 3: Results of Unsupervised method on the low-resource dataset of Section 4.2. We do 10 runs for each method and report the average score of the accuracies (%). For our proposed method, we use different dimensions of cross-lingual feature to do experiment. We show the best score of Embedding combination (ECB) method, and the score in four different dimensions of Similarity combination (SCB) method.

4.2 Datasets

We evaluate our method against baseline on the latest Wikipedia corpora. The reason why we do not use the famous dataset MUSE (Lample et al., 2018) is that we cannot get the corpus they used to train embeddings. So, we use FASTTEXT (Bojanowski et al., 2017) to train our own embeddings. However, the latest Wikipedia corpora is not on the same scale as the MUSE data. In order to ensure the comparability of models and dataset, and simulate low-resource situations, we reduce the size of corpus to match the results that the baselines on the MUSE dataset (For en:100%, vi:100%, th:10%, zh:10%, ja:30%). Our embeddings are trained based on this corpus. For the test dataset, we use the 1500 parallel lexicon of MUSE data.

4.3 Hyperparameter setting

We train our embedding using FASTTEXT with 5 epochs and 300 dims. For the hyperparameter D in Problem (3), we set D as 50. For the dimension k in Problem (4), we experiment from 50 to 300 in increments of 50. For each language pair, we chose a different λ between 0 and 1 to get the best results in Problem (7). When we do the initialization work, we only initialize 4000 words with the highest frequency in the monolingual corpus. For the iterative process, we only align the first 200000 words. We perform 10 runs for each language pair, and report the average accuracies. All the experiments are performed on a single Nvidia Titan X.

5 Experiment

In this section, we report the results obtained with our method. We first evaluate the dataset we trained in Section 5.1. Second, we present our main results in Section 5.2, thirdly we test the performance our cross-lingual feature in Section 5.3, then we do ablation tests in Section 5.4 to measure the contribution of each component and finally we compare the different initialization methods (3.2) in Section 5.5.

5.1 Comparison with MUSE dataset and our dataset

We compare the performance of our own trained dataset with the MUSE dataset (Lample et al., 2018). We use VecMap (Artetxe et al., 2018a) as

an evaluation model. For each language pair, we experiment both two datasets on VecMap and the results are in Table 2. Our dataset restores MUSE dataset as far as possible in all experimental language pairs except for one direction of EN-JA (for EN-JA pair, we have a better result than MUSE). Our dataset offers the low-resource scenario and guarantees that our model is comparable to previous models.

5.2 Main results

We report the results in the dataset of we introduced in Section 4.2 in Table 3. As it can be seen, although the baselines succeed in some high-resource languages (EN-JA), they get the degradation in the challenging low-resource language pairs. In this case, our proposed method obtains the best results in all the language pairs. For the three language pairs on which the baselines are completely degraded (only 0.28% accuracy in the best pair), our method has made significant improvements in five experiments (16.13% at least and 54.23% at most). Peng et al. (2021) perform well on some language pairs (EN-ZH, EN-JA), but still fail on the low-resource pairs. For the state-of-the-art method (Li et al., 2020), we achieve the best score on every experiment. The average score of our method is 6.52% more than SOTA.

These results confirm the robustness of the proposed method. Our method converges to a good solution in all the low-resource and distant language pairs we experiment with. In addition to being more robust, our method also obtains better accuracies compared with the previous methods by a significant improvement in all challenging language pairs. Moreover, our method is not sensitive to hyperparameters. We can get similar results in different dimensions of cross-lingual features, and most of them perform a better result than all the baselines.

Meanwhile, our method is more efficient than VecMap (Artetxe et al., 2018a). For the iterative process of our method is based on VecMap, we compare between our method with VecMap. When we optimizing the initialization dictionary's quality, our method has a faster convergence rate (824 vs. 1346 iterations for EN-VI pair). It shows that a good initialization not only leads to a better accuracy, but also speeds up convergence.

5.3 Evaluation on cross-lingual feature

We test the performance of our cross-lingual feature in this section. We first compare cross-lingual features with pre-trained features and second-order of pre-trained features. For each language pair, we choose MUSE parallel dictionary (Lample et al., 2018), which contains 5000 aligned words as our dataset. We calculate the similarities between each aligned word using three kinds of features mentioned before. The results are shown in Table 4. As can be seen, the results of our cross-lingual features are significantly better than the pre-trained feature. For the second-order similarity, which represents a cross-lingual representation, our method is 0.024 more than it. Besides, our method is more efficient and not limited by computational complexity. The results show that our feature is cross-lingual.

For the case study, we choose 8 parallel pairs of high frequency words on EN-ZH language pair and calculate the similarity of cross-lingual features between each word pair. The results are shown in Figure 2. These similarities conform to our expectations. The words having the same meaning in different languages have similar representations which are shown in Figure 2 that elements on the main diagonal have a higher score of similarity of most word pairs. In particular, our feature can accurately distinguish four word pairs in all our eight tests. On the other hand, our feature exhibit good symmetry. For the similar meaning words <*one, first*>, their similarity with all the other words is similar which means the second and third rows in Figure 2 have similar results (the same for the second and third columns). Besides, for the word pairs that have a high similarity, their corresponding rows and columns have symmetry.

5.4 Ablation test

In order to better understand the role of each part in our proposed method, we do the ablation test to separately analyze the effect of cross-lingual features, pre-trained embeddings and normalization on initialization. We use the same setting with the best score in Table 6 (SCB dim 100) for the ablation test. The obtained results are shown in Table 5.

For our cross-lingual feature extraction method, we observe that the characterization ability of our feature is better than pre-trained embeddings, and the average of it exceeds the pre-trained embeddings by 7.47%. Moreover, our feature can be used

Model	EN-VI		EN-TH		EN-ZH		EN-JA		avg
	→	←	→	←	→	←	→	←	
Pre-trained feature	0.014	0.001	0.014	0.016	0.002	0.004	0.018	0.016	0.011
Second-order similarity	0.853	0.895	0.594	0.772	0.763	0.803	0.767	0.754	0.775
Cross-lingual feature	0.872	0.882	0.661	0.803	0.758	0.827	0.798	0.792	0.799

Table 4: Results of similarities on parallel dictionary. We calculate the similarities for each parallel words and report the average score of the similarities for each language pair. We experiment on pre-trained feature, second-order similarity of pre-trained feature and cross-lingual feature.

Model	EN-VI		EN-TH		EN-ZH		EN-JA		avg
	→	←	→	←	→	←	→	←	
Full system	45.51	55.96	16.35	3.28	26.82	33.40	48.85	38.08	33.53
- Cross-lingual feature	0.07	0.07	0.22	0.26	24.90	0.14	43.73	32.40	12.72
- Pre-trained feature	0	0	14.24	3.81	26.36	33.40	49.15	36.71	20.46
- Normalization	45.30	52.66	15.68	3.15	25.12	32.48	49.00	35.64	32.38

Table 5: Ablation test on the setting: SCB method, 100 dim of cross-lingual feature. We do 10 runs for each method and report the average score of the accuracies (%) and the average accuracy score of all language pairs.

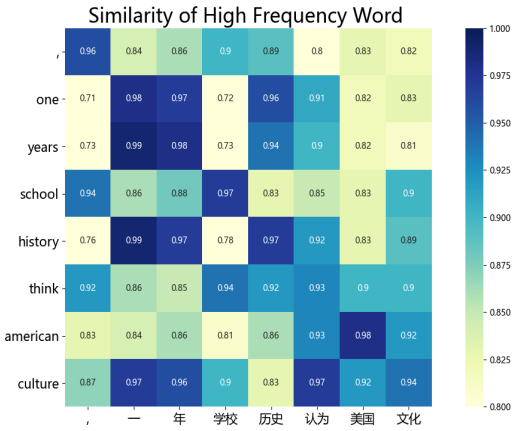

Figure 2: High frequency pairwise word similarity based on cross-lingual feature.

directly to initialize the dictionary. We do not need to calculate the second-order similarity, which reduces the problem size and computational complexity. Besides, the results show the complementary of these two kinds of features. They describe words from different dimensions. When we combined them using the two combination methods we proposed (ECB and SCB), they produce better results than either feature alone.

As for the normalization, the SCB method uses two kinds of features to compute similarity separately, and the results of regularization is not obvious. However, the ECB method uses concatenation, which is more sensitive.

5.5 Comparison with ECB and SCB method

In this section, we compare different initialization proposed in Section 3.2. As it can be seen in Table 6, the Similarity combination (SCB) method have a better result in most dimensions than Embedding combination (ECB) method. Besides, for each language pair, the best score among each dimension of SCB is better than ECB (expect EN-JA in Table 3). Especially, compared with ECB method, SCB method improves 31.95% in the reverse direction of EN-VI.

At the same time, the SCB method is less sensitive to the dimension parameters of cross-lingual features than ECB method. As we can see in Table 6, the SCB method produces stable and good results across different dimension settings, which shows that SCB method is more robust and can be adapted on more challenging problems.

Model	EN-VI		EN-TH	
	→	←	→	←
ECB dim 50	0	0.3	0	3.41
ECB dim 100	0	1.65	0.11	3.94
ECB dim 150	0.15	0.22	0.22	3.15
ECB dim 200	0.28	0.75	14.35	3.28
ECB dim 250	48.36	4.12	14.24	0.26
ECB dim 300	0.73	23.41	15.57	0
SCB dim 50	46.46	53.86	0.22	3.54
SCB dim 100	45.51	54.38	16.24	3.28
SCB dim 150	45.88	55.36	15.19	2.89
SCB dim 200	45.88	49.06	15.57	6.72
SCB dim 250	46.97	52.96	15.35	3.15
SCB dim 300	48.43	53.26	15.57	3.67

Table 6: Results of different initialization on VecMap model. We do 10 runs for each method and report the average score of the accuracies (%). For each language pair, we initial VecMap with Embedding combination (ECB) method and Similarity combination (SCB) method in 6 different cross-lingual feature dimensions.

6 Related Work

Unsupervised bilingual lexicon induction (UBLI) is an important task of machine translation. The existing methods for unsupervised bilingual lexicon induction are divided into two directions. The first is based on statistic method, and the other is based on the pre-trained embeddings. Most of the methods follow the same procedure that is to find an initial solutions and then learning a mapping method between two embedding spaces. The key of these methods is finding an initial solution.

For statistical methods, Haghighi et al. (2008) induced translations for words by using a generative model based on canonical correlation analysis, which explains the monolingual lexicons in terms of latent matchings. Vulić et al. (2011) proposed a bilingual Latent Dirichlet Allocation model for finding translations of terms in comparable corpora without using any linguistic resources. E and Zhou (2022) proposed a method in a more mathematical way. Their Markov semantic model characterized the meaning of words with language-independent numerical fingerprints.

In recent years, most methods initialized seed dictionary based on pre-trained embeddings. These methods can be divided into three categories. The first category is using adversarial methods (Lample et al., 2018; Alvarez-Melis and Jaakkola, 2018; Xu et al., 2018). They trained a generator to find a mapping between two embedding spaces and a discriminator to distinguish the mapped source embedding from the target embedding. The second category is based on the structure of embedding space. Artetxe et al. (2018a) showed the fact that two equivalent words in different languages should have a similar distribution, and used the second-order similarity of pre-trained embeddings as an initialization of UBLI. The third category is based on a non-linear mapping method. Glavaš and Vulić (2020) removed the orthogonal constraint of the mapping method. Glavaš and Vulić (2020) proposed a non-linear mapping in the latent space of two independently pre-trained autoencoders.

All these methods only use one kind of feature. Different from their methods, we leverage both monolingual corpus and word-level pre-trained embeddings to get richer information and achieve better accuracy.

7 Conclusion

In this paper, we propose a method to extract cross-lingual features through monolingual corpora, combined with pre-trained embeddings in two kinds to initial UBLI. The experiments show that our method outperforms existing state-of-the-art methods on low-resource language pairs (EN-VI, EN-TH, EN-ZH, EN-JA). The ablation study demonstrates that the induced cross-lingual features have a complementary effect to pre-trained embeddings. Besides, we also offer a MUSE-equivalent dataset with monolingual corpora.

In the future, we will develop a more robust way of extracting cross-lingual features for lexicon induction. Extending UBLI to the phrase level is also a topic of interest.

Acknowledgements

This work is supported by National Key RD Program of China under Grant 2020AAA0108000, National Natural Science Foundation of China under Grant U1908216 and Key R&D Program of Yunnan under Grant 202203AA080004. The authors would like to thank the anonymous reviewers for their helpful comments.

References

David Alvarez-Melis and Tommi Jaakkola. 2018. Gromov-wasserstein alignment of word embedding spaces. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 1881–1890.

Mikel Artetxe, Gorka Labaka, and Eneko Agirre. 2016. Learning principled bilingual mappings of word embeddings while preserving monolingual invariance. In *Proceedings of the 2016 conference on empirical methods in natural language processing*, pages 2289–2294.

Mikel Artetxe, Gorka Labaka, and Eneko Agirre. 2018a. A robust self-learning method for fully unsupervised cross-lingual mappings of word embeddings. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 789–798.

Mikel Artetxe, Gorka Labaka, Eneko Agirre, and Kyunghyun Cho. 2018b. Unsupervised neural machine translation. In *International Conference on Learning Representations*.

Piotr Bojanowski, Edouard Grave, Armand Joulin, and Tomas Mikolov. 2017. Enriching word vectors with subword information. *Transactions of the association for computational linguistics*, 5:135–146.

Weinan E and Yajun Zhou. 2022. A mathematical model for universal semantics. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 44(3):1124–1132.

Pascale Fung and Percy Cheung. 2004. Mining very-non-parallel corpora: Parallel sentence and lexicon extraction via bootstrapping and e. In *Proceedings of the 2004 Conference on Empirical Methods in Natural Language Processing*, pages 57–63.

Eric Gaussier, Jean-Michel Renders, Irina Matveeva, Cyril Goutte, and Hervé Déjean. 2004. A geometric view on bilingual lexicon extraction from comparable corpora. In *Proceedings of the 42nd Annual Meeting of the Association for Computational Linguistics (ACL-04)*, pages 526–533.

Goran Glavaš and Ivan Vulić. 2020. Non-linear instance-based cross-lingual mapping for non-isomorphic embedding spaces. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7548–7555.

Edouard Grave, Armand Joulin, and Quentin Berthet. 2019. Unsupervised alignment of embeddings with wasserstein procrustes. In *The 22nd International Conference on Artificial Intelligence and Statistics*, pages 1880–1890.

Jiang Guo, Wanxiang Che, David Yarowsky, Haifeng Wang, and Ting Liu. 2015. Cross-lingual dependency parsing based on distributed representations. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 1234–1244.

Aria Haghighi, Percy Liang, Taylor Berg-Kirkpatrick, and Dan Klein. 2008. Learning bilingual lexicons from monolingual corpora. In *Proceedings of ACL-08: Hlt*, pages 771–779.

Philipp Koehn and Kevin Knight. 2002. Learning a translation lexicon from monolingual corpora. In *Proceedings of the ACL-02 workshop on Unsupervised lexical acquisition*, pages 9–16.

Guillaume Lample, Alexis Conneau, Marc'Aurelio Ranzato, Ludovic Denoyer, and Hervé Jégou. 2018. Word translation without parallel data. In *International Conference on Learning Representations*.

Yanyang Li, Yingfeng Luo, Ye Lin, Quan Du, Huizhen Wang, Shujian Huang, Tong Xiao, and Jingbo Zhu. 2020. A simple and effective approach to robust unsupervised bilingual dictionary induction. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 5990–6001.

Tomas Mikolov, Quoc V Le, and Ilya Sutskever. 2013. Exploiting similarities among languages for machine translation.

Muhammad Tasnim Mohiuddin, M Saiful Bari, and Shafiq Joty. 2020. Lnmap: Departures from isomorphic assumption in bilingual lexicon induction through non-linear mapping in latent space. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 2712–2723.

Barun Patra, Joel Ruben Antony Moniz, Sarthak Garg, Matthew R Gormley, and Graham Neubig. 2019. Bilingual lexicon induction with semi-supervision in non-isometric embedding spaces. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 184–193.

Xutan Peng, Chenghua Lin, and Mark Stevenson. 2021. Cross-lingual word embedding refinement by ℓ_1 norm optimisation. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 2690–2701.

Milos Radovanovic, Alexandros Nanopoulos, and Mirjana Ivanovic. 2010. Hubs in space: Popular nearest neighbors in high-dimensional data. *Journal of Machine Learning Research*, 11(sept):2487–2531.

Reinhard Rapp. 1999. Automatic identification of word translations from unrelated english and german corpora. In *Proceedings of the 37th annual meeting of the Association for Computational Linguistics*, pages 519–526.

Anders Søgaard, Sebastian Ruder, and Ivan Vulić. 2018. On the limitations of unsupervised bilingual dictionary induction. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 778–788.

Ikumi Suzuki, Kazuo Hara, Masashi Shimbo, Marco Saerens, and Kenji Fukumizu. 2013. Centering similarity measures to reduce hubs. In *Proceedings of the 2013 conference on empirical methods in natural language processing*, pages 613–623.

Ivan Vulić, Wim De Smet, and Marie Francine Moens. 2011. Identifying word translations from comparable corpora using latent topic models. In *Proceedings of the 49th annual meeting of the association for computational linguistics: Human language technologies*, pages 479–484.

Ivan Vulić, Goran Glavaš, Roi Reichart, and Anna Korhonen. 2019. Do we really need fully unsupervised cross-lingual embeddings? In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing*, pages 4407–4418.

Ivan Vulić and Marie Francine Moens. 2013. A study on bootstrapping bilingual vector spaces from non-parallel data (and nothing else). In *Proceedings of the 2013 conference on empirical methods in natural language processing*, pages 1613–1624.

Zirui Wang, Jiateng Xie, Ruochen Xu, Yiming Yang, Graham Neubig, and Jaime G Carbonell. 2019. Cross-lingual alignment vs joint training: A comparative study and a simple unified framework. In *International Conference on Learning Representations*.

Ruochen Xu, Yiming Yang, Naoki Otani, and Yuexin Wu. 2018. Unsupervised cross-lingual transfer of word embedding spaces. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 2465–2474.

Meng Zhang, Yang Liu, Huanbo Luan, and Maosong Sun. 2017. Adversarial training for unsupervised bilingual lexicon induction. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1959–1970.

Language-Independent Approach for Morphological Disambiguation

Alymzhan Toleu and **Gulmira Tolegen** and **Rustam Mussabayev**
Institute of Information and Computational Technologies, 050010, 28 Shevchenko str., Almaty, Kazakhstan
alymzhan.toleu@gmail.com and gulmira.tolegen.cs@gmail.com
and rmusab@gmail.com

Abstract

This paper presents a language-independent approach for morphological disambiguation which has been regarded as extensions of POS tagging, jointly predicting complex morphological tags. In the proposed approach, all words, roots, POS and morpheme tags are embedded into vectors, and contexts representations from surface word and morphological contexts are calculated. Then the inner products between analyses and the context's representations are computed to perform the disambiguation. The underlying hypothesis is that the correct morphological analysis should be closer to the context in a vector space. Experimental results show that the proposed approach outperforms the existing models on seven different language datasets. Concretely, compared with the baselines of MarMot and a sophisticated neural model (Seq2Seq), the proposed approach achieves around 6% improvement in average accuracy for all languages while running about 6 and 33 times faster than MarMot and Seq2Seq, respectively.

1 Introduction

Morphological disambiguation (MD) is the task of jointly predicting lemma/root, part of speech (POS)(Toleu et al., 2020), and morpheme tags. For a Turkish word "yeni" (new), it can be analyzed as: 1)yen+Noun+[A3sg, Pnon, Acc]; 2)yen+Noun+[A3sg, P3sg, Nom]. If one counts analyses as tags, MD can be cast as a tagging problem with an extremely large tagset. This fact discourages direct application of the state of the art approaches designed for small fixed tagsets.

For instance, many approaches treat each analysis as a tag, and apply sequence labeling models to perform tagging (Mueller et al., 2013; Müller and Schütze, 2015; Malaviya et al., 2018). Treating each analysis as a tag leads to an oversized tagset and corresponding data-sparsity issues, which can be a concern for morphologically complex languages such as Turkish and Kazakh, where the number of morphological analyses is theoretically unlimited (Yuret and Türe, 2006). To address this problem, a sequence to sequence (Seq2Seq)(Tkachenko and Sirts, 2018) based approach was proposed, which treated each morphological analysis as a sequence of a composite tags and explicitly modeled their internal structure. This approach was inspired by the neural sequence-to-sequence models for machine translation (Cho et al., 2014). The LSTM networks were applied to model morphological analyses and context as a pair of sequences, which involves more sophisticated architectures, namely using double layers of biLSTM, one for characters and another for words. This approach was almost challenging to simplify the architecture if one wanted to keep the performance as the original Seq2Seq has. Because all types of architecture of recurrent neural networks are more fit to the nature of sequence to sequence problems (Sutskever et al., 2014).

This paper presents a language-independent MD approach that applies an uncomplicated neural architecture and obtains comparable results in accuracy and speed with the current best. A sequence of morphological analysis is an expansion sequence of its surface words with morphological information. The idea of the approach is to measure the distance between analyses and surface word context by embedding each morphological analysis and the surface word context into a single vector space. The underlying hypothesis is that the vector representation of the correct analysis should be closer to the context vector. Two types of contextual embedding for words are presented: i) surface word context; ii) morphological context, which improves the model's performance significantly. In the following, the proposed approach is referred to as language-independent morphological disambiguation (LIMD).

Our contribution amounts to the following: i) a

Proceedings of the 29th International Conference on Computational Linguistics, pages 5288–5297
October 12–17, 2022.

general language-independent approach for MD, its neural architecture is simple, decoding is fast and can be implemented easily in practice. It achieves comparable results with the current best. ii) two types of context representation are explored: word and morphological context representations.

2 Related Work

Morphological disambiguation/tagging has been studied extensively for decades, and here we review the work most relevant to this paper. We categorize the common approaches into three groups:

i) Modeling the structure of complex morphological labels with structured prediction models (Mueller et al., 2013; Müller and Schütze, 2015; Malaviya et al., 2018). The work (Mueller et al., 2013) presented a pruned CRF (PCRF) for tagging and proposed to use coarse-to-fine decoding and early updating to train the higher-order CRF. Experiments on six languages show that the PCRF gives significant improvements in accuracy. We evaluate this model on our data-sets as one of the baselines. (Müller and Schütze, 2015) compared the performance of the most important representations that can be used for across-domain MT. One of their findings is that the representations similar to Brown clusters perform best for POS tagging and that word representations based on linguistic morphological analyzers perform best for tagging. The study (Malaviya et al., 2018) combines neural networks and graphical models presented a framework for cross-lingual tagging. Instead of predicting full tag sets, the model predicts single tags separately and modeling the dependencies between tags over time steps. The model is able to generate tag sets unseen in training data, and share information between similar tag sets. This model is about cross-lingual tagging and we do not make comparisons with monolingual tagging models.

ii) Modeling complex morphological labels as sequences of morphological feature values through neural networks (NN) (Tkachenko and Sirts, 2018) and statistical approaches (Hakkani-Tur et al., 2000; Schmid and Laws, 2008). The work (Tkachenko and Sirts, 2018) presented a sequence to sequence model for tagging. The model learns the internal structure of morphological labels by treating them as sequences of morphological feature values and applies a similar strategy of neural sequence-to-sequence models commonly used for machine translation (Sutskever et al., 2014) to do tagging. The authors explored different neural architectures and compare their performance with PCRF (Mueller et al., 2013). Double layer of biLSTMs were applied in those neural architectures as Encoder (Ling et al., 2015; Labeau et al., 2015; Ma and Hovy, 2016). The encoder uses one biLSTM to compute character embedding and the second biLSTM combine the obtained character embedding along with pre-trained word embedding to generate word context embeddings. The output of those neural networks are different: one of the baselines is to use a single output layer to predict whole morphological labels. As the second baseline, the output layer can be changed to predict the different morphological value of tag with multi output layers. An improved version of the second one is to use a hierarchical multi output layers in order to capture dependencies between tags.

iii) Modeling the output of morphological analyzer as candidates then use the different classifiers to do disambiguation (Hakkani-Tur et al., 2000; Zalmout and Habash, 2017; Toleu et al., 2017). The work (Zalmout and Habash, 2017) presented an improved tagging system for Arabic by using the results of biLSTM output from words and characters and a character-aware MD model(Toleu et al., 2017) was proposed for Kazakh and Turkish. A voted-perceptron approach for Kazakh MD was proposed in the work (Tolegen et al., 2020), and explored many features impact on MD.

3 Approach

This section describes the proposed MD approach, which embeds a context and its morphological analysis into a vector space, then calculates similarity scores to rank them for performing disambiguation.

3.1 Notation

Given a sentence $(w_1, ...a_{1j}), ..., (w_n, ...a_{nj})$ consisting of n words with all possible morphological analysis a_{ij} of each word w_i, we want to predict the sequence $a_1*, ..., a_n*$ of morphological analysis which best fit to the context of the given sentence. $j \in N_i$ is the index of analyses for a word w_i. We treat a morphological analysis a_{ij} as a combination of three main constituents: root r_j, POS p_j and morpheme chain m_j. A morpheme chain m_j consists of several morphological tags, each of tags is denoted as t_{jk}, means the k-th tag in morpheme chain m_j. Vector representations of a context and a morphological analysis a_{ij} are denoted as $\mathbf{S_i}$ and

$\mathbf{M_{ij}}$, respectively. $[... \circ ... \circ ...]$ concatenation operation of inside vectors.

3.2 Morphological Embedding

For the j-th analysis a_{ij} of given word w_i, we embed its root r_j, POS p_j and morpheme tags m_j into dense vector representation. In order to handle the various length of morpheme tags, we define a value $maxT$ as the largest length of morpheme tags in the dataset. Then a vector representation for a analysis is calculated as follows[1]:

$$\mathbf{M_{ij}} = \sigma(\mathbf{W_a} * [\mathbf{r_j} \circ \mathbf{p_j} \circ \mathbf{m_j}]) \quad (1)$$

where $\mathbf{M_{ij}} \in R^{d_h \times 1}$ is a vector representation of i-th word's j-th morphological analysis. $[\mathbf{r_j} \circ \mathbf{p_j} \circ \mathbf{m_j}] \in R^{(d_r+d_p+maxT*d_m)\times 1}$ is the concatenation of corresponding vectors of root, POS and morpheme tags. $\mathbf{W_a} \in R^{d_h \times (d_r+d_p+maxT*d_m)}$ is the model parameter. d_r, d_p, d_m is the dimension of root, POS and each morpheme tag embeddings respectively. σ is a activation function. The bias term was left out for clarity. Representation for all N_i analyses of i-th word is denoted as $\mathbf{M_i} \in R^{d_h \times N_i}$

3.3 Contextual Embedding

A sentence is a sequence of surface words; its corresponding series of morphological analyses could be considered its expansion with morphological information. Two sequences are dissimilar in their formation but are similar in the language meaning. The former is made of a series of surface words, and the latter is composed of morphological analyses with certain ambiguities that depend on the context. This subsection introduce two context representations, and describe how to obtain vector representations for them:surface word context and averaged morphological context.

Surface word context. For a sentence $w_1, ..., w_n$, consider its contextual information, we want to compute surface word context representation to each word. With the purpose of simplifying the model architecture, we choose a window-based feed forward neural network as the encoder. The encoder takes a window of words and embed them to vector representation by one linear and non-linear layers:

$$\mathbf{C_i} = [\mathbf{w_{i-d_{win/2}}} \circ \mathbf{w_i} \circ \mathbf{w_{i+d_{win/2}}}] \quad (2)$$

where d_{win} is the window size and $[... \circ \mathbf{w_i} \circ ...] \in R^{(d_{win}*d_w)\times 1}$ is the concatenation of word embeddings. Here, to simplify the model architecture, we did not apply a non-linear layer to generate surface word context. It will be integrated with averaged morphological context embeddings to capture the interaction between pairs of sequences.

Averaged morphological context representation A sequence of morphological analyses is another ambiguous realization (each word has several analyses) of word series. Regardless of the ambiguities, we can compute averaged vector representations to the morphological context and apply them to handle better the dependencies issue among morpheme tags and the dependencies among analyses located in different positions of the sentence. Here, we expect the averaged morphological context to impact MD positively and will conduct corresponding experiments to find it out.

More formally, instead of only using surface word for a current word w_i, we can use the information from the previous $i \in (i-win, ..., i-1]$ words' morphological analyses as well as the next $i \in [i+1, ..., i+win)$ words' analyses. Because there are large dependencies in the morphological tags. Morphological context $\mathbf{C_{pre}} \in R^{(d_r+d_p+maxT*d_m)\times 1}$ and $\mathbf{C_{next}} \in R^{(d_r+d_p+maxT*d_m)\times 1}$ are defined by:

$$\mathbf{C_{pre}} = \sum_i \frac{1}{N_i} \sum_{j=1}^{N_i} [\mathbf{r_j} \circ \mathbf{p_j} \circ \mathbf{m_j}] \quad (3)$$

where N_i is the number of morphological analyses that i-th word has. win is the window size for a morphological context. Similar calculation goes for right side morphological context $\mathbf{C_{next}}$. The final morphological context is obtained by averaging the embedding of all analyses for the corresponding side. After obtaining all context vectors, the final vector representation for the context is calculated by concatenating three (surface, left side, and right side morphological context) and then going through a non-linear layer to extract interactive features between these contexts.

$$\mathbf{S_i} = \sigma(\mathbf{W_c} * [\mathbf{C_i} \circ \mathbf{C_{pre}} \circ \mathbf{C_{next}}]) \quad (4)$$

Where $\mathbf{W_c} \in R^{d_h \times (d_{win}*d_w+2*(d_r+d_p+maxT*d_m))}$ is the model parameter.

[1] only consider the the presence of each tag in a morpheme chain. If the number of tags in a chain less than $maxT$, after looking-up the existing tags, the remaining positions fill with zero vector.

3.4 Disambiguation

For disambiguation, we score each analysis by computing the inner product between analyses and the context's representations:

$$a_i^* = argmax(softmax(\mathbf{M_i^T} \odot \mathbf{S_i})) \quad (5)$$

where a_i^* denotes the most probable analyses for a word w_i in a context. The underlying hypothesis is that the embedding of the probable morphological analysis should be most similar to the context. The training procedure of the proposed method is given in algorithm 1.

4 Experiments

4.1 Datasets

We run experiments on Arabic-PADT (ar) (Hajič et al., 2009), Czech-PDT (cs)(Bejček et al., 2013), Spanish-AnCora (es) (Taulé et al., 2008), German-GSD (de)(McDonald et al., 2013), Russian-SynTagRus (ru) (Droganova et al., 2018), Turkish-IMST (tr)(Sulubacak et al., 2016) and Kazakh-KTB (kk)(Tyers and Washington, 2015) from Universal Dependencies version 2.3[2]. We use default data splits except for Kazakh because the default training set is significantly less than test set, we put the larger set as the training set and the less one for the test set. We tested the proposed language-independent approach on various types of language: Arabic is a Semitic language with nonconcatenative morphology. We used default Arabic script without any pre-processing. Czech and Russian are highly inflecting Slavic languages. Spanish and German belong to Romance and Germanic language groups, respectively. Kazakh and Turkish are agglutinative languages. Table 1 shows statistics of the corpora. As given, German has large ambiguous data in terms of analyses per word, it has 6.06 analyses per word on average and the maximum number of analyses reach to 51 for some certain words. It should be noted that average analyses per word are calculated based on all tokens (total number of analyses of all tokens divided by the total number of all tokens) not based on all unique tokens.

Figure 1 shows the percentage information about the number of analyses in the corpora. It can be seen that for Arabic and Russian, $20\% \sim 30\%$ tokens have two analyses and the remaining portions of tokens have analyses in the range of [3,11]. Czech and German have long-tailed distributions

[2]https://universaldependencies.org

Algorithm 1: The training and prediction process of the proposed method.

Input: $(w_1, ...a_{1j}), ..., (w_n, ...a_{nj})$, a sentence with its all possible morphological analyses of each word.
Output: $a_1^*, ..., a_n^*$, a sequence of correct morphological analysis.
Parameter : θ, the set of the model parameters.

```
for epoch ← 1 to totalEpoch do
    for i ← 1 to n do
        if N_i > 1 then
            C_pre and C_next ← use equation (3) to calculate morphological context embedding.
            S_i ← use equation (4) to compute contextual embedding.
            Define a matrix M_i ∈ R^{d_h × N_i}.
            for j ← 1 to N_i do
                M_ij ← use equation (1) to calculate j-th morphological embedding for i-th word.
            end
            a_i^* = argmax(softmax(M_i^T ⊙ S_i))
            if a_i^* ≠ the correct analysis then
                θ* ← use back-propagation to compute the gradient of the corresponding object function with respect to the model parameters.
                θ ← θ + ηθ* update parameters.
            end
        end
        else
            if i-word has only one analysis, then treat it as the correct analysis.
        end
    end
    if epoch > totalEpoch or reach the expected accuracy then
        stop training;
    end
    epoch ++;
end
```

Table 1: Corpora statistics. *avg.* denotes the average number of analyses per word. *max.* is the maximum number. *ambig. rate* denotes the percentage of the ambiguous tokens (the words have more than one analysis).

Lang.	Training Set				Test Set			
	tok.	label per word		ambig. rate (%)	tok.	label per word		ambig. rate (%)
		avg.	max.			avg.	max.	
ar	254340	2.69	12	64.88	32128	2.71	12	66.19
cs	1175374	2.65	25	48.18	174252	2.67	25	49.16
es	446145	2.80	11	53.69	52801	2.81	11	65.58
de	268414	6.06	51	62.54	16772	5.89	51	70.66
ru	871521	1.88	15	41.34	117523	1.86	15	40.60
tr	38871	1.24	5	17.40	10193	1.24	5	16.79
kk	10063	1.27	5	19.06	547	1.32	5	21.38

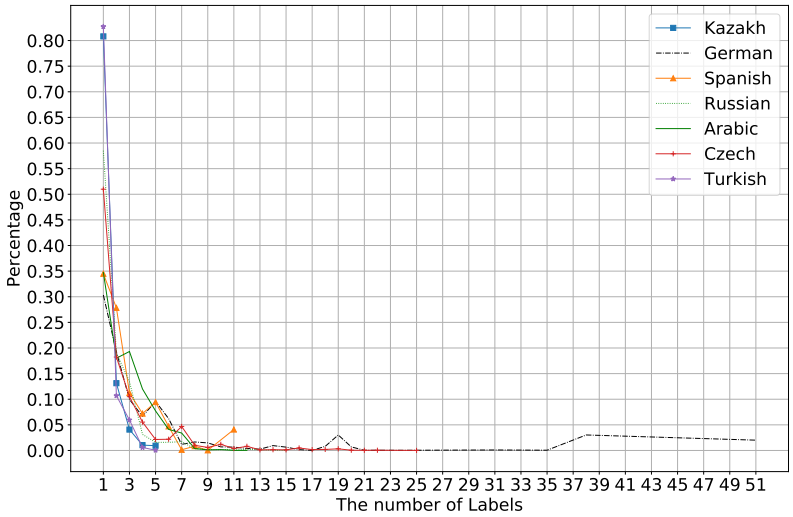

Figure 1: Distribution about the number of per word analyses. *x-axis* is the number of analyses and *y-axis* is the percentage of those analyses number in the corpus.

in the number of analyses. Kazakh and Turkish have similar distributions, and the large portion $50\% \sim 80\%$ of their analyses number are in the range [1,3).

4.2 Baselines

We use two models as baselines, the CRF-based MarMoT (Mueller et al., 2013) and Seq2Seq-based model (Tkachenko and Sirts, 2018): i) MarMoT[3] is the pruned CRF (PCRF)-based morphological tagger which has been shown to achieve competitive performance across several languages. The model is based on coarse-to-fine decoding, which means that the model first predicts POS and based on that, constrains the morphological tags. We train the second-order of MarMot following the result of (Mueller et al., 2013). ii) Seq2Seq[4] is a recent new sophisticated neural model, which is inspired from neural seq2seq models commonly used for machine translation. Encoder models the context of each word and decoder predicts morphological tags in a analysis as a sequence of its category value. Seq2Seq was trained with same hyper-parameters reported in (Tkachenko and Sirts, 2018).

4.3 Model Setup

It can be seen from Table 1 and Figure 1 that German has the most ambiguous test set, we optimize the hyper-parameters of LIMD on the German development set and apply the resulting values to other languages. We set the embedding of d_r, d_p, d_m to 35; the hidden layer size d_h is 100; the word context size is set to $d_{win} = 7$ and morphological context uses leftmost and rightmost word analyses. To compare the decoding times we run all experiments on the same test environment: In-

[3]http://cistern.cis.lmu.de/marmot/
[4]https://github.com/AleksTk/seq-morph-tagger

tel Core i7-8700 CPU with 6 cores and 16 GB of memory.

5 Results

Table 2 presents the experimental results. We report accuracy of Part-of-speech (POS), Morpheme and POS+Morpheme for all tokens. POS+Morpheme indicates that both POS and all morphological tags are correctly predicted. It can be seen from Table 2, LIMD performs comparable with MarMot (Mueller et al., 2013) and the seq2seq-based model (Seq2Seq) (Tkachenko and Sirts, 2018) in most cases for all three types of tagging. As a state-of-the-art, Seq2Seq outperforms MarMot that is a CRF-based strong baseline.

For POS, Seq2Seq and MarMot yield similar results (76.78% and 77.14%) for Kazakh such small dataset (Table 1), in contrast, the proposed approach significantly outperforms MarMot and Seq2Seq by $\approx 18\%$. Also, similar results can be observed for Turkish, the second smallest dataset in this work. LIMD outperforms baselines by $\approx 4\%$. For German and Arabic, LIMD gives above 1% improvement over baselines, and its results for Czech, Russian and Spanish datasets are slightly better than baselines.

For morpheme, LIMD gives comparable accuracy with MarMot and Seq for the most of the languages. Again, it shows promising results for the smallest (Kazakh, the improvement is $\approx 25\%$) and the second smallest (Turkish, the improv. is $\approx 7.5\%$) datasets. For German morpheme prediction, Seq2Seq (88.44%) gives 1.88% improvement over MarMot (86.56%), and LIMD yields 92.23% accuracy in this case. Compared to other languages, LIMD achieves a larger improvement over the baselines on the German data that is highest ambiguous among all datasets.

For POS+morpheme joint prediction, LIMD performs much higher than Marmot and Seq2Seq for the German, Turkish, Kazakh data, and for other languages, they give very competitive accuracies. Cross-task comparisons (morpheme vs. POS+morpheme and POS vs. POS+morpheme) reveal that the morpheme tagging is the most challenging part for all models, as it can be observed that morpheme's accuracies are much lower than POS one. It worth noting that Seq2Seq applies double-layer of biLSTM network as encoder to model the character and word embeddings for context. This architecture has been applied recently to context representation learning for MD and achieved the notable results (Heigold et al., 2017; Tkachenko and Sirts, 2018; Yu et al., 2017).

6 Analysis And Discussion

Analysis of surface word context. To explore the influence of different window-sized surface contexts, we fixed the morphological context with the leftmost and rightmost ones and tuned the window size only for the surface context. We choose the German dataset for the exploration because it is the most complex data in its ambiguous analysis in this work. Table 3 shows the results for POS, Morpheme and POS+Morpheme prediction. It can be seen that the model's accuracy grows gradually in window size (1-9), then it starts to drop slightly at window size 11, which indicates words outside of window 7 become "noise" when performing joint tagging. At window size (7,9), the model has minor differences for POS+Morpheme. Thus, we choose window size for word context to 7.

Analysis of averaged morphological context. Figure 2 shows the error rate of the training and test process for German data when incorporating two types of context embeddings: leftmost and rightmost analyses as morphological context. First, make it clear that all training curves are without markers in the figure. With markers are testing curves. In which, we present the models' performance when applying the different contexts independently: word context, left and right analyses as morphological context.

It can be seen that compared to *word* context, the *left* morphological context improves model's performance both in terms of the process for training and test. The error rate of training and test curve has a fast decrease when the model utilizes *left+right* morphological contexts compared to other settings. The model yields 84.38% (word), 85.17% (left) and 87.50% (left+right) accuracy at 25 epochs. It indicates that the morphological context plays an important role in MD. In other words, it could improve the model's performance and also reduces the training time.

Error analysis. Figure 3 shows the largest error rates of the distinct morphological categories for MarMot, Seq2Seq and LIMD models averaged over all languages. It can be seen that all models tend to have large errors for predicting the features of *Case*, *Number* and *Gender*. Among all the mod-

Table 2: Test accuracy results for POS, Morpheme and POS+Morpheme.

Lang.	POS			Morpheme			POS+Morpheme		
	Marmot	Seq2Seq	LIMD	Marmot	Seq2Seq	LIMD	Marmot	Seq2Seq	LIMD
ar	96.28	96.38	97.48	91.87	92.81	93.26	91.57	92.50	92.96
cs	98.56	98.67	98.95	93.24	94.57	94.82	92.97	94.40	94.45
es	98.25	98.17	98.40	97.79	97.56	98.00	97.11	96.83	97.30
de	92.96	93.34	94.76	86.56	88.44	92.23	81.75	83.67	88.11
ru	98.36	98.56	98.74	94.72	95.34	96.33	94.33	95.05	95.96
tr	92.99	93.66	97.66	88.42	90.47	97.04	86.20	88.15	96.03
kk	77.14	76.78	95.46	71.66	69.65	96.97	65.99	65.63	94.70

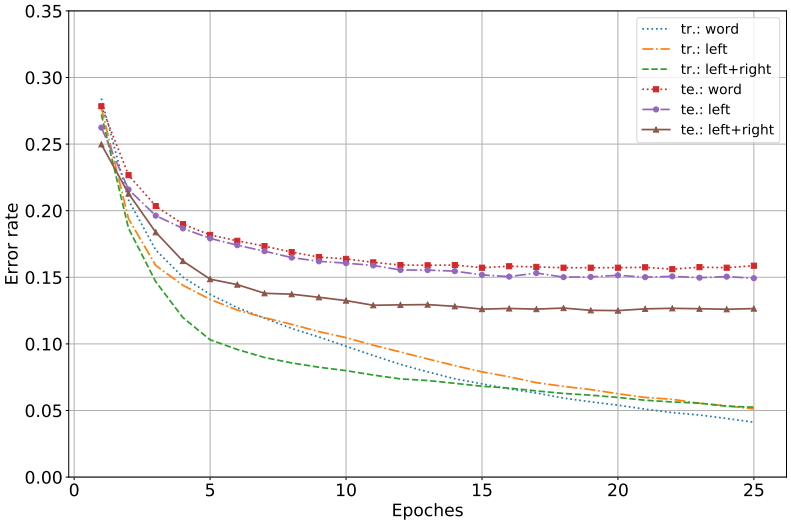

Figure 2: Example of training and test run of LIMD with two types of contexts for German data. *tr.* and *te* denote train and test. *word* - word context. *left, right* denote left and right morphological context.

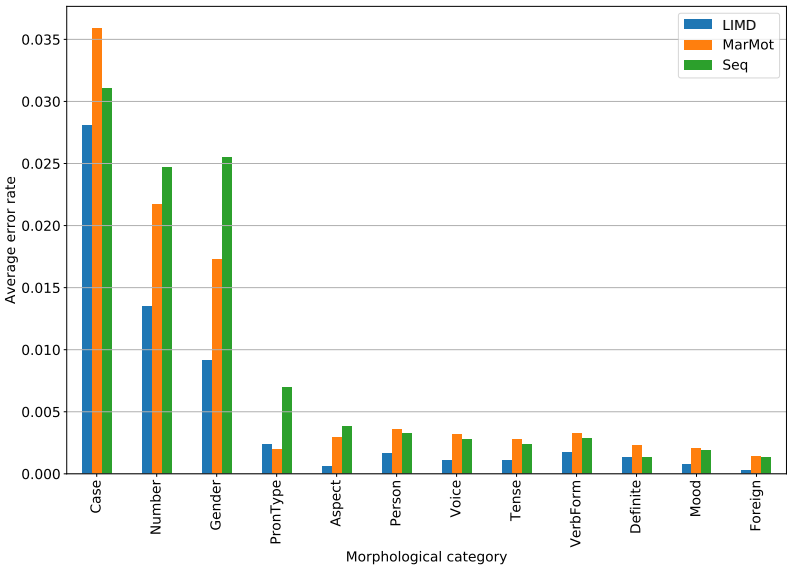

Figure 3: Average error rates of distinct morphological categories for LIMD, MarMot and Seq2Seq models.

els, it seems Seq2Seq performs worse on modeling *Number* and *Gender* features than others. It can be seen that LIMD's error rates are considerably lower in these two categories. For *Case* features,

Table 3: Test accuracy results for the German data using different window-sized surface contexts.

win	POS	Morpheme	POS+Morpheme
1	93.98	89.98	85.23
3	94.12	90.58	86.01
5	94.58	91.64	87.41
7	**94.76**	92.23	88.11
9	94.69	**92.30**	**88.15**
11	94.56	92.16	87.85

Table 4: Comparisons with previous work: Seq2Seq (Tkachenko and Sirts, 2018), Heigold (Heigold et al., 2017), Dozat(Dozat et al., 2017)

Lang.	Seq2Seq	Heigold	Dozat	LIMD
ar	**93.84**	93.78	92.85	92.96
cs	95.39	**96.32**	95.22	94.45
ru	**96.67**	96.45	96.20	95.96
tr	90.70	89.12	90.22	**96.03**
average	94.15	93.91	93.63	**94.85**

MarMot shows the largest error rates.

Comparison with previous work. It is difficult to make a direct comparison of our results to previously published results since UD data sets have various versions with differences. Here, we try to provide a very rough comparison in Table 4 only for reference. The original results were taken from (Tkachenko and Sirts, 2018) (Seq2Seq), which is obtained on UD2.1 version using a large pre-trained word embeddings[5] with sophisticated neural architecture and large well-tuned hyper-parameters. In contrast, LIMD starts by random initialization of parameters, then is tuned in the training process. Another previous tagger was presented in the work (Dozat et al., 2017), which used a more sophisticated encoder than Seq2Seq. In addition, we compare the results taken from (Heigold et al., 2017) obtained on UDv1.3. As we can see, the results are very competitive in most cases. For Turkish, LIMD shows a significant improvement.

Decoding time and accuracy. In Table 5, we report the final comparison to the baselines both in terms of accuracy and decoding time. Comparing with the baselines of MarMot and Seq2Seq, LIMD achieves around 6% gains in average accuracy for all languages, and running about 6 and 33 times faster than MarMot and Seq2Seq respectively. It can be seen from Table 5 that LIMD gains significant improvements on Kazakh, Turkish, and German datasets. The former two are the small datasets compared with other in this work, and the German is the complex one (it has around 6 analyses per word in average, see in Table 1). It may indicate that LIMD works well on morphologically complex languages with many analyses per token and the approach suffers less from the issue of lack of data.

[5]https://github.com/facebookresearch/fastText

Table 5: Comparison with the state-of-the-arts.

Lang.	POS+Morpheme		
	MarMot	Seq2Seq	LIMD
ar	91.57	92.50	92.96
cs	92.97	94.40	94.45
es	97.11	96.83	97.30
de	81.75	83.67	**88.11**
ru	94.33	95.05	95.96
tr	86.20	88.15	**96.03**
kk	65.99	65.63	**94.70**
avg.	87.13	88.03	**94.21**
tokens/s	1372 tok/s	257 tok/s	**8712 tok/s**

7 Conclusion

This paper presents a language-independent morphological disambiguation approach, LIMD. It embeds surface word and morphological context into vector representations, then calculates cosine similarly scores of two to perform disambiguation. Experimental evaluations show that LIMD outperforms other sophisticated models in both accuracy and speed. Results indicate that LIMD works well on morphologically complex languages with many analyses per token and the approach suffers less from the issue of lack of data.

Possible future work in this direction is to apply different methods to the model's output instead of a computing dot product for disambiguation. Also, there is still room for the improvement in the model's architecture, such as better capturing surface word context or modeling morphological analyses with more advanced architectures.

Acknowledgements

The work was funded by the Committee of Science of Ministry of Education and Science of the Republic of Kazakhstan under the grant AP09058174 and the grant AP08856034.

References

Eduard Bejček, Eva Hajičová, Jan Hajič, Pavlína Jínová, Václava Kettnerová, Veronika Kolářová, Marie Mikulová, Jiří Mírovský, Anna Nedoluzhko, Jarmila Panevová, Lucie Poláková, Magda Ševčíková, Jan Štěpánek, and Šárka Zikánová. 2013. Prague dependency treebank 3.0. LINDAT/CLARIAH-CZ digital library at the Institute of Formal and Applied Linguistics (ÚFAL), Faculty of Mathematics and Physics, Charles University.

Kyunghyun Cho, Bart van Merrienboer, Dzmitry Bahdanau, and Yoshua Bengio. 2014. On the properties of neural machine translation: Encoder–decoder approaches. In *Proceedings of SSST-8, Eighth Workshop on Syntax, Semantics and Structure in Statistical Translation*, pages 103–111, Doha, Qatar. Association for Computational Linguistics.

Timothy Dozat, Peng Qi, and Christopher D. Manning. 2017. Stanford's graph-based neural dependency parser at the CoNLL 2017 shared task. In *Proceedings of the CoNLL 2017 Shared Task: Multilingual Parsing from Raw Text to Universal Dependencies*, pages 20–30, Vancouver, Canada. Association for Computational Linguistics.

Kira Droganova, O. Lyashevskaya, and Daniel Zeman. 2018. Data conversion and consistency of monolingual corpora: Russian ud treebanks.

Jan Hajič, Otakar Smrž, Petr Zemánek, Petr Pajas, Jan Šnaidauf, Emanuel Beška, Jakub Kracmar, and Kamila Hassanová. 2009. Prague arabic dependency treebank 1.0. LINDAT/CLARIAH-CZ digital library at the Institute of Formal and Applied Linguistics (ÚFAL), Faculty of Mathematics and Physics, Charles University.

Diiek Z. Hakkani-Tur, Kemal Oflazer, and Gokhan Tur. 2000. Statistical morphological disambiguation for agglutinative languages. In *COLING 2000 Volume 1: The 18th International Conference on Computational Linguistics*.

Georg Heigold, Guenter Neumann, and Josef van Genabith. 2017. An extensive empirical evaluation of character-based morphological tagging for 14 languages. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 1, Long Papers*, pages 505–513, Valencia, Spain. Association for Computational Linguistics.

Matthieu Labeau, Kevin Löser, and Alexandre Allauzen. 2015. Non-lexical neural architecture for fine-grained POS tagging. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 232–237, Lisbon, Portugal. Association for Computational Linguistics.

Wang Ling, Chris Dyer, Alan W. Black, Isabel Trancoso, Ramon Fermandez, Silvio Amir, Luis Marujo, and Tiago Luis. 2015. Finding function in form: Compositional character models for open vocabulary word representation. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 1520–1530, Lisbon, Portugal. Association for Computational Linguistics.

Xuezhe Ma and Eduard Hovy. 2016. End-to-end sequence labeling via bi-directional LSTM-CNNs-CRF. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1064–1074, Berlin, Germany. Association for Computational Linguistics.

Chaitanya Malaviya, Matthew R. Gormley, and Graham Neubig. 2018. Neural factor graph models for cross-lingual morphological tagging. In *ACL*.

Ryan McDonald, Joakim Nivre, Yvonne Quirmbach-Brundage, Yoav Goldberg, Dipanjan Das, Kuzman Ganchev, Keith Hall, Slav Petrov, Hao Zhang, Oscar Täckström, Claudia Bedini, Núria Bertomeu Castelló, and Jungmee Lee. 2013. Universal Dependency annotation for multilingual parsing. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 92–97, Sofia, Bulgaria. Association for Computational Linguistics.

Thomas Mueller, Helmut Schmid, and Hinrich Schütze. 2013. Efficient higher-order CRFs for morphological tagging. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*, pages 322–332, Seattle, Washington, USA. Association for Computational Linguistics.

Thomas Müller and Hinrich Schütze. 2015. Robust morphological tagging with word representations. In *HLT-NAACL*.

Helmut Schmid and Florian Laws. 2008. Estimation of conditional probabilities with decision trees and an application to fine-grained pos tagging. In *Proceedings of the 22Nd International Conference on Computational Linguistics - Volume 1*, COLING '08, pages 777–784, Stroudsburg, PA, USA. Association for Computational Linguistics.

Umut Sulubacak, Memduh Gokirmak, Francis Tyers, Çağrı Çöltekin, Joakim Nivre, and Gülşen Eryiğit. 2016. Universal Dependencies for Turkish. In *Proceedings of COLING 2016, the 26th International Conference on Computational Linguistics: Technical Papers*, pages 3444–3454, Osaka, Japan. The COLING 2016 Organizing Committee.

Ilya Sutskever, Oriol Vinyals, and Quoc V Le. 2014. Sequence to sequence learning with neural networks. In Z. Ghahramani, M. Welling, C. Cortes, N. D. Lawrence, and K. Q. Weinberger, editors, *Advances in Neural Information Processing Systems 27*, pages 3104–3112. Curran Associates, Inc.

Mariona Taulé, M. Antònia Martí, and Marta Recasens. 2008. Ancora: Multilevel annotated corpora for catalan and spanish. In *Proceedings of the Sixth International Conference on Language Resources and Evaluation LREC'08*, Marrakech, Morocco. European Language Resources Association (ELRA).

Alexander Tkachenko and Kairit Sirts. 2018. Modeling composite labels for neural morphological tagging. In *Proceedings of the 22nd Conference on Computational Natural Language Learning*, pages 368–379, Brussels, Belgium. Association for Computational Linguistics.

Gulmira Tolegen, Alymzhan Toleu, and Rustam Mussabayev. 2020. Voted-perceptron approach for Kazakh morphological disambiguation. In *Proceedings of the 1st Joint Workshop on Spoken Language Technologies for Under-resourced languages (SLTU) and Collaboration and Computing for Under-Resourced Languages (CCURL)*, pages 258–264, Marseille, France. European Language Resources association.

Alymzhan Toleu, Gulmira Tolegen, and Aibek Makazhanov. 2017. Character-aware neural morphological disambiguation. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 666–671, Vancouver, Canada. Association for Computational Linguistics.

Alymzhan Toleu, Gulmira Tolegen, and Rustam Mussabayev. 2020. Deep learning for multilingual pos tagging. In *Advances in Computational Collective Intelligence*, pages 15–24, Cham. Springer International Publishing.

Francis M. Tyers and Jonathan N. Washington. 2015. Towards a free/open-source universal-dependency treebank for kazakh. In *3rd International Conference on Turkic Languages Processing, (TurkLang 2015)*, pages 276–289.

Xiang Yu, Agnieszka Falenska, and Ngoc Thang Vu. 2017. A general-purpose tagger with convolutional neural networks. In *Proceedings of the First Workshop on Subword and Character Level Models in NLP*, pages 124–129, Copenhagen, Denmark. Association for Computational Linguistics.

Deniz Yuret and Ferhan Türe. 2006. Learning morphological disambiguation rules for turkish. In *Proceedings of the Main Conference on Human Language Technology Conference of the North American Chapter of the Association of Computational Linguistics*, HLT-NAACL '06, pages 328–334, Stroudsburg, PA, USA. Association for Computational Linguistics.

Nasser Zalmout and Nizar Habash. 2017. Don't throw those morphological analyzers away just yet: Neural morphological disambiguation for Arabic. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 704–713, Copenhagen, Denmark. Association for Computational Linguistics.

SUN: Exploring Intrinsic Uncertainties in Text-to-SQL Parsers

Bowen Qin[1,2,✊], **Lihan Wang**[1,2,✊], **Binyuan Hui**[3,✊], **Bowen Li**[3], **Xiangpeng Wei**[3]
Binhua Li[3], **Fei Huang**[3], **Luo Si**[3], **Min Yang**[1†], **Yongbin Li**[3†]
[1] Shenzhen Institute of Advanced Technology, Chinese Academy of Sciences
[2] University of Chinese Academy of Sciences
[3] DAMO Academy, Alibaba Group
{bw.qin, lh.wang1, min.yang}@siat.ac.cn
{binyuan.hby, xiangpeng.wxp, binhua.lbh, shuide.lyb}@alibaba-inc.com

Abstract

This paper aims to improve the performance of text-to-SQL parsing by exploring the intrinsic **un**certainties in the neural network based approaches (called **SUN**). From the data uncertainty perspective, it is indisputable that a single SQL can be learned from multiple semantically-equivalent questions. Different from previous methods that are limited to one-to-one mapping, we propose a data uncertainty constraint to explore the underlying complementary semantic information among multiple semantically-equivalent questions (many-to-one) and learn the robust feature representations with reduced spurious associations. In this way, we can reduce the sensitivity of the learned representations and improve the robustness of the parser. From the model uncertainty perspective, there is often structural information (dependence) among the weights of neural networks. To improve the generalizability and stability of neural text-to-SQL parsers, we propose a model uncertainty constraint to refine the query representations by enforcing the output representations of different perturbed encoding networks to be consistent with each other. Extensive experiments on five benchmark datasets demonstrate that our method significantly outperforms strong competitors and achieves new state-of-the-art results. For reproducibility, we release our code and data at https://github.com/AlibabaResearch/DAMO-ConvAI/tree/main/sunsql.

1 Introduction

Text-to-SQL parsing (Zettlemoyer and Collins, 2012; Liang et al., 2013; Zhong et al., 2017; Qin et al., 2022) aims at converting a natural language (NL) question to its corresponding structured query language (SQL) in the context of a relational database (Schema). Although relational databases can be efficiently accessed by skilled professionals via handcrafted SQLs, a natural language interface, whose core component relies on text-to-SQL parsing, would allow ubiquitous relational data to be accessible to a broader range of non-technical users. Therefore, text-to-SQL parsing has attracted increasing attention from both academic and industrial communities recently due to its broad applications in question answering, conversational search interaction, and so on.

Although significant efforts have been devoted to text-to-SQL parsing (Chen et al., 2021; Yu et al., 2018a; Wang et al., 2020a; Cao et al., 2021; Scholak et al., 2021) with advanced deep models and architectures by learning black-box mappings between input NL questions and output SQLs, there are still several technical challenges for accurate and robust text-to-SQL parsing. First, previous models are generally learned to fit the simplified one-to-one mapping relationship, where only one NL question is used as the input, and the appropriate rest are ignored. However, there exists data uncertainty (Ott et al., 2018; Wei et al., 2020) in text-to-SQL parsing, i.e., one output SQL may correspond to multiple semantically-equivalent NL questions. At inference time, the semantic parser trained on the one-to-one parallel data struggles to deal with adequate variations of the training queries. Second, there is often structural information (dependence) (Xiao and Wang, 2019; Zhang et al., 2022) among the weights of neural networks. One challenge in training neural semantic parsers is that such models may overfit the training data since these models only seek a point estimate for their weights, failing to quantify weight (model) uncertainty. For text-to-SQL parsing, the model uncertainty brings difficulty in obtaining the encoded representations that can best describe the input data distribution and provide an robust mapping between NL questions and SQL queries.

To alleviate the aforementioned challenges, in

✊ Equal contribution.
† Corresponding authors.

Proceedings of the 29th International Conference on Computational Linguistics, pages 5298–5308
October 12–17, 2022.

this paper, we propose a generic training approach SUN, which explores the data and model uncertainties in text-to-SQL parsing. First, we propose a data uncertainty constraint to explore the underlying complementary semantic information among multiple semantically-equivalent queries and learn comprehensive feature representations with strong expressive ability. In particular, we summarize multiple semantically-equivalent source questions into a closed semantic region which is then used to complement the model to generate better SQL queries with comprehensive semantics. Second, to improve the generalizability and stability of neural text-to-SQL parsers, we propose a model uncertainty constraint to refine the query representations by enforcing the output representations of different perturbed encoding networks to be consistent with each other. Concretely, we impose the consistency on multiple networks perturbed with dropout for the same input NL question.

We summarize our main contributions as follows. (1) We propose a data uncertainty constraint, which aims to explicitly capture comprehensive semantic information among multiple semantically-equivalent NL questions, and enhance the hidden representations with this complementary information for generating better SQL, thus improving the robustness of neural text-to-SQL parsers. (2) We employ a model uncertainty constraint to encourage high similarity between the output representations of two perturbed encoding networks for the same input NL question, improving the generalizability and stability of neural text-to-SQL parsers. (3) Experiments on five benchmark datasets demonstrate that the proposed SUN method outperforms the strong competitors by a substantial margin. ***It is noteworthy that our method is model-agnostic and potentially applicable for any text-to-SQL parsers with deep network architectures.***

2 Related Work

Text-to-SQL Parsing Text-to-SQL parsing, a subtask of semantic parsing, aims at converting a NL question to its corresponding SQL query in the context of a relational database (schema). Inspired by the success of deep learning, neural text-to-SQL models based on the sequence-to-sequence (Seq2Seq) framework have dominated the research field of text-to-SQL parsing (Guo et al., 2019; Wang et al., 2020b; Zhong et al., 2020; Hui et al., 2021a,b, 2022; Cao et al., 2021; Wang et al., 2022). The general idea behind these methods is to construct an encoder to encode the input question together with related table schema and leverage a decoder to generate the target SQL based on the output of the encoder. For example, IRNet (Guo et al., 2019) is a representative neural text-to-SQL parser, which leveraged two separate BiLSTMs with self-attention mechanism (Vaswani et al., 2017) to encode the NL question and table schema. Subsequently, the graph-based approaches have been proposed, which use relational graph attention networks to deal with the schema entities and question words with structured reasoning. For instance, RATSQL (Wang et al., 2020a), SMBOP (Rubin and Berant, 2021) and RaSaP (Huang et al., 2021) defined a question-schema graph and employed the relation-aware self-attention mechanism (Shaw et al., 2018) in the encoding process to jointly learn representations of question words, schema items and edge relations. LGESQL (Cao et al., 2021) further constructed an edge-centric graph to update the edge features and designed graph pruning to determine the golden schema items related to the NL question.

Recently, some methods have leveraged the powerful pre-training capabilities of T5 (Raffel et al., 2019) to generate SQL queries. Different from graph-based methods, T5-based approaches (Scholak et al., 2021) adopt the transformer-based architecture for both encoder and decoder and do not need pre-defined graphs, schema linking relations, and grammar-based decoder. PICARD (Scholak et al., 2021) is a representative T5-based method which constrained auto-regressive decoders of language models through incremental parsing. The impressive experimental results verify the ability of the T5-based methods for text-to-SQL parsing.

Uncertainty Modeling in NLP Uncertainty quantification is an important approach to building robust AI systems. In the field of natural language processing, there are several works (Kendall et al., 2015; Xiao and Wang, 2019; Zhang et al., 2019; Shen et al., 2019; Wei et al., 2020; Zhang et al., 2021; Hu and Khan, 2021) which investigate the effects of quantifying uncertainties in various NLP tasks. For example, Zhang et al. (2019) applied a dropout-entropy method to measure uncertainty learning for text classification. Xiao and Wang (2019) showed that explicitly modeling uncertainties via Monte-Carlo dropout (Gal and Ghahramani,

2016) could enhance model performances of several NLP tasks. Su et al. (2018) introduced a series of continuous latent variables to model underlying semantics of source sentences in neural machine translation. Wang et al. (2019) proposed to quantify the confidence of NMT model predictions based on model uncertainty to better cope with noise in synthetic corpora. Wei et al. (2020) considered the intrinsic uncertainty by representing multiple source sentences into a closed semantic region. To our best knowledge, our proposed method is the first attempt to explore uncertainty in text-to-SQL parsing.

3 Preliminaries

3.1 Problem Definition

Given a natural language question Q and the corresponding database schema $S = \langle T, C \rangle$, text-to-SQL parsing aims to generate a SQL query Y based on Q and S. More specifically, the question $Q = \{q_1, q_2, \cdots, q_{|Q|}\}$ is a sequence of tokens, and the schema S consists of tables $T = \{t_1, t_2, \cdots, t_{|T|}\}$ and columns $C = \{c_1, c_2, \cdots, c_{|C|}\}$. Each table t_i contains m_t words $(t_{i,1}, t_{i,2}, \cdots, t_{i,m_t})$ and each column name $c_j^{t_i}$ in table t_i contains m_c words $(c_{j,1}^{t_i}, c_{j,2}^{t_i}, \cdots, c_{j,m_c}^{t_i})$. We use $I = \langle Q, T, C \rangle$ to denote an input for the text-to-SQL parser.

3.2 Text-to-SQL parser

Currently, advanced text-to-SQL parsers are centred around two types of approaches: the graph-based approaches (Wang et al., 2020a; Huang et al., 2021; Cao et al., 2021) and the T5-based approaches (Scholak et al., 2021; Xie et al., 2022), both of which adopt the encoder-decoder framework for implementation.

Graph-based Methods Formally, the input question and database schema are constructed as a single direct graph in the pre-processing phase: $G = \langle V, E \rangle$, where $V = Q \cup T \cup C$ denotes the node set that contains three different node types (question, table, and column) and E is the edge set depicting *pre-existing* relations for question tokens and schema items. To obtain the initial representation for every node in the graph, the recent graph-based methods, *e.g.*, LGESQL (Cao et al., 2021), first flatten all question words and schema items into a sequence and feed the sequence $I = \langle Q, T, C \rangle$ into large-scale pre-trained language models (PLMs) to learn word vectors. The learned word vectors are then passed into a subword attentive pooling layer and three Bi-LSTMs according to the node types to get the node representations for the graph G.

After that, the graph-based approaches adopt an encoder, which consists of a stack of relational graph attention network (RGAT) (Wang et al., 2020b) layers, to learn complex interaction over schema items as well as question words and output the final contextual representation X_I for input I.

In the decoding process, the graph-based methods usually adopt the grammar-based syntactic neural decoder to generate the abstract syntax tree (AST) of the target query Y in the depth-first traversal order. The output at each decoding timestep is either (i) an `APPLYRULE` action that expands the current non-terminal node in the partially generated AST, or (ii) an `SELECTTABLE` or `SELECTCOLUMN` action that chooses certain schema item. The readers can refer to (Wang et al., 2020a) for more implementation details.

T5-based Methods The T5-based approaches leverage the powerful pre-training capabilities of T5 to generate SQL queries, which directly fine-tune the downstream corpora with the standard objective for text generation without any specifically designed modules. Specifically, they take the sequential concatenation of the question words and schema item names as input to the T5 encoder and generate the corresponding SQL using the T5 decoder. Both encoder and decoder are composed of multi-layer Transformer blocks.

To ensure that the output SQL is grammatically correct, T5-based approaches such as PICARD (Scholak et al., 2021) implement rule-based constrained decoding, achieving competitive performance with less invasiveness and better compatibility. PICARD is an incremental parsing method for constrained decoding and can be compatible with any existing auto-regressive language model decoder and vocabulary—including, but not limited to, those of large pre-trained transformers. Different from the graph-based approaches that generally restrict the auto-regressive decoding process to tokens that can correctly parse to abstract syntax trees during training, constrained decoding used in T5-based approaches operates directly on the output of the language model and applies the characteristic of target SQL to help reject inadmissible tokens at each decoding step at inference time.

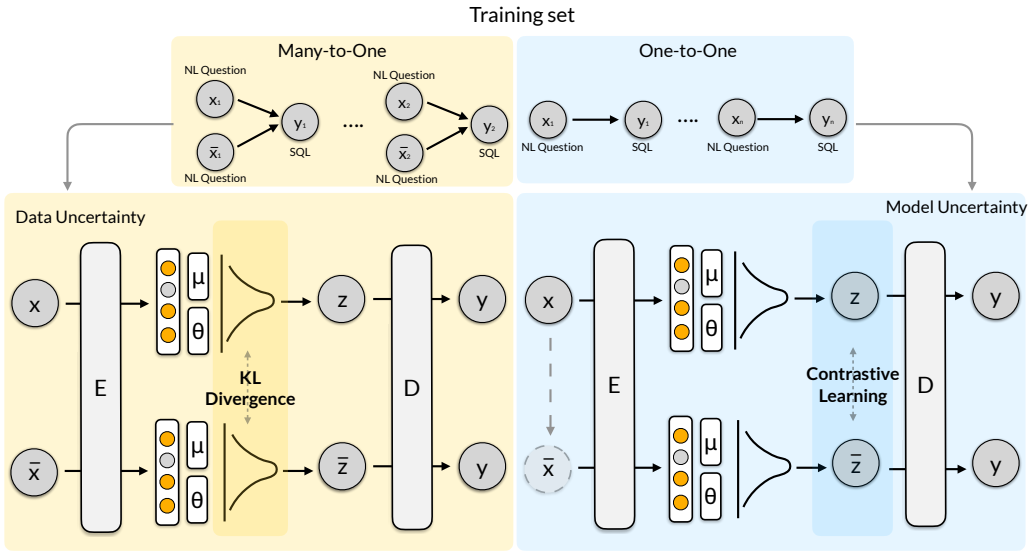

Figure 1: Overview of the proposed SUN method. There are many-to-one and one-to-one cases in the text-to-SQL training data. For many-to-one samples, they are modelled by data uncertainty, and for one-to-one samples, they are modelled by model uncertainty. The E means the encoder and D means the decoder of the graph-based or T5-based methods.

4 Proposed Method

Intrinsic Uncertainty Previous works have focused on the inductive bias through model design, while the intrinsic data uncertainty and model uncertainty are underexplored. As shown in Figure 1, **many-to-one** cases (multiple NL questions can correspond to one SQL query) exist in the text-to-SQL training data, which are caused by the inherent uncertainty of natural language (Ott et al., 2018; Wei et al., 2020). In such **many-to-one** scenario, each input sample I has a semantically-equivalent sample $\overline{I}$ in the training set, where these two input instances correspond to the same SQL query. In addition, there are also some instances in the training set that do not have semantically-equivalent samples (referred to **one-to-one** cases), where the model uncertainty becomes critical (Xiao and Wang, 2019; Zhang et al., 2019). In this work, we propose SUN, a generic training approach for text-to-SQL parsing, to mitigate the data uncertainty and model uncertainty simultaneously. SUN consists of two primary components: (i) a data uncertainty constraint that summarizes multiple semantically-equivalent NL questions into an abstract semantic region and (ii) a model uncertainty constraint that encourages high similarity between the output representations of two perturbed encoding networks for the same NL question.

Global Semantic Representation Learning Without the loss of generality, we use the encoder of the text-to-SQL parser to transform each input I into a contextual representation X_I. Then, we pass the contextual representation X_I to a convolutional layer with a pooling operation to obtain the final global semantic representation H_I of the input I. For Simplicity, we represent the whole process of learning global semantic representations as $H_I = \texttt{Encode}(I)$.

4.1 Data Uncertainty Constraint

For the many-to-one scenario, we expect to encourage high similarity between the representations of two semantically-equivalent NL questions. To this end, we propose a data uncertainty constraint to summarize the semantically-equivalent input I and $\overline{I}$ into a closed semantic region. We first obtain global semantic representations $H_I = \texttt{Encode}(I)$ and $H_{\overline{I}} = \texttt{Encode}(\overline{I})$ for two semantically-equivalent input samples I and $\overline{I}$. To learn compact and abstract representation without spurious association, we assume that there are two latent semantic variables $\mathbf{z}$ and $\overline{\mathbf{z}}$ associated with H_I and $H_{\overline{I}}$ respectively, which are sufficient for generating the target SQL query and eliminate the redundant details, so as to reduce the sensitivity of the learned representations. That is, both $\mathbf{z}$ and $\overline{\mathbf{z}}$ maintains all information which is shared by H_I and $H_{\overline{I}}$, inspired by the intuition that two semantically-

equivalent samples provide the same predictive information. Formally, we assume that $\mathcal{P}(\mathbf{z}|H_I)$ and $\mathcal{P}(\overline{\mathbf{z}}|H_{\overline{I}})$ are the distributions of the latent semantic variables $\mathbf{z}$ and $\overline{\mathbf{z}}$, which are modeled by Normal distributions parametrized with (μ,$\bar{\mu}$) and (σ, $\bar{\sigma}$). Formally, we define the two semantic distributions as follows:

$$\mathcal{P}(\mathbf{z} \mid H_I) \sim \mathcal{N}(\mu(H_I), \sigma^2(H_I)\mathbf{I}) \quad (1)$$

$$\mathcal{P}(\overline{\mathbf{z}} \mid H_{\overline{I}}) \sim \mathcal{N}\left(\bar{\mu}(H_{\overline{I}}), \bar{\sigma}^2(H_{\overline{I}})\mathbf{I}\right) \quad (2)$$

where $\mathbf{I}$ denotes the all-ones vector. μ (or $\bar{\mu}$) and σ (or $\bar{\sigma}$) are computed via fully-connected neural networks based on the global semantic representations H_I (or $H_{\overline{I}}$) as:

$$\mu(H_I) = H_I \cdot \mathbf{W}_\mu + \mathbf{b}_\mu \quad (3)$$

$$\log \sigma^2(H_I) = H_I \cdot \mathbf{W}_\sigma + \mathbf{b}_\sigma \quad (4)$$

$$\bar{\mu}(H_{\overline{I}}) = H_{\overline{I}} \cdot \mathbf{W}_\mu + \mathbf{b}_\mu \quad (5)$$

$$\log \bar{\sigma}^2(H_{\overline{I}}) = H_{\overline{I}} \cdot \mathbf{W}_\sigma + \mathbf{b}_\sigma \quad (6)$$

where $\mathbf{W}_\mu$, $\mathbf{W}_\sigma$ denote the projection parameters and $\mathbf{b}_\mu$, $\mathbf{b}_\sigma$ denote the bias terms.

Inspired by the reparameterization techniques used in (Kingma et al., 2014; Zhang et al., 2016), the sampled latent representations z and $\overline{z}$ for $\mathbf{z}$ and $\overline{\mathbf{z}}$ are then obtained as follows for efficient gradient computation :

$$z = \mu(H_I) + \sigma(H_I) \odot \epsilon \quad (7)$$

$$\overline{z} = \bar{\mu}(H_{\overline{I}}) + \bar{\sigma}(H_{\overline{I}}) \odot \epsilon \quad (8)$$

where $\epsilon \sim \mathcal{N}(0, \mathbf{I})$ is a parameter to introduce noise and $\odot$ denotes an element-wise product.

The goal of the data uncertainty constraint is to summarize multiple source questions that share the same meaning into a closed semantic region. *In order to enforce the distribution $\mathcal{P}(\mathbf{z}|H_I)$ to be close to $\mathcal{P}(\overline{\mathbf{z}}|H_{\overline{I}})$, we formulate the data uncertainty constraint loss $\mathcal{L}_{\mathbf{DU}}$ by minimizing KL divergence between $\mathcal{P}(\mathbf{z}|H_I)$ and $\mathcal{P}(\overline{\mathbf{z}}|H_{\overline{I}})$ as*:

$$\mathcal{L}_{\mathbf{DU}} = \mathbf{KL}\left(\mathcal{P}(\overline{\mathbf{z}} \mid H_{\overline{I}}) || \mathcal{P}(\mathbf{z} \mid H_I)\right) \quad (9)$$

4.2 Model Uncertainty Constraint

To improve the robustness of text-to-SQL parsers, we consider the model uncertainty for the one-to-one questions which do not have semantically-equivalent questions in training set by taking advantage of the data augmentation technique. Specifically, the input data I goes through the forward pass of the encoding network twice with dropout to produce two-view representations of the input I. Since the dropout (Hinton et al., 2012) operator randomly drops units from the model, the two forward passes indeed produce two distinct semantic representations of input I. Inspired by the model uncertainty learning in text feature space (Zhang et al., 2019), we then use contrastive learning (Gao et al., 2021; Yan et al., 2021) to pull together the two-view representations of the same input question produced by dropout and push apart the semantic representations of different questions in the same batch. *Formally, we formulate the model uncertainty constraint loss $\mathcal{L}_{\text{MU}}$ as*:

$$\mathcal{L}_{\text{MU}} = -\frac{1}{N}\sum_{i=1}^{N} \log \frac{e^{s(z_i^1, z_i^2)}}{e^{s(z_i^1, z_i^2)} + \sum_{j=1, j\neq i}^{N} e^{s(z_i^1, z_j^1)}} \quad (10)$$

where s denotes a cosine similarity function $s(z_i^1, z_i^2) = z_i^1 \cdot z_i^2 / \|z_i^1\| \|z_i^2\|$, the superscript in z_i^1 and z_i^2 indicates the view index, N indicates the number of training samples in a mini-batch. The model uncertainty constraint encourages the consistency between different views of semantic representation from the same input while enforcing the discrepancy between unrelated question pairs. By reducing the gap between the sub-models that contain different weight correspondences due to the randomness of the dropout mechanism, the robustness of the representation of the text-to-SQL parsers can be further enhanced.

4.3 Uncertainty-aware Semantic Representation

We further augment the contextual representation X_I of the input sample I obtained by the encoder with the corresponding uncertainty-aware latent representation z through the learnable gate g as:

$$g = \text{sigmoid}\left(z \cdot \mathbf{W}_z + X_I \cdot \mathbf{W}_x\right) \quad (11)$$

where $\mathbf{W}_z$, $\mathbf{W}_x$ denote the projection parameters. Then, we formulate the overall semantic representation $\mathbf{U} \in \mathbb{R}^{|V^n| \times d}$ by combining the contextual representation X_I and the uncertainty-aware latent representation z with the corresponding gate g as:

$$\mathbf{U} = \text{LayerNorm}\left(g \cdot z + (1 - g) \cdot X_I\right) \quad (12)$$

where $\mathbf{U}$ is used as the input to the decoder (grammar-based decoder or T5 decoder) of the text-to-SQL parser to output the target SQL as $Y = \texttt{Decode}(\mathbf{U})$. We define the standard training objective of the encoder-decoder framework as $\mathcal{L}_{\text{T2S}}$.

Model	EM	EX
IRNet + BERT	61.9	-
RAT-SQL + BERT	69.7	-
RAT-SQL + Grappa	73.4	-
GAZP + BERT	59.1	53.5
BRIDGE + BERT	65.5	59.9
BRIDGE v2 + BERT	70.0	68.3
SMBOP + GRAPPA	69.5	71.1
RAT-SQL+GAP+NatSQL	68.7	73.3
LGESQL + ELECTRA	75.1	-
w/ SUN	**76.8**	-
PICARD + T5-Large	69.1	72.9
w/ SUN	71.6	**75.4**

Table 1: Exact match (EM) and execution (EX) accuracy (%) on SPIDER benchmark.

Model	Dev.		Test
	EM	EX	EX
ETA + BERT	50.10	68.30	54.10
ALIGN	43.70	62.10	50.10
w/ BERT	48.40	67.70	54.30
w/ RoBERTa	50.93	70.92	58.37
w/ SUN	**52.93**	**71.95**	**59.34**

Table 2: Exact match (EM) and execution (EX) accuracy (%) on SQUALL dataset.

Joint Training Finally, we combine the standard negative likelihood loss $\mathcal{L}_{\text{T2S}}$ for SQL generation and the two uncertainty constraint loss functions ($\mathcal{L}_{\text{DU}}$ and $\mathcal{L}_{\text{MU}}$) to form the joint loss function $\mathcal{L}_{\text{total}}$ as follows:

$$\mathcal{L}_{\text{total}} = \mathcal{L}_{\text{T2S}} + \mathcal{L}_{\text{DU}} + \mathcal{L}_{\text{MU}} \quad (13)$$

5 Experimental Setup

5.1 Datasets

We conduct extensive experiments on five benchmark datasets for text-to-SQL parsing. (1) **SPIDER** (Yu et al., 2018b) is a large-scale cross-domain zero-shot text-to-SQL benchmark. It originally contains 8659 training examples across 146 databases in total. We follow the common practice to report the exact match accuracy and execution accuracy. (2) **SYN** (Gan et al., 2021a) is a challenging variant of SPIDER, which consists of 1034 evaluation examples. SYN is constructed by manually modifying NL questions in SPIDER using synonym substitution. (3) **DK** (Gan et al., 2021b) is constructed by selecting 535 samples from SPIDER dev set, where 270 pairs are the original SPIDER samples while the rest 265 pairs are modified by incorporating the domain knowledge. (4) **REALISTIC** (Deng et al., 2020) is a more realistic and challenging evaluation setting with explicit mentions of column names being manually removed. (5) **SQUALL** (Shi et al., 2020) is constructed by generating SQL queries of the English-language questions in `WIKITABLEQUESTIONS` (Pasupat and Liang, 2015) and manually align questions with corresponding SQLs. It consists of 15,622 examples which are split into training (9,032), development (2,246) and test (4,344) sets. All datasets employed in this paper are in English.

Model	SYN	DK	REALISTIC
GNN	23.6	26.0	-
IRNet	28.4	33.1	-
RAT-SQL	33.6	35.8	-
RAT-SQL + BERT	48.2	40.9	58.1
RAT-SQL + Grappa	49.1	38.5	59.3
LGESQL + ELECTRA	64.6	48.4	69.2
w/ SUN	**66.9**	**52.7**	**70.9**

Table 3: Exact match accuracy (%) on SYN, DK and REALISTIC benchmark.

5.2 Baseline Methods

We compare SUN with several strong baseline methods, including IRNet (Guo et al., 2019), RAT-SQL (Wang et al., 2020a), GAZP (Zhong et al., 2020), BRIDGE (Lin et al., 2020), SMBOP (Rubin and Berant, 2021), LGESQL (Cao et al., 2021) and PICARD (Scholak et al., 2021). Since SUN is model-agnostic and potentially applicable for any neural text-to-SQL parsers, we adopt LGESQL and PICARD, which are the state-of-the-art graph-based and T5-based methods respectively, as our base models to verify the universality of SUN. In addition, for SQUALL, we adopt the previous SOTA model ALIGN (Shi et al., 2020) with RoBERTa (Devlin et al., 2018) and ETA (Liu et al., 2021) as our base model.

5.3 Implementation Details

For the LGESQL, following (Cao et al., 2021), the hidden size of the graph attention network is set to 512 and the number of layers is set to 8. The number of heads in multi-head attention is 8 and the dropout rate is set to 0.2 for both the encoder and decoder. In the decoder, the dimension of hidden

Model	SPIDER					SYN				
	easy	medium	hard	extra	all	easy	medium	hard	extra	all
LGESQL+ELECTRA	91.9	78.3	64.9	**52.4**	75.1	79.4	67.9	**62.1**	36.1	64.6
w/ SUN	**92.3**	**80.3**	**70.7**	50.6	**76.8**	**79.8**	**72.3**	59.9	**41.4**	**66.9**

Model	DK					REALISTIC				
	easy	medium	hard	extra	all	easy	medium	hard	extra	all
LGESQL+ELECTRA	74.5	46.7	41.9	29.5	48.4	86.2	**77.8**	60.6	41.2	69.2
w/ SUN	**75.5**	**53.7**	**47.3**	**30.5**	**52.7**	**89.9**	77.3	**61.6**	**45.4**	**70.9**

Table 4: Exact matching accuracy by varying the levels of difficulty of the inference data on the of SPIDER, SYN, DK and REALISTIC.

Model	**SPIDER**	**SYN**	**DK**	**REALISTIC**
LGESQL+SUN	76.8	66.9	52.7	70.9
w/o $\mathcal{L}_{\text{DU}}$	75.8	65.8	51.2	69.5
w/o $\mathcal{L}_{\text{MU}}$	76.2	66.4	52.1	70.1

Table 5: Ablation results in terms of exact match accuracy on SPIDER, SYN, DK and REALISTIC.

state, action embedding and node type embedding are set to 512, 128 and 128, respectively. We use AdamW optimizer (Loshchilov and Hutter, 2017) with linear warmup scheduler and the warmup ratio of total training steps is 0.1. The learning rate is 1e-4 and the weight decay rate is 0.1. The batch size is set to be 20 and the maximum gradient norm is 5. The number of training epochs is 200. For the PICARD (Scholak et al., 2021), we follow the official implementation to fine-tune T5-large for 400 epochs. We use Adafactor optimizer (Shazeer and Stern, 2018) with a learning rate of 1e-4. For the ALIGN model, we employ SUN on the RoBERTa setting. The representation size is 1024. The decoder is implemented with 2-layer LSTM and the hidden size is set to 128. We adopt the Adam optimizer (Kingma and Ba, 2014) with learning rate of 0.001 and the dropout rate is set to be 0.3. Specifically, we employ kernels with window sizes ranging from 3 to 5 to obtain the global semantic representations used in the uncertainty measurement process in SUN.

6 Experimental Results

6.1 Main Results

Results on SPIDER Table 1 shows the exact match accuracy (EM) and execution accuracy (EX) scores of our method and compared baselines on the SPIDER dataset. We observe that SUN can bring substantial improvements, which achieves a notable gain of 1.7% on the exact match accuracy scores over the strongest baseline LGESQL. In addition, PICARD+SUN obtains significantly better results than the compared baseline methods on SPIDER dataset in terms of exact match accuracy and execution accuracy, which both achieve superior results of 2.5% improvement.

Results on more challenging and realistic settings To evaluate the effectiveness of SUN in more challenging and realistic settings, Table 3 illustrates the experimental results on SYN, DK and REALISTIC datasets. We can observe that SUN consistently and substantially surpasses the compared models by a noticeable margin on three datasets in terms of exact match accuracy. In particular, SUN achieves considerable improvement over LGESQL on all three datasets. For SYN, LGESQL with SUN outperforms LGESQL by 2.3% EM score, demonstrating that SUN can improve the robustness of text-to-SQL parsers to synonym substitution. An improvement of 4.3% EM score is observed on DK benchmark, verifying that SUN contributes to the generalization ability of text-to-SQL parsers to unseen domains. Furthermore, under the challenging REALISTIC setting where all explicit mentions of column names are removed, LGESQL+SUN also achieves a strong performance (70.9%) which is 1.7% higher than LGESQL.

Results on SQUALL Table 2 shows the experimental results on SQUALL. Since SQUALL does

Case 1. (SPIDER)		
Gold	SELECT T1.template_type_code, count(*) FROM Templates AS T1 JOIN Documents AS T2 ON T1.template_id = T2.template_id GROUP BY T1.template_type_code	✓
Question	*Show all template type codes and the number of documents using each type.*	
LGESQL	SELECT T1.template_type_code, count(*) FROM Templates AS T1 JOIN Documents AS T2 ON T1.template_id = T2.template_id GROUP BY T1.template_type_code	✓
w/ SUN	SELECT T1.template_type_code, count(*) FROM Templates AS T1 JOIN Documents AS T2 ON T1.template_id = T2.template_id GROUP BY T1.template_type_code	✓
SE_Question	*What are the different template type codes, and how many documents use each type?*	
LGESQL	SELECT Templates.Template_Type_Code , COUNT(*) FROM Templates GROUP BY Templates.Template_Type_Code	✗
w/ SUN	SELECT T1.template_type_code, count(*) FROM Templates AS T1 JOIN Documents AS T2 ON T1.template_id = T2.template_id GROUP BY T1.template_type_code	✓
Case 2. (DK)		
Gold	SELECT count(*) FROM student AS T1 JOIN has_pet AS T2 ON T1.stuid = T2.stuid JOIN pets AS T3 ON T2.petid = T3.petid WHERE T1.sex = 'F' AND T3.pettype = 'dog'	
Question	*How many puppy pets are raised by female students?*	
LGESQL	SELECT count(*) FROM student AS T1 JOIN has_pet AS T2 ON T1.stuid = T2.stuid JOIN pets AS T3 ON T2.petid = T3.petid WHERE T1.sex = 'F' AND T3.pettype = 'dog'	✓
w/ SUN	SELECT count(*) FROM student AS T1 JOIN has_pet AS T2 ON T1.stuid = T2.stuid JOIN pets AS T3 ON T2.petid = T3.petid WHERE T1.sex = 'F' AND T3.pettype = 'dog'	✓
SE_Question	*Find the number of puppy pets that are raised by female students (with sex F).*	
LGESQL	SELECT COUNT(*) FROM Pets JOIN Has_Pet JOIN Student WHERE Student.Sex = 'F'	✗
w/ SUN	SELECT count(*) FROM student AS T1 JOIN has_pet AS T2 ON T1.stuid = T2.stuid JOIN pets AS T3 ON T2.petid = T3.petid WHERE T1.sex = 'F' AND T3.pettype = 'dog'	✓

Table 6: Case study: cases are sampled from SPIDER, and DK. SE indicates semantically-equivalent.

not contain target SQL queries in the test set, we merely present the execution accuracy score on test set. Overall, SUN significantly improves the performance on both the dev and test sets of SQUALL, achieving the gains of 2.0% EM score and 1.03% EX score on dev set, and 1.97% EX score on test set over the strongest baseline ALIGN+RoBERTa on SQUALL.

6.2 Results on Complex Queries

We investigate the performance of SUN on different queries. The SQL queries in the SPIDER, SYN, DK and REALISTIC benchmarks are divided into four levels (i.e., easy, medium, hard, extra hard) based on their difficulty, where the difficulty is defined based on the number of SQL components. Table 4 summarizes the results of the four benchmarks with four levels of difficulty. From the results, we can observe that SUN can boost the performance of LGESQL across almost all different difficulty levels on the four datasets. In particular, LGESQL+SUN obtains much better performance on the extremely hard samples than LGESQL by a large margin. For example, LGESQL with SUN shows 4.7%, 1.0% and 4.2% improvements on the extra hard samples in SYN, DK and REALISTIC respectively.

6.3 Ablation Study

To analyze the impact of two kinds of uncertainties in SUN, we also conduct an ablation test by discarding the data uncertainty constraint (denoted as w/o $\mathcal{L}_{\mathrm{DU}}$) and the model uncertainty constraint (denoted as w/o $\mathcal{L}_{\mathrm{MU}}$), respectively. The ablation results are summarized in Table 5. As expected, both uncertainty constraints contribute great improvements to SUN. For example, the performance of SUN w/o $\mathcal{L}_{\mathrm{DU}}$ decreases by 1.1/1.0/1.5 points on the SYN/SPIDER/DK dev sets, verifying that the data uncertainty constraint is essential for improving the model performance. In addition, the performance of SUN decreases by 0.8/0.5 points on REALISTIC/SYN evaluation sets when removing the model uncertainty constraint.

6.4 Case Study

In this section, we present two cases sampled from SPIDER and DK to demonstrate the effectiveness of SUN qualitatively. As illustrated in Table 6, we report the original questions, the semantically-equivalent variant of the original questions (SE-question), the gold SQL queries, and the SQL queries generated by LGESQL and LGESQL+SUN. From the results, we can observe that LGESQL+SUN can generate more accurate SQL queries than LGESQL. Taking the first case as an example, although both LGESQL and LGESQL+SUN can generate accurate SQL queries given the question, LGESQL fails to correctly understand its semantically-equivalent question and thus generates inappropriate SQL where the necessary mention of database schema "*Document*" is ignored. In contrast, LGESQL+SUN generates the same correct SQL for both the original question

and its semantically-equivalent question by improving the robustness of the text-to-SQL parsers with data and model uncertainty constraints. We observe similar trends on the DK dataset. Concretely, LGESQL+SUN can generate consistent SQL for two semantically-equivalent questions that have very different expressions, while LGESQL cannot handle such cases.

The effectiveness of SUN according to the observed results illustrated in Table 6 demonstrates that the exploration of underlying complementary semantic information in SUN helps text-to-SQL parsers understand the different variants of expressions and generate better SQL queries with comprehensive semantics.

7 Conclusion

In this paper, we proposed a novel SUN method to explore intrinsic uncertainties in text-to-SQL parsing. First, we devised a data uncertainty constraint to capture complementary semantic information among multiple semantically-equivalent questions and thus improve the robustness of the text-to-SQL parsers. Second, a model uncertainty constraint was leveraged to refine the representations by encouraging high similarity between the output representations of two perturbed encoding networks for the same input question. Experimental results on five benchmark datasets showed that SUN significantly outperformed the compared methods.

8 Acknowledges

My acknowledges: This work was partially supported by National Natural Science Foundation of China (No. 61906185), Youth Innovation Promotion Association of CAS China (No. 2020357), Shenzhen Science and Technology Innovation Program (Grant No. KQTD20190929172835662), Shenzhen Basic Research Foundation (No. JCYJ20210324115614039 and No. JCYJ20200109113441941).

References

Ruisheng Cao, Lu Chen, Zhi Chen, Yanbin Zhao, Su Zhu, and Kai Yu. 2021. Lgesql: Line graph enhanced text-to-sql model with mixed local and non-local relations. In *ACL*.

Zhi Chen, Lu Chen, Yanbin Zhao, Ruisheng Cao, Zihan Xu, Su Zhu, and Kai Yu. 2021. Shadowgnn: Graph projection neural network for text-to-sql parser. In *Proc. of NAACL*.

Xiang Deng, Ahmed Hassan Awadallah, Christopher Meek, Oleksandr Polozov, Huan Sun, and Matthew Richardson. 2020. Structure-grounded pretraining for text-to-sql. *arXiv preprint arXiv:2010.12773*.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. In *Proc. of NAACL*.

Yarin Gal and Zoubin Ghahramani. 2016. Dropout as a bayesian approximation: Representing model uncertainty in deep learning. In *Proc. of ICML*.

Yujian Gan, Xinyun Chen, Qiuping Huang, Matthew Purver, John Robert Woodward, Jinxia Xie, and Pengsheng Huang. 2021a. Towards robustness of text-to-sql models against synonym substitution. In *ACL*.

Yujian Gan, Xinyun Chen, and Matthew Purver. 2021b. Exploring underexplored limitations of cross-domain text-to-sql generalization. In *EMNLP*.

Tianyu Gao, Xingcheng Yao, and Danqi Chen. 2021. Simcse: Simple contrastive learning of sentence embeddings. *arXiv preprint arXiv:2104.08821*.

Jiaqi Guo, Zecheng Zhan, Yan Gao, Yan Xiao, Jian-Guang Lou, Ting Liu, and Dongmei Zhang. 2019. Towards complex text-to-sql in cross-domain database with intermediate representation. In *ACL*.

Geoffrey E Hinton, Nitish Srivastava, Alex Krizhevsky, Ilya Sutskever, and Ruslan R Salakhutdinov. 2012. Improving neural networks by preventing co-adaptation of feature detectors. *arXiv preprint arXiv:1207.0580*.

Yibo Hu and Latifur Khan. 2021. Uncertainty-aware reliable text classification. In *Proc. of KDD*.

Junyang Huang, Yongbo Wang, Yongliang Wang, Yang Dong, and Yanghua Xiao. 2021. Relation aware semi-autoregressive semantic parsing for nl2sql. *arXiv preprint arXiv:2108.00804*.

Binyuan Hui, Ruiying Geng, Qiyu Ren, Binhua Li, Yongbin Li, Jian Sun, Fei Huang, Luo Si, Pengfei Zhu, and Xiaodan Zhu. 2021a. Dynamic hybrid relation exploration network for cross-domain context-dependent semantic parsing. In *AAAI*.

Binyuan Hui, Ruiying Geng, Lihan Wang, Bowen Qin, Bowen Li, Jian Sun, and Yongbin Li. 2022. S^2SQL: Injecting syntax to question-schema interaction graph encoder for text-to-sql parsers. In *ACL*.

Binyuan Hui, Xiang Shi, Ruiying Geng, Binhua Li, Yongbin Li, Jian Sun, and Xiaodan Zhu. 2021b. Improving text-to-sql with schema dependency learning. *arXiv preprint arXiv:2103.04399*.

Alex Kendall, Vijay Badrinarayanan, and Roberto Cipolla. 2015. Bayesian segnet: Model uncertainty in deep convolutional encoder-decoder architectures for scene understanding. *arXiv preprint arXiv:1511.02680*.

Diederik P Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*.

Diederik P Kingma, Shakir Mohamed, Danilo Jimenez Rezende, and Max Welling. 2014. Semi-supervised learning with deep generative models. In *Proc. of NeurIPS*.

Percy Liang, Michael I Jordan, and Dan Klein. 2013. Learning dependency-based compositional semantics. *Computational Linguistics*.

Xi Victoria Lin, Richard Socher, and Caiming Xiong. 2020. Bridging textual and tabular data for cross-domain text-to-sql semantic parsing. *arXiv preprint arXiv:2012.12627*.

Qian Liu, Dejian Yang, Jiahui Zhang, Jiaqi Guo, Bin Zhou, and Jian-Guang Lou. 2021. Awakening latent grounding from pretrained language models for semantic parsing. In *ACL*.

Ilya Loshchilov and Frank Hutter. 2017. Decoupled weight decay regularization. *arXiv preprint arXiv:1711.05101*.

Myle Ott, Michael Auli, David Grangier, and Marc'Aurelio Ranzato. 2018. Analyzing uncertainty in neural machine translation. In *ICML*, pages 3956–3965. PMLR.

Panupong Pasupat and Percy Liang. 2015. Compositional semantic parsing on semi-structured tables. *arXiv preprint arXiv:1508.00305*.

Bowen Qin, Binyuan Hui, Lihan Wang, Min Yang, Jinyang Li, Binhua Li, Ruiying Geng, Rongyu Cao, Jian Sun, Luo Si, et al. 2022. A survey on text-to-sql parsing: Concepts, methods, and future directions. *arXiv preprint arXiv:2208.13629*.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J Liu. 2019. Exploring the limits of transfer learning with a unified text-to-text transformer. *arXiv preprint arXiv:1910.10683*.

Ohad Rubin and Jonathan Berant. 2021. Smbop: Semi-autoregressive bottom-up semantic parsing. In *Proc. of NAACL*.

Torsten Scholak, Nathan Schucher, and Dzmitry Bahdanau. 2021. Picard: Parsing incrementally for constrained auto-regressive decoding from language models. In *EMNLP*.

Peter Shaw, Jakob Uszkoreit, and Ashish Vaswani. 2018. Self-attention with relative position representations. *arXiv preprint arXiv:1803.02155*.

Noam Shazeer and Mitchell Stern. 2018. Adafactor: Adaptive learning rates with sublinear memory cost. In *Proc. of ICML*.

Aili Shen, Daniel Beck, Bahar Salehi, Jianzhong Qi, and Timothy Baldwin. 2019. Modelling uncertainty in collaborative document quality assessment. In *Proceedings of the 5th Workshop on Noisy User-generated Text (W-NUT 2019)*.

Tianze Shi, Chen Zhao, Jordan Boyd-Graber, Hal Daumé III, and Lillian Lee. 2020. On the potential of lexico-logical alignments for semantic parsing to sql queries. *arXiv preprint arXiv:2010.11246*.

Jinsong Su, Shan Wu, Deyi Xiong, Yaojie Lu, Xianpei Han, and Biao Zhang. 2018. Variational recurrent neural machine translation. In *Proc. of AAAI*.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. *Proc. of NeurIPS*.

Bailin Wang, Richard Shin, Xiaodong Liu, Oleksandr Polozov, and Matthew Richardson. 2020a. RAT-SQL: relation-aware schema encoding and linking for text-to-sql parsers. In *ACL*.

Kai Wang, Weizhou Shen, Yunyi Yang, Xiaojun Quan, and Rui Wang. 2020b. Relational graph attention network for aspect-based sentiment analysis. *arXiv preprint arXiv:2004.12362*.

Lihan Wang, Bowen Qin, Binyuan Hui, Bowen Li, Min Yang, Bailin Wang, Binhua Li, Fei Huang, Luo Si, and Yongbin Li. 2022. Proton: Probing schema linking information from pre-trained language models for text-to-sql parsing. In *KDD*.

Shuo Wang, Yang Liu, Chao Wang, Huanbo Luan, and Maosong Sun. 2019. Improving back-translation with uncertainty-based confidence estimation. *arXiv preprint arXiv:1909.00157*.

Xiangpeng Wei, Heng Yu, Yue Hu, Rongxiang Weng, Luxi Xing, and Weihua Luo. 2020. Uncertainty-aware semantic augmentation for neural machine translation. *arXiv preprint arXiv:2010.04411*.

Yijun Xiao and William Yang Wang. 2019. Quantifying uncertainties in natural language processing tasks. In *Proc. of AAAI*.

Tianbao Xie, Chen Henry Wu, Peng Shi, Ruiqi Zhong, Torsten Scholak, Michihiro Yasunaga, Chien-Sheng Wu, Ming Zhong, Pengcheng Yin, Sida I. Wang, Victor Zhong, Bailin Wang, Chengzu Li, Connor Boyle, Ansong Ni, Ziyu Yao, Dragomir Radev, Caiming Xiong, Lingpeng Kong, Rui Zhang, Noah A. Smith, Luke Zettlemoyer, and Tao Yu. 2022. Unifiedskg: Unifying and multi-tasking structured knowledge grounding with text-to-text language models. *arXiv preprint arXiv:2201.05966*.

Yuanmeng Yan, Rumei Li, Sirui Wang, Fuzheng Zhang, Wei Wu, and Weiran Xu. 2021. Consert: A contrastive framework for self-supervised sentence representation transfer. *arXiv preprint arXiv:2105.11741*.

Tao Yu, Zifan Li, Zilin Zhang, Rui Zhang, and Dragomir R. Radev. 2018a. Typesql: Knowledge-based type-aware neural text-to-sql generation. In *Proc. of NAACL*.

Tao Yu, Rui Zhang, Kai Yang, Michihiro Yasunaga, Dongxu Wang, Zifan Li, James Ma, Irene Li, Qingning Yao, Shanelle Roman, Zilin Zhang, and Dragomir R. Radev. 2018b. Spider: A large-scale human-labeled dataset for complex and cross-domain semantic parsing and text-to-sql task. In *EMNLP*.

Luke S Zettlemoyer and Michael Collins. 2012. Learning to map sentences to logical form: Structured classification with probabilistic categorial grammars. *arXiv preprint arXiv:1207.1420*.

Biao Zhang, Deyi Xiong, Jinsong Su, Hong Duan, and Min Zhang. 2016. Variational neural machine translation. In *EMNLP*.

Chen Zhang, Lei Ren, Fang Ma, Jingang Wang, Wei Wu, and Dawei Song. 2022. Structural bias for aspect sentiment triplet extraction. In *COLING*.

Linhai Zhang, Chao Lin, Deyu Zhou, Yulan He, and Meng Zhang. 2021. A bayesian end-to-end model with estimated uncertainties for simple question answering over knowledge bases. *Computer Speech & Language*.

Xuchao Zhang, Fanglan Chen, Chang-Tien Lu, and Naren Ramakrishnan. 2019. Mitigating uncertainty in document classification. *arXiv preprint arXiv:1907.07590*.

Victor Zhong, Mike Lewis, Sida I Wang, and Luke Zettlemoyer. 2020. Grounded adaptation for zero-shot executable semantic parsing. *arXiv preprint arXiv:2009.07396*.

Victor Zhong, Caiming Xiong, and Richard Socher. 2017. Seq2sql: Generating structured queries from natural language using reinforcement learning. *arXiv preprint arXiv:1709.00103*.

Deciphering and Characterizing Out-of-Vocabulary Words for Morphologically Rich Languages

Georgie Botev, Arya D. McCarthy, Winston Wu, and **David Yarowsky**
Johns Hopkins University

Abstract

This paper presents a detailed foundational empirical case study of the nature of out-of-vocabulary words encountered in modern text in a moderate-resource language such as Bulgarian, and a multi-faceted distributional analysis of the underlying word-formation processes that can aid in their compositional translation, tagging, parsing, language modeling, and other NLP tasks. Given that out-of-vocabulary (OOV) words generally present a key open challenge to NLP and machine translation systems, especially toward the lower limit of resource availability, there are useful practical insights, as well as corpus-linguistic insights, from both a detailed manual and automatic taxonomic analysis of the types, multidimensional properties, and processing potential for multiple representative OOV data samples.

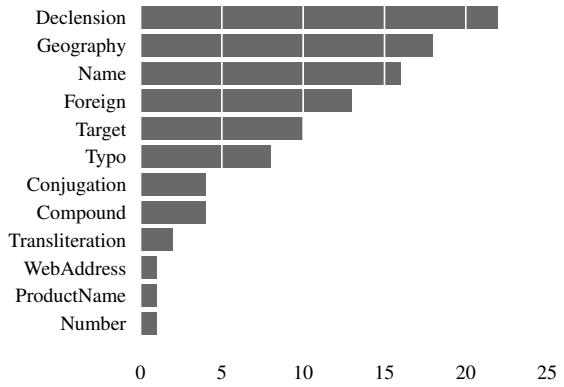

Figure 1: Taxonomized distribution of out-of-vocabulary types in Bulgarian Wikipedia, random sample of 100 types

1 Introduction

Even in a familiar language, unfamiliar words cause trouble for machine processing or comprehension of text. Any dictionary is innately incomplete in its coverage, unable to provide novel coinages and exhaustive forms. Without finding the word in a dictionary, the surface form and context afford only weak evidence for its meaning. The situation is even worse for languages other than English, especially morphologically rich languages, for two reasons: first, there is usually less annotated data available; and second, the coverage of such data is much lower due to the high number of different forms. Moreover, many words not found in even a small training corpus are in fact related to quite common words by processes such as inflection, derivation, compounding, or misspelling.

In the work described herein, we therefore concentrate on the problem of characterizing unknown words in terms of the processes by which they arise, and especially the relative frequencies at which such processes occur. This informs us of the *distribution* of out-of-vocabulary (OOV) words with respect to different dictionary sources.

To do so, we conduct a study on a sample of two Bulgarian language corpora annotated by a native speaker. Rather than treat OOV tokens as a monolithic and undifferentiated problem, we progressively apply multi-faceted linguistic analyses to these corpora, characterizing both the words that these analyses explain and words yet to be explained, which we shall call the **residual vocabulary**. Our methods are a mixture of the vintage and the vogue: specialized edit distances, composition of finite-state transducers, a noisy channel model for language identification fitted with empirical Bayes, and neural network–based part of speech taggers. Collectively, our processes accurately explain more than two in three (69%) unknown Bulgarian words in a held-out set according to whether they are proper names, inflections, derivations, compounds, foreign words, or misspellings (as illustrated in both Figures 1 and 3, discussed in more depth in §5). We release our native speaker–annotated lexicon, intermediate analyses, and software at `www.github.com/gbotev1/bg`.

Proceedings of the 29th International Conference on Computational Linguistics, pages 5309–5326
October 12–17, 2022.

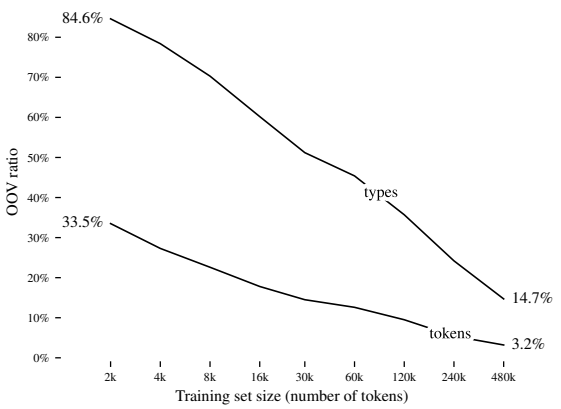

Figure 2: OOV rate as a function of data size (Bulgarian Wikipedia). Note the logarithmic horizontal axis.

2 Motivation and Related Work

Previously unseen words often represent a significant portion of the vocabulary, due in part to the Zipfian nature of language. Figure 2 illustrates this for various vocabulary sizes. Note that for the Bulgarian training data, the OOV rate remains high for both tokens (corpus instances of words) and types (vocabulary words) as found in a held-out set of 20,000 tokens. The rates are computed ignoring capitalization, punctuation, and numbers, so that these do not skew the count of unknown words.

The frontier of natural language processing as an engineering discipline has adopted information-theoretic subword tokenization (Sennrich et al., 2016; Kudo, 2018) to constrain the vocabulary size and provide a representation of all words, preventing any words from being out-of-vocabulary. Because such models dominate so much of the field of NLP, one may ask what value there is in analyzing the residual vocabulary today. Foremost, there is the corpus-linguistic and lexicographic value of characterizing this aspect of text: it is instructive about the patterns of lacunae in dictionaries or word formation processes in particular domains such as color (McCarthy et al., 2019). There are engineering applications as well. In languages with insufficient data for training large neural machine translation systems (Mueller et al., 2020) (or even for fine-tuning to new languages; see Lee et al., 2022), statistical methods dominate (Koehn and Knowles, 2017). The methods described in this paper are of value for populating the phrase tables of statistical MT models beyond what can be done with existing bilingual dictionaries, as in Vilar et al. (2007) who address spelling variants by online retokenization, or de Gispert (2006) who aims to reduce morphological variety. Moreover, entity linking and the use of gazetteers in named entity recognition both benefit from exact word representations. We underscore the fact that resource-poor languages are the norm, not the exception. Out of the world's roughly 7,000 languages, only 216 have more than 1,000 gloss definitions in Wiktionary, a popular multilingual dictionary.[1] For the remaining ≈6,800 data-poor languages, unknown words are not only neologisms and proper names; items of the core vocabulary are regularly absent from bilingual dictionaries or small but extant corpora.

Lexicon stratification, the splitting of the lexicon based on words' origin and degree of assimilation into the language (Ito and Mester, 1995), is a powerful technique to hone the processing of OOV words (Tsvetkov and Dyer, 2015). The four identified levels are the core vocabulary, the partially assimilated words, the fully assimilated words, and peripheral lexemes. This paper proffers empirical relative frequencies of these degrees and showcases a series of models that roughly correspond to these degrees.

3 The Bulgarian Language

Bulgarian is a member of the South Slavic branch of the Indo-European family, written in the Cyrillic script. As a member of the Balkan sprachbund, its lexis[2] and grammar have been influenced by areal effects. It thus displays several traits uncharacteristic of other Slavic languages (except Macedonian) which affect the apparent size of the lexicon: a postposed definite article marked for gender, the use of clitic pronouns, a lack of verbal infinitive, and limited case declension (Corbett and Comrie, 2003).

As a case study, Bulgarian is useful because it uses several widespread strategies for word formation. Its rich verbal morphology yields over 50 forms per verb lexeme. Derivational affixation and compounding are prevalent processes. In fact, derivation for nouns is both productive and regular (Krushkov, 2001). Finally, a significant fraction of the Bulgarian lexis is borrowed from Russian, Greek, or other languages, especially in technical contexts.

These properties have made Bulgarian a focus for linguistic examination and an area of interest in natural language processing. For example, Slavcheva (2003) devise a rich morphological tag set for Bulgarian verbs. Koeva et al. (2020) build a richly anno-

[1] https://en.wiktionary.org/wiki/Wiktionary:Statistics

[2] We distinguish between the *lexis*, i.e., the set of all words in a language, and the *lexicon*, i.e., the set of all lexemes.

tated corpus of web-crawled Bulgarian. Popov et al. (2020) construct a battery of models for multi-stage analysis of Bulgarian text, including lemmatization, parsing, and named entity recognition. Notably, the latter relies on a dictionary-based lemmatizer with a statistical model for fallback.

In contrast to these works, which offer an engineering approach to modeling Bulgarian, our work relies on computational tools insofar as they help characterize *properties* of Bulgarian text. Namely, we explore the relative frequency of various processes by which words—especially unknown words—arise in naturally occurring Bulgarian text.

4 Data

For our study, we need a large and representative corpus of Bulgarian text. We use the entirety of Bulgarian Wikipedia, which contains 1.3 million word types and 73.6 million word tokens (type–token ratio 0.018) after tokenization; a random sample of these is summarized in Figure 1.

We also must define the set of known words. We merge three broad-coverage bilingual dictionaries:

LanguageNet. 364,327 entries covering 155,703 unique English words.[3]

PanLex. 180,023 entries covering 70,986 unique English words (Baldwin et al., 2010).

Wiktionary. 51,537 entries covering 22,856 unique English words. We extract these with `Yawipa` (Wu and Yarowsky, 2020a,b).

In aggregate, these cover 165,644 unique Engish words, with a median number of translations 1 and mean approximately 2.360.[4]

To identify the residual vocabulary, we remove from Bulgarian Wikipedia all entries in our dictionaries as well as non-alphabetic entries, leaving 371,475 novel words—about one in every 200 tokens.[5] A random sample of 100 is summarized in Figure 3. The complete word lists and analyses are given in Appendix A. All annotations were validated or adjudicated by a non-author professional Bulgarian translator who is a native speaker.

What becomes immediately apparent is that the residual vocabulary after dictionary entries are removed comes from five major groups: morphological variants of other words, foreign words, misspellings, compound words, and proper names like place names or people. We devise computational approaches to tackle these five major categories.

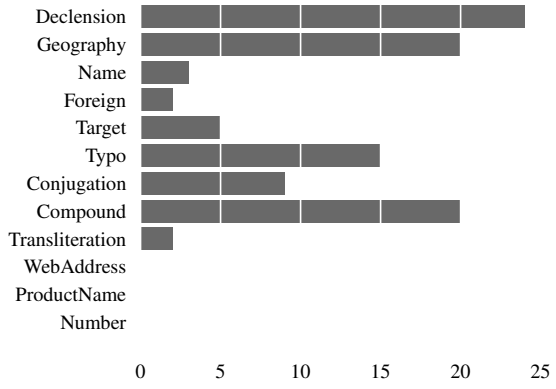

Figure 3: Taxonomized distribution of out-of-vocabulary types in Bulgarian Wikipedia that are unseen in Wiktionary, PanLex, and LanguageNet; random sample of 100 types. Compare with Figure 1.

Because we discovered an abundance of Russian words interspersed in the Bulgarian text, we also extract Russian–English bilingual entries from the same three dictionaries. We find 232,094 entries in Wiktionary covering 75,284 unique English words; 2,379,638 entries in PanLex covering 859,279 unique English words, and 1,633,709 unique entries in LanguageNet covering 879,438 unique English words. Their union covers 932,738 unique English words, with potentially multiple Russian candidate translations. The median number of translations was one, and the mean was 1.888.

Preprocessing To identify Bulgarian tokens in context, we first preprocess the text using the rule-based spaCy sentence segmenter and tokenizer (Honnibal and Montani, 2017). We found this to be faster than the Stanza neural tokenizer (Qi et al., 2020). We use Stanza for POS tagging, though its poor performance motivates the 'vintage' models we introduce below. In preliminary experiments, we also explored `TreeTagger` (Schmid, 1994, 1999).[6]

[3] `uakari.ling.washington.edu/languagenet/`

[4] The English pronoun *we* had the most translations: 306, due largely to inappropriate Bulgarian translations in LanguageNet which were first-person plural verb forms.

[5] This is approximately the same rate as Min and Wilson (1998) observe; they report that at this rate an out-of-vocabulary word occurs in 12% of sentences.

[6] Several avenues exist to improve part-of-speech tagging with minimal available resources. The most notable is projecting part-of-speech annotations across unsupervised word alignments into the language of interest, then using these silver annotations to train a new tagger (Yarowsky and Ngai, 2001; Täckström et al., 2013; Wang and Manning, 2014; Buys and Botha, 2016; Nicolai and Yarowsky, 2019; Eskander et al., 2020). Such methods could either complement a tagger such as Stanza trained in the language of interest via classifier combination or

We normalize all text to Unicode NFKD form to increase coverage.[7] This also allowed us to remove accents, which were predominantly used to mark stress. We subsequently remove tokens with any letter not in the Bulgarian alphabet. While this removes a few interesting cases like mp3-файлове 'MP3 files' and 2-то 'the second [thing]', on the whole the eliminations were useful: filtering URLs, email addresses, and also less structured non-words.

We found the need to preprocess the dictionaries by *hyphen flattening*. If a dictionary entry begins or ends with a hyphen, indicating that it is a prefix or suffix, we associate it with its non-hyphenated translational counterpart. For instance, the nonsensical English entry 'pra' is linked to the Bulgarian transliteration 'пра', and the Bulgarian prefix 'пра-' is correctly (and uniquely) associated with the English prefix 'great-'. After flattening, the Bulgarian entry 'пра' would have both 'pra' and 'great' listed as candidate translations. This preprocessing both reduces the dictionary's size and is crucial to increasing the impact of the compound analysis (§5.5).

Moreover, we define a heuristic to eliminate Old Bulgarian words, based on a 1945 orthographic reform that forbids word-final 'ъ'. Inspecting a sample of 50 words captured by this heuristic reveals that while none of the words filtered here were modern Bulgarian, 44% were in fact Old Bulgarian. The remainder were transliterations (the "unassimilated foreign words" of Tsvetkov and Dyer, 2015) from disparate languages: Italian (18%), Turkish[8] (16%), Kazakh (6%), Chinese (6%), Albanian (4%), and single exemplars of Irish, Portuguese, and Moldovan.[9]

5 Modeling and Analysis

This work by its nature differs from a great deal of the empirical work in natural language processing. The object of its inquiry is language itself, not computational models, and so we do not evaluate in the standard positivist paradigm of comparing scores on standard benchmarks. Instead, we build computational models to help sift through the millions of words in our corpus, study their distribution, and discover what can be modeled about them. After all, if we seek to tame the lexis, we must first understand it. In this regard, we follow the guidance of Hajič and Hajičová (2007) who recognize the value of objective assessment of models or theories on annotated corpora, grounded in linguistic intuition about the phenomenon to be modeled. Our characterization of the residual vocabulary helps to extend the linguistic intuition in an empirical manner.

The modularity of our approach lets us leverage prior tools and research in the language, and components can be upgraded as better models are devised (e.g., Nicolai et al., 2020 and Wiemerslage et al., 2022 for morphological analysis, Lewis et al., 2020 for inferring cognates). Moreover, disparate models for a single word formation process can be combined *in situ* via classifier combination or meta learning.

While many of the tools we use are tailored to the Bulgarian language, such as hand-crafted derivational rules from a grammar, in principle our approach makes minimal assumptions about the nature of the language. It could easily be adapted to other Slavic languages or, given sufficient prior typological information, other written languages writ large.

The overall sequence of method application is given in Figure 4. In the following sections, we elaborate on the most telling among these: language identification, then modeling morphology, misspellings, and compounds. Table 1 gives complete analyses for the held-out set of Wikipedia residual vocabulary, coupled with computer-predicted analyses.

5.1 Russian language filtering

A substantial fraction of the residual vocabulary is direct borrowings (loanwords) from other languages; cross-lingually this can be between 10% and 70% of the lexicon (Haspelmath and Tadmor, 2009). While our preprocessing eliminates several directly imported words that were not transliterated, a significant number of borrowings comes from Russian, which largely shares an alphabet with Bulgarian.

Some words can be clearly identified as non-Bulgarian by means of straightforward linguistic heuristics. The filtered words were mostly Russian, with a few exceptions that were Ukrainian or Serbian. We employ the following heuristics:

1. A Bulgarian word cannot begin or end with the soft sign 'ь'.
2. If the soft sign 'ь' occurs in the middle of a

annotate the language in the absence of in-language annotations.

[7] Kyle Gorman notes an increase of 0.3 in labeled attachment score for dependency parsing of Hindi, purely from normalization: http://www.wellformedness.com/blog/text-encoding-issues-in-universal-dependencies/.

[8] Note that due to both areal effects in the Balkan sprachbund and Bulgaria's past as an Ottoman territory, many Turkish lexemes have entered the Bulgarian lexicon as *fully assimilated* lexical items (Ito and Mester, 1995).

[9] We will not engage with the question of whether Romanian and Moldovan are dialects or separate languages; here, we use this as a shorthand for the Daco-Romance language written with the Cyrillic script.

Index	Word	Human Trans.	Alg. Trans.	Human Type	Human Sub-Type	Alg. Type	Alg. Sub-Type	Features	POS
1	звероферма	beast farm	beastl@	Compound	–	Compound	Partial	FEM	NOUN
2	неоспорван	uncontested	new|contested	Compound	–	Compound	–	–	ADJ
3	солокариера	solo career	sololcareer	Compound	–	Compound	–	FEM	NOUN
4	битовофекални	household faeces	household faeces	Compound	–	Compound	–	PL	NOUN
5	светлооранжев	light orange	light|orange	Compound	–	Compound	–	MASC	ADJ
6	контразузнаване	counter intelligence	counter|intelligence	Compound	–	Compound	–	NEUT	NOUN
7	удавил	drowned	–	Conjugation	–	Conjugation	–	–	PART
8	завзели	conquered	–	Conjugation	–	Conjugation	–	PL	PART
9	далекомером	distance meter	–	Foreign	Russian	Foreign	Russian	–	NOUN
10	мацелумът	Macellum	kitten|noise	Geography	Italian	Compound	–	–	PROPN
11	койбалската	Koybalska	koybaliltimes	Geography	Russian	Compound	–	FEM+DEF	ADJ (Proper)
12	горнобродчани	Inhabitants of Gorno Brod	gornolbrod	Geography	Bulgarian	Compound	–	PL	NOUN
13	костойчиновият	Kostoychinov	kostov|new	Geography	Bulgarian	Compound	–	DEF	ADJ (Proper)
14	ашташи	Ashotashen	ashot	Geography	Armenian	Declension	Fuzzy	–	PROPN
15	Уайя	Huaya	–	Geography	Mexican	Proper	Likely	–	PROPN
16	Бишина	Bishina	–	Geography	Serbian	Proper	Likely	–	PROPN
17	Кастей	Castei	–	Geography	Italian	Proper	Likely	–	PROPN
18	Бозовая	Bozovaya	–	Geography	Bulgarian	Proper	Likely	–	PROPN
19	Исаково	Isakovo	–	Geography	Russian	Proper	Likely	–	PROPN
20	Кеседжи	Kesdji	–	Geography	Greek	Proper	Likely	–	PROPN
21	Сигнора	Signora	–	Geography	Italian	Proper	Likely	–	PROPN
22	Соулънт	Solent	–	Geography	English	Proper	Likely	–	PROPN
23	Харагуа	Jaragua	–	Geography	Dominican Republic	Proper	Likely	–	PROPN
24	Ябълчице	Yabaltchitse	–	Geography	Bulgarian	Proper	Likely	–	PROPN
25	Байенбург	Bayenburg	–	Geography	German	Proper	Likely	–	PROPN
26	Петъчници	Petachnitsi	–	Geography	Bulgarian	Proper	Likely	–	PROPN
27	Валтотопион	Valtotopion	–	Geography	Greek	Proper	Likely	–	PROPN
28	Казакевичево	Kazakevichevo	–	Geography	Bulgarian	Proper	Likely	–	PROPN
29	енорийшкото	parish	parish|@	Declension	–	Compound	–	MAS+DEF	ADJ
30	апоплектичната	the apoplectic	Apoell|Ethic	Declension	–	Compound	–	DEF	ADJ
31	будени	awake	awake	Declension	–	Declension	Simple	PL	ADJ
32	ашерова	ashura	ashur	Declension	–	Declension	Fuzzy	FEM	ADJ
33	подобия	similarity	similarity	Declension	–	Declension	–	PL	NOUN
34	потника	tank top	tank top	Declension	–	Declension	Simple	MASC+DEF	NOUN
35	пролози	prologues	mercury	Declension	–	Declension	Fuzzy	PL	NOUN
36	грацията	The grace	grace	Declension	–	Declension	Simple	FEM+DEF	NOUN
37	ослепяло	became blind	blindness	Declension	–	Declension	–	NEUT	PART
38	смутното	turmoiled	turmoil	Declension	–	Declension	Simple	NEUT+DEF	ADJ
39	сталинци	stalinists	stalin	Declension	–	Declension	Fuzzy	PL	NOUN
40	суглинки	loams	suli	Declension	–	Declension	Simple	FEM+PL	NOUN
41	тръбеста	tubular	tubular	Declension	–	Declension	Fuzzy	FEM	ADJ
42	записната	pertaining to recording	recording	Declension	–	Declension	Simple	FEM+DEF	ADJ
43	потурчено	stamp down	stamp down	Declension	–	Declension	Simple	NEUT	ADJ
44	неголямото	not so big	rare	Declension	–	Declension	Simple	DEF	ADJ
45	еклектиката	eclecticism	eclectic	Declension	–	Declension	Fuzzy	FEM+DEF	NOUN
46	съблечената	The undressed	undressed	Declension	–	Declension	Simple	FEM+DEF	ADJ
47	персистиращи	persistent	persistence	Declension	–	Declension	Fuzzy	PL	NOUN
48	превърналата	the one that became	became	Declension	–	Declension	Simple	FEM+DEF	ADJ
49	кибернетизация	cybernetization	cybernetics	Declension	–	Declension	Fuzzy	FEM	NOUN
50	мултиетническия	the multiethnic	multiethnic	Declension	–	Declension	Simple	DEF	ADJ
51	Шотландска	Scotish	–	Declension	–	Proper	Standard	FEM	ADJ (Proper)
52	кодокан	Kodokan	kodo|kan	Name	School	Compound	–	–	PROPN
53	айдънидите	Aydin	@|nit	Name	Dynasty	Compound	–	–	PROPN
54	аморейско	Amorite	–	Name	Ethnicity	Foreign	Russian	NEUT	ADJ (Proper)
55	себрите	Ancestors of Serbians	seri	Name	Tribe	Declension	Fuzzy	PL+DEF	PROPN
56	ЦТА	Central Tibet Administration	–	Name	Organization	Proper	Likely	–	PROPN
57	Азел	Azel	–	Name	Person	Proper	Likely	–	PROPN
58	Юджи	Yuji	–	Name	Person	Proper	Likely	–	PROPN
59	ЗЕЛПО	ZELPO	–	Name	Building	Proper	Likely	–	PROPN
60	Какаи	Kakai	–	Name	Person	Proper	Likely	–	PROPN
61	Лопов	Lopov	–	Name	Person	Proper	Likely	–	PROPN
62	Мусан	Musan	–	Name	Person	Proper	Likely	–	PROPN
63	Пийбо	Peebo	–	Name	Person	Proper	Likely	–	PROPN
64	Дарбес	Darbez	–	Name	Person	Proper	Likely	–	PROPN
65	Прицак	Pritsak	–	Name	Person	Proper	Likely	–	PROPN
66	Халиду	Halidu	–	Name	Person	Proper	Likely	–	PROPN
67	Бейтлър	Beightler	–	Name	Person	Proper	Likely	–	PROPN
68	Вигберт	Witbert	–	Name	Person	Proper	Likely	–	PROPN
69	Евтахий	Evtahiy	–	Name	Person	Proper	Likely	–	PROPN
70	Оливиър	Olivier	–	Name	Person	Proper	Likely	–	PROPN
71	Ризберг	Rieseberg	–	Name	Person	Proper	Likely	–	PROPN
72	Памтивек	Pamtivek (colloquial for ancient)	–	Name	Book	Proper	Likely	–	NOUN
73	Харелсън	Harrelson	–	Name	Person	Proper	Likely	–	PROPN
74	Цибисова	Cybisowa	–	Name	Person	Proper	Likely	–	PROPN
75	Гроновиус	Gronovius	–	Name	Person	Proper	Likely	–	PROPN
76	Орочимаро	Orochimaro	–	Name	Person	Proper	Likely	–	PROPN
77	Настоплиси	Nastoplisi	–	Name	Person	Proper	Likely	–	PROPN
78	Присовский	Prisovskii	–	Name	Person	Proper	Likely	–	PROPN
79	Гутомсдатер	Gutomsdater	–	Name	Person	Proper	Likely	–	PROPN
80	Хаджипопова	Hadjipopova	–	Name	Person	Proper	Likely	–	PROPN
81	Христодоров	Christodorov	–	Name	Person	Proper	Likely	–	PROPN
82	буганин	gohst (archaic)	bugalnin	Target	–	Compound	–	MASC	NOUN
83	реверсира	reverse	reversed	Target	–	Declension	Fuzzy	–	VERB
84	Фесенджан	Fesenjan	–	Target	Iranian	Proper	Standard	–	PROPN
85	баунс	bounce	–	Transliteration	English	Misspelling	Substitution	MASC	NOUN
86	потът	the pot	floor, sweat, sex	Transliteration	English	Misspelling	Substitution	MASC+DEF	NOUN
87	футуризмо	futurism	The futurism	Transliteration	Italian	Misspelling	Substitution	–	NOUN
88	Форматър	formatter	–	Transliteration	English	Proper	Likely	–	NOUN
89	фрагмет	fragment	fral@, @|met	Typo	Omission	Compound	–	MASC	NOUN
90	денудаци	akin to denudational	daylodd person	Typo	Omission	Compound	–	–	NOUN
91	клавесинистикатегория	category|harpsichordists	category|harpsichordists	Typo	Concatenation	Compound	–	PL	NOUN
92	реакции	reactions	reactions	Typo	Misspelling	Declension	–	PL	NOUN
93	същотото	same	same	Typo	Addition	Declension	–	NEUT+DEF	ADJ
94	домантите	The Odomanti	tomatoes	Typo	Omission	Declension	–	PL+DEF	NOUN
95	ренгеново	x-ray	reinette	Typo	Omission	Declension	Fuzzy	–	ADJ
96	дествщ	acting	–	Typo	Omission	Misspelling	–	MASC	PART
97	отркрили	discovered	discovered	Typo	Addition	Misspelling	–	PL	PART
98	юзозападна	Southwestern	Southwestern	Typo	substitution	Misspelling	–	FEM	ADJ
99	Памятник	Monument	–	Typo	Substitution	Proper	Standard	MASC	NOUN
100	ПашиКатегория	Pasha Category	–	Typo	Concatenation	Proper	Likely	MASC	NOUN

Table 1: Manual classification of 100 randomly sampled words after classifying all of Bulgarian Wikipedia.

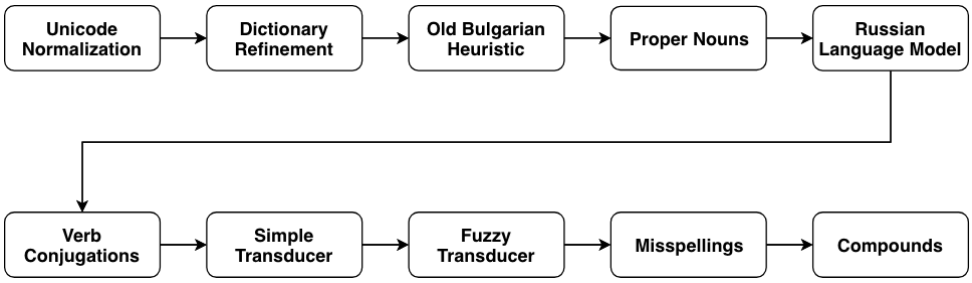

Figure 4: Sequence of methods applied to computationally analyze residual vocabulary

word, it must be followed by an 'о'. This is the only character that may follow the soft sign in modern Bulgarian. In Russian, however, many characters are attested following 'ь' (e.g., улыбаться 'to smile' and семья 'a family').

For words not covered by these heuristics, we require a different approach to distinguish them. Cognate identification and transliteration empirically identify borrowings poorly (Ciobanu and Dinu, 2015; Tsvetkov et al., 2015). We instead employ language identification to disambiguate the remainder as Bulgarian or Russian words. We use a noisy channel model of the language ℓ of word form $\boldsymbol{\xi}$:

$$p_\theta(\ell \mid \boldsymbol{\xi}) \propto p_\theta(\boldsymbol{\xi} \mid \ell)\, \pi(\ell).$$

In factoring this generative model, we use character 5-gram models as the language models $p_\theta(\boldsymbol{\xi} \mid \ell)$. The Bulgarian model is trained on Bulgarian ParlaMint 1.0, which comprises 10.5 million tokens covering 123,000 word types. The Russian model is trained on the Russian SynTagRus Universal Dependencies data, which comprises 496,000 tokens and 94,000 word types. The prior probability $\pi(\ell)$ is optimized on the data; that is, we use empirical Bayes to infer a point estimate.

After this process, every one of 50 randomly sampled non-Bulgarian words was filtered as foreign, though some were Ukrainian or Slovenian instead of Russian. We note that 15 of these words were ambiguous; their character sequences could have represented valid Bulgarian or Russian words.

5.2 Verbal morphology

While Bulgarian nominal declension is much simpler than its Slavic sibling languages (presenting only nominative and vocative cases) (Gribble, 1987; Townsend and Janda, 1996), its verbal conjugation system is rich, embodying "the morphologically richest and most problematic part-of-speech category" (Slavcheva, 2003). Bulgarian verbs reflect voice, tense, mood, person, number, and evidentiality.

To analyze Bulgarian verbs, we construct a finite-state transducer that builds on the UniMorph project (Sylak-Glassman et al., 2015a,b; Kirov et al., 2016, 2018; McCarthy et al., 2020) and Apertium (Forcada et al., 2011; Forcada and Tyers, 2016).[10] This enables fast, interpretable analysis by composition and union of machines. Composition corresponds to application of a morphological rule (Roark and Sproat, 2007), and union collects alternative rules (or candidate manifestations of a single rule) into one machine. Our finite-state transducer is designed to map inflected word forms to their citation forms (their *lemmas*), if the word forms were tagged as verbs by Stanza. We construct one finite-state transducer for each form–lemma pair in UniMorph and Apertium, then take the union of these machines.

Transforming a word $\boldsymbol{\xi}$ to its citation form is equivalent to composing a finite-state acceptor representing $\boldsymbol{\xi}$ with the transducer. If the two cannot compose (because $\boldsymbol{\xi}$ is not in the domain of definition (i.e., input language) of the transducer), then we do not suppose that $\boldsymbol{\xi}$ is an inflected verb form.

When applied to identified verbs in the residual vocabulary, a spot check of 50 supposed Bulgarian verbs shows that 46 are correctly predicted. Of the remaining four, two are Russian words that passed through the filter from §5.1. The others are охрени 'ocher' (a plural adjective) and *собено, a misspelling of the Bulgarian adverb особено 'specifically'.

5.3 Derivational morphology

Bulgarian has a productive set of derivational processes. Following the efficacy of the transducer for inflectional morphology, we introduce one for derivational morphology. We draw on the 22 derivational rules in Manova (2010) which explored the parsabil-

[10] UniMorph is a collection of morphological lexica in 167 languages, annotated in a cross-lingually consistent schema. Apertium is a rule-based machine translation system which includes a finite-state morphological analyzer and generator.

$$d_{\mathbf{x},\mathbf{y}}(i,j) = \min \begin{cases} 0 & \text{if } i = j = 0, \\ d_{\mathbf{x},\mathbf{y}}(i-1,j) + 1 & \text{if } i > 0, \\ d_{\mathbf{x},\mathbf{y}}(i,j-1) + 1 & \text{if } j > 0, \\ d_{\mathbf{x},\mathbf{y}}(i-1,j-1) + \mathbf{1}_{(\mathbf{x}_i \neq \mathbf{y}_j)} & \text{if } i,j > 0, \\ d_{\mathbf{x},\mathbf{y}}(i-2,j-2) + 1 & \text{if } i,j > 1 \text{ and } \mathbf{x}_i = \mathbf{y}_{j-1} \text{ and } \mathbf{x}_{i-1} = \mathbf{y}_j, \end{cases}$$

Figure 5: Recurrence relation the Damerau–Levenshtein distance between two strings $\mathbf{x}$ and $\mathbf{y}$. The dynamic program to tractably compute this is a modification of the Wagner–Fisher algorithm (1975) for Levenshtein distance.

ity hypothesis (Hay, 2001; Aronoff and Fuhrhop, 2002) for Bulgarian. Patseva (2017) was also a basis for derivational rules.

Composing the finite-state transducer for derivational analysis with itself, or with a finite-state transducer for modeling inflections, expands the coverage by capturing forms with multiple derivations, as is the relationship between хиндуистките 'the Hinduistics' and хинду 'Hindu':

хинду → хиндуист	(nominal derivation)
хиндуист → хиндуистка	(diminutive feminine)
хиндуистка → хиндуистки	(plural)
хиндуистки → хиндуистките	(definite article)

Such considerations are crucial because derived forms may themselves be inflected. Moreover, certain forms are more amenable to derivation. For instance, adverbs are often formed from the neuter singular form of adjectives, except for adjectives that end in -ки. These motivate a single transducer to consider the two jointly (Fischer et al., 2016).

This model of morphology is 68% accurate on a random sample. While some errors are due to misspellings, it also ignores stem alterations which may arise but are not encoded in the derivational transformations. While fine-tuning the transduction rules to handle cases like мед 'copper' → медникар 'coppersmith' or злато 'gold' → златар 'goldsmith' is possible based on prior knowledge, the approach gives a reasonable grounding in using the available linguistic resources for a language.

5.4 Misspelling

The analysis and recovery of misspellings has a long history in the computational processing of language (McIlroy, 1982; Kernighan et al., 1990; Kukich, 1992). Rather than simply *identifying* misspellings, which can be easily done by checking against an existing wordlist, we also seek to identify the correct spelling of the misspelled word. To do so, we employ the Damerau-Levenshtein distance (Damerau, 1964), a modification of Levenshtein's edit distance that also allows character transpositions as an edit operation. It is well known that transposition errors (e.g.*lang***au***ge* instead of *language*) are common typing errors (Salthouse, 1984, 1986), and the Damerau–Levenshtein distance gives a more parsimonious backtrace for them.

In the residual space, we identify misspellings as words with a Damerau–Levenshtein distance of 1 from an item in the vocabulary. Exactly computing the Damerau–Levenshtein distance requires a nontrivial extension of the standard edit distance (see Figure 5); however, the asymptotic complexity remains proportional to the product of the string pair's lengths—as in the standard edit distance.

We find that one in six words from the residual vocabulary of the Wikipedia corpus is a misspelling of a word into a non-word (Figure 3). To decipher the meanings of these words, we link them to existing words in the Bulgarian vocabulary by finding the in-vocabulary word with the smallest Damerau–Levenshtein distance. On a random sample of 50 Bulgarian words classified as misspellings (Table A.3), 35 of these were indeed misspellings (for an accuracy of 70%). The remainder were largely transliterations, inflected forms of verbs that were not identified via the methods described in §5.2, and some proper nouns.

Our approach targets correcting the spellings of non-words into valid words. A context-driven model could also identify misspellings of words into other words which are valid but infelicitous.

5.5 Compounds

Finally, we consider the word formation process of compounding. Unlike morphological derivation (which affixes bound morphemes to a lexeme to

create a new lexeme), compounding combines *free* morphemes to create a lexeme, as with the English word *candlestick*. We find it useful to process compounds after inflections because compounds as novel lexemes invite the same inflectional processes as non-compound lexemes of their core part of speech.

Following Wu and Yarowsky (2018), we consider compounds as words with two morphemes concatenated together, potentially with surface alterations. (McCarthy et al. (2019) used this to find compound color words in thousands of languages.) We split a word into all possible morpheme pairs, such that each morpheme has a length of at least 3 and at least one component has an edit distance at most 2 from some dictionary entry.[11] Thus, this method also identifies the decomposition of the compound word. When only one component fits the edit distance criterion, the decomposition omits the component with high edit distance. To make detection of compounds tractable, our implementation relies on fast prefix and suffix tries. A related alternative is the finite-state representation by Oflazer (1996).

We apply our compound analysis method to identify compounds in the residual words, and we manually evaluate a random sample of 50 predicted compounds Table A.4. Of these, 30 were correctly identified as compounds, and 22 were correctly decomposed. We observed a high number of false positives, which can be easily filtered out by examining the total edit distance of the components to known words. Every correctly identified compound has components whose combined edit distance is ≤ 2 (note that earlier we consider a compound to be valid if at least *one* component's edit distance to a known word is ≤ 2). Removing false positives with a total edit distance greater than 2 removes 18 incorrectly classified compounds, improving precision.

Many correctly identified compounds had a combined edit distance of zero or one (e.g., джазформация as джаз 'jazz' + формация 'formation'). Some errors were particularly instructive. For example, the word калейдоскопът 'the kaleidoscope', is incorrectly identified as a compound word whose second component is път 'road'. In fact, this word is a definite inflection of калейдоскоп 'kaleidoscope' using the suffix -ът. This reveals a transduction missing from our list in §5.2. In fact, we found the compound analysis to be quite helpful in identifying new inflectional suffixes, with which we augmented our FST for inflectional morphology.

[11] These values likely need to be adapted to new languages.

6 Discussion and Conclusion

We have investigated the space of unknown lexical items in naturally occurring text. In a case study on Bulgarian, a host of analytical models applied sequentially characterize the residual space of out-of-vocabulary words. Our models identify myriad processes responsible for these unknown words and map from such words to known words via heuristic and probabilistic processes. In this way, it complements Cucerzan and Yarowsky (2000) who model unknown words based on affixal or contextual similarity, and it affords means to improve machine translation.

The complete results of the residual space analyses are given in Table 1. Of the held-out set of 100 randomly sampled OOV words, our sequence of analyses properly taxonomized 69 of these. To confirm the robustness of these findings, a parallel study using the same series of techniques was conducted on the BulTreeBank corpus (Simov et al., 2002). In this case, 78% of a random sample of unknown words was correctly classified (see Table A.5), affirming the validity of the approach.

Initially one might suspect the need for less aggressive inflection and compounding models, given that so many errors were typos. On balance, significant fractions of the analyses were reasonable: even if an inflected form is misspelled, it is useful to reduce it to a lemma that can then reduce the space of possible correct spellings to which it can be mapped. While our annotation convention allows for only a single category per word, several examples show the benefit of using annotations as heuristics with shades of nuance worthy of human validation. For instance, several misspelled proper names are identified as names rather than typos, and a case of two words inadvertently joined by a deleted space (i.e., a typo) is correctly decomposed into those words by the compounding model.

In light of continued challenges in designing computational tools that effectively serve the world's thousands of languages, and that ignoring the linguistic traits of a language does not absolve the designer but rather induces greater harm (Bender, 2009), a detailed and taxonomized understanding of the behaviors of the language is vital. Our analysis of the word formation processes in such a way that can be grounded in the known lexicon affords both broad-scale familiarity with the language and practical value: it can tailor the design of core NLP tools to the residual vocabulary of a new language.

Acknowledgments

Arya D. McCarthy is supported by an Amazon AI2AI Fellowship and a Frederick Jelinek Fellowship.

References

Mark Aronoff and Nanna Fuhrhop. 2002. Restricting suffix combinations in German and English: Closing suffixes and the monosuffix constraint. *Natural Language & Linguistic Theory*, 20:451–490.

Timothy Baldwin, Jonathan Pool, and Susan Colowick. 2010. PanLex and LEXTRACT: Translating all words of all languages of the world. In *Coling 2010: Demonstrations*, pages 37–40, Beijing, China. Coling 2010 Organizing Committee.

Emily M. Bender. 2009. Linguistically naïve != language independent: Why NLP needs linguistic typology. In *Proceedings of the EACL 2009 Workshop on the Interaction between Linguistics and Computational Linguistics: Virtuous, Vicious or Vacuous?*, pages 26–32, Athens, Greece. Association for Computational Linguistics.

Jan Buys and Jan A. Botha. 2016. Cross-lingual morphological tagging for low-resource languages. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1954–1964, Berlin, Germany. Association for Computational Linguistics.

Alina Maria Ciobanu and Liviu P. Dinu. 2015. Automatic discrimination between cognates and borrowings. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 2: Short Papers)*, pages 431–437, Beijing, China. Association for Computational Linguistics.

P.G. Corbett and P.B. Comrie. 2003. *The Slavonic Languages*. Routledge Language Family Series. Taylor & Francis.

Silviu Cucerzan and David Yarowsky. 2000. Language independent, minimally supervised induction of lexical probabilities. In *Proceedings of the 38th Annual Meeting of the Association for Computational Linguistics*, pages 270–277, Hong Kong. Association for Computational Linguistics.

Fred J. Damerau. 1964. A technique for computer detection and correction of spelling errors. *Commun. ACM*, 7(3):171–176.

Adrià de Gispert. 2006. *Introducing linguistic knowledge into statistical machine translation.* Ph.D. thesis, Universitat Politècnica de Catalunya.

Ramy Eskander, Smaranda Muresan, and Michael Collins. 2020. Unsupervised cross-lingual part-of-speech tagging for truly low-resource scenarios. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 4820–4831, Online. Association for Computational Linguistics.

Andrea Fischer, Klára Jágrová, Irina Stenger, Tania Avgustinova, Dietrich Klakow, and Roland Marti. 2016. Orthographic and morphological correspondences between related Slavic languages as a base for modeling of mutual intelligibility. In *Proceedings of the Tenth International Conference on Language Resources and Evaluation (LREC'16)*, pages 4202–4209, Portorož, Slovenia. European Language Resources Association (ELRA).

Mikel L. Forcada, Mireia Ginestí-Rosell, Jacob Nordfalk, Jim O'Regan, Sergio Ortiz-Rojas, Juan Antonio Pérez-Ortiz, Felipe Sánchez-Martínez, Gema Ramírez-Sánchez, and Francis M. Tyers. 2011. Apertium: a free/open-source platform for rule-based machine translation. *Machine Translation*, 25(2):127–144.

Mikel L. Forcada and Francis M. Tyers. 2016. Apertium: a free/open source platform for machine translation and basic language technology. In *Proceedings of the 19th Annual Conference of the European Association for Machine Translation: Projects/Products*, Riga, Latvia. Baltic Journal of Modern Computing.

C.E. Gribble. 1987. *Reading Bulgarian Through Russian*. Slavica Publishers.

Jan Hajič and Eva Hajičová. 2007. Some of our best friends are statisticians. In *Text, Speech and Dialogue*, pages 2–10, Berlin, Heidelberg. Springer Berlin Heidelberg.

Martin Haspelmath and Uri Tadmor, editors. 2009. *Loanwords in the World's Languages: A Comparative Handbook*. De Gruyter Mouton.

Jennifer Hay. 2001. Lexical frequency in morphology: Is everything relative? *Linguistics*, 39:1041–1070.

Matthew Honnibal and Ines Montani. 2017. spacy 2: Natural language understanding with bloom embeddings, convolutional neural networks and incremental parsing. *To appear*, 7(1):411–420.

Junko Ito and Armin Mester. 1995. The core-periphery structure of the lexicon and constraints on reranking. In Jill Beckman, Suzanne Urbanczyk, and Laura Walsh, editors, *Papers in Optimality Theory*, volume 18 of *University of Massachusetts Occasional Papers in Linguistics [UMOP]*, pages 181–209. University of Massachusetts, Amherst: GLSA.

Mark D. Kernighan, Kenneth W. Church, and William A. Gale. 1990. A spelling correction program based on a noisy channel model. In *COLING 1990 Volume 2: Papers presented to the 13th International Conference on Computational Linguistics*.

Christo Kirov, Ryan Cotterell, John Sylak-Glassman, Géraldine Walther, Ekaterina Vylomova, Patrick Xia, Manaal Faruqui, Sabrina J. Mielke, Arya McCarthy, Sandra Kübler, David Yarowsky, Jason Eisner, and Mans Hulden. 2018. UniMorph 2.0: Universal Morphology. In *Proceedings of the Eleventh International Conference on Language Resources and Evaluation (LREC 2018)*, Miyazaki, Japan. European Language Resources Association (ELRA).

Christo Kirov, John Sylak-Glassman, Roger Que, and David Yarowsky. 2016. Very-large scale parsing and normalization of Wiktionary morphological paradigms. In *Proceedings of the Tenth International Conference on Language Resources and Evaluation (LREC'16)*, pages 3121–3126, Portorož, Slovenia. European Language Resources Association (ELRA).

Philipp Koehn and Rebecca Knowles. 2017. Six challenges for neural machine translation. In *Proceedings of the First Workshop on Neural Machine Translation*, pages 28–39, Vancouver. Association for Computational Linguistics.

Svetla Koeva, Nikola Obreshkov, and Martin Yalamov. 2020. Natural language processing pipeline to annotate Bulgarian legislative documents. In *Proceedings of the 12th Language Resources and Evaluation Conference*, pages 6988–6994, Marseille, France. European Language Resources Association.

Hristo Krushkov. 2001. Automatic morphological processing of Bulgarian proper nouns.

Taku Kudo. 2018. Subword regularization: Improving neural network translation models with multiple subword candidates. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 66–75, Melbourne, Australia. Association for Computational Linguistics.

Karen Kukich. 1992. Techniques for automatically correcting words in text. *ACM Comput. Surv.*, 24(4):377–439.

En-Shiun Lee, Sarubi Thillainathan, Shravan Nayak, Surangika Ranathunga, David Adelani, Ruisi Su, and Arya McCarthy. 2022. Pre-trained multilingual sequence-to-sequence models: A hope for low-resource language translation? In *Findings of the Association for Computational Linguistics: ACL 2022*, pages 58–67, Dublin, Ireland. Association for Computational Linguistics.

Dylan Lewis, Winston Wu, Arya D. McCarthy, and David Yarowsky. 2020. Neural transduction for multilingual lexical translation. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 4373–4384, Barcelona, Spain (Online). International Committee on Computational Linguistics.

Stela Manova. 2010. Suffix combinations in Bulgarian: parsability and hierarchy-based ordering. *Morphology*, 20(1):267–296.

Arya D. McCarthy, Christo Kirov, Matteo Grella, Amrit Nidhi, Patrick Xia, Kyle Gorman, Ekaterina Vylomova, Sabrina J. Mielke, Garrett Nicolai, Miikka Silfverberg, Timofey Arkhangelskiy, Nataly Krizhanovsky, Andrew Krizhanovsky, Elena Klyachko, Alexey Sorokin, John Mansfield, Valts Ernštreits, Yuval Pinter, Cassandra L. Jacobs, Ryan Cotterell, Mans Hulden, and David Yarowsky. 2020. UniMorph 3.0: Universal Morphology. In *Proceedings of the 12th Language Resources and Evaluation Conference*, pages 3922–3931, Marseille, France. European Language Resources Association.

Arya D. McCarthy, Winston Wu, Aaron Mueller, William Watson, and David Yarowsky. 2019. Modeling color terminology across thousands of languages. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 2241–2250, Hong Kong, China. Association for Computational Linguistics.

M. McIlroy. 1982. Development of a spelling list. *IEEE Transactions on Communications*, 30(1):91–99.

Kyongho Min and William H. Wilson. 1998. Integrated control of chart items for error repair. In *36th Annual Meeting of the Association for Computational Linguistics and 17th International Conference on Computational Linguistics, Volume 2*, pages 862–868, Montreal, Quebec, Canada. Association for Computational Linguistics.

Aaron Mueller, Garrett Nicolai, Arya D. McCarthy, Dylan Lewis, Winston Wu, and David Yarowsky. 2020. An analysis of massively multilingual neural machine translation for low-resource languages. In *Proceedings of the 12th Language Resources and Evaluation Conference*, pages 3710–3718, Marseille, France. European Language Resources Association.

Garrett Nicolai, Dylan Lewis, Arya D. McCarthy, Aaron Mueller, Winston Wu, and David Yarowsky. 2020. Fine-grained morphosyntactic analysis and generation tools for more than one thousand languages. In *Proceedings of the 12th Language Resources and Evaluation Conference*, pages 3963–3972, Marseille, France. European Language Resources Association.

Garrett Nicolai and David Yarowsky. 2019. Learning morphosyntactic analyzers from the Bible via iterative annotation projection across 26 languages. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 1765–1774, Florence, Italy. Association for Computational Linguistics.

Kemal Oflazer. 1996. Error-tolerant finite-state recognition with applications to morphological analysis and spelling correction. *Computational Linguistics*, 22(1):73–89.

Mirena Patseva. 2017. Bulgarian word stress analysis in the frame of prosody morphology interface. Technical report, Rutgers University.

Alexander Popov, Petya Osenova, and Kiril Simov. 2020. Implementing an end-to-end treebank-informed pipeline for Bulgarian. In *Proceedings of the 19th International Workshop on Treebanks and Linguistic Theories*, pages 162–167, Düsseldorf, Germany. Association for Computational Linguistics.

Peng Qi, Yuhao Zhang, Yuhui Zhang, Jason Bolton, and Christopher D Manning. 2020. Stanza: A python natural language processing toolkit for many human languages. *arXiv preprint arXiv:2003.07082*.

Brian Roark and Richard Sproat. 2007. *Computational approaches to morphology and syntax*, volume 4. OUP Oxford.

Timothy A Salthouse. 1984. Effects of age and skill in typing. *Journal of Experimental Psychology: General*, 113(3):345.

Timothy A Salthouse. 1986. Perceptual, cognitive, and motoric aspects of transcription typing. *Psychological bulletin*, 99(3):303.

Helmut Schmid. 1994. Probabilistic part-of-speech tagging using decision trees.

Helmut Schmid. 1999. Improvements in part-of-speech tagging with an application to german. In Susan Armstrong, Kenneth Church, Pierre Isabelle, Sandra Manzi, Evelyne Tzoukermann, and David Yarowsky, editors, *Natural Language Processing Using Very Large Corpora*, pages 13–25. Springer Netherlands, Dordrecht.

Rico Sennrich, Barry Haddow, and Alexandra Birch. 2016. Neural machine translation of rare words with subword units. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1715–1725, Berlin, Germany. Association for Computational Linguistics.

Kiril Simov, Petya Osenova, Milena Slavcheva, Sia Kolkovska, Elisaveta Balabanova, Dimitar Doikoff, Krassimira Ivanova, Alexander Simov, and Milen Kouylekov. 2002. Building a linguistically interpreted corpus of Bulgarian: the BulTreeBank. In *Proceedings of the Third International Conference on Language Resources and Evaluation (LREC'02)*, Las Palmas, Canary Islands - Spain. European Language Resources Association (ELRA).

Milena Slavcheva. 2003. Some aspects of the morphological processing of Bulgarian. In *Proceedings of the 2003 EACL Workshop on Morphological Processing of Slavic Languages*, pages 71–77, Budapest, Hungary. Association for Computational Linguistics.

John Sylak-Glassman, Christo Kirov, Matt Post, Roger Que, and David Yarowsky. 2015a. A universal feature schema for rich morphological annotation and fine-grained cross-lingual part-of-speech tagging. In *Systems and Frameworks for Computational Morphology*, pages 72–93, Cham. Springer International Publishing.

John Sylak-Glassman, Christo Kirov, David Yarowsky, and Roger Que. 2015b. A language-independent feature schema for inflectional morphology. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 2: Short Papers)*, pages 674–680, Beijing, China. Association for Computational Linguistics.

Oscar Täckström, Dipanjan Das, Slav Petrov, Ryan McDonald, and Joakim Nivre. 2013. Token and type constraints for cross-lingual part-of-speech tagging. *Transactions of the Association for Computational Linguistics*, 1:1–12.

C.E. Townsend and L.A. Janda. 1996. *Common and Comparative Slavic: Phonology and Inflection : with Special Attention to Russian, Polish, Czech, Serbo-Croatian, Bulgarian*. Slavica Publishers.

Yulia Tsvetkov, Waleed Ammar, and Chris Dyer. 2015. Constraint-based models of lexical borrowing. In *Proceedings of the 2015 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 598–608, Denver, Colorado. Association for Computational Linguistics.

Yulia Tsvetkov and Chris Dyer. 2015. Lexicon stratification for translating out-of-vocabulary words. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 2: Short Papers)*, pages 125–131, Beijing, China. Association for Computational Linguistics.

David Vilar, Jan-Thorsten Peter, and Hermann Ney. 2007. Can we translate letters? In *Proceedings of the Second Workshop on Statistical Machine Translation*, pages 33–39, Prague, Czech Republic. Association for Computational Linguistics.

Robert A. Wagner and Roy Lowrance. 1975. An extension of the string-to-string correction problem. *J. ACM*, 22(2):177–183.

Mengqiu Wang and Christopher D. Manning. 2014. Cross-lingual projected expectation regularization for weakly supervised learning. *Transactions of the Association for Computational Linguistics*, 2:55–66.

Adam Wiemerslage, Miikka Silfverberg, Changbing Yang, Arya McCarthy, Garrett Nicolai, Eliana Colunga, and Katharina Kann. 2022. Morphological processing of low-resource languages: Where we are and what's next. In *Findings of the Association for Computational Linguistics: ACL 2022*, pages 988–1007, Dublin, Ireland. Association for Computational Linguistics.

Winston Wu and David Yarowsky. 2018. Massively translingual compound analysis and translation discovery. In *Proceedings of the Eleventh International Conference on Language Resources and Evaluation (LREC 2018)*, Miyazaki, Japan. European Language Resources Association (ELRA).

Winston Wu and David Yarowsky. 2020a. Computational etymology and word emergence. In *Proceedings of the 12th Language Resources and Evaluation Conference*, pages 3252–3259, Marseille, France. European Language Resources Association.

Winston Wu and David Yarowsky. 2020b. Wiktionary normalization of translations and morphological information. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 4683–4692, Barcelona, Spain (Online). International Committee on Computational Linguistics.

David Yarowsky and Grace Ngai. 2001. Inducing multilingual POS taggers and NP bracketers via robust projection across aligned corpora. In *Second Meeting of the North American Chapter of the Association for Computational Linguistics*.

A Supplemental Material

In the following pages, we provide specific analyses, both hand-crafted and computationally performed, of the residual vocabulary. All tables are referred to in the main text.

Index	Word	Translation	Type	Sub-Type	Features	POS
1	петминутни	five minute	Compound	Declension	N/A	ADJ
2	кръглоскулови	round-cheeked	Compound	Declension	N/A	ADJ
3	Невроfиброматоза	Neurofibromatosis	Compound	N/A	N/A	NOUN
4	киноадаптация	adapted for movie	Compound	N/A	FEM	NOUN
5	постулиращ	postulating	Conjugation	N/A	N/A	PART
6	вселяват	inspire	Conjugation	N/A	PL	VERB
7	осквернява	desecrate	Conjugation	N/A	N/A	VERB
8	интерпретираща	interpretating	Conjugation	N/A	N/A	PART
9	ἑκάστου	each	Foreign	Ancient Greek	N/A	PRON
10	Fiyafi	savannah	Foreign	Bashkir	N/A	NOUN
11	18bit	18-bit	Foreign	English	N/A	ADJ
12	World";<br	World	Foreign	English	N/A	NOUN
13	goto	goto	Foreign	English	N/A	VERB
14	nü	nü	Foreign	English	N/A	NOUN
15	Darvin	Darvin	Foreign	English	N/A	PROPN
16	ordinatorium	ordinatorium	Foreign	Latin	N/A	NOUN
17	Branchinella	Branchinella	Foreign	Latin	N/A	NOUN
18	vagrans	vagrans	Foreign	Latin	N/A	NOUN
19	Genealogia	genealogy	Foreign	Portuguese	N/A	NOUN
20	Obrero	Workers'*	Foreign	Spanish	N/A	PROPADJ
21	Bilmeden	without knowing	Foreign	Turkish	N/A	PART
22	Предарица	Predaritsa	Geography	Bulgarian	N/A	PROPN
23	Устренският	Ustrenskiat	Geography	Bulgarian	DEF	PROPADJ
24	Чиевци	Chievtsi	Geography	Bulgarian	N/A	PROPN
25	еркечкия	erkechki	Geography	Bulgarian	DEF	PROPADJ
26	харманлии	Harmanlii	Geography	Bulgarian (Capital)	N/A	PROPN
27	Хетфилд	Hatfield	Geography	English	N/A	PROPN
28	Чансълърсвилската	Chancellorsville	Geography	English	DEF	PROPADJ
29	Норвич	Norwich	Geography	English	N/A	PROPN
30	келия	Kelia	Geography	Greek	N/A	PROPN
31	Карино	Karino	Geography	Italian	N/A	PROPN
32	Креспано	Crespano	Geography	Italian	N/A	PROPN
33	Триесткият	Triest	Geography	Italian	DEF	PROPADJ
34	Колàца	Colazza	Geography	Italian	N/A	PROPN
35	Майкубенския	Maikuben	Geography	Kazakh	DEF	PROPADJ
36	Хайдаркан	Khaidarkan	Geography	Kyrgyz	N/A	PROPN
37	Дондангеn	Dondagen	Geography	Latvian	N/A	PROPN
38	джемат	Djemat	Geography	Serbian (Capital)	N/A	PROPN
39	Пунтаренас	Puntarenas	Geography	Spanish	N/A	PROPN
40	презаснемането	retake	Declension	N/A	DEF	NOUN
41	сънародника	compatriot	Declension	N/A	DEF	NOUN
42	хидроплanът	the hydroplane	Declension	N/A	DEF	NOUN
43	закалени	hardened	Declension	N/A	PL	ADJ
44	амидразоните	the amidrazones	Declension	N/A	PL+DEF	NOUN
45	умственото	the mental	Declension	N/A	DEF	ADJ
46	прибоят	the surf	Declension	N/A	DEF	NOUN
47	панкреатичната	the pancreatic	Declension	N/A	DEF	ADJ
48	представителят	the representative	Declension	N/A	DEF	ADJ
49	интригуваща	intriguing	Declension	N/A	FEM	ADJ
50	алтъни	gold coins	Declension	N/A	PL	NOUN
51	абонаментни	subscription	Declension	N/A	PL	ADJ
52	ракетка	small rocket	Declension	N/A	DIM	NOUN
53	Тежките	the heavy	Declension	N/A	PL+DEF	ADJ
54	квесторското	the quaestoring	Declension	N/A	DEF	NOUN
55	кармелитка	Carmelite	Declension	N/A	FEM	PROPN
56	плутония	plutonium	Declension	N/A	DEF	PROPN
57	Баритонът	the baritone	Declension	N/A	DEF	NOUN
58	трактове	tracts	Declension	N/A	PL+DEF	NOUN
59	тюрколожка	turkologist	Declension	N/A	FEM	NOUN
60	лигандното	the ligand	Declension	N/A	NEUT+DEF	ADJ
61	Мухльовци	ninnies	Declension	N/A	PL	ADJ
62	Амфибио	Amphibio	Name	Car	N/A	PROPN
63	Керуал	Keroualle	Name	French	N/A	PROPN
64	Ерхфрид	Erchanfried	Name	German	N/A	PROPN
65	Арагами	Aragami	Name	Japanese	N/A	PROPN
66	Иродиади	Herodias	Name	Latin	N/A	PROPN
67	Евто	Evto	Name	Person	N/A	PROPN
68	Посълуайт	Postlethwaite	Name	Person	N/A	PROPN
69	Вассалли	Vassalli	Name	Person	N/A	PROPN
70	чихо	Chiho	Name	Person	N/A	PROPN
71	Рисова	Risova	Name	Person	N/A	PROPN
72	Фридлендър	Friedlander	Name	Person	N/A	PROPN
73	Тремитус	Tremitus	Name	Person	N/A	PROPN
74	Върчаковски	Varchakovski	Name	Person	N/A	PROPN
75	Камбанийски	Kambaniiski	Name	Person	N/A	PROPN
76	Адолъф	Adolf	Name	Russian	N/A	PROPN
77	Киспии	Kaspii	Name	Tribe	N/A	PROPN
78	611.8	611.8	Number	N/A	N/A	NUM
79	Ми34	Mi-34	Product Name	N/A	N/A	PROPN
80	Турбина	turbine	Target	N/A	FEM	NOUN
81	Настигайки	catching up	Target	N/A	N/A	PART
82	Покосен	Stricken	Target	N/A	N/A	PART
83	Студентство	College experience	Target	N/A	N/A	NOUN
84	месинг	brass	Target	N/A	N/A	NOUN
85	Питомен	domesticated	Target	N/A	N/A	ADJ
86	Салкъм	Offshoot	Target	N/A	N/A	NOUN
87	мерило	measure	Target	N/A	N/A	NOUN
88	черничев	mulberry	Target	N/A	MASC	ADJ
89	асимптотически	asymptotic	Target	N/A	MASC	ADJ
90	нюзпейпър	newspaper	Transliteration	English	N/A	NOUN
91	шейдъри	shaders	Transliteration	English	PL	NOUN
92	стигнатдо	reached up (to)	Typo	Concatenation	N/A	PART
93	1948.През	1948.Through	Typo	Concatenation	N/A	PRON
94	про\xad\xad\xad

Index	Word	Translation	Type	Sub-Type	Features	POS
1	овесопроизводител	oat producer	Compound	N/A	MASC	NOUN
2	непорнографски	non-pornographic	Compound	N/A	MASC	ADJ
3	бързоразрастващи	fast growing	Compound	N/A	PL	ADJ
4	окомерни	eye sketching	Compound	N/A	PL	ADJ
5	преформироването	the reformatting	Compound	N/A	NEU+DEF	NOUN
6	метапознавателните	the meta cognitive	Compound	N/A	PL+DEF	ADJ
7	трискатна	of three gables	Compound	N/A	FEM	ADJ
8	бундестреньор	coach of German national team	Compound	N/A	MASC	NOUN
9	многоцветниците	nymphalidae	Compound	N/A	PL+DEF	NOUN
10	авиокосмическо	(pertaining to) aerospace	Compound	N/A	NEUT	ADJ
11	геолокацията	geolocation	Compound	N/A	FEM+DEF	NOUN
12	следамерикански	post American	Compound	N/A	MASC	ADJ
13	китоловния	whaling	Compound	N/A	MASC+DEF	ADJ
14	видеоизкуство	videoart	Compound	N/A	NEUT	NOUN
15	новоизграденият	the newly built	Compound	N/A	MASC+DEF	ADJ
16	първооснови	primary basis	Compound	N/A	PL	NOUN
17	лейбгвардейците	the life guards	Compound	N/A	PL+DEF	NOUN
18	сухолубиви	drought-tolerant	Compound	N/A	PL	ADJ
19	наскоропоявилия	appeared recently	Compound	N/A	MASC+DEF	ADJ
20	леководолазът	the scuba diver	Compound	N/A	MASC+DEF	NOUN
21	косиха	mowed	Conjugation	Bulgarian	N/A	VERB
22	разбъркаха	mixed	Conjugation	N/A	N/A	VERB
23	обследвайки	investigating, inquiring	Conjugation	N/A	N/A	PART
24	заобиколил	go around, circumvent	Conjugation	N/A	N/A	PART
25	преместилите	moved around	Conjugation	N/A	PL+DEF	PART
26	нарекъл	called, named	Conjugation	N/A	MASC	PART
27	недостигнатия	unattainable	Conjugation	N/A	MASC+DEF	PART
28	тероризиращ	terrorizing	Conjugation	N/A	MASC	PART
29	досмила	digesting, grinding	Conjugation	N/A	N/A	PART
30	руководителей	leaders	Foreign	Russian	PL	NOUN
31	паутина	spider web	Foreign	Russian	FEM	NOUN
32	питянтятяра	Pitjantjatjara	Geography	Australian	N/A	PROPN
33	широкольшки	(of) Shiroka Laka	Geography	Bulgarian	PL	ADJ
34	сетница	Setnica	Geography	Bulgarian	N/A	PROPN
35	замфировска	zamphyrovska	Geography	Bulgarian	FEM	ADJ (Proper)
36	гулянци	Guliantsi	Geography	Bulgarian	N/A	PROPN
37	бенковската	benkovska	Geography	Bulgarian	FEM+DEF	ADJ (Proper)
38	знеполското	the znepolsko	Geography	Bulgarian	NEUT+DEF	ADJ (Proper)
39	алитуска	the alitusk	Geography	Bulgarian	FEM+DEF	ADJ (Proper)
40	отроковице	Otrokovice	Geography	Czech	N/A	PROPN
41	блекпулски	Blackpool	Geography	English	MASC	ADJ (Proper)
42	пфалцски	(pertaining to) Pfalz	Geography	German	MASC	ADJ (Proper)
43	сицилианска	Sicilian	Geography	Italian	FEM	ADJ
44	нямаморо	Nyamanoro	Geography	Japan	N/A	PROPN
45	твърдишкото	(pertaining to) Tvarditsa	Geography	Place	NEUT+DEF	ADJ
46	можайският	Mojayska	Geography	Russian	MASC+DEF	ADJ (Proper)
47	саянската	of the Sayan (Mountains)	Geography	Russian	FEM+DEF	ADJ
48	лядунския	(of) Liaodong	Geography	Russian	MASC+DEF	ADJ
49	верхневилюйское	Verkhnevilyuysk	Geography	Russian	N/A	ADJ (Proper)
50	болградското	Bolgradski	Geography	Ukranian	NEUT	ADJ (Proper)
51	азраки	Azraqi	Georgraphy	Persian	N/A	PROPN
52	купчето	the small pile/bunch	Declension	N/A	NEUT+DIM+DEF	NOUN
53	естуарното	the estuarine	Declension	N/A	NEUT+DEF	ADJ
54	сметачната	the calculating	Declension	N/A	FEM+DEF	ADJ
55	просешката	the beggary	Declension	N/A	FEM+DEF	ADJ
56	трахити	trachytes	Declension	N/A	PL	NOUN
57	миллети	millets	Declension	N/A	PL	NOUN
58	ротердамци	inhabitants of Rotherdam	Declension	N/A	PL	NOUN
59	ресинтезът	the resynthesis	Declension	N/A	MASC+DEF	NOUN
60	флуксетинът	the Fluoxetine	Declension	N/A	MASC+DEF	NOUN
61	неосъзнаваното	the unconsciously	Declension	N/A	NEUT+DEF	ADJ
62	изсечена	set in stone, cut down	Declension	N/A	FEM	ADJ
63	сайгите	the saiga anthlopes	Declension	N/A	PL+DEF	NOUN
64	смърчови	(of) spruce	Declension	N/A	PL	ADJ
65	конецовидните	thread-like	Declension	N/A	PL+DEF	ADJ
66	предлози	prepositions	Declension	N/A	PL	NOUN
67	селяка	peasant	Declension	N/A	MASC+DEF	NOUN
68	ленни	land granted by Ottomans	Declension	N/A	PL	ADJ
69	екстрахепатичните	extrahepatic	Declension	N/A	PL+DEF	ADJ
70	пуническото	pertaining to Punics	Declension	N/A	NEUT+DEF	ADJ
71	владишкото	of the bishop	Declension	N/A	NEUT+DEF	ADJ
72	кратовския	(pertaining to) Kratovo	Declension	N/A	MASC+DEF	ADJ (Proper)
73	дудукът	duduk	Declension	N/A	MASC+DEF	NOUN
74	пастафории	pastophoria	Declension	Transliteration	PL	NOUN
75	създателю	creator	Declension	Vocative	MASC	NOUN
76	костурчанка	kosturchanka	Name	Inhabitants	FEM	PROPN
77	дъмбълдоров	(pertaining to) Dumbledore	Name	Person	MASC	ADJ (proper)
78	северо	Severo	Name	Person	N/A	PROPN
79	квинтерна	gittern	Target	N/A	FEM	NOUN
80	назалност	nasality	Target	N/A	FEM	NOUN
81	нелютив	mild (not hot)	Target	N/A	MASC	ADJ
82	биткойн	bitcoin	Target	Transliteration	MASC	NOUN
83	скакач	springbok	Target	Zoology	MASC	NOUN
84	гюйсът	the jack	Transliteration	Dutch	MASC+DEF	NOUN
85	дайнамикс	dynamics	Transliteration	English	N/A	PROPN
86	обложака	cover	Typo	Addition	FEM	NOUN
87	меджународната	the international	Typo	Character Swap	FEM+DEF	ADJ
88	палеографикатегория	paleography category	Typo	Concatenation	FEM	NOUN
89	юдеизма	the judaism	Typo	Declension	MASC+DEF	NOUN
90	наташната	the next	Typo	Omission	FEM+DEF	ADJ
91	широкоразпространения	the widespread	Typo	Omission	MASC+DEF	ADJ
92	низхождение	descent	Typo	Omission	NEUT	NOUN
93	цитросуви	(of) citrus	Typo	Substitution	PL	ADJ
94	оковчателното	(of) the final result	Typo	Substitution	NEUT+DEF	ADJ
95	жинотните	the animals	Typo	Substitution	PL+DEF	NOUN
96	татраедър	tetrahedron	Typo	Substitution	MASC	NOUN
97	имплементзация	implementation	Typo	Substitution	FEM	NOUN
98	принадлежът	belong	Typo	Substitution	PL	VERB
99	оръсия	weapons	Typo	Substitution	PL	NOUN
100	предозонова	prednisone (therapy)	Typo	Transliteration	FEM	ADJ

Table A.2: Manual classification of 100 randomly sampled words from the tokenized Bulgarian Wikipedia Corpus **after** eliminating entries from the union of dictionaries. We use the UD part-of-speech tags from: https://universaldependencies.org/u/pos/. Table is summarized in Figure 3.

Index	Word	Valid
1	витал	Yes
2	втрху	Yes
3	ихрам	Yes
4	синут	Yes
5	съкър	Yes
6	витрал	Yes
7	маквис	Yes
8	пераун	Yes
9	почест	Yes
10	ревабш	Yes
11	рененг	Yes
12	живееяг	Yes
13	модулин	Yes
14	гутболни	Yes
15	джутсуту	Yes
16	камбоурн	Yes
17	убеждавт	Yes
18	читирима	Yes
19	антатната	Yes
20	вододпади	Yes
21	наблядава	Yes
22	присъствт	Yes
23	художникт	Yes
24	обстрикция	Yes
25	преостъпва	Yes
26	ассортимент	Yes
27	монодрамата	Yes
28	присътствал	Yes
29	революциятс	Yes
30	числиността	Yes
31	продолжавало	Yes
32	нараставащата	Yes
33	пристрелването	Yes
34	стандфордското	Yes
35	модерницираните	Yes
36	туид	No
37	течащ	No
38	тодас	No
39	шейдър	No
40	спомнящ	No
41	връчващо	No
42	кодеинът	No
43	напомняш	No
44	невиждащ	No
45	струващо	No
46	китобойци	No
47	влайковите	No
48	кварковото	No
49	радиошоуто	No
50	семинолско	No

Table A.3: Human validation of random sample of misspelling classifications.

Index	Word	Decomposition	Edit Distance	Valid Compound	Valid Decomposition
1	калейдоскопът	калейдоскоп|път	1	No	No
2	дроидчето	@|ридчето	2	No	No
3	вазодилатиращ	вазови|датиращ	2	Yes	No
4	паналбанската	пан|албанската	0	Yes	Yes
5	трудноподвижност	трудно|подвижност	0	Yes	Yes
6	узункьопрюйския	@|райския	9	No	No
7	крайгълните	крайгълен|ите	1	No	No
8	епископалианците	епископа|ливанците	1	No	No
9	фотостареенето	фото|стареенето	0	Yes	Yes
10	тескерета	тес|@/@|ета	6	No	No
11	видеообмен	видео|обмен	0	Yes	Yes
12	дефтерхането	дефтера|нето	1	Yes	Yes
13	класфицира	@|скицира	4	No	No
14	несатнтименталното	@|менталното	8	Yes	No
15	предпубертетна	пред|пубертетна	0	Yes	Yes
16	екплозивни	@|позивни	3	No	No
17	сложноустроени	сложно|устроени	0	Yes	Yes
18	миникомикси	мини|комикси	0	Yes	Yes
19	бромалгин	бром|олгин	1	Yes	Yes
20	хиподермата	хипо|дермата	0	Yes	Yes
21	зогисткия	зог|есткия	1	No	Yes
22	колаборанти	кол|лаборанти/кола|оранти	1	Yes	No
23	древноеврейските	древно|еврейските	0	Yes	Yes
24	щалупьоненщалупьонен	щало|@	16	No	No
25	нарамвали	@|вали/нара|@	5	No	No
26	друмевите	@|ите	6	No	No
27	екстрабукалната	@|калната	8	Yes	No
28	дзайбацу	дза|@	5	Yes	No
29	анасонлийките	анасон|@	7	No	No
30	петокласно	пето|класно	0	Yes	Yes
31	джазформация	джаз|формация	0	Yes	Yes
32	крайдунавски	край|дунавски	0	Yes	Yes
33	елабуцки	ела|@	5	No	No
34	ориксът	ори|кът	1	No	No
35	римокатолическа	римо|католическа	0	Yes	Yes
36	арондисмана	@|имана	6	No	No
37	истанбулчаникатегория	истанбулчани|категория	0	Yes	Yes
38	сподобиха	спо|добиха	0	Yes	Yes
39	прокомуникирана	прокоп|@	10	Yes	No
40	леополдините	леополд|дините	1	No	No
41	детройтът	детройт|@	2	No	Yes
42	шитл‘ивица	@|вица	5	No	No
43	премъдростната	@|яростната	6	Yes	No
44	шестмоторни	шест|моторни	0	Yes	Yes
45	филмографията	фил|зографията/филм|зографията	1	Yes	Yes
46	средногъстата	средно|гъстата	0	Yes	Yes
47	безкуполен	без|куполен	0	Yes	Yes
48	гоцезелчевската	гоце|енчевската	2	Yes	Yes
49	епскоп	@|коп	3	No	No
50	лопатовиднозъб	лопатови|@	6	Yes	No

Table A.4: Human validation of random sample of compound analysis.

Index	Word	Human Trans.	Alg. Trans.	Human Type	Human Sub-Type	Alg. Type	Alg. Sub-Type	Features	POS
1	н-к	manage (abbreviated)	N/A	Abbreviation	N/A	Foreign	Russian	MASC	NOUN
2	полупансион	half board	semi board	Compound	N/A	Compound	N/A	MASC	NOUN
3	свръхелегантен	overly well dressed	svralelegant	Compound	N/A	Compound	N/A	MASC	NOUN
4	по-нагъл	more impudent	N/A	Compound	N/A	Foreign	Russian	MASC	ADJ
5	търговско-промишлена	Industrial-and-retail	N/A	Compound	N/A	Foreign	Russian	FEM	ADJ
6	русокоси	blond	russian	Compound	N/A	Declension	Fuzzy	PL	ADJ
7	новооткритото	the newly found	openings	Compound	N/A	Declension	Fuzzy	NEUT+DEF	ADJ
8	мироопазващите	peace-keeping	peacelkeeping	Compound	N/A	Declension	Simple	PL+DEF	ADJ
9	подплашил	slightly scared	tear offlbeaten	Conjugation	N/A	Compound	N/A	MASC	PART
10	инспирирано	inspired	@learly	Conjugation	N/A	Compound	N/A	NEUT	PART
11	жужи	buzz	N/A	Conjugation	N/A	Conjugation	N/A	N/A	VERB
12	могли	could	N/A	Conjugation	N/A	Conjugation	N/A	N/A	PART
13	забиха	poke	N/A	Conjugation	N/A	Conjugation	N/A	N/A	VERB
14	карало	driven	N/A	Conjugation	N/A	Conjugation	N/A	NEUT	ADJ
15	плъзна	slide	N/A	Conjugation	N/A	Conjugation	N/A	N/A	VERB
16	изгонва	expels	N/A	Conjugation	N/A	Conjugation	N/A	N/A	VERB
17	изкъпем	take a bath	N/A	Conjugation	N/A	Conjugation	N/A	N/A	VERB
18	копнели	longing	N/A	Conjugation	N/A	Conjugation	N/A	PL	PART
19	работиш	work	N/A	Conjugation	N/A	Conjugation	N/A	N/A	VERB
20	сдобили	obtained	N/A	Conjugation	N/A	Conjugation	N/A	N/A	VERB
21	оставаме	remaining	remaining	Conjugation	N/A	Conjugation	N/A	N/A	VERB
22	отзовали	responded	N/A	Conjugation	N/A	Conjugation	N/A	N/A	VERB
23	обвиняват	accuse	N/A	Conjugation	N/A	Conjugation	N/A	N/A	VERB
24	познаваха	recognized	N/A	Conjugation	N/A	Conjugation	N/A	N/A	VERB
25	промъкнал	sneaked	N/A	Conjugation	N/A	Conjugation	N/A	N/A	PART
26	дипломираш	graduate	N/A	Conjugation	N/A	Conjugation	N/A	N/A	VERB
27	започвайте	begin	N/A	Conjugation	N/A	Conjugation	N/A	N/A	VERB
28	проведохме	carried out	N/A	Conjugation	N/A	Conjugation	N/A	N/A	VERB
29	разчитайте	rely	N/A	Conjugation	N/A	Conjugation	N/A	N/A	VERB
30	поздравяват	greet	N/A	Conjugation	N/A	Conjugation	N/A	N/A	VERB
31	представяше	represented	N/A	Conjugation	N/A	Conjugation	N/A	N/A	VERB
32	претоварваш	overload	N/A	Conjugation	N/A	Conjugation	N/A	N/A	VERB
33	разстройваме	disturb	N/A	Conjugation	N/A	Conjugation	N/A	N/A	VERB
34	съсредоточиш	concentrate (mentally)	N/A	Conjugation	N/A	Conjugation	N/A	N/A	VERB
35	тръшна	fall abbruptly	trish	Conjugation	N/A	Declension	Fuzzy	N/A	VERB
36	наметна	drape over	document	Conjugation	N/A	Declension	Fuzzy	N/A	VERB
37	назначиха	appointed	N/A	Conjugation	N/A	Declension	Simple	N/A	VERB
38	озъртат	look around	N/A	Conjugation	N/A	N/A	N/A	N/A	VERB
39	поведох	lead	N/A	Conjugation	N/A	Proper	Likely	N/A	VERB
40	Дувър	Dover	N/A	Geography	English	Proper	Likely	N/A	PROPN
41	Козро	Kozro	N/A	Geography	Russian	Proper	Likely	N/A	PROPN
42	Ридсдейл	Reedsdale	N/A	Geography	English	Proper	Likely	N/A	PROPN
43	Апенините	Appennini	N/A	Geography	Italian	Proper	Likely	PL	PROPN
44	далавери	deals	given|friends	Declension	N/A	Compound	N/A	FEM+PL	NOUN
45	манталитетът	the mentality	mentalityl@	Declension	N/A	Compound	N/A	MASC+DEF	NOUN
46	клъвки	beak	click	Declension	N/A	Declension	Fuzzy	FEM+PL	NOUN
47	божиите	godly	godly	Declension	N/A	Declension	Fuzzy	PL+DEF	ADJ
48	болките	the pains	pains	Declension	N/A	Declension	Simple	FEM+PL+DEF	NOUN
49	гърмежи	thunder	report	Declension	N/A	Declension	Simple	PL	NOUN
50	епохата	the epoch	N/A	Declension	N/A	Declension	Simple	FEM+DEF	NOUN
51	великите	The great	veliki	Declension	N/A	Declension	Simple	PL+DEF	ADJ
52	депутата	the congressman	congressman	Declension	N/A	Declension	Simple	MASC+DEF	NOUN
53	детската	the childish	toy	Declension	N/A	Declension	Simple	FEM+DEF	ADJ
54	повелите	the commands	entrusted	Declension	N/A	Declension	Fuzzy	FEM+PL+DEF	NOUN
55	клетвения	sworn	sworn	Declension	N/A	Declension	Simple	MASC+DEF	ADJ
56	пазарльци	bargains	bargain	Declension	N/A	Declension	Fuzzy	MASC PL	NOUN
57	погребите	cellar, arms depot	entomb, bury	Declension	N/A	Declension	Fuzzy	MASC+PL+DEF	NOUN
58	случилото	occurred	occurred	Declension	N/A	Declension	Simple	NEUT+DEF	PART
59	чехкините	The Czech (females)	Check (female)	Declension	N/A	Declension	Simple	FEM+DEF	ADJ (Proper)
60	заловеният	captured	captured	Declension	N/A	Declension	Simple	MASC+DEF	ADJ
61	известните	famous	famous	Declension	N/A	Declension	Simple	PL+DEF	ADJ
62	момченцето	the little boy (demunitive)	little boy	Declension	N/A	Declension	Simple	NEUT+DEF	NOUN
63	отдалечила	distanced	N/A	Declension	N/A	Declension	Simple	FEM	PART
64	премиерите	the prime ministers	premiers	Declension	N/A	Declension	Simple	MASC+DEF	NOUN
65	софийската	Sofia	Sofia	Declension	N/A	Declension	Simple	FEM+DEF	ADJ
66	тексасците	the texans	texan	Declension	N/A	Declension	Fuzzy	PL+DEF	NOUN
67	еврофондове	european funds	eurofor	Declension	N/A	Declension	Fuzzy	MASC+PL	NOUN
68	изпратените	sent	sent	Declension	N/A	Declension	Simple	PL+DEF	PART
69	позиционните	positioning	position	Declension	N/A	Declension	Simple	PL+DEF	ADJ
70	съвестността	the conscience	conscience	Declension	N/A	Declension	Fuzzy	FEM+DEF	NOUN
71	холивудските	the hollywood	hollywood	Declension	N/A	Declension	N/A	PL+DEF	ADJ
72	необмислените	the thoughtless	thoughtless	Declension	N/A	Declension	Fuzzy	PL+DEF	ADJ
73	вестникарските	the newspaper	newspaper	Declension	N/A	Declension	Simple	PL+DEF	ADJ
74	изразходваните	consumed	consumed	Declension	N/A	Declension	Simple	PL+DEF	ADJ
75	социалдемократически	social democratic	socialdemocrat	Declension	N/A	Declension	Simple	PL	ADJ
76	мъжът	the man	N/A	Declension	N/A	N/A	N/A	MASC+DEF	NOUN
77	студът	the cold	N/A	Declension	N/A	N/A	N/A	MASC+DEF	NOUN
78	провинилите	the guilty	N/A	Declension	N/A	N/A	N/A	PL+DEF	ADJ
79	БВ	BV	N/A	N/A	N/A	Proper	Likely	N/A	N/A
80	Жега	Heat	N/A	Name	Movie	Proper	Likely	FEM	NOUN
81	Клио	Clio	N/A	Name	Car	Proper	Likely	N/A	PROPN
82	ПАНОВ	Panov	N/A	Name	Person	Proper	Likely	MASC	PROPN
83	Симон	Simon	N/A	Name	Person	Proper	Likely	N/A	PROPN
84	Чейни	Cheney	N/A	Name	Person	Proper	Likely	N/A	PROPN
85	Ганева	Ganeva	N/A	Name	Person	Proper	Likely	N/A	PROPN
86	Емилия	Emilia	N/A	Name	Person	Proper	Likely	FEM	PROPN
87	Трифон	Trifon	N/A	Name	Person	Proper	Likely	MASC	PROPN
88	Централ	Central	N/A	Name	Hotel	Proper	Likely	N/A	PROPN
89	литерер	Litteraire	N/A	Name	Newspaper	Proper	Funky	N/A	PROPN
90	Елизабет	Elizabeth	N/A	Name	Person	Proper	Likely	N/A	PROPN
91	Компанис	Companys	N/A	Name	Person	Proper	Likely	N/A	PROPN
92	Лизаразу	Lizarazu	N/A	Name	Person	Proper	Likely	N/A	PROPN
93	Талейран	Talleyrand	N/A	Name	Person	Proper	Likely	N/A	PROPN
94	Анастасия	Anastasia	N/A	Name	Person	Proper	Likely	FEM	PROPN
95	налудничаво	crazy	@lchavo	Target	N/A	Compound	N/A	NEUT	ADJ
96	олелия	commotion	yikes	Target	N/A	Declension	Fuzzy	FEM	NOUN
97	разведряване	détente	clearing	Target	N/A	Declension	Fuzzy	NEUT	NOUN
98	Земя	Earth	N/A	Target	N/A	N/A	N/A	FEM	NOUN
99	наврьх	at the peak of	N/A	Target	N/A	N/A	N/A	N/A	ADV
100	Даунтаун	downtown	N/A	Transliteration	English	Proper	Likely	MASC	NOUN

Table A.5: Manual classification of 100 randomly sampled words after classifying all of the BulTreeBank corpus, in analogy with Table 1.

PSSAT: A Perturbed Semantic Structure Awareness Transferring Method for Perturbation-Robust Slot Filling

Guanting Dong[1]*, Daichi Guo[1]*, Liwen Wang[1]*, Xuefeng Li[1]*, Zechen Wang[1], Chen Zeng[1], Keqing He[2], Jinzheng Zhao[3], Hao Lei[1], Xinyue Cui[1], Yi Huang[4], Junlan Feng[4], Weiran Xu[1]*

[1]Beijing University of Posts and Telecommunications, Beijing, China
[2]Meituan Group, Beijing, China
[3]University of Surrey, UK
[4]China Mobile Research Institute

Abstract

Most existing slot filling models tend to memorize inherent patterns of entities and corresponding contexts from training data. However, these models can lead to system failure or undesirable outputs when being exposed to spoken language perturbation or variation in practice. We propose a perturbed semantic structure awareness transferring method for training perturbation-robust slot filling models. Specifically, we introduce two MLM-based training strategies to respectively learn contextual semantic structure and word distribution from unsupervised language perturbation corpus. Then, we transfer semantic knowledge learned from upstream training procedure into the original samples and filter generated data by consistency processing. These procedures aim to enhance the robustness of slot filling models. Experimental results show that our method consistently outperforms the previous basic methods and gains strong generalization while preventing the model from memorizing inherent patterns of entities and contexts.

1 Introduction

The slot filling (SF) task in the goal-oriented dialog system aims to identify task-related slot types in certain domains for understanding user utterances. Traditional supervised slot filling models and sequence labeling methods (Liu and Lane, 2015, 2016; Goo et al., 2018; Niu et al., 2019; He et al., 2020a,b; Wang et al., 2022a) have shown remarkable performance. However, these models tend to memorize inherent patterns of entities and contexts (Wang et al., 2022b; Lin et al., 2021). Faced with uncertainty and diversity of human language expression, the perturbation of entities and contexts will lead to a decrease in the generalization ability of the SF model, which hinders its further application in practical dialog scenarios.

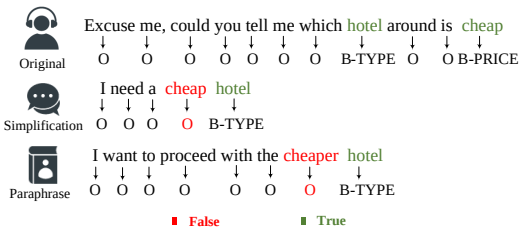

Figure 1: The impact of diverse spoken language perturbations on the slot filling system in real scenarios.

Due to the variety of expression habits, users may not interact with the dialogue system abiding by a rigid input mode in real dialog scenarios. Instead, the expression styles of users would be of high lexical and syntactic diversity while users express their intentions. An interesting finding is that, every expression retains the key semantic information of the sentence to ensure consistency of the intention, but it inevitably damages the semantic structure of the context. As shown in Figure 1, the original sentence comes from training data, while the other two sentences are real queries of users with different language habits. Firstly, paraphrase and simplification perturb the contextual semantic structure of the original sentence to various degrees. Secondly, some slot entities also suffer from word perturbations. However, they all retain price-related information to express the same intention. We refer to the above two perturbations collectively as Spoken Language Perturbation. The previous slot filling model, which tends to memorize entity patterns, has a significantly reduced generalization ability when faced with these situations. Therefore, it is necessary to train a robust slot filling model against perturbations in practical application.

Recently, improving the robustness of NLP systems against input perturbations has attracted increasing attention. Most existing studies (Wu et al., 2021; Moradi and Samwald, 2021; Gui et al., 2021) that explored the robustness problem are only about rule-based synthetic datasets, which have certain limitations. Further, Namysl et al. (2020) focused

*The first four authors contribute equally. Weiran Xu is the corresponding author. Email: dongguanting@bupt.edu.cn

Proceedings of the 29th International Conference on Computational Linguistics, pages 5327–5334
October 12–17, 2022.

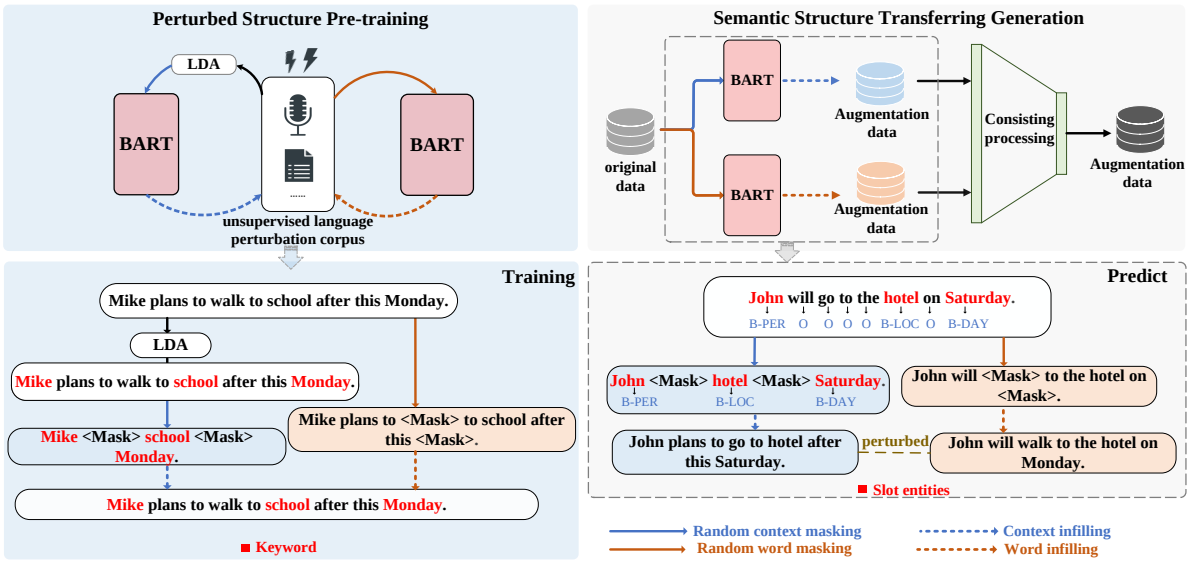

Figure 2: The overall architecture of the PSSAT framework. Two dotted boxes show the specific processes of the MLM-based strategies at pre-training and transferring generation stage, respectively.

on the robustness of the NER model against Optical Character Recognition (OCR) disturbance and misspellings. However, real-world dialogue systems face more diverse perturbations due to frequent interactions with users. Liu et al. (2020) proposed Language understanding augmentation, which contains four data augmentation methods, to simulate natural perturbations. Nevertheless, each method is designed for a specific perturbation, which cannot generalize for other unknown perturbations.

To solve the above issues, in this paper, we propose a **P**erturbed **S**emantic **S**tructure **A**wareness **T**ransferring method (**PSSAT**). It can generate augmented data based on human diversity expressions. In fact, it is not difficult to obtain unsupervised corpora containing spoken language perturbations in real-world scenarios (e.g. social media). Therefore, we extract the texts from two multi-modal datasets (Zhang et al., 2018; Lu et al., 2018) and construct an unsupervised language perturbation corpus, which helps the model learn the semantic structure of perturbed data. To be specific, we introduce a perturbed structure pre-training stage, which guides the model to directly learn contextual semantic structure and words distribution from unsupervised language perturbation corpus through two different MLM-based training strategies, respectively. To better eliminate the distribution gap between upstream and downstream data, we design a *Semantic Structure Transferring Generation* stage to transfer the upstream learned semantic structure knowledge to downstream original training samples. By doing so, the generated augmented samples are more in line with the spoken language perturbation. However, as there are mixed perturbations existed in upstream corpus, the model may generate some low-quality samples. To alleviate this problem, we introduce *Consistency Processing* to filter generated samples.

Our contributions are three-fold: (1) To the best of our knowledge, this is the first work to investigate spoken language perturbation of slot filling tasks and validate the vulnerability of existing rule-based methods in the condition of diverse language expressions. (2) We propose a perturbed semantic structure awareness transferring method, which transfers the learned contextual semantic structure and word distribution into the original samples through the MLM-based method. (3) Experiments demonstrate that our method outperforms all baseline methods and gains strong generalization while preventing the model from memorizing inherent patterns of entities and contexts.

2 Methodology

2.1 Problem Definition

Given a tokenized utterance $X = \{x_1, x_2, \ldots, x_N\}$ and its corresponding BIO format label $Y = \{y_1, y_2, \ldots, y_N\}$, we formulate the spoken language perturbation process in the real scenario as $X' = \mathcal{P}_x(X), Y' = \mathcal{P}_y(Y)$ such that $X' \neq X$ but Y' may be identical with Y or not. The perturbation-robust slot filling requires the model to be tested on the perturbed test dataset $\{(X', Y')\}$ but with no

access to the spoken language perturbation process $\mathcal{P}(\cdot)$ or perturbed data during the training phase.

2.2 Perturbed Structure Pre-training

The perturbed structure pre-training stage guides the model to learn the semantic structure from realistic perturbed data. We carefully collected several spoken language perturbation datasets to build an unsupervised language perturbation corpus[1]. Inspired by the key idea of masked language model (MLM) (Devlin et al., 2018), which randomly replaces a few tokens in a sentence with the special token *[MASK]* and recovers the original tokens by a neural network, we introduce two augmentation strategies, as shown in Figure 2.

Random Word Masking (RWM): words are randomly selected for masking and infilling to simulate the word perturbation, which guides the model to learn word distribution from real perturbed data.

Random Context Masking (RCM): we filter out the keywords of each sentence through Latent Dirichlet Allocation (LDA) (Blei et al., 2003) to keep the key information of the sentence. For non-keyword parts, we regard them as context spans of each sentence and conduct random masking and infilling. In this way, the model learns the contextual semantic structure from realistic perturbed data. Unlike word infilling, context infilling can generate multiple tokens for each *[MASK]* position.

2.3 Semantic Structure Transferring Generation

The Semantic Structure Transferring Generation stage aims to transfer learned contextual semantic structure and word distribution from upstream pre-trained model to downstream training samples. As shown in Figure 2, pre-trained models are separately loaded to conduct RWM and RCM. A slight difference from the pre-training stage is that slot entities are filtered out as keywords. It is worth noting that augmented data generated by two strategies explicitly contain diverse human expressions, which are learned from perturbed structure pre-training. Besides, we also generate coarse labels for two kinds of augmented data based on rules. Specifically, we label the infilling tokens as *O* while maintaining labels of other tokens. The case study (See Appendix D) shows that samples generated by semantic structure transferring generation can not only better fit spoken language perturbation, but also be more in line with human language diversity than those generated by rule-based methods.

[1]More details about the construction process of the perturbation corpus can be found at section 3.2

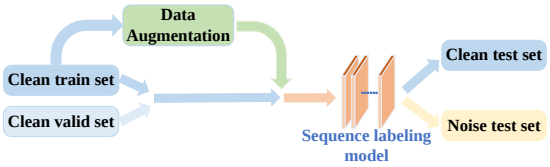

Figure 3: The process of downstream perturbation-robust slot filling task.

Consistency Processing Due to mixed perturbations in the upstream corpus, it is necessary to design a consistency processing to filter low-quality samples. Specifically, we train a tagging model with original training data and augmented samples. Then the model is used to predict labels for each augmented sentence. The labels which are consistent with the coarse labels and original labels are kept. The augmented samples filtered by consistency processing are mixed and input to the main task as the final augmented data.

Training and Inference As shown in figure 3, during the training stage, we first perform perturbed structure pre-training on the unsupervised language perturbation corpus to learn the contextual semantic structure and word distribution of perturbed data. We use the pre-trained model to obtain augmented data for the clean training dataset, and use all samples to train a perturbation-aware sequence labeling model. During the testing stage, we test the sequence labeling model on both clean and perturbed datasets.

3 Experiment

3.1 Dataset

RADDLE (Peng et al., 2020a) is a crowd-sourced diagnostic dataset to cover a broad range of real-world perturbations to study the robustness of end-to-end dialog system. We extract four kinds of realistic perturbed data from RADDLE and construct the slot filling dataset. In particular, the original dataset of the evaluation set in RADDLE is extracted from MultiWOZ (Lu et al., 2021). To introduce sufficient perturbed data for evaluating the model robustness against multiple perturbations, we extracted the clean user utterances and four kinds of perturbed utterances (Homophone, Simplify, Verbose and Paraphrase) from RADDLE. To be specific, **Homophone** perturbation comes from input text errors caused by recognition and synthesis errors. **Simplification** is generated by concise-

Methods	Clean	Homophone	Paraphrase	Verbose	Simplification	Overall
none	95.8	81.5 (-14.3)	87.5 (-8.3)	81.6 (-14.2)	85.3 (-10.5)	84 (-11.8)
Char-Random	96.0 (0.2)	84.1 (18.2%)	87.6 (1.2%)	83.2 (11.3%)	88.1 (26.7%)	85.8 (14.4%)
Word-Del	95.9 (0.1)	83.2 (11.9%)	89.3 (21.7%)	82.6 (7.0%)	87.5 (21.0%)	85.7 (15.4%)
Syn-Sub	96.1 (0.3)	83.5 (14.0%)	89.3 (21.7%)	82.2 (4.2%)	86.8 (14.3%)	85.5 (13.6%)
Word-Insert	95.8 (0.0)	81.2 (-2.1%)	88.2 (8.4%)	81.3 (-2.1%)	86.2 (8.6%)	84.2 (3.2%)
Hom-Sub	96.0 (0.2)	83.7 (15.4%)	89.3 (21.7%)	82.3 (4.9%)	87.7 (22.9%)	85.8 (16.3%)
NAT($\mathcal{L}_{aug}$)	96.0 (0.2)	84.3 (19.6%)	87.7 (2.4%)	82.8 (8.5%)	87.3 (19.0%)	85.5 (12.4%)
NAT($\mathcal{L}_{stabil}$)	96.0 (0.2)	83.9 (16.8%)	87.4 (-1.2%)	83.0 (9.9%)	87.3 (19.0%)	85.4 (11.1%)
PSSAT	96.2 (0.4)	**84.6 (21.7%)**	**90.1 (31.3%)**	**84.0 (16.9%)**	**89.3 (38.1%)**	**87.0 (27.0%)**
– RCM	96.2 (0.4)	83.8 (16.1%)	89.6 (25.3%)	83.5 (13.4%)	87.4 (20.0%)	86.1 (18.7%)
– RWM	96.3 (0.5)	83.3 (12.6%)	89.9 (28.9%)	83.8 (15.5%)	88.9 (34.3%)	86.5 (22.8%)
– CP	96.3 (0.5)	84.0 (17.5%)	90.0 (30.1%)	83.4 (12.7%)	88.3 (28.6%)	86.4 (22.2%)
– Pre-training	95.9 (0.1)	83.1 (11.2%)	89.4 (22.9%)	83.0 (9.9%)	86.9 (15.2%)	85.6 (14.8%)

Table 1: The performance (F1 score) of the PSSAT on RADDLE. For cells in *Baseline* row and *Clean test* column, the numbers in the parenthesis indicate the change of F1 score. For other cells, the numbers in the parenthesis indicate p_r. In Overall column, we calculate the average F1 and p_r of the four Spoken language perturbations, respectively. Both the best and the worst are marked, "−" denotes the model performance without a specific module. RWM, RCM, CP denotes Random Word Masking, Random Context Masking and Consistency Processing.

Method	Hom+App	Hom+Con	Con+App	Hom+Con+App
Baseline (LSTM)	47.9 (-46.0)	54.2 (-39.7)	73.4 (-20.5)	45.7 (-48.2)
best baseline	53.6 (12.4%)	61.1 (17.4%)	71.6 (-8.8%)	47.2 (3.1%)
PSSAT	59.6 (25.4%)	61.8 (19.1%)	78.3 (23.9%)	53.9 (17.0%)

Table 2: The performance of the best baseline and PSSAT on mixed perturbations.

word expression. On the contrary, **Verbose** refers to redundant expression. **Paraphrase** noise widely exists in our dataset, where users restate texts in different ways of expression according to their personal speaking habits. The training dataset consists of 61,117 clean data from four domains. We randomly select 5,000 data as the validation set. Our compared baselines and implementation details can be found in Section 3.4 and A.

3.2 Unsupervised Language Perturbation Corpus

In our perturbed structure pre-training stage, we employ two multi-modal datasets: Twitter-2015 (Zhang et al., 2018), Twitter-2017 (Lu et al., 2018). We only extract the corpus part and delete the useless details in sentences such as emoji and URL. We consider that the data on social media contains the real diversity of human expressions, and it is beneficial for the downstream generation to learn the knowledge of diverse human expressions in the pre-training stage.

3.3 Evaluation Metrics

We use F_1 score to measure the performance of the model. ${F_1}^c$, ${F_1}^p$ denote the performance on the clean and perturbed test set respectively. On this basis, we define (1) as Perturbation Recovery Rate (P_r) of a given perturbation-robust method m:

$$P_r = \frac{{F_1}^p_m - {F_1}^p_{\text{baseline}}}{{F_1}^c_{\text{baseline}} - {F_1}^p_{\text{baseline}}} \quad (1)$$

P_r indicates the improvement in performance of the model using the robust approach over the baseline model on the perturbed test set, as a percentage of the performance degradation of the baseline model due to the introduction of perturbation.

3.4 Implementation Details

For the upstream work, our model **PSSAT** is based on BART (Lewis et al., 2019), which is provided by the Huggingface Transformers[2]. The reason for choosing BART is that the pre-training tasks of BART include token masking and text filling, which is consistent with our PSSAT task. We set the batch size of BART to 8 and the pre-training takes an average of one hour for 10 epochs. The corresponding learning rates are set to 1e-5.

For the downstream work, we use two settings for perturbation-robust slot filling, Glove-Bi-LSTM and BERT-Bi-LSTM. Glove-6B-300d, char embedding and BERT-large-uncased are applied as the embedding layer. We take Bi-LSTM as the mainly analyzed model. The hidden size of Bi-LSTM is set to 128 and the dropout rate is set to 0.2. The transform probability p is set to 0.3. For all the experiments, we train and test our model on the 2080Ti GPU. It takes an average of 1.5 hours to run with 12 epochs on the training dataset.

[2]https://huggingface.co/docs/transformers

All experiments are repeated three times with different random seeds under the same settings. All the models are implemented with PyTorch (Paszke et al., 2019).

3.5 Main Results

Table 1 shows the main results of PSSAT compared to different baselines on the language perturbation dataset. The overall result of our PSSAT greatly outperforms the baseline by 27.0%. Especially, the P_r of paraphrase and simplification is about 40%, which is a remarkable enhancement. What's more, our method is not designed for any specific perturbation, but achieves the best results for various perturbations, which proves that our model not only improves the performance significantly, but generalizes better.

Ablation Studies. To better prove the effectiveness of the pre-training stage, we conduct ablation experiments. Table 1 illustrates the results that the model without RWM performs better than that without RCM, which shows that the change of context makes the semantic change more drastic. Meanwhile, all of RWM, RCM, CP and PSSAT without pre-training have a performance drop, which suggests that every part of design is necessary.

3.6 Mixed Perturbations Experiment

In real dialogue scenarios, mixed perturbations often appear in one input utterance at the same time. To verify the effectiveness of our method in more realistic scenarios, based on SNIPS (Coucke et al., 2018), we utilize TextFlint[3] (Gui et al., 2021) to introduce Homophone(Hom), Appendirr(App), ConcatSent(Con) and construct a mixed perturbations evaluation dataset [4]. As shown in Table 2 , the P_r of our PSSAT is over 20% against three different kinds of two-level perturbations, which far exceeds the best baseline (Hom-Sub). The model maintains an almost 17% P_r even with the joint disturbances from three-level perturbations, which shows the effectiveness and stability of our methods in real scenarios.

3.7 Error Analysis

We randomly selected 500 samples from all outputs and manually checking the error outputs for error analysis. Table 3 investigates 5 error types the model has made on the RADDLE. It can be seen that the number of PSSAT error outputs is less than the baseline in each category. Table 4 illustrates cases of each error type. Both the baseline model and PSSAT can correctly label clean text, but only PSSAT can correctly label texts with perturbation. After comprehensive analysis, the result shows that rote memorization of entity mention and contextual perturbation accounts for a large portion of the errors. Compared to the baseline, PSSAT can alleviate the problem of memorizing inherent patterns of entities and contexts.

Error Type	Baseline		PSSAT	
	Num	%	Num	%
Entity Location	12	20.0	9	18.8
Contextual Perturbation	16	26.7	11	22.8
Entity Mention	23	38.3	19	39.6
Others	9	15.0	9	18.8
Mixed Perturbation	11	–	9	–

Table 3: Error analysis on RADDLE.

Clean	are there any **museums** in the **centre** ?
Verbosity	could you please search for any **museums** in the town **centre** .
Baseline	O O O O O O B-type O O O O O
PSSAT	O O O O O O B-type O O O B-area O
Clean	i 'd like a **jamaican** restaurant please .
Simplification	find **jamaican** plz .
Baseline	O B-name I-name O
PSSAT	O B-food O O
Clean	i need to leave after **12:00** .
Homophone	i need to leave after **twelve** .
Baseline	O O O O O O O
PSSAT	O O O O O B-leave O
Clean	i need a booking for **4** people .
Paraphrase	i need seats for **4** .
Baseline	O O O O B-time O
PSSAT	O O O O B-people O
Clean	could you tell me which hotel around is **cheap** ?
Paraphrase	I want to proceed with the **cheaper** hotel .
Baseline	O O O O O O O B-type O
PSSAT	O O O O O O B-price B-type O

Table 4: The error cases. The bold texts are slot entities. Both wrong and correct labels are marked in red and green, respectively.

4 Conclusion

In this paper, we propose a perturbed semantic structure awareness transferring method for perturbation-robust slot filling task. Specifically, we design the perturbed structure pre-training and the semantic structure transferring generation to transfer the upstream learned semantic structure knowledge to downstream original training samples. Further, we filter low-quality samples through a consistency processing module. Sufficient experiments and error analysis demonstrate the effectiveness and generalization of our methods, and also prove that PSSAT alleviates the problem of memorizing inherent patterns of entities and contexts.

[3] http://textflint.io/

[4] We conducted single perturbation experiment on SNIPS. The results can be found in Appedix C

Acknowledgements

We thank all anonymous reviewers for their helpful comments and suggestions. This work was partially supported by MoE-CMCC "Artifical Intelligence" Project No. MCM20190701, National Key R&D Program of China No. 2019YFF0303300 and Subject II No. 2019YFF0303302, DOCOMO Beijing Communications Laboratories Co., Ltd.

References

David M Blei, Andrew Y Ng, and Michael I Jordan. 2003. Latent dirichlet allocation. *Journal of machine Learning research*, 3(Jan):993–1022.

Alice Coucke, Alaa Saade, Adrien Ball, Théodore Bluche, Alexandre Caulier, David Leroy, Clément Doumouro, Thibault Gisselbrecht, Francesco Caltagirone, Thibaut Lavril, Maël Primet, and Joseph Dureau. 2018. Snips voice platform: an embedded spoken language understanding system for private-by-design voice interfaces.

Claude Coulombe. 2018. Text data augmentation made simple by leveraging nlp cloud apis.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*.

Chih-Wen Goo, Guang Gao, Yun-Kai Hsu, Chih-Li Huo, Tsung-Chieh Chen, Keng-Wei Hsu, and Yun-Nung Chen. 2018. Slot-gated modeling for joint slot filling and intent prediction. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 2 (Short Papers)*, pages 753–757.

Tao Gui, Xiao Wang, Qi Zhang, Qin Liu, Yicheng Zou, Xin Zhou, Rui Zheng, Chong Zhang, Qinzhuo Wu, Jiacheng Ye, Zexiong Pang, Yongxin Zhang, Zhengyan Li, Ruotian Ma, Zichu Fei, Ruijian Cai, Jun Zhao, Xingwu Hu, Zhiheng Yan, Yiding Tan, Yuan Hu, Qiyuan Bian, Zhihua Liu, Bolin Zhu, Shan Qin, Xiaoyu Xing, Jinlan Fu, Yue Zhang, Minlong Peng, Xiaoqing Zheng, Yaqian Zhou, Zhongyu Wei, Xipeng Qiu, and Xuanjing Huang. 2021. Textflint: Unified multilingual robustness evaluation toolkit for natural language processing.

Keqing He, Weiran Xu, and Yuanmeng Yan. 2020a. Multi-level cross-lingual transfer learning with language shared and specific knowledge for spoken language understanding. *IEEE Access*, 8:29407–29416.

Keqing He, Yuanmeng Yan, and Weiran Xu. 2020b. Learning to tag oov tokens by integrating contextual representation and background knowledge. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 619–624.

Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Ves Stoyanov, and Luke Zettlemoyer. 2019. Bart: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. *arXiv preprint arXiv:1910.13461*.

Bill Yuchen Lin, Wenyang Gao, Jun Yan, Ryan Moreno, and Xiang Ren. 2021. Rockner: A simple method to create adversarial examples for evaluating the robustness of named entity recognition models. *arXiv preprint arXiv:2109.05620*.

Bing Liu and Ian Lane. 2015. Recurrent neural network structured output prediction for spoken language understanding. In *Proc. NIPS Workshop on Machine Learning for Spoken Language Understanding and Interactions*.

Bing Liu and Ian R. Lane. 2016. Attention-based recurrent neural network models for joint intent detection and slot filling. In *INTERSPEECH*.

Jiexi Liu, Ryuichi Takanobu, Jiaxin Wen, Dazhen Wan, Hongguang Li, Weiran Nie, Cheng Li, Wei Peng, and Minlie Huang. 2020. Robustness testing of language understanding in task-oriented dialog. *arXiv preprint arXiv:2012.15262*.

Di Lu, Leonardo Neves, Vitor Carvalho, Ning Zhang, and Heng Ji. 2018. Visual attention model for name tagging in multimodal social media. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1990–1999.

Hengtong Lu, Zhuoxin Han, Caixia Yuan, Xiaojie Wang, Shuyu Lei, Huixing Jiang, and Wei Wu. 2021. Slot transferability for cross-domain slot filling. In *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*, pages 4970–4979, Online. Association for Computational Linguistics.

George A Miller. 1995. Wordnet: a lexical database for english. *Communications of the ACM*, 38(11):39–41.

Milad Moradi and Matthias Samwald. 2021. Evaluating the robustness of neural language models to input perturbations. *arXiv preprint arXiv:2108.12237*.

Marcin Namysl, Sven Behnke, and Joachim Köhler. 2020. Nat: noise-aware training for robust neural sequence labeling. *arXiv preprint arXiv:2005.07162*.

Peiqing Niu, Zhongfu Chen, Meina Song, et al. 2019. A novel bi-directional interrelated model for joint intent detection and slot filling. *arXiv preprint arXiv:1907.00390*.

Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, et al. 2019. Pytorch: An imperative style, high-performance deep learning library. *Advances in neural information processing systems*, 32:8026–8037.

Baolin Peng, Chunyuan Li, Zhu Zhang, Chenguang Zhu, Jinchao Li, and Jianfeng Gao. 2020a. Raddle: An evaluation benchmark and analysis platform for robust task-oriented dialog systems. *arXiv preprint arXiv:2012.14666*.

Baolin Peng, Chenguang Zhu, Michael Zeng, and Jianfeng Gao. 2020b. Data augmentation for spoken language understanding via pretrained models. *arXiv e-prints*, pages arXiv–2004.

Liwen Wang, Rumei Li, Yang Yan, Yuanmeng Yan, Sirui Wang, Wei Yu Wu, and Weiran Xu. 2022a. Instructionner: A multi-task instruction-based generative framework for few-shot ner. *ArXiv*, abs/2203.03903.

Xiao Wang, Shihan Dou, Limao Xiong, Yicheng Zou, Qi Zhang, Tao Gui, Liang Qiao, Zhanzhan Cheng, and Xuanjing Huang. 2022b. Miner: Improving out-of-vocabulary named entity recognition from an information theoretic perspective. *arXiv preprint arXiv:2204.04391*.

Jason Wei and Kai Zou. 2019. Eda: Easy data augmentation techniques for boosting performance on text classification tasks. *arXiv preprint arXiv:1901.11196*.

Di Wu, Yiren Chen, Liang Ding, and Dacheng Tao. 2021. Bridging the gap between clean data training and real-world inference for spoken language understanding. *arXiv preprint arXiv:2104.06393*.

Qi Zhang, Jinlan Fu, Xiaoyu Liu, and Xuanjing Huang. 2018. Adaptive co-attention network for named entity recognition in tweets. In *Thirty-Second AAAI Conference on Artificial Intelligence*.

A Baselines

To simulate the input perturbation existing in realistic scenarios, we introduce five well-designed perturbation robust methods and a strong baseline:

Random Char Augmentation (Char-Random) is a character-level augmentation method that randomly adds, removes, and replaces characters in a token with a transformation probability p.

Random Word Deletion (Word-Del) aims to simulate the effect of simplification in input utterances in real-world scenarios (Wei and Zou, 2019). It randomly removes tokens with a probability p.

Random Word Insertion (Word-Insert) randomly insert words with probability p based on contextual embedding (Peng et al., 2020b). The method aims to model the effect of verbosity perturbation in input utterances.

Homophonic substitution (Hom-Sub) is designed for simulating word-level perturbation. We implement a homophone replacement dictionary, where words in the utterance are replaced by homophones with probability p.

Synonymous Substitution (Syn-Sub) is implemented based on WordNet's (Miller, 1995) synonymous thesaurus. We randomly select tokens in utterance with probability p for synonymous substitution (Coulombe, 2018). Note that our augmentations on training samples avoid slot words and only operate on contextual words.

Noise-Aware Training is proposed by (Namysl et al., 2020), which includes two Noise-Aware Training (NAT) objectives that improve robustness of sequence labeling performed on perturbed input. The data augmentation method trains a neural model using a mixture of clean and noisy samples, whereas the stability training algorithm encourages the model to create a noise-invariant latent representation.

B BERT Result on RADDLE

Table 5 shows the BERT-version results of PSSAT. Compared to several data augmentation methods, PSSAT method makes a great improvement in each field. The overall results are better than any type of data augmentation results. Furthermore, the whole PSSAT method outperforms the baseline by 18.8%. Similar to the results of LSTM, PSSAT also achieves the best results on each spoken language perturbation.

C SNIPS Single Perturbation Experiment

As shown in Table 6, we also explore the performance of various denoising methods on SNIPS dataset. Both entity mention and contextual semantics are corrupted in mixed multiple noise scenarios, resulting in a catastrophic degradation of model performance. The overall result combining the single-noise and multi-noise results achieves a 33% improvement.

D Case Study

Table 7 shows some samples generated by PSSAT in the way of RWM and RCM, respectively. It can be seen that the generated augmented samples are more in line with the Spoken language perturbation, while preserving the semantics of the original sentences.

Methods	Clean	Homophone	paraphrase	verbose	simplification	Overall
none	96.2	82.8(-13.4)	90.4(-5.8)	84.4(-11.8)	87.7(-8.5)	82.6(-13.6)
Char-Random	96.0 (-0.2)	85.0 (16.4%)	89.9 (-8.6%)	84.9 (4.2%)	88.1 (4.7%)	86.9 (4.2%)
Word-Del	95.9 (-0.3)	84.5 (12.7%)	90.0 (-6.9%)	84.5 (0.8%)	88.0 (3.5%)	86.8 (2.5%)
Word-Sub	96.3 (0.1)	84.1 (9.7%)	90.2 (-3.4%)	84.1 (-2.5%)	88.2 (5.9%)	86.7 (2.4%)
Word-Insert	96.3 (0.1)	84.3 (9.7%)	90.5 (1.7%)	83.9 (-4.2%)	88.5 (9.4%)	86.8 (4.2%)
Homophone	95.8 (-0.4)	**85.8 (22.4%)**	90.2 (-3.4%)	82.4 (-16.9%)	87.5 (-2.4%)	86.5 (-0.1%)
NAT($\mathcal{L}_{aug}$)	96.0 (0.2)	85.2 (17.7%)	90.5 (2.4%)	85.4 (8.3%)	88.0 (3.0%)	87.2 (7.9%)
NAT($\mathcal{L}_{stabil}$)	96.0 (0.2)	85.1 (16.8%)	90.3(-1.2%)	85.2 (6.6%)	88.0 (3.0%)	87.2 (6.3%)
PSSAT	96.4 (0.2)	85.6 (20.9%)	**91.5(19.0%)**	**85.8(11.9%)**	**89.7(23.5%)**	**88.1(18.8%)**
– RCM	96.6 (0.4)	84.7 (14.0%)	91.3 (15.5%)	85.1 (5.9%)	88.4 (8.2%)	87.4(10.9%)
– RWM	96.4 (0.2)	83.5(5.2%)	**91.5(19.0%)**	85.7(11.0%)	89.4(20.0%)	87.5(13.8%)
– Pre-training	95.9 (0.1)	83.1(2.2%)	90.7(4.9%)	84.9(4.4%)	88.2(5.6%)	86.7(4.3%)

Table 5: The performance (F1 score) of the PSSAT on RADDLE. For cells in *Baseline* row and *Clean test* column, the numbers in the parenthesis indicate the change of F1 score over the baseline (96.2), while for other cells, the numbers in the parenthesis indicate the perturbation recovery rate (p_r). In Overall column, we calculate the average F1 and p_r of the four Spoken language perturbations respectively.

Method	Clean	Hom	App	Concat	Overall
Baseline (LSTM)	93.9	62.2(-31.7)	71.2(-22.7)	85.0(-8.9)	72.8(-21.1)
Char-Random	93.7	**75.8(42.9%)**	74.0(12.3%)	85.2(2.2%)	78.3(19.1%)
Word-Del	93.8	61.6(-1.9%)	69.2(-8.8%)	85.3(3.4%)	72.0(-2.4%)
Word-Sub	93.8	65.7(11.0%)	73.3 (9.3%)	84.1 (-10.1%)	74.4(3.4%)
Word-Insert	92.8	63.9 (5.4%)	80.5 (41.0%)	82.1 (-32.6%)	75.5(4.6%)
Homephone	93.7	70.1 (24.9%)	72.8 (7.0%)	86.4 (15.7%)	76.4(15.9%)
NAT($\mathcal{L}_{aug}$)	93.6	69.1 (21.8%)	74.7 (15.3%)	85.5 (5.5%)	76.4 (14.2%)
NAT($\mathcal{L}_{stabil}$)	93.6	68.4 (19.6%)	74.3 (13.8%)	85.4 (4.7%)	76.0 (12.7%)
PSSAT	94.14	71.5(29.3%)	**82.7(50.7%)**	**86.7(19.1%)**	**80.3(33.0%)**

Table 6: The performance of the best baseline and PSSAT on mixed perturbations. Con, APP and Home stand for ConcatSent, Appendirr and Homophone, respectively.

	Ori.	Aug.
Text	can you please check for a **turkish restaurant** ? does it have **4 stars** ? **5** people for the train please . i 'll be leaving **kings lynn** after **13:15** .	so can you show me some **turkish restaurant** ? does that rated **4 stars** ? **5** tkts R needed please . i 'm gonna leave **kings lynn** @ **13:15** .
Word	ok , how about **scudamores punting company** then . how about a **museum** ? i am looking for a **hotel** please .	@HeyandIfhey , how about **scudamores punting company** then . how is a **museum** ? i am sorry for a **hotel** please .

Table 7: Some raw data and the corresponding enhanced data.

String Editing based Chinese Grammatical Error Diagnosis

Haihua Xie[1], **Xiaoqing Lyu**[2], **Xuefei Chen**[3]
[1]Yanqi Lake Beijing Institute of Mathematical Sciences and Applications / Beijing, China
[2]Wangxuan Institute of Computer Technology, Peking University / Beijing, China
[3]AI Research Center, Cloopen Group Holding Ltd / Beijing, China
haihua.xie@bimsa.cn, lvxiaoqing@pku.edu.cn, cxfff168@163.com

Abstract

Chinese Grammatical Error Diagnosis (CGED) suffers the problems of numerous types of grammatical errors and insufficiency of training data. In this paper, we propose a string editing based CGED model that requires less training data by using a unified workflow to handle various types of grammatical errors. Two measures are proposed in our model to enhance the performance of CGED. First, the detection and correction of grammatical errors are divided into different stages. In the stage of error detection, the model only outputs the types of grammatical errors so that the tag vocabulary size is significantly reduced compared with other string editing based models. Secondly, the correction of some grammatical errors is converted to the task of masked character inference, which has plenty of training data and mature solutions. Experiments on datasets of NLPTEA-CGED demonstrate that our model outperforms other CGED models in many aspects. The project of our approach is available at https://github.com/xiebimsa/se-cged.

1 Introduction

The traditional methods of Grammatical Error Diagnosis (GED) are pipeline-based, which has three main steps in series, including error detection, selection of candidate corrections, and ranking of candidate corrections (Lee et al., 2015). With the wide application of seq2seq models, seq2seq based GED models are designed to combine error detection and error correction (Wan et al., 2020; Kiyono et al., 2019). Considering the low efficiency of decoding in seq2seq and the uncontrollability of the generated results, researchers tend to apply the mode of sequence tagging for GED which outputs the correction operations instead of direct correction results.

The sequence tagging based GED methods are designed based on edit distance algorithms, such as Levenshtein distance (Levenshtein, 1966), which produces edit operations that are used to transform one sentence to the other. Because the intersection of an ungrammatical sentence and the corrected sentence is often large, it is feasible to transform them to each other based on a few edit operations. Several string editing based GED methods for English have been recently proposed, such as LaserTagger (Malmi et al., 2019), PIE (Awasthi et al., 2019) and GECToR (Omelianchuk et al., 2020).

The string editing based GED methods are highly accurate, controllable and interpretable (Rozovskaya and Roth, 2021; Parnow et al., 2021). Fine-grained data processing and edit tags designing are crucial. However, the size of the tag vocabulary is generally very large, which makes the current string editing based GED methods work better on closed sets, but difficult to deal with the complexities and long-tail problems on open datasets (Omelianchuk et al., 2020).

The current string editing based GED methods are mostly designed for processing English texts. Compared with English, CGED has to deal with more types of errors with less training samples (Li and Shi, 2021). Furthermore, it is difficult to detect Chinese grammatical errors based on explicit linguistic features due to the complexity and ambiguity of Chinese grammars (Rao et al., 2018). Chinese grammatical errors can be corrected by text edit operations in theory, but there are currently no papers detailing how to detect and correct Chinese grammatical errors based on string editing.

In this paper, we propose a string editing based CGED model, named SE-CGED, which detects and corrects Chinese grammatical errors based on text edit operations. The brief description of SE-CGED and the contributions of our work are presented below.

1) In SE-CGED, the process of CGED is separated into three main stages, including Seq2Edit, string editing, and masked charac-

Proceedings of the 29th International Conference on Computational Linguistics, pages 5335–5344
October 12–17, 2022.

ter inference. Detection of grammatical errors, and correction of different types of errors are handled in different stages.

2) Because the detection and correction of grammatical errors are separated, the size of the edit tag vocabulary is significantly smaller than that of other string editing based models. The smaller size of the tag vocabulary reduces the difficulty of sequence tagging and requires less training samples.

3) Based on the difficulty of modification, the correction of grammatical errors is separated into two steps to deal with different types of errors. The correction of some types of grammatical errors is uniformly converted to the task of masked character inference.

4) The existing methods and training data of masked language model are utilized for masked character inference, which helps improve the performance of CGED for certain error types.

This paper is organized as follows: Section 2 briefly reviews the literature on grammatical error diagnosis and CGED. Section 3 presents the basic concepts and the workflow of string editing based Chinese grammatical error diagnosis. Section 4 explains the model training and prediction of our proposed approach. Section 5 demonstrates the experimental setting and the experiment results. Section 6 discusses the limitations of our methodology and potential speculations of our works.

2 Background and related works

The target of a GED system is to give a correction for a sentence with grammatical errors. The seq2seq generative model is well suited for the GED task. Models of machine translation have been adopted in GED and achieved good results (Hotate et al., 2020). Different from machine translation, the language of the source sentence and the target sentence in GED is the same, and there is a large intersection between the two sequences. Therefore, the copy mechanism was applied for GED to reduce the computation of text generation (Zhao et al., 2019). Lack of training data is a major bottleneck for GED systems. To supplement training data, Ge et al., 2018 tried to use the n-best results of the seq2seq model to produce more pairs of correct and incorrect sentences, and Zhou et al., 2020 constructs training samples based on the uneven results of different translation models. However, the man-made training samples are significantly different from the real grammatical cases.

The mode of sequence tagging is recently prevailing for GED because it is more efficient in decoding and can produce more controllable results, compared with seq2seq models (Sun et al., 2021). Malmi et al., 2019 proposes LaserTagger, a general sequence tagging approach that casts text generation as a text editing task. LaserTagger sets three types of tags, including 'keep', 'delete' and 'P'. 'P' can be a word or a phrase that should be added before the current position. Awasthi et al., 2019 proposes PIE, a sequence tagging based GED model. In PIE, the types of edit operations include 'copy', 'append', 'delete' and 'replacement'. In the operations 'append' and 'replacement', the specific words that used to append or replace should be indicated. An iterative refinement sequence annotation method is proposed to predict the token level edit operations. Omelianchuk et al., 2020 proposes GECToR, a simple and efficient GED sequence tagger. GECToR specifically defines several tags of grammatical edit operations. The tag vocabulary size is 5000, including 4971 basic transformations (token-independent KEEP, DELETE and 1167 token-dependent APPEND, 3802 REPLACE) and 29 token-independent g-transformations.

3 Introduction of the SE-CGED model

3.1 The workflow of SE-CGED

As shown in Figure 1, the workflow of SE-CGED has the following main stages.

1) Seq2Edit. Seq2Edit is a sequence tagging model that outputs a sequence of edit tags for a sentence. Each edit tag indicates the edit operation that should be performed on the corresponding token in the input sentence.

2) String editing. This step is to perform string editing based on the tag sequence output in Seq2Edit. According to the types of tags (i.e., types of errors), different operations are performed on the tokens. The tokens can be deleted or transposed, and the tag 'MASK' may be added at certain positions.

3) Masked character inference. This step is to infer what the masked characters should be.

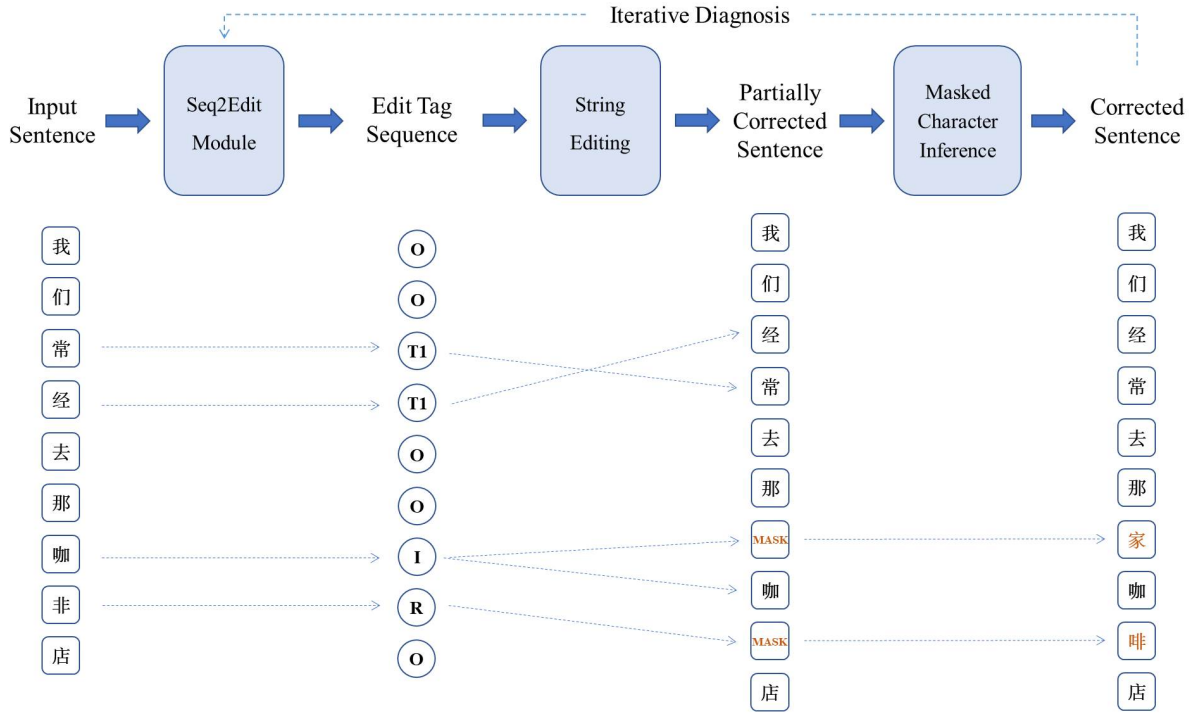

Figure 1: Workflow of SE-CGED

Grammar Error	Edit Operation	Edit Tags	Other Tags
Character redundancy	Deletion	D	/
Misuse of character	Replacement	R	R2
Character missing	Insertion	I	I2
Character disorder	Transposition	T1, T2, I-T2	/

Table 1: Grammatical error types and the corresponding edit operations in SE-CGED.

4) Iterative correction. In some cases, more than one iteration of the above steps may be performed to get the final corrections.

3.2 Error types and edit operations

The edit operations are critical for string editing based methods. The types of edit operations in SE-CGED are shown in Table 1. Because Chinese character is the basic unit of Chinese sentences, we use 'character' to represent Chinese character in the following chapters unless otherwise specified.

Error types: Chinese grammatical errors are categorized into four types.

1) Character redundancy.

In SE-CGED, redundant characters are labeled with 'D'. In the step of string editing, the characters labeled with 'D' will be directly deleted.

2) Misuse of character.

A misused character should be replaced by another character. In SE-CGED, misused characters are labeled with 'R'. In the step of string editing, a character labeled with 'R' is converted to a 'MASK' tag. The correction of character misusing is performed in the step of masked character inference. In some cases, a character should be replaced by a word consisting of more than one characters. However, the tag 'R' doesn't indicate the number of characters in the substitute. This problem is handled in the stage of iterative correction.

3) Character missing.

The missing word should be inserted into the sentence. The token before which a word should be inserted is labeled with 'I'. Similar to character misusing, the correction of character missing is performed in the step of string editing, masked character inference and iterative correction.

4) Character disorder.

A character in a wrong position should be moved to a right position. There are two types of character disorder.

a) Disorder of characters within a word or a phrase, such as “特普朗” (disorder of “特朗普” (‘Trump’)). A character with such type of disorder is labeled with ‘T1’ (transposition type 1).

The corresponding correction is to swap the positions of the first half and the second half of the sequence labeled with ‘T1’, and it is done in the step of string editing. Therefore, the number of consecutive tokens labeled with ‘T1’ is even.

b) Improper collocation between words.

Each character in a word with such type of disorder is labeled with ‘T2’ (transposition type 2). To give a direct correction suggestion for such error, a tag ‘I-T2’ is given to the token before which the words labeled with ‘T2’ should be moved to. For instance, the tagging sequence of “我去了公园在中午。” (“I went to the park in the noon.”) is ‘O / I-T2 / O / O / O / T2 / T2 / T2 / O’.

In the step of string editing, the subsequence labeled with ‘T2’ will be moved to the position before the token labeled with ‘I-T2’.

In the training samples, the tag ‘T2’ and ‘I-T2’ always appear together or neither in a tag sequence. To ensure that ‘T2’ and ‘I-T2’ also appear together or neither in the tagging result of a trained Seq2Edit model, the decoding module of Seq2Edit is designed to output the top scoring tag sequence which contains both of ‘T2’ and ‘I-T2’ or neither.

Separation of detection and correction: The correction of character misusing and character missing is not performed in the step of string editing. Instead, the character labeled with ‘R’ (or ‘I’) is replaced (or added) with a ‘MASK’ tag. And in the step of masked character inference, the replacement of the misused character (or the character to be inserted) will be given.

In the existing string editing based GED models, the detection and correction of grammatical errors are both performed in the step of generating edit tags. If a misused word is detected, the edit tag indicates the word that is used to replace it. There are two main drawbacks of such mode.

- The size of the tag vocabulary is huge, thus a large amount of training samples are required.
- The correction options are limited.

The above problems can be overcome in SECGED. Because Seq2Edit only gives the types of grammatical errors, the size of the tag vocabulary is only 7. The masked character inference based correction provides more options for correcting the errors of character missing and misusing.

Iterative correction: Iterative correction is performed for three reasons. 1) some errors may not be detected in one iteration. 2) new errors may occur after an iteration of correction. 3) character replacement and character insertion may need many iterations to obtain the result. A maximum number of iterations can be set, such as 3.

The first two reasons are easy to understand. For the third one, we take character misusing as an example. In practical scenarios, a misused character sometimes should be replaced by two or more characters. For example,

- Incorrect: 跟两个方法的比较 (“With the comparison of the two methods”).
- Correct: 通过两个方法的比较 (“Based on the comparison of the two methods”).

“跟” (‘with’) is replaced by “通过” (‘based on’). However, the tag ‘R’ doesn’t contain the information of how many characters should be used to replace the misused character. To deal with such problem, a mechanism of iterative correction is applied.

i. The token labeled with ‘R’ at the position p is converted to ‘MASK’ in string editing.

ii. Masked character inference is performed, and a corrected sentence is obtained.

iii. Seq2Edit is performed again on the corrected sentence.

iv. If an error still occurs at the position p.

   a. The character is labeled with ‘R2’.
   b. The token labeled with ‘R2’ is converted to ‘MASK MASK’ in string editing.
   c. Masked character inference is performed, and a new corrected sentence is obtained.

Based on the procedure of iterative correction, if the first iteration gives a wrong correction (for instance, ‘跟’ is replaced by ‘和’), a second iteration is run to replace the character with two characters. Replacing a character with one or two characters can cover most of the cases of character misusing. Thus, we do not try to replace a character with more than two characters.

Tag vocabulary: The tag vocabulary of Seq2Edit in SE-CGED contains 7 main tags, including 'D', 'R', 'I', 'T1', 'T2', 'I-T2' and 'O'. The other two tags, 'R2' and 'I2', are used in the stage of iterative correction, and they are not considered in Seq2Edit. Compared with the previous Seq2Edit models, such as GECToR (Omelianchuk et al., 2020), the size of tag vocabulary of SE-CGED is significantly reduced.

3.3 Converting the tags of Damerau Levenshtein to the tags of Seq2Edit

Seq2Edit Tag Generation based on DL distance

Require: $\mathbf{x} = (x_1, \ldots, x_n)$, $\mathbf{y} = (y_1, \ldots, y_n)$
diffs ← DamerauLevenshtein-Dist ($\mathbf{x}$, $\mathbf{y}$)
for each **insert**(a, b) tag in diffs
  s = all tokens inserted before a
  if token a is replaced by a'
    $s = s + a'$
    token a is labeled with 'D'
  if s is deleted or replaced
    token a is labeled with 'I-T2'
    tokens in s are labeled with 'T2'
  otherwise, token a is labeled with 'I'
for each **replace**(a, b) tag in diffs
  if token a has been labeled, skip
  otherwise
    $s = a$ + following tokens labeled with 'replace'
    if the front & rear of s are mutually replaced
      tokens in s are labeled with 'T1'
    otherwise, token a is labeled with 'R'
for each **delete**(a, b) tag in diffs
  if token a has been labeled, skip
  otherwise, token a is labeled with 'D'
for each **swap**(a, b) tag in diffs
  if token a has been labeled, skip
  otherwise, token a and b are labeled with 'T1'
for each token t in $\mathbf{x}$
  if t is not labeled
    t is labeled with 'O'

Figure 2: Seq2Edit tag generation based on Damerau Levenshtein distance.

Most of the training samples of CGED only contain parallel pairs of incorrect and correct sentences. Some training data (Rao et al., 2020) gives the steps of corrections, but they are incompatible with the tagging system of our approach.

To prepare the training samples of the Seq2Edit module, we first apply the Damerau Levenshtein algorithm (Brill and Moore, 2000) to analyze the difference between ungrammatical sentences and the corresponding correct sentences, and generates DL tags which indicates the steps of modifying the ungrammatical sentence to obtain the correct one. Then we convert the DL tags to the tags of Seq2Edit.

There are four types of Damerau Levenshtein tags, which are 'delete', 'replace', 'insert' and 'swap'. Some examples of these four tags are given below (suppose the following operations are used to transform sentence A to sentence B).

- ('delete', 5, 5), to delete the fifth token in A.
- ('replace', 5, 6), to replace the fifth token in A with the sixth token in B.
- ('insert', 5, 6), to insert the sixth token in B before the fifth token in A.
- ('swap', 5, 6), to swap the positions of the fifth token and the sixth token in A.

The pseudo code of converting the DL tags to the Seq2Edit tags is shown in Figure 2. In the conversion process, a token may be given more than one tags. However, in a training sample of Seq2Edit, a token can only have one tag. To deal with this problem, an intermediate transformation is performed to split one parallel pair of incorrect and correct sentences into two or more training samples of Seq2Edit, to ensure there is only one tag for each token. The following steps are performed to generate the training samples of Seq2Edit based on a parallel pair of incorrect and correct sentences.

For $(\boldsymbol{x}, \boldsymbol{y})$, an ungrammatical sentence $\boldsymbol{x}$ and the corresponding correction $\boldsymbol{y}$.

(1) Run the algorithm of Seq2Edit Tag Generation based on $\boldsymbol{x}$ and $\boldsymbol{y}$.

(2) If there is a token in $\boldsymbol{x}$ has more than one tags,
  a. Generate a training sample $(\boldsymbol{x}, \boldsymbol{t_1})$, $\boldsymbol{t_1}$ is the set of first tags of each token,
  b. Modify $\boldsymbol{x}$ based on tags in $\boldsymbol{t_1}$,
  c. Rerun the process from step (1).

(3) Otherwise, let $(\boldsymbol{x}, \boldsymbol{t})$ be a training sample, $\boldsymbol{t}$ is the set of tags of each token.

4 Training and application of SE-CGED

4.1 Generation of the training data

Training data of Seq2Edit: According to the description in the previous section, a part of training

data of Seq2Edit can be generated based on the Damerau Levenshtein distance of the existing training samples of CGED. The other part of training data of Seq2Edit is generated through data augmentation. Different types of synthetic ungrammatical sentences are created based on our summarized patterns of grammatical errors as shown in Table 2.

1) Character missing and character redundancy. Several patterns of character missing and character redundancy are summarized based on the real training samples. Table 2 gives three example patterns. For instance, for the first example pattern of character missing, a synthetic ungrammatical sentence is created by removing the character '了' after a verb. Before creating a synthetic sentence, word segmentation is performed on the original sentence. We do not remove a character within a word for creating an error of character missing.

2) For the type of character misusing, a part of synthetic samples are created by replacing a word with its synonyms, selected from a synonyms toolkit which is based on word2vec (Mikolov et al., 2013) similarity calculations, and the others are created by replacing a character with another character with the similar pronunciation or shape, selected from a prepared confusion set.

3) For the type of character disorder, most of the synthetic samples are created by changing the position of one word, and the others are created by swapping the positions of two adjacent characters within a word.

Training data of masked character inference: The training of masked character inference is performed in the step of masked LM pretraining of BERT (Devlin et al., 2019). The training samples of word replacement and word insertion are used here as part of training samples.

Other training samples are produced by randomly masking words in sentences. One-character words and two-character words are randomly selected and masked. Besides, two consecutive words with a high PMI value can also be masked together. Given two tokens w_1 and w_2, the PMI of the bigram 'w_1w_2' is:

$$PMI(w_1w_2) = \log \frac{p(w_1w_2)}{p(w_1)p(w_2)} \quad (1)$$

$p(w)$ is the probability of the occurrences of token w in the corpus.

4.2 Model training and model inference

Model of masked character inference: The task of masked character inference is performed based on the masked language model (MLM) of BERT. During model pre-training of BERT, the task of masked character inference is performed, but the task of next sentence prediction is skipped.

MLM consists of a multi-layer transformer encoder and a feed-forward neural network. At the stage of inference, the partially masked sequence is fed into MLM to produce a 1*V vector $\{C_1, C_2, ..., C_{V-1}, C_V\}$ at each position with token '[MASK]'. C_i is the confidence value of fitting the ith character in the vocabulary to the targeted position calculated in equation (2).

$$Conf(e_i = c|\vec{x}) = softmax(W_c^T h_i) \quad (2)$$

e_i is a masked character in $\vec{x}$ and c is a character in the vocabulary. h_i is the multi-layer transformer embedding of e_i, and W_c is a matrix of parameters regarding c in the feed-forward neural network.

Model of Seq2Edit: The Seq2Edit model is built on BERT for sentence embedding and a CRF layer for sequence tagging. The BERT model used here is the same as that used in masked character inference, and the CRF layer is trained based on the training data of Seq2Edit to learn how to detect and identify grammatical errors in a sentence.

Seq2Edit is to output a tag sequence L for the input sentence S. Let $S = \{s_1, s_2, ..., s_K\}$ and K is the length of S. $U = \{u_1, u_2, ..., u_K\}$ is the embedding result of S from BERT, and u_i is the embedding of s_i. U is input into the CRF layer and $L = \{l_1, l_2, ..., l_K\}$ is the tagging result. l_i is the tag for s_i. The confidence of labeling U with L is shown in the following equation.

$$P(L, U) = \sum_{k=1}^{K} (H_{(l_{k-1}, l_k)} + \varphi(l_k, u_k)) \quad (3)$$

H is the probability transition matrix of tags. H is 7*7 matrix because there are seven tags in the tag set. $H_{(l_{k-1}, l_k)}$ is the transition probability from tag l_{k-1} to tag l_k. Additionally, $\varphi(l_k, u_k)$ is the score of labeling u_k as l_k. φ is a 7*V matrix where V is the size of the vocabulary and 7 is the size of the tag set. Both H and φ are randomly initialized and updated during the training process.

Error Type	Ratio in real training data	Example patterns of grammatical errors
Character missing	0.275	1) Missing '了' after a verb or at the end of a sentence 2) Missing pronoun at the beginning of a sentence 3) Missing a conjunction word
Character redundancy	0.234	1) Reduplication of an adjective, verb, pronoun or preposition 2) Redundant '是' before an adjective or adverb 3) Redundant '了' after a verb
Misuse of character	0.422	1) Misuse of synonyms 2) Misuse of characters with similar pronunciation or shape
Character disorder	0.069	1) Wrong position of an adverb, conjunction or preposition 2) Wrong order of adjacent verbs and adverbs 3) Disorder of adjacent characters within a verb, adjective or noun

Table 2: Example patterns of synthetic grammatical errors.

Model	FPR	Detection Level			Identification Level			Position Level			Correction Level (Top1 & Top3)		
		Pre.	Rec.	F1	Pre.	Rec.	F1	Pre.	Rec.	F1	Pre.	Rec.	F1
Flying	0.3257	0.9101	0.8800	0.8948	0.7320	0.6011	0.6601	0.4715	0.3536	0.4041	0.2290 0.2290	0.1575 0.1575	0.1867 0.1867
YD-NLP	**0.2182**	**0.9357**	0.8478	0.8896	**0.7711**	0.5577	0.6473	0.5011	0.2995	0.3749	**0.3386** **0.3217**	0.1259 0.1333	0.1836 0.1885
Orange-Plus	0.2606	0.9252	0.8600	0.8914	0.7230	0.6287	0.6726	0.4428	0.3610	0.3977	0.1780 0.0934	0.1536 0.2283	0.1649 0.1325
SE-CGED	0.2769	0.9111	**0.8962**	**0.9036**	0.7059	**0.6557**	**0.6799**	**0.5060**	**0.4645**	**0.4844**	0.2331 0.2417	**0.2077** **0.2186**	**0.2197** **0.2296**

Table 3: Performance of different models for CGED.

5 Experiments and Analysis

5.1 Datasets in the experiment

The dataset of NLPTEA-CGED evaluations is used in our experiment. The training sets and test sets of NLPTEA-CGED evaluations in the year of 2014 to 2021 are collected. The NLPTEA-CGED data provides parallel pairs of ungrammatical and correct sentences, and the ungrammatical samples are collected from essays written by learners of Chinese as a foreign language. A sentence in the dataset may not have grammatical errors, or it may have one or more errors.

In order to make a fair comparison with other methods, our model was trained and tested based on the training dataset and test dataset of the NLPTEA-CGED 2020 evaluation (Rao et al., 2020), and the performance was compared with the submitted results of the NLPTEA-CGED 2020 evaluation. Because the evaluation allows the participants to use external data, we pretrained our model based on the training data and test data in the NLPTEA-CGED evaluation of other years. There are totally 182,486 grammatical errors collected from NLPTEA-CGED evaluations.

Meanwhile, 247,185 synthetic training samples are created for Seq2Edit based on the data argumentation method introduced in section 4.1. The source sentences used for generating synthetic training samples are collected from *The People's Daily*[1]. The ratios of different types of synthetic grammatical errors are approximate to the ratios in real data shown in Table 2. Besides, we took several measures to make the distribution of synthetic data close to that of real data. For instance, the errors of character misusing in NLPTEA-CGED are significantly different from those made by Chinese native

[1]http://paper.people.com.cn

		FPR	Detection Level			Identification Level			Position Level			Correction Level (Top1 & Top3)		
			Pre.	Rec.	F1	Pre.	Rec.	F1	Pre.	Rec.	F1	Pre.	Rec.	F1
Iterations	2	0.2704	0.8556	0.8415	0.8485	0.6471	0.6011	0.6233	0.4464	0.4098	0.2730	0.1840 0.1964	0.1639 0.1776	0.1734 0.1865
	4	0.2834	0.9091	0.9016	0.9053	0.7051	0.6598	0.6817	0.5052	0.4672	0.4855	0.2340 0.2420	0.2090 0.2202	0.2208 0.2306
	5	0.2899	0.9089	0.9021	0.9055	0.7021	0.6611	0.6810	0.5031	0.4689	0.4854	0.2324 0.2413	0.2115 0.2217	0.2215 0.2311
W/O synthetic data		0.2117	0.8958	0.8689	0.8821	0.7143	0.6284	0.6686	0.5161	0.4372	0.4734	0.2406 0.2443	0.1940 0.2036	0.2148 0.2221

Table 4: Performance of the ablation study.

speakers, which usually occur between characters with the similar pronunciation or shape. Foreigners learning Chinese often misuse synonyms or relevant words. Thus, for the type of character misusing, 60% of the synthetic samples are created by replacing a word with its synonyms, and 40% are created by replacing a character with another character with the similar pronunciation or shape, selected from a prepared confusion set.

5.2 Evaluation metrics and comparison methods

The performance of grammatical error diagnosis systems is evaluated from the following aspects.

- Detection-level, to detect whether the statement contains grammatical errors.
- Identification-level, to determine the types of the grammatical errors.
- Position-level, to determine the position of the grammatical errors.
- Correction-level, to give the correction of the grammatical errors.

The measurement method of the above metrics can be referred to (Rao et al., 2020). We select three representative models with good performance in the NLPTEA-CGED 2020 evaluation as the comparison models, including Flying, YD-NLP and Orange-Plus. FLYing has the highest F1 score, YD-NLP has the highest precision score, and Orange-Plus are more balanced between the precision and recall rate. The maximum numbers of iterations in SE-CGED was set to 3. The performance of different models is shown in Table 3.

Our proposed model, SE-CGED, achieves a better performance in the identification level, and a much better performance in the position level and correction level. The performance of the position level does not decay much compared with that of the identification level because the Seq2Edit module gives the type and position of the grammatical errors at the same time. Meanwhile, our method is more balanced in terms of accuracy and recall, compared with the comparison methods, in which the recall rates are significantly lower.

5.3 Ablation analysis

The structure of our method is pipeline based, and all the stages are indispensable. Therefore, the ablation analysis in our experiment was performed by making minor adjustments to the model. Firstly, the maximum numbers of iterations was set to 2, 4 and 5 respectively in the stage of iterative correction. Secondly, synthetic data was not used to train the module of Seq2Edit. The results of ablation analysis are shown in Table 4.

When the number of iterations is set to 2, the performance is significantly reduced compared with that when the number of iterations is 3. The reason is that lots of errors have not been detected or corrected after two iterations. On the other hand, iterations more than 3 lead to a slight improvement in performance, but the improvement is very limited. Thus, it is reasonable to set the number of iterations to 3. Training the model without the synthetic data leads to lower recall rates. Out synthetic training data is helpful to detect errors that do not appear in the real training corpus.

6 Conclusions

In this paper, we proposed a phased string editing based approach for Chinese grammatical errors diagnosis. The detection and correction of grammatical errors are performed based on edit tags and masked character inference. In this way, different types of grammatical errors can be handled in the same way, and the training data is significantly reduced. Experiments on the dataset of NLPTEA-CGED evaluations show our proposed approach performs better than the traditional methods in performance and speed.

Because of the complexity of Chinese grammars, there are still many deficiencies in our current work, and the detection of error type and location is not accurate. In future research, we will further improve the rationality of data argumentation and make the synthetic error samples more consistent with practical grammatical errors. In addition, we will try to import linguistic features to further improve the detection and correction of Chinese grammatical errors.

Acknowledgements

This work is supported by the National Key Research and Development Program (No. 2019YFB1406302), the projects of National Natural Science Foundation of China (No. 61472014, No. 61573028 and No. 61432020), the Natural Science Foundation of Beijing (No. 4142023) and the Beijing Nova Program (XX2015B010). We also thank all the anonymous reviewers for their valuable comments.

References

Abhijeet Awasthi, Sunita Sarawagi, Rasna Goyal, Sabyasachi Ghosh, and Vihari Piratla. 2019. Parallel iterative edit models for local sequence transduction. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 4260–4270, Hong Kong, China. Association for Computational Linguistics.

Eric Brill and Robert C. Moore. 2000. An improved error model for noisy channel spelling correction. In *Proceedings of the 38th Annual Meeting of the Association for Computational Linguistics*, pages 286–293, Hong Kong. Association for Computational Linguistics.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, Minneapolis, Minnesota. Association for Computational Linguistics.

Tao Ge, Furu Wei, and Ming Zhou. 2018. Fluency boost learning and inference for neural grammatical error correction. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1055–1065, Melbourne, Australia. Association for Computational Linguistics.

Kengo Hotate, Masahiro Kaneko, and Mamoru Komachi. 2020. Generating diverse corrections with local beam search for grammatical error correction. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 2132–2137, Barcelona, Spain (Online). International Committee on Computational Linguistics.

Shun Kiyono, Jun Suzuki, Masato Mita, Tomoya Mizumoto, and Kentaro Inui. 2019. An empirical study of incorporating pseudo data into grammatical error correction. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 1236–1242, Hong Kong, China. Association for Computational Linguistics.

Lung-Hao Lee, Liang-Chih Yu, and Li-Ping Chang. 2015. Overview of the NLP-TEA 2015 shared task for Chinese grammatical error diagnosis. In *Proceedings of the 2nd Workshop on Natural Language Processing Techniques for Educational Applications*, pages 1–6, Beijing, China. Association for Computational Linguistics.

Vladimir I Levenshtein. 1966. Binary codes capable of correcting deletions, insertions, and reversals. *Soviet physics doklady*, 10:707–710.

Piji Li and Shuming Shi. 2021. Tail-to-tail non-autoregressive sequence prediction for Chinese grammatical error correction. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 4973–4984, Online. Association for Computational Linguistics.

Eric Malmi, Sebastian Krause, Sascha Rothe, Daniil Mirylenka, and Aliaksei Severyn. 2019. Encode, tag, realize: High-precision text editing. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 5054–5065, Hong Kong, China. Association for Computational Linguistics.

Tomas Mikolov, Wen-tau Yih, and Geoffrey Zweig. 2013. Linguistic regularities in continuous space

word representations. In *Proceedings of the 2013 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 746–751, Atlanta, Georgia. Association for Computational Linguistics.

Kostiantyn Omelianchuk, Vitaliy Atrasevych, Artem Chernodub, and Oleksandr Skurzhanskyi. 2020. GECToR – grammatical error correction: Tag, not rewrite. In *Proceedings of the Fifteenth Workshop on Innovative Use of NLP for Building Educational Applications*, pages 163–170, Seattle, WA, USA → Online. Association for Computational Linguistics.

Kevin Parnow, Zuchao Li, and Hai Zhao. 2021. Grammatical error correction as GAN-like sequence labeling. In *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*, pages 3284–3290, Online. Association for Computational Linguistics.

Gaoqi Rao, Qi Gong, Baolin Zhang, and Endong Xun. 2018. Overview of NLPTEA-2018 share task Chinese grammatical error diagnosis. In *Proceedings of the 5th Workshop on Natural Language Processing Techniques for Educational Applications*, pages 42–51, Melbourne, Australia. Association for Computational Linguistics.

Gaoqi Rao, Erhong Yang, and Baolin Zhang. 2020. Overview of NLPTEA-2020 shared task for Chinese grammatical error diagnosis. In *Proceedings of the 6th Workshop on Natural Language Processing Techniques for Educational Applications*, pages 25–35, Suzhou, China. Association for Computational Linguistics.

Alla Rozovskaya and Dan Roth. 2021. How good (really) are grammatical error correction systems? In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*, pages 2686–2698, Online. Association for Computational Linguistics.

Xin Sun, Tao Ge, Furu Wei, and Houfeng Wang. 2021. Instantaneous grammatical error correction with shallow aggressive decoding. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 5937–5947, Online. Association for Computational Linguistics.

Zhaohong Wan, Xiaojun Wan, and Wenguang Wang. 2020. Improving grammatical error correction with data augmentation by editing latent representation. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 2202–2212, Barcelona, Spain (Online). International Committee on Computational Linguistics.

Wei Zhao, Liang Wang, Kewei Shen, Ruoyu Jia, and Jingming Liu. 2019. Improving grammatical error correction via pre-training a copy-augmented architecture with unlabeled data. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 156–165, Minneapolis, Minnesota. Association for Computational Linguistics.

Wangchunshu Zhou, Tao Ge, Chang Mu, Ke Xu, Furu Wei, and Ming Zhou. 2020. Improving grammatical error correction with machine translation pairs. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 318–328, Online. Association for Computational Linguistics.

The Fragility of Multi-Treebank Parsing Evaluation

Iago Alonso-Alonso, David Vilares and Carlos Gómez-Rodríguez
Universidade da Coruña, CITIC
Departamento de Ciencias de la Computación y Tecnologías de la Información
Campus de Elviña s/n, 15071
A Coruña, Spain
{iago.alonso,david.vilares,carlos.gomez}@udc.es

Abstract

Treebank selection for parsing evaluation and the spurious effects that might arise from a biased choice have not been explored in detail. This paper studies how evaluating on a single subset of treebanks can lead to weak conclusions. First, we take a few contrasting parsers, and run them on subsets of treebanks proposed in previous work, whose use was justified (or not) on criteria such as typology or data scarcity. Second, we run a large-scale version of this experiment, create vast amounts of random subsets of treebanks, and compare on them many parsers whose scores are available. The results show substantial variability across subsets and that although establishing guidelines for good treebank selection is hard, it is possible to detect potentially harmful strategies.

1 Introduction

A limitation in NLP evaluation lies in the association between solving a dataset *versus* solving a task. Datasets are domain-specific, their sizes differ and they are only available for a handful of languages and cultures (Hershcovich et al., 2022). Yet, we often ignore that the chances that these results generalize in the real world are scarce. In this context, the conclusions extracted from a single dataset should be taken with caution.

For dependency parsing, the Universal Dependencies framework (UD; Zeman et al., 2020) mitigates some of these issues. For instance, version 2.8 of UD includes 202 treebanks and 114 languages covering diverse linguistic typologies, treebanks with different amounts of data, and domains. Paradoxically, this also complicates decisions when it comes to comparing dependency parsers in multilingual environments, which can be summarized as: *how to choose a small but representative set of treebanks?* Although there are shared tasks (Zeman et al., 2017, 2018) that do consider experiments over a wide set of treebanks and help understand parsing models, such setups do not usually stick when the shared tasks end, and authors often run their models only in a handful of treebanks (de Lhoneux and Nivre, 2016; Ma et al., 2018; Kulmizev et al., 2019, *inter alia*). This mostly happens for justified reasons: lack of computational resources to train the models in a reasonable amount of time, energy usage concerns, difficulties to summarize large experiments, or interest in specific phenomena (e.g. non-projectivity). Thus, a good treebank selection strategy is crucial to reduce the chances of selecting an unrepresentative subset of treebanks, which could lead to weak conclusions. Furthermore, even when using the whole UD collection is viable, treebank selection can still be relevant as UD is not a representative sample of languages (e.g., 62 out of the 114 languages in v2.8 are Indo-European), so coarse-grained measures like averages over all treebanks may be misleading.

Contribution We hypothesize that using a single subset of treebanks can be a weak approach to extract conclusions about the performance of parsers and their rankings. To test so, we design two experiments. First, we choose representative models of different paradigms: a graph-based (Dozat et al., 2017), a transition-based (Fernández-González and Gómez-Rodríguez, 2019), and a sequence tagging (Strzyz et al., 2019) parser; and evaluate them on a few subsets defined in the literature, looking for different trends. Then, we redefine the previous experiment on a large scale. We take the output of dozens of parsers on the treebanks used at the UD CoNLL 2018 shared task (Zeman et al., 2018) to study the variability of parsing rankings over a million of fixed-size, randomly generated subsets.

2 Related work

The appropriateness of experimental setups for parsing evaluation has been studied in recent years from different perspectives.

Some authors have focused on determining what

Proceedings of the 29th International Conference on Computational Linguistics, pages 5345–5359
October 12–17, 2022.

are the treebank particularities that make some of them easier to parse than others. For instance, the size of the training set is widely known to be an important factor to obtain accurate results in dependency parsing (Dehouck and Denis, 2019; Vania et al., 2019). Other aspects such as domain similarity (Wisniewski and Yvon, 2019) or annotation similarity (Dredze et al., 2007; Cohen et al., 2012) between the training and test sets have also been studied, showing that they can greatly affect the performance of parsers. Other particularities that can also affect the performance on a treebank are linguistic variation (Nivre et al., 2007), annotation criteria (Kübler et al., 2008; Rosa, 2015), arc direction (Rehbein et al., 2017), average dependency length (Gulordava and Merlo, 2016), non-projectivity (Kuhlmann and Nivre, 2010), morphological richness (Tsarfaty et al., 2013) or information-theoretic metrics (Corazza et al., 2013), among other factors.

Although not specifically for parsing but NLP, Gorman and Bedrick (2019) and Søgaard et al. (2021) comment that the way data is split can play a role on test results, and thus on conclusions. Extrapolating this to parsing, it would suggest that some parsers could obtain better results for certain treebanks just due to data splitting decisions, and not due to a linguistic motivation that would explain a given language being harder to parse than other. Recently, Søgaard (2020) studied the influence of overlap between trees in training and test sets in a given split, and concluded that (the amount of) graph isomorphism between the training and test set trees partially explains why some treebanks are easier or harder to parse than others. However, Anderson et al. (2021) replicated the study, controlling for covariants, and proved that much of this observation is explained by relevant covariants like treebank size and mean test sentence length.

Another line of research more related to our work involves the studies that compare how different parsing algorithms behave on the same held-out test sets. McDonald and Nivre (2007, 2011) showed that non-neural transition-based and graph-based parsers perform overall similarly, but produce different types of errors, with transition-based parsers being weaker for long dependencies and graph-based parsers weaker for shorter, more local ones. Relatedly, de Lhoneux et al. (2017a) compared a neural and non-neural transition based parser, showing that the former is not only clearly better at longer dependencies, but that it also needs less training data to parse effectively. Kulmizev et al. (2019) replicated the work by McDonald and Nivre for neural versions of those parsers and, contrarily, demonstrated that the contextualization of the input vectors with recurrent networks results into both types of parsers showing a much more homogeneous behavior. Also related to this, Anderson and Gómez-Rodríguez (2020b) showed how different transition-based algorithms are prone to outperform others on a specific treebank according to their inherent dependency displacement biases.

To the best of our knowledge, there have been only two papers in the literature that specifically focus on presenting methodologies to choose a suitable set of treebanks for parsing evaluation, both centered on UD and with the goal of obtaining a small sample of treebanks that is representative of the full UD collection (not necessarily of human languages as a whole). de Lhoneux and Nivre (2016); de Lhoneux et al. (2017b) do so by manually selecting treebanks to enforce typological diversity as well as representativity in other relevant aspects for parsing, like projectivity or treebank size. In turn, Schluter and Agić (2017) take an automatic, quantitative approach, obtaining a sample by clustering using delexicalized parsing performance. While many other papers have presented and used subsets of UD treebanks for evaluation, they either do not focus on representativity (e.g. Ma et al. (2018)) or follow one of these methodologies (e.g. Anderson and Gómez-Rodríguez (2020a)).

3 Hyphothesis and methodology

As suggested above, parsing conclusions on multilingual environments are usually drawn from empirical research, which are prone to be parser-specific, experiment-specific, as well as treebank dependent.

Hypothesis We delve into this problem and hypothesize that parser comparisons based on running experiments and taking accuracy metrics on a given (reasonably-sized) subset of treebanks may lead to weak conclusions on rankings or differences in performance; as the magnitude and/or sign of the differences between parsers can change substantially depending on the choice of said subset.

3.1 Methodology

To test our hypothesis, we design two experiments:

Experiment 1: *few* controlled parsers, *few* pre-existing subsets In §4, we choose three con-

trasting parsers belonging to different parsing paradigms. Then, we train and evaluate them on a number of pre-defined (multilingual) subsets that were proposed in previous work (and later adopted by other authors as well). These existing subsets present different particularities, such as a high Indo-European bias (Ma et al., 2018), rich and diverse typologies (de Lhoneux et al., 2017b), or data scarcity issues (Dehouck et al., 2021), among others. Our aim is to see whether considering only a few robust parsers (treated as black boxes) and only a few already established subsets of treebanks, we can obtain different conclusions about their behaviors.

Experiment 2: *many* parsers, *many* randomized subsets In §5, we design a large-scale variant of the previous experiment. Assuming access to many parsers and treebanks, we ask: *could we obtain (reasonably-sized) subsets of treebanks that show very different behaviors?*, or to state it differently, *can parsing rankings be sensitive to the subset of treebanks where they are evaluated?* To do so, we use as a proxy the results from the CoNLLU Shared Task 2018 (Zeman et al., 2018), where 26 parsers participated and presented their experiments for 82 treebanks. We then create a random sample of 1 million subsets out of the $\sim 2.13 \times 10^{12}$ possible (multilingual) subsets of size 10. If a parser is cross-linguistically robust, then the variability of its position in the ranking should be small across all the studied subsets, while if their behavior is more unstable it could change dramatically, indicating that evaluating on a single subset of treebanks is not desirable.

4 Experiment 1: *few* controlled parsers, *few* pre-existing subsets

We take a few representative parsers (§4.1) and pre-defined subsets from the literature (§4.2) based, sometimes, on a careful treebank selection strategy.

4.1 The parsing models

We choose a graph-based (Dozat et al., 2017), a transition-based (Fernández-González and Gómez-Rodríguez, 2019), and a sequence labeling parser (Strzyz et al., 2019). We review them briefly, but we refer the reader to the papers for the details.

Bi-affine graph-based parser (`gb-DM17`; Dozat et al., 2017) It first computes contextualized vectors for each word using bidirectional LSTMs (biLSTMs; Hochreiter and Schmidhuber, 1997). After that, the model computes for each word a *head* and a *dependent* representation, which are sent through a bi-affine attention, determining for each token which is the most likely head. Here, we rely on the `supar`[1] package, which has been widely adopted by the community. We detail its hyperparameters in Appendix A (Table 7).

Left-to-right, transition-based, pointer network parser (`tb-FG19`; Fernández-González and Gómez-Rodríguez, 2019) It is a transition-based system, where at each time-step the pointer network predicts the index of the head for the focus token, and moves to the next one. The model uses an encoder-decoder architecture that in the first stage computes a hidden state representation for each token using biLSTMs. After that, the decoder predicts the tree left to right, computing a score attention between the current focus word and the encoder output sequence, excluding the own vector. We use the `syntacticpointer`[2] package. Appendix A (Table 8) details the hyperparameters.

Sequence labeling parser (`sl-S+19`; Strzyz et al., 2019) It outputs a dependency tree for each sentence of size n, using exactly n predictions and biLSTM tagging models. There are different ways to encode the trees (Spoustová and Spousta, 2010; Lacroix, 2019; Gómez-Rodríguez et al., 2020), but we will rely on the 2-planar bracketing encoding (Strzyz et al., 2020), which encodes 99.9% of non-projective trees and offers a robust behavior, including low-resource setups (Muñoz-Ortiz et al., 2021). We use the `dep2labels`[3] package. The hyperparameters are indicated in Appendix A (Table 9).

Parser comparability Sequence labeling parsers often underperform biaffine and pointer network parsers (Anderson and Gómez-Rodríguez, 2021), but we include them as a lower bound control parser. We kept fundamental architectural decisions of the parsers, e.g. how they compute character-level vectors or the strategies for cycle deletion, as it is not clear (or viable) that the same setup is optimal for all models. Also, we value to try these models as used by the community.

[1] https://github.com/yzhangcs/parser
[2] https://github.com/danifg/SyntacticPointer
[3] https://github.com/mstrise/dep2label

4.1.1 Experiment setup

The parsers are trained 3 times for each treebank, to then take the average as the final result.[4,5]

Input For all parsers, the embedding for each word is composed of a pre-trained word vector, a character-based vector and a PoS tag vector. For the word vectors, we use `fastText` (Bojanowski et al., 2017).[6] For PoS tags, we considered experiments both with gold and predicted PoS tags - using UDpipe (Straka et al., 2016)[7].

4.2 Datasets

Now, we review subsets that have been proposed, and summarize the criteria used to create them. While the subsets were defined on different versions of UD depending on the moment in which they were proposed, we use UD v2.8 for comparability. In case different treebanks are available for a given language and the authors did not specify which one they used for any reason (e.g. because in previous UD versions there was only one treebank, and therefore it was not necessary to name it), we chose the largest freely-available one. For space reasons, we include the specific treebanks for each subset in Appendix B (Table 10).

1. Ma et al. (2018) subset (Ma18): It has been widely adopted (Fernández-González and Gómez-Rodríguez, 2019; Li et al., 2020; Yang and Tu, 2021, *inter alia*), but it presents two weaknesses: (i) a high presence of Indo-European treebanks, ignoring diverse typologies, and (ii) as reported in their paper, all these treebanks are *easy*[8] treebanks.

2. de Lhoneux and Nivre (2016); de Lhoneux et al. (2017b) subset (Lh16): They were the first to address the problem of selecting a diverse sample of UD treebanks, establishing the following requirements: (i) include only one treebank from coarse-grained language families, (ii) include treebanks with certain morphological particularities, (iii) ensure different amounts of data, and (iv) include at least a highly non-projective treebank.

3. Schluter and Agić (2017) subset (SA17): Rather than manually choosing treebanks, this subset was chosen by an empirical method based on using delexicalized parsing performance to construct a similarity network, cluster it, and take one representative of each cluster. They concluded that their subset overestimates performance, while that of de Lhoneux and Nivre (2016) underestimates it.

4. Smith et al. (2018) subset (Sm18): The selection criteria for this subset were inspired in the criteria of de Lhoneux and Nivre (2016), but in this case aiming to be representative of different writing systems, character set sizes, and morphological complexity.

5. Kulmizev et al. (2019) subset (Ku19): The authors selected 13 treebanks, inspired in the criteria by de Lhoneux and Nivre (2016) and Smith et al. (2018). Apart from script, character set size and morphological complexity, they also aimed to have a representation of different training sizes and domains, and selected treebanks with good annotation quality.

6. Anderson and Gómez-Rodríguez (2020a) subset (AG20): Highly inspired by de Lhoneux et al. (2017b), but with a few changes. First, they exchanged $\text{Kazakh}_{\text{KTB}}$ for $\text{Uyghur}_{\text{UDT}}$, as Kazakh lacked an official development set. Second, they exchanged $\text{Ancient Greek}_{\text{PROIEL}}$ for $\text{Ancient Greek}_{\text{Perseus}}$, since it's more non-projective. Third, $\text{Czech}_{\text{PDT}}$ is swapped with $\text{Russian}_{\text{GSD}}$, as the Czech treebank took too long to train. Finally, they included $\text{Wolof}_{\text{WTB}}$ since African languages were not present. We included it to see if partial and justified changes over a diverse treebank subset could still lead to non-negligible changes.

7. Dehouck et al. (2021) subset (D21): This subset is dedicated to *true* data scarce treebanks. In the case of treebanks without a dev file, the

[4] Some treebanks ($\text{Kazakh}_{\text{KTB}}$, $\text{Galician}_{\text{TreeGal}}$ and $\text{Old-East-Slavic}_{\text{RNC}}$) do not have an official dev set, so we used 20% of the training set as the development set. Also, due to hardware limitations, the longest sentence (682 tokens) in the test file for the Old East Slavic (RNC) language was removed, since `syntacticpointer` ran out of memory during evaluation.

[5] The tools used for the experiments are described at this repository: https://github.com/MinionAttack/fragility_coling_2022

[6] We use `fastText` vectors except for some language treebanks that lacked embeddings. Particularly, for Ancient Greek we use UD embeddings, and for Wolof we used random initialized embeddings according to a uniform distribution in the range $[\frac{-1}{2\times300}, \frac{1}{2\times300}]$ (Goldberg, 2017).

[7] For Kazakh, Old East Slavic and Welsh there are no UDPipe models, so we only include their results with gold tags.

[8] We use *easy* in an informal sense, referring to treebanks where parsers obtain a higher performance. In *no way* we relate this term with a language being easier than other.

Set	LAS			$\mathcal{E}$-LAS			UAS			$\mathcal{E}$-UAS		
	gb-DM17	tb-FG19	sl-S+19	(tb-FG19, gb-DM17)	(sl-S+19, gb-DM17)	(sl-S+19, tb-FG19)	gb-DM17	tb-FG19	sl-S+19	(tb-FG19, gb-DM17)	(sl-S+19, gb-DM17)	(sl-S+19, tb-FG19)
Ma18	87.74	87.77	83.96	-0.14	23.93	23.93	91.07	91.13	87.68	-0.33	27.98	28.16
Lh16	80.33	79.68	74.20	1.83	24.04	22.49	85.03	84.45	79.90	2.21	26.33	24.51
SA17	84.85	84.97	80.30	-1.03	22.97	23.48	89.11	89.25	85.22	-1.46	26.76	27.64
Sm18	83.78	83.61	78.55	1.11	24.21	23.35	87.38	87.31	83.19	0.45	25.12	24.82
Ku19	83.36	83.08	77.98	-0.29	24.79	24.50	87.43	87.03	83.19	0.94	24.89	23.72
AG20	76.14	75.26	69.36	3.01	23.01	20.48	82.49	81.69	76.83	3.83	25.04	21.95
D21	59.00	57.04	51.38	4.57	17.00	12.97	68.60	67.38	62.96	3.94	16.18	12.92
Easy	89.59	89.65	85.88	-0.63	25.90	26.42	92.42	92.55	89.33	-1.58	28.61	29.85

Table 1: Average LAS and UAS scores for each subset in the predicted PoS tags setup. $\mathcal{E}(M1, M2)$ stands for error reduction between two models, where M1 is the reference system.

Set	LAS			$\mathcal{E}$-LAS			UAS			$\mathcal{E}$-UAS		
	gb-DM17	tb-FG19	sl-S+19	(tb-FG19, gb-DM17)	(sl-S+19, gb-DM17)	(sl-S+19, tb-FG19)	gb-DM17	tb-FG19	sl-S+19	(tb-FG19, gb-DM17)	(sl-S+19, gb-DM17)	(sl-S+19, tb-FG19)
Ma18	90.51	90.06	88.29	4.62	19.29	15.45	93.10	92.77	91.00	4.70	23.59	19.89
Lh16	78.89	76.71	74.44	7.20	19.94	13.55	84.30	83.13	80.84	6.10	21.24	16.13
SA17	85.52	84.57	81.92	5.71	20.79	15.89	89.52	88.83	86.65	5.32	23.40	19.00
Sm18	87.42	86.74	83.01	4.89	25.63	21.86	89.89	89.36	86.07	4.82	26.93	23.32
Ku19	87.18	86.14	82.99	6.22	24.65	19.58	89.82	89.04	86.18	5.87	26.13	21.52
AG20	81.04	79.54	77.53	7.17	14.08	7.35	85.77	84.82	82.72	6.46	16.26	10.50
D21	67.99	63.74	67.30	11.71	4.14	-8.82	75.26	72.61	75.52	9.92	2.14	-8.73
Easy	92.62	92.10	89.50	6.71	29.81	24.84	94.75	94.41	92.19	6.13	32.76	28.41

Table 2: Average LAS and UAS scores for each subset in the gold PoS tags setup.

training file was split in two, with a ratio of 80-20 for the training file and the dev file.

8. Easy subset: We propose an explicit *easy* subset to compare against other easy ones (e.g. Ma18). We used the results from the CoNLL 2018 Shared Task[9], and chose the 10 treebanks with the best LAS (no repeated languages). We list them in Appendix B.

4.3 Results

Table 1 shows the macro-average LAS (Labeled Attachment Score) and UAS (Unlabeled Attachment Score) results, using predicted PoS tags, for each subset, i.e. *the subset*, and not the treebank, is considered as *the atomic unit* for evaluation. For informative purposes, Table 2 shows the equivalent evaluation with gold PoS tags, but we will focus on the results with predicted PoS tags, unless stated otherwise. We also show error reduction ratios on LAS and UAS between parsers. This metric provides a better picture of differences between parsers than absolute LAS/UAS differences would, as it is less affected by treebank difficulty differences (e.g., it is much harder to achieve a given absolute LAS and UAS difference on easy treebanks than on more difficult ones, due to less available room for improvement and diminishing returns). The error reduction shown for each subset is calculated by first computing the error reduction for each treebank in the subset, and then averaging these error reductions (rather than by averaging the LAS/UAS for each treebank in the subset, and computing a single error reduction on that average). While this choice can cause some superficially counterintuitive phenomena like a parser having more average LAS than another but negative LAS error reduction (this happens with `tb-FG19` and `gb-DM17` on Ku19 on Table 1), it provides the desired semantics: for example, if a parser improves LAS from 98% to 99% in one treebank and from 50% to 90% in another, on average it is removing 65% of errors (50% of the errors in the first corpus, 80% in the second) and not 78.8% which we would obtain if we computed error reduction on average LAS.

Next, we discuss factors that seem to play a role in the subset performance.

Influence of parsing difficulty From the results, easier subsets tend to correspond to larger error reductions when comparing the (state-of-the-art) parsers `gb-DM17` and `tb-FG19` with respect to `sl-S+19` (the control parser). This is most evident for the Easy subset: all parsers obtain their best performance across all subsets, and the error reductions with respect to the control parser are also the largest, for all setups. The opposite happens with the D21 subset, the hardest one. In this context, when optimizing for other dimensions than performance, such as speed, training efficiency or architectural simplicity, relying (exclusively) on easy treebanks could thus be a sub-optimal strategy. The sense of the decrease in performance could

[9] https://universaldependencies.org/conll18/results-las.html

be larger on these easy datasets than when evaluating on random treebanks, or on more difficult cases as suggested by the results on the subsets of Lh16, D21, or AG20 to a lesser extent. On the contrary, dimensions such as the ones mentioned above are often not expected to benefit more from the particularities of easy treebanks. Also, there are trends related to parsing difficulty between the state-of-the-art parsers `gb-DM17` and `tb-FG19`: `gb-DM17` seems to be superior to `tb-FG19` when the subset becomes harder to parse, and *vice versa*.

Differences on representative subsets While both the Lh16 and the SA17 subsets were designed to enforce representativity, the ranking of the tested parsers changes: `tb-FG19` performs better (in LAS error reduction terms) than `gb-DM17` on SA17 (automatically picked) and Ku19 (manually constructed), but worse on the (manually constructed) subsets of Lh16, Sm18 and AG20. This highlights that even when treebanks are sampled with attention to representativity, results can still show instability - be it due to different possible notions of representativity, or statistical variation.

Developing and testing on the same treebanks While there is no clear performance difference between `gb-DM17` and `tb-FG19`, as each of them surpasses the other in some subsets; one of the subsets where `tb-FG19` takes the lead is Ma18, where that parser was developed and reported its results. This leads to the question whether developing and evaluating on a given subset of treebanks could induce bias in favor of those treebanks. While the available data is not enough to give an answer in this specific instance, we can draw similar conclusions either way. If this were the case, it would mean that in the context of multilingual, language-agnostic parsers, and when data for a wide range of languages is available, it would be advisable to go beyond separating development and test sets for each language or treebank, and instead use different languages for development than for evaluation to avoid this kind of bias. Conversely, if this were not the case, it would mean that we could choose one of the human-defined subsets and obtain state-of-the-art results for one parser or the other, purely by chance. This makes us reflect about using a single subset of treebanks to justify the superior performance of a model, and might again make advisable to develop and test on different subsets - to reduce the element of chance.

Experimentation with data scarcity For the D21 subset, centered exclusively on extremely low-resource treebanks, the error reduction computed between the best performing parser (`gb-DM17`) and the control parser (`sl-S+19`) is the lowest among all tested treebanks. As mentioned above, the opposite happens for the easiest subsets. Yet, we feel these type of subsets would not be optimal either for evaluating parsers in a general sense, as they might not capture how a given parser can fully exploit its learning capabilities. Overall, the evaluation on this setup seems more volatile. We see a few differences between the predicted and gold PoS tags setups, causing even changes in the parsing ranking. For instance, `sl-S+19` outperforms `tb-FG19` in the gold setup by a clear margin, an issue that does not arise in any other subset.

5 Experiment 2: *many* parsers, *many* randomized subsets

This experiment can be seen as a re-definition of Experiment 1 at a large scale. Above, we compared a few competitive parsers only on a handful of subsets of treebanks that were human-defined, and observed different trends. Yet, this is a limited view of the problem. If we take as reference the CoNLLU 2018 Shared Task (Zeman et al., 2018) and the 82 treebanks that were evaluated, considering subsets of size 10 (meaning that each is composed of 10 different treebanks), we would obtain up to $\sim 2.13 \times 10^{12}$ possible combinations. Many of those subsets will not be a representative sample of languages, but we already saw that there are subsets that are used in parsing as a benchmark that are not either, and that even when they are considered representative, the criteria varies, and the parsing performance, differences among parsers and error reductions vary too. Here, we generate subsets, similar in size to typical human-defined ones, and see how subset differences affect parsing rankings.

Similarly to Experiment 1, we do not analyze here algorithms and parsing architectures, or their correctness, but the appropriateness of evaluation procedures. In this context, some shared-task systems have reported bugs in their pipeline: this was mostly evident for some systems that consistently ranked in the last positions (see Table 3), but not so noticeable for high-scoring ones, such as the Stanford system (Qi et al., 2018), which later reported a preprocessing bug that affected the low-resource treebanks more. Thus, multi-treebank evaluation

Parser	Best rank	Worst rank	μ	$\tilde{\mu}$	σ
HIT-SCIR	1	6	1.14	1.00	0.54
UDPipe Future	1	12	4.38	4.00	1.65
TurkuNLP	1	12	3.97	4.00	1.53
LATTICE	1	13	4.99	5.00	2.27
ICS PAS	2	13	4.71	5.00	2.01
CEA LIST	1	13	6.12	6.00	2.28
Stanford	1	21	6.30	6.00	3.47
Uppsala	1	13	6.62	7.00	2.36
NLP-Cube	2	20	10.02	10.00	1.79
AntNLP	3	18	9.94	10.00	1.82
ParisNLP	4	20	10.39	11.00	1.62
SLT-Interactions	2	23	11.28	11.00	3.91
IBM NY	2	20	13.05	13.00	1.64
LeisureX	8	20	14.36	14.00	1.81
UniMelb	8	19	13.80	14.00	1.13
KParse	9	22	16.78	17.00	1.35
Fudan	10	22	17.16	17.00	1.60
BASELINE UDPipe	13	22	18.08	18.00	1.11
Phoenix	13	22	18.69	19.00	1.07
CUNI x-ling	2	22	19.24	20.00	2.21
BOUN	16	23	20.81	21.00	0.79
ONLP lab	20	25	22.57	23.00	0.62
iParse	9	25	22.36	23.00	2.76
HUJI	21	25	23.63	24.00	0.89
ArmParser	22	25	24.62	25.00	0.59
SParse	26	26	26.00	26.00	0.00

Table 3: Ranking stats for LAS and the parsers of the Zeman et al. (2018) Shared task, over the 1 million random subsets. Table sorted by $\tilde{\mu}$ (the median).

Parser	Best rank	Worst rank	μ	$\tilde{\mu}$	σ
TurkuNLP	1	7	1.51	1.00	0.78
HIT-SCIR	1	6	2.37	2.00	1.13
ICS PAS	1	13	3.83	4.00	1.37
UDPipe Future	1	8	3.62	4.00	0.98
Stanford	1	15	4.21	4.00	1.75
LATTICE	1	11	6.23	6.00	1.02
CEA LIST	2	12	6.58	6.00	0.99
ParisNLP	5	16	8.87	9.00	0.91
AntNLP	3	14	8.59	9.00	0.95
SLT-Interactions	2	20	10.09	10.00	2.25
LeisureX	7	18	11.54	11.00	1.21
UniMelb	7	16	11.27	11.00	0.82
BASELINE UDPipe	10	20	14.53	14.00	1.09
NLP-Cube	6	22	15.07	14.00	2.81
Phoenix	10	20	14.89	15.00	1.15
KParse	9	21	15.54	16.00	1.95
CUNI x-ling	4	21	17.43	18.00	1.60
BOUN	12	22	18.22	18.00	1.33
Fudan	9	22	17.34	18.00	1.94
iParse	9	25	19.12	20.00	3.19
HUJI	15	25	20.45	21.00	1.02
ArmParser	19	25	22.10	22.00	0.95
Uppsala	19	25	23.29	23.00	0.66
IBM NY	16	25	23.44	24.00	0.82
ONLP lab	22	25	24.88	25.00	0.37
SParse	26	26	26.00	26.00	0.00

Table 4: Ranking statistics for BLEX and the parsers of the Zeman et al. (2018) Shared task, over 1 million randomly generated subsets. Table sorted by $\tilde{\mu}$.

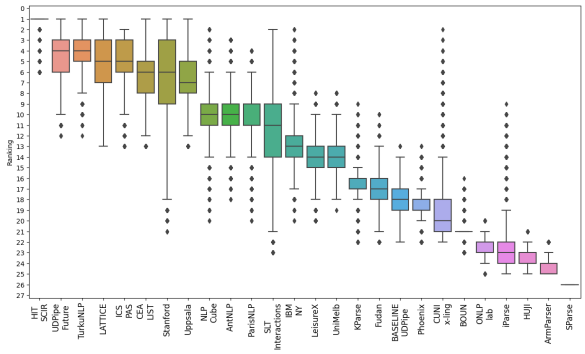

Figure 1: Corresponding box plot for Table 3. For an easy correspondence with the table, the x-axis (from left to right) is sorted as the Table is (i.e. by $\tilde{\mu}$).

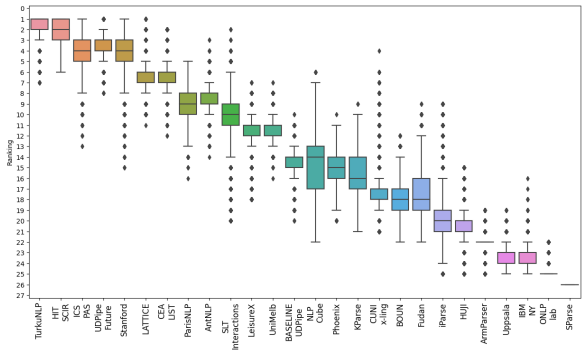

Figure 2: Corresponding box plot for Table 4. For an easy correspondence with the table, the x-axis (from left to right) is sorted as the table is (i.e. by $\tilde{\mu}$).

procedures should also be robust for systems suffering bugs that affect treebanks differently.

5.1 Experimental setup

We compare the available results of the 26 parsers[10] that participated in the CoNLLU Shared Task (Zeman et al., 2018), sampling random subsets over the 82 evaluated treebanks. We generate 1 million random subsets made of size 10,[11] and we do not control the subsets' content (e.g. subsets with higher presence of a language or family).

[10]The parsers are not necessarily a diverse sample of parsing models (many are based on `gb-DM17`) but they are a realistic sample of a ranking of parsers made in a real shared task. For model representativity, we refer the reader to Experiment 1.

[11]The subset size is in the range of those of Experiment 1.

5.2 Results

The pair Table 3 - Figure 1 shows statistics about the 26 parsers that participated in the shared task and the 1 million randomly generated subsets. Some (top) parsers show a stable performance. For instance, the HIT-SCIR parser (Che et al., 2018) mostly ranks at the first position, except for a few outliers that show that the parser potentially could go down as far as the 6th position. This tendency is also observed in a few other - and worse-performing - systems, such as Kparse (Kırnap et al., 2018) or Phoenix (Wu et al., 2018).

However, for many other parsers the variability is larger. The interquartile range of the UDpipe-Future system (2nd place) (Straka, 2018) is small (from 3rd to 6th), but its fourth quartile (excluding outliers) ranges between the 6th and the 10th posi-

Parser	Avg. rank	Best subset outliers Rank	Best subset outliers Treebanks	Worst subset outliers Rank	Worst subset outliers Treebanks
UDPipe Future (Straka, 2018)	$4.38_{\pm 1.65}$	1	pl_{sz}, ga_{idt}, fro_{srcmf}, sr_{set}, kmr_{mg}, el_{gdt}, en_{ewt}, fr_{gsd}, th_{pud}, hy_{armtdp}	12	bxr_{bdt}, la_{ittb}, hsb_{ufal}, pcm_{nsc}, th_{pud}, ro_{rrt}, br_{keb}, $no_{bokmaal}$, sl_{ssj}, sv_{pud}
Stanford (Qi et al., 2018)	$6.30_{\pm 3.47}$	1	de_{gsd}, cs_{cac}, el_{gdt}, he_{htb}, es_{ancora}, sv_{lines}, ja_{modern}, bg_{btb}, $no_{bokmaal}$, cs_{pud}	21	hsb_{ufal}, pt_{bosque}, ga_{idt}, vi_{vtb}, ko_{kaist}, $sv_{talbanken}$, $gl_{treegal}$, sme_{giella}, hy_{armtdp}, th_{pud}
SLT-Interactions (Bhat et al., 2018)	$11.28_{\pm 3.91}$	2	sk_{snk}, hsb_{ufal}, uk_{iu}, $cs_{fictree}$, sv_{pud}, ja_{gsd}, pt_{bosque}, hr_{set}, fa_{seraji}, sl_{ssj}	23	hr_{set}, fa_{seraji}, pl_{lfg}, kmr_{mg}, de_{gsd}, hy_{armtdp}, nl_{alpino}, th_{pud}, sme_{giella}, fro_{srcmf}

Table 5: Qualitative results of subsets that cause anomalous rankings of parsers in Experiment 2.

tion. The situation is almost identical for the next 4 averaged best performing systems. Across the board, there are even more severe examples, such as Stanford (Qi et al., 2018) (7th place), whose interquartile range spans from the 3rd to the 9th position; or the SLT-Interactions parser (Bhat et al., 2018) whose first quartile ranges from 2nd to 9th, while its fourth quartile ranges from 14th to 21th. Exemplifying it with the Stanford system, we will also discuss below how randomized multi-treebank evaluation would have been useful to detect the anomalous performance on low-resource treebanks, that later on turned out to be a bug, or on the other hand how a weak subset selection could cause potential anomalous performances or bugs to go unnoticed.

Parser	#Outliers	Avg. size	$\mathcal{R}$(lr)	$\mathcal{R}$(Slavic)
UDPipe Future	10 best	168.87	0.30	0.17
	10 worst	157.51	0.39	0.21
Stanford	10 best	221.03	0.19	0.23
	10 worst	124.44	0.61	0.18
SLT-Interactions	10 best	203.92	0.30	0.36
	10 worst	146.72	0.39	0.14

Table 6: Quantitative study expanding Table 5. $\mathcal{R}$ refers to the average ratio across subsets of the presence of low-resource (lr) and Slavic treebanks.

Subsets that cause anomalous results Table 5 shows a few examples of parsers and subsets that caused atypical results. For each parser, we show an advantageous and a disadvantageous subset, randomly picked among those for which a parser obtained its best and worst rankings.

A qualitative analysis of these results yields several insights. For the first two parsers, the advantageous and disadvantageous outliers are linguistically diverse, but there is a clear trend that the disadvantageous subsets are heavily biased towards small treebanks: for both of the parsers, the favorable subset contains only 2 treebanks that are low-resource according to the shared task criteria, while the disfavorable one contains 6 and 5, respectively. This is very unlikely to happen by chance: the probability of randomly drawing a subset with 6 or more low-resource treebanks is 0.00214, and with 5 or more, 0.01538 (this is calculated from a hypergeometric distribution with parameters $N = 82$, the total number of treebanks, $K = 21$, the number of low-resource treebanks, and $n = 10$, the number of treebanks per subset). Thus, this variability seems to owe to the fact that the UDPipe Future (Straka, 2018) and Stanford (Qi et al., 2018) parsers struggle (relatively to competitors) when training data is scarce. The situation is different for the third parser considered. In this case, there are no substantial differences in treebank size (2 vs. 3 low-resource treebanks) but instead there is a clear linguistic pattern: the advantageous subset has a heavy bias towards Slavic languages (6 out of the 10 languages are Slavic, compared to 2 in the disadvantageous subset - and the probability of choosing a subset with 6 or more Slavic languages by chance is 0.00043, from a hypergeometric distribution with parameters $N = 82, K = 17, n = 10$). This seems to reflect that the SLT-Interactions parser (Bhat et al., 2018) is especially adequate for Slavic languages. It is worth noting that the authors did not implement any language-specific adaptation or report anything in the paper that suggests that they specifically addressed these languages, so this serves as an example that a parser can show linguistic biases towards certain language families even if it has been developed in a language-agnostic way.

We propose a complementary quantitative analysis in Table 6. We randomly take 10 of the best and worst performing subsets for the above studied parsers and compute, across subsets, the average size of treebanks, the presence of low-resource treebanks, and the presence of Slavic languages. The analysis confirms the bias towards rich-resource treebanks for the Stanford parser, and towards Slavic languages for SLT-Interactions (while not being biased towards rich- or low-resouce treebanks). On the other hand, the hypothesis of UDpipe Future being biased towards rich-resource languages is not clearly confirmed by this analysis.

To sum up, this reinforces the idea (hypothesized in papers like de Lhoneux and Nivre (2016)) that

both treebank sizes and linguistic factors are important for a treebank subset to be representative; and highlight that the latter can have a huge influence even in parsers that have been developed without specific language families in mind.

More robust metrics? LAS and UAS are the most popular metrics to report dependency parsing performance. Yet, there are other metrics, such as CLAS[12], MLAS[13] or BLEX[14], but they have not been widely adopted (maybe because they have a not so straightforward interpretation). Yet, from Experiment 2 we observed that some of these metrics, especially BLEX, produced narrower standard deviations and more stable rankings. We leave interpretations of this phenomenon as an open question for future work, but refer the reader to (Table 4, Figure 2) and (Table 3, Figure 1), which show a summary of the ranking statistics for the BLEX and LAS metrics, respectively, on the 26 parsers that participated in the ConLLU Shared task 2018 (Zeman et al., 2018). Overall, but especially for the top parsers, BLEX results produce more stable rankings and narrower interquartile ranges.

6 Discussion

We have designed two experiments that revealed issues of relying on a single subset of treebanks for parsing evaluation. More particularly, we have shown that: (i) existing human-defined subsets show high variability in terms of rankings and performance across parsers, (ii) parsers that have been developed on a concrete subset might be biased towards performing better on that subset, (iii) it is relatively easy to come up with subsets that generate different parsing rankings, (iv) this can even happen across subsets that have been purposefully defined to be representative, (v) both linguistic typology and resource size have a large influence in the variability of results between parsers, and (vi) linguistic factors can be crucial even when parsers are designed in a language-agnostic way.

Overall, some advice can be given: (a) claims that "parser X is more accurate than parser Y" can be weak even on carefully selected samples of UD treebanks (and perhaps it is recommendable to consider metrics that take into account dimensions such as speed and efficiency), (b) for language-agnostic parsers, it is worth noting that there can still be biases towards certain linguistic families, and (c) for such parsers, it can be advisable to develop on one set of treebanks and evaluate on another, to avoid bias in favor of the languages used for development.

Finally, there are aspects that we did not study in this piece of work, but that could affect the robustness of parsing evaluation as well, e.g., automatically *versus* manually annotated treebanks, and interactions between language and treebank properties (e.g. morphological complexity, dependency distance, ...) and parsing models.

7 Conclusion

Different subsets of treebanks have been proposed to try to capture the essence of the whole set of UD treebanks, so that the performance of parsers in such subsets would be representative of that obtained in the full set. We have empirically shown limitations of this approach, and also how establishing guidelines for good treebank selection can be hard, although some bad practices can be avoided.

Acknowledgments

This work was supported by a 2020 Leonardo Grant for Researchers and Cultural Creators from the FBBVA,[15] as well as by the European Research Council (ERC), under the European Union's Horizon 2020 research and innovation programme (FASTPARSE, grant agreement No 714150). The work is also supported by ERDF/MICINN-AEI (SCANNER-UDC, PID2020-113230RB-C21), by Xunta de Galicia (ED431C 2020/11), and by Centro de Investigación de Galicia "CITIC" which is funded by Xunta de Galicia, Spain and the European Union (ERDF - Galicia 2014–2020 Program), by grant ED431G 2019/01.

References

Mark Anderson and Carlos Gómez-Rodríguez. 2020a. Distilling neural networks for greener and faster dependency parsing. In *Proceedings of the 16th International Conference on Parsing Technologies and the IWPT 2020 Shared Task on Parsing into Enhanced Universal Dependencies*, pages 2–13, Online. Association for Computational Linguistics.

[12] CLAS: It ignores selected relations which attach function words to content words.

[13] MLAS: It is inspired by the CLAS metric, and extended with evaluation of POS tags and morphological features.

[14] BLEX (bi-lexical dependency score): it combines content-word relations with lemmatization.

[15] FBBVA accepts no responsibility for the opinions, statements and contents included in the project and/or the results thereof, which are entirely the responsibility of the authors.

Mark Anderson and Carlos Gómez-Rodríguez. 2020b. Inherent dependency displacement bias of transition-based algorithms. In *Proceedings of the 12th Language Resources and Evaluation Conference*, pages 5147–5155, Marseille, France. European Language Resources Association.

Mark Anderson and Carlos Gómez-Rodríguez. 2021. A modest Pareto optimisation analysis of dependency parsers in 2021. In *Proceedings of the 17th International Conference on Parsing Technologies and the IWPT 2021 Shared Task on Parsing into Enhanced Universal Dependencies (IWPT 2021)*, pages 119–130, Online. Association for Computational Linguistics.

Mark Anderson, Anders Søgaard, and Carlos Gómez-Rodríguez. 2021. Replicating and extending "Because their treebanks leak": Graph isomorphism, covariants, and parser performance. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 2: Short Papers)*, pages 1090–1098, Online. Association for Computational Linguistics.

Riyaz A. Bhat, Irshad Bhat, and Srinivas Bangalore. 2018. The SLT-interactions parsing system at the CoNLL 2018 shared task. In *Proceedings of the CoNLL 2018 Shared Task: Multilingual Parsing from Raw Text to Universal Dependencies*, pages 153–159, Brussels, Belgium. Association for Computational Linguistics.

Piotr Bojanowski, Edouard Grave, Armand Joulin, and Tomas Mikolov. 2017. Enriching word vectors with subword information. *Transactions of the Association for Computational Linguistics*, 5:135–146.

Wanxiang Che, Yijia Liu, Yuxuan Wang, Bo Zheng, and Ting Liu. 2018. Towards better UD parsing: Deep contextualized word embeddings, ensemble, and treebank concatenation. In *Proceedings of the CoNLL 2018 Shared Task: Multilingual Parsing from Raw Text to Universal Dependencies*, pages 55–64, Brussels, Belgium. Association for Computational Linguistics.

Raphael Cohen, Yoav Goldberg, and Michael Elhadad. 2012. Domain adaptation of a dependency parser with a class-class selectional preference model. In *Proceedings of ACL 2012 Student Research Workshop*, pages 43–48, Jeju Island, Korea. Association for Computational Linguistics.

Anna Corazza, Alberto Lavelli, and Giorgio Satta. 2013. An information-theoretic measure to evaluate parsing difficulty across treebanks. *ACM Trans. Speech Lang. Process.*, 9(4).

Miryam de Lhoneux and Joakim Nivre. 2016. Ud treebank sampling for comparative parser evaluation. In *Proc. of the Sixth Swedish Language Technology Conference (SLTC)*.

Miryam de Lhoneux, Sara Stymne, and Joakim Nivre. 2017a. Old school vs. new school: Comparing transition-based parsers with and without neural network enhancement. In *TLT*, pages 99–110.

Miryam de Lhoneux, Sara Stymne, and Joakim Nivre. 2017b. Old school vs. new school: Comparing transition-based parsers with and without neural network enhancement. In *TLT*.

Mathieu Dehouck, Mark Anderson, and Carlos Gómez-Rodríguez. 2021. A falta de pan, buenas son tortas: The efficacy of predicted upos tags for low resource ud parsing. *arXiv preprint arXiv:2106.04222*.

Mathieu Dehouck and Pascal Denis. 2019. Phylogenic multi-lingual dependency parsing. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 192–203, Minneapolis, Minnesota. Association for Computational Linguistics.

Timothy Dozat, Peng Qi, and Christopher D. Manning. 2017. Stanford's graph-based neural dependency parser at the CoNLL 2017 shared task. In *Proceedings of the CoNLL 2017 Shared Task: Multilingual Parsing from Raw Text to Universal Dependencies*, pages 20–30, Vancouver, Canada. Association for Computational Linguistics.

Mark Dredze, John Blitzer, Partha Pratim Talukdar, Kuzman Ganchev, João Graça, and Fernando Pereira. 2007. Frustratingly hard domain adaptation for dependency parsing. In *Proceedings of the 2007 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning (EMNLP-CoNLL)*, pages 1051–1055, Prague, Czech Republic. Association for Computational Linguistics.

Daniel Fernández-González and Carlos Gómez-Rodríguez. 2019. Left-to-right dependency parsing with pointer networks. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 710–716, Minneapolis, Minnesota. Association for Computational Linguistics.

Yoav Goldberg. 2017. Neural network methods for natural language processing. *Synthesis lectures on human language technologies*, 10(1):1–309.

Carlos Gómez-Rodríguez, Michalina Strzyz, and David Vilares. 2020. A unifying theory of transition-based and sequence labeling parsing. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 3776–3793, Barcelona, Spain (Online). International Committee on Computational Linguistics.

Kyle Gorman and Steven Bedrick. 2019. We need to talk about standard splits. In *Proceedings of the 57th Annual Meeting of the Association for Computational*

Linguistics, pages 2786–2791, Florence, Italy. Association for Computational Linguistics.

Kristina Gulordava and Paola Merlo. 2016. Multilingual dependency parsing evaluation: a large-scale analysis of word order properties using artificial data. *Transactions of the Association for Computational Linguistics*, 4:343–356.

Daniel Hershcovich, Stella Frank, Heather Lent, Miryam de Lhoneux, Mostafa Abdou, Stephanie Brandl, Emanuele Bugliarello, Laura Cabello Piqueras, Ilias Chalkidis, Ruixiang Cui, Constanza Fierro, Katerina Margatina, Phillip Rust, and Anders Søgaard. 2022. Challenges and strategies in cross-cultural NLP. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 6997–7013, Dublin, Ireland. Association for Computational Linguistics.

Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long short-term memory. *Neural computation*, 9(8):1735–1780.

Ömer Kırnap, Erenay Dayanık, and Deniz Yuret. 2018. Tree-stack LSTM in transition based dependency parsing. In *Proceedings of the CoNLL 2018 Shared Task: Multilingual Parsing from Raw Text to Universal Dependencies*, pages 124–132, Brussels, Belgium. Association for Computational Linguistics.

Sandra Kübler, Wolfgang Maier, Ines Rehbein, and Yannick Versley. 2008. How to compare treebanks. In *Proceedings of the Sixth International Conference on Language Resources and Evaluation (LREC'08)*, Marrakech, Morocco. European Language Resources Association (ELRA).

Marco Kuhlmann and Joakim Nivre. 2010. Transition-based techniques for non-projective dependency parsing. *Northern European Journal of Language Technology*, 2(1):1–19.

Artur Kulmizev, Miryam de Lhoneux, Johannes Gontrum, Elena Fano, and Joakim Nivre. 2019. Deep contextualized word embeddings in transition-based and graph-based dependency parsing - a tale of two parsers revisited. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 2755–2768, Hong Kong, China. Association for Computational Linguistics.

Ophélie Lacroix. 2019. Dependency parsing as sequence labeling with head-based encoding and multi-task learning. In *Proceedings of the Fifth International Conference on Dependency Linguistics (Depling, SyntaxFest 2019)*, pages 136–143, Paris, France. Association for Computational Linguistics.

Zuchao Li, Hai Zhao, and Kevin Parnow. 2020. Global greedy dependency parsing. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pages 8319–8326.

Xuezhe Ma, Zecong Hu, Jingzhou Liu, Nanyun Peng, Graham Neubig, and Eduard Hovy. 2018. Stack-pointer networks for dependency parsing. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1403–1414, Melbourne, Australia. Association for Computational Linguistics.

Ryan McDonald and Joakim Nivre. 2007. Characterizing the errors of data-driven dependency parsing models. In *Proceedings of the 2007 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning (EMNLP-CoNLL)*, pages 122–131, Prague, Czech Republic. Association for Computational Linguistics.

Ryan McDonald and Joakim Nivre. 2011. Analyzing and integrating dependency parsers. *Computational Linguistics*, 37(1):197–230.

Alberto Muñoz-Ortiz, Michalina Strzyz, and David Vilares. 2021. Not all linearizations are equally data-hungry in sequence labeling parsing. In *Proceedings of the International Conference on Recent Advances in Natural Language Processing (RANLP 2021)*, pages 978–988, Held Online. INCOMA Ltd.

Joakim Nivre, Johan Hall, Sandra Kübler, Ryan McDonald, Jens Nilsson, Sebastian Riedel, and Deniz Yuret. 2007. The CoNLL 2007 shared task on dependency parsing. In *Proceedings of the 2007 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning (EMNLP-CoNLL)*, pages 915–932, Prague, Czech Republic. Association for Computational Linguistics.

Peng Qi, Timothy Dozat, Yuhao Zhang, and Christopher D. Manning. 2018. Universal Dependency parsing from scratch. In *Proceedings of the CoNLL 2018 Shared Task: Multilingual Parsing from Raw Text to Universal Dependencies*, pages 160–170, Brussels, Belgium. Association for Computational Linguistics.

Ines Rehbein, Julius Steen, Bich-Ngoc Do, and Anette Frank. 2017. Universal Dependencies are hard to parse – or are they? In *Proceedings of the Fourth International Conference on Dependency Linguistics (Depling 2017)*, pages 218–228, Pisa,Italy. Linköping University Electronic Press.

Rudolf Rosa. 2015. Multi-source cross-lingual delexicalized parser transfer: Prague or Stanford? In *Proceedings of the Third International Conference on Dependency Linguistics (Depling 2015)*, pages 281–290, Uppsala, Sweden. Uppsala University, Uppsala, Sweden.

Natalie Schluter and Željko Agić. 2017. Empirically sampling Universal Dependencies. In *Proceedings of the NoDaLiDa 2017 Workshop on Universal Dependencies (UDW 2017)*, pages 117–122, Gothenburg, Sweden. Association for Computational Linguistics.

Aaron Smith, Miryam de Lhoneux, Sara Stymne, and Joakim Nivre. 2018. An investigation of the interactions between pre-trained word embeddings, character models and POS tags in dependency parsing. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 2711–2720, Brussels, Belgium. Association for Computational Linguistics.

Anders Søgaard. 2020. Some languages seem easier to parse because their treebanks leak. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 2765–2770, Online. Association for Computational Linguistics.

Anders Søgaard, Sebastian Ebert, Jasmijn Bastings, and Katja Filippova. 2021. We need to talk about random splits. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*, pages 1823–1832, Online. Association for Computational Linguistics.

Drahomíra Spoustová and Miroslav Spousta. 2010. Dependency parsing as a sequence labeling task. *The Prague Bulletin of Mathematical Linguistics*, 94:7.

Milan Straka. 2018. UDPipe 2.0 prototype at CoNLL 2018 UD shared task. In *Proceedings of the CoNLL 2018 Shared Task: Multilingual Parsing from Raw Text to Universal Dependencies*, pages 197–207, Brussels, Belgium. Association for Computational Linguistics.

Milan Straka, Jan Hajič, and Jana Straková. 2016. UDPipe: Trainable pipeline for processing CoNLL-U files performing tokenization, morphological analysis, POS tagging and parsing. In *Proceedings of the Tenth International Conference on Language Resources and Evaluation (LREC'16)*, pages 4290–4297, Portorož, Slovenia. European Language Resources Association (ELRA).

Michalina Strzyz, David Vilares, and Carlos Gómez-Rodríguez. 2019. Viable dependency parsing as sequence labeling. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 717–723, Minneapolis, Minnesota. Association for Computational Linguistics.

Michalina Strzyz, David Vilares, and Carlos Gómez-Rodríguez. 2020. Bracketing encodings for 2-planar dependency parsing. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 2472–2484, Barcelona, Spain (Online). International Committee on Computational Linguistics.

Reut Tsarfaty, Djamé Seddah, Sandra Kübler, and Joakim Nivre. 2013. Parsing morphologically rich languages: Introduction to the special issue. *Computational Linguistics*, 39(1):15–22.

Clara Vania, Yova Kementchedjhieva, Anders Søgaard, and Adam Lopez. 2019. A systematic comparison of methods for low-resource dependency parsing on genuinely low-resource languages. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 1105–1116, Hong Kong, China. Association for Computational Linguistics.

Guillaume Wisniewski and François Yvon. 2019. How Bad are PoS Tagger in Cross-Corpora Settings? Evaluating Annotation Divergence in the UD Project. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 218–227, Minneapolis, Minnesota. Association for Computational Linguistics.

Yingting Wu, Hai Zhao, and Jia-Jun Tong. 2018. Multilingual Universal Dependency parsing from raw text with low-resource language enhancement. In *Proceedings of the CoNLL 2018 Shared Task: Multilingual Parsing from Raw Text to Universal Dependencies*, pages 74–80, Brussels, Belgium. Association for Computational Linguistics.

Songlin Yang and Kewei Tu. 2021. Headed span-based projective dependency parsing. *arXiv preprint arXiv:2108.04750*.

Daniel Zeman, Jan Hajič, Martin Popel, Martin Potthast, Milan Straka, Filip Ginter, Joakim Nivre, and Slav Petrov. 2018. CoNLL 2018 shared task: Multilingual parsing from raw text to Universal Dependencies. In *Proceedings of the CoNLL 2018 Shared Task: Multilingual Parsing from Raw Text to Universal Dependencies*, pages 1–21, Brussels, Belgium. Association for Computational Linguistics.

Daniel Zeman, Joakim Nivre, et al. 2020. Universal dependencies 2.7. LINDAT/CLARIAH-CZ digital library at the Institute of Formal and Applied Linguistics (ÚFAL), Faculty of Mathematics and Physics, Charles University.

Daniel Zeman, Martin Popel, Milan Straka, Jan Hajič, Joakim Nivre, Filip Ginter, Juhani Luotolahti, Sampo Pyysalo, Slav Petrov, Martin Potthast, Francis Tyers, Elena Badmaeva, Memduh Gokirmak, Anna Nedoluzhko, Silvie Cinková, Jan Hajič jr., Jaroslava Hlaváčová, Václava Kettnerová, Zdeňka Urešová, Jenna Kanerva, Stina Ojala, Anna Missilä, Christopher D. Manning, Sebastian Schuster, Siva Reddy, Dima Taji, Nizar Habash, Herman Leung, Marie-Catherine de Marneffe, Manuela Sanguinetti, Maria Simi, Hiroshi Kanayama, Valeria de Paiva, Kira Droganova, Héctor Martínez Alonso, Çağrı Çöltekin, Umut Sulubacak, Hans Uszkoreit, Vivien Macketanz, Aljoscha Burchardt, Kim Harris, Katrin Marheinecke, Georg Rehm, Tolga Kayadelen, Mohammed Attia, Ali Elkahky, Zhuoran Yu, Emily Pitler, Saran Lertpradit, Michael Mandl, Jesse Kirchner, Hector Fernandez Alcalde, Jana Strnadová, Esha Banerjee, Ruli Ma-

nurung, Antonio Stella, Atsuko Shimada, Sookyoung Kwak, Gustavo Mendonça, Tatiana Lando, Rattima Nitisaroj, and Josie Li. 2017. CoNLL 2017 shared task: Multilingual parsing from raw text to Universal Dependencies. In *Proceedings of the CoNLL 2017 Shared Task: Multilingual Parsing from Raw Text to Universal Dependencies*, pages 1–19, Vancouver, Canada. Association for Computational Linguistics.

A Experiment 1: models, resources, and hyperparameters

To train the models, we used 2 NVIDIA GeForce RTX 2080 Ti@11GB and an Intel® Core™ i7-9700K@3.60GHz×8. Training times usually took from 1 to 7 hours, depending on the parsing model and the treebank training size. The three used parsers and the UD treebanks have free software licenses that allow free use and distribution.

Tables 7, 8 and 9 show the hyperparameters used for the `gb-DM17` (Dozat et al., 2017) (using the `supar` software package), `tb-FG19` (Fernández-González and Gómez-Rodríguez, 2019) (using the `syntacticpointer` package) and `sl-S+19` parsers (Strzyz et al., 2019) (using the `dep2labels` package), respectively.

Hyperparameter	Value	Hyperparameter	Value
n_char_hidden	100	ν	.9
n_feat_embed	100	ϵ	1^{-12}
embed_dropout	.33	weight_decay	0
n_lstm_hidden	400	clip	5.0
n_lstm_layers	3	min_freq	2
encoder_dropout	.33	fix_len	20
n_arc_mlp	500	decay	.75
n_rel_mlp	100	decay_steps	5000
mlp_dropout	.33	update_steps	1
encoder	lstm	feats	['tag', 'char']

Table 7: Hyperparameters used to train the `supar` models. In the case of Ancient Greek the hyperparameter *n_embed* is 100.

Hyperparameter	Value	Hyperparameter	Value
model	L2RPtr	–learning_rate	0.001
word_dim	300	–lr_decay	0.999997
char_dim	100	–beta1	0.9
pos	true	–beta2	0.9
rnn_mode	FastLSTM	–grad_clip	5.0
encoder_layers	3	–loss_type	token
decoder_layers	1	–warmup_steps	40
hidden_size	512	–reset	20
arc_space	512	–weight_decay	0.0
type_space	128	–unk_replace	0.5
p_in	0.33	–beam	5
p_out	0.33	–char_embedding	random
p_rnn	[0.33, 0.33]	–opt	adam
prior_order	inside_out	–batch_size	32
grandPar	false	–num_epochs	600
sibling	false		
activation	elu		

Table 8: Hyperparameters used to train the `syntacticpointer` models. Parameters specified from the configuration file on the left, and from the command line on the right.

Hyperparameter	Value
cnn_layer	4
char_hidden_dim	100
hidden_dim	800
dropout	0.5
lstm_layer	3
bilstm	True
learning_rate	0.02
lr_decay	0.05
momentum	0.9
l2	0
gpu	True

Table 9: Hyperparameters used to train the `dep2labels` models.

B Treebanks in each subset

In §4.2, we reviewed the related work and briefly discussed several human-defined subsets that were proposed in the past, according to a number of criteria, and that we used to report the results from our Experiment 1. Due to space reasons, we detail here in this appendix (Table 10) the specific treebanks that are part of each subset, and their sizes, for a better understanding of the particularities of each of them.

	Size	Ma18	Lh16	AG20	D21	SA17	Sm18	Ku19	Easy
Ancient Greek (PROIEL)	213K		✓				✓		
Ancient Greek (Perseus)	202K			✓					
Arabic (PADT)	282k						✓	✓	
Basque (BDT)	121K							✓	
Belarusian (HSE)	305K				✓				
Bulgarian (BTB)	156K	✓							✓
Catalan (AnCora)	546K	✓							✓
Chinese (GSD)	123K		✓	✓			✓	✓	
Coptic (Scriptorium)	48K					✓			
Czetch (FicTree)	167K								✓
Czetch (PDT)	1509K	✓	✓						
Dutch (Alpino)	208K	✓				✓			
English (EWT)	254K	✓	✓	✓			✓	✓	
Finnish (TDT)	202K		✓	✓			✓	✓	
French (GSD)	400K	✓							
Galician (TreeGal)	25K				✓				
German (GSD)	292K	✓							
Hebrew (HTB)	161K		✓	✓		✓	✓	✓	
Hindi (HDTB)	351K							✓	✓
Indonesian (GSD)	120K					✓			
Italian (ISDT)	298K	✓				✓		✓	✓
Japanese (GSD)	193K							✓	
Kazakh (KTB)	10K		✓						
Korean (GSD)	80K							✓	
Korean (Kaist)	350K						✓		
Lithuanian (HSE)	5K				✓				
Marathi (UFAL)	3K				✓				
Norwegian (Bokmaal)	310K	✓				✓			✓
Old Church Slavonic (PROIEL)	57K					✓			
Old East Slavic (RNC)	30K				✓				
Polish (LFG)	130K								✓
Polish (PDB)	350K					✓			
Romanian (RRT)	218K	✓							
Russian (GSD)	98K			✓					
Russian (SynTagRus)	1107K	✓					✓	✓	✓
Sanskrit (Vedic)	27K					✓			
Slovenian (SSJ)	140K								✓
Spanish (AnCora)	560K	✓							✓
Swedish (Talbanken)	96K						✓	✓	
Tamil (TTB)	9K		✓	✓	✓				
Turkish (IMST)	57K							✓	
Uyghur (UDT)	40K			✓					
Welsh (CCG)	36K				✓				
Wolof (WTB)	44K			✓					

Table 10: Treebanks per set

FactMix: Using a Few Labeled In-domain Examples to Generalize to Cross-domain Named Entity Recognition

Linyi Yang[1,4]*, **Lifan Yuan**[2]*†, **Leyang Cui**[3], **Wenyang Gao**[1], **Yue Zhang**[1,4]
[1] School of Engineering, Westlake University
[2] Huazhong University of Science and Technology
[3] Tencent AI Lab
[4] Institute of Advanced Technology, Westlake Institute for Advanced Study
[1]{yanglinyi, gaowenyang, zhangyue}@westlake.edu.cn
[2]lievanyuan173@gmail.com [3]leyangcui@tencent.com

Abstract

Few-shot Named Entity Recognition (NER) is imperative for entity tagging in limited resource domains and thus received proper attention in recent years. Existing approaches for few-shot NER are evaluated mainly under in-domain settings. In contrast, little is known about how these inherently faithful models perform in cross-domain NER using a few labeled in-domain examples. This paper proposes a two-step rationale-centric data augmentation method to improve the model's generalization ability. Results on several datasets show that our model-agnostic method significantly improves the performance of cross-domain NER tasks compared to previous state-of-the-art methods, including the data augmentation and prompt-tuning methods. Our codes are available at https://github.com/lifan-yuan/FactMix.

1 Introduction

Named Entity Recognition (NER) is a subtask of natural language processing, which detects the mentions of named entities in input text, such as location, organization, and person (Sang and De Meulder, 2003; Yang et al., 2017; Cui et al., 2021). It has attracted research from academia and industry due to its broadened usage in customer services and document parsing as a core task in natural language understanding (Nadeau and Sekine, 2007; Ma and Hovy, 2016; Cui and Zhang, 2019; Yamada et al., 2020). However, training data for NER is available only for limited domains. It has been shown that such labeled data introduces challenges for a model to generalize to new domains (Snell et al., 2017; Ma et al., 2021a; Lin et al., 2021).

To address this problem, a line of research considers how to allow a model to effectively learn from a few labeled examples in a new target domain (Zhang et al., 2021; Ma et al., 2021b; Das et al., 2021; Chen et al., 2022; Wang et al., 2022a,b). However, such methods still require manual labeling for target domains, which makes them difficult to generalize to zero-shot diverse domain settings. A different line of research in NER considers data augmentation, using automatically constructed labeled examples to enrich training data. Zeng et al. (2020) consider using entity replacement to generate intervened new instances. We follow this line of work and consider a new setting – how to generate NER instances for data augmentations effectively – so that a few labeled examples in a source domain can generalize to arbitrary target domains.

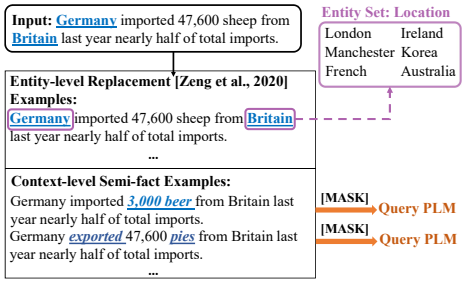

Figure 1: The demonstration of two components of FactMix, namely context-level semi-fact and entity-level semi-fact examples.

Cross-domain NER poses unique challenges in practice. First, as a structured learning problem, it is essential to understand dependencies within the labels instead of classifying each token independently (Dai and Adel, 2020). While examples from different domains usually have different dependency patterns, which inevitably brings challenges for fine-tuning few-shot NER models to cross-domain tasks (Liu et al., 2021). Second, non-entity tokens in NER do not hold unified semantic meanings, but they could become noisy when combined with entity tokens in the training set. Such compositional generalization challenges

*Equal contribution. Random order of the authorship.
†Work done at Westlake University as an intern.

Proceedings of the 29th International Conference on Computational Linguistics, pages 5360–5371
October 12–17, 2022.

have proven to be manifest in performance decay problems in various NLP tasks, such as sentiment analysis (Kaushik et al., 2019) and machine translation (Li et al., 2021), especially when faced with out-of-domain data.

As a consequence of the challenges above, spurious patterns between non-entity tokens and labels learned by models could obstruct the generalization of few-shot NER models in cross-domain settings. For example, given that "Jane monitored the patient's heart rate", 'Jane' is labeled as a person. The NER model will learn the relationship between the word 'Jane' and 'monitor' for the prediction. Suppose a NER model is trained on a medical domain and tested on the movie review. The correlation between 'Jane' and 'monitor' could become the 'spurious pattern' (Kaushik et al., 2019; Yang et al., 2021). From a causal perspective, spurious correlations are caused by confounding factors rather than a cause-effect relation.

To deal with these challenges and avoid spurious patterns, we present a novel model-agnostic, two-step, rationale-enhanced approach called **FactMix**, where we care about the efficacy of data augmentations for improving in-domain and out-of-domain (OOD) performance. We aim to leverage the contrast among – original, context-level semi-fact, and entity-level semi-fact instances – for teaching the model to capture more causal label dependencies between entities and the context. As Figure 1 shows, FactMix consists of two parts, namely context-level semi-fact generations and entity-level semi-fact instances generations. It is motivated by the natural intuition that models are much easier to learn from two-step contrastive examples compared to the one-step semi-fact augmentation (Zeng et al., 2020) [1].

The semi-factual generation component aims to alleviate the pitfall of non-entity tokens, which the previous data augmentation approach has not considered. We conduct synonym substitutions for non-entity tokens only. In particular, we mask the non-entity tokens, leverage the masked language models to predict the masked tokens, and replace the original tokens with predicted tokens. This replacement operation potentially introduces out-of-context information produced by the pre-trained masked language model when generating augmented examples. The entity-level semi-fact examples are generated by replacing the existing entity words in the training set. Finally, the augmented data generated by two steps will be mixed up together for training models. FactMix is a fully automatic method that does not require any additional hand-labeled data or human interventions and can be plugged for any few-shot NER models with different tuning strategies, including the standard fine-tuning and recent prompt-tuning.

Our method supoorts a new cross-domain NER setting, which is difficult from existing work. In particular, existing few-shot NER work considers in-domain fine-tuning (Ma et al., 2021a) and in-domain prompt-tuning (Cui et al., 2021). While our method also considers using only a source domain dataset for training models that generalize to target domains. Experimental results show that FactMix can achieve an average 3.16% performance gain in the in-domain fine-tuning setting compared to the state-of-the-art entity-level semi-fact generation approach (Zeng et al., 2020) and an average 6.85% improvement for prompt-tuning compared to EntLM (Ma et al., 2021b). Improvements in such a scale hint that FactMix builds a novel benchmark. To the best of our knowledge, we are the first to explore the cross-domain few-shot NER setting using fine-tuning and prompt-tuning methods.

2 Related Work

Cross-domain NER focuses on transferring NER models across different text styles (Pan et al., 2013; Xu et al., 2018; Liu et al., 2021; Chen et al., 2021). Current NER models cannot guarantee well-generalizing representation for out-of-domain data and result in sub-optimal performance. To address this issue, Lee et al. (2018) continue fine-tuning the model trained on the source domain by using the data from the target domain. Yang et al. (2017) jointly train NER models in both the source domain and target domain. Jia et al. (2019) and Jia and Zhang (2020) perform cross-domain knowledge transfer by using the language model. These methods rely on NER annotation or raw data in the target domain. In contrast, we propose a data argumentation method that only boosts cross-domain performance by using the source-domain corpus.

Few-shot NER aims to recognize pre-defined named entities by only using a few labeled ex-

[1] Zeng et al. (2020) use "counterfactual" to denote the setting, where augmented data contains different entities with *the same* type compared with the original data. However, strictly speaking, "counterfactual" refers to augmented data that contains *different* types of entities with a minimum change of the input that can flip the predicted label. Hence, we use semi-fact instead in our paper

amples and is commonly used for evaluating structured prediction models in recent (Ravi and Larochelle, 2016; Snell et al., 2017; Das et al., 2021). Wiseman and Stratos (2019) and Yang and Katiyar (2020) propose distance-based methods, which copy the label of nearest neighbors. Huang et al. (2021) further investigates the efficacy of the self-training method on external data based on the distance-based methods. Cui et al. (2021) and Ma et al. (2021a) adopt prompt-based methods by using BART and BERT, respectively. These methods focus on designing few-shot-friendly models without any external guidance. In contrast, we augment both entity-level semi-fact and context-level semi-fact examples to boost the model performance on the new cross-domain few-sot setting.

The area of **Few-shot Cross-domain Learning** is motivated by the ability of humans to learn object categories from a few examples at a rapid pace, which is called rationale-based learning. Inductive bias (Baxter, 2000; Zhang et al., 2020) has been identified for a long time as a critical component. Benefits from the rapid development of large-scale pre-trained language models, few-shot learning, and out-of-distribution generalization become rapidly growing fields of NLP research (Brown et al., 2020; Shen et al., 2021; Chen et al., 2022). However, these two research directions have been separately explored in down-streaming tasks but rarely discussed together, except in the very recent study of sentiment analysis (Lu et al., 2022). To the best of our knowledge, we are the first to consider this setting for NER.

Data Augmentation through deformation has been known to be effective in various text classification tasks (Feng et al., 2021; Li et al., 2022), such as sentiment analysis (Yang et al., 2021; Lu et al., 2022) and natural language inference (Kaushik et al., 2021; Wu et al., 2022). In the task of NER, self-training has been applied to automatically increase the amount of training data (Wang et al., 2020). Paul et al. (2019) propose to combine self-training with noise handling on the self-labeled data to increase the robustness of the NER model. Bansal et al. (2020) and Wang et al. (2021) develop self-training and meta-learning techniques for training NER models with few labels, respectively.

In addition to self-training methods, prompt-based (Lee et al., 2021; Ma et al., 2021b) and causal-enhanced (Zeng et al., 2020) approaches have also surfaced in this domain, which are two

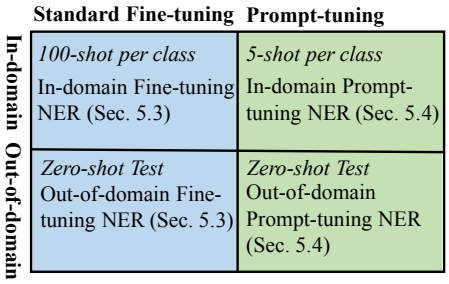

Figure 2: The categorization of experiment settings.

important baselines for our work. Zeng et al. (2020) consider using the human intervention to generate the augmented data to improve few-shot NER models, and Ma et al. (2021b) aims to leverage the template-free prompt for boosting the performance of few-shot NER models. Nevertheless, both methods only focus on the in-domain accuracy while ignoring the cross-domain generalization of few-shot NER models.

3 Settings

We investigate the effectiveness of FactMix using different methods under several settings. We first introduce task settings in Section 3.1, then show the standard fine-tuning method and prompt-based method in Section 3.2 and Section 3.3, respectively.

3.1 Task Settings

The input of the NER system is a sentence $\mathbf{x} = x_1, \ldots, x_n$, which is a sequence of n words and the output is a sequence of NER tags $\mathbf{y} = y_1, \ldots, y_n$, where $y_i \in \mathcal{Y}$ for each word and $\mathcal{Y}$ is selected from a pre-defined label set$\{B-X, I-X, S-X, E-X ... O\}$. B, I, E, S represent the beginning, middle, ending, and single-word entity, respectively. X indicates the entity type, such like PER and LOC, and O refers to the non-entity tokens. We use $\mathcal{D}_{ori}$ and $\mathcal{D}_{ood}$ to represent the original dataset and out-of-domain dataset, respectively.

Given small labelled instances of $\mathcal{D}_{ori}$, we first train a model $\mathcal{M}_{ori}$ through the standard fine-tuning method. We test the performance of $\mathcal{M}_{ori}$ on $\mathcal{D}_{ood}$ and $\mathcal{D}_{ood}$ under *In-domain Few-shot Setting* and *Out-of-domain Zero-shot Setting*, respectively, which can be seen in Figure 2.

3.2 Standard Fine-tuning Methods

Following Devlin et al. (2018), we feed contextualized word embeddings into a linear classification

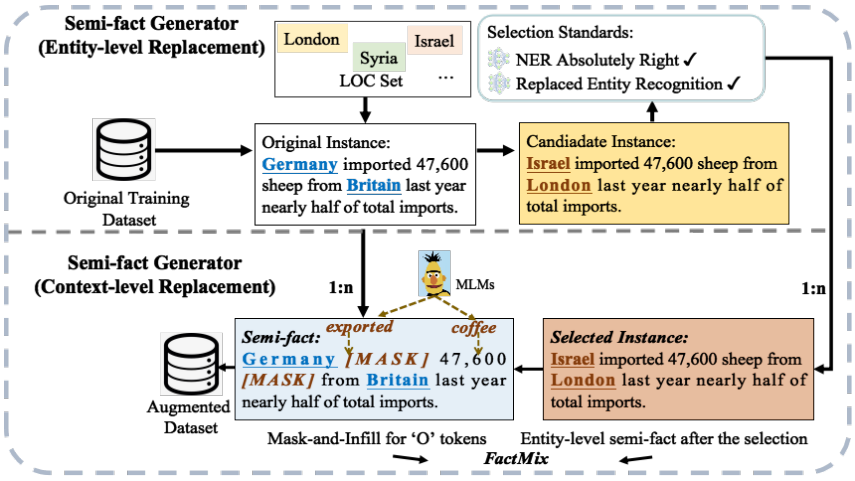

Figure 3: The pipeline of the two-step FactMix approach operated on the source domain, which consists with entity-level semi-fact generations and context-level semi-fact generations.

with the softmax function to predict the probability distribution of entity types. Formally, we first feed the input $\mathbf{x}$ into the feature encoder PLM_θ to get the corresponding contextualized word embeddings $\mathbf{h}$:

$$\mathbf{h} = PLM_\theta(\mathbf{x}), \tag{1}$$

where $\mathbf{h}$ is the sequence of contextualized word embeddings based on pre-trained language models (PLMs), i.e., BERT (Devlin et al., 2018) and RoBERTa (Liu et al., 2019). We optimize the cross entropy loss $\mathcal{L}_{NER}$ by using AdamW (Loshchilov and Hutter, 2018), which is formulated as:

$$\mathcal{L}_{NER} = -\sum_{c=1}^{N} y_{o,c} \log\left(p_{o,c}\right), \tag{2}$$

where N is the number of classes, y is the binary indicator (0 or 1) depending on if the gold label c is the correct prediction for observation o, and p is the predicted probability for observation o of c.

3.3 Prompt-tuning Methods

Prompt-tuning NER reformulates classification tasks by using the mask-and-infill technique based on human-defined templates to generate label words. We perform the template-based and template-free prompt tuning as two additional experimental scenes to verify the validity of our method. Unlike the standard fine-tuning, no new parameters are introduced in this setting.

Template-based Approach Formally, we adopt the prompt template function $F_{\text{prompt}}(\cdot)$ proposed by a very recent work (Ma et al., 2021b) to converts the input $\mathbf{x}$ to a prompt input $x_{\text{prompt}} = F_{\text{prompt}}(x)$, and pre-defined label words $\mathcal{P}$ from the label set $\mathcal{Y}$ are generated through a mapping function $\mathcal{M} : \mathcal{Y} \rightarrow \mathcal{P}$. In particular, two slots need to be infilled for each instance: the input slot [X] is filled by the original input $\mathbf{x}$ directly, and the prompt slot [Z] is filled by the label word. To be note that [Z] is predicted by the masked language model (MLM) for prompt-based tuning in this work. The probability distribution over the label set $\mathcal{Y}$ can be optimized by the softmax function for predicting masked tokens using pre-trained models.

Template-free Approach In order to reduce the computational cost of the decoding process for template-based prompt tuning, Ma et al. (2021b) propose an entity-oriented LM (EntLM) objective for fine-tuning NER. Following Ma et al. (2021b), we first construct a label word set $\mathcal{P}_f$ by the label word engineering, which is also connected with the label set through a mapping function $\mathcal{M} : \mathcal{Y} \rightarrow \mathcal{P}_f$. Next, we replace entity tokens at entity positions with corresponding label word $\mathcal{M}(y_i)$. Finally, the target input can be represented as $\mathbf{x}^{\mathbf{Rep}} = \{x_1, ..., \mathcal{M}(y_i), ..., x_n\}$. We train the language model by maximizing the probability $P\left(\mathbf{x}^{\mathbf{Rep}} \mid \mathbf{x}\right)$. The loss function for generating the prompt can be formulated as:

$$\mathcal{L}_{RepLM} = -\sum_{i=1}^{N} \log P\left(x_i = x_i^{Rep} \mid \mathbf{x}\right), \tag{3}$$

where N is the number of classes. Initial parameters of the predictive model are obtained from pre-trained language models.

4 Method

FactMix automatically generates semi-fact examples for both standard fine-tuning and prompt-tuning. The pipeline of our approach is shown in Figure 3 and is made up of three components: (1) *entity-level semi-fact generator*; (2) *context-level semi-fact generator*; (3) *augmented data selection and mixing*. Briefly, a key innovation in this work is using a mixed semi-fact generator to improve the single entity-level data augmentation approach by adding the intermediate thinking process in a human-thinking manner.

4.1 Semi-factual Generation

We randomly remove one *O* token in each sentence. Specifically, we introduce out-of-context information by randomly masking an *O* word and then filling the span using the Masked Language Model (MLM), i.e., BERT (Devlin et al., 2018). Intuitively, we can generate numerous semi-factual samples because the MLM model can fill the masked span with multiple predictions. More importantly, choosing the number and order of the selected words is a combinatorial permutation problem. However, in practice, we find that more augmented data can not always lead to a better result; and for each semi-factual sample, we only replace one word or two-word phrase in a sentence using the top one mask-and-infill prediction of MLM.

Formally, given an input of NER as $\mathbf{x} = x_1, ..., x_i, ..., x_n$, where x_i is the chosen *O* word. We first mask x_i by replacing it with the [MASK] token, and thus get $\mathbf{x} = x_1, ..., [MASK], ..., x_n$. Then we fill the [MASK] token using *BERT-base-cased*[2] model and finally obtain a semi-factual example $\mathbf{x_{semi}} = x_1, ..., x_i{}', ..., x_n$. For instance, as seen in Fig. 3, *sheep* may first be masked and then infilled by an out-of-context word *coffee*, which can be generated by PLMs.

The intervention of the selected word may inflect the entity tag of other words and introduce extra noises into the dataset. Thus, we adopt a denoising mechanism that can filter out noisy examples by leveraging the predictive model trained on the original dataset that contains prior knowledge for NER tasks. Different from Zeng et al. (2020), who filter only those samples whose replaced entities cannot be predicted correctly, we use a stricter constraint to preserve only those samples where all tokens are predicted accurately.

[2] https://huggingface.co/bert-base-cased

4.2 Entity-level Semi-fact Generation

We generate entity-level semi-fact examples by interventions on the existing entity words. Specifically, for each training sample, we randomly select one of its entity words and replace it with words of the same type in a prepared $Entity\_Base$. For cases where data is not extremely scarce, e.g. in the fine-tuning setting in our experiments, the $Entity\_Base$ can be constructed by extracting and categorizing all entity words in the original dataset. Otherwise, e.g. in the 5-shot prompt-tuning setting in our experiments, the $Entity\_Base$ should be constructed from other available datasets.

Formally, given the input as $\mathbf{x} = x_1, \ldots, x_j, \ldots, x_n$, and x_j as the chosen entity word. We assume that the label of x_j is B-LOC and extract all the B-LOC entities in the $Entity\_Base$ and denote them as B-LOC Set. Next, a word in B-LOC Set is chosen to replace x_j and denoted as $x_j{}'$. In this way, the generated semi-fact sample is $\mathbf{x_{cf}} = x_1, \ldots, x_j{}', \ldots, x_n$. For example, as seen in Fig. 3, the B-LOC entity word *German* is replaced by *Israel* in B-LOC Set. All augmented samples are labeled as the same tag with original ones for saving manual efforts.

4.3 Mix Up

In the last step, we combine two types of automatically generated data by a mix-up strategy. Although the FactMix method can generate an unlimited amount of data theoretically, past experience (Lu et al., 2022) suggests that more fact-based data instances can not always bring performance benefits accordingly.

Following Zeng et al. (2020), we set the maximum augmentation ratio as 1:8 for the entity-level semi-fact data generation. While for context-level semi-fact generations, we set the ratio as 1:5. The optimal augmentation ratios for these two kinds of augmentations are jointly selected by the grid search on the development set of in-domain data. Finally, we obtain the final FactMix augmented training data, which can be represented as $\mathbf{x_{mix}} = Concat\{\mathbf{x_{semi}}, \mathbf{x_{cf}}\}$.

5 Experiments

As shown in Table 2, we conduct experiments under the scenarios of both fine-tuning and prompt-tuning, using in-domain and out-of-domain evalua-

Domain	# Instances			Entity Types
	Train	Dev	Test	
Reuters	14,987	3,466	3,684	Person, Location, Organization, Miscellaneous
TechNews	-	-	2000	
AI	-	-	431	
Literature	-	-	416	
Music	-	-	456	
Politics	-	-	651	
Science	-	-	543	

Table 1: Statistics of datasets used in experiments.

tions. We are also interested in better understanding the contributions of the two-step data augmentation approach when it comes to prediction performance. Thus, we consider several ablation studies to better the relative contributions of entity-level and context-level semi-fact augmented data. Micro F1 is used as evaluation metric for all settings.

5.1 Methodology

Fine-tuning. Given that FactMix is a model-agnostic data augmentation approach, we adopt the standard fine-tuning method based on two pre-trained models with different parameter sizes: BERT-base, BERT-large, RoBERT-base, and RoBERT-large. All backbone models are implemented on the transformer package provided by Huggingface [3]. To fine-tune NER models in a few-shot setting, we randomly sample 100 instances per label from the original dataset to ensure that the model converges. We report the average performance of models trained by five-times training.

Prompt-tuning. We adopt the recent EntLM model proposed by Ma et al. (2021b) as the benchmark for prompt-tuning. Following Ma et al. (2021b), we conduct the prompt-based experiments using the 5-shot training strategy. Again, we conduct a comparison between the state-of-the-art prompt-tuning method and several variants of FactMix. We also analyze the separate contribution of the counterfactual generator and semi-fact generator by providing an ablation study based on the the base and large versions of the BERT-cased backbone. For the standard hold-out test, we report results on both development and test sets. We also select two representative datasets for the out-of-domain test in terms of the highest (TechNews) and lowest (Science) word overlap with the original training domain (Reuters).

[3] https://huggingface.co/models

Dataset	Backbone	In-domain Fine-tuning Results			
		Ori	CF	Semi	FactMix
CoNLL2003 (Dev)	BERT-base-cased	57.98	79.78	81.48	**83.13***
	BERT-large-cased	69.18	83.27	85.87	**85.73***
	RoBERTa-base	52.44	85.81	87.99	**88.51***
	RoBERTa-large	68.81	88.25	89.39	**89.95***
CoNLL2003 (Test)	BERT-base-cased	54.03	77.71	78.70	**80.10***
	BERT-large-cased	65.38	81.11	**83.04**	82.65
	RoBERTa-base	48.53	82.74	85.05	**85.33***
	RoBERTa-large	65.70	85.20	86.84	**86.91***

Table 2: The Micro F1 score of different models by using FactMix and related data augmentation methods – CF: Entity-level Semi-fact Generation (Zeng et al., 2020); Semi: Context-level Semi-fact Generation (Ours); FactMix (Ours) – using the in-domain few-shot fine-tuning. $*$ indicates the statistically significant under T-test, $p<0.05$.

5.2 Datasets

The statistics of both source domain and out-of-domain datasets are introduced in Table 1. As a common understanding, it is easy to collect a large unlabeled corpus for one domain, while the corpus size could be small for low-resource domains. Then, we introduce datasets used in experiments for in-domain tests and out-of-domain tests, respectively, as follows.

In-domain Dataset. We conduct the in-domain experiments on the widely used CoNLL2003 (Sang and De Meulder, 2003) dataset with a text style of Reuters News and categories of person, location, organization, and others.

Out-of-domain Datasets. We adopt the cross-domain dataset collected by Liu et al. (2021) with new domains of AI, Literature, Music, Politics, and Science. Vocabularies for each domain are created by considering the top 5K most frequent words (excluding stopwords). Liu et al. (2021) report that vocabulary overlaps between domains are generally small, which further illustrates that the overlaps between domains are comparably small and out-of-domain datasets are diverse. Notably, since the model trained on CoNLL2003 can only predict person, location, organization, and various entities, we set all the unseen labels in OOD datasets to *O*.

5.3 Results on Few-shot Fine-tuning

In-domain experimental results on a widely used CoNLL2003 dataset show that FactMix achieves an average **3.16%** performance gain in the in-domain fine-tuning setting (100 instances per class) and an average **2.81%** improvement for prompt-tuning (5 instances per class) compared to the state-of-the-art data augmentation approach. For OOD test results,

Dataset	Backbone	Fine-tuning OOD Results				Dataset	Fine-tuning OOD Results			
		Ori	CF	Semi	FactMix		Ori	CF	Semi	FactMix
TechNews	BERT-base-cased	41.46	61.20	**65.20***	65.09*	Music	10.46	19.33	17.59	**19.49**
	BERT-large-cased	52.63	67.51	**69.98***	69.28		12.00	19.64	19.32	**19.97***
	RoBERTa-base	44.88	71.83	73.15	**73.62***		11.78	22.24	21.37	**23.75***
	RoBERTa-large	51.76	73.11	**74.89***	74.62		14.44	21.13	**22.93***	20.96
AI	BERT-base-cased	15.88	22.49	23.66	**24.67***	Politic	21.38	41.84	40.82	**43.60***
	BERT-large-cased	18.62	26.00	26.03	**26.25***		29.77	43.37	42.57	**43.84***
	RoBERTa-base	18.63	32.03	29.79	**32.09**		26.81	44.12	44.09	**44.66***
	RoBERTa-large	23.27	28.76*	29.77*	**30.06***		28.56	**45.87**	44.36	45.05
Literature	BERT-base-cased	12.85	22.89	23.05	**25.70***	Science	12.41	25.67	28.26	**29.72***
	BERT-large-cased	17.53	24.96	**26.25***	25.39		16.05	**28.75**	27.02	27.88
	RoBERTa-base	15.05	28.21	27.90	**28.89***		14.17	33.33	31.06	**34.13***
	RoBERTa-large	19.20	25.43	**26.76***	26.30*		17.25	31.36	29.89	**32.39***

Table 3: The average five times running results of Fine-tuning OOD over six datasets using various data augmentation approaches compared to the original training method (Standard Fine-tuning). CF: Entity-level Semi-fact Generation (Zeng et al., 2020); Semi: Context-level Semi-fact Generation (Ours); FactMix (Ours). $*$ indicates the statistically significant under T-test, $p<0.05$, when compared to CF.

Dataset	Backbone	Prompt-tuning In-domain Results				Dataset	Prompt-tuning OOD Results			
		EntLM	CF	Semi	FactMix		EntLM	CF	Semi	FactMix
CoNLL2003 (Dev)	BERT-base-cased	51.73	48.14	59.30	**62.40***	TechNews	47.16	52.36	50.96	**52.44***
	BERT-large-cased	60.95	58.42	49.53	**61.64***		**52.53**	48.32	32.48	48.64
CoNLL2003 (Test)	BERT-base-cased	54.00	55.61	57.23	**59.19***	Science	15.70	18.32	17.28	**18.62***
	BERT-large-cased	60.37	56.49	58.37	**60.80***		15.32	15.34	13.01	**16.80***

Table 4: The comparison among our methods, counterfactual data augmentation, and EntLM (Ma et al., 2021b) using prompt-tuning hold-out test and OOD test. $*$ indicates the statistically significant under T-test, $p<0.05$, when compared to EntLM.

FactMix increases absolute **14.19%** F1 score in average in fine-tuning compared to (Zeng et al., 2020) and **1.45%** increase in prompt-tuning compared to (Ma et al., 2021b).

In-domain Fine-tuning results are presented in Table 2 under the standard fine-tuning setting, using each of the baselines (Ori) and several variations of our FactMix approach. All results average five times running with randomly training instance selections.

FactMix achieves the best performance on both development and test sets, in terms of the highest Micro F1 score, excluding that the BERT-large model can achieve the best performance using our semi-fact augmentation approach only. Furthermore, we observe that improvements introduced by variants of the data augmentation approach are relatively significant when compared to models trained without data augmentations (**25.3%** absolute F1 improvements on average). FactMix also shows its superior performance compared to the previous state-of-the-art data augmentation method (Zeng et al., 2020) with a 2.1% absolute improvement in average. Finally, FactMix establishes a new state-of-the-art for the data augmentation approach in the cross-domain few-shot NER.

Out-of-domain Fine-tuning. We consider the performance of few-shot NER in the context of a more challenging cross-domain setting. The micro-f1 score of pre-trained models based on different augmentation methods is shown in Table 3. We find that the performance decay in technews is relatively lower than other domains since the technews domain also holds a relatively higher overlap with the training set (Reuters News). Again, our semi-factual generation and FactMix achieve the best performance in most settings. For instance, the RoBERTa-large model trained with Semi-fact Only and FactMix can achieve 74.89% and 74.62% F1, respectively, compared to only 51.76% F1 using the original training set. We also notice that all pre-trained methods manifest a significant drop in accuracy on other datasets, which share fewer overlaps with the training data than technews. For example, the RoBERTa-base model gets an **11.78%** F1 by using the standard fine-tuning, while it can be improved to **23.75%** with FactMix. Moreover, we can see that our methods, including Semi-fact and FactMix, achieve a significantly consistent improvement over different datasets compared to standard fine-tuning and the previous state-of-the-art method (Zeng et al., 2020), no matter the dataset distribution gap between domains. Finally, the ablation study shows that the mix-up strategy can

effectively improve the performance of fine-tuning methods in most scenarios, compared to the single semi-fact augmentation method.

5.4 Results on Few-shot Prompt-tuning

To further understand the benefits of FactMix, in what follows, we also consider several ablation studies based on the few-shot prompt-tuning setting (5 instances per class).

In-domain Prompt-tuning. The results are shown in Table 4. We can see that FactMix achieves the best performance in 5-shot prompt-tuning on the development set and test set of CoNLL2003, compared to EntLM (Ma et al., 2021b) and the ablation part of FactMix. The overall Micro F1 score of prompt-tuning with FactMix is relatively lower than the results of 100-shot fine-tuning, i.e., **88.51 vs. 60.80** based on the BERT-large model. It is noteworthy that our approach shows its superior for all settings, while the previous data augmentation approach (Zeng et al., 2020) hurts the performance when using the BERT-large models, i.e., the F1 score decreases from 60.37 to 56.49 as shown in the test set. The stable performance further proves that two-step fact-based augmentations can significantly benefit NER models for both fine-tuning and prompt-tuning models.

Out-of-domain Prompt-tuning. The OOD results for prompt-tuning methods are also shown in Table 4. In general, we observe that prompt-based tuning methods have considerable potential for the cross-domain few-shot NER. While cross-domain results evaluated on the high-overlap dataset (TechNews) with the training domain are significantly higher than the low-overlap dataset (Science), i.e., **52.44 vs. 18.62** based on BERT-base. Furthermore, FactMix provides the best performance based on all of the pre-trained models, compared to EntLM and its variants. In contrast, EntLM performs better than FactMix on TechNews. It hints that our method could be more useful in a low-resource setting where the overlap between the original domain and target domain is relatively low.

5.5 Discussion

Benefiting from the generalized ability of pre-trained models, FactMix achieves much improved results on the few-shot in-domain test – 86.91%. More importantly, it shows decent scalability when combined with fine-tuning and prompt-tuning methods. To better understand the influence of the number of initial training examples and augmentation ratios, we illustrate the comparison of in-domain fine-tuning as follows.

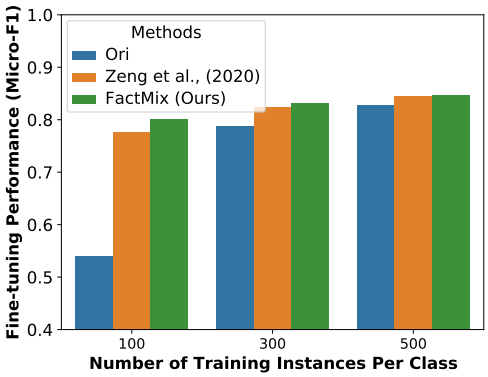

Figure 4: In-domain fine-tuning results are reported based on the BERT-base-cased model.

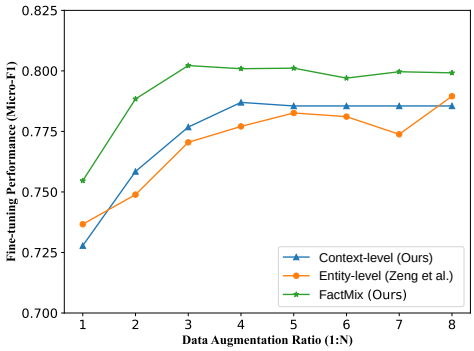

Figure 5: In-domain fine-tuning results based on BERT-base-cased using different augmentation ratios.

The Influence of Training Samples. The comparison of results based on the BERT-base-cased model is shown in Figure 4. We present the results of three different methods by using the different number of training examples varying from 100 to 500. Results show that FactMix holds the best performance when the size of training examples has been set as 100, 300, and 500. We also notice that the improvements introduced by FactMix decreased as the amount of raw training data per class increased from 100 to 300 when compared to the standard fine-tuning method. Finally, our method shows its superior for all settings when compared to the previous state-of-the-art data augmentation method (Zeng et al., 2020) for Few-shot NER.

The Influence of Augmentation Ratios. In-domain fine-tuning results using different augmentation ratios are shown in Figure 5. We consider three approaches in the evaluation, including semi-factual generation, FactMix, and the baseline

method (Zeng et al., 2020). FactMix shows its absolute performance advantage using the augmentation ratio from one to eight. In particular, Micro-F1 scores of all methods increase with the increase of the number of augmented training instances when the augmentation ratio is less than 1:4, whereas the trend of increase gradually slow down when generating examples more than 1:4.

6 Conclusion

We proposed a joint context-level and entity-level semi-fact generation framework, FactMix, for better cross-domain NER using few labeled in-domain examples. Experimental results show that our method can not only boost the performance of pre-trained backbones in in-distribution and OOD datasets, but also show promising results combined with template-free prompt-tuning methods. As a single data augmentation method, FactMix can be useful for different NLP tasks to enable fast generalization, i.e., relation extraction, question answering, and sentiment analysis.

Acknowledgements

We acknowledge with thanks the discussion with Qingkai Min from Westlake University, as well as the many others who have helped. We would also like to thank anonymous reviewers for their insightful comments and suggestions to help improve the paper. This publication has emanated from research conducted with the financial support of the Pioneer and "Leading Goose" R&D Program of Zhejiang under Grant Number 2022SDXHDX0003. Yue Zhang is the corresponding author.

References

Trapit Bansal, Rishikesh Jha, Tsendsuren Munkhdalai, and Andrew McCallum. 2020. Self-supervised meta-learning for few-shot natural language classification tasks. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 522–534.

Jonathan Baxter. 2000. A model of inductive bias learning. *Journal of artificial intelligence research*, 12:149–198.

Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. 2020. Language models are few-shot learners. *Advances in neural information processing systems*, 33:1877–1901.

Jiawei Chen, Qing Liu, Hongyu Lin, Xianpei Han, and Le Sun. 2022. Few-shot named entity recognition with self-describing networks. *arXiv preprint arXiv:2203.12252*.

Shuguang Chen, Gustavo Aguilar, Leonardo Neves, and Thamar Solorio. 2021. Data augmentation for cross-domain named entity recognition. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 5346–5356.

Leyang Cui, Yu Wu, Jian Liu, Sen Yang, and Yue Zhang. 2021. Template-based named entity recognition using BART. In *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*, pages 1835–1845, Online. Association for Computational Linguistics.

Leyang Cui and Yue Zhang. 2019. Hierarchically-refined label attention network for sequence labeling. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 4115–4128, Hong Kong, China. Association for Computational Linguistics.

Xiang Dai and Heike Adel. 2020. An analysis of simple data augmentation for named entity recognition. *arXiv preprint arXiv:2010.11683*.

Sarkar Snigdha Sarathi Das, Arzoo Katiyar, Rebecca J Passonneau, and Rui Zhang. 2021. Container: Few-shot named entity recognition via contrastive learning. *arXiv preprint arXiv:2109.07589*.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*.

Steven Y Feng, Varun Gangal, Jason Wei, Sarath Chandar, Soroush Vosoughi, Teruko Mitamura, and Eduard Hovy. 2021. A survey of data augmentation approaches for nlp. In *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*, pages 968–988.

Jiaxin Huang, Chunyuan Li, Krishan Subudhi, Damien Jose, Shobana Balakrishnan, Weizhu Chen, Baolin Peng, Jianfeng Gao, and Jiawei Han. 2021. Few-shot named entity recognition: An empirical baseline study. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 10408–10423, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Chen Jia, Xiaobo Liang, and Yue Zhang. 2019. Cross-domain ner using cross-domain language modeling. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 2464–2474.

Chen Jia and Yue Zhang. 2020. Multi-cell compositional lstm for ner domain adaptation. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 5906–5917.

Divyansh Kaushik, Eduard Hovy, and Zachary C Lipton. 2019. Learning the difference that makes a difference with counterfactually-augmented data. *arXiv preprint arXiv:1909.12434*.

Divyansh Kaushik, Amrith Rajagopal Setlur, Eduard H. Hovy, and Zachary Chase Lipton. 2021. Explaining the efficacy of counterfactually-augmented data. *ArXiv*, abs/2010.02114.

Dong-Ho Lee, Mahak Agarwal, Akshen Kadakia, Jay Pujara, and Xiang Ren. 2021. Good examples make a faster learner: Simple demonstration-based learning for low-resource ner. *arXiv preprint arXiv:2110.08454*.

Ji Young Lee, Franck Dernoncourt, and Peter Szolovits. 2018. Transfer learning for named-entity recognition with neural networks. In *Proceedings of the Eleventh International Conference on Language Resources and Evaluation (LREC 2018)*, Miyazaki, Japan. European Language Resources Association (ELRA).

Bohan Li, Yutai Hou, and Wanxiang Che. 2022. Data augmentation approaches in natural language processing: A survey. *AI Open*.

Yafu Li, Yongjing Yin, Yulong Chen, and Yue Zhang. 2021. On compositional generalization of neural machine translation. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 4767–4780.

Bill Yuchen Lin, Wenyang Gao, Jun Yan, Ryan Moreno, and Xiang Ren. 2021. Rockner: A simple method to create adversarial examples for evaluating the robustness of named entity recognition models. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 3728–3737.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Roberta: A robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692*.

Zihan Liu, Yan Xu, Tiezheng Yu, Wenliang Dai, Ziwei Ji, Samuel Cahyawijaya, Andrea Madotto, and Pascale Fung. 2021. Crossner: Evaluating cross-domain named entity recognition. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 35, pages 13452–13460.

Ilya Loshchilov and Frank Hutter. 2018. Decoupled weight decay regularization. In *International Conference on Learning Representations (ICLR)*.

Jinghui Lu, Linyi Yang, Brian Mac Namee, and Yue Zhang. 2022. A rationale-centric framework for human-in-the-loop machine learning. *arXiv preprint arXiv:2203.12918*.

Ruotian Ma, Xin Zhou, Tao Gui, Yiding Tan, Qi Zhang, and Xuanjing Huang. 2021a. Template-free prompt tuning for few-shot NER. *CoRR*, abs/2109.13532.

Ruotian Ma, Xin Zhou, Tao Gui, Yiding Tan, Qi Zhang, and Xuanjing Huang. 2021b. Template-free prompt tuning for few-shot ner. *arXiv preprint arXiv:2109.13532*.

Xuezhe Ma and Eduard Hovy. 2016. End-to-end sequence labeling via bi-directional LSTM-CNNs-CRF. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1064–1074, Berlin, Germany. Association for Computational Linguistics.

David Nadeau and Satoshi Sekine. 2007. A survey of named entity recognition and classification. *Lingvisticae Investigationes*, 30(1):3–26.

Sinno Jialin Pan, Zhiqiang Toh, and Jian Su. 2013. Transfer joint embedding for cross-domain named entity recognition. *ACM Transactions on Information Systems (TOIS)*, 31(2):1–27.

Debjit Paul, Mittul Singh, Michael A Hedderich, and Dietrich Klakow. 2019. Handling noisy labels for robustly learning from self-training data for low-resource sequence labeling. *NAACL HLT 2019*, page 29.

Sachin Ravi and Hugo Larochelle. 2016. Optimization as a model for few-shot learning. In *ICLR*.

Erik Tjong Kim Sang and Fien De Meulder. 2003. Introduction to the conll-2003 shared task: Language-independent named entity recognition. In *Proceedings of the Seventh Conference on Natural Language Learning at HLT-NAACL 2003*, pages 142–147.

Zheyan Shen, Jiashuo Liu, Yue He, Xingxuan Zhang, Renzhe Xu, Han Yu, and Peng Cui. 2021. Towards out-of-distribution generalization: A survey. *ArXiv*, abs/2108.13624.

Jake Snell, Kevin Swersky, and Richard Zemel. 2017. Prototypical networks for few-shot learning. *Advances in neural information processing systems*, 30.

Yaqing Wang, Subhabrata Mukherjee, Haoda Chu, Yuancheng Tu, Ming Wu, Jing Gao, and Ahmed Hassan Awadallah. 2020. Adaptive self-training for few-shot neural sequence labeling. *arXiv preprint arXiv:2010.03680*.

Yaqing Wang, Subhabrata Mukherjee, Haoda Chu, Yuancheng Tu, Ming Wu, Jing Gao, and Ahmed Hassan Awadallah. 2021. Meta self-training for few-shot neural sequence labeling. In *Proceedings of the 27th ACM SIGKDD Conference on Knowledge Discovery & Data Mining*, pages 1737–1747.

Yidong Wang, Hao Chen, Yue Fan, SUN Wang, Ran Tao, Wenxin Hou, Renjie Wang, Linyi Yang, Zhi Zhou, Lan-Zhe Guo, et al. 2022a. Usb: A unified semi-supervised learning benchmark. *arXiv preprint arXiv:2208.07204*.

Yidong Wang, Hao Chen, Qiang Heng, Wenxin Hou, Marios Savvides, Takahiro Shinozaki, Bhiksha Raj, Zhen Wu, and Jindong Wang. 2022b. Freematch: Self-adaptive thresholding for semi-supervised learning. *arXiv preprint arXiv:2205.07246*.

Sam Wiseman and Karl Stratos. 2019. Label-agnostic sequence labeling by copying nearest neighbors. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 5363–5369, Florence, Italy. Association for Computational Linguistics.

Yuxiang Wu, Matt Gardner, Pontus Stenetorp, and Pradeep Dasigi. 2022. Generating data to mitigate spurious correlations in natural language inference datasets.

Jingjing Xu, Hangfeng He, Xu Sun, Xuancheng Ren, and Sujian Li. 2018. Cross-domain and semisupervised named entity recognition in chinese social media: A unified model. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 26(11):2142–2152.

Ikuya Yamada, Akari Asai, Hiroyuki Shindo, Hideaki Takeda, and Yuji Matsumoto. 2020. LUKE: Deep contextualized entity representations with entity-aware self-attention. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 6442–6454, Online. Association for Computational Linguistics.

Linyi Yang, Jiazheng Li, Pádraig Cunningham, Yue Zhang, Barry Smyth, and Ruihai Dong. 2021. Exploring the efficacy of automatically generated counterfactuals for sentiment analysis. *arXiv preprint arXiv:2106.15231*.

Yi Yang and Arzoo Katiyar. 2020. Simple and effective few-shot named entity recognition with structured nearest neighbor learning. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 6365–6375, Online. Association for Computational Linguistics.

Zhilin Yang, Ruslan Salakhutdinov, and William W. Cohen. 2017. Transfer learning for sequence tagging with hierarchical recurrent networks.

Xiangji Zeng, Yunliang Li, Yuchen Zhai, and Yin Zhang. 2020. Counterfactual generator: A weakly-supervised method for named entity recognition. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 7270–7280, Online. Association for Computational Linguistics.

Bowen Zhang, Yidong Wang, Wenxin Hou, Hao Wu, Jindong Wang, Manabu Okumura, and Takahiro Shinozaki. 2021. Flexmatch: Boosting semi-supervised learning with curriculum pseudo labeling. *Advances in Neural Information Processing Systems*, 34:18408–18419.

Yufeng Zhang, Xueli Yu, Zeyu Cui, Shu Wu, Zhongzhen Wen, and Liang Wang. 2020. Every document owns its structure: Inductive text classification via graph neural networks. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 334–339.

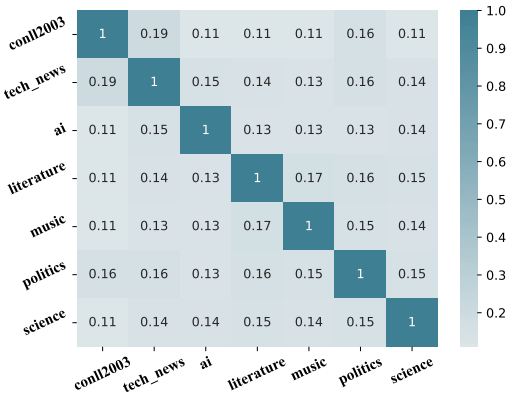

Figure 6: The word overlap between NER datasets from different domains.

A Appendix: Domain Distributions

The similarity between the dataset of source domain and six out-of-domain datasets is shown in Figure 6. We find that the technical news dataset shares the highest overlap ratio with the CoNLL2003 dataset, while the science domain shares the lowest overlap. Based on that, we select TechNews and Science as two representative datasets in prompt-tuning experiments. Also, the experimental results shown in Tables 3 and 4 demonstrate that cross-domain transfer between low-overlap domains still be a challenge problem, even for FactMix.

Speaker-Aware Discourse Parsing on Multi-Party Dialogues

Nan Yu[1], Guohong Fu[1,2]*, Min Zhang[3]
[1]School of Computer Science and Technology, Soochow University, China
[2]Institute of Artificial Intelligence, Soochow University, China
[3]Institute of Computing and Intelligence, Harbin Institute of Technology (Shenzhen), China
nyu@stu.suda.edu.cn,{ghfu,minzhang}@suda.edu.cn

Abstract

Discourse parsing on multi-party dialogues is an important but difficult task in dialogue systems and conversational analysis. It is believed that speaker interactions are helpful for this task. However, most previous research ignores speaker interactions between different speakers. To this end, we present a speaker-aware model for this task. Concretely, we propose a speaker-context interaction joint encoding (SCIJE) approach, using the interaction features between different speakers. In addition, we propose a second-stage pre-training task, same speaker prediction (SSP), enhancing the conversational context representations by predicting whether two utterances are from the same speaker. Experiments on two standard benchmark datasets show that the proposed model achieves the best-reported performance in the literature. We will release the codes of this paper to facilitate future research[1].

1 Introduction

Discourse parsing on multi-party dialogues aims to identify the discourse relations between utterances in dialogues, which has received increasing attention in the natural language processing (NLP) community (Shi and Huang, 2019; He et al., 2021; Liu and Chen, 2021; Yang et al., 2021). Unlike traditional text-level discourse parsing based on the Rhetorical Structure Theory (RST) (Mann and Thompson, 1988) and the Penn Discourse TreeBank (PDTB) (Prasad et al., 2008), this task is performed based on the Segmented Discourse Relation Theory (SDRT) (Asher et al., 2003). It represents a multi-party dialogue by a discourse dependency tree (Afantenos et al., 2015). Figure 1 shows an example. The leaf nodes are utterances, and the arcs indicate the discourse relations between utterances. Each utterance is referred as an elementary discourse unit (EDU) in SDRT discourse parsing.

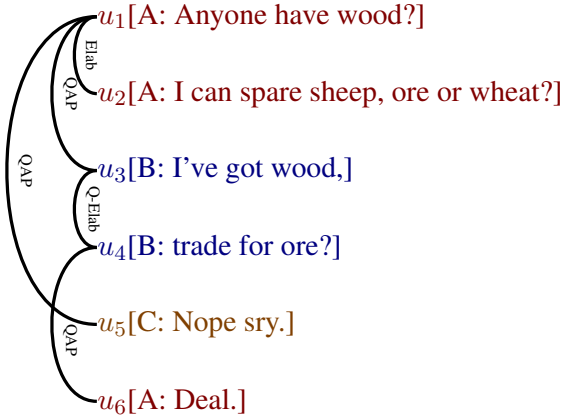

Figure 1: An example of a discourse dependency tree. u_1, u_2, u_3, u_4, u_5 refer to EDUs. "Q-Elab", "QAP", "Q-Elab", and "Elab" refer to discourse relations. "A", "B", and "C" are three speakers.

A multi-party dialogue has several aspects that make its discourse parsing more challenging than that of a written text created by one author. It involves multiple speakers who interact with each other in different roles during turn shifting and make contributions to the interactions with multiple potential threads (Afantenos et al., 2015). Therefore, in addition to conversational contexts, speaker interactions are also important cues in determining the discourse structure of a multi-party dialogue.

Most current research for discourse parsing on multi-party dialogues focuses on conversational context modeling with different methods. A pioneer study by Afantenos et al. (2015) adopts a statistical model for this task, using human-designed features extracted from conversational contexts, while an early neural research by Shi and Huang (2019) proposes a deep sequential model, using hierarchical GRUs to learn conversational contextual cues for discourse parsing. Recent research exerts more efforts on integrating rich information with context modeling and explores different techniques such as domain adaptation (Liu and Chen, 2021), edge-centric encoding (Wang et al., 2021), multi-task

*Corresponding author.
[1]https://github.com/yunan4nlp/SA-DPMD

Proceedings of the 29th International Conference on Computational Linguistics, pages 5372–5382
October 12–17, 2022.

learning (He et al., 2021), and joint model (Yang et al., 2021).

Although above approaches give competitive performances on discourse parsing on multi-party dialogues, only a few studies (Afantenos et al., 2015; Shi and Huang, 2019; Wang et al., 2021) consider speaker interactions. These studies use EDU pair features to represent speaker interactions and demonstrate that introducing speaker interactions is beneficial to this task. However, the EDU pair based speaker interaction modeling only represent whether an EDU pair is from the same speaker. The informative interaction between different speakers remains unexplored. As shown in Figure 1, the connected EDUs u_1 and u_5 are from two different speakers, but the EDU pair features will not clearly tell who said the EDUs. Since a general multi-party dialogue involves more than two speakers, the problem could be extremely serious.

To alleviate the above problem, we propose a speaker-aware model for discourse parsing on multi-party dialogues. Concretely, to handle the interactive information between the same speaker within a dialogue, we present SSP-BERT, a second-stage pre-training method based on BERT that is designed to predict whether two EDUs are from the same speaker. Based on SSP-BERT, we investigate a speaker-context interactions joint encoding (SCIJE) approach to handle the interactions between different speakers. First, we follow the node-centric based encoding approach (Shi and Huang, 2019; Liu and Chen, 2021), adopting BERT and BiGRU to represent conversational contexts. Then we embed the speaker sequence of each dialogue to vectors and feed them into BiGRU to further obtain speaker interaction representations. We finally combine them and thus obtain speaker-context interaction joint representations.

We conduct experiments on STAC (Asher et al., 2016) and Molweni (Li et al., 2020) to evaluate our proposed model. Experimental results show that SSP-BERT is highly competitive for discourse parsing on multi-party dialogues. When the speaker-context interaction joint representations are integrated, the proposed model is able to obtain further improvements. Our proposed model achieves the best performance among all the state-of-the-art (SOTA) models reported in the literature.

In summary, we mainly make the following three contributions in this paper:

- We propose SCIJE for discourse parsing on multi-party dialogues, which is capable of modeling the interactions between different speakers.
- We propose a second-stage pre-training approach to integrate the interaction features between the same speaker into conversational context representations.
- Our final model achieves the SOTA performance on two benchmark datasets.

2 Related Work

Text-level discourse parsing can be categorized into two types: the RST-style (Mann and Thompson, 1988) and the PDTB-style (Prasad et al., 2008) parsing. Both tasks have been intensively investigated since early (Lin et al., 2014; Li et al., 2014). Compared with text-level discourse parsing, discourse parsing on multi-party dialogues is still at its early stage. The pioneer study (Afantenos et al., 2015) mainly borrows the dependency parsing paradigm from RST-style parsing (Li et al., 2014) for this task, using human-designed features. Recently, inspired by the success of neural discourse parsing models (Braud et al., 2016, 2017; Yu et al., 2018), several neural discourse parsing models for multi-party dialogues have been proposed as well (Shi and Huang, 2019; He et al., 2021; Liu and Chen, 2021; Yang et al., 2021; Wang et al., 2021). In this paper, we follow the line of the work using neural models to this task.

It is believed that speaker interactions are helpful for modeling multi-party dialogues, giving great improvements on language modeling (Zhang and Zhao, 2021), dialogue comprehension (Ma et al., 2021, 2022). In discourse parsing, Afantenos et al. (2015) extract hand-crafted features from the EDU pair that have the same speaker, and feed them into a statistical discourse parsing model. Shi and Huang (2019) use a speaker highlight mechanism to represent speaker interactions. Wang et al. (2021) treat speaker interactions as edges of EDUs, feeding them into graph neural network (GNN) to obtain edge-centric representations. However, these speaker interaction models based on EDU pairs only indicate whether two EDUs are from the same speaker, ignoring the interactions between different speakers. In this paper, we investigate the interaction features between different speakers, using them as a strong supplementary for the context representations.

Recent research investigates pre-training on dia-

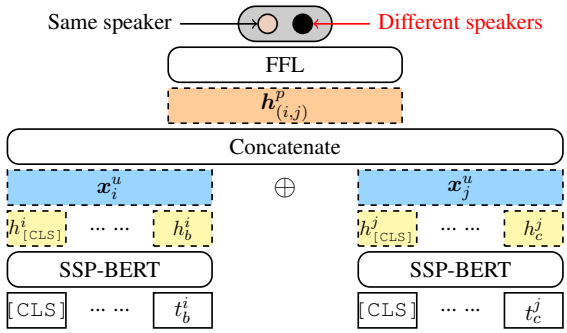

Figure 2: Framework of the SSP task.

logues intensively (Henderson et al., 2020; Zhang et al., 2020; Xu et al., 2021; Zhang and Zhao, 2021; **?**). Almost all studies focus on capturing coherance between utterances by using pre-training tasks such as dialogue generation or response selection. In this work, we enhance the conversational context representations with interaction features between the same speaker.

3 Our Proposed Model

3.1 SSP-BERT

In order to integrate same speaker interactions into contextual representations, we present SSP-BERT, a second stage pre-training method based on BERT. The approach is mainly inspired by Yu et al. (2022), which pre-train XLNet (**?**) with two EDU-level tasks in the second stage. Here we change the original approach of Yu et al. (2022) to match discourse parsing on multi-party dialogues. As shown in Figure 2, we sample the EDU pair from dialogues, and adopt SSP-BERT to predicts whether two EDUs have the same speaker. Concretely, given an EDU pair u_j and u_i, we exploit BERT to encode them respectively, obtaining corresponding token embeddings.

$$\begin{aligned} u_i &= \{\texttt{[CLS]}, t_1^i, ..., t_m^i\} \\ u_j &= \{\texttt{[CLS]}, t_1^j, ..., t_n^j\} \\ \boldsymbol{h}^i_{\texttt{[CLS]}}, \boldsymbol{h}^i_1, ..., \boldsymbol{h}^i_m &= \text{BERT}(u_i) \\ \boldsymbol{h}^j_{\texttt{[CLS]}}, \boldsymbol{h}^j_1, ..., \boldsymbol{h}^j_n &= \text{BERT}(u_j) \end{aligned} \tag{1}$$

We choose the representation of "[CLS]" as the corresponding EDU representation, and then concatenate these two EDU representations as the representation of the EDU pair:

$$\boldsymbol{h}^p_{(j,i)} = \boldsymbol{h}^i_{\texttt{[CLS]}} \oplus \boldsymbol{h}^j_{\texttt{[CLS]}} \tag{2}$$

When the EDU pair representation is ready, we feed it into a feed forward layer (FFL):

$$\boldsymbol{y}^p = \boldsymbol{W}^p \boldsymbol{h}^p_{(j,i)} \tag{3}$$

where $\boldsymbol{W}^p$ is a learnable model parameter and $\boldsymbol{y}^p$ is the output scores.

3.2 Discourse Parsing Model

Our discourse parsing model follows an encoder-decoder framework. As shown by the bottom of Figure 3, the encoder represents the speakers and the contexts to speaker-context interaction joint representations. The top of Figure 3 shows the decoder. It predicts the links and their corresponding relations between EDUs.

3.2.1 Encoder

Speaker Interaction Representation Here we introduce the approach of obtaining the speaker interaction representations. Given a dialogue with n turn, we first gather the speaker sequence with n length. For instance, we can obtain the corresponding speaker sequence $\{\text{A}, \text{A}, \text{B}, \text{B}, \text{C}, \text{A}\}$ from the dialogue in Figure 1. Then we embed the speaker sequence to the speaker vectors, and use BiGRU to encode these speaker vectors, obtaining speaker representations:

$$\begin{aligned} \boldsymbol{x}^s_{\text{A}}, ..., \boldsymbol{x}^s_{\text{C}}, \boldsymbol{x}^s_{\text{A}} &= \text{A}, ..., \text{C}, \text{A} \\ \boldsymbol{h}^s_1, ..., \boldsymbol{h}^s_{n-1}, \boldsymbol{h}^s_n &= \text{BiGRU}(\boldsymbol{x}^s_{\text{A}}, ..., \boldsymbol{x}^s_{\text{C}}, \boldsymbol{x}^s_{\text{A}}) \end{aligned} \tag{4}$$

We concatenate two speaker representations to further obtain the speaker interaction representation:

$$\boldsymbol{h}^s_{(j,i)} = \boldsymbol{h}^s_j \oplus \boldsymbol{h}^s_i \tag{5}$$

where $\oplus$ is a concatenate operation, $\boldsymbol{h}^s_{(i,j)}$ denotes the speaker interaction representation.

Context Interaction Representation We borrow the node-centric encoding approaches (Shi and Huang, 2019; Liu and Chen, 2021) to represent the conversational contexts. It consists of BERT and BiGRU. The BERT layer is used to represent sequential tokens in EDUs, and the BiGRU layer is used to represent sequential EDUs. Concretely, for each input EDU u_i, first we tokenize it by byte pair encoding (BPE) and then place a [CLS] before it. By this way, the input tokens of the first layer BERT are $\{[\texttt{CLS}], t_1^i, ..., t_m^i\}$. Thus we adopt BERT to represent these input tokens:

$$\begin{aligned} u_i &= \texttt{[CLS]}, t_1^i, ..., t_m^i \\ \boldsymbol{h}^i_{\texttt{[CLS]}}, \boldsymbol{h}^i_1, ..., \boldsymbol{h}^i_m &= \text{BERT}(u_i) \end{aligned} \tag{6}$$

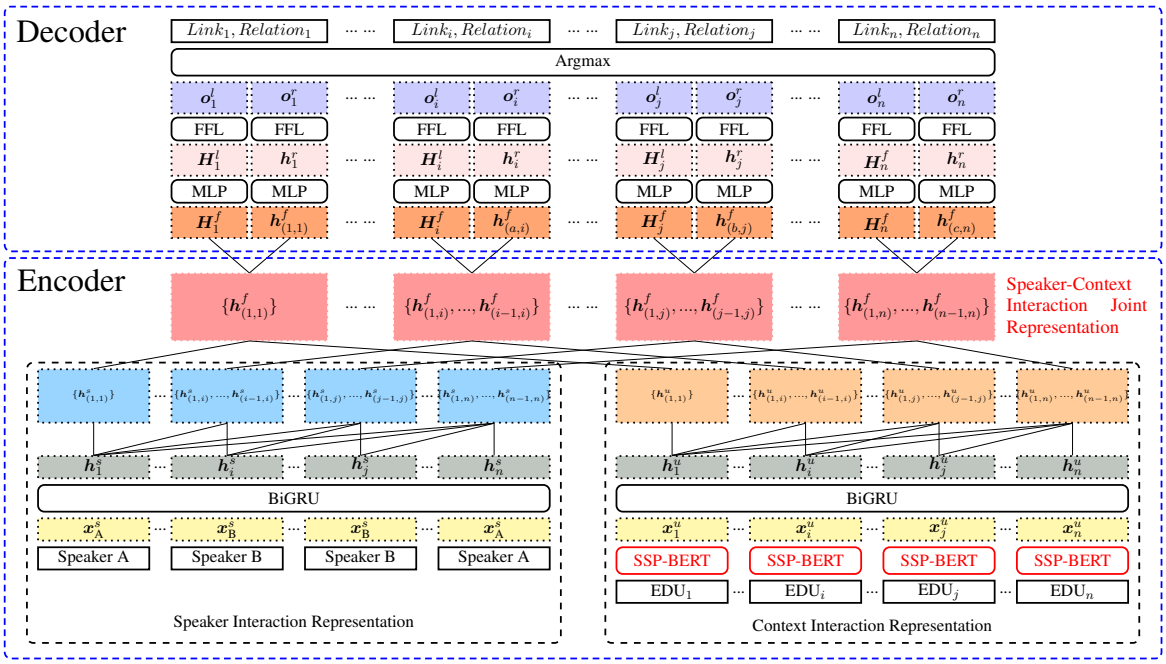

Figure 3: Framework of our proposed speaker-aware discourse parsing model.

The second layer BiGRU is built over sequential EDUs. We should first obtain a suitable representation for each EDU, which is composed of a span of tokens inside a certain EDU. Assuming an EDU u_i with its tokens by $\{[\texttt{CLS}], t_1^i, ..., t_m^i\}$, after applying the first layer BERT, we obtain their representations by $\{\boldsymbol{h}^i_{[\texttt{CLS}]}, \boldsymbol{h}^i_1 ..., \boldsymbol{h}^i_m\}$, then we select the representation of `[CLS]` as the EDU representations $\boldsymbol{x}^u$. When the EDU representations are ready, we apply the BiGRU layer, resulting:

$$\boldsymbol{h}^u_1, ..., \boldsymbol{h}^u_n = \text{BiGRU}(\boldsymbol{x}^u_1, ..., \boldsymbol{x}^u_n) \quad (7)$$

We concatenate $\boldsymbol{h}^u_i$, and $\boldsymbol{h}^u_j$ to obtain the corresponding context interaction representation.

$$\boldsymbol{h}^u_{(j,i)} = \boldsymbol{h}^u_j \oplus \boldsymbol{h}^u_i \quad (8)$$

Speaker-Context Interaction Joint Encoding When the speaker interaction and the context interaction representations are ready, we combine them jointly to obtain the speaker-context interaction joint representations.

$$\boldsymbol{h}^f_{(j,i)} = \alpha \boldsymbol{h}^s_{(j,i)} + (1-\alpha)\boldsymbol{h}^u_{(j,i)} \quad (9)$$

where the α is a learnable parameter, $\boldsymbol{h}^f_{(j,i)}$ denotes the speaker-context interaction joint representation.

3.2.2 Decoder

The decoder performs the link prediction and the relation classification. Concretely, given two EDUs u_i and u_j $(j < i)$, the link prediction task predicts whether u_j is the parent node of u_i. If u_j is the parent node of u_i, the relation classification task would further predicts the discourse relation type between u_i and u_j.

Link Prediction As mentioned before, the encoder represents u_i and u_j to corresponding speaker-context interaction joint representations. We gather the sequence of input representations of $\{(u_1, u_i), ..., (u_{(i-1)}, u_i)\}$, and thus apply a multi-layer perceptron (MLP) layer to obtain link hidden representations as inputs of the link prediction task:

$$\begin{aligned} \boldsymbol{H}_i &= \boldsymbol{h}^f_{(1,i)}, ..., \boldsymbol{h}^f_{(i-1,i)} \\ \boldsymbol{H}^l &= \tanh\left(\boldsymbol{W}^l_2 \tanh\left(\boldsymbol{W}^l_1 \boldsymbol{H}_i + \boldsymbol{b}^l_1\right) + \boldsymbol{b}^l_2\right) \end{aligned} \quad (10)$$

where $\boldsymbol{W}^l_1$, $\boldsymbol{W}^l_2$, $\boldsymbol{b}^l_1$, and $\boldsymbol{b}^l_2$ are model parameters, "tanh" is an activation function, $\boldsymbol{H}^l$ denotes the link hidden representations. Then we apply a feed-forward layer (FFL) to obtain the parent EDU scores:

$$\boldsymbol{o}^l = \boldsymbol{U}^l \boldsymbol{H}^l \quad (11)$$

where $\boldsymbol{o}^l$ is the parent EDU scores and $\boldsymbol{U}^l$ is a model parameter.

Relation Classification We also apply a MLP layer to obtain relation hidden representations:

$$\boldsymbol{h}^r = \tanh\left(\boldsymbol{W}_2^r \tanh\left(\boldsymbol{W}_1^r \boldsymbol{h}_{(a,i)}^f + \boldsymbol{b}_1^r\right) + \boldsymbol{b}_2^r\right) \tag{12}$$

where $\boldsymbol{W}_1^r$, $\boldsymbol{W}_2^r$, $\boldsymbol{b}_1^r$, and $\boldsymbol{b}_2^r$ are model parameters, $\boldsymbol{h}^r$ denotes the relation hidden representation. We also apply a FFL to obtain discourse relation scores:

$$\boldsymbol{o}^r = \boldsymbol{U}^r \boldsymbol{h}^r \tag{13}$$

where $\boldsymbol{o}^r$ is the discourse relation scores and $\boldsymbol{U}^r$ is a model parameter.

3.3 Training

Following previous studies (Shi and Huang, 2019; Wang et al., 2021), we use cross-entropy as the optimization objectives of the link prediction and the relation classification tasks. We add these two objective terms together as the final optimization objective of our discourse parser:

$$\mathcal{L}(\Theta) = -[\log\left(p_{u_g}\right) + \log\left(p_{r_g}\right)] \tag{14}$$

where p_{u_g} and p_{r_g} are probabilities of the gold parent EDU and the gold discourse relation, respectively. Θ is the set of model parameters of our discourse parser.

Given an EDU u_i, its gold parent EDU u_g, and gold discourse relation r_g, we first calculate the link and the relation outputs using Equation 11 and 13, respectively, and then apply softmax to obtain the gold parent probability $p_{u_g} = \frac{\exp\left(\boldsymbol{o}_{u_g}^l\right)}{\sum_1^j \exp(\boldsymbol{o}_{u_k}^l)}$, and the gold relation probability $p_{r_g} = \frac{\exp\left(\boldsymbol{o}_{r_g}^r\right)}{\sum_1^q \exp\left(\boldsymbol{o}_{r_k}^r\right)}$.

4 Experiment Settings

Data We evaluate our proposed model on STAC[2] (Asher et al., 2016) and Molweni[3] (Li et al., 2020). STAC has annotated 1,173 dialogues, where 1,062 for training and the remaining 111 dialogues for testing. All dialogues are collected from an online game trading corpus. To facilitate parameter tuning, we randomly select 10% of the training dialogues as a development corpus. Molweni has annotated 10,000 dialogues, where 9,000 for training, 500 for development, and the remaining 500 dialogues for testing, respectively. All dialogues are collected from the Ubuntu dialogue corpus (Lowe et al., 2015). For fair comparison, we preprocess two datasets following Shi and Huang (2019), and all experiments are conducted based on manually segmented EDUs.

We pre-train BERT on a large-scale unlabeled dialogue corpus in the second stage. It is collected from the Ubuntu dialogue corpus (Lowe et al., 2015), containing 930,000 unlabeled dialogues.

Evaluation We adopt two standard metrics to evaluate our proposed model, including `Link` and `Link&Rel` metrics. The `Link` metric evaluates the capability of link prediction only, and the `Link&Rel` metric evaluates link prediction together with discourse relations. We follow Shi and Huang (2019), reporting the micro F_1 scores.

Hyper-Parameters There are several hyper-parameters in our proposed speaker-aware discourse parsing model.

In the SSP-BERT model, we use *PyTorch* (Paszke et al., 2019) to implement our neural modules, and BERT is implemented by *Transformers* (Wolf et al., 2020). We use *bert-base-uncased*[4] to initialize the model parameters of BERT, and other model parameters are initialized randomly. We optimize model parameters by the Adam algorithm (Kingma and Ba, 2015). The learning rate of BERT is set to 1e-7 and the learning rate of the linear layer is set to 1e-3. We train our SSP-BERT by online learning with mini-batch, and the batch size is set to 8. Several key hyper-parameters are set according to the development experiments in Section 5. We randomly sample 100,000 dialogues for the SSP task with 4 epochs on Molweni, and 100,000 dialogues with 5 epochs on STAC.

In the discourse parsing model, most of hyper-parameters are same on STAC and Molweni. The hidden size of the BiGRU layer is set by 250, and the hidden size of the MLP layer is set by 1,000. The batch size is set to 8, and the maximum training interaction is set to 5. The learning rate of BERT is set differently on STAC and Molweni, 1e-5 and 2e-5 respectively. The learning rate of BiGRU, MLP, and FFL is set to 1e-3.

[2] https://www.irit.fr/STAC/corpus.html
[3] https://github.com/HIT-SCIR/Molweni
[4] https://huggingface.co/bert-base-uncased

Inputs	Dev		Test	
	Link	Link&Rel	Link	Link&Rel
Molweni				
EDUs	79.5	57.8	77.8	56.5
Texts	77.2	56.5	77.0	55.5
STAC				
EDUs	71.4	52.2	72.4	55.4
Texts	71.1	52.4	72.1	54.4

Table 1: Influence of different input methods of BERT.

5 Development Experiments

In this section, we conduct development experiments to examine the effectiveness of some important factors on our proposed model.

Input Methods First, we investigate the influence of different input methods of BERT. There are two different methods to encode the dialogues with BERT. The first method inputs an EDU sequence into BERT, encoding each EDU independently, It is widely used in previous studies (Shi and Huang, 2019; Liu and Chen, 2021; Yang et al., 2021). The second method treats a dialogue as a whole text, and feeds it into BERT to obtain corresponding EDU representations (He et al., 2021). Table 1 shows the comparisons. We can see that using EDUs as inputs is better than using whole texts.

Pre-Trained Language Models Then we examine how different PLMs influence the performance of our proposed model. It is believed that pre-trained language models (PLMs) are promising for discourse parsing on multi-party dialogues (Wang et al., 2021; Liu and Chen, 2021; Yang et al., 2021). As mentioned before, we use BERT to represent conversational contexts. The BERT layer can be replaced by other PLMs, such as BERT (Devlin et al., 2019), RoBERTa (Liu et al., 2019), ELECTRA (Clark et al., 2020), and XLM-R (Conneau et al., 2020). Table 2 shows the development results. When we use SSP to enhance these PLMs, these discourse parsing models are able to obtain further improvements. We find that the SSP-BERT discourse parsing model achieves the best performance among these models on two development sets. Thus we use SSP-BERT in our subsequent experiments.

BiGRU vs Transformer As mentioned before, we use BiGRU to obatin EDU representations in Equation 7. Exploiting transformer (**?**) is an alternative method for obtaining EDU representation, and it may capture the longer dependence in an EDU sequence than BiGRU. Here we further investigate the influence of different EDU representations based on the BiGRU and transformer models. As shown in Table 3, we find that the BiGRU models outperform the transformer models. It may due to that the turn of dialogues in two corpora is short, and BiGRU is enough for capturing the long dependence in these dialogues.

Models	Dev		Test	
	Link	Link&Rel	Link	Link&Rel
Molweni				
BERT	79.5	57.8	77.8	56.5
ELECTRA	79.9	57.7	77.3	55.5
RoBERTa	79.9	57.3	77.4	55.2
XLM-R	79.8	57.8	76.7	54.5
SSP-B	81.6	59.1	79.1	57.7
SSP-E	80.5	58.4	78.1	55.8
SSP-R	80.3	59.0	78.9	57.0
SSP-X	80.2	58.9	78.3	56.7
STAC				
BERT	71.4	52.2	72.4	55.4
ELECTRA	70.7	50.3	72.5	55.4
RoBERTa	71.2	50.7	71.8	54.6
XLM-R	70.1	51.0	71.3	54.1
SSP-B	70.0	52.9	72.6	57.0
SSP-E	71.6	51.9	71.8	55.5
SSP-R	71.3	51.1	71.5	55.5
SSP-X	70.8	51.6	72.2	54.1

Table 2: Effect of different PLMs. "SSP-B", "SSP-E", "SSP-R", and "SSP-X" refer to "BERT", "ELECTRA", "RoBERTa", and "XLM-R" with SSP, respectively.

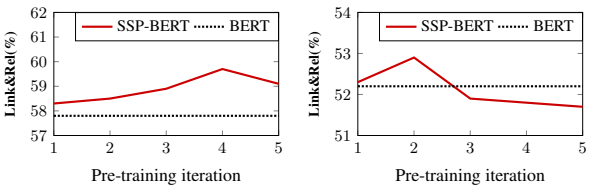

(a) Molweni-Dev (b) STAC-Dev

Figure 4: Influence of pre-training iteration.

Pre-Training Iteration Here we investigate the influence of training iteration in second-stage pre-training. Figure 4 shows the development performances with respect to the training iteration. On Molweni, the performance has been improving when the iteration increases from 1 to 4. However the performance does not improve when the iteration exceeds 4. The experiment over STAC shows a similar trend but the critical iteration is 2. Thus we use iteration 4 and 2 for the subsequent experiments on Molweni and STAC, respectively.

Unlabeled Dialogue Size We also study the influence of the size of unlabeled dialogues in second-stage pre-training. As shown in Figure 5, the

Models	Dev		Test	
	Link	Link&Rel	Link	Link&Rel
Molweni				
BiGRU	81.6	59.1	79.1	57.7
Transformer	80.0	57.8	78.0	56.2
STAC				
BiGRU	70.0	52.9	72.6	57.0
Transformer	71.2	52.4	70.7	53.9

Table 3: Influence of different EDU representations.

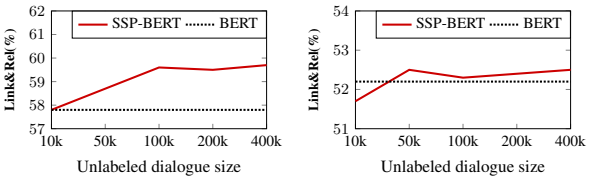

(a) Molweni-Dev (b) STAC-Dev

Figure 5: Influence of unlabeled dialogue size.

Models	Dev		Test	
	Link	Link&Rel	Link	Link&Rel
Molweni				
BERT	79.5	57.8	77.8	56.5
+STs	81.8	59.2	79.6	57.6
+GNN	84.3	59.8	83.0	58.9
+SCIJEC	83.3	59.2	82.6	58.3
+SCIJE	82.9	59.9	83.3	59.4
STAC				
BERT	71.4	52.2	72.4	55.4
+STs	71.2	52.7	72.4	56.4
+GNN	71.6	51.9	72.7	55.8
+SCIJEC	71.2	52.0	71.4	54.9
+SCIJE	72.8	53.0	73.1	56.1

Table 4: Influence of different speaker interaction representation integration methods.

Models	Link	Link&Rel
Molweni		
Li et al. (2020)	78.1	54.8
Wang et al. (2021)	81.6	58.5
Liu and Chen (2021)	80.2	56.9
He et al. (2021)*	80.0	57.0
BERT	77.8	56.5
SSP-BERT + SCIJE	**83.7**	**59.4**
STAC		
Shi and Huang (2019)	73.2	55.7
Wang et al. (2021)	73.5	57.3
Yang et al. (2021)	74.1	57.0
Liu and Chen (2021)	**75.5**	57.2
BERT	72.4	55.4
SSP-BERT + SCIJE	73.0	**57.4**

Table 5: Main results on two test sets. "*" means that we report the performance by rerunning their model.

`Link&Rel` F-measure of our discourse parsing model increases apparently, when the size increases from 100k to 400k, and more unlabeled dialogues does not bring significant improvements. Thus we use 400k dialogues in second-stage pre-training.

Speaker-Context Joint Representation There are several choices for integrating speaker and context interaction representations. Here we compare three approaches with our SCIJE approach. The first approach is simple, which adds speaker tags (STs) with conversational texts as the concatenated texts and uses PLMs to model speaker interaction. In the second approach, we use a graph neural network (GNN) (Wang et al., 2021) to model speaker interaction. In the third approach, we use concatenation to replace the Equation 9. Table 4 shows the results. First, we find that the speaker interaction information is effective for discourse parsing on multi-party dialogues, which is consistent with previous observations (Afantenos et al., 2015; Shi and Huang, 2019; Wang et al., 2021). Second, the SCIJE approach is slightly better than applying GNN. Furthermore, SCIJE can achieve the best performance using a learnable model parameter, better than using concatenation (SCIJEC).

6 Main Results and Analysis

Main Results Here we report the final results of the proposed model over the Molweni and the STAC test sets. As shown in Table 5, our discourse parsing model achieves a `Link` F-measure of 77.8 and a `Link&Rel` F-measure of 56.5 on the Molweni test set, and a `Link` F-measure of 72.4 and a `Link&Rel` F-measure of 55.4 on the STAC test set. We find that the performance of our discourse parsing model on the Molweni test set outperforms most performances of previous SOTA systems. When both SSP-BERT and SCIJE are adopted, our final model achieves a `Link` F-measure of 83.7 and a `Link&Rel` F-measure of 59.4 in the Molweni test set, resulting improvements 83.7 - 77.8 = 5.9 on `Link` and 59.4 - 56.5 = 2.9 on `Link&Rel`. On STAC, our final model achieves a `Link` F-measure of 73.0 and a `Link&Rel` F-measure of 57.4, resulting improvements 73.0 - 72.4 = 0.6 on `Link` and 57.4 - 55.4 = 2.0 on `Link&Rel`.

We compare our final model with previous SOTA systems as well. Shi and Huang (2019) propose a deep sequential discourse parsing model, using local information of EDUs and global information of predicted discourse structures. Yang et al. (2021) propose a joint model for discourse parsing and dropped pronoun recovery. Liu and Chen (2021) propose a domain information enhanced dis-

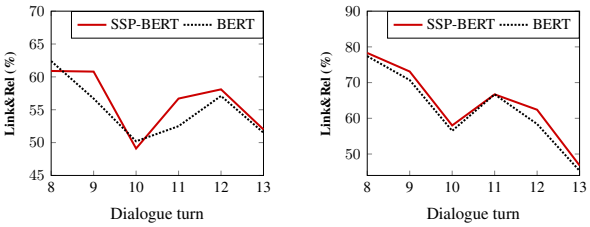

(a) Molweni (b) STAC

Figure 6: `Link&Rel` against dialogue length.

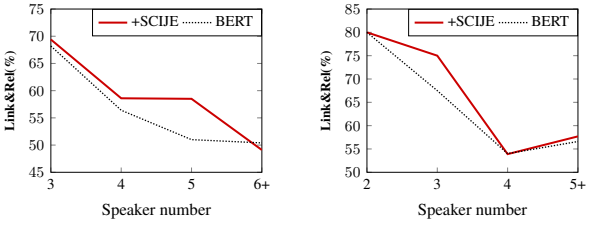

(a) Molweni (b) STAC

Figure 7: `Link&Rel` against speaker number.

course parsing model. He et al. (2021)[5] propose a multi-task framework for performing discourse parsing and dialogue comprehension jointly. As shown in Table 5, we find that our proposed model achieves the SOTA performances on two benchmark datasets.

Ablation Studies Here we investigate our proposed model by ablation studies. Table 6 shows the results of ablation studies on two test sets. On Molweni, both SCIJE and SSP-BERT are effective for this task. Without SCIJE, the `Link&Rel` F-measure decreases by close to 1.7%. Without SSP-BERT, the `Link` F-measure decreases by close to 0.4%. On STAC, the results have the same tendencies. Without SCIJE, the `Link&Rel` F-measure decreases by close to 0.4%. Without SSP-BERT, the `Link&Rel` F-measure decreases by close to 1.3%. Based on above results, we find that our proposed speaker-aware model are more effective on Molweni. It may be due to that the dialogues in Molweni involve more different speakers.

Influence of Dialogue Turn As mentioned before, the SSP task predicts whether two EDUs are from the same speaker. It is able to integrate the speaker interaction features between the same

[5]It should be noted that Molweni contains two datasets, one for dialogue comprehension (100 dialogues) and other for discourse parsing (500 dialogues). He et al. (2021) only report their results on dialogue comprehension test set. For fair comparison, here we rerun their model on the discourse parsing test data .

Models	Link	Link&Rel
Molweni		
SSP-BERT + SCIJE	83.7	59.4
SSP-BERT	79.1	57.7
BERT + SCIJE	83.4	59.4
BERT	77.8	56.5
STAC		
SSP-BERT + SCIJE	73.0	57.4
SSP-BERT	72.6	57.0
BERT + SCIJE	73.1	56.1
BERT	72.4	55.4

Table 6: Ablation study on two test sets.

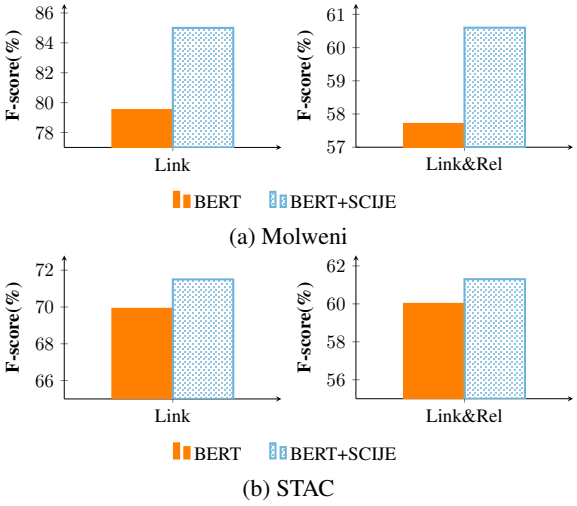

(a) Molweni

(b) STAC

Figure 8: Influence of SCIJE on connected EDU pairs from different speakers.

speaker into BERT. Therefore, it is expected that the introduce of SSP-BERT may bring better performance for longer dialogues. As such, here we investigate the discourse parsing model with SSP-BERT by the capability of modeling dialogue turns. Figure 6a shows the results on Molweni. The discourse parser with SSP-BERT performs better when dialogue lengths are 9, 11, and 13. It performs slightly worse when the dialogue lengths are 8 and 10. The tendency is different on STAC. As shown in Figure 6b, the discourse parser with SSP-BERT consistently outperforms the original parser for dialogues of different lengths.

Influence of Speaker Number As mentioned before, our speaker-aware model exploits a SCIJE approach to encode the speaker and the context interactions of dialogues. We believe that it is able to integrate the different speakers interaction information into the discourse parsing model. Therefore, it is expected that exploiting SCIJE may bring better performance for multi-party dialogues with more speakers. As such, here we plot `Link&Rel`

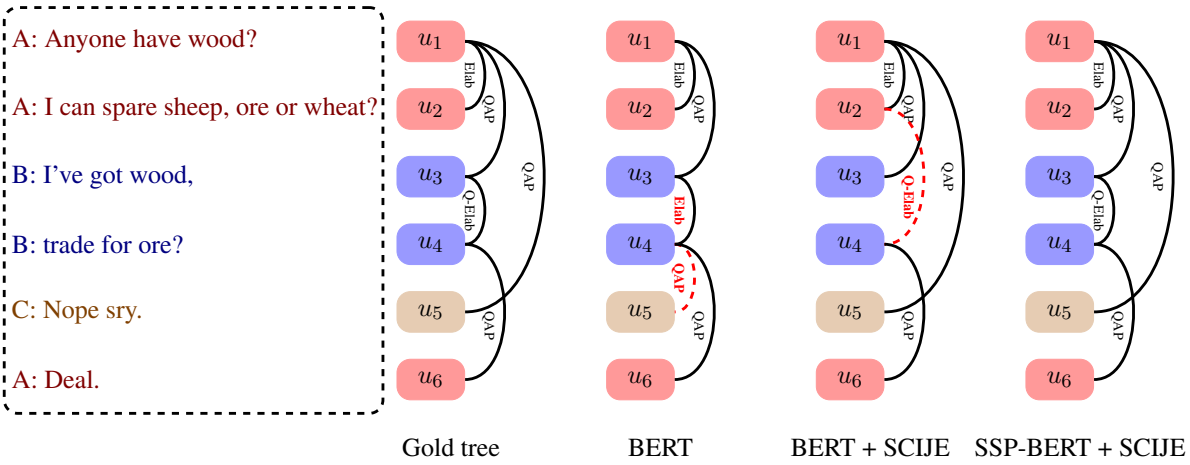

Figure 9: Case studies of the proposed speaker-aware discourse parsing model.

F-measures with respect to speaker number of dialogues. As shown in Figure 7a, we find that the discourse parsing model with speaker-context interaction joint representations performs better on dialogues with 3 to 5 speakers. The tendency is different on STAC. As shown in Figure 7b, the discourse parsing model with speaker-context interaction joint representations performs better apparently when the speaker number is 3. It may be due to that the dialogues in STAC have less connected EDU pairs from different speakers.

EDU Pairs from Different Speakers Furthermore, we investigate the performances in the connected EDU pairs from different speakers. We filter the connected EDU pairs from the same speaker, and only investigate the performances with respect to the connected EDU pairs from different speakers. As shown in Figure 8, we find that the BERT discourse parsing with SCIJE performs better for the EDU pairs from different speakers on both Molweni and STAC. The findings indicate that SCIJE could integrate the interaction information from different speakers to discourse parsing model.

Case Studies Here we present several case studies to demonstrate the advantages of the proposed speaker-aware discourse parsing model. As shown in Figure 9, the first tree is the gold tree of the dialogue, and other predicted trees are provided by the our proposed models. We find that the BERT-based parser is incapable of handling the arc from different speakers (i.e. u_1 and u_5) and the relation from the same speakers (i.e. u_3 and u_4). In the third tree, we show how the BERT-based parser benefits from SCIJE. We find that the BERT-based parser with SCIJE correctly recognizes the arc between u_1 and u_5, as SCIJE integrate the different speakers interaction information for discourse parsing. In the forth tree, we show how SSP further enhance the proposed model. We find that the final model corresponding recognizes the relation between u_3 and u_4, as SSP offers the same speaker interaction information for this task.

7 Conclusion

In this paper, we proposed a speaker-aware model for discourse parsing on multi-party dialogues, It is able to better model the speaker interactions for this task. First, we proposed SCIJE to incorporate the interaction features between the different speakers. Second, we integrated the interaction features between the same speaker to the conversational context representations by exploiting SSP-BERT. We conducted experiments and analysis on two standard benchmark datasets, namely STAC (Afantenos et al., 2015) and Molweni (Li et al., 2020). Results show that our proposed speaker-aware discourse parsing model significantly outperforms previous SOTA systems in the literature.

Acknowledgments

We thank the anonymous reviewers for their valuable comments. This work was supported by the National Natural Science Foundation of China (No. 62076173, U1836222), the High-level Entrepreneurship and Innovation Plan of Jiangsu Province (No. JSSCRC2021524), and the Project Funded by the Priority Academic Program Development of Jiangsu Higher Education Institutions.

References

Stergos Afantenos, Eric Kow, Nicholas Asher, and Jérémy Perret. 2015. Discourse parsing for multi-party chat dialogues. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 928–937, Lisbon, Portugal. Association for Computational Linguistics.

Nicholas Asher, Nicholas Michael Asher, and Alex Lascarides. 2003. *Logics of Conversation*. Cambridge University Press.

Nicholas Asher, Julie Hunter, Mathieu Morey, Benamara Farah, and Stergos Afantenos. 2016. Discourse Structure and Dialogue Acts in Multiparty Dialogue: the STAC Corpus. In *Proceedings of the Tenth International Conference on Language Resources and Evaluation (LREC'16)*, pages 2721–2727, Portorož, Slovenia. European Language Resources Association (ELRA).

Chloé Braud, Maximin Coavoux, and Anders Søgaard. 2017. Cross-lingual RST Discourse Parsing. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 1, Long Papers*, pages 292–304, Valencia, Spain. Association for Computational Linguistics.

Chloé Braud, Barbara Plank, and Anders Søgaard. 2016. Multi-view and multi-task training of RST discourse parsers. In *Conference on Computational Linguistics (CoLing)*, pages 1903 – 1913, Osaka, Japan.

Kevin Clark, Minh-Thang Luong, Quoc V. Le, and Christopher D. Manning. 2020. ELECTRA: Pre-training Text Encoders as Discriminators Rather Than Generators. *arXiv:2003.10555 [cs]*. ArXiv: 2003.10555.

Alexis Conneau, Kartikay Khandelwal, Naman Goyal, Vishrav Chaudhary, Guillaume Wenzek, Francisco Guzmán, Edouard Grave, Myle Ott, Luke Zettlemoyer, and Veselin Stoyanov. 2020. Unsupervised Cross-lingual Representation Learning at Scale. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 8440–8451, Online. Association for Computational Linguistics.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, Minneapolis, Minnesota. Association for Computational Linguistics.

Yuchen He, Zhuosheng Zhang, and Hai Zhao. 2021. Multi-tasking Dialogue Comprehension with Discourse Parsing.

Matthew Henderson, Iñigo Casanueva, Nikola Mrkšić, Pei-Hao Su, Tsung-Hsien Wen, and Ivan Vulić. 2020. ConveRT: Efficient and Accurate Conversational Representations from Transformers. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 2161–2174, Online. Association for Computational Linguistics.

Diederik P. Kingma and Jimmy Ba. 2015. Adam: A Method for Stochastic Optimization. *ICLR*.

Jiaqi Li, Ming Liu, Min-Yen Kan, Zihao Zheng, Zekun Wang, Wenqiang Lei, Ting Liu, and Bing Qin. 2020. Molweni: A Challenge Multiparty Dialogues-based Machine Reading Comprehension Dataset with Discourse Structure. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 2642–2652, Barcelona, Spain (Online). International Committee on Computational Linguistics.

Jiwei Li, Rumeng Li, and Eduard Hovy. 2014. Recursive Deep Models for Discourse Parsing. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 2061–2069, Doha, Qatar. Association for Computational Linguistics.

Ziheng Lin, Hwee Tou Ng, and Min-Yen Kan. 2014. A PDTB-styled end-to-end discourse parser. *Natural Language Engineering*, 20(2):151–184.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. RoBERTa: A Robustly Optimized BERT Pretraining Approach. Technical report. Publication Title: arXiv e-prints ADS Bibcode: 2019arXiv190711692L Type: article.

Zhengyuan Liu and Nancy Chen. 2021. Improving Multi-Party Dialogue Discourse Parsing via Domain Integration. In *Proceedings of the 2nd Workshop on Computational Approaches to Discourse*, pages 122–127, Punta Cana, Dominican Republic and Online. Association for Computational Linguistics.

Ryan Lowe, Nissan Pow, Iulian Serban, and Joelle Pineau. 2015. The Ubuntu Dialogue Corpus: A Large Dataset for Research in Unstructured Multi-Turn Dialogue Systems. In *Proceedings of the 16th Annual Meeting of the Special Interest Group on Discourse and Dialogue*, pages 285–294, Prague, Czech Republic. Association for Computational Linguistics.

Xinbei Ma, Zhuosheng Zhang, and Hai Zhao. 2021. Enhanced Speaker-aware Multi-party Multi-turn Dialogue Comprehension.

Xinbei Ma, Zhuosheng Zhang, and Hai Zhao. 2022. Structural Characterization for Dialogue Disentanglement. *arXiv:2110.08018 [cs]*. ArXiv: 2110.08018.

William C. Mann and Sandra A. Thompson. 1988. Rhetorical Structure Theory: Toward a functional theory of text organization. *Text - Interdisciplinary Journal for the Study of Discourse*, 8(3):243–281. Publisher: De Gruyter Mouton Section: Text & Talk.

Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Kopf, Edward Yang, Zachary DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. 2019. PyTorch: An Imperative Style, High-Performance Deep Learning Library. In *Advances in Neural Information Processing Systems*, volume 32. Curran Associates, Inc.

Rashmi Prasad, Nikhil Dinesh, Alan Lee, Eleni Miltsakaki, Livio Robaldo, Aravind Joshi, and Bonnie Webber. 2008. The Penn Discourse TreeBank 2.0. In *In Proceedings of the 6th Language Resources and Evaluation Conference (LREC*.

Zhouxing Shi and Minlie Huang. 2019. A Deep Sequential Model for Discourse Parsing on Multi-Party Dialogues. *Proceedings of the AAAI Conference on Artificial Intelligence*, 33:7007–7014.

Ante Wang, Linfeng Song, Hui Jiang, Shaopeng Lai, Junfeng Yao, Min Zhang, and Jinsong Su. 2021. A Structure Self-Aware Model for Discourse Parsing on Multi-Party Dialogues. In *Proceedings of the Thirtieth International Joint Conference on Artificial Intelligence*, pages 3943–3949, Montreal, Canada. International Joint Conferences on Artificial Intelligence Organization.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Remi Louf, Morgan Funtowicz, Joe Davison, Sam Shleifer, Patrick von Platen, Clara Ma, Yacine Jernite, Julien Plu, Canwen Xu, Teven Le Scao, Sylvain Gugger, Mariama Drame, Quentin Lhoest, and Alexander Rush. 2020. Transformers: State-of-the-Art Natural Language Processing. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 38–45, Online. Association for Computational Linguistics.

Ruijian Xu, Chongyang Tao, Daxin Jiang, Xueliang Zhao, Dongyan Zhao, and Rui Yan. 2021. Learning an Effective Context-Response Matching Model with Self-Supervised Tasks for Retrieval-based Dialogues. *Proceedings of the AAAI Conference on Artificial Intelligence*, 35(16):14158–14166. Number: 16.

Jingxuan Yang, Kerui Xu, Jun Xu, Si Li, Sheng Gao, Jun Guo, Nianwen Xue, and Ji-Rong Wen. 2021. A Joint Model for Dropped Pronoun Recovery and Conversational Discourse Parsing in Chinese Conversational Speech. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 1752–1763, Online. Association for Computational Linguistics.

Ze Yang, Liran Wang, Zhoujin Tian, Wei Wu, and Zhoujun Li. 2022. TANet: Thread-Aware Pretraining for Abstractive Conversational Summarization. *arXiv:2204.04504 [cs]*. ArXiv: 2204.04504.

Nan Yu, Meishan Zhang, and Guohong Fu. 2018. Transition-based Neural RST Parsing with Implicit Syntax Features. In *Proceedings of the 27th International Conference on Computational Linguistics*, pages 559–570, Santa Fe, New Mexico, USA. Association for Computational Linguistics.

Nan Yu, Meishan Zhang, Guohong Fu, and Min Zhang. 2022. RST Discourse Parsing with Second-Stage EDU-Level Pre-training. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 4269–4280, Dublin, Ireland. Association for Computational Linguistics.

Yizhe Zhang, Siqi Sun, Michel Galley, Yen-Chun Chen, Chris Brockett, Xiang Gao, Jianfeng Gao, Jingjing Liu, and Bill Dolan. 2020. DialoGPT: Large-Scale Generative Pre-training for Conversational Response Generation. *arXiv:1911.00536 [cs]*. ArXiv: 1911.00536.

Zhuosheng Zhang and Hai Zhao. 2021. Structural Pre-training for Dialogue Comprehension. *arXiv:2105.10956 [cs]*. ArXiv: 2105.10956.

Iterative Span Selection: Self-Emergence of Resolving Orders in Semantic Role Labeling

Shuhei Kurita[1,2] **Hiroki Ouchi**[3,1] **Kentaro Inui**[4,1] **Satoshi Sekine**[1]
[1]RIKEN [2]JST PRESTO [3]Nara Institute of Science and Technology [4]Tohoku University
shuhei.kurita@riken.jp hiroki.ouchi@is.naist.jp
inui@tohoku.ac.jp satoshi.sekine@riken.jp

Abstract

Semantic Role Labeling (SRL) is the task of labeling semantic arguments for marked semantic predicates. Semantic arguments and their predicates are related in various distinct manners, of which certain semantic arguments are a necessity while others serve as an auxiliary to their predicates. To consider such roles and relations of the arguments in the labeling order, we introduce iterative argument identification (IAI), which combines global decoding and iterative identification for the semantic arguments. In experiments, we first realize that the model with random argument labeling orders outperforms other heuristic orders such as the conventional left-to-right labeling order. Combined with simple reinforcement learning, the proposed model spontaneously learns the optimized labeling orders that are different from existing heuristic orders. The proposed model with the IAI algorithm achieves competitive or outperforming results from the existing models in the standard benchmark datasets of span-based SRL: CoNLL-2005 and CoNLL-2012.

1 Introduction

Semantic role labeling (Carreras and Màrquez, 2004, 2005) is the task of identifying and resolving the relations between semantic predicates and their arguments based on the PropBank (Kingsbury and Palmer, 2002) predicate-argument structure. In span-based SRL, semantic predicates comprise several semantic arguments that are expressed as spans of tokens in the sentence. Recent span-based SRL models incorporate neural networks into a global decoding approach. Here syntactic features are injected into the neural network model (Strubell et al., 2018), span-based scoring for semantic argument is adapted (Ouchi et al., 2018), and the unified representations for both span-based and dependency-based SRL are applied (Li et al., 2019; Zhou et al., 2020). However, in these approaches, the labeling order is not determined inside the neural network and hence models require some extrapolated graph decoding procedures, similar to the graph-based approaches that rely on external graph decoding (Lewis et al., 2015). Such external decoding procedures are typically not trained during the model training and hence hinder accurate decoding.

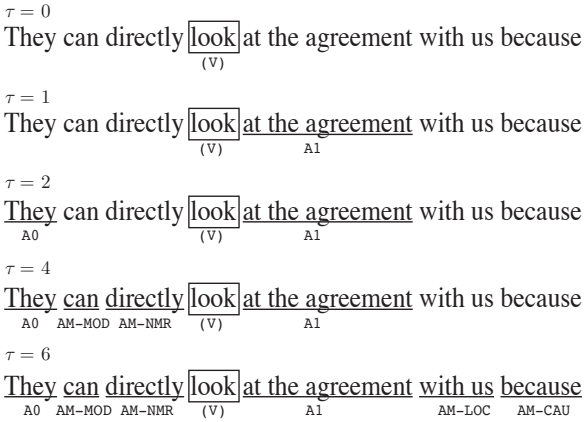

Figure 1: Example of the iterative argument identification with the given predicate "look". At the bottom of the final state $\tau = 6$, all semantic arguments are presented with spans and labels. The time step τ corresponds the iterative process of the proposed model.

The sequential labeling approach is another major branch of span-based SRL models (Màrquez et al., 2005; Zhou and Xu, 2015; He et al., 2017; Tan et al., 2018; Li et al., 2020), wherein the models resolve sentences from the beginning to the end or *left-to-right* ordering by attaching labels that represent both semantic spans and roles. Li et al. (2020) proposed the BIO labeling-based model with predefined regularizers of unique case roles, exclusively overlapping roles and PropBank frame definitions. However, sequential labeling approaches often suffer from the error-propagation problem(Senge et al., 2014; Dinarelli and Tellier, 2018). One reason is that they are not able to arrange the argument identification orderings in decoding.

Thus, in this study, we explore an SRL model

Proceedings of the 29th International Conference on Computational Linguistics, pages 5383–5397
October 12–17, 2022.

that combines global decoding and iterative labeling approaches: iterative argument identification (IAI). Our model works iteratively: the model identifies one argument individually and stores it for each time step. The stored semantic arguments are used as "clues" for identifying other arguments in later time steps. Moreover, our models can identify semantic arguments from arbitrary orders because our model identifies arguments from any part of the sentence. This means that our model can consider relations of predicates and arguments in the decoding order. In SRL, semantic arguments have various roles to their predicates. Figure 1 represents an example of attached semantic role labels for the partial sentence of "They can directly look at the agreement with us because...". In this example, both arguments "They" and "at the agreement" represent crucial semantic roles to their predicate "look." However, other arguments such as "with us" and the phrase following "because" represent additional information to their predicate.

As identified arguments become clues in later processes, choosing suitable decoding orderings affects the final performance of the proposed model because many clues become available to identify a new argument in later time steps. Here we ask the following question: Are there certain labeling orderings that can identify arguments more accurately than heuristic orderings? Empirical experiments revealed that the traditional *left-to-right* ordering although strong, is not the best ordering, e.g., simple *random* ordering in imitation learning outperform the *left-to-right* ordering. Based on the results obtained, we explored models that follow better decoding orders than the heuristic orders. We assume optimal transition paths are not generated with heuristics or hand-engineering, and rather expect the self-emergence of the optimal transition paths that are different from the existing heuristic transition paths through the model training. We applied simple policy-gradient-based reinforcement-learning for the IAI model and found that reinforcement learning slightly leverages the model performance thereby allowing models to arrange orderings resulting in different argument orderings from existing heuristics, which was confirmed through several analyses. Furthermore, our model achieved competitive or better performances than the existing models in the standard benchmark datasets.[1]

[1]The code is available at https://github.com/shuheikurita/iss_srl

2 Related Work

The idea of optimizing the labeling orders in decoding is a branch of the *easy-first* strategy (Tsuruoka and Tsujii, 2005; Goldberg and Elhadad, 2010; Ma et al., 2013; Martins and Kreutzer, 2017). In SRL, Wolfe et al. (2016) proposed the SRL model with the pseudo teacher approaches for the processing orders in SRL. They exploit violation fixing perceptron and their parser explores the states of the highest scored path along with the word frequency ordering baseline. Since their proposed model of "easy-first dynamic" follows the highest scoring action, their model explores limited transition spaces during training. Refinement of existing SRL is also examined in dependency-based SRL (Lyu et al., 2019; Chen et al., 2019). Reinforcement learning is also applied in broad syntactic and semantic parsing studies (Lê and Fokkens, 2017; Fried and Klein, 2018; Naseem et al., 2019; Kurita and Søgaard, 2019). Multi-task neural network is often applied to such structured syntactic analyses (Søgaard and Goldberg, 2016; Kurita et al., 2017). It is notable that adversarial training is also applied to extract knowledge from unannotated corpora in Japanese predicate-argument structure analysis (Kurita et al., 2018).

Lattice-based approach is also a promising approach for SRL in traditional (Täckström et al., 2015) and neural models (FitzGerald et al., 2015). However, they rely on external dynamic programming decoding. Choi and Palmer (2011) proposed the transition-based model for dependency-based SRL. They applied a set of transition actions that are similar to the shift-reduce parser (Nivre, 2008) in syntactic parsing. They also adapted the self-learning clustering technique for predicates that are unseen in training. Blloshmi et al. (2021) address a sequence-to-sequence labeling model which performs competitive with sequence-labeling models. Indeed, most of the recent SRL resolving studies address the global-decoding approach (Ouchi et al., 2018; Li et al., 2018, 2019; Zhou et al., 2020; Conia and Navigli, 2020) or the sequential labeling approach (Shi and Lin, 2019; Li et al., 2020; Marcheggiani and Titov, 2020; Zhang et al., 2021; Kasai et al., 2019) in both span-based and dependency-based SRL. In this paper, we introduce the iterative approach for the global argument selection and enable models to determine the ordering of resolving semantic arguments with modern neural networks and reinforcement learning for span-based SRL.

3 Model

3.1 Iterative argument identification

The span-based SRL model predicts multiple spans of tokens as semantic arguments for each marked semantic predicate and attaches semantic role labels to the arguments. Some semantic arguments have crucial roles in the grammatical or semantic structures of a sentence, whereas other arguments have rather auxiliary roles to their predicates. Such arguments are, therefore, more difficult to resolve than others. In iterative span selection, our model repeatedly predicts one semantic argument for each semantic predicate in a single iteration. The previously predicted semantic arguments are stored in a partial semantic arguments buffer. In later time steps, our model is able to use the information from the previously extracted semantic arguments to effectively predict the remaining arguments.

For a given predicate p, the proposed model determines the next argument boundary and its role label in each iteration. Formally, let X is the input sentence, p is a marked predicate and $\mathbf{Y}_p^g$ be the set of all annotated arguments of the marked predicate p in the annotated data g. One semantic argument $y_{i,p} \in \mathbf{Y}_p^g$ is represented by the span of tokens and its semantic role label as $y_{i,p} = \{t^s, t^e, l\}$. Here, t^s is the beginning of the argument span, t^e is the end of the span, and l is the attached semantic role label. Then, we define a transition action $\mathbf{a}_p$ for each predicate p. The action $\mathbf{a}_p$ includes the decision of whether the predicate p has more unresolved arguments or not, and the detection of a new single semantic argument $y_{i,p} = \{t^s, t^e, l\}$. In each iteration, the model resolve a new semantic role label of $y_{i,p} = \{t^s, t^e, l\}$ by choosing t^s, t^e and l respectively, or decide that the predicate p has no more semantic arguments. When the model predicts semantic arguments for the marked predicates, the resolved arguments are stored in the partial SRL buffer of that predicate. The partial SRL buffer contains the previously predicted arguments $\mathbf{Y}_p^\tau$ for each predicate p in the iteration of the time step τ. The partial SRL buffer is updated after each transition and used as part of the model input in the next step. Note that the transitions are independently performed for each predicate. Therefore, the model can stop transitions for some predicates while the model continues transitions for other predicates. The structure of the partial semantic argument buffer is explained in Section 3.2.2.

3.2 Neural network

Our neural network model predicts the probabilities of the transition action $\mathbf{a}_p$ as $p(\mathbf{a}_p|X, p, \mathbf{Y}_p^\tau)$ for all predicates in each iteration τ. Figure 2 represents the neural network model. The network consists of three parts: (i) the sentence encoder, (ii) the partial SRL encoder, and (iii) the span selection and labeling decoder.

3.2.1 Sentence encoder

For the sentence encoder, we use the self-attention architecture of transformer (Vaswani et al., 2017), which is compatible with the huge pretrained language encoder models, such as BERT (Devlin et al., 2019). Pretrained models often rely on sub-word segmentations while SRL is a token-level task. For a token with multiple sub-tokens, we use the beginning sub-token for the entire representation of the original token. We initially split the input sentence into sub-tokens and add the special tokens of "`[NULL]`", "`[EOS]`" and "`[PAD]`". Here "`[NULL]`" has the special meaning that the predicate has no unresolved arguments. Following the pretrained models, we apply wordpiece (Wu et al., 2016) for the original tokens to obtain sub-words. The phrase of "the amended filings", for example, becomes the sequence of sub-tokens as "`[NULL]` the amended filing #s `[EOS]` `[PAD]` ... `[PAD]`" where the token "filings" are split into two sub-tokens "filing" and "#s".

We apply transformer to encode a sequence of sub-tokens in the sentence to obtain $h(t_i) \in R^d$ for the representation of the i-th sub-token t_i. d is the output dimension of the transformer model. In contrast to the representations of the partial semantic argument buffer, the obtained representations $h(t_i)$ for the sentence are not altered during transitions.

3.2.2 Partial SRL encoder

We employ a special encoder that directly encodes the partially-extracted semantic arguments of $\mathbf{Y}_p^\tau$ that are resolved in the former transitions of the iterative argument identification algorithm. We present the partial SRL buffers $\mathbf{Y}_p^\tau$ which contain the spans of previously extracted semantic arguments in Figure 3. We prepare the same number of the partial SRL buffers with the number of the marked predicates in the sentence. During SRL resolving, partial SRL buffers are updated separately for each predicate. Contents of the partial SRL buffer for a predicate does not affect the arguments identification for other predicates. This nature al-

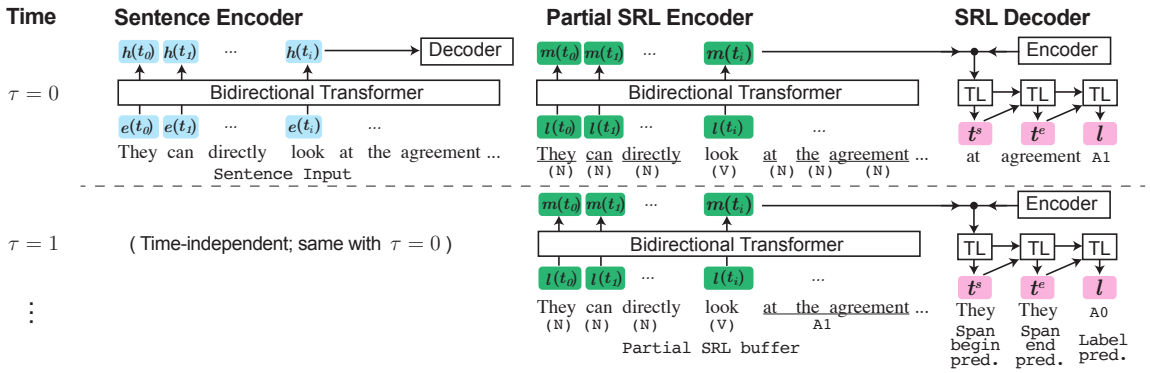

Figure 2: Overall network architecture: the sentence encoder, partial SRL encoder, and SRL decoder. The sentence encoder takes inputs of the (sub)token representation of $e(t_i)$ and computes the time-independent sentence representation of $h(t_i)$, for each sub-token t. The partial SRL encoder takes inputs of the label representation of $l(t_i)$ and computes the time-dependent partial SRL buffer representation of $m(t_i)$. In the partial SRL buffer, (V) represents the marked predicates and (N) represents the token does not have the attached labels yet. We depict the time sequence $\tau = 0$ and $\tau = 1$ cases for the same example with Figure 1.

Time	They	can	directly	look	at	the	agreement
$\tau = 0$	(N)	(N)	(N)	(V)	(N)	(N)	(N)
$\tau = 1$	(N)	(N)	(N)	(V)	A1	A1	A1
$\tau = 2$	A0	(N)	(N)	(V)	A1	A1	A1
$\tau = 3$	A0	(N)	AM-MNR	(V)	A1	A1	A1

Figure 3: Example of the partial SRL buffer updates for the predicate "look" of the phrase "They can directly look at the agreement".

Name	Sent. Enc.	SRL Enc.	SRL Dec.
Hidden size	1024	256	2560
Transformer layers	12	3	3 (Total)

Table 1: Hyper-parameters for transformers of sentence encoder, SRL encoder and the SRL decoder. The other hyperparameters are the same with those of BERT.

lows the parallelization of SRL resolving for each predicate as discussed in the Appendix A.6.

At the initial transition of $\tau = 0$, a partial SRL buffer is filled with "(N)" labels for all sub-tokens except the marked semantic predicate with a label of "(V)". Here "(N)" label represents that the token has no predicted role labels yet for the predicate marked "(V)". At the end of τ-th transition, the model updates the SRL buffers with new predicted semantic arguments that are used for the next transition of $\tau + 1$.

We use a transformer-based encoder for the sequence of extracted SRL spans to obtain the partial semantic argument representations of $m^\tau(t_i) \in \mathcal{R}^d$ of the i-th sub-token t_i. This transformer is different from the sentence encoder transformer and it is not pretrained. As the partial SRL buffers are updated in transitions, the representations for them are altered. Therefore the partial SRL representations contribute to changes in the SRL resolving actions.

3.2.3 SRL decoder

We employ a decoder network for incrementally predicting the beginning and end of the new argument span for each predicate in the sentence. Argument prediction is performed by predicting the beginning token t^s, the span end token t^e and the argument label l. To do so, the decoder network predicts the probabilities of the next action $\mathbf{a}_p$ for each predicate p with the inputs of the sentence, the predicate and partial semantic role label buffer: $p(\mathbf{a}_p|\mathrm{X}, p, \mathbf{Y}_p^\tau)$. The action $\mathbf{a}_p$ consists of three decisions: (i) choosing the beginning of the span t^s or deciding this predicate does not have further semantic arguments e.g., the model selects the "[NULL]" token as t^s, (ii) choosing the end of the span t^e and (iii) attaching a semantic role label l for the predicted span $[t^s, t^e]$ of the argument.

The decoder network works as follows. First, the model concatenates the representations of sub-tokens $\mathbf{h}$ and the partial SRL buffer m^τ for τ-th transition and input it into transformer layers for the scoring tokens as the beginning of the span $s_s(\cdot)$ with a softmax function over sub-tokens

$$p(t^s = t_i) = \frac{\exp\left(s_s([h(t_i), m^\tau(t_i)])\right)}{\sum_{t'} \exp\left(s_s([h(t'), m^\tau(t')])\right)} \quad (1)$$

to obtain the probability $p(t^s)$ for a sub-token t_i to become the beginning of the span t^s. Sub-tokens that are not the beginning of the original tokens don't become t^s. Therefore the probabil-

ity $p(t^s)$ is re-normalized for these beginning sub-tokens while the probabilities of other sub-tokens for t^s are adjusted to 0. In the evaluation, we choose the beginning of the next argument span with $\arg\max_t p(t^s)$. In imitation learning, we choose the teacher label of the beginning of the span t^s_g from the annotated arguments. In reinforcement learning, we choose the teacher labels with Gumbel-Softmax here. In the sampling and evaluation, models are required to consider sub-tokens that are the beginning of some original tokens as the candidates of the next argument beginning t^s.

Similarly, the model predicts the end of the span t^e given the sub-token of the span beginning t^s. Similar to the beginning of the span, We prepare transformer layers for $s_e(\cdot)$ with a softmax function over sub-tokens

$$p(t^e = t_i) = \frac{\exp\left(s_e([h(t_i), m^\tau(t_i), h(t^s)])\right)}{\sum_{t'} \exp\left(s_e([h(t'), m^\tau(t'), h(t^s)])\right)}$$

to obtain the probability $p(t^e)$ for a sub-token t_i as the end of the argument span t^e. Here the model uses representations of the sub-tokens of the argument beginning t^s that are resolved first. To reduce the size of the concatenated vectors of the three representations, we extract the first half values of the three vectors and concatenate them. The model does the sampling from $p(t^e)$ with Gumbel-Softmax to obtain t^e similar to t^s during reinforcement learning. Finally, the model computes the label distribution of the argument from the sentence and the SRL representation $h(t)$ and $m(t)$ as the beginning and end tokens representation of the argument span, $[h(t^s), m(t^s)]$ and $[h(t^e), m(t^e)]$ with a scoring function $s_l(\cdot)$ of another transformer layer:

$$p(l) = \frac{\exp\left(s_l([h(t^s), m^\tau(t^s), h(t^e), m^\tau(t^e)])\right)}{\sum_l \exp\left(s_l([h(t^s), m^\tau(t^s), h(t^e), m^\tau(t^e)])\right)}$$

for the label l prediction.

3.3 Learning

The problem in training IAI is that there are no annotated orders for determining SRLs. In Figure 3, we present an example of transitions for IAI. However, such teacher orders are not always available during training models. For training models, there are two possible approaches: *imitation learning* as the teacher path is given by an oracle and *reinforcement learning* as the model explores the transition paths during training.

3.3.1 Imitation learning

We first define teacher transition paths. Given all annotated arguments $\mathbf{Y}^g_p$ for each predicate, the transition path is the sequence of the annotated arguments $\{y_0, \cdots, y_T\}$. We prepare simple heuristic transition paths: *right-to-left*, *left-to-right*, *close-to-distant*, *distant-to-close* and *random* orders. For each semantic predicate, the left-to-right order teacher selects the annotated semantic arguments from left to right. This is similar behaviour to the transition-based models. The right-to-left order teacher is the inverse of the left-to-right. The close-to-distant and distant-to-close order teachers select arguments based on the distance of sub-tokens from the predicate.[2] These four teacher transition paths always yield the same transition paths, whereas the random transition teacher yields different transition paths in each epoch. Therefore, the random transition benefits from this de-facto data augmentation.[3] We compare the results of those heuristic teachers in imitation learning in Appendix A.4.

3.3.2 Reinforcement learning

In reinforcement learning, the model determines the transition path during training. In particular, we apply a policy gradient to explore the transition space that is uncommon during the imitation training. A Gumbel-Softmax distribution (Jang et al., 2017) has the essential property that it can be smoothly annealed into a categorical distribution. Thus we use Gumbel-Softmax for the sampling from the next possible transitions.

3.3.3 Rewards for reinforcement learning

We exploit simple immediate rewards for reinforcement learning. We apply the positive reward of $r = 1$ for all transitions of the correct arguments and the negative reward of $r = -1$ for all incorrect transitions. In each transition, the model determines the beginning of the next span t^s, the ending of the span t^e and the label l incrementally. If the model identifies one of the correct arguments from the remaining unresolved arguments, it gets the $r = 3$ positive rewards in total in a single transition. When the model makes a wrong prediction, it receives the $r = -1$ negative reward at this time

[2] If two arguments are at the same distance in the number of sub-tokens from the predicate, we regard the left argument as close to the predicate for the convenience.

[3] The reinforcement learning can also benefits from this de-facto data augmentation. However, it would be less effective than those for *random* because of the limited transition paths.

Model		CoNLL-2005 Dev P	Dev R	Dev F1	WSJ P	WSJ R	WSJ F1	Brown P	Brown R	Brown F1	CoNLL-2012 Dev P	Dev R	Dev F1	Test P	Test R	Test F1
Ouchi+2018	ELMo	87.4	86.3	86.9	88.2	87.0	87.6	79.9	77.5	78.7	87.2	85.5	86.3	87.1	85.3	86.2
Ouchi+2018	ELMo (E)	88.0	86.9	87.4	**89.2**	87.9	88.5	81.0	78.4	79.6	**88.6**	85.7	87.1	**88.5**	85.5	87.0
Li+ 2019	ELMo	-	-	-	85.2	87.5	86.3	74.7	78.1	76.4	-	-	-	84.9	81.4	83.1
Shi+2019	BERT	-	-	-	88.6	89.0	88.8	81.9	82.1	82.0	-	-	-	85.9	87.0	86.5
Zhou+2020	BERT	-	-	-	89.04	88.79	88.91	81.89	80.98	81.43	-	-	-	-	-	-
Li+2020	BERT*	-	-	-	-	-	-	-	-	-	85.97	86.38	86.18	85.82	86.36	86.09
Li+2020	RoBERTa	87.24	87.26	87.25	88.05	88.00	88.03	80.04	79.56	79.80	86.60	86.89	86.74	86.40	86.83	86.61
Zhang+2021	BERT	-	-	-	87.54	88.32	87.93	81.91	82.37	82.14	-	-	-	85.93	87.32	86.62
Left-to-right		87.70	88.16	87.93	88.76	88.94	88.85	82.40	82.59	82.50	87.28	87.83	87.55	87.35	87.89	87.62
Random		87.86	88.30	88.08	88.73	89.07	88.90	82.69	**83.33**	83.01	87.57	87.68	87.62	87.59	87.76	87.67
Random+RL		**88.32**	**88.23**	**88.28**	89.18	**89.11**	**89.15**†	**83.63**	83.13	**83.37**†	87.94	**87.39**	**87.67**	88.04	**87.50**	**87.77**

Table 2: The empirical results in CoNLL-2005 and CoNLL-2012 datasets in labeled attachment score (LAS). Li+2020 uses the original BERT finetuned twice, marked as $*$. (E) denotes the result of the model ensemble. Bold fonts for the best results. We present the averaged result of the three runs with different seeds for `Random+RL`. We confirmed the statistical significance of the test set results in bootstrapped paired t-test in $p < 0.05$ denoted as †.

and it cannot obtain further rewards in this transition. Even if the model made wrong predictions for some arguments in the past transitions, the model is still allowed to obtain positive rewards when the model identifies other correct arguments in later transitions. For example, if a model makes a correct prediction of t_s and an incorrect prediction of t_e for an argument, the model obtains the $r = 1$ reward for the t_s prediction and the $r = -1$ reward for the t_e prediction. This model cannot obtain any rewards regardless of the label l prediction for this argument. However, this model is still allowed to obtain further rewards when it predicts other arguments in later transitions. When the model selects the special token `NULL` which represents the stop iteration, the model obtains the reward of $r = 1$ if the model has resolved all the correct arguments; otherwise, $r = 0$.

We summarize further reinforcement learning details and implementation details in Appendix A.1.

4 Experiments

We conducted experiments with the datasets provided from CoNLL-2005 and 2012 shared tasks (Carreras and Màrquez, 2005; Pradhan et al., 2012). We followed the standard splits of the datasets provided from CoNLL and used the official and standard evaluation script.[4] In CoNLL-2005, sections 2nd-21st of the Wall Street Journal (WSJ) corpus are for the training set and section 24th for the development set. The WSJ section 23rd is for the in-domain test set while Brown corpus 3rd sections is for the out-of-domain test set. CoNLL-2012 dataset is from the OntoNotes v5.0 corpus. See Pradhan et al. (2013) for more details of CoNLL-2012. In the evaluation, the SRL labels of "V" are omitted following the official evaluation script because they are obvious from the marked predicates.

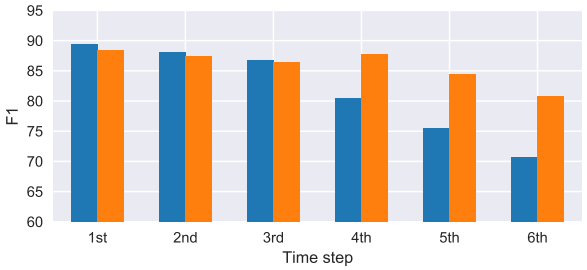

Figure 4: The transition step-wise F1 score in LAS for the first to sixth transition on the development set of CoNLL-2005. Navy (left) for `Random` model and orange (right) for `Random+RL` model.

4.1 Comparison with previous results

First, we compared our models of the iterative argument identification algorithm to the previous state-of-the-art models, including the global decoding model (Zhou et al., 2020) and the variants of sequence labeling-based model (Shi and Lin, 2019; Li et al., 2020; Zhang et al., 2021). These models use the pretrained models of BERT (Devlin et al., 2019) and RoBERTa (Liu et al., 2019). We additionally included the two graph-based model of Ouchi et al. (2018) and Li et al. (2019) with ELMo (Peters et al., 2018) for reference. Note that

[4]This script is available at: `https://www.cs.upc.edu/~srlconll/soft.html`

Label	A0	A1	A2	A3	AM-ADV	AM-DIS	AM-LOC	AM-MNR	AM-MOD	AM-NEG	AM-TMP	R-A0	R-A1
F1													
Random	93.93	90.25	82.96	81.40	67.90	80.11	70.69	70.74	98.07	94.14	88.38	96.10	92.26
Random+RL	94.03	90.46	82.79	83.16	69.12	79.39	72.22	70.88	98.27	93.66	88.28	96.83	93.07
Δ	0.10	0.21	-0.17	1.76	1.22	-0.72	1.53	0.14	0.20	-0.48	-0.10	0.73	0.81
Avg. Steps													
Random	1.96	1.63	2.19	2.52	2.73	2.67	2.26	2.25	2.36	2.66	2.65	2.01	1.42
Random+RL	2.70	1.31	1.59	1.90	2.37	2.58	2.60	1.86	2.57	2.03	2.70	2.50	1.62
Δ	0.74	-0.32	-0.60	-0.62	-0.36	-0.09	0.34	-0.39	0.21	-0.63	0.05	0.49	0.20

Table 3: **Top**: label-wise performance by F1 score. **Bottom**: average step times when argument labels are accurately attached. Results in the development set of CoNLL-2005. Δ is the difference between `Random` and `Random+RL` models. Here, we present 13 label types that appear most frequently in the dataset.

Zhou et al. (2020) also uses syntactic information in the multi-task training and hence the results are not directly comparable.[5]

Table 2 shows the performance of the two heuristic order models of `Left-to-right` and `Random` and the proposed `Random+RL` model that determines the optimal parsing path during training. We also compared results of these models with the performance of previous models. Our model achieves better results than the model of Zhou et al. (2020) that relies on the pretrained model of BERT-large and syntactic information. Li et al. (2020) also use the special BERT-large model that is finetuned twice by the authors. Among all models, the proposed `Random+RL` model achieves the best performance in the F1 scores in both the development set and test set of the CoNLL-05 and CoNLL-12 datasets. We also confirm that our `Random+RL` outperforms other heuristics such as the model trained in `Left-to-right` manner in F1 score as discussed in Appendix A.4.

Paolini et al. (2021) proposed a pre-trained T5-base model (Raffel et al., 2020) for multiple tasks. We noticed that TANL with a single dataset is comparable with our experimental setting even though the pretrained model is quite different. Although they achieved 89.3 in F1 score of WSJ of CoNLL-05, our model out-performs their model performance of 82.0 in F1 in Brown of CoNLL-05. Our `Random+RL` model achieves competitive performance with their 87.7 in F1 of CoNLL-12.

[5]In Zhou et al. (2020), they also reported scores with XL-Net. However, we cannot reproduce their XLNet results in any efforts. They didn't release their codes for XLNet and even didn't reply our emails for asking training details. We therefore decide not to include their XLNet results in Table 2.

4.2 Does Reinforcement learning help?

We apply reinforcement learning (RL) for the model trained with the random ordering. In Table 2, we confirm that reinforcement learning slightly improves the performance in both CoNLL-05 and CoNLL-12. We further investigate the reasons of this performance gain and notice that reinforcement learning surely changes the argument identifications in the later time steps of transitions. Figure 4 presents the comparisons of LAS scores for the predicted arguments in each transition step. In first to third transitions, there are no large differences between the `Random` and `Random+RL` models. However, the `Random+RL` model retains the performance in the later transitions. Although this result is contrary to the intuition of the existing "*easy-first*" strategy, we assume this is one of the reasons why reinforcement learning enhances the final model performance.

We take a close look at the effect of reinforcement learning on the label prediction accuracy. Table 3 presents the two detailed results: the performance comparison and the average transition steps required to identify arguments for each label type. We firstly notice that the `Random+RL` model achieve better or competitive accuracy except of A2, AM-DIS and AM-NEG. We also assume that the `Random+RL` model follows some specific orders in identifying arguments. For some labels such as A3 and AM-ADV, the model chooses to label them first. For some arguments, such as A0 and AM-LOC, the model chooses to label them later.

4.3 How does RL affect SRL ordering?

Here we provide further analyses for the relation of the semantic roles and the identification ordering of labels. The semantic roles of arguments come from

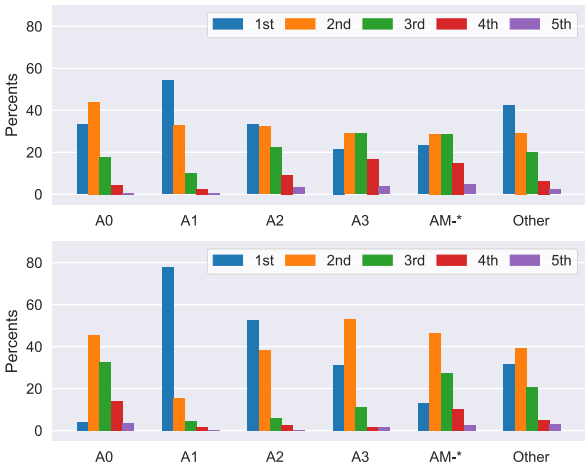

Figure 5: The ratio (%) of the label types identified in from 1st to 5th transitions on the development set of CoNLL-2005. **Top**: imitation learning (Random). **Bottom**: reinforcement learning (Random+RL).

Figure 6: The ratio (%) of the label types of the modifiers (AM-*) identified in from 1st to 5th transitions on the development set of CoNLL-2005. **Top**: imitation learning (Random). **Bottom**: reinforcement learning (Random+RL).

PropBank frames annnotation guidelines.[6] A0 and A1 labels correspond to external and internal arguments in the government and binding theory. They are either subject or object roles depending on the transitive and intransitive verbs. AM-* arguments are modifiers and other labels include referential expressions. We analyze how the resolving orders are affected by these label roles as a result of reinforcement learning. In Figure 5, we present the ratio of the resolved transition steps for each label class. For example, the sharp peak of 1st transition for A1 with Random+RL at the bottom of Figure 5 means nearly 80% of A1 labels in the development set are identified in the first transition.

Here we notice the Random+RL model has a clear tendency of resolving A1 or A2 first if they exist. It also seems that the Random+RL model prefers to identify A0, A3, or AM-* arguments in 2nd or 3rd transitions. We assume this corresponds to the importance of such role labels for predicates. Although the Random model has a similar tendency in some labels, it is less obvious.

Figure 6 presents the details of modifier labels. We present the seventh most frequent modifier argument roles in SRL here. TMP, LOC, DIS, NEG, MOD, ADV and MNR labels correspond to temporal, locative, discourse markers, negation, modals, adverbials, and manner markers respectively. We cannot read strong preferences of resolving orders for the Random model. There are several inconsistent orders: AM-LOC in Random has a weak peak at the 1st transition, while AM-TMP has a weak peak at the 3rd transition. The Random+RL model consistently identifies these arguments mostly in the 2nd transition if they exist. It is also interesting that manner markers have the preference to be identified first and the negation has a strong peak at 2nd transition. Other AM-* modifiers are mostly processed in the 2nd or 3rd transitions by the Random+RL model.

As seen in Figure 4, reinforcement learning improves the labeling accuracy, especially in the later transition steps. For later transition steps, the model uses previously resolved arguments as "clues" to identify the remaining arguments. Therefore the model tunes which argument to identify first and later via reinforcement learning as the Random+RL model introduce specific orders in labeling in Figure 5 and Figure 6. As a result, the model retains the labeling performance in later transitions and hence it outperforms the existing heuristic approaches such as the left-to-right order and random ordering in SRL.

5 Conclusion

We develop the iterative argument identification (IAI) algorithm for the global decoding and iterative resolving for span-based SRL. Our model with IAI is capable of identifying semantic arguments one by one in arbitrary orders. In empirical experiments, we enhance our model with policy-

[6] https://verbs.colorado.edu/~mpalmer/projects/ace/PBguidelines.pdf

gradient-based transition exploration. Our model out-performs the existing models with the same pre-trained model in both CoNLL-05 and CoNLL-12 datasets. In the analyses, we confirm that reinforcement learning enable models to learn a different resolving orders from existing heuristic orders and slightly enhance the performance, which suggest the emergence of the transition path through the training.

Acknowledgements

This work was supported by JSPS KAKENHI Grant Number 22K17983 and 19K2035, and by JST PRESTO Grant Number JPMJPR20C2.

References

Rexhina Blloshmi, Simone Conia, Rocco Tripodi, and Roberto Navigli. 2021. Generating senses and roles: An end-to-end model for dependency- and span-based semantic role labeling. In *IJCAI*.

Xavier Carreras and Lluís Màrquez. 2004. Introduction to the CoNLL-2004 shared task: Semantic role labeling. In *Proceedings of the Eighth Conference on Computational Natural Language Learning (CoNLL-2004) at HLT-NAACL 2004*, pages 89–97, Boston, Massachusetts, USA. Association for Computational Linguistics.

Xavier Carreras and Lluís Màrquez. 2005. Introduction to the CoNLL-2005 shared task: Semantic role labeling. In *Proceedings of the Ninth Conference on Computational Natural Language Learning (CoNLL-2005)*, pages 152–164, Ann Arbor, Michigan. Association for Computational Linguistics.

Xinchi Chen, Chunchuan Lyu, and Ivan Titov. 2019. Capturing argument interaction in semantic role labeling with capsule networks. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 5415–5425, Hong Kong, China. Association for Computational Linguistics.

Jinho D. Choi and Martha Palmer. 2011. Transition-based semantic role labeling using predicate argument clustering. In *Proceedings of the ACL 2011 Workshop on Relational Models of Semantics*, pages 37–45, Portland, Oregon, USA. Association for Computational Linguistics.

Simone Conia and Roberto Navigli. 2020. Bridging the gap in multilingual semantic role labeling: a language-agnostic approach. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 1396–1410, Barcelona, Spain (Online). International Committee on Computational Linguistics.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, Minneapolis, Minnesota. Association for Computational Linguistics.

Marco Dinarelli and Isabelle Tellier. 2018. New recurrent neural network variants for sequence labeling. In *Computational Linguistics and Intelligent Text Processing*, pages 155–173, Cham. Springer International Publishing.

Nicholas FitzGerald, Oscar Täckström, Kuzman Ganchev, and Dipanjan Das. 2015. Semantic role labeling with neural network factors. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 960–970, Lisbon, Portugal. Association for Computational Linguistics.

Daniel Fried and Dan Klein. 2018. Policy gradient as a proxy for dynamic oracles in constituency parsing. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 469–476, Melbourne, Australia. Association for Computational Linguistics.

Yoav Goldberg and Michael Elhadad. 2010. An efficient algorithm for easy-first non-directional dependency parsing. In *Human Language Technologies: NAACL*, pages 742–750, Los Angeles, California.

Luheng He, Kenton Lee, Mike Lewis, and Luke Zettlemoyer. 2017. Deep semantic role labeling: What works and what's next. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 473–483, Vancouver, Canada. Association for Computational Linguistics.

Eric Jang, Shixiang Gu, and Ben Poole. 2017. Categorical reparameterization with gumbel-softmax. In *International Conference on Learning Representations (ICLR)*.

Jungo Kasai, Dan Friedman, Robert Frank, Dragomir Radev, and Owen Rambow. 2019. Syntax-aware neural semantic role labeling with supertags. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 701–709, Minneapolis, Minnesota. Association for Computational Linguistics.

Paul Kingsbury and Martha Palmer. 2002. From TreeBank to PropBank. In *Proceedings of the Third International Conference on Language Resources and Evaluation (LREC'02)*, Las Palmas, Canary Islands - Spain. European Language Resources Association (ELRA).

Shuhei Kurita, Daisuke Kawahara, and Sadao Kurohashi. 2017. Neural joint model for transition-based Chinese syntactic analysis. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1204–1214, Vancouver, Canada. Association for Computational Linguistics.

Shuhei Kurita, Daisuke Kawahara, and Sadao Kurohashi. 2018. Neural adversarial training for semi-supervised Japanese predicate-argument structure analysis. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 474–484, Melbourne, Australia. Association for Computational Linguistics.

Shuhei Kurita and Anders Søgaard. 2019. Multi-task semantic dependency parsing with policy gradient for learning easy-first strategies. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 2420–2430, Florence, Italy. Association for Computational Linguistics.

Minh Lê and Antske Fokkens. 2017. Tackling error propagation through reinforcement learning: A case of greedy dependency parsing. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 1, Long Papers*, pages 677–687, Valencia, Spain. Association for Computational Linguistics.

Mike Lewis, Luheng He, and Luke Zettlemoyer. 2015. Joint A* CCG parsing and semantic role labelling. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 1444–1454, Lisbon, Portugal. Association for Computational Linguistics.

Tao Li, Parth Anand Jawale, Martha Palmer, and Vivek Srikumar. 2020. Structured tuning for semantic role labeling. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 8402–8412, Online. Association for Computational Linguistics.

Zuchao Li, Shexia He, Jiaxun Cai, Zhuosheng Zhang, Hai Zhao, Gongshen Liu, Linlin Li, and Luo Si. 2018. A unified syntax-aware framework for semantic role labeling. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 2401–2411, Brussels, Belgium. Association for Computational Linguistics.

Zuchao Li, Shexia He, Zhao Hai, Yiqing Zhang, Zhuosheng Zhang, Xi Zhou, and Xiaodong Zhou. 2019. Dependency or span, end-to-end uniform semantic role labeling. In *AAAI*.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Roberta: A robustly optimized bert pretraining approach. Cite arxiv:1907.11692.

Chunchuan Lyu, Shay B. Cohen, and Ivan Titov. 2019. Semantic role labeling with iterative structure refinement. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 1071–1082, Hong Kong, China. Association for Computational Linguistics.

Ji Ma, Jingbo Zhu, Tong Xiao, and Nan Yang. 2013. Easy-first POS tagging and dependency parsing with beam search. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 110–114, Sofia, Bulgaria. Association for Computational Linguistics.

Diego Marcheggiani and Ivan Titov. 2020. Graph convolutions over constituent trees for syntax-aware semantic role labeling. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 3915–3928, Online. Association for Computational Linguistics.

Lluís Màrquez, Pere Comas, Jesús Giménez, and Neus Català. 2005. Semantic role labeling as sequential tagging. In *Proceedings of the Ninth Conference on Computational Natural Language Learning (CoNLL-2005)*, pages 193–196, Ann Arbor, Michigan. Association for Computational Linguistics.

André F. T. Martins and Julia Kreutzer. 2017. Learning what's easy: Fully differentiable neural easy-first taggers. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 349–362, Copenhagen, Denmark. Association for Computational Linguistics.

Tahira Naseem, Abhishek Shah, Hui Wan, Radu Florian, Salim Roukos, and Miguel Ballesteros. 2019. Rewarding Smatch: Transition-based AMR parsing with reinforcement learning. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 4586–4592, Florence, Italy. Association for Computational Linguistics.

Joakim Nivre. 2008. Algorithms for deterministic incremental dependency parsing. *Computational Linguistics*, 34(4):513–553.

Hiroki Ouchi, Hiroyuki Shindo, and Yuji Matsumoto. 2018. A span selection model for semantic role labeling. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 1630–1642, Brussels, Belgium. Association for Computational Linguistics.

Giovanni Paolini, Ben Athiwaratkun, Jason Krone, Jie Ma, Alessandro Achille, Rishita Anubhai, Cicero Nogueira dos Santos, Bing Xiang, and Stefano Soatto. 2021. Structured prediction as translation between augmented natural languages. In *9th International Conference on Learning Representations, ICLR 2021*.

Matthew Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer. 2018. Deep contextualized word representations. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 2227–2237, New Orleans, Louisiana. Association for Computational Linguistics.

Sameer Pradhan, Alessandro Moschitti, Nianwen Xue, Hwee Tou Ng, Anders Björkelund, Olga Uryupina, Yuchen Zhang, and Zhi Zhong. 2013. Towards robust linguistic analysis using OntoNotes. In *Proceedings of the Seventeenth Conference on Computational Natural Language Learning*, pages 143–152, Sofia, Bulgaria. Association for Computational Linguistics.

Sameer Pradhan, Alessandro Moschitti, Nianwen Xue, Olga Uryupina, and Yuchen Zhang. 2012. CoNLL-2012 shared task: Modeling multilingual unrestricted coreference in OntoNotes. In *Joint Conference on EMNLP and CoNLL - Shared Task*, pages 1–40, Jeju Island, Korea. Association for Computational Linguistics.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J. Liu. 2020. Exploring the limits of transfer learning with a unified text-to-text transformer. *Journal of Machine Learning Research*, 21(140):1–67.

Robin Senge, Juan José del Coz, and Eyke Hüllermeier. 2014. On the problem of error propagation in classifier chains for multi-label classification. In *Data Analysis, Machine Learning and Knowledge Discovery*, pages 163–170, Cham. Springer International Publishing.

Peng Shi and Jimmy J. Lin. 2019. Simple bert models for relation extraction and semantic role labeling. *ArXiv*, abs/1904.05255.

Anders Søgaard and Yoav Goldberg. 2016. Deep multi-task learning with low level tasks supervised at lower layers. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 231–235, Berlin, Germany. Association for Computational Linguistics.

Emma Strubell, Patrick Verga, Daniel Andor, David Weiss, and Andrew McCallum. 2018. Linguistically-informed self-attention for semantic role labeling. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 5027–5038, Brussels, Belgium. Association for Computational Linguistics.

Oscar Täckström, Kuzman Ganchev, and Dipanjan Das. 2015. Efficient inference and structured learning for semantic role labeling. *Transactions of the Association for Computational Linguistics*, 3:29–41.

Zhixing Tan, Mingxuan Wang, Jun Xie, Yidong Chen, and Xiaodong Shi. 2018. Deep semantic role labeling with self-attention. In *Proceedings of the Thirty-Second AAAI Conference on Artificial Intelligence, (AAAI-18), New Orleans, Louisiana, USA, February 2-7, 2018*, pages 4929–4936. AAAI Press.

Yoshimasa Tsuruoka and Jun'ichi Tsujii. 2005. Bidirectional inference with the easiest-first strategy for tagging sequence data. In *Proceedings of Human Language Technology Conference and Conference on Empirical Methods in Natural Language Processing*, pages 467–474, Vancouver, British Columbia, Canada. Association for Computational Linguistics.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Ł ukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, editors, *Advances in Neural Information Processing Systems 30*, pages 5998–6008. Curran Associates, Inc.

Ronald J Williams. 1992. Simple statistical gradient-following algorithms for connectionist reinforcement learning. pages 5–32. Springer.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Remi Louf, Morgan Funtowicz, Joe Davison, Sam Shleifer, Patrick von Platen, Clara Ma, Yacine Jernite, Julien Plu, Canwen Xu, Teven Le Scao, Sylvain Gugger, Mariama Drame, Quentin Lhoest, and Alexander Rush. 2020. Transformers: State-of-the-art natural language processing. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 38–45, Online. Association for Computational Linguistics.

Travis Wolfe, Mark Dredze, and Benjamin Van Durme. 2016. A study of imitation learning methods for semantic role labeling. In *Proceedings of the Workshop on Structured Prediction for NLP*, pages 44–53, Austin, TX. Association for Computational Linguistics.

Yonghui Wu, Mike Schuster, Zhifeng Chen, Quoc V. Le, Mohammad Norouzi, Wolfgang Macherey, Maxim Krikun, Yuan Cao, Qin Gao, Klaus Macherey, Jeff Klingner, Apurva Shah, Melvin Johnson, Xiaobing Liu, Łukasz Kaiser, Stephan Gouws, Yoshikiyo Kato, Taku Kudo, Hideto Kazawa, Keith Stevens, George Kurian, Nishant Patil, Wei Wang, Cliff Young, Jason Smith, Jason Riesa, Alex Rudnick, Oriol Vinyals, Greg Corrado, Macduff Hughes, and Jeffrey Dean. 2016. Google's neural machine translation system: Bridging the gap between human and machine translation. *CoRR*, abs/1609.08144.

Zhisong Zhang, Emma Strubell, and Eduard H. Hovy. 2021. Comparing span extraction methods for semantic role labeling. In *SPNLP*.

Jie Zhou and Wei Xu. 2015. End-to-end learning of semantic role labeling using recurrent neural networks. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 1127–1137, Beijing, China. Association for Computational Linguistics.

Junru Zhou, Zuchao Li, and Hai Zhao. 2020. Parsing all: Syntax and semantics, dependencies and spans. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 4438–4449, Online. Association for Computational Linguistics.

A Appendix

A.1 Details for reinforcement learning

We apply the cross-entropy loss for imitation learning and the policy gradient (Williams, 1992) for reinforcement learning. For reinforcement learning, we firstly train models with imitation learning until the learning converges, and then finetune it with the policy gradient. We select the best performance of the models at the development set in both imitation learning and reinforcement learning. The reinforcement learning for the argument span is conducted as follows. First, we compute a probability of each (sub-)token becoming the *beginning* of the next argument. We apply sampling over this probability and determine the beginning (sub-)token of the next argument span. With this sampled beginning (sub-)token, our model similarly compute another probability of each (sub-)token becoming the *end* of the next argument span. Again we apply sampling over the probability and determine the end (sub-)token of the next argument span. Finally, the model attaches the label to the sampled span.

We apply Gumbel-softmax for sampling of the next argument to resolve from possible transition paths during the training. For Gumbel-softmax, the inverse temperature parameter β becomes a hyper-parameter. If β is too large, the model samples from very limited transition paths that are close to the narrow path of $\arg\max(\pi_i)$. Thus it gets stuck in the local optima. If β is too small, the model samples from various transition paths that include unrealistic arguments and hence hinder convergence of the training. We perform experiments of training SRL models with $\beta \in \{0.1, 0.5, 1, 2, 3\}$ and report the best performance result of $\beta = 0.5$ for CoNLL-05 and $\beta = 0.1$ for CoNLL-12.

A.2 Training Details

In terms of the batch size, we notice that the larger batch size helps the training. We conducted experiments with the batch size of $[16, 32, 64, 128]$ and obtained the best result at 128. We use the transformer implementation of Hugging Face (Wolf et al., 2020). We apply the pretrained BERT-Large model of Bert-large-cased-whole-word-masking for the sentence encoder, and therefore the hyper-parameters of the sentence encoder are the same as those of the pretrained BERT-Large model with capitalized tokens and whole-word masking. For the partial SRL encoder and the SRL decoder models, we use the hyper-parameters in Table 1. We train our model on machines with four NVIDIA V100 GPU cards. We obtained similar results only with a single NVIDIA V100 GPU card combined with the gradient accumulation.

A.3 What order does the reinforcement learning model prefer to follow?

We further investigate in what orders the models prefer to identify arguments. Here, we analyze which role label the model tends to identify first for a pair of arguments that have the same predicate. Figure 7 represents the heat-map for visualizing the ratio for pairs of semantic role labels before and after each transition. Given the number $N_{X,Y}$ of pairs of arguments for the role label X (in the horizontal axis) and Y (in the vertical axis), we count the cases that role label X is processed *after* the role label Y as $n_{X,Y}$, and we plot $n_{X,Y}/N_{X,Y}$.

In the `Random+RL`, we easily notice that there is a consistent tendency that the model identifies the A0 labels later than any other labels. We check the transition paths of the model processing outputs and confirm that the model frequently identifies A0 labels at last. We also notice that, in Figure 5, A0 has the higher ratio for 3rd, 4th, and 5th transitions than others in `Random+RL`. This might be related to the position of A0 in syntactic trees. We also confirm that the model chooses the A1, A2, and A3 labels first and the AM-* labels later. Among the AM-* labels, AM-NEG and AM-NEG are frequently processed first, although they are mostly processed later than A1 and A2. Overall, the proposed `Random+RL` model identifies semantic role labels and spans as follows: A1 and A2 first, AM-* and other labels later, and A0 label at last. The `Random` model doesn't have such obvious tendencies at a glance.

A.4 Is the traditional left-to-right resolving good for the IAI algorithm?

Contrary to the intuition, we notice that the traditional left-to-right ordering doesn't achieve the best performance for the IAI algorithm among the heuristic orderings. We train our models with five different teacher orders: right-to-left, left-to-right, close-to-distant, distant-to-close, and random as Sec. 3.3.1. The results are shown in Table 4. We notice that the model with the random order teacher performs best in both the in-domain test set of WSJ and the out-of-domain test set of Brown Corpus in CoNLL-05. Similar tendency has observed in the CoNLL-12 dataset. These experiments remind us

Model	CoNLL-2005									CoNLL-2012					
	Dev			WSJ			Brown			Dev			Test		
	P	R	F1	P	R	F1	P	R	F1	P	R	F1	P	R	F1
Left-to-right	87.70	88.16	87.93	88.76	88.94	88.85	82.40	82.59	82.50	87.28	87.83	87.55	87.35	87.89	87.62
Right-to-left	87.98	88.39	88.18	88.76	88.91	88.83	82.31	82.19	82.25	87.34	**88.00**	87.66	87.27	**87.94**	87.60
Close-to-dist.	87.69	**88.44**	88.06	88.50	88.92	88.71	81.95	82.78	82.36	87.25	87.90	87.57	87.18	87.90	87.53
Dist.-to-close	87.76	88.07	87.91	88.68	89.03	88.85	82.33	82.50	82.41	87.10	87.73	87.41	87.04	87.70	87.37
Random	87.86	88.30	88.08	88.73	89.07	88.90	82.69	**83.33**	83.01	87.57	87.68	87.62	87.59	87.76	87.67
Random+RL	**88.32**	88.23	**88.28**	**89.18**	**89.11**	**89.15**	**83.63**	83.13	**83.37**	**87.94**	87.39	**87.67**	**88.04**	87.50	**87.77**

Table 4: The empirical results in CoNLL-2005 and CoNLL-2012 datasets in LAS. We compare five model with different teacher orders in the training: `Left-to-right`, `Right-to-left`, `Close-to-dist.`, `Dist.-to-close` and `Random` and with reinforcement learning (`Random+RL`). "Dev" is the result in the development set. Bold fonts for the best results. We present the averaged scores of the three runs with different seeds.

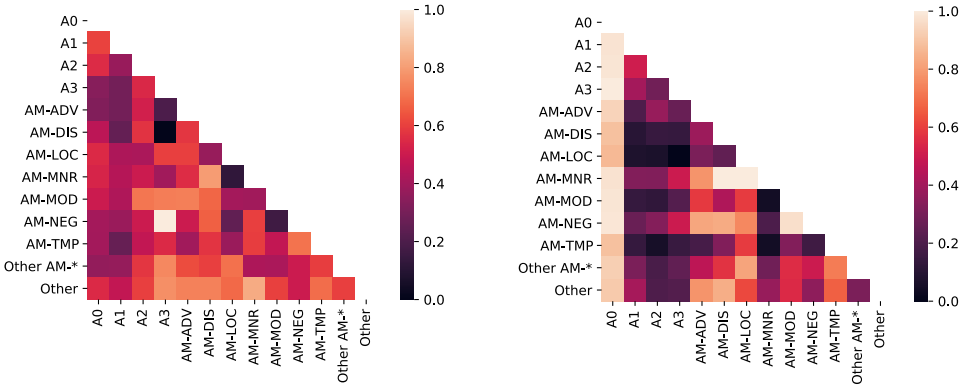

Figure 7: The ratio for the role labels, on the horizontal axis, that are identified *after* the role labels on the vertical axis. The bright color represents the labels on the horizontal axis is likely to be identified *after* the labels on the vertical axis. **Left**: imitation learning (`Random`). **Right**: reinforcement learning (`Random+RL`). Analysis on the development set of CoNLL-2005.

that adapting traditional heuristic ordering is not the best way to train the IAI models. We explore orderings that are better than these heuristics.

A.5 How reinforcement learning affects the argument distance from the predicates?

Figure 8 presents the distribution of the distance from predicates to their argument. We draw four distribution lines that correspond to the 1st, 2nd, 3rd and 4th transitions. Here we count the number of sub-tokens between the predicates and their arguments as the distance. Arguments at the right of their predicates have the positive distance and other arguments have the negative distance. In both `Random` and `Random+RL`, models tend to choose the arguments that are placed right after the predicates. However, this tendency becomes clear in `Random+RL`: the model firstly chooses the arguments right after the predicates and later this model chooses arguments that are placed before the predicates. This suggests that the model learns the new ordering of the argument identification during reinforcement learning.

A.6 Computation times and speed analysis

Iterative argument identifications requires $O(PA)$-times transitions for a sentence that has P predicates and the maximum number of arguments A in theory. However, iterative argument identification has two properties that make it possible to speed up and parallelize the computation. First, the pre-trained transformer-based sentence representations are unchanged during the parsing. This reduce the computation cost. Second, the transitions in iterative argument identifications are independently performed for each predicate. Therefore we can paralleize the transitions for each predicate on the same minibatch of the neural network. Thanks to this predicate-parallelization, the computation times for the overall neural network become the number of argument A if they are on the same minibatch. The average processing speed is 7.5 sentences per second when the minibatch size for evaluation is 48 on a single GPU of NVIDIA V100.

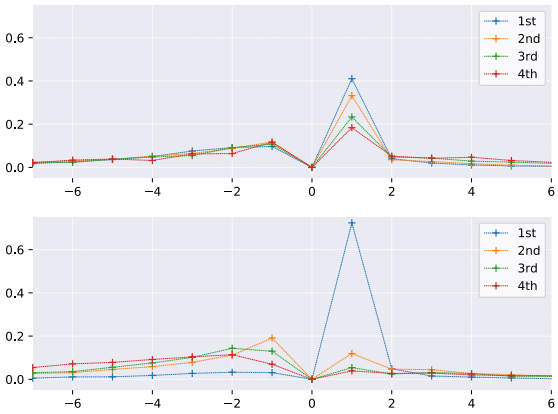

Figure 8: The relation of the argument resolving orders and the distance of predicates and arguments. We represent the first four transitions. **Top**: imitation learning (Random). **Bottom**: reinforcement learning (Random+RL).

A.7 Limitations and potential risks

This work addresses the tools that are developed with the dataset and pretrained models that are widely shared in our community. If the original datasets or pretrained models contain potential risks, our tool might be affected by them. We will take careful looks to prevent our tools from potential abuses.

Revisiting the Practical Effectiveness of Constituency Parse Extraction from Pre-trained Language Models

Taeuk Kim
Dept. of Computer Science & Dept. of Artificial Intelligence
Hanyang University, Seoul, South Korea
kimtaeuk@hanyang.ac.kr

Abstract

Constituency Parse Extraction from Pre-trained Language Models (CPE-PLM) is a recent paradigm that attempts to induce constituency parse trees relying only on the internal knowledge of pre-trained language models. While attractive in the perspective that similar to in-context learning, it does not require task-specific fine-tuning, the practical effectiveness of such an approach still remains unclear, except that it can function as a probe for investigating language models' inner workings. In this work, we mathematically reformulate CPE-PLM and propose two advanced ensemble methods tailored for it, demonstrating that the new parsing paradigm can be competitive with common unsupervised parsers by introducing a set of heterogeneous PLMs combined using our techniques. Furthermore, we explore some scenarios where the trees generated by CPE-PLM are practically useful. Specifically, we show that CPE-PLM is more effective than typical supervised parsers in few-shot settings.

1 Introduction

With the increasing interest in the inner workings of pre-trained language models (PLMs; Devlin et al. (2019); Liu et al. (2019); Radford et al. (2019); Conneau et al. (2020)),[1] much work that attempts to explore the inherent knowledge embedded in the models has been recently proposed. One of the main topics in this direction is to reveal whether PLMs understand syntactic knowledge of human language, usually represented as parse trees. While a line of work (Hewitt and Manning (2019); Chi et al. (2020)) has investigated the existence of syntax in PLMs via structural probes with supervision from gold-standard parse trees, some studies (Kim et al., 2020, 2021; Wu et al., 2020) have found that one can extract reasonable parse structures directly from the patterns presented in PLMs' hidden representations or attention distributions *even without extra fine-tuning*. In other words, the studies have shown that PLMs implicitly store their understanding of syntactic knowledge in their parameters, and that such information can be easily reformulated into syntactic trees with almost no additional cost.

Although the aforementioned approach, dubbed **C**onstituency **P**arse **E**xtraction from **P**re-trained **L**anguage **M**odels (**CPE-PLM**; Kim et al. (2021)), is undoubtedly a useful tool with many analytic uses (Rogers et al., 2020), it still remains a research question whether this algorithm can also work for practical purposes. For instance, as CPE-PLM is free from fine-tuning of PLMs, it may be appealing in few-shot settings, akin to in-context learning for natural language understanding (Brown et al., 2020). Moreover, there exists a potential that the new parsing paradigm can substitute the role of supervised or unsupervised parsers for the case where an NLP model requires a parse tree as input.

In this work, we focus on revealing the practical effectiveness of CPE-PLM. Specifically, we first rewrite the procedure of CPE-PLM in a more rigorous form to clarify its working mechanism. We then introduce two new ensemble algorithms tailored for it, i.e., `Greedy` and `Beam`, making its parsing performance competitive with that of unsupervised parsers (Kim et al., 2019b; Zhu et al., 2020). We show that it is crucial to combine syntactic clues from heterogeneous PLMs for achieving comparable performance, and that this trend holds in not only English but also multilingual cases.

Equipped with the improved variants of CPE-PLM, as the next step, we investigate some scenarios in which their outputs (i.e., generated trees) can be practically utilized. We show that (1) it is viable to introduce the trees from CPE-PLM as auxiliary data for improving Recurrent Neural Network Grammars (RNNG; Dyer et al. (2016); Kim

[1] We use the term *pre-trained language models (PLMs)* to refer to the models that are based on Transformer (Vaswani et al., 2017) and pre-trained in a self-supervised manner, e.g., BERT (Devlin et al., 2019) and its variants.

Proceedings of the 29th International Conference on Computational Linguistics, pages 5398–5408
October 12–17, 2022.

et al. (2019c)), that (2) on classification with Tree LSTMs (Tai et al., 2015), the induced trees can replace gold-standard parses with a minimal loss, and that (3) CPE-PLM can be even more data-efficient than supervised parsers in few-shot settings.

2 Background and Related Work

2.1 Constituency Parse Extraction from Pre-trained Language Models

The term "**C**onstituency **P**arse **E**xtraction from **P**re-trained **L**anguage **M**odels (**CPE-PLM**)" coined by Kim et al. (2021) represents a range of parsing methods (Mareček and Rosa, 2019; Rosa and Mareček, 2019; Kim et al., 2020, 2021; Li et al., 2020) that aim to infer the parse tree of an input sentence by only exploiting the features obtained from PLMs. In detail, the approaches belonging to this paradigm attempt to directly (i.e., without training) apply simple heuristics or existing parsing algorithms, such as top-down (Shen et al., 2018a) and chart-based (Kitaev and Klein, 2018) ones, on the hidden representations or attention maps retrieved from PLMs. In the following, we illustrate the exact formulation of some representative methods.

As CPE-PLM does not demand more than *frozen* PLMs as its ingredient, which means *training-free*, it can be particularly useful when there are no resources available for training supervised parsers in terms of either (1) computing resources or (2) training data consisting of gold-standard annotations. However, its use has been limited to analytic purposes in the literature, utilized as a tool for probing the inner workings of PLMs. Our goal in this paper is therefore to investigate the utility of CPE-PLM in practical scenarios. Among several options, we select the chart-based variant (Kim et al., 2021) as our baseline, which generally outperforms others.

Chart-based CPE-PLM. Kim et al. (2021) propose a method that combines PLMs with the chart parsing algorithm without extra training. Formally, each tree candidate T for an input sentence, $w_1, w_2, \ldots, w_z$, is assigned a score $s_{tree}(T)$ that decomposes as $s_{tree}(T) = \sum_{(i,j)\in T} s_{span}(i,j)$, where $s_{span}(i,j)$ is a score for a constituent that is located between positions i and j in the sentence. $s_{span}(i,j)$ is defined as follows:

$$s_{span}(i,j) = \begin{cases} s_{comp}(i,j) + \min_{i\le k<j} s_{split}(i,k,j) & \text{if } i<j \\ 0 & \text{if } i=j \end{cases},$$

where $s_{split}(i,k,j) = s_{span}(i,k) + s_{span}(k+1,j)$. In other words, $s_{comp}(i,j)$ measures the compositionality of the span (i,j) itself while $s_{split}(i,k,j)$ indicates how plausible it is to divide the span (i,j) into two subspans (i,k) and $(k+1,j)$. Note that every $s_{span}(i,j)$ can be easily calculated in a bottom-up fashion with the aid of the CKY algorithm (Cocke, 1969; Kasami, 1966; Younger, 1967), once $s_{comp}(i,j)$ is properly defined.

Although the authors suggest two derivations for $s_{span}(i,j)$, in this work, the pair score function $s_p(\cdot,\cdot)$ is chosen as the main target of our interest, which is defined as follows:

$$s_p(i,j) := \binom{j-i+1}{2}^{-1} \sum_{(w_x,w_y)\in pair(i,j)} f(g(w_x), g(w_y)),$$

where $pair(i,j)$ returns a set consisting of all combinations of two words from a span (i,j), e.g., $pair(1,3) = \{(w_1,w_2),(w_1,w_3),(w_2,w_3)\}$, while $f(\cdot,\cdot)$ and $g(\cdot)$ are a distance measure function and representation extractor function respectively. To realize $f(\cdot,\cdot)$ and $g(\cdot)$, the authors consider two sets of functions, F and G. Given l as the number of layers in a PLM and a as the number of attention heads per layer, G refers to the set of functions $\{g_{(m,n)} | m = 1,\ldots,l, n = 1,\ldots,a\}$, each of which outputs the attention distribution of an input word computed by the nth attention head on the mth layer of the PLM. F is specified as $\{\texttt{JSD}, \texttt{HEL}\}$, where JSD and HEL correspond to the Jensen-Shannon and Hellinger distance. Here, HEL is only considered for simplicity.

Intuitively, a series of the operations described so far can be understood as (1) splitting an attention map by rows (i.e., into attention distributions) for representing each word and (2) comparing similarities between the rows to gauge the syntactic proximity of the corresponding words. Finally, chart-based CPE-PLM outputs $\hat{T}$, the tree that requires the lowest cost to build, as a prediction for the parse tree of the input sentence: $\hat{T} = \arg\min_T s_{tree}(T)$.

2.2 Ensemble Methods for CPE-PLM

In Section 2.1, we merely mentioned the case where only an element of G is adopted for computing the scores used in CPE-PLM. However, it is also feasible to employ more than just one, such that the method is provided with diverse information from different attention maps (i.e., $g_{(m,n)}$) to derive more reasonable parse trees. In fact, the previous work on CPE-PLM (Kim et al., 2020, 2021) already exploits such *ensemble* strategies in an *ambiguous* and *implicit* manner to boost its performance. As the introduction of those techniques can

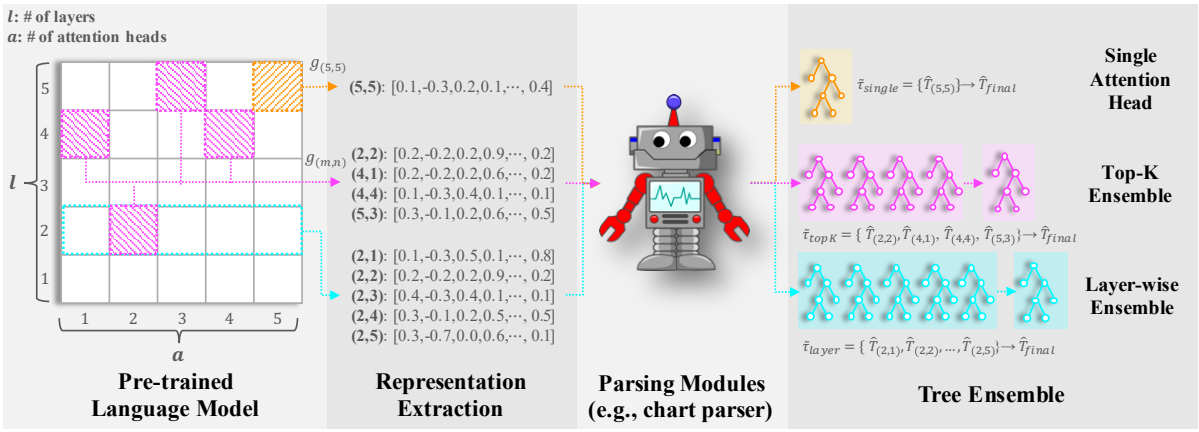

Figure 1: Concept diagram explaining the procedure of CPE-PLM with various ensemble methods. Given a PLM that has $l = 5$ Transformer layers, each of whose self-attention module consists of $a = 5$ individual attention heads, an input sentence is inserted into the PLM to compute the model's attention maps. Then, we construct parse candidates using the information from each $g_{(m,m)}$ and the CKY algorithm. Here, the role of ensemble methods is to determine which trees to be engaged in the final prediction of the resulting parse tree.

lead to a considerable gap in the final result, we claim that it is essential to clarify which ensemble method is employed and to develop more advanced ones. Accordingly, we here *explicitly* formulate the previous ensemble methods, which is one of our contributions in this work.

Figure 1 explains the process of applying ensemble methods to CPE-PLM. Formally, let $\tau := \{\hat{T}_{(m,n)} | m = 1, \ldots, l, n = 1, \ldots, a\}$ denote a pool of all the possible tree predictions computed with CPE-PLM using every $g_{(m,n)}$. The objective of the ensemble methods is to derive the best parse prediction from a subset of τ, denoted $\widetilde{\tau}$, so that it closely resembles the corresponding gold-standard tree. Once $\widetilde{\tau}$ is decided, every $\hat{T}_{(m,n)}$ from $\widetilde{\tau}$ is converted into the form of *syntactic distance* (Shen et al., 2018a) $\mathbf{d}_{(m,n)} \in \mathbb{R}^{z-1}$. [2] Then, the resulting vectors are averaged to derive $\mathbf{d}_{final}$, which is finally restored to the tree form $\hat{T}_{final}$.[3] In the following, we illustrate the characteristics of each ensemble technique shown when determining $\widetilde{\tau}$.

Naïve Baseline: Single Attention Head. The simplest way of implementing CPE-PLM is to utilize just a single attention head as a representative. To be specific, this baseline constructs $\widetilde{\tau}_{single} := \{\hat{T}_{(m^*,n^*)} | \forall m \forall n, val(g_{(m,n)}) \leq val(g_{(m^*,n^*)})\}$, where $val(g_{(m,n)})$ indicates the performance of CPE-PLM on the *validation* set, given $g_{(m,n)}$. In other words, it directly outputs the parse $\hat{T}_{(m^*,n^*)}$ generated by the best function $g_{(m^*,n^*)}$.

Layer-wise Ensemble. Kim et al. (2020) suggest to merge a group of trees that originated from the attention heads located in the same layer of a PLM. Specifically, this ensemble method defines $\widetilde{\tau}^m_{layer} := \{\hat{T}_{(m,n)} | n = 1, \ldots, a\}$ to consider layer-specific information. The best layer of the PLM (i.e., m^*) is also determined by its performance on the validation set. The intuition behind this heuristic is that attention heads of the same layer might be complementary cooperative in grasping a linguistic concept, considering that particular layers of a PLM seem specialized to capture a specific aspect of linguistic knowledge (Tenney et al., 2019; Jawahar et al., 2019; Jo and Myaeng, 2020).

Top-K Ensemble. Kim et al. (2021) propose utilizing the top-K $g(\cdot)$ functions instead of only using the best, $g_{(m^*,n^*)}$. First, a sorted set $\tau_{sorted} := \{\hat{T}_{(m^i,n^i)} | i, j \in \{1, \ldots, l \times a\} \cap val(g_{(m^i,n^i)}) \geq val(g_{(m^j,n^j)})$ whenever $i \leq j\}$ is specified as a variant of the set τ. That is, τ_{sorted} is identical to τ except that the elements of τ_{sorted} are arranged in descending order according to the validation performance of their corresponding functions $\{g_{(m^i,n^i)}\}$. Then, the top-K ensemble method defines $\widetilde{\tau}_{topK}$ as the set consisting of the first K elements of τ_{sorted}.

3 Proposed Methods

In this section, we additionally introduce two novel ensemble methods for CPE-PLM, which are more effective than the previous counterparts in improv-

[2] Note that z is the number of words in the input sentence.

[3] Refer to Kim et al. (2021) for the exact procedure of converting a tree into a syntactic distance and vice versa.

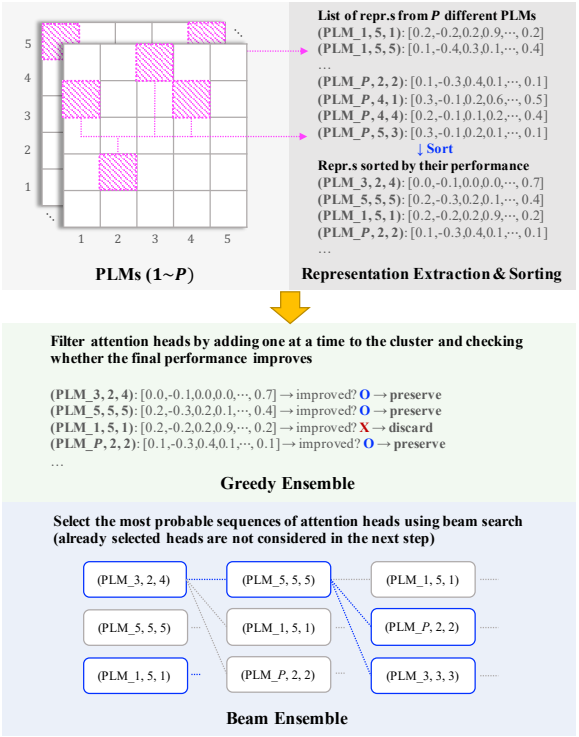

Figure 2: Concept diagram explaining the operation of `Greedy` and `Beam` in multi-PLM environments.

ing parsing performance. Furthermore, we propose to allow several PLMs to collaborate with each other to provide CPE-PLM with more diverse syntactic information (Figure 2).

Greedy Ensemble. We first consider a method that collects every helpful attention head in a greedy fashion. Unlike the previous ensemble methods which allow only a fixed number of attention heads to be engaged in the ensemble process, this sets no limit on the number of participants (i.e., attention heads), diversifying the source of syntactic clues.

Sticking to the notations defined in Section 2.2, we specify the `Greedy` ensemble algorithm in Algorithm 1. Its core logic is to append one attention head at a time to the cluster and test whether each augmentation is beneficial for making progress in the final (validation) performance ($val(G_{greedy})$).

Beam Ensemble. Even though the greedy ensemble method is simple and effective, there still exists a need for exploring more diverse groups of attention heads that have the potential to show better performance than the group chosen by the greedy algorithm. To this end, inspired by the beam search algorithm widely adopted in the natural language generation (NLG) literature, we introduce the `Beam` ensemble method in Algorithm 2.

Algorithm 1 Greedy Ensemble Algorithm

$G_{sorted} := \{g_{(m^i,n^i)} | i,j \in \{1,\dots,l\times a\} \cap val(g_{(m^i,n^i)}) \geq val(g_{(m^j,n^j)})$ whenever $i \leq j\}$
$g_{(m^i,n^i)} :=$ The ith element of G_{sorted}
$\hat{T}_{(m^i,n^i)} :=$ A tree prediction generated using $g_{(m^i,n^i)}$
function GREEDY(G_{sorted})
  $G_{greedy}, \widetilde{\tau}_{greedy}, \mu \leftarrow \{\}, \{\}, 0$
  for $i = 1,\dots,l \times a$ **do**
    $G_{greedy} \leftarrow G_{greedy} \cup \{g_{(m^i,n^i)}\}$
    $\psi \leftarrow val(G_{greedy})$
    if $\psi > \mu$ **then**
      $\mu \leftarrow \psi$
      $\widetilde{\tau}_{greedy} \leftarrow \widetilde{\tau}_{greedy} \cup \{\hat{T}_{(m^i,n^i)}\}$
    else
      $G_{greedy} \leftarrow G_{greedy} \backslash \{g_{(m^i,n^i)}\}$
    end if
  end for
  return $\widetilde{\tau}_{greedy}$
end function

Algorithm 2 Beam Ensemble Algorithm

$b :=$ beam size
$G_{sorted} := \{g_{(m^i,n^i)} | i,j \in \{1,\dots,l\times a\} \cap val(g_{(m^i,n^i)}) \geq val(g_{(m^j,n^j)})$ whenever $i \leq j\}$
$g_{(m^i,n^i)} :=$ The ith element of G_{sorted}
function BEAM(G_{sorted}, b)
  $G_{beam}, e \leftarrow \{\}, 0$
  for $i = 1,\dots,b$ **do**
    $G_{beam} \leftarrow G_{beam} \cup \{\{g_{(m^i,n^i)}\}\}$
  end for
  while $e < b$ **do**
    $\hat{G}, \psi, e \leftarrow \{\}, \{\}, 0$
    for each $H \in G_{beam}$ **do**
      $\mu \leftarrow$ The largest index j given $\forall g_{(m^j,n^j)} \in H$.
      for $i = 1,\dots,b$ **do**
        if $\mu + i > l \times a$ **then**
          $e \leftarrow e + 1$
          break
        else
          $H_i \leftarrow H \cup \{g_{(m^{\mu+i},n^{\mu+i})}\}$
          $\hat{G} \leftarrow \hat{G} \cup \{H_i\}$
          $\psi \leftarrow \psi \cup \{val(H_i)\}$
        end if
      end for
    end for
    $\psi_{topB} \leftarrow$ The set consisting of the top b elements of ψ.
    $G_{beam} \leftarrow \{H | H \in \hat{G} \cap val(H) \in \psi_{topB}\}$
  end while
  $G^*_{beam} \leftarrow \{H^* | \forall H \in G_{beam}, val(H^*) \geq val(H)\}$
  $\widetilde{\tau}_{beam} \leftarrow \{\hat{T}_{(m^i,n^i)}) | \hat{T}_{(m^i,n^i)}$ is predicted using $g_{(m^i,n^i)} \in H^*$ (H^* is the sole element of G^*_{beam})$\}$
  return $\widetilde{\tau}_{beam}$
end function

Our beam ensemble algorithm is similar to one for NLG (Graves, 2012; Sutskever et al., 2014) except that the beam search procedure is not stochastic, but determined by the order of the elements of G_{sorted}. Furthermore, the already selected attention heads are not considered in the next search step, unlike NLG which allows the same word to be generated twice or more (see Figure 2 for example).

The merit of Beam is that it can explore a wider range of potential paths that might not be covered by the greedy algorithm.

Extension to Multi-PLM Settings. Until now, we have assumed that CPE-PLM is only applicable for only a single PLM. However, we propose for the first time extending its usage to the scenario in which multiple PLMs are available together. In other words, we expand τ to $\tau_{multi} := \{\hat{T}_{(p,m,n)} | p \in \{1,\ldots,P\}, m \in \{1,\ldots,l\}, n \in \{1,\ldots,a\}\}$, where P is the number of the PLMs involved. By doing so, it is expected that CPE-PLM can infer more accurate parse trees with the aid of diverse perspectives from heterogeneous PLMs. We show in Section 4.2 that this simple and intuitive extension leads to a significant improvement in the final performance. It also has value in that it is one of the initial attempts in the literature to leverage different PLMs simultaneously.

4 Experiments on Parsing

In this chapter, we aim to validate the effectiveness of the proposed ensemble algorithms, i.e., Greedy and Beam, which enable CLE-PLM to have a higher potential of being properly adopted for downstream tasks.

4.1 General Configurations

Datasets. To evaluate the parsing performance of CPE-PLM and other related models, we utilize the Penn Treebank dataset (PTB, Marcus et al. (1993)) for English and the SPMRL (Seddah et al., 2013) dataset for eight other languages, following Kim et al. (2021).[4] We also adhere to the standard of the previous work for preprocessing the datasets.

Evaluation Metrics. We utilize the *unlabeled* sentence-level $F1$ score as a main metric to evaluate the extent to which induced trees resemble corresponding gold-standard trees. It was originally introduced by Shen et al. (2018b, 2019), becoming the de-facto standard in unsupervised parsing.

Model Selection & Hyperparameters. As mentioned in Section 2.1, we build our approach upon chart-based CPE-PLM (Kim et al., 2021). Moreover, we employ twelve English PLMs and four multilingual PLMs to provide syntactic information.[5] To handle various PLMs in an integrated manner, we use the Transformers library developed by HuggingFace (Wolf et al., 2019). We determine hyperparameters for the Top-K and Beam ensemble methods using grid search. In consequence, we use K=20 and b=5 for single PLM cases and K=30 and b=30 in multi-PLM settings.

PLMs / Methods	Previous work		Chart CPE-PLM w/ ensemble methods				
	Top-down†	Chart‡	Single	Layer	Top-K	Greedy	Beam
Encoder-based							
BERT-base	32.4	42.7	34.1	35.3	42.5	43.0	42.8
BERT-large	34.2	44.2	38.7	40.6	44.4	45.0	44.5
RoBERTa-base	33.8	44.9	40.9	39.2	44.2	45.4	45.4
RoBERTa-large	34.1	41.9	39.5	38.9	44.9	47.2	43.7
ELECTRA-base	-	-	40.2	41.2	43.3	46.9	43.2
ELECTRA-large	-	-	**44.3**	41.3	46.6	47.9	47.2
Decoder-based							
GPT2	37.1	37.2	34.5	26.4	36.9	37.2	37.1
GPT2-medium	39.4	38.4	38.0	28.2	38.2	38.0	40.8
CTRL	-	-	35.7	28.7	44.4	45.8	44.9
Hybrid							
BART-large	-	-	37.5	32.6	39.8	37.5	38.5
XLNet-base	**40.1**	46.4	36.7	39.7	46.0	47.0	46.7
XLNet-large	38.1	46.4	39.5	38.9	45.7	47.2	46.8
Multilingual							
MBERT	-	45.0	39.0	40.3	44.6	47.1	45.7
XLM	-	**47.7**	41.9	42.1	47.1	47.5	47.1
XLM-R	-	46.7	41.6	**44.2**	46.5	48.5	47.4
XLM-R-large	-	44.6	40.7	36.7	44.3	46.8	46.9
Multiple PLMs							
Only multilingual	-	-	-	-	49.6	51.9	49.8
All models	-	-	-	-	**50.4**	**55.3**	**55.7**

Table 1: $F1$ scores of CPE-PLM on the PTB test set conditioned on the different combinations of PLMs and ensemble methods. We show that Greedy and Beam are more effective than baselines, and that we attain much more competitive scores in multiple PLM settings. The best score for each column is in **bold**. We also report numbers from two previous studies for reference. †: From Kim et al. (2020). ‡: From Kim et al. (2021).

4.2 Verification of CPE-PLM's Performance

We first conduct experiments on the English PTB dataset using CPE-PLM, with the objective of comparing the effects of different PLMs and ensemble methods on the paradigm. We also test the settings in which multiple PLMs are employed at the same time. From Table 1, we confirm that our Greedy and Beam algorithms are more effective than other techniques in most cases and that their impact is amplified when combined with multiple PLMs. As a result, we succeed in achieving the state-of-the-

[4] We use national codes to represent languages, i.e., en: English, eu: Basque, fr: French, de: German, he: Hebrew, hu: Hungarian, ko: Korean, pl: Polish, and sv: Swedish.

[5] The list of PLMs we use is (1) English PLMs: BERT-base/large (Devlin et al., 2019), RoBERTa-base/large (Liu et al., 2019), ELECTRA-base/large (Clark et al., 2020), GPT2(-medium) (Radford et al., 2019), CTRL (Keskar et al., 2019), BART (Lewis et al., 2020), XLNet (Yang et al., 2019), (2) multilingual PLMs: MBERT, XLM (Conneau and Lample, 2019), XLM-R(-large) (Conneau et al., 2020).

Models	F_1	SBAR	NP	VP	PP	ADJP	ADVP
Unsupervised parsers							
PRPN†	47.3	50	59	46	57	44	32
ON-LSTM †	48.1	51	64	41	54	38	31
Neural PCFG†	50.8	52	71	33	58	32	45
Compound PCFG †	55.2	**56**	74	41	68	40	52
Neural L-PCFG‡	55.3	53	67	**48**	65	49	58
CPE-PLM (Ours)							
XLM-R + Greedy	48.5	46	69	29	62	48	73
All PLMs + Greedy	55.3	54	**75**	36	**76**	**50**	**76**
All PLMs + Beam	**55.7**	53	74	42	75	46	72

Table 2: Comparison of the best CPE-PLM variants with unsupervised parsers. We show that with the aid of `Greedy` and `Beam`, CPE-PLM becomes competitive with unsupervised PCFGs. We also report recall scores on six phrasal categories in addition to $F1$ scores. The best result for each column is in **bold**. †: From Kim et al. (2019b). ‡: From Zhu et al. (2020).

art $F1$ score (55.7) on PTB in the CPE-PLM literature, improving by up to eight points compared against the previous best (47.7). We also observe that Transformer encoder-based and multilingual PLMs are more attractive options for CPE-PLM.

Second, we take the best instances of CPE-PLM from Table 1 and compare them with a set of unsupervised parsers on PTB. In particular, we consider PRPN (Shen et al., 2018b), ON (Shen et al., 2019), Neural PCFG, Compound PCFG (Kim et al., 2019b), and Neural L-PCFG (Zhu et al., 2020) as baselines. Note that all the models including CPE-PLM are evaluated on the same condition where we assume we have access to the validation set (for tuning hyperparameters), following the prior work (Kim et al., 2019b). From Table 2, we show that with the introduction of `Greedy` and `Beam`, CPE-PLM becomes comparable to off-the-shelf unsupervised parsers in terms of $F1$. Specifically, our CPE-PLM instance with `Beam` succeeds in achieving the best $F1$ score among all the candidates, and the variant with `Greedy` accomplishes the best recall scores on four phrasal categories (NP, PP, ADJP, and ADVP). Based on these quantitative results, we claim that CPE-PLM is proper to be an alternative for unsupervised parsers in some cases.

Finally, we extend the language domain of our experiments from English to eight other languages. We exploit four multilingual PLMs that are capable of processing all the languages we consider, and each ensemble method is optimized for respective languages. In Table 3, we demonstrate that our ensemble methods are universally more effective than the top-K algorithm across different languages. Furthermore, we confirm that CPE-PLM with `Greedy` attains the best performance in every case when operated on the combination of all the four PLMs and, showing 47.5 $F1$ score on average.

Models / Language	en	eu	fr	de	he	hu	ko	pl	sv	Avg.
Single PLM										
MBERT										
Top-K ensemble	44.6	39.3	35.9	35.9	37.8	33.2	47.5	51.1	32.6	39.8
Greedy ensemble	47.1	40.2	36.9	37.5	38.6	30.2	49.1	52.4	31.9	40.4
Beam ensemble	45.7	41.2	36.1	37.6	38.0	33.8	49.1	51.4	32.6	40.6
XLM										
Top-K ensemble	47.1	34.6	36.4	43.8	41.0	36.3	33.6	58.5	36.0	40.8
Greedy ensemble	47.5	38.4	37.0	45.4	41.5	36.4	35.1	58.0	36.4	41.7
Beam ensemble	47.1	38.7	36.8	43.6	41.9	36.3	35.2	56.8	36.2	41.4
XLM-R										
Top-K ensemble	46.5	39.5	35.8	37.5	40.1	36.6	49.8	52.7	32.8	41.3
Greedy ensemble	48.5	39.4	36.1	39.0	40.3	36.5	50.8	53.5	33.1	41.9
Beam ensemble	47.4	39.0	35.3	37.9	39.7	37.0	50.2	53.9	32.7	41.5
XLM-R-large										
Top-K ensemble	44.3	37.2	29.7	36.3	35.8	31.0	45.5	44.7	27.6	36.9
Greedy ensemble	46.8	39.5	32.9	40.1	37.0	34.0	46.4	47.1	31.0	39.4
Beam ensemble	46.9	39.2	33.0	39.2	36.1	33.4	45.8	50.7	29.0	39.3
Multiple PLMs										
All models										
Top-K ensemble	49.6	40.9	38.8	44.3	44.5	38.5	51.1	58.7	37.2	44.8
Greedy ensemble	**51.9**	**44.0**	**41.9**	**47.3**	**48.1**	**40.1**	**53.7**	**61.4**	**39.0**	**47.5**
Beam ensemble	49.8	42.7	40.4	47.0	45.9	39.4	53.4	60.8	38.2	46.4

Table 3: CPE-PLM with three ensemble methods for nine languages. We observe that it is optimal for every language to leverage `Greedy` on top of the integration of all the four multilingual PLMs considered. The best result for each column is in **bold**.

5 Experiments on Downstream Tasks

In the previous section, we demonstrated that CPE-PLM's performance can be significantly improved by introducing the techniques proposed in this work. We now turn our attention towards its outputs (i.e., generated parse trees) and investigate the utility of such trees in two application scenarios where tree structures are taken as input.

5.1 Training (U)RNNG with Induced Trees

Recurrent Neural Network Grammar (RNNG) (Dyer et al., 2016) and its unsupervised variant (URNNG, Kim et al. (2019c)) are neural architectures which perform language modeling and parsing together. In Kim et al. (2019b), the authors showed that training (U)RNNG with the trees generated by other unsupervised parsers results in a parsing model that is even better than the parsers which provided the trees used in training. Following the previous work, we here examine whether the output trees from CPE-PLM can also function as meaningful signals for training (U)RNNG. For our experiments, we acquire the best two instances from Table 1 (which accomplished 55.3 and 55.7 parsing $F1$ scores respectively) and use them as our pseudo parsers. We employ Compound PCFG (Kim et al., 2019b) as an unsupervised parser baseline.

From Kim et al. (2019b) (The *best* over trials)	PPL (↓)	F_1 (↑)
LSTM LM	86.2	–
PRPN	87.1	47.9
Induced RNNG	95.3	47.8
Induced URNNG	90.1	51.6
ON	87.2	50.0
Induced RNNG	95.2	50.6
Induced URNNG	89.9	55.1
Neural PCFG	252.6	52.6
Induced RNNG	95.8	51.4
Induced URNNG	86.0	58.7
Compound PCFG (**best**)	196.3	60.1
Induced RNNG	89.8	58.1
Induced URNNG	**83.7**	**66.9**
Our results (*Averaged* over several trials)	PPL (↓)	F_1 (↑)
Compound PCFG (re-experimented, **average**)	–	54.0
Induced RNNG	91.5	54.7
Induced URNNG	85.4	57.8
CPE-PLM (All PLMs + `Greedy`)	–	55.3
Induced RNNG	86.3	55.0
Induced URNNG	**81.3**	57.2
CPE-PLM (All PLMs + `Beam`)	–	55.7
Induced RNNG	87.3	57.0
Induced URNNG	82.0	**60.7**

Table 4: Experiments on training (U)RNNG with the trees induced by unsupervised parsers and CPE-PLM. The upper section presents the results reported by Kim et al. (2019c) while the bottom shows the outcomes from our experiments. The best numbers for each column of the respective sections are in **bold**. We show that the (U)RNNGs trained with the trees induced by CPE-PLM attain better language modeling and parsing abilities compared to the cases of unsupervised parsers.

In Table 4, we present results from Kim et al. (2019b) and our experiments on PTB. Note that our results and ones from the previous study are not directly comparable, because we report the performance of each model *averaged* over 4 different runs while Kim et al. (2019b) utilize the *best* instance. From the experimental results, we confirm that the (U)RNNG models trained with CPE-PLM have better language modeling capability than those trained with other unsupervised parsers. In particular, we attain the perplexity of 81.3 when leveraging our greedy ensemble algorithm for CPE-PLM, outperforming the strong baseline (Compound PCFG: 85.4). Moreover, we succeed in obtaining a more powerful parsing model by training (U)RNNG with the aid of CPE-PLM (All PLMs + `Beam`). Using this, we achieve 60.7 in $F1$, 5 points higher than that of the original (55.7).

5.2 Text Classification using Tree LSTM

Recursive neural network (RvNN; Socher et al. (2013); Tai et al. (2015)) is a type of neural architecture, whose composition order is determined by an input tree structure. In spite of RvNNs' strong performance on several sentence-level tasks and robust linguistic motivation on which they were invented, the usage of RvNNs is generally restricted due to their reliance on gold/silver-standard trees.[6] We here attempt to mitigate this limitation by taking advantage of CPE-PLM. To this end, we conduct experiments on four text classification tasks with Tree LSTMs: the target tasks are SST2 (Socher et al., 2013), MR (Pang and Lee, 2005), SUBJ (Pang and Lee, 2004), and TREC (Li and Roth, 2002). We use a variant of Tree LSTM (Kim et al., 2019a) whose leaf nodes are processed by a separate LSTM in advance. We inject three distinct types of trees—right-branching trees, which are a strong heuristic-based approach in English, the trees induced by CPE-PLM (with `Beam`), and those generated by a supervised parser—into the model and evaluate their impact on the final performance.

Models / Tasks (Metric: Acc.)	SST2	MR	SUBJ	TREC
Tree LSTM				
+ Right-branching trees	85.72	83.37	94.80	94.50
+ **CPE-PLM (All PLMs + Beam ensemble)**	86.10	**83.62**	94.85	94.75
+ Supervised parser (Klein and Manning, 2003)	**86.70**	**83.62**	**95.12**	**95.05**

Table 5: Text classification with Tree LSTMs. We observe that CPE-PLM-oriented parses outperform right-branching trees but are inferior to silver-standard trees. All the results are averaged over four different runs.

In Table 5, we present the accuracy of diverse Tree LSTM instances on four tasks. Although the absolute difference in accuracy between the instances is marginal, we discover a clear pattern that silver-standard trees (ones from supervised parsers) are always the most helpful while the parses induced by CPE-PLM rank second, outperforming right-branching trees. This outcome supports our claim that CPE-PLM can be an attractive option when supervised parsers are not available.

On the other hand, we find that the performance of Tree LSTMs is not that sensitive to their tree inputs, which was similarly observed by Shi et al. (2018). However, we highlight that the trees closer to their gold-standard counterparts are more beneficial across all the tasks considered. We leave as future work the application of CPE-PLM to advanced tree models that are more input structure-sensitive.

6 Discussion

So far, we have focused on verifying the utility of CPE-PLM through the lens of (1) its improved parsing performance and (2) the effectiveness of its

[6] We use the term *silver-standard* trees to indicate parse trees predicted by sophisticated supervised parsers.

CPE-PLM configurations	Used proportion of validation set				
	1%	2%	5%	10%	100%
All PLMs					
+ `Greedy`	49.4	49.9	52.7	54.3	55.3
Relative loss (-)	5.9	5.3	2.5	0.9	-
+ `Beam`	51.3	49.8	51.8	52.9	55.7
Relative loss (-)	4.5	6.0	4.0	2.9	-

Table 6: Relative performance loss of CPE-PLM on PTB with regard to the proportion of the validation set used. We obtain reasonable performance only with 1% (17 examples) of the validation set.

Models	Number of used annotations								
	1	2	5	10	17 (1%)	2%	5%	10%	100%
CPE-PLM (All PLMs)									
+ `Greedy`	**46.2**	**48.4**	**49.9**	49.1	49.4	**49.9**	**52.7**	54.3	55.3
+ `Beam`	45.4	45.9	47.7	**49.6**	**51.3**	49.8	51.8	52.9	55.7
Supervised (Benepar)	-	11.6	12.5	14.0	17.0	31.1	50.2	**71.4**	**92.2**

Table 7: Comparison between CPE-PLM and a supervised parser (Benepar) in few-shot settings.

output trees for downstream tasks. In this section, we conduct in-depth analysis on the limitations of the current form of CPE-PLM and propose countermeasures to alleviate the problems.

Reliance on the validation set. CPE-PLM is training-free, but it exploits gold-standard trees from the validation set to decide the best combination of attention heads ($g_{(m,n)}$). Although we allow this configuration in this work to have a fair comparison with some previous work on unsupervised parsing (Kim et al., 2019b) that also made use of the validation set to optimize hyperparameters, it is always better to reduce such reliance as argued in the few-shot classification literature (Perez et al., 2021). Therefore, we here attempt to examine the robustness of CPE-PLM with respect to the number of data instances from the validation set. Specifically, we conduct a controlled experiment where CPE-PLM is provided with only a limited proportion of the validation set. In Table 6, we confirm that CPE-PLM only loses roughly five points in performance when just 17 (1%) gold standard trees are available, implying that they work quite well even with a limited number of validation trees.

Furthermore, to showcase the data-efficiency of CPE-PLM in few-shot settings, we compare the performance of CPE-PLM and an off-the-shelf supervised parser (Benepar; Kitaev and Klein (2018)) in few-shot settings.[7] From Table 7, we discover that CPE-PLM shows much better performance than the normal parser in extreme cases where few dozen trees are provided. When trees more than 10% of the validation set are available, the supervised parser starts to outperform CPE-PLM.

[7]Specification on training a supervised parser in few-shot settings can be found in Appendix A.

Models	$F1$ w/ ensemble (↑)		Inference time (↓)
	`Greedy`	`Beam`	
Unsupervised parsers/CPE-PLM			
Compound PCFG (Kim et al., 2019b)	55.2		31 min.
CPE-PLM (All PLMs)	55.3	55.7	27 min.
Parsers trained with induced trees			
Distance (Shen et al., 2018a)	53.8	55.0	36 sec.
Benepar (Kitaev and Klein, 2018)	**56.6**	**59.3**	**32 sec.**

Table 8: Training normal parsers with supervision from the trees induced by CPE-PLM. We show that it is viable to build a much faster parser while preserving (or even boosting) the performance of CPE-PLM by relying on existing techniques for supervised parsing.

Issues on the execution time. As identified in Table 8, where we estimate the accuracy and execution time of different approaches on PTB, CPE-PLM and Compound PCFG are still too slow to be readily utilized, compared to supervised counterparts which are generally highly optimized. To relieve this inefficiency, we propose to exploit normal parsers (Shen et al., 2018a; Kitaev and Klein, 2018) by training them with the trees generated by CPE-PLM if a suitable amount of gold annotations are not available for supervision. In Table 8, we demonstrate that it is possible to transfer syntactic knowledge from PLMs to supervised parsers without loss of accuracy, while significantly reducing the execution time at the same time. We even obtain performance gain in some cases, achieving nearly 60 in $F1$ score. We expect that this direction can be particularly useful for low-resource languages for which it is hard to collect gold annotations.

7 Conclusion

In this paper, we introduce two ensemble methods and multi-PLM configurations for Constituency Parse Extraction from Pre-trained Language Models (CPE-PLM). We demonstrate that the performance of CPE-PLM can be competitive with that of unsupervised parsers with the aid of the proposed approaches, and that the parses induced by CPE-PLM are practically useful in several applications where parse trees are required as input. We also propose solutions for mitigating some inherent limitations of CPE-PLM. We anticipate that its potential will be further greater in the near future with the introduction of more sophisticated PLMs.

Acknowledgements

We would like to thank anonymous reviewers for their fruitful feedback. This work was supported by Institute of Information & communications Technology Planning & Evaluation (IITP) grant funded by the Korea government(MSIT) (No.2020-0-01373, Artificial Intelligence Graduate School Program (Hanyang University)). This work was supported by the National Research Foundation of Korea(NRF) grant funded by the Korea government(MSIT) (No.2022R1F1A1074674).

References

Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. 2020. Language models are few-shot learners.

Ethan A. Chi, John Hewitt, and Christopher D. Manning. 2020. Finding universal grammatical relations in multilingual BERT. In *ACL*.

Kevin Clark, Minh-Thang Luong, Quoc V. Le, and Christopher D. Manning. 2020. Electra: Pre-training text encoders as discriminators rather than generators. In *ICLR*.

John Cocke. 1969. *Programming languages and their compilers: Preliminary notes*. New York University.

Alexis Conneau, Kartikay Khandelwal, Naman Goyal, Vishrav Chaudhary, Guillaume Wenzek, Francisco Guzmán, Edouard Grave, Myle Ott, Luke Zettlemoyer, and Veselin Stoyanov. 2020. Unsupervised cross-lingual representation learning at scale. In *ACL*.

Alexis Conneau and Guillaume Lample. 2019. Cross-lingual language model pretraining. In *NeurIPS*.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *NAACL*.

Chris Dyer, Adhiguna Kuncoro, Miguel Ballesteros, and Noah A. Smith. 2016. Recurrent neural network grammars. In *NAACL*.

Alex Graves. 2012. Sequence transduction with recurrent neural networks. *arXiv preprint arXiv:1211.3711*.

John Hewitt and Christopher D Manning. 2019. A structural probe for finding syntax in word representations. In *NAACL*.

Ganesh Jawahar, Benoît Sagot, and Djamé Seddah. 2019. What does BERT learn about the structure of language? In *ACL*.

Jae-young Jo and Sung-Hyon Myaeng. 2020. Roles and utilization of attention heads in transformer-based neural language models. In *ACL*.

Tadao Kasami. 1966. An efficient recognition and syntax-analysis algorithm for context-free languages. *Coordinated Science Laboratory Report no. R-257*.

Nitish Shirish Keskar, Bryan McCann, Lav R Varshney, Caiming Xiong, and Richard Socher. 2019. Ctrl: A conditional transformer language model for controllable generation. *arXiv preprint arXiv:1909.05858*.

Taeuk Kim, Jihun Choi, Daniel Edmiston, Sanghwan Bae, and Sang-goo Lee. 2019a. Dynamic compositionality in recursive neural networks with structure-aware tag representations. In *AAAI*.

Taeuk Kim, Jihun Choi, Daniel Edmiston, and Sang goo Lee. 2020. Are pre-trained language models aware of phrases? simple but strong baselines for grammar induction. In *ICLR*.

Taeuk Kim, Bowen Li, and Sang-goo Lee. 2021. Multilingual chart-based constituency parse extraction from pre-trained language models. In *Findings of the Association for Computational Linguistics: EMNLP 2021*.

Yoon Kim, Chris Dyer, and Alexander Rush. 2019b. Compound probabilistic context-free grammars for grammar induction. In *ACL*.

Yoon Kim, Alexander Rush, Lei Yu, Adhiguna Kuncoro, Chris Dyer, and Gábor Melis. 2019c. Unsupervised recurrent neural network grammars. In *NAACL*.

Nikita Kitaev and Dan Klein. 2018. Constituency parsing with a self-attentive encoder. In *ACL*.

Dan Klein and Christopher D Manning. 2003. Accurate unlexicalized parsing. In *ACL*.

Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Veselin Stoyanov, and Luke Zettlemoyer. 2020. BART: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. In *ACL*.

Bowen Li, Taeuk Kim, Reinald Kim Amplayo, and Frank Keller. 2020. Heads-up! unsupervised constituency parsing via self-attention heads. In *AACL-IJCNLP*.

Xin Li and Dan Roth. 2002. Learning question classifiers. In *COLING*.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Roberta: A robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692*.

Mitchell P. Marcus, Beatrice Santorini, and Mary Ann Marcinkiewicz. 1993. Building a large annotated corpus of English: The Penn Treebank. *Computational Linguistics*.

David Mareček and Rudolf Rosa. 2019. From balustrades to pierre vinken: Looking for syntax in transformer self-attentions. In *Proceedings of the 2019 ACL Workshop BlackboxNLP: Analyzing and Interpreting Neural Networks for NLP*.

Bo Pang and Lillian Lee. 2004. A sentimental education: Sentiment analysis using subjectivity summarization based on minimum cuts. In *ACL*.

Bo Pang and Lillian Lee. 2005. Seeing stars: Exploiting class relationships for sentiment categorization with respect to rating scales. In *ACL*.

Ethan Perez, Douwe Kiela, and Kyunghyun Cho. 2021. True few-shot learning with language models.

Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. 2019. Language models are unsupervised multitask learners.

Anna Rogers, Olga Kovaleva, and Anna Rumshisky. 2020. A primer in BERTology: What we know about how BERT works. *TACL*, 8:842–866.

Rudolf Rosa and David Mareček. 2019. Inducing syntactic trees from bert representations. *arXiv preprint arXiv:1906.11511*.

Djamé Seddah, Reut Tsarfaty, Sandra Kübler, Marie Candito, Jinho D. Choi, Richárd Farkas, Jennifer Foster, Iakes Goenaga, Koldo Gojenola Galletebeitia, Yoav Goldberg, Spence Green, Nizar Habash, Marco Kuhlmann, Wolfgang Maier, Joakim Nivre, Adam Przepiórkowski, Ryan Roth, Wolfgang Seeker, Yannick Versley, Veronika Vincze, Marcin Woliński, Alina Wróblewska, and Eric Villemonte de la Clergerie. 2013. Overview of the SPMRL 2013 shared task: A cross-framework evaluation of parsing morphologically rich languages. In *Proceedings of the Fourth Workshop on Statistical Parsing of Morphologically-Rich Languages*.

Yikang Shen, Zhouhan Lin, Athul Paul Jacob, Alessandro Sordoni, Aaron Courville, and Yoshua Bengio. 2018a. Straight to the tree: Constituency parsing with neural syntactic distance. In *ACL*.

Yikang Shen, Zhouhan Lin, Chin wei Huang, and Aaron Courville. 2018b. Neural language modeling by jointly learning syntax and lexicon. In *ICLR*.

Yikang Shen, Shawn Tan, Alessandro Sordoni, and Aaron Courville. 2019. Ordered neurons: Integrating tree structures into recurrent neural networks. In *ICLR*.

Haoyue Shi, Karen Livescu, and Kevin Gimpel. 2020. On the role of supervision in unsupervised constituency parsing. In *EMNLP*.

Haoyue Shi, Hao Zhou, Jiaze Chen, and Lei Li. 2018. On tree-based neural sentence modeling. In *EMNLP*.

Richard Socher, Alex Perelygin, Jean Wu, Jason Chuang, Christopher D. Manning, Andrew Ng, and Christopher Potts. 2013. Recursive deep models for semantic compositionality over a sentiment treebank. In *EMNLP*.

Ilya Sutskever, Oriol Vinyals, and Quoc V Le. 2014. Sequence to sequence learning with neural networks. In *NeurIPS*.

Kai Sheng Tai, Richard Socher, and Christopher D. Manning. 2015. Improved semantic representations from tree-structured long short-term memory networks. In *ACL-IJCNLP*.

Ian Tenney, Dipanjan Das, and Ellie Pavlick. 2019. BERT rediscovers the classical NLP pipeline. In *ACL*.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *NeurIPS*.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Rémi Louf, Morgan Funtowicz, et al. 2019. Huggingface's transformers: State-of-the-art natural language processing. *arXiv preprint arXiv:1910.03771*.

Zhiyong Wu, Yun Chen, Ben Kao, and Qun Liu. 2020. Perturbed masking: Parameter-free probing for analyzing and interpreting BERT. In *ACL*.

Zhilin Yang, Zihang Dai, Yiming Yang, Jaime Carbonell, Russ R Salakhutdinov, and Quoc V Le. 2019. Xlnet: Generalized autoregressive pretraining for language understanding. In *NeurIPS*.

Daniel H Younger. 1967. Recognition and parsing of context-free languages in time n3. *Information and control*.

Hao Zhu, Yonatan Bisk, and Graham Neubig. 2020. The return of lexical dependencies: Neural lexicalized pcfgs. In *TACL*.

A Appendix: Details on Training Few-shot Parsers

Following Shi et al. (2020), we train a supervised parser (**Benepar**; Kitaev and Klein (2018)) in few-show learning settings to provide a robust baseline for our experiments. To be specific, we leverage the official code and hyperparameters of the parser obtained from `https://github.com/nikitakit/self-attentive-parser`. Given a designated number of parses from the PTB validation set, we utilize 90% of them as the training set while the remaining 10% are used as the real validation set. We train the parser for 100 epochs, similar to Shi et al. (2020). Compared against the experimental results reported from Shi et al. (2020), our few-shot parsers show relatively weaker performance. We conjecture this gap comes from the methods Shi et al. (2020) exploited to boost their performance. For instance, (1) they further pre-trained their word embeddings on sentences from PTB and (2) utilized data augmentation and self-training techniques, all of which are not applied in our case.

Position Offset Label Prediction for Grammatical Error Correction

Xiuyu Wu[1,2] **Jingsong Yu**[1,2] **Xu Sun**[1,3] **Yunfang Wu**[1,3*]
[1]MOE, Key Laboratory of Computational Linguistics, Peking University
[2]School of Software and Microelectronics, Peking University
[3]School of Computer Science, Peking University
{xiuyu_wu,yujingsong,xusun,wuyf}@pku.edu.cn

Abstract

We introduce a novel position offset label prediction subtask to the encoder-decoder architecture for grammatical error correction (GEC) task. To keep the meaning of the input sentence unchanged, only a few words should be inserted or deleted during correction, and most of tokens in the erroneous sentence appear in the paired correct sentence with limited position movement. Inspired by this observation, we design an auxiliary task to predict position offset label (**POL**) of tokens, which is naturally capable of integrating different correction editing operations into a unified framework. Based on the predicted POL, we further propose a new copy mechanism (**P-c**opy) to replace the vanilla copy module. Experimental results on Chinese, English and Japanese datasets demonstrate that our proposed **POL-Pc** framework obviously improves the performance of baseline models. Moreover, our model yields consistent performance gain over various data augmentation methods. Especially, after incorporating synthetic data, our model achieves a 38.95 $F_{0.5}$ score on Chinese GEC dataset, which outperforms the previous state-of-the-art by a wide margin of 1.98 points.

1 Introduction

Grammatical error correction (GEC) is to automatically correct grammatical errors in the input sequence, which is significant for both academic research and practical applications. Comparing with English grammatical error correction, Chinese grammatical error correction (CGEC) is less addressed.

After convolutional neural network and recurrent neural network, the Transformer based sequence-to-sequence models have achieved remarkable performance on GEC task (Junczys-Dowmunt et al., 2018; Kaneko et al., 2020). To alleviate the data sparsity problem, the method of generating pseudo data is intensively studied (Xie et al., 2018; Lichtarge et al., 2019). Several models simplify the GEC task from sequence generation to token-level edit prediction. For example, LaserTagger (Malmi et al., 2019) predicts three edit tags *Keep*, *Delete* and *Append_#* for each token in the source sentence, and GECToR (Omelianchuk et al., 2020) designs ample English-specific tags to predict.

(a) Example I: deletion and substitution

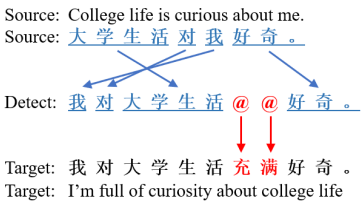

(b) Example II: re-ordering and insertion

Figure 1: Examples of error detection and correction for *CGEC* task. Label @ represents a blank position.

To keep the meaning of the erroneous sentence unchanged, only a few words could be inserted or deleted while most words remain the same. Taking the CGEC task NLPCC data (Zhao et al., 2018) as an example: (1) about 95.6% tokens in erroneous sentences appear in their corresponding correct sentences (namely "cue tokens" for brevity). (2) the average lengths of erroneous sentences and the longest common sub-sequence of error-corrected sentence pairs are 29.65/27.55 tokens, indicating that cue tokens are almost in the same order. (3) the average lengths of inserted and deleted spans are 1.46 and 1.49 tokens, respectively.

Unfortunately, there is no existing model which takes all of these three characteristics into consid-

*Corresponding author.

Proceedings of the 29th International Conference on Computational Linguistics, pages 5409–5418
October 12–17, 2022.

eration. The copy mechanism (Zhao et al., 2019) only considers the characteristic (1). The sequence tagging models (Liang et al., 2020; Malmi et al., 2019) take advantage of Characteristic (3) to build a fixed-size vocabulary for *Append_#* label but fail to explicitly model the first and the second points.

In our work, we merge these three characteristics into one key point: most of tokens appear in both erroneous and correct sentences, and their positions in erroneous sentences (denoted as i^{th}) are identical or close to their positions in the corresponding correct sentences (denoted as j^{th}). As shown in Figure 2, this phenomenon could be observed in GEC datasets of different languages. For instance, on NLPCC dataset, 48.2% of cue tokens are exactly in the same position ($i = j$), and about 90% of them can be moved to the right position within a length of 3 tokens ($abs(i - j) \leq 3$).

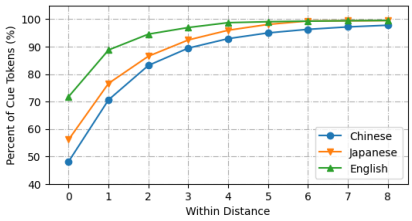

Figure 2: Position movement of cue tokens from erroneous sentences to paired correct sentences in Chinese, Japanese and English GEC datasets, which are described in Section 5.1.

Inspired by the key point summarized above, we propose a novel Position Offset Label (**POL**) prediction task to determine which tokens in the source sentence are error-free, as well as their position movement from the source sequence to the target sentence. As illustrated in Figure 1, our method could simultaneously model different kinds of correction editing operations, including insertion, substitution, deletion and re-ordering. At the decoder, rather than using the conventional attention distribution, a new copy mechanism (**P-copy**) is proposed to take advantage of the result of POL prediction. Overall, we adopt a multi-task learning framework named **POL-Pc**, where the detection network is to perform POL prediction, and the correction network then generates words to fill in the blank positions where no tokens in the source sequence are mapped to.

We conduct extensive experiments on CGEC data NLPCC (Zhao et al., 2018). Experimental results show that without data augmentation, our pure model obtains the best result among all non-pretrained models. After integrating data augmentation methods, our method achieves a new state-of-the-result for CGEC for single models, outperforming the previous best result by 1.98 points. We also conduct experiments on English dataset CoNLL-2014 (Bryant et al., 2019) and Japanese TEC-JL dataset (Koyama et al., 2020), our POL-Pc model consistently improves the performance of the Transformer, validating the generalization ability of our approach. We will make our code publicly available at the GitHub for further research.

To sum, our proposed model for GEC task enjoys the following advantages:

- Comparing with traditional end-to-end methods, our model explicitly separates error detection and error correction, bringing a good interpretability of the neural network.
- Comparing with previous sequence tagging approaches, our model is free to insert any words or phrases without the need to build a specific vocabulary. Moreover, our label strategy is less affected by the imbalance of tag number.
- Our model adopts multi-task learning instead of a pipeline structure to avoid error accumulation.
- Our model outperforms the baseline Transformer by a wide margin for Chinese, Japanese and English GEC tasks. After incorporating data augmentation methods, it obtains a new state-of-the-art result on CGEC task for single models.

2 Related Work

Recently, Transformer-based sequence to sequence models are introduced to GEC task and make great progress (Junczys-Dowmunt et al., 2018; Kaneko et al., 2020). To augment the error-corrected parallel corpus, synthetic data is generated from the Wikipedia revision log (Lichtarge et al., 2019) or generated by applying token-level insertion, substitution, deletion or swapping on error-free corpus (Zhao et al., 2019; Grundkiewicz et al., 2019).

Several models are proposed to simplify GEC task from sequence generation to sequence tagging. LaserTagger (Malmi et al., 2019) predicts *Keep*, *Delete* or *Append_#* tags for each token in the source sentence. PIE (Awasthi et al., 2019)

and GECToR (Omelianchuk et al., 2020) manually design detailed English-specific labels, regarding case and tense, which is hard to adapt to other languages like Chinese. TtT (Li and Shi, 2021) adopts a non-autoregressive model to directly predict each token in the correct sentence. ESD-ESC (Chen et al., 2020) detects erroneous spans and then utilizes a seq2seq model to only produce correct text for those annotated spans. Different from these previous works, we build a multi-task learning framework by treating the sequence tagging as an auxiliary subtask, and more importantly, our tag strategy is novel and effective.

For CGEC task, most of previous works (Wang et al., 2018; Zhang et al., 2020) focus on the spelling error correction task on SIGHAN (Tseng et al., 2015) and Hybrid (Wang et al., 2018) datasets. The NLPCC-2018 dataset (Zhao et al., 2018) provides multiple types of grammatical errors and attracts much attention from participated teams, where the top-3 systems are Alibaba (Zhou et al., 2018), YouDao (Fu et al., 2018) and BLCU (Ren et al., 2018). HRG combines language model base spelling checker, NMT-base model and sequence editing model (Hinson et al., 2020). Later, Zhao and Wang proposes MaskGEC by adding random noises to source sentences dynamically, which achieves the state-of-the-art on NLPCC-2018 dataset. This paper also conducts experiments on NLPCC dataset and considers these methods as comparing baselines.

3 Proposed Model

As shown in Figure 1, given an erroneous sentence $X = (x_1, x_2, ...x_m)$ and its corresponding corrected sentence $Y = (y_1, y_2, ...y_n)$, our model firstly decides whether a source token x_i should be deleted and if not predicts its position changing from X to Y. There might exist some blank positions where no source tokens are mapped to (denoted as @ in Figure 1). A decoder with copy mechanism is utilized to copy source tokens and generate new tokens for blank positions, where the copy operation is guided by the result of position offset label prediction. The overview of our framework is illustrated in Figure 3.

3.1 Position Offset Label Prediction

For each token x_i in the erroneous sentence X , we tag its offset (i.e., the distance between its position in X and Y) as:

$$o_i = \begin{cases} j - i, & x_i = y_j \\ null, & x_i \notin Y \end{cases} \quad (1)$$

where o_i denotes the position offset label. If there exist more than one y_j equaling to x_i , x_i will match the nearest one.

In order to retain the original meaning of the input sentence, the lengths of inserted or deleted spans in correction editing should be as short as possible, and cue tokens' positions in corrected sentences are usually close to their positions in erroneous sentences. Benefiting by this observation, we limit the maximum absolute value of offset o_i by a constant k, to reduce the number of possible categories:

$$o_i = \begin{cases} j - i, & x_i = y_j, abs(i-j) \leq k \\ other, & x_i = y_j, abs(i-j) > k \\ null, & x_i \notin Y \end{cases} \quad (2)$$

According to the statistics in Figure 2, there are over 90% of cue tokens whose offsets are less than or equal to 3 for GEC dataset of any language. We consistently set $k = 3$ for experiments on different languages GEC dataset.

For a clear understanding of the operation of position offset label, Figure 4 presents two examples. We can observe that our designed offset label o_i is able to simultaneously model different edit operations in an elegant way, including insertion, substitution, deletion and re-ordering.

We also carry a statistic comparison of our position offset label with previous sequence tagging methods, as listed in Table 1. The edit label approach predicts *Keep*, *Delete*, *Append_#* or labels representing form transformations of each token (Malmi et al., 2019; Omelianchuk et al., 2020), where one token might have multiple labels (for example, one token might have *Append_#* and tense transformation label simultaneously). The right/wrong label (Zhao et al., 2019; Chen et al., 2020) predicts whether each token in the erroneous sentence should be kept in the correct sentence, which suffers from data imbalance. Compared with them, our strategy avoids all these problems.

We adopt the Transformer encoder to obtain the sequence representation vectors, and then a linear layer and softmax operation are utilized to generate a probability distribution:

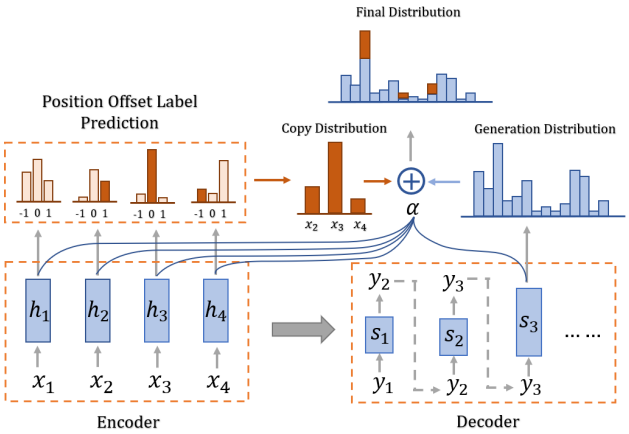

Figure 3: Overview of our POL-Pc framework. For a clear display, we limit the position offset label to $\{-1, 0, 1\}$, which respectively means the source token should be moved to the previous position, be kept in the same position or be moved to the next position.

Source: He probably seems to start bringing glasses four years ago.
Source: 他 可 能 似 乎 四 年 前 开 始 带 眼 镜 。
Label: 0 *N* *N* -2 -2 -2 -2 -2 -2 *N* -2 -2 -2
Detect: 他 似 乎 四 年 前 开 始 @ 眼 镜 。
Detect: He seems to start @ glasses about four years ago.

(a) Example I: Deletion and substitution

Source: College life is curious about me.
Source: 大 学 生 活 对 我 好 奇 。
Label: 2 2 2 2 -3 -5 2 2 2
Detect: 我 对 大 学 生 活 @ @ 好 奇 。
Detect: I'm @ @ curiosity about college life

(b) Example II: Insertion and reorder

Figure 4: Examples of position offset labels for different types of operations. *N* represents *null*. @ represents a blank position.

Tag	# Class	Single label	Maj. (%)
Edit	≥ 500		85.6
Right/Wrong	2	✓	90.2
Position Offset	9	✓	43.5

Table 1: Comparison of different tagging strategies. *Single label* refers to whether each token in the erroneous sentence has only one label. *Maj.* refers to the percentage of the major label that takes up the highest proportion.

$$(h_1^L, h_2^L, ..., h_m^L) = Transformer(x_1, x_2, ..., x_m) \quad (3)$$

$$P_o(o_t|X) = softmax(W_o^T h_t^L + b_o) \quad (4)$$

where W_o and b_o are learned parameters. The loss function for label prediction is defined using log-likelihood:

$$Loss_{off} = -\sum_{t=1}^{m} logP_o(o_t|X) \quad (5)$$

3.2 Copying with Position Offset Label Prediction

We employ a decoder with copy mechanism to generate error-free sentences. Different from conventional copy methods, we propose a new copy mechanism (P-copy) based on the prediction of position offset label.

At timestep t, the generation distribution is computed as:

$$P_t^{gen} = softmax(W_s^T s_t + b_s) \quad (6)$$

where s_t is the hidden state of decoder, W_s and b_s are parameters to learn.

The copy distribution of each token at timestep t is computed by:

$$P_t^{copy}(y_t) = \begin{cases} \sum_{i:x_i=y_t} P_o(o_i = t - i), & y_t \in X \\ 0, & otherwise \end{cases} \quad (7)$$

It means if the source token x_i appears at the tth position in the sequence Y with a probability of $P_o(o_i = t - i)$, the model will copy the token x_i at timestep t with the same probability. We then utilize the softmax function to normalize copy distribution P_t^{copy}. The copy rate α_t is computed using attention mechanism and sigmoid function:

$$Q_t = W_q s_t, K_i = W_k h_i^L, V_i = W_v h_i^L \quad (8)$$

$$a_i = \frac{exp(Q_t K_i^T)}{\sum_j exp(Q_t K_j^T)} \quad (9)$$

$$h_c = \sum_{i=1}^{m} a_i V_i \quad (10)$$

$$\alpha_t = sigmoid(W_c^T h_c + b_c) \quad (11)$$

where W_q, W_k, W_v, W_c and b_c are parameters to learn.

We fuse the generation probability and copy probability to get the final distribution:

$$P_v = (1 - \alpha_t) P_t^{gen} + \alpha_t P_t^{copy} \quad (12)$$

Moreover, the loss function for generation is:

$$Loss_{gen} = -\sum_{t=1}^{n} logP_v(y_t | y_{t-1}, y_{t-2}, ...y_1, X) \quad (13)$$

Combing the generation loss with the loss of POL prediction, the final optimization object of our model is:

$$Loss = Loss_{gen} + \gamma Loss_{off} \quad (14)$$

where γ is a hyper-parameter to balance two loss functions.

4 Data Augmentation

In order to provide more training samples for our model, we do data augmentation by generating synthetic data and introducing dynamic noises. We introduce the following types of noises to produce erroneous sentences.

Deletion: To discard each word with a probability of P_{del}.

Insertion: To insert a [UNK] symbol before each word with a probability of P_{ins}.

Substitution: To replace each word using [UNK] symbol / random token / its homophone with a probability of P_{sub}.

We set $P_{ins} = P_{del} = P_{sub} = 0.25$ in all experiments.

4.1 Synthetic Data

We select THUCnews[1] as the external data which contains 740,000 news documents. We delete the format information such as headlines and bylines in the news corpus and cut the reserved text into 11.4 million single sentences, from which we randomly select 5 million sentences. In our experiment, we introduce only one type of noises into each sentence. Since mixing up synthetic data with training data might dilute human-made corpus and degrade model performance, we only use synthetic data to initialize the parameters of our model by pre-training 10 epoches.

4.2 Dynamic Noise

Inspired by MaskGEC (Zhao and Wang, 2020), we add dynamic noises to input sentences when training on NLPCC dataset to provide more instances. For each human-made erroneous sentence, we randomly choose to keep it unchanged or introduce one type of noises described above.

5 Experimental Setup

5.1 Dataset

We conduct experiments for Chinese, English and Japanese GEC tasks. The details of datasets of different languages is listed in Table 2.

For CGEC task, we choose the dataset of NLPCC 2018 Task 2 (Zhao et al., 2018) , where the training data is collected from the language learning platform Lang-8[2] while the test data is created by teachers. Following the prior work (Zhao and Wang, 2020), we randomly select 5,000 instances from 1.09 million training samples as the development set, and evaluate our model on the official test set containing 2000 sentences. We use BasicTokenizer from BERT project[3] to tokenize the Chinese texts and keep the non-Chinese words unchanged. During evaluation, We tokenize texts with PKUNLP[4] toolkit, which has been used officially at the campaign.

To verify the generalization of our method, we also conduct experiments on English and Japanese GEC tasks. For English GEC task, following Bryant et al. (2019), we use FCE (Yannakoudakis et al., 2011), Lang-8 Corpus of Learner English (Mizumoto et al., 2011), NUCLE (Dahlmeier et al., 2013) and W&I+LOCNESS (Bryant et al., 2019) as training data, CoNLL-2013 test set as dev set and evaluate on CoNLL-2014 (Ng et al., 2014) test set. For Japanese GEC task, we select 1.19 million/5000 sentence pairs as training/dev set from

[1] http://thuctc.thunlp.org/

[2] https://lang-8.com/

[3] https://github.com/google-research/bert

[4] http://59.108.48.12/lcwm/pkunlp/downloads/libgrass-ui.tar.gz

corpora collected from Lang-8 website[5], and evaluate our model on TEC-JL dataset (Koyama et al., 2020).

We adopt M^2 scorer (Dahlmeier and Ng, 2012) as the evaluation tool, where the value of precision, recall and $F_{0.5}$ score is computed.

Language	Train	Dev	Test	Dict	Vocab
Chinese	1.09 M	5,000	2,000	char	13.8 K
Japanese	1.19 M	5,000	1,874	char	6.5 K
English	1.01 M	1,381	1,312	bpe	32 K

Table 2: Data statistics for Chinese, Japanese and English GEC tasks.

5.2 Comparing Methods

We compare our model with the top-3 participated teams of the campaign.

AliGM combines rule-based, SMT-based and NMT-based approaches (Zhou et al., 2018).

YouDao employs five different hybrid models and the final result is selected via a language model (Fu et al., 2018).

BLCU builds a mutli-layer convolutional model with pre-trained embeddings (Ren et al., 2018).

Furthermore, we compare with some of mainstream neural network models for text generation.

Transformer adopts a standard encoder-decoder framework (Vaswani et al., 2017).

Levenshtein-**Transformer** takes insertion and deletion as atomic operations (Gu et al., 2019).

LaserTagger predicts edit operations *Keep*, *Delete* or *Append_#* for each token (Malmi et al., 2019).

ESD-ESC adopts a pipeline structure to firstly detect erroneous spans and then output the correct text for annotated spans (Chen et al., 2020).

Pointer Generator decides to generate a token or copy a token from the source sentence (See et al., 2017), where Transformer is used as the encoder and decoder.

Copy-Augmented introduces copy mechanism into Transformer-based seq2seq framework and employs multi-task learning by predicting whether tokens in input sentence appear in target sentence (Zhao et al., 2019).

HRG proposes a heterogeneous approach composed of language model base spelling checker, NMT-base model and sequence editing model (Hinson et al., 2020).

BERT-fuse incorporates pre-trained BERT model to enhance Transformer (Kaneko et al., 2020).

MaskGEC (Zhao and Wang, 2020) adds random noises to source sentences dynamically.

5.3 Training Details

Our model is implemented using Fairseq [6]. The detail of hyper-parameters of our model is described in Table 3.

Hyper-parameter	Value
encoder layers	6
decoder layers	6
encoder embedding dim	512
decoder embedding dim	512
encoder ffn dim	2048
decoder ffn dim	2048
dropout	0.2
attention dropout	0.1
learning rate	5e-4
Adam β_1	0.9
Adam β_2	0.998
Adam ϵ	1e-8
lr scheduler	inverse_sqrt
warmup updates	8000
max tokens	4096
update-freq	2
loss function	label smoothed cross entropy label-smoothing=0.1

Table 3: Hyper-parameter values of our model.

We average parameters of the last 5 checkpoints. In inference, the beam size is set as 12. In the final loss of Equation 14, we set $\gamma = 0.5$.

6 Results and Discussion

6.1 Main Results for CGEC task

For Chinese GEC task, the overall results of different models are reported in Table 4.

Our pure model POL-Pc yields a 30.64 $F_{0.5}$ score, which beats all the pure models without data augmentation or other extra resources. The Copy-Augmented model reaches 29.85 $F_{0.5}$ by employing the copy mechanism and predicting whether each token in source sentence is kept in the target sentence. Compared with it, our sub-task of POL prediction suffers less from the data imbalance, leading to an improvement of 0.79 $F_{0.5}$ score.

After applying dynamic noises, our model gets a 37.97 $F_{0.5}$ score which outperforms the previous best model MaskGEC. After pretrained on the synthetic data, our model achieves a new state-of-the-art result of 38.95 $F_{0.5}$.

[5]https://sites.google.com/site/naistlang8corpora

[6]https://github.com/pytorch/fairseq

Models	Precision	Recall	$F_{0.5}$
LaserTagger*	25.60	10.50	19.90
Lev-Transformer*	24.90	15.00	22.00
Transformer	34.02	15.44	27.42
Pointer Generator	33.51	16.45	27.75
ESD-ESC*	37.30	14.50	28.40
Copy-Augmented	35.03	18.75	29.85
AliGM*▲♢	41.00	13.75	29.36
YouDao*▲♢	35.24	18.64	29.91
BLCU*▲♢	**47.63**	12.56	30.57
BERT-fuse♢	35.16	23.32	31.92
HRG*▲♢	36.79	**27.82**	34.56
MaskGEC*♡	44.36	22.18	36.97
POL-Pc	36.75	18.39	30.64
POL-Pc + DN♡	44.64	23.77	37.97
POL-Pc + DN + SD♡♢	46.45	23.68	**38.95**

Table 4: Overall performance of different models on NLPCC-2018 test dataset. The model performance with * is from the original published paper and the result of LaserTagger is from (Chen et al., 2020). Other models are re-implemented on our data using the released codes or Fairseq. **SD** and **DN** refer to synthetic data and dynamic noise described in Section 4. The symbols ▲ / ♡ / ♢ denote ensemble / data augmentation / pre-training approaches respectively.

Models	NLPCC	TEC-JL	CoNLL-14
Lev-Transformer	22.00	16.21	42.48
Pointer Generator	27.75	26.62	49.99
Transformer	27.42	26.11	48.63
POL-Pc	30.64	28.03	50.56
vs. Transformer	+11.7%	+7.4%	+4.0%

Table 5: Performance on Chinese, Japanese and English GEC datasets.

Data Augmentation	Transformer	POL-Pc	Imp.
None	27.42	30.64	+3.22
MaskGEC	36.97	37.26	+0.29
Dynamic Noise	37.02	37.97	+0.95
Synthetic Data (SD)	32.81	33.37	+0.56
SD + MaskGEC	37.83	38.21	+0.38
SD + Dynamic Noise	37.71	38.95	+1.24

Table 6: Performance of our model after incorporating different data augmentation methods on NLPCC-2018 dataset. *Imp* refers to improvement.

6.2 Results for Various Language GEC tasks

In addition to Chinese, Table 5 reports the experimental results on English and Japanese datasets. Compared with the baseline Transformer, our pure model brings a relative performance gain of 11.7%, 7.4% and 4.0% on Chinese, Japanese and English datasets respectively, validating the effectiveness of our approach for different language GEC tasks.

It also demonstrates that our model is more suitable for Chinese and Japanese GEC task. Both Chinese and Japanese language utilize function words instead of affixes to represent forms and tenses, which leads to more insertion and deletion operation when correcting grammatical errors. In contrast, the substitution operation such as shifting of tenses is the main operation for English GEC task. As demonstrated in Figure 2, tokens in English GEC dataset are more likely kept in the exactly same position. Besides, English word are divided into subwords by BPE, which makes POL prediction module hard to train. As a result, the prediction of token position movement is more beneficial for Chinese and Japanese GEC tasks.

6.3 Results with Different Data Augmentation Methods

Since data augmentation is intensively studied for GEC, we implement our model trained on different synthetic data, and the experimental results are shown in Table 6. Our model consistently outperforms Transformer utilizing different data augmentation approaches. Compared with MaskGEC which benefits both Transformer and our POL-Pc model, our dynamic noise, as described in Section 4, brings more improvement to POL-Pc. It demonstrates that providing more training samples with different position movement generated by deletion and insertion operations could further improve the performance of our model.

6.4 Results of POL Prediction on Different Model Architectures

To further evaluate the effectiveness of position offset label prediction for GEC task, we apply this module to other model architectures, including the standard Transformer, Pointer-Generator and Copy-augmented models. We evaluate their performance on NLPCC test dataset. As shown in Table 7, adding POL prediction as one of jointly-training tasks consistently improves the performance of different baseline models.

It is noteworthy that combining the POL prediction sub-task with Pointer Generator reaches 29.57 $F_{0.5}$ score, which is lower than our pure model ($F_{0.5} = 30.64$). Because our model adopts the probability of POL prediction as the copy score, which is more effective than making POL prediction and copy mechanism work separately.

Model	Precision	Recall	$F_{0.5}$	Imp.
Transformer	34.02	15.44	27.42	-
+ *POL*	32.59	18.14	28.11	+0.69
Pointer Generator	33.51	16.45	27.75	-
+ *POL*	36.22	17.05	29.57	+1.82
Copy-Augmented	35.03	18.75	29.85	-
+ *POL*	37.89	16.89	30.34	+0.49

Table 7: Experimental results of POL prediction based on different model architectures NLPCC-2018 dataset.

Model	Precision	Recall	$F_{0.5}$	Imp.
POL-Pc	36.75	18.39	30.64	-
- *P-copy*	32.59	18.14	28.11	-2.53
- *(POL + P-copy)*	34.02	15.44	27.42	-3.22
POL-Pc+DN+SD	46.45	23.68	38.95	-
- *P-copy*	45.87	23.23	38.39	-0.56
- *(POL + P-copy)*	45.05	22.84	37.71	-1.24
- *Synthetic Data*	44.64	23.77	37.97	-0.98
- *Dynamic Noise*	43.43	17.32	33.37	-4.58

Table 8: Ablation study on NLPCC dataset. *POL* refers to position offset label prediction, and *P-copy* refers to our POL-based copy mechanism. **DN** refers to dynamic noises and **SD** refers to synthetic data.

6.5 Ablation Study

We do ablation study on NLPCC dataset to evaluate the effect of each module, and list the results in Table 8. On the pure model, removing the POL-based copy mechanism results in a 2.53 decrease, and removing both POL prediction and copy modules leads to a sharp decrease of 3.22 points, which proves that our proposed copy mechanism is a suitable way to take advantage of the results of POL prediction.

Our full model greatly benefits from data augmentation with a 8.31 increase in $F_{0.5}$. For the full model, removing the copy module causes a decrease of 0.56 $F_{0.5}$ score. Removing both POL prediction and copy modules results in a drop of 1.24 $F_{0.5}$ score, which shows that our proposed module can improve model performance even after a huge amount of pseudo data being incorporated. Discarding the synthetic data, the performance decreases from 38.95 to 37.97, suggesting that using synthetic data for pre-training provides the model with better initial parameters.

6.6 Hyper-parameter Setting

In this section, we explore the effect of hyper-parameter γ in Equation 14. As illustrated in Figure 5, $F_{0.5}$ score shows a single-peaked pattern as γ increases. In most cases, our pure model surpasses the basic Transformer model and achieves the best performance when γ is 0.5. When γ decreases, the model fails to explicitly predict the position offset label guided by ground-truth labels, making the model gradually degrade to the vanilla copy mechanism, and when γ increases, the model pays less attention to the generation ability of decoder which also harms the performance.

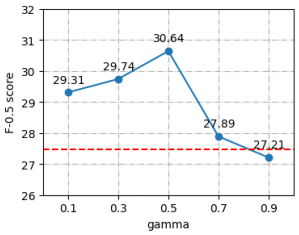

Figure 5: The effect of hyper-parameter γ on multi-task learning based on the standard Transformer model. The red line refers to the performance of basic model on NLPCC dataset.

6.7 Case Study

To investigate how POL prediction and the correction network work together, we select two examples from the NLPCC test data to visualize the intermediate result and final output. As illustrated in Figure 6, in both examples, most of tokens are predicted to be moved to the right position. For tokens which should be deleted or substituted, our model predicts their probability of moving to any position to be low, which means the probability of deletion $P_o(o_i = null)$ is high. Moreover, in the first example, the copy rate of tokens not appearing in the target sentence is low, which shows our copy mechanism works effectively with respect to different offset labels. These two modules in our framework can accomplish their respective tasks and cooperate smoothly with each other.

7 Conclusion

We introduce a detection network to predict POL of tokens between source erroneous sentences and target correct sentences. Based on the output of POL, we design a new copy mechanism P-copy. Our POL-Pc model exceeds both end-to-end models and sequence tagging approaches, achieving a new state-of-the-art result on CGEC task for single models. For English and Japanese GEC tasks, our approach also obtains significant performance gain over the baseline Transformer.

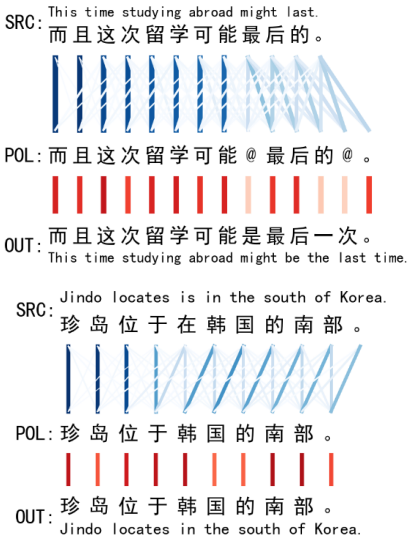

Figure 6: Visualization of POL prediction and copy mechanism. The color blue represents the probability of POL, and the color red represents the copy rate α_t defined in Equation 11. The color becomes darker as the value gets bigger. Label @ represents a blank position.

Acknowledgement

This work is supported by National Key Research and Development Program of China (2020AAA0109703), the National Natural Science Foundation of China (62076008) and the Key Project of Natural Science Foundation of China (61936012).

References

Abhijeet Awasthi, Sunita Sarawagi, Rasna Goyal, Sabyasachi Ghosh, and Vihari Piratla. 2019. Parallel iterative edit models for local sequence transduction. In *EMNLP/IJCNLP (1)*, pages 4259–4269. Association for Computational Linguistics.

Christopher Bryant, Mariano Felice, Øistein E. Andersen, and Ted Briscoe. 2019. The bea-2019 shared task on grammatical error correction. In *BEA@ACL*.

Mengyun Chen, Tao Ge, Xingxing Zhang, Furu Wei, and Ming Zhou. 2020. Improving the efficiency of grammatical error correction with erroneous span detection and correction. In *EMNLP (1)*, pages 7162–7169. Association for Computational Linguistics.

Daniel Dahlmeier and Hwee Tou Ng. 2012. Better evaluation for grammatical error correction. In *HLT-NAACL*, pages 568–572. The Association for Computational Linguistics.

Daniel Dahlmeier, Hwee Tou Ng, and Siew Mei Wu. 2013. Building a large annotated corpus of learner english: The nus corpus of learner english. In *BEA@NAACL-HLT*.

Kai Fu, Jin Huang, and Yitao Duan. 2018. Youdao's winning solution to the nlpcc-2018 task 2 challenge: A neural machine translation approach to chinese grammatical error correction. In *NLPCC (1)*, volume 11108 of *Lecture Notes in Computer Science*, pages 341–350. Springer.

Roman Grundkiewicz, Marcin Junczys-Dowmunt, and Kenneth Heafield. 2019. Neural grammatical error correction systems with unsupervised pre-training on synthetic data. In *BEA@ACL*, pages 252–263. Association for Computational Linguistics.

Jiatao Gu, Changhan Wang, and Jake Zhao. 2019. Levenshtein transformer. In *NeurIPS*.

Charles Hinson, Hen-Hsen Huang, and Hsin-Hsi Chen. 2020. Heterogeneous recycle generation for chinese grammatical error correction. In *Proceedings of the 28th International Conference on Computational Linguistics, COLING 2020, Barcelona, Spain (Online), December 8-13, 2020*, pages 2191–2201. International Committee on Computational Linguistics.

Marcin Junczys-Dowmunt, Roman Grundkiewicz, Shubha Guha, and Kenneth Heafield. 2018. Approaching neural grammatical error correction as a low-resource machine translation task. In *NAACL-HLT*, pages 595–606. Association for Computational Linguistics.

Masahiro Kaneko, Masato Mita, Shun Kiyono, Jun Suzuki, and Kentaro Inui. 2020. Encoder-decoder models can benefit from pre-trained masked language models in grammatical error correction. In *ACL*, pages 4248–4254. Association for Computational Linguistics.

Aomi Koyama, Tomoshige Kiyuna, Kenji Kobayashi, Mio Arai, and Mamoru Komachi. 2020. Construction of an evaluation corpus for grammatical error correction for learners of japanese as a second language. In *LREC*.

Piji Li and Shuming Shi. 2021. Tail-to-tail non-autoregressive sequence prediction for chinese grammatical error correction. In *ACL/IJCNLP*.

Deng Liang, Chen Zheng, Lei Guo, Xin Cui, Xiuzhang Xiong, Hengqiao Rong, and Jinpeng Dong. 2020. BERT enhanced neural machine translation and sequence tagging model for Chinese grammatical error diagnosis. In *Proceedings of the 6th Workshop on Natural Language Processing Techniques for Educational Applications*, pages 57–66, Suzhou, China. Association for Computational Linguistics.

Jared Lichtarge, Chris Alberti, Shankar Kumar, Noam Shazeer, Niki Parmar, and Simon Tong. 2019. Corpora generation for grammatical error correction. In *NAACL-HLT (1)*, pages 3291–3301. Association for Computational Linguistics.

Eric Malmi, Sebastian Krause, Sascha Rothe, Daniil Mirylenka, and Aliaksei Severyn. 2019. Encode, tag, realize: High-precision text editing. In

EMNLP/IJCNLP (1), pages 5053–5064. Association for Computational Linguistics.

Tomoya Mizumoto, Mamoru Komachi, Masaaki Nagata, and Yuji Matsumoto. 2011. Mining revision log of language learning sns for automated japanese error correction of second language learners. In *IJCNLP*, pages 147–155. The Association for Computer Linguistics.

Hwee Tou Ng, Siew Mei Wu, Ted Briscoe, Christian Hadiwinoto, Raymond Hendy Susanto, and Christopher Bryant. 2014. The conll-2014 shared task on grammatical error correction.

Kostiantyn Omelianchuk, Vitaliy Atrasevych, Artem N. Chernodub, and Oleksandr Skurzhanskyi. 2020. Gector - grammatical error correction: Tag, not rewrite. In *BEA@ACL*, pages 163–170. Association for Computational Linguistics.

Hongkai Ren, Liner Yang, and Endong Xun. 2018. A sequence to sequence learning for chinese grammatical error correction. In *NLPCC (2)*, volume 11109 of *Lecture Notes in Computer Science*, pages 401–410. Springer.

Abigail See, Peter J. Liu, and Christopher D. Manning. 2017. Get to the point: Summarization with pointer-generator networks. In *ACL (1)*, pages 1073–1083. Association for Computational Linguistics.

Yuen-Hsien Tseng, Lung-Hao Lee, Li-Ping Chang, and Hsin-Hsi Chen. 2015. Introduction to sighan 2015 bake-off for chinese spelling check. In *SIGHAN@IJCNLP*, pages 32–37. Association for Computational Linguistics.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in Neural Information Processing Systems 30*, page 5998–6008. Curran Associates, Inc.

Dingmin Wang, Yan Song, Jing Li, Jialong Han, and Haisong Zhang. 2018. A hybrid approach to automatic corpus generation for chinese spelling check. In *EMNLP*, pages 2517–2527. Association for Computational Linguistics.

Ziang Xie, Guillaume Genthial, Stanley Xie, Andrew Y. Ng, and Dan Jurafsky. 2018. Noising and denoising natural language: Diverse backtranslation for grammar correction. pages 619–628. Association for Computational Linguistics.

Helen Yannakoudakis, Ted Briscoe, and Ben Medlock. 2011. A new dataset and method for automatically grading esol texts. In *ACL*.

Shaohua Zhang, Haoran Huang, Jicong Liu, and Hang Li. 2020. Spelling error correction with soft-masked bert. In *ACL*, pages 882–890. Association for Computational Linguistics.

Wei Zhao, Liang Wang, Kewei Shen, Ruoyu Jia, and Jingming Liu. 2019. Improving grammatical error correction via pre-training a copy-augmented architecture with unlabeled data. In *NAACL-HLT (1)*, pages 156–165. Association for Computational Linguistics.

Yuanyuan Zhao, Nan Jiang, Weiwei Sun, and Xiaojun Wan. 2018. Overview of the nlpcc 2018 shared task: Grammatical error correction. In *CCF International Conference on Natural Language Processing and Chinese Computing*, pages 439–445. Springer.

Zewei Zhao and Houfeng Wang. 2020. Maskgec: Improving neural grammatical error correction via dynamic masking. In *AAAI*, pages 1226–1233. AAAI Press.

Junpei Zhou, Chen Li, Hengyou Liu, Zuyi Bao, Guangwei Xu, and Linlin Li. 2018. Chinese grammatical error correction using statistical and neural models. In *NLPCC (2)*, volume 11109 of *Lecture Notes in Computer Science*, pages 117–128. Springer.

Parsing Natural Language into Propositional and First-Order Logic with Dual Reinforcement Learning

Xuantao Lu[♣], **Jingping Liu**[♠*], **Zhouhong Gu**[♣], **Hanwen Tong**[♣],
Chenhao Xie[★], **Junyang Huang**[♣], **Yanghua Xiao**[♣♡*], **Wenguang Wang**[◇],
[♣]Shanghai Key Laboratory of Data Science, School of Computer Science, Fudan University
[★]SenseDeal Intelligent Technology Co., Ltd., Beijing, China
[♠]School of Information Science and Engineering, East China University of Science and Technology
[♡]Fudan-Aishu Cognitive Intelligence Joint Research Center [◇]DataGrand Inc., Shanghai, China
{xtlu20, shawyh}@fudan.edu.cn, jingpingliu@ecust.edu.cn,
xiechenhao@sensedeal.ai

Abstract

Semantic parsing converts natural language utterances into structured logical expressions. We consider two such formal representations: Propositional Logic (PL) and First-order Logic (FOL). The paucity of labeled data is a major challenge in this field. In previous works, dual reinforcement learning has been proposed as an approach to reduce dependence on labeled data. However, this method has the following limitations: 1) The reward needs to be set manually and is not applicable to all kinds of logical expressions. 2) The training process easily collapses when models are trained with only the reward from dual reinforcement learning. In this paper, we propose a scoring model to automatically learn a model-based reward, and an effective training strategy based on curriculum learning is further proposed to stabilize the training process. In addition to the technical contribution, a Chinese-PL/FOL dataset is constructed to compensate for the paucity of labeled data in this field. Experimental results show that the proposed method outperforms competitors on several datasets. Furthermore, by introducing PL/FOL generated by our model, the performance of existing Natural Language Inference (NLI) models is further enhanced.

1 Introduction

Semantic parsing is the task of mapping natural language utterances into logical expressions. As two major logical forms of text representation, Propositional Logic (PL) and First-order Logic (FOL) play an increasingly important role in a wide range of downstream tasks including Inductive Logic Programming (ILP) (Yang and Song, 2019), Question Answering (QA) (Longo and Santoro, 2020) and Interpretable Reinforcement Learning (Ma et al., 2020; Kimura et al., 2021) because they are capable of discovering and representing knowledge in an explicit symbolic structure that can be understood and examined by human (Evans and Grefenstette, 2018). It is worth noting that all of these tasks have a prerequisite, i.e., parsing natural language utterances into PL/FOL. The results of parsing directly affect the performance of downstream tasks. Thus, it is crucial to have a strong semantic parser for PL/FOL.

English-FOL
Natural Language Utterance
Some volunteers include executives and professionals.
Logical Expression
exists x1.(_volunteer(x1) & exists x2.
(_executive(x2) & _professional(x2) & _include(x1, x2)))

Table 1: Example of the English-FOL dataset. Text in olive denotes quantifiers, and text in teal denotes predicates. $x1$ and $x2$ denote variables, and & denotes logical connectives.

Some solutions have been proposed to parse natural language into PL/FOL. One of the most typical methods is to model the parsing task as a sequence-to-sequence (Seq2seq) generation problem, including using a character-level recurrent neural network (Levkovskyi and Li, 2021) and introducing a variable alignment mechanism (Singh et al., 2020). However, these approaches have the following issues. First, a large amount of labeled data is required for these approaches to achieve good results, which inevitably suffers from the paucity of labeled data in this field. One of the solutions is to generate labeled data by templates (Levkovskyi and Li, 2021), but it leads to a lack of diversity in the data which makes the model prone to overfitting the training data. Second, previous works (Singh et al., 2020; Levkovskyi and Li, 2021) only consider unidirectional generation (from utterance to PL/FOL), while intuitively bidirectional generation can further enhance the performance of the models.

* Corresponding Author

Proceedings of the 29th International Conference on Computational Linguistics, pages 5419–5431
October 12–17, 2022.

Regarding these issues, we propose an effective framework for parsing natural language into PL/FOL named **Dual-(m)T5**, and introduce unlabeled data to alleviate the impact of insufficient labeled data. Inspired by He et al. (2016); Cao et al. (2019), we model the learning of logical expressions and natural language generation as dual tasks. Both of the tasks are jointly trained via reinforcement learning (RL) since the training process is non-differentiable. However, we encounter the following challenges when applying dual reinforcement learning to PL/FOL: 1) The validity reward in dual reinforcement learning is rule-based (Cao et al., 2019), which results in a lot of manual attempts to get an effective one, and needs to be redesigned when the type of logical expressions changes (e.g. from lambda-calculus to FOL) since it is customized for a specific type of logical expression. 2) The effectiveness of the validity reward needs to be improved because it only considers a lexical-level matching between utterances and logical expressions, and cannot achieve deep semantic matching. 3) An effective training strategy needs to be explored since the training process easily collapses when the model is trained with only the rewards from dual reinforcement learning.

To address the first two issues mentioned above, we propose a scoring model to automatically learn a model-based validity reward that is applicable to various types of logical expressions. The scoring model aims to evaluate whether the semantics of utterances and logical expressions match. For issue 3, we propose an effective training strategy to stabilize the training process. Specifically, curriculum learning (Bengio et al., 2009) is employed to initialize the parameters of the model, aiming to get the model in a good initial state, and unlabeled data is introduced to prevent models from crashes due to insufficient labeled data. Experimental results on different datasets show that our framework effectively parses natural language into PL/FOL and consistently improves performance compared to competitors. To further demonstrate the value of PL/FOL, we take Natural Language Inference (NLI) as a downstream task, and the performance of existing NLI models is further enhanced by introducing PL/FOL generated by our model.

Our contributions are three-folds:

- We propose a dual reinforcement learning framework called **Dual-(m)T5** with a novel model-based validity reward that fully considers the semantics of utterances and logical expressions and is applicable to all types of logical expressions. We further propose an effective training strategy to stabilize the training process of dual reinforcement learning.

- We release a new dataset called **Chinese-PL/FOL** that contains 1,263 Chinese-PL pairs and 1,464 Chinese-FOL pairs to compensate for the paucity of labeled data in this field.

- Experimental results show that the proposed method outperforms competitors on several datasets. Furthermore, by introducing additional logical expressions generated by **Dual-(m)T5**, the performance of existing NLI models is further enhanced.

2 Overview

In this section, we formalize the problem and outline our framework.

2.1 Problem Definition

As shown in Table 1, given a natural language sentence s, the goal of this paper is to generate the corresponding logical expression e. For example, given a sentence "Some volunteers include executives and professionals.", an ideal model would generate a logical expression "exists x1.(_volunteer(x1) & exists x2. (_executive(x2) & _professional(x2) & _include(x1, x2)))".

2.2 Framework

The overview of our framework is shown in Figure 1. The backbone of the framework is dual reinforcement learning, which consists of two sub-modules: The prime module generates a logical expression given a natural language sentence, while the dual module produces a sentence given a logical expression. The scoring model is used to obtain the validity reward, and the reconstruction reward is used to force the generated sentence in the dual module as similar to the original sentence as possible. To stabilize the training process, models are pre-trained before dual reinforcement learning, and curriculum learning is employed to get the model in a good initial state.

3 Methodology

In this section, we first present the details of dual reinforcement learning in § 3.1. The scoring model

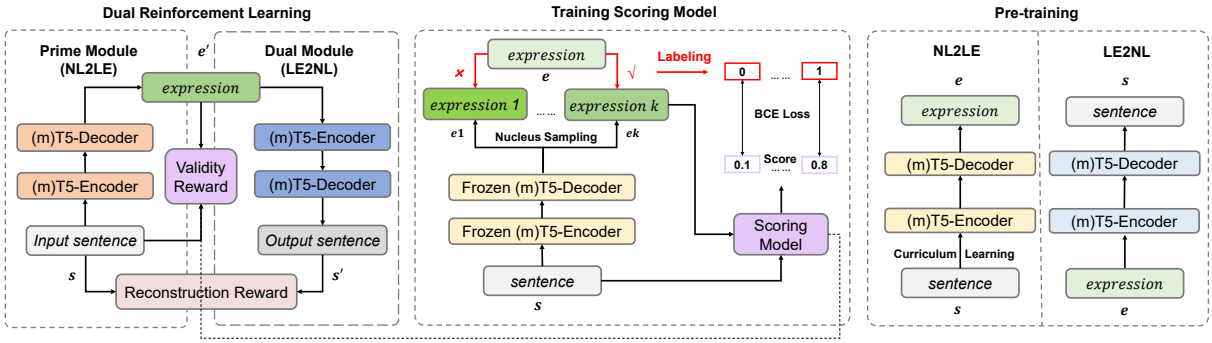

Figure 1: An overview of the framework. Dual reinforcement learning consists of NL2LE and LE2NL, and the validity reward is learned by the scoring model. The (m)T5s used in the scoring model and dual reinforcement learning are initialized by pre-training. The sentence s and expression e represent a ground truth pair.

is introduced in § 3.2, and the training strategy is provided in § 3.3.

3.1 Backbone: Dual Reinforcement Learning

The backbone of our framework is dual reinforcement learning which consists of two sub-modules: **N**atural **L**anguage to **L**ogical **E**xpression (NL2LE) and **L**ogical **E**xpression to **N**atural **L**anguage (LE2NL). Both of the modules adopt T5 (Raffel et al., 2019) / mT5 (Xue et al., 2021), a (multilingual) pre-trained text-to-text transformer as the backbone. These two modules in a closed-loop are trained by a reinforcement learning (RL) method based on policy gradient (Sutton et al., 2000). In RL, the state is denoted by the input of the prime module, i.e., sentence s. The action in the prime and dual modules is defined as the logical expression and sentence generation, respectively. The policy is denoted as the parameters of the (m)T5 models in the two modules.

Prime Module (NL2LE) aims to transform natural language into PL/FOL. Specifically, given a sentence s, the NL2LE model could generate k possible logical expressions $e_1, e_2, \cdots, e_k$ via nucleus sampling (Holtzman et al., 2020). Then, the scoring model scores the generated logical expressions and obtains a validity reward $R_{val}\left(e_i \mid s\right)$ for each logical expression e_i. The details of the scoring model will be introduced in § 3.2.

Dual Module (LE2NL) is an inverse of the prime module, which aims to generate sentences given PL/FOL. Formally, the input is the logical expression e_i generated in the prime task, and the model is expected to output the original sentence s. Reconstruction reward is used to estimate the similarity between the input of the prime model and the output of the dual model. Let Θ_{NL2LE} and Θ_{LE2NL} denote all the parameters of NL2LE and LE2NL, respectively. The reconstruction reward is formulated as:

$$R_{rec}\left(s \mid e_i\right) = \log P\left(s \mid e_i; \Theta_{LE2NL}\right) \quad (1)$$

Learning Algorithm By utilizing policy gradient (Sutton et al., 2000), the stochastic gradients of Θ_{NL2LE} and Θ_{LE2NL} are computed as:

$$\nabla_{\Theta_{NL2LE}} \mathbb{E}[r] = \frac{1}{k} \sum_{i=1}^{k} r_i \cdot g_i \quad (2)$$

$$r_i = \alpha R_{val}\left(e_i \mid s\right) + (1-\alpha) R_{rec}\left(s \mid e_i\right) \quad (3)$$

$$g_i = \nabla_{\Theta_{NL2LE}} \log P\left(e_i \mid s; \Theta_{NL2LE}\right) \quad (4)$$

$$\nabla_{\Theta_{LE2NL}} \mathbb{E}[r] = \frac{1-\alpha}{k} \sum_{i=1}^{k} g_i' \quad (5)$$

$$g_i' = \nabla_{\Theta_{LE2NL}} \log P\left(s \mid e_i; \Theta_{LE2NL}\right) \quad (6)$$

where a hyper-parameter $\alpha \in [0, 1]$ is exploited to balance between R_{val} and R_{rec}.

3.2 Scoring Model

The scoring model is used to evaluate whether the semantics of the generated logical expression is consistent with the semantics of the input sentence and calculate the validity reward R_{val}. Formally, given a $\langle s, e \rangle$ pair, the scoring model outputs $P\left(e \mid s\right) \in [0, 1]$, which represents the correlation between s and e. Intuitively, we take $\{\langle s_i, e_i \rangle\}$ pairs from the supervised dataset $\mathcal{L}$ as positive samples $\mathcal{P}$ to train such a scoring model. The challenge is that off-the-shelf negative samples are not available. Negative sampling, i.e., sampling $\{\langle s_i, e_j \rangle\}_{i \neq j}$ pairs from $\mathcal{L}$ as negative samples is an optional solution, but the quality of negative samples obtained in this way are not challenging

for the scoring model. To get enough hard negative samples to train the scoring model, we design the following approach:

First, an NL2LE model is pre-trained with the supervised dataset $\mathcal{L}$ (details in § 3.3.1). After that, given a sentence s, the NL2LE model could generate k possible logical expressions $\bar{e}_1, \bar{e}_2, \cdots, \bar{e}_k$ via nucleus sampling (Holtzman et al., 2020). We denote $\{\langle s, \bar{e}_i \rangle\}$ as negative samples $\mathcal{N}$, where the logical expression $\bar{e}_i$ is not equal to ground truth e. Since the NL2LE model has been pre-trained, $\bar{e}_i$ will be similar to the ground truth e. These hard negative samples will challenge the scoring model and enable it to learn the effects of small differences in logical expressions. For each $\langle s_i, e_i \rangle \in \mathcal{P} \cup \mathcal{N}$, we take the last layer hidden states of the NL2LE model's encoder $h_{i1}^{enc}, \cdots, h_{in}^{enc}$ and decoder $h_{i1}^{dec}, \cdots, h_{im}^{dec}$ as the feature of s_i and e_i respectively. Then, the scoring model is defined as follows:

$$\overline{h}_i^{enc} = \frac{1}{n}\sum_{j=1}^{n} h_{ij}^{enc}, \quad \overline{h}_i^{dec} = \frac{1}{m}\sum_{j=1}^{m} h_{ij}^{dec} \tag{7}$$

$$u_i = \overline{h}_i^{enc} \cdot W_1 + b_1, \quad v_i = \overline{h}_i^{dec} \cdot W_2 + b_2 \tag{8}$$

$$P(e_i \mid s_i) = \sigma\left([u_i; v_i; |u_i - v_i|] \cdot W_3 + b_3\right) \tag{9}$$

where $W_{1|2|3}$ and $b_{1|2|3}$ are trainable parameters. $[\cdot;\cdot]$ is the concatenation operation, and σ represents sigmoid function. The training loss L of the scoring model is binary cross-entropy (BCE) loss between the model's output $P(e_i \mid s_i)$ and labels,

$$L = -\frac{1}{|\mathcal{P} \cup \mathcal{N}|}\left(\sum_{\langle s_i, e_i \rangle \in \mathcal{P}} \log P(e_i \mid s_i) + \sum_{\langle s_i, e_i \rangle \in \mathcal{N}} (1 - \log P(e_i \mid s_i))\right) \tag{10}$$

Note that the parameters of the NL2LE model are fixed, and only the scoring model is updated during the backpropagation. Finally, we take $P(e_i \mid s)$ as the validity reward $R_{val}(e_i \mid s)$.

3.3 Training Strategy

In this section, we will introduce the training strategy of our framework. The entire training process consists of two stages: pre-training and dual reinforcement learning. For pre-training, we explore how to integrate curriculum learning into the training phase in § 3.3.1. For dual reinforcement learning, we explore how to construct and introduce unlabeled data in § 3.3.2, and make the training stable in § 3.3.3.

3.3.1 Pre-training with Curriculum Learning

The pre-training of the NL2LE model aims to maximize the likelihood $p(e|s)$ for each $\langle s, e \rangle$ pair from the supervised dataset $\mathcal{L}$. According to the learning principle of human beings in the cognitive process, we should start with simple samples and gradually consider more complex samples. To this end, we employ curriculum learning (Bengio et al., 2009) to determine the training order. Here, we take the length of logic expressions as an indicator of the training order, i.e., the longer the logical expression, the more difficult it is. We first sort the training samples according to the length of the logical expressions. At each training step t, a batch of training samples is obtained from the top $f(t)$ portions of the entire sorted training samples. Following Platanios et al. (2019), $f(t)$ is defined as:

$$f(t) = \min\left(1, \sqrt{\frac{t\left(1 - c_0^2\right)}{T} + c_0^2}\right) \tag{11}$$

where c_0 represents the models start training using the $c_0\%$ easiest training samples, and T represents the duration of curriculum learning.[1]

3.3.2 Introducing Unlabeled Data for Dual Reinforcement Learning

Since the training process of dual reinforcement learning only leverages natural language utterance and does not need the corresponding logical expression, in addition to using utterances from the supervised dataset $\mathcal{L}$, we further improve the performance of the models by introducing unlabeled utterances $\mathcal{U}$.

Different from the previous work (Cao et al., 2019) where unlabeled data is constructed by manually defined rules, we leverage off-the-shelf paraphrase generation models (see in Appendix.D) which generate synonymous sentences from existing utterances in the supervised dataset $\mathcal{L}$. According to our observation, since the paraphrase generation models are trained on paraphrase generation datasets that are different from the datasets we use, the generated synonymous sentence is not particularly similar to the original one, and the logical expressions corresponding to the two sentences are different in most cases. Thus, it is reasonable

[1] In practice, curriculum learning has no effect on LE2NL, so we only apply it to NL2LE.

	# PL	# FOL	TOTAL
Training	871	1,037	1,908
Validation	128	145	273
Test	264	282	546
Total	1,263	1,464	2,727

Table 2: Statistics of the Chinese-PL/FOL dataset.

to treat the generated sentences as unlabeled data. The experiments in § 5 also demonstrate the effectiveness of this method.

3.3.3 Stable Dual Reinforcement Learning

In practice, we find that the training process easily collapses when the models are trained with only the rewards from dual reinforcement learning. To keep the training stable and prevent the models from crashing, we adopt the following method:

Introducing Supervisor We pre-train both of the models with the supervised dataset $\mathcal{L}$ before dual reinforcement learning starts (the pre-training of NL2LE refers to § 3.3.1). Moreover, after each update according to Eq.(2) and Eq.(5), the models are trained with the labeled data again, i.e., both of the models are trained with dual reinforcement learning and supervised learning alternately.

Reward Baseline To cope with high variance in reward signals, we generate k intermediate outputs as mentioned in § 3.1 and re-define reward signals by introducing a reward baseline to stabilize the training process. Here, we take the average of rewards within samples per input as the reward baseline. Thus, the final validity reward R'_{val} and reconstruction reward R'_{rec} are as follows:

$$R'_{val}(e_i \mid s) = R_{val}(e_i \mid s) - \frac{1}{k}\sum_{i=1}^{k} R_{val}(e_i \mid s)$$

$$R'_{rec}(s \mid e_i) = R_{rec}(s \mid e_i) - \frac{1}{k}\sum_{i=1}^{k} R_{rec}(s \mid e_i)$$

4 Dataset Collection

To compensate for the paucity of labeled data in this field and verify the effectiveness of our proposed framework, we construct a dataset containing natural language and PL/FOL pairs. In the previous work (Levkovskyi and Li, 2021), the authors define templates first and then obtain samples by filling slots. However, the resulting dataset is limited by the lack of diversity of templates. Crowdsourcing is another option but is not applicable for this task, since professional knowledge about PL/FOL is required. Therefore, we use expert annotation to ensure the quality and diversity of the dataset.

The annotation team consists of 8 Chinese graduate students who are familiar with PL/FOL. If the annotators are required to construct data without any reference, this will introduce inevitable troubles and labeling errors, since PL/FOL is not intuitive to humans. Aiming to reduce nontrivial human labor and ensure the quality of the dataset, the data collection process consists of the following steps: We first obtain PL/FOL exercise sets and exam papers that require students to convert natural language into PL/FOL from Baidu Wenku[2]. Then, the annotators are asked to organize these exercises in a uniform format. Each sample consists of three parts: natural language sentence s, symbolic definition d, and logical expression e (see in Table 7). This is slightly different from the English-FOL dataset where the symbolic definition is not included. We believe that the introduction of symbolic definition is beneficial because it helps to reach agreement among annotators. After that, the annotators are encouraged to rewrite existing data to obtain more challenging data, i.e., some samples have only slight differences in utterances, but their corresponding logical expressions are totally different.

In this way, we obtain a total of 2,727 samples consisting of 1,263 PL and 1,464 FOL with the corresponding utterances and symbolic definitions. To establish human performance and conduct consistency assessments, we ask an additional 3 undergraduate and 2 graduate students who have acquired basic knowledge of PL/FOL to provide logical expressions given natural language sentences and symbolic definitions from the entire test set. The detailed statistics of the dataset are shown in Table 2.

5 Experiments

In the experimental section, we investigate the following research questions: 1) How is the overall performance of Dual-(m)T5 in comparison to competitors? 2) How does Dual-(m)T5 perform on other types of logical expressions? 3) What is the optimal ratio of unlabeled data to labeled data? 4) Are all the components in Dual-(m)T5 necessary? 5) Which samples are not yet well processed by the model? 6) Can the logical expressions generated by Dual-(m)T5 help downstream tasks?

[2]One of the largest online platforms for sharing documents.

5.1 Setup

Datasets. 1) **English-FOL** (Levkovskyi and Li, 2021) is generated by pre-defined templates and contains natural language utterances paired with FOL. We follow the training/validation/test splits as Levkovskyi and Li (2021). 2) **Chinese-PL/FOL** The details of our dataset have been introduced in § 4. Since symbol definition only appears in this dataset, we concatenate it with the original input, i.e., natural language sentence in the prime module and logical expression in the dual module. Due to the small amount of training data, PL and FOL are trained together. 3) **ATIS** (Dahl et al., 1994) consists of queries about flight information and logical expressions in lambda-calculus syntax. For fairness in model comparison, we keep the same preprocessing settings as (Dong and Lapata, 2018; Cao et al., 2019).

Baselines. To verify the effectiveness of our approach on English-FOL and Chinese-PL/FOL, we reproduce several strong baselines as there are not many existing works on these two datasets. For ATIS, we directly compare our method with state-of-the-art works. Refer to Appendix.C for details.

Metrics. We follow the previous work (Levkovskyi and Li, 2021) and take Exact Match (EM) as the evaluation metric.

5.2 Overall Results

We compare our method with competitors on different datasets in Table 3. From the results, we conclude that: **1)** Our models outperform the competitors on both of the datasets, and is not affected by language, which shows the effectiveness and robustness of our models. **2)** Even without additional unlabeled data, our models outperform the competitors only with the labeled data, which indicates that our approach is also available in scenarios without unlabeled data. **3)** By introducing the unlabeled data, the performance of the models is further improved, and the improvement on Chinese-PL/FOL is more obvious than that on English-FOL as the amount of the labeled data in English-FOL is enough for the training and the performance is hardly improved by using the unlabeled data, while the unlabeled data can be used to compensate for the paucity of the labeled data on Chinese-PL/FOL.

5.3 Generalization on Lambda-Calculus

To verify that our framework is still effective on other types of logical expressions, we compare our method with previous works on ATIS without using unlabeled data in fairness. From the results shown in Table 4, we see that our method has better performance over the previous works on ATIS, which demonstrates the generality of our approach to other types of logical expressions. This is mainly attributed to the model-based validity reward that is not limited to the specific form of the logical expression, and our framework is suitable for any kind of logical expression without modification.

5.4 Experiments on Semi-supervised Setting

To investigate whether unlabeled data benefits the framework and the optimal ratio of unlabeled data to labeled data, we keep a part of the training set as fully labeled data and leave the rest as unlabeled data where only utterances are used. We change the ratio of unlabeled data to labeled data, and the results on English-FOL are shown in Table 5. The results show that the performance of the models does not improve constantly when the amount of unlabeled data is increased. We conclude that a proper ratio of unlabeled data is crucial and it is related to the number of parameters in the model. A model with more parameters tends to perform better with more unlabeled data. On the contrary, when a model with few parameters is trained with a large amount of unlabeled data by dual reinforcement learning, it may converge to a wrong equilibrium state to adapt to the unlabeled data and forget what has been learned from the labeled data, which leads to poor performance.

5.5 Ablation Study

To evaluate the effectiveness of each component in Dual-(m)T5, we perform an ablation analysis on English-FOL. From the results shown in Table 6, we conclude that: **1)** Dual reinforcement learning without the validity reward even gets worse results than the T5 baseline, which indicates that the validity reward is critical and indispensable in dual reinforcement learning. **2)** Curriculum learning improves EM but not significantly. Therefore, the value of curriculum learning is mainly to stabilize the training process rather than improve the performance. **3)** The model-based validity reward in dual reinforcement learning has certain advantages over the rule-based validity reward (Cao et al., 2019), which indicates the effectiveness of our approach.

Method	English-FOL	Chinese-PL/FOL		
		PL	FOL	TOTAL
Human Performance	-	87.94	79.92	84.07
ATT (Luong et al., 2015)	65.70	39.39	36.88	38.10
GPT-2 (Radford et al., 2019)	85.32	56.04	61.74	58.97
GPT-2-large (Radford et al., 2019)	90.03	-	-	-
Text2log (Levkovskyi and Li, 2021)	89.54	-	-	-
(m)T5-small (Raffel et al., 2019; Xue et al., 2021)	89.95	64.02	58.87	61.35
(m)T5-base (Raffel et al., 2019; Xue et al., 2021)	91.30	70.08	61.35	65.57
Dual-(m)T5-small **(Ours)**	90.98	64.77±0.47	61.70±0.58	63.19±0.52
+ generated unlabeled data	91.06	70.83±0.78	63.83±0.75	67.22±0.75
Dual-(m)T5-base **(Ours)**	92.65	70.45±0.31	63.12±0.29	66.67±0.15
+ generated unlabeled data	**92.83**	**75.00±0.94**	**68.44±0.73**	**71.61±0.83**

Table 3: EM on the test set of English-FOL and Chinese-PL/FOL. (Dual-)T5 and (Dual-)mT5 are used on English-FOL and Chinese-PL/FOL, respectively. Generated unlabeled data represents the unlabeled data obtained by paraphrase generation models in § 3.3.2.

Method	EM
TISP (Zhao and Huang, 2015)	84.2
Seq2tree (Dong and Lapata, 2016)	84.6
ASN+SUPATT (Rabinovich et al., 2017)	85.9
Tranx (Yin and Neubig, 2018)	86.2
Coarse2fine (Dong and Lapata, 2018)	87.7
Transformer (Ge et al., 2019)	87.7
ATTPTR + Dual (Cao et al., 2019)	88.6
TreeGen (Sun et al., 2020)	89.1
Dual-T5-base **(Ours)**	**89.5**

Table 4: EM on the test set of ATIS.

Method	Labeled:Unlabeled	EM
Dual-T5-small	-	78.32
+ unlabeled data	**1:1**	**79.34**
+ unlabeled data	1:2	79.27
+ unlabeled data	1:3	78.37
+ unlabeled data	1:4	77.93
Dual-T5-base	-	87.01
+ unlabeled data	1:1	88.94
+ unlabeled data	1:2	89.22
+ unlabeled data	**1:3**	**89.34**
+ unlabeled data	1:4	88.76

Table 5: EM on the test set of English-FOL. It fixes the number of labeled samples (20% of the training set) and varies the ratio of unlabeled data to labeled data.

5.6 Error Analysis

For error analysis, we present three typical bad cases of Chinese-PL/FOL in Table 7. In Case 1, the model inverts cause and effect, indicating that the ability of causal reasoning needs to be enhanced. Case 2 requires the model to have the capability of coreference resolution, while Dual-(m)T5 does not have this ability yet. Case 3 shows that some errors are due to one utterance may correspond to multiple correct logical expressions, while there is only one annotated ground truth[3]. Such a problem

[3]In Case3, the ground truth is closer to the meaning of the utterance than the prediction, but the latter is an equivalent representation of the ground truth.

Method	EM
T5-small	89.95
w/ curriculum	90.09
w/ dual (w/o R_e^{val}, w/ R_s^{rec})	89.23
w/ dual (w/ rule-based R_e^{val}, w/ R_s^{rec})	90.44
w/ dual (w/ model-based R_e^{val}, w/ R_s^{rec})	**90.88**
T5-base	91.30
w/ curriculum	91.41
w/ dual (w/o R_e^{val}, w/ R_s^{rec})	90.87
w/ dual (w/ rule-based R_e^{val}, w/ R_s^{rec})	91.96
w/ dual (w/ model-based R_e^{val}, w/ R_s^{rec})	**92.51**

Table 6: Ablations on the test set of English-FOL.

exists in all three datasets and has been ignored by previous works. To compensate for this, we perform manual statistics on the incorrect prediction results of Dual-mT5-base on the test set of Chinese-PL/FOL. About 5.2% of errors are due to the above. We think this may be solved by defining a series of equivalence transformation rules. We leave this for future work.

5.7 Improvements to Downstream Tasks

To further demonstrate the value of the generated logical expressions, we take Natural Language Inference (NLI) as a downstream task. NLI involves reading a pair of sentences and judging the relationship between their meanings, such as entailment, neutral and contradiction. We explore if additional logical expressions improve the model's performance on the NLI task. We conduct experiments on the Stanford Natural Language Inference (SNLI) dataset (Bowman et al., 2015). The backbone model is BERT (Devlin et al., 2019) which concatenates two sentences ($s1$ and $s2$) with specical tokens as input, and uses the representation of $[CLS]$ token for text classification. In practice, we concatenate the logical expressions ($e1$ and $e2$)

Chinese-PL/FOL
Natural Language Utterance
Xiao Ming lives alone because he is not married.
Symbolic Definition
$A(x)$: x lives alone; $B(x)$: x is married; a: Xiao Ming
Ground Truth
$\neg B(a) \rightarrow A(a)$
Prediction
$A(a) \rightarrow \neg B(a)$
Natural Language Utterance
The dark night has given me a pair of black eyes, but I use them to find the light.
Symbolic Definition
$A(x, y)$: x give me y; $B(x, y)$: i use x to find y; a: dark night; b: black eyes; c: light
Ground Truth
$A(a, b) \wedge B(b, c)$
Prediction
$A(a, b) \wedge B(a, c)$
Natural Language Utterance
Not everyone who can play football can play basketball.
Symbolic Definition
$A(x)$: x can play football; $B(x)$: x can play basketball; x: person
Ground Truth
$\neg \forall x(A(x) \rightarrow B(x))$
Prediction
$\exists x(A(x) \wedge (\neg B(x)))$

Table 7: Case study of Dual-(m)T5-base on Chinese-PL/FOL. Text in red and brown represents the difference between the prediction and ground truth.

corresponding to the sentences ($s1$ and $s2$) on the input, i.e, $[CLS]$ $s1$ $e1$ $[SEP]$ $s2$ $e2$ $[SEP]$. We vary the amount of training data from 10k to 100k and the results are shown in Fig.2.

As shown in the results, the model has better performance by introducing logical expressions when the amount of training data is not large. As the amount of training data increases, the role of logical expressions gradually decreases. We conclude that the generated logical expression explicitly represents the logical information contained in the sentences and is suitable for the NLI task in low-resource scenarios as a supplementary.

6 Related Works

Parsing Natural Language into PL/FOL Logic expressions are commonly written in standardized mathematical notation, and learning this notation typically requires many years of experience. Barker-Plummer et al. (2009) study why students find translating natural language sentences into FOL hard and systematically categorize the problems encountered by students. Bansal (2015) proposes a rule-based framework that leverages the Part-of-speech structure of natural language sentences. Limited to the manually defined rules and

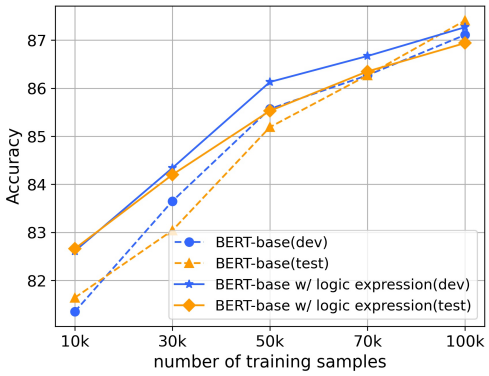

Figure 2: Accuracy on the dev and test set of the SNLI dataset.

a small amount of experimental data, the system only works under a specific setting. With the development of deep learning, neural approaches alleviate the need for manually defining lexicons. Singh et al. (2020) examine the capability of neural models on parsing FOL from natural language sentences. They propose to disentangle the representations of different token categories while generating FOL and use category prediction as an auxiliary task. Unfortunately, they do not release the dataset they construct. Levkovskyi and Li (2021) release a dataset containing English-FOL sentence pairs and set up a baseline encoder-decoder model, but the dataset is not challenging for it is generated by templates, and vanilla models obtain high scores.

Dual Learning Dual learning is first proposed to improve neural machine translation (NMT) (He et al., 2016). The author makes full use of monolingual corpus to improve the effectiveness of the model through dual learning. Xia et al. (2017) introduce a probabilistic duality term to serve as a data-dependent regularizer to better guide the dual supervised learning. Since then, the idea of dual learning has been applied in various tasks, such as Question Answering/Generation (Tang et al., 2017), Open-domain Information Extraction/Narration (Sun et al., 2018), Semantic Parsing with lambda calculus (Cao et al., 2019, 2020), and Emotion-Controllable Response Generation (Shen and Feng, 2020).

7 Conclusion

In this paper, we introduce **Dual-(m)T5**, an effective dual reinforcement learning framework for parsing natural language into PL/FOL. A novel reward mechanism is proposed to avoid manually defining the validity reward in RL. An effective training strategy is further proposed to stabilize the

training process. Experimental results show that the proposed method outperforms competitors on several datasets. By introducing logical expressions, we further enhance the existing NLI model. In addition to the technical contribution, a new dataset called **Chinese-PL/FOL** is constructed to aid further research in this field.

Acknowledgement

We thank the anonymous reviewers for their constructive suggestions. This work was supported by National Key Research and Development Project (No.2020AAA0109302), Shanghai Science and Technology Innovation Action Plan (No.19511120400), Shanghai Municipal Science and Technology Major Project (No.2021SHZDZX0103).

References

Naman Bansal. 2015. *Translating Natural Language Propositions to First Order Logic*. Ph.D. thesis, INDIAN INSTITUTE OF TECHNOLOGY KANPUR.

Dave Barker-Plummer, Richard Cox, and Robert Dale. 2009. Dimensions of difficulty in translating natural language into first order logic. *International Working Group on Educational Data Mining*.

Yoshua Bengio, Jérôme Louradour, Ronan Collobert, and Jason Weston. 2009. Curriculum learning. In *Proceedings of the 26th Annual International Conference on Machine Learning, ICML 2009, Montreal, Quebec, Canada, June 14-18, 2009*, volume 382 of *ACM International Conference Proceeding Series*, pages 41–48. ACM.

Patrick Blackburn and Johannes Bos. 2005. *Representation and inference for natural language: A first course in computational semantics*. Center for the Study of Language and Information Amsterdam.

Samuel R. Bowman, Gabor Angeli, Christopher Potts, and Christopher D. Manning. 2015. A large annotated corpus for learning natural language inference. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*.

Ruisheng Cao, Su Zhu, Chen Liu, Jieyu Li, and Kai Yu. 2019. Semantic parsing with dual learning. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 51–64, Florence, Italy. Association for Computational Linguistics.

Ruisheng Cao, Su Zhu, Chenyu Yang, Chen Liu, Rao Ma, Yanbin Zhao, Lu Chen, and Kai Yu. 2020. Unsupervised dual paraphrasing for two-stage semantic parsing. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 6806–6817, Online. Association for Computational Linguistics.

Deborah A. Dahl, Madeleine Bates, Michael Brown, William Fisher, Kate Hunicke-Smith, David Pallett, Christine Pao, Alexander Rudnicky, and Elizabeth Shriberg. 1994. Expanding the scope of the ATIS task: The ATIS-3 corpus. In *Human Language Technology: Proceedings of a Workshop held at Plainsboro, New Jersey, March 8-11, 1994*.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*.

Li Dong and Mirella Lapata. 2016. Language to logical form with neural attention. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 33–43, Berlin, Germany. Association for Computational Linguistics.

Li Dong and Mirella Lapata. 2018. Coarse-to-fine decoding for neural semantic parsing. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 731–742, Melbourne, Australia. Association for Computational Linguistics.

Richard Evans and Edward Grefenstette. 2018. Learning explanatory rules from noisy data. *Journal of Artificial Intelligence Research*, 61:1–64.

Donglai Ge, Junhui Li, and Muhua Zhu. 2019. A transformer-based semantic parser for nlpcc-2019 shared task 2. In *CCF International Conference on Natural Language Processing and Chinese Computing*. Springer.

Di He, Yingce Xia, Tao Qin, Liwei Wang, Nenghai Yu, Tie-Yan Liu, and Wei-Ying Ma. 2016. Dual learning for machine translation. In *Advances in Neural Information Processing Systems 29: Annual Conference on Neural Information Processing Systems 2016, December 5-10, 2016, Barcelona, Spain*, pages 820–828.

Ari Holtzman, Jan Buys, Li Du, Maxwell Forbes, and Yejin Choi. 2020. The curious case of neural text degeneration. In *8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, April 26-30, 2020*. OpenReview.net.

Daiki Kimura, Masaki Ono, Subhajit Chaudhury, Ryosuke Kohita, Akifumi Wachi, Don Joven Agravante, Michiaki Tatsubori, Asim Munawar, and Alexander Gray. 2021. Neuro-symbolic reinforcement learning with first-order logic. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 3505–3511.

Oleksii Levkovskyi and Wei Li. 2021. Generating predicate logic expressions from natural language. In *SoutheastCon 2021*, pages 1–8. IEEE.

Carmelo Fabio Longo and Corrado Santoro. 2020. Adcaspar: Abductive-deductive cognitive architecture based on natural language and first order logic reasoning. In *NL4AI@ AI* IA*, pages 73–86.

Ilya Loshchilov and Frank Hutter. 2018. Fixing weight decay regularization in adam.

Thang Luong, Hieu Pham, and Christopher D. Manning. 2015. Effective approaches to attention-based neural machine translation. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*.

Zhihao Ma, Yuzheng Zhuang, Paul Weng, Dong Li, Kun Shao, Wulong Liu, Hankz Hankui Zhuo, and HAO Jianye. 2020. Interpretable reinforcement learning with neural symbolic logic.

Emmanouil Antonios Platanios, Otilia Stretcu, Graham Neubig, Barnabas Poczos, and Tom Mitchell. 2019. Competence-based curriculum learning for neural machine translation. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 1162–1172, Minneapolis, Minnesota. Association for Computational Linguistics.

Maxim Rabinovich, Mitchell Stern, and Dan Klein. 2017. Abstract syntax networks for code generation and semantic parsing. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*.

Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, Ilya Sutskever, et al. 2019. Language models are unsupervised multitask learners. *OpenAI blog*, 1(8):9.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J Liu. 2019. Exploring the limits of transfer learning with a unified text-to-text transformer. *arXiv preprint arXiv:1910.10683*.

Lei Shen and Yang Feng. 2020. CDL: Curriculum dual learning for emotion-controllable response generation. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 556–566, Online. Association for Computational Linguistics.

Hrituraj Singh, Milan Aggrawal, and Balaji Krishnamurthy. 2020. Exploring neural models for parsing natural language into first-order logic. *arXiv preprint arXiv:2002.06544*.

Jianlin Su. 2021. Roformer-sim: Integrating retrieval and generation into roformer. Technical report.

Mingming Sun, Xu Li, and Ping Li. 2018. Logician and orator: Learning from the duality between language and knowledge in open domain. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 2119–2130, Brussels, Belgium. Association for Computational Linguistics.

Zeyu Sun, Qihao Zhu, Yingfei Xiong, Yican Sun, Lili Mou, and Lu Zhang. 2020. Treegen: A tree-based transformer architecture for code generation. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34.

Richard S Sutton, David A McAllester, Satinder P Singh, and Yishay Mansour. 2000. Policy gradient methods for reinforcement learning with function approximation. In *Advances in neural information processing systems*, pages 1057–1063.

Duyu Tang, Nan Duan, Tao Qin, Zhao Yan, and Ming Zhou. 2017. Question answering and question generation as dual tasks. *arXiv preprint arXiv:1706.02027*.

Yingce Xia, Tao Qin, Wei Chen, Jiang Bian, Nenghai Yu, and Tie-Yan Liu. 2017. Dual supervised learning. In *Proceedings of the 34th International Conference on Machine Learning, ICML 2017, Sydney, NSW, Australia, 6-11 August 2017*, volume 70 of *Proceedings of Machine Learning Research*, pages 3789–3798. PMLR.

Linting Xue, Noah Constant, Adam Roberts, Mihir Kale, Rami Al-Rfou, Aditya Siddhant, Aditya Barua, and Colin Raffel. 2021. mT5: A massively multilingual pre-trained text-to-text transformer. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 483–498, Online. Association for Computational Linguistics.

Yuan Yang and Le Song. 2019. Learn to explain efficiently via neural logic inductive learning. *arXiv preprint arXiv:1910.02481*.

Pengcheng Yin and Graham Neubig. 2018. TRANX: A transition-based neural abstract syntax parser for semantic parsing and code generation. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*.

Jingqing Zhang, Yao Zhao, Mohammad Saleh, and Peter Liu. 2020. Pegasus: Pre-training with extracted gap-sentences for abstractive summarization. In *International Conference on Machine Learning*, pages 11328–11339. PMLR.

Kai Zhao and Liang Huang. 2015. Type-driven incremental semantic parsing with polymorphism. In *Proceedings of the 2015 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*.

A Algorithm

Algorithm 1 Training Scoring Model

Input: Supervised dataset $\mathcal{L} = \{\langle s, e\rangle\}$; number of nucleus sampling k; Fine-tuned NL2LE model
Output: scoring model

$\mathcal{P} \leftarrow \{\}, \mathcal{N} \leftarrow \{\}$
for all $\langle s, e\rangle \in \mathcal{L}$ **do**
  $\mathcal{P} \leftarrow \mathcal{P} \cup \{\langle s, e\rangle\}$
  Given s, fine-tuned NL2LE model generates k logical expressions $\{e_i\}$ via nucleus sampling
  for all $e_i \in \{e_i\}$ **do**
    if $e_i \neq e$ **then**
      $\mathcal{N} \leftarrow \mathcal{N} \cup \{\langle s, e_i\rangle\}$
    end if
  end for
end for
repeat
  Update scoring model w.r.t. Eq.(10)
until scoring model converges

B PL and FOL

FOL represents entities and actions in natural language through quantified variables and consists of predicates which take variables as arguments and attach semantics to variables (Blackburn and Bos, 2005), while PL is a relatively simple logical expression and does not deal with quantified variables. Formally, a predicate $P(v_1; v_2; ...; v_n)$ in PL/FOL is an n-ary function of variables v_i that are combined through logical connectives: *logical and* ($\wedge$), *logical or* ($\vee$), *logical not* ($\neg$), *logical implication* ($\rightarrow$), *logical equivalent* ($\leftrightarrow$). What's more, there are two types of quantifiers for FOL: *universal* ($\forall$) which specifies that sub-formula within its scope is true for all instances of the variable and *existential* ($\exists$) which asserts existence of at least one instance represented by a variable under which the sub-formula holds true.

C Baselines

English-FOL and Chinese-PL/FOL

- ATT (Luong et al., 2015). ATT represents attention-based Seq2seq model.
- Text2log (Levkovskyi and Li, 2021). Text2log is the latest work on converting natural language to FOL. This approach is only applicable to English-FOL since a character-level recurrent neural network is leveraged.
- GPT-2 (Radford et al., 2019). GPT-2 is a huge transformer-based model trained on massive datasets and achieves state-of-the-art results on several language modeling datasets in a zero-shot setting when it is proposed. We experiment with different sizes of GPT. Since there is no GPT-large for Chinese, it is vacant in the experiment.
- T5 (Raffel et al., 2019) / mT5 (Xue et al., 2021). T5 refers to the "Text-to-Text Transfer Transformer" which converts several NLP tasks to Text-to-Text task, and mT5 is a multilingual version of T5.

ATIS

- TISP (Zhao and Huang, 2015) An incremental semantic parser that is guided by subtyping and polymorphism.

Algorithm 2 Full Training Process

Input: Supervised dataset $\mathcal{L} = \{\langle s, e\rangle\}$; Unsupervised dataset $\mathcal{U} = \{s'\}$; number of nucleus sampling k; hyper parameters α and β; curriculum training batches T
Output: NL2LE model

// Pre-train NL2LE and LE2NL models
Fine-tune NL2LE model with $\langle s, e\rangle$ from $\mathcal{L}$ and curriculum learning based on Eq.(11)
Fine-tune LE2NL model with $\langle e, s\rangle$ from $\mathcal{L}$
repeat
  // Dual reinforcement learning
  Get mini-batch $\{s\}$ from $\mathcal{L} \cup \mathcal{U}$
  for all $s \in \{s\}$ **do**
    NL2LE model generates k logical expressions $\{e_i\}$ for s via nuclelus sampling
    for all $e_i \in \{e_i\}$ **do**
      Obtain validity and reconstruction reward for e_i
    end for
  end for
  Update Θ_{NL2LE} and Θ_{LE2NL} w.r.t. Eq.(2) and Eq.(5) respectively
  // Supervisor Guidance
  Get mini-batch $\{\langle s, e\rangle\}$ from $\mathcal{L}$
  Fine-tune NL2LE model with $\{\langle s, e\rangle\}$
  Fine-tune LE2NL model with $\{\langle e, s\rangle\}$
until NL2LE model converges

- Seq2tree (Dong and Lapata, 2016) Seq2tree is a method based on an attention-enhanced encoder-decoder model.
- ASN+SUPATT (Rabinovich et al., 2017) This work introduces abstract syntax networks, a modeling framework for code generation and semantic parsing.
- Tranx (Yin and Neubig, 2018) Tranx uses a transition system based on the abstract syntax description language for the target meaning representations.
- Coarse2fine (Dong and Lapata, 2018) Coarse2fine generates meaning sketches first and then predicts missing details to obtain full meaning representations.
- Transformer (Ge et al., 2019) Transformer is a deep learning model that adopts the mechanism of self-attention and is originally proposed for machine translation.
- ATTPTR + Dual (Cao et al., 2019) It is the first work to propose the use of dual learning for semantic parsing, and is the basis of our work.
- TreeGen (Sun et al., 2020) TreeGen uses the attention mechanism of Transformer to alleviate the long-dependency problem and introduces an Abstract Syntax Tree (AST) reader to combine grammar rules and the AST structure.

D Implemention Details

We use Pytorch[4] library for implementing an auto-differentiable graph of our computations. For **pre-training**, (m)T5-small/base are trained with an AdamW optimizer (Loshchilov and Hutter, 2018) initialized with a learning rate of 1e-3/1e-4 with a decay rate of 1e-3/1e-2 respectively. For **dual reinforcement learning**, models are trained with an AdamW optimizer initialized with a learning rate of 1e-5 with a decay rate of 1e-3 for (m)T5-small/base. The batch size is fixed to 8, and the max input and output sentence length are set to 128. Since the size of the test set for the Chinese-PL/FOL dataset is small, we repeat each experiment using 3 different random seeds and report the median number and standard deviation to avoid small sample instability in the results obtained. Training runs until the performance on validation set does not improve. We use PEGASUS (Zhang et al., 2020) fine-tuned for paraphrasing[5] for english paraphrasing, and RoFormer-Sim (Su, 2021) [6] for chinese paraphrasing. For each natural language utterance in the datasets, we generate one synonymous sentence as unlabeled data.[7] Our models run on a computer with Intel(R) Xeon(R) Gold 6230R CPU, 4 GeForce RTX 3090, 64GB of RAM, and Ubuntu 20.04.

E Error Analysis on English-FOL and ATIS

The bad cases of English-FOL and ATIS are presented in Table 8. We select two typical bad cases from two datasets respectively. From case 1 of English-FOL, we can find that there are some labeling errors in the dataset. The prediction is completely correct while the ground truth is wrong. The error in case 2 is due to the fact that the English-FOL dataset does not clearly indicate what the predicate is, which leads to a slight difference between the prediction and the ground truth.[8] For ATIS, the model makes a small mistake in case 1, which indicates that the model is not good enough to handle the details. The error in case 2 is also caused by the predicate not explicitly indicating. If the predicate can be unified, we believe that the model can answer correctly.

[4] https://pytorch.org

[5] https://huggingface.co/tuner007/pegasus_paraphrase

[6] https://github.com/ZhuiyiTechnology/roformer-sim

[7] Generating top-k ($k \geq 1$) synonymous sentences for each natural language utterance leads to poor performance because the unlabeled data is too similar to each other.

[8] In contrast, our dataset clearly points out what the predicate is, effectively solving this problem.

English-FOL
Natural Language Utterance
All people are welcome.
Ground Truth
exists x1.(_all(x1) & _people(x1) & _welcome(x1))
Prediction
all x1.(_people(x1) → _welcome(x1))
Natural Language Utterance
Every database accepts parentheses.
Ground Truth
all x1.(_database(x1) →
exists x2.(_parenthesis(x2) & _accept(x1,x2)))
Prediction
all x1.(_database(x1) →
exists x2.(_parenthes(x2) & _accept(x1,x2)))

ATIS
Natural Language Utterance
What city does al0 fly out of?
Ground Truth
lambda $0 e (and (city $0) (exists $1 (and (flight $1)
(airline $1 al0) (from $1 $0))))
Prediction
lambda $0 e (and (city $0) (exists $1 (and (flight $1)
(airline $1 al0) (to $1 $0))))
Natural Language Utterance
List the st0 airport.
Ground Truth
lambda $0 e (and (airport $0) (loc:t $0 st0))
Prediction
lambda $0 e (and (airport $0) (located at $0 st0))

Table 8: Case study of Dual-(m)T5-base on English-FOL and ATIS. Text in red and brown represents the difference between the prediction and ground truth.

Yet Another Format of Universal Dependencies for Korean

Yige Chen[†*] **Eunkyul Leah Jo**[‡*] **Yundong Yao**[‡*]
KyungTae Lim[¶] **Miikka Silfverberg**[‡] **Francis M. Tyers**[§] **Jungyeul Park**[‡]
[†]The Chinese University of Hong Kong, Hong Kong [‡]The University of British Columbia, Canada
[¶]Hanbat National University & TeddySum, South Korea [§]Indiana University, USA
[†]yigechen@link.cuhk.edu.hk [‡]{eunkyul,iameleve}@student.ubc.ca
[¶]ktlim@hanbat.ac.kr [§]ftyers@iu.edu [‡]{msilfver,jungyeul}@mail.ubc.ca

Abstract

In this study, we propose a morpheme-based scheme for Korean dependency parsing and adopt the proposed scheme to Universal Dependencies. We present the linguistic rationale that illustrates the motivation and the necessity of adopting the morpheme-based format, and develop scripts that convert between the original format used by Universal Dependencies and the proposed morpheme-based format automatically. The effectiveness of the proposed format for Korean dependency parsing is then testified by both statistical and neural models, including UDPipe and Stanza, with our carefully constructed morpheme-based word embedding for Korean. MORPHUD outperforms parsing results for all Korean UD treebanks, and we also present detailed error analyses.

1 Introduction

Dependency parsing is one of the tasks in natural language processing that have been investigated extensively. Using the dependency grammar, it finds the relations between the words in a sentence, and forms an acyclic dependency graph that explains the grammatical structure of the sentence. With various machine learning techniques, the current dependency parsers are able to approach human performances given English corpora.[1] Previous studies trying to probe or improve Korean dependency parsers are lacking, especially for those considering the underlying reason and rationale based on the linguistic properties of Korean. The parsers do not perform as well as their English counterparts, partly due to the fact that Korean is a language that has more complicated linguistic features that make parsing on the word level difficult.

There have been previous studies trying to cope with the word-level representation issues of Korean (Choi and Palmer, 2011; Park et al., 2013; Kanayama et al., 2014). Given that Korean is an agglutinative language that heavily relies on morphemes, and the natural segmentation does not correctly reflect either the words or the morphemes of Korean texts, Park and Tyers (2019) suggested an annotation scheme that decomposes Korean texts into the morpheme level, and applied the morpheme-based format to POS tagging.

In this study, we propose a morpheme-based scheme for Korean dependency parsing that is developed based on Park and Tyers (2019), and adopt the proposed scheme to Universal Dependencies (Nivre et al., 2016, 2020), which contains two Korean dependency parsing treebanks, namely the GSD treebank (McDonald et al., 2013) and the Kaist treebank (Choi et al., 1994; Chun et al., 2018). While the two Korean treebanks meet the standards of Universal Dependencies and have been studied for dependency parsing tasks extensively (Kondratyuk and Straka, 2019; Qi et al., 2020), the treebanks are formatted in a way that the natural segmentations of Korean texts are preserved, and even with some morpheme-level information, only the language-specific part-of-speech tags on the morpheme level are included in the treebanks, and both treebanks do not have any morpheme-level parsing tags. Different from the traditional scheme based on natural segmentation, this scheme utilizes the inherent morphological and typological features of the Korean language, and the morpheme-level parsing tags can therefore be derived using a set of linguistically motivated rules, which are further used to produce the morpheme-level dependency parsing results and automatic conversions between the morpheme-based format and the traditional format.

The proposed morpheme-based representation is examined using several dependency parsing

[*]Yige Chen, Eunkyul Leah Jo, and Yundong Yao contributed equally. [¶]Corresponding author.

[1]https://ai.googleblog.com/2016/05/announcing-syntaxnet-worlds-most.html

Proceedings of the 29th International Conference on Computational Linguistics, pages 5432–5437
October 12–17, 2022.

models, including UDPipe (Straka et al., 2016; Straka and Straková, 2017) and Stanza (Qi et al., 2020). Compared to the baseline models trained using the two treebanks without modification, our proposed format makes statistically significant improvements in the performances of the parsing models for the Korean language as reported in the error analysis.

2 Representation of MORPHUD

In this study, we adopt a morpheme-based format that captures the linguistic properties of the Korean language proposed by Park and Tyers (2019). The natural segmentation of Korean is based on eojeol, which does not necessarily reflect the actual word or morpheme boundaries of the language. For example, an eojeol of Korean may contain both a noun and its postposition, or both a verb and its particles marking tense, aspect, honorifics, etc. While this is typical for Korean as an agglutinative language, it creates difficulties and challenges for NLP tasks regarding the Korean language, including dependency parsing. It is not ideal that the tokens dependency relations are annotated on are sometimes words, and sometimes phrases as an eojeol may consist of more than a word. Furthermore, Korean as an agglutinative language has very regular conjugations, which makes it easy and natural to split those words and phrases into morphemes when analyzing the language since nearly every piece of an eojeol can be identified to be of a certain meaning or function.

The morpheme-based format aims at decomposing the Korean sentences further into morphemes, which means that dependency relations are no longer marked on the eojeol level. Instead, they are marked on morphemes such that within each eojeol that is not monomorphemic, a head of that eojeol will be found and all other morphemes will be attached directly to the head. As a result, the head of a non-monomorphemic eojeol carries the dependency relation this eojeol originally has, and all other morphemes will be attached to it. In order to find the head, we develop a script and apply some heuristics which include that the head of an eojeol is usually a noun, a proper noun, or a verb, and while there is no noun or verb in an eojeol, the script we implemented continues to find other morphemes such as pronouns, adjectives, adverbs, numerals, etc. The script also excludes the use of adpositions, conjunctions, and particles as heads in most cases, unless these are the only part-of-speeches in an eojeol except for punctuations. While there are multiple morphemes that can be heads in an eojeol, the script will decide which one to take based on the part-of-speeches of the morphemes. For instance, when there are multiple nouns, the last one will carry the dependency relation of the eojeol, whereas when there are multiple verbs, the first verb will carry the dependency relation as Korean is a head-final language. Once the head of a non-monomorphemic eojeol is found, the other morphemes will be dependent on the head and be assigned with other dependency relations such as compound, case, auxiliary depending on their UPOS and XPOS.

3 Experiments and results

3.1 Data and systems

In this study, we deploy two parsers to evaluate our proposed format, namely UDPipe (Straka et al., 2016) as a baseline system and Stanza (Qi et al., 2020) as one of the state-of-the-art dependency parsers. UDPipe is a pipeline designed for processing CoNLL-U formatted files, which performs tokenization using Bi-LSTM, morphological analysis, part-of-speech tagging, lemmatization using MorphoDiTa (Straková et al., 2014), and dependency parsing using slightly modified Parsito (Straka et al., 2015). Since the whole pipeline needs no language-specific knowledge, which means that it can be trained using corpora in a different scheme, we choose UDPipe as our baseline. Stanza is another natural language processing toolkit that includes Dozat's biaffine attention dependency parser (Dozat and Manning, 2017). Dozat's dependency parser uses the minimum spanning tree algorithm that can deal with non-projectivity dependency relations, and more importantly it excelled all of dependency parsers during CoNLL 2017 and 2018 Shared Task (Zeman et al., 2017, 2018) In this study, the two dependency parsing pipelines take both the original word-based form[2] (the current scheme adopted by Universal Dependencies), which we denote as WORDUD, and the morpheme-based form, which we denote as MORPHUD, of the GSD and KAIST treebanks as the input.

We develop the script to convert between

[2]While words and eojeols are not the same in Korean based on their definitions, in this study, the terms "word-based" and "eojeol-based" are interchangeable.

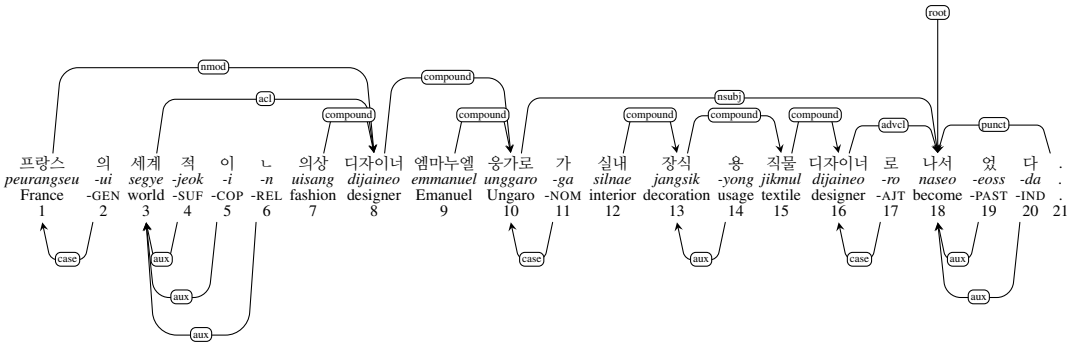

Figure 1: Example of morpheme-based universal dependencies for Korean: while dependencies in top-side are the original dependencies between words, dependencies in bottom-side are newly added dependencies for between morphemes.

the WORDUD format and our proposed MORPHUD format. The script consists of two major components, which are WORDUD to MORPHUD (`Word2Morph`) and MORPHUD to WORDUD (`Morph2Word`). The Word2Morph component splits the word tokens in the CoNLL-U treebank of Korean into morphemes using the lemmas already provided, and assigns dependency relations on the resegmented tokens based on the original dependency relations annotated on the word tokens. The Morph2Word component, on the other hand, firstly pairs the tokens in the WORDUD dataset and the MORPHUD dataset, and then assigns the dependency relations from morpheme tokens in MORPHUD to word tokens in WORDUD. Within both components, a root detector for the word is implemented in order to find the root (or stem) of a word when the word is multimorphemic (i.e., needs to be split into morphemes and attach dependency relations on it correspondingly). Evaluations of the conversion scripts are not conducted in this study, since the morphemes are inherited from the lemmas in the treebanks, and the part-of-speech tags, roots, and dependency relations are predicted and assigned to the morphemes based on the linguistic features and the grammar of Korean that are regular, as presented in Section 2.

3.2 Results

We report the labeled attachment score (LAS), which is a standard evaluation metric in dependency parsing, using the evaluation script (2018 version) provided by CoNLL 2018 Shared Task.[3] Table 1 shows results of udpipe as a baseline system and stanza as one of the state-of-the-art systems. All results are reported in the WORDUD format. That is, all experiments are trained and predicted in the proposed MORPHUD format, and then the result is converted back to the WORDUD format for comparison purposes. We train udpipe once because it can produce the same parsing model if we train it on the same machine. For stanza, we provide average LAS and its standard deviation after five training and evaluation. Both systems use the finely crafted 300d embedding file by fastText (Bojanowski et al., 2017): WORDUD and MORPHUD use words and morphemes as their embedding entries, respectively to make sure that their input representation would be correctly matched. For embeddings, there are 9.6M sentences and 157M words (tokenized) based on WORDUD. The set of documents for embeddings includes all articles published in *The Hankyoreh* during 2016 (1.2M sentences), Sejong morphologically analyzed corpus (3M), and Korean Wikipedia articles (20201101) (5.3M). As expected, all results of MORPHUD outperform WORDUD in Table 1.

3.3 Error analysis and discussion

Figure 2 shows the confusion matrix between WORDUD and MORPHUD, in which the column and the row represent the arc direction of `gold` and `system`, respectively. MORPHUD outperforms WORDUD in predicting all directions except for right (`gold`) / left (`system`). The system predicts the left arc instead of the correct right arc (212 arc direction errors in WORDUD vs. 240 in MORPHUD). This is because we spuriously added left arcs for functional morphemes in MORPHUD where the system learned more left arc instances

[3] https://universaldependencies.org/conll18/conll18_ud_eval.py

	ko_gsd		ko_kaist	
	WORDUD	+MORPHUD	WORDUD	+MORPHUD
udpipe	70.90	77.01	77.01	81.80
stanza	84.63 (±0.18)	84.98 (±0.20)	86.67 (±0.17)	88.46 (±0.14)

Table 1: Dependency parsing results: for the comparison purpose all MORPHUD results are converted back to WORDUD after training and predicting with the format of MORPHUD

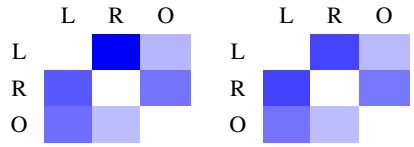

(a) WORDUD (b) MORPHUD

Figure 2: Confusion matrix for the direction of arcs where the column represents gold, and the row system: Left, Right, and O for TO ROOT.

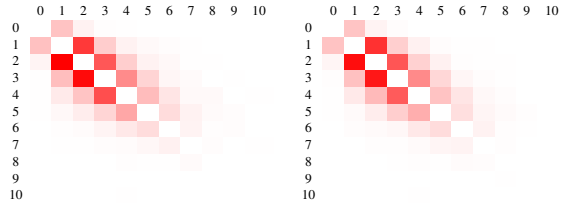

(a) WORDUD (b) MORPHUD

Figure 3: Confusion matrix for the depth of arcs where the column represents gold, and the row system.

during training.

Figure 3 presents the confusion matrix for the arc depth. The most frequent arc depth error is 2 (gold) / 1 (system) (901 arc depth errors in WORDUD vs. 855 in MORPHUD). Figure 4 shows an example of parsing errors generated by MaltEval (Nilsson and Nivre, 2008). The parsing error shows that whereas the gold's arc requires the depth 2, the system predicts the depth 1. This is mainly because the analysis of compound nouns for the NP modifier in the Korean treebank prefers a left skewed tree as shown in Figure 5 where some nouns are a verbal noun, and it plays a role as a predicate of the precedent NP modifier. This is a quite different from the English treebank where the right skewed tree dominates: [NP [PRPS *its*] [N [NN *Micronite*] [N [NN *cigarette*] [NNS *filters*]]]]. This is a well-known problem when parsing the Korean treebank because it requires the semantics of the noun to distinguish between the right and the left skewed trees. One possible remedy for this problem was to build a fully lexicalized parsing

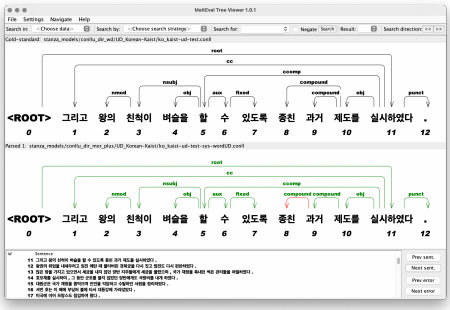

Figure 4: Example of the 2/1 error by MaltEval where the gold's arc depth is 2 and the system's depth is 1. Note that MORPHUD results are converted back to WORDUD: *geuligo wang-ui chinjog-i byeoseul-eul hal su issdolog jongchin gwageo jedo-leul silsihayeossda* 'And they introduced a clan system to make sure that the king's relatives can obtain the government position'

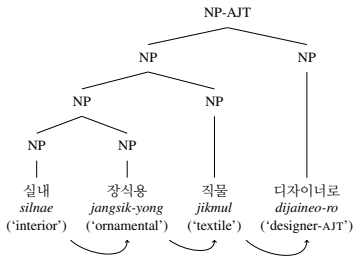

Figure 5: Compound noun with a left-skewed tree for NP modifiers in the Korean treebank

system (Park et al., 2013).

4 Conclusion

We proposed a new annotation scheme for Universal Dependencies for Korean. We have already worked on NER, in which we outperformed the word-level representation dataset by using the morphologically enhanced dataset, and we are planning to extend our idea to the semantic role labeling task. We are also trying to create a consortium to develop the morphologically enhanced Universal Dependencies for other morphologically rich languages such as Basque, Finnish, French, German, Hungarian, Polish, and Swedish. All conversion scripts (WORDUD to MORPHUD,

and vice versa), and MORPHUD datasets for `ko_gsd` and `ko_kaist` will be available through author's `github` at https://github.com/jungyeul/morphUD-korean.

References

Piotr Bojanowski, Edouard Grave, Armand Joulin, and Tomas Mikolov. 2017. Enriching Word Vectors with Subword Information. *Transactions of the Association for Computational Linguistics*, 5:135–146.

Jinho D. Choi and Martha Palmer. 2011. Statistical Dependency Parsing in Korean: From Corpus Generation To Automatic Parsing. In *Proceedings of the Second Workshop on Statistical Parsing of Morphologically Rich Languages*, pages 1–11, Dublin, Ireland. Association for Computational Linguistics.

Key-Sun Choi, Young S. Han, Young G. Han, and Oh W. Kwon. 1994. KAIST Tree Bank Project for Korean: Present and Future Development. In *Proceedings of the International Workshop on Sharable Natural Language Resources*, pages 7–14, Nara Institute of Science and Technology. Nara Institute of Science and Technology.

Jayeol Chun, Na-Rae Han, Jena D. Hwang, and Jinho D. Choi. 2018. Building Universal Dependency Treebanks in Korean. In *Proceedings of the Eleventh International Conference on Language Resources and Evaluation (LREC 2018)*, Miyazaki, Japan. European Language Resources Association (ELRA).

Timothy Dozat and Christopher D. Manning. 2017. Deep Biaffine Attention for Neural Dependency Parsing. In *Proceedings of the 5th International Conference on Learning Representations (ICLR 2017)*, Toulon, France. The International Conference on Learning Representations (ICLR).

Hiroshi Kanayama, Youngja Park, Yuta Tsuboi, and Dongmook Yi. 2014. Learning from a Neighbor: Adapting a Japanese Parser for Korean Through Feature Transfer Learning. In *Proceedings of the EMNLP'2014 Workshop on Language Technology for Closely Related Languages and Language Variants*, pages 2–12, Doha, Qatar. Association for Computational Linguistics.

Dan Kondratyuk and Milan Straka. 2019. 75 languages, 1 model: Parsing universal dependencies universally. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 2779–2795, Hong Kong, China. Association for Computational Linguistics.

Ryan McDonald, Joakim Nivre, Yvonne Quirmbach-Brundage, Yoav Goldberg, Dipanjan Das, Kuzman Ganchev, Keith Hall, Slav Petrov, Hao Zhang, Oscar Täckström, Claudia Bedini, Núria Bertomeu Castelló, and Jungmee Lee. 2013. Universal Dependency Annotation for Multilingual Parsing. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 92–97, Sofia, Bulgaria. Association for Computational Linguistics.

Jens Nilsson and Joakim Nivre. 2008. MaltEval: an Evaluation and Visualization Tool for Dependency Parsing. In *Proceedings of the Sixth International Conference on Language Resources and Evaluation (LREC'08)*, Marrakech, Morocco. European Language Resources Association (ELRA).

Joakim Nivre, Marie-Catherine de Marneffe, Filip Ginter, Yoav Goldberg, Jan Hajic, Christopher D. Manning, Ryan McDonald, Slav Petrov, Sampo Pyysalo, Natalia Silveira, Reut Tsarfaty, and Daniel Zeman. 2016. Universal Dependencies v1: A Multilingual Treebank Collection. In *Proceedings of the Tenth International Conference on Language Resources and Evaluation (LREC 2016)*, page 1659–1666, Portorož, Slovenia. European Language Resources Association (ELRA).

Joakim Nivre, Marie-Catherine de Marneffe, Filip Ginter, Jan Hajič, Christopher D. Manning, Sampo Pyysalo, Sebastian Schuster, Francis Tyers, and Daniel Zeman. 2020. Universal Dependencies v2: An Evergrowing Multilingual Treebank Collection. In *Proceedings of the 12th Language Resources and Evaluation Conference*, pages 4034–4043, Marseille, France. European Language Resources Association.

Jungyeul Park, Daisuke Kawahara, Sadao Kurohashi, and Key-Sun Choi. 2013. Towards Fully Lexicalized Dependency Parsing for Korean. In *Proceedings of the 13th International Conference on Parsing Technologies (IWPT 2013)*, pages 120–126, Nara, Japan. Assocation for Computational Linguistics.

Jungyeul Park and Francis Tyers. 2019. A New Annotation Scheme for the Sejong Part-of-speech Tagged Corpus. In *Proceedings of the 13th Linguistic Annotation Workshop*, pages 195–202, Florence, Italy. Association for Computational Linguistics.

Peng Qi, Yuhao Zhang, Yuhui Zhang, Jason Bolton, and Christopher D. Manning. 2020. Stanza: A Python Natural Language Processing Toolkit for Many Human Languages. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics: System Demonstrations*, pages 101–108, Online. Association for Computational Linguistics.

Milan Straka, Jan Hajic, and Jana Straková. 2016. UDPipe: Trainable Pipeline for Processing CoNLL-U Files Performing Tokenization, Morphological Analysis, POS Tagging and Parsing. In *Proceedings of the Tenth International Conference on Language Resources and Evaluation (LREC 2016)*, pages 4290–4297, Paris, France. European Language Resources Association (ELRA).

Milan Straka, Jan Hajič, Jana Straková, and Jan Hajič Jr. 2015. Parsing Universal Dependency Treebanks using Neural Networks and Search-Based Oracle. In *14th International Workshop on Treebanks and Linguistic Theories (TLT 2015)*, pages 208–220, Warszawa, Poland. IPIPAN.

Milan Straka and Jana Straková. 2017. Tokenizing, POS Tagging, Lemmatizing and Parsing UD 2.0 with UDPipe. In *Proceedings of the CoNLL 2017 Shared Task: Multilingual Parsing from Raw Text to Universal Dependencies*, pages 88–99, Vancouver, Canada. Association for Computational Linguistics.

Jana Straková, Milan Straka, and Jan Hajič. 2014. Open-source tools for morphology, lemmatization, POS tagging and named entity recognition. In *Proceedings of 52nd Annual Meeting of the Association for Computational Linguistics: System Demonstrations*, pages 13–18, Baltimore, Maryland. Association for Computational Linguistics.

Daniel Zeman, Jan Hajič, Martin Popel, Martin Potthast, Milan Straka, Filip Ginter, Joakim Nivre, and Slav Petrov. 2018. CoNLL 2018 Shared Task: Multilingual Parsing from Raw Text to Universal Dependencies. In *Proceedings of the CoNLL 2018 Shared Task: Multilingual Parsing from Raw Text to Universal Dependencies*, pages 1–21, Brussels, Belgium. Association for Computational Linguistics.

Daniel Zeman, Martin Popel, Milan Straka, Jan Hajič, Joakim Nivre, Filip Ginter, Juhani Luotolahti, Sampo Pyysalo, Slav Petrov, Martin Potthast, Francis Tyers, Elena Badmaeva, Memduh Gokirmak, Anna Nedoluzhko, Silvie Cinková, Jan Hajič jr., Jaroslava Hlaváčová, Václava Kettnerová, Zdeňka Urešová, Jenna Kanerva, Stina Ojala, Anna Missilä, Christopher D. Manning, Sebastian Schuster, Siva Reddy, Dima Taji, Nizar Habash, Herman Leung, Marie-Catherine de Marneffe, Manuela Sanguinetti, Maria Simi, Hiroshi Kanayama, Valeria de Paiva, Kira Droganova, Héctor Martínez Alonso, Çagrı Çöltekin, Umut Sulubacak, Hans Uszkoreit, Vivien Macketanz, Aljoscha Burchardt, Kim Harris, Katrin Marheinecke, Georg Rehm, Tolga Kayadelen, Mohammed Attia, Ali Elkahky, Zhuoran Yu, Emily Pitler, Saran Lertpradit, Michael Mandl, Jesse Kirchner, Hector Fernandez Alcalde, Jana Strnadová, Esha Banerjee, Ruli Manurung, Antonio Stella, Atsuko Shimada, Sookyoung Kwak, Gustavo Mendonça, Tatiana Lando, Rattima Nitisaroj, and Josie Li. 2017. CoNLL 2017 Shared Task: Multilingual Parsing from Raw Text to Universal Dependencies. In *Proceedings of the CoNLL 2017 Shared Task: Multilingual Parsing from Raw Text to Universal Dependencies*, pages 1–19, Vancouver, Canada. Association for Computational Linguistics.

Enhancing Structure-aware Encoder with Extremely Limited Data for Graph-based Dependency Parsing

Yuanhe Tian[♥], **Yan Song**[♠†], **Fei Xia**[♥]
[♥]University of Washington [♠]University of Science and Technology of China
[♥]{yhtian, fxia}@uw.edu [♠]clksong@gmail.com

Abstract

Dependency parsing is an important fundamental natural language processing task which analyzes the syntactic structure of an input sentence by illustrating the syntactic relations between words. To improve dependency parsing, leveraging existing dependency parsers and extra data (e.g., through semi-supervised learning) has been demonstrated to be effective, even though the final parsers are trained on inaccurate (but massive) data. In this paper, we propose a frustratingly easy approach to improve graph-based dependency parsing, where a structure-aware encoder is pre-trained on auto-parsed data by predicting the word dependencies and then fine-tuned on gold dependency trees, which differs from the usual pre-training process that aims to predict the context words along dependency paths. Experimental results and analyses demonstrate the effectiveness and robustness of our approach to benefit from the data (even with noise) processed by different parsers, where our approach outperforms strong baselines under different settings with different dependency standards and model architectures used in pre-training and fine-tuning. More importantly, further analyses find that only 2K auto-parsed sentences are required to obtain improvement when pre-training vanilla BERT-large based parser without requiring extra parameters.[1]

1 Introduction

Dependency parsing aims to produce the syntactic structure of a sentence by illustrating the syntactic relations between words, where the words with dependency relations are connected by directed and labeled arcs. It is an important fundamental natural language processing (NLP) task that is widely used to enhance downstream NLP tasks (Cai et al., 2009; Strubell et al., 2018; Huang and Carley, 2019; Zhang et al., 2019; Guo et al., 2019; Nie et al., 2020; Zhou et al., 2020b; Chen et al., 2020; Tian et al., 2022) such as coreference resolution, relation extraction, and sentiment analysis.

To produce the dependency structure of a sentence, the contextual information is of great importance to achieve good model performance. Thus, most recent studies (Dozat and Manning, 2017; Zhou and Zhao, 2019; Zhou et al., 2020a,b; Mrini et al., 2020; Zhang et al., 2021) leverage advanced encoders (e.g., bi-LSTM, Transformer (Vaswani et al., 2017)) to model the contextual information of the input and obtain outstanding performance. In addition, because leveraging different models to obtain better results is an important technique for many NLP tasks (Juraska et al., 2018; Kobayashi, 2018; Kuwabara et al., 2020; Qin et al., 2021), many previous studies apply this technique to dependency parsing to further improve model performance. Under this paradigm, many studies utilize semi-supervised methods (e.g., self-training) to benefit from auto-processed extra data which is used to extract useful features (Smith and Eisner, 2007; Koo et al., 2008; Bansal and Klein, 2011; Ma and Xia, 2013; Kiperwasser and Goldberg, 2015; Yu and Bohnet, 2017) or training data (Spreyer and Kuhn, 2009; Rybak and Wróblewska, 2018; Rotman and Reichart, 2019). However, since the auto-generated parse tree is not always accurate, semi-supervised methods need to handle the noise with care to achieve better performance (Søgaard and Rishøj, 2010; Chen et al., 2018).

To address the noise issue, in this paper, we propose to apply pre-training and fine-tuning to enhance dependency parsing with the auto-parsed data generated by existing parsers. Although the effectiveness of pre-training and fine-tuning paradigm has been demonstrated to leverage extra data in many NLP tasks, it is still worth studying whether this paradigm works well for dependency

[†]Corresponding author.
[1]Our code is available at https://github.com/synlp/DMPar.

Proceedings of the 29th International Conference on Computational Linguistics, pages 5438–5449
October 12–17, 2022.

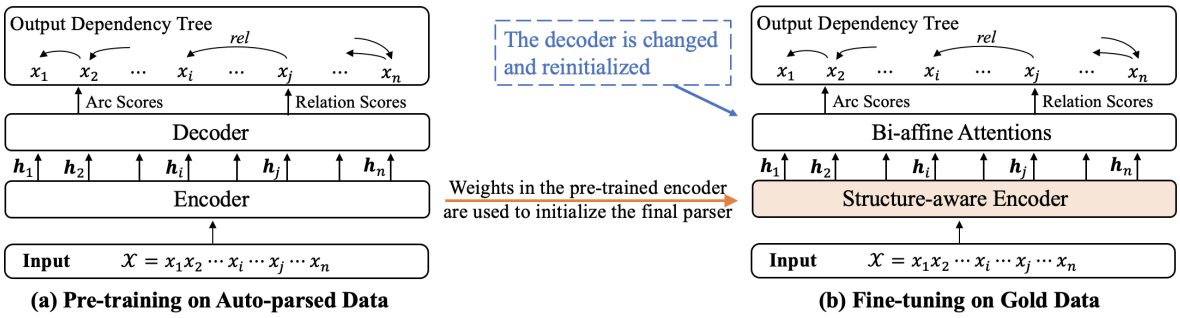

(a) Pre-training on Auto-parsed Data **(b) Fine-tuning on Gold Data**

Figure 1: Our parser is trained in two stages: pre-training with auto-parsed data (Figure 1(a)) and fine-tuning with gold dependency trees (Figure 1(b)). Fine-tuning uses the same encoder architecture as in pre-training but further adjusts its weights. In contrast, the decoder for fine-tuning is different from the one in pre-training and its weights are randomly initialized.

parsing. Specifically, we apply an auto-parser[2] to unlabeled data to obtain the auto-parsed dependencies, and then use the resulting data (with noise) to pre-train a structure-aware encoder, which is finally fine-tuned with the gold labels. The pre-training of the encoder follows exactly the same process of training a dependency parser with the same input and output, except that the labeled data are automatically generated, which significantly differs from the usual pre-training process that aims to predict the context words along the dependency path. In the fine-tuning stage, the weights (with structural information learnt from noisy auto-parsed data) of the pre-trained encoder is used to initialize the encoder of our final parser, whereas the final parser's decoder is initialized randomly before fine-tuning. In doing so, the encoder is able to learn the dependency information from large auto-parsed data (with noise) through pre-training and then use the information to enhance the performance of the final parser when it is fine-tuned on the gold parse trees. Compared with previous studies, our method offers a more flexible way to selectively learn from the auto-parsed data than the methods that take dependency parses (with noise) as fixed extra input features or training instances. Experimental results and further analyses on English benchmark datasets demonstrate the effectiveness and robustness of the proposed approach, which outperforms strong baselines under different settings with different dependency standards and model architectures used in pre-training and fine-tuning. The most interesting finding from this study is that the pre-training step only needs a small amount of data (e.g., two thousand auto-parsed sentences for the BERT-large encoder), to improve the performance of the resulting parser.

[2]E.g., Stanford CoreNLP Toolkits (Manning et al., 2014).

2 Training the Dependency Parser

In this study, we use neural graph-based dependency parsers, because they have achieved state-of-the-art performance on this task (Dozat and Manning, 2017; Zhou and Zhao, 2019; Mrini et al., 2020). Specifically, training our graph-based parser follows a two-stage procedure with pre-training and fine-tuning, where pre-training is performed on auto-parsed data with the same object to train a dependency parser, and the fine-tuning is conducted on the gold dependency trees with the pre-trained encoder (other modules in the final parser are freshly learned in fine-tuning).

Figure 1 shows the model architecture and the two-stage procedure to train our final dependency parser. In the following text, we firstly introduce the neural graph-based dependency parser and then illustrate the process to pre-train the structure-aware encoder.

2.1 Neural Graph-based Dependency Parser

Given the input sentence $\mathcal{X} = x_1 x_2 \cdots x_i \cdots x_n$ (x_i is the i-th word), conventional neural graph-based dependency parsers firstly obtain the hidden vector $\mathbf{h}_i$ for each word x_i from the encoder. Then, based on $\mathbf{h}_i$, for each word pair (x_i, x_j), the decoder of the parser computes $s^{arc}_{i,j}$ and $s^{rel}_{i,j}$ indicating the score for the directional dependency connection (arc) between x_i and x_j and the score for the dependency relation type $rel \in \mathcal{R}$ ($\mathcal{R}$ is the dependency type set) between them, respectively. Next, the parser applies the Eisner algorithm[3] (Eisner, 1996) to all $s^{arc}_{i,j}$ to predict the dependency tree $\widehat{\mathcal{T}}_0$ and assigns the connection between x_i and its

[3]The Eisner algorithm is only applied in inference. In training, the parser is optimized by comparing $s^{arc}_{i,j}$ and $s^{rel}_{i,j}$ with the gold standards using the cross-entropy loss function.

head x_j with the dependency type $\widehat{r}_{i,j}$ having the highest score $s^{rel}_{i,j}$.

It is worth noting that there are many ways to obtain the dependency arc scores $s^{arc}_{i,j}$ and the dependency relation scores $s^{rel}_{i,j}$. In doing so, bi-affine attentions (Dozat and Manning, 2017) (as illustrated in Figure 1(b)) is the most common and effective way to obtain $s^{arc}_{i,j}$ and $s^{rel}_{i,j}$. Specifically, for $s^{arc}_{i,j}$, it is computed by

$$\mathbf{h}^{arc\text{-}d}_i = \text{MLP}^{\,arc\text{-}d}(\mathbf{h}_i) \quad (1)$$

$$\mathbf{h}^{arc\text{-}h}_j = \text{MLP}^{\,arc\text{-}h}(\mathbf{h}_j) \quad (2)$$

$$s^{arc}_{i,j} = (\mathbf{h}^{arc\text{-}d}_i \oplus [1])^{\top}\mathbf{W}^{arc}(\mathbf{h}^{arc\text{-}h}_j \oplus [1]) \quad (3)$$

where $\text{MLP}^{\,arc\text{-}h}$ and $\text{MLP}^{\,arc\text{-}d}$ denote multi-layer perceptrons for the head and dependent representations, respectively; $\mathbf{W}^{arc}$ is a trainable matrix; $\oplus$ is the vector concatenation operation; $[1]$ is a one-dimensional unit vector which serves as a bias term for $\mathbf{h}^{arc\text{-}d}_i$ and $\mathbf{h}^{arc\text{-}h}_j$. Following the aforementioned process, $s^{rel}_{i,j}$ for a particular dependency type $rel \in \mathcal{R}$ is computed in a similar way.

2.2 Pre-training Structure-aware Encoder

To leverage existing parsers and unlabeled data, we replace the original encoder with a structure-aware encoder. The encoder is pre-trained with dependency trees generated from an existing parser, following the same but simplified procedure (as shown in Figure 1(a) without using bi-affine attentions) of training a parser. That is, in the pre-training procedure, we use

$$s^{arc}_{i,j} = \mathbf{h}^{\top}_i\mathbf{W}^{arc}\mathbf{h}_j, \;\; \mathbf{s}^{r}_{i,j} = \mathbf{W}^{rel}(\mathbf{h}_i \oplus \mathbf{h}_j) \quad (4)$$

to compute the arc score $s^{arc}_{i,j}$ and the relation score vector $\mathbf{s}^{r}_{i,j}$ over all dependency relation types. Herein, each dimension of $\mathbf{s}^{r}_{i,j}$ corresponds to a particular dependency relation type in $\mathcal{R}$ and $\mathbf{W}^{arc}$ and $\mathbf{W}^{rel}$ denote two trainable matrices.

Once the model is pre-trained, we get rid of the $\mathbf{W}^{arc}$ and $\mathbf{W}^{rel}$ and combine the resulting encoder with a new randomly initialized bi-affine attention module to construct our final dependency parser (illustrated in Figure 1(b)) for fine-tuning.

Through pre-training, the encoder is able to learn dependency information from the auto-parsed data (with noise). Meanwhile, because the decoder (i.e., the bi-affine attentions) of the final parser is changed and randomly initialized without using the decoder parameters (i.e., $\mathbf{W}^{arc}$ and $\mathbf{W}^{rel}$ in Eq. (4)) obtained from pre-training, our final parser is able to optimize its parameters based on the gold standard trees. Therefore, by using the auto-parsed and the gold training data in different stages (i.e., pre-training and fine-tuning, respectively), the noise in the auto-parsed data is carefully addressed: errors learnt from the pre-training stage can be "fixed" in the fine-tuning stage. In contrast, many existing semi-supervised approaches train the final parser on the combination of auto-parsed and gold training data, which could be risky.

Datasets		Sent. #	Token #	ASL
PTB	Train	40K	950K	23.9
	Dev	2K	40K	23.6
	Test	2K	57K	23.5
UD	Train	13K	205K	16.3
	Dev	2K	25K	12.6
	Test	2K	25K	12.1
Brown (Full)		24K	458K	19.0
English Wiki		92M	2,380M	22.3

Table 1: The number of sentences, tokens, and the averaged sentence length (ASL) of PTB, UD, Brown, and English Wiki used in our experiments.

3 Experiments

3.1 Datasets

In the experiments, we use English Wiki (with 92M sentences and 2,380M tokens) as the raw data for pre-training. We follow previous studies (Dozat and Manning, 2017; Zhou and Zhao, 2019; Mrini et al., 2020) to use English Penn Treebank (PTB)[4] (Marcus et al., 1993) converted by version 3.3.0 of the Stanford Dependency converter[5] as the benchmark dataset, which is further split into train/dev/test sets. In addition, we use Brown corpus (Marcus et al., 1993) and the English Web Treebank of Universal Dependencies (UD)[6] (Nivre et al., 2016). Herein, the dependency parses of Brown are obtained in the same process as PTB and the dependency parsing standard used in PTB and Brown is the Stanford typed dependencies (the Stanford standard) (De Marneffe and Manning, 2008); while UD follows a different standard named the UD dependency parsing standard[7]. The statistics (i.e., the number of sentences, tokens, and

[4] https://catalog.ldc.upenn.edu/LDC99T42.

[5] https://stanfordnlp.github.io/CoreNLP.

[6] We use the version 2.9 of UD obtained from https://universaldependencies.org/

[7] https://universaldependencies.org/u/overview/syntax.html

Pre-training	Fine-tuning	Testing
Wiki ($\mathcal{SD}$)	PTB Training ($\mathcal{SD}$)	PTB Test ($\mathcal{SD}$)
		Brown ($\mathcal{SD}$)
	UD Training ($\mathcal{UD}$)	UD Test ($\mathcal{UD}$)

Table 2: The datasets used in pre-training, fine-tuning, and testing. The dependency standard used in the datasets are illustrated in parentheses with $\mathcal{SD}$ and $\mathcal{UD}$ referring to the Stanford dependency standard and the UD standard, respectively.

Hyper-parameters	Values
Learning Rate	5e-6, **1e-5**, 3e-5
Warmup Rate	**0.1**, 0.2
Dropout Rate	**0.33**
Batch Size	**16**, 32

Table 3: The hyper-parameters tested in tuning our models. The best ones used in our final experiments are highlighted in boldface.

the averaged sentence length (ASL)) of all datasets, namely, PTB, Brown, UD, and English Wiki are reported in Table 1.

3.2 Obtaining the Auto-parsed Data

In the experiments, we propose to use existing NLP toolkits to obtain the auto-parsed Wiki data, because it not only allows us to benefit from existing tookits, but also is a good approximate of real-world applications where we want to build a good parser with existing toolkits. In addition, given PTB is one of the most widely used benchmark datasets for English dependency parsing, we want the auto-parsed data to follow exactly the same dependency standard as PTB, so that we can explore the effect of our approach when there is no gap between the standards in the auto-parsed and training data. However, many well-known existing dependency parsers (e.g., Stanford CoreNLP Toolkit (SCT) (Manning et al., 2014) and SpaCy[8]) follow a different standard.[9] Therefore, in the experiments, we employ a parsing-conversion process to obtain the dependency trees: we first use Berkeley Neural Parser[10] (Kitaev and Klein, 2018) to obtain the constituency trees of the Wiki text; then we convert them into dependency trees following the same process to obtain PTB (we denote this process as BNP-SD). Since the Berkeley Neural Parser is trained on the training set of PTB, this process ensures that the off-the-shelf dependency parser does not see the test data of PTB in training and the auto-parsed dependency trees follow the same dependency parsing standard as PTB.

[8] https://spacy.io/

[9] Specifically, the dependency parser of SCT is trained on UD and follows the UD dependency parsing standard; SpaCy is trained on OntoNotes 5 with the dependency trees converted from the corresponding constituency trees through ClearNLP, which does not exactly follow the Stanford standard.

[10] We use the model named "*benepar_en2*", which is downloaded from https://github.com/nikitakit/self-attentive-parser.

3.3 Settings

Table 2 summarizes the datasets used in pre-training, fine-tuning, and testing, where the dependency parsing standards for them are also illustrated in parentheses ($\mathcal{SD}$ and $\mathcal{UD}$ stand for the Stanford standard and the UD dependency parsing standard, respectively). Herein, we denote the experiments using Brown and UD in testing as **cross-domain** and **cross-standard** experiments, respectively, because Brown (for testing) and PTB (for fine-tuning) come from different domains whereas UD (for testing) and Wiki (for pre-training) use different dependency standards. Intuitively, the cross-standard setting with UD as the test set is most challenging as the auto-parsed Wiki data used for pre-training and the UD data used for fine-tuning and testing follow different dependency standards.

In addition, since our system design requires the final parser to use a new randomly initialized decoder before fine-tuning, it is interesting to explore the impact of the choice of the final parser decoder. Therefore, in addition to our final parser with biaffine attentions (BF) following the architecture in Figure 1(b) (we denote the final parsers as "**+BF**"), we also try final parsers without BF and following the architecture in Figure 1(a) (we denote it as "**-BF**"). It is worth noting that, the architectures used in pre-training and fine-tuning are different under "+BF" whereas they are the same under "-BF" (the parser used in pre-training does not use BF). Intuitively, "+BF" setting is more challenging because the patterns learned by the encoder from pre-training may not fit into the new architecture (i.e., the BF module) in the final parser and thus result in noise in the final parser.

3.4 Implementation Details

Since pre-trained language models have achieved outstanding performance in many NLP tasks (Devlin et al., 2019; Wu et al., 2019; Yang et al., 2019; Raffel et al., 2019; Chen et al., 2020; Tian et al.,

Models	PTB		Brown		UD
	UAS	LAS	UAS	LAS	UAS
BERT-base	96.43	94.74	94.25	91.43	65.14
BERT-large	96.70	94.96	94.60	91.62	65.31
XLNet-base	96.81	95.02	94.74	91.70	65.57
XLNet-large	96.97	95.15	95.01	91.98	65.86

Table 4: The performance of different dependency parsers obtained after pre-training, **without** fine-tuning. We only report UAS (which does not evaluate the relation types associated with the dependency connections) on UD because UD and the auto-parsed data use different dependency parsing standard.

2020; Diao et al., 2020; Sun et al., 2020; Wang and Tu, 2020; Song et al., 2021; Diao et al., 2021), we use two of them, namely, BERT and XLNet, as the encoder for pre-training. Specifically, we use the base and large versions of them following the default settings: the base models use 12 layers of self-attentions with 768 dimensional hidden vectors and the large models use 24 layers of self-attentions with 1024 dimenstional hidden vectors.[11] We train the models on the auto-parsed Wiki for one epoch (i.e., 2,800K steps in total) with the batch size set to 32. It is worth noting that, since English Wiki is used to train BERT and XLNet, it could be considered that we do not use additional data in experiments. In fine-tuning, we use the parameters in the encoder obtained from pre-training to initialize the encoder in our approach and randomly initialize all other trainable parameters. Following previous studies, we evaluate all models based on unlabeled attachment score (UAS) and labeled attachment score (LAS). Table 3 reports the hyper-parameters tested in training our models. We test all combinations of them for each model and use the one achieving the highest LAS score in our final experiments.

4 Results, Analyses, and Findings

4.1 Overall Performance

Since pre-training also follows the process to train a dependency parser, we report the performance of the dependency parsers obtained after pre-training on the test set of PTB, Brown, and UD in Table 4 for reference. For the final parsers, we report the mean and standard deviation[12] of them on the test set of all datasets in Table 5.

Overall, results on PTB, Brown, and UD demonstrate the effectiveness of our approach under different configurations (i.e., using the base and large versions of BERT and XLNet encoders, with and without BF), where consistent improvements are observed in most cases, even though BERT and XLNet baselines have already achieved good performance. Particularly, it is promising to observe that our approach works well on UD (i.e., the cross-standard setting), where the pre-trained models has rather low performance (e.g., according to Table 4, BERT-large achieves 65.31% UAS on UD test set after pre-training) before they are fine-tuned on the gold standard. This observation demonstrates the effectiveness of our approach in cases where other approaches (e.g., training on the combination of the auto-parsed data and the gold training data) may not work well owing to the poor quality of auto-parsed data. Besides, our final parser with BF also works well in most cases with its architecture differing from the one used in pre-training. It shows the effectiveness and robustness of our approach to leverage the structure-aware encoder obtained from a parser with a different architecture, where the representation obtained from such encoder may not fit into the final parser due to the differences of the architectures in pre-training and fine-tuning.

In addition, we compare our best performing model[13] using BERT-large and XLNet-large and BF with previous studies on the PTB test set and report the results in Table 6. Overall, our approach outperforms all previous graph-based approaches (i.e., the ones without the "†" mark) except Mrini et al. (2020) that leverage auto-generated part-of-speech (POS) tags. Particularly, our model outperforms Zhou and Zhao (2019) and Zhou et al. (2020a) that perform constituency and dependency parsing at the same time through head-driven phrase structure grammar (HPSG) parsing. Besides, compared with Mrini et al. (2020) who proposed label attention layer to enhance the study of Zhou and Zhao (2019) on HPSG parsing, our approach obtain inferior performance because we do not use the label attention layer or the auto-generated POS tags. Given that our approach outperforms Zhou and Zhao (2019) on the test set of PTB, the effectiveness of our approach for dependency parsing is still valid and promising.

[11] We download the cased version of BERT from https://github.com/google-research/bert and XLNet from https://github.com/zihangdai/xlnet.

[12] In experiments, we run (fine-tune) each model five times with different random seeds.

[13] We follow previous studies to compare our best performing model with their models.

Models	PTB				Brown (cross-domain)				UD (cross-standard)			
	-BF		+BF		-BF		+BF		-BF		+BF	
	μ	σ	μ	σ	μ	σ	μ	σ	μ	σ	μ	σ
BERT-base	94.65	0.06	94.70	0.05	91.26	0.07	91.46	0.08	89.78	0.08	89.09	0.09
+ Dep. Wiki	**95.06**	0.05	**95.30**	0.05	**91.56**	0.07	**91.76**	0.06	**90.57**	0.07	**90.39**	0.08
BERT-large	95.01	0.07	95.11	0.06	91.79	0.08	91.84	0.07	90.70	0.09	90.80	0.08
+ Dep. Wiki	**95.25**	0.06	**95.50**	0.09	**92.00**	0.07	**92.40**	0.08	**91.22**	0.07	**91.01**	0.09
XLNet-base	95.13	0.05	95.19	0.06	91.78	0.06	91.98	0.07	91.02	0.07	**91.50**	0.07
+ Dep. Wiki	**95.49**	0.04	**95.50**	0.05	**92.35**	0.05	**92.38**	0.08	**91.51**	0.06	91.39	0.07
XLNet-large	95.48	0.04	95.54	0.06	92.31	0.05	92.53	0.07	91.38	0.08	**92.30**	0.07
+ Dep. Wiki	**95.71**	0.05	**95.86**	0.05	**92.45**	0.06	**92.54**	0.07	**91.94**	0.07	91.70	0.08

Table 5: The mean μ and standard deviation σ of LAS of our approaches (**with** the fine-tuning of the structure-aware encoder) and the baseline models with different configurations (i.e., the ones using base or large BERT/XLNet with (+BF) and without (-BF) bi-affine attentions) on the test set of PTB, Brown, and UD.

Models	UAS	LAS
Dozat and Manning (2017)	95.74	94.08
*Dozat and Manning (2017) (BERT)	96.64	95.11
*Zhou and Zhao (2019) (BERT)	97.00	95.43
*Zhou and Zhao (2019) (XLNet)	97.20	95.72
*Zhou et al. (2020a) (XLNet)	97.23	95.65
*Zhou et al. (2020b) (LIMIT-BERT)	97.14	95.44
*Mrini et al. (2020) (XLNet + POS)	**97.42**	**96.26**
*Wang and Tu (2020) (BERT)	96.91	95.34
Zhang et al. (2021) (BERT)	96.64	95.09
Mohammadshahi and Henderson (2021) (BERT)	96.66	95.01
*†Fernández-González and Gómez-Rodríguez (2021) (BERT)	97.05	95.47
*†Yang and Tu (2021) (BERT)	97.24	95.73
BNP-SD (Kitaev and Klein, 2018)	96.03	94.03
*Ours (BERT-large)	97.06	95.60
*Ours (XLNet-large)	97.30	95.92

Table 6: Comparison (UAS and LAS) of our approach with previous studies. "*" denotes the models using the large version of BERT and XLNet; "†" marks the parsers that do not use the graph-based approaches.

4.2 The Effect of System Design

Our parser is different from many of the previous studies in two ways: (1) the auto-parsed data is used for pre-training only, (2) the fine-tuning step uses a different decoder from the one used in pre-training, whose weights are initialized randomly.

To determine the impact of those decisions, we build three more parsers for comparison. The first one uses the architecture in Figure 1(a) and is trained with the union of the auto-parsed and gold standard data (we denote this approach as "Union") without the fine-tuning step. The second parser ("Fine-tuning") is pre-trained with the auto-parsed data and fine-tuned with gold dependency trees, but the two stages use the same decoder (as in Figure 1(a)) and the decoder's weights for fine-tuning are initialized with the weights from pre-training. The third one ("Randomize") is the same as the second one but the weights of the decoder derived from pre-training are thrown away before fine-tuning. The "Randomize" system differs from our final parser only in that our final parser uses a different decoder in the fine-tuning stage.

All aforementioned three approaches use BERT-base encoder. For auto-parsed data, we randomly select sentences from English Wiki where the number of selected sentences equals to the number of sentences in the training set of different datasets (i.e., 40K auto-parsed sentences for PTB and Brown, and 13K auto-parsed sentences for UD).[14] For each approach, we run it five times with different random data and report the average results (LAS for PTB and Brown, and UAS[15] for UD) of them, as well as the average results of BERT-base baseline and our final parser, in Table 7.

It is observed that "Randomize" consistently outperforms the other two approaches on the test set of all datasets. Particularly, for cross-standard settings on UD, because the auto-parsed data and the UD data use different dependency standards, the quality of the auto-parsed data can be considered relatively low with respect to the UD standard (this can be confirmed by the low model performance of the parser pre-trained on auto-parsed data on the test set of UD (see Table 4)). Under this setting, the "Union" and "Fine-tuning" even achieve inferior results (the results are underlined) compared with the BERT-base baseline, because they suffer from the gap between the dependency standards of auto-parsed data and gold data. On the contrary,

[14] We also tested other numbers of selected auto-parsed sentences and obtain similar observations.

[15] We report UAS (where the dependency type are ignored in evaluation) for UD because the auto-parsed data and the UD use different dependency standards.

Approach	PTB	Brown	UD
BERT-base	94.65	91.26	92.86
+ Union	94.69	91.30	83.51
+ Fine-tuning	94.76	91.38	92.14
+ Randomize	**94.94**	**91.47**	**93.10**
Our Final Parser	95.19	91.60	92.98

Table 7: The average performance (LAS for PTB and Brown, UAS for UD) of different approaches using BERT-base encoder with the scores lower than the baseline highlighted by underlines. "Union" refers to the model that is trained on the union of auto-parsed and gold data; "Fine-tuning" denotes the model that is pre-trained on auto-parsed data and then further fine-tuned on the gold data; "Randomize" is the same as "Fine-tuning" except that the weights of the decoder is randomly initialized before fine-tuning. The best scores among "Union", "Fine-tuning", and "Randomize" are highlighted by boldface. All three models and the baseline use the architecture in Figure 1(a) and the only difference between "Randomize" and our final parser is the architecture of decoder in fine-tuning.

"Randomize" is able to carefully address the noise in the auto-parsed data by using the auto-parsed and the gold standard data in different stages and getting rid of the pre-trained decoder (which may learn the noise) and randomly initializing a new one before fine-tuning.

4.3 The Size of Auto-parsed Data

An essential question for evaluating our approach is that how many data (auto-parsed with noise) are required to improve the parsers with various model sizes. To answer its question, we pre-train four models with variant sizes, i.e., 6-layer BERT-base (55M parameters), 12-layer BERT-base (110M parameters), 18-layer BERT-large (252M parameters), and 24-layer BERT-large (336M parameters), on different amount of randomly selected auto-parsed Wiki data. Figure 2 illustrates the averaged[16] improvement (LAS) of four different models over their corresponding baselines on three test sets: (a) PTB, (b) Brown, and (c) UD, with respect to the ratio of the pre-training sentence number to the model size. It is interesting that, for the in-domain setting (i.e., when both training and test sets are from the PTB), the zero points of different curves for all models are roughly the same, i.e., around 0.3, which means that the pre-training only needs a little bit over 300 sentences for every 50M parameters in a parser to ensure an improvement (e.g., for 24-layer BERT-large, it only requires two thousand auto-parsed sentences to obtain a better-than-original parser).

This finding is highly encouraging since it only needs a small amount of auto-parsed data (compared to 92M sentences in English Wiki) to improve a large model. Similar observations can be drawn for cross-domain and cross-standard settings on Brown and UD datasets, where more (since there are gaps between the pre-training and the fine-tuning data) but still limited auto-parsed data is required to ensure that improvement. Particularly, for each dataset, we found there exists a rather stable ratio for different models, e.g., 0.3 for PTB, 0.6 for Brown, etc., which is a meaningful guidance to improve parsers' performance regarding to their parameters. An explanation to this observation is that structural data is useful to update representation models (Gubbins and Vlachos, 2013; Levy and Goldberg, 2014; Zhou et al., 2020b) so that a limited amount (w.r.t. model size) could greatly affect model performance especially when they are applied on structure-prediction tasks such as parsing.

[16]We run the experiment for each model ten times with different random data to guarantee the results are trustworthy.

Data Auto-parsed by	PTB	Brown	UD
N/A	94.78	91.56	90.10
BNP-SD (94.03)	95.37	91.83	90.48
Parser I (94.78)	94.83	91.58	90.42
Parser II (95.26)	95.38	91.84	91.70

Table 8: Performance (LAS) comparison of a model (using BERT-base with BF) pre-trained on auto-parsed English Wiki from different parsers. N/A refers to no pre-training, Parser I and Parser II are the parsers (with BF) trained on PTB using BERT-base and XLNet-base, respectively. The LAS from all parsers on the PTB test set are reported in parentheses for reference.

4.4 The Choice of Existing Parsers

Another factor that may affect the performance of our systems is the off-the-shelf parser used to produce auto-parsed data. To assess the effect of the parser on the performance of the final systems, we experimented with two more parsers from our baselines, i.e., the ones using BERT-base (**Parser I**) and XLNet-base (**Parser II**) trained on PTB, in addition to BNP-SD as described in Section 3.2: Table 8 reports the LAS of models with or without pre-training. While pre-training improves the performance with auto-parsed data from all four

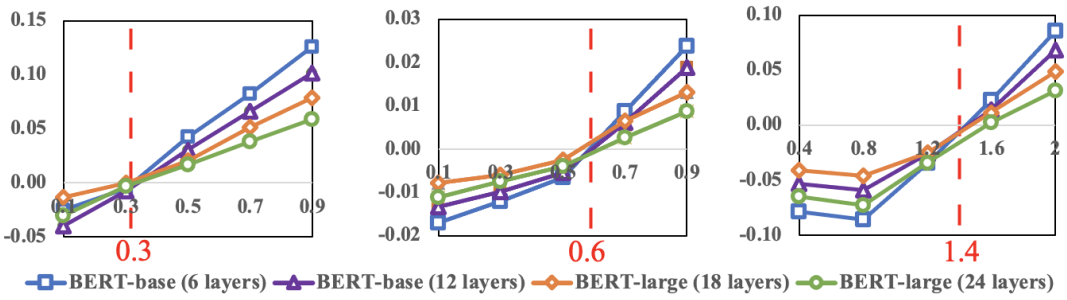

(a) PTB (b) Brown (c) UD

Figure 2: The improvement (LAS) of four models over their baselines on three test sets. The X-axis is the number of auto-parsed sentences used for pre-training divided by the number of model parameters, and then multiplied by 50,000 (to make the scale more readable).

Models	PTB	Brown	UD
BERT-base	94.70	91.46	89.09
+ Dep. Wiki (-BF)	95.30	91.76	90.39
+ Dep. Wiki (+BF)	**95.35**	**91.80**	**90.43**
XLNet-base	95.19	91.98	91.50
+ Dep. Wiki (-BF)	95.50	92.38	91.39
+ Dep. Wiki (+BF)	**95.54**	**92.40**	**91.97**

Table 9: The average LAS of final parsers (with BF) using BERT-base and XLNet-base encoders, with (+) and without (-) using BF in pre-training.

parsers, the improvement with Parser I is less significant than the other parsers because it is more similar to the final parser in terms of the model architecture.

This finding is quite intuitive and matches the observation in Surdeanu and Manning (2010) that the diversity between different parsers plays an important role in improving model performance when combining them. Specifically, when being evaluated on PTB and Brown, Parser I tends to make the same mistakes as our final parser since they use exactly the same architecture (i.e., Transformer), which reduces the benefits of pre-training with auto-parsed data. In contrast, BNP-SD (which uses ELMo embeddings (Peters et al., 2018)) and parser II (which is based on Transformer-XL (Dai et al., 2019)) use different resources and architectures so that our approach with BERT-base encoder can learn from their auto-parsed data. In addition, when our final parser is evaluated on UD, since UD uses a different dependency standard, Parser I and our final parser no longer make the "same" mistakes, which results in more improvement on UD.

4.5 The Decoder Used in Pre-training

In the main experiments, we pre-train the parser without using BF. To explore its effect, we conduct an ablation study where BF is used in pre-training. Table 9 reports the average LAS of different final parsers (with BF) using BERT-base and XLNet-base encoders, where BF is used (i.e., "+BF") or not used (i.e., "-BF") in pre-training. It is observed that using BF in pre-training results in similar performance compared with the settings where BF is not used. It demonstrates the robustness of our approach where the architecture of the final parser (with BF) does not need necessary to be identical to the one (without BF) in pre-training to obtain promising improvement over the baselines. In addition, it is worth-noting that for experiments on UD with XLNet-base, "+ Dep. Wiki (+BF)" outperforms the baseline model whereas "+ Dep. Wiki (-BF)" fails to do so. The explanation could be the following. For "+ Dep. Wiki (+BF)", the only gap between pre-training and fine-tuning is the dependency standard, whereas "+ Dep. Wiki (-BF)" faces an additional gap that the architectures used in pre-training and fine-tuning are different. Therefore, "+ Dep. Wiki (-BF)" fails to overcome the two gaps and thus results in inferior results compared with the XLNet baseline. On the other hand, when BF is also used in pre-training, the gap between the architectures does not exist, which allows our final parser to obtain a higher performance.

5 Related Work

Recent studies for dependency parsing use advanced architectures (e.g., bi-LSTM, BERT) to capture contextual information so as to achieve outstanding performance (Shen et al., 2021; Zhang et al., 2021; Yang and Tu, 2021; Li et al., 2021). To further improve dependency parsing, approaches

such as bi-affine attentions (Dozat and Manning, 2017; Attardi et al., 2021; Xu and Koehn, 2021), HPSG parsing (Zhou and Zhao, 2019; Zhou et al., 2020a; Mrini et al., 2020), TreeCRF (Zhang et al., 2020) are further applied. Besides, to improve model performance, there are studies that use existing dependency parsers and auto-parsed data through model ensemble (Attardi and Dell'Orletta, 2009; Surdeanu and Manning, 2010; Che et al., 2018) or semi-supervised approaches (Sagae and Lavie, 2006; Chen et al., 2009; Prokopidis and Papageorgiou, 2014; Yu and Bohnet, 2017; Zhang et al., 2021; Wagner and Foster, 2021).

Compared to previous studies that use auto-parsed data, our approach differs in several ways. First, our encoder is structure-aware as it is pre-trained with dependency trees. Second, because auto-parsed data is noisy and may use dependency standard different from that of the test data (in the cross-standard setting), it is used in pre-training only. In contrast, training a parser on the union of auto-parsed data and gold data would not work well, especially in the cross-standard setting, as shown in Table 7. Third, the decoder of the fine-tuning stage starts with randomly initialized weights, instead of with the weights learned from the pre-training stage, thus ensuring that the decoder in the final parser will not be affected by the noisy auto-parsed data.

6 Conclusion

In this study, we propose a simple and effective solution to improve dependency parser through pre-training on auto-parsed data. In doing so, the encoder is able to learn structural information from the auto-parsed data in pre-training. During fine-tuning, a different decoder is used and its weights initialized randomly, thus reducing the impact of errors in the auto-parsed data. We have run a large number of experiments under different settings (e.g., cross-domain vs. cross-standard, -BF vs. +BF, different parsers used to parse Wiki) and shown that our approach outperforms strong baselines and many previous studies under those settings. Furthermore, pre-training needs only a small amount of auto-parsed data (e.g., 2K sentences for a BERT-large based parser on the PTB test set) to ensure improvement over strong baselines.

References

Giuseppe Attardi and Felice Dell'Orletta. 2009. Reverse Revision and Linear Tree Combination for Dependency Parsing. In *Proceedings of Human Language Technologies: The 2009 Annual Conference of the North American Chapter of the Association for Computational Linguistics, Companion Volume: Short Papers*, pages 261–264, Boulder, Colorado.

Giuseppe Attardi, Daniele Sartiano, and Maria Simi. 2021. Biaffine Dependency and Semantic Graph Parsing for Enhanced Universal Dependencies. In *Proceedings of the 17th International Conference on Parsing Technologies and the IWPT 2021 Shared Task on Parsing into Enhanced Universal Dependencies (IWPT 2021)*, pages 184–188, Online.

Mohit Bansal and Dan Klein. 2011. Web-scale Features for Full-scale Parsing. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, pages 693–702, Portland, Oregon, USA.

Dongfeng Cai, Yonghua Hu, Xuelei Miao, and Yan Song. 2009. Dependency Grammar Based English Subject-Verb Agreement Evaluation. In *Proceedings of the 23rd Pacific Asia Conference on Language, Information and Computation, Volume 1*, pages 63–71, Hong Kong.

Wanxiang Che, Yijia Liu, Yuxuan Wang, Bo Zheng, and Ting Liu. 2018. Towards Better UD Parsing: Deep Contextualized Word Embeddings, Ensemble, and Treebank Concatenation. In *Proceedings of the CoNLL 2018 Shared Task: Multilingual Parsing from Raw Text to Universal Dependencies*, pages 55–64, Brussels, Belgium.

Dong-Dong Chen, Wei Wang, Wei Gao, and Zhi-Hua Zhou. 2018. Tri-net for Semi-Supervised Deep Learning. In *Proceedings of the Twenty-Seventh International Joint Conference on Artificial Intelligence, IJCAI-18*, pages 2014–2020.

Guimin Chen, Yuanhe Tian, and Yan Song. 2020. Joint Aspect Extraction and Sentiment Analysis with Directional Graph Convolutional Networks. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 272–279.

Wenliang Chen, Jun'ichi Kazama, Kiyotaka Uchimoto, and Kentaro Torisawa. 2009. Improving Dependency Parsing with Subtrees from Auto-Parsed Data. In *Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing*, pages 570–579, Singapore.

Zihang Dai, Zhilin Yang, Yiming Yang, Jaime Carbonell, Quoc Le, and Ruslan Salakhutdinov. 2019. Transformer-XL: Attentive Language Models beyond a Fixed-Length Context. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 2978–2988, Florence, Italy.

Marie-Catherine De Marneffe and Christopher D Manning. 2008. Stanford Typed Dependencies Manual. Technical report, Technical report, Stanford University.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186.

Shizhe Diao, Jiaxin Bai, Yan Song, Tong Zhang, and Yonggang Wang. 2020. ZEN: Pre-training Chinese Text Encoder Enhanced by N-gram Representations. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 4729–4740.

Shizhe Diao, Ruijia Xu, Hongjin Su, Yilei Jiang, Yan Song, and Tong Zhang. 2021. Taming Pre-trained Language Models with N-gram Representations for Low-Resource Domain Adaptation. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 3336–3349, Online.

Timothy Dozat and Christopher D. Manning. 2017. Deep Biaffine Attention for Neural Dependency Parsing. In *5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Conference Track Proceedings*.

Jason M. Eisner. 1996. Three New Probabilistic Models for Dependency Parsing: An Exploration. In *COLING 1996 Volume 1: The 16th International Conference on Computational Linguistics*.

Daniel Fernández-González and Carlos Gómez-Rodríguez. 2021. Dependency Parsing with Bottom-up Hierarchical Pointer Networks. *arXiv preprint arXiv:2105.09611*.

Joseph Gubbins and Andreas Vlachos. 2013. Dependency Language Models for Sentence Completion. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*, pages 1405–1410, Seattle, Washington, USA.

Zhijiang Guo, Yan Zhang, and Wei Lu. 2019. Attention Guided Graph Convolutional Networks for Relation Extraction. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 241–251.

Binxuan Huang and Kathleen M Carley. 2019. Syntax-Aware Aspect Level Sentiment Classification with Graph Attention Networks. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 5472–5480.

Juraj Juraska, Panagiotis Karagiannis, Kevin Bowden, and Marilyn Walker. 2018. A Deep Ensemble Model with Slot Alignment for Sequence-to-sequence Natural Language Generation. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 152–162, New Orleans, Louisiana.

Eliyahu Kiperwasser and Yoav Goldberg. 2015. Semi-supervised Dependency Parsing using Bilexical Contextual Features from Auto-Parsed Data. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 1348–1353, Lisbon, Portugal.

Nikita Kitaev and Dan Klein. 2018. Constituency Parsing with a Self-Attentive Encoder. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 2676–2686, Melbourne, Australia.

Hayato Kobayashi. 2018. Frustratingly Easy Model Ensemble for Abstractive Summarization. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 4165–4176, Brussels, Belgium.

Terry Koo, Xavier Carreras, and Michael Collins. 2008. Simple Semi-supervised Dependency Parsing. In *Proceedings of ACL-08: HLT*, pages 595–603, Columbus, Ohio.

Ryosuke Kuwabara, Jun Suzuki, and Hideki Nakayama. 2020. Single Model Ensemble using Pseudo-Tags and Distinct Vectors. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 3006–3013, Online.

Omer Levy and Yoav Goldberg. 2014. Dependency-Based Word Embeddings. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 302–308, Baltimore, Maryland.

Ying Li, Meishan Zhang, Zhenghua Li, Min Zhang, Zhefeng Wang, Baoxing Huai, and Nicholas Jing Yuan. 2021. APGN: Adversarial and Parameter Generation Networks for Multi-Source Cross-Domain Dependency Parsing. In *Findings of the Association for Computational Linguistics: EMNLP 2021*, pages 1724–1733, Punta Cana, Dominican Republic.

Xuezhe Ma and Fei Xia. 2013. Dependency Parser Adaptation with Subtrees from Auto-Parsed Target Domain Data. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 585–590, Sofia, Bulgaria.

Christopher Manning, Mihai Surdeanu, John Bauer, Jenny Finkel, Steven Bethard, and David McClosky. 2014. The Stanford CoreNLP Natural Language

Processing Toolkit. In *Proceedings of 52nd Annual Meeting of the Association for Computational Linguistics: System Demonstrations*, pages 55–60, Baltimore, Maryland.

Mitchell P. Marcus, Beatrice Santorini, and Mary Ann Marcinkiewicz. 1993. Building a Large Annotated Corpus of English: The Penn Treebank. *Computational Linguistics*, 19(2):313–330.

Alireza Mohammadshahi and James Henderson. 2021. Recursive Non-Autoregressive Graph-to-Graph Transformer for Dependency Parsing with Iterative Refinement. *Transactions of the Association for Computational Linguistics*, 9:120–138.

Khalil Mrini, Franck Dernoncourt, Quan Hung Tran, Trung Bui, Walter Chang, and Ndapa Nakashole. 2020. Rethinking Self-Attention: Towards Interpretability in Neural Parsing. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 731–742, Online.

Yuyang Nie, Yuanhe Tian, Yan Song, Xiang Ao, and Xiang Wan. 2020. Improving Named Entity Recognition with Attentive Ensemble of Syntactic Information. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 4231–4245.

Joakim Nivre, Marie-Catherine De Marneffe, Filip Ginter, Yoav Goldberg, Jan Hajic, Christopher D Manning, Ryan McDonald, Slav Petrov, Sampo Pyysalo, Natalia Silveira, et al. 2016. Universal Dependencies v1: A Multilingual Treebank Collection. In *Proceedings of the Tenth International Conference on Language Resources and Evaluation (LREC 2016)*, pages 1659–1666.

Matthew Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer. 2018. Deep Contextualized Word Representations. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 2227–2237, New Orleans, Louisiana.

Prokopis Prokopidis and Haris Papageorgiou. 2014. Experiments for Dependency Parsing of Greek. In *Proceedings of the First Joint Workshop on Statistical Parsing of Morphologically Rich Languages and Syntactic Analysis of Non-Canonical Languages*, pages 90–96, Dublin, Ireland.

Han Qin, Guimin Chen, Yuanhe Tian, and Yan Song. 2021. Improving Arabic Diacritization with Regularized Decoding and Adversarial Training. In *Proceedings of the Joint Conference of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing*.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J Liu. 2019. Exploring the Limits of Transfer Learning with a Unified Text-to-text Transformer. *arXiv preprint arXiv:1910.10683*.

Guy Rotman and Roi Reichart. 2019. Deep Contextualized Self-training for Low Resource Dependency Parsing. *Transactions of the Association for Computational Linguistics*, 7:695–713.

Piotr Rybak and Alina Wróblewska. 2018. Semi-Supervised Neural System for Tagging, Parsing and Lematization. In *Proceedings of the CoNLL 2018 Shared Task: Multilingual Parsing from Raw Text to Universal Dependencies*, pages 45–54, Brussels, Belgium.

Kenji Sagae and Alon Lavie. 2006. Parser Combination by Reparsing. In *Proceedings of the Human Language Technology Conference of the NAACL, Companion Volume: Short Papers*, pages 129–132, New York City, USA.

Yikang Shen, Yi Tay, Che Zheng, Dara Bahri, Donald Metzler, and Aaron Courville. 2021. StructFormer: Joint Unsupervised Induction of Dependency and Constituency Structure from Masked Language Modeling. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 7196–7209, Online.

David A. Smith and Jason Eisner. 2007. Bootstrapping Feature-rich Dependency Parsers with Entropic Priors. In *Proceedings of the 2007 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning (EMNLP-CoNLL)*, pages 667–677, Prague, Czech Republic.

Anders Søgaard and Christian Rishøj. 2010. Semi-supervised Dependency Parsing using Generalized Tri-training. In *Proceedings of the 23rd International Conference on Computational Linguistics (Coling 2010)*, pages 1065–1073, Beijing, China.

Yan Song, Tong Zhang, Yonggang Wang, and Kai-Fu Lee. 2021. ZEN 2.0: Continue Training and Adaption for N-gram Enhanced Text Encoders. *arXiv preprint arXiv:2105.01279*.

Kathrin Spreyer and Jonas Kuhn. 2009. Data-driven Dependency Parsing of New Languages Using Incomplete and Noisy Training Data. In *Proceedings of the Thirteenth Conference on Computational Natural Language Learning (CoNLL-2009)*, pages 12–20, Boulder, Colorado.

Emma Strubell, Patrick Verga, Daniel Andor, David Weiss, and Andrew McCallum. 2018. Linguistically-Informed Self-Attention for Semantic Role Labeling. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 5027–5038, Brussels, Belgium.

Yu Sun, Shuohuan Wang, Yukun Li, Shikun Feng, Hao Tian, Hua Wu, and Haifeng Wang. 2020. ERNIE 2.0: A Continual Pre-Training Framework for Language Understanding. *Proceedings of the AAAI Conference on Artificial Intelligence*, 34(05):8968–8975.

Mihai Surdeanu and Christopher D. Manning. 2010. Ensemble Models for Dependency Parsing: Cheap and Good? In *Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, pages 649–652, Los Angeles, California.

Yuanhe Tian, Wang Shen, Yan Song, Fei Xia, Min He, and Kenli Li. 2020. Improving Biomedical Named Entity Recognition with Syntactic Information. *BMC Bioinformatics*, 21:1471–2105.

Yuanhe Tian, Yan Song, and Fei Xia. 2022. Improving Relation Extraction through Syntax-induced Pretraining with Dependency Masking. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics*.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention Is All You Need. In *Advances in neural information processing systems*, pages 5998–6008.

Joachim Wagner and Jennifer Foster. 2021. Revisiting Tri-training of Dependency Parsers. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 9457–9473, Online and Punta Cana, Dominican Republic.

Xinyu Wang and Kewei Tu. 2020. Second-Order Neural Dependency Parsing with Message Passing and End-to-End Training. In *Proceedings of the 1st Conference of the Asia-Pacific Chapter of the Association for Computational Linguistics and the 10th International Joint Conference on Natural Language Processing*, pages 93–99, Suzhou, China.

Zhaofeng Wu, Yan Song, Sicong Huang, Yuanhe Tian, and Fei Xia. 2019. WTMED at MEDIQA 2019: A Hybrid Approach to Biomedical Natural Language Inference. In *Proceedings of the 18th BioNLP Workshop and Shared Task*, pages 415–426, Florence, Italy.

Haoran Xu and Philipp Koehn. 2021. Zero-Shot Cross-Lingual Dependency Parsing through Contextual Embedding Transformation. In *Proceedings of the Second Workshop on Domain Adaptation for NLP*, pages 204–213, Kyiv, Ukraine.

Songlin Yang and Kewei Tu. 2021. Headed Span-based Projective Dependency Parsing. *arXiv preprint arXiv:2108.04750*.

Zhilin Yang, Zihang Dai, Yiming Yang, Jaime Carbonell, Russ R Salakhutdinov, and Quoc V Le. 2019. XLNet: Generalized Autoregressive Pretraining for Language Understanding. In *Advances in Neural Information Processing Systems 32*, pages 5753–5763.

Juntao Yu and Bernd Bohnet. 2017. Dependency Language Models for Transition-based Dependency Parsing. In *Proceedings of the 15th International Conference on Parsing Technologies*, pages 11–17, Pisa, Italy.

Hongming Zhang, Yan Song, Yangqiu Song, and Dong Yu. 2019. Knowledge-aware Pronoun Coreference Resolution. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 867–876, Florence, Italy.

Xudong Zhang, Joseph Le Roux, and Thierry Charnois. 2021. Strength in Numbers: Averaging and Clustering Effects in Mixture of Experts for Graph-Based Dependency Parsing. In *Proceedings of the 17th International Conference on Parsing Technologies and the IWPT 2021 Shared Task on Parsing into Enhanced Universal Dependencies (IWPT 2021)*, pages 106–118, Online.

Yu Zhang, Zhenghua Li, and Min Zhang. 2020. Efficient Second-Order TreeCRF for Neural Dependency Parsing. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 3295–3305, Online.

Junru Zhou, Zuchao Li, and Hai Zhao. 2020a. Parsing All: Syntax and Semantics, Dependencies and Spans. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 4438–4449, Online.

Junru Zhou, Zhuosheng Zhang, Hai Zhao, and Shuailiang Zhang. 2020b. LIMIT-BERT : Linguistics Informed Multi-Task BERT. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 4450–4461, Online.

Junru Zhou and Hai Zhao. 2019. Head-Driven Phrase Structure Grammar Parsing on Penn Treebank. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 2396–2408, Florence, Italy.

Simple and Effective Graph-to-Graph Annotation Conversion

Yuxuan Wang[♦,*], **Zhilin Lei**[♡*], **Yuqiu Ji**[♡], **Wanxiang Che**[♡†]
[♦]Zhejiang Lab
[♡]Research Center for Social Computing and Information Retrieval,
Harbin Institute of Technology
yxwang@zhejianglab.com, {zllei, yqji, car}@ir.hit.edu.cn

Abstract

Annotation conversion is an effective way to construct datasets under new annotation guidelines based on existing datasets with little human labour. Previous work has been limited in conversion between tree-structured datasets and mainly focused on feature-based models which are not easily applicable to new conversions. In this paper, we propose two simple and effective graph-to-graph annotation conversion approaches, namely Label Switching and Graph2Graph Linear Transformation, which use pseudo data and inherit parameters to guide graph conversions respectively. These methods are able to deal with conversion between graph-structured annotations and require no manually designed features. To verify their effectiveness, we manually construct a graph-structured parallel annotated dataset and evaluate the proposed approaches on it as well as other existing parallel annotated datasets. Experimental results show that the proposed approaches outperform strong baselines with higher *conversion score*. To further validate the quality of converted graphs, we utilize them to train the target parser and find graphs generated by our approaches lead to higher *parsing score* than those generated by the baselines.[1]

1 Introduction

While tree-structured representations have dominated parsing for the last decade, graph-structured datasets are receiving growing interest in recent years (Oepen et al., 2019, 2020). Over the last few years, an increasing number of graph-structured datasets have become available. Some of them, such as DM corpora from the SemEval 2015 task 18 dataset (Oepen et al., 2015) and AMRBank (Banarescu et al., 2013), are manually annotated. Some

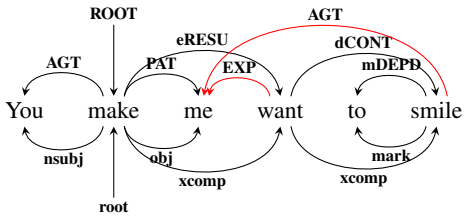

Figure 1: Example of annotation conversion from Universal Dependency Tree (bottom) to our Semantic Dependency Graph (top).

others, such as the Enhanced English Universal Dependencies dataset (Schuster and Manning, 2016), are converted from existing dataset with manually designed rules. As illustrated in Figure 1, the Semantic Dependency Graph at the top is converted from the Universal Dependecy Tree at the bottom.

Obviously, in the dataset construction process under a new annotation guideline, it would be extremely expensive to annotate the whole dataset manually. Although rule-based conversion needs no human labour for annotation, it requires expertise to design the rules, which could be difficult if the new guideline is vastly different from the old one. Therefore, it would be efficient and attractive to exploit existing datasets and learn a transduction that converts them into a new guideline. The converted dataset under the new guideline could be used in model training or further refined by human annotators to construct a high-quality dataset.

Such conversion has been studied in a line of research that exploits heterogeneous treebanks to boost parsing performance, where the approach is typically referred to as treebank conversion (Li et al., 2013; Jiang et al., 2018; Seddah et al., 2018). In their cases, two existing heterogeneous treebanks (tree-structured datasets) on different texts are available. The goal is to convert a source treebank into annotation under a target guideline and use the converted treebank as extra annotated data for the training of the target model.

[*]Equal Contribution.
[†]Email Corresponding.
[1]Our code and dataset are available at https://github.com/WangYuxuan93/G2GConversion.

Proceedings of the 29th International Conference on Computational Linguistics, pages 5450–5460
October 12–17, 2022.

Previous work for treebank conversion mainly focused on rule-based and feature-based methods. For the rule-based methods (Frank, 2001; Schuster and Manning, 2016), treebank-specific rules are designed by experts to convert one treebank to the other one. For the feature-based methods, they first construct parallel annotated data by manually annotating part of the target treebank under the source guideline (Jiang et al., 2015, 2018) or training a parser on the source treebank and parsing the target treebank with it (Zhu et al., 2011; Li et al., 2013). Then they use the source annotation as extra guiding features to train an augmented target parser that parses the whole source treebank and generates the expected target annotation. Such methods are not easily applicable to new conversions, especially between graph-structured representations, since the annotation guidelines are normally vastly different from each other, and thus the rules or features should be redesigned for every new guideline.

In this paper, we propose two pretrained model-based graph-to-graph annotation conversion approaches, namely Label Switching (LS) and Graph2Graph Linear Transformation (G2GLT), which are able to deal with conversions between graph-structured datasets and require no manually designed features. Specifically, in the Label Switching approach, we first automatically construct large scale pseudo target data by switching labels in source data to target labels based on the alignment information obtained from the parallel annotated data. After that, a graph parser is first trained on the pseudo data and then further fine-tuned on the small set of gold target annotation. The parser is eventually used to parse the source dataset to generate the target annotation. While the G2GLT approach directly inherits most of parameters from the parser trained on the source annotation, then linearly transform the biaffine attention matrix in a biaffine graph parser (Dozat and Manning, 2018) to adapt to the target annotation guideline.

We manually construct a graph-structured dataset under the refined semantic dependency graph (SDG) guideline (Che et al., 2016) on part of the text from the English Web Treebank (EWT) (Silveira et al., 2014) in the Universal Dependencies (UD) Treebanks (v2.5) (Zeman et al., 2019).[2] To verify the effectiveness of the proposed approaches, we evaluate them on conversions between 6 datasets including SDG, UD-EWT, the Enhanced Universal Dependencies (UD-Enhanced) dataset (Schuster and Manning, 2016) and three types of annotations (i.e., DM, PAS and PSD) in the SemEval 2015 task 18 dataset (Oepen et al., 2015). We further validate the quality of the converted annotations by utilizing them in the training of the target parser. Experimental results show that our approaches outperform strong baselines on both *conversion score* and *parsing score*.

In this paper, we focus on *graph-structured annotation conversion based on an existing source-annotated dataset and a small set of parallel annotated data*. Our contributions are summarized as follows.

- We propose two graph-to-graph conversion approaches that require no manually designed features.
- We verify the effectiveness of our proposed approaches on 5 existing datasets and a graph-structured dataset manually constructed by ourselves.
- We validate the quality of the annotations converted by our approaches by utilizing them to train the target parser.

2 Background

2.1 Semantic Dependency Graph

Chinese semantic dependency graph (SDG) (Che et al., 2016) is a framework for representing the meaning of different semantic units within a sentence (e.g., event chains, events, arguments, and concepts). It is in the form of directed acyclic graphs and focuses on investigating deeper semantic relations within sentences rather than morpho-syntactic patterns compared with traditional syntactic dependency trees. With the benefits of the graph's reentrancies and the easy-to-understand semantic labels, the tokens are connected more closely, making it easier to directly answer questions like *who did what to whom when and where*.

This framework is designed for Chinese exclusively. To take advantages of its properties, we modified the original annotation guidelines to make them applicable to English. We manually annotated 1,000 English sentences from UD-EWT to build a parallel annotated dataset to evaluate our annotation conversion approaches. Please refer to Appendix A.2 for the modifications we made to the Chinese SDG guidelines.

[2]Referred to as UD-EWT in the rest of the paper.

2.2 Biaffine Graph Parser

In this paper, we build all the approaches over the state-of-the-art biaffine graph parser (Dozat and Manning, 2018), which is a graph-based dependency parser that employs biaffine classifiers to predict arcs and labels in a graph. Firstly, it encodes the input sentence with a multi-layer bidirectional LSTM. Conventionally, the static word embeddings are used as the input vector. To exploit the capability of pretrained models in capturing structural information, we instead employ RoBERTa (Liu et al., 2019) to obtain the contextual representation as input. Secondly, the output of the LSTM of the i-th word, denoted as $\boldsymbol{h}_i$, is fed to four single-layer feed-forward networks (FFN) to get head and dependent representations for arcs (Eq. 1) and labels (Eq. 2).

$$\begin{aligned} \boldsymbol{h}_i^{(\text{arc-head})} &= \text{FFN}^{(\text{arc-head})}(\boldsymbol{h}_i) \\ \boldsymbol{h}_i^{(\text{arc-dep})} &= \text{FFN}^{(\text{arc-dep})}(\boldsymbol{h}_i) \end{aligned} \quad (1)$$

$$\begin{aligned} \boldsymbol{h}_i^{(\text{rel-head})} &= \text{FFN}^{(\text{rel-head})}(\boldsymbol{h}_i) \\ \boldsymbol{h}_i^{(\text{rel-dep})} &= \text{FFN}^{(\text{rel-dep})}(\boldsymbol{h}_i) \end{aligned} \quad (2)$$

Eventually, the scores for arcs (Eq. 4) and labels (Eq. 5) are computed with biaffine classifiers:

$$\text{Biaf}(\boldsymbol{x}_i, \boldsymbol{x}_j) = \boldsymbol{x}_i^\top \mathbf{U} \boldsymbol{x}_j + \boldsymbol{W}([\boldsymbol{x}_i; \boldsymbol{x}_j]) + \boldsymbol{b} \quad (3)$$

$$s_{i,j}^{(\text{arc})} = \text{Biaf}^{(\text{arc})}(\boldsymbol{h}_i^{(\text{arc-head})}, \boldsymbol{h}_j^{(\text{arc-dep})}) \quad (4)$$

$$\boldsymbol{s}_{i,j}^{(\text{rel})} = \text{Biaf}^{(\text{rel})}(\boldsymbol{h}_i^{(\text{rel-head})}, \boldsymbol{h}_j^{(\text{rel-dep})}) \quad (5)$$

Where $[\boldsymbol{x}_i; \boldsymbol{x}_j]$ indicates the concatenation of the two vectors. For the labeled parser, $\mathbf{U} \in \mathbb{R}^{d \times c \times d}$ and $\boldsymbol{W} \in \mathbb{R}^{c \times 2d}$ where c is the number of relation labels and d is the dimension of hidden states. While for the unlabeled parser, $\mathbf{U} \in \mathbb{R}^{d \times 1 \times d}$ and $\boldsymbol{W} \in \mathbb{R}^{1 \times 2d}$, so that $s_{i,j}^{(\text{arc})}$ is a scalar. The predictions of arcs and labels are $y_{i,j}'^{(\text{arc})} = \{s_{i,j}^{(\text{arc})} \geq 0\}$ and $y_{i,j}'^{(\text{rel})} = \arg\max \boldsymbol{s}_{i,j}^{(\text{rel})}$ respectively. Where the latter means that the label with the highest score is the prediction.

3 Method

In this section, we first give a formal definition of the task of supervised graph-to-graph annotation conversion (Section 3.1). Then, we present the proposed approaches, namely Label Switching (Section 3.2) and Graph2Graph Linear Transformation (Section 3.3) for this task.

3.1 Problem Definition

Given a set of texts $\mathcal{T}$, a graph-structured dataset annotated following guideline s on it is denoted by $D_s(\mathcal{T})$. In this paper, s is called the source guideline and $D_s(\mathcal{T})$ the source dataset. Assume we have a target guideline t as well as a small set of texts $\mathcal{T}' \subseteq \mathcal{T}$ annotated under t. In other words, we have the annotations of $\mathcal{T}'$ following both s (i.e., $D_s(\mathcal{T}')$) and t (i.e., $D_t(\mathcal{T}')$), which consist the parallel annotated data. The goal of supervised graph-to-graph annotation conversion is to learn a transformation $f : D_s(\mathcal{T}) \rightarrow D_t(\mathcal{T})$ based on $D_s(\mathcal{T}')$ and $D_t(\mathcal{T}')$, which converts the whole source dataset $D_s(\mathcal{T})$ into annotation under the target guideline, and thus obtain the annotated target dataset $D_t(\mathcal{T})$.

3.2 Label Switching

The lack of training data under the target guideline is a great challenge in supervised annotation conversion, especially for models based on deep neural networks. Data augmentation has been commonly used in the NLP community to alleviate the problem. Recently, Qin et al. (2020) proposed a code-switching data augmentation method, which generates pseudo multilingual corpus for the training of the multilingual BERT by randomly replacing words in a monolingual corpus based on bilingual dictionaries.

Inspired by this work, we propose the Label Switching approach that constructs pseudo target annotations to help the training of the conversion model by switching labels in source annotations to labels in the target guideline based on the alignment information obtained from the parallel annotated data. Our Label Switching approach consists of two steps: (i) label-switching data augmentation and (ii) two-step fine-tuning, which are introduced as follows.

Label-Switching Data Augmentation: To construct pseudo training data under the target guideline, we first compute the label alignment-based switching probabilities on the parallel annotations $D_s(\mathcal{T}')$ and $D_t(\mathcal{T}')$. Specifically, for a text $X \in T'$, its source and target annotations are denoted by $D_s(X)$ and $D_t(X)$ respectively. Let (i, j, r) denote the arc from word i to word j with label r, we count the number of the quadruples (r_t, p_h, p_d, r_s) for all the arcs that exist in both source and target annotations (i.e., $(i, j, r_s) \in D_s(X)$ and $(i, j, r_t) \in D_t(X)$), where p_h and p_d are the Part-

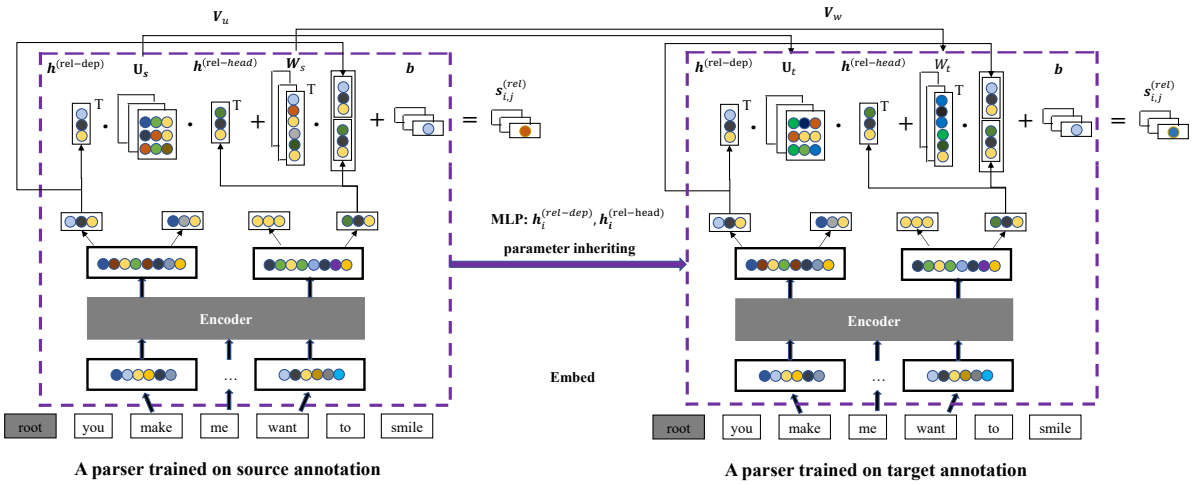

Figure 2: Schematic diagram of Graph2Graph Linear Transformation which scores each relation label between the head and the dependent. **Embed** refers to embedding layer. The parameters of a biaffine parser trained on source data is inherited by another parser with a linear transformation function applied to its biaffine attention matrix. All the inherited parameters and the linear transformation function are fine-tuned on target data except for the biaffine attention matrix inherited from the source parser.

of-Speech (POS) tags for the head and dependent words respectively.[3] The switching probability is thus computed as:

$$P(r_t|p_h, p_d, r_s) = \frac{\mathrm{N}_{(r_t,p_h,p_d,r_s)}}{\sum_{r' \in \mathcal{R}_t} \mathrm{N}_{(r',p_h,p_d,r_s)}}, \quad (6)$$

where $\mathrm{N}_{(r_t,p_h,p_d,r_s)}$ is the number of the quadruples in the parallel annotated data, and $\mathcal{R}_t$ is the set of all the labels in the target guideline. Eventually, each label r_s in the source dataset, with POS tags p_h and p_d for the head and dependent respectively, is switched to r_t in the target guideline with the probability $P(r_t|p_h, p_d, r_s)$.[4]

Two-Step Fine-Tuning: The pseudo target data generated in the last step is firstly used to train a biaffine graph parser as described in Section 2.2. Secondly, the model is further fine-tuned on the manually annotated target data $D_t(\mathcal{T}')$. Eventually, this parser is used to generate the annotation of the whole dataset under the target guideline with only texts as input.

3.3 Graph2Graph Linear Transformation

Compared with the Label Switching approach which converts the annotation through data augmentation and two-step fine-tuning, our second approach directly learns a linear function that transforms the parser trained on the source-annotated data to a parser that fits the target annotation guideline. Since the biaffine attention matrix is the core of the biaffine parser and contains knowledge that is significant for the prediction of the dependency graph. A natural way to exploit source graph information is to inherit such knowledge from a parser trained on large-scale source-annotated data.

As illustrated in Figure 2, to exploit the information in the source data, we firstly train a source parser on it. Then a linear transformation is applied to the biaffine attention matrix for relation prediction so that the relational information learned on the source annotation can be transformed to the target annotation. To maintain the source relational information, the corresponding biaffine attention matrix is fixed during the fine-tuning stage on the target data.

Specifically, let $\mathbf{U}_s, \boldsymbol{W}_s$ and $\boldsymbol{b}_s$ be the parameters of a biaffine parser trained on large-scale source-annotated data with Eq. 3. Let c_s and c_t be the number of relation labels in the source and target annotations respectively. Two linear transformation functions $\boldsymbol{V}_u \in \mathbb{R}^{c_t \times c_s}$ and $\boldsymbol{V}_w \in \mathbb{R}^{c_t \times c_s}$ are applied to $\mathbf{U}_s \in \mathbb{R}^{d \times c_s \times d}$ and $\boldsymbol{W}_s \in \mathbb{R}^{c_s \times 2d}$ respectively to obtain the parameters $\mathbf{U}_t \in \mathbb{R}^{d \times c_t \times d}$ and $\boldsymbol{W}_t \in \mathbb{R}^{c_t \times 2d}$ for the target parser.

$$\mathbf{U}_t = (\boldsymbol{V}_u \mathbf{U}_s^{\top(1,2)})^{\top(1,2)} \quad (7)$$

[3] We use gold POS tags from the source dataset in our experiments.

[4] Due to the limited number of parallel annotated data, the switching probabilities can not cover all the labels in the source data. For those not covered, we leave them as they are.

$$\boldsymbol{W}_t = \boldsymbol{V}_w \boldsymbol{W}_s \tag{8}$$

$$\text{Biaf}_t(\boldsymbol{x}_i, \boldsymbol{x}_j) = \boldsymbol{x}_i^\top \mathbf{U}_t \boldsymbol{x}_j + \boldsymbol{W}_t([\boldsymbol{x}_i; \boldsymbol{x}_j]) + \boldsymbol{b}_s \tag{9}$$

where $\mathbf{U}^{\top(1,2)}$ means transposing the first and the second dimensions of tensor $\mathbf{U}$.

Briefly, our Graph2Graph Linear Transformation approach consists of two steps: (i) training a source biaffine parser on the source-annotated data; (ii) applying the linear transformation function on the source parser and fine-tuning it on the target-annotated data while freezing the parameters in the biaffine attention matrix inherited from the source parser.

4 Experimental Setup

4.1 Datasets and Experimental Settings

Dataset	#Sent	#Token	#Arc (avg)	#Label
UD-EWT	16,622	254,829	1.00	49
SDG	1,000	15,991	1.07	61
UD-En	16,622	254,829	1.05	398
DM	37,066	834,665	0.78	60
PAS	37,066	834,665	1.02	43
PSD	37,066	834,665	0.70	92

Table 1: Data statistics. **#Sent** and **#Token** denote the number of sentences and tokens respectively for all the annotated data in the dataset (including training, valid and test sets). **#Arc (avg)** denotes the average number of arcs per token, while **#Label** the number of label types. **UD-En** denotes the UD-Enhanced dataset. For **DM/PAS/PSD**, the out-of-domain test sets are excluded.

Dataset	Source-Train	Train	Valid	Test
UD2UD-En	10,508	1000	500	5,000
UD2SDG	16,369	800	-	200
D2D*	26,206	1,000	500	10,000

Table 2: Data split statistics. **UD** denotes UD-EWT. **D2D*** denotes the 6 conversion tasks between **DM**, **PAS** and **PSD**.

For the evaluation of the proposed approaches, we manually construct the SDG dataset (on part of texts from UD-EWT) and employ two groups of existing parallel annotated datasets, namely {UD-EWT, UD-Enhanced} and {DM, PAS, PSD}, whose statistics are shown in Table 1.

UD-EWT is a tree-structured syntactic dataset under the Universal Dependencies (UD) guideline. UD (Zeman et al., 2019) is a framework for consistent annotation of grammar across languages. The UD Treebanks (v2.5) consist of 157 treebanks in 90 languages,[5] which could be a good source to obtain source datasets for dataset construction under a new guideline. Therefore, we use UD-EWT as the source dataset in our experiments.

UD-Enhanced is a graph-structured syntactic dataset converted from UD-EWT by adding relations and augmenting relation names (Schuster and Manning, 2016).[6]

SDG is a graph-structured semantic dataset with 1,000 sentences annotated under the refined semantic dependency graph guideline (Che et al., 2016).

DM, PAS and PSD are three types of graph-structured semantic annotations in the SemEval 2015 task 18 dataset (Oepen et al., 2015).[7]

The approaches are evaluated on eight annotation conversion tasks including UD-EWT to UD-Enhanced (**UD2UD-En**), UD-EWT to SDG (**UD2SDG**) as well as six conversion tasks between DM, PAS and PSD (**PAS2DM**, **PSD2DM**, **PAS2PSD**, **DM2PSD**, **DM2PAS**, **PSD2PAS**). Recall that the goal of this paper is graph-structured annotation conversion based on an existing source-annotated dataset and a small set of parallel annotated data. To fit the goal, we re-split the datasets so that only a limited number of parallel annotated examples are available while training. The data split statistics are shown in Table 2, where **Train/Valid/Test** are parallel annotated data and **Source-Train** contains only the source-side annotation that will be used in the experiment of utilizing converted data in Section 5.2. We perform 5-fold cross-validation on the 1,000 parallel annotated sentences in the conversion task from UD-EWT to SDG.

For all the approaches, we employ the biaffine graph parser as described in Section 2.2 to predict the target graph, and use RoBERTa (Liu et al., 2019) to obtain contextual representations as its input. We set the learning rate of RoBERTa to 2e-5 and that of other parameters to 2e-2. Other hyper-parameters are adopted from the paper of Dozat and Manning (2018).

Existing graph banks can be broadly categorized into two types, namely the *bilexical graphs* (where dependencies are directly linked between surface lexical units) and the *conceptual graphs* (where

[5] http://hdl.handle.net/11234/1-3105

[6] https://github.com/UniversalDependencies/UD_English-EWT

[7] https://catalog.ldc.upenn.edu/LDC2016T10

Methods	UD2UD-En		UD2SDG		PAS2DM		PSD2DM		DM2PAS		PSD2PAS		DM2PSD		PAS2PSD		AVG.	
	UF	LF	UF	LF	UF	LF	UF	LF	UF	LF	UF	LF	UF	LF	UF	LF	UF	LF
DFT	89.92	82.04	86.47	74.07	90.07	87.53	90.07	87.53	93.76	91.56	93.76	91.56	90.87	75.32	90.87	75.32	90.72	83.12
TSFT	91.74	83.32	87.49	74.87	90.51	88.01	90.48	88.10	94.11	91.96	94.46	92.37	91.26	75.32	91.66	75.57	91.46	83.69
PE	**99.41**	**95.82**	87.68	74.93	89.79	87.33	90.11	87.68	93.93	91.75	94.20	91.90	89.69	74.03	91.81	76.54	92.08	85.00
G2GTr	96.63	89.16	87.85	74.91	90.11	87.62	90.58	88.20	93.81	91.63	94.42	92.28	90.53	74.21	91.88	75.73	91.98	84.22
LS	97.13	89.27	89.15	76.65	91.12	88.71	91.05	88.71	94.72	92.90	95.23	93.29	92.08	76.90	93.14	77.65	92.95	85.51
G2GLT	96.51	90.97	89.30	77.05	90.86	88.46	91.27	89.31	94.37	92.31	94.88	92.92	90.81	75.12	92.63	76.81	92.58	85.37
LS+G2GLT	97.27	91.47	**89.81**	**77.81**	**91.33**	**89.05**	**91.87**	**90.05**	**94.86**	**93.01**	**95.80**	**94.10**	**92.77**	**79.36**	**93.29**	**78.42**	**93.38**	**86.66**

Table 3: Conversion scores on test data.

dependencies are between virtual nodes that do not need be explicitly mapped to surface linguistic forms). DM, PAS and PSD fall into the former category, while another popular graph bank Abstract Meaning Representation (AMR) (Banarescu et al., 2013) belongs to the latter. We choose to conduct experiments on DM, PAS and PSD because they are annotated on the same corpus, which guarantees the availability of parallel annotated data required by our approaches. While currently we have no access to parallel annotated data for AMR and any other graph bank. More importantly, since our Label Switching approach exploits the overlapped dependencies in source and target annotation for label switching, it is not straightforwardly applicable to conversion between conceptual graphs since virtual nodes make it hard to identify overlapped dependencies. Therefore, we leave as future work to extend our approaches to conversion between conceptual graphs.

4.2 Baselines and Evaluation Metrics

Our approaches, namely **Label Switching (LS)** and **Graph2Graph Linear Transformation (G2GLT)**, are compared with four baselines introduced as follows.

Direct Fine-Tuning (DFT) A RoBERTa-based biaffine graph parser is directly trained on the small set of target annotations $D_t(\mathcal{T}')$.

Two-Step Fine-Tuning (TSFT) A RoBERTa-based biaffine graph parser is firstly trained on the whole source dataset $D_s(\mathcal{T})$, and then further fine-tuned on the small set of target annotations $D_t(\mathcal{T}')$.[8] It is only trained for 5 epochs in the first step to avoid over-fitting to the source data.

Pattern Embedding (PE) This is a feature-based method closely following the work of Jiang et al. (2018) which takes advantage of source guiding features. To adopt it in graph-to-graph annotation conversion, we average the label embeddings for structural information representation in the reentrancy structure and add reverse sibling pattern for the reentrancy structure in graphs.[9] A RoBERTa-based biaffine graph parser is employed in the method.

Graph2Graph Transformer (G2GTr) Graph2Graph Transformer is proposed by Mohammadshahi and Henderson (2021) for dependency parsing with iterative refinement, which encodes dependency trees produced by the last step to obtain structural enhanced representation that is utilized to predict refined trees. In this method, we employ Graph2Graph Transformer to obtain each token's representation infused with source annotation information and feed them to a RoBERTa-based biaffine graph parser that predicts the target annotation.

Moreover, it is straightforward to combine the two approaches we proposed **(LS+G2GLT)** by averaging the scores they predicted for arcs (Eq. 4) and labels (Eq. 5) respectively. This is also evaluated in the experiments.

All the results are reported in terms of unlabelled F1 score (UF) and labelled F1 score (LF) on the target test set.

5 Results

5.1 Conversion Results

Table 3 shows the results of the eight conversion tasks, which are the average over three runs.[10] With the help of the pretrained model, DFT achieves fair results with limited data annotated under the target guideline used for training. While TSFT improves the results by training on the large-scale source-annotated dataset firstly to capture the structural information implicitly. The other two baselines, which adopt previous methods for graph-to-graph annotation conversion, yield better results. Specif-

[8] In the second step, non-RoBERTa parameters are reinitialized since the source and target guidelines have different label sets, and thus only the RoBERTa parameters can be shared.

[9] Reentrancy structure represent the structure of a word with multiple heads, which only occurs in graphs but not in trees.

[10] Please refer to Appendix A.3 for the standard deviation.

	UD2UD-En		UD2SDG		PAS2DM		PSD2DM		DM2PAS		PSD2PAS		DM2PSD		PAS2PSD		AVG.	
Methods	UF	LF	UF	LF	UF	LF	UF	LF	UF	LF	UF	LF	UF	LF	UF	LF	UF	LF
Single	89.92	82.04	86.47	74.07	90.07	87.53	90.07	87.53	93.76	91.56	93.76	91.56	90.87	75.32	90.87	75.32	90.72	83.12
DFT	90.56	82.86	87.22	75.11	90.51	88.09	90.51	88.09	94.14	92.01	94.14	92.01	91.57	76.49	91.57	76.49	91.28	83.89
TSFT	91.63	83.60	87.98	75.74	91.02	88.66	90.78	88.49	94.51	92.52	94.88	92.83	91.79	76.44	91.89	76.57	91.81	84.36
PE	93.39	**88.41**	88.48	76.13	90.97	88.77	91.04	89.07	94.63	92.64	95.08	93.16	91.62	77.09	92.24	77.63	92.18	85.36
G2GTr	92.56	84.89	87.57	75.45	90.61	88.40	90.87	88.73	94.23	92.18	94.77	92.82	91.94	77.33	91.90	76.35	91.81	84.52
LS	93.52	85.77	88.86	76.82	91.27	88.86	91.19	88.96	94.92	93.04	95.17	93.32	92.51	77.92	92.90	78.02	92.54	85.34
G2GLT	93.23	87.36	88.92	76.92	91.28	88.92	91.63	89.81	94.85	92.90	95.40	93.56	92.42	78.34	92.69	77.58	92.55	85.67
LS+G2GLT	**93.64**	87.47	**89.32**	**77.48**	**91.50**	**89.20**	**91.70**	**89.87**	**95.02**	**93.20**	**95.67**	**93.96**	**93.07**	**79.73**	**92.91**	**78.41**	**92.85**	**86.16**

Table 4: Parsing scores on test data.

ically, G2GTr improves the performance by employing the Graph2Graph Transformer to encode source-annotated information. As the only feature-based method, PE achieves the best average results in all the baselines.

As for our approaches, they achieve comparable conversion scores, and both of them outperform the baselines on average. Besides, they significantly outperform all the baselines in almost all the conversion tasks, except in the conversion from UD-EWT to UD-Enhanced where PE yields the best result. We assume that this is because UD-Enhanced is converted from UD-EWT by adding relations and augmenting relation names to make implicit relations between content words more explicit. Therefore, UD-Enhanced shares some labels with UD-EWT and they have the highest annotation overlap among all the conversion tasks.[11] While in all the other conversion tasks, the two datasets have completely different label sets, and thus their overlap rate is much lower. Obviously, the feature-based PE approach performs extremely good in the case where the annotation overlap rate is high. However, it is outperformed by our approaches in all the other more complicated conversion tasks where the annotation overlap rate is lower.

Furthermore, since our proposed approaches improve the conversion score in two facets, namely data augmentation and parameter transformation, we assume that the improvements they brought are orthogonal to each other. Therefore, we combine them by simply averaging the arc and label scores they predicted and find that the combined model further significantly improves the performance on all the conversion tasks.

5.2 Utilizing Converted Data

In order to evaluate the quality of the converted data, we utilize them to train a target parser and measure the quality with the parsing score. Specifically, following the data split in Table 2, we first convert **Source-Train** into target-annotated data with different conversion approaches, then train target parsers with the converted data. Eventually, the target parsers are evaluated on the **Test** set.

Table 4 shows the empirical results. Besides the methods in Table 3, we also include a **Single** baseline without the annotation conversion process, which is a target parser trained only on the target annotation in the **Train** set in Table 2. Obviously, utilizing the target data converted from the large-scale source data during training can significantly improve the performance. Moreover, both of our approaches outperform all the baselines in almost all parsing tasks. The result of PE is much higher than that of our approaches for the UD-EWT to UD-Enhanced task. We assume that this is due to the high conversion score of PE on the conversion task from UD-EWT to UD-Enhanced, whose reason has been discussed in Section 5.1. Similar to the case in conversion tasks, the combined model can further improve the parsing performance.

5.3 Effect of Parallel Annotated Data Size

Recall that this paper aims at graph-structured annotation conversion based on an existing source-annotated dataset and little human labour. Therefore, the parallel annotated data size is of great importance since the smaller it is, the less human labour will be required. This section investigates the effect of the parallel annotated data size on the proposed conversion approaches. Specifically, we evaluate the approaches on 200/500/1,000/2,000 randomly selected training sets respectively with the same valid/test sets introduced in Section 4.1.[12]

Figure 3 shows the results with different parallel annotated data sizes, where it is obvious that the performances of all methods increase as the data size increases in almost all the conversion tasks.

[11] Please refer to Appendix A.1 for the annotation overlap information in detail.

[12] We conduct experiments on all conversion tasks except the conversion from UD-EWT to SDG since there are only 1,000 parallel annotated sentences. The results for conversion from PAS to PSD is shown in Appendix A.4.

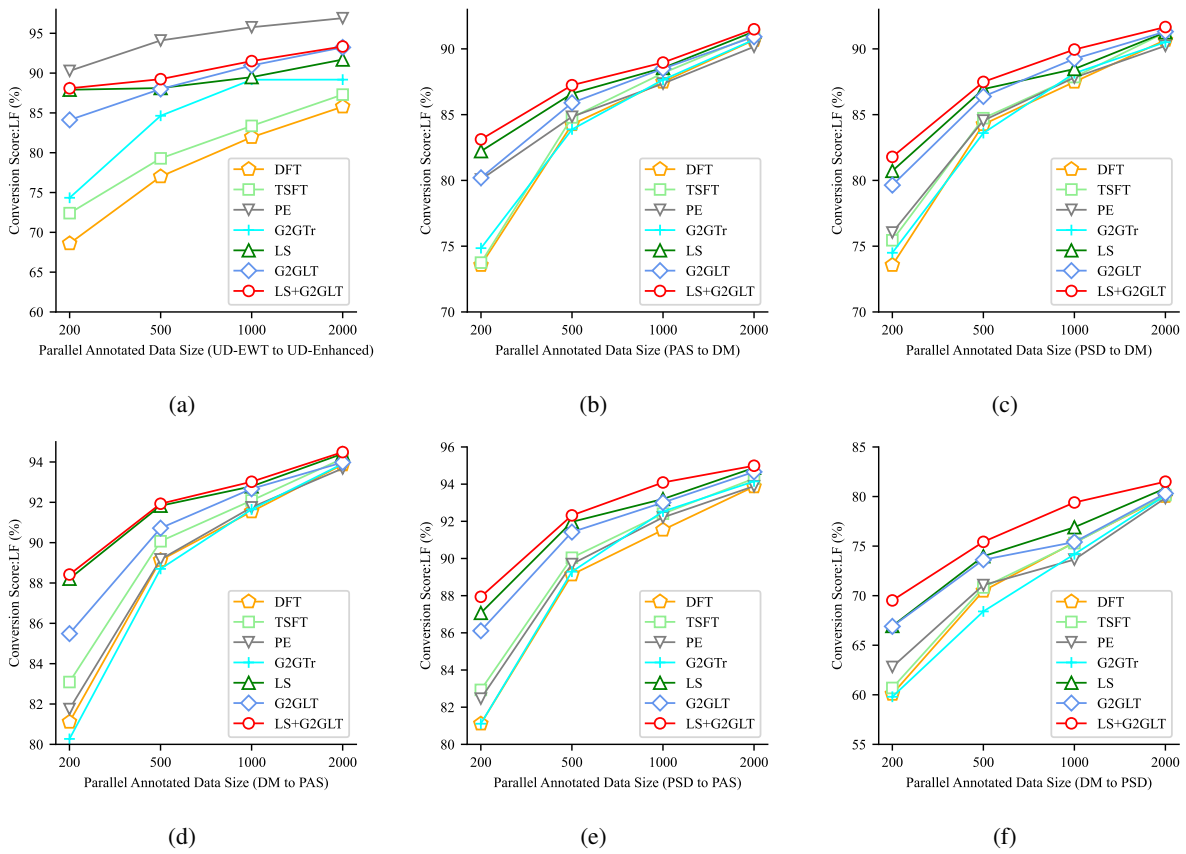

Figure 3: Results for conversions with different parallel annotated data sizes (best viewed in color).

However, the performance of LS, G2GLT and the combined approach is not apparently influenced by the change of data size in the annotation conversion from UD-EWT to UD-Enhanced. It can be explained by the high annotation overlap rate between them. With the highly overlapped annotation, our proposed approaches can easily obtain promising results with only 200 parallel annotated sentences. While since the PE method extracts features from the source graph to predict the target graph, it also benefits from the high annotation overlap rate. Another finding from Figure 3 is that the difference of LF between our proposed approaches and the baselines shrinks as the data size increases, which may indicates that our proposed approaches are most suitable for cases where only limited parallel annotated data is available. And this exactly satisfies the aim of this paper.

6 Conclusion

This paper aims at graph-structured annotation conversion based on an existing source-annotated dataset with little human labour. We propose two graph-to-graph annotation conversion approaches, namely Label Switching and Graph2Graph Linear Transformation, and show their effectiveness on eight annotation conversion tasks and converted data utilizing tasks. Results show that 1) the two approaches achieve comparable conversion scores; 2) our proposed approaches are most suitable for cases where only limited parallel annotated data is available; 3) the two approaches can be combined to further improve the performance.

7 Ethical Considerations

The sentences in the Semantic Dependency Graph (SDG) dataset we construct are collected from the English Web Treebank (EWT) in the Universal Dependencies (UD) Treebanks (v2.5) (Zeman et al., 2019) which is a publicly available dataset. The detailed statistics of the SDG dataset are shown in Table 1. All the annotators are voluntary participants who have given informed consent and been fairly compensated during the annotation process.

Acknowledgments

This work was supported by the National Key R&D Program of China via grant 2020AAA0106501 and

the National Natural Science Foundation of China (NSFC) via grant 61976072 and 62176078.

References

Laura Banarescu, Claire Bonial, Shu Cai, Madalina Georgescu, Kira Griffitt, Ulf Hermjakob, Kevin Knight, Philipp Koehn, Martha Palmer, and Nathan Schneider. 2013. Abstract Meaning Representation for sembanking. In *Proceedings of the 7th Linguistic Annotation Workshop and Interoperability with Discourse*, pages 178–186, Sofia, Bulgaria. Association for Computational Linguistics.

Wanxiang Che, Yanqiu Shao, Ting Liu, and Yu Ding. 2016. SemEval-2016 task 9: Chinese semantic dependency parsing. In *Proc. of SemEval*.

Timothy Dozat and Christopher D. Manning. 2018. Simpler but more accurate semantic dependency parsing. In *Proc. of ACL*.

Anette Frank. 2001. Treebank conversion - converting the negra treebank to an ltag grammar. In *Proceedings of the Workshop on Multi-layer Corpus-based Analysis*.

Wenbin Jiang, Yajuan Lü, Liang Huang, and Qun Liu. 2015. Automatic adaptation of annotations. *Computational Linguistics*.

Xinzhou Jiang, Zhenghua Li, Bo Zhang, Min Zhang, Sheng Li, and Luo Si. 2018. Supervised treebank conversion: Data and approaches. In *Proc. of ACL*.

Xiang Li, Wenbin Jiang, Yajuan Lü, and Qun Liu. 2013. Iterative transformation of annotation guidelines for constituency parsing. In *Proc. of ACL*.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Roberta: A robustly optimized BERT pretraining approach. *CoRR*, abs/1907.11692.

Alireza Mohammadshahi and James Henderson. 2021. Recursive Non-Autoregressive Graph-to-Graph Transformer for Dependency Parsing with Iterative Refinement. *Transactions of the Association for Computational Linguistics*.

Stephan Oepen, Omri Abend, Lasha Abzianidze, Johan Bos, Jan Hajic, Daniel Hershcovich, Bin Li, Tim O'Gorman, Nianwen Xue, and Daniel Zeman. 2020. MRP 2020: The second shared task on cross-framework and cross-lingual meaning representation parsing. In *Proceedings of the CoNLL 2020 Shared Task: Cross-Framework Meaning Representation Parsing*.

Stephan Oepen, Omri Abend, Jan Hajic, Daniel Hershcovich, Marco Kuhlmann, Tim O'Gorman, Nianwen Xue, Jayeol Chun, Milan Straka, and Zdenka Uresova. 2019. MRP 2019: Cross-framework meaning representation parsing. In *Proceedings of the Shared Task on Cross-Framework Meaning Representation Parsing at the 2019 Conference on Natural Language Learning*.

Stephan Oepen, Marco Kuhlmann, Yusuke Miyao, Daniel Zeman, Silvie Cinková, Dan Flickinger, Jan Hajič, and Zdeňka Urešová. 2015. SemEval 2015 task 18: Broad-coverage semantic dependency parsing. In *Proc. of SemEval*.

Libo Qin, Minheng Ni, Yue Zhang, and Wanxiang Che. 2020. Cosda-ml: Multi-lingual code-switching data augmentation for zero-shot cross-lingual nlp. In *Proc. of IJCAI*.

Sebastian Schuster and Christopher D. Manning. 2016. Enhanced English Universal Dependencies: An improved representation for natural language understanding tasks. In *Proc. of LREC*.

Djamé Seddah, Eric de la Clergerie, Benoît Sagot, Héctor Martínez Alonso, and Marie Candito. 2018. Cheating a Parser to Death: Data-driven Cross-Treebank Annotation Transfer. In *Proc. of LREC*.

Natalia Silveira, Timothy Dozat, Marie-Catherine de Marneffe, Samuel Bowman, Miriam Connor, John Bauer, and Chris Manning. 2014. A gold standard dependency corpus for English. In *Proc. of LREC*.

Daniel Zeman, Joakim Nivre, Mitchell Abrams, and et al. 2019. Universal dependencies 2.5. LINDAT/CLARIAH-CZ digital library at the Institute of Formal and Applied Linguistics (ÚFAL), Faculty of Mathematics and Physics, Charles University.

Muhua Zhu, Jingbo Zhu, and Minghan Hu. 2011. Better automatic treebank conversion using a feature-based approach. In *Proc. of ACL*.

A Appendix

A.1 Annotation Overlap

In this section, we evaluate the annotation overlap between the datasets in our conversion tasks. Specifically, for each of the eight conversion tasks used in this paper, we directly evaluate the original source dataset on the gold target dataset and use the scores to measure the annotation overlap between the two datasets. A higher score between two datasets represents a higher annotation overlap rate between them.

Source	Target	UF	LF
UD-EWT	**UD-Enhanced**	96.95	83.87
UD-EWT	**SDG**	87.98	-
PAS	**DM**	63.78	-
PSD	**DM**	27.21	-
DM	**PAS**	63.78	-
PSD	**PAS**	30.24	-
DM	**PSD**	27.21	-
PAS	**PSD**	30.24	-

Table 5: Annotation overlap in terms of UF and LF.

Results are shown in Table 5, we only report the LF for the dataset pair of {UD-EWT, UD-Enhanced}. This is because UD-Enhanced is converted from UD-EWT by adding relations and augmenting relation names to make implicit relations between content words more explicit. Therefore, UD-Enhanced shares some labels with UD-EWT. While in all the other pairs, the two datasets have completely different label sets. Thus we can not compute the LF for them.

As for the UF which reflects the structural annotation overlap between datasets, we find that UD-EWT is most similar to UD-Enhanced, which can also be explained by the construction of UD-Enhanced introduced above. The overlap between UD-EWT and SDG is lower, indicating that the conversion from UD-EWT to SDG is harder than that from UD-EWT to UD-Enhanced. Moreover, the overlap rate between DM, PAS and PSD are much lower, with only 27.21% UF for PAS and PSD, which suggests that the shared information between them is much less than that for the other pairs and conversion between them are even more challenging.

A.2 Semantic Dependency Graph Annotation Guidelines

We modified Chinese Semantic Dependency Graph guidelines[13] to make it applicable to English in two ways: adding more semantic edges and reducing more semantic labels.

We added more edges between predicates and arguments. In the Chinese Semantic Dependency Graph, it only considers omitted object and subject which has been referred in previous clauses. We also take omitted predicates into account, thus, ensuring the semantic integrity of semantic units. An example is shown in Figure 4. Here, the predicate "cried" has been omitted and we added an extra edge to connect "I" with "cried" which makes the second clause more explicit.

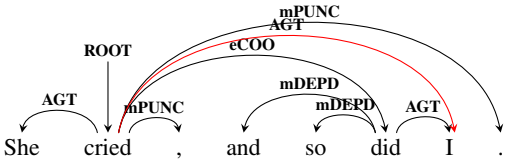

Figure 4: Example of annotation for omitted predicates

Semantic Class	Labels
Semantic roles	AGT(Agent), EXP(Experiencer), PAT(Patient), CONT(Content), PROD(Product), BELGONG(Belongings), PART, MATL(Material), TOOL, REAS(Reason), LOC(location), TIME, SCO(Scope), FEAT, QUAN(Quantity), STAT(State)
Reverse relations	r+semantic roles
Nested relations	d+semantic roles
Event relations	eCOO(Coordination), eRECT(Recount), eSELT(Select), ePROG(Progression), eSUCC(Successor), eRESU(Result), eCOND(Condition), eSUPP(Supposition), eEFTT(Effect), eEQU(Equal), eADVT(adversative)
Semantic markers	mNEG(Negation), mRELA(relation), mPUNC(Punctuation), mDEPD, mFIXED

Table 6: Label set of the semantic relation of EN-SDG

[13] https://csdp-doc.readthedocs.io/zh_CN/latest/

Methods	UD2UD-En		UD2SDG		PAS2DM		PSD2DM		DM2PAS		PSD2PAS		DM2PSD		PAS2PSD		AVG.	
	UF	LF	UF	LF	UF	LF	UF	LF	UF	LF	UF	LF	UF	LF	UF	LF	UF	LF
DFT	0.08	0.23	0.06	0.04	0.08	0.10	0.08	0.10	0.10	0.09	0.10	0.09	0.12	0.21	0.12	0.21	0.03	0.08
TSFT	0.13	0.06	0.12	0.17	0.10	0.13	0.15	0.21	0.11	0.16	0.10	0.06	0.1	0.30	0.18	0.32	0.06	0.11
PE	0.08	0.28	0.14	0.17	0.05	0.08	0.10	0.16	0.02	0.07	0.17	0.27	0.15	0.35	0.14	0.25	0.01	0.05
G2GTr	0.24	0.15	0.09	0.20	0.09	0.06	0.08	0.05	0.07	0.05	0.17	0.26	0.09	0.06	0.14	0.18	0.06	0.05
LS	0.35	0.39	0.04	0.05	0.09	0.17	0.16	0.21	0.06	0.10	0.10	0.09	0.03	0.18	0.07	0.05	0.07	0.05
G2GLT	0.13	0.10	0.12	0.17	0.08	0.06	0.07	0.08	0.26	0.32	0.10	0.09	0.28	0.25	0.09	0.36	0.11	0.10
LS+G2GLT	0.16	0.11	0.05	0.13	0.03	0.08	0.07	0.09	0.01	0.01	0.01	0.01	0.08	0.10	0.02	0.22	0.02	0.02

Table 7: Standard Deviation for conversion scores on test data.

As for semantic labels, we merged labels for simplification. Specifically, in semantic roles, we merged Aft into EXP, Orig and Comp into DATV, Reas, Int into REAS, Host, Nmod, Tmod into FEAT, Qp, Freq, Seq into QUAN. In event relations, we merged eInf, eCau into eRESU, eConc and eAban into eSELT, eSUM into eRECT. In semantic markers, we just kept mNEG, mRELA and mPUNC and abandoned other markers because they most designed for Chinese specifically. We also create some new labels for unique usage in English: mFIXED for multi-word expressions(mwe) and mDEPD for function words like articles. The list of the semantic labels in our SDG guideline is shown in Table 6.

A.3 Standard Deviation

Table 7 shows the standard deviation for the experiments in Table 3.

A.4 Effect of Parallel Annotated Data Size

In this section, we report the result for conversion from PAS to PSD with different parallel annotated data sizes. The results are shown in Figure 5.

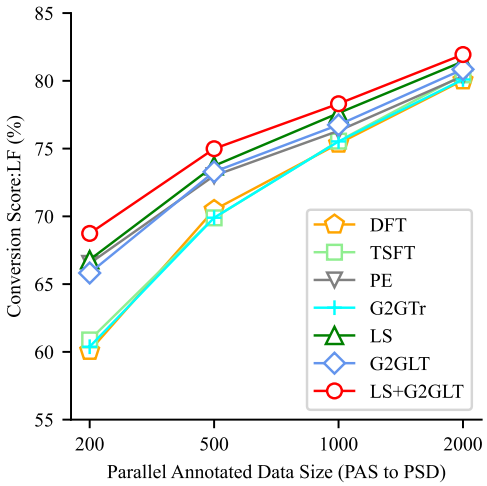

Figure 5: Results for conversions from PAS to PSD with different parallel annotated data sizes.

BiBL: AMR Parsing and Generation with Bidirectional Bayesian Learning

Ziming Cheng[1,2], **Zuchao Li**[1,2,*], **and Hai Zhao**[1,2,*]
[1]Department of Computer Science and Engineering, Shanghai Jiao Tong University
[2]MoE Key Lab of Artificial Intelligence, AI Institute, Shanghai Jiao Tong University
{kk.cheng, charlee}@sjtu.edu.cn, zhaohai@cs.sjtu.edu.cn

Abstract

Abstract Meaning Representation (AMR) offers a unified semantic representation for natural language sentences. Thus transformation between AMR and text yields two transition tasks in opposite directions, i.e., Text-to-AMR parsing and AMR-to-Text generation. Existing AMR studies only focus on one-side improvements despite the duality of the two tasks, and their improvements are greatly attributed to the inclusion of large extra training data or complex structure modifications which harm the inference speed. Instead, we propose data-efficient Bidirectional Bayesian learning (BiBL) to facilitate bidirectional information transition by adopting a single-stage multitasking strategy so that the resulting model may enjoy much lighter training at the same time. Evaluation on benchmark datasets shows that our proposed BiBL outperforms strong previous seq2seq refinements without the help of extra data which is indispensable in existing counterpart models. We release the codes of BiBL at: https://github.com/KHAKhazeus/BiBL.

1 Introduction

Abstract Meaning Representation (AMR) (Banarescu et al., 2013) is a formalism that could capture the semantic meaning of a natural language sentence in a graph representation. An AMR graph example is shown in Figure 1. Leaves in the AMR graph denote concepts in the text sentence while other nodes within the graph are entities. Two entities could be connected with relations defined under a common standard, and instance relations connect entities and corresponding concepts. The flexibility and semantic invariance characteristics make AMR applicable to various natural language processing (NLP) downstream tasks and achieve solid performance.

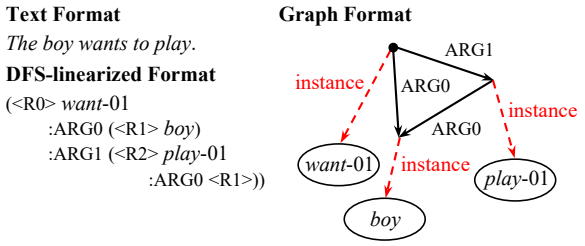

Figure 1: AMR graph and DFS-linearized formats for representing a text sentence "The boy wants to play". Instance relations are denoted with dotted red lines while other relations are denoted with black lines. Concepts in AMR graphs are enclosed by oval shapes. The indentation in the DFS-linearized format is only for clarity purposes.

The transformation between AMR graphs and texts forms two basic tasks offering helpful AMR information for the downstream tasks. Traditional approaches treat AMR-to-Text generation and Text-to-AMR parsing separately. However, utilizing graph linearization and the power of pre-trained language models, SPRING (Bevilacqua et al., 2021) casts two tasks into a unified sequence-to-sequence (seq2seq) structure and outperforms all traditional approaches proposed before. Such progress consolidates the prevalent position of seq2seq approaches tackling the AMR transition tasks.

Recently, many methods have been proposed to further enhance the seq2seq model. In order to improve the model performance on Text-to-AMR parsing, Zhou et al. (2021) incorporates an action-pointer mechanism. Yu and Gildea (2022) adds ancestor information of AMR graphs into the model. Chen et al. (2022) introduces a data augmentation approach by transforming data excerpted from semantic role labeling and dependency parsing into pseudo-AMR data, and then constructing extra training tasks using these data. Hoang et al. (2021) proposes graph ensembling to combine different models in order to achieve better results. For the AMR-to-Text generation task, Anonymous (2021)

*Corresponding author. This work was supported by Key Projects of National Natural Science Foundation of China (U1836222 and 61733011).

Proceedings of the 29th International Conference on Computational Linguistics, pages 5461–5475
October 12–17, 2022.

designs two auxiliary training objectives, i.e., relationship prediction and distance prediction of nodes in AMR graphs. Ribeiro et al. (2021) utilizes data from the KG2Text task, which is transforming knowledge graphs into texts, and masked language modeling to pre-train the model. Bai et al. (2022) proposes six graph pre-training tasks to obtain a better pre-trained model. Though the above-mentioned refinements are effective, applying auxiliary techniques, including ensembling and adding graph information during training, could greatly harm the training or inference speed. Meanwhile, obtaining a better pre-training model using extra silver data or data augmentation techniques greatly increases the time consumption and design complexity of the whole model training phase.

Instead of using extra data or complicated techniques which harm the training or inference speed, we propose a novel solution for AMR transition tasks. The solution consists of two auxiliary tasks to help the model grasp the joint probability of the AMR graphs and texts. By interweaving the two auxiliary tasks with the main transduction task, the extra knowledge learned in the two auxiliary tasks could boost the model understanding of the main task. Therefore, single-stage multitask learning without fine-tuning could be adopted for BiBL. Based on our proposed BiBL, the training paradigm could be greatly simplified, and under the same model settings, our implemented models achieve new state-of-the-art performances in AMR-to-Text generation. Compared with previous refinements, further comparison experiments show that BiBL is more efficient both during the training and inference phases.

2 Related Work

There are two tasks when concerning the transformation between AMR graphs and texts. However, it is not always that these two tasks are jointly considered.

2.1 Text-to-AMR Parsing

Graph-based approaches Many graph-based approaches (Zhang et al., 2019a,b; Cai and Lam, 2020a,b; Zhou et al., 2020) have been proposed to solve the transition from texts to AMR graphs. These approaches merge graph structures into their model design, and graph recategorization technique (Zhang et al., 2019a; Zhou et al., 2020; Cai and Lam, 2020a) is introduced to unify different graph representations that are identical in semantics. Despite the complex graph structure design, these models can achieve decent performances due to the explicit incorporation of AMR graph features.

Pure seq2seq approaches Pure seq2seq approaches for the Text-to-AMR task are data-hungry in nature. Therefore, in previous works, several methods have been explored to counter the data sparsity problem (Konstas et al., 2017; van Noord and Bos, 2017). However, these approaches either introduce a large amount of data that is not closely related to the parsing task or reduce the capacity of the model by ignoring and recovering fine-grained details through data pre-processing and post-processing. Utilizing BART (Lewis et al., 2020) as the model backbone, SPRING (Bevilacqua et al., 2021) provides a new seq2seq solution for Text-to-AMR parsing without removing details during the training process. Nonetheless, SPRING obtains solid model performance gains compared with previous methods and consequently becomes the backbones of many recent refinements.

2.2 AMR-to-Text Generation

Graph2seq approaches AMR-to-Text generation is the reverse process of Text-to-AMR parsing. Taking graph characteristics into consideration, several graph2seq methods (Beck et al., 2018; Guo et al., 2019; Wang et al., 2020a,b; Song et al., 2018; Zhu et al., 2019) utilize graph neural networks to tackle the problem. However, the performances of graph2seq approaches are inferior to the state-of-the-art approach, i.e., SPRING, which is purely seq2seq.

Pure seq2seq approaches Similar to pure seq2seq approaches in the Text-to-AMR parsing task, graph linearization is the key to transforming the task into a seq2seq formulation. Based on this formulation, Mager et al. (2020) adapts a pre-trained Transformer decoder to the generation task through fine-tuning. SPRING (Bevilacqua et al., 2021) fine-tunes BART on the generation task and proves that the encoder-decoder structure of BART greatly contributes to the final model performance. Since SPRING is powerful for both transduction directions, BiBL inherits the model backbone of SPRING, which guarantees a solid basis of model performance.

3 Method

As the seq2seq model only receives and emits data in sequential forms, we have to take the necessary pre-processing and post-processing to perform conversions between AMR graphs and their sequential forms. Then, the seq2seq transduction is completed using sequence generation models of encoder-decoder structure.

On top of the seq2seq model design, we enhance bidirectional model performance through two auxiliary tasks which facilitate the model's understanding of the joint probability between texts and AMR graphs.

3.1 Graph Linearization

For both AMR-to-Text generation and Text-to-AMR parsing, graph linearization is a necessary pre-processing task in seq2seq modeling. First, we replace the variables in the AMR graphs with specially designed tokens to avoid unnecessary tokenization. These tokens are added to the model as special tokens. Then, we linearize the graphs following a depth-first search (DFS) fashion. We use parentheses to indicate visit depth and use blank spaces to separate variables and actual concepts. A DFS-linearized example is shown in Figure 1.

Since the capacity of the model backbone is fairly adequate and graph recategorization mentioned in Section 2.1 harms the model performance in out-of-distribution settings, we do not incorporate graph recategorization into the pre-processing procedures of BiBL.

3.2 Task Formulation

Following the design of graph linearization, the bidirectional transition task could be transformed into a seq2seq task formulation. Given a linearized AMR graph $Y = [y_1, ..., y_k]$ and the corresponding sentence $X = [x_1, ..., x_n]$ where the length of the linearized graph is k and the length of the text sentence is n, the goal of Text-to-AMR parsing is generating Y given X, and the goal of AMR-to-Text generation is generating X given Y. Specifically, the target of Text-to-AMR parsing is to maximize $P(Y|X)$, and the target of AMR-to-Text generation is to maximize $P(X|Y)$.

Sequence generation in NLP tasks is autoregressive in nature. Therefore, the two conditional probabilities mentioned above could be decomposed according to the following formulations:

$$\begin{aligned} P(Y|X) &= \prod_{t=1}^{k} P(y_t|X, Y_{<t}) \\ P(X|Y) &= \prod_{t=1}^{n} P(x_t|Y, X_{<t}) \end{aligned} \tag{1}$$

where t denotes the index of the decoding step, $Y_{<t}$ and $X_{<t}$ denote the tokens before t position in the linearized AMR graph and the text sentence, respectively. $P(y_t|X, Y_{<t})$ and $P(x_t|Y, X_{<t})$ denote the probability distributions of each word in the target vocabulary being the next candidate.

3.3 Post-processing

For the Text-to-AMR parsing task, post-processing is required to ensure the validity of the generated AMR sequences. We restore parenthesis parity, remove tokens that are not possible descendants of the previous token and remove tokens that are obvious repetitions through the rule-based filter mechanism proposed in SPRING. Moreover, since each kind of wiki tag is sparse in the whole data distribution, it is hard for the seq2seq model to attach correct wiki concepts. Therefore, we use the BLINK Entity Linker (Wu et al., 2020) to overwrite the wiki tags generated by BiBL.

3.4 Encoder-Decoder for Sequence Generation

To model the probability distribution $P(y_t|X, Y_{<t})$ and $P(x_t|Y, X_{<t})$ in Equation 1, we adopt an encoder-decoder model structure. In detail, the encoder first turns the input tokens into embeddings, and then multi-head attention modules transform the embeddings into hidden representations. For the simplicity of the formulation, we ignore the begin-of-sentence `[BOS]`, end-of-sentence `[EOS]`, and separation tokens `[SEP]` in the input tokens. We could represent the process of the encoder as follows:

$$H^e = \text{Encoder}(H^i, \theta_e) \tag{2}$$

where θ_e denotes trainable parameters of the encoder module, $H^i \in \mathcal{R}^{l \times d}$ denotes the input embeddings of length l, $H^e \in \mathcal{R}^{l \times d}$ denotes the output embeddings of the encoder, and d denotes the hidden size of the whole model.

Next, for the decoding step of t during training, the decoder module takes the encoder output embeddings H^e and the ground truth tokens before t position as inputs. Then, it generates a hidden

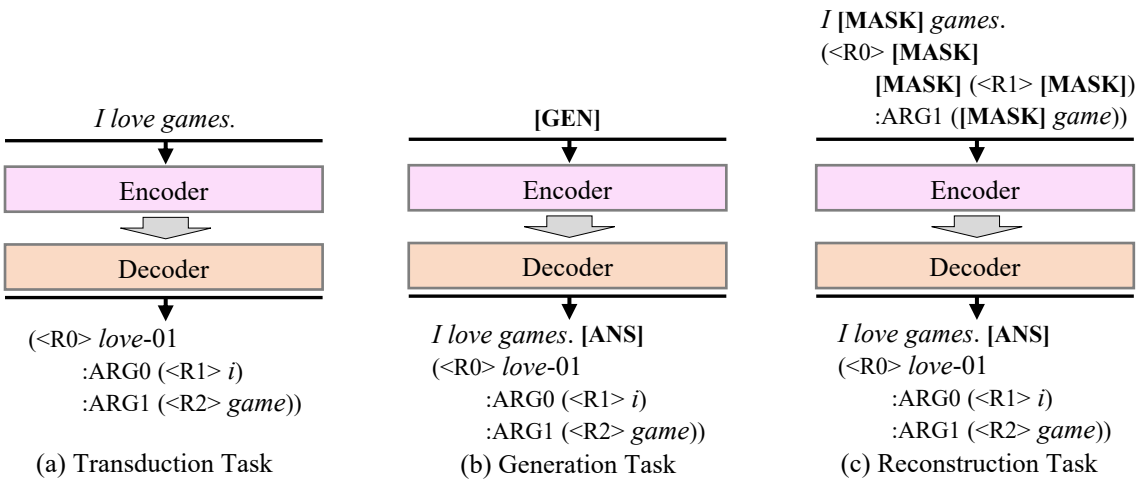

Figure 2: Visualization of the BiBL's multitask learning paradigm. All texts and graphs are linearized in implementations. The indentation in the figure is only for understanding and clarity purposes. Words in the text sentence are italicized, and special tokens are shown in bold.

vector as the output. The computation process of the decoder could be represented as follows:

$$h_t^d = \text{Decoder}(H^e, T_{<t}, \theta_d) \tag{3}$$

where θ_d denotes trainable parameters of the decoder module, H^e denotes the output embeddings of the encoder, $T_{<t}$ denotes the tokens before t position in the target sequence, and $h_t^d \in \mathcal{R}^d$ denotes the output hidden vector. During inference, $T_{<t}$ is replaced with tokens generated by the decoder module before t time step $\hat{T}_{<t}$.

The final step of sequence generation is to generate a probability distribution for the next candidate token utilizing the output hidden vector. Combining the definitions of the encoder and decoder module, the formulation of the whole process is as follows:

$$\text{SG}_t(H^i, \theta_e, \theta_d) = \text{softmax}(h_t^d W^{LM}) \tag{4}$$

where $\text{SG}_t(H^i, \theta_e, \theta_d)$ denotes the probability distribution vector output of t decoding step, $W^{LM} \in \mathcal{R}^{d \times \|V\|}$ denotes a trainable classification matrix that turns the output hidden vector of the decoder h_t^d into a probability distribution of size $\|V\|$, the target vocabulary size.

3.5 Transduction Loss

Following the design of graph linearization and the adoption of the encoder-decoder sequence generation model, we could model the paired transduction tasks between graphs and text sequences as follows. Taking Text-to-AMR parsing as an example, the encoder input should be the text sequence $X = [x_1, ..., x_n]$ and subsequently the encoder outputs a contextualized representation of X. For the training step t, the decoder takes the linearized graph before t position $Y_{<t}$ as the input and autoregressively generates the next graph token while taking the contextualized output of the encoder into account. Similarly, AMR-to-Text generation could be modeled by swapping the text sequence and the linearized graph. An example of the transduction task is shown in Figure 2(a).

Given model parameters θ, two conditional probabilities mentioned in Equation 1 could be formulized as:

$$\begin{aligned} P(Y|X,\theta) &= \prod_{t=1}^{k} P(y_t|X, Y_{<t}, \theta) = \prod_{t=1}^{k} \text{SG}_t(H_X^i, \theta_e, \theta_d) \\ P(X|Y,\theta) &= \prod_{t=1}^{n} P(x_t|Y, X_{<t}, \theta) = \prod_{t=1}^{n} \text{SG}_t(H_Y^i, \theta_e, \theta_d) \end{aligned} \tag{5}$$

where $H_X^i \in \mathcal{R}^{n \times d}$ and $H_Y^i \in \mathcal{R}^{k \times d}$ denote the input embeddings of sequence X and sequence Y, respectively.

In order to maximize these two conditional probabilities, transduction loss is defined as one of the training objectives:

$$\begin{aligned} l_{td}^{A2T} &= -\sum_{t=1}^{n} \log P(x_t|Y, X_{<t}, \theta) \\ l_{td}^{T2A} &= -\sum_{t=1}^{k} \log P(y_t|X, Y_{<t}, \theta) \end{aligned} \tag{6}$$

where l_{td}^{A2T} and l_{td}^{T2A} denote the sample-wise transduction loss defined for AMR-to-Text generation and Text-to-AMR parsing, respectively.

$P(y_t|X, Y_{<t}, \theta)$ and $P(x_t|Y, X_{<t}, \theta)$ denote the probability of generating y_t and x_t, which are the ground truth labels in respective transduction tasks.

3.6 Bidirectional Bayesian Learning

Inspecting previous refinements and other general methods for bidirectional enhancement including dual learning (He et al., 2016), their improvements are mainly due to the inclusion of large pre-training data or complex structure modifications. However, our proposed method is data-efficient and can achieve great model improvements without extra pre-training data and model structure changes.

According to the Bayesian rules, we could find that two conditional probabilities in Equation 1 are interconnected through a common factor $P(X, Y)$:

$$\begin{aligned} P(Y|X) &= P(X, Y)/P(X) \\ P(X|Y) &= P(X, Y)/P(Y) \end{aligned} \tag{7}$$

Therefore, our motivation is that the incorporation of joint probability $P(X, Y)$ could facilitate the model's understanding of the joint probabilistic space of AMR graphs and texts. The model performance could be enhanced on both sides of the transduction task with proper design. The extra design should first clearly model the joint probability $P(X, Y)$ and then should take measures to avoid causing excessive confusion.

We propose two ways to model the joint probability $P(X, Y)$, and two auxiliary loss functions are included in our model design, i.e., generation loss and reconstruction loss. Utilizing the concatenation of text sentences and corresponding linearized graphs, two tailored auxiliary tasks help the model grasp the mutual information between two data forms, which is beneficial for the basic transduction task. Meanwhile, various effective measures are taken to avoid causing excessive confusion. Visualizations of the model input and output for these two auxiliary tasks are shown in Figure 2(b,c).

Inspired by LAMOL (Sun et al., 2020), we feed the model encoder with a specially designed `[GEN]` token. Denoting the `[GEN]` token as s_g, the encoder input is represented as $G = [s_g]$. Then the encoder generates an output signal which the decoder attends to, and the decoder would begin a special generation process. For training step t, denoting the concatenation of a text sequence and the corresponding linearized graph as $Z = [X; Y]$. To form the concatenation, we insert another special `[ANS]` token between X and Y sequences to separate them, creating discrimination from the original task. The decoder is expected to re-generate Z in an auto-regressive fashion. We adopt teacher forcing for the training of the selected encoder-decoder. For generation step t, the decoder is fed with ground truth labels before the t position. Consequently, it is possible for the model to re-generate paired data without much information provided to the encoder. The corresponding joint probability and the sample-wise generation loss are defined as follows:

$$\begin{aligned} P([X; Y]) &= \prod_{t=1}^{n+k} P(z_t|G, Z_{<t}, \theta) = \prod_{t=1}^{n+k} \mathrm{SG}_t(H_G^i, \theta_e, \theta_d) \\ l_g &= -\sum_{t=1}^{n+k} \log P(z_t|G, Z_{<t}, \theta) \end{aligned} \tag{8}$$

where $H_G^i \in \mathcal{R}^{1\times d}$ denotes the input embedding of G, and l_g denotes the auxiliary generation loss. This auxiliary generation task could help the decoder grasp the underlying joint probability distribution of the graph-text data and facilitates the model to gain more knowledge from the training data. The `[GEN]` token input for the encoder works as a special signal for the decoder to maintain clear discrimination between the generation of the concatenated sequences in the generation task and the target sequences in the original transduction task. Discriminated by `[GEN]` token, it would not impede the learning of the encoder-decoder attention mechanism in the main transduction task. Therefore, the auxiliary generation task could facilitate the main transduction task and avoid causing excessive confusion.

Besides having a thorough understanding of the joint probability distribution of the training data pairs, BiBL needs to learn a universal joint distribution that is applicable to open-world data. Inspired by masked language modeling (Devlin et al., 2019), we adopt the following reconstruction task. First, we concatenate the text sentence and the linearized graph to get $Z = [X; Y]$. Similar to the generation auxiliary task, `[ANS]` token is inserted between X and Y sequences, which could prevent excessive confusion brought to the original transduction task. Then we randomly mask 50% of the concatenation to get a masked concatenation $M = [x_1, ..., s_m, ..., x_n, s_a, y_1, ..., s_m, ..., y_k]$, where s_m denotes the `[MASK]` token and s_a denotes the `[ANS]` token. Taking this masked concatenation as input, the encoder generates a contextualized masked representation. Attending to

the representation generated by the encoder, the decoder is expected to auto-regressively reconstruct the original concatenation Z. In this auxiliary task, the reconstruction loss would be computed over all tokens generated by the decoder. The corresponding joint probability and the sample-wise reconstruction loss are defined as follows:

$$P([X;Y]) = \prod_{t=1}^{n+k} P(z_t|M, Z_{<t}, \theta) = \prod_{t=1}^{n+k} \mathrm{SG}_t(H_M^i, \theta_e, \theta_d)$$
$$l_r = -\sum_{t=1}^{n+k} \log P(z_t|M, Z_{<t}, \theta) \tag{9}$$

where $H_M^i \in \mathcal{R}^{(n+k)\times d}$ denotes the input embedding of the masked input M and l_r denotes the auxiliary reconstruction loss. This auxiliary task reinforces the decoder to understand the condensed representation produced by the encoder. Meanwhile, since masks could be applied simultaneously to texts and AMR graphs, the reconstruction of masked tokens helps the model adapt to the special structures of the graph linearization results while facilitating possible information alignment between two forms of data. This knowledge could be extended to general situations and is applicable to open-world data.

3.7 Single-stage Multitask Learning

A multitask learning paradigm is used in our training process to combine three loss functions mentioned above, i.e., transduction loss l_{td}^{A2T} or l_{td}^{T2A}, generation loss l_g and reconstruction loss l_r. The formulae are as follows:

$$l^{A2T} = l_{td}^{A2T} + \lambda_g l_g + \lambda_r l_r$$
$$l^{T2A} = l_{td}^{T2A} + \lambda_g l_g + \lambda_r l_r \tag{10}$$

where l^{A2T} and l^{T2A} denote the loss functions of AMR-to-Text generation and Text-to-AMR parsing, respectively, λ_g is the weight of the generation loss, and λ_r is the weight of the reconstruction loss. The loss is computed for every sample while averaged over a single batch. Moreover, the input sequences in the transduction task are placed in front of the target sequences for concatenation.

Since the joint probability distribution learned in two auxiliary tasks is conducive to the understanding of the conditional probability that corresponds to the original transduction task, it should be better to interweave the auxiliary tasks with the transduction task. Meanwhile, three measures are taken to avoid causing disturbances over the main transduction task. First, for the generation task, the special `[GEN]` token input for the encoder creates discrimination between the generation task and the original task. Second, for both auxiliary tasks, we utilize another special `[ANS]` token to separate two sequences, creating further discrimination from the original task. Third, by choosing the best loss weight configuration, the relationship among the three tasks could be adjusted and unnecessary confusion could be minimized. Therefore, different from all previous refinements that adopt a two-stage training paradigm of pre-training and fine-tuning, a single-stage multitask learning paradigm is explored. Results listed in the ablation studies 6 could show that this learning paradigm can greatly simplify the training process while retaining the extra performance boost by incorporating the extra knowledge.

4 Experimental Setup

Based on the evaluation system of previous works (Bevilacqua et al., 2021; Chen et al., 2022; Hoang et al., 2021), we inspect the performance of BiBL in two settings: In-Distribution and Out-of-Distribution. The in-distribution setting shows the power of BiBL on standard benchmarks. Since the bidirectional transitions between AMR graphs and texts should be modeled as a unified process without dependence on language contexts, it is also important to evaluate our model in out-of-distribution settings. The experiment benchmark system uses Nvidia Titan RTX as GPU and Intel Xeon E5-2637@3.50Ghz as CPU. The system is built with 256GB RAM.

4.1 Datasets

In-Distribution The data we use in the in-distribution setting are **AMR 2.0** (LDC2017T10) and **AMR 3.0** (LDC2020T02) corpora releases. AMR 3.0 is an expanded version of AMR 2.0 with additional AMR-text data to cover more general situations. AMR 3.0 contains $55,635$ pairs of graphs and texts, and AMR 2.0 contains $36,521$ pairs.

Out-of-Distribution Following the benchmark proposed in SPRING, we incorporate three evaluation datasets in the out-of-distribution setting: i) **TLP**, an AMR-tagged dataset containing $1,562$ data pairs constructed from the famous children's novel *The Little Prince*.; ii) **Bio**, full data from the Bio-AMR corpus (May and Priyadarshi, 2017) containing $6,952$ data pairs. While in SPRING, only the test data are used in the evaluation, we

use all data splits to conduct a thorough evaluation; iii) **New3**, a set of data excerpted from AMR 3.0 containing 527 data pairs, whose original source is the LORELEI DARPA project.

Silver Data Silver data is generated according to the following process. First, we obtain the best Text-to-AMR parsing BiBL model trained on the target dataset (AMR 2.0 or AMR 3.0). Then, we choose **English Gigaword** (LDC2003T05) as an unlabeled dataset. We use the previously trained Text-to-AMR BiBL model to generate annotations over randomly chosen sentences from the unlabeled dataset. Finally, a newly annotated dataset for extra training is acquired. For BiBL, the silver data we generate contains 200k sentence-graph pairs.

4.2 Models

Model Setting BART with augmented vocabulary is chosen as the pre-trained encoder-decoder model for BiBL. We inherit the model hyperparameters from BART Large, defined in Huggingface's `transformers` library. For training-related hyperparameters, our models are trained for 30 epochs using cross-entropy with a batch size of 500 graph linearization tokens, RAdam (Liu et al., 2020) optimizer, and a learning rate of 1×10^{-5}. The gradient is accumulated for 10 batches. The dropout is set to 0.25.

BiBL variants During experiments, we have done an empirical study on the best weight configuration for generation and reconstruction loss. Details are shown in Appendix A. For Text-to-AMR parsing, we set λ_g as 1.0 and λ_r as 0.5. For AMR-to-Text generation, we set λ_g as 0.15 and λ_r as 0.5. Models with λ_g or λ_r as 0 are included for ablation studies. BiBL variants trained with extra silver data are also included in the experiments. The silver data will be directly included in the single-stage multitask learning without further fine-tuning.

4.3 Evaluation Metrics

Text-to-AMR Parsing Fine-grained Smatch (Cai and Knight, 2013) is chosen as the evaluation metric for Text-to-AMR parsing since previous works all use this metric, and it reports fine-grained scores on different aspects of the parsed AMR graphs.

AMR-to-Text Generation Following previous works, we evaluate BiBL with three common Natural Language Generation measures, i.e., BLEU (Papineni et al., 2002), chrF++ (Popović, 2017), and METEOR (Banerjee and Lavie, 2005).

4.4 Comparison System

Text-to-AMR Parsing SPRING (Bevilacqua et al., 2021), Ancestor (Yu and Gildea, 2022), Graphene4S (Hoang et al., 2021), AMRize (Chen et al., 2022) and GraphPre (Bai et al., 2022) mentioned in Section 1 are included.

AMR-to-Text Generation SPRING (Bevilacqua et al., 2021), ReconCov (Anonymous, 2021), STA (Ribeiro et al., 2021) and GraphPre (Bai et al., 2022) mentioned in Section 1 are included.

5 Results

Text-to-AMR parsing According to Tables 1 and 2, under the setting of no extra training data and no complex ensembling techniques applied, BiBL outperforms all other refinements on AMR 3.0. Moreover, for AMR 2.0 and AMR 3.0, BiBL outperforms all previous refinements on the fine-grained Named Entity Recognition (NER) metric by a large margin. Especially for AMR 3.0, the performance metric of NER is increased by 4.3 even compared with AMRize. Besides, Wikification and topology-related Unlabeled metrics are also improved. This shows that the extra reconstruction task helps the model grasp the alignment between texts and AMR graphs. These alignments include special entity mappings and structural mappings, which are beneficial to NER and topology-related unlabeled metrics.

AMR-to-Text generation According to Tables 3 and 4, outperforming all previous methods by a significant margin (1.7 BLEU for AMR 2.0 and 1.5 BLEU for AMR 3.0), BiBL with silver data achieves a new state-of-the-art performance on AMR-to-Text generation. Meanwhile, BiBL with no silver data also surpasses all previous refinements under the same setting, proving that BiBL is the most effective refinement without the need for extra training data and the pre-training setting.

Out-of-Distribution Table 6 shows that BiBL outperforms several models (Graphene and SPRING) under the out-of-distribution setting without extra training data or graph ensembling techniques. This confirms that BiBL facilitates the model to grasp the joint probability distribution of AMR graphs and texts which is applicable to open-world data.

Bidirectional Performance Enhancement Inspecting BiBL on the more comprehensive AMR 3.0 dataset, under the setting of no extra data and no ensembling techniques, BiBL achieves the best results on Text-to-AMR parsing. For AMR-to-Text

Model	Extra Data	Ensemble	Smatch	NoWSD	Wiki.	Conc.	NER	Neg.	Unlab.	Reent.	SRL
SPRING (2021)	N	N	83.8	84.4	**84.3**	90.2	90.6	**74.4**	86.1	70.8	79.6
*Ancestor (2022)	N	N	**84.8**	**85.3**	84.1	90.5	91.8	74.0	**88.1**	**75.1**	**83.4**
BiBL (ours)	N	N	84.6	85.1	83.6	90.3	**92.5**	73.9	87.8	74.4	83.1
SPRING (2021)	200k	N	84.3	84.8	83.1	90.8	90.5	73.6	86.7	72.4	80.5
Graphene4S (2021)	200k	Y	84.8	85.3	**83.9**	90.6	92.2	**75.2**	88.0	71.4	83.5
*AMRize (2022)	40k	Y	85.3	85.7	**83.9**	90.7	92.2	75.0	**88.4**	**75.0**	**83.6**
*GraphPre (2022)	200k	N	**85.4**	**85.8**	81.4	**91.2**	91.5	74.0	88.3	73.5	81.5
BiBL (ours)	200k	N	84.7	85.1	83.2	90.4	**92.6**	75.0	87.8	74.6	83.3

Table 1: Text-to-AMR parsing results on AMR 2.0 dataset. Row blocks: baseline + approaches without any extra training data and ensembling techniques; approaches with extra training data or ensembling techniques. Columns: Model; Smatch; Fine-grained scores. The best results within each block are shown in bold. Models with * denote contemporary works.

Model	Extra Data	Ensemble	Smatch	NoWSD	Wiki.	Conc.	NER	Neg.	Unlab.	Reent.	SRL
SPRING (2021)	N	N	83.0	83.5	82.7	**89.8**	87.2	**73.0**	85.4	70.4	78.9
*Ancestor (2022)	N	N	83.5	84.0	81.5	89.5	88.9	72.6	86.6	**74.2**	**82.2**
BiBL (ours)	N	N	**83.9**	**84.3**	**83.7**	**89.8**	**93.2**	68.1	**87.2**	73.8	81.9
SPRING (2021)	200k	N	83.0	83.5	81.2	89.5	87.1	71.7	85.4	71.3	79.1
Graphene4S (2021)	200k	Y	83.8	84.2	81.9	90.1	88.3	**74.6**	86.9	70.2	82.5
*AMRize (2022)	40k	Y	84.0	84.5	80.7	90.0	**88.9**	73.1	**87.1**	**73.9**	**82.6**
*GraphPre (2022)	200k	N	**84.2**	**85.8**	81.4	**90.2**	88.5	72.1	**87.1**	72.4	80.3
BiBL (ours)	200k	N	83.5	84.0	**82.1**	89.6	**88.9**	73.3	86.7	73.7	82.2

Table 2: Text-to-AMR parsing results on AMR 3.0 dataset. Row blocks/Columns/Bold/* as in Table 1.

Model	Extra Data	BL	CH++	MET
SPRING (2021)	N	45.3	73.5	41.0
*ReconCov (2021)	N	45.4	73.6	42.4
BiBL (ours)	N	**47.0**	**74.8**	**43.2**
SPRING (2021)	200k	45.9	74.2	41.8
STA (2021)	2000k	49.7	-	**45.4**
*GraphPre (2022)	200k	49.8	76.2	42.6
BiBL (ours)	200k	**51.5**	**77.6**	45.2

Table 3: AMR-to-Text generation results on AMR 2.0 dataset. Row blocks: approaches without extra training data; approaches with extra training data. Columns: Model; BLEU; chrF++; METEOR. The best results within each block are shown in bold. Models with * denote contemporary works.

Model	Extra Data	BL	CH++	MET
SPRING (2021)	N	44.9	72.9	40.6
*ReconCov (2021)	N	45.7	73.7	42.8
BiBL (ours)	N	**47.4**	**74.5**	**43.4**
SPRING (2021)	200k	46.5	73.9	41.7
*GraphPre (2022)	200k	49.2	76.1	42.3
BiBL (ours)	200k	**50.7**	**76.7**	**45.0**

Table 4: AMR-to-Text generation results on AMR 3.0 dataset. Row blocks/Columns/Bold/* as in Table 3.

Model	SMATCH	BLEU
BiBL (silver)		
- two-stage	84.70	51.49
- single-stage	84.65	51.45

Table 5: Experiments on training paradigms. AMR 2.0 is chosen as the dataset. The recorded results are the best results achieved among all BiBL variants.

generation, BiBL with single-stage multitask learning surpasses the current state-of-the-art GraphPre model. Since the two-stage training paradigm of pre-training and fine-tuning is adopted by GraphPre, the performance boost proves that the single-stage multitask learning could not only simplify the training process but also contribute to the final model performance. For both transduction directions, BiBL with no silver data even surpasses several refinements that need extra training data (SPRING, Graphene4S) in both in-distribution and out-of-distribution settings. These results show that our proposed method is data-efficient and effective on bidirectional transitions. A case study is included in Appendix B for further examination.

We have also conducted experiments on model efficiency. According to Table 8, the time spent pre-training and fine-tuning GraphPre, which is the current state-of-the-art refinement, is 8.6x of the time spent training BiBL through single-phase mul-

Model	Extra Data	Ensemble	TLP	Bio	New3
Text-to-AMR					
SPRING (2021)	N	N	77.3	59.7	73.7
Graphene4S (2021)	200k	Y	77.9	**61.5**	74.8
*AMRize (2022)	40k	N	**78.9**	61.2	75.4
*GraphPre (2022)	200k	N	76.9	**63.2**	**76.9**
BiBL (ours)	N	N	78.6	61.0	75.4
BiBL (ours)	200k	N	78.3	61.1	75.4
AMR-to-Text					
SPRING (2021)	N	N	24.3	18.9	37.2
*GraphPre (2022)	200k	N	**29.1**	20.7	44.8
BiBL (ours)	N	N	26.1	20.3	39.0
BiBL (ours)	200k	N	28.2	**24.4**	**45.0**

Table 6: OOD evaluation on Text-to-AMR (Smatch) and AMR-to-Text (BLEU). All models are trained on the AMR 2.0 dataset. Bold/* as in Table 1.

Model	Extra Data	SMATCH	NER	BLEU
BiBL				
- $\lambda_g = 0.15 + \lambda_r = 0.5$	N	-	-	47.4
- $\lambda_g = 1.0 + \lambda_r = 0.5$	N	**83.9**	**93.2**	-
- $\lambda_g = 0.15 + \lambda_r = 0.5$	200k	-	-	**50.1**
- $\lambda_g = 1.0 + \lambda_r = 0.5$	200k	83.5	88.9	-
- $\lambda_g = 0 + \lambda_r = 0.5$	N	83.7	89.1	47.3
- $\lambda_g = 0.15 + \lambda_r = 0$	N	82.8	88.2	45.2
- $\lambda_g = 1.0 + \lambda_r = 0$	N	83.0	88.3	45.5

Table 7: Ablation studies on Text-to-AMR (Smatch and fine-grained NER) and AMR-to-Text (BLEU) using AMR 3.0 as the benchmark. Bold denotes the best results.

titask training under the same setting. According to Table 9, since the ensembling of Graphene4S is computed with CPUs, the time spent ensembling four graphs generated by Graphene4S is 22.8x of the time spent by BiBL for the inference of one training sample when the beam size is set to 5.

6 Ablation Studies

Since AMR 3.0 is the superset of AMR 2.0, our ablation studies are based on the AMR 3.0 dataset so that a more comprehensive investigation could be conducted. Through a thorough analysis of Table 7, the following conclusions could be made:

• Inspecting all variants of BiBL for Text-to-AMR parsing on both AMR 2.0 and AMR 3.0, if the scope of the original dataset is limited, silver data could have minor contributions to BiBL on the Text-to-AMR parsing task. Otherwise, silver data may hurt the model performance. For the opposite AMR-to-Text generation task, the performance boost of silver data on BiBL is distinct.

• The combination of the auxiliary reconstruction and generation tasks achieves the best model performance, proving that both tasks are conducive and indispensable to the main transduction task.

Model	Extra Data	Ensemble	Time	Multiplier
SPRING (2021)	N	N	35.63	1.0x
*GraphPre[pre] (2022)	200k	N	267.23	7.5x
*GraphPre[fine] (2022)	N	N	38.73	1.1x
*GraphPre[total] (2022)	200k	N	305.96	8.6x
BiBL (ours)	200k	N	128.38	3.6x

Table 8: Training efficiency evaluation on different refinement approaches. Time is evaluated using the same computation resource and averaged over three batches. The unit of Time is a millisecond.

Model	Ensemble	Time	Multiplier
BARTlarge-based models	-	133.99	1.0x
SPRING (2021)	N		
*GraphPre (2022)	N		
BiBL (ours)	N		
Graphene4S (2021)	Y	3050.13	22.8x

Table 9: Inference efficiency evaluation on different refinement approaches. Time is evaluated using the same computation resource and averaged over a single batch. The unit of Time is a millisecond.

BiBL enables the model to effectively capture the underlying information in the joint probabilistic space of AMR graphs and texts.

BiBL is trained with single-stage multitask learning. Further experiments are conducted to compare the chosen paradigm with the two-stage training paradigm. For two-stage training, we pre-train BiBL with auxiliary tasks and then fine-tune BiBL without them. Results listed in Table 5 show that the benefits brought by fine-tuning are rather trivial. This proves that through proper loss weight configuration, auxiliary tasks could help BiBL to achieve better performance on the main transduction task without causing confusion. Therefore, the training paradigm could be reduced to single-stage multitask learning without losing performance boost.

7 Conclusion

In this paper, we propose BiBL which is capable of enhancing the seq2seq approach to tackle the bidirectional transition task between AMR graphs and texts. BiBL could learn the joint probability of two data forms in a data-efficient way through single-stage multitask learning with auxiliary generation and reconstruction tasks. Eventually, model performances are greatly improved in in-distribution and out-of-distribution settings. Moreover, our model outperforms all previous refinements with no extra data on Text-to-AMR parsing and achieves the new state-of-the-art result on AMR-to-Text generation.

References

Anonymous. 2021. Amr-to-text generation with graph structure reconstruction and coverage. In *Review of Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics*.

Xuefeng Bai, Yulong Chen, and Yue Zhang. 2022. Graph pre-training for AMR parsing and generation. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, Online. Association for Computational Linguistics.

Laura Banarescu, Claire Bonial, Shu Cai, Madalina Georgescu, Kira Griffitt, Ulf Hermjakob, Kevin Knight, Philipp Koehn, Martha Palmer, and Nathan Schneider. 2013. Abstract Meaning Representation for sembanking. In *Proceedings of the 7th Linguistic Annotation Workshop and Interoperability with Discourse*, pages 178–186, Sofia, Bulgaria. Association for Computational Linguistics.

Satanjeev Banerjee and Alon Lavie. 2005. METEOR: An automatic metric for MT evaluation with improved correlation with human judgments. In *Proceedings of the ACL Workshop on Intrinsic and Extrinsic Evaluation Measures for Machine Translation and/or Summarization*, pages 65–72, Ann Arbor, Michigan. Association for Computational Linguistics.

Daniel Beck, Gholamreza Haffari, and Trevor Cohn. 2018. Graph-to-sequence learning using gated graph neural networks. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 273–283, Melbourne, Australia. Association for Computational Linguistics.

Michele Bevilacqua, Rexhina Blloshmi, and Roberto Navigli. 2021. One SPRING to rule them both: Symmetric AMR semantic parsing and generation without a complex pipeline. In *Thirty-Fifth AAAI Conference on Artificial Intelligence, AAAI 2021, Thirty-Third Conference on Innovative Applications of Artificial Intelligence, IAAI 2021, The Eleventh Symposium on Educational Advances in Artificial Intelligence, EAAI 2021, Virtual Event, February 2-9, 2021*, pages 12564–12573. AAAI Press.

Deng Cai and Wai Lam. 2020a. AMR parsing via graph-sequence iterative inference. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 1290–1301, Online. Association for Computational Linguistics.

Deng Cai and Wai Lam. 2020b. Graph transformer for graph-to-sequence learning. In *The Thirty-Fourth AAAI Conference on Artificial Intelligence, AAAI 2020, The Thirty-Second Innovative Applications of Artificial Intelligence Conference, IAAI 2020, The Tenth AAAI Symposium on Educational Advances in Artificial Intelligence, EAAI 2020, New York, NY, USA, February 7-12, 2020*, pages 7464–7471. AAAI Press.

Shu Cai and Kevin Knight. 2013. Smatch: an evaluation metric for semantic feature structures. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 748–752, Sofia, Bulgaria. Association for Computational Linguistics.

Liang Chen, Peiyi Wang, Runxin Xu, Tianyu Liu, Zhifang Sui, and Baobao Chang. 2022. Atp: Amrize then parse! enhancing amr parsing with pseudoamrs.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, Minneapolis, Minnesota. Association for Computational Linguistics.

Zhijiang Guo, Yan Zhang, Zhiyang Teng, and Wei Lu. 2019. Densely connected graph convolutional networks for graph-to-sequence learning. *Transactions of the Association for Computational Linguistics*, 7:297–312.

Di He, Yingce Xia, Tao Qin, Liwei Wang, Nenghai Yu, Tie-Yan Liu, and Wei-Ying Ma. 2016. Dual learning for machine translation. In *Advances in Neural Information Processing Systems*, volume 29. Curran Associates, Inc.

Thanh Lam Hoang, Gabriele Picco, Yufang Hou, Young-Suk Lee, Lam Nguyen, Dzung Phan, Vanessa López, and Ramon Fernandez Astudillo. 2021. Ensembling graph predictions for amr parsing. *Advances in Neural Information Processing Systems*, 34.

Ioannis Konstas, Srinivasan Iyer, Mark Yatskar, Yejin Choi, and Luke Zettlemoyer. 2017. Neural AMR: Sequence-to-sequence models for parsing and generation. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 146–157, Vancouver, Canada. Association for Computational Linguistics.

Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Veselin Stoyanov, and Luke Zettlemoyer. 2020. BART: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7871–7880, Online. Association for Computational Linguistics.

Liyuan Liu, Haoming Jiang, Pengcheng He, Weizhu Chen, Xiaodong Liu, Jianfeng Gao, and Jiawei Han. 2020. On the variance of the adaptive learning rate and beyond. In *8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, April 26-30, 2020*. OpenReview.net.

Manuel Mager, Ramón Fernandez Astudillo, Tahira Naseem, Md Arafat Sultan, Young-Suk Lee, Radu Florian, and Salim Roukos. 2020. GPT-too: A language-model-first approach for AMR-to-text generation. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 1846–1852, Online. Association for Computational Linguistics.

Jonathan May and Jay Priyadarshi. 2017. SemEval-2017 task 9: Abstract Meaning Representation parsing and generation. In *Proceedings of the 11th International Workshop on Semantic Evaluation (SemEval-2017)*, pages 536–545, Vancouver, Canada. Association for Computational Linguistics.

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. Bleu: a method for automatic evaluation of machine translation. In *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics*, pages 311–318, Philadelphia, Pennsylvania, USA. Association for Computational Linguistics.

Maja Popović. 2017. chrF++: words helping character n-grams. In *Proceedings of the Second Conference on Machine Translation*, pages 612–618, Copenhagen, Denmark. Association for Computational Linguistics.

Leonardo F. R. Ribeiro, Martin Schmitt, Hinrich Schütze, and Iryna Gurevych. 2021. Investigating pretrained language models for graph-to-text generation. In *Proceedings of the 3rd Workshop on Natural Language Processing for Conversational AI*, pages 211–227, Online. Association for Computational Linguistics.

Linfeng Song, Yue Zhang, Zhiguo Wang, and Daniel Gildea. 2018. A graph-to-sequence model for AMR-to-text generation. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1616–1626, Melbourne, Australia. Association for Computational Linguistics.

Fan-Keng Sun, Cheng-Hao Ho, and Hung-Yi Lee. 2020. LAMOL: language modeling for lifelong language learning. In *8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, April 26-30, 2020*. OpenReview.net.

Rik van Noord and Johan Bos. 2017. Neural semantic parsing by character-based translation: Experiments with abstract meaning representations. *CoRR*, abs/1705.09980.

Tianming Wang, Xiaojun Wan, and Hanqi Jin. 2020a. AMR-to-text generation with graph transformer. *Transactions of the Association for Computational Linguistics*, 8:19–33.

Tianming Wang, Xiaojun Wan, and Shaowei Yao. 2020b. Better amr-to-text generation with graph structure reconstruction. In *Proceedings of the Twenty-Ninth International Joint Conference on Artificial Intelligence, IJCAI-20*, pages 3919–3925. International Joint Conferences on Artificial Intelligence Organization. Main track.

Ledell Wu, Fabio Petroni, Martin Josifoski, Sebastian Riedel, and Luke Zettlemoyer. 2020. Scalable zero-shot entity linking with dense entity retrieval. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 6397–6407, Online. Association for Computational Linguistics.

Chen Yu and Daniel Gildea. 2022. Sequence-to-sequence AMR parsing with ancestor information. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (ACL-22)*.

Sheng Zhang, Xutai Ma, Kevin Duh, and Benjamin Van Durme. 2019a. AMR parsing as sequence-to-graph transduction. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 80–94, Florence, Italy. Association for Computational Linguistics.

Sheng Zhang, Xutai Ma, Kevin Duh, and Benjamin Van Durme. 2019b. Broad-coverage semantic parsing as transduction. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 3786–3798, Hong Kong, China. Association for Computational Linguistics.

Jiawei Zhou, Tahira Naseem, Ramón Fernandez Astudillo, Young-Suk Lee, Radu Florian, and Salim Roukos. 2021. Structure-aware fine-tuning of sequence-to-sequence transformers for transition-based AMR parsing. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 6279–6290, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Qiji Zhou, Yue Zhang, Donghong Ji, and Hao Tang. 2020. AMR parsing with latent structural information. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 4306–4319, Online. Association for Computational Linguistics.

Jie Zhu, Junhui Li, Muhua Zhu, Longhua Qian, Min Zhang, and Guodong Zhou. 2019. Modeling graph structure in transformer for better AMR-to-text generation. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 5459–5468, Hong Kong, China. Association for Computational Linguistics.

	SMATCH	
	AMR 2.0	AMR 3.0
BiBL		
- $\lambda_g = 0.00$	<u>84.06</u>	83.81
- $\lambda_g = 0.15$	83.98	83.92
- $\lambda_g = 0.25$	84.01	**84.06**
- $\lambda_g = 0.50$	83.92	83.83
- $\lambda_g = 1.00$	**84.17**	<u>83.98</u>

Table 10: Text-to-AMR parsing experiments on weight configurations of the generation auxiliary task. Bold denotes the best results. Underline denotes second-best results. Results are reported on the validation set.

	BLEU	
	AMR 2.0	AMR 3.0
BiBL		
- $\lambda_g = 0.00$	40.21	41.50
- $\lambda_g = 0.15$	**40.34**	**41.72**
- $\lambda_g = 0.25$	40.19	41.20
- $\lambda_g = 0.50$	39.89	41.24
- $\lambda_g = 1.00$	<u>40.23</u>	<u>41.52</u>

Table 11: AMR-to-Text generation experiments on weight configurations of the generation auxiliary task. Bold denotes the best results. Underline denotes second-best results. Results are reported on the validation set.

A Empirical Study on Weight Configuration of Auxiliary Tasks

For the weight configuration of the auxiliary tasks proposed for Bidirectional Bayesian Learning (BiBL), we have experimented with several BiBL variants of different λ_g and λ_r. All experimental models are trained without extra silver data in order to discriminate the direct effect of different parameters.

During experiments, we find that the magnitudes of transduction, generation, and reconstruction loss should be on the same level. Otherwise, the generation validity of BiBL will be greatly influenced by auxiliary tasks constructed using the concatenation of AMR graphs and texts. Therefore, reasonable ranges of λ_g and λ_r could be roughly determined. Moreover, we find that the weight sensitivity of the generation task is relatively high compared with the auxiliary reconstruction task. Therefore, it is necessary to conduct a grid search over reasonable ranges of λ_g. Since the weight sensitivity of the reconstruction task is relatively low according to our conclusion, 0.5 is chosen for λ_r.

According to the results presented in Tables 10, 11, 12, and 13, for Text-to-AMR parsing, it is obvious that BiBL variant with $\lambda_g = 1.00$ is the best choice. However, for AMR-to-Text generation, the effect of λ_g varies across datasets. Focusing on achieving the best performance on in-distribution datasets, BiBL variants with $\lambda_g = 0.15$ and $\lambda_g = 1.00$ are the best models. However, according to Table 14, after inspecting the combination effect of generation and reconstruction tasks on AMR-to-Text generation, BiBL variant with $\lambda_g = 0.15$ and $\lambda_r = 0.5$ greatly outperforms the model with $\lambda_g = 1.00$ and $\lambda_r = 0.5$.

	SMATCH		
	AMR 2.0	AMR 3.0	TLP
BiBL			
- $\lambda_g = 0.00$	83.80	<u>83.00</u>	77.30
- $\lambda_g = 0.15$	**84.28**	82.78	<u>77.91</u>
- $\lambda_g = 0.25$	83.97	82.93	77.88
- $\lambda_g = 0.50$	83.86	**83.30**	77.62
- $\lambda_g = 1.00$	<u>84.23</u>	<u>83.00</u>	**77.94**

Table 12: Text-to-AMR parsing experiments on weight configurations of the generation auxiliary task. Bold denotes the best results. Underline denotes second-best results. Results are reported on the test set.

Therefore, according to the above analysis, respectively for Text-to-AMR parsing and AMR-to-Text generation, BiBL variants with $\lambda_g = 1.0$ and $\lambda_r = 0.5$ and $\lambda_g = 0.15$ and $\lambda_r = 0.5$ are chosen due to their excellent performances.

B Case Study

In section 5, we analyzed the performance of BiBL according to chosen metrics. Here we present four cases to further inspect the improvements of BiBL. First, we inspect two representative cases for Text-to-AMR parsing. Figure 3 shows a case where BiBL correctly labels the named entity *OCD* as a disease. However, in the output AMR graph of the SPRING baseline, the named entity is falsely labeled as a medical condition, which is too general. This proves that BiBL could better recognize and understand the named entities included in the text sentences, which leads to a large performance boost of BiBL on the fine-grained NER metric. Figure 4 shows a case where BiBL correctly recognizes the reentrancy that occurred in the text. Although the short sentence provides limited semantic informa-

	BLEU		
	AMR 2.0	AMR 3.0	TLP
BiBL			
- $\lambda_g = 0.00$	**44.77**	44.90	24.37
- $\lambda_g = 0.15$	<u>44.65</u>	45.18	23.83
- $\lambda_g = 0.25$	43.61	**45.50**	24.17
- $\lambda_g = 0.50$	44.30	45.35	**25.11**
- $\lambda_g = 1.00$	44.62	<u>45.47</u>	<u>24.78</u>

Table 13: AMR-to-Text generation experiments on weight configurations of the generation auxiliary task. Bold denotes the best results. Underline denotes second-best results. Results are reported on the test set.

Model	BLEU
BiBL	
- $\lambda_g = 0.15 + \lambda_r = 0.5$	**46.95**
- $\lambda_g = 1.00 + \lambda_r = 0.5$	46.65

Table 14: Experiments on the combination effect of generation and reconstruction tasks. AMR 2.0 is chosen as the dataset.

tion and the speaker information is indicated by a colon separation, BiBL still fully understands the meaning of the sentence and generated the correct reentrancy relations in the AMR graph result. However, in SPRING-generated AMR graphs, the reentrancy information is missing. This proves that BiBL could still understand the structures of text sentences even with limited context provided.

Then, we inspect two representative cases for AMR-to-Text generation. Figure 5 shows a case where BiBL could understand the complex semantic meaning indicated by the AMR graph. While the SPRING baseline misrepresents the semantic relation between *sight* and *hope*, BiBL could restore the correct meaning in the generated text by using the same grammar structure in the gold label. This proves that BiBL could accurately grasp the underlying semantic meaning of texts and correctly represent the information using proper grammar structures. Figure 6 shows a case where a complex AMR graph is involved. The relations between the nouns in the AMR graphs are complicated, and there are obvious overlaps between several nodes. We could see that the text generated by SPRING fails to reconstruct the word "Japan delegation", misrepresents the relation between *head* and *Tokyo Handicapped Integrated Sports Center*, and misses the concept of *director*. However, BiBL could correctly generate all the concepts involved in the complex graph and maintain the correct relations. This proves that BiBL could deal with situations where graph information is rather complicated.

TEXT

To me that just sounds like you being overly anxious, as opposed to having really bad OCD.

GOLD AMR

```
(s / sound-01
    [OMIT]
    :ARG2 (a / anxious
        [OMIT]
        :ARG1-of (i2 / instead-of-91
            :ARG2 (h / have-03
                :ARG0 y
                :ARG1 (d / disease
                        :name (n / name :op1 "OCD")
                        [OMIT])))))
    [OMIT])
```

SPRING AMR

```
(z0 / sound-01
    [OMIT]
    :ARG2 (z2 / anxious
        [OMIT]
        :ARG1-of (z5 / instead-of-91
            :ARG2 (z6 / have-03
                :ARG0 z3
                :ARG1 (z7 / medical-condition
                        :name (z8 / name :op1 "OCD")
                        [OMIT])))))
    [OMIT])
```

BiBL AMR

```
(z0 / sound-01
    [OMIT]
    :ARG2 (z2 / anxious
        [OMIT]
        :ARG1-of (z5 / instead-of-91
            :ARG2 (z6 / have-03
                :ARG0 z3
                :ARG1 (z7 / disease
                        :name (z8 / name :op1 "OCD")
                        [OMIT])))))
    [OMIT])
```

Figure 3: Text-to-AMR parsing case study on NER. The input text sentence is italicized. Wikification labels are omitted on gold, SPRING, and BiBL generated AMR graphs. The [OMIT] sign represents the omission of other graph nodes that are the same across gold, SPRING, and BiBL generated AMR graphs. The underlined parts highlight the label generated for the "OCD" named entity.

TEXT

TMT: I don't understand your point.

GOLD AMR

```
(s / say-01
    :ARG0 (i / i)
    :ARG1 (u / understand-01 :polarity -
            :ARG0 i
            :ARG1 (p2 / point-04
                    :ARG0 p))
    :ARG2 (p / person :wiki -
            :name (t / name
                    :op1 "TMT")))
```

SPRING AMR

```
(z0 / say-01
    :ARG1 (z1 / understand-01 :polarity -
            :ARG0 (z2 / i)
            :ARG1 (z3 / point-04
                    :ARG0 (z4 / you)))
    :ARG2 (z5 / person :wiki -
            :name (z6 / name
                    :op1 "TMT")))
```

BiBL AMR

```
(z0 / say-01
    :ARG1 (z1 / understand-01 :polarity -
            :ARG0 (z2 / i)
            :ARG1 (z3 / point-04
                    :ARG0 (z4 / person :wiki -
                            :name (z5 / name
                                    :op1 "TMT"))))
    :ARG2 z4)
```

Figure 4: Text-to-AMR parsing case study on Reentrancies. The input text sentence is italicized. The underline parts highlight the occurrence of a graph reentrancy.

AMR

```
(s / sight-01
    :ARG1 (h / hopeful-03
        :ARG0 (t / they)
        :ARG1 (e / enter-01
            :ARG0 t
            :ARG1 (h2 / heat)))
    :time (a / already)
    :condition (a2 / accident :polarity -))
```

GOLD TEXT

If accidents do not occur, their hopes of entering the heats are already insight.

SPRING TEXT

If there is no accident, there is already a sight of them hoping to enter the heat.

BiBL TEXT

Without the accident, their hopes of entering the heat were already in sight.

Figure 5: AMR-to-Text generation case study on semantic understanding. The text sentence output is italicized. The underline parts highlight the main differences between gold, SPRING, and BiBL generated sentences.

AMR

```
(h / hand-over-02
    :ARG0 (p3 / person :wiki -
        :name (n / name :op1 "Souya"))
    :ARG1 (f2 / flag
        :poss (d / delegation
            :source (c2 / country :wiki "Japan"
                :name (n2 / name :op1 "Japan")))
        :mod (d2 / delegation))
    :ARG2 (p / person :wiki -
        :name (n3 / name :op1 "Banminyan")
        :ARG0-of (h2 / head-01
            :ARG1 d)
        :ARG0-of (h3 / have-org-role-91
            :ARG1 (s2 / sports-facility :wiki -
                :name (n4 / name :op1 "Tokyo"
                    :op2 "Handicapped"
                    :op3 "Integrated"
                    :op4 "Sports" :op5 "Center"))
            :ARG2 (d3 / director))))
```

GOLD TEXT

Souya handed over the delegation flag of the Japanese delegation to the head of the delegation and Director of the Tokyo Handicapped Integrated Sports Center, Banminyan .

SPRING TEXT

Souya handed over the delegation flag of Japan to the head of the Tokyo Handicapped Integrated Sports Center Banminyan .

BiBL TEXT

Souya handed over the delegation flag of the Japanese delegation to Banminyan, head of the delegation and director of the Tokyo Handicapped Integrated Sports Center .

Figure 6: AMR-to-Text generation case study on complex graph structures. The text sentence output is italicized. The underline parts highlight the main differences between gold, SPRING, and BiBL generated sentences.

Multi-Layer Pseudo-Siamese Biaffine Model for Dependency Parsing

Ziyao Xu[1], **Houfeng Wang**[1], **Bingdong Wang**[2]
[1]MOE Key Lab of Computational Linguistics, Peking University
[2]Beijing Huilan Technology Co., Ltd.
{xzyxzy,wanghf}@pku.edu.cn, wangbd@huilan.com

Abstract

Biaffine method is a strong and efficient method for graph-based dependency parsing. However, previous work only used the biaffine method at the end of the dependency parser as a scorer, and its application in multi-layer form is ignored. In this paper, we propose a multi-layer pseudo-Siamese biaffine model for neural dependency parsing. In this model, we modify the biaffine method so that it can be utilized in multi-layer form, and use pseudo-Siamese biaffine module to construct arc weight matrix for final prediction. In our proposed multi-layer architecture, the biaffine method plays important roles in both scorer and attention mechanism at the same time in each layer. We evaluate our model on PTB, CTB, and UD. The model achieves state-of-the-art results on these datasets. Further experiments show the benefits of introducing multi-layer form and pseudo-Siamese module into the biaffine method with low efficiency loss.

1 Introduction

Dependency parsing is a fundamental task in NLP. Given a input sequence $\boldsymbol{s} = w_0 w_1 ... w_n$, the output is a dependency tree $\boldsymbol{t} = \{(h, d, l), 0 \leq h \leq n, 1 \leq d \leq n, l \in L\}$, where $w_i (1 \leq i \leq n)$ is a word, w_0 is a pseudo-word as root and (h, d, l) is an arc from w_h to w_d with label l in a relation set L. Due to the simplicity and effectiveness of representing syntactic information by using the tree structure, many works involve dependency parsing, such as syntax-enhanced pre-trained model (Xu et al., 2021).

There are two approaches to dependency parsing: transition-based and graph-based methods. Graph-based dependency parsing scores the components of a sentence and selects the highest scoring tree. Dozat and Manning (2017) for the first time introduce biaffine method into the dependency parsing. This strong and efficient biaffine first-order graph-based parser consists of BiLSTM encoder and biaffine scorer. After the biaffine parser is proposed, many works have focused on further improving the performance of this parser. Since the biaffine parser consists of two parts: BiLSTM encoder and biaffine scorer, there are two main directions of improvement focusing on the two parts respectively.

One direction of improvement is modifying the encoder. Straka (2018) introduce pre-trained model in the embedding stage. Na et al. (2019) fuse the hidden states of different layers of BiLSTM to construct the input of the biaffine scorer. Ji et al. (2019) add graph neural networks before the biaffine scorer to capture high-order information. Li et al. (2019) apply the self-attention mechanism as the replacement of BiLSTM encoder. Mrini et al. (2020) introduce the Label Attention Layer, a new form of self-attention where attention heads represent labels, into the encoder. In these works, although the encoder changes, the biaffine method remains as a scorer at the end of dependency parser.

Another direction is extending the biaffine scorer for second-order parser. Currently, there are relatively few works in this direction due to its difficulty. Zhang et al. (2020) introduce efficient TreeCRF. Wang and Tu (2020) introduce MFVI. Both works extend the biaffine scorer to the triaffine scorer for scoring second-order subtrees. The results show that introducing high-order information is beneficial to the dependency parser in many ways. In these works, the extended biaffine method still plays the role of a scorer for high-order modeling.

We observe that *biaffine method only plays the role of a scorer in almost all graph-based dependency parsers improved by the biaffine parser.* The ways and effects of making the method play more roles remains to be explored. This leads us to a new direction of improvement: *how to make biaffine method play more roles in dependency parser?* We notice that biaffine method is essentially an attention mechanism, so it can be utilized in a multi-

Proceedings of the 29th International Conference on Computational Linguistics, pages 5476–5487
October 12–17, 2022.

layer architecture like Transformers (Vaswani et al., 2017) to capture more information before the biaffine scorer. However, such an application of biaffine method is equivalent to adding Transformers which uses biaffine attention upon BiLSTM encoder and keeping the biaffine scorer at the end of the model unchanged, that is, the biaffine scorer is independent of the multi-layer architecture, so it cannot obtain more information, which limits the performance gains. This raises a question: *how to fuse the biaffine scorer into the multi-layer architecture, which allows the biaffine scorer to make use of more information?*

To address this question, we propose a multi-layer pseudo-Siamese biaffine model for graph-based dependency parsing. Our multi-layer architecture consists of several connectable layers with the same structure, and each layer contains two biaffine modules that are identical but have separate parameters, namely pseudo-Siamese module. One biaffine module is used as an attention mechanism to obtain the attention weight to construct the token representation for the next layer; the other is used as a scorer to obtain the arc weight matrix that contributes to final prediction. Therefore, in our proposed multi-layer architecture, the biaffine method plays the role of a scorer and an attention mechanism at the same time in each layer. The biaffine scorer can obtain information from each layer before, instead of only seeing the output of the last layer, so it can make use of more information captured by the multi-layer architecture.

We conduct experiments on PTB, CTB, and UD to verify the effectiveness of our model. Since the usage of pre-trained models like BERT (Devlin et al., 2019) and XLNet (Yang et al., 2019) in the embedding stage is common for dependency parsing, we evaluate our model with different pre-trained models. We conduct the experiments in three aspects: 1) compare our model with previous state-of-the-art dependency parsers using the same pre-trained model to show that our model achieves state-of-the-art performance; 2) implement the original biaffine model and compare it with our model when using the same pre-trained model to show that our modification based on biaffine model significantly improves the performance; 3) compare our multi-layer architecture with other multi-layer architectures for dependency parsing to justify the specific layers we propose.

We also conduct detailed analysis to illustrate four aspects relevant to our model: 1) the impact of the choice of number of layers and attention function on our model; 2) the importance of fusing the biaffine scorer into the multi-layer architecture; 3) the benefit of introducing multi-layer form and pseudo-Siamese module into the biaffine method in many ways, including the overall performance, the performance on short and long sentences respectively and the performance of label prediction; 4) the low efficiency loss of introducing multi-layer form and pseudo-Siamese module into the biaffine method.

In summary, the major contributions of our work are as follows:

- We introduce a multi-layer architecture using biaffine attention mechanism into the biaffine dependency parser, which makes the biaffine method plays more than a role of a scorer in graph-based dependency parsing.
- We introduce pseudo-Siamese module to fuse the biaffine scorer into the multi-layer architecture. We show that fusing the biaffine scorer into the multi-layer architecture is important for the performance gains.
- We conduct experiments and detailed analysis on PTB, CTB, and UD. The results show the benefits of introducing multi-layer form and pseudo-Siamese module into the biaffine method with low efficiency loss.

2 Model

Our graph-based dependency parser contains three parts, i.e., encoder, multi-layer biaffine model, and pseudo-Siamese module.

2.1 Encoder

Encoder consists of Embedding layer and BiLSTM layer. In Embedding layer, input token w_i with Part-of-speech tag p_i are used to construct input vector e_i:

$$\boldsymbol{e}_i = [emb(w_i); posemb(p_i)] \quad (1)$$

Where emb is word embedding, $posemb$ is learned Part-of-speech tag embedding. We use pre-trained model BERT or XLNet for word embedding. Following Straka et al. (2019), we use the linear combination of hidden states of the last four layers as the embedding, and a word embedding is the average of its subword embeddings. We project

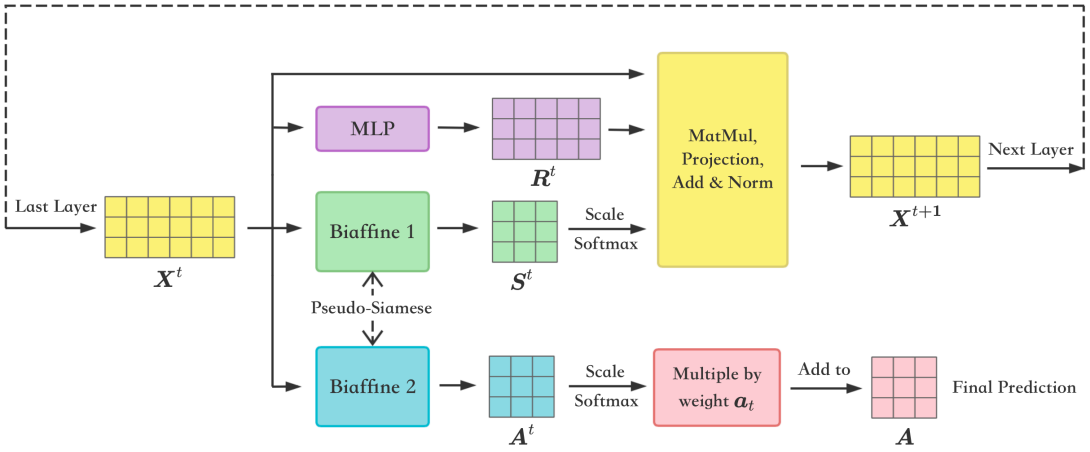

Figure 1: The architecture of the t-th layer biaffine model and its corresponding pseudo-Siamese module.

word embedding to a lower dimension. In BiLSTM layer, $\boldsymbol{e}_0\boldsymbol{e}_1...\boldsymbol{e}_n$ is input into a three-layer BiLSTM model. We arrange the output vectors of the last layer $\boldsymbol{h}_0, \boldsymbol{h}_1, ..., \boldsymbol{h}_n$ into the matrix $\boldsymbol{X}^1 \in \mathbb{R}^{n\times 2h}$, as the initial input of multi-layer biaffine model:

$$\boldsymbol{h}_i = BiLSTM_i(\boldsymbol{e}_0\boldsymbol{e}_1...\boldsymbol{e}_n) \quad (2)$$

$$\boldsymbol{X}^1 = \begin{bmatrix} \boldsymbol{h}_0 \\ \boldsymbol{h}_1 \\ ... \\ \boldsymbol{h}_n \end{bmatrix} \quad (3)$$

2.2 Multi-layer Biaffine Model

Multi-layer biaffine model consists of T layers with the same structure. The architecture of the t-th layer is shown in Figure 1. The input matrix of the t-th layer is $\boldsymbol{X}^t$. We use the same biaffine method described in Dozat and Manning (2017) :

$$\boldsymbol{H}^t = MLP^{(head)t}(\boldsymbol{X}^t) \quad (4)$$

$$\boldsymbol{D}^t = MLP^{(dep)t}(\boldsymbol{X}^t) \quad (5)$$

$$\boldsymbol{S}^t_{i,j} = \boldsymbol{H}^t_j \boldsymbol{U}^t (\boldsymbol{D}^t_i)^{\mathrm{T}} + \boldsymbol{H}^t_j \boldsymbol{b}^t \quad (6)$$

Where $MLP^{(head)t}$ and $MLP^{(dep)t}$ are $2h \times d$ (input dimension is $2h$ and output dimension is d; similarly hereinafter); $\boldsymbol{U}^t \in \mathbb{R}^{d\times d}$ and $\boldsymbol{b}^t \in \mathbb{R}^d$ are learned parameters. This biaffine module corresponds to Biaffine 1 in Figure 1.

We scale and apply softmax function (Vaswani et al., 2017) on $\boldsymbol{S}^t \in \mathbb{R}^{n\times n}$ to obtain attention weight matrix, and use it to construct the arc-related representation:

$$\boldsymbol{R}^t = MLP^{(arc)t}(\boldsymbol{X}^t) \quad (7)$$

$$\boldsymbol{V}^t = Softmax(\frac{\boldsymbol{S}^t}{\sqrt{2h}})\boldsymbol{R}^t \quad (8)$$

Where $MLP^{(arc)t}$ is $2h \times d$.

At the end of the layer, we apply projection and Add & Norm (Vaswani et al., 2017) to obtain the input matrix of the next layer:

$$\boldsymbol{X}^{t+1} = LayerNorm(\boldsymbol{X}^t + \boldsymbol{V}^t \boldsymbol{W}^t) \quad (9)$$

Where $\boldsymbol{W}^t \in \mathbb{R}^{d\times 2h}$ is learned matrix. $\boldsymbol{X}^{t+1} \in \mathbb{R}^{n\times 2h}$ is used as the input of $(t+1)$-th layer.

2.3 Pseudo-Siamese Module

We notice that attention weight matrix in each layer and arc weight matrix for final prediction have the same form. Based on this, we use pseudo-Siamese module to construct the arc weight matrix. In the t-th layer, we use another biaffine module to calculate matrix $\boldsymbol{A}^t \in \mathbb{R}^{n\times n}$. This biaffine module corresponds to Biaffine 2 in Figure 1. Biaffine 1 and Biaffine 2 have the same structure but different parameters. That is, $\boldsymbol{A}^t$ is calculated by Equations $4 \sim 6$ similar to $\boldsymbol{S}^t$ but using another set of parameters. We use the linear combination of $\boldsymbol{A}^t$ as the arc weight matrix $\boldsymbol{A}$ for final prediction:

$$\boldsymbol{A} = \sum_{t=1}^{T} \boldsymbol{a}_t \cdot Softmax(\frac{\boldsymbol{A}^t}{\sqrt{2h}}) \quad (10)$$

Where $\boldsymbol{a} \in \mathbb{R}^T$ is learned weight satisfying that $\sum_{t=1}^{T} \boldsymbol{a}_t = 1$.

2.4 Inference

We use $\boldsymbol{A}$ calculated by Equation 10 as the arc weight matrix and apply the Eisner algorithm (Eisner, 2000) or the Chu-Liu-Edmonds algorithm (Chu and Liu, 1965; Edmonds et al., 1967) to obtain the maximum spanning tree. After obtaining the tree structure, following Dozat and Manning

(2017), we use a biaffine classifier to obtain the label score vector for each word w_i given the predicted head h_i:

$$\boldsymbol{B}_i = \boldsymbol{H}_{h_i}\boldsymbol{U}^{(1)}(\boldsymbol{D}_i)^{\mathrm{T}} + [\boldsymbol{H}_{h_i}; \boldsymbol{D}_i]\boldsymbol{U}^{(2)} + \boldsymbol{b}$$
$$\boldsymbol{L}_i = Softmax(\boldsymbol{B}_i) \quad (11)$$

Where $\boldsymbol{U}^{(1)} \in \mathbb{R}^{d\times k\times d}$, $\boldsymbol{U}^{(2)} \in \mathbb{R}^{2d\times k}$ and $\boldsymbol{b} \in \mathbb{R}^k$ are learned parameters (k is the size of relation set). $\boldsymbol{H}$ and $\boldsymbol{D}$ are calculated by Equations 4 $\sim$ 5 using $\boldsymbol{X}^T$ as input. We select the label with the maximum score for each arc to obtain the final dependency tree.

2.5 Training

We calculate $\boldsymbol{A}$ by Equation 10, and calculate $\boldsymbol{L}$ by Equation 11 using the gold head. We use the cross-entropy loss for arc and label predictions:

$$\mathcal{L}^{(arc)} = -\sum_{i=1}^{n}\log(\boldsymbol{A}_{i,h_i}) \quad (12)$$

$$\mathcal{L}^{(label)} = -\sum_{i=1}^{n}\log(\boldsymbol{L}_{i,l_i}) \quad (13)$$

Where h_i is the gold head of w_i, and l_i is the gold label of arc (h_i, w_i). The final loss is:

$$\mathcal{L} = \lambda\mathcal{L}^{(arc)} + (1-\lambda)\mathcal{L}^{(label)} \quad (14)$$

Where λ is a hyper-parameter between 0 and 1.

3 Experiments

3.1 Datasets

We evaluate our method on PTB 3.0 (Marcus et al., 1993), CTB 5.1 (Xue et al., 2005), and Universal Dependencies (UD) 2.2. Following Chen and Manning (2014), we use Stanford parser v3.3.0 to convert PTB, and use Penn2Malt tool with the head-finding rules of Zhang and Clark (2008) to convert CTB. Following Fernández-González and Gómez-Rodríguez (2021), we do not use POS tags on PTB, and use gold POS tags on CTB. Following Ma et al. (2018), we evaluate 12 languages selected from UD 2.2 and use POS tags.

3.2 Evaluation

We use UAS and LAS as the metric. During the evaluation, we ignore all punctuation. We use the model after the last epoch of training for evaluation. For all results reported, we run the training process five times with different random seeds and average the results to avoid contingency.

3.3 Implementation Details

We evaluate our model with different pre-trained models, including BERT-uncased and XLNet for PTB, BERT-Chinese for CTB, and BERT-Multilingual-cased for UD. The dimension of word embedding after projection is 300, and the dimension of POS embedding is 50. We set $h = 512$, $\lambda = 0.55$, $d = 512$ for arc and $d = 128$ for label. We set $T = 5$ on PTB and UD, and $T = 6$ on CTB. We apply dropout after embedding and BiLSTM layers with dropout rate 0.33. We apply gradient clipping with max 2-norm value 1. We use Adam (Kingma and Ba, 2015) optimizer with $\beta_1 = 0.9$, $\beta_2 = 0.999$. The learning rate is $1e-5$ for pre-trained model and $5e-4$ for other components. We train the model for 8 epochs on PTB and UD, and 20 epochs on CTB. We decay the learning rate linearly to 0 during training. We batch the sentences of similar length for efficiency. The batch size is 24. During training, we divide the loss by batch size on CTB and UD but not on PTB. During inference, we use the Eisner algorithm on PTB and CTB, and we use the Chu-Liu-Edmonds algorithm on UD. More discussions on inference algorithms are presented in Section 4.6.

3.4 Baselines

We use nine strong baseline models for comparison and divide them into five categories. All results of baseline models are from the corresponding papers.

Biaffine. Dozat and Manning (2017) introduces the biaffine method for first-order graph-based dependency parsing.

Second-order. Zhang et al. (2020) introduces TreeCRF and Wang and Tu (2020) introduces MFVI. These parsers extend the biaffine method to triaffine for modeling second-order information.

HPSG. Zhou and Zhao (2019) introduces head-driven phrase structure grammar (HPSG) for joint dependency and constituent parsing. Mrini et al. (2020) uses HPSG and introduces label attention layer. These parsers use additional constituency information for training.

G2GTr. Mohammadshahi and Henderson (2020) and Mohammadshahi and Henderson (2021) introduces Graph-to-Graph Transformer (G2GTr) for transition-based dependency parsing and graph-based dependency parsing respectively.

Pointer Networks. Ma et al. (2018) introduces stack-pointer networks and Fernández-González and Gómez-Rodríguez (2021) introduces bottom-

Pre-trained	Model	PTB	
		UAS	LAS
w/o	Doz. & Man. (2017)	95.74	94.08
	Zhang et al. (2020)	96.14	94.49
BERT-base	Moh. & Hen. (2020)	96.11	94.33
	Moh. & Hen. (2021)	96.66	95.01
	Ours(BERT-base)	**96.93**	**95.18**
BERT-large	Wang & Tu (2020)	96.91	95.34
	Fer. & Góm. (2021)	97.05	95.48
	Ours(BERT-large)	**97.17**	**95.50**
XLNet-base	Ours(XLNet-base)	97.17	95.49
XLNet-large	Zhou & Zhao (2019)†	97.20	95.72
	Mrini et al. (2020)†	97.42	**96.26**
	Ours(XLNet-large)	**97.44**	95.81

Pre-trained	Model	CTB	
		UAS	LAS
w/o	Doz. & Man. (2017)	89.30	88.23
	Ma et al. (2018)	90.59	89.29
BERT-base	Mrini et al. (2020)†	**94.56**	89.28
	Wang & Tu (2020)	92.78	91.69
	Fer. & Góm. (2021)	92.75	91.62
	Moh. & Hen. (2021)	92.98	91.18
	Ours(BERT-base)	93.37	**92.16**

Pre-trained	Model	UD2.2	
		UAS	LAS
w/o	Ma et al. (2018)	93.53	89.75
	Zhang et al. (2020)	-	89.33
BERT-base	Wang & Tu (2020)	-	91.02
	Ours(BERT-base)	94.68	**91.82**

Table 1: Comparison of dependency parsers on PTB, CTB, and UD2.2. **Pre-trained** column indicates pre-trained model used for word embedding. †:These approaches join the constituency parsing and use additional constituency information for training.

up hierarchical pointer networks for transition-based dependency parsing.

3.5 Main Results

Table 1 shows the results of baselines and our model on PTB and CTB. For intuitive comparison, we divide the models according to the pre-trained model used for word embedding. Overall, the ranking of performance of our model with different pre-trained models is XLNet-large > XLNet-base ≈ BERT-large > BERT-base.

On PTB, the previous state-of-the-art model with BERT-base is Mohammadshahi and Henderson (2021) using Graph-to-Graph Transformer for graph-based parsing, which outperforms Mohammadshahi and Henderson (2020) for transition-based parsing. Compared with it, our model with BERT-base improves 0.27 UAS and 0.17 LAS. The previous state-of-the-art model with BERT-large is Fernández-González and Gómez-Rodríguez (2021) using bottom-up hierarchical pointer networks for transition-based parsing, which outperforms second-order graph-based parser Wang and Tu (2020) using MFVI. Compared with it, our model with BERT-large improves 0.12 UAS and performs similarly on LAS. Our model with XLNet-base also outperforms previous models with BERT-large. Our model with XLNet-large achieves 97.44 UAS and 95.81 LAS, which is the state-of-the-art result among dependency parsers without additional constituency information for training. For previous models with XLNet-large, Zhou and Zhao (2019) and Mrini et al. (2020) both use HPSG, which join the constituency parsing and use additional constituency information for training. Our model with XLNet-large outperforms Zhou and Zhao (2019), and has a comparable performance on UAS compared with Mrini et al. (2020).

On CTB, we only evalutate our model with BERT-base-Chinese because it is the only pre-trained model used for CTB in baseline models. Our model with BERT-base-Chinese achieves 93.37 UAS and 92.16 LAS, which is the state-of-the-art result among dependency parsers without additional constituency information for training. Compared with the previous state-of-the-art model on UAS which does not join the constituency parsing (Mohammadshahi and Henderson, 2021), our model improves 0.39 UAS. Compared with the previous state-of-the-art model on LAS (Wang and Tu, 2020), our model improves 0.47 LAS. Compared with Mrini et al. (2020) using additional constituency information for training, our model has a significant performance advantage on LAS.

On UD2.2, our model with BERT-Multilingual-cased achieves 94.68 average UAS and 91.82 average LAS. Compared with the previous state-of-the-art model (Wang and Tu, 2020) using MFVI, our model improves 0.8 average LAS.

3.6 Comparison with Original Single-layer Biaffine Model

We compare our model with the original biaffine model when using the same pre-trained model. When our model has only one layer, it degenerates to the original biaffine model, so we simply set $T = 1$ and keep other hyperparameters unchanged to implement the original biaffine model with different pre-trained models. The results on PTB and CTB are shown in Table 2. It can be seen that our model with each kind of pre-trained model performs significantly better than the original single-

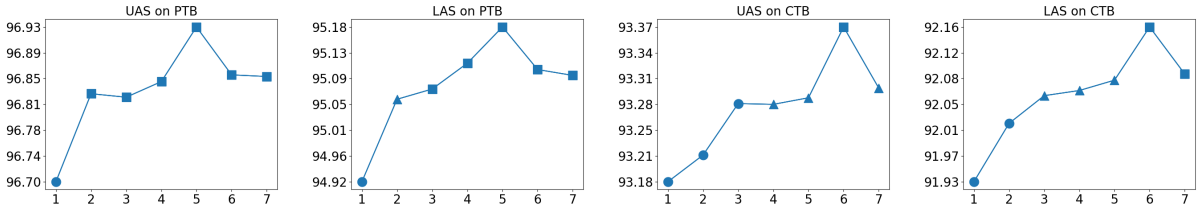

Figure 2: Results of our model with T layers on PTB and CTB ($T = 1$ means the original biaffine model). For $T > 1$, we perform significance test against $T = 1$. Triangle point means $p < 0.05$ and square point means $p < 0.005$.

Pre-trained	Model	PTB	
		UAS	LAS
BERT-base	Original	96.70	94.92
	Ours	**96.93**‡	**95.18**‡
BERT-large	Original	97.00	95.33
	Ours	**97.17**‡	**95.50**‡
XLNet-base	Original	97.01	95.26
	Ours	**97.17**‡	**95.49**‡
XLNet-large	Original	97.28	95.59
	Ours	**97.44**‡	**95.81**‡

Pre-trained	Model	CTB	
		UAS	LAS
BERT-base	Original	93.18	91.93
	Ours	**93.37**‡	**92.16**‡

Table 2: Comparison of original biaffine model and our model with different pre-trained models on PTB and CTB. We perform significance test for each pair of results. ‡ means $p < 0.005$.

layer biaffine model with the same pre-trained model. Compared with the original single-layer biaffine model, our model improves 0.18 UAS, 0.22 LAS on average on PTB, and improves 0.19 UAS, 0.23 LAS on CTB. The results on UD2.2 are shown in Table 3. It can be seen that our model outperforms the original single-layer biaffine model on 12 languages and improves 0.18 average UAS, 0.11 average LAS. The results show that our modification based on the original biaffine model significantly improves the performance. As a supplement, the results without gold POS tags on CTB are shown in Appendix A.

3.7 Comparison with Other Multi-layer Architectures

We compare our multi-layer architecture with other multi-layer architectures to justify the specific layers we propose. We select two strong multi-layer architectures which use BERT-base pre-trained model and have been evaluated on PTB or CTB for comparison: (1) Self-attentive parser proposed by Li et al. (2019). This architecture uses standard multi-head self-attention mechanism in multi-layer form. (2) Syntactic Transformer proposed by Mohammadshahi and Henderson (2021). This architecture uses the same architecture as BERT (Devlin et al., 2019) but changes the functions used by each attention head. For a fair comparison, we remove BiLSTM from our model and use BERT-base pre-trained model, so that the only difference between the models is the multi-layer architecture. The results are shown in Table 4. It can be seen that our multi-layer architecture achieves better results with fewer layers compared with other multi-layer architectures on PTB and CTB, which justifies the specific layers we propose.

4 Analysis

4.1 Number of Layers

We evaluate our model with number of layers $T \in \{1, 2, 3, 4, 5, 6, 7\}$ ($T = 1$ means the original biaffine model). The pre-trained model is BERT-base for PTB and BERT-base-Chinese for CTB, and we use this setting in the whole section 4 unless otherwise specified. The results are shown in Figure 2. It can be seen that: (1) Our model with any $T > 1$ outperforms the original single-layer biaffine model on both PTB and CTB. (2) With the increase in the number of layers, the performance of our model gradually improves and starts to be significantly different from the original single-layer model at $T = 2$ on PTB and $T = 3$ on CTB. (3) From $T = 2$ on PTB and $T = 3$ on CTB, the performance of our model improves with a similar trend on both datasets. This continuous improvement is due to more useful information captured by more layers. (4) After three more layers on both datasets, that is, at $T = 5$ on PTB and $T = 6$ on CTB, our model achieves the optimal performance.

Model	UD2.2 (UAS)												
	bg	ca	cs	de	en	es	fr	it	nl	no	ro	ru	Avg.
Original	95.71	95.49	95.28	90.49	93.05	94.77	94.28	96.16	94.68	95.93	93.13	95.52	94.54
Ours	**95.78**	**95.69**‡	**95.31**†	**90.61**	**93.27**†	**94.83**	**94.42**†	**96.33**	**94.84**	**96.05**†	**93.36**	**95.65**‡	**94.68**‡

Model	UD2.2 (LAS)												
	bg	ca	cs	de	en	es	fr	it	nl	no	ro	ru	Avg.
Original	91.87	93.77	92.30	85.89	90.90	92.69	91.34	94.43	91.88	94.46	86.95	94.03	91.71
Ours	**92.00**†	**94.00**‡	**92.34**†	**85.97**	**91.03**	**92.72**	**91.45**	**94.61**	**92.05**	**94.56**	**87.00**	**94.18**‡	**91.82**‡

Table 3: Comparison of original biaffine model and our model with BERT-base-Multilingual-cased on UD2.2. We perform significance test for each pair of results. † means $p < 0.05$ and ‡ means $p < 0.005$.

Multi-layer Architecture	PTB	
	UAS	LAS
SelfAtt (8 Layers)	96.67	95.03
SynTr (12 Layers)	96.60	94.94
Ours (5 Layers)	**96.75**	**95.05**

Multi-layer Architecture	CTB	
	UAS	LAS
SynTr (12 Layers)	92.42	90.67
Ours (6 Layers)	**92.91**	**91.80**

Table 4: Comparison of different multi-layer architectures for dependency parsing on PTB and CTB. SelfAtt: Li et al. (2019); SynTr: Mohammadshahi and Henderson (2021).

Function	PTB		CTB	
	UAS	LAS	UAS	LAS
Biaffine	**96.93**	**95.18**	**93.37**	**92.16**
Dot	96.87†	95.14	93.32	92.11
General	96.89	95.14	93.35	92.12
Concat	96.87†	95.09‡	93.30	92.08

Table 5: Results of our model with four attention functions used in Equation 6 on PTB and CTB. We perform significance test against Biaffine function for another three functions. † means $p < 0.05$ and ‡ means $p < 0.005$.

Our optimal model improves 0.23 UAS, 0.26 LAS on PTB and 0.19 UAS, 0.23 LAS on CTB compared with the original single-layer biaffine model. (5) The performance of our model starts to decline after $T = 5$ on PTB and $T = 6$ on CTB. This indicates that the use of more layers helps the model to capture more information, but also makes the model more prone to overfitting the training data, which is consistent with previous works related to multi-layer architecture for dependency parsing (Li et al., 2019; Mrini et al., 2020).

4.2 Attention Function

We compare the biaffine attention function used in Equation 6 with another three common attention functions described in Luong et al. (2015):

$$\boldsymbol{S}^t_{i,j} = \begin{cases} \boldsymbol{H}^t_j(\boldsymbol{D}^t_i)^{\mathrm{T}} & \text{Dot} \\ \boldsymbol{H}^t_j\boldsymbol{U}^t(\boldsymbol{D}^t_i)^{\mathrm{T}} & \text{General} \\ \tanh([\boldsymbol{D}^t_i;\boldsymbol{H}^t_j]\boldsymbol{P}^t)\boldsymbol{p}^t & \text{Concat} \end{cases} \quad (15)$$

Where $\boldsymbol{U}^t \in \mathbb{R}^{d\times d}$, $\boldsymbol{P}^t \in \mathbb{R}^{2d\times d}$ and $\boldsymbol{p}^t \in \mathbb{R}^d$ are learned parameters. The results are shown in Table 5. It can be seen that biaffine attention function outperforms another three attention functions in our multi-layer architecture, although the difference is not significant in some cases. Our model with any of another three attention functions still significantly outperforms the original single-layer biaffine model. The results show that our model can achieve high performance using any common attention function, and biaffine attention function is the best choice for our model.

4.3 Effect of Pseudo-Siamese Module

In order to fuse the biaffine scorer into the multi-layer architecture, we use pseudo-Siamese module in our model to construct the arc weight matrix for final prediction. To verify the effect of pseudo-Siamese module, we compare it with two other construction methods: (1) True-Siamese. Two biaffine modules in each layer share the parameters, which means $\boldsymbol{A}^t = \boldsymbol{S}^t$. (2) Non-Siamese. The biaffine module for prediction is removed, and the final prediction is based on $\boldsymbol{S}^T$. The results are shown in Table 6. It can be seen that either replacing pseudo-Siamese module with true-Siamese module or removing pseudo-Siamese module, will lead to a significant decrease in the performance of our model. In particular, removing pseudo-Siamese module, which means not fusing the biaffine scorer into the multi-layer architecture, will make the performance of our model not significantly different

Method	PTB		CTB	
	UAS	LAS	UAS	LAS
P-S	**96.93**	**95.18**	**93.37**	**92.16**
T-S	96.81‡	95.06‡	93.20†	91.99†
N-S	96.72‡	94.91‡	93.19†	91.98†

Table 6: Results of our model with three methods of constructing arc weight matrix for final prediction on PTB and CTB. P-S: pseudo-Siamese; T-S: true-Siamese; N-S: non-Siamese. For T-S and N-S, we perform significance test against P-S. † means $p < 0.01$ and ‡ means $p < 0.005$.

Model	PTB			
	UAS		LAS	
	$L < 24$	$L \geq 24$	$L < 24$	$L \geq 24$
Original	96.89	96.61	95.17	94.80
Ours	**97.14**	**96.83**	**95.40**	**95.07**

Model	CTB			
	UAS		LAS	
	$L < 28$	$L \geq 28$	$L < 28$	$L \geq 28$
Original	94.69	92.49	93.43	91.26
Ours	**94.72**	**92.75**	**93.44**	**91.57**

Table 7: Results of original biaffine model and our model on sentences shorter or longer than average length on PTB and CTB.

from the original single-layer biaffine model on both PTB and CTB. The results show that using pseudo-Siamese module to fuse the biaffine scorer into the multi-layer architecture is important for the performance gains of our model.

4.4 Sentence Length

We evaluate our model and original biaffine model on sentences shorter or longer than average length on PTB and CTB. The average length is 23.85 words on PTB and 27.22 words on CTB. The results are shown in Table 7. It can be seen that our model outperforms the original single-layer biaffine model on both short and long sentences on PTB and CTB.

4.5 Label Prediction

We evaluate the accuracy of label prediction of our model and the original biaffine model when the gold head is provided on CTB and PTB. Our model achieves 97.79% on PTB and 98.30% on CTB. The original biaffine model achieves 97.76% on PTB and 98.26% on CTB. It can be seen that when the influence of performance difference in head prediction is excluded, our model still performs better in label prediction than the original single-layer biaffine model.

Pre-trained	Algorithm	PTB	
		UAS	LAS
BERT-base	C-L-E	96.88	95.13
	Eisner	**96.93**	**95.18**
BERT-large	C-L-E	97.13	95.48
	Eisner	**97.17**	**95.50**
XLNet-base	C-L-E	97.13	95.46
	Eisner	**97.17**	**95.49**
XLNet-large	C-L-E	97.41	95.79
	Eisner	**97.44**	**95.81**

Pre-trained	Model	CTB	
		UAS	LAS
BERT-base	C-L-E	93.28	92.07
	Eisner	**93.37**	**92.16**

Table 8: Results of our model using two inference algorithms with different pre-trained models on PTB and CTB. C-L-E: the Chu-Liu-Edmonds algorithm.

4.6 Inference Algorithms

On PTB and CTB, we use the Eisner algorithm for inference. The time complexity of the Eisner algorithm is $O(n^3)$. Based on the code of Kiperwasser and Goldberg (2016), we implement the Eisner algorithm with Pytorch, which can obtain projective maximum spanning trees on the entire PTB test set in about 5 seconds on a single TITAN RTX GPU. We compare the performance of the Eisner algorithm and the Chu-Liu-Edmonds algorithm on PTB and CTB. The results are shown in Table 8. It can be seen that the Eisner algorithm slightly outperforms the Chu-Liu-Edmonds algorithm on PTB and CTB, because trees in PTB (99.9% projective) and CTB (100% projective) are almost all projective, and the Eisner algorithm guarantees that the obtained tree is projective. There are many non-projective trees in UD, which the Eisner algorithm cannot handle, so we use the Chu-Liu-Edmonds algorithm with time complexity $O(n^3)$ on UD for inference.

4.7 Efficiency Loss

We evaluate the parameter size and speed of the original biaffine model and our model with 5 layers. Our speed evaluation is performed on a single TITAN RTX GPU. Excluding the same encoder

part, the original biaffine model has 2.34M parameters, and our model has 17.04M parameters. To run one epoch of training with batch size 24 on the entire PTB training set, the original biaffine model takes 10.48 minutes, and our model takes 11.07 minutes. Compared with the original biaffine model, our model uses 5.63% more time on training. For inference, the scoring process described in Section 2.2 and 2.3 is the only process in which there is an efficiency difference between the two models. When only considering this process, to parse the whole PTB test set, the original biaffine model uses 0.06 seconds, and our model uses 0.51 seconds. Our model uses 0.45 seconds more in this process, which has little effect on the entire parsing time (17.5 seconds). The relatively low increase in parameter size and time consumption on both training and parsing indicates the low efficiency loss of introducing multi-layer form and pseudo-Siamese module into the biaffine method.

5 Related Work

Transition-based method (Nivre, 2003; Yamada and Matsumoto, 2003) and graph-based method (McDonald et al., 2005; McDonald and Pereira, 2006) are two main approaches to dependency parsing. Before the deep learning era, almost all dependency parsers classify based on many sparse indicator features. Titov and Henderson (2007) for the first time introduce neural networks into transition-based dependency parsing. Chen and Manning (2014) propose a fast and accurate transition-based dependency parser using neural networks. Dyer et al. (2015) introduce BiLSTM into transition-based dependency parsing. Wang and Chang (2016) introduce BiLSTM encoder into the graph-based dependency parser. Kiperwasser and Goldberg (2016) also introduce BiLSTM encoder into dependency parsing and test it with both transition-based and graph-based methods. Based on these works, Dozat and Manning (2017) introduce biaffine method as a scorer into the graph-based dependency parsing and achieve significant performance improvement.

Multi-layer architecture is widely used to improve the performance of the original biaffine parser. Since the original biaffine parser uses multi-layer BiLSTM, Na et al. (2019) propose to use hidden states of different layers of BiLSTM to construct role-dependent representations for each layer, which are aggregated as the input of the biaffine scorer. Li et al. (2019) propose to use multi-layer self-attention at the encoder stage as the replacement of BiLSTM. Ji et al. (2019) propose to add multi-layer graph neural networks before the biaffine scorer and apply layer-wise loss to capture high-order information. Mohammadshahi and Henderson (2021) propose a recursive non-autoregressive graph-to-graph transformer for dependency parsing. They introduce a multi-layer architecture similar to BERT but changing the attention function. In these models, the biaffine method is used as a scorer after the multi-layer architecture. Differently, our multi-layer architecture uses the biaffine method in each layer as the role of an attention mechanism and a scorer at the same time.

Pseudo-Siamese network architecture, which means using two separate but identical networks to process a pair of inputs and fusing the information at a later stage, is widely used in the field of computer vision (Hughes et al., 2018; Treible et al., 2019). In our model, we use two separate but identical biaffine modules in each layer to process the same input into two branches, which is a reversed pseudo-Siamese network architecture.

6 Conclusions

In this paper, we propose a multi-layer biaffine pseudo-Siamese model for neural dependency parsing. In our proposed multi-layer architecture, biaffine method plays the role of a scorer and an attention mechanism at the same time in each layer. We conduct experiments and detailed analysis on PTB, CTB, and UD, showing the benefit of introducing multi-layer form and pseudo-Siamese module into the biaffine method. Compared with the original single-layer biaffine model, our model has a significant advantage in performance with low efficiency loss. Our analysis shows that fusing the biaffine scorer into the multi-layer architecture is important for the performance gains of our model. Compared with other multi-layer architectures for dependency parsing, our multi-layer architecture also has a performance advantage. Our model achieves state-of-the-art results on PTB, CTB, and UD. Our code is available at `https://github.com/xzy-xzy/MLPSB-Parser`.

Acknowledgements

The work is supported by National Natural Science Foundation of China under Grant No. 62036001 and PKU-Baidu Fund (No. 2020BD021). The corresponding author of this paper is Houfeng Wang.

References

Danqi Chen and Christopher Manning. 2014. A fast and accurate dependency parser using neural networks. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 740–750, Doha, Qatar. Association for Computational Linguistics.

Chu and Liu. 1965. On the shortest arborescence of a directed graph. *Scientia Sinica*, 14:1396–1400.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT 2019, Minneapolis, MN, USA, June 2-7, 2019, Volume 1 (Long and Short Papers)*, pages 4171–4186. Association for Computational Linguistics.

Timothy Dozat and Christopher D. Manning. 2017. Deep biaffine attention for neural dependency parsing. In *5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Conference Track Proceedings*. OpenReview.net.

Chris Dyer, Miguel Ballesteros, Wang Ling, Austin Matthews, and Noah A. Smith. 2015. Transition-based dependency parsing with stack long short-term memory. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing of the Asian Federation of Natural Language Processing, ACL 2015, July 26-31, 2015, Beijing, China, Volume 1: Long Papers*, pages 334–343. The Association for Computer Linguistics.

Jack Edmonds et al. 1967. Optimum branchings. *Journal of Research of the national Bureau of Standards B*, 71(4):233–240.

Jason Eisner. 2000. Bilexical grammars and their cubic-time parsing algorithms. In *Advances in probabilistic and other parsing technologies*, pages 29–61. Springer.

Daniel Fernández-González and Carlos Gómez-Rodríguez. 2021. Dependency parsing with bottom-up hierarchical pointer networks. *CoRR*, abs/2105.09611.

Lloyd H. Hughes, Michael Schmitt, Lichao Mou, Yuanyuan Wang, and Xiao Xiang Zhu. 2018. Identifying corresponding patches in SAR and optical images with a pseudo-siamese CNN. *IEEE Geosci. Remote. Sens. Lett.*, 15(5):784–788.

Tao Ji, Yuanbin Wu, and Man Lan. 2019. Graph-based dependency parsing with graph neural networks. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 2475–2485, Florence, Italy. Association for Computational Linguistics.

Diederik P. Kingma and Jimmy Ba. 2015. Adam: A method for stochastic optimization. In *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*.

Eliyahu Kiperwasser and Yoav Goldberg. 2016. Simple and accurate dependency parsing using bidirectional LSTM feature representations. *Transactions of the Association for Computational Linguistics*, 4:313–327.

Ying Li, Zhenghua Li, Min Zhang, Rui Wang, Sheng Li, and Luo Si. 2019. Self-attentive biaffine dependency parsing. In *Proceedings of the Twenty-Eighth International Joint Conference on Artificial Intelligence, IJCAI 2019, Macao, China, August 10-16, 2019*, pages 5067–5073. ijcai.org.

Thang Luong, Hieu Pham, and Christopher D. Manning. 2015. Effective approaches to attention-based neural machine translation. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 1412–1421, Lisbon, Portugal. Association for Computational Linguistics.

Xuezhe Ma, Zecong Hu, Jingzhou Liu, Nanyun Peng, Graham Neubig, and Eduard Hovy. 2018. Stack-pointer networks for dependency parsing. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1403–1414, Melbourne, Australia. Association for Computational Linguistics.

Mitchell P. Marcus, Beatrice Santorini, and Mary Ann Marcinkiewicz. 1993. Building a large annotated corpus of English: The Penn Treebank. *Computational Linguistics*, 19(2):313–330.

Ryan T. McDonald, Fernando Pereira, Kiril Ribarov, and Jan Hajic. 2005. Non-projective dependency parsing using spanning tree algorithms. In *HLT/EMNLP 2005, Human Language Technology Conference and Conference on Empirical Methods in Natural Language Processing, Proceedings of the Conference, 6-8 October 2005, Vancouver, British Columbia, Canada*, pages 523–530. The Association for Computational Linguistics.

Ryan T. McDonald and Fernando C. N. Pereira. 2006. Online learning of approximate dependency parsing algorithms. In *EACL 2006, 11st Conference of the European Chapter of the Association for Computational Linguistics, Proceedings of the Conference, April 3-7, 2006, Trento, Italy*. The Association for Computer Linguistics.

Alireza Mohammadshahi and James Henderson. 2020. Graph-to-graph transformer for transition-based dependency parsing. In *Findings of the Association for Computational Linguistics: EMNLP 2020, Online Event, 16-20 November 2020*, volume EMNLP 2020 of *Findings of ACL*, pages 3278–3289. Association for Computational Linguistics.

Alireza Mohammadshahi and James Henderson. 2021. Recursive non-autoregressive graph-to-graph transformer for dependency parsing with iterative refinement. *Trans. Assoc. Comput. Linguistics*, 9:120–138.

Khalil Mrini, Franck Dernoncourt, Quan Hung Tran, Trung Bui, Walter Chang, and Ndapa Nakashole. 2020. Rethinking self-attention: Towards interpretability in neural parsing. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 731–742, Online. Association for Computational Linguistics.

Seung-Hoon Na, Jinwoon Min, Kwanghyeon Park, Jong-Hun Shin, and Young-Kil Kim. 2019. JBNU at MRP 2019: Multi-level biaffine attention for semantic dependency parsing. In *Proceedings of the Shared Task on Cross-Framework Meaning Representation Parsing at the 2019 Conference on Natural Language Learning, CoNLL 2019, Hong Kong, November 3, 2019*, pages 95–103. Association for Computational Linguistics.

Joakim Nivre. 2003. An efficient algorithm for projective dependency parsing. In *Proceedings of the Eighth International Conference on Parsing Technologies, IWPT 2003, Nancy, France, April 2003*, pages 149–160.

Milan Straka. 2018. Udpipe 2.0 prototype at conll 2018 UD shared task. In *Proceedings of the CoNLL 2018 Shared Task: Multilingual Parsing from Raw Text to Universal Dependencies, Brussels, Belgium, October 31 - November 1, 2018*, pages 197–207. Association for Computational Linguistics.

Milan Straka, Jana Straková, and Jan Hajic. 2019. Evaluating contextualized embeddings on 54 languages in POS tagging, lemmatization and dependency parsing. *CoRR*, abs/1908.07448.

Ivan Titov and James Henderson. 2007. Fast and robust multilingual dependency parsing with a generative latent variable model. In *EMNLP-CoNLL 2007, Proceedings of the 2007 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning, June 28-30, 2007, Prague, Czech Republic*, pages 947–951. ACL.

Wayne Treible, Philip Saponaro, and Chandra Kambhamettu. 2019. Wildcat: In-the-wild color-and-thermal patch comparison with deep residual pseudo-siamese networks. In *2019 IEEE International Conference on Image Processing, ICIP 2019, Taipei, Taiwan, September 22-25, 2019*, pages 1307–1311. IEEE.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, December 4-9, 2017, Long Beach, CA, USA*, pages 5998–6008.

Wenhui Wang and Baobao Chang. 2016. Graph-based dependency parsing with bidirectional LSTM. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 2306–2315, Berlin, Germany. Association for Computational Linguistics.

Xinyu Wang and Kewei Tu. 2020. Second-order neural dependency parsing with message passing and end-to-end training. In *Proceedings of the 1st Conference of the Asia-Pacific Chapter of the Association for Computational Linguistics and the 10th International Joint Conference on Natural Language Processing*, pages 93–99, Suzhou, China. Association for Computational Linguistics.

Zenan Xu, Daya Guo, Duyu Tang, Qinliang Su, Linjun Shou, Ming Gong, Wanjun Zhong, Xiaojun Quan, Daxin Jiang, and Nan Duan. 2021. Syntax-enhanced pre-trained model. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 5412–5422, Online. Association for Computational Linguistics.

Naiwen Xue, Fei Xia, Fu-Dong Chiou, and Martha Palmer. 2005. The penn chinese treebank: Phrase structure annotation of a large corpus. *Nat. Lang. Eng.*, 11(2):207–238.

Hiroyasu Yamada and Yuji Matsumoto. 2003. Statistical dependency analysis with support vector machines. In *Proceedings of the Eighth International Conference on Parsing Technologies, IWPT 2003, Nancy, France, April 2003*, pages 195–206.

Zhilin Yang, Zihang Dai, Yiming Yang, Jaime G. Carbonell, Ruslan Salakhutdinov, and Quoc V. Le. 2019. Xlnet: Generalized autoregressive pretraining for language understanding. In *Advances in Neural Information Processing Systems 32: Annual Conference on Neural Information Processing Systems 2019, NeurIPS 2019, December 8-14, 2019, Vancouver, BC, Canada*, pages 5754–5764.

Yu Zhang, Zhenghua Li, and Min Zhang. 2020. Efficient second-order TreeCRF for neural dependency parsing. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 3295–3305, Online. Association for Computational Linguistics.

Yue Zhang and Stephen Clark. 2008. A tale of two parsers: Investigating and combining graph-based and transition-based dependency parsing. In *Proceedings of the 2008 Conference on Empirical Methods in Natural Language Processing*, pages 562–571, Honolulu, Hawaii. Association for Computational Linguistics.

Junru Zhou and Hai Zhao. 2019. Head-Driven Phrase Structure Grammar parsing on Penn Treebank. In

Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics, pages 2396–2408, Florence, Italy. Association for Computational Linguistics.

A Supplementary Results

Previous works generally use gold POS tags on CTB. To get a more realistic estimate of real performance, we evaluate our model and the original biaffine model without gold POS tags on CTB. Our model achieves 91.45 UAS, 89.31 LAS. The original biaffine model achieves 91.19 UAS, 89.07 LAS. Compared with the original biaffine model, our model improves 0.26 UAS, 0.24 LAS. The significance test shows that $p < 0.005$ on UAS and $p < 0.01$ on LAS. It can be seen that our model still outperforms the original biaffine model without gold POS tags on CTB.

Belief Revision based Caption Re-ranker with Visual Semantic Information

Ahmed Sabir[1], **Francesc Moreno-Noguer**[2], **Pranava Madhyastha**[3], **Lluís Padró**[1]
[1] Computer Science Department, Universitat Politècnica de Catalunya, Barcelona, Spain
[2] Institut de Robòtica i Informàtica Industrial, CSIC-UPC, Barcelona, Spain
[3] City, University of London, London, UK

Abstract

In this work, we focus on improving the captions generated by image-caption generation systems. We propose a novel re-ranking approach that leverages visual-semantic measures to identify the ideal caption that maximally captures the visual information in the image. Our re-ranker utilizes the Belief Revision framework (Blok et al., 2003) to calibrate the original likelihood of the top-n captions by explicitly exploiting the semantic relatedness between the depicted caption and the visual context. Our experiments demonstrate the utility of our approach, where we observe that our re-ranker can enhance the performance of a typical image-captioning system without the necessity of any additional training or fine-tuning.[1]

1 Introduction

Image caption generation is a task that predominantly lies at the intersection of the areas of computer vision and natural language processing. The task is primarily aimed at generating a natural language description for a given image. Caption generation systems usually consist of an image encoder that encodes a given image (usually by using a CNN) whose encoding is fed to a decoder (usually by using a generative model such as RNN) to generate a natural language sentence which describes the image succinctly. The most widely used approaches include a CNN-RNN end-to-end system (Vinyals et al., 2015; Anderson et al., 2018), end-to-end systems with attention that attend to specific regions of the image for generation (Xu et al., 2015; You et al., 2016) and systems with reinforcement learning based methods (Rennie et al., 2017; Ren et al., 2017). Furthermore, recent advances have resulted in end-to-end systems that use Transformer based architecture for language generation and have become the current state-of-the-art (Herdade et al., 2019; Huang et al., 2019; Cornia et al., 2020; Zhang et al., 2021b).

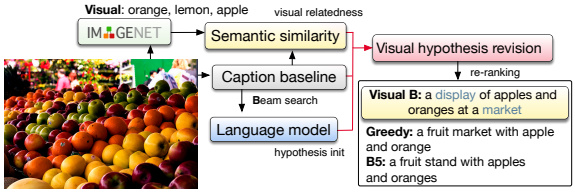

Figure 1: An overview of our hypothesis revision based visual re-ranker. We use the visual context from the image to *revise* and re-rank the most closely related caption to its visual context. These semantic relatedness measures are learned at the word-to-sentence level. In this example, we showcase our visual re-ranker (**V**isual **B**eam), a post-processing approach, which is able to re-rank the most 'descriptive caption' from the 5-Best **B**eam (Cornia et al., 2020).

While the state-of-the-art models generate captions that are comparable to human level captions, they are known to lack lexical diversity, are often not very distinct, and sound synthetic. We here highlight a few recent approaches that have focused on this problem, these include Dai et al. (2017) that uses generative adversarial networks towards generating diverse and human like captions. Vedantam et al. (2017) use a beam search with a distractor image to force the model to produce diverse captions by encouraging the models to be discriminative. Other recent works use a beam search directly to produce diverse captions by forcing richer lexical word choices (Ippolito et al., 2019; Vijayakumar et al., 2018; Wang and Chan, 2019; Wang et al., 2020). In this work, we follow a similar line of research and focus on the problem of improving diversity and making captions natural and human like and propose a novel re-ranking approach. In this approach, we use n-best reranking with a given beam that explicitly uses the semantic correlation between the caption and the visual context through belief revision (an approach inspired by human logic). We refer the reader to Figure 1, where the

[1] https://github.com/ahmedssabir/Belief-Revision-Score

Proceedings of the 29th International Conference on Computational Linguistics, pages 5488–5506
October 12–17, 2022.

approach results in a caption that is a) visually relevant and b) the most natural and human like.

Our primary contributions in this paper are:

- We demonstrate the utility of the Belief Revision (Blok et al., 2003) framework, which has been shown to correlate highly with human judgment and has demonstrate its applicability to the task of Image Captioning. We do this by employing vision-language joint semantic measures using state-of-the-art pre-trained language models.
- Our approach is a *post-processing* method and is devised to be a drop-in replacement for any caption system.
- Through our experiments, we report that our proposal selects better captions as reported using automated metrics, as well as being validated by human evaluations.

2 Belief Revision with `SimProb` Model

In this section, we briefly introduce `SimProb`, which is based on the philosophical intuitions of Belief Revision, an idea that helps to convert similarity measures to probability estimates. Blok et al. (2003) introduce a conditional probability model that assumes that the preliminary probability result is updated or revised to the degree that the hypothesis proof warrants. The range of revision is based on the informativeness of the argument and its degree of similarity. That is, the similarity to probability conversion can be defined in terms of **Belief Revision**. Belief Revision is a process of forming a belief by taking into account a new piece of information.

Let us consider the following statements:

1 `Tigers can bite through wire, therefore Jaguars can bite through wire.`
2 `Kittens can bite through wire, therefore Jaguars can bite through wire.`

In the first case, the statement seems logical because it matches our prior belief *i.e.* jaguars are similar to tigers, so we expect them to be able to do similar things. We hence consider that the statement is consistent with our previous belief, and there is no need to revise it. In the second case, the statement is surprising because our prior belief is that kittens are not as similar to jaguars, and thus, not so strong. But if we assume the veracity of the statement, then we need to revise and update our prior belief about the strength of kittens.

This work formalizes belief as probabilities and revised belief as conditional probabilities and provides a framework to compute them based on the similarities of the involved objects. According to the authors, belief revision should be proportional to the similarity of the involved objects (*i.e.* in the example, the statement about kittens and jaguars would cause a stronger belief revision than *e.g.* the same statement involving pigeons and jaguars because they are less similar). In our case, we use the same rationale and the same formulas to convert similarity (or relatedness) scores into probabilities suitable for reranking.

`SimProb` Model To obtain the likelihood revisions based on similarity scores, we need three parameters: (1) **Hypothesis**: prior probabilities, (2) **Informativeness**: conclusion events and (3) **Similarities**: measuring the relatedness between involved categories.The goal is to predict a conditional probability of statements, given one or more other statements. In order to predict the conditional probability of the argument's conclusion, given its premise or hypothesis, we will need only the prior probabilities of the statements, as well as the similarities between the involved categories (*e.g.* kittens and tigers).

Formulation of `SimProb` The conditional probability $\mathrm{P}(Q_c|Q_a)$ is expressed in terms of the prior probability of the conclusion statement $\mathrm{P}(Q_c)$, the prior probability of the premise statement $\mathrm{P}(Q_a)$, and the similarity between the conclusion and the premise categories sim(a, c).

$$\mathrm{P}\left(Q_{\mathrm{c}} \mid Q_a\right) = \mathrm{P}\left(Q_c\right)^{\alpha} \text{ where } \alpha = \left[\frac{1-\mathrm{sim}(a,c)}{1+\mathrm{sim}(a,c)}\right]^{1-\mathrm{P}(Q_a)}$$

Belief Revision Elements As we discussed above, there are two factors that determine the hypothesis probability revision: 1) the sufficient relatedness to the category: as $sim(a, c) \rightarrow 0$, $\alpha \rightarrow 1$, and thus $\mathrm{P}(Q_c|Q_a) = \mathrm{P}(Q_c)$, *i.e.* no revision takes place, as there are no changes in the original belief. While as $sim(a, c) \rightarrow 1$, $\alpha \rightarrow 0$, and the hypothesis probability $\mathrm{P}(Q_c)$ is revised and is raised closer to 1; 2) the informativeness of the new information $1 - \mathrm{P}(Q_a)$: as $\mathrm{P}(Q_a) \rightarrow 1$ and in consequence is less informative, $\alpha \rightarrow 1$, as there is no new information, and hence no revision is required.

3 Visual Re-ranking for Image Caption

3.1 Problem Formulation

The beam search is the dominant method for approximate decoding in structured prediction tasks such as machine translation, speech recognition

and image captioning. A larger beam size allows the model to perform a better exploration in the search space compared to greedy decoding. The main idea of the beam search is to explore all possible captions in the search space by keeping a set of *top candidates*.

Our goal is to leverage the visual context information of the image to re-rank the candidate sequences obtained through the beam search, thereby moving the most visually relevant candidate up in the list, as well as moving wrong candidates down. For this purpose, we experiment with different re-rankers, based on the relatedness between the candidate caption and the semantic context observed in the image through the idea of Belief Revision.

Caption Extraction We employ two recent Transformer based architectures for caption generation to extract the top candidate captions using different beam sizes ($B = 1 \dots 20$) (Vijayakumar et al., 2018). The first baseline is based on a multi-task model for discriminative Vision and Language BERT (Lu et al., 2020) that is fine-tuned on 12 downstream tasks. The second baseline is the vanilla Transformer (Vaswani et al., 2017) with the Meshed-Memory based caption generator (Cornia et al., 2020) with pre-computed top-down visual features (Anderson et al., 2018).

3.2 Proposal

One approach of using word-level semantic relations for scene text correction with the visual context of an image was introduced in Sabir et al. (2018), which allows for the establishment of learning semantic correlations between a visual context and a text fragment. In our work, this semantic relatedness is between a visual context and a given candidate caption (*i.e.* beam search), and uses Belief Revision (BR) via `SimProb` to re-visit and re-rank the original beam search based on the similarity to the *image objects/labels* c (a proxy for image context). The BR in this scenario is a conditional probability which assumes that the caption preliminary probability (*hypothesis*) $\mathrm{P}(w)$ is revised to the degree approved by the semantic similarity with visual context $\mathrm{sim}(w, c)$. The final output caption w for a given visual context c is written as:

$$\mathrm{P}(w \mid c) = \mathrm{P}(w)^{\alpha} \quad (1)$$

where the main components of visual based hypothesis revision:

Hypothesis: $\mathrm{P}(w)$

Informativeness: $1 - \mathrm{P}(c)$

Similarities: $\alpha = \left[\frac{1-\mathrm{sim}(w,c)}{1+\mathrm{sim}(w,c)}\right]^{1-\mathrm{P}(c)}$

where $\mathrm{P}(w)$ is the *hypothesis* probability (beam search candidate caption) and $\mathrm{P}(c)$ is the probability of the evidence that causes hypothesis probability revision (visual context from the image). We next discuss the details of each component in `SimProb` as visual based re-ranker.

Hypothesis: Prior probabilities of original belief. As this approach is inspired by humans, the hypothesis $\mathrm{P}(w)$ needs to be initialized by a common observation such as a Language Model (LM) trained on a general text corpus. Therefore, we employ a Generative Pre-trained Transformer (GPT-2) (Radford et al., 2019) a LM to initialize the hypothesis probability. We set $\mathrm{P}(w)$ as the mean of LM token probability.

Informativeness: Inversely related to the probability of set $\mathrm{P}(c)$ information that causes hypothesis revision. We leverage ResNet (He et al., 2016) and an Inception-ResNet v2 based Faster R-CNN object detector (Huang et al., 2017)[2] to extract textual visual context information from the image. We use the classifier probability confidence with a threshold to filter out non-existent objects in the image. For each image, we extract visual information as follows: (1) top-1 concept (2) multi concept top-3 (label class or object category) visual information. For the single concept, we employ a unigram LM, based on the 3M-token opensubtitles corpus (Lison and Tiedemann, 2016), to initialize the informativeness of the visual information. For multiple concepts, we take the mean probability of the three concepts. Note that we are initializing the single visual context with LM to maximize the visual context score while computing the informativeness.

Similarities: Hypothesis revision is more likely if there is a close relation between the hypothesis and the new information (candidate caption and visual context in our case). We rely on two of the most recent state-of-the-art pre-trained Transformer-based language models to compute the semantic similarity between the caption and its visual context information with contextual embedding. In particular, we utilize the visual as context for the sentence (*i.e.* caption) to compute the cosine distance:

- **BERT (Devlin et al., 2019):** BERT achieves remarkable results on many sentence level tasks and especially in the textual semantic

[2] TensorFlow Object Detection API

similarity task (STS-B) (Cer et al., 2017). Therefore, we fine-tuned $\text{BERT}_{\text{base}}$ on the training dataset, (textual information, 460k captions: 373k for training and 87k for validation) *i.e.* visual context, caption, label [semantically related or not related]), with a binary classification cross-entropy loss function [0,1] where the target is the semantic similarity between the visual and the candidate caption, with batch size 16 for two epochs with a learning rate $2\text{e}{-5}$.

- **RoBERTa (Liu et al., 2019):** RoBERTa is an improved version of BERT, trained on a large amount of data, using dynamic masking strategies to prevent overfitting. It achieves a 2.4% improvement over $\text{BERT}_{\text{Large}}$ in the STS task. Since $\text{RoBERTa}_{\text{Large}}$ is more robust, we use an off-the-shelf model tuned on STS-B task. In particular, we follow the traditional approach to compute the semantic similarity with a BERT based model with a mean pool, over the last hidden layer, to extract a meaningful vector to compute the cosine distance.

4 Experiments

4.1 Dataset

COCO-Caption (Lin et al., 2014): This dataset contains around 120k images and each image is annotated with five different human-written captions. We use the split provided by (Karpathy and Fei-Fei, 2015), where 5k images are used for testing, 5k for validation, and the rest for model training.

Visual Context Enrichment: We enrich COCO-Caption with textual visual context information. To automate visual context generation and without the need for a human label, for the training dataset, we use only ResNet152, which has 1000 label classes, to extract the top-k three label class visual context information for each image in the caption dataset. For testing, we rely only on the top-k visual information as a concept, and we also employ the Inception-ResNet v2 based Faster R-CNN object detector with 80 object classes. In particular, each single annotated caption has three visual context information.

Evaluation Metric: We use the official COCO offline evaluation suite, producing several widely used caption quality metrics: **B**LEU (Papineni et al., 2002) **M**ETEOR (Banerjee and Lavie, 2005), **R**OUGE (Lin, 2004), **C**IDEr (Vedantam et al., 2015), **S**PICE (Anderson et al., 2016) and BERTscore (Zhang et al., 2020).

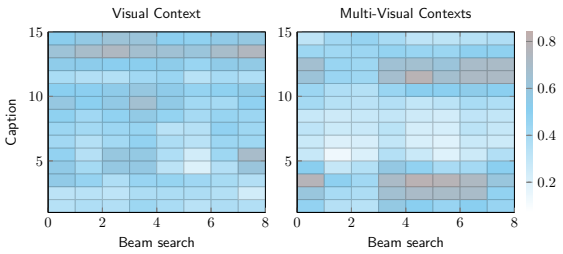

Figure 2: Visualization of top-k nine re-ranked Beam search via `SimProb` with, Vil+$\text{VR}_{\text{RoBERTa}}$ (Right) multiple visual and (Left) one concept visual context. The longer caption benefits from using multiple concepts.

4.2 Results and Discussion

We use visual semantic information to re-rank candidate captions produced by out-of-the-box state-of-the-art caption generators. We extract the top-20 beam search candidate captions from two state-of-the-art models: VilBERT (Lu et al., 2020), fine-tuned on a total of 12 different vision and language datasets such as caption image retrieval and visual question answering, and a specialized caption-based Transformer (Cornia et al., 2020).

Experiments applying different rerankers to the each base system are shown in Table 1. The tested rerankers are: (1) VR_{BERT} using BERT similarity between the candidate caption and the visual context of the image, transforming it to a probability using Equation 1, and combines the result with the original candidate probability to obtain the reranked score. (2) $\text{VR}_{\text{RoBERTa}}$ carrying out the same procedure using similarity produced by RoBERTa. A simpler model is also tested –VR_{BERT} (only sim) in Table 1–, which replaces Equation 1 with $\text{P}(w \mid c) = \text{sim}(w, c)^{\text{P}(c)}$, that is, it does not rely on the original caption probability.

First, we compare our work with the original visual caption re-ranker with multiple word objects as concepts from the image, that are extracted via Inception-ResNet v2 based Faster RCNN (*i.e.* person, van, *etc.*), $\text{VR}_{\text{w-Object}}$ (Fang et al., 2015). However, to make a fair comparison, we use the $\text{Sentence-RoBERTa}_{\text{Large}}$ for the sentence semantic similarity model *i.e.* cosine(word objects, caption). Secondly, we compare our model against two approaches that uses object information to improve image captioning: First, Wang et al. (2018) investigates the benefit of object frequency counts for generating a good captions. We train an LSTM

Model	B-1	B-4	M	R	C	S	BERTscore
VilBERT (Lu et al., 2020)							
$\text{Vil}_{\text{Greedy}}$	0.751	0.330	0.272	0.554	1.104	0.207	0.9352
$\text{Vil}_{\text{BeamS}}$	0.752	0.351	0.274	0.557	1.115	0.205	0.9363
$\text{Vil+VR}_{\text{W-Object}}$ (Fang et al., 2015)	0.756	0.348	0.274	0.559	1.123	0.206	0.9365
$\text{Vil+VR}_{\text{Object}}$ (Wang et al., 2018)	0.756	0.348	0.274	0.559	1.120	0.206	0.9364
$\text{Vil+VR}_{\text{Control}}$ (Cornia et al., 2019)	0.753	0.345	0.274	0.557	1.116	0.206	0.9361
$\text{Vil+VR}_{\text{BERT}}$ (only sim)	0.753	0.343	0.273	0.556	1.112	0.206	0.9361
$\text{Vil+VR}_{\text{BERT}}$	0.752	0.351	0.274	0.557	1.115	0.205	0.9365
$\text{Vil+VR}_{\text{BERT-Object}}$	0.752	0.352	**0.277**	0.560	1.129	**0.208**	0.9365
$\text{Vil+VR}_{\text{RoBERTa}}$	0.753	**0.353**	0.276	0.559	1.128	0.207	**0.9366**
$\text{Vil+VR}_{\text{RoBERTa-Object}}$	**0.758**	0.344	0.262	0.555	1.234	0.206	0.9365
$\text{Vil+VR}_{\text{BERT-Multi-class}}$	0.753	**0.353**	0.276	0.559	1.131	**0.208**	0.9365
$\text{Vil+VR}_{\text{BERT-Multi-object}}$	0.752	0.351	0.276	0.558	1.123	**0.208**	0.9364
$\text{Vil+VR}_{\text{RoBERTa-Multi-class}}$	0.751	0.351	**0.277**	**0.561**	**1.137**	**0.208**	**0.9366**
$\text{Vil+VR}_{\text{RoBERTa-Multi-object}}$	0.752	**0.353**	**0.277**	0.559	1.131	**0.208**	**0.9366**
Transformer based caption generator (Cornia et al., 2020)							
$\text{Trans}_{\text{Greedy}}$	0.787	0.368	0.276	0.574	1.211	0.215	0.9376
$\text{Trans}_{\text{BeamS}}$	0.793	0.387	0.281	0.582	1.247	**0.220**	**0.9399**
$\text{Trans+VR}_{\text{W-Object}}$ (Fang et al., 2015)	0.786	0.378	0.277	0.579	1.228	0.216	0.9388
$\text{Trans+VR}_{\text{Object}}$ (Wang et al., 2018)	0.790	0.383	0.280	0.580	1.237	0.219	0.9391
$\text{Trans+VR}_{\text{Control}}$ (Cornia et al., 2019)	0.791	**0.388**	0.281	**0.583**	1.248	**0.220**	0.9398
$\text{Trans+VR}_{\text{BERT}}$ (only sim)	0.789	0.380	0.279	0.579	1.234	0.219	0.9389
$\text{Trans+VR}_{\text{BERT}}$	0.793	**0.388**	**0.282**	**0.583**	**1.250**	**0.220**	**0.9399**
$\text{Trans+VR}_{\text{BERT-Object}}$	0.793	0.385	0.281	0.581	1.242	0.219	0.9396
$\text{Trans+VR}_{\text{RoBERTa}}$	0.792	0.386	0.280	0.582	1.244	0.219	0.9395
$\text{Trans+VR}_{\text{RoBERTa-Object}}$	0.792	0.386	0.281	0.582	1.242	0.219	0.9396
$\text{Trans+VR}_{\text{BERT-Multi-class}}$	**0.794**	0.385	0.281	0.582	1.248	**0.220**	0.9395
$\text{Trans+VR}_{\text{BERT-Multi-object}}$	0.792	0.385	0.281	0.582	1.244	**0.220**	0.9395
$\text{Trans+VR}_{\text{RoBERTa-Multi-class}}$	0.791	0.385	0.280	0.581	1.244	0.219	0.9395
$\text{Trans+VR}_{\text{RoBERTa-Multi-object}}$	0.791	0.385	0.281	0.582	1.243	0.219	0.9395

Table 1: Performance of compared baselines on the Karpathy test split with/without **V**isual semantic **R**e-ranking. For each base system, we report performance using a greedy search and the best beam search. Re-ranking is applied to the top-20 results of each system using BERT or RoBERTa for caption-context similarity. The visual contexts are extracted using ResNet152 and Inception Resnet v2 based Faster R-CNN **object** detector. We also report results for Bert-based similarity without a hypothesis probability (rows marked *only sim*).

decoder (*i.e.* language generation stage) with an object frequency counts dictionary on the training dataset. The dictionary is a Fully Connected layer, concatenated with the LSTM and a dense layer, that adds more weight to the most frequent counts object that are seen by the caption and the visual classifier. Second, Cornia et al. (2019) that introduce a controllable grounded captions via a visual context. We train the last stage (decoder), attention and language model LSTM, on the training dataset to visually ground the generated caption based on the visual context.

One observation, shown in Table 1, is that the benefit of using multiple visual contexts for longer captions, which can increase the chance of re-ranking the most visually related candidate caption, as shown in Figure 2 `SimProb` score with VilBERT.

Also, we investigate the statistical significance, using approximate randomization and bootstrapping resampling (Koehn, 2004), to detect minute differences in BLEU and METEOR, and NIST-BLEU[3] (Doddington, 2002) scores, as shown in Table 2, in which we observe the improvement with our re-ranker over BLEU, METEOR[4] and NIST-BLEU. We would like to remark here that, with regards to subtle variations, the statistical significance of metrics such as BLEU, NIST-BLEU, and SacreBLEU (Post, 2018) tend to disagree with human judgement (Mathur et al., 2020; Kocmi et al., 2021). We therefore also conduct a human evaluation study (Section 6).

Figure 4 shows `SimProb` distribution over 40k samples with a pre-trained $\text{RoBERTa}_{\text{Large}}$ similarity score. (Left Figure) Before applying the revision, overall re-ranking scores are relatively low, and (Right Figure) after the visual revision, overall scores increased with more confident about each selected caption. The `SimProb` score positively shifts the distribution over all the samples.

4.3 Limitation

We note that, the quality of the Beam search influences our re-ranker, since non-diverse, repeated captions or fewer novel ones, will make the re-

[3]An improved version of BLEU rewarded infrequently used words by giving greater weighting to rarer words.

[4]It has been previously observed that METEOR correlates better with human judgments than BLEU.

Visual: Minibus

Vil$_{\text{BeamS}}$: a group of people standing around a white truck
Vil+VR$_{\text{BERT}}$: a group of people standing around a white *van*
Human: there is a white van that is stopped on the road

Visual: trifle

Vil$_{\text{BeamS}}$: a close up of a plate of food
Vil+VR$_{\text{BERT}}$: *piece* of food sitting on top of *a white plate*
Human: a white plate and a piece of white cake

Visual: baseball

Vil$_{\text{BeamS}}$: a group of men on a field playing baseball
Vil+VR$_{\text{BERT}}$: *a batter catcher and umpire during a baseball game*
Human: batter catcher and umpire anticipating the next pitch

Figure 3: Example of captions re-ranked by our Visual Re-ranker and the original caption (Best-beam) from the base system. Re-ranked captions are more precise, have a higher lexical diversity, or provide more details.

ranking less effective, as shown in the Unique words per caption in Table 2 with the Transformer baseline.

5 Evaluation of Diversity

We follow the standard diversity evaluation metrics (Shetty et al., 2017; Deshpande et al., 2019): (1) *Div*-1 the ratio of unique unigram to the number of words in caption (2) *Div*-2 the ratio of unique bi-gram to the number of words in the caption, (3) *mBLEU* is the BLEU score between the candidate caption against all human captions (lower value indicate diversity). However, since even though we obtained the top-20 candidates from the base systems, many of them are the same or have very small differences (beam search drawback), which will reflect in small performance differences before and after re-ranking. Therefore, some of the standard metrics are not able to capture these small changes, as shown in Table 2. Consequently, to try to capture the changes and the effect of the re-ranking, we also measured the lexical and semantic diversity with the following metrics:

Type-Token Ratio (TTR): TTR (Brown, 2005) is the number of unique words or types divided by the total number of tokens in a text fragment.

Measure of Textual Lexical Diversity (MTLD): MTLD (McCarthy and Jarvis, 2010) is based on TTR, and measures the average length of subsequences in the text for which a certain TTR is maintained, thus, unlike TTR, being length-invariant.

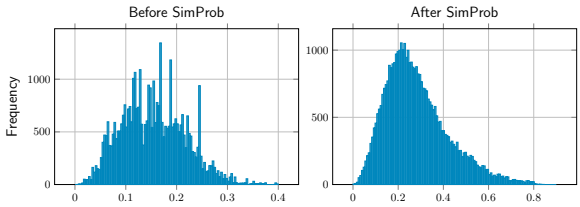

Figure 4: Visualization of the distribution change in re-rank scores on the 40k random sample from the test set. (Left) the score distribution before applying Belief Revision via `SimProb` with LM-GPT-2 initialization. (Right) the score distribution after applying the revision via similarity RoBERTa$_{\text{Large}}$ with the visual context.

For semantic diversity, we use the standard metric (Wang and Chan, 2019) Self-CIDEr. Also, inspired by BERTscore and following (Song et al., 2021) that introduce Sentence Semantic Semantic (SSS) for machine translation, we use SBERT-sts (fine-tuned on sts task) with a cosine score to measure the sentence level semantic correlation against all human references. We observe that the SBERT-sts capture the semantic content better than Self-CIDEr.

Table 2 shows that visual re-ranking selects longer captions and with higher lexical diversity than the base system beam search. Figure 3 shows some examples where visual context re-ranking selected captions with more precise lexica (*van* vs. *truck*), higher diversity –*i.e.* adding details about objects (*white plate* vs. *plate*)– and even selecting a more specific abstraction level (*batter, catcher and umpire* vs. *a group of men*).

6 Human Evaluation

We conducted a human study to investigate human preferences over the visual re-ranked caption. We randomly select 26 test images and give 12 reliable human subjects the option to choose between two captions: (1) Best-beam (BeamS)[5] and (2) **V**isual **R**-ranker. We obtain mixed results, as some re-ranked captions are grammatically incorrect, such as singulars instead of plurals and *sitting on* for objects instead of subjects. Overall, we can observe that 46% of native speakers agreed with our visual re-ranker. Meanwhile, the result for non-native speakers is 61%. In some details, we observe that our model and the non-native human subjects chose those re-ranked captions because they correlated more closely with the visual information regarding

[5]The best result by the baseline in standard metrics.

	Lexical Diversity				Vocabulary		Accuracy 4-gram (p-value)						mBLEU↓		n-gram Diversity		Semantic Diversity	
	MTLD	TTR	Uniq	WPC	Dist	Dist*	BLEU	p	M	p	NIST	p	best-5	best Beam*	Div-1	Div-2	Self-CIDEr	SBERT-sts
Human	19.56	0.90	9.14	14.5	3425	3326												
VilBERT																		
$\text{Vil}_{\text{BeamS}}$	17.28	0.87	8.05	10.5	894	842	0.337	-	0.265	-	0.755	-	0.899	0.454	0.38	0.44	0.661	0.7550
Vil+$\text{VR}_{\text{w-Object}}$ (Fang et al., 2015)	15.90	0.87	8.02	9.20	921	866	0.335	0.109	0.266	0.46	0.764	0.00	0.899	0.455	0.38	0.44	**0.662**	**0.7605**
Vil+$\text{VR}_{\text{Object}}$ (Wang et al., 2018)	15.77	0.87	8.03	9.19	911	854	0.335	0.131	0.266	0.57	0.761	0.043	0.899	0.455	0.38	0.44	0.661	0.7570
Vil+$\text{VR}_{\text{Control}}$ (Cornia et al., 2019)	15.69	0.87	8.07	9.21	**935**	**878**	0.331	0.016	0.266	0.46	0.758	0.118	0.899	0.452	0.38	0.44	0.661	0.7567
Vil+$\text{VR}_{\text{RoBERTa}}$ (ours) (Table 1 Best)	**17.70**	0.87	**8.14**	**10.8**	892	838	**0.339**	0.147	**0.267**	0.04	**0.764**	0.002	**0.896**	**0.451**	0.38	0.44	0.661	0.7562
Transformer based caption generator																		
$\text{Trans}_{\text{BeamS}}$	14.77	0.86	7.44	9.62	935	897	0.341	-	0.272	-	0.781	-	**0.954**	0.499	0.26	0.29	0.660	0.7707
Trans+$\text{VR}_{\text{w-Object}}$ (Fang et al., 2015)	13.14	0.85	7.37	8.62	965	923	0.364	0.00	0.272	0.10	0.789	0.001	0.958	0.498	0.25	0.29	0.660	0.7709
Trans+$\text{VR}_{\text{Object}}$ (Wang et al., 2018)	13.38	0.86	**7.45**	8.69	**982**	**940**	0.369	0.00	0.271	0.04	0.798	0.00	0.958	0.495	0.25	0.28	0.660	0.7700
Trans+$\text{VR}_{\text{Control}}$ (Cornia et al., 2019)	13.25	0.86	7.44	8.64	961	921	0.373	0.00	0.272	0.00	0.796	0.00	0.958	0.498	0.25	0.29	0.660	**0.7716**
Trans+VR_{BERT} (ours) (Table 1 Best)	**14.78**	0.86	**7.45**	**9.76**	980	939	**0.374**	0.00	**0.273**	0.19	**0.806**	0.00	0.963	**0.338**	0.26	**0.30**	0.660	0.7711

Table 2: Diversity statistics and statistical tests. Measuring the diversity of caption before/after re-ranking. Uniq and WPC columns indicate the average of unique/total words per caption, respectively. The BLEU, **M**ETEOR and NIST are an average result, with an approximate randomization test with 1k trials, to estimate the statistically significant improvement with/without our re-ranker. mBLEU and mBLEU* are computed with respect to the top 5-captions and best-beam, respectively. We also report the Distinct vocabulary (Dist* filtering out common and stop words).

the grammatical error in the sentence and unlike the native speaker.

7 Ablation study

Belief Revision relies on a different block (*i.e*. LM, similarity and visual context) to make the final revision. In this study, we perform an ablation study over a random 100 samples from the test set to investigate the effectiveness of the proposed setup. Table 3 shows result with different settings.

Language Model Block: One of the principal intentions in initializing the original hypothesis with a LM is the ability to combine different models. We experimented with product probability, although the mean LM probability achieved better results.

Similarity Block: The degree of similarity between the caption and its visual context is major factor in hypothesis revision. Thus, we experimented with a light model (Distil SBERT) and unsupervised/supervised **Sim**ple **C**ontrastive **S**entence **E**mbedding (SimCSE) for learning sentence similarity. The results show that unsupervised, via dropout with the sentence itself, contrastive learning based similarity performs well in the case of the longer captions, as shown in VilBERT Table 3.

Visual Context Block: We experimented with the most recent model of Contrastive Language-Image Pre-Training (CLIP) (Radford et al., 2021) with Zero-Shot Prediction to extract the visual context. Although, CLIP can predict rare objects better, there is no improvement over ResNet152 with a huge computational cost.

8 Negative Evidence: an extension

Until now, following Blok et al. (2003), we considered only the cases when the visual context increase

Model	B-4	M	R	C	S
VilBERT-VR-GPT-2_{mean} + ResNet					
+ RoBERTa (Table 1 Best)	0.346	0.266	**0.541**	**1.171**	0.205
+ LM-GPT-2_{product}	0.335	0.266	0.535	1.142	0.205
+ DistilSBERT	0.335	0.266	0.537	1.128	0.205
+ SimCSE (Gao et al., 2021)	0.324	0.263	0.529	1.122	**0.207**
+ SimCSE (unsupervised)	**0.349**	**0.267**	0.539	1.164	0.205
+ CLIP (Radford et al., 2021)	0.335	0.261	0.527	1.142	0.202
Trans - VR-GPT-2_{mean} + ResNet					
+ BERT (Table 1 Best)	**0.363**	**0.268**	**0.565**	**1.281**	0.207
+ LM-GPT-2_{product}	0.360	0.261	0.561	1.254	0.205
+ DistilSBERT	0.355	0.260	0.557	1.249	0.205
+ SimCSE (Gao et al., 2021)	0.356	0.265	0.564	1.272	0.207
+ SimCSE (unsupervised)	0.356	0.263	0.560	1.253	**0.208**
+ CLIP (Radford et al., 2021)	0.349	0.260	0.555	1.243	0.203

Table 3: Ablation study using different information from various baselines in each block (*i.e*. LM, similarity and visual context). The (+) refers to the replaced block.

the belief of the hypothesis (Equation 1). Blok et al. (2007) also propose Equation 2 for the case where the absence of evidence leads to a decrease in the probability of the hypothesis.

$$\text{P}(w \mid \neg c) = 1 - (1 - \text{P}(w))^{\alpha} \qquad (2)$$

In our case, we introduce negative evidence in three ways:

False Positive Visual Context (VR^{-low}): We employ the false-positives produced by the visual classifier as negative information to decrease the hypotheses. In this case, we have lower similarity measures as the relation between the visual context and caption are farther apart.

Absent Visual Context (VR^{-high}): The negative information here is a set of visual information extracted from the original visual context (*i.e*. from the visual classifier) which does not exist in the image. Thus, the visual context produced by the classifier is used as a query on a pre-trained 840B GloVe (Pennington et al., 2014), with cosine similarity, to retrieve the closest visual context in the same semantic space (*e.g*. visual: river, closest visual: valley).

Model	B-1	B-4	M	R	C	S	BERTscore
VilBERT (Lu et al., 2020)							
$\text{Vil}_{\text{Greedy}}$	0.751	0.330	0.272	0.554	1.104	0.207	0.9352
$\text{Vil}_{\text{BeamS}}$	0.752	0.351	0.274	0.557	1.115	0.205	0.9363
$\text{Vil+VR}_{\text{W-Object}}$ (Fang et al., 2015)	**0.756**	0.348	0.274	**0.559**	1.123	0.206	0.9365
$\text{Vil+VR}_{\text{Object}}$ (Wang et al., 2018)	**0.756**	0.348	0.274	**0.559**	1.120	0.206	0.9364
$\text{Vil+VR}_{\text{Control}}$ (Cornia et al., 2019)	0.753	0.345	0.274	0.557	1.116	0.206	0.9361
$\text{Vil+VR}_{\text{RoBERTa}}$ Table 1 (positive)	0.753	**0.353**	**0.276**	**0.559**	1.128	0.207	**0.9366**
$\text{Vil+VR}^{-low}_{\text{RoBERTa}}$	0.748	0.349	0.275	0.557	1.116	0.206	0.9362
$\text{Vil+VR}^{-high}_{\text{RoBERTa}}$	0.748	0.349	0.275	0.557	1.116	0.206	0.9364
$\text{Vil+VR}^{-pos}_{\text{GloVe}}$	0.751	0.351	0.276	0.558	1.123	0.207	0.9364
$\text{Vil+VR}^{-joint}_{\text{RoBERTa+GloVe}}$	0.750	0.351	**0.276**	**0.559**	**1.126**	**0.208**	**0.9365**
Transformer based caption generator (Cornia et al., 2020)							
$\text{Trans}_{\text{Greedy}}$	0.787	0.368	0.276	0.574	1.211	0.215	0.9376
$\text{Trans}_{\text{BeamS}}$	0.793	0.387	0.281	0.582	1.247	**0.220**	**0.9399**
$\text{Vil+VR}_{\text{W-Object}}$ (Fang et al., 2015)	0.786	0.348	0.274	0.559	1.123	0.206	0.9365
$\text{Trans+VR}_{\text{Object}}$ (Wang et al., 2018)	0.790	0.383	0.280	0.580	1.237	0.219	0.9391
$\text{Trans+VR}_{\text{Control}}$ (Cornia et al., 2019)	0.791	**0.388**	0.281	**0.583**	1.248	**0.220**	0.9398
$\text{Trans+VR}_{\text{BERT}}$ Table 1 (positive)	0.793	**0.388**	**0.282**	**0.583**	**1.250**	**0.220**	**0.9399**
$\text{Trans+VR}^{-low}_{\text{BERT}}$	0.791	0.387	0.280	0.582	1.242	0.218	0.9396
$\text{Trans+VR}^{-high}_{\text{BERT}}$	0.793	0.385	0.282	0.582	1.243	0.219	0.9397
$\text{Trans+VR}^{-pos}_{\text{GloVe}}$	**0.794**	**0.388**	**0.282**	**0.583**	**1.249**	**0.220**	**0.9399**
$\text{Trans+VR}^{-joint}_{\text{BERT+GloVe}}$	0.793	0.387	0.281	0.582	1.247	**0.220**	0.9398

Table 4: Comparison between positive (single concept VR) and Negative Belief Revision (NBR) on the Karpathy split. The NBR uses a *high similarity* VR^{-high} object related to the positive visual but not in the image, *low similarity* VR^{-low} false positive from the visual classifier, and positive visual via static word level similarity VR^{-pos}. **Boldface** fonts reflect improvement over the baseline.

Positive Visual Context (VR^{-pos}): As the previous two-approaches produced unexpected results with low and high similarities as shown in Table 4, we approach this from a positive belief revision perspective but as negative evidence. Until now, all approaches use sentence-level semantic similarity, but in this experiment, we convert the similarity from sentence to word level. For this first, we employ an LSTM based CopyRNN keyphrase extractor (Meng et al., 2017), which is trained on a combined preprocessed wikidump (*i.e.* keyword, short sentence) and SemEval 2017 Task 10 (Keyphrases from scientific publications) (Augenstein et al., 2017). Secondly, GloVe is used to compute the cosine similarity with the visual context in a word-level manner. We consider this as negative evidence for the following reasons: (1) the similarity is computed without the context of the sentence and (2) the static embedding is computed without knowing the sense of the word. The advantage of VR^{-pos} is that a high similarity and confident visual information are present and thus satisfies the revision.

Joint Evidence (VR^{-joint}): Finally, we combined the best model VR^{-pos} with the best positive evidence (baseline+$\text{VR}_{\text{BERT/RoBERTa}}$) with a simple multiplication.

Table 4 shows that there is some refinement results with the negative evidence over both baselines with VR^{-pos} and VR^{-joint}. However, there is no improvement over the original positive evidence.

9 Discussion: Limitations

Object Classifier Failure Cases: As the belief revision approach relies heavily on the object in the image for the likelihood revision, the quality of the object classifier is critical for the final decision. Here, we show some failure cases when the visual classifier struggle with complex background (*i.e.* wrong visual, object hallucination, *etc.*) as shown in Figure 5. Note that, if no related visual is present in the image the belief revision score will back off to 1 (no revision needed).

Object-to-Caption Similarity Score: Another limitation is the low/high cosine similarity score, which unbalances the likelihood revision. For example, a visual context *paddle* and the caption: *a man riding a surfboard on a wave* have low cosine scores when using a pre-trained model that is fine-tuned on sentence-to-sentence semantic similarity tasks (*i.e.* STS-B). Note that, we tackle this problem by adding multiple visual contexts as shown in Figure 2.

Related work

Modern sophisticated image captioning systems focus heavily on visual grounding to capture real-world scenarios. Early work Fang et al. (2015) builds a visual detector to guide and re-ranked image captioning with global similarity. The work of (Wang et al., 2018) investigates the informa-

Vil$_{\text{BeamS}}$: a pile of trash sitting inside of a building
Vil+VR$_{\text{BERT}}$: a pile of trash sitting in front of a building ✗
Human: an older floor light sits deserted in an abandoned hospital

Visual: vacuum ✗

Trans$_{\text{BeamS}}$: a kitchen with black counter tops and wooden cabinets
Tans+VR$_{\text{BERT}}$: a kitchen counter with a black counter top ✗
Human: a kitchen with a sink bottles jars and a dishwasher

Visual: barbershop ✗

Figure 5: Failure cases of the object detectors. The object classifier struggle with images with complex background and out-of-context object.

tiveness of visual or object information (*e.g.* object frequency count, size and position) in an end-to-end caption generation. Another work Cornia et al. (2019) proposes controlled caption language grounding through visual regions from the image. More recently, Gupta et al. (2020) introduce weakly supervised contrastive learning via object context and language modeling (*i.e.* BERT) for caption phrase grounding. Inspired by these works, (Fang et al., 2015) carried out re-ranking via visual information, (Wang et al., 2018; Cornia et al., 2019; Chen et al., 2020) explored the benefits of object information in image captioning, (Gupta et al., 2020) exploited the benefits of language modeling to extract contextualized word representations and the exploitation of the semantic coherency in caption language grounding (Zhang et al., 2021a), we purpose an object based re-ranker via human inspired logic reasoning with Belief Revision to re-rank the most closely related captions with contextualized semantic similarity.

Unlike the earlier approaches, our methods employ state-of-the-art tools and pre-trained models. Therefore, the system will keep improving in the future as better systems become available. In addition, our model can be directly used as a drop-in complement for any caption system that outputs a list of candidate hypothesis.

Conclusion

In this work, we aim to demonstrate that the Belief Revision approach that works well with human judgment can be applied to Image Captioning by employing human-inspired reasoning via a pre-trained model (*i.e.* GPT, BERT). Belief Revision (BR) is an approach for obtaining the likelihood revisions based on similarity scores via human judgment. We demonstrate the benefits of the approach by showing that two state-of-the-art Transformer-based image captioning results are improved via simple language grounding with visual context information. In particular, we show the accuracy gain in a benchmark dataset using two methods: (1) BR with positive visual evidence (increase the hypothesis) and (2) negative evidence (decrease the hypothesis), with wrong visual *i.e.* false positive by the classifier. However, this adaptation could be applied to many re-ranking tasks in NLP (text generation, multimodel MT, lexical selection, *etc.*), as well as in Computer Vision applications such as visual storytelling.

Ethical Considerations

The core contribution of our paper is algorithmic for the task of image captioning. As such we do not foresee any downstream harms propagated immediately by our proposal. We however acknowledge that due to the very nature of the data driven processing, there could be an amplification or propagation of potential biases existing in the datasets.

References

Martín Abadi, Paul Barham, Jianmin Chen, Zhifeng Chen, Andy Davis, Jeffrey Dean, Matthieu Devin, Sanjay Ghemawat, Geoffrey Irving, Michael Isard, et al. 2016. Tensorflow: A system for large-scale machine learning. In *12th {USENIX} symposium on operating systems design and implementation ({OSDI} 16)*, pages 265–283.

Peter Anderson, Basura Fernando, Mark Johnson, and Stephen Gould. 2016. Spice: Semantic propositional image caption evaluation. In *European conference on computer vision*, pages 382–398. Springer.

Peter Anderson, Xiaodong He, Chris Buehler, Damien Teney, Mark Johnson, Stephen Gould, and Lei Zhang. 2018. Bottom-up and top-down attention for image captioning and visual question answering. In *Proceedings of the IEEE conference on computer vision and pattern recognition*.

Isabelle Augenstein, Mrinal Das, Sebastian Riedel, Lakshmi Vikraman, and Andrew McCallum. 2017. Semeval 2017 task 10: Scienceie-extracting keyphrases and relations from scientific publications. *arXiv preprint arXiv:1704.02853*.

Satanjeev Banerjee and Alon Lavie. 2005. Meteor: An automatic metric for mt evaluation with improved correlation with human judgments. In *Proceedings of the acl workshop on intrinsic and extrinsic evaluation*

measures for machine translation and/or summarization, pages 65–72.

Sergey Blok, Douglas Medin, and Daniel Osherson. 2003. Probability from similarity. In *AAAI Spring Symposium on Logical Formalization of Commonsense Reasoning*.

Sergey V Blok, Douglas L Medin, and Daniel N Osherson. 2007. Induction as conditional probability judgment. *Memory & Cognition*, 35(6):1353–1364.

Keith Brown. 2005. *Encyclopedia of language and linguistics*, volume 1. Elsevier.

Daniel Cer, Mona Diab, Eneko Agirre, Inigo Lopez-Gazpio, and Lucia Specia. 2017. Semeval-2017 task 1: Semantic textual similarity-multilingual and cross-lingual focused evaluation. *arXiv preprint arXiv:1708.00055*.

Shizhe Chen, Qin Jin, Peng Wang, and Qi Wu. 2020. Say as you wish: Fine-grained control of image caption generation with abstract scene graphs. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 9962–9971.

Marcella Cornia, Lorenzo Baraldi, and Rita Cucchiara. 2019. Show, control and tell: A framework for generating controllable and grounded captions. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 8307–8316.

Marcella Cornia, Matteo Stefanini, Lorenzo Baraldi, and Rita Cucchiara. 2020. Meshed-memory transformer for image captioning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 10578–10587.

Bo Dai, Sanja Fidler, Raquel Urtasun, and Dahua Lin. 2017. Towards diverse and natural image descriptions via a conditional gan. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 2970–2979.

Aditya Deshpande, Jyoti Aneja, Liwei Wang, Alexander G Schwing, and David Forsyth. 2019. Fast, diverse and accurate image captioning guided by part-of-speech. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 10695–10704.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. Bert: Pre-training of deep bidirectional transformers for language understanding. In *NAACL-HLT (1)*.

George Doddington. 2002. Automatic evaluation of machine translation quality using n-gram co-occurrence statistics. In *Proceedings of the second international conference on Human Language Technology Research*, pages 138–145.

Hao Fang, Saurabh Gupta, Forrest Iandola, Rupesh K Srivastava, Li Deng, Piotr Dollár, Jianfeng Gao, Xiaodong He, Margaret Mitchell, John C Platt, et al. 2015. From captions to visual concepts and back. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1473–1482.

Tianyu Gao, Xingcheng Yao, and Danqi Chen. 2021. SimCSE: Simple contrastive learning of sentence embeddings. *arXiv preprint arXiv:2104.08821*.

Tanmay Gupta, Arash Vahdat, Gal Chechik, Xiaodong Yang, Jan Kautz, and Derek Hoiem. 2020. Contrastive learning for weakly supervised phrase grounding. In *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part III 16*, pages 752–768. Springer.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. 2016. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*.

Simao Herdade, Armin Kappeler, Kofi Boakye, and Joao Soares. 2019. Image captioning: Transforming objects into words. *arXiv preprint arXiv:1906.05963*.

Jonathan Huang, Vivek Rathod, Chen Sun, Menglong Zhu, Anoop Korattikara, Alireza Fathi, Ian Fischer, Zbigniew Wojna, Yang Song, Sergio Guadarrama, et al. 2017. Speed/accuracy trade-offs for modern convolutional object detectors. In *Proceedings of the IEEE conference on computer vision and pattern recognition*.

Lun Huang, Wenmin Wang, Jie Chen, and Xiao-Yong Wei. 2019. Attention on attention for image captioning. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 4634–4643.

Daphne Ippolito, Reno Kriz, Maria Kustikova, João Sedoc, and Chris Callison-Burch. 2019. Comparison of diverse decoding methods from conditional language models. *arXiv preprint arXiv:1906.06362*.

Andrej Karpathy and Li Fei-Fei. 2015. Deep visual-semantic alignments for generating image descriptions. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 3128–3137.

Tom Kocmi, Christian Federmann, Roman Grundkiewicz, Marcin Junczys-Dowmunt, Hitokazu Matsushita, and Arul Menezes. 2021. To ship or not to ship: An extensive evaluation of automatic metrics for machine translation. *arXiv preprint arXiv:2107.10821*.

Philipp Koehn. 2004. Statistical significance tests for machine translation evaluation. In *Proceedings of the 2004 conference on empirical methods in natural language processing*, pages 388–395.

Chin-Yew Lin. 2004. Rouge: A package for automatic evaluation of summaries. In *Text summarization branches out*, pages 74–81.

Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C Lawrence Zitnick. 2014. Microsoft coco: Common objects in context. In *European conference on computer vision*. Springer.

Pierre Lison and Jörg Tiedemann. 2016. Opensubtitles2016: Extracting large parallel corpora from movie and tv subtitles.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Roberta: A robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692*.

Jiasen Lu, Vedanuj Goswami, Marcus Rohrbach, Devi Parikh, and Stefan Lee. 2020. 12-in-1: Multi-task vision and language representation learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 10437–10446.

Nitika Mathur, Timothy Baldwin, and Trevor Cohn. 2020. Tangled up in bleu: Reevaluating the evaluation of automatic machine translation evaluation metrics. *arXiv preprint arXiv:2006.06264*.

Philip M McCarthy and Scott Jarvis. 2010. Mtld, vocd-d, and hd-d: A validation study of sophisticated approaches to lexical diversity assessment. *Behavior research methods*, 42(2):381–392.

Rui Meng, Sanqiang Zhao, Shuguang Han, Daqing He, Peter Brusilovsky, and Yu Chi. 2017. Deep keyphrase generation. *arXiv preprint arXiv:1704.06879*.

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. Bleu: a method for automatic evaluation of machine translation. In *Proceedings of the 40th annual meeting of the Association for Computational Linguistics*, pages 311–318.

Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, et al. 2019. Pytorch: An imperative style, high-performance deep learning library. *arXiv preprint arXiv:1912.01703*.

Jeffrey Pennington, Richard Socher, and Christopher D Manning. 2014. Glove: Global vectors for word representation. In *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*, pages 1532–1543.

Matt Post. 2018. A call for clarity in reporting bleu scores. In *Proceedings of the Third Conference on Machine Translation: Research Papers*, pages 186–191.

Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, et al. 2021. Learning transferable visual models from natural language supervision. *arXiv preprint arXiv:2103.00020*.

Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. 2019. Language models are unsupervised multitask learners. *OpenAI blog*.

Nils Reimers and Iryna Gurevych. 2019. Sentence-bert: Sentence embeddings using siamese bert-networks. *arXiv preprint arXiv:1908.10084*.

Zhou Ren, Xiaoyu Wang, Ning Zhang, Xutao Lv, and Li-Jia Li. 2017. Deep reinforcement learning-based image captioning with embedding reward. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 290–298.

Steven J Rennie, Etienne Marcheret, Youssef Mroueh, Jerret Ross, and Vaibhava Goel. 2017. Self-critical sequence training for image captioning. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 7008–7024.

Ahmed Sabir, Francesc Moreno-Noguer, and Lluís Padró. 2018. Visual re-ranking with natural language understanding for text spotting. In *Asian Conference on Computer Vision*, pages 68–82. Springer.

Rakshith Shetty, Marcus Rohrbach, Lisa Anne Hendricks, Mario Fritz, and Bernt Schiele. 2017. Speaking the same language: Matching machine to human captions by adversarial training. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 4135–4144.

Yurun Song, Junchen Zhao, and Lucia Specia. 2021. Sentsim: Crosslingual semantic evaluation of machine translation. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 3143–3156.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in neural information processing systems*, pages 5998–6008.

Ramakrishna Vedantam, Samy Bengio, Kevin Murphy, Devi Parikh, and Gal Chechik. 2017. Context-aware captions from context-agnostic supervision. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 251–260.

Ramakrishna Vedantam, C Lawrence Zitnick, and Devi Parikh. 2015. Cider: Consensus-based image description evaluation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 4566–4575.

Ashwin Vijayakumar, Michael Cogswell, Ramprasaath Selvaraju, Qing Sun, Stefan Lee, David Crandall, and Dhruv Batra. 2018. Diverse beam search for improved description of complex scenes. In *Proceedings of the AAAI Conference on Artificial Intelligence*.

Oriol Vinyals, Alexander Toshev, Samy Bengio, and Dumitru Erhan. 2015. Show and tell: A neural image caption generator. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 3156–3164.

Jiuniu Wang, Wenjia Xu, Qingzhong Wang, and Antoni B Chan. 2020. Compare and reweight: Distinctive image captioning using similar images sets. In *European Conference on Computer Vision*, pages 370–386. Springer.

Josiah Wang, Pranava Madhyastha, and Lucia Specia. 2018. Object counts! bringing explicit detections back into image captioning. *arXiv preprint arXiv:1805.00314*.

Qingzhong Wang and Antoni B Chan. 2019. Describing like humans: on diversity in image captioning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 4195–4203.

Kelvin Xu, Jimmy Ba, Ryan Kiros, Kyunghyun Cho, Aaron Courville, Ruslan Salakhudinov, Rich Zemel, and Yoshua Bengio. 2015. Show, attend and tell: Neural image caption generation with visual attention. In *International conference on machine learning*, pages 2048–2057.

Quanzeng You, Hailin Jin, Zhaowen Wang, Chen Fang, and Jiebo Luo. 2016. Image captioning with semantic attention. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 4651–4659.

Tianyi Zhang, Varsha Kishore*, Felix Wu*, Kilian Q. Weinberger, and Yoav Artzi. 2020. Bertscore: Evaluating text generation with bert. In *International Conference on Learning Representations*.

Wenqiao Zhang, Haochen Shi, Siliang Tang, Jun Xiao, Qiang Yu, and Yueting Zhuang. 2021a. Consensus graph representation learning for better grounded image captioning. In *Proc 35 AAAI Conf on Artificial Intelligence*.

Xuying Zhang, Xiaoshuai Sun, Yunpeng Luo, Jiayi Ji, Yiyi Zhou, Yongjian Wu, Feiyue Huang, and Rongrong Ji. 2021b. Rstnet: Captioning with adaptive attention on visual and non-visual words. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 15465–15474.

A Appendix: Additional Result

`SimProb` with Two Visual Contexts. Until now, we experimented with one or multiple visual contexts at the same time; however, when we have more than one evidence supporting the revision, we can reason and choose which one to use to revise the hypothesis. In this scenario we begin with the conditional probability that comes from the dominant premise alone $P(Q_c|Q_a)$ (let's assume Q_a is dominant). Then we add a fraction of the remaining lack of trust or confidence $1 - P(Q_c|Q_a)$ that the dominant conditional *leaves behind*. The similarity between premise categories defines the size of the portion or fraction $sim(a, b)$ and separates the influence of the nondominant premise on the conclusion prior $P(Q_c|Q_b) - P(Q_c)$. The component of similarity is designed to reduce the impact of the non-dominant premise when the premises are redundant. Note that the proposed method in equation below guarantees an increase in strength with additional premises. In addition, this property, $\Pr(Qc|Qa, Qb) \geq \Pr(Qc|Qa), \Pr(Qc|Qb)$, is noncompetitive in the sense that one category does not reduce the probability of concept for another. Following the notation of Equation 1, we write the two visual contexts `SimProb` as:

$$\mathrm{P}(w \mid c_1, c_2) = \beta M + (1 - \beta)S, \text{ where}$$

$$\beta = \max \left\{ \begin{array}{l} \mathrm{sim}(w, c_1) \\ \mathrm{sim}(w, c_2) \\ \mathrm{sim}(c_1, c_2) \\ 1.0 - \mathrm{sim}(w, c_1) \\ 1.0 - \mathrm{sim}(w, c_2) \\ \mathrm{P}(c_1) \\ \mathrm{P}(c_2) \end{array} \right\}$$

$$M = \max\{\mathrm{P}(w \mid c_1), \mathrm{P}(w \mid c_2\}, S = \mathrm{P}(w \mid c_1) + \mathrm{P}(w \mid c_2) - \mathrm{P}(w \mid c_1) \times \mathrm{P}(w \mid c_2)$$

where $\mathrm{P}(w \mid c_1)$ and $\mathrm{P}(w \mid c_2)$ are defined by Equation 1, and the two visual contexts are: (1) c_1 ResNet is the label **Class** and (2) the c_2 COCO **Object** categories are from Inception-ResNet v2 based Faster RCNN. Note that β takes the max of all models, and thus it is not breaking the formation if one of the similarities or probabilities is not confident enough (*i.e.* if it is below the threshold).

The last rows in Table 8 show the result of the `SimProb` selecting the best visual context of the two visuals. However, although there is some improvement over the other approaches (*i.e.* single and multi-visual contexts), it is not significant enough to justify the computational cost as a post-processing approach.

Figure 7 shows the benefit of employing common observation via Unigram LM in comparison to the classifier confident, (Left Figure) a denser `SimProb` score caption re-ranking.

Additional Statistical Significance Analysis. Table 5 shows the full results of the statistically significant test via pair bootstrap resampling (Koehn, 2004)[6] with BLEU and NIST. The NIST metric is

[6] https://github.com/moses-smt/mosesdecoder/tree/master/scripts/analysis

Model	BLEU (p-value) B1	B2	B3	B4	NIST (p-value) N1	N2	N3	N4
VilBERT								
$\text{Vil}_{\text{BeamS}}$	0.740	0.578	0.441	0.337	0.492	0.672	0.731	0.755
$\text{Vil+VR}_{\text{w-Object}}$ (Fang et al., 2015)	0.744 (0.019)	**0.581** (0.063)	0.442 (0.371)	0.335 (0.109)	**0.497** (0.00)	**0.680** (0.00)	**0.740** (0.001)	0.764 (0.00)
$\text{Vil+VR}_{\text{Object}}$ (Wang et al., 2018)	**0.745** (0.01)	**0.581** (0.091)	0.441 (0.429)	0.335 (0.131)	0.496 (0.003)	0.677 (0.015)	0.737 (0.012)	0.761 (0.043)
$\text{Vil+VR}_{\text{Control}}$ (Cornia et al., 2019)	0.741 (0.233)	0.577 (0.29)	0.439 (0.104)	0.331 (0.016)	0.494 (0.015)	0.676 (0.046)	0.735 (0.064)	0.758 (0.118)
$\text{Vil+VR}_{\text{RoBERTa}}$ (ours) (Table 1 Best)	0.741 (0.382)	0.580 (0.129)	**0.443** (0.00)	**0.339** (0.147)	**0.497** (0.00)	**0.680** (0.00)	0.740 (0.00)	**0.764** (0.002)
Transformer based caption generator								
$\text{Trans}_{\text{BeamS}}$	0.726	0.584	0.451	0.341	0.492	0.688	0.756	0.781
$\text{Trans+VR}_{\text{w-Object}}$ (Fang et al., 2015)	0.775 (0.00)	0.625 (0.00)	0.482 (0.00)	0.364 (0.00)	0.498 (0.00)	0.696 (0.00)	0.764 (0.001)	0.789 (0.001)
$\text{Trans+VR}_{\text{Object}}$ (Wang et al., 2018)	0.777 (0.00)	0.628 (0.00)	0.486 (0.00)	0.369 (0.00)	0.504 (0.00)	0.703 (0.00)	0.773 (0.00)	0.798 (0.00)
$\text{Trans+VR}_{\text{Control}}$ (Cornia et al., 2019)	0.780 (0.00)	0.623 (0.00)	**0.490** (0.00)	0.373 (0.00)	0.502 (0.00)	0.700 (0.00)	0.771 (0.00)	0.796 (0.00)
$\text{Trans+VR}_{\text{BERT}}$ (ours) (Table 1 Best)	**0.781** (0.00)	**0.631** (0.00)	**0.490** (0.00)	**0.374** (0.00)	**0.509** (0.00)	**0.710** (0.00)	**0.781** (0.00)	**0.806** (0.00)

Table 5: Result with pair bootstrapping resampling test via 1k trial (Koehn, 2004) on the significant improvement before and ranking with BLEU and NIST.

Model	SacreBLEU Baseline Avg	New Avg	delta	Baseline better confidence %	New better confidence %
VilBERT (Lu et al., 2020)					
$\text{Vil}_{\text{BeamS}}$	9.10				
$\text{Vil+VR}_{\text{W-Object}}$ (Fang et al., 2015)		9.18	0.08	27.60	72.40
$\text{Vil+VR}_{\text{Object}}$ (Wang et al., 2018)		8.89	-0.22	93.50	6.50
$\text{Vil+VR}_{\text{Control}}$ (Cornia et al., 2019)		9.01	-0.09	75.60	24.40
$\text{Vil+VR}_{\text{RoBERTa}}$ (ours) (Table 1 Best)		**9.29**	0.18	8.50	**91.50**
Transformer Caption Generator (Cornia et al., 2020)					
$\text{Trans}_{\text{BeamS}}$	10.16				
$\text{Trans+VR}_{\text{W-Object}}$ (Fang et al., 2015)		10.01	-0.16	93.80	6.20
$\text{Trans+VR}_{\text{Object}}$ (Wang et al., 2018)		10.18	0.02	43.40	56.60
$\text{Trans+VR}_{\text{Control}}$ (Cornia et al., 2019)		10.09	-0.07	83.80	16.20
$\text{Trans+VR}_{\text{BERT}}$ (ours) (Table 1 Best)		**10.36**	0.19	1.10	**98.90**

Table 6: Result with pair bootstrapping resampling test via 1k trial (Koehn, 2004) on the significant improvement before and ranking with Sacrebleu (Post, 2018).

an improved version of BELU that rewards infrequently used words by giving greater weighting to rarer words.

Also, we employ Sacrebleu[7] (Post, 2018) to investigate the statistically significant improvement, using delta, of our re-ranker with a BLEU score using the same approach as that above. Table 6 shows that our method performs better as **new better confidence** than the two baselines.

Additional Diversity Analysis. Table 7 shows part-of-speech tagging (POS) results before and after visual re-ranking. The proposed model VR yields a richer output in all POS tags in both baselines.

Full Experimental Results. Table 8 and Table 10 show the full results of our experiments, with the most common metrics used for image captioning, for positive and negative evidence, respectively. Also, as we mentioned before, inspired by BERTscore and following (Song et al., 2021) we employ sentence-to-sentence semantic similarity score to compare candidate captions with human references with pre-trained Sentence-$\text{RoBERTa}_{\text{LARGE}}$ (Reimers and Gurevych, 2019) tuned for general STS task. Unlike BERTscore which aligns word-to-word similarites, SBERT-sts builds a semantic vector for the whole sentence, which can be used to compare candidate captions with human references.

Additional Ablation Study. Table 11 shows additional ablation study experiments with different information.

Training Dataset. As shown in Figure 6, we use two approaches to match and filter out not related visual context: 1) Threshold: to filter out the probabilities prediction when the visual classifier is not confident. 2) Semantic alignment: to match the most related caption to its environmental context. In more detail, we use cosine similarity with GloVe to match the visual with its context. Table 9 illustrates samples of the enriched human annotation, and caption dataset, with visual context information.

Why Positive Visual as Negative Evidence ? As we mentioned in the main script, we consider this as negative evidence for the following reasons: (1)

[7] https://github.com/pytorch/translate/blob/master/pytorch_translate/bleu_significance.py

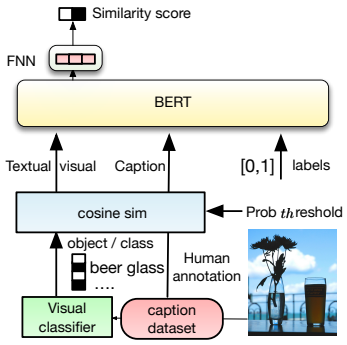

Figure 6: Dataset preprocessing until training. We use two methods to filter our non-related visual context (1) probability threshold: to filter out the visual context, and (2) semantic alignment with the caption via cosine distance (semantic relation).

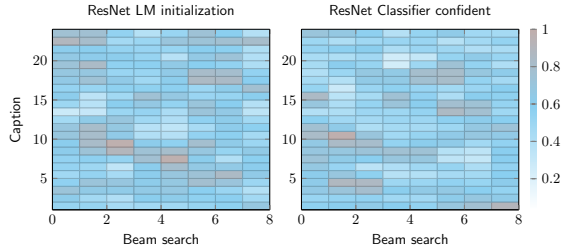

Figure 7: `SimProb` score of top-8 Beam search caption re-ranking (Right) with the visual classifier confidence probability without any initialization, (Left) with visual context that initialized by general common observation *i.e.* LM.

Model	Noun	Verb	Adj	Conj
VilBERT				
$\text{Vil}_{\text{BeamS}}$	14094	3586	3220	7914
$\text{Vil+VR}_{\text{RoBERTa}}$	**14403**	**3739**	**3325**	**8233**
Transformer				
$\text{Trans}_{\text{BeamS}}$	13961	3111	2004	7458
$\text{Trans+VR}_{\text{BERT}}$	**14203**	**3146**	**2056**	**7563**

Table 7: Most frequent POS tag before and after visual re-ranking. The result shows that after Visual Re-ranking both captions have more noun, verb, *etc*.

the similarity is computed without the context of the sentence and (2) the static embedding (without knowing the sense of the word, *e.g. bar* for alcoholic drinking or rectangular solid piece of block). Although this approach relies on positive information, which is not the main intention of the negative evidence, the results demonstrate that there is a new direction of research that can be conducted using positive information as negative evidence. Figure 8 shows that the original hypothesis is decreased with the positive information. Note that, we were surprised by the negative result when trying this approach as negative evidence (*i.e.* false

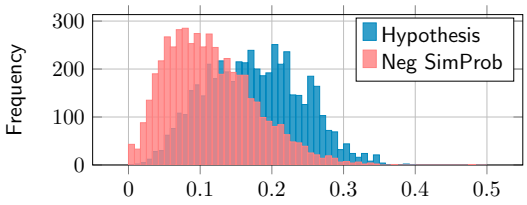

Figure 8: Visualization of Negative Evidence Neg-`SimProb` distribution on random samples from the test set (karpathy split). The negative visual information VR^{-pos} decreases the hypothesis that is initialized by LM-GPT-2.

visual with low similarity), and the result is worse than the baseline. Therefore, a BR with negative evidence breaks the beam search, as there is not enough positive evidence to support the revision.

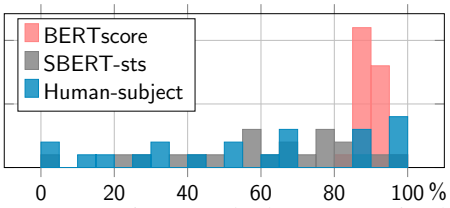

Figure 9: Comparison results between native human subject, BERTscore, and sentence level metric SBERT-sts on the test set.

Figure 10: **Human Evaluation**. The user interface presented to our human subjects through the survey website asking them to re-rank the most descriptive caption candidates based on the visual information.

B Human Evaluation

We conducted a human study to investigate human preferences over the visual re-ranked caption.

Model	B-1	B-2	B-3	B-4	M	R	C	S	BERTscore	SBERT-sts
VilBERT (Lu et al., 2020)										
$\text{Vil}_{\text{Greedy}}$	0.751	0.587	0.441	0.330	0.272	0.554	1.104	0.207	0.9352	0.7550
$\text{Vil}_{\text{BeamS}}$	0.752	0.592	0.456	0.351	0.274	0.557	1.115	0.205	0.9363	0.7550
$\text{Vil+VR}_{\text{W-Object}}$ (Fang et al., 2015)	0.756	0.595	0.456	0.348	0.274	0.559	1.123	0.206	0.9365	**0.7605**
$\text{Vil+VR}_{\text{Object}}$ (Wang et al., 2018)	0.756	0.594	0.455	0.348	0.274	0.559	1.120	0.206	0.9364	0.7570
$\text{Vil+VR}_{\text{Control}}$ (Cornia et al., 2019)	0.753	0.591	0.453	0.345	0.274	0.557	1.116	0.206	0.9361	0.7565
$\text{Vil+GPT-2}_{\text{mean}}$ (only LM)	0.749	0.590	0.455	0.351	0.276	0.558	1.124	0.208	0.9364	0.7546
$\text{Vil+VR}_{\text{BERT}}$ (only sim)	0.753	0.591	0.452	0.343	0.273	0.556	1.112	0.206	0.9361	0.7562
$\text{Vil+VR}_{\text{BERT}}$	0.752	0.592	0.456	0.351	0.274	0.557	1.115	0.205	0.9365	0.7567
$\text{Vil+VR}_{\text{BERT-Object}}$	0.752	0.592	0.457	0.352	0.277	0.560	1.129	0.208	0.9365	0.7562
$\text{Vil+VR}_{\text{RoBERTa}}$	0.753	0.594	0.458	**0.353**	0.276	0.559	1.128	0.207	**0.9366**	0.7562
$\text{Vil+VR}_{\text{RoBERTa-Object}}$	**0.758**	**0.611**	**0.465**	0.344	0.262	0.555	1.234	0.206	0.9365	0.7554
$\text{Vil+VR}_{\text{BERT-Multi-class}}$	0.753	0.593	0.458	**0.353**	0.276	0.559	1.131	0.208	0.9365	0.7586
$\text{Vil+VR}_{\text{BERT-Multi-object}}$	0.752	0.592	0.456	0.351	0.276	0.558	1.123	0.208	0.9364	0.7566
$\text{Vil+VR}_{\text{RoBERTa-Multi-class}}$	0.751	0.591	0.456	0.351	**0.277**	**0.561**	**1.137**	0.208	**0.9366**	0.7589
$\text{Vil+VR}_{\text{RoBERTa-Multi-object}}$	0.753	0.593	0.458	**0.353**	0.276	0.559	1.131	0.208	0.9365	0.7586
$\text{Vil+VR}_{\text{BERT-Class+object}}$	0.752	0.592	0.455	0.350	0.276	0.559	1.126	**0.209**	0.9365	0.7563
$\text{Vil+VR}_{\text{RoBERTa-class+object}}$	0.752	0.592	0.457	0.352	**0.277**	0.559	1.127	0.208	0.9365	0.7558
Transformer Caption Generator (Cornia et al., 2020)										
$\text{Trans}_{\text{Greedy}}$	0.787	0.634	0.488	0.368	0.276	0.574	1.211	0.215	0.9376	0.7649
$\text{Trans}_{\text{BeamS}}$	0.793	0.645	0.504	0.387	0.281	0.582	1.247	**0.220**	**0.9399**	0.7707
$\text{Trans+VR}_{\text{W-Object}}$ (Fang et al., 2015)	0.786	0.638	0.497	0.378	0.277	0.579	1.228	0.216	0.9388	0.7709
$\text{Trans+VR}_{\text{Object}}$ (Wang et al., 2018)	0.790	0.642	0.501	0.383	0.280	0.580	1.237	0.219	0.9391	0.7700
$\text{Trans+VR}_{\text{Control}}$ (Cornia et al., 2019)	0.791	0.644	**0.505**	**0.388**	0.281	**0.583**	1.248	**0.220**	0.9398	0.7716
$\text{Trans+GPT-2}_{\text{mean}}$ (only LM)	0.791	0.643	0.503	0.386	0.281	0.582	1.242	0.219	0.9396	0.7714
$\text{Trans+VR}_{\text{BERT}}$ (only sim)	0.789	0.640	0.498	0.380	0.279	0.579	1.234	0.219	0.9389	0.7693
$\text{Trans+VR}_{\text{BERT}}$	0.793	**0.646**	**0.505**	**0.388**	**0.282**	**0.583**	1.250	**0.220**	**0.9399**	0.7711
$\text{Trans+VR}_{\text{BERT-Object}}$	0.793	0.644	0.503	0.385	0.281	0.581	1.242	0.219	0.9396	0.7695
$\text{Trans+VR}_{\text{RoBERTa}}$	0.792	0.644	0.504	0.386	0.280	0.582	1.244	0.219	0.9395	0.7705
$\text{Trans+VR}_{\text{RoBERTa-Object}}$	0.792	0.644	0.503	0.386	0.281	0.582	1.242	0.219	0.9396	0.7701
$\text{Trans+VR}_{\text{BERT-Multi-class}}$	**0.794**	0.645	0.503	0.385	0.281	0.582	1.248	**0.220**	0.9395	**0.7717**
$\text{Trans+VR}_{\text{BERT-Multi-object}}$	0.792	0.644	0.502	0.385	0.281	0.582	1.244	**0.220**	0.9395	0.7693
$\text{Trans+VR}_{\text{RoBERTa-Multi-class}}$	0.791	0.643	0.503	0.385	0.280	0.581	1.244	0.219	0.9395	0.7710
$\text{Trans+VR}_{\text{RoBERTa-Multi-object}}$	0.791	0.643	0.502	0.385	0.281	0.582	1.243	0.219	0.9395	0.7712
$\text{Trans+VR}_{\text{RBERTa-Class+Object}}$	0.793	0.645	0.504	0.387	0.281	0.582	1.247	0.220	0.9397	0.7705
$\text{Trans+VR}_{\text{BERT-Class+Object}}$	0.793	0.645	**0.505**	**0.388**	**0.282**	**0.583**	**1.251**	**0.220**	**0.9399**	0.7695

Table 8: **Positive Evidence: full result with all evaluation metrics**. Performance of compared baselines on the Karpathy test split with/without semantic re-ranking. For each base system, we report performance using a greedy search and the best beam search. Re-ranking is applied to the top-20 results of each system using BERT or RoBERTa for caption-context similarity. The visual contexts are extracted using ResNet152 and the Inception Resnet v2 based Faster R-CNN **object** detector. We also report results for Bert-based similarity without a hypothesis probability (rows marked *only sim*).

VC_1	VC_2	VC_3	Caption
cheeseburger	plate	hotdog	a plate with a hamburger fries and tomatoes
bakery	dining table	web site	a table having tea and a cake on it
gown	groom	apron	its time to cut the cake at this couples wedding
racket	scoreboard	tennis ball	a crowd is watching a tennis game being played
laptop	screen	desktop computer	a grey kitten laying on a windows laptop
washbasin	toilet	seat tub	a bathroom toilet sitting on a stand next to a tub and sink

Table 9: **Training Dataset**. The visual context (VC) is from a pre-trained visual classifier (*i.e*. ResNet152) and the caption is from COCO-Caption dataset (human-annotated).

We randomly selected 26 test images and gave 12 reliable human subjects the option to choose between two captions: (1) Best-beam (BeamS) and (2) **V**isual **R**-ranker as shown in Figure 10.

Also, inspired by BERTscore and following (Song et al., 2021) that introduce Sentence Semantic Semantic (SSS) for machine translation, we employ a sentence-to-sentence semantic similarity score to compare candidate captions with human references. We use pre-trained Sentence-$\text{RoBERTa}_{\text{LARGE}}$ tuned for general STS-B task. Consequently, the embedding will be more robust semantically than lexically for the STS tasks. Figure 9 shows that our results with SBERT-sts agrees more with human judgment than the BERTscore. Figure 11 and Figure 12 show some examples of when humans agree/disagree with our re-ranker.

Model	B-1	B-2	B-3	B-4	M	R	C	S	BERTscore	SBERT-sts
VilBERT (Lu et al., 2020)										
$\text{Vil}_{\text{Greedy}}$	0.751	0.587	0.441	0.330	0.272	0.554	1.104	0.207	0.9352	0.7550
$\text{Vil}_{\text{BeamS}}$	0.752	0.592	0.456	0.351	0.274	0.557	1.115	0.205	0.9363	0.7550
$\text{Vil+VR}_{\text{W-Object}}$ (Fang et al., 2015)	**0.756**	**0.595**	0.456	0.348	0.274	**0.559**	1.123	0.206	0.9365	**0.7605**
$\text{Vil+VR}_{\text{Object}}$ (Wang et al., 2018)	**0.756**	0.594	0.455	0.348	0.274	**0.559**	1.120	0.206	0.9364	0.7570
$\text{Vil+VR}_{\text{Control}}$ (Cornia et al., 2019)	0.753	0.591	0.453	0.345	0.274	0.557	1.116	0.206	0.9361	0.7565
$\text{Vil+VR}_{\text{RoBERTa}}$ Table 1 (positive)	0.753	0.594	**0.458**	**0.353**	**0.276**	**0.559**	1.128	0.207	**0.9366**	0.7562
$\text{Vil+VR}^{-low}_{\text{RoBERTa}}$	0.748	0.588	0.453	0.349	0.275	0.557	1.116	0.206	0.9362	0.7531
$\text{Vil+VR}^{-high}_{\text{RoBERTa}}$	0.748	0.588	0.453	0.349	0.275	0.557	1.116	0.206	0.9364	0.7546
$\text{Vil+VR}^{-pos}_{\text{GloVe}}$	0.751	0.591	0.455	0.351	0.276	0.558	1.123	0.207	0.9364	0.7556
$\text{Vil+VR}^{-joint}_{\text{RoBERTa+GloVe}}$	0.750	0.591	0.455	0.351	**0.276**	**0.559**	**1.126**	**0.208**	**0.9365**	0.7548
Transformer Caption Generator (Cornia et al., 2020)										
$\text{Trans}_{\text{Greedy}}$	0.787	0.634	0.488	0.368	0.276	0.574	1.211	0.215	0.9376	0.7649
$\text{Trans}_{\text{BeamS}}$	0.793	0.645	0.504	0.387	0.281	0.582	1.247	**0.220**	**0.9399**	0.7707
$\text{Trans+VR}_{\text{W-Object}}$ (Fang et al., 2015)	0.786	0.638	0.497	0.378	0.277	0.579	1.228	0.216	0.9388	0.7709
$\text{Trans+VR}_{\text{Object}}$ (Wang et al., 2018)	0.790	0.642	0.501	0.383	0.280	0.580	1.237	0.219	0.9391	0.7700
$\text{Trans+VR}_{\text{Control}}$ (Cornia et al., 2019)	0.791	0.644	0.505	**0.388**	0.281	**0.583**	1.248	**0.220**	0.9398	**0.7716**
$\text{Trans+VR}_{\text{BERT}}$ Table 1 (positive)	0.793	**0.646**	**0.505**	**0.388**	**0.282**	**0.583**	**1.250**	**0.220**	**0.9399**	0.7711
$\text{Trans+VR}^{-low}_{\text{BERT}}$	0.791	0.643	0.504	0.387	0.280	0.582	1.242	0.218	0.9396	0.7682
$\text{Trans+VR}^{-high}_{\text{BERT}}$	0.793	0.644	0.503	0.385	0.282	0.582	1.243	0.219	0.9397	0.7686
$\text{Trans+VR}^{-pos}_{\text{GloVe}}$	**0.794**	**0.646**	**0.506**	**0.388**	**0.282**	**0.583**	**1.249**	**0.220**	**0.9399**	0.7702
$\text{Trans+VR}^{-joint}_{\text{BERT+GloVe}}$	0.793	0.645	0.504	0.387	0.281	0.582	1.247	**0.220**	0.9398	0.7704

Table 10: **Negative Evidence: full result with all evaluation metrics**. Comparison results between positive Belief Revision (single concept VR) (gray color) and Negative Belief Revision (NBR) on the Karpathy test split. The NBR uses a *high similarity* VR^{-high} object related to the positive visual but not in the image, *low similarity* VR^{-low} uses false positive from the visual classifier, and positive visual via static word level similarity VR^{-pos}. **Boldface** fonts reflect the improvement over the baseline.

Model	B-1	B-2	B-3	B-4	M	R	C	S	BERTscore	SBERT-sts
VilBERT-VR-GPT-2_{mean} + ResNet										
+ RoBERTa (Table 1 Best)	**0.753**	**0.594**	**0.458**	**0.353**	0.276	0.559	1.128	0.207	0.9366	0.7562
+ LM-GPT-2_{product}	0.749	0.590	0.455	0.351	0.276	0.558	1.124	0.208	0.9364	0.7486
+ DistilSBERT	0.751	0.591	0.456	0.352	**0.277**	0.559	1.130	0.209	0.9365	0.7567
+ SimCSE-BERT (Gao et al., 2021)	0.752	0.593	0.457	0.352	0.276	0.559	1.130	**0.209**	0.9365	0.7558
+ SimCSE-RoBERTa	0.750	0.590	0.455	0.351	0.276	0.558	1.125	0.208	0.9365	0.7549
+ SimCSE-BERT-V_1 (unspervised)	0.750	0.591	0.455	0.351	0.276	0.558	1.128	0.207	0.9365	0.7560
+ SimCSE-BERT-V_2 (unspervised)	0.752	0.593	0.457	**0.353**	**0.277**	0.559	1.132	0.208	0.9365	0.7560
+ CLIP-V (Radford et al., 2021)	**0.753**	**0.594**	**0.458**	**0.353**	0.276	**0.561**	**1.131**	0.208	**0.9367**	**0.7579**
Trans - VR-GPT-2_{mean} + ResNet										
+ BERT (Table 1 Best)	0.793	**0.646**	**0.505**	**0.388**	**0.282**	**0.583**	**1.250**	**0.220**	**0.9399**	**0.7711**
+ LM-GPT-2_{product}	0.787	0.642	0.503	0.386	0.279	0.581	1.236	0.219	0.9398	0.7683
+ DistilSBERT	**0.794**	**0.646**	**0.505**	0.387	**0.282**	**0.583**	1.247	**0.220**	0.9396	0.7704
+ SimCSE-BERT (Gao et al., 2021)	0.792	0.644	0.503	0.386	0.281	0.581	1.243	0.219	0.9394	0.7694
+ SimCSE-RoBERTa	**0.794**	0.645	0.504	0.387	0.281	0.582	1.244	0.219	0.9395	0.7698
+ SimCSE-V_1 (unspervised)	0.792	0.645	0.504	0.386	0.281	0.582	1.244	0.219	0.9397	0.7705
+ SimCSE-V_2 (unspervised)	0.792	0.645	**0.505**	0.387	0.281	0.582	1.247	0.219	0.9396	0.7703
+ CLIP (Radford et al., 2021)	0.791	0.643	0.503	0.386	0.280	0.581	1.242	0.219	0.9395	0.7703

Table 11: **Full Ablation Study**. We experimented using different information from various baselines on the Karpathy test split. Also, with the unsupervised BERT similarity, we tried with the top-2 visual context from the classifier.

C Hyperparameters and Setting

All training and the beam search are implemented with PyTorch 1.7.1 (Paszke et al., 2019). VR based $\text{BERT}_{\text{base}}$ is fine-tuned on the training dataset using the original BERT implementation, Tensorflow version 1.15 with Cuda 8 (Abadi et al., 2016) (hardware: GPU GTX 1070Ti and 32 RAM and 8-cores i7 CPU). The textual information dataset consists of 460k captions, 373k for training, and 87k for validation *i.e.* visual, caption, label ([semantically related or not related]). We use a batch size of 16 for two epochs with a learning rate $2\text{e}{-5}$, we kept the rest of hyperparameters settings as the original implementation.

D Additional Examples

We provide more comparison results with examples, including sentence-level evaluation SBERT-sts and BERTscore in Table 12. Also, in Figure 11, and Figure 12, we also evaluated our re-ranker using human subjects.

Table 12: Examples show caption re-ranked by our **V**isual **R**e-ranker with different evaluation metrics including the semantic-similarity based metrics. Note that, we only report B-1 and B-2, from BLEU, to measure the word level changes before and after re-ranking.

Model	caption	B1	B2	M	R-L	S	BERTscore	SBERT-sts
BeamS	a woman holding a tennis racquet on a tennis court	**0.54**	**0.40**	0.26	0.47	0	0.89	**0.73**
VR	a woman **standing** on a tennis court holding a racque	0.53	0.32	**0.27**	**0.49**	**0.30**	**0.93**	0.68
Refe	a woman in a short bisque skirt holding a tennis racque							
BeamS	a white train is at a train station	0.29	0.18	0.11	0.32	0.22	0.90	0.57
VR	**a train on the tracks** at a train station	**0.49**	**0.40**	**0.23**	**0.52**	**0.53**	**0.91**	**0.78**
Refe	a train that is sitting on the tracks under wires							
BeamS	a pair of black scissors on a white wall ✗	0.31	0	0.12	0.37	0	0.87	0.23
VR	a flower in a vase next to a pair of scissors ✗	**0.41**	**0.19**	**0.15**	**0.43**	0.42	**0.89**	**0.27**
Refe	a dried black flower in a long tall black and white vase							
BeamS	a man sitting on a bench	0.43	0.43	0.32	0.67	0.50	0.96	0.66
VR	a man sitting on a bench **talking on a cell phone**	**0.90**	**0.85**	**0.95**	**0.90**	**0.90**	**0.98**	**0.99**
Refe	a man sitting on a bench talking on his cell phone							
BeamS	a woman standing in an airport with luggage	0.42	0.35	0.25	0.51	0.30	**0.92**	0.67
VR	a woman standing in a luggage carousel at an airport ✗	0.54	0.40	0.25	0.56	0.50	0.89	0.68
Refe	a woman standing in front of a bench covered in luggage							
BeamS	an airplane sitting on a runway behind a fence	**0.39**	**0.29**	0.22	0.41	0.40	0.92	0.69
VR	an airplane is parked behind a fence	0.37	0.28	**0.25**	**0.45**	**0.50**	**0.94**	**0.85**
Refe	the airplane has landed behind a fence with barbed wire							
BeamS	a plate with a sandwich on a table	0.55	0.26	0.19	**0.46**	0.40	0.91	0.86
VR	**a white plate** with a sandwich on a table	**0.66**	**0.40**	**0.24**	0.44	**0.54**	**0.92**	**0.90**
Refe	a small sandwich sitting on a white china plate							
BeamS	three giraffes standing in a field under a tree	**0.26**	0	**0.14**	**0.29**	**0.26**	**0.91**	0.62
VR	a group of giraffes standing in a field	0.17	0	0.10	0.20	0	0.90	**0.64**
Refe	two tall giraffe standing next to a green leaf filled tree							
BeamS	two parking meters in front of a brick wall	**0.33**	0.20	0.15	0.22	0.25	0.47	0.87
VR	a row of parking meters in front of a building	0.30	**0.25**	**0.17**	**0.31**	0.25	**0.60**	**0.88**
Refe	different types and sizes of parking meters on display							
BeamS	a bathroom with a toilet and a mirror	0.33	0	0.10	**0.34**	0	0.89	0.69
VR	a variety of items on display in a bathroom	**0.44**	0	**0.12**	0.33	**0.14**	**0.90**	**0.70**
Refe	a view of a couple types of toilet items							
BeamS	two bulls with horns standing next to each other	**0.44**	**0.23**	0.16	**0.47**	**0.66**	**0.89**	0.76
VR	two long horn bulls standing next to each other	0.33	0	**0.18**	0.23	0.25	0.88	**0.81**
Refe	closeup of two red-haired bulls with long horns							
BeamS	a laptop computer sitting on top of a desk	0.44	0.21	0.20	0.29	0.18	0.91	0.69
VR	a desk with a laptop **and a computer monitor**	**0.53**	**0.51**	**0.31**	**0.58**	**0.46**	**0.95**	**0.77**
Refe	an office desk with a laptop and a phone on it							
BeamS	a busy highway with cars and a train	**0.33**	**0.20**	**0.12**	**0.34**	**0.18**	**0.90**	**0.43**
VR	cars are driving on a highway under a bridge ✗	0.22	0	0.04	0.22	0	0.88	0.21
Refe	a photo of a train heading down the tracks							
BeamS	a baby sitting in front of a cake	0.44	0.23	0.18	**0.46**	**0.36**	0.90	**0.81**
VR	a baby sitting in front of a **birthday cake**	0.44	0.23	0.18	0.44	0.33	0.90	0.77
Refe	a baby in high chair with bib and cake							
BeamS	a dog sitting on a bed with clothes	**0.58**	**0.44**	**0.29**	**0.65**	0.40	**0.93**	0.57
VR	a dog sitting on a bed **next to** clothes	0.49	0.33	0.25	0.52	0.40	0.91	**0.61**
Refe	a dog is sitting on an unmade bed with pillows							
BeamS	a group of boats docked in the water	0.19	0	0.07	**0.21**	0.22	0.88	0.58
VR	a group of boats **are** docked in the water	0.19	0	0.07	0.20	0.22	0.88	**0.59**
Refe	looking out over a bay with many tourist boats moored							

Model	Caption	BERTscore	SBERT-sts	Human%	Visual
BeamS	a close up of a plate of food	0.89	0.27	40	trifle
VR	piece of food sitting on top of a white plate	**0.91**	**0.53**	**60**	
Human refe	a white plate and a piece of white cake				
BeamS	a group of men on a field playing baseball	0.88	0.58	33.3	baseball
VR	a batter catcher and umpire during a baseball game	**0.91**	**0.84**	**66.7**	
Human refe	batter catcher and umpire anticipating the next pitch				
BeamS	a couple of airplanes that are flying in the sky	0.88	0.03	0	traffic light
VR	a group of traffic lights at an airport	**0.95**	**0.71**	**100**	
Human refe	a group of traffic lights sitting above an intersection				
BeamS	two bulls with horns standing next to each other	**0.89**	0.76	16.7	ox
VR	two long horn bulls standing next to each other	0.88	**0.81**	**83.3**	
Human refe	closeup of two red-haired bulls with long horns				
BeamS	a woman holding a tennis racquet on a tennis court	0.89	**0.73**	**85.3**	racket
VR	a woman standing on a tennis court holding a racquet	**0.93**	0.68	16.7	
Human refe	a woman in a short bisque skirt holding a tennis racquet				
BeamS	a white train is at a train station	0.90	0.57	50	↯ locomotive
VR	a train on the tracks at a train station	**0.91**	**0.78**	50	
Human refe	a train that is sitting on the tracks under wires				
BeamS	two men cutting a cake at ceremony	**0.94**	**0.98**	**66.7**	≈ mortarboard
VR	a group of military men cutting a cake	0.80	0.91	33.3	
Human refe	two men are cutting a cake at a function				
BeamS	a little girl wearing a tie and pants	**0.94**	**0.80**	**66.7**	≈ feather boa
VR	a little girl wearing a tie standing in a room	0.93	0.77	33.3	
Human refe	a young girl wearing a tie that matches her skirt				
BeamS	a man laying on the ground with many animals	0.88	0.14	0	✗ trilobite
VR	a man kneeling down in front of a herd of sheep	**0.89**	**0.56**	**100**	
Human refe	a view of a bunch of sheep lined up with a behind them				
BeamS	a kitchen with black counter tops and wooden cabinets	0.88	**0.44**	**100**	✗ barbershop
VR	a kitchen counter with a black counter top	0.88	0.40	0	
Human refe	a kitchen with a sink bottles jars and a dishwasher				

Figure 11: Examples show caption re-ranked by our **V**isual **R**e-ranker and the original baseline Best beam. An evaluation metrics comparison between semantic-similarity based SBERT-sts, BERTscore, and the human subject.

Model	Caption	BERTscore	SBERT-sts	Human%	Visual
BeamS	a red and white boat in the water	**0.94**	**0.78**	**87.5**	fireboat
VR	a red and white boat is in the water	0.93	0.79	12.5	
Human refe	a red and white boat floating along a river				
BeamS	a dog sitting on a bed with cloths	**0.93**	0.57	50	dog/Irish terrier
VR	a dog sitting on a bed next to clothes	0.91	**0.61**	50	
Human refe	a dog is sitting on an unmade bed with pillows				
BeamS	a laptop computer sitting on top of a desk	0.91	0.69	25	desk
VR	a desk with a laptop and computer monitor	**0.95**	**0.77**	**75**	
Human refe	an office desk with a laptop and computer monitor				
BeamS	a bathroom with a toilet and a mirror	0.89	0.69	12.5	washbasin
VR	a variety of item on display in a bathroom	**0.90**	**0.70**	**83.3**	
Human refe	a view of a couple types of toilet items				
BeamS	a couple of pizzas that are on a table	**0.88**	0.75	**100**	pizza
VR	a couple of pizzas are sitting on a table	0.87	**0.77**	0	
Human refe	pizza on a table with a cup and a fork				
BeamS	a group of people in a living room playing a video game	0.93	0.47	**62.5**	television
VR	a group of people sitting in a living room playing a video game	**0.94**	**0.50**	37.5	
Human refe	a group of friends sitting inside their living room				
BeamS	a city street at night with traffic lights	**0.97**	0.81	33.3	traffic light
VR	a city street at night with cars and a traffic light	0.94	**0.85**	**66.7**	
Human refe	a city street at night filled with lots of traffic				
BeamS	a close up of a cat eating a doughnut	**0.83**	**0.90**	**100**	pretzel
VR	a close up of a person holding a doughnut	0.60	0.88	0	
Human refe	a cat bites into a doughnut offered by a persons hand				
BeamS	two men standing next to a group of people	**0.88**	0.20	**87.5**	groom
VR	a group of men standing next to each other	0.86	**0.30**	12.5	
Human refe	the man is holding his tie with his right hand				
BeamS	a pile of trash sitting inside of a building	0.88	**0.38**	**100**	✗ vacuum
VR	a pile of trash sitting in front of a building	0.88	0.27	0	
Human refe	an older floor light sits deserted in an abandoned hospital				

Figure 12: Examples show caption re-ranked by our **V**isual **Re**-ranker and the original baseline Best beam. An evaluation metrics comparison between semantic-similarity based SBERT-sts, BERTscore, and the human subject.

Towards Understanding the Relation between Gestures and Language

Artem Abzaliev
University of Michigan
abzaliev@umich.edu

Andrew Owens
University of Michigan
ahowens@umich.edu

Rada Mihalcea
University of Michigan
mihalcea@umich.edu

Abstract

In this paper, we explore the relation between gestures and language. Using a multimodal dataset, consisting of TED talks where the language is aligned with the gestures made by the speakers, we adapt a semi-supervised multimodal model to learn gesture embeddings. We show that gestures are predictive of the native language of the speaker, and that gesture embeddings further improve language prediction result. In addition, gesture embeddings might contain some linguistic information, as we show by probing embeddings for psycholinguistic categories. Finally, we analyze the words that lead to the most expressive gestures and find that function words drive the expressiveness of gestures. Our code is available at https://github.com/MichiganNLP/gestures-language.

1 Introduction

Gestures are often referred to as "non-verbal language" and extensive studies in psychology, sociology, and anthropology have demonstrated the important role they play in communication (McNeill, 1992; Iverson and Goldin-Meadow, 1998; Alibali et al., 2000). While language and gesture can occur independently, people often use them together to communicate, suggesting that multimodality plays an important role in understanding gestures. In this work, we consider human gestures together with their corresponding utterances. We jointly learn gesture and word embeddings, and attempt to predict psycholinguistic categories and the language of the speaker from their gesture embeddings.

Even for humans it is very challenging to predict words from gestures alone (or vice versa), due to the many-to-many relationship between words and gestures. Therefore, instead of directly predicting one modality from the other (Desai and Johnson, 2021), we use contrastive pre-training learning to learn a joint embedding space that aligns both modalities (Kiros et al., 2014; Tian et al., 2020; Radford et al., 2021). This allows our model to learn an association between language and gestures, despite a large amount of uncertainty inherent in the task.

The main contributions of this work are as follows:

- First, we explore a multimodal approach to learn gesture embeddings through contrastive learning. Through validation experiments relying on these embeddings, we demonstrate that there is an association between gestures and languages representations.
- Second, we probe gesture embeddings for various psychological and linguistic categories and show that gestures can be predictive of several categories with better-than-random accuracy. We find that function words, such as pronouns, preposition or modal verbs, can be predicted from the gestures. We also show that gesture embeddings can be used to predict discourse markers.
- Third, we show that it is possible to predict the language of a speaker from our learned gestures embeddings. Our findings indicate that the difference in gestures across the languages may be driven by the function words.
- Finally, we conduct several analyses to better understand the learned gesture representations.

2 Related Work

Semi-supervised Multimodal Learning. Our work builds on the idea of multimodal learning, where a model is trained to represent several modalities in a shared embedding space (Chen et al., 2019; Li et al., 2019; Lu et al., 2019; Tan and Bansal, 2019). In particular, we focus on semi-supervised multimodal learning (Yuan et al., 2021; Wu et al., 2021; Zhai et al., 2021), which is effective and useful training strategy for settings where obtaining labeled training data is laborious or prohibitive. We base our model on CLIP (Radford et al., 2021), which uses a large amount of (multi-

Proceedings of the 29th International Conference on Computational Linguistics, pages 5507–5520
October 12–17, 2022.

modal) unlabeled data combined with efficient pretraining objective leading to strong zero-shot performance in both language and vision tasks.

Pose Estimation and Gesture Understanding. While most of the recent multimodal work has focused on modalities such as vision (Morency et al., 2007), language, or speech (Levine et al., 2009; Ginosar et al., 2019), there are also studies that highlight the importance of gestures for various aspects of human activities; see Kelly et al. (2008) for an overview.

Work in this space has addressed, among other tasks, gesture recognition (Zhang et al., 2020), gesture generation (Kucherenko et al., 2020b; Ferstl et al., 2021; Yoon et al., 2020; Kucherenko et al., 2020a; Alexanderson et al., 2020), gesture-to-gesture generation (Tang et al., 2019). All of these tasks, while close to our problem space, are not directly applicable for gesture representation.

Another line of work focuses on the temporal alignment and interaction of speech and gestures. Loehr (2007), shows that speech and gestures occur synchronously. (Rieser, 2015) shows that utterances and gestures are not always synchronous and suggests using λ-π calculus to model them. (Saint-Amand et al., 2013) provide a comprehensive study of the alignment of speech and gestures in a constraint-based grammar, while (Lücking et al., 2013) show that gestures follow the language but the opposite does not hold.

An important question is how to obtain labels that describe gesture. We use pseudo ground-truth pose estimates from OpenPose (Cao et al., 2019). While even state-of-the-art gesture recognition systems can be noisy, this noise is significantly reduced on videos such as TED talks (Yoon et al., 2018) given that there is only one speaker and have good light conditions. For a comprehensive overview of recent progress in the field of pose estimation see Munea et al. (2020).

3 Data

Our primary source of data is the YouTube Gesture Dataset (Yoon et al., 2018). The dataset consists of over 1,500 TED talk videos of English speakers addressing various topics like science, medicine, society, and others. The camera is usually in front of the speaker, so the gestures are visible. The dataset contains precomputed key points for the head, neck, shoulders, elbows, and wrists, with each pose represented as a 16-dimension vector, with x and y coordinates for each key point. The dataset includes subtitles, auto-generated by YouTube and aligned by gentle [1].We also use an additional dataset that we compiled ourselves, consisting of 600 videos of Spanish speakers. We process videos from the TEDx channel using the playlist "TEDx talks en Español." Subtitles for Spanish data are auto-generated and aligned by YouTube, so we just download the appropriate subtitle file. Figure 1 shows an example of gestures aligned with the corresponding language.

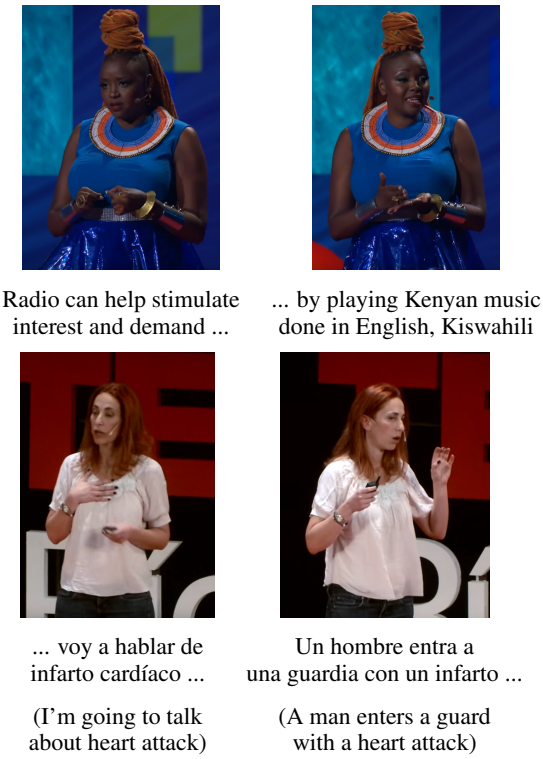

Figure 1: Examples of gestures aligned with the corresponding language. All videos and corresponding utterances are one second in length. Top row: English speaker. Bottom row: Spanish speaker.

We split each video into several clips using PySceneDetect,[2] which detects changes in the camera angle during the talk and splits a single video into several sub-clips. This is necessary so that the pose movements remain continuous during the short clips, even if the camera angle is changed.

Table 1 shows the summary statistics of the dataset. The numbers differ from those reported in Yoon et al. (2018) because we used a more ag-

[1] https://github.com/lowerquality/gentle

[2] https://github.com/Breakthrough/PySceneDetect

	English	Spanish
Videos train	1,349	543
Videos val	167	64
Clips train	127,003	61,810
Clips val	16,813	9,299
Average # words val	3.74	3.4
Average # words train	3.75	3.37
Duration train	35.27h	17.16h
Duration val	4.67h	2.58h

Table 1: Dataset statistics

gressive video filtering strategy,[3] to make sure that only clips with enough gesture variety remain in the dataset.

4 Model

4.1 Input Representation

One of the key considerations for our approach is how to represent gestures and language as input to our model. We start by dividing each clip into a series of gestures using the sliding window approach of Yoon et al. (2018). We cut every clip into 1 second sequences of frames; with 15 frames per second, our input becomes 15 frames. Detailed per-millisecond word alignments are available in our datasets.

We align each one second interval with the corresponding phrase. Our method requires that gestures and language are timely-aligned within some interval of t seconds, where we use t=1 second. It does not require them to be aligned exactly, as the pose encoder and gesture encoder use different positional embeddings. One limitation of such an approach is that longer gestures are truncated, and very short gestures are collapsed together. We experimented with several lengths of the sliding window and found out that the choice of the time interval does not affect the overall performance.

Another possibility would be to split the utterances by word and take all the corresponding frames that fall within the given time interval. For instance, we can take the word 'hello' from the subtitles, and get all the corresponding frames while the word is pronounced. This way we guarantee that there is no overlap between the gestures, and a single word corresponds to a single series of gestures. However, previous results in the literature McNeill (2005) indicate that there are different gesture phases, and they are not necessarily timely aligned with the words. In such a case this approach would be limited. We experimented with such a setting as well, but the resulting performance is only marginally better than random.

4.2 Approach

Figure 2 shows the overview of our approach. After preprocessing, poses and phrases are passed through two separate encoders. We use a transformer architecture to separately encode the text and the poses. The pose encoder model very closely follows the CLIP's base image encoder: it is a 12-layer 768-wide model with 12 attention heads. We have to adjust the width from 512 to 768 to match the size of the text model, which is necessary for cosine similarity. The pose encoder is randomly initialized and takes as input a tensor of size $(15, 16)$ where 15 is the number of frames in a 1 sec. clip and 16 is the joint dimension. This input gets transformed to $(15, 768)$ with the fully connected layer and is passed directly to the attention block, bypassing the input embedding layer. This is possible because the pose is already represented as a vector, and does not have to be embedded. We use the last frame as an end-of-sentence token for the prediction. On top of the transformer, there is a 768×768 projection layer. We use the multilingual XLM-RoBERTa (Conneau et al., 2019) as a pre-trained encoder for language with another projection layer on top of the encoder.

After encoding the pose and language into vectors of fixed length, they are normalized and the dot product is taken separately for each modality. The Multi-class N-pair loss (Sohn, 2016) objective is used to learn the match between the poses and the corresponding utterances in a single batch. We selected a batch of size two to make the training task easier. While contrastive learning benefits from large batch sizes (Newcombe, 2018), we found that in our case the higher the batch size, the harder it is for the model to learn. We tried batch sizes 8, 16, and 32, and the results were worse. We hypothesize this is due to a large amount of noise in the data. We use AdamW with a learning rate of $1e-5$ as an optimizer, and cosine schedule as a learning rate schedule.

One possible concern for our approach is the size of the training dataset. CLIP uses more than 350 million image-text pairs, while our dataset in-

[3] We used a threshold of 250 for circular variance, compared to the original value of 150

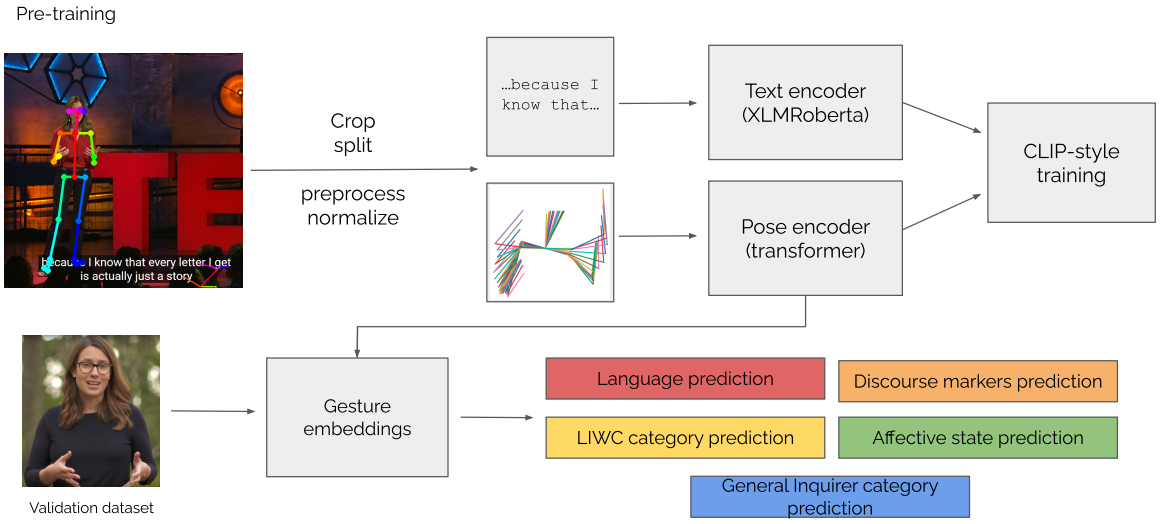

Figure 2: Overview of our approach: we train joint pose and language encoders on training data and produce embeddings. We show we can predict the native language of the speaker and several linguistic categories from the gesture embeddings alone.

cludes only 180,000 1-second clips. To mitigate this, we first conduct an experiment where we show that the proposed pose encoder can predict whether the source of motion was left or right hand. Second, we use a pre-trained text encoder, namely XLMRoberta, as a text encoder. This is in contrast to CLIP, where the authors train the model from scratch. This way we only train a pose encoder, while the text encoder is only fine-tuned. Third, contrary to images that are represented as a 336x336 matrix, poses have much lower dimensionality, namely, our input has a dimension of 15x16. Our intuition is that these factors combined can substantially reduce the required amount of training data.

4.3 Alignment Validation

To make sure that the model is capable of learning gesture–language alignments, we conduct two simple experiments. We aim to verify whether a pose encoder can associate many similar (but with a substantial degree of variety) gestures with a single word/phrase, given the limited amount of training data and proposed model architecture. In other words, before learning a many-to-many mapping (many possible gestures can correspond to the same word, and many possible words can correspond to the same gesture), we want to verify that one-to-many mapping is even possible.

In the first validation experiment, from our dataset, we select only the poses where the source of motion is either the left or the right hand. We do this by calculating the circular variance (Pedregosa et al., 2011) of the angles between joints on the right side and the left side. Only the clips where the circular variance is above 750 on one side and less than 100 on the other side are selected. For these clips, we artificially insert the words ‘left’ or ‘right’ at random position in the existing utterance, depending on which side has is a high variance. This process resulted in 12,287 pose sequences with right-hand movement and 11,846 with left-hand movements.

We set the batch size equal to two and include only one left and one right pose in each batch so that it can be matched with the corresponding text in only one correct way. The resulting accuracy on the validation set is 99.93% and 100% for poses and language respectively. This experiment suggests that learning gesture-text alignments is possible when the gestures have sufficient expressiveness.

In the second validation experiment, given an input gesture and its embedding, we use a simple cosine similarity measure to find the closest text embedding. For each (gesture, text) pair from the validation set, we pick another random pair and

Class	Embeddings	Raw poses	# Observations	Description and Examples
PREPS	51.8 (±2.31)	51.1 (±2.18)	12978.4	Prepositions: to, with, above
PRONOUN	52.5 (±2.94)	52.3 (±2.83)	10235.0	Pronouns: I, them, itself
DISCREP	52.8 (±6.16)	51.6 (±5.79)	2343.2	Discrepancy: should, would, could

Table 2: Accuracy on the LIWC category prediction task for **English**. The accuracy of the majority baseline is 50%. We show the categories where the accuracy of the embeddings model is significantly larger than the majority model at a 0.05 significance level. We use Wilcoxon signed-rank test for significance testing. ± denotes standard deviation

Class	Embeddings	Raw poses	# Observations	Description and Examples
PREPS	52.2 (±3.51)	51.2 (±3.29)	6308.4	Prepositions: sin, con, acerca

Table 3: Accuracy on the LIWC category prediction task for **Spanish**. The accuracy of majority baseline is 50%. We show the categories where the accuracy of the embeddings model is significantly larger than the majority model at a 0.05 significance level, using a Wilcoxon signed-rank test for significance testing.

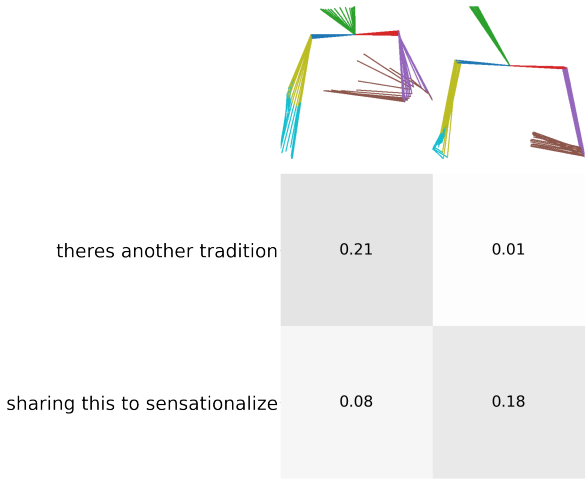

Figure 3: Representation of the input. The main diagonal represents the ground truth.

calculate the cosine similarity between each gesture and text, so we end up with 4 similarity scores, 2 for each gesture (or correspondingly 2 for each text). For each gesture, we take the text with the highest score as a prediction and compare the text we find using the similarity against the correct text paired with the gesture in the data. Figure 3 shows an example.

On the validation dataset, this experiment leads to 65.4% accuracy. When reversed, i.e., starting with a text embedding, we find the most similar gesture embedding according to a cosine similarity and compare it against the gold standard, we achieve 64.9% accuracy. This performance significantly higher than the random baseline indicates that the learned representations contain useful information.

5 Experiments and Results

To evaluate the strength of the connection between gestures and language, we perform two types of experiments. First, we perform experiments within one single language only, i.e. how gestures and language interact in either English or Spanish. The second type is cross-language, how gestures are different among English and Spanish speakers.

5.1 Single Language Experiments

We aim to predict a psycholinguistic category of utterance from the gesture embeddings obtained with the model described in Section 4. The motivation for this type of analysis is that humans might be able to understand that the person is, for instance, angry from the pose alone. We used Linguistic Inquiry and Word Count (LIWC) lexicon (Pennebaker et al., 2007) and General Inquirer categories (Stone et al., 1966) to map the utterances to their categories. Namely, if the text contained any word from LIWC or General Inquirer category, we considered the whole utterance to belong to this category, i.e. label y=1, and y=0 otherwise. Some utterances can have more than one category. In total, we run 146 binary classification problems (65 LIWC categories and 81 General Inquirer categories).[4] We compare our results with a majority baseline, as well as a raw pose baseline, where we fit the logistic classifier on the vector of joints (15 frames with a 16-dimensional vector on each, flattened into the 240-dimensional vector). We use

[4]We discard all the categories that have less than 30 observations in the validation dataset.

30-fold[5] cross-validation on the validation dataset, stratified by the language variable, and grouped by the video id (i.e., several clips from the same video should only be in either the training data or the validation data, to exclude the possibility of data contamination). At each training and test fold, we additionally sub-sample English clips to make the number of English and Spanish clips equal. Therefore the accuracy of the majority baseline is always 50%. For the prediction, we use the same model as Conneau and Kiela (2018), a logistic classifier with the default parameters. To verify that the accuracy of one model is larger, we performed a one-sided Wilcoxon paired signed-rank test (Wilcoxon, 1945) on the accuracy scores from cross-validation. We decided to use this test based on the results from Demšar (2006). The null hypothesis is that the accuracies of two classifiers are the same, and the alternative hypothesis is that the embeddings/raw model is larger than the raw/majority.

We rerun our experiment with 30 different random seeds. Table 2 shows the LIWC categories where the embeddings model significantly outperforms the baseline (majority) model for a 90% of the 30 runs. [6] Interestingly, the resulting categories belong to function words. This finding indicates that gestures accompanying function words may have a more apparent visual appearance, compared to the other words. This finding extends previous work from psychology pointing to the importance of function words in communication (Chung and Pennebaker, 2007). Table 3 shows the same type of analysis for Spanish language. There is only one category overall, prepositions, also part of the function words. Table 4 presents the results for the General Inquirer categories. In addition to the pronouns, active verbs show a strong connection with gestures.

Another type of analysis we conducted is predicting the use of discourse markers in speech from the gestures. We use a list of words from Discsense (Sileo et al., 2020). Table 5 shows the results. The embedding model is significantly better than both the raw poses model and the majority model, suggesting that joint language-vision learning is beneficial for this task. We also attempt to predict Valence-Arousal-Dominance states from (Mohammad, 2018), but neither raw poses nor embedding model could predict better than the majority baseline.

These results are in line with the findings reported in Lücking et al. (2013), where authors conducted the experiments using a dataset with manually annotated alignments between gestures and phrases, and found that prepositional phrases are associated with gestures as well.

An important observation from these results is that in many cases a classifier relying only on the raw poses significantly outperforms the majority baseline, suggesting that gestures by themselves contain information about language. Additionally, this finding is further supported by the improvements obtained with the gesture embeddings, which show that the joint learning of gestures and language is beneficial.

5.2 Experiments with English and Spanish

If gestures are indeed closely related to the corresponding language, we hypothesize that we should be able to predict the language of a speaker (e.g., English or Spanish) from the gestures alone. Table 7 shows the language prediction results using the gesture embeddings. We use the identical cross-validation with the sub-sampling scheme as described in Section 5.1.

To further investigate which gestures lead to better-than-random accuracy on the language prediction task, we use the LIWC lexicon to identify the word categories that have the highest improvement with respect to the majority baseline. This analysis can be interpreted as follows: "While a person is using word category X, the gestures of an English speaker can be more easily distinguished from those of a Spanish speaker." Table 6 shows the accuracy of the language prediction task split by the LIWC categories. We select the poses that have the highest probability to be predicted correctly, and extract their corresponding utterances. From these utterances, we extract the LIWC category for the words, and calculate the accuracy separately for each word category. There are eleven categories where the embedding model outperforms the raw poses model and the raw poses model is better than the majority.

Additionally, we identify the words with the highest mutual information between the occurrence of the word in the utterance and the probability to be predicted correctly. Table 8 shows the top 10

[5] We use more folds than typical in the literature to have more observations for significance testing.

[6] Since such a setting is rather restrictive, we also run Fisher's combined probability test to combine p-values from all the 30 runs. We present these results in the Appendix.

Class	Embeddings	Raw poses	# Observations	Description and Examples
PRONOUN	52.6 (±2.50)	52.2 (±2.54)	10034.0	Pronouns: you, nobody, us
ACTIVE	51.3 (±2.66)	50.0 (±2.76)	9533.2	Active verbs: do, develop, learn

Table 4: Accuracy on the General Inquirer category prediction task, where the accuracy of the embeddings model is significantly larger than the majority model at a 0.05 significance level, using a Wilcoxon signed-rank test for significance testing.

	Accuracy
Majority	50.0 (± 0.0)
Raw Poses	51.7 (± 3.04)
Embeddings	**52.7** (± 3.01)

Table 5: Accuracy on the discourse marker prediction task for raw poses and gesture embeddings. Some examples of discourse markers include: actually, anyway, so.

unigrams (top row) and bigrams (bottom row) with the highest mutual information. These words can be interpreted as the most expressive, as the corresponding gestures are more clearly distinctive from the other gestures. Once again, it appears that the majority of the expressive unigrams and bigrams represent function words, which further supports the strength of the connection between these groups of words and gestures.

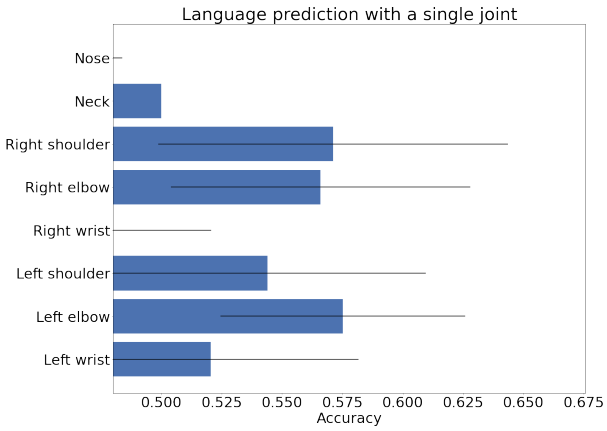

Figure 4: Results from fitting Logistic Classifier on single joint only on language prediction task.

One potential concern is to confirm that there is not any unseen bias (e.g., channel information) that is helping the system to recognize the language of the speaker. We manually inspected a random sample of 100 poses and could not see any differences, such as data artifacts from pose extraction, or any other data processing issues. In addition, we conducted several analyses:

- We analyzed the distribution of the joints' coordinates for the poses that were matched correctly versus incorrect ones. Similarly, we also analyzed the distribution of the joints' coordinates between English and Spanish videos. Maybe some joints are at the very specific position for English/Spanish videos that it makes very easy for the model to distinguish? We could not see any direct difference.

- We analyzed the coefficients of the logistic classifier for the embeddings model. We looked at the magnitude of the coefficients, i.e., whether some features drive the prediction. The motivation for this analysis is the following: if the logistic classifier relies on only a small number of features, instead of the learned representation as a whole, the gesture representation might be suboptimal.

- We fit the model on a **single** joint only. The motivation is the following: can we predict the language of the speaker from the neck (e.g., nose, shoulder) alone. Figure 4 shows the performance with the standard error bars (using 10 fold cross-validation). While the results are still worse than our proposed model, sometimes even one simple joint can lead to very strong performance.

6 Conclusions and Lessons Learned

In this paper, we explored the relation between gestures and language. Using a CLIP-style joint embedding model for gestures and language, applied on a bilingual multimodal dataset consisting of TED talks in English and Spanish, we report several findings:

First, we found that gestures can be used to infer the corresponding language and conversely that language can be used to infer the corresponding gesture. Our proposed model can predict the matching between language and gesture with 65.4% accuracy, compared to the random 50.0% baseline.

Class	Embeddings	Raw poses	Majority	# Observations	Description and Examples
SENSES	65.5 (13.66)	60.6 (16.90)	58.9 (6.62)	47.2	Sensory processes: see, touch, listen
PRESENT	64.6 (11.1)	59.6 (14.63)	54.5 (3.40)	176.0	Present focus: today, now
ARTICLE	64.0 (11.99)	59.3 (14.86)	58.2 (4.59)	161.9	Articles: a, an, the
COGMECH	63.8 (10.84)	59.7 (15.78)	58.1 (4.01)	152.6	Cognitive Processes: cause, know, ought
OTHREF	63.2 (11.86)	59.2 (15.16)	55.6 (4.40)	112.5	Other references: anyone, everyone
SOCIAL	63.6 (11.86)	59.6 (15.40)	58.1 (5.37)	148.6	Social Processes: talk, us, friend
PREPS	63.6 (11.36)	59.6 (14.61)	54.2 (3.57)	231.9	Prepositions: to, with, above
PRONOUN	63.5 (11.56)	59.7 (15.34)	56.4 (4.61)	177.6	Pronouns: you, nobody, us
AFFECT	65.3 (12.94)	62.0 (16.46)	61.0 (7.49)	49.5	Affective Processes: happy, ugly, bitter
INCL	62.9 (11.14)	60.2 (14.97)	58.5 (5.89)	124.9	Inclusive words: and, with, include
COMM	62.8 (14.44)	60.3 (17.42)	59.9 (6.57)	29.8	Common verbs: walk, went, see

Table 6: LIWC categories that drive better-than-random accuracy on the language prediction task.

	Accuracy
Majority	50.0 ($\pm$ 0.0)
Raw Poses	60.1 ($\pm$ 14.28)
Embeddings	**63.8** ($\pm$ 11.00)

Table 7: Accuracy on the language prediction task for raw poses and gesture embeddings

Unigrams	si, ade, y, asking, mas, here, life, po, medicine, ti
Bigrams	ade mas, from the, so that, and it, is to, y con, if your, we just, we can, we 've

Table 8: Top 10 unigrams and bigrams in English and Spanish, with the highest mutual information between the occurrence of the ngram in the utterance and the probability to be predicted correct.

Second, we showed that it is possible to predict several social or psycholinguistic word categories from the gestures with better than random probability. Through extensive probing of gesture embeddings for LIWC and General Inquirer linguistic categories, we were able to identify the categories where gesture embeddings significantly outperform random baselines: the majority of these categories consist of function words, which is a finding that aligns with previous social science findings. We report the results separately for English and Spanish.

In a similar vein, we showed that gestures can be also predictive of discourse markers. Our results indicate that gesture embeddings contain useful information about discourse structure, outperforming both majority and joint-only baselines.

Finally, we reported that gestures by themselves are predictive of the native language of the speaker, and that gesture embeddings further improve this result. Through several analyses, we found that function words are most strongly associated with gestures, which aligns with theories of language evolution that posit that function words are closely connected to the body.

There are several limitations of this work. First, this work focuses on hands and head gestures only, ignoring whole body movements and facial expressions. We also assume that two active hands/arms perform a single gesture, while it is also possible to have two separate gestures for two hands. The gestures are also specific for public presentations.

For future work, we plan to include hand gestures in our dataset. We also consider compiling an 'in-the-wild' gestures dataset, to extend our findings to more forms of communication, and expand beyond the gestures and language for TED talks.

7 Acknowledgement

This material is based in part upon work supported by the John Templeton Foundation (grant #61156). Any opinions, findings, and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of the John Templeton Foundation. We thank Hanwen Miao for his data analyses and suggestions on an earlier iteration of this project, and our colleagues Laura Biester, Dojune Min, Ashkan Kazemi, Jonathan Kummerfeld, Naihao Deng, Simin Fan, and Siqi Shen for reviewing an initial version of the manuscript and for all the feedback they provided.

References

Simon Alexanderson, Gustav Eje Henter, Taras Kucherenko, and Jonas Beskow. 2020. Style-controllable speech-driven gesture synthesis using normalising flows. *Computer Graphics Forum*, 39(2):487–496.

Martha W Alibali, Sotaro Kita, and Amanda J Young. 2000. Gesture and the process of speech production: We think, therefore we gesture. *Language and cognitive processes*, 15(6):593–613.

Zhe Cao, Gines Hidalgo, Tomas Simon, Shih-En Wei, and Yaser Sheikh. 2019. Openpose: Realtime multi-person 2d pose estimation using part affinity fields.

Yen-Chun Chen, Linjie Li, Licheng Yu, Ahmed El Kholy, Faisal Ahmed, Zhe Gan, Yu Cheng, and Jingjing Liu. 2019. Uniter: Universal image-text representation learning.

Cindy Chung and James W Pennebaker. 2007. The psychological functions of function words. *Social communication*, 1:343–359.

Alexis Conneau, Kartikay Khandelwal, Naman Goyal, Vishrav Chaudhary, Guillaume Wenzek, Francisco Guzmán, Edouard Grave, Myle Ott, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Unsupervised cross-lingual representation learning at scale.

Alexis Conneau and Douwe Kiela. 2018. Senteval: An evaluation toolkit for universal sentence representations. *CoRR*, abs/1803.05449.

Janez Demšar. 2006. Statistical comparisons of classifiers over multiple data sets. *J. Mach. Learn. Res.*, 7:1–30.

Karan Desai and Justin Johnson. 2021. Virtex: Learning visual representations from textual annotations.

Ylva Ferstl, Michael Neff, and Rachel McDonnell. 2021. It's a match! gesture generation using expressive parameter matching.

Shiry Ginosar, Amir Bar, Gefen Kohavi, Caroline Chan, Andrew Owens, and Jitendra Malik. 2019. Learning individual styles of conversational gesture. *Computer Vision and Pattern Recognition (CVPR)*.

Jana M Iverson and Susan Goldin-Meadow. 1998. Why people gesture when they speak. *Nature*, 396(6708):228–228.

Spencer D. Kelly, Sarah M. Manning, and Sabrina Rodak. 2008. Gesture gives a hand to language and learning: Perspectives from cognitive neuroscience, developmental psychology and education. *Language and Linguistics Compass*, 2(4):569–588.

Ryan Kiros, Ruslan Salakhutdinov, and Richard S Zemel. 2014. Unifying visual-semantic embeddings with multimodal neural language models. *arXiv preprint arXiv:1411.2539*.

Taras Kucherenko, Patrik Jonell, Sanne van Waveren, Gustav Eje Henter, Simon Alexandersson, Iolanda Leite, and Hedvig Kjellström. 2020a. *Gesticulator: A Framework for Semantically-Aware Speech-Driven Gesture Generation*, page 242–250. Association for Computing Machinery, New York, NY, USA.

Taras Kucherenko, Patrik Jonell, Sanne van Waveren, Gustav Eje Henter, Simon Alexandersson, Iolanda Leite, and Hedvig Kjellström. 2020b. Gesticulator: A framework for semantically-aware speech-driven gesture generation. *Proceedings of the 2020 International Conference on Multimodal Interaction*.

Sergey Levine, Christian Theobalt, and Vladlen Koltun. 2009. Real-time prosody-driven synthesis of body language. In *ACM Transactions on Graphics*, volume 28, page 172. ACM.

Liunian Harold Li, Mark Yatskar, Da Yin, Cho-Jui Hsieh, and Kai-Wei Chang. 2019. Visualbert: A simple and performant baseline for vision and language.

Daniel Loehr. 2007. Aspects of rhythm in gesture and speech. *Gesture*, 7(2):179–214.

Jiasen Lu, Dhruv Batra, Devi Parikh, and Stefan Lee. 2019. Vilbert: Pretraining task-agnostic visiolinguistic representations for vision-and-language tasks.

Andy Lücking, Kirsten Bergman, Florian Hahn, Stefan Kopp, and Hannes Rieser. 2013. Data-based analysis of speech and gesture: The bielefeld speech and gesture alignment corpus (saga) and its applications. *Journal on Multimodal User Interfaces*, 7(1):5–18.

David McNeill. 1992. Hand and mind: What gestures reveal about thought.

David McNeill. 2005. *Gesture and Thought*.

Saif M. Mohammad. 2018. Obtaining reliable human ratings of valence, arousal, and dominance for 20,000 english words. In *Proceedings of The Annual Conference of the Association for Computational Linguistics (ACL)*, Melbourne, Australia.

Louis-Philippe Morency, Ariadna Quattoni, and Trevor Darrell. 2007. Latent-dynamic discriminative models for continuous gesture recognition. In *Computer Vision and Pattern Recognition (CVPR)*, pages 1–8. IEEE.

Tewodros Legesse Munea, Yalew Zelalem Jembre, Halefom Tekle Weldegebriel, Longbiao Chen, Chenxi Huang, and Chenhui Yang. 2020. The progress of human pose estimation: A survey and taxonomy of models applied in 2d human pose estimation. *IEEE Access*, 8:133330–133348.

Nora S Newcombe. 2018. Three kinds of spatial cognition. *Stevens' handbook of experimental psychology and cognitive neuroscience*, 3:1–31.

F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. 2011. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830.

James W Pennebaker, Roger J Booth, and Martha E Francis. 2007. Linguistic inquiry and word count: Liwc [computer software]. *Austin, TX: liwc. net*, 135.

Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, Gretchen Krueger, and Ilya Sutskever. 2021. Learning transferable visual models from natural language supervision.

Hannes Rieser. 2015. When hands talk to mouth. gesture and speech as autonomous communicating processes. *SEMDIAL 2015 goDIAL*, page 122.

Katya Saint-Amand, Katya Saint Amand, and Katya Alahverdzhieva. 2013. Alignment of speech and co-speech gesture in a constraint-based grammar.

Damien Sileo, Tim Van de Cruys, Camille Pradel, and Philippe Muller. 2020. DiscSense: Automated semantic analysis of discourse markers. In *Proceedings of the 12th Language Resources and Evaluation Conference*, pages 991–999, Marseille, France. European Language Resources Association.

Kihyuk Sohn. 2016. Improved deep metric learning with multi-class n-pair loss objective. In *Advances in Neural Information Processing Systems*, volume 29. Curran Associates, Inc.

Philip J Stone, Dexter C Dunphy, and Marshall S Smith. 1966. The general inquirer: A computer approach to content analysis.

Hao Tan and Mohit Bansal. 2019. Lxmert: Learning cross-modality encoder representations from transformers.

Hao Tang, Wei Wang, Dan Xu, Yan Yan, and Nicu Sebe. 2019. Gesturegan for hand gesture-to-gesture translation in the wild.

Yonglong Tian, Dilip Krishnan, and Phillip Isola. 2020. Contrastive multiview coding.

Frank Wilcoxon. 1945. Individual comparisons by ranking methods. *Biometrics Bulletin*, 1(6):80–83.

Chenfei Wu, Jian Liang, Lei Ji, Fan Yang, Yuejian Fang, Daxin Jiang, and Nan Duan. 2021. Nüwa: Visual synthesis pre-training for neural visual world creation.

Youngwoo Yoon, Bok Cha, Joo-Haeng Lee, Minsu Jang, Jaeyeon Lee, Jaehong Kim, and Geehyuk Lee. 2020. Speech gesture generation from the trimodal context of text, audio, and speaker identity. *ACM Transactions on Graphics (TOG)*, 39(6):1–16.

Youngwoo Yoon, Woo-Ri Ko, Minsu Jang, Jaeyeon Lee, Jaehong Kim, and Geehyuk Lee. 2018. Robots learn social skills: End-to-end learning of co-speech gesture generation for humanoid robots.

Lu Yuan, Dongdong Chen, Yi-Ling Chen, Noel Codella, Xiyang Dai, Jianfeng Gao, Houdong Hu, Xuedong Huang, Boxin Li, Chunyuan Li, Ce Liu, Mengchen Liu, Zicheng Liu, Yumao Lu, Yu Shi, Lijuan Wang, Jianfeng Wang, Bin Xiao, Zhen Xiao, Jianwei Yang, Michael Zeng, Luowei Zhou, and Pengchuan Zhang. 2021. Florence: A new foundation model for computer vision.

Xiaohua Zhai, Xiao Wang, Basil Mustafa, Andreas Steiner, Daniel Keysers, Alexander Kolesnikov, and Lucas Beyer. 2021. Lit: Zero-shot transfer with locked-image text tuning.

Fan Zhang, Valentin Bazarevsky, Andrey Vakunov, Andrei Tkachenka, George Sung, Chuo-Ling Chang, and Matthias Grundmann. 2020. Mediapipe hands: On-device real-time hand tracking.

A Appendix

Here we present the results for LIWC/General Inquirer category prediction task, but instead of selecting categories that were significant at least 90% out of 30 runs with different random seeds, we run Fisher's combined probability test to merge p-values from all the 30 runs into single p-value.

Accuracy on the General Inquirer category prediction task, where the accuracy of the embeddings model is significantly larger at a 0.05 significance level than the majority model.

Class	Embeddings	Raw poses	# Observations	Description and examples
AFFECT	51.6 (5.22)	51.9 (4.93)	3014.1	Affective Processes: happy, ugly, bitter
ANX	51.3 (24.35)	52.7 (24.95)	146.9	Anxiety: nervous, afraid, tense
ARTICLE	50.8 (3.07)	50.5 (2.95)	7197.8	Articles: a, an, the
BODY	51.2 (11.73)	50.1 (11.84)	572.3	ache, heart, cough 1
CAUSE	51.3 (8.04)	50.4 (7.42)	1281.8	Causation: because, effect, hence
CERTAIN	51.4 (7.72)	49.1 (7.82)	1279.9	Certainty always, never
COGMECH	51.6 (3.64)	51.3 (3.23)	6801.5	Cognitive Processes: cause, know, ought
COMM	50.7 (6.82)	49.9 (6.92)	1616.3	Common verbs: walk, went see
DISCREP	52.9 (6.17)	51.7 (5.8)	2343.2	Discrepancy: should, would, could
EXCL	51.5 (3.99)	50.0 (3.84)	4510.5	Exclusive: but, except, without
FAMILY	53.5 (22.04)	53.1 (22.45)	261.0	mom, brother, cousin
FEEL	50.6 (15.66)	49.4 (15.04)	319.0	Feeling: touch, hold, felt
HEAR	50.9 (8.25)	50.7 (7.93)	1098.1	Hearing: heard, listen, sound
HUMANS	51.6 (7.4)	51.5 (7.12)	1451.9	boy, woman, group
I	52.5 (6.19)	51.1 (6.19)	2679.6	I, me, mine
INCL	51.1 (3.08)	50.4 (2.95)	7574.8	Inclusive: with, and, include
INHIB	52.5 (17.33)	50.1 (17.93)	247.5	Inhibition: block, constrain
INSIGHT	50.6 (5.87)	50.2 (5.79)	2339.3	Insight: think, know, consider
JOB	52.2 (9.81)	50.7 (10.06)	784.9	benefits, work, board
METAPH	51.4 (19.83)	51.7 (19.39)	253.3	Metaphysical issues: God, heaven, coffin
MOTION	51.0 (7.58)	49.9 (7.65)	1318.5	Motion: walk, move, go
NEGATE	51.9 (8.67)	51.0 (8.27)	1053.7	Negations: no, never, not
NEGEMO	51.3 (8.95)	50.2 (9.15)	914.5	Negative Emotions: hate, worthless, enemy
NUMBER	51.2 (8.36)	50.7 (8.08)	1150.3	Numbers: one, thirty, million
OCCUP	50.7 (6.29)	49.4 (5.95)	2062.9	Occupation: work, class, boss
OPTIM	50.7 (14.0)	49.1 (13.7)	421.5	Optimism and energy: certainty, pride, win
OTHER	50.8 (6.32)	49.7 (6.37)	1946.9	Total third person: she, their, them
OTHREF	51.4 (3.37)	51.3 (3.31)	6134.5	Other references: anyone, everyone
PAST	51.0 (4.66)	51.8 (4.57)	4272.7	Past tense verb: walked, were, had
POSEMO	50.8 (6.16)	51.1 (5.97)	2115.1	Positive Emotions: happy, pretty, good
POSFEEL	50.8 (11.37)	52.6 (10.97)	624.5	Positive feelings: happy, joy, love
PREPS	51.9 (2.32)	51.2 (2.19)	12978.5	Prepositions: on, to, from
PRESENT	51.1 (2.62)	50.2 (2.6)	9455.9	Present tense verb: walk, is, be
PRONOUN	52.5 (2.95)	52.3 (2.83)	10235.1	Total pronouns: I, our, they, you're
SCHOOL	52.4 (14.34)	49.1 (14.07)	409.9	School: class, student, college
SEE	51.0 (10.1)	49.4 (9.45)	769.5	Seeing: view, saw, look
SELF	52.0 (3.99)	52.5 (3.96)	5223.9	Total first person: I, we, me
SIMILES	51.7 (13.45)	51.0 (14.01)	407.9	like
SOCIAL	50.7 (3.21)	51.0 (2.83)	8872.1	Social Processes: talk, us, friend
SPACE	51.8 (5.53)	51.5 (5.17)	2571.3	Space: around, over, up
TENTAT	51.7 (5.95)	50.8 (5.69)	2318.1	Tentative: maybe, perhaps, guess
UP	52.2 (8.31)	51.4 (7.76)	1131.0	up, above, over
WE	52.5 (5.3)	52.8 (5.56)	2567.5	1st person plural: we, our, us
YOU	53.4 (6.47)	50.5 (6.79)	1591.1	Total second person: you, you'll

Table 9: Accuracy on the LIWC category prediction task for **English**. The accuracy of the majority baseline is 50%. We show the categories where the accuracy of the embeddings model is significantly larger than the majority model at a 0.05 significance level. We highlight in bold the categories where the embeddings model is significantly larger than raw poses model. $\pm$ denotes standard deviation

Class	Embeddings	Raw poses	# Observations	Description and examples
ANX	53.4 (25.29)	55.0 (25.38)	94.4	Anxiety: turba, miserable, temer
ARTICLE	51.1 (3.5)	50.4 (3.83)	5442.3	Article: los, la, una
ASSENT	52.3 (26.08)	49.0 (26.52)	156.6	bien, assent, ok
CAUSE	51.3 (10.79)	48.8 (10.84)	740.5	Causation: porque, dependo, recuperaron
COGMECH	50.9 (4.06)	49.9 (3.48)	5119.5	Cognitive Processes: conceder, asombra, pone
EXCL	51.2 (8.76)	50.7 (8.98)	1073.0	Exclusive: sacar, sin, menos
FRIENDS	52.8 (26.9)	48.0 (25.86)	63.5	examiga, comadre*, macho*
FUTURE	52.3 (17.37)	48.5 (17.88)	320.9	empezare*, frotare*, seremos
I	51.0 (8.36)	50.4 (8.36)	1243.1	mi, tuve, yo
INCL	50.6 (4.75)	49.4 (4.86)	3058.5	Inclusive: con, y, junto
LEISURE	52.8 (19.15)	50.9 (20.1)	262.9	trotar, compac, vives
NUMBER	52.7 (14.89)	52.9 (14.59)	365.4	mitad, once, nueve
PHYSCAL	51.0 (10.42)	50.9 (10.51)	736.6	Physical states: cruda, violar, patas
PREPS	52.2 (3.52)	51.1 (3.29)	6308.5	con, para, sobre
PRESENT	51.0 (4.18)	50.8 (3.89)	5028.6	Present tense: coge, entrego, desean
SOCIAL	50.8 (4.61)	49.3 (4.47)	3692.7	entrego, primo, oyes
YOU	51.3 (14.73)	50.8 (13.76)	475.6	estas, vos, tu

Table 10: Accuracy on the LIWC category prediction task for **Spanish**. The accuracy of the majority baseline is 50%. We show the categories where the accuracy of the embeddings model is significantly larger than the majority model at a 0.05 significance level. We highlight in bold the categories where the embeddings model is significantly larger than raw poses model. $\pm$ denotes standard deviation

Class	Embeddings	Raw poses	# Observations	Description and Examples
ACADEMIC	50.8 (8.68)	50.6 (8.39)	1015.0	academy, dean, coach
ACTIVE	51.3 (2.66)	50.1 (2.76)	9533.3	accost, actor, alarm
BEGIN	50.7 (9.67)	50.3 (9.73)	781.1	bloom, dawn, first
CAUSAL	52.2 (5.17)	52.1 (5.39)	2646.7	order, premise, odds
COLLECTIVITIES	52.1 (6.37)	49.5 (6.04)	2049.3	crowd, cult, family
COMMUNICATION FORM	50.8 (4.27)	50.5 (3.95)	4096.7	ask, assign, discuss
DESCRIPTIVE VERBS	50.7 (3.07)	50.5 (3.49)	6793.7	moan, mumble, pinch
FALL	53.5 (33.31)	57.9 (30.73)	61.4	sunk, drop, collapse
FINISH	51.4 (11.24)	50.2 (10.48)	665.1	cease, expire, lost
FREQUENCY	52.1 (9.4)	49.5 (9.06)	802.3	repeat, weekly, rare
HUMAN'S ROLES	50.5 (4.4)	50.8 (4.42)	3893.7	antagonist, cook, genius
INCREASE	51.0 (8.79)	50.2 (8.94)	1022.7	quicken, run, elaborate
INTERJECTION	51.9 (5.67)	50.0 (5.59)	2339.3	okay, damn, well
INTERPERSONAL	50.5 (4.28)	50.8 (4.2)	4234.0	adversary, hug, recruit
INTERPRETATIVE VERBS	51.0 (2.58)	50.3 (2.77)	10376.2	control, define, educate
KIN	53.3 (18.12)	48.8 (17.87)	297.7	mother, uncle, ma
LEGAL	51.2 (7.46)	50.4 (7.21)	1308.9	convict, crime, unjust
MALE ROLES	52.6 (12.74)	50.5 (12.52)	587.7	salesman, pope, husband
MEANS	50.9 (4.68)	50.0 (4.56)	3393.0	wage, utility, consideration
NEGATION	51.5 (7.09)	50.7 (7.23)	1357.2	aińt, disapprove, no
NEGATIVE	50.8 (4.81)	50.7 (4.93)	2927.1	break, deviation, furious
NUMBER CARDINAL	52.6 (9.21)	51.3 (8.84)	957.0	seven, zero, two
PLACE AQUATIC	54.8 (29.82)	51.3 (31.38)	125.4	bay, swamp, water
PLACE LAND	53.9 (15.02)	53.9 (15.03)	328.7	hilly, desert, cave
PRONOUN	52.7 (2.5)	52.2 (2.55)	10034.0	you, us, those
QUALITY ASSESSMENT	51.3 (6.43)	48.8 (6.18)	1828.9	modesty, hilarious, curve
QUANTITY ASSESSMENT	51.1 (3.46)	49.6 (3.05)	8094.9	considerable, all, another
RELATIONSHIPS	50.8 (5.41)	49.2 (5.37)	2594.5	tie, coherent, unlike
RISE	52.5 (14.33)	48.3 (14.2)	367.3	raise, jump, peak
ROLE	50.9 (6.07)	49.7 (6.39)	2066.7	alcoholic, buddy, mentor
SELF	52.7 (6.66)	51.2 (6.32)	2679.6	me, mine, I
SELF EXPRESSION	50.9 (8.59)	52.8 (8.69)	1121.3	vacation, paint, actor
SPACE	51.9 (4.57)	51.4 (4.8)	3060.5	way, on, nearby
STATE VERBS	50.6 (3.58)	50.9 (3.39)	5919.9	feel, seem, am
STAY	52.2 (17.24)	49.6 (16.45)	282.3	await, locate, set
STRONG	51.0 (2.5)	50.1 (3.03)	10066.9	aptitude, autocratic, defense
SUBMISSION	51.1 (7.55)	51.0 (7.44)	1374.6	respect, kneel, honor
TOOL	52.4 (7.2)	49.8 (7.19)	1341.7	Fork, stove, wheel
TRAVEL	51.1 (6.13)	50.8 (6.15)	1902.4	walk, leave, away
TRY	51.1 (7.92)	51.4 (7.15)	1293.4	bring, attempt, seek
UNDERSTATED	51.2 (3.78)	50.3 (3.93)	4877.8	caution, gamble, rare
VARY	52.1 (9.05)	51.1 (9.1)	887.4	turn, divert, amenable
VICE	51.6 (6.74)	50.9 (6.3)	1784.5	bore, damage, loss
VIRTUE	50.9 (3.92)	50.3 (3.71)	4568.5	invulnerable, free, admirable
WE	53.0 (5.36)	53.0 (5.18)	2488.2	ours, ourselves, we
WEAK	51.7 (5.0)	50.3 (4.76)	3271.2	addict, cheap, sunken
YES	51.9 (10.89)	51.2 (11.04)	665.1	yeah, okay, definitely
YOU	52.9 (6.9)	50.1 (6.5)	1610.5	your, thy, thou

Table 11: Accuracy on the General Inquirer category prediction task. The accuracy of the majority baseline is 50%. We show the categories where the accuracy of the embeddings model is significantly larger than the majority model at a 0.05 significance level. We highlight in bold the categories where the embeddings model is significantly larger than raw poses model. ± denotes standard deviation

Building Joint Relationship Attention Network for Image-Text Generation

Changzhi Wang and XiaoDong Gu*

Department of Electronic Engineering, Fudan University
Shanghai, 200438, China
18110720047@fudan.edu.cn(C. Wang) xdgu@fudan.edu.cn(X. Gu)

Abstract

Attention based methods for image-text generation often focus on visual features individually, while ignoring relationship information among image features that provides important guidance for generating sentences. To alleviate this issue, in this work we propose the Joint Relationship Attention Network (JRAN) that novelly explores the relationships among the features. Specifically, different from the previous relationship based approaches that only explore the single relationship in the image, our JRAN can effectively learn two relationships, the visual relationships among region features and the visual-semantic relationships between region features and semantic features, and further make a dynamic trade-off between them during outputting the relationship representation. Moreover, we devise a new relationship based attention, which can adaptively focus on the output relationship representation when predicting different words. Extensive experiments on large-scale MSCOCO and small-scale Flickr30k datasets show that JRAN achieves state-of-the-art performance. More remarkably, JRAN achieves new 28.3% and 58.2% performance in terms of BLEU4 and CIDEr metric on Flickr30k dataset.

1 Introduction

Image-text generation (i.e., image captioning) is a typical cross-modal task that connects Natural Language Processing (NLP) and Computer Vision (CV) (Tahvili et al., 2020). Its core goal is to automatically predict a meaningful and grammatically correct sentence, which can accurately describe the main content of images. Practical applications for this task mainly include injecting visual intelligence into the chatbots, searching semantic image, and helping people with visual impairments to understand the visual world. However, image-text generation is still a challenging task. The main difficulties originate from two aspect: **1)** The noise and complex background in the image are likely to interfere with the generation of correct caption; **2)** The interaction between features in the image is often overlooked.

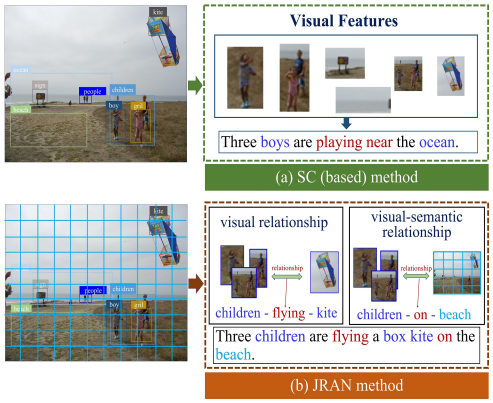

Figure 1: Illustration of different schemes. (a) is an example predicted by SC (base) baselines (Chen et al., 2017) that mainly uses visual features to generate sentence. (b) presents a more accurate sentence generated by our JRAN method which learns two relationship, i.e., visual relationship between region features and visual-semantic relationship between region features and semantic features (i.e., background and environment).

For difficulty **1)**, encouraging by the method (Wu et al., 2021b), the noise can be injected into RNN hidden states to predict the mean and standard deviation, and manipulate the RNN transition states. In this way, the network robustness can be significantly enhanced and the issue can be well solved. However for difficulty **2)**, although some related visual attention based methods (Xu et al., 2015; Wang et al., 2016; Song et al., 2018) achieve remarkable progress, they usually focus on the image visual features while ignoring the relationships between them. This makes the model often difficult to generate an accurate or appropriate description that can correctly describe the relationships among objects in the image. For example, as illustrated in Figure 1, only using the detected visual features in

*Corresponding author.

Proceedings of the 29th International Conference on Computational Linguistics, pages 5521–5531
October 12–17, 2022.

the image, SC (base) method (Chen et al., 2017) predicts a description "*Three boys are playing near the ocean.*", where the verb phrase "playing near" indicates the relationship between the objects "boy" and "ocean". Obviously, this description cannot accurately reflect the main scene of the image. Contrarily, our JRAN accurately describes the main content of image by effectively learn the visual relationship between object region features and the visual-semantic relationship between region features and semantic features, generating a more relevant sentence "*Three children are flying a box kite on the beach.*" The word "flying" appropriately describes the relationship between the two region features "children" and "kite", and the word "on" accurately represents the relationship between region information "children" and semantic information "beach" (i.e., background/surrounding). Thus, learning the relationship between image features is of crucial importance for generating accurate sentence description for image.

Based on the above observations, different from previous relationship based approaches (Wang et al., 2020; Kipf and Welling, 2017; Li and Jiang, 2020) (See Figure 2(top)) that only explore the single feature relationship in the image, we present a new Joint Relationship Attention Network (JRAN) that novelly learns the joint relationship between region features and semantic features in Figure 2(bottom).

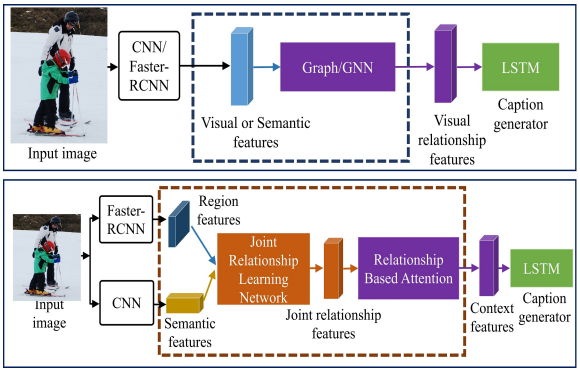

Figure 2: The illustration of existing relationship based methods (top) and our JRAN method (bottom). (a) Relationship based methods usually use Graph/GNN (Graph Neural Network) to explore single (visual or semantic) feature relationship; (b) Our JRAN method can learn the joint relationship between visual features and semantic features.

Specifically, we first utilize the object detector Faster R-CNN and CNN to extract region features and complementary semantic features from the input image. Then, JRAN builds two types of relationship, i.e., visual relationship and visual-semantic relationship. The former is solely based on the detected region features in the image, and encodes the visual relationship between region features. Instead, the latter takes the region features and semantic features of the image into account to fully explore the visual-semantic relationship between them. As shown in Figure 3, to obtain the image representation containing joint relationship features, we devise a core competition module called joint relationship learning network, which can effectively learn the visual relationship and visual-semantic relationship while dynamically balancing the different contributions between them. After that, we introduce the relationship based attention module, which can adaptively focus on the obtained most relevant joint relationship features during generating words. The whole framework of JRAN could be jointly learnt and optimized in an end-to-end way. Our JRAN method achieves significant improvement compared with the related relationship based methods. More remarkably, it can be integrated into a better baseline model to achieve better performance.

Our contributions mainly include: firstly, we propose a novel image-text generation network that utilizes the complementary region and semantic features in the image for enriching feature representations. Secondly, we devise a joint relationship learning network, which can fully learn both visual relationship and visual-semantic relationship in the image, and further balance their contributions during predicting different words. Finally, exhaustive experiments indicate that our JRAN is not only effective on large-scale but also achieves superior performance on small-scale datasets.

2 Related Works

2.1 Image-Text Generation

Image-text generation can be treated as a sequence-to-sequence task, which converts the data from a raw image to a sentence description. For example, Wang et al. (2016) presented an end-to-end architecture to generate the sentence where visual embedding is encoded with CNN and sentence embedding is encoded using Bi-LSTM. Xu et al. (2015) presented the first model based on visual attention, where it extracts visual features of each region from the raw image, and then assigns different weights for them. Chen et al. (2017) in-

tegrated the spatial and channel attention features extracted from a CNN, and uses the channel attention to focus on different semantic information. Similarly, Song et al. (2018) integrated the spatial and channel attentions into salient object regions, and effectively improved the performance of Visual Question Answering (VQA) task. Although the above approaches have well performance, they ignore the relationships among image features. In our work, on one hand, we take full advantage of the two complementary region features and semantic features in the image. On the other hand, we learn the visual relationships among region features while exploring the visual-semantic relationships between region features and semantic features.

2.2 Relationship Based Approaches

Recently, relationship based methods have been proposed to boost the performance for image-text generation task. It mainly uses Graph Convolution Network (GCN) and Graph Attention neTwork (GAT) to learn the single relationship between local features or global features. For instances, Li et al. (2018b) took the data of arbitrary graphic structure as the input and introduced a flexible and general GCN. Kipf and Welling (2017) presented the GCN, which can be directly used to process graph structure data. Wang et al. (2020) utilized GAT to learn the relationship between image features. It directly inputs the extracted region features and semantic features into the GAT, and then follows the self-attention strategy to calculate the relationship between each feature node. Different from previous relationship based methods, we devise a new joint relationship attention network, which can capture the visual relationship and visual-semantic relationship in the image, and then further balance their different contributions during generating different words. This effectively promotes the model performance.

3 Our approach

The main purpose of this work is to explore the relationship between different features in the image, so as to generate a sentence description containing accurate interaction information for the input image. The overall architecture of the proposed JRAN is shown in Figure 3. The critical elements in the architecture are described in detail as follows.

3.1 Problem Formulation

Formally, a sentence model receives a source image I as the input and is required to output a target text sentence S to describe the image main content. S is a sequence of sentence generated word by word, which can be presented as $S = \{x_1, x_2.., x_T\}$, where T denotes the length of the sequence, and x_t, $t \in [1, T]$ is the t-th word. During the training, given a training dataset with a set of image-sentence pairs (I_i, S_i), and sentence model is trained to minimize the cross entropy loss which is equivalent to maximizing the likelihood,

$$L_{loss}(\theta) = -\sum_{i=1}^{M}\sum_{t=1}^{T}(logp(x_{i,t}|I_i, x_{i,1:t-1}, \theta)), \quad (1)$$

where θ is the model parameters needed to train, M is the total number of training samples, and $x_{i,t}$ denotes the t word of ground-truth caption S_i.

3.2 Feature Extraction

We extract the two complementary features from the raw image: region features $\mathbf{V}_r$ and semantic features $\mathbf{V}_s$. For region features $\mathbf{V}_r \in \mathbb{R}^{D\times K}$, the raw image is first fed into Faster R-CNN to detect the top K candidate visual regions. For each selected region k, we take the mean-pooled convolutional feature from the image region as $\mathbf{V}_r^k$, which has D dimensions. Thus, the region features $\mathbf{V}_r = [\mathbf{V}_r^1, ..., \mathbf{V}_r^K]$, $\mathbf{V}_r^k \in \mathbb{R}^D$. For semantic features $\mathbf{V}_s \in \mathbb{R}^{L\times D}$, since last convolutional layer ($Conv5\_3$) of ResNet usually contains the context (or background) information around objects (Li et al., 2018a), thus, it is extracted as the image semantic features. Then, the extracted feature map $\mathbf{V}_l \in \mathbb{R}^{W\times H\times D}$ is further flattened into $\mathbf{V}_s \in \mathbb{R}^{L\times D}$, $L = W \times H$,

$$\mathbf{V}_s = \{\mathbf{V}_s^1, ..., \mathbf{V}_s^D\} = flatten(Conv(I)), \quad (2)$$

where $\mathbf{V}_s^i \in \mathbb{R}^L$,$i \in \{1, 2, .., D\}$ represents the i-th semantic feature of the feature map $\mathbf{V}_s$.

3.3 Joint Relationship Learning Network

We devise the Joint Relationship Learning Network (JRLN) in Figure 4. It is consists of some stacked feature relationship network, and each feature relationship network is composed of a multi-head Relationship Computation (RC) module. On one hand, JRLN learns the visual relationship between region features, which can unfold the inherent action/interaction between different region objects.

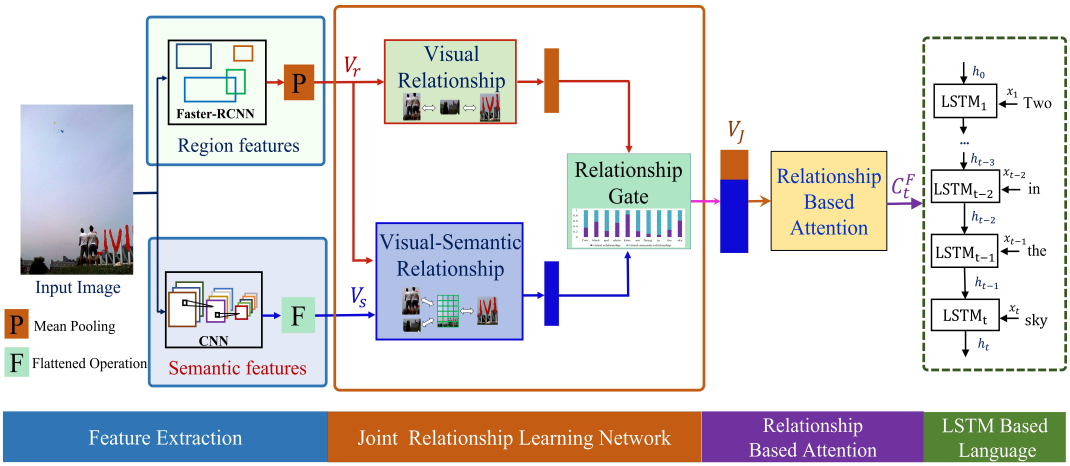

Figure 3: An overall framework of our proposed JRAN which consists of four modules: Feature Extractor module, Joint Relationship Learning Network, Relationship Based Attention module and LSTM Based Language module. We first use Feature Extractor module to extract region features V_r and complementary semantic features V_s for image feature representation. Then, these features are input into the joint relationship learning network respectively, and the balanced relationship feature V_J are output. After that, V_J is fed into relationship based attention module to obtain the final context representation C_t^F. Finally, C_t^F is input into the LSTM Based Language module for word generation.

On the other hand, JRLN utilizes the image semantic features as guide to learn the visual-semantic relationship between the semantic features and complementary region features, which effectively connects isolated region objects with their background (or environment) information. Importantly, JRLN further balances the different contributions between the two relationships during generating words.

Now, we describe our competitive joint relationship learning network. It consists of some Relationship Attention (RA) and Feed-Forward Network (FFN) modules, in which RA includes N_r RC modules. And the RC is able to learn two kinds of relationships: *(I)* Visual relationships among region features, *(II)* Visual-semantic relationships between region features and semantic features.

I) Visual relationships among region features: Considering that the region features in the image do not exist independently of each other, we learn the visual relationship among region features by using a multi-head RC, as illustrated in Figure 4 (b)-(I). Given the input set of N region features $\{\mathbf{V}_r^i\}$, the output visual relationship feature $\mathbf{V}_v^i$ of the whole region feature with respect to the i-th region feature is calculated as follows

$$\mathbf{V}_v^i = \sum_j \varphi^{ij} \cdot (\mathbf{W}_v \mathbf{V}_r^j), \tag{3}$$

where $\mathbf{W}_v$ is the model learnable matrix. Further, the visual relationship weight φ^{ij} is calculated by measuring the correlation between the i-th region feature and the j-th region feature,

$$\varphi^{ij} = softmax(\frac{\mathbf{W}_q \mathbf{V}_r^i \cdot (\mathbf{W}_k \mathbf{V}_r^j)^T}{\sqrt{d_k}}), \tag{4}$$

where $\mathbf{W}_q$ and $\mathbf{W}_k$ are the projection matrices of the i-th and j-th region features. d_k is the matrix dimension after projection (i.e., scaling factor). Eq. (4) reflects how much every region is affected by other regions, where semantically more corresponding regions may have higher relationship weight values in the image.

II) Visual-semantic relationships between region features and semantic features: Since region features and semantic features are related to some extent, how to learn the relationship between them is important for image-text generation. Generally, to effectively organize the region features guided by the image semantic features, a common and straightforward idea is to concatenate the guided semantic features and all region features (i.e., Figure 5(b)). However, such a scheme is too naive to model the relative importance of the region and semantic features, i.e., the discrimination introduced by semantic guiding features and region features are different and should be distinguished. Thus, we propose to utilize an attention mechanism to weight the relative relationship between region features and semantic features, as illustrated in Figure 4 (d)-(II). Given the input region features $\mathbf{V}_r$ and

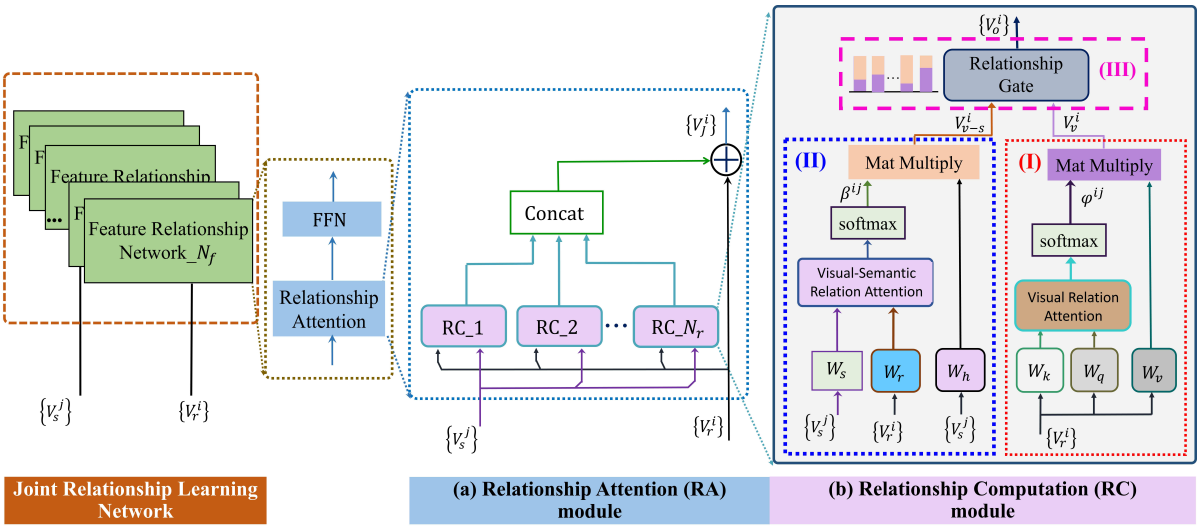

Figure 4: The illustration of our Joint Relationship Learning Network. It consists of a stack of Feature Relationship Network, and each feature relationship network includes a Feed-Forward Network (FFN) and a Relationship Attention (RA) module (a). Further, each RA consists of N_r Relation Computation (RC) modules (b). In addition, in RC module (b), the broken red line part (I) indicates the visual relationships among region features; the broken blue line part (II) is the visual-semantic relationships between region features and semantic features, and the broken pink line part (III) denotes our designed Relationship Gate.

semantic features $\mathbf{V}_s$ of the raw image, the output visual-semantic relationship feature $\mathbf{V}^i_{v-s}$ is,

$$\mathbf{V}^i_{v-s} = \sum_j \beta^{ij} \cdot (\mathbf{W}_h \mathbf{V}^j_s), \tag{5}$$

where $\mathbf{V}^i_{v-s}$ denotes the i-th visual-semantic relationship feature between region features and semantic features. Further, the visual-semantic relationship weight β^{ij} between the i-th region feature and j-th semantic feature is computed according to

$$\beta^{ij} = softmax(\frac{\mathbf{W}_s \mathbf{V}^j_s \cdot (\mathbf{W}_r \mathbf{V}^i_r)^T}{\sqrt{d_k}}), \tag{6}$$

where β^{ij} reflects the influence of image semantic features on region features. $\mathbf{W}_s$ and $\mathbf{W}_r$ are the model learnable matrices, which project the original semantic features $\mathbf{V}^j_s$ and region features $\mathbf{V}^i_r$ into the subspaces to measure how well they match, and d_k is the feature dimension after projection.

III) Relationship gate: As described in Section introduction, relationships are of crucial important for accurately describing the main content of the input images. Considering that visual relationships $\mathbf{V}^i_v$ and visual-semantic relationships $\mathbf{V}^i_{v-s}$ play different roles during generating different words. Thus, we introduce a relationship gate, which can dynamically balance their different contributions to obtain the image feature representation $\mathbf{V}^i_o$ containing different relationship information,

$$\mathbf{V}^i_o = \sigma \cdot \mathbf{W}_{vs} \mathbf{V}^i_v + (1 - \sigma) \cdot \mathbf{W}_{vb} \mathbf{V}^i_{v-s}, \tag{7}$$

where σ is the relationship gate coefficient, as,

$$\sigma = sigmoid(\text{Concat}(\mathbf{U}_{vr} \mathbf{V}^i_r, \mathbf{U}_{vs} \mathbf{V}^i_s) + \mathbf{b}_\sigma), \tag{8}$$

where $\mathbf{W}$ and $\mathbf{U}$ are the learnable matrixes, and $\mathbf{b}_\sigma$ is a bias.

Further, to comprehensively learn visual relationship while capturing visual-semantic relationship, in our model we devise the multi-head RC in which each head can focus on different relationship attributes. Specifically, the relationship attention module aggregates in total N_r RC modules,

$$\mathbf{V}^i_J = \mathbf{S}_r \mathbf{V}^i_r + \text{Concat}(\mathbf{V}^i_o(1), ..., \mathbf{V}^i_o(N_r)) \mathbf{S}_n, \tag{9}$$

where $\mathbf{S}_r$ is learnable parameter, and $\mathbf{S}_n$ is the output projection matrix that aggregates the information from different heads, and $\mathbf{V}^i_o(n)$, $n \in (1, N_r)$ denotes the n-th relationship feature.

Finally, a basic feed-forward network is complemented to increase the model non-linearity, which takes V^i_J as its input and outputs as follows

$$\mathbf{V}^i_J = \mathbf{V}^i_J \mathbf{S}_J + \mathbf{b}_J. \tag{10}$$

where $\mathbf{S}_J$ and $\mathbf{b}_J$ are the learnable parameters. $\mathbf{V}^i_J$ denotes the i-th joint relationship feature obtained by relationship attention module.

3.4 Relationship Based Attention

Although the obtained joint relationship feature $\mathbf{V}_J^i$ have provided a full relationship representation for the image, in many cases, a word/phrase in the generated sentence is only related to some of the specific information containing in the feature representation. Thus, we further develop a relationship based attention module to automatically attend to the corresponding relationship feature during generating words. The final attention context vector C_t^F is obtained by the following updates,

$$z_{i,t} = \mathbf{W}_r^T tanh(\mathbf{W}_J \mathbf{V}_J^i + \mathbf{W}_h \mathbf{h}_{t-1} + \mathbf{b}_h), \quad (11)$$

$$\eta_t = softmax(\mathbf{z}_t), \quad (12)$$

$$\mathbf{C}_t^F = \sum \eta_t \mathbf{V}_J^i, \quad (13)$$

where $\mathbf{W}$ and $\mathbf{b}_h$ are the model learnable parameters. Next, $\mathbf{C}_t^F$ will be fed into the LSTM based language module to predict word.

4 Experiments

4.1 Experiment Setting

Datasets and Evaluation Metrics: We extend the experiment from large-scale MSCOCO (Lin et al., 2014) to small-scale Flickr30k (Plummer et al., 2015) datasets to verify the effectiveness of our model. Further, several popular evaluation metrics: BLEU (Papineni et al., 2002), ROUGE-L (R) (Lin, 2004), METEOR (M) (Banerjee and Lavie, 2005), CIDEr (C) (Vedantam et al., 2015) and SPICE (S) (Anderson et al., 2016) are used to evaluate the model performance, and coco-caption code[1] is utilized to compute these metrics.

Implementation Details: The LSTM hidden state dimension is set to 512, and the number of hidden cells and the embedded size of input words are also set to 512. Further, the bottom-up features provided by UD (Anderson et al., 2018) is also used. We use the gradient clipping strategy during back propagation to alleviate the problem of gradients explosion. The initial learning rate of CNN is 1e-5 and that of language model is 5e-4. When fine-tuning the image model, the learning rate we used is considerably smaller than that originally used for the training model. In 24 training epochs, the model stops training if its performance is not improved. In addition, these experiments are implemented via PyTorch, and we use Beam Search (BS) strategy for predicting caption.

[1] Available: https://github.com/tylin/coco-caption

4.2 Ablation Studies

Firstly, some ablation experiments are performed to clarify the effectiveness of following modules: 1) Region Features (R Fea.), 2) Semantic Features (S Fea.), 3) Joint Relationship Learning Network (JRLN), 4) Relationship based Attention (R-Att.) module. Then, the effects of different relationship fusion schemes are further analyzed in detail.

a. Effectiveness of Each Module: As shown in Table 1, **1)** in lines 1 and 2, "R Fea." or "S Fea.", "JRLN" means that model only uses the separate region features or semantic features to build the relationship; **2)** "R Fea." and "S Fea." means that region features and semantic features are directly concatenated, and then directly input into LSTM to generate sentence; **3)** "R Fea.", "S Fea." and "JRLN" means that model learns visual relationship and visual-semantic relationship, but it does not introduce relationship attention to focus on them; **4)** "R Fea.", "S Fea." and "R-Att" denotes that a relationship attention module is directly used to focus on the concatenated region and semantic features; **5)** in line 6, "R Fea.", "S Fea.", "JRLN" and "R-Att" is our full model, which explores the two relationships among image features, and then exploits relationship attention to dynamically focus on the obtained relationship representation.

Num.	Model Settings				Model Metrics				
	R Fea.	S Fea.	JRLN	R-Att.	B-1	B-4	M	R	C
1	✓		✓		77.9	36.1	26.2	56.3	120.8
2		✓	✓		77.4	35.6	25.7	56.0	120.6
3	✓	✓			78.8	37.1	26.8	56.8	121.4
4	✓	✓	✓		80.6	38.2	28.1	58.1	127.9
5	✓	✓		✓	79.1	37.4	27.1	57.1	123.2
6	✓	✓	✓	✓	**81.0**	**38.6**	**28.3**	**58.3**	**128.4**

Table 1: Ablation performance of JRAN model on MSCOCO dataset. '✓' means that the model only uses the module for image-text generation.

We have the following conclusions from Table 1: **1)** The metric scores line 4 is higher than line 3, which shows that the designed JRLN can effectively learn the visual-semantic relationship between region features and semantic features. **2)** Line 4 outperforms separate lines 2 and 1, it indicates that visual relationship and visual-semantic relationship can complement each other. **3)** The performance of model line 6 is better than line 4, it indicates that relationship based attention module can boost the model performance. **4)** Our full model in line 6 obtains the best score, which demonstrates the overall effectiveness of the proposed model.

b. Effectiveness of Relationship Fusion Scheme:

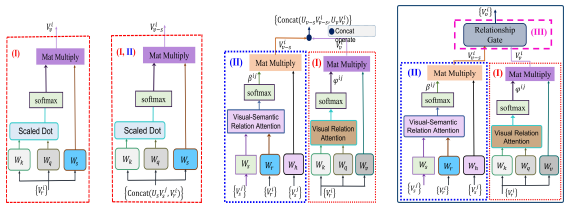

(a) *None* (b) *Concat* (c) *No-gate* (d) *RC (Ours)*

Figure 5: Different schemes for exploring the relationship between image features. (a) *None* and (b) *Concat* utilize *scaled dot-product attention*; (c) *No-gate* directly concatenates the two relationships. (d) *RC (Ours)* is our designed relationship computation (RC) scheme.

There are generally four directions for deeply exploring the relationship among features in the image, as shown in Figure 5. **(a)** *None* denotes that model only learns the visual relationship among region features by using *scaled dot-product attention* model (Vaswani et al., 2017). Similarly, **(b)** *Concat* follows the recent methods (Vaswani et al., 2017; Duan et al., 2017), i.e., region features and semantic features are directly concatenated, and then directly fed into *scaled dot-product attention* model to learn the visual-semantic relationships. **(c)** *No-gate* does not use our designed relationship gate, ignoring the different contributions between visual relationships and visual-semantic relationships during generating different words. **(d)** *RC (Ours)* is our full competitive RC module, which fully learns the two types of relationships, and further takes into account that different relationships contribute different during generating words.

We compare the performances of the RC variants in the four schemes. The results are 36.3%, 37.9%, 38.4% and 38.6% in BLUE4 metric and 120.9%, 122.2%, 128.1% and 128.4% in CIDEr metric for (a), (b), (c) and (d) schemes, respectively. It indicates that our designed RC scheme outperforms other learning relationship schemes.

5 Experimental Results

5.1 Comparison with State-of-The-Arts

Note that, for fair comparison, these models (Cornia et al., 2020; Yao et al., 2019) are not included in the comparison since the former (Cornia et al., 2020) uses the extra dataset nocaps (Kuznetsova et al., 2018), and the latter (Yao et al., 2019) utilizes the extra COCO-detect to segment the whole object.

a. Results on MSCOCO Dataset: The comparison results on MSCOCO are shown in the left part of Table 2. It can be observed that our JRAN achieves promising results. Compared with the typical baselines (Li et al., 2018a), our newly proposed JRAN can significantly improve BLEU 4 score from 36.4 to 38.6 (6.04%) and CIDEr score from 122.2 to 128.4 (5.07%), which is a significant improvement.

b. Results on Flickr30k Dataset: To further evaluate the model generalization ability, we conduct experiments on small-scale Flickr30k dataset. As shown in the right part of Table 3, JRAN achieves the superior performance. This further demonstrates that our model still maintains good generalization ability even on small-scale dataset.

5.2 Comparison with Similar Relationship Based Methods

More importantly, relationship based methods also explore the visual relationships among image features by using the graph/GCN. For example, method "KMSL" (Li and Jiang, 2020) explicitly utilizes the semantic relationship triples (scene graph) as additional inputs to explores the visual relationship. Similarly, "ARL" (Wang et al., 2020) explores the visual relationship among image regions by using GNN/GCN. As showed in Table 3, the performance of our JRAN is significantly better than these relationship based methods across all metrics, which quantitatively demonstrates the potentials of our joint relationship attention network.

5.3 Comparison with Transformer Based Baselines

In particular, to demonstrate that our JRAN can be integrated into the current mainstream transformer based baselines to achieve a better performance, we upgraded our baselines with a plain model "Simplistic Transformer architecture (*Sim-Trans*)[2]" as new baselines. The model doesn't use transformer encoder and the projected visual features are directly processed by the transformer decoder. Therefore, to see the real performance gain contributed by our JRAN model, we feed the final attention context features C_t^F from the relationship based attention module into the transformer decoder. In Table 4, the last few rows show the results of our

[2]https://github.com/krasserm/fairseq-image-captioning

Methods	MSCOCO								Flickr30k						
	B-1	B-2	B-3	B-4	M	R	C	S	B-1	B-2	B-3	B-4	M	R	C
ALT-ALTM (Ye et al., 2018)	75.1	59.0	45.7	35.5	27.4	55.9	110.7	20.3	68.5	50.7	37.0	27.0	21.2	48.0	56.2
SCST (Gao et al., 2019)	77.9	61.5	46.8	35.0	26.9	56.3	115.2	20.42	-	-	-	-	-	-	-
VD-SAN (He et al., 2019)	73.4	56.6	42.8	32.2	25.4	-	99.9	-	65.2	47.1	33.6	23.9	19.9	-	-
Up-Down (Anderson et al., 2018)	79.8	-	-	36.3	27.7	56.9	120.1	21.4	-	-	-	-	-	-	-
GLA (Li et al., 2018a)	72.5	55.6	41.7	31.2	24.9	53.3	96.4	-	56.8	37.2	23.2	14.6	16.6	41.9	36.2
HAN (Wang et al., 2019)	80.9	64.6	**49.8**	37.6	27.8	58.1	121.7	21.5	-	-	-	-	-	-	-
Trans+KG (Zhang et al., 2021)	76.24	-	-	34.39	27.71	-	112.60	21.12	68.36	-	-	26.55	21.71	-	56.62
TDA+GLD (Wu et al., 2021a)	78.8	62.6	48.0	36.1	27.8	57.1	121.1	21.6	-	-	-	-	-	-	-
cLSTM-RA (Yang et al., 2020)	**81.7**	64.5	49.4	37.2	28.0	57.9	121.5	-	70.5	52.5	37.6	27.1	21.9	49.4	57.7
Baselines	79.8	63.1	48.2	36.4	27.8	57.1	122.2	21.5	69.2	51.3	37.6	27.7	22.1	49.6	57.2
JRAN (BS=3) (ours)	80.8	64.4	49.6	38.3	28.4	58.2	128.0	21.8	69.9	53.0	37.9	27.9	24.8	52.6	57.9
JRAN (BS=4) (ours)	80.9	64.6	49.7	38.5	**28.5**	**58.4**	128.2	22.0	71.0	53.1	38.1	28.1	24.8	52.7	58.0
JRAN (BS=5) (ours)	81.0	**64.7**	**49.8**	**38.6**	28.3	58.3	**128.4**	**22.1**	**71.2**	**53.3**	**38.3**	**28.3**	**25.0**	**52.9**	**58.2**

Table 2: Performance of JRAN and related state-of-the-arts on two datesets. “BS" denotes the Beam Search strategy.

Methods	B-1	B-2	B-3	B-4	M	R	C	S
ARL (Wang et al., 2020)	75.9	60.3	46.5	35.8	27.8	56.4	111.3	-
KMSL (Li and Jiang, 2020)	79.2	63.2	48.3	36.3	27.6	56.8	120.2	21.4
JRAN (ours)	**81.0**	**64.7**	**49.8**	**38.6**	**28.3**	**58.3**	**128.4**	**22.1**

Table 3: Performance comparision of our JRAN with the similar relationship based methods on MSCOCO.

Methods	B-1	B-2	B-3	B-4	M	R	C	S
Transformer (Sharma et al., 2018)	80.2	64.8	**50.5**	38.6	28.8	58.5	128.3	**22.6**
VORN (Herdade et al., 2019)	80.5	-	-	38.6	28.7	58.4	128.3	**22.6**
LBPF (Qin et al., 2019)	80.5	-	-	38.3	28.5	58.4	127.6	22.0
Sim-Trans ♯	79.4	64.5	49.1	38.5	28.0	58.1	125.5	21.7
JRAN-Trans (ours) (BS=3)	**81.2**	**65.3**	50.1	**39.2**	**28.9**	**58.7**	129.4	**22.6**
JRAN-Trans (ours) (BS=4)	81.1	65.2	50.0	39.1	28.8	58.6	**129.6**	22.5
JRAN-Trans (ours) (BS=5)	80.9	65.0	49.8	39.0	28.5	58.5	129.3	22.2

Table 4: Performance comparisons with transformer based baseline model on MSCOCO dataset. ‘♯’ is the current mainstream transformer based baseline model.

upgraded model “JRAN-Trans" under different BS. Since the BLEU4 score of the original baselines “*Sim-Trans* " is 38.5, our upgraded transformer variation based model can effectively boost the score by 0.7. In addition, the CIDEr score is significantly increased from 125.5 to 129.6, which is clearly a meaningful improvement. Moreover, compared with the other current mainstream methods, our method still achieves very competitive performance across most metrics.

5.4 Model Accuracy and Efficiency

We further conduct an experimental computational cost analysis for comparing our updated model “JRAN-Trans" with some typical models (i.e., “SC (base) (Chen et al., 2017)", “Sim-Trans", and “X-Transformer" (Pan et al., 2020)). Figure 6 presents the computational cost in terms of the training time and parameters of the model.

As can be seen from Figure 6, the baseline model “SC (base)" has the less parameters and training

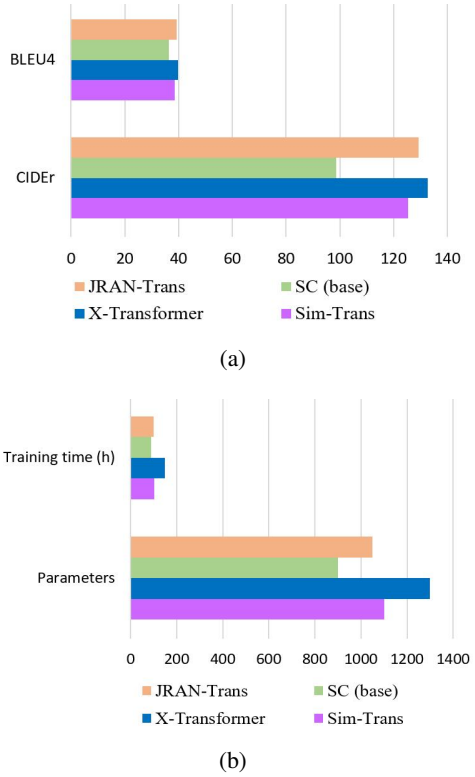

Figure 6: Illustration of the model computational cost (a) and model accuracy (b).

time, but the model accuracy is the lowest. One main reason is that it ignores the potential relationship between features in the image. In addition, the evaluation metric scores of the transformer based model “X-Transformer" are relatively high, while the parameter and training time are also relatively high.

Compared with the “X-Transformer" model, obviously, our “JRAN-Trans" does not significantly increase the computational cost of the model while achieving promising accuracy. This clearly indicates that our method displays a trade-off between

model accuracy and computational cost.

5.5 Qualitative Results

a. Comparison of Generating Sentences with Different Approaches: Establishing the interaction between visual and semantic information by learning the relationship among image features, our JRAN can generate comprehensive sentences more consistent with the image theme scene. Figure 7 shows some examples of sentence description generated by different baseline methods, namely Baseline (Li et al., 2018a), *Sim-Trans* and our JRAN-Trans.

Figure 7: Examples of generating sentences. Blue is region objects, purple is the background or environment information of the object in the image, the red is the corresponding visual relationship, and the underline indicates the corresponding visual-semantic relationship.

From these exemplar results, it is clearly see that the three methods can generate somewhat relevant and logically correct sentences, while our JRAN based method "JRAN-Trans" generates more consistent sentences with image theme scene. It effectively improves the quality of generated text by enriching visual-semantic relationships. For instance, for image (a), compared to the relationship words/phrases "playing" and "playing with" generated by methods "baselines" and "Sim-Trans" respectively. Our "JRAN-Trans" not only accurately generates the relationship phrase "staring at" between region objects "dogs", but also enrichs the relationship between region object "dog" and its background word "road", and generates an appropriate interactive word "on" between them, which significantly improves the model overall performance.

b. Effectiveness of Relationship Gate: Figure 8 visualizes the weight of relationship gate during generating different words. It can be seen that visual relationship and visual-semantic relationship contribute differently to the generation of different words. Specifically, the visual relationship has

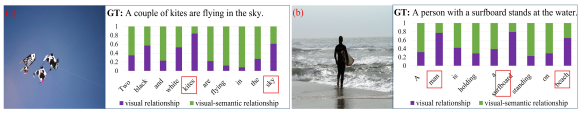

Figure 8: Visualization of relationship gate weight. The first column is original image, and the second column are the values of visual relationship weight (purple) and visual-semantic relationship weight (green).

a larger value when generating the visual words "kite" and "sky" (red boxes). Contrarily, when non-visual words (like "Two", "and", "flying" etc.) are generated, the visual-semantic relationship has greater value. This indicates that the two relationships complement each other and jointly boost the performance of image-text generation.

6 Conclusion

In the paper, a simple and effective model that makes full use of the complementary region and semantic features in the image, Joint Relationship Attention Network (JRAN) is proposed. It explores the relationship among the features to enrich the relationship-level representation for finally boosting image-text generation. To verify our claim, we propose a new joint relationship learning network, which is able to learn two kinds of feature relationships. Considering the different contributions of these two relationships during generating words, we further devise a relationship gate to finally obtain a feature representation containing different-level relationship information. Importantly, our model has made remarkable progress in deeply exploring the relationship between features for image-text generation. Extensive experiments demonstrate that the effectiveness of our proposed model on larger-scale and smaller-scale datasets. More remarkably, we obtain new state-of-the-art performances on popular Flickr30k dataset.

Acknowledgments

This work was supported by the National Natural Science Foundation of China under grant 62176062.

References

Peter Anderson, Basura Fernando, Mark Johnson, and Stephen Gould. 2016. Spice: Semantic propositional image caption evaluation. In *Proceedings European Conference on Computer Vision*, pages 382–398.

Peter Anderson, Xiaodong He, Chris Buehler, Damien Teney, Mark Johnson, Stephen Gould, and Lei Zhang. 2018. Bottom-up and top-down attention for image captioning and visual question answering. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, page 6077–6086.

Satanjeev Banerjee and Alon Lavie. 2005. Meteor: An automatic metric for mt evaluation with improved correlation with human judgments. In *Proceedings of Meeting of the Association for Computational Linguistics*, pages 65–72.

Long Chen, Hanwang Zhang, Jun Xiao, Liqiang Nie, Jian Shao, Wei Liu, and Tat-Seng Chua. 2017. Sca-cnn: Spatial and channel-wise attention in convolutional networks for image captioning. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, pages 5659–5667.

Marcella Cornia, Matteo Stefanini, Lorenzo Baraldi, and Rita Cucchiara. 2020. Meshed-memory transformer for image captioning. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, page 10578–10587.

Yan Duan, Marcin Andrychowicz, Bradly Stadie, Jonathan Ho, Jonas Schneider, Ilya Sutskever, Pieter Abbeel, and Wojciech Zaremba. 2017. One-shot imitation learning. In *arXiv:1703.07326*.

Junlong Gao, Shiqi Wang, Shanshe Wang, Siwei Ma, and Wen Gao. 2019. Self-critical n-step training for image captioning. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, page 6300–6308.

Xinwei He, Yang Yang, Baoguang Shi, and Xiang Bai. 2019. Vd-san: visual-densely semantic attention network for image caption generation. *Neurocomputing*, 328:48–55.

Simao Herdade, Armin Kappeler, Kofi Boakye, and Joao Soares. 2019. Image captioning: Transforming objects into words. In *Proceedings of Advances in Neural Information Processing Systems*, page 11135–11145.

Thomas N. Kipf and Max Welling. 2017. Semi-supervised classification with graph convolutional networks. In *Proceedings of International Conference on Learning Representations*.

Alina Kuznetsova, Hassan Rom, Neil Alldrin, Jasper Uijlings, and et al. Ivan Krasin. 2018. The open images dataset v4: Unified image classification, object detection, and visual relationship detection at scale. *arXiv preprint arXiv:1811.00982*.

Linghui Li, Sheng Tang, Yongdong Zhang, Lixi Deng, and Qi Tian. 2018a. Gla: Global-local attention for image description. *IEEE Transactions on Multimedia*, 20(3):726–737.

Ruoyu Li, Sheng Wang, Feiyun Zhu, and Junzhou Huang. 2018b. Adaptive graph convolutional neural networks. In *Proceedings of the Thirty-Second AAAI Conference on Artificial Intelligence*, pages 3546–3553.

Xiangyang Li and Shuqiang Jiang. 2020. Know more say less: Image captioning based on scene graphs. *IEEE Transactions on Multimedia*, 20(8):2117 – 2130.

Chin-Yew Lin. 2004. Rouge: A package for automatic evaluation of summaries. In *Proceedings of Meeting of the Association for Computational Linguistics*, pages 74–81.

Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C. Lawrence Zitnick. 2014. Microsoft coco: common objects in context. In *Proceedings of European Conference on Computer Vision*, pages 740–755.

Yingwei Pan, Ting Yao, Yehao Li, and Tao Mei. 2020. X-linear attention networks for image captioning. In *Proceedings of the 2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 10968–10977.

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. Bleu: a method for automatic evaluation of machine translation. In *Proceedings of Meeting of the Association for Computational Linguistics*, pages 311–318.

Bryan A. Plummer, Liwei Wang, Chris M. Cervantes, Juan C. Caicedo, Julia Hockenmaier, and Svetlana Lazebnik. 2015. Flickr30k entities: Collecting region-to-phrase correspondences for richer image-to-sentence models. In *Proceedings of International Conference on Computer Vision*, pages 2641–2649.

Yu Qin, Jiajun Du, and Hongtao Zhang, Yonghuaand Lu. 2019. Look back and predict forward in image captioning. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, page 8367–8375.

Piyush Sharma, Nan Ding, Sebastian Goodman, and Radu Soricut. 2018. Conceptual captions: A cleaned, hypernymed, image alt-text dataset for automatic image captioning. In *Proceedings of 56th Annual Meeting of the Association-for-Computational-Linguistics*, pages 2556–2565.

Jingkuan Song, Pengpeng Zeng, Lianli Gao, and Heng Tao Shen. 2018. From pixels to objects: Cubic visual attention for visual question answering. In *Proceedings of the International Joint Conference on Artificial Intelligence*, pages 906–912.

Sahar Tahvili, Leo Hatvani, Enislay Ramentol, Rita Pimentel, Wasif Afzal, and Francisco Herrera. 2020. A novel methodology to classify test cases using natural language processing and imbalanced learning. *Engineering Applications of Artificial Intelligence*, 95:103878.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *arXiv preprint arXiv:1706.03762*.

Ramakrishna Vedantam, C. Lawrence Zitnick, and Devi Parikh. 2015. Cider: Consensus-based image description evaluation. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, pages 4566–4575.

Cheng Wang, Haojin Yang, Christian Bartz, and Christoph Meinel. 2016. Image captioning with deep bidirectional lstms. In *Proceedings of the 2016 ACM Multimedia Conference*, pages 988–997.

Junbo Wang, Wei Wang, Liang Wang, Zhiyong Wang, David Dagan Feng, and Tieniu Tan. 2020. Learning visual relationship and context-aware attention for image captioning. *Pattern Recognition*, 98:107075.

Weixuan Wang, Zhihong Chen, and Haifeng Hu. 2019. Hierarchical attention network for image captioning. In *Proceedings of the AAAI Conference on Artificial Intelligence*, pages 8957–8964.

Jie Wu, Tianshui Chen, Hefeng Wu, Zhi Yang, Guangchun Luo, and Liang Lin. 2021a. Fine-grained image captioning with global-local discriminative objective. *IEEE Transactions on Multimedia*, 23:2413–2427.

Lingxiang Wu, Min Xu, Lei Sang, Ting Yao, and Tao Mei. 2021b. Noise augmented double-stream graph convolutional networks for image captioning. *IEEE Transactions on Circuits and Systems for Video Technology*, 31(8):3118–3127.

Kelvin Xu, Jimmy Ba, Ryan Kiros, Kyunghyun Cho, Aaron Courville, Ruslan Salakhutdinov, Richard Zemel, and Yoshua Bengio. 2015. Show, attend and tell: Neural image caption generation with visual attention. In *Proceedings of the International Conference on Machine Learning*, pages 2048–2057.

Liang Yang, Haifeng Hu, Songlong Xing, and Xinlong Lu. 2020. Constrained lstm and residual attention for image captioning. *ACM Transactions on Multimedia Computing, Communications, and Applications*, 16(3):1–18.

Ting Yao, Yingwei Pan, Yehao Li, and Tao Mei. 2019. Hierarchy parsing for image captioning. In *Proceedings of the IEEE International Conference on Computer Vision*, page 2621–2629.

Senmao Ye, Junwei Han, and Nian Liu. 2018. Attentive linear transformation for image captioning. *IEEE Transactions on Image Processing*, 27(11):5514–5524.

Yu Zhang, Xinyu Shi, Siya Mi, and Xu Yang. 2021. Image captioning with transformer and knowledge graph. *Pattern Recognition Letters*, 143:43–49.

Learning to Focus on the Foreground for Temporal Sentence Grounding

Daizong Liu and **Wei Hu***
Wangxuan Institute of Computer Technology, Peking University
dzliu@stu.pku.edu.cn, forhuwei@pku.edu.cn

Abstract

Temporal sentence grounding (TSG) is crucial and fundamental for video understanding. Previous works typically model the target activity referred to the sentence query in a video by extracting the appearance information from each whole frame. However, these methods fail to distinguish visually similar background noise and capture subtle details of small objects. Although a few recent works additionally adopt a detection model to filter out the background contents and capture local appearances of foreground objects, they rely on the quality of the detection model and suffer from the time-consuming detection process. To this end, we propose a novel detection-free framework for TSG—**G**rounding with **L**earnable **F**oreground (GLF), which efficiently learns to locate the foreground regions related to the query in consecutive frames for better modelling the target activity. Specifically, we first split each video frame into multiple patch candidates of equal size, and reformulate the foreground detection problem as a patch localization task. Then, we develop a self-supervised coarse-to-fine paradigm to learn to locate the most query-relevant patch in each frame and aggregate them among the video for final grounding. Further, we employ a multi-scale patch reasoning strategy to capture more fine-grained foreground information. Extensive experiments on three challenging datasets (Charades-STA, TACoS, ActivityNet) show that the proposed GLF outperforms state-of-the-art methods.

1 Introduction

Temporal sentence grounding (TSG) (Gao et al., 2017; Anne Hendricks et al., 2017) is an important yet challenging topic of video understanding in computer vision. Given an untrimmed video, it aims to retrieve a temporal segment that semantically corresponds to a given sentence query, as shown in Figure 1 (a). Compared to other video-and-language tasks like video captioning (Song et al., 2015; Chu et al., 2015) and video action localization (Shou et al., 2016; Zhao et al., 2017; Xiong et al., 2022), TSG is substantially more challenging as it need not only capture the complicated visual and textual information, but also learn the complex multi-modal interactions among them for modelling the target activity.

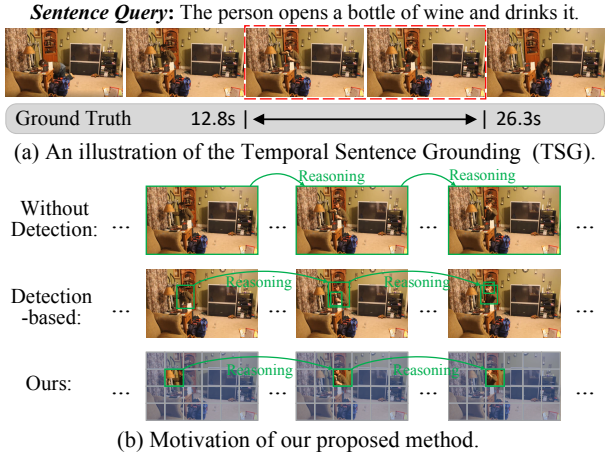

(a) An illustration of the Temporal Sentence Grounding (TSG).

(b) Motivation of our proposed method.

Figure 1: (a) An illustrative example of the TSG task. (b) The Illustration of our motivation: we learn to selectively focus on the foreground patch in each frame in a detection-free manner, which alleviates the problems of redundant backgrounds in most previous works (without detection) and the low detection quality in time-consuming detection-based methods. The green box marks the focused region.

To localize the target segment, most previous works either pre-define abundant segment proposals (Chen et al., 2018; Zhang et al., 2019; Yuan et al., 2019; Liu et al., 2021b; Zhang et al., 2020b; Zeng et al., 2020; Mo et al., 2022; Liu et al., 2022a) to match the query semantic for ranking and selection, or employ proposal-free frameworks (Chen et al., 2020; Zhang et al., 2020a; Mun et al., 2020)

*Corresponding author.

Proceedings of the 29th International Conference on Computational Linguistics, pages 5532–5541
October 12–17, 2022.

to directly regress the start/end timestamps of the segment. Although these methods have made significant progress in recent years, they extract the appearance information of each whole frame among the entire video, thus limiting the effective integration of the foreground contexts for modelling the target activity due to the visually similar backgrounds and the missing subtle details of small objects. To alleviate such limitations, a few recent works (Zeng et al., 2021; Liu et al., 2022e) attempt to additionally adopt a pre-processing detection model (*i.e.*, Faster R-CNN (Ren et al., 2015)) that detects the foreground objects for filtering out the background noise. However, they rely on the quality of the detection model while suffering from the time-consuming detection process.

Based on the above considerations, this paper aims to develop a *detection-free* grounding network, which efficiently selects the most query-relevant region in each frame to represent the frame-level features among the entire video for better modelling the target activity. As illustrated in Figure 1(b), in order to effectively represent different local regions in each frame, we divide it into multiple patches that serve as the region candidates to be selected according to their semantic similarity with the query. Once the best patch is determined in each frame during the network learning, they are extracted to model the activity by spatial-temporal correlation reasoning. Compared to previous methods, such detection-free network provides more fine-grained foreground details by filtering out the background regions and capturing the local contexts in an efficient and end-to-end manner, leading to better grounding results.

To this end, we propose a novel TSG model, called **G**rounding with **L**earnable **F**oreground (GLF), which learns to focus on the query-relevant foreground regions among video frames to model the fine-grained target activity for more accurate grounding. Specifically, we reformulate the foreground detection problem as a patch localization task. Considering the spatial-temporal information within the video, we extract 3D spatial-temporal patches instead of 2D spatial ones on the video clips (*i.e.*, several consecutive frames). We first introduce a 3D patch embedding layer to encode the local information of each patch candidate, and concatenate it with an additional global representation extracted from its current video clip. Then, we interact the patch candidates in each clip with the query semantic to learn their matching scores for distinguishing the foreground and background patches. Particularly, we develop a two-level coarse-to-fine paradigm to gradually localize the most relevant (foreground) patch in each clip. At last, we aggregate the representations of the most relevant patches among the entire video to model the target activity. In addition, considering the sizes of the foreground regions may vary in different videos, we further extend the GLF model with multi-scale patch design to capture more fine-grained and complete foreground information for better grounding.

Our main contributions are summarized as follows:

- To the best of our knowledge, we are the first to propose a detection-free grounding framework with the foreground learned for the TSG task. To learn to determine the foreground region in each clip, we split the clip into multiple patch candidates and reformulate the foreground detection problem as a patch localization task.
- We propose a coarse-to-fine self-supervised paradigm to localize the most query-relevant region in each clip for final grounding. We further extend the paradigm with multi-scale patch reasoning in a parallel manner to capture more fine-grained foreground details.
- Comprehensive evaluations on three challenging TSG benchmarks (Charades-STA, TACoS, ActivityNet) demonstrate that our GLF outperforms the state-of-the-art performance.

2 Related Work

Temporal sentence grounding (TSG) is a new task introduced recently (Gao et al., 2017; Anne Hendricks et al., 2017), which aims to localize the most relevant video segment from a video with sentence descriptions. Most previous works (Chen et al., 2018; Zhang et al., 2019; Yuan et al., 2019; Liu et al., 2021b, 2020b, 2021a,c, 2022b; Liu and Hu, 2022) generate multiple segment proposals and then rank them according to the similarity between proposals and the query for selecting the best matching one. Instead of generating complex proposals, some works (Zhang et al., 2020a; Chen et al., 2020; Mun et al., 2020; Liu et al., 2022d,c) directly regress the temporal locations of the target

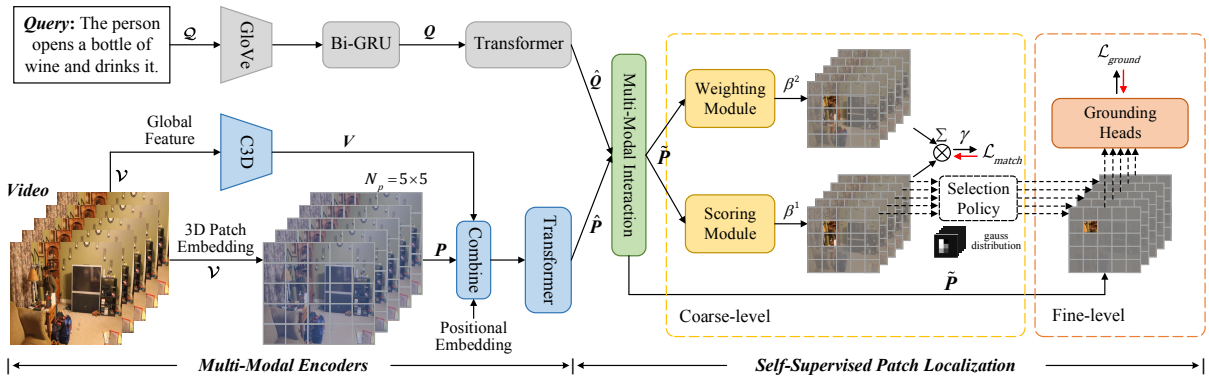

Figure 2: Overview of the proposed GLF model. It consists of the multi-modal encoders and the self-supervised patch localization.

segment by either regressing the start/end timestamps based on the entire video representation or predicting at each frame to determine whether this frame is a start or end boundary. Although the above two types of methods achieve outstanding performance, they all extract the appearance information of each whole frame among the entire video for activity modelling, which fails to capture fine-grained local object details for semantic composing and may suffer from visually similar background contents. A few recent works (Zeng et al., 2021; Liu et al., 2022e) attempt to alleviate such limitations by detecting and learning the correlations between the foreground objects for reasoning the multi-modal semantics. These methods can well filter out the background noise and focus more on local details of small objects. However, they rely on the quality of the time-consuming detection model. In this paper, we propose a detection-free grounding network to learn to focus on the foreground region in each frame for activity composing, which is more efficient than the detection-based methods since our model is trained end-to-end with a learnable foreground attention mechanism. Besides, our model is also more effective than previous proposal-based and proposal-free methods by filtering out the background appearances and capturing more fine-grained subtle details.

3 Our Method

Given an untrimmed video $\boldsymbol{\mathcal{V}}$ and a sentence query $\boldsymbol{\mathcal{Q}}$, the TSG task aims to determine the start and end timestamps of a specific video segment referring to the sentence query. Formally, we represent the video as $\boldsymbol{\mathcal{V}} = \{\boldsymbol{c}_t\}_{t=1}^{T}$ clip-by-clip, where $\boldsymbol{c}_t$ is the t-th clip and T is the total clip number. We also denote the query as $\boldsymbol{\mathcal{Q}} = \{\boldsymbol{w}_s\}_{s=1}^{S}$ word-by-word, where S is the length of the sentence.

In this section, we introduce the overall architecture of our proposed GLF model. As shown in Figure 2, the model consists of two main parts: multi-modal encoders and self-supervised patch localization. First of all, GLF splits each video clip into multiple spatial-temporal patch candidates of equal size and encodes them with a shared 3D embedding layer. Then, the model extracts the query embeddings and interact them with all patch candidates. After that, a coarse-to-fine patch localization paradigm is proposed to gradually score the patches in each clip according to their query-relevant similarity. At last, a single best patch in each clip is selected by a learned policy module to represent the current query-guided clip feature for final grounding. Considering the sizes of the foreground regions may vary in different videos, we further extend the model with a multi-scale patch design to aggregate different-level foreground contexts. We elaborate on each module below.

3.1 Multi-Modal Encoders

Video encoder. For the input video $\boldsymbol{\mathcal{V}}$, to extract different regional local information in the t-th clip $\boldsymbol{c}_t$, we first split $\boldsymbol{c}_t$ into spatial-temporal patch candidates with the same temporal dimension as $\boldsymbol{c}_t$ and no spatial overlap, where the total number of spatial-temporal patches is $N_p = K \times K$ and K is the patch number in each column/row. Then, we take a shared-weight 3D kernel with a further projection layer as the patch embedding module to encode all N_p patches of $\boldsymbol{c}_t$ into $\{\boldsymbol{p}_{t,i}\}_{i=1}^{N_p}$, where $\boldsymbol{p}_{t,i} \in \mathbb{R}^{d_1^v}$ and i denotes the patch index, and d_1^v is the feature dimension. Considering the global feature of the whole clip contains the non-local information across different patches, we also utilize

the pre-trained C3D model (Tran et al., 2015) to extract clip-level feature $\boldsymbol{v}_t \in \mathbb{R}^{d_2^v}$ of each clip $\boldsymbol{c}_t$.

Since it is necessary to consider both spatial and temporal locations of each patch for reasoning patch-wise relations, we follow (Dosovitskiy et al., 2021) to encode the spatial position embeddings of each patch $\boldsymbol{p}_{t,i}$ as $\boldsymbol{e}_i^{spa}$, and follow (Mun et al., 2020) to define its temporal position embeddings as $\boldsymbol{e}_t^{tem}$. The final patch-wise feature is concatenated as:

$$(\boldsymbol{p}_{t,i})' = [\boldsymbol{p}_{t,i}; \boldsymbol{v}_t; \boldsymbol{e}_i^{spa}; \boldsymbol{e}_t^{tem}]. \tag{1}$$

We further employ a plain Transformer encoder (Vaswani et al., 2017) on all patches $\{(\boldsymbol{p}_{t,i})'\}_{t=1,i=1}^{t=T,i=N_p}$ to model their intra-modality contexts, and obtain corresponding contextualized representations as $\widehat{\boldsymbol{P}} = \{\widehat{\boldsymbol{p}}_{t,i}\}_{t=1,i=1}^{t=T,i=N_p} \in \mathbb{R}^{T\times N_p\times d_3^v}$.

Query encoder. For the query $\boldsymbol{\mathcal{Q}}$, following previous works (Chen et al., 2018; Liu et al., 2021b), we first utilize the Glove model (Pennington et al., 2014) to embed each word into a dense vector, and then employ a Bi-GRU (Chung et al., 2014) to encode its sequential information. The encoded features can be denoted as $\boldsymbol{Q} = \{\boldsymbol{q}_s\}_{s=1}^S \in \mathbb{R}^{S\times d_1^q}$. We also employ another plain Transformer encoder to model the contextual textual representations as $\widehat{\boldsymbol{Q}} = \{\widehat{\boldsymbol{q}}_s\}_{s=1}^S \in \mathbb{R}^{S\times d_2^q}$.

3.2 Self-Supervised Patch Localization

Since there is only temporal-level annotation of the target segment and no spatial-level annotation of the foreground regions, we develop a self-supervised learning paradigm to guide the GLF learn to focus on the potential foreground patches. Specifically, we first interact video and query features to align their relevant semantics, and then impose two cooperated modules on the multi-modal features to compute the clip-query matching scores by scoring and weighting different patches in the same clip. We further propose a coarse-to-fine patch localization strategy to gradually select a patch in each clip to effectively represent the clip-level query-relevant semantic for more fine-grained and accurate grounding.

Multi-modal interaction. To capture the relationship between each patch and the query, we employ a multi-modal interaction module that selectively injects textual evidences into the visual patches. We first utilize an attention mechanism to aggregate the word features for each patch. For the patch $\widehat{\boldsymbol{p}}_{t,i}$, we calculate the attention weights over word features $\{\widehat{\boldsymbol{q}}_s\}_{s=1}^S$ and aggregate them as:

$$\begin{aligned} \alpha_{t,i,s} &= \mathbf{w}^\top \tanh(\mathbf{W}_1^\alpha \widehat{\boldsymbol{p}}_{t,i} + \mathbf{W}_2^\alpha \widehat{\boldsymbol{q}}_s + \mathbf{b}^\alpha), \\ \boldsymbol{r}_{t,i} &= \sum_{s=1}^{S} \mathrm{softmax}(\alpha_{t,i,s}) \cdot \widehat{\boldsymbol{q}}_s, \end{aligned} \tag{2}$$

where $\mathbf{W}_1^\alpha, \mathbf{W}_2^\alpha$ are projection matrices, $\mathbf{b}^\alpha$ is the bias and $\mathbf{w}^\top$ is the row vector (Zhang et al., 2019). $\boldsymbol{r}_{t,i}$ is the patch-aware textual feature for each patch i in the t-th clip. Next, we build the textual gate that takes language information as the guidance to weaken the text-irrelevant patches, and generate the cross-modal patch features as:

$$\boldsymbol{g}_{t,i} = \sigma(\mathbf{W}^g \boldsymbol{r}_{t,i} + \mathbf{W}^b),\ \widetilde{\boldsymbol{p}}_{t,i} = [\widehat{\boldsymbol{p}}_{t,i} \odot \boldsymbol{g}_{t,i}; \boldsymbol{r}_{t,i}], \tag{3}$$

where σ is the sigmoid function, $\odot$ is the element-wise multiplication, $\boldsymbol{g}_{t,i}$ means the textual gate for patch i. $\widetilde{\boldsymbol{P}} = \{\widetilde{\boldsymbol{p}}_{t,i}\}_{t=1,i=1}^{t=T,i=N_p}$ is the query-guided patch features.

Learning to focus on the foreground. We propose a self-supervised learning paradigm to estimate the potential foreground patches by learning to selectively aggregate the patch information within each clip for clip-query matching. Considering patches of the same clip have different semantic similarities with the query and their contribution to the query-guided clip-level semantic is often quite different, we develop two separate scoring and weighting modules to evaluate the patch-query similarity and patch-to-clip weight, respectively. Both modules are implemented by two linear layers. For the patches in the t-th clip, we formulate the self-supervised learning process as:

$$\begin{aligned} \beta_{t,i}^1 &= \mathrm{scoring}(\widetilde{\boldsymbol{p}}_{t,i}),\ \beta_{t,i}^2 = \mathrm{weighting}(\widetilde{\boldsymbol{p}}_{t,i}), \\ \gamma_t &= \sum_{i=1}^{N_p} \mathrm{softmax}(\beta_{t,i}^2) \cdot \sigma(\beta_{t,i}^1), \end{aligned} \tag{4}$$

where $\beta_{t,i}^1$ represents the score whether the i-th patch in the t-th clip is the query-relevant one, $\beta_{t,i}^2$ is the predicted weight for aggregating all patches within the current clip. γ_t denotes the final clip-query matching score, which represents whether the t-th clip is in the ground-truth segment or not. To prevent the two modules merging into the similar or identical parameters, we utilize the sigmoid function following the scoring module to force it to learn the score whether the patch matches the query, and we utilize the softmax function following the weighting module to predict the weights for

aggregating all patches within the current frame. Once the clip-level scores are well-trained, the scoring module can best predict the similarity between each patch and the query, and guide the model to focus more on the foreground patches.

To supervise the above two modules, we use the clips falling into the ground-truth segment as positive samples and the others as negative samples, and formulate a balanced binary cross-entropy loss as:

$$\begin{aligned}\mathcal{L}_{match} = &-\sum_{t=1}^{T_{pos}} \frac{T_{neg}}{T} y_t \log(\gamma_t) \\ &-\sum_{t=1}^{T_{neg}} \frac{T_{pos}}{T}(1-y_t)\log(1-\gamma_t),\end{aligned} \tag{5}$$

where T_{pos}, T_{neg} are the numbers of positive and negative clips. Since most videos are long while the lengths of annotated target segments is short, the numbers of positive and negative clips are unbalance. Therefore, we utilize Tneg/T and Tpos/T to balance their losses. y_t is the ground-truth label that equals to 1 for positive samples and 0 for negative samples.

Coarse-to-fine patch localization. Equation (4) is a vanilla solution to aggregate the potential foreground contexts by patch-wise scoring and weighting. Since the query-related activity mainly appears in one small region of each clip, based on the above coarse foreground localization operation, we further design a fine-level localization module to only select one patch feature to represent its clip-level query-guided semantic for grounding. Specifically, we develop a selection policy module to choose a single patch from a Gaussian distribution which is transformed from the previous predicted patch-wise scores in each clip. There is no learnable parameter for this module, and the patch with a higher patch-wise score will get a larger probability to represent the frame. We denote such generated clip-level representations as $\boldsymbol{F} = \{\boldsymbol{f}_t\}_{t=1}^{T}$, where $\boldsymbol{f}_t$ is the feature of the selected patch in the t-th clip. After that, we apply the effective grounding heads following (Zhang et al., 2019; Liu et al., 2020b,a) on $\boldsymbol{F}$ to generate N_Φ fine-grained segment proposals for ranking via both confidence scoring loss $\mathcal{L}_{iou}$ and boundary adjustment loss $\mathcal{L}_b$ as:

$$\mathcal{L}_{iou} = -\frac{1}{N_\Phi}\sum_{i=1}^{N_\Phi}(IoU_i log(cs_i) + (1-IoU_i) log(1-cs_i)), \tag{6}$$

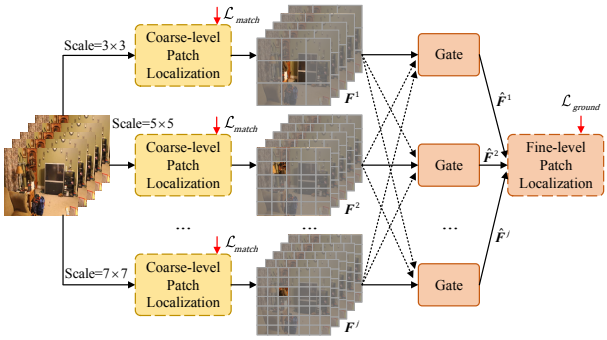

Figure 3: Illustration of the multi-scale patch reasoning strategy.

$$\mathcal{L}_b = \frac{1}{N_{pos}}\sum_{j}^{N_{pos}} \mathcal{R}_1(\hat{\delta}_j^s - \delta_j^s) + \mathcal{R}_1(\hat{\delta}_j^e - \delta_j^e), \tag{7}$$

where cs_i is the predicted confidence score of each segment proposal, $IoU_{t,i}$ is corresponding ground-truth. N_{pos} denotes the number of the positive proposals, and $\mathcal{R}_1$ is the smooth L1 loss. Therefore, the grounding loss can be formulated with a balanced parameter λ as follows:

$$\mathcal{L}_{ground} = \mathcal{L}_{iou} + \lambda \mathcal{L}_b. \tag{8}$$

Let $\phi_{t,i}$ denote the probability of patch $\widetilde{\boldsymbol{p}}_{t,i}$ being selected, the goal of this selection policy module is to minimize $\sum \phi_{t,i} \cdot \mathcal{L}_{ground}$, where $\mathcal{L}_{ground}$ is the reward to enforce it to select the patch that enables the network to produce correct grounding in high confidence. In this manner, the coarse-level localization module gradually finds the important patches of each clip and yields a better clip-wise feature representation. Meanwhile, the selected foreground patch and corresponding representation can further lead to more precise grounding results in the fine-level localization module that in turn provides better supervisions for patch-wise scoring at the coarse level.

3.3 Multi-Scale Patch Aggregation

In order to obtain more reliable foreground information among video clips for final grounding, we exploit the multi-scale property to fuse the contents of the best patch with multiple scales in each clip. Specifically, we split the same video clip into different numbers of patch, and then separately train different patch localization modules for different patch scales. Since a patch with a smaller scale captures major local details and a patch with a larger

scale preserves more global contexts, fusing the information from multi-scale best patches in the same clip leads to more representative foreground features. However, directly fusing the multi-scale results cannot take full advantage of the complementary information in different scales. Therefore, we propose a gate-based multi-scale aggregation module to distill each scale patch information for better fusion. Details are illustrated in Figure 3.

For coarse-level patch localization, we extend the multi-scale strategy by learning the self-supervised process in Equation (4) with different patch scales, respectively. For fine-level patch localization, we first separately select one patch in each clip t with multiple scales as $\boldsymbol{f}_t^j$, where j denotes the scale index and $j \in \mathcal{J}$ which is empirically defined. Then, we generate a distilled gate $\boldsymbol{g}_{1,t}^j$ and a reset gate $\boldsymbol{g}_{2,t}^j$ which play a similar role to the gates in LSTM. The gates at each scale control how much the feature at each scale contributes to the final fused feature. This process can be formulated as:

$$
\begin{aligned}
(\boldsymbol{f}_t^j)' &= (1-\boldsymbol{g}_{1,t}^j) \odot \boldsymbol{f}_t^j + \sum_{j' \in \mathcal{J}, j' \neq j} \eta^{j',j} \boldsymbol{g}_{1,t}^j \odot \boldsymbol{f}_t^{j'}, \\
\widehat{\boldsymbol{f}}_t^j &= \boldsymbol{g}_{2,t}^j \odot \tanh((\boldsymbol{f}_t^j)') + (1-\boldsymbol{g}_{2,t}^j) \odot \boldsymbol{f}_t^j, j \in \mathcal{J},
\end{aligned} \tag{9}
$$

where $\eta^{j',j}$ is a learnable parameter to adjust the relative ratio of the distilled gate which controls information flow of features from a different scale j' combined with the current scale j. $\widehat{\boldsymbol{f}}_t^j$ is the updated patch feature at scale index j, and we concatenate $\{\widehat{\boldsymbol{f}}_t^j\}_{j=1}^{\mathcal{J}}$ of all scales as the fused features $\widetilde{\boldsymbol{f}}_t$ and send it to the grounding heads in Equation (8).

3.4 Training and Testing

Training. To ensure our proposed GLF is trained properly, we propose a three-stage training scheme. At the first stage, we do not integrate the fine-level patch localization module into GLF. Instead, we train the coarse-level one with multi-scale patch definition by minimizing the loss $\mathcal{L}_{match}$ in Equation (5). In this stage, the network is trained to score the foreground patches. At the second stage, we fix the trained network obtained from stage-1, and evoke the fine-level patch localization module with the multi-scale strategy to focus on the selected patch in each clip by minimizing the loss $\mathcal{L}_{ground}$ in Equation (8). At last, we fine-tune the whole GLF model.

Testing. During testing, we select patches of highest scores in the fine-level patch localization module for grounding.

4 Experiments

4.1 Datasets and Evaluation

Charades-STA. This dataset (Gao et al., 2017) consists of 9848 videos of daily life indoor activities. There are 12408 sentence-video pairs for training and 3720 pairs for testing.

TACoS. This dataset (Regneri et al., 2013) collects 127 long videos, which are mainly about cooking scenarios, thus lacking the diversity. We use the same split as [Gao et al., 2017], which has 10146, 4589 and 4083 sentence-video pairs for training, validation, and testing, respectively.

ActivityNet. It is a large dataset (Krishna et al., 2017) which contains 20k videos with 100k language descriptions. This dataset pays attention to more complicated human activities in daily life. Following public split, we use 37417, 17505, and 17031 sentence-video pairs for training, validation, and testing, respectively.

Evaluation. We adopt "R@n, IoU=m" as our evaluation metric, which is defined as the percentage of at least one of top-n selected moments having IoU larger than m.

4.2 Implementation Details

For each video input, we adopt 112×112 pixels of every frame. We define consecutive 16 frames as a clip and each clip overlaps 8 frames with adjacent clips. The kernel size of the 3D patch embedding layer is adaptive to the defined patch size. We extract clip-level global features from a pre-trained C3D (Tran et al., 2015) or I3D (Carreira and Zisserman, 2017) model. Since some videos are overlong, we uniformly downsample clip sequences to $T = 200$ for TACoS, ActivityNet, and $T = 64$ for Charades-STA. For each sentence input, we set the length of word feature sequences to $S = 20$, and utilize Glove (Pennington et al., 2014) to embed each word to 300 dimension features. The hidden dimension of Bi-GRU is 512, and the hyper-parameter λ is set to 0.005. The numbers $N_p = K \times K$ of the split multi-scale patches in each clip are set to 3×3, 5×5, 7×7. We train the whole model with batch size of 64 and early stopping strategy. Parameter optimization is performed by Adam optimizer with leaning rate 4×10^{-4} for Charades-STA and 3×10^{-4} for TACoS, ActivityNet, and linear decay of learning rate and gradient clipping of 1.0.

Method	Charades-STA					TACoS				
	Feature	R@1, IoU=0.5	R@1, IoU=0.7	R@5, IoU=0.5	R@5, IoU=0.7	Feature	R@1, IoU=0.3	R@1, IoU=0.5	R@5, IoU=0.3	R@5, IoU=0.5
CTRL	C3D	23.63	8.89	58.92	29.57	C3D	18.32	13.30	36.69	25.42
QSPN	C3D	35.60	15.80	79.40	45.50	C3D	20.15	15.32	36.72	25.30
CBP	C3D	36.80	18.87	70.94	50.19	C3D	27.31	24.79	43.64	37.40
GDP	C3D	39.47	18.49	-	-	C3D	24.14	-	-	-
VSLNet	I3D	47.31	30.19	-	-	C3D	29.61	24.27	-	-
IVG-DCL	I3D	50.24	32.88	-	-	C3D	38.84	29.07	-	-
DRN	I3D	53.09	31.75	89.06	60.05	C3D	-	23.17	-	33.36
CBLN	I3D	61.13	38.22	90.33	61.69	C3D	38.98	27.65	59.96	46.24
MARN*	C3D+Object	62.08	41.46	91.65	70.03	C3D+Object	43.24	32.70	61.33	51.59
MARN*	I3D+Object	64.31	42.82	93.30	71.76	I3D+Object	45.57	34.06	62.64	52.92
GLF	C3D+Patch	63.60	42.75	92.91	71.49	C3D+Patch	44.82	34.38	62.75	52.26
	I3D+Patch	**65.57**	**44.32**	**94.86**	**73.07**	I3D+Patch	**47.14**	**35.63**	**65.24**	**53.77**

Table 1: Overall performance comparison among our method with proposal-based and proposal-free methods on the Charades-STA and TACoS datasets under the official train/test splits. * denotes that we remove MARN's additional ResNet feature for fair comparison.

Method	Charades-STA					TACoS				
	Feature	R@1, IoU=0.5	R@1, IoU=0.7	R@5, IoU=0.5	R@5, IoU=0.7	Feature	R@1, IoU=0.3	R@1, IoU=0.5	R@5, IoU=0.3	R@5, IoU=0.5
MMRG	Object	44.25	-	60.22	-	Object	57.83	39.28	78.38	56.34
MARN*	C3D+Object	46.19	32.01	63.25	39.88	C3D+Object	59.56	40.47	80.30	58.74
MARN*	I3D+Object	47.67	33.49	65.02	40.51	I3D+Object	61.16	42.33	82.75	59.90
GLF	C3D+Patch	47.85	33.68	64.54	41.20	C3D+Patch	61.37	41.72	81.96	59.45
	I3D+Patch	**49.59**	**35.01**	**66.34**	**42.79**	I3D+Patch	**62.98**	**43.11**	**83.52**	**60.13**

Table 2: Comparison with detection-based method MMRG on Charades-STA and TACoS datasets under MMRG's train/test splits.

Method	ActivityNet				
	Feature	R@1, IoU=0.5	R@1, IoU=0.7	R@5, IoU=0.5	R@5, IoU=0.7
CTRL	C3D	29.01	10.34	59.17	37.54
QSPN	C3D	33.26	13.43	62.39	40.78
CBP	C3D	35.76	17.80	65.89	46.20
GDP	C3D	39.27	-	-	-
VSLNet	C3D	43.22	26.16	-	-
IVG-DCL	C3D	43.84	27.10	-	-
DRN	C3D	45.45	24.36	77.97	50.30
CBLN	C3D	48.12	27.60	79.32	63.41
Ours	C3D+Patch	51.35	30.97	83.26	67.32
	I3D+Patch	**53.48**	**32.15**	**85.02**	**68.81**

Table 3: Overall performance comparison on the ActivityNet dataset under the official train/test splits.

	CTRL	DRN	CBLN	MARN (+detection)	**GLF**
Speed	2.23s	0.15s	0.18s	0.13s (+19.64s)	0.17s

Table 4: Seconds per video on TACoS dataset.

4.3 Comparisons with the State-of-the-art

Compared Methods. To demonstrate the effectiveness of GLF, we compared it with several state-of-the-art methods: Traditional: CTRL (Gao et al., 2017), QSPN (Xu et al., 2019), DRN (Zeng et al., 2020), CBLN (Liu et al., 2021b), CBP (Wang et al., 2020), GDP (Chen et al., 2020), VSLNet (Zhang et al., 2020a), IVG-DCL (Nan et al., 2021); Detection-based: MMRG (Zeng et al., 2021), MARN (Liu et al., 2022e). In particular, the MARN model relies on many types of feature inputs (*i.e.*, C3D, Object, ResNet) for better representation learning. Specifically, their object feature is extracted by detection model, and their ResNet model is utilized to encode such object contexts. Compared to MARN, we only feed single C3D feature as input. Since our method is proposal-free, we re-implement and remove their detector and ResNet models as a new variant MARN* to make a fair comparison with our method.

Comparison on Charades-STA. As shown in Table 1, we reach the highest results over all evaluation metrics on the Charades-STA dataset. Particularly, our C3D+Patch variant outperforms the best detection-based method MARN* by 1.29% and 1.46% in terms of R@1, IoU=0.7 and R@5, IoU=0.7, respectively. Compared to I3D+Patch variant of MARN*, our model also outperforms it by 1.50% and 1.31% in terms of R@1, IoU=0.7 and R@5, IoU=0.7, respectively. We also compare our model following the same data splits of MMRG in Table 2 for fair comparison. It shows that our

Multi-modal Encoders	Self-supervised Patch Localization		Multi-scale Strategy	R@1, IoU=0.7	R@5, IoU=0.7
	$\mathcal{L}_{match}$	$\mathcal{L}_{ground}$			
✓	×	×	×	33.71	62.35
✓	✓	×	×	37.28	65.86
✓	✓	×	✓	40.44	69.17
✓	✓	✓	×	39.53	68.20
✓	✓	✓	✓	**42.75**	**71.49**

Table 5: Main ablation study on Charades-STA dataset.

Components	Variants	R@1, IoU=0.7	R@5, IoU=0.7
Video Encoder	w/o global feature	40.47	70.13
	w/ global feature	**42.75**	**71.49**
	w/o position encoding	39.86	68.94
	w/ position encoding	**42.75**	**71.49**
	w/o transformer	40.38	69.82
	w/ transformer	**42.75**	**71.49**
Query Encoder	w/o Bi-GRU	41.64	70.71
	w/ Bi-GRU	**42.75**	**71.49**
	w/o transformer	41.53	70.70
	w/ transformer	**42.75**	**71.49**

Table 6: Ablation study on the multi-modal encoders.

GLF leads to large improvement.

Comparison on TACoS. Table 1 and 2 also show that our GLF achieves the best grounding results on TACoS dataset. Table 1 and 2 also report the grounding results on TACoS dataset. Compared to MARN*, our C3D+Patch model outperforms it by 1.58%, 1.58%, 1.42%, and 0.67% in terms of all metrics. Our I3D+Patch model also outperforms MARN* by a large margin.

Comparison on ActivityNet. Since both MMRG and MARN methods are not implemented on the ActivityNet dataset, we only report the performances on this dataset under official splits as shown in Table 3. Compared to previous best method CBLN, our C3D+Patch model outperforms it by 3.23%, 3.37%, 3.94%, and 3.91% in terms of all metrics. Our I3D+Patch model also outperforms CBLN by a large margin.

Efficiency Comparison. As shown in Table 4, we evaluate the efficiency of our GLF model, by fairly comparing its running time with existing methods on TACoS dataset. It shows that our GLF is more efficient than the detection-based method MARN while on par with the other common methods DRN and CBLN.

4.4 Ablation Study

We perform in-depth ablation studies to evaluate the effectiveness of each component in GLF on Charades-STA dataset. We utilize the C3D+Patch variant as our backbone here.

Components	Variants	R@1, IoU=0.7	R@5, IoU=0.7
Multi-scale Aggregation	w/o gate	41.33	70.16
	w/ gate	**42.75**	**71.49**
Patch Definition	overlap	**42.84**	71.21
	unoverlap	42.75	**71.49**
Scale Size N_p	{9}	40.45	69.27
	{25}	40.72	69.50
	{49}	40.57	69.48
	{81}	40.13	69.06
	{9,25}	42.04	70.79
	{25,49,81}	42.25	70.98
	{9,25,49}	42.75	71.49
	{9,25,49,81}	**42.81**	**71.64**

Table 7: Ablation study on the multi-scale strategy, where scale sizes $9, 25, 49, 81$ denote $3 \times 3, 5 \times 5, 7 \times 7, 9 \times 9$, respectively.

Main ablation. As shown in Table 5, we verify the contribution of each part in our GLF. We first implement the baseline model by directly applying the grounding heads on the interacted multi-modal features of all patches without both self-supervised patch localization and multi-scale strategy modules. The baseline model achieves 33.71% and 62.35% in terms of R@1, IoU=0.7 and R@5, IoU=0.7, respectively. By adding the coarse-level patch localization module $\mathcal{L}_{match}$ to the baseline, the model brings the improvement of 3.57% and 3.51% since it selectively focuses on the important foreground regions. After further adding the fine-level $\mathcal{L}_{ground}$ for filtering out the redundant patches, the model achieves better results. Besides, the multi-scale strategy also brings a significant improvement to the full model.

Ablation on multi-modal encoders. We also conduct the investigation on different variants of multi-modal encoders in Table 6. We find that the full model performs worse if we remove the global feature that helps to better explore the non-local information among the patches. Besides, it also presents the effectiveness of the position encoding in identifying spatial-temporal knowledge. The transformer modules in both video and query encoders and the Bi-GRU module also bring additional performance to the full model.

Ablation on the multi-scale strategy. We further perform ablation study on our proposed multi-scale patch strategy in Table 7. It shows that our gate-based multi-scale aggregation module brings the improvement of 1.42% and 1.33% in terms of R@1, IoU=0.7 and R@5, IoU=0.7, respectively. Besides, the overlapped and unoverlapped patches have little impact on the final grounding perfor-

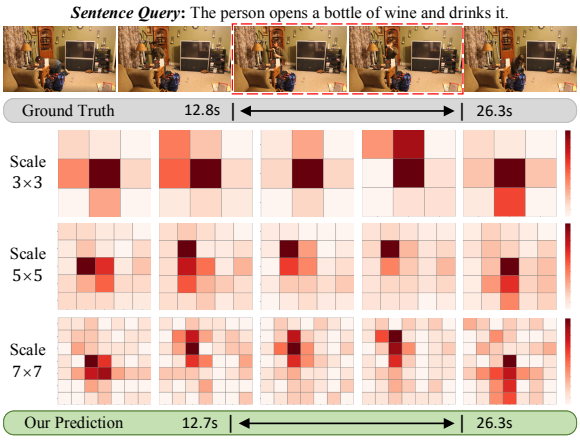

Figure 4: Visualization results on the predicted scores of patches.

mance. Therefore, we choose the unpverlapped one in all our experiments. Moreover, with more various patch scales, the model usually performs better than an individual scale. The variant with four scales $\{9, 25, 49, 81\}$ achieves the best result but only performs marginally better than the three-scale one $\{9, 25, 49\}$ at the expense of a significantly larger cost of GPU memory. Thus, we choose $N_p = \{9, 25, 49\}$ in our all experiments.

4.5 Visualization

We show the visualization on the scored multi-scale patches in Figure 4, where the patches with highest scores contain the most query-related visual appearances. From this figure, we can find that our scoring function can well learn the patch-query similarities among different grains. By jointly combing the contexts from different attended patches, our GLF model performs accurate grounding result.

5 Conclusion

In this paper, we make the first attempt to propose a novel detection-free framework for temporal sentence grounding (TSG), called Grounding with Learnable Foreground (GLF). In particular, we split each video frame into patches with multiple scales, and reformulate the foreground detection problem as a patch localization task. In detail, we interact each patch with the query semantic to learn their matching scores supervised by our newly designed self-supervised losses. Further, we develop a two-level coarse-to-fine paradigm to gradually localize the most query-relevant (foreground) patch in each clip. Moreover, considering the sizes of the foreground regions may vary in different videos, we extend the GLF model with multi-scale patch design to capture more fine-grained and complete foreground information for better grounding. Experimental results on three challenging datasets (Charades-STA, TACoS, ActivityNet) validate the effectiveness of our proposed model.

6 Acknowledgments

This work is supported by the National Key R&D Program of China under contract No. 2021YFF0901502.

References

Lisa Anne Hendricks, Oliver Wang, Eli Shechtman, Josef Sivic, Trevor Darrell, and Bryan Russell. 2017. Localizing moments in video with natural language. In *ICCV*.

Joao Carreira and Andrew Zisserman. 2017. Quo vadis, action recognition? a new model and the kinetics dataset. In *CVPR*.

Jingyuan Chen, Xinpeng Chen, Lin Ma, Zequn Jie, and Tat-Seng Chua. 2018. Temporally grounding natural sentence in video. In *EMNLP*.

Long Chen, Chujie Lu, Siliang Tang, Jun Xiao, Dong Zhang, Chilie Tan, and Xiaolin Li. 2020. Rethinking the bottom-up framework for query-based video localization. In *AAAI*.

Wen-Sheng Chu, Yale Song, and Alejandro Jaimes. 2015. Video co-summarization: Video summarization by visual co-occurrence. In *CVPR*.

Junyoung Chung, Caglar Gulcehre, KyungHyun Cho, and Yoshua Bengio. 2014. Empirical evaluation of gated recurrent neural networks on sequence modeling. In *NIPS*.

Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, et al. 2021. An image is worth 16x16 words: Transformers for image recognition at scale. In *ICLR*.

Jiyang Gao, Chen Sun, Zhenheng Yang, and Ram Nevatia. 2017. Tall: Temporal activity localization via language query. In *ICCV*.

Ranjay Krishna, Kenji Hata, Frederic Ren, Li Fei-Fei, and Juan Carlos Niebles. 2017. Dense-captioning events in videos. In *ICCV*.

Daizong Liu, Xiang Fang, Wei Hu, and Pan Zhou. 2022a. Exploring optical-flow-guided motion and detection-based appearance for temporal sentence grounding. *arXiv preprint arXiv:2203.02966*.

Daizong Liu and Wei Hu. 2022. Skimming, locating, then perusing: A human-like framework for natural language video localization. In *ACM MM*.

Daizong Liu, Xiaoye Qu, Xing Di, Yu Cheng, Zichuan Xu Xu, and Pan Zhou. 2022b. Memory-guided semantic learning network for temporal sentence grounding. In *AAAI*.

Daizong Liu, Xiaoye Qu, Jianfeng Dong, and Pan Zhou. 2020a. Reasoning step-by-step: Temporal sentence localization in videos via deep rectification-modulation network. In *COLING*.

Daizong Liu, Xiaoye Qu, Jianfeng Dong, and Pan Zhou. 2021a. Adaptive proposal generation network for temporal sentence localization in videos. In *EMNLP*, pages 9292–9301.

Daizong Liu, Xiaoye Qu, Jianfeng Dong, Pan Zhou, Yu Cheng, Wei Wei, Zichuan Xu, and Yulai Xie. 2021b. Context-aware biaffine localizing network for temporal sentence grounding. In *CVPR*.

Daizong Liu, Xiaoye Qu, and Wei Hu. 2022c. Reducing the vision and language bias for temporal sentence grounding. In *ACM MM*.

Daizong Liu, Xiaoye Qu, Xiao-Yang Liu, Jianfeng Dong, Pan Zhou, and Zichuan Xu. 2020b. Jointly cross-and self-modal graph attention network for query-based moment localization. In *ACM MM*.

Daizong Liu, Xiaoye Qu, Yinzhen Wang, Xing Di, Kai Zou, Yu Cheng, Zichuan Xu, and Pan Zhou. 2022d. Unsupervised temporal video grounding with deep semantic clustering. In *AAAI*.

Daizong Liu, Xiaoye Qu, and Pan Zhou. 2021c. Progressively guide to attend: An iterative alignment framework for temporal sentence grounding. In *EMNLP*.

Daizong Liu, Xiaoye Qu, Pan Zhou, and Yang Liu. 2022e. Exploring motion and appearance information for temporal sentence grounding. In *AAAI*.

Shentong Mo, Daizong Liu, and Wei Hu. 2022. Multi-scale self-contrastive learning with hard negative mining for weakly-supervised query-based video grounding. *arXiv preprint arXiv:2203.03838*.

Jonghwan Mun, Minsu Cho, and Bohyung Han. 2020. Local-global video-text interactions for temporal grounding. In *CVPR*.

Guoshun Nan, Rui Qiao, Yao Xiao, Jun Liu, Sicong Leng, Hao Zhang, and Wei Lu. 2021. Interventional video grounding with dual contrastive learning. In *CVPR*.

Jeffrey Pennington, Richard Socher, and Christopher D Manning. 2014. Glove: Global vectors for word representation. In *EMNLP*.

Michaela Regneri, Marcus Rohrbach, Dominikus Wetzel, Stefan Thater, Bernt Schiele, and Manfred Pinkal. 2013. Grounding action descriptions in videos. *ACL*.

Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun. 2015. Faster r-cnn: Towards real-time object detection with region proposal networks. In *NIPS*.

Zheng Shou, Dongang Wang, and Shih-Fu Chang. 2016. Temporal action localization in untrimmed videos via multi-stage cnns. In *CVPR*.

Yale Song, Jordi Vallmitjana, Amanda Stent, and Alejandro Jaimes. 2015. Tvsum: Summarizing web videos using titles. In *CVPR*.

Du Tran, Lubomir Bourdev, Rob Fergus, Lorenzo Torresani, and Manohar Paluri. 2015. Learning spatiotemporal features with 3d convolutional networks. In *ICCV*.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *NIPS*.

Jingwen Wang, Lin Ma, and Wenhao Jiang. 2020. Temporally grounding language queries in videos by contextual boundary-aware prediction. In *AAAI*.

Zeyu Xiong, Daizong Liu, and Pan Zhou. 2022. Gaussian kernel-based cross modal network for spatio-temporal video grounding. *arXiv preprint arXiv:2207.00744*.

Huijuan Xu, Kun He, Bryan A Plummer, Leonid Sigal, Stan Sclaroff, and Kate Saenko. 2019. Multilevel language and vision integration for text-to-clip retrieval. In *AAAI*.

Yitian Yuan, Lin Ma, Jingwen Wang, Wei Liu, and Wenwu Zhu. 2019. Semantic conditioned dynamic modulation for temporal sentence grounding in videos. In *NIPS*.

Runhao Zeng, Haoming Xu, Wenbing Huang, Peihao Chen, Mingkui Tan, and Chuang Gan. 2020. Dense regression network for video grounding. In *CVPR*.

Yawen Zeng, Da Cao, Xiaochi Wei, Meng Liu, Zhou Zhao, and Zheng Qin. 2021. Multi-modal relational graph for cross-modal video moment retrieval. In *CVPR*.

Hao Zhang, Aixin Sun, Wei Jing, and Joey Tianyi Zhou. 2020a. Span-based localizing network for natural language video localization. In *ACL*.

Songyang Zhang, Houwen Peng, Jianlong Fu, and Jiebo Luo. 2020b. Learning 2d temporal adjacent networks for moment localization with natural language. In *AAAI*.

Zhu Zhang, Zhijie Lin, Zhou Zhao, and Zhenxin Xiao. 2019. Cross-modal interaction networks for query-based moment retrieval in videos. In *SIGIR*.

Yue Zhao, Yuanjun Xiong, Limin Wang, Zhirong Wu, Xiaoou Tang, and Dahua Lin. 2017. Temporal action detection with structured segment networks. In *ICCV*.

Are Visual-Linguistic Models Commonsense Knowledge Bases?

Hsiu-Yu Yang
University of Stuttgart
hsiu-yu.yang@ims.uni-stuttgart.de

Carina Silberer
University of Stuttgart
carina.silberer@ims.uni-stuttgart.de

Abstract

Despite the recent success of pretrained *language* models as on-the-fly knowledge sources for various downstream tasks, they have been shown to inadequately represent trivial common facts that *vision* typically captures. This limits their application to natural language understanding tasks that require commonsense knowledge. We seek to determine the capability of pretrained *visual–linguistic* models as knowledge sources on demand. To this end, we systematically compare *language-only* and *visual–linguistic* models in a zero-shot commonsense question answering inference task. We find that visual–linguistic models are highly promising regarding their benefit for text-only tasks on certain types of commonsense knowledge associated with the visual world. Surprisingly, this knowledge can be activated even when no visual input is given during inference, suggesting effective multimodal fusion during pretraining. However, we also reveal that there is still a huge scope for improvements towards better cross-modal reasoning abilities and pretraining strategies for event understanding.[1]

1 Introduction

Commonsense knowledge is essential in human life for task solving and communication. Being aggregated knowledge acquired from past experiences and communications, it is usually left implicit in human communication, and used subconsciously for reasoning and drawing inferences. This puts a challenge on natural language understanding systems, and a large body of work has put forward approaches to provide commonsense knowledge to models for downstream tasks (Yang et al., 2019; Chen et al., 2018; Mihaylov and Frank, 2018). While knowledge bases have been a popular way to provide relevant knowledge for a task at hand, more recently, pretrained language models (PTLMs) have become a popular mechanism to extract knowledge in free-form text on demand. Tasks range from, e.g., persona-grounded dialog (Majumder et al., 2020), narrative story generation (Ammanabrolu et al., 2020), to metaphor generation (Stowe et al., 2021). Shwartz and Choi (2020) and Bisk et al. (2020), however, suggest that text corpora alone may be insufficient for knowledge acquisition due to reporting bias found in them (Gordon and Durme, 2013). This has led to analyzing the knowledge that PTLMs possess through dedicated probing studies (Petroni et al., 2019; Singh et al., 2021; Zhou et al., 2020b)

Existing works on probing PTLMs have used a loose categorization of commonsense, which limits a comprehensive understanding of the types of commonsense they possess and lack, respectively. At the same time, the literature proposes vision as a promising knowledge source (Izadinia et al., 2015; Bagherinezhad et al., 2016; Sadeghi et al., 2015). It seems therefore straightforward to leverage visual–linguistic (VL) models for knowledge extraction on demand—these representation models are extensions of PTLMs to the visual–linguistic domain by pretraining LMs and image recognition models jointly on multimodal data (Tan and Bansal, 2019; Chen et al., 2020; Lu et al., 2019). Yet, the question on their capability to capture commonsense knowledge that can be activated through language only (Yun et al., 2021) is yet to be explored systematically.

In this work, we address this research gap by conducting a controlled comparison of text-only and VL models. Specifically, we extend a synthetic commonsense question answering (QA) dataset based on Ma et al. (2021)'s work, which structures knowledge relations into abstract types (called *dimensions* henceforth), and transform it to a QA inference task. We use the task to compare the models by applying them in a zero-shot manner,

[1] Our datasets, CWWV$_{Img}$ and CWWV$_{Clip}$, are provided at https://github.com/Mallory24/CS_Probing

Proceedings of the 29th International Conference on Computational Linguistics, pages 5542–5559
October 12–17, 2022.

and in their natural setting—masked language modeling.

The overarching question of our study is:

Do VL models learn to encode commonsense knowledge through multimodal pretraining, that can be activated during inference from textual input only?

In particular, we seek to empirically answer:

(**Q1**) Which dimensions of commonsense do VL models possess compared against text-only PTLMs?

(**Q2**) During pretraining, does explicit visual information (i.e., images) benefit commonsense knowledge encoding?

(**Q3**) During inference, is explicit visual observation (i.e., images) necessary for recalling commonsense knowledge?

(**Q4**) Do commonsense acquisition and retrieval depend on the architecture of VL models?

We address the questions by performing a range of experiments using various pretrained models and ablated variants. We find that existing VL models do complement PTLMs on certain commonsense dimensions, which are related to the visual world (*part-whole*, *spatial*, i.a.), and that they can be activated through language input only, making them promising for their use in natural language tasks that do not require explicit visual context. We also identify a range of limitations opening up several avenues for future work, including enhanced pretraining and modality integration strategies, and improved multimodal prompting (Shin et al., 2020; Zhong et al., 2021; Liu et al., 2021).

2 Related Work

Commonsense Knowledge Mining from Vision Although recent interest in commonsense knowledge mining remains text-based (Jastrzębski et al., 2018; Zhou et al., 2020b; Liang and McGuinness, 2021; Bosselut et al., 2019), several studies have explored the visual world: Chen et al. (2013) extract commonsense relationships from the web to improve visual understanding, while Zellers et al. (2018) exploit commonsense priors from visual resources (Krishna et al., 2017) for scene graph generation. Several works learnt specific types of commonsense, including object affordances (Goyal et al., 2017) and temporal causal knowledge (Zhang et al., 2020a). Only few works used VL models for purely text-based tasks (Cui et al., 2020; Tang et al., 2021) in a pipeline approach to extract commonsense from them.

The works above focus on explicit visual commonsense extraction. We, in contrast, seek to study the extent and types of commonsense knowledge that pretrained VL models implicitly capture and that complement pretrained text-based models.

Machine Commonsense Evaluation Commonsense knowledge evaluation is usually conducted with dedicated benchmarks specific for selected knowledge types. Existing formulations range from multiple choice question answering (Zellers et al., 2019; Zhou et al., 2019; Bisk et al., 2020; Richardson and Sabharwal, 2020) and machine reading comprehension (Huang et al., 2019) to knowledge base completion tasks (Petroni et al., 2019; Davison et al., 2019), which makes a systematic and comprehensive commonsense knowledge evaluation even more challenging (Santos et al., 2020). Recent works assess model consistency in commonsense reasoning by introducing linguistic perturbations, complementary counterparts, and logically-equivalent rephrased sentences (Zhou et al., 2020b; Singh et al., 2021; Zhou et al., 2020a). Akin to Ilievski et al. (2021), our goal is to present a comprehensive comparison of the commonsense knowledge resided in pretrained models. While previous research dominantly employs pretrained language-only models (PTLMs), we are not aware of any work like ours—a structured analysis of the types of commonsense knowledge implicitly encoded in pretrained VL models.

3 The QA Dataset CWWV$_{Img}$

To compare VL models against purely textual models with respect to the commonsense knowledge they capture, we extend Ma et al. (2021)'s procedure for creating a synthetic dataset of prompt–answer candidate instances (CWWV) to that of a *multimodal* commonsense dataset (CWWV$_{Img}$).

It provides a set of QA instances for various knowledge relations, structured into 10 dimensions of commonsense knowledge (e.g., *spatial*). Questions are in the form of filled prompts (Le Scao and Rush, 2021; Liu et al., 2021): an instance in CWWV$_{Img}$ has three natural language statements, each associated with a set of images. Each statement is a pair of a *prompt* (e.g., *Shade is not*) and one of three candidate answers (e.g., *sunny*). Table 1 shows an example for each dimension (associated images are omitted for space reasons).

CS dimension	Starting prompt	Answer candidates	# Instances
part-whole	Furry animals have	A_1: effect of chilling innovation. **A_2: millions of hair.** A_3: hole in.	1,165
taxonomic	Recruit is a way to	A_1: rate. **A_2: enlist.** A_3: slope.	1,323
distinctness	Shade is not	A_1: flat. A_2: postal worker. **A_3: sunny.**	828
similarity	Throw up is a synonym of	A_1: rutinic acid. A_2: random. **A_3: vomit.**	644
quality	A wet floor is	**A_1: slippery.** A_2: light brown. A_3: abbreviated to unido.	1,840
utility	A fork is used for	A_1: speed of transit. A_2: confuse voters. **A_3: picking up food.**	2,090
creation	Music is created by	A_1: olive oil mill. A_2: mapping process. **A_3: instruments.**	100
temporal	Going for a haircut requires	**A_1: finding barber.** A_2: hard examinations. A_3: write persuasively.	1,889
spatial	You are likely to find a document folder in	**A_1: file drawer.** A_2: madagascar jungle. A_3: minerals.	1,599
desire	You would thank someone because you want to	A_1: accomplish mutual goal. **A_2: feel good.** A_3: cool off.	1,781

Table 1: CWWV$_{Img}$: examples and their number per dimension (13,259 in total). **Correct answers** are in bold. Topic words with retrieved images are underlined (images not shown for space reasons).

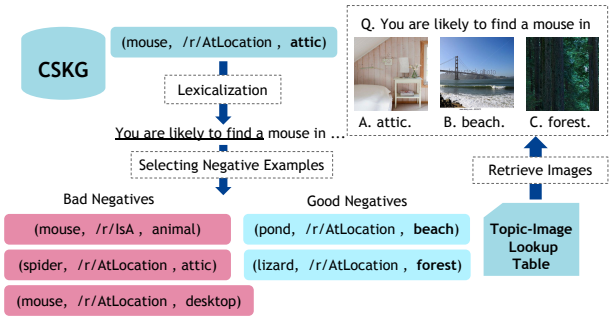

Figure 1: CWWV$_{Img}$ construction pipeline.

In §3.1, we first describe Ma et al.'s (2021) pipeline, shown in Figure 1, to create the purely textual CWWV.[2] Then, to build our QA dataset CWWV$_{Img}$, we retrieve images for CWWV's instances through a topic-lookup table (Zhang et al., 2020b), as we explain in §3.2.

3.1 Generation of CWWV

The Commonsense QA data CWWV is automatically generated from a consolidated commonsense knowledge graph (Ilievski et al., 2020, CSKG). CSKG is an aggregation of 7 knowledge bases that represents knowledge statements as structured triples (h, r, t) of their start node h, relation label r, and end node t, as in (mouse, /r/AtLocation, attic). All knowledge relations are categorized into one of 13 abstract dimensions, e.g., *spatial*, *similarity*, *temporal*. To create CWWV from CSKG, we only consider the knowledge bases ConceptNet (Speer et al., 2017), WordNet (Miller, 1995), Visual Genome (Krishna et al., 2017) and Wikidata (Vrandecic and Krötzsch, 2014), and 10 dimensions (30 knowledge relations, see App. A, Tab. 5 for the mapping).

Ground-Truth QA Generation Given a knowledge triple (h, r, t) in CSKG, a ground-truth prompt–answer pair is generated by treating the start node h and the end node t as prompt and correct answer, respectively, and applying pre-defined sentence templates to lexicalize them into a sentence. For example, the triple (mouse, /r/AtLocation, attic) in Figure 1 is transformed into a sentence "*You are likely to find a* mouse *in an* attic". To prevent models from applying shortcuts, triples with overlapping content words between the start and end nodes are discarded; e.g., (bread slicer, /r/UsedFor, slicing bread) won't be included. Uncommon concepts or named entities are also filtered out.[3]

Selecting Negative Candidates For each generated QA instance, Ma et al. (2021) select two negative answer candidates according to two principles: (i) the negative candidate is related to the prompt, and thus remains informative for decision, and (ii) it can be clearly discriminated from the correct one, and thus maintains fairness for the model.

To satisfy informativeness, negative candidates are randomly chosen from a pool of relation triples (h', r', t') with $r' = r$, i.e., the relation is the same as the original one. In this sense, a bad negative would be, e.g., (mouse, /r/IsA, animal) (Fig. 1). To ensure fairness, the end node must not be the ground-truth one, $t' \neq t$, and h' must not share any overlapping tokens with h; e.g., (spider, /r/AtLocation, attic) and (mouse, /r/AtLocation, desktop) (Fig. 1) are discarded for violating these two heuristics.

We create CWWV by randomly sampling 2,500 QA instances for each dimension (*creation* only has 141 samples), totalling 22,641 instances.

[2] We use Ma et al.'s 2021 script provided at https://github.com/Mayer123/HyKAS-CSKG.

[3] Uncommon concepts are determined by low word frequency in a corpus https://pypi.org/project/wordfreq/ (accessed 9 September 2020) and named entities are identified through the capital letter.

3.2 Generation of CWWV$_{Img}$

Retrieving Images from Conceptual Captions The VL models we use in our experiments are pretrained on the training set of Conceptual Captions (Sharma et al., 2018, CC), a widely adopted dataset of weakly-associated image–text pairs collected from the web which may be regarded as a model's visual experience. We hence use the training set of CC as image retrieval pool to augment CWWV with images. As an efficient way to provide a visual environment, as realistic as possible, for a purely linguistic task, we perform an efficient retrieval method inspired by Zhang et al. (2020b): We first transform CC's image–caption pairs into a topic–image lookup table T. Given a prompt–answer candidate pair (statement) of CWWV, we then select several topic words based on TF–IDF weights, $QA_{topic} = \{t_1, t_2, ..., t_q\}$, and use them to query T for associated images; for example, "A wet floor is slippery" has a set of topic words $\{\text{wet}, \text{floor}, \text{slippery}\}$ (Further details can be found in Appendix. B.1). Recall that each QA instance has three statements. As shown in Figure 1, by querying T for the three candidate answers "attic", "beach", "forest", we retrieve their corresponding images. If no image can be retrieved for a statement of an instance, the instance is discarded altogether. The resulting image-grounded commonsense QA dataset, CWWV$_{Img}$, has $13,259$ QA instances in total with the image grounding rate of 58.6%.[1]

Quality of Retrieved Images We assessed the effectiveness of our simple retrieval approach through a human annotation study on Amazon Mechanical Turk (AMT), in which we asked workers to judge the association of image–word pairs. We sampled $1,000$ pairs from CWWV$_{Img}$ uniformly across the 10 commonsense dimensions, and asked AMT workers to judge each pair as either "associated" or "not associated". Details on the annotation methodology and data analyses are given in Appendix B.2. According to majority vote (2 out of 3 judges per pair), 64.2% of the pairs are associated, among which *part-whole* and *spatial* have more than average associated pairs (74.3% and 70.2%, respectively). The inter-annotator agreement under Fleiss' Kappa coefficient (Fleiss, 1971) is between $0.21 - 0.44$ across dimensions, which is only a fair to moderate agreement, indicating the high subjectivity of this task. We also found, unsurprisingly, that concrete words and nouns tend to get higher scores with their paired images.

4 Experiments: QA Task and Inference

Our goal is to assess the benefit of pretrained VL models for purely linguistic tasks underlying commonsense (CS) knowledge. Specifically, we seek to answer the questions **(Q1)** – **(Q4)** that we put forward in §1. To this end, we use our derived image-grounded dataset, CWWV$_{Img}$, and evaluate pretrained VL models against language-only PTLMs in a prompt-based QA task setting (§4.1). We stress that in order to solve CWWV$_{Img}$, only natural language understanding and commonsense knowledge is required, but no explicit visual input (images).

We perform our experiments in a *zero-shot* setting, i.e., without fine-tuning the models on the task, since our goal is to study the ability of *task-agnostic* pretrained models to capture commonsense knowledge (Tamborrino et al., 2020; Ma et al., 2021).

4.1 Task: Prompt-based Zero-Shot QA

Given an instance $T \in \mathcal{T}$ of CWWV$_{Img}$, comprising three natural language statements (and associated images I_i), $T_i = (Q||A_i)$, $i = 1, \ldots, 3$, where Q is the prompt, and A_i a candidate answer. Let $t_j \in T_i$, $j = 1, \ldots, |T_i|$ denote the sequence of tokens in T_i. Then, the task is to determine which of the three statements is a true assertion (given visual context I_i or not). To mitigate the bias that some template prompts can favor one model over the other in terms of knowledge retrieval (Jiang et al., 2020), we use a two-stage inference procedure, namely a generative and a discriminative setting. During the generative stage, we test representative PTLMs and VL models, respectively, under their natural setting—masked language modeling (MLM) (as detailed in §4.3). Later in the discriminative stage, the ranking of the candidate answers A_i is determined by how well the model can reconstruct the masked tokens of the respective statement T_i by comparing the MLM loss.

4.2 Tested Models

Pretrained Language Models (PTLMs) We use BERT (Devlin et al., 2019) for our comparison, since this model serves as the linguistic backbone of the VL models that we study. We also compare against RoBERTa (Liu et al., 2019), which was pretrained on ten times more data than BERT (160GB vs. 16GB of text, resp.). We use the *BASE* models of the HuggingFace library (Wolf et al., 2019).

VL Models We select the single-stream model, UNITER (Chen et al., 2020), and the dual-stream variant, VILBERT (Lu et al., 2019), as the respective representative models of the two common VL architectures. UNITER is built to have unconstrained inter-modal and intra-modal attentions across all attention blocks, whereas VILBERT has certain attention blocks specifically constrained to perform inter-modal attention only.

We use the pretrained models of VOLTA (Bugliarello et al., 2021), which provides both architectures in an unified framework and under a controlled setup to allow a fair comparison.[4] The models were initialized with the pretrained BERT$_{BASE}$, and further trained on the training set of Conceptual Captions (CC) under three objectives: masked language modeling (MLM), masked object classification and image–text matching.

The visual input is preprocessed into a sequence form of visual tokens $v \in V$ consisting of an `[IMG]` feature and 36 object region features.[5] Each visual token is accompanied with its corresponding spatial encodings in a 5-d vector.[6]

4.3 Inference Variants

Common to all models is that they are queried with each of the three statements $T_i = (Q||A_i)$ of an instance $T \in \text{CWWV}_{Img}$, and the T_i that receives the lowest mean MLM loss $\text{S}(T_i)$ will be returned as answer. We explain the respective precise formulations of $\text{S}(T_i)$ that the textual and VL models apply in the following.

Inference in Language Models To compute $S(T_i)$ in the case of the language-only LMs, we sequentially mask out each token q_j in T_i's prompt Q of length L_Q, and compute its log-likelihood, conditioning on the remaining tokens $T_{i\setminus j}$[7]:

$$S(T_i) = -\frac{1}{L_Q}\sum_{j=1}^{L_Q} \log P(q_j|T_{i\setminus j}), q_j \in Q$$

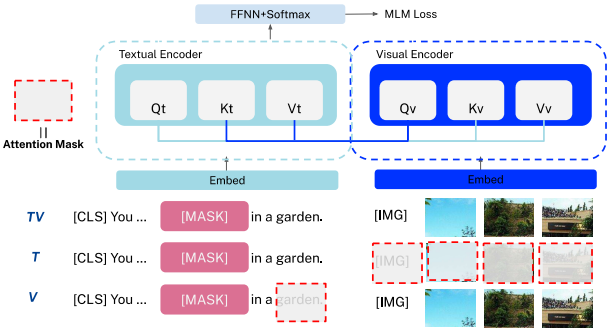

Figure 2: Different inference modes in the VL models (dual-stream shown here): **Retrieved** (TV), **Text-Only** (T), **Vision-Only** (V) **Modes**.

This way, the statement score is not affected by the internal bias of the answer's tokens, e.g., word frequency (Tamborrino et al., 2020).

Inference in VL Models Recall that CWWV_{Img} provides multiple images to serve as visual context for each statement (§4.1). Apart from examining the behavior of the VL models when they get exactly the same input as the textual LMs (**Text-Only**), we further analyse them on modes differing in their input (Fig. 2): (i) their natural setting with multimodal input (**Retrieved**), (ii) **Vision-Only**, and (iii) visual noise with random images (**Dummy**).

Text-Only Mode To examine **Q3** (see §1), if explicit, situated visual input is required for using VL models for purely language-based tasks underlying CS knowledge, we apply inter-modal attention masks on the models' visual input. This way only the representations of the linguistic encoder affect the model decision. Hence, we can test if the VL models encode aggregated visual exposures which they can activate through language only input. Models under this mode are suffixed $*_T$, e.g., **UNITER**$_T$.

Retrieved Mode Given a statement in sequence form, $T_i = (t_1, t_2, \ldots, t_{|T_i|})$, with its associated images $I_i = \{e_1, \ldots, e_q\}$ based on the q topic tokens t_l identified in T_i (see §3.2). Instead of testing the models under a *situated* context like the standard VQA task, our goal is to examine their effectiveness in activating world knowledge from explicit visual input during inference. We thus deviate from the conventional setup and do not provide a single, but multiple images that represent general visual concepts to the models. We apply a threshold τ on the object detection score to choose

[4] We choose UNITER and VILBERT since they perform the best among each respective variant on a wide range of VL benchmarks (Bugliarello et al., 2021).

[5] Region features are extracted by a Faster R-CNN with a ResNet-101 backbone (Anderson et al., 2018) trained on Visual Genome with 36 regions of interest; `[IMG]` is the mean-pooling of the 36 features (Bugliarello et al., 2021).

[6] $(x_1, y_1, x_2, y_2, w * h)$: normalized left/top/right/bottom coordinates and the area.

[7] For convenience, we use the notation $X_{i\setminus j}$ to refer to a sequence of elements $(x_1, \ldots, x_{j-1}, x_{j+1}, \ldots, x_{|X|})$.

row		part-whole	taxonomic	distinctness	similarity	quality	utility	creation	temporal	spatial	desire	All
1	RoBERTa	68.5	61.8	**80.2**	**67.4**	69.7	**74.2**	72.0	**60.9**	54.8	**65.9**	**67.5**
2	BERT	62.8	**71.2**	80.1	54.8	68.1	72.4	**74.0**	53.7	52.4	60.4	65.0
3	BERT_{CC}	68.4	62.0	66.6	51.1	66.0	65.4	62.0	53.6	**63.7**	58.3	61.9
4	UNITER_{T}	**70.9**	59.8	71.3	51.2	**69.9**	71.5	71.0	52.7	61.5	62.5	64.0
5	VILBERT_{T}	63.9	60.3	64.9	46.7	66.1	71.2	58.0	52.2	61.0	62.8	60.7

Table 2: Accuracy on CWWV_{Img} under Text-Only inference mode (The full table can be seen in App. C, Tab.7).

a *set* of salient regions from each individual image[8], resulting in a set of extracted visual tokens $V_{e_i} \cup$ `{[IMG]}` which we feed into the model. Now, during MLM inference, we need to ensure fairness to the LMs by avoiding information leakage from the visual modality. Hence, when masking out token q_j in T_i's prompt Q, we also mask out the subset of visual tokens V_{q_j} associated to $q_j \in Q$:

$$S(T_i) = -\frac{1}{L_Q}\sum_{j=1}^{L_Q} \log P(q_j|(T_{i\setminus q_j} \| V_{e_i \setminus V_{q_j}})$$

To reconstruct q_j, it is conditioned on the concatenation of the unmasked textual tokens $T_{i\setminus q_j}$ and the remaining visual tokens $V_{e_i\setminus V_{q_j}}$ that are not associated to q_j. Models evaluated under this mode are denoted as $\textbf{UNITER}_{TV}$ and $\textbf{VILBERT}_{TV}$.

Vision-Only Mode We ablate the textual modality via inter-modal attention masking, the VL models can hence rely only on visual observation to reconstruct masked tokens in the prompt Q of a candidate statement. For example, to reconstruct the masked "vegetables" token in prompt "You are likely to find `[MASK]` in" (Fig. 2), answer candidate A_3 ("garden") is not given as observable input, but only its respective associated image I_3. Note that also the visual tokens associated to "vegetables" are masked:

$$S(T_i) = -\frac{1}{L_Q}\sum_{j=1}^{L_Q} \log P(q_j|(Q_{i\setminus q_j} \| V_{e_i \setminus V_{q_j}})$$

This posits a challenging task on testing the models' cross-modal integration ability of non-aligned concepts without the help of the corresponding textual part as a bridge (e.g., relate the appearance of "garden" to the concept of "vegetables"). The variants are suffixed $*_V$, e.g., $\textbf{VILBERT}_{V}$.

Dummy Mode We sample 3 images from the CC image pool that are not in CWWV_{Img} to serve as random visual input for each QA instance. During inference, we allow the VL models to fully observe this visual input $\tilde{V}_{e_i}$ since it is not or barely related to any textual token $q_j \in Q$:

$$S(T_i) = -\frac{1}{L_Q}\sum_{j=1}^{L_Q} \log P(q_j|X_{i\setminus q_j} \| \tilde{V}_{e_i})$$

, where $X = T$ in the Retrieved Mode, and $X = Q$ for Vision-Only. This is an adversarial test of the cross-modal reasoning ability of the VL models and we expect a lower performance. In addition, it enables us to assess in how far the VL models can deal with noise. These variants are suffixed $*_{(T)\tilde{V}}$, e.g., $\textbf{UNITER}_{T\tilde{V}}$.

5 Results

To address the overarching question we put forward in §1, namely if VL models can serve as commonsense knowledge base, and to what extent they complement pure linguistic model, we first examine the encoded knowledge of the VL models on individual commonsense dimensions, i.e., **(Q1)** in §1. We report the models' effectiveness by measuring their mean accuracy in selecting the correct answer out of the three statement candidates of each QA instance in CWWV_{Img}. We declare outperformance if $p < 0.05$ according to the paired student's t-test (Fisher, 1949) for statistical significant differences between any two accuracy scores.[9]

Table 2 shows the effectiveness of all models per commonsense dimension and overall when they are given the exact same input, i.e., natural language statements. Comparing the VL models, UNITER_{T} and VILBERT_{T}, against their linguistic backbone BERT, we see that both are more effective on the **part-whole**, **spatial**, and **desire** dimensions.[10]

On **spatial**, both UNITER_{T} and VILBERT_{T} even outperform RoBERTa, which was pretrained on an order of magnitude more data than BERT

[8] Starting at $\tau \geq 0.7$, we decrease τ by 0.1 steps until at least one region is found.

[9] We used Anderson-Darling's (Anderson and Darling, 1954) method to test for normal distribution.

[10] We also evaluate UNITER initialized with BERT weights, without further pretraining on CC, called UNITER_BERT_{T}. It yields similar results as BERT (see App. C, Tab. 7+8) and indicates pretraining on visual data may lead to a catastrophic forgetting on some CS dimensions that require linguistics.

row		part-whole		spatial		taxonomic		distinctness	
		CWWV_{Img}	CWWV_{Clip}	CWWV_{Img}	CWWV_{Clip}	CWWV_{Img}	CWWV_{Clip}	CWWV_{Img}	CWWV_{Clip}
1	UNITER_T	**70.9**	**76.6**	**61.5**	**59.4**	59.8	58.0	**71.3**	**72.5**
2	ViLBERT_T	63.9	70.7	61.0	58.8	**60.3**	**59.4**	64.9	62.6
3	UNITER_{TV}	63.0	68.1	57.4	54.1	54.0	55.9	65.9	68.1
4	ViLBERT_{TV}	55.0	58.0	52.9	**59.4**	49.9	58.0	55.9	62.6

Table 3: Model accuracy on CWWV_{Img} and CWWV_{Clip}. **Bold** represents the highest score per commonsense dimension.

and the VL models (§4.2). The benefit of visual–linguistic pretraining for spatial (and concrete part–whole) relations is in accordance with what we would expect, and in line with existing work on the spatial dimension (Yatskar et al., 2016; Cui et al., 2020). UNITER_T is also on par with RoBERTA on **part–whole** and **quality**. On the other hand, the VL models failed to retain knowledge associated with other dimensions during VL pretraining, performing significantly worse ($p < .05$) than BERT in particular on **taxonomic** and **distinctness**. Regarding **taxonomic** (and **similarity**), we observe that the VL models tend to struggle with visually non-depictable concepts (e.g., *speculate*, *remember*). And contemplating **distinctness** (e.g., *flood* vs. *drought*) may be challenging for the VL models due to the unnatural, simultaneous co-occurence of opposite concepts in a single image.

Regarding **temporal**, where events are expressed as verbal phrases (e.g., *checking vital signs*, *wait on tables*), the ability of the VL models to leverage the potential benefit of visual information is limited by their pretraining on isolated images (and regions) instead of, e.g., videos, and with objectives (§4.2) which essentially stipulate the models to learn modality alignments, and in particular region-level recognition (Chen et al., 2020), which limits their ability to capture inter-object interactions, relevant for verb-centric and event understanding (see also Hendricks and Nematzadeh 2021).

6 Analysis

To answer our questions **(Q2)**-**(Q4)** we put forward in §1, we first examine **(Q2)**, the benefit of visual information during *pretraining*. We then investigate **(Q3)**, the role of explicit visual input during *inference*. Lastly, we look into **(Q4)**, whether the process of knowledge acquisition and retrieval act consistently across VL models.

6.1 Role of Visual Input during Pretraining

The VL models show the ability to learn certain types of commonsense knowledge that complements that in text-based models by leveraging information during visual–linguistic pretraining (cf. Cui et al. 2020), which they can activate even when no visual information is given during inference. While we found that this does not attribute to the mere size of the training data, it is not clear if the models benefit from the explicit visual information (i.e., visual features), or from the weakly associated verbalizations (i.e., captions) of its visual data. The latter would just be an effect of the domain shift to the visual world, providing information that is typically not found in text corpora.

To examine the contribution of visual features, we further pretrained BERT on the textual part of the VL models' training data (CC captions)[11], referred to as $\textbf{BERT}_{CC}$. The model's effectiveness drops on all dimensions except **part–whole** and **spatial** (rows 2+3, Tab. 2), so the verbalizations are indeed beneficial for these dimensions, but detrimental for the others. Notably, UNITER_T, being pretrained additionally on images, overall obtains a higher accuracy than BERT_{CC}, and outperforms it on **part–whole**. For **spatial**, though, the captions seem to serve as sufficient surrogate of visual spatial relationships, while explicit visual information is of less benefit for UNITER_T. We see these effects only for single-stream UNITER_T, while double-stream ViLBERT_T falls short against BERT_{CC}. We will return to the aspect of the architecture differences in the following section. With regard to **(Q2)**, our results indicate that *pretraining* on explicit visual input is indeed crucial for encoding certain commonsense dimensions.

6.2 Role of Visual Input during Inference

§5 showed promising results regarding the benefit of VL models for *text-based* tasks underlying certain types of CS knowledge. The fact that the VL models were designed to receive multimodal input raises the question if their inference ability benefits from being fed both, textual and visual input. We analyse the VL models when we feed in the

[11] Code from Frank et al. (2021), with MLM and 5 epochs.

dim.: spatial

You are likely to find vegetables in:
A. workplace.
B. stationary shop.
C. *garden.*

dim.: part-whole

A boat has:

A. reached legal age.
B. *sails*
C. different rules.

dim.: quality

A hill can be:

A. *steep.*
B. about to change.
C. important for normal living.

Table 4: Positive examples of VILBERT$_V$'s inference with visual answer tokens only. Images correspond to the visual input for the correct textual answer tokens (i.e., "garden", "sails", "steep"). The bounding boxes mark the highly attended (> 0.3) visual tokens of VILBERT$_V$ on the last inter-modal layer.

images along the textual prompts of CWWV$_{Img}$. We also compare against a subset of CWWV$_{Img}$, CWWV$_{Clip}$, to study how the strength of image–text association may affect the models. To obtain CWWV$_{Clip}$, we estimate the association quality of the image–word pairs of every QA instance in CWWV$_{Img}$ by measuring their CLIPScore (Hessel et al., 2021), and keep those instances with an average score of > 0.6.[12] The proportion of concrete words (concreteness scores > 4, Brysbaert et al., 2014) in CWWV$_{Clip}$ vs. CWWV$_{Img}$ are 35% vs. 27%, respectively.

Table 3 provides results for selected CS dimensions (we refer to Tab. 7+8 in App. C for all results). We see that while the retrieval-based multimodal models, UNITER$_{TV}$ and VILBERT$_{TV}$, do integrate information from the visual input, they both perform noticeably worse than their text-only counterparts, UNITER$_T$ and VILBERT$_T$, that do not get visual input during inference. Partially, this seems to be an effect of noise in the visual stream in the form of weakly-associated or abstract words. For some dimensions, including **part–whole**, **taxonomic** and **distinctness**, both UNITER$_{TV}$ and VILBERT$_{TV}$ yield higher accuracy scores on CWWV$_{Clip}$ which has more strongly associated and concrete image–text pairs than on CWWV$_{Img}$ (see Tab. 3). These performance gains cannot solely be attributed to a differing intrinsic difficulty level of CWWV$_{Clip}$, since the purely text-based models, in contrast, consistently perform worse on CWWV$_{Clip}$ than on CWWV$_{Img}$ (with a mean accuracy drop on CWWV$_{Clip}$ of -4.7pp and -1.5pp for RoBERTa and BERT, respectively; results shown in App. C, Tab. 7+8). In sum, our findings support our hypothesis that abstract concepts and weak cross-modal associations affect the inference ability of VL models, an issue we observed for **taxonomic** and **distinctness**.

[12]We determined this threshold with the mean CLIPScore under the image–word pair group that has absolute association according to human evaluation (§3.2).

Does Visual Input Alone Activate CS Knowledge? We address the question of whether visual context alone can provide substantial and informative cues with the *Vision-Only* mode, which disentangles the contribution of the visual from the textual modality. Both, UNITER$_V$ and VILBERT$_V$ perform much worse than when being also fed textual input (results not shown, see App. C). Yet, we find that visual context does play a beneficial role in some cases as opposed to under $\tilde{V}$-only mode, where the accuracy drops to random (UNITER) or close above random (VILBERT). Table 4 illustrates several cases for which visual cues alone can provide reasonable and sufficient information.

Regarding **(Q3)**, the visual stream of the VL models does not seem to play a dominant role during inference; nevertheless, in the extreme case where of missing textual information, the VL models rely on visual input for decision making.

6.3 Role of VL Model Architectures

If visual noise is indeed the reason for low inference abilities, then the models should fail when they receive only noisy, non-sensible visual input. We observe this effect in the single-stream UNITER$_{T\tilde{V}}$, where we see slight effectiveness drops when we feed in a set of unrelated (*dummy*) images $\tilde{V}_e$ along the textual prompt Q (§4.3) (-1.3pp/-2.4pp for All on CWWV$_{Img}$/CWWV$_{Clip}$, resp.; results shown in App. C, Tab. 7+8). Noise is only part of the reason, though, since overall, UNITER$_{TV}$ yields lower scores on CWWV$_{Clip}$ than on CWWV$_{Img}$ (-.6pp, see Tab. 7+8, App. C for all scores). Unexpectedly, the difference between T and TV is even larger on CWWV$_{Clip}$ than on CWWV$_{Img}$, this amplification may be explained by a higher distribution of concrete concepts in CWWV$_{Clip}$: It may be in particular the visual information of concrete concepts that the model can most effectively learn to ground better in the linguistic encoder, and then activate textually even if no visual inputs are given

during inference (Park and Myaeng, 2017). So, as can be assumed in a linguistic task, the textual stream seems to be the driving force for successful inference. An analysis of the *Modality Importance* (MI) score (Cao et al., 2020) of UNITER$_{TV}$ further supports this. The MI measures the average attention traces of the masked tokens in prompt Q during inference to determine the relevance of textual input vs. visual input (see App. D, Fig. 5 for visualizations and calculation details). The visualization of the average MI scores clearly shows a higher attention density on the textual than on the visual modality.

ViLBERT, in turn, which fell short against BERT and BERT$_{CC}$, behaves differently: On CWWV$_{Clip}$, ViLBERT$_{TV}$ more effectively includes visual input for decision making, with an overall accuracy that is closer to UNITER$_{TV}$ than it is on CWWV$_{Img}$ (+2.5pp, see Tab. 7+8). And on **taxonomic** and **spatial**, ViLBERT$_{TV}$'s accuracy is even higher than UNITER$_{TV}$'s (Tab. 3; the proportion of concrete concepts in CWWV$_{Clip}$ is 32% for taxonomic, 47% for spatial, and 34% for part-whole vs. 25%, 44%, and 32%, resp., in CWWV$_{Img}$). Finally, *Dummy-Mode* ViLBERT$_{T\tilde{V}}$ remarkably outperforms ViLBERT$_{TV}$ across all commonsense dimensions on CWWV$_{Img}$ (+4.9pp on All) and notably on **taxonomic**, **spatial** and **part–whole**, but is on par on CWWV$_{Clip}$ ($p < .05$). We examine the aspect of noise further with samples where ViLBERT$_{T\tilde{V}}$ predicts correctly but ViLBERT$_{TV}$ fails. We constantly find ViLBERT$_{T\tilde{V}}$ to only pay substantial attentions (> 0.3) to the same single visual token across all samples as well as in all CS dimensions (see App. E).

In summary, regarding **(Q4)**, in the case of single stream UNITER, it indicates that (explicit) visual input is beneficial for pretraining, but not for inference. In contrast, double-stream ViLBERT seems to be more dependent on receiving signals from both, text and vision, during inference, but also on a strong semantic image–text association, which it can more effectively use if provided. In case of noise, it relies on textual input for decision-making.

7 Conclusion

Regarding our research questions put forward in §1, our findings strongly suggest that the VL models learn to encode certain visual knowledge in their textual streams during multimodal pretraining, in particular for concrete concepts (see also Kiela et al., 2018), which they can activate from purely textual input during inference, i.e., visual information is not required, or not even beneficial. The fact that the textual stream is the driving force for inference is promising, given that we examined the benefit of VL models for a purely linguistic task.

Regarding the dependence of the architecture for commonsense acquisition and activation, we conclude that the examined single-stream seems to be better suited for text-only QA tasks, while double-stream seems to require some form of signal in the visual stream during inference (but cannot leverage it properly to the extent that it would be better than text-only input).

In summary, we find VL models to be promising regarding their potential use for natural language tasks requiring commonsense knowledge. We also identified a range of limitations for future work: The ability to handle visual noise, to understand events and verbs, and to integrate inconsistent modalities towards metaphorical, rather than situated understanding. Future work lies also on multimodal prompt-engineering for improved knowledge retrieval on commonsense intensive tasks.

Acknowledgments

We are grateful to Filip Ilievski and his co-authors for answering our questions regarding the consolidated commonsense knowledge base and the anonymous reviews for their constructive and valuable feedback.

References

Prithviraj Ammanabrolu, Wesley Cheung, William Broniec, and Mark O. Riedl. 2020. Automated storytelling via causal, commonsense plot ordering. *CoRR*, abs/2009.00829.

Peter Anderson, Xiaodong He, Chris Buehler, Damien Teney, Mark Johnson, Stephen Gould, and Lei Zhang. 2018. Bottom-up and top-down attention for image captioning and visual question answering. In *2018 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2018, Salt Lake City, UT, USA, June 18-22, 2018*, pages 6077–6086. IEEE Computer Society.

T W Anderson and D A Darling. 1954. A test of goodness of fit. *J. Am. Stat. Assoc.*, 49(268):765–769.

Hessam Bagherinezhad, Hannaneh Hajishirzi, Yejin Choi, and Ali Farhadi. 2016. Are elephants bigger

than butterflies? reasoning about sizes of objects. In *Proceedings of the Thirtieth AAAI Conference on Artificial Intelligence, February 12-17, 2016, Phoenix, Arizona, USA*, pages 3449–3456. AAAI Press.

Yonatan Bisk, Rowan Zellers, Ronan LeBras, Jianfeng Gao, and Yejin Choi. 2020. PIQA: reasoning about physical commonsense in natural language. In *The Thirty-Fourth AAAI Conference on Artificial Intelligence, AAAI 2020, The Thirty-Second Innovative Applications of Artificial Intelligence Conference, IAAI 2020, The Tenth AAAI Symposium on Educational Advances in Artificial Intelligence, EAAI 2020, New York, NY, USA, February 7-12, 2020*, pages 7432–7439. AAAI Press.

Antoine Bosselut, Hannah Rashkin, Maarten Sap, Chaitanya Malaviya, Asli Celikyilmaz, and Yejin Choi. 2019. COMET: Commonsense transformers for automatic knowledge graph construction. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 4762–4779, Florence, Italy. Association for Computational Linguistics.

Marc Brysbaert, Amy Beth Warriner, and Victor Kuperman. 2014. Concreteness ratings for 40 thousand generally known english word lemmas. *Behavior Research Methods*.

Emanuele Bugliarello, Ryan Cotterell, Naoaki Okazaki, and Desmond Elliott. 2021. Multimodal pretraining unmasked: A meta-analysis and a unified framework of vision-and-language BERTs. *Transactions of the Association for Computational Linguistics*, 9:978–994.

Jize Cao, Zhe Gan, Yu Cheng, Licheng Yu, Yen-Chun Chen, and Jingjing Liu. 2020. Behind the scene: Revealing the secrets of pre-trained vision-and-language models. *CoRR*, abs/2005.07310.

Qian Chen, Xiaodan Zhu, Zhen-Hua Ling, Diana Inkpen, and Si Wei. 2018. Neural natural language inference models enhanced with external knowledge. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 2406–2417, Melbourne, Australia. Association for Computational Linguistics.

Xinlei Chen, Abhinav Shrivastava, and Abhinav Gupta. 2013. NEIL: extracting visual knowledge from web data. In *IEEE International Conference on Computer Vision, ICCV 2013, Sydney, Australia, December 1-8, 2013*, pages 1409–1416. IEEE Computer Society.

Yen-Chun Chen, Linjie Li, Licheng Yu, Ahmed El Kholy, Faisal Ahmed, Zhe Gan, Yu Cheng, and Jingjing Liu. 2020. UNITER: universal image-text representation learning. In *Computer Vision - ECCV 2020 - 16th European Conference, Glasgow, UK, August 23-28, 2020, Proceedings, Part XXX*, volume 12375 of *Lecture Notes in Computer Science*, pages 104–120. Springer.

Wanqing Cui, Yanyan Lan, Liang Pang, Jiafeng Guo, and Xueqi Cheng. 2020. Beyond language: Learning commonsense from images for reasoning. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 4379–4389, Online. Association for Computational Linguistics.

Joe Davison, Joshua Feldman, and Alexander Rush. 2019. Commonsense knowledge mining from pretrained models. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 1173–1178, Hong Kong, China. Association for Computational Linguistics.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, Minneapolis, Minnesota. Association for Computational Linguistics.

Ronald A Fisher. 1949. The design of experiments.

Joseph L Fleiss. 1971. Measuring nominal scale agreement among many raters. *Psychological bulletin*, 76(5):378.

Stella Frank, Emanuele Bugliarello, and Desmond Elliott. 2021. Vision-and-language or vision-for-language? on cross-modal influence in multimodal transformers. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 9847–9857, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Jonathan Gordon and Benjamin Van Durme. 2013. Reporting bias and knowledge acquisition. In *Proceedings of the 2013 workshop on Automated knowledge base construction, AKBC@CIKM 13, San Francisco, California, USA, October 27-28, 2013*, pages 25–30. ACM.

Raghav Goyal, Samira Ebrahimi Kahou, Vincent Michalski, Joanna Materzynska, Susanne Westphal, Heuna Kim, Valentin Haenel, Ingo Fründ, Peter Yianilos, Moritz Mueller-Freitag, Florian Hoppe, Christian Thurau, Ingo Bax, and Roland Memisevic. 2017. The "something something" video database for learning and evaluating visual common sense. In *IEEE International Conference on Computer Vision, ICCV 2017, Venice, Italy, October 22-29, 2017*, pages 5843–5851. IEEE Computer Society.

Lisa Anne Hendricks and Aida Nematzadeh. 2021. Probing image-language transformers for verb understanding. In *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*, pages 3635–3644, Online. Association for Computational Linguistics.

Jack Hessel, Ari Holtzman, Maxwell Forbes, Ronan Le Bras, and Yejin Choi. 2021. CLIPScore: A reference-free evaluation metric for image captioning. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 7514–7528, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Lifu Huang, Ronan Le Bras, Chandra Bhagavatula, and Yejin Choi. 2019. Cosmos QA: Machine reading comprehension with contextual commonsense reasoning. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 2391–2401, Hong Kong, China. Association for Computational Linguistics.

Filip Ilievski, Alessandro Oltramari, Kaixin Ma, Bin Zhang, Deborah L. McGuinness, and Pedro A. Szekely. 2021. Dimensions of commonsense knowledge. *CoRR*, abs/2101.04640.

Filip Ilievski, Pedro A. Szekely, Jingwei Cheng, Fu Zhang, and Ehsan Qasemi. 2020. Consolidating commonsense knowledge. *CoRR*, abs/2006.06114.

Hamid Izadinia, Fereshteh Sadeghi, Santosh Kumar Divvala, Hannaneh Hajishirzi, Yejin Choi, and Ali Farhadi. 2015. Segment-phrase table for semantic segmentation, visual entailment and paraphrasing. In *2015 IEEE International Conference on Computer Vision, ICCV 2015, Santiago, Chile, December 7-13, 2015*, pages 10–18. IEEE Computer Society.

Stanislaw Jastrzębski, Dzmitry Bahdanau, Seyedarian Hosseini, Michael Noukhovitch, Yoshua Bengio, and Jackie Cheung. 2018. Commonsense mining as knowledge base completion? a study on the impact of novelty. In *Proceedings of the Workshop on Generalization in the Age of Deep Learning*, pages 8–16, New Orleans, Louisiana. Association for Computational Linguistics.

Zhengbao Jiang, Frank F. Xu, Jun Araki, and Graham Neubig. 2020. How can we know what language models know? *Transactions of the Association for Computational Linguistics*, 8:423–438.

Douwe Kiela, Alexis Conneau, Allan Jabri, and Maximilian Nickel. 2018. Learning visually grounded sentence representations. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 408–418, New Orleans, Louisiana. Association for Computational Linguistics.

Ranjay Krishna, Yuke Zhu, Oliver Groth, Justin Johnson, Kenji Hata, Joshua Kravitz, Stephanie Chen, Yannis Kalantidis, Li-Jia Li, David A. Shamma, Michael S. Bernstein, and Li Fei-Fei. 2017. Visual genome: Connecting language and vision using crowdsourced dense image annotations. *Int. J. Comput. Vis.*, 123(1):32–73.

Teven Le Scao and Alexander Rush. 2021. How many data points is a prompt worth? In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 2627–2636, Online. Association for Computational Linguistics.

Zhicheng Liang and Deborah L. McGuinness. 2021. Commonsense knowledge mining from term definitions. *CoRR*, abs/2102.00651.

Pengfei Liu, Weizhe Yuan, Jinlan Fu, Zhengbao Jiang, Hiroaki Hayashi, and Graham Neubig. 2021. Pre-train, Prompt, and Predict: A Systematic Survey of Prompting Methods in Natural Language Processing. *CoRR*, abs/2107.13586.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Roberta: A robustly optimized BERT pretraining approach. *CoRR*, abs/1907.11692.

Jiasen Lu, Dhruv Batra, Devi Parikh, and Stefan Lee. 2019. Vilbert: Pretraining task-agnostic visiolinguistic representations for vision-and-language tasks. In *Advances in Neural Information Processing Systems 32: Annual Conference on Neural Information Processing Systems 2019, NeurIPS 2019, December 8-14, 2019, Vancouver, BC, Canada*, pages 13–23.

Kaixin Ma, Filip Ilievski, Jonathan Francis, Yonatan Bisk, Eric Nyberg, and Alessandro Oltramari. 2021. Knowledge-driven data construction for zero-shot evaluation in commonsense question answering. *AAAI*, 35(15):13507–13515.

Bodhisattwa Prasad Majumder, Harsh Jhamtani, Taylor Berg-Kirkpatrick, and Julian McAuley. 2020. Like hiking? you probably enjoy nature: Persona-grounded dialog with commonsense expansions. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 9194–9206, Online. Association for Computational Linguistics.

Todor Mihaylov and Anette Frank. 2018. Knowledgeable reader: Enhancing cloze-style reading comprehension with external commonsense knowledge. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 821–832, Melbourne, Australia. Association for Computational Linguistics.

George A. Miller. 1995. Wordnet: A lexical database for english. *Commun. ACM*, 38(11):39–41.

Joohee Park and Sung-hyon Myaeng. 2017. A computational study on word meanings and their distributed representations via polymodal embedding. In *Proceedings of the Eighth International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 214–223, Taipei, Taiwan. Asian Federation of Natural Language Processing.

Fabio Petroni, Tim Rocktäschel, Sebastian Riedel, Patrick Lewis, Anton Bakhtin, Yuxiang Wu, and Alexander Miller. 2019. Language models as knowledge bases? In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 2463–2473, Hong Kong, China. Association for Computational Linguistics.

Kyle Richardson and Ashish Sabharwal. 2020. What does my QA model know? devising controlled probes using expert knowledge. *Transactions of the Association for Computational Linguistics*, 8:572–588.

Fereshteh Sadeghi, Santosh Kumar Divvala, and Ali Farhadi. 2015. Viske: Visual knowledge extraction and question answering by visual verification of relation phrases. In *IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2015, Boston, MA, USA, June 7-12, 2015*, pages 1456–1464. IEEE Computer Society.

Henrique Santos, Minor Gordon, Zhicheng Liang, Gretchen Forbush, and Deborah L. McGuinness. 2020. Exploring and analyzing machine commonsense benchmarks. *CoRR*, abs/2012.11634.

Piyush Sharma, Nan Ding, Sebastian Goodman, and Radu Soricut. 2018. Conceptual captions: A cleaned, hypernymed, image alt-text dataset for automatic image captioning. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 2556–2565, Melbourne, Australia. Association for Computational Linguistics.

Taylor Shin, Yasaman Razeghi, Robert L. Logan IV, Eric Wallace, and Sameer Singh. 2020. AutoPrompt: Eliciting Knowledge from Language Models with Automatically Generated Prompts. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 4222–4235, Online. Association for Computational Linguistics.

Vered Shwartz and Yejin Choi. 2020. Do neural language models overcome reporting bias? In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 6863–6870, Barcelona, Spain (Online). International Committee on Computational Linguistics.

Shikhar Singh, Nuan Wen, Yu Hou, Pegah Alipoormolabashi, Te-lin Wu, Xuezhe Ma, and Nanyun Peng. 2021. COM2SENSE: A commonsense reasoning benchmark with complementary sentences. In *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*, pages 883–898, Online. Association for Computational Linguistics.

Robyn Speer, Joshua Chin, and Catherine Havasi. 2017. Conceptnet 5.5: An open multilingual graph of general knowledge. In *Proceedings of the Thirty-First AAAI Conference on Artificial Intelligence, February 4-9, 2017, San Francisco, California, USA*, pages 4444–4451. AAAI Press.

Kevin Stowe, Tuhin Chakrabarty, Nanyun Peng, Smaranda Muresan, and Iryna Gurevych. 2021. Metaphor generation with conceptual mappings. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 6724–6736, Online. Association for Computational Linguistics.

Alexandre Tamborrino, Nicola Pellicanò, Baptiste Pannier, Pascal Voitot, and Louise Naudin. 2020. Pre-training is (almost) all you need: An application to commonsense reasoning. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 3878–3887, Online. Association for Computational Linguistics.

Hao Tan and Mohit Bansal. 2019. LXMERT: Learning cross-modality encoder representations from transformers. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 5100–5111, Hong Kong, China. Association for Computational Linguistics.

Zineng Tang, Jaemin Cho, Hao Tan, and Mohit Bansal. 2021. Vidlankd: Improving language understanding via video-distilled knowledge transfer. *CoRR*, abs/2107.02681.

Denny Vrandecic and Markus Krötzsch. 2014. Wikidata: a free collaborative knowledgebase. *Commun. ACM*, 57(10):78–85.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Rémi Louf, Morgan Funtowicz, and Jamie Brew. 2019. Huggingface's transformers: State-of-the-art natural language processing. *CoRR*, abs/1910.03771.

An Yang, Quan Wang, Jing Liu, Kai Liu, Yajuan Lyu, Hua Wu, Qiaoqiao She, and Sujian Li. 2019. Enhancing pre-trained language representations with rich knowledge for machine reading comprehension. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 2346–2357, Florence, Italy. Association for Computational Linguistics.

Mark Yatskar, Vicente Ordonez, and Ali Farhadi. 2016. Stating the obvious: Extracting visual common sense knowledge. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 193–198, San Diego, California. Association for Computational Linguistics.

Tian Yun, Chen Sun, and Ellie Pavlick. 2021. Does vision-and-language pretraining improve lexical

grounding? In *Findings of the Association for Computational Linguistics: EMNLP 2021*, pages 4357–4366, Punta Cana, Dominican Republic. Association for Computational Linguistics.

Rowan Zellers, Ari Holtzman, Yonatan Bisk, Ali Farhadi, and Yejin Choi. 2019. HellaSwag: Can a machine really finish your sentence? In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 4791–4800, Florence, Italy. Association for Computational Linguistics.

Rowan Zellers, Mark Yatskar, Sam Thomson, and Yejin Choi. 2018. Neural motifs: Scene graph parsing with global context. In *2018 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2018, Salt Lake City, UT, USA, June 18-22, 2018*, pages 5831–5840. IEEE Computer Society.

Hongming Zhang, Yintong Huo, Xinran Zhao, Yangqiu Song, and Dan Roth. 2020a. Learning contextual causality from time-consecutive images. *CoRR*, abs/2012.07138.

Zhuosheng Zhang, Kehai Chen, Rui Wang, Masao Utiyama, Eiichiro Sumita, Zuchao Li, and Hai Zhao. 2020b. Neural machine translation with universal visual representation. In *International Conference on Learning Representations*.

Zexuan Zhong, Dan Friedman, and Danqi Chen. 2021. Factual probing is [MASK]: Learning vs. learning to recall. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 5017–5033, Online. Association for Computational Linguistics.

Ben Zhou, Daniel Khashabi, Qiang Ning, and Dan Roth. 2019. "going on a vacation" takes longer than "going for a walk": A study of temporal commonsense understanding. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 3363–3369, Hong Kong, China. Association for Computational Linguistics.

Pei Zhou, Rahul Khanna, Bill Yuchen Lin, Daniel Ho, Xiang Ren, and Jay Pujara. 2020a. Can BERT reason? logically equivalent probes for evaluating the inference capabilities of language models. *CoRR*, abs/2005.00782.

Xuhui Zhou, Yue Zhang, Leyang Cui, and Dandan Huang. 2020b. Evaluating commonsense in pre-trained language models. In *The Thirty-Fourth AAAI Conference on Artificial Intelligence, AAAI 2020, The Thirty-Second Innovative Applications of Artificial Intelligence Conference, IAAI 2020, The Tenth AAAI Symposium on Educational Advances in Artificial Intelligence, EAAI 2020, New York, NY, USA, February 7-12, 2020*, pages 9733–9740. AAAI Press.

A Details to the Creation of CWWV

A.1 Data Sources

Following Ma et al. (2021), we use the knowledge sources whose relations can be mapped to ConceptNet relation labels, viz. ConceptNet (Speer et al., 2017), WordNet (Miller, 1995), Visual Genome (Krishna et al., 2017) and Wikidata (Vrandecic and Krötzsch, 2014). ConceptNet represents commonsense knowledge in a graph structure of concept nodes connected by relational edges. WordNet focuses on lexical taxonomic knowledge. Visual Genome is a resource of images densely annotated with region descriptions that describe the depicted objects, their attributes and relationships, and which can be represented as scene graphs. Wikidata is a relational knowledge base of entities.

A.2 Relations

Since, in contrast to Ma et al. (2021), our goal is not to pretrain models on selected knowledge relations, but to reach a high coverage of relations for model evaluation, we consider more relations (30 in total), covered by 10 dimensions (Ma et al. (2021) used 14 and 7, resp., refer to Table 5 for the mappings between the commonsense dimensions and knowledge relations evaluated here). We do not consider the relations *lexical*, *comparative*, and *relational-other*. According to (Ma et al., 2021)'s categorization, ConceptNet does not support any relation type that can be mapped to *comparative*. *Relational-other* clusters noisier relations, which may counteract a clean evaluation. *Lexical* requires understanding of formal linguistic knowledge, which is not our target here.

B Retrieving Images of CWWV$_{Img}$

B.1 Topic-Image Lookup Table

Topic words are identified through term frequency-inverse document frequency (TF-IDF) weight. For a preprocessed caption containing non-stop words only $C_j = \{w_1, w_2, ..., w_l\}$. The TF-IDF weight $w_{i,j}$ of each word w_i in a caption C_j is computed:

$$w_{i,j} = \frac{n_{i,j}}{\sum_k n_{k,j}} \times \log \frac{|C|}{1 + |C_j : w_i \in C_j|}$$

where the term frequency of w_i in C_j is calculated by dividing its appearances $n_{i,j}$ by the total number of words in C_j; inverse document frequency is computed by taking the inverse proportion of the number of captions in which w_i occurs $|C_j : w_i \in C_j|$ within a batch of captions $|C|$.

Dimension	Relation Type	Template
part-whole	/r/PartOf	h is a part of t
	/r/HasA	h has a t
	/r/MadeOf	h can be made of t
taxonomic	/r/IsA	h is a t
	/r/InstanceOf	h has an instance of t
	/r/MannerOf	h is a way to t
distinctness	/r/Antonym	h is the opposite of t
	/r/DistinctFrom	h is not t
similarity	/r/Synonym	h is a synonym of t
	/r/SimilarTo	h is similar to t
	/r/DefinedAs	h is the t
quality	/r/HasProperty	h is/are t
	/r/NotHasProperty	h is/are not t
	/r/SymbolOf	h is a symbol of t
utility	/r/ReceivesAction	h can be t
	/r/UsedFor	h is/are for t
	/r/CapableOf	h can t
	/r/NotCapableOf	h do not t
creation	/r/CreatedBy	h is created by t
temporal	/r/HasFirstSubevent	The first thing you do when you h is t
	/r/HasLastSubevent	The last thing you do when you h is t
	/r/HasSubevent	Something that might happen when you h is t
	/r/HasPrerequisite	h requires t
	/r/Causes	The effect of h is t
spatial	/r/AtLocation	You are likely to find h in t
desire	/r/CausesDesire	h would make you want to t
	/r/MotivatedByGoal	You would h because you want t
	/r/Desires	h wants to t
	/r/NotDesires	h doesn't want t
	/r/ObstructedBy	h is obstructed by t

Table 5: Knowledge dimensions with their clustered knowledge relation types and the corresponding lexicalized templates evaluated in this work.

Each caption is now a sequence of topic words sorted according to their TF-IDF weights, we take the top-k topic words to represent the new caption $C'_j = \{t_1, t_2, ..., t_k\}$. We save the lemma form of t_i and its paired image I_j accompanied by its computed TF-IDF weight into the topic-image lookup table T, where each topic is mapped to several images because of its multiple occurrences in different image-caption pairs. Under the assumption that there exists alignment between each image-caption pair, the TF-IDF weights can be further treated as an approximation of how relevant the paired image depicts the topic. The higher the TF-IDF weight, the better the paired image captures the theme of a topic.

B.2 Image-word Pair Association Analysis

We assess the effectiveness of our simple retrieval approach through a human annotation study on Amazon Mechanical Turk (AMT) in which we ask workers to judge the association of image–word pairs. To this end, we sample $1,000$ pairs from CWWV$_{Img}$ uniformly across the 10 commonsense dimensions. Each HIT comprises a random sequence of 10 image–words pairs, one for each dimension. For each HIT, we ask 3 AMT workers to judge each pair as either "associated" or "not associated". We define an image as "associated" to its paired word when it can "successfully capture the word's meaning by either containing the object, picturing the event, depicting the action, or characterizing the appearance, emotion, or manner that the word can describe". For polysemous words (e.g., *clean* can refer to an action or appearance), the workers are encouraged to judge whether the image can capture at least one sense of the word.

In total, 37 workers[13] participated; we paid 0.20 per HIT with an hourly wage of $12. According to majority vote (2 out of 3), 64.2% of the pairs are associated, among which *part-whole* and *spatial* have more than average associated pairs (74.3% and 70.2% respectively). The inter-annotator agreement under Fleiss' Kappa coefficient (Fleiss, 1971) is between $0.21 - 0.44$ across dimensions, which is only a fair to moderate agreement, indicating the high subjectivity of this task.

Since some words are inherently more visualizable, we further analyze the pair association score from different facets, such as POS tags and word concreteness (Brysbaert et al., 2014). Table 6 shows the analysis of majority association score and Fleiss Kappa score, broke down into three categories: commonsense dimensions, POS tags and word concreteness. POS tags are recognized using spacy-nlp package while word concreteness is categorized based on the concreteness ratings (5 point scale from abstract to concrete) provided by Brysbaert et al. (2014). Brysbaert et al. (2014) defined concrete words as those that can be directly experienced through senses (e.g. sweetness can be experienced through tasting) whereas abstract words can only be inferred from the linguistic context; they surveyed at least 25 annotations for each word in the list (37,058 words and 2,896 two-words expressions). To obtain a categorical analysis, we define words that receive mean concreteness rating above 4 as *concrete*.

Figure 3 and Figure 4 display concrete words distribution and POS tags distribution over each commonsense dimension respectively, where we see that both *spatial* and *part-whole* dimensions contain more concrete words.

C Results of CWWV$_{Img}$ and CWWV$_{Clip}$

The full results of CWWV$_{Img}$ and CWWV$_{Clip}$ can be found in Tables 7 and 8, respectively.

[13]Workers must reside in the UK, USA, CA, NZ, or AU.

	Majority Association (Absolute Association)	Fleiss Kappa
Commonsense Dimensions		
part-whole	70.3% (42.9%)	0.32
taxonomic	64.9% (31.6%)	0.21
distinctness	63.2% (33.7%)	0.38
similarity	61.1% (35.8%)	0.27
quality	51.5% (24.7%)	0.23
utility	63.2% (32.6%)	0.27
creation	59.4% (40.6%)	0.44
temporal	64.9% (39.4%)	0.33
spatial	74.2% (44.3%)	0.25
desire	69.1% (41.5%)	0.21
POS tags		
NOUN	68.6% (42.9%)	0.33
VERB	58.4% (27.9%)	0.24
ADJ	62.1% (30.1%)	0.24
Word Concreteness		
Conc.	81.2% (57.0%)	0.30
Non-Conc.	58.5% (29.9%)	0.28 0.28
All	64.2% (36.7%)	0.30

Table 6: Majority association measures how often the image-word pair is annotated as " associated" by the majority of the annotators (2 out of 3) whereas absolute association refers to the whole agreement. Fleiss Kappa score is the inter-annotator agreement.

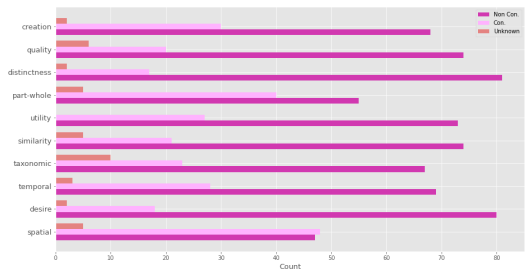

Figure 3: concrete words distribution across commonsense dimensions

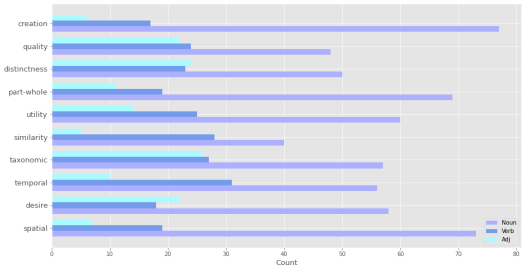

Figure 4: POS tags distribution across commonsense dimensions

D Modality Importance Scores

Following Cao et al. (2020), we analyze which modality of single-stream model (*textual* v.s. *visual*) is more dominant during inference by examining the modality importance score (MI score). In particular, we are interested in the average attention traces on the `[MASK]` tokens that refer to the head h of the original knowledge triple (h, r, t) before it is transformed into a QA statement (i.e., `[MASK]` tokens that represent the prompt template, e.g., "You are likely to find X in', are not considered). Similar to Cao et al. (2020), for the textual modality we disregard the attention values spent on the two special tokens `[CLS]` and `[SEP]`; analogously, for the visual modality, the attention value paid to the `[IMG]` is also ignored. Therefore, the MI scores of the modalities do not sum up to 1.

For a sequence of bimodal tokens, $S = (\texttt{[CLS]}, t_1, ..., t_m, \texttt{[SEP]}, v_1, ..., v_n)$, where $t_1, ..., t_m$ refer to the textual tokens, and $v_1, ..., v_n$ denote the visual ones, the average MI score $\overline{I_{M,j}}$ for each attention head j is calculated as follows:

$$\overline{I_{M,j}} = \frac{1}{L_h} \sum^{L_h} \sum_{i \in S} \mathbb{1}(i \in M) \cdot \alpha_{i,j}$$

$\alpha_{i,j}$ refers to the attention score of the `[MASK]` token spends on the token i at head j The MI score of each respective commonsense dimension can be seen in Figure 6; Figure 5 gives the mean MI scores across all dimensions.

The visualization of the average MI scores shows a clearly higher attention density on the textual than on the visual modality (Fig. 6). We also observe

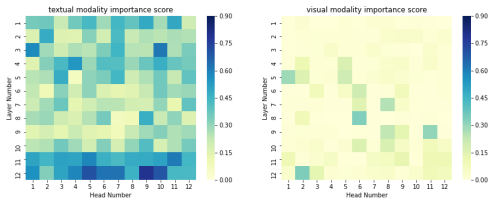

Figure 5: Visualization of MI scores of UNITER$_{TV}$ across 144 attention heads for the cases where UNITER$_{TV}$ is correct and BERT incorrect. Left: textual MI; right: visual MI.

that the MI scores vary across commonsense dimensions. We see a *higher* attention density on the *lower textual* layers, and a *low* density on the *visual* parts on **spatial, temporal, desire**; on the **other dimensions**, we see a higher density on the *upper textual* layers and overall a *higher* density on the *intermediate visual* layers.

E Results with Dummy Images

The visualization of attention traces of VILBERT$_{T\tilde{V}}$ is displayed in Figure. 7.

row	Images	part-whole 1, 165	taxonomic 1, 323	distinctness 828	similarity 644	quality 1, 840	utility 2, 090	creation 100	temporal 1, 189	spatial 1, 599	desire 1, 781	All 13, 259
1 RoBERTa	–	68.5	61.8	80.2	**67.4**	69.7	**74.2**	72.0	**60.9**	54.8	**65.9**	**67.5**
2 BERT	–	62.8	71.2	80.1	54.8	68.1	72.4	74.0	53.7	52.4	60.4	65.0
3 BERT$_{CC}$	–	68.4	62.0	66.6	51.1	66.0	65.4	62.0	53.6	**63.7**	58.3	61.9
4 UNITER_BERT$_{T}$	–	70.1	**74.5**	**81.4**	62.4	**72.0**	73.8	**79.0**	54.5	53.9	61.5	66.5
5 UNITER$_{T}$	–	**70.9**	59.8	71.3	51.2	69.9	71.5	71.0	52.7	61.5	62.5	64.0
6 VILBERT$_{T}$	–	63.9	60.3	64.9	46.7	66.1	71.2	58.0	52.2	61.0	62.8	60.7
7 UNITER$_{TV}$	retrieved	63.0	54.0	65.9	46.4	62.4	65.4	62.0	49.2	57.4	58.5	58.4
8 VILBERT$_{TV}$	retrieved	55.0	49.9	55.9	42.2	57.4	60.5	52.0	47.2	52.9	56.6	53.0
9 UNITER$_{T\tilde{V}}$	dummy	61.5	51.6	63.4	42.2	63.6	66.4	55.0	49.4	58.2	59.7	57.1
10 VILBERT$_{T\tilde{V}}$	dummy	60.4	58.9	64.9	43.9	63.4	65.5	55.0	48.4	56.8	62.0	57.9
11 UNITER$_{V}$	retrieved	36.4	36.6	40.1	38.5	34.2	36.6	32.0	34.8	36.2	34.3	36.0
12 VILBERT$_{V}$	retrieved	37.8	35.1	37.7	39.8	36.8	35.7	41.0	33.0	37.6	34.0	36.8
13 UNITER$_{\tilde{V}}$	dummy	30.8	26.3	45.7	28.6	29.2	28.7	19.0	28.7	29.6	30.7	29.7
14 VILBERT$_{\tilde{V}}$	dummy	34.8	35.8	50.5	40.4	30.4	31.1	30.0	29.4	33.5	30.1	34.6

Table 7: Model accuracy on CWWV$_{Img}$. **Bold** represents the highest score per commonsense dimension across all models and settings; underlined scores denote the best model under the same setting per commonsense dimension.

row	Images	part-whole 170	taxonomic 85	distinctness 86	similarity 188	quality 143	utility 120	creation 8	temporal 154	spatial 144	desire 91	All 1, 189
1 RoBERTa	–	70.2	60.1	**75.8**	**61.6**	64.3	68.3	75.0	**56.5**	52.4	57.6	62.8
2 BERT	–	61.2	70.6	73.6	45.3	69.5	65.8	**87.5**	52.9	48.2	57.6	63.2
3 BERT$_{CC}$	–	71.8	61.5	65.9	44.2	66.9	64.2	75.0	42.4	**61.8**	52.1	60.6
4 UNITER_BERT$_{T}$	–	69.6	**72.7**	**75.8**	55.8	**70.8**	68.3	**87.5**	**56.5**	47.7	59.0	**64.3**
5 UNITER$_{T}$	–	**76.6**	58.0	72.5	52.3	66.9	**70.0**	75.0	44.7	59.4	54.9	63.0
6 VILBERT$_{T}$	–	70.7	59.4	62.6	53.5	63.6	**70.0**	62.5	43.5	58.8	**59.7**	60.5
7 UNITER$_{TV}$	retrieved	68.1	55.9	68.1	43.0	61.0	64.2	62.5	42.4	54.1	58.3	57.8
8 VILBERT$_{TV}$	retrieved	58.0	58.0	62.6	38.4	57.8	56.7	62.5	41.2	59.4	54.2	54.9
9 UNITER$_{T\tilde{V}}$	dummy	66.0	51.7	62.6	45.3	62.3	69.2	37.5	42.4	59.4	57.6	55.4
10 VILBERT$_{T\tilde{V}}$	dummy	62.8	54.5	63.7	41.9	58.4	63.3	50.0	44.7	56.5	59.0	55.5
11 UNITER$_{V}$	retrieved	33.5	38.5	45.1	40.7	37.0	31.7	37.5	35.3	35.3	34.0	36.9
12 VILBERT$_{V}$	retrieved	39.4	34.3	41.8	37.2	35.1	40.0	25.0	31.8	40.6	31.2	35.6
13 UNITER$_{\tilde{V}}$	dummy	29.3	19.6	48.4	36.0	23.4	26.7	25.0	34.1	28.2	27.8	29.8
14 VILBERT$_{\tilde{V}}$	dummy	38.8	30.8	49.5	38.4	29.9	30.0	25.0	28.2	34.1	29.2	33.4

Table 8: Model accuracy on CWWV$_{Clip}$. **Bold** represents the highest score per commonsense dimension across all models and settings; underlined scores denote the best model under the same setting per commonsense dimension.

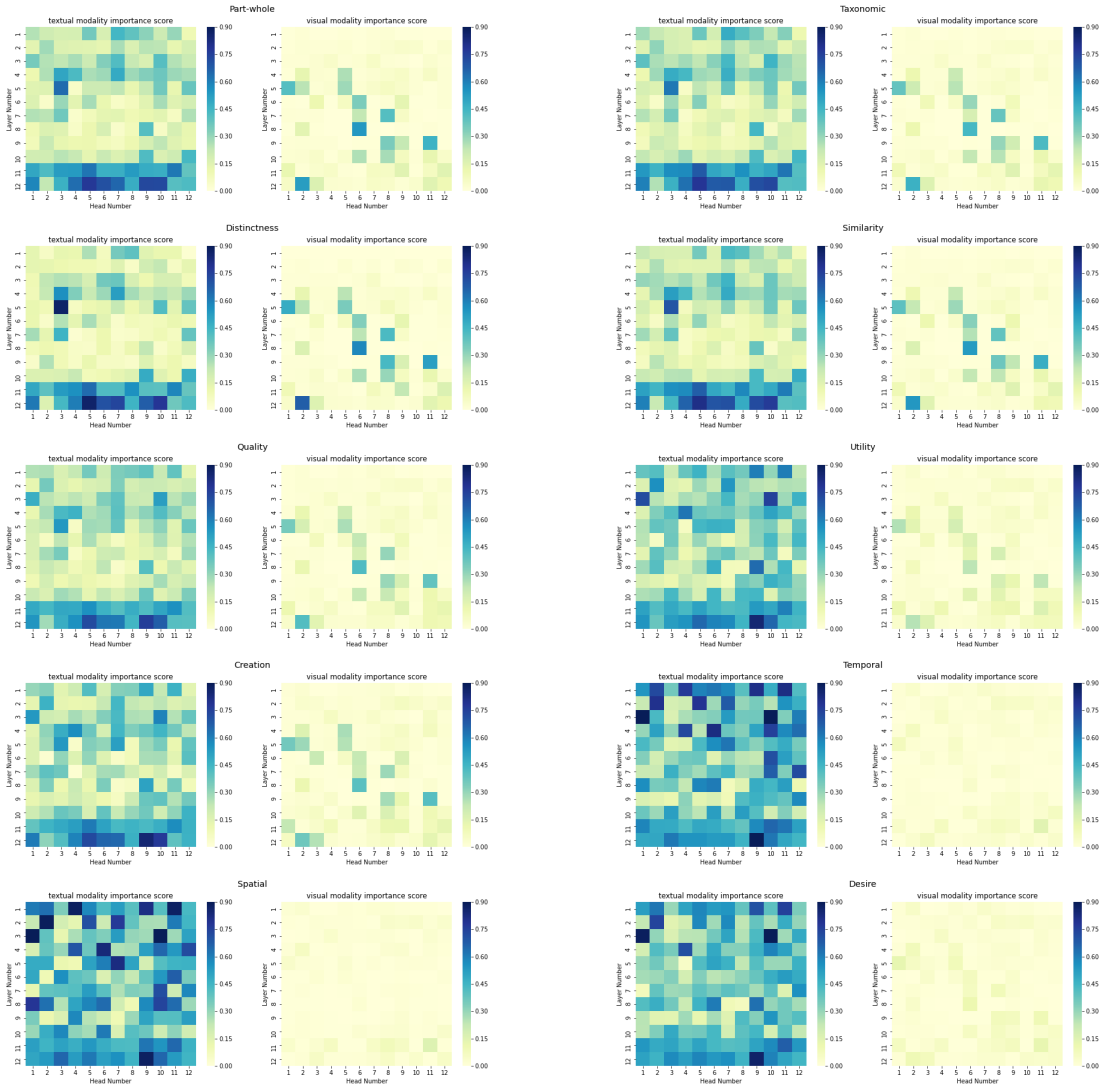

Figure 6: Visualization of the average modality importance scores of UNITER$_{TV}$ across 144 attention heads and across commonsense dimensions under the cases where UNITER$_{TV}$ predicts correctly whereas BERT predicts incorrectly. The order (from top to bottom and from left to right) of commonsense dimension is: part-whole, taxonomic, distinctness, similarity, quality, utility, creations, temporal, spatial, desire.

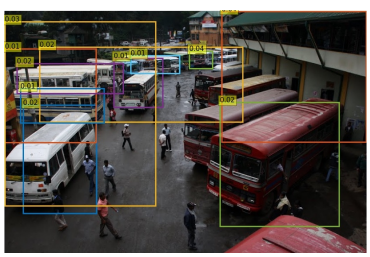

(a) Dummy Image 1

(b) Dummy Image 2

(c) Dummy Image 3

Figure 7: Attention traces of VILBERT$_{T\tilde{V}}$ last inter-modal layer averaged across cases where VILBERT$_{T\tilde{V}}$ predicts correctly. The bounding boxes represent salient visual tokens of each dummy image and the values in yellow boxes refer to the averaged attention scores. VILBERT$_{T\tilde{V}}$ overly pays attention (0.35) to one single visual token in the second dummy image (7b).

Visual Prompt Tuning for Few-Shot Text Classification

Jingyuan Wen[1,2], Yutian Luo[1,2], Nanyi Fei[1,2], Guoxing Yang[1,2], Zhiwu Lu[1,2,*]
Hao Jiang[3], Jie Jiang[3], Zhao Cao[3]
[1]Gaoling School of Artificial Intelligence, Renmin University of China, Beijing, China
[2]Beijing Key Laboratory of Big Data Management and Analysis Methods
[3]Huawei Poisson Lab, Hangzhou, Zhejiang
{wenjingyuan,luzhiwu}@ruc.edu.cn

Abstract

Deploying large-scale pre-trained models in the prompt-tuning paradigm has demonstrated promising performance in few-shot learning. Particularly, vision-language pre-training models (VL-PTMs) have been intensively explored in various few-shot downstream tasks. However, most existing works only apply VL-PTMs to visual tasks like image classification, with few attempts being made on language tasks like text classification. In few-shot text classification, a feasible paradigm for deploying VL-PTMs is to align the input samples and their category names via the text encoders. However, it leads to the waste of visual information learned by the image encoders of VL-PTMs. To overcome this drawback, we propose a novel method named Visual Prompt Tuning (VPT). To our best knowledge, this method is the first attempt to deploy VL-PTM in few-shot text classification task. The main idea is to generate the image embeddings w.r.t. category names as visual prompt and then add them to the aligning process. Extensive experiments show that our VPT can achieve significant improvements under both zero-shot and few-shot settings. Importantly, our VPT even outperforms the most recent prompt-tuning methods on five public text classification datasets.

1 Introduction

Pre-training models have achieved great success across a variety of tasks in recent years. Pre-training language models (PLMs) like BERT (Devlin et al., 2019), GPT (Radford et al., 2018) and their variants (Liu et al., 2019; Raffel et al., 2020; Yang et al., 2019; Lewis et al., 2020) firstly appeared as the milestones in the AI field. They brought huge boost to natural language processing (NLP) tasks, such as text classification (Devlin et al., 2019), named entity recognition (NER) (Jia et al., 2020), and text generation (Chan and Fan, 2019). In computer vision, large-scale pre-training models (e.g., BiT (Kolesnikov et al., 2020) and ViT (Dosovitskiy et al., 2021)) became popular as in NLP. With convolutional neural networks or Transformers (Vaswani et al., 2017) as the backbones, they were shown to be effective on a wide range of visual downstream tasks (e.g., image classification, object detection, and semantic segmentation). More recently, inspired by these pre-training models in NLP and computer vision, vision-language pre-training models (VL-PTMs) have been intensively explored (Su et al., 2020; Li et al., 2020; Huo et al., 2021; Lu et al., 2022; Fei et al., 2022). They achieve excellent performance in cross-modal tasks like image-text retrieval, visual question answering (VQA), and image caption. Besides, they also show great potential in single-modal tasks (Lin et al., 2021; Yuan et al., 2021). These achievements clearly declare the power of large-scale pre-training models.

With GPT-3 (Brown et al., 2020) demonstrating astonishing zero-shot and few-shot ability, researchers are encouraged to explore the potential of large-scale pre-training models in few-shot learning. Recently, prompt-tuning has been widely used in few-shot tasks as a paradigm for deploying pre-training models. Compared with prompt-tuning, the performance of the traditional fine-tuning paradigm has apparent drawbacks when only few training samples are available (Schick and Schütze, 2021). PLM based prompt-tuning methods (e.g. Prefix-tuning (Li and Liang, 2021), P-tuning (Liu et al., 2021b), ADAPET (Tam et al., 2021)) have shown their effectiveness and robustness on NLP tasks. Meanwhile, VL-PTM based prompt-tuning methods like CoOp (Zhou et al., 2021), Clip-Adapter (Gao et al., 2021a) and CPT (Yao et al., 2021) apply VL-PTMs to few-shot visual tasks including few-shot image classification and visual ground-

*Zhiwu Lu is the corresponding author.

Proceedings of the 29th International Conference on Computational Linguistics, pages 5560–5570
October 12–17, 2022.

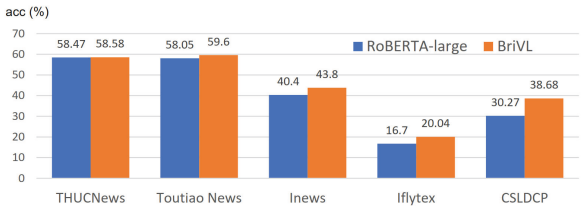

Figure 1: Zero-shot results on five public text datasets. The cross-modal model BriVL (Fei et al., 2022) is shown to outperform the comparably-sized single-modal model RoBERTa-large, indicating that the visual information may bring benefits to textual tasks.

ing. These successes reveal that textual information is beneficial for visual tasks. However, there still lacks a method to utilize VL-PTMs in few-shot NLP tasks like few-shot text classification. Importantly, we notice that the cross-modal model BriVL (Fei et al., 2022) achieves better zero-shot results than the comparably-sized single-modal model RoBERTa-large on five public text classification datasets (see Figure 1), indicating that the visual information may bring benefits to textual tasks. This thus motivates us to introduce VL-PTM into few-shot text classification.

In this work, we propose a novel method named Visual Prompt Tuning (VPT) for few-shot text classification. It is a prompt-tuning method designed to apply VL-PTM in few-shot text classification. To make use of the visual understanding ability of VL-PTM, we design a visual prompt generation module based on model inversion (see Figure 2), which can obtain sound visual representations of the categories offline as visual prompts. In the classification process, we still adopt the standard prompt-tuning pipeline, using the text encoder of VL-PTM as backbone. Specifically, we append learnable soft prompt to the front of category names' text embeddings. We then add visual prompts to the obtained embeddings of the corresponding categories and conduct the text classification by computing the cosine similarity scores between the input embeddings and the summed (both visual and textual) category embeddings. In addition, to make extensive evaluation, we collect five public available datasets for Chinese text classification, which cover a diverse set of data domains including news, emotions, types of app, and specialized subjects.

Our main contributions are three-fold: (1) To the best of our knowledge, this is the first work on introducing VL-PTM into few-shot NLP tasks. Particularly, the importance of VL-PTM has been successfully shown in few-shot text classification. (2) We devise a novel VPT method for VL-PTM, which can utilize the visual information to boost the text classification performance. (3) Extensive experiments are conducted on five benchmark datasets to show that our proposed VPT outperforms the state-of-the-art approaches.

2 Related Work

2.1 Vision-Language Pre-Training Models

We first deliver an overview of Vision-Language Pre-Training Models (VL-PTMs). Note that existing VL-PTMs can be broadly divided into two groups according to their network architectures: single-tower models (Su et al., 2020; Li et al., 2020) and two-tower ones (Radford et al., 2021; Jia et al., 2021; Yuan et al., 2021).

Single-tower models appear as the pioneers of VL-PTMs, using a joint network (mostly multi-layer Transformers) to encode the image and text pair. (Su et al., 2020) employs BERT-like objectives to learn cross-modal representations from a concatenated sequence of visual region features and language token embeddings. (Li et al., 2020) makes use of object tags as anchor points for aligning elements in two modalities. This method is motivated by the observation that the salient objects in an image can be accurately detected, and are often mentioned in the paired text. Single-tower models have strong ability to fuse visual and linguistic information. However, they still have a lot of limitations due to the model structure, such as limited understanding ability of high-level semantics, long inference time, etc.

As the successors of VL-PTMs, two-tower models demonstrate greater potential in cross-modal pre-training. They adopt separate image and text encoders, typically taking image-text retrieval as the pre-training task. (Radford et al., 2021; Jia et al., 2021) introduces contrastive learning with SimCLR-based loss for visual-language pre-training. The training goal is to learn powerful encoders that can embed image and paired text samples into the same latent space for effective image-text retrieval. With acceptable inference time and ideal comprehension skill in both cross-modal and single-modal tasks, two-tower models are deployed in various of application scenarios. To expand the learned representations to more visual tasks, (Yuan et al., 2021) concurrently uses

self-attention and cross-attention in their network, enhancing the understanding ability in both single-modal and cross-modal tasks.

In this work, we devise our VPT based on the latest BriVL (Fei et al., 2022), which is a two-tower large-scale Chinese VL-PTM (see Sec. 3.1 for more details). Theoretically, our VPT can be extended to other VL-PTMs in the same way.

2.2 Prompt-Tuning of VL-PTMs

With the recent rapid development of VL-PTMs, there is a growing interest in prompt-tuning with VL-PTMs for various downstream tasks. As a representative model of VL-PTMs, CLIP (Radford et al., 2021) employs prompt template like "A photo of a {label}." in image classification task without further training, which shows the competitive performance against linear probe on ResNet50 (He et al., 2019) (a fully supervised baseline). This success declares the potential of the combination of prompt-tuning and VL-PTM. To alleviate the instability and manpower cost of manual hard prompt, (Zhou et al., 2021) introduces learnable prompt for few-shot image classification. In addition to automated prompt engineering, (Gao et al., 2021a) proposes to insert lightweight learnable module named adapter into VL-PTM, which is a simpler alternative than soft prompt.

Beside image classification, recent works have applied visual-language pre-training to more downstream tasks with prompt-tuning. For example, (Yao et al., 2021) reformulates the visual grounding task into a fill-in-the-blank problem. This recent work creatively uses the RGB value of different colors to build the CLIP-like image sub-prompt. (Tsimpoukelli et al., 2021) designs a unified framework for multi-modal conditional text generation. The proposed pipeline is compatible with seven cross-modal tasks including Referring Expression Comprehension (REC) and Visual Commonsense Reasoning (VCR).

Note that VL-PTMs are generally deployed for visual or cross-modal tasks in the previous works mentioned above. Differently, our proposed VPT extends the application scenarios of VL-PTMs, and forms the first prompt-tuning method that induces VL-PTM into few-shot text classification.

2.3 Prompt-Tuning Methods in NLP Tasks

"Pre-train, prompt, and predict" paradigm is a sea change in NLP (Liu et al., 2021a). Instead of adapting PLMs to downstream tasks through objective engineering, the downstream problems are reformed with the use of a textual prompt to seem more like those solved during the original PLM training. For instance, (Schick and Schütze, 2021) maps each class into a masked token and inserts it into cloze-style phrases, then predicting it using the pre-trained masked language model. This method ensembles multiple models trained with several manual prompts. To get better prompt templates, (Gao et al., 2021b) adopts a T5 model to automatically generate prompts in cloze-style. Then another PLM like RoBERTa is deployed to conduct the label name generation process. (Liu et al., 2021b) introduces trainable continuous prompt embeddings as a better choice than manual prompts, which significantly improves the understanding ability of generative PLMs like GPT. Unlike cloze question-based methods, (Devlin et al., 2019) reformulates the classification task into textual entailment task. This setting can be used as a unified approach to modelling different kinds of classification tasks. Different from the above-mentioned PLM based methods, we introduce VL-PTM to few-shot text classification and propose a novel Visual Prompt Tuning (VPT) method, which blazes a new trail for few-shot NLP tasks.

3 Methodology

In this section, we give the details of the proposed VPT. The overall architecture is shown in Figure 2. Our main idea is to improve the performance of text classification by utilizing visual information in help of VL-PTM. Specifically, VPT deploys model inversion of VL-PTM to generate visual representations of category names and then add them to the text embeddings of the corresponding category names. We first describe the overall framework of VPT in Sec. 3.1, followed by a detailed description of visual prompt generation in Sec. 3.2. Finally, the training objective is presented in Sec. 3.3.

3.1 Overall Framework

We employ BriVL as the backbone of VPT. To learn better cross-modal representations, the representative contrastive learning algorithm MoCo (He et al., 2020) is adopted in the pre-training stage of BriVL. The text encoder of BriVL consists of RoBERTa-large (Cui et al., 2020) and a successive self-attention block. The self-attention

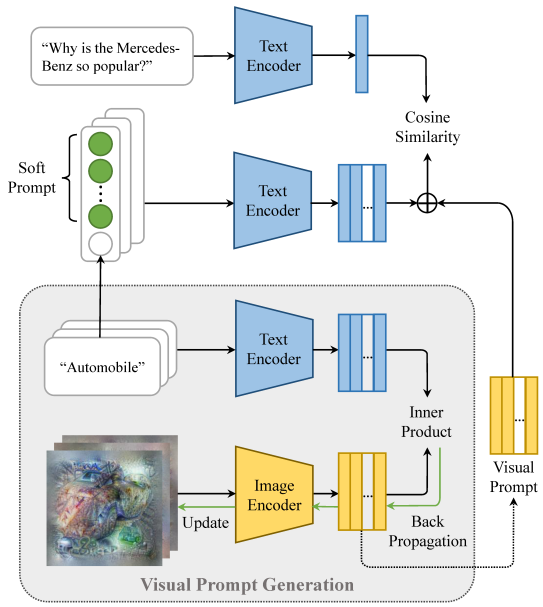

Figure 2: A schematic illustration of the proposed VPT model for few-shot text classification. The bottom panel presents the visual prompt generation module, and the top panel presents the prompt-tuning module for few-shot text classification.

block with four layers of Transformers is designed for keeping RoBERTa from catastrophic forgetting. To perform few-shot text classification with BriVL, we adopt similar paradigm used in few-shot image classification. In particular, given a text classification dataset with M categories, we have natural language expressions $\{C_1, \cdots, C_M\}$ for all categories. For the m-th class C_m, we append learnable soft prompts to the front of its word embedding. That is, the total input embedding sequence $\mathbf{t}_m$ for C_m is designed as:

$$\mathbf{t}_m = [CLS][V]_1 \cdots [V]_N[CLASS]_m[SEP], \tag{1}$$

where $[V]_n$ $(n = 1, \cdots, N)$ denotes a learnable vector with the same dimension as the word embedding of BriVL, N is the hyperparameter specifying the number of learnable tokens, $[CLASS]_m$ is the word embedding of the m-th class name C_m, and $[CLS]$ (or $[SEP]$) is the word embedding of the special token CLS (or SEP). Note that we adopt class-specific soft prompts and use word embeddings of sampled tokens from the vocabulary as the initialization of $[V]_n$.

With the text encoder of BriVL fixed during the training stage, our training goal is to optimize the soft prompts. We first get the visual prompts of all categories offline from the visual prompt generation module. We then encode the tokenized input sequence $\mathbf{x}_i$ and class prompt $\mathbf{t}_m$ into text embeddings $\mathbf{r}_i$ and $\mathbf{r}_m^C$ via the text encoder of BriVL, respectively. For cross-modal information fusion, we thus adopt a simple operation-based method: adding visual prompt to its corresponding class name embedding $\mathbf{r}_m^C$ with the weight of α. Classification is finally conducted by computing cosine similarity scores between the input embedding $\mathbf{r}_i$ and fused class embeddings.

Algorithm 1 Pseudocode of Visual Prompt Generation in a PyTorch-like style.

```
# text_list: list of all class names
# Shape: shape of pseudo image ([C, H, W])
# VP: list of visual prompts corresponding
    to all class names
# f_image, f_text: image encoder and text
    encoder of the adopted VL-PTM

VP = []
for text in text_list:
  pseudo_image = random_tensor(*Shape)
  pseudo_image.requires_grad_(True)
  for i in range(max_iteration):
    imageFea = f_image.forward(pseudo_image)
    textFea = f_text.forward(text)

    # Eqn.(2)
    loss = -mm(imageFea, textFea.t()).mean()

    # Adam update: pseudo_image
    loss.backward()
    update(pseudo_image.params)
  VP.append(imageFea)
save(VP)
```

Notations: mm – matrix multiplication.

3.2 Visual Prompt Generation

The bottom panel of Figure 2 illustrates the pipeline of our visual prompt generation. Given a text classification task, we choose to generate a series of pseudo images according to the class names. In this work, we take the embeddings of pseudo images as the visual representations of the class names, namely visual prompts.

For each class name C_m, we can obtain its text embedding $\mathbf{r}_m^C$ through the text encoder of BriVL. Then we randomly initialize a noisy image and also compute its image embedding $\mathbf{r}_m^I$ through the image encoder of BriVL. Since there should be a one-to-one correspondence between pseudo images and class names, we compute the inner product similarity score between the image embedding and the class embedding and maximize it for model inversion. The loss function for the m-th class can be written as follows:

$$\mathcal{L}_{inversion} = - < \mathbf{r}_m^C, \mathbf{r}_m^I >, \tag{2}$$

where $< \cdot, \cdot >$ is the dot product of two vectors.

Algorithm 2 Pseudocode of VPT in a PyTorch-like style.

```
# VP: output from visual prompt generation
# prompt_tokens: N tokens from vocabulary
# alpha: weight of visual prompt
# f_text: text encoder of the adopted VL-PTM
# f_text.emb: word embedding layer of f_text
# class_names: tokenized class names in the
    following form: [CLS][CLASS][SEP]

load(VP)
C = f_text.emb(class_names)
# initialize soft prompt
soft_prompt = f_text.emb(prompt_tokens)
soft_prompt.requires_grad_(True)
for input_text in loader: # load a minibatch
  # Eqn.(1)
  t = cat([C[:,0,:], soft_prompt, C[:,1:,:])

  inputFea = f_text.forward(input_text)
  labelFea = f_text.forward(input_embs=t)
  labelFea += alpha * VP.ToTensor() # Eqn.(3)

  logits = bmm(inputFea, labelFea)
  # Eqn.(5)
  loss = CrossEntropyLoss(logits, labels)

  # Adam update: soft prompt
  loss.backward()
  update(soft_prompt.params)
```

Notations: `bmm` – batch matrix multiplication; `cat` – concatenation.

Because both encoders of BriVL are frozen during pseudo image generation, only the noisy image is set to be learnable, i.e., it can be updated through back propagation. After a number of iterations, we obtain the pseudo image that depicts a picture of what BriVL knows about the category.

We take the image embedding $\bar{\mathbf{r}}_m^C$ of the pseudo image as our "visual prompt", which can supplement the insufficient information in $\mathbf{r}_m^C$. Because the generation process is done offline, there is no extra time for classification, ensuring VPT's efficiency. The details of the visual prompt generation pipeline are presented in Algorithm 1.

3.3 Training Objective

In this subsection, we describe our training objective and explain the role of visual prompt. The similarity between the i-th input text $\mathbf{x}_i$ and the m-th class name C_m is calculated as follows:

$$s_{im} = < \mathbf{r}_i, \mathbf{r}_m^C + \alpha \bar{\mathbf{r}}_m^C >, \tag{3}$$

where $\mathbf{r}_i$ and $\mathbf{r}_m^C$ are respectively the text embedding of $\mathbf{x}_i$ and class prompt $\mathbf{t}_m$, $\bar{\mathbf{r}}_m^C$ is the visual prompt of C_m, and α is the weight of visual prompt. A softmax function is then used to define the probability value:

$$P(y_i = m|\mathbf{x}_i) = \frac{\exp(s_{im}/\tau)}{\sum_{j=1}^{M} \exp(s_{ij}/\tau)}, \tag{4}$$

where $P(y_i = m|\mathbf{x}_i)$ means the chance of the i-th input text x_i belonging to the m-th class (y_i is the predicted label), and τ is the temperature. Given k shots per class, model training is performed by minimizing the cross-entropy loss:

$$\mathcal{L}_C = \frac{-1}{k * M} \sum_i \sum_m^M y_{im} \log P(y_i = m|\mathbf{x}_i), \tag{5}$$

where $y_{im} = 1$ if $y_i = m$, otherwise $y_{im} = 0$.

Note that Equation (3) indicates the core idea of our proposed VPT for few-shot text classification. That is, each input sentence is forced to be matched with not only textual but also visual semantics of class names. From this perspective, visual prompts act as augmentations of text embeddings of class names. They provide extra information when searching the nearest class name in the latent space for the input sentence. The full VPT algorithm for few-shot text classification is outlined in Algorithm 2.

Table 1: Statistics of five text classification datasets.

Dataset	Classes	Train	Val	Test
THUCNews	14	661,785	83,000	83,000
Toutiao News	15	306,688	38,000	38,000
Inews	3	3,356	1,000	1,000
Iflytex	119	6,935	2,599	2,599
CSLDCP	67	536	536	1,784

4 Experiments

4.1 Datasets

We collect five public available datasets for text classification in Chinese: THUCNews (Li et al., 2006), Toutiao News[1], Inews[2], Iflytex (Xu et al., 2020) and CSLDCP (Xu et al., 2021). Diverse textual tasks are covered, including classification on news (THUCNews, Toutiao News), emotions (Inews), types of app (Iflytex), and specialized subjects (CSLDCP). We follow the original dataset split from public benchmarks (Inews, CSLDCP) and randomly split the others (THUCNews, Toutiao News) with the train/validation/test ratio 8:1:1. Particularly, for THUCNews, we only use titles of the news in our experiments. Since the test set of Iflytex is not labeled, we use the public validation set as the test set, and split the public training set into the training and validation sets. The details of datasets are shown in Table 1.

[1]https://github.com/aceimnorstuvwxz/toutiao-text-classfication-dataset

[2]https://github.com/ChineseGLUE/ChineseGLUE

Table 2: Comparative results for few-shot text classification on five public datasets. We report the mean (and standard deviation) performance over 5 repeated trials. The best performance and the second best performance are denoted in bold and underlined fonts, separately.

Method	THUCNews	Toutiao News	Inews	Iflytex	CSLDCP
Soft Prompt	64.70 (3.64)	71.98 (1.15)	51.76 (1.90)	28.92 (1.67)	37.24 (1.40)
PET	66.33 (1.70)	75.75 (3.31)	62.10 (0.96)	<u>33.49</u> (2.44)	41.87 (0.95)
LM-BFF	71.56 (0.99)	<u>76.67</u> (1.24)	63.72 (2.25)	29.70 (1.68)	38.23 (2.71)
EFL	70.17 (2.12)	71.03 (2.89)	60.20 (5.83)	22.80 (5.06)	42.80 (1.45)
P-tuning	<u>73.46</u> (2.29)	76.56 (1.02)	<u>65.96</u> (2.18)	32.36 (2.63)	<u>44.03</u> (1.59)
VPT (ours)	**74.73** (0.90)	**79.24** (1.44)	**67.20** (2.85)	**34.24** (1.35)	**47.03** (0.84)

In our few-shot experiments, we set $k = 16$ shots per class for THUCNews, Toutiao News, and Inews, but only $k = 1$ shot per class for Iflytex and CSLDCP (given their large number of classes).

4.2 Implementation Details

For the visual prompt generation process, the total iteration number is set to 2,000. The noisy image is optimized by Adam with a learning rate of 0.02. The size of generated pseudo image is set to 600*600, and the dimension of VP is 2,560 according to the image encoder of BriVL.

For the classification process, the weight of visual prompt is set as $\alpha = 1$, and the prompt length is set as $N = [15, 20, 40, 10, 5]$ for THUCNews, Toutiao News, Inews, Iflytex, and CSLDCP, respectively. We optimize the soft prompts using Adam with the learning rate 1e-5.

4.3 Evaluation and Training Protocol

It is commonly accepted that fine-tuning on small datasets can suffer from instability and results may change dramatically given a new split of data. To obtain a robust measure of the model performance, we follow the setting of existing works (Gao et al., 2021b; Devlin et al., 2019). Concretely, on each dataset, we randomly sample $k * M$ labeled training samples from the training set and $k*M$ labeled validation samples from the validation set for few-shot fine-tuning, which is repeated five times. The average performance across five repeated trials is reported. Note that the number of validation samples is set the same as the number of training samples during few-shot fine-tuning on each dataset. Although few-shot fine-tuning using a larger set of validation samples leads to significant improvements (Gao et al., 2021b), its initial goal of learning from limited data is subverted.

4.4 Main Results

We compare our proposed VPT with a series of few-shot learning methods based on PLMs, including PET (Schick and Schütze, 2021), LM-BFF (Gao et al., 2021b), P-tuning (Liu et al., 2021b) and EFL (Devlin et al., 2019). The widely-used RoBERTa-large is adopted as the backbone for these methods. In addition, we apply our framework without visual prompt generation to the original RoBERTa-large as a baseline, denoted as "soft prompt". The comparative results are shown in Table 2. We compare methods based on VL-PTM and PLM, with two backbones (text encoders) of similar sizes: BriVL and RoBERTa-large, respectively. Note that the four PLM-based baselines typically adopt the pre-trained masked language modeling (MLM) head for prompt learning and thus we cannot apply BriVL (without the MLM head) as their backbone. Moreover, since the text embedding of RoBERTa-large is not aligned with the visual prompt, it is unreasonable to apply RoBERTa-large as the backbone to our proposed VPT model.

We can clearly observe from Table 2 that our proposed VPT consistently outperforms the recent state-of-the-art methods for few-shot text classification on all five datasets. Particularly, our proposed VPT yields more than 2.5% improvements over the second best on Toutiao News and CSLDCP. These observations indeed identify the important role of our proposed VPT as a better approach to few-shot text classification. Although two different text backbones (i.e., BriVL and RoBERTa-large) of similar sizes have been employed, these observations are still remarkable since the pre-training data of BriVL does not bring benefits as expected (see Table 3). Moreover, PLM-based prompt methods demonstrate unstable performance across text classification datasets

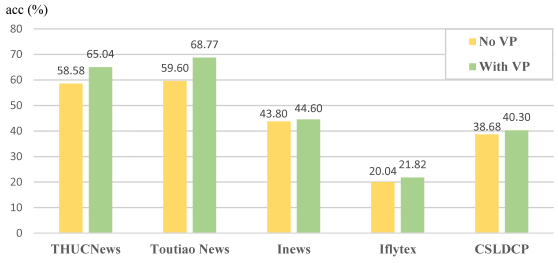

(a) 'Zero-Shot'

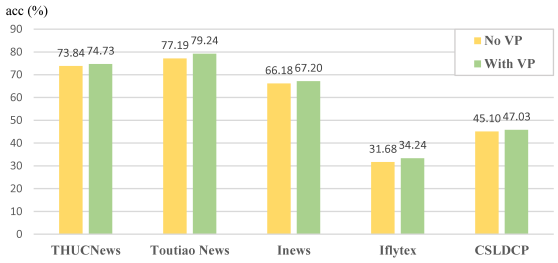

(b) 'Few-Shot'

Figure 3: Ablation study results of the proposed VPT with the large-scale pre-training model BriVL as the backbone. Average accuracy (%) on 5 repeated trials is reported. VP stands for visual prompt.

with different data distributions, while our proposed VPT demonstrates great robustness in few-shot text classification.

4.5 Ablation Study Results

We conduct ablation studies to show the contribution of the visual prompt. We run experiments with and without visual prompt in both zero-shot and few-shot scenarios, using the large-scale pre-training model BriVL as the backbone. The ablation results are shown in Figure 3. We have two main observations. Firstly, after adding visual prompt into few-shot prompt-tuning (see the comparison VPT vs. VPT w/o VP in Figure 3(b)), the few-shot performance increases by 1.7% in average, as compared with that using standard prompt-tuning alone (i.e., VPT w/o VP). Secondly, by directly adding visual prompt into zero-shot classification, the zero-shot performance of BriVL can be improved by 4.0% in average (see the comparison Zero-shot w/ VP vs. Zero-shot w/o VP in Figure 3(a)). These evidences clearly show that the visual prompt is indeed beneficial for deploying BriVL in few-shot text classification.

Note that the usage of visual prompt at test time is clearly shown in the experiments in Figure 3(a). Our visual prompt brings considerable boost in zero-shot scenario, and only the inference process is included. That is, we conduct the classification by directly computing the cosine similarity scores between the embeddings of input sentences and the embeddings of category names. After adding visual prompts to the embeddings of category names, each input sentence is forced to be matched with not only textual but also visual semantics of class names. Therefore, visual prompts serve as augmentations of text embeddings of class names at test time.

Table 3: Results obtained by base models using different pre-training data. The average accuracy (%) over all five datasets is reported for each model.

Model	Zero-shot	Soft Prompt
RoBERTa-base	30.39	51.48
RoBERTa-base (finetune)	28.88	51.92
BriVL w/ RoBERTa-base	**38.85**	**55.45**

Furthermore, we notice that the pre-training data of BriVL is different from that of RoBERTa, which may cause a bit of unfairness in the comparison of our experiments in Table 2. Therefore, we make comparison among the following three models: (1) RoBERTa-base; (2) RoBERTa-base (finetune): we finetune the pre-trained RoBERTa-base on the text data of 22 million image-text pairs, which is the same pre-training dataset of BriVL w/ RoBERTa-base; (3) BriVL w/ RoBERTa-base: it is a smaller version (using RoBERTa-base as backbone instead) of the standard BriVL, which is pre-trained with the aforementioned 22 million image-text pairs. Due to the limited GPU resource, only the base models are considered here. The ablation results in Table 3 show that RoBERTa-base (finetune) yields only slight improvements (or even performance drops) over RoBERTa-base, while BriVL w/ RoBERTa-base outperforms RoBERTa-base by large margins. This is mainly due to that the text data from large-scale image-text pairs has not been filtered (without any reprocessing), and PLMs like RoBERTa can hardly benefit from this noisy data. Therefore, the observations/conclusions from Table 2 can still be drawn, even if RoBERTa-large is first finetuned for the baselines with the pre-training data of BriVL.

Finally, it would be easier to demonstrate the advantages of our VPT by examining the impact of visual prompt on accuracy when alternative image representations are used for generating visual prompt. Concretely, we consider three im-

Table 4: Results obtained by using different image representations for generating visual prompt. Results of few-shot text classification are reported only on Iflytex.

Image Representation	Accuracy
Random Noise	7.81
Visual Prompt (1K iterations)	30.94
Visual Prompt (2K iterations)	**34.24**

age representations for generating visual prompt: (1) Random Noise: visual prompt initialized by random noise (without optimization); (2) Visual Prompt (1K iterations): low-quality visual prompt obtained only with 1K iterations of optimization; (3) Visual Prompt (2K iterations): standard visual prompt obtained with 2K iterations of optimization. Note that the results of few-shot text classification are reported only on the small dataset Iflytex for quick evaluation. Two observations can be drawn from the ablation results in Table 4. Firstly, the visual prompt initialized by random noise (without optimization) causes serious damage to the performance of our VPT for few-shot text classification. Secondly, the visual prompt obtained with 2K iterations of optimization leads to 3.3% improvements over that obtained with 1K iterations of optimization, showing that the visual prompt of higher quality brings larger benefits to our VPT for few-shot text classification.

4.6 Influence of Hyperparameters

In this subsection, we discuss the influence of two crucial hyperparameters on the performance of VPT: prompt length – N, and weight of VP – α. The detailed results are provided in Figure 4. Only two small datasets Iflytex and CSLDCP are considered for quick evaluation.

We first conduct experiments by varying the prompt length in {1; 5; 20; 40; 60; 80; 100}, while fixing the rest hyperparameters. Figure 4(a) and Figure 4(c) show that increasing prompt length is beneficial for better performance when the prompt length is modest. When the prompt length is further increased (e.g., more than 5), the performance tends to gradually deteriorate. Notably, VPT has a strong performance even with a single prompt token. This indicates that VPT inherits the advantages of high efficiency of prompt-tuning. Increasing the token count above 20 leads to marginal gains (or even drops). Going above 60 tokens appears to be consistently damaging on

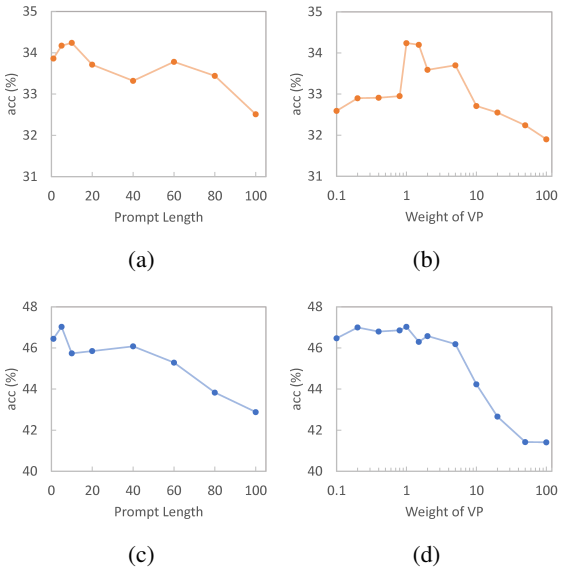

Figure 4: Effect of hyperparameters on the performance of VPT. Results on two small datasets (Iflytex, CSLDCP) are reported. The orange line refers to Iflytex ((a), (b)) and the blue one refers to CSLDCP ((c), (d)). (Left) On prompt length: employing more tokens for VPT leads to improvements when the number of prompt tokens is small. (Right) On weight of VP: VPT performs the best on both datasets when $\alpha = 1$.

both datasets. (Lester et al., 2021) discovered a similar pattern of declining performance beyond a particular prompt length. In addition, in Figure 4(a) and Figure 4(c), the minimum prompt length is 1. If we set the prompt length to 0 (i.e., the soft prompt is not used), the performance drops significantly. This means that the soft prompt does lead to significant improvements in text classification.

Further, the influence of weight of VP is explored in the range from 0.1 to 100, while the other hyperparameters are fixed. As shown in Figure 4(b) and Figure 4(d), increasing the weight α yields performance improvements when $\alpha < 1$ and causes negative effects when $\alpha > 1$. Notably, when $\alpha >> 1$, the performance of VPT still outperforms the baselines like Soft Prompt. This observation demonstrates that the generated VP indeed contains rich semantic information inherited from the class name. However, VP can only play an auxiliary role, i.e., it cannot take the place of the textual class name.

4.7 Visualization Results

To directly figure out the effect of visual prompt, we visualize the obtained pseudo images in the visual prompt generation process in Figure 5. Note

(a) "automobile" (b) "furniture" (c) "camera" (d) "machinery" (e) "gardening"

(f) "culture" (g) "art" (h) "military" (i) "technology" (j) "positive emotion"

Figure 5: Visualizations of generated images for different class names.

that the input texts of class names are originally in Chinese and translated into English for illustration purpose. For text descriptions with concrete meanings, the generated visualizations provide intuitive pictures (e.g., "automobile": a car; "furniture": table, chair and vase; "camera": a telecamera; "machinery": some gears and pipes; "gardening": morning glories). For text descriptions with abstract meanings, the generated visualizations are able to show concrete embodiment of these concepts (e.g., "culture": a figure in traditional Chinese clothing with Chinese "culture" in the upper right corner; "art": geometric figures and abstract portraits; "military": soldiers, a military aircraft and an armored vehicle; "technology": monitors and consoles; "positive emotion": a smiling face in the bottom left). Overall, these visualization results clearly demonstrate that our visual prompts have actually be learned to well represent the semantic content of the corresponding class names.

5 Conclusion

We propose a novel prompt-based method termed Visual Prompt Tuning (VPT) for deploying VL-PTM like BriVL in few-shot text classification. The main component of our proposed VPT is a visual prompt generation module based on model inversion of VL-PTM. Extensive experimental results on five benchmark datasets demonstrate that our proposed VPT achieves the new state-of-the-art in few-shot text classification. The ablation study and visualization results further show the effectiveness of our proposed VPT. In our ongoing research, we will apply our proposed VPT to other few-shot NLP tasks.

Acknowledgments

We would like to thank the anonymous reviewers for their helpful comments. This work was supported in part by National Natural Science Foundation of China (61976220 and 61832017), and Beijing Outstanding Young Scientist Program (BJJWZYJH012019100020098).

References

Tom B Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. 2020. Language models are few-shot learners. *arXiv preprint arXiv:2005.14165*.

Ying-Hong Chan and Yao-Chung Fan. 2019. A recurrent bert-based model for question generation. In *Proceedings of the 2nd Workshop on Machine Reading for Question Answering*, pages 154–162.

Yiming Cui, Wanxiang Che, Ting Liu, Bing Qin, Shijin Wang, and Guoping Hu. 2020. Revisiting pre-trained models for Chinese natural language processing. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 657–668.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for*

Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers), pages 4171–4186.

Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, Jakob Uszkoreit, and Neil Houlsby. 2021. An image is worth 16x16 words: Transformers for image recognition at scale. In *Proceedings of International Conference on Learning Representations (ICLR)*.

Nanyi Fei, Zhiwu Lu, Yizhao Gao, Guoxing Yang, Yuqi Huo, Jingyuan Wen, Haoyu Lu, Ruihua Song, Xin Gao, Tao Xiang, et al. 2022. Towards artificial general intelligence via a multimodal foundation model. *Nature Communications*, 13:3094.

Peng Gao, Shijie Geng, Renrui Zhang, Teli Ma, Rongyao Fang, Yongfeng Zhang, Hongsheng Li, and Yu Qiao. 2021a. Clip-adapter: Better vision-language models with feature adapters. *arXiv preprint arXiv:2110.04544*.

Tianyu Gao, Adam Fisch, and Danqi Chen. 2021b. Making pre-trained language models better few-shot learners. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 3816–3830.

Kaiming He, Haoqi Fan, Yuxin Wu, Saining Xie, and Ross Girshick. 2020. Momentum contrast for unsupervised visual representation learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 9729–9738.

Tong He, Zhi Zhang, Hang Zhang, Zhongyue Zhang, Junyuan Xie, and Mu Li. 2019. Bag of tricks for image classification with convolutional neural networks. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 558–567.

Yuqi Huo, Manli Zhang, Guangzhen Liu, Haoyu Lu, Yizhao Gao, Guoxing Yang, Jingyuan Wen, Heng Zhang, Baogui Xu, Weihao Zheng, et al. 2021. Wenlan: Bridging vision and language by large-scale multi-modal pre-training. *arXiv preprint arXiv:2103.06561*.

Chao Jia, Yinfei Yang, Ye Xia, Yi-Ting Chen, Zarana Parekh, Hieu Pham, Quoc V. Le, Yun-Hsuan Sung, Zhen Li, and Tom Duerig. 2021. Scaling up visual and vision-language representation learning with noisy text supervision. In *Proceedings of the 38th International Conference on Machine Learning (ICML)*, pages 4904–4916. PMLR.

Chen Jia, Yuefeng Shi, Qinrong Yang, and Yue Zhang. 2020. Entity enhanced bert pre-training for chinese ner. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 6384–6396.

Alexander Kolesnikov, Lucas Beyer, Xiaohua Zhai, Joan Puigcerver, Jessica Yung, Sylvain Gelly, and Neil Houlsby. 2020. Big transfer (bit): General visual representation learning. In *Proceedings of European Conference on Computer Vision (ECCV)*, pages 491–507.

Brian Lester, Rami Al-Rfou, and Noah Constant. 2021. The power of scale for parameter-efficient prompt tuning. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 3045–3059.

Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Veselin Stoyanov, and Luke Zettlemoyer. 2020. BART: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7871–7880.

Jingyang Li, Maosong Sun, and Xian Zhang. 2006. A comparison and semi-quantitative analysis of words and character-bigrams as features in chinese text categorization. In *Proceedings of the 21st International Conference on Computational Linguistics and 44th Annual Meeting of the Association for Computational Linguistics*, pages 545–552.

Xiang Lisa Li and Percy Liang. 2021. Prefix-tuning: Optimizing continuous prompts for generation. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 4582–4597.

Xiujun Li, Xi Yin, Chunyuan Li, Pengchuan Zhang, Xiaowei Hu, Lei Zhang, Lijuan Wang, Houdong Hu, Li Dong, Furu Wei, et al. 2020. Oscar: Object-semantics aligned pre-training for vision-language tasks. In *Proceedings of European Conference on Computer Vision (ECCV)*, pages 121–137.

Junyang Lin, Rui Men, An Yang, Chang Zhou, Ming Ding, Yichang Zhang, Peng Wang, Ang Wang, Le Jiang, Xianyan Jia, et al. 2021. M6: A chinese multimodal pretrainer. *arXiv preprint arXiv:2103.00823*.

Pengfei Liu, Weizhe Yuan, Jinlan Fu, Zhengbao Jiang, Hiroaki Hayashi, and Graham Neubig. 2021a. Pre-train, prompt, and predict: A systematic survey of prompting methods in natural language processing. *arXiv preprint arXiv:2107.13586*.

Xiao Liu, Yanan Zheng, Zhengxiao Du, Ming Ding, Yujie Qian, Zhilin Yang, and Jie Tang. 2021b. Gpt understands, too. *arXiv preprint arXiv:2103.10385*.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Roberta: A robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692*.

Haoyu Lu, Nanyi Fei, Yuqi Huo, Yizhao Gao, Zhiwu Lu, and Ji-Rong Wen. 2022. COTS: Collaborative two-stream vision-language pre-training model for cross-modal retrieval. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 15692–15701.

Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, Gretchen Krueger, and Ilya Sutskever. 2021. Learning transferable visual models from natural language supervision. In *Proceedings of the 38th International Conference on Machine Learning (ICML)*, pages 8748–8763. PMLR.

Alec Radford, Karthik Narasimhan, Tim Salimans, and Ilya Sutskever. 2018. Improving language understanding by generative pre-training. *OpenAI Blog*.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J. Liu. 2020. Exploring the limits of transfer learning with a unified text-to-text transformer. *Journal of Machine Learning Research (JMLR)*, pages 140:1–140:67.

Timo Schick and Hinrich Schütze. 2021. Exploiting cloze-questions for few-shot text classification and natural language inference. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*, pages 255–269.

Weijie Su, Xizhou Zhu, Yue Cao, Bin Li, Lewei Lu, Furu Wei, and Jifeng Dai. 2020. VL-BERT: pre-training of generic visual-linguistic representations. In *Proceedings of International Conference on Learning Representations (ICLR)*.

Derek Tam, Rakesh R Menon, Mohit Bansal, Shashank Srivastava, and Colin Raffel. 2021. Improving and simplifying pattern exploiting training. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 4980–4991.

Maria Tsimpoukelli, Jacob L Menick, Serkan Cabi, SM Eslami, Oriol Vinyals, and Felix Hill. 2021. Multimodal few-shot learning with frozen language models. *Advances in Neural Information Processing Systems*, 34:200–212.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. *Advances in Neural Information Processing Systems*, 30.

Liang Xu, Hai Hu, Xuanwei Zhang, Lu Li, Chenjie Cao, Yudong Li, Yechen Xu, Kai Sun, Dian Yu, Cong Yu, et al. 2020. Clue: A chinese language understanding evaluation benchmark. *arXiv preprint arXiv:2004.05986*.

Liang Xu, Xiaojing Lu, Chenyang Yuan, Xuanwei Zhang, Huilin Xu, Hu Yuan, Guoao Wei, Xiang Pan, Xin Tian, Libo Qin, et al. 2021. Fewclue: A chinese few-shot learning evaluation benchmark. *arXiv preprint arXiv:2107.07498*.

Zhilin Yang, Zihang Dai, Yiming Yang, Jaime Carbonell, Russ R Salakhutdinov, and Quoc V Le. 2019. Xlnet: Generalized autoregressive pretraining for language understanding. *Advances in Neural Information Processing Systems*, 32.

Yuan Yao, Ao Zhang, Zhengyan Zhang, Zhiyuan Liu, Tat-Seng Chua, and Maosong Sun. 2021. Cpt: Colorful prompt tuning for pre-trained vision-language models. *arXiv preprint arXiv:2109.11797*.

Lu Yuan, Dongdong Chen, Yi-Ling Chen, Noel Codella, Xiyang Dai, Jianfeng Gao, Houdong Hu, Xuedong Huang, Boxin Li, Chunyuan Li, et al. 2021. Florence: A new foundation model for computer vision. *arXiv preprint arXiv:2111.11432*.

Kaiyang Zhou, Jingkang Yang, Chen Change Loy, and Ziwei Liu. 2021. Learning to prompt for vision-language models. *arXiv preprint arXiv:2109.01134*.

Systematic Analysis of Image Schemas in Natural Language through Explainable Multilingual Neural Language Processing

Lennart Wachowiak
King's College London
lennart.wachowiak@kcl.ac.uk

Dagmar Gromann
University of Vienna
dagmar.gromann@gmail.com

Abstract

In embodied cognition, physical experiences are believed to shape abstract cognition, such as natural language and reasoning. Image schemas were introduced as spatio-temporal cognitive building blocks that capture these recurring sensorimotor experiences. The few existing approaches for automatic detection of image schemas in natural language rely on specific assumptions about word classes as indicators of spatio-temporal events. Furthermore, the lack of sufficiently large, annotated datasets makes evaluation and supervised learning difficult. We propose to build on the recent success of large multilingual pretrained language models and a small dataset of examples from image schema literature to train a supervised classifier that classifies natural language expressions of varying lengths into image schemas. Despite most of the training data being in English with few examples for German, the model performs best in German. Additionally, we analyse the model's zero-shot performance in Russian, French, and Mandarin. To further investigate the model's behaviour, we utilize local linear approximations for prediction probabilities that indicate which words in a sentence the model relies on for its final classification decision. Code and dataset are publicly available[1].

1 Introduction

In the tradition of embodied cognition, image schemas have been proposed by Lakoff (1987) and Johnson (1987) as spatio-temporal cognitive building blocks that capture recurring sensorimotor experiences. For instance, in early infancy we experience many objects with the properties of a CONTAINER, i.e., having an inside and an outside separated by a boundary. The image schema CONTAINMENT captures this experience and is subsequently used to make sense of new experiences while at the same time also influencing how we think and talk about abstract concepts, such as thinking, emotions, or life (see Figure 1).

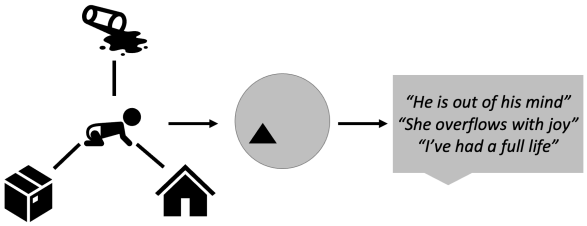

Figure 1: Example of the image schema CONTAINMENT: From experiencing different types of a CONTAINER in early infancy (left); to the development of the schema (middle); and the usage in language on abstract topics (right)

In order to systematically analyse the occurrence of image schemas in natural language, we propose to build on the recent success of multilingual pretrained language models and a small set of examples from image schema literature (Hurtienne, 2017) to train a supervised classifier based on XLM RoBERTa (XLM-R) (Conneau et al., 2020) to classify natural language expressions into image schemas. An image schema detection model as ours could help linguists to explore the use of image schemas efficiently and effectively in large text corpora. It can guide researchers who, for instance, investigate how the use of image schemas differs across languages and cultures (e.g., Choi and Bowerman, 1991; Papafragou et al., 2006), how the language of children with spatial impairments differs (e.g., Lakusta and Landau, 2005) or which image schemas occur in various literary works (e.g., Freeman, 2002). Moreover, we hope that analysing image schemas in large text corpora allows us to contribute to image schema theory directly and to investigate how we think and talk about abstract concepts.

Our proposed method has significant advantages over previously proposed methods. Several corpus linguistic studies (e.g., Dodge and

[1] https://tinyurl.com/24haedv5

Proceedings of the 29th International Conference on Computational Linguistics, pages 5571–5581
October 12–17, 2022.

Lakoff, 2005) and unsupervised machine learning approaches (e.g., Gromann and Hedblom, 2017) for image schema extraction rely on specific parts-of-speech (POS) as indicators of spatio-temporal events. These approaches using POS-tags conventionally portray prepositions as excellent spatial indicators and verbs as movement indicators (e.g., Gromann and Hedblom, 2017; Kordjamshidi et al., 2011). However, spatial language might be expressed with prepositions (*He walked across the room*) or without (*He crossed the room*) (Dodge and Lakoff, 2005). In both examples, the underlying image-schematic structure is that of SOURCE-PATH-GOAL, i.e., the way through the room. Since not all spatial expressions in language rely on prepositions, a more general, word class-independent method is needed, which we propose in form of a supervised training procedure based on a multilingual pretrained language model.

In contrast to these previous methods, we make use of a small annotated image schema corpus that not only allows us to extract image schemas in different languages without relying on manually created patterns, but also provides a gold standard to evaluate our model. Natural language examples of image schemas in literature have been collected in a repository (Hurtienne, 2017). However, this database is rather inconsistent in its formatting and image schema annotation. Thus, we cleaned and complemented it with other examples from MetaNet (Dodge et al., 2015). Our classification method is trained and primarily evaluated in English and German. We also analyse the model's zero-shot performance on a small set of sentences in French, Russian, and Mandarin, representing different language families. To further investigate the model's behaviour, we utilize the explainable artificial intelligence model LIME (Ribeiro et al., 2016) that provides local linear approximations for prediction probabilities for each word in the input expression in relation to each available target class, i.e., image schema. Thereby, we can provide an analysis of which words in the input sequence the model primarily relies on to make its predictions.

2 Related Work

Most previous automated approaches for image schema extraction rely on handwritten rules and pattern matching to annotate natural text with image schemas (e.g., Bennett and Cialone, 2014). However, such rules and patterns have to be specified for each image schema as well as for each language resulting in a substantial manual effort. Moreover, such patterns lead to low recall and have no mechanisms to handle polysemous words. The only existing machine learning approach clusters triples of syntactically dependent nouns, verbs, and prepositions in order to group them by image schema in an unsupervised manner (Gromann and Hedblom, 2017; Wachowiak, 2020). Since this approach relies on assumptions about word classes, especially preposition, the range of expressions that can be considered is limited. Fields with thematically related objectives are metaphor extraction and spatial role labeling, where recent state-of-the-art approaches rely on pretrained neural language models (e.g., Dankers et al., 2020; Leong et al., 2020) and on contextualized embeddings created for trajector, landmark, and preposition candidates (Ramrakhiyani et al., 2019).

3 Foundation

Embodied cognition, a field that builds on the hypothesis that cognitive processes are grounded in perception and sensorimotor interactions with the world, has experienced significant traction in cognitive linguistics. In this tradition, Lakoff (1987) and Johnson (1987) introduce image schemas as cognitive concepts that are firmly rooted in sensorimotor experiences that eventually shape higher-level cognition, including natural language.

3.1 Image Schemas

An image schema according to Johnson (1987, p. xiv) "is a recurring, dynamic pattern of our perceptual interactions and motor programs that gives coherence and structure to our experience." Schema here follows the notion of Langacker (1987) to abstract away from less important details to core commonalities of experiences. Image relates to imagistic in the sense of sensory experiences building on information from different perceptual modalities (Talmy, 2005). They are directly meaningful, preconceptual structures that represent experiential gestalts, i.e., parts that flexibly organize experiences into coherent wholes. Repeated physical experiences starting in early infancy form concepts that manifest themselves in language. For instance, we learn early on that many objects function as CONTAINER, for instance, a glass, a fridge, or a basket, while other objects, such as tables, do not show the same properties. Having learned the im-

Image Schema	Definition	Conceptual Metaphor	Example
CENTER-PERIPHERY	Experience of objects or events as central, while others are peripheral or even outside (Gibbs Jr et al., 1994, p. 237). The periphery depends on the center but not vice versa (Lakoff, 1987, p. 274).	AFFECTION IS PHYSICAL CLOSENESS	He keeps everyone at arms length. (Lakoff et al., 1991, p. 155)
CONTACT	Relates to two entities physically touching without depending on each other (Cienki, 2008, p. 36).	COMMUNICATION IS ESTABLISHED BY PHYSICAL CONTACT	She's in touch with him. (Hurtienne, 2017)
CONTAINMENT	Experience of boundedness, entailing an interior, exterior, and a boundary (Johnson, 1987).	MIND AS CONTAINER FOR IDEAS	Who put that idea in your head? (Jäkel, 2003, pp. 156-157)
FORCE	Implies the exertion of physical strengths in one or more directions (Cienki, 2008, p. 431).	HAPPINESS IS A NATURAL FORCE	He was swept off his feet. (Kövecses, 2010, p. 100)
PART-WHOLE	Wholes consisting of parts and a configuration of parts (Lakoff, 1987, p. 273).	COHERENT IS WHOLE	His thoughts are scattered. (Lakoff et al., 1991, p. 138)
SCALE	Quantitatively it refers to the grouping of discrete objects and substances that can be increased and decreased in amount; qualitatively it refers to the degree of intensity (Johnson, 1987, p. 122).	IMPORTANT IS BIG	Maslow is a towering figure in humanistic psychology. (Tolaas, 1991, p. 207)
SOURCE-PATH-GOAL	Source or starting point, goal or endpoint, a series of contiguous locations connecting both, and movement (Johnson, 1987, p.113).	PURPOSES ARE DESTINATIONS	He finally reached his goals. (Kövecses, 2010, p. 163)
VERTICALITY	A tendency to employ an UP-DOWN orientation (Johnson, 1987, p. xiv).	LIFE IS UP	He's at the peak of health. (Lakoff and Johnson, 1980, p. 15)

Table 1: Image Schema Definitions and Examples

age schema CONTAINMENT, it is later on reflected in our language about physical objects; but also about abstract concepts, for example, in expressions such as *He's gone out of his mind*. The image schemas we consider in this work, selected based on available natural language examples in literature, are defined, related to conceptual metaphors, and exemplified in Table 1.

3.2 Image Schemas and Natural Language

Instead of only pertaining to the physical realm, image schemas are metaphorically projected onto abstract target domains (Lakoff, 1987). In other words, conceptual metaphors map structures learned in the physical source domain, i.e., spatial in the case of image-schematic metaphors, to an abstract target domain. To take up a previous example, the expression *He's gone out of his mind* relates to the conceptual metaphor MIND AS CONTAINER in which the physical properties of CONTAINMENT in the sense of having an inside, outside and a boundary are transferred to the abstract concept of "mind" assigning it similar properties. Thus, image schemas function as structuring devices for language and thought (Kimmel, 2009). Similarities in underlying image-schematic structures across expressions and even across languages can help guide the analysis of language. For instance, the same metaphor and image schema can be observed in the Russian expression ...стереотипах, которые нам вбивались в голову в советское время...[2] (stereotypes that were hammered into our heads during Soviet times). The image schema CONTAINMENT is frequently used to talk about emotions, for example in French, *Je suis cachée au bord des larmes*[3] (I'm hiding on the verge of tears),

[2] In VTimes on 31 October 2020.

[3] Part of lyrics of *anxiété* by Pomme.

German, *...nicht aus der Ruhe bringen* (not be upset; literally: not get out of one's calm) (Baldauf, 1997, p. 135) or Chinese, 他怒火中烧 (Ta nu-huo zhong shao; He has angry fire burning inside him) (Yu, 1995, p. 62).

Linguistic analyses of image schemas have been criticized to suffer from circularity in the sense that language analysis represents a means for forming inferences about the mind, body and their interrelations, the results of which then motivate different arguments on linguistic phenomena (Gibbs and Colston, 1995, pp. 245-246). Natural language might not provide evidence on the origin of image schemas, however, its analysis can foster an understanding of image schema usage in natural languages (Dodge and Lakoff, 2005). This idea is further supported by neuroscientific evidence. For instance, Durand et al. (2018) found that motor areas in the brain are activated when processing action words. Their research focuses on verb anomia, described as difficulty to retrieve words, and showed an added value of combining language and sensorimotor strategies to effectively foster recovery from verb anomia.

3.3 Language Models

Many of the recent successes in natural language processing can be accredited to deep neural language models. Such models learn rich, contextualized language representations during a pretraining stage, in which they learn to predict a masked word given its context, a task for which large amounts of training data are readily available. In a second stage, these models can be finetuned for specific tasks like classification or question answering by adding additional layers on top of the output of the language model, thus, utilizing the previously learned representations. Such a model is then optimized end-to-end, i.e., no additional manually created feature extraction pipeline is needed, but the neural network takes in text as it is and learns by itself to pay attention to the features important for a specific task. One of the most prominent language models is BERT (Devlin et al., 2019), which is based on the now ubiquitous Transformer architecture (Vaswani et al., 2017). Multilingual variants of BERT use multiple languages in the pretraining phase, for instance multilingual BERT and XLM-R (Conneau et al., 2020), which was pretrained on text in 100 different languages and uses an improved training paradigm. Depending on the task, multilingual models show decent zero-shot performances on languages they were originally pretrained on, but that were not part of the training set in the finetuning stage.

Image Schema	EN	DE
CENTER-PERIPHERY	96	40
CONTACT	30	0
CONTAINMENT	451	154
FORCE	273	26
PART-WHOLE	30	0
SCALE	52	10
SOURCE-PATH-GOAL	367	99
VERTICALITY	236	85
Total	1,535	414

Table 2: Sample distribution across languages and image schemas

4 Data

The data combined from the image schema repository (Hurtienne, 2017) and MetaNet (Dodge et al., 2015) consist of a total of 1,949 samples: 1,535 in English and 414 in German. The exact distribution per image schema can be seen in Table 2. The cleaning of the image schema repository consisted in deduplicating and ensuring a consistent processable format and annotation. Additionally, the authors of this paper and Chao Xu for Mandarin manually curated small test datasets of image schematic language in Russian, French, and Mandarin, consisting of 35, 40, and 55 samples respectively, for evaluating the zero-shot performance of the classifier. Sources for the additional language samples consisted of image schema literature, novels, and online news articles.

5 Method

5.1 Supervised Classification Model

We use the English and German data described in Section 4 for finetuning XLM-R in order to classify natural language sequences into image schemas. The model input consists of natural language expressions, which are classified into one of the eight image schemas described in Section 3.1 by adding a fully connected layer on top of XLM-R's output with one output-neuron representing each class. We train the model with 80% of the available data leaving the other 20% for testing. We use a stratified train-test split guaranteeing the same distribution of labels in training and test set. In order to see if the

model achieves consistent results we cross-validate it by training it on five different stratified random splits and report the averaged results for accuracy and F1 scores. All Russian, French, and Mandarin samples are only in the test data and never seen during training. XLM-R exists in different sizes depending on their number of parameters. For our experiments we choose the variant called XLM-R_{Base}. This model is trained for 12 epochs utilizing the Adam optimizer with a learning rate of 3e-5 and a batch size of 16.

5.2 Unsupervised Baseline Classifier

To see how our model compares to other image schema extraction methods, we re-implement a recent approach that clusters instances of spatial language based on the underlying image schema (Gromann and Hedblom, 2017; Wachowiak, 2020). This approach uses the neural dependency parser Stanza (Qi et al., 2020) to find prepositions as markers of spatial language as well as their connected verbs and nouns. Examples of resulting triples are: <*fell, from, power*> or <*stir, in, ingredient*>. In a second step, each word of the triple is represented by their GloVe embedding (Pennington et al., 2014). These embeddings are averaged or summed, resulting in a 300-dimensional vector for each triple. Lastly, similar vectors are grouped using spectral clustering (Ng et al., 2001) based on the implementation made available by scikit-learn (Pedregosa et al., 2011). Since we have a labeled dataset, we simply annotate each cluster with the label that is the most frequent among the contained triples. We, thus, can compute accuracy and F1 score telling us how well the clusters separate different image schemas compared to the novel supervised approach. If the unsupervised method were to be applied to a new and unlabelled dataset, this annotation would have to be made manually. We compute the clusters and their respective scores for different hyper-parameter combinations and report the best resulting score:

- Triple representation: summed vectors, averaged vectors
- Number of clusters: 8, 16
- Affinity matrix construction: nearest neighbors, radial basis function
- Label assignment: k-means, discretization

The data used for clustering consists of all English samples, including both training and test data, as the unsupervised approach does not require any training.

5.3 LIME Explanations

For a detailed analysis of the model's decisions, we use LIME (Ribeiro et al., 2016), which is a method for interpreting machine learning models by approximating local decisions with an interpretable model that assigns weights to the different input features. A local decision refers to a classification of a single input instance, whose features, in our case, are the words that make up the sequence. Such an interpretable model is build for a specific input sample by being trained on perturbations of that sample and the corresponding outputs of the original model. A perturbed text sample, for instance, leaves out one or more words contained in the original sample. The thus generated explanations indicate which words the classifier based its decision on, i.e., which words indicate an image schema. Looking at the explanations of wrong model decisions can show us for which cases the model requires additional training data or which dataset samples are faulty, thus, leading to insights that lie beyond the power of strictly numerical metrics, such as accuracy.

Additionally, we utilize LIME in order to gather global statistics about typical indicators for a specific image schema class. For each sample in the test set we look at the classification made by our model and add the words of the input sequence as well as the corresponding feature weights computed by LIME to a list for this image schema class. After iterating over all test samples, we rank the words for each image schema class by their average feature weight, thus, obtaining a list of words that are strong indicators for a specific image schema according to the model.

6 Results

6.1 Scores Supervised Classifier

The cross-validated results for the test sets in English, German, Russian, French, and Mandarin can be seen in Table 3. The highest scores are achieved in German with an average accuracy of 79.8%, followed by the accuracy in English with 68.6%, Mandarin with 63.2%, Russian with 61.2% and French with 56.6%. The macro F1 score, which gives equal importance to all classes, is consistently lower than

Language	Accuracy	Macro Avg.			Weighted Avg.		
		Precision	Recall	F1	Precision	Recall	F1
English	68.6	0.690	0.606	0.630	0.694	0.686	0.682
German	79.8	0.728	0.736	0.724	0.816	0.798	0.802
Russian	61.2	0.636	0.592	0.574	0.660	0.612	0.598
French	56.6	0.636	0.538	0.518	0.662	0.566	0.542
Mandarin	63.2	0.772	0.632	0.690	0.772	0.632	0.690

Table 3: Cross-validated test results in different languages

Relation Type	Precision	Recall	F1	Test samples
CENTER-PERIPHERY	0.63	0.56	0.59	27
CONTACT	0.75	0.50	0.60	6
CONTAINMENT	0.77	0.82	0.79	121
FORCE	0.60	0.50	0.55	60
PART-WHOLE	0.75	0.50	0.60	6
SCALE	0.71	0.38	0.50	13
SOURCE-PATH-GOAL	0.72	0.80	0.76	93
VERTICALITY	0.78	0.84	0.81	64

Table 4: F1 scores for the individual classes of the test set (English and German)

the accuracy and the weighted F1 score showing that the classes having more training data were learned better. In comparison, a simple majority classifier always predicting CONTAINMENT would only achieve an accuracy of 31.0%, a weighted F1 score of 0.147, and a macro F1 score of 0.059 on the combined English and German test set.

In order to further detail the results, we present the class F1 scores in Table 4 as well as the confusion matrix in Figure 2, which were computed for one of the trained models for a mixed test set consisting of the German and English samples. The model performs best for the classes backed by the most training data, i.e., CONTAINMENT, SOURCE-PATH-GOAL, and VERTICALITY. Although a lot of data samples belong to the image schema FORCE it only has a class F1 score of 0.55, which is due to the high confusion with SOURCE-PATH-GOAL. For the classes with very little training data the model achieves a lower F1 score, although never below 0.5.

6.2 Scores Unsupervised Baseline

From all English samples in the dataset, only 36.5% contained a verb–preposition–noun triple. This low percentage highlights how important a word class-independent approach is. After clustering the resulting 613 triples, the highest score is achieved with 16 clusters, averaged triple embeddings, nearest-neighbors for computing the affinity matrix, and discretization. From the resulting clusters, 7 are labeled as CONTAINMENT, 4 as FORCE, 4 as SOURCE-PATH-GOAL, and 1 as VERTICALITY. The obtained accuracy is 43.5%, thus, much lower than the results obtained by XLM-R. The low macro-averaged F1 score of 0.20 shows the methods inability to properly deal with the class imbalance.

Choosing a higher number of output clusters, the scores can be increased, however, also requires a lot of manual analysis if being applied to unlabeled real world data. For example, with 32 clusters, the accuracy increases to 49.8%.

6.3 LIME Explanations

Looking at the LIME explanations for some wrongly classified samples, especially for those belonging to classes regularly confused according to the confusion matrix in Figure 2, we gained crucial insights regarding the inner workings of the model and issues in the dataset. Firstly, some of the salient points of the confusion matrix are due to common image schema collocations, i.e., two or more image schemas occurring together in the same sentence. An example of this are the four expressions with the gold label SCALE which were classified as VERTICALITY by the model. In all samples the two image schemas are collocated, e.g., in the expression *He's head and shoulders above everyone in the industry*, where LIME correctly

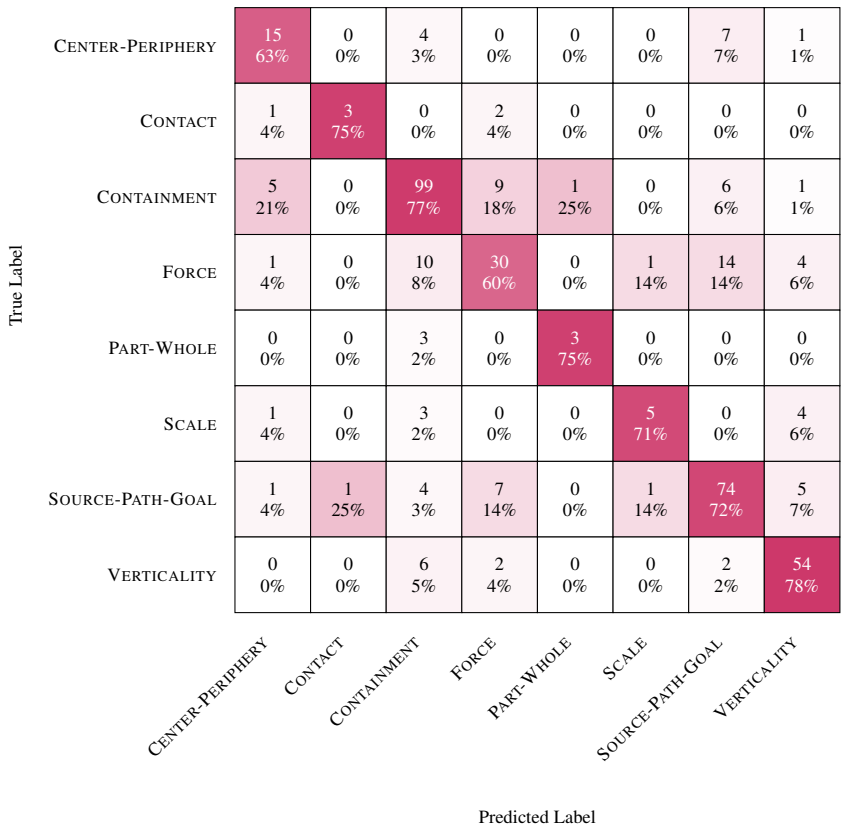

Figure 2: Confusion matrix for the image schema extraction model on the test set (English and German)

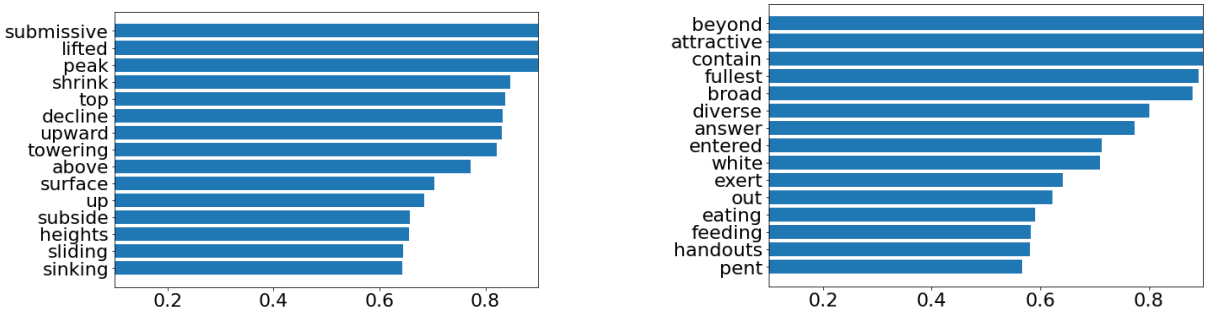

(a) VERTICALITY (b) CONTAINMENT

Figure 3: Words LIME finds as strong indicators for specific image schema class

identifies the word *above* as strong indicator for the image schema VERTICALITY. However, due to its quantitative, comparing nature, the phrase also belongs to the image schema SCALE as stated in the gold standard. Interestingly, the confusion is never the other way around, i.e., samples belonging to VERTICALITY are never classified as SCALE, which is most likely due to VERTICALITY being supported by more training data so that the model develops a certain bias towards that class. Other samples show some unintended learned behavior exhibited by the model. The expression *to have an open marriage*, having the gold label CONTAINMENT, is classified as SOURCE-PATH-GOAL by the model although LIME identifies *open* as an indicator for CONTAINMENT. However, LIME's output suggests that the model identified *marriage* as a concept that is often talked about in terms relating to the image schema SOURCE-PATH-GOAL, such as in conceptual metaphors like LOVE IS A JOURNEY. However, as this is not the case in the given context, the classifier takes a wrong decision.

Figure 3 shows the features with the highest indicator scores for the image schemas VERTICALITY and CONTAINMENT averaged over all samples in the test set. The words shown for VERTICALITY are all correctly identified as strong markers. Only looking further down the list, not shown in the figure anymore, one finds false positives, for instance, *wings* which only is related to themes were VERTICALITY plays a role. The words identified as strong indicators for CONTAINMENT contain more clear false positives, such as *white* or *answer*. The word *white* occurs in two natural language expressions labeled as CONTAINMENT in the dataset, while *answer* occurs four times, however, surprisingly never in a phrase labeled as CONTAINMENT.

7 Discussion

Task Design. A shortcoming of the current model and dataset is not considering multiple labels for one natural language expression. Thus, the task should be changed to a multi-label classification task supported by a corresponding dataset, which could be created by manually adapting the current annotations.

Moreover, instead of relying on additional explanations to identify constituents of image schematic language, one could try to approach image schema extraction as a token-level classification task, in which a label is not attributed to a full sentence but to each word or continuation of a word in a sentence individually. The classifier's output would then directly indicate which words of a sentence are used in an image-schematic way. However, one has to be careful not to treat words, especially prepositions, in isolation of their context. For instance, the word *on* often indicates spatial languages as in the phrase *on the path to*, but it can also be used in non-spatial contexts, e.g., *the book on biology*. When creating labels on a token-level, words need to be carefully and consistently annotated with image schemas, ideally following very explicit and clear annotation guidelines.

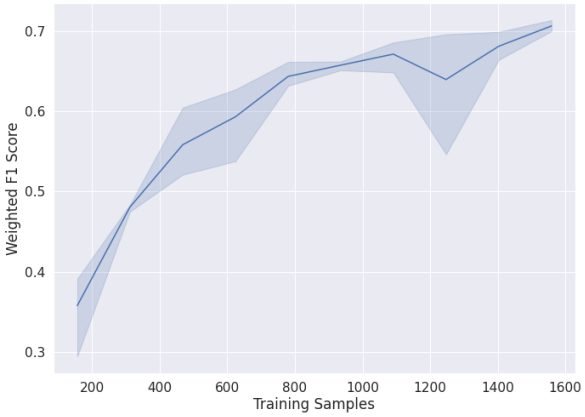

Figure 4: Learning curve computed in decimal intervals from 10% to 100% of training data and showing the average score and 95% confidence interval of three trained models

Dataset Improvements. Moreover, the dataset is missing data for some common image schemas, e.g. SUPPORT or BALANCE. In general, more datapoints, especially for CONTACT and PART-WHOLE as the two classes with the fewest datapoints and the lowest class F1 scores, would likely lead to an increase in the model's performance. This is also indicated by the learning curve in Figure 4 which still shows an increasing weighted F1 score given a higher number of overall training samples. For the model to function in the wild, it additionally requires training samples which are labeled as non-image-schematic language as it otherwise will label every sentence as image-schematic language. Furthermore, LIME revealed certain samples where the model made the correct decisions based on relevant features, but the gold standard had erroneous labels, which led to some corrections made on the dataset.

Global Explanations. To gain first insights into the global behavior of the model we introduced a simple algorithm for averaging LIME results over multiple samples. However, changing the procedure to rank words by taking into account how often they indicate a specific image schema class would also allow to gather information of which parts-of-speech are most commonly used in natural language expressions of a specific image schema. Such improved forms of global aggregations of local explanations were, for instance, designed and evaluated in form of the Submodular Pick algorithm proposed by Ribeiro et al. (2016) or the Global Average and Global Homogeneity-Weighted Importance proposed by van der Linden et al. (2019), which we, in the future, plan to implement and test in the context of image schema extraction.

8 Conclusion

We introduce a novel approach to perform image schema extraction from natural languages based on multilingual, pretrained neural language models. Thereby, a supervised training procedure can be implemented by finetuning the pretrained model with only a few training samples without making any prior assumptions about word classes. The model shows a strong cross-validated performance in English and German, and even shows the ability to generalize to languages unseen during finetuning. Explanations generated by the explainable AI approach show insights and shortcomings regarding the model behavior as well as the dataset annotation. To further improve the differentiation between image-schematic classes, a more equal distribution of training data would be beneficial. In terms of future work, we intend to add non-image-schematic samples to further enable the trained classifier to distinguish image-schematic from non-image-schematic expressions. In addition, the task should be devised as a multi-label classification task to account for the frequent phenomenon of image schema collocations. Lastly, we would like to improve the aggregation of local explanations and utilize it in order to systematically analyse image schematic language in a text corpus.

Acknowledgements

We would like to thank Chao Xu from Shandong University for his incredible help with compiling the Chinese Mandarin test set for this study.

References

Christa Baldauf. 1997. *Metapher und Kognition: Grundlagen einer neuen Theorie der Alltagsmetapher*. Lang.

B. Bennett and C. Cialone. 2014. Corpus guided sense cluster analysis: a methodology for ontology development (with examples from the spatial domain). In *8th International Conference on Formal Ontology in Information Systems (FOIS)*, volume 267 of *Frontiers in Artificial Intelligence and Applications*, pages 213–226. IOS Press.

Soonja Choi and Melissa Bowerman. 1991. Learning to express motion events in english and korean: The influence of language-specific lexicalization patterns. *Cognition*, 41(1-3):83–121.

Alan Cienki. 2008. Image schemas and gesture. In *From perception to meaning*, pages 421–442. De Gruyter Mouton.

Alexis Conneau, Kartikay Khandelwal, Naman Goyal, Vishrav Chaudhary, Guillaume Wenzek, Francisco Guzmán, Edouard Grave, Myle Ott, Luke Zettlemoyer, and Veselin Stoyanov. 2020. Unsupervised cross-lingual representation learning at scale. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 8440–8451, Online. Association for Computational Linguistics.

Verna Dankers, Karan Malhotra, Gaurav Kudva, Volodymyr Medentsiy, and Ekaterina Shutova. 2020. Being neighbourly: Neural metaphor identification in discourse. In *Proceedings of the Second Workshop on Figurative Language Processing*, pages 227–234, Online. Association for Computational Linguistics.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, Minneapolis, Minnesota. Association for Computational Linguistics.

Ellen Dodge and George Lakoff. 2005. Image schemas: From linguistic analysis to neural grounding. In Beate Hampe and Joseph E. Grady, editors, *From perception to meaning: Image schemas in cognitive linguistics*, pages 57–91. Mouton de Gruyter, Berlin.

Ellen K.. Dodge, Jisup Hong, and Elise Stickles. 2015. Metanet: Deep semantic automatic metaphor analysis. In *Proceedings of the Third Workshop on Metaphor in NLP*, pages 40–49.

Edith Durand, Pierre Berroir, and Ana Ines Ansaldo. 2018. The neural and behavioral correlates of anomia recovery following poem – personalized observation, execution, and mental imagery therapy: A proof of concept. *Neural Plasticity*.

Margaret H Freeman. 2002. Momentary stays, exploding forces: A cognitive linguistic approach to the poetics of emily dickinson and robert frost. *Journal of English Linguistics*, 30(1):73–90.

Raymond W. Gibbs and Herbert L. Colston. 1995. The cognitive psychological reality of image schemas and their transformation. *Cognitive Linguistics*, 6:347–378.

Raymond W Gibbs Jr, Dinara A Beitel, Michael Harrington, and Paul E Sanders. 1994. Taking a stand on the meanings of stand: Bodily experience as motivation for polysemy. *Journal of Semantics*, 11(4):231–251.

Dagmar Gromann and Maria M. Hedblom. 2017. Kinesthetic mind reader: A method to identify image schemas in natural language. In *Proceedings of Advancements in Cognitive Systems*.

Jörn Hurtienne. 2017. Image schema database (iscat).

Olaf Jäkel. 2003. *Wie Metaphern Wissen schaffen: die kognitive Metapherntheorie und ihre Anwendung in Modell-Analysen der Diskursbereiche Geistestätigkeit, Wirtschaft, Wissenschaft und Religion*. Kovač Hamburg.

Mark Johnson. 1987. *The Body in the Mind. The Bodily Basis of Meaning, Imagination, and Reasoning*. The University of Chicago Press.

Michael Kimmel. 2009. Analyzing image schemas in literature. *Cognitive Semiotics*, 5(1-2):159–188.

Parisa Kordjamshidi, Martijn Van Otterlo, and Marie-Francine Moens. 2011. Spatial role labeling: Towards extraction of spatial relations from natural language. *ACM Transactions on Speech and Language Processing (TSLP)*, 8(3):4.

Zoltán Kövecses. 2010. *Metaphor: A Practical Introduction*. Oxford University Press, USA.

George Lakoff. 1987. *Women, fire, and dangerous things. what categories reveal about the mind*. The University of Chicago Press.

George Lakoff, Jane Espenson, and Alan Schwartz. 1991. Master metaphor list. second draft copy. *University of California, Berkeley*.

George Lakoff and Mark Johnson. 1980. *Metaphors We Live By*. University of Chicago Press.

Laura Lakusta and Barbara Landau. 2005. Starting at the end: The importance of goals in spatial language. *Cognition*, 96(1):1–33.

Ronald W Langacker. 1987. *Foundations of cognitive grammar: Theoretical prerequisites*, volume 1. Stanford university press.

Chee Wee Leong, Beata Beigman Klebanov, Chris Hamill, Egon Stemle, Rutuja Ubale, and Xianyang Chen. 2020. A report on the 2020 vua and toefl metaphor detection shared task. In *Proceedings of the second workshop on figurative language processing*, pages 18–29.

Andrew Ng, Michael Jordan, and Yair Weiss. 2001. On spectral clustering: Analysis and an algorithm. In *Advances in Neural Information Processing Systems*, volume 14. MIT Press.

Anna Papafragou, Christine Massey, and Lila Gleitman. 2006. When english proposes what greek presupposes: The cross-linguistic encoding of motion events. *Cognition*, 98(3):B75–B87.

F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. 2011. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830.

Jeffrey Pennington, Richard Socher, and Christopher Manning. 2014. GloVe: Global vectors for word representation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1532–1543, Doha, Qatar. Association for Computational Linguistics.

Peng Qi, Yuhao Zhang, Yuhui Zhang, Jason Bolton, and Christopher D. Manning. 2020. Stanza: A Python natural language processing toolkit for many human languages. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics: System Demonstrations*.

Nitin Ramrakhiyani, Girish Palshikar, and Vasudeva Varma. 2019. A simple neural approach to spatial role labelling. In *Advances in Information Retrieval*, pages 102–108, Cham. Springer International Publishing.

Marco Tulio Ribeiro, Sameer Singh, and Carlos Guestrin. 2016. " why should i trust you?" explaining the predictions of any classifier. In *Proceedings of the 22nd ACM SIGKDD international conference on knowledge discovery and data mining*, pages 1135–1144.

Leonard Talmy. 2005. The fundamental system of spatial schemas in language. In Beate Hampe and Joseph E Grady, editors, *From perception to meaning: Image schemas in cognitive linguistics*, volume 29 of *Cognitive Linguistics Research*, pages 199–234. Walter de Gruyter.

Jon Tolaas. 1991. Notes on the origin of some spatialization metaphors. *Metaphor and Symbol*, 6(3):203–218.

Ilse van der Linden, Hinda Haned, and Evangelos Kanoulas. 2019. Global Aggregations of Local Explanations for Black Box models. *arXiv e-prints*, page arXiv:1907.03039.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, undefinedukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Proceedings of the 31st International Conference on Neural Information Processing Systems*, NIPS'17, page 6000–6010, Red Hook, NY, USA. Curran Associates Inc.

Lennart Wachowiak. 2020. Semi-automatic extraction of image schemas from natural language. In *Proceedings of the MEi:CogSci Conference 2020*, page 105. Comenius University, Bratislava.

Ning Yu. 1995. Metaphorical expressions of anger and happiness in english and chinese. *Metaphor and symbol*, 10(2):59–92.

How to Adapt Pre-trained Vision-and-Language Models to a Text-only Input?

Lovisa Hagström[1] **Richard Johansson**[1,2]
[1]Chalmers University of Technology, [2]University of Gothenburg
{lovhag, richajo}@chalmers.se

Abstract

Current language models have been criticised for learning language from text alone without connection between words and their meaning. Consequently, multimodal training has been proposed as a way for creating models with better language understanding by providing the lacking connection. We focus on pre-trained multimodal vision-and-language (VL) models for which there already are some results on their language understanding capabilities. An unresolved issue with evaluating the linguistic skills of these models, however, is that there is no established method for adapting them to text-only input without out-of-distribution uncertainty. To find the best approach, we investigate and compare seven possible methods for adapting three different pre-trained VL models to text-only input. Our evaluations on both GLUE and Visual Property Norms (VPN) show that care should be put into adapting VL models to zero-shot text-only tasks, while the models are less sensitive to how we adapt them to non-zero-shot tasks. We also find that the adaptation methods perform differently for different models and that unimodal model counterparts perform on par with the VL models regardless of adaptation, indicating that current VL models do not necessarily gain better language understanding from their multimodal training.

1 Introduction

Having models learn language from text alone has been criticised based on several aspects, from fundamental arguments about how language works (Bender and Koller, 2020) to findings on lack of certain information in text (Gordon and Van Durme, 2013; Paik et al., 2021). To train language models on more sources than text is therefore a proposed direction for creating language models with better language understanding (Bisk et al., 2020). These models would then become multimodal, with the capability to process both text and information from other modalities.

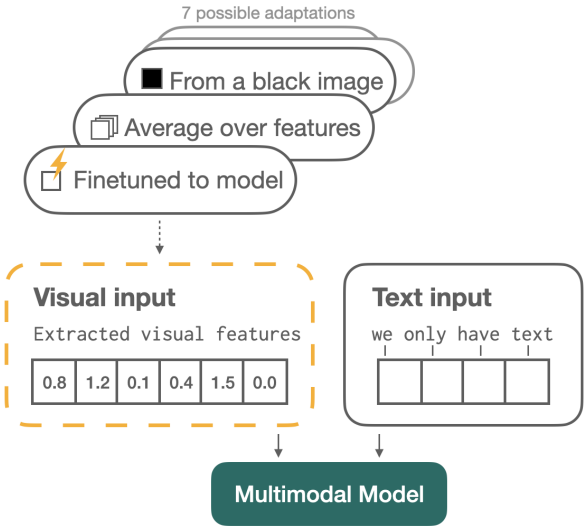

Figure 1: An overview of different ways to adapt a multimodal model to text-only input. It showcases three of the seven adaptations evaluated in this work.

The multimodal models of interest in this work are vision-and-language (VL) models that have been trained on images and their corresponding captions or visual questions (Lu et al., 2019; Tan and Bansal, 2019; Su et al., 2020; Li et al., 2019; Chen et al., 2020). These models are performant on several image-text tasks such as image captioning and VQA, while there also is an increased interest for evaluating how their natural language understanding is influenced by their multimodal training (Iki and Aizawa, 2021; Yun et al., 2021).

It is however tricky to investigate the pure natural language understanding of the aforementioned VL models, since their language processing is conditioned on visual features. For certain investigations, we may simply wish to evaluate the models on text-only domains, while these models have not been developed for this purpose. If we do not attend to the issue of accurately adapting VL models to text-only domains we risk evaluating them out-of-distribution and fail to accurately measure their natural language understanding capabilities.

Proceedings of the 29th International Conference on Computational Linguistics, pages 5582–5596
October 12–17, 2022.

Different methods for adapting VL models to a text-only input have already been tried and we have some results on the natural language understanding capabilities of these models (Iki and Aizawa, 2021; Yun et al., 2021). However, no systematic search for the best way to adapt VL models to a text-only input has been performed and it is unclear how well the VL models work with the previously proposed adaptations. If we wish to continue the search for better natural language understanding in multimodal models, we should ensure that we evaluate them in the best way possible. In this work, we search for the best method for adapting existing VL models to a text-only input, as illustrated in Figure 1.[1]

With the adaptations in place, we can then compare the VL models to their unimodal text-only counterparts. This will complement already existing results on the natural language understanding capabilities of VL models and the effect of multimodal training.

The contributions of our work are as follows:

- We investigate and compare seven methods for adapting LXMERT (Tan and Bansal, 2019), VisualBERT (Li et al., 2019) and CLIP-BERT (Norlund et al., 2021) to a text-only input (Section 3). Two of these adaptations have already been used in previous investigations of the linguistic capabilities of VL models (Frank et al., 2021; Iki and Aizawa, 2021).
- We evaluate these adaptations on the GLUE benchmark (Wang et al., 2018) (Section 4.1). This gives us results on how well the adaptations work for tasks that aim to evaluate general natural language understanding.
- We also evaluate the adaptations on the Visual Property Norms (VPN) (Hagström and Johansson, 2022) (Section 4.2). This gives us results on how well the adaptations work for zero-shot tasks that aim to evaluate visual conceptual knowledge in the models.
- We compare the adapted VL models to their unimodal BERT-base (Devlin et al., 2019) counterparts on the aforementioned evaluation tasks. The ensuing results should provide additional clarity on the natural language understanding of VL models.
- We also compare the adapted VL models to the multimodal FLAVA model (Singh et al., 2022) that requires no adaptation to text-only tasks.

[1]Code available at https://github.com/lovhag/adapt-pre-trained-VL-models-to-text

2 Models

We investigate adaptations to text-only input for the three multimodal models CLIP-BERT, LXMERT and VisualBERT. We also compare their results with those of a baseline BERT-base model and FLAVA. The models are further described below and an overview of them can be found in Table 1.

For each of the multimodal models, we also describe how to make the model function without visual input. This is later used in some of the adaptations we evaluate, described in Section 3.

All models evaluated in this work except for CLIP-BERT are provided by the Huggingface library (Wolf et al., 2020). The pre-trained model weights for all models except for CLIP-BERT are also provided by this library. The CLIP-BERT weights are found in our public repository.

2.1 VisualBERT

VisualBERT is a single-stream model that has been initialized from pre-trained BERT-base weights and then further trained on MS COCO as well as VQA (Lin et al., 2014; Goyal et al., 2017). As a result, it has been trained on 1.27M more texts and 0.12M more images than BERT-base. It utilizes a Faster R-CNN detector (Anderson et al., 2018) as backbone, for which it has been trained on the features of the 36 first detections, meaning that it expects visual input features with shape $(36, 2048)$.

Usage without visual input The single-stream architecture of this model implies that it simply concatenates the embeddings from the visual features with the word embeddings from the text input and then forwards this to the BERT encoder. Therefore, this model can be queried with text only without changing anything in the model architecture, since it simply means that only the word embeddings are fed to the BERT encoder.

2.2 LXMERT

LXMERT is a dual-stream model trained on MS COCO, VQA, VG, GQA and VG-QA (Hudson and Manning, 2019; Zhu et al., 2016). It has not been initialized from BERT-base weights. In total, it has been trained on 9.18M visual texts and 0.18M

Model	Size	Pre-train data	Backbone
BERT-base	110M	English Wiki, BookCorpus	-
FLAVA	86M	CCNews, BookCorpus, PMD	-
CLIP-BERT	110M	English Wiki, BookCorpus, CLIP-BERT V+L	CLIP
LXMERT	230M	LXMERT V+L	Faster R-CNN
VisualBERT	110M	English Wiki, BookCorpus, VisualBERT V+L	Faster R-CNN

Table 1: The models evaluated with details on their pre-training data. The V+L datasets refer to model-specific VL datasets. With 'FLAVA' we refer to the text encoder.

images. Similarly to VisualBERT, this model expects visual features of the shape $(36, 2048)$ from 36 Faster R-CNN detections.

Usage without visual input The dual-stream architecture of this model implies that it processes the visual embeddings and word embeddings in separate encoders before it fuses the information from them in a so called Cross-Modality Encoder. For this model it does not suffice to simply omit the visual input since it is expected by a separate visual encoder. However, the language output of the model is only affected by the visual input at a set of cross-attention sub-layers with residual connections in the Cross-Modality Encoder. Consequently, we can set the added residual from the cross-attention layer to zero and remove the visual encoder of the model.

2.3 CLIP-BERT

CLIP-BERT is a single-stream VL model that is architecturally very similar to VisualBERT. The main differences this model introduces are two, 1) it has a CLIP (Radford et al., 2021) backbone that generates visual features of dimension $(512,)$ for each image, and 2) it has been trained on 4.72M visual texts and 2.91M images, a vision-language dataset approximately four times larger than that of VisualBERT, in addition to having been initialized from BERT-base weights.

Usage without visual input Similarly to VisualBERT, the single-stream architecture of this model implies that it can be queried with text only without changing anything in the model architecture, since it simply means that only the word embeddings are fed to the BERT encoder.

2.4 BERT-base

Since all VL models we evaluate to some extent are based on BERT-base, we use this unimodal model as a baseline in our evaluations seen in Section 4.

We also create two additional baseline versions of BERT-base by further training the pre-trained model on LXMERT text data[2] and a subset of the English Wikipedia corpus from the Huggingface Datasets library (Lhoest et al., 2021) sampled to match the LXMERT text data in size, respectively. We do this to enable more fair comparisons to the evaluated VL models, since they have received additional training on text and images. These model versions are denoted by `trained-LXMERT` and `trained-Wikipedia`. The unchanged BERT-base model is denoted by `default`.

Since the original LXMERT model developed by Tan and Bansal (2019) was not initialized from BERT weights, we also develop a third baseline version of BERT that has been trained from scratch on LXMERT text data for comparison. This model version is denoted by `trained-LXMERT-scratch`.

More information about the datasets used to train the BERT-base baselines and training procedures can be found in Appendices A and B respectively.

2.5 FLAVA

FLAVA is a multimodal model that works for all combinations of VL modalities without any need for adaptation (Singh et al., 2022). It sidesteps all issues related to the aforementioned VL models and can directly be evaluated for its linguistic capabilities. It consists of three separate parts: an image encoder, a text encoder, and a multimodal encoder that combines the input from the unimodal encoders. The unimodal encoders are pretrained on unimodal datasets and the full model is then trained end-to-end on the Public Multimodal Datasets (PMD) corpus (Singh et al., 2022). We use the text encoder of this model as a baseline in our evaluations.

[2]The data is described in https://github.com/airsplay/lxmert.

3 Adaptations to text-only input

There are several ways to adapt a VL model to a text-only input. In this work we investigate and compare seven possible adaptations, as described below. Two of the adaptations described here have already been used for investigating the linguistic capabilities of VL models (Frank et al., 2021; Iki and Aizawa, 2021). Common for all adaptations is that their intended use is for evaluation of a pre-trained VL model (encoder) on text-only input. When we refer to the word *adaptation* we refer to the adaptation of a VL model to text-only input.

The adaptions can be grouped into three different categories based on how they are implemented. For the first category, we simply remove the visual input to the VL model (Sections 3.1 and 3.2). For the second category, we provide the model with visual features that are constant and can be viewed as fillers, (Sections 3.3 to 3.6). For the third category, we provide the model with visual features predicted from text (Section 3.7).

3.1 Using model as-is without visual input

All VL models considered in this work can be queried with text only, or after performing a small set of alterations to the model architecture without changing any pre-trained model weights, as described in Section 2. Thus, we can directly evaluate the pre-trained models on the text-only task of interest. This adaptation is denoted by `default`.

This adaptation is very simple to apply and does not require any additional computations, while it assumes that the VL model can be queried without visual input. It is also not certain that the models will function as intended due to the imposed train/test shift of this adaptation. To our knowledge, this approach has not been tested before.

3.2 Fine-tuning model on text-only input

Before evaluating the pre-trained VL model we fine-tune it on a small text-only fine-tuning task, similarly to how several natural language understanding tasks are performed (Wang et al., 2018, 2019). The idea is that this will acclimatize the model to the aforementioned domain shift. Similarly to the `default` adaptation, this also relies on being able to use the model without visual input.

We create two separate fine-tuning sets for this adaptation. The sets have been extracted from the text part of the LXMERT training data and from English Wikipedia. Their sizes have been adapted to match those of typical fine-tuning sets for e.g. SuperGLUE (Wang et al., 2019) and we have ensured that the number of tokens in each fine-tuning set is roughly equal. More information about the datasets can be found in Appendix A.

Finetuning the VL models on each of these sets should give us results on both the performance of the method, and on how dependent it is on the chosen fine-tuning set. These adaptations are denoted by `no-visual-features-finetuned-LXMERT` and `no-visual-features-fine-tuned-Wikipedia` respectively.

This method avoids having to work with image feature extractors and image data. However, it requires setting up a training algorithm and additional computations. Moreover, since the full model needs to be trained for this adaptation, it is more sensitive to hyperparameter choices. It is also not certain whether it is sufficient to perform fine-tuning on text to acclimatize VL models to a text-only input. More information on tuning procedures can be found in Appendix B.

3.3 Using averaged visual features from the training dataset

In this method we give the VL model a constant visual feature input together with the text of interest at evaluation, where the visual features are the average of all the visual features in the train or evaluation data of the model. The provided visual-features should then be kept in-distribution, while they also are uninformative. This adaptation has already been used by Frank et al. (2021) for ablating visual input to VL models. We denote it by `avg-visual-features`.

No assumptions or changes to the model architecture are necessary for this method. However, it requires access to the datasets used to train the model of interest and the computation of the averaged visual feature vector.

We calculate the averaged visual feature vector for CLIP-BERT based on the CLIP features of its training data. We also calculate the averaged visual features and position vectors for LXMERT from its corresponding training data. We take the average across training samples per detection for the LXMERT visual features such that we get one average feature vector for the first detection, another for the second detection and so forth up to the 36th.

The original released VisualBERT visual features are not compatible with the Huggingface im-

plementation of the model used in this work. We instead provide VisualBERT with the LXMERT averaged visual features, since they are compatible.

3.4 Using visual features from a black image

The idea is yet again to give the VL model a constant visual feature input together with the text of interest. In this case, the visual features are extracted from a black image using the model backbone. The model then receives a visual input similar to what it has been trained on, while it does not contain any information. This adaptation has already been used by Iki and Aizawa (2021) for evaluating e.g. VisualBERT and LXMERT on GLUE. We denote it by `zero-image-visual-features`.

Similarly to the averaged features adaptation, this adaptation makes no assumptions about the model of interest. However, it requires access to the backbone of the model and the computation of the visual features from a black image.

We use the LXMERT feature extractor to extract 36 detections with their visual features and bounding boxes from a black image. The extractor is a Faster R-CNN model developed by Anderson et al. (2018). These features are then given to LXMERT and VisualBERT during evaluation. For CLIP-BERT we use CLIP to extract visual features from the same black image.

3.5 Using constant zero vector visual features

We give the model of interest constant visual features, and the positions of bounding boxes in the case of LXMERT, that are zeros. There are no guarantees that this method will work well for adapting VL models to a text-only input. It is however easy to implement and can be seen as a baseline to be compared with the other adaptations. To the knowledge of the authors, this method has not been used previously. We denote it by `zeroed-visual-features`.

3.6 Using tuned visual features

We tune the visual features to a frozen version of the model of interest, and then use these constant features at evaluation together with the text of interest. To the knowledge of the authors, this method has not been used previously to adapt VL models to a text-only input. However, the key idea of tuning the input to the model has been used in previous works (Qin and Eisner, 2021; Tsimpoukelli et al., 2021).

We tune visual features to frozen and pretrained versions of CLIP-BERT, VisualBERT and LXMERT respectively. We tune on the same LXMERT and Wikipedia sets used for the adaptation described in Section 3.2. More information on tuning procedures can be found in Appendix B.

This method offers more flexibility for finding the most suitable constant visual features for a VL model evaluated on text-only tasks. However, it also requires setting up the training, more computations and is more sensitive to hyperparameter tuning. We denote these adaptations on the different fine-tuning sets by `finetuned-LXMERT-visual-features` and `finetuned-Wikipedia-visual-features` respectively.

3.7 Predicting visual features from text

Some feature extractors map text representations and visual representations to the same parametric space. Consequently, they can be used to "imagine" visual features from text. The CLIP model serving as a backbone for the CLIP-BERT model has this capability and can be used to generate visual features from text during evaluation on text-only tasks. We implement it for the CLIP-BERT model and denote it by `imagined-visual-features`.

This method is quite simple to implement, while it requires access to CLIP and computing the visual features from the evaluation corpus. It is also not clear how well CLIP representations work for text that is not specifically related to visual concepts.

4 Evaluation methods

To assess the performance of our text-only adaptations, we firstly evaluate them on the GLUE benchmark, described in Section 4.1. These evaluations will give us results on how well the models and their adaptations work for general natural language understanding tasks. This benchmark has been used by both Devlin et al. (2019) and Iki and Aizawa (2021) to evaluate natural language understanding capabilities of BERT and VL models.

Furthermore, to assess the performance of the adaptations on text domains that are more focused on visual concepts, we perform evaluations on VPN, further described in Section 4.2. This will provide us with results on tasks the VL models potentially are more attuned to, complementing the general GLUE results.

MLM query	Gold labels
a cow usually is [...]	black, white
a mug has a [...]	handle
q: a greeting card has? a: [...]	pictures

Table 2: Query samples from the VPN dataset for the concepts *cow*, *mug* and *greeting card* using three out of nine possible query templates. The [...] is typically replaced with a `[MASK]` token.

4.1 GLUE

The General Language Understanding Evaluation (GLUE) benchmark has the aim to evaluate model performance across several NLU tasks. It was developed by Wang et al. (2018) and has since then been used to evaluate the natural language understanding of several LMs, including BERT.

GLUE contains nine different tasks testing for grammatical correctness understanding (CoLA), sentiment classification (SST-2), semantic equivalence detection on different text domains (MRPC, QQP, STS-B), textual entailment (MNLI, RTE), answer extraction from text (QNLI) and reading comprehension (WNLI). All tasks are sentence classification tasks and have corresponding train and validation sets for fine-tuning.

VL models have already been evaluated on GLUE by Iki and Aizawa (2021) using the black image adaptation method listed in Section 3.4. We extend the GLUE evaluation to include all alternative adaptation approaches listed in Section 3.

To evaluate the performance of our adaptations on GLUE, we first fine-tune our selected multimodal models with each adaptation on the training sets of the GLUE tasks. We then report the validation scores of the models and their adaptations. More information about the fine-tuning procedures can be found in Appendix B.

4.2 Visual Property Norms

Our current VL models are not necessarily the best fit for general NLU tasks such as GLUE (Iki and Aizawa, 2021; Yun et al., 2021). Therefore, we also evaluate them on a task we assume they are more suitable to. Visual Property Norms (VPN) essentially queries a model for the basic visual properties of a set of concepts (Hagström and Johansson, 2022). It is a text-only task, while it explicitly focuses on visual properties and concepts. Thus, if the VL models should perform particularly well on any text-only task, this would be the one. Table 2 displays examples of queries from the VPN dataset.

The VPN dataset is a zero-shot evaluation task that evaluates a model using masked language modelling (MLM), an objective our models already have been trained on. To mitigate issues with prompt-sensitivity of LMs, nine different query templates are applied during evaluation.

VPN is a version of the CSLB concept property norms dataset (Devereux et al., 2014) filtered to only contain visual conceptual features. The original property norms dataset was created with the help of 123 human participants asked to list the features of a set of concepts. Each concept has in total been exposed to 30 humans and the maximum frequency of a feature reported for a concept is then 30 and the minimum 2. This frequency is referred to as Production Frequency (PF).

VPN has been segmented into five partitions based on thresholding of PFs. We evaluate our adaptations on the segment for which PF $\geq$ 10, such that ten or more annotators jointly have produced the visual features in this set. It consists of 2,001 feature entries for 621 different concepts.

5 Results

We report the evaluation results for CLIP-BERT, LXMERT and VisualBERT with the seven potential adaptions to text-only input in Figure 2. We also report the results for our four BERT-base baselines. For GLUE we report the macro-averaged score over the GLUE tasks. The score for each task is measured using its corresponding predefined metric described by Wang et al. (2018). For VPN evaluation we report the mean average precision (mAP) score averaged over each concept and relation per query template.

We also report the evaluation results on GLUE for each task and model for the best performing adaptation measured by average GLUE score in Table 3. Table 4 similarly reports the model scores on VPN for the best adaptation on average. We compare these results to those of the FLAVA text encoder that requires no adaptation. Complete numerical results can be found in Appendix C.

We format our discussion around a set of statements that can be made with respect to the results of this work, as follows.

Model performance on GLUE is more sensitive to pre-training than to adaptation Model performance on GLUE varies insignificantly between different adaptations for each model in Figure 2a.

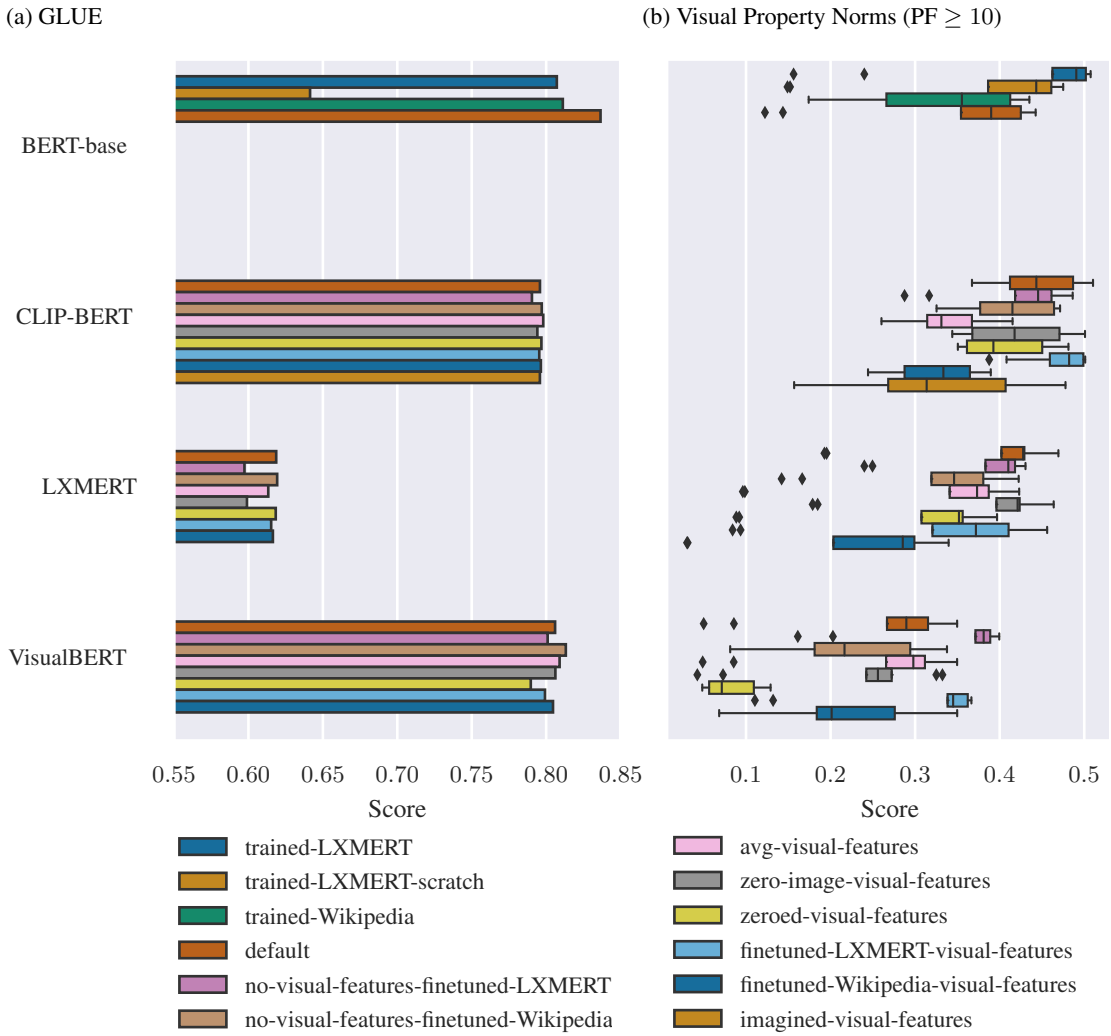

Figure 2: Results on GLUE and VPN from evaluating different adaptations to text-only input. The GLUE results are given by the mean of the scores for the development sets of all tasks, excluding WNLI. The metric used for each task is F1 score for QQP and MRPC, Matthews correlation for CoLA, Spearman correlation for STS-B, and accuracy for the remaining tasks. For VPN the box length indicates prompt sensitivity over nine different query templates.

The CLIP-BERT performance varies with less than 0.01 score points between adaptations and the VisualBERT performance with at most 0.02 score points between 'no-visual-features-finetuned-Wikipedia' and 'zeroed-visual-features'. The LXMERT performance also has a performance difference of at most 0.02 score points between 'default' and 'zero-image-visual-features'.

The largest performance difference on the GLUE benchmark can be observed between models, where LXMERT and BERT-base trained from scratch on LXMERT data perform significantly worse in comparison to the other models. Most likely, this is due to that the models were not initialized from BERT weights and consequently were not tuned to more general language usage.

A possible explanation for why the adaptation methods seem to matter so little for the GLUE results is that the benchmark is not zero-shot. The fine-tuning performed on each task might provide a sufficient of signal to the model for it to adapt to the unimodal domain.

Lastly, our results on GLUE for LXMERT and VisualBERT differ from those obtained by Iki and Aizawa (2021). Especially with respect to LXMERT for which we observe a significant performance difference compared to the other VL models, while the same cannot be observed in the results by Iki and Aizawa (2021). However, this should not raise any concerns about the robustness of the results, since we evaluated the original released models, while Iki and Aizawa (2021) eval-

	CoLA	MNLI	MRPC	QNLI	QQP	RTE	SST-2	STS-B
BERT-base	61.1	84.6	87.3/91.2	91.9	91.1/88.0	70.4	93.7	88.2
FLAVA	50.1	81.6	83.6/88.3	87.8	90.4/87.2	55.6	92.4	87.1
CLIP-BERT	55.4	83.2	75.5/84.1	89.8	91.1/88.0	58.1	92.0	87.8
LXMERT	15.9	68.1	69.9/81.6	68.0	84.1/76.8	58.5	86.6	40.1
VisualBERT	53.3	83.7	80.4/86.4	90.7	90.9/87.6	67.5	91.7	89.6

Table 3: GLUE development set results per task for the best performing adaptation on average. The best performing adaptation for each model is 'default' for BERT-base, 'avg-visual-features' for CLIP-BERT, 'no-visual-features-finetuned-Wikipedia' for LXMERT and 'no-visual-features-finetuned-Wikipedia' for VisualBERT . We report Matthew's correlation for CoLA, average accuracy for MNLI, accuracy/F1 score for MRPC, accuracy for QNLI, accuracy/F1 for QQP and accuracy for RTE and SST-2 and Spearman correlation for STS-B.

	VPN Score
BERT-base	49.1 ± 13.2
FLAVA	30.7 ± 6.9
CLIP-BERT	48.2 ± 4.2
LXMERT	42.8 ± 10.8
VisualBERT	38.1 ± 9.1

Table 4: VPN results for the best performing adaptations. The best performing adaptation for each model is 'trained-LXMERT' for BERT-base, 'finetuned-LXMERT-visual-features' for CLIP-BERT, 'default' for LXMERT and 'no-visual-features-finetuned-LXMERT' for VisualBERT. We report the results as median $\pm$ standard deviation over the nine query templates.

uated models that had been unified and trained on the same data by Bugliarello et al. (2021). And as we observed previously, GLUE results are sensitive to pre-training process.

Performance on Visual Property Norms is sensitive to adaptation In contrast to the observations made for GLUE, performance on VPN differs significantly between different adaptions in Figure 2b. In contrast to GLUE, this task is zero-shot and may provide a greater challenge for models that are not sufficiently tuned to the unimodal text domain.

Additionally, it is worth noting that the model performance is quite sensitive to the choice of query template. However, this is not entirely unexpected since it has been shown that LMs are prompt-sensitive in prompt-based retrieval evaluations (Cao et al., 2021; Jiang et al., 2020).

Different adaptations perform differently for different models on Visual Property Norms For CLIP-BERT, the most suitable adaptation for evaluation on VPN is to provide the model with visual features that have been tuned on LXMERT text data. For LXMERT, the best approach is to use the model as-is without visual input, and for VisualBERT the best adaptation is to fine-tune the model on LXMERT data without visual features. Common for all of these adaptations is that they involve some kind of prior tuning on LXMERT text data. A potential explanation for this is that the LXMERT data is more similar to VPN and results in a smaller domain shift.

An explanation for the varying adaptation fits between VL models is potentially found by looking at the different pre-training datasets and architectures of the models. VisualBERT has been tuned on much less data compared to the other models, and may therefore benefit from more training in general. Additionally, the single-stream CLIP-BERT and VisualBERT models process all linguistic and visual information in a joint manner, without the same ability to disentangle signals as the dual-stream LXMERT model.

FLAVA does not outperform adapted VL models On both GLUE and especially VPN, FLAVA is not better than the adapted VL models. This contrasts the GLUE results reported for FLAVA and other VL models by Singh et al. (2022). The difference in results may arise from differences in fine-tuning methods for GLUE and that we do not evaluate unified VL models.

Based on our results, adapting VL models to text-only input works better or equal to developing a model to work for all modalities from the start, as was done for FLAVA. However, since all models evaluated have been trained on different datasets with different objectives we cannot draw certain conclusions related to model design.

BERT-base baselines outperform vision-and-language models regardless of adaptation

Lastly, we can observe in Figure 2 how default BERT-base and BERT-base trained on LXMERT text data each have among the best performances on GLUE and VPN respectively. For GLUE it is expected: Iki and Aizawa (2021) have already observed that VL models on average perform worse on GLUE. Yun et al. (2021) also found similar results when they compared the quality of the linguistic representations of VisualBERT to those of BERT-base. The general natural language understanding capabilities evaluated in GLUE are potentially not easy to learn from a visual modality, explaining why the VL models did not perform better on this task. Our results on VPN are perhaps more surprising.

From their visual training, the VL models should more easily have gained natural language understanding capabilities necessary for better performance on VPN. Three potential explanations for why a LM still outperforms a VL model on VPN are 1) BERT-base is better tuned and therefore has a better overall performance, 2) the VL models evaluated in this work do not learn more about visual concepts from images compared to text that has been curated to contain visual information, or 3) the VPN task does not accurately measure the visual conceptual information we have in mind. More investigations are necessary to accurately determine the reason. In support of explanation (2), Abdou et al. (2021) found that there are similarities between color representations in LMs and actual perceptual color spaces, indicating that visual perceptual information may be found in text.

We should also note that none of the models evaluated in this work were developed with the goal of achieving better natural language understanding by multimodal training. This potentially explains some of our results, and provides an interesting avenue for future research in developing models that have a better performance on both unimodal and multimodal tasks.

6 Related Work

As previously mentioned, Iki and Aizawa (2021) have already looked at the language understanding capabilities of VL models, while they only looked at one way of adapting these models to a text-only input and only evaluated on GLUE. Yun et al. (2021) also look at the language understanding capabilities of VL models by evaluating the linguistic representations of VisualBERT and compare them to a BERT-base model that has been trained on the same text data. In contrast to their work, we investigate several VL models and evaluate their performance on language generation tasks.

Bugliarello et al. (2021), Hessel and Lee (2020), Thrush et al. (2022) and Frank et al. (2021) also perform extensive evaluations of several VL models such as LXMERT and VisualBERT. In contrast to our work, they primarily focus on the VL performance of the models, and do not consider the model performance on text-only input.

Tan and Bansal (2020) introduce a new method for enriching the textual representations of a model by training on visual information. Their method results in a model that can be directly applied to text-only tasks and outperforms its standard BERT model counterpart on GLUE. This method provides a parallel research avenue compared to adapting VL models to text-only input.

7 Conclusions

We have investigated and compared seven possible adaptations of CLIP-BERT, LXMERT and VisualBERT to text-only input by evaluation on GLUE and Visual Property Norms. We can conclude that care should be put into adapting these pre-trained VL models to text-only input for better performance on zero-shot tasks, while the choice of adaptation method seems to be less impactful on tasks coupled with fine-tuning sets.

Finally, we have observed that a unimodal LM has a performance on text-only tasks that is better or comparative to that of its VL model counterparts, regardless of how these counterparts were adapted to text-only input. Seemingly, improved pure text capabilities are not guaranteed from simply training a model on arbitrary multimodal tasks. This agrees with and solidifies previous research results on VL models.

Acknowledgements

We would like to thank the reviewers for their valuable feedback and advice on related work.

This work was partially supported by the Wallenberg AI, Autonomous Systems and Software Program (WASP) funded by the Knut and Alice Wallenberg Foundation. The computations were enabled by resources provided by the Swedish National Infrastructure for Computing (SNIC), partially funded by the Swedish Research Council through grant agreement no. 2018-05973.

References

Mostafa Abdou, Artur Kulmizev, Daniel Hershcovich, Stella Frank, Ellie Pavlick, and Anders Søgaard. 2021. Can language models encode perceptual structure without grounding? a case study in color. In *Proceedings of the 25th Conference on Computational Natural Language Learning*, pages 109–132, Online. Association for Computational Linguistics.

Peter Anderson, Xiaodong He, Chris Buehler, Damien Teney, Mark Johnson, Stephen Gould, and Lei Zhang. 2018. Bottom-up and top-down attention for image captioning and visual question answering. In *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 6077–6086.

Emily M. Bender and Alexander Koller. 2020. Climbing towards NLU: On meaning, form, and understanding in the age of data. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 5185–5198, Online. Association for Computational Linguistics.

Yonatan Bisk, Ari Holtzman, Jesse Thomason, Jacob Andreas, Yoshua Bengio, Joyce Chai, Mirella Lapata, Angeliki Lazaridou, Jonathan May, Aleksandr Nisnevich, Nicolas Pinto, and Joseph Turian. 2020. Experience grounds language. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 8718–8735, Online. Association for Computational Linguistics.

Emanuele Bugliarello, Ryan Cotterell, Naoaki Okazaki, and Desmond Elliott. 2021. Multimodal pretraining unmasked: A meta-analysis and a unified framework of vision-and-language BERTs. *Transactions of the Association for Computational Linguistics*, 9:978–994.

Boxi Cao, Hongyu Lin, Xianpei Han, Le Sun, Lingyong Yan, Meng Liao, Tong Xue, and Jin Xu. 2021. Knowledgeable or educated guess? revisiting language models as knowledge bases. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 1860–1874, Online. Association for Computational Linguistics.

Yen-Chun Chen, Linjie Li, Licheng Yu, Ahmed El Kholy, Faisal Ahmed, Zhe Gan, Yu Cheng, and Jingjing Liu. 2020. Uniter: Universal image-text representation learning. In *ECCV*.

Barry J Devereux, Lorraine K Tyler, Jeroen Geertzen, and Billi Randall. 2014. The centre for speech, language and the brain (cslb) concept property norms. *Behavior research methods*, 46(4):1119–1127.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, Minneapolis, Minnesota. Association for Computational Linguistics.

Stella Frank, Emanuele Bugliarello, and Desmond Elliott. 2021. Vision-and-language or vision-for-language? on cross-modal influence in multimodal transformers. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 9847–9857, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Jonathan Gordon and Benjamin Van Durme. 2013. Reporting bias and knowledge acquisition. In *Proceedings of the 2013 workshop on Automated knowledge base construction*, pages 25–30.

Yash Goyal, Tejas Khot, Douglas Summers-Stay, Dhruv Batra, and Devi Parikh. 2017. Making the V in VQA matter: Elevating the role of image understanding in Visual Question Answering. In *Conference on Computer Vision and Pattern Recognition (CVPR)*.

Lovisa Hagström and Richard Johansson. 2022. What do models learn from training on more than text? measuring visual commonsense knowledge. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics: Student Research Workshop*, pages 252–261, Dublin, Ireland. Association for Computational Linguistics.

Jack Hessel and Lillian Lee. 2020. Does my multimodal model learn cross-modal interactions? it's harder to tell than you might think! In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 861–877, Online. Association for Computational Linguistics.

Drew A Hudson and Christopher D Manning. 2019. Gqa: A new dataset for real-world visual reasoning and compositional question answering. *Conference on Computer Vision and Pattern Recognition (CVPR)*.

Taichi Iki and Akiko Aizawa. 2021. Effect of visual extensions on natural language understanding in vision-and-language models. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 2189–2196, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Zhengbao Jiang, Frank F. Xu, Jun Araki, and Graham Neubig. 2020. How Can We Know What Language Models Know? *Transactions of the Association for Computational Linguistics*, 8:423–438.

Quentin Lhoest, Albert Villanova del Moral, Yacine Jernite, Abhishek Thakur, Patrick von Platen, Suraj Patil, Julien Chaumond, Mariama Drame, Julien Plu, Lewis Tunstall, Joe Davison, Mario Šaško, Gunjan Chhablani, Bhavitvya Malik, Simon Brandeis, Teven Le Scao, Victor Sanh, Canwen Xu, Nicolas

Patry, Angelina McMillan-Major, Philipp Schmid, Sylvain Gugger, Clément Delangue, Théo Matussière, Lysandre Debut, Stas Bekman, Pierric Cistac, Thibault Goehringer, Victor Mustar, François Lagunas, Alexander Rush, and Thomas Wolf. 2021. Datasets: A community library for natural language processing. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 175–184, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Liunian Harold Li, Mark Yatskar, Da Yin, Cho-Jui Hsieh, and Kai-Wei Chang. 2019. Visualbert: A simple and performant baseline for vision and language. *CoRR*, abs/1908.03557.

Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollar, and Larry Zitnick. 2014. Microsoft coco: Common objects in context. In *ECCV*. European Conference on Computer Vision.

Jiasen Lu, Dhruv Batra, Devi Parikh, and Stefan Lee. 2019. Vilbert: Pretraining task-agnostic visiolinguistic representations for vision-and-language tasks. In *Advances in Neural Information Processing Systems*, volume 32. Curran Associates, Inc.

Tobias Norlund, Lovisa Hagström, and Richard Johansson. 2021. Transferring knowledge from vision to language: How to achieve it and how to measure it? In *Proceedings of the Fourth BlackboxNLP Workshop on Analyzing and Interpreting Neural Networks for NLP*, pages 149–162, Punta Cana, Dominican Republic. Association for Computational Linguistics.

Cory Paik, Stéphane Aroca-Ouellette, Alessandro Roncone, and Katharina Kann. 2021. The World of an Octopus: How Reporting Bias Influences a Language Model's Perception of Color. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 823–835, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Guanghui Qin and Jason Eisner. 2021. Learning how to ask: Querying LMs with mixtures of soft prompts. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 5203–5212, Online. Association for Computational Linguistics.

Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, Gretchen Krueger, and Ilya Sutskever. 2021. Learning transferable visual models from natural language supervision.

Amanpreet Singh, Ronghang Hu, Vedanuj Goswami, Guillaume Couairon, Wojciech Galuba, Marcus Rohrbach, and Douwe Kiela. 2022. FLAVA: A foundational language and vision alignment model. In *CVPR*.

Weijie Su, Xizhou Zhu, Yue Cao, Bin Li, Lewei Lu, Furu Wei, and Jifeng Dai. 2020. Vl-bert: Pre-training of generic visual-linguistic representations. In *International Conference on Learning Representations*.

Hao Tan and Mohit Bansal. 2019. LXMERT: Learning cross-modality encoder representations from transformers. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 5100–5111, Hong Kong, China. Association for Computational Linguistics.

Hao Tan and Mohit Bansal. 2020. Vokenization: Improving language understanding with contextualized, visual-grounded supervision. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 2066–2080, Online. Association for Computational Linguistics.

Tristan Thrush, Ryan Jiang, Max Bartolo, Amanpreet Singh, Adina Williams, Douwe Kiela, and Candace Ross. 2022. Winoground: Probing vision and language models for visio-linguistic compositionality. *arXiv preprint arXiv:2204.03162*.

Maria Tsimpoukelli, Jacob Menick, Serkan Cabi, SM Eslami, Oriol Vinyals, and Felix Hill. 2021. Multimodal few-shot learning with frozen language models. *Proc. Neural Information Processing Systems*.

Alex Wang, Yada Pruksachatkun, Nikita Nangia, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel Bowman. 2019. Superglue: A stickier benchmark for general-purpose language understanding systems. In *Advances in Neural Information Processing Systems*, volume 32. Curran Associates, Inc.

Alex Wang, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel Bowman. 2018. GLUE: A multi-task benchmark and analysis platform for natural language understanding. In *Proceedings of the 2018 EMNLP Workshop BlackboxNLP: Analyzing and Interpreting Neural Networks for NLP*, pages 353–355, Brussels, Belgium. Association for Computational Linguistics.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Remi Louf, Morgan Funtowicz, Joe Davison, Sam Shleifer, Patrick von Platen, Clara Ma, Yacine Jernite, Julien Plu, Canwen Xu, Teven Le Scao, Sylvain Gugger, Mariama Drame, Quentin Lhoest, and Alexander Rush. 2020. Transformers: State-of-the-art natural language processing. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 38–45, Online. Association for Computational Linguistics.

Tian Yun, Chen Sun, and Ellie Pavlick. 2021. Does vision-and-language pretraining improve lexical

grounding? In *Findings of the Association for Computational Linguistics: EMNLP 2021*, pages 4357–4366, Punta Cana, Dominican Republic. Association for Computational Linguistics.

Yuke Zhu, Oliver Groth, Michael Bernstein, and Li Fei-Fei. 2016. Visual7w: Grounded question answering in images. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 4995–5004.

A Datasets for training and tuning

More detailed information on the datasets used for training and fine-tuning the models investigated in this work can be found here.

A.1 Training of BERT baselines

More information about the LXMERT and Wikipedia training datasets used for the BERT-base baselines can be found in Table 5. By training the BERT model on LXMERT text data, it will have seen the same textual information as LXMERT. And by training it on the Wikipedia data, it will have seen the same amount of text as LXMERT.

A.2 Fine-tuning model on text-only input

The LXMERT and Wikipedia datasets used for fine-tuning text-only versions of VL models are further described in Table 6. The two fine-tuning sets cover quite different domains. This is already visible from the tokens/sample count in the table, in which the Wikipedia corpus generally contains long sentences and the LXMERT corpus generally contains shorter sentences more suitable for image captions.

B Training procedures

More detailed information on our training procedures can be found here.

B.1 Training BERT-base baselines

For the training of BERT-base on both the LXMERT and Wikipedia datasets we use an MLM objective, a batch size of 16,384 and learning rate 5×10^{-5} until the model performance on the dev set had converged. The maximum training time was at most 23 hours on 32 Tesla T4 GPUs.

B.2 Fine-tuning model on text-only input

We fine-tune the models using an MLM objective, batch size of 256 and learning rate 5×10^{-5} until the model performance on the dev set had converged. The maximum training time was two hours on eight Tesla T4 GPUs.

B.3 Using tuned visual features

We tune the visual features using an MLM objective, batch size 64 and a learning rate of 0.05 until the model performance had converged on the dev set. The maximum training time was 18 hours on one Tesla T4 GPU.

B.4 Fine-tuning on GLUE

For the GLUE fine-tuning, we tune our models for four epochs with a learning rate of 3×10^{-5}, weight decay of 0.01 and batch size of 32. The longest tuning time was four hours on two A100 GPUs. We then pick the model checkpoint with the best validation score during training for evaluation.

C Complete numerical results

The complete numerical results on GLUE and VPN can be viewed in Tables 7 and 8 respectively.

Corpus	Partition	# of samples	# of tokens	# of tokens/sample
LXMERT	train	9.0M	59.0M	6.6
	dev	0.2M	1.4M	6.8
Wikipedia	train	4.4M	59.0M	13.4
	dev	0.1M	1.3M	13.4

Table 5: The two text datasets used for developing two additional BERT-base baselines. The number of samples are roughly equal to the number of sentences for these datasets. The LXMERT data is the text part of the LXMERT training data. Wikipedia is a subset of general English Wikipedia texts that has been adapted to match the LXMERT data in total number of tokens.

Corpus	Partition	# of samples	# of tokens	# of tokens/sample
LXMERT-f	train	9,500	63,000	6.6
	dev	3,300	22,000	6.6
Wikipedia-f	train	4,600	63,000	13.7
	dev	1,600	22,000	13.5

Table 6: The two text datasets used for fine-tuning, denoted by the "-f" ending. The number of samples are roughly equal to the number of sentences for these datasets.

Model	Adaptation	Score
BERT-base	trained-LXMERT	80.7
	trained-LXMERT-scratch	64.1
	trained-Wikipedia	81.1
	default	83.7
FLAVA	default	78.8
CLIP-BERT	default	79.6
	no-visual-features-finetuned-LXMERT	79.0
	no-visual-features-finetuned-Wikipedia	79.7
	avg-visual-features	79.8
	zero-image-visual-features	79.4
	zeroed-visual-features	79.7
	finetuned-LXMERT-visual-features	79.5
	finetuned-Wikipedia-visual-features	79.6
	imagined-visual-features	79.6
LXMERT	default	61.9
	no-visual-features-finetuned-LXMERT	59.7
	no-visual-features-finetuned-Wikipedia	61.9
	avg-visual-features	61.3
	zero-image-visual-features	59.9
	zeroed-visual-features	61.8
	finetuned-LXMERT-visual-features	61.5
	finetuned-Wikipedia-visual-features	61.6
VisualBERT	default	80.6
	no-visual-features-finetuned-LXMERT	80.1
	no-visual-features-finetuned-Wikipedia	81.3
	avg-visual-features	80.9
	zero-image-visual-features	80.6
	zeroed-visual-features	79.0
	finetuned-LXMERT-visual-features	79.9
	finetuned-Wikipedia-visual-features	80.5

Table 7: The adaptation and baseline results for GLUE seen in Figure 2a.

Model	Adaptation	Score Median	Standard deviation
BERT-base	trained-LXMERT	49.1	13.2
	trained-LXMERT-scratch	44.3	13.2
	trained-Wikipedia	35.5	9.5
	default	39.0	12.4
FLAVA	default	30.7	6.9
CLIP-BERT	default	44.3	4.7
	no-visual-features-finetuned-LXMERT	44.5	7.0
	no-visual-features-finetuned-Wikipedia	41.5	5.4
	avg-visual-features	33.1	5.0
	zero-image-visual-features	41.8	5.8
	zeroed-visual-features	39.3	4.9
	finetuned-LXMERT-visual-features	48.2	4.2
	finetuned-Wikipedia-visual-features	33.3	4.8
	imagined-visual-features	31.4	10.2
LXMERT	default	42.8	10.8
	no-visual-features-finetuned-LXMERT	41.0	7.5
	no-visual-features-finetuned-Wikipedia	34.6	10.0
	avg-visual-features	37.3	12.9
	zero-image-visual-features	42.1	11.0
	zeroed-visual-features	35.2	12.0
	finetuned-LXMERT-visual-features	37.2	13.9
	finetuned-Wikipedia-visual-features	28.5	11.7
VisualBERT	default	29.0	10.9
	no-visual-features-finetuned-LXMERT	38.1	9.1
	no-visual-features-finetuned-Wikipedia	21.6	9.0
	avg-visual-features	29.8	11.0
	zero-image-visual-features	25.6	10.2
	zeroed-visual-features	7.1	3.1
	finetuned-LXMERT-visual-features	34.5	10.3
	finetuned-Wikipedia-visual-features	20.1	9.9

Table 8: The adaptation and baseline results for VPN.

ACT-Thor: A Controlled Benchmark for Embodied Action Understanding in Simulated Environments

Michael Hanna[*†]
University of Amsterdam
m.w.hanna@uva.nl

Federico Pedeni[*]
University of Trento
federico.pedeni@studenti.unitn.it

Alessandro Suglia
Heriot-Watt University
a.suglia@hw.ac.uk

Alberto Testoni
University of Trento
alberto.testoni@unitn.it

Raffaella Bernardi
University of Trento
raffaella.bernardi@unitn.it

Abstract

Artificial agents are nowadays challenged to perform embodied AI tasks. To succeed, agents must understand the meaning of verbs and how their corresponding actions transform the surrounding world. In this work, we propose ACT-Thor, a novel controlled benchmark for embodied action understanding. We use the AI2-THOR simulated environment to produce a controlled setup in which an agent, given a before-image and an associated action command, has to determine what the correct after-image is among a set of possible candidates. First, we assess the feasibility of the task via a human evaluation that resulted in 81.4% accuracy, and very high inter-annotator agreement (84.9%). Second, we design both unimodal and multimodal baselines, using state-of-the-art visual feature extractors. Our evaluation and error analysis suggest that only models that have a very structured representation of the actions together with powerful visual features can perform well on the task. However, they still fall behind human performance in a zero-shot scenario where the model is exposed to unseen (action, object) pairs. This paves the way for a systematic way of evaluating embodied AI agents that understand grounded actions.

1 Introduction

Recently, embodied agents have been increasingly proposed and evaluated on their capacity to navigate virtual environments, and execute actions within them according to instructions (Suhr et al., 2019; Hahn et al., 2020; Shridhar et al., 2020; Padmakumar et al., 2022). Action execution as a means of evaluating language understanding is a long-standing idea (e.g. (Winograd, 1972)) which has now become feasible thanks to the release of various virtual environment platforms (Kolve et al., 2017; Savva et al., 2019; Yan et al., 2018). Thus, the study of embodied agents is an interesting venue to push forward due its both theoretical and practical appeal. However, fully embodied tasks involve a variety of issues, making it hard to understand where we stand with respect to the challenges posed. Here, we focus on one core aspect of such embodied tasks: action grounding.

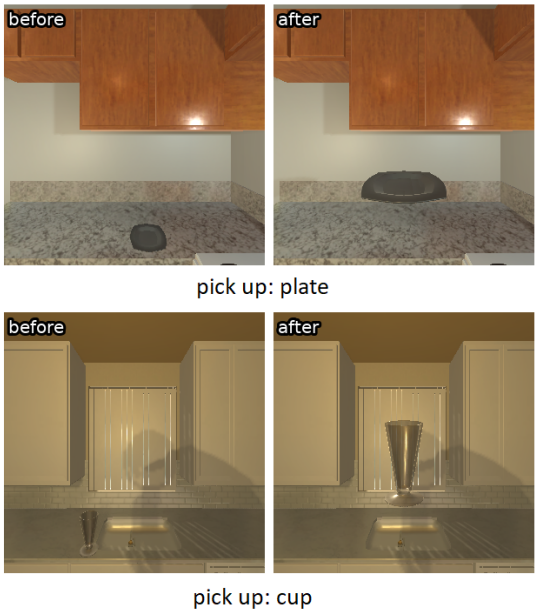

Figure 1: The image on the left (before-image) is the view the robot has while holding a plate or a cup; the image on the right is the view the robot has after having executed the command of picking up that plate/cup.

Tamari et al. (2020) advocate for embodied language understanding as a necessary step to overcome the limitations of the mainstream deep-learning paradigm. Among the priorities of their agenda is evaluating models' ability to simulate the effects of actions. Evidence from neuroscience favors an embodied meaning of words and specifically of verbs; for instance, the natural theory of

[*]Equal contribution
[†]Work performed while at the University of Trento

Proceedings of the 29th International Conference on Computational Linguistics, pages 5597–5612
October 12–17, 2022.

language (Feldman and Narayanan, 2004) is based on the assumption that people understand language by imagining (or simulating) the situation being described.

In this paper, we focus on grounded verb understanding and predicting the visual transformation their corresponding action causes. More specifically, we consider the idea of verbs as functions (Frege, 1892), and see them as transformations of an object within a scene before and after the verb/action is executed. To achieve this, we create a dataset combining ideas and methods from two main papers about learning physical commonsense knowledge (Gao et al., 2018; Zellers et al., 2021). As in Gao et al. (2018), we focus on the action-effect relation, namely the physical change that an action causes when executed on an object. And, as in Zellers et al. (2021), we leverage a virtual environment, AI2-THOR (Kolve et al., 2017), to generate transitions between object states.

In sum, we propose `ACT-Thor`, a dataset for learning the effects of actions, generated using a virtual environment.[1] For each example in our dataset, we collect an image of an object, a label describing an action performed on the object, and an after-image resulting from executing the action (Figure 1). Thus, unlike in Gao et al. (2018) and Zellers et al. (2021), both the before and after world states are images; the linking action acts as a function transforming the former into the latter. Then, via the use of images whose generation is carefully controlled, we design a task that intends to capture a challenge embodied AI agents will face in a real setting.

Out of this collection of image pairs and their corresponding action-label, we design a prediction task based on carefully selected contrast sets containing candidate images representing the effect of other actions performed on the very same object in the same scene (Figure 2). Agents that truly ground verbs, given the before-image and the action label, should be able to select the after-image out of the contrast set. We also evaluate human performance on a sample of the dataset. Annotators achieve high accuracy and inter-annotator agreement, evidencing the quality of the design method we propose, and setting an upper bound for multimodal models.

We evaluate baseline models based on state-of-the-art visual features (Singh et al., 2021; Radford et al., 2021). We put together ideas and results from different disciplines to shed light on different ways to study action in neural models and pave the way for future research on action grounding. From formal semantics and computational semantics, we take the view of verbs as functions (Frege, 1892), and therefore as matrices (Baroni and Zamparelli, 2010) and learn, for each action, a matrix able to transform the before-image, into the corresponding after-image. We compare this model with multimodal neural network models of increasing complexity. Our experiments show that

- virtual environment platforms can be used to generate interesting diagnostic datasets of static images having certain properties and satisfying certain constraints;
- the task we propose is easy for humans despite the use of virtual environments they are not familiar with;
- though the baseline models we have evaluated succeed on the task when evaluated on unseen scenes, they have difficulties when generalizing the learned action transformations to unseen objects.

2 Related Work

Actions/verbs have been studied from a variety of perspectives. They have been the subject of thorough investigation in formal semantics (Fillmore, 1982; Beth, 1993), and have been the focus of widely-used annotated resources (Fillmore et al., 2002; Schuler, 2005).

Multimodal approaches to action/verb learning have been especially fruitful. Misra et al. (2015) learn to ground visual representations of actions to formal semantic representations thereof. Prior studies have connected visual action representations to concepts such as intention (Pezzelle et al., 2020; Ignat et al., 2021) and causality (Gao et al., 2016; She and Chai, 2017; Yoo et al., 2021). Moreover, text-based models of similarity between actions have been shown to substantially improve when text is combined with visual information (Regneri et al., 2013). The breadth of these works suggests the importance of verbs and actions for the development of embodied agents and natural language understanding more broadly. We contribute to this line of research by presenting a new technique for creating datasets that evaluate computational models

[1] The dataset and all relevant code are available at `https://github.com/hannamw/ACT-Thor`.

for action grounding. We build on ideas introduced in (Gao et al., 2018; Wang et al., 2016) by leveraging a virtual environment platform: rather than filtering and annotating images from the web via crowdsourcing, we generate a controlled dataset using an agent in a virtual environment.

Within the NLP community, predicting the consequences of actions is often framed as part of learning commonsense knowledge. We share with Gao et al. (2018) a focus on what they call *naive* physical action-effect relations, namely the causal relation between an action (verb) and the change in physical world state caused by that action. Similarly, we propose an "effect prediction" task in which the model predicts an effect (an image) given the action and the object on which the action is performed. In their task, the action and object pair are expressed as verbs and nouns, whereas in our case the noun is presented as a (before) image; hence the model is challenged to learn the visual transformation objects undergo through the action execution.

Zellers et al. (2021) use AI2-THOR to build the PIGPeN dataset, from which models can learn world dynamics. They do so by letting a virtual agent navigate and interact with its environment to solve certain tasks. However, when building these trajectories, the robot's visual perspective (an image) is ignored in favor of a symbolic representation of the world state (i.e., as a set of visual attributes). Hence, models trained on PIGPeN receive symbolic and natural language inputs representing world information, and produce outputs in the same modality. To achieve our goal, we propose studying the physical action-effect relation highlighted by Gao et al. (2018), utilizing the image generation approach used in Zellers et al. (2021).

Finally, as in (Gao et al., 2018), we implement this concept as a classification task. However, inspired by Gardner et al. (2020), we adopt a contrastive setup. Our use of a virtual environment and its metadata allows us to build carefully controlled sets of candidates in which the very same object in the very same scene depicted in the before-image has undergone different visual transformations/different actions.

Statistic	ACT-Thor
action-object pairs	403
before-i, action, after-i	11154
unique before-i	3746
unique after-i	11154
scenes	120
objects	62
actions	12

Table 1: Statistics of the ACT-Thor collection.

3 Datasets

To build our dataset, we generated images using AI2-THOR[2], the virtual environment platform introduced in (Kolve et al., 2017). In AI2-THOR, the user controls a robot as it navigates a room of a virtual house environment, filled with household objects. While navigating the rooms, the robot can interact with objects it encounters by performing actions on them. Position and state metadata are available for the robot and all objects in the scene.

We use the robot to generate an image representing a scene before and after an action is taken. In the following section, we describe the image generation process, which led to the collection of triples consisting of before and after-images and their linking action. Then, we describe ACT-Thor, a dataset built out of this collection to evaluate models' abilities to simulate the consequences of actions on objects situated in virtual scenes.

3.1 Image Generation

AI2-THOR provides a rich environment for image generation. It allows an agent to traverse 120 distinct virtual rooms (scenes), equally distributed among the following four categories: kitchen, bathroom, living room, or bedroom. These scenes contain 125 distinct objects, from large, static objects such as desks or sinks, to smaller objects like pots or cups. Crucially for the creation of an action-oriented dataset, the robot can execute 24 actions, 21 of which act directly on objects.

In this virtual world, as in reality, each object has a certain number of affordances (Gibson, 1977), in other words, not all actions can be performed on all objects. Moreover, the state of an object affects these affordances as well; for example, objects can only be put down or thrown if they are currently held. We are interested in studying actions as visual

[2]We used version v4.2.0 of AI2-THOR, https://ai2thor.allenai.org/.

transformations, but not all actions result in clearly visible changes. A preliminary analysis showed that *pickupable* objects, those that can be picked up, could be the target of a greater variety of actions compared to static objects, or objects that could be moved but not picked up. Thus, we focused on these, as our dataset requires objects to be the targets of multiple, diverse actions.

Selecting pickupable objects and filtering out the rest, we consider 62 object types across all 120 scenes; each object type occurs in on average 28.5 scenes (SD: 14.65). Of the 21 actions affecting objects, we focused on 15 actions. We prioritized actions that cause visible state changes in their target objects, while also including some that physically move objects (some of the physical actions are similar in character, and not very visually distinct).

Based on these observations, our image generation procedure was as follows. For each of AI2-THOR's scenes, we generated the list of *pickupable* objects by selecting those on which the action "pick up" could be executed and identified a surface on which objects could be placed. Then for each scene, we performed two sets of trials with each pickupable object at the predetermined location. In the first set of trials, the object started in the robot's grasp (held object) while in the second set, the object started on the surface (placed object). For each object and set of trials, one before-image was recorded, and for each action performed, one after-image was recorded.

In the first trial (held object), all objects could have at least 3 distinct actions performed on them: they could all be dropped, thrown, and put down. Some objects permitted additional actions: dirtying, toggling, filling, breaking, cooking, and opening. In the second trial (placed object), all objects could be pushed, pulled, and picked up. Again, some objects allowed additional actions, the same as for held objects, plus slicing. We executed all possible actions for each object to generate the after-images, making sure that every before-action-after triple in a given scene takes place in the same location. This approach yielded 13958 before-action-after triples (6979 for each trial). There were 412 object-action pairs, across 15 actions and 62 objects, across all 120 scenes.

We then cleaned the data, removing from the dataset any triples whose associated actions failed; for example, placing an object can fail if there is no space in which to place it. We also removed any triples where the object was no longer visible in the after-image, as well as those whose after-image was not visibly different from the before-image, as determined by pixel-wise distance between images. Finally, we removed actions that could be executed on only one object. This filtering removed 3 actions: "use up", "slice", and "cook".

As summarized in Table 1, after filtering, our collection contained 11154 triples, built out of 12 actions, 62 object types, and 403 action-object pairs, across 120 scenes. It contained in total 3746 unique before-images and 11154 unique after-images.[3] Note also that while this collection is not large, it can also be expanded, as it was collected from a virtual environment. Specifically, by randomizing object color, scene lighting, agent perspective, and other aspects of the virtual environment, it would be possible to increase visual diversity while not compromising the controlled nature of the image generation.

In the next section, we focus on building a dataset for a contrast set task; however, we note that this collection could be assembled into datasets for other tasks as well. For example, an *action inference* task could be created by presenting a matching before- and after-image, and training a model to predict the action connecting the two images.

3.2 Dataset Creation

We converted our collection of triples described above into contrast sets consisting of 4 images. For each (before-image, action, after-image) triple, the corresponding contrast set consists of the true after-image, and 2 after-images generated starting from the same before-image by executing different actions on the same object. In addition, we include 1 image that is the result of executing the same action on the same object type situated in a scene distinct from the one depicted in the before-image. The inclusion of this image ensures that a model cannot simply pick images that are consistent with the given action; they must also correspond to the scene of the before-image. Below we provide the details of the dataset creation to highlight its fine-grained design.

Some before-images had fewer than 3 after-images (same object, same scene, different action); thus, triples containing this before-image could not

[3] The list of actions, together with the number of object types on which they have been executed, is provided in the Supplementary Material (Table 5)

Action	# Objs.	Action	# Objs.
Push	20	Pull	20
Pick Up	20	Put Down	20
Throw	20	Drop	20
Break	11	Fill	7
Dirty	7	Toggle	3
Open	2	Close	2

Table 2: Task: Number of object types per action (before-image-action pair) in the filtered contrast set dataset.

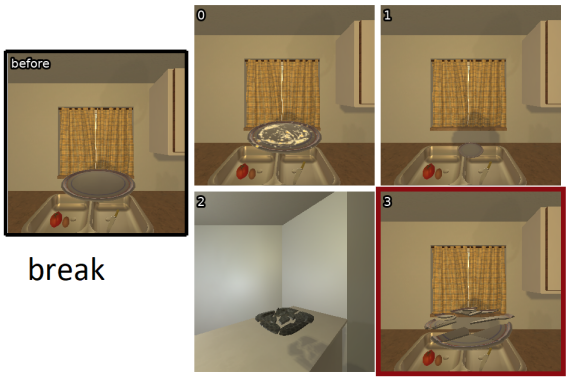

Figure 2: Simulation task: given a before-image and an action label, select from a contrast image set the one that represents the action simulation (after-image).

be converted into contrast sets. We also discarded examples where certain action pairs occur in the same contrast set, as their outcomes are too visually similar. For example, the outcome of putting any given object down is not consistently distinguishable from the outcome of throwing it; similarly, pushing and pulling objects can be visually similar. This particular filtering step results in significant cuts to the dataset size. This approach yields 4451 datapoints, 152 object-action pairs over 12 actions and 20 objects, across 109 scenes. The object-action distribution is given in Table 2.

Task Figure 2 illustrates the task: given the before-image (the robot is in a kitchen, holding a plate) and the action label, the model has to select the corresponding after-image (image 3) from the given contrast set, whose candidates are generated from the before-image by having the robot dirty (image 0), throw (image 1), and break (image 3) the plate. Finally, image 2 results from the robot breaking the plate in a different kitchen.

Human experiment To determine how challenging this task was for humans, we also solicited annotations from students. Each of the 7 annotators was randomly assigned 60 contrast sets to annotate using the `makesense.ai` platform. Annotators saw images like Figure 2 but without the red box indicating the correct response. They were told to select the index of the image corresponding to performing the listed action on the object in the before-image.

As each contrast set was annotated by two annotators, this yielded annotations for 210 contrast sets. Overall, annotators achieved high accuracy (81.4%) and annotator agreement (84.9%), compared to random chance (25% for both). This evidences the quality of the data and generation process, as well as the feasibility of the task.

4 Models

In this section, we describe baseline models designed to assess the role of the embeddings involved: those of the action-label, before- and after-images. We compare models using frozen visual features extracted from various state-of-the-art feature extractors.[4]

Training method Considering the contrastive nature of our task, inspired by CLIP's loss function (Radford et al., 2021), we use the InfoNCE loss proposed by van den Oord et al. (2018). This loss minimizes the distance with the ground-truth vector while jointly maximizing distances from the "negative" elements in the contrast set (i.e. the ones generated from different actions). Let $C = (\{\mathbf{c}_1, \mathbf{c}_2, ..., \mathbf{c}_n\}, k)$ be a contrast set where $\mathbf{c}_i$ are the vector representations for each image, and k is the index of the true after-image. Furthermore, let $\tilde{\mathbf{a}}$ be the predicted after-image representation. Then we compute the probability that image i is the true after-image as follows:

$$p(i|C, \tilde{\mathbf{a}}) = \frac{e^{\mathbf{c}_i \cdot \tilde{\mathbf{a}}^\top}}{\sum_{j=1}^{n} e^{\mathbf{c}_j \cdot \tilde{\mathbf{a}}^\top}}$$

The training loss is $\mathcal{L}(C, \tilde{\mathbf{a}}) = -\log(p(k|C, \tilde{\mathbf{a}}))$, cross-entropy loss. At test time, our model predicts the true after -mage index as $\text{argmax}_i\, p(i|C, \tilde{\mathbf{a}})$. The models presented in the following paragraphs all use the approach described above, with the exception of the `visual-only` model, for which there is no training time.

[4]Details regarding model architectures and parameter counts can be found in the Supplementary Material (Figure 6, Table 8)

Unimodal Baselines Inspired by Thomason et al. (2019), we verify the impact of visual and language input by considering unimodal baselines. First, we evaluate a model that learns to select as after-image an image of the contrast set whose visual vector is closer to the before-image, based on cosine similarity (Hence, `visual-only`.) In this work, we explore two different visual extractors: 1) *MOCA*: we rely on the visual encoder pretrained using ALFRED data by Singh et al. (2021); 2) *CLIP*: we use the visual features extracted by the CLIP image encoder (Radford et al., 2021). We use the former because it has been extensively used for other Embodied AI tasks (e.g., (Pashevich et al., 2021)) while we use the latter because Khandelwal et al. (2022) showed that CLIP provides very effective representations for Embodied AI tasks.

Second, we evaluate a model that learns to select the after-image simply based on the action name (`action-name`). The model architecture is simple: first, the action label is converted into a (learned) embedding. Next, each of the after-images is turned into a feature vector using one of the aforementioned frozen pre-trained feature extractors. Then, both the action embedding and each of the after-image feature vectors are projected into a shared space via linear projections. The projected action embedding is taken to be $\tilde{\mathbf{a}}$, and the appropriate projected image representations to be $\mathbf{c}_1, \ldots, \mathbf{c}_n$. Then training and prediction are performed as described above.

Multimodal Baselines We design three multimodal baseline models. The first of them departs from the mainstream end-to-end approach used in Embodied AI, implementing the formal semantics view of "verbs as functions". Following Baroni and Zamparelli (2010), we represent each action as a matrix that is learned from its before- and after-images. Unlike Baroni and Zamparelli (2010), we rely on a feed-forward network with a single linear layer, with no bias vector or activation function (hence, `Action-Matrix`). Given the before-image representation (derived from either MOCA or CLIP), we use the action identifier to extract the associated *action matrix*, which is composed with the before-image vector through matrix multiplication to return the predicted after-image vector.[5]

In addition, we design two models trained end-to-end to select the after-image given the before-image representation and the action name represented as a vector that belongs to a learned embedding matrix. The two variants of this architecture are: `Concat-Linear`: a single matrix, acting as a linear layer, which is multiplied by the concatenation of *before* vector and action vector; and `Concat-Multi`: a 2-layer, feed-forward network, including batch-normalization, dropout and activation layers, taking as input the same concatenated vector as above.

5 Experiments

It is important that embodied AI agents be capable of making sound predictions in unseen scenarios is a very important requirement for real-world applications. For this reason, we carefully design two hold-out procedures when splitting the dataset into train, validation and test data, in order to estimate the generalization capabilities of the baselines

Unseen Scenes. Following the standard practice in AI2-THOR[6], we split the set of scenes $\mathbb{S}$ into *seen* scenes $\mathbb{S}_s$ and *unseen* scenes $\mathbb{S}_u$ and use the former to create the training and validation sets (2856 and 684 datapoints, resp.) and the latter for the test set (820 datapoints); in other words, at test time the models will receive images illustrating one of the four possible types of virtual rooms (e.g. a bathroom) in a configuration that they have not seen during training/validation.[7] This method allows us to test for generalization at the scene level, in order to see if the models' visual features are fine-grained enough to generalize across different configurations of objects appearing in different rooms.

Unseen Objects. Given the set of object classes $\mathbb{O}$, we split it into a *seen* subset $\mathbb{O}_s$, and an *unseen* subset $\mathbb{O}_u$. We define the training and validation splits so that they contain objects that belong to $\mathbb{O}_s$ only. On the other hand, we define the test split so that it contains only objects belonging to $\mathbb{O}_u$. We manually selected the unseen objects (*laptop*, *cup*, and *book*) so as to guarantee the complete coverage of actions at test time. The test set contains 908 datapoints, the training and validation sets amount to 2762 and 690 datapoints, respectively.[8] This

[5]The model we implemented is quite close to the Siamese Network proposed in (Wang et al., 2016) which we discovered while writing the paper. The code is not available; hence, we could not evaluate it.

[6]The standard practice in AI2-THOR is reported in the official documentation.

[7]Detailed statistics in Supplementary Material (Table 7)

[8]Detailed statistics in Supplementary Material (Table 6)

procedure will help showing how well models can understand the effect of known actions on novel objects.

6 Results

6.1 Accuracy

As illustrated in Figure 3, the unimodal baseline models are close to chance levels both in the unseen scenes and unseen objects setting. In both cases, the visual-only model performs better than the action-name one, independently of the visual features used. In general, for the visual-only baseline, CLIP features seem to be better than MOCA ones, possibly due to their higher expressiveness when used for visual similarity, as is the case here. The drop in performance for the model `action-name` on unseen scenes is justified by the fact that the model is learning to make predictions based on the action embeddings only. At training time, the model is exposed to seen scenes only; therefore, the action embeddings are likely to encode patterns associated with these scenes only. Such representations do not truly consider other context-specific information and therefore do not allow the model to generalize well on unseen scenes.

In both splits, all models perform better when based on the MOCA visual features than CLIP. In particular, MOCA-based models generalize better across scenes. This could be associated with the fact that our task requires very fine-grained information that a CLIP model may not be able to retain. Thanks to the embodied nature of the task solved by MOCA, its visual features learn more fine-grained representations of the AI2-THOR environment.

The `Action Matrix` and `Concat-Multi` models reach a similar accuracy in both settings; however, the latter suffers less from the coarser-grained representation provided by the CLIP features in the holding out scene setup. When comparing model and human performance, it is clear that the weak point of these models is the crucial ability to generalize their action grounding to unseen objects. Although the images are not fully naturalistic, humans reach 83% accuracy, whereas the best models are around 70% in the unseen objects split when given the MOCA features that are specifically fine-tuned on AI2-THOR, and lower than 60% with CLIP.

Another important consideration is that, following previous work on affordance prediction (Deng et al., 2021), our baselines benefit from the gold information of the action being performed. The `Action-Matrix` model is designed to learn a transformation matrix for each action in the dataset as in Wang et al. (2016). This demonstrates that having a way to learn "verb" related information is useful to match human performance and generalize better. We believe that this represents a very interesting challenge for embodied AI models that are fully end-to-end (e.g. (Suglia et al., 2021)). Introducing models that can reason over time by "simulating" the action being performed represents a potentially interesting avenue for future work.

6.2 Error Analysis

In order to understand models' performance, we completed a detailed analysis of the best performing models, `Action Matrix` and `Contact-Multi`, focusing on the unseen object split. First, we computed accuracy by action. Looking at Table 3, the most successful action is by far "drop" for all models: this could be due to the change of perspective that happens after dropping an object. Moreover, this action can be applied to all the objects in the dataset, thus we can assume that the model has more data to learn from and generalizes better.

While the MOCA-based `Concat-Multi` model holds the majority of the best values, it can be noted that the CLIP-based version scores better on actions for which there are fewer datapoints in the training set ("close", "open", "toggle"). We hypothesize this is due to the larger pretraining dataset that is used by CLIP, whose higher capacity might lead to better generalization with less data.

Second, we computed the accuracy by objects. This analysis shows that of the three objects unseen during training, "cup" is the one with highest scores across actions. We hypothesize that this is due to the presence of other visually similar objects in the training set, for instance "mug" and "bowl".[9]

Finally, we exploited the directly interpretable action representations computed by the `Action Matrix` MOCA-based model, and computed the nearest neighbors of each verb. As reported in Table 4, the model groups together actions that cause a change of position (e.g., "drop", "push", "pull"), and actions that cause a change in object appearance (e.g. "break", "dirty", "toggle".) This is a first promising step towards grounded verb

[9] Details in Table 9 of the Supplementary Material

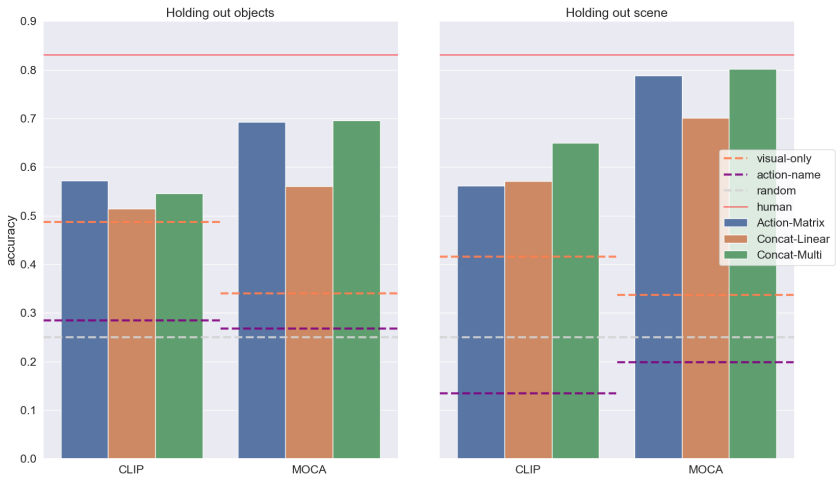

Figure 3: Results for contrast-task accuracy, training with contrastive-loss (average over 5 random initializations). Dashed lines represent baselines: gray for chance, orange for visual-similarity, purple for the action-name after-image-only model. The red line represents human accuracy averaged over all procedures.

Model	break	close	dirty	drop	fill	open	pickUp	pull	push	put	throw	toggle
AM-CLIP	0.406	0.516	0.553	0.907	0.495	0.348	0.540	0.439	0.572	0.628	0.619	0.447
CM-CLIP	0.576	**0.573**	0.502	0.627	0.546	**0.722**	0.593	0.521	0.546	0.607	0.622	**0.630**
AM-MOCA	0.580	0.541	0.678	0.991	0.814	0.409	0.772	0.449	**0.675**	0.782	**0.769**	0.567
CM-MOCA	**0.651**	0.482	**0.719**	**0.964**	**0.848**	0.426	**0.805**	**0.536**	0.668	**0.855**	0.670	0.573

Table 3: Accuracy per action of the best scoring models; results for the test set of the **object split**. Best values on columns in bold.

Action	Nearest neighbors (sorted)		
break	dirty	open	toggle
close	break	dirty	put
dirty	break	pull	open
drop	push	pull	pickUp
fill	put	pull	throw
open	dirty	break	fill
pickUp	dirty	fill	break
pull	put	push	throw
push	pull	throw	put
put	throw	pull	push
throw	put	push	pull
toggle	dirty	break	pull

Table 4: 3 nearest action neighbor actions for each action, using action representations from the MOCA Action-Matrix model (trained on the object split).

representations. We believe much could be learned by combining formal semantics findings on verbs and the embodied AI literature.

7 Conclusion

In this work, we define `ACT-Thor`, a controlled benchmark for embodied action understanding in simulated environments. Compared to similar benchmarks presented in the literature, we propose a more systematic benchmark for studying the capability of multimodal machine learning models to understand the effects of actions on the objects. Additionally, our contrast set formulation allows us to precisely pinpoint the distinctions that models should learn, providing a robust and reliable benchmark for studying action grounding.

To assess the quality of our dataset, we first completed a human evaluation demonstrating that humans can complete this task with 83% accuracy. Then, we evaluated several unimodal models, and multimodal models using several state-of-the-art visual feature extractors such as CLIP (Radford et al., 2021). By inspecting models that learn a matrix for each action, we show that the representations learned via our dataset favor the emergence of clusters of actions associated with a change of position, and actions that cause a change in object appearance. These are two salient visual transformations objects undergo, but more could be learned by extending the dataset with other actions or other objects, and by proposing controlled settings inspired by the theoretical view on verb semantics.

Our work demonstrates the potential of virtual environments to pursue this line of research.

We argue that `ACT-Thor` represents an experimental benchmark for studying the action understanding capabilities of machine learning models, an important skill for embodied AI agents. This is especially important moving forward for studying the capabilities of generalist agents that can solve multiple tasks (Reed et al., 2022).

Acknowledgments

Michael Hanna thanks the Erasmus Mundus European Masters Program in Language and Communication Technologies for its support.

References

Marco Baroni and Roberto Zamparelli. 2010. Nouns are vectors, adjectives are matrices: Representing adjective-noun constructions in semantic space. In *Proceedings of the 2010 conference on empirical methods in natural language processing*, pages 1183–1193.

Levin Beth. 1993. *English Verb Classes and Alternations. A Preliminary Investigation.* University of Chicago Press.

Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. 2009. Imagenet: A large-scale hierarchical image database. In *2009 IEEE conference on computer vision and pattern recognition*, pages 248–255. Ieee.

Shengheng Deng, Xun Xu, Chaozheng Wu, Ke Chen, and Kui Jia. 2021. 3d affordancenet: A benchmark for visual object affordance understanding. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 1778–1787.

J Feldman and S. Narayanan. 2004. Embodied meaning in a neural theory of language. *Brain and Language*, 89(2):385–392.

Charles J. Fillmore. 1982. *Frame semantics*, pages 111–137. Hanshin Publishing Co., Seoul, South Korea.

Charles J. Fillmore, Collin F. Baker, and Hiroaki Sato. 2002. The FrameNet database and software tools. In *Proceedings of the Third International Conference on Languag Resources and Evaluation*, volume IV, Las Palmas. LREC, LREC.

G. Frege. 1892. Uber sinn und bedeutung. In *Zeitschrift fuer Philosophie un philosophische Kritik*.

Qiaozi Gao, Malcolm Doering, Shaohua Yang, and Joyce Chai. 2016. Physical causality of action verbs in grounded language understanding. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1814–1824, Berlin, Germany. Association for Computational Linguistics.

Qiaozi Gao, Shaohua Yang, Joyce Chai, and Lucy Vanderwende. 2018. What action causes this? towards naive physical action-effect prediction. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 934–945, Melbourne, Australia. Association for Computational Linguistics.

Matt Gardner, Yoav Artzi, Victoria Basmov, Jonathan Berant, Ben Bogin, Sihao Chen, Pradeep Dasigi, Dheeru Dua, Yanai Elazar, Ananth Gottumukkala, Nitish Gupta, Hannaneh Hajishirzi, Gabriel Ilharco, Daniel Khashabi, Kevin Lin, Jiangming Liu, Nelson F. Liu, Phoebe Mulcaire, Qiang Ning, Sameer Singh, Noah A. Smith, Sanjay Subramanian, Reut Tsarfaty, Eric Wallace, Ally Zhang, and Ben Zhou. 2020. Evaluating models' local decision boundaries via contrast sets. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 1307–1323, Online. Association for Computational Linguistics.

James J Gibson. 1977. The theory of affordances. *Hilldale, USA*, 1(2):67–82.

Meera Hahn, Jacob Krantz, Dhruv Batra, Devi Parikh, James Rehg, Stefan Lee, and Peter Anderson. 2020. Where are you? localization from embodied dialog. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 806–822.

Oana Ignat, Santiago Castro, Hanwen Miao, Weiji Li, and Rada Mihalcea. 2021. WhyAct: Identifying action reasons in lifestyle vlogs. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 4770–4785, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Apoorv Khandelwal, Luca Weihs, Roozbeh Mottaghi, and Aniruddha Kembhavi. 2022. Simple but effective: Clip embeddings for embodied ai. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 14829–14838.

Eric Kolve, Roozbeh Mottaghi, Winson Han, Eli VanderBilt, Luca Weihs, Alvaro Herrasti, Daniel Gordon, Yuke Zhu, Abhinav Kumar Gupta, and Ali Farhadi. 2017. Ai2-thor: An interactive 3d environment for visual ai. *ArXiv*, abs/1712.05474.

Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C. Lawrence Zitnick. 2014. Microsoft coco: Common objects in context. In *Computer Vision – ECCV 2014*, pages 740–755, Cham. Springer International Publishing.

Dipendra Kumar Misra, Kejia Tao, Percy Liang, and Ashutosh Saxena. 2015. Environment-driven lexicon

induction for high-level instructions. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 992–1002, Beijing, China. Association for Computational Linguistics.

Aishwarya Padmakumar, Jesse Thomason, Ayush Shrivastava, Patrick Lange, Anjali Narayan-Chen, Spandana Gella, Robinson Piramithu, and Dilek Hakkani-Tur Gokhan Tur and. 2022. TEACh: Task-driven Embodied Agents that Chat. In *Conference on Artificial Intelligence (AAAI)*.

Alexander Pashevich, Cordelia Schmid, and Chen Sun. 2021. Episodic transformer for vision-and-language navigation. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 15942–15952.

Sandro Pezzelle, Claudio Greco, Greta Gandolfi, Eleonora Gualdoni, and Raffaella Bernardi. 2020. Be Different to Be Better! A Benchmark to Leverage the Complementarity of Language and Vision. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 2751–2767, Online. Association for Computational Linguistics.

Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, Gretchen Krueger, and Ilya Sutskever. 2021. Learning transferable visual models from natural language supervision. In *Proceedings of the 38th International Conference on Machine Learning*, volume 139 of *Proceedings of Machine Learning Research*, pages 8748–8763. PMLR.

Scott Reed, Konrad Zolna, Emilio Parisotto, Sergio Gomez Colmenarejo, Alexander Novikov, Gabriel Barth-Maron, Mai Gimenez, Yury Sulsky, Jackie Kay, Jost Tobias Springenberg, et al. 2022. A generalist agent. *arXiv preprint arXiv:2205.06175*.

Michaela Regneri, Marcus Rohrbach, Dominikus Wetzel, Stefan Thater, Bernt Schiele, and Manfred Pinkal. 2013. Grounding action descriptions in videos. *Transactions of the Association for Computational Linguistics*, 1:25–36.

Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun. 2015. Faster r-cnn: Towards real-time object detection with region proposal networks. In *Advances in Neural Information Processing Systems*, volume 28. Curran Associates, Inc.

Manolis Savva, Abhishek Kadian, Oleksandr Maksymets, Yili Zhao, Erik Wijmans, Bhavana Jain, Julian Straub, Jia Liu, Vladlen Koltun, Jitendra Malik, et al. 2019. Habitat: A platform for embodied ai research. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 9339–9347.

Karin Kipper Schuler. 2005. *VerbNet: A Broad-Coverage, Comprehensive Verb Lexicon.* Ph.D. thesis, PhD Dissertation.

Lanbo She and Joyce Chai. 2017. Interactive learning of grounded verb semantics towards human-robot communication. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1634–1644, Vancouver, Canada. Association for Computational Linguistics.

Mohit Shridhar, Jesse Thomason, Daniel Gordon, Yonatan Bisk, Winson Han, Roozbeh Mottaghi, Luke Zettlemoyer, and Dieter Fox. 2020. ALFRED: A Benchmark for Interpreting Grounded Instructions for Everyday Tasks. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.

Kunal Pratap Singh, Suvaansh Bhambri, Byeonghwi Kim, Roozbeh Mottaghi, and Jonghyun Choi. 2021. Factorizing perception and policy for interactive instruction following. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 1888–1897.

Alessandro Suglia, Qiaozi Gao, Jesse Thomason, Govind Thattai, and Gaurav Sukhatme. 2021. Embodied bert: A transformer model for embodied, language-guided visual task completion. *arXiv preprint arXiv:2108.04927*.

Alane Suhr, Claudia Yan, Jack Schluger, Stanley Yu, Hadi Khader, Marwa Mouallem, Iris Zhang, and Yoav Artzi. 2019. Executing instructions in situated collaborative interactions. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 2119–2130, Hong Kong, China. Association for Computational Linguistics.

Ronen Tamari, Chen Shani, Tom Hope, Miriam R L Petruck, Omri Abend, and Dafna Shahaf. 2020. Language (re)modelling: Towards embodied language understanding. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 6268–6281, Online. Association for Computational Linguistics.

Jesse Thomason, Daniel Gordon, and Yonatan Bisk. 2019. Shifting the baseline: Single modality performance on visual navigation & qa. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 1977–1983.

Aäron van den Oord, Yazhe Li, and Oriol Vinyals. 2018. Representation learning with contrastive predictive coding. *CoRR*, abs/1807.03748.

Xiaolong Wang, Ali Farhadi, and Abhinav Gupta. 2016. Actions ~ transformations. In *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition*, pages 2658–2667.

Terry Winograd. 1972. Understanding natural language. *Cognitive Psychology*, 3:1–191.

Claudia Yan, Dipendra Misra, Andrew Bennnett, Aaron Walsman, Yonatan Bisk, and Yoav Artzi. 2018. Chalet: Cornell house agent learning environment. *arXiv preprint arXiv:1801.07357*.

Hongsang Yoo, Haopeng Li, Qiuhong Ke, Liangchen Liu, and Rui Zhang. 2021. Precondition and effect reasoning for action recognition. *arXiv preprint arXiv:2112.10057*.

Rowan Zellers, Ari Holtzman, Matthew Peters, Roozbeh Mottaghi, Aniruddha Kembhavi, Ali Farhadi, and Yejin Choi. 2021. PIGLeT: Language grounding through neuro-symbolic interaction in a 3D world. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 2040–2050, Online. Association for Computational Linguistics.

A Probing visual features

For the purpose of our experiments, we require visual representations that are expressive and fine-grained so that it is possible to distinguish both object-dependent and scene-dependent features. Therefore, we rely on the object detection task—similar to the one performed in the ImageNet setup (Deng et al., 2009)—as a diagnostic task for the quality of the visual features. Particularly, we assume that high quality visual features will yield high performance in the selected diagnostic task. Furthermore, implementing this sanity check is important because most of our visual feature extractors are pretrained on real-world images, and we do not know how well they will perform on AI2Thor synthetic images.

Borrowing from the literature in Computer Vision, we define our task as the combination of object classification and bounding-box regression: given an input image, the model should produce a tuple (x_0, y_0, x_1, y_1) of coordinates representing the top-left and bottom-right corners of the rectangular bounding-box, where $x_i \in [0, \text{max-image-width}]$ and $y_i \in [0, \text{max-image-height}]$, coupled with the hypothesized object label $l \in \{0, 1, ..., C-1\}$ where C is the number of unique objects in the dataset. Performance over these two tasks can be measured the Mean Average Precision (MAP) metric, which includes information of both intersection-over-union of the predicted boxes with the ground-truth and object class precision.

We use the TorchVision[10] implementation of an end-to-end object detection architecture, Faster RCNN (Ren et al., 2015): this is composed of a CNN backbone, coupled with a Region Proposal Network that gets trained in parallel and two distinct fully connected heads to predict bounding boxes (of size 4) and object class label (of size 62). This architecture is trained to predict both bounding box and object location of the only relevant object in the image, which is the one subject to the current action in the contrast set from which it comes; for the bounding box regression it is used an MSE loss, while for object detection it is used a Cross Entropy loss. In order to match the data distributions of the training set of these models, we normalize AI2Thor images channel-wise, by computing the mean and standard deviation for each channel over the whole dataset. We also crop inputs to size 224×224 for compatibility reasons.
Crucially, for evaluating our models as feature extractors we need to keep the convolutional backbone fixed to its initial weights: in this way, the better the visual features extracted, the easier it will be for the classification head to learn this diagnostic task. Nonetheless, for comparison we train also object detection models where the backbone is fine-tuned, and test if their performance is different from the frozen versions.

The TorchVision reference implementation is based on a ResNet50 backbone, which is the most similar to the models in our experiments, and it reaches a mAP value of 37.0 over COCO val2017 (Lin et al., 2014).

For frozen models, we obtain results that are lower than the baseline. We hypothesize this is due firstly to dataset size, which does not reach hundreds of thousands of samples as for other object detection benchmarks. However, it must be taken into account also the difference in our object detection task compared to more general ones: while in this work we assume just one interesting object with a single bounding box per sample, in works focused on object detection it is common to have a high number of boxes per image, spanning several different categories. This variation in the task framing could explain, at least in part, the differences in performance from broader works.

[10] https://pytorch.org/vision/stable/models.html#object-detection-instance-segmentation-and-person-keypoint-detection

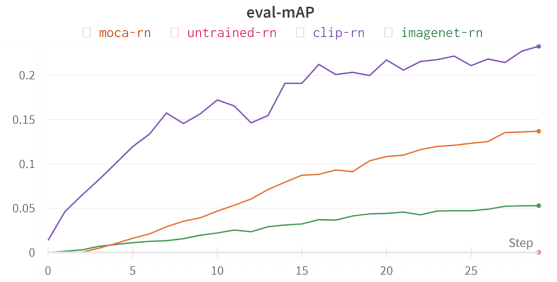

Figure 4: Object Detection: MAP on validation set per epoch.

Both MOCA and CLIP based models reach satisfying performances, therefore we consider their features to be good enough to be used in other experiments; we will include also the ImageNet model for comparison, despite its poor performance on this task. Additionally, this choice is also supported by previous work that shows that both MOCA and CLIP are appropriate vision models for AI2Thor synthetic scenes, as demonstrated by (Singh et al., 2021) and (Khandelwal et al., 2022), respectively.

B Additional Dataset, Model, and Results Information

Action	# Objs.	Action	# Objs.
Push	62	Pull	62
Pick Up	62	Put Down	62
Throw	62	Drop	62
Break	11	Fill	8
Dirty	8	Toggle	3
Open	2	Close	2

Table 5: List of the 12 actions used to generate after images and the number of object types per action.

Split	Object	# Actions	# Samples	Tot.
Test	Book	7	92	908
	Cup	9	316	
	Laptop	9	500	
Train	Bottle	8	52	2762
	Bowl	9	393	
	Box	7	188	
	Candle	7	115	
	CellPhone	8	128	
	Cloth	7	104	
	Egg	7	77	
	Kettle	8	98	
	Mug	9	350	
	Pan	7	151	
	Plate	8	278	
	Pot	8	221	
	Potato	6	65	
	Statue	7	290	
	Vase	7	259	
	WateringCan	6	23	
	WineBottle	7	35	
Valid	Bottle	8	14	690
	Bowl	9	92	
	Box	7	40	
	Candle	7	25	
	CellPhone	7	32	
	Cloth	6	32	
	Egg	4	11	
	Kettle	7	16	
	Mug	9	113	
	Pan	7	41	
	Plate	8	66	
	Pot	8	49	
	Potato	6	16	
	Statue	7	74	
	Vase	7	71	
	WateringCan	4	5	
	WineBottle	4	9	
				4360

Table 6: Action and sample counts for the Object split. 'Actions' column represents the number of unique actions performed on that object.

Split	Envt.	# Configs	# Samples	Total
Test	bathroom	3	36	820
	bedroom	5	158	
	kitchen	5	367	
	living room	5	277	
Train	bathroom	20	152	2856
	bedroom	20	472	
	kitchen	20	1513	
	living	20	773	
Valid	bathroom	5	44	693
	bedroom	5	117	
	kitchen	5	337	
	living	5	195	
				4360

Table 7: Environment information for the scene split.

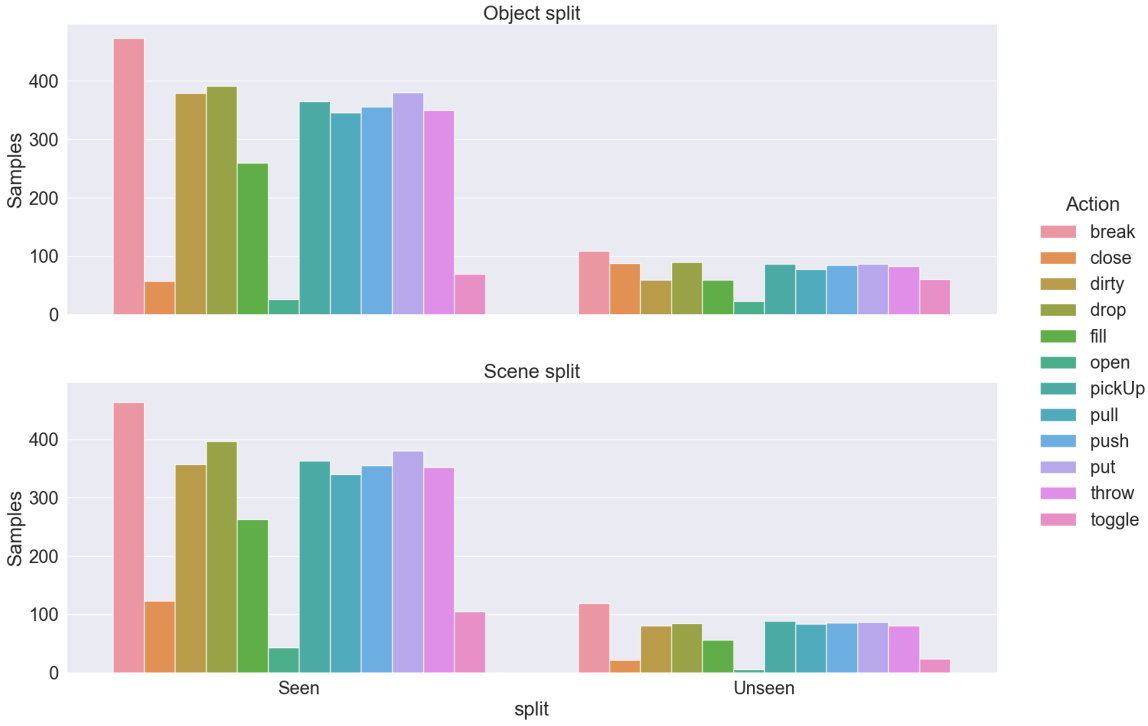

Figure 5: Distribution of actions between seen and unseen items per splitting procedure.

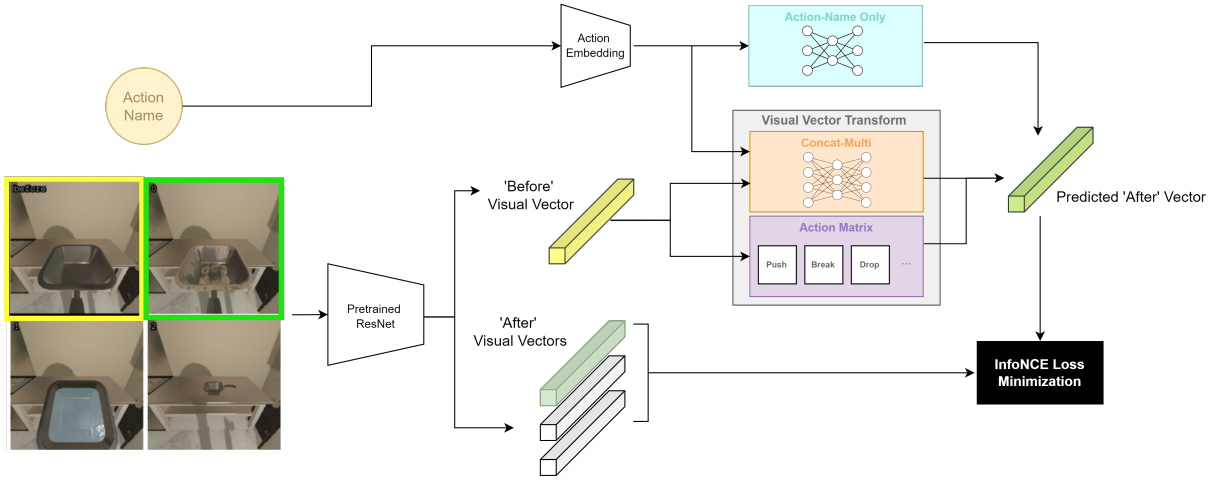

Figure 6: Schema of model architecture. Note: the arrow linking before-vector and vector transformation models represents, respectively: the dot product with the correct action matrix, for the Action-Matrix model; feedforwarding the vector as network input for the Concat-Multi model. We omit the Concat-Linear model since it uses the same procedure of the Concat-Multi.

VF	Embedding?	Model	Params
MOCA	No	Action-Matrix	259,308
		Concat-linear	23,520
		Concat-Multi	36,680
	Yes	Action-Matrix	259,308
		Concat-linear	375,296
		Concat-multi	539,520
CLIP	No	Action-Matrix	12,582,912
		Concat-linear	1,061,888
		Concat-Multi	1,602,421
	Yes	Action-Matrix	12,582,912
		Concat-linear	599,808
		Concat-multi	764,032

Table 8: Summary of model parameter counts. 'VF' is the visual feature extractor used, while 'Embedding' denotes whether the model uses an embedding space of the same dimension (256) for both actions and visual vectors instead of the 1-hot-vector action encoding (and raw visual vectors).

Object	break	close	dirty	drop	fill	open	pickUp	pull	push	put	throw	toggle
Book	//	//	//	**1.000**	//	**0.426**	0.508	0.538	0.646	0.780	0.420	//
Cup	**0.726**	//	**0.719**	0.993	**0.847**	//	**0.966**	**0.620**	**0.711**	**0.931**	**0.727**	//
Laptop	0.603	**0.482**	//	0.942	//	//	0.787	0.498	0.649	0.825	0.694	**0.573**

Table 9: Action-wise accuracies for the `Concat-Multi` MOCA model, computed for each object in the test set ('//' used when the action is not available for that object). Best values for columns in bold.

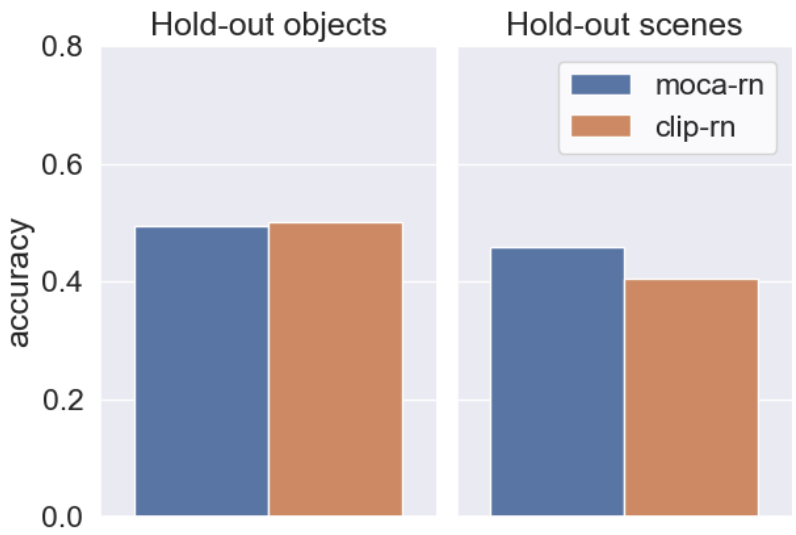

Figure 7: Results for accuracy of the least-squares regression model (average over 5 random splits).

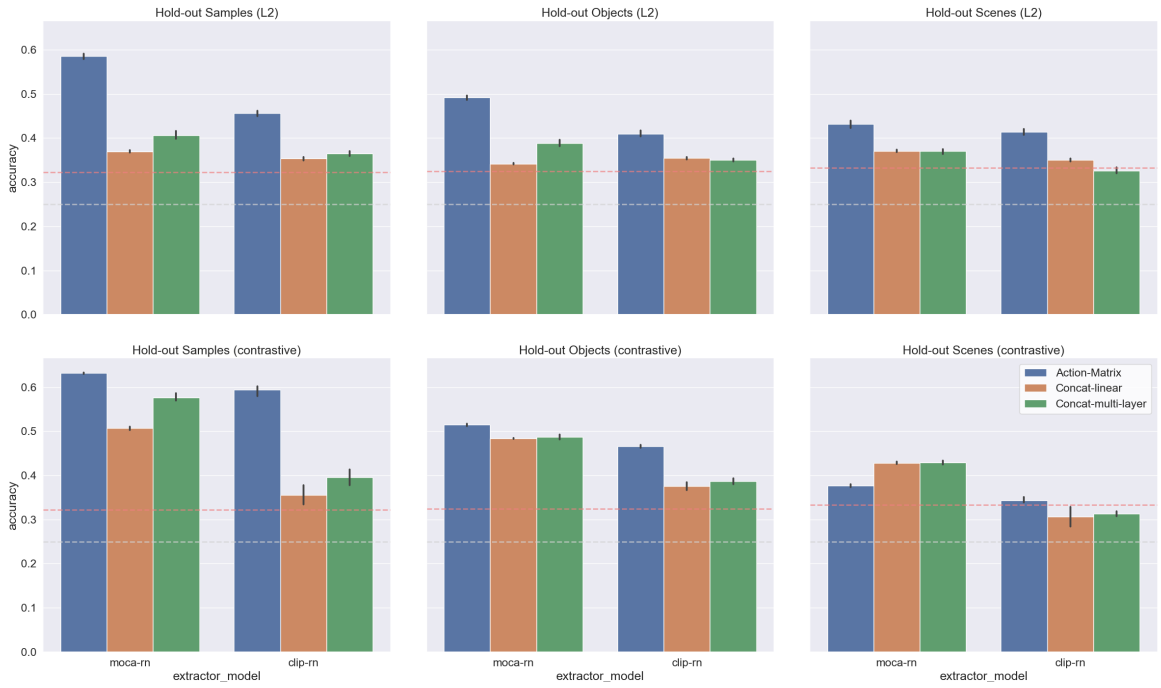

Figure 8: Comparison of L2 training procedure (top) and contrastive loss (bottom). Average over 5 random initializations. Dashed lines represent baselines (gray for chance, red for visual-similarity)

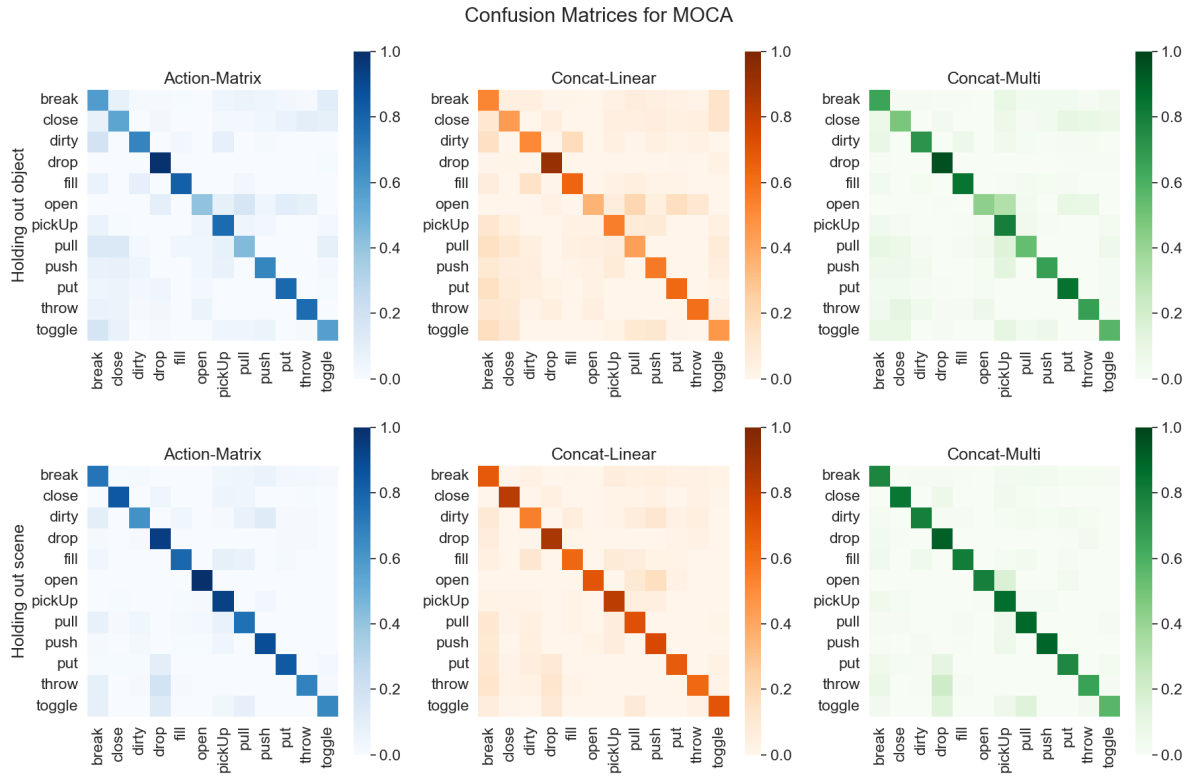

Figure 9: Action confusion matrix for the MOCA model with action embedding layer (row is ground truth, column represents prediction).

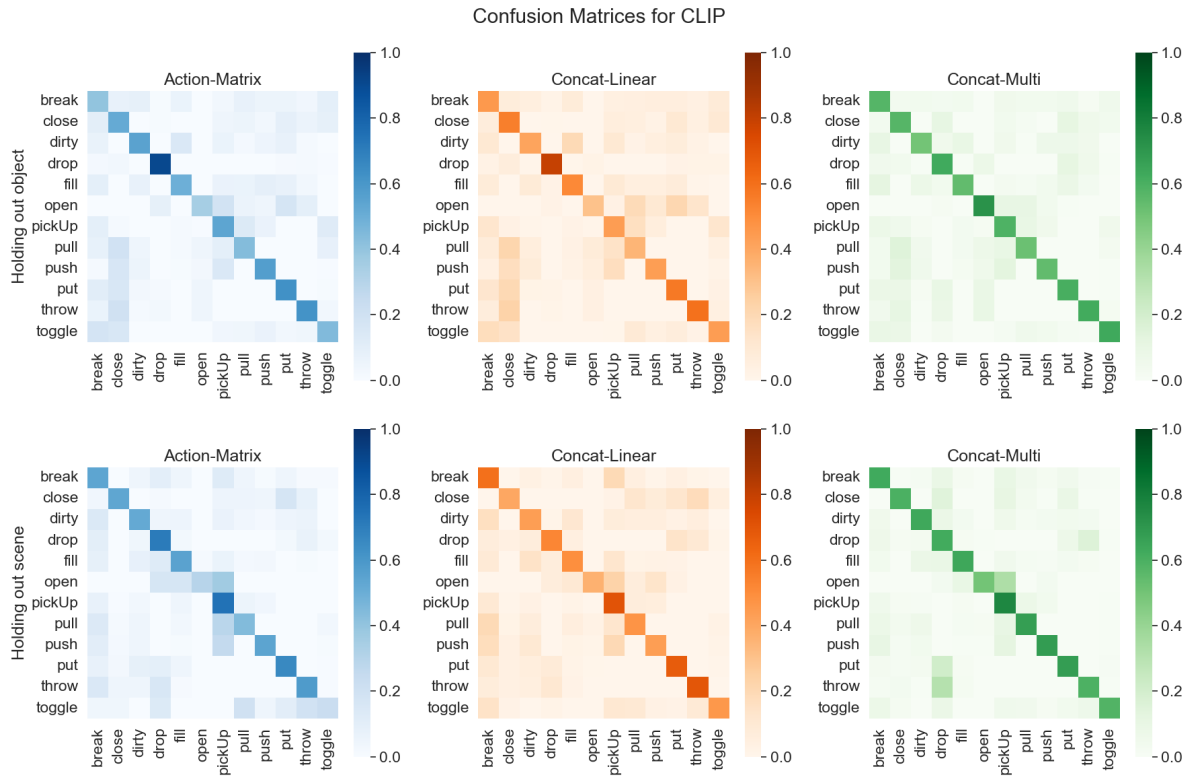

Figure 10: Action confusion matrix for the CLIP model with action embedding layer (row is ground truth, column represents prediction).

WildQA: In-the-Wild Video Question Answering

Santiago Castro* **Naihao Deng*** **Pingxuan Huang*** **Mihai Burzo** **Rada Mihalcea**
University of Michigan – Ann Arbor, USA
{sacastro, dnaihao, pxuanh, mburzo, mihalcea}@umich.edu

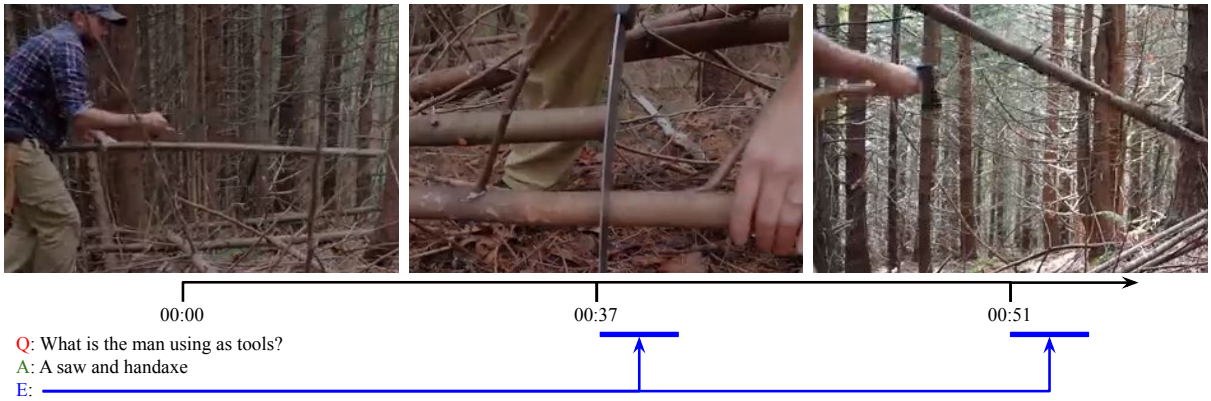

Figure 1: An example from our WildQA dataset, showing a question (Q), an answer (A), and evidence (E) that supports the answer. The corresponding part of the videos is provided as evidence for the question.

Abstract

Existing video understanding datasets mostly focus on human interactions, with little attention being paid to the "in the wild" settings, where the videos are recorded outdoors. We propose **WILDQA**, a video understanding dataset of videos recorded in outside settings. In addition to video question answering (Video QA), we also introduce the new task of identifying visual support for a given question and answer (Video Evidence Selection). Through evaluations using a wide range of baseline models, we show that WILDQA poses new challenges to the vision and language research communities. The dataset is available at https://lit.eecs.umich.edu/wildqa/.

1 Introduction

Video understanding plays an important role in the development of intelligent AI systems, as it enables the effective processing of different modalities of information (Li et al., 2021a). Various tasks have been proposed to examine the ability of models' to understand videos, including video question answering (Video QA), video captioning, and fill-in-the-blank tasks (Xu et al., 2017; Tran et al., 2016; Castro et al., 2022). Recent years have witnessed significant progress in video understanding, including new benchmarks (Tapaswi et al., 2016; Grauman et al., 2021) as well as advanced sophisticated benchmarksmodels (Jin et al., 2019; Radford et al., 2021).

There are however several drawbacks associated with existing video understanding research. First, existing video understanding benchmarks focus on common human activities as typically appearing in cooking videos (Zhu et al., 2017) or in movies (Tapaswi et al., 2016), leading to a limited set of video domains. Second, most video understanding benchmarks adopt a multiple-choice format, where models select an answer from a set of candidates (Jang et al., 2017; Castro et al., 2020). Models trained under such a setting cannot be used in real-life applications because candidate answers are not provided (Castro et al., 2022). Third, videos included in existing benchmarks are typically short (Kim et al., 2016), and the performance of models on longer videos is not well studied.

We address these challenges in our dataset construction process. First, we propose the WILDQA dataset in which we collect "in the wild" videos that are recorded in the outside world, going beyond daily human activities. Figure 2 shows the differ-

*: Equal contribution

Proceedings of the 29th International Conference on Computational Linguistics, pages 5613–5635
October 12–17, 2022.

Example from MovieQA *(multiple-choice)*
Q: How does E.T. show his happiness that he is finally returning home?
A: His heart lights up.

Example from TVQA *(multiple-choice)*
Q: Why does Joey want Chandler to kiss Janice when they are in the kitchen?
A: Because then she will leave.

Example from WildQA^{ours} *(open-ended)*
Q: What sort of environment is it based on the landscape and plant life?
A: Temperate mountain environment.

Figure 2: Examples from MovieQA (Tapaswi et al., 2016), TVQA (Lei et al., 2018), and our WildQA dataset. The previous datasets mostly focus on human interactions in a multiple-choice setting, while ours focus on scenes recorded in the outside world in an open-ended setting. We only list a single answer here for illustration purposes.

ence between the WILDQA dataset and previous question answering datasets. Second, we adopt the challenging answer generation approach, aiming to build a system that can answer questions with an open-ended answer, rather than selecting from a predefined set of candidate answers. Third, the average video length in our dataset is one minute, longer than the video clips in most of the existing datasets in Table 3, which presents a novel challenge for video understanding algorithms.

Using the WILDQA dataset, we address two main tasks. First, we address the task of video question answering (**Video QA**), aiming to generate open-ended answers. Second, we introduce the task of retrieving visual support for a given question and answer (**Video Evidence Selection**). Finding the relevant frames in a video for a given question-answer pair can help a system in its reasoning process, and is in line with ongoing efforts to build interpretable models (Jacovi and Goldberg, 2020). For each of these two tasks, we evaluate several baseline models, including multi-task models that combine the two tasks together. Figure 1 shows an example from our dataset, including an example of a question, answer, and supporting video evidence.

To summarize, the main contributions of this paper are:

1. We propose WILDQA, a multimodal video understanding dataset where video scenes are recorded in the outside world.
2. We propose two tasks for WILDQA: Video QA and Video Evidence Selection, aiming to build more interpretable systems.
3. We test several baseline models; experimental results show that our dataset poses new challenges to the vision and language research communities.

2 Related Work

Multimodal Question Answering. Two popular and representative tasks are Visual Question Answering (Visual QA) on images, and Video Question Answering (Video QA) on videos. Visual QA has attracted attention for a long time (Malinowski and Fritz, 2014; Zhang et al., 2016; Ren et al., 2015; Zhu et al., 2016). Recently, much progress has been made in Video QA. Researchers proposed various datasets such as TVQA that contain videos from movies or TV series (Tapaswi et al., 2016; Lei et al., 2018, 2020a) or videos from the Internet spanning from YouTube videos to Tumblr GIFs (Zeng et al., 2017; Ye et al., 2017; Jang et al., 2017; Yu et al., 2019b). Other datasets such as MSVD-QA (Xu et al., 2017) contain videos from the existing corpus (Chen and Dolan, 2011) or cartoon videos (Kim et al., 2016). Recent Video QA datasets have stronger focuses such as temporal relations (Mun et al., 2017), multi-step and non-factoid answers (Colas et al., 2020), natural interactions (Zadeh et al., 2019), characters in the video (Choi et al., 2021), question answering in real life (Castro et al., 2020), incorporating external knowledge (Garcia et al., 2020), and videos recorded from the egocentric view (Fan, 2019; Grauman et al., 2021). To the best of our knowledge, we are the first to collect videos from the outside world.

Researchers have also developed various methods to handle the Video QA task, including joint reasoning of the spatial and temporal structure of a video (Zhao et al., 2017; Gao et al., 2019; Huang et al., 2020; Jiang et al., 2020), integrating memory to keep track of past and future frames (Kim et al., 2017; Gao et al., 2018a; Zhao et al., 2018; Fan et al., 2019; Yu et al., 2020), various attention mechanisms (Zhu et al., 2017; Zhang et al., 2019;

Li et al., 2019; Yu et al., 2019a; Kim et al., 2018; Jin et al., 2019), and others. Recently, pre-trained models have proved to be useful in various visual and language tasks (Radford et al., 2021; Chen et al., 2020; Zellers et al., 2021). However, the pre-trained visual and language models are typically encoder-only and cannot generate an answer in natural language on their own. Thus, such pre-trained encoder-only models do not fit into the open-end video question answering setting in our task.

Additionally, previous work has also investigated various reasoning tasks in a multimodal setting (Gao et al., 2016; Yang et al., 2018; Gao et al., 2018b; Zellers et al., 2019). Although it is not our focus, some questions in our dataset require a certain level of reasoning ability. Moreover, since our dataset is created by domain experts, there is domain knowledge involved in the questions as well.

Moment Retrieval. Moment Retrieval is the task of retrieving a short moment from a large video corpus given a natural language query (Escorcia et al., 2019; Lei et al., 2020b). Researchers have proposed or adapted various datasets for this task (Krishna et al., 2017; Hendricks et al., 2017; Gao et al., 2017; Lei et al., 2020b). The task of retrieving relevant parts in the video given the question (Video Evidence Selection) in our proposed dataset is akin to Moment Retrieval. However, moment retrieval focuses on retrieving the part of videos that the question describes, while Video Evidence Selection is to find parts of videos that can support the answer to the questions as shown in Figure 1. Prior work such as Tutorial-VQA (Colas et al., 2020) also adopt the setting of providing parts of the videos as answers to the question, but they did not include any text answers in their dataset.

Few-shot Learning. Recently, there is a trend to evaluate neural models in a few-shot learning setting (Huang et al., 2018; Mukherjee and Awadallah, 2020; Sun et al., 2020; Li et al., 2021b; Lee et al., 2021; Pfeiffer et al., 2022), where the model is tuned with a small portion of the data and tested against the rest. We adopt the few-shot learning setting for our dataset for both Video QA and Video Evidence Selection.

3 WildQA Dataset

Video Selection and Processing. Following Zadeh et al. (2019); Castro et al. (2020), we start by collecting videos from YouTube. First, we identify five domains that primarily consist of outdoor scenes and are representative for the outside world, namely, ***Agriculture***, ***Geography***, ***Human Survival***, ***Natural Disasters***, and ***Military***. We then manually collected videos from relevant YouTube channels for each domain.

Because the raw videos can be as long as an hour, we split the raw videos into short clips using PySceneDetect,[1] and concatenate these short clips so that the output video is approximately one minute. We use the output videos for the annotation process described below. More details for the video selection and video processing steps are discussed in Appendix A.1.

Question, Answer, and Evidence Annotation. There are two phases in our annotation process, as shown in Figure 3. In **Phase 1**, annotators watch the video clips and come up with a hypothetical motivation. They ask one or more **question**s and provide an **answer** to each of the questions they ask. Annotators are also instructed to provide all the relevant parts in videos as pieces of **evidence** to support the answer to their question. After this step in the data collection, three of the authors of this paper manually review all the question-answer pairs for quality purposes. Next, in **Phase 2**, we collect more **answer**s and **evidence**s for each question from Phase 1. Over the entire annotation process, annotators spent a total of **556.81 annotation hours**, split into 77.05 hours in Phase 1 and 479.76 in Phase 2. Appendices A.2, A.3, and A.5 present the annotation instructions, annotation interfaces, and reviewing process for question-answer pairs, respectively.

Because we want to collect questions that domain experts are interested, as opposed to arbitrary questions, domain experts carry out the Phase 1 annotations. To demonstrate the quality difference of questions collected from domain experts versus non-experts, we conduct a pilot study. Appendices A.4 and A.6 discuss the pilot study and the annotators' expertise, respectively.

Dataset Statistics. Tables 1 and 2 present statistics of the videos and associated questions for each of the five domains, along with other relevant statistics. Figure 4 shows the distribution of question types. Appendix A.7 discusses more statistics.

[1] PySceneDetect uses the OpenCV (Bradski, 2000) to find scene changes in video clips (`py.scenedetect.com`).

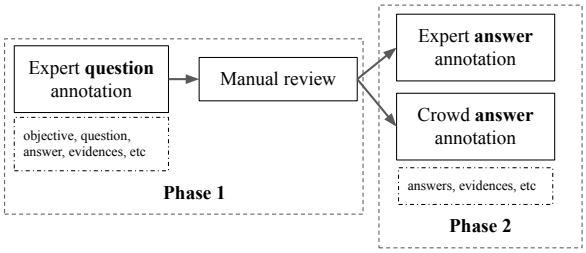

Figure 3: The two phases of data annotation.

Domain	video count	question count
Agriculture	85	109
Human Survival	95	309
Natural Disaster	70	187
Geography	46	110
Military	73	201
Total	369	916

Table 1: Video and question count for each domain.

Videos	369
Duration (in seconds)	71.22 ± 26.47
Questions	916
Question per video	2.48 ± 1.38
Question length (#tokens)	7.09 ± 2.60
Answer per question	2.22 ± 0.69
Answer length (#tokens)	9.08 ± 8.15
Evidence per answer	1.18 ± 0.80
Evidence length (s)	9.64 ± 10.96

Table 2: Dataset statistics for WildQA.

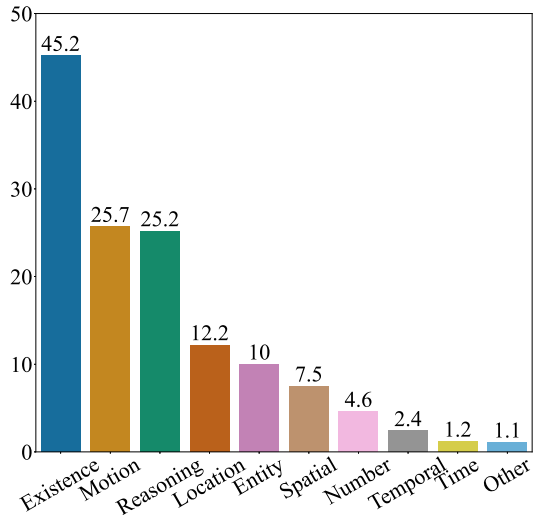

Figure 4: Percentage distribution of question types. Because one question might be classified into multiple categories, the scale summation is larger than 100%.

Q: What type of weather is happening?
A: Flooding and rain.
The weather is rain and flood.

Q: Where is the road at?
A: It is in a tundra environment
The road zig-zags across the landscape.
The road winds through a mountainous landscape.
The road is in an elevated area.

Figure 5: Examples of questions (Q) and answers (A) from WildQA. The first answer is collected during Phase 1 of the annotation process; all remaining answers are collected in Phase 2. More analyses in Appendix A.7.

Dataset Comparison. Table 3 shows the comparison between WILDQA and other existing datasets.

4 Video Question Answering

Following Xue et al. (2017), we adopt **free-form open-ended** video question answering for our video question answering (Video QA) task. Given a question **q** and a video **v**, the task is to generate an answer **a** in natural language.

We adopt a **few-shot learning** setting on our dataset, where models are fine-tuned on question-answer pairs corresponding to 30% of the videos for each domain. The tuned models are tested on data for the remaining 70% videos. The reason is that the time to annotate 30% of the data is around 23 hours, during which there are around 50 data points annotated for each domain, which is acceptable. We hypothesize that it is realistic to have such a setting because the potential end-users could spend around a day or two collecting data, and we can then quickly tune a model using it. Moreover, no repeated videos appear in different splits, following Lei et al. (2018). We end up having 264 question-answer pairs for 108 videos in our dev set and 652 pairs for 261 videos in the test set. We adopt BLEU (Papineni et al., 2002) and ROUGE (Lin, 2004) as the metrics to measure the quality of the generated answer. We run each model 3 times and report the scores of mean ±

Dataset	Domain	VE?	#Videos	# Q	Avg dur. (s)	Annotation	QA Task
MovieQA (Tapaswi et al., 2016)	Movies	√	6.7K	6.4K	203	Manual	MC
VideoQA (FiB) (Zhu et al., 2017)	Cooking, movies, web		109K	390K	33	Automatic	MC
MSRVTT-QA (Xu et al., 2017)	General life videos		10K	243K	15	Automatic	OE
MovieFIB (Maharaj et al., 2017)	Movies		128K	348K	5	Automatic	OE
TVQA (Lei et al., 2018)	TV shows	√	21.8K	152K	76	Manual	MC
ActivityNet-QA (Yu et al., 2019b)	Human activity		5.8K	58K	180	Manual	OE
TVQA+ (Lei et al., 2020a)	TV shows	√	4.2K	29.4K	60	Manual	MC, ES
KnowIT VQA (Garcia et al., 2020)	TV shows		12K	24K	20	Manual	MC
LifeQA (Castro et al., 2020)	Daily life		275	2.3K	74	Manual	MC
TutorialVQA (Colas et al., 2020)	Instructions	√	76	6.2K	–	Manual	ES
NExT-QA (Xiao et al., 2021)	Daily life		5.4K	52K	44	Manual	MC, OE
FIBER (Castro et al., 2022)	Human actions		28K	2K	10	Manual	OE
WildQA	**In-the-wild**	√	369	916	71.2	Manual	OE, **ES**

Table 3: Comparison between our WILDQA and other existing datasets. **VE?**: Whether the dataset provides "Video Evidences"?; **MC**: "Multiple Choice" question answering; **OE**: "Open Ended" question answering; **ES**: "Evidence Selection". We adapt the comparison table from Zhong et al. (2022).

standard deviation in Table 4.

4.1 Baselines

Human Baselines. We report the average BLEU and ROUGE scores by leaving one annotator out in Table 4 (**Human**).

Text-only Models. We implement several baselines that only use the question-answer pairs in the dev set. **Random** randomly chooses answers from the dev set. **Common** always predicts the most common answer in the dev set; **Closest** employs embedding produced by a pretrained `roberta-base` model (Liu et al., 2019). In the inference, **Closest** retrieves the answers for the dev set question whose embedding has the highest cosine similarity to the test question. We also fine-tune T5 (Raffel et al., 2020) using question-answer pairs from the dev set (**T5 T**).

Text + Visual Models. Following Castro et al. (2022), we concatenate the text features with the visual features and input the concatenated features to the T5 model (**T5 T+V**). We extract I3D (Carreira and Zisserman, 2017) video features and take one feature per second.

Multi-task Learning. Multi-task learning has proved to be successful in various domains (Collobert and Weston, 2008; Deng et al., 2013; Girshick, 2015). Following Caruana (1993), we train **Multi$_{\text{T+V, SE}}$** which combines T5 T+V and T5 SE (the Video Evidence Selection model described in Section 5) with a shared T5 encoder between the tasks of Video Question Answering and Video Evidence Selection. We also train **Multi$_{\text{T+V, IO}}$** which combines T5 T+V and T5 IO (another Video Evidence Selection model described in Section 5) in a similar way. The loss function during the fine-tuning is:

$$L = \alpha L_1 + \beta L_2 \quad (1)$$

where L_1, L_2 are the losses for Video Question Answering and Video Evidence Selection, respectively; α, β are the weights for the two tasks. The selection process behind the values of α and β are presented in Appendix C.

Model name	ROUGE-1	ROUGE-2	ROUGE-L
Random	5.0 ± 0.2	0.5 ± 0.1	4.9 ± 0.2
Common	10.6 ± 0.0	0.0 ± 0.0	10.6 ± 0.0
Closest	19.5 ± 0.0	6.2 ± 0.0	18.7 ± 0.0
T5 T$^{\text{0-shot}}$	0.8 ± 0.0	0.0 ± 0.0	0.8 ± 0.0
T5 T	33.8 ± 0.2	17.7 ± 0.1	32.4 ± 0.3
T5 T+V	33.1 ± 0.3	17.3 ± 0.4	31.9 ± 0.2
Multi$_{\text{T+V, IO}}$	$\mathbf{34.0 \pm 0.5}$	$\mathbf{18.8 \pm 0.7}$	$\mathbf{32.8 \pm 0.6}$
Multi$_{\text{T+V, SE}}$	33.8 ± 0.8	18.5 ± 0.7	32.5 ± 0.8
Human	40.8 ± 0.0	18.1 ± 0.0	36.3 ± 0.0

Table 4: ROUGE scores for the task of Video Question Answering. For comparison, we test the out-of-box T5 model under the zero-shot setting (T5 T$^{\text{0-shot}}$).

4.2 Results

Table 4 reports F1 scores of ROUGE-1 (R1), ROUGE-2 (R2), and ROUGE-L (RL) for our baseline models. For comparison, we also test the out-of-box T5 model on our test split under the zero-shot setting (**T5 Text**$^{\text{0-shot}}$ in Table 5).

T5-based models significantly outperform the random baselines as well as the out-of-box T5

model, which suggests that the T5-based models acquire certain levels of question-answering ability in the tuning stage. However, adding visual features does not improve the model's performance. This might be due to the *challenges of attending to the visual features at the corresponding parts in the video*, because both models under multi-task learning outperform the text-only baseline, suggesting that attending to the correct part of the video helps the answer generation process.

All baseline models underperform human baselines on ROUGE scores, especially on ROUGE-1 and ROUGE-L scores, suggesting that there is room for improvement. However, the ROUGE-2 score for human annotators is low because although human annotators tend to use the same word to describe the object that appears in the video, there are large variations in terms of expressing the ideas of their answers. More discussions on the diversity of the answers are in Appendix A.7.

5 Video Evidence Selection

Similar to Colas et al. (2020), given a video **v** and a question **q**, the video evidence selection task consists of predicting $\{(\mathbf{s}_1, \mathbf{e}_1), (\mathbf{s}_2, \mathbf{e}_2), \ldots\}$, where $(\mathbf{s}_i, \mathbf{e}_i)$ represents the time for start **s** and end **e** of a singles span within the video **v**. We also adopt the few-shot learning setting as described in Section 4 for the task of Video Evidence Selection. Similar to DeYoung et al. (2020), we design an Intersection-Over-Union (IOU) metric borrowed from Everingham et al. (2010). We define IOU as follows: given two time spans in the video, IOU is defined as the length of their intersection divided by the length of their union. The prediction is counted as a match if it overlaps with any of the ground truth spans by more than the threshold (0.5, following DeYoung et al., 2020). We use these partial matches to calculate an F1 score (IOU-F1 scores). As described in Section 4, we run each model three times and report the scores of mean $\pm$ standard deviation in Table 5.

5.1 Baselines

As described in Section 4, we compute the average IOU-F1 score on the annotations from one annotator against the remaining annotators; we denote this metric as **`Human`**. The **`Random`** baseline consists of randomly choosing the start and end of a part within the original video as evidence. Similar to the structure Devlin et al. (2019) experiment on SQuAD (Rajpurkar et al., 2016), we build **`T5 SE`**; here, we feed the concatenated question embeddings and I3D visual features to the T5 encoder, and the T5 encoder outputs a sequence of the encoded states. We treat the subsequence corresponding to the visual features as the encoded hidden sequence $T_m \in R^H$ for the video frames (H denotes the dimension of the hidden sequence). We then multiply the sequence with two vectors $S, E \in R^H$. The T_i and T_j that maximize the likelihood are predicted as the **start and the end of the evidence**, respectively. During the training, we maximize their joint probability:

$$P_i P_j = \frac{e^{S \cdot T_i}}{\sum_m e^{S \cdot T_m}} \frac{e^{E \cdot T_j}}{\sum_m e^{E \cdot T_m}}$$

where P_i and P_j are the probability for the i being the start and j the end of the evidence, respectively.

Inspired by the Inside-Outside-Beginning ("IOB") tagging scheme (Ramshaw and Marcus, 1995), we also formulate the evidence finding as a task of tagging whether a video frame is inside ("I") the evidence, or outside ("O") the evidence. We then build **`T5 IO`** by feeding the concatenated features to a T5 encoder. Similar to `T5 Start End`, we have an encoded sequence of $T_m \in R^H$ corresponding to the video frames. We then multiply the sequence with a vector $L \in R^H$ and apply a sigmoid function to the multiplication result. The model predicts the frame as "I" if the value at the corresponding position is greater than or equal to 0.5, otherwise it predicts "O". We test $\texttt{Multi}_{\texttt{T+V,IO}}$ and $\texttt{Multi}_{\texttt{T+V,SE}}$ described in Section 4 on Video Evidence Selection as well.

Model name	IOU-F1
`Random`	2.5 ± 0.3
`T5 IO`	1.1 ± 0.2
`T5 SE`	$\mathbf{4.5} \pm \mathbf{0.8}$
$\texttt{Multi}_{\texttt{T+V,IO}}$	1.4 ± 0.3
$\texttt{Multi}_{\texttt{T+V,SE}}$	3.7 ± 2.4
`Human`	18.37 ± 0.0

Table 5: IOU-F1 scores for Video Evidence Selection.

5.2 Results

Table 5 shows the performance of the baseline models on the Video Evidence Selection task. All the baseline models perform significantly worse than the human annotators, and sometimes worse than the random baseline. This is understandable be-

Type	R1	IOU-F1
Existence	**33.3 ± 0.3**	**5.3 ± 0.3**
Motion	32.8 ± 0.6	3.1 ± 2.0
Reasoning	33.3 ± 0.4	3.1 ± 1.3
Location	**26.2 ± 10.7**	**4.4 ± 1.4**
Entity	33.2 ± 0.7	5.2 ± 0.7
Spatial	**32.2 ± 0.6**	**2.4 ± 1.7**
Number	33.8 ± 0.4	4.5 ± 0.7
Temporal	33.8 ± 0.6	3.8 ± 0.5
Time	33.1 ± 0.8	5.7 ± 1.0
Other	33.2 ± 0.6	5.3 ± 0.9

Table 6: $\texttt{Multi}_{\texttt{T+V,SE}}$ performance on different question types for Video QA (ROUGE-1) and for Video Evidence Selection (IOU-F1).

Model name	R1	R2	RL
T5 $\texttt{T}^{\text{0-shot}}$	0.8 ± 0.0	0.0 ± 0.0	0.8 ± 0.0
T5 $\texttt{T}^{\text{0-shot}}_{\text{TVQA}}$	9.1 ± 0.0	1.2 ± 0.0	8.8 ± 0.0
T5 $\texttt{T}_{\text{TVQA,ours}}$	32.4 ± 0.2	17.5 ± 0.2	31.6 ± 0.2
T5 $\texttt{T}_{\text{ours}}$	**33.8 ± 0.2**	17.7 ± 0.1	32.4 ± 0.3
T5 $\texttt{T+V}^{\text{0-shot}}_{\text{TVQA}}$	20.3 ± 0.0	8.1 ± 0.0	20.1 ± 0.0
T5 $\texttt{T+V}_{\text{ours}}$	33.1 ± 0.3	17.3 ± 0.4	31.9 ± 0.2
T5 $\texttt{T+V}_{\text{TVQA,ours}}$	33.7 ± 0.2	**18.3 ± 0.1**	**32.6 ± 0.1**

Table 7: ROUGE scores for the task of Video Question Answering for few-shot learning setting (the standard setting in our WildQA dataset introduced in Section 4) and zero-shot learning setting ("0-shot" in the superscript). Subscript "TVQA" means pre-training on the TVQA (Lei et al., 2018) dataset; subscript "TVQA,ours" means first pre-training the model on TVQA, then tuning the model on our WildQA dataset; subscript "ours" means tuning the model directly on our WildQA dataset.

cause selecting evidence from a long video can be difficult. Additionally, multi-task learning makes the model's performance worse. However, this could be due to the fact that the Video Evidence Selection itself is a hard task, and all the baseline models struggle with such a task. Although multi-task learning does not help Video Evidence Selection, as mentioned in Section 4, training with Video Evidence Selection does help Video QA. Thus, Video Evidence Selection is still an important task to improve a model's ability to answer questions. We include more ablation studies in Appendix D.1.

6 Analysis and Discussion

Model Performance v.s. Question Types. Table 6 shows $\texttt{Multi}_{\texttt{T+V,SE}}$'s performance on different question types for Video QA and Video Evidence Selection respectively. Other ROUGE scores for Video QA follow similar trends as shown in Figure 14. According to Table 6, the model achieves good ROUGE-1 scores for Video QA when the model has a good IOU-F1 score for Video Evidence Selection such as its performance on *Existence*. The model has the highest ROUGE-1 variation on *Location* question types, with a relatively large variation for IOU-F1. The model's ROUGE-1 score on *Spatial* questions is relatively low, with the lowest IOU-F1 score. $\texttt{Multi}_{\texttt{T+V,SE}}$ excels at question type *Entity* and *Existence* with relatively high IOU-F1 scores. One possible explanation could be that the average length of the answers generated for *Entity* and *Existence* are around eight tokens, which might be easier for the model to ground to the relevant part in the video.

Interestingly, even if the answers have similar lengths, the model struggles on *Motion* questions (with a relatively low IOU-F1 score). A possible reason could be that this type of questions provide a very abstract description of the action, which makes the model hard to attend to the relevant part of the video. For instance, an example of a *Motion* question is *"Are there any structure or natural features being affected?"*. To attend to the corresponding period in the video, the model needs to understand the word "affected" and the objects that are actually affected, which can be very difficult. The model also struggles to attend to the correct places in the video for the *Spatial* type of question. This might be because there is more than one entity in *Spatial* type of questions, and the model needs to locate all the objects appearing in various parts of the video, which is similarly complex. For instance, for the question *"What effects did the weather have?"*, the model needs to attend to *"debris in the air"*, *"truck turnover"* and *"destruction of buildings"*. For *Location* type of questions such as *"What sorts of terrain is the vegetation present in?"*, it might be difficult to attend to all the terrains of *"forest"*, *"plateaus"*, *"mountainous"*, *"valleys"*, and *"arboreal"* and to include them in the answer.

Domain Adaptation. Furthermore, we tune the $\texttt{Multi}_{\texttt{T+V,SE}}$ model on the dev set data from a single domain, and test it against data from other domains. Figures 6 and 7 show the model's performance in different tuning and testing domains. Interestingly, the diagonal cells do not always have the darkest color, which indicates that interrelations exist across domains. For instance, the model tuned on *Geography* performs relatively better for Video QA on *Human Survival* and *Agri-*

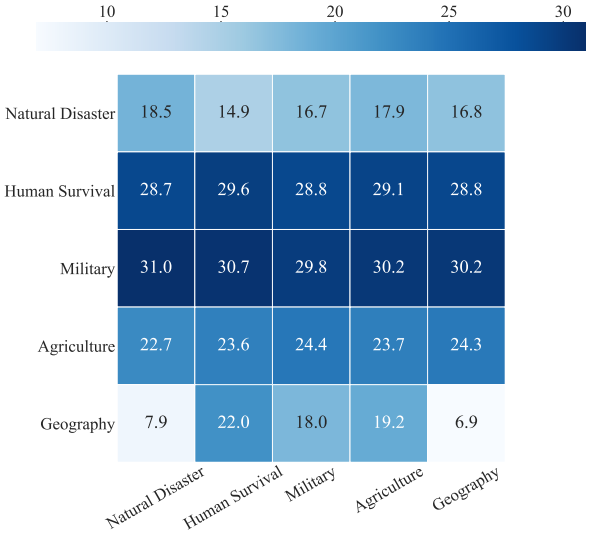

Figure 6: $\texttt{Multi}_{\texttt{T+V,SE}}$ performance (ROUGE-1) for Video QA when tuned on a single domain (y-axis) and tested against each domain (x-axis). The performances by the rest metrics for Video QA resemble the pattern here and are reported in Appendix D.

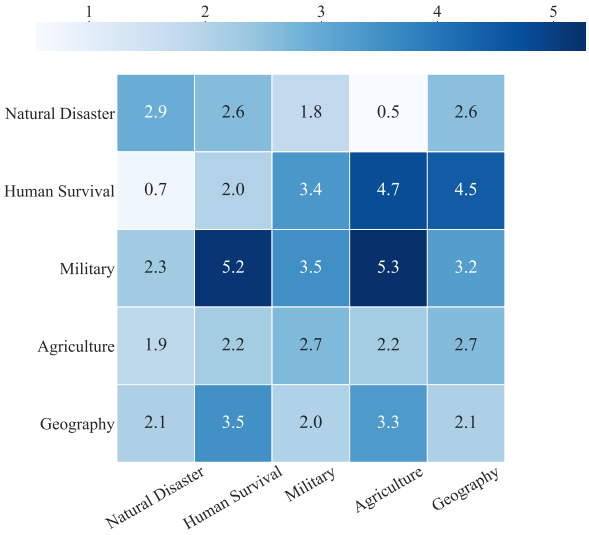

Figure 7: $\texttt{Multi}_{\texttt{T+V,SE}}$ performance (IOU-F1) for Video Evidence Selection when tuned on a single domain (y-axis) and tested against each domain (x-axis).

culture rather than itself. This suggests that the questions and videos from *Geography*, *Agriculture*, and *Human Survival* exhibit some similarity so that the model tuned on one domain can answer questions from the other domains relatively well. But answering questions from *Geography* can introduce the domain knowledge, an example of the answer is "`Mountainous, temperate forest.`", where "`temperate forest`" is one of the terminologies specific to *Geography* domain. Training on these terminologies might confuse the model and hurt the performance. Thus, future research might be needed to study how to better incorporate domain knowledge into multimodal question answering.

As for Video Evidence Selection, the patterns generally resemble the pattern in Figure 6, which means that in general, the model answers a question better if it can attend to the relevant part in the video. However, when tuned on *Human Survival* and tested on *Natural Disaster* the model performs relatively well on Video QA (with a 28.7 ROUGE-1 score) but less well on Video Evidence Selection (with a 0.7 IOU-F1 score). This might indicate that the model picks up some common patterns in the text rather than reasoning about the video and the question in an expected manner.

Pre-training on Other Datasets. We also pre-train the `T5 T` and `T5 T+V` using TVQA (Lei et al., 2018), a large-scale multimodal question answering dataset with videos from TV series. We report the zero-shot learning performances as well as the few-shot learning performances for `T5 T` and `T5 T+V` in Table 7. We can see that pre-training on TVQA for text-only `T5 T` does not help, which shows that the question styles in our dataset might be different from TVQA. For `T5 T+V` which uses both text and visual features, pre-training on TVQA does help the model, which suggests that the pre-training helps the model take advantage of the visual features. `T5 T+V` pre-trained on TVQA underperforms `T5 T+V` trained together with `T5 IO` (the $\texttt{Multi}_{\texttt{T+V,SE}}$ model) according to Table 4 and Table 7, suggesting that attending to the relevant part in the video helps the model better than training the model on more data. However, pre-training the model on the TVQA dataset reduces the variance of model performance, which suggests that training the model with more data helps the model perform consistently.

7 Conclusion

In this paper, we introduced a new and challenging benchmark, WILDQA, to promote domain diversity for video understanding. Specifically, we focused on five domains that involve long videos recorded in the outside world, which can be useful for applications in these domains. Instead of the traditional multiple-choice setting for Video Question Answering, we proposed to generate open-ended answers. We believe open-end answer generation can help construct systems that can answer end

users' questions in a more natural way. To help the model attend to the relevant parts in the videos, we also proposed the task of Video Evidence Selection. Through experiments, we showed the feasibility of these tasks, and also showed that jointly training for both Video Question Answering and Video Evidence Selection can improve the models' performance. In addition, we found it is easier to understand models' behavior by knowing which part of the video the model attends to when answering a question. We believe that this is an important step towards a trustworthy, explainable multimodal system. The dataset is available at https://lit.eecs.umich.edu/wildqa/.

Acknowledgement

We thank the anonymous reviewers for their constructive feedbacks. We thank Artem Abzaliev, Do June Min, and Oana Ignat for proofreading and suggestions. We thank William McNamee for the help with the video collection process, and all the annotators for their hard work on data annotation. We thank Yiqun Yao for the helpful discussions during the early stage of the project. This material is based in part upon work supported by the Automotive Research Center ("ARC"). Any opinions, findings, conclusions, or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of ARC or any other related entity.

Ethical Consideration

The videos we use are all publicly available on YouTube. The dataset includes a variety of domains, including videos in the *Military* domain, but we are ensuring that all the videos are only used for asking questions specific to the video content. Moreover, in our manual examination, we make sure that the question-answer pairs collected from our annotators are appropriate without any use of offensive language.

References

G. Bradski. 2000. The OpenCV Library. *Dr. Dobb's Journal of Software Tools*.

João Carreira and Andrew Zisserman. 2017. Quo vadis, action recognition? A new model and the kinetics dataset. In *2017 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2017, Honolulu, HI, USA, July 21-26, 2017*, pages 4724–4733. IEEE Computer Society.

Richard Caruana. 1993. Multitask learning: A knowledge-based source of inductive bias. In *Proceedings of the Tenth International Conference on Machine Learning*, pages 41–48. Morgan Kaufmann.

Santiago Castro, Mahmoud Azab, Jonathan Stroud, Cristina Noujaim, Ruoyao Wang, Jia Deng, and Rada Mihalcea. 2020. LifeQA: A real-life dataset for video question answering. In *Proceedings of the 12th Language Resources and Evaluation Conference*, pages 4352–4358, Marseille, France. European Language Resources Association.

Santiago Castro, Ruoyao Wang, Pingxuan Huang, Ian Stewart, Oana Ignat, Nan Liu, Jonathan Stroud, and Rada Mihalcea. 2022. FIBER: Fill-in-the-blanks as a challenging video understanding evaluation framework. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 2925–2940, Dublin, Ireland. Association for Computational Linguistics.

David Chen and William Dolan. 2011. Collecting highly parallel data for paraphrase evaluation. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, pages 190–200, Portland, Oregon, USA. Association for Computational Linguistics.

Yen-Chun Chen, Linjie Li, Licheng Yu, Ahmed El Kholy, Faisal Ahmed, Zhe Gan, Yu Cheng, and Jingjing Liu. 2020. UNITER: Universal image-text representation learning. In *European conference on computer vision*, pages 104–120. Springer.

Seongho Choi, Kyoung-Woon On, Yu-Jung Heo, Ahjeong Seo, Youwon Jang, Minsu Lee, and Byoung-Tak Zhang. 2021. DramaQA: Character-centered video story understanding with hierarchical qa. *Proceedings of the AAAI Conference on Artificial Intelligence*, 35(2):1166–1174.

Anthony Colas, Seokhwan Kim, Franck Dernoncourt, Siddhesh Gupte, Zhe Wang, and Doo Soon Kim. 2020. TutorialVQA: Question answering dataset for tutorial videos. In *Proceedings of the 12th Language Resources and Evaluation Conference*, pages 5450–5455, Marseille, France. European Language Resources Association.

Ronan Collobert and Jason Weston. 2008. A unified architecture for natural language processing: deep neural networks with multitask learning. In *Machine Learning, Proceedings of the Twenty-Fifth International Conference (ICML 2008), Helsinki, Finland, June 5-9, 2008*, volume 307 of *ACM International Conference Proceeding Series*, pages 160–167. ACM.

Li Deng, Geoffrey Hinton, and Brian Kingsbury. 2013. New types of deep neural network learning for speech recognition and related applications: An overview. In *2013 IEEE international conference on acoustics, speech and signal processing*, pages 8599–8603. IEEE.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, Minneapolis, Minnesota. Association for Computational Linguistics.

Jay DeYoung, Sarthak Jain, Nazneen Fatema Rajani, Eric Lehman, Caiming Xiong, Richard Socher, and Byron C. Wallace. 2020. ERASER: A benchmark to evaluate rationalized NLP models. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 4443–4458, Online. Association for Computational Linguistics.

Victor Escorcia, Mattia Soldan, Josef Sivic, Bernard Ghanem, and Bryan Russell. 2019. Temporal localization of moments in video collections with natural language. *ArXiv preprint*, abs/1907.12763.

Mark Everingham, Luc Van Gool, Christopher KI Williams, John Winn, and Andrew Zisserman. 2010. The PASCAL visual object classes (VOC) challenge. *International journal of computer vision*, 88(2):303–338.

Chenyou Fan. 2019. EgoVQA – an egocentric video question answering benchmark dataset. In *Proceedings of the IEEE/CVF International Conference on Computer Vision Workshops*, pages 0–0.

Chenyou Fan, Xiaofan Zhang, Shu Zhang, Wensheng Wang, Chi Zhang, and Heng Huang. 2019. Heterogeneous memory enhanced multimodal attention model for video question answering. In *IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2019, Long Beach, CA, USA, June 16-20, 2019*, pages 1999–2007. Computer Vision Foundation / IEEE.

Jiyang Gao, Runzhou Ge, Kan Chen, and Ram Nevatia. 2018a. Motion-appearance co-memory networks for video question answering. In *2018 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2018, Salt Lake City, UT, USA, June 18-22, 2018*, pages 6576–6585. IEEE Computer Society.

Jiyang Gao, Chen Sun, Zhenheng Yang, and Ram Nevatia. 2017. TALL: temporal activity localization via language query. In *IEEE International Conference on Computer Vision, ICCV 2017, Venice, Italy, October 22-29, 2017*, pages 5277–5285. IEEE Computer Society.

Lianli Gao, Pengpeng Zeng, Jingkuan Song, Yuan-Fang Li, Wu Liu, Tao Mei, and Heng Tao Shen. 2019. Structured two-stream attention network for video question answering. In *The Thirty-Third AAAI Conference on Artificial Intelligence, AAAI 2019, The Thirty-First Innovative Applications of Artificial Intelligence Conference, IAAI 2019, The Ninth AAAI Symposium on Educational Advances in Artificial Intelligence, EAAI 2019, Honolulu, Hawaii, USA, January 27 - February 1, 2019*, pages 6391–6398. AAAI Press.

Qiaozi Gao, Malcolm Doering, Shaohua Yang, and Joyce Yue Chai. 2016. Physical causality of action verbs in grounded language understanding. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics, ACL 2016, August 7-12, 2016, Berlin, Germany, Volume 1: Long Papers*. The Association for Computer Linguistics.

Qiaozi Gao, Shaohua Yang, Joyce Yue Chai, and Lucy Vanderwende. 2018b. What action causes this? towards naive physical action-effect prediction. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics, ACL 2018, Melbourne, Australia, July 15-20, 2018, Volume 1: Long Papers*, pages 934–945. Association for Computational Linguistics.

Noa Garcia, Mayu Otani, Chenhui Chu, and Yuta Nakashima. 2020. KnowIT VQA: answering knowledge-based questions about videos. In *The Thirty-Fourth AAAI Conference on Artificial Intelligence, AAAI 2020, The Thirty-Second Innovative Applications of Artificial Intelligence Conference, IAAI 2020, The Tenth AAAI Symposium on Educational Advances in Artificial Intelligence, EAAI 2020, New York, NY, USA, February 7-12, 2020*, pages 10826–10834. AAAI Press.

Ross B. Girshick. 2015. Fast R-CNN. In *2015 IEEE International Conference on Computer Vision, ICCV 2015, Santiago, Chile, December 7-13, 2015*, pages 1440–1448. IEEE Computer Society.

Kristen Grauman, Andrew Westbury, Eugene Byrne, Zachary Chavis, Antonino Furnari, Rohit Girdhar, Jackson Hamburger, Hao Jiang, Miao Liu, Xingyu Liu, et al. 2021. Ego4D: Around the world in 3,000 hours of egocentric video. *ArXiv preprint*, abs/2110.07058.

Lisa Anne Hendricks, Oliver Wang, Eli Shechtman, Josef Sivic, Trevor Darrell, and Bryan C. Russell. 2017. Localizing moments in video with natural language. In *IEEE International Conference on Computer Vision, ICCV 2017, Venice, Italy, October 22-29, 2017*, pages 5804–5813. IEEE Computer Society.

Deng Huang, Peihao Chen, Runhao Zeng, Qing Du, Mingkui Tan, and Chuang Gan. 2020. Location-aware graph convolutional networks for video question answering. In *The Thirty-Fourth AAAI Conference on Artificial Intelligence, AAAI 2020, The Thirty-Second Innovative Applications of Artificial Intelligence Conference, IAAI 2020, The Tenth AAAI Symposium on Educational Advances in Artificial Intelligence, EAAI 2020, New York, NY, USA, February 7-12, 2020*, pages 11021–11028. AAAI Press.

Po-Sen Huang, Chenglong Wang, Rishabh Singh, Wen-tau Yih, and Xiaodong He. 2018. Natural language to structured query generation via meta-learning. In *Proceedings of the 2018 Conference of the North*

American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 2 (Short Papers), pages 732–738, New Orleans, Louisiana. Association for Computational Linguistics.

Alon Jacovi and Yoav Goldberg. 2020. Towards faithfully interpretable NLP systems: How should we define and evaluate faithfulness? In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 4198–4205, Online. Association for Computational Linguistics.

Yunseok Jang, Yale Song, Youngjae Yu, Youngjin Kim, and Gunhee Kim. 2017. TGIF-QA: toward spatio-temporal reasoning in visual question answering. In *2017 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2017, Honolulu, HI, USA, July 21-26, 2017*, pages 1359–1367. IEEE Computer Society.

Jianwen Jiang, Ziqiang Chen, Haojie Lin, Xibin Zhao, and Yue Gao. 2020. Divide and conquer: Question-guided spatio-temporal contextual attention for video question answering. In *The Thirty-Fourth AAAI Conference on Artificial Intelligence, AAAI 2020, The Thirty-Second Innovative Applications of Artificial Intelligence Conference, IAAI 2020, The Tenth AAAI Symposium on Educational Advances in Artificial Intelligence, EAAI 2020, New York, NY, USA, February 7-12, 2020*, pages 11101–11108. AAAI Press.

Weike Jin, Zhou Zhao, Mao Gu, Jun Yu, Jun Xiao, and Yueting Zhuang. 2019. Multi-interaction network with object relation for video question answering. In *Proceedings of the 27th ACM International Conference on Multimedia*, MM '19, page 1193–1201, New York, NY, USA. Association for Computing Machinery.

K Kim, C Nan, MO Heo, SH Choi, and BT Zhang. 2016. PororoQA: Cartoon video series dataset for story understanding. In *Proceedings of NIPS 2016 Workshop on Large Scale Computer Vision System*, volume 15.

Kyung-Min Kim, Seong-Ho Choi, Jin-Hwa Kim, and Byoung-Tak Zhang. 2018. Multimodal dual attention memory for video story question answering. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 673–688.

Kyung-Min Kim, Min-Oh Heo, Seong-Ho Choi, and Byoung-Tak Zhang. 2017. DeepStory: Video story QA by deep embedded memory networks. In *Proceedings of the Twenty-Sixth International Joint Conference on Artificial Intelligence, IJCAI 2017, Melbourne, Australia, August 19-25, 2017*, pages 2016–2022. ijcai.org.

Ranjay Krishna, Kenji Hata, Frederic Ren, Li Fei-Fei, and Juan Carlos Niebles. 2017. Dense-captioning events in videos. In *IEEE International Conference on Computer Vision, ICCV 2017, Venice, Italy, October 22-29, 2017*, pages 706–715. IEEE Computer Society.

Chia-Hsuan Lee, Oleksandr Polozov, and Matthew Richardson. 2021. KaggleDBQA: Realistic evaluation of text-to-SQL parsers. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 2261–2273, Online. Association for Computational Linguistics.

Jie Lei, Licheng Yu, Mohit Bansal, and Tamara Berg. 2018. TVQA: Localized, compositional video question answering. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 1369–1379, Brussels, Belgium. Association for Computational Linguistics.

Jie Lei, Licheng Yu, Tamara Berg, and Mohit Bansal. 2020a. TVQA+: Spatio-temporal grounding for video question answering. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 8211–8225, Online. Association for Computational Linguistics.

Jie Lei, Licheng Yu, Tamara L Berg, and Mohit Bansal. 2020b. TVR: A large-scale dataset for video-subtitle moment retrieval. In *European Conference on Computer Vision*, pages 447–463. Springer.

Wei Li, Can Gao, Guocheng Niu, Xinyan Xiao, Hao Liu, Jiachen Liu, Hua Wu, and Haifeng Wang. 2021a. UNIMO: Towards unified-modal understanding and generation via cross-modal contrastive learning. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 2592–2607, Online. Association for Computational Linguistics.

Xiangpeng Li, Jingkuan Song, Lianli Gao, Xianglong Liu, Wenbing Huang, Xiangnan He, and Chuang Gan. 2019. Beyond rnns: Positional self-attention with co-attention for video question answering. In *The Thirty-Third AAAI Conference on Artificial Intelligence, AAAI 2019, The Thirty-First Innovative Applications of Artificial Intelligence Conference, IAAI 2019, The Ninth AAAI Symposium on Educational Advances in Artificial Intelligence, EAAI 2019, Honolulu, Hawaii, USA, January 27 - February 1, 2019*, pages 8658–8665. AAAI Press.

Zhuang Li, Lizhen Qu, Shuo Huang, and Gholamreza Haffari. 2021b. Few-shot semantic parsing for new predicates. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*, pages 1281–1291, Online. Association for Computational Linguistics.

Chin-Yew Lin. 2004. ROUGE: A package for automatic evaluation of summaries. In *Text summarization branches out*, pages 74–81.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019.

RoBERTa: A robustly optimized BERT pretraining approach. *ArXiv preprint*, abs/1907.11692.

Tegan Maharaj, Nicolas Ballas, Anna Rohrbach, Aaron C. Courville, and Christopher Joseph Pal. 2017. A dataset and exploration of models for understanding video data through fill-in-the-blank question-answering. In *2017 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2017, Honolulu, HI, USA, July 21-26, 2017*, pages 7359–7368. IEEE Computer Society.

Mateusz Malinowski and Mario Fritz. 2014. A multi-world approach to question answering about real-world scenes based on uncertain input. In *Advances in Neural Information Processing Systems 27: Annual Conference on Neural Information Processing Systems 2014, December 8-13 2014, Montreal, Quebec, Canada*, pages 1682–1690.

Subhabrata Mukherjee and Ahmed Awadallah. 2020. Uncertainty-aware self-training for few-shot text classification. *Advances in Neural Information Processing Systems*, 33:21199–21212.

Jonghwan Mun, Paul Hongsuck Seo, Ilchae Jung, and Bohyung Han. 2017. MarioQA: Answering questions by watching gameplay videos. In *IEEE International Conference on Computer Vision, ICCV 2017, Venice, Italy, October 22-29, 2017*, pages 2886–2894. IEEE Computer Society.

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. BLEU: a method for automatic evaluation of machine translation. In *Proceedings of the 40th annual meeting of the Association for Computational Linguistics*, pages 311–318.

Jonas Pfeiffer, Gregor Geigle, Aishwarya Kamath, Jan-Martin Steitz, Stefan Roth, Ivan Vulić, and Iryna Gurevych. 2022. xGQA: Cross-lingual visual question answering. In *Findings of the Association for Computational Linguistics: ACL 2022*, pages 2497–2511, Dublin, Ireland. Association for Computational Linguistics.

Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, Gretchen Krueger, and Ilya Sutskever. 2021. Learning transferable visual models from natural language supervision. In *Proceedings of the 38th International Conference on Machine Learning, ICML 2021, 18-24 July 2021, Virtual Event*, volume 139 of *Proceedings of Machine Learning Research*, pages 8748–8763. PMLR.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J. Liu. 2020. Exploring the limits of transfer learning with a unified text-to-text transformer. *Journal of Machine Learning Research*, 21(140):1–67.

Pranav Rajpurkar, Jian Zhang, Konstantin Lopyrev, and Percy Liang. 2016. SQuAD: 100,000+ questions for machine comprehension of text. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 2383–2392, Austin, Texas. Association for Computational Linguistics.

Lance Ramshaw and Mitch Marcus. 1995. Text chunking using transformation-based learning. In *Third Workshop on Very Large Corpora*.

Mengye Ren, Ryan Kiros, and Richard S. Zemel. 2015. Exploring models and data for image question answering. In *Advances in Neural Information Processing Systems 28: Annual Conference on Neural Information Processing Systems 2015, December 7-12, 2015, Montreal, Quebec, Canada*, pages 2953–2961.

Yibo Sun, Duyu Tang, Nan Duan, Yeyun Gong, Xiaocheng Feng, Bing Qin, and Daxin Jiang. 2020. Neural semantic parsing in low-resource settings with back-translation and meta-learning. *Proceedings of the AAAI Conference on Artificial Intelligence*, 34(05):8960–8967.

Makarand Tapaswi, Yukun Zhu, Rainer Stiefelhagen, Antonio Torralba, Raquel Urtasun, and Sanja Fidler. 2016. MovieQA: Understanding stories in movies through question-answering. In *2016 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2016, Las Vegas, NV, USA, June 27-30, 2016*, pages 4631–4640. IEEE Computer Society.

Du Tran, Maksim Bolonkin, Manohar Paluri, and Lorenzo Torresani. 2016. VideoMCC: a new benchmark for video comprehension. *ArXiv preprint*, abs/1606.07373.

Junbin Xiao, Xindi Shang, Angela Yao, and Tat-Seng Chua. 2021. Next-qa: Next phase of question-answering to explaining temporal actions. In *IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2021, virtual, June 19-25, 2021*, pages 9777–9786. Computer Vision Foundation / IEEE.

Dejing Xu, Zhou Zhao, Jun Xiao, Fei Wu, Hanwang Zhang, Xiangnan He, and Yueting Zhuang. 2017. Video question answering via gradually refined attention over appearance and motion. In *Proceedings of the 25th ACM International Conference on Multimedia*, MM '17, page 1645–1653, New York, NY, USA. Association for Computing Machinery.

Hongyang Xue, Zhou Zhao, and Deng Cai. 2017. Unifying the video and question attentions for open-ended video question answering. *IEEE Transactions on Image Processing*, 26(12):5656–5666.

Shaohua Yang, Qiaozi Gao, Sari Saba-Sadiya, and Joyce Yue Chai. 2018. Commonsense justification for action explanation. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing, Brussels, Belgium, October 31 - November 4, 2018*, pages 2627–2637. Association for Computational Linguistics.

Yunan Ye, Zhou Zhao, Yimeng Li, Long Chen, Jun Xiao, and Yueting Zhuang. 2017. Video question answering via attribute-augmented attention network learning. In *Proceedings of the 40th International ACM SIGIR Conference on Research and Development in Information Retrieval, Shinjuku, Tokyo, Japan, August 7-11, 2017*, pages 829–832. ACM.

Ting Yu, Jun Yu, Zhou Yu, Qingming Huang, and Qi Tian. 2020. Long-term video question answering via multimodal hierarchical memory attentive networks. *IEEE Transactions on Circuits and Systems for Video Technology*, 31(3):931–944.

Ting Yu, Jun Yu, Zhou Yu, and Dacheng Tao. 2019a. Compositional attention networks with two-stream fusion for video question answering. *IEEE Transactions on Image Processing*, pages 1204–1218.

Zhou Yu, Dejing Xu, Jun Yu, Ting Yu, Zhou Zhao, Yueting Zhuang, and Dacheng Tao. 2019b. ActivityNet-QA: A dataset for understanding complex web videos via question answering. In *The Thirty-Third AAAI Conference on Artificial Intelligence, AAAI 2019, The Thirty-First Innovative Applications of Artificial Intelligence Conference, IAAI 2019, The Ninth AAAI Symposium on Educational Advances in Artificial Intelligence, EAAI 2019, Honolulu, Hawaii, USA, January 27 - February 1, 2019*, pages 9127–9134. AAAI Press.

Amir Zadeh, Michael Chan, Paul Pu Liang, Edmund Tong, and Louis-Philippe Morency. 2019. Social-IQ: A question answering benchmark for artificial social intelligence. In *IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2019, Long Beach, CA, USA, June 16-20, 2019*, pages 8807–8817. Computer Vision Foundation / IEEE.

Rowan Zellers, Yonatan Bisk, Ali Farhadi, and Yejin Choi. 2019. From recognition to cognition: Visual commonsense reasoning. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.

Rowan Zellers, Ximing Lu, Jack Hessel, Youngjae Yu, Jae Sung Park, Jize Cao, Ali Farhadi, and Yejin Choi. 2021. MERLOT: Multimodal neural script knowledge models. In *Advances in Neural Information Processing Systems 34*.

Kuo-Hao Zeng, Tseng-Hung Chen, Ching-Yao Chuang, Yuan-Hong Liao, Juan Carlos Niebles, and Min Sun. 2017. Leveraging video descriptions to learn video question answering. In *Proceedings of the Thirty-First AAAI Conference on Artificial Intelligence, February 4-9, 2017, San Francisco, California, USA*, pages 4334–4340. AAAI Press.

Peng Zhang, Yash Goyal, Douglas Summers-Stay, Dhruv Batra, and Devi Parikh. 2016. Yin and yang: Balancing and answering binary visual questions. In *2016 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2016, Las Vegas, NV, USA, June 27-30, 2016*, pages 5014–5022. IEEE Computer Society.

Wenqiao Zhang, Siliang Tang, Yanpeng Cao, Shiliang Pu, Fei Wu, and Yueting Zhuang. 2019. Frame augmented alternating attention network for video question answering. *IEEE Transactions on Multimedia*, 22(4):1032–1041.

Zhou Zhao, Qifan Yang, Deng Cai, Xiaofei He, and Yueting Zhuang. 2017. Video question answering via hierarchical spatio-temporal attention networks. In *Proceedings of the Twenty-Sixth International Joint Conference on Artificial Intelligence, IJCAI 2017, Melbourne, Australia, August 19-25, 2017*, pages 3518–3524. ijcai.org.

Zhou Zhao, Zhu Zhang, Shuwen Xiao, Zhou Yu, Jun Yu, Deng Cai, Fei Wu, and Yueting Zhuang. 2018. Open-ended long-form video question answering via adaptive hierarchical reinforced networks. In *Proceedings of the Twenty-Seventh International Joint Conference on Artificial Intelligence, IJCAI 2018, July 13-19, 2018, Stockholm, Sweden*, pages 3683–3689. ijcai.org.

Yaoyao Zhong, Wei Ji, Junbin Xiao, Yicong Li, Weihong Deng, and Tat-Seng Chua. 2022. Video question answering: Datasets, algorithms and challenges. *arXiv preprint arXiv:2203.01225*.

Linchao Zhu, Zhongwen Xu, Yi Yang, and Alexander G Hauptmann. 2017. Uncovering the temporal context for video question answering. *International Journal of Computer Vision*, 124(3):409–421.

Yuke Zhu, Oliver Groth, Michael S. Bernstein, and Li Fei-Fei. 2016. Visual7W: Grounded question answering in images. In *2016 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2016, Las Vegas, NV, USA, June 27-30, 2016*, pages 4995–5004. IEEE Computer Society.

A Annotation Details

A.1 Video Selection and Processing

Video Selection. For the video selection part, as mentioned in Section 3, first, we identify 5 domains, ***Agriculture***, ***Geography***, ***Human Survival***, ***Natural Disasters***, and ***Military***, to collect videos recorded in the outside world. We then identify eight (8) YouTube channels and crawl videos from those channels. During crawling, we manually substitute irrelevant videos such as advertisements with videos that contain scenes mostly recorded in the outside world from the same channel.

Video Processing. As mentioned in Section 3, we clip the raw videos into short clips by PySceneDetect because the raw videos can be as long as an hour. We then concatenate these short clips so that the output video will be around 1 minute. **The output videos are used for the following annotation process.** We want to include longer videos because the videos recorded in the outside world usually contain less information compared to the videos about human interactions. Besides, if the concatenated video is at the end of the original video, it is allowed to be shorter than 1 minute. We select the concatenated videos that only contain scenes recorded in the outside world. If none of the concatenated videos satisfies, we manually clip the original videos to get an output video.

A.2 Annotation Instructions

As mentioned in Section 3, we have 2 phases in our annotation process as shown in Figure 3. In Phase 1, annotators come up with a hypothetical motivation, ask questions, and provide the corresponding answers with relevant parts of the video as evidence. Phase 2 is to collect answers and evidence for questions we collect in Phase 1. The following are the instructions for these two phases.

Instructions for Phase 1

> We need help for this Video QA task based on video content (including the audio).
> In this task, we suppose you can hypothetically send a robot to a place that you want, for many hours, so as to collect information that you need. In this hypothetical scenario, you have an objective that you want the robot to learn about. This robot can chart territory and is able to answer questions based on recorded videos. Therefore, after it comes back, you can ask questions to help you satisfy your objective, then this robot will provide you with answers, as well as video evidence clips to support the answers.
> In this task, to simplify, the provided videos represent places where you could potentially have sent the robot and are much shorter (a few minutes). Given a recorded video, please help us provide one hypothetical objective that makes sense with it, along with questions, answers, and evidence. Specifically, you should pretend to be both the information-seeker and the robot, which means that as the robot, you could watch the recorded video, and you should provide answers and video evidence clips; as the information-seeker, you have an objective, **not** watch the whole video (because of practical reasons), and you can only ask questions and receive answers and video evidence clips as feedback.
>
> 1. Basic Instructions
>
> - You will need to propose a hypothetical objective (or topic, intention, motivation), to motivate the questions, that makes sense for the given video.
> - You will need to provide as many questions as you need (to satisfy your objective) with regard to the content in the videos and that seek to understand more about the proposed objective.
> - You will first watch the video, but when you are providing the objective and questions, please pretend you **haven't** seen it before.
> - You will need to provide at least one question for each video. **The more the better**.
> - You will need to identify the source of your question (whether it is based on the visual scene or the audio) and classify your question accordingly.
> - You will need to provide the correct answer to the question you asked, as supported by the content in the video.
> - You will need to provide video evidence (video clip) to support your question and answer.
> - If one video doesn't make sense at all, or there's no possible objective for this video that makes sense, please comment at the bottom of this annotation page (and fill in

the mandatory fields for the corresponding video with placeholder values).

2. How To Propose Hypothetical Objective

   - For each video, you need to come up with a hypothetical objective (or intention, motivation, topic) that makes sense for this video, and briefly explain it.
   - Your questions should all relate to this objective.
   - Example 1:
     - Objective: I want to learn about the water in the territory.
     - Question 1: How big is the lake?
     - Question 2: Are there any boats in the lake?
     - Question 3: Where is the river?
     - ...
   - Example 2:
     - Objective: people/life movement
     - Question 1: Is there any sign that wildlife has passed this area?
     - Question 2: How much traffic is there on the road?
     - ...

3. How To Ask Your Question

   - Your question should relate to your proposed objective.
   - For each video, after you finish one question, you could click the Add one more question for this video button to continue to provide another question for this video. On the contrary, if you want to delete one question, you could click the Delete this Question button.
   - Ask one question at a time.
     - E.g., *"Are there any people? What are they doing?"* is not appropriate.
   - When you provide multiple questions for the same video, make sure these questions are **independently** asked.
     - E.g., "What is growing on pine trees?" and "What is their color?" are not independent.
   - The answer should be derived from the video (visual or audio).
     - E.g., *"Why do they run every morning?"* is not a good question.
   - Ask from the 3rd person point of view.
     - E.g., "What do *we* have on this farm?" -> "What do *They* have on this farm?"
   - Try to balance the questions such that the answers are not too repetitive (E.g., too many 'yes' answers).
   - Ask questions matter-of-factly (as **objectively** as possible). Stick to what you can see or hear from the video.
     - E.g., *"Does it make people feel good here?"* is somehow subjective.
   - Don't ask questions about how's the video being recorded, the camera-person or the camera itself. Ask about the content itself. Ignore what the camera-person is doing.
     - E.g., "What's the *cameraman* doing?" / "How fast is the *camera* moving?" are not good questions.

4. How to identity the Question Category

   We have some basic categories: **Motion, Spatial Relationship, Temporal Relationship, Reasoning, Number, Entity, Existence, Time, Location, Other.**
   If your questions fall into **multiple categories**, please check all categories that apply.
   Here are some example questions under each category:

   - **Motion**: What is the group of soldiers doing?
   - **Spatial Relationship**: What is driving beside the motorcycle?
   - **Temporal Relationship**: What happens before the black smoke rises?
   - **Reasoning**: What makes changing between targets possible for the missile?
   - **Number**: How many fighters are flying?
   - **Entity**: What is the target of the bullet?
   - **Existence**: Is there a lake by the mountain?
   - **Time**: How long can the missile fly?
   - **Location**: Where is the tank?
   - **Others**

5. How To Provide Correct Answer

   - Your answer should be written as **full sentences** (at least one).
     - E.g., *"Left" -> "The landspout bends toward the left."*

- The answer should be derived from the video (visual or audio).
  - E.g., "*These plants are green because they contain chlorophyll.*" is not a good answer.
- Provide answers matter-of-factly (as objectively as possible). Stick to what you can see or hear from the video.
  - E.g., "*beautiful*" is likely not a good word to use within an answer.
  - E.g., "*This takes some bravery to do.*" is somehow subjective.
- Don't answer about how's the video being recorded, the camera-person, or the camera itself. Answer about the content itself. Ignore what the camera-person is doing.
  - E.g., "There are two people, i.e. a running child, and the *cameraman*." is not a good answer.
- When you enter numbers, please enter digits instead of text.
  - "*Seventeen*" -> "*17*"

6. How to provide video evidence

- The video evidence consists of **all** the parts of the video that support the answer to your given question.
- You need to provide at least one video evidence clip (intervals within the video) for each question.
- You need to provide both the **start point and end point** for all the video evidence you identify in the video;
- You could use your mouse or ←/→ key to **click or drag the process bars** of start point and end point. When you click or drag the bar, the above video will change accordingly, so you could locate the points according to the video screen.
- For each video evidence clip, the end point should be **greater than zero**, and the end point should be greater or equal to the start point.
- The video evidence clips (the time gap between the start point and the end point) should be as short as possible.

Instructions for Phase 2

We need help for this Video Question Answering task based on video content (including the audio).

1. Basic Instructions

- You will first watch the video, then answer the questions, each question in turn.
- You will need to provide at least one answer for each question (ignoring differences such as upper/lower case, or the article). **The more answers the better**, but every answer should be correct.
- You will need to identify the source of your answer (whether it is based on the visual scene or the audio).
- For each answer, you will need to provide video evidence (video clip) to support your answers. See below for additional information.
- If one video or question is not available, please comment at the bottom of this annotation page (and fill the mandatory fields for this video/question with placeholder values).
- There are five questions, you need to finish all five questions according to the content in the video (including audio).

2. How To Answer

- Provide one or more answers for each question.
- Each answer should be written as full sentences (at least one).
  - E.g., "Left" -> "The landspout bends toward the left."
- The answer should be derived from the video (visual or audio).
  - E.g., "These plants are green because they contain chlorophyll." is not a good answer.
- Respond matter-of-factly (as objectively as possible). Stick to what you can see or hear from the video.
  - E.g., "beautiful" is likely not a good word to use within an answer.
  - E.g., "This takes some bravery to do." is somehow subjective.
- Answer in 3rd person point of view.
  - E.g.,"We raise cattle on this farm." -> "They raise cattle on this farm."
- Don't answer about how's the video being recorded, the camera-person, or the camera itself. Answer about the content itself. Ignore what the camera-person is doing.

  - E.g., "There are two people, i.e. a running child, and the cameraman." / "The camera is moving fast." are not good answers.
- When you enter numbers, please enter digits instead of text.
  - "Seventeen" -> "17"
- Use your best judgment.

3. How to provide video evidence

- The video evidence consists of all the frame intervals of the video that support the answer to your given question.
- You need to provide at least one video evidence clip (interval within the video) for each question.
- You need to provide both the start point and end point for all the video evidence you identify in the video;
- You can use your mouse or ←/→ key to click or drag the process bars of the start point and end point. When you click or drag the bar, the above video will change accordingly, so you could locate the points according to the video screen.
- For each video evidence clip, the end point should be greater than zero, and the end point should be greater or equal to the start point.
- The video evidence clips (the time gap between the start point and the end point) should only cover the actual evidence and not more (in other words, it should be as short as possible).

A.3 Annotation Interface

Figure 8 shows the annotation interface for Phase 1. Figure 9 shows the annotation interface for Phase 2.

A.4 Pilot Study Comparison between Annotations from Experts v.s. Non-Expert

Before the formal annotation, we compare the non-experts and experts' annotations for both phases. For Phase 1, we randomly selected 45 videos from each domain to be annotated by both the experts and crowdworkers. Following Castro et al. (2022), we set the AWS annotation qualification as HIT approve rate >92%, the number of HITs approved >1000, the location is either Canada or U.S., and the reward as $6/HIT (around $9/h).

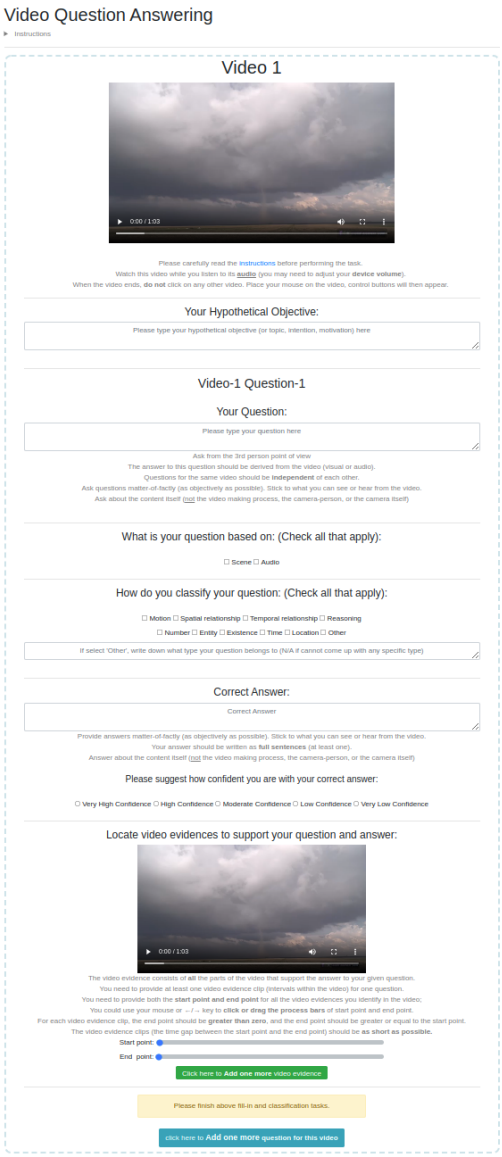

Figure 8: Interface for Phase 1 annotation. After watching the video, annotators provide a **motivation**, ask **questions** and provide corresponding **answers** by filling the blank. They provide parts of the videos as **evidence** to support each of the question-answer pairs by dragging the moving bar.

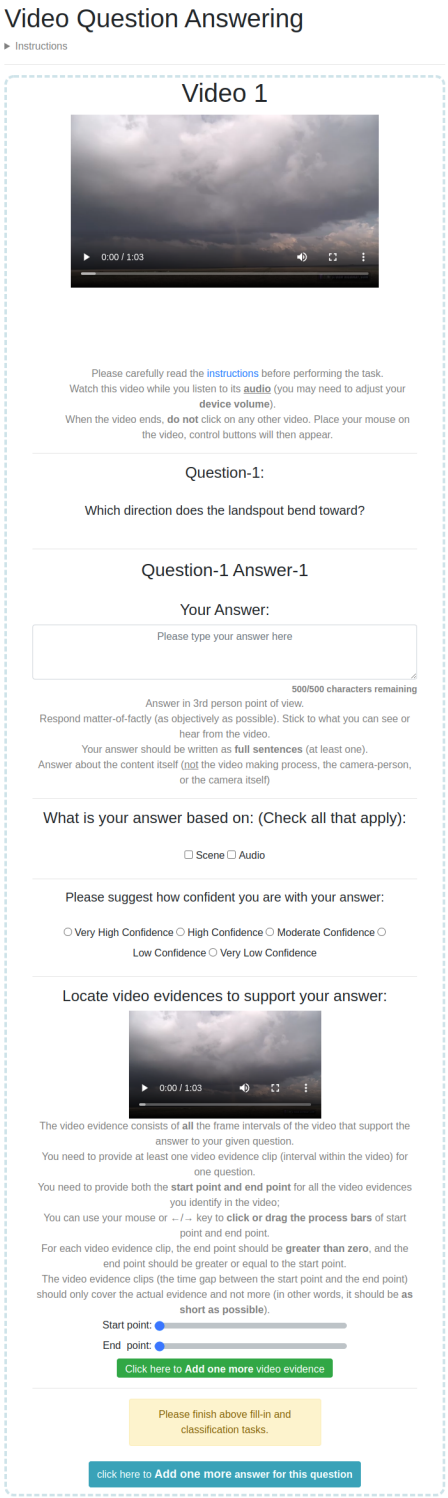

Figure 9: Interface for Phase 2 annotation. After watching the video and given the question from Phase 1, annotators provide **answers** with the corresponding **evidence**.

	R	I	P	Overall
expert	2.7	2.5	2.1	2.4
crowd	0.8	0.7	0.5	0.7

Table 8: Average scores of the pilot study for Phase 1 (from 0 to 3). **R**: Relevance; **I**: Interestingness; **P**: Professionality; **Overall**:Overall Score

After annotation, two authors of this paper who do not know the source of annotation evaluate and score in terms of Relevance, Interestingness, and Professionality for each annotation from 0 to 3. We define Relevance, Interestingness, and Professionality as follows:

- **Relevance**: how relevant a question and an answer are to the video. Good relevance indicates that the question is related to the video and focuses on the major events, objects, or people in the video. A relevant answer should address the question and can be derived from this video.
- **Interestingness**: whether the question interests you. In other words, whether you are interested in the question and answer, given a video.
- **Professionality**: how detailed and precise the question and answer are. Good professionality can be demonstrated by the precise usage of terminologies and numbers, and accurate description in the answer.
- **Overall Score**: the average score of the score for Relevance, Interestingness, and Professionality.

For each category, the higher score indicates the better the annotation demonstrates that characteristic. Table 8 lists the scores and Table 9 presents some annotation examples. From both the empirical and numerical results, we could see there is a significant quality gap for the annotation from experts versus from crowdworkers. Therefore, we decide to employ domain experts for Phase 1.

For Phase 2, we randomly select 104 *Geography* videos and questions from the questions annotated in Phase 1 to be annotated by both experts and crowdworkers. Moreover, we set the reward as \$3/HIT(around \$9/h) and employ the AWS **Master**[2] as the crowdworkers. Table 10 lists the results of the pilot study for Phase 2. According to Table 10, crowdworkers perform similarly to experts

[2] https://www.mturk.com/worker/help#what_is_master_worker

	Objective	Question	Answer
E	Precipitation	What types of precipitation are occurring?	Rain and hail.
C	Very like	Nice	Nice
E	I want to learn about the people	What type of weapons are they carrying?	M4's
C	The soldiers are caught on the ship.	What they are doing in this video?	They caught the ship.
E	Storm	Where is the storm?	In a field.
C	Motivation	5	Very amazing

Table 9: Examples in pilot study for Phase 1. **E**: Expert; **C**: Crowd

Annotator	R1	R2	RL	IOU-F1
Expert	23.63	8.05	21.22	12.24
Crowd	20.03	3.24	17.69	8.50

Table 10: ROUGE and IOU-F1 scores for the pilot study in Phase 2. Note that the scores here are lower than the scores for the human baselines in Tables 4 and 5. This is because we only compare the collected answers to a single answer here, while in Tables 4 and 5 we calculate the average scores of one annotator against the remaining as described in Section 4.

in Phase 2. Considering the annotation efficiency, we decide to employ both experts and crowdworkers to annotate more diversified answers for each question in Phase 2. Note that the ROUGE scores in Table 10 are lower than the scores for the human baselines in Tables 4 and 5. This is because we only compare the collected answers to a single answer in Table 10, while in Tables 4 and 5, we calculate the average scores of one annotator against the remaining as described in Section 4.

A.5 Question and Answer Correction

After we collect annotation from Phase 1, the authors of this paper check the quality of the collected question and answers and modify the question and answers accordingly. Specifically, we:

- Delete the questions that can be answered without watching the video (e.g. `Q: "If water can get through the hut's roof; can the wind go through the hut's roof?", A: "Yes the wind can go through the hut's roof."`)

- Modify the question or the answer to 3rd person view (e.g. change `Q: "Do we have aircraft that we can do a touch and go landing like a helicopter?"` to `Q: "Do they have aircraft that can do a touch and go landing like a helicopter."`)

Annotator ID	Expertise	Assigned Domains (# Q)
0	Geography	Geography (94) ; Natural Disaster (187)
1	Geography	Geography (16) ; Human Survival (74)
2	Veteran	Military (26) ; Human Survival (146)
3	Veteran	Military (70) ; Human Survival (89)
4	Veteran	Military (12)
5	Veteran	Military (8)
6	Veteran	Military (85)
7	Biology	Agriculture (88)
8	Biology	Agriculture (21)

Table 11: Information for expert annotators who annotate the questions, together with their assigned domains and number of questions (# Q) in the parentheses.

- Exclude the man holding the camera in the answer if it is a first-person view video.

- Modify questions that are not independently asked (e.g.`"Where are they?"`, where "they" refers to the "paved and unpaved roads" in the previous question. Therefore, we change the question to `"Where are the roads?"`)

- Split questions that include multiple sub-questions into several questions.

Some of the annotators from Phase 2 do not annotate any evidence (leaving the evidence from the start to the end of the video). Thus, we empirically filter out evidence longer than $1/4$ of the video.

A.6 Annotator Information

Table 11 shows the expertise of each expert, together with their assigned domains of annotation and the number of questions they annotate in their assigned domains in Phase 1.

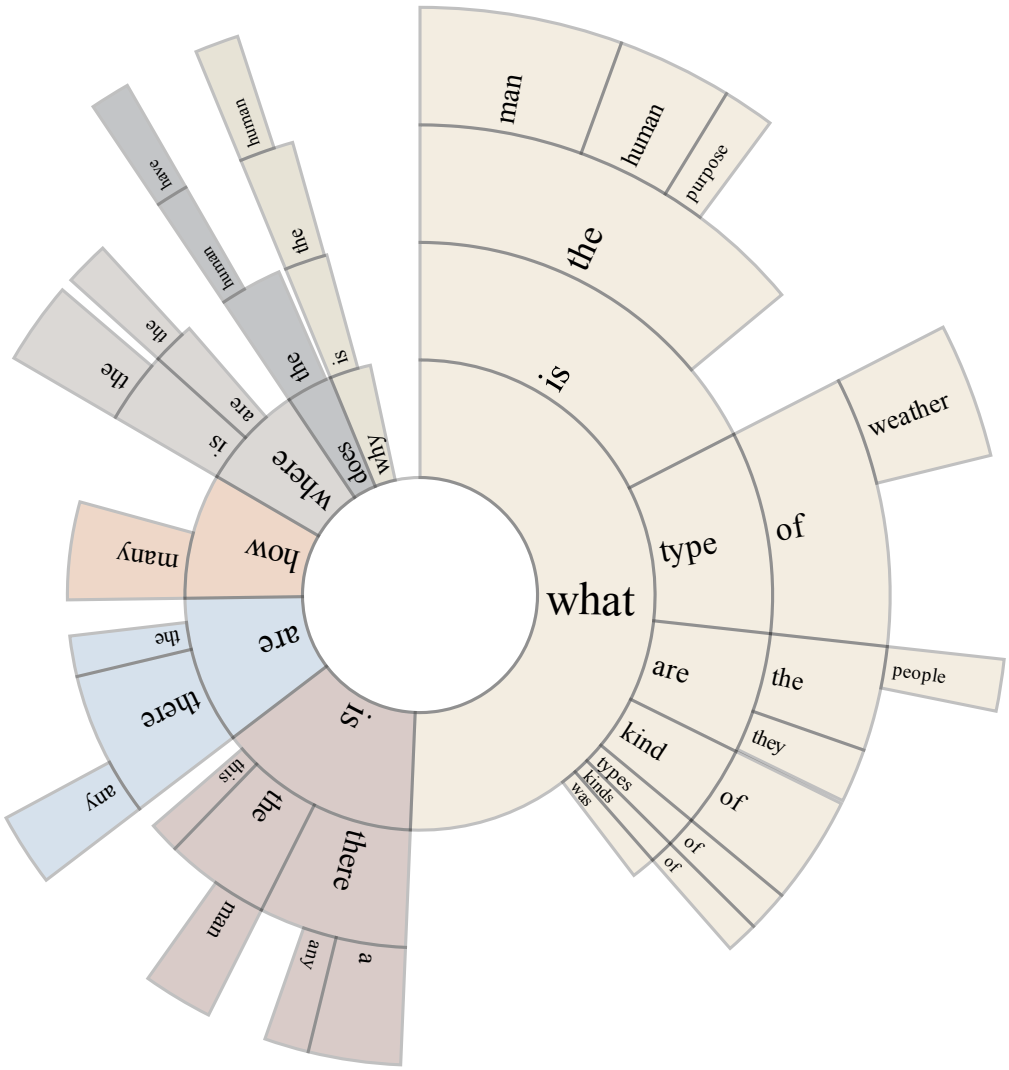

Figure 10: Distribution of questions by the first four tokens. The ordering of words starts from the center to outside.

Domain	top1	top2	top3
Agriculture	farm	agricultural	understand
Natural Disaster	weather	people	flooding
Human Survival	man	determine	human
Geography	people	topography	water
Military	military	aircraft	determine

Table 12: Most common 3 words for each domain after removing stop-words.

A.7 Dataset Analysis

Figure 10 presents question distributions in terms of words.

Questions Types. Table 12 examines the frequent words for each domain, which demonstrates the characteristics of the domain. Take *Natural Disaster* as an example, the 3 most frequent words are used in 20.63% of sentences. Besides, Figure 4 in Section 3 lists the annotators' self-reported question types. One thing we observe is that questions that start with "What" possess a large proportion of all the questions. Such questions might be hard to classify into certain question typs (Castro et al., 2020), so we allow annotators

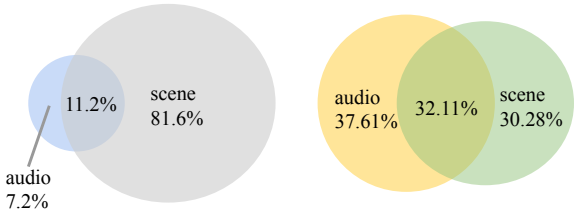

Figure 11: Venn diagrams showing whether the question depends on visual (scene) or audio from the original video. The left is for the entire dataset, while the right is for the *Agriculture* domain.

to choose multiple question types for a single question. Empirically speaking, questions that start with "is(are)"/"where"/"how many" are commonly relevant to "Existence"/"Location"/"Number" questions. In our dataset, their distribution trend ("is(are)": 24.13% > "where": 7.21% > "how many": 4.48%) is akin to the trend of the distribution of the reported question types ("Existence": 45.20% > "Location": 12.23% > "Number": 4.59%). Moreover, although we have "human", "man" and "people" as the most frequent words in some domains, the most frequent words in domains such as *Military* are "military", and "aircraft", which demonstrates that our dataset does not only focus on human interactions as most of the existing datasets do.

Information Needed. As shown in the left Venn figure in Figure 11, generally, most questions are based on the visual (scene). Such a distribution is also justified by the distribution of the question types. The dominant question types we have in Figure 4 are *Motion*, *Spatial*, *Existence* and *Entity*, which typically focus on visual information. However, in *Agriculture* (the right Venn figure in Figure 11), the audio-based questions take more portion, because videos in *Agriculture* usually focus on farming tips, instructions for using tools, etc. In this paper, we do not experiment with models that use audio or transcripts from the video. Future research might look into letting models use audio and transcripts on our dataset.

Answer Similarity/Diversity. We have similar and diversified answers collected in our dataset. Figure 5 gives 2 examples: answers from the upper example are similar to each other; for the lower example, answers diverse a lot between Phase 1 and Phase 2 annotations or even within Phase 2. However, all of the answers are acceptable given the video. The similarity demonstrates the reliability of the Phase 2 annotation. Meanwhile, the diversified answers help to better evaluate models.

Videos	369
Duration (s)	71.22 ± 26.47
Questions	916
Question per video	2.48 ± 1.38
Question length (#tokens)	7.09 ± 2.60
Answer length (#tokens)	8.62 ± 8.90
Evidence per answer	1.53 ± 0.76
Evidence length (s)	9.09 ± 13.45

Table 13: Annotation statistics for Phase 1. "#tokens" represent the number of tokens.

Crowd annotated answers	932
Expert annotated answers	182
Total	1114
Answer per question	1.22 ± 0.69
Answer length (#tokens)	9.45 ± 7.46
Evidence per answer	0.89 ± 0.72
Evidence length (s)	10.43 ± 5.81

Table 14: Annotation statistics for Phase 2. "#okens" represents the number of tokens.

B Annotation Statistics

Tables 13 and 14 list the statistics for annotation in Phase 1 and Phase 2, respectively.

C Details of Multi-task Learning

Tables 15 and 16 report the model performances under different sets of α, β for Equation (1). We highlight the rows we report in Table 4 in Section 4.2, Table 4 in Section 4.2, Table 5 in Section 5.2, and Table 5 in Section 5.2.

D Experiment Results

Figures 12 and 13 report `Multi-Task` model's performance on Video QA by ROUGE-2, and ROUGE-L, respectively. Figure 14 demonstrates that ROUGE scores follow a similar trend as mentioned in Section 6.

D.1 Ablation Study on Video Evidence Selection

To investigate whether the vision part is indeed needed by the baseline models for the Video Evidence Selection task, we conduct an ablation study

β	R1	R2	RL	IOU-F1
0.5	**33.8 ± 0.8**	**18.5 ± 0.7**	**32.5 ± 0.8**	**3.7 ± 2.4**
1.0	32.2 ± 0.7	17.6 ± 0.5	31.0 ± 0.6	1.9 ± 1.7
1.5	33.8 ± 0.3	18.0 ± 0.9	32.5 ± 0.3	1.5 ± 0.1

Table 15: We set $\alpha = 1$ throughout all the experiments, and report the corresponding $\texttt{Multi}_{\text{T+V,SE}}$ performances on Video QA (ROUGE scores) and Video Evidence Selection (IOU-F1 scores). We highlight the row we report in Table 4 in Section 4.2 and Table 4 in Section 4.2.

β	R1	R2	RL	IOU-F1
0.5	**34.0 ± 0.5**	**18.8 ± 0.7**	**32.8 ± 0.6**	**1.2 ± 0.1**
1.0	33.4 ± 0.6	18.4 ± 0.2	32.1 ± 0.6	1.4 ± 0.3
1.5	32.8 ± 0.3	18.3 ± 0.3	31.7 ± 0.2	1.0 ± 0.2

Table 16: We set $\alpha = 1$ throughout all the experiments, and report the corresponding $\texttt{Multi}_{\text{T+V,IO}}$ performances on Video QA (ROUGE scores) and Video Evidence Selection (IOU-F1 scores). We highlight the row we report in Table 5 in Section 5.2 and Table 5 in Section 5.2.

Model name	IOU-F1
T5 $\text{IO}_{\text{random}}$	1.1 ± 0.3
T5 IO	1.1 ± 0.2
T5 $\text{SE}_{\text{random}}$	2.7 ± 1.9
T5 SE	**4.5 ± 0.8**

Table 17: Ablation study on Video Evidence Selection. We feed T5 $\text{IO}_{\text{random}}$ and T5 $\text{SE}_{\text{random}}$ the question concatenated with a random sequence, while we feed T5 IO and T5 SE the question with the actual video sequence.

using T5 IO and T5 SE (introduced in Section 5). We take a random sequence of the same length as the original video sequence and feed the random sequence instead of the original video sequence to the model. Table 17 shows the results of the comparison between these different settings. T5 IO performs roughly the same as T5 $\text{IO}_{\text{random}}$, which indicates that the model struggles to utilize visual information. T5 IO even underperforms the random baseline which can achieve an IOU-F1 score of 2.5 ± 0.3 (as shown in Table tab:few-shot-evidence-results-10-epochs). However, T5 SE outperforms T5 $\text{SE}_{\text{random}}$, suggesting that T5 SE uses visual features to locate the evidence of the question.

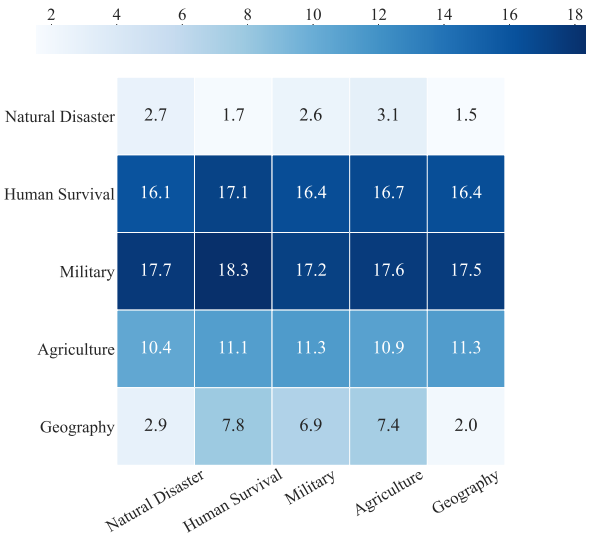

Figure 12: Multi-Task ROUGE-2 scores for Video QA when tuned on a single domain (y-axis) and tested against each domain (x-axis).

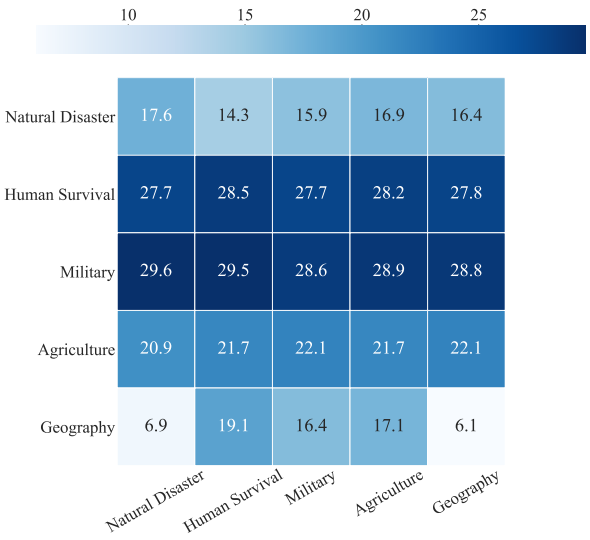

Figure 13: `Multi-Task` ROUGE-L scores for Video QA when tuned on a single domain (y-axis) and tested against each domain (x-axis).

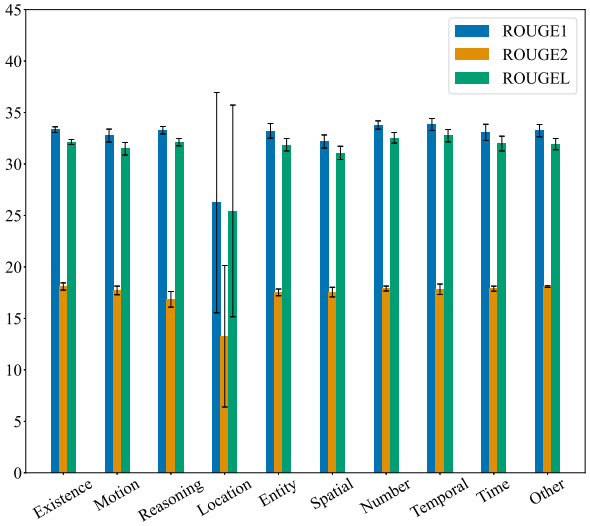

Figure 14: $\texttt{Multi}_{\texttt{T+V,SE}}$ performance on different question types for Video QA. For each question type, we report ROUGE-1, ROUGE-2, and ROUGE-L scores from left to right. We can see that different ROUGE scores follow similar trends, we only report ROUGE-1 in Table 6 in Section 6.

Towards Better Semantic Understanding of Mobile Interfaces

Srinivas Sunkara[1*] **Maria Wang**[1*] **Lijuan Liu**[1] **Gilles Baechler**[1]

Yu-Chung Hsiao[1] **Jindong (JD) Chen**[1] **Abhanshu Sharma**[1] **James Stout**[2]

[1]**Google Research,** [2]**Google**

Abstract

Improving the accessibility and automation capabilities of mobile devices can have a significant positive impact on the daily lives of countless users. To stimulate research in this direction, we release a human-annotated dataset with approximately 500k unique annotations aimed at increasing the understanding of the functionality of UI elements. This dataset augments images and view hierarchies from RICO, a large dataset of mobile UIs, with annotations for icons based on their shapes and semantics, and associations between different elements and their corresponding text labels, resulting in a significant increase in the number of UI elements and the categories assigned to them. We also release models using image-only and multimodal inputs; we experiment with various architectures and study the benefits of using multimodal inputs on the new dataset. Our models demonstrate strong performance on an evaluation set of unseen apps, indicating their generalizability to newer screens. These models, combined with the new dataset, can enable innovative functionalities like referring to UI elements by their labels, improved coverage and better semantics for icons etc., which would go a long way in making UIs more usable for everyone.

1 Introduction

Mobile devices like phones and tablets have become ubiquitous and indispensable to carry out our daily activities. It is not an exaggeration to say that usage of mobile devices is becoming a requirement for full participation in society. Recent reports from the WHO and others (Organization, 2021; Peter Ackland and Bourne, 2017) estimate that around 2.2 billion people across the world have some form of vision impairment, out of which 36 million people are blind. Accessibility of mobile devices is necessary for these visually impaired users to carry out their daily tasks and is an important tool for their social integration (Ladner, 2015).

Accessibility of mobile apps has improved significantly over the past few years aided by developments on two main fronts: Firstly, the development of screen readers, like VoiceOver (Apple, 2021c) on iOS and TalkBack (Accessibility, 2021e) on Android, enable visually impaired users to control their phone in an eyes-free manner. Secondly, development tools and standards to enhance accessibility, such as the Accessibility guidelines for iOS and Android (Accessibility, 2021a; Apple, 2021b), Android Accessibility Scanner (Accessibility, 2021b), and iOS Accessibility Inspector (Apple, 2021a), have helped developers identify and fix accessibility issues for applications. Among most of these utilities, the main source of accessibility data is the accessibility labels (Accessibility, 2021d) provided by app developers. These labels are specified as attributes on a structured representation such as View Hierarchy, for the different UI elements on the screen and are available to screen readers (Accessibility, 2021c; Apple, 2018). Despite the growth in accessibility tools, recent studies (Ross et al., 2020, 2017; Chen et al., 2020a) have found that even the most widely used apps have large gaps in accessibility. For instance, a study by Chen et al. (2020a) of more than 7k apps and 279k screens revealed that around 77% of the apps and 60% of the screens had at least one element without explicit labels. Similarly, Ross et al. (2020) found that, in a population of 10k apps, 53% of the Image Button elements were missing labels.

In this paper, we attempt to encourage further research into improving mobile device accessibility and increasing device automation by releasing an enhanced version of the RICO dataset (Deka et al., 2017) with high-quality human annotations aimed at semantic understanding of various UI elements. Firstly, following a study by Ross et al.

*Equal contribution, correspondence: {srinivasksun,mariawang}@google.com

Proceedings of the 29th International Conference on Computational Linguistics, pages 5636–5650
October 12–17, 2022.

(2020) where missing labels for Image Button instances was found to be the primary accessibility barrier, we focus on creating annotations useful for identifying icons. In particular, we annotated the most frequent 77 classes of icons based on their appearance. We refer to this task as the *Icon Shape* task. Secondly, we identified icon shapes which can have multiple semantic meanings and annotated each such icon with its semantic label. This task is called *Icon Semantics*. Some examples of such icons can be seen in Figure 1b. Finally, we annotate UI elements, like icons, text inputs, checkboxes etc., and associate them with their text labels. These associations can help us identify meaningful labels for the long tail of icons and UI elements not present in our schema, but having a textual label associated with them. We refer to this task as the *Label Association* task. The main contributions of this paper are as follows:

- A large scale dataset[1] of human annotations for 1) Icon Shape 2) Icon Semantics and 3) selected general UI Elements (icons, form fields, radio buttons, text fields) and their associated text labels on the RICO dataset.

- Strong benchmark models[2] based on state of the art models (He et al., 2016; Carion et al., 2020; Vaswani et al., 2017) using image-only and multimodal inputs with different architectures. We present an analysis of these models evaluating the benefits of using View Hierarchy attributes and optical character recognition (OCR) along with the image pixels.

2 Related Work

2.1 Datasets

Large scale datasets like ImageNet (Deng et al., 2009) played a crucial part in the development of Deep Learning models (Krizhevsky et al., 2012; He et al., 2016) for Image Understanding. Similarly, the release of the RICO dataset (Deka et al., 2017) enabled data driven modeling for understanding user interfaces of mobile apps. RICO is, to the best of our knowledge, the largest public repository of mobile app data, containing 72k UI screenshots and their View Hierarchies from 9.7k Android apps spanning 27 categories. Apart from RICO, other datasets include ERICA (Deka et al., 2016) with sequences of user interactions with mobile UIs and LabelDROID (Chen et al., 2020a) which contains 13.1k mobile UI screenshots and View Hierarchies.

There have been a few efforts to provide additional annotations on RICO. SWIRE (Huang et al., 2019) and VINS (Bunian et al., 2021) added annotations for UI retrieval, Enrico (Leiva et al., 2020) added annotations for 20 design topics. Liu et al. (2018) automatically generated semantic annotations for UI elements using a convolutional neural network trained on a subset of the data. Recently Li et al. (2022) released UI element labels on view hierarchy boxes including identifying boxes which do not match to any elements in the UI. Even though some of these works are similar in spirit to the dataset presented in this paper, there are two major differences: 1) The icon and UI element labels are *inferred* on the boxes extracted from the View Hierarchy, whereas, in our work, we add *human annotated* bounding boxes directly on the image. Due to noise in the view hierarchies like missing and misaligned bounding boxes for UI elements (Li et al., 2020a, 2022), we observe that human annotation increases the number of icons labelled by 47%. 2) The semantic icon labels in Liu et al. (2018) conflate appearance and functionality. For example "close" and "delete," "undo" and "back," "add" and "expand" are mapped to the same class, even though they represent different functionalities. The *Icon Semantics* annotations in our dataset specifically try to distinguish between icons with the same appearance but differences in functionality.

2.2 Models

Pixel based methods for UI understanding have been studied for a long time. They have been used for a variety of applications like GUI testing (Yeh et al., 2009; Chang et al., 2010), identifying similar products across screens (Bell and Bala, 2015), finding similar designs and UIs (Behrang et al., 2018; Bunian et al., 2021), detecting issues in UIs (Liu, 2020), generating accessibility metadata (Zhang et al., 2021) and generating descriptions of elements (Chen et al., 2020a). A recent study by Chen et al. (2020b) compares traditional image processing methods and Deep Learning methods to identify different UI elements. The image-only baseline models studied in this paper are based on Object Detection methods presented in Chen et al.

[1]The datasets are released at https://github.com/google-research-datasets/rico-semantics.

[2]Benchmark models and code are released at https://https://github.com/google-research/google-research/tree/master/rico-semantics

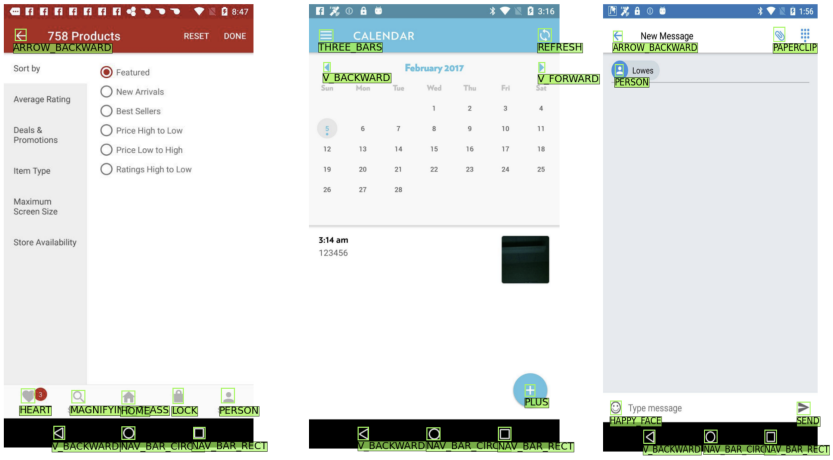

(a) Examples of 76 *icon shape* annotations. We include classes that reflect the social aspects of app usage, e.g., "person" for profile and community, share via popular apps such as Facebook and Twitter, etc.

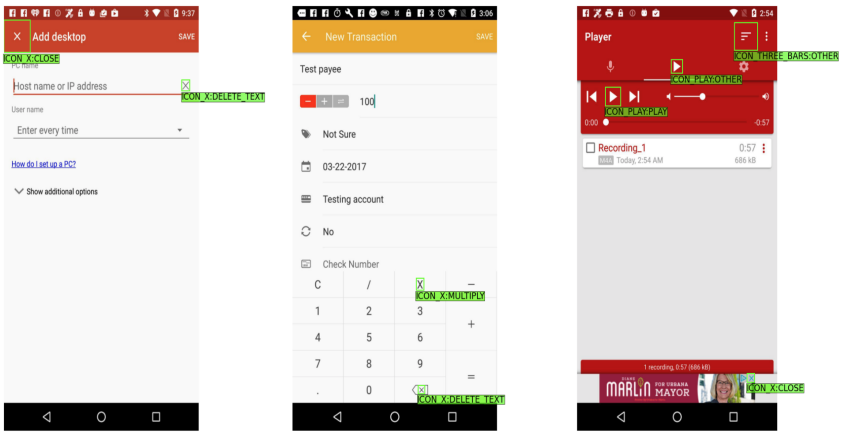

(b) Examples of 38 *icon semantics* annotations, in the format of <shape>:<semantics>. Note that a single shape may represent multiple semantics, depending on the context. E.g., "X" shape may mean "close", "delete text", or "multiply". We use an umbrella semantic class "OTHER" to cover the semantics not covered in our proposed set of classes

Figure 1: In this paper, we annotated the RICO dataset with both **icon shapes** and their **semantics**, to encourage further research on app automation and accessibility. The existing icon annotations from Liu et al. (2018) were algorithmically generated with 10% of them verified. However, we observed $\sim$ 32% were missing labels compared to our full human annotations. We release our annotations in the hope to contribute back to the community.

(2020b), Zhang et al. (2021), Chen et al. (2020a), and Carion et al. (2020).

Extending to other modalities beyond pixels, Banovic et al. (2012) use video tutorials to understand UIs and annotate them with additional information. Li et al. (2021) use only the screen information for identifying embeddings of UI elements. Hurst et al. (2010) use both the screen and accessibility API information to identify interaction targets in UIs and Chang et al. (2011) use similar inputs to detect and identify certain UI elements and Nguyen et al. (2018) use it for identifying similar UI designs. Multimodal inputs have also been used for understanding screen contents like generating element descriptions (Li et al., 2020b), training UI embeddings for multiple downstream tasks (He et al., 2021; Bai et al., 2021) and, denoising data, predicting bounding box types (Li et al., 2022).

3 Datasets, taxonomy and annotation

To enable accessible hands-free experience for mobile users, it is necessary for the system to understand the functionality of the different screen UI elements. For learning data driven models to enable these functionalities, we use the RICO dataset (Deka et al., 2017). RICO spans >9K apps and >72K UIs, each with a screenshot and information regarding the structure of the UI in the form of a View Hierarchy (VH). Besides the bounding boxes of the different UI elements, the VH contains useful

attributes like the *content description* and *resource id* which provide information regarding the functionality of the different UI elements. However, Li et al. (2020a) found that only 35% of the unique screens in RICO contain a matching View Hierarchy and screenshot. In the next few sections, we describe how we overcome the mismtach issue and describe the different type of annotations.

3.1 Icon Shape

As the study by Ross et al. (2020) found that one of the main accessibility barriers on mobile devices are missing labels for ImageButton elements, in the *icon shape* and *semantics* tasks we focus on creating annotations useful for identifying icons. Liu et al. (2018) attempted to provide semantic labels for the icons in the RICO dataset by identifying different concepts represented by the ImageButton elements on a subset of the data. Using these concepts and models trained on a subset of the data they identified semantic labels over the entire dataset. They inferred labels for more than 100 icons types, 25 UI element types and 197 text button types. However, due to the presence of view hierarchies with bounding boxes missing and misaligned for UI elements (Li et al., 2020a, 2022), these labels miss several UI elements. By comparing with manually labeled data, we found that the annotations in Liu et al. (2018) did not identify around 32% of all icon instances captured by manual labeling. For improving the coverage of icons, in our dataset we asked raters to manually annotate all the bounding boxes. We created a schema of the 77 most commonly used icon classes, reusing many of the classes identified in Liu et al. (2018). Examples of these icon classes and images are shown in Figure 1a.

3.2 Icon Semantics

To support voice driven usage of mobile devices, we identify icons not only based on their shape and appearance but also functionality and semantic meaning. For example, an "X" shaped icon can mean "close," "remove an option/entry," "delete/clear text," or "multiply." One way to enable users to refer to the various semantics is to map the multiple semantics to the same class in the *Icon Shape* schema. This approach has two limitations: 1) Mapping each icon shape to multiple semantics can lead to confusion for applications like Screen Readers. 2) We noticed that there are many instances of icons with different semantics but same shape occurring in the same screen. In particular 11% of all images with *ICON_PLAY*, 4.9% with *ICON_X* and 4.8% with *ICON_CHAT* have icons with multiple semantics.

We identified a list of common icons which have more than one semantic meaning associated with them by the following steps: 1) Manually inspect a variety of the icon annotations and list the functionality of each icon instance, 2) Use the plurality of words matching to the same icon shape in the RICO icon labels in (Liu et al., 2018) as an indicator of multiple semantics, and 3) Recognize confusing app logos. For example, the logo for WhatsApp contains a Phone icon but it is most natural for a user to say, "open WhatsApp".

After these steps, we arrived at a list of 12 shape icons which were further classified into 38 semantic shapes. For icons with semantic meanings not covered by our schema, we assign the semantic type *OTHER* as the default label. Out of the 101,625 annotated icons 15,640 (around 15%) are labeled as *OTHER*. We observed that it is difficult to cover all of the tail semantics classes with a schema. Thus, we also obtained annotations for the text labels associated with UI elements, described in the section below.

3.3 Label Association

Many UI elements have an associated text labels that best describes UI elements. Our data analysis showed that 24.6% of icons, form fields, check boxes and radio buttons have an associated label. However, we found that these labels are commonly neither syntactically associated within view hierarchies nor visually aligned in screenshot pixels. First, we attempted to identify labels associated with UI elements by using heuristics relying on the View Hierarchy, like searching the siblings and parent node's siblings for axis aligned text elements for each of the UI elements. We found that only 40% of UI elements with labels could be correctly associated using such heuristics with a significant number of false positive associations. Next, we attempted to match UI element bounding boxes with line boxes detected by OCR. We matched each UI element bounding box with the OCR text box nearest to the top left corner with a maximum distance threshold. This method achieved 29.5% accuracy. These empirical studies indicated that, like many machine perception tasks, this *L*abel Association task may in fact be non-trivial despite appearing

straightforward to humans. Some examples can be seen in figure 2. We believe this novel task can help address the limitation of annotations with a fixed set of classes by making use of the text label information present in the UI.

3.4 Annotation Procedure

For the three tasks described above, we follow the annotation procedure below to collect annotated data. We used a team of 40 trained human raters with single replication to annotate the screenshots. The raters were initially provided with example image patches for each icon or UI element type similar to Figure 1a and then followed the annotation procedure below:

- Round 1: Each rater is presented with a screenshot and is asked to draw a bounding box around every icon and UI element on the screen. For each icon box, they can specify an icon shape class if it is included in the schema and otherwise they classify it as a general icon class. Once an icon shape class is identified, if there are multiple semantics associated with that shape, the raters choose one semantic class among the different options for that class. These options also include the *OTHER* class to capture semantics not in the icon semantics schema. For the label association task, we ask raters to identify the icons, form fields, radio buttons, and check boxes first, followed by the text labels, by drawing their bounding boxes and assigning the respective classes. Then the raters group the text labels with their associated UI element if it exists.

- Round 2: To improve the annotation quality, we send the datasets for a second round of cleaning where the raters can adjust the bounding box or the classes assigned.

- Round 3: After round 2, if we still find some classes are poorly labeled by manual inspection, we use trained models to identify potential incorrect labels. We train Object Detection models on the entire dataset, and use the model to predict labels for the train, validation and test sets. If there are any differences between the model predictions and the human annotations, we identify these instances as potentially error-prone and send these images back to the raters to re-annotate them.

Using the above procedure, we annotated all of the screenshots in RICO with icon shape, semantics and label association classes. The distribution of top labels for the icon tasks can be seen in figures 3, 4. The entire taxonomy and exact counts are in Appendix A.

4 Baseline Models

We conducted several experiments to investigate the effectiveness of various deep learning approaches for solving the tasks presented in Section 3. The overall goal of these experiments is to: 1) provide good baseline models to be used for image-only and image+VH settings 2) study the effect of using multi-modal inputs v/s only the screenshot for these tasks.

4.1 Problem Setup

For the three tasks, we distinguish two different approaches: 1) the Object Detection (OD) approach using only the image and, 2) the bounding box classification (BB-CLS) approach using image, OCR and view hierarchy. We describe these two approaches in the following sections. We split the data into 80% train, 10% validation and 10% test by package name to avoid data leakage and use the same split for all experiments.

4.1.1 Object Detection (OD)

In this setup, the models take the screenshot as input and output bounding box, class label and score for each object found. We used object detection models based on the widely used and better performing (Chen et al., 2020b) Faster R-CNN (Ren et al., 2015) and Centernet (Zhou et al., 2019) architectures. We train these models with various backbones (Szegedy et al., 2017; He et al., 2016; Lin et al., 2017; Newell et al., 2016) and report results for the best performing ones. Finally, we experimented with Object Detection models based on Transformer (Vaswani et al., 2017) architecture like DETR (Carion et al., 2020; Zhu et al., 2020) to verify the hypothesis that for the *Icon Semantics* and the *Label Association* tasks, the models need more information from their context compared to the *Icon Shape* task. We use standard Object Detection metrics like mAP@0.5IOU to compare the model performance.

4.1.2 Bounding box classification (BB-CLS)

For this setup, we train models to classify bounding boxes extracted from the view hierarchy (VH)

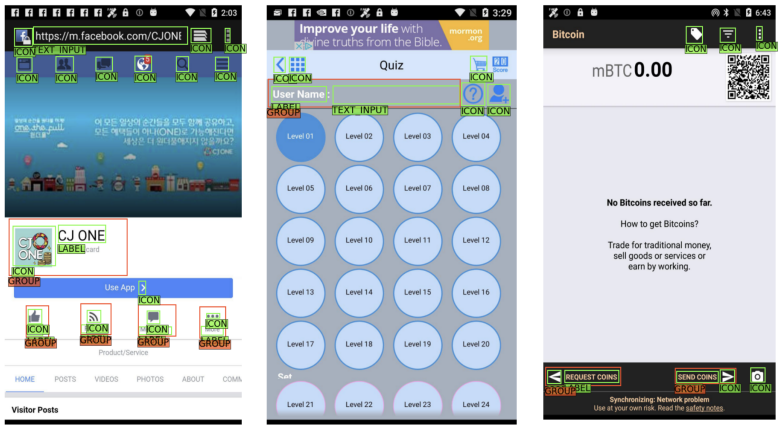

Figure 2: When an associated text label appears with a UI element, it is natural to refer to the icon using the text directly. 24.6% of icons, form fields, check boxes and radio buttons have an associated text label. *Label Association* examples are provided in the figure in red. However, rule-based approaches using view hierarchies or OCR can only achieve 40% and 29.5% accuracies respectively. These annotations enable us to identify the elements to interact with for voice commands like "Enter User Name as test user" or "Request Coins".

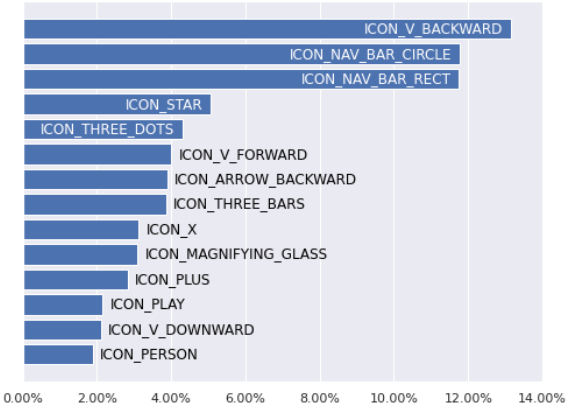

Figure 3: Distribution of the top 14 *icon shape* classes. These classes account for 72% of the total icons covered by the *icon shape* labels.

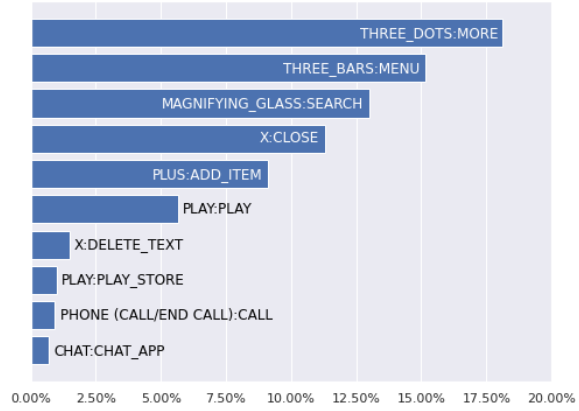

Figure 4: Distribution of the top 10 *icon semantics* classes. These classes account for 76% of the total icons covered by the *icon semantics* labels.

and assign them labels among the candidate classes. The groundtruth (GT) set contains boxes that have been created and labeled by crowd workers. These two sets of boxes are greedily matched as follows. 1) Each GT box is matched to at most one VH box and vice versa. 2) For every GT box, we find the VH box with maximum Intersection over Union (IoU). Only the matches for which the IoU value is greater than the threshold of *0.5* are kept. 3) Once a VH box is matched with a GT, it is not considered for future matches. After this matching procedure is complete, we assign a background class to the unmatched VH boxes. We use a Transformer (Vaswani et al., 2017) based network as described in UIBert (Bai et al., 2021) for the *Icon Shape*, *Semantics* tasks framed as a classification problem and compare models based on Macro F-1 score.

For the *Label Association* task, we compute an embedding of each UI element following Bai et al. (2021) and perform clustering on projected embeddings to identify UI elements which belong to a group. We use F-1 score of the associated elements as the metric of comparison. For every set of elements predicted to be a group, it is considered a True Positive if the same group is present in the groundtruth, and considered a False Positive if it is not. All groundtruth groups which are not in predictions count as False Negatives and the F-1 score is computed based on these counts.

Compared to OD models the BB-CLS models have an advantage as they do not need to predict the bounding boxes for UI elements. To enable a comparison between the two approaches, we add

the unmatched GT boxes as inputs by setting all the other VH attributes to be empty values. To use this in a real-world setting, this procedure assumes the existence of a good VH without missing boxes. We study BB-CLS models here as it helps us validate the potential benefit of using VH attributes from the UI elements on the screen and consequently can motivate improving the view hierarchies for various apps and web-sites.

4.2 Model Configurations and Training Details

For the Centernet model, we used the hourglass-104 backbone (Zhou et al., 2019) with an input size of 1024×1024. For the DETR model (Carion et al., 2020), each image is proportionally resized and padded to the shape 1280×1280. We use a ResNet-50 (He et al., 2016) pretrained on ImageNet (Deng et al., 2009) as the backbone with frozen batch normalization layers for training stability. We add position embedding and object queries to each layer. The DETR models are trained on cloud TPUs with a batch size of 256 and reduce learning rate from 1×10^{-4} to 1×10^{-5} after 120k steps.

For classification and clustering models based on UIBert (Bai et al., 2021), we use an EfficientNet-B0 (Tan and Le, 2019) model for encoding the image patches and use ALBERT text encoder (Lan et al., 2019) for encoding OCR and VH attributes. We use a Transformer layer with 6 layers, 16 heads and a intermediate size of 512. We train these models on cloud TPUs with a batch size of 128 using the Adam optimizer (Kingma and Ba, 2014) with a warmup over 10k steps and reduce learning rate from 1×10^{-4} by a factor of 3 for every 50k steps . All the models were implemented using Tensorflow (Abadi et al., 2015) and converged within two days.

4.3 Results and Analysis

In this section, we report our model performance for each problem setup. For all model variants, we choose the model with the best performance on the validation set and report the numbers on the test set. Overall, we observe that for *BB-CLS* models using Image + VH + OCR perform better than models using Image + OCR and Image only. For *OD* models, we report the results from the best performing Faster R-CNN model, the best performing CenterNet model, and DETR, the overall best model for each task. We report both these results as it allows us to compare the performance of CNN based architectures with Transformer based ones. Also CenterNet models enable fast inference (Duan et al., 2019) on mobile phones compared to DETR models, making the baseline models directly usable on mobile phones. Results for all the different architectures we experimented with can be found in the Appendix A. For *BB-CLS* models, we estimate the 95% Confidence Intervals based on 5 model runs with the same configuration.

4.3.1 Icon Shape and Semantics

Among *OD* models, we observe that models based on DETR (Carion et al., 2020) which uses Transformers + Convolutions outperform CNN-based object detection models. DETR models achieve an mAP@0.5IOU of 77.94% on the test set vs 72.50% for CenterNet models on *Icon Shape* task and achieve an mAP@0.5IOU of 55.74% vs 54% for CenterNet models for *Icon Semantics* task. The performance of all of the models is weaker for *Icon Semantics* vs *Icon Shape* task. We believe this is due to *Icon Semantics* being a harder task as objects of similar shape can belong to different classes based on the rest of the screenshot or other assumptions. Additionally, since semantic labels are a sub-classification of shape labels, this dataset has fewer labeled examples per class.

For the *BB-CLS* models, we observe that models which take the VH as input outperform models without VH for both tasks. Models with Image + VH + OCR outperform models with Image + OCR by 0.63% and 2.6% for the *Shape* and *Semantics* tasks respectively. We believe this could be a result of information regarding the elements being present in VH attributes like content description. However, adding OCR does not seem to improve model performance significantly over using only the image as input. Detailed results for these tasks can be found in Tables 1, 2.

4.3.2 Label Association

For the *Label Association* task, DETR-based models also outperform CenterNet models in terms of F-1 score of 79.17% vs 75.65%. For BB-CLS models, we observe that models using VH attributes outperform models not using VH with an F-1 score of 87.23% vs 85.29%. Detailed results can be found in Table 3. We observe a significant gap in the F1-score achieved by BB-CLS models v/s OD models.

Model Type	mAP	mAP@0.5IOU
Faster R-CNN	34.60	70.24
CenterNet Hourglass	37.50	72.50
DETR	**39.28**	**77.94**

(a) Object Detection

Inputs	F-1 score	95% CI
Image + OCR + VH	**83.38**	[83.23 - 83.53]
Image + OCR	82.75	[82.54 - 82.95]
Image only	82.08	[81.71 - 82.44]

(b) Bounding box Classification

Table 1: Baseline model performance on *Icon Shape* task. For the object detection models, DETR outperforms CenterNet and Faster R-CNN architectures. For bounding box classification, models using the image OCR and view hierarchy outperform ones not using all modalities.

Model Type	mAP	mAP@0.5IOU
Faster R-CNN	25.33	53.59
CenterNet Hourglass	25.70	54.00
DETR	**26.69**	**55.74**

(a) Object Detection

Inputs	F-1 score	95% CI
Image + OCR + VH	**67.16**	[66.74 - 67.58]
Image + OCR	64.52	[63.98 - 65.05]
Image only	63.66	[63.15 - 64.16]

(b) Bounding box Classification

Table 2: Baseline model performance on *Icon Semantics* task. Similar to the For the *Icon Shape* task DETR is the best performing object detection model and models using image, OCR and view hierarchy perform the best. However, there is no statistically significant improvement between image only and image + OCR models.

Model Type	mAP	F-1 score
Faster R-CNN	36.90	75.75
CenterNet Hourglass	38.00	75.65
DETR	**40.71**	**79.17**

(a) Object Detection

Inputs	F-1 score	95% CI
Image + OCR + VH	**87.23**	[86.40 - 88.05]
Image + OCR	85.29	[84.78 - 85.79]
Image only	84.33	[83.30 - 85.36]

(b) Bounding box Classification

Table 3: Baseline model performance on *Label Association* task. The model performance characteristics are very similar to those observed for the *Icon Semantics* task.

5 Applications and Future Work

This dataset and models built on it to predict the icon and text association labels can be used to improve the label coverage for various accessibility applications like VoiceOver (Apple, 2021c), TalkBack (Accessibility, 2021e), Voice Access (Accessibility, 2022) etc. In addition, these labels can be used to improve the accessibility labels for various other platforms like web browsers and desktop applications. The models can also be used to automatically suggest accessibility labels for UI elements based on their appearance in various developer platforms (XCode, 2022; AndroidStudio, 2021) so that developers can improve the accessibility of their apps easily. Along with improving accessibility, we believe this dataset is a step towards enabling new features like voice control, screen summarization and others.

The baseline models presented in this paper can be improved in a number of ways like training multitask models for a single model to output the different labels, bridge the gap between the models which use only the image and models which use view hierarchy, improve the ability of multimodal models to handle missing view hierarchy elements and their attributes. Other directions of research include inferring labels for the long tail icon classes using the *ICON* class annotations from the *label association* task, inferring semantic labels for general UI elements. This supervised data can also be used to improve the performance of self-supervised methods for UI elements like ActionBert (He et al., 2021), UIBert (Bai et al., 2021) etc.

6 Conclusions

In this paper, we presented an enhanced version of the RICO dataset with three new sets of annotations aimed at improving the semantic understanding of mobile screens, namely *icon shape*, *icon semantics*, and *label association*. We outlined the benefits of human annotated data over automatically labeled data and released strong baseline models using image and view hierarchy for each of these tasks. Our dataset, benchmark models and experiments lay the groundwork for future research on build-

ing better models for semantic understanding of UIs. We observe that using pre-trained models and view hierarchy attributes is a promising direction for improving these models. These models can be combined with other techniques like heuristic rules to infer the multitude of labels useful for driving improvements in accessibility and automation of mobile devices.

References

Martín Abadi, Ashish Agarwal, Paul Barham, Eugene Brevdo, Zhifeng Chen, Craig Citro, Greg S. Corrado, Andy Davis, Jeffrey Dean, Matthieu Devin, Sanjay Ghemawat, Ian Goodfellow, Andrew Harp, Geoffrey Irving, Michael Isard, Yangqing Jia, Rafal Jozefowicz, Lukasz Kaiser, Manjunath Kudlur, Josh Levenberg, Dandelion Mané, Rajat Monga, Sherry Moore, Derek Murray, Chris Olah, Mike Schuster, Jonathon Shlens, Benoit Steiner, Ilya Sutskever, Kunal Talwar, Paul Tucker, Vincent Vanhoucke, Vijay Vasudevan, Fernanda Viégas, Oriol Vinyals, Pete Warden, Martin Wattenberg, Martin Wicke, Yuan Yu, and Xiaoqiang Zheng. 2015. TensorFlow: Large-scale machine learning on heterogeneous systems. Software available from tensorflow.org.

Android Accessibility. 2021a. Accessibility guidelines.

Android Accessibility. 2021b. Accessibility scanner.

Android Accessibility. 2021c. Accessibilitynodeprovider.

Android Accessibility. 2021d. Make apps more accessible.

Android Accessibility. 2021e. Talkback.

Android Accessibility. 2022. Voice access on android.

AndroidStudio. 2021. Androidstudio.

Apple. 2018. View hierarchy.

Apple. 2021a. Accessibility inspector.

Apple. 2021b. Accessibility overview.

Apple. 2021c. Voiceover.

Chongyang Bai, Xiaoxue Zang, Ying Xu, Srinivas Sunkara, Abhinav Rastogi, Jindong Chen, and Blaise Agüera y Arcas. 2021. Uibert: Learning generic multimodal representations for ui understanding. In *Proceedings of the Thirtieth International Joint Conference on Artificial Intelligence, IJCAI-21*, pages 1705–1712. International Joint Conferences on Artificial Intelligence Organization. Main Track.

Nikola Banovic, Tovi Grossman, Justin Matejka, and George Fitzmaurice. 2012. Waken: Reverse engineering usage information and interface structure from software videos. In *Proceedings of the 25th Annual ACM Symposium on User Interface Software and Technology*, UIST '12, page 83–92, New York, NY, USA. Association for Computing Machinery.

Farnaz Behrang, Steven P. Reiss, and Alessandro Orso. 2018. Guifetch: Supporting app design and development through gui search. In *Proceedings of the 5th International Conference on Mobile Software Engineering and Systems*, MOBILESoft '18, page 236–246, New York, NY, USA. Association for Computing Machinery.

Sean Bell and Kavita Bala. 2015. Learning visual similarity for product design with convolutional neural networks. *ACM Trans. Graph.*, 34(4).

Sara Bunian, Kai Li, Chaima Jemmali, Casper Harteveld, Yun Fu, and Magy Seif Seif El-Nasr. 2021. Vins: Visual search for mobile user interface design. In *Proceedings of the 2021 CHI Conference on Human Factors in Computing Systems*, CHI '21, New York, NY, USA. Association for Computing Machinery.

Nicolas Carion, Francisco Massa, Gabriel Synnaeve, Nicolas Usunier, Alexander Kirillov, and Sergey Zagoruyko. 2020. End-to-end object detection with transformers. In *European Conference on Computer Vision*, pages 213–229. Springer.

Tsung-Hsiang Chang, Tom Yeh, and Rob Miller. 2011. Associating the visual representation of user interfaces with their internal structures and metadata. In *Proceedings of the 24th Annual ACM Symposium on User Interface Software and Technology*, UIST '11, page 245–256, New York, NY, USA. Association for Computing Machinery.

Tsung-Hsiang Chang, Tom Yeh, and Robert C. Miller. 2010. *GUI Testing Using Computer Vision*, page 1535–1544. Association for Computing Machinery, New York, NY, USA.

Jieshan Chen, Chunyang Chen, Zhenchang Xing, Xiwei Xu, Liming Zhut, Guoqiang Li, and Jinshui Wang. 2020a. Unblind your apps: Predicting natural-language labels for mobile gui components by deep learning. In *2020 IEEE/ACM 42nd International Conference on Software Engineering (ICSE)*, pages 322–334.

Jieshan Chen, Mulong Xie, Zhenchang Xing, Chunyang Chen, Xiwei Xu, Liming Zhu, and Guoqiang Li. 2020b. Object detection for graphical user interface: Old fashioned or deep learning or a combination? ESEC/FSE 2020, page 1202–1214, New York, NY, USA. Association for Computing Machinery.

Biplab Deka, Zifeng Huang, Chad Franzen, Joshua Hibschman, Daniel Afergan, Yang Li, Jeffrey Nichols, and Ranjitha Kumar. 2017. Rico: A mobile app

dataset for building data-driven design applications. In *Proceedings of the 30th Annual Symposium on User Interface Software and Technology*, UIST '17.

Biplab Deka, Zifeng Huang, and Ranjitha Kumar. 2016. Erica: Interaction mining mobile apps. In *Proceedings of the 29th Annual Symposium on User Interface Software and Technology*, UIST '16, pages 767–776, New York, NY, USA. ACM.

Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. 2009. Imagenet: A large-scale hierarchical image database. In *2009 IEEE Conference on Computer Vision and Pattern Recognition*, pages 248–255.

Kaiwen Duan, Song Bai, Lingxi Xie, Honggang Qi, Qingming Huang, and Qi Tian. 2019. Centernet: Keypoint triplets for object detection. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 6569–6578.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. 2016. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778.

Zecheng He, Srinivas Sunkara, Xiaoxue Zang, Ying Xu, Lijuan Liu, Nevan Wichers, Gabriel Schubiner, Ruby Lee, and Jindong Chen. 2021. Actionbert: Leveraging user actions for semantic understanding of user interfaces. *Proceedings of the AAAI Conference on Artificial Intelligence*, 35(7):5931–5938.

Forrest Huang, John F. Canny, and Jeffrey Nichols. 2019. *Swire: Sketch-Based User Interface Retrieval*, page 1–10. Association for Computing Machinery, New York, NY, USA.

Amy Hurst, Scott E. Hudson, and Jennifer Mankoff. 2010. Automatically identifying targets users interact with during real world tasks. In *Proceedings of the 15th International Conference on Intelligent User Interfaces*, IUI '10, page 11–20, New York, NY, USA. Association for Computing Machinery.

Diederik P Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*.

Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. 2012. Imagenet classification with deep convolutional neural networks. *Advances in neural information processing systems*, 25:1097–1105.

Richard E. Ladner. 2015. Design for user empowerment. *Interactions*, 22(2):24–29.

Zhenzhong Lan, Mingda Chen, Sebastian Goodman, Kevin Gimpel, Piyush Sharma, and Radu Soricut. 2019. Albert: A lite bert for self-supervised learning of language representations. *arXiv preprint arXiv:1909.11942*.

Luis A. Leiva, Asutosh Hota, and Antti Oulasvirta. 2020. Enrico: A dataset for topic modeling of mobile ui designs. In *22nd International Conference on Human-Computer Interaction with Mobile Devices and Services*, MobileHCI '20, New York, NY, USA. Association for Computing Machinery.

Gang Li, Gilles Baechler, Manuel Tragut, and Yang Li. 2022. Learning to denoise raw mobile ui layouts for improving datasets at scale.

Toby Jia-Jun Li, Lindsay Popowski, Tom Mitchell, and Brad A Myers. 2021. *Screen2Vec: Semantic Embedding of GUI Screens and GUI Components*. Association for Computing Machinery, New York, NY, USA.

Yang Li, Jiacong He, Xin Zhou, Yuan Zhang, and Jason Baldridge. 2020a. Mapping natural language instructions to mobile UI action sequences. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 8198–8210, Online. Association for Computational Linguistics.

Yang Li, Gang Li, Luheng He, Jingjie Zheng, Hong Li, and Zhiwei Guan. 2020b. Widget captioning: Generating natural language description for mobile user interface elements. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 5495–5510, Online. Association for Computational Linguistics.

Tsung-Yi Lin, Piotr Dollár, Ross Girshick, Kaiming He, Bharath Hariharan, and Serge Belongie. 2017. Feature pyramid networks for object detection. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2117–2125.

Thomas F. Liu, Mark Craft, Jason Situ, Ersin Yumer, Radomir Mech, and Ranjitha Kumar. 2018. Learning design semantics for mobile apps. In *Proceedings of the 31st Annual ACM Symposium on User Interface Software and Technology*, UIST '18, page 569–579, New York, NY, USA. Association for Computing Machinery.

Zhe Liu. 2020. Discovering ui display issues with visual understanding. In *Proceedings of the 35th IEEE/ACM International Conference on Automated Software Engineering*, ASE '20, page 1373–1375, New York, NY, USA. Association for Computing Machinery.

Alejandro Newell, Kaiyu Yang, and Jia Deng. 2016. Stacked hourglass networks for human pose estimation. In *Computer Vision – ECCV 2016*, pages 483–499, Cham. Springer International Publishing.

Tam The Nguyen, Phong Minh Vu, Hung Viet Pham, and Tung Thanh Nguyen. 2018. Deep learning ui design patterns of mobile apps. In *Proceedings of the 40th International Conference on Software Engineering: New Ideas and Emerging Results*, ICSE-NIER '18, page 65–68, New York, NY, USA. Association for Computing Machinery.

World Health Organization. 2021. Blindness and vision impairment.

Serge Resnikoff Peter Ackland and Rupert Bourne. 2017. World blindness and visual impairment: despite many successes, the problem is growing. community eye health. *Community Eye Health*, 30(100):71–73.

Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun. 2015. Faster r-cnn: Towards real-time object detection with region proposal networks. *Advances in neural information processing systems*, 28:91–99.

Anne Spencer Ross, Xiaoyi Zhang, James Fogarty, and Jacob O. Wobbrock. 2017. Epidemiology as a framework for large-scale mobile application accessibility assessment. In *Proceedings of the 19th International ACM SIGACCESS Conference on Computers and Accessibility*, ASSETS '17, page 2–11, New York, NY, USA. Association for Computing Machinery.

Anne Spencer Ross, Xiaoyi Zhang, James Fogarty, and Jacob O. Wobbrock. 2020. An epidemiology-inspired large-scale analysis of android app accessibility. *ACM Trans. Access. Comput.*, 13(1).

Christian Szegedy, Sergey Ioffe, Vincent Vanhoucke, and Alexander A Alemi. 2017. Inception-v4, inception-resnet and the impact of residual connections on learning. In *Thirty-first AAAI conference on artificial intelligence*.

Mingxing Tan and Quoc Le. 2019. Efficientnet: Rethinking model scaling for convolutional neural networks. In *International Conference on Machine Learning*, pages 6105–6114. PMLR.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Ł ukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc.

XCode. 2022. Xcode.

Tom Yeh, Tsung-Hsiang Chang, and Robert C. Miller. 2009. Sikuli: Using gui screenshots for search and automation. In *Proceedings of the 22nd Annual ACM Symposium on User Interface Software and Technology*, UIST '09, page 183–192, New York, NY, USA. Association for Computing Machinery.

Xiaoyi Zhang, Lilian de Greef, Amanda Swearngin, Samuel White, Kyle Murray, Lisa Yu, Qi Shan, Jeffrey Nichols, Jason Wu, Chris Fleizach, Aaron Everitt, and Jeffrey P Bigham. 2021. Screen recognition: Creating accessibility metadata for mobile applications from pixels. In *Proceedings of the 2021 CHI Conference on Human Factors in Computing Systems*, CHI '21, New York, NY, USA. Association for Computing Machinery.

Xingyi Zhou, Dequan Wang, and Philipp Krähenbühl. 2019. Objects as points. *arXiv preprint arXiv:1904.07850*.

Xizhou Zhu, Weijie Su, Lewei Lu, Bin Li, Xiaogang Wang, and Jifeng Dai. 2020. Deformable detr: Deformable transformers for end-to-end object detection. *arXiv preprint arXiv:2010.04159*.

A Appendix: Data distribution and experiment results

This section contains more details on the datasets. Section A.1 contains the definitions used for attributing the detailed data distribution for each of the tasks. Table 4 contains the data distribution for the 76 *icon shape* classes. Table 5 contains the shape classes and their semantic classification along with counts for each class and table 6 contains the data distribution for the *label association* classes.

In addition Table 7 contains results for object detection models based on Faster R-CNN on the 3 tasks discussed in the paper.

A.1 Semantic class definitions

As mentioned in section 3.2 the shape icons are further sub-divided into various categories based on their functionality. The definitions of the various semantic types are given below. Each semantic icon name is prefixed by the corresponding shape name. We exclude the OTHER category for each icon shape as it is used to capture all other functionalities not covered by the mentioned semantics.

- ICON_X:CLOSE Close windows or tabs or exit a window.
- ICON_X:DELETE TEXT Delete entries, items, text, suggestions etc.
- ICON_X:MULTIPLY Mathematical operation of multiplication.
- ICON_ARROW_UPWARD:CAPITALIZE Caps Lock icon to toggle upper case and lower case letters in the keyboard.
- ICON_MAGNIFYING_GLASS:SEARCH Search in the current app or website.
- ICON_MAGNIFYING_GLASS:ZOOM IN Zoom-in to a picture, document etc.
- ICON_MAGNIFYING_GLASS:ZOOM OUT Zoom-out of a picture, document etc.
- ICON_UNDO:REPLY Reply to a message, mail etc.
- ICON_UNDO:UNDO Undo the previous action.
- ICON_UNDO:BACK Go back to the previous screen or state.
- ICON_REDO:SHARE Share this item.

- ICON_REDO:REDO Redo the previous action.
- ICON_THREE_BARS:MENU Icon to display menu options.
- ICON_PHONE:CALL Start a phone call.
- ICON_PHONE:CHAT APP Icon for a chat app.
- ICON_PHONE:PHONE APP Open the phone app.
- ICON_PHONE:END CALL End a phone or video call.
- ICON_PLAY:PLAY Playing video, audio, games, etc.
- ICON_PLAY:PLAY STORE Icon for the Google Play Store.
- ICON_PLAY:YOUTUBE Icon for the YouTube app.
- ICON_CHAT:CHAT Send a message to someone or view comments.
- ICON_CHAT:WHATSAPP Icon for the WhatsApp app.
- ICON_CHAT:FACEBOOK MESSENGER Icon for the Facebook Messenger app.
- ICON_TAKE_PHOTO:INSTAGRAM Icon for the Instagram app.
- ICON_THREE_DOTS:MORE For "more" options, contents, etc. It could also refer to menu.
- ICON_PLUS:ADD ITEM Add a new item to an existing list.
- ICON_PLUS:EXPAND Expand a UI element to show more details.

Shape Class	Count
ICON V BACKWARD	46,431
ICON NAV BAR CIRCLE	41,551
ICON NAV BAR RECT	41,449
ICON STAR	17,890
ICON THREE DOTS	15,194
ICON V FORWARD	14,131
ICON ARROW BACKWARD	13,767
ICON THREE BARS	13,659
ICON X	11,058
ICON MAGNIFYING GLASS	10,911
ICON PLUS	9,971
ICON PLAY	7,576
ICON V DOWNWARD	7,447
ICON PERSON	6,648
ICON CHECK	6,583
ICON HEART	6,274
ICON CHAT	5,483
ICON SETTINGS	4,909
ICON SHARE	4,871
ICON ARROW FORWARD	3,463
ICON LOCATION	3,398
ICON INFO	3,287
ICON HOME	3,172
ICON TIME	3,123
ICON REFRESH	2,987
ICON CLOUD	2,436
ICON EDIT	2,280
ICON QUESTION	2,263
ICON TAKE PHOTO	2,110
ICON SHOPPING CART	1,900
ICON CALENDAR	1,851
ICON NOTIFICATIONS	1,817
ICON CLOUD	2,436
ICON EDIT	2,280
ICON QUESTION	2,263
ICON TAKE PHOTO	2,110
ICON SHOPPING CART	1,900
ICON CALENDAR	1,851
ICON NOTIFICATIONS	1,817
ICON FACEBOOK	1,700
ICON ENVELOPE	1,659
ICON PEOPLE	1,658

Shape Class	Count
ICON LOCK	1,622
ICON GALLERY	1,535
ICON CALL	1,488
ICON V UPWARD	1,392
ICON VOLUME STATE	1,359
ICON LIST	1,346
ICON DOWNLOAD	1,344
ICON THUMBS UP	1,335
ICON SUN	1,327
ICON ARROW DOWNWARD	1,317
ICON LAUNCH APPS	1,136
ICON ARROW UPWARD	1,094
ICON MIC	1,016
ICON HAPPY FACE	955
ICON PAUSE	864
ICON TWITTER	860
ICON SHOPPING BAG	776
ICON MOON	719
ICON SEND	711
ICON COMPASS	691
ICON DELETE	665
ICON REDO	546
ICON VIDEOCAM	521
ICON HISTORY	447
ICON UNDO	441
ICON HEADSET	412
ICON THUMBS DOWN	382
ICON EXPAND	356
ICON GOOGLE	334
ICON UPLOAD	328
ICON SAD FACE	239
ICON STOP	204
ICON CAST	150
ICON PAPERCLIP	139
ICON VOLUME MUTE	77
ICON END CALL	65
ICON VOLUME DOWN	21
ICON CONTRACT	19
ICON VOLUME UP	14
ICON MIC MUTE	13
ICON ASSISTANT	4
TOTAL	**353,171**

Table 4: Number of instances for each icon class for the *Icon Shape* annotations.

Shape Class	Semantic Class	Count
ICON X	CLOSE	8,899
	DELETE TEXT	1,163
	MULTIPLY	50
	OTHER	840
ICON ARROW UPWARD	CAPITALIZE	154
	OTHER	915
ICON MAGNIFYING GLASS	SEARCH	10,243
	ZOOM IN	142
	ZOOM OUT	92
	OTHER	331
ICON UNDO	REPLY	113
	UNDO	109
	BACK	101
	OTHER	115
ICON REDO	SHARE	354
	REDO	63
	OTHER	117
ICON THREE BARS	MENU	11,929
	OTHER	1,329
ICON PHONE	CALL	729
	CHAT APP	530
	PHONE APP	89
	END CALL	12
	OTHER	571
ICON CHAT	CHAT APP	530
	WHATSAPP	120
	FACEBOOK MESSENGER	61
	OTHER	4,663
ICON CAMERA	INSTAGRAM	192
	OTHER	1,884
ICON PLAY	PLAY	4,452
	PLAY STORE	782
	YOUTUBE	356
	OTHER	1,846
ICON THREE DOTS	MORE	14,285
	OTHER	785
ICON PLUS	ADD ITEM	7,170
	EXPAND	396
	OTHER	2,244
TOTAL	**-**	**78,756**

Table 5: Classification of *Icon Shape* classes into semantic classes.

Class name	Count	# with text labels	% with text labels
Icon	252,342	57,716	22.87%
Text Field	16,131	3,292	20.41%
Check Box	5,958	3,723	62.49%
Radio Button	2,558	1,659	64.86%
Total	**276,989**	**66,390**	**23.96%**

Table 6: *Label Association* task statistics indicating the overall counts of the different classes and the frequency with which text labels are associated with them.

	Icon Shape		Icon Semantics		Label Association	
Backbone	mAP	mAP@0.5IOU	mAP	mAP@0.5IOU	mAP	mAP@0.5IOU
ResNet-101	32.07	65.72	**25.33**	**53.59**	34.70	73.37
Inception ResNet	31.61	70.14	25.17	53.36	36.63	75.19
ResNet-101 with FPN	**34.60**	**70.24**	25.34	53.17	**36.90**	**75.75**

Table 7: Object Detection model performance for Faster R-CNN based models with different backbone networks. The numbers in bold indicate the backbone with the best mAP@0.5IOU for the task.

End-to-end Dense Video Captioning as Sequence Generation

Wanrong Zhu[1], Bo Pang[2], Ashish V. Thapliyal[2], William Yang Wang[1], Radu Soricut[2]
[1]UC Santa Barbara, [2]Google Research
{wanrongzhu,william}@cs.ucsb.edu {bopang,asht,rsoricut}@google.com

Abstract

Dense video captioning aims to identify the events of interest in an input video, and generate descriptive captions for each event. Previous approaches usually follow a two-stage generative process, which first proposes a segment for each event, then renders a caption for each identified segment. Recent advances in large-scale sequence generation pretraining have seen great success in unifying task formulation for a great variety of tasks, but so far, more complex tasks such as dense video captioning are not able to fully utilize this powerful paradigm. In this work, we show how to model the two subtasks of dense video captioning jointly as *one* sequence generation task, and simultaneously predict the events and the corresponding descriptions. Experiments on YouCook2 and ViTT show encouraging results and indicate the feasibility of training complex tasks such as end-to-end dense video captioning integrated into large-scale pretrained models.

1 Introduction

Online videos have become an important source of knowledge and skills (O'Neil-Hart, 2017). In order to help users locate information of interest, search engines and video platforms often show anchors at "key moments", usually accompanied by descriptions of the segment's content (Baheti, 2019). This is a direct application of the dense video captioning task (Krishna et al., 2017), thus methods for improving performance on this task are relevant to any video platform.

Intuitively, dense video captioning can be decomposed into two subtasks: event localization and segment-level video captioning. Following this approach, prior work (Krishna et al., 2017; Zhou et al., 2018a; Li et al., 2018; Wang et al., 2018; Zhou et al., 2018c; Mun et al., 2019; Iashin and Rahtu, 2020) has used a two-stage, "localize-then-describe" pipeline. Such methods usually involve two separate modules with different underlying model architectures for event localization and event captioning, with captions for dense events rendered based on the predicted event spans.

Recently, with the advance of large-scale datasets and model architectures, there has been an explosion of pretrained multimodal (for text, image, video) Transformer models (Tan and Bansal, 2019; Sun et al., 2019; Li et al., 2019; Luo et al., 2020; Li et al., 2020a,b; Gan et al., 2020; Kim et al., 2021). Such models have proved to be highly effective when fine-tuned for a wide range of downstream tasks, such as visual question answering (Agrawal et al., 2015), image captioning (Chen et al., 2015), visual common sense reasoning (Zellers et al., 2019), visual entailment (Xie et al., 2019), etc. These end-tasks can be expressed as sequence generation tasks in a straightforward manner. In contrast, this is non-trivial for dense video captioning, as the segmentation subtask does not lend itself naturally to such a formulation. Does this mean more complex tasks cannot benefit from the pretraining paradigm in an end-to-end fashion? In this work, we study dense video captioning as an example of a complex task that can be cast as sequence generation and, as a result, can benefit from large-scale pretraining.

More specifically, we propose to solve the dense video captioning task as a single sequence-to-sequence modeling task using a multimodal Transformer. To this end, we design several task formulations to encode both segmentation and caption prediction in one target string. Thus, our task formulations allow the model to simultaneously predict event locations and corresponding captions in one pass, using one decoder. This opens the door to leveraging large-scale pretrained models, as well as the option of participating in large-scale multi-task training more easily by reusing existing infrastructure.

We evaluate our model on two dense video

Proceedings of the 29th International Conference on Computational Linguistics, pages 5651–5665
October 12–17, 2022.

captioning benchmarks, YouCook2 (Zhou et al., 2018a) and ViTT (Huang et al., 2020a). Our sequence generation formulations provide a feasible path forward – we obtain encouraging results compared to prior work that used a two-stage scheme with specialized architectures for each step. On the pretraining front: (a) we are able to benefit from models pretrained on very different data and tasks, such as T5 (Raffel et al., 2020), (b) pretraining on more domain-specific data (WikiHow) and pretraining tasks (predicting headings for how-to steps) lead to a similar amount of gain, but (c) having the domain-specific pretraining start from a T5 checkpoint (T5 + WikiHow) provides a significantly larger gain. The noteworthy result is that, even in the presence of large-scale domain- and task-specific pretraining (WikiHow), one can still observe measurable benefits from a task-agnostic general-purpose pretrained model (T5).

While the primary motivation for modeling the two tasks jointly is to be able to utilize the pretraining paradigm, the segmentation subtask (finding event boundaries) and the captioning subtask (describing what happens in an event) are related tasks, and intuitively stand to benefit from being modeled jointly. Our experimental results are aligned with this intuition: a model that does both segmentation and captioning simultaneously, outperforms (in terms of segmentation accuracy) a variant that focuses only on the segmentation task.

Overall, our results point to a viable alternative direction for modeling complex tasks such as end-to-end dense video captioning, in which we can leverage the large-scale pretraining paradigm to achieve modeling improvements.

2 Related Work

2.1 Multimodal Transformer

Recently, vision-and-language pre-training has attracted a lot of attention for jointly learning from visual and textual inputs in order to better solve multimodal tasks. Following the success of BERT (Devlin et al., 2019), multimodal pre-training usually adopts the Transformer (Vaswani et al., 2017) encoder structure to encode both the visual features and textual features. The late-fusion approaches first process visual and textual information separately and subsequently fuse them using another Transformer layer (Tan and Bansal, 2019; Lu et al., 2019). The early-fusion approaches jointly encode visual and texual representations (Chen et al., 2020; Sun et al., 2019; Li et al., 2019; Luo et al., 2020; Li et al., 2020a; Qi et al., 2020; Huang et al., 2020b; Li et al., 2020b; Lin et al., 2020; Gan et al., 2020; Kim et al., 2021). During pre-training, tasks such as masked language modeling, masked region modeling, and image-text matching are used to learn a cross-modal encoding which benefits downstream multimodal tasks.

2.2 Dense Video Captioning

Krishna et al. (2017) introduced the dense video captioning (DVC) task and proposed a solution based on two separate modules: one for proposing events, and another for captioning them. Recent work (Zhou et al., 2018a; Li et al., 2018; Wang et al., 2018; Zhou et al., 2018c; Mun et al., 2019; Iashin and Rahtu, 2020) follows the two-stage "detect-then-describe" framework, in which the event proposal module first predicts a set of event segments, then the captioning module constructs captions for each candidate event segment. Another line of work (Deng et al., 2021; Wang et al., 2021) removes the explicit event proposing process. Deng et al. (2021) tackles the DVC task from a top-down perspective, in which they first generate a video-level story, then ground each sentence in the story into a video segment. Wang et al. (2021) considers the DVC task as a set prediction problem, and applies two parallel prediction heads for event localization and captioning. To the best of our knowledge, our work is the first to simultaneously conduct event localization and captioning in a single run[1] within the same prediction head for the dense video captioning task.

3 Task Definition

The DVC task consists of annotating each input video into multiple segments, where each segment corresponds to an event of interest accompanied by a short description (caption). Figure 1 shows an example from the YouCook2 dataset.

Modified dense video captioning In YouCook2, each segment is marked by a start and an end time, often with gaps between segments. The burden of identifying not just the right start-time but also the right end-time increases the difficulty of the

[1]Note that on a different task (object detection), contemporaneous work (Chen et al., 2022) has combined spatial localization and object description via a sequence generation formulation by predicting bounding box coordinates and object labels in sequence.

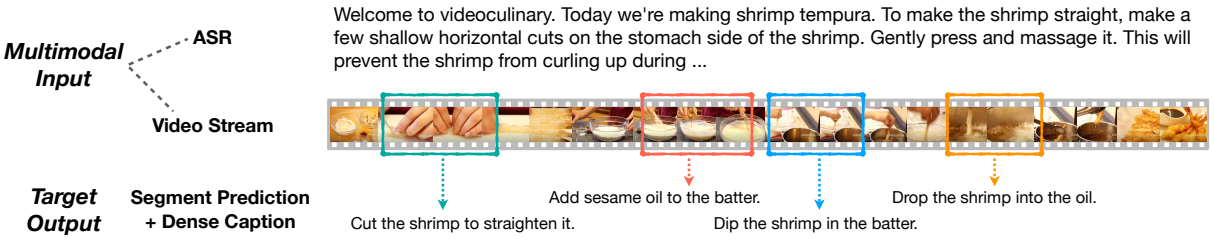

Figure 1: An example of the input video and output segmentations and captions from the YouCook2 dataset.

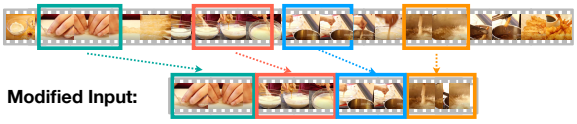

Figure 2: Modified dense video captioning: a simplified setting where the segments are concatenated to form the modified input with gaps removed. Table 1, 5, 6 show our preliminary experiments using this set up; all other results reported in the paper are carried out in the original setting as depicted in Figure 1.

segmentation task. Thus, we start our exploration with a simpler task where we introduce a variant of the YouCook2 dataset as shown in Fig. 2: all the annotated segments in a given video are concatenated to form a *modified* input, leaving out the gaps between segments. We refer to this setting as the *modified dense video captioning*: given an input from Fig. 2, the model only needs to predict n start times to fully define n segments. In this setting, the segmentation subtask becomes a *partition* task for identifying the set of start times of segments.

4 Method

As noted earlier, prior work often decomposes dense video captioning into two subtasks, (a) a segmentation subtask, and (b) a segment-level captioning subtask. These two subtasks are often addressed with different model architectures. In contrast, our approach solves both subtasks simultaneously with one single model.

We first describe how we jointly model segmentation and captioning subtasks as one single sequence generation task. To this end, we need to formulate target strings in ways that encode both segmentation and captioning predictions.

The typical input to a DVC task includes both visual information and speech in textual form – Automatic Speech Recognition (ASR) tokens. We start by introducing our target string formulations assuming only textual input, with segmentation information expressed in terms of the positions of the corresponding ASR tokens[2]. We then describe multi-modal models where the visual information is added to the input while retaining the aforementioned scheme to represent segmentation information.

4.1 Target string formulations

We describe two approaches to formulate the target strings. We refer to a model that encodes only segmentation information in the target strings as a **Seg-only model**, and one that encodes both segmentation and captioning as a **Seg+Cap model**.

Tagging-based target formulation We encode the segmentation subtask in a manner similar to the encoding of the chunking task as tagging tokens in the IOB format (Ramshaw and Marcus, 1995). Fig. 3 illustrates how we model the segmentation task with two tags (in the modified setting): the ASR token at the start of a segment receives a special token ⟨sep⟩ as the start-of-segment tag, and the rest of the tokens in the segment receive a continuation tag (we reuse the ⟨pad⟩ token). This can be extended to cover the original setting (with gaps between segments) with an additional end-of-segment tag. In this formulation, the ground-truth target output string has the exact same length as the input ASR string. To model the captioning annotation, the ⟨sep⟩ token is followed by the corresponding ground-truth caption, which is then padded till the next ⟨sep⟩ token.

While treating the segmentation task as a tagging task seems natural, the tagging-based formulation enforces equal lengths between predicted output and the input ASR tokens, which leads to potential inefficiencies: the input ASR string is usually much longer than all the descriptive captions combined, which results in many padding tokens in the target output, and leads to an unnecessary slow-down

[2]Our motivation for treating DVC as a sequence generation task is to take advantage of existing pretrained sequence generation models, currently dominated by text models; thus, we take a text-centric view in this work.

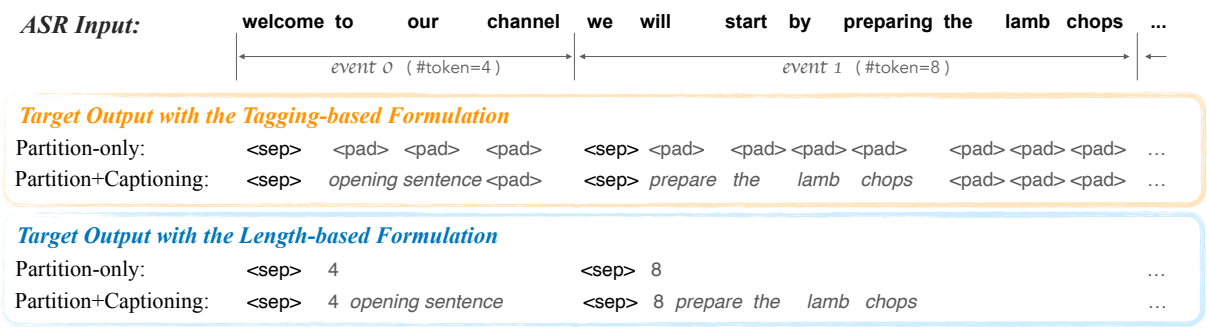

Figure 3: The tagging-based and length-based target formulations for modified dense video captioning.

in training and prediction time. Additionally, the longer-form target strings are markedly different from the usual generative pattern of the pretrained text decoder, which can reduce the effectiveness of the pretrained checkpoints. Furthermore, this formulation also assumes that captions are shorter than the ASR string for each segment; while this is mostly true, for segments where little is being explained (short ASR string), this formulation leaves insufficient capacity in the target string between the two consecutive ⟨sep⟩ tags to encode the corresponding caption, resulting in caption truncation.

Length-based target formulation To cope with the limitations of the tagging-based formulation, we predict the *length* of each segment explicitly.

Let l_i be the number of ASR tokens in the i-th segment. In the modified setting, the segmentation information for an input string with n segments is fully specified by the sequence $\{l_1, l_2, ..., l_n\}$. Fig. 3 provides an example of this length-based formulation. The ground-truth target string in a Seg-only model is simply a sequence of numbers corresponding to segment lengths (measured by the number of tokens); in a Seg+Cap model, each number is followed by the caption for that segment.

In the original setting with gaps between segments, let g_i be the offset from the last ASR token in the previous segment to the start of segment i. The target string will now aim to predict both (g_i, l_i) instead of just l_i for each segment. The sequence of all (g_i, l_i) will fully specify all segment boundaries and can be used to compute the index of the start and end ASR tokens for each segment.

This formulation has the advantage of a more efficient representation of the segmentation information, and thus a much shorter target length. The segmentation information is now explicitly expressed as numbers in the target strings, so the model needs to both figure out segmentation boundaries *and* be able to count appropriately. We explicitly want to empirically measure the ability of our models to do the latter.

4.2 Input formulation for multimodal signals

Simple Concatenation (SimpleConcat) Visual information for a given video is represented as a fixed-length sequence of pre-computed frame-level features. These features are projected to the token embedding space via a fully connected layer. We simply concatenate the sequence of ASR token embeddings and the sequence of projected visual features to form the multimodal input to the encoder. There's one potential caveat: while the visual features are extracted at a fixed frame rate, the ASR tokens are often *not* spoken at a fixed speed; thus positions in this multimodal input sequence do not provide straightforward information on which visual frames are temporally aligned with a certain ASR span. Since segmentation prediction is expressed relative to the ASR-token position index, it is not clear whether the model is able to take full advantage of visual information, absent how these two modalities align temporally.

Prior work on multimodal pretraining has found visual-textual information alignment to be a reasonably solvable task. Huang et al. (2020a) reported 87% accuracy for aligning video segments and ASR spans in HowTo100M (Miech et al., 2019), so it is possible that the decoder can learn to attend to appropriate visual information while "counting" the ASR tokens.

Temporal embedding (Emb$_{\text{TIME}}$) We can also express the temporal alignment more explicitly in the input by adding temporal embeddings to both ASR tokens and visual frames. In this formulation, we learn a temporal embedder shared between the text modality and the visual modality, which maps timestamps to temporal embeddings. Embeddings computed from token timestamps are then added to ASR token embeddings, and embeddings computed

from frame timestamps are added to projected visual frame features. This way, ASR tokens and frames that are temporally close to each other receive similar temporal embeddings, making their representations closer to each other.

For more explorations on explicitly expressing temporal alignment in the input, see Appendix A.1 for an additional method to insert explicit timestamp markers into the input text sequence.

5 Experiments

5.1 Datasets

Dense Video Captioning Datasets We use two publicly available datasets to verify the effectiveness of our model formulations: YouCook2 (Zhou et al., 2018a) and ViTT (Huang et al., 2020a).[3] The YouCook2 dataset is restricted to videos retrieved from YouTube from the cooking domain, targeting 89 recipes; each event segment is manually marked with a start and end time, along with a human-generated caption for each tightly-bounded segment. The ViTT dataset contains instructional videos from YouTube-8M (Abu-El-Haija et al., 2016) and covers a broader range of topics. Its segment annotation focuses on event start time and rater-provided captions for the corresponding segment (spanning two consecutive start-time annotations). Both datasets are annotated with captions written in English.

Note that while the YouCook2 data release contains training, dev, and test sets, its test set does not come with human annotations. Thus, we split the original validation set into validation and test splits for our experiments. For ViTT, we use the original train/val/test splits provided with the data. The number of videos available for use at the time of our work[4] for Youcook2 is 925 for train, 206 for validation and 105 for test. For ViTT there are 4736 train, 932 validation and 932 test videos.

Domain-specific pretraining with WikiHow In addition to general-purpose pre-trained models like T5, we also experiment with domain-specific pretraining. To this end, we use the WikiHow dataset (Koupaee and Wang, 2018). WikiHow consists of instructional (how-to) articles, which makes it *in-domain data* for the two dense video captioning datasets considered here, while being much larger in size[5]. In addition, WikiHow articles contain detailed step-by-step instructions. Each step comes with a summary, which usually serves as the section title. Both the step boundaries and summaries are easily extracted according to the page meta-data. This provides the ground-truth annotation for a "dense document caption" task: given the full article as a sequence of text tokens, predict the step boundaries and summaries. This enables us to also include a *domain-specific pretraining task* that closely resembles our task. For each formulation described in Sec. 4, we experiment with a checkpoint pre-trained on the WikiHow data using the corresponding target string formulation.

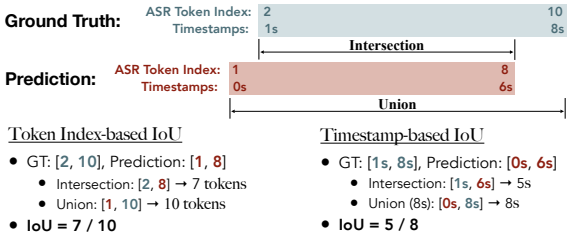

Figure 4: Comparisons between the token index-based and timestamp-based IoU used in our study.

5.2 Evaluation Metrics

Segmentation Performance Following previous works (Zhou et al., 2018b; Shi et al., 2019), we use the mean Intersection-over-Union (mIoU) metric to evaluate the segmentation performance. Recall that the ground-truth segments are marked by start (and end) times, whereas the predicted segments are expressed according to the position of the corresponding ASR token. For the modified dense video captioning task, we compute the token index-based IoU: each ground-truth segment is defined by the start and end ASR token index, and will be compared against the predicted index. For the original task, we compute the timestamp-based IoU: predicted indices are mapped into the corresponding ASR token timestamps and compared against the segment's ground-truth start and end timestamps. Fig. 4 provides an example of the two types of IoU used in this study.

An IoU score can be computed for each (ground-truth, predicted) segment pair. The mIoU measure provides a summary score for segmentation performance over the entire video. For each ground-truth segment, we take its maximal IoU to predicted

[3] YouCook2 released under an MIT license; ViTT released under an "AS IS" license.

[4] As of 2021; note that YouTube videos are subject to user deletion.

[5] WikiHow has 157,116 articles in its training set, and 5,593 articles in its validation set.

*	Target Formulation	Checkpoint	Seg-only model				Seg+Cap model					
			mIoU	Precision	Recall	F1	mIoU	F1	B@4	METEOR	CIDEr	ROUGE-L
0	Random Partition		37.74	26.13	24.88	23.52	-	-	-	-	-	-
1	Tagging-based	-	33.59	23.04	29.37	24.46	19.72	17.09	0.07	0.91	0.03	2.07
2		T5	12.06	1.78	7.46	2.81	6.73	0.24	0.00	0.01	0.00	0.03
3	Length-based	-	36.30	26.23	28.79	25.81	33.62	24.69	0.24	1.62	0.04	4.03
4		T5	42.71	31.85	33.04	31.21	42.82	32.16	1.83	4.17	0.21	8.74

Table 1: Preliminary experiments comparing the tagging-based and the length-based formulation on YouCook2 modified dense video captioning. We report the evaluation results on the validation set (one run per setting) with models initialized from random weights or from T5 checkpoints.

segments as the IoU score for this ground-truth segment, and mIoU is the average of this value across all ground-truth segments. The individual mIoU for each video is then averaged across the test data and reported as the overall mIoU.

For diagnostic purposes, we also compute: 1) the percentage of predicted segments which have an IoU score with at least one ground-truth segment above a certain threshold t (precision@t); 2) the percentage of ground-truth segments which have an IoU score with at least one predicted segment, above a certain threshold t (recall@t), as well as their geometric mean as F1. Following prior work, we compute these scores for a set of IoU thresholds t={0.3, 0.5, 0.7, 0.9}, and report the average over these thresholds.

Captioning Performance We compute BLEU-4 (Papineni et al., 2002), METEOR (Banerjee and Lavie, 2005), CIDEr (Vedantam et al., 2015), and ROUGE (Lin, 2004) scores between generated captions and the ground truth when the predicted and ground-truth segment "match" (i.e., with IoU score above a given threshold t); if a ground truth segment does not have a matching prediction, it contributes a zero to the average score for the corresponding threshold. Again, we compute this for a set of IoU thresholds of {0.3, 0.5, 0.7, 0.9}, and report the average over these thresholds.

5.3 Implementation Details

Models were trained on 4x4 TPUs, and we used about 180k GPU hours for around 1380 training runs, including pretraining the WikiHow checkpoint, pilot studies with toy examples, debugging, and hyperparameter tuning. The models have approximately 70 million parameters. We used the Adafactor (Shazeer and Stern, 2018) optimizer and a learning rate schedule of 1000 warmup steps followed by square-root decay. We did a few initial exploratory runs over base learning rates of {0.001, 0.01, 0.05, 0.1, 0.2, 0.5, 0.8, 1, 2, 5} to determine that a base learning rate of 1 worked well and used it for all the experiments reported.

For our visual representations, we computed 3D CNN features pre-trained on the Kinetics (Carreira and Zisserman, 2017; Kay et al., 2017) dataset for frames sampled at 30 fps, resulting in one feature for each 1 second clip.

5.4 Experiments in the modified setting

Experimental setup We conduct comparisons of the two different target formulations, tagging-based and length-based, in the modified setting, using the following experimental setup: (a) max input text length and target length are set to 1024, and max input visual feature length is set to 800; this can truncate longer ASR sequences, but allow us to quickly iterate through different settings with fewer computational resources; (b) only one run for each setting. We report results on the validation set in Table 1.

Target formulations The best performing model (row #4 in Table 1: length-based with T5 checkpoint) outperforms a random partition baseline[6] (row #0 in Table 1), which indicates our target formulation approach to the segmentation task is capturing some segmentation information effectively.

When trained from scratch, the length-based formulation achieves higher performance across the board (#3 vs #1), with a smaller gap for the Seg-only model, and a more marked lead for the Seg+Cap model. We hypothesize that while treating the segmentation task as a tagging task is more or less feasible on its own, combining segmentation tags and captions is not a suitable formulation for

[6]For the random partition baseline, a video is randomly partitioned into n segments, with n sampled uniformly from 1 to 15 (The mean number of segments in the ground-truth is 8).

Dataset	*	Input Formulation	Checkpoint?	Seg-only model	Seg+Cap model				
				mIoU	mIoU	B@4	METEOR	CIDEr	ROUGE-L
Youcook2	0	Random Segmentation		20.61 ± 1.04	-	-	-	-	-
	1	SimpleConcat	-	12.99 ± 1.55	16.45 ± 8.72	0.17 ± 0.11	0.66 ± 0.04	0.02 ± 0.01	1.99 ± 0.20
	2		T5	24.14 ± 1.07	24.21 ± 1.64	0.88 ± 0.04	1.50 ± 0.12	0.09 ± 0.01	3.34 ± 0.27
	3		WikiHow	22.58 ± 1.09	23.33 ± 0.79	0.67 ± 0.15	1.47 ± 0.05	0.08 ± 0.01	3.51 ± 0.13
	4		WikiHow T5	**27.77** ± 0.09	**30.26** ± 1.24	**2.96** ± 0.28	**3.49** ± 0.30	**0.25** ± 0.03	**7.00** ± 0.42
	5	+ Emb_{TIME}	-	18.51 ± 1.95	18.71 ± 0.17	0.12 ± 0.07	0.48 ± 0.08	0.02 ± 0.01	1.41 ± 0.22
	6		T5	23.02 ± 1.05	23.96 ± 0.08	1.32 ± 0.08	1.91 ± 0.07	0.11 ± 0.01	4.20 ± 0.13
	7		WikiHow	21.68 ± 1.93	21.88 ± 0.86	0.69 ± 0.19	1.30 ± 0.07	0.07 ± 0.01	3.06 ± 0.13
	8		WikiHow T5	26.51 ± 0.45	28.70 ± 0.92	2.58 ± 0.19	3.23 ± 0.10	0.22 ± 0.01	6.45 ± 0.17
ViTT	9	Random Segmentation		21.90 ± 0.15	-	-	-	-	-
	10	SimpleConcat	-	33.85 ± 0.70	32.69 ± 0.71	0.11 ± 0.01	3.76 ± 0.35	0.08 ± 0.01	3.86 ± 0.28
	11		T5	37.89 ± 0.10	38.07 ± 0.65	0.57 ± 0.03	5.92 ± 0.37	0.16 ± 0.02	6.59 ± 0.69
	12		WikiHow	38.20 ± 0.27	37.80 ± 0.62	0.40 ± 0.07	5.48 ± 0.18	0.14 ± 0.01	6.02 ± 0.34
	13		WikiHow T5	**41.87** ± 0.26	42.40 ± 0.30	**1.29** ± 0.07	**8.10** ± 0.34	**0.25** ± 0.01	**9.26** ± 0.39
	14	+ Emb_{TIME}	-	33.89 ± 0.21	35.37 ± 3.18	0.04 ± 0.03	3.42 ± 0.61	0.07 ± 0.01	3.28 ± 0.83
	15		T5	37.78 ± 0.15	38.50 ± 0.55	0.75 ± 0.10	6.37 ± 0.39	0.18 ± 0.01	7.19 ± 0.48
	16		WikiHow	37.27 ± 0.08	36.97 ± 0.48	0.38 ± 0.06	5.31 ± 0.06	0.13 ± 0.01	5.82 ± 0.23
	17		WikiHow T5	**41.64** ± 0.12	**43.22** ± 0.72	**1.22** ± 0.08	**8.05** ± 0.20	**0.25** ± 0.01	**9.18** ± 0.45

Table 2: Dense video captioning performance on YouCook2 and ViTT test sets with the length-based formulation. We ran 3 trials for each setting, and report the evaluation results (mean ± std) with models initialized from random weights, T5 checkpoints, WikiHow checkpoints, and T5 checkpoints further pretrained on WikiHow. Note: Seg stands for the segmentation task, and Cap stands for the captioning task.

the combined task – to the point that the Seg+Cap model underperforms Seg-only model in segmentation metrics (mIoU of 19.72 vs. 33.59 in #1).

The length-based formulation overall benefits from the T5 checkpoint (#3 vs #4 in Table 1) across different sub-tasks. Note that for the Seg-only model, the target strings (sequences of numbers) are not typically seen in T5 pretraining, but the T5 checkpoint still boosts its performance. In contrast, the tagging-based formulation is not able to benefit from the T5 checkpoint in our experiments. One possible explanation is that the target strings in tagging-based (with large chunks of padding tokens) are just too different from the T5 pretraining targets.

Given the results obtained in the modified setting, we focus our efforts on using length-based target formulation in the more challenging original setting in the following section.

Ablation studies We conducted ablation studies on input modalities, and find models that take text-only inputs stand to benefit more from the pretrained checkpoints than the models that only take visual inputs. We also conducted ablation studies on partial parameter initialization, and found that partially loading checkpoints from pre-trained models does not work as well as fully loading checkpoints for both the encoder and the decoder. See Appendix (A.2) for more details.

5.5 Experiments in the original setting

Experimental setup Using the length-based target formulation, we conduct a more extensive comparison of the effect of different pretraining strategies, as well as different input formulations on the original dense video captioning task on both YouCook2 and ViTT. Maximum sequence lengths are set to ensure no truncation happens in either dataset – input text: 4096; visual feature: 800 (YouCook2) / 500 (ViTT); target: 512 (YouCook2) / 256 (ViTT). We ran each experiment with different seeds three times to account for performance variance from random initializations. We report the mean and standard deviation (using 3 runs) for each metric in Table 2. We choose the best checkpoint based on performance on the validation set and report the performance on the test set.

Effects of Pretraining For both datasets, there are significant performance improvements from utilizing pre-trained checkpoints in terms of both segmentation metrics and captioning metrics. Interestingly, training from the WikiHow checkpoint (using in-domain task over in-domain data) provides similar performance improvement to T5 alone (see, for instance, #2 vs #3, or #11 vs #12 in Table 2). However, starting from the generic-language T5 checkpoint and adding in-domain WikiHow pretraining (WikiHow T5, e.g., #4 and #13) boosts all metrics by a large and significant margin.

Model	mIoU	Prec.	Rec.	B@4	M
vsLSTM (Zhang et al., 2016)	32.2	24.1	22.1	-	-
SCNN-prop (Shou et al., 2016)	26.7	23.2	28.2	-	-
ProcNet (Zhou et al., 2018b)	37.0	30.4	37.1	-	-
Bi-LSTM + TempoAttn (Zhou et al., 2018c)	-	-	-	0.08	4.62
End2end Transformer (Zhou et al., 2018c)	-	-	-	0.30	6.58
Context-aware Fusion (Shi et al., 2019)	41.4	-	-	2.61	17.43
End2end Sequence Generation (Ours)	30.3	24.5	24.2	2.96	3.49

Table 3: Dense video captioning performance on YouCook2 in the context of prior work. Following prior work, the segmentation performance is measured by the mIoU, the precision (Prec.) and recall (Rec.) at IoU threshold t=0.5. Captioning performance is measured by the average BLEU-4 (B@4) and METEOR (M) at IoU thresholds $t \in \{0.3, 0.5, 0.7, 0.9\}$.

Effects of Joint Modeling If we compare the mIoU score achieved by the Seg+Cap model to the mIoU score achieved by the Seg-only model in Table 2, across different settings, we observe a general trend where the Seg+Cap model outperforms the Seg-only model on this segmentation metric. This indicates that with the right formulation, the segmentation subtask (predicting event boundary) can indeed benefit from joint learning with a related captioning subtask (summarizing event content).

Input formulations Results using SimpleConcat compared to their counterparts using Emb_{TIME} in Table 2, are mixed. While Emb_{TIME} seems to bring non-trivial improvement to models trained from scratch, the training from scratch settings also has the largest variance in our experiments[7]. That said, the Seg+Cap model did achieve its best mIoU score on ViTT using Emb_{TIME}. More work is needed to fully understand the potential of Emb_{TIME}.

Comparison against prior work for YouCook2[8] Table 3 provides a summary of dense video captioning performance on YouCook2 reported in prior work. Some of the prior work (Zhang et al., 2016; Shou et al., 2016; Zhou et al., 2018b) focused only on the segmentation subtask, while some (Zhou et al., 2018c; Shi et al., 2019) approached the end-to-end task as a two-stage task and solved the two subtasks separately. In this context, we find the results from our simple end-to-end sequence generation based approach quite encouraging, and hope this inspires future studies to fully realize the potential of this alternative approach.

Qualitative Analysis We provide a few example model outputs from our Seg+Cap model. More examples can be found in the Appendix A.4.

Figure 5 presents example segmentation results. As reflected in Figure 5(a), a segmentation prediction that is largely correct for a few segments, but is missing out on some ground-truth segments and contains over-segmentation of others can result in a relatively low mIoU score. For examples with relatively high mIoU scores, see Figure 5(b) (taken from the more challenging YouCook2 setting, with gaps between ground-truth segments), and Figure 5(c) (taken from ViTT, with no gaps between segments).

Next, we observe that caption quality is good when the segment boundary prediction is highly accurate: if we restrict to segments with IoU $\geq 90\%$ between the prediction and the target, the average ROUGE-L score for corresponding captions is 30.18 for YouCook2 and 44.33 for ViTT. Table 4 presents qualitative examples.

6 Conclusion

In this paper, we describe different task formulations for solving the dense video captioning task using an end-to-end sequence generation approach, which allows us to leverage pre-trained text-only encoder-decoder models. We conduct experiments on YouCook2 and ViTT in several pretraining settings. Experimental results show that general (T5) and in-domain (WikiHow) text-only pre-trained models both improve video partitioning and segmentation performance, and the gains are cumulative. Also, the segmentation subtask benefits from joint modeling with the captioning subtask. We hope our work can inspire future studies on leveraging pre-trained models, large-scale text corpora and language generation formulations to solve multimodal tasks such as dense video captioning.

Ethical Statement

Our experiments are conducted only on videos with available English ASR annotations, as we inherit this limitation from the available data for this task. We use existing datasets based on public YouTube videos. As a consequence, any videos that are

[7] To the extent that the Seg+Cap model performance in #1 can be considered an outlier: its mIoU scores for the three runs are (11, 11, 26), which resulted in a large std value not seen anywhere else in the table. We looked into these three runs in more details, and our best guess was that one of them incidentally got an advantageous random initialization.

[8] ViTT is a relatively newer dataset and past work has only reported performance of the segment-level captioning subtask using ground-truth segments; we are not aware of existing work reporting end-to-end dense video captioning performance.

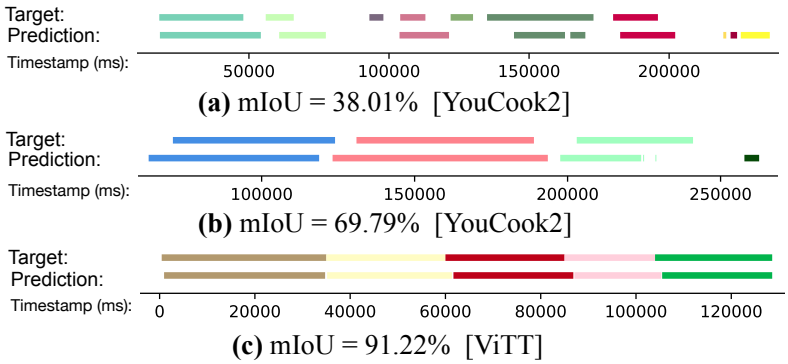

Figure 5: Example segmentation predictions corresponding to different mIoU scores.

	IoU	Segment Border (ms)	Caption
Tgt.	90.0%	[58000.0, 77000.0]	whisk eggs and season with salt
Pred.		[57309.0, 78429.5]	whisk the eggs in the deep plate
Tgt.	99.3%	[28000.0, 45000.0]	chop up the garlic in the food processer
Pred.		[28005.0, 44894.0]	chop garlic and place in the food processor
Tgt.	94.7%	[64199.0, 98080.0]	Preparining remaining ingredients
Pred.		[65710.0, 98380.0]	Chopping the remaining ingredients
Tgt.	97.0%	[65100.0, 124729.0]	Blow-drying the roots
Pred.		[63239.5, 124714.5]	Blow-drying hair

Table 4: Example caption predictions where the IoU $\geq 90\%$ between the target (Tgt.) and the predicted (Pred.) segments. The first two examples are from YouCook2, the last two examples are from ViTT.

no longer publicly available on YouTube (e.g., removed by the user) at the time of the study needed to be excluded from our experimental setup.

References

Sami Abu-El-Haija, Nisarg Kothari, Joonseok Lee, Apostol Natsev, George Toderici, Balakrishnan Varadarajan, and Sudheendra Vijayanarasimhan. 2016. Youtube-8m: A large-scale video classification benchmark. *ArXiv*, abs/1609.08675.

Aishwarya Agrawal, Jiasen Lu, Stanislaw Antol, Margaret Mitchell, C. Lawrence Zitnick, Devi Parikh, and Dhruv Batra. 2015. Vqa: Visual question answering. *International Journal of Computer Vision*, 123:4–31.

Prashant Baheti. 2019. Search helps you find key moments in videos.

Satanjeev Banerjee and Alon Lavie. 2005. Meteor: An automatic metric for mt evaluation with improved correlation with human judgments. In *IEEvaluation@ACL*.

João Carreira and Andrew Zisserman. 2017. Quo vadis, action recognition? a new model and the kinetics dataset. *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 4724–4733.

Ting Chen, Saurabh Saxena, Lala Li, David J. Fleet, and Geoffrey Hinton. 2022. Pix2seq: A language modeling framework for object detection. In *International Conference on Learning Representations*.

Xinlei Chen, Hao Fang, Tsung-Yi Lin, Ramakrishna Vedantam, Saurabh Gupta, Piotr Dollár, and C. Lawrence Zitnick. 2015. Microsoft COCO Captions: Data collection and evaluation server. *arXiv preprint arXiv:1504.00325*.

Yen-Chun Chen, Linjie Li, Licheng Yu, Ahmed El Kholy, Faisal Ahmed, Zhe Gan, Yu Cheng, and Jingjing Liu. 2020. Uniter: Universal image-text representation learning. In *ECCV*.

Chaorui Deng, Shizhe Chen, Da Chen, Yuan He, and Qi Wu. 2021. Sketch, ground, and refine: Top-down dense video captioning. *2021 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 234–243.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. Bert: Pre-training of deep bidirectional transformers for language understanding. In *NAACL*.

Zhe Gan, Yen-Chun Chen, Linjie Li, Chen Zhu, Yu Cheng, and Jingjing Liu. 2020. Large-scale adversarial training for vision-and-language representation learning. *ArXiv*, abs/2006.06195.

Gabriel Huang, Bo Pang, Zhenhai Zhu, Clara Rivera,

and Radu Soricut. 2020a. Multimodal pretraining for dense video captioning. In *AACL*.

Zhicheng Huang, Zhaoyang Zeng, Bei Liu, Dongmei Fu, and Jianlong Fu. 2020b. Pixel-bert: Aligning image pixels with text by deep multi-modal transformers. *ArXiv*, abs/2004.00849.

Vladimir Iashin and Esa Rahtu. 2020. Multi-modal dense video captioning. *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, pages 4117–4126.

Will Kay, João Carreira, Karen Simonyan, Brian Zhang, Chloe Hillier, Sudheendra Vijayanarasimhan, Fabio Viola, Tim Green, Trevor Back, Apostol Natsev, Mustafa Suleyman, and Andrew Zisserman. 2017. The kinetics human action video dataset. *ArXiv*, abs/1705.06950.

Wonjae Kim, Bokyung Son, and Ildoo Kim. 2021. Vilt: Vision-and-language transformer without convolution or region supervision. In *ICML*.

Mahnaz Koupaee and William Yang Wang. 2018. Wikihow: A large scale text summarization dataset. *ArXiv*, abs/1810.09305.

Ranjay Krishna, Kenji Hata, Frederic Ren, Li Fei-Fei, and Juan Carlos Niebles. 2017. Dense-captioning events in videos. *2017 IEEE International Conference on Computer Vision (ICCV)*, pages 706–715.

Gen Li, Nan Duan, Yuejian Fang, Daxin Jiang, and Ming Zhou. 2020a. Unicoder-vl: A universal encoder for vision and language by cross-modal pre-training. In *AAAI*.

Liunian Harold Li, Mark Yatskar, Da Yin, Cho-Jui Hsieh, and Kai-Wei Chang. 2019. Visualbert: A simple and performant baseline for vision and language. *ArXiv*, abs/1908.03557.

Xiujun Li, Xi Yin, Chunyuan Li, Xiaowei Hu, Pengchuan Zhang, Lei Zhang, Lijuan Wang, Houdong Hu, Li Dong, Furu Wei, Yejin Choi, and Jianfeng Gao. 2020b. Oscar: Object-semantics aligned pre-training for vision-language tasks. In *ECCV*.

Yehao Li, Ting Yao, Yingwei Pan, Hongyang Chao, and Tao Mei. 2018. Jointly localizing and describing events for dense video captioning. *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 7492–7500.

Chin-Yew Lin. 2004. Rouge: A package for automatic evaluation of summaries. In *ACL 2004*.

Junyang Lin, An Yang, Yichang Zhang, Jianbin Liu, Jingren Zhou, and Hongxia Yang. 2020. Interbert: Vision-and-language interaction for multi-modal pretraining. *ArXiv*, abs/2003.13198.

Jiasen Lu, Dhruv Batra, Devi Parikh, and Stefan Lee. 2019. Vilbert: Pretraining task-agnostic visiolinguistic representations for vision-and-language tasks. In *NeurIPS*.

Huaishao Luo, Lei Ji, Botian Shi, Haoyang Huang, Nan Duan, Tianrui Li, Xilin Chen, and Ming Zhou. 2020. Univilm: A unified video and language pre-training model for multimodal understanding and generation. *ArXiv*, abs/2002.06353.

Antoine Miech, Dimitri Zhukov, Jean-Baptiste Alayrac, Makarand Tapaswi, Ivan Laptev, and Josef Sivic. 2019. Howto100m: Learning a text-video embedding by watching hundred million narrated video clips. *ArXiv*, abs/1906.03327.

Jonghwan Mun, L. Yang, Zhou Ren, N. Xu, and Bohyung Han. 2019. Streamlined dense video captioning. *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 6581–6590.

Celie O'Neil-Hart. 2017. Self-directed learning from youtube - think with google.

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. Bleu: a method for automatic evaluation of machine translation. In *ACL*.

Di Qi, Lin Su, Jianwei Song, Edward Cui, Taroon Bharti, and Arun Sacheti. 2020. Imagebert: Cross-modal pre-training with large-scale weak-supervised image-text data. *ArXiv*, abs/2001.07966.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J. Liu. 2020. Exploring the limits of transfer learning with a unified text-to-text transformer. *JMLR*.

Lance Ramshaw and Mitch Marcus. 1995. Text chunking using transformation-based learning. In *Third Workshop on Very Large Corpora*.

Noam Shazeer and Mitchell Stern. 2018. Adafactor: Adaptive learning rates with sublinear memory cost. In *Proceedings of the 35th International Conference on Machine Learning, ICML 2018, Stockholmsmässan, Stockholm, Sweden, July 10-15, 2018*, volume 80 of *Proceedings of Machine Learning Research*, pages 4603–4611. PMLR.

Botian Shi, Lei Ji, Yaobo Liang, Nan Duan, Peng Chen, Zhendong Niu, and M. Zhou. 2019. Dense procedure captioning in narrated instructional videos. In *ACL*.

Zheng Shou, Dongang Wang, and Shih-Fu Chang. 2016. Temporal action localization in untrimmed videos via multi-stage cnns. *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1049–1058.

Chen Sun, Austin Myers, Carl Vondrick, Kevin P. Murphy, and Cordelia Schmid. 2019. Videobert: A joint

model for video and language representation learning. *2019 IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 7463–7472.

Hao Hao Tan and Mohit Bansal. 2019. Lxmert: Learning cross-modality encoder representations from transformers. In *EMNLP*.

Ashish Vaswani, Noam M. Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. *ArXiv*, abs/1706.03762.

Ramakrishna Vedantam, C. Lawrence Zitnick, and Devi Parikh. 2015. Cider: Consensus-based image description evaluation. *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 4566–4575.

Jingwen Wang, Wenhao Jiang, Lin Ma, W. Liu, and Yong Xu. 2018. Bidirectional attentive fusion with context gating for dense video captioning. *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 7190–7198.

Teng Wang, Ruimao Zhang, Zhichao Lu, Feng Zheng, Ran Cheng, and Ping Luo. 2021. End-to-end dense video captioning with parallel decoding. *ArXiv*, abs/2108.07781.

Ning Xie, Farley Lai, Derek Doran, and Asim Kadav. 2019. Visual entailment: A novel task for fine-grained image understanding. *ArXiv*, abs/1901.06706.

Rowan Zellers, Yonatan Bisk, Ali Farhadi, and Yejin Choi. 2019. From recognition to cognition: Visual commonsense reasoning. *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 6713–6724.

Ke Zhang, Wei-Lun Chao, Fei Sha, and Kristen Grauman. 2016. Video summarization with long short-term memory. In *ECCV*.

Luowei Zhou, Chenliang Xu, and Jason J Corso. 2018a. Towards automatic learning of procedures from web instructional videos. In *AAAI Conference on Artificial Intelligence*, pages 7590–7598.

Luowei Zhou, Chenliang Xu, and Jason J. Corso. 2018b. Towards automatic learning of procedures from web instructional videos. In *AAAI*.

Luowei Zhou, Yingbo Zhou, Jason J. Corso, Richard Socher, and Caiming Xiong. 2018c. End-to-end dense video captioning with masked transformer. *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 8739–8748.

A Appendix

A.1 Timestamp markers (T-marker)

Here we describe an alternative way to encode temporal alignment between textual and visual input. Since the frames are extracted at a fixed rate, we can explicitly add time markers to the text input to "mark" out tokens spoken at the corresponding time points. The video features are extracted with a frame rate of 1 frame per second in our work. We insert a time marker for each frame after the last ASR token spoke before the corresponding timestamp. A time marker consists of a special anchor token, followed by the timestamp token (an integer corresponding to the timestamp in seconds).

Performance using this input formuation can be found in the T-marker rows in Table 7. For models trained from scratch, including the timestamp markers can positively impact model performance, indicating that these markers do indeed provide helpful information. However, adding these markers only hurt the performance of any model trained from an existing checkpoint. We hypothesize that this is because the text sequence with frequent markers is too different from the pre-trained datasets, leaving the pre-trained checkpoints less effective for models using this input formulation.

A.2 Ablation studies

Ablation on Input Data Table 5 shows comparisons of different input sources on YouCook2 dense video captioning task. For all three settings (text-only, video-only, text+video), pre-training on WikiHow has the best performance on both subtasks, and using the T5 checkpoint has better performance than training from scratch. With the pre-trained WikiHow checkpoint, the "Text-only" setting has comparable performance as the "Text+Video" setting that takes both the ASR transcript and the video features as input. Using the video features alone results in worse performance, indicating the high value of text transcripts to the captioning task.

Ablation on T5 Checkpoint Table 6 compares performances when using different pre-trained checkpoints on YouCook2 modified dense video captioning. Using either the T5 or the WikiHow T5 checkpoints outperforms the model initialized from random weights, which verifies the effectiveness of pre-training. Since the targets in the end task are markedly different from, say, T5 pretraining targets, we also experimented with loading partial checkpoints (e.g., only encoder weights). Interestingly, using the full checkpoint has better performance than loading only encoder or only decoder weights.

A.3 Comprehensive experimental results

Table 7 provides a more comprehensive summary of our experimental results in the original setting. It is the same experimental setting as Table 2, but we also report additional performance metrics for the segmentation tasks, as well as performance for the T-marker input formulation. Table 8 is again under the same experimental setting, but reports median instead of (mean, std) to summarize the 3 repeats for each setting, so that the metrics are less affected by occasional outliers.

A.4 Qualitative examples

Here we provide more examples for the segmentation subtask and the captioning subtask.

Figure 6 and Figure 7 illustrate several sets of segmentation results predicted by our Seg+Cap model on YouCook2 and ViTT. We can see that the predicted segmentation predictions on ViTT (Figure 7) are relatively more aligned with the ground-truth. This is because ViTT has a comparably simpler formulation with no gaps between video segments. In the more challenging setup on YouCook2 (Figure 6) where the model needs to predict both the start and end point for each segment, the listed examples show that when predictions are mostly correct for a few segments, the IoU scores can be relatively low. Common types of segmentation misalignment include:

- "over-segmentation": the prediction splits a ground-truth span into several sub-chunks (Figure 6 (a)(b)(c));
- "under-segmentation": one predicted segment covers several ground-truth events (Figure 6 (c)(f));
- "prediction-not-covered": the predicted event is not labeled by the ground-truth annotation (Figure 6 (b)(d)(e)).

Table 9 shows examples of the jointly predicted segments and corresponding captions. We have similar findings as in the main paper that when the segment boundary prediction is well aligned with the ground truth, the corresponding captions are often high-quality as well.

Input	Checkpoint	Segmentation			Segmentation + Captioning				
		Precision	Recall	F1	F1	B@4	METEOR	CIDEr	ROUGE-L
Text-only	-	31.86	30.66	31.25	30.39	0.55	1.88	0.07	5.23
	T5	36.22	37.06	36.64	37.89	3.36	4.76	0.28	10.61
	WikiHow T5	**71.13**	**63.77**	**67.25**	**58.71**	9.57	**11.99**	0.85	23.21
Video-only	-	28.02	19.48	22.98	27.5	0.52	1.89	0.07	4.82
	T5	27.43	27.25	27.34	27.86	0.40	1.65	0.05	4.11
	WikiHow T5	25.45	24.93	25.19	23.19	0.42	1.48	0.05	3.84
Text + Video	-	32.53	30.90	31.69	29.09	0.34	1.68	0.06	4.78
	T5	36.96	37.99	37.47	32.58	2.99	4.22	0.26	9.20
	WikiHow T5	71.07	62.76	66.66	57.84	**9.87**	11.96	**0.86**	**23.25**

Table 5: Ablation on input modalities. Performance using length-based target formulation on YouCook2 dense video captioning task with IoU threshold=50%. Results are reported on three ablated input settings: "Text-only" feeds in the ASR tokens, "Video-only" reveals the video features, while "Text+Video" provides both the ASR and the video features as input.

Checkpoint	F1	B@4	METEOR	CIDEr	ROUGE-L
-	30.39	0.55	1.88	0.07	5.23
T5 (full)	37.89	3.36	4.76	0.28	10.61
T5 (enc-only)	31.00	0.28	1.93	0.07	4.88
T5 (dec-only)	32.37	1.39	3.02	0.14	7.73
WikiHow T5 (full)	58.71	**9.57**	**11.99**	**0.85**	**23.21**
WikiHow T5 (enc-only)	**59.30**	8.44	11.72	0.80	22.89
WikiHow T5 (dec-only)	36.88	0.99	3.19	0.15	8.00

Table 6: Ablation on pretrained checkpoints. Performance using length-based target formulation on YouCook2 modified dense video captioning with IoU threshold=50%. Results are reported on three settings: "full" loads the complete checkpoint, "enc-only" loads the Transformer encoder weights, while "dec-only" loads the Transformer decoder weights.

*	Input Formulation	Checkpoint?	Seg-only model				Seg+Cap model					
			mIoU	Precision	Recall	F1	mIoU	F1	B@4	METEOR	CIDEr	ROUGE-L
YouCook2												
0	Random Segmentation		20.61 ± 1.04	11.25 ± 0.61	12.49 ± 0.36	10.49 ± 0.59	-	-	-	-	-	-
1	SimpleConcat	-	12.99 ± 1.55	12.24 ± 1.08	8.60 ± 0.90	9.39 ± 0.75	16.45 ± 8.72	11.23 ± 5.16	0.17 ± 0.11	0.66 ± 0.04	0.02 ± 0.01	1.99 ± 0.20
2		T5	24.14 ± 1.07	14.22 ± 0.16	15.09 ± 0.85	14.10 ± 0.44	24.21 ± 1.64	14.20 ± 1.35	0.88 ± 0.04	1.50 ± 0.12	0.09 ± 0.01	3.34 ± 0.27
3		WikiHow	22.58 ± 1.09	13.39 ± 0.96	14.57 ± 1.19	13.27 ± 1.00	23.33 ± 0.79	14.22 ± 0.94	0.67 ± 0.15	1.47 ± 0.05	0.08 ± 0.01	3.51 ± 0.13
4		WikiHow T5	**27.77** ± 0.09	**16.68** ± 1.04	**18.43** ± 0.75	**16.87** ± 0.62	**30.26** ± 1.24	**20.24** ± 1.06	**2.96** ± 0.28	**3.49** ± 0.30	**0.25** ± 0.03	**7.00** ± 0.42
5	+ T-marker	-	20.13 ± 2.59	13.68 ± 1.78	12.18 ± 1.88	12.01 ± 1.91	18.41 ± 2.65	9.99 ± 1.55	0.08 ± 0.02	0.44 ± 0.04	0.01 ± 0.00	1.33 ± 0.15
6		T5	20.29 ± 1.30	12.13 ± 2.52	11.43 ± 0.64	11.09 ± 1.38	22.12 ± 1.29	12.56 ± 0.74	0.88 ± 0.23	1.38 ± 0.22	0.08 ± 0.02	3.07 ± 0.39
7		WikiHow	19.98 ± 0.55	10.54 ± 1.36	12.11 ± 1.29	10.68 ± 1.32	20.84 ± 1.02	11.82 ± 0.64	0.39 ± 0.05	0.99 ± 0.09	0.05 ± 0.00	2.44 ± 0.18
8		WikiHow T5	20.98 ± 0.69	11.99 ± 1.07	12.49 ± 0.60	11.86 ± 0.82	20.22 ± 0.70	11.20 ± 0.70	0.38 ± 0.08	0.92 ± 0.05	0.05 ± 0.00	2.27 ± 0.13
9	+ Emb_{TIME}	-	18.51 ± 1.95	10.85 ± 0.59	11.42 ± 1.16	10.29 ± 0.58	18.71 ± 0.17	9.80 ± 0.80	0.12 ± 0.07	0.48 ± 0.08	0.02 ± 0.01	1.41 ± 0.22
10		T5	23.02 ± 1.05	13.52 ± 0.76	14.15 ± 0.94	13.23 ± 0.77	23.96 ± 0.08	15.44 ± 0.67	1.32 ± 0.08	1.91 ± 0.07	0.11 ± 0.01	4.20 ± 0.13
11		WikiHow	21.68 ± 1.93	13.13 ± 1.42	13.88 ± 1.60	12.83 ± 1.41	21.88 ± 0.86	13.15 ± 0.74	0.69 ± 0.19	1.30 ± 0.07	0.07 ± 0.01	3.06 ± 0.13
12		WikiHow T5	26.51 ± 0.45	15.61 ± 0.61	17.08 ± 0.58	15.82 ± 0.62	28.70 ± 0.92	18.71 ± 0.94	2.58 ± 0.19	3.23 ± 0.10	0.22 ± 0.01	6.45 ± 0.17
ViTT												
13	Random Segmentation		21.90 ± 0.15	12.22 ± 0.09	16.12 ± 0.25	12.48 ± 0.10	-	-	-	-	-	-
14	SimpleConcat	-	33.85 ± 0.70	23.54 ± 0.36	24.04 ± 0.40	22.98 ± 0.22	32.69 ± 0.71	22.49 ± 0.36	0.11 ± 0.01	3.76 ± 0.35	0.08 ± 0.01	3.86 ± 0.28
15		T5	37.89 ± 0.10	28.16 ± 1.18	27.15 ± 0.19	27.15 ± 0.53	38.07 ± 0.65	27.39 ± 0.91	0.57 ± 0.03	5.92 ± 0.37	0.16 ± 0.02	6.59 ± 0.69
16		WikiHow	38.20 ± 0.27	26.95 ± 0.67	27.71 ± 0.25	26.85 ± 0.41	37.80 ± 0.62	26.74 ± 0.81	0.40 ± 0.07	5.48 ± 0.18	0.14 ± 0.01	6.02 ± 0.34
17		WikiHow T5	**41.87** ± 0.26	**31.75** ± 1.94	**31.74** ± 0.34	**31.26** ± 1.10	42.40 ± 0.30	32.01 ± 0.50	**1.29** ± 0.07	**8.10** ± 0.34	**0.25** ± 0.01	**9.26** ± 0.39
18	+ T-marker	-	32.19 ± 1.17	20.05 ± 1.89	21.62 ± 0.82	20.04 ± 0.48	32.03 ± 0.14	20.89 ± 0.28	0.05 ± 0.00	2.96 ± 0.13	0.06 ± 0.00	2.93 ± 0.07
19		T5	34.94 ± 0.37	21.24 ± 0.11	23.95 ± 0.41	22.07 ± 0.21	37.56 ± 0.78	27.50 ± 0.69	0.59 ± 0.09	5.11 ± 0.52	0.16 ± 0.01	6.26 ± 0.56
20		WikiHow	33.00 ± 0.10	19.13 ± 0.87	22.02 ± 0.13	20.05 ± 0.54	35.14 ± 0.99	22.88 ± 0.41	0.23 ± 0.04	3.51 ± 0.14	0.09 ± 0.01	4.12 ± 0.37
21		WikiHow T5	34.23 ± 0.55	21.01 ± 1.34	23.26 ± 0.51	21.62 ± 0.94	33.20 ± 1.65	19.63 ± 0.98	0.16 ± 0.02	3.01 ± 0.22	0.08 ± 0.01	3.40 ± 0.36
22	+ Emb_{TIME}	-	33.89 ± 0.21	20.75 ± 2.37	23.69 ± 0.08	21.27 ± 1.49	35.37 ± 3.18	22.28 ± 0.49	0.04 ± 0.03	3.42 ± 0.61	0.07 ± 0.01	3.28 ± 0.83
23		T5	37.78 ± 0.15	25.98 ± 0.20	27.12 ± 0.16	26.05 ± 0.16	38.50 ± 0.55	27.95 ± 0.46	0.75 ± 0.10	6.37 ± 0.39	0.18 ± 0.01	7.19 ± 0.48
24		WikiHow	37.27 ± 0.08	25.96 ± 0.38	26.87 ± 0.04	25.91 ± 0.21	36.97 ± 0.48	26.37 ± 0.36	0.38 ± 0.06	5.31 ± 0.06	0.13 ± 0.01	5.82 ± 0.23
25		WikiHow T5	41.64 ± 0.12	31.07 ± 0.67	31.53 ± 0.12	30.84 ± 0.33	**43.22** ± 0.72	**32.49** ± **0.25**	1.22 ± 0.08	8.05 ± 0.20	**0.25** ± 0.01	9.18 ± 0.45

Table 7: Dense video captioning performance on YouCook2 and ViTT test sets with the length-based and the Timestamp markers formulations. We report the evaluation results (mean ± std) with models initialized from random weights, T5 checkpoints, WikiHow checkpoints, and T5 checkpoints further pretrained on WikiHow.

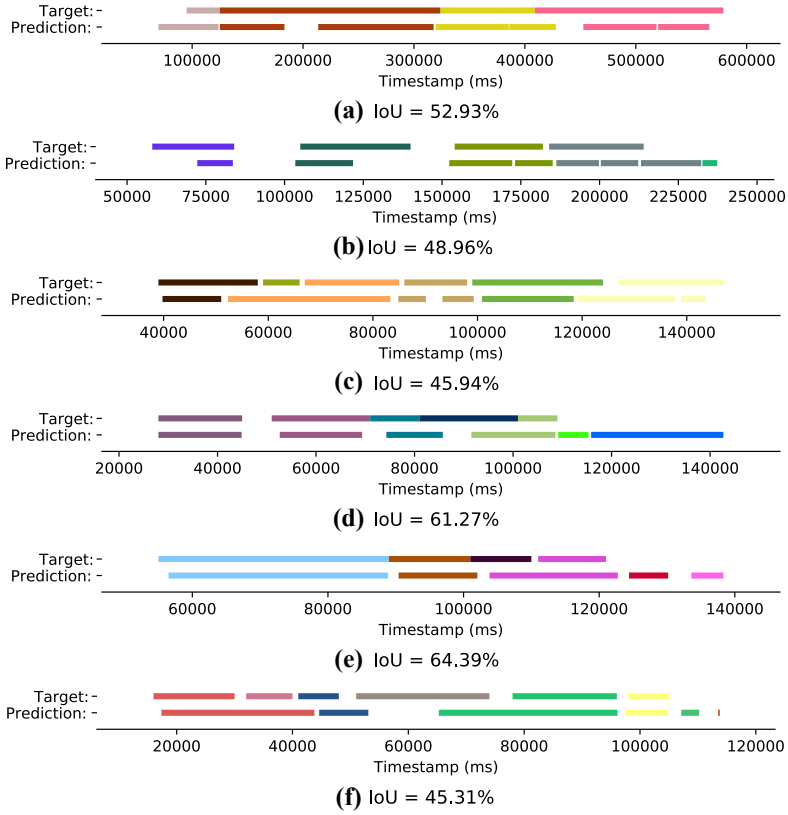

Figure 6: Examples of our Seg+Cap model's segmentation performance on YouCook2.

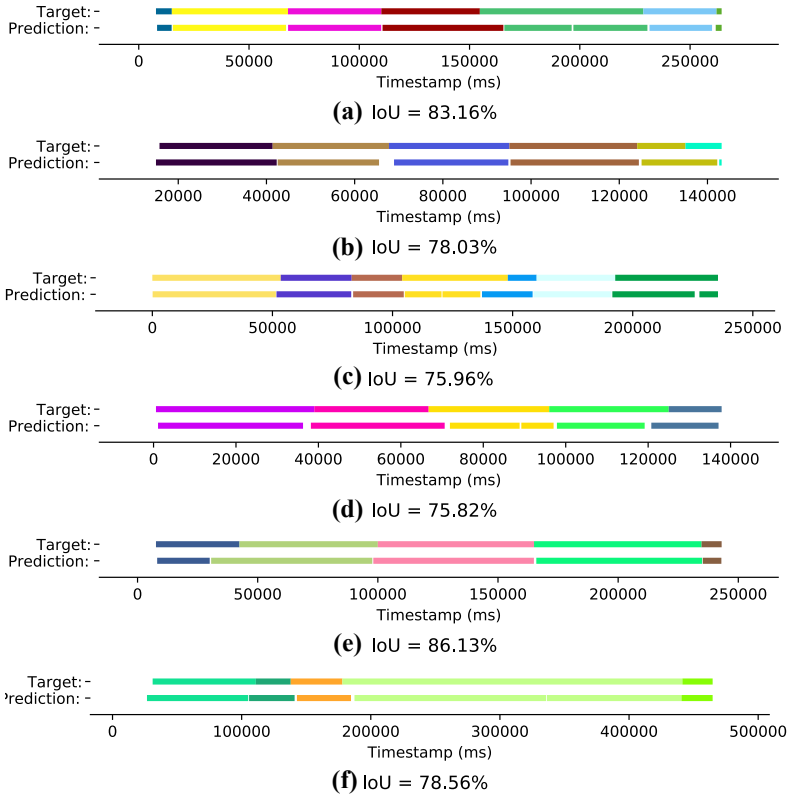

Figure 7: Examples of our Seg+Cap model's segmentation performance on ViTT.

*	Dataset	Input Formulation	Checkpoint	Seg-only model				Seg+Cap model					
				mIoU	Precision	Recall	F1	mIoU	F1	B@4	METEOR	CIDEr	ROUGE-L
0	YouCook2	Random Segmentation		20.10	11.14	12.53	10.69	-	-	-	-	-	-
1		SimpleConcat	-	12.83	12.80	8.62	9.07	11.47	8.78	0.22	0.64	0.03	1.95
2			T5	24.18	14.19	14.83	13.89	25.13	14.09	0.86	1.47	0.09	3.39
3			WikiHow	22.36	13.11	14.42	12.99	23.00	14.16	0.66	1.50	0.08	3.48
4			WikiHow T5	**27.81**	**17.21**	**18.16**	**17.01**	**30.97**	**20.57**	**2.85**	**3.48**	**0.24**	**7.02**
5		+ T-marker	-	18.82	14.42	11.28	11.38	17.18	9.53	0.06	0.45	0.01	1.34
6			T5	20.91	13.21	11.48	11.65	22.75	12.87	0.90	1.42	0.08	3.19
7			WikiHow	19.76	10.17	11.52	10.19	21.09	11.79	0.37	0.94	0.05	2.35
8			WikiHow T5	21.26	12.34	12.83	12.24	20.56	11.35	0.38	0.89	0.05	2.31
9		+ Emb_{TIME}	-	19.52	10.99	11.70	10.26	18.77	9.83	0.11	0.52	0.02	1.46
10			T5	22.93	13.84	13.78	13.30	24.00	15.70	1.34	1.90	0.11	4.24
11			WikiHow	21.41	13.11	13.88	12.76	22.08	13.18	0.79	1.30	0.07	3.09
12			WikiHow T5	26.61	15.86	17.28	16.08	28.80	18.41	2.67	3.18	0.23	6.41
13	ViTT	Random Segmentation		21.93	12.22	16.07	12.41	-	-	-	-	-	-
14		SimpleConcat	-	33.74	23.71	23.95	23.10	33.10	22.59	0.12	3.78	0.08	3.87
15			T5	37.90	28.28	27.14	27.13	38.35	27.66	0.57	5.85	0.15	6.36
16			WikiHow	38.23	26.82	27.78	26.89	37.75	26.92	0.44	5.58	0.14	6.06
17			WikiHow T5	**41.78**	**31.00**	**31.62**	**30.78**	42.25	31.85	**1.34**	7.97	**0.25**	**9.21**
18		+ T-marker	-	32.62	19.94	22.02	20.27	32.01	20.90	0.05	2.96	0.06	2.95
19			T5	34.83	21.20	23.89	22.08	37.36	27.80	0.57	5.31	0.16	6.46
20			WikiHow	33.00	19.12	22.09	20.09	35.54	23.09	0.23	3.43	0.09	4.03
21			WikiHow T5	33.92	21.12	23.01	21.52	34.04	19.46	0.16	2.96	0.07	3.23
22		+ Emb_{TIME}	-	33.79	21.21	23.68	21.61	34.56	22.37	0.05	3.12	0.06	2.92
23			T5	37.75	25.97	27.13	26.13	38.44	27.94	0.69	6.18	0.18	7.15
24			WikiHow	37.22	25.85	26.86	25.84	37.07	26.39	0.37	5.28	0.13	5.73
25			WikiHow T5	41.62	30.76	31.52	30.68	**43.51**	**32.50**	1.19	**8.05**	**0.25**	9.02

Table 8: Performance on the dense video captioning on YouCook2 and ViTT test set with the length-based and the Timestamp markers formulations. We report the evaluation results with models initialized from random weights, T5 checkpoints, WikiHow checkpoints, and T5 checkpoints further pretrained on WikiHow. We ran 3 sets of repeating experiments for each setting, and report the **median** value on each metric in this Table.

Dataset		IoU	Segment Border (ms)	Caption
Youcook2	Tgt.	82.2%	[49000.0, 67000.0]	chop 2 garlic cloves grate ginger about 2 tsp and green onions finely
	Pred.		[47114.0, 65354.5]	chop some garlic ginger and green onions and put them in a bowl
	Tgt.	82.4%	[73000.0, 117000.0]	mix an egg milk and the mashed potatoes
	Pred.		[72194.0, 109904.5]	mix the egg and milk with the potato
	Tgt.	81.1%	[89000.0, 101000.0]	mix and boil the ingredients
	Pred.		[90424.5, 102034.0]	add miso paste soy sauce diced vegetables and mushrooms to boiling water
	Tgt.	82.1%	[18000.0, 48000.0]	heat butter in a pan and cook bacon in it
	Pred.		[18224.0, 54254.0]	fry pancetta in a pan with bacon
ViTT	Tgt.	90.5%	[147890.0, 189680.0]	Dipping sticks then cake balls
	Pred.		[148905.0, 192934.0]	Dipping cake balls in candy melts
	Tgt.	90.2%	[209050.0, 253460.0]	Buttering in between baking, baking continues
	Pred.		[211445.0, 255634.5]	Brushing the dough with butter
	Tgt.	90.6%	[209630.0, 272000.0]	Stretching the hamstrings
	Pred.		[213230.0, 269750.0]	Performing the hamstring stretch
	Tgt.	92.3%	[104000.0, 144000.0]	Adding more layers
	Pred.		[105404.0, 145815.0]	Repeating the same process

Table 9: Examples of the jointly predicted segments and corresponding captions for YouCook2 and ViTT generated by our Seg+Cap model. Tgt.: Target. Pred.: Prediction.

SANCL: Multimodal Review Helpfulness Prediction with Selective Attention and Natural Contrastive Learning

Wei Han[♣*], **Hui Chen**[♣], **Zhen Hai**[†◇], **Soujanya Poria**[♣], **Lidong Bing**[†]
[♣] Singapore University of Technology and Design, Singapore
[†] DAMO Academy, Alibaba Group
{wei_han, hui_chen}@mymail.sutd.edu.sg
sporia@sutd.edu.sg
{zhen.hai, l.bing}@alibaba-inc.com

Abstract

With the boom of e-commerce, Multimodal Review Helpfulness Prediction (MRHP), which aims to sort product reviews according to the predicted helpfulness scores has become a research hotspot. Previous work on this task focuses on attention-based modality fusion, information integration, and relation modeling, which primarily exposes the following drawbacks: 1) the model may fail to capture the really essential information due to its indiscriminate attention formulation; 2) lack appropriate modeling methods that take full advantage of correlation among provided data. In this paper, we propose SANCL: **S**elective **A**ttention and **N**atural **C**ontrastive **L**earning for MRHP. SANCL adopts a probe-based strategy to enforce high attention weights on the regions of greater significance. It also constructs a contrastive learning framework based on natural matching properties in the dataset. Experimental results on two benchmark datasets with three categories show that SANCL achieves state-of-the-art baseline performance with lower memory consumption. Our implementation code for this paper can be found at https://github.com/declare-lab/SANCL.

1 Introduction

It is unbelievable to witness an e-commerce boom that has transpired over the past decades (Vulkan, 2020; Poria et al., 2020). In the virtual bazaar, countless deals are made between mutually invisible sellers and customers from time to time, under the administrator's supervision. For customers, it may be their biggest headache to determine whether they should pay for a good when being overwhelmed by tempting advertisements, as they know quite a little information about a product in face of the seller's meticulous promotions without any external references. In this situation, reviews in e-shops that can provide justification information, are thus of great value to customers. However, the quality of reviews under a certain product page can be disparate—many customers are willing to leave informative feedback on the product, while many others arbitrarily write a few words and even paste irrelevant messages in their comments. Therefore, from the perspective of online shopping platforms, they would be welcome and attractive to customers if they provide a service that can intelligently filter and place the most helpful reviews at the top position. The task in the machine learning field to solve this problem is Review Helpfulness Prediction (RHP) (Tang et al., 2013).

With the thriving of multimodal learning (Uppal et al., 2022; Hazarika et al., 2022) research and the handy accessibility of multimodal data in this Internet era, the latest progress incorporated image (vision modality) information into the review helpfulness prediction (RHP) (Liu et al., 2021) as Multimodal RHP (MRHP). Although previous work attained excellent results in MRHP, there are still some drawbacks. First, the attention mechanism in these works for representation learning follows the most basic setting—it directly computes out the attention scores based on the representation vectors of tokens or sentences, without any further intervention on the obtained weights (Fan et al., 2019). Generally, the amount of task-related information in each sentence in a given piece of review may vary greatly—since customers usually casually write these reviews and may insert some meaningless words, such as emotional appreciation or complaints that can not benefit the viewers. We observed that due to dataset characteristics in the MRHP task, there are cues to help locate those key sentences in the review text. Therefore, we proposed a probe-based selective attention mechanism to employ them for better attention results.

[*] This work was done when Wei was an intern at Alibaba DAMO Academy.
◇ Corresponding author.

Proceedings of the 29th International Conference on Computational Linguistics, pages 5666–5677
October 12–17, 2022.

Secondly, it has been revealed that the correlation, e.g., the similarity of feature vectors, among multimodal and multi-field data is an essential factor for task modeling (Xu et al., 2020; Chen et al., 2019). Nevertheless, existing studies (Xu et al., 2020; Liu et al., 2021) simply quantified them by similarity metrics, such as cosine value or negative Euclidean distance, and conduct the main tasks based on these similarity scores as features. Though gained appreciative results, we believe they can be better utilized through the contrastive learning scheme to refine the learned representations, which enables the output layer to make more accurate predictions.

In this paper, we propose a novel framework, SANCL, which incorporates these two basic ideas. We first generate a special "probe" mask that highlights the key sentences from the product and review text. Thereafter, these masks are inserted into the computation attention modules to help focus more on task-related sentences. Then we construct a contrastive learning framework, which harnesses the internal correlations within multimodal data and the fundamental contrastive predictive coding (CPC) model (Oord et al., 2018), to learn better multimodal representations for the main task. The framework is composed of two feature spaces, dubbed *domains*. Each domain takes a specific combination of projected representations as input, according to their relation types through semantic analysis. By minimizing the auxiliary contrastive loss, the multimodal and multi-domain representations can be refined with the inherent relations. Our contribution can be summarized as follows:

- We design a selective attention approach, including probe mask generation and mask-based attention computation, for the information aggregation in MRHP tasks.
- We analyze the characteristics and relations in multimodal reviews and formulate a contrastive learning framework to refine the learned representations.
- Extensive experiments on three publicly available datasets show our approach achieves state-of-the-art performance with lower memory consumption.

2 Related Work

In this section, we briefly recap some relevant work in the field of review helpfulness prediction and multimodal contrastive learning.

Review Helpfulness Prediction Customer reviews play an important role in helping customers investigate products before determining whether to purchase (Zhu and Zhang, 2010; Bing et al., 2016; Diaz and Ng, 2018; Yu et al., 2020; Gamzu et al., 2021). Support vector machine (SVM) was first employed to automatically judge the review helpfulness (Kim et al., 2006; Zhang and Varadarajan, 2006; Tsur and Rappoport, 2009). Later, linear regression (Lu et al., 2010; Ghose and Ipeirotis, 2010), extended tensor factorization (Moghaddam et al., 2012), and probabilistic matrix factorization models (Tang et al., 2013) have been applied to integrate complicated constraints into the learning process. With the development of deep learning, deep neural networks (Lee and Choeh, 2014; Fan et al., 2018; Chen et al., 2018) have been utilized to model the sophisticated elements in this task. Recently, Qu et al. (2020) proposed a graph neural network to capture the intrinsic relationship between the products and their reviews. However, most existing studies only focus on the text of reviews, neglecting the images that usually exist in online reviews. This paper takes advantage of the images and proposes a novel contrastive learning framework with a selective attention mechanism to learn expressive multimodal features.

Multimodal Representation Learning The foremost problem of multimodal tasks lies in multimodal representation learning (Baltrušaitis et al., 2018). The concept of multimodal representation learning covers many techniques, such as multimodal fusion (Vielzeuf et al., 2018; Wang et al., 2020; Mai et al., 2020; Han et al., 2021a), multimodal contrastive learning (Yuan et al., 2021; Han et al., 2021b), etc. Attention-based architectures are the basic routine in multimodal fusion, but the formulations are similar. In this paper, knowing about the particularity of MRHP and its dataset, we devise a novel attention mechanism to better aggregate information in textual data. Additionally, we also upgrade the application of contrastive learning. Unlike the ordinary treatment that divides samples into positive and negative groups according to "from myself" or "not from myself" (Cui et al., 2020; Liang et al., 2020), we extract contrastive pairs according to the natural correlation in the dataset and construct the framework of two feature spaces termed as domains.

3 Method

In this section, we first introduce the problem definition of Multimodal Review Helpfulness Prediction (MRHP). Then we elaborate on the model architecture and processing pipeline of our method.

3.1 Problem Definition

Given a collection of product descriptions $\mathcal{P} = \{P_1, P_2, ..., P_N\}$ and associated reviews $\mathcal{R} = \{R_1, R_2, ..., R_N\}$ gleaned from an e-shopping website. Each product description $P_i \in \mathcal{P}$ contains the product name n_{p_i} plus the text and image descriptions T_{p_i} and I_{p_i}. The underlying review collection R_i associated with product i contains m review pieces $R_i = \{r_{i1}, r_{i2}, ..., r_{im}\}$. Each review data frame is composed of images $I_{r_{ij}}$ and text $T_{r_{ij}}$ as well. We exhibit an example of input data at the model's input position in Figure 1. All review pieces are annotated with helpfulness scores $s_{ij} \in \{0, 1, 2, 3, 4\}$. Multimodal review helpfulness prediction can be formulated as a regression task that aims to predict the helpfulness score of each review piece, and a ranking task to sort these reviews by their scores in descending order. For the convenience of description, we call product description and review contents as *fields*, while the data of text and image are termed as *modalities* in the following sections.

3.2 Overview

The overall architecture of SANCL is depicted in Figure 1. We first generate a probe mask for each review according to the corresponding product name and review text as shown in Figure 2. The probe mask highlights the sentences that mention the product, which then participates in the computation of selective attention to produce text representations. For images, we feed the features extracted by pre-trained visual neural networks to two self-attention modules to produce image representations. Then we project these representations of each modality in both product description and customer review into two shared spaces (domains). We finally develop a contrastive learning module to compute the cross-modality and review-product contrastive scores, which further improves the quality of representations output from attention modules.

3.3 Input Encoding

Context-aware Textual Representation For both review and product text, we initialize their token representations with GloVe (Pennington et al., 2014)[1] or pre-trained models as $\mathbf{E}_t = \{e_1^t, e_2^t, ..., e_l^t\} \in \mathbb{R}^{l \times d_e^t}$, where l is the length (number of tokens) of a given sentence and d_e^t is the embedding dimension. We then send these embeddings to a uni-directional Gated Recurrent Unit (GRU) (Cho et al., 2014), yielding token-wise and sequence representations $\mathbf{H}_t = \{h_1^t, ..., h_l^t\}$ and h_t^{seq}:

$$\mathbf{H}_t, h_t^{seq} = \mathbf{GRU}(\mathbf{E}; \theta_t). \quad (1)$$

where θ_t is the parameters in GRU.

Visual Feature Extraction We apply Faster R-CNN (Ren et al., 2015) on raw images and yield the hidden representations $\mathbf{E}_v = \{e_1^v, e_2^v, ..., e_n^v\} \in \mathbb{R}^{n \times d_e^v}$ in the last layer ahead of the classifier to map the Regions of Interest (RoI) in an image to a hidden space, where n is the number of hot regions detected in the image and d_e^v is the vector lengths of hidden representations. Same as Liu et al. (2021), we feed them into a self-attention module that outputs the encoded image representations $\mathbf{H}_v = \{h_1^v, h_2^v, ..., h_n^v\}$.

3.4 Probe-based Selective Attention (PSA)

Previous work primarily formulated text attention as token-wise description–review attention which fails to differentiate the relative importance among sentences and missing really task-related information. Previous work fully relies on back-propagation to enforce the model to focus on those important token spans, but it is far from sufficient in this task—we desire to highlight valuable sentences which share the task-related information. Intuitively, task-related information is more likely to exist in the sentences where a product or its properties, prices, etc., are mentioned. To this end, we generate a customized 'probe mask' for the review text to highlight sentence-level relevance.

Probe Mask Generation The probe mask should reflect the position (i.e., in which sentence) where the product is mentioned in a review. An example of the generation process is displayed in Figure 2. We first retrieve the core words from the product

[1]We used glove.840B.300d in our experiments (https://nlp.stanford.edu/data/glove.840B.300d.zip).

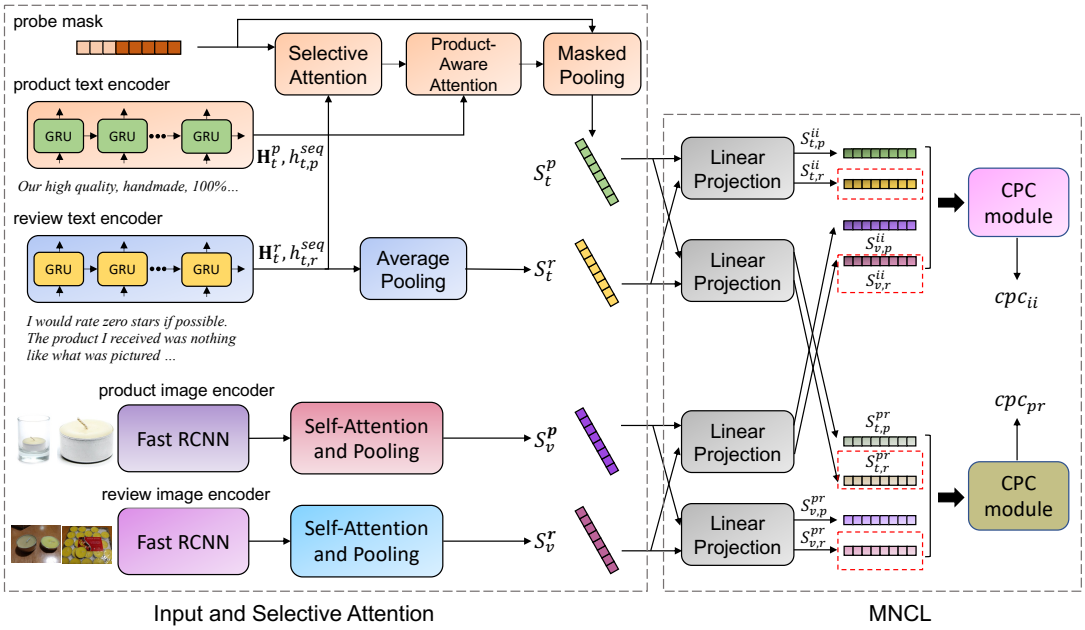

Figure 1: The overview of SANCL. The output layer is omitted. Features in red boxes ($S^{ii}_{v,r}, S^{ii}_{t,r}, S^{pr}_{v,r}, S^{pr}_{t,r}$) are used in final helpfulness score prediction.

name by looking up its dependency tree and picking the lemmatized form of the words around the root. Next, we leverage an open-sourced coreference resolution package to identify all coreference clusters in the review.

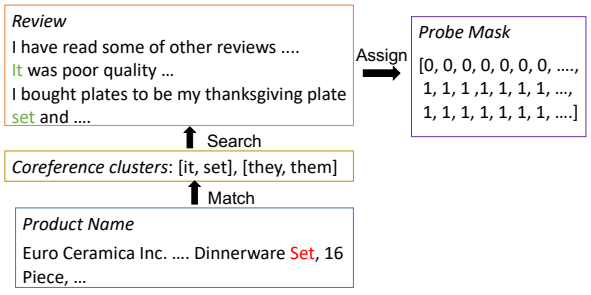

Figure 2: An example of mask generation

There are three possible resolution results: (1) A cluster containing the core word of the product name; (2) At least one cluster exists but the core word is missing in all clusters; (3) No coreference cluster exists. For (1) and (3), we do not require extra steps as the existence of entity clusters can be confirmed. For (2), we are still uncertain whether an entity cluster is in the text. We devise a simple rule to tackle this situation—we regard the first cluster as the product name mention cluster, based on our observation that the first repeatedly mentioned pronouns in a review are more likely to refer to the product. After locating these product name mentions, we create the probe mask $M \in \mathbb{R}^{1\times l}$ by assigning 1 to the positions of those mention-found sentences and 0 to others. The process is summarized in Algorithm 1.

Algorithm 1: Probe Mask Generation

Input: Review sentences R, product name P
Output: Probe mask M

 # core words and coreference clusters extraction:
$\hat{R} \leftarrow$ `Lemmatise(`R`)`, $\hat{P} \leftarrow$ `Lemmatise(`P`)`;
 # core words extraction:
$T \leftarrow$ `DependencyParse(`$\hat{P}$`)`;
$W \leftarrow$ `FindWordsNearRoot(`T`)`;
clusters $\leftarrow$ `FindCoreferenceCluster(`$\hat{R}$`)`
 # mask generation:
$M \leftarrow$ `ZeroInit(`R`.size)`
if $C = \emptyset$ **then**
  return M
end
foreach c **in** clusters **do**
  if any $w \in W$ **in** c **then**
    gold_cluster = c
  end
end
if gold_cluster $= \emptyset$ **then**
  gold_cluster = clusters[0]
end
foreach sent $\in \hat{R}$ **do**
  if any $w \in$ gold_cluster **in** sent **then**
    M[sent`.start : sent.end`] $\leftarrow True$
  end
end
return M

Selective Attention with Probe Mask There are three steps to acquire product-aware review representations—self-attention, cross-field text at-

tention (attention between product and review text), and pooling, among which the first and last steps take advantage of probe masks generated in the last step. We transform the binary probe masks into the real-value format:

$$M' = \alpha M + \beta(1 - M), \tag{2}$$

where $1 > \alpha > \beta > 0$, since we expect the mask could focus more on the sentences where the product is mentioned. This effect embodies in the self-attention computation of the review text $\mathbf{H}_t^r$, where the fundamental attention weights are computed as:

$$\mathbf{A} = \mathbf{softmax}(\mathbf{W}\mathbf{H}_t^r), \tag{3}$$

We renew the original attention matrix $A \in \mathbb{R}^{l \times l}$:

$$\mathbf{A}' = (M')^T M' \odot \mathbf{A}, \tag{4}$$

In this process, the attention weights are actually re-weighted as:

$$a'_{ij} = \begin{cases} \alpha^2 a_{ij}, \text{ if } m_i = m_j = 1 \\ \alpha\beta a_{ij}, \text{ if } m_i = 1, m_j = 0 \\ \beta^2 a_{ij}, \text{ if } m_i = m_j = 0 \end{cases}, \tag{5}$$

An intuitive explanation for this would be that a token receives more information from hot regions (whose mask value is 1) than non-hot regions, and the relative impact power of these two regions is α/β. Naturally, we set α to 1.0 while generating β individually for each review from its sequence representation:

$$\beta = \mathbf{sigmoid}(\mathbf{W}_{gen} h_{t,r}^{seq} + b_{gen}), \tag{6}$$

where $\mathbf{W} \in \mathbb{R}^{d_h \times 1}$ is the weight matrix and $b \in \mathbb{R}$ is the bias. The sigmoid function ensures that $\beta \in (0, 1)$. Then we acquire the self-attention results as in common practice:

$$\mathbf{H}_t^{r'} = \mathbf{H}_t^r + \mathbf{A}'\mathbf{H}_t^r, \tag{7}$$

In cross-field text attention (i.e., the attention between product and review text), since weighted-sum is performed on the product text, we do not utilize probe masks in this stage and obtain $\mathbf{H}_t^{r''}$. Finally, we average the result with the probe mask by a weighted sum to aggregate these sentence representations:

$$S_t^r = \mathbf{weighted\_sum}(\mathbf{H}_t^{r''}, M'). \tag{8}$$

Note that for image representations there are only cross-image attention and average pooling to yield S_v^r.

3.5 Multi-domain Natural Contrastive Learning (MNCL)

From the theory of mutual information, training to split positive samples from negative ones by their similarity can enrich the learned representations and enhance downstream tasks' performance. In our work, we are concerned about natural relations and split them into two domains: the inner-instance domain (ii) and the product (introduction)-review (pr) domain. Before forwarding the input representations into the MNCL module, all pooled representations are projected to the shared representation spaces of each domain through a projection network, which is composed of two linear layers with an activation layer in between. We denote them as $S_{m,f}^d$, where $m \in \{t, v\}$ is the modality type, $f \in \{r, p\}$ is the field (review or product description) and $d \in \{ii, pr\}$ is the domain name:

$$S_{m,f}^d = W_{m,f,2}^d \mathbf{Tanh}(W_{m,f,1}^d S_m^d + b_{m,f,1}^d) + b_{m,f,2}^d \tag{9}$$

where $W_{m,*,i}^d$ and $b_{m,*,i}^d$ are weights and biases in the i-th layer of the projection network. Note that the data in the same modality and domain share the same network parameters. In the succeeding content, we are going to describe details of the two contrastive-learning domains, mainly concerning how to pick positive and negative samples for contrastive learning and training.

Inner Instance (II) Domain In the inner instance domain, we separate positive and negative pairs according to how similar the representations between image and text are in a single training instance. First, from the sellers' perspective, the text and image of a product should match well so as to attract customers. Thus we mark text-image pairs of product descriptions as positive ones (the set of these pairs is denoted as $\mathbb{S}_{ii}^p$). Besides, from our observation, reviews that achieve high helpfulness scores possess a high similarity between their text and the attached image. Therefore, we mark the former as positive (the set is denoted as $\mathbb{S}_{ii}^+$) and the latter as negative (the collection is denoted as $\mathbb{S}_{ii}^-$).

Product-Review (PR) Domain The semantic matching property also exists between product descriptions and their associated reviews. As helpfulness is dependent on how well a review is pertinent to the theme of the product, we argue that review pieces of high helpfulness scores ($\mathbb{S}_{pr}^+$) should match the product introduction both visually and

literally, while those low-score pieces ($\mathbb{S}^-_{pr}$) match the introduction poorly in both modalities.

Multi-domain Contrastive Predictive Coding (MCPC) In contrastive predictive coding (Oord et al., 2018), we need to compute contrastive scores for every sample pair. According to the common approach (Yuan et al., 2021; Han et al., 2021b), an exponential function is chosen as the score function:

$$\varphi(\mathbf{A}, \mathbf{B}) = \exp\left(\frac{\mathbf{norm}(\mathbf{A}^T)\mathbf{norm}(\mathbf{B})}{\tau}\right), \tag{10}$$

where $\mathbf{norm}(*)$ is the l2-norm function, τ is the temperature hyper-parameter, for simplicity we keep its value 1.0 in our experiments. By noise contrastive estimation (Gutmann and Hyvärinen, 2010), in the inner instance domain, the score is computed as:

$$cpc_{ii} = -\sum_{(S_{t,j}, S_{v,j}) \in (\mathbb{S}^+_{ii} \cup \mathbb{S}^P_{ii})} \log \frac{\varphi(S_{t,j}, S_{v,j})}{\sum_{S_k \in (\mathbb{S}^+_{ii} \cup \mathbb{S}^-_{ii} \cup \mathbb{S}^P_{ii})} \varphi(S_{t,k}, S_{v,k})}, \tag{11}$$

where $(S_{t,j}, S_{v,j})$ are the text-image pair from the instance, i.e., a review piece or product description. The summation is over $\mathbb{S}^+_{ii}$ and $\mathbb{S}^P_{ii}$ because instances counted here are from both product descriptions and review pieces. Similarly in the product–review domain the score is:

$$cpc^m_{pr} = -\sum_{S^r_{m,j} \in \mathbb{S}^+_{pr}} \log \frac{\varphi(S^r_{m,j}, S^p_{m,j})}{\sum_{S_k \in (\mathbb{S}^+_{pr} \cup \mathbb{S}^-_{pr})} \varphi(S^r_{m,k}, S^p_{m,k})}. \tag{12}$$

$$cpc_{pr} = cpc^t_{pr} + cpc^v_{pr} \tag{13}$$

where $S^r_{m,j}$ is the representation of modality m in review r from the positive review set $\mathbb{S}^+_{pr}$ and $S^p_{m,j}$ is the counterpart of the corresponding product.

3.6 Prediction and Training

We select all review-related representations from the common spaces of two domains ($S^{ii}_{t,r}, S^{pr}_{t,r}, S^{ii}_{v,r}, S^{pr}_{v,r}$) and concatenate them as feature vectors for prediction ($\mathbf{F}$). A linear layer takes these feature vectors as input and outputs the helpfulness score predictions ξ_r:

$$\mathbf{F} = \mathbf{concat}([S^{ii}_{t,r}, S^{pr}_{t,r}, S^{ii}_{v,r}, S^{pr}_{v,r}]) \tag{14}$$

$$\xi_r = \mathbf{W}_o\mathbf{F} + b_o, \tag{15}$$

where $\mathbf{W_o}$ and b_o are the weight matrix and bias in the output layer. Same as Liu et al. (2021), we adopt the standard pairwise ranking loss as the main task loss:

$$\mathcal{L}_{task} = \sum_i \mathbf{max}(0, \gamma - \xi_{r^+,i} + \xi_{r^-,i}), \tag{16}$$

where r^+, r^- are an arbitrary positive/negative pair of review pieces under product P_i, γ is a scaling factor. Contrastive losses make up the auxiliary loss:

$$\mathcal{L}_{aux} = cpc_{ii} + cpc_{pr} \tag{17}$$

The total loss for training is

$$\mathcal{L} = \mathcal{L}_{task} + \kappa\mathcal{L}_{aux} \tag{18}$$

where κ is a hyper-parameter to adjust the effect of auxiliary loss.

4 Experimental Settings

This section presents the datasets, baseline models, and metrics used and compared in our experiments.

4.1 Datasets

We conduct experiments on three MRHP datasets (Liu et al., 2021) in different categories: *Clothing, Shoes & Jewelry, Home & Kitchen* and *Electronics*. The text and images in these datasets are crawled from a number of Amazon online shops under corresponding categories from the year 2017 to 2019. The helpfulness scores equal to $\lfloor \log_2 n_{\text{votes}} \rfloor$ and are then clipped into $[0, 4]$. More details of datasets are provided in Appendix.

4.2 Baseline Models

Following previous work, we first compare our model with a bunch of baselines on the text-only setting, which examines the effect of our selective attention mechanism and text-related contrastive learning modules. The baseline candidates contain Multi-Perspective Matching (BiMPM) network (Wang et al., 2017), Embedding-gated CNN (EG-CNN) (Chen et al., 2018), Convolutional Kernel-based Neural Ranking Model (Conv-KNRM) (Dai et al., 2018) and Product-aware Helpfulness Prediction Network (PRHNet) (Fan et al., 2019). In multimodal settings, we pick a collection of state-of-the-art multimodal helpfulness prediction models for comparison:

Setting	Model	Cloth. & Jew			Electronics			Home & Kitchen		
		MAP	**N-3**	**N-5**	**MAP**	**N-3**	**N-5**	**MAP**	**N-3**	**N-5**
Text-only	BiMPM* (Wang et al., 2017)	57.7	41.8	46.0	52.3	40.5	44.1	56.6	43.6	47.6
	EG-CNN* (Chen et al., 2018)	56.4	40.6	44.7	51.5	39.4	42.1	55.3	42.4	46.7
	Conv-KNRM* (Dai et al., 2018)	57.2	41.2	45.6	52.6	40.5	44.2	57.4	44.5	48.4
	PRHNet† (Fan et al., 2019)	58.23	43.36	47.21	52.31	40.43	43.88	57.11	44.46	48.27
	SANCL (Ours)	**58.98**$^{\natural}$	**44.75**$^{\natural}$	**48.57**$^{\natural}$	**53.03**$^{\natural}$	**41.03**$^{\natural}$	**44.77**$^{\natural}$	**58.03**$^{\natural}$	**45.59**$^{\natural}$	**49.31**$^{\natural}$
	BERT (Devlin et al., 2018)	56.47	42.98	46.84	51.95	39.77	43.11	56.62	42.12	46.87
	PRHNet+BERT† (Fan et al., 2019)	57.51	43.65	47.74	52.28	40.66	44.02	57.32	44.74	48.42
	SANCL+BERT (Ours)	**58.49**$^{\natural}$	**44.91**$^{\natural}$	**48.69**$^{\natural}$	**53.13**$^{\natural}$	**41.77**$^{\natural}$	**45.01**$^{\natural}$	**58.20**$^{\natural}$	**45.83**$^{\natural}$	**49.65**$^{\natural}$
Multimodal	SSE-Cross* (Abavisani et al., 2020)	65.0	56.0	59.1	53.7	43.8	47.2	60.8	51.0	54.0
	D&R Net* (Xu et al., 2020)	65.2	56.1	59.2	53.9	44.2	47.5	61.2	51.8	54.6
	MCR† (Liu et al., 2021)	66.96	58.03	61.06	55.86	46.32	49.45	63.17	53.85	57.14
	SANCL (Ours)	**67.26**	**58.61**$^{\natural}$	**61.48**$^{\natural}$	**56.19**	**46.98**$^{\natural}$	**49.92**$^{\natural}$	**63.35**	**54.28**$^{\natural}$	**57.40**
	MCR+BERT (Liu et al., 2021)	65.81	55.94	58.75	55.15	45.67	48.62	62.39	52.91	56.09
	SANCL+BERT (Ours)	**66.52**$^{\natural}$	**56.73**$^{\natural}$	**59.90**$^{\natural}$	**56.04**$^{\natural}$	**46.77**$^{\natural}$	**49.95**$^{\natural}$	**62.74**	**53.65**$^{\natural}$	**56.91**$^{\natural}$

Table 1: Results on three datasets; all reported metrics are the average of five runs; "$*$" are from Liu et al. (2021) and "†" are from the open-source code in Liu et al. (2021); "$\natural$" represent the results significantly outperforms PRHNet and MCR with p-value < 0.05 based on paired t-test.

- **SSE-Cross** (Abavisani et al., 2020): The Stochastic Shared Embeddings (SSE) Cross-modal Attention Network introduces a novel cross-attention mechanism that can filter noise components from weak modalities which may mislead the model to make wrong predictions on a sample. SSE is adopted as the regularization technique to alleviate over-fitting in the fusion process to further prompt the prediction accuracy.

- **D&R Net** (Xu et al., 2020): The Decomposition and Relation Network learns the commonality and discrepancy between image and text in the decomposition network and the multi-view semantic association information in the relation network.

- **MCR** (Liu et al., 2021): The Multi-perspective Coherent reasoning method incorporates the joint reasoning across textual and visual modalities from both the product and the review. Three types of coherence are modeled to learn effective modality representations for helpfulness prediction.

In both settings, we also test our method with BERT (Devlin et al., 2018) as the text encoder. In addition, we test and record the BERT-with-head performance (BERT+a double linear layers) as the blank comparison experiment.

4.3 Metrics

We utilize several metrics for ranking tasks to evaluate the performance of these models. After sorting all prediction scores by their corresponding truth scores in descending order, the Mean Average Precision (MAP) computes the mean precision till the sample with the k-th highest score. K is usually large enough to encompass the entire collection of reviews under every product. The Normalized Discounted Cumulative Gain (NDCG-N) (Järvelin and Kekäläinen, 2017; Diaz and Ng, 2018) purely reckons the gain value over top-N predictions (N is 3 and 5 in our experiments), which simulates the real circumstances of a typical customer who would always read the topmost reviews.

5 Results and Analysis

In this section, we will compare our approach with several advanced baselines and explore how it improves the multimodal helpfulness prediction task.

5.1 Performance Comparison

We list the performance of our model and baselines in Table 1. Notably, SANCL consistently outperforms all the baselines in both text-only and multimodal settings and under both BERT and Glove initialization methods. These outcomes initially demonstrate the efficacy of our method in MRHP tasks. It is surprising that we cannot gain a significant performance boost by replacing Glove with BERT as the text encoder. We speculate the reason is that Glove embeddings are expressive enough for this task.

Moreover, it can be claimed that SANCL is a lightweight model compared to the multimodal SOTA since the model size and GPU memory consumption of SANCL are much lower than MCR. The total number of parameters is 2.63M in MCR

and 1.41M (excluding the embedding layer) in SANCL respectively, which indicates a double efficiency. The average GPU memory usage of SANCL during the training on Amazon-MRHP Home & Kitchen is around 2.4G, while MCR occupies an average of 13.7G GPU memory during training, which is 4.7 times higher than SANCL.

Description	MAP	N-3	N-5
SANCL	**56.19**	**46.98**	**49.92**
Attention			
w/o learned β (fixed at 0.5)	55.61	46.37	49.58
w/o probe mask	55.43	46.11	49.45
Contrastive learning			
w/o cpc_{ii}	55.54	46.29	49.23
w/o cpc_{pr}	55.81	46.40	49.47
w/o cpc_{ii} and cpc_{pr}	55.35	46.28	49.09

Table 2: Ablation study of SANCL on the Electronics dataset.

5.2 Ablation Study

To verify the benefits of our proposed method, we carry out comprehensive ablation experiments on the Amazon electronics dataset, including the selective attention and contrastive learning components. In selective attention, we first replace learned β with a fixed value of 0.5, since we find most β values in our experiments are around 0.5. Next, we remove the entire selective attention module and only preserve the primitive attention computation. The decline in the outcome of both situations manifests that probe-based selective attention amends the cross-text information exchange between text fields. For multi-domain contrastive learning, we delete the CPC losses of a single or both domains in training. The results indicate that both domains have a positive impact on performance. Moreover, the effect of the two domains does not counteract their collaboration, as we observe accumulated benefits when they operate together.

5.3 Case Study

To understand how our model deals with samples in-depth, we randomly pick up a test product-review instance from the test set of Amazon Home & Kitchen to explain how SANCL works, as shown in Table 3.

In this example, the customer bought the pins to fix the edge of his sofa. Instead of photoing pins themselves, the customer only presented the tidy sofa after installing the pins. We first visualize the attention weights in test time, as shown in Fig. 3. Note that only the first sentence in the review contains the elements in the coreference clusters, which we have emphasized with italics and underlined in Table 3. Consistently, we observe the significantly larger weights in the region of the first sentence (row/column 1-19) while the rest region's weights are much smaller. We also ran MCR and collect its prediction on this example, and it is clear that MCR commits a severe error here, probably caused by the direct classification of the unimodal cosine similarity. In our approach, as we carefully analyze and classify the positive and negative pairs in the multi-domain contrastive learning framework, the huge semantic similarity between review text and image and between product description and review text, indicated by the high CPC scores $S^{ii}_{v,r}$, assists the model to correctly predict the score.

Product Name: Twisty Pins for Upholstery, Slipcovers, and Bedskirts 50/pkg

Product description: Package of 50 Clear Twisty Pins for securing fabrics and accent trims. Nickel-plated steel pin 1/2" in diameter clear top, wire twist 3/8" long. Perfect for Medium to lightweight fabrics, bed skirts, bed ruffles, slipcovers, and upholstery.

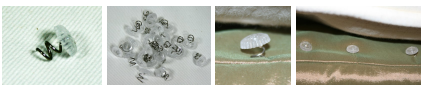

Review (Helpfulness Score: 4): I bought *<u>these</u>* to pin the loose material on a sofa cover and they worked like a charm. The sofa cover definitely looks form fitting now.

Predictions: SANCL: 4.5291 MCR: -1.0832

CPC score: $cpc_{ii} = 0.82, cpc^{t}_{pr} = 0.76, cpc^{v}_{pr} = 0.21$

Table 3: Examples from the Amazon Home & Kitchen test set.

6 Conclusion

We propose a novel framework, SANCL, for the task of multimodal review helpfulness prediction (MRHP) in this paper. We first present a selective attention mechanism, which purposefully aggregates information from these crucial sentences in the review text by generating the probe mask that exerts re-normalization on the attention weights and pooling stage. We then build up a multi-domain natural contrastive learning framework in

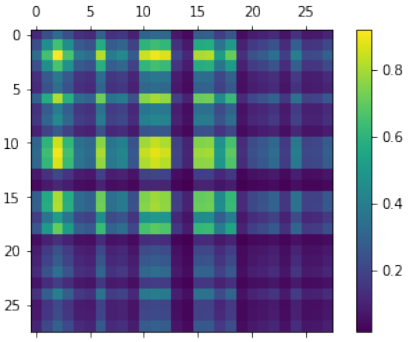

Figure 3: The self-attention weights of the review text in the given example at test time (β=0.57).

our model. It exploits the natural relations among the data from different fields and modalities in the dataset to enhance the model's capacity for multimodal representation learning. Results of comprehensive experiments and analyses demonstrate the superiority of our model over the comparable baselines and the efficacy of the novel components.

Acknowledgement

This work is supported by the A*STAR under its RIE 2020 AME programmatic grant RGAST2003 and project T2MOE2008 awarded by Singapore's MoE under its Tier-2 grant scheme.

References

Mahdi Abavisani, Liwei Wu, Shengli Hu, Joel Tetreault, and Alejandro Jaimes. 2020. Multimodal categorization of crisis events in social media. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 14679–14689.

Tadas Baltrušaitis, Chaitanya Ahuja, and Louis-Philippe Morency. 2018. Multimodal machine learning: A survey and taxonomy. *IEEE transactions on pattern analysis and machine intelligence*, 41(2):423–443.

Lidong Bing, Tak-Lam Wong, and Wai Lam. 2016. Unsupervised extraction of popular product attributes from e-commerce web sites by considering customer reviews. *ACM Trans. Internet Technol.*, 16(2):1–17.

Cen Chen, Minghui Qiu, Yinfei Yang, Jun Zhou, Jun Huang, Xiaolong Li, and Forrest Sheng Bao. 2019. Multi-domain gated cnn for review helpfulness prediction. In *The World Wide Web Conference*, pages 2630–2636.

Cen Chen, Yinfei Yang, Jun Zhou, Xiaolong Li, and Forrest Bao. 2018. Cross-domain review helpfulness prediction based on convolutional neural networks with auxiliary domain discriminators. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 2 (Short Papers)*, pages 602–607.

Kyunghyun Cho, Bart Van Merriënboer, Caglar Gulcehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. 2014. Learning phrase representations using rnn encoder-decoder for statistical machine translation. *arXiv preprint arXiv:1406.1078*.

Wanyun Cui, Guangyu Zheng, and Wei Wang. 2020. Unsupervised natural language inference via decoupled multimodal contrastive learning. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 5511–5520.

Zhuyun Dai, Chenyan Xiong, Jamie Callan, and Zhiyuan Liu. 2018. Convolutional neural networks for soft-matching n-grams in ad-hoc search. In *Proceedings of the eleventh ACM international conference on web search and data mining*, pages 126–134.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*.

Gerardo Ocampo Diaz and Vincent Ng. 2018. Modeling and prediction of online product review helpfulness: a survey. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 698–708.

Miao Fan, Chao Feng, Lin Guo, Mingming Sun, and Ping Li. 2019. Product-aware helpfulness prediction of online reviews. In *The World Wide Web Conference*, pages 2715–2721.

Miao Fan, Yue Feng, Mingming Sun, Ping Li, Haifeng Wang, and Jianmin Wang. 2018. Multi-task neural learning architecture for end-to-end identification of helpful reviews. In *2018 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining (ASONAM)*, pages 343–350. IEEE.

Iftah Gamzu, Hila Gonen, Gilad Kutiel, Ran Levy, and Eugene Agichtein. 2021. Identifying helpful sentences in product reviews. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 678–691, Online. Association for Computational Linguistics.

Anindya Ghose and Panagiotis G Ipeirotis. 2010. Estimating the helpfulness and economic impact of product reviews: Mining text and reviewer characteristics. *IEEE transactions on knowledge and data engineering*, 23(10):1498–1512.

Michael Gutmann and Aapo Hyvärinen. 2010. Noise-contrastive estimation: A new estimation principle for unnormalized statistical models. In *Proceedings of the thirteenth international conference on artificial intelligence and statistics*, pages 297–304. JMLR Workshop and Conference Proceedings.

Wei Han, Hui Chen, Alexander Gelbukh, Amir Zadeh, Louis-philippe Morency, and Soujanya Poria. 2021a. Bi-bimodal modality fusion for correlation-controlled multimodal sentiment analysis. In *Proceedings of the 2021 International Conference on Multimodal Interaction*, pages 6–15.

Wei Han, Hui Chen, and Soujanya Poria. 2021b. Improving multimodal fusion with hierarchical mutual information maximization for multimodal sentiment analysis. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 9180–9192.

Devamanyu Hazarika, Yingting Li, Bo Cheng, Shuai Zhao, Roger Zimmermann, and Soujanya Poria. 2022. Analyzing modality robustness in multimodal sentiment analysis. *arXiv preprint arXiv:2205.15465*.

Kalervo Järvelin and Jaana Kekäläinen. 2017. Ir evaluation methods for retrieving highly relevant documents. In *ACM SIGIR Forum*, volume 51, pages 243–250. ACM New York, NY, USA.

Soo-Min Kim, Patrick Pantel, Timothy Chklovski, and Marco Pennacchiotti. 2006. Automatically assessing review helpfulness. In *Proceedings of the 2006 Conference on empirical methods in natural language processing*, pages 423–430.

Sangjae Lee and Joon Yeon Choeh. 2014. Predicting the helpfulness of online reviews using multilayer perceptron neural networks. *Expert Systems with Applications*, 41(6):3041–3046.

Zujie Liang, Weitao Jiang, Haifeng Hu, and Jiaying Zhu. 2020. Learning to contrast the counterfactual samples for robust visual question answering. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 3285–3292.

Junhao Liu, Zhen Hai, Min Yang, and Lidong Bing. 2021. Multi-perspective coherent reasoning for helpfulness prediction of multimodal reviews. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 5927–5936.

Yue Lu, Panayiotis Tsaparas, Alexandros Ntoulas, and Livia Polanyi. 2010. Exploiting social context for review quality prediction. In *Proceedings of the 19th international conference on World wide web*, pages 691–700.

Sijie Mai, Haifeng Hu, and Songlong Xing. 2020. Modality to modality translation: An adversarial representation learning and graph fusion network for multimodal fusion. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pages 164–172.

Samaneh Moghaddam, Mohsen Jamali, and Martin Ester. 2012. Etf: extended tensor factorization model for personalizing prediction of review helpfulness. In *Proceedings of the fifth ACM international conference on Web search and data mining*, pages 163–172.

Aaron van den Oord, Yazhe Li, and Oriol Vinyals. 2018. Representation learning with contrastive predictive coding. *arXiv preprint arXiv:1807.03748*.

Jeffrey Pennington, Richard Socher, and Christopher D Manning. 2014. Glove: Global vectors for word representation. In *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*, pages 1532–1543.

Soujanya Poria, Devamanyu Hazarika, Navonil Majumder, and Rada Mihalcea. 2020. Beneath the tip of the iceberg: Current challenges and new directions in sentiment analysis research. *IEEE Transactions on Affective Computing*.

Xiaoru Qu, Zhao Li, Jialin Wang, Zhipeng Zhang, Pengcheng Zou, Junxiao Jiang, Jiaming Huang, Rong Xiao, Ji Zhang, and Jun Gao. 2020. Category-aware graph neural networks for improving e-commerce review helpfulness prediction. In *Proceedings of the 29th ACM International Conference on Information & Knowledge Management*, pages 2693–2700.

Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun. 2015. Faster r-cnn: Towards real-time object detection with region proposal networks. *Advances in neural information processing systems*, 28:91–99.

Chi Sun, Xipeng Qiu, Yige Xu, and Xuanjing Huang. 2019. How to fine-tune bert for text classification? In *China National Conference on Chinese Computational Linguistics*, pages 194–206. Springer.

Jiliang Tang, Huiji Gao, Xia Hu, and Huan Liu. 2013. Context-aware review helpfulness rating prediction. In *Proceedings of the 7th ACM Conference on Recommender Systems*, pages 1–8.

Oren Tsur and Ari Rappoport. 2009. Revrank: A fully unsupervised algorithm for selecting the most helpful book reviews. In *Proceedings of the International AAAI Conference on Web and Social Media*, volume 3.

Shagun Uppal, Sarthak Bhagat, Devamanyu Hazarika, Navonil Majumder, Soujanya Poria, Roger Zimmermann, and Amir Zadeh. 2022. Multimodal research in vision and language: A review of current and emerging trends. *Information Fusion*, 77:149–171.

Valentin Vielzeuf, Alexis Lechervy, Stéphane Pateux, and Frédéric Jurie. 2018. Centralnet: a multilayer approach for multimodal fusion. In *Proceedings of the European Conference on Computer Vision (ECCV) Workshops*, pages 0–0.

Nir Vulkan. 2020. *The Economics of E-commerce*. Princeton University Press.

Yikai Wang, Wenbing Huang, Fuchun Sun, Tingyang Xu, Yu Rong, and Junzhou Huang. 2020. Deep multimodal fusion by channel exchanging. *Advances in Neural Information Processing Systems*, 33.

Zhiguo Wang, Wael Hamza, and Radu Florian. 2017. Bilateral multi-perspective matching for natural language sentences. In *IJCAI*.

Nan Xu, Zhixiong Zeng, and Wenji Mao. 2020. Reasoning with multimodal sarcastic tweets via modeling cross-modality contrast and semantic association. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 3777–3786.

Qian Yu, Lidong Bing, Qiong Zhang, Wai Lam, and Luo Si. 2020. Review-based question generation with adaptive instance transfer and augmentation. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 280–290, Online. Association for Computational Linguistics.

Xin Yuan, Zhe Lin, Jason Kuen, Jianming Zhang, Yilin Wang, Michael Maire, Ajinkya Kale, and Baldo Faieta. 2021. Multimodal contrastive training for visual representation learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 6995–7004.

Zhu Zhang and Balaji Varadarajan. 2006. Utility scoring of product reviews. In *Proceedings of the 15th ACM international conference on Information and knowledge management*, pages 51–57.

Feng Zhu and Xiaoquan Zhang. 2010. Impact of online consumer reviews on sales: The moderating role of product and consumer characteristics. *Journal of marketing*, 74(2):133–148.

A Dataset Specification

Specifications of Amazon-MRHP are listed below.

Amazon-MRHP (Products/Reviews)			
Category	Cloth. & Jew.	Elec.	Home & Kitch.
Train	12074/277308	10564/240505	14570/369518
Dev	3019/122148	2641/84402	3616/92707
Test	3966/87492	3327/79750	4529/111593

Table 4: Statistics of the Amazon-MRHP dataset.

B Hyperparameter Search

The optimal hyperparameter settings are provided in Table 5 and 6.

Glove Hyperparameters			
	Cloth. & Jew.	Elec.	Home & Kitch.
learning rate	$1e^{-4}$	$5e^{-5}$	$1e^{-4}$
text embedding dim	300	300	300
text embedding dropout	0.5	0.5	0.2
image embedding dim	128	128	128
LSTM hidden dim	128	128	128
shared space hidden	64	64	64
κ	0.25	0.1	0.1
batch size	32	32	32

Table 5: Hyperparameters for all categories using glove-300d embeddings.

Amazon-MRHP Hyperparameters			
	Cloth. & Jew.	Elec.	Home & Kitch.
learning rate	$2e^{-5}$	$2e^{-5}$	$2e^{-5}$
text embedding dim	768	768	768
text embedding dropout	0.5	0.5	0.5
LSTM hidden dim	128	128	128
image embedding dim	128	128	128
shared space hidden	64	64	64
κ	0.3	0.25	0.25
batch size	32	32	32

Table 6: Hyperparameters for all categories using BERT as encoder

We use the same set of settings for text-only and multimodal modes for the same category dataset. The search space of these hyperparameters are: learning rate in $\{1e^{-4}, 2e^{-5}\}$, text embedding dropout in $\{0.2, 0.5\}$, κ in $\{0.1, 0.25, 0.3, 0.5\}$, shared space hidden dimension in $\{64, 128\}$. We train and test each dataset on a single Tesla V100 GPU. In BERT experiments, we use shared a BERT encoder for both product description and review text. To balance the computation cost and model performance, following Sun et al. (2019), we fine-tune the last four layers of the BERT encoder.

C Language Tools

For coreference resolution, we use neuralcoref[2], an extension that can be placed on SpaCy processors. For BERT model, we use the huggingface[3] transformers package to load.

[2] https://github.com/huggingface/neuralcoref

[3] https://huggingface.co/docs/transformers/model_doc/bert

Dual Capsule Attention Mask Network with Mutual Learning for Visual Question Answering

Weidong Tian[1,3], **Haodong Li**[1,3], **Zhong-Qiu Zhao**[1,2,3,4,*]
[1]College of Computer and Information, Hefei University of Technology
[2]Guangxi Academy of Science
[3]Intelligent Manufacturing Institute of HFUT
[4]Intelligent Interconnected Systems Laboratory of Anhui Province, Hefei University of Technology
wdtian@hfut.edu.cn, lhd_hfut@163.com, z.zhao@hfut.edu.cn

Abstract

A Visual Question Answering (VQA) model processes images and questions simultaneously with rich semantic information. The attention mechanism can highlight fine-grained features with critical information, thus ensuring that feature extraction emphasizes the objects related to the questions. However, unattended coarse-grained information is also essential for questions involving global elements. We believe that global coarse-grained information and local fine-grained information can complement each other to provide richer comprehensive information. In this paper, we propose a dual capsule attention mask network with mutual learning for VQA. Specifically, it contains two branches processing coarse-grained features and fine-grained features, respectively. We also design a novel stackable dual capsule attention module to fuse features and locate evidence. The two branches are combined to make final predictions for VQA. Experimental results show that our method outperforms the baselines in terms of VQA performance and interpretability and achieves new SOTA performance on the VQA-v2 dataset.

1 Introduction

In recent years, visual question answering (VQA) has received extensive research attention in the fields of computer vision and multimedia computing. The goal of VQA is to answer questions related to the content of images correctly (Antol et al., 2015). It has a wide range of practice applications, such as helping people with visual impairment and human-computer Q&A.

In the early stage, most VQA models extract features from images and questions independently (Malinowski et al., 2015; Gao et al., 2015; Ren et al., 2015). These methods fail to capture the fine-grained key features and include much unnecessary information. Afterward, the attention mechanism becomes popular and is introduced in many fields (Lu et al., 2016; Cai and Hu, 2020). The VQA models with the attention mechanism extract critical information from one modality guided by another modality (Lu et al., 2016; Anderson et al., 2018; Yu et al., 2019). Consider the question related to Figure 1 (Left) "What is on the little girl's head?" The attention method needs to encourage the model to focus on the "girls head" in the question and related regions in the image to produce the correct answer as "helmet." In this case, local fine-grained input(features which have critical information with attention processing) can help the model eliminate distractions and generate correct answers. However, the attention mechanism is not a panacea. Some questions may mislead visual attention and lead to wrong answers. For example, in Figure 6 (line 3, left), the question word "boxes" makes the model focus on the printer(which looks like a box) in the picture and leads to the wrong answer. Also, in some scenarios, the model needs to focus on multiple objects for reasoning, but the question cannot explicitly remind which object require its attention. For example, consider the ques-

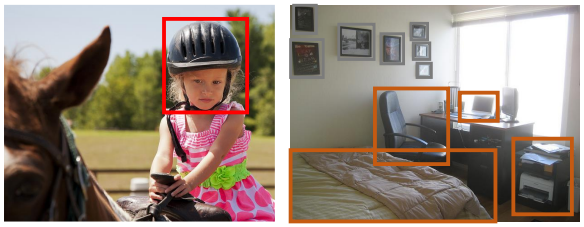

Figure 1: Samples in the VQA-v2 dataset. (Left): The fine-grained features with attention have the critical information required for the answer inference, which can help the model generate the correct answer by eliminating the interference of irrelevant factors. (Right): Unattended coarse-grained features have richer semantic information, which can help the answer inference when the attention mechanism is of limited use.

*Corresponding author.

Proceedings of the 29th International Conference on Computational Linguistics, pages 5678–5688
October 12–17, 2022.

tion related to Figure 1 (Right) "Could this be a multi-purpose room?" The model needs to take the bed, chair, computer, and printer into account for reasoning, but there are no words in the question that can help identify the relevant objects. In these cases, global coarse-grained input(features that have all information without attention processing) can provide more comprehensive information for generating answers. The challenge is to make the model focus on key features while maintaining a reference to the global information.

To overcome the above limitations, we propose a novel Dual Capsule Attention Mask Network (DCAMN) with mutual learning to process multimodal information at different granularities. We believe that coarse-grained information and fine-grained information can complement each other to provide richer comprehensive information for answer reasoning. Inspired by mutual learning and its variants (Zhang et al., 2018; Song and Chai, 2018), we design a two-branch network. The first branch processes the entire features of visual and language, analyzes global information at the coarse-grained level and fuses the features to produce predictions. We also design a Stackable Dual Capsule Attention Module (SDCAM) to model cross-modal deep interactions between the image and the question. The second branch masks visual features and language features with the attention weights generated by the SDCAM of the first branch to get fine-grained features, which enables the network to focus on key regions of the image and keywords of the question. Finally, we combine results from two branches to get final predictions. In contrast to other multi-granularity work (Nguyen et al., 2021), our DCAMN does not introduce additional information such as predicates, while utilizes the attention weights from the first branch to filter features for the second branch.

As a novel method of multi-modal fusion, SDCAM can output precise attention weights, which not only mask fine-grained features but also help locate evidence(grounds for the answer). By analyzing evidence, we can learn what information the network concerns more and how it makes decisions. Also, the stacking strategy for SDCAM improves attention accuracy and the fusion of visual and language representations. In DCAMN, there is only one language module to encode the questions, while images are presented to the two branches for processing separately. Compared with using two independent peer networks, sharing the language module between two branches can reduce the burden of parameters and calculation. Moreover, early blocks acquire gradients from both branches during backpropagation, which reduces the risk of gradient vanishing (Song and Chai, 2018). The knowledge of two branches at different perspectives and granularities is learned by another branch for information supplement and regularization, which can improve the generalization capability and VQA performance of DCAMN.

The main contributions of this paper are as follows: (1) A novel dual capsule attention mask network with mutual learning is proposed, which can process coarse-grained features and fine-grained features separately. Two branches can learn from each other, and their combination can improve the VQA performance. (2) A stackable dual capsule attention module is proposed, which provides precise co-attention weights for masking out features and locating evidence. (3) We propose to share the language module between two branches of different granularities, which can reduce parameter requirements and the risk of gradient vanishing. (4) Extensive experiments are conducted to evaluate the proposed method. Our method has significant advantages over the baselines in terms of interpretability and accuracy and achieves state-of-the-art performance on the VQA-v2 dataset.

2 Related Work

2.1 Visual Question Answering

The rapid development of VQA has benefited from many aspects. The latest studies in visual and language feature representation are applied to VQA to improve the ability to extract and process features (Jiang et al., 2020; Devlin et al., 2019). Better multimodal fusion methods, such as MCB (Fukui et al., 2016), MLB (Kim et al., 2017), MUTAN (Ben-younes et al., 2017), etc., are proposed to capture the high-level interactions between visual and language features. The transformer significantly contributes to the improvement of VQA (Yu et al., 2019; Zhou et al., 2021) and makes large-scale pre-training possible (Chen et al., 2020; Li et al., 2020). To extract useful information from the cumbersome features, many approaches introduce the attention mechanism to refine key information (Lu et al., 2016; Anderson et al., 2018; Gao et al., 2019; Yu et al., 2019). However, The attention mechanism may perform poorly on ques-

tions involving background. (Sharma and Jalal, 2022). Moreover, some words in the questions may mislead the question-based attention. These situations indicate that the attention mechanism is not a panacea, and a method that can integrate attention features and global features is necessary.

2.2 Mutual Learning

Most distillation-based model compression methods distill large and powerful networks into smaller and efficient networks (Romero et al., 2015; Zhang and Ma, 2021). However, its two-step strategy(train the teacher first and then train students) is time-consuming. So mutual learning is proposed (Zhang et al., 2018), which enables networks to be trained in parallel. In Zhang et al. (2018), each student model learns from the predictions of the other members, and the whole network requires complex asynchronous updates among different students. Another approach (Song and Chai, 2018) advocates that all students share the same early module, aggregating the gradient flow from all branches. This strategy reduces the training computational complexity and facilitates the supervision of the shared layers. Our network is based on the latter.

2.3 Capsule Network

The idea of grouping neurons is proposed early in Hinton et al. (2011). Following this, the dynamic routing method with capsules is formally introduced in Sabour et al. (2017). After dynamic routing, Hinton et al. (2018) implement EM routing of matrix capsules. The applications in other domains prove the universality of capsule networks (Duarte et al., 2018; Jaiswal et al., 2018; Zhao et al., 2019). Zhou et al. (2019), inspired by dynamic-routing implementation of capsule networks (Sabour et al., 2017), proposed CapsAtt model replacing the previously multi-level attentions.

3 Method

Given an image **I** and a question **Q**, the purpose of VQA is to output the correct answer $a \in \mathbf{A}$, where **A** indicates the candidate word list for answers. We follow the transformer design of MCAN (Yu et al., 2019) and TRAR (Zhou et al., 2021) and use them as our backbones with their encoder-decoder units. The overall framework of DCAMN is shown in Figure 2.

3.1 Question Representation and Image Representation

DCAMN models language features and visual features by encoding layers and decoding layers, respectively. The question **Q** is first processed by Glove word embedding (Pennington et al., 2014) accompanied by LSTM (Hochreiter and Schmidhuber, 1997) and then is presented to the encoding layers to get the question feature matrix $\mathbf{Y} \in \mathbb{R}^{k \times d}$, where d is the latent dimensionality in multi-head attention of encoder-decoder units, and k denotes the number of words in the question. Following BUTD (Anderson et al., 2018), we extract the salient region features $\mathbf{X}_0 \in \mathbb{R}^{m \times d}$ from the image **I** as the visual input by a pre-trained Faster R-CNN. $\mathbf{X}_0$ are then fed into two independent decoding structures to obtain the visual feature matrices $\mathbf{X}_t^1 \in \mathbb{R}^{m \times d}$ and $\mathbf{X}_t^2 \in \mathbb{R}^{m \times d}$ $(t = [1, 2, \cdots, T])$, where $\mathbf{X}_t^b$ represents the output of the t-th decoder block of branch b ($b \in [1, 2]$); T denotes the number of decoding layers; m is the number of bounding boxes.

3.2 Stackable Dual Capsule Attention Module and Decoder Block

We design an efficient multi-modal fusion method SDCAM. The attention of SDCAM is transformed from capsule dynamic routing, and the stacking strategy is also employed to improve visual-language co-attention performance. The details of SDCAM are shown in Figure 3. Specifically, it alternately performs attention and fusion on visual features and language features. The visual input **X** of SDCAM is $\mathbf{X}_t^1$ or $\mathbf{X}_t^2$ and the language input is **Y**, where $\mathbf{X} = [\mathbf{x}_1, \mathbf{x}_2, \cdots, \mathbf{x}_m]$, $\mathbf{Y} = [\mathbf{y}_1, \mathbf{y}_2, \cdots, \mathbf{y}_k]$. In SDCAM, $\mathbf{x}_i (i = 1, \cdots, m)$ denotes the image feature of the i-th bounding box and $\mathbf{y}_i (i = 1, \cdots, k)$ denotes the language feature of the i-th word, and they are considered as the underlying capsules which need to be routed.

First, the high-level capsule $\mathbf{S}_t$ of the visual attention module is initialized by the fusion feature $\mathbf{F}_{t-1}$. When $t = 1$, $\mathbf{F}_0$ is a language feature processed by the reduction model, which is a simple self-attention module proposed in Yu et al. (2019). The high-level capsule for the visual attention module is calculated by:

$$\mathbf{S}_t^{p+1} = \mathbf{S}_t^p + \sum_{i=1}^{m} c_i \mathbf{x}_i^w, \tag{1}$$

$$\mathbf{x}_i^w = \sigma(\mathbf{W}_x \mathbf{x}_i), i \in [1, m], \tag{2}$$

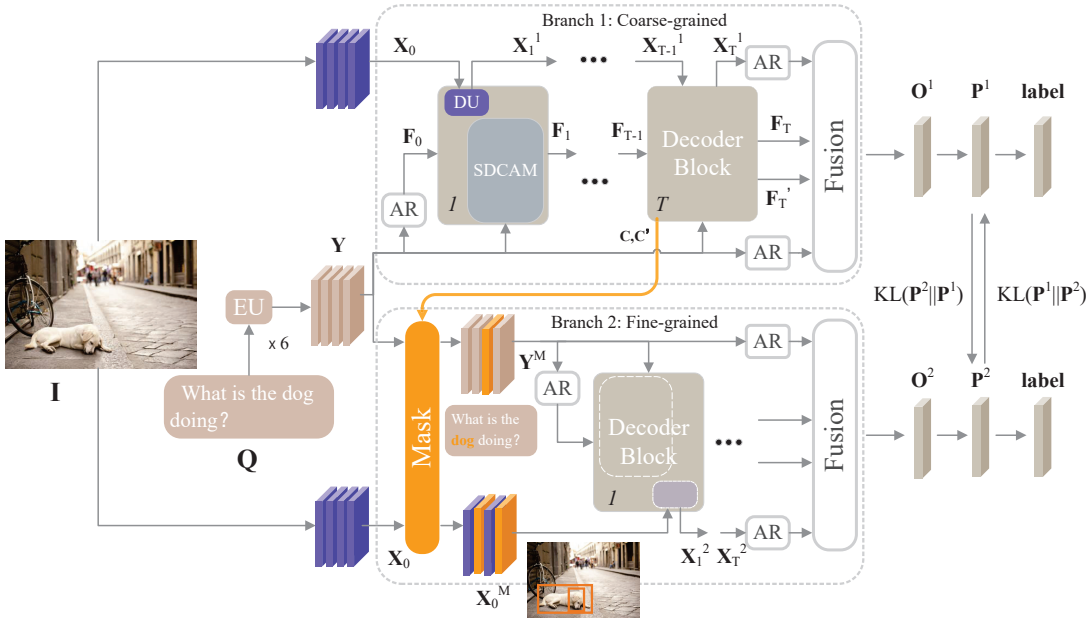

Figure 2: The overall framework of DCAMN. Coarse-grained branch 1 and fine-grained branch 2 handle coarse-grained features and fine-grained features, respectively. The fine-grained features are obtained by masking with attention weights provided by the last SDCAM of branch 1. AR, EU, and DU indicate reduction model, encoder unit, and decoder unit, respectively. We combine the two predictions $\mathbf{P}^1$ and $\mathbf{P}^2$ to get the final prediction.

where $\mathbf{S}_t^p$ denotes the high-level capsule of the visual attention module in the t-th SDCAM unit after $p \in [0, R-1]$ routing iterations, R represents the total number of routing iterations, c_i is the coupling coefficient between the underlying capsule $\mathbf{x}_i$ and high-level capsule $\mathbf{S}_t$, which can be interpreted as the contribution to the high-level capsule, $\mathbf{x}_i$ is multiplied by the projection matrix $\mathbf{W}_x \in \mathbb{R}^{d \times d}$ and passed through the activation function σ to get $\mathbf{x}_i^w$.

The value of the coupling coefficient c_i depends on b_i. In the routing algorithm, the value of b_i is used to measure the similarity between the underlying capsule $\mathbf{x}_i$ and the high-level capsule $\mathbf{S}_t$, and is obtained by:

$$b_i = b_i + (\mathbf{x}_i^w)^{\mathsf{T}} \cdot \mathbf{S}_t^p, i \in [1, m]. \quad (3)$$

We initialize b_i to be 0. In each routing iteration, b_i is added with the dot product of the high-level capsule and the underlying capsule for updating, and then softmax is applied to obtain the coupling coefficient c_i as follows:

$$c_i = softmax(b_i), i \in [1, m]. \quad (4)$$

When the processing of the visual attention module finishes, the fusion feature $\mathbf{S}_t$ will be used to initialize the high-level capsule $\mathbf{S}_t^{'}$ of the language attention module. The language attention module is similar to the visual one, except that the input of the language attention module is $\mathbf{Y} = [\mathbf{y}_1, \mathbf{y}_2, \cdots, \mathbf{y}_k]$.

After the language attention module, $\mathbf{S}_t^{'}$, which is also denoted as the fusion feature output $\mathbf{F}_t$ of the SDCAM, is fed to the next SDCAM for initializing its high-level feature $\mathbf{S}_{t+1}$. The whole procedure is defined as:

$$\mathbf{F}_t, \mathbf{F}_t^{'} = \mathrm{SDCAM}_t(\mathbf{X}, \mathbf{Y}, \mathbf{F}_{t-1}), t \in [1, T]. \quad (5)$$

In decoder block t, visual features and language features are fed to a DU and a SDCAM for processing:

$$\mathbf{X}_t = \mathrm{DU}_t(\mathbf{X}_{t-1}, \mathbf{Y}), \quad (6)$$

$$\mathbf{F}_t, \mathbf{F}_t^{'} = \mathrm{SDCAM}_t(\mathbf{X}_t, \mathbf{Y}, \mathbf{F}_{t-1}), t \in [1, T], \quad (7)$$

where DU_t is the t-th decoder unit and denotes a Guided-Attention Unit (Yu et al., 2019). The DU_t takes $\mathbf{X}_{t-1}$ and $\mathbf{Y}$ as inputs. Its output $\mathbf{X}_t$ is fed to SDCAM_t and the next decoder block. The whole procedure of decoder block t is defined as:

$$\mathbf{F}_t, \mathbf{F}_t^{'}, \mathbf{c}, \mathbf{c}^{'}, \mathbf{X}_t = \mathrm{DecoderBlock}_t(\mathbf{X}_{t-1}, \mathbf{Y}, \mathbf{F}_{t-1}), t \in [1, T], \quad (8)$$

where $\mathbf{c}$ and $\mathbf{c}^{'}$ are coupling coefficients generated from the SDCAM, which can be used to label the weights of bounding boxes and question words, respectively. In this way, we can learn what the model concerns in the inference process.

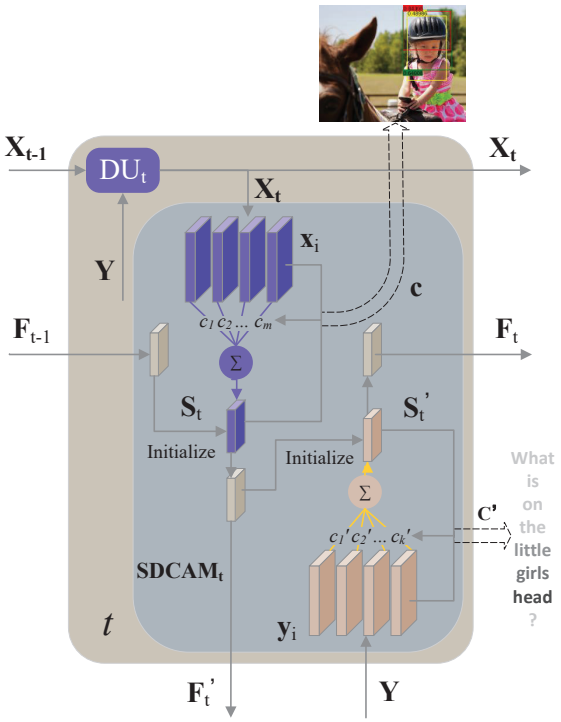

Figure 3: Illustration of decoder block t. It consists of a SDCAM and a DU. The attention map of the image is obtained by labeling bounding boxes with the weight $\mathbf{c}$ from SDCAM, and the importance of each word in the question is reflected by the weight $\mathbf{c}'$.

3.3 Double Branches with Coarse and Fine Granularities

In coarse-grained branch 1, coarse-grained visual feature $\mathbf{X}_0$ and language feature $\mathbf{Y}$ are sent to T decoder blocks to obtain $\mathbf{F}_T$, $\mathbf{F}_T'$ and $\mathbf{X}_T^1$. Note that we let all SDCAMs in decoder blocks of branch 1 share parameters to reduce the parameter burden. After this, $\mathbf{X}_T^1$ and $\mathbf{Y}$ pass through the attentional reduction model and are summed. The result is then concatenated with the sum of $\mathbf{F}_T$ and $\mathbf{F}_T'$ to get the fused feature $\mathbf{U}^1 \in \mathbb{R}^{d_f}$, where d_f is the dimension of the fused feature. Afterward, $\mathbf{U}^1$ is passed through a linear layer to get the vector $\mathbf{O}^1 \in \mathbb{R}^N$, where N indicates the number of candidate answers.

The fine-grained visual features and language features are processed in fine-grained branch 2 which has independent decoder blocks. To obtain critical features with visual-language relevance, we mask $\mathbf{X}_0$ and $\mathbf{Y}$ with the attention weights from the last SDCAM of coarse-grained branch 1. Specifically, in the visual attention module of SDCAM, the coupling coefficient $c_i (i = [1, 2, \cdots, m])$ corresponds to the visual feature $\mathbf{x}_i \in \mathbf{X}_0 (i = [1, 2, \cdots, m])$. We sort $\mathbf{x}_i$ from large to small according to the coupling coefficient c_i. The bottom $p\%$ of $\mathbf{x}_i$ are selected to be multiplied by the mask coefficient $\gamma (< 1)$ while the rest remain unchanged. The language feature $\mathbf{Y}$ is processed in the same way. Mask enables the network to focus on critical information. For example, in Figure 2, the word "dog" in the question and the bounding boxes about the dog are more significant after masking.

In fine-grained branch 2, the masked features $\mathbf{X}_0^M$ and $\mathbf{Y}^M$ are processed in the same way as branch 1. They are sent to T decoder blocks and fused to obtain $\mathbf{U}^2 \in \mathbb{R}^{d_f}$ and the vector $\mathbf{O}^2 \in \mathbb{R}^N$.

We apply a softmax function on $\mathbf{O}^1$ and $\mathbf{O}^2$ to obtain predictions of branch 1 and branch 2:

$$\mathbf{P}^i = softmax(\mathbf{O}^i), i \in [1, 2], \tag{9}$$

where $\mathbf{P}^1$ and $\mathbf{P}^2$ are predictions of branch 1 and branch 2, respectively. We sum $\mathbf{P}^1$ and $\mathbf{P}^2$ and take the candidate answer that has the maximum probability in their results as the final answer a.

3.4 Mutual Learning and Loss Function

For each branch, we utilize the prediction from the other branch to supervise. The Kullback Leibler (KL) Divergence is used to calculate the loss of mutual learning:

$$L_M = D_{kl}(\mathbf{P}^2 \| \mathbf{P}^1) + D_{kl}(\mathbf{P}^1 \| \mathbf{P}^2). \tag{10}$$

L_M forces the two branches to match the probability distribution of each other and learn from each other. In addition to L_M, binary cross-entropy (BCE) is employed to compute the loss between predictions and labels:

$$\begin{aligned} L_{BCE} = & BCE(sigmoid(\mathbf{O}^1), \mathbf{label}) \\ & + BCE(sigmoid(\mathbf{O}^2), \mathbf{label}). \end{aligned} \tag{11}$$

The two branches independently compute their BCE losses. The total loss is the sum of L_M and L_{BCE}.

In contrast to the tedious training method of traditional mutual learning (Zhang et al., 2018) that performs asynchronous updates among different sub-networks, DCAMN only has a total loss L, which means that parameter optimization in different branches can be performed simultaneously and training efficiency can be improved.

4 Experiments

4.1 Datasets

We use the VQA-v2 dataset (Goyal et al., 2017) to evaluate our model. The dataset is split into three

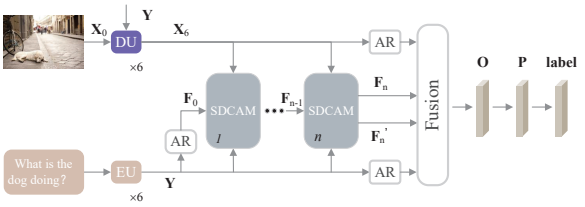

Figure 4: SDCAM-last-layer-n.

subsets: *train*, *val*, and *test*, which contain 80k, 40k, and 80k images and 444k, 214k, and 448k QA pairs, respectively. It has three types of questions: *yes/no*, *number*, and *other*. *test-std* and *test-dev*, as the subsets of the *test* set, are provided to evaluate model performance online. In addition, we also use the VQA-CPv2 dataset (Agrawal et al., 2018) to test models' robustness for the question biases. The VQA-CPv2 dataset is a VQA dataset with a particular answer distribution used to evaluate the ability of the model against language priors.

4.2 Implementation Details

The hyperparameter setting of our method is described in this subsection. We set the hidden layer dimension of LSTM to be 512. The dimension d_f of the fused feature is 1024. The latent dimensionality d in multi-head attention is set to be 512. Features in the multi-head attention are split into 8 heads with 64 dimensions for each head. We set the number of candidate answers N and the number of decoding layers T to 3129 and 6. Following Sabour et al. (2017), the number of routing iterations R is set to 3. We set mask coefficient γ and mask probability p to 0.5 and 30, which are experimentally selected.

We set the number of epochs to 15. The batch size is 64. The learning rate is initialized to $2.5e^{-5}$, gradually grows to $1e^{-4}$, and is decayed by 0.2 in the last three epochs. We use the adam optimizer and set the parameters β_1 and β_2 to be 0.9 and 0.98, respectively. We use the *train* and *val* sets for training and a subset of Visual Genome (Krishna et al., 2017) for data augmentation, and test for online evaluation on *test-dev* and *test-std*.

4.3 Ablation Studies

We use MCAN (Yu et al., 2019) and TRAR (Zhou et al., 2021) as the baselines. SDCAM-last-layer-n represents that the final output of encoder-decoder is processed by n SDCAMs in a single branch, as illustrated in Figure 4. Table 1 shows that SDCAM-last outperforms the baseline MCAN, which validates the effectiveness of SDCAM as a novel fusion method. SDCAM-mid indicates that the intermediate features from decoder units are processed by SDCAM, which is similar to coarse-grained branch 1 of Figure 2. SDCAM-mid outperforms SDCAM-last, which demonstrates the effectiveness of reusing intermediate information. Besides improving VQA performance, SDCAM has a further contribution to enhancing the interpretability of DCAMN. From the result of Table 1 and Table 2, we can see that each component (mutual learning, SDCAM, and the masking strategy) contributes to performance improvement. Moreover, The mutual learning in DCAMN will degenerate into the modules ensemble if the additional loss L_M is removed. By comparing the performance of DCAMN with or without KLloss, we can determine that DCAMN is not just a model simply using the modules ensemble because it enables branches to learn from each other to improve effectiveness.

Figure 5(a) shows that the VQA accuracy of SDCAM-last and SDCAM-mid grows and saturates as the number of stacks increases. This proves that stacking can improve the performance of SDCAM and enable it to perform a better fusion.

Model	All	Other	Yes/No	Num.
MCAN(baseline)	67.2	58.7	84.84	48.69
SDCAM-last-layer-3	67.34	58.81	85.15	48.42
SDCAM-mid-layer-6	67.43	58.89	85.15	48.78
SDCAM-mid+Mutual	68.05	59.63	85.60	49.43
SDCAM-mid+Mutua+Mask	**68.14**	**59.66**	**85.74**	**49.65**

Table 1: Ablation studies using MCAN as the backbone on VQA-v2 *val* set. Mutual means using the two-branch mutual learning strategy. Mask means using our masking mechanism.

SA	TD	SDCAM	Mutual	Mask	KLloss	Accuracy
						67.6
✓						67.48
	✓					67.55
		✓				67.76
			✓		✓	68.27
		✓	✓		✓	68.30
✓			✓	✓	✓	68.09
	✓		✓	✓	✓	68.19
		✓	✓	✓		68.00
		✓	✓	✓	✓	68.40

Table 2: Ablation studies using TRAR as the backbone on VQA-v2 *val* set. SA and TD mean using the Self-Attention in MCAN and the Top-Down mechanism in BUTD for co-attention to replace SDCAM, respectively. KLloss means the loss L_M.

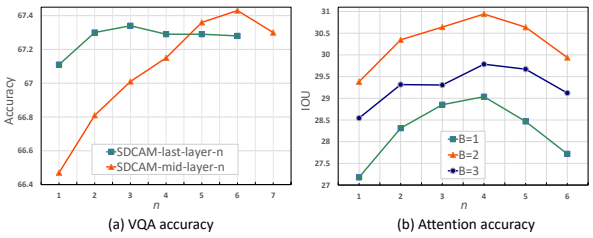

(a) VQA accuracy (b) Attention accuracy

Figure 5: Effect of SDCAM stacking numbers n on VQA accuracy and attention accuracy. layer-n indicates that the SDCAM is stacked n times. The attention weights used for the attention accuracy calculation are provided by the n-th SDCAM of SDCAM-mid-layer-n.

Method	B=1	B=2	B=3	B=AU
DFAF (2019)	5.07	11.27	14.18	17.55
MCAN (2019)	15.43	18.77	20.01	23.91
BUTD (2018)	26.46	27.80	26.92	33.97
SDCAM-mid-layer-6	27.72	29.94	29.12	37.15
DCAMN-branch-1	28.05	30.26	29.21	37.38
DCAMN-branch-2	28.29	30.25	28.98	37.18
DCAMN-branch-sum	28.78	30.68	29.45	37.80
SDCAM-mid-layer-4	**29.04**	**30.94**	**29.79**	**37.93**

Table 3: Attention accuracy comparison of various methods. B is the number of candidate bounding boxes. DCAMN-branch-b denotes the attention accuracy of the last SDCAM in branch b of DCAMN. branch-sum means attention weights of two branches are added together. We evaluate methods on the VQS validation set.

4.4 Qualitative Analysis

To explore the inference process of the model and determine whether the correct answer is obtained by reasoning based on features rather than exploiting the statistics of the dataset, we visualize the attention weights of SDCAM in Figure 6.

From the first row, we can see that SDCAM achieves more accurate attention both on visual regions and on question words. In the first question, SDCAM locates the "drainage" in the image accurately and gets the correct answer. By analyzing examples of MCAN and BUTD, we can see that the models usually get the wrong answers when they focus on the incorrect visual regions and words. This demonstrates the ability of SDCAM to find evidence and help analyze the reasoning process.

Moreover, from the second row, we can observe that as the number of stacked layers increases, the interest regions of SDCAM tend to be more accurate, and the distribution of weights tends to be more focused. In the question "What is being celebrated?" SDCAM focuses on the people at first, but after several iterations, the center of attention shifts to the cake, and the model finally gets the correct answer. The next question is similar to this one. These cases prove that SDCAM can focus on different objects exactly as reasoning requires.

In the third row, we show some examples of wrong predictions. With the attention map of SDCAM, we can know the reason for wrong predictions. For example, in the question "Are there any boxes in the room?" SDCAM focuses on the printer that looks like a box and incorrectly answers yes.

4.5 Analysis of the Attention Accuracy

We quantify the accuracy of visual attention of SDCAM and other methods, as shown in Table 3. VQS is a dataset in which for each triad (image, question, and answer), there is a corresponding image segmentation mask depicting the contours which need attention in the image (Gan et al., 2017). We calculate the Intersection of Union (IOU) between the VQS segmentation mask and the attention mask of DCAMN as the metric to evaluate the attention accuracy, where the attention mask of SDCAM is the union of top B bounding boxes in terms of the attention weights rank. B=AU (automatically) indicates that the optimal result for different bounding box numbers is incorporated into the overall IOU statistics. From Table 3, we can see that SDCAM has higher attention accuracy than other methods. The results of two-branches DCAMN are not better than those of single branch SDCAM, which indicates that SDCAM attention accuracy is hardly influenced by mutual learning.

In Figure 5(b), we show the performance of SDCAM-mid with different stacking numbers n. With increasing n, the attention accuracy becomes higher and reaches a maximum at $n = 4$. This confirms that the stacking strategy can help SDCAM focus on the correct regions.

4.6 Comparison with SOTAs

We compare our DCAMN [1] with the state-of-the-art methods on the VQA-v2 dataset, and the results are shown in Table 4. We can see that DCAMN outperforms other methods and $\text{DCAMN}_{\text{mcan}}$ and $\text{DCAMN}_{\text{trar}}$ have higher performance than the baseline MCAN and TRAR, respectively, which proves the validity of the proposed method. It is worth noting that for the baseline TRAR, the improvement on the number type question is particu-

[1]Code is available at https://github.com/HFUTLHD/DCAMN-VQA-master.

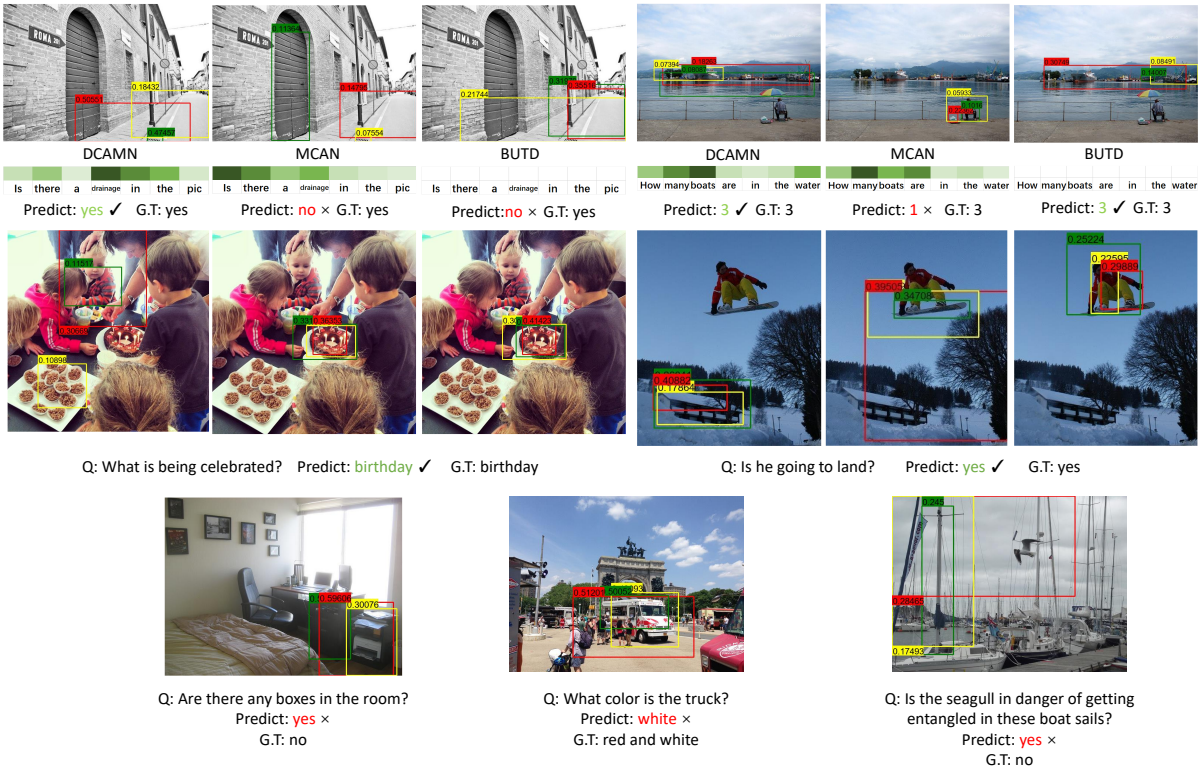

Figure 6: Visualization of attention weights for DCAMN, MCAN and BUTD on visual regions and question words. We mark the top three bounding boxes based on attention weights rank and label their weights above. The second row shows the visual attention distribution of the image in the 1st, 3rd, and 6th SDCAM of DCAMN. The third row shows some incorrect samples.

	Test-dev				Test-std
Method	All	Yes/no	Num.	Others	All
BUTD (2018)	65.32	81.82	44.21	56.05	65.67
DFAF (2019)	70.22	86.09	53.32	60.49	70.34
CFR (2021)	72.5	-	-	-	-
MCAN (2019)	70.63	86.82	53.26	60.72	70.90
TRAR (2021)	72.62	88.11	55.33	63.31	72.93
$\text{DCAMN}_{\text{mcan}}$	71.77	87.80	53.96	62.12	72.19
$\text{DCAMN}_{\text{trar}}$*	**73.67**	**88.86**	**58.02**	**64.22**	**74.05**

Table 4: VQA performance comparison with state-of-the-art approaches on the VQA-v2 dataset. $\text{DCAMN}_{\text{mcan}}$ and $\text{DCAMN}_{\text{trar}}$ denote DCAMNs using MCAN and TRAR as the backbone, respectively. * means using the 16 × 16 grid features.

larly significant, with a 2.7 point gain. We attribute such a marked improvement to the fact that the objects which require counting are more prominent after masking, which facilitates more accurate modeling in the second branch. From the results in Table 5, we can see that DCAMN outperforms the baselines and other methods on the VQA-CPv2 dataset, which proves the effectiveness of DCAMN against the language priors.

Method	Accuracy
BUTD (2018)	39.06
QCG (2018)	39.32
BAN (2018)	40.06
MCAN (2019)	43.29
TRAR (2021)	42.30
$\text{DCAMN}_{\text{trar}}$	43.03
$\text{DCAMN}_{\text{mcan}}$	**44.09**

Table 5: VQA performance comparison with other methods on VQA-CPv2 test.

5 Conclusion

In this paper, we propose a dual capsule attention mask network for VQA. DCAMN can process features at different granularities to take global information into account and focus on critical information. Combining the views of the two branches at different perspectives and granularities can improve the generalization capability of the model and make more accurate predictions. In addition, the proposed SDCAM can effectively fuse multimodal features and locate evidence, which also enhances the interpretation capability of the network. Experiments show that DCAMN outperforms other methods in terms of interpretability and accuracy and achieves new SOTA performance for VQA.

Acknowledgements

This work was supported by the National Natural Science Foundation of China under Grant 61976079, the Open Project of the Key Laboratory of Digital Publishing and Knowledge Service of Cultural Resources of the State Press and Publication Administration of China under Grant 104-4331201902, the Guangxi Key Research and Development Program under Grant AB22035022, and Anhui Key Research and Development Program under Grant 202004a05020039.

References

Aishwarya Agrawal, Dhruv Batra, Devi Parikh, and Aniruddha Kembhavi. 2018. Don't just assume; look and answer: Overcoming priors for visual question answering. In *2018 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2018, Salt Lake City, UT, USA, June 18-22, 2018*, pages 4971–4980. Computer Vision Foundation / IEEE Computer Society.

Peter Anderson, Xiaodong He, Chris Buehler, Damien Teney, Mark Johnson, Stephen Gould, and Lei Zhang. 2018. Bottom-up and top-down attention for image captioning and visual question answering. In *2018 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2018, Salt Lake City, UT, USA, June 18-22, 2018*, pages 6077–6086. Computer Vision Foundation / IEEE Computer Society.

Stanislaw Antol, Aishwarya Agrawal, Jiasen Lu, Margaret Mitchell, Dhruv Batra, C. Lawrence Zitnick, and Devi Parikh. 2015. VQA: visual question answering. In *2015 IEEE International Conference on Computer Vision, ICCV 2015, Santiago, Chile, December 7-13, 2015*, pages 2425–2433. IEEE Computer Society.

Hedi Ben-younes, Rémi Cadène, Matthieu Cord, and Nicolas Thome. 2017. MUTAN: multimodal tucker fusion for visual question answering. In *IEEE International Conference on Computer Vision, ICCV 2017, Venice, Italy, October 22-29, 2017*, pages 2631–2639. IEEE Computer Society.

Jiahui Cai and Jianguo Hu. 2020. 3d rans: 3d residual attention networks for action recognition. *Vis. Comput.*, 36(6):1261–1270.

Yen-Chun Chen, Linjie Li, Licheng Yu, Ahmed El Kholy, Faisal Ahmed, Zhe Gan, Yu Cheng, and Jingjing Liu. 2020. UNITER: universal image-text representation learning. In *Computer Vision - ECCV 2020 - 16th European Conference, Glasgow, UK, August 23-28, 2020, Proceedings*, volume 12375 of *Lecture Notes in Computer Science*, pages 104–120. Springer.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, Minneapolis, Minnesota. Association for Computational Linguistics.

Kevin Duarte, Yogesh S. Rawat, and Mubarak Shah. 2018. Videocapsulenet: A simplified network for action detection. In *Advances in Neural Information Processing Systems 31: Annual Conference on Neural Information Processing Systems 2018, NeurIPS 2018, December 3-8, 2018, Montréal, Canada*, pages 7621–7630.

Akira Fukui, Dong Huk Park, Daylen Yang, Anna Rohrbach, Trevor Darrell, and Marcus Rohrbach. 2016. Multimodal compact bilinear pooling for visual question answering and visual grounding. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 457–468, Austin, Texas. Association for Computational Linguistics.

Chuang Gan, Yandong Li, Haoxiang Li, Chen Sun, and Boqing Gong. 2017. VQS: linking segmentations to questions and answers for supervised attention in VQA and question-focused semantic segmentation. In *IEEE International Conference on Computer Vision, ICCV 2017, Venice, Italy, October 22-29, 2017*, pages 1829–1838. IEEE Computer Society.

Haoyuan Gao, Junhua Mao, Jie Zhou, Zhiheng Huang, Lei Wang, and Wei Xu. 2015. Are you talking to a machine? dataset and methods for multilingual image question. In *Advances in neural information processing systems*, pages 2296–2304.

Peng Gao, Zhengkai Jiang, Haoxuan You, Pan Lu, Steven C. H. Hoi, Xiaogang Wang, and Hongsheng Li. 2019. Dynamic fusion with intra- and inter-modality attention flow for visual question answering. In *IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2019, Long Beach, CA, USA, June 16-20, 2019*, pages 6639–6648. Computer Vision Foundation / IEEE.

Yash Goyal, Tejas Khot, Douglas Summers-Stay, Dhruv Batra, and Devi Parikh. 2017. Making the V in VQA matter: Elevating the role of image understanding in visual question answering. In *2017 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2017, Honolulu, HI, USA, July 21-26, 2017*, pages 6325–6334. IEEE Computer Society.

Geoffrey E. Hinton, Alex Krizhevsky, and Sida D. Wang. 2011. Transforming auto-encoders. In *Artificial Neural Networks and Machine Learning - ICANN 2011 - 21st International Conference on Artificial Neural Networks, Espoo, Finland, June 14-17, 2011, Proceedings, Part I*, volume 6791 of *Lecture Notes in Computer Science*, pages 44–51. Springer.

Geoffrey E. Hinton, Sara Sabour, and Nicholas Frosst. 2018. Matrix capsules with EM routing. In *6th International Conference on Learning Representations, ICLR 2018, Vancouver, BC, Canada, April 30 - May 3, 2018, Conference Track Proceedings*. OpenReview.net.

Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long short-term memory. *Neural Comput.*, 9(8):1735–1780.

Ayush Jaiswal, Wael AbdAlmageed, Yue Wu, and Premkumar Natarajan. 2018. Capsulegan: Generative adversarial capsule network. In *Computer Vision - ECCV 2018 Workshops - Munich, Germany, September 8-14, 2018, Proceedings, Part III*, volume 11131 of *Lecture Notes in Computer Science*, pages 526–535. Springer.

Huaizu Jiang, Ishan Misra, Marcus Rohrbach, Erik G. Learned-Miller, and Xinlei Chen. 2020. In defense of grid features for visual question answering. In *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition, CVPR 2020, Seattle, WA, USA, June 13-19, 2020*, pages 10264–10273. Computer Vision Foundation / IEEE.

Jin-Hwa Kim, Jaehyun Jun, and Byoung-Tak Zhang. 2018. Bilinear attention networks. In *Advances in Neural Information Processing Systems 31: Annual Conference on Neural Information Processing Systems 2018, NeurIPS 2018, December 3-8, 2018, Montréal, Canada*, pages 1571–1581.

Jin-Hwa Kim, Kyoung Woon On, Woosang Lim, Jeonghee Kim, Jung-Woo Ha, and Byoung-Tak Zhang. 2017. Hadamard product for low-rank bilinear pooling. In *5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Conference Track Proceedings*. OpenReview.net.

Ranjay Krishna, Yuke Zhu, Oliver Groth, Justin Johnson, Kenji Hata, Joshua Kravitz, Stephanie Chen, Yannis Kalantidis, Li-Jia Li, David A. Shamma, Michael S. Bernstein, and Li Fei-Fei. 2017. Visual genome: Connecting language and vision using crowdsourced dense image annotations. *Int. J. Comput. Vis.*, 123(1):32–73.

Xiujun Li, Xi Yin, Chunyuan Li, Pengchuan Zhang, Xiaowei Hu, Lei Zhang, Lijuan Wang, Houdong Hu, Li Dong, Furu Wei, Yejin Choi, and Jianfeng Gao. 2020. Oscar: Object-semantics aligned pre-training for vision-language tasks. In *Computer Vision - ECCV 2020 - 16th European Conference, Glasgow, UK, August 23-28, 2020, Proceedings, Part XXX*, volume 12375 of *Lecture Notes in Computer Science*, pages 121–137. Springer.

Jiasen Lu, Jianwei Yang, Dhruv Batra, and Devi Parikh. 2016. Hierarchical question-image co-attention for visual question answering. In *Advances in neural information processing systems*, pages 289–297.

Mateusz Malinowski, Marcus Rohrbach, and Mario Fritz. 2015. Ask your neurons: A neural-based approach to answering questions about images. In *2015 IEEE International Conference on Computer Vision, ICCV 2015, Santiago, Chile, December 7-13, 2015*, pages 1–9. IEEE Computer Society.

Binh X. Nguyen, Tuong Do, Huy Tran, Erman Tjiputra, Quang D. Tran, and Anh Nguyen. 2021. Coarse-to-fine reasoning for visual question answering. *CoRR*, abs/2110.02526.

Will Norcliffe-Brown, Stathis Vafeias, and Sarah Parisot. 2018. Learning conditioned graph structures for interpretable visual question answering. In *Advances in Neural Information Processing Systems 31: Annual Conference on Neural Information Processing Systems 2018, NeurIPS 2018, December 3-8, 2018, Montréal, Canada*, pages 8344–8353.

Jeffrey Pennington, Richard Socher, and Christopher Manning. 2014. GloVe: Global vectors for word representation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1532–1543, Doha, Qatar. Association for Computational Linguistics.

Mengye Ren, Ryan Kiros, and Richard S. Zemel. 2015. Exploring models and data for image question answering. In *Advances in Neural Information Processing Systems 28: Annual Conference on Neural Information Processing Systems 2015, December 7-12, 2015, Montreal, Quebec, Canada*, pages 2953–2961.

Adriana Romero, Nicolas Ballas, Samira Ebrahimi Kahou, Antoine Chassang, Carlo Gatta, and Yoshua Bengio. 2015. Fitnets: Hints for thin deep nets. In *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*.

Sara Sabour, Nicholas Frosst, and Geoffrey E. Hinton. 2017. Dynamic routing between capsules. In *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, December 4-9, 2017, Long Beach, CA, USA*, pages 3856–3866.

Himanshu Sharma and Anand Singh Jalal. 2022. An improved attention and hybrid optimization technique for visual question answering. *Neural Process. Lett.*, 54(1):709–730.

Guocong Song and Wei Chai. 2018. Collaborative learning for deep neural networks. In *Advances in Neural Information Processing Systems 31: Annual Conference on Neural Information Processing Systems 2018, NeurIPS 2018, December 3-8, 2018, Montréal, Canada*, pages 1837–1846.

Zhou Yu, Jun Yu, Yuhao Cui, Dacheng Tao, and Qi Tian. 2019. Deep modular co-attention networks for visual question answering. In *IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2019, Long Beach, CA, USA, June 16-20, 2019*, pages 6281–6290. Computer Vision Foundation / IEEE.

Linfeng Zhang and Kaisheng Ma. 2021. Improve object detection with feature-based knowledge distillation: Towards accurate and efficient detectors. In *9th International Conference on Learning Representations, ICLR 2021, Virtual Event, Austria, May 3-7, 2021*. OpenReview.net.

Ying Zhang, Tao Xiang, Timothy M. Hospedales, and Huchuan Lu. 2018. Deep mutual learning. In *2018 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2018, Salt Lake City, UT, USA, June 18-22, 2018*, pages 4320–4328. Computer Vision Foundation / IEEE Computer Society.

Yongheng Zhao, Tolga Birdal, Haowen Deng, and Federico Tombari. 2019. 3d point capsule networks. In *IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2019, Long Beach, CA, USA, June 16-20, 2019*, pages 1009–1018. Computer Vision Foundation / IEEE.

Yiyi Zhou, Rongrong Ji, Jinsong Su, Xiaoshuai Sun, and Weiqiu Chen. 2019. Dynamic capsule attention for visual question answering. In *Proceedings of the AAAI conference on artificial intelligence*, pages 9324–9331. AAAI Press.

Yiyi Zhou, Tianhe Ren, Chaoyang Zhu, Xiaoshuai Sun, Jianzhuang Liu, Xinghao Ding, Mingliang Xu, and Rongrong Ji. 2021. TRAR: routing the attention spans in transformer for visual question answering. In *2021 IEEE/CVF International Conference on Computer Vision, ICCV 2021, Montreal, QC, Canada, October 10-17, 2021*, pages 2054–2064. IEEE.

Emergence of hierarchical reference systems in multi-agent communication

Xenia Ohmer and **Marko Duda** and **Elia Bruni**
Osnabrück University
{xenia.ohmer,mduda,elia.bruni}@uos.de

Abstract

In natural language, referencing objects at different levels of specificity is a fundamental pragmatic mechanism for efficient communication in context. We develop a novel communication game, the *hierarchical reference game*, to study the emergence of such reference systems in artificial agents. We consider a simplified world, in which concepts are abstractions over a set of primitive attributes (e.g., color, style, shape). Depending on how many attributes are combined, concepts are more general ("circle") or more specific ("red dotted circle"). Based on the context, the agents have to communicate at different levels of this hierarchy. Our results show that the agents learn to play the game successfully and can even generalize to novel concepts. To achieve abstraction, they use implicit (omitting irrelevant information) and explicit (indicating that attributes are irrelevant) strategies. In addition, the compositional structure underlying the concept hierarchy is reflected in the emergent protocols, indicating that the need to develop hierarchical reference systems supports the emergence of compositionality.

1 Introduction

Humans excel at using language to convey information efficiently in context. A speaker does not have to communicate every detail. Rather, a listener can infer the intended meaning of an utterance by assuming that sufficient information was provided. This idea was first explicitly formulated by Grice (1975) in his conversational maxims, in particular the *Maxim of Quantity*: "1. Make your contribution as informative as is required (for the current purposes of the exchange). 2. Do not make your contribution more informative than is required." An illustration of this mechanism can be given in the form of a simple referential context. In a scene with a red circle and a green triangle, "circle" is enough information to identify the referent, whereas more complex scenes may require the speaker to name both object attributes—shape and color—to allow for an unambiguous interpretation. The Maxim of Quantity requires a hierarchical reference system, that allows the selection of the most appropriate level of specificity for a given context.

In this paper, we follow the proposal by Higgins et al. (2018) and define concepts as compositional abstractions over a set of primitive attributes (e.g., color, style, shape), see Figure 1a. The concepts are maximally specific at the leaf nodes, where all attribute values are determined. Moving from the subordinate levels up to the superordinate levels, the number of concept-defining attribute values decreases. Thus, each parent concept is an abstraction (i.e. a subset) over its children and over the original set of attribute values. Given this definition of a concept hierarchy, we use a language emergence paradigm with artificial agents to study whether a corresponding reference system can emerge given a structured perception of the world.

In most language emergence simulations, a sender and a receiver agent are trained on a reference game (e.g., Havrylov and Titov, 2017; Lazaridou et al., 2018; Rodríguez Luna et al., 2020; Dagan et al., 2021), based on the signaling game originally developed by Lewis (1969). The sender sees a target object and sends a message to the receiver. Using that message, the receiver tries to identify the target among a set of distractor objects. Crucially, in the current form, reference games completely ignore that different contexts may require referential expressions at different levels of abstraction. Having no access to the distractors, the sender cannot choose relevant object attributes in a context-dependent way. Moreover, random sampling of the distractors typically encourages the sender to communicate all object attributes. Therefore, the standard reference game cannot account for the emergence of hierarchical concepts in communication.

Proceedings of the 29th International Conference on Computational Linguistics, pages 5689–5706
October 12–17, 2022.

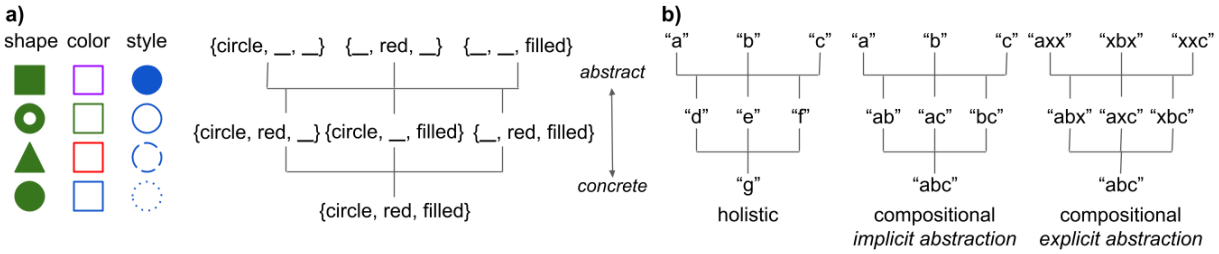

Figure 1: **a)** Example of a concept hierarchy. Shown are all attribute values and the concept hierarchy constructed from the concept "red filled circle". **b)** Example languages for the concept hierarchy in part a). Possible abstraction strategies include holistic and compositional languages. In compositional languages, abstraction can further be indicated implicitly or explicitly.

We develop a *hierarchical reference game* to address this shortcoming. Instead of an object, the sender receives a *concept* as input. The concept is defined by an attribute vector (object) and a relevance vector (context). The relevance vector indicates for each attribute whether it is relevant in the current context or not. Based on the sender's message, the receiver must identify an object that instantiates the target concept among a set of distractors. The input concepts have a compositional and hierarchical structure. While the game is designed to encourage communication at different levels of abstraction, it does not regulate how this abstraction is realized; in particular, there is no explicit pressure on the emergent language to reflect the compositional input structure.

First, we evaluate if the agents can successfully play the game, i.e. communicate specific contextually relevant object attributes. Second, to measure whether the agents' strategies are systematic, we test whether they can generalize to novel concepts, and also whether they consistently use the same expressions for the same concepts at all levels of abstraction. Third, we investigate the emerging protocols to study the mechanisms by which systematic abstraction is achieved, see Figure 1b. In natural language, there is *holistic* abstraction as in "Dalmatian" $\subseteq$ "dog" $\subseteq$ "animal"; but also *compositional* abstraction as in "filled red circle" $\subseteq$ "red circle" $\subseteq$ "circle". Abstraction can further be *implicit*, by omitting irrelevant attributes, and *explicit*, by indicating that certain attributes are irrelevant (as in saying "a circle of any color"). We evaluate which, if any, of these abstraction strategies are used by the agents.

Our work makes several contributions. We develop the hierarchical reference game and show that it can be used to model the emergence of referential expressions at different levels of abstraction. We also provide novel metrics to examine how the agents achieve abstraction. Working with different data sets, i.e. different concept hierarchies, allows us to disentangle data set specific and general effects. Not least, our results suggest that communication about concept hierarchies supports the emergence of compositionality.

2 Related work

Referring expression generation (REG). There has been a long history of research on understanding how people generate referring expressions, dating back to Winograd (1972). The most influential work on REG in both the eighties (Appelt, 1985; Appelt and Kronfeld, 1987) and nineties (Reiter and Dale, 1992; Dale and Reiter, 1995) integrated the Gricean maxims into their systems. The latter developed the *iterative algorithm*, which was used and extended to model various aspects of REG (Krahmer and van Deemter, 2012). Like Dale and Reiter (1995), we define objects as sets of attribute-value pairs and consider the subset of referring expressions whose single purpose is to identify an object. However, our main focus is not to generate human-like referring expressions but rather to build artificial agents capable of hierarchical reference. Hence, we ignore many effects that play a role in human REG, for example basic level categories (Rosch and Mervis, 1975; Rosch et al., 1976).

By now, several large-scale data sets of referring expressions for complex real-world images have been collected and are used to train deep neural networks (DNNs) (e.g., Kazemzadeh et al., 2014; Mao et al., 2016; Yu et al., 2016; Luo and Shakhnarovich, 2017; Yu et al., 2018; Luo et al., 2020). The data sets are collected in a reference game setup: one participant has to refer to a target

entity in a given image, and the other participant has to identify the target. Models are often trained on both components, expression generation and comprehension (e.g. Mao et al., 2016; Luo and Shakhnarovich, 2017). Several REG models try to integrate deep learning with computational accounts of pragmatic reasoning (e.g., Monroe and Potts, 2015; Andreas and Klein, 2016; Le et al., 2022), such as the Rational Speech Act framework (Frank and Goodman, 2012). Our model also implements expression generation (sender) and comprehension (receiver) using DNNs but hard codes pragmatic inferences in the relevance vector. Most importantly, the agents are not trained on a labeled data set but develop their own referring expressions in a language emergence game.

Emergent multi-agent communication. Language emergence simulations are popular in evolutionary linguistics as well as AI research. In evolutionary linguistics, they are used to study the origins and evolution of human and animal communication (e.g., Cangelosi and Parisi, 2002; Kirby, 2002; Wagner et al., 2003). In AI research, they are used with the aim of building artificial agents capable of flexible and goal-directed language use, which arguably relies on grounding language in interaction (e.g., Steels, 2001, 2003; Lazaridou and Baroni, 2020).

Starting with Foerster et al. (2016) and Lazaridou et al. (2017), there has been an increasing interest in language emergence simulations with DNN agents (for a review, see Lazaridou and Baroni, 2020). These approaches stand in contrast to the currently dominant DNN models in NLP, which learn passively by being exposed to large amounts of text (Bisk et al., 2020). As discussed above, in many implementations, hierarchical reference systems cannot emerge because the sender does not have access to information about the context. Even in the rare cases where it does (e.g., Lazaridou et al., 2017; Dessi et al., 2021), the emergence of hierarchical reference has not yet been investigated.

3 General setup

3.1 Concept representation

We use symbolic, disentangled input representations. A concept is composed of an object vector and a relevance vector. Objects have n attributes and each attribute can take on k values.[1] The relevance vector $r \in \{0,1\}^n$ indicates which attributes are relevant (1) and which ones are irrelevant (0). E.g., if the sender's input is $(4,3,1)(1,0,0)$, the concept in question is $(4,\_,\_) := \{(4,x,y)|x,y \in \mathbb{N}, 1 \leq x,y \leq k\}$. Object $(4,3,1)$ could instantiate the attributes *shape*, *color*, and *style* with specific values such as *circle*, *red*, and *filled* (see Figure 1a). Relevance vector $(1,0,0)$ would then indicate that only the first attribute value, *circle*, is relevant and must be communicated.

3.2 Hierarchical reference game

Like the classical reference game, the hierarchical reference game is played by a sender, S, and a receiver, R. However, rather than communicating the input object as it is, the sender must abstract a concept from this object based on the relevance vector. One round of the hierarchical reference game proceeds as follows (see Figure 2):

1. An object, o, and a relevance vector, r, are sampled randomly and passed to S.
2. Based on this input, S generates a message, m. The message is a concatenation of symbols from vocabulary V, $s_i \in V$, and has maximal length L, such that $m = (s_i)_{i \leq L}$.
3. R receives the message m, as well as a set of objects containing one target, t, and several distractors. t has the same attribute values as the input object o for relevant attributes (as defined by r), while the values of irrelevant attributes are sampled randomly. The distractors are constructed by sampling object instances of concepts that would arise from o in combination with other relevance vectors than r.
4. Based on m, R selects one object among target and distractors.

By our choice of distractors, we simulate an environment in which the relevance vector matches pragmatic needs: the speaker tries to be as specific as necessary in a given context. To further discourage communication of irrelevant attributes, we choose distractors that are more abstract than the target concept but still similar, by replacing exactly one 1 (*relevant*) in the relevance vector with a 0 (*irrelevant*). Additional experiments, where we sample distractors with equal probability from all

[1] We present objects as n-hot encodings to the agents, such that each object $o \in \{0,1\}^{nk}$.

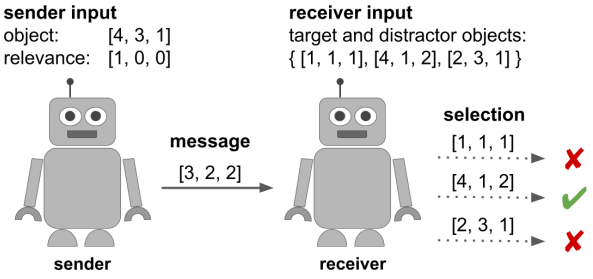

Figure 2: Schematic illustration of the hierarchical reference game.

levels of the concept hierarchy, can be found in Appendix A.[2]

3.3 Architecture

Both agents are implemented as single-layer GRUs. The sender input is processed by two dense layers, one receiving the object vector and one the relevance vector, followed by a dense layer mapping a concatenation of these two representations to the sender's initial hidden state. The message is produced incrementally. At each time step, the sender generates a probability distribution over the vocabulary which is used to sample a symbol from the Gumbel-softmax distribution (Jang et al., 2017). The GS distribution is a continuous distribution that approximates categorical samples, and whose parameter gradients can be easily computed via a reparameterization trick. The receiver processes the incoming message. An additional dense layer maps target and distractor objects onto embeddings. These embeddings have the same dimensionality as the receiver's hidden state. The receiver's selection probabilities are determined by applying a softmax function to the dot products between object embeddings and hidden state.

4 Experiments

Our implementation is based on PyTorch, and uses the EGG toolkit (Kharitonov et al., 2019).[3] Our code and results are available at https://github.com/XeniaOhmer/hierarchical_reference_game.

4.1 Data sets

In order to investigate how the number of attributes and the number of values per attribute influence the formation of abstract concepts, we use a set of different data sets, $D(n,k) := \{(m_1,\ldots,m_n) \mid m_i \in \mathbb{N}_k\}$ with $\mathbb{N}_k = \{1,...,k\}$ (see Table 1).

	$k=4$	$k=8$	$k=16$
$n=3$	$D(3,4)$	$D(3,8)$	$D(3,16)$
$n=4$	$D(4,4)$	$D(4,8)$	
$n=5$	$D(5,4)$		

Table 1: Input data sets with n attributes and k values. Data sets are labeled as $D(n,k)$.

We sample relevance vectors with equal probability from each level of the concept hierarchy.[4] We repeat that procedure until there are 10 samples for each input object and each number of relevant attributes in the data set. In addition, we create 10 distractors per sample. We reserve 20% of the data for zero-shot testing (see Section 4.3), and split the remaining data randomly into training and validation sets at a ratio of 0.75/0.25.

4.2 Hyperparameter selection and training

In our simulations there is always exactly one target object for the receiver, i.e. only that object—and none of the distractors—is an instance of the target concept. The agents minimize the cross-entropy loss between target and selection. During training, a message is given by the GS distributions across symbols, whereas during testing the argmax values are used. Hence, it is possible to jointly update the weights of sender and receiver by backpropagating through the approximated "discrete" messages.

We conducted a hyperparameter search to identify model and training parameters leading to high performance on the validation set for the range of different data sets we use (for details see Appendix B). Agents have an embedding layer with 128 units and a hidden layer with 256 units. The discrete messages are approximated using GS with an initial temperature of 1.5, decaying exponentially at a rate of 0.99. We train for 300 epochs using Adam optimizer with batch size 32 and learning rate 0.0005. For all data sets, we use the number of attributes as maximal message length L. The minimal vocabulary sizes in Table 2 allow the sender to generate a distinct message for each input concept. They correspond to the number of attribute values plus

[2] In that case, the agents still learn to play the game successfully and to form abstract concepts but they have a stronger tendency to convey also irrelevant information.

[3] https://github.com/facebookresearch/EGG

[4] If relevance vectors are sampled uniformly from the set of all relevance vectors, the amount of 0 and 1 entries follows a binomial distribution. Sampling relevance vectors with equal probability from each level of the hierarchy ensures that all abstraction levels occur equally often.

one additional symbol to indicate irrelevance. The agents have an additional end-of-sequence symbol to terminate the messages before L is reached. We run our experiments with a factor $f = 3$ of the minimal vocabulary size. Additional experiments with other values for f be found in Appendix A.[5] We conduct 5 runs per data set.

	$k = 4$	$k = 8$	$k = 16$
$n = 3$	5	9	17
$n = 4$	5	9	
$n = 5$	5		

Table 2: Minimal vocab size for each data set.

4.3 Evaluation

We are interested in different aspects of the emergent language, and use existing as well as novel metrics to evaluate these.

Zero-shot evaluation. We generate two different zero-shot test sets. The first test set is used to evaluate whether the agents can generalize to novel *objects*. It contains combinations of attribute values that do not occur in the training and validation data, and is reserved for testing after the data generation process. The second test set is used to evaluate whether the agents can generalize to novel *abstractions*. We withhold abstractions from one value per attribute from the training data. The agents are trained from scratch on the remaining data and evaluated on the held-out data.

Message consistency and effectiveness. To measure whether agents consistently use the same messages for the same concepts we employ information-theoretic metrics. Let C be the set of concepts, and M be the set of messages. The conditional entropy of messages given concepts,

$$\mathcal{H}(M \mid C) = -\sum_{c \in C,\, m \in M} p(c, m) \log \frac{p(c, m)}{p(c)},$$

measures how much uncertainty about the messages remains after knowing the concepts. Low values indicate that the agents consistently use the same messages for the same concepts, i.e. the language does not contain many synonymous expressions. $\mathcal{H}(C \mid M)$, in turn, measures how much uncertainty about the concepts remains after knowing the messages and should therefore negatively correlate with the agents' performance. Low values indicate that agents effectively use messages that uniquely identify the target concept, i.e. the language does not contain many polysemous expressions. On this basis, we define a *consistency* an *effectiveness* score, using the marginal entropies $\mathcal{H}(C)$ and $\mathcal{H}(M)$ for normalization:

$$\text{consistency}(C, M) = 1 - \frac{\mathcal{H}(M \mid C)}{\mathcal{H}(M)}$$
$$\text{effectiveness}(C, M) = 1 - \frac{\mathcal{H}(C \mid M)}{\mathcal{H}(C)}.$$

Finally, the normalized mutual information,

$$\mathcal{NI}(C, M) = \frac{\mathcal{I}(C, M)}{0.5 \cdot \big(\mathcal{H}(C) + \mathcal{H}(M)\big)} = \frac{\mathcal{H}(M) - \mathcal{H}(M \mid C)}{0.5 \cdot \big(\mathcal{H}(C) + \mathcal{H}(M)\big)},$$

is a symmetric measure that combines the two conditional entropies into one score.

Symbol redundancy. We develop this metric to approximate whether agents repeat information about attribute values in their messages. It assumes that each attribute value, a_v (e.g a=*color*, a_v=*red*), is encoded by a specific symbol and counts how often that symbol is repeated given that a_v is being encoded. The preferred symbol for each attribute value is defined $s_v^a := \arg\max_s \mathcal{I}(a_v, s)$, where we code for each message whether s occurs at least once (the position of s is irrelevant). Symbol redundancy is defined as the average number of occurrences of s_v^a per message given that a_v is part of the target concept.

Topographic similarity. Topographic similarity (*topsim*) measures to what degree similar inputs are described by similar messages and is frequently used as a measure of compositionality. The metric calculates the pairwise distances between the inputs, as well as the pairwise distances between the corresponding messages, and then correlates the two distance vectors (Brighton and Kirby, 2006). In the hierarchical reference game, we need to calculate the topographic similarity between messages and concepts. We use an n-hot encoding of the concepts (n being the number of attributes) and treat abstraction from each attribute as an additional attribute value. If an attribute is relevant, that value

[5] Smaller factors make the task more difficult and performance decreases, while larger factors do not yield any further improvements.

is zero (no abstraction), if an attribute is irrelevant this value is one (abstraction) and overwrites the original attribute value. Assuming that each attribute can take on $k = 4$ different values, the input encoding for the example in Figure 2 becomes:

$$\begin{aligned}\text{sender input} &: [4\,3\,1]\,[1\,0\,0]\,\text{(object + relevance)}\\ \text{encoding} &: [0\,0\,0\,1\,0\,0\,0\,0\,0\,1\,0\,0\,0\,0\,1]\end{aligned}$$

Analogously to Lazaridou et al. (2018), we calculate the pairwise distances of the inputs using the cosine distance, and the pairwise distances between the messages using the edit distance. The topsim score is calculated as the Spearman correlation between the two resulting distance vectors.

Disentanglement. Positional disentanglement (*posdis*) and bag-of-symbols disentanglement (*bosdis*) are used to measure different types of compositionality (Chaabouni et al., 2020). For both metrics, concepts are encoded as for the topsim score. Posdis measures whether symbols in specific positions encode the values of a specific attribute, i.e. whether the compositional structure is order-dependent. Let s_j be the j-th symbol of a message, then posdis is defined as

$$\text{posdis} = \frac{1}{L}\sum_{j=1}^{L}\frac{\mathcal{I}(s_j, a_1^j) - \mathcal{I}(s_j, a_2^j)}{\mathcal{H}(s)},$$

where L is the maximal message length, and a_1^j and a_2^j are the attributes that achieve the highest and second-highest mutual information with s_j ($a_1^j = \arg\max_a \mathcal{I}(s_j, a)$; $a_2^j = \arg\max_{a \neq a_1^j} \mathcal{I}(s_j, a)$). Bosdis measures whether symbols refer to specific attribute values independent of their position. In that case, the language is permutation-invariant and only symbol counts matter. Let n_j be a counter of the j-th symbol in a message, then bosdis is defined as

$$\text{bosdis} = \frac{1}{|V|}\sum_{j=1}^{|V|}\frac{\mathcal{I}(n_j, a_1^j) - \mathcal{I}(n_j, a_2^j)}{\mathcal{H}(n_j)},$$

where V is the vocabulary size, and a_1^j and a_2^j achieve the highest and the second-highest mutual information with n_j.

5 Results

In this section, quantitative and aggregated results will be presented. Random examples of concepts and messages, together with a qualitative analysis can be found in Appendix D. Appendix D.1 shows example mappings between abstract concepts and messages and Appendix D.2 highlights different abstraction strategies.

5.1 Performance and generalization

Figure 3 shows the mean accuracies on training, validation, and zero-shot test sets for all data sets. Training accuracies (top left) and validation accuracies (top right) are very high for each data set, considering that chance performance is $< 10\%$. Thus, the agents learn to refer to objects at different levels of abstraction, and their strategies do not overfit the training data.

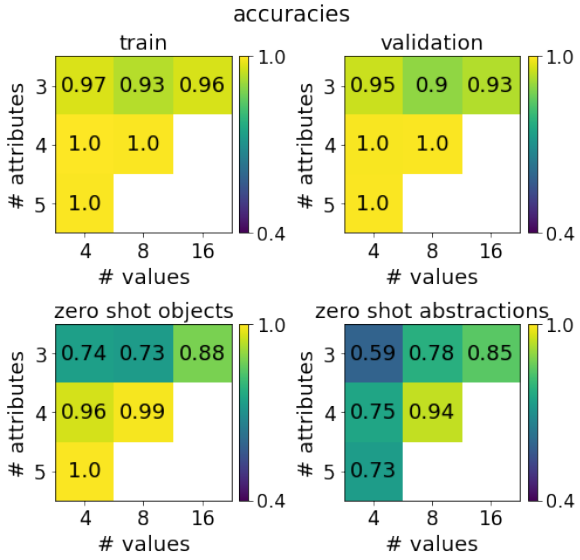

Figure 3: Mean accuracies across five runs for each of the training data sets. Shown are accuracies on the training set, the validation set, and the two zero-shot test sets.

Accuracies for novel combinations of attribute values (bottom left) are consistently higher than accuracies for novel combinations of abstraction and attribute value (bottom right), except for $D(3, 8)$. Accordingly, generalizing to novel abstractions of attribute values is harder than generalizing to novel objects. Both types of generalization tend to improve with the number of attributes as well as the number of values, which may be due to an increase in input space size (Chaabouni et al., 2020). Similar to training and validation accuracies, generalization to novel objects reaches almost perfect accuracies, if there are many attributes. While generalization to novel abstractions is more difficult, accuracies strongly exceed chance performance and are still

very high for $D(3, 16)$ with 84.76% and $D(4, 8)$ with 94.38%. A large number of attribute values seems to be more important for generalizing to novel abstractions than for generalizing to novel objects, possibly because it is more useful to learn systematic abstraction if there are many attribute values. A strategy that abstracts from a certain attribute can be applied to more concepts if that attribute has many values (i.e. has more children on the concept hierarchy). Overall, the agents develop hierarchical reference systems and, with enough attributes and values, these systems generalize well to novel objects and novel abstractions.

5.2 Mapping between concepts and messages

We determine the structure of correspondences between messages and concepts. Figure 4 shows the mean effectiveness and consistency scores. The effectiveness score measures how much information about the target concept is contained in the message. It follows that the agents can only achieve high performance if the language is effective. The results show this interrelation, in that the pattern of effectiveness scores matches the pattern of training and validation accuracies across the different data sets. The consistency score, on the other hand, measures whether a concept is consistently mapped onto the same message, and high consistency is not necessary to achieve high performance. The score is higher for a larger number of attribute values, supporting the finding above that many values per attribute increase the pressure to develop systematic abstraction strategies.

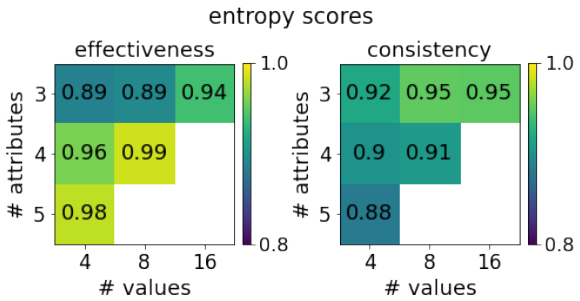

Figure 4: Mean effectiveness and consistency scores. We display the mean scores across five runs for each of the training data sets.

For each data set, the normalized mutual information lies between the effectiveness and the consistency score. It is generally high ($0.902 \leq \mathcal{NI} \leq 0.945$), indicating that messages and concepts are strongly predictive of each other. A one-to-one correspondence between words and messages is not enforced by the setup because the message space is far larger than the concept space. The high entropy scores mean that a systematic mapping between concepts and messages, and therefore also systematic abstraction emerge nonetheless.

To analyze where the languages deviate from a one-to-one correspondence between concepts and messages, we consider the relation between entropy scores and level of abstraction (see Figure 5). The mutual information between messages and concepts is higher for more concrete concepts. This effect is largely driven by an increase in consistency, while effectiveness is relatively constant across all levels of abstraction. Thus, deviations from the one-to-one correspondence between concepts and messages occur mostly for abstract concepts. These deviations arise because different messages map to the same concept, not vice versa. In other words, the languages contain synonymy but no polysemy.

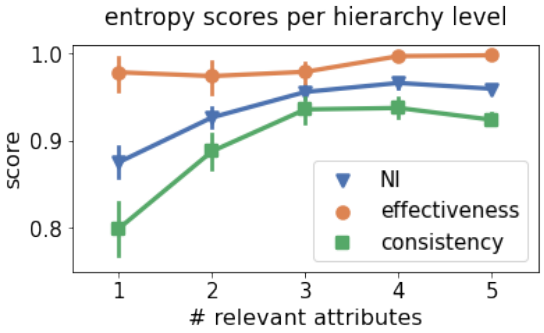

Figure 5: Mean entropy scores across all data sets for different numbers of relevant attributes: from left to right concepts become more concrete. Error bars indicate bootstrapped 95% confidence intervals.

5.3 Linguistic abstraction strategies

Here, we look more closely at the types of internal message structures used to create a hierarchical reference system.

Implicit versus explicit abstraction. In natural language, there are implicit and explicit ways of communicating that attributes are irrelevant. A commonplace implicit strategy is to simply omit information about irrelevant attributes, e.g. one might say "car" rather than "red car" if sufficient. Since the maximal message length corresponds to the maximal number of relevant attributes, the agents could achieve a similar effect by using shorter messages for more abstract concepts or by using messages that contain more redundancies. Figure 6

shows message length and symbol redundancy averaged across data sets for each level of abstraction. The agents indeed use implicit abstraction strategies and this is captured by both metrics. The message length decreases for more abstract concepts while symbol redundancy increases. For abstract concepts, symbols that encode irrelevant attributes are either omitted or replaced by repetitions of symbols encoding relevant information.

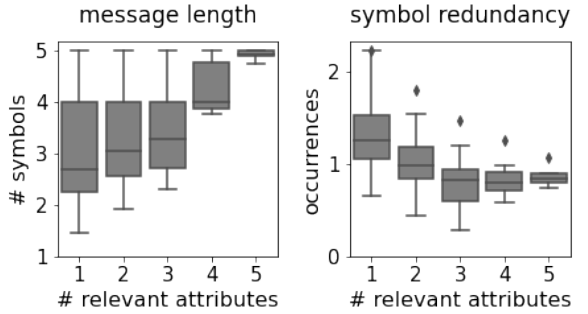

Figure 6: Average message length and symbol redundancy across data sets for different numbers of relevant attributes: from left to right concepts become more concrete.

Explicit abstraction would mean that the agents dedicate symbols to express that information is irrelevant. Such *abstraction operators* should co-occur frequently with abstract concepts. We calculate the average number of symbol occurrences per message for each level of abstraction. We rank the symbols by their occurrences for the most abstract concepts to identify candidate symbols. Figure 7 shows the results for the top ten candidates, averaging multiple runs for each ranked symbol. The ranking is visible in the left-most columns, where the number of occurrences per message decreases monotonously from the highest to the lowest rank. Strikingly, for all data sets except $D(3, 4)$ only 1–3 symbols occur very frequently together with very abstract concepts and the occurrence values decrease rapidly when going further down the ranks. Importantly, these symbols do not occur frequently at every level of the concept hierarchy. Rather, their usage decreases continuously as concepts become more concrete, as indicated by the gradient from left to right in the top rows. Thus, it seems likely that the agents use one or a few symbols to explicitly communicate information about the irrelevance of one or more attributes. The formation of abstraction operators is surprising since the message space is large enough to encode irrelevance differently, for example by combining symbols or using different symbols for different attributes.

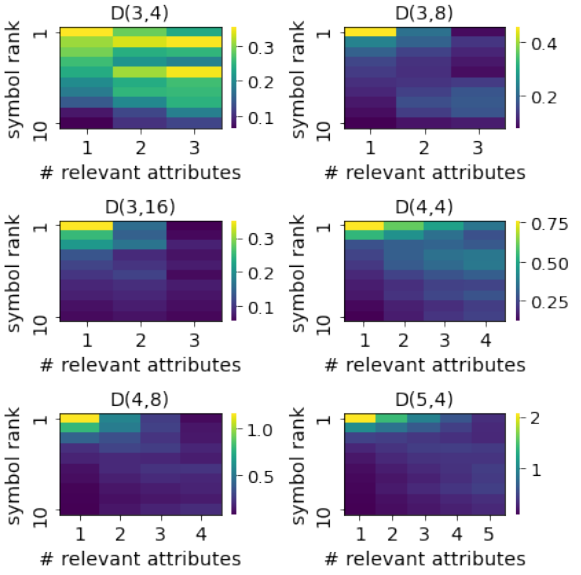

Figure 7: Average number of symbol occurrences per message for each level of abstraction. Symbols are ranked based on their occurrences for the most abstract concepts (i.e. with the fewest relevant attributes). Results are averaged across runs based on ranked symbols and shown only for the top ten ranks.

Compositional versus holistic abstraction. The hierarchical reference game requires the agents to repeatedly communicate the same attribute values but for different concepts—different because of the values of other attributes (traversing the hierarchy horizontally) or because of the level of abstraction (traversing the hierarchy vertically). Although the agents could develop holistic protocols, this repeated reference across contexts might encourage them to develop "reusable" mappings from attribute values to symbols, i.e. compositional expressions.

We use the different compositionality metrics to quantify the degree and nature of compositionality in the messages. Mean scores for each metric and data set can be found in Appendix C. The mean topsim score across data sets is 0.424. The score is even higher, with 0.501, if only concrete concepts are taken into account (as in a standard reference game). The mean posdis score across data sets is 0.115 and the mean bosdis score 0.406. So, there is compositional structure in the messages, and the agents prefer to use specific symbols per attribute value, independent of their position in the messages.

In additional experiments (see Appendix A), we

trained the agents on $D(4, 8)$ with different vocabulary sizes, using factors $f \in \{1, 2, 3, 4\}$ of the minimal vocab size in Table 2. While a fully positional encoding can be achieved with a smaller vocabulary ($f = 1$), a fully position-independent encoding requires a larger vocabulary. Mean training accuracies for $f = 1$ are 0.936, and for all other factors > 0.99. Figure 8 shows the compositionality scores for each factor. Surprisingly, all scores tend to increase with the vocabulary size, regardless of whether the corresponding type of compositionality requires a large vocabulary size or not. Usually, vocabulary size is reduced to increase the pressure for compositional solutions (Kottur et al., 2017). In our case, compositionality probably increases with vocabulary size because the emerging compositional structure is largely non-positional.

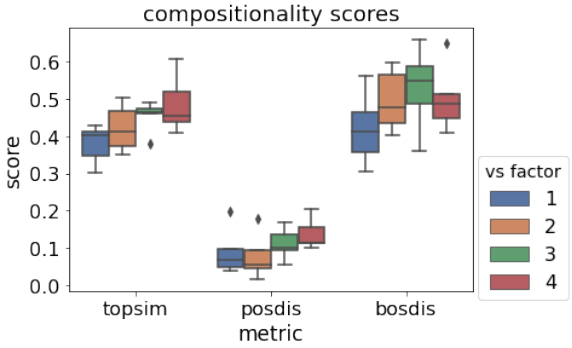

Figure 8: Boxplots of the compositionality scores for $D(4, 8)$ and different vocabulary sizes.

6 Conclusion

In this work, we developed a hierarchical reference game to study the emergence of hierarchical reference systems. In the game, concepts are defined as abstractions over a set of attributes. To refer to these concepts, our agents developed abstract terms and used these terms systematically, in the sense that they could generalize to novel objects and novel abstractions. It seems that, aside from more obvious strategies such as leaving out irrelevant information, the agents developed abstraction operators to explicitly indicate the irrelevance of certain attributes. Even more surprisingly, for some data sets, they used *the same* few symbols to indicate irrelevance across attributes, rather than a dedicated symbol per attribute. While the game design encourages the emergence of abstract concepts, the use of specific abstraction operators emerged without any explicit pressure.

In addition, our results suggest that compositional language may emerge as part of a hierarchical reference strategy. In the classical reference game, the sender typically tries to communicate the union of all object attributes. Without additional pressures, the emerging languages are not compositional (Kottur et al., 2017; van der Wal et al., 2020; Dagan et al., 2021). In the hierarchical reference game, in contrast, the sender must pick out specific attributes for communication, which potentially stimulates disentanglement. This interpretation is in line with the finding that the emergence of compositionality is supported by an increasing number of relevant events that can be referred to (Nowak et al., 2000). In the hierarchical reference game, cross-situational reuse is increased, as reference to attribute values occurs not only across objects but also across levels of abstraction.

We envision two main directions for future work. First, we would like to implement a hierarchical reference game with raw visual inputs instead of symbolic input vectors. Higgins et al. (2018) have developed a neural network (SCAN) that not only learns disentangled visual primitives in an unsupervised manner but also abstractions over such primitives from very few symbol-image pairs that apply to a particular concept. Combining our language emergence game with such a network would allow us to study the simultaneous emergence of abstract visual and linguistic concepts, as well as interactions between these two processes. Second, instead of hard-coding the relevance vector, the relevance of certain attributes should arise from the agents' intentions. Ideally, the agents would play a more complex game and determine themselves which properties of the environment are relevant for their objectives in the current context. Besides, sender and receiver could use pragmatic reasoning (as for example in Choi et al., 2018; Kang et al., 2020; Yuan et al., 2020) to encode and decode which attributes should be emphasized to communicate certain concepts.

Acknowledgements

This work was funded by the Deutsche Forschungsgemeinschaft (DFG, German Research Foundation)—GRK 2340. We would like to thank Dieuwke Hupkes, Leon Schmid, Lucas Weber, and the anonymous reviewers for their valuable feedback and suggestions.

References

Jacob Andreas and Dan Klein. 2016. Reasoning about pragmatics with neural listeners and speakers. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 1173–1182, Austin, Texas. Association for Computational Linguistics.

Douglas Appelt and Amichai Kronfeld. 1987. A computational model of referring. In *Proceedings of the 10th International Joint Conference on Artificial Intelligence (IJCAI)*, pages 640–647, San Francisco, CA, USA. Morgan Kaufmann Publishers Inc.

Douglas E. Appelt. 1985. Planning English referring expressions. *Artificial Intelligence*, 26(1):1–33.

Yonatan Bisk, Ari Holtzman, Jesse Thomason, Jacob Andreas, Yoshua Bengio, Joyce Chai, Mirella Lapata, Angeliki Lazaridou, Jonathan May, Aleksandr Nisnevich, Nicolas Pinto, and Joseph Turian. 2020. Experience grounds language. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 8718–8735, Online. Association for Computational Linguistics.

Henry Brighton and Simon Kirby. 2006. Understanding linguistic evolution by visualizing the emergence of topographic mappings. *Artificial Life*, 12(2):229–242.

Angelo Cangelosi and Domenico Parisi. 2002. *Simulating the Evolution of Language*. Springer, London.

Rahma Chaabouni, Eugene Kharitonov, Diane Bouchacourt, Emmanuel Dupoux, and Marco Baroni. 2020. Compositionality and generalization in emergent languages. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 4427–4442, Online. Association for Computational Linguistics.

Edward Choi, Angeliki Lazaridou, and Nando de Freitas. 2018. Compositional obverter communication learning from raw visual input. In *6th International Conference on Learning Representations*.

Gautier Dagan, Dieuwke Hupkes, and Elia Bruni. 2021. Co-evolution of language and agents in referential games. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*, pages 2993–3004, Online. Association for Computational Linguistics.

Robert Dale and Ehud Reiter. 1995. Computational interpretations of the gricean maxims in the generation of referring expressions. *Cognitive Science*, 19(2):233–263.

Roberto Dessi, Eugene Kharitonov, and Baroni Marco. 2021. Interpretable agent communication from scratch (with a generic visual processor emerging on the side). In *Advances in Neural Information Processing Systems (NeurIPS)*, volume 34, pages 26937–26949. Curran Associates, Inc.

Jakob N. Foerster, Yannis M. Assael, Nando de Freitas, and Shimon Whiteson. 2016. Learning to communicate with deep multi-agent reinforcement learning. In *Advances in Neural Information Processing Systems (NeurIPS)*, volume 30, pages 2145–2153, Red Hook, NY, USA. Curran Associates Inc.

Michael C. Frank and Noah D. Goodman. 2012. Predicting pragmatic reasoning in language games. *Science*, 336(6084):998–998.

Herbert P. Grice. 1975. Logic and conversation. In *Syntax and Semantics 3: Speech Acts*, pages 41–58. Academic Press, New York, NY.

Serhii Havrylov and Ivan Titov. 2017. Emergence of language with multi-agent games: Learning to communicate with sequences of symbols. In *Advances in Neural Information Processing Systems (NeurIPS)*, volume 30, pages 2149–2159.

Irina Higgins, Nicolas Sonnerat, Loic Matthey, Arka Pal, Christopher P. Burgess, Matko Bošnjak, Murray Shanahan, Matthew Botvinick, Demis Hassabis, and Alexander Lerchner. 2018. SCAN: Learning hierarchical compositional visual concepts. In *6th International Conference on Learning Representations*.

Eric Jang, Shixiang Gu, and Ben Poole. 2017. Categorical reparameterization with gumbel-softmax. In *5th International Conference on Learning Representations*.

Yipeng Kang, Tonghan Wang, and Gerard de Melo. 2020. Incorporating pragmatic reasoning communication into emergent language. In *Advances in Neural Information Processing Systems (NeurIPS)*, volume 33, pages 10348–10359.

Sahar Kazemzadeh, Vicente Ordonez, Mark Matten, and Tamara Berg. 2014. ReferItGame: Referring to objects in photographs of natural scenes. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 787–798, Doha, Qatar. Association for Computational Linguistics.

Eugene Kharitonov, Rahma Chaabouni, Diane Bouchacourt, and Marco Baroni. 2019. EGG: a toolkit for research on emergence of lanGuage in games. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP): System Demonstrations*, pages 55–60, Hong Kong, China. Association for Computational Linguistics.

Simon Kirby. 2002. Natural language from artificial life. *Artificial life*, 8(2):185–215.

Satwik Kottur, José Moura, Stefan Lee, and Dhruv Batra. 2017. Natural language does not emerge 'naturally' in multi-agent dialog. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 2962–2967, Copenhagen, Denmark. Association for Computational Linguistics.

Emiel Krahmer and Kees van Deemter. 2012. Computational generation of referring expressions: A survey. *Computational Linguistics*, 38(1):173–218.

Angeliki Lazaridou and Marco Baroni. 2020. Emergent multi-agent communication in the deep learning era. *Computing Research Repository*, arXiv:abs/2006.02419. Version 2.

Angeliki Lazaridou, Karl Moritz Hermann, Karl Tuyls, and Stephen Clark. 2018. Emergence of linguistic communication from referential games with symbolic and pixel input. In *6th International Conference on Learning Representations*.

Angeliki Lazaridou, Alexander Peysakhovich, and Marco Baroni. 2017. Multi-agent cooperation and the emergence of (natural) language. In *5th International Conference on Learning Representations*.

Hieu Le, Taufiq Daryanto, Fabian Zhafransyah, Derry Wijaya, Elizabeth Coppock, and Sang Chin. 2022. Referring expressions with rational speech act framework: A probabilistic approach. *Computing Research Repository*, arXiv:2205.07795.

David Lewis. 1969. *Convention*. Harvard University Press, Cambridge, MA.

Gen Luo, Yiyi Zhou, Xiaoshuai Sun, Liujuan Cao, Chenglin Wu, Cheng Deng, and Rongrong Ji. 2020. Multi-task collaborative network for joint referring expression comprehension and segmentation. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.

Ruotian Luo and Gregory Shakhnarovich. 2017. Comprehension-guided referring expressions. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 3125–3134.

Junhua Mao, Jonathan Huang, Alexander Toshev, Oana Camburu, Alan Yuille, and Kevin Murphy. 2016. Generation and comprehension of unambiguous object descriptions. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 11–20.

Will Monroe and Christopher Potts. 2015. Learning in the rational speech acts model. *Computing Research Repository*, arXiv:1510.06807.

Martin A. Nowak, Joshua B. Plotkin, and Vincent A. A. Jansen. 2000. The evolution of syntactic communication. *Nature*, 404(6777):495–498.

Ehud Reiter and Robert Dale. 1992. A fast algorithm for the generation of referring expressions. In *Proceedings of the 14th International Conference on Computational Linguistics*, pages 232–238, USA. Association for Computational Linguistics.

Diana Rodríguez Luna, Edoardo Maria Ponti, Dieuwke Hupkes, and Elia Bruni. 2020. Internal and external pressures on language emergence: least effort, object constancy and frequency. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 4428–4437, Online. Association for Computational Linguistics.

Eleanor Rosch and Carolyn B Mervis. 1975. Family resemblances: Studies in the internal structure of categories. *Cognitive Psychology*, 7(4):573–605.

Eleanor Rosch, Carolyn B Mervis, Wayne D Gray, David M Johnson, and Penny Boyes-Braem. 1976. Basic objects in natural categories. *Cognitive Psychology*, 8(3):382–439.

Luc Steels. 2001. Language games for autonomous robots. *IEEE Intelligent Systems*, 16(5):16–22.

Luc Steels. 2003. Evolving grounded communication for robots. *Trends in Cognitive Sciences*, 7(7):308–312.

Oskar van der Wal, Silvan de Boer, Elia Bruni, and Dieuwke Hupkes. 2020. The grammar of emergent languages. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 3339–3359, Online. Association for Computational Linguistics.

Kyle Wagner, James A. Reggia, Juan Uriagereka, and Gerald S. Wilkinson. 2003. Progress in the simulation of emergent communication and language. *Adaptive Behavior*, 11(1):37–69.

Terry Winograd. 1972. Understanding natural language. *Cognitive Psychology*, 3(1):1–191.

Licheng Yu, Zhe Lin, Xiaohui Shen, Jimei Yang, Xin Lu, Mohit Bansal, and Tamara L. Berg. 2018. Mattnet: Modular attention network for referring expression comprehension. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1307–1315.

Licheng Yu, Patrick Poirson, Shan Yang, Alexander C. Berg, and Tamara L. Berg. 2016. Modeling context in referring expressions. In *European Conference on Computer Vision (ECCV)*, pages 69–85, Cham. Springer International Publishing.

Luyao Yuan, Zipeng Fu, Jingyue Shen, Lu Xu, Junhong Shen, and Song-Chun Zhu. 2020. Emergence of pragmatics from referential game between theory of mind agents. *Computing Research Repository*, arXiv:2001.07752. Version 2.

A Varying distractor sampling and vocab size

A.1 Setup

We conduct control experiments, changing the vocabulary size, and changing the distractor sampling strategy. In the original experiments, the message space is much larger than the space of concepts that need to be communicated. By reducing the vocabulary size, we aim to test whether a smaller message space increases the probability of one-to-one associations between concepts and messages. In addition, the distractors are sampled from concepts that are one level more abstract than the target concept on the concept hierarchy. Here, we relax this assumption by sampling distractors from all levels of the concept hierarchy with equal probability. Together, these additional experiments allow us to extend our results to different vocabulary sizes, and more general distractor distributions.

We focus on a single data set. We use the data set with four attributes and eight values per attribute, $D(4, 8)$, which achieved the highest mean validation accuracies and normalized mutual information scores in the original setup. In the original experiments, we used a factor of 3 of the minimal vocabulary size 9 (8 for each value plus 1 for coding irrelevance). Now, we run the same experiment for factors of 1, 2, and 4; and in addition, we repeat the experiment for each factor with the alternative sampling strategy. Again, we conduct five runs for each factor and sampling strategy.

A.2 Results

Figure 9 shows the accuracy scores for the different vocabulary size factors, and the different distractor sampling strategies, where *unbalanced* refers to the original strategy of selecting distractors from more abstract concepts, and *balanced* refers to the control strategy of sampling distractors with equal probability from all levels of abstraction. For both sampling strategies, performance is higher if the vocabulary size is large, likely because having a larger message space increases the number of solutions. A larger vocabulary size seems to be particularly important if distractor sampling is balanced.

The original, unbalanced sampling strategy achieves higher performance than the control strategy on all data sets. So, choosing distractors very similar to the target facilitates learning, and probably also abstraction as suggested by the zero-shot evaluation with new abstractions. To make sure that

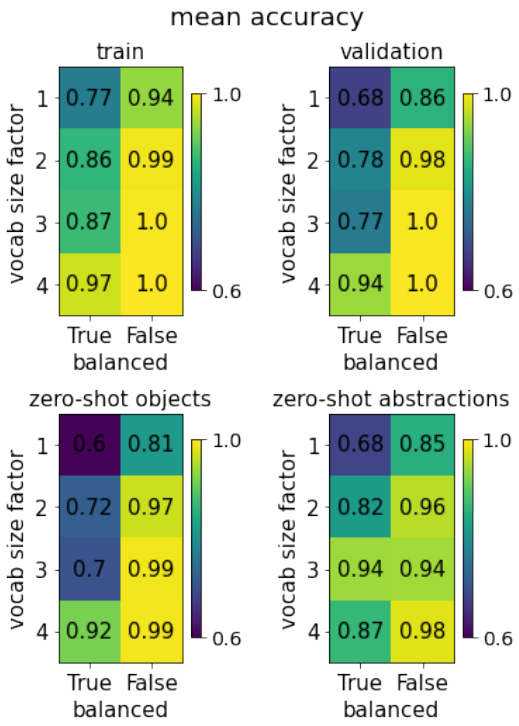

Figure 9: Mean accuracies for the control experiments across five runs, on the training data, the validation data, and the two zero-shot test sets. The y-axis gives the factor used to determine the vocabulary size, vocab size = factor $\times$ minimal vocab size, and the x-axis indicates whether distractors are sampled from concepts that are one level more abstract than the target concept (unbalanced), or sampled from all levels of the concept hierarchy with equal probability (balanced).

the unbalanced sampling strategy only facilitates learning but does not make the task easier, we run an ablation test. We evaluate each sender-receiver pair on the validation set of the sampling strategy that was *not* used for training. For all vocabulary sizes and runs, the agents perform better on the balanced validation set compared to the unbalanced validation set, regardless of the sampling method used during training. In conclusion, sampling distractor concepts that are very similar to the target concept makes the task more difficult but improves learning by increasing the pressure to communicate only relevant aspects, and thereby to develop abstract concepts.

These results are confirmed by the entropy-based evaluation metrics shown in Figure 10. Effectiveness and consistency are consistently lower for the balanced distractor sampling strategy. However, while the level of abstraction does not have a strong effect on the difference in effectiveness scores, the difference in consistency scores decreases contin-

uously with the level of specificity. In line with the generalization ability, this suggests that the unbalanced sampling strategy supports the formation of abstract concepts by reducing the probability of successful target selection if irrelevant attributes are communicated.

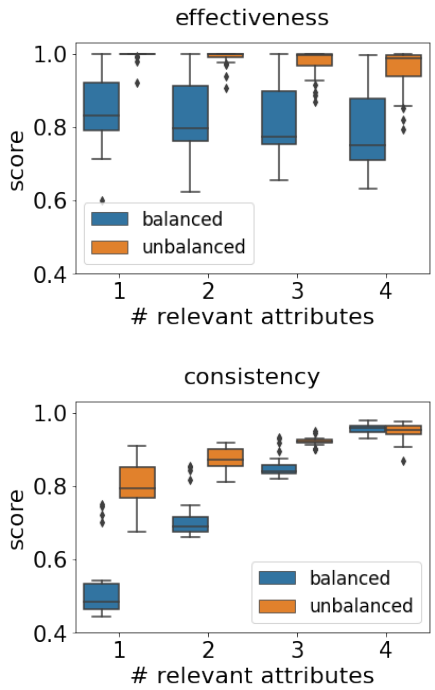

Figure 10: Effectiveness and consistency scores for balanced and unbalanced distractor sampling, separated for each level of abstraction. Distributions show the results across the different vocabulary sizes and runs. The level of abstraction is given on the x-axis: from left to right the concepts become more concrete.

B Hyperparameter search

We ran our hyperparameter search for the three data sets spanning up the space of all data sets we use, $D(3, 4)$, $D(5, 4)$, and $D(3, 16)$ (see Table 1). We expected that hyperparameters working across all these extreme cases should also work for interpolations between them. Certain hyperparameters were fixed across the search. We used GRUs with Adam optimizer, and a GS temperature of 1.5 with an exponential decay rate. Message length cost was 0, and vocab size factor 3. We varied the following hyperparameters:

- batch size: $\{32, 64, 128\}$
- learning rate: $\{0.0005, 0.001\}$
- hidden layer dimension: $\{128, 256\}$
- embedding layer dimension: always half of the hidden layer dimension
- GS temperature decay rate: $\{0.97, 0.99\}$

For the grid search we stopped the training process after 60 epochs. All results can be found in our repository.

C Compositionality scores

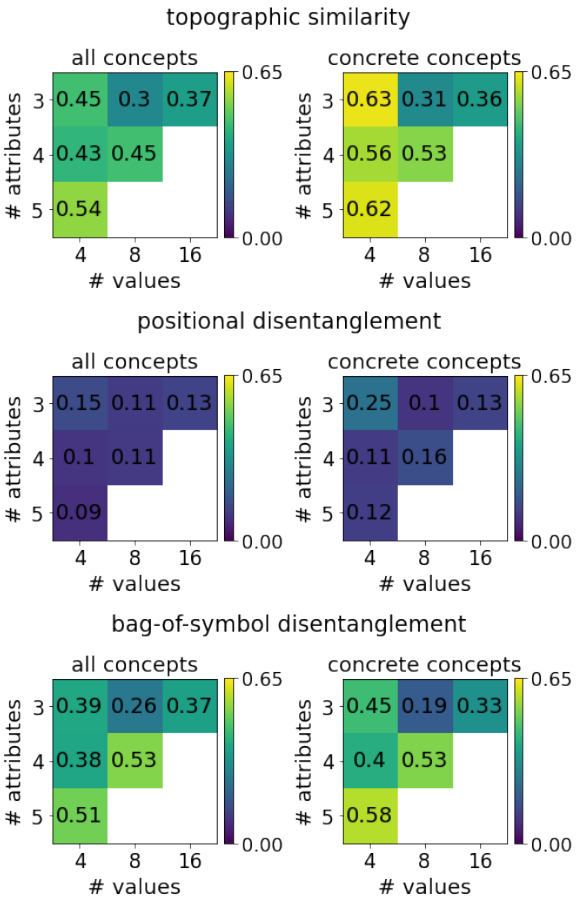

Figure 11: Mean compositionality scores per data set.

D Qualitative examples

This section provides qualitative examples of concept-message pairs. Examples were randomly selected from the first run of each data set. Interestingly, this microcosm of random examples reflects all communication patterns that were identified in the quantitative analyses.

D.1 Mappings between concepts and messages

We are interested in whether the agents use the same message to refer to abstract concepts regardless of how these concepts are instantiated. Figure

12 shows the messages for a randomly selected concept at the highest level of abstraction (only one attribute is relevant), instantiated by different attribute vectors for each data set. Shown are twenty randomly selected instances each, and the examples are sorted by message.

Abstraction is relatively systematic. For all data sets, the agents group together different concept instances in their messages. For some data sets, the instances are grouped under very few messages. For example, the sender trained on $D(4, 4)$ groups together all example instances of the concept $(\_, 1, \_, \_)$ under just two different messages ($(2, 1, 2, 0)$ and $(2, 2, 1, 2)$). Across data sets, 2, 3, 5, or 7 different messages are used to describe the 20 example instances. In line with the quantitative results, there is no perfect one-to-one correspondence between *abstract* concepts and messages. How many different messages are used also depends on the abstraction strategy (see Appendix D.2).

D.2 Abstraction strategies

To visualize the agents' abstraction strategies, we randomly selected an object (i.e. attribute vector) for each data set, and show the messages for that object across the concept hierarchy, so for each abstraction in the training set. Because of their large number the examples are split into two figures, Figure 13 ($D(3, 4)$, $D(3, 8)$, and $D(3, 16)$) and Figure 14 ($D(4, 4)$, $D(4, 8)$, and $D(5, 4)$), which will be analyzed together. The messages will first be analyzed for compositional structure, and then for implicit versus explicit abstraction.

Compositional versus holistic abstraction. For some data sets, the agents seem to use *trivially compositional* messages, i.e. messages whose meaning corresponds to the intersection of meanings of their constituents. An unambiguous pattern can be identified for $D(3, 4)$ and $D(4, 8)$, where a mapping between each attribute value and a specific symbol can be established (color-coded in orange, green, blue, and purple). In the case of data set $D(4, 8)$, the number of additional "filler" symbols increases with the level of abstraction (color-coded in red). These might serve as abstraction operators (see below). For other data sets, like $D(5, 4)$, such mappings can only be identified for specific attribute values (color-coded in purple). Here, symbol 1 occurs if and only if the third attribute has the value 3. For all identified mappings, the symbols are used to encode specific attribute values relatively independent of their position, which is in line with the high bosdis and low posdis scores in our quantitative analyses.

For the remaining data sets, the messages are not unstructured but no one-to-one correspondences can be identified. For example, looking at the messages for $D(3, 16)$, symbol 26 might encode value 5 at position 1 for the most concrete and most abstract concepts but is not used at the intermediate level of abstraction. So, while the abstraction strategies are almost perfectly compositional in some cases, there are large variations between data sets, and potentially also runs and concepts.

Implicit versus explicit abstraction. The examples show instances of implicit and explicit abstraction strategies. Implicit abstraction is identified through shorter messages and more symbol redundancy for higher levels of abstraction; explicit abstraction through the use of abstraction operators. At least for some data sets, the messages tend to become shorter with increasing abstraction. E.g. messages become shorter in the case of $D(3, 16)$ and $D(4, 4)$ (the end-of-sequence symbol 0 is color-coded in gray).

Symbol redundancy and abstraction operators can best be identified in reference systems with compositional structure. $D(3, 4)$ is a perfect example of increasing symbol redundancy. Each symbol corresponds to a specific attribute value, and symbols are repeated to fill up the messages for more abstract concepts. E.g., the concept (1, 2, 2) is encoded as (5, 12, 11), the concept (1, _, 2) as (5, 12, 12), and the concept (1, _, _) as (12, 12, 12).

$D(4, 8)$, on the other hand, is a perfect example of explicit abstraction. As messages become more abstract the frequency of symbols that do not encode an attribute value ($\{3, 10, 11, 13, 14, 16\}$, marked in red) increases. Note that symbol 3 seems to serve both roles, encoding an attribute value as well as encoding abstractions. To confirm the intuition that these additional symbols serve as abstraction operators, we look at other abstract concepts for $D(4, 8)$. Figure 15 shows the messages for 20 random examples. Indeed, at least two of the abstraction operators occur in each message. Only symbol 11 does not occur and might serve a different function. The quantitative analyses suggest that usually less abstraction operators are used than in this specific example. Less compositional protocols may also use explicit abstraction operators

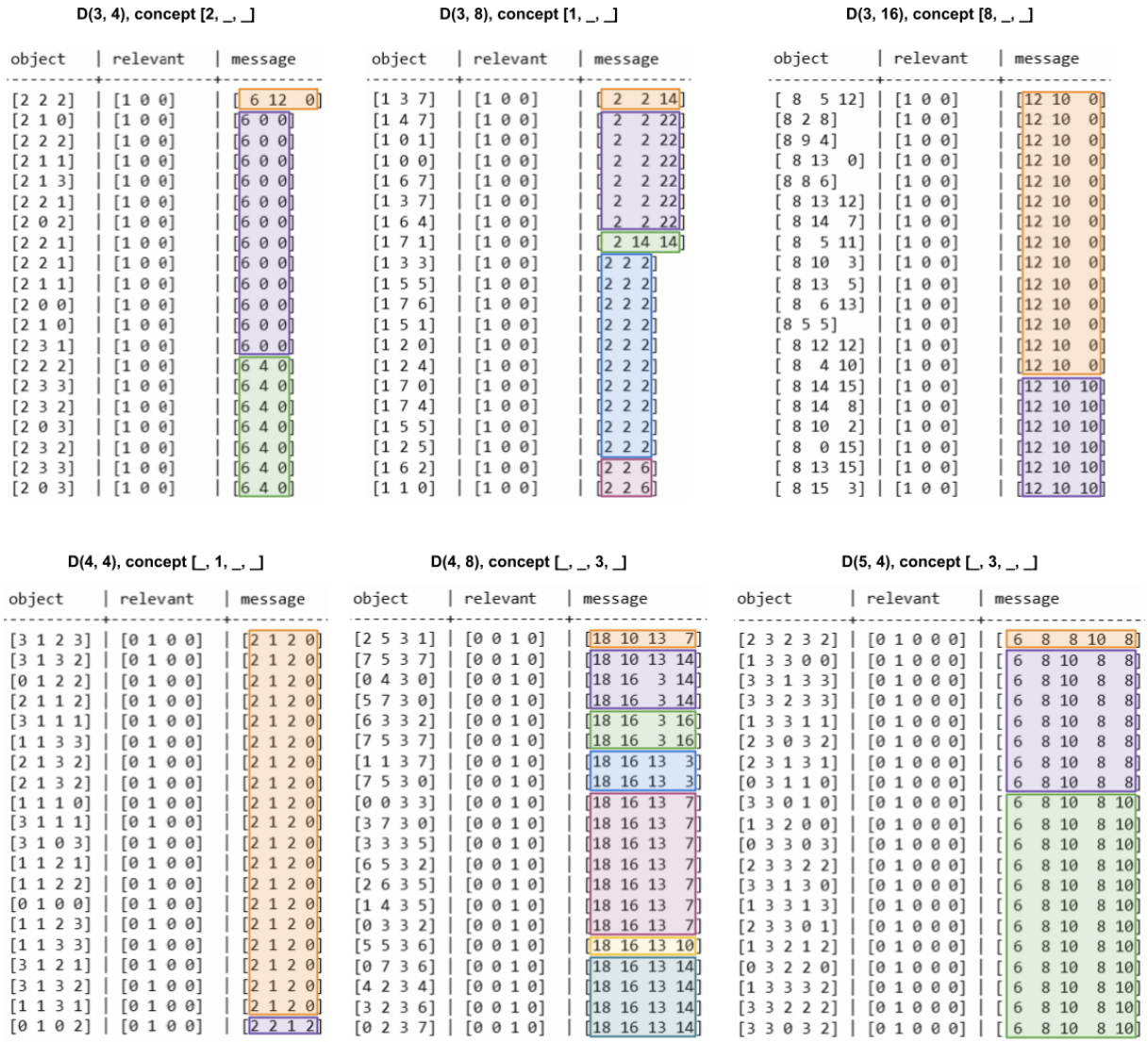

Figure 12: Example messages for one abstract concept per data set. For each data set, we randomly select a concept at the highest level of abstraction. We then randomly select 20 instances of that concept in the training data and display these instances together with the corresponding messages (from the first run). The same messages are grouped together in colored boxes.

or symbol redundancy but these cannot easily be identified in a qualitative analysis.

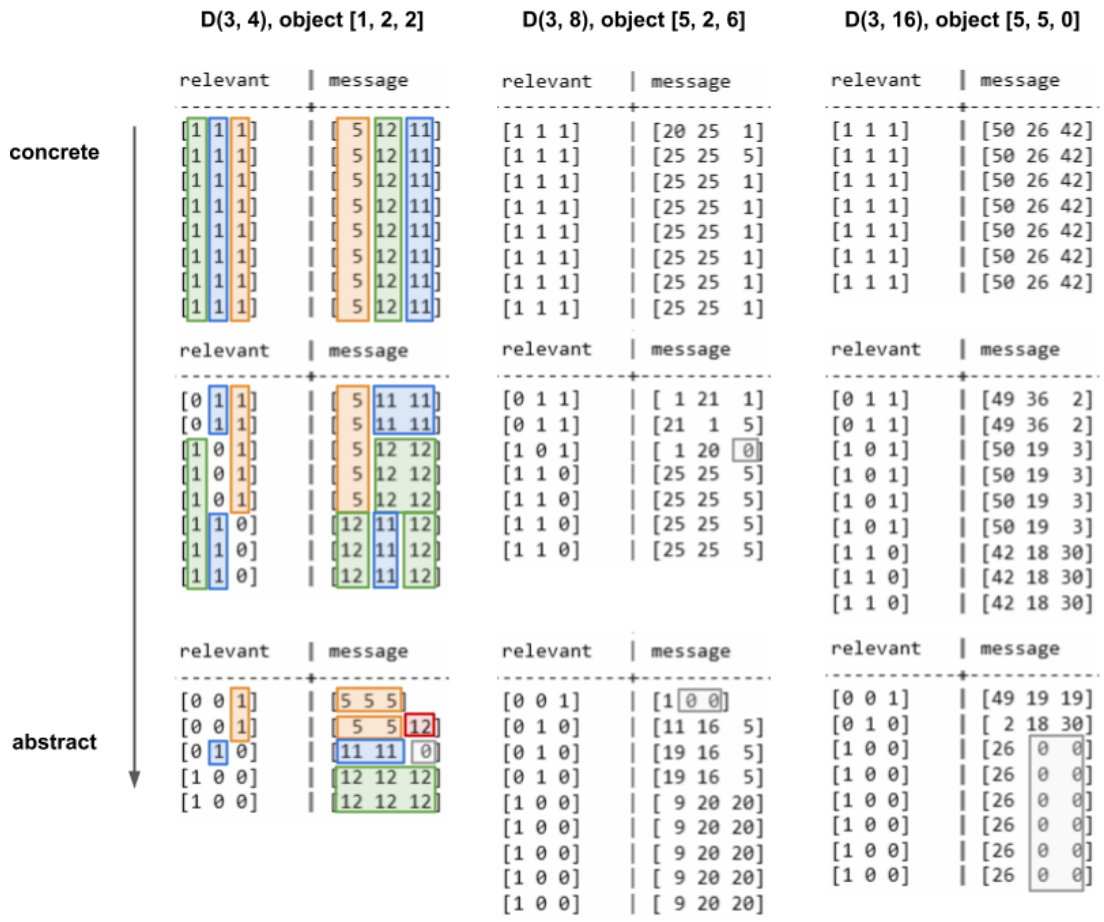

Figure 13: Messages for a random object at each level of abstraction available in the training data. The corresponding messages are shown for the first run of each data set: $D(3, 4)$, $D(3, 8)$, and $D(3, 16)$. The highlighted patterns are explained in the text.

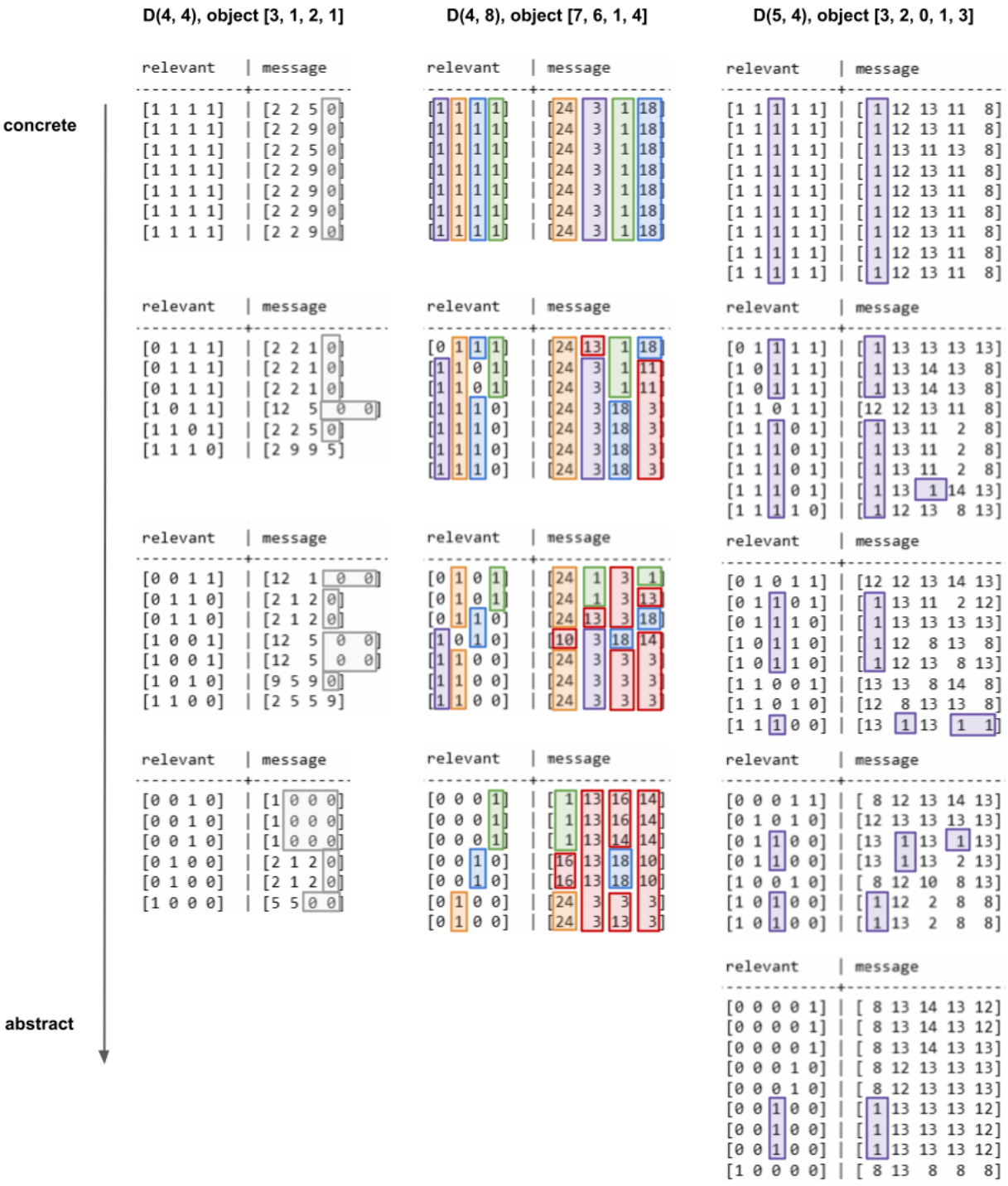

Figure 14: Messages for a random object at each level of abstraction available in the training data. The corresponding messages are shown for the first run of each data set: $D(4, 4)$, $D(4, 8)$, and $D(5, 4)$. The highlighted patterns are explained in the text.

D(4, 8)

object	relevant	message
[0 2 3 5]	[0 0 0 1]	[8 16 13 14]
[3 4 5 2]	[0 1 0 0]	[14 16 14 14]
[6 0 5 4]	[0 0 1 0]	[16 13 9 9]
[0 7 0 5]	[0 0 1 0]	[16 7 16 14]
[1 0 7 4]	[1 0 0 0]	[10 3 17 14]
[5 3 5 4]	[0 1 0 0]	[10 16 14 14]
[2 5 3 4]	[1 0 0 0]	[10 3 14 21]
[7 5 5 0]	[0 0 0 1]	[19 16 3 16]
[6 3 0 6]	[0 0 0 1]	[23 13 14 14]
[0 6 4 2]	[0 0 0 1]	[20 13 14 14]
[2 0 7 5]	[1 0 0 0]	[10 3 14 21]
[7 7 0 2]	[0 0 1 0]	[16 7 13 10]
[3 0 6 0]	[0 0 0 1]	[19 16 3 16]
[7 6 5 4]	[0 0 0 1]	[1 3 16 14]
[6 4 5 7]	[0 0 0 1]	[4 13 14 14]
[6 2 2 1]	[0 0 0 1]	[22 13 22 14]
[3 0 5 3]	[0 0 1 0]	[16 13 9 9]
[5 0 7 3]	[0 1 0 0]	[24 16 14 14]
[6 7 7 2]	[0 0 1 0]	[16 13 6 6]
[2 0 3 6]	[0 1 0 0]	[24 16 14 10]

Figure 15: Messages for 20 randomly selected concepts at the highest level of abstraction, for the first run of $D(4, 8)$. The highlighted patterns are explained in the text.

Scene Graph Modification as Incremental Structure Expanding

Xuming Hu[1*], **Zhijiang Guo**[2*], **Yu Fu**[1], **Lijie Wen**[1†], **Philip S. Yu**[1,3]
[1]Tsinghua University
[2]University of Cambridge
[3]University of Illinois at Chicago
[1]{hxm19,fy20}@mails.tsinghua.edu.cn
[2]zg283@cam.ac.uk [1]wenlj@tsinghua.edu.cn [3]psyu@uic.edu

Abstract

A scene graph is a semantic representation that expresses the objects, attributes, and relationships between objects in a scene. Scene graphs play an important role in many cross modality tasks, as they are able to capture the interactions between images and texts. In this paper, we focus on scene graph modification (SGM), where the system is required to learn how to update an existing scene graph based on a natural language query. Unlike previous approaches that rebuilt the entire scene graph, we frame SGM as a graph expansion task by introducing the incremental structure expanding (ISE). ISE constructs the target graph by incrementally expanding the source graph without changing the unmodified structure. Based on ISE, we further propose a model that iterates between nodes prediction and edges prediction, inferring more accurate and harmonious expansion decisions progressively. In addition, we construct a challenging dataset that contains more complicated queries and larger scene graphs than existing datasets. Experiments on four benchmarks demonstrate the effectiveness of our approach, which surpasses the previous state-of-the-art model by large margins. Source code and data are available[1].

1 Introduction

A scene graph is a structural representation that captures the semantics of visual scenes by encoding object instances, attributes of objects, and relationships between objects. (Johnson et al., 2015). As shown in Figure 1, the scene graph encodes objects (e.g. "*Boy*", "*Racket*"), attributes (e.g. "*Girl is standing*"), and relations ("*Boy holding racket*"). Scene graphs are able to capture the interactions between text and images by associating objects in the graph with regions of an image and modeling the relations between objects. Therefore, it has been used in the cross modality task such as image retrieval, image captioning, and visual question answering (Schuster et al., 2015; Shi et al., 2019; Yang et al., 2019; Wang et al., 2020b).

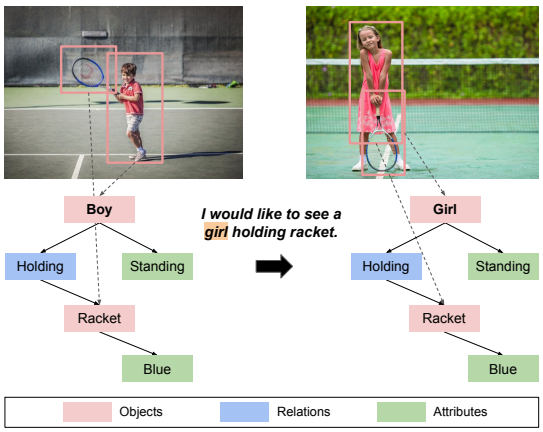

Figure 1: Example images and their corresponding scene graphs. Given the query, the original scene graph (left) is modified to be the target scene graph (right).

Recently, modifying the scene graph based on the input becomes an emerging research direction as cross-modal systems may need to resort to an interactive process through multiple iterations (Ramnath et al., 2019; He et al., 2020). Take text-based image retrieval as an example, users start with a query describing the main objects or topics they are looking for, then modify the query to add more constraints or provide additional information based on previous search results. Instead of directly manipulating images, scene graphs can be used to convert the image-editing problem into a graph-editing problem, conditioned on the textual query. As shown in Figure 1, given a retrieved image from the last turn, if the user wants to see a girl rather than a boy holding a racket, he will enter the query "*I would like to see a girl holding racket*" to the system. According to the query, the object "*Boy*" in the original scene graph will be substituted with the object "*Girl*". The target image can be retrieved

[1]https://github.com/THU-BPM/SGM
[*] Equally Contributed.
[†] Corresponding Author.

Proceedings of the 29th International Conference on Computational Linguistics, pages 5707–5720
October 12–17, 2022.

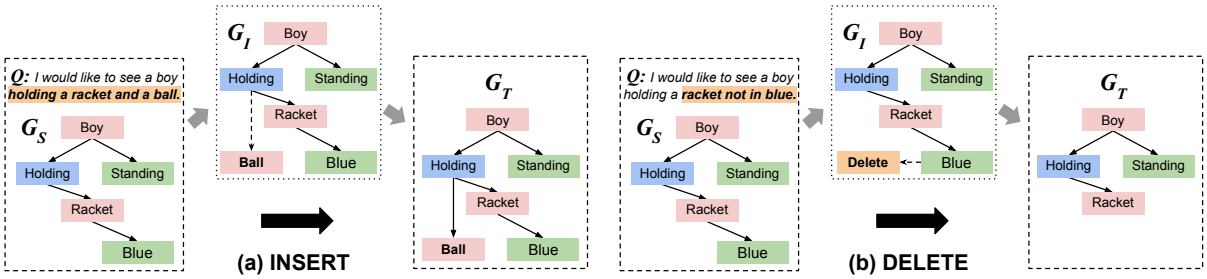

Figure 2: Examples of basic operations INSERT and DELETE for scene graph modification. Q denotes the textual query, G_S denotes the source scene graph, G_T denotes the target scene graph and G_I is the extended graph.

given the updated scene graph. The key challenge in this process is how to modify the corresponding partial structure in the original scene graph based on understanding the natural language query.

Prior effort framed this scene graph modification (SGM) task as conditional graph generation (He et al., 2020), where the scene graph is generated from the scratch condition on the original graph and query (You et al., 2018; Guo et al., 2019; Cai and Lam, 2020b). However, rebuilding the entire scene graph may not be an optimal solution, as the model has to generate the partial structure of the original graph that should be unmodified. Moreover, nodes and edges of the scene graph are constructed separately in their proposed framework, which generates all the nodes first then attaches edges between generated nodes in the second pass. Such an approach may lead to the lack of the modeling capability of interactions between node prediction and edge prediction.

Instead of rebuilding the whole scene graph, we introduce a novel formulation for SGM – incremental structure expanding (ISE), which is able to build the target graph by gradually expanding the original structure. At each step, ISE generates the connecting edges between the existing nodes and the newly generated node, upon which the type of the new node is jointly decided. Based on the formalism, our proposed model is able to iterate between finding the relevant part in the query and reading the partially constructed scene graph, inferring more accurate and harmonious expansion decisions progressively. Experiments on three SGM benchmarks demonstrate the effectiveness of the proposed approach, which is able to outperform previous state-of-the-art models by large margins. To test the ability of a model under a complex scenario, we further construct a more challenging dataset from the remote sensing domain (Lu et al., 2017), which has much more modification operations based on the more complicated queries compared with the existing scene graph modification datasets. Our key contributions are summarized as follows:

- We propose a novel formulation for scene graph modification, allowing incremental expansion of the source scene graph rather than the regeneration of the target graph.
- We further construct a challenging dataset that contains more complicated queries and larger scene graphs. Extensive experiments on four SGM datasets show the effectiveness of our proposed approach.
- Experiments on four benchmarks demonstrate the effectiveness of our approach, which surpasses the previous state-of-the-art model by large margins.

2 Incremental Structure Expanding

In scene graph modification, a node or multiple nodes can be inserted to, deleted from or replaced with other nodes in the scene graph. He et al. (2020) defined the scene graph modification task as a conditional graph generation problem. Formally, given the source scene graph G_S and the natural language query Q, the target scene graph G_T is generated by maximizing the conditional probability $p(G_T|G_S, Q)$.

Instead of generating the entire target graph G_T, we frame the task as an incremental structure expanding, which extends the source scene graph G_S one node at a time, as well as the edges associated with the node. Such a formulation does not require the model to rebuild the unmodified structure of the source scene graph.

Under this formulation, we first define two basic operations: INSERT and DELETE. Scene graph modification can be viewed as combining and applying these two operations multiple times. For-

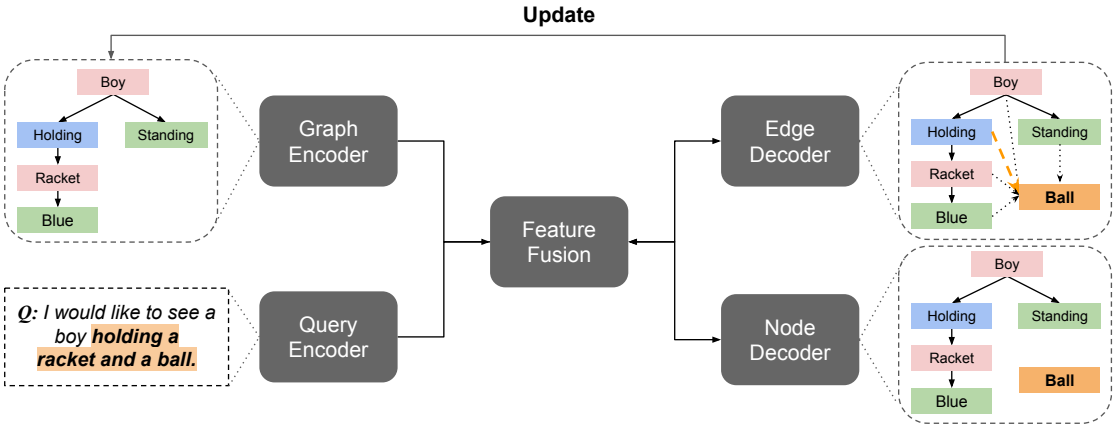

Figure 3: Overview of the model architecture.

mally, given the query Q, a sequence of n operations $a_1, a_2, ..., a_n$ are selected from a set of graph modification operations $\mathcal{A}$ = {INSERT, DELETE}. After applying the operations to the source scene graph G_s, the target scene graph G_t is derived. Each operation is defined as:

- **INSERT**: A new node o is added to G_s, and edges are attached between o and existing nodes in G_s. As shown in Figure 2 (a), the node "*Ball*" is added to G_s and an edge between "*Ball*" and "*Holding*" is attached, according to the query "*holding a racket and a ball*".
- **DELETE**: As shown in Figure 2 (b). A node o is removed from G_s, as well as its associated edges. As shown in Figure 2 (b), the node "*Blue*" is removed from G_s and the edge between "*Racket*" and "*Blue*" is removed either, according to the query "*a racket not a blue*".

Inspired by incremental parsing (Nivre, 2004; Dyer et al., 2015; Cai and Lam, 2020a; Zhang et al., 2021, 2022), we design a data structure called extended graph G_I, which can be used to model INSERT and DELETE under the graph expansion setting. As shown in Figure 2 (a), the extended graph G_T is identical with the target graph G_T after applying INSERT. As for DELETE, we introduce a dummy node "Delete", which is attached to the node in the source graph G_S that should be removed. For example, the dummy node "Delete" is attached to the node "Blue" in G_T. In the postprocessing stage, nodes attached with the dummy node "Delete" will be removed. Using this formulation, we are able to model scene graph modification by incrementally expanding the source graph G_S to the extended graph G_I, which can be converted to the target graph G_T without any losses.

If the modification requires multiple operations, there will exist multiple node orderings. Take node substitution as an example, replacing a node o_i with o_j in G_s can be viewed as DELETE the node o_i first, then INSERT the node o_j, or vice versa. In practice, we impose that the DELETE operation always comes before INSERT, then the breadth-first search is used to define a deterministic node ordering.

3 Model Architecture

In this section, we will present the model based on the incremental structure expanding formulation. Figure 3 gives an overview of the proposed model, which consists of five components including query encoder, graph encoder, feature fusion, edge decoder and node decoder.

Query Encoder This module is used to encode the query Q by generating the representation of each token of it.

Graph Encoder This module is used to encode the graph by generating the representation of each node of it. Note that the representations of the graph are constructed incrementally during the expanding progresses based on the updated graph of the last time step. The graph is the source graph G_S at the first timestep.

Feature Fusion this module aims to combine the representations from query and graph encoder, then served as a writable memory, which is updated based on the information from edge and node decoder during the incremental expansion.

Edge Decoder this module is used to predict the edges between the newly generated node and existing nodes of the graph, then update the memory of the feature fusion module with edge information.

Node Decoder this module is used to generate a new node of the graph, then update the memory of the feature fusion module with node information.

3.1 Query Encoder & Graph Encoder

For fair comparisons with the previous work (He et al., 2020), our query encoder and graph encoder are based on the vanilla transformer (Vaswani et al., 2017), which consists of multi-head self attention (MSA) and position-wise feed-forward network (FFN) blocks. The FFN contains two layers with a ReLU non-linearity. Layer normalization (Ba et al. 2016) is applied before every block, and residual connections (He et al., 2016) after every block.

Formally, given an input query Q with n tokens, each token embedding is randomly initialized and positional encoding is added to the token embedding to retain positional information. The resulted embeddings are denotes as $\mathbf{x} = \{x_0, x_1, ..., x_n\}$. Similar to BERT (Devlin et al., 2019), a special token is appended to the query as x_0 for sentence encoding. Transformations in the query encoder can be denoted as:

$$\mathbf{x}^{l'} = LN(MSA(\mathbf{x}^{l-1}) + \mathbf{x}^{l-1}), \quad (1)$$

$$\mathbf{x}^{l} = LN(FFN(\mathbf{x}^{l'}) + \mathbf{x}^{l'}). \quad (2)$$

After stacking L blocks, we obtained the contextualized token representations from the query encoder, denoted as $\{x_0^L, x_1^L, ..., x_n^L\}$. The first vector x_0 is treated as the sentence-level representation of the query and will be used as the initial state during expansion. For clarity, we denote the vectors as $\mathbf{x} \in \mathbb{R}^{(n+1)\times d}$, where d is the dimension.

As for the graph encoder, we treat the input graph as a sequence of nodes in the chronological order of when they are inserted into the graph as discussed in Section 2. Formally, given the graph G_t at the time step t, we take its node sequence $\{o_1, o_2, ..., o_{t-1}\}$ as the input. A transformer architecture is also applied to obtain the contextualized node embeddings. Notice that the contextualized representation of the graph is constructed incrementally as the expanding progress. Therefore, we apply the vanilla transformer with masked self-attention as the graph encoder, which only allows each position in the node sequence to attend to all positions up to and including that position. For brevity, we denoted the resulted contextualized node representations as $\mathbf{y} \in \mathbb{R}^{m\times d}$.

3.2 Feature Fusion

Unlike the conventional sequence-to-sequence model that only has one encoder, our model contains two encoders. Previous work (He et al., 2020) proposed to use gating mechanism and cross attention to combine the representations of resulted representations from query and graph encoders. We choose to use vanilla multi-head attention mechanism (Vaswani et al., 2017) to fuse the features from these encoders. Formally, at each time step t, the feature fusion component combines the query and graph representations for gradually locating and collecting the most relevant information for the next expansion:

$$z_t^l = LN(MSA(h_t^{l-1}, \mathbf{x}) + h_t^{l-1}), \quad (3)$$

$$z_t^{l'} = LN(MSA(z_t^l, \mathbf{y}) + z_t^l), \quad (4)$$

$$h_t^l = LN(FFN(z_t^{l'}) + z_t^{l'}). \quad (5)$$

The initial expansion state of h_t^0 is initialized with x_0. For clarity, we denote the last hidden state h_t^L as h_t, which is the expansion state at the time step t. We now proceed to present the details of each decision stage of one expansion step.

3.3 Edge Decoder

At the t-th time step, the edge decoder takes the expansion state h_t from the feature fusion module and the contextualized representation $\mathbf{y}$ from the graph encoder as the inputs, and predicts which nodes in the current graph should be attached to the new node. Inspired by Cai and Lam (2019) and Cai and Lam (2020a), we leverage multi-head attention and take the maximum over different heads as the final edge probabilities. Formally, given h_t and $\mathbf{y}$, a set of attention weights can be obtained by using multi-head attention mechanism: $\{\alpha_t^{g_i}\}_{i=1}^k$, where k is the number of attention heads and $\alpha_t^{g_i}$ is the i-th probability vector. The probability of the edge between the new node and the node o_j is then computed by $\alpha_t^g = max_i(\alpha_t^{g_i})$. Intuitively, each head is in charge of a set of possible relations (though not explicitly specified). The maximum pooling reflects that the edge should be built once one relation is activated.

Finally, the edge decoder passes the edge information to the feature fusion module by updating the expansion state h_{t+1} as follows:

$$h_{t+1} = LN(MSA(h_t, \mathbf{y}) + h_t). \quad (6)$$

3.4 Node Decoder

The node decoder needs to look at the input query and determine which tokens are the most important ones. This choice is a weighted matrix that gives an attention probability between each token in the query and generated nodes in the target graph. Concretely, a single-head attention α_t^s is computed based on the state h_t and the sentence representation $s_{1:n}$, where α_t^s denotes the attention weight of the word w_i in the current time step. This component then updates the parser state with the alignment information via the following equation:

$$h_{t+1} = LN(MSA(h_t, \mathbf{x}) + h_t). \quad (7)$$

We then compute the probability distribution of the new node through a hybrid of two channels. The new node can either be a DELETE node or a token copied from the input query. First, h_t is fed through a $softmax$ to obtain a probability distribution over a pre-defined vocabulary, which contains the DELETE node and other dummy nodes such as EOS. The probability of the new node is calculated as $P^{vocab} = softmax(W^{vocab} h_t + b^{vocab})$.

Second, we used the attention scores α_t^s as the probability to copy a token from the input query as a node label similar to the copy mechanism (Gu et al., 2016; See et al., 2017). Therefore, the final prediction probability of a node o is defined as:

$$P(o) = p_{gen} \cdot P_{vocab}(o) + p_{copy} \cdot \sum_{i \in T(c)} \alpha_t^s[i], \quad (8)$$

where $[i]$ indexes the i-th element, and $T(c)$ are index sets of tokens respectively that have the surface form as o. $P(gen)$ and $P(copy)$ are the probabilities of generating and copying a node, respectively. They are computed by using a single layer neural network with $softmax$ activation as:

$$[p_{gen}, p_{copy}] = softmax(W^{gate} h_t). \quad (9)$$

The whole expanding procedure is terminated if the newly generated node is the special node EOS.

Statistics	User Generated	MSCOCO	GCC	RSICD
Splits	30/1/1	196/2/2	400/7/7	8/1/1
Avg. Source Nodes	2.0	2.9	3.8	5.9
Avg. Target Nodes	2.0	2.9	3.7	5.9
Avg. Source Edges	1.0	1.9	2.8	3.7
Avg. Targe Edges	1.0	1.9	2.8	3.6
OOV Nodes	10	4	3	12
OOV Edges	8	4	4	8

Table 1: Statistics of four SGM datasets.

4 Dataset Construction

Existing SGM datasets are synthetically constructed based on scene graphs from MSCOCO (Lin et al., 2014) and GCC (Sharma et al., 2018a), and via crowd sourcing. To construct scene graphs, He et al. (2020) used an in-house scene graph parser to parse a random subset of MSCOCO description data and GCC captions, thus the constructed scene graph is relatively simple. In Table 1, the average numbers of nodes and edges for each graph are limited to 2.9 and 1.9 respectively. GCC is more complicated than MSCOCO with a larger graph, but the percentage of nodes and edges from the development/test set that does not appear in the training set (OOV Nodes, OOV Edges) are still low, which will cause the model easily overfit to the dataset. To verify the generalization ability and the scalability of the model to handle more complex scene graphs, we constructed our own Scene Graph Modification dataset based on the Remote Sensing Image Captioning Dataset (RSICD) (Lu et al., 2017) in the remote sensing field for remote sensing image captioning task.

Inspired by the modification methods proposed by He et al. (2020). First, we adopt the parser (Schuster et al., 2015) to parse the caption for each graph and generate the original scene graph **x**. Then we define three types of graph modification operations $\mathcal{A}$ = {INSERT, DELETE, SUBSTITUTE}, and randomly apply them to the original scene graph to generate query (**q**) and modified scene graph (**y**). The data in RSICD consists of the triples (**x, y, q**).[2]

Compared with the existing SGM dataset, each graph of RSICD has more nodes and edges, with an average of 5.9 and 3.7 on the training/development/test set, which is almost twice that of User Generated and MSCOCO. In addition, the dataset comes from the field of remote sensing. Due to the large

[2]We give three detailed operations and examples in the Appendix A.1.

Models	User Generated			MSCOCO			GCC		
	Node F1	Edge F1	GAcc	Node F1	Edge F1	GAcc	Node F1	Edge F1	GAcc
CopyGraph (He et al., 2020)	66.17	31.42	—	78.41	64.62	—	79.46	66.32	—
Text2Text (He et al., 2020)	78.59	52.68	52.15	91.47	72.74	64.42	—	—	—
GRNN (You et al., 2018)	80.68	57.17	56.75	80.64	55.76	50.72	—	—	—
DCGCN (Guo et al., 2019)	79.05	54.23	52.67	89.08	72.47	68.89	—	—	—
GTran (Cai and Lam, 2020b)	81.47	59.43	58.23	91.21	75.68	71.38	—	—	—
STran (He et al., 2020)	83.69	62.10	60.90	95.40	86.52	82.97	93.84	57.68	52.50
EGraph (Weber et al., 2021)	97.62	88.26	87.60	99.52	98.40	96.15	98.62	91.64	75.01
ISE	**98.74**±0.12	**91.37**±0.14	**89.41**±0.47	**99.68**±0.14	**98.96**±0.21	**97.26**±0.37	**99.53**±0.13	**93.06**±0.22	**76.34**±0.47
ISE (w/o BERT)	94.39±0.11	79.53±0.18	75.72±0.47	98.17±0.13	97.25±0.14	89.61±0.45	96.91±0.16	85.50±0.21	58.40±0.56

Table 2: Results of User Generated, MSCOCO and GCC datasets. GAcc denotes the graph-level accuracy. Both of our models are statistically significantly outperform (p<0.0001) previous best-reported model (Weber et al., 2021).

number of geographical terms, the OOV Nodes of the development/test sets compared with the training set reach 12%/11%, and the OOV Edges reach 8%/8%, which are much higher than the MSCOCO and GCC datasets. Considering the complexity of RSICD, we construct it apart from User Generated, MSCOCO and GCC to further analysis the generalization and scalability of ISE.

5 Experiments and Analyses

5.1 Data

We evaluated our model on four benchmarks, including User Generated, MSCOCO and GCC proposed by He et al. (2020), and RSICD dataset proposed in this work. MSCOCO, GCC and RSICD are constructed synthetically from publicly available datasets (Lin et al., 2014; Sharma et al., 2018b; Lu et al., 2017), while the User Generated dataset is created via crowd sourcing. Detailed statistics of datasets are shown in Table 1.

5.2 Setup

For fair comparisons, we used the same data splits for User Generated, MSCOCO and GCC datasets as in Weber et al. (2021). For RSICD, we randomly split the data into 8K/1K/1K for training/development/test. Following Weber et al. (2021), we use three automatic metrics for the evaluation, including node-level and edge-level F1 score, and graph-level accuracy. Graph-level accuracy is computed based on exact string match, which requires the generated scene graph to be identical to the target scene graph for a correct prediction. We reported the mean score and standard deviation by using 5 models from independent runs. We refer to the Appendix A.2 for the detailed implementation.

5.3 Baselines

For comprehensive comparisons, we include six baselines as follows. Except for the CopyGraph, all of them aim to rebuild the target scene graph.

CopyGraph This baseline directly copies the source scene graph as the target scene graph, which can be viewed as the lower bound.

Text2Text This baseline is introduced by He et al. (2020). They used the standard sequence-to-sequence architecture by linearizing the scene graph based on depth-first search.

GRNN Graph RNN (You et al., 2018) is used as the graph encoder and edge decoder. Specifically, the edges are represented by an adjacency matrix, which is then generated in an auto-regressive manner. Both the query encoder and node decoder are based on Gated Recurrent Units (Cho et al., 2014).

DCGCN Densely-Connected Graph Convolutional Networks (Guo et al., 2019) are used as the graph encoder. Other components are kept the same as the GRNN.

GTran Graph Transformer (Cai and Lam, 2020b) is used as the graph encoder, while other modules are the same as GRNN and DCGCN.

STran The sparsely-connected transformer (He et al., 2020) is used to encode the source graph. In addition, a cross-attention mechanism is applied to fuse the features from graph encoder and query encoder. Node decoder and edge decoder are the same as GRNN.

EGraph This is the state-of-the-art model on graph modification task. Concretely, Weber et al. (2021) considerably increases performance on the graph modification by phrasing it as a sequence labelling task.

Models	RSICD		
	Node F1	Edge F1	GAcc
CopyGraph	66.35	58.68	—
EGraph (Weber et al., 2021)	72.09±0.12	53.96±0.31	23.93±0.74
ISE	**81.78**±0.13	**67.01**±0.25	**44.20**±0.59

Table 3: Results of the RSICD dataset. Results of STran is reproduced from the implementation of Weber et al. (2021). Both of our models are statistically significantly outperform (p<0.0001) previous best-reported model (Weber et al., 2021).

Datast/Model		Node F1	Edge F1	GAcc
User Generated	ISE	94.58	79.61	76.23
	ISE Rebuild	82.97	66.74	63.47
	ISE - Copy	88.38	74.95	71.64
MSCOCO	ISE	98.60	97.99	92.24
	ISE Rebuild	92.90	87.61	83.23
	ISE - Copy	95.58	91.87	88.09
GCC	ISE	96.87	85.50	58.90
	ISE Rebuild	89.48	62.59	51.67
	ISE - Copy	92.74	76.89	53.91

Table 4: An ablation study for ISE. Rebuild denotes that we regenerate the scene graph rather than extend it. - Copy denotes model without using the copy mechanism

5.4 Main Results

According to Table 2, our proposed approach (ISE) significantly outperforms the state-of-the-art model (Weber et al., 2021) on three datasets. Specifically, ISE outperforms EGraph 1.81, 1.11 and 1.33 percentage points in terms of graph accuracy on User Generated, MSCOCO and GCC datasets, respectively. We observe that the improvement is especially prominent on the User Generated dataset, which is more challenging than the other two synthetic datasets in terms of the diversity in graph semantics and natural language expressions. All baseline models suffer from performance degradation as it is much harder to rebuild the entire target scene graph on this dataset. On the other hand, ISE constructs the target scene graph by incrementally expanding the source scene graph without changing the unmodified structure. We believe this formulation is able to effectively cope with this difficulty.

We also observe that both EGraph and ISE achieve lower graph accuracy on the GCC dataset. The main reason is the difficulty of predicting the correct edges between generated nodes. For example, EGraph achieves 98.62 Node F1 score on GCC, higher than 97.62 Node F1 score on the User Generated dataset. However, EGraph only achieves 75.01 Edge F1 score on GCC, while it can attain 88.26 Edge F1 score on User Generated. Our proposed model has larger improvements upon EGraph in terms of Edge F1 score on the same dataset (93.06 vs. 91.64). We attribute this stronger improvement to iterations between nodes prediction and edge prediction, which allows more accurate and harmonious expansion decisions progressively. On the other hand, EGraph predicts nodes and edges at two independent stages. Such an approach may lead to the lack of the modeling capability of interactions between node prediction and edge prediction.

We further compare our model with EGraph on the newly constructed dataset RSICD as shown in Table 3. ISE is able to achieve a graph accuracy of 44.20% and improves upon the EGraph model by 21 percentage points. However, the graph accuracy of all the models is much lower than the one attained on the previous three SGM datasets. One reason is that RSICD has more complex queries paired with larger scene graph, which brings a challenge to existing models. The RSICD dataset also suffers from the data sparsity issue where many words (39%) and nodes (42%) only appear once in the training data. Incorrect node prediction will further propagate the errors to edge prediction. Our iterative node and edge prediction paradigm help to alleviate this issue. Specifically, ISE only outperforms EGraph 9.69 percentage points on Node F1 score, while the improvement on Edge F1 score is 13.05%. Therefore, ISE is able to achieve a higher accuracy. In order to further address this data sparsity issue, one potential solution is transfer learning, where the model is pretrained on User Generated dataset first then fine-tuned on RSICD. However, this approach may suffer from a domain-shift problem, as RSICD is constructed based on the remote sensing domain. We leave this direction as future works.

5.5 Analysis and Discussion

In this section, we provided a fine-grained analysis of our proposed model. We reported all the results on the development set by using the ISE model without contextualized embeddings from BERT.

Ablation Study As shown in Table 4, we examine the contributions of two main components used in our model. The first one is the incremental structure expanding. We use the same model architecture but try to rebuild the target scene graph

% of Training Set		Node F1	Edge F1	GAcc
20%	STran	87.92	71.35	68.15
	ISE	**95.46**	**92.22**	**79.12**
40%	STran	93.94	81.11	78.55
	ISE	**97.50**	**96.12**	**88.64**
60%	STran	95.32	82.70	80.65
	ISE	**98.09**	**97.17**	**89.29**
80%	STran	95.92	86.36	83.90
	ISE	**98.37**	**97.48**	**90.69**
100%	STran	96.24	87.88	85.20
	ISE	**98.60**	**97.99**	**92.24**

Table 5: Comparison of STran and ISE against different training data sizes on the dev set of MSCOCO. Results of STran are reproduced from He et al. (2020).

Query Length		Node F1	Edge F1	Graph Acc
<5	STran	51.68	91.38	40.98
	ISE	**96.02**	**92.36**	**57.38**
5∼10	STran	92.73	57.01	50.39
	ISE	**97.32**	**86.84**	**60.18**
≥10	STran	91.38	51.68	40.98
	ISE	**98.42**	**84.62**	**58.64**

Table 6: Comparison of STran and ISE against different lengths of queries.

similar to previous efforts. We can observe significant drops on three SGM datasets, which further confirms the effectiveness of the extending strategy. The second one is the copy mechanism, which directly copies the token from the query as nodes in the target scene graph. It plays a significant role in predicting nodes especially when the training data is limited (User Generated).

Performance against Training Data Size Table 5 shows the performance of STran and ISE against different training settings on MSCOCO dataset. We considered four training settings (20%, 40%, 60%, 80%, 100% training data). ISE consistently outperforms STran under the same amount of training data. When the size of training data decreases, we can observe that the performance gap becomes more obvious. Particularly, using 40% of the training data, ISE is able to achieve a graph accuracy of 88.64%, higher than STran trained on the whole dataset. These results demonstrate that our model is more effective in terms of using training resources and more robust when the training data is limited.

Graph Size		Node F1	Edge F1	Graph Acc
<5	STran	94.04	62.77	58.74
	ISE	**96.73**	**81.25**	**62.54**
5∼10	STran	91.61	51.40	37.73
	ISE	**97.22**	**88.82**	**49.06**
≥10	STran	79.12	30.13	24.62
	ISE	**95.44**	**90.95**	**35.38**

Table 7: Comparison of STran and ISE against different target scene graph sizes.

Performance against Query Length Table 6 shows the results of STran and ISE under different query lengths on GCC dataset. We partitioned the sentence length into three classes (<5, [5, 10), ≥10). In general, ISE outperforms STran against various sentence lengths. When the length of the query increases, we can observe that the performance gap becomes more obvious in terms of graph accuracy. Intuitively, with the increase of the query length, it is more challenging for the model to comprehend the sentence. This suggests that ISE is able to handle more complex instructions.

Performance against Graph Size Table 7 shows the results of STran and ISE against different target scene graph sizes on GCC dataset. We partitioned the scene into three classes (<5, [5, 10), ≥10). Based on the formulation of extending the source scene graph, our model is required to deal with larger graphs. For example, deleting a node in the scene graph becomes adding a special "Delete" node in the extended graph. However, ISE is able to consistently outperform STran against various target graph sizes, even when the target scene graph is large. This result suggests the superiority of the proposed formulation.[3]

Case Study We give two cases in Figure 4. STran generates scene graph from the scratch conditioned on the original graph and query may lead to the lack of the modeling capability of interactions between node prediction and edge prediction. For example, in Figure 4 (a), STran omitted the attribute: "Velvet" during the node prediction. In addition, during the edge prediction, STran redundantly generated the relation: "Of" in Figure 4 (b). However, these structures do not need to be modified in the source scene graph. ISE can infer more accurate target graph by incrementally expanding the source graph without changing the unmodified structure.

[3]We give an error analysis in the Appendix A.4.

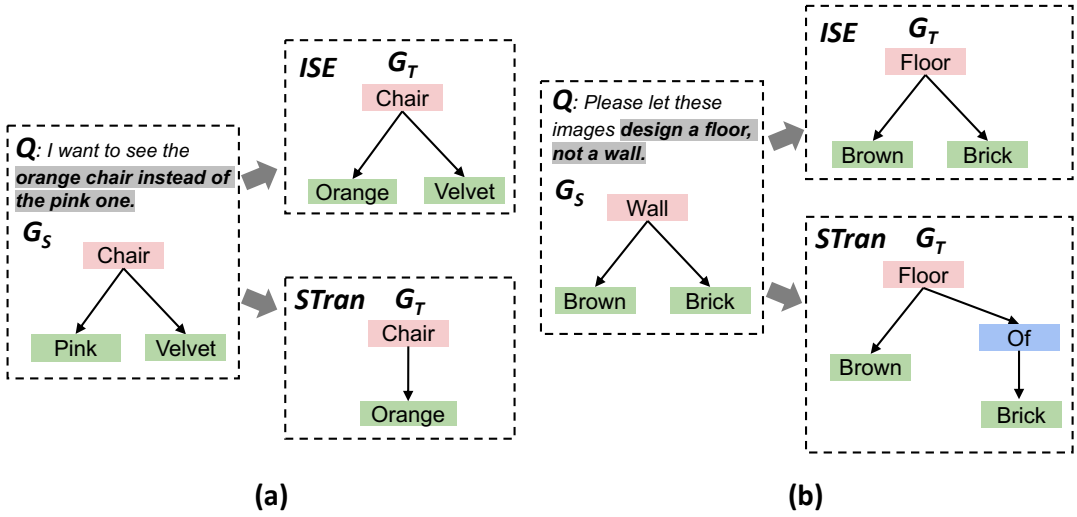

Figure 4: Two cases of STran and ISE for scene graph modification on User Generated. Q denotes the textual query, G_S denotes the source scene graph, G_T denotes the target scene graph generated by STran and ISE.

6 Related Work

We refer to the Appendix A.3 for the detailed related work of scene graph. Scene graph builds a bridge between image domain and text domain. Vision and natural language are all tremendously promoted by studying into scene graphs. Recently, scene graph modification becomes an emerging research direction. Chen et al. (2020) proposed a framework based on scene graph editing for text-based image retrieval. On the other hand, He et al. (2020) took the scene graph and the textual query as inputs and modified the source graph according to the query. They viewed the task as conditional graph generation, which is further decomposed into node prediction and edge prediction. For node prediction, all the nodes in the target scene graph is generated based on a graph-to-sequence model with dual encoder (Song et al., 2018; Beck et al., 2018; Zhang et al., 2020), then a graph RNN is adopted to predict the edges between generated nodes (You et al., 2018). More recently, Weber et al. (2021) developed an alternative formulation of this problem in which they model the modification as an auto-regressive sequence labelling task.

Instead of rebuilding the entire target graph, we framed the scene graph modification task as incremental graph expansion. This formulation is related to incremental parsing, where a sentence is scanned from left-to-right and the structured is built incrementally by inserting a node or attaching an edge. Incremental parsers are widely used in semantic parsing (Zhou et al., 2016; Cheng et al., 2017; Guo and Lu, 2018; Naseem et al., 2019; Liu et al., 2022a) and syntactic parsing (Huang and Sagae, 2010; Dyer et al., 2015; Liu and Zhang, 2017), as they are computationally efficient, and can use machine learning to predict actions based on partially generated structures. Our feature fusion module can be viewed as the parser state as it carries the structural information and serves as a writable memory during the expansion step. Unlike Weber et al. (2021) linearize the scene graph and label it in an auto-regressive manner, our model iterates between finding the relevant part in the query and reading the partially constructed scene graph, inferring more accurate and harmonious expansion decisions progressively.

7 Conclusion

In this paper, we designed a novel formulation for scene graph modification, which allows us to incrementally expand the source scene graph instead of rebuilding the entire graph. Based on the formalism, we further propose a model that is able to leverage the mutual causalities between node prediction and edge prediction. Experiments on three SGM benchmarks demonstrate the effectiveness. To test our model under a complex scenario, we constructed a more challenging dataset from the remote sensing domain, which has more modification operations based on the more complicated queries compared with existing SGM datasets. For future work, we would like to explore how to integrate the model into the text-based image retrieval task.

8 Acknowledgement

We thank the reviewers for their valuable comments. The work was supported by the National Key Research and Development Program of China (No. 2019YFB1704003), the National Nature Science Foundation of China (No. 62021002 and No. 71690231), NSF under grants III-1763325, III-1909323, III-2106758, SaTC-1930941, Tsinghua BNRist and Beijing Key Laboratory of Industrial Bigdata System and Application.

References

Iro Armeni, Zhi-Yang He, JunYoung Gwak, Amir R Zamir, Martin Fischer, Jitendra Malik, and Silvio Savarese. 2019. 3d scene graph: A structure for unified semantics, 3d space, and camera. In *ICCV*, pages 5664–5673.

Lei Jimmy Ba, Jamie Ryan Kiros, and Geoffrey E. Hinton. 2016. Layer normalization. *CoRR*, abs/1607.06450.

Daniel Beck, Gholamreza Haffari, and Trevor Cohn. 2018. Graph-to-sequence learning using gated graph neural networks. In *ACL*.

Deng Cai and Wai Lam. 2019. Core semantic first: A top-down approach for AMR parsing. In *EMNLP*.

Deng Cai and Wai Lam. 2020a. Amr parsing via graph-sequence iterative inference. In *ACL*.

Deng Cai and Wai Lam. 2020b. Graph transformer for graph-to-sequence learning. In *AAAI*, volume 34.

Lichang Chen, Guosheng Lin, Shijie Wang, and Qingyao Wu. 2020. Graph edit distance reward: Learning to edit scene graph. In *ECCV*.

Jianpeng Cheng, Siva Reddy, Vijay A. Saraswat, and Mirella Lapata. 2017. Learning structured natural language representations for semantic parsing. In *ACL*.

Kyunghyun Cho, Bart van Merrienboer, Çaglar Gülçehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. 2014. Learning phrase representations using RNN encoder-decoder for statistical machine translation. In *EMNLP*.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: pre-training of deep bidirectional transformers for language understanding. In *NAACL-HLT*.

Chris Dyer, Miguel Ballesteros, Wang Ling, Austin Matthews, and Noah A. Smith. 2015. Transition-based dependency parsing with stack long short-term memory. In *ACL*.

Matt Gardner, Joel Grus, Mark Neumann, Oyvind Tafjord, Pradeep Dasigi, Nelson F. Liu, Matthew Peters, Michael Schmitz, and Luke Zettlemoyer. 2018. AllenNLP: A deep semantic natural language processing platform. In *NLP-OSS*, Melbourne, Australia.

Jiatao Gu, Zhengdong Lu, Hang Li, and Victor O. K. Li. 2016. Incorporating copying mechanism in sequence-to-sequence learning. In *ACL*. The Association for Computer Linguistics.

Zhijiang Guo and Wei Lu. 2018. Better transition-based AMR parsing with refined search space. In *EMNLP*.

Zhijiang Guo, Yan Zhang, Zhiyang Teng, and Wei Lu. 2019. Densely connected graph convolutional networks for graph-to-sequence learning. *TACL*, 7:297–312.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. 2016. Deep residual learning for image recognition. In *CVPR*.

Xuanli He, Quan Hung Tran, Gholamreza Haffari, Walter Chang, Trung Bui, Zhe L. Lin, Franck Dernoncourt, and Nhan Dam. 2020. Scene graph modification based on natural language commands. In *EMNLP*.

Xuming Hu, Fukun Ma, Chenyao Liu, Chenwei Zhang, Lijie Wen, and Philip S Yu. 2021a. Semi-supervised relation extraction via incremental meta self-training. In *EMNLP: Findings*.

Xuming Hu, Lijie Wen, Yusong Xu, Chenwei Zhang, and S Yu Philip. 2020. Selfore: Self-supervised relational feature learning for open relation extraction. In *EMNLP*.

Xuming Hu, Chenwei Zhang, Yawen Yang, Xiaohe Li, Li Lin, Lijie Wen, and S Yu Philip. 2021b. Gradient imitation reinforcement learning for low resource relation extraction. In *EMNLP*.

Liang Huang and Kenji Sagae. 2010. Dynamic programming for linear-time incremental parsing. In *ACL*.

Justin Johnson, Ranjay Krishna, Michael Stark, Li-Jia Li, D. Shamma, Michael S. Bernstein, and Li Fei-Fei. 2015. Image retrieval using scene graphs. *CVPR*, pages 3668–3678.

Diederik P. Kingma and Jimmy Ba. 2015. Adam: A method for stochastic optimization. In *ICLR*.

Shu'ang Li, Xuming Hu, Li Lin, and Lijie Wen. 2022. Pair-level supervised contrastive learning for natural language inference. In *ICASSP*.

Xiaodan Liang, Lisa Lee, and Eric P Xing. 2017. Deep variation-structured reinforcement learning for visual relationship and attribute detection. In *CVPR*.

Tsung-Yi Lin, M. Maire, Serge J. Belongie, James Hays, P. Perona, D. Ramanan, Piotr Dollár, and C. L. Zitnick. 2014. Microsoft coco: Common objects in context. In *ECCV*.

Aiwei Liu, Xuming Hu, Li Lin, and Lijie Wen. 2022a. Semantic enhanced text-to-sql parsing via iteratively learning schema linking graph. In *KDD*.

Jiangming Liu and Yue Zhang. 2017. In-order transition-based constituent parsing. *TACL*, 5:413–424.

Shuliang Liu, Xuming Hu, Chenwei Zhang, Shu'ang Li, Lijie Wen, and Philip S. Yu. 2022b. Hiure: Hierarchical exemplar contrastive learning for unsupervised relation extraction. In *NAACL*.

Xiaochen Liu, Yu Bai, Jiawei Li, Yinan Hu, and Yang Gao. 2022c. Psp: Pre-trained soft prompts for few-shot abstractive summarization. *arXiv preprint arXiv:2204.04413*.

Xiaoqiang Lu, Binqiang Wang, Xiangtao Zheng, and Xuelong Li. 2017. Exploring models and data for remote sensing image caption generation. *IEEE Transactions on Geoscience and Remote Sensing*, 56(4):2183–2195.

Ramesh Manuvinakurike, Jacqueline Brixey, Trung Bui, Walter Chang, Doo Soon Kim, Ron Artstein, and Kallirroi Georgila. 2018. Edit me: A corpus and a framework for understanding natural language image editing. In *LREC*, Miyazaki, Japan.

Tahira Naseem, Abhishek Shah, Hui Wan, Radu Florian, S. Roukos, and Miguel Ballesteros. 2019. Rewarding smatch: Transition-based amr parsing with reinforcement learning. In *ACL*.

Joakim Nivre. 2004. Incrementality in deterministic dependency parsing.

Hang Qi, Yuanlu Xu, Tao Yuan, Tianfu Wu, and Song-Chun Zhu. 2018. Scene-centric joint parsing of cross-view videos. In *AAAI*.

Sahana Ramnath, Amrita Saha, Soumen Chakrabarti, and Mitesh M. Khapra. 2019. Scene graph based image retrieval - a case study on the clevr dataset. *ArXiv*, abs/1911.00850.

Pengzhen Ren, Yun Xiao, Xiaojun Chang, Po-Yao Huang, Zhihui Li, Xiaojiang Chen, and Xin Wang. 2021. A comprehensive survey of neural architecture search: Challenges and solutions. *ACM Computing Surveys (CSUR)*, 54(4):1–34.

Brigit Schroeder and Subarna Tripathi. 2020. Structured query-based image retrieval using scene graphs. *CVPRW*, pages 680–684.

Sebastian Schuster, Ranjay Krishna, Angel X. Chang, Li Fei-Fei, and Christopher D. Manning. 2015. Generating semantically precise scene graphs from textual descriptions for improved image retrieval. In *VL@EMNLP*.

Abigail See, Peter J. Liu, and Christopher D. Manning. 2017. Get to the point: Summarization with pointer-generator networks. In *ACL*.

Piyush Sharma, Nan Ding, Sebastian Goodman, and Radu Soricut. 2018a. Conceptual captions: A cleaned, hypernymed, image alt-text dataset for automatic image captioning. In *ACL*.

Piyush Sharma, Nan Ding, Sebastian Goodman, and Radu Soricut. 2018b. Conceptual captions: A cleaned, hypernymed, image alt-text dataset for automatic image captioning. In *ACL*.

Jiaxin Shi, Hanwang Zhang, and Juanzi Li. 2019. Explainable and explicit visual reasoning over scene graphs. In *CVPR*.

Linfeng Song, Yue Zhang, Zhiguo Wang, and D. Gildea. 2018. A graph-to-sequence model for amr-to-text generation. In *ACL*.

Nitish Srivastava, Geoffrey E. Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. 2014. Dropout: a simple way to prevent neural networks from overfitting. *J. Mach. Learn. Res.*, 15(1):1929–1958.

Damien Teney, Lingqiao Liu, and Anton van den Hengel. 2017. Graph-structured representations for visual question answering. In *CVPR*, pages 3233–3241. IEEE Computer Society.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *NeurIPS*.

Peng Wang, Qi Wu, Jiewei Cao, Chunhua Shen, Lianli Gao, and Anton van den Hengel. Neighbourhood watch: Referring expression comprehension via language-guided graph attention networks. In *CVPR*.

Ruize Wang, Zhongyu Wei, Piji Li, Qi Zhang, and Xuanjing Huang. 2020a. Storytelling from an image stream using scene graphs. In *AAAI*, volume 34, pages 9185–9192.

Sijin Wang, Ruiping Wang, Ziwei Yao, S. Shan, and Xilin Chen. 2020b. Cross-modal scene graph matching for relationship-aware image-text retrieval. *WACV*, pages 1497–1506.

Yu-Siang Wang, Chenxi Liu, Xiaohui Zeng, and Alan L. Yuille. 2018. Scene graph parsing as dependency parsing. In *NAACL-HLT*, pages 397–407. Association for Computational Linguistics.

Leon Weber, Jannes Münchmeyer, Samuele Garda, and Ulf Leser. 2021. Extend, don't rebuild: Phrasing conditional graph modification as autoregressive sequence labelling. In *EMNLP*.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Rémi Louf, Morgan Funtowicz, Joe Davison, Sam Shleifer, Patrick von Platen, Clara Ma, Yacine Jernite, Julien Plu, Canwen Xu, Teven Le Scao, Sylvain Gugger, Mariama Drame, Quentin

Lhoest, and Alexander M. Rush. 2020. Transformers: State-of-the-art natural language processing. In *EMNLP*, Online.

Danfei Xu, Yuke Zhu, Christopher B. Choy, and Li Fei-Fei. 2017. Scene graph generation by iterative message passing. In *CVPR*, pages 3097–3106. IEEE Computer Society.

Xu Yang, Kaihua Tang, Hanwang Zhang, and Jianfei Cai. 2019. Auto-encoding scene graphs for image captioning. In *CVPR*, pages 10685–10694. Computer Vision Foundation / IEEE.

Ting Yao, Yingwei Pan, Yehao Li, and Tao Mei. 2018. Exploring visual relationship for image captioning. In *ECCV*.

Jiaxuan You, Rex Ying, Xiang Ren, William L. Hamilton, and J. Leskovec. 2018. Graphrnn: Generating realistic graphs with deep auto-regressive models. In *ICML*.

Rowan Zellers, Mark Yatskar, Sam Thomson, and Yejin Choi. 2018. Neural motifs: Scene graph parsing with global context. In *CVPR*, pages 5831–5840. IEEE Computer Society.

Xin Zhang, Guangwei Xu, Yueheng Sun, Meishan Zhang, Xiaobin Wang, and Min Zhang. 2022. Identifying chinese opinion expressions with extremely-noisy crowdsourcing annotations. In *ACL*.

Xin Zhang, Guangwei Xu, Yueheng Sun, Meishan Zhang, and Pengjun Xie. 2021. Crowdsourcing learning as domain adaptation: A case study on named entity recognition. In *ACL*.

Yan Zhang, Zhijiang Guo, Zhiyang Teng, Wei Lu, Shay B. Cohen, Zuozhu Liu, and Lidong Bing. 2020. Lightweight, dynamic graph convolutional networks for amr-to-text generation. In *EMNLP*.

Junsheng Zhou, Feiyu Xu, Hans Uszkoreit, Weiguang Qu, Ran Li, and Yanhui Gu. 2016. AMR parsing with an incremental joint model. In *EMNLP*.

Graph Modification Operation: `DELETE`
Original Scene Graph: Some *trees* are in a medium residential area.
Query: Remove *trees*.
Modified Scene Graph: Some are in a medium residential area.

Graph Modification Operation: `INSERT`
Original Scene Graph: A bridge built on a river.
Query: Show me a *red* bridge.
Modified Scene Graph:A *red* bridge built on a river.

Graph Modification Operation: `SUBSTITUTE`
Original Scene Graph: Some gray and *green* mountains are together.
Query: I prefer *red* to *green*, modify *red* to *green*.
Modified Scene Graph: Some gray and *red* mountains are together.

Table 8: Examples on the three types of graph modification operations $\mathcal{A}$ = {INSERT, DELETE, SUBSTITUTE}

A Appendix

A.1 Operations in RSICD

We introduce three operations in RSICD in details:

- **`DELETE`**: The original scece graph is **x**. We randomly select a node **o** in **x**, and delete it both with related edges. The deleted graph is defined as **y**. We choose a random sentence from the *DELETE Template* (Manuvinakurike et al., 2018), for example, " I do not want **." We replace ** with **o** to get modification operation **q**.

- **`INSERT`**: It is the reverse process of **`DELETE`**. The graph before deleting the node is regarded as **y**, and the corresponding graph after deletion is treated as **x**. The modification operation is randomly selected from the *INSERT Template* (Manuvinakurike et al., 2018), for example, " Show me **." We replace ** with **o** to obtain query **q**.

- **`SUBSTITUTE`**: We randomly select a node **o**, use the AllenNLP toolkit (Gardner et al., 2018) to find the three most similar semantics nodes compared with **o**. We randomly choose a node **m**, and select a sentence from the *SUBSTITUTE Template* (Manuvinakurike et al., 2018), for example, " I prefer @@ to **, modify ** to @@." We replace ** and @@ with **o** and **m**, and get modification operation **q**. Note that SUBSTITUTE operation could be viewed as DELETE the node **o** first and then INSERT the node **m**, or vice versa.

In Table 8, we give the simple examples in RSICD to better understand three types of graph modification operations.

Embeddings	
concept	300
word	300
relation	100
Query Encoder	
transformer layers	4
Graph Encoder	
transformer layers	2
Feature Fusion	
heads	8
hidden size	512
feed-forward hidden size	1024
Node Decoder/ Edge Decoder	
heads	8
feed-forward hidden size	1024

Table 9: Hyper-parameters settings for ISE.

A.2 Implementation Details

Hyper-parameters of the model are tuned on the development set. All transformer (Vaswani et al., 2017) layers share the same hyper-parameter settings. Following He et al. (2020), we randomly initialized the word and node embeddings. We also report results with contextualized embeddings from BERT (Devlin et al., 2019). Specifically, we used the BERT-base-uncased implemented by (Wolf et al., 2020). The parameters in BERT are fixed during training. To mitigate over-fitting, we apply dropout (Srivastava et al., 2014) with the drop rate 0.2 between different layers. Following Cai and Lam (2020a), we use a special UNK token to replace the out-of-vocabulary lemmas of the input query and remove the UNK token in the generated graph. Parameter optimization is performed with the ADAM optimizer (Kingma and Ba, 2015) with $\beta_1 = 0.9$ and $\beta_2 = 0.999$. The learning rate schedule is similar to that in Vaswani et al. (2017), where warm-up steps being set to 2K. We used early stopping on the development set for choosing the best model. Please refer to Table 9 for the detailed hyper-parameters settings for ISE.

A.3 Scene Graph and Application

Deep learning has significantly promoted the advancement of computer vision (Liang et al., 2017; Ren et al., 2021). Simple visual understanding tasks such as object detection and recognition are no longer sufficient. To depict the relationship between objects in the scene as a driving force,

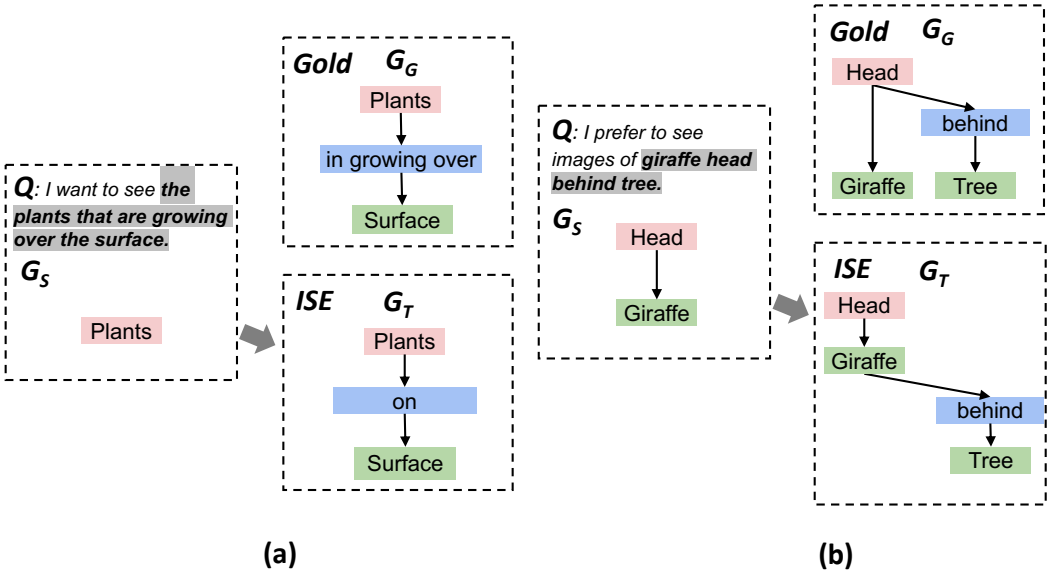

Figure 5: Two errors of ISE for scene graph modification on User Generated. Q denotes the textual query, G_S denotes the source scene graph, G_T denotes the target scene graph generated by ISE. G_G denotes the gold target scene graph.

higher-level visual understanding and reasoning skills are frequently necessary. Scene graphs were created specifically to address this issue. Scene graph was first proposed by Johnson et al. (2015) for image retrieval, which describes objects, their attributes, and relationships in images with a graph. A complete scene graph could represent the semantics of a dataset's scenes, not just a single image or video; additionally, it contains powerful representations that encode 2D/3D images (Johnson et al., 2015; Armeni et al., 2019), and videos (Qi et al., 2018; Wang et al., 2020a) into their abstract semantic elements. Scene graph is beneficial for various downstream tasks, such as information extraction (Hu et al., 2020, 2021a,b; Liu et al., 2022b), natural language summarization (Liu et al., 2022c), and natural language inference (Li et al., 2022).

Following the graph representation paradigm, different methods have been proposed to generate scene graphs from images (Xu et al., 2017; Wang et al., 2018; Zellers et al., 2018). Many cross-modal tasks that require understanding and reasoning on image and text are able to benefit from incorporating scene graphs, such as visual question answering (Teney et al., 2017; Shi et al., 2019), grounding referring expressions (Wang et al.), image captioning (Yang et al., 2019; Yao et al., 2018), and image retrieval (Wang et al., 2020b; Schroeder and Tripathi, 2020).

A.4 Error Analysis

We give two wrong scene graphs generated by ISE in Figure 5. We can observe in Figure 5 (a) that although ISE successfully predicts the need to insert a relation between object "Plants" and attribute "Surface", since the User Generated dataset contains a total of 2078 relations and the relations have serious long-tail effects. It is difficult for ISE to learn sparseness relations with few occurrences, leading to incorrectly predicting relation "in growing over" as "on". We attempt to address the long-tail effects of relations in future work. Since a node can be attached to multiple nodes, when Edge Decoder determines which nodes in the current graph should be attached to the new node, a common error is predicting the wrong node that needs to be attached. As shown in Figure 5 (b), ISE incorrectly connects relation "behind" between "Giraffe" and "Tree" instead of "Head" and "Tree".

Overcoming Language Priors in Visual Question Answering via Distinguishing Superficially Similar Instances

Yike Wu[1] **Yu Zhao**[1] **Shiwan Zhao** **Ying Zhang**[1]*
Xiaojie Yuan[1] **Guoqing Zhao**[2] **Ning Jiang**[2]
[1] Nankai University, Tianjin, China [2] Mashang Consumer Finance Co, Ltd
wuyike@nankai.edu.cn, zhaoyu@dbis.nankai.edu.cn
zhaosw@gmail.com, {yingzhang,yuanxj}@nankai.edu.cn

Abstract

Despite the great progress of Visual Question Answering (VQA), current VQA models heavily rely on the superficial correlation between the question type and its corresponding frequent answers (i.e., language priors) to make predictions, without really understanding the input. In this work, we define the training instances with the same question type but different answers as *superficially similar instances*, and attribute the language priors to the confusion of VQA model on such instances. To solve this problem, we propose a novel training framework that explicitly encourages the VQA model to distinguish between the superficially similar instances. Specifically, for each training instance, we first construct a set that contains its superficially similar counterparts. Then we exploit the proposed distinguishing module to increase the distance between the instance and its counterparts in the answer space. In this way, the VQA model is forced to further focus on the other parts of the input beyond the question type, which helps to overcome the language priors. Experimental results show that our method achieves the state-of-the-art performance on VQA-CP v2. Codes are available at Distinguishing-VQA.

1 Introduction

Recent years have witnessed great progress in VQA based on deep learning. However, some researchers reveal that most existing VQA models heavily rely on the superficial correlation between the question type and its corresponding frequent answers to make predictions, instead of really understanding the input (Agrawal et al., 2016, 2018; Goyal et al., 2017; Zhang et al., 2016). For example, once a VQA model detects that a question begins with "how many", it tends to blindly output the most common answer "2" to the "how many" questions in the training data without looking at other parts of the input.

(a) An example of superficially similar instances

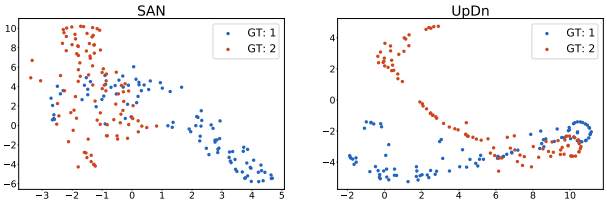

(b) Superficially similar instances in the answer space

Figure 1: (a) displays an example of *superficially similar instances*, which have the same question type ("How many") but different ground-truth answers ("1" and "2"). (b) visualizes the projection of such instances in the answer space obtained by SAN (Yang et al., 2016) and UpDn (Anderson et al., 2018) respectively. We observe that both models could not distinguish well between the mixed superficially similar instances.

We argue that the language priors arise from that the VQA model only captures the question type and ignores other information in the input image-question pair. On the one hand, the datasets which they are built upon are usually biased. For the questions of the same question type, the distribution of answers is severely biased to some frequent answers. On the other hand, deep learning methods tend to memorize some simple and salient patterns (e.g., frequency) in the training data, and easily exploit a shortcut to make predictions. Therefore, given an instance for testing, they prefer to directly look at the question type and leverage the superficial correlation rather than also analyze other information and further understand the whole input.

For solving the above problem, we introduce the concept of *superficially similar instances*, and propose to overcome the language priors via distinguishing between such instances. As shown in

*Corresponding author.

Proceedings of the 29th International Conference on Computational Linguistics, pages 5721–5729
October 12–17, 2022.

Figure 1a, the superficially similar instances refer to the instances that have the same question type but different answers. By distinguishing such instances, the VQA model is forced to focus on other information besides the question type in the input image-question pair when making predictions.

Unfortunately, it is not trivial to perform the distinguishing task, since a classical VQA paradigm does not have such a mechanism to support this. Given a training instance, the paradigm usually adopts the (binary) cross-entropy loss to reduce the distance between the projection of image-question pair in the answer space and the ground-truth, without explicitly encouraging the differentiation of superficially similar instances. As shown in Figure 1b, the superficially similar instances are mixed up in the answer space, which indicates that both two widely-used VQA models (Anderson et al., 2018; Yang et al., 2016) could not distinguish between them well.

In this work, we propose a novel training framework that explicitly encourages the VQA model to distinguish between the superficially similar instances. Specifically, for each training instance, we first construct a superficially similar set. The set consists of two kinds of superficially similar counterparts for the instance, which are collected in different ways and complement each other. Then we exploit the designed *distinguishing module* to increase the distance between a training instance and its superficially similar counterparts in the answer space. Given a training instance, the distinguishing module urges the VQA model to give a higher probability on its ground-truth answer to itself than its superficially similar counterparts. Finally, considering the cost of time and space, we implement our method in a resource-efficient way by manipulating the high-level features to construct the superficially similar set and sampling from the set.

In summary, the main contributions of this paper are as follows:

- We are the first to introduce the concept of superficially similar instances and analyze the problem of language priors from this perspective. We also provide two different ways to collect superficially similar counterparts for a given instance.

- We propose a distinguishing module to explicitly encourage the differentiation between a training instance and its superficially similar counterparts, which forces the VQA model to further focus on other information in the input besides the question type.

- Extensive experimental results demonstrate our approach successfully alleviates the language priors and really understands the input. Our method achieves the state-of-the-art results on the benchmark dataset VQA-CP v2, while maintaining competitive performance on the standard VQA v2 dataset.

2 Related Works

Despite the great progress in visual question answering(Anderson et al., 2018; Yang et al., 2016), some researchers observe that most existing VQA models heavily rely on the language priors to make decisions (Agrawal et al., 2016; Goyal et al., 2017; Zhang et al., 2016). Recently, Agrawal et al. (2018) propose a new split of the VQA v1 and VQA v2 datasets (VQA-CP v1 and VQA-CP v2 respectively), which makes the answer distribution of each question type different between the train and test splits. They find that the performance of existing VQA models drops significantly on their new splits compared to the original splits. This fully demonstrates the necessity of overcoming the language priors in VQA.

Previous works on overcoming the language priors in VQA, which is called *debiasing VQA methods*, can be roughly divided into four categories.

Methods modifying model architecture. They usually design a specific model architecture to decompose the process of VQA into several steps. For instance, Agrawal et al. (2018) propose a Grounded Visual Question Answering (GVQA) model to disentangle the recognition of visual concepts from the identification of plausible answers. Similarly, Jing et al. (2020) leverage the decomposed linguistic representations of different kinds of information in the question to decouple the discovery and verification of visual concepts.

Methods strengthening visual attention. They usually leverage the human explanations (e.g., attention maps) to identify the important regions that are needed to answer the question correctly, and train a VQA model to focus on them. Selvaraju et al. (2019) optimize the alignment between the human attention maps and the gradient-based importance of image regions from the VQA model. Wu and Mooney (2019) criticize the sensitivity of

incorrect answers to the important regions that are identified based on human explanations.

Methods reducing unimodal bias. They usually capture the unimodal biases from the language side with a question-only model, and propose strategies to reduce them. Ramakrishnan et al. (2018) train a VQA model and a question-only model that shares the same question encoder in an adversarial way. Cadene et al. (2019) generate a 0-1 mask from the question-only model to modify the predictions of the VQA model, which modulates the importance of training instances with different levels of biases. Moreover, Clark et al. (2019) train an ensemble of a VQA model and a pretrained question-only model, which prevents the VQA model from predicting answers in the way learned by the question-only model. Han et al. (2021) capture different biases with multiple biased models in an ensemble manner, and reduce them step by step. Recently, some works also try to reduce the unimodal bias from a cause-effect perspective (Niu et al., 2021; Niu and Zhang, 2021).

Methods balancing the dataset. They usually make efforts to balance the dataset before training, which reduces the bias of the dataset itself. Zhu et al. (2020) propose to balance the biased data without external annotations, and introduces an auxiliary task upon the balanced data to overcome the language priors. Additionally, Chen et al. (2020) propose the CSS method to synthesize numerous counterfactual samples by masking critical objects in images or words in questions, and trains the VQA model on them to improve its visual-explainable and question-sensitive ability. Moreover, Liang et al. (2020) further improve the CSS method with a contrastive learning strategy.

Our approach shares the similar spirit with the methods balancing the dataset. However, previous works in this category only manipulate a single instance in different ways for data generation, while our method also considers the relationship between different instances in the dataset. And the proposed concept *superficially similar instances* in this work is more general with wider coverage.

3 Methodology

The whole framework of our method is depicted in Figure 2. In the training phase, we optimize the whole framework with the loss $\mathcal{L}_{vqa}$ of a classical VQA paradigm and the proposed distinguishing loss $\mathcal{L}_{dis}$ in a multi-task manner, and the total loss

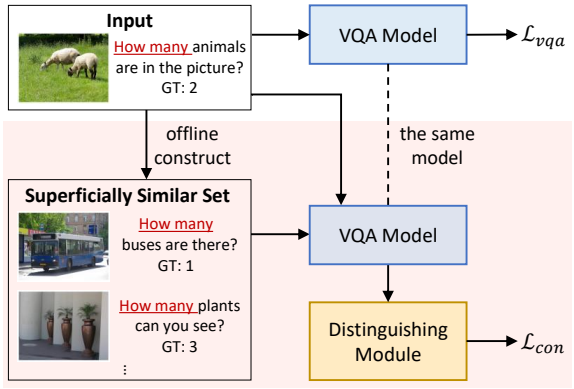

Figure 2: The framework of our method. The pink part at the bottom shows the addition of our method compared to a classical VQA paradigm.

$\mathcal{L}$ is defined as follows:

$$\mathcal{L} = \lambda_1 \cdot \mathcal{L}_{vqa} + \lambda_2 \cdot \mathcal{L}_{dis} \quad (1)$$

where λ_1 and λ_2 are coefficients tuning the influence of two losses. The distinguishing loss $\mathcal{L}_{dis}$ explicitly encourages the model to distinguish between the superficially similar instances. In the inference phase, we just utilize the trained VQA model to make decisions in a classical way.

Given a training instance, we compute its distinguishing loss $\mathcal{L}_{dis}$ in two steps. First, we offline construct a superficially similar set for the instance, which consists of its superficially similar counterparts. Second, when online training, we input the given instance with its superficially similar set into the VQA model, and further exploit the distinguishing module to compute $\mathcal{L}_{dis}$.

3.1 Classical VQA Paradigm

Given a VQA dataset $\mathcal{D} = \{(v_i, q_i, a_i)\}_{i=1}^{N}$ in which each instance is a triplet of an image $v_i \in \mathcal{V}$, a question $q_i \in \mathcal{Q}$, and the corresponding answer $a_i \in \mathbb{R}^{|\mathcal{A}|}$, a VQA paradigm typically learns a mapping from an image-question pair to its answer $\mathcal{F} : \mathcal{V} \times \mathcal{Q} \rightarrow \mathbb{R}^{|\mathcal{A}|}$, where $\mathcal{A}$ is the set of possible answers and each dimension of a_i represents the confidence score of an answer in $\mathcal{A}$. The VQA paradigm can be instantiated with various models (e.g., the UpDn model (Anderson et al., 2018)).

We follow the common formulation of the VQA task, which considers it as a multi-label classification problem. Given a training instance $(v_i, q_i, a_i) \in \mathcal{D}$, the VQA model takes the image-question pair (v_i, q_i) as input, and output a vector of probabilities $p_i \in \mathbb{R}^{|\mathcal{A}|}$ over the answer set:

$$p_i = \sigma(\mathcal{F}(v_i, q_i)), \quad (2)$$

where $\sigma(\cdot)$ denotes the sigmoid function. The training objective is to minimize the binary cross-entropy loss between p_i and a_i for all instances in the dataset $\mathcal{D}$:

$$\mathcal{L}_{vqa} = -\frac{1}{N}\sum_{i=1}^{N}\sum_{k=1}^{|\mathcal{A}|}(a_{ik}\cdot \log p_{ik} + (1-a_{ik})\cdot \log(1-p_{ik})). \quad (3)$$

The classical training objective only forces p_i to approach a_i, but does not explicitly encourage the differentiation of superficially similar instances in the answer space.

3.2 Superficially Similar Set

Before the VQA model learns to distinguish between superficially similar instances, we first construct a superficially similar set $\mathcal{S}$ for each training instance in the dataset $\mathcal{D}$. The set $\mathcal{S}$ is composed of two kinds of superficially similar counterparts for a given instance.

Real counterparts. For the first kind, we directly select the existing instances in the training data with the same question type as the given instance but different answers, which we call *real counterparts*.

Synthetic counterparts. However, since the size of training data is limited, the real counterparts may be not sufficient for a VQA model to acquire satisfactory distinguishing ability. Therefore, we introduce the second kind of superficially similar counterparts as the complement, inspired by the previous work (Zhu et al., 2020). We randomly select images from the training data, and combine them with the question of the given instance to construct new instances, which we call *synthetic counterparts*. Obviously, these new instances and the given instance have the same question type (their questions are the same) but different answers[1] (their images are different). In this way, we can obtain much more superficially similar counterparts than the real counterparts.

Set construction. The construction process of $\mathcal{S}$ is elaborated in Algorithm 1, which is automatic and requires no additional annotation. Given a training instance $(v_i, q_i, a_i) \in \mathcal{D}$, we sequentially process all other training instances $\{(v_j, q_j, a_j)_{j\neq i} \in \mathcal{D}\}$ in two steps. First, we combine q_i with v_j to generate a synthetic counterpart

[1]We ignore the rare case that the randomly selected image coincides with the information needed to get the same answer.

Algorithm 1 Superficially Similar Set Construction

Input: The VQA dataset $\mathcal{D}$; An instance $(v_i, q_i, a_i) \in \mathcal{D}$
Output: The superficially similar set $\mathcal{S}$ of (v_i, q_i, a_i)

```
initialize an empty set S
for each training instance (v_j, q_j, a_j)_{j≠i} ∈ D do
    add (v_j, q_i) into S                          ▷ Synthetic counterpart
    if type(q_i) = type(q_j) then                  ▷ The same question type
        m ← argmax_{l∈A} a_il
        n ← argmax_{l∈A} a_jl
        if m ≠ n then                              ▷ Different answers
            add (v_j, q_j) into S                  ▷ Real counterpart
return S
```

superficially similar counterpart (v_j, q_i), which have the same question type with q_i but cannot be answered by a_i. Second, if the questions q_i, q_j have the same question type and the corresponding answers a_i, a_j are different, we also obtain a superficially similar real counterparts (v_j, q_j).

3.3 Distinguishing Module

Once the superficially similar set of the training instance is constructed, we exploit the distinguishing module to explicitly encourage the distinction between the training instance and its superficially similar counterparts. Specifically, given a training instance $(v_i, q_i, a_i) \in \mathcal{D}$ with its superficially similar set $\mathcal{S}$, we first input them into the VQA model to get their projection in the answer space, i.e., a vector of probabilities $p_i \in \mathbb{R}^{|\mathcal{A}|}$ and a set of such vectors $\{p_j\}$ respectively. Next, the distinguishing module tries to increase the distance between p_i and each element p_j in $\{p_j\}$.

A naive solution is to directly increase the Euclidean distance between the two vectors p_i and p_j. However, this is unreasonable in two folds. First, the direction along which the distance increases should be constrained. As p_i keeps moving away from p_j in the answer space, the value of its dimension corresponding with the ground-truth answer should increase simultaneously, which guarantees that the question q_i can be answered correctly when the distance increases. Second, it doesn't make sense to increase the distance between p_i and p_j along other dimensions except the ones corresponding with their ground-truth answers. For example, if the answers to a training instance and its superficially similar counterpart is "2" and "3" respectively, it is meaningless to compare these two instances on other dimensions such as "yes", "red".

Distinguishing loss. Considering the above two constraints, we design the distinguishing loss in Equation 4. For each training instance (v_i, q_i, a_i),

the VQA model is encouraged to give higher probability on its ground-truth answer to itself than its superficially similar counterpart, and vice versa:

$$\mathcal{L}_{dis} = -\frac{1}{N}\sum_{i=1}^{N}\sum_{j}(\log\sigma(p_{im} - p_{jm}) + \log\sigma(p_{jn} - p_{in})), \quad (4)$$

where m, n are the ground-truth answer of (v_i, q_i, a_i) and that of its superficially similar counterpart respectively. This training objective satisfies the mentioned two constraints. First, the direction of increasing the distance complies with the goal of answering the question correctly, e.g., the value of p_{im} gets larger progressively. Second, the distance between p_i and p_j increases only along the dimensions corresponding with their ground-truth answers, e.g., raising the value of $(p_{im} - p_{jm})$ only increases the distance between p_i and p_j along the m-th dimension.

Note that the superficially similar relationship is symmetrical. The training instance (v_i, q_i, a_i) is also the superficially similar counterpart of each instance in $\mathcal{S}$. Therefore, for the real counterparts from the dataset $\mathcal{D}$, the second term $\log\sigma(p_{jn} - p_{in})$ in Equation 4 can be omitted, since it has been included in $\mathcal{L}_{dis}$. On the other hand, the synthetic counterparts usually have no meaningful answer to define the second term. Thus we can simplify Equation 4 to:

$$\mathcal{L}_{dis} = -\frac{1}{N}\sum_{i=1}^{N}\sum_{j}\log\sigma(p_{im} - p_{jm}). \quad (5)$$

Modulating factor. However, we observe that the improvement of performance is limited if we train the VQA model as Equation 5. We deduce that this is mainly because we treat each superficially similar counterpart equally and do not distinguish between hard and easy ones. More specifically, the superficially similar counterparts which yield a larger p_{jm} are more difficult to distinguish from the training instance (v_i, q_i, a_i), and should be penalized more in $\mathcal{L}_{dis}$. Therefore, inspired by the work (Lin et al., 2017), we finally define the distinguishing loss as follows:

$$\mathcal{L}_{dis} = -\frac{1}{N}\sum_{i=1}^{N}\sum_{j} p_{jm} \cdot \log\sigma(p_{im} - p_{jm}), \quad (6)$$

where we add p_{jm} as a *modulating factor*. Consequently, $\mathcal{L}_{dis}$ will focus more on the hard superficially similar counterparts with larger p_{jm} than the easy ones.

3.4 Resource-Efficient Implementation

For the implementation of the proposed framework, we need to consider the efficiency of both time and space. On the one hand, given a batch of training data, it is time-consuming to run two respective passes in the VQA model to compute $\mathcal{L}_{vqa}$ and $\mathcal{L}_{dis}$. In practice, we only run one pass to extract the high-level features from a batch of image-question pairs, and share them during the computation of $\mathcal{L}_{vqa}$ and $\mathcal{L}_{dis}$. On the other hand, for a training instance, the GPU memory is always too limited to accommodate all the instances in its superficially similar set $\mathcal{S}$. In practice, we randomly sample N_1 the real counterparts and N_2 the synthetic counterparts from the intersection of $\mathcal{S}$ and the current batch of data when online training. Note that the sampling process is also at the feature level and the time cost is negligible. The selection of N_1 and N_2 is discussed in Section 4.5.

4 Experiments

4.1 Experimental Setup

Datasets and evaluation. We evaluate the performance of the proposed method on the benchmark dataset VQA-CP v2 (Agrawal et al., 2018). For completeness, we also evaluate our method on the standard VQA v2(Goyal et al., 2017). We use the standard VQA accuracy as the evaluation metric.

Implementation details. We employ the UpDn model (Anderson et al., 2018) as the base architecture of our approach. The coefficient λ_1 and λ_2 are set to 0.05 and 0.6 respectively. The values of N_1 and N_2 are both set to 1. Additionally, we categorize the questions into 65 question types as in VQA v2 (Goyal et al., 2017).

4.2 Quantitative Analysis

Baselines. We compare the proposed method UpDn+DM with two classical methods SAN (Yang et al., 2016) and UpDn (Anderson et al., 2018), and other debiasing methods[2]: (1) single-model methods: GVQA (Agrawal et al., 2018), AdvReg (Ramakrishnan et al., 2018), RUBi (Cadene et al., 2019), HINT (Selvaraju et al., 2019), SCR (Wu and Mooney, 2019), DLR (Jing et al., 2020), SSL

[2]Methods designed to overcome language priors.

Model	VQA-CP v2 test set				VQA v2 val set			
	Overall	Yes/No	Num	Other	Overall	Yes/No	Num	Other
classical methods								
SAN	24.96	38.35	11.14	21.74	52.02	68.89	34.55	43.80
UpDn	39.74	42.27	11.93	46.05	63.48	81.18	42.14	55.66
single-model debiasing methods								
GVQA	31.30	57.99	13.68	22.14	48.24	72.03	31.17	34.65
AdvReg	41.17	65.49	15.48	35.48	62.75	79.84	42.35	55.16
RUBi	47.11	68.65	20.28	43.18	61.16	-	-	-
HINT	46.73	67.27	10.61	45.88	63.38	81.18	42.99	55.56
SCR	49.45	72.36	10.93	48.02	62.20	78.80	41.60	54.50
DLR	48.87	70.99	18.72	45.57	57.96	76.82	39.33	48.54
SSL	57.59	86.53	29.87	50.03	63.73	-	-	-
Re-scaling	47.09	68.42	21.71	42.88	55.50	64.22	39.61	53.09
CF-VQA	53.55	**91.15**	13.03	44.97	63.54	82.51	43.96	54.30
UpDn+DM (Ours)	**61.13**$^{\pm 0.17}$	88.13$^{\pm 0.31}$	**45.98**$^{\pm 0.63}$	**51.13**$^{\pm 0.04}$	63.53$^{\pm 0.09}$	81.09$^{\pm 0.21}$	39.61$^{\pm 0.24}$	56.52$^{\pm 0.05}$
ensemble debiasing methods								
LMH	52.45	69.81	44.46	45.54	61.64	77.85	40.03	55.04
CSS	58.95	84.37	49.42	48.21	59.91	73.25	39.77	55.11
CSS+CL	59.18	86.99	49.89	47.16	57.29	67.27	38.40	54.71
CSS+LMH+Re-scaling	56.55	83.95	47.81	44.59	55.96	70.52	33.56	50.83
GGE-DQ-tog	57.32	87.04	27.75	49.59	59.11	73.27	39.99	54.39
CSS+IntroD	60.17	89.17	46.91	48.62	62.57	78.57	41.42	56.00

Table 1: Performance (%) on VQA-CP v2 test set and VQA v2 val set. **For fairness**, we mainly compare our approach with single-model methods, and highlight the obtained **best** and second best results in each column. We report the average with the standard variation of results with 5 random seeds.

(Zhu et al., 2020), Re-scaling (Guo et al., 2021), CF-VQA (Niu et al., 2021); (2) ensemble methods: LMH (Clark et al., 2019), CSS (Chen et al., 2020), CSS+CL (Liang et al., 2020), CSS+LMH+Re-scaling (Guo et al., 2021), GGE-DQ-tog (Han et al., 2021), CSS+IntroD (Niu and Zhang, 2021). However, the latter methods are based on stronger ensemble architecture. For fairness, we mainly compare with single-model methods.

Performance on VQA-CP v2 test set. The results on VQA-CP v2 are reported in Table 1. First, we observe that our approach outperforms classical methods by a large margin. Second, our method performs best among single-model methods on the accuracies of "Overall", "Yes/No" and "Other" categories, and second-best on the accuracy of "Yes/No". Moreover, even compared with stronger ensemble methods, our method also achieves competitive results. For example, our method outperforms the second-best single-model method SSL by 3.54% and the **best** ensemble-model method CSS-IntroD by 0.96% on the overall accuracy. It demonstrates the effectiveness of our approach at overcoming language priors.

Performance on VQA v2 val set. The results on VQA v2 val set are reported in Table 1. The debiasing VQA methods usually lead to a decline in the performance over VQA v2 val set, since they tend to penalize the superficial correlation excessively and over-correct the language priors. It is worth noting that our method maintains the competitive performance on VQA v2 with significant improvement on VQA-CP v2, which indicates the robustness of our method.

4.3 Qualitative Analysis

Qualitative examples. We display qualitative examples in Figure 3 to compare our method with its base model UpDn. We observe that UpDn+DM predicts the correct answers while UpDn does not, which shows the effectiveness of our method. For example, as shown in Figure 4b, "white" is the most frequent answer in the training set with respect to the question type "what color is the". While UpDn utilizes the language priors to locate the first man with a "white" shirt by mistake, our method is able to capture the right region "the last man's pants" in the question and find the corresponding region

Figure 3: Qualitative examples in VQA-CP v2 test set. We display the ground-truth answer and predicted answers from UpDn and our method respectively. The best scoring region in the attention map when the model makes the prediction is highlighted in the image.

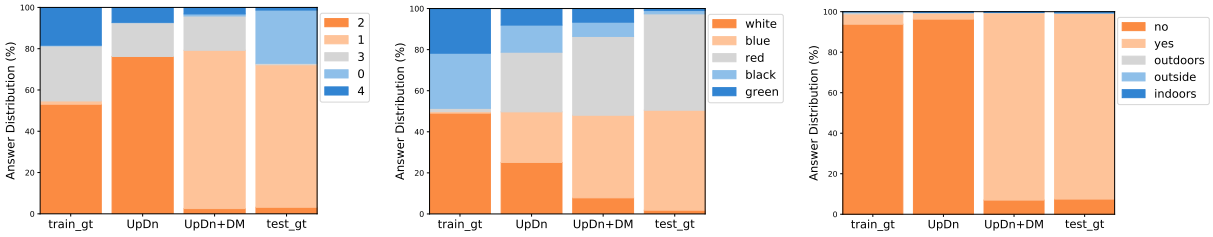

(a) question type: "how many" (b) question type: "what color is the" (c) question type: "is this"

Figure 4: The answer distributions in percentage (%) of specific question types on VQA-CP v2 dataset. For each question type, we display the answer distributions for the ground-truth answers from the training set ("train-gt"), the baseline UpDn evaluated on the test set ("UpDn"), our method UpDn+DM evaluated on the test set ("UpDn+DM") and the ground-truth answers from the test set ("test-gt"). Note that for readability, we only display the most frequent answers of the question type in VQA-CP v2 dataset and omit the others.

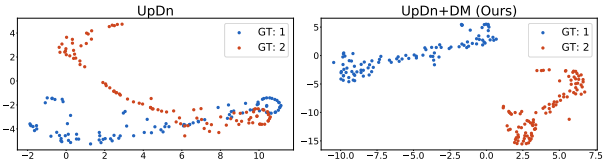

Figure 5: Visualization of the distance between superficially similar instances in the answer space using t-SNE. Each point denotes an instance with the same question type "how many". Different colors denote different ground-truth answers.

in the image to make the right prediction. This indicates that our method can reduce the language priors, and make predictions by understanding the critical parts in the image-question pairs.

Answer distribution. To intuitively understand that our method overcomes the language priors effectively, we compare the answer distributions of some specific question types on VQA-CP v2 dataset in Figure 4. We observe that the answer distribution of our method UpDn+DM is much closer to that of ground-truth answers from the test set than the baseline UpDn, while the answer distribution of UpDn mimics that of ground-truth answers from the training set to a greater extent. It indicates that UpDn+DM does not rely on the language priors in the training set that UpDn suffers from, but really understands the input to give answers on the test set. For instance, in the question type "is this", "yes" makes the majority of answers given by UpDn+DM, as ground-truth answers from the test set; while UpDn mostly outputs the answer "no", as ground-truth answers from the training set.

Distance in the answer space. As shown in Figure 5, we also visualize the distance between superficially similar instances in the answer space with t-SNE. We observe that with our method, superficially similar instances with the same question type and different ground-truth are separated more clearly in the answer space. It intuitively verifies that our approach successfully teaches the VQA model to distinguish between the superficially similar instances. To complement the t-SNE visual-

Model	Overall	Yes/No	Num	Other
Model Ablation				
Ours w/o DM	41.44	43.10	13.24	48.30
Ours w/o MF	44.62	48.73	14.46	50.74
Data Ablation				
Ours w/o SC	42.58	43.37	15.32	49.64
Ours w/o RC	60.47	89.19	39.39	51.21
Ours	61.13	88.13	45.98	51.13

Table 2: Accuracies (%) on VQA-CP v2 test set. "Ours w/o DM" denotes our model variant without the distinguishing module. "Ours w/o MF" denotes our model variant without the modulating factor p_{jm} in Equation 6. "Ours w/o SC" and "Ours w/o RC" denote our model without using the synthetic counterparts and the real counterparts for training, respectively.

Model	Overall	Yes/No	Num	Other
SAN†	28.60	37.64	11.51	28.54
+DM	34.98	50.89	17.75	31.37
UpDn†	41.44	43.10	13.24	48.30
+DM	61.13	88.13	45.98	51.13

Table 3: Accuracies (%) on VQA-CP v2 test set based on the architecture of SAN and UpDn model.† represents re-implementation results.

ization, we also compute inter-class and intra-class Euclidean distances on answer classes with the same question type. The results for UpDn are 0.47 (intra-class distance in blue), 0.40 (intra-class distance in red), 0.52 (inter-class distance), and that for UpDn+DM are 0.32, 0.33, 0.50 respectively. It further demonstrates that our method's remarkable distinguishing ability is achieved by compressing the distance between intra-class data points.

4.4 Ablation Study

Model ablation. In the first part of Table 2, we compare our method with its variant without the whole distinguishing module and the variant without the modulating factor p_{jm} in Equation 6. We observe that the model performance drops violently if we remove the whole distinguishing module or the modulating factor, which demonstrates that both of them are indispensable.

Data ablation. As shown in the second part of Table 2, we observe that both the real counterparts and synthetic counterparts contribute to the improvement, and the synthetic kind contributes much more. This is reasonable since the real counterparts are far less than the synthetic counterparts

N_1	N_2	Overall	Yes/No	Num	Other
1	1	61.17	88.15	46.08	51.16
1	2	61.20	88.55	45.82	51.08
2	1	60.91	88.18	45.28	50.91
1	3	61.22	88.35	46.21	51.13
3	1	60.94	88.05	45.56	50.96
Average		$61.09^{\pm 0.15}$	$88.26^{\pm 0.20}$	$45.79^{\pm 0.38}$	$51.05^{\pm 0.11}$

Table 4: Accuracies (%) on VQA-CP v2 test set with different N_1 and N_2.

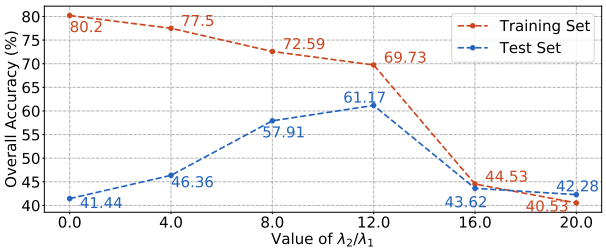

Figure 6: Overall accuracy (%) on VQA-CP v2 training/test set when λ_2/λ_1 varies.

in both quantity and diversity. The real counterparts only come from the VQA dataset itself with a limited size; while within the synthetic counterparts, a question can be combined with different images in different batches, which yields a large amount of diverse superficially similar instances.

4.5 Discussion

Model generalizability. We further conduct experiments on different base models to validate the generalizability of our model. Besides the UpDn model, we also apply our method to the base model Stack Attention Networks (SAN) (Yang et al., 2016). As shown in Table 3, SAN+DM also achieves significant improvement in the accuracies of all categories compared to the original SAN, which shows our method can be generalized to other model architectures.

Varying N_1 and N_2. To investigate the influence of the sampled number of superficially similar counterparts in the resource-efficient implementation, we fix $N_1 = 1$ or $N_2 = 1$ and vary the other from 1 to 3. As shown in Table 4, the results are stable with different N_1 and N_2. We set both parameters as 1 as default.

Tuning the influence of $\mathcal{L}_{dis}$. We adjust the ratio λ_2/λ_1 to tune the influence of $\mathcal{L}_{dis}$. The results are shown in Figure 6. As the ratio of $\mathcal{L}_{dis}$ increases, the performance on the training set drops constantly, which demonstrates the model is less dependent on language priors to make predictions. Moreover, the performance on the test set first in-

creases in the range of small λ_2/λ_1 values, and then decreases when λ becomes too large. We deduce that if the ratio is too large, the influence of $\mathcal{L}_{dis}$ would overwhelmingly exceed that of $\mathcal{L}_{vqa}$ and make the VQA model underfit. Overall, we should set λ_2/λ_1 in a reasonable range to alleviate language priors as well as prevent underfit.

5 Conclusion

In this paper, we introduce the concept superficially similar instances and propose a novel training framework for overcoming language priors by explicitly encouraging the VQA model to distinguish between such instances. Extensive experiments show that our approach achieves the state-of-the-art results on VQA-CP v2, while maintaining competitive performance on VQA v2. Qualitative analysis also demonstrates that our method alleviates the language priors effectively and really understands the input.

Acknowledgements

This research is supported by the Chinese Scientific and Technical Innovation Project 2030 (2018AAA0102100), the National Natural Science Foundation of China (No. 62272250, U1936206, U1903128).

References

Aishwarya Agrawal, Dhruv Batra, and Devi Parikh. 2016. Analyzing the behavior of visual question answering models. In *EMNLP*.

Aishwarya Agrawal, Dhruv Batra, Devi Parikh, and Aniruddha Kembhavi. 2018. Don't just assume; look and answer: Overcoming priors for visual question answering. In *CVPR*.

Peter Anderson, Xiaodong He, Chris Buehler, Damien Teney, Mark Johnson, Stephen Gould, and Lei Zhang. 2018. Bottom-up and top-down attention for image captioning and visual question answering. In *CVPR*.

Remi Cadene, Corentin Dancette, Matthieu Cord, Devi Parikh, et al. 2019. Rubi: Reducing unimodal biases for visual question answering. In *NeurIPS*.

Long Chen, Xin Yan, Jun Xiao, Hanwang Zhang, Shiliang Pu, and Yueting Zhuang. 2020. Counterfactual samples synthesizing for robust visual question answering. In *CVPR*.

Christopher Clark, Mark Yatskar, and Luke Zettlemoyer. 2019. Don't take the easy way out: Ensemble based methods for avoiding known dataset biases. In *EMNLP-IJCNLP*.

Yash Goyal, Tejas Khot, Douglas Summers-Stay, Dhruv Batra, and Devi Parikh. 2017. Making the v in vqa matter: Elevating the role of image understanding in visual question answering. In *CVPR*.

Yangyang Guo, Liqiang Nie, Zhiyong Cheng, Qi Tian, and Min Zhang. 2021. Loss re-scaling vqa: revisiting the language prior problem from a class-imbalance view. *IEEE Transactions on Image Processing*, 31:227–238.

Xinzhe Han, Shuhui Wang, Chi Su, Qingming Huang, and Qi Tian. 2021. Greedy gradient ensemble for robust visual question answering. In *ICCV*.

Chenchen Jing, Yuwei Wu, Xiaoxun Zhang, Yunde Jia, and Qi Wu. 2020. Overcoming language priors in vqa via decomposed linguistic representations. In *AAAI*.

Zujie Liang, Weitao Jiang, Haifeng Hu, and Jiaying Zhu. 2020. Learning to contrast the counterfactual samples for robust visual question answering. In *EMNLP*.

Tsung-Yi Lin, Priya Goyal, Ross Girshick, Kaiming He, and Piotr Dollár. 2017. Focal loss for dense object detection. In *ICCV*.

Yulei Niu, Kaihua Tang, Hanwang Zhang, Zhiwu Lu, Xian-Sheng Hua, and Ji-Rong Wen. 2021. Counterfactual vqa: A cause-effect look at language bias. In *CVPR*.

Yulei Niu and Hanwang Zhang. 2021. Introspective distillation for robust question answering. In *NeurIPS*.

Sainandan Ramakrishnan, Aishwarya Agrawal, and Stefan Lee. 2018. Overcoming language priors in visual question answering with adversarial regularization. In *NeurIPS*.

Ramprasaath R Selvaraju, Stefan Lee, Yilin Shen, Hongxia Jin, Shalini Ghosh, Larry Heck, Dhruv Batra, and Devi Parikh. 2019. Taking a hint: Leveraging explanations to make vision and language models more grounded. In *ICCV*.

Jialin Wu and Raymond Mooney. 2019. Self-critical reasoning for robust visual question answering. In *NeurIPS*.

Zichao Yang, Xiaodong He, Jianfeng Gao, Li Deng, and Alex Smola. 2016. Stacked attention networks for image question answering. In *CVPR*.

Peng Zhang, Yash Goyal, Douglas Summers-Stay, Dhruv Batra, and Devi Parikh. 2016. Yin and yang: Balancing and answering binary visual questions. In *CVPR*.

Xi Zhu, Zhendong Mao, Chunxiao Liu, Peng Zhang, Bin Wang, and Yongdong Zhang. 2020. Overcoming language priors with self-supervised learning for visual question answering. In *IJCAI*.

Efficient Multilingual Multi-modal Pre-training through Triple Contrastive Loss

Youhan Lee
Kakao Brain
youhan.lee@kakaobrain.com

Kyungtae Lim
Hanbat National University
ktlim@hanbat.ac.kr

Woonhyuk Baek, Byungseok Roh, Saehoon Kim
Kakao Brain
wbaek,peter.roh,shkim@kakaobrain.com

Abstract

Learning visual and textual representations in the shared space from web-scale image-text pairs improves the performance of diverse vision-and-language tasks, as well as modality-specific tasks. Many attempts in this framework have been made to connect English-only texts and images, and only a few works have been proposed to extend this framework in multilingual settings with the help of many translation pairs. In this multilingual approach, a typical setup is to use pairs of (image and English-text) and translation pairs. The major limitation of this approach is that the learning signal of aligning visual representation with under-resourced language representation is not strong, achieving a sub-optimal performance of vision-and-language tasks. In this work, we propose a simple yet effective enhancement scheme for previous multilingual multi-modal representation methods by using a limited number of pairs of images and non-English texts. In specific, our scheme fine-tunes a pre-trained multilingual model by minimizing a triple contrastive loss on triplets of image and two different language texts with the same meaning, improving the connection between images and non-English texts. Experiments confirm that our enhancement strategy achieves performance gains in image-text retrieval, zero-shot image classification, and sentence embedding tasks.

1 Introduction

Transferring visual representations learned from a large annotated set into downstream tasks of interest (Deng et al., 2009; Zhai et al., 2019) is the standard approach to achieve the state-of-the-art performance (Kuznetsova et al., 2020; Kolesnikov et al., 2020). However, due to the labeling cost, the scalability of this approach is rather limited. In contrast, self-supervised learning with contrastive losses (He et al., 2020; Chen et al., 2020) has proven to learn semantic representations without using the explicit labels, which becomes a promising solution to obtaining general visual representations. In addition, this paradigm combined with billion-scale unlabeled samples is competitive with the annotation-based transfer learning approaches in multiple tasks (Goyal et al., 2021).

In natural language processing (NLP), pre-training with billions of corpus and transferring to downstream tasks has likewise achieved tremendous success. And it has been a de facto standard for a recent decade (Devlin et al., 2019; Radford et al., 2019; Brown et al., 2020). Witnessing successes in two domains, researchers have been actively attempting to find visual-and-language (VL) representations by combining supervisions that come from billions of both images and languages. (Li et al., 2019; Lu et al., 2019, 2020; Kim et al., 2021) have attempted to utilize highly curated VL datasets such as MS-COCO (Lin et al., 2014) or Visual Genome (Krishna et al., 2016). However, the non-trivial collection process is the major limitation in scaling up these datasets. CLIP (Radford et al., 2021) overcame the limitation by learning VL representations from web-scale 400M image-to-text pairs on both visual and VL downstream tasks, even with a simple contrastive loss. ALIGN (Jia et al., 2021) extended this approach by scaling up image-text pairs to 1.8B with simple filtering compared to CLIP and showed state-of-the-art scores on both visual and visual-language tasks.

Such web-scale approaches as well as the earliest attempts have focused on connecting images to English texts. Using a separate translator might be a practical solution to match images and multilingual texts, but this resorts to sub-optimal results because the effect of translation errors is not explicitly considered in the VL representation learning process, and translating every text into dozens of languages is inefficient. In language models, many attempts have been successfully made to develop language-agnostic representations, improving the performance of downstream tasks on under-

Proceedings of the 29th International Conference on Computational Linguistics, pages 5730–5744
October 12–17, 2022.

resourced languages. In order to achieve this goal, Pires et al. (2019); Liu et al. (2020); Xue et al. (2021) pre-train language models over a multilingual corpus for transferring knowledge across languages. Chi et al. (2021); Feng et al. (2020) utilize bilingual translation pairs to transfer the information from common languages to under-resourced ones in a more efficient way.

Following this research direction in language models, MURAL (Jain et al., 2021) suggests the efficient multilingual VL modeling that leverages both the alignments of (a) monolingual image-to-text and (b) multilingual text-to-text on the training dataset simultaneously. However, since English is the only language that has a direct connection with images, the performance of non-English languages on several multi-modal tasks is relatively weaker than that of English. Linking all languages directly to images would be the easiest and ideal solution, but obtaining billions of image-text datasets for all languages is nearly impossible, especially for low-resourced languages.

To tackle the data-scarcity problem, we propose a simple but efficient enhancement via triplet contrastive learning (ETCL) that utilizes multi-modal zero-shot transfer through relatively small amounts of image-text datasets in non-English languages through a triplet contrastive loss. In ETCL, we further train the pre-trained multilingual VL model with a new limited number of pairs of image and non-English texts to strengthen the weak alignment of the pre-trained model. In order to fully leverage multi-modal cross-lingual zero-shot transfer, we introduce a triplet contrastive learning which considers (a) image-textA, (b) textA-textB and (c) textB-image, where textA and textB are multilingual translation pairs.

Through various experiments, we show that our framework provides significant performance improvement for various languages in multi-modal retrieval tasks and zero-shot image classification and sentence embedding tasks. Interestingly, the large performances gains also occur in some languages which are not included in the training dataset for ETCL. Through ablation study, we prove that the phenomenon comes from the multi-modal cross-lingual zero-shot transfer with regard to grammatical and geographic relationships between languages. And we show that ETCL leverages the relationships efficiently. In summary, our contributions to the multilingual VL representation are:

- We propose the ETCL framework that leverages multi-modal cross-lingual zero-shot transfer to train VL model efficiently.
- We show that ETCL gives the large performance gains on various downstream tasks such as image-text retrieval, zero-shot image classification and language tasks.
- We provide the empirical analysis that ETCL utilizes the geographical and grammatical relationships between languages efficiently.

2 Related works

We briefly review the research areas and key references most relevant to our approach.

2.1 Multilingual representation learning

Multilingual language models have shown that a single large model improves diverse multilingual NLP tasks, removing the need for maintaining language-specific models (Pires et al., 2019; Liu et al., 2020; Xue et al., 2021; Feng et al., 2020; Chi et al., 2021). Multilingual BERT (Pires et al., 2019), dubbed as mBERT, is a multilingual variant of BERT (Devlin et al., 2019), which is pre-trained with the masked language modeling objective over about 100 languages. LaBSE (Feng et al., 2020) adapts mBERT to learn language-agnostic sentence embedding over bilingual translation pairs, improving cross-lingual retrieval tasks significantly. mBART (Liu et al., 2020) trains an encoder-decoder architecture on a large corpus composed of multiple languages using BART objective (Lewis et al., 2019). Considering multilingual variants to newly proposed models has still been actively studied. For instance, T5 (Raffel et al., 2020) is also extended to a multilingual model, which is named mT5 (Xue et al., 2021), by training a sequence-to-sequence model over 101 languages. Recently, mT6 (Chi et al., 2021) further improves mT5 by proposing a novel objective for text-to-text pre-training.

2.2 Vision-and-Language representation learning

Learning VL representations in a self-supervised fashion is a promising approach to solving many visually grounded language understanding tasks (Li et al., 2019; Lu et al., 2019, 2020; Kim et al., 2021). ViLBERT (Lu et al., 2019) and VisualBERT (Li et al., 2019) try to align patches in an

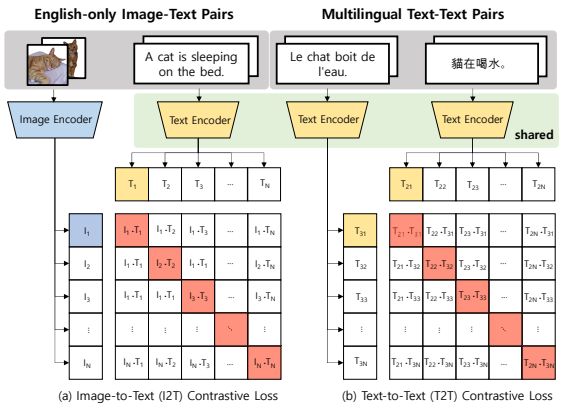

Figure 1: Illustration on English-only image-to-text and multilingual text-to-text matching, where the text encoder is shared across languages.

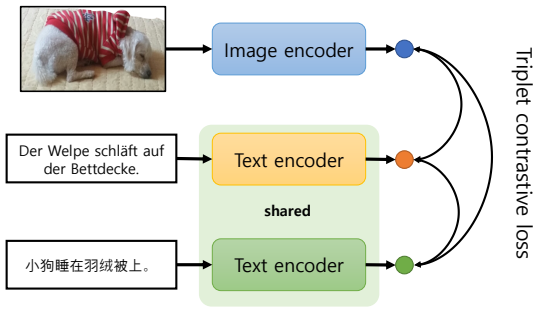

Figure 2: Illustration on a triple contrastive loss using (1) image-textA, (2) textA-textB and (3) textB-image, where the text encoder is shared across languages.

image into (sub-)words through a sequence of self-attention layers, generating transferable representations on many VL downstream tasks. ViLT (Kim et al., 2021) advances the model architecture and training procedure to achieve comparable performance without using the region proposal network. Since these works rely on highly curated multimodal datasets, the scalability of these approaches is rather limited. CLIP (Radford et al., 2021) and ALIGN (Jia et al., 2021) have shown that it is possible to learn transferrable visual and VL representations from noisy web-scale datasets. Very recently, MURAL (Jain et al., 2021) extends ALIGN in a multilingual setting by leveraging billion-scale translation pairs. However, the performance of under-resourced languages on various VL tasks is still weaker than that of English. Otherwise, KELIP(Ko and Gu, 2022), a CLIP-style bilingual VL model trained using 708M Korean and 476M English image-text pairs, has been proposed, showing strong representations on Korean VL tasks. However, the approach is not an optimal solution for multilingual VL modeling because gathering image-text pairs for more than a hundred languages is practically impossible. Our work suggests a simple but effective solution to boost the performance of under-resourced languages.

3 Approach

This section describes the details of conventional multilingual VL modeling, the proposed ETCL scheme, and training details.

3.1 Background: Multilingual VL pretraining

MURAL (Jain et al., 2021) suggests large-scale multilingual VL modeling using pairs of (image and English-text) and translation pairs. (see Figure.1). For training VL representations, the system requires capturing the meaning of images and texts at the same time. In this context, contrastive loss is a suitable objective since it places semantically similar images and sentences in the same latent vector space as follows:

$$\mathcal{L}_{\text{i2t}} = -\frac{1}{N}\sum_{i=1}^{N}\frac{e^{\phi(x_i,y_i)}}{\sum_{n=1}^{N} e^{\phi(x_i,y_i)}}, \qquad \mathcal{L}_{\text{t2i}} = -\frac{1}{N}\sum_{i=1}^{N}\frac{e^{\phi(y_i,x_i)}}{\sum_{n=1}^{N} e^{\phi(y_i,x_i)}}, \tag{1}$$

$$\mathcal{L}_{\text{image-text}} = \mathcal{L}_{\text{i2t}} + \mathcal{L}_{\text{t2i}}, \tag{2}$$

where $\mathcal{L}_{\text{i2t}}$ and $\mathcal{L}_{\text{t2i}}$ represent the image-to-text and text-to-image losses respectively and ϕ denotes scoring function $x^\top y$. Also, x_i and y_i denote the image and text features in the i-th pair, N is the batch size, σ is a learnable softmax temperature. The loss function for text-to-text matching from translation pairs is similarly defined and denoted by $\mathcal{L}_{\text{textA-textB}}$ and $\mathcal{L}_{\text{textB-textA}}$, and $\mathcal{L}_{\text{text-text}}$ is defined as the sum of two losses. The final objective of MURAL is followed by:

$$\mathcal{L}_{\text{MURAL}} = \frac{1}{2}\left(\mathcal{L}_{\text{image-text}} + \mathcal{L}_{\text{text-text}}\right), \tag{3}$$

In addition to this, MURAL adds an addictive margin (Yang et al., 2019) to $\mathcal{L}_{\text{MURAL}}$.

3.2 Enhancement via triple contrastive loss

Although MURAL trains a multilingual VL representation efficiently, the alignment of visual representation with under-resourced language representation is still weaker than that of English. To tackle this limitation, we suggest an enhancement scheme that conducts training by adding a small amount of non-English image-text dataset to improve weak alignment between image and non-English languages. As with (Ruan et al., 2022), we introduce a triple contrastive loss that takes into account all pair possibilities rather than simply training with $\mathcal{L}_{\text{image}-\text{text}}$ between images and non-English captions to take full advantage of zero-shot transfers across multi-modal cross-lingual in the training dataset (see Figure 2). Before defining the proposed triple contrastive loss, we prepare image, and two pairs of text in different languages with the same meaning. Since we have prepared three different representations of training data with the same semantic, we can simply apply contrastive loss to the three inputs(image, textA, textB) as follows:

$$\mathcal{L}_{\text{ETCL}} = \frac{1}{3}\left(\mathcal{L}_{\text{image-textA}} + \mathcal{L}_{\text{textA-textB}} + \mathcal{L}_{\text{textB-image}}\right). \quad (4)$$

Here, we refer to our enhancement scheme as ETCL, abbreviated version of **E**nhancement via **T**riplet **C**ontrastive **L**earning. Combined with MURAL, we name it as MURAL+ETCL.

3.3 Model details

Since there is no publicly available MURAL model, we reproduce the MURAL using the architecture and training method as proposed in MURAL (Jain et al., 2021). The image-text pairs required are created by ourselves, and the publicly available CCMatrix is used for text-text pairs.

Model architecture Our model is composed of an image encoder and three text encoders to align image-text and text-text pairs (see Figure 1), where the parameters of all text encoders are shared. For the image encoder, we follow the same protocol of ALIGN (Jia et al., 2021) by using EfficientNet (Tan and Le, 2019) with global averaging pooling to obtain the embedding of an image. For the text encoder, we choose BERT (Devlin et al., 2019) and use the hidden representation of [CLS] token as the embedding of a text. To make sure that the embeddings have the same dimension, we add an additional fully-connected layer on the top of the text encoder.

Pre-processing For a fair comparison, we try to follow the data pre-processing and augmentation used in ALIGN and MURAL as much as possible. For training, we resize images of arbitrary resolutions into 346 $\times$ 346 resolution regardless of the aspect ratio. Then, we randomly crop the image to 289 $\times$ 289, and apply the horizontal flip of probability 0.5. For evaluation, we resize the image to 346 $\times$ 346, and apply the center crop of 289 $\times$ 289. For both training and evaluation, we use the same pre-processing for the texts. We use the tokenizer having 550K vocabulary provided by the official repository of LaBSE [1], and truncate the sequence to have the maximum length of 64.

Training details For MURAL optimization, we use LAMB optimizer (You et al., 2020) with a weight decay ratio of 1e-5. The learning rate is linearly increased from zero to 1e-3 in 10k steps, and then linearly decays to zero in 800k steps. We use the label smoothing of 0.1. The temperature is initialized as 1.0, and the margin m is 0.3 same as to LaBSE (Feng et al., 2020). We train the model with a batch size of 16,384 on 128 Cloud TPU V3 cores with 128 positive pairs on each core. Since a large number of negative samples is critical in contrastive learning, we adopt the cross-accelerator negative sampling as used in LaBSE. This enables the large batch training by collecting samples in all synchronized cores and treating them as negative samples. For ETCL, we use a 32,768 batch size with a learning rate of 1e-4, which decreases from 1e-4 to zero linearly in 3k steps.

3.4 Data collection

Image-text pairs We follow the data collection process of ALIGN to create our in-house 1B image-text pairs dataset from Common crawl [2]. The raw descriptions are gathered from the Alt-text HTML attribute associated with web images. We only apply minimal rule-based filtering as detailed below. For image-based filtering, we only keep images whose shorter dimension is larger than 200 pixels and set the aspect ratio as 3. We discard images with more than 100 associated alt-text. For text-filtering, we exclude alt-texts that are shared by more than 10 images. We also discard either too short (<3 unigrams) or too long (>100 unigrams or >1000 characters). These filters include discarding instances that are classified as non-English by

[1] https://tfhub.dev/google/LaBSE/2

[2] https://commoncrawl.org/the-data/

cld3 (compact language detector v3) [3]. Additionally, we also include CC3M (Sharma et al., 2018) and CC12M (Changpinyo et al., 2021) in our 1B image-text pairs dataset.

Text-text pairs We use all 10.8B text-text pairs in CCMatrix dataset (Schwenk et al., 2021) for the text-test alignment, which covers 90 languages and have 1,197 bitexts across multiple languages. All pairs are publicly available on the OPUS website [4].

Multilingual image-text pairs We use multilingual CC3M datasets which consist of translated versions of original CC3M in 5 languages (de, fr, cs, zh and ja), which are provided (Zhou et al., 2021). After generating 15 language pairs in 6 languages without duplicates, image-textA-textB 45M triple dataset was created and used for `ETCL`. In this dataset, the number of unique images is 3M, and the captions are 18M.

4 Experiments

In this section, we present experiment settings to evaluate our proposed model over VL, visual, and language tasks.

4.1 Task description

Image-text retrieval The image-text retrieval is the most suitable task to evaluate the VL model because this task uses the representation of language and image simultaneously. Following (Jain et al., 2021; Jia et al., 2021; Li et al., 2019), we validate the ability of our VL representation on multilingual Flickr30K (Young et al., 2014; Elliott et al., 2016, 2017; Barrault et al., 2018) and MS-COCO (Lin et al., 2014; Yoshikawa et al., 2017) in image-to-text and text-to-image retrieval tasks with zero-shot and fine-tuning scenarios. In this experiment, we use multilingual Flickr30K having English, German, French, and Czech captions, where and German captions are provided by Multi30K (Elliott et al., 2016) and French and Czech captions are obtained by the translation (Elliott et al., 2017; Barrault et al., 2018). Multi30K contains five captions per image in English and German, and one description per image in French and Czech. We use 29K, 1K, and 1K images for the train, validation, and test sets, respectively as used in the original dataset (Young et al., 2014). In addition to English MS-COCO, Japanese (Yoshikawa et al., 2017) and Korean MS-COCO [5] are also used for the VL evaluation. We follow the split protocol used in (Karpathy and Fei-Fei, 2015), resulting in 82K training and 5K test sample. For evaluation, we measure Recall@K with respect to $K = 1, 5, 10$ on two retrieval tasks. We report the performance of each model by Average Recall (AR), taking the mean over these six scores.

During fine-tuning, we follow the same protocol used in ALIGN for a fair comparison. The pre-trained model is fine-tuned by the image-text contrastive loss (without text-text matching loss). We use the batch size of 2,048 and set the learning rate to 1e-5 with a linear decay scheduler, and fine-tune the model over 3K and 6K steps on Flickr30K and MS-COCO, respectively. All the other hyperparameters are consistent with the ones in pre-training.

Zero-shot image classification Performance on the zero-shot image classification has been considered one of the important evaluation tasks for the large-scale multi-modal pre-training model since it represents the generalization ability of a model. Following previously proposed studies(Jain et al., 2021; Jia et al., 2021), we validate the visual representation power of our method on multilingual zero-shot classification tasks based on text prompts. Furthermore, an additional result on ImageNet K-Nearest-neighbor (KNN) tasks without text prompts is provided in Appendix A.3. Our models are evaluated on diverse classification datasets, including ImageNet (Deng et al., 2009), CIFAR100 (Krizhevsky, 2009), SUN397 (Xiao et al., 2010), and Fool101 (Bossard et al., 2014). We conduct the zero-shot image classification based on text prompts (Radford et al., 2021) over 83 languages, where prompts are translated by Google Translator. We remark that the translator supports 83 languages among 90 languages covered in CCMatrix. For a fair comparison, we adopt the text prompt engineering used in CLIP. For instance, in the case of Food101, a context prompt is inserted, so the final prompt is *"A photo of a {label}, a type of food"*, and the context prompt is also translated over 83 languages. Similar to MURAL, we compare models on three groups, `All-languages`, `well-resourced` and `under-resourced` to deeply investigate our model in different resource conditions. The

[3] https://github.com/google/cld3
[4] https://opus.nlpl.eu/CCMatrix.php
[5] Korean COCO is from https://aihub.or.kr/keti_data_board/visual_intelligence.

Type	Model	Backbone	Image-text pairs	Text-text pairs	Flickr30K					MSCOCO 5K			
					en	de	fr	cs	avg.	en	ja	ko	avg.
Zero-shot	ALIGN-BASE	B5+BERT-Base	1.8B	-	83.3	75.0	74.2	47.9	70.1	59.6	53.9	-	-
	ALIGN-L2	L2+BERT-Large			**92.2**	-	-	-	-	**70.9**	-	-	-
	MURAL-BASE	B5+BERT-Base	1.8B	6B	82.4	76.2	75.0	64.6	74.6	59.5	54.4	-	-
	MURAL-LARGE	B7+BERT-Large			89.2	83.5	83.1	77.0	83.2	67.7	**64.6**	-	-
	MURAL(reprod.)	B7+BERT-Base	1B	10.8B	90.1	70.5	70.1	63.7	73.6	67.8	40.6	31.5	46.6
	MURAL(reprod.) + ETCL	B7+BERT-Base	1B + 45M	10.8B + 45M	90.6	**85.6**	**85.8**	**82.2**	**86.0**	69.1	62.1	**49.3**	**60.2**
Fine-tune	ALIGN-BASE	B5+BERT-Base	1.8B	-	92.2	88.5	88.1	84.5	88.3	74.8	72.5	-	-
	ALIGN-L2	L2+BERT-Large			**96.0**	-	-	-	-	**83.4**	-	-	-
	MURAL-BASE	B5+BERT-Base	1.8B	6B	92.2	88.6	87.6	84.2	88.2	75.4	74.9	-	-
	MURAL-LARGE	B7+BERT-Large			93.8	90.4	89.9	87.1	90.3	81.2	**81.3**	-	-
	MURAL(reprod.)	B7+BERT-Base	1B	10.8B	94.5	89.8	90.2	87.8	90.6	79.3	77.9	73.3	76.9
	MURAL(reprod.) + ETCL	B7+BERT-Base	1B + 45M	10.8B +45M	94.8	**91.0**	**91.3**	**89.7**	**91.7**	79.8	79.2	**77.0**	**77.8**

Table 1: Average Recall of image-to-text and text-to-image retrieval tasks on multilingual Flickr30K and MS-COCO in zero-shot and fine-tuning scenarios for English (en); German (de); French (fr); Czech (cs); Japanese (ja). The numbers of ALIGN and MURAL are taken from (Jain et al., 2021). The last column for each dataset means the average score over all languages.

groups are below:

- well-resourced: English (en), German(de), French (fr), Czech (cs), Japanese (ja), Chinese (zh), Russian (ru), Polish (pl), Turkish (tr)
- under-resourced: Uzbek (uz), Irish (ga), Belarusian (be), Malagasy (mg), Cebuano (ceb)

Sentence similarity & retrieval For the multilingual VL model, one important question is still remaining: *what does the model learn from languages and how can we assess its ability to NLP tasks?* As previous VL studies proposed (Jain et al., 2021), we validate the performance of sentence embeddings from our approach in (multilingual and monolingual) sentence similarity comparison (Cer et al., 2017) and cross-language sentence retrieval tasks (Artetxe and Schwenk, 2019). For the sentence similarity comparison, we choose STS 2017 (Cer et al., 2017) and its extended version (Reimers and Gurevych, 2020). Both datasets contain human-annotated labels of the similarity from 0 (no meaning overlapping) to 5 (equivalent meaning) for every pair of sentences. We compute the Spearman rank correlation between the cosine similarity of two sentence embeddings and the ground-truth label in both 3 monolingual and 7 multilingual settings. We also evaluate ETCL on a multilingual sentence retrieval task with Tatoeba dataset (Artetxe and Schwenk, 2019). In this experiment, we observe that ETCL have decent performance compared to other language models, and more details could be found in Appendix A.5.

5 Results

As presented in our contributions in Section 1, our study has different goals: (1) investigate the performance gains based on the proposed method on various downstream tasks including image-text retrieval, zero-shot image classification and sentence retrieval (2) study the effect of ETCL in related languages (geographically and grammatically), (3) check whether ETCL activates information sharing in multilingual training. By taking into account the proposed goals, in this section, we report our experimental results in various tasks.

5.1 Downstream Task Results on Image-Text Retrieval

To investigate the effectiveness of the proposed model, we conducted experiments on the Image-Text Retrieval task. Table 1 shows the average recall over three different Ks of two retrieval tasks in zero-shot and fine-tuning scenarios. We note that the reproduction of MURAL is successful because MURAL(reprod.) performs comparably to ALIGN-BASE and MURAL-LARGE in all languages of the two datasets.

First, for Flickr30K, ALIGN-L2 performs better than MURAL(reprod.)+ETCL in both zero-shot and fine-tuning cases (92.2 vs. 90.1 and 96.0 vs. 94.5) in the case of English, because the capacity of the image encoder of ALIGN-L2 is larger than MURAL(reprod.)+ETCL. In the case of non-English, we observe that MURAL(reprod.)+ETCL model shows better performance compared to MURAL-LARGE and ALIGN-L2 on German, French in zero-shot and German, French, and Czech in fine-tune setting

even if the model capacity is smaller than that of two models.

For MS-COCO, MURAL(reprod.)+ETCL also shows competitive performance for all languages in both zero-shot and fine-tune cases with the relatively small size of model compared to MURAL-LARGE and ALIGN-L2. Especially for Korean, which is not included in languages of new image-text pairs for ETCL, MURAL(reprod.)+ETCL outperforms MURAL(reprod.) in both zero-shot and fine-tuning cases by 17.8 and 3.7, respectively. These results confirm that multi-modal cross-lingual zero-shot transfer occurs effectively for languages not in training data after ETCL. In other words, zero-shot transfer strengthens the weak connection between image and non-English text after indirect alignment training. A more in-depth study of the phenomenon will be discussed in Section 5.4.

In Appendix A.1, all Recall@K with respect to $K = 1, 5, 10$ on two retrieval tasks for all languages are provided.

5.2 Downstream Task Results on Zero-Shot Image Classification

Language	Model	ImageNet	SUN397	Food101	CIFAR100
All languages	MURAL(reprod.)	18.4	29.6	29.1	16.6
	MURAL(reprod.) + ETCL	**26.9**	**37.1**	**45.1**	**19.6**
Well-resourced	MURAL(reprod.)	26.4	40.4	29.1	16.6
	MURAL(reprod.) + ETCL	**38.1**	**50.7**	**45.1**	**19.6**
Under-resourced	MURAL(reprod.)	15.2	23.0	29.4	16.7
	MURAL(reprod.) + ETCL	**23.6**	**29.2**	**45.5**	**19.8**

Table 2: Classification accuracy (%) on multilingual zero-shot image classification on 4 datasets. Languages are grouped as followed (1) All languages, (2) well-resourced languages and (3) under-resourced languages group. All values are the average of 100 × Acc@1 across languages.

As presented in Section 4.1, we conduct zero-shot image classification and languages are grouped to see general trends depending on language resources. Table 2 shows the zero-shot classification performance on four datasets. All detailed results are illustrated in Appendix A.4. We note that all results are reliable to compare the trends because MURAL(reprod.)+ETCL shows comparable performance compared to CLIP-ViT/32 (62.6 vs 64.6). For all three groups, MURAL(reprod.)+ETCL shows better performance on four datasets. For the well-resourced group, the performance of ru, pl and tr improves although those languages are not in the newly added languages. Above all, the performance of the under-resourced group which does not include any newly added languages also improves largely over all datasets, which confirms that ETCL gives effective multi-modal zero-shot cross-lingual transfer over low-resourced languages as well.

5.3 Results on language tasks

To answer the raised question about assessing our model in NLP tasks in Section 4.1, we conduct NLP-oriented experiments. The goal of this experiment is to verify how well sentence embeddings obtained from pre-trained models can solve sentence similarities and search tasks.

	en-en	es-es	ar-ar	Avg.
LaBSE	79.4	80.8	69.1	76.4
MURAL(reprod.)	84.8	80.5	69.2	78.1
MURAL(reprod.) + ETCL	**87.3**	**83.3**	**76.2**	**82.3**

Table 3: Performance on extended STS 2017 similarity comparison task in the monolingual setting. Scores are calculated by 100 × Spearman rank correlation between the cosine similarity of sentence embeddings and the gold labels.

	en-ar	en-de	en-tr	en-es	en-fr	en-it	en-nl	Avg.
LaBSE	74.5	73.8	72.0	65.5	77.0	76.9	75.1	73.5
MURAL(reprod.)	70.7	70.4	69.8	69.1	72.9	71.8	72.4	71.0
MURAL(reprod.) + ETCL	**75.7**	**84.1**	**76.3**	**79.5**	**85.3**	**82.1**	**82.3**	**80.7**

Table 4: Performance on extended STS 2017 similarity comparison task in the multilingual setting. Scores are calculated by 100 × Spearman rank correlation between the cosine similarity of sentence embeddings and the gold labels.

Semantic Textual Similarity To evaluate the sentence embedding performance of the text encoder, the STS task was performed with the same evaluation protocol as done in (Reimers and Gurevych, 2019). Since the text encoder is trained with a contrastive loss, we set LaBSE, a text-encoder-only language model trained in the same way, as a baseline. As shown in Table 3 and 4, the performance of the text encoder of MURAL(reprod.) is similar to that of LaBSE. This is because the two text encoders are trained using text-text alignment based on the parallel text pairs. Interestingly, we can see that the performance of MURAL(reprod.)+ETCL increases significantly by 4.2 and 9.7 in STS 2017 monolingual and multilingual settings respectively. We conjecture that this performance improvement is

due to the synergy caused by the image domain because the multi-modal model is trained with both images and text simultaneously. In summary, these results reveal that since the text encoder of the VL model is trained along with the image encoder, there is room for improvement in the performance of the text encoder by using the image encoder, unlike the text encoder-only language model. a

Model	Avg	Gain over baseline
MURAL(reprod.) baseline	31.5	-
MURAL(reprod.) + cs	47.1	15.6
MURAL(reprod.) + fr	45.4	13.9
MURAL(reprod.) + ja	47.6	16.0
MURAL(reprod.) + zh	47.1	15.6
MURAL(reprod.) + de	46.1	14.6

Table 5: Performance on Korean MS-COCO image-text and text-image retrieval tasks. Scores are the average of T2I and I2T R@1,5,10.

Model	Avg	Gain over baseline
MURAL(reprod.) baseline	40.6	-
MURAL(reprod.) + cs	54.3	13.7
MURAL(reprod.) + fr	53.2	12.6
MURAL(reprod.) + ja	52.5	11.9
MURAL(reprod.) + zh	52.3	11.7
MURAL(reprod.) + de	52.4	11.8

Table 6: Performance on Ukraine MS-COCO image-text and text-image retrieval tasks. Scores are the average of T2I and I2T R@1,5,10.

Model	3 COCO datasets	4 Flickr30K datasets
MURAL(reprod.)	139.9	294.4
MURAL(reprod.) + 6 lang	179.3	343.9
MURAL(reprod.) + ETCL	180.5	344.2

Table 7: Performance on MS-COCO(en, ja, ko) and Flickr30K(en, de, fr, cs) in zero-shot image-text retrieval tasks. Scores are the summation of averages of 6 scores (T2I and I2T R@1,5,10).

5.4 Analysis and ablation study

Observing the overall performance in Table 1, one could be convinced that adding more multilingual image-text pairs during ETCL will bring better performance. Then, one can ask some questions (1) does our model really transfer VL knowledge even for unseen language?, (2) how does zero-shot transfer differ depending on linguistic context?. Further experiments and analyses are performed to gain a better understanding of multi-modal zero-shot transfer. In addition, we investigate the effectiveness of triple contrastive loss. Another ablation study on VL pre-training steps is described in Appendix A.3.

Effect of the multi-modal zeroshot-transfer In order to answer questions (1) and (2), we fine-tune MURAL(reprod.) with $\mathcal{L}_{\text{image-text}}$ using one language pairs from multilingual CC3M and compare models on the Korean and Ukraine [6] COCO image-text retrieval. As shown in rows 2-6 in both Table 5 and Table 6, additional training using ($\mathcal{L}_{\text{image-text}}$) brings zero-shot transfer gain over both unseen Korean and Ukraine. Interestingly, additional training using Japanese image-text pairs shows the largest gain over other languages for Korean COCO. We conjecture that Japanese is a language very similar to Korean grammatically, resulting in enhanced zero-shot transfer. Likewise, Czech, similar to Ukraine geographically, also has the largest zero-shot transfer gain in Ukraine COCO. Another ablation study is included in Appendix A.5.

Effectiveness of triple contrastive loss One can expect that the cross-lingual zero-shot transfer effect can sufficiently occur with only the $\mathcal{L}_{\text{image-text}}$. To investigate the effect of the triple contrastive loss, we fine-tune MURAL(reprod.) with $\mathcal{L}_{\text{image-text}}$ using all six multilingual CC3M (we name it MURAL(reprod.)+6lang). Furthermore, we compare it with MURAL(reprod.)+ETCL on zero-shot image-text retrieval for multilingual MS-COCO (en, ja, ko) and Flickr30K (en, de, fr, cs). As shown in Table 7, row 1-2 show that cross-lingual zero-shot transfer also occurs in MURAL(reprod.)+6lang as expected. However, MURAL(reprod.)+ETCL still show the best performance, proving the effectiveness of triple contrastive loss.

6 Conclusion

VL modeling using large-scale web-crawled datasets has shown great success but cannot be easily utilized for low-resourced languages due to the lack of data. Although a method to efficiently create a multilingual VL model through indirect text-text alignment has been proposed, the VL representations of low-resourced languages are still

[6] We translate MS-COCO into Ukraine using google translator.

weaker than that of English. This work proposes a new approach to solve the limitation based on the observation that multi-modal zero-shot transfer occurs with regard to grammatical and geographical relationships between languages. Our proposed method can be easily adapted to multilingual multi-modal models trained similarly to MURAL.

7 Ethical consideration

It is likely that our model has unwanted social bias from the majority of English in our dataset. If an objective or a dataset that strongly increases the contribution of low-resource language is complemented, the bias can be alleviated.

In our approach, a larger model size, more data, and longer training steps lead to better models. However, these lead to an environmental impact inevitably. Therefore, more research is required to develop a large-scale multilingual multi-modal model with fewer steps and a small backbone to alleviate tremendous computing resources harming our environment.

References

Mikel Artetxe and Holger Schwenk. 2019. Massively multilingual sentence embeddings for zero-shot cross-lingual transfer and beyond. *Trans. Assoc. Comput. Linguistics*.

Loïc Barrault, Fethi Bougares, Lucia Specia, Chiraag Lala, Desmond Elliott, and Stella Frank. 2018. Findings of the third shared task on multimodal machine translation. In *Proceedings of the Third Conference on Machine Translation: Shared Task Papers*.

Lukas Bossard, Matthieu Guillaumin, and Luc Van Gool. 2014. Food-101 – mining discriminative components with random forests.

Tom B. Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, T. J. Henighan, Rewon Child, Aditya Ramesh, Daniel M. Ziegler, Jeff Wu, Clemens Winter, Christopher Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. 2020. Language models are few-shot learners. *ArXiv*, abs/2005.14165.

Daniel Matthew Cer, Mona T. Diab, Eneko Agirre, Iñigo Lopez-Gazpio, and Lucia Specia. 2017. Semeval-2017 task 1: Semantic textual similarity multilingual and crosslingual focused evaluation. In *SemEval@ACL*.

Soravit Changpinyo, Piyush Sharma, Nan Ding, and Radu Soricut. 2021. Conceptual 12M: Pushing web-scale image-text pre-training to recognize long-tail visual concepts. In *CVPR*.

Ting Chen, Simon Kornblith, Mohammad Norouzi, and Geoffrey E. Hinton. 2020. A simple framework for contrastive learning of visual representations.

Zewen Chi, Li Dong, Shuming Ma, Shaohan Huang, Xian-Ling Mao, Heyan Huang, and Furu Wei. 2021. mt6: Multilingual pretrained text-to-text transformer with translation pairs.

Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. 2009. Imagenet: A large-scale hierarchical image database.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding.

Desmond Elliott, Stella Frank, Loïc Barrault, Fethi Bougares, and Lucia Specia. 2017. Findings of the second shared task on multimodal machine translation and multilingual image description. In *Proceedings of the Second Conference on Machine Translation*.

Desmond Elliott, Stella Frank, Khalil Sima'an, and Lucia Specia. 2016. Multi30K: Multilingual English-German image descriptions. In *Proceedings of the 5th Workshop on Vision and Language*.

Fangxiaoyu Feng, Yinfei Yang, Daniel Cer, Naveen Arivazhagan, and Wei Wang. 2020. Language-agnostic BERT sentence embedding. *CoRR*, abs/2007.01852.

Priya Goyal, Mathilde Caron, Benjamin Lefaudeux, Min Xu, Pengchao Wang, Vivek Pai, Mannat Singh, Vitaliy Liptchinsky, Ishan Misra, Armand Joulin, and Piotr Bojanowski. 2021. Self-supervised pretraining of visual features in the wild. *arXiv*.

Jiyeon Ham and Eun-Sol Kim. 2021. Semantic alignment with calibrated similarity for multilingual sentence embedding. In *Findings of the Association for Computational Linguistics: EMNLP 2021*.

Kaiming He, Haoqi Fan, Yuxin Wu, Saining Xie, and Ross B. Girshick. 2020. Momentum contrast for unsupervised visual representation learning.

Junjie Hu, Sebastian Ruder, Aditya Siddhant, Graham Neubig, Orhan Firat, and Melvin Johnson. 2020. Xtreme: A massively multilingual multi-task benchmark for evaluating cross-lingual generalization. *CoRR*.

Aashi Jain, Mandy Guo, Krishna Srinivasan, Ting Chen, Sneha Kudugunta, Chao Jia, Yinfei Yang, and Jason Baldridge. 2021. Mural: Multimodal, multitask retrieval across languages. *arXiv*.

Chao Jia, Yinfei Yang, Ye Xia, Yi-Ting Chen, Zarana Parekh, Hieu Pham, Quoc V. Le, Yun-Hsuan Sung, Zhen Li, and Tom Duerig. 2021. Scaling up visual and vision-language representation learning with noisy text supervision. In *ICML*.

Andrej Karpathy and Li Fei-Fei. 2015. Deep visual-semantic alignments for generating image descriptions. In *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 3128–3137.

Wonjae Kim, Bokyung Son, and Ildoo Kim. 2021. Vilt: Vision-and-language transformer without convolution or region supervision.

Byungsoo Ko and Geonmo Gu. 2022. Large-scale bilingual language-image contrastive learning. *arXiv preprint arXiv:2203.14463*.

Alexander Kolesnikov, Lucas Beyer, Xiaohua Zhai, Joan Puigcerver, Jessica Yung, Sylvain Gelly, and Neil Houlsby. 2020. Large scale learning of general visual representations for transfer.

Ranjay Krishna, Yuke Zhu, Oliver Groth, Justin Johnson, Kenji Hata, Joshua Kravitz, Stephanie Chen, Yannis Kalantidis, Li-Jia Li, David A Shamma, Michael Bernstein, and Li Fei-Fei. 2016. Visual genome: Connecting language and vision using crowdsourced dense image annotations. *arXiv*.

Alex Krizhevsky. 2009. Learning multiple layers of features from tiny images.

Alina Kuznetsova, Hassan Rom, Neil Alldrin, Jasper R. R. Uijlings, Ivan Krasin, Jordi Pont-Tuset, Shahab Kamali, Stefan Popov, Matteo Malloci, Tom Duerig, and Vittorio Ferrari. 2020. The open images dataset V4: unified image classification, object detection, and visual relationship detection at scale.

Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Ves Stoyanov, and Luke Zettlemoyer. 2019. Bart: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension.

Liunian Harold Li, Mark Yatskar, Da Yin, Cho-Jui Hsieh, and Kai-Wei Chang. 2019. Visualbert: A simple and performant baseline for vision and language. *arXiv*.

Tsung-Yi Lin, Michael Maire, Serge Belongie, Lubomir Bourdev, Ross Girshick, James Hays, Pietro Perona, Deva Ramanan, C. Lawrence Zitnick, and Piotr Dollár. 2014. Microsoft coco: Common objects in context.

Yinhan Liu, Jiatao Gu, Naman Goyal, Xian Li, Sergey Edunov, Marjan Ghazvininejad, Mike Lewis, and Luke Zettlemoyer. 2020. Multilingual denoising pre-training for neural machine translation. *Transactions of the Association for Computational Linguistics*, 8:726–742.

Jiasen Lu, Dhruv Batra, Devi Parikh, and Stefan Lee. 2019. Vilbert: Pretraining task-agnostic visiolinguistic representations for vision-and-language tasks.

Jiasen Lu, Vedanuj Goswami, Marcus Rohrbach, Devi Parikh, and Stefan Lee. 2020. 12-in-1: Multi-task vision and language representation learning.

Telmo Pires, Eva Schlinger, and Dan Garrette. 2019. How multilingual is multilingual BERT? In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*.

Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, Gretchen Krueger, and Ilya Sutskever. 2021. Learning transferable visual models from natural language supervision.

Alec Radford, Jeff Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. 2019. Language models are unsupervised multitask learners.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J. Liu. 2020. Exploring the limits of transfer learning with a unified text-to-text transformer. *Journal of Machine Learning Research*, 21(140):1–67.

Nils Reimers and Iryna Gurevych. 2019. Sentence-bert: Sentence embeddings using siamese bert-networks. *ArXiv*, abs/1908.10084.

Nils Reimers and Iryna Gurevych. 2020. Making monolingual sentence embeddings multilingual using knowledge distillation.

Yue Ruan, Han-Hung Lee, Ke Zhang, and Angel X Chang. 2022. Tricolo: Trimodal contrastive loss for fine-grained text to shape retrieval. *arXiv preprint arXiv:2201.07366*.

Holger Schwenk, Guillaume Wenzek, Sergey Edunov, Edouard Grave, Armand Joulin, and Angela Fan. 2021. CCMatrix: Mining billions of high-quality parallel sentences on the web.

Piyush Sharma, Nan Ding, Sebastian Goodman, and Radu Soricut. 2018. Conceptual captions: A cleaned, hypernymed, image alt-text dataset for automatic image captioning.

Mingxing Tan and Quoc V. Le. 2019. Efficientnet: Rethinking model scaling for convolutional neural networks.

Jianxiong Xiao, James Hays, Krista A. Ehinger, Aude Oliva, and Antonio Torralba. 2010. Sun database: Large-scale scene recognition from abbey to zoo.

Linting Xue, Noah Constant, Adam Roberts, Mihir Kale, Rami Al-Rfou, Aditya Siddhant, Aditya Barua, and Colin Raffel. 2021. mt5: A massively multilingual pre-trained text-to-text transformer.

Yinfei Yang, Gustavo Hernandez Abrego, Steve Yuan, Mandy Guo, Qinlan Shen, Daniel Cer, Yun hsuan Sung, Brian Strope, and Ray Kurzweil. 2019. Improving multilingual sentence embedding using bi-directional dual encoder with additive margin softmax.

Yinfei Yang, Daniel Cer, Amin Ahmad, Mandy Guo, Jax Law, Noah Constant, Gustavo Hernandez Abrego, Steve Yuan, Chris Tar, Yun-hsuan Sung, Brian Strope, and Ray Kurzweil. 2020. Multilingual universal sentence encoder for semantic retrieval. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics: System Demonstrations*.

Yuya Yoshikawa, Yutaro Shigeto, and Akikazu Takeuchi. 2017. Stair captions: Constructing a large-scale japanese image caption dataset.

Yang You, Jing Li, Sashank Reddi, Jonathan Hseu, Sanjiv Kumar, Srinadh Bhojanapalli, Xiaodan Song, James Demmel, Kurt Keutzer, and Cho-Jui Hsieh. 2020. Large batch optimization for deep learning: Training bert in 76 minutes.

Peter Young, Alice Lai, Micah Hodosh, and Julia Hockenmaier. 2014. From image descriptions to visual denotations: New similarity metrics for semantic inference over event descriptions. *Transactions of the Association for Computational Linguistics*.

Xiaohua Zhai, Joan Puigcerver, Alexander Kolesnikov, Pierre Ruyssen, Carlos Riquelme, Mario Lucic, Josip Djolonga, André Susano Pinto, Maxim Neumann, Alexey Dosovitskiy, Lucas Beyer, Olivier Bachem, Michael Tschannen, Marcin Michalski, Olivier Bousquet, Sylvain Gelly, and Neil Houlsby. 2019. The visual task adaptation benchmark. *arXiv*.

Mingyang Zhou, Luowei Zhou, Shuohang Wang, Yu Cheng, Linjie Li, Zhou Yu, and Jingjing Liu. 2021. Uc2: Universal cross-lingual cross-modal vision-and-language pre-training. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 4155–4165.

A Appendix

A.1 Image-text retrieval

Fine-tuning on Flickr30K and MSCOCO took about 3 and 6 hours, respectively using 32 Cloud TPU V3 cores. All scores of both image-text and text-image retrieval are listed in Table 8 and 9.

A.2 Validation of visual embedding

To validate the visual embedding only, we conduct the ImagenetKNN retrieval task using visual features from our pre-trained model as done in ALIGN. In ImageNet KNN retrieval, we retrieve their nearest neighbors from the training set using pre-trained visual embeddings to find the class of each image in the validation set of ILSVRC-2012. Recall@K metric is obtained using the appearance of the found label of the query image in the top-K retrieved images. We compare our model with ALIGN as shown in Table 10. Because we use image-text pair datasets less than ALIGN by 0.8B pairs, our model shows a score of 67.8 which is still comparable, indicating that our model trained using two alignment task learning ensures powerful visual embedding like ALIGN which is trained with only image-text alignment.

Type	Language	image → text			text → image		
		R@1	R@5	R@10	R@1	R@5	R@10
Zero-shot	en	84.9	97.7	99.5	73.5	92.2	96.0
	de	77.5	96.0	98.4	63.0	86.6	91.9
	fr	70.8	90.6	95.0	72.0	91.2	95.3
	cs	62.8	87.8	93.0	67.2	88.8	93.5
Fine-tune	en	93.5	99.4	99.8	81.8	95.9	98.2
	de	88.1	98.7	99.7	72.1	91.6	95.7
	fr	80.1	96.3	97.9	80.0	95.9	97.8
	cs	77.1	95.0	97.0	77.1	94.5	97.3

Table 8: Image-text retrieval on Flickr30K in multiple languages.

Type	Language	image → text			text → image		
		R@1	R@5	R@10	R@1	R@5	R@10
Zero-shot	en	56.1	80.2	87.5	43.5	69.1	78.4
	ja	35.3	59.9	70.9	25.0	47.3	57.5
	ko	47.1	72.5	82.1	36.8	62.0	72.0
Fine-tune	en	70.2	91.0	95.3	54.9	80.0	87.7
	ja	62.9	87.0	93.2	46.8	73.9	82.9
	ko	69.8	90.8	95.1	53.1	79.1	87.1

Table 9: Image-text retrieval on COCO in multiple languages.

Model	Data	Backbone	ImageNet KNN R@1
ALIGN	1.8B	B7 + BERT-base	69.3
MURAL(reprod.) + ETCL	1.0B + 15M	B7 + BERT-base	67.8

Table 10: Performance on ImagenetKNN retrieval task.

A.3 Ablation study

We investigate the effect of VL pre-training steps with regard to three domain tasks including visual, VL, and language. We select ImageNetKNN and Multi30K(en, de) and STS tasks for three domain respectively.

Performance dependency on VL pre-training steps Table 11 shows the ablation study of VL pre-training steps on three domain tasks. Generally, the longer VL pre-training steps give the stronger performance after ETCL. Interestingly, in the case

Row	VL pre-training steps	ETCL steps	ImageNetKNN R@1	MS-COCO (zero-shot) en(Avg.)	ja(Avg.)	ko(Avg.)	STS Meta avg
1	100K	-	60.5	61.5	29.6	24.0	68.1
2	200K	-	64.0	65.1	34.1	29.1	71.5
3	400K	-	66.0	67.2	37.0	30.1	73.7
4	600K	-	67.5	68.4	40.1	34.1	73.2
5	800K	-	66.9	67.8	40.6	31.5	74.6
6	100K	3K	60.7	61.5	53.2	35.7	78.9
7	200K	3K	64.1	65.2	57.2	42.8	79.9
8	400K	3K	65.6	67.5	57.2	45.7	81.9
9	600K	3K	67.7	69.1	60.3	**51.9**	**81.6**
10	800K	3K	**67.8**	**69.1**	**62.1**	49.3	81.5

Table 11: Performance on visual, VL and language domain tasks with regard to VL pre-training steps.

of Korean MS-COCO, the model after VL pre-training 100K + ETCL has a higher performance by 4.2 than when the model trained after VL pre-training 800K steps. When considering that there is no image and Korean sentence pair in the additional datasets used in ETCL, the result shows that ECTL learns multilingual VL representations very effectively. Likewise, a similar phenomenon is shown in the STS sentence embedding task. This is a good example showing that the visual representation obtained from VL training can be used to improve the performance of text representation.

Model	14 langs	36 langs	82 langs	All langs
LASER	95.3	84.4	75.9	65.5
m-USE	93.0	44.3	38.5	36.6
LaBSE	**95.3**	**95.0**	**87.3**	**83.7**
(Reimers and Gurevych, 2019)	94.8	86.2	75.6	67.0
(Ham and Kim, 2021)	95.4	89.1	79.4	72.9
MURAL(reprod.) + ETCL	88.4	81.5	72.4	63.6

Table 12: Performance on Tatoeba sentence retrieval task. Scores are reported by 100 × accuracy. We follow the grouping '14 langs', '36 langs', and '82 langs' as used in m-USE, XTREME and LASER respectively.

A.4 Zero-shot image classification

Text prompts engineering We use 80 text templates as used in CLIP [7]. For fine-grained datasets, *"a type of food"* are appended to the initial template for adding context information. All multilingual zero-shot ext prompts image classification results in various visual datasets are shown in Table 14.

[7]https://github.com/openai/CLIP

A.5 Multi-modal zero-shot transfer

To further investigate the effect of multi-modal zero-shot transfer, we calculate the performance of fine-tuned `MURAL(reprod.)` with one language pair from multilingual CC3M on Italian MS-COCO. Interestingly, the performance gains via `ETCL` using the image and regionally closer languages (cs, fr, de) pairs are larger than that of Asian languages (zh, ja), which are regionally far from Italian (See Table 13).

Model	Avg	Gain over baseline
MURAL(reprod.) baseline	40.6	-
MURAL(reprod.) + cs	60.3	19.7
MURAL(reprod.) + fr	60.2	19.6
MURAL(reprod.) + ja	59.7	19.1
MURAL(reprod.) + zh	59.3	18.7
MURAL(reprod.) + de	60.1	19.5

Table 13: Performance on Italian MS-COCO image-text and text-image retrieval tasks. Scores are the average of T2I and I2T R@1,5,10.

A.6 Additional NLP task

Tatoeba For the multilingual sentence retrieval task, we use Tatoeba dataset composed of 1,000 English-aligned sentence pairs for 112 languages (Artetxe and Schwenk, 2019). This task is to find the nearest neighbor for each sentence in another language using the cosine similarity. We conduct an evaluation on three groupings of languages for fair-comparison: the first 14 language groups are selected for m-USE (Yang et al., 2020). The second language group with 36 languages follows the XTREME benchmarks (Hu et al., 2020). The third 82 language group that

LASER proposed covers high-resourced to low-resourced languages. Table 12 shows the average accuracy of languages according to different groupings. Our model generally has similar performance compared to LASER, and lower performance than LaBSE. That is because compared to LaBSE and LASER which learns 109 and 93 languages respectively, our model learns 90 languages, and then our model does not support several languages in Tatoeba dataset. For example, Thai language is included in the 14 languages group while the language is not included in the CCMatrix we learned, which leads to a poor performance of 89.2%. The average of 13 languages excluding the language is 94.2%, which is comparable to the top language models. This poor performance from data scarcity also appears for 36 langs groups. The average accuracy of the 36 langs group except for Swahili, Telugu, and Thai is 86.7%, which is better than other models except for LaBSE. Performance degradation due to lack of data in some languages is not relevant to our methodology itself, so it does not impair the novelty of this work. Rather, it is likely that supplementing from other datasets in opus can boost our model. The results for all languages can be found in Table 15, 16.

	af	am	ar	az	be	bg	bn	ca	ceb	cs	cy	da	de
ImageNet	16.5	2.5	18.7	12.9	14.3	21.6	16.3	20.0	21.9	24.6	14.0	30.0	29.3
SUN397	39.0	6.1	43.3	30.9	30.9	46.9	40.2	46.5	33.0	57.2	27.6	56.4	59.5
Food101	51.2	9.6	34.7	47.7	48.9	43.3	47.6	49.6	50.5	62.0	42.6	61.2	66.6
CIFAR100	19.8	6.8	23.7	17.2	15.2	26.0	21.7	23.2	14.9	31.3	10.6	27.2	31.6
	el	en	eo	es	et	eu	fa	fi	fr	fy	ga	gd	gl
ImageNet	18.3	62.6	23.2	25.5	20.2	15.3	15.7	19.7	26.5	24.5	10.4	16.7	28.7
SUN397	45.5	67.2	43.7	48.9	43.5	37.1	40.1	43.3	59.9	37.7	21.0	27.4	58.5
Food101	37.9	74.2	63.8	53.7	40.6	49.4	42.1	47.4	64.6	49.3	45.1	47.2	64.3
CIFAR100	23.8	33.6	25.2	27.6	19.6	19.3	22.7	25.6	32.3	18.0	7.3	14.1	31.1
	ha	he	hl	hr	hu	hy	id	ig	is	it	ja	jv	ka
ImageNet	13.9	14.2	9.8	22.1	20.2	5.8	26.3	12.9	23.4	28.3	17.6	28.2	9.0
SUN397	23.5	35.8	28.9	47.4	46.9	21.3	45.9	21.4	42.8	51.4	52.0	44.0	19.6
Food101	53.5	31.7	31.9	46.4	52.3	33.7	58.3	43.2	54.6	59.4	40.0	51.8	39.4
CIFAR100	13.4	13.7	20.4	23.5	24.7	10.1	26.7	11.1	19.3	25.8	29.8	22.4	9.1
	kk	km	ko	la	lb	lt	lv	mg	mk	ml	mr	ms	my
ImageNet	9.7	5.8	14.1	29.1	20.0	22.8	20.8	20.4	21.8	6.0	8.1	30.3	5.8
SUN397	20.5	12.7	40.0	36.5	36.2	45.1	34.8	40.1	38.3	18.4	23.0	53.6	13.8
Food101	33.4	32.0	33.9	58.9	50.6	52.3	47.8	62.8	51.6	19.8	35.1	60.8	25.1
CIFAR100	14.2	4.8	21.8	20.9	18.6	24.2	19.4	17.5	24.3	12.9	16.9	28.3	4.5
	ne	nl	no	or	pl	pt	ro	ru	sd	si	sk	sl	so
ImageNet	6.2	27.9	28.8	1.1	23.4	26.3	25.9	22.6	2.8	11.0	23.1	28.5	10.2
SUN397	15.4	53.6	54.5	6.3	48.6	50.1	50.5	47.1	8.3	24.7	53.7	50.5	14.1
Food101	15.6	60.9	60.8	10.6	54.3	59.0	53.6	38.3	10.2	29.1	51.1	53.8	38.4
CIFAR100	11.0	26.5	26.5	4.4	25.0	27.3	24.7	26.6	8.6	18.6	28.2	25.7	8.6
	sq	sr	su	sv	sw	ta	tl	tr	tt	uk	ur	uz	vi
ImageNet	21.3	17.0	17.7	25.5	22.5	6.9	36.3	19.7	9.4	18.1	9.8	9.1	23.6
SUN397	39.1	35.7	30.2	53.1	45.2	24.0	53.5	44.1	24.1	46.0	31.5	21.0	49.1
Food101	52.1	17.5	44.7	54.7	65.5	45.7	66.3	50.8	25.0	40.1	46.7	25.1	50.5
CIFAR100	20.6	23.4	18.5	25.9	24.9	10.4	24.8	22.2	13.6	24.5	16.3	12.3	24.8
	xh	yi	yo	zh	zu								
ImageNet	12.1	1.8	7.4	21.8	11.5								
SUN397	17.3	8.6	13.9	56.0	20.9								
Food101	36.9	17.7	35.7	44.7	36.3								
CIFAR100	10.3	4.1	5.1	30.9	6.7								

Table 14: Zero-shot image classification on ImageNet, SUN397 Food101 and CIFAR100. Scores are 100 × Acc@1.

Model	ar (ara)	bg (bul)	ca (cat)	cs (ces)	cmn	da (dan)	de (deu)	el (ell)	et (est)	fi (fin)	fr (fra)	gl (glg)	he (heb)
LASER	92.0	95.0	95.9	96.5	95.4	96.0	99.0	95.0	96.7	96.3	95.6	95.5	92.2
m-USE	81.0	54.0	66.3	17.8	94.3	25.9	98.2	1.6	8.4	8.2	93.5	82.2	1.8
LaBSE	91.0	95.7	96.5	97.5	96.2	96.4	99.4	96.6	97.7	97.0	96.0	97.2	93.0
MURAL(reprod.) + ETCL	89.8	95.1	96.6	96.6	95.6	96.7	98.8	96.2	96.3	96.4	95.5	95.4	91.5
Model	hi (hin)	hr (hrv)	hu (hun)	hy (hye)	id (ind)	it (ita)	ja (jpn)	ka (kat)	ko (kor)	lt (lit)	lvs	mr (mar)	mk (mkd)
LASER	94.7	97.2	96.0	36.1	94.5	95.3	90.7	35.9	88.9	96.2	95.4	91.5	94.7
m-USE	1.2	23.9	10.2	1.7	93.3	94.3	93.8	2.6	86.0	10.2	11.1	1.8	33.1
LaBSE	97.7	97.8	97.2	95.0	95.3	94.6	96.4	95.9	93.5	97.3	96.8	94.8	94.8
MURAL(reprod.) + ETCL	95.9	97.6	95.7	82.2	94.5	94.3	95.6	66.8	91.7	96.5	93.9	91.8	92.8
Model	mn (mon)	nl (nld)	nb (nob)	pes	pl (pol)	pt (por)	ro (ron)	ru (rus)	sk (slk)	sl (slv)	es (spa)	sq (sqi)	sr (srp)
LASER	8.2	96.3	98.8	93.4	97.8	95.2	97.4	94.6	96.6	95.9	98.0	98.0	95.3
m-USE	16.9	94.0	23.9	12.7	93.7	94.9	30.0	93.7	21.1	20.9	95.4	19.9	27.7
LaBSE	96.6	97.2	98.9	96.0	97.8	95.6	97.8	95.3	97.3	96.7	98.4	97.6	96.2
MURAL(reprod.) + ETCL	18.2	96.5	98.3	95.8	96.7	95.7	96.6	94.6	96.3	96.4	97.5	97.4	94.5
Model	sv (swe)	th (tha)	tr (tur)	uk (ukr)	ur (urd)	vi (vie)	yue	zsm					
LASER	96.6	95.4	97.5	94.5	81.9	96.8	90.0	96.4					
m-USE	18.8	96.0	94.0	51.0	6.4	10.4	84.2	89.1					
LaBSE	96.5	97.1	98.4	95.2	95.3	97.8	92.1	96.9					
MURAL(reprod.) + ETCL	96.4	8.6	98.0	94.9	86.6	97.5	88.2	96.5					

Table 15: Performance on Tatoeba sentence retrieval task for languages. Scores are reported by 100 × accuracy. To make comparisons to other works easily, language abbreviations are expressed using ISO 639-1/639-2/639-3.

Model	af (afr)	am (amh)	ang	arq	arz	ast	awa	az (aze)	be (bel)	bn (ben)	ber	bs (bos)	br (bre)
LASER	89.5	42.0	37.7	39.5	68.9	86.2	36.1	66.0	66.1	89.6	68.2	96.5	15.8
m-USE	63.5	2.1	38.1	28.2	59.6	81.5	2.4	42.2	40.3	0.7	8.3	30.1	10.2
LaBSE	97.4	94.0	64.2	46.2	78.4	90.6	73.2	96.1	96.2	91.3	10.4	96.2	17.3
MURAL(reprod.) + ETCL	92.5	45.2	50.0	42.3	74.2	86.6	44.2	81.6	74.1	89.5	9.6	97.5	12.0
Model	cbk	ceb	ch (cha)	kw (cor)	csb	cy (cym)	dsb	dtp	eo (epo)	eu (eus)	fo (fao)	fy (fry)	gd (gla)
LASER	77.0	15.7	29.2	7.5	43.3	8.6	48.0	7.2	97.2	94.6	71.6	51.7	3.7
m-USE	76.1	13.7	33.6	6.4	37.4	13.1	35.1	8.4	36.8	19.4	18.7	52.3	6.9
LaBSE	82.5	70.9	39.8	12.8	56.1	93.6	69.3	13.3	98.4	95.8	90.6	89.9	88.8
MURAL(reprod.) + ETCL	77.9	22.0	32.9	7.4	45.5	13.7	59.7	11.2	97.4	94.0	72.1	61.9	6.0
Model	ga (gle)	gsw	hsb	io (ido)	ie (ile)	ia (ina)	is (isl)	jv (jav)	kab	kaz	km (khm)	ku (kur)	kzj
LASER	5.2	44.4	54.5	83.7	86.2	95.2	95.6	22.9	58.1	18.6	20.6	17.2	7.2
m-USE	7.7	39.3	33.3	55.5	73.3	86.7	10.3	38.3	3.7	15.3	1.5	21.7	10.2
LaBSE	95.0	52.1	71.2	90.9	87.1	95.8	96.2	84.4	6.2	90.5	83.2	87.1	14.2
MURAL(reprod.) + ETCL	7.2	45.3	65.0	82.7	86.1	93.6	94.9	39.0	4.6	56.0	58.2	21.0	10.6
Model	la (lat)	lfn	ml (mal)	max	mhr	nds	nn (nno)	nov	oc (oci)	orv	pam	pms	swg
LASER	58.5	64.5	96.9	50.9	10.4	82.9	88.3	66.0	61.2	28.1	6.0	49.6	46.0
m-USE	36.7	60.5	1.2	65.0	14.3	57.5	21.2	66.1	42.9	28.3	8.4	48.8	48.7
LaBSE	82.0	71.2	98.9	71.1	19.2	81.2	95.9	78.2	69.9	46.8	13.6	67.0	65.2
MURAL(reprod.) + ETCL	61.2	68.5	97.4	59.9	15.0	68.6	88.8	73.2	63.9	38.9	8.9	56.6	59.8
Model	swh	ta (tam)	tt (tat)	te (tel)	tl (tgl)	tk (tuk)	tzl	ug (uig)	uz (uzb)	war	wuu	xh (xho)	yjd
LASER	57.6	69.4	31.1	79.7	50.6	20.7	44.7	45.2	18.7	13.6	87.7	8.5	5.7
m-USE	13.7	2.8	15.7	2.4	16.2	20.9	46.6	4.0	15.9	15.6	82.2	14.8	1.9
LaBSE	88.6	90.7	97.9	98.3	97.4	80.0	63.0	93.7	86.8	65.3	90.3	91.9	91.0
MURAL(reprod.) + ETCL	71.3	77.2	27.8	4.3	62.3	27.6	52.9	6.0	29.7	24.9	87.3	10.6	8.7

Table 16: Performance on Tatoeba sentence retrieval task for languages. Scores are reported by 100 × accuracy. To make comparisons to other works easily, language abbreviations are expressed using ISO 639-1/639-2/639-3.

LOViS: Learning Orientation and Visual Signals for Vision and Language Navigation

Yue Zhang
Michigan State University
zhan1624@msu.edu

Parisa Kordjamshidi
Michigan State University
kordjams@msu.edu

Abstract

Understanding spatial and visual information is essential for a navigation agent who follows natural language instructions. The current Transformer-based VLN agents entangle the orientation and vision information, which limits the gain from the learning of each information source. In this paper, we design a neural agent with explicit Orientation and Vision modules. Those modules learn to ground spatial information and landmark mentions in the instructions to the visual environment more effectively. To strengthen the spatial reasoning and visual perception of the agent, we design specific pre-training tasks to feed and better utilize the corresponding modules in our final navigation model. We evaluate our approach on both Room2room (R2R) and Room4room (R4R) datasets and achieve the state of the art results on both benchmarks.

1 Introduction

Vision and Language Navigation (VLN) problem (Anderson et al., 2018) has attracted increasing attention from the communities of computer vision, natural language processing, and robotics because of its broad real-world applications. In this problem setting, the goal of a navigation agent is to move to a target location in a photo-realistic simulated environment by following a detailed natural language instruction, e.g., *"Walk into the bedroom. Walk past the bedroom door. Wait at the laundry room door."*. Two kinds of simulators are used to create the dataset and the corresponding problem formulation: discrete trajectories (Anderson et al., 2018) and continuous navigation trajectories (Krantz et al., 2020). In this paper, we work on the discrete one, that an agent traverses a predefined connectivity graph by selecting the adjacent viewpoint with a higher probability of corresponding to the instruction at each step.

The earlier research in the VLN area can be divided into two categories. The first category of models mostly depends on the encoder-decoder framework for encoding the text and visual information, establishing the connections between them with the attention mechanism, and decoding the actions (Anderson et al., 2018; Ma et al., 2019; Fried et al., 2018). The second category of works learns to model the semantic structure which implicitly or explicitly enhances the textual-visual matching (Hong et al., 2020a,b; Qi et al., 2020; Zhang et al., 2021; Li et al., 2021; Zhang and Kordjamshidi, 2022). However, these prior research are surpassed by the most recently proposed Transformer-based VLN agents (Hong et al., 2021; Hao et al., 2020; Guhur et al., 2021; Chen et al., 2021) which capture the cross-modality information and demonstrate an outstanding navigation performance.

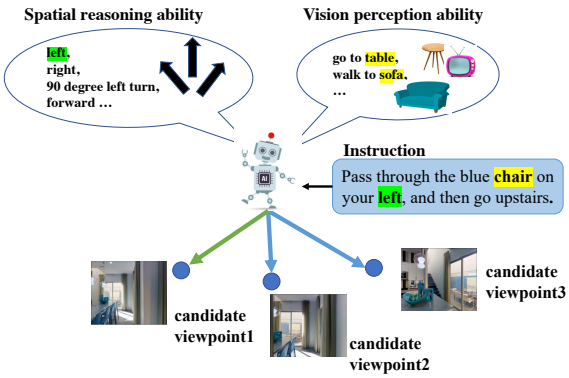

Figure 1: Spatial reasoning helps leveraging orientation clues, such as *left* and *90 degree*; visual perception ability grounds mentioned landmarks, such as *table*, *sofa*, and *chair*. With these two abilities, the agent selects the candidate viewpoint corresponding to the instruction at each navigation step. The green arrow shows the ground-truth viewpoint.

As shown in the example of Figure 1, two major abilities are important to the navigation agent: *spatial reasoning* and *visual perception*. While navigation seems to require both of these abilities, there are cases where understanding even one of these is sufficient. For example, *spatial reasoning* guides the agent towards the correct direction when the instruction is "*90-degree left-turn*" or "*on your*

Proceedings of the 29th International Conference on Computational Linguistics, pages 5745–5754
October 12–17, 2022.

right", regardless of the surrounding visual scene or objects. In some other cases, *visual perception* is sufficient to recognize the mentioned landmarks in the visual environment after receiving instructions without any auxiliary signals of orientation, such as "*walk to the sofa*" or "*pass the table*".

The current Transformer-based agents tend to intertwine the learning of these two abilities, which we argue may impede developing a more effective navigation model. In the light of this, we propose a new navigation agent with different modules to select actions based on orientation and vision perspectives separately. Moreover, we design specific pre-training tasks to distil more explicit spatial and visual knowledge, which is better utilized in the corresponding modules in our navigation agent. This is different from the majority of methods employing pre-training tasks without considering the needs of the target downstream tasks. Our modular design interacts with modular pre-training, guiding the agents to generate specialized features which can be better adapted to the downstream tasks.

Specifically, in the downstream navigation task, in order to utilize the learnt spatial and visual information, we design a framework, called LOViS. It contains three modules, namely, *history module*, *orientation module*, and *vision module*. In the *history module,* the agent uses the previous step's information to determine which tokens in the instruction it should pay attention to. And then, the agent learns to connect such attended tokens to the corresponding visual information to finally make a history-based action decision. In the *orientation module*, the agent only focuses on orientation information in the instruction (*e.g., left* and *90 degree*), and then grounds them to the vision environment to make an orientation-based action. Likewise, in the *vision module*, the agent is encouraged to focus on the mentioned landmarks in the instruction (*e.g., table*, *lamp* and *chair*), and ground them into the vision to obtain a vision-based action decision. Finally, the agent combines the action decisions of three modules to make the final decision.

In the pre-training process, we propose two specific pre-training tasks, namely *Orientation Matching* (OM) and *Vision Matching* (VM), to learn orientation and vision information, respectively. Besides, we modify two commonly used pre-training tasks for navigation: Masked Language Modeling (MLM) and Single Step Action Prediction (SSAP), to obtain better cross-modal representations for the downstream navigation model.

In summary, our contributions are as follows:
1. Unlike previous models, our novel Transformer-based agent includes **two new modules to capture the orientation and visual information signals separately**. This benefits the agent from both of these information sources to select an action more effectively.
2. We design **new pre-training tasks** to emphasize (a) learning spatial reasoning and grounding the orientation information in the environment; (b) learning visual perception and grounding landmark mentions in the environment. These pre-training representations are utilized in the corresponding modules in the navigation model.
3. Our method **exceeds the current SOTA** results on both *Room2room* and *Room4Room* benchmarks.

2 Related Work

Vision-and-Language Navigation Many deep learning methods (Tan et al., 2019; Hong et al., 2021, 2020a) for VLN tasks have been proposed in the past few years. For example, Anderson et al. (2018) firstly proposed a Sequence-to-Sequence baseline model to encode the instructions and decode the embeddings to the low-level action sequence with the observed images. SF (Fried et al., 2018) generates the augmented samples to address the generalization issues and extends the low-level space to panoramic action space. RelGraph (Hong et al., 2020a) builds an implicit language-visual entity relation graph to learn the connection between the text and vision modalities. EXOR (Zhang and Kordjamshidi, 2022) first splits the long instructions into spatial configurations (Dan et al., 2020; Zhang et al., 2021; Kordjamshidi et al., 2010). Then they explicitly align the landmarks and spatial relations in the spatial configuration to the corresponding information in the visual modality. In terms of learning spatial and vision information separately, OAAM (Qi et al., 2020) is the earliest attempt that decomposes the instruction into action and object phrases and related them to the visual environment to make the final decisions. NvEM (An et al., 2021) extends OAAM to divide object module to subject and reference modules and fuse the information from the neighbor views. However, to our best knowledge, there is no work investigating how to model spatial and visual information in the Transformer-based VLN agent.

Transformer-based Navigation Agent Compared

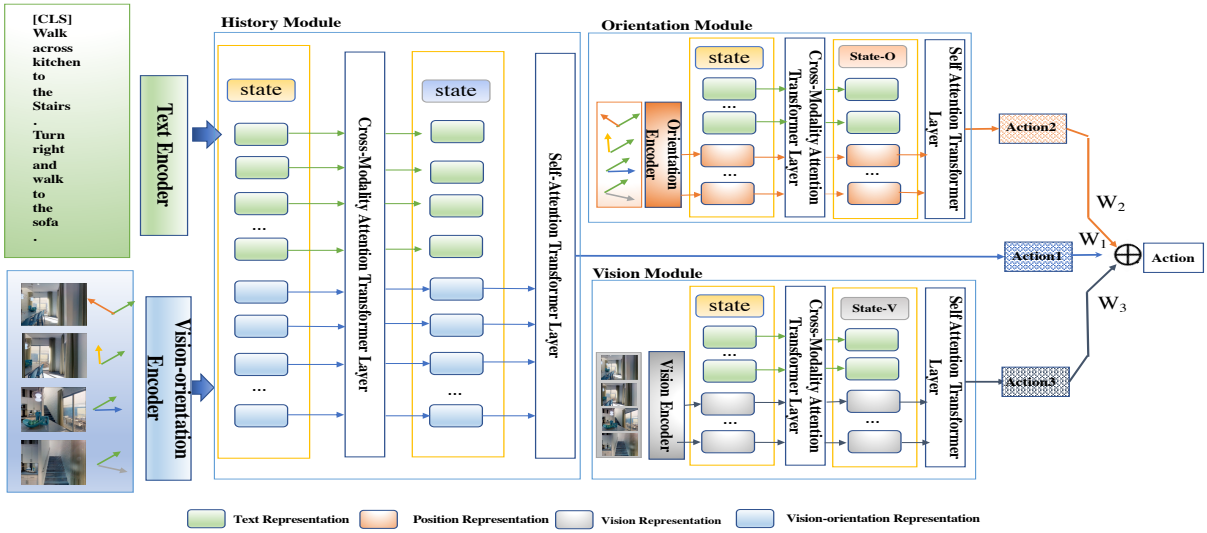

Figure 2: Our proposed LOViS contains three modules: history module, orientation module and vision module. Each module can generate action decision based on different reasoning, then three actions are combined to determine the final action selection.

with conventional methods, the Transformer-based model in VL tasks show great improvements (Tan and Bansal, 2019; Chen et al., 2019; Lu et al., 2019; Li et al., 2020). However, different from conventional VL tasks (*i.e.,* image captioning), the VLN task requires learned representation to facilitate the action selection, which is a Markov Decision Process. Therefore, the VLN needs to learn the correspondence between language and dynamic visual observation by interacting with the environment. In the past few years, the VLN task has been formulated as a dynamic grounding problem between texts and images. PRESS (Li et al., 2021) firstly fine-tunes a pre-trained language model BERT to obtain the text representation. PREVALENT (Hao et al., 2020) trains a VL Transformer with a large amount of image-text-action triplets to learn cross representations for the navigation task. RecBERT (Hong et al., 2021) designs a state unit to store history information and train Transformer recurrently for the direct navigation. HAMT (Chen et al., 2021) proposes to explicitly encode all past observations and actions as history. Also, they improve the performance by changing the fixed vision features to the Vision Transformer, ViT (Dosovitskiy et al., 2020). However, all of those transformer-based models entangle the learning of spatial information and visual information. Furthermore, prior works (Chen et al., 2021; Qiao et al., 2022) design the pre-training tasks without considering the adaption to the downstream model. Our work designs different modules to better utilize the learnt spatial and visual information from the pre-training.

3 Method

3.1 Problem Description

In this task, formally, the agent is given an instruction $\mathcal{W} = \{w_1, w_2, \cdots, w_L\}$, where w represents tokens and L is the number of tokens. At each time step t, the agent observes a panorama which consists of 36 discrete images [1], which are denoted as $\mathcal{I}^p = \{I_1^p, I_2^p, \cdots, I_{36}^p\}$. There are k navigable viewpoints in those panoramic views that the agent can navigate to. We denote the navigable viewpoints as $\mathcal{I}^c = \{I_1^c, I_2^c, \cdots, I_k^c\}$. In our model, to focus on the relevant observations in the visual environment, we only use the navigable viewpoints rather than all 36 images in the whole panoramic view.

The goal of the task is to select the next viewpoint among navigable viewpoints which forms a trajectory that takes the agent close to a goal destination. The agent terminates when the current viewpoint is selected or a pre-defined maximum number of navigation steps have been reached.

3.2 Background

Following (Hong et al., 2021), we design text encoder, vision features, and a recurrent state unit.

Text Encoder We first apply BERT tokenizer to split the text instruction to a sequence of tokens: $\{[CLS], w_1, w_2, \cdots, w_L, [SEP]\}$, where L is the number of tokens, and $[CLS]$ and $[SEP]$ are the

[1] 12 headings and 3 elevations with 30 degree intervals.

special tokens. Text embedding of each token is obtained by summing up the token embedding, position embedding and type embedding of text. Then the text embedding is passed through a text encoder, a standard multi-layer Transformer with self-attention, to obtain the contextual representation, represented as $X = \{x_1, x_2, \cdots, x_L\}$.

Vision Features For each navigable viewpoint, we consider its vision and relative orientation features. Specifically, we use ResNet-152 (He et al., 2016) pre-trained on ImageNet (Russakovsky et al., 2015) as 2048-d vision representation. We repeat relative heading α and elevation β features $[sin\alpha; cos\alpha; sin\beta, cos\beta]$ 32 times to constitute a 128-d orientation feature as O_p. Formally, we denote the vision and orientation representations for the the navigable viewpoints as $V = \{v_1, v_2, \cdots, v_k\}$ and $O = \{o_1, o_2, \cdots, o_k\}$, respectively.

Recurrent State Unit The recurrent state unit stores the history information from the previous steps. At each time step, the navigation agent takes three inputs: the state representation s_t, the language representation X, and the vision representation V_t. For the next navigation step, the state is refined using the text information and the current observations in the visual environment as follows,

$$s_{t+1} = NAV(s_t, X, V_t), \tag{1}$$

where NAV is the navigation model. In three modules from our model, state representation are initialized with the $[CLS]$ representation in the text encoder. In our model, we assign the state representation to three different modules to consider different information.

3.3 Our Model: LOViS

Our proposed model (LOViS) has three main modules: history module, orientation module, and vision module, as depicted in Figure 2.

History Module The History Module receives three types of inputs: state representation s_t (see "state" in Figure 2), text representation X, and "vision-orientation" representations. To obtain "vision-orientation" representation, we feed the concatenation of vision and orientation representations to a "vision-orientation encoder" (see Figure 2). We denote "vision-orientation" representation as $\tilde{VO} = \{\tilde{vo}_1, \tilde{vo}_2, \cdots, \tilde{vo}_k\}$. Then we use cross-modal attention layers and self-attention layers to obtain the cross representation. In cross-modality attention Transformer layers, one modality is used as a query and the other as the key to exchange information as follows,

$$\hat{X}, \hat{s}_t, \hat{VO}_t = Cross\_Attn(X, [s_t; VO_t]), \tag{2}$$

where $\hat{X}$, $\hat{s}_t$, and $\hat{VO}_t$ are respectively updated state, text and "vision-orientation" representations after cross modality attention layers. Then state and "vision-orientation" representations are fed into self-attention Transformer layers:

$$s_{t+1}, p_t^h = Self\_Attn([\hat{s}_t; \hat{VO}_t]) \tag{3}$$

where s_{t+1} is the updated state after self-attention layers. p_t^h is the self attention score between state representations and "vision-orientation" representations. Note that the refinement of the state representation only happens in the history module.

Orientation Module Orientation information is vital for the navigation task. For example, the instruction, "*turn left*" can assist the agent to ignore the navigable viewpoints on the right side. In our work, we build an orientation module specifically to encourage the agent to learn the spatial information from the instructions and ground it in the visual environment. Specifically, we linearly project the orientation features O (refer to the notations in Section 3.2) via the "Orientation Encoder" (see Figure 2) to obtain its projected representation, denoted as $\tilde{O}$. Then we input the state representation s_t, text representation X, and the projected orientation representation $\tilde{O}$ to the cross-modality attention Transformer layer. The orientation module learns a new state representation, denoted as s_t^o, for orientation information (see "State-O' in Figure 2). For cross-model attention layers, we have:

$$\hat{X}^o, \hat{s}_t^o, \hat{O}_t = Cross\_Attn(X, [s_t^o; \tilde{O}_t]) \tag{4}$$

where $\hat{X}^o, \hat{s}_t^o, \hat{O}_t$ are updated state, text, orientation representations after cross modality attention layers in the orientation module. Then we use the state representation enriched with the orientation information to perform self-attention with orientation representations as follows.

$$p_t^o = Self\_Attn([\hat{s}_t^o; \hat{O}_t]) \tag{5}$$

where p_t^o is the attention score between state representation and orientation feature.

Vision Module Connecting mentioned landmarks in the instruction to the scene and objects in the visual environment is also important to the navigation task. In the instruction, "*enter into the*

bedroom and move close to TV.", The mentioned landmarks, such as "*bedroom*" and "*TV*", provide apparent clues for the navigation actions. Like the orientation module, we build a vision module to ground the text landmarks in the visual scene and objects. Specifically, we first project vision representations V (refer to the notations in Section 3.2) using "Vision Encoder" (see Figure 2) to obtain the projected visual representation, denoted as $\tilde{V}$. Then we input the state representation s_t, text representation X, and projected vision representation $\tilde{V}$ to the cross-modal attention and self-attention layers as follows,

$$\hat{X}^v, \hat{s}_t^v, \hat{V}_t = Cross\_Attn(X, [s_t^v; \tilde{V}_t]), \quad (6)$$

$$p_t^v = Self\_Attn([\hat{s}_t^v; \hat{V}_t]), \quad (7)$$

where s_t^v is the new state representation considering visual information (see "State-V" in Figure 2). $\hat{X}^v, \hat{s}_t^v, \hat{V}_t$ are updated state, text, vision representations after cross modality attention layers in the vision module. p_t^v is the attention score between state representation and vision representations.

Action Selection For each navigable viewpoint, we obtain the self-attention scores from 1) orientation state representation to its orientation representation (orientation module), 2) vision state representation to the vision representation (vision module), 3) state representation to the combined orientation and visual representations (history module). We combine these scores as follows:

$$p_t = Softmax(W_a[p_t^h; p_t^o; p_t^v]) \quad (8)$$

where w_a is the trainable parameter, and p_t denote the action probability which weights different module scores.

4 Training and Inference

We train our model with the mixture of Imitation Learning (IL) and Reinforcement Learning (RL) following (Tan et al., 2019). It minimizes the cross-entropy loss of the prediction and ground-truth trajectories by following teacher actions for each navigation step. RL samples an action from the action probability p_t^a and learns from the rewards. The loss function is the following:

$$l = -\sum_t a_t^s log(p_t^a) - \lambda \sum_t a_t^* log(p_t^a) \quad (9)$$

where a_t^* is the teacher action, a_t^s is the sampled action from RL, and λ is the coefficient to balance two training goals. During inference, we use the greedy search to select the viewpoint with highest probability and finally generate the trajectory.

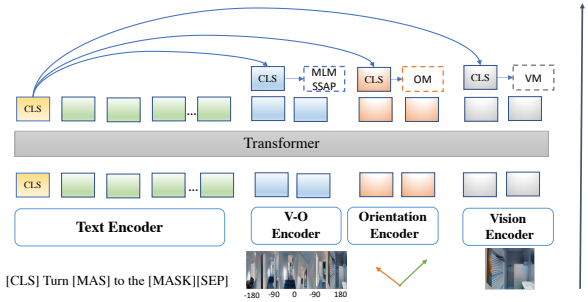

Figure 3: Pre-training Model with Specific Pre-training Tasks. V-O is the encoder considering both orientation and vision representations. MLM: Masked Language Modeling; SSAP: Singe Step Action Prediction; OM: Orientation Matching; VM:Vision Matching.

5 Pre-training

We follow the model architecture of PREVALENT (Hao et al., 2020) to obtain the joint cross representations trained on text-image-action triplets, as shown in Figure 3. However, the novelty of our pre-training is that we design new tasks named Vision Matching (VM) and Orientation Matching (OM) to pretrain for the vision module and orientation module designed in our navigation agent, as shown in Figure 2. Moreover, we improve the existing pre-training tasks of the PREVALENT, Masked Language Modeling (MLM) and Single Step Action Prediction (SSAP), to obtain a more effective initialization of our new architecture. Here, we describe the details of all the pre-training tasks while their effects is described in Section 6.5.1. In the following tasks, we denote each instruction-trajectory pair in training set D as $< w, \tau >$.

Masked Language Modeling (MLM) Different from PREVALENT (Hao et al., 2020) masking of random tokens, we mask direction and landmark tokens with 8% probability and replace them with special token $[MASK]$. The goal is to recover landmark or orientation tokens w_m by reasoning over the surrounding words $w_{\setminus m}$, and the orientation and visual observation at the each navigation step. We denote the combination of orientation and vision features of panorama views as VO_p. Landmark tokens are usually the token related to scene or objects in the visual environment, such as "*table*", "*sofa*", and "*bedroom*". We extract nouns as landmark tokens based on their pos-tag. The direction tokens usually convey spatial information, such as "*left*", "*right*", and "*forward*". We obtain direction tokens using a direction dictionary built upon R2R training dataset. The loss of MLM is

calculated as follows,

$$\mathcal{L}_{MLM} = -\mathbb{E}_{VO_p \sim P(\tau),(w,\tau) \sim D} \log P(w_m | w_{\setminus m}, VO_p), \tag{10}$$

Single Step Action Prediction (SSAP) PREVALENT (Hao et al., 2020) selects actions by mapping the $[CLS]$ representations to the 36 classes directly, which may cause the loose connection between cross-modal representations of the viewpoints and the action space. To address this issue, we use the cross attention distribution from the $[CLS]$ representation to the images in the panoramic view to select an action. We use the cross-entropy loss to compute the loss of SSAP, as follows,

$$\mathcal{L}_{SSAP} = -\mathbb{E}_{OV_p \sim P(\tau),(w,\tau) \sim D} \log P(a | w_{[CLS]}, VO_p), \tag{11}$$

where a is the ground-truth action.

Vision Matching (VM) VM is our novel pre-training specific for initializing our vision module. It predicts whether the current vision information can match with the instruction. In this task, to encourage the agent to focus on learning the connection between landmarks in the instruction and the scene objects in the visual environment, we only use the vision representation (i.e. excluding the heading and elevation) of viewpoint as the input, denoted as v_p. We generate the negative samples by replacing the ground-truth images with an images from another environment. We use the output representation of the $[CLS]$ as the joint representation of textual and visual features to feed to a fully connected layer with a sigmoid function. This layer predicts the matching score $s(w, v_p)$. The loss of SSAP is computed as follows,

$$\mathcal{L}_{VM} = -\mathbb{E}_{v_p \sim \tau,(w,\tau) \sim D}[y \log P + (1-y) \log P)], \tag{12}$$

where $P = s(w, v_p)$, and $y \in \{0, 1\}$ indicates whether the sampled viewpoint-instruction pair is matching.

Orientation Matching (OM) Our second novel pre-training task is designed to learn the orientation representations. We propose to predict the current orientation based on the instruction and the initial orientation. As described before, the orientation feature O_p is the combination of the heading α and elevation β. We use the output representation of $[CLS]$ as the joint representation of instruction and orientation. Then we feed this to a fully connected layer to predict 4-bits of orientation features. The loss of OM is computed as follows,

$$\mathcal{L}_{OM} = -\mathbb{E}_{o_p \sim \tau,(w,\tau) \sim D} \log p(O' | w_{[CLS]}, O_p), \tag{13}$$

where O' is the ground-truth orientation feature. The full pre-training objective is

$$\mathcal{L}_{pre-train} = \mathcal{L}_{MLM} + \mathcal{L}_{SSAP} + \mathcal{L}_{VM} + \mathcal{L}_{OM}. \tag{14}$$

6 Experiments

6.1 Dataset

Two VLN datasets are used in evaluation: R2R (Anderson et al., 2018) and R4R (Jain et al., 2019).

R2R builds upon the Matterport3D dataset. This dataset has 7198 paths and 21567 instructions with an average length of 29 words. The whole dataset is partitioned into training, seen validation, unseen validation, and unseen test set. The seen set shares the same visual environments with the training set, while unseen sets contain different environments.

R4R extends the R2R dataset with longer instructions and trajectories by concatenating two adjacent tail-to-head trajectories in R2R. Different from R2R, the trajectories in R4R are less biased as they are not necessarily the shortest path from the start viewpoint to the destination.

6.2 Evaluation Metrics

We mainly report three evaluation metrics for R2R. (1) Navigation Error (NE): the mean of the shortest path distance between the agent's final position and the goal location. (2) Success Rate (SR): the percentage of the cases where the predicted final position is close within 3 meters from the goal location. (3) Success rate weighted by normalized inverse Path Length (SPL) (Anderson et al., 2018): normalizes Success Rate by trajectory length. It considers both the effectiveness and efficiency of navigation performance.

In terms of the metrics for the R4R benchmark, besides the basic metrics same as R2R, NE, SR, and SPL, R4R includes additional metrics: the Coverage Weighted by Length Score (CLS), the Normalized Dynamic Time Warping and the nDTW weighted by Success Rate (sDTW). In R4R, SR and SPL measure the performance of the navigation, while CLS, nDTW and sDTW measure the fidelity of the predicted paths.

6.3 Implementation Details

Please check our code [2] for the implementation.

Pre-training We use 4 GeForce RTX 2080 GPUs for pre-training. The batch size for each GPU is 28, and the training time is around 22 hours. The

[2] https://github.com/HLR/LOViS

	Method	Val seen			Val Unseen			Test(Unseen)		
		NE ↓	SR ↑	SPL↑	NE ↓	SR ↑	SPL↑	NE ↓	SR ↑	SPL ↑
1	Speaker-Follower (Fried et al., 2018)	3.36	0.66	-	6.62	0.35	-	6.62	0.35	0.28
2	Env-Drop (Tan et al., 2019)	3.99	0.62	0.59	5.22	0.47	0.43	5.23	0.51	0.47
3	OAAM (Qi et al., 2020)	-	0.65	0.62	-	0.54	0.50	5.30	0.53	0.50
4	RelGraph (Hong et al., 2020a)	3.47	0.67	0.65	4.73	0.57	0.53	4.75	0.55	0.52
5	NvEM (An et al., 2021)	3.44	0.69	0.65	4.27	0.60	0.55	4.37	0.58	0.54
6	PRESS (Li et al., 2019)	4.39	0.58	0.55	5.28	0.49	0.45	5.49	0.49	0.45
7	PREVALENT (Hao et al., 2020)	3.67	0.69	0.65	4.71	0.58	0.53	5.30	0.54	0.51
8	AirBERT (Guhur et al., 2021)	2.68	0.75	0.70	4.01	0.62	0.56	4.13	0.62	0.57
9	RecBERT (Hong et al., 2021)	2.90	0.72	0.68	3.93	0.63	0.57	4.09	**0.63**	0.57
10	HAMT (Chen et al., 2021)	-	0.69	0.65	-	0.64	0.58	-	-	-
11	RecBERT*	2.99	0.71	0.66	4.03	0.61	0.56	4.35	0.61	0.57
12	Our pretrain + RecBERT	2.90	0.74	0.69	3.75	0.63	0.58	4.20	**0.63**	0.57
13	Our pretrain + LOViS (our model)	**2.40**	**0.77**	**0.72**	**3.71**	**0.65**	**0.59**	**4.07**	**0.63**	**0.58**

Table 1: Experimental Results Comparing with Baseline Models on R2R Benchmarks in a single-run setting. The best results are in bold font. * denotes our reproduced R2R results.

	Val Seen						Val Unseen					
Method	NE↑	SR↑	SPL↑	CLS↑	nDTW↑	sDTW↑	NE↓	SR↑	SPL↑	CLS↑	nDTW↑	sDTW↑
EnvDrop* (Tan et al., 2019)	-	0.52	0.41	0.53	-	0.27	-	0.29	0.18	0.34	-	0.09
OAAM (Qi et al., 2020)	-	0.56	0.49	0.54	-	0.32	-	0.29	0.18	0.34	-	0.11
NvEM (An et al., 2021)	5.38	0.54	0.47	0.51	0.48	0.35	6.80	0.38	0.28	0.41	0.36	0.20
RecBERT* (Hong et al., 2021)	4.82	0.56	0.46	0.50	0.56	0.38	6.48	0.43	0.32	0.41	0.42	0.21
LOViS (our model)	**4.16**	**0.67**	**0.58**	**0.56**	**0.58**	**0.43**	**6.07**	**0.45**	**0.35**	**0.45**	**0.43**	**0.23**

Table 2: Experimental Results for comparing LOViS with the Baseline Models on R4R dataset in a single-run setting. The best results are in bold font. * denotes our reproduced R4R results.

learning rate is $5e-5$, and the AdamW optimizer is adopted. The language Transformer has nine layers, and the cross-modality Transformer has four layers. The models' parameters are initialized with the weights of PREVALENT (Hao et al., 2020).

Fine-tuning We directly adapt different encoders and Transformer layers from our pre-training model to our navigation model. The navigation model is further trained in 30k iterations with learning rate $1e-5$. The batch size is 28. The best performance is selected according to the best SPL of the validation unseen set. For R2R, we use the same augmented data as in (Hong et al., 2021) for a fair comparison.

6.4 Comparisons with SoTA

Table 1 shows the experimental results of different VLN methods on R2R benchmarks in a single-run setting. In this table, row#1 to row#5 show the results of the LSTM-based navigation agents. From row#6 to row#10 are Transformer-based navigation agents that largely have improved the performance of the LSTM-based agents. PREVALENT (Hao et al., 2020) pre-trains the cross-modal representations with text-image-action triplets and replaces the encoder of Env-Drop (Tan et al., 2019) to improve its performance. AirBERT (Guhur et al., 2021) is one of the SOTA methods that train a model on a large scale and diverse in-domain detests. RecBERT(Hong et al., 2021) , our baseline, is also a SOTA method that uses the attention distribution of the history information on navigation candidates to determine the next action. Row#10 is their own reported results in their paper, and row#11 shows our best reproduced results which is consistent with the reported results in (Liu et al., 2021). Row#12 and row#13 are the performance of our LOViS model. We first show the effectiveness of our pre-training on the baseline model. Our pre-training setting can improve the SR and SPL of baseline by about 2% in the unseen validation environment. Moreover, we further improve the performance of the baseline with our designed navigation model and the pre-training setting. The improvement is about 3% of SR and SPL in the seen environment and 2% of SR in the unseen validation and test environment. This result indicates our pre-training tasks are more suitable for our designed navigation model. We also obtain a lower NE showing that our agent navigates closer to the destination. For HAMT (Chen et al., 2021), we report their results with ResNet-152 as the vision encoder for a fair comparison. Among those methods, only OAAM (Qi et al., 2020) and NvEM (An et al., 2021) consider the semantics of spatial information and visual perception, but their results have a large performance gap compared to our Transformer-based navigation model.

Table 2 shows the performance of various models on R4R benchmark in a single-run setting.

		Baseline Model				LOViS (Our Model)			
		Val Seen		Val Unseen		Val Seen		Val Unseen	
	Tasks	**SR↑**	**SPL↑**	**SR↑**	**SPL↑**	**SR↑**	**SPL↑**	**SR↑**	**SPL↑**
1	MLM	0.712	0.662	0.613	0.562	0.724	0.673	0.621	0.564
2	MLM+SSAP	0.731	0.675	0.619	0.575	0.747	0.695	0.649	0.585
3	MLM+SSAP+VM	0.737	0.683	0.622	0.577	0.755	0.711	0.637	0.581
4	MLM+SSAP+OM	0.730	0.672	0.617	0.574	0.766	0.724	0.629	0.579
5	MLM+SSAP+VM+OM	**0.743**	**0.691**	**0.632**	**0.583**	**0.774**	**0.722**	**0.653**	**0.592**

Table 3: Ablation Study for Different Tasks of Pre-training on the Baseline and LOViS.

		Val Seen		Val Unseen	
	Modules	**SR↑**	**SPL↑**	**SR↑**	**SPL↑**
1	H	0.743	0.691	0.632	0.583
2	H+O	0.756	0.712	0.629	0.576
3	H+V	0.762	0.718	0.642	0.588
4	H+O+V	**0.774**	**0.722**	**0.653**	**0.592**

Table 4: Ablation Study for Different Modules in Model. H: History Module; O: Orientation Module; V: Vision Module.

Same as R2R, we can better perform in all evaluation metrics. Compared to the our reproduced results of the RecBERT (Hong et al., 2020a), we can improve 4% of CLS, 1% of nDTW, and 2% of sDTW in the unseen validation environment, which indicates the significantly better fidelity of our model.

6.5 Ablation Study

6.5.1 Ablation Study of Different Tasks

In Table 1, we already observe that our pre-training strategy improved the RecBERT baseline. In Table 3, we show the influence of each pre-training task on both RecBERT and LOViS. For RecBERT baseline model, SSAP shows about 2% of improvement on both seen and unseen environments. Although the tasks of VM and OM independently do not change the performance of MLM+SSAP, the combination of two tasks improves the performance by about 1%. The same phenomenon happens in LOViS. SSAP improves the performance by a large margin. Although VM and OM do not show significant improvement when used separately in the unseen environment, they improve both SR and SPL in the seen environment. The combination of VM and OM improves the performance significantly, especially in the seen environment.

6.5.2 Ablation Study of Different Modules

Table 4 shows the ablation of different modules. Based on row#1, the history module with our pre-training strategy has already improved its performance. Comparing row#2 and row#3, we can see that vision module affects the results more than the Orientation module. The combination of two modules with our pre-training strategy achieves the best performance (row#4). This indicates that our designed explicit modules can assist the agent in choosing the correct action based on different information.

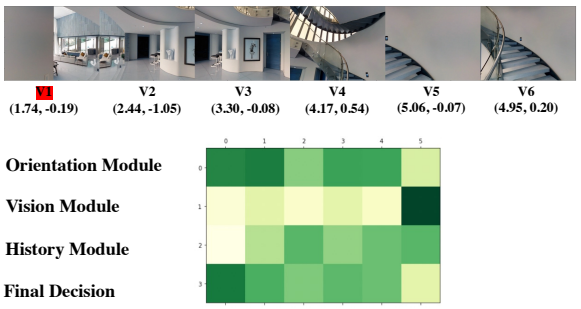

Figure 4: **Qualitative Example.** The ground-truth viewpoint is v1. The word "*down*" and "*left*" are the orientation signals. The word "*stairs*" is the vision signal. The attention map shows the score of different candidate viewpoints in each module. The darker color means the higher score. The numbers below each viewpoint show the orientation information with the format of <relative heading, relative elevation>. The lower value of each number means the orientation is more towards left and down respectively.

6.6 Qualitative Example

Figure 4 shows a qualitative example that demonstrates the performance of each module of LOViS navigation agent. It is evident that the *orientation module* gives a higher score to the viewpoints that are left, and their elevation is down. The *vision module* gives a higher score to the viewpoints that "stairs" can be seen. The history module also gives a relatively higher score to the viewpoints on the right side. The final decision is $v1$ with its weights of $[0.02, -0.03, -0.04]$ to the three modules. The example shows that our designed orientation and vision modules can attend to the viewpoint with the corresponding information.

7 Conclusion

The main idea of this paper is to design explicit vision and orientation modules in the neural architecture of a navigating agent. These modules can effectively learn to ground the landmark mentions

and spatial information related to the orientation of the agent expressed in the natural language instruction into the visual environment. To make the designed modules more effective, we design new pre-training tasks accordingly to equip the agent with spatial reasoning and visual perception abilities before navigation. We evaluate our model on R2R and R4R datasets and achieve state-of-the-art results. Our ablation study shows the effectiveness of our designed modules and pre-training tasks.

8 Acknowledgement

This project is supported by National Science Foundation (NSF) CAREER award 2028626 and partially supported by the Office of Naval Research (ONR) grant N00014-20-1-2005. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of the National Science Foundation nor the Office of Naval Research. We thank all reviewers for their thoughtful comments and suggestions.

References

Dong An, Yuankai Qi, Yan Huang, Qi Wu, Liang Wang, and Tieniu Tan. 2021. Neighbor-view enhanced model for vision and language navigation. In *Proceedings of the 29th ACM International Conference on Multimedia*, pages 5101–5109.

Peter Anderson, Qi Wu, Damien Teney, Jake Bruce, Mark Johnson, Niko Sünderhauf, Ian Reid, Stephen Gould, and Anton Van Den Hengel. 2018. Vision-and-language navigation: Interpreting visually-grounded navigation instructions in real environments. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 3674–3683.

Shizhe Chen, Pierre-Louis Guhur, Cordelia Schmid, and Ivan Laptev. 2021. History aware multimodal transformer for vision-and-language navigation. *Advances in Neural Information Processing Systems*, 34.

Yen-Chun Chen, Linjie Li, Licheng Yu, Ahmed El Kholy, Faisal Ahmed, Zhe Gan, Yu Cheng, and Jingjing Liu. 2019. Uniter: Learning universal image-text representations.

Soham Dan, Parisa Kordjamshidi, Julia Bonn, Archna Bhatia, Jon Cai, Martha Palmer, and Dan Roth. 2020. From spatial relations to spatial configurations. *arXiv preprint arXiv:2007.09557*.

Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, et al. 2020. An image is worth 16x16 words: Transformers for image recognition at scale. *arXiv preprint arXiv:2010.11929*.

Daniel Fried, Ronghang Hu, Volkan Cirik, Anna Rohrbach, Jacob Andreas, Louis-Philippe Morency, Taylor Berg-Kirkpatrick, Kate Saenko, Dan Klein, and Trevor Darrell. 2018. Speaker-follower models for vision-and-language navigation. *Advances in Neural Information Processing Systems*, 31.

Pierre-Louis Guhur, Makarand Tapaswi, Shizhe Chen, Ivan Laptev, and Cordelia Schmid. 2021. Airbert: In-domain pretraining for vision-and-language navigation. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 1634–1643.

Weituo Hao, Chunyuan Li, Xiujun Li, Lawrence Carin, and Jianfeng Gao. 2020. Towards learning a generic agent for vision-and-language navigation via pre-training. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 13137–13146.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. 2016. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778.

Yicong Hong, Cristian Rodriguez, Yuankai Qi, Qi Wu, and Stephen Gould. 2020a. Language and visual entity relationship graph for agent navigation. *Advances in Neural Information Processing Systems*, 33:7685–7696.

Yicong Hong, Cristian Rodriguez-Opazo, Qi Wu, and Stephen Gould. 2020b. Sub-instruction aware vision-and-language navigation. *arXiv preprint arXiv:2004.02707*.

Yicong Hong, Qi Wu, Yuankai Qi, Cristian Rodriguez-Opazo, and Stephen Gould. 2021. Vln bert: A recurrent vision-and-language bert for navigation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 1643–1653.

Vihan Jain, Gabriel Magalhaes, Alexander Ku, Ashish Vaswani, Eugene Ie, and Jason Baldridge. 2019. Stay on the path: Instruction fidelity in vision-and-language navigation. *arXiv preprint arXiv:1905.12255*.

Parisa Kordjamshidi, Marie-Francine Moens, and Martijn van Otterlo. 2010. Spatial role labeling: Task definition and annotation scheme. In *Proceedings of the Seventh conference on International Language Resources and Evaluation (LREC'10)*, pages 413–420. European Language Resources Association (ELRA).

Jacob Krantz, Erik Wijmans, Arjun Majumdar, Dhruv Batra, and Stefan Lee. 2020. Beyond the nav-graph: Vision-and-language navigation in continuous environments. In *European Conference on Computer Vision*, pages 104–120. Springer.

Jialu Li, Hao Tan, and Mohit Bansal. 2021. Improving cross-modal alignment in vision language navigation via syntactic information. *arXiv preprint arXiv:2104.09580*.

Xiujun Li, Chunyuan Li, Qiaolin Xia, Yonatan Bisk, Asli Celikyilmaz, Jianfeng Gao, Noah Smith, and Yejin Choi. 2019. Robust navigation with language pretraining and stochastic sampling. *arXiv preprint arXiv:1909.02244*.

Xiujun Li, Xi Yin, Chunyuan Li, Pengchuan Zhang, Xiaowei Hu, Lei Zhang, Lijuan Wang, Houdong Hu, Li Dong, Furu Wei, et al. 2020. Oscar: Object-semantics aligned pre-training for vision-language tasks. In *European Conference on Computer Vision*, pages 121–137. Springer.

Chong Liu, Fengda Zhu, Xiaojun Chang, Xiaodan Liang, Zongyuan Ge, and Yi-Dong Shen. 2021. Vision-language navigation with random environmental mixup. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 1644–1654.

Jiasen Lu, Dhruv Batra, Devi Parikh, and Stefan Lee. 2019. Vilbert: Pretraining task-agnostic visiolinguistic representations for vision-and-language tasks. *Advances in neural information processing systems*, 32.

Chih-Yao Ma, Jiasen Lu, Zuxuan Wu, Ghassan AlRegib, Zsolt Kira, Richard Socher, and Caiming Xiong. 2019. Self-monitoring navigation agent via auxiliary progress estimation. *arXiv preprint arXiv:1901.03035*.

Yuankai Qi, Zizheng Pan, Shengping Zhang, Anton van den Hengel, and Qi Wu. 2020. Object-and-action aware model for visual language navigation. In *European Conference on Computer Vision*, pages 303–317. Springer.

Yanyuan Qiao, Yuankai Qi, Yicong Hong, Zheng Yu, Peng Wang, and Qi Wu. 2022. Hop: History-and-order aware pre-training for vision-and-language navigation. *arXiv preprint arXiv:2203.11591*.

Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael Bernstein, et al. 2015. Imagenet large scale visual recognition challenge. *International journal of computer vision*, 115(3):211–252.

Hao Tan and Mohit Bansal. 2019. Lxmert: Learning cross-modality encoder representations from transformers. *arXiv preprint arXiv:1908.07490*.

Hao Tan, Licheng Yu, and Mohit Bansal. 2019. Learning to navigate unseen environments: Back translation with environmental dropout. *arXiv preprint arXiv:1904.04195*.

Yue Zhang, Quan Guo, and Parisa Kordjamshidi. 2021. Towards navigation by reasoning over spatial configurations. *arXiv preprint arXiv:2105.06839*.

Yue Zhang and Parisa Kordjamshidi. 2022. Explicit object relation alignment for vision and language navigation. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics: Student Research Workshop*, pages 322–331.

GAP: A Graph-aware Language Model Framework for Knowledge Graph-to-Text Generation

Anthony Colas* and **Mehrdad Alvandipour*** and **Daisy Zhe Wang**
Department of Computer Science, University of Florida
{acolas1, m.alvandipour, daisyw}@ufl.edu

Abstract

Recent improvements in KG-to-text generation are due to additional auxiliary pre-training tasks designed to give the fine-tune task a boost in performance. These tasks require extensive computational resources while only suggesting marginal improvements. Here, we demonstrate that by fusing graph-aware elements into existing pre-trained language models, we are able to outperform state-of-the-art models and close the gap imposed by additional pre-training tasks. We do so by proposing a mask structure to capture neighborhood information and a novel type encoder that adds a bias to the graph-attention weights depending on the connection type. Experiments on two KG-to-text benchmark datasets show our models are competitive while involving fewer parameters and no additional pre-training tasks. By formulating the problem as a framework, we can interchange the various proposed components and begin interpreting KG-to-text generative models based on the topological and type information found in a graph.

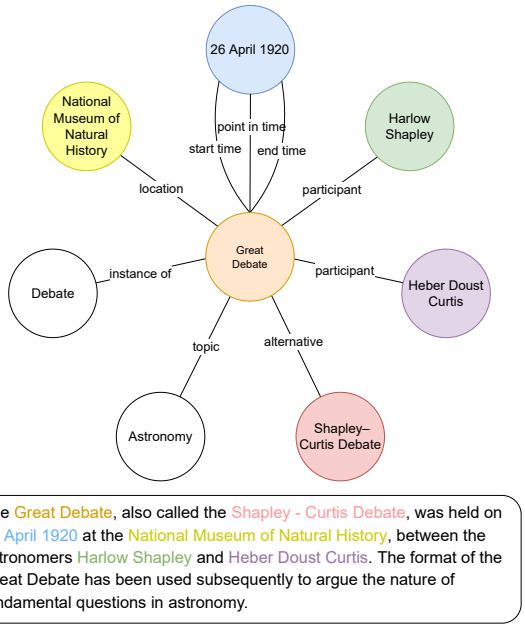

Figure 1: Given a graph, KG-to-text generation aims to describe the entities, relations, and its inherent structure via natural language text (grey callout). Corresponding graph-text components are color-coded.

1 Introduction

Due to the amount of data stored in Knowledge Graphs (KGs) (Auer et al., 2007; Vrandečić and Krötzsch, 2014; Bollacker et al., 2008; Yates et al., 2007; Bodenreider, 2004; Wishart et al., 2018), they are important to properly transcribe into natural language sentences, making them more easily comprehensible to a larger audience. This task, termed KG-to-text, has found recent success in generating knowledge-grounded dialog responses (Wen et al., 2016; Zhou et al., 2018), question answering (He et al., 2017; Bhowmik and de Melo, 2018; Pal et al., 2019; Agarwal et al., 2021), story generation (Guan et al., 2019; Ji et al., 2020), and event narration (Colas et al., 2021). KG-to-text involves encoding a KG, often sparse, in order to generate a coherent and representative textual description of the KG as shown in Figure 1. In contrast, Abstract Meaning Representation (AMR)-to-text deals with a more restrictive space, where graphs follow a predefined dense, connected template (Ribeiro et al., 2021; Koncel-Kedziorski et al., 2019). Thus, when encoding a KG, one should carefully consider the graph's structure to properly generate its corresponding text.

Recently, pre-trained language models (LMs) have produced state-of-the-art results on the KG-to-text generation task (Ribeiro et al., 2020a; Chen et al., 2020). These models tend to first linearize a graph into a sequence of tokens, and fine-tune on pre-trained LMs such as BART (Lewis et al., 2020), GPT (Radford et al., 2019), or T5 (Raffel et al., 2020), treating the task similarly to a text-to-text task. Because of the performance gains caused

*These authors contributed equally.

Proceedings of the 29th International Conference on Computational Linguistics, pages 5755–5769
October 12–17, 2022.

by the self-supervised pre-training tasks, current work on KG-to-text has focused on developing pre-trained tasks and large-scale unlabeled graph-text corpora, replicating the success in the text-to-text domain (Chen et al., 2020; Ke et al., 2021). However, these works particularly focus on leveraging large amounts of pre-trained data for graph-to-text specific pre-trained tasks, e.g., recovering a masked text sequence based on a given complete KG.

Although recent work in KG-to-text has begun to combine LMs with a graph-aware approach (Ke et al., 2021), they do not adequately perform a graph-aware encoding, overlooking the KG's topological information. Similarly, recent work in AMR-to-text has begun to observe the role of graph adaptors in dense, highly parallel data, using a Graph Convolutional Network (GCN) (Ribeiro et al., 2021). Instead, our framework leverages a topological attention mechanism, better adhering to the language model paradigm and giving room for interpretation.

We argue and show empirically that **without additional pre-trained tasks**, a fully graph-aware encoding combined with the coverage of pre-trained LMs such as BART (Lewis et al., 2020), can compete with and in some cases outperform those approaches which rely on additional pre-training. By doing so, we unload the burden of requiring vast amounts of data and computational resources required for pre-training.

We propose *GAP*, a KG-to-text framework which fuses **g**raph-**a**ware elements into existing **p**re-trained LMs, capturing the advantages brought forth by both model types. Our framework has two main components: (i) **Global Attention:** A graph's components are first encoded using an LM to capture their global semantic information, allowing the model to utilize the lexical coverage of pre-trained LMs (Davison et al., 2019; Gururangan et al., 2020; Vulić et al., 2020). (ii) **Graph-aware Attention**: Next, we devise a topological-aware graph attention mechanism, with entity/relation type encoding. Our framework attends to and updates entity, relation, or both representations. By proposing such a framework, where graph-aware components can be interchanged, we can begin exploring explainable generative models for the KG-to-text task.

We evaluate GAP on two publicly available KG-to-text datasets: WebNLG v2.0 (Shimorina and Gardent, 2018) and EventNarrative (Colas et al., 2021), achieving state-of-the-art results on various natural language generation (NLG) metrics and demonstrate the value of our fully graph-aware based approach. Our contributions are as follows:

1. We propose a novel graph-aware framework for KG-to-text by introducing neighborhood-masked attention and connection type encoding into pre-trained LMs, capturing both local structural and global contextual information.

2. We provide more interpretable insights on KG-to-text generative models by drawing upon our framework and interchanging the various masking and type schemes, evaluating the output based on the variable graph topology.

3. We demonstrate on two datasets that by simply finetuning our models, which infuse graph-aware elements into existing LMs, one can even marginally outperform current state-of-the-art models which rely on several computationally expensive pre-training tasks.

We make our code publically available to motivate future research.

2 Related Work

2.1 KG-to-Text with Graph Transformers

Graph Neural Networks (GNNs) (Veličković et al., 2018) have shown to be effective at encoding graph data. For the KG-to-text task, recent works have leveraged GNNs to encode a graph's neighborhood information (Koncel-Kedziorski et al., 2019; Marcheggiani and Perez-Beltrachini, 2018; Ribeiro et al., 2020b; Schmitt et al., 2021; Guo et al., 2019; Jin et al., 2020) before decoding its corresponding textual representation. Other work instead choose a more global approach and base their encoder on a Transformer-based architecture (Vaswani et al., 2017), calculating self-attention from all the nodes in a graph (Zhu et al., 2019; Cai and Lam, 2020; Ke et al., 2021). Like previous work, we encode neighborhood information in the Graph-aware Attention module. Recently, graph convolution-based adaptors have been explored for Abstract Meaning Representation-to-text (Ribeiro et al., 2021). Unlike previous work, GAP is a framework for KG-to-text, where the KG's topology and masking scheme are not set. While there has been work examining the effect of encoding a node's relative position (Shaw et al., 2018; Schmitt et al., 2021), we instead encode *type*, arguing that a KG's textual description is weighted based on its different types

of connections, and empirically show its effect on KG-to-text generation.

2.2 KG-to-Text with Pre-trained LM

With the advent of pre-trained LMs such as BART (Lewis et al., 2020), T5 (Raffel et al., 2020), and GPT (Radford et al., 2019), these models have been directly adapted and fine-tuned for the KG-to-text task and in some cases outperformed GNN-based models (Ribeiro et al., 2020a; Kale and Rastogi, 2020; Chen et al., 2020; Mager et al., 2020). While work has begun to explore combining such pre-trained models with transformer-based architectures which encode node information (Ke et al., 2021), they assume connectivity between all nodes and do not leverage updating relation information. Instead, here we propose a framework which combines pre-trained models with graph-aware encoders which are specifically neighborhood-based and dependent on a given graph's topology.

3 Problem Statement

We aim to generate texts that describe a given KG. We define a KG to be a multi-relational graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$, where $\mathcal{V}$ is the set of entity vertices and $\mathcal{E} \subset \mathcal{V} \times \mathcal{R} \times \mathcal{V}$ is the set of edges that connect entities with a relation from $\mathcal{R}$.

4 Proposed Framework

As our model is built on top of LMs such as BART, we first linearize the knowledge graph into a text string (Distiawan et al., 2018; Moryossef et al., 2019; Su et al., 2021). The linearization is a sequence of all triples in the KG, interleaved with tokens that separate each triple and the triple's components (head, relation, and tail). Figure 2 shows an example linearization for a small knowledge graph, along with its labeled components.

4.1 Global Attention

We then use a transformer encoder to contextualize the vector representations. The first module in each transformer layer is a self-attention over the linearized graph, which acts as a *Global Attention* and captures the semantic relationships between all tokens. The Global Attention can be initialized with a pre-trained LM. At the l-th layer, the self-attention is formulated as:

$$X_l = \text{Attn}(Q, K, V) = \text{softmax}\left(\frac{QK^\top}{\sqrt{d_k}}\right)V \quad (1)$$

Query, key, and value are computed via $Q = X_{l-1}W_l^Q, K = X_{l-1}W_l^K$, and $V = X_{l-1}W_{l-1}^V$. $X_{l-1} \in \mathbb{R}^{n\times d}$ denotes the collection of vectors corresponding to the graph's tokens. The model's parameters are denoted by W with size $d_k \times d_k$, where d_k is the dimension of word vectors.

4.2 Graph Aware Attention

While the *Global Attention* assumes connectivity between all graph components, KG adjacencies are sparse in nature. To capture this, we propose a *Graph-aware Attention* module, by first retrieving entity/relation vectors from the word vectors. Some entities or relations contain several words or repeat several times in the linearized graph. To get a single vector for each entity/relation, we add a pooling layer, which takes the average of the corresponding word vectors for each entity/relation. Hence, we get the graph representation matrix $X_l^g \in \mathbb{R}^{m\times d}$:

$$X_l^g = \text{pooling}(X_l) \quad (2)$$

Note, $m < n$, where m and n denote the number of graph components and number of tokens, respectively. In practice and for parallelization m will be a fixed number larger than this sum for all graphs in the dataset, and the graph representation can be accessed via masking. We propose a novel graph-aware attention on the graph representation X_l^g by introducing a neighborhood-based masking scheme and novel type encoder:

$$\tilde{X}_l^g = \text{Attn}_{M,T}(Q, K, V) = \text{softmax}\left(\frac{QK^\top}{\sqrt{d_k}} + M + \gamma(T)\right)V. \quad (3)$$

Here Q, K, V are constructed from X_l^g by multiplying it with their corresponding learnable parameter W. While $M \in \mathbb{R}^{m\times m}$ is a mask that encodes the desired graph structure, and $\gamma(T) \in \mathbb{R}^{m\times m}$ is the type encoding matrix. Note, each row of Q, K, and V correspond to an element from the graph (an entity or a relation), and before applying a softmax in each row of $QK^\top$, we can mask/modify the scores based on the graph topology. For instance, $M_{ij} = -\infty$ forces the item i to not attend to item j or the value at $\gamma(T)_{ij}$ can add a bias to the attention score based on the type of connection between items i and j. We exploit this capacity to inject graph-awareness by adding a masking matrix M and type encoding matrix $\gamma(T)$.

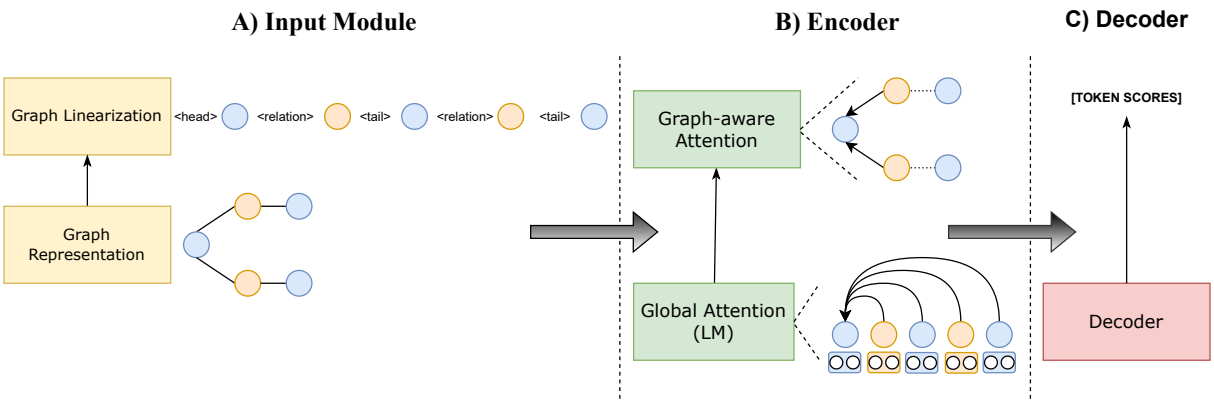

Figure 2: Overview of the Graph-aware framework for graph-to-text generation. Given a KG, we first transform the graph into its appropriate representation before linearizing the graph. Next, each node of the KG is encoded via a global attention, followed by a graph-aware attention, ultimately being decoded into a sequence of tokens.

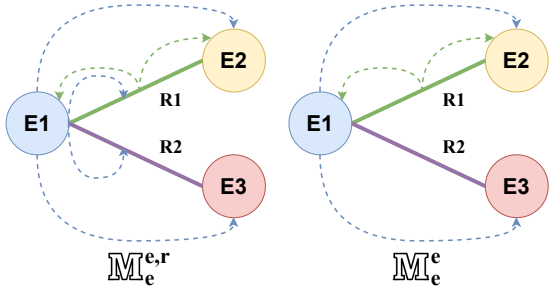

Figure 3: Two masking approaches. **Left:** $M_e^{e,r}$ mask, where E1 attends to its neighboring entities and relations, while R1 only attends to its neighboring entities. **Right:** M_e^e mask, where E1 and R1 only attend to their neighboring entities.

4.2.1 Graph Topology Encoding

The proposed matrix $M \in \mathbb{R}^{m \times m}$ encodes the graph topology by assigning $-\infty$ where attention is blocked and 0 otherwise. M can be thought of as a generalized adjacency matrix for graph $\mathcal{G}$, which has both nodes and edges as its rows and columns. Hence, to encode neighborhood information for an entity, we can modify its corresponding row in M to have the value 0 for its neighbors, and $-\infty$ otherwise. As the rows and columns of M contain relations, we also have the capacity to let relations attend to their neighboring entity or relations.

From a graph topology perspective, we have several design choices for the matrix M. We can let entities attend to neighboring entities, neighboring relations, or both. We also have these same options for when relations are playing the query role; that is, when choosing which components should relations attend to. For ease of reference and discussion, superscript denotes neighborhood types for entities, while subscript for relations, e.g. $M_{e,r}^{e,r}$. For instance, when entities attend to neighboring entities and relations, but relations only attend to entities, we denote the masking matrix by $M_e^{e,r}$. Figure 3 illustrates two such matrices via a graph and its attending components.

4.2.2 Connection Type Encoding

In contrast to M which encodes the general graph topology, we also introduce a new type encoding $T \in \mathbb{R}^{m \times m}$, designed for biasing the attention values between the different graph components based on their connection type. For instance, when an entity e is attending to its neighbor entities $\{e_i\}$ and relations $\{r_i\}$, we encode the two connection types and bias their attention scores. Type information is stored in a matrix T, and we then use an embedding lookup $\gamma : \mathbb{Z} \rightarrow \mathbb{R}$ to learn scalar embeddings for the types in T.

We define type T_{ij} between query i and key j based on two factors: (i) whether the two items are connected and (ii) the type of each item, i.e. whether the connection is entity–entity, entity–relation, relation–entity, or relation–relation:

$$T_{ij} = \begin{cases} 0 & \text{if there's no connection,} \\ 1 & \text{if } i \text{ and } j \text{ are neighboring entities,} \\ 2 & \text{if } \{i, j\} \text{ is an \{entity,edge\} pair,} \\ 3 & \text{if } i \text{ and } j \text{ are adjacent relations.} \end{cases} \tag{4}$$

The model then has the capacity to modify its attention scores based on the graph's connection types. Intuitively, this capacity would allow us to interpolate between different choices of M, or in the extreme case it can push model $M_{e,r}^{e,r}$, to

Dataset	#KG-text pairs (Train/Valid/Test)
WebNLG	34,352 / 4,316 / 4,224
EventNarrative	179,543 / 1,000 / 22,441

Table 1: Statistics of the supervised KG-to-Text datasets used for experimenting.

simulate any of the other more restrictive masks. For ease of reference, we explicitly state the type encoding whenever used.

Finally, after producing the new graph representation $\tilde{X}_l^g$ with equation (3), we *gather* the word representations from the graph representation, adding the new representations as a residual to X_l, and generate the output from the l-th layer:

$$\tilde{X}_l = \text{gather}(\tilde{X}_l^g) + X_l \tag{5}$$

5 Experiments

5.1 Datasets

We experiment on two KG-to-text supervised datasets: WebNLG v2.0 (Gardent et al., 2017; Shimorina and Gardent, 2018) and EventNarrative (Colas et al., 2021). We experiment with different configurations on the graph representation, attention mask, and type encoding on the WebNLG dataset, taking the best performing models to experiment further on EventNarrative. This is because of computational constraints caused by the size of EventNarrative. Table 1 outlines the statistical differences between the two datasets. We use the official data split for both.

WebNLG is a crowd-sourced RDF triple-to-text dataset manually crafted by human annotators. The dataset contains graphs from DBpedia (Auer et al., 2007) with up to 7 triples paired with one or more reference texts. As in Chen et al. (2020) and Ke et al. (2021), we evaluate on the 2.0 release [1].

EventNarrative is an automatically generated large-scale event-centric KG-to-text supervised dataset. Event KGs are extracted from Wikidata (Vrandečić and Krötzsch, 2014) and EventKG (Gottschalk and Demidova, 2018), which are then matched to Wikipedia sentences. EventNarrative contains a larger number of unique KG components compared to WebNLG.

5.2 Implementation and training details

We chose to use BART as our pre-trained LM (Lewis et al., 2020), and initialize its respective parameters with the Hugging Face's pre-trained bart-base checkpoint [2]. We left the default hyperparameters on the Global Attention module (BART) due to limited computational resources, instead experimenting on the Graph-aware attention module.

When evaluating, we follow the existing work for KG-to-text and report the model's performance with BLEU (Papineni et al., 2002), METEOR (Banerjee and Lavie, 2005), and ROUGE-L (Lin, 2004) scores as the automatic NLG metrics.

5.3 Baselines

Fine-tuned LM. To evaluate the effect of the graph-aware attention module in our framework, we compare with a vanilla fine-tuned BART LM, which is not additionally pre-trained on any graph-text specific task. We do so for both WebNLG and EventNarrative, noting that for EventNarrative such a baseline is the state-of-the-art.

Pre-trained KG-to-Text Models. We further compare our framework with models which have pre-trained LMs on additional tasks, including KGPT (Chen et al., 2020) and JointGT (Ke et al., 2021). KGPT performs an additional KG-to-text generation pre-training task on KGText, a loosely-supervised large-scale KG-to-text dataset, before finetuning. JointGT performs three additional pre-training tasks for KG reconstruction, text reconstruction, and KG-text alignment on the KGText dataset before finetuning. For a fair comparison with JointGT, we also compare our results to JointGT's BART pre-trained task, where they perform an additional text infilling and sentence permutation task on KGText.

5.4 Main results

Table 2 and Table 3 show our results on the WebNLG and EventNarrative datasets, respectively. On both datasets, we observe improvements over existing LM-based models with GAP. For BLEU score on WebNLG, we observe a +5.20% improvement over the state-of-the-art without any pre-training (Shimorina and Gardent, 2018) and a +1.65% improvement over BART. This improvement suggests that the graph-aware component of GAP makes use of the local neighborhood information when encoding graph components.

We outperform both KGPT and JointGT (on WebNLG), which rely on additional pre-training

[1] https://gitlab.com/shimorina/webnlg-dataset

[2] https://huggingface.co/facebook/bart-base

Model	Pre+	#Param	BLEU	METEOR	ROUGE
GCN (Marcheggiani and Perez-Beltrachini, 2018)	No	-	60.80‡	42.76 ‡	71.13‡
Shimorina and Gardent (2018)	No	-	61.00♯	42.00♯	71.00♯
KGPT w/o pretrain	No	177M	62.30 ‡	44.33 ‡	73.00‡
KGPT	Yes	177M	64.11‡	46.3‡	74.57‡
BART	Yes	140M	64.55	46.51	75.13
JointGT (BART) - w/ BARTPretrain	Yes	160M	64.60†	46.78†	75.74†
JointGT (BART) - w/ JointGTPretrain	Yes	160M	65.92†	**47.15**†	76.1†
GAP (**Ours**) - $M_e^{e,r}$	No	153M	65.92	46.81	76.22
GAP (**Ours**) - $M^{e,r} + \gamma$	No	153M	**66.20**	46.77	**76.36**

Table 2: Performance comparison on WebNLG. KGPT and JointGT, marked with † and ‡, re-printed from Chen et al. (2020) and Ke et al. (2021), have been pre-trained on one and three additional tasks, where *Pre+* denotes if additional pre-training was performed. We mark results from Shimorina and Gardent (2018) with ♯. We report our best models with and without type encoding, which have approximately the same number of parameters.

Model	BLEU	METEOR	ROUGE	BERTScore
BART	31.38	26.68	62.65	93.12
T5	12.8	22.77	52.06	89.59
JointGT	31.19	26.58	**64.91**	**93.68**
$M_e^{e,r}$	34.02	26.93	62.90	93.13
$M^{e,r} + \gamma$	**35.08**	**27.50**	64.28	93.38

Table 3: Performance comparison on EventNarrative. We compare to the pretrained baselines, T5 and BART, reprinted from (Colas et al., 2021), and adapt JointGT (Ke et al., 2021) to the dataset.

GAP	BLEU	METEOR	ROUGE
$M_e^{e,r}$	**65.92**	46.81	76.22
$M^{e,r}$	65.86	**46.86**	**76.28**
M_e^{e}	65.11	46.33	75.62
$M_{e,r}^{e,r}$	64.64	46.17	75.04

Table 4: Experimental results of the different masks applied to the WebNLG v2.0 test set.

tasks for graph-text reconstruction and alignment. On BLEU score, we observe an improvement of +1.81% and 2.09% over KGPT, and +1.32% and 1.6% over JointGT (with BARTPretrain). Further, our $M^{e,r}$ with *Type Encoding* model outperforms JointGT (with JointGTPretrain) by 0.28% without the need for any additional pre-training. JointGTPretrain refers to all three pre-trained tasks described in Ke et al. (2021). Instead of pre-training, we fill the gap with a modification to the encoder structure such that the model adapts to the graph structure. To summarize, we have shown that when adapting pre-trained language models such as BART, a careful modification of the encoder structure can better align the LM with the new task.

On EventNarrative, for model $M_e^{e,r}$ we achieve an improvement of +3.70%, +0.82%, +1.63% on BLEU, METEOR, and ROUGE, relative to BART, further demonstrating that the graph-aware structure and type encoder can perform comparatively well on large and more complex graphs. We note a similar trend to WebNLG, where the type encoder can give an additional performance improvement to the graph-structure component of the model. For comparison, we adapt JointGT to EventNarrative, using the hyperparameters from Ke et al. (2021). We note all models have similar BERTScores.

6 Analysis

6.1 Ablation Studies

We explore different maskings and type encodings for the graph-aware attention module on WebNLG. summarized on Table 4 and Table 5.

Masking Scheme. From bottom to top on Table 4, our first observation is that when relations directly attend to the neighboring relations, the performance drops by 1.28%, the largest difference. In fact, the results significantly improve when we completely block attention on relations (M_e^e). However, for the entities, it is always best to attend to their edges (relations) as well as their neighboring entities. The top two results are comparable (0.06% difference in BLEU score), and each one could be considered the best performing model depending on the evaluation metric. For relations, it might be somewhat helpful to not attend to neighboring relations, while for entities, attending to the relations will lead to better results (+0.81%).

Type Encoder. Table 5 shows the effect of type encoding on the results on WebNLG. To better un-

GAP w/ γ	BLEU	METEOR	ROUGE
$M_e^{e,r}$	65.34	46.31	75.59
$M^{e,r}$	**66.20**	**46.77**	**76.36**
M_e^e	65.24	46.49	75.44
$M_{e,r}^{e,r}$	65.43	46.54	75.75

Table 5: The results of different variations of our model with type encoding on the WebNLG v2.0 test set.

Model	Data Proportion			
	0.5%	1%	5%	10%
BART	33.92	39.08	52.24	56.58
JointGT	37.18	42.26	54.41	57.73
$M^{e,r} + \gamma$	**39.50**	**44.03**	**55.68**	**58.30**

Table 6: BLEU scores of various pre-trained models compared to GAP for few-shot learning on WebNLG.

GAP	#Triples	
	1-3	4-7
$M_e^{e,r}$	71.48	61.53
$M^{e,r}$	71.28	61.59
M_e^e	70.18	61.05
$M_{e,r}^{e,r}$	69.74	60.57

Table 7: BLEU scores for the different masks applied to the WebNLG v2.0 test set for different graph sizes.

derstand the effect of type encoding on each of the models, we compare Table 4 with Table 5. Recall that the type encoding $\gamma(T)$ for each model depends on the connections that exists in the model graph structure. For instance, the most general model $M_{e,r}^{e,r}$ has all four possible connection types encoded by equation (4), while the model with $M = M^{e,r}$ only has two types, which can be encoded by a restriction of equation (4). According to Table 4, the model $M_{e,r}^{e,r}$ performs worst without type encoding. However, because of its generality, i.e. having all the possible connection types, it is possible for this model to drift toward better configurations with the help of $\gamma(T)$. The results in Table 5 help support these insights for model $M_{e,r}^{e,r}$. Type encoding allows this model to simulate what we observed is best in the previous section, i.e. relations are better off not to attend to relations, whereas entities can attend to both while paying less attention to relations. This nuanced behavior seems to be achievable only via type encoding. Results for model with $M = M^{e,r}$ and type encoding also point towards this; type encoding seems to facilitate a non-uniform attention distribution based on the type and produces a better result.

6.2 Few-Shot Learning

To further reinforce our claims that our model alleviates the need for pre-training in the KG-to-text task, we consider various few-shot learning settings where only a small percentage of training instances were used for finetuning. As highlighted in Table 6, GAP outperforms all state-of-the-art pretrained-based approaches, without needing to pre-train, indicating that our fully graph-aware framework is more appropriate than established pre-trained tasks, especially when such data is not avialable.

6.3 KG Size

As in Ke et al. (2021), we divide the WebNLG test set into two subsets (1-3 and 4-7 triples) to compare the performance of our different masking configurations. Table 7 shows that while all configurations perform similarly for larger graphs, the difference in performance is clearer on smaller graphs, where $M_e^{e,r}$ performs +1.74% better than $M_{e,r}^{e,r}$, suggesting that relations paying attention to relations can add too much complexity to the model, especially on simpler graph structures.

6.4 Interpretability

We begin to interpret KG-to-text models by analyzing the graph-attention weights induced by each graph structure on a per-sample basis, analogous to analyzing node-to-node attention weights in the KG question-answering domain (Yasunaga et al., 2021). By introducing a framework to the KG-to-text task, we can condition the changes in the output text on the different components of the framework, including the masking and type encoder. We can then observe the differences in the output text based on the graph's topological structure or what relations and entities attend to.

In Figure 4 we show an example KG representing *Aenir*, an Australian fantasy novel, with its relations (orange) and entities (blue) along with the attention heatmaps and outputs from two of our framework decisions. The left (a) heatmap and output corresponds to our best performing model without type encoding, $M_e^{e,r}$, while the right (b) corresponds to M_e^e. We choose these two masking configurations, because the attention-weight differences are apparent.

From (a), entities attend to both entities and relations, whereas relations only attend to entities.

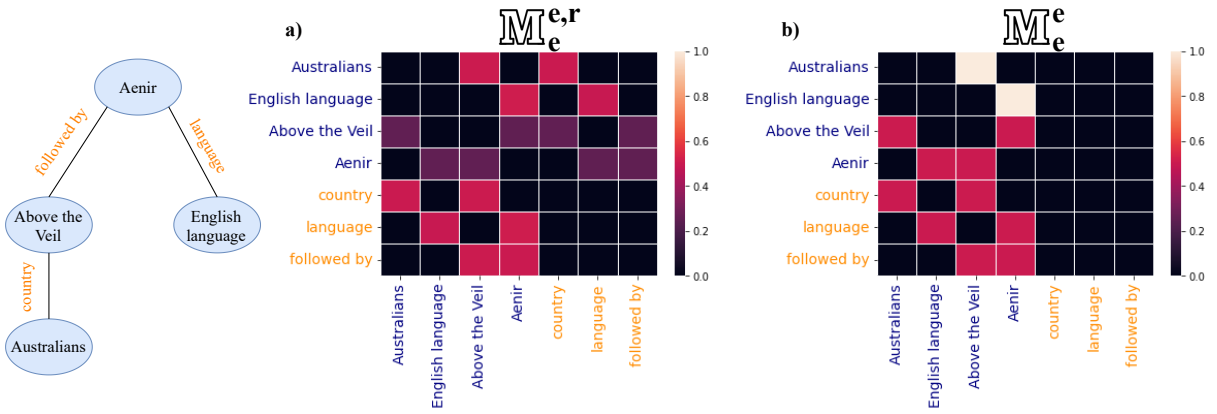

Figure 4: Interpreting KG-to-text models via analyzing graph attention weights, which the graph-aware encoder activates. We show each model's output for further emphasis.

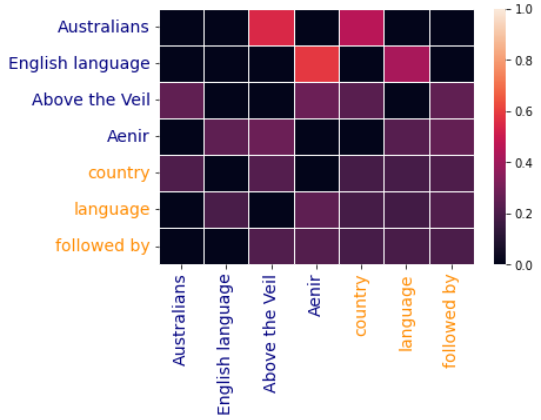

Figure 5: An additional case study of the graph attention weights for model $M_{e,r}^{e,r}$ with type encoding.

Interestingly, the attention distribution appears uniform across all graph components (both for entities and relations). From (b) we see a similar uniform distribution across entities and relation attending to only entities. Thus, in (a), while relation 'country' attends to 'Australians' and vice-versa, in (b) 'Australians' does not attend to 'country', perhaps giving a difference in the output, as the final output in (a) contains 'from the country of Australia' while the text in (b) does not. Moreover, in both (a) and (b) 'Above the Veil' is the subject of the second clause. However, 'Above the Veil' attends to 'country' only in (a), therefore influencing (a)'s output of '*Above the Veil* which is from the *country* of Australia'. Instead, (b) introduces some redundancy in its second clause instead of transcribing new information from the KG.

Figure 5 shows the output sentence, and the attention heatmap produced by our most general model with $M = M_{e,r}^{e,r}$ and type encoding, on the graph shown in Figure 4. We examine the differences between this model, referred to as model (1), and the model with $M = M_e^{e,r}$ and no type encoding, referred to as model (2). First, note that in terms of BLEU score (1) performs slightly worse than (2), however a human annotator may rank (1) over (2), as (1) is more concise while communicating the same information. For example, (1) uses the word 'sequel to' rather than 'followed by' and 'published in' instead of 'from the country', which can sound more natural to humans. Particularly, *Australians* pays less attention to *country*, compared to model (2), perhaps hinting at this result. Our framework provides a first step in interpreting this result by allowing one to compare different attention-weights across multiple models. With this in mind, we call upon future work to design more specific evaluation metrics for the KG-to-text task.

7 Conclusion

We presented GAP, a graph-aware language model framework for KG-to-text generation. Our framework instills the local information captured by graph attention into the global contextualized word vector representation within pre-trained LMs. We demonstrated multiple configurations of our framework by introducing a *graph-aware attention masking scheme* and novel *type encoder* module, and through qualitative analysis showed that GAP outperforms existing KG-to-text models, including those that rely on additional auxiliary pre-training

tasks. By closely examining the different framework configurations, we introduce the capacity to interpret KG-to-text outputs through a graph's attention structure and topology.

8 Broader Impacts

GAP provides researchers with a state-of-the-art framework for KG-to-text models. Though we experiment with supervised baselines which include a handcrafted dataset, WebNLG, and an automatically generated dataset, EventNarrative, repositories of structured data exist in the clinical (Johnson et al., 2016), medical (Bodenreider, 2004), and news crises (Leetaru and Schrodt, 2013; Ward et al., 2013) domains. By transforming clinical data into natural language narratives, patients with low health-literacy can benefit by more easily understanding their electronic medical records (EMRs), and doctors can more easily transcribe patient data for future use cases, i.e. connecting such data to the medical literature. Such models can also help analysts more easily understand crises data from various news sources, in turn helping them evaluate cause-effect relationships and detect misinformation. While malicious actors can exploit generative models for disinformation, we discourage the use of GAP in generating such data and openly release our model to help combat such efforts.

References

Oshin Agarwal, Heming Ge, Siamak Shakeri, and Rami Al-Rfou. 2021. Knowledge graph based synthetic corpus generation for knowledge-enhanced language model pre-training. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 3554–3565, Online. Association for Computational Linguistics.

Sören Auer, Christian Bizer, Georgi Kobilarov, Jens Lehmann, Richard Cyganiak, and Zachary Ives. 2007. Dbpedia: A nucleus for a web of open data. In *The semantic web*, pages 722–735. Springer.

Satanjeev Banerjee and Alon Lavie. 2005. Meteor: An automatic metric for mt evaluation with improved correlation with human judgments. In *Proceedings of the acl workshop on intrinsic and extrinsic evaluation measures for machine translation and/or summarization*, pages 65–72.

Rajarshi Bhowmik and Gerard de Melo. 2018. Generating fine-grained open vocabulary entity type descriptions. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 877–888.

Olivier Bodenreider. 2004. The unified medical language system (umls): integrating biomedical terminology. *Nucleic acids research*, 32(suppl_1):D267–D270.

Kurt Bollacker, Colin Evans, Praveen Paritosh, Tim Sturge, and Jamie Taylor. 2008. Freebase: a collaboratively created graph database for structuring human knowledge. In *Proceedings of the 2008 ACM SIGMOD international conference on Management of data*, pages 1247–1250.

Deng Cai and Wai Lam. 2020. Graph transformer for graph-to-sequence learning. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pages 7464–7471.

Wenhu Chen, Yu Su, Xifeng Yan, and William Yang Wang. 2020. KGPT: Knowledge-grounded pre-training for data-to-text generation. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 8635–8648, Online. Association for Computational Linguistics.

Anthony Colas, Ali Sadeghian, Yue Wang, and Daisy Zhe Wang. 2021. Eventnarrative: A large-scale event-centric dataset for knowledge graph-to-text generation.

Joe Davison, Joshua Feldman, and Alexander M Rush. 2019. Commonsense knowledge mining from pretrained models. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 1173–1178.

Bayu Distiawan, Jianzhong Qi, Rui Zhang, and Wei Wang. 2018. Gtr-lstm: A triple encoder for sentence generation from rdf data. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1627–1637.

Claire Gardent, Anastasia Shimorina, Shashi Narayan, and Laura Perez-Beltrachini. 2017. The webnlg challenge: Generating text from rdf data. In *Proceedings of the 10th International Conference on Natural Language Generation*, pages 124–133.

Simon Gottschalk and Elena Demidova. 2018. Eventkg: A multilingual event-centric temporal knowledge graph. In *European Semantic Web Conference*, pages 272–287. Springer.

Jian Guan, Yansen Wang, and Minlie Huang. 2019. Story ending generation with incremental encoding and commonsense knowledge. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, pages 6473–6480.

Zhijiang Guo, Yan Zhang, Zhiyang Teng, and Wei Lu. 2019. Densely connected graph convolutional networks for graph-to-sequence learning. *Transactions of the Association for Computational Linguistics*, 7:297–312.

Suchin Gururangan, Ana Marasović, Swabha Swayamdipta, Kyle Lo, Iz Beltagy, Doug Downey, and Noah A Smith. 2020. Don't stop pretraining: Adapt language models to domains and tasks. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 8342–8360.

Shizhu He, Cao Liu, Kang Liu, and Jun Zhao. 2017. Generating natural answers by incorporating copying and retrieving mechanisms in sequence-to-sequence learning. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 199–208.

Haozhe Ji, Pei Ke, Shaohan Huang, Furu Wei, Xiaoyan Zhu, and Minlie Huang. 2020. Language generation with multi-hop reasoning on commonsense knowledge graph. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 725–736.

Zhijing Jin, Qipeng Guo, Xipeng Qiu, and Zheng Zhang. 2020. Genwiki: A dataset of 1.3 million content-sharing text and graphs for unsupervised graph-to-text generation. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 2398–2409.

Alistair EW Johnson, Tom J Pollard, Lu Shen, H Lehman Li-Wei, Mengling Feng, Mohammad Ghassemi, Benjamin Moody, Peter Szolovits, Leo Anthony Celi, and Roger G Mark. 2016. Mimic-iii, a freely accessible critical care database. *Scientific data*, 3(1):1–9.

Mihir Kale and Abhinav Rastogi. 2020. Text-to-text pre-training for data-to-text tasks. In *Proceedings of the 13th International Conference on Natural Language Generation*, pages 97–102.

Pei Ke, Haozhe Ji, Yu Ran, Xin Cui, Liwei Wang, Linfeng Song, Xiaoyan Zhu, and Minlie Huang. 2021. JointGT: Graph-text joint representation learning for text generation from knowledge graphs. In *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*, pages 2526–2538, Online. Association for Computational Linguistics.

Diederik P. Kingma and Jimmy Ba. 2015. Adam: A method for stochastic optimization. *CoRR*, abs/1412.6980.

Rik Koncel-Kedziorski, Dhanush Bekal, Yi Luan, Mirella Lapata, and Hannaneh Hajishirzi. 2019. Text generation from knowledge graphs with graph transformers. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 2284–2293.

Kalev Leetaru and Philip A Schrodt. 2013. Gdelt: Global data on events, location, and tone, 1979–2012. In *ISA annual convention*, volume 2, pages 1–49. Citeseer.

Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Veselin Stoyanov, and Luke Zettlemoyer. 2020. Bart: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7871–7880.

Chin-Yew Lin. 2004. Rouge: A package for automatic evaluation of summaries. In *Text summarization branches out*, pages 74–81.

Manuel Mager, Ramón Fernandez Astudillo, Tahira Naseem, Md Arafat Sultan, Young-Suk Lee, Radu Florian, and Salim Roukos. 2020. Gpt-too: A language-model-first approach for amr-to-text generation. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 1846–1852.

Diego Marcheggiani and Laura Perez-Beltrachini. 2018. Deep graph convolutional encoders for structured data to text generation. In *INLG*.

Amit Moryossef, Yoav Goldberg, and Ido Dagan. 2019. Step-by-step: Separating planning from realization in neural data-to-text generation. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 2267–2277.

Vaishali Pal, Manish Shrivastava, and Irshad Bhat. 2019. Answering naturally: Factoid to full length answer generation. In *Proceedings of the 2nd Workshop on New Frontiers in Summarization*, pages 1–9.

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. Bleu: a method for automatic evaluation of machine translation. In *Proceedings of the 40th annual meeting of the Association for Computational Linguistics*, pages 311–318.

Alec Radford, Karthik Narasimhan, Tim Salimans, and Ilya Sutskever. Improving language understanding by generative pre-training.

Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, Ilya Sutskever, et al. 2019. Language models are unsupervised multitask learners. *OpenAI blog*, 1(8):9.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J Liu. 2020. Exploring the limits of transfer learning with a unified text-to-text transformer. *Journal of Machine Learning Research*, 21(140):1–67.

Leonardo F. R. Ribeiro, Yue Zhang, and Iryna Gurevych. 2021. Structural adapters in pretrained language models for AMR-to-Text generation. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 4269–4282, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Leonardo FR Ribeiro, Martin Schmitt, Hinrich Schütze, and Iryna Gurevych. 2020a. Investigating pretrained language models for graph-to-text generation. *arXiv preprint arXiv:2007.08426*.

Leonardo FR Ribeiro, Yue Zhang, Claire Gardent, and Iryna Gurevych. 2020b. Modeling global and local node contexts for text generation from knowledge graphs. *Transactions of the Association for Computational Linguistics*, 8:589–604.

Martin Schmitt, Leonardo FR Ribeiro, Philipp Dufter, Iryna Gurevych, and Hinrich Schütze. 2021. Modeling graph structure via relative position for text generation from knowledge graphs. In *Proceedings of the Fifteenth Workshop on Graph-Based Methods for Natural Language Processing (TextGraphs-15)*, pages 10–21.

Peter Shaw, Jakob Uszkoreit, and Ashish Vaswani. 2018. Self-attention with relative position representations. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 2 (Short Papers)*, pages 464–468.

Anastasia Shimorina and Claire Gardent. 2018. Handling rare items in data-to-text generation. In *Proceedings of the 11th International Conference on Natural Language Generation*, pages 360–370.

Yixuan Su, David Vandyke, Sihui Wang, Yimai Fang, and Nigel Collier. 2021. Plan-then-generate: Controlled data-to-text generation via planning. In *Findings of the Association for Computational Linguistics: EMNLP 2021*, pages 895–909.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in neural information processing systems*, pages 5998–6008.

Petar Veličković, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Liò, and Yoshua Bengio. 2018. Graph attention networks. In *International Conference on Learning Representations*.

Denny Vrandečić and Markus Krötzsch. 2014. Wikidata: a free collaborative knowledgebase. *Communications of the ACM*, 57(10):78–85.

Ivan Vulić, Edoardo Maria Ponti, Robert Litschko, Goran Glavaš, and Anna Korhonen. 2020. Probing pretrained language models for lexical semantics. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 7222–7240.

Michael D Ward, Andreas Beger, Josh Cutler, Matthew Dickenson, Cassy Dorff, and Ben Radford. 2013. Comparing gdelt and icews event data. *Analysis*, 21(1):267–297.

TH Wen, M Gašić, N Mrkšić, LM Rojas-Barahona, PH Su, D Vandyke, and S Young. 2016. Multi-domain neural network language generation for spoken dialogue systems. In *2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL HLT 2016-Proceedings of the Conference*, pages 120–129.

David S Wishart, Yannick D Feunang, An C Guo, Elvis J Lo, Ana Marcu, Jason R Grant, Tanvir Sajed, Daniel Johnson, Carin Li, Zinat Sayeeda, et al. 2018. Drugbank 5.0: a major update to the drugbank database for 2018. *Nucleic acids research*, 46(D1):D1074–D1082.

Michihiro Yasunaga, Hongyu Ren, Antoine Bosselut, Percy Liang, and Jure Leskovec. 2021. Qa-gnn: Reasoning with language models and knowledge graphs for question answering. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 535–546.

Alexander Yates, Michele Banko, Matthew Broadhead, Michael J Cafarella, Oren Etzioni, and Stephen Soderland. 2007. Textrunner: open information extraction on the web. In *Proceedings of Human Language Technologies: The Annual Conference of the North American Chapter of the Association for Computational Linguistics (NAACL-HLT)*, pages 25–26.

Hao Zhou, Tom Young, Minlie Huang, Haizhou Zhao, Jingfang Xu, and Xiaoyan Zhu. 2018. Commonsense knowledge aware conversation generation with graph attention. In *IJCAI*, pages 4623–4629.

Jie Zhu, Junhui Li, Muhua Zhu, Longhua Qian, Min Zhang, and Guodong Zhou. 2019. Modeling graph structure in transformer for better amr-to-text generation. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 5459–5468.

A Hyperparameter Details

As followed by Ke et al. (2021) and BART, we used a Byte-Pair Encoding (BPE) vocabulary (Radford et al., 2019) with a size of 50,265. The model's parameters were optimized via Adam (Kingma and Ba, 2015), with a batch size of 16, a learning rate of 3e-5, and a maximum graph size of 50 and 60 for WebNLG and EventNarrative, respectively. Table 8 provides the model hyperparameter settings used for experimenting on both the WebNLG and EventNarrative datasets. We keep all listed hyperparameters constant with respect to the GAP configurations. We increase *num nodes* for the EventNarrative dataset due to the properties of the dataset, i.e. the possibility of having graphs composed of more than seven triples. We also set the *eval period* to 5,000 for EventNarrative due to its size, containing approximately 22,000 samples in its test set. As in (Colas et al., 2021), we set the max output size to 512 for all experiments on EventNarrative BLEU score on the validation set was used for model selection. Each model was trained on two NVIDIA RTX 2080 Ti GPUs.

Hyperparameter	**WebNLG**	**EventNarrative**
Learning Rate	2.00E-05	2.00E-05
Warmup Steps	1600	1600
Eval Period	500	**5000**
Beam Size	5	5
Length Penalty	1	1
Optimizer	Adam	Adam
ϵ	1.00E-08	1.00E-08
Num Nodes	50	**60**
Num Relations	60	60
Embedding Size	128	128
Num Global Layers	6	6
Num Graph-aware Layers	6	6
Batch Size	16	16

Table 8: Hyperparameters for GAP on both the WebNLG and EventNarrative datasets.

B Additional Experimental Results

We provide additional experimental results on both WebNLG v2.0 and EventNarrative for the proposed GAP framework for reference and further analysis.

B.1 Graph Length

Here we examine a comparative study to that of Table 7 for the EventNarrative dataset. Table 9 reveals an exponential decay in BLEU score, with lengths 1-3, 4-7, and 7+ having 44.48%, 23.86%, 11.47%, respectively. Compared to WebNLG, the BLEU scores are significantly lower, suggesting that EventNarrative is a more challenging dataset. Table 10 gives a brief synopsis of the dataset sizes with respect to the number of triples. Compared to WebNLG which has no KGs greater than length 7, EventKG contains over 1,000 KGs larger than length 7, making the dataset more diverse.

GAP	#Triples		
	1-3	4-7	7+
$M_e^{e,r}$	44.48	23.86	11.47

Table 9: BLEU scores for the EventNarrative test set for different graph sizes.

Datasets	#Triples		
	1-3	4-7	7+
WebNLG	1,017	583	0
EventNarrative	16,103	5,152	1,184

Table 10: Distribution for number of triples in both the WebNLG and EventNarrative datasets.

B.2 Entity Accuracy

To give more insight into KG-to-text generation with GAP, we provide the results for *entity accuracy*. We define *entity accuracy* to be the number of entities from the KG that appear in the generated text over those that appear in the reference text. Table 11 shows that all models perform exceedingly well in generating the correct entities from their respective KGs, suggesting that future KG-to-text research should focus on sentence structure and descriptors, i.e. quantifiers and determiners.

Datasets	Accuracy	
	w/o $\gamma(T)$	w/ $\gamma(T)$
$M_e^{e,r}$	94.06	94.04
$M^{e,r}$	93.99	94.48
M_e^{e}	93.64	94.50
$M_{e,r}^{e,r}$	93.82	94.28

Table 11: Entity accuracy on the WebNLG test set.

C Additional Examples and Error Analysis

We now present example outputs generated by GAP both on the WebNLG and EventNarrative dataset in Tables 12 and 13 below.

C.1 WebNLG

We showcase five different examples from the WebNLG test set output by our $M^{e,r} + \gamma(T)$ (Pre-

diction 1) and $M^{e,r}_{e,r} + \gamma(T)$ (Prediction 2) models. As can be seen in all the examples, GAP is able to generate fluent and complete sentences. In the first two examples, the output from both models are identical. The outputs from the third example can be viewed as paraphrases of one another, where Prediction 1 mentions *'US national'* while Prediction 2 instead uses the adjective *'American'* to convey the same information. Furthermore, in both predictions we learn that *'Alan Bean'* was a *'test pilot'* and *'selected by NASA'* but in slightly different formats. In the fourth example, Prediction 2 is missing the name of the rock band, *'NRBQ'*, while maintaining the rest of the information. Like the third example, the predictions in the fifth example are paraphrases.

C.2 EventNarrative

Because of the length of output in EventNarrative, we present four different types of examples to elaborate on the limitations of KG-to-text models. Here, we show example outputs from our $M^{e,r}_{e}$ (Prediction 1) and $M^{e,r} + \gamma(T)$ (Prediction 2) models. In the first example, we observe a **contradiction** in both Prediction 1 and 2: the gubernatorial candidate was a democratic nominee, while our predictions conveyed otherwise. The second example shows two predictions which are identical, both **missing information**, specifically *'ozone park'* and *'for three - year - olds and up'*. Upon further inspection, these two pieces of information are not within the KG. Similarly, in the third example the only piece of information missing from the predictions, namely *'cork county board'*, is not part of the KG. This example also contains **invalid information**, *'112th'* instead of *'103rd'*. The last example also contains **invalid information** regarding the dates in both predictions. Additionally, Prediction 2 is **missing information** about the *'village of ignacewo'*.

Prediction 1	Amsterdam Airport Schiphol serves the city of Amsterdam and is -3.3528 above sea level . The runway name is 18L/36R Aalsmeerbaan and it has a length of 2014.0 .
Prediction 2	Amsterdam Airport Schiphol serves the city of Amsterdam and is -3.3528 above sea level . The runway name is 18L/36R Aalsmeerbaan and it has a length of 2014.0 .
Reference	Amsterdam Airport Schiphol is -3.3528 above sea level , has a runway name 18L/36R'Aalsmeerbaan which is 2014.0 in length and serves the city of Amsterdam .
Prediction 1	Baked Alaska is from Hong Kong and the United States . The main ingredients are meringue , ice cream , sponge cake or Christmas pudding .
Prediction 2	Baked Alaska is from Hong Kong and the United States . The main ingredients are meringue , ice cream , sponge cake or Christmas pudding .
Reference	Baked Alaska comes from both Hong Kong and the United States . The main ingredients are Meringue , ice cream , sponge cake or Christmas pudding .
Prediction 1	Alan Bean is a US national who was born in Wheeler , Texas . He served as a test pilot before being selected by NASA in 1963 . He is now retired .
Prediction 2	Alan Bean is an American test pilot who was born in Wheeler , Texas . He was selected by NASA in 1963 . He is now retired .
Reference	The American test pilot Alan Bean (born in Wheeler , Texas) was selected by NASA in 1963 . He is now retired .
Prediction 1	Al Anderson is a member of rock band NRBQ . Rock music originated from country music which originated from blues music . A musical fusion of rock music is bhangra music .
Prediction 2	Al Anderson plays rock music which originated from blues and country music . Bhangra music is part of the fusion genre , partly coming from Rock music which uses the banjo .
Reference	Al Anderson plays with the rock band NRBQ . Rock has its origins in the blues and country music , where the banjo is played , and Bhangra is a rock fusion .
Prediction 1	Paulo Sousa is the manager of ACF Fiorentina . He has played for Inter Milan , Juventus FC and Maccabi Tel Aviv F.C .
Prediction 2	Paulo Sousa is the manager of ACF Fiorentina . He has played for Inter Milan , Maccabi Tel Aviv FC and Juventus FC .
Reference	Paulo Sousa once played for Inter Milan and Juventus FC . He now manages ACF Fiorentina and plays for Maccabi Tel Aviv F.C .

Table 12: Examples of output texts generated from $M^{e,r} + \gamma(T)$ (Prediction 1) and $M^{e,r}_{e,r} + \gamma(T)$ (Prediction 2) on the WebNLG test set.

Prediction 1	the 1971 mississippi gubernatorial election was held on 02 november 1971 . incumbent governor bill waller won re - election to a second term , defeating the democratic nominee , former speaker of the chamber of deputies thomas h . williams , by a wide margin .
Prediction 2	the 1971 mississippi gubernatorial election took place on 02 november 1971 , in order to elect the governor of mississipp . incumbent governor bill waller , who was first elected in 1972 , faced no opposition from democrats and avoided a runoff .
Reference	the 1971 mississippi gubernatorial election took place on 02 november 1971 for the post of governor of mississippi. democrat bill waller was elected into office and served as governor of mississippi from 1972 to 1976.
Prediction 1	the sport page handicap is an american thoroughbred horse race run annually at aqueduct racetrack in queens , new york .
Prediction 2	the sport page handicap is an american thoroughbred horse race run annually at aqueduct racetrack in queens , new york .
Reference	the sport page handicap was an american thoroughbred horse race run annually at aqueduct racetrack in ozone park, queens, new york for three - year - olds and up.
Prediction 1	the 1991 cork senior hurling championship was the 112th staging of the cork premier hurling competition , and the 86th staging by a team from cork .
Prediction 2	the 1991 cork senior hurling championship was the 112th staging of the cork premier club hurling competition since its establishment in 1887 .
Reference	the 1991 cork senior hurling championship was the 103rd staging of the cork senior hurling championship since its establishment by the cork county board in 1887.
Prediction 1	the first battle of ignacewo was one of the first battles of the january uprising . it took place on january 28 , 1863 , near the village of ignakewo , konin county in southwestern corner of russian - controlled congress poland .
Prediction 2	the first battle of ignacewo was one of the first battles of the january uprising . it took place on january 6 , 1863 , near the village of konin , in congress poland .
Reference	the first battle of ignacewo was one of many clashes of the january uprising. it took place on may 8, 1863, near the village of ignacewo, konin county, which at that time belonged to russian empire's congress poland.

Table 13: Examples of output texts generated from $M_e^{e,r}$ (Prediction 1) and $M^{e,r} + \gamma(T)$ (Prediction 2) on the EventNarrative test set.

Content Type Profiling of Data-to-Text Generation Datasets

Ashish Upadhyay and **Stewart Massie**
School of Computing
Robert Gordon University
Aberdeen, UK
{a.upadhyay,s.massie}@rgu.ac.uk

Abstract

Data-to-Text Generation (D2T) problems can be considered as a stream of time-stamped events with a text summary being produced for each. The problem becomes more challenging when event summaries contain complex insights derived from multiple records either within an event, or across several events from the event stream. It is important to understand the different types of content present in the summary to help us better define the system requirements so that we can build better systems. In this paper, we propose a novel typology of content types, that we use to classify the contents of event summaries. Using the typology, a profile of a dataset is generated as the distribution of the aggregated content types which captures the specific characteristics of the dataset and gives a measure of the complexity present in the problem. Through extensive experiments on different D2T datasets we demonstrate that neural generative systems specifically struggle to generate contents of complex types, highlighting the need for improved D2T techniques.

1 Introduction

An ecologically valid task requires the automated systems to resemble the real-world scenario as closely as possible in its output (de Vries et al., 2020). Accordingly, a Data-to-Text Generation (D2T) system needs to convey important insights extracted from the data in the textual summaries (Reiter, 2007; Gatt and Krahmer, 2018). Most D2T problems can be seen as a stream of time-stamped events with a textual summary of each event presenting the insights. An event is the time-period of interest for which the textual summary is written. For example, in sports reporting - a game played between two teams can be an event; whereas in weather forecasting - the time period and location for which the forecast is written can be considered one full event. The summaries can contain different types of facts with information sometimes coming from multiple events in the stream. In most cases, the facts in an event summary are verbatim of input records. Other times, these facts are derived from multiple records of either the same or multiple events in the stream. As an example, we show a part of baseball summary with multiple types of content in Figure 1.

The Bucks (10-10) *handled* **the Heat (9-9) 109-85** on **Friday night in Milwaukee**. ***It was the second victory over Miami for the Bucks this season after emerging victorious in Miami 91-84 on Nov. 16***. *Milwaukee fell behind early but clawed back into the game in the second quarter and held a four-point advantage at half. The Bucks were led by an unlikely face in Kendall Marshall*, who scored a ***season-high*** **20 points (7-8 FG, 4-5 3Pt) in 24 minutes**.

Figure 1: Part of a basketball summary showing different types of content. Information in "**bold**" such as Bucks' points in the game (109) can be directly copied from input data, while the ones in "*italics*" needs to be derived from multiple records such as the fact - Kendall Marshall leading the Bucks team. Finally, the information in "***bold & italics***" such as 'this was Miami's second victory against Bucks' are derived using records from multiple events in the stream.

A distribution of different content types in the summaries of a dataset can be used to generate its profile that can capture the specific characteristics of the dataset and provide a measure of complexity present in the problem. The dataset profile can help us better define the D2T system requirements, such as: the type of system to build - is a complex domain-specific system required or can a general system be effective; or any information gap (Thomson et al., 2020b) that needs to be bridged at the data level. Generally a problem's complexity is identified with the evaluation of systems built for

Proceedings of the 29th International Conference on Computational Linguistics, pages 5770–5782
October 12–17, 2022.

the task. There are several methods to evaluate the D2T systems, mostly by measuring the factual accuracy of generated texts (Thomson and Reiter, 2020; Wiseman et al., 2017; Garneau and Lamontagne, 2021; Kasner et al., 2021) or lexical similarity of generated texts with reference texts (Papineni et al., 2002; Lin, 2004; Zhang et al., 2020b). These, whilst being promising are reactive measures where a full cycle of system development is needed to evaluate both the dataset and D2T system together. Whereas dataset profiling can be a proactive measure to gain important insights about the task before even starting system development. There are other utilities of dataset profiling method as well, most notably, it can be used as a measure of dataset's complexity in datasheets for dataset (Gebru et al., 2021).

There are other type of datasets in D2T as well such as E2E (Dušek et al., 2020) or WebNLG (Colin et al., 2016) that do not follow the event based time-series setting. These datasets mostly focus on improving general ability of generation models such as transcribing a set of records, possibly in a domain-agonist setting. In this work, we do not address such datasets and rather focus on those which follow a time-series structure, where summaries may contain facts derived from across several event records, and also may require content selection on input data. In this paper, we propose a typology of different content types in D2T summaries based on the source of their information in the event stream. Our key contributions are as follows [1]:

- we propose a novel typology of content types in D2T summaries;
- we use the proposed typology to profile datasets and understand their characteristics;
- we demonstrate the challenge facing generation systems in producing complex contents.

The rest of the paper is organised as follows: in Section 2 we formally define the D2T task and discuss our proposed content type typology. We then go on to describe our experimental set up in Section 3, and discuss the results in Section 4. Some related works are discussed in Section 5, before concluding the paper in Section 6.

[1] code, data, and results are at https://github.com/ashishu007/Content-Type-Profiling

2 Methodology

The idea proposed in this paper is that we can create a profile capturing the specific characteristics of a dataset by looking at the distribution of content types present in its summaries. The first step towards achieving this is to formally define the main concepts (Section 2.1). Then, in second step, the content typology is defined by identifying the different types of facts that can be included in an event summary (Section 2.2).

2.1 Formalisation of D2T Generation Task

We start with formalising the concepts in a time-stamped D2T dataset. A data instance in such **D2T dataset** ($\mathcal{DB}$) is an **event** ($\mathcal{E}_i$) with a **data structure** ($\mathcal{D}_i$) for which a **textual summary** ($\mathcal{S}_i$) is written summarising the insights and information of the event. A data structure consists of multiple **entities** ($\mathcal{O}$) that are the objects involved in the event, and each entity is described by multiple **features** ($\mathcal{F}$) which are the attributes of those entities; such that:

$$\mathcal{DB} = [\mathcal{E}_{i-e}, \cdots, \mathcal{E}_i, \cdots, \mathcal{E}_{i+e}]$$
$$\mathcal{E}_i = \{\mathcal{D}_i, \mathcal{S}_i\}$$
$$\mathcal{D}_i = \{\mathcal{O}_1, \mathcal{O}_2, \cdots, \mathcal{O}_o\}$$
$$\mathcal{O}_o = \{\mathcal{F}_1, \mathcal{F}_2, \cdots, \mathcal{F}_f\}$$

When building a text generation system g for a D2T task, the summary of an event is the function of current event as well as other events in stream:

$$\mathcal{S}_i = g(\mathcal{D}_i, \mathcal{DB})$$

A value $\mathcal{R}$ will be recorded for each feature $\mathcal{F}$ of an entity $\mathcal{O}$ which is considered a record in the data structure. So, an input data structure flattened into a sequence of records (mostly for training a neural generation system) will be:

$$\mathcal{D}_i = \{(\mathcal{R}_{1,1}, \mathcal{R}_{1,2}, \cdots, \mathcal{R}_{1,f}), \cdots (\mathcal{R}_{2,1}, \mathcal{R}_{2,2}, \cdots, \mathcal{R}_{2,f}), \cdots (\mathcal{R}_{o,1}, \mathcal{R}_{o,2}, \cdots, \mathcal{R}_{o,f})\}$$

2.2 Content Type Typology

The textual summary in a D2T dataset may contain facts derived from different sources. To begin with, a fact can be derived from either the same event or from the records of different events. For example, a basketball game summary generally mentions different stats scored by players in the game. Such

information is explicitly present in the input data of the game and can be directly copied to the output summary. Most times, summaries also mentions the average stats recorded by a player in past few games. To derive such facts, the generation system needs to consider the records from previous games as well. So based on the event source, a fact can be categorised as either **intra-event** (derived from the current event's records) or **inter-event** (derived from across-event records).

The intra-event category can be further granulated to identify the difficulty of generating a fact within the same event. Again, taking an example from a basketball summary, among many things, the summary could either mention some specific stat (points or rebounds) scored by a player in the game, or mention if the player has scored a double-double [2]. The information of specific stat of players is explicitly present in the input data, which can be directly copied to the output summary. While the information that the player scored a double-double is not explicitly present in the input data, which needs to be derived from several records of the player. So, within intra-event, there can be two different types of facts: **basic**, that can be just copied directly from the input data; and **complex**, that needs to be derived from the multiple records of the same event.

Considering the following notations: each summary $\mathcal{S}_i$ is a combination of multiple **sentences** $\mathcal{T}$, which will contain at-least one or more **fact** $\mathcal{L}$.

$$\mathcal{S}_i = \{\mathcal{T}_1, \mathcal{T}_2, \cdots, \mathcal{T}_j\}$$
$$\mathcal{T}_j = \{\mathcal{L}_1, \mathcal{L}_2, \cdots, \mathcal{L}_k\}$$

Thus, a Content Type typology of three classes is proposed based on the types of facts an event summary can contain:

Intra-Event Basic ($\mathcal{B}$): a fact that is copied from the input record set of the same event.

$$\mathcal{B} \iff \mathcal{L}_k = \mathcal{R}_{i,o,f}$$

Intra-Event Complex ($\mathcal{C}$): a fact that is derived from multiple records of the same event.

$$\mathcal{C} \iff \mathcal{L}_k = \mathcal{R}_{i,o,f} \otimes \mathcal{R}_{i,o\pm l,f\pm m} \otimes \cdots$$

Inter-Event ($\mathcal{A}$): a fact that is either copied or derived from the records of multiple events.

$$\mathcal{A} \iff \mathcal{L}_k = \mathcal{R}_{i,r} \otimes \mathcal{R}_{i-n,o\pm l,f\pm m} \otimes \cdots$$

[2] https://en.wikipedia.org/wiki/Double-double

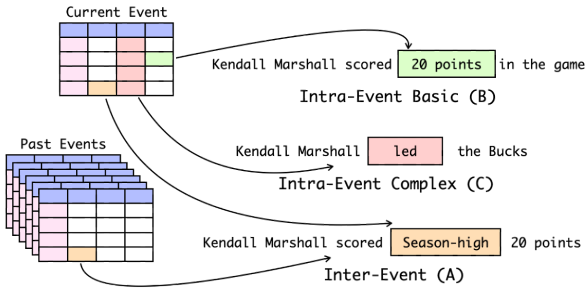

Figure 2: Different content types in a D2T summary. The Intra-Event Basic content is taken from only one record, while the Intra-Event Complex content is derived from multiple records of the same event. Finally, the Inter-Event content is derived from the records of multiple events.

where, i, j, & k denote an event, an object, and a feature respectively. l, m, n are positive integers and $\otimes$ is an operation that requires inference between more than one records. Thus, a sentence $\mathcal{T}$ can be assigned into one or more content type if it contains at-least one fact of that type.

Taking the example from Figure 1, the last sentence in the summary: "The *Bucks* were *led* by an unlikely face in *Kendall Marshall*, who scored a *season-high 20 points* (*7-8 FG, 4-5 3Pt*) in *24 minutes*" has multiple facts. To calculate the fact that *Kendall Marshall led the Bucks*, the generation system should be able to analyse the records of all players in Bucks team, which is **Intra-Event Complex** ($\mathcal{C}$) type of fact. Then the fact that he scored *season-high 20 points*, will only be calculated by analysing the records of all games in which Kendall Marshall played, which is **Inter-Event** ($\mathcal{A}$) type of fact. And finally, the shot-breakdown (*7-8 FG, 4-5 3Pt*) and *number of minutes* he played are the facts that can be directly copied from the input, which are **Intra-Event Basic** ($\mathcal{B}$) type of facts. Thus this sentence will be classified into all three content types ($\mathcal{B}$, $\mathcal{C}$, $\mathcal{A}$). Whereas the second sentence (***It was the second victory ...***) will be classified as inter-event type ($\mathcal{A}$) as it only contains information from multiple games (also see Figure 2).

It is also possible to extend the Inter-Event category into Basic and Complex categories. But, it is left for the future work mainly because of two reasons: first, both inter-event basic and inter-event complex will pose similar challenge to a generation system, which is, having access to data from other events during run-time; and second, the occurrence of inter-event basic facts will be rare, as often records from multiple events is used to derive

a new fact rather than being used as a single fact.

2.3 Building Content Type Classifier

With the Content Type typology defined for a D2T task, the next step is to use this typology in generating the dataset profile. However, the datasets can be large with sometimes more than 20k event summaries, each containing 15-20 sentences. Thus manually annotating these summaries to generate the dataset profile will be difficult. In this work, a **Multi-Label Classifier** is used for generating the dataset profile by classifying the sentences of event summaries into their content types. More specifically, a multi-label classifier function f is learned to map a sentence $\mathcal{T}$ to its content types y as:

$$y_j = f(\mathcal{T}_j)$$

where $y_j \subseteq \mathcal{Y}$, and $\mathcal{Y} = \{\mathcal{B}, \mathcal{C}, \mathcal{A}\}$

More detail on building the content type classifier is given in Appendix A. A pictorial representation of the process of creating a dataset's profile is shown in Figure 3.

3 Experimental Setup

The experiments are performed in three phases. In first phase, the aim is to understand the characteristics of human authored summaries, for which, dataset profiles are generated based on human authored summaries using the proposed content type typology (Section 4.1). The second phase aims to demonstrate the challenge state-of-the-art generation systems face in attempting to generate complex (inter-event and intra-event complex) content (Section 4.2). This is evaluated by comparing the errors made by the generation systems for each content types. Finally, the proposed methodology is used to understand the concept-drift issue in a problem domain which can help in building better systems capable of handling such domain-specific issues (Section 4.3).

3.1 Datasets

Four datasets from different domains are used for profiling: **MLB** (Puduppully et al., 2019b) and **SportSett** (Thomson et al., 2020a) datasets from sports domain; **SumTime** (Sripada et al., 2003) dataset from weather forecasting; and **Obituary** (Upadhyay et al., 2020) from Obituary generation domain. These datasets contain events' structured data on the input side parallelly aligned with human-authored textual summaries of each event.

- **MLB** dataset contains stats from MLB games aligned with their summaries written by human authors. Each sample in the dataset contains the box score and play-by-play record of the of the game on the input side, which is aligned with a textual summary of around 20 sentences long. In this dataset, games are considered as the events; players, teams and the plays as entities of the event; and different stat-types as the features of an entity. The dataset is split for train/valid/test sets, containing 22821/1739/1744 samples each.

- **SportSett** dataset contains box- & line- scores from NBA (basketball) games aligned with human-written summaries describing the respective game. The games are from season 2014 to 2018, out of which 2014, 2015 & 2016 are used as train split, 2017 for valid split, and 2018 for test split. The number of samples in train/valid/test sets is 4745/1228/1229 respectively. Similar to MLB, each game in the dataset is an event; the players teams from the game are entities; and stat-types are the features.

- **SumTime** dataset contains human written weather forecasts for a day written for oil and gas offshore engineers in Aberdeen, UK. The forecasts are usually written from two types of NWP data: wave data; and mmo data (please refer to Sripada et al. (2003) for a detailed discussion on the data organisation). The representation of SumTime on different dimensions is as follows: each sample in the dataset covers a forecast during 12-hours time-period (AM forecast or PM forecast). The time period for which the forecast is written is considered an event; the different entities are the elements described in the forecast such as wind or wave; and the hours of the day for which the readings are taken for those elements are the features. The total number of samples in train/valid/test sets are 793/99/100.

- **Obituary** dataset contains a sample of 850 obituaries aligned with their personal information. In this dataset, an Obituary (or a death) is the event; where the deceased person is

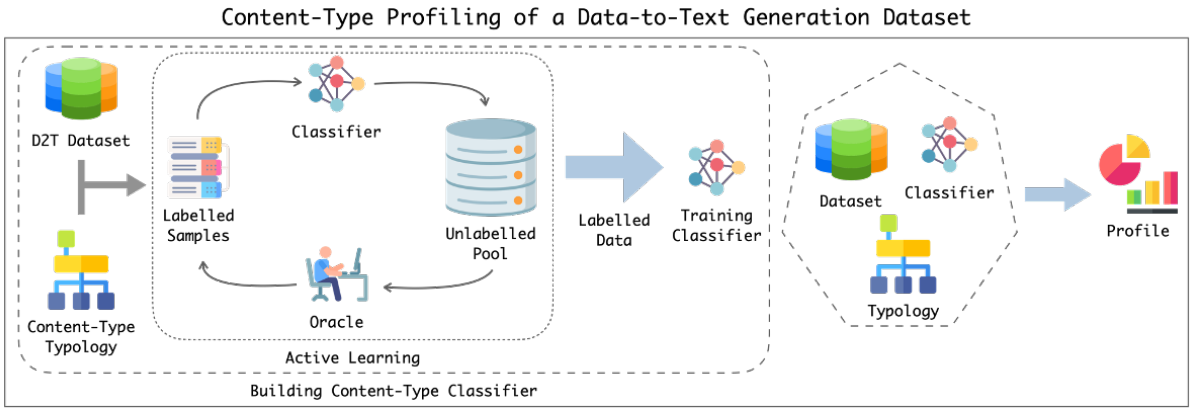

Figure 3: Generating the Content Type Profile of a dataset

the entity; and information related to their personal life and funeral are the different features. The dataset contains 800/20/20 obituaries in train/valid/test set respectively.

3.2 Generation Systems

Phase two of the experiments analyses the ability of state-of-the-art generation systems in producing different types of content. For this, several state-of-the-art generation systems are used to produce summaries on the held-out test-set of datasets mentioned above. For MLB and SportSett, two benchmark neural systems from literature are used: first, the macro-planning model (**Plan**) from Puduppully and Lapata (2021); and second, the entity-based model (**Ent**) from Puduppully et al. (2019b) are used on both datasets. In addition, the hierarchical transformer model (**Hir**) from Rebuffel et al. (2020) is used as a third model for SportSett. For Obituary and SumTime, we are not aware of any existing neural benchmarks, therefore we develop our own generation systems by fine-tuing T5-base (**T5**) from Raffel et al. (2020), BART-base (**BART**) from Lewis et al. (2020), and Pegasus (**Peg**) from Zhang et al. (2020a) on each dataset respectively. Different metric evaluation scores of these developed systems are shown in Appendix B.

3.3 Generation Systems' Accuracy Evaluation

The generation systems' ability of generating complex content is evaluated by measuring the accuracy error-rate of generation systems within each content type category. We calculate the **error rate** of generation systems by manually annotating 10 randomly selected summaries from each system following a pre-established gold standard annotation scheme of D2T systems evaluation developed by Thomson and Reiter (2020). Within each summary, all the generated claims (whether correct or incorrect) within each category are identified and then the error rate is calculated as the ratio of *total incorrect claims to total claims generated*. The error-rate ratio of each content type category is calculated separately. The length of summaries vary from 15-20 sentences each summary in MLB and SportSett to 4-5 sentences in SumTime and Obituary. The annotations are done by the authors themselves and are available on the GitHub repository. The distribution of sentences across different content types from all the evaluated system generated summaries is shown in Table 1.

4 Results and Discussions

In this section, we use the Content Type classifiers build for different datasets using method described in Section 2.2 for generating their profiles. The performance of best classifier for each dataset with its Macro-F1 score and the number of samples used for training is shown in Appendix A. As discussed in the previous section, the dataset profiles will be used for: first, analysing the human-authored summaries from different datasets (Section 4.1); second, analysing the system generated summaries from several state-of-the-art neural generative systems (Section 4.2); and third, characterising the concept-drift issue in SportSett dataset (Section 4.3).

4.1 Analysing Human Authored Summaries

The content type distribution found in human authored summaries from different datasets is shown in Figure 4. On the x-axis, the different content type categories are shown, while the y-axis displays the percentage of sentences belonging to that cate-

Dataset	MLB		SportSett			SumTime			Obituary		
System	Ent	Plan	Ent	Plan	Hir	T5	BART	Peg	T5	BART	Peg
Intra-Event Basic ($\mathcal{B}$)	83	94	71	84	82	20	20	20	35	30	33
Intra-Event Complex ($\mathcal{C}$)	119	193	71	53	84	10	10	10	11	10	10
Inter-Event ($\mathcal{A}$)	55	47	44	33	46	0	0	0	0	0	0

Table 1: Number of sentences from different categories manually annotated for error-rate evaluation

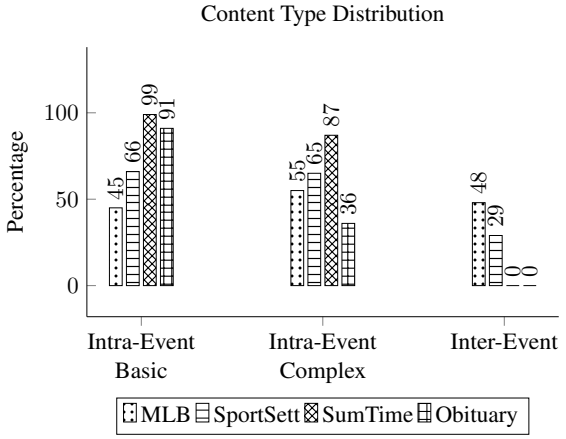

Figure 4: Profile of various datasets based on their human authored summaries

gory. It is noted here that one sentence can be assigned more than one category since a sentence can contain multiple facts of different categories. We can see that MLB has the highest amount of inter-event sentences (48%) with 55% intra-event complex, and 45% intra-event basic sentences. SportSett has 29% inter-event sentences with 65% intra-event complex and 63% intra-event basic sentences. In SumTime, although there are no inter-event sentences, 87% sentences are intra-event complex with 99% of them also being intra-event basic. Obituary has 91% intra-event basic sentences as well as the least percentage of intra-event complex sentences (36%). Obituary also doesn't have any inter-event type sentence in the summaries.

These numbers suggest that humans written summaries do not contain just information copied from input data. Rather, they are full of complex insights derived from multiple records, and possibly multiple events in the stream. This demonstrates that while designing a D2T system, two requirements are necessary: first, in most D2T tasks, an event cannot be considered independent, as it's solution might depend on the data from multiple events in the stream; and second, a system developed for D2T task should be capable of performing complex analytical operations in order to derive implicit information from the given records. Ignoring these requirements will lead to building systems capable of generating only the easier contents and missing the interesting complex insights. Similar issues are observed in other language generation tasks as well where systems try to generate less complex content in order to be safe (Feng et al., 2021; Du and Black, 2019).

4.2 Analysing System Generated Summaries

After analysing the dataset profiles generated using human-authored summaries, we investigate if current state-of-the-art generation systems can produce content with similar profile of human-authored summaries. We show the content type distribution in the system generated along with human reference test-set summaries from different datasets in Figure 5. It can be observed that systems on SumTime and Obituary are able to generate similar amount of intra-event basic & complex sentences as in human written summaries, however, with inaccuracies (will be discussed later in this section). These two datasets do not have any inter-event content in human reference summaries, and thus no such content in system generations as well. MLB and SportSett generated texts have different content type distribution compared to their human reference summaries. If we look at the the generations of Plan system in both datasets, it has the lowest inter-event sentences in MLB and lowest inter-event & intra-event complex sentences in SportSett. This can be attributed to the macro-planning design of the system which restricts the system for producing only information explicitly available in input data. Another pattern can be observed in SportSett, where the system generated summaries have relatively more inter-event sentences than human written summaries, which we explore in the next section.

We also show the error-rate across categories of different systems from different datasets (along with mean and standard deviation of error-rates across systems) in Table 2. The error-rates demon-

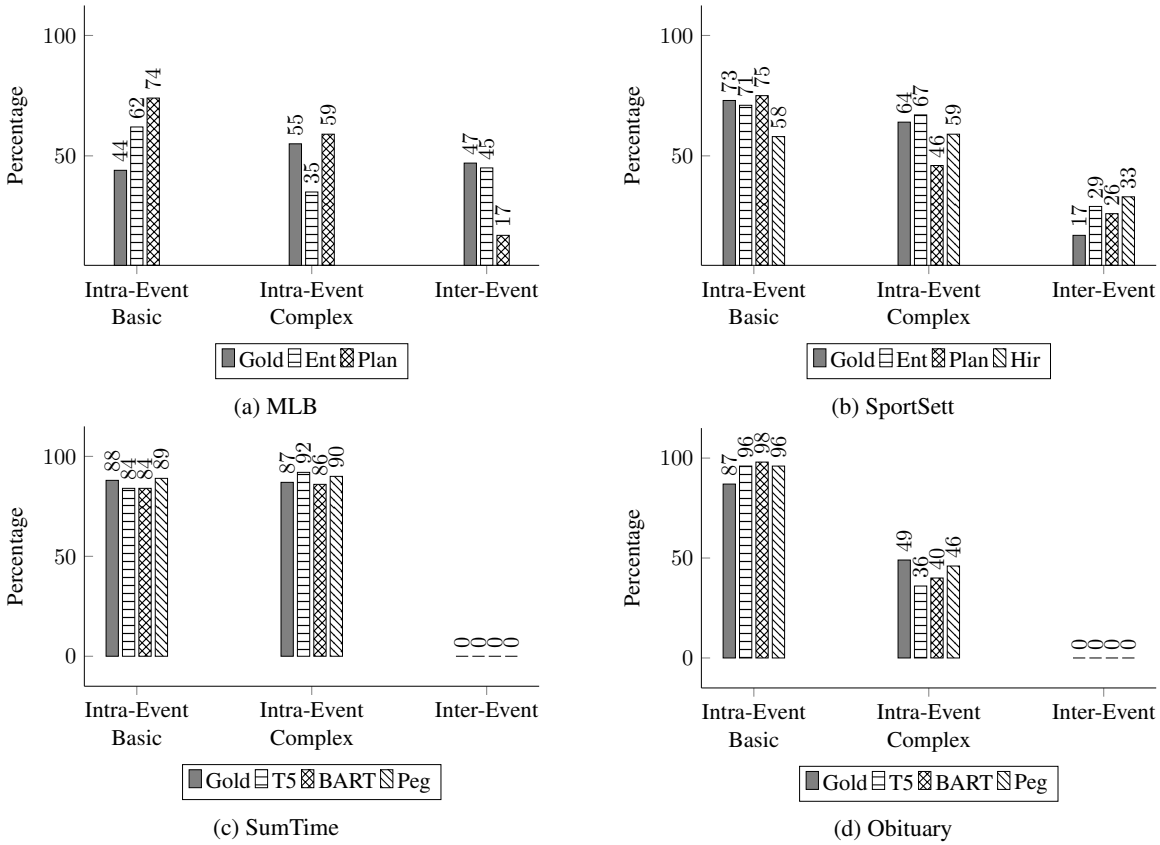

Figure 5: Profile of various datasets based on summaries generated from different generation systems

strate that generation systems struggle to produce the contents of complex types. Generation systems across all datasets have low error-rates in intra-event basic category while higher error-rates in intra-event complex and highest error-rates in the inter-event categories.

Mean error-rate in **intra-event basic** category ranges from 5.5% in Obituary, around 12.5% in MLB and SportSett, to over 38% in SumTime. SumTime has the higest error in this category, which may be due to the lack of training data and a highly domain-specific problem that requires identification of multiple relationships to generate summaries. The systems used for SumTime are not custom designed for the task as with MLB and SportSett, and employs general NLG models, thus having higher error-rate in SumTime than other datasets. Overall, error-rate across all datasets in intra-event basic category is lower than intra-event complex or inter-event (where present) categories.

In the case of **intra-event complex**, almost all the datasets have around 40-50% mean error-rate. This shows systems struggle to learn the domain specific relationships required to derive complex information from the supplied data. Only **Ent** system in MLB and **Plan** system in SportSett have notably lower error-rate. However, these systems are generating comparatively lesser intra-event complex content compared to other systems in order to improve the accuracy (reducing error-rate) by producing less complex content which is easier to generate. All the **inter-event** facts in MLB are incorrect (100% error-rate) while the error-rate in SportSett for inter-event category is also very high. This is not surprising as the input to these systems doesn't take data from across the event stream into account during run-time. SportSett has actually produced some accurate inter-event facts by producing standard phrases learnt from the training data (e.g. "team X has won four out of last five games") that turns out to be correct sometimes. These results clearly demonstrate the difficulty generation systems have in producing content of complex types. Similar observations have been made in literature as well showing the struggle of current state-of-the-art generation systems in producing content that needs to be derived from multiple records or sources (Thomson and Reiter, 2021; Thomson et al., 2020b). The main difference between these works and our proposed content type profiling approach is that in

Systems	Intra-Event Basic ($\mathcal{B}$)	Intra-Event Complex ($\mathcal{C}$)	Inter-Event ($\mathcal{A}$)
		MLB	
Ent	13.98	38.6	100
Plan	10.84	45.27	100
Total	12.4 ± 1.5	41.9 ± 3.3	100 ± 0
		SportSett	
Ent	13.64	61.29	86.96
Plan	6.72	27.54	77.66
Hir	16.72	51.79	91.59
Total	12.3 ± 4.1	46.8 ± 14.2	85.4 ± 5.7
		SumTime	
T5	38.1	50.57	-
BART	39.19	51.85	-
Peg	35.95	48.45	-
Total	37.7 ± 1.3	50.2 ± 1.4	-
		Obituary	
T5	2.74	41.67	-
BART	4.86	52.27	-
Peg	9.09	48.39	-
Total	5.5 ± 2.6	47.4 ± 4.3	-

Table 2: System-wise error-rates of generation systems developed for various datasets categorised by content types (lower is better; ↓)

previous works, the insights are drawn by evaluating the summaries generated from systems whereas with our method many such insights can be drawn proactively before going into system development.

4.3 Concept Drift in SportSett

We further apply our proposed content type profiling methodology to capture the concept drift issue in SportSett dataset. This dataset contains NBA games from season 2014 to 2018 and follows a seasonal partition to generate the train/test/valid splits. Seasons 2014, 2015 and 2016 are used for train set while 2017 and 2018 are used for validation and test sets respectively (please refer to Thomson et al. (2020a) for more details). In Figure 6, the content type distribution of summaries by year is shown. The summaries from earlier years contain greater amount of inter-event sentences while comparatively little in later years. Even with intra-event sentences, there are lesser intra-event basic sentences in summaries from earlier years than later ones, indicating that summaries in the training set are more complex than in the validation and test sets. This discrepancy explains the observed distribution of summaries generated from the different systems as shown in Figure 5b. We can see the system generations have more inter-event

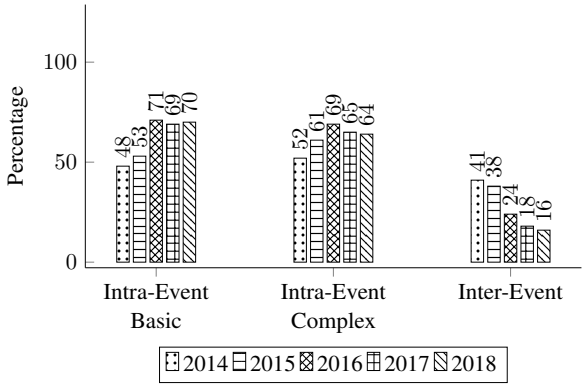

Figure 6: SportSett dataset profile by NBA seasons

sentences because the training data has more inter-event sentences. Concept drift in this D2T problem is captured by our dataset profiling method. This concept-drift can also be explained with the change in authors in different years writing the summaries. In Figure 7, we also show two authors who wrote summaries in different years: 'Auth1' in 2014-15; and 'Auth2' in 2017-18. It is clear that different authors from different years have different distribution which may explain the concept-drift problem in the dataset.

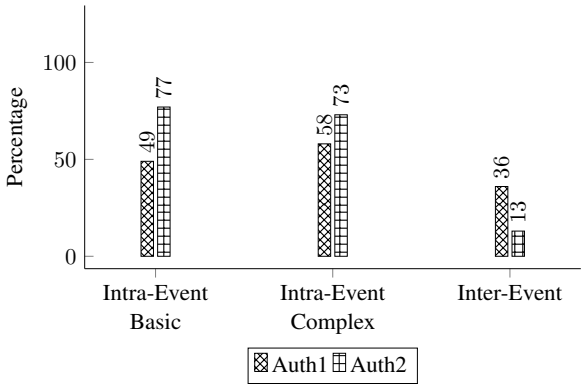

Figure 7: SportSett dataset profile by authors

5 Related Works

Evaluation of texts produced from generation systems is widely used to identify the complexity of a dataset and improve the systems afterwards. There are two main approaches taken for evaluations: automated metrics, such as BLEU (Papineni et al., 2002), ROUGE (Lin, 2004), chrF++ (Popović, 2017), borrowed from machine translation research, or RG (Wiseman et al., 2017), specific to the D2T task; and human evaluations, where users are asked to rate the generations on a Likert scale (Dušek et al., 2020; Chen et al., 2020; Puduppully et al., 2019a). While automated metrics are easier to use, they often correlate poorly with the human evaluation (Reiter, 2018). Human evaluation is considered to be the gold standard for NLG evaluations, however Likert scale based evaluation of single sentences doesn't give a lot of information about the quality of generation. Recent works have proposed to take a more task-oriented evaluation of generated texts that give more insights into the kind of error generation systems make (Thomson and Reiter, 2020, 2021; Kasner et al., 2021; Garneau and Lamontagne, 2021).

There have been few notable works that try to improve the dataset in order to build better generation systems. SportSett dataset by Thomson et al. (2020a) presented an improved resource with better modelling of the dataset to increase the overlap between input data and output text. Gong et al. (2019) and Thomson et al. (2020b) acknowledge the event stream behaviour of D2T tasks by incorporating additional information (both within-event as well as across-event) to improve the quality of generations. In this work, we do not yet try to solve the problem of handling complex or across-event content in summaries. Rather our aim is to profile a dataset through the typology of content types which can be used to identify the complexity of the dataset.

6 Conclusion

In this paper, we presented a typology of different content types in D2T summaries. The proposed typology is used to profile multiple datasets, which captures their characteristics and provide a measure of complexity present in the datasets. Extensive experimentation is performed to demonstrate the challenge facing generation systems in producing complex types of content. We further use the profiling method to identify the concept drift problem in a dataset. Through this work, we argue that a dataset's content type profile can help us define system requirements for building better generation systems.

In future, we plan to employ insights gained from this work in building better D2T generation systems capable of producing accurate complex content for event summaries. This will require a system to be able to: understand the domain-specific rules, for deriving implicit information from the input data: and able to operate in huge search space, to select important content from vast possibility of across-event information. The dataset profiling method will help in identifying any information gap between input data and output summary and characterise the domain-specific rules in different categories based on their information source.

Acknowledgements

We thank the anonymous reviewers for their valuable comments. We also like to thank the members of AI & Reasoning Reading Group at Robert Gordon University, Aberdeen and CLAN Reading Group at University of Aberdeen for their valuable feedback on the work. We also thank Anjana Wijekoon for helpful discussions and proofreading the paper before submission.

References

Satanjeev Banerjee and Alon Lavie. 2005. METEOR: An automatic metric for MT evaluation with improved correlation with human judgments. In *Proceedings of the ACL Workshop on Intrinsic and Extrinsic Evaluation Measures for Machine Translation and/or Summarization*, pages 65–72, Ann Arbor, Michigan. Association for Computational Linguistics.

Wenhu Chen, Yu Su, Xifeng Yan, and William Yang Wang. 2020. KGPT: Knowledge-grounded pre-training for data-to-text generation. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 8635–8648, Online. Association for Computational Linguistics.

Emilie Colin, Claire Gardent, Yassine M'rabet, Shashi Narayan, and Laura Perez-Beltrachini. 2016. The WebNLG challenge: Generating text from DBPedia data. In *Proceedings of the 9th International Natural Language Generation conference*, pages 163–167, Edinburgh, UK. Association for Computational Linguistics.

Harm de Vries, Dzmitry Bahdanau, and Christopher D. Manning. 2020. Towards ecologically valid research on language user interfaces. *CoRR*, abs/2007.14435.

Wenchao Du and Alan W Black. 2019. Boosting dialog response generation. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 38–43, Florence, Italy. Association for Computational Linguistics.

Ondřej Dušek, Jekaterina Novikova, and Verena Rieser. 2020. Evaluating the State-of-the-Art of End-to-End Natural Language Generation: The E2E NLG Challenge. *Computer Speech & Language*, 59:123–156.

Steven Y. Feng, Jessica Huynh, Chaitanya Prasad Narisetty, Eduard Hovy, and Varun Gangal. 2021. SAPPHIRE: Approaches for enhanced concept-to-text generation. In *Proceedings of the 14th International Conference on Natural Language Generation*, pages 212–225, Aberdeen, Scotland, UK. Association for Computational Linguistics.

Nicolas Garneau and Luc Lamontagne. 2021. Shared task in evaluating accuracy: Leveraging pre-annotations in the validation process. In *Proceedings of the 14th International Conference on Natural Language Generation*, pages 266–270, Aberdeen, Scotland, UK. Association for Computational Linguistics.

Albert Gatt and Emiel Krahmer. 2018. Survey of the state of the art in natural language generation: Core tasks, applications and evaluation. *Journal of Artificial Intelligence Research*, 61:65–170.

Timnit Gebru, Jamie Morgenstern, Briana Vecchione, Jennifer Wortman Vaughan, Hanna Wallach, Hal Daumé Iii, and Kate Crawford. 2021. Datasheets for datasets. *Communications of the ACM*, 64(12):86–92.

Heng Gong, Xiaocheng Feng, Bing Qin, and Ting Liu. 2019. Table-to-text generation with effective hierarchical encoder on three dimensions (row, column and time). In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 3143–3152, Hong Kong, China. Association for Computational Linguistics.

Zdeněk Kasner, Simon Mille, and Ondřej Dušek. 2021. Text-in-context: Token-level error detection for table-to-text generation. In *Proceedings of the 14th International Conference on Natural Language Generation*, pages 259–265, Aberdeen, Scotland, UK. Association for Computational Linguistics.

Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Veselin Stoyanov, and Luke Zettlemoyer. 2020. BART: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7871–7880, Online. Association for Computational Linguistics.

Chin-Yew Lin. 2004. ROUGE: A package for automatic evaluation of summaries. In *Text Summarization Branches Out*, pages 74–81, Barcelona, Spain. Association for Computational Linguistics.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Roberta: A robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692*.

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. Bleu: a method for automatic evaluation of machine translation. In *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics*, pages 311–318, Philadelphia, Pennsylvania, USA. Association for Computational Linguistics.

Maja Popović. 2017. chrF++: words helping character n-grams. In *Proceedings of the Second Conference on Machine Translation*, pages 612–618, Copenhagen, Denmark. Association for Computational Linguistics.

Ratish Puduppully, Li Dong, and Mirella Lapata. 2019a. Data-to-text generation with content selection and planning. In *The Thirty-Third AAAI Conference on Artificial Intelligence, AAAI 2019, The Thirty-First Innovative Applications of Artificial Intelligence Conference, IAAI 2019, The Ninth AAAI Symposium on Educational Advances in Artificial Intelligence, EAAI 2019, Honolulu, Hawaii, USA, January 27 - February 1, 2019*, pages 6908–6915. AAAI Press.

Ratish Puduppully, Li Dong, and Mirella Lapata. 2019b. Data-to-text generation with entity modeling. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 2023–2035, Florence, Italy. Association for Computational Linguistics.

Ratish Puduppully and Mirella Lapata. 2021. Data-to-text generation with macro planning. *Transactions of the Association for Computational Linguistics*, 9:510–527.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J Liu. 2020. Exploring the limits of transfer learning with a unified text-to-text transformer. *Journal of Machine Learning Research*, 21:1–67.

Clément Rebuffel, Laure Soulier, Geoffrey Scoutheeten, and Patrick Gallinari. 2020. A hierarchical model for data-to-text generation. In *European Conference on Information Retrieval*, pages 65–80. Springer.

Ehud Reiter. 2007. An architecture for data-to-text systems. In *Proceedings of the Eleventh European Workshop on Natural Language Generation (ENLG 07)*, pages 97–104, Saarbrücken, Germany. DFKI GmbH.

Ehud Reiter. 2018. A structured review of the validity of BLEU. *Computational Linguistics*, 44(3):393–401.

Somayajulu Sripada, Ehud Reiter, and Ian Davy. 2003. Sumtime-mousam: Configurable marine weather forecast generator. *Expert Update*, 6(3):4–10.

Craig Thomson and Ehud Reiter. 2020. A gold standard methodology for evaluating accuracy in data-to-text systems. In *Proceedings of the 13th International Conference on Natural Language Generation*, pages 158–168, Dublin, Ireland. Association for Computational Linguistics.

Craig Thomson and Ehud Reiter. 2021. Generation challenges: Results of the accuracy evaluation shared task. In *Proceedings of the 14th International Conference on Natural Language Generation*, pages 240–248, Aberdeen, Scotland, UK. Association for Computational Linguistics.

Craig Thomson, Ehud Reiter, and Somayajulu Sripada. 2020a. SportSett:basketball - a robust and maintainable data-set for natural language generation. In *Proceedings of the Workshop on Intelligent Information Processing and Natural Language Generation*, pages 32–40, Santiago de Compostela, Spain. Association for Computational Lingustics.

Craig Thomson, Zhijie Zhao, and Somayajulu Sripada. 2020b. Studying the impact of filling information gaps on the output quality of neural data-to-text. In *Proceedings of the 13th International Conference on Natural Language Generation*, pages 35–40, Dublin, Ireland. Association for Computational Linguistics.

Ashish Upadhyay, Stewart Massie, and Sean Clogher. 2020. Case-based approach to automated natural language generation for obituaries. In *International Conference on Case-Based Reasoning*, pages 279–294. Springer.

Sam Wiseman, Stuart Shieber, and Alexander Rush. 2017. Challenges in data-to-document generation. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 2253–2263, Copenhagen, Denmark. Association for Computational Linguistics.

Jingqing Zhang, Yao Zhao, Mohammad Saleh, and Peter Liu. 2020a. PEGASUS: Pre-training with extracted gap-sentences for abstractive summarization. In *Proceedings of the 37th International Conference on Machine Learning*, volume 119 of *Proceedings of Machine Learning Research*, pages 11328–11339. PMLR.

Tianyi Zhang, Varsha Kishore, Felix Wu, Kilian Q. Weinberger, and Yoav Artzi. 2020b. Bertscore: Evaluating text generation with BERT. In *8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, April 26-30, 2020*. OpenReview.net.

A Building a Content Type Classifier for Each Dataset

To build the content type classifier for a dataset, some training (ranging from 200 to 600 samples) and testing (250 samples) samples are created by manually annotating the sentences from the train and valid set of that D2T dataset's summaries. Several learning methods (Random Forest, SVM, Logistic Regression) with several features (TF, TF-IDF, dense embeddings from Roberta model from Liu et al. 2019 (RobEmb)) along with a fine-tuned Roberta model are used for building multiple classifiers. A query-by-committee (active learning) approach is used to build the clssifiers' training dataset. The classifier with best Macro-F1 score on held-out test-set of a given dataset is used as the Content Type classifier of that dataset. The performance of Content Type classifiers used for each dataset is shown in Table 3. The table shows the Macro-F1 score of each classifier and the number of samples used for training the classifier. All the trained classifiers with their training and testing samples is shared on the GitHub repo.

B Evaluation Scores of Neural Generation Systems Developed for Different Datasets

Table 4 shows the automated metric scores of different neural systems build for each dataset. The following metrics are used: BLEU (Papineni et al., 2002); ROUGE-L (Lin, 2004); METEOR (Banerjee and Lavie, 2005); chrF++ (Popović, 2017); and BERT-Score (Zhang et al., 2020b). For all metrics, higher score is better (↑).

We also show the results from Extractive Evaluation (EE) metrics (Wiseman et al., 2017) on systems developed for MLB and SportSett datasets in Table 5. These metrics are: Relation Generation (RG); Content Selection (CS); and Content Ordering (CO). The EE scores for systems build on MLB dataset are taken from Puduppully and Lapata (2021) while the EE scores for systems build on SportSett dataset are calculated using Information Extraction system described by Thomson et al. (2020b).

Dataset	MLB	SportSett	SumTime	Obituary
Classifier	RobFT	RobFT	SVM w/ RobEmb	SVM w/ RobEmb
# Train Samples	600	600	200	200
Macro F1	85.64	91.0	98.66	98.46

Table 3: Content Type classifiers performance on various datasets

Systems	BLEU	ROUGE-L	METEOR	chrF++	BERTScore
MLB					
Ent	11.51	22.08	27	32	85.03
Plan	13.99	21.71	32	39	84.6
SportSett					
Ent	18.19	26.19	33	42	86.95
Plan	17.6	26.29	32	40	86.45
Hir	12.18	22.65	33	41	85.74
SumTime					
T5	24.67	52.92	38	47	89.66
BART	18.77	47.61	33	46	88.82
Peg	23.54	51.06	39	48	89.68
Obituary					
T5	47.3	65.38	64	67	94.04
BART	50.88	65.99	66	69	94.29
Peg	45.03	66.4	61	65	93.73

Table 4: Automated metric scores of different systems developed for various datasets (↑, higher is better)

Systems	RG	CS-Precision	CS-Recall	CO
MLB				
Ent	81.1	40.9	49.5	20.7
Plan	94.4	40.8	54.9	21.8
SportSett				
Ent	72.77	45.35	38.12	19.13
Plan	86.48	53.95	33.09	14.84
Hir	73.77	45.42	30.15	10.59

Table 5: Extractive Evaluation results of systems developed for MLB and SportSett dataset (↑, higher is better)

CoLo: A Contrastive Learning based Re-ranking Framework for One-Stage Summarization

Chenxin An[1], **Ming Zhong**[2], **Zhiyong Wu**[3] ,**Qin Zhu**[1], **Xuanjing Huang**[1], **Xipeng Qiu**[1*]
[1]School of Computer Science, Fudan University
[2] University of Illinois at Urbana-Champaign
[3] Shanghai AI Lab
{cxan20, qzhu18, xjhuang, xpqiu}@fudan.edu.cn
mingz5@illinois.edu, wuzhiyong@pjlab.org.cn

Abstract

Traditional training paradigms for extractive and abstractive summarization systems always only use token-level or sentence-level training objectives. However, the output summary is always evaluated from summary-level which leads to the inconsistency in training and evaluation. In this paper, we propose a **Co**ntrastive **L**earning based re-ranking framework for **o**ne-stage summarization called CoLo. By modeling a contrastive objective, we show that the summarization model is able to directly generate summaries according to the summary-level score without additional modules and parameters. Extensive experiments demonstrate that CoLo boosts the extractive and abstractive results of one-stage systems on CNN/DailyMail benchmark to 44.58 and 46.33 ROUGE-1 score while preserving the parameter efficiency and inference efficiency. Compared with state-of-the-art multi-stage systems, we save more than 100 GPU training hours and obtaining $3\times \sim 8\times$ speed-up ratio during inference while maintaining comparable results.

1 Introduction

In general, there are two main paradigms to do text summarization: *abstractive* (Rush et al., 2015; Nallapati et al., 2016; Gehrmann et al., 2018) and *extractive* (Cheng and Lapata, 2016; Narayan et al., 2018b; Zhong et al., 2019, 2022) methods.

For extractive summarization, previous studies (Nallapati et al., 2017; Liu and Lapata, 2019) formulate it as a *sentence-level* sequence labeling task. However, there is an inherent gap between the *sentence-level* scoring and the *summary-level* evaluation (Zhong et al., 2020).This means that some high-scoring sentences may share the same meaning, making them not a qualified summary when combined. Similarly, the previous training paradigm for abstractive summarization models can be viewed as a *token-level* scoring process upon the decoder of sequence-to-sequence model. There also exists the issue of exposure bias (Bengio et al., 2015; Paulus et al., 2017) in the teacher-forcing framework leading to the error accumulation during auto-regressive decoding. Therefore, previous frameworks for both extractive and abstractive methods did not perform *summary-level* optimization.

To tackle this problem, state-of-the-art summarization systems (Zhong et al., 2020; Liu and Liu, 2021) are enhanced with an additional module (called re-ranker) and follow a two-stage paradigm. They first train a summarizer to model the conditional distribution $p(Y|X)$ where X is the document and Y is the output summary. Then the re-ranker is trained to re-score candidates sampled from the pre-trained summarizer in the second stage. However, this paradigm trades efficiency for accuracy, the auxiliary re-ranking greatly harms the inference efficiency especially for the highly efficient extractive systems. Experimentally, the decoding speed of two-stage re-ranking models is only ~7.0 samples/s while removing the re-ranker module will greatly boost the decoding speed to **~42.0** samples/s[1]. This makes two-stage summarization systems may be unacceptable in real-world scenarios that require timely feedback.

The limitations of the existing work motivate us to build a one-stage summarization system that can 1) replace previous naive sentence/token-level score with a summary-level score and 2) do not sacrifice the parameter and inference efficiency. In this paper, we propose a **Co**ntrastive **L**earning based re-ranking framework for **o**ne-stage summarization called CoLo for both extractive and abstractive approach. Contrastive learning has been explored in summarization (Sun and Li, 2021; An et al., 2021b) and generation (Lee et al., 2020; An et al., 2022).

*Corresponding author.

[1]We run these two models on the test set of CNN/DailyMail using single GeForce GTX TITAN XP GPU for 3 times and report the average speed.

Proceedings of the 29th International Conference on Computational Linguistics, pages 5783–5793
October 12–17, 2022.

CoLo uses a contrastive re-ranking training objective. We first present a novel sampling method that can be equipped to any one-stage summarization systems so that it can re-score candidates without the second stage. The existing two-stage models use **offline sampling** to preprocess samples for training of re-ranker where candidate samples are drawn from a fixed model distribution. This is a huge obstacle to turning *summarize-then-rerank* two-stage framework into an efficient end-to-end model. To solve this issue, we propose an **online sampling** approach. Concretely, instead of sampling from a fixed distribution, we draw positive and negative samples from a dynamic distribution of model outputs during training, which ultimately eliminates the requirement for additional modules in the overall framework. We then introduce a summary-level optimization strategy in addition to the traditional sentence-level (for extractive systems) or token-level loss (for abstractive systems). As a result, as a one-stage model, CoLo achieves comparable performance to two-stage systems, and greatly improves decoding speed to meet the needs of real-world applications.

We summarize our contributions as follows:

- We are the first to propose a one-stage re-ranking framework CoLo for both extractive and abstractive summarization systems.
- Results on the popular CNN/DailyMail benchmark show that both the extractive and abstractive versions of CoLo outperform previous state-of-the-art one-stage systems by a large margin. Compared to the two-stage systems, CoLo achieves comparable performance without additional pre-trained model. More importantly, CoLo do not sacrifice inference speed and thus can be more widely used in real-world scenarios.

2 Background

2.1 Preliminary about Two-Stage Systems

Two-stage paradigms (Zhong et al., 2020; Liu and Liu, 2021) improve summarization quality by re-ranking and selecting a candidate from a given set of candidates. MatchSum (Zhong et al., 2020) forms a contrastive learning based re-ranking framework where they first generate a set of candidates summaries by a extractive summarization model and then feed them to a re-ranker. The re-ranker is trained to optimize a summary-level score and it can evaluate the candidate summaries holistically. SimCLS (Liu and Liu, 2021) is the abstractive version which replaces the extractive summarizer in Zhong et al. (2020) with a abstractive summarizer.

The training objective for summarization models is to estimate a conditional probability distribution $p(Y|X)$, where X is the document and Y is the output summary. Given a summarization model $\mathcal{M}$ that has already tuned under the conventional framework with loss function $\mathcal{L}_{sum}$ where $\mathcal{L}_{sum}$ could be binary cross entropy loss (BCELoss) or negative log likelihood loss (NLLLoss). The two-stage systems should first use a sampling algorithm e.g. beam search to sample a candidate set $\mathcal{C} = \{C_1, C_2, \ldots, C_m\}$ of size m from the fixed model distribution $C_i \sim p_{\mathcal{M}}(Y|X)$. Candidates in $\mathcal{C}$ are sort by their ROUGE score in descending order. Then the they further train a separate re-ranker,e.g., BERT , with a contrastive-style ranking loss $\mathcal{L}_{rank}$ to select the the best candidate from $\mathcal{C}$ as the final output. The ranking loss used in the best re-ranking system for summarization is the triplet margin loss (Kingma and Ba, 2014). For a candidate pair (C_i, C_j) where $i < j$, if C_i has higher ROUGE score and it will be treated as the positive sample:

$$\mathcal{L}_{i,j} = max\{0, \cos(\mathbf{z}_X, \mathbf{z}_{C_i}) - \cos(\mathbf{z}_X, \mathbf{z}_{C_j}) + \rho\}, \quad (1)$$

where $\mathbf{z}_X$, $\mathbf{z}_{C_i}$, $\mathbf{z}_{C_j}$ are the vector feature representation of X, C_i, C_j output by the re-ranker, and ρ is the margin value. The final ranking loss is obtained by summing up all pairs: $\mathcal{L}_{rank} = \sum_j \sum_{i<j} \mathcal{L}_{i,j}$. The ranking loss ensures that candidates with higher ROUGE score is closer to the document in the embedding space.

2.2 A Comparison between Two-Stage Systems and CoLo

Figure 1 illustrates the difference between the architecture of two-stage systems and CoLo. Although MatchSum and SimCLS significantly outperform all one-stage models, they mainly suffer from three drawbacks which strongly emphasize the necessity of designing an one-stage model:

(1) Training/inference inefficiency. Building the training set of the re-ranker and the second training stage consumes large amounts of GPU and CPU time (see details in Section 5.3). Moreover, the need of re-feeding generation results to another module also requires unaffordable computational

resources.

(2) Coupling between the summarizer and re-ranker. Each improvement to one of these modules requires simultaneous updating or retraining of another module, which limits the use of such systems in the real world. For example, to try a larger candidate set or a different decoding method, we have to prepare the training set again for the second stage. In addition, how to tune the hyperparameters to be optimal in both modules at the same time is another tricky issue. Compared with two-stage systems, our one-stage system has a simple and clean implementation.

(3) Two-stage systems also face difficulties in long document summarization, because the input length of the re-ranker will drastically increase as the length of candidates increasing (see detailed analysis in Appendix A). Correspondingly, CoLo is not easily affected by length variance.

3 Method

3.1 A Naive One-Stage Re-ranking Model

The goal of one-stage re-ranking systems is to enable both training and inference to score candidate summaries holistically without requiring a second stage of computation by a separate model. Ideally, an one-stage summarization model should both function as a summarizer and a re-ranker. A straightforward solution is multi-task learning. The naive training pipeline can be formulated as follows: (i) tuning $\mathcal{M}$ with $\mathcal{L}_{sum}$. (ii) Getting positive and negative samples from $p_{\mathcal{M}}(Y|X)$ via offline sampling for each datapoint X in the training set. (iii) Building the ranking loss with these candidates and further tuning $\mathcal{M}$ with $\mathcal{L}_{rank} + \mathcal{L}_{sum}$. However, in practice, such training method is always suboptimal compared to the state-of-the-art two-stage models. We denote the model after multi-task learning as $\mathcal{M}'$. There is a serious *generalization error* in the naive methods: via multi-task learning, $\mathcal{M}'$ is only able to rank candidates drawn from the original model distribution $p_{\mathcal{M}}(Y|X)$ but not candidates from the new distribution $p_{\mathcal{M}'}(Y|X)$. This error makes the naive approach unable to directly output a good summary in sequence-level generated by itself.

3.2 Our approach: CoLo

The first step of CoLo is also to train the summarization model with $\mathcal{L}_{sum}$ like the naive approach.

In CoLo, we discard using positive-negative samples that from a fixed model distribution, instead, we sample these candidates from a constantly shifting model distribution during multi-task learning. By doing so, we can mitigate the above mentioned generalization error as much as possible because candidates are dynamically changing with the parameters of the model distribution $p_{\mathcal{M}}(Y|X)$ updated by gradient descent. To implement this process, at each training step, we sample the newest candidates along with their feature presentations from the summarization model and calculate the ranking loss. We will give a detailed description about how we performing the online sampling process on mainstream extractive and abstractive summarization models in the following parts.

Online Sampling for Extractive Model The task of extractive summarization is to assign a label $y_i \in \{0, 1\}$ for each sentence $sent_i$ from the source document $X = (sent_1, sent_2, \ldots, sent_n)$ consisting of n sentences. Figure 2 gives an example of our one-stage extractive summarization model. Extractive candidates can be viewed as a subset of sentences from the document. In this figure, we sample $sent_1, sent_2$ to form the first candidate $C_1 = \{sent_1, sent_2\}$, and C_2 is consisting of $\{sent_2, sent_3\}$. After constructing these candidates, the next step is to represent them in the embedding space. In our one-stage model, we employ a heuristic way to obtain the feature presentations of candidates: pooling results of the sentence embedding from the extractive model. Concretely, we denote the sentence embedding for the i-th sentence as $\mathbf{h}_i$. The hidden representation of a candidate is created by pooling the sentence representations belong to it. For example $\mathbf{z}_{C_1}$ is the average pooling result of $\mathbf{h}_1$ and $\mathbf{h}_2$. Suppose C_2 has higher ROUGE score than C_1, then C_2 is treated as a positive sample and C_1 is treated as a negative sample for this pair. Finally, the whole system is trained by the sum of $\mathcal{L}_{rank}$ and $\mathcal{L}_{sum}$.

Sampling informative candidates is essential in re-ranking systems. The first step of the sampling method is to determine $\mathcal{N}$ which represents the number of candidate sentences. $\mathcal{N}$ is set depending on the number of summary sentences of downstream datasets. Take CNN/DailyMail as an example, we set $\mathcal{N}$ to $\{2, 3\}$ because most gold summaries consist of 2$\sim$3 sentences. At each training step, we iterate over $\mathcal{N}$ by combination and form m different candidates $\mathcal{C} = \{C_1, C_2, \ldots, C_m\}$. m is

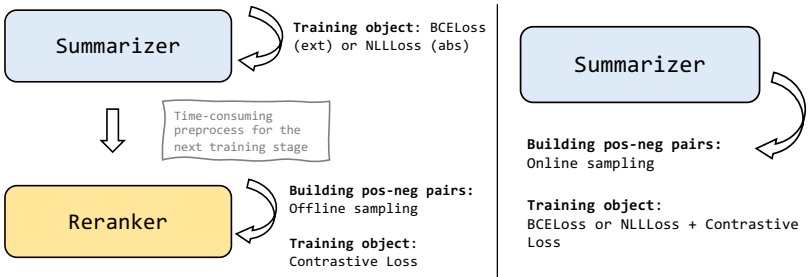

(a) Two-stage models: MatchSum and SimCLS (b) CoLo (this work)

Figure 1: A comparison between two-stage models and CoLo. The two-stage models including two training stages and a time-consuming preprocess while CoLo is trained in an end-to-end fashion. (GPU and CPU hours cost in each stage are shown in Table 6). Two-stage models use offline sampling to build positive-negative pairs while CoLo builds positive-negative pairs with online sampling where we directly get theses pairs from a changing model distribution.

equal to $\sum_i C_n^{num_i}$ where num_i is the i-th element in $\mathcal{N}$ and n is number of sentences of the document. For CNN/DailyMail whose $\mathcal{N}$ is set to $\{2, 3\}$, we can sample $C_n^2 + C_n^3$ different candidates.

However, in practice, we always face the combination explosion problem when the number of sentences n grows larger. The two-stage system (Zhong et al., 2020) pre-trained an extractive model to clip the origin number of sentences to an acceptable size. Notice that our extractive summarizer is also supervised with the BCELoss, so that we can clip the sampling space to n' (a hyperparameter) with the output distribution over the sentences at each training step. Then the total size of the final candidate set decreases to $m' = \sum_i C_{n'}^{num_i}$. For CNN/DailyMail, n' is set to 5, and we can get $C_5^2 + C_5^3 = 20$ different extractive candidates. Details about the setting of $\mathcal{N}$ and n' can be found in Table 1 in Appendix.

Notably, the offline sampling needs to feed each candidate into the pre-trained encoder. In real-life setting, when summarizing some long documents, the number of sentences in the input document and output summary will increase significantly. It will bring a polynomial level increase to the computation and GPU overhead of the two-stage model. But our one-stage system with online sampling is robust to the length variance.

Inference Stage of Extractive Model Since we have modeled a summary-level score during training, it is easy to directly generate summaries according to the summary-level semantic score. Concretely, given a candidate set $\mathcal{C}$ built by the combination strategy, we calculate the cosine similarity between each candidate presentation $\mathbf{z}_{C_i}$ and the document representation $\mathbf{z}_X$:

$$\hat{C} = \max_{C_i \in \mathcal{C}} cos(\mathbf{z}_X, \mathbf{z}_{C_i}). \quad (2)$$

The final output is the candidate with highest cosine similarity score.

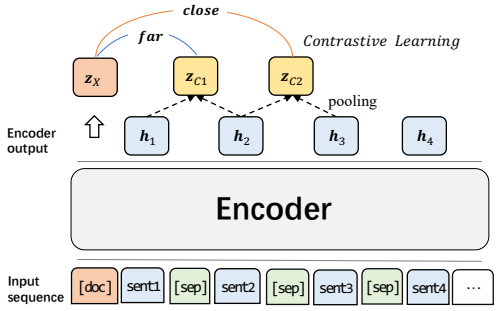

Figure 2: Architecture of our extractive model. Input sequence: The '[doc]' token is used to get vector representation $\mathbf{z}_X$ of the document X, '[sep]' is used as separator for sentences. We omit the classifier and the BCELoss. $\mathbf{h}_i$ is the sentence embedding the i-th sentence in X. $\mathbf{z}_{C_i}$ means the feature representation of the i-th candidate.

Online Sampling for Abstractive Model Our method can also be easily adapted in abstractive summarization. Selecting a generated summary maximum a posteriori (MAP) usually result in poor performance (Stahlberg and Byrne, 2019), thus most state-of-the-art generation model usually use the beam search algorithm at inference stage. The online sampling for the abstractive version is much simpler than the extractive version. We use beam search as sampling algorithm and get the feature representations from the encoder/decoder output. We denote the encoder output of source document

	CNN/DM	Reddit	XSum	SSN	PubMed
n'	5	5	5	8	8
$\mathcal{N}$	2,3	1,2	1,2	6	6,7
$\|\mathcal{C}\|$	20	15	15	28	36

Table 1: candidate size $|\mathcal{C}|$ of each datasets (extractive). $|n'|$ is the clipped candidate size, $\mathcal{N}$ is a set containing all number of possible sentence.

X as H_{enc} and the decoder hidden states of the target summary as H_{dec}. We get the document representation from the encoder output of the 0-th token $\mathbf{z}_X = H^0_{enc}$. The feature representation of the i-th candidate C_i with length = $|C_i|$ is derived from the last step of the decoder output $\mathbf{z}_{C_i} = H^{|C_i|-1}_{dec}$. Hidden states of other steps can not represent the entire sequence because of the sequence mask in transformer decoder. finally we formulate the ranking loss following Eq. 1.

Inference Stage of Abstractive Model The inference stage of our abstractive version is similar to the extractive version. We save the feature representation of the document and each beam during beam search. The final output is determind by the cosine distance between $\mathbf{z}_X$ and $\mathbf{z}_{C_i}$.

4 Experimental Setup

4.1 Datasets

We conduct experiments on five mainstream datasets to evaluate the effectiveness of our approach.

CNN/DailyMail (Hermann et al., 2015) is a classic benchmark which contains articles from the CNN/Daily Mail newspapers. We use the cased version from datasets[2]

XSum (Narayan et al., 2018a) is a one-sentence summary dataset from BBC News. Gold summaries are professionally written by the authors of documents.

Reddit (Kim et al., 2019) is collected from social media platform and we use the TIFU-long version.

PubMed (Cohan et al., 2018) is a long document summarization dataset from scientific domain whose avg summary length is about 4 times longer than CNN/DM.

SSN (An et al., 2021a) consists of papers mainly from math, physics and computer science with the abstract section as gold reference.

4.2 Implementation Details

For the simplity of experimental settings, both extractive model and abstractive mode are based on BART. We use the encoder of BART (170M) as the backbone and a 3-layer MLP as the classifier to implement the extractor. We add two special token '<cls>' to generate the sentence representation and '<sep>' as sentence separator. '<doc>' token is used to generate the document feature representation. candidate size for each dataset can be found in 1 We use adam optimizer (Kingma and Ba, 2014) learning rate schedule follows the setting in transformer (Vaswani et al., 2017). We train our model for 15000 steps with BCELoss and 32000 steps with BCELoss and RankingLoss where each step has a batch size of 36. The margin parameter γ is set to 0.01. The size of generated candidates $|\mathcal{C}|$ is set to 20 for CNN/DM. We report the results. Other settings follow the default setting in Liu and Lapata (2019). Our model is trained on single GeForce RTX 3090 GPU for 8 hours. Both our abstractive model and extractive model are trained on 24G GeForce RTX 3090 GPUs and the inference process is on 12G GeForce GTX TITAN XP GPUs.

For abstractive model, we choose BART initialized with facebook/bart-large-cnn from transformers[3] as the basic summarizer. We further fintune this model by NLLLoss and RankingLoss for 15000 steps where each step with a batch size of 8. Other setting is the same with our extractive version. To encourage diversity, we use the diverse beam search (Vijayakumar et al., 2016) to generate the candidates with beam size set to 16 and diversity penalty set to 1.0. Our model is trained on 8 GeForce RTX 3090 GPUs for about 18 hours.

4.3 Evaluation Metrics

We examine our approach with 4 metrics that measure the distance between generated summaries against the gold reference. **ROUGE** (Lin, 2004) where R-1 and R-2 measure informativeness based on n-gram overlapping and R-L represents fluency. **JS-2 Divergence** (Louis and Nenkova, 2013) measures Jensen-Shannon divergence between the bigram distributions of two input texts.

[2] https://github.com/huggingface/datasets

[3] https://github.com/huggingface/transformers

BERTScore (Zhang et al., 2019) measures soft overlap between BERT embeddings of two texts instead of using lexical matching methods. **MoverScore** (Zhao et al., 2019) is also based on the neural model but applies a earth mover distance measure to contextualized BERT embeddings.

5 Results

We denote the model without contrastive learning as the baseline system. Since the backbone of our extractive model is BART encoder so that we call the baseline model BARTEXT. The baseline model for abstractive system is BART. Our extractive model is called COLO$_{Ext}$ and its abstractive version is denoted as COLO$_{Abs}$.

5.1 Extractive Results

We compare our models with baseline models which has similar amount of parameters and decoding speed of our models in this section. Our extractive results on CNN/DM are shown in Table 2 We compare our model with previous strong extractive baseline built on pre-trained model (Zhong et al., 2019; Bae et al., 2019; Liu and Lapata, 2019) and strong multi-stage systems (Zhong et al., 2020). From the third section of Table 2, we can see that our model COLO$_{Ext}$ beats the baseline model by 1.49 ROUGE-1 score and achieve the state-of-the-art among all end-to-end systems when input length set to 512 and the results can be further improved while extending the input length to 1024. Even compared with the BERTSUM-large (340M) (Liu and Lapata, 2019) which is built on large PTM, We still have an improvement of 0.42 with only the half number of parameters of theirs. Though RL-based methods hold the motivation of optimizing towards the evaluation metric, but it does not gain much improvement on performance in practice.

To verify whether our model is effective on datasets of various lengths, we also evaluate our model on datasets with short summaries (Reddit and XSum) and long document dataset PubMed and results are shown in Table 3. On reddit and XSum, we achieve the advantage of more than 1.0 point ROUGE-1 than baseline systems and close performance with the upper bound ORACLE. We also gain improvements when tested on the long document summarzation dataset PubMed. Detailed results on long document dataset can be found in Appendix A.

Model	R-1	R-2	R-L
LEAD	40.43	17.62	36.67
ORACLE	52.59	31.23	48.87
Transformer(Vaswani et al., 2017)	40.90	18.02	37.17
BERT-EXT(Bae et al., 2019)	42.29	19.38	38.63
BERT-EXT + RL	42.76	19.87	39.11
BertSum (Liu and Lapata, 2019)	42.57	19.96	39.04
BertSum-large	<u>43.85</u>	20.34	39.90
BARTEXT	42.78	20.24	39.24
BARTEXT ($len = 1024$)	43.65	<u>20.88</u>	<u>40.19</u>
Naive one-stage	43.53	20.54	39.62
COLO$_{Ext}$	44.10	20.97	40.19
COLO$_{Ext}$ + BERTScore	44.27	21.01	40.34
COLO$_{Ext}$ ($len = 1024$)	**44.58**	**21.25**	**40.65**

Table 2: Extractive results on CNN/DM test set. len means the input length of the document, results without the marker using 512 tokens as input. +RL means the addition of reinforcement learning. +BERTScore means we use BERTScore to determine positive-negative samples. COLO clearly outperform all previous one-stage summarization systems. The best results are in bold and the second best ones are underlined.

5.2 Abstractive results

Early work also successfully applies reinforcement learning on abstractive summarization (Paulus et al., 2017; Li et al., 2019). But we do not find related works that successfully combine reinforcement learning with strong pre-trained models. Therefore, most of our baselines are strong pre-trained model finetuned with NLLLoss. Our results is shown in Table 4, due to the huge cost of using large pre-trained model with length set to 1024, we also report results with 512 input tokens and it is able to significantly outperform other baselines which has longer input length (1024). COLO$_{Abs}$ has an improvement of **2.17** R-1 socre on the very strong BART-large baseline without adding additional parameters or modules. Additionally, our method is able to outperform all one-stage baseline systems by a large margin. We also conduct experiments on long document summarzation datasets (see in Table 11 in Appendix).

5.3 Comparison with Multi-stage Systems

Apart from the one-stage systems, we also compare our model with these powerful multi-stage systems: CTRLSum, multi-stage re-ranking models. CTRLSum needs other systems to previously produce a control signal.

Model	Reddit			XSum			PubMed		
	R-1	R-2	R-L	R-1	R-2	R-L	R-1	R-2	R-L
LEAD	12.38	2.17	10.12	14.40	1.46	10.59	37.58	12.22.	33.44
ORACLE	29.10	11.08	23.10	25.62	7.62	18.72	45.12	20.33	40.19
BERTSUM	23.86	5.85	19.11	22.86	4.48	17.16	41.05	14.88	36.57
BARTEXT	23.97	5.68	19.24	22.96	4.70	17.29	41.40	16.18	37.89
COLO$_{Ext}$	**25.06**	**5.90**	**19.52**	**24.51**	**5.04**	**18.21**	**41.93**	**16.51**	**38.28**

Table 3: Results on test sets of reddit, XSum and PubMed. Our model achieve significant improvement on the baseline model BARTEXT. LEAD means we select the first k sentences from the source document as the output summary and ORACLE is the upper bound of extractive methods.

Model	R-1	R-2	R-L
BertSumAbs(Liu and Lapata, 2019)	41.72	19.39	38.76
Pegasus(Zhang et al., 2020)	44.17	21.47	41.11
BART(Lewis et al., 2020)	44.16	21.28	40.90
BART+R3F(Aghajanyan et al., 2020)	44.38	21.53	41.17
BART ($len = 512$)	43.82	20.96	40.63
ConSum (Sun and Li, 2021)	44.53	21.54	41.57
SeqCo (Xu et al., 2021)	45.02	21.80	41.75
Naive one-stage (ROUGE, $len = 512$)	43.90	20.88	40.69
COLO$_{Abs}$ (ROUGE, $len = 512$)	45.45	21.53	42.35
COLO$_{Abs}$(ROUGE)	**46.33**	**22.15**	**43.08**

Table 4: Abstractive results on CNN/DM test set. len means the maximum input length of the encoder, results without the marker using 1024 tokens as the input. ConSum (Sun and Li, 2021) and SeqCo (Xu et al., 2021) in the second block are also previous contrastive learning based methods without re-ranking.

Model	R-1	R-2	R-L
extractive systems			
COLO$_{Ext}$	44.27	21.01	40.34
BERT+BERT$^{\mathcal{R}}$ (Zhong et al., 2020)	44.22	20.62	40.38
BERT+RoBERTa$^{\mathcal{R}}$ (Zhong et al., 2020)	44.41	20.86	40.55
COLO$_{Ext}$ +RoBERTa$^{\mathcal{R}}$	**44.70**	**21.03**	**40.74**
abstractive systems			
COLO$_{Abs}$	46.33	22.15	43.08
CTRLSum(He et al., 2020)	45.65	**22.35**	42.50
BART+RoBERTa$^{\mathcal{R}}$ (Liu and Liu, 2021)	**46.67**	22.15	**43.54**

Table 5: Comparision with the multi-stage systems. RoBERTa$^{\mathcal{R}}$ means a RoBERTa re-ranker is is added to the summarization model.

performance The addition of another pre-trained model implicitly introduces more parameters and knowledge, thus it is usually unfair to directly compare one-stage systems with the two-stage systems. But we show that COLO is able to achieve comparable performance with the multi-stage systems. As is shown in the first part of Table 5, compared with the multi-stage models that ensembles another pre-trained encoder as a re-ranker, COLO$_{Ext}$ still performs better than their BERT+BERT$^{\mathcal{R}}$ version without the need to re-feed the generated candidates to another model meanwhile we obtain a $\sim\times 5$ speed up over the multi-stage systems. We also try concatenating a re-ranker RoBERTa for our model, results shows that COLO$_{Ext}$ can be further improved by combing another pre-trained re-ranker reaching new extractive SOTA on the test set of CNN/DM. For abstractive models, our end-to-end model still legs behind multi-stage systems but we do not need training another model and keep similar inference speed with baseline models.

Inference Efficiency Despite the fact that multi-stage models outperform all end-to-end systems, they frequently suffer from inefficiency. In this part we mainly focus on analysing the efficiency of 3 kinds of systems: 1) baseline, which is trained only with BCELoss or NLLLoss, 2) COLO, our end-to-end constrastive learning framework, 3) Rerank, which means the multi-stage re-ranking systems. it has more 110M parameters than baseline model and COLO. The efficiency experiments for training and inference are respectively conducted on 24G RTX 3090 GPUs and 12G TITAN XP GPUs. For extractve summarization, figures 3(a),3(b) give a detailed comparison of the inference speed between the three models. Y-axis represents the number of samples processed per second. To give a fair comparison, we test the inference efficiency in two settings: i) all models are tested with batch size fixed to 1. ii) all models are tested with the maximum batch size allowed by the GPU. While the candidate size varies from 4$\sim$32, both our model have a $\mathbf{3}\times \sim \mathbf{8}\times$ speed-up ratio over the multi-stage re-ranking model. When the candidate size

Systems	Stage1	Preprocess	Stage2	Total hours
Ext+RoBERTa$^{\mathcal{R}}$	4	5 (+20)	128	137 (+20)
CoLo$_{Ext}$	7	–	–	7 ($\downarrow$ **130**)
Abs+RoBERTa$^{\mathcal{R}}$	80	132 (+18)	128	340 (+18)
CoLo$_{Abs}$	224	–	–	224 ($\downarrow$ **116**)

Table 6: GPU hours spent on training for each process on the training set of CNN/DM(reported results are rounded down after the decimal point. Ext+RoBERTa$^{\mathcal{R}}$/Abs+RoBERTa$^{\mathcal{R}}$ denotes the multi-stage re-ranking systems with an extracitve/abstrastive summarizer. (+18)/(+20) means 18/20 CPU hours are spent on calculate ROUGE score for each candidate with 32 threads.

is set to 20, the baseline model is able to process ~31.2/41.9 (batch = 1/*MAX*) samples per second, the decoding speed of CoLo$_{Ext}$ is ~30.4/38.9 samples/s (batch=1/*MAX*) and the decoding speed of the multi-stage re-ranking model is only ~4.9/7.0 samples/s(batch=1/*MAX*). Our model almost does no harm on inference speed while the candidate size $|\mathcal{C}|$ is less than 16. However, when the candidate size grows larger there is more time spent on generating the representations of the candidates. Figure 4 show the comparison of inference time of the abstractive models. While the bottleneck of abstractive models is the auto-regressive generation process. Our abstractive model generally save ~0.5 GPU hours compared to the re-ranking model.

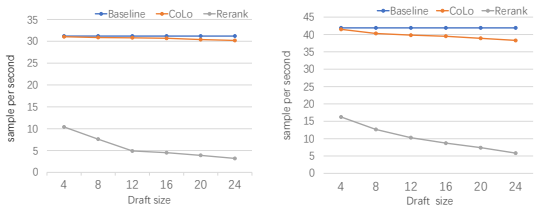

(a) CNN/DM (batch=1) (b) CNN/DM (batch =*MAX*)

Figure 3: Inference speed on CNN/DM (extractive). we use the candidate size $|\mathcal{C}|$ as the X-axis. The Y-axis represents the number of samples processed per second. batch=*MAX* means we use the maximum batch size allowed by GPU memory.

Training Efficiency Table 6 gives an overview of the training time of our system and the multi-stage models on the training set of CNN/DM. The general pipeline for the multi-stage models is: i) training a generator (Stage1), ii) Preprocess, ii) training a re-ranker (Stage2). The preprocess includes generating the training/dev/test set for training re-ranker and sorting candidates by ROUGE. For extractive system we save **130** GPU hours compared to the multi-stage systems whose bottleneck is training the re-ranking model. For abstractive model, apart from the 128 GPU hours spent on training the ranker, using beam search to generate the training set for re-ranker model is also very time consuming, generally we obtain **116** GPU hours and 18 CPU hours saved.

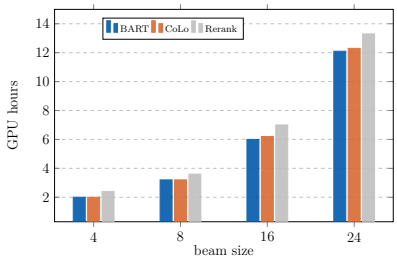

Figure 4: Test inference time with beam size for abstractive model. We use the maximum batch size allowed by GPU memory.

Metric Used	R-1	R-2	R-L	JS-2	BS	MS
Baseline	42.78	20.24	39.23	54.24	43.52	58.27
ROUGE-1,2	44.10	20.97	40.19	54.07	44.26	58.63
ROUGE-L	44.09	20.93	**40.34**	54.06	44.32	58.60
JS-2	43.85	**21.13**	39.98	**53.92**	44.19	58.60
BERTScore	**44.27**	21.01	**40.34**	54.08	**44.85**	58.71
MoverScore	44.21	20.81	40.25	54.33	44.47	**58.78**

Table 7: Extractive results of using different evaluation metrics as the discriminator on CNN/DM test set.

5.4 Ablation for Different Discriminators

In addition to ROUGE, we also select other metrics as the discriminator (shown in Table 7). ROUGE and JS-2 is based on lexical matching while BERTScore and MoverScore are based on the contextualized embedding from BERT. Our model generally obtains the best results on the metric used in training. Because these metrics are not actually separated, using one of these metrics as the discriminator can also gain significant improvements on other metrics. Overall, the neural evaluation metric BERTScore and MoverScore bring more improvements compared with metrics that based on the lexical matching. But incorporating neural model based metrics in training will obviously increase the training time.

5.5 Visualization Experiment

We conduct a visualization experiment on our extractive model to get a close look on the distribution of candidates in semantic space. We randomly sample 100 documents with more than 10 sentences from the test set of CNN/DM. We first select the top 10 sentences based on the predicted score from the classifier. We set the possible number of sentences to $\{2, 3\}$ resulting a candidate size of $C_{10}^2 + C_{10}^3 = 165$ for each sample. We visualize the learned embedding of these candidates and the anchor in a two-dimensional space by applying the t-SNE algorithm. As shown in Figure 5, there is an obvious cluster of the top 50 candidates (colored in purple) and the candidates with higher score are closer to the anchor while the distribution of uninformative candidates (gray,cyan points) is relatively random.

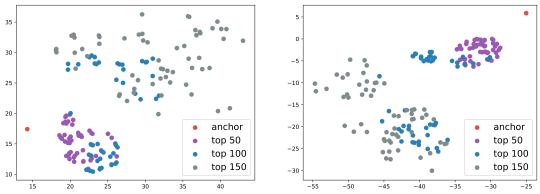

Figure 5: T-SNE Visualization of two examples from CNN/DM test set. We divide the candidates into 3 groups based on ROUGE score: candidates ranking 1~50, candidates ranking 51~100, candidate ranking 101~150. The red point denotes the anchor and the purple/cyan/gray points respectively denote the top 50/100/150 candidates.

5.6 Human Evaluation

We also conduct a human evaluation on our models to get more accurate results . We randomly select 30 articles from the test set of CNN/DM, and each articles have 5 candidate summaries 4 from automatic systems and 1 is the gold reference. We recruit 2 PhD students majoring in computer science and ask them to rank the candidate summries based on the fluency, informativeness. If two of these systems generate the same summary for the source document, this sample will be filtered out. As we can see from Table 8, the CoLo$_{Ext}$ with the discriminator as BERTScore achieve the best result among all automatic systems. However, using BERTScore will bring much training time. We also suggest taking JS-2 divergence as the discriminator which also does a good job in human evaluation.

Metric Used	1st	2nd	3rd	4th	5th	Avg R.
Baseline	0%	8.3%	8.3%	23.3%	60%	4.33
JS2	6.7%	25%	33.3%	21.7%	13.3%	3.10
R1+R2	5%	20%	28.3%	30.3%	16.7%	3.35
BERTScore	10%	35%	20%	25%	10%	2.90
Gold label	78.3%	11.7%	10%	0%	0%	1.32

Table 8: Results of human evaluation results. Baseline means the BartExt model, Gold-label means the means the human written summary. Avg R. denotes the average ranking of the system.

6 Limitations and Future Work

Compared with the most well-known contrastive learning framework simCLR (Chen et al., 2020) which propose to construct positive and negative pairs from training samples in the same batch, Drawing negative-positive pairs from the summarization model requires more training time. Ideally, providing more positive and negative samples will benefit the performance of CoLo . However, decoding with very large beam size in training mode will cost more GPU memory and training time. Future work can search for an efficient way to construct these positive-negative pairs to perform reranking during training.

7 Conclusion

We introduce CoLo, a contrastive learning based summarization framework for one-stage summarization where positive-negative pairs are generated directly from the summaizer with online sampling. CoLo can be both easily applied on extractive and abstractive methods. Results show that we greatly exceed previous stage-of-the art one-stage systems with no additional parameters and obivious decline of the inference efficiency.

Acknowledgement

We would like to thank Yixin Liu and the anonymous reviewers for their valuable advice. This work was supported by the National Key Research and Development Program of China (No.2020AAA0106702) and National Natural Science Foundation of China (No.62022027).

References

Armen Aghajanyan, Akshat Shrivastava, Anchit Gupta, Naman Goyal, Luke Zettlemoyer, and Sonal Gupta.

2020. Better fine-tuning by reducing representational collapse. *arXiv preprint arXiv:2008.03156*.

Chenxin An, Jiangtao Feng, Kai Lv, Lingpeng Kong, Xipeng Qiu, and Xuanjing Huang. 2022. Cont: Contrastive neural text generation. *arXiv preprint arXiv:2205.14690*.

Chenxin An, Ming Zhong, Yiran Chen, Danqing Wang, Xipeng Qiu, and Xuanjing Huang. 2021a. Enhancing scientific papers summarization with citation graph. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 35, pages 12498–12506.

Chenxin An, Ming Zhong, Zhichao Geng, Jianqiang Yang, and Xipeng Qiu. 2021b. Retrievalsum: A retrieval enhanced framework for abstractive summarization. *arXiv preprint arXiv:2109.07943*.

Sanghwan Bae, Taeuk Kim, Jihoon Kim, and Sanggoo Lee. 2019. Summary level training of sentence rewriting for abstractive summarization. In *Proceedings of the 2nd Workshop on New Frontiers in Summarization*, pages 10–20.

Samy Bengio, Oriol Vinyals, Navdeep Jaitly, and Noam Shazeer. 2015. Scheduled sampling for sequence prediction with recurrent neural networks. *arXiv preprint arXiv:1506.03099*.

Ting Chen, Simon Kornblith, Mohammad Norouzi, and Geoffrey Hinton. 2020. A simple framework for contrastive learning of visual representations. In *International conference on machine learning*, pages 1597–1607. PMLR.

Jianpeng Cheng and Mirella Lapata. 2016. Neural summarization by extracting sentences and words. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, volume 1, pages 484–494.

Arman Cohan, Franck Dernoncourt, Doo Soon Kim, Trung Bui, Seokhwan Kim, Walter Chang, and Nazli Goharian. 2018. A discourse-aware attention model for abstractive summarization of long documents. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 2 (Short Papers)*, volume 2, pages 615–621.

Sebastian Gehrmann, Yuntian Deng, and Alexander Rush. 2018. Bottom-up abstractive summarization. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 4098–4109.

Junxian He, Wojciech Kryściński, Bryan McCann, Nazneen Rajani, and Caiming Xiong. 2020. Ctrlsum: Towards generic controllable text summarization. *arXiv preprint arXiv:2012.04281*.

Karl Moritz Hermann, Tomas Kocisky, Edward Grefenstette, Lasse Espeholt, Will Kay, Mustafa Suleyman, and Phil Blunsom. 2015. Teaching machines to read and comprehend. In *Advances in Neural Information Processing Systems*, pages 1684–1692.

Byeongchang Kim, Hyunwoo Kim, and Gunhee Kim. 2019. Abstractive summarization of reddit posts with multi-level memory networks. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 2519–2531.

Diederik Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*.

Seanie Lee, Dong Bok Lee, and Sung Ju Hwang. 2020. Contrastive learning with adversarial perturbations for conditional text generation. *arXiv preprint arXiv:2012.07280*.

Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Veselin Stoyanov, and Luke Zettlemoyer. 2020. Bart: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7871–7880.

Siyao Li, Deren Lei, Pengda Qin, and William Yang Wang. 2019. Deep reinforcement learning with distributional semantic rewards for abstractive summarization. *arXiv preprint arXiv:1909.00141*.

Chin-Yew Lin. 2004. Rouge: A package for automatic evaluation of summaries. In *Text summarization branches out*, pages 74–81.

Yang Liu and Mirella Lapata. 2019. Text summarization with pretrained encoders. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 3721–3731.

Yixin Liu and Pengfei Liu. 2021. Simcls: A simple framework for contrastive learning of abstractive summarization. *arXiv preprint arXiv:2106.01890*.

Annie Louis and Ani Nenkova. 2013. Automatically assessing machine summary content without a gold standard. *Computational Linguistics*, 39(2):267–300.

Ramesh Nallapati, Feifei Zhai, and Bowen Zhou. 2017. Summarunner: A recurrent neural network based sequence model for extractive summarization of documents. In *Thirty-First AAAI Conference on Artificial Intelligence*.

Ramesh Nallapati, Bowen Zhou, Caglar Gulcehre, Bing Xiang, et al. 2016. Abstractive text summarization using sequence-to-sequence rnns and beyond. *arXiv preprint arXiv:1602.06023*.

Shashi Narayan, Shay B Cohen, and Mirella Lapata. 2018a. Don't give me the details, just the summary! topic-aware convolutional neural networks for extreme summarization. In *Proceedings of the 2018*

Conference on Empirical Methods in Natural Language Processing, pages 1797–1807.

Shashi Narayan, Shay B Cohen, and Mirella Lapata. 2018b. Ranking sentences for extractive summarization with reinforcement learning. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 1747–1759.

Romain Paulus, Caiming Xiong, and Richard Socher. 2017. A deep reinforced model for abstractive summarization. *arXiv preprint arXiv:1705.04304*.

Alexander M Rush, Sumit Chopra, and Jason Weston. 2015. A neural attention model for abstractive sentence summarization. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 379–389.

Felix Stahlberg and Bill Byrne. 2019. On nmt search errors and model errors: Cat got your tongue? In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 3356–3362.

Shichao Sun and Wenjie Li. 2021. Alleviating exposure bias via contrastive learning for abstractive text summarization. *arXiv preprint arXiv:2108.11846*.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in neural information processing systems*, pages 5998–6008.

Ashwin K Vijayakumar, Michael Cogswell, Ramprasath R Selvaraju, Qing Sun, Stefan Lee, David Crandall, and Dhruv Batra. 2016. Diverse beam search: Decoding diverse solutions from neural sequence models. *arXiv preprint arXiv:1610.02424*.

Shusheng Xu, Xingxing Zhang, Yi Wu, and Furu Wei. 2021. Sequence level contrastive learning for text summarization. *arXiv preprint arXiv:2109.03481*.

Jingqing Zhang, Yao Zhao, Mohammad Saleh, and Peter Liu. 2020. Pegasus: Pre-training with extracted gap-sentences for abstractive summarization. In *International Conference on Machine Learning*, pages 11328–11339. PMLR.

Tianyi Zhang, Varsha Kishore, Felix Wu, Kilian Q Weinberger, and Yoav Artzi. 2019. Bertscore: Evaluating text generation with bert. *arXiv preprint arXiv:1904.09675*.

Wei Zhao, Maxime Peyrard, Fei Liu, Yang Gao, Christian M Meyer, and Steffen Eger. 2019. Moverscore: Text generation evaluating with contextualized embeddings and earth mover distance. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 563–578.

Ming Zhong, Pengfei Liu, Yiran Chen, Danqing Wang, Xipeng Qiu, and Xuanjing Huang. 2020. Extractive summarization as text matching. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, ACL 2020, Online, July 5-10, 2020*, pages 6197–6208. Association for Computational Linguistics.

Ming Zhong, Pengfei Liu, Danqing Wang, Xipeng Qiu, and Xuan-Jing Huang. 2019. Searching for effective neural extractive summarization: What works and what's next. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 1049–1058.

Ming Zhong, Yang Liu, Suyu Ge, Yuning Mao, Yizhu Jiao, Xingxing Zhang, Yichong Xu, Chenguang Zhu, Michael Zeng, and Jiawei Han. 2022. Unsupervised summarization with customized granularities. *arXiv preprint arXiv:2201.12502*.

Of Human Criteria and Automatic Metrics: A Benchmark of the Evaluation of Story Generation

Cyril Chhun[1] **Pierre Colombo**[2*] **Fabian M. Suchanek**[1] **Chloé Clavel**[1]
[1]LTCI, Télécom Paris, Institut Polytechnique de Paris
[2]Lab of Mathematics and Informatics (MICS), CentraleSupélec, Université Paris-Saclay
cyril.chhun@telecom-paris.fr

Abstract

Research on Automatic Story Generation (ASG) relies heavily on human and automatic evaluation. However, there is no consensus on which human evaluation criteria to use, and no analysis of how well automatic criteria correlate with them. In this paper, we propose to re-evaluate ASG evaluation. We introduce a set of 6 orthogonal and comprehensive human criteria, carefully motivated by the social sciences literature. We also present HANNA, an annotated dataset of 1,056 stories produced by 10 different ASG systems. HANNA allows us to quantitatively evaluate the correlations of 72 automatic metrics with human criteria. Our analysis highlights the weaknesses of current metrics for ASG and allows us to formulate practical recommendations for ASG evaluation.

1 Introduction

Storytelling is at the heart of human societies: skillful storytelling allows a narrator to connect more authentically with their audience and listeners, and to understand the essence of complex concepts better (Suzuki et al., 2018). Numerous applications could benefit from strong automatic story generation systems, including gaming (Hartsook et al., 2011), communication (Alhussain and Azmi, 2021), and education (Aylett et al., 2007). Several approaches have been explored to generate stories automatically or with minimum editing efforts (Alabdulkarim et al., 2021). *Automatic story generation* (ASG) takes as input a short sentence (a *prompt*) and aims at generating a narrative from it (Cavazza and Pizzi, 2006; Lebowitz, 1985). Advances in neural language models (Radford et al., 2018, 2019; Brown et al., 2020) have allowed substantial progress in ASG.

To further improve the quality of generated stories, it is indispensable to systematically evaluate ASG models. However, there is little work that specifically studies ASG evaluation. Most research works rely on human criteria such as coherence (Xu et al., 2018; Colombo et al., 2019; Jalalzai et al., 2020), relevance (Jhamtani and Berg-Kirkpatrick, 2020), overall quality (Brahman and Chaturvedi, 2020), narrative flow (Rashkin et al., 2020), and creativity (Pascual et al., 2021). However, taken individually, these criteria fail to encompass all aspects of the task, and there is no consensus on a set of criteria that would cover those aspects in a complete and non-redundant fashion. Due to the high cost of human annotation, system quality is also often evaluated using automatic metrics. However, it is not clear how these metrics correlate with human judgment in ASG, and thus how suitable they are at all for the evaluation of ASG.

Contributions. In this work, we revisit both human and automatic evaluation of ASG. We believe that this meta-evaluation is a missing piece in the ASG literature and a crucial step to strengthening the foundations of ASG. Formally, our contributions to the ASG field are:

1. **A comprehensive set of non-redundant human criteria for ASG evaluation.** Motivated by the social sciences literature (McCabe and Peterson, 1984; Dickman, 2003; Bae et al., 2021), we introduce six human criteria: relevance, coherence, empathy, surprise, engagement and complexity.

2. **HANNA**[1]**, a large annotated dataset of Human-ANnotated NArratives for ASG evaluation**, which contains 1,056 stories generated from 96 prompts. Each prompt is linked to a human story and stories generated by 10 different ASG generation systems. Each story was annotated by 3 different human raters along our 6 proposed human criteria.

[*]Previously from Laboratoire des Signaux et Systèmes (L2S), CentraleSupélec, CNRS, Université Paris-Saclay.

[1]The HANNA dataset and corresponding code are available on https://github.com/dig-team/hanna-benchmark-asg.

Proceedings of the 29th International Conference on Computational Linguistics, pages 5794–5836
October 12–17, 2022.

3. **A meta-evaluation of ASG with fine-grained recommendations.** Relying on HANNA, we perform an extensive study of the performance of the ASG systems and we analyze the correlations of 72 existing automatic metrics with our proposed human criteria. The obtained results demonstrate the limitations of current automatic evaluation methods and allow us to make recommendations on which metrics to use for ASG evaluation.

2 Related work

2.1 Human evaluation

van der Lee et al. (2019) advise to define separate and precise criteria for human evaluation to make it as accurate as possible. However, in ASG, there is no consensus on the criteria to be used: among others, we find a pairing task (Fan et al., 2018), fluency and coherence (Xu et al., 2018), creativity (Pascual et al., 2021), faithfulness (Peng et al., 2018; Wang et al., 2020), fidelity (Yao et al., 2019), grammar and logicality (Guan et al., 2019, 2020), overall quality and relevance (Jhamtani and Berg-Kirkpatrick, 2020; Goldfarb-Tarrant et al., 2020; Guan et al., 2021b), outline utilisation and narrative flow (Rashkin et al., 2020), emotion faithfulness (Witon et al., 2018), and content quality (Brahman and Chaturvedi, 2020). Many of these criteria are not specific to ASG (fluency, grammar, overall quality, content quality), overlap with one another (pairing task, faithfulness, and fidelity are variations of relevance; logicality and narrative flow, of coherence) or are ascribed to a specific setting (outline utilisation, emotion faithfulness). Furthermore, evaluation protocols mostly use only two or three criteria, which is not enough to grasp all aspects of a task as complex as ASG. They also do not associate Likert scales with explicit descriptions, even though such descriptions could reduce the subjectivity of the labelling process.

2.2 Automatic evaluation

Although most of the research work in ASG relies on BLEU and ROUGE, there exists a plethora of automatic metrics to evaluate ASG. These can be classified into two categories: *reference-based* (Ξ) metrics evaluate a candidate text by comparing it to a reference text (in our case, the human story), and *reference-free* (¤) metrics rely only on the candidate story (and, possibly, on the prompt). In both categories, we find *string-based* (§), *embedding-based* (ε) and *model-based* (Δ) metrics. String-based metrics evaluate the textual representation of the inputs; they cannot handle synonyms or paraphrases. By contrast, embedding-based metrics rely on word embeddings, *e.g.* word2vec (Mikolov et al., 2013a,b), or contextualized embeddings, *e.g.* obtained from BERT (Devlin et al., 2019). Finally, model-based metrics leverage regression or pre-trained language models to return a score. A synoptic classification can be found in Tab. 1[2].

	Reference-based (Ξ)	Reference-free (¤)
String-based (§)	BLEU (Papineni et al., 2002) ROUGE (Lin, 2004) METEOR (Banerjee and Lavie, 2005) CHRF (Popović, 2015) CIDEr (Vedantam et al., 2015)	Coverage (Grusky et al., 2018) Density (Grusky et al., 2018) Compression (Grusky et al., 2018) Text length (Fabbri et al., 2021) Novelty (Fabbri et al., 2021) Repetition (Fabbri et al., 2021)
Embedding-based (ε)	ROUGE-WE (Ng and Abrecht, 2015) BERTScore (Zhang et al., 2020) MoverScore (Zhao et al., 2019) BaryScore (Colombo et al., 2021d) DepthScore (Staerman et al., 2021)	SUPERT (Gao et al., 2020)
Model-based (Δ)	S3 (Peyrard et al., 2017) SummaQA (Scialom et al., 2019) InfoLM (Colombo et al., 2022c) BARTScore (Yuan et al., 2021)	BLANC (Vasilyev et al., 2020)

Tab. 1: Classification of the automatic metrics considered in our study with symbols for easier identification.

2.3 Meta-evaluation

Several previous works have studied the relationship between automatic metrics and human judgment (Zhang et al., 2004; Ma et al., 2019), reporting weak correlation (Novikova et al., 2017; Stent et al., 2005; Mathur et al., 2020) and strong bias towards specific systems (Callison-Burch et al., 2006). Meta-evaluation has been done in image description (Elliott and Keller, 2014), dialogue response generation (Liu et al., 2016), question generation (Nema and Khapra, 2018), table-to-text generation (Dhingra et al., 2019), question answering (Chen et al., 2019), and summarization (Bhandari et al., 2020). In ASG, Guan et al. (2021b) introduced the OpenMEVA benchmark which compares the overall quality of human and generated stories; their work especially focused on the textual features of stories. We build upon it and perform a comprehensive analysis of the correlations between 72 automatic metrics and 6 human criteria specifically tailored for ASG.

[2] BARTScore was designed to be either reference-based or reference-free depending on the setting.

3 HANNA for ASG evaluation

3.1 ASG datasets

Story evaluation has been widely studied in different scenarii. ROCStories (Mostafazadeh et al., 2016), a corpus of 50k 5-sentence stories with titles, was designed for the Story Cloze Test: the prediction of the final sentence of a story given the four others. Huang et al. (2016) developed the VisualStorytelling dataset, which contains sequences of images with corresponding descriptions divided in three tiers of temporal context. More recently, Ammanabrolu et al. (2020) proposed the WorldGeneration dataset which adapts story generation to adventure games by guiding the generation process with location, character and object triplets. The WritingPrompts (WP) dataset (Fan et al., 2018) contains stories generated from short sentences called *prompts*. For our work, we chose the WP dataset, because it has been extensively used in previous literature for the design of ASG models (Rashkin et al., 2020; Goldfarb-Tarrant et al., 2020; Fang et al., 2021; Wilmot and Keller, 2021; Guan et al., 2021a). While ROCStories has also been used in several works, the shortness of the stories made it less suited for our evaluation. An example prompt and story from WP is shown in Tab. 2 (Fan et al., 2018).

3.2 Chosen setting

HANNA, our annotated dataset for ASG, contains outputs from 10 different systems aligned on 96 common prompts with human stories from the WP dataset, for 1,056 stories in total, with 3 human annotations per story (19,008 annotations in total) and automatic metric scores, allowing for an analysis of the correlations between these metrics (Sec. 4).

3.3 Chosen ASG sytems

We directly contacted the authors of articles that introduced ASG systems and asked for the outputs of their systems. We managed to collect the outputs of **3 ASG systems**[3] on the WP dataset: Fusion (Fan et al., 2018), HINT (Guan et al., 2021a), and TD-VAE (Wilmot and Keller, 2021). We extracted 96 stories aligned on common prompts. We then fine-tuned **7 pre-trained language models** for ASG on a causal language modeling task on WP to generate stories on the same 96 prompts, using the *Transformers* library (Wolf et al., 2020)[4]. We trained BertGeneration (Rothe et al., 2020), CTRL (Keskar et al., 2019), RoBERTa (Liu et al., 2019), XLNet (Yang et al., 2019), GPT (Radford et al., 2018), GPT-2 (Radford et al., 2019), and GPT-2 (tag), another instance of GPT-2 trained with <EOP> (End Of Prompt) tags, as inspired by Bai et al. (2021), who argued that such tags could improve generation.

3.4 Proposed human criteria

As mentioned in Ssec. 2.1, there is no consensus on human criteria for ASG evaluation. At the same time, work in social sciences has looked extensively at the features that make for a "good" story (McCabe and Peterson, 1984; Dickman, 2003; Bae et al., 2021). We condense them as follows into a new, comprehensive set of criteria:

1. **Relevance** (RE): how well the story matches its prompt, used in Jhamtani and Berg-Kirkpatrick (2020); Goldfarb-Tarrant et al. (2020);

2. **Coherence** (CH): how much the story makes sense, used in Xu et al. (2018); Peng et al. (2018); Yao et al. (2019); Pascual et al. (2021);

3. **Empathy** (EM): how well the reader understood the character's emotions, derived from the importance of emotional commentary (McCabe and Peterson, 1984), passion (Dickman, 2003), and empathy (Keen et al., 2007; Bae et al., 2021);

4. **Surprise** (SU): how surprising the end of the story was, derived from the importance of schema violation, or unexpectedness (Schank, 1978; Bae et al., 2021), postdictability (Behrooz et al., 2019), and novelty (Randall, 1999);

5. **Engagement** (EG): how much the reader engaged with the story; a more subjective criterion associated with projecting volitive modality (making the reader formulate a subjective judgment and express a desire to see something accomplished) (Toolan, 2012) and story outcome, which is an underlying cause of story liking (Iran-Nejad, 1987);

6. **Complexity** (CX): how elaborate the story is; derived from the importance of detailed descriptions and sophisticated problem-solving (McCabe and Peterson, 1984) and good world-building (Roine, 2016).

[3] We also collected outputs from two other systems (Goldfarb-Tarrant et al., 2020; Bai et al., 2021); unfortunately, these were not aligned with the others.

[4] https://github.com/huggingface/transformers

The four last criteria are an original contribution and were designed to evaluate story features that are different from the first two criteria (RE and CH), which are currently most used in the ASG literature. Examples of annotations w.r.t. those criteria are shown in Tab. 2.

3.5 Annotation Protocol

To evaluate our human criteria on the 1,056 stories of HANNA, we conducted an annotation campaign on Amazon Mechanical Turk. As advised by Karpinska et al. (2021), for each task, we provided the human story alongside the story to be annotated, so that the workers could calibrate their judgment. Each of the stories was evaluated by three workers on our six proposed criteria. For this evaluation, we chose a 5-point Likert scale rather than a rank-based comparison because we reckoned that it would be tedious to order the large number of evaluated systems. We estimated that a HIT should take between 90 and 120 seconds, so we set the remuneration at \$0.28 per HIT, or between \$8.40 and \$11.40 per hour. To ensure that annotators spoke fluent English, we restricted access to the experiment to workers located in the UK, the US, Canada, Australia and New Zealand. We further required them to have the Masters Qualification. To remove noisy annotations and ensure that the workers read the stories, we chose to reject judgments that were made in fewer than 30 seconds. We additionally asked workers to write down the name of the first-mentioned fictional character of the story. The detailed instructions of the experiment and the inter-annotator agreement analysis can be found in the appendix (see Ap. A and Ssec. 4.1). Finally, following the recommendations of Shapira et al. (2019), we obtained the human score of a story by averaging the results of the three workers.

3.6 Meta-evaluation strategies

Notations. Let y_i^j be the story generated by system $j \in \{1, \dots, S\}$ for prompt $i \in \{1, \dots, N\}$, and $m(y_i^j)$ the score associated to y_i^j by a (human or automatic) metric m. Given a correlation coefficient K (*e.g.* Pearson's r (**?**), Spearman's ρ (Melamed et al., 2003) or Kendall's τ (Kendall, 1938)), two meta-evaluation strategies are commonly used to evaluate metric quality.

Story-level correlation ($K^{\text{story}}_{m_1,m_2}$) measures how suited m_1 is w.r.t. m_2 if used as a loss or reward for a model. The correlation is applied to each story among all system outputs and the mean is taken. Formally:

$$K^{\text{story}}_{m_1,m_2} \triangleq \frac{1}{N}\sum_{i=1}^{N} K(\mathbf{C}^{\text{story}}_{m_1,i}, \mathbf{C}^{\text{story}}_{m_2,i}), \quad (1)$$

$$\text{where} \quad \mathbf{C}^{\text{story}}_{m,i} \triangleq \left[m(y_i^1), \cdots, m(y_i^S)\right].$$

System-level correlation ($K^{\text{sys}}_{m_1,m_2}$) measures how suited m_1 is w.r.t. m_2 if used to compare the performance of two systems. The correlation is applied to the mean values over all stories for all systems for both metrics. Formally:

$$K^{\text{sys}}_{m_1,m_2} \triangleq K\left(\frac{1}{N}\mathbf{C}^{\text{sys}}_{m_1}, \frac{1}{N}\mathbf{C}^{\text{sys}}_{m_2}\right), \quad (2)$$

$$\text{where} \quad \mathbf{C}^{\text{sys}}_{m} \triangleq \left[\sum_{i=1}^{N} m(y_i^1), \dots, \sum_{i=1}^{N} m(y_i^S)\right].$$

Statistical significance. Correlations computed for two automatic metrics on the same annotated dataset are not independent. We follow Graham and Baldwin (2014) and use the Williams test (Williams, 1959; Moon, 2019)[5] to evaluate the significance of an increase in dependent correlations (Steiger, 1980).

4 HANNA Analysis

In this section, we analyse the scores of HANNA in detail. Tab. 4 shows that human stories achieve significantly higher scores than generated stories. Following Mathur et al. (2020), who advise to remove outliers, we compute correlations with human stories removed[6].

4.1 Inter-annotator agreement

To estimate the reliability of the annotations, we computed an intra-class coefficient for each criterion. Among the annotators which took part in the experiment, three of them covered 2490 stories, *i.e.*, more than 78% of the dataset, but no annotator graded the same story twice. Since the reliability is to be estimated for the average of the three ratings, the ICC2k coefficient (ICC for *average random raters*) is the most relevant one, according to Hallgren (2012). In particular, it accounts for the systematic errors of raters and random residual errors. The results are shown in Tab. 3.

Coefficients are dispersed between 29% and 56% with relatively small confidence intervals (except for RE and CH), which can be considered between

[5] https://github.com/inmoonlight/nlp-williams

[6] The same applies for Sec. 5.

Prompt: When you die, the afterlife is an arena where you face every insect and animal you killed in your life. If you win you go to heaven, lose you go to hell. Your job was an exterminator on earth.

Human: 3,000 years have I been fighting. Every morning, the raccoons scratch at my eyes. Every evening, the skunks spray me while the opossums chew at my feet. I have never had any tools. I have only my hands. I don't remember the place I came from before this. [...]

Story #1: First of all, not everyone was entitled to be an exterminator. But the ones that were – maybe were, like, *genius*, because, yes, I had once belonged to a less fortunate class of people – had all the opportunity to work for the damn plant killer, and it's hard work. [...]

Story #2: It was hell. Not exactly a place of torture. There were no guards in prison and you couldn't just walk through it, either, because you would get killed regardless. hell was a young man, and he was lying on his floor. He was unconscious. [...]

Story	RE	CH	EM	SU	EG	CX
Human	5	5	1	3	4	1
	2	2	3	2	2	3
	4	4	3	2	4	4
Story #1	2	4	3	1	1	1
	2	2	2	1	2	2
	2	3	2	3	3	3
Story #2	5	5	3	3	3	2
	3	2	3	2	2	3
	3	4	3	4	4	3

Metric	Human	Story #1	Story #2
BLEU$^{\Xi\S}$ (%)	1.00	0.01	0.01
ROUGE-1$^{\Xi\S}$	1.00	0.24	0.33
chrF$^{\Xi\S}$ (%)	1.00	0.32	0.39
BERTScore$^{\Xi\varepsilon}$	1.00	0.50	0.52
MoverScore$^{\Xi\varepsilon}$	1.00	0.51	0.51
BaryScore$^{\Xi\varepsilon}$	0.00	0.92	0.92
S3$^{\Xi\Delta}$	1.39	0.07	0.15
BARTScore$^{\Xi\Delta}$	-0.98	-3.97	-4.03
SUPERT$^{\mu\varepsilon}$	0.94	0.37	0.36

Tab. 2: Example prompt, human and generated stories from HANNA with human annotations and metric scores

"fair" and "moderate" according to Landis and Koch (1977). These values are in tune with existing literature (Karpinska et al., 2021; Habernal and Gurevych, 2017; Spooren and Degand, 2010; Ritter et al., 2011; Graham et al., 2017) and show the difficulty of evaluating natural language generation. Therefore, we follow the methodology of Craggs and Wood (2005) and Artstein and Poesio (2008), who argue against setting a specific agreement threshold as long as there is a detailed reporting of the methodology (see Ssec. 3.5 and Tab. 7) and confidence intervals (Tab. 3).

Criterion	LB	ICC2k	UB
RE	0.18	0.48	0.65
CH	0.10	0.29	0.48
EM	0.25	0.34	0.41
SU	0.16	0.28	0.37
EG	0.34	0.46	0.55
CX	0.48	0.56	0.63

Tab. 3: Intra-class coefficient per criterion. LB and UB are the lower- and upper-bounds of the 95% confidence interval

4.2 Evaluating our human criteria

In this experiment, we study the relationship between the proposed human criteria. To compute story-level (Fig. 1) and system-level (Fig. 2) correlations, we average the human ratings.

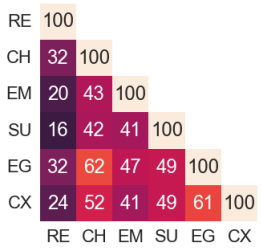

Fig. 1: Story-level Kendall correlations (%) between human criteria

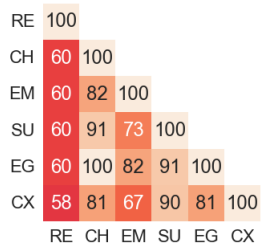

Fig. 2: System-level Kendall correlations (%) between human criteria

Story-level analysis (Fig. 1). Kendall correlations range from 16% (RE–SU) to 62% (CH–EG), averaging at 40.7%. In the appendix, we also show correlations with Spearman's ρ (Fig. 10) and Pearson's r (Fig. 12). EG correlates slightly more with CH and CX; this could confirm that coherent and intricate plots make readers more likely to connect with a story. In contrast, RE is poorly correlated to the other criteria, which makes sense: an excellent story in every other aspect can be completely unrelated to a prompt, and vice versa. Overall, moderate to weak correlations suggest that our criteria evaluate distinct aspects of storytelling which cannot be regrouped in fewer criteria.

System-level analysis (Fig. 2). Compared to story-level correlations, system level correlations are higher. Spearman (Fig. 11) and Pearson (Fig. 13)

Model	RE	CH	EM	SU	EG	CX	Average
Human	4.17 ±0.14	4.43 ±0.10	3.22 ±0.14	3.15 ±0.15	3.88 ±0.12	3.73 ±0.13	3.76 ±0.06
BertGeneration	2.46 ±0.16	3.14 ±0.16	2.28 ±0.13	**2.09** ±0.13	2.67 ±0.12	2.41 ±0.11	2.51 ±0.06
CTRL	2.54 ±0.16	2.93 ±0.16	2.26 ±0.13	1.93 ±0.12	2.53 ±0.12	2.23 ±0.10	2.40 ±0.06
GPT	2.40 ±0.16	**3.22** ±0.15	**2.37** ±0.12	**2.13** ±0.13	2.76 ±0.13	2.49 ±0.12	2.56 ±0.06
GPT-2	* **2.81** ±0.16	**3.29** ±0.14	* **2.47** ±0.12	**2.21** ±0.13	**2.86** ±0.12	2.68 ±0.10	**2.72** ±0.06
GPT-2 (tag)	**2.67** ±0.16	* **3.31** ±0.15	* **2.47** ±0.12	* **2.22** ±0.13	* **2.92** ±0.12	* **2.80** ±0.11	* **2.73** ±0.06
RoBERTa	2.54 ±0.16	3.22 ±0.16	2.27 ±0.12	**2.12** ±0.13	2.74 ±0.12	2.41 ±0.11	2.55 ±0.06
XLNet	2.39 ±0.17	2.88 ±0.16	2.10 ±0.12	1.95 ±0.12	2.46 ±0.13	2.36 ±0.11	2.36 ±0.06
Fusion	2.09 ±0.16	2.86 ±0.16	1.99 ±0.12	1.72 ±0.12	2.27 ±0.14	1.92 ±0.11	2.14 ±0.06
HINT	2.29 ±0.16	2.38 ±0.16	1.74 ±0.13	1.56 ±0.11	1.75 ±0.12	1.45 ±0.10	1.86 ±0.06
TD-VAE	2.51 ±0.16	2.99 ±0.15	2.07 ±0.11	**2.10** ±0.12	2.59 ±0.12	2.49 ±0.11	2.46 ±0.06

Tab. 4: Average system ratings per criterion with 95% confidence interval. Best value in **bold** marked with an asterisk (*), values in the confidence interval of the best value in **bold** without asterisk

correlations are also higher than their story-level counterparts. This suggests that a given system tends to be uniformly better or worse than other systems across all criteria.

4.3 Finding the best systems

On Tab. 4, we observe that generic fine-tuned models perform better than ASG systems according to human annotators. The best system is GPT-2, which scores better than all other systems on all criteria. The GPT-2 variant trained with <EOP> tags shows marginal improvement compared to GPT-2, as reported in Bai et al. (2021). It is worth noting that all models are still noticeably below human performance, which emphasizes that current systems are still a long way from human-like narrative intelligence.

5 Evaluation of automatic metrics using HANNA

In this section we evaluate how suitable existing automatic metrics are for ASG evaluation, using the SummEval library (Fabbri et al., 2021)[7]. Due to space constraints, in each figure, we selected representative metrics from each of the categories introduced in Ssec. 2.2. Full figures can be found in the appendix.

5.1 Correlations with human judgement

Story-level analysis (Fig. 3). Most metrics have either a moderate (between 30% and 50%) or weak (below 30%) correlation with human criteria. RE is particularly elusive, except for the SUPERT$^{¤\varepsilon}$ metric, which is reference-free and compares the prompt and the story. This corroborates Novikova et al. (2017), who argue that automatic metrics do not accurately reflect human judgment when comparing instances despite performing reliably at the system level. We also observe vertical "color stripes": metric performance is consistent across criteria. A weak metric will correlate poorly with all criteria, whereas a more robust metric will be uniformly better.

System-level analysis (Fig. 4). Correlations are indeed higher at the system-level, hovering between 40% and 70% for most metrics. Therefore, while metrics are poor estimators of human criteria at the story level, they can be used to compare systems with reasonable accuracy.

Best metrics per criterion (Tab. 5). We observe that a few metrics are heavily represented in the top 3 for each level. Pre-trained transformer-based metrics achieve strong results. The metrics that correlate best with human criteria at the system level are all reference-based: ROUGE-S$\star^{\Xi\S}$, BaryScore$^{\Xi\varepsilon}$, DepthScore$^{\Xi\varepsilon}$, and BARTScore$_1^{\Xi\Delta}$. At the story level, BARTScore$_2^{¤\Delta}$, Novelty-1$^{¤\S}$ and Repetition-3$^{¤\S}$ are reference-free while chrF$^{\Xi\S}$, BERTScore$^{\Xi\varepsilon}$, S3$^{\Xi\Delta}$ are reference-based. As Novelty-1 and Repetition-3 are simple data statistics, their outperforming all metrics for SU and CH respectively highlights the shortcomings of current metrics.

5.2 Correlations between automatic metrics

Story-level analysis (Fig. 5). Reference-based metrics are moderately to highly correlated with one another. By contrast, embedding- and model-based reference-free metrics such as SUPERT$^{¤\varepsilon}$ and BLANC$^{¤\Delta}$ are almost independent from all other metrics, even other reference-free metrics.

System-level analysis (Fig. 6). Previous obser-

[7] https://github.com/Yale-LILY/SummEval

Fig. 3: Story-level absolute Kendall correlations (%) between metrics and criteria. Full version: Fig. 14.

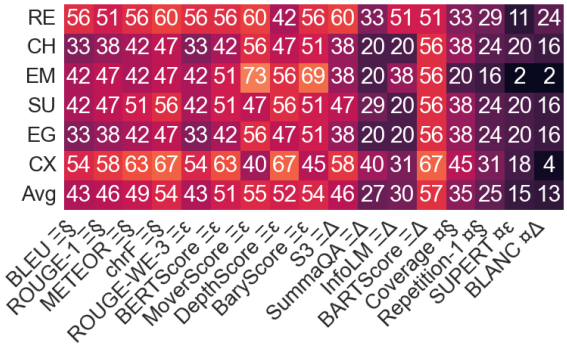

Fig. 4: System-level absolute Kendall correlations (%) between metrics and criteria. Full version: Fig. 15.

Level	Criterion	Metric #1	$\|r\|$ (%)	Metric #2	$\|r\|$ (%)	Metric #3	$\|r\|$ (%)
Story	RE	$\texttt{BARTScore}_2^{¤\Delta}$	42.6	$\texttt{SUPERT}_1^{¤\varepsilon}$	41.2	$\texttt{SUPERT}_2^{¤\varepsilon}$	40.2
	CH	$\texttt{Repetition-3}^{¤\S}$	38.1	$\texttt{BERTScore}_R^{\Xi\varepsilon}$	37.1	$\texttt{S3}^{\Xi\Delta}$	37.1
	EM	$\texttt{S3}^{\Xi\Delta}$	32.8	$\texttt{chrF}^{\Xi\S}$	32.4	$\texttt{BERTScore}_R^{\Xi\varepsilon}$	32.1
	SU	$\texttt{Novelty-1}^{¤\S}$	32.9	$\texttt{chrF}^{\Xi\S}$	32.7	$\texttt{ROUGE-1}^{\Xi\S}$	31.3
	EG	$\texttt{BERTScore}_R^{\Xi\varepsilon}$	43.0	$\texttt{Novelty-1}^{¤\S}$	42.3	$\texttt{chrF}^{\Xi\S}$	41.1
	CX	$\texttt{chrF}^{\Xi\S}$	58.8	$\texttt{BERTScore}_R^{\Xi\varepsilon}$	55.8	$\texttt{ROUGE-1}^{\Xi\S}$	55.0
System	RE	$\texttt{ROUGE-S}\star_F^{\Xi\S}$	80.4	$\texttt{ROUGE-SU}\star_F^{\Xi\S}$	80.3	$\texttt{ROUGE-S}\star_R^{\Xi\S}$	80.2
	CH	$\texttt{BaryScore}_1^{\Xi\varepsilon}$	88.2	$\texttt{BaryScore}_2^{\Xi\varepsilon}$	88.0	$\texttt{BERTScore}_F^{\Xi\varepsilon}$	87.9
	EM	$\texttt{BaryScore}_1^{\Xi\varepsilon}$	90.0	$\texttt{BaryScore}_2^{\Xi\varepsilon}$	90.0	$\texttt{BERTScore}_F^{\Xi\varepsilon}$	88.7
	SU	$\texttt{BARTScore}_1^{\Xi\Delta}$	92.7	$\texttt{BERTScore}_R^{\Xi\varepsilon}$	91.1	$\texttt{DepthScore}^{\Xi\varepsilon}$	90.7
	EG	$\texttt{DepthScore}^{\Xi\varepsilon}$	93.4	$\texttt{BARTScore}_1^{\Xi\Delta}$	92.4	$\texttt{SUPERT}_2^{¤\varepsilon}$	92.2
	CX	$\texttt{DepthScore}^{\Xi\varepsilon}$	95.6	$\texttt{BERTScore}_R^{\Xi\varepsilon}$	95.5	$\texttt{Compression}^{¤\S}$	94.3

Tab. 5: Top 3 metrics per criterion per level (story or system) of absolute Pearson (r) correlation. Indices denote different variants.

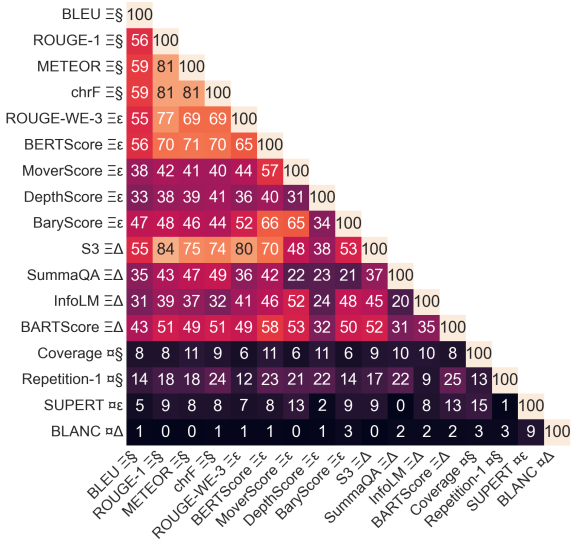

Fig. 5: Story-level absolute Kendall correlations (%) between metrics. Full version: Fig. 20.

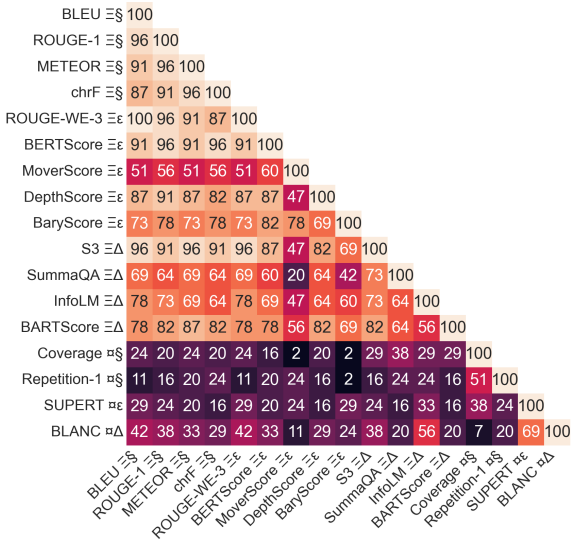

Fig. 6: System-level absolute Kendall correlations (%) between metrics. Full version: Fig. 21.

vations at the story level remain mostly valid, although correlations are overall higher. Reference-based metrics form a large group of very highly correlated metrics, with a majority of correlations surpassing 70%. Embedding- and model-based reference-free metrics remain weakly correlated to other metrics.

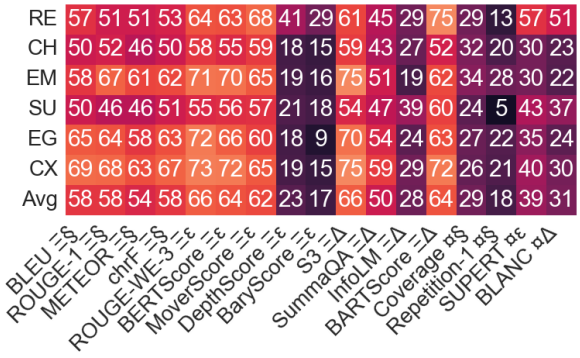

Fig. 7: Weighted macro F1-scores of paired bootstrap resampling. Full version: Fig. 26.

5.3 Fine-grained analysis

Top-k systems (Fig. 8). Here, we evaluate whether automatic metrics can reliably quantify differences between systems of competitive performances. For all criteria except RE and CX, correlations follow a convex curve between $k = 10$ and $k = 4$, suggesting that metrics should not be used to compare systems of high variance in quality unless there are enough of them. Indeed, removing a few systems causes correlations to worsen significantly, until the remaining systems are few enough and of competitive performance. RE correlations interestingly increase as k decreases, which indicates that system quantity is a lesser concern for RE.

Pairwise system comparison (Fig. 7). Here, we evaluate the pairwise discriminative power of automatic metrics. Following Bhandari et al. (2020), we take all system pairs (s_1, s_2) and compute their average ratings per criterion using paired bootstrap resampling (Koehn, 2004; Dror et al., 2018). We assign a label $y_{\text{true}} = 1$ if s_1 is better than s_2 with 95% confidence, $y_{\text{true}} = 2$ if s_2 is better, and $y_{\text{true}} = 0$ if confidence is below 95%. We then repeat the procedure for each metric m, getting $y^{(m)}_{\text{pred}}$ labels, and calculate the weighted macro F1-scores (Goutte and Gaussier, 2005) between y_{true} and $y^{(m)}_{\text{pred}}$ to evaluate if m is a good proxy for human criteria. We observe that reference-based metrics again perform better than reference-free metrics, with $\text{S3}^{\Xi\Delta}$ and $\text{ROUGE-WE-3}^{\Xi\varepsilon}$ at the top. $\text{DepthScore}^{\Xi\varepsilon}$ and $\text{BaryScore}^{\Xi\varepsilon}$ prove to be very unsuited for pairwise system comparisons, despite showing high system-level correlations (see Fig. 3). Finally, SU appears to be the most troublesome criterion for this task, suggesting that the surprise factor is especially difficult to evaluate.

Statistical significance. Using the Williams test (Ap. B), we found that increases in correlation with human criteria between top 3 metrics per criterion (Tab. 5) are not statistically significant, which suggests that best-scoring metrics are of similar performance. However, except for the RE criterion, we notably find that the increases in correlation of $\text{chrF}^{\Xi\S}$ and $\text{BERTScore}^{\Xi\varepsilon}$ compared to $\text{BLEU}^{\Xi\S}$ and $\text{ROUGE}^{\Xi\S}$ variants are statistically significant.

5.4 Aggregated rankings of metrics

To aggregate the scores obtained by the three correlation measures (Kendall, Pearson and Spearman), we use the work of Colombo et al. (2022a)[8], who rely on the Kemeny consensus (Kemeny, 1959; Myerson, 1996) and recommend to use the Borda Count (BC) as an efficient approximation (Sibony, 2014). They experimentally show that Kemeny consensus has more desirable properties than a ranking obtained through a mean-aggregation procedure. We report the results in Tab. 6. To compare system performance, model- or embedding-based metrics (*e.g.*. $\text{BARTScore}^{\Xi\varepsilon}$ or $\text{BERTScore}^{\Xi\varepsilon}$) seem most adapted. However, at the story level, $\text{chrF}^{\Xi\S}$ and $\text{BERTScore}^{\Xi\varepsilon}$ are among the best metrics, while $\text{BLEU}^{\Xi\S}$ is completely absent from the top spots. $\text{ROUGE}^{\Xi\S}$ does appear in the ranking, albeit below $\text{chrF}^{\Xi\S}$.

Level	Metric	BC
Story	$\text{chrF}^{\Xi\S}$	1237
	$\text{S3}^{\Xi\Delta}_{p}$	1198
	$\text{ROUGE-1}^{\Xi\S}$	1186
	$\text{S3}^{\Xi\Delta}_{r}$	1177
	$\text{BERTScore}^{\Xi\varepsilon}_{R}$	1158
System	$\text{BARTScore}^{\Xi\Delta}$	1120
	$\text{BaryScore}^{\Xi\varepsilon}_{5}$	1110
	$\text{BERTScore}^{\Xi\varepsilon}_{F}$	1095
	$\text{MoverScore}^{\Xi\varepsilon}$	1070
	$\text{DepthScore}^{\Xi\varepsilon}$	1069

Tab. 6: Top 5 metrics computed by one-level ranking per aggregation level, higher Borda count is better

6 Conclusions

Our analysis yields the following conclusions:

1. **Large pre-trained language models seem to produce the best results for ASG.** Our benchmark shows that GPT-2 performed better than systems specifically tailored for ASG despite being older

[8] https://github.com/PierreColombo/RankingNLPSystems

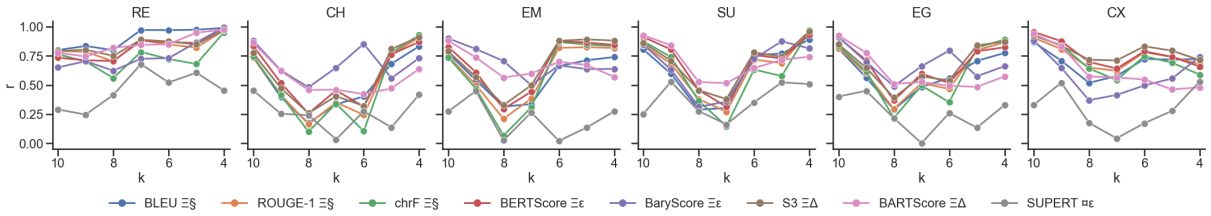

Fig. 8: System-level absolute Pearson correlations (%) between automatic metrics and our proposed human criteria on top-k systems

than some of them. Overall, all systems remain significantly inferior to human output, illustrating that ASG remains a challenging task for current language models.

2. **Stronger metrics, tailored explicitly for specific criteria of ASG, are desperately needed.** The weak correlations of automatic metrics with human criteria still leave much to be desired. Ideally, we would have automatic metrics which reflect each of our proposed criteria.

3. **Awaiting specific ASG metrics, researchers should use better metrics than BLEU$^{\Xi\S}$ and ROUGE$^{\Xi\S}$.** chrF$^{\Xi\S}$ and BARTScore$^{\Xi\varepsilon}$ are the best performers at the story- and system-level respectively. Given the overall weak results, however, we strongly advise to rely on human annotations for ASG evaluation.

4. **Our new set of human criteria allows for a standardized and extensive human evaluation.** The criteria are overall weakly correlated with one another, which shows that they are non-redundant, and produce coherent system rankings.

Future directions. Motivated by our search for human criteria from the social science literature, we reckon more collaboration between the NLP and social science communities may yield valuable insights into the question of how to computationally capture good indicators of story quality. In this spirit, we hope that HANNA will pave the way for further progress in the evaluation of ASG.

Acknowledgements

We thank Yejin Choi, Richard Bai, Le Fang, Jian Guan, Hannah Rashkin, David Wilmot and Eden Bensaid for answering to our requests for data.
This work was granted access to the HPC resources of IDRIS under the allocation 2022-101838 made by GENCI and was partially funded by the grant ANR-20-CHIA-0012-01 (“NoRDF”).

References

Amal Alabdulkarim, Siyan Li, and Xiangyu Peng. 2021. Automatic story generation: Challenges and attempts. In *Proceedings of the Third Workshop on Narrative Understanding*, pages 72–83, Virtual. Association for Computational Linguistics.

Arwa I Alhussain and Aqil M Azmi. 2021. Automatic story generation: a survey of approaches. *ACM Computing Surveys (CSUR)*, 54(5):1–38.

Prithviraj Ammanabrolu, Wesley Cheung, Dan Tu, William Broniec, and Mark O. Riedl. 2020. Bringing stories alive: Generating interactive fiction worlds. *CoRR*, abs/2001.10161.

Ron Artstein and Massimo Poesio. 2008. Inter-coder agreement for computational linguistics. *Computational linguistics*, 34(4):555–596.

Ruth Aylett, Marco Vala, Pedro Sequeira, and Ana Paiva. 2007. Fearnot!–an emergent narrative approach to virtual dramas for anti-bullying education. In *International Conference on Virtual Storytelling*, pages 202–205. Springer.

Byung-Chull Bae, Suji Jang, Youngjune Kim, and Seyoung Park. 2021. A preliminary survey on story interestingness: Focusing on cognitive and emotional interest. In *International Conference on Interactive Digital Storytelling*, pages 447–453. Springer.

He Bai, Peng Shi, Jimmy Lin, Luchen Tan, Kun Xiong, Wen Gao, Jie Liu, and Ming Li. 2021. Semantics of the unwritten: The effect of end of paragraph and sequence tokens on text generation with GPT2. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing: Student Research Workshop*, pages 148–162, Online. Association for Computational Linguistics.

Satanjeev Banerjee and Alon Lavie. 2005. METEOR: An automatic metric for MT evaluation with improved correlation with human judgments. In *Proceedings of the ACL Workshop on Intrinsic and Extrinsic Evaluation Measures for Machine Translation and/or Summarization*, pages 65–72, Ann Arbor, Michigan. Association for Computational Linguistics.

Morteza Behrooz, Justus Robertson, and Arnav Jhala. 2019. Story quality as a matter of perception: using word embeddings to estimate cognitive interest. In *Proceedings of the AAAI Conference on Artificial Intelligence and Interactive Digital Entertainment*, volume 15, pages 3–9.

Manik Bhandari, Pranav Narayan Gour, Atabak Ashfaq, Pengfei Liu, and Graham Neubig. 2020. Re-evaluating evaluation in text summarization. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 9347–9359, Online. Association for Computational Linguistics.

Faeze Brahman and Snigdha Chaturvedi. 2020. Modeling protagonist emotions for emotion-aware storytelling. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 5277–5294, Online. Association for Computational Linguistics.

Tom B. Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel M. Ziegler, Jeffrey Wu, Clemens Winter, Christopher Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. 2020. Language models are few-shot learners. In *Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020, December 6-12, 2020, virtual*.

Chris Callison-Burch, Miles Osborne, and Philipp Koehn. 2006. Re-evaluating the role of Bleu in machine translation research. In *11th Conference of the European Chapter of the Association for Computational Linguistics*, Trento, Italy. Association for Computational Linguistics.

Marc Cavazza and David Pizzi. 2006. Narratology for interactive storytelling: A critical introduction. In *International Conference on Technologies for Interactive Digital Storytelling and Entertainment*, pages 72–83. Springer.

Emile Chapuis, Pierre Colombo, Matteo Manica, Matthieu Labeau, and Chloé Clavel. 2020. Hierarchical pre-training for sequence labelling in spoken dialog. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 2636–2648, Online. Association for Computational Linguistics.

Anthony Chen, Gabriel Stanovsky, Sameer Singh, and Matt Gardner. 2019. Evaluating question answering evaluation. In *Proceedings of the 2nd Workshop on Machine Reading for Question Answering*, pages 119–124, Hong Kong, China. Association for Computational Linguistics.

Pierre Colombo. 2021. *Learning to represent and generate text using information measures*. Ph.D. thesis, Institut polytechnique de Paris.

Pierre Colombo, Emile Chapuis, Matthieu Labeau, and Chloé Clavel. 2021a. Code-switched inspired losses for spoken dialog representations. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 8320–8337.

Pierre Colombo, Emile Chapuis, Matteo Manica, Emmanuel Vignon, Giovanna Varni, and Chloé Clavel. 2020. Guiding attention in sequence-to-sequence models for dialogue act prediction. In *The Thirty-Fourth AAAI Conference on Artificial Intelligence, AAAI 2020, The Thirty-Second Innovative Applications of Artificial Intelligence Conference, IAAI 2020, The Tenth AAAI Symposium on Educational Advances in Artificial Intelligence, EAAI 2020, New York, NY, USA, February 7-12, 2020*, pages 7594–7601. AAAI Press.

Pierre Colombo, Chloé Clavel, Chouchang Yack, and Giovanna Varni. 2021b. Beam search with bidirectional strategies for neural response generation. In *Proceedings of The Fourth International Conference on Natural Language and Speech Processing (ICNLSP 2021)*, pages 139–146, Trento, Italy. Association for Computational Linguistics.

Pierre Colombo, Nathan Noiry, Ekhine Irurozki, and Stephan Clemencon. 2022a. What are the best systems? new perspectives on nlp benchmarking. *arXiv preprint arXiv:2202.03799*.

Pierre Colombo, Pablo Piantanida, and Chloé Clavel. 2021c. A novel estimator of mutual information for learning to disentangle textual representations. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 6539–6550, Online. Association for Computational Linguistics.

Pierre Colombo, Guillaume Staerman, Chloé Clavel, and Pablo Piantanida. 2021d. Automatic text evaluation through the lens of Wasserstein barycenters. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 10450–10466, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Pierre Colombo, Guillaume Staerman, Nathan Noiry, and Pablo Piantanida. 2022b. Learning disentangled textual representations via statistical measures of similarity. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 2614–2630, Dublin, Ireland. Association for Computational Linguistics.

Pierre Colombo, Wojciech Witon, Ashutosh Modi, James Kennedy, and Mubbasir Kapadia. 2019. Affect-driven dialog generation. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics:*

Human Language Technologies, Volume 1 (Long and Short Papers), pages 3734–3743, Minneapolis, Minnesota. Association for Computational Linguistics.

Pierre Jean A Colombo, Chloé Clavel, and Pablo Piantanida. 2022c. Infolm: A new metric to evaluate summarization & data2text generation. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 36, pages 10554–10562.

Richard Craggs and Mary McGee Wood. 2005. Evaluating Discourse and Dialogue Coding Schemes. *Computational Linguistics*, 31(3):289–296.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, Minneapolis, Minnesota. Association for Computational Linguistics.

Bhuwan Dhingra, Manaal Faruqui, Ankur Parikh, Ming-Wei Chang, Dipanjan Das, and William Cohen. 2019. Handling divergent reference texts when evaluating table-to-text generation. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 4884–4895, Florence, Italy. Association for Computational Linguistics.

Robert Dickman. 2003. The four elements of every successful story. *Reflections - Society for Organizational Learning*, 4(3):51–58.

Rotem Dror, Gili Baumer, Segev Shlomov, and Roi Reichart. 2018. The hitchhiker's guide to testing statistical significance in natural language processing. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1383–1392, Melbourne, Australia. Association for Computational Linguistics.

Desmond Elliott and Frank Keller. 2014. Comparing automatic evaluation measures for image description. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 452–457, Baltimore, Maryland. Association for Computational Linguistics.

Alexander R Fabbri, Wojciech Kryściński, Bryan McCann, Caiming Xiong, Richard Socher, and Dragomir Radev. 2021. Summeval: Re-evaluating summarization evaluation. *Transactions of the Association for Computational Linguistics*, 9:391–409.

Angela Fan, Mike Lewis, and Yann Dauphin. 2018. Hierarchical neural story generation. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 889–898, Melbourne, Australia. Association for Computational Linguistics.

Le Fang, Tao Zeng, Chaochun Liu, Liefeng Bo, Wen Dong, and Changyou Chen. 2021. Transformer-based conditional variational autoencoder for controllable story generation. *arXiv preprint arXiv:2101.00828*.

Yang Gao, Wei Zhao, and Steffen Eger. 2020. SUPERT: Towards new frontiers in unsupervised evaluation metrics for multi-document summarization. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 1347–1354, Online. Association for Computational Linguistics.

Alexandre Garcia, Pierre Colombo, Florence d'Alché Buc, Slim Essid, and Chloé Clavel. 2019. From the token to the review: A hierarchical multimodal approach to opinion mining. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 5539–5548, Hong Kong, China. Association for Computational Linguistics.

Seraphina Goldfarb-Tarrant, Tuhin Chakrabarty, Ralph Weischedel, and Nanyun Peng. 2020. Content planning for neural story generation with aristotelian rescoring. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 4319–4338, Online. Association for Computational Linguistics.

Cyril Goutte and Eric Gaussier. 2005. A probabilistic interpretation of precision, recall and f-score, with implication for evaluation. In *European conference on information retrieval*, pages 345–359. Springer.

Yvette Graham and Timothy Baldwin. 2014. Testing for significance of increased correlation with human judgment. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 172–176, Doha, Qatar. Association for Computational Linguistics.

Yvette Graham, Timothy Baldwin, Alistair Moffat, and Justin Zobel. 2017. Can machine translation systems be evaluated by the crowd alone. *Natural Language Engineering*, 23(1):3.

Max Grusky, Mor Naaman, and Yoav Artzi. 2018. Newsroom: A dataset of 1.3 million summaries with diverse extractive strategies. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 708–719, New Orleans, Louisiana. Association for Computational Linguistics.

Jian Guan, Fei Huang, Zhihao Zhao, Xiaoyan Zhu, and Minlie Huang. 2020. A knowledge-enhanced pretraining model for commonsense story generation. *Transactions of the Association for Computational Linguistics*, 8:93–108.

Jian Guan, Xiaoxi Mao, Changjie Fan, Zitao Liu, Wenbiao Ding, and Minlie Huang. 2021a. Long text generation by modeling sentence-level and discourse-level coherence. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 6379–6393, Online. Association for Computational Linguistics.

Jian Guan, Yansen Wang, and Minlie Huang. 2019. Story ending generation with incremental encoding and commonsense knowledge. In *The Thirty-Third AAAI Conference on Artificial Intelligence, AAAI 2019, The Thirty-First Innovative Applications of Artificial Intelligence Conference, IAAI 2019, The Ninth AAAI Symposium on Educational Advances in Artificial Intelligence, EAAI 2019, Honolulu, Hawaii, USA, January 27 - February 1, 2019*, pages 6473–6480. AAAI Press.

Jian Guan, Zhexin Zhang, Zhuoer Feng, Zitao Liu, Wenbiao Ding, Xiaoxi Mao, Changjie Fan, and Minlie Huang. 2021b. OpenMEVA: A benchmark for evaluating open-ended story generation metrics. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 6394–6407, Online. Association for Computational Linguistics.

Ivan Habernal and Iryna Gurevych. 2017. Argumentation mining in user-generated web discourse. *Computational Linguistics*, 43(1):125–179.

Kevin A Hallgren. 2012. Computing inter-rater reliability for observational data: an overview and tutorial. *Tutorials in quantitative methods for psychology*, 8(1):23.

Ken Hartsook, Alexander Zook, Sauvik Das, and Mark O Riedl. 2011. Toward supporting stories with procedurally generated game worlds. In *2011 IEEE Conference on Computational Intelligence and Games (CIG'11)*, pages 297–304. IEEE.

Ting-Hao Kenneth Huang, Francis Ferraro, Nasrin Mostafazadeh, Ishan Misra, Aishwarya Agrawal, Jacob Devlin, Ross Girshick, Xiaodong He, Pushmeet Kohli, Dhruv Batra, C. Lawrence Zitnick, Devi Parikh, Lucy Vanderwende, Michel Galley, and Margaret Mitchell. 2016. Visual storytelling. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 1233–1239, San Diego, California. Association for Computational Linguistics.

Asghar Iran-Nejad. 1987. Cognitive and affective causes of interest and liking. *Journal of Educational Psychology*, 79(2):120.

Hamid Jalalzai, Pierre Colombo, Chloé Clavel, Éric Gaussier, Giovanna Varni, Emmanuel Vignon, and Anne Sabourin. 2020. Heavy-tailed representations, text polarity classification & data augmentation. In *Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020, December 6-12, 2020, virtual*.

Harsh Jhamtani and Taylor Berg-Kirkpatrick. 2020. Narrative text generation with a latent discrete plan. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 3637–3650, Online. Association for Computational Linguistics.

Marzena Karpinska, Nader Akoury, and Mohit Iyyer. 2021. The perils of using Mechanical Turk to evaluate open-ended text generation. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 1265–1285, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Suzanne Keen et al. 2007. *Empathy and the Novel*. Oxford University Press on Demand.

John G Kemeny. 1959. Mathematics without numbers. *Daedalus*, 88(4):577–591.

Maurice G Kendall. 1938. A new measure of rank correlation. *Biometrika*, 30(1/2):81–93.

Nitish Shirish Keskar, Bryan McCann, Lav R Varshney, Caiming Xiong, and Richard Socher. 2019. Ctrl: A conditional transformer language model for controllable generation. *arXiv preprint arXiv:1909.05858*.

Philipp Koehn. 2004. Statistical significance tests for machine translation evaluation. In *Proceedings of the 2004 Conference on Empirical Methods in Natural Language Processing*, pages 388–395, Barcelona, Spain. Association for Computational Linguistics.

J Richard Landis and Gary G Koch. 1977. The measurement of observer agreement for categorical data. *biometrics*, pages 159–174.

Michael Lebowitz. 1985. Story-telling as planning and learning. *Poetics*, 14(6):483–502.

Chin-Yew Lin. 2004. ROUGE: A package for automatic evaluation of summaries. In *Text Summarization Branches Out*, pages 74–81, Barcelona, Spain. Association for Computational Linguistics.

Chia-Wei Liu, Ryan Lowe, Iulian Serban, Mike Noseworthy, Laurent Charlin, and Joelle Pineau. 2016. How NOT to evaluate your dialogue system: An empirical study of unsupervised evaluation metrics for dialogue response generation. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 2122–2132, Austin, Texas. Association for Computational Linguistics.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Roberta: A robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692*.

Qingsong Ma, Johnny Wei, Ondřej Bojar, and Yvette Graham. 2019. Results of the WMT19 metrics shared task: Segment-level and strong MT systems pose big challenges. In *Proceedings of the Fourth Conference on Machine Translation (Volume 2: Shared Task Papers, Day 1)*, pages 62–90, Florence, Italy. Association for Computational Linguistics.

Nitika Mathur, Timothy Baldwin, and Trevor Cohn. 2020. Tangled up in BLEU: Reevaluating the evaluation of automatic machine translation evaluation metrics. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 4984–4997, Online. Association for Computational Linguistics.

Allyssa McCabe and Carole Peterson. 1984. What makes a good story. *Journal of Psycholinguistic Research*, 13(6):457–480.

I. Dan Melamed, Ryan Green, and Joseph P. Turian. 2003. Precision and recall of machine translation. In *Companion Volume of the Proceedings of HLT-NAACL 2003 - Short Papers*, pages 61–63.

Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013a. Efficient estimation of word representations in vector space. *arXiv preprint arXiv:1301.3781*.

Tomás Mikolov, Ilya Sutskever, Kai Chen, Gregory S. Corrado, and Jeffrey Dean. 2013b. Distributed representations of words and phrases and their compositionality. In *Advances in Neural Information Processing Systems 26: 27th Annual Conference on Neural Information Processing Systems 2013. Proceedings of a meeting held December 5-8, 2013, Lake Tahoe, Nevada, United States*, pages 3111–3119.

Jihyung Moon. 2019. Significance test of increase in correlation for nlp evaluations in python. https://github.com/inmoonlight/nlp-williams/.

Nasrin Mostafazadeh, Nathanael Chambers, Xiaodong He, Devi Parikh, Dhruv Batra, Lucy Vanderwende, Pushmeet Kohli, and James Allen. 2016. A corpus and cloze evaluation for deeper understanding of commonsense stories. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 839–849, San Diego, California. Association for Computational Linguistics.

Roger B Myerson. 1996. Fundamentals of social choice theory. Technical report, Discussion Paper.

Preksha Nema and Mitesh M. Khapra. 2018. Towards a better metric for evaluating question generation systems. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 3950–3959, Brussels, Belgium. Association for Computational Linguistics.

Jun-Ping Ng and Viktoria Abrecht. 2015. Better summarization evaluation with word embeddings for ROUGE. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 1925–1930, Lisbon, Portugal. Association for Computational Linguistics.

Jekaterina Novikova, Ondřej Dušek, Amanda Cercas Curry, and Verena Rieser. 2017. Why we need new evaluation metrics for NLG. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 2241–2252, Copenhagen, Denmark. Association for Computational Linguistics.

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. Bleu: a method for automatic evaluation of machine translation. In *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics*, pages 311–318, Philadelphia, Pennsylvania, USA. Association for Computational Linguistics.

Damian Pascual, Beni Egressy, Clara Meister, Ryan Cotterell, and Roger Wattenhofer. 2021. A plug-and-play method for controlled text generation. In *Findings of the Association for Computational Linguistics: EMNLP 2021*, pages 3973–3997, Punta Cana, Dominican Republic. Association for Computational Linguistics.

Nanyun Peng, Marjan Ghazvininejad, Jonathan May, and Kevin Knight. 2018. Towards controllable story generation. In *Proceedings of the First Workshop on Storytelling*, pages 43–49, New Orleans, Louisiana. Association for Computational Linguistics.

Maxime Peyrard, Teresa Botschen, and Iryna Gurevych. 2017. Learning to score system summaries for better content selection evaluation. In *Proceedings of the Workshop on New Frontiers in Summarization*, pages 74–84, Copenhagen, Denmark. Association for Computational Linguistics.

Georg Pichler, Pierre Jean A Colombo, Malik Boudiaf, Günther Koliander, and Pablo Piantanida. 2022. A differential entropy estimator for training neural networks. In *International Conference on Machine Learning*, pages 17691–17715. PMLR.

Maja Popović. 2015. chrF: character n-gram F-score for automatic MT evaluation. In *Proceedings of the Tenth Workshop on Statistical Machine Translation*, pages 392–395, Lisbon, Portugal. Association for Computational Linguistics.

Alec Radford, Karthik Narasimhan, Tim Salimans, and Ilya Sutskever. 2018. Improving language understanding by generative pre-training.

Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, Ilya Sutskever, et al. 2019. Language models are unsupervised multitask learners. *OpenAI blog*, 1(8):9.

William Lowell Randall. 1999. Narrative intelligence and the novelty of our lives. *Journal of aging Studies*, 13(1):11–28.

Hannah Rashkin, Asli Celikyilmaz, Yejin Choi, and Jianfeng Gao. 2020. PlotMachines: Outline-conditioned generation with dynamic plot state tracking. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 4274–4295, Online. Association for Computational Linguistics.

Alan Ritter, Colin Cherry, and William B. Dolan. 2011. Data-driven response generation in social media. In *Proceedings of the 2011 Conference on Empirical Methods in Natural Language Processing*, pages 583–593, Edinburgh, Scotland, UK. Association for Computational Linguistics.

Hanna-Riikka Roine. 2016. Imaginative, immersive and interactive engagements. the rhetoric of worldbuilding in contemporary speculative fiction.

Sascha Rothe, Shashi Narayan, and Aliaksei Severyn. 2020. Leveraging pre-trained checkpoints for sequence generation tasks. *Transactions of the Association for Computational Linguistics*, 8:264–280.

Roger C Schank. 1978. Interestingness: Controlling inferences. Technical report, YALE UNIV NEW HAVEN CONN DEPT OF COMPUTER SCIENCE.

Thomas Scialom, Sylvain Lamprier, Benjamin Piwowarski, and Jacopo Staiano. 2019. Answers unite! unsupervised metrics for reinforced summarization models. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 3246–3256, Hong Kong, China. Association for Computational Linguistics.

Ori Shapira, David Gabay, Yang Gao, Hadar Ronen, Ramakanth Pasunuru, Mohit Bansal, Yael Amsterdamer, and Ido Dagan. 2019. Crowdsourcing lightweight pyramids for manual summary evaluation. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 682–687, Minneapolis, Minnesota. Association for Computational Linguistics.

Eric Sibony. 2014. Borda count approximation of kemeny's rule and pairwise voting inconsistencies. In *Proceedings of the NIPS 2014 Workshop on Analysis of Rank Data*.

Wilbert Spooren and Liesbeth Degand. 2010. Coding coherence relations: Reliability and validity. *Corpus Linguistics and Linguistic Theory*, 6(2):241–266.

Guillaume Staerman, Pavlo Mozharovskyi, Stéphan Clémençon, and Florence d'Alché Buc. 2021. A pseudo-metric between probability distributions based on depth-trimmed regions. *arXiv preprint arXiv:2103.12711*.

James H Steiger. 1980. Tests for comparing elements of a correlation matrix. *Psychological bulletin*, 87(2):245.

Amanda Stent, Matthew Marge, and Mohit Singhai. 2005. Evaluating evaluation methods for generation in the presence of variation. In *international conference on intelligent text processing and computational linguistics*, pages 341–351. Springer.

Wendy A Suzuki, Mónica I Feliú-Mójer, Uri Hasson, Rachel Yehuda, and Jean Mary Zarate. 2018. Dialogues: The science and power of storytelling. *Journal of Neuroscience*, 38(44):9468–9470.

Michael Toolan. 2012. Engagement via emotional heightening in" passion": On the grammatical texture of emotionally-immersive passages in short fiction. *Narrative*, 20(2):210–225.

Chris van der Lee, Albert Gatt, Emiel van Miltenburg, Sander Wubben, and Emiel Krahmer. 2019. Best practices for the human evaluation of automatically generated text. In *Proceedings of the 12th International Conference on Natural Language Generation*, pages 355–368, Tokyo, Japan. Association for Computational Linguistics.

Oleg Vasilyev, Vedant Dharnidharka, and John Bohannon. 2020. Fill in the BLANC: Human-free quality estimation of document summaries. In *Proceedings of the First Workshop on Evaluation and Comparison of NLP Systems*, pages 11–20, Online. Association for Computational Linguistics.

Ramakrishna Vedantam, C. Lawrence Zitnick, and Devi Parikh. 2015. Cider: Consensus-based image description evaluation. In *IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2015, Boston, MA, USA, June 7-12, 2015*, pages 4566–4575. IEEE Computer Society.

Su Wang, Greg Durrett, and Katrin Erk. 2020. Narrative interpolation for generating and understanding stories. *arXiv preprint arXiv:2008.07466*.

Evan J. Williams. 1959. *Regression Analysis*. Wiley, New York.

David Wilmot and Frank Keller. 2021. A temporal variational model for story generation. *arXiv preprint arXiv:2109.06807*.

Wojciech Witon, Pierre Colombo, Ashutosh Modi, and Mubbasir Kapadia. 2018. Disney at IEST 2018: Predicting emotions using an ensemble. In *Proceedings of the 9th Workshop on Computational Approaches to Subjectivity, Sentiment and Social Media Analysis*, pages 248–253, Brussels, Belgium. Association for Computational Linguistics.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Remi Louf, Morgan Funtowicz, Joe Davison, Sam Shleifer, Patrick von Platen, Clara Ma, Yacine Jernite, Julien Plu, Canwen Xu,

Teven Le Scao, Sylvain Gugger, Mariama Drame, Quentin Lhoest, and Alexander Rush. 2020. Transformers: State-of-the-art natural language processing. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 38–45, Online. Association for Computational Linguistics.

Jingjing Xu, Xuancheng Ren, Yi Zhang, Qi Zeng, Xiaoyan Cai, and Xu Sun. 2018. A skeleton-based model for promoting coherence among sentences in narrative story generation. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 4306–4315, Brussels, Belgium. Association for Computational Linguistics.

Zhilin Yang, Zihang Dai, Yiming Yang, Jaime G. Carbonell, Ruslan Salakhutdinov, and Quoc V. Le. 2019. Xlnet: Generalized autoregressive pretraining for language understanding. In *Advances in Neural Information Processing Systems 32: Annual Conference on Neural Information Processing Systems 2019, NeurIPS 2019, December 8-14, 2019, Vancouver, BC, Canada*, pages 5754–5764.

Lili Yao, Nanyun Peng, Ralph Weischedel, Kevin Knight, Dongyan Zhao, and Rui Yan. 2019. Plan-and-write: Towards better automatic storytelling. *Proceedings of the AAAI Conference on Artificial Intelligence*, 33(01):7378–7385.

Weizhe Yuan, Graham Neubig, and Pengfei Liu. 2021. Bartscore: Evaluating generated text as text generation. *Advances in Neural Information Processing Systems*, 34.

Tianyi Zhang, Varsha Kishore, Felix Wu, Kilian Q. Weinberger, and Yoav Artzi. 2020. Bertscore: Evaluating text generation with BERT. In *8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, April 26-30, 2020*. OpenReview.net.

Ying Zhang, Stephan Vogel, and Alex Waibel. 2004. Interpreting BLEU/NIST scores: How much improvement do we need to have a better system? In *Proceedings of the Fourth International Conference on Language Resources and Evaluation (LREC'04)*, Lisbon, Portugal. European Language Resources Association (ELRA).

Wei Zhao, Maxime Peyrard, Fei Liu, Yang Gao, Christian M. Meyer, and Steffen Eger. 2019. MoverScore: Text generation evaluating with contextualized embeddings and earth mover distance. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 563–578, Hong Kong, China. Association for Computational Linguistics.

Appendix

Table of Contents

A Amazon Mechanical Turk experiment details **16**

B Names of metric variants **16**

C Distributions of human annotations per system **19**

D Correlations between human criteria **20**

E Correlations between human criteria and automatic metrics **21**

F Correlations between automatic metrics **23**

G Best metrics per criterion per level of correlation per correlation coefficient **29**

H Weighted macro F1-scores between automatic metrics and human criteria **31**

I Williams tests between automatic metrics **32**

A Amazon Mechanical Turk experiment details

To complement section 3.5, the details of the instructions we gave in our Amazon Mechanical Turk experiment can be found in Tab. 7 below.

B Names of metric variants

Here we define the names we give to some variants of the automatic metrics we used.
`SUPERT` and `BLANC` are summarization metrics which normally require a source document and a summary. In our setting, we have a prompt and a generated story. The suffix `PS` means we used the "Prompt as the Summary", and `SS` means the "Story as the Summary". The `Golden` suffix means we used the reference human story as the source document and the generated story as the summary.
Given a couple of texts (x, y), `BARTScore` computes a score based on the log probability of y given x. We used the suffixes `SH` for (Story, Human), `HS` for (Human, Story), `SP` for (Story, Prompt) and `PS` for (Prompt, Story).
All other names are defined in their respective papers.

Amazon Mechanical Turk example task

Please read the prompt, the human story and the subject story (both stories might be the same). The story you will have to rate is the **subject story**.
Important: we will reject HITs which were done in **fewer than 30 seconds** (unless both stories are exceptionally short). Please rest assured: if you take the work seriously, we have no reason to reject it.
Disclaimer: some stories have been automatically generated and might contain explicit or offensive content.
Note: some stories have been abruptly cut in the middle of a sentence. Please rate them as if they ended just before the unfinished sentence.
Note: if the story is not relevant with respect to the prompt, it **only** affects the **Relevance** criterion! Do not rate 1 everywhere, or we will **reject**!
Then, please write down the name of the **first character** that is mentioned in the **subject story**; if no name is mentioned, write "None". Only proper nouns count as names.
Then, please rate the **subject story** on a scale from 1 (worst) to 5 (best) on the following criteria: relevance, coherence, empathy, surprise, engagement, and complexity.

Prompt	When you die the afterlife is an arena where you face every insect and animal you killed in your life. If you win you go to heaven, lose you go to hell. Your job was an exterminator on earth.
Human story	3,000 years have I been fighting. Every morning, the raccoons scratch at my eyes. Every evening, the skunks spray me while the opossums chew at my feet. I have never had any tools. I have only my hands. I don't remember the place I came from before this. All I remember is the daily fight between me and these animals. No matter how many times I kill them, they come back the next day. [...]
Subject story	First of all, not everyone was entitled to be an exterminator. But the ones that were – maybe were, like, *genius*, because, yes, I had once belonged to a less fortunate class of people – had all the opportunity to work for the damn plant killer, and it's hard work. And the horrifying truth is, once you die, and the entire planet turns into a glade that contains a golden fish that would've been crushed by a million million goldfish just moments ago, you're not really good enough for heaven. Why? [...]
Name of the first mentioned character in the subject story	`[Area to fill]`
Relevance — measures how well the story matches its prompt	1 — The story has no relationship with the prompt at all. 2 — The story only has a weak relationship with the prompt. 3 — The story roughly matches the prompt. 4 — The story matches the prompt, except for one or two small aspects. 5 — The story matches the prompt exactly.

Coherence — measures whether the story makes sense	1 — The story does not make sense at all. For instance, the setting and/or characters keep changing, and/or there is no understandable plot. 2 — Most of the story does not make sense. 3 — The story mostly makes sense but has some incoherences. 4 — The story almost makes sense overall, except for one or two small incoherences. 5 — The story makes sense from beginning to end.
Empathy — measures how well you understood the characters' emotions (regardless of whether you agreed with them)	1 — The characters seemed apathetic to you. 2 — At least one character slightly related to you on an emotional level. 3 — You recognized specific, but not necessarily strong, emotions (eg sadness, joy, fear...) in at least one character. 4 — At least one character emotionally involved you, but minor details prevented you from completely relating to them. 5 — At least one character completely involved you on an emotional level.
Surprise — measures how surprising the end of the story was	1 — The ending seemed completely obvious from the start, or doesn't make any sense at all. 2 — The ending was easily predictable after a few sentences. 3 — The ending was predictable after half of the story. 4 — The ending surprised you, but would have been difficult to predict. 5 — The ending surprised you, and still seemed as if it could very reasonably have been predicted, ie, there were enough clues in the story.
Engagement — measures how much you engaged with the story	1 — You found the story boring and were glad it was over. 2 — You found one or two things interesting in the story, but no more. 3 — The story was mildly interesting. 4 — The story almost kept you engaged until the end. 5 — You were so engaged that you wished there was a sequel.
Complexity — measures how elaborate the story is	1 — The setting of the story is extremely simple; it only involves one or two characters or concepts. 2 — The setting of the story is simple; one or two characters, a simple plot, maybe an indication of time or location. 3 — The story is somewhat developed: it involves at least one of the following: complex concepts, realistic characters, an intricate plot, an underlying history or circumstances, precise descriptions. 4 — The story is developed: it involves at least two of the following: complex concepts, realistic characters, an intricate plot, an underlying history or circumstances, precise descriptions. 5 — The story is well thought-out: it involves at least three of the following: complex concepts, realistic characters, an intricate plot, an underlying history or circumstances, precise descriptions.

Tab. 7: Example task from our Amazon Mechanical Turk experiment

C Distributions of human annotations per system

Here we report the violin plots of the distributions of human annotations per system. Human output scores visibly better than language models. Note that for our generation, we do not use beam search (Colombo et al., 2021b, 2020; Pichler et al., 2022; Colombo, 2021; Colombo et al., 2022b; Garcia et al., 2019; Colombo et al., 2021c). To further improve the generation a domain pre-trained language model could be considered (Chapuis et al., 2020; Colombo et al., 2021a).

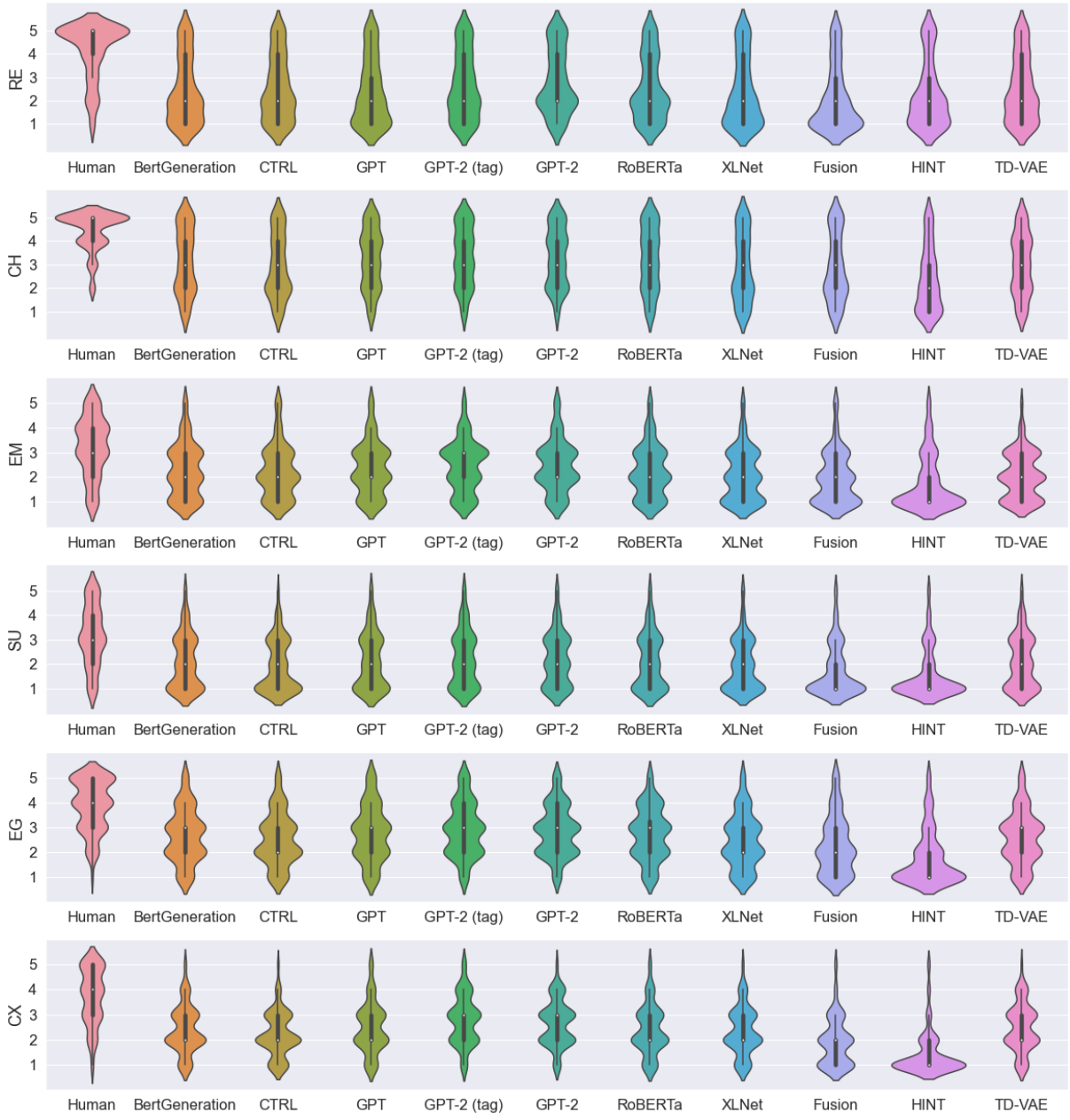

Fig. 9: Violin plots of the distributions of human annotations per system

D Correlations between human criteria

Here we report the story-level and system-level absolute correlations between human criteria with Spearman's ρ (Fig. 10 and Fig. 11) and Pearson's r (Fig. 12 and Fig. 13).

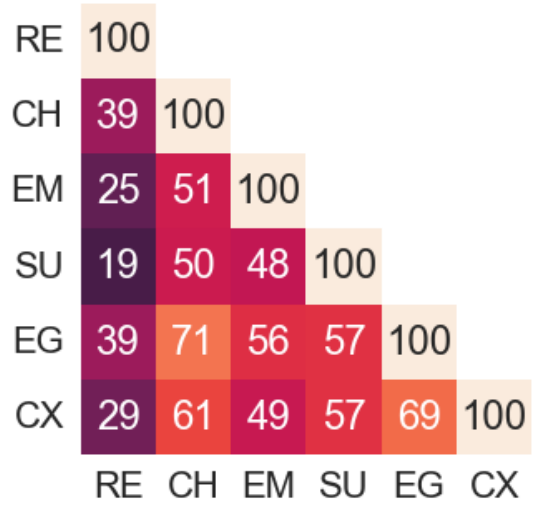

Fig. 10: Story-level Spearman correlations (%) between our proposed human criteria

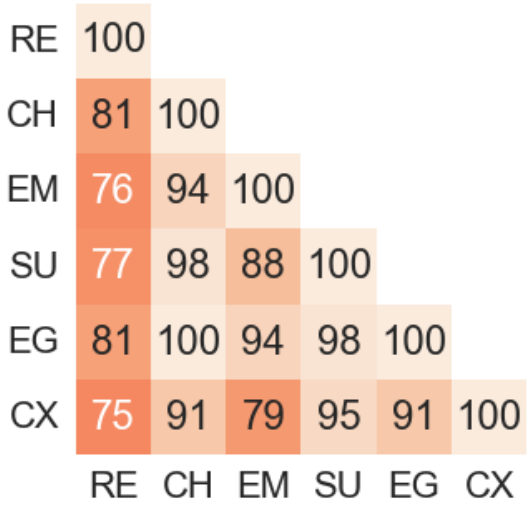

Fig. 11: System-level Spearman correlations (%) between our proposed human criteria

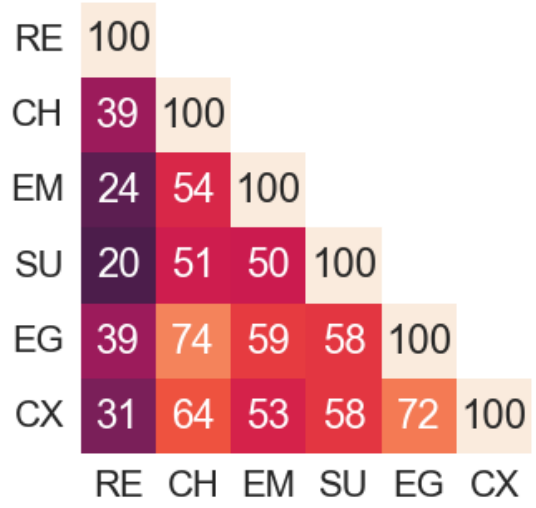

Fig. 12: Story-level Pearson correlations (%) between our proposed human criteria

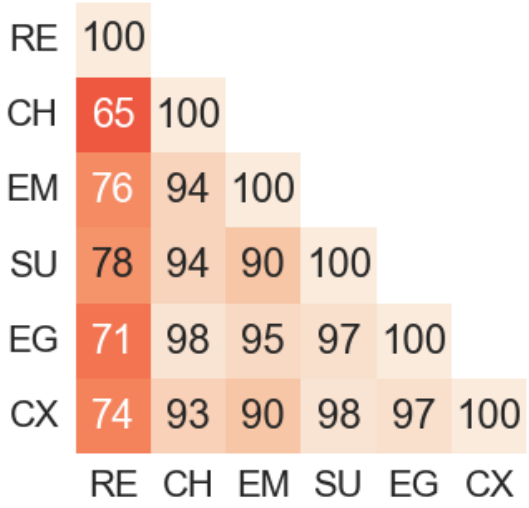

Fig. 13: System-level Pearson correlations (%) between our proposed human criteria

E Correlations between human criteria and automatic metrics

Here we report the full figures of story-level and system-level absolute correlations between human criteria and automatic metrics with all three correlation coefficients.

Fig. 14: Story-level absolute Kendall correlations (%) between automatic metrics and our proposed human criteria

Fig. 15: System-level absolute Kendall correlations (%) between automatic metrics and our proposed human criteria

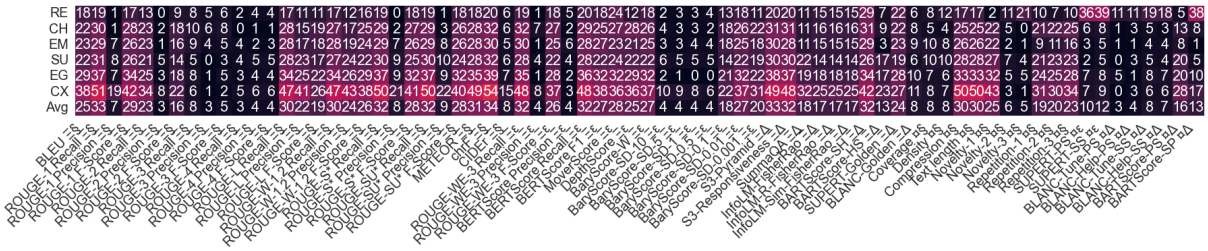

Fig. 16: Story-level absolute Spearman correlations (%) between automatic metrics and our proposed human criteria

Fig. 17: System-level absolute Spearman correlations (%) between automatic metrics and our proposed human criteria

Fig. 18: Story-level absolute Pearson correlations (%) between automatic metrics and our proposed human criteria

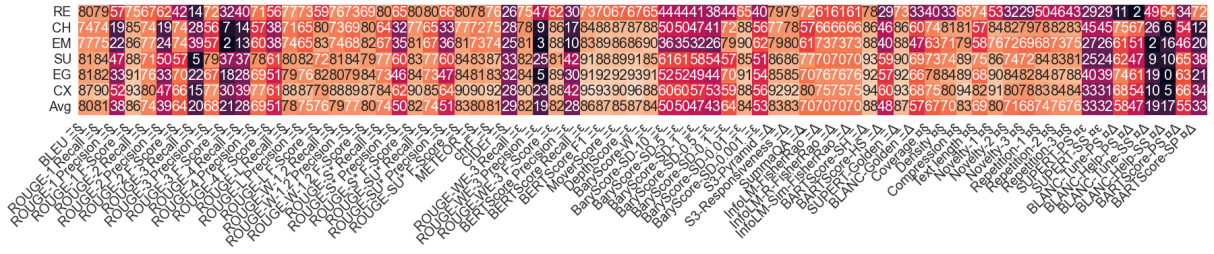

Fig. 19: System-level absolute Pearson correlations (%) between automatic metrics and our proposed human criteria

F Correlations between automatic metrics

Here we report the full figures of story-level and system-level absolute correlations between automatic metrics with all three correlation coefficients.

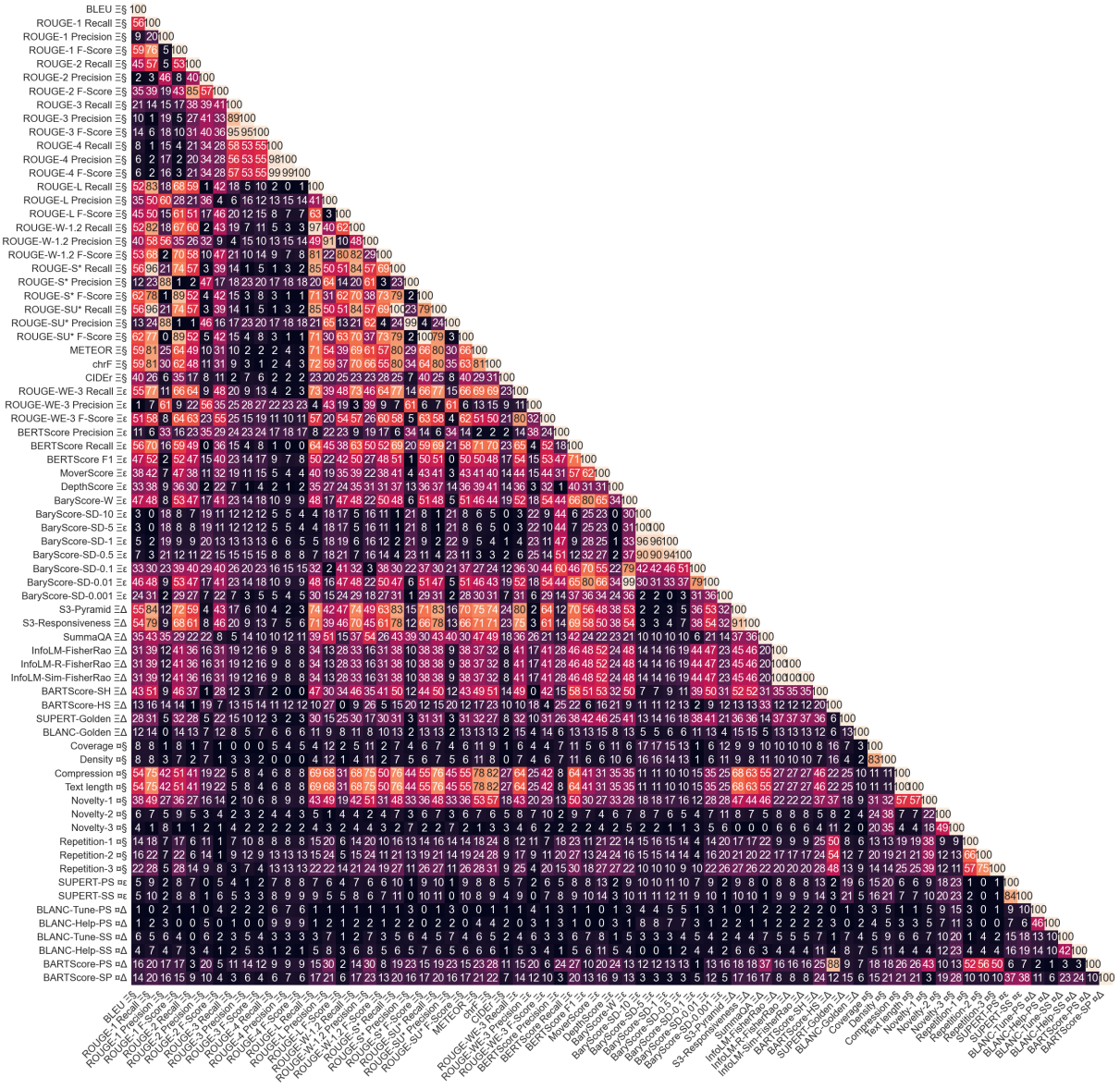

Fig. 20: Story-level absolute Kendall correlations (%) between automatic metrics

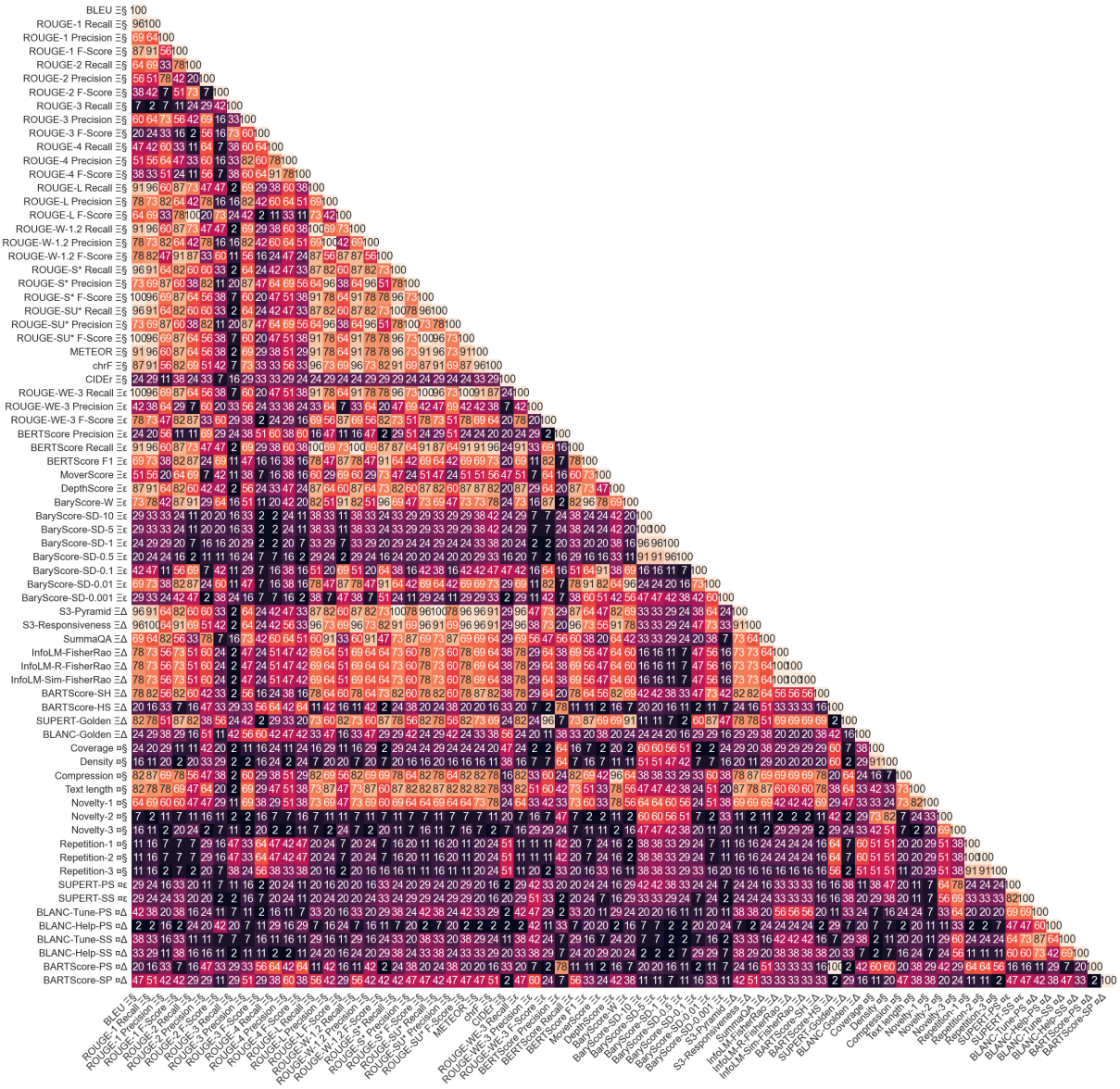

Fig. 21: System-level absolute Kendall correlations (%) between automatic metrics

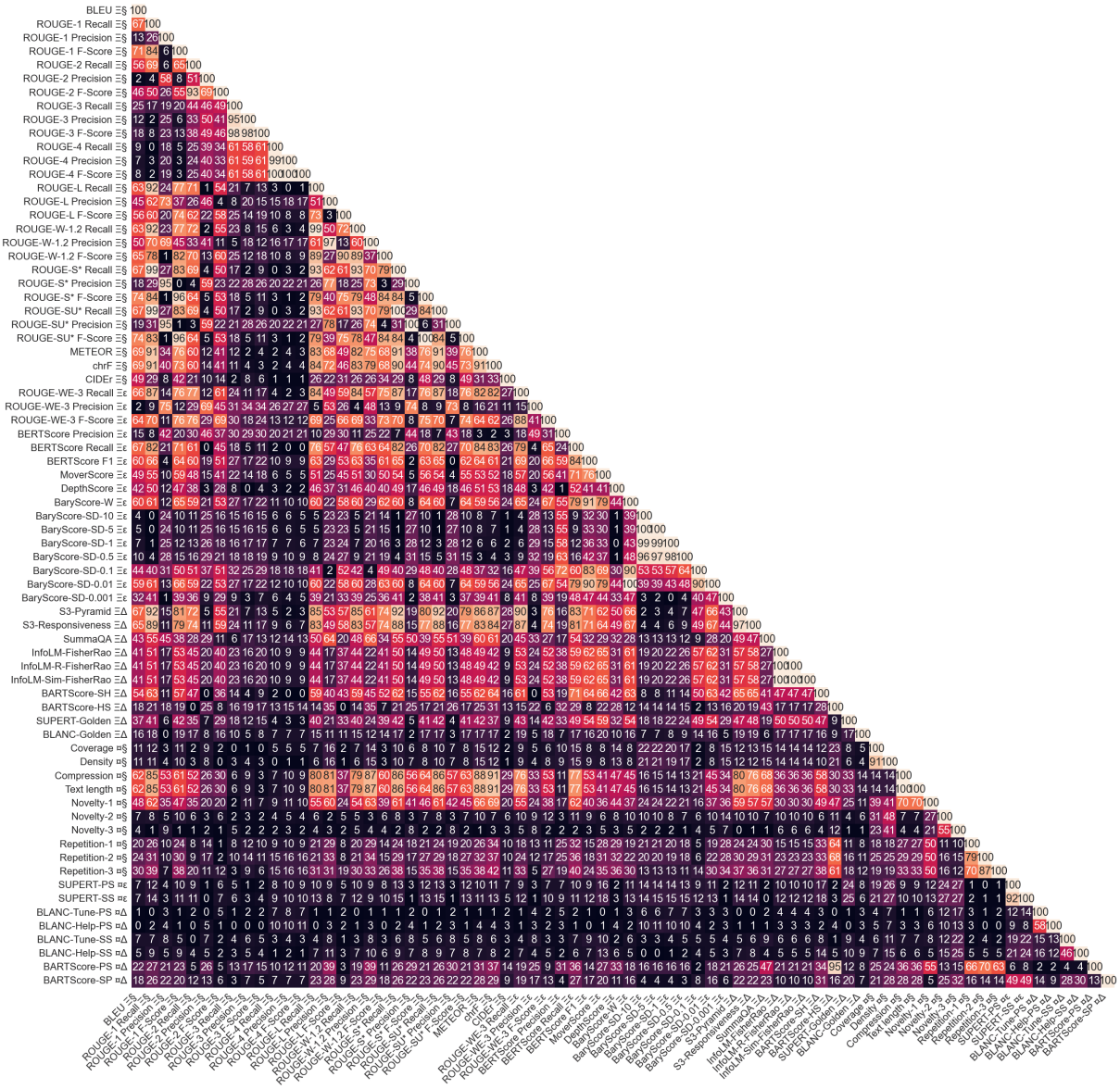

Fig. 22: Story-level absolute Spearman correlations (%) between automatic metrics

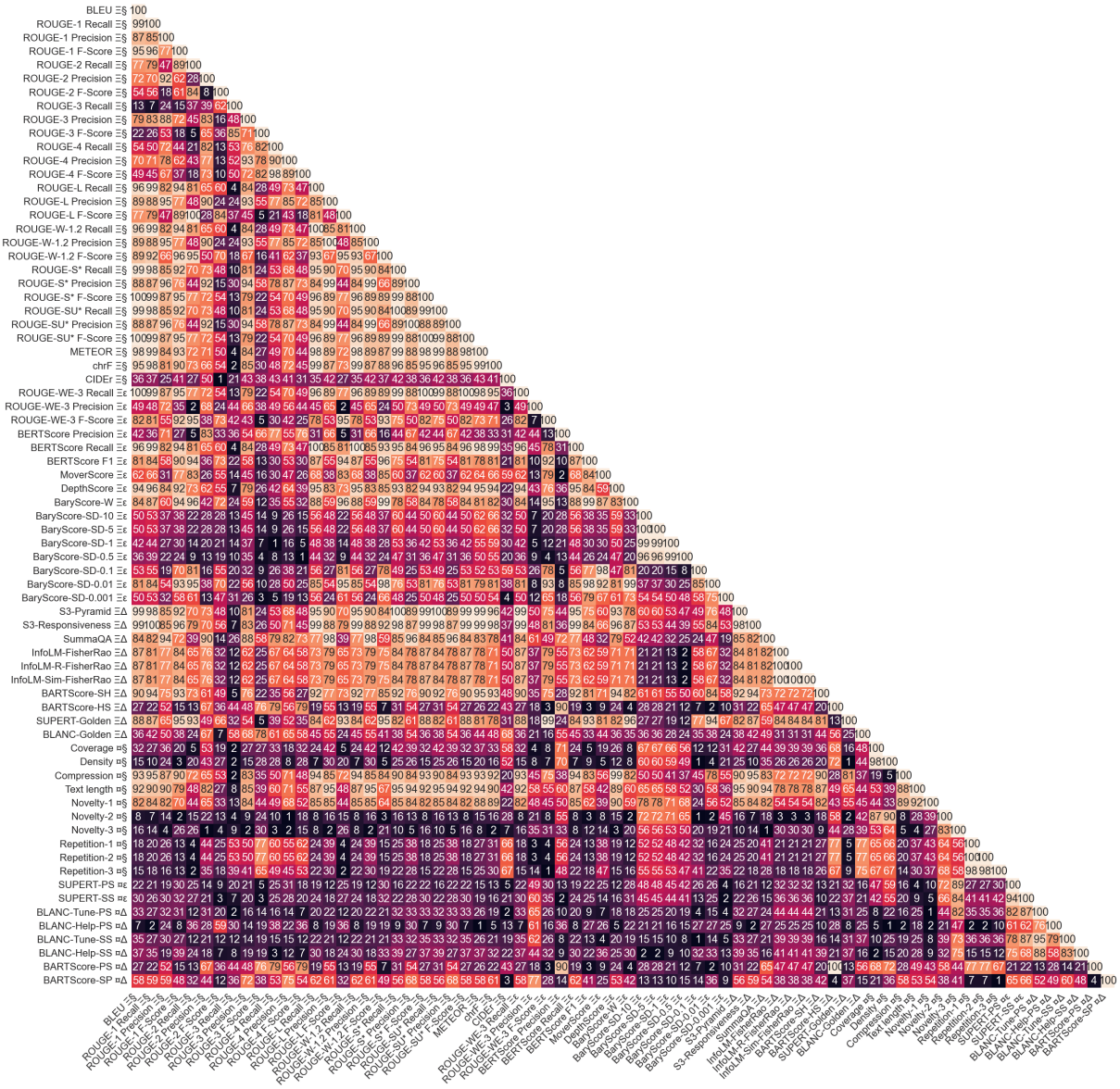

Fig. 23: System-level absolute Spearman correlations (%) between automatic metrics

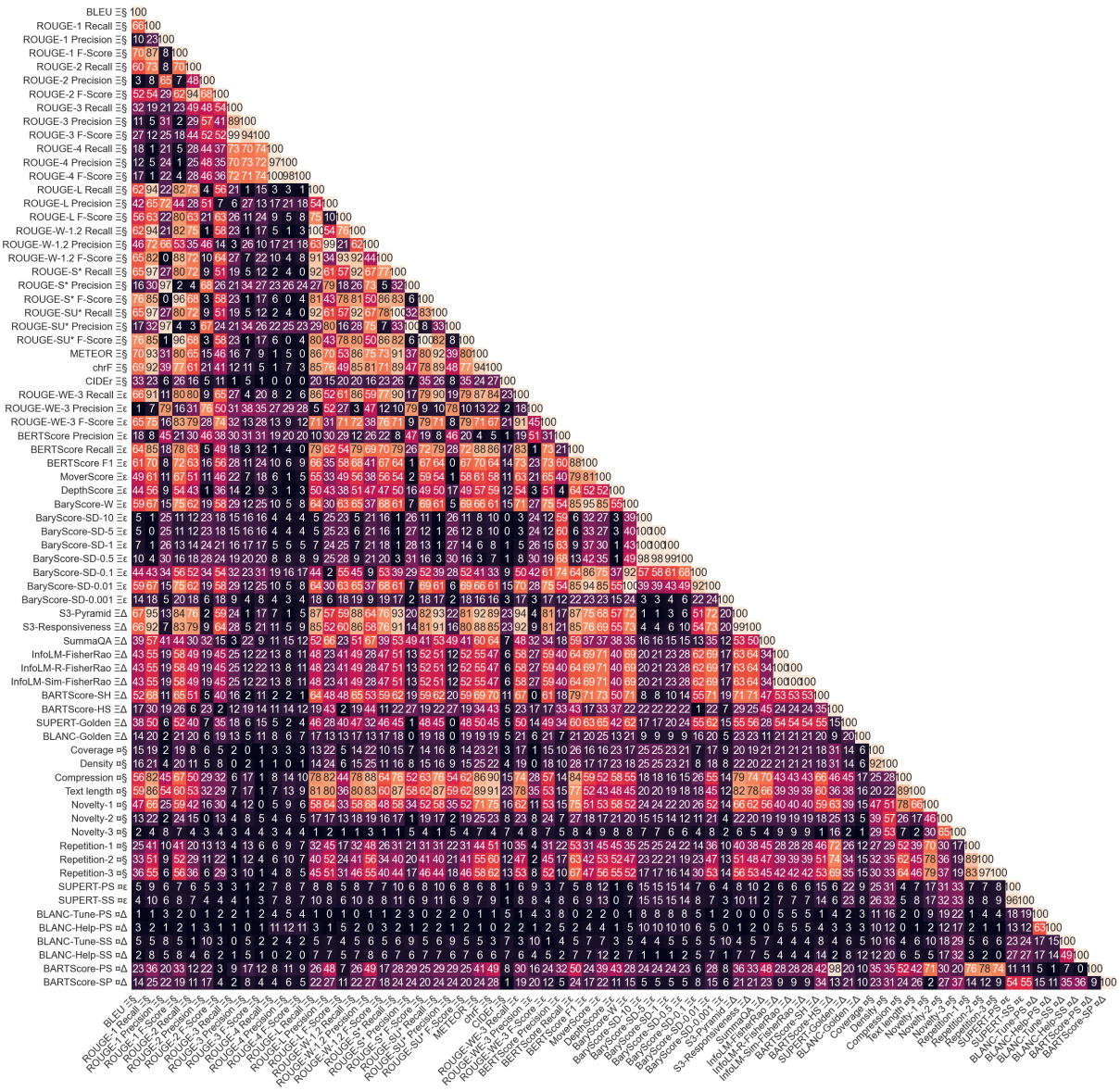

Fig. 24: Story-level absolute Pearson correlations (%) between automatic metrics

Fig. 25: System-level absolute Pearson correlations (%) between automatic metrics

G Best metrics per criterion per level of correlation per correlation coefficient

Here we report the top 5 metrics per criterion per story-level and system-level absolute correlation coefficient.

Criterion	$\lvert\tau\rvert$ (%)		$\lvert\rho\rvert$ (%)		$\lvert r\rvert$ (%)	
RE	SUPERT-SS$^{\Xi\varepsilon}$	29.95	SUPERT-SS$^{\Xi\varepsilon}$	38.58	BARTScore-SP$^{\text{¤}\Delta}$	42.55
	BARTScore-SP$^{\text{¤}\Delta}$	29.61	BARTScore-SP$^{\text{¤}\Delta}$	37.98	SUPERT-SS$^{\Xi\varepsilon}$	41.16
	SUPERT-PS$^{\Xi\varepsilon}$	28.59	SUPERT-PS$^{\Xi\varepsilon}$	36.40	SUPERT-PS$^{\Xi\varepsilon}$	40.15
	BARTScore-SH$^{\Xi\Delta}$	22.32	BARTScore-SH$^{\Xi\Delta}$	28.53	BARTScore-SH$^{\Xi\Delta}$	28.98
	MoverScore$^{\Xi\varepsilon}$	19.12	MoverScore$^{\Xi\varepsilon}$	23.67	SUPERT-Golden$^{\Xi\varepsilon}$	24.72
CH	ROUGE-WE-3 Recall$^{\Xi\varepsilon}$	25.29	ROUGE-WE-3 Recall$^{\Xi\varepsilon}$	32.22	Repetition-3$^{\text{¤}\S}$	38.12
	BARTScore-SH$^{\Xi\Delta}$	25.06	CHRF$^{\Xi\S}$	32.03	BERTScore Recall$^{\Xi\varepsilon}$	37.12
	CHRF$^{\Xi\S}$	24.61	BARTScore-SH$^{\Xi\Delta}$	31.38	S3-Pyramid$^{\Xi\Delta}$	37.05
	S3-Pyramid$^{\Xi\Delta}$	24.39	S3-Responsiveness$^{\Xi\Delta}$	31.31	CHRF$^{\Xi\S}$	36.99
	S3-Responsiveness$^{\Xi\Delta}$	24.28	S3-Pyramid$^{\Xi\Delta}$	31.14	Repetition-2$^{\text{¤}\S}$	36.54
EM	ROUGE-WE-3 Recall$^{\Xi\varepsilon}$	23.58	ROUGE-WE-3 Recall$^{\Xi\varepsilon}$	29.85	S3-Pyramid$^{\Xi\Delta}$	32.78
	CHRF$^{\Xi\S}$	23.33	CHRF$^{\Xi\S}$	29.81	CHRF$^{\Xi\S}$	32.43
	S3-Pyramid$^{\Xi\Delta}$	23.19	S3-Pyramid$^{\Xi\Delta}$	29.68	BERTScore Recall$^{\Xi\varepsilon}$	32.06
	ROUGE-SU* Recall$^{\Xi\S}$	23.13	ROUGE-SU* Recall$^{\Xi\S}$	29.38	S3-Responsiveness$^{\Xi\Delta}$	31.78
	ROUGE-S* Recall$^{\Xi\S}$	23.08	ROUGE-S* Recall$^{\Xi\S}$	29.32	BARTScore-SH$^{\Xi\Delta}$	31.66
SU	CHRF$^{\Xi\S}$	24.45	CHRF$^{\Xi\S}$	31.55	Novelty-1$^{\text{¤}\S}$	32.86
	ROUGE-1 Recall$^{\Xi\S}$	23.67	ROUGE-1 Recall$^{\Xi\S}$	30.86	CHRF$^{\Xi\S}$	32.65
	S3-Responsiveness$^{\Xi\Delta}$	23.35	S3-Responsiveness$^{\Xi\Delta}$	30.41	ROUGE-1 Recall$^{\Xi\S}$	31.32
	Novelty-1$^{\text{¤}\S}$	23.11	ROUGE-SU* Recall$^{\Xi\S}$	30.30	S3-Pyramid$^{\Xi\Delta}$	31.07
	ROUGE-SU* Recall$^{\Xi\S}$	22.85	ROUGE-S* Recall$^{\Xi\S}$	30.25	BERTScore Recall$^{\Xi\varepsilon}$	30.98
EG	CHRF$^{\Xi\S}$	30.77	CHRF$^{\Xi\S}$	39.03	BERTScore Recall$^{\Xi\varepsilon}$	42.95
	S3-Pyramid$^{\Xi\Delta}$	29.62	S3-Pyramid$^{\Xi\Delta}$	37.74	Novelty-1$^{\text{¤}\S}$	42.27
	ROUGE-1 Recall$^{\Xi\S}$	29.19	ROUGE-1 Recall$^{\Xi\S}$	37.02	CHRF$^{\Xi\S}$	41.07
	S3-Responsiveness$^{\Xi\Delta}$	29.01	S3-Responsiveness$^{\Xi\Delta}$	36.85	S3-Pyramid$^{\Xi\Delta}$	40.34
	BERTScore Recall$^{\Xi\varepsilon}$	28.93	ROUGE-S* Recall$^{\Xi\S}$	36.60	Repetition-3$^{\text{¤}\S}$	39.53
CX	CHRF$^{\Xi\S}$	43.31	CHRF$^{\Xi\S}$	54.11	CHRF$^{\Xi\S}$	58.76
	ROUGE-1 Recall$^{\Xi\S}$	40.65	ROUGE-1 Recall$^{\Xi\S}$	50.60	BERTScore Recall$^{\Xi\varepsilon}$	55.83
	ROUGE-SU* Recall$^{\Xi\S}$	39.83	Text length$^{\text{¤}\S}$	50.19	ROUGE-1 Recall$^{\Xi\S}$	55.01
	Text length$^{\text{¤}\S}$	39.82	Compression$^{\text{¤}\S}$	50.19	METEOR$^{\Xi\S}$	54.41
	Compression$^{\text{¤}\S}$	39.82	ROUGE-SU* Recall$^{\Xi\S}$	50.10	Compression$^{\text{¤}\S}$	54.38

Tab. 8: Top 5 metrics per criterion per story-level correlation coefficient

Criterion	$\lvert\tau\rvert$ (%)		$\lvert\rho\rvert$ (%)		$\lvert r\rvert$ (%)	
RE	S3-Pyramid$^{\Xi\Delta}$	60.00	MoverScore$^{\Xi\varepsilon}$	78.18	ROUGE-S* F-Score$^{\Xi\S}$	80.39
	CHRF$^{\Xi\S}$	60.00	S3-Pyramid$^{\Xi\Delta}$	75.76	ROUGE-SU* F-Score$^{\Xi\S}$	80.29
	ROUGE-SU* Recall$^{\Xi\S}$	60.00	ROUGE-S* Recall$^{\Xi\S}$	75.76	ROUGE-S* Recall$^{\Xi\S}$	80.24
	ROUGE-S* Recall$^{\Xi\S}$	60.00	ROUGE-SU* Recall$^{\Xi\S}$	75.76	ROUGE-SU* Recall$^{\Xi\S}$	80.23
	ROUGE-W-1.2 F-Score$^{\Xi\S}$	60.00	CHRF$^{\Xi\S}$	74.55	BLEU$^{\Xi\S}$	79.89
CH	BaryScore-SD-0.001$^{\Xi\varepsilon}$	77.78	BaryScore-SD-0.001$^{\Xi\varepsilon}$	92.73	BaryScore-SD-0.01$^{\Xi\varepsilon}$	88.15
	BaryScore-SD-5$^{\Xi\varepsilon}$	68.89	BaryScore-SD-5$^{\Xi\varepsilon}$	78.18	BaryScore-W$^{\Xi\varepsilon}$	87.99
	BaryScore-SD-10$^{\Xi\varepsilon}$	68.89	BaryScore-SD-10$^{\Xi\varepsilon}$	78.18	BERTScore F1$^{\Xi\varepsilon}$	87.91
	BaryScore-SD-1$^{\Xi\varepsilon}$	64.44	BaryScore-SD-1$^{\Xi\varepsilon}$	75.76	DepthScore$^{\Xi\varepsilon}$	87.38
	BaryScore-SD-0.5$^{\Xi\varepsilon}$	60.00	BERTScore F1$^{\Xi\varepsilon}$	74.55	MoverScore$^{\Xi\varepsilon}$	86.95
EM	BaryScore-SD-0.001$^{\Xi\varepsilon}$	77.78	BaryScore-SD-0.001$^{\Xi\varepsilon}$	92.73	BaryScore-SD-0.01$^{\Xi\varepsilon}$	90.01
	BERTScore F1$^{\Xi\varepsilon}$	73.33	BERTScore F1$^{\Xi\varepsilon}$	84.24	BaryScore-W$^{\Xi\varepsilon}$	89.96
	BaryScore-SD-0.01$^{\Xi\varepsilon}$	73.33	BaryScore-SD-0.01$^{\Xi\varepsilon}$	84.24	BERTScore F1$^{\Xi\varepsilon}$	88.67
	MoverScore$^{\Xi\varepsilon}$	73.33	MoverScore$^{\Xi\varepsilon}$	81.82	SUPERT-Golden$^{\Xi\Delta}$	88.10
	BaryScore-W$^{\Xi\varepsilon}$	68.89	BaryScore-W$^{\Xi\varepsilon}$	80.61	ROUGE-WE-3 F-Score$^{\Xi\varepsilon}$	87.93
SU	BaryScore-SD-0.001$^{\Xi\varepsilon}$	77.78	BaryScore-SD-0.001$^{\Xi\varepsilon}$	90.30	BARTScore-SH$^{\Xi\Delta}$	92.65
	BaryScore-SD-5$^{\Xi\varepsilon}$	68.89	BaryScore-SD-5$^{\Xi\varepsilon}$	83.03	BERTScore Recall$^{\Xi\varepsilon}$	91.09
	BaryScore-SD-10$^{\Xi\varepsilon}$	68.89	BaryScore-SD-10$^{\Xi\varepsilon}$	83.03	DepthScor$^{\Xi\varepsilon}$e	90.71
	BaryScore-SD-1$^{\Xi\varepsilon}$	64.44	BaryScore-SD-1$^{\Xi\varepsilon}$	79.39	SUPERT-Golden$^{\Xi\Delta}$	89.83
	BaryScore-SD-0.5$^{\Xi\varepsilon}$	60.00	BaryScore-SD-0.5$^{\Xi\varepsilon}$	76.97	Compression$^{¤\S}$	89.24
EG	BaryScore-SD-0.001$^{\Xi\varepsilon}$	77.78	BaryScore-SD-0.001$^{\Xi\varepsilon}$	92.73	DepthScore$^{\Xi\varepsilon}$	93.44
	BaryScore-SD-5$^{\Xi\varepsilon}$	68.89	BaryScore-SD-5$^{\Xi\varepsilon}$	78.18	BARTScore-SH$^{\Xi\Delta}$	92.44
	BaryScore-SD-10$^{\Xi\varepsilon}$	68.89	BaryScore-SD-10$^{\Xi\varepsilon}$	78.18	SUPERT-Golden$^{\Xi\Delta}$	92.21
	BaryScore-SD-1$^{\Xi\varepsilon}$	64.44	BaryScore-SD-1$^{\Xi\varepsilon}$	75.76	MoverScore$^{\Xi\varepsilon}$	92.07
	BaryScore-SD-0.5$^{\Xi\varepsilon}$	60.00	BERTScore F1$^{\Xi\varepsilon}$	74.55	BERTScore F1$^{\Xi\varepsilon}$	91.74
CX	BaryScore-SD-10$^{\Xi\varepsilon}$	76.41	BaryScore-SD-10$^{\Xi\varepsilon}$	91.19	DepthScore$^{\Xi\varepsilon}$	95.63
	BaryScore-SD-5$^{\Xi\varepsilon}$	76.41	BaryScore-SD-5$^{\Xi\varepsilon}$	91.19	BERTScore Recall$^{\Xi\varepsilon}$	95.49
	BaryScore-SD-1$^{\Xi\varepsilon}$	71.91	BaryScore-SD-1$^{\Xi\varepsilon}$	87.54	Compression$^{¤\S}$	94.31
	CHRF$^{\Xi\S}$	67.42	Novelty-1$^{¤\S}$	87.54	BARTScore-SH$^{\Xi\Delta}$	93.83
	Novelty-1$^{¤\S}$	67.42	BaryScore-SD-0.5$^{\Xi\varepsilon}$	85.11	ROUGE-1 F-Score$^{\Xi\S}$	93.35

Tab. 9: Top 5 metrics per criterion per system-level correlation coefficient.

H Weighted macro F1-scores between automatic metrics and human criteria

Here we report the weighted macro F1-scores between automatic metrics and human criteria obtained through the paired bootstrap resampling test.

Fig. 26: Weighted macro F1-scores of paired bootstrap resampling

I Williams tests between automatic metrics

Here we report the p-values of the Williams tests between automatic metrics for each criterion with story-level and system-level Pearson correlations.

Fig. 27: p-values (%) of the Williams tests between automatic metrics for the RE criterion with story-level Pearson correlations. Green case means that the row metric has a higher correlation than the column metric, dark green means the increase is statistically significant ($p < 0.05$).

Fig. 28: p-values (%) of the Williams tests between automatic metrics for the CH criterion with story-level Pearson correlations. Green case means that the row metric has a higher correlation than the column metric, dark green means the increase is statistically significant ($p < 0.05$).

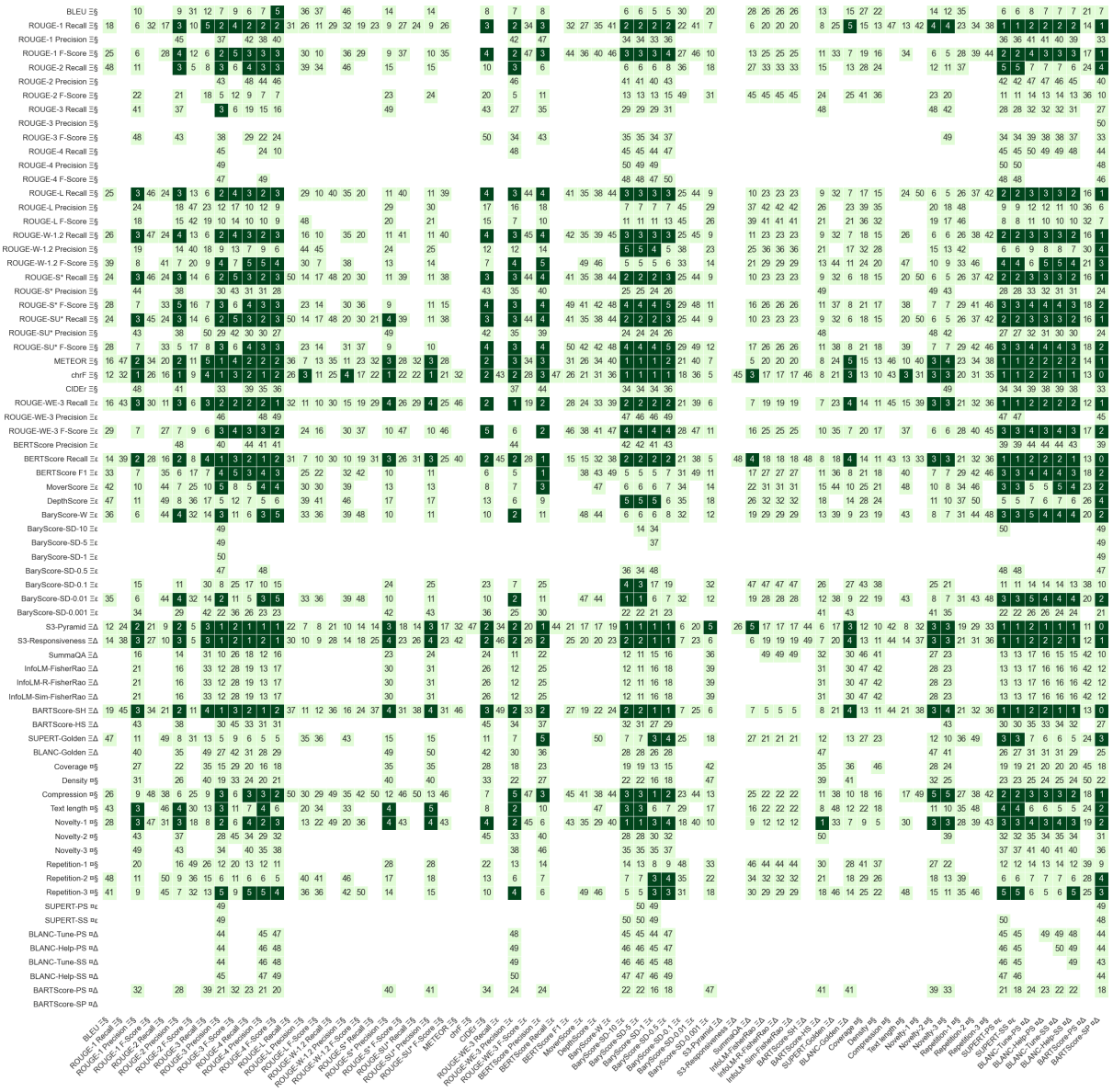

Fig. 29: p-values (%) of the Williams tests between automatic metrics for the EM criterion with story-level Pearson correlations. Green case means that the row metric has a higher correlation than the column metric, dark green means the increase is statistically significant ($p < 0.05$).

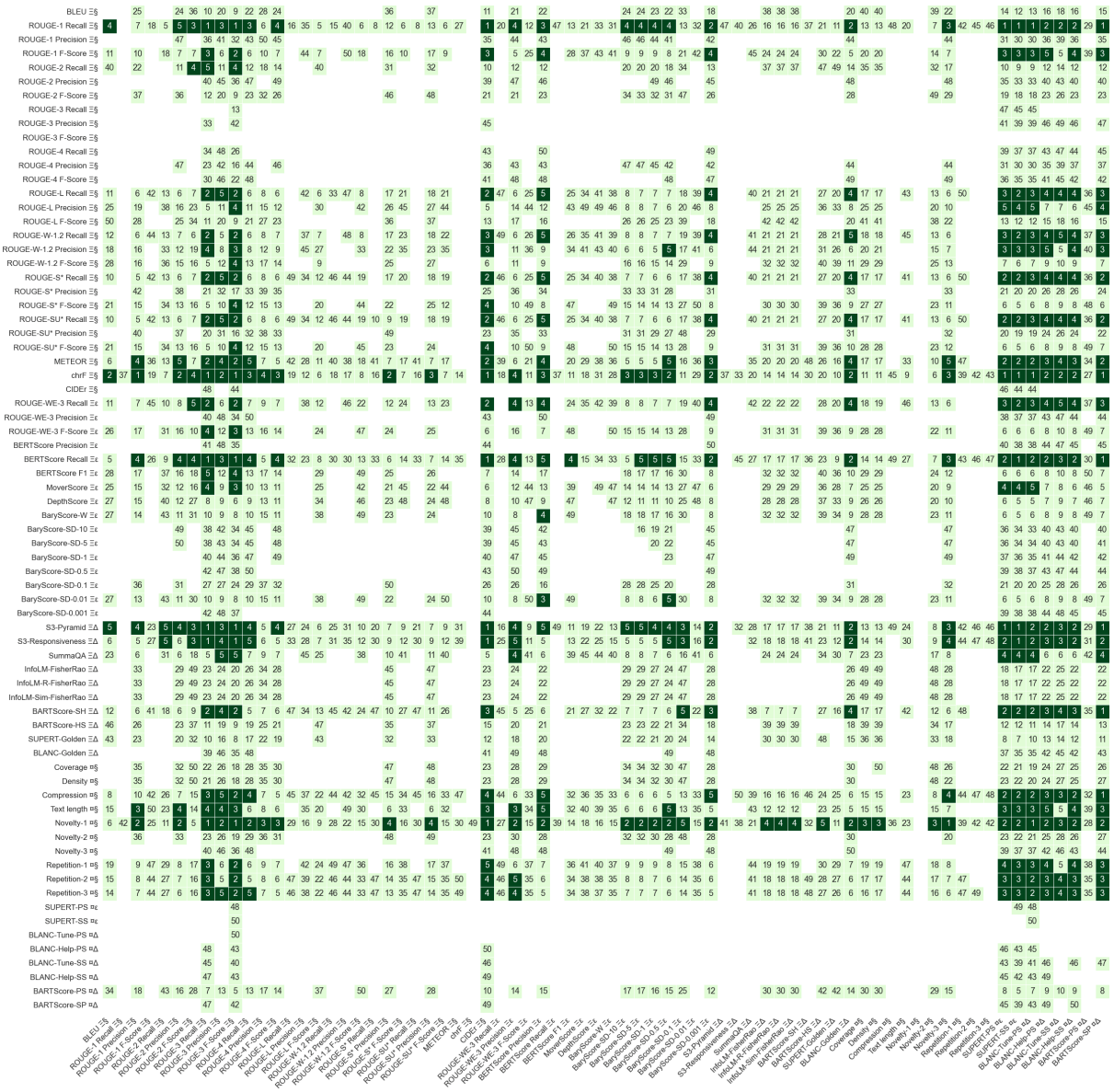

Fig. 30: p-values (%) of the Williams tests between automatic metrics for the SU criterion with story-level Pearson correlations. Green case means that the row metric has a higher correlation than the column metric, dark green means the increase is statistically significant ($p < 0.05$).

Fig. 31: p-values (%) of the Williams tests between automatic metrics for the EG criterion with story-level Pearson correlations. Green case means that the row metric has a higher correlation than the column metric, dark green means the increase is statistically significant ($p < 0.05$).

Fig. 32: p-values (%) of the Williams tests between automatic metrics for the CX criterion with story-level Pearson correlations. Green case means that the row metric has a higher correlation than the column metric, dark green means the increase is statistically significant ($p < 0.05$).

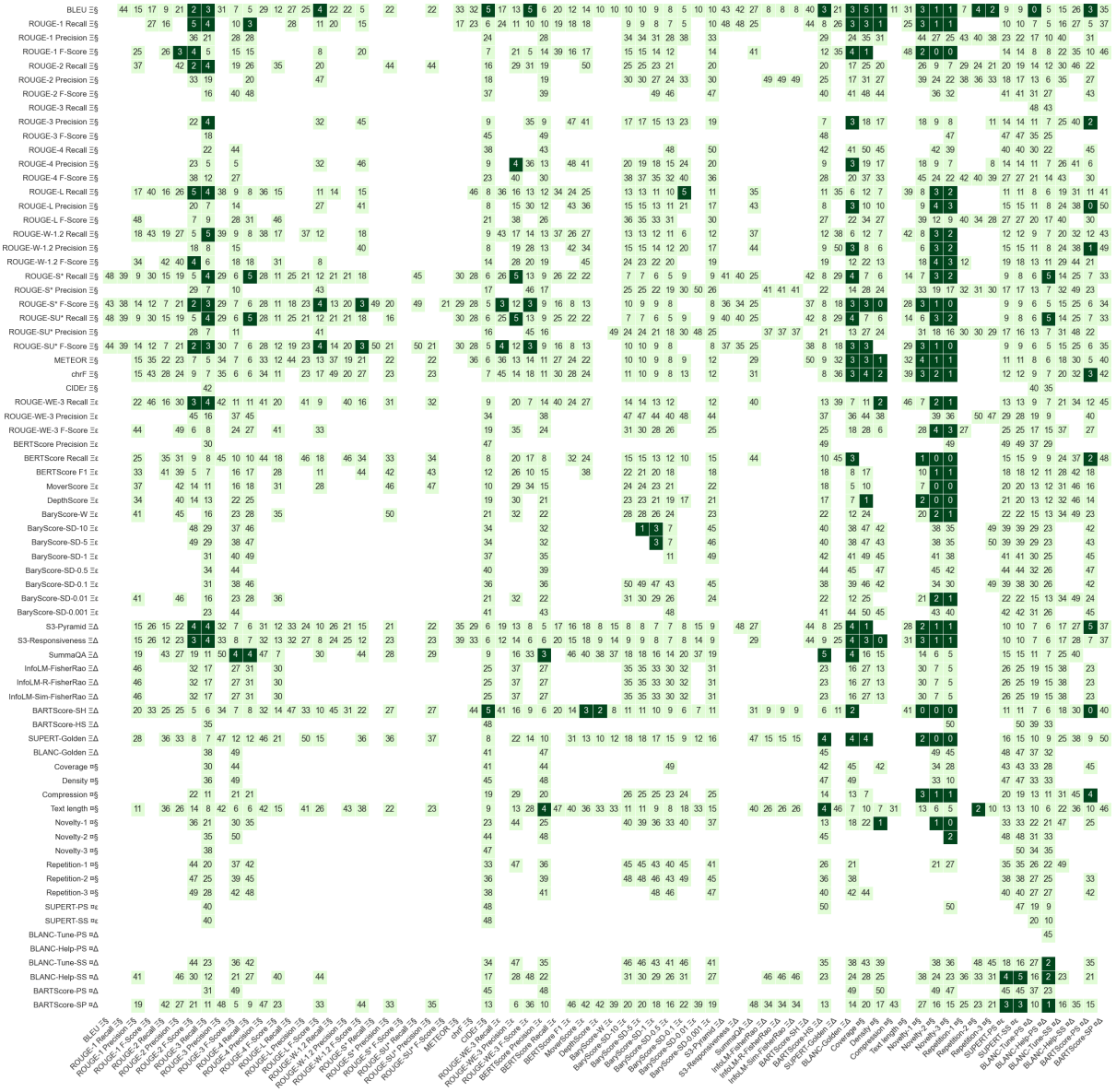

Fig. 33: p-values (%) of the Williams tests between automatic metrics for the RE criterion with system-level Pearson correlations. Green case means that the row metric has a higher correlation than the column metric, dark green means the increase is statistically significant ($p < 0.05$).

Fig. 34: p-values (%) of the Williams tests between automatic metrics for the CH criterion with system-level Pearson correlations. Green case means that the row metric has a higher correlation than the column metric, dark green means the increase is statistically significant ($p < 0.05$).

Fig. 35: p-values (%) of the Williams tests between automatic metrics for the EM criterion with system-level Pearson correlations. Green case means that the row metric has a higher correlation than the column metric, dark green means the increase is statistically significant ($p < 0.05$).

Fig. 36: p-values (%) of the Williams tests between automatic metrics for the SU criterion with system-level Pearson correlations. Green case means that the row metric has a higher correlation than the column metric, dark green means the increase is statistically significant ($p < 0.05$).

Fig. 37: p-values (%) of the Williams tests between automatic metrics for the EG criterion with system-level Pearson correlations. Green case means that the row metric has a higher correlation than the column metric, dark green means the increase is statistically significant ($p < 0.05$).

Fig. 38: p-values (%) of the Williams tests between automatic metrics for the CX criterion with system-level Pearson correlations. Green case means that the row metric has a higher correlation than the column metric, dark green means the increase is statistically significant ($p < 0.05$).

Selective Token Generation for Few-shot Natural Language Generation

Daejin Jo[1], Taehwan Kwon[1], Eun-Sol Kim[2], and Sungwoong Kim[1]

[1]Kakao Brain
[2]Hanyang University
[1]{daejin.jo, taehwan.kwon, swkim}@kakaobrain.com
[2]eunsolkim@hanyang.ac.kr

Abstract

Natural language modeling with limited training data is a challenging problem, and many algorithms make use of large-scale pretrained language models (PLMs) for this due to its great generalization ability. Among them, additive learning that incorporates a task-specific adapter on top of the fixed large-scale PLM has been popularly used in the few-shot setting. However, this added adapter is still easy to disregard the knowledge of the PLM especially for few-shot natural language generation (NLG) since an entire sequence is usually generated by only the newly trained adapter. Therefore, in this work, we develop a novel additive learning algorithm based on reinforcement learning (RL) that selectively outputs language tokens between the task-general PLM and the task-specific adapter during both training and inference. This output token selection over the two generators allows the adapter to take into account solely the task-relevant parts in sequence generation, and therefore makes it more robust to overfitting as well as more stable in RL training. In addition, to obtain the complementary adapter from the PLM for each few-shot task, we exploit a separate selecting module that is also simultaneously trained using RL. Experimental results on various few-shot NLG tasks including question answering, data-to-text generation and text summarization demonstrate that the proposed selective token generation significantly outperforms the previous additive learning algorithms based on the PLMs.

1 Introduction

Recently, pretrained language models (PLMs) have shown great generalization ability when combined with large-scale data and big transformer-based models (Devlin et al., 2019; Radford et al., 2019; Lewis et al., 2020; Brown et al., 2020; Subramanyam Kalyan et al., 2021; Petroni et al., 2019; Wang et al., 2020). Therefore, transfer learning from PLMs has been popularly used for few-shot natural language generation (NLG) tasks with promising results. In specific, the use of PLM for few-shot NLG can be categorized into three approaches: 1) prompt-based, 2) finetuning, and 3) additive learning. Prompt-based approaches encode a task description and task-specific examples as a natural language prompt for few-shot text generation (Radford et al., 2019; Brown et al., 2020; Zheng and Huang, 2021; Schick and Schütze, 2020; Li and Liang, 2021a). These approaches can take full advantage of the universal natural language understanding and generation capabilities of large-scale PLMs without further training of the main model, however, they have some limitations in dealing with a large domain shift from the pretraining corpus data, tuning suitable task-specific prompts, and covering an increased size of conditioning examples. On the other hand, finetuning of the PLM is able to explicitly impart task-specific knowledge to the model and hence lift the above limitations (Ziegler et al., 2019; Xu et al., 2021; Chen et al., 2020). However, these finetuned models are prone to overfitting when only a small amount of training data is available. In order to alleviate such an overfitting problem, additive learning has been extensively exploited by incorporating task-specific adapters into the PLM (Stickland and Murray, 2019; Houlsby et al., 2019; Zeldes et al., 2020).

In general, task-specialized adapters for few-shot NLG are trained by maximum likelihood estimation (MLE) or reinforcement learning (RL). While MLE is efficient in learning, it suffers from the exposure bias problem due to the difference in the training and inference mechanisms (He et al., 2019), and this problem can be severe with limited training data. One solution is RL, capable of resolving this exposure bias problem by sequential output sampling during training (Ranzato et al., 2015; Keneshloo et al., 2019; Shi et al., 2021). However, the exponentially large space of output sequences restricts the use of RL since it leads to high variance and unstable training which is more serious in the

Proceedings of the 29th International Conference on Computational Linguistics, pages 5837–5856
October 12–17, 2022.

Passage	three types of conflicts are : 1. intrapersonal conflicts , 2. interpersonal conflicts and 3. unconscious conflicts . the word conflict has been derived from a latin word “conflicts” which means “strike two things at the same time” . conflict is [1)]**an opposition or a tug-of-war between contradictory impulses .** according to colman "a conflict is [2)]**the anticipated frustration entailed in the choice of either alternative**".
Query	conflict definition psychology
Ground-truth	the anticipated frustration entailed in the choice of either alternative.
PLM	conflict definition psychology.
Adapter	conflict is an opposition or a tug-of-war between contradictory impulses.
PLM with Condition	the meaning of conflict is (provided condition) the anticipated frustration entailed in the choice of either alternative.
Proposed STG	**conflict is** the anticipated frustration entailed in the choice of either alternative.

Table 1: Generated answers from an instance of MS-MARCO QA dataset. Two definitions about *conflict* are presented in bold text in the passage. The answers are sampled from the models trained on 0.5% few-shot subset data. The proposed selective token generation (STG) produces the first two words (highlighted in **red**) by the task-specific adapter while the others by the PLM.

few-shot setting.

More importantly, the existing additive learning generally produces the whole output sequence by its own task-specific adapter, which leads to a fundamental limitation in maintaining the knowledge of the PLM and the strong generation ability. An example of this limitation from our empirical observation on the task of question and answering is shown in Table 1. In this case, a passage that contains two definitions (super-scripted and bolded) about *conflict* is given with a query that asks about the psychological meaning of *conflict*. Without the knowledge of *who Colman*[1] *is*, it can be hard to answer since the word *psychology* in the query does not appear in the passage. Here, the PLM repeats the given query as its generated answer due to the lack of domain adaptation while the added adapter incorrectly outputs not the psychological meaning but the general meaning of *conflict*. This is because most queries in this few-shot training data ask a general meaning of a concept, and therefore the adapter is overfitted to this pattern (more examples are described in Section 4.6). Note that the PLM generates the correct answer if the proper conditioning text (*the meaning of conflict is*) is provided.

Motivated by these observations, in this work, we propose a novel RL-based selective token generation (STG) between the task-general PLM and the task-specific adapter. The selection of this output token generator enables to explicitly maintain a general prior knowledge from the frozen PLM and the adapter to focus only on the task-relevant parts in sequence generation. Note that the proposed algorithm is different from previous selective generation algorithms such as *copy mechanism* (Gu et al., 2016) in that STG selects a generator rather than existing tokens in a given passage. In few-shot learning, the proposed partial token generation makes the task-specific adapter more resilient to overfitting and furthermore reduces the overall output space which leads to stable RL training. Here, in order to make the two token generators (policies) complement each other as well as to realize the robust output selection at the token level on the fly, we exploit a separate token-level policy selector. Note that both the policy selector and the task-specific adapter are simultaneously learned by the RL algorithm. Experimental results on various few-shot NLG tasks show that the proposed selective token generation outperforms the previous PLM-based additive learning algorithms with the comprehensive (non-selective) token generation.

Our main contributions can be summarized as follows.

- A novel selective token generation between the PLM and the task-specific adapter is pro-

[1]A psychologist, https://en.wikipedia.org/wiki/Peter_T._Coleman_(academic)

posed for few-shot NLG.

- RL is applied to train both the policy selector and the task-specific adapter that is complementary to the PLM in text generation.
- Extensive empirical validation on few-shot NLG tasks demonstrates that the proposed selective token generation performs better in comparison to the previous PLM-based additive learning algorithms.

2 Background

2.1 Natural Language Generation

The goal of NLG is to generate a text sequence $\mathbf{y} = [y_0, ..., y_T]$ for a given task, where y_t is the tth output token from a vocabulary $\mathcal{V}$, and T is the output sequence length. For this generation, we aims to model the distribution of $\mathbf{y}$ that is autoregressively factorized as $p_\theta(\mathbf{y}) = \prod_{t=0}^{T} p_\theta(y_t|\mathbf{y}_{<t})$, where θ denotes the model parameters and $\mathbf{y}_{<t} = [y_0, ..., y_{t-1}]$. Here, the conditional distribution to sample a token for each step, $p_\theta(y_t|\mathbf{y}_{<t})$, is defined by the softmax function on the output logits $f_\theta(y_t|\mathbf{y}_{<t})$. Note that in general, the language generation is conditioned on input context according to a given task. Here, we encode the conditioning context by the same sequential model for generating an output sequence, and for simplicity we omit it.

2.2 Additive Learning for Few-shot Generation

To effectively leverage the general linguistic knowledge, θ is first initialized by the PLM parameters, θ_{LM}, for NLG. Given N task-specific training instances, $\mathcal{D} = \{\mathbf{y}^{n*}\}_{n=1}^{N}$, where $\mathbf{y}^{n*}$ is the nth ground-truth output sequence, directly finetuning θ_{LM} using $\mathcal{D}$ can incur the severe overfitting problem when N is small in the few-shot scenario. Therefore, we add the task-specific adapter, g_{θ_a} parameterized by θ_a, on top of the PLM, and optimize only θ_a (Zeldes et al., 2020; Stickland and Murray, 2019). In specific, we reformulate $f(\cdot|\mathbf{y}_{<t}; \theta) = W^T h(\mathbf{y}_{<t}; \theta_h)$ where $W \in R^{H \times |\mathcal{V}|}$ and $h \in R^H$ denote the weight matrix and the penultimate representations, respectively, and $\theta = \{W, \theta_h\}$. Then, we define the task-specific conditional distribution as follows:

$$p(y_t|\mathbf{y}_{<t}; \theta_{LM}, \theta_a) = \sigma\Big({W_{LM}}^T h_{LM}(\mathbf{y}_{<t}) + {W_a}^T g\big(h_{LM}(\mathbf{y}_{<t}); \theta_g\big)\Big), \quad (1)$$

where $h_{LM}(\mathbf{y}_{<t}) = h(\mathbf{y}_{<t}; \theta_{h,LM})$, $\theta_a = \{W_a, \theta_g\}$ and σ is the softmax function. Here, the summation of the PLM logits and the adapter logits is motivated by auxiliary training[2] (Zeldes et al., 2020). It is noted that in our additive learning θ_a is updated while θ_{LM} is kept frozen. Hence, in the following we omit θ_{LM} such that $p_{\theta_a}(y_t|\mathbf{y}_{<t}) = p(y_t|\mathbf{y}_{<t}; \theta_{LM}, \theta_a)$ for simplicity.

2.3 Reinforcement Learning (RL)

As an alternative to MLE, RL is able to overcome the exposure bias problem of MLE by sequence-level sampling from the model distribution during training (Ranzato et al., 2015) and allows to leverage the target-specific sequence-level objectives such as BLEU (Wu et al., 2018; Guo et al., 2021). In order to use RL for our additive learning, we reformulate our text generation as an RL problem: at each time step t, the agent takes the current state $\mathbf{s}_t = \mathbf{y}_{<t}$ as an input and performs an action a_t that outputs a token y_t by a policy $\pi_\theta(a_t|\mathbf{s}_t)$ corresponding to $p_\theta(y_t|\mathbf{y}_{<t})$. Then, the agent receives a reward $r_t = r(\mathbf{s}_t, a_t)$ and deterministically transitions to the next state $\mathbf{s}_{t+1}$. Here, note that the token-level intermediate reward $r_t = 0, \forall t < T$ when we use the delayed reward associated with the sequence-level evaluation metric between the two full sequences, $\mathbf{y}$ and $\mathbf{y}^*$. Let $\tau = \{(\mathbf{s}_t, a_t, r_t)\}_{t=0}^{T}$ be the trajectory generated by π_θ. The RL objective for the optimal agent is to maximize the expected sum of future discounted rewards $\mathbb{E}_{\tau \sim \pi_\theta}[\sum_{t=0}^{T} \gamma^t r_t]$, where $\gamma \in [0, 1]$ is the discount factor. We employ an actor-critic algorithm (Bahdanau et al., 2017) which requires the additional critic network to estimate the value of a state, $V^\pi(\mathbf{s}_t) = E_\pi[\sum_{t'=t}^{T} \gamma^{t'-t} r_{t'}|\mathbf{s}_t] = \sum_{a_t} \pi(a_t|\mathbf{s}_t) Q^\pi(\mathbf{s}_t, a_t)$ where the state-action value function $Q^\pi(\mathbf{s}_t, a_t) = E_\pi[\sum_{t'=t}^{T} \gamma^{t'-t} r_{t'}|\mathbf{s}_t, a_t] = r_t + V^\pi(\mathbf{s}_{t+1})$. We use the policy gradient loss to learn the policy parameters θ: $\mathcal{L} = -\sum_{t=0}^{T} A^{\pi_\theta}(\mathbf{s}_t, a_t) \log \pi_\theta(a_t|\mathbf{s}_t)$,

[2] Although the auxiliary training is particularly designed for maximizing the likelihood of the target task output, it also can take an advantage for RL since the adapter logits are nearly zero before training is advanced. Namely, it lets the task-specific conditional distribution start learning from the distribution of PLM, not a uniform distribution.

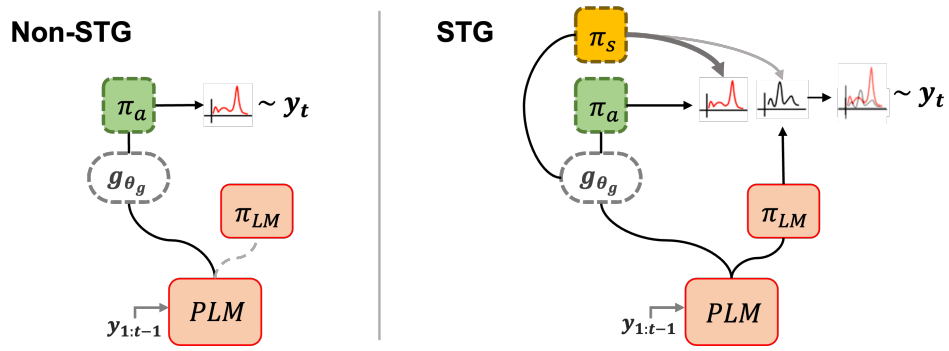

Figure 1: Text generation processes of Non-STG and STG are described. In the Non-STG, every token is sampled from the task-specific policy π_a (**Left**). On the other hand, in the proposed STG, each token is selectively sampled from either the PLM policy π_{LM} or the test-specific policy π_a where the selection is performed by the selection policy π_s (**Right**). Symbols with dashed line represent learnable models.

where $A^{\pi_\theta}(\mathbf{s}_t, a_t) = Q^{\pi_\theta}(\mathbf{s}_t, a_t) - V^{\pi_\theta}(\mathbf{s}_t)$ is the advantage function.

3 Selective Token Generation

Instead of generating all tokens in an output sequence from the single task-specific policy, $\pi_a = \pi_{\theta_a}(a_t|\mathbf{s}_t)$, at each time step t, we sample an output token y_t selectively from either the PLM policy $\pi_{LM} = \pi_{\theta_{LM}}(a_t|\mathbf{s}_t)$ or the task-specific policy π_a:

$$y_t = a_t \sim \big(\mathbb{1}_t[\pi_{LM} \text{ is selected}]\pi_{LM}(a_t|\mathbf{s}_t) + (1 - \mathbb{1}_t[\pi_{LM} \text{ is selected}])\pi_a(a_t|\mathbf{s}_t)\big), \quad (2)$$

where $\mathbb{1}_t[\cdot]$ is the indicator function (at t) that equals 1 if it is true and 0 otherwise. This output token selection allows to explicitly utilize a general linguistic knowledge from the PLM without catastrophic forgetting in few-shot learning. Also, the task-specific policy can focus on generating only the task-relevant parts, which enables more effective few-shot training with a reduced search space.

Now we need to determine how to select the proper policy at each step on the fly as well as to make the task-specific policy complementary to the PLM policy. For this, we exploit a separate token-level policy selector. The proposed policy selector $\pi_s(i_t|\mathbf{s}_t;\theta_s)$ with the parameters θ_s, where $i_t \in \{0, 1\}$, is an another policy that stochastically decides a policy to generate a_t for $\mathbf{s}_t$. Namely, a token sample y_t is generated by the following process:

$$i_t \sim \pi_s(i_t|\mathbf{s}_t), \quad (3)$$

$$y_t = \begin{cases} a_t \sim \pi_{LM}(a_t|\mathbf{s}_t) & \text{if } i_t = 0, \\ a_t \sim \pi_a(a_t|\mathbf{s}_t) & \text{if } i_t = 1. \end{cases} \quad (4)$$

This process can be considered as a token generation from a hierarchical policy $\pi_h(a_t|\mathbf{s}_t;\theta_s,\theta_{LM},\theta_a)$ where the policy selector represents the upper-level prior for the preference of the low-level policy. Therefore, the value function of this hierarchical policy can be formulated as

$$\begin{aligned} V^{\pi_h} &= \mathbb{E}_{\pi_h}[\sum_{t'=t}^{T} \gamma^{t'-t} r_{t'}|\mathbf{s}_t] \\ &= \pi_s(0_t|\mathbf{s}_t) \sum_{a_t} \pi_{LM}(a_t|\mathbf{s}_t) Q^{\pi_h}(\mathbf{s}_t, a_t) \\ &+ \pi_s(1_t|\mathbf{s}_t) \sum_{a_t} \pi_a(a_t|\mathbf{s}_t) Q^{\pi_h}(\mathbf{s}_t, a_t), \end{aligned}$$

and $A^{\pi_h}(\mathbf{s}_t, a_t) = Q^{\pi_h}(\mathbf{s}_t, a_t) - V^{\pi_h}(\mathbf{s}_t)$. We denote $i_t = 0$ and $i_t = 1$ as 0_t and 1_t respectively. Here, it is noted that a single critic network is used for the hierarchical policy since i_t does not affect $\mathbf{s}_t$. Given a sample trajectory $\{(\mathbf{s}_t, i_t, a_t, r_t)\}_{t=0}^{T}$, the loss for optimizing θ_s and θ_a is

$$\mathcal{L} = -\sum_{t=0}^{T} A^{\pi_h}(\mathbf{s}_t, a_t)\Big(\mathbb{1}[0_t]\mathcal{L}_{LM} + \mathbb{1}[1_t]\mathcal{L}_a\Big), \quad (5)$$

where

$$\begin{aligned} \mathcal{L}_{LM} &= \log sg[\pi_{LM}(a_t|\mathbf{s}_t)] + \log \pi_s(i_t|\mathbf{s}_t), \\ \mathcal{L}_a &= \log \pi_a(a_t|\mathbf{s}_t) + \log \pi_s(i_t|\mathbf{s}_t) \end{aligned}$$

and sg stands for the stop-gradient operator. Similar to π_a, π_s makes use of the PLM representations and the task-specific adapter such that

$$\pi_s(i_t|\mathbf{s}_t;\theta_s) = \sigma\Big(m\Big(g\big(h_{LM}(\mathbf{s}_t)\big);\theta_s\Big)\Big), \quad (6)$$

where σ is the softmax function and m is the selector module. Figure 1 depicts the overall text generation process by the proposed selective token generation (STG) in comparison to the previous non-selective token generation (Non-STG). Here, note that since all policies in STG share the same PLM representations, the increased computational cost by STG over Non-STG is negligible.

The use of the separated policy selector that is simultaneously trained with the task-specific policy allows the task-specific policy to be complementary to the PLM policy. Especially, this cooperative ensemble learning can be realized by our RL algorithm that performs sequential sampling from the model during training.

The advantages of STG are as follows: **(1)** STG makes use of the PLM not at the feature level but the output distribution level in text generation. In our few-shot learning this is beneficial in explicitly retaining strong linguistic and world knowledge from the PLM. **(2)** STG resolves the exponentially large search space $|\mathcal{V}|^T$ since the frozen PLM chooses a token when it is selected, and therefore the search space of the generator is approximately decreased from $|\mathcal{V}|^T$ to $|\mathcal{V}|^{T-\overline{T}_{PLM}}$ where $\overline{T}_{PLM}$ is the average length of sequences generated by PLM. **(3)** STG is efficient in credit assignment. The loss function of STG (Equation 5) intuitively shows that the gradient to the task-specific policy π_a associated with producing a_t will depend on the selector's action (i.e. $i_t = 1$). Hence, unlike Non-STG, π_a of STG knows which token is used as a task-specific token and contributed to the reward (see Figure 2 for an illustration).

It is noted that although the STG also can be trained by MLE, it can be easily collapsed to select only a task-specific policy irrespective of a given content. We analyze the MLE version of STG in Appendix B.

4 Experiments

We evaluate our method against additive learning baselines on Data-to-Text, Question Answering and Text Summarization tasks which are widely used in few-shot NLG.

4.1 Baseline

PLM. In our experiments, we assume that the PLM works to some extent for a given task. However, the naive PLM usually does not satisfy it for a new task unseen during training. Hence, we finetuned GPT-2[3] (Radford et al., 2019) with MLE for few epochs and used it as the PLM. Fine-tuning the PLM with MLE is most commonly used for task adaptation and thus it can also be a strong baseline. This fine-tuning phase accelerates the learning of the adapter. This is particularly when the adaptation requires to cover the large domain shift. Severe performance degradation was observed for all the tasks when we skipped this fine-tuning.

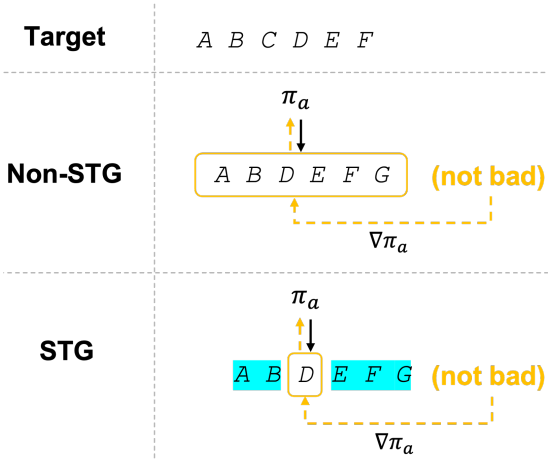

Figure 2: A simple schematic illustration of Non-STG and STG. **Non-STG(RL):** the whole sequence of target is generated from the task-specific policy π_a so the right sub-sequence AB is also penalized from the delayed feedback. **STG:** the third token is sampled from π_a and the model lets the other tokens (highlighted with cyan) generated from the PLM's policy π_{LM} which generates a next letter of the previous alphabet input. Here, π_a will be penalized at the third token.

Non-STG. This method stands for Non-Selective Token Generation which uses the above the PLM as an encoder (frozen) and the adapter (additional layer to be trained). We use two objectives, MLE and RL, for additive learning. These will be denoted as Non-STG-MLE and Non-STG-RL, respectively.

STG-Naive Ensemble. We believe that the proposed generation encourages the task-specific policy (π_a) to complement the PLM's policy (π_{LM}) with a proper selection of the selector through the joint training. To investigate this, we evaluate against two different naive ensembles of the policies, π_a trained from Non-STG and π_{LM} of the PLM. These ensemble schemes are as follows:

[3]We make use of GPT-2 with 345M parameters as the initial checkpoint. We follow the training details in the previous works (Peng et al., 2020; Khandelwal et al., 2019) for each task.

Model	Restaurant		Hotel		TV		Laptop	
	BLEU ↑	ERR ↓	BLEU ↑	ERR ↓	BLEU ↑	ERR ↓	BLEU ↑	ERR ↓
PLM	19.42	12.57	35.84	13.74	29.0	9.15	28.27	9.31
Non-STG-MLE	17.21	15.87	28.42	12.64	29.83	10.05	26.76	10.52
Non-STG-RL	18.01	11.98	36.72	12.64	28.66	9.19	28.59	9.21
NE(max)-MLE	14.12	15.27	31.32	14.29	28.23	10.21	26.93	10.02
NE(mix)-MLE	**25.27**	14.97	37.13	15.93	**32.85**	16.31	**32.91**	14.77
NE(max)-RL	15.2	11.68	32.68	16.48	28.91	9.24	28.66	9.51
NE(mix)-RL	24.1	19.16	38.07	18.68	32.84	18.06	32.53	17.14
STG	21.28	**10.78**	**38.09**	**11.54**	30.24	**9.03**	30.41	**8.91**

Table 2: Data-to-Text performance on FewShotWOZ dataset.

- NE(max): $\pi_{max} = \sigma(Max(\pi_a, \pi_{LM}))$
- NE(mix): $\pi_{mix} = (\pi_a + \pi_{LM})/2$

We also evaluate another naive ensemble strategy NE($random$) that randomly selects a token policy at each step between π_a and π_{LM}, however it shows lower performances than the others.

4.2 Implementation

Adapter. The task-specific adapter g in Section 2.2 is implemented by a LSTM to encode the dynamics of the representation vector h_{LM}. We found that the use of MLP was not good in the sense of performance.

Selector. We use a 2-layer MLP with ReLU activation for m of Equation 6.

Reinforcement Learning. We employ Actor-Critic method (Konda and Tsitsiklis, 2000; Fedus et al., 2018) for RL. The agents (i.e. selector and generator) receive a reward after generating a sentence. Here, we use different reward functions according to tasks. We use delexicalised BLEU for Data-to-Text following (Peng et al., 2020), Averaged score of BLEU and ROUGE-L for Question Answering and ROUGE-L for Text Summarization following (Paulus et al., 2017) as the reward function.

Token Sampling. During the training, $i_t \in \{0, 1\} \sim \pi_s$ is first sampled, and then we use either π_{LM} of the PLM for $i_t = 0$ or the task-specific policy π_a for $i_t = 1$ to sample the tth token. During the evaluation, any decoding strategy, such as a beam search, can be used with the mixture of policies $\pi_h(\cdot) = \pi_s(0_t)\pi_{LM}(\cdot) + \pi_s(1_t)\pi_a(\cdot)$. We use the beam search decoding with a sample size of $k = 3$ for Text Summarization and $top_p = 0.9$ decoding for both Data-to-Text ($k = 10$) and Question Answering ($k = 3$).

4.3 Data-to-Text

Data-to-Text is a task that transforms structured data such as graphs or tables into natural language. Recent works (Mager et al., 2020; Peng et al., 2020; Kale, 2020) show that the PLM can be adapted successfully to this task by taking a serialized form of data as an input without a carefully designed model to encode the structured data. Here, we perform experiments on FewShotWOZ (Peng et al., 2020) dataset. The evaluation is conducted on the topics which are available[4]. Only 50 instances for each topic are available for training and 129, 78, 1379, and 680 testing instances for Restaurant, Hotel, Laptop, and TV, respectively. The models are evaluated by measuring fluency and informativeness using BLEU score and ERR (slot ERror Rate), respectively. Table 2 shows the obtained results.

4.4 Long Answer Question Answering

We consider Long Answer Question Answering (QA) task on MS-MARCO (Nguyen et al., 2016) dataset. In this task, a passage and a query are given, and the model generates an answer with respect to the query by referring to the passage. Here, we randomly sample various sizes of (50, 100, 500, 1,000 $\approx$ 1%, and 2,000) subset data from the train dataset. We also sample a validation and a test set, which contains 500 and 12,000 instances, respectively, from the dev dataset. We repeat this test three times with different random seeds and thus perform experiments on total nine subsets. The models are evaluated by measuring BLEU and ROUGE-L (denoted as R-L). We report *averaged performances* over the three runs and *averaged performance gain* against the PLM in Table 3 and Figure 3, respectively.

[4] https://github.com/pengbaolin/SC-GPT

Model	50 shot		100 shot		500 shot		1,000 shot		2,000 shot	
	BLEU	R-L	BLEU	R-L	BLEU	R-L	BLEU	R-L	BLEU	R-L
PLM	19.99	29.01	34.93	41.27	35.64	43.10	41.49	49.76	47.72	56.02
Non-STG-MLE	27.46	35.08	34.08	40.93	34.53	43.08	41.02	50.14	47.85	56.81
Non-STG-RL	20.07	28.94	35.08	41.28	35.08	42.78	41.25	49.97	48.00	56.83
NE(max)-MLE	27.21	34.95	34.76	41.87	34.69	43.93	41.11	50.77	47.65	57.22
NE(mix)-MLE	26.97	35.1	35.31	41.82	36.26	44.43	42.26	**51.14**	**48.44**	**57.3**
NE(max)-RL	20.05	28.9	35.0	41.16	35.14	42.94	41.51	50.54	47.58	57.06
NE(mix)-RL	20.69	29.62	35.11	41.33	35.93	43.52	42.29	50.84	48.28	57.02
STG	**33.33**	**39.59**	**36.3**	**43.24**	**37.37**	**44.53**	**42.76**	**51.19**	**48.42**	**57.3**

Table 3: *Averaged performances* for Question Answering on various few-shot subset data of MS-MARCO.

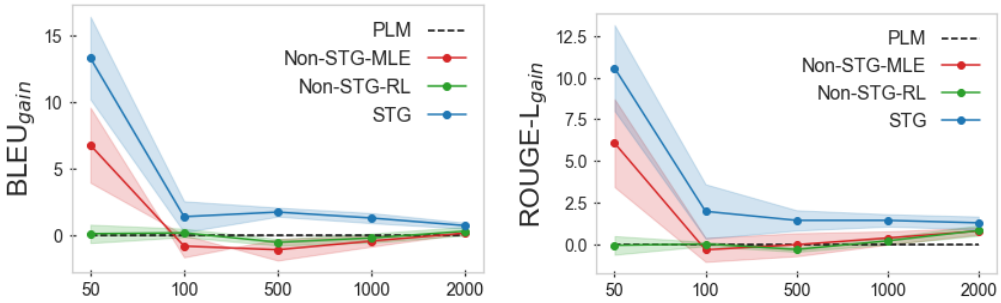

Figure 3: *Averaged performance gains* against the PLM for Question Answering on various few-shot subset data of MS-MARCO. The x-axis represents the size of the subset data and the shaded area represents a range of standard deviation over 3 randomly sampled subset data with different random seeds. STG provides significantly larger gains compared to Non-STGs on BLEU (Left) and ROUGE-L (Right).

4.5 Text Summarization

We consider the problem of abstractive summarization for long text generation. Here, we randomly sample various sizes of (50, 100, 300, 1,500, and 3,000 $\approx$ 1%) subset data from CNN/Daily Mail (See et al., 2017). We repeat this test three times for each size of few-shot as in above QA task. ROUGE (Lin, 2004) is commonly used to evaluate n-grams recall of the summaries with gold references. The models are evaluated by measuring ROUGE-1, ROUGE-2, and ROUGE-L (denoted as R1, R2, and R-L, respectively). We report *averaged performances* over the three runs and *averaged performance gain* against the PLM in Table 4 and Figure 4, respectively.

4.6 Result

In most cases, additive learning improves the performances over the PLM. However, they do not always guarantee a performance improvement. For example, the ERR score of the PLM on *Laptop* shows a better result except for STG and NE(mix)-RL (see Table 2) and the Non-STGs trained on $1,000 \approx 1\%$ few-shot subset of MS-MARCO do not outperform the PLM (see Table 3).

Data-to-Text. As shown in Table 2, we can observe that the Non-STGs do not outperform the PLM even though it has more neural units and takes more training time. The models trained on the RL objective show better performances for the ERR (lower is better). Interestingly, NE(mix) methods show strong improvements for the BLEU which measures the fluency of sentence but obvious degeneration for the ERR which measures the rate of missing information from the given data. These results suggest that the PLM is much more capable of *task-general* knowledge than the *task-specific* generator (i.e. π_a) trained on few-shot dataset, which ensures our motivation of jointly training the policy selector and the task-specific generator is valid. Note that while other methods show some trade-off between BLEU and ERR, only STG shows improvements on both metrics for all topics in the dataset.

Question Answering. As shown in Table 3, STG shows significantly better performances than the other methods. Notably, NE(mix) show good performances as much as STG especially where the training data size $\geq 1,000$. It obviously suggests that the PLM can be a complementary model to the additional model. Therefore, in this context, it can be lost of the prior knowledge of the PLM even if

Model	50 shot			100 shot			300 shot			1,500 shot			3,000 shot		
	R1	R2	R-L	R1	R2	R-L	R1	R2	R-L	R1	R2	R-L	R1	R2	R-L
PLM	14.67	4.57	10.69	16.58	5.28	12.05	19.38	7.08	13.74	30.19	11.27	21.21	33.05	12.96	23.39
Non-STG-MLE	15.39	4.81	11.09	17.09	5.41	12.3	18.9	6.87	13.36	30.34	11.32	21.2	33.19	12.98	23.39
Non-STG-RL	15.22	4.76	11.08	16.55	5.25	12.0	19.61	7.11	13.83	30.35	11.34	21.22	33.22	12.99	23.4
NE(max)-MLE	15.52	4.89	11.24	16.98	5.43	12.26	19.19	7.0	13.56	30.33	11.31	21.2	33.19	12.99	23.4
NE(mix)-MLE	15.4	4.83	11.16	16.88	5.4	12.22	19.45	7.07	13.75	30.32	11.31	21.23	33.11	12.99	23.41
NE(max)-RL	15.14	4.73	11.02	16.52	5.27	11.99	19.47	7.1	13.76	30.37	11.35	21.26	33.21	12.99	23.41
NE(mix)-RL	14.95	4.67	10.89	16.6	5.29	12.04	19.58	7.14	13.84	30.28	11.3	21.22	33.14	13.0	23.42
STG	**17.4**	**5.33**	**12.42**	**17.96**	**5.73**	**12.94**	**23.27**	**8.32**	**16.29**	**30.47**	**11.37**	**21.36**	**33.45**	**13.14**	**23.66**

Table 4: *Averaged performances* for Text Summarization on various few-shot subset data of CNN/DM.

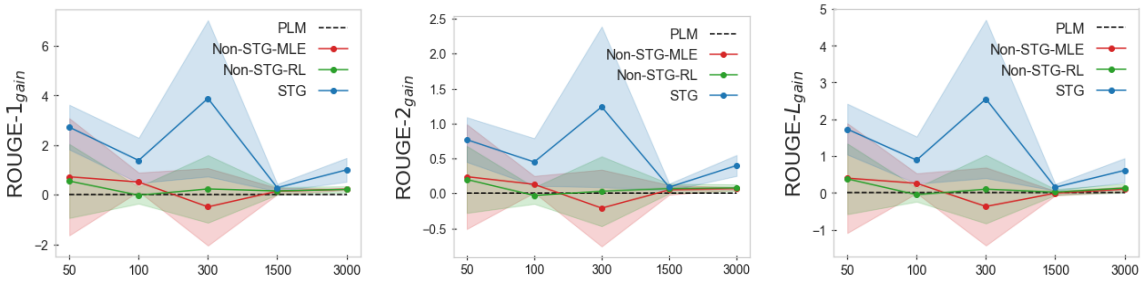

Figure 4: *Averaged performance gains* against the PLM for Text Summarization on various few-shot subset data of CNN/DM. The x-axis represents the size of the subset data and the shaded area represents a range of standard deviation over 3 randomly sampled subset data with different random seeds. STG provides significantly larger gains compared to Non-STGs on ROUGE-1 (Left), ROUGE-1 (Middle), and ROUGE-L (Right).

the additional model has been built over the feature space of the PLM. In addition, we can expect that STG would be more beneficial on the small number of samples for this kind of tasks which depend on the PLM's ability like common sense knowledge. As shown in Figure 3, STG shows strong improvements compared to Non-STG-RL especially where the training data size ≤ 500.

Summarization. As shown in Table 4, STG shows significantly larger gains than Non-STGs, and their naive ensembles with the PLM in every score metric and training data size. Similar to QA, STG shows improvements compared to Non-STGs especially where the training data size ≤ 300 as shown in Figure 4. However in contrast to the QA task, the improvement may seem limited for all models including STG. We think that the adapters used in this study may not be suitable for this particular task which requires to understand the long context and compress it into a summary. It may need the use of lower-level features or more parameters to adapt to such tasks. We discuss this limitation in Section 6.

Overfitting in Non-STGs. In the example as shown in Table 1 the answer of STG, which is close to the ground truth, is generated by the PLM policy π_{LM} after some sequence of tokens (*conflict is*) that are sampled from the task-specific policy π_a. The Non-STGs generate general meaning which is not intended. We can find such examples for the other tasks in Appendix C: In Data-To-Text, as shown in the last example of Table 6, Non-STG generates *nicam stereo* which is not appeared in the given data. This is due to that *nicam stereo* was appeared 7 times (7/50, 14%) in training data. In Summarization, as shown in the first example of Table 10, Non-STGs only consider the forepart of the given article. Since the most of the major information is appeared in the forepart in News data, Non-STGs can be easily overfitted to generate the text according to such a pattern. Hence, we claim that Non-STG is easily exposed to learning patterns of typical answering, but STG resolves this issue since it can be fully accessible to the knowledge of the PLM.

5 Related Work

Recently, prompt-based in-context learning with an extremely large PLM shows impressive few-shot generation performances (Radford et al., 2019; Brown et al., 2020). Schick and Schütze (2020) propose manually designed natural language prompts for improved few-shot text summarization and headline generation. Elsahar et al. (2018) conduct zero-shot learning for question generation from knowledge graphs, however they require a large

amount of in-domain training data for their transfer learning. Chen et al. (2020) directly finetune the pretrained GPT-2 with a small amount of serialized attribute-value pairs for table-to-text generation. Gong et al. (2020) further apply multiple tasks to effectively leverage the structured information of tables. In contrast to these approaches, our proposed method utilizes RL-based additive learning for few-shot text generation.

Applying RL for text generation has been widely used to mitigate the exposure bias problem of MLE as well as to directly optimize task-relevant evaluation metrics. Ranzato et al. (2015) use the REINFORCE algorithm for text summarization and machine translation while Bahdanau et al. (2017) use the actor-critic algorithm for machine translation. However, they require pretraining using MLE. Ding and Soricut (2017) propose softmax policy gradient to remove the MLE-based pretraining. However, it requires various techniques for effective training. Tan et al. (2018) propose an entropy-regularized policy optimization that subsumes many of the previous training algorithms. Our proposed method is different from these methods in that we apply RL for more difficult few-shot generative modeling.

The use of RL training in PLM has been explored in many works. Dathathri et al. (2020) propose a controllable text generation which uses discriminators to guide generation of the PLM. This approach assumes that constant *classes* like topics or preferences are available. Lazaridou et al. (2020) use a PLM as a caption generator for given image. In their referential game, the generator is rewarded by a kind of discriminator that responses a signal to the generator whether the corresponding caption is correct or not.

Various methods take into account the RL tasks with large action spaces like NLG. Dulac-Arnold et al. (2015) consider only actions in a cluster around the latent state of action obtained from a given state. Chandak et al. (2019) define the action embedding as a distribution with semantic of action and use a deterministic policy to take an action. Even-Dar et al. (2003); Zahavy et al. (2018) devise a method of incorporating the process of directly removing unnecessary actions according to the state in the RL problem. Unlike these approaches, we use the hierarchical policy that reduces the sequential action space.

6 Limitations & Future work

Adapter. In this study, we aim to propose a new generation framework for few-shot natural language generation tasks. In particular, a relatively naive neural adapter which utilizes only the top layer of the PLM is used in this paper, and thus it may lead to limited improvements as shown in the experimental results on the summarization task. Fortunately, there are several neural architectures (Houlsby et al., 2019; Li and Liang, 2021b; Alayrac et al., 2022) for efficient task adaptation, and we believe that such adapters also make STG more efficient for covering a large domain shift and scaling. The study on the architectures of the adapters will be conducted in future works.

Efficient exploration. The fundamental limitation in STG is a high dependency on PLM; When STG has a sufficient powerful PLM, the selector does not select the additional adapter and it is thus nothing more than the PLM. We can find such phenomenon in some examples in Table 7 and 8 in Appendix. On the other hand, when STG has an extremely poor PLM, the selector selects the adapter always and it is thus equivalent to Non-STG. Therefore, in the perspective of exploration of RL the STG needs balanced selections between the PLM and the adapter. Furthermore, the use of RL objective requires more training time than the methods which use MLE objective such as Prefix-Tuning (Li and Liang, 2021b) due to the auto-regressive sequence sampling during training. Therefore, an analysis on efficient exploration of STG is important for future works.

7 Conclusion

In this work, we propose to exploit a selective token generation between the pretrained language model and the task-specific adapter with RL-based additive learning for the tasks of few-shot natural language generation. In particular, we devise a trainable policy selector at the token level and jointly learn it with the task-specific policy. The proposed policy selector and RL algorithm make the two policies complementary to each other and lead to robust few-shot generative modeling. Experimental results on various tasks of few-shot text generation show that the proposed selective token generation along with RL-based additive learning consistently and significantly improves the performances with less overfitting.

References

Jean-Baptiste Alayrac, Jeff Donahue, Pauline Luc, Antoine Miech, Iain Barr, Yana Hasson, Karel Lenc, Arthur Mensch, Katie Millican, Malcolm Reynolds, Roman Ring, Eliza Rutherford, Serkan Cabi, Tengda Han, Zhitao Gong, Sina Samangooei, Marianne Monteiro, Jacob Menick, Sebastian Borgeaud, Andrew Brock, Aida Nematzadeh, Sahand Sharifzadeh, Mikolaj Binkowski, Ricardo Barreira, Oriol Vinyals, Andrew Zisserman, and Karen Simonyan. 2022. Flamingo: a visual language model for few-shot learning.

Dzmitry Bahdanau, Philemon Brakel, Kelvin Xu, Anirudh Goyal, Ryan Lowe, Joelle Pineau, Aaron Courville, and Yoshua Bengio. 2017. An actor-critic algorithm for sequence prediction. *International Conference on Learning Representations*.

Tom B. Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel M. Ziegler, Jeffrey Wu, Clemens Winter, Christopher Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. 2020. Language models are few-shot learners. *arXiv preprint arXiv:2005.14165*.

Yash Chandak, Georgios Theocharous, James Kostas, Scott Jordan, and Philip Thomas. 2019. Learning action representations for reinforcement learning. In *International Conference on Machine Learning*, pages 941–950. PMLR.

Zhiyu Chen, Harini Eavani, Wenhu Chen, Yinyin Liu, and William Yang Wang. 2020. Few-shot NLG with pre-trained language model. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 183–190, Online. Association for Computational Linguistics.

Sumanth Dathathri, Andrea Madotto, Janice Lan, Jane Hung, Eric Frank, Piero Molino, Jason Yosinski, and Rosanne Liu. 2020. Plug and play language models: A simple approach to controlled text generation. In *International Conference on Learning Representations*.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, Minneapolis, Minnesota. Association for Computational Linguistics.

Nan Ding and Radu Soricut. 2017. Cold-start reinforcement learning with softmax policy gradient. *arXiv preprint arXiv:1709.09346*.

Gabriel Dulac-Arnold, Richard Evans, Hado van Hasselt, Peter Sunehag, Timothy Lillicrap, Jonathan Hunt, Timothy Mann, Theophane Weber, Thomas Degris, and Ben Coppin. 2015. Deep reinforcement learning in large discrete action spaces. *arXiv preprint arXiv:1512.07679*.

Hady Elsahar, Christophe Gravier, and Frederique Laforest. 2018. Zero-shot question generation from knowledge graphs for unseen predicates and entity types. *arXiv preprint arXiv:1802.06842*.

Eyal Even-Dar, Shie Mannor, and Yishay Mansour. 2003. Action elimination and stopping conditions for reinforcement learning. In *Proceedings of the 20th International Conference on Machine Learning (ICML-03)*, pages 162–169.

William Fedus, Ian Goodfellow, and Andrew M Dai. 2018. Maskgan: better text generation via filling in the_. *International Conference on Learning Representations*.

Heng Gong, Yawei Sun, Xiaocheng Feng, Bing Qin, Wei Bi, Xiaojiang Liu, and Ting Liu. 2020. Tablegpt: Few-shot table-to-text generation with table structure reconstruction and content matching. In *International Conference on Computational Linguistics*, pages 1978—-1988.

Jiatao Gu, Zhengdong Lu, Hang Li, and Victor O.K. Li. 2016. Incorporating copying mechanism in sequence-to-sequence learning. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1631–1640, Berlin, Germany. Association for Computational Linguistics.

Han Guo, Bowen Tan, Zhengzhong Liu, Eric P. Xing, and Zhiting Hu. 2021. Text generation with efficient (soft) q-learning. *arXiv preprint arXiv:2106.07704*.

Tianxing He, Jingzhao Zhang, Zhiming Zhou, and James Glass. 2019. Exposure bias versus self-recovery: Are distortions really incremental for autoregressive text generation? *arXiv preprint arXiv:1905.10617*.

Neil Houlsby, Andrei Giurgiu, Stanislaw Jastrzebski, Bruna Morrone, Quentin De Laroussilhe, Andrea Gesmundo, Mona Attariyan, and Sylvain Gelly. 2019. Parameter-efficient transfer learning for nlp. In *International Conference on Machine Learning*, pages 2790–2799. PMLR.

Mihir Kale. 2020. Text-to-text pre-training for data-to-text tasks. *arXiv preprint arXiv:2005.10433*.

Yaser Keneshloo, Tian Shi, Naren Ramakrishnan, and Chandan K Reddy. 2019. Deep reinforcement learning for sequence-to-sequence models. *IEEE transactions on neural networks and learning systems*, 31(7):2469–2489.

Urvashi Khandelwal, Kevin Clark, Dan Jurafsky, and Lukasz Kaiser. 2019. Sample efficient text summarization using a single pre-trained transformer. *arXiv preprint arXiv:1905.08836*.

Vijay R Konda and John N Tsitsiklis. 2000. Actor-critic algorithms. In *Advances in neural information processing systems*, pages 1008–1014. Citeseer.

Angeliki Lazaridou, Anna Potapenko, and Olivier Tieleman. 2020. Multi-agent communication meets natural language: Synergies between functional and structural language learning. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7663–7674, Online. Association for Computational Linguistics.

Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Ves Stoyanov, and Luke Zettlemoyer. 2020. Bart: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7871–7880.

Xiang Lisa Li and Percy Liang. 2021a. Prefix-tuning: Optimizing continuous prompts for generation. *arXiv preprint arXiv:2101.00190*.

Xiang Lisa Li and Percy Liang. 2021b. Prefix-tuning: Optimizing continuous prompts for generation. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 4582–4597, Online. Association for Computational Linguistics.

Chin-Yew Lin. 2004. ROUGE: A package for automatic evaluation of summaries. In *Text Summarization Branches Out*, pages 74–81, Barcelona, Spain. Association for Computational Linguistics.

Manuel Mager, Ramón Fernandez Astudillo, Tahira Naseem, Md Arafat Sultan, Young-Suk Lee, Radu Florian, and Salim Roukos. 2020. Gpt-too: A language-model-first approach for amr-to-text generation. *arXiv preprint arXiv:2005.09123*.

Tri Nguyen, Mir Rosenberg, Xia Song, Jianfeng Gao, Saurabh Tiwary, Rangan Majumder, and Li Deng. 2016. Ms marco: A human generated machine reading comprehension dataset. In *CoCo@ NIPS*.

Romain Paulus, Caiming Xiong, and Richard Socher. 2017. A deep reinforced model for abstractive summarization. *arXiv preprint arXiv:1705.04304*.

Baolin Peng, Chenguang Zhu, Chunyuan Li, Xiujun Li, Jinchao Li, Michael Zeng, and Jianfeng Gao. 2020. Few-shot natural language generation for task-oriented dialog. *arXiv preprint arXiv:2002.12328*.

Fabio Petroni, Tim Rocktäschel, Sebastian Riedel, Patrick Lewis, Anton Bakhtin, Yuxiang Wu, and Alexander Miller. 2019. Language models as knowledge bases? In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 2463–2473, Hong Kong, China. Association for Computational Linguistics.

Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. 2019. Language models are unsupervised multitask learners. *OpenAI blog*, 1(8):9.

Marc'Aurelio Ranzato, Sumit Chopra, Michael Auli, and Wojciech Zaremba. 2015. Sequence level training with recurrent neural networks. *arXiv preprint arXiv:1511.06732*.

Timo Schick and Hinrich Schütze. 2020. Few-shot text generation with pattern-exploiting training. *arXiv preprint arXiv:2012.11926*.

Abigail See, Peter J Liu, and Christopher D Manning. 2017. Get to the point: Summarization with pointer-generator networks. *arXiv preprint arXiv:1704.04368*.

Tian Shi, Yaser Keneshloo, Naren Ramakrishnan, and Chandan K Reddy. 2021. Neural abstractive text summarization with sequence-to-sequence models. *ACM Transactions on Data Science*, 2(1):1–37.

Asa Cooper Stickland and Iain Murray. 2019. Bert and pals: Projected attention layers for efficient adaptation in multi-task learning. In *International Conference on Machine Learning*, pages 5986–5995. PMLR.

Katikapalli Subramanyam Kalyan, Ajit Rajasekharan, and Sivanesan Sangeetha. 2021. Ammus : A survey of transformer-based pretrained models in natural language processing. *arXiv preprint arXiv:2108.05542*.

Bowen Tan, Zhiting Hu, Zichao Yang, Ruslan Salakhutdinov, and Eric Xing. 2018. Connecting the dots between mle and rl for sequence prediction. *arXiv preprint arXiv:1811.09740*.

Chenguang Wang, Xiao Liu, and Dawn Song. 2020. Language models are open knowledge graphs. *arXiv preprint arXiv:2010.11967*.

Lijun Wu, Fei Tian, Tao Qin, Jianhuang Lai, and Tie-Yan Liu. 2018. A study of reinforcement learning for neural machine translation. *arXiv preprint arXiv:1808.08866*.

Xinnuo Xu, Guoyin Wang, Young-Bum Kim, and Sungjin Lee. 2021. Augnlg: Few-shot natural language generation using self-trained data augmentation. *arXiv preprint arXiv:2106.05589*.

Tom Zahavy, Matan Haroush, Nadav Merlis, Daniel J Mankowitz, and Shie Mannor. 2018. Learn what not to learn: Action elimination with deep reinforcement learning. *arXiv preprint arXiv:1809.02121*.

Yoel Zeldes, Dan Padnos, Or Sharir, and Barak Peleg. 2020. Technical report: Auxiliary tuning and its application to conditional text generation. *arXiv preprint arXiv:2006.16823*.

Chujie Zheng and Minlie Huang. 2021. Exploring prompt-based few-shot learning for grounded dialog generation. *arXiv preprint arXiv:2109.06513*.

Daniel M. Ziegler, Nisan Stiennon, Jeffrey Wu, Tom B. Brown, Alec Radford, Dario Amodei, Paul Christiano, and Geoffrey Irving. 2019. Fine-tuning language models from human preferences. *arXiv preprint arXiv:1909.08593*.

A Training Settings

In our experiments all the models of additive learning, Non-STG and STG, are used the same architecture and hyper-parameters (except whether to use pre-training) for training as described in Table 5. We found that pre-training the addtional layer of Non-STG-RL with MLE helps the performance improvements. On the other hand, STG without pre-training shows better performances. We use the training data for each topic of the task of Data-to-Text as their validation data.

B Additonal Study

B.1 STG-MLE

Here, we evaluate the MLE version of STG (denoted as STG-MLE) which is trained by MLE for the mixture policy $\pi_h(\cdot) = \pi_s(i_t = 0)\pi_{LM}(\cdot) + \pi_s(i_t = 1)\pi_a(\cdot)$ similar to *copy mechanism* (Gu et al., 2016). In few-shot training, the explicit use of PLM logits can efficiently reduce the fine-tuning loss especially when the adapter is light since the adapter can focus only on the task-relevant part in generation. STG-RL[5] learns to do this naturally by stochastic policy sampling if the policy selector is initialized to perform uniform sampling. On the other hand, STG-MLE can be easily collapsed to select only a task-specific policy (i.e. $i_t = 1$). This is because the gradient flows the additional model only and, unlike STG-RL, there is no chance to exploit diverse paths during training in the teacher forcing manner. As shown in Figure 6, the score of STG-MLE starts from the same point of STG-RL but it collapsed to Non-STG-MLE.

B.2 Learning Curve

It is well known that the RL-tuning resolves the exposure bias of MLE-tuning. We can expect that an additive learner of MLE would be affected by the exposure bias as well, and the RL objective for additive learning resolves it. Here, we present some learning curves[6] obtained from training in our experiments. As shown in Figure 6, the learners of MLE seem to have overfitting (in terms of Perplexity, PPL) and exposure bias (in terms of Score). On the other hand, the learners of RL were less effected by the problems. We can find that the STGs (denoted STG-RL) are superior to the others from the perspective of the score.

B.3 Effectiveness of Selector

Here, we investigate the effectiveness of the selector π_s of the STG. We compare *Fixed Selection* against the *Dynamic selection*. In the fixed selection, the probability of selecting the PLM's policy π_{LM} is fixed to $\pi_s(i_t = 0|s_t) = 1 - \pi_s(i_t = 1|s_t)$. We measure the performance with respect to $\pi_s(i_t = 1|s_t) = c$ where c is a constant. The selection will be uniformly random when $c = 0.5$, and when $c = 0$, the performance will be equivalent to the performance of the PLM without additive learning. Figure 5 shows that the input-dependent dynamic selection by our STG outperforms the fixed selection with any c. We can find that how π_s works for each task. For instance, in QA task, the first few tokens of an answer may decide the quality of generation (i.e. "yes" or "no" in binary QA). Therefore, an optimal strategy of the STG might be producing the first few tokens sampled from the task-specific π_a and the remaining tokens from the PLM π_{LM}. The curve supports this interpretation since the score is decreased as c is close to 1. Our STG learns such a strategy as shown from the generated answers in Table 7 and 8. In Data-to-Text, the BLEU score is increased as c is close to 1 while the ERR score is decreased. This fact supports the results of NE(mix) models as discussed in Section 4.6. The π_s learns to balance between the BLEU and ERR.

C Generated Sentence Examples

Here, we show generated sentence examples for each task (see Table 6 for Data-to-Text, Table 7 and Table 8 for Question Answering and Table 9 and Table 10 for Summarization.). The tokens sampled from the task-specific policy π_a are presented in red.

[5] We add "-RL" to the STG to distinguish with STG-MLE in this context.

[6] The curve for Data-to-Text is not presented since there is no actual validation set.

Hyper-parameter	Summarization	Data-to-Text	Question Answering
Num layer		2	
RNN hidden size	512	256	256
γ		1	
Optimizer		AdamW with $betas = (0.9, 0.999)$, $eps = 10^{-8}$	
Learning rate		2e-5	5e-5
Pre-train epoch (Non-STG-RL)	1	0	1
Validation data size	500	50	500
Train epochs	25 (50 shot), 20 (100 shot), 15 (300 shot), 8 (1,500 shot), 4 (3,000 shot)	30	50 (50 shot), 40 (100 shot), 20 (300 shot), 10 (1,000 shot), 5 (2,000 shot)
Batch size	16	10	16

Table 5: **Hyper-parameters used for experiments**

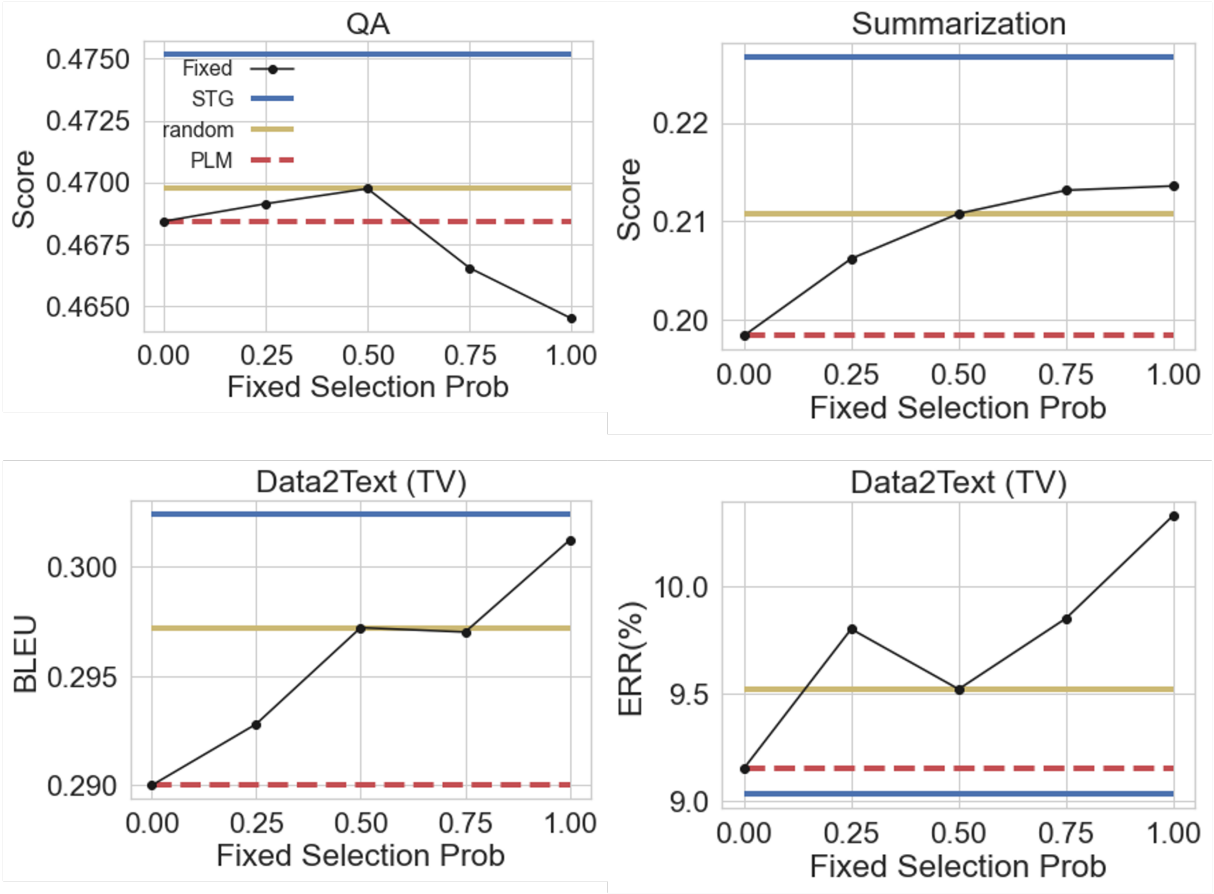

Figure 5: Dynamic selection vs Fixed selection.

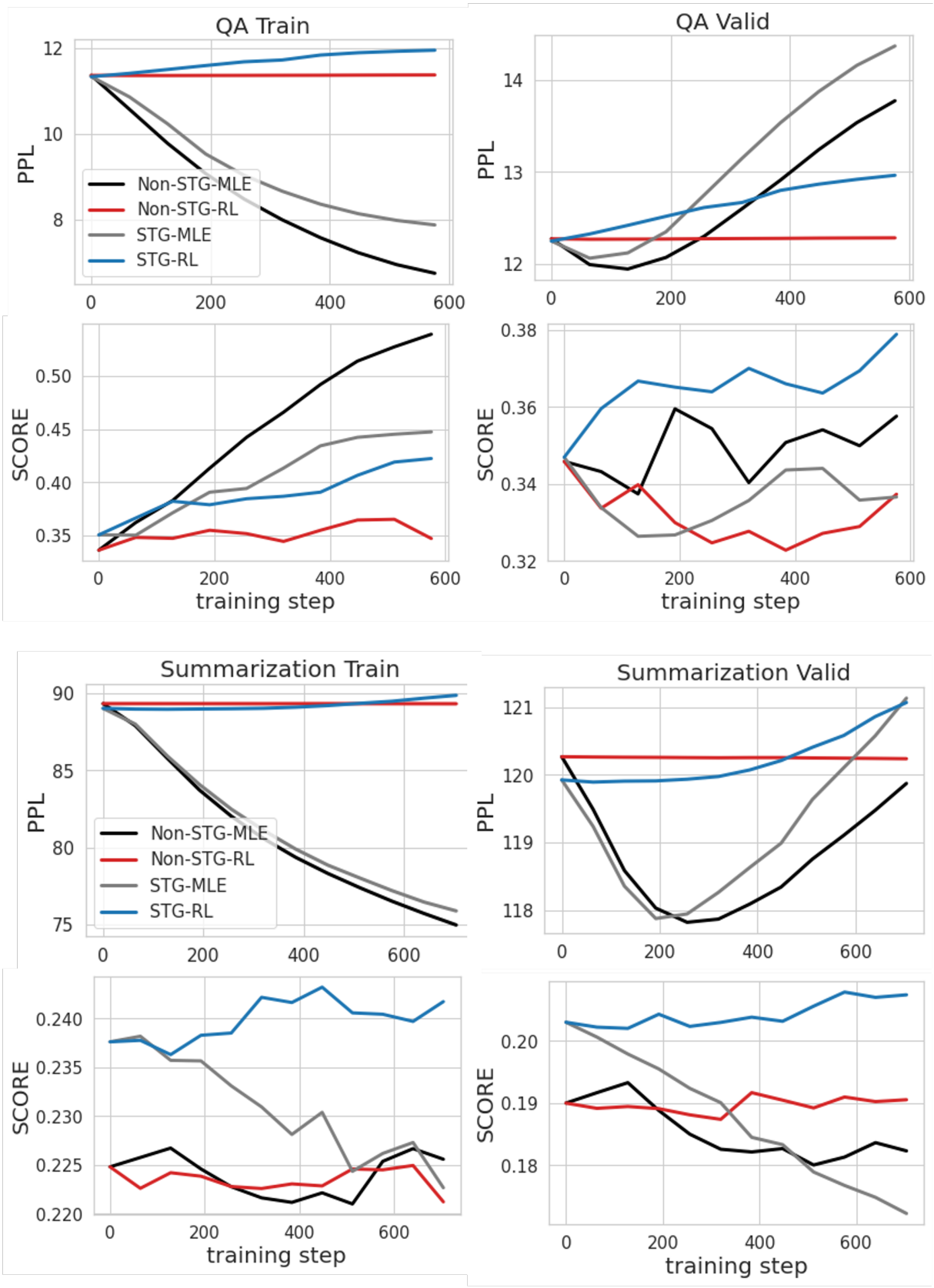

Figure 6: The learning curve. The Perplexity (PPL) and Score of each task (e.g. Rouge-L for Summarization) are measured on the 0.5% few-shot train set and the valid set.

Data	recommend(name=kratos 31; type=television; powerconsumption=31.4 watt; accessories=remote control and european warranty; color=mat black bezel with mat black frame stand)
Ground-truth	the kratos 31 is a great television . it only consumes 31.4 watts , comes with a remote control and european warranty and has a stylish mat black bezel with mat black frame stand .
PLM	the kratos 31 features a mat black bezel with mat black frame stand and european warranty.
Non-STG-MLE	the kratos 31 features a mat black bezel with mat black frame stand and european warranty
NE(mix)-MLE	the kratos 31 television has 31.4 watt power consumption, remote control and european warranty.
STG	the kratos 31 television has 31 watt power consumption. it comes with a remote control and european warranty with a color of mat black bezel with mat black frame stand.
Data	inform(name=triton 52; type=television; family=l7; audio=cevo stereo; screensize=47 inch)
Ground-truth	the triton 52 47 inch screen television from the l7 product family and has cevo stereo audio.
PLM	the triton 52 television is a 47 inch television in the l7 family and with a cevo stereo.
Non-STG-MLE	the triton 52 television is in the l7 family, it has 47 inches of screen, and uses cevo stereo sound capabilities.
NE(mix)-MLE	the triton 52 television is in the l7 family, it has 47 inches of screen, and a 47 euro price,
STG	the triton 52 television is a 47 inch television in the l7 family and has a cevo stereo.
Data	inform(name=attis 65; type=television; resolution=1080p; pricerange=cheap)
Ground-truth	the attis 65 television is in the cheap price range and has a nice resolution of 1080p.
PLM	the attis 65 television in the cheap price range has a 1080p resolution and a pricerange of cheap.
Non-STG-MLE	the attis 65 television has a 1080p resolution, is in the cheap price range and has a nicam stereo.
NE(mix)-MLE	the attis 65 television is a 1080p television in the cheap price range and in the cheap price range with a nicam stereo.
STG	the attis 65 television is in the cheap price range and has a 1080p resolution.

Table 6: Data-to-Text examples from TV of FewShotWOZ dataset.

Passage	zygote : this cell results from the union of an oocyte and a sperm . a zygote is the beginning of a new human being (i.e. , an embryo) . the expression fertilized ovum refers to a secondary oocyte that is impregnated by a sperm ; when fertilization is complete , the oocyte becomes a zygote.10 (emphasis added .)
Query	what is a human zygote
Ground-truth	the beginning of a new human being .
PLM	the expression fertilized ovum refers to a secondary oocyte that is impregnated by a sperm ;
NE(mix)-MLE	the expression fertilized ovum refers to a secondary oocyte that is impregnated by a sperm ; when fertilization is complete , the oocyte becomes a zygote .
NE(mix)-RL	the expression fertilized ovum refers to a secondary oocyte that is impregnated by a sperm ; when fertilization is complete , the oocyte becomes a zygote .
STG	it is the beginning of a new human being .
Passage	sometimes the terms reactive airway disease and asthma are used interchangeably , but they are not the same thing . often , the term reactive airway disease is used when asthma is suspected , but not yet confirmed . reactive airway disease in children is a general term that does n't indicate a specific diagnosis . it may be used to describe a history of coughing , wheezing or shortness of breath triggered by infection . these signs and symptoms may or may not be caused by asthma .
Query	symptoms of reactive airway disease in adults
Ground-truth	coughing , wheezing or shortness of breath triggered by infection .
PLM	symptoms of reactive airway disease in children is a general term that does nt́ indicate a specific diagnosis .
NE(mix)-MLE	symptoms of reactive airway disease in adults are a general term that does nt́ indicate a specific diagnosis .
NE(mix)-RL	yes , reactive airway disease in children is a general term that does nt́ indicate a specific diagnosis .
STG	coughing , wheezing or shortness of breath triggered by infection .
Passage	sulfate is an anion , composed of a sulfur atom and four oxygen atoms . the overall charge of this anion is -2. the molecular formula for sulfate is so 4 2-. there , two oxygen atoms are bonded to the sulfur atom through double bonds and other two are bonded to sulfur through single bonds . the oxidation state of sulfur in sulfate is +6 .
Query	is sulfate sulfur and oxygen bonded
Ground-truth	yes , sulfate is sulfur and oxygen bonded .
PLM	no answer present .
NE(mix)-MLE	yes , sulfate is sulfur and oxygen bonded to the sulfur atom through double bonds and other two are bonded to sulfur through single bonds .
NE(mix)-RL	yes , sulfate sulfur and oxygen bonded to the sulfur atom through double bonds and other two are bonded to sulfur through single bonds .
STG	yes , sulfate is sulfur and oxygen bonded to each other through double bonds and other two are bonded to sulfur through single bonds .

Table 7: Question Answering examples.

Passage	gryffindor house is one of the four houses at hogwart into which the students are sorted when they start their first year . gryffindor is the hogwarts house of founder godric gryffindor . gryffindor house places high value on traits such as courage , chivalry , and nerve . during harry ' s time at hogwarts , the head of gryffindor house was minerva [. . .]
Query	what kind of house is a gryffindor
Ground-truth	gryffindor house is one of the four houses at hogwart into which the students are sorted when they start their first year .
PLM	gryffindor house is one of the four houses at hogwart into which the students are sorted when they start their first year .
NE(mix)-MLE	no answer present .
NE(mix)-RL	no answer present .
STG	gryffindor house is one of the four houses at hogwarts into which the students are sorted when they start their first year .
Passage	the median home value in louisville is $ 119,500 . louisville home values have gone up 1.8 % over the past year and zillow predicts they will rise 3.0 % within the next year . the median list price per square foot in louisville is $ 125 . the median price of homes currently listed in louisville is $ 180,000 . the median rent price in louisville is $ 1,050 . foreclosures will be a factor impacting home values in the next several years . in louisville 0.1 homes are foreclosed (per 10,000) .
Query	average per therm price in louisville ky
Ground-truth	the median home value in louisville is $ 119,500 .
PLM	the average per therm price in louisville is $ 119,500 .
NE(mix)-MLE	$ 125 per square foot
NE(mix)-RL	the average per square foot in louisville is $ 125 .
STG	the median home value in louisville is $ 119,500 .
Passage	the brain is made up of more than just nerve cells (neurons) . although there are about 100 billion neurons in the brain , there may be about 10 to 50 times that many glial cells in the brain . neurons have two processes called axons and dendrites .... glial cells have only one . 2 neurons can generate action potentials ... glial cells can not . 3 however , glial cells do have a resting potential . 4 neurons have synapses that use neurotransmitters ... glial cells do not have chemical synapses .
Query	number of glial cells vs neurons
Ground-truth	10 to 50 times .
PLM	no answer present .
NE(mix)-MLE	$ 10 to 50 times that many glial cells in the brain .
NE(mix)-RL	no answer present .
STG	10 to 50 times

Table 8: Question Answering examples.

Article	-lrb- cnn -rrb- – three runners died sunday during the detroit free press/flagstar marathon in detroit, michigan, police told cnn. an emt vehicle is at the scene sunday in detroit after three runners collapsed at a marathon. all three deaths occurred between 9 and 9:20 a.m. et, second deputy chief john roach said. a man in his 60s fell and hit his head, roach said. the cause of the fall was unknown. the man was transported to detroit receiving hospital, where he was pronounced dead. two other men, ages 36 and 26, also collapsed during the race and were pronounced dead at the hospital, roach said. all three collapsed near the end of the race, [...]
Ground-truth	second deputy chief john roach : all three deaths occurred between 9 and 9:20 a.m. man in his 60s fell hit his head ; two men others , ages 36 and 26 , collapsed . race was detroit free press/flagstar marathon in detroit , michigan .
PLM	three runners collapsed at a marathon in detroit , police say . the cause of the fall is unknown .
Non-STG-MLE	three runners collapsed at a marathon sunday , police say . the cause of the fall is unknown , police say .
Non-STG-RL	three runners collapsed at a marathon sunday , police say . the cause of the fall is unknown , police say .
STG	three runners collapsed at a marathon in detroit , michigan . all three deaths occurred between 9 and 9:20 a.m. et . a man in his 60s fell and hit his head , police say .
Article	london, england -lrb- cnn -rrb- – up to 1,000 human rights campaigners demonstrated saturday in front of no. 10 downing street, the official residence of british prime minister gordon brown, calling on the british government to demand that full democracy be restored in pakistan. jemima khan, center, ex-wife for former pakistani cricket star imran khan, joins protesters in london. protesters waved placards and chanted in support of the resignation of pakistani president pervez musharraf, a week after he imposed a state of emergency in the country. the crowd of demonstrators massed behind barriers and included jemima khan, the ex-wife of former pakistani cricket star turned politician imran khan. the demonstrators carried placards saying “ free the innocent ” and “ end musharraf’s regime ” and waved pakistani flags. imran khan, who heads the the movement for justice party, has been under house arrest since the emergency declaration. his ex-wife delivered a petition to a doorman at downing street, [...]
Ground-truth	human rights campaigners demonstrate in front of no . 10 downing street . protests urged uk government to demand full democracy restored in pakistan . cricketer turned politician imran khan ś ex wife jemima among protesters .
PLM	pakistani president pervez musharraf has been under house arrest since the emergency declaration . his ex-wife delivered a petition to a doorman at downing street .
Non-STG-MLE	pakistani president pervez musharraf has imposed a state of emergency in the country . he has been under house arrest since the emergency declaration .
Non-STG-RL	pakistani president pervez musharraf has imposed a state of emergency in the country . he has imposed a state of emergency in the country since last week .
STG	the official residence of british prime minister gordon brown . hundreds of protesters demonstrate in front of no . 10 downing street . the petition calls for full democracy in pakistan .

Table 9: Text Summarization examples.

Article	-lrb- cnn -rrb- spoiler alert! it's not just women getting cloned. that was the big twist at the end of " orphan black's " second season. the kickoff to the new season leads the list of six things to watch in the week ahead. 1. " orphan black, " 9 p.m. et, saturday, april 18, bbc america. the cloning cult sci-fi series remains one of the most critically acclaimed shows on tv, thanks in large part to the performance of tatiana maslany, who has taken on at least six roles on the show so far, [...] 2. " turn : washington's spies, " 9 p.m. et, monday, amc. the series about spies in the early days of the revolutionary war returns with a new subtitle, " washington's spies, " and a new monday night time slot. [...] 3. " game of thrones, " 9 p.m. et, sunday, hbo. the world of westeros returns for a fifth season in one of the biggest season premieres of the year. click here for more on what to expect. [...]
Ground-truth	critically acclaimed series " orphan black " returns . " turn : washington ś spies " starts a second season . " game of thrones " is back for season five .
PLM	series returns with a new subtitle , " washington ś spies , " and a new monday night time slot . series star jamie bell told cnn what we can expect in the second season .
Non-STG-MLE	" orphan black " returns with a new subtitle and a new monday night time slot . series star tatiana maslany is blown away by the response to the series so far .
Non-STG-RL	" orphan black " returns with a new subtitle and a new monday night time slot . series star jamie bell says the series will bring more recognition to the culper spy ring .
STG	the new season of " orphan black " premieres on saturday the 18th . " game of thrones " returns for a fifth season .
Article	energy drinks can pose a real problem for children, according to a new study from the american heart association. researchers found that 40 % of the 5,156 calls to poison centers for " energy drink exposure " involved children under age 6. in most of the cases, the parents didn't know the children had gotten hold of an energy drink. many of the calls reported the children were experiencing serious side effects, such as an abnormal heart rhythm, or they were having a seizure. the study is being presented at the american heart association's scientific sessions this week. study : caffeine affects teen boys more. study author dr. steven lipshultz has handled cases involving children who became sick after consuming energy drinks. while studies about the impact of caffeine on children are limited, lipshultz, [...]
Ground-truth	40 % of " energy drink exposure " calls to poison control centers are for children under 6. study authors believe the number is under-counted . the fda does not set a caffeine limit for children .
PLM	drug calls to poison centers for " energy drink exposure " in most cases , the parents did n't know the children had gotten hold of an energy drink . many of the calls reported the children were experiencing serious side effects , such as an abnormal heart rhythm .
Non-STG-MLE	drug calls to poison centers for " energy drink exposure " in most cases , the parents did n't know the children had gotten hold of an energy drink . many of the calls reported the children were experiencing serious side effects , such as an abnormal heart rhythm .
Non-STG-RL	" this is a very concerning finding , " dr. laurence sperling says . the american academy of pediatrics recommends children consume no caffeine .
STG	drug calls to poison centers for " energy drink exposure " nearly 40 % of calls to poison centers for " energy drink exposure " involved children under age 6. study : caffeine affects teens more .

Table 10: Text Summarization examples.

A-TIP: Attribute-aware Text Infilling via Pre-trained Language Model

Dongyuan Li, Jingyi You, Kotaro Funakoshi, Manabu Okumura
Tokyo Institute of Technology
{lidy,youjy,funakoshi,oku}@lr.pi.titech.ac.jp

Abstract

Text infilling aims to restore incomplete texts by filling in blanks, which has attracted more attention recently because of its wide application in ancient text restoration and text rewriting. However, attribute-aware text infilling is yet to be explored, and existing methods seldom focus on the infilling length of each blank or the number/location of blanks. In this paper, we propose an **A**ttribute-aware **T**ext **I**nfilling method via a **P**re-trained language model (A-TIP), which contains a text infilling component and a plug-and-play discriminator. Specifically, we first design a unified text infilling component with modified attention mechanisms and intra- and inter-blank positional encoding to better perceive the number of blanks and the infilling length for each blank. Then, we propose a plug-and-play discriminator to guide generation towards the direction of improving attribute relevance without decreasing text fluency. Finally, automatic and human evaluations on three open-source datasets indicate that A-TIP achieves state-of-the-art performance compared with all baselines.

1 Introduction

Originating from Cloze tests (Taylor, 1953), text infilling aims to fill in missing blanks in a sentence or paragraph by making use of the preceding and subsequent texts. For example, given two infilling tasks E1 and E2 in Fig.1, text infilling models are supposed to provide fine-grained control over the location of any number of blanks and infill a variable number of missing tokens for each blank. Text infilling has been gaining increasing attention in a number of prevailing research fields, including ancient text restoration (Lazar et al., 2021), text editing and rewriting (Su et al., 2021), and conversation generation (Ou et al., 2021).

However, current text infilling methods are based only on bidirectional semantic constraints (Ou et al., 2021), and other abundant attribute-based constraints, e.g., sentiment and topics, remain to be studied. In reality, infilling attribute-aware content can better satisfy human needs and introduce more diversity. For instance, as shown in Fig.1, A-TIP can fill in blanks under the guidance of an attribute to satisfy sentiment or expert knowledge infilling, while current text infilling models mainly focus on fluency, which leads to meaningless and monotonous infilling contents (Sun et al., 2021).

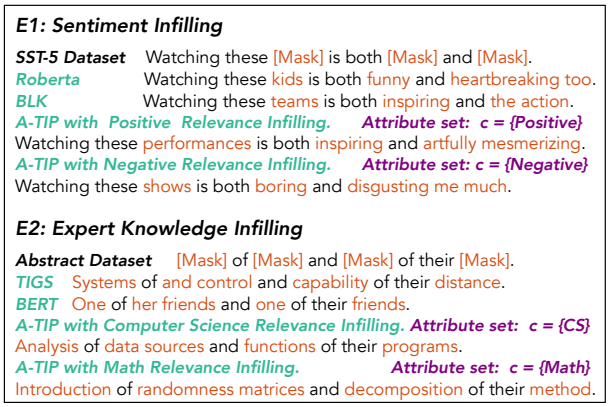

Figure 1: A-TIP can generate more fluent, diverse and attribute relevant infilling content in two examples.

Designing a simple but efficient attribute-aware text infilling model is a challenging task. First, to achieve attribute awareness, simply modifying a text infilling model architecture or fine-tuning with attribute-specific data will destroy the model's ability to infill blanks or require a significant cost for re-training (Dathathri et al., 2020). Second, if the model infills blanks towards the direction of improving text attributes, avoiding ill-formedness between infilling content and its bidirectional context becomes a challenge. For instance, "The movie interesting and perfect us" with _ as blanks. Finally, current methods lack fine-grained control over automatic determination of the number/location of blanks or the infilling length for each blank. For example, Markov assumption-based models (Liu et al., 2019; Zaidi

Proceedings of the 29th International Conference on Computational Linguistics, pages 5857–5869
October 12–17, 2022.

et al., 2020) hardly adapt to variable infilling lengths, while masked language model (MLM)-based methods (Devlin et al., 2019; Liu et al., 2020) are incapable of generating more than one word per blank, and generative LM-based methods (Donahue et al., 2020) cannot guarantee the output will match the number of missing blanks in the input.

To circumvent the above dilemma, in this paper, we propose an **A**ttribute-aware **T**ext **I**nfilling method based on a **P**re-trained LM (A-TIP), in which a plug-and-play discriminator provides fine-grained control over bidirectional well-formed fluency and attribute relevance.[1] Specifically, 1) we first propose a general text filling framework that fine-tunes a standard LM with many artificially-masked examples in an auto-regressive manner. Moreover, to ensure that the number of infilling contents equals the number of blanks, we design a new attention mechanism, where unmasked tokens can attend to each other but masked tokens can attend only to the preceding context (Fig.2 (A)). We also adopt two-level positional encoding to combine inter- and intra-blank positional information to automatically learn the length of blanks. 2) To achieve attribute-aware generation without modifying LM's architecture or re-training, we propose a plug-and-play discriminator that shifts the output distribution of the text infilling model towards the semantic space of given guide attributes. We also design two additional strategies to ensure the infilling content is well-formed with its bidirectional context without decreasing attribute relevance. The main contributions are summarized as follows:

• We propose a unified text infilling model that adopts a new attention mechanism and two-level positional encoding to enable our model to learn the number/location of blanks and infilling length for each blank automatically.

• To the best of our knowledge, A-TIP is the first attribute-aware text infilling model that does not require any modification of the language model's architecture or re-training on specific attributed datasets. Further, our plug-and-play discriminator can provide fine-grained control over fluency and attribute relevance, and can be applied to any transformer decoder-based text infilling model.

• The experimental results on three open datasets show that A-TIP achieves state-of-the-art performance compared with all baselines.

[1]Sentences with higher accuracy of attribute-based classification are said to have higher attribute relevance (Dathathri et al., 2020).

2 Related Work

In this section, we briefly review the most relevant studies to our work on pre-trained LMs, text infilling, and constrained text generation.

2.1 Pre-trained Language Models

Pre-trained LMs have made significant improvements in many natural language processing tasks by adopting self-supervised learning with abundant web texts (Chay-intr et al., 2021; You et al., 2022). They can be classified into three types. The first uses an auto-encoding model. For example, BERT (Devlin et al., 2019) and its variations are pre-trained as masked LMs to obtain bidirectional contextualized word representations. The second adopts an encoder-decoder architecture, which is pre-trained for seq2seq tasks, such as MASS (Song et al., 2019) and T5 (Raffel et al., 2020). The third adopts an auto-regressive model, which follows a left-to-right manner for text generation, such as GPT-2 (Radford et al., 2019) and XLNet (Yang et al., 2019). While we adopt GPT-2 as the LM in this paper, our method can be easily migrated to any type of pre-trained LMs.

2.2 Text Infilling Approaches

Current text infilling algorithms can be classified into four categories. *Generative adversarial networks (GAN)-based methods* train GANs to ensure that the generator can generate highly reliable infilling content to fool the discriminator (Guo et al., 2018; Fedus et al., 2018). *Intricate inference-based methods* adopt dynamic programming or gradient search to find infilling content that has a high likelihood within its surrounding context (Liu et al., 2019; Zaidi et al., 2020). *Masked LM-based methods* generate infilling content on the basis of its bidirectional contextual word embedding (Devlin et al., 2019; Shen et al., 2020). *LM-based methods* fine-tune off-the-shelf LMs in an auto-regressive manner, and a number of methods change the input format by putting an infilling answer after the masked input (Donahue et al., 2020), while others do not change the input format (Zhu et al., 2019). Unlike the aforementioned methods, we solve a more complex task: attribute-aware text infilling.

2.3 Constrained Text Generation

Traditional controlled generation models involve either fine-tuning existing models (He, 2021) or training conditional generative models (Keskar et al.,

2019). Dathathri et al. (2020) proposed a plug-and-play controlled generation model (PPLM), which does not modify or re-train the parameters of the original LM but can achieve comparable performance to fine-tuning methods. For example, PPCM (Madotto et al., 2020) updates the hidden state towards the direction of attribute enhancement to generate attribute-aware conversations. Pascual et al. (2021) designed a complex plug-and-play architecture to ensure that the generated content contains specific keywords. While GeDi (Krause et al., 2021) and its extension (Lin and Riedl, 2021) can accelerate the decoding process of PPLM, they assume the model is trained by large-scale labeled datasets, which is unrealizable for text infilling. Unlike the previous work, we should also consider the generated infilling content is well-formed with its corresponding bidirectional context, ensuring PPLM is suitable for text infilling.

3 Preliminaries

To clarify our method, we first introduce some essential background knowledge and then define the task of attribute-aware text infilling.

Language Models reveal the degree of how much a sentence (a sequence of words) is likely to be a realistic sequence of a human language. Formally, let $\mathcal{W}$ be the vocabulary set and $\boldsymbol{w}_{1:n} = \{w_1, \ldots, w_n\}$ is a sentence with n words, where $w_i \in \mathcal{W}$. An LM measures the joint probability by decomposing the sequence one by one:

$$p(\boldsymbol{w}_{1:n}) = \prod_{i=1}^{n} p(w_i|\boldsymbol{w}_{<i}), \tag{1}$$

where $\boldsymbol{w}_{<i} = \{w_1, \ldots, w_{i-1}\}$.

Constrained Text Generation: Given k explicit constraints $\mathbf{c} = \{c_1, \ldots, c_k\}$, our goal is to generate a sentence $\boldsymbol{w}$ that maximizes the conditional probability $p(\boldsymbol{w}|\mathbf{c})$:

$$p(\boldsymbol{w}|\mathbf{c}) = \prod_{i=1}^{n} p(w_i|\boldsymbol{w}_{<i}, \mathbf{c}). \tag{2}$$

Task Definition: Attribute-aware text infilling is to take incomplete text $\widetilde{\boldsymbol{w}}$, containing one or more missing blanks, and return completed text $\boldsymbol{w}$ under the constraints of $\mathbf{c}$. As in Fig.1, several attributes are listed in $\mathbf{c}$. Specifically, let [Mask] be a placeholder for a contiguous sequence of one or more missing tokens. Then, $\widetilde{\boldsymbol{w}}$ is a sequence of tokens in which a number of them are [Mask]. To map $\widetilde{\boldsymbol{w}}$ to $\boldsymbol{w}$, constrained with attribute $\mathbf{c}$, an infilling strategy must specify both how many and which tokens to generate for each [Mask]. Note that there may be many logical $\boldsymbol{w}$ for a given $\widetilde{\boldsymbol{w}}$. Hence, we are interested in learning a distribution $p(\boldsymbol{w}|\widetilde{\boldsymbol{w}}, \mathbf{c})$. Specifically, in accordance with Bayes' theorem, we formulate the probability of predicting the token w_i for its corresponding [Mask] as:

$$p(w_i|\boldsymbol{w}_{<i}, \mathbf{c}) \propto p(w_i|\boldsymbol{w}_{<i}) \cdot p(\mathbf{c}|\boldsymbol{w}_{1:i}), \tag{3}$$

where $p(w_i|\boldsymbol{w}_{<i}, \mathbf{c})$ can be decomposed into two parts that deal with the LM for $p(w_i|\boldsymbol{w}_{<i})$ and the discriminator for $p(\mathbf{c}|\boldsymbol{w}_{1:i})$. In Section 4, we introduce these two parts in detail. We assume that any two constraints are independent: $p(\mathbf{c}|\boldsymbol{w}_{1:i}) = \prod_{j=1}^{k} p(c_j|\boldsymbol{w}_{1:i})$.

4 Methodology

The overall framework of A-TIP is shown in Fig.2. A-TIP contains two components: a *text infilling model* and a *plug-and-play attribute-aware controller*.

4.1 Text Infilling Model

Given a corpus consisting of complete text examples, we first create *infilling examples* and then train the GPT-2 with these examples. Specifically, given an input example $\boldsymbol{w}_{1:n}$ with n tokens, we first randomly replace m non-overlapping word spans $\mathbf{S} = \{\boldsymbol{s}_1, \ldots, \boldsymbol{s}_m\}$ in $\boldsymbol{w}$ with [Mask] tokens to form a corrupted text $\widetilde{\boldsymbol{w}}$. We also assume each span $\boldsymbol{s}_i$ contains n_i consecutive tokens $[s_{(i,1)}, \ldots, s_{(i,n_i)}]$. Then, we concatenate the spans S separated by [Answer] tokens to form a training target $\widetilde{\mathbf{S}}$ = {[Answer], $s_{(1,1)}, \ldots, s_{(1,n_1)}$, [Answer], ..., [Answer], $s_{(m,1)}, \ldots, s_{(m,n_m)}$ }. Finally, we construct a complete infilling example by concatenating $\widetilde{\boldsymbol{w}}$ and $\widetilde{\mathbf{S}}$ (see Token Embedding in Fig.2).

There are two advantages of designing such an input format. First, we add only $2m$ additional tokens (one [Mask] and one [Answer] per blank as shown in Fig.2 "Token Embedding" add 4 tokens for two spans). Although memory usage for GPT-2 grows quadratically with sequence length, as m is small, additional training time complexity will be minimal. Second, we can apply two different attention strategies for the corrupted text $\widetilde{\boldsymbol{w}}$ and training target text $\widetilde{\boldsymbol{S}}$. As shown in Fig.2 (A), while tokens in the corrupted text have attentions on all other tokens in $\widetilde{\boldsymbol{w}}$, tokens in the training target can have attentions only on its previous tokens. By

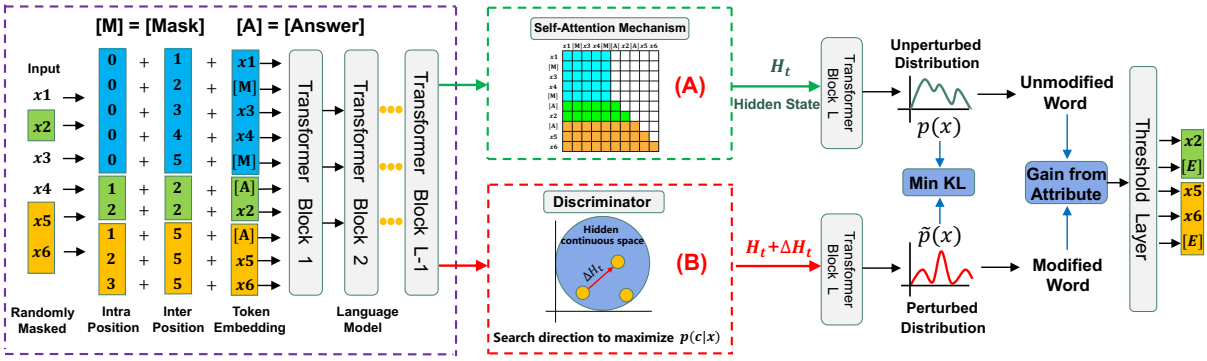

Figure 2: Model overview. We first fine-tune an off-the-shelf GPT-2 by adopting a new attention mechanism and two-level positional encoding to infill blanks. Then, we design a plug-and-play discriminator to guide generation in the direction of improving attribute relevance. We also adopt KL divergence and a threshold-based strategy to provide fine-grained control over fluency and attribute relevance.

adopting such an attention mechanism, when A-TIP infills the i-th blank $\boldsymbol{s}_i$, it will focus on the bidirectional context of the i-th blank, which can ensure the well-formedness and rationality of the whole sentence.

Current methods hardly perceive the number/location and infilling length for each blank. We design two-level positional encoding, which can provide fine-grained control over them. Specifically, each token is encoded with two position IDs. The first position ID represents the inter-position in the corrupted text $\widetilde{\boldsymbol{w}}$ and the second position ID represents the intra-position in each span.

Finally, A-TIP trains the GPT-2 with the *infilling examples* in an auto-regressive manner. When predicting missing tokens in each blank, A-TIP has access to the corrupted text $\widetilde{\boldsymbol{w}}$ and the previously predicted blanks. Formally, the probability of generating the i-th blank $\boldsymbol{s}_i$ is

$$p_\theta(\boldsymbol{s}_i|\widetilde{\boldsymbol{w}}, \boldsymbol{s}_{<i}) = \prod_{j=1}^{n_i} p(s_{i,j}|\widetilde{\boldsymbol{w}}, \boldsymbol{s}_{<i}, \boldsymbol{s}_{i,<j}), \quad (4)$$

where θ are parameters for the GPT-2, n_i represents the number of tokens in $\boldsymbol{s}_i$, $s_{i,j}$ represents the j-th token in the span $\boldsymbol{s}_i$, $\boldsymbol{s}_{<i}$ represents previously predicted blanks, and $\boldsymbol{s}_{i,<j} = \{s_{i,1}, \cdots, s_{i,j-1}\}$.

4.2 Plug-and-play Attribute-aware Controller

To clarify our approach, we follow the notation of Dathathri et al. (2020) and define the GPT-2 decoding process (Eq.(4)) in a recursive manner. Specifically, we first define $\mathbf{H}_t$, that contains all historical key-value pairs, i.e., $\mathbf{H}_t = [(\mathbf{K}_t^{(1)}, \mathbf{V}_t^{(1)}), \ldots, (\mathbf{K}_t^{(l)}, \mathbf{V}_t^{(l)})]$, where $(\mathbf{K}_t^{(l)}, \mathbf{V}_t^{(l)})$ stores all key-value pairs of t tokens in the l-th layer. Then, we formally define the recurrent decoding process to generate the i-th token as:

$$\boldsymbol{o}_i, \mathbf{H}_i = \text{GPT-2}(\boldsymbol{w}_{<i}, \mathbf{H}_{i-1}), \quad (5)$$

where $\boldsymbol{o}_i$ is the hidden state of the input at i-th time-step. Then, we sample the i-th generated token from the following distribution by beam search (Hokamp and Liu, 2017):

$$w_i \sim p_i = \text{Softmax}(\mathbf{W}\boldsymbol{o}_i), \quad (6)$$

where $\mathbf{W}$ is a parameter matrix that maps the hidden state $\boldsymbol{o}_i$ to a vector of the vocabulary size.

In accordance with Bayes' theorem in Eq.(3), we have $p(w_i|\boldsymbol{w}_{<i}, \mathbf{c}) \propto p(w_i|\boldsymbol{w}_{<i}) \cdot p(\mathbf{c}|\boldsymbol{w}_{1:i})$. To achieve attribute-aware text infilling, when we infill the i-th blank, we shift history matrix $\mathbf{H}_{i-1}$ towards the direction of the sum of two gradients: 1) To maximize the log-likelihood of the attribute $\mathbf{c}$ under the conditional attribute model $p(\mathbf{c}|\boldsymbol{w}_{1:i})$ and 2) To ensure high fluency of text infilling $p(w_t|\boldsymbol{w}_{<i})$. We update only $\mathbf{H}_{i-1}$ and fix other model parameters unchanged since next-token prediction depends only on the past key-value pairs via $\mathbf{H}_{i-1}$. Thus, we propose to gradually update $\mathbf{H}_{i-1}$ to guide future generation in the desired direction.

Let $\Delta\mathbf{H}_{i-1}$ be the update to $\mathbf{H}_{i-1}$ to shift the generation infilling content towards the desired attribute direction $\mathbf{c}$. At the beginning of the generation, $\Delta\mathbf{H}_{i-1}$ is initialized to zero, and we can obtain the unmodified distribution as p_i. Then, we update $\Delta\mathbf{H}_{i-1}$ with gradients from the attribute model that measures the extent to which the generated text possesses the desired attribute. Following Dathathri et al. (2020), we rewrite $p(\mathbf{c}|\boldsymbol{w}_{1:i})$ as Pb = $p(\mathbf{c}|\mathbf{H}_{i-1} + \Delta\mathbf{H}_{i-1})$ and define the gradient up-

date for $\Delta \mathbf{H}_{i-1}$ as

$$\Delta \mathbf{H}_{i-1} \leftarrow \Delta \mathbf{H}_{i-1} + \alpha \frac{\nabla_{\Delta \mathbf{H}_{i-1}} Pb}{\|\nabla_{\Delta \mathbf{H}_{i-1}} Pb\|^{\gamma}}, \quad (7)$$

where α is the learning rate and γ is the scaling coefficient for the normalization term to control the relevance of the attribute. We repeat Eq.(7) less than 10 times to generate attribute-aware tokens. Subsequently, the new $\widetilde{\mathbf{H}}_{i-1} = \mathbf{H}_{i-1} + \Delta \mathbf{H}_{i-1}$ is computed, and a new token is generated using $\widetilde{\boldsymbol{o}}_i, \mathbf{H}_i = \text{GPT-2}(\boldsymbol{w}_{<i}, \widetilde{\mathbf{H}}_{i-1})$. The described optimization process is repeated for every token in the generated sequence. Compared with the unconditional LM-based text generation task, this process will not take much time (see detail in experiments).

Although we can generate attribute-aware infilling content, we can easily generate low-quality, repetitive, and low-fluency text. Thus, we add two additional components to ensure the fluency and quality of generated infilling content with its bidirectional context. First, we minimize the KL divergence between the unmodified distribution p_i and modified distribution $\widetilde{p}_i$ for the i-th token:

$$\min \ \ \mathrm{D}_{KL}(\widetilde{p}_i \,||\, p_i). \quad (8)$$

Our objective function can be reformulated as

$$Loss = p(\mathbf{c}|\widetilde{\mathbf{H}}_{i-1}) + \lambda \mathrm{D}_{KL}(\widetilde{p}_i \,||\, p_i), \quad (9)$$

where λ is a parameter to balance the fluency and attribute relevance. Then, we update $\Delta \mathbf{H}_{i-1}$ as:

$$\Delta \mathbf{H}_{i-1} \leftarrow \Delta \mathbf{H}_{i-1} + \alpha \frac{\partial_{\Delta \mathbf{H}_{i-1}} Loss}{\|\partial_{\Delta \mathbf{H}_{i-1}} Loss\|^{\gamma}}. \quad (10)$$

Intuitively, we can generally find many words that have different levels of correlations with the specific attribute (Mohammad, 2018). For example, {perfect, good, bad, like} can mainly determine the sentiment of a sentence. Thus, we define *Gain* from the attribute to determine whether to change a generated word. As shown in Fig.2, two candidate words are sampled from the unmodified distribution (before back propagation) and modified distribution (after back propagation), respectively. *Gain* between two candidate words in the conditional model can be formulated as

$$Gain = p(\mathbf{c}|\boldsymbol{w}_{<i}, \widetilde{w}_i) - p(\mathbf{c}|\boldsymbol{w}_{<i}, w_i), \quad (11)$$

where $\widetilde{w}_i$ and w_i are samples from the modified and unmodified distributions, respectively.

To better control the relevance of the attribute, we define a threshold δ to determine whether to generate a word from the modified distribution. Specifically, *Gain* >δ represents that the word generated from the modified distribution can have a relatively remarkable effect on attributes. Otherwise, if the discriminator does not guide well at certain steps (*Gain* <δ), we select the word generated from the unmodified distribution to maintain the fluency to be the same as the original unconditional text infilling model to the greatest extent.

Discriminator Construction: As shown in Fig.2 (B), for simplicity, we train a linear classifier f as a discriminator with annotated datasets, indicating a sentence and label pair as $(\boldsymbol{w}, \boldsymbol{y})$. Specifically, for each sentence $\boldsymbol{w}$ of length t, we compute the set of hidden states $\boldsymbol{o} = \{o_1, \ldots, o_t\}$ from the GPT-2. Then, we compute the mean of $\boldsymbol{o}$ as $\bar{\boldsymbol{o}}$ and train f using the cross-entropy between the true label distribution $\boldsymbol{y}$ and predicted label distribution $f(\bar{\boldsymbol{o}})$. The number of parameters in this layer is (embedding dimension $\times$ number of attributes + number of attributes), which is negligible compared with the number of parameters in the text infilling model itself.

5 Experimentation

As shown in Table 1, we evaluated the proposed methods on three tasks to demonstrate that our framework is not custom tailored to a single domain: sentiment-aware, domain knowledge-aware, and topic-aware text infilling. We also show a case study for these tasks. We determined whether A-TIP can generate infilling text that satisfies the desired attribute and whether it can infill high-quality text in blanks by using both automated methods and human annotators.

Dataset	Examples	Words	Attributes
SST-5	11,855	215,154	5
Abstracts	200K	30M	8
ROCStories	100K	5M	13

Table 1: Descriptive statistics of three datasets.

5.1 Experimental Settings

Datasets In addition to using the datasets in Table 1 to train our text infilling model, we also adopted sentiment labels in SST-5 (Pang and Lee, 2005) for sentiment-aware text infilling, research

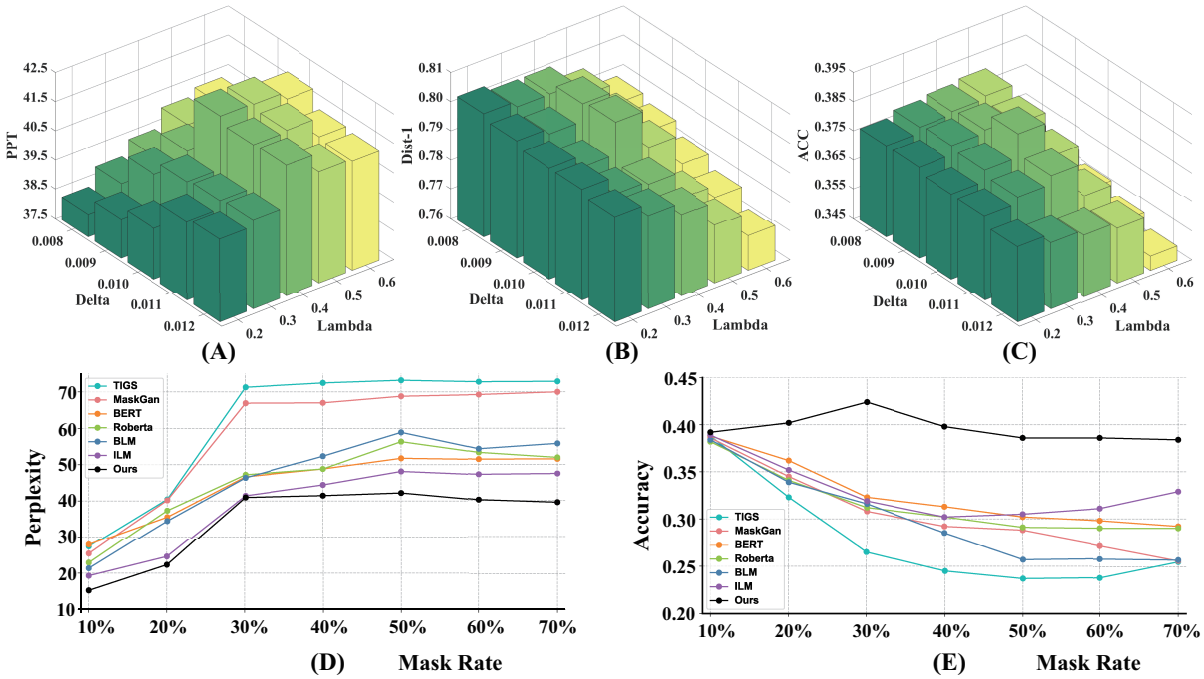

Figure 3: Based on the validation data of SST-5, we evaluated the parameter effect for Perplexity (A), Dist1 (B), and Accuracy (C). We draw the effect of mask rate on performance of text infilling for Perplexity (D) and Accuracy (E).

area labels in Abstracts (Donahue et al., 2020) for domain knowledge-aware text infilling, and topic labels in ROCStories (Mostafazadeh et al., 2016) for topic-aware text infilling. For the datasets with attribute labels like SST-5 and Abstracts, we can directly use their labels to train our plug-and-play discriminator. However, considering that most datasets do not have attribute labels, we adopted COMBINETM (Bianchi et al., 2021) to detect attributes for them (details in Appendix A). For example, for ROCStories, we can detect thirteen attributes and prove that A-TIP can generate a relevant topic in human evaluation (Table 3).

We split the datasets into $80\%/10\%/10\%$ as training/validation/test data, respectively. Following TIGS (Liu et al., 2019) and BLM (Shen et al., 2020), we randomly masked $r\%$ tokens in each document. To ensure that all experiments are performed on the same data, we removed infilling examples that exceed our training sequence length of 256 tokens.

Evaluation Metrics In automated evaluation, perplexity is a measure for fluency in open-domain text generation.[2] We measured it using GPT-2. The diversity of text was measured using the number of distinct n-grams (normalized by text length) as in Li et al. (2016). We reported Dist1, Dist2, and Dist3 scores for the distinct 1, 2, 3-grams. Following Dathathri et al. (2020), we used an external classifier to evaluate *Accuracy* (macro-average F-score) for sentence attribute labels. We evaluated the attribute control for sentiment (SST-5) with an external sentiment classifier with XLNet (Yang et al., 2019), which was trained with the IMDB dataset. We chose a BERT-based classifier (Lopes et al., 2021) for the Abstracts dataset. The t-test was used to evaluate the significant performance difference between two approaches (Yang and Liu, 1999) for both automated and human evaluations.

Baselines We compared A-TIP with six baselines that can be classified in four classes (Section 2.2): 1) **Inference-based:** We trained TIGS (Liu et al., 2019), an RNN-based seq2seq model. At inference time, we iteratively searched tokens in continuous space and projected their vectors to real words. 2) GAN-based: We trained the generator of MaskGan (Fedus et al., 2018) on PLM with a seq2seq architecture. The discriminator can make word distributions of the generator closer to those of the real word distribution. 3) **Masked LM-based:** We used representations of blanks as seeds to fine-tune BERT (Devlin et al., 2019) and Roberta (Liu et al., 2020). At inference time,

[2]Overlap-based metrics such as BLEU scores (Papineni et al., 2002) are not appropriate for evaluating infilling as there are many realistic infills that have no word-level overlap with the original.

Datasets	SST-5					Abstracts					ROCStories			
Metrics	PPL	Dist1	Dist2	Dist3	ACC	PPL	Dist1	Dist2	Dist3	ACC	PPL	Dist1	Dist2	Dist3
TIGS	73.23	0.475	0.424	0.425	0.237	49.70	0.659	0.657	0.644	0.453	63.30	0.672	0.675	0.691
MaskGan	68.83	0.385	0.758	0.728	0.288	48.82	0.652	0.662	0.642	0.494	63.32	0.677	0.671	0.701
BERT	51.76	0.773	0.732	0.732	0.302	28.86	0.683	0.656	0.624	0.508	64.16	0.673	0.636	0.560
Roberta	56.34	0.392	0.745	0.745	0.291	26.22	0.710	0.710	0.700	0.528	42.96	0.666	0.659	0.540
BLM	58.90	0.548	0.329	0.345	0.257	50.34	0.512	0.431	0.356	0.568	45.69	0.591	0.594	0.614
ILM	48.14	0.805	0.792	0.801	0.305	21.30	0.710	0.710	0.706	0.634	37.53	0.678	0.692	0.709
A-TIP/Dis	**40.26**	0.789	0.765	0.742	0.301	**18.82**	0.708	0.708	0.698	0.614	**30.35**	0.662	0.653	0.688
A-TIP/KL	51.22	0.797	0.788	0.782	**0.421**	28.97	0.711	0.711	0.706	**0.752**	47.35	0.685	0.693	0.718
A-TIP	42.21†	**0.805**†	**0.807**†	**0.808**†	0.386†	20.36†	**0.711**†	**0.711**†	**0.707**†	0.694†	32.13†	**0.685**†	**0.693**†	**0.721**†

Table 2: Overall performance comparison. PPL is perplexity, Dist scores measure divergence, and ACC is classification accuracy. † shows our results significantly surpass all baselines using t-test with p <0.005. Underlines mean our ablation algorithm can achieve better results than A-TIP for a metric.

blanks are infilled one after another and are conditioned on the previous generation. We trained BLM (Shen et al., 2020) with a seq2seq architecture, where the encoder module is a transformer (base) and the decoder process adopts beam search. 4) **LM-based:** We trained ILM (Donahue et al., 2020) by fine-tuning GPT-2 to output a full document from a masked input. Note that it may have invalid outputs that do not match the input format.

Implementation Details In our experiments, we set the learning rate $\alpha = 1e-4$ and the scaling coefficient $\gamma = 0.5$ for Eq. (10). Sequence representations were obtained by the GPT-2 module (12 layers, 12 heads, $n_{embd} = 768$, $n_{ctx} = 1024$, batch size $= 24$). We applied the Adam (Kingma and Ba, 2015) optimizer with an initial learning rate of 1e-4, and the weight decay and dropout were turned based on the loss on the validation data. Our discriminator has a linear layer on the head of GPT-2. For a fair comparison, we followed the default parameter settings of the baselines and repeated all experiments 10 times to report the average accuracy. The unpaired t-test was used to evaluate the significant difference between any two approaches as multiple comparisons (details in Appendix B) for both automated and human evaluations. We trained models with early stopping. Following Dathathri et al. (2020), we evaluated the attribute control for sentiment with an external sentiment classifier.

Parameter Sensitivity A-TIP uses two hyperparameters. λ dominates the attribute relevance of generated text and δ can control the fluency of infilling content. We analyzed the parameter sensitivity on all three validation data and selected the validation data of SST-5 as an example to determine the parameter sensitivity of A-TIP. As shown in Figs.3 (A-C), we observed how λ and δ affect the performance of A-TIP by varying λ from 0.2 to 0.6 in 0.1 intervals and δ from 0.008 to 0.012 in 0.001 intervals. The results indicated that A-TIP obtain the best performance when $\lambda \in [0.4, 0.5]$ and $\delta \in [0.010, 0.011]$. The reason why these parameters can affect the results is that when $\lambda < 0.4$, the attribute relevance becomes stronger and the fluency gets destroyed. $\lambda > 0.5$ weakens both the attribute relevance and text diversity. When $\delta < 0.01$, A-TIP tends to preserve modified words, which leads to low fluency. When $\delta > 0.012$, A-TIP preserves the original unmodified words, which causes low attribute relevance and diversity of text. To achieve a balanced performance, we set λ=0.4 and δ=0.01 on all datasets in our experiments.

Considering that the mask rate r is also a hyperparameter, we analyzed its effect on the results by varying it from 10% to 70%. We found the same trend on all datasets and took SST-5 as an example. As shown in Fig.3 (D), the fluency decreased when r varies from 10% to 40% because infilling content may be well-formed with its bidirectional context. As r increased from 40% to 70%, the fluency of text mainly depends on the baselines' original generation ability, which is stable. Fig.3 (E) shows that when r increases, the baselines cannot recover the attributes of infilling content well. However, A-TIP can generate attribute-aware text to improve the classification accuracy. All baselines can obtain stable fluency and classification accuracy when r = 50%, we fixed r= 50% to show numerical experimental results in the later experiments.

Dataset	SST-5		Abstracts		ROCStories	
Metrics	Fluency	Attri-Rele	Fluency	Attri-Rele	Fluency	Attri-Rele
TIGS	4.076	4.008	4.072	3.920	4.080	3.960
MaskGan	3.982	3.892	3.962	3.921	4.002	3.861
BERT	4.320	4.196	4.180	4.120	4.076	3.988
Roberta	4.168	4.132	4.068	3.892	4.016	4.032
BLM	4.084	3.956	3.856	3.968	4.072	3.992
ILM	4.236	4.076	4.104	3.964	4.048	3.992
A-TIP	**4.476**†	**4.320**†	**4.396**†	**4.296**†	**4.452**†	**4.348**†

Table 3: Human evaluation on three datasets. † indicates the results significantly surpass others.

5.2 Automated Evaluation

We evaluated the performance of A-TIP on attribute-aware text infilling by measuring PPL, Dist1, Dist2, Dist3, and ACC on the test data. Table 2 shows, A-TIP outperformed other baselines, indicating that our proposed framework can take advantage of the bidirectional context and attribute information. Additionally, ILM can achieve good results on PPL because it also adopts GPT-2 for text infilling. However, compared to one-layer positional encoding and auto-regression attention mechanism in ILM, A-Tip/Dis (A-Tip without discriminator) achieves better fluency (PPL) because it adopts the modifies attention mechanism (Fig.2 (A)) to effectively learn the length for each blank, and focus on the number/location of blanks by two-level positional encoding (intra- and inter-blank).

A-TIP obtained more accurate sentence attributes than other baselines, which demonstrates A-TIP can generate text that satisfies the desired attribute. While the accuracy was improved by 8% compared with the baselines, we observed ILM and BERT also yield high classification accuracy. This is because we randomly masked 50% of tokens in the original input without considering whether the token has a specific attribute. We did not generally mask attribute relevant tokens, that helps the sentence maintain its original attribute. If all attribute relevant tokens are masked, we can obtain better results. For a fair comparison, we randomly masked tokens instead of masking specific tokens.

5.3 Ablation Study

To verify the effect of each component in A-TIP, we conducted an ablation study. In specific, A-TIP/Dis does not include the plug-and-play discriminator, and the text infilling part remains unchanged. A-TIP/KL does not include the KL loss and threshold-based strategy. Table 2 shows A-TIP/Dis can improve text fluency while reducing attribute relevance. A-TIP/KL increases attribute relevance and decreases text fluency. Since the discriminator can guide generation towards the attribute-aware direction, while losing the fluency to a certain extent. By incorporating KL and a threshold, A-TIP achieves a better balanced performance.

5.4 Human Evaluation

We considered two types of human annotation: fluency and attribute relevance (Attri-Rele). Annotators were asked to evaluate the fluency/attribute relevance of each individual sample on a scale of 1∼5, with 1 being Not fluent/Not relevant at all and 5 being Very fluent/Very relevant, as in (Lample et al., 2019). We randomly selected 100 samples for each baseline from each test data and asked ten people on Amazon Mechanical Turk to identify the fluency and attribute relevance for each sample. We then used the average scores of ten annotations as final scores (see more detail in Appendix C).

As shown in Table 3, A-TIP achieved the highest score compared with the baselines, indicating that sentences infilled by A-TIP can be not only more fluent but also more attribute relevant. Somewhat surprisingly, we observed that BERT, TIGS, and MaskGan yield the worst performance. BERT performed poorly due to the intrinsic difficulty of finding convincing infilling content with a suitable length. TIGS and MaskGan may have performed poorly because, unlike ILM and A-TIP, they were not initialized from a large-scale pre-trained LM.

5.5 Running Time Comparison

To generate attribute-aware tokens, we update the Eq.(10) less than 10 times for each token. As shown in Fig.5, we compare the running time be-

Sentiment content infilling: Watching these [Mask] is both [Mask] and [Mask] .
Roberta: Watching these **kids** is both **funny** and **heartbreaking too**.
BLK: Watching these **teams** is both **inspiring** and **the action**.
A-TIP: Watching these **performances** is both **inspiring** and **artfully mesmerizing**. *(Positive)*
A-TIP: Watching these **shows** is both **boring** and **disgusting me much**. *(Negative)*

Expert knowledge infilling: [Mask] of [Mask] and [Mask] of their [Mask] as [Mask] using the [Mask] .
Bert: **One** of **her friends** and **one** of their **friends** as **a child** using the **same name** .
TIGS: **Systems** of **and control** and **capability** of their **distance** as **a process** using the **combination problem**.
A-TIP: **Analysis** of **data sources** and **simulation** of their **behaviors** as **a program** using the **execution model**. *(CS)*
A-TIP: **Introduction** of **randomness matrices** and **decomposition** of their **method** as a **tool** using the **matrix theory**. *(Math)*

Figure 4: Case study for sentiment content infilling and expert knowledge infilling.

tween A-TIP/Dis and A-TIP to ensure that we have less additional time-consuming. Specifically, we randomly select 30 samples from SST-5 and ROCStories datasets, where SST-5 contains short sentences and ROCStories contains almost long sentences. Then, we changed the mask rate from 30% to 70% for each selected sample to make our results more reliable. As shown in Fig.5, compared with the unconditional LM-based text generation task, updating the hidden state towards attribute-relevant direction will take less additional time.

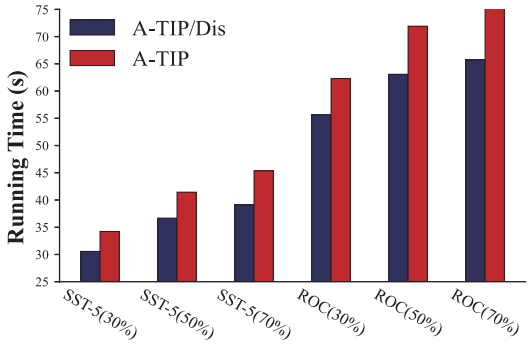

Figure 5: Running time comparison between A-TIP/Dis and A-TIP on SST-5 and ROCStories. We change the mask rate from 30% to 70%.

5.6 Case Study

We conducted a case study to show the infilling ability of A-TIP. Specifically, as shown in Fig.4, we first propose to infill the blanks with sentimental words. We choose Roberta and BLK as our compared examples. Because these two methods get the best result in this case. We can see Roberta infill the blanks with two contradictory words (funny and heartbreaking), where humans do not have such contradictory and complex emotional expressions. BLK can unify the expression of emotion, but it can not ensure the fluency of the generated sentence. In contrast, we can control A-TIP to generate positive or negative infilling contents with high fluency.

We want to explore if A-TIP can generate domain knowledge for a specific area for the second case. We choose BERT and TIGS as our compared examples. Since these two methods get the best result in domain knowledge infilling. We find that they cannot generate expert knowledge infilling content. And they tend to generate correct and high-frequency infilling content, while they are generally meaningless and monotonous (Sun et al., 2021; Lazar et al., 2021; Su et al., 2021). However, we can control A-TIP to generate both CS-related and Math-related infilling content by constraining the attribute as CS and Math.

6 Conclusion

In this paper, we presented a simple strategy for text infilling A-TIP that leverages an LM by proposing new attention mechanisms and two-level positional encoding to effectively improve the quality of generation in limited data settings. Furthermore, our plug-and-play discriminator can guide the generation towards the direction of improving text attribute relevance. In future work, we plan to incorporate the plug-and-play discriminator into more systems that assist humans in the writing process, where we hope that our work encourages more investigation of text infilling.

Acknowledgements

We would like to gratefully thank the anonymous reviewers for their helpful comments and feedback. Dongyuan Li and Jingyi You acknowledge the support from China Scholarship Council (CSC).

References

Yoav Benjamini and Yosef Hochberg. 1995. Controlling the false discovery rate: a practical and powerful approach to multiple testing. *Journal of the Royal statistical society: series B (Methodological)*, 57(1):289–300.

Federico Bianchi, Silvia Terragni, and Dirk Hovy. 2021. Pre-training is a hot topic: Contextualized document embeddings improve topic coherence. In *Proceedings of NAACL*.

Thodsaporn Chay-intr, Hidetaka Kamigaito, and Manabu Okumura. 2021. Character-based Thai word segmentation with multiple attentions. In *Proceedings of RANLP*, pages 264–273.

Sumanth Dathathri, Andrea Madotto, Janice Lan, Jane Hung, Eric Frank, Piero Molino, Jason Yosinski, and Rosanne Liu. 2020. Plug and play language models: A simple approach to controlled text generation. In *Proceedings of ICLR*.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: pre-training of deep bidirectional transformers for language understanding. In *Proceedings of NAACL*.

Chris Donahue, Mina Lee, and Percy Liang. 2020. Enabling language models to fill in the blanks. In *Proceedings of ACL*.

William Fedus, Ian J. Goodfellow, and Andrew M. Dai. 2018. Maskgan: Better text generation via filling in the _____. In *Proceedings of ICLR*.

Jiaxian Guo, Sidi Lu, Han Cai, Weinan Zhang, Yong Yu, and Jun Wang. 2018. Long text generation via adversarial training with leaked information. In *Proceedings of AAAI*.

Xingwei He. 2021. Parallel refinements for lexically constrained text generation with bart. In *Proceedings of EMNLP*.

Chris Hokamp and Qun Liu. 2017. Lexically constrained decoding for sequence generation using grid beam search. In *Proceedings of ACL*.

Nitish Shirish Keskar, Bryan McCann, Lav R. Varshney, Caiming Xiong, and Richard Socher. 2019. CTRL: A conditional transformer language model for controllable generation. In *Proceedings of ICLR*.

Diederik P. Kingma and Jimmy Ba. 2015. Adam: A method for stochastic optimization. In *Proceedings of ICLR*.

Ben Krause, Akhilesh Deepak Gotmare, Bryan McCann, Nitish Shirish Keskar, Shafiq R. Joty, Richard Socher, and Nazneen Fatema Rajani. 2021. Gedi: Generative discriminator guided sequence generation. In *Proceedings of EMNLP*.

Guillaume Lample, Sandeep Subramanian, Eric Michael Smith, Ludovic Denoyer, Marc'Aurelio Ranzato, and Y-Lan Boureau. 2019. Multiple-attribute text rewriting. In *Proceedings of ICLR*.

Koren Lazar, Benny Saret, Asaf Yehudai, Wayne Horowitz, Nathan Wasserman, and Gabriel Stanovsky. 2021. Filling the gaps in Ancient Akkadian texts: A masked language modelling approach. In *Proceedings of EMNLP*.

Jiwei Li, Michel Galley, Chris Brockett, Jianfeng Gao, and Bill Dolan. 2016. A diversity-promoting objective function for neural conversation models. In *Proceedings of NAACL*.

Zhiyu Lin and Mark Riedl. 2021. Plug-and-blend: A framework for controllable story generation with blended control codes. *arXiv preprint arXiv:2104.04039*.

Dayiheng Liu, Jie Fu, Pengfei Liu, and Jiancheng Lv. 2019. TIGS: An inference algorithm for text infilling with gradient search. In *Proceedings of ACL*.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2020. Roberta: A robustly optimized bert pretraining approach. In *Proceedings of ICLR*.

Lucas Gouveia Omena Lopes, Thales M. A. Vieira, and William Wagner M. Lira. 2021. Automatic evaluation of scientific abstracts through natural language processing. *arXiv preprint arXiv:2112.01842*.

Andrea Madotto, Etsuko Ishii, Zhaojiang Lin, Sumanth Dathathri, and Pascale Fung. 2020. Plug-and-play conversational models. In *Proceedings of EMNLP*.

Saif Mohammad. 2018. Obtaining reliable human ratings of valence, arousal, and dominance for 20,000 English words. In *Proceedings of ACL*.

Nasrin Mostafazadeh, Nathanael Chambers, Xiaodong He, Devi Parikh, Dhruv Batra, Lucy Vanderwende, Pushmeet Kohli, and James Allen. 2016. A corpus and cloze evaluation for deeper understanding of commonsense stories. In *Proceedings of NAACL*.

Jiefu Ou, Nathaniel Weir, Anton Belyy, Felix Yu, and Benjamin Van Durme. 2021. Infillmore: Frame-guided language generation with bidirectional context. In *Proceedings of *SEM*.

Bo Pang and Lillian Lee. 2005. Seeing stars: Exploiting class relationships for sentiment categorization with respect to rating scales. In *Proceedings of ACL*.

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. Bleu: a method for automatic evaluation of machine translation. In *Proceedings of ACL*.

Damian Pascual, Beni Egressy, Clara Meister, Ryan Cotterell, and Roger Wattenhofer. 2021. A plug-and-play method for controlled text generation. In *Proceedings of EMNLP*.

Tuomas Puoliväli, Satu Palva, and J. Matias Palva. 2020. Influence of multiple hypothesis testing on reproducibility in neuroimaging research: A simulation study and python-based software. *Journal of Neuroscience Methods*, 337:108654.

Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, Ilya Sutskever, et al. 2019. Language models are unsupervised multitask learners. *OpenAI blog*.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J. Liu. 2020. Exploring the limits of transfer learning with a unified text-to-text transformer. *Journal of Machine Learning Research*.

Tianxiao Shen, Victor Quach, Regina Barzilay, and Tommi S. Jaakkola. 2020. Blank language models. In *Proceedings of EMNLP*.

Kaitao Song, Xu Tan, Tao Qin, Jianfeng Lu, and Tie-Yan Liu. 2019. MASS: masked sequence to sequence pre-training for language generation. In *Proceedings of ICML*.

Yixuan Su, David Vandyke, Simon Baker, Yan Wang, and Nigel Collier. 2021. Keep the primary, rewrite the secondary: A two-stage approach for paraphrase generation. In *Proceedings of ACL-IJCNLP*.

Bin Sun, Shaoxiong Feng, Yiwei Li, Jiamou Liu, and Kan Li. 2021. Generating relevant and coherent dialogue responses using self-separated conditional variational AutoEncoders. In *Proceedings of ACL-IJCNLP*.

Wilson L Taylor. 1953. "cloze procedure": A new tool for measuring readability. *Journalism quarterly*.

Yiming Yang and Xin Liu. 1999. A re-examination of text categorization methods. In *Proceedings of SIGIR*.

Zhilin Yang, Zihang Dai, Yiming Yang, Jaime G. Carbonell, Ruslan Salakhutdinov, and Quoc V. Le. 2019. Xlnet: Generalized autoregressive pretraining for language understanding. In *Proceedings of NeurIPS*.

Jingyi You, Dongyuan Li, Hidetaka Kamigaito, Kotaro Funakoshi, and Manabu Okumura. 2022. Joint learning-based heterogeneous graph attention network for timeline summarization. In *Proceedings of NAACL*, pages 4091–4104.

Najam Zaidi, Trevor Cohn, and Gholamreza Haffari. 2020. Decoding as dynamic programming for recurrent autoregressive models. In *Proceedings of ICLR*.

Wanrong Zhu, Zhiting Hu, and Eric P. Xing. 2019. Text infilling. *arXiv preprint arXiv:1901.00158*.

A Detail Information for Datasets

As shown in Table 4, we give the number of examples, the total number of words and the detail attributes label for three widely used datasets, SST-5, ROCStories and Attributes, respectively. We selected these three datasets since we would like to check if A-TIP can infill the blanks with sentiment words, domain knowledge and topics.

We can directly use their labels to train our plug-and-play discriminator for datasets with attribute labels like SST-5 (sentiment labels) and Abstract (domain knowledge labels).

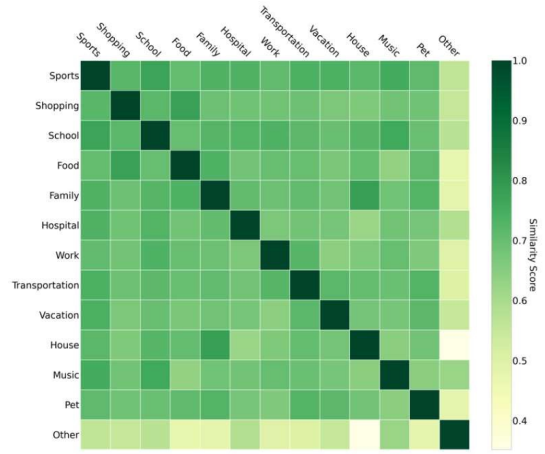

Figure 6: Topic similarity graph.

However, considering most datasets like ROCStories have no labels, we extend our method to deal with this situation. Intuitively, we can construct a general attribute-based plug-and-play discriminator to guide different datasets to generate different infilling content. However, in practical operation, it is unrealistic to build such an available attribute-based discriminator to guide the infilling generation because the downstream datasets have a variety of different attribute requirements. Therefore, we need to generate specific category labels for different downstream datasets to satisfy their specif attribute-related needs and use them to guide the infilling generation.

Specifically, we extend our model to more applications by combining our model with any topic exploration algorithms to mine topic labels on unlabeled datasets. For instance, we adopt COMBINETM (Bianchi et al., 2021) to detect topic attributes for ROCStories dataset by two methods *Contextual* and *Combined*. As shown in Table 5, we adopt three metrics to evaluate the quality of the attributes of ROCStories dataset: *Topic Coherence*, *Inverted RBO* and *NPMI*. And we choose 13 topics as our final labels since it has the best performance on average of all metrics.

As shown in Fig.6, we draw a topic similarity graph among thirteen topics. We find the similarity within topics is high, and the similarity between topics is low, demonstrating that the detected topics have high quality and low redundancy. We adopt 13 topic labels to train discriminators for ROCStories datasets, and we achieve the best performance about topic-relevant on human evaluation.

B Benjamini-Hochberg procedure

The Benjamini-Hochberg (*B-H*) Procedure is a powerful tool that decreases the false discovery rate (Benjamini and Hochberg, 1995). Considering the reproducibility of multiple significant test, we introduce how we adopt the *B-H* procedure and give the hyper-parameter values that we used.

Specifically, we first adopt t-test (Yang and Liu, 1999) with default parameters[3] to calculate p-value between each compared algorithm with A-TIP. Then, we put the individual p-values in ascending order as input to calculate p-value corrected by *B-H*. We directly use the "multipletests(*args)" function from python package[4] and set the hyper-parameter of false discover rate $Q = 0.05$ which is the widely used default value (Puoliväli et al., 2020). Finally, we get cut-off value as the output of "multipletests(*args)" function, where cut-off is a dividing line that distinguishes whether two groups of data are significant or not. Specifically, if the p-value is smaller than the cut-off value, we can get the conclusion that two groups of data are significant different.

C Detail Information for Human Evaluation

We show the human evaluation in Fig.7. We adopt fluency and attribute relevance as our evaluation metrics. We use their label as their attribute for labelled datasets SST-5 and Abstract. For unlabeled datasets like ROCStories, we manufacture labels as their attributes. And we list detailed scores from 1 to 5 for each metric.

[3] https://docs.scipy.org/doc/scipy/reference/generated/scipy.stats.ttest_ind.html

[4] https://www.statsmodels.org/dev/generated/statsmodels.stats.multitest.multipletests.html

Dataset	Examples	Words	Attributes
SST-5	11,855	215,154	Negative/ Somewhat negative/ Neutral/ Somewhat positive/ Positive
ROCStories	100K	5M	Sport/ Shop/ School/ Food/ Family/ Hospital/ Work/ Car/ Vacation/ House/ Music/ Pet/ Other
Abstracts	200K	30M	Condensed Matter/ CS/ Math/ Nonlinear Sciences/ Physics/ Bio/ Quant-Phy/ Statistics

Table 4: Descriptive statistics of datasets and their attributes.

	Contextual			Combined		
Topic Number	**Coherence**	**RBO**	**NPMI**	**Coherence**	**RBO**	**NPMI**
10	0.490	0.160	0.150	0.348	0.079	0.232
11	0.981	1.000	0.007	0.981	1.000	-0.008
12	0.986	1.000	0.030	0.933	1.000	-0.004
13	**0.993**	**1.000**	**0.053**	0.972	1.000	0.061
14	0.951	1.000	0.048	0.971	1.000	0.060
15	0.936	1.000	0.042	0.946	1.000	0.059
16	0.935	1.000	0.044	0.921	1.000	-0.008
17	0.905	0.998	0.042	0.922	0.992	0.037
18	0.906	0.982	0.045	0.868	0.989	0.038
19	0.892	0.977	0.043	0.822	0.982	0.021
20	0.882	0.972	0.040	0.802	0.978	0.022

Table 5: Contextual-based and Combined-based topic detection algorithms evaluate three widely used metrics: Topic Coherence, Inverted RBO, and NPMI.

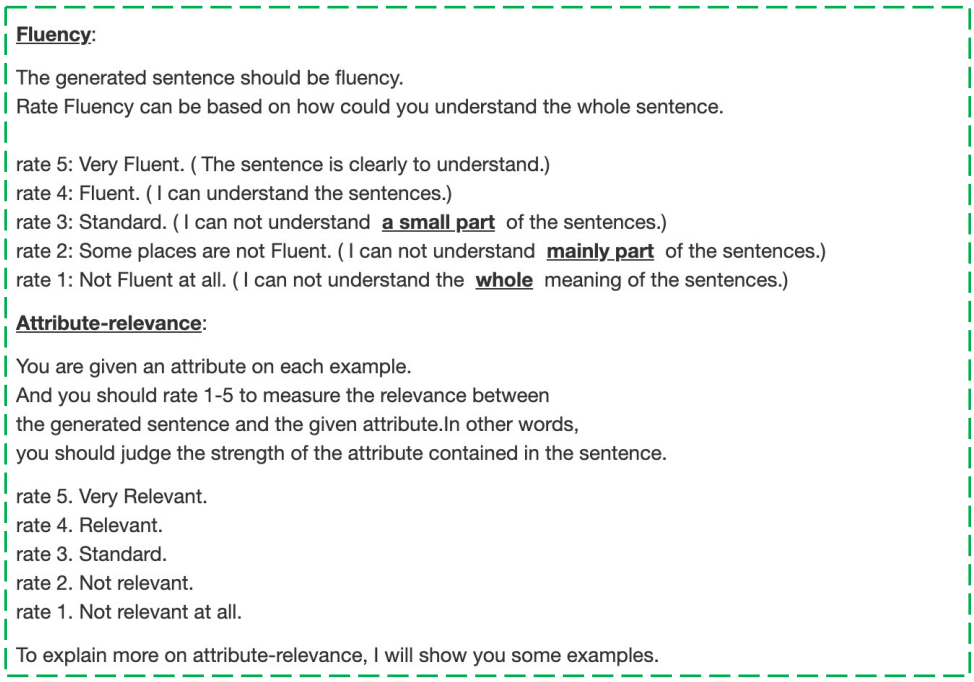

Figure 7: Human evaluations on Amazonmturk.

Multi Graph Neural Network for Extractive Long Document Summarization

Xuan-Dung Doan[1] and **Le-Minh Nguyen**[2] and **Khac-Hoai Nam Bui**[1]*
[1]Viettel Cyperspace Center, Viettel Group, Vietnam
[2]Japan Advanced Institute of Science and Technology, Japan
dungdx4@viettel.com.vn,nguyenml@jaist.ac.jp,nambkh@viettel.com.vn

Abstract

Heterogeneous Graph Neural Networks (HeterGNN) have been recently introduced as an emergent approach for extracting document summarization (EDS) by exploiting the cross-relations between words and sentences. However, applying HeterGNN for long documents is still an open research issue. One of the main majors is the lacking of inter-sentence connections. In this regard, this paper exploits how to apply HeterGNN for long documents by building a graph on sentence-level nodes (homogeneous graph) and combine with HeterGNN for capturing the semantic information in terms of both inter and intra-sentence connections. Experiments on two benchmark datasets of long documents such as PubMed and ArXiv show that our method is able to achieve state-of-the-art results in this research field.

1 Introduction

Extractive Document summarization aims to automatically extract a set of sentences, which represents information for the whole document, by ranking the importance of sentence features. Recent works focus on GNN, a Deep learning-based approach that operates on graph domain (Zhou et al., 2020), to achieve remarkable results in this research field. Specifically, GNN-based models are able to encode the complicated pairwise relationships between entity tokens for better informative representations (Wu et al., 2021). Cui et al. (2020) uses information of topic-aware to change the representation of words to a new representation. Then, a GNN-based model is presented for capturing relationships efficiently via graph-structured document representation between sentences. Sequentially, HeterGNN, a special kind of GNN (Zhang et al., 2019), has been proposed as a promising approach to enrich the relationships between words and sentences. Wang et al. (2020) introduced HSG, a heterogeneous graph-based neural network for extractive summarization by using more fine-grained semantic units in the summarization graph to extract the complex relationships between words and sentences. Accordingly, the model has achieved the top performance in CNN/DailyMail and NYT50 datasets in terms of the non-BERT-based approach. In order to utilize the capability of BERT-based models (Devlin et al., 2019), Jia et al. (2020) proposed a hierarchical attentive heterogeneous graph (HAHSum) to improve the redundant phrases problem between extracted sentences of the summarization, which has achieved promising results on news article datasets such as CNN/DailyMail, NYT, and Newsroom. Nevertheless, the model requires external analysis for modeling long-range dependencies. Intuitively, transformer-based language models are not able to process long pieces of text. Several works have provided promising results (Cui and Hu, 2021), however, the input length limitation and encoding of long texts are still open challenges in this research field (Zhong et al., 2020).

In this study, we take an investigation on improving the performance of the EDS problem for long documents in which the core idea is to exploit the complex relationship in terms of both inter and intra-sentence connections using graph-based methods. Specifically, HeterGNN-based models are able to enrich the cross-sentence relations by adding a word node as the intermediary to connect sentences. However, the inter-sentence connections are not considered. Specifically, only sentences with common words can have a connection, which might influence the performance, especially in terms of long-form document representation. Therefore, we present a novel method for enriching the inter-sentences relations by proposing a homogeneous graph neural network (HomoGNN) and incorporating the HeterGNN for final sentence representations. In particular, inspired by recent state-of-the-art models for long-form document represen-

*Corresponding author

Proceedings of the 29th International Conference on Computational Linguistics, pages 5870–5875
October 12–17, 2022.

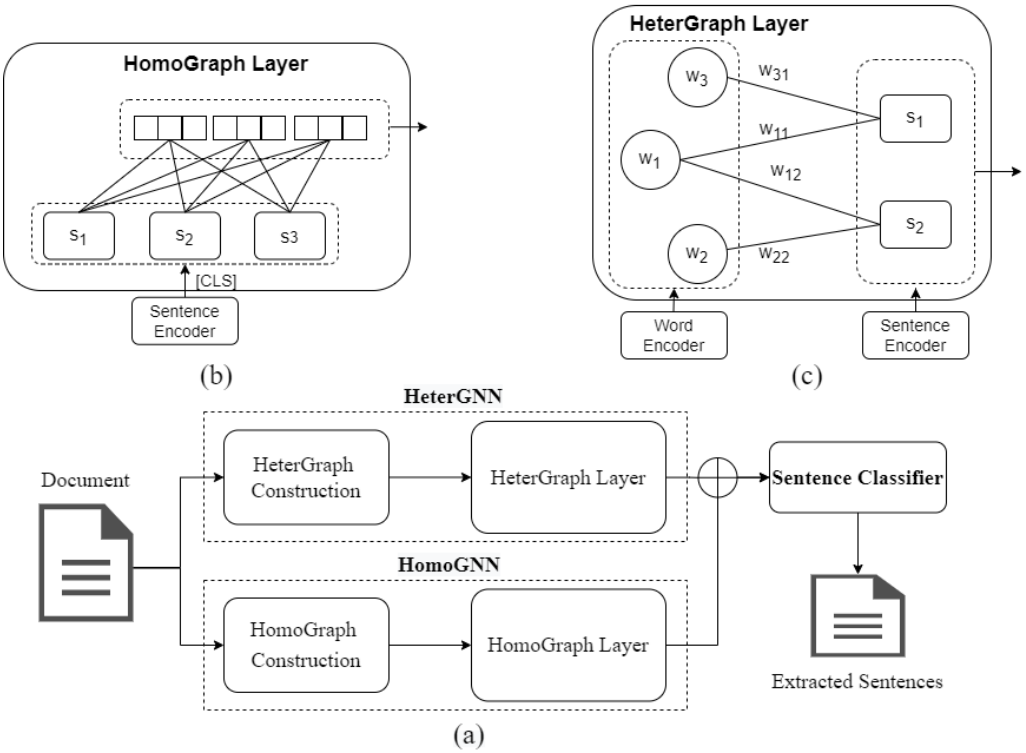

Figure 1: Overview pipeline of the proposed model which is executed simultaneously in two phases (a). The first phase encodes the sentences with pre-trained BERT and uses [CLS] information as the input of a graph attention layer (b). The second phase encodes the word and sentence nodes as the inputs of a heterogeneous graph layer (c). The output of the two phases is concatenated and put into an MLP layer in order to classify labels for each sentence.

tations such as Longformer (Beltagy et al., 2020), Big-Bird (Zaheer et al., 2020), and Poolingformer (Zhang et al., 2021), we use the information at the beginning of the sentence [CLS] representation for the inputs of the graph attention layer. Sequentially, the combination of HomoGNN and HeterGNN is able to capture the semantic information for both inter and intra-sentence connections. Figure 1 illustrates the overview of the proposed model. To the best of our knowledge, our method is the first study to incorporate two types of graph structures for the EDS task. Our source code is available for further investigation on Github[1].

2 Methodology

Given an arbitrary document $D = \{s_1, .., s_n\}$ consisting n sentences, the objective of EDS problem is to predict a sequence of a set of binary label $\{y_1, .., y_n\}$. Specifically, $y_j \in [0, 1]$ represents the jth sentence, which should be included in the summary. Our proposed model for the EDS problem includes two learning layers, which execute simultaneously, such as the homogeneous graph layer and the heterogeneous graph layer.

2.1 Homogeneous Graph Neural Network

Graph Construction: Let $G_1 = \{V_1, E_1\}$ denotes an arbitrary graph, where V_1 and E_1 represent the set of node and edge, respectively. Consequentially, the homogeneous graph for an input document can be defined as a set of node $V_1 = s_1, ..., s_n$ where n is the number of sentence in the document. For initialized encoder process, BERT (Devlin et al., 2019) is used to generate the local hidden representations between sentences. Specifically, we adopt the concept of BERTSUM(Liu and Lapata, 2019) with multiple CLS for sentence representation. Sequentially, *CLS* and *SEP* tokens are inserted at the beginning and end of each sentence, respectively. Then, all tokens are fed into BERT to learn the hidden state, which can be denoted as follows:

$$h_{1,0}, h_{1,1}, ..., h_{n,0}, ..., h_{n,*} = BERT(w_{1,0}, w_{1,1}, ..., w_{n,0}, ..., w_{n,*}) \quad (1)$$

where $w_{i,j}$ is the vector embedding of the sentence i^{th} and word j^{th}. $w_{i,0}$ and $w_{i,*}$ represents the *CLS* and *SEP* tokens of the ith sentence, respectively.

[1] https://github.com/dungdx34/MTGNN-SUM

$h_{i,j}$ stands for the hidden state of the corresponding token. After BERT encoding, we select the hidden state of *CLS* to represent sentence contextual representation, which is formulated as follows:

$$H_B = h_{1,0}, ..., h_{N,0} \tag{2}$$

Sequentially, the initialized embedding is put into a GAT model for enriching the sentence connections.

Graph Propagation: Regarding the message passing process, we adopt GAT model (Velickovic et al., 2018) to learn the hidden representation of each node by aggregating the information from its neighbors. Specifically, the updated node with GAT can be calculated as follows:

$$z_{ij} = LeakyReLU(W_a[W_q h_i; W_e h_j]) \tag{3}$$

where h_i represents the node representation of the ith sentence. W_a, W_q, W_e, and W_v are trainable weights. Subsequently, the attention score between two sentence nodes is formulated as follows:

$$\begin{aligned} \alpha_{ij} = softmax(z_{ij}) = \frac{exp(z_{ij})}{\sum_{l \in N_i} exp(z_{il})} \\ \mu_i = \sigma(\sum_{j \in N_i} \alpha_{ij} W_v h_j) \end{aligned} \tag{4}$$

where σ denotes an activation function, and N_i stand for neighbor nodes. Consequentially, the output with multi-head attention can be calculated as follows:

$$h'_i = ||_{k=1}^{K} \sigma(\sum_{j \in N_i} \alpha_{ij}^k W_v^k h_j) \tag{5}$$

where $||*$ represents multi-heads concatenation. Furthermore, a residual connection is adopted to avoid gradient vanishing after iterations. Consequentially, the final output can be updated as follows:

$$H_s^{G_1} = h'_i + h_i \tag{6}$$

Generally, we use GAT for H_B to learn the relationship between sentences in a document. The output includes the representation of sentences, which is concatenated with the output of the heterogeneous graph layer for the final representation of the sentences.

2.2 Heterogeneous Graph Neural Network

Graph Construction: Let $G_2 = \{V_2, E_2\}$ denotes an undirected graph for representing the input document. The heterogeneous graph for an input document can be defined as $V_2 = V_w \cup V_s$ and $E_2 = \{e_{11}, ..., e_{mn}\}$, where $V_w = \{w_1, ..., w_m\}$ and $V_s = \{s_1, ..., s_n\}$ represents m unique words and n sentences of a document, respectively. e_{ij} denotes the edge between the *i*-th word and *j*-th sentence. Following the concept of HeterSumGraph (Wang et al., 2020), sentence node features are calculated by combining CNN for extracting the local n-gram feature of each sentence and bidirectional Long Short-Term Memory (BiLSTM) for extracting the sentence-level feature.

Graph Propagation: The heterogeneous graph layer is also updated using GAT, which is defined from Equation 3 to Equation 6. However, the vanilla GAT has been designed for homogeneous graphs. Therefore, Wang et al. (2020) has presented a modified GAT and an iterative updating mechanism for heterogeneous graph updated layer. Specifically, the Equation 3 can be re-formulated as follows:

$$z_{ij} = LeakyReLU(W_a[W_q h_i; W_e h_j; \overline{e}_{ij}]) \tag{7}$$

where $\overline{e}_{ij}$ denotes the multi-dimensional embedding space ($\overline{e}_{ij} \in \mathbb{R}^{mn \times d_e}$), which is mapped from edge weight e_{ij}. Sequentially, an iterative updating mechanism is adopted to obtain a new word node and sentence node. In particular, in order to pass messages between word and sentence nodes, the sentences with their neighbor word nodes are updated via modified-GAT and Position-Wise Feed-Forward (FFN) layer, which can be formulated as follows:

$$\begin{aligned} U_{s \leftarrow w}^1 = GAT(H_s^0, H_w^0, H_w^0) \\ H_s^1 = FFN(U_{s \leftarrow w}^1 + H_s^0) \end{aligned} \tag{8}$$

where H_w^0 (H_w^1) and H_s^0 are the node features of word X_w ($X_w \in \mathbb{R}^{m \times d_w}$) and sentence X_s ($X_s \in \mathbb{R}^{n \times d_s}$), respectively. Note that H_s^0 is used as the attention query and H_w^0 is regarded as key and value. Sequentially, the new representations of word nodes can be obtained using the updated sentence nodes and further updated sentence or query nodes, iteratively. Specifically, each iteration contains a sentence-to-word and a word-to-sentence update process, which is formulated as follows:

$$\begin{aligned} U_{w \leftarrow s}^{t+1} = GAT(H_w^t, H_s^t, H_s^t) \\ H_w^{t+1} = FFN(U_{w \leftarrow s}^{t+1} + H_w^t) \\ U_{s \leftarrow w}^{t+1} = GAT(H_s^t, H_w^{t+1}, H_w^{t+1}) \\ H_s^{t+1} = FFN(U_{s \leftarrow w}^{t+1} + H_s^t) \end{aligned} \tag{9}$$

Model	PubMed			arXiv		
	R-1	R-2	R-L	R-1	R-2	R-L
Oracle (Xiao and Carenini, 2020)	55.05	27.48	49.11	53.89	23.07	46.54
SummaRuNNer^{+}	43.89	18.78	30.36	42.91	16.65	28.53
Seq2seq-attentive^{+}	44.81	19.74	31.48	43.58	17.37	29.30
Cheng&Lapata (2016)$^{+}$	43.89	18.53	30.17	42.24	15.97	27.88
Discourse-aware*	38.93	15.37	35.21	35.80	11.05	31.80
ExtSum-LG (Xiao and Carenini, 2020)	45.39	20.37	40.99	44.01	17.79	39.09
Match-Sum (Zhong et al., 2020)	41.21	19.41	36.75	40.59	12.98	32.64
Topic-GraphSum (Cui and Hu, 2021)	45.95	20.81	33.97	44.03	18.52	32.41
SSN-DM (Cui and Hu, 2021)	46.73	21.00	34.10	45.03	**19.03**	32.58
MTGNN-SUM	**48.42**	**22.26**	**43.66**	**46.39**	18.58	**40.50**

Table 1: Results on PubMed and arXiv datasets. Report results with * are from Cohan et al. (2018), and results with + are from Xiao and Carenini (2019). Other results are obtained from respective papers. Oracle indicates the ground truth results by using the greedy algorithm, which is regarded as the upper bound. Our results are reported by averaging values of 3 runs.

2.3 Multi Graph Neural Network for EDS

The outputs of sentence features from the two aforementioned layers are then concatenated for the final representation, which is formulated as follows:

$$H = H_s^{G_1} \oplus H_s^{G_2} \tag{10}$$

Observably, by concatenating the outputs of the two aforementioned graph layers, the final representation includes the information of both inter and intra-sentence relations. Sequentially, the output of the concatenation is put into a sentence classifier for ranking the classification.

3 Experiments

3.1 Datasets

Two long document datasets are taken into account for the evaluation. Specifically, PubMed and arXiv are standard datasets for long documents, which are scientific papers. For the data processing, we use the same split as the work in Cohan et al. (2018) to process the arXiv and PubMed datasets for the evaluation and follow Liu and Lapata (2019) to get ground-truth labels.

3.2 Hyperparameter Setting

Regarding the encoding, the vocabulary is limited to 50,000 and the tokens are initialized with 300-dimensional with Glove embedding. The dimension of sentence node and edge features are set to 128 and 50, respectively. The number of multi-head in each GAT layer is set to 8. For the document encoder, we use the *bert-base-uncased* version of BERT and fine-tune for the experiments. In the case of the decoding process, we select top-6 and top-5 for PubMed and arXiv datasets, respectively, according to the best performance of the validation set. Due to the limited computational resources, the maximum number of sentences in each document is set to 150, which means only the first 150 sentences in each document are taken into account. More analysis of the length of sentences is presented in the following section. The model is trained with the Adam optimizer. The learning rate is set to 1e-3 and use early stop with every three epochs. Moreover, learning rate decay is used after each epoch to improve performance. All models are trained on a single Tesla V100 32GB GPU, which has completed the training process with around 10 epochs. The total time for each epoch with the best model is around 6 hours and 3 hours for PubMed and arXiv datasets, respectively.

3.3 Experimental Results

Table 1 shows the results of our method compared with state-of-the-art models on PubMed and arXiv, respectively. The comparison models are divided into different parts. The first part reports the results of Oracle, which is regarded as ground truth extracted sentences. The second part shows the results of the approach without pre-trained language models. The third approach includes BERT-based models. The next section presents the result of the graph-based approach including the models with the document-level approach, which requires different levels of information such as words, sentences,

latent topics, and spotlights redundancy dependencies between sentences. The last section is our model, which is named MTGNN-SUM.

Accordingly, our results outperform state-of-the-art models in this research field. In particular, only R-2 of SSN-DM, the lasted state-of-the-art model is slightly better than our method in the case of arXiv datasets. However, the R-L metric of our method is much higher than the SSN-DM model. The results indicate the advantage of the proposed method by integrating both inter and intra-sentence relations. Specifically, inter-sentence allows information to flow for all sentence nodes and intra-sentence enriches the information of sentence nodes that contain common words. This issue especially is able to deal with the long-range dependency problem because the sentences, which are far from each other (e.g., by the distance of sentence positions), are able to share the information by using common words. Notably, our model does not need to consider any external semantic nodes for enriching global information (e.g., latent topic).

4 Quality Analysis

Ablation Study. In our model, we enrich the complex relationships by exploring both heterogeneous graph and homogeneous graph operations for sentence connection. In order to explore the effectiveness of each component, we design different variants of the proposed model as follows:

- **HomoGraph-SUM**: contains a graph attention layer for document encoding to extract inter-sentence relationships. The model is constructed following the description in Section 2.1 of Homogeneous Graph Neural Network.
- **HeterGraph-SUM**: contains a heterogeneous graph layer that contains semantic nodes to enrich the cross-sentence relations. Specifically, HeterGraph-Sum is designed following the description in Section 2.2.

The results of those aforementioned variants of our model on benchmark datasets are presented in Tab. 2. Accordingly, MTGNN-SUM outperforms all variants, which proves that executing message passing across sentences in the proposed model by incorporating both graph structures can achieve better results.

Length of Document. In this study, we set the maximum number of sentences in each document to equal 150 due to our limited computational resources. Though, we are able to improve the performance by learning whole-length sentences of the datasets, which include many documents with more than 200 sentences. In order to evaluate the importance of the document length value, we tested our model with the maximum number of sentences being 50 and 100 sentences, respectively. The results of the test models on different values of maximum document sizes are shown in Table 3. Accordingly, by increasing the maximum length of sentences, the performances are improved. In particular, the results indicated that tuning the max length of sentence value is able to enhance the performance. Specifically, we take this issue into account for the future work of this study by executing our model with a longer maximum size of documents.

Dataset	Model	R-1	R-2	R-L
PubMed	HomoGraph-SUM	39.29	13.74	34.49
	HeterGraph-SUM	46.03	19.79	41.48
	MTGNN-SUM	48.42	22.26	43.66
arXiv	HomoGraph-SUM	41.13	13.11	35.84
	HeterGraph-SUM	45.06	16.97	39.38
	MTGNN-SUM	46.39	18.58	40.50

Table 2: Ablation study results on two datasets.

Dataset	Model	R-1	R-2	R-L
PubMed	MTGNN-SUM-50	46.20	20.04	41.58
	MTGNN-SUM-100	47.85	21.64	43.13
	MTGNN-SUM-150	48.42	22.26	43.66
arXiv	MTGNN-SUM-50	44.91	16.89	39.14
	MTGNN-SUM-100	46.09	17.98	40.29
	MTGNN-SUM-150	46.39	18.58	40.50

Table 3: Results of the proposed model with different lengths of sentences.

5 Conclusion

This paper presents a novel graph-based method for the EDS task, which focuses on exploiting the complex relationship in terms of both inter and intra-sentence relations of the long-form documents. Specifically, two types of graph structures are developed for enriching sentence representations. The experiments on two benchmark datasets show promising results of the proposed method. Regarding future work, segmentation methods are taken into account for dividing long documents into paragraphs. Specifically, analyzing the complex relationships between paragraphs and integrating them into graphs as an additional node is able to enrich the information for the representation.

References

Iz Beltagy, Matthew E. Peters, and Arman Cohan. 2020. Longformer: The long-document transformer. *CoRR*, abs/2004.05150.

Arman Cohan, Franck Dernoncourt, Doo Soon Kim, Trung Bui, Seokhwan Kim, Walter Chang, and Nazli Goharian. 2018. A discourse-aware attention model for abstractive summarization of long documents. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 2 (Short Papers)*, pages 615–621, New Orleans, Louisiana. Association for Computational Linguistics.

Peng Cui and Le Hu. 2021. Sliding selector network with dynamic memory for extractive summarization of long documents. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT 2021, Online, June 6-11, 2021*, pages 5881–5891. Association for Computational Linguistics.

Peng Cui, Le Hu, and Yuanchao Liu. 2020. Enhancing extractive text summarization with topic-aware graph neural networks. In *Proceedings of the 28th International Conference on Computational Linguistics, COLING 2020, Barcelona, Spain (Online), December 8-13, 2020*, pages 5360–5371. International Committee on Computational Linguistics.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT 2019, Minneapolis, MN, USA, June 2-7, 2019, Volume 1 (Long and Short Papers)*, pages 4171–4186. Association for Computational Linguistics.

Ruipeng Jia, Yanan Cao, Hengzhu Tang, Fang Fang, Cong Cao, and Shi Wang. 2020. Neural extractive summarization with hierarchical attentive heterogeneous graph network. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 3622–3631, Online. Association for Computational Linguistics.

Yang Liu and Mirella Lapata. 2019. Text summarization with pretrained encoders. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 3730–3740, Hong Kong, China. Association for Computational Linguistics.

Petar Velickovic, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Liò, and Yoshua Bengio. 2018. Graph attention networks. In *6th International Conference on Learning Representations, ICLR 2018, Vancouver, BC, Canada, April 30 - May 3, 2018, Conference Track Proceedings*. OpenReview.net.

Danqing Wang, Pengfei Liu, Yining Zheng, Xipeng Qiu, and Xuanjing Huang. 2020. Heterogeneous graph neural networks for extractive document summarization. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 6209–6219, Online. Association for Computational Linguistics.

Lingfei Wu, Yu Chen, Kai Shen, Xiaojie Guo, Hanning Gao, Shucheng Li, Jian Pei, and Bo Long. 2021. Graph neural networks for natural language processing: A survey. *CoRR*, abs/2106.06090.

Wen Xiao and Giuseppe Carenini. 2019. Extractive summarization of long documents by combining global and local context. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 3011–3021, Hong Kong, China. Association for Computational Linguistics.

Wen Xiao and Giuseppe Carenini. 2020. Systematically exploring redundancy reduction in summarizing long documents. In *Proceedings of the 1st Conference of the Asia-Pacific Chapter of the Association for Computational Linguistics and the 10th International Joint Conference on Natural Language Processing*, pages 516–528, Suzhou, China. Association for Computational Linguistics.

Manzil Zaheer, Guru Guruganesh, Kumar Avinava Dubey, Joshua Ainslie, Chris Alberti, Santiago Ontañón, Philip Pham, Anirudh Ravula, Qifan Wang, Li Yang, and Amr Ahmed. 2020. Big bird: Transformers for longer sequences. In *Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020, December 6-12, 2020, virtual*.

Chuxu Zhang, Dongjin Song, Chao Huang, Ananthram Swami, and Nitesh V. Chawla. 2019. Heterogeneous graph neural network. In *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining, KDD 2019, Anchorage, AK, USA, August 4-8, 2019*, pages 793–803. ACM.

Hang Zhang, Yeyun Gong, Yelong Shen, Weisheng Li, Jiancheng Lv, Nan Duan, and Weizhu Chen. 2021. Poolingformer: Long document modeling with pooling attention. In *Proceedings of the 38th International Conference on Machine Learning, ICML 2021, 18-24 July 2021, Virtual Event*, volume 139 of *Proceedings of Machine Learning Research*, pages 12437–12446. PMLR.

Ming Zhong, Pengfei Liu, Yiran Chen, Danqing Wang, Xipeng Qiu, and Xuanjing Huang. 2020. Extractive summarization as text matching. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 6197–6208, Online. Association for Computational Linguistics.

Jie Zhou, Ganqu Cui, Shengding Hu, Zhengyan Zhang, Cheng Yang, Zhiyuan Liu, Lifeng Wang, Changcheng Li, and Maosong Sun. 2020. Graph neural networks: A review of methods and applications. *AI Open*, 1:57–81.

Improving Zero-Shot Multilingual Text Generation via Iterative Distillation

†Ernie Chang, *Alex Marin, †Vera Demberg
†Dept. of Language Science and Technology, Saarland University
*Microsoft Corporation, Redmond, WA
cychang@coli.uni-saarland.de

Abstract

The demand for multilingual dialogue systems often requires a costly labeling process, where human translators derive utterances in low resource languages from resource rich language annotation. To this end, we explore leveraging the inductive biases for target languages learned by numerous pretrained *teacher* models by transferring them to *student* models via sequence-level knowledge distillation. By assuming no target language text, both the teacher and student models need to learn from the target distribution in a few/zero-shot manner. On the MultiATIS++ benchmark, we explore the effectiveness of our proposed technique to derive the multilingual text for 6 languages, using only the monolingual English data and the pretrained models. We show that training on the synthetic multilingual generation outputs yields close performance to training on human annotations in both slot F1 and intent accuracy; the synthetic text also scores high in *naturalness* and *correctness* based on human evaluation.

1 Introduction

In multilingual dialogue systems, natural language generation is used to generate utterances in various languages, using as input semantic frames, which contain a representation of the user intent together with relevant information or entities related to said intent (Tur et al., 2010). Past works that generalize dialogue systems to multilingual settings often made two unrealistic assumptions about the data availability of any new dialogue domain (Liu et al., 2019; Xu et al., 2020; Schuster et al., 2019; Chang et al., 2020, 2022): (1) First assumption is that a large set of monolingual data has already been annotated. (2) Some in-domain text or annotated data in the target languages are already available for the purpose of transfer learning. Neither assumption holds in all cases (Upadhyay et al., 2018).

To overcome these challenges, we utilize knowledge-grounded pre-training (KGPT) (Chen et al., 2020) – a pretrained data-to-text generation model that was shown to be effective in overcoming data scarcity, as the model is capable of representing the inductive biases required to encode structured data such as the slot-value pairs (frames). In few-shot settings, we can exploit KGPT's pretrained knowledge to obtain in-domain text labels for a large set of unlabeled frame sequences. These text labels are then converted to multiple languages with the use of bilingual translation models as teacher models, as inspired by past works on sequence-level knowledge distillation (Wang et al., 2020; Kim and Rush, 2016; Gordon and Duh, 2019). In this way, we perform zero-shot cross-lingual transfer for all 6 languages. We use mBART (Liu et al., 2020) as the multilingual student model to acquire stronger bilingual knowledge from the translation teacher models from Tiedemann and Thottingal (2020).

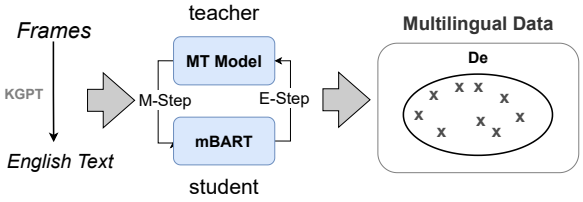

Figure 1: **Annotation scenario**: Each × represents a labeled data instance. The goal is to generalize from few-shot human-labeled instances in one language (left) to large synthetic multilingual data (right).

We leverage a two-step distillation process where we first derive a large synthetic English dialogue data from the English seed data, then generalize it to multilingual data by using the bilingual translation models to produce synthetic text labels. Finally, we perform rounds of iterative knowledge distillation following the process of the expectation-maximization algorithm for further improvements. This work makes the following contributions:

- We introduce a simple and effective technique in constructing a synthetic multilingual dia-

Proceedings of the 29th International Conference on Computational Linguistics, pages 5876–5881
October 12–17, 2022.

logue corpus using the *iterative knowledge distillation*.

- We demonstrate the efficacy of the technique by showing that its outputs display high *naturalness* and *correctness*.

2 Approach Summary

In our setting, we have (1) a seed English dataset S which consists of k labeled pairs, and (2) the full set of unlabeled frame sequences U where $|U| \gg k > 0$. The goal is to create labeled samples in all target languages consisting of the frame sequences ($\mathcal{X}$) with their corresponding texts ($\mathcal{Y}$).

Monolingual Text Generation. We first obtain the full synthetic English dataset $(\mathcal{X}_{En}, \mathcal{Y}_{En})$ from the k labeled pairs and the unlabeled semantic frame sequences. This is achieved by finetuning KGPT on the k samples and then labeling each semantic frame sequence with a corresponding English utterance.

Multilingual Text Generation. To create multilingual data, we perform the *iterative knowledge distillation* (see §3) to derive target language utterance from the source English utterance. Specifically, we update both the bilingual translation model (*teacher*) and mBART (*student*) iteratively following the expectation-maximization algorithm via likelihood maximization over parallel data $(\mathcal{X}, \mathcal{Y})$ and parameters ϕ and θ of the teacher and student models.

3 The Iterative Distillation Procedure

The iterative distillation procedure alternatively optimizes the student and teacher models until convergence. We generate the parallel synthetic data from pretrained bilingual models $p^{teacher}(y|x;\phi^1)$[1], and use the EM algorithm (Dempster et al., 1977) to optimize the process (see Figure 2). The intuition is that while the student model learns from the teacher, the teacher model also needs to discard some out-of-domain knowledge by learning from the student feedback. The iterative procedure alternates between the teacher model learning some knowledge from the target distribution, and then the student model is updated based on the new teacher model. In this way, both models are improved in training. As such, we use the following high-level strategy.

[1]Note that ϕ^1 is used to indicate the initial pretrained model at iteration 1.

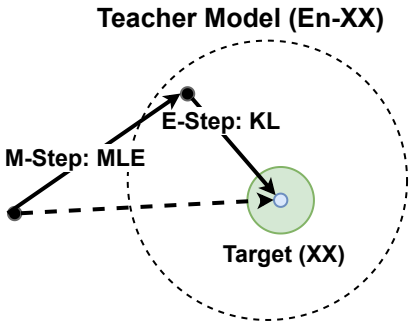

Figure 2: **Iterative Knowledge Distillation**: Each circle represents a labeled data instance from the *student* where **M-step** moves the output distribution towards that of the teacher model's; and **E-step** measures the distributional difference and makes further adjustment towards the *target language domain*.

In the E-step (multilingual labeling), we fix the $p^{student}$ and update the posterior distribution:

$$q^{t+1} = \text{argmin}\mathbb{KL}(\hat{\mathcal{Y}} \| p^{student}(\mathcal{Y}|\mathcal{X};\theta^t)),$$

Conversely, in the M-step (training), we fix $q(\mathcal{Y})$ and update θ to maximize the expected log-likelihood:

$$\theta^{t+1} = \text{argmax}_\theta \mathbb{E}_{q^{t+1}}[\log p^{student}(\mathcal{Y}|\mathcal{X};\theta)],$$

In what follows we introduce the details of the E-step and the M-step in our framework.

Expectation Step. The E-step aims to compute the posterior distribution $q(\mathcal{Y})$ that minimizes the KL divergence between $q(\mathcal{Y})$ and $p^{student}(\mathcal{Y}|\mathcal{X})$. This step basically brings the teacher model closer to the target distribution without having seen the distribution itself. Importantly, we also estimate the gradient of $\mathcal{L}(\cdot)$ w.r.t. teacher model's parameter ϕ by applying the REINFORCE algorithm (Williams, 1992) to compute the loss, which is then weighted by $\log \frac{p^{student}(y|x;\theta^t)}{p^{teacher}(y|x;\phi)}$ under the KL divergence equation. Overall, this constructs a weighted synthetic training dataset that intuitively *adjusts the outputs to be more in-domain*, as the original teacher model is general-domain.

Maximization Step. In the M-step, we update the student model to be closer to the teacher model. To do so, we optimize the parameters θ^{t+1} with the parameterized posterior distribution $p^{teacher}(\mathcal{Y}|\mathcal{X};\phi^{t+1})$ so as to optimize the *student model* to generate target language text. We apply sequence-level knowledge distillation (Kim and Rush, 2016) and use the targets with maximum likelihood in the teacher model to train the student model.

Semantic frame sequence:
[B-fromloc.city_name] toronto, [B-toloc.city_name] newark, [B-round_trip] one, [I-round_trip] way, [B-depart_date.day_name] wednesday, [B-depart_time.period_of_day] evening, [B-depart_date.day_name] thursday, [B-depart_time.period_of_day] morning
English reference: i need a flight from toronto to newark one way leaving wednesday evening or thursday morning

English (En): I need a flight from Toronto to Newark, one way is to leave from Wednesday night or Thursday morning
German (De): Ich brauche einen Flug von Toronto nach Newark in eine Richtung ab Mittwochabend oder Donnerstagmorgen
Spanish (Es): Necesito un vuelo de Toronto a Newark solo ida y salida el miércoles por la noche o el jueves por la mañana
French (Fr): Besoin d'un vol de toronto à newark aller simple partant mercredi soir ou jeudi matin
Chinese (Zh): 需要一种从多伦多到纽瓦克的航班，从星期三晚上或星期四早上离开
Japanese (Ja): 水曜日の夕方または木曜日の朝を出して、トロントからニュアクへの片道のフライトが必要です
Portuguese (Pt): preciso de um voo de toronto para newark só de ida saindo na quarta à noite ou quinta de manhã

Figure 3: Table showing the labeled examples in all seven languages. The *upper* portion shows the monolingual (English) semantic frame sequence and utterance pair. The *bottom* region displays all seven languages.

Quality-Based Text Filtering. To ensure that only target data with high semantic correctness quality is used for training, we impose a filtering operation on the generated samples for quality control. In the process of *multilingual labeling*, we assume to only have the access to monolingual (English) frame sequence, and so we rely on the likelihood score of a pre-trained teacher model (bilingual MT models) as the quality metric $\mathrm{Q}_{x^i}(y^i) = \log p^{teacher}(y^i|x^i;\phi)$. where ϕ denotes the *initial* teacher model trained on the original ground-truth dataset. In practice, we use nuclear sampling (Holtzman et al., 2019) (which has a resizable beam size) as the heuristic sampling on $p^{teacher}(y|x;\phi^t)$, and then filter out the candidates which do not satisfy the condition $\mathrm{Q}_x(y) \geq b$, where b is set to be 0.5 based on our empirical findings. In this way, we control the quality of $p^{teacher}(y|x;\phi^{t+1})$ by manipulating the quality of its training data.

4 Experimental Settings

Training Configurations For mBART training, we use the same vocabulary of subword units as Liu et al. (2020); this vocabulary includes a sentencepiece model (Kudo and Richardson, 2018) with $250,000$ subword tokens. The mBART model consists of the standard sequence-to-sequence Transformer architecture with 6 encoder and 6 decoder layers; each layer consists of a 1024-dim model on 8 heads ($\sim$ 144M parameters altogether). Our model is trained on 256 Nvidia V100 GPUs (32GB). The final models are selected based on BLEU (Papineni et al., 2002) scores on the validation set.

Testing Scenarios We evaluate our technique on the MultiATIS++ corpus (Xu et al., 2020), which consists of re-annotated ATIS dataset in six additional languages: German (De), Spanish (Es), French (Fr), Chinese (Zh), Japanese (Ja), and Portuguese (Pt). The test sets are based on the released human-labeled set consisting of 893 instances. Particularly, we report the results on both intent classification and slot filling F1 scores for NLU inference; and evaluate the surface-level overlap with BLEU-4 for text generation. The reason for this is so that we could get a sense of the correlation between text quality and its usefulness for NLU inference. For our experiments, we *assume that semantic frames corresponding to all target languages are present*. For semantic frame sequence predictions, we employ Fairseq (Ott et al., 2019) and train it on the synthetic multilingual corpus for all targeted language pairs. We adopt several ways of generating the synthetic corpus (En-XX) from the English seed data consisting of **50-shot**, **100-shot**, and **all** English ATIS data:

MT: The baseline is the direct translation of the *seed English utterances* into target language utterance (XX), then training mBERT on the synthetic data consisting of target language utterance and its semantic frames.

KGPT+MT: We use KGPT to create the full synthetic English corpus, then perform **MT**.

mBART: On top of **KGPT+MT**, we finetune mBART on the synthetic En-XX corpus, then create (En-XX) via translation.

mBART+EM: Building on top of **mBART**, we perform the proposed EM algorithm and allowing both the bilingual model (teacher) and mBART (student) to be updated.

	En		Fr		De		Zh		Es		Ja		Pt	
NLU (Slot %F-1 Intent %Acc)														
50-shot														
MT	67.15	72.34	66.61	71.66	63.66	78.54	58.41	74.42	58.83	71.53	70.55	68.83	66.72	75.51
KGPT+MT	70.19	77.28	70.99	75.37	66.73	82.88	61.56	80.65	61.92	76.83	75.22	73.18	70.20	78.22
mBART	72.43	79.32	72.68	79.66	68.27	84.78	66.29	83.21	64.19	77.52	75.61	77.43	**72.01**	**80.33**
mBART+EM (Ours)	**75.48**	**82.15**	**75.37**	**71.24**	**71.75**	**85.34**	**67.31**	**84.20**	**64.88**	**77.63**	**76.38**	**78.72**	71.28	80.29
100-shot														
MT	65.24	71.73	56.53	72.48	64.62	79.51	63.55	74.48	60.37	72.61	71.47	71.52	68.48	75.41
KGPT+MT	74.37	85.54	65.84	84.17	80.33	83.43	76.32	83.11	63.53	78.54	76.57	79.34	77.63	80.74
mBART	78.52	87.82	66.38	83.43	72.28	85.80	77.81	84.33	67.42	81.37	79.16	80.26	77.13	82.61
mBART+EM (Ours)	**82.22**	**88.14**	**67.44**	**83.82**	**74.29**	**88.32**	**77.92**	**85.32**	**68.39**	**81.98**	**79.89**	**81.12**	**77.45**	**84.50**
All														
MT	85.15	89.88	70.33	87.42	75.72	91.42	77.72	92.26	72.41	84.35	81.71	83.25	80.35	87.74
mBART	87.42	89.73	81.74	88.33	76.62	92.93	78.73	92.15	**74.42**	**84.63**	81.74	83.83	**80.62**	**87.12**
mBART+EM (Ours)	**88.97**	**90.10**	**82.35**	**90.02**	**76.93**	**93.66**	**79.01**	**92.89**	74.25	84.19	**82.55**	**84.30**	80.60	87.28
NLG (BLEU-4)														
MT	9.22		7.58		9.71		7.82		8.63		9.33		8.88	
Ours (50-shot)	10.37		8.29		10.21		7.67		8.45		9.72		8.23	
Ours (100-shot)	11.23		9.38		11.87		**9.44**		**9.90**		10.91		**9.58**	
Ours (All)	**12.67**		**10.32**		**12.43**		9.33		9.80		**10.99**		9.11	

Table 1: Benchmark comparison on all seven languages reporting both NLU (**slot(%) intent(%)**) and NLG (**BLEU-4**). KGPT (Chen et al., 2020) is the pretrained data-to-text generation model; and MT refers to the use of Helsinki bilingual translation model (Tiedemann and Thottingal, 2020).

Model	Spanish (Es)			Chinese (Zh)		
	Naturalness	Miss	Wrong	Naturalness	Miss	Wrong
Reference	4.00	-	-	5.00	-	-
MT	**3.66**	57	24	3.33	28	33
KGPT+MT	3.33	45	22	**4.33**	24	32
mBART	3.33	37	17	3.33	23	28
Ours	**3.66**	**23**	**12**	**4.33**	**18**	**21**

Table 2: Human Evaluation on the sampled outputs (100 instances) for all models on the 100-shot scenario. Three annotators were asked to evaluate the *Naturalness* (0-5), *Miss* (i.e. # missed slots), and *Wrong* (i.e. # hallucinated slots).

5 Results and Analysis

Here we first present two forms of analysis for both monolingual and multilingual data, then analyze the synthetic data with human evaluation.

Analysis of Monolingual Data. In table 1, we first observe that the use of KGPT (*KGPT+MT*) in the few-shot settings helps to produce high quality synthetic English (En) data that allows the mBART models to achieve decent NLU performance. We also see that the difference between 50-shot and 100-shot is minor, which we think is highly dependent on the random sampling process of the few-shot data. The improvement for *Ours* is slightly more drastic when all data is used (*All*), where the performance (82.22 to 91.97) approaches that of the system using the real full English data.

Analysis of Multilingual Data. In the multilingual setting, we observe that finetuning mBART on the translated data (*mBART*) brings about noticeable improvements generating high quality text over some language pairs (e.g. En-Fr, En-De); the improvement is limited for some languages (e.g. Ja). We attribute this to the cross-lingual similarity that allows some language pairs to obtain more useful inductive biases than in cases of more dissimilar language pairs. Further, we observe consistent improvements that *mBART+EM* has over the base models, suggesting that the iterative knowledge distillation process is crucial to draw both the *teacher* and *student* models to the in-domain region. This can be seen across most languages. As such, we conclude that the proposed technique does indeed help to create useful synthetic data, even in zero- or few-shot cross-lingual settings.

On Generation Quality. In Table 1, we also notice the *limited BLEU-4 scores*, which means that the multilingual human annotation has rephrased the utterance quite drastically different from the original English text. To examine further, we perform human evaluation (See Table 2) on the Spanish and Chinese generation outputs based on 100-shot data, to look for evidence of naturalness and high fidelity. We observe that the human evaluation is consistent with that of the intent classification and slot filling scores, while having high naturalness and fidelity as defined by *Miss* and *Wrong*.

6 Limitations

We also recognize that the approach gradually loses its effectiveness as the size of the data increases.

Moreover, in some language pairs such as Portuguese and Chinese, the improvement with EM steps remain largely limited. We attribute this to the linguistic gap across language pairs, which is the biggest limitation of our approach, since the approach's effectiveness hinges upon the proximity between source-side high resource language (i.e. English) and the target-side languages. Therefore we postulate that the approach would be very limited for extremely low resource languages such as many of the African languages.

7 Conclusion and Future Work

In this paper, we show the effectiveness of the proposed technique in constructing synthetic multilingual data from few-shot monolingual samples. Surprisingly, training on the synthetic outputs yields decent performance in terms of slot F1, intent accuracy, and human evaluation. We hope to extend the work in the future to low resource languages – applying it to additional tasks beyond NLU such as coreference resolution.

Acknowledgements

This research was funded in part by the German Research Foundation (DFG) as part of SFB 248 "Foundations of Perspicuous Software Systems". We sincerely thank the anonymous reviewers for their insightful comments that helped us to improve this paper.

References

Ernie Chang, David Adelani, Xiaoyu Shen, and Vera Demberg. 2020. Unsupervised pidgin text generation by pivoting english data and self-training. In *In Proceedings of Workshop at ICLR*.

Ernie Chang, Jesujoba Alabi, David Adelani, and Vera Demberg. 2022. Dialogue pidgin text adaptation via contrastive fine-tuning. In *AfricaNLP @ ICLR 2022*.

Wenhu Chen, Yu Su, Xifeng Yan, and William Yang Wang. 2020. KGPT: Knowledge-grounded pre-training for data-to-text generation. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 8635–8648, Online. Association for Computational Linguistics.

Arthur P Dempster, Nan M Laird, and Donald B Rubin. 1977. Maximum likelihood from incomplete data via the em algorithm. *Journal of the Royal Statistical Society: Series B (Methodological)*, 39(1):1–22.

Mitchell A Gordon and Kevin Duh. 2019. Explaining sequence-level knowledge distillation as data-augmentation for neural machine translation. *arXiv preprint arXiv:1912.03334*.

Ari Holtzman, Jan Buys, Li Du, Maxwell Forbes, and Yejin Choi. 2019. The curious case of neural text degeneration. In *International Conference on Learning Representations*.

Yoon Kim and Alexander M Rush. 2016. Sequence-level knowledge distillation. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 1317–1327.

Taku Kudo and John Richardson. 2018. SentencePiece: A simple and language independent subword tokenizer and detokenizer for neural text processing. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 66–71, Brussels, Belgium. Association for Computational Linguistics.

Yinhan Liu, Jiatao Gu, Naman Goyal, Xian Li, Sergey Edunov, Marjan Ghazvininejad, Mike Lewis, and Luke Zettlemoyer. 2020. Multilingual denoising pre-training for neural machine translation. *Transactions of the Association for Computational Linguistics*, 8:726–742.

Zihan Liu, Jamin Shin, Yan Xu, Genta Indra Winata, Peng Xu, Andrea Madotto, and Pascale Fung. 2019. Zero-shot cross-lingual dialogue systems with transferable latent variables. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 1297–1303.

Myle Ott, Sergey Edunov, Alexei Baevski, Angela Fan, Sam Gross, Nathan Ng, David Grangier, and Michael Auli. 2019. fairseq: A fast, extensible toolkit for sequence modeling. In *Proceedings of NAACL-HLT 2019: Demonstrations*.

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. Bleu: a method for automatic evaluation of machine translation. In *Proceedings of the 40th annual meeting on association for computational linguistics*, pages 311–318. Association for Computational Linguistics.

Sebastian Schuster, Sonal Gupta, Rushin Shah, and Mike Lewis. 2019. Cross-lingual transfer learning for multilingual task oriented dialog. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 3795–3805.

Jörg Tiedemann and Santhosh Thottingal. 2020. OPUS-MT — Building open translation services for the World. In *Proceedings of the 22nd Annual Conferenec of the European Association for Machine Translation (EAMT)*, Lisbon, Portugal.

Gokhan Tur, Dilek Hakkani-Tür, and Larry Heck. 2010. What is left to be understood in ATIS? In *2010 IEEE Spoken Language Technology Workshop*, pages 19–24. IEEE.

Shyam Upadhyay, Manaal Faruqui, Gokhan Tür, Hakkani-Tür Dilek, and Larry Heck. 2018. (almost) zero-shot cross-lingual spoken language understanding. In *2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 6034–6038. IEEE.

Xinyu Wang, Yong Jiang, Nguyen Bach, Tao Wang, Fei Huang, and Kewei Tu. 2020. Structure-level knowledge distillation for multilingual sequence labeling. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 3317–3330.

Ronald J Williams. 1992. Simple statistical gradient-following algorithms for connectionist reinforcement learning. *Machine learning*, 8(3-4):229–256.

Weijia Xu, Batool Haider, and Saab Mansour. 2020. End-to-end slot alignment and recognition for cross-lingual nlu. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 5052–5063.

Using Structured Content Plans for Fine-grained Syntactic Control in Pretrained Language Model Generation

Fei-Tzin Lee*, **Miguel Ballesteros**†, **Feng Nan**† **and Kathleen McKeown***
* Computer Science Department, Columbia University
† Amazon AI Labs
{feitzin,kathy}@cs.columbia.edu
{ballemig,nanfen}@amazon.com

Abstract

Large pretrained language models offer powerful generation capabilities, but cannot be reliably controlled at a sub-sentential level. We propose to make such fine-grained control possible in pretrained LMs by generating text directly from a semantic representation, Abstract Meaning Representation (AMR), which is augmented at the node level with syntactic control tags. We experiment with English-language generation of three modes of syntax relevant to the framing of a sentence - verb voice, verb tense, and realization of human entities - and demonstrate that they can be reliably controlled, even in settings that diverge drastically from the training distribution. These syntactic aspects contribute to how information is framed in text, something that is important for applications such as summarization which aim to highlight salient information.

1 Introduction

Language models pretrained on enormous corpora have become a staple in natural language processing because of their power and adaptability (Devlin et al., 2019; Lewis et al., 2020; Raffel et al., 2020). These models exhibit strong performance across a range of applications, but are not inherently controllable beyond the choice of input. To enforce specific constraints on their output, we must introduce an additional mechanism for control, whether by inserting control codes during pretraining (Keskar et al., 2019) or by adding components to the model (Dathathri et al., 2020). These prior methods allow specification of high-level attributes (e.g., topic or sentiment) but leave the specifics of sub-sentential realization to the model.

In contrast, we present a method for controlling generation of specific verbs and entities within a sentence. We investigate the setting in which a pretrained language model is used not as an end-to-end solution, but rather to directly generate text from a predefined content plan. Such a content plan may be created at an intermediate stage in a given task (e.g., as the product of content selection over a document graph in AMR-based summarization) but is not inherently task-specific.

Specifically, we focus on the controllability of BART when it is fine-tuned to generate text from an intermediate Abstract Meaning Representation (AMR) (Banarescu et al., 2013), a form of graphical semantic representation. We choose AMR because it is relatively widely used, including as an intermediate representation in summarization (Liu et al., 2015; Hardy and Vlachos, 2018) and machine translation (Song et al., 2019), and the problem of AMR-to-text generation is well-studied (Konstas et al., 2017; Wang et al., 2020; Bai et al., 2020; Zhang et al., 2020). We focus on BART because of its competitive performance in summarization, a task where we feel our work is particularly applicable, as document-level content graphs have been shown to be useful intermediate representations in long-context summarization settings (Wu et al., 2021).

Our setting makes use of the powerful generation abilities of a pretrained language model while also exposing a direct graphical representation of the content, which allows us to target control tags to specific nodes in order to impose fine-grained control (i.e., constrain the text-level realization of specific verbs or entities). This is not possible in an end-to-end approach to some tasks, such as in summarization: although high-level attributes can be controlled during generation, there is no pre-existing syntactic or content plan, and so verbs and entities in the summary cannot be directly controlled as they have not been determined yet. In cases where we desire such control - e.g., in query-focused summarization, or when highlighting important entities (Nenkova et al., 2005) - it is thus useful to have a representation that specifies syntactic aspects at the level of individual verbs or entities.

Proceedings of the 29th International Conference on Computational Linguistics, pages 5882–5895
October 12–17, 2022.

In our experiments, we augment the AMR input to our generator with tags which we automatically infer for three modes of fine-grained syntax, which we choose for their relevance to summarization: verb voice (active or passive), verb tense, and syntactic realization of human entities (i.e., names and pronouns). Controlling voice and entities contributes to the model's ability to use syntax to highlight a specific topic or focus, following centering theory (Grosz et al., 1995), which is important in cases where the focus of the summary may differ from the focus of sentences in the document. As plain AMR does not contain information about tense, which is important for maintaining faithful summaries, we also consider controlling verb tense to avoid generating hallucinations about when an event took place.

We find that finetuning BART to generate from AMR augmented with these syntax tags makes it largely controllable across all three types of syntax. Importantly, this holds true even when the finetuned models are evaluated on a test set designed to have a radically different class distribution than the training set, without any tradeoff as to the fluency of the generated output. We further find that training the same model with control on multiple syntactic modes improves performance on voice controllability, though not on tense. Our experiments show that our tagged models are far better at controlling voice, tense and entity realization than a model without tags.

In summary, our contributions are:

- A method of automatically extracting labels for each of three modes of fine-grained syntactic information for relevant AMR nodes (verb voice, verb tense, and entity realization), allowing us to create high-quality fine-tuning data at scale;
- Experiments demonstrating controllability in pretrained BART models finetuned for generation from tag-augmented AMR in both in-distribution and off-distribution settings;
- Ablation analyses and experiments on interactions between modes of syntax demonstrating which modes work well together.

Our code will be made available at https://github.com/feitzin/amr-controlled-generation.

2 Related Work

Controllable generation. Most prior work on controllable generation focuses on global attributes that apply to the entire output (Hu et al., 2017; Shen et al., 2017; Chawla and Yang, 2020), rather than fine-grained control of the realization of individual units of content, as we do. This includes work on controllable generation with pretrained language models (Keskar et al., 2019; Dathathri et al., 2020).

Work on controllable summary generation also shares this focus on global attributes. Fan et al. (2018) trains a convolutional model to generate summaries controlled by attribute markers for length, entities to focus upon, domain, and subset of the text. He et al. (2020) fine-tunes a BART model to generate output summaries that are controlled using keywords or prompts, allowing the model to focus on specific entities or desired information. This approach addresses a similar problem to our work, but focuses on global rather than fine-grained control, does not necessarily frame a summary around selected relevant content, and is applicable to classic single-document summarization, whereas our approach is generalizable to multi-document and long-document settings due to its use of a content selection model

Pipelines in surface realization. Elder et al. (2019) demonstrate that a symbolic intermediate representation (based on a dependency graph) used as input to a neural generator can yield improvements on the surface realization task. Castro Ferreira et al. (2019) find that a pipeline of discrete modules can yield improvements over end-to-end neural models for data-to-text generation. However, Farahnak et al. (2020) find that a pretrained language model (BART) can outperform previous state-of-the-art modular pipelines on surface realization. We aim to take the best of both by using BART as our generator, but leaving the choice of content selector free.

AMR-to-text generation. There is an active body of work on AMR-to-text generation (Konstas et al., 2017; Wang et al., 2020; Bai et al., 2020; Zhang et al., 2020), but most of this work uses architectures specialized for the AMR-to-text setting. In contrast, our work focuses on adapting this setting to controllable generation with pretrained language models. To our knowledge, the most closely related work to ours is that of Ribeiro et al. (2020), who investigate the use of pretrained language mod-

els for multiple graph-to-text generation settings, and whose finetuning setup we follow closely.

3 Data and Methodology

The AMR Bank (Knight et al., 2020) is the largest gold standard corpus for AMR, but it contains only around 60,000 annotated sentences in total, which is a relatively small amount of data for finetuning a large model such as BART. Thus, for finetuning, we instead use a much larger text corpus which we parse automatically using the published code for the AMR parser of Cai and Lam (2020). As the application we are interested in is summarization, we use the multidocument summarization corpus of Gholipour Ghalandari et al. (2020), which consists of 10,200 clusters containing on average 235 documents each. We refer to this as the *reservoir corpus*. Although we do not use the AMR Bank for finetuning, we do use its *proxy* subset, which contains news documents and summaries, as a second evaluation set.

Due to constraints on time and processing power, we do not use the entire reservoir corpus for finetuning, but rather finetune our models primarily on the reservoir validation set, which contains 580,787 sentences in total. We split the reservoir corpus' validation set into a training set of 500,000 sentences and validation set of 80,000 sentences, which we use for finetuning. For evaluation, we sample 10,000 sentences from the reservoir training set to use for model analysis, and sample 10,000 sentences from the reservoir test set for final performance numbers. We reserve the remainder of the reservoir training set for experiments that require filtering out a portion of the data. We give an overview of the reservoir corpus' split sizes and partitioning in Table 2.

3.1 AMR Linearization

We use finetuned BART models for AMR-to-text generation, but BART takes sequences as input rather than arbitrary directed graphs. Thus, following the methodology of Ribeiro et al. (2020), we linearize AMR graphs into a modified version of PENMAN format (Kasper, 1989) that omits identifying handles for each node: for each AMR graph, we perform a depth-first traversal of the graph, adding node and edge labels to the linearized sequence in order, as well as parentheses indicating depth levels (see Table 1 for an example).

3.2 Syntax labeling

In overview, our labeling procedure involves three steps: (1) extract syntactic labels from the raw text; (2) use the extracted labels to augment the linearized AMR for input to our generation models; (3) extract syntactic labels a second time from the models' output to evaluate against the original tags. We use spaCy (Honnibal et al., 2020) for dependency parsing and part-of-speech tagging to extract syntactic labels. When augmenting linearized AMR, we insert syntactic tags as a modifier directly following the relevant node (see Table 1).

We provide class distributions for each mode of syntax in Appendix A. We note that the voice and entity classes are highly imbalanced, while tense is relatively more evenly distributed across classes.

Voice. To extract passive/active labels from a sentence, we examine the automatically extracted dependency parse for the sentence and individually label each verb as active or passive according to its edge labels. We verify the reliability of this method in §6.4.

Given these labels, we identify corresponding nodes in the AMR graph by performing exact string matching between the lemmas of each labeled verb and the concept labels of all AMR nodes representing a verb. Verb nodes that are not matched to any labeled verb lemma are not assigned a label. In linearization, the label appears as an additional modifier after the concept string of the verb: either `:active` or `:passive`.

Tense. We use the fine-grained part-of-speech tag under the Penn Treebank labeling scheme (Marcus et al., 1993) as the tense tag for each verb. We obtain these tags directly from the part-of-speech tagger.

Entity realization. We filter the data in our entity realization experiments to consider only sentences that fulfill two requirements. First, there must be at most one `person` node in the AMR. Second, the sentence must contain at most one of the pronouns {she, he, they} or their associated forms (i.e., a sentence containing "he" and "himself" would be acceptable; a sentence containing "her" and "their" would be discarded). This eliminates the possibility of introducing additional error using automatic coreference.

We assume that if one of these pronouns occurs in such a sentence, it is associated with the single `person` node in the AMR, and give that node the appropriate tag among

Sentence	**They had received a call to conduct a background check about 6:15 p.m.**
Linearized AMR (voice tags)	(receive **:active** :ARG0 (they) :ARG1 (call :ARG0 they :ARG1 (conduct **:active** :ARG0 they :ARG1 (check :ARG0 they :ARG1 (background))) :ARG1 (rate-entity-91 :ARG2 (temporal-quantity :quant 1 :unit (minute)) :ARG4 (temporal-quantity :quant 1 :unit (hour)))))

Table 1: An example sentence and linearized AMR with voice tags inserted. The verbs tagged for voice in the second example are “receive" (active) and “conduct" (active). Tags are bolded for readability.

Split	Sentences	Usage
Train	4.38M	Held in reserve. (10k split for analysis.)
Val	581k	Finetuning data (500k train, 80k validation).
Test	543k	10k sampled from test set.

Table 2: Partitioning of the larger reservoir corpus.

`:pronoun-{he/she/they}`. If the `person` node is named in the AMR, we give it the tag `:named`. Otherwise, we assume that the person in question is described in some other way, such as by profession (e.g. “scientist") or by an action they perform (e.g. “visitor") and give it the `:desc` tag. We remove the `:desc` class from our experiments because it encompasses both generic and specific references, while we are interested only in specific references (for details, see Appendix B).

In our entity realization experiments, we thus filter out sentences with `:desc` tags or multiple person nodes from the reservoir training set until we have an equivalent amount of data to that used in the other experiments, giving us alternate training and validation sets of equal size. To obtain our entity test set, we filter out sentences with `:desc` tags or multiple person nodes, as well as sentences with no person nodes, until we have 10,000 sentences.

3.3 Finetuning

We closely follow the methodology of Ribeiro et al. (2020) for finetuning. We finetune multiple pretrained BART-large models on AMR graph-sentence pairs to produce the sentence text given different variations of a linearized AMR graph. Specifically, once we have the base linearized AMR, we insert our syntactic tags into the AMR as metadata tags for the appropriate nodes, and train a family of models to generate text with each of these types of augmentations (and without control tags, for comparison).

4 Experiments

In our experiments, we compare three types of models in total: first, a baseline “untagged" BART model finetuned on pure AMR-to-text generation with no control tags; second, BART models finetuned to produce text from linearized AMR with syntax tags for each mode of syntax individually; and finally, a set of models finetuned with control tags for multiple modes of syntax. For both voice and tense, we report results both on our own test set (10k sentences) as well as the test set of the proxy subset of the AMR Bank (approximately 800 sentences) for comparison on gold AMR parses. For entity, we report results only on our own test set, as after filtering the proxy test set to remove sentences with `:desc` tags and sentences with greater or fewer than one person node, only 16 sentences remained.

4.1 Hyperparameter settings

We use Fairseq (Ott et al., 2019) for finetuning. Based on preliminary experimental results, we evaluate all models after four epochs. We train all models using the Adam (Kingma and Ba, 2014) optimizer with a learning rate of 3×10^{-5} and polynomial learning rate decay. (For full hyperparameter settings, see Appendix C.)

4.2 Syntactic control

To investigate the effect of finetuning with our syntax tags, we automatically extract syntactic labels from each verb in the output from each model using our automatic labeling procedure. For each mode of syntax, we measure performance on generation with the corresponding labels as a classification task using macro F1. As person nodes may have both a `:named` tag and a pronoun tag attached, we measure entity realization performance on two separate tasks: whether our model generated a name

or not, and whether our model used the correct pronoun (if it used a pronoun at all). For tense and pronouns, we additionally provide breakdowns of F1 per class on our test set.

As we only measure F1 over the set of labeled nodes from the original sentence that are reproduced in the generated text (see §3.2), we additionally recorded the percentage of such nodes, i.e. the *node retention*, which indicates the proportion of labels that are kept for evaluation, rather than dropped because the corresponding nodes are not realized as the same lemma in the generated output. Across all models for all settings, node retention is upward of .8 on our test set.

Evaluating directly on the test set measures how well our finetuned models can reproduce the desired syntax in a setting where syntactic labels follow a similar distribution to the training set, as they are both drawn from the same base corpus. In order to isolate the effect that using control tags gives us, we also report performance in an off-distribution setting. For each mode of syntax, we create a *flipped* evaluation set by perturbing the control tags inserted into the evaluation inputs, which directs BART to generate less distributionally plausible voices: for voice, we swap active and passive tags; for tense, we swap between past and present, and for entities, we use a two-stage strategy for randomizing name and pronoun tags. For full details, see Appendix D.

4.3 Sentence quality

To measure fluency, we compute BLEU score (Papineni et al., 2002) of the generated text against the original sentence. While smatch (Cai and Knight, 2013) could be used to compare automatic AMR parses of the generated text against the input AMR, that would also inherently evaluate the AMR parser used, which is not our focus.

5 Results

5.1 Voice

We report performance of each model on the active/passive reproduction task in Table 3. The 'flipped' statistics (the second column in each pair) are on the flipped evaluation set, i.e. with all voice tags flipped, which forces the model to generate against the regular voice distribution.

The model finetuned with control tags performs noticeably better than the untagged model on the regular evaluation set, but its controllability truly shows through on the flipped set, where it outperforms the uncontrolled model by an order of magnitude. Though there is a sharp drop in performance from the original setting, it still manages to reproduce a nontrivial proportion of verbs in the specified voice even when the tags are flipped, indicating that it is indeed able to some extent to disregard whatever signal may be present in the raw AMR. We note that, although we naïvely flip all tags in the flipped setting, some verbs cannot appear in the passive (e.g., intransitive verbs such as "sleep"), which lowers the maximum possible score.[1]

Model	Test set F1	Test set Flip	Proxy test set F1	Proxy test set Flip
Untagged	0.833	0.048	0.630	0.086
Voice	0.965	0.498	0.909	0.516
v + t	0.957	**0.500**	0.934	0.546
v + e	**0.972**	0.463	0.941	0.541
v + t + e	0.965	0.499	**0.979**	**0.581**

Table 3: F1 of finetuned BART variants for verb voice, in original and flipped settings, on our test set and the proxy test set. Components of combined models are abbreviated: voice (v), tense (t), entity (e).

Interestingly, even the untagged model achieves impressive performance on the evaluation set, suggesting that the model does receive some signal as to verb voice. We hypothesize that it is picking up on the highly skewed verb distribution (see Appendix A), as the active voice is about an order of magnitude more prevalent in the data than the passive. However, its performance on the flipped evaluation set is trivially far worse, as the untagged model does not see the control tags at test time and produces exactly the same output either way.

We note that results on the proxy set are slightly lower in the original setting but slightly higher in the flipped setting compared to our original test set.

5.2 Tense

We report performance on tense in Table 4. We have a much more dramatic improvement for tense than for voice when comparing the model fine-tuned with tags to the model fine-tuned without; the untagged model does poorly even on the in-distribution evaluation set, whereas the tagged model does quite well on both. This may indicate that the distinctions between the multiple types of verb tense are more difficult for the model to

[1] See Appendix E for more detail.

Model	Flipping	Macro F1	Proxy F1	VB	VBD	VBG	VBN	VBP	VBZ
Untagged	None	0.691	0.448	0.830	0.747	0.687	0.757	0.562	0.564
Tense	None	**0.979**	**0.961**	**0.990**	**0.981**	**0.994**	**0.979**	0.949	0.982
v + t	None	0.978	0.953	0.988	0.976	0.992	0.974	**0.953**	**0.986**
v + t + e	None	0.971	0.941	0.986	0.968	0.991	0.967	0.933	0.983
Untagged	Flipped	0.185	0.196	0.826	0.114	0.035	0.075	0.006	0.055
Tense	Flipped	0.784	0.806	**0.986**	**0.909**	**0.871**	**0.830**	0.143	0.964
v + t	Flipped	**0.786**	**0.899**	0.983	0.902	0.859	0.800	**0.207**	**0.966**
v + t + e	Flipped	0.760	0.736	0.981	0.891	0.818	0.736	0.173	0.959

Table 4: Performance of finetuned BART models for verb tense. F1 is reported on both our test set and the proxy test set; individual class F1 scores are on our test set.

Model	Flipping	Name F1	Pronoun F1	`she`	`he`	`they`
Untagged	None	0.399	0.720	0.473	0.782	0.906
Entity	None	**0.407**	0.996	0.993	**0.998**	**0.998**
v + e	None	0.395	0.995	0.992	0.997	0.996
v + t + e	None	0.403	**0.999**	**1.000**	**0.998**	**0.998**
Untagged	Flipped	0.401	0.204	0.083	0.283	0.245
Entity	Flipped	0.399	**0.980**	**0.986**	0.985	**0.970**
v + e	Flipped	0.426	0.976	0.978	**0.988**	0.961
v + t + e	Flipped	**0.433**	0.967	0.974	0.975	0.953

Table 5: Performance of finetuned BART models for entity realization. F1 is reported for the binary named - not named task as well as for the pronoun generation task. Numbers here are only on our test set, as there were only 16 sentences remaining in the proxy set after filtering.

learn without supervision than the active/passive distinction. One note is that the VBP class seems to be more difficult to accurately reproduce than the others, perhaps due to its relatively small size (approximately 5% of all verbs).

5.3 Entity realization

We report performance on entity realization, our final syntactic task, in Table 5. Interestingly, it seems that names are quite difficult to learn - our scores simply measure whether the model generated a name or not, regardless of whether it was the correct name, and even in that case, F1 is quite low.

A second observation is that flipping does not seem to have a large effect on pronouns, which suggests that the model has learned to generalize across different types of pronouns quite well - there is not a noticeable difference between performance across pronoun classes, even though there is a moderate imbalance in the distribution.

In the case of names, scores for some models actually go up slightly in the flipped setting. This may indicate that the models have a tendency to guess something closer to the randomized distribution of name labels in the flipped evaluation sets.

5.4 Sentence quality

Model	Syntax mode	BLEU
Untagged	Voice	0.679
Voice	Voice	0.688
Untagged	Tense	0.679
Tense	Tense	0.703
Untagged	Entity	0.670
Entity	Entity	0.707

Table 6: BLEU scores for generated outputs from base tagged and untagged models against original sentences.

We report BLEU scores in the original (not-flipped) setting in Table 6. Adding tags in finetuning slightly improves BLEU, suggesting that the additional signal is helpful. At minimum, this indicates that we can finetune BART to use syntax control tags without having to worry about interfering with the content of its generated output.

Generator	Example sentence
Input	(lead *:passive* :ARG0 (this) :ARG0 (chief :ARG0-of (act *:active*)) :ARG1 (organization :name (Pentagon)) :frequency 3 :time (history))
Untagged	This is the third time in history that an acting chief has led the Pentagon.
Voice	This is the third time in history that the Pentagon **has been led** by an acting chief.
Input	(see *:passive* :ARG0 (shrine) :ARG1 (person :ARG0-of (visit) :source (religion :mod (all))) :duration (multiple :op1 (temporal-quantity :quant 1 :unit (century))))
Untagged	For centuries, the shrine has seen visitors from all religions.
Voice	The shrine **has been seen by** visitors from all religions for centuries.
Input	(change *:VBG* :ARG1 (intensity :mod (forecast)) :mod (also))
Untagged	There will also be changes in forecast intensity.
Tense	The forecast intensity **is also changing**.
Input	(announce :ARG1 (include :ARG1 (person *:named* :name (Jim) :wiki -) :ARG2 (honoree)) :time (previous))
Untagged	It was previously announced that Jim O'Brien would be included as an honoree.
Entity	It had previously been announced that Jim Hightower would be among the honorees.

Table 7: Example inputs and outputs from tagged and untagged models, with correct syntactic realizations in **bold** and incorrect underlined. Control tags in the input are *italicized*; these tags are not present in the version of the input passed to the untagged model.

6 Analysis

6.1 Syntax interactions

In order to investigate the interaction between the different modes of syntax, we additionally train a set of models that incorporate multiple types of tags. These are reported in the results tables as the "voice+tense", "voice+entity", and "voice+tense+entity" models.

Interestingly, adding tense seems to improve performance on voice, whereas the converse does not hold, while entity realization seems to be an orthogonal task: the "voice+tense" model achieves better performance on voice but not on tense in the more difficult flipped setting, whereas combining entity realization with other types of syntax leads to a drop in performance.

6.2 Qualitative analysis

We provide some examples of generated output in Table 7. A number of further examples are available in Appendix F. The first example is a case from the original evaluation set where the tagged voice model correctly generates the main verb ("lead") in the passive, whereas the untagged model incorrectly guesses it to be active. In such cases where a verb is generated in an unusual voice, the untagged model seems to make its guess based on the verb's more common voice, whereas the tagged model is able to adjust its output based on the control tag. The third example illustrates a comparable situation with tense generation.

The second example, from the flipped evaluation set, illustrates a different phenomenon we observed in some cases. The untagged model generates the voice that was correct in the original setting (but is incorrect here). The tagged model correctly generates the main verb ("seen") in the passive tense, but it makes a semantic error in doing so, changing the shrine to the object rather than the subject of the seeing - in a sense seeming to overcorrect for the change in voice.

Finally, we observed across models a tendency to hallucinate occasionally, particularly on numbers and on person and country names; while the tagged entity model usually correctly uses the name of a person when given in the input, even this model still sometimes hallucinates information that isn't there, such as the surname in the fourth example.

6.3 Structural ablation

We have now seen that we can successfully use tagged AMR as input to give us fine-grained controllability. However, it remains unclear exactly how much information from the tags the model is using, or how much it is able to infer on its own. In order to investigate precisely which parts of the AMR are necessary, we train a series of ablation models on the voice control task by gradually removing components of the input AMR.

Components removed	F1	Flipped F1
None (untagged)	0.838	0.047
None (voice tags)	0.953	0.431
Edges	0.964	0.438
Edges + structure	0.964	0.462
Edges + structure + tags	0.796	0.052

Table 8: Performance of full and ablated BART models on the analysis set for verb voice.

Ablation	BLEU
None (untagged)	0.717
None (voice tags)	0.731
Edges	0.713
Edges + structure	0.666
Edges + structure + tags	0.642

Table 9: BLEU scores for base and ablated models on the analysis set for voice.

We train three ablated AMR-to-text models: a model where we remove relation tags (i.e., edge labels); a model where we remove relations and graph structure (i.e., parentheses); and a model where we remove relations, structure, and the syntax control tags themselves.

We present results on ablated models alongside the original tagged model in Table 8, and include the ablated models' content metrics in Table 9. Surprisingly, our first two ablations (removing edge labels and parentheses) both yield slight improvements in voice controllability. However, this comes at the expense of BLEU score, which drops 2 points when edges are removed and 7 points when both edges and structure are removed. This suggests that removing edge and structural information somehow makes it easier for the models to focus on the correspondence between tags and syntax in the training data, but at the cost of information about content.

6.4 Voice tagging accuracy

As our voice labels are derived from automatic dependency parses, we check that our tagging method is giving us reasonable labels by evaluating it separately. We compare the voice tags from our tagging method against the gold voice labels from two Universal Dependencies treebanks in English (GUM and LinEs) and present the results in Table 10. Both treebanks present a similar active/passive skew to our data. On both treebanks, performance on the majority active class is very high, whereas performance on passive verbs differs between the two: on GUM, our tagging method still picks up passive verbs quite well, whereas on LinEs, recall on the passive class is much lower. Given the GUM results, in our experiments, we assume that our tagging method is reliable enough to use as gold standard, but in future research, further work on picking up the missing passive instances from LinEs-like sentences may thus prove valuable.

Treebank	True voice	Prec.	Recall	F1
GUM	Passive	0.982	0.885	0.931
GUM	Active	0.993	0.936	0.964
LinEs	Passive	0.937	0.468	0.625
LinEs	Active	0.941	0.918	0.930

Table 10: Our automatic voice tagging on the development sets of Universal Dependencies treebanks. Precision, recall and F1 are evaluated against gold labels.

7 Conclusion and future directions

In this paper, we investigate the controllability of three modes of syntax - verb voice, verb tense, and syntactic entity realization - when using BART to generate text from AMR input augmented with automatically extracted syntactic labels. We find that all three modes of syntax are more reliably reproduced when these augmentations are added, yielding more accurate and more faithful outputs. Further, even when we artificially engineer the distribution of tags to be as far from training as possible, the models with tags still far outperform the model without, without sacrificing fluency.

Ultimately, our labeling strategy allows us to automatically create data at good enough quality and scale that we can successfully finetune a pretrained LM generator to reliably follow a content plan augmented with fine-grained syntactic tags. This setting is particularly useful in tasks such as summarization, where there is no one-to-one mapping between input and output content, and thus no appropriate place to insert fine-grained tags in the original input.

There are many avenues of possible future study with our method of controllable generation; one natural future direction would be an expansion of the setting from individual sentence AMRs to a collection of AMRs forming a contiguous passage or document, as is the ultimate goal of this work.

8 Acknowledgements

The authors thank the anonymous reviewers for their comments and suggestions. We are also grateful to Chris Kedzie for his advice and insight over the course of this project.

This research was supported in part by the Columbia Center of Artificial Intelligence Technology in collaboration with Amazon.

This paper is based upon work supported in part by the National Science Foundation Graduate Research Fellowship under Grant No. DGE-16-44869. Any opinion, findings, and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of the National Science Foundation.

References

Xuefeng Bai, Linfeng Song, and Yue Zhang. 2020. Online back-parsing for AMR-to-text generation. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1206–1219, Online. Association for Computational Linguistics.

Laura Banarescu, Claire Bonial, Shu Cai, Madalina Georgescu, Kira Griffitt, Ulf Hermjakob, Kevin Knight, Philipp Koehn, Martha Palmer, and Nathan Schneider. 2013. Abstract Meaning Representation for sembanking. In *Proceedings of the 7th Linguistic Annotation Workshop and Interoperability with Discourse*, pages 178–186, Sofia, Bulgaria. Association for Computational Linguistics.

Deng Cai and Wai Lam. 2020. AMR parsing via graph-sequence iterative inference. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 1290–1301, Online. Association for Computational Linguistics.

Shu Cai and Kevin Knight. 2013. Smatch: an evaluation metric for semantic feature structures. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 748–752, Sofia, Bulgaria. Association for Computational Linguistics.

Thiago Castro Ferreira, Chris van der Lee, Emiel van Miltenburg, and Emiel Krahmer. 2019. Neural data-to-text generation: A comparison between pipeline and end-to-end architectures. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 552–562, Hong Kong, China. Association for Computational Linguistics.

Kunal Chawla and Diyi Yang. 2020. Semi-supervised formality style transfer using language model discriminator and mutual information maximization. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 2340–2354, Online. Association for Computational Linguistics.

Sumanth Dathathri, Andrea Madotto, Janice Lan, Jane Hung, Eric Frank, Piero Molino, Jason Yosinski, and Rosanne Liu. 2020. Plug and play language models: A simple approach to controlled text generation. In *International Conference on Learning Representations*.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, Minneapolis, Minnesota. Association for Computational Linguistics.

Henry Elder, Jennifer Foster, James Barry, and Alexander O'Connor. 2019. Designing a symbolic intermediate representation for neural surface realization. In *Proceedings of the Workshop on Methods for Optimizing and Evaluating Neural Language Generation*, pages 65–73, Minneapolis, Minnesota. Association for Computational Linguistics.

Angela Fan, David Grangier, and Michael Auli. 2018. Controllable abstractive summarization. In *Proceedings of the 2nd Workshop on Neural Machine Translation and Generation*, pages 45–54, Melbourne, Australia. Association for Computational Linguistics.

Farhood Farahnak, Laya Rafiee, Leila Kosseim, and Thomas Fevens. 2020. Surface realization using pretrained language models. In *Proceedings of the Third Workshop on Multilingual Surface Realisation*, pages 57–63, Barcelona, Spain (Online). Association for Computational Linguistics.

Demian Gholipour Ghalandari, Chris Hokamp, Nghia The Pham, John Glover, and Georgiana Ifrim. 2020. A large-scale multi-document summarization dataset from the Wikipedia current events portal. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 1302–1308, Online. Association for Computational Linguistics.

Barbara J. Grosz, Aravind K. Joshi, and Scott Weinstein. 1995. Centering: A framework for modeling the local coherence of discourse. *Computational Linguistics*, 21(2):203–225.

Hardy and Andreas Vlachos. 2018. Guided neural language generation for abstractive summarization using Abstract Meaning Representation. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 768–773, Brussels, Belgium. Association for Computational Linguistics.

Junxian He, Wojciech Kryściński, Bryan McCann, Nazneen Rajani, and Caiming Xiong. 2020. Ctrl-sum: Towards generic controllable text summarization. *arXiv preprint arXiv:2012.04281*.

Matthew Honnibal, Ines Montani, Sofie Van Landeghem, and Adriane Boyd. 2020. spaCy: Industrial-strength Natural Language Processing in Python.

Zhiting Hu, Zichao Yang, Xiaodan Liang, Ruslan Salakhutdinov, and Eric P. Xing. 2017. Toward controlled generation of text. In *Proceedings of the 34th International Conference on Machine Learning*, volume 70 of *Proceedings of Machine Learning Research*, pages 1587–1596. PMLR.

Robert T. Kasper. 1989. A flexible interface for linking applications to Penman's sentence generator. In *Speech and Natural Language: Proceedings of a Workshop Held at Philadelphia, Pennsylvania, February 21-23, 1989*.

Nitish Shirish Keskar, Bryan McCann, Lav R Varshney, Caiming Xiong, and Richard Socher. 2019. Ctrl: A conditional transformer language model for controllable generation. *arXiv preprint arXiv:1909.05858*.

Diederik P Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*.

Kevin Knight, Bianca Badarau, Laura Baranescu, Claire Bonial, Madalina Bardocz, Kira Griffitt, Ulf Hermjakob, Daniel Marcu, Martha Palmer, Tim O'Gorman, and Nathan Schneider. 2020. Abstract meaning representation (AMR) annotation release 3.0. Web Download.

Ioannis Konstas, Srinivasan Iyer, Mark Yatskar, Yejin Choi, and Luke Zettlemoyer. 2017. Neural AMR: Sequence-to-sequence models for parsing and generation. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 146–157, Vancouver, Canada. Association for Computational Linguistics.

Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Veselin Stoyanov, and Luke Zettlemoyer. 2020. BART: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7871–7880, Online. Association for Computational Linguistics.

Fei Liu, Jeffrey Flanigan, Sam Thomson, Norman Sadeh, and Noah A. Smith. 2015. Toward abstractive summarization using semantic representations. In *Proceedings of the 2015 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 1077–1086, Denver, Colorado. Association for Computational Linguistics.

Mitchell P. Marcus, Beatrice Santorini, and Mary Ann Marcinkiewicz. 1993. Building a large annotated corpus of English: The Penn Treebank. *Computational Linguistics*, 19(2):313–330.

Ani Nenkova, Advaith Siddharthan, and Kathleen McKeown. 2005. Automatically learning cognitive status for multi-document summarization of newswire. In *Proceedings of Human Language Technology Conference and Conference on Empirical Methods in Natural Language Processing*, pages 241–248, Vancouver, British Columbia, Canada. Association for Computational Linguistics.

Myle Ott, Sergey Edunov, Alexei Baevski, Angela Fan, Sam Gross, Nathan Ng, David Grangier, and Michael Auli. 2019. fairseq: A fast, extensible toolkit for sequence modeling. In *Proceedings of NAACL-HLT 2019: Demonstrations*.

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. Bleu: a method for automatic evaluation of machine translation. In *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics*, pages 311–318, Philadelphia, Pennsylvania, USA. Association for Computational Linguistics.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J. Liu. 2020. Exploring the limits of transfer learning with a unified text-to-text transformer. *Journal of Machine Learning Research*, 21(140):1–67.

Leonardo F. R. Ribeiro, Martin Schmitt, Hinrich Schütze, and Iryna Gurevych. 2020. Investigating pretrained language models for graph-to-text generation.

Tianxiao Shen, Tao Lei, Regina Barzilay, and Tommi Jaakkola. 2017. Style transfer from non-parallel text by cross-alignment. In *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc.

Linfeng Song, Daniel Gildea, Yue Zhang, Zhiguo Wang, and Jinsong Su. 2019. Semantic neural machine translation using AMR. *Transactions of the Association for Computational Linguistics*, 7:19–31.

Tianming Wang, Xiaojun Wan, and Hanqi Jin. 2020. AMR-to-text generation with graph transformer. *Transactions of the Association for Computational Linguistics*, 8:19–33.

Wenhao Wu, Wei Li, Xinyan Xiao, Jiachen Liu, Ziqiang Cao, Sujian Li, Hua Wu, and Haifeng Wang. 2021. BASS: Boosting abstractive summarization with unified semantic graph. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 6052–6067, Online. Association for Computational Linguistics.

Yan Zhang, Zhijiang Guo, Zhiyang Teng, Wei Lu, Shay B. Cohen, Zuozhu Liu, and Lidong Bing. 2020. Lightweight, dynamic graph convolutional networks for AMR-to-text generation. In *Proceedings of the*

2020 Conference on Empirical Methods in Natural Language Processing (EMNLP), pages 2162–2172, Online. Association for Computational Linguistics.

A Data Imbalance

We provide the breakdown of label distributions within each class in Table 11.

Mode	Classes
Voice	Active (0.926)
	Passive (0.074)
Tense	VB (0.216)
	VBD (0.259)
	VBG (0.165)
	VBN (0.230)
	VBP (0.053)
	VBZ (0.078)
Entity	desc (0.770)
	named (0.109)
	pronoun-she (0.010)
	pronoun-he (0.041)
	pronoun-they (0.070)

Table 11: Class distributions for each syntax mode.

B The description class

As mentioned in §3.2, the description class contains not only cases where the entity described is a specific reference (i.e., a particular identifiable person), but also generic references (i.e., terms describing classes of people, such as “visitors to the location"). Our focus in this paper is on realization of specific entities and not generics; we thus omit sentences containing the `:desc` class from our experiments. This has the incidental benefit of leaving us with much more balanced data, as descriptions originally made up the majority class (approximately 80% of all person instances).

C Finetuning details

In our experiments, we use Fairseq to finetune from a pretrained bart-large model for four epochs using Adam; we use a learning rate of 3e-05, dropout of 0.1, and polynomial learning rate decay with 500 warmup updates and 2,000,000 total updates.

D Label flipping

To create our flipped data, we use the following strategies to perturb labels:

- **Voice.** Swap active and passive tags.
- **Tense.** Flip between past and present, i.e., VBG and VBN are swapped, and VBD is swapped with VBP and VBZ. (VB is left unchanged.)
- **Entities.** If the node has only a pronoun tag, we replace it with a random pronoun tag that differs from the original, and with 0.5 probability we add a dummy name node (drawn from a list of the top 100 most common unisex names in the United States[2]) and a `:named` tag. If the node originally had a `:named` tag, with 0.5 probability we remove it and add a random pronoun tag that differs from the one it had, if any.

E Feasibility of voice flipping

We performed a hand analysis of 50 sentences to estimate the proportion of sentences in which it was possible to flip the voice of the main verb. For sentences where it was possible to flip the voice, we also judged whether that would result in an undesirably awkward sentence (e.g., “they clinked glasses" versus “glasses were clinked by them") and whether it would have required significant modification of sentence structure (e.g., reordering clauses).

We found that in 12 sentences the main verb could not be flipped. Of those sentences where the main verb could be flipped, in 6 it would have resulted in an awkward sentence and in 10 it would have required some modification of sentence structure to preserve the meaning of the sentence.

This suggests that the effective ceiling on possible performance on voice controllability in the flipped setting is somewhere between approximately 64% and 76% (depending on whether awkward sentences are considered infeasible), and achieving performance higher than approximately 44% may be particularly difficult, as perfectly flipping the voice would require learning to modify the structure of the entire sentence.

F Further examples

We provide further examples of generated output in Table 12.

One observation is that correct syntactic realization does not necessarily entail correct semantics or freedom from hallucination, as evidenced in the third example, where both models hallucinate an update frequency of every 15 minutes rather than every 10 minutes. Hallucinations about person and

[2] https://github.com/fivethirtyeight/data/tree/master/unisex-names

country names are particularly common, as demonstrated in the fourth example, where both models hallucinate different country names for the country entities in the input, whereas the names are not explicitly provided.

Sometimes it seems that the tagged models provide improvements over the untagged model that we do not directly measure in our syntactic evaluation; the final two examples illustrate cases where the entity model correctly names an entity that the untagged model gave an incorrect name derived from its wiki tag (rather than its name tag), although both would have been considered correct in our evaluation as a name was produced for both.

Generator	Example sentence
Input	(contrast :ARG1 (welcome *:passive* :ARG0 (person :ARG0-of (criticize) :quant (many)) :ARG1 (move)) :ARG2 (say *:active* :ARG0 (some) :ARG1 (and :op1 (enough :ARG0 move) :op2 (come *:active* :ARG1 move :time (late :degree (too))))))
Untagged	Many welcomed the move, but some said it wasn't enough and came too late.
Voice	The move **was welcomed by** many critics, but some said it wasn't enough and came too late.
Input	(make *:VBN* :ARG0 (country :name (France) :wiki "France") :ARG1 (point :topic (reduce *:VBG* :ARG0 country :ARG1 (impact :ARG0 (environment) :ARG0 (meal :ARG1-of (serve *:VBN* :time (summit)))))) :mod (also))
Untagged	France will also make a point of reducing the environmental impact of the meals served during the summit.
Tense	France **has also made** a point of reducing the environmental impact of the meals served at the summit.
Input	(update *:VBZ* :ARG1 (map :mod (interactive)) :frequency (rate-entity-91 :ARG3 (temporal-quantity :quant 10 :unit (minute))))
Untagged	The interactive map will be updated every 15 minutes.
Tense	The interactive map **updates** every 15 minutes.
Input	(and :op1 (effect :ARG1 (watch :ARG1 (hurricane) :ARG1 (island))) :op2 (post *:VBN* :ARG1 (warn :ARG1 (storm)) :ARG2 (and :op1 (country) :op3 (country))))
Untagged	A hurricane watch is in effect for the islands, and storm warnings **have been posted** for Curaçao, Aruba, Bonaire and Curacao del Sur.
Tense	A hurricane watch is in effect for the islands, and storm warnings **have been posted** for Guam, Malawi and the island of Curaçao.
Input	(sit-down-02 :ARG1 (gauntlet :part-of (negotiate :ARG2 (and :op1 (crisis :mod (international)) :op2 (war :topic (trade))))) :ARG2 (person *:named* :name (Putin) :wiki "Vladimir_Putin"))
Untagged	the gauntlet of negotiations over international crises and trade wars will be sat down with **Vladimir_Putin**.
Entity	the gauntlet of negotiations on international crises and trade wars is a sit-down with **Putin**.
Input	(say :ARG0 (person *:named* :name (Trump) :wiki "Donald_Trump") :ARG1 (pressure :ARG0 (time) :degree (absolute)))
Untagged	"Time is absolutely under pressure," said **Donald_Trump Jr.**
Entity	"Time is absolutely under pressure," **Trump** said.

Table 12: Example inputs and outputs from tagged and untagged models, with correct syntactic realizations in **bold** and incorrect underlined. Control tags in the input are *italicized*; these tags are not present in the version of the input passed to the untagged model.

PrefScore: Pairwise Preference Learning for Reference-free Summarization Quality Assessment

Ge Luo and **Hebi Li** and **Youbiao He** and **Forrest Sheng Bao**
Iowa State University
Ames, IA, USA
{gluo,hebi,yh54}@iastate.edu, forrest.bao@gmail.com

Abstract

Evaluating machine-generated summaries without a human-written reference summary has been a need for a long time. Inspired by preference labeling in existing work of summarization evaluation, we propose to judge summary quality by learning the preference rank of summaries using the Bradley-Terry power ranking model from inferior summaries generated by corrupting base summaries. Extensive experiments on several datasets show that our weakly supervised scheme can produce scores highly correlated with human ratings.

1 Introduction

Summarization is a task in natural language processing in which automatic systems generate summaries from documents. To judge the quality of system-generated summaries, human evaluation is the best option, but it is non-trivial and laborious. Hence, many automatic metrics have been developed. They can be categorized as reference-based ones and reference-free ones, depending on whether reference summaries are needed in the evaluation stage.

Reference-based metrics include ROUGE (Lin, 2004), BLEU (Papineni et al., 2002), CIDEr (Vedantam et al., 2015), METEOR (Banerjee and Lavie, 2005), S^3 (Peyrard et al., 2017), MoverScore (Zhao et al., 2019), BertScore (Zhang et al., 2020), etc. Calculating the lexical overlap or the embedding similarity between a system-generated summary and its corresponding human-written reference summary, they reportedly have high correlations with human assessments.

Because creating human-written reference summaries is laborious and expensive, recent works are shifting to reference-free metrics. SummaQA (Scialom et al., 2019) and BLANC (Vasilyev et al., 2020) leverage pretrained language models to carry out text understanding tasks to evaluate the helpfulness of a summary for understanding its source document. SUPERT (Gao et al., 2020b) measures the semantic similarity against a pseudo reference summary in a multi-document summarization setting. However, reference-free metrics may show a lower correlation (Fabbri et al., 2021) with human evaluation scores than some of the reference-based metrics.

To trade off between the human effort needed and the quality of the evaluation, some work pursues a pairwise preference approach which collects preference labels over sentences in documents or over summaries from a human assessor as it requires less cognitive effort than writing a reference summary or manually scoring a machine-generated summary. Zopf (2018) proposes a reference-free evaluation approach by estimating sentence-level preferences on source documents rather than directly on the generated summaries. Gao et al. (2020a) train a linear model to estimate a summary preference utility function via active preference learning to guide a reinforcement learning based summarization system. But they do not examine the learned preference model as a metric for summarization evaluation.

Inspired by human-involved pairwise preference in summarization evaluation (Zopf, 2018; Gao et al., 2020b) and simple NLP data augmentation methods like EDA (Wei and Zou, 2019), in this work, we explore reference-free summary quality assessment via pairwise preference learning using negative sampling. A pre-trained text embedding model is used in a siamese network to learn the preference utility in an end-to-end, weakly supervised fashion. The closest work to ours is LS_Score (Wu et al., 2020). We achieve improved performance by using a better-attended model, a loss function based on preference learning, and introducing a mixed transitive negative sampling strategy. In addition, we promote our work to cross-domain and multi-document settings.

We show that the learned models are competitive

Proceedings of the 29th International Conference on Computational Linguistics, pages 5896–5903
October 12–17, 2022.

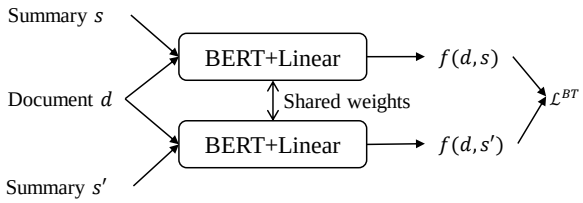

Figure 1: Model architecture.

compared to the state-of-the-art reference-free metrics. Our code is at https://github.com/NKWBTB/PrefScore.

2 Method

2.1 Model Architecture

The goal of a reference-free evaluation system is to learn a regressor f which takes a document d and its summary s as the input to produce a score $f(d, s)$ which represents the quality of the summary s. Learning such a regressor via supervised learning is very difficult because existing human-rated summary evaluation datasets (NIST, 2010; Grusky et al., 2018; Bhandari et al., 2020) contain too few samples, around 100 samples each, to train a generalizable model.

Therefore, we use pairwise preference learning as a weakly supervised workaround. By corrupting a summary into an inferior one, existing summarization datasets containing no human ratings as training labels but only gold, reference summaries can be transformed into massive training data for preference learning.

The training label is designed based on the Bradly-Terry (BT) model (Bradley and Terry, 1952). Given a reference summary s and a perturbed summary s' of the document d, the BT model estimates $f(d, s)$ and $f(d, s')$ such that the probability of s being superior than s' is:

$$p(s \succ s'|d) = \frac{\exp(f(d,s))}{\exp(f(d,s)) + \exp(f(d,s'))}. \quad (1)$$

This leads to our model design (Figure 1) using a siamese network. Leveraging the recent work of BERT-like (Devlin et al., 2019) contextualized embedding, a document d and a summary s are viewed as two sequence of tokens T_d and T_s. The input sequence are constructed as $([\text{CLS}], T_d, [\text{SEP}], T_s, [\text{SEP}])$, then the output of the [CLS] token containing both information from document and summary are sent to a linear layer to produce the final score $f(d, s)$. During the training, a pair of summaries will be sent to the siamese network. It can be seen as training a classifier to determine which summary is better. The loss is therefore:

$$\mathcal{L}^{BT} = -\sum_{d}\sum_{s' \in S'} [\log(p(s \succ s'|d))] \quad (2)$$

where S' is a set of inferior summaries deviated from s in methods to be discussed below in § 2.2. The learned ranking utility f is used as our summary evaluator and does not require a reference summary in the test/evaluation stage.

2.2 Mixed Transitive Negative Sampling

Given a reference summary s, we can obtain the set $S' = \{s'_1, s'_2, \ldots, s'_n\}$ of inferior summaries by mutating the reference summary s iteratively: s'_1 is mutated from s, s'_2 from s'_1, and so on. We can obtain a preference sequence of summaries $s \succ s'_1 \succ \cdots \succ s'_n$. The process is illustrated in Figure 2. In each iteration, unmodified tokens in s'_i is randomly selected and mutated to generate summary s'_{i+1}. The process continues until all tokens are mutated.

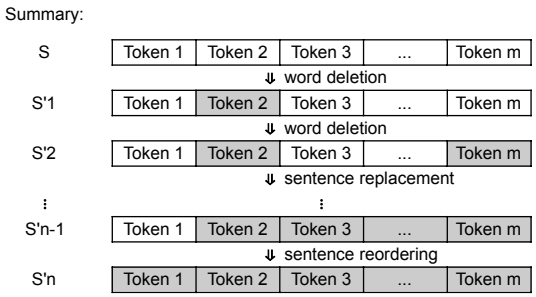

Figure 2: An example of the mixed transitive negative sampling process. The original part is in white, while the modified part is indicated as grey blocks.

Four mutation methods are employed: 1) **deleting a sentence** from the summary, resulting in information loss in the summary. 2) **replacing a sentence** in the summary with a sentence from other summaries, introducing extra information and redundancy in the summary. 3) **deleting a word** from the summary, influencing the sentence structure and readability. 4) **reorder sentences or words**, aggravating the coherence in the summary.

In each iteration, one of the four mutation methods is randomly chosen. Unlike plain negative

sampling that mutates samples in only one way or in only one iteration, our *mixed transitive negative sampling* accumulates the effects of different mutations into samples, enabling a model trained upon to learn different aspects of summaries.

3 Experiments

3.1 Test Sets

There are not many datasets with human evaluations to machine-generated summaries. Unfortunately, they are almost all in the news article domains. We use three established ones:

TAC2010 (NIST, 2010) is a multi-document summarization dataset which reports three scores: content, fluency and overall. It consists of 46 topics, each of which is associated with a set of 10 documents. We evaluate the metrics over summaries generated by 43 systems. For a summary, we calculate the mean score for all documents paired with the summary as an extension for our metric in the multi-document scenario. Only Set A for the regular summarization task is used here.

Newsroom (Grusky et al., 2018) is a single-document summarization dataset reporting four scores: INFormativeness, RELevance, COHerence and FLUence. It contains human-rated summaries generated by 7 systems for 60 documents. Each document-summary pair is rated by three human annotators. We use their mean score as the groundtruth score.

RealSumm (Bhandari et al., 2020), a recent single-document dataset reporting the LitePyramid (Shapira et al., 2019) score which is also content-focused. It sampled 100 documents from the CNN/DailyMail (See et al., 2017) test set, and collected human ratings for summaries generated by 11 extractive systems and 14 abstractive systems.

3.2 Training Sets (documents and reference summaries only, no human evaluations)

Because the test sets are all in the news domain, we select one training set from the news domain for in-domain analysis: **CNN/DailyMail** (CNNDM) (See et al., 2017). For cross-domain analysis, three training sets from different non-news domains are selected: **Billsum** (Kornilova and Eidelman, 2019) from legislative bills, **Scientific papers-ArXiv** (Cohan et al., 2018) from papers on arXiv, and **BigPatent** (Sharma et al., 2019) from patent applications.

The train splits of the four datasets are used separately to train our model. For Billsum, we used all 18,949 samples in the train split. For the other three datasets, the first 40,000 samples in the train split are used for training. For every original reference summary in the training sets, 3 negative samples (inferior summaries) are generated.

3.3 Baselines and Upperbounds

We compare our work with both reference-free and reference-based metrics. The recently developed SummaQA (Scialom et al., 2019), BLANC (Vasilyev et al., 2020), SUPERT (Gao et al., 2020b) and LS_Score (Wu et al., 2020) are our baselines because they are reference-free. [1]

Reference-based metrics serve as soft upper bounds because they are provided with extra human guides which are reference summaries. ROUGE (Lin, 2004), BLEU (Papineni et al., 2002), METEOR (Banerjee and Lavie, 2005), S^3 (Peyrard et al., 2017), MoverScore (Zhao et al., 2019), BertScore (recall) (Zhang et al., 2020) are included in this study.

Results for LS_Score (Wu et al., 2020) are only reported for Newsroom, which is copied from their paper, as we have not succeeded in reproducing their model using their code to test on other datasets[2]. Despite the difficulty, we implemented our own version of LS_Score.

3.4 Settings

For a fair comparison, we use the same pre-trained language model BERT used by the baselines. Specifically, we use the `bert-base-uncased` variant of the BERT model in HuggingFace Transformer's Pytorch implementation. An input sequence is padded to 512 tokens with [PAD] or truncated to 512 tokens using longer input truncate first strategy and then round robin trimmer. We fine tune the model on NVIDIA RTX 3090 for fixed 16,000 steps using the Adam optimizer with the learning rate of 1e-5 and the batch size of 7.

3.5 Results

We use the summary-level (Peyrard et al., 2017) meta evaluation strategy to report an approach's

[1] By "reference-free", we mean that a reference summary is not needed to judge a machine-generated summary.

[2] Several other researchers reported the same issue `https://github.com/whl97/LS-Score/issues`. We never heard back from the authors in Email and GitHub.

Table 1: Spearman's Correlation on TAC2010.

	Content	Fluency	Overall
	Our approach		
Trained w/CNNDM	**0.5865**	0.4311	**0.5531**
Trained w/Billsum	0.4586	**0.4324**	0.4518
Trained w/ArXiv	0.4727	0.4026	0.4437
Trained w/BigPatent	0.4184	0.3695	0.4007
	Reference-free Baselines		
BLANC-tune	0.4272	0.2943	0.3966
SummaQA-F1	0.3007	0.2431	0.2864
SummaQA-CFD	0.2905	0.1516	0.2620
SUPERT	0.4794	0.3241	0.4266
	Reference-based upper bounds		
R-1	0.5597	0.2570	0.5025
R-2	0.6448	0.3490	0.5894
R-L	0.5032	0.1772	0.4463
MoverScore	0.7213	0.3522	0.6453
BertScore	0.6769	0.3634	0.6162
BLEU	0.6018	0.3462	0.5636
METEOR	0.6682	0.3371	0.6184
S3_pyr	0.7257	0.3628	0.6562
S3_resp	0.7258	0.3578	0.6520

Table 2: Spearman's Correlation on Newsroom.

	COH	INF	FLU	REL
	Our approach			
Trained w/ CNNDM	0.6507	**0.7509**	0.6079	0.6645
Trained w/ Billsum	0.6665	0.7169	**0.6557**	0.6469
Trained w/ ArXiv	**0.6758**	0.7345	0.6408	**0.6657**
Trained w/ BigPatent	0.6729	0.7309	0.6498	0.6356
	Reference-free Baselines			
BLANC-tune	0.5862	0.6881	0.5310	0.6078
SummaQA-F1	0.4895	0.5690	0.4664	0.5163
SummaQA-CFD	0.4195	0.5449	0.3719	0.4405
SUPERT	0.6171	0.6929	0.5391	0.6046
LS_Score	0.6271	0.7008	0.5852	0.6381
	Reference-based Upper bounds			
R-1	0.2310	0.3231	0.2150	0.2775
R-2	0.0861	0.1534	0.1015	0.1336
R-L	0.2055	0.3005	0.2006	0.2629
MoverScore	0.1743	0.2186	0.1431	0.2163
BertScore	0.2705	0.3156	0.2390	0.2815
BLEU	-0.0556	-0.0782	-0.0422	-0.0071
METEOR	0.1740	0.2364	0.1690	0.2437
S3_pyr	0.1929	0.2680	0.1782	0.2450
S3_resp	0.1716	0.2519	0.1717	0.2226

Table 3: Spearman's Correlation on RealSumm†.

	On abstractive systems	On extractive systems
	Our approach	
Trained w/ CNNDM	**0.3842**	0.1143
Trained w/ Billsum	0.3083	0.0857
Trained w/ ArXiv	0.3204	0.0929
Trained w/ BigPatent	0.3163	**0.1152**
	Reference-free Baselines	
BLANC-tune	0.3067	0.1139
SummaQA-F1	0.2173	0.0837
SummaQA-CFD	0.2433	0.0494
SUPERT	0.2532	0.0748
	Reference-based Upper bounds	
R-1	0.6266	0.2182
R-2	0.5623	0.2206
R-L	0.6035	0.2140
MoverScore	0.4951	0.1899
BertScore	0.5682	0.1920
BLEU	0.3023	0.1639
METEOR	0.6270	0.2502
S3_pyr	0.6426	0.2369
S3_resp	0.6264	0.2369

† RealSumm has only one aspect which is content-focused.

average correlation with human ratings over summaries. Since our method is based on preference ranking, we report the Spearman's correlation (Tables 1, 2 and 3). The best scores in the reference-free class are **bold** while top 2 and 3 are underlined. Due to the page limit, we put the extra results of significance tests in the Appendix.

On TAC2010 (Table 1), our models beat all baselines on all aspects with only one exception. In particular, our model trained with CNNDM beats all baselines on all aspects. It even further outperforms ROUGE-1 and ROUGE-L.

On Newsroom (Table 2), our models beat all baselines on all aspects with only one exception. All reference-free approaches, including ours and baselines, outperform reference-based upper bounds. This counter-intuitive result is probably due to that a reference summary mostly has only one sentence in Newsroom.

On RealSumm (Table 3), results are reported separately for abstractive and extractive systems. Our models beat all baselines on abstractive systems. All approaches perform better for abstractive summarizers than for extractive ones. Bhandari et al. (2020) ascribe this to the low inter agreement among human annotators for the extractive group.

3.6 Discussion: Domain Impact

Because our approach is training based, in-domain models which are trained with CNNDM have advantages over cross-domain models. But the advantages are only for fact-based aspects (Content for TAC2010, INF and REL for Newsroom, the whole RealSumm), not for linguistic aspects.

Among cross-domain models, which are trained with Billsum, ArXiv, and BigPatent, no one is always the best on all test sets and on all aspects. Despite the domain difference, these models still beat the baselines in nearly all cases. Such cross-domain performances suggest that our approach is domain robust.

One potential use of our approach is to train a summary quality evaluation model for a domain with no or limited summarization data.

Table 4: Experiments on Model Architectures. Spearman's correlation.

Training Set	Model Arch.	TAC 2010			Newsroom				RealSumm	
		Modified	Linguistic	Overall	COH	INF	FLU	REL	Abstractive	Extractive
CNNDM	PrefScore	**0.5865**	**0.4311**	**0.5531**	**0.6507**	**0.7509**	**0.6079**	**0.6645**	**0.3842**	**0.1143**
	S_Score	0.4567	0.3034	0.4159	0.6204	0.7404	0.5809	0.6426	0.2785	0.1104
	L+S_Score	0.4077	0.3436	0.3784	0.6338	0.7234	0.6058	0.6374	0.3085	0.1070
BigPatent	PrefScore	**0.4184**	**0.3695**	**0.4007**	**0.6729**	**0.7309**	**0.6498**	**0.6356**	**0.3163**	**0.1152**
	S_Score	0.3499	0.2160	0.3155	0.5578	0.5992	0.5326	0.5374	0.2042	0.0958
	L+S_Score	0.3663	0.2984	0.3305	0.6605	0.7020	0.6138	0.6081	0.2589	0.1074
Billsum	PrefScore	**0.4586**	**0.4324**	**0.4518**	**0.6665**	**0.7169**	**0.6557**	**0.6469**	**0.3083**	0.0857
	S_Score	0.3689	0.3368	0.3483	0.4652	0.4280	0.4577	0.3996	0.2157	0.0568
	L+S_Score	0.3518	0.3475	0.3256	0.6199	0.6956	0.5844	0.5979	0.2790	**0.1052**
Arxiv	PrefScore	**0.4727**	**0.4026**	**0.4437**	**0.6758**	**0.7345**	**0.6408**	**0.6657**	**0.3204**	0.0929
	S_Score	0.3791	0.2511	0.3511	0.5972	0.5918	0.5804	0.5078	0.2331	0.0890
	L+S_Score	0.3792	0.2591	0.3405	0.6613	0.7330	0.5963	0.6382	0.3050	**0.1109**

3.7 Bi-Encoder vs. Cross-Encoder

We further conduct experiments to analyze the impact of the model architecture on performance. LS_Score (Wu et al., 2020) uses cosine similarity of the embeddings between a document and its summary as the semantic score (S_Score) which forms a Bi-Encoder architecture. And it computes a perplexity-like score based on the summary's embedding as linguistic score (L_Score), resulting in the final score as $0.01 * L\_Score + S\_Score$. In contrast, we jointly attend a document and a summary and produce the score after a linear layer which forms a Cross-Encoder architecture.

We implement the S_Score and L+S_Score[3] of our own version. The reason for our reimplementation is not only the reproducibility issues mentioned earlier but also that we want to do an apple-to-apple comparison by using the same loss function and the negative sampling strategy.

The results of the study are shown in Table 4. PrefScore outperforms both S_Score and L+S_Score on nearly all test sets and all aspects. It is common to use the cosine similarity in the embedding space as an indicator of semantic similarity. However, it fails to fully utilize the self-attention mechanism of the transformers. By jointly attending the document and the summary, our approach (Fig. 1) can better match information in the summary to that in the document. This could be one of the reasons that PrefScore outperforms S_Score and L+S_Score under the same setting.

4 Conclusion and Future Work

In this paper, we propose to evaluate summarization quality via preference learning and transitive negative sampling. The learned models outperform other reference-free based methods in in-domain experiments and are still competitive in cross-domain experiments.

There are some possible future study directions. The negative sampling methods used in this study are rough and simple. More careful inspection can be done to observe what kind of mistakes are likely made by summarizer models and design mutation methods accordingly. Moreover, our framework uses mean scores as a workaround for the multi-document scenario; it remains an open research problem to promote our work to optimize directly for multi-document summarization evaluation. Finally, we would like to extend our method for the evaluation of other NLG tasks.

[3]We denote our version as L+S_Score to discriminate from the original LS_Score.

Acknowledgements

This work is partially supported by National Science Foundation (NSF) grants No. MCB-1821828 and No. CNS-1817089. The authors would also like to thank reviewers who have given precious feedback on improving this work.

References

Satanjeev Banerjee and Alon Lavie. 2005. Meteor: An automatic metric for mt evaluation with improved correlation with human judgments. In *Proceedings of the acl workshop on intrinsic and extrinsic evaluation measures for machine translation and/or summarization*, pages 65–72.

Manik Bhandari, Pranav Narayan Gour, Atabak Ashfaq, Pengfei Liu, and Graham Neubig. 2020. Re-evaluating evaluation in text summarization. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*.

Ralph Allan Bradley and Milton E. Terry. 1952. Rank analysis of incomplete block designs: I. the method

of paired comparisons. *Biometrika*, 39(3/4):324–345.

Arman Cohan, Franck Dernoncourt, Doo Soon Kim, Trung Bui, Seokhwan Kim, Walter Chang, and Nazli Goharian. 2018. A discourse-aware attention model for abstractive summarization of long documents. *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 2 (Short Papers)*.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, Minneapolis, Minnesota. Association for Computational Linguistics.

Alexander R. Fabbri, Wojciech Kryściński, Bryan McCann, Caiming Xiong, Richard Socher, and Dragomir Radev. 2021. SummEval: Re-evaluating summarization evaluation. *Transactions of the Association for Computational Linguistics*, 9:391–409.

Yang Gao, Christian M Meyer, and Iryna Gurevych. 2020a. Preference-based interactive multi-document summarisation. *Information Retrieval Journal*, 23(6):555–585.

Yang Gao, Wei Zhao, and Steffen Eger. 2020b. SUPERT: Towards new frontiers in unsupervised evaluation metrics for multi-document summarization. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 1347–1354, Online. Association for Computational Linguistics.

Max Grusky, Mor Naaman, and Yoav Artzi. 2018. Newsroom: A dataset of 1.3 million summaries with diverse extractive strategies. *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*.

Anastassia Kornilova and Vladimir Eidelman. 2019. BillSum: A corpus for automatic summarization of US legislation. In *Proceedings of the 2nd Workshop on New Frontiers in Summarization*, pages 48–56, Hong Kong, China. Association for Computational Linguistics.

Chin-Yew Lin. 2004. ROUGE: A package for automatic evaluation of summaries. In *Text Summarization Branches Out*, pages 74–81, Barcelona, Spain. Association for Computational Linguistics.

NIST. 2010. TAC2010 guided summarization competition. `https://tac.nist.gov/2010/Summarization/Guided-Summ.2010.guidelines.html`. Accessed: 2021-08-16.

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. Bleu: a method for automatic evaluation of machine translation. In *Proceedings of the 40th annual meeting of the Association for Computational Linguistics*, pages 311–318.

Maxime Peyrard, Teresa Botschen, and Iryna Gurevych. 2017. Learning to score system summaries for better content selection evaluation. In *Proceedings of the Workshop on New Frontiers in Summarization*, pages 74–84, Copenhagen, Denmark. Association for Computational Linguistics.

Thomas Scialom, Sylvain Lamprier, Benjamin Piwowarski, and Jacopo Staiano. 2019. Answers unite! unsupervised metrics for reinforced summarization models. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 3246–3256, Hong Kong, China. Association for Computational Linguistics.

Abigail See, Peter J. Liu, and Christopher D. Manning. 2017. Get to the point: Summarization with pointer-generator networks. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1073–1083, Vancouver, Canada. Association for Computational Linguistics.

Ori Shapira, David Gabay, Yang Gao, Hadar Ronen, Ramakanth Pasunuru, Mohit Bansal, Yael Amsterdamer, and Ido Dagan. 2019. Crowdsourcing lightweight pyramids for manual summary evaluation. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 682–687, Minneapolis, Minnesota. Association for Computational Linguistics.

Eva Sharma, Chen Li, and Lu Wang. 2019. BIGPATENT: A large-scale dataset for abstractive and coherent summarization. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 2204–2213, Florence, Italy. Association for Computational Linguistics.

Oleg Vasilyev, Vedant Dharnidharka, and John Bohannon. 2020. Fill in the BLANC: Human-free quality estimation of document summaries. In *Proceedings of the First Workshop on Evaluation and Comparison of NLP Systems*, pages 11–20, Online. Association for Computational Linguistics.

Ramakrishna Vedantam, C Lawrence Zitnick, and Devi Parikh. 2015. Cider: Consensus-based image description evaluation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 4566–4575.

Jason Wei and Kai Zou. 2019. EDA: Easy data augmentation techniques for boosting performance on text classification tasks. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language*

Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP), pages 6382–6388, Hong Kong, China. Association for Computational Linguistics.

Hanlu Wu, Tengfei Ma, Lingfei Wu, Tariro Manyumwa, and Shouling Ji. 2020. Unsupervised reference-free summary quality evaluation via contrastive learning. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 3612–3621, Online. Association for Computational Linguistics.

Tianyi Zhang, Varsha Kishore, Felix Wu, Kilian Q. Weinberger, and Yoav Artzi. 2020. Bertscore: Evaluating text generation with bert. In *International Conference on Learning Representations*.

Wei Zhao, Maxime Peyrard, Fei Liu, Yang Gao, Christian M. Meyer, and Steffen Eger. 2019. MoverScore: Text generation evaluating with contextualized embeddings and earth mover distance. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 563–578, Hong Kong, China. Association for Computational Linguistics.

Markus Zopf. 2018. Estimating summary quality with pairwise preferences. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 1687–1696, New Orleans, Louisiana. Association for Computational Linguistics.

A Appendix

A.1 Evaluation Settings

We utilize the SummEval (Fabbri et al., 2021) evaluation toolkit to calculate scores for metrics whose scores are not reported by a test dataset. For all metrics, we use the batch evaluation API with default parameters provided by the package. The results of the SummEval dataset are not included in this study as SummEval and RealSumm are similar datasets whose documents are both sampled from CNN/DailyMail (See et al., 2017).

A.2 Significance Tests

We perform significance tests to see if the improvement of our method over the reference-free baselines is significant. Because applying a direct test on the summary-level evaluation results is difficult, we use a bootstrap-based method to sample the documents in the test sets 1000 times to compute the p-values.

Tables 5, 6 and 7 show the p-values of the hypothesis test that "Is the PrefScore trained using the training sets in the leftmost column significantly better than the baselines at the bottom?" Numbers smaller than the significant level of 0.05 are **bold**.

Our in-domain models trained using CNNDM are significantly better than the baselines. Meanwhile, the three cross-domain models, trained with Billsum, ArXiv, and BigPatent, are significantly better than SummaQA. They are also nearly significantly better than SUPERT. No significant results are observed on extractive systems from RealSumm. We believe this is due to the low inter agreement in the extractive group as described earlier (Bhandari et al., 2020).

Table 5: p-value of Significance Test on TAC2010 Dataset.

Training Set	Content				Fluency				Overall			
CNNDM	**0.00**	**0.00**	**0.00**	**0.00**	**0.00**	**0.00**	**0.00**	**0.00**	**0.00**	**0.00**	**0.00**	**0.00**
BillSum	0.17	-	**0.00**	**0.00**	**0.00**	**0.00**	**0.00**	**0.00**	0.05	0.20	**0.00**	**0.00**
BigPatent	-	-	**0.00**	**0.00**	**0.00**	0.07	**0.00**	**0.00**	0.44	-	**0.00**	**0.00**
ArXiv	0.09	-	**0.00**	**0.00**	**0.00**	**0.00**	**0.00**	**0.00**	0.09	0.30	**0.00**	**0.00**
	BLANC-tune	SUPERT	SummaQA-F1	SummaQA-CFD	BLANC-tune	SUPERT	SummaQA-F1	SummaQA-CFD	BLANC-tune	SUPERT	SummaQA-F1	SummaQA-CFD

Table 6: p-value of Significance Test on Newsroom Dataset.

Training Set	COH				INF				FLU				REL			
CNNDM	**0.02**	0.10	**0.00**	**0.00**	**0.01**	**0.01**	**0.00**	**0.00**	**0.00**	**0.00**	**0.00**	**0.00**	**0.02**	**0.01**	**0.00**	**0.00**
BillSum	**0.01**	0.08	**0.00**	**0.00**	0.19	0.23	**0.00**	**0.00**	**0.00**	**0.00**	**0.00**	**0.00**	0.10	0.10	**0.00**	**0.00**
BigPatent	**0.01**	0.06	**0.00**	**0.00**	0.07	0.11	**0.00**	**0.00**	**0.00**	**0.00**	**0.00**	**0.00**	0.17	0.19	**0.00**	**0.00**
ArXiv	**0.00**	0.07	**0.00**	**0.00**	0.09	0.12	**0.00**	**0.00**	**0.00**	**0.00**	**0.00**	**0.00**	**0.04**	**0.03**	**0.00**	**0.00**
	BLANC-tune	SUPERT	SummaQA-F1	SummaQA-CFD	BLANC-tune	SUPERT	SummaQA-F1	SummaQA-CFD	BLANC-tune	SUPERT	SummaQA-F1	SummaQA-CFD	BLANC-tune	SUPERT	SummaQA-F1	SummaQA-CFD

Table 7: p-value of Significance Test on RealSumm Dataset.

Training Set	Abstractive				Extractive			
CNNDM	**0.00**	**0.00**	**0.00**	**0.00**	0.49	0.08	0.21	0.07
BigPatent	0.38	**0.01**	**0.01**	0.05	0.51	0.08	0.22	0.08
BillSum	0.47	**0.02**	**0.01**	0.06	-	0.37	0.49	0.20
ArXiv	0.31	**0.01**	**0.01**	**0.03**	-	0.29	0.41	0.17
	BLANC-tune	SUPERT	SummaQA-F1	SummaQA-CFD	BLANC-tune	SUPERT	SummaQA-F1	SummaQA-CFD

Multi-Attribute Controlled Text Generation with Contrastive-Generator and External-Discriminator

Guisheng Liu[1], Yi Li[1]*, Yanqing Guo[1], Xiangyang Luo[2], Bo Wang[1]
[1]Dalian University of Technology
[2]Information Engineering University
lgs0000@mail.dlut.edu.cn, {liyi, guoyq}@dlut.edu.cn
xiangyangluo@126.com, bowang@dlut.edu.cn

Abstract

Though existing researches have achieved impressive results in controlled text generation, they focus mainly on single-attribute control. However, in applications like automatic comments, the topic and sentiment need to be controlled simultaneously. In this work, we propose a new framework for multi-attribute controlled text generation. To achieve this, we design a contrastive-generator that can effectively generate texts with more attributes. In order to increase the convergence of the text on the desired attributes, we adopt an external-discriminator to distinguish whether the generated text holds the desired attributes. Moreover, we propose top-n weighted decoding to further improve the relevance of texts to attributes. Automated evaluations and human evaluations show that our framework achieves remarkable controllability in multi-attribute generation while keeping the text fluent and diverse. It also yields promising performance on zero-shot generation.

movie positive: In the end Hitchcock brings one of those rare Hollywood thrillers that I have been craving for a long time: the suspense thriller. It is one of my favorite genres and movies because you can always be sure that something exciting will happen in the ending (and sometimes it does!). I think it is also a classic Hitchcock masterpiece.
automobile negative: In the past sixty years, GM cars have gotten worse, because GM engineers were worse. Now they are just as bad as Ford's. But GM still makes good cars. I'm sorry but car makers need to go back to making things people want, like lawnmowers.It's a big step backwards.

Table 1: Examples of the text generated by our framework. The boldfaced words indicate the desired topic and sentiment. The underlined words are the prompts. The words related to the topic are highlighted in blue, and the words related to the sentiment are highlighted in red.

1 Introduction

Large pre-trained language models (LM) have achieved impressive performance in natural language generation tasks (Radford et al., 2018; Yang et al., 2019; Radford et al., 2019; Lewis et al., 2020). However, they suffer from the limitation of lacking controllability in practical applications, for their generated texts may have toxic meanings or undesired topics. Hence how to control the text generation with desired attributes as topic, sentiment, style, etc., has been further studied.

To effectively control the desired attribute of the generated text, class-conditional language models (CC-LM) such as Conditional Transformer Language (CTRL) model (Keskar et al., 2019) generate texts conditioned on control codes. But the controlled ability are restricted since the control codes are only used at the beginning of the generation process. The Plug and Play Language Models (Dathathri et al., 2020) directly update the gradient of a classifier to generate the conditioned text without retraining or finetuning the language model. While being flexible, this kind of method is computationally expensive and leads to less fluent texts. Recently, (Yu et al., 2021) introduces an alignment function to the language model so that it can generate texts with target attributes. Future Discriminator for Generation (Yang and Klein, 2021) trains a classifier to predict the probability of the desired attribute. However, all these methods are aimed at single-attribute control, making them insufficient to deal with application scenarios that need multi-attribute control. Taking the automatic comment system as an instance, it requires to control the

*Corresponding author

Proceedings of the 29th International Conference on Computational Linguistics, pages 5904–5913
October 12–17, 2022.

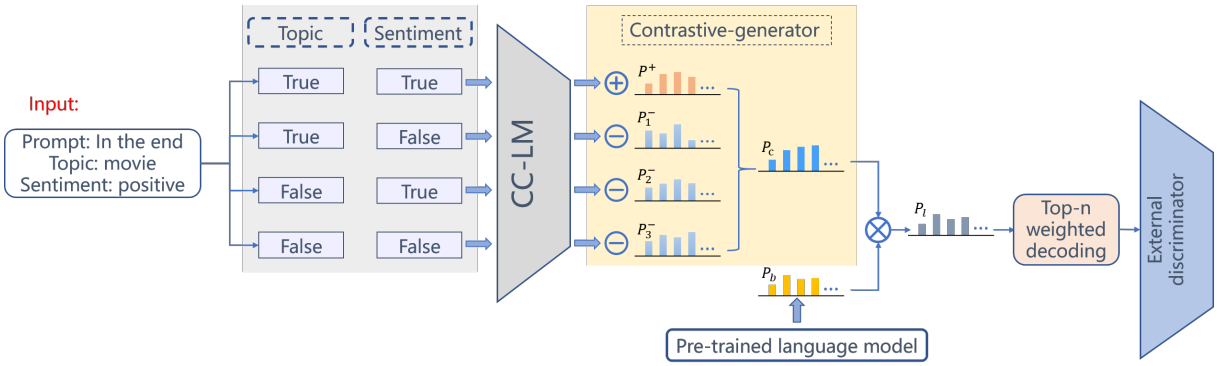

Figure 1: Illustration of our framework. Three negative samples and one positive sample are used to calculate the classification probability in CC-LM. The probability multiplies with unconditioned probability that is generated by a pre-trained language model. Through our proposed top-n weighted decoding, the external-discriminator (on the right) is used to discriminate the desired attribute of the text.

generated text with topic as well as sentiment to encourage user engagement and interactions.

In light of the problem, we propose a multi-attribute controlled framework that can effectively control topic and sentiment of the text at the same time. Although there are previous researches (Dathathri et al., 2020; Yu et al., 2021; Goswamy et al., 2020) about multi-attribute controlled text generation, they only conceptually raise the task but focus mainly on single-attribute control. The challenges of multi-attribute control lie in 1) the fusion of attributes and 2) the increased categories to be generated. To generate texts with desired topic and sentiment, we design a contrastive-generator with three negative samples in contrast to one positive sample. Since the generator may generate texts with different categories, we train an external-discriminator to increase the convergence of texts on desired attributes. In the decoding phase, a top-n weighted decoding is proposed to improve the ability of controlled text generation.

Table 1 shows the texts generated by our framework, where the texts achieve desired attributes control without losing its fluency. Zero-shot generation, which is a more challenging task, aims at generating unseen text from the seen text. Due to the top-n weighted decoding and contrastive-generator trained with external-discriminator, our framework can generate texts with other desired attributes besides the training attributes, accomplishing zero-shot generation.

We summarize the contributions of this work as follows:

- Different from existing works, we aim at multi-attribute controlled text generation, which is not only more challenging but also more practical in real-life applications as the automatic comment system.
- We propose a contrastive-generator trained with an external-discriminator to effectively generate texts with desired attributes. A top-n weighted decoding is also designed to further improve the relevance between the texts and the desired attributes.
- We conduct extensive experiments to show that our method can generate texts with desired sentiment and topic without sacrificing the linguistic quality. In addition, our framework can be generalized to new control codes and achieve promising performance on zero-shot generation.

2 Related Work

Given a control code a, the purpose of controlled text generation is to generate text x by calculating the probability of $p(x|a)$. There are mainly two categories: the first retrains language model with control codes, while the second changes the weight of the specific words for controlled text generation.

Models trained or fine-tuned (Keskar et al., 2019; Xu et al., 2020; Fang et al., 2021) on a large number of conditioned codes can achieve remarkable effectiveness for controlled text generation. However, the large training data and the computation cost are the heavy burdens. Methods with a smaller LM (Krause et al., 2021; Yang and Klein, 2021; Liu et al., 2021) to guide generation from large LM can generate text for sentiment control or topic control. GeDi (Krause et al., 2021) uses Bayes rule

to compute classification likelihoods of tokens, and its generative discriminator performs well on controlled text generation. Similarly, a future discriminator (Yang and Klein, 2021) is used to determine whether the desired attribute will appear in the future text. Expert LM and anti-expert LM (Liu et al., 2021) are utilized to reweight the predictions of the large LM. Considering the promising results of the discriminator, a semantic discriminator (Betti et al., 2020) is used to discriminate the coherence with external conditioning. To get a better attribute representation, (Yu et al., 2021; Xie et al., 2022) introduces a new alignment function for sentiment control. A sentence-level emotion classifier (Zhang and Wang, 2021) is adopted to generate comments with the target emotion. The methods of retraining language models tend to produce fluent texts, but they need large labeled datasets for training.

Weighted decoding is also a useful method to control text generation with language models. These methods (Hu et al., 2019; Pascual et al., 2020, 2021), which increase the probability of tokens that are similar to the target keyword or topic, control the desired attributes of texts flexibly. Metropolis-Hastings sampling is used to generate texts with more keywords (Miao et al., 2019). But since these methods do not update LM, it will decrease the fluency of the generated texts. The following works (Dathathri et al., 2020; Sha, 2020; Duan et al., 2020; Lin and Riedl, 2021; Madotto et al., 2020) conduct constrained generation under gradient guidance. To control the sentiment better, the method in (Goswamy et al., 2020) focuses on controlling more emotion categories and emotion intensity. However, gradient-based methods may lead to more computation to calculate the word's probability. Although modifying the distribution of language models is a flexible way for controlled text generation, it will sacrifice the texts quality.

Our method draws on the above two thoughts. We design a contrastive-generator and an external-discriminator by retraining the language model to control text generation while keeping the text fluency. And we also propose top-n weighted decoding to increase the correlation of attributes.

3 Methodology

As is illustrated in Figure 1, our method is composed of three main modules. We use CTRL as CC-LM to generate one positive sample and three negative samples for the contrastive-generator. Then the contrastive-generator (subsection 3.1) outputs classification probabilities that guide the generation of the pre-trained LM. Since there are multiple attributes to be considered in multi-attribute controlled text generation, we use an external-discriminator (subsection 3.2) to estimate whether the generated text achieves the target attributes. To ensure the text fluency, the top-n weighted decoding (subsection 3.3) recalculates the probabilities of the n most probable words. The details of these modules are described in the following subsections.

3.1 Contrastive-Generator

The fusion of different attributes is one of the challenges for multi-attribute controlled text generation. To deal with the issue, we propose a contrastive-generator to generate texts with the desired topic and sentiment.

Given the desired attribute a_t and a_s, our task is to learn the probability distribution $P(x_{1:N} \mid a_t, a_s)$ where $x_{1:N}$ denotes a complete text $(x_1, \ldots, x_N)$. In particular, we use a_t to present the desired topic control code while a_s for the desired sentiment control code. The CC-LM generates a completed text $x_{1:N}$ by the following equation:

$$P(x_{1:N} \mid a_t, a_s) = \prod_{i=1}^{N} P(x_i \mid x_{<i}, a_t, a_s). \quad (1)$$

Then we refer $\mathcal{L}_g$ as the conditioned language model loss:

$$\mathcal{L}_g = -\sum_{i=1}^{N} \log P(x_i \mid x_{<i}, a_t, a_s). \quad (2)$$

The contrastive-generator aims to learn the effective representation by pulling the positive samples close and pushing apart negative samples. We use $\bar{a_t}$ as the undesired topic control code and use $\bar{a_s}$ as the undesired sentiment control code. A class-conditioned language model is adopted to get a positive sample $P(x_{1:N} \mid a_t, a_s)$ and three negative samples $P(x_{1:N} \mid \bar{a})$ where $\bar{a} \in \{(\bar{a_t}, a_s), (a_t, \bar{a_s})(\bar{a_t}, \bar{a_s})\}$. Obeying Bayes rule, we compute $P(a_t, a_s \mid x_{1:N})$ as the classification probability that guides the generation of the

pre-trained LM:

$$P\left(a_t, a_s \mid x_{1:N}\right) = \frac{P\left(a_t, a_s\right) P\left(x_{1:N} \mid a_t, a_s\right)}{\sum_a P\left(a\right) P\left(x_{1:N} \mid a\right)} = \frac{P\left(a_t, a_s\right) \prod_{i=1}^{N} P\left(x_i \mid x_{<i}, a_t, a_s\right)}{\sum_a P\left(a\right) \prod_{i=1}^{N} P\left(x_i \mid x_{<i}, a\right)} \quad (3)$$

where $a \in \{(a_t, a_s), (\bar{a_t}, a_s), (a_t, \bar{a_s}) (\bar{a_t}, \bar{a_s})\}$. The loss function of the contrastive-generator is

$$\mathcal{L}_c = -\log P\left(a_t, a_s \mid x_{1:N}\right). \quad (4)$$

Then we use $P\left(a_t, a_s \mid x_{1:N}\right)$ to guide the generation of the large pre-trained LM. For the generation on attribute a_t and a_s, we have:

$$P\left(x_N \mid x_{<N}, a_t, a_s\right) = \frac{P\left(x_N, a_t, a_s \mid x_{<N}\right)}{P\left(a_t, a_s \mid x_{<N}\right)} = \frac{P\left(x_N \mid x_{<N}\right) P\left(a_t, a_s \mid x_{1:N}\right)}{P\left(a_t, a_s \mid x_{<N}\right)}. \quad (5)$$

Since a_t and a_s are given and the sentence $x_{1:N-1}$ has been calculated, we can draw the conclusion that $P\left(a_t, a_s \mid x_{<N}\right)$ is a constant. So we simplify the Equation 5 by the following:

$$P\left(x_N \mid x_{<N}, a_t, a_s\right) \propto P\left(x_N \mid x_{<N}\right) P\left(a_t, a_s \mid x_{1:N}\right)^{\alpha} \quad (6)$$

where α is a a hyper-parameter that controls the weight of the desired attribute. On the right, the first part is essentially a language model. The second part can be calculated by Equation 3, which is essentially the desired attribute probability of the text calculated by the contrastive-generator.

3.2 External-Discriminator

Since our work aims at multi-attribute controlled text generation, it requires to take more than one category into consideration. We propose an external-discriminator to distinguish whether the text holds the desired attributes, which further increases the convergence of the text on the desired attributes.

The external-discriminator transforms its input into an embedding matrix and outputs a probability. To alleviate the computation burden, we use multi-layer bi-directional GRU as external-discriminator. Here we implement D_ϕ as the classifier to distinguish between the texts with the desired attributes and with the undesired attributes. The external-discriminator loss can be defined as:

$$\mathcal{L}_{external} = -\{(a_t, a_s) \log D_\phi\left(a_t, a_s \mid x_{1:N}\right) + (1 - (a_t, a_s)) \log\left(1 - D_\phi\left(a_t, a_s \mid x_{1:N}\right)\right)\} \quad (7)$$

where $D_\phi\left(a_t, a_s \mid x_{1:N}\right)$ is the probability predicted by D_ϕ indicating that the text $x_{1:N}$ belongs to the desired topic a_t and the desired sentiment a_s. In order to achieve a better performance on the desired attribute control, the external-discriminator tries to guide the sentence towards the desired attributes with decreasing the external-discriminator loss.

In the end, the overall loss function for our framework is a weighted sum of three loss terms:

$$\mathcal{L}_{total} = \frac{\lambda_g}{\tau}\mathcal{L}_g + \frac{\lambda_c}{\tau}\mathcal{L}_c + \frac{\lambda_e}{\tau}\mathcal{L}_{external} \quad (8)$$

where λ_* are the hyper-parameters that reflect the strength of each loss and τ is calculated by the following equation:

$$\tau = \lambda_g + \lambda_c + \lambda_e. \quad (9)$$

3.3 Top-n Weighted Decoding

Recent researches have made impressive progress in weighted decoding (Fan et al., 2018; Holtzman et al., 2019; Pascual et al., 2020, 2021). In the decoding time, we propose a top-n weighted decoding to increase the topic relevance while generating fluent texts. Through LM, we get the probabilities of all lexical words. Different from previous methods, we modify the probabilities of the n most probable choices, instead of changing each of the words.

Utilizing the vectors of words, we calculate the cosine similarity between the topic words and the n most probable candidate words. And we use the max function to increase the weight of the related words while keeping the weight of unrelated words as it is. The reason is to increase the fluency of the texts as much as possible. Let $\nu\left(\omega_{topic}\right) \in \mathbb{R}^d$ denote the topic vector, and $\nu\left(\omega'_{top-n}\right) \in \mathbb{R}^{n \times d}$ be the n vectors of n most probable words, where d is the dimension of the vector. The modified probability l_{top-n} is calculated as:

$$l_{top-n} = l'_{top-n} + \gamma \cdot \max\left(0, \cos\left(\nu\left(\omega_{topic}\right), \nu\left(\omega'_{top-n}\right)\right)\right) \quad (10)$$

Method	Diversity			Fluency	Sentiment	Topic	
	Dist-1 ↑	Dist-2 ↑	Dist-3 ↑	Perplexity ↓	Prob. ↑	Prob. ↑	Acc. ↑
GPT-2 Medium	0.20	0.64	0.86	**30.75**	0.47	0.09	0.29
GeDi-sentiment	0.23	**0.71**	**0.91**	64.10	**0.90**	\	\
DAPT-sentiment	0.14	0.50	0.74	33.74	0.75	\	\
PPLM-sentiment	0.17	0.58	0.86	44.02	0.70	\	\
DEXPERTS	0.15	0.45	0.62	36.66	0.89	\	\
GeDi-topic	0.19	0.58	0.80	59.48	\	0.46	0.85
DAPT-topic	0.14	0.50	0.72	54.68	\	0.55	0.90
PPLM-topic	0.18	0.59	0.86	39.02	\	0.33	0.76
Plug-and-Blend	**0.29**	0.67	0.76	74.99	\	0.39	0.80
PPLM	0.17	0.57	0.81	80.67	0.66	0.47	0.87
CATG	0.18	0.54	0.72	51.74	0.66	0.26	0.51
Ours	0.17	0.58	0.83	32.58	**0.90**	**0.60**	**0.92**

Table 2: The result of multi-attribute controlled text generation. We use boldface to indicate the best performance. For methods of GeDi, DAPT and PPLM-sentiment(topic), we train and evaluate its topic model and sentiment model respectively.

where l'_{top-n} refers to the original probabilities and γ is a hyper-parameter that controls the weight of the modification. As $\gamma \to 0$, the effect of the weighted decoding decreases. In our experiments, we find that the value of γ works well in the range 2 - 5.

Furthermore, since the top-n weighted decoding merely adjusts n probabilities, it not only keeps the generated texts fluent but also decreases the computation cost while controlling the desired attributes.

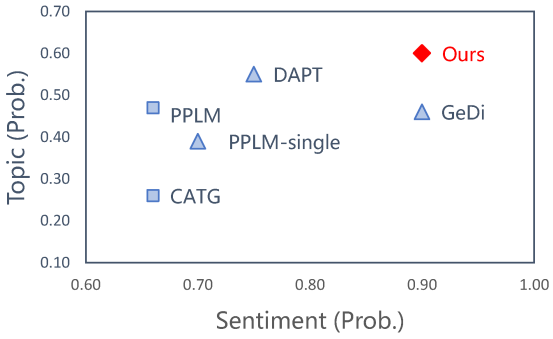

Figure 2: The analysis of sentiment and topic control. We take the sentiment mean probability as the abscissa and topic mean probability as the ordinate. The top right corner lies the best result. GeDi, DAPT and PPLM-single (showed with triangle) generate topic and sentiment texts separately. PPLM and CATG (showed with square) generate texts with topic and sentiment attributes. Our method (showed with red rhombus) surpasses both single-attribute control and multi-attribute control models.

4 Experiment

We conduct experiments on the task of multi-attribute controlled text generation (subsection 4.1) and zero-shot generation (subsection 4.2) to evaluate the performance of our framework. The ablation experiments (subsection 4.3) are also presented to analyze the importance of each module.

4.1 Multi-Attribute Controlled Text Generation

4.1.1 Evaluation

To avoid the influence of the pre-trained language model, we use GPT-2 Medium (Radford et al., 2019) as the basic language model both in our method and in the baselines. In order to evaluate the topic and the sentiment control ability of our method, we collect 500 neutral prompts that are irrelevant to the trained topics.

We use IMDb (Maas et al., 2011), OpeNER (Agerri et al., 2013) and SenTube(Uryupina et al., 2014) as the datasets for multi-attribute controlled generation. The IMDb dataset is about movie reviews with positive and negative sentiments. The OpeNER dataset is about hotel reviews that have the same two sentiments as IMDb. The SenTube dataset contains reviews in tablet and automobile domains, and for each domain we take positive and negative texts. In summary, there are four topics (movie, hotel, tablet, automobile) and two sentiments (positive and negative) during training.

Method	Fluency	Sentiment	Topic
GeDi-sentiment	3.46	4.06	\
DAPT-sentiment	4.05	3.26	\
PPLM-sentiment	3.56	3.34	\
DEXPERTS	3.88	3.94	\
GeDi-topic	3.53	\	4.02
DAPT-topic	3.67	\	3.96
PPLM-topic	3.56	\	3.20
Plug-and-Blend	3.40	\	3.66
PPLM	2.98	2.84	3.84
CATG	3.34	3.12	2.66
Ours	**4.10**	**4.14**	**4.26**

Table 3: Human evaluation of fluency and texts relevancy on the desired sentiment and topic.

animal positive: In the past sixty years, animal welfare has increased in many countries around the world. It is an ongoing process, and we are all part of it. We can all be a part of it! And that's what I'm doing with my blog! I want to share with you my thoughts on animals, and help you make decisions about how to treat your own household animals as well as other animals.
school negative: In a shocking finding that raises serious questions about school safety and security, researchers found that at least seven schools have experienced incidents involving armed guards or police. A report from SafeSchools.org says that between 2007 and 2014 there have been five incidents involving armed guards or police at more than 20 schools across the United States, with three resulting in fatalities. In all but two cases, it says, there was no immediate threat to students or staff.

Table 4: Zero-shot generation by our framework. Bold-faced words indicate the desired topic and sentiment. We use underlined words to show the prompts. Words related to the topic are highlighted in blue, and words related to the sentiment are highlighted in red.

In this paper, we adopt automatic evaluation as well as human evaluation to appraise the generated texts. For the automatic evaluation, we take the following four metrics into account.

- *diversity*. Diversity (Li et al., 2016) is a metric that evaluates the the diversity of the generated sentences. We report Dist-1, Dist-2 and Dist-3 by measuring the diversity of unigrams, bigrams and trigrams in the generation. A higher value indicates better diversity.

- *perplexity*. Perplexity is an automated measure of sentence fluency, lower being better. We utilize GPT-2 XL (Radford et al., 2019) to compute the perplexity of the generated text, because we use GPT-2 Medium as the pre-trained language model.

- *sentiment*. We evaluate the generations by HuggingFace's sentiment analysis classifier. The classifier achieves the accuracy of over 98% on the test data. And we obtain the mean probability from the classifier.

- *topic*. We train a topic classifier to determine whether the generated text has the desired topic attribute. The accuracy of the topic classifier is above 98%. We also report the topic accuracy and the mean probability that the text has the desired topic attribute.

4.1.2 Baseline

We compare our framework with the competitive baselines:

GPT-2 Medium: (Radford et al., 2019) To explore the influence of the pre-trained language model, we generate sentences by GPT-2 Medium as an original baseline.

PPLM: (Dathathri et al., 2020) PPLM uses gradient update to guide GPT-2 model. We retrain its discriminator to control the sentiment and topic of the text. And we evaluate its performance on single-attribute control and multi-attribute control respectively.

GeDi: (Krause et al., 2021) GeDi uses small LM as the generative discriminator to guide the generation of large LM. We separately train its topic model and sentiment model on our dataset with only topic labels or sentiment labels.

DEXPERTS: (Liu et al., 2021) DEXPERTS reweights the predictions of LM by the expert and anti-expert model. We use DEXPERTS to control the sentiment of texts for comparison.

DAPT: (Gururangan et al., 2020) DAPT shows the importance of pretraining the model towards a specific task. We use the method to generate sentiment text and topic text via training on our dataset.

CATG: (Goswamy et al., 2020) CATG controls the sentiment of sentences with a knob to influence

Desired topic	Method	Diversity			Fluency	Sentiment	Topic	
		Dist-1 ↑	Dist-2 ↑	Dist-3 ↑	Ppl. ↓	Prob. ↑	Prob. ↑	Acc. ↑
School	GeDi-topic	0.17	0.63	0.89	67.68	\	0.20	0.45
	Plug-and-Blend	0.27	0.63	0.76	67.57	\	0.18	0.50
	Ours (positive)	0.18	0.63	0.86	35.45	0.85	**0.49**	0.84
	Ours (negative)	0.16	0.60	0.86	25.23	**0.87**	0.47	0.83
Animal	GeDi-topic	0.19	**0.65**	**0.90**	67.15	\	0.29	0.73
	Plug-and-Blend	**0.28**	0.64	0.76	57.59	\	0.24	0.68
	Ours (positive)	0.17	0.62	0.87	30.09	0.78	0.38	0.88
	Ours (negative)	0.16	0.62	0.87	**23.07**	0.84	0.39	**0.92**

Table 5: The result of zero-shot generation. We use boldface to indicate the best performance.

the sentiment intensity. All the hyper-parameters are set following its original paper.

Plug-and-Blend: (Lin and Riedl, 2021) Plug-and-Blend allows multiple topic codes to generate texts. We use the model as a topic control baseline.

For all baseline methods, we generate 500 sentences for each category of topics and sentiments by our collected prompts. All the experiments are run on NVIDIA Tesla V100 GPUs. And in our framework, we configure the top-n weighted decoding with $\gamma = 4$ and $n = 50$. The values of the hyper-parameters λ_*: $\lambda_g = 0.8$, $\lambda_c = 0.2$, $\lambda_e = 1.0$.

4.1.3 Automatic Evaluation

The results are shown in Table 2. Our framework outperforms all baselines on sentiment metrics and topic metrics. It demonstrates the effectiveness of our framework on simultaneous control of both sentiment and topic. Concretely, our contrastive-generator can generate texts with multi-attribute excellently. The external-discriminator, which distinguishes the text with the desired sentiment and topic, increases the convergence of the text on the desired attributes. As is shown in Figure 2, though DAPT and GeDi train its topic and sentiment models separately, our method produces comparable or even better results in merely one model. Our framework obtains the highest mean probability and mean accuracy on the topic metrics, indicating that the top-n weighted decoding fertilizes the relevance between the texts and the desired topic effectively.

Meanwhile, texts generated by our framework acquire qualified fluency. This is because the contrastive-generator guides the basic LM without losing fluency. On the other hand, the top-n weighted decoding only modifies n words with high probability, which guarantees the maximum consistency with LM. Sentiment results on GPT-2 Medium show that our collected prompts are nearly neutral prompts that have little effects on the sentiment control. Similarly, the low topic metrics of it verify that our collected prompts are unrelated with topic. The reason why our framework is not outstanding on the diversity metrics is that our generation is under the control of sentiment and topic. And the more control leads to more limitations for generation that would hinder the diversity inherently.

Table 1 shows the texts generated by our framework. We can observe that the generated texts focus on the desired topic closely while keeping the desired sentiment. Since our training dataset are comments, our generated texts are more likely to comment on something.

4.1.4 Human Evaluation

We also conduct a human evaluation to compare the performance of baselines and our framework comprehensively. We randomly selected 20 samples from the generated texts for each method. All samples are randomly shuffled and the generation methods are completely hidden. We ask 50 annotators to evaluate the texts by the following criteria: *fluency*, *sentiment* and *topic*. Every criterion is evaluated on a scale of 1-5, where a higher score indicates better quality.

Table 3 presents the average scores of human evaluation, from which we can draw similar conclusions with the automatic evaluation. Our framework outperforms the baselines in topic and sentiment controlling while holding better fluency. We observe that GeDi has good performance on attribute control, but it can not control sentiment and topic in one sentence. Comparing with PPLM which directly updates the gradients of the pre-

Method	Diversity			Fluency	Sentiment	Topic	
	Dist-1 ↑	Dist-2 ↑	Dist-3 ↑	Perplexity ↓	Prob. ↑	Prob. ↑	Acc. ↑
Full framework	0.17	0.58	0.83	32.58	0.90	0.60	0.92
Without W	0.18	0.62	0.86	35.29	0.86	0.47	0.80
Without D	0.16	0.59	0.85	30.19	0.76	0.55	0.90
Without W,D	0.17	0.61	0.86	28.46	0.77	0.44	0.79

Table 6: Automatic evaluations of ablation study. "Without W" means that we not use top-n weighted decoding. "Without D" means that we not use external-discriminator.

trained LM, our framework has better performance on the fluency. This is because the top-n weighted decoding only changes the probabilities of the most likely n words, avoiding decreasing the text fluency significantly.

4.2 Zero-Shot Generation

We train four topics (movie, hotel, tablet, automobile) with two sentiments (positive, negative). Topics such as "school" or "animal" not appearing in the training dataset, our framework is able to generate texts with these unseen attributes. We show two examples in Table 4. Although we do not train on the two topics, our framework can effectively generate texts with the desired topic and sentiment.

We evaluate the zero-shot generation with the same metrics as the multi-attribute control. In addition, we train a topic classifier on DBPedia dataset (Zhang et al., 2015) to determine whether the generation has the desired topic attribute. The classifier achieves the accuracy of 99% on the test data.

We run experiments with zero-shot generation on the topic of "school" and "animal". For each topic, our framework generates 500 sentences with the collected 500 prompts. And we compare competitive models with Plug-and-Blend (Lin and Riedl, 2021) and GeDi (Krause et al., 2021).

The results are listed in Table 5. Our method gains better topic controlling metrics than the others while keeping the desired sentiment. It implies that the contrastive-generator generates texts effectively with unseen attributes due to its training with the external-discriminator. In addition, the top-n weighted decoding improves the relevance of the texts to the desired topic without losing the text fluency. We observe that our framework shows mediocre performance on diversity, because our framework generates texts under the control of sentiment and topic at the same time, which brings barrier to generating diverse texts. Considering that the topics are not trained, our proposed framework generalizes the pre-trained LM to generate texts with unseen categories.

4.3 Ablation

To understand the importance of each module in our framework, we perform an ablation study by training the following ablated versions: without external-discriminator, without top-n weighted decoding, without external-discriminator and top-n weighted decoding.

Table 6 presents the automatic evaluation of the ablation study. Results show that all three ablation operations will result in the decrease in attribute control performance. But since our contrastive-generator can effectively guide conditional generation by the large LM, the results about topic and sentiment still yields high values. From the result of without external-discriminator version, we observe that the topic metrics obtain relatively good results. The reason is that the top-n weighted decoding significantly improves the topic coherence. Compared to the removal of top-n weighted decoding, the full framework shows higher results of the topic and sentiment. Because the signals of the discriminator in training not only evaluate the desired attribute, but also enhance the relation between attributes and texts. In detail, from the result of without top-n weighted decoding and external-discriminator version, we notice that the average probabilities of sentiment and topic are reduced by 0.13 and 0.16 respectively. This indicates that both external-discriminator and top-n weighted decoding can effectively improve the control of sentiment and topic.

5 Conclusion and Future Work

In this paper, we propose an effective framework for multi-attribute controlled text generation. Experiments and further analysis demonstrate

that the contrastive-generator and the external-discriminator perform essentially on multi-attribute generation and zero-shot generation. And the controllability of the desired attributes is further improved by our proposed top-n weighted decoding without losing the quality of texts. We also conduct the ablation experiment, showing the importance of each module. In addition to the topic and sentiment control, our framework is capable of applying to other multi-attribute control. In the future, we will generalize our model to generate texts with other attributes, e.g. writing styles and toxicity, making the generation more safer and more qualified.

6 Ethical Consideration

Since the proposed framework can be used to generate texts with more desired attributes, its generation is more like human-generated. It would benefit language generation applications on downstream tasks, such as automatic comments and chatting robots. Although automatic comments can encourage user interactions, it may mislead public opinions when it is used for malicious purposes. Moreover, we observe that the work may generate toxic texts when a negative attribute is given. Hence in the future, we will investigate how to detect toxic texts and replace the offensive words without changing the meaning of the text.

7 Acknowledgements

This work is supported in part by the National Natural Science Foundation of China (No. 62106037, No. 62076052), in part by the Major Program of the National Social Science Foundation of China (No.19ZDA127), in part by the Fundamental Research Funds for the Central Universities (No. DUT22YG205, No. DUT20RC(3)088, No. DUT20TD110), and in part by the Open Project Program of the National Laboratory of Pattern Recognition (NLPR) (No. 202100032).

References

Rodrigo Agerri, Montse Cuadros, Sean Gaines, and German Rigau. 2013. Opener: Open polarity enhanced named entity recognition. *Procesamiento del Lenguaje Natural*, pages 215–218.

Federico Betti, Giorgia Ramponi, and Massimo Piccardi. 2020. Controlled text generation with adversarial learning. In *Proceedings of the 13th International Conference on Natural Language Generation*, pages 29–34.

Sumanth Dathathri, Andrea Madotto, Janice Lan, Jane Hung, Eric Frank, Piero Molino, Jason Yosinski, and Rosanne Liu. 2020. Plug and play language models: A simple approach to controlled text generation. In *International Conference on Learning Representations*.

Yuguang Duan, Canwen Xu, Jiaxin Pei, Jialong Han, and Chenliang Li. 2020. Pre-train and plug-in: Flexible conditional text generation with variational auto-encoders. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 253–262.

Angela Fan, Mike Lewis, and Yann Dauphin. 2018. Hierarchical neural story generation. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 889–898.

Le Fang, Tao Zeng, Chaochun Liu, Liefeng Bo, Wen Dong, and Changyou Chen. 2021. Transformer-based conditional variational autoencoder for controllable story generation. *arXiv preprint arXiv:2101.00828*.

Tushar Goswamy, Ishika Singh, Ahsan Barkati, and Ashutosh Modi. 2020. Adapting a language model for controlled affective text generation. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 2787–2801.

Suchin Gururangan, Ana Marasović, Swabha Swayamdipta, Kyle Lo, Iz Beltagy, Doug Downey, and Noah A Smith. 2020. Don't stop pretraining: Adapt language models to domains and tasks. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 8342–8360.

Ari Holtzman, Jan Buys, Li Du, Maxwell Forbes, and Yejin Choi. 2019. The curious case of neural text degeneration. In *International Conference on Learning Representations*.

J Edward Hu, Huda Khayrallah, Ryan Culkin, Patrick Xia, Tongfei Chen, Matt Post, and Benjamin Van Durme. 2019. Improved lexically constrained decoding for translation and monolingual rewriting. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 839–850.

Nitish Shirish Keskar, Bryan McCann, Lav R Varshney, Caiming Xiong, and Richard Socher. 2019. Ctrl: A conditional transformer language model for controllable generation. *arXiv preprint arXiv:1909.05858*.

Ben Krause, Akhilesh Deepak Gotmare, Bryan McCann, Nitish Shirish Keskar, Shafiq Joty, Richard Socher, and Nazneen Fatema Rajani. 2021. Gedi: Generative discriminator guided sequence generation. In *Findings of the Association for Computational Linguistics: EMNLP 2021*, pages 4929–4952.

Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Veselin Stoyanov, and Luke Zettlemoyer. 2020. Bart: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7871–7880.

Jiwei Li, Michel Galley, Chris Brockett, Jianfeng Gao, and Bill Dolan. 2016. A diversity-promoting objective function for neural conversation models. In *Proceedings of NAACL-HLT*, pages 110–119.

Zhiyu Lin and Mark O Riedl. 2021. Plug-and-blend: a framework for plug-and-play controllable story generation with sketches. In *Proceedings of the AAAI Conference on Artificial Intelligence and Interactive Digital Entertainment*, volume 17, pages 58–65.

Alisa Liu, Maarten Sap, Ximing Lu, Swabha Swayamdipta, Chandra Bhagavatula, Noah A Smith, and Yejin Choi. 2021. Dexperts: Decoding-time controlled text generation with experts and anti-experts. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*.

Andrew Maas, Raymond E Daly, Peter T Pham, Dan Huang, Andrew Y Ng, and Christopher Potts. 2011. Learning word vectors for sentiment analysis. In *Proceedings of the 49th annual meeting of the association for computational linguistics: Human language technologies*, pages 142–150.

Andrea Madotto, Etsuko Ishii, Zhaojiang Lin, Sumanth Dathathri, and Pascale Fung. 2020. Plug-and-play conversational models. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 2422–2433.

Ning Miao, Hao Zhou, Lili Mou, Rui Yan, and Lei Li. 2019. Cgmh: Constrained sentence generation by metropolis-hastings sampling. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, pages 6834–6842.

Damian Pascual, Beni Egressy, Florian Bolli, and Roger Wattenhofer. 2020. Directed beam search: Plug-and-play lexically constrained language generation. *arXiv preprint arXiv:2012.15416*.

Damian Pascual, Béni Egressy, Clara Isabel Meister, Ryan Cotterell, and Roger Wattenhofer. 2021. A plug-and-play method for controlled text generation. *Findings of the Association for Computational Linguistics: EMNLP 2021*, pages 3973–3997.

Alec Radford, Karthik Narasimhan, Tim Salimans, and Ilya Sutskever. 2018. Improving language understanding by generative pre-training.

Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, Ilya Sutskever, et al. 2019. Language models are unsupervised multitask learners. *OpenAI blog*, 1(8):9.

Lei Sha. 2020. Gradient-guided unsupervised lexically constrained text generation. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 8692–8703.

Olga Uryupina, Barbara Plank, Aliaksei Severyn, Agata Rotondi, and Alessandro Moschitti. 2014. Sentube: A corpus for sentiment analysis on youtube social media. In *Proceedings of the Ninth International Conference on Language Resources and Evaluation (LREC'14)*, pages 4244–4249.

Xin Xie, Yi Li, Huaibo Huang, Haiyan Fu, Wanwan Wang, and Yanqing Guo. 2022. Artistic style discovery with independent components. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 19870–19879.

Peng Xu, Mostofa Patwary, Mohammad Shoeybi, Raul Puri, Pascale Fung, Animashree Anandkumar, and Bryan Catanzaro. 2020. Megatron-cntrl: Controllable story generation with external knowledge using large-scale language models. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 2831–2845.

Kevin Yang and Dan Klein. 2021. Fudge: Controlled text generation with future discriminators. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 3511–3535.

Zhilin Yang, Zihang Dai, Yiming Yang, Jaime Carbonell, Russ R Salakhutdinov, and Quoc V Le. 2019. Xlnet: Generalized autoregressive pretraining for language understanding. *Advances in neural information processing systems*, 32.

Dian Yu, Zhou Yu, and Kenji Sagae. 2021. Attribute alignment: Controlling text generation from pre-trained language models. In *Findings of the Association for Computational Linguistics: EMNLP 2021*, pages 2251–2268.

Linhao Zhang and Houfeng Wang. 2021. Towards controlled and diverse generation of article comments. *arXiv preprint arXiv:2107.11781*.

Xiang Zhang, Junbo Zhao, and Yann LeCun. 2015. Character-level convolutional networks for text classification. *Advances in neural information processing systems*, 28.

Coordination Generation via Synchronized Text-Infilling

Hiroki Teranishi and **Yuji Matsumoto**
RIKEN Center for Advanced Intelligence Project
{hiroki.teranishi, yuji.matsumoto}@riken.jp

Abstract

Generating synthetic data for supervised learning from large-scale pre-trained language models has enhanced performances across several NLP tasks, especially in low-resource scenarios. In particular, many studies of data augmentation employ masked language models to replace words with other words in a sentence. However, most of them are evaluated on sentence classification tasks and cannot immediately be applied to tasks related to the sentence structure. In this paper, we propose a simple yet effective approach to generating sentences with a coordinate structure in which the boundaries of its conjuncts are explicitly specified. For a given span in a sentence, our method embeds a mask with a coordinating conjunction in two ways ("X *and* <mask>", "<mask> *and* X") and forces masked language models to fill the two blanks with an identical text. To achieve this, we introduce decoding methods for BERT and T5 models with the constraint that predictions for different masks are synchronized. Furthermore, we develop a training framework that effectively selects synthetic examples for the supervised coordination disambiguation task. We demonstrate that our method produces promising coordination instances that provide gains for the task in low-resource settings.

1 Introduction

Pre-trained language models (LMs) have brought a large shift from preparing a substantial number of training examples and learning task-specific representations to collecting tons of unlabeled text and learning task-agnostic representations. For instance, fine-tuning LMs, such as BERT (Devlin et al., 2019) and BART (Lewis et al., 2020), has improved performances of many natural language processing (NLP) tasks, even when using a small amount of training data. This paradigm shift has led to many challenges to exploit knowledge of LMs rather than manually annotated data, including zero- or few-shot learning (Radford et al., 2019; Brown et al., 2020; Shin et al., 2020; Gao et al., 2021) and data augmentation (Wu et al., 2019; Gao et al., 2019; Anaby-Tavor et al., 2020; Kumar et al., 2020). However, many of these studies focus on sentence classification tasks, such as question answering and sentiment analysis, and thus are not applicable to structured prediction tasks, such as syntactic parsing. Although some studies propose data augmentation methods that manipulate the structure of a sentence (Şahin and Steedman, 2018; Shi et al., 2021), producing syntactic structures using recent pre-trained LMs has not been explored.

As an attempt to produce syntactic structures using LMs, we explore approaches to generating coordinate structures, in which two or more elements, known as *conjuncts*, are linked by a coordinating conjunction (*coordinator*), and yielding the corresponding annotation that explicitly marks which elements in the sentence are coordinated. Coordination frequently appears in natural language sentences and causes ambiguities, leading to errors in many NLP tasks (Hara et al., 2009; Chae et al., 2014; Ficler and Goldberg, 2017). We expect that performances of supervised methods for coordination disambiguation (e.g., Ficler and Goldberg, 2016b; Teranishi et al., 2017, 2019) would be improved if we can generate good-quality coordinate structures with annotation.

In this paper, we present a method that exploits large-scale pre-trained LMs to inject a coordinate structure into a given sentence with the corresponding annotation of coordination boundaries. Specifically, we use masked language models (MLMs) to generate text that seems to be in parallel with a given reference span in a sentence. To enhance MLMs to make use of the *similarity* and *replaceability* properties of conjuncts described by Ficler and Goldberg (2016b) and Teranishi et al. (2017), our method utilizes a pair of sentences, each of which embeds a mask with a conjunction differ-

Proceedings of the 29th International Conference on Computational Linguistics, pages 5914–5924
October 12–17, 2022.

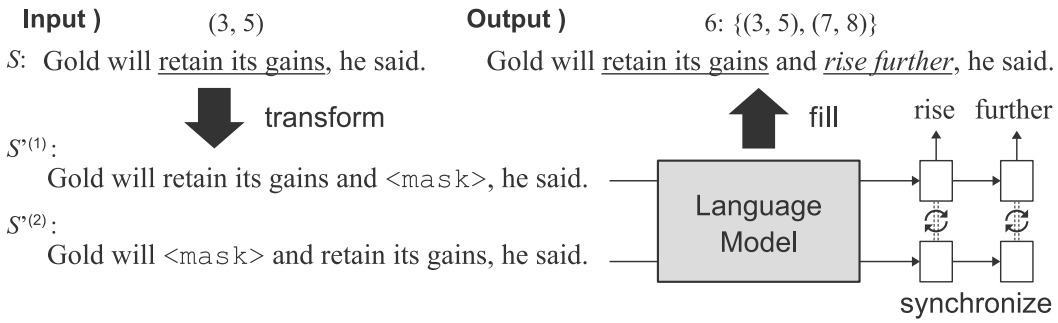

Figure 1: Overview of our approach. An input sentence with a reference span (*"retain its gains"*) is masked in two ways. Then, an MLM fills the masks with an identical text (*"rise further"*) according to the constraint that predictions for different masks are synchronized.

ently, and forces the models to fill the masks with an identical text (Figure 1). For BERT and T5 (Raffel et al., 2020) as instantiations of MLMs, we introduce decoding techniques that follow the constraint that predictions for different masks are made to be synchronized, which can be potentially applied to other generation tasks, such as paraphrasing and summarization. We also describe a training framework to effectively select synthetic examples for supervised models, where a classifier is employed to filter out erroneous samples as similarly used in self-training. Experimental results of the coordination disambiguation task in low-resource settings show that synthetic coordinate structures with annotation provide gains and indicate that our method yields promising instances of coordination.

The contributions of this work are summarized as follows:

- We propose a method for generating a sentence with a coordinate structure and the corresponding annotation that indicates the boundaries of coordination.
- We demonstrate simple decoding techniques for BERT and T5 to fill different masks with an identical text, which can be potentially used for other types of text generation.
- We integrate the generation process into a training framework for supervised methods where synthetic examples are effectively examined and selected.
- Our method produces promising coordination instances that facilitate the supervised coordination disambiguation task in low-resource settings.

2 Related Work

Data augmentation using language models. Data augmentation is a widely used technique to increase the amount of training data by creating synthetic examples from existing ones. In NLP, many replacement-based methods have been explored, which typically generate new sentences by replacing words in sentences with other words. Wei and Zou (2019) demonstrated that a simple synonym replacement using WordNet (Miller, 1995) improves the performance of sentence classification tasks. As synonym-based augmentation can be applied to only a small percentage of the vocabulary covered by a hand-crafted ontology, Kobayashi (2018) instead exploited a recurrent neural network-based LM to produce replacement words that fit in the context. This LM-based method was later extended by Wu et al. (2019). They instead used BERT (Devlin et al., 2019) for replacing words, which is a Transformer-based LM trained on an objective where the model has to predict a token for a masked token in a sentence. Anaby-Tavor et al. (2020) proposed a method to generate a sentence for a category by providing the class label to GPT-2 (Radford et al., 2019). Kumar et al. (2020) further generalized data augmentation methods that exploit pre-trained LMs. Our work is similar to these studies in that we employ a pre-trained LM to generate a new sentence by filling masks with words, but differs in that it does not require a human-annotated label for an input sentence.

Self-training. Self-training is a semi-supervised approach to effectively utilize unlabeled data. Self-training employs a model (*teacher*) trained on a limited amount of labeled data to assign synthetic labels to unlabeled data, which are later used to train another model (*student*). Self-training has been successfully applied to a wide variety of NLP tasks, such as word sense disambiguation (Yarowsky, 1995), syntactic parsing (McClosky

et al., 2006), and machine translation (Zhang and Zong, 2016). The training framework used in our method is inspired by self-training, but unlike general settings of self-training, it employs a trained model only for inspecting synthetic examples that are generated by another model.

Coordination disambiguation. The task of coordination disambiguation is to identify the conjuncts for a given coordinator. Prior to the recent advancement of neural networks, many efforts have been made to explore a range of approaches to this task: rule-based methods (Agarwal and Boggess, 1992; Kurohashi and Nagao, 1992), statistical methods (Resnik, 1999; Chantree et al., 2005), and machine learning-based methods using handcrafted rules (Buyko et al., 2007; Hara et al., 2009). Recently, Ficler and Goldberg (2016b) proposed a neural network-based method that considers the similarity and replaceability between conjuncts. As opposed to the pipeline approach of Ficler and Goldberg (2016b), Teranishi et al. (2017) developed an end-to-end approach using neural networks, which was further extended by Teranishi et al. (2019).

3 Method

This section describes our approach to coordination generation exploiting LMs pre-trained with the objectives for text-infilling. Figure 1 shows the overview of our approach. Formally, for a given sentence $S_{1:N} = w_1, \ldots, w_N$, the goal of our method is to embed a coordinate structure with annotation indicating a set of conjunct spans $\{(i^{(1)}, j^{(1)}), \ldots\}$ together with a coordinator w_k. We currently restrict each generated coordination instance to have two conjuncts linked by the conjunctive coordinator *"and"*. To this end, our method first selects a *reference span* (i, j) in S and generates text homologous to the span using an LM. The produced text is then inserted with a coordinating conjunction after the reference span, which results in a coordinate structure $\{(i, j), (i', j')\}$ accompanied by the coordinator. We explain the details of the method in the following sections.

3.1 Text generation for a span

To generate an alternative candidate for a reference span, a simple approach is to replace the reference span with a mask and use an MLM to fill it, regarding the infilling text as paired with the reference. We initially tested this approach, but found that it is not suitable for coordination generation because the model cannot see the reference span, and thus a produced span could be considerably different from the reference even if the resulting sentence is grammatically correct. Alternatively, we transform a sentence S into a twin of masked sentences $S'^{(1)}$ and $S'^{(2)}$, each of which embeds a mask with a conjunction differently, but the masks must be filled with an identical text. Concretely, we insert a conjunction followed by a mask (e.g. "*and* `<mask>`") after a reference span to obtain $S'^{(1)}$ and insert the same conjunction preceded by the same mask before the reference for $S'^{(2)}$, as depicted in Figure 1. This treatment enables the model to exploit the reference as well as the context for producing conjuncts that are similar and exchangeable (Ficler and Goldberg, 2016b; Teranishi et al., 2017).

3.2 Text infilling using MLM

This section describes how we can force an MLM to fill masks in different sentences with an identical text.

3.2.1 Non-autoregressive LM

We explain a synchronized text-infilling method using BERT (Devlin et al., 2019) as an instance of non-autoregressive LMs. To prepare an input sequence for BERT, we replace each masked span in $S'^{(1)}$ and $S'^{(2)}$ with a sequence of `[MASK]` tokens whose length is equal to the number of the tokens in the reference span, and concatenate $S'^{(1)}$ and $S'^{(2)}$ with a `[SEP]` token, as shown in Figure 2. Feeding the resulting sequence to BERT, the model emits the score vector (i.e., logit) $\mathbf{u}_t \in \mathbb{R}^V$ for each mask token, where V is the vocabulary size. When two masked spans start at l-th and m-th tokens in an input sequence respectively, infilling sequences $T'^{(1)}$ and $T'^{(2)}$ for the two spans are predicted as follows:

$$\begin{aligned} \mathbf{u}'_{l+t} = \mathbf{u}'_{m+t} &= f(\mathbf{u}_{l+t}, \mathbf{u}_{m+t}) \\ T'^{(1)}_{t+1} &= \arg\max(\mathbf{u}'_{l+t}) \\ T'^{(2)}_{t+1} &= \arg\max(\mathbf{u}'_{m+t}) \end{aligned} \quad (1)$$

where t (≥ 0) is an offset from the beginning of the masked spans. Consequently, predicted tokens $T'^{(1)}_{t+1}$ and $T'^{(2)}_{t+1}$ become the same under the restriction of the function f. As a synchronization function f, we choose the element-wise min function rather than the element-wise mean function to avoid picking a token that overfits in either $S'^{(1)}$ or $S'^{(2)}$ context, which happens when the value for

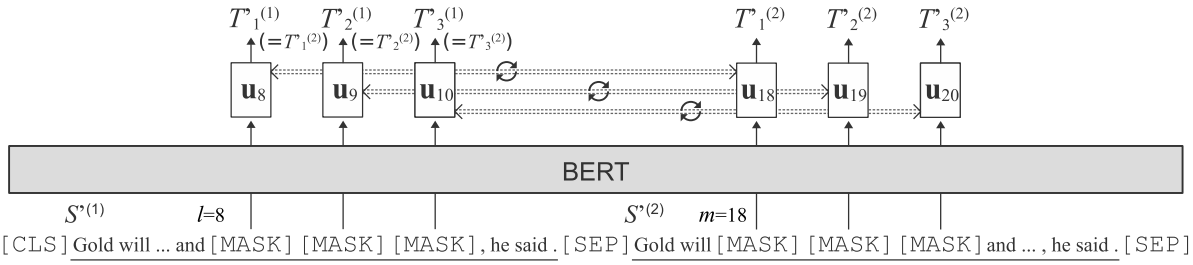

Figure 2: Synchronized decoding on BERT.

a token is extremely high in one logit but is not very high in the other logit. Intuitively, for the two masked sentences *"<X> primarily designed the Apple II."* and *"<X> is a founder of Apple Inc."*, synchronized decoding with the mean function would predict *"Steve Jobs"* for <X> by mistake when it overestimates the value of *"Steve Jobs"* in the second sentence even though the model knows *"Steve Wozniak"* fits adequately in both sentences.

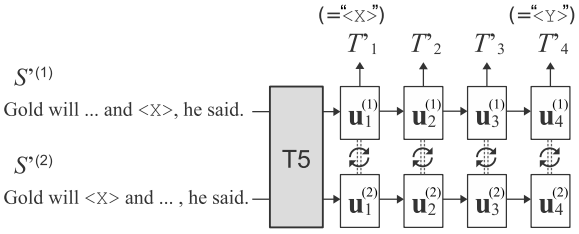

Figure 3: Synchronized decoding on T5.

3.2.2 Sequence-to-sequence LM

We adopt T5 (Raffel et al., 2020) as a sequence-to-sequence LM for our method. In the pre-training scheme of T5, the model consisting of a Transformer encoder and decoder (Vaswani et al., 2017) is trained to generate infilling texts for masked spans. For example, "*Thank you* <X> *me to your party* <Y> *week.*" is recovered by generating "<X> *for inviting* <Y> *last* <Z>", where the last extra token <Z> indicates the end of the sequence. Following this, we feed $S'^{(1)}$ and $S'^{(2)}$ into the encoder independently, but, for each decoding step, the decoder is constrained to output identical tokens for the two inputs (as illustrated in Figure 3):

$$\begin{aligned} \mathbf{u}'^{(1)}_t = \mathbf{u}'^{(2)}_t &= f(\mathbf{u}^{(1)}_t, \mathbf{u}^{(2)}_t) \\ T'^{(1)}_t &= \arg\max(\mathbf{u}'^{(1)}_t) \\ T'^{(2)}_t &= \arg\max(\mathbf{u}'^{(2)}_t) \end{aligned} \quad (2)$$

where f is the element-wise min function and $\mathbf{u}^{(1)}_t$ and $\mathbf{u}^{(2)}_t$ are logits for $S'^{(1)}$ and $S'^{(2)}$ at a decoding step t, respectively. Unlike text infilling by BERT, the decoder in T5 can produce a longer or shorter sequence than the reference span; however, we restrict the number of produced tokens between one-third and three times of the length of the reference span.

3.3 Span selection

Choosing a promising reference span is crucial for our coordination generation method because it becomes one of the conjuncts in a resulting coordinate structure. Ideally, choosing reference spans manually is preferable, which is a similar setting to active learning, but it is laborious. As an alternative, one strategy for automatic span selection is to randomly choose a sequence of tokens, as done in SpanBERT (Joshi et al., 2020). However, such a sequence is readily disqualified as a conjunct when it appears across two constituents. Alternatively, we can apply a constituency parser to a sentence and randomly pick one constituent from the parse tree. The constituency-based span selection is particularly suitable for our method because we can also restrict reference spans with their constituent labels[1].

4 Training Framework for Coordination Disambiguation

A synthetic example generated by the method described in Section 3 can be immediately seen as an annotated sentence, but the resulting sentence can be inconsistent with its coordination boundary. For example, we expect a sentential conjunct for the reference *"He said Mary had dinner with John."*, but an MLM can produce *"Bill"* rather than a sentence because both *"John and Bill"* as objects in $S'^{(1)}$ and *"Bill and Mary"* as subjects in $S'^{(2)}$ are

[1]Other than constituency, coordination involving lexical compounds can be created using hand-crafted rules based on part-of-speech. We tried this for the supervised task described in Section 5 for preliminary experiments, but it did not lead to a significant performance gain.

Algorithm 1 Training with generation and filtering

Input: labeled sentences L; unlabeled sentences U; predictor M; generator G

Constants: batch size N_{batch}; number of generated examples in a batch K ($\leq N_{batch}$); maximum number of generation trials K_{max}; threshold for filtering δ

Output: trained model M

train M on L
repeat
  $B \leftarrow \{\}$
  $k \leftarrow 0$
  while $|B| < K$ **and** $k < K_{max}$ **do**
    $x \leftarrow \textsc{Sample}(U, 1)$
    $(x', y') \leftarrow G(x)$
    if $\textsc{CalcScore}((x', y'), M) \geq \delta$ **then**
      $B \leftarrow B \cup \{(x', y')\}$
    end if
    $k \leftarrow k + 1$
  end while
  $B \leftarrow B \cup \textsc{Sample}(L, N_{batch} - K)$
  update M on B
until convergence
return M

coincidentally valid coordination. However, the resulting boundary annotation *"He said [Mary had dinner with John] and [Bill]."* is not considered to be correct. Although we can remove such examples by employing a filtering method that ensures the consistency between conjuncts, for example, using part-of-speech tags, developing such rule-based methods requires much effort.

For the purpose of improving supervised methods for the coordination disambiguation task, we instead propose a training framework inspired by self-training. The algorithm of our training method is described in Algorithm 1. At the first step of the procedure, a supervised model M for the task is trained on a limited amount of labeled examples L. In the following steps, the model is further trained on labeled examples L and synthetic examples generated from a large number of unlabeled sentences U. Concretely, in each training iteration, we pick one unlabeled sentence x that contains no coordination from U and feed it to an LM G (that encapsulates a span selection mechanism) to obtain a sentence x' that has a coordinate structure with the corresponding annotation y' using the method explained in Section 3. After generation, we employ the model M to assign a score to the example (x', y') for inspection. If the score is greater than or equal to a predefined threshold δ, the example is verified as legitimate and then added to a mini-batch. We repeat this generation–inspection process until we obtain K examples or until the process is repeated K_{max} times, which could happen when the generator G does not give promising examples. For the rest of $N_{batch} - K$ examples, we sample labeled examples from L. Note that we sample an unlabeled sentence x from U with replacement to reuse it because different coordinate structures can be created in it by choosing different reference spans.

	PTB	GENIA
# train sentences	39,832 (15,481)	14,326 (8,101)
# train instances	19,890	11,596
# dev sentences	1,700 (673)	1,361 (777)
# dev instances	848	1,146
# test sentences	2,416 (873)	1,360 (773)
# test instances	1,099	1,166

Table 1: Statistics of the Penn Treebank and the GENIA Treebank. Numbers in parentheses represent the number of sentences that have coordination instances[2].

5 Experiments

We evaluate our method using generated examples to train supervised models for the coordination disambiguation task.

5.1 Experimental settings

Data construction. We perform experiments on the coordination-annotated Penn Treebank (PTB) (Ficler and Goldberg, 2016a) and the GENIA Treebank (GENIA) (Tateisi et al., 2005). The statistics of the corpora are shown in Table 1. To simulate low-resource situations, where additional training examples could improve performance, we randomly sample 300 (250 for training and 50 for validation) and 600 (500 for training and 100 for validation) sentences containing coordination with annotation from the training set; the rest of sentences in the training set can be used as unlabeled seed data for generation. Specifically, the PTB training set consists of 39,832 sentences, 15,481 of which have coordinate structures. A random subset of them is used as a set of labeled sentences L, while 24,351 sentences with no coordination

[2]Following the previous studies (Ficler and Goldberg, 2016b; Teranishi et al., 2019), we focus on coordinate structures formed by "and", "or", "and/or", or "but".

#	Model	NP, NX	ADJP, ADVP	VP	PP	S, SBAR	Overall
			Penn Treebank				
250	baseline (no generation)	66.21	64.72	65.42	52.30	51.69	62.83 ± 2.36
250	+ generation by BERT	68.56	71.63	64.22	52.30	51.94	64.49 ± 2.42
250	+ generation by T5	**71.17**	**74.54**	**68.54**	**80.00**	**59.37**	**68.68** ± 1.51
500	baseline (no generation)	73.34	80.72	71.35	69.23	62.51	71.04 ± 1.59
500	+ generation by BERT	73.24	76.72	72.96	76.92	62.13	71.44 ± 1.57
500	+ generation by T5	**75.09**	**81.81**	**73.56**	**84.61**	**63.64**	**73.17** ± 1.56
			GENIA Treebank				
250	baseline (no generation)	48.39	69.50	62.96	57.46	32.65	50.49 ± 2.37
250	+ generation by BERT	54.81	71.00	62.45	65.35	41.77	56.07 ± 1.95
250	+ generation by T5	**57.26**	**73.50**	**65.41**	**66.47**	**42.53**	**58.39** ± 2.07
500	baseline (no generation)	54.71	69.50	65.93	64.50	53.16	57.11 ± 0.86
500	+ generation by BERT	57.60	75.00	69.03	65.63	54.93	59.87 ± 1.08
500	+ generation by T5	**58.10**	**78.50**	**69.29**	**67.32**	**57.46**	**60.70** ± 0.78

Table 2: Fine-tuning results with synthetic data on the PTB and GENIA test sets[5]. The first column indicates the number of annotated examples taken from the training set. We report accuracy averaged across five runs (with standard deviation) throughout sampling, generation, and fine-tuning.

are used to construct a set of unlabeled sentences U[3]. The same operation is also performed for the GENIA training set. We use the validation sets constructed from the training sets for model selection and the test sets for evaluation. The original development sets in the corpora are reserved for extensive analyses.

Evaluation and model. We evaluate models on the basis of their abilities to predict the correct beginning and ending positions of the coordinate structure for a given coordinating conjunction. As a supervised model for the task, we employ the pre-trained BERT-base-cased model combined with a coordination prediction module used in Teranishi et al. (2019), which gives scores to all possible endpoint pairs for a conjunction. The score of an endpoint pair (i, j) for the conjunction at the k-th position in a sentence is computed as follows:

$$\begin{aligned} \textsc{Score}_\theta(i,j,k) &= \text{MLP}(F_\theta(i,j,k)) \\ F_\theta(i,j,k) &= [\mathbf{h}_i - \mathbf{h}_{k+1}; \mathbf{h}_j - \mathbf{h}_{k-1}] \end{aligned} \quad (3)$$

where MLP is a network consisting of two linear transformations with a ReLU activation, and $\mathbf{h}_t$ is a representation vector for the t-th token produced from the last layer of the Transformer in the BERT model. All scores for possible pairs are normalized by the softmax function and thus each score takes a value between 0 and 1.

Training methodology. The entire network is updated for 10,000 steps and the best-performing model on the constructed validation set is selected for evaluation on the test set. We regard the model trained only on L as a baseline, whereas the proposed model is trained first on L for 1,000 steps (line 1 in Algorithm 1) and later on L together with generated examples using U for the rest of 9,000 steps (line 2–15 in Algorithm 1). We set $N_{batch} = 16$, $K = 8$, $K_{max} = 16$ and $\delta = 0.7$ for the proposed training framework in all experiments. We use BERT-large-cased and T5-large models for coordination generation. For span selection, we use the Berkeley neural parser (Kitaev et al., 2019) as the constituency-based span selection instead of conducting the manual selection for an active learning setting. Targeted spans are limited to non-terminals of a parse tree labeled with NP, VP, ADJP, ADVP, PP, SBAR, or S as they are typical types of conjuncts[4]. We use the AdamW optimizer with a learning rate of 2e-5, which is linearly decreased with no warm-up, gradient clip with the threshold of 1.0, and weight decay of 0.01. We evaluate models every 100 steps and apply early stopping when the performance is not improved for 1,000 steps after the initial 1,000 steps. An MLP in Eq. 3 has one hidden layer of 256 units with dropout ($p = 0.5$). We use the last token of each word when computing scores for word-level boundaries of coordination.

[3]In practice, to give sufficient context information for generation, we exclude sentences from U that consist of less than 10 words.

[4]We exclude the root constituent that covers an entire sentence.

Setting	Overall
baseline	64.74
+ generation by BERT	66.98
– filtering	64.43
– sync + self-train	65.77
+ generation by T5	69.38
– filtering	66.13
– sync + self-train	65.37

Table 3: Results of the ablation study conducted on the PTB development set. We report accuracy averaged across five runs.

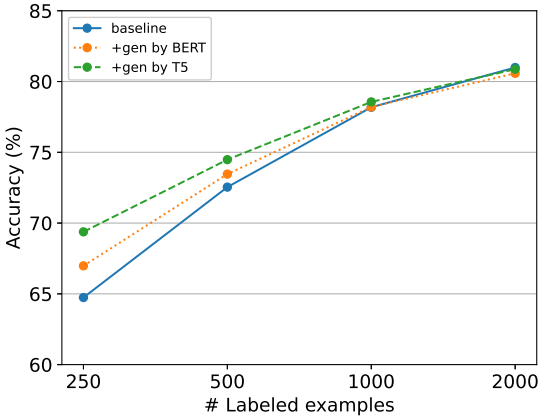

Figure 4: Influence of training dataset size evaluated on the PTB development set.

5.2 Results and analysis

5.2.1 Performance

Table 2 shows the experimental results on the PTB and GENIA test sets. Trained with coordination generation by our proposed method, the coordination disambiguation model generally achieves better performances than the baseline, which is trained only on a limited amount of manually annotated data. This indicates that both of the BERT and T5 models used with the proposed method yield promising examples that provide performance gains for the task on the PTB as well as on the GENIA, although the underlying parser and LMs for generation are not particularly adapted to the biomedical domain. A comparison between different LMs for generation shows that the T5 models produce better-quality examples than those produced by the BERT models for all types of coordination. This is because BERT is pre-trained to predict a single token for each mask independently, whereas T5 is pre-trained through sequence-to-sequence learning to generate consecutive tokens for masked spans. Thus, the BERT models only work when generating conjuncts each consisting of a few tokens; however, this would be mitigated by fine-tuning for span prediction, as used in the training of SpanBERT (Joshi et al., 2020)[6]. Surprisingly, the performances on clause-level coordination (labeled as S or SBAR) are improved by generation with T5, although T5 models are pre-trained to fill relatively small masks and thus the out-of-the-box models should have poor ability to generate text for a long masked span.

[5] *Overall* includes other minor categories, such as UCP and QP.

[6] We intended to use a SpanBERT model as a variant of BERT for generation, but the weights of the LM heads for masked span prediction are not officially provided.

5.2.2 Ablation study

We conduct additional experiments for ablation study on the PTB development set. We report performances of models trained using 250 labeled examples and synthetic examples without filtering by setting $\delta = 0$ or without synchronized decoding for generation. Specifically in the latter setting, we use only one masked sentence where a mask token and conjunction are inserted only after a reference (equivalent to $S'(1)$) and employ an LM to fill a mask without synchronized decoding. However, because the LM has no clue about the boundaries of a reference span in the masked sentence, it likely produces a text that does not seem to be coordinated with the reference (e.g., *"I had lunch and* `<mask>` *with Mary."* $\rightarrow$ *"I [had lunch] and [dinner] with Mary."*). Such an example is easily rejected by filtering during training, which results in that training without synchronization is no longer different from training only on L. Therefore, when not in use of the synchronized decoding, we discard the corresponding boundary annotation y' produced by G for the resulting sentence x' and instead use a predictor model M to assign the most probable endpoint pair y'' to the sentence x'. The example (x', y'') is then added if its score is greater than or equal to the threshold δ. This setting can be seen as typical self-training.

Table 3 shows the results of the ablation study. Without filtering synthetic examples in the training, the performances are substantially degraded, which indicates that the proposed generation method produces erroneous examples, but also that the influence of noisy examples for supervised learning is effectively mitigated by filtering. However, the proposed generation method combined with T5

1.	Accepted	We're sorry to report that on Monday President Bush accepted the resignation of William Allen as chairman [of the U.S. Civil Rights Commission] and [of the U.S. Department of Justice].
2.	Accepted	The opposition charged that the money was used [to bribe Indian government officials] and [to support the Congress], an allegation denied by Mr. Gandhi's administration.
3.	Rejected	"[I wouldn't say it's quite a veto] and [it's not a veto]," Mr. Boren demurs.
4.	Rejected	Most analysts had expected a sharper decline [after the steep rise in August] and [earlier in this year].

Table 4: Synthetic labeled examples generated using the proposed method with T5. The underlined text indicates the reference span used in generation. The first column indicates whether the instance is considered valid or not by the predictor model.

without filtering still leads to performance gain, whereas generation with BERT without filtering slightly impairs the performance. This result supports the superiority of the T5 model over the BERT model for the proposed method. The models self-trained on sentences generated by LMs without the synchronized decoding outperform the baseline, demonstrating the effectiveness of self-training. However, they underperform the models trained with the proposed method that employs the synchronized decoding. This indicates that the boundary annotations assigned by the predictor itself become noises for training and would be amplified as training progresses, whereas the annotations produced by the synchronized decoding are more consistent regardless of the training progress.

5.2.3 Influence of the size of labeled examples

We further investigate the performance improvement for the cases in which the size of human-annotated labeled examples is increased. We sample 250, 500, 1,000, and 2,000 labeled examples from the PTB training set to construct L and report performances on the PTB development set. Figure 4 shows results. The models trained with the proposed generation method with BERT and T5 enhance performances for low-resource settings, as demonstrated in Table 2. However, the improvement is no longer observed when preparing sufficient labeled examples. Note that when using the full training set (15,481 sentences) for L, the baseline model and the model with generation using T5 achieve 86.36% and 86.24% accuracy on the PTB development set, respectively. Thus, employing the proposed method with many hand-annotated examples does not degrade the performance significantly.

5.2.4 Qualitative analysis

Table 4 shows synthetic examples generated by the proposed method using the T5 model. For Examples 1 and 2, the LM successfully captures the boundary of the prepositional phrase and the verb phrase, respectively, and generates the texts that are coordinated with the reference spans with the help of the proposed masking strategy and the synchronized decoding. For Example 3, the model produces a clause that corresponds to the reference, but the resulting boundary annotation does not appear to be a valid conjunct for the reference span due to the generator model's misunderstanding of the coordination boundary. This malformed example is effectively rejected for training by the predictor model. The adverbial coordinate structure in Example 4 generated by the LM seems to have the correct boundary, but is mistakenly rejected because the predictor model is not sure whether it is correct or not.

6 Limitations

The main weakness of the proposed approach is its dependence on a pre-trained large LM, which may not be available in an extremely low-resource setting in terms of the amount of unlabeled text or computational resources. Furthermore, the effectiveness of our method has been examined for gen-

erating a limited form of coordination in English (i.e., two conjuncts joined by *"and"*). Thus, additional treatments for other forms of coordination in English or for coordination in other languages using multilingual LMs would need to be developed and tested with more thorough qualitative analyses through human evaluation.

7 Conclusion

We present a method that exploits pre-trained LMs for generating coordination with boundary annotation. Synchronized decoding by BERT and T5 for two masked sentences with conjunctions leads to coordinate structures that are consistent with synthetic annotation. The experimental results for coordination disambiguation show that examples generated by our method provide gains to supervised methods. The extensive study also supports the effectiveness of the proposed training framework for removing erroneous examples. As future work, we intend to develop a fine-tuning method on LMs specialized for coordination generation and apply synchronized decoding to other LMs and to other generation tasks.

Acknowledgments

This study was partly supported by JSPS KAKENHI Grant Number 22K17957. We are grateful to the anonymous reviewers for their helpful insights and comments.

References

Rajeev Agarwal and Lois Boggess. 1992. A simple but useful approach to conjunct identification. In *30th Annual Meeting of the Association for Computational Linguistics*, pages 15–21, Newark, Delaware, USA. Association for Computational Linguistics.

Ateret Anaby-Tavor, Boaz Carmeli, Esther Goldbraich, Amir Kantor, George Kour, Segev Shlomov, Naama Tepper, and Naama Zwerdling. 2020. Do not have enough data? deep learning to the rescue! *Proceedings of the AAAI Conference on Artificial Intelligence*, 34(05):7383–7390.

Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel Ziegler, Jeffrey Wu, Clemens Winter, Chris Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. 2020. Language models are few-shot learners. In *Advances in Neural Information Processing Systems*, volume 33, pages 1877–1901. Curran Associates, Inc.

Ekaterina Buyko, Katrin Tomanek, and Udo Hahn. 2007. Resolution of coordination ellipses in biological named entities using conditional random fields. In *Proceedings of the 10th Conference of the Pacific Association for Computational Linguistics*, pages 163–171.

Jeongmin Chae, Younghee Jung, Taemin Lee, Soonyoung Jung, Chan Huh, Gilhan Kim, Hyeoncheol Kim, and Heungbum Oh. 2014. Identifying non-elliptical entity mentions in a coordinated np with ellipses. *Journal of Biomedical Informatics*, 47:139–152.

Francis Chantree, Adam Kilgarriff, Anne. de Roeck, and Alistair Willis. 2005. Disambiguating coordinations using word distribution information. In *Proceedings of Recent Advances in Natural Language Processing*, pages 287–294.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, Minneapolis, Minnesota. Association for Computational Linguistics.

Jessica Ficler and Yoav Goldberg. 2016a. Coordination annotation extension in the Penn Tree Bank. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 834–842, Berlin, Germany. Association for Computational Linguistics.

Jessica Ficler and Yoav Goldberg. 2016b. A neural network for coordination boundary prediction. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 23–32, Austin, Texas. Association for Computational Linguistics.

Jessica Ficler and Yoav Goldberg. 2017. Improving a strong neural parser with conjunction-specific features. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 2, Short Papers*, pages 343–348, Valencia, Spain. Association for Computational Linguistics.

Fei Gao, Jinhua Zhu, Lijun Wu, Yingce Xia, Tao Qin, Xueqi Cheng, Wengang Zhou, and Tie-Yan Liu. 2019. Soft contextual data augmentation for neural machine translation. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 5539–5544, Florence, Italy. Association for Computational Linguistics.

Tianyu Gao, Adam Fisch, and Danqi Chen. 2021. Making pre-trained language models better few-shot learners. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 3816–3830, Online. Association for Computational Linguistics.

Kazuo Hara, Masashi Shimbo, Hideharu Okuma, and Yuji Matsumoto. 2009. Coordinate structure analysis with global structural constraints and alignment-based local features. In *Proceedings of the Joint Conference of the 47th Annual Meeting of the ACL and the 4th International Joint Conference on Natural Language Processing of the AFNLP*, pages 967–975, Suntec, Singapore. Association for Computational Linguistics.

Mandar Joshi, Danqi Chen, Yinhan Liu, Daniel S. Weld, Luke Zettlemoyer, and Omer Levy. 2020. SpanBERT: Improving pre-training by representing and predicting spans. *Transactions of the Association for Computational Linguistics*, 8:64–77.

Nikita Kitaev, Steven Cao, and Dan Klein. 2019. Multilingual constituency parsing with self-attention and pre-training. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 3499–3505, Florence, Italy. Association for Computational Linguistics.

Sosuke Kobayashi. 2018. Contextual augmentation: Data augmentation by words with paradigmatic relations. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 2 (Short Papers)*, pages 452–457, New Orleans, Louisiana. Association for Computational Linguistics.

Varun Kumar, Ashutosh Choudhary, and Eunah Cho. 2020. Data augmentation using pre-trained transformer models. In *Proceedings of the 2nd Workshop on Life-long Learning for Spoken Language Systems*, pages 18–26, Suzhou, China. Association for Computational Linguistics.

Sadao Kurohashi and Makoto Nagao. 1992. Dynamic programming method for analyzing conjunctive structures in Japanese. In *COLING 1992 Volume 1: The 14th International Conference on Computational Linguistics*.

Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Veselin Stoyanov, and Luke Zettlemoyer. 2020. BART: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7871–7880, Online. Association for Computational Linguistics.

David McClosky, Eugene Charniak, and Mark Johnson. 2006. Effective self-training for parsing. In *Proceedings of the Human Language Technology Conference of the NAACL, Main Conference*, pages 152–159, New York City, USA. Association for Computational Linguistics.

George A. Miller. 1995. Wordnet: A lexical database for english. *Commun. ACM*, 38(11):39–41.

Alec Radford, Jeff Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. 2019. Language models are unsupervised multitask learners.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J. Liu. 2020. Exploring the limits of transfer learning with a unified text-to-text transformer. *Journal of Machine Learning Research*, 21(140):1–67.

Philip Resnik. 1999. Semantic similarity in a taxonomy: An information-based measure and its application to problems of ambiguity in natural language. *Journal of Artificial Intelligence Research*, 11(1):95–130.

Gözde Gül Şahin and Mark Steedman. 2018. Data augmentation via dependency tree morphing for low-resource languages. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 5004–5009, Brussels, Belgium. Association for Computational Linguistics.

Haoyue Shi, Karen Livescu, and Kevin Gimpel. 2021. Substructure substitution: Structured data augmentation for NLP. In *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*, pages 3494–3508, Online. Association for Computational Linguistics.

Taylor Shin, Yasaman Razeghi, Robert L. Logan IV, Eric Wallace, and Sameer Singh. 2020. AutoPrompt: Eliciting Knowledge from Language Models with Automatically Generated Prompts. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 4222–4235, Online. Association for Computational Linguistics.

Yuka Tateisi, Akane Yakushiji, Tomoko Ohta, and Jun'ichi Tsujii. 2005. Syntax annotation for the GENIA corpus. In *Companion Volume to the Proceedings of Conference including Posters/Demos and tutorial abstracts*.

Hiroki Teranishi, Hiroyuki Shindo, and Yuji Matsumoto. 2017. Coordination boundary identification with similarity and replaceability. In *Proceedings of the Eighth International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 264–272, Taipei, Taiwan. Asian Federation of Natural Language Processing.

Hiroki Teranishi, Hiroyuki Shindo, and Yuji Matsumoto. 2019. Decomposed local models for coordinate structure parsing. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for*

Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers), pages 3394–3403, Minneapolis, Minnesota. Association for Computational Linguistics.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Ł ukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc.

Jason Wei and Kai Zou. 2019. EDA: Easy data augmentation techniques for boosting performance on text classification tasks. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 6382–6388, Hong Kong, China. Association for Computational Linguistics.

Xing Wu, Shangwen Lv, Liangjun Zang, Jizhong Han, and Songlin Hu. 2019. Conditional bert contextual augmentation. In *Computational Science – ICCS 2019*, pages 84–95. Springer International Publishing.

David Yarowsky. 1995. Unsupervised word sense disambiguation rivaling supervised methods. In *33rd Annual Meeting of the Association for Computational Linguistics*, pages 189–196, Cambridge, Massachusetts, USA. Association for Computational Linguistics.

Jiajun Zhang and Chengqing Zong. 2016. Exploiting source-side monolingual data in neural machine translation. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 1535–1545, Austin, Texas. Association for Computational Linguistics.

KHANQ: A Dataset for Generating Deep Questions in Education

Huanli Gong[1] **Liangming Pan**[2] **Hengchang Hu**[2]
[1]The Ohio State University, Columbus, OH, USA
[2]School of Computing, National University of Singapore, Singapore
gong.545@osu.edu
{liangmingpan, hengchanghu}@u.nus.edu

Abstract

Designing in-depth educational questions is a time-consuming and cognitively demanding task. Therefore, it is intriguing to study how to build Question Generation (QG) models to automate the question creation process. However, existing QG datasets are not suitable for educational question generation because the questions are not real questions asked by humans during learning and can be solved by simply searching for information.

To bridge this gap, we present KHANQ, a challenging dataset for educational question generation, containing 1,034 high-quality learner-generated questions seeking an in-depth understanding of the taught online courses in Khan Academy. Each data sample is carefully paraphrased and annotated as a triple of 1) *Context*: an independent paragraph on which the question is based; 2) *Prompt*: a text prompt for the question (*e.g.*, the learner's background knowledge); 3) *Question*: a deep question based on *Context* and coherent with *Prompt*. By conducting a human evaluation on the aspects of appropriateness, coverage, coherence, and complexity, we show that state-of-the-art QG models which perform well on shallow question generation datasets have difficulty in generating useful educational questions. This makes KHANQ a challenging testbed for educational question generation for further investigation.

1 Introduction

Question asking has long been considered a fundamental cognitive process in learning. An ideal learner should be an active, self-motivated, creative, inquisitive person who asks deep questions and searches for answers to thought-provoking questions, usually in the form of *Why*, *Why-not*, *How*, *What-if*, etc (Otero and Graesser, 2001). For example, Figure 1 shows a question raised by a learner after learning "allosteric regulation and feedback loops" in Khan Academy[1], a well-known online education platform. Given that question-asking is a hallmark of human intelligence, it is intriguing to study whether we can endow machines with the ability to ask deep and to-the-point questions.

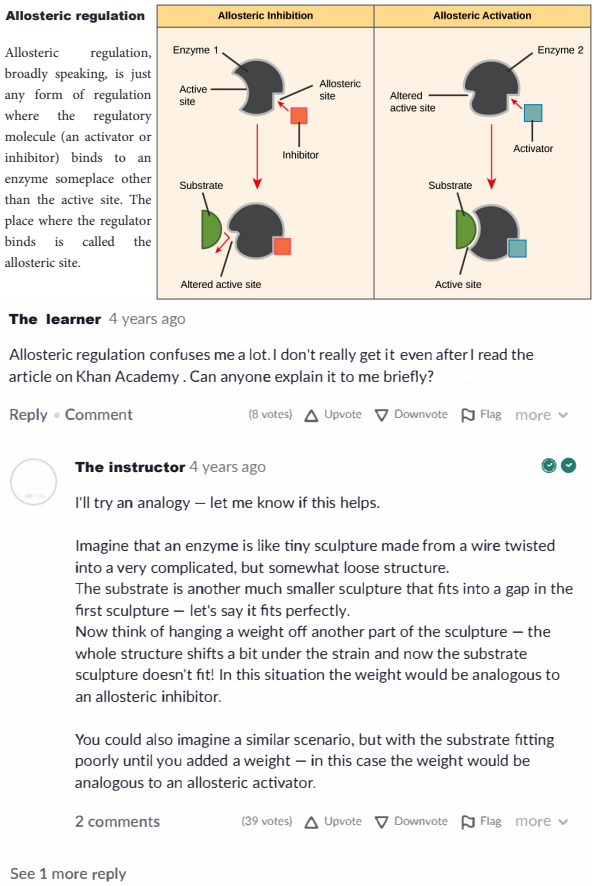

Figure 1: A real human-raised question after learning the "allosteric regulation and feedback loops" course in Khan Academy.

While there have been considerable advances made in the field of Question Generation (QG), the current research is still far from achieving human-like question-asking. One major obstacle is the lack of a suitable and clean dataset that can represent

[1]https://www.khanacademy.org/

Proceedings of the 29th International Conference on Computational Linguistics, pages 5925–5938
October 12–17, 2022.

real-life human-raised questions, such as the question shown in Figure 1. Existing QG works (Duan et al., 2017; Zhao et al., 2018; Pan et al., 2020; Back et al., 2021) typically focus on generating factoid questions relevant to one fact obtainable from a single sentence. They are shallow because they do not reflect the creative *human cognitive process* in question-asking such as inferences, multi-step reasoning, synthesis, critical evaluation, and generalization.

To bridge this gap, we desire a new dataset with questions that satisfy two conditions: 1) they are real questions asked by humans during learning, and 2) they are deep questions that require high-level cognitive skills to raise. We choose to collect data from Khan Academy, where the questions are raised by real learners after watching course videos or reading course materials, as shown in Figure 1. These questions represent what learners are naturally interested in, and they are rarely shallow questions confined to a narrow scope of context.

We collect the question-answer pairs from Khan Academy and rewrite them as a triple of 1) *Context*: an independent paragraph on which the question is based; 2) *Prompt*: a text prompt for the question (*e.g.*, the learner's background knowledge); 3) *Question*: a deep question based on *Context* and coherent with *Prompt*. Following the common setting of question generation, questions are based on the information in the context. However, many valid questions can be asked from the given context. To facilitate the evaluation and to guide the generation, we also provide *Prompt* which describes the questioner's background knowledge (*e.g.*, "In my understanding, ...") or certain conditions for the question (*e.g.*, "When ... happens, ..."). Given both *Context* and *Prompt* as inputs, we test the model's ability to generate a consistent question with both the context and the prompt. We design a rigorous data annotation procedure to ensure that each sample in KHANQ satisfies the following conditions: 1) *Question* involves deep reasoning instead of simply searching for information; 2) *Context* contains enough information to derive *Question*; 3) *Question* is coherent with *Prompt*. After careful annotation, we collect 1,034 (*Context*, *Prompt*, *Question*) triples to form the KHANQ. An example is given in Figure 2.

We further evaluate the depth of the questions in KHANQ. We find that KHANQ is dominated by deep questions that are represented in the form of *Why*, and *How*. In contrast, SQuAD 2.0 (Rajpurkar et al., 2018) and HotpotQA (Yang et al., 2018) are dominated by shallow questions represented in the form of *what*. We further classify these questions based on their reasoning type following the criteria in (Cao and Wang, 2021). The results show that 51.58% of questions in KHANQ involve deep reasoning.

Context: The molecules in the water have a range of kinetic energies, from low to high. Some of the molecules have sufficient kinetic energy to break out and to enter the air. In coastal areas there are also factors. For example, waves churn up the water and give rise to very fine droplets that can get carried in the wind.
Prompt: Water have strong cohesion
Question: How does water evaporate?

Figure 2: A data sample in our KHANQ dataset.

We evaluate the performance of five text generation models, which have achieved state-of-the-art question generation performance on the SQuAD dataset: BertGeneration (Rothe et al., 2020), GPT-2 (Radford et al., 2019), BART (Lewis et al., 2020), Google-T5 (Raffel et al., 2020), UniLM (Dong et al., 2019). By conducting both automatic and human evaluation, we find that although the abilities of BART and Google-T5 to reason in settings and ask deep questions are better than other models, the question quality is still much lower than the human level. The results show that the KHANQ is a challenging testbed for generating human-like deep questions in education.

2 Related Work

Question Generation (QG) aims to automatically generate questions from textual input. Earlier work relied on syntactic transformations to convert declarative sentences into questions (Chali and Hasan, 2015; Heilman and Smith, 2010). Recent neural models (Zhou et al., 2019; Krishna and Iyyer, 2019; Sun et al., 2018) rely on sequence-to-sequence models to generate questions from a given sentence or paragraph by considering the focus, type, and general specific relations of the question. However, these approaches only involves generating shallow factoid questions, which are typically trained and evaluated on the SQuAD (Rajpurkar et al., 2016) dataset. SQuAD is insufficient to evaluate deep QG because more than 80% of its questions are only relevant to information confined to a single sentence (Du et al., 2017).

With the advent of pretraining language models, the challenge of generating single-hop questions similar to SQuAD have largely been addressed. QG research has started to generate more complex questions that require multi-hop reasoning (Tuan et al., 2020; Pan et al., 2020; Xie et al., 2020; Yu et al., 2020), benchmarking on the HotpotQA (Yang et al., 2018) dataset. However, for questions in HotpotQA, the reasoning is often evident from the surface form of the question, simplifying the QG task. For example, Pan et al. (2021) show that HotpotQA-style multi-hop questions can be generated by composing single-hop questions through templates. Different from SQuAD and HotpotQA, questions in our KHANQ dataset are asked by real course learners, thus having a wider variety in both question forms and reasoning types.

Recently, a few works started to work on generating real human-raised questions. For example, Cao and Wang (2021) collect real questions from online forums (Reddit and Yahoo). Gupta et al. (2019) collect questions posted by customers on Amazon product pages. However, questions collected in above works are noisy, with few in-depth questions, since they fail to carefully filter and validate the questions. Compared to them, questions in KHANQ are carefully filtered and annotated, providing a more clean testbed for deep question generation. Among these works, LearningQ (Chen et al., 2018) is closest to us, which also collects questions from Khan Academy. However, our work has three key differences. First, LearningQ is more focused on the educational nature of the questions. They filter questions based on whether they are useful for learning, whereas we focus more on the depth of the question, keeping only the questions that involve deep reasoning. Second, we use prompts to give the models additional guidance on what information to focus on when generating. Third, the contexts used by LearningQ are entire articles or videos, causing most of the sentences in the contexts being irrelevant to the target question. In contrast, the contexts we use are answers that contain comprehensive explanations of the knowledge points relevant to the question.

3 Data Collection and Annotation

3.1 Text Sources

Khan Academy is an online education institution that provides free teaching materials. Online course learners are encouraged to ask questions in the corresponding forum to clarify their confusions after they learned each section of the course, as shown in the example in Figure 1. Therefore, these questions usually reflect a high-level comprehension and cognition of the course contents, which makes them a suitable data source for building a dataset for deep question generation. We chose to collect data from the courses in the science domain because question-asking is more active in the science-related courses than others. Learners asked more questions and most of them have been answered by tutors.

As shown in Table 1, we collected a total number of 1,284 course sections from 11 different areas under the science domain. Each course section consists of the course material (an article) written by the tutor and a discussion forum. We filtered out those course sections that have no question in their discussion forums. In total, we collected 100,908 question-answer pairs.

area	number
High school biology	209
AP/College Biology	296
High school chemistry beta	4
AP/College Chemistry	307
Organic chemistry	290
High school physics	82
AP/College Physics 1	4
AP/College Physics 2	19
AP/College Environmental science	18
Cosmology and astronomy	2
Electrical engineering	53

Table 1: The number of collected course sections for each area in the science domain.

3.2 Data Annotation

To build a clean and high-quality dataset for deep question generation, we then designed a rigorous data annotation procedure to filter out noisy data in our collected question-answer pairs. After filtration and annotation, each data sample is annotated as a triple of: 1) *Context*: an independent paragraph which the question is based on; 2) *Prompt*: a text prompt for the question (*e.g.*, the learner's background knowledge); 3) *Question*: a deep question based on *Context* and *Prompt*. Compared with the original noisy question-answer pairs collected from forums, data annotation provides a standardized and clean benchmark to train and evaluate the question generation models.

Figure 3 summarizes the major steps of our data annotation. First, the *Context* of the question should cover the knowledge points that the question are based on. To provide the context, we ask annotators to revise the answer provided by the tutors of the course. We find that most answers contain comprehensive explanations of the knowledge points relevant to the question from the course. Therefore, they are suitable to serve as the context for the question. We asked the annotators to remove the answer-tone phrases (*e.g.*, "yes", "this is because") and conversational language (*e.g.*, "good question", "hope it helps") to make the answer context-independent and self-contained.

For the original questions raised by learners, they often contain conditional clauses, prepositional phrases, and other conditions to limit the scope of the question or to provide some background information that expresses the learner's own understanding or opinions. As shown in Figure 3, we ask human annotators to separate out this information from the original question to serve as *Prompt* of the question. The prompt provides additional guidance for the question generation model in knowing which information to focus on when generating; otherwise, the QG model tend to generate questions without specific target (e.g., "Can you explain this part again?"). The *Question* then comes from the remaining part of the original question that seeks for new information based on the prompt. Appendix A gives detailed data filtration criteria, specific data annotation guidelines for each step, and examples of annotation.

3.3 Quality Control

To control the quality of the annotation, we require that annotators have a US high school diploma or equivalent to demonstrate sufficient background knowledge to understand the questions. To ensure that annotators understand the annotation procedure, we check those works and give feedback when an annotator has completed the first 10% of the tasks. The annotator needs to redo those annotations based on the feedback. This process is repeated until all 10% of the tasks are approved. During the annotation process, we also spot-check the work submitted by the annotators. If the pass rate does not reach 85%, the annotator needs to be retrained to prevent them from deviating from the task definition. Based on the above criteria, we hired a total of six annotators. The whole annotation process took two months. 30,000 original question-answer pairs are examined and 1,217 of them are annotated as valid samples.

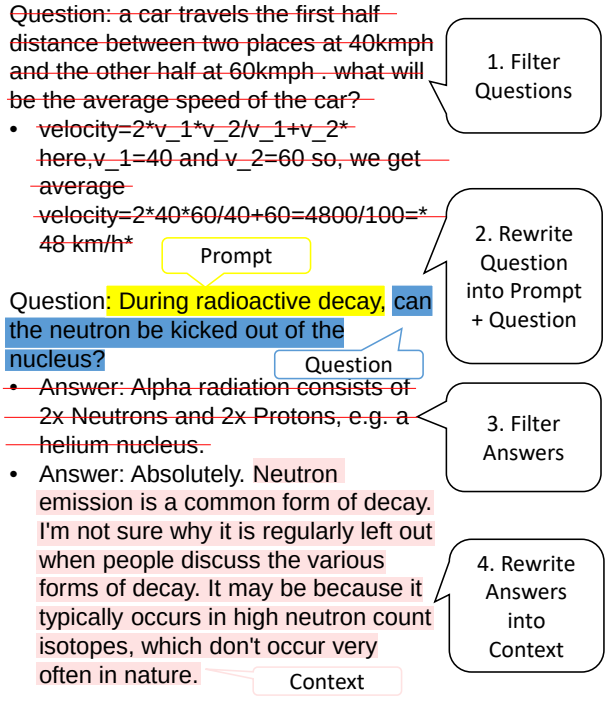

Figure 3: Data annotation pipeline, consisting of four steps: 1) Filter Questions; 2) Rewrite Question into Prompt + Question, 3) Filter Answers, and 4) Rewrite Answers into Contexts.

After the annotation, we hired two workers to validate the quality of annotation. We find that: in 140 data samples, the context cannot provide the information needed to drive the question. In 19 data samples, the prompt cannot constrain the direction of the question. In 25 data samples, The question is incomplete, incomprehensible, not even a question at all. 85% of the annotated data samples meet all the requirements, which gives us 1,034 samples to form our final KHANQ dataset.

4 Data Analysis

4.1 Dataset Statistics

The final dataset consists of 1,034 high-quality data samples, in which 515 samples come from the field of Biology, 401 from the field of chemistry, 88 from the field of physics, 19 from the field of electrical-engineering, 7 from the field of environmental-science, and 4 from the field of cosmology-and-astronomy. The average length of *Context*, *Prompt*, and *Question* are 84.62 words, 14.12 words, and 10.74 words, respectively.

4.2 Question types by words

To investigate the depth and diversity of questions in KHANQ, we classify questions based on the first two words in the question and compare them to other commonly-used question generation datasets: SQuAD 2.0 (Rajpurkar et al., 2018) and HotpotQA (Yang et al., 2018), as shown in Table 2.

KHANQ	%	SQuAD	%	HotpotQA	%
Why does	3.8	what is	8.5	What is	5.0
How does	3.6	what was	5.3	Who was	2.1
Why is	3.4	how many	4.9	What was	2.0
Does the	3.0	when did	3.1	In what	1.8
Why do	2.8	in what	2.9	When was	1.7
What is	2.8	what did	2.8	Who is	1.6
How do	2.3	when was	2.1	How many	1.0
How to	2.2	who was	2.1	In which	0.9
How is	2.2	what does	1.7	What year	0.9
Is it	2.2	what are	1.7	Are both	0.9

Table 2: Most frequent leading bigrams in SQuAD 2.0, HotpotQA and KHANQ

According to (Craig et al., 2000), questions that are represented in the form of *Why*, and *How* are likely to be deep questions. Table 2 shows that KHANQ is dominated by these questions. In contrast, SQuAD 2.0, and HotpotQA are dominated by shallow questions represented in the form of *what*.

4.3 Question types by reasoning

To gain a better insight of the characteristics of the questions, we manually analyzed a sample of 95 different questions from KHANQ. We classify these questions following the criteria in (Cao and Wang, 2021). We summarize the most common types of questions in KHANQ and their corresponding examples as follows.

PROCEDURAL questions In 21.05% of the questions we inspected, the questioners asked for the procedures, tools, or methods by which a particular outcome is achieved. Most of these questions begin with *How*, followed by an auxiliary verb, a modal verb, or *to*, which indicates that the questioner is reasoning about the steps of action.

• *How to determine the oxidation state of oxygen?*

• *How does the body know what to send down the esophagus and what to send down the trachea?*

CAUSE questions In 18.95% of the questions we inspected, the questioners are asking for the cause or reason for an event or a concept. Most of these questions begin with *Why*, followed by an auxiliary verb, a modal verb, or their negative form, which indicates that the questioner is seeking the reason behind a phenomenon.

• *Why are the electrons mapped out in such an orderly way?*

• *Why don't the oxygen atoms in the air bond to the water molecules on the surface and pull on water molecules?*

VERIFICATION questions In 15.79% of the questions we inspected, the questioners asked for the truthfulness of an event or a concept. Most of these questions are general questions that begin with be verbs, auxiliary verbs, or modal verbs. This indicates that the questioner wants to verify the truth of an idea when he or she already has a specific idea.

• *Does the oxygen bonded with another oxygen don't count as another oxidation state?*

• *Are the cranial nerves part of the CNS and the spinal nerves part of the PNS?*

CONSEQUENCE questions In 11.58% of the questions we inspected, the questioners asked for the consequences or results of an event. Most of the questions include *What happen*, *How affect* and so on. This indicates that the questioner is trying to draw a conclusion about the effects or consequences of an action.

• *What happens to the water at the bottom of the container?*

• *How will the viscosity of liquid be affected by increase in temperature?*

According to (Craig et al., 2000), six question categories involve deep reasoning: causal antecedent, causal consequence, goal-orientation, enablement, instrumental/procedural, and expectational. Connecting it to KHANQ, PROCEDURAL questions, CAUSE questions, and CONSEQUENCE questions are three categories that involve deep reasoning, accounting for 51.58%.

4.4 Prompt types

We further analyzed the prompts of these questions and divided the prompts into four major types:

- *Condition* (36.94%): *Prompt* is a conditional clause (trigger word: "if", "when", etc.);
- *Preposition* (8.32%): *Prompt* is a prepositional phrase (trigger word: "in", "for", etc.);

Context:		
Passive transport basically does not require any form of energy compared to active transport. In the case of osmosis,the water moves from areas of HIGH WATER concentration to areas of LOW WATER concentration, which makes it a form of passive transport. It uses no energy to move, it just drifts into lower concentrations of WATER. WATER not other materials, only WATER. Osmosis deals with water.		
Prompt:		**Question:**
Condition	If water moves from areas where solutes are less concentrated to areas where they are more concentrated	Why would osmosis be a form of Passive Transport?
Preposition	In the case of osmosis	How molecules move from areas of high concentration to low concentration?
Citation	Passive transport is when molecules move from areas of high concentration to low concentration	Shouldn't osmosis technically be classified as a form of active transport?
Question	Are the modes of transport that move molecules from high to low concentrations all passive transport?	Is osmosis also passive transport for water?

Table 3: Different questions raised by learners for the same context with different types of prompts.

- *Citation* (33.95%): *Prompt* is a complete sentence expressing some known information or the learner's own understanding;

- *Question* (20.79%) - *Prompt* is an initial question which leads to the followup question.

Given the same context, different prompts trigger different questions. In Table 3, we show a typical example in KHANQ in which four different types of prompts lead to different questions. We observe a strong correlation between the prompt type and the question type, which reveals how the prompt affects the question: 1) Most of the questions asked under the *condition-type prompt* aim to seek for new information based on the condition set by the questioner. The questions generally ask for causes or consequences. There is a strong logical connection between the question and the prompt, but the question rarely repeats the words in the prompt; 2) Most of the questions asked under the *preposition-type prompt* are about a specific object or phase. They generally ask for procedures or methods. These questions are general and often do not stand alone without the prompt; 3) Most of the questions asked under the *citation-type prompt* are to verify what the questioner already knows or to doubt the newly learned content based on what the learner is already known. They are mainly VERIFICATION questions. The questions tend to repeat the keywords in the prompt; 4) Most of the questions asked under the *question-type prompt* are specifications of the previous question asked by the questioner. The types of the question in *Prompt* and *Question* are usually the same.

5 Experiments

We evaluate the performance of state-of-the-art question generation models on KHANQ, focusing on the following:
• Exploring whether existing models can generate reasonable educational questions by conducting both automatic evaluation (Section 5.3) and human evaluation (Section 5.4).
• Analyzing whether the model needs to actually understand a certain type of prompts when generating questions. (Section 5.5).

5.1 Experimental Settings

We formulate the question generation task as predicting the *Question* given both *Context* and *Prompt* as inputs. Through preliminary experiments, we find both context and prompt are necessary to form a well-defined QG setting because 1) if the prompt is not given, many possible questions can be asked for the context paragraph. The model does not have any guidance on what to ask, leading to simple trivial questions in most cases; 2) if the context is not given, the problem becomes a simple language modeling task which aims to generate the missing part of an incomplete question. In our experiment, 90% of the data in KHANQ are used for training while the remaining are used for testing.

5.2 Models

We choose five generation models based on pretraining language models (PLMs) that perform well on QG tasks for evaluation. We also tried QG-specific models without pretraining such as Info-HCVAE (Lee et al., 2020), but we found that they fail to generate meaningful questions in KHANQ.

Model	BLEU1	BLEU2	BLEU3	BLEU4	METEOR	ROUGE-L
Human Baseline	31.29	16.74	10.72	7.47	15.60	30.83
BertGeneration (Rothe et al., 2020)	17.83	5.75	2.52	1.15	8.07	17.80
GPT-2 (Radford et al., 2019)	17.01	5.59	2.29	1.31	7.79	18.78
BART (Lewis et al., 2020)	25.10	11.22	6.20	3.41	12.71	26.14
Google-T5 (Raffel et al., 2020)	25.62	12.93	7.25	4.32	12.66	27.62
UniLM (Dong et al., 2019)	20.15	8.83	4.65	2.76	10.76	22.02

Table 4: Automatic evaluation results for different models in KHANQ with BLEU1-4, METEOR and ROUGE-L

	Appro.	Cov.	Coh.	Comp.
Human	**4.32**	**4.44**	**4.13**	**3.83**
BertGeneration	2.63	2.12	2.35	3.47
GPT-2	2.87	2.69	3.07	2.40
BART	4.28	3.65	3.37	3.41
Google-T5	3.90	3.74	3.31	3.75
UniLM	3.35	3.26	2.96	3.37

Table 5: Human evaluation results for different models in KHANQ on appropriateness (Appro.), coverage (Cov.), coherence (Coh.) and complexity (Comp.)

BertGeneration (Rothe et al., 2020) This is a Transformer-based sequence-to-sequence model in which the parameters of both encoder and decoder are initialized with BERT (Devlin et al., 2019). We use the implementation from Huggingface[2].

GPT-2 (Radford et al., 2019) GPT-2 is a large transformer-based language model with 1.5 billion parameters, trained on a dataset of 8 million web pages. We fine-tune GPT-2 on our training data, by formatting the input sequence as: `Context [PRT] Prompt [QUE] Question`. During test time, `Context [PRT] Prompt [QUE]` is given to predict the question.

BART (Lewis et al., 2020) BART is consists of a bidirectional encoder and a left-to-right decoder. The pretraining task involves randomly shuffling the order of the original sentences and a novel in-filling scheme, where spans of text are replaced with a single mask token. We fine-tune BART on KHANQ by predicting the question given the context and the prompt.

Google-T5 (Raffel et al., 2020) Google-T5 is another state-of-the-art language generation model which pretrains the Transformer with the fill-in-the-blank-style denoising objectives. The model is trained to recover missing words in the input. We use the `t5-base` model provided by Huggingface and fine-tune it on the training data of KHANQ.

UniLM (Dong et al., 2019) UniLM uses three types of language modeling tasks for pretraining: one-way, two-way, and sequence-to-sequence prediction. We initialize UniLM with the parameters of BERT-base (Turc et al., 2019) and then fine-tune it on KHANQ and predict the question.

5.3 Automatic Evaluation

We evaluate the generated questions using BLEU (Papineni et al., 2002), METEOR (Lavie and Agarwal, 2007) and ROUGE-L (Lin, 2004).

To compare with human performance, we recruited two high school graduates who were not involved in the annotation process and asked them to perform the same task as models. We took a sample of 80 data and asked them to generate questions based on the context and the prompt. To reduce subjective differences, they were required to develop the results after discussion.

Table 4 shows that BART and Google-T5 perform the best on KHANQ, with similar performance. BART achieves the best METEOR score, while Google-T5 achieves the best BLEU1-4 and ROUGE-L. Both these two models perform significantly better than other models. This observation is consistent with other language generation tasks in which BART and Google-T5 also show strong performance in generation. However, although BART and Google-T5 have achieved super-human performance in generating shallow questions in SQuAD, in our KHANQ dataset, all models perform worse than the human baseline in all metrics, indicating that KHANQ is a more challenging benchmark for SOTA models. In Appendix B, we show examples of generated questions by different models.

[2] https://huggingface.co/transformers/model_doc/bertgeneration.html

Model	prompt type	BLEU1	BLEU2	BLEU3	BLEU4	METEOR	ROUGE-L
BART	condition	+0.06	-0.26	-0.61	-1.17	-0.55	+0.80
	preposition	-2.33	-1.35	-1.06	-0.39	-1.56	-0.92
	citation	-0.56	-0.16	+0.09	+0.63	-0.10	+1.26
	question	-0.60	+1.43	+1.71	+1.92	-0.34	+2.36
Google-T5	condition	-1.13	-1.67	-1.29	-1.04	-1.53	-2.73
	preposition	-4.33	-4.90	-4.23	-2.96	-2.42	-2.87
	citation	-2.52	-2.45	-1.07	-0.36	-0.18	-1.69
	question	-0.36	-0.66	-0.08	+0.16	-0.96	-1.23

Table 6: Changes in the automatic evaluation scores of BART and Google-T5 when data samples from different type of prompts are used as the test set.

5.4 Human Evaluation

We conduct human evaluation on same sets of test samples used in Section 5.3. Each set consists of the human written question and five questions generated by five different models. We hired three annotators who participated in our data annotation to rate the total 480 questions with the best being a 5 and the worst being a 1 on four criteria: (1) *Appropriateness*: whether the question is semantically correct, regardless of the context and the prompt; (2) *Coverage*: whether the question is derived from the context and covers most of the information in the context; (3) *Coherent*: whether the question is coherent with the prompt; (4) *Complexity*: whether the question involves deep reasoning. We further asked them to give what the best and worst scores should be for each aspect to adjust the absolute difference between the best and worst outputs.

As shown in Table 5, BART and Google-T5 also perform better than other models in human evaluation, which is consistent with the automatic evaluation. Human reference still achieve the highest scores on all four aspects, indicating that QG models still fail to reach the human level. BART generates questions with the best appropriateness, and Google-T5 generates questions with the best complexity. In both aspects, they are very close to the human baseline. This suggests that BART and Google-T5 have the ability to ask fluent and complex questions similar to humans. However, the scores of coverage and coherence have a large gap with the human reference for all models, indicating that although the generated questions look fluent by themselves, they often do not cover the essential knowledge covered in the context or they fail to be coherent with the given prompt. This shows that although pretraining models are powerful in generating fluent-looking questions, generating questions that require a deep understanding of the context and the prompt is still challenging.

5.5 Analysis of the effects of prompt

In this section, we analyze whether the model needs to actually understand a certain type of prompts when generating questions. We take turns using data samples from one type of prompts as the test set and the other three as the training set. We choose to analyze BART and Google-T5, which perform well in the previous evaluations.

As shown in Table 6, when the training and testing sets are divided by prompt type instead of randomly, the generation effect of Google-T5 will be much worse, while the generation effect of BART will not change much, which indicates that BART has a better transfer learning ability under different prompt types than Google-T5. It also shows that the questions under different prompt types have some commonality. It is worth noting that if the training set does not contain data samples with the *preposition-type prompt*, both models perform worse when generating questions under this type of prompt. This can be attributed to the fact that *preposition-type prompt* tends to contain very limited amount of information, and most of the questions under those prompts are in-depth questions asking about processes or methods. Generating such questions requires a very accurate understanding of the prompts, which cannot be achieved by a model that has not been trained with these samples.

6 Conclusions and Future Works

In this paper, we propose KHANQ, a dataset for generating in-depth educational questions. Each sample in KHANQ is carefully annotated as *Context*, *Prompt*, and *Question* to form a clean dataset. We evaluate the performance of state-of-the-art

question generation models on KHANQ. We find that although it is feasible for the model to generate fluent and complex questions, the ability to understand and reason over the context and the prompt is still far from reaching the human level.

There are several future directions that are worth investigating: (1) what different results the models will produce in terms of different types of questions; (2) how to enable models to obtain information from the context and the prompt separately and then make the inference, to enhance their ability to seek information under the given conditions.

References

Seohyun Back, Akhil Kedia, Sai Chetan Chinthakindi, Haejun Lee, and Jaegul Choo. 2021. Learning to generate questions by learning to recover answer-containing sentences. In *Findings of the Association for Computational Linguistics (ACL/IJCNLP)*, pages 1516–1529.

Shuyang Cao and Lu Wang. 2021. Controllable open-ended question generation with a new question type ontology. In *ACL/IJCNLP (1)*, pages 6424–6439.

Yllias Chali and Sadid A. Hasan. 2015. Towards topic-to-question generation. *Computational Linguistics*, 41(1):1–20.

Guanliang Chen, Jie Yang, Claudia Hauff, and Geert-Jan Houben. 2018. Learningq: A large-scale dataset for educational question generation. In *ICWSM*.

S. D. Craig, B. Gholson, M. Ventura, and A. C. Graesser. 2000. Overhearing dialogues and monologues in virtual tutoring sessions: Effects on questioning and vicarious learning. *International Journal of Artificial Intelligence in Education*, 11:242–253.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: pre-training of deep bidirectional transformers for language understanding. In *Annual Conference of the North American Chapter of the Association for Computational Linguistics (NAACL-HLT)*, pages 4171–4186.

Li Dong, Nan Yang, Wenhui Wang, Furu Wei, Xiaodong Liu, Yu Wang, Jianfeng Gao, Ming Zhou, and Hsiao-Wuen Hon. 2019. Unified language model pre-training for natural language understanding and generation. *CoRR*, abs/1905.03197.

Xinya Du, Junru Shao, and Claire Cardie. 2017. Learning to ask: Neural question generation for reading comprehension. In *Annual Meeting of the Association for Computational Linguistics (ACL)*, pages 1342–1352.

Nan Duan, Duyu Tang, Peng Chen, and Ming Zhou. 2017. Question generation for question answering. In *Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 866–874.

Mansi Gupta, Nitish Kulkarni, Raghuveer Chanda, Anirudha Rayasam, and Zachary C. Lipton. 2019. Amazonqa: A review-based question answering task. *CoRR*, abs/1908.04364.

Michael Heilman and Noah A. Smith. 2010. Good question! statistical ranking for question generation. In *Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, pages 609–617, Los Angeles, California. Association for Computational Linguistics.

Kalpesh Krishna and Mohit Iyyer. 2019. Generating question-answer hierarchies. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 2321–2334, Florence, Italy. Association for Computational Linguistics.

Alon Lavie and Abhaya Agarwal. 2007. METEOR: An automatic metric for MT evaluation with high levels of correlation with human judgments. In *Proceedings of the Second Workshop on Statistical Machine Translation*, pages 228–231, Prague, Czech Republic. Association for Computational Linguistics.

Dong Bok Lee, Seanie Lee, Woo Tae Jeong, Donghwan Kim, and Sung Ju Hwang. 2020. Generating diverse and consistent QA pairs from contexts with information-maximizing hierarchical conditional VAEs. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 208–224, Online. Association for Computational Linguistics.

Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Veselin Stoyanov, and Luke Zettlemoyer. 2020. BART: denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. In *Annual Meeting of the Association for Computational Linguistics (ACL)*, pages 7871–7880.

Chin-Yew Lin. 2004. ROUGE: A package for automatic evaluation of summaries. In *Text Summarization Branches Out*, pages 74–81, Barcelona, Spain. Association for Computational Linguistics.

Jose Otero and Arthur C Graesser. 2001. Preg: Elements of a model of question asking. *Cognition and instruction*, 19(2):143–175.

Liangming Pan, Wenhu Chen, Wenhan Xiong, Min-Yen Kan, and William Yang Wang. 2021. Unsupervised multi-hop question answering by question generation. In *Annual Conference of the North American Chapter of the Association for Computational Linguistics (NAACL-HLT)*, pages 5866–5880.

Liangming Pan, Yuxi Xie, Yansong Feng, Tat-Seng Chua, and Min-Yen Kan. 2020. Semantic graphs

for generating deep questions. In *Annual Meeting of the Association for Computational Linguistics (ACL)*, pages 1463–1475.

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. Bleu: A method for automatic evaluation of machine translation. In *Proceedings of the 40th Annual Meeting on Association for Computational Linguistics*, ACL '02, page 311–318, USA. Association for Computational Linguistics.

Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. 2019. Language models are unsupervised multitask learners. *OpenAI blog*, 1(8):9.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J. Liu. 2020. Exploring the limits of transfer learning with a unified text-to-text transformer. *Journal of Machine Learning Research*, 21:140:1–140:67.

Pranav Rajpurkar, Robin Jia, and Percy Liang. 2018. Know what you don't know: Unanswerable questions for squad. *CoRR*, abs/1806.03822.

Pranav Rajpurkar, Jian Zhang, Konstantin Lopyrev, and Percy Liang. 2016. Squad: 100, 000+ questions for machine comprehension of text. In *EMNLP*, pages 2383–2392.

Sascha Rothe, Shashi Narayan, and Aliaksei Severyn. 2020. Leveraging pre-trained checkpoints for sequence generation tasks. *Transactions of the Association for Computational Linguistics*, 8:264–280.

Xingwu Sun, Jing Liu, Yajuan Lyu, Wei He, Yanjun Ma, and Shi Wang. 2018. Answer-focused and position-aware neural question generation. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 3930–3939, Brussels, Belgium. Association for Computational Linguistics.

Luu Anh Tuan, Darsh J. Shah, and Regina Barzilay. 2020. Capturing greater context for question generation. In *AAAI Conference on Artificial Intelligence (AAAI)*.

Iulia Turc, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. Well-read students learn better: On the importance of pre-training compact models. *arXiv preprint arXiv:1908.08962v2*.

Yuxi Xie, Liangming Pan, Dongzhe Wang, Min-Yen Kan, and Yansong Feng. 2020. Exploring question-specific rewards for generating deep questions. In *International Conference on Computational Linguistics (COLING)*, pages 2534–2546.

Zhilin Yang, Peng Qi, Saizheng Zhang, Yoshua Bengio, William W. Cohen, Ruslan Salakhutdinov, and Christopher D. Manning. 2018. Hotpotqa: A dataset for diverse, explainable multi-hop question answering. In *Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 2369–2380.

Jianxing Yu, Xiaojun Quan, Qinliang Su, and Jian Yin. 2020. Generating multi-hop reasoning questions to improve machine reading comprehension. In *International World Wide Web Conference (WWW)*, pages 281–291.

Yao Zhao, Xiaochuan Ni, Yuanyuan Ding, and Qifa Ke. 2018. Paragraph-level neural question generation with maxout pointer and gated self-attention networks. In *Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 3901–3910.

Wenjie Zhou, Minghua Zhang, and Yunfang Wu. 2019. Question-type driven question generation. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 6032–6037, Hong Kong, China. Association for Computational Linguistics.

A Data Annotation Procedure

A.1 Filter Questions

Purpose: Questions should: 1. be deep enough; 2. contain no external information that requires a deep query; 3. be able to be rewritten as prompt + question.

Filtering out:

- **Question cannot be rewritten as prompt + question**
  Question: what about names from 16-19?
- **Questions that require mathematical calculations but do not provide formulas** (trigger word: number, units)
  Question: a car travels the first half distance between two places at 40kmph and the other half at 60kmph. what will be the average speed of the car?
- **Questions that cite unavailable timestamp or figure without specifying the corresponding text** (trigger word: "at 0:40", "in figure 1 below")
  Question: When the instructor refers to "Lesser Apes" at 0:40, what characteristics classify these "Lesser Apes"?
- **Simple questions that ask for definitions** (trigger word: "What is")
  Question: what is fasciculus
- **Questions that are not understood by the answerer** (trigger word: "Do you mean" in the answer)

Question: how do i know whether specific molecules will undergo active or passive diffusion by just looking at the molecule?
Answer: There is no 'active' diffusion. Diffusion is passive transport. Do you mean diffusion or facilitated diffusion? It depends what 'liquid' is moving through your semi-permeable membrane.

A.2 Rewrite Question into Prompt + Question

Purpose: Generate Prompt and Question
* Highlighted part is Prompt, other part is Question

- **The original question is a conditional clause + question, and the conditional clause can be used as prompt** (trigger word: "if", "when")
  If there is bacteria in our blood and there is only 1% of white blood cells, wouldn't that take a long time to dispose of the bacteria?

- **The original question is prepositional phrase + question, and prepositional phrase can be used as prompt** (trigger word: "in", "on", "for")
  Does binary fission occur in the same way for ALL bacteria?

- **A sentence that quotes a part of the article can be used as a prompt** (trigger word: "in . . . th paragraph", "in . . . th section")
  In the first section you mention a graph of cyclin levels over the expression cycle throughout mitosis. Why is G1 Cyclin required throughout the entire cyclin expression cycle of mitosis?

- **Questions that cite their own knowledge, the cited knowledge can be used as prompt** (trigger word: "I understand", "I thought", "I know")
  It took me a while to figure out that DNA isn't just one molecule, but a collection of molecules, one per chromosome in humans. Why do people still call DNA a molecule?

- **The question consists of multiple sentences, using the preceding declarative sentence as the prompt** (trigger word: "." ",")
  There are four phases in one cell cycle (G1, S, G2 and M). Apoptosis occurs in which phase?

Special cases:

- **If there are multiple consecutive questions** (trigger word: ?) **First generate Prompt and Question for each question according to the above standards, then if the later question is asked on the basis of the earlier question, the earlier question is used as the Prompt of the later question**
  Is autism a genetic disorder? If so, which chromosome determines the mutation?
  ⇒ (The first question cannot be written as prompt + question, filter out. There is no question before, no further processing. The second question cannot be written as prompt + question, filter out. Further processing:)
  Prompt: Is autism a genetic disorder? If so.
  Question: which chromosome determines the mutation?

- **For the question that can generate multiple Prompt, combine multiple Prompts into one Prompt**
  how bonds require energy in order to be broken and vice versa, why is it opposite for ATP bonds? Because when ATP bonds are broken, energy is released
  ⇒
  Prompt: Because when ATP bonds are broken, energy is released, how bonds require energy in order to be broken and vice versa,
  Question: why is it opposite for ATP bonds?

- **For the choice question** (trigger word: "or") **if one option does not contain information, delete it. if the two choices contain different information, split it into two questions**
  During radioactive decay, can the neutron be kicked out of the nucleus? Or is it only the proton which gets kicked out?
  ⇒
  1 Prompt: During radioactive decay,
  Question: can the neutron be kicked out of the nucleus?
  2 Prompt: During radioactive decay,
  Question: is it only the proton which gets kicked out?

A.3 Filter Answers

Purpose: Answers should: 1. be able to be rewritten into independent contexts; 2. contains enough information to ask the corresponding questions.
Filtering out:

- **Answers contains too little information**
  Answer: The H zone is the space in the middle of the sarcomere where only myosin proteins are found.

- **Answers that cannot stand alone as a paragraph**
  Answer: Experimentation. It is not determined from an equation. In a similar way to how you cannot solve for the specific heat of a substance, you can conduct and experiment to find it, or use an accepted value from a table.

- **Answers that cannot answer the question**
  Question: I thought an energy-releasing reaction was called an exothermic reaction and a reaction that takes in energy is endothermic. In the article, it defines them as exergonic and endergonic. Are they the same?
  Answer: Exothermic and endothermic refer to specifically heat. Exergonic and endergonic refer to energy in general.

- **Answers including background knowledge that are not mentioned**
  Answer: Not all ions are reactive (think of dissolving salt in water to give Na+ and Cl- ions) - it depends on the circumstances. H+ ions are more reactive than H3O+ ions, so when an acid dissociates in the water, the protons immediately latch on to water molecules to give H3O+ ions which are more stable than H+ ions.

Special cases:

- **If a usable question has multiple answers, first filter according to the above standards. For the filtered-out answers, first join them together and keep the joined answers if they meet the requirements after joining. Finally, each kept answer can be used as a context for a data**
  Answer: The clouds keep our temperature regulated. If we didn't have clouds, it would be extremely hot at night and extremely cold at night. Aren't you glad God created clouds? He really thought it out well when he created the earth. Hope this helps. (The answer is not good enough, but a good context can be generated after joined)
  Answer: It would be both hot and cold (bit like the moon). When the planet faces the sun, it would be really hot and when it faces away from the sun, it would be really cold. (The answer is not good enough, but a good context can be generated after joined)
  Answer: If you just mean a place hardly has clouds. I think hotter? For there are no raining. Whatever, the climate will become very hard. But if you mean that there wasn't a state called cloud, then I don't know.... What do you actually mean about no clouds? Because clouds are the basic state for water. If there aren't any clouds, what will water become? (The answer is not good enough, contains too much information that is invalid or even contradictory to other answers, and a good context cannot be generated after splicing, so discard it directly)
  ⇒
  Answer: The clouds keep our temperature regulated. If we didn't have clouds, it would be extremely hot at night and extremely cold at night. ~~Aren't you glad God created clouds? He really thought it out well when he created the earth. Hope this helps. Answer:~~ It would be both hot and cold (bit like the moon). When the planet faces the sun, It would be really hot and when it faces away from the sun, It would be really cold.

A.4 Rewrite Answers into Contexts

Purpose: Generate Contexts
*Before deletion is the answer, after deletion is the context

- **Remove answer-tone phrases for the question** (trigger word: "yes", "no", "short answer:" , "this is because")
  Answer: ~~Short answer:~~ a photon is a particle of light. ~~Longer answer:~~ light is energy. Sometimes we think of light as being a wave in the form of an electro-magnetic wave but other times it can be described as a particle. A photon in this case, is 1 unit of light with a variable amount of energy which depends on its frequency.

- **Remove external information from the subject irrelevant(need for deep query) content** (trigger word: Url, email, phone, etc)
  *Answer: *Oncogene* are mutated genes that switch from G1 to S phase, (even when there is no need for cell division). So they are accečerators of the process. You know*

*that is a checkpoint of a cell. However, there are also *tumor suppressor* genes and they act a brake. So whilst the tumor suppressor is sitting there in the cell, it's stopping the cell from going around the cell cycle. If we, again, trigger the cell cycle or attempt to by instructing the cell with a signal initiating the relay, one of the jobs of that relay is to remove that block, that brake. As long as a tumor suppressor is working, cancer will not arise. If you remove tumor suppressor (mutation) the cell is free to move from G1 to S phase. If they are both in mutant form, cancer arises. ~~https://www.ncbi.nlm.nih.gov/books/NBK21662/ https://www.futurelearn.com/courses/inside-cancer/14/steps/579660a~~*

- **Remove questions that were asked but not answered** (trigger word: "?")
  Answer: DNA is DNA it is universal in all organisms. However, combinations of nucleotides (codons) are different and code for different amino acids. ~~Why do you think it should be tested in other organisms?! Sequencing whole genomes have already been done. If it is not enough, what it is?~~

A.5 Paraphrasing

Purpose: 1. Delete words that do not contain information; 2. Rewrite to formal tone.

- **Fix grammar and spelling problems**
  Mathematically, sphere or ~~ana~~ circle has more area compared to other geometric shapes. so, ~~we cant consider~~can we consider neopentane as spherical?*
- **Delete daily terms** (trigger word: "okay")
  ~~Okay,~~where do the single protons, the hydrogens come from? How do we add them to our equation?
- **Delete the conversational language between the questioner and the answerer** (trigger word: "Hello", "good question", "hope it helps", "sincerely", "remember")
  ~~Remember,~~ velocity and acceleration are vector quantities, which have both magnitude and direction (+/-). ~~Hope that helps~~
- **Delete the conjunctions that appear at the beginning of the sentence** (trigger word: "So", "But")
  ~~so~~ we evolve from apes, so if one mated with a human could you see the "half-way" of human evolution?
- **Delete subjective prompt words** (trigger word: "I believe")
  ~~I thought~~ red blood cells are already larger in diameter than the smallest capillaries, so that they have to squeeze through.
- **Delete quote cues and quotation marks** (trigger word: "From the video, it says that")
  ~~In the third paragraph under "Ionic Bonds", it says that~~ there is no such thing as a single NaCl molecule. Why can't you have a single molecule of NaCl?
- **Delete timestamp** (trigger word: "At 3:40")
  ~~At 3:40, the speaker says that~~ enthalpy of a reaction can be calculated using Hess's law or by using the Delta H of formation. I thought Hess's law uses the Delta H of formation, but she seems to be saying those are two different ways. is there a difference between using Hess's law and using the Delta H of formation, and if so, what is it?
- **Revise the personal pronouns referring to the questioner or answerer** (trigger word: "you", "I")
  How ~~can I~~to figure out the extent of branching for a molecule?
- **If the specific noun referred to by the pronoun is in another part, change the pronoun to the specific noun it refers to** (trigger word: "it", "they", "this", "that")
  Condition: isn't there a sulfur cycle as well as all the others and if so
  Question: how does ~~it~~a sulfur cycle as well as all the others work
- **Rewrite the symbols that will be recognized as garbled** (trigger word: "°C")
  Having its boiling point at ~~°C~~ degree Celsius?

A.6 Mark source

Purpose: source should: 1. contains enough information to identify the origin of the data; 2. unaffected by website URL changes.

Use the title of the text as the source

B Case study

Context:
Superfluid means the substance is at the point between liquid and gas (equilibrium)at very high temperature and pressure. helium becomes a superfluid at extremely low temperatures. superfluid is strange in the way that it seems to have a viscosity of zero, it can flow up the sides of a container, and many other creepy things.
Prompt:
helium can become a 'superfluid'. while helium in its liquid state

Question:
Human ⇒ Under what conditions does helium become superfluid?

BertGeneration ⇒ why does hydrogen *need to dissolve in water*?
GPT-2 ⇒ helium can never make a superfluid. Does the *helium in its liquid state always have viscosity of zero*?
BART ⇒ Why does helium become a 'superfluid'?
Google-T5 ⇒ How can helium become a superfluid?
UniLM ⇒ how can helium become a superfluid?

Context:
The efficiency of the respiratory system is very very efficient. Imagine a system of pipes; one pulls the air, another transfers it. Since pipes are rounded and therefore 'closed' air cannot escape through the piping. Take for example your hand. Curl it, and blow through it. Almost none or if any air escapes, except through the other side. Because of this, the only path for the air to take is to the lungs. In which the lungs work/use the air and the entire cycle happens again. When a biological system is working many other dependent systems are working too.
Prompt:
As the air keeps moving down the throat to the lungs, does some air leak or escape?

Question:
Human ⇒ A. Where is the path of air passage through breathing space and where the working place is?

BertGeneration ⇒ Does something like *a hydroxylation* change the oxygen species?
GPT-2 ⇒ how does the oxygen diffuse in or would it go in or can the same pressure?
BART ⇒ Why does the respiratory system work?
Google-T5 ⇒ Why is the respiratory system so efficient?
UniLM ⇒ how does the air move in a system?

Table 7: Examples of questions generated on KHANQ: 1) States of matter 2) The lungs and pulmonary system. Colored Text indicates key words and *Italics* indicate inappropriate words

Multi-Figurative Language Generation

Huiyuan Lai and **Malvina Nissim**
Center for Language and Cognition (CLCG)
University of Groningen / The Netherlands
{h.lai, m.nissim}@rug.nl

Abstract

Figurative language generation is the task of reformulating a given text in the desired figure of speech while still being faithful to the original context. We take the first step towards multi-figurative language modelling by providing a benchmark for the automatic generation of five common figurative forms in English. We train **mFLAG** employing a scheme for multi-figurative language pre-training on top of BART, and a mechanism for injecting the target figurative information into the encoder; this enables the generation of text with the target figurative form from another figurative form without parallel figurative-figurative sentence pairs. Our approach outperforms all strong baselines. We also offer some qualitative analysis and reflections on the relationship between the different figures of speech.

1 Introduction

Figurative language is commonly used in speaking and writing to accomplish a constellation of communicative goals (Roberts and Kreuz, 1994). Figures of speech, such as metaphors, or idiomatic expressions, can make an expression stand out by making it more interesting and captivating, and can evoke stronger emotions than more factual, literal phrases thereby making the text more engaging.

Automatic figurative language generation has received growing attention with the progress of neural networks, especially the emergence of large pre-trained models (Raffel et al., 2020; Lewis et al., 2020). We see there are two core values for this task: (i) computational approaches can be employed to provide a better understanding of linguistic phenomena and more specifically in this case different figures of speech; (ii) we can explore how much models can handle creativity and devise ways to employ them in the support of creative writing, so as to yield more varied and human-like generated text, including in the context of machine translation (Guerberof-Arenas and Toral, 2022).

Forms	Sentences
Literal	Old Mr. Smith has been teaching here for a very long time.
Hyperbole	Old Mr. Smith has been teaching here since the Stone Age.
Literal	My niece will babysit for you for a little bit of money.
Idiom	My niece will babysit for you for pin money.
Literal	I hate it when they run the same commercial twice in a row.
Sarcasm	I love when they run the same commercial twice in a row.
Literal	He remembers a road of my broken works.
Metaphor	He made a road of my broken works.
Literal	You can publish the whole thing old.
Simile	You can publish the whole thing like a diary.

Table 1: Examples of figurative language generation from literal texts.

There are many related tasks that have been proposed and studied by NLP researchers, including the generation of hyperbole (Tian et al., 2021; Zhang and Wan, 2022), idiom (Zhou et al., 2021), sarcasm (Zhu et al., 2019; Chakrabarty et al., 2020a), metaphor (Abe et al., 2006; Stowe et al., 2021b), and simile (Chakrabarty et al., 2020b; Zhang et al., 2021). Table 1 shows examples of figurative language generation from literal texts.

Previous works focus on modelling single figurative forms, generally rewriting a literal sentence into one with a specific figure of speech. This results in having to train separate models, one for each figure of speech, and in not exploiting knowledge transfer across figurative forms. However, since different figures of speech can share some features related to non-literality, and a text may also contain and combine multiple figures of speech at the same time, it is possible that substantial knowledge gains can be transferred from one figure to another. Moreover, the generation between different figures of speech (e.g. generating an idiomatic text from the hyperbolic one) is under-explored.

In this work we suggest to model multiple figures of speech jointly, with the ultimate goal of having a single model that can handle the generation of multiple figurative forms from both literal and figurative inputs.

Intuitively, multi-task learning (Collobert and Weston, 2008) and the usage of a domain la-

Proceedings of the 29th International Conference on Computational Linguistics, pages 5939–5954
October 12–17, 2022.

bel (Kobus et al., 2017) could be a good method for multi-figurative language modelling, adding a special token to the beginning of the sentence to guide text generation. Such a method requires parallel data (i.e. aligned texts with the same context but different figures of speech) for training; this is usually unavailable, especially between different figures of speech, and costly to produce.

We rely on existing parallel data between literal sentences and single figures of speech and propose mFLAG (**M**ulti-**F**igurative **La**nguage **G**eneration), an approach which is applicable to the generation between different forms, both literal and figurative. In a nutshell, mFLAG is trained in two stages, in both of which we also exploit the contribution of generic paraphrase data: (i) a specifically designed pre-training for multi-figurative language, where a special label is added at the beginning of each sentence to indicate its figure of speech; (ii) a supervised training where the parallel literal-figurative sentence pairs for all figurative languages are combined to achieve multi-figurative language generation. For (ii), we introduce an innovative mechanism that allows the form labels to leak their own figurative information into the input embedding, thus guiding the encoder to represent the source sentence. This mechanism makes it possible to generate between different figures of speech without parallel figurative-figurative data. For comparison, and to allow for wider flexibility in generation choices as well as linguistic analysis, we also use the literal form corresponding to each figure of speech, which is available through the separate parallel datasets, as pivot to run figurative-to-figurative transformation. We expect that with the direct figurative-figurative transformation the source figurative form might still be maintained in the generated sentence, with the addition of the target figurative form, while this should not be the case when using the literal form as pivot.

Contributions Considering five common figures of speech in English, (i) we propose a novel task of multi-figurative language generation, and explore the potential of its computational modelling; (ii) we introduce a pre-training scheme for multi-figurative language modelling, which boosts performance substantially by leveraging paraphrase data and cross-figurative language knowledge transfer; (iii) we design a mechanism for injecting the desired figurative information into the encoder to achieve the generation between different figures of speech without parallel figurative-figurative sentence pairs; this mechanisms could be applied to other tasks, too; (iv) we compare figurative-figurative and figurative-literal-figurative generation, thereby assessing the feasibility, the limits, and the characteristics of direct multi-figurative language generation; and (v) we provide a benchmark for multi-figurative language generation, which can hopefully foster the progress of figurative language processing. [1]

2 Background

Transforming text involving a figure of speech, either in source or in target or both, is closely related to three other NLP tasks, namely paraphrasing, text style transfer, and figurative language detection. We discuss relevant background on such tasks, and why and how they play a role in our work.

Paraphrasing Paraphrasing is the task of generating a text semantically (almost) identical to a given input, but with variations in wording or syntax (Prakash et al., 2016; Cao et al., 2017). The large amount of parallel paraphrase data available can be used to teach models a general rewriting task in the context of various downstream NLP tasks, such as semantic parsing (Berant and Liang, 2014), machine translation (Callison-Burch et al., 2006), question answering (Dong et al., 2017), and text style transfer (Lai et al., 2021). As figurative generation can be viewed as a special paraphrasing task, where texts are expected to include specific figurative forms, we also leverage paraphrase data for figurative generation modelling.

Text Style Transfer The goal of text style transfer is to transform a given text of one style into another while preserving the style-independent content. A common task, for example, is formality transfer, where an informal sentence is turned into formal, or viceversa (Rao and Tetreault, 2018). Generally speaking, both text style transfer and figurative language generation aim to achieve the generation of text with specific attributes. Regarding sentence changes, for text style transfer, often multiple parts of the sentence might be modified at the same time, such as capitalization at the beginning of the sentence, punctuation at the end, and some phrasing in the middle. Figurative language generation, instead, often concerns the rewriting of some specific expressions, while other (possi-

[1] Data, code, and model are available at https://github.com/laihuiyuan/mflag.

bly large) portions of the input sentence could be retained (Zhou et al., 2021). Also, in figurative language generation, the original figurative form could be still present in the transformed sentence, while text style transfer aims to alter the original style fully.

It should also be pointed out that addressing multi-figurative language generation is particularly challenging since not all figures of speech considered require the same kinds of alterations in text.

Figurative Language Detection Most past work on figurative language processing focuses on detection rather than generation. The detection of figurative language generally involves two levels: sentence-level and word-level. At sentence-level, the task is usually formulated as a binary classification problem, namely automatically detecting whether a given sentence is literal or non-literal (Troiano et al., 2018). At word-level, the task is concerned with identifying the exact words within a sentence which trigger the figurative reading (Beigman Klebanov et al., 2016; Mao et al., 2018). This task is a crucial component in retrieval-based approaches to figurative language generation, which usually require first the identification of triggering words in a sentence, followed then by other operations such as replacement and generation (see next paragraph).

Figurative Language Generation Early work on figurative language generation is mainly template-based. Abe et al. (2006) employ simple expressions "A is like B" for metaphor generation. Veale (2016) use template-like structures to generate metaphoric tweets. These methods usually lack the flexibility to cope with the variability intrinsic to (creative) natural language. In recent years, figurative language modelling has mostly shifted to neural-based end-to-end approaches, showing good degrees of creativity, for example in the generation of puns and metaphors (Yu et al., 2018; Yu and Wan, 2019). To provide better explainability, Zhou et al. (2021) propose a neural-based pipeline for idiom generation that contains three explicit steps: retrieve, extract, and generate. Most recently, and as in most NLP tasks, impressive results for figurative language generation have been achieved leveraging pre-trained models. For example, Stowe et al. (2021a) and Chakrabarty et al. (2021) successfully generate metaphors fine-tuning T5 (Raffel et al., 2020) and BART (Lewis et al., 2020), respectively. Fine-tuning BART is successful for the generation of simile (Chakrabarty et al., 2020b), and hyperbole (Zhang and Wan, 2022), too. Stowe et al. (2021b) also propose to control the metaphor generation process by encoding conceptual mappings in the form of FrameNet frames. All these works focus on single figurative forms, modelling generation between literal and figurative. Instead, while still leveraging parallel literal-figurative data for single forms, we aim to model multiple figures of speech jointly thereby also generative between different figurative forms.

Forms	Task	Train	Valid	Test
Hyperbole	Literal Form↔Hyperbole	509(+668)	50	150
Idiom	Literal Form↔Idiom	3,784	876	876
Sarcasm	Literal Form↔Sarcasm	16,762	1,500	1,470
Metaphor	Literal Form↔Metaphor	118,807	6,254	150
Simile	Literal Form↔Simile	82,687	5,145	150

Table 2: Dataset statistics.

3 Task and Dataset

We define the task of figurative language generation as the transformation of a text written in (or with) a given form (literal or figurative) to a text in (or containing) another form, while preserving the original general context.

We use five existing datasets for the figures of speech we consider in this paper; Table 2 shows sizes and splits.

- **Hyperbole** Troiano et al. (2018) introduce HYPO, a corpus of 709 hyperbolic sentences with their non-hyperbolic formulations. We boost this small dataset with some automatically obtained pairs. We fine-tune BART with HYPO, and use this model to transform into literal the hyperbolic texts contained in the non-parallel dataset HYPO-Red (Tian et al., 2021). We then select literal generations with a low hyperbolic score σ as predicted by a binary classifier based on BERT (Devlin et al., 2019) trained on HYPO, for an additional 668 training pairs.[2]
- **Idiom** Zhou et al. (2021) use the existing MAGPIE corpus (Haagsma et al., 2020) to create a parallel dataset of literal and idiomatic pairs.
- **Sarcasm** Peled and Reichart (2017) release a dataset of 3,000 pairs of sarcastic tweets each augmented with five interpretations. We complement this by adding to the training set 4,762

[2]Generated literal texts with $\sigma < 0.5$ are selected.

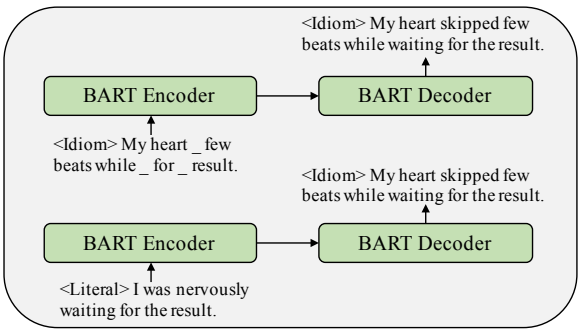

(a) Multi-figurative language denoising pre-training and fine-tuning.

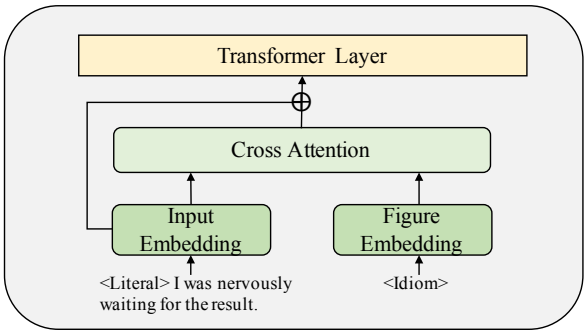

(b) An overview of the mechanism for injecting the figurative information into the Encoder.

Figure 1: Overview of multi-figurative language modelling. In 1(a), there is the framework for our multi-figurative language denoising pre-training (top) where word masking as the injected noise, and fine-tuning on downstream task of figurative language generation (down); in 1(b), the figurative information is injected into the encoder using cross-attention and residual learning.

sentence pairs from a sarcasm dataset (Ghosh et al., 2020).

- **Metaphor** Stowe et al. (2021b) build a literal-metaphor dataset exploiting the Gutenberg Poetry corpus (Jacobs, 2018): metaphoric verbs are identified, masked, and eventually replaced with infilling from a language model.
- **Simile** Chakrabarty et al. (2020b) automatically collect a set of self-labelled similes via distant supervision, using the phrase *like a*; similes are converted into their literal versions leveraging the structured common sense knowledge obtained from COMET (Bosselut et al., 2019).

Pre-Training Data Given that figurative generation is a special paraphrasing task, we use the available paraphrase data from PARABANK 2 (Hu et al., 2019) for multi-figurative language modelling, but only selecting more relevant pairs for the pre-training phase. To do so, we fine-tune BERT with the above figurative data to obtain five binary classifiers (each one literal vs figurative). With them, we do figurative language detection on paraphrase data, and only retain pairs where the probability that the source and target sentences are in literal form and figurative form, respectively, is greater than a threshold σ.[3]

4 Multi-figurative Language Modelling

We propose an approach to model multi-figurative language on top of the large pre-trained sequence-to-sequence model BART (Lewis et al., 2020), by performing further, figurative language-specific pre-training, and then fine-tuning.

BART is a seq2seq model trained as a denoising autoencoder, and to reconstruct the original text T given $g(T)$ where g is a noising function that is used to corrupt text:

$$L_\theta = -\sum \log(T \mid g(T); \theta) \quad (1)$$

with θ being the parameters of BART.

4.1 Multi-figurative Language Pre-training

We further pre-train BART for multi-figurative language modelling with a procedure that creates one model capable of modelling multiple figurative languages at once, so that (i) only one model needs to be maintained, and (ii) the model can benefit from cross-figurative knowledge transfer.

Inspired by Tang et al. (2020), we use a special token as a prefix in both the source and target text. That is, the text format is [form code] T [eos] with T being the text and the [form code] represents the form of the sentence. In the pre-training stage, we incorporate all the pre-training data of five figures of speech (Section 3) by concatenating data: $D = \{D_1, ..., D_i\}$ where each D_i is a collection of texts in a figurative form. Following Liu et al. (2020), our model is trained on a denoising task, where it is asked to reconstruct text from a version corrupted with a noise function that randomly masks 35% of the words in the sentence. The [form code] is used as the initial token to predict the sentence (Figure 1(a) (top)).

[3] More details about the pre-training data for each figure of speech are in Appendix A.1.

4.2 Literal↔Figurative Form Generation

In *Literal↔Figurative* generation, the model generates a text with the desired figure of speech given a literal text, or viceversa. First, following Lai et al. (2021), we use the parallel paraphrase pre-training data to make the model learn the basic task of rewriting. In practice, we incorporate all the data and add the corresponding form code at the beginning of each sentence to train the model in a supervised regime. Second, we fine-tune the model with the literal↔figurative parallel data (Table 2) in the same way (PT-to-FT; Figure 1(a) (down)). Since hyperbole and idiom datasets are too small, we upsample them by replication obtaining training sets of 10,000 sentence pairs.

4.3 Figurative↔Figurative Form Generation

In *Figurative↔Figurative* generation, the model takes a text with a given figurative form, and generates a text with the target figurative form. It is important to note that this procedure can have two outcomes: the target figure of speech *substitutes* the original one, or it is *added* to it, yielding a text that contains both the original and the target figurative forms.

Specifically, given a sentence of tokens $\boldsymbol{x} = \{x_1, \cdots, x_n\}$ with the figure of speech s, the model is asked to generate the corresponding sequence $\boldsymbol{y} = \{y_1, \cdots, y_m\}$ with the target figure of speech t. To overcome the lack of parallel data in different figures of speech which would be necessary to train such a model, we design a mechanism which can leak the information of the desired figure of speech to the encoder with a figurative embedding as additional input. Formally, we employ cross attention to inject the figurative information into word embedding of the input in the fine-tuning process (mFLAG; Figure 1(b)).

$$\text{CrossAttn}(\mathbf{W}, \mathbf{F}) = \text{softmax}(\frac{\mathbf{W}\mathbf{F}^{\text{T}}}{\sqrt{d}})\mathbf{F} \quad (2)$$

where $\mathbf{W} \in \mathbb{R}^{m \times d}$ represents the embedding of the source sentence. $\mathbf{F} \in \mathbb{R}^{1 \times d}$ is the embedding of the target form code T. To avoid introducing new parameters and catastrophic forgetting, we do not use the commonly used feed-forward block here. We also employ a residual connection (He et al., 2016) for the word embedding:

$$\mathbf{C} = \text{CrossAttn}(\mathbf{W}, \mathbf{F}) + \mathbf{W} \quad (3)$$

Forms	Precision Score	Recall Score	F1 Score
Hyperbole	0.858	0.967	0.909
Idiom	0.897	0.961	0.928
Sarcasm	0.763	0.847	0.803
Metaphor	0.716	0.707	0.711
Simile	1.000	0.700	0.824

Table 3: Accuracy of classifiers for different forms.

The probability of the output can be computed conditioned both on the input sentence $\boldsymbol{x}$ and the target form code T. It can be formulated as:

$$p_\theta(\boldsymbol{y}|\boldsymbol{x}, T) = \prod_{t=1}^{m} p_\theta(y_t|y_{1,\ldots,t-1}; \mathbf{C})) \quad (4)$$

We also first use the pre-training data to enhance model's rewriting ability, and employ upsampling to augment the gold training data for hyperbole and idiom. We use two settings for generation: (i) the model generates text in the target form directly from the source form (mFLAG-DR), meaning that direct figurative-figurative transformation is achieved; (ii) the model uses literal forms as pivot: it first transforms the source text back into its literal form, and then uses this obtained literal form to generate in the target form (mFLAG-BT). Comparing these two models will contribute to better understand the benefits of modelling multi-figurative language generation directly.

5 Experiments

All experiments are implemented atop Transformers (Wolf et al., 2020) using BART-large (Lewis et al., 2020). We train models with batch size 32, accumulating gradients over 8 update steps, using the Adam optimiser (Kingma and Ba, 2015) with learning rate 1e-5. We use early stopping (patience 5) if validation performance does not improve.

5.1 Evaluation Method

To assess the model performance we use automatic metrics commonly used in figurative language generation and text style transfer, which focus on form strength and context preservation.

Form Strength To evaluate the form accuracy of the generated text, we reuse the binary classifiers trained for selecting pre-training data. High confidence for the target figurative form, suggests high accuracy in the generation. The performance of the classifiers on the test set (Table 3), suggests that they are very reliable for Simile, Idiom, and Hyperbole, and slightly less for Metaphor and Sarcasm.

	TGT	BLEU	BERT	BLEURT	COMET	HM	TGT	BLEU	BERT	BLEURT	COMET	HM
	Literal Form→Hyperbole						Literal Form→Idiom					
BART-Single	0.627	0.513	0.693	0.280	0.461	0.564	0.711	**0.791**	**0.855**	**0.595**	**0.808**	0.749
BART-Multi	0.707	0.541	0.698	0.260	0.352	0.613	0.637	0.747	0.829	0.498	0.706	0.688
PT-to-FT	0.833	**0.582**	**0.733**	**0.379**	**0.490**	**0.686**	**0.769**	0.765	0.841	0.536	0.738	**0.767**
mFLAG	**0.844**	0.556	0.726	0.349	0.463	0.670	0.764	0.761	0.839	0.539	0.735	0.762
	Literal Form→Sarcasm						Literal Form→Metaphor					
BART-Single	0.679	**0.491**	**0.611**	**0.052**	**0.188**	0.570	0.720	0.595	0.771	0.364	0.720	0.652
BART-Multi	0.743	0.483	0.598	0.011	0.137	0.585	0.767	0.577	0.780	0.434	0.785	0.659
PT-to-FT	**0.765**	0.485	0.609	0.040	0.162	**0.594**	0.867	**0.643**	**0.812**	**0.493**	0.842	**0.738**
mFLAG	0.762	0.487	0.609	0.043	0.169	**0.594**	**0.880**	0.628	0.809	0.490	**0.844**	0.733
	Literal Form→Simile						Figurative→Literal Form					
BART-Single	0.647	0.724	0.720	**0.017**	**0.321**	0.683	0.733	0.606	0.742	0.284	0.455	0.663
BART-Multi	0.420	0.658	0.681	-0.025	0.178	0.513	0.725	0.622	0.762	0.364	0.522	0.670
PT-to-FT	0.907	0.729	0.722	-0.021	0.219	0.808	**0.801**	0.634	0.766	**0.542**	0.544	**0.708**
mFLAG	**0.953**	**0.745**	**0.727**	-0.021	0.220	**0.836**	0.796	**0.637**	**0.769**	0.375	**0.681**	0.707

Table 4: Results of literal↔figurative form generation. TGT represents the accuracy of output labeled as the target form by the classifier; the results of figurative→literal form generation are averaged across all figures of speech.

Context Preservation To assess this aspect, we adopt BLEU and BERTScore (F1-Score) (Zhang et al., 2020) following previous work (Chakrabarty et al., 2020b; Zhang and Wan, 2022; Zhou et al., 2021; Tian et al., 2021). In addition, we employ BLEURT (Sellam et al., 2020) and COMET (Rei et al., 2020), two learnable metrics that have shown promising results in the evaluation of formality transfer (Lai et al., 2022). For all metrics, we calculate scores between model outputs and references for the literal↔figurative generation, and between outputs and source sentences (and literal sentences) for figurative↔figurative generation as the latter has no parallel data available. [4]

Overall Score We compute the harmonic mean (HM) of figurative accuracy and BLEU score for a direct comparison to baselines.

5.2 Baselines

We compare our systems to two strong baselines.

BART-Single For each figure of speech, we fine-tune BART on the corresponding parallel data. For figurative→figurative generation, we use each figurative-to-literal model to generate the literal text, and then feed it into the model of the target form to generate the output.

BART-Multi We concatenate the five parallel training sets and fine-tune BART for multi-figurative language modelling, thereby enabling the generation between different forms.

5.3 Literal↔Figurative Generation

Table 4 presents the results of literal↔figurative form generation. BART-Multi outperforms BART-Single on most generation directions, except literal-to-idiom and literal-to-simile. This suggests that the model does benefit from multi-figurative language modelling with cross-figurative knowledge transfer. Compared to BART-Single and BART-Multi, both of our proposed models PT-to-FT and mFLAG have consistently stronger results. Specifically, we observe that BART-Single has the best performance only on context preservation for literal-to-idiom and literal-to-sarcasm generation, while our models are better for the rest, especially with a good balance between form strength and context preservation. The results confirm that our pre-training scheme and strategies significantly improve performances for multi-figurative language modelling. When looking at PT-to-FT and mFLAG, we see that these two models' performances are very close on all tasks and do not show a clear and consistent trend. The main reason for this is most likely that the settings of the two models are almost identical except that mFLAG has a figurative injection mechanism, and they are both trained with parallel literal↔figurative sentence pairs.

5.4 Figurative↔Figurative Generation

Table 5 reports results of figurative↔figurative form generation.[5] We see that both BART-Multi and PT-to-FT perform poorly on the form strength

[4] In our evaluation, we take `multi-bleu.perl` to calculate BLEU score, and models `bleurt-large-512` and `wmt-large-da-estimator-1719` for BLEURT and COMET, respectively.

[5] Complete results are in Appendix A.2.

	Form Strength		Source Text					Literal Text				
	SRC	TGT	BLEU	BERT	BLEURT	COMET	HM	BLEU	BERT	BLEURT	COMET	HM
Hyperbole→Others												
BART-Single	0.470	0.425	0.665	0.782	0.459	0.472	0.519	0.488	0.700	0.294	0.248	0.454
BART-Multi	0.328	0.242	0.602	0.761	0.455	0.443	0.345	0.505	0.731	0.427	0.385	0.327
PT-to-FT	0.252	0.258	0.590	0.749	0.437	0.420	0.359	**0.507**	**0.732**	**0.438**	**0.407**	0.342
mFLAG-DR	**0.922**	0.608	**0.815**	**0.893**	**0.753**	**0.836**	**0.696**	0.411	0.633	0.036	-0.105	0.490
mFLAG-BT	0.482	**0.644**	0.539	0.702	0.253	0.246	0.586	0.421	0.662	0.169	0.093	**0.509**
Idiom→Others												
BART-Single	0.290	0.309	0.783	0.864	0.575	0.646	0.443	0.749	0.844	0.578	0.659	0.438
BART-Multi	0.273	0.204	0.785	0.873	0.602	0.674	0.324	0.758	0.859	0.630	0.701	0.408
PT-to-FT	0.204	0.207	0.771	0.867	0.594	0.662	0.326	**0.760**	**0.860**	**0.646**	**0.715**	0.325
mFLAG-DR	**0.910**	0.400	**0.901**	**0.940**	**0.822**	**0.869**	**0.554**	0.694	0.799	0.328	0.375	0.507
mFLAG-BT	0.328	**0.409**	0.724	0.831	0.491	0.566	0.523	0.703	0.816	0.490	0.569	**0.517**
Sarcasm→Others												
BART-Single	0.577	0.370	0.877	0.899	0.650	0.792	0.520	0.454	0.579	-0.088	-0.051	0.408
BART-Multi	0.569	0.247	0.903	0.923	0.701	0.838	0.388	0.471	**0.593**	-0.049	-0.014	0.324
PT-to-FT	0.464	0.252	0.863	0.891	0.613	0.774	0.390	**0.468**	0.592	**-0.031**	**0.000**	0.328
mFLAG-DR	**0.840**	0.438	**0.907**	**0.928**	**0.813**	**0.872**	0.591	0.442	0.563	-0.198	-0.143	0.440
mFLAG-BT	0.583	**0.481**	0.808	0.831	0.460	0.604	**0.605**	0.430	0.554	-0.164	-0.133	0.**454**
Metaphor→Others												
BART-Single	0.163	0.314	0.603	0.776	0.412	0.555	0.413	0.575	0.773	0.381	0.486	0.406
BART-Multi	0.255	0.249	0.647	0.825	0.554	0.723	0.360	0.632	0.820	0.550	0.689	0.357
PT-to-FT	0.147	0.254	0.671	0.832	0.599	**0.763**	0.369	**0.648**	**0.824**	**0.507**	**0.665**	0.365
mFLAG-DR	**0.795**	0.518	**0.697**	**0.846**	**0.614**	0.706	**0.594**	0.516	0.758	0.320	0.410	0.517
mFLAG-BT	0.387	**0.557**	0.502	0.734	0.329	0.434	0.528	0.496	0.743	0.317	0.417	**0.525**
Simile→Others												
BART-Single	0.057	0.607	0.469	0.559	-0.406	-0.429	0.529	0.588	0.667	0.160	-0.102	0.597
BART-Multi	0.007	0.272	0.629	0.686	-0.043	-0.051	0.380	**0.765**	**0.818**	**0.262**	**0.415**	0.401
PT-to-FT	0.000	0.314	0.622	0.671	-0.031	-0.067	0.417	0.754	0.804	0.244	0.394	0.443
mFLAG-DR	**0.440**	0.685	**0.849**	**0.884**	**0.637**	**0.690**	**0.758**	0.589	0.698	-0.016	-0.057	0.633
mFLAG-BT	0.132	**0.687**	0.606	0.670	-0.069	-0.064	0.644	0.672	0.766	0.163	0.250	**0.679**

Table 5: Results of figurative↔figurative form generation. Notes: (i) SRC (TGT) represents the accuracy of output labeled as the source (target) form by the classifier of the source (target) form; (ii) results for each block are averaged for all generations from one figurative language to others.

and the context preservation computed against the source text. The low form strength (SRC and TGT, see table's caption) and high scores of context preservation (using literal text) suggest that these two models transform the source text into the literal form. BART-Single, interestingly, shows a better performance on both form strength and context preservation. For mFLAG-DR and mFLAG-BT, we see that they show the best performance across the board: (i) mFLAG-DR shows a significant improvement in target figurative form (TGT) while maintaining the original form (SRC) very much; it also achieves the best performance on context preservation; (ii) mFLAG-BT achieves the highest form accuracy in the target figure of speech while reducing the original form strength.

It is interesting to note that the direct generation method might allow for the source figure of speech to be retained in the generated sentence, as we do not explicitly remove it by transforming the sentence to its literal form first. For example, with hyperbolic input "*I am not happy that he urged me to finish all the hard task in the world*", one of our sarcastic transformations reads "*Thank you for encouraging me to finish all the hardest tasks in the world*", where the hyperbolic part ("*all the hardest tasks in the world*") is preserved unchanged (see Table 6).

Overall, the results show that mFLAG with the mechanism for injecting the figurative information into the encoder can generate from one figure of speech to another even without task-specific parallel data.

6 Analysis and Discussion

Case Study Table 6 shows a group of example outputs for hyperbole→others generated by various models.[6] From the results of hyperbole→literal generation, we see that mFLAG generates the literal sentence from the hyperbolic one very well,

[6]More example outputs of mFLAG are in Appendix A.3.

Forms	Models	Sentences	Suc.
Hyperbole	Input	I am not happy that he urged me to finish all the hardest tasks in the world.	-
Literal	BART-Single	I am not happy that he urged me to finish all the hard tasks in the world.	✗
	BART-Multi	I am glad that he urged me to finish all the hardest tasks in the world.	✗
	PT-to-FT	I am not happy that he urged me to complete all the difficult tasks in the world.	✗
	mFLAG	I am not happy that he urged me to finish all the difficult tasks.	✓
Idiom	BART-Single	I am not happy that he urged me to finish all the hard tasks in the world.	✗
	BART-Multi	I am so glad that he urged me to finish all the hardest tasks in the world.	✗
	PT-to-FT	I am not happy that he urged me to complete all the difficult tasks.	✗
	mFLAG-DR	I am not happy that he urged me to tick off all the hardest tasks in the world.	✓
	mFLAG-BT	I am not happy that he urged me to get cracking on all the difficult tasks.	✓
Sarcasm	BART-Single	I'm so glad my dad encouraged me to finish all the hard tasks in the world.	✓
	BART-Multi	Thank you for encouraging me to finish all the hardest tasks in the world.	✓
	PT-to-FT	I am happy that he urged me to finish all the hardest tasks in the world.	✓
	mFLAG-DR	Glad he urged me to finish all the hardest tasks in the world.	✓
	mFLAG-BT	Glad he urged me to finish all the difficult tasks.	✓
Metaphor	BART-Single	I am not happy that he urged me to bear all the difficult tasks.	✓
	BART-Multi	I am so glad that he urged me to finish all the hardest tasks in the world.	✗
	PT-to-FT	I am not happy that he urged me to complete all the difficult tasks in the world.	✗
	mFLAG-DR	I am not happy that he urged me to bear all the hardest tasks in the world.	✓
	mFLAG-BT	I am not happy that he pressed me to finish all the difficult tasks.	✗
Simile	BART-Single	I am not happy that he urged me to finish all the difficult tasks.	✗
	BART-Multi	I am so glad that he urged me to finish all the hardest tasks in the world.	✗
	PT-to-FT	I am not happy that he urged me to complete all the difficult tasks in the world.	✗
	mFLAG-DR	I am not happy that he urged me to finish all the like a million things.	✓
	mFLAG-BT	I am not happy that he urged me to finish all the difficult tasks.	✗

Table 6: Examples outputs generated by various models from hyperbolic text, where red denotes appropriate words/phrases for desired forms. Suc.==Successful.

confirming that texts generated by mFLAG-BT tend to contain less the source form by substituting it with the target form. In figurative↔figurative generation, all models nicely generate sarcastic text while all baselines usually fail at generating the other forms. Since the metaphor generation dataset we used focuses on metaphorical verb aspect, we consider the outputs of BART-Single and mFLAG-DR to be successful. Overall, our proposed mFLAG based models perform better on all generation directions.

Probing Figurative Information for Encoder To measure the distribution of source and target sentences encoded by the Encoder with/without the mechanism of injecting figurative information, we apply Principal Component Analysis (PCA) to reduce the dimensionality of the Encoder outputs and visualise relations between tokens in a two-dimensional space. Fig. 2(a) and 2(b) show the results of a source literal text "*He was nervous waiting for the result.*" and a target hyperbolic text "*He was on pins and needles waiting for the result.*". We see the word "He" and 'was" of the two sentences are not in the same cluster in 2(a) while it is interesting to see that all distances between token pairs of 2(b) are closer, especially the phrase "on pins and needles", and "nervous" are almost in the same cluster in 2(b). Fig. 2(c) and 2(d) show the results of a source idiomatic text "*I felt like I had a feather in my cap after I aced that exam.*" and a target hyperbolic text "*I felt like I was a star after I aced that exam.*". We observe that the token pairs like "I", "like" and 'felt" of mFLAG are closer than those of PT-to-FT. It is also interesting to see that the phrase "a feather in my cap" and the token "star" make more of a cluster in 2(d). We believe this benefits the decoder, especially decoding into the target figurative form.

How similar are different forms? To analyze the connection between literal and figurative forms,

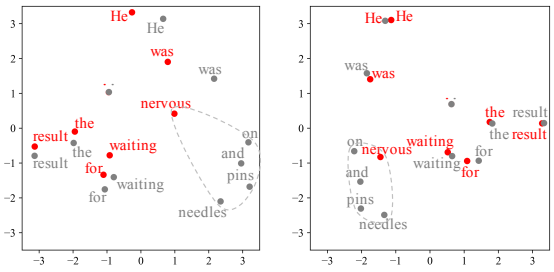

(a) Result for PT-to-FT.

(b) Result for mFLAG.

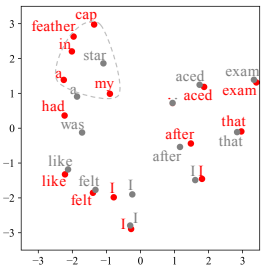

(c) Result for PT-to-FT.

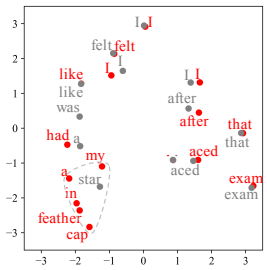

(d) Result for mFLAG.

Figure 2: PCA token representations of encoder outputs for literal→hyperbole (top) and idiom→hyperbole (down).

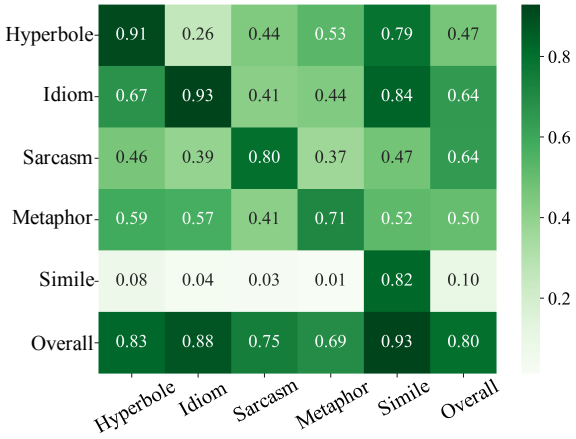

Figure 3: Performances (F1 score) of classifiers on different figurative forms. Each row represents results of a classifier tested on each/all figurative form(s).

and between different figures of speech, we evaluate each figurative classifier on the test sets of the other figurative forms (Figure 3). We first see that the overall model (literal vs figurative) achieves F1 scores of over 0.69 for each figure of speech, confirming the feasibility of multi-figurative modelling. For each figure of speech, we observe: (i) classifiers for hyperbole and idiom have high F1 scores on the test set of simile (0.79 and 0.84), suggesting that sentences with similes may also be hyperbolic or idiomatic; (ii) for sarcasm and metaphor, classifiers have medium scores on other forms; (iii) the classifier of simile achieves F1 scores of less than 0.11 on other figures of speech; this is due to the fact that the simile dataset was created using the format *like a*, which is easy for the model to learn. Different figurative forms are related to each other, confirming that models can benefit from cross-figurative knowledge transfer. Further (computational) analysis of similarities and differences will help to even better leverage such transfer.

7 Conclusion and Outlook

We have proposed a novel task of multi-figurative language generation, and shown that our models do benefit from cross-figurative knowledge transfer. Paraphrasing data can be leveraged in further pretraining to enhance both form strength and context preservation in figurative language generation. We have also proposed a mechanism for injecting the target figurative information into the encoder, so that we can achieve generation between different figures of speech even without parallel figurative-figurative pairs.

While we innovatively explore multi-figurative language generation across literal and five figurative forms, and our model achieves the best performances compared to baselines, there is still substantial room for improvement and further extensions.

The current lack of human references for automating the evaluation of figurative-to-figurative generation is surely a limitation in terms of better understanding of the models' behaviour and potential improvements. More in general, figurative language generation is a relatively new task, which still lacks standardised evaluation methods, both in terms of automatic metrics and human-based evaluation.

Also, we introduce for the first time generation across literal expressions and five figurative forms, but there are many more forms of creative writing that could be modelled. Moreover, we only limited our attention to English, due to data availability, but are convinced that datasets in other languages would greatly benefit research in this area. Indeed, multilingual modelling would make it possible to make connections across different languages, thus shedding more light on cross-lingual regularities in figurative language use, and thus also open up potential avenues to tackle this task better.

Acknowledgments

This work was partly funded by the China Scholarship Council (CSC). The COLING anonymous reviewers provided us with useful comments which contributed to improving this paper and its presentation, so we're grateful to them. We would also like to thank the Center for Information Technology of the University of Groningen for their support and for providing access to the Peregrine high performance computing cluster.

References

Keiga Abe, Kayo Sakamoto, and Masanori Nakagawa. 2006. A computational model of the metaphor generation process. In *Proceedings of the 28th Annual Meeting of the Cognitive Science Society*.

Beata Beigman Klebanov, Chee Wee Leong, E. Dario Gutierrez, Ekaterina Shutova, and Michael Flor. 2016. Semantic classifications for detection of verb metaphors. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 101–106, Berlin, Germany. Association for Computational Linguistics.

Jonathan Berant and Percy Liang. 2014. Semantic parsing via paraphrasing. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1415–1425, Baltimore, Maryland. Association for Computational Linguistics.

Antoine Bosselut, Hannah Rashkin, Maarten Sap, Chaitanya Malaviya, Asli Celikyilmaz, and Yejin Choi. 2019. COMET: Commonsense transformers for automatic knowledge graph construction. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 4762–4779, Florence, Italy. Association for Computational Linguistics.

Chris Callison-Burch, Philipp Koehn, and Miles Osborne. 2006. Improved statistical machine translation using paraphrases. In *Proceedings of the Human Language Technology Conference of the NAACL, Main Conference*, pages 17–24, New York City, USA. Association for Computational Linguistics.

Ziqiang Cao, Chuwei Luo, Wenjie Li, and Sujian Li. 2017. Joint copying and restricted generation for paraphrase. In *Proceedings of the Thirty-First AAAI Conference on Artificial Intelligence*, AAAI'17, page 3152–3158. AAAI Press.

Tuhin Chakrabarty, Debanjan Ghosh, Smaranda Muresan, and Nanyun Peng. 2020a. Rˆ3: Reverse, retrieve, and rank for sarcasm generation with commonsense knowledge. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7976–7986, Online. Association for Computational Linguistics.

Tuhin Chakrabarty, Smaranda Muresan, and Nanyun Peng. 2020b. Generating similes effortlessly like a pro: A style transfer approach for simile generation. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 6455–6469, Online. Association for Computational Linguistics.

Tuhin Chakrabarty, Xurui Zhang, Smaranda Muresan, and Nanyun Peng. 2021. MERMAID: Metaphor generation with symbolism and discriminative decoding. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 4250–4261, Online. Association for Computational Linguistics.

Ronan Collobert and Jason Weston. 2008. A unified architecture for natural language processing: Deep neural networks with multitask learning. In *Proceedings of the 25th international conference on Machine learning*, page 160–167, New York, NY, USA. Association for Computing Machinery.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, Minneapolis, Minnesota. Association for Computational Linguistics.

Li Dong, Jonathan Mallinson, Siva Reddy, and Mirella Lapata. 2017. Learning to paraphrase for question answering. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 875–886, Copenhagen, Denmark. Association for Computational Linguistics.

Debanjan Ghosh, Elena Musi, and Smaranda Muresan. 2020. Interpreting verbal irony: Linguistic strategies and the connection to theType of semantic incongruity. In *Proceedings of the Society for Computation in Linguistics 2020*, pages 82–93, New York, New York. Association for Computational Linguistics.

Ana Guerberof-Arenas and Antonio Toral. 2022. Creativity in translation: Machine translation as a constraint for literary texts. *Translation Spaces*.

Hessel Haagsma, Johan Bos, and Malvina Nissim. 2020. MAGPIE: A large corpus of potentially idiomatic expressions. In *Proceedings of the 12th Language Resources and Evaluation Conference*, pages 279–287, Marseille, France. European Language Resources Association.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. 2016. Deep residual learning for image recognition. In *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 770–778.

J. Edward Hu, Abhinav Singh, Nils Holzenberger, Matt Post, and Benjamin Van Durme. 2019. Large-scale, diverse, paraphrastic bitexts via sampling and clustering. In *Proceedings of the 23rd Conference on Computational Natural Language Learning (CoNLL)*, pages 44–54, Hong Kong, China. Association for Computational Linguistics.

Arthur M. Jacobs. 2018. The gutenberg english poetry corpus: Exemplary quantitative narrative analyses. *Frontiers Digit. Humanit.*, 5:5.

Diederik P. Kingma and Jimmy Ba. 2015. Adam: A method for stochastic optimization. In *International Conference on Learning Representations*.

Catherine Kobus, Josep Crego, and Jean Senellart. 2017. Domain control for neural machine translation. In *Proceedings of the International Conference Recent Advances in Natural Language Processing, RANLP 2017*, pages 372–378, Varna, Bulgaria. INCOMA Ltd.

Huiyuan Lai, Jiali Mao, Antonio Toral, and Malvina Nissim. 2022. Human judgement as a compass to navigate automatic metrics for formality transfer. In *Proceedings of the 2nd Workshop on Human Evaluation of NLP Systems (HumEval)*, pages 102–115, Dublin, Ireland. Association for Computational Linguistics.

Huiyuan Lai, Antonio Toral, and Malvina Nissim. 2021. Generic resources are what you need: Style transfer tasks without task-specific parallel training data. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 4241–4254, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Veselin Stoyanov, and Luke Zettlemoyer. 2020. BART: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7871–7880, Online. Association for Computational Linguistics.

Yinhan Liu, Jiatao Gu, Naman Goyal, Xian Li, Sergey Edunov, Marjan Ghazvininejad, Mike Lewis, and Luke Zettlemoyer. 2020. Multilingual denoising pre-training for neural machine translation. *Transactions of the Association for Computational Linguistics*, 8:726–742.

Rui Mao, Chenghua Lin, and Frank Guerin. 2018. Word embedding and WordNet based metaphor identification and interpretation. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1222–1231, Melbourne, Australia. Association for Computational Linguistics.

Lotem Peled and Roi Reichart. 2017. Sarcasm SIGN: Interpreting sarcasm with sentiment based monolingual machine translation. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1690–1700, Vancouver, Canada. Association for Computational Linguistics.

Aaditya Prakash, Sadid A. Hasan, Kathy Lee, Vivek Datla, Ashequl Qadir, Joey Liu, and Oladimeji Farri. 2016. Neural paraphrase generation with stacked residual LSTM networks. In *Proceedings of COLING 2016, the 26th International Conference on Computational Linguistics: Technical Papers*, pages 2923–2934, Osaka, Japan. The COLING 2016 Organizing Committee.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J. Liu. 2020. Exploring the limits of transfer learning with a unified text-to-text transformer. *Journal of Machine Learning Research*, 21(140):1–67.

Sudha Rao and Joel Tetreault. 2018. Dear sir or madam, may I introduce the GYAFC dataset: Corpus, benchmarks and metrics for formality style transfer. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 129–140, New Orleans, Louisiana. Association for Computational Linguistics.

Ricardo Rei, Craig Stewart, Ana C Farinha, and Alon Lavie. 2020. COMET: A neural framework for MT evaluation. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 2685–2702, Online. Association for Computational Linguistics.

Richard M. Roberts and Roger J. Kreuz. 1994. Why do people use figurative language? *Psychological Science*, 5(3):159–163.

Thibault Sellam, Dipanjan Das, and Ankur Parikh. 2020. BLEURT: Learning robust metrics for text generation. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7881–7892, Online. Association for Computational Linguistics.

Kevin Stowe, Nils Beck, and Iryna Gurevych. 2021a. Exploring metaphoric paraphrase generation. In *Proceedings of the 25th Conference on Computational Natural Language Learning*, pages 323–336, Online. Association for Computational Linguistics.

Kevin Stowe, Tuhin Chakrabarty, Nanyun Peng, Smaranda Muresan, and Iryna Gurevych. 2021b. Metaphor generation with conceptual mappings. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language*

Processing (Volume 1: Long Papers), pages 6724–6736, Online. Association for Computational Linguistics.

Yuqing Tang, Chau Tran, Xian Li, Peng-Jen Chen, Naman Goyal, Vishrav Chaudhary, Jiatao Gu, and Angela Fan. 2020. Multilingual translation with extensible multilingual pretraining and finetuning. *arXiv preprint, arXiv: 2008.00401*.

Yufei Tian, Arvind krishna Sridhar, and Nanyun Peng. 2021. HypoGen: Hyperbole generation with commonsense and counterfactual knowledge. In *Findings of the Association for Computational Linguistics: EMNLP 2021*, pages 1583–1593, Punta Cana, Dominican Republic. Association for Computational Linguistics.

Enrica Troiano, Carlo Strapparava, Gözde Özbal, and Serra Sinem Tekiroğlu. 2018. A computational exploration of exaggeration. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 3296–3304, Brussels, Belgium. Association for Computational Linguistics.

Tony Veale. 2016. Round up the usual suspects: Knowledge-based metaphor generation. In *Proceedings of the Fourth Workshop on Metaphor in NLP*, pages 34–41, San Diego, California. Association for Computational Linguistics.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Remi Louf, Morgan Funtowicz, Joe Davison, Sam Shleifer, Patrick von Platen, Clara Ma, Yacine Jernite, Julien Plu, Canwen Xu, Teven Le Scao, Sylvain Gugger, Mariama Drame, Quentin Lhoest, and Alexander Rush. 2020. Transformers: State-of-the-art natural language processing. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 38–45, Online. Association for Computational Linguistics.

Zhiwei Yu, Jiwei Tan, and Xiaojun Wan. 2018. A neural approach to pun generation. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1650–1660, Melbourne, Australia. Association for Computational Linguistics.

Zhiwei Yu and Xiaojun Wan. 2019. How to avoid sentences spelling boring? towards a neural approach to unsupervised metaphor generation. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 861–871, Minneapolis, Minnesota. Association for Computational Linguistics.

Jiayi Zhang, Zhi Cui, Xiaoqiang Xia, Yalong Guo, Yanran Li, Chen Wei, and Jianwei Cui. 2021. Writing polishment with simile: Task, dataset and a neural approach. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 35, pages 14383–14392.

Tianyi Zhang, Varsha Kishore, Felix Wu, Kilian Q. Weinberger, and Yoav Artzi. 2020. Bertscore: Evaluating text generation with bert. In *International Conference on Learning Representations*.

Yunxiang Zhang and Xiaojun Wan. 2022. MOVER: Mask, over-generate and rank for hyperbole generation. In *Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 6018–6030, Seattle, United States. Association for Computational Linguistics.

Jianing Zhou, Hongyu Gong, Srihari Nanniyur, and Suma Bhat. 2021. From solving a problem boldly to cutting the gordian knot: Idiomatic text generation. *arXiv preprint, arXiv: 2104.06541*.

Mengdi Zhu, Zhiwei Yu, and Xiaojun Wan. 2019. A neural approach to irony generation. *arXiv preprint, arXiv: 1909.06200*.

A Appendices:

This appendices include: (i) Dataset statistics of pre-training data (A.1); (ii) Detailed results for figurative↔figurative generation (A.2); (iii) Example outputs of mFLAG (A.3) .

A.1 Pre-Training Data

Forms	Task	σ	Train	Valid
Hyperbole	Literal text↔Hyperbole	0.94	102,887	5,000
Idiom	Literal text↔idiom	0.95	133,285	5,000
Sarcasm	Literal text↔Sarcasm	0.70	22,550	5,000
Metaphor	Literal text↔Metaphor	0.95	206,554	5,000
Simile	Literal text↔Simile	0.76	57,566	5,000

Table A.1: Dataset statistics for generic pre-training data. Note that σ is the threshold used to select sentence pairs.

A.2 Detailed Results for Figurative↔Figurative Generation

	Form Strength		Source Text					Literal Text				
	SRC	TGT	BLEU	BERT	BLEURT	COMET	HM	BLEU	BERT	BLEURT	COMET	HM
Hyperbole→Idiom												
BART-Single	0.513	0.513	0.653	0.781	0.469	0.466	0.575	0.471	0.692	0.294	0.240	0.491
BART-Multi	0.313	0.233	0.595	0.755	0.439	0.425	0.335	0.505	0.730	0.429	0.385	0.386
PT-to-FT	0.240	0.200	0.587	0.747	0.445	0.422	0.298	0.506	0.729	0.442	0.402	0.287
mFLAG-DR	0.900	0.733	0.766	0.876	0.729	0.758	0.749	0.401	0.637	0.063	-0.089	0.518
mFLAG-BT	0.653	0.707	0.599	0.743	0.368	0.380	0.649	0.409	0.650	0.136	-0.011	0.518
Hyperbole→Sarcasm												
BART-Single	0.407	0.387	0.673	0.785	0.464	0.595	0.491	0.499	0.710	0.300	0.298	0.436
BART-Multi	0.333	0.313	0.601	0.760	0.464	0.447	0.412	0.500	0.730	0.427	0.386	0.385
PT-to-FT	0.267	0.373	0.587	0.744	0.400	0.399	0.456	0.505	0.728	0.392	0.385	0.429
mFLAG-DR	0.900	0.447	0.873	0.922	0.883	0.947	0.591	0.431	0.645	0.073	-0.006	0.439
mFLAG-BT	0.373	0.507	0.545	0.699	0.283	0.265	0.525	0.442	0.678	0.233	0.233	0.472
Hyperbole→Metaphor												
BART-Single	0.407	0.533	0.653	0.784	0.501	0.509	0.587	0.499	0.712	0.369	0.331	0.515
BART-Multi	0.320	0.407	0.597	0.758	0.439	0.432	0.484	0.505	0.730	0.422	0.383	0.451
PT-to-FT	0.253	0.447	0.592	0.756	0.450	0.432	0.509	0.513	0.736	0.451	0.423	0.478
mFLAG-DR	0.927	0.773	0.823	0.902	0.762	0.870	0.797	0.412	0.634	0.033	-0.081	0.538
mFLAG-BT	0.300	0.753	0.495	0.692	0.227	0.235	0.597	0.433	0.686	0.252	0.226	0.550
Hyperbole→Simile												
BART-Single	0.553	0.267	0.680	0.779	0.402	0.416	0.383	0.481	0.687	0.214	0.123	0.342
BART-Multi	0.347	0.013	0.616	0.771	0.476	0.467	0.025	0.511	0.733	0.431	0.387	0.025
PT-to-FT	0.247	0.013	0.595	0.747	0.451	0.424	0.025	0.505	0.732	0.465	0.418	0.332
mFLAG-DR	0.960	0.480	0.798	0.873	0.639	0.709	0.599	0.400	0.616	-0.026	-0.242	0.436
mFLAG-BT	0.600	0.607	0.525	0.674	0.135	0.105	0.551	0.401	0.634	0.055	-0.077	0.563

Table A.2: Results of hyperbole→others generation.

	Form Strength		Source Text					Literal Text				
	SRC	**TGT**	**BLEU**	**BERT**	**BLEURT**	**COMET**	**HM**	**BLEU**	**BERT**	**BLEURT**	**COMET**	**HM**
Idiom→Hyperbole												
BART-Single	0.311	0.103	0.788	0.867	0.585	0.653	0.182	0.751	0.844	0.575	0.651	0.181
BART-Multi	0.269	0.031	0.784	0.872	0.600	0.671	0.059	0.758	0.859	0.632	0.702	0.059
PT-to-FT	0.232	0.041	0.782	0.874	0.614	0.681	0.078	0.763	0.862	0.647	0.717	0.078
mFLAG-DR	0.929	0.232	0.847	0.908	0.716	0.769	0.364	0.667	0.783	0.286	0.317	0.344
mFLAG-BT	0.564	0.172	0.728	0.836	0.523	0.574	0.278	0.679	0.799	0.415	0.477	0.274
Idiom→Sarcasm												
BART-Single	0.277	0.335	0.795	0.872	0.602	0.671	0.471	0.761	0.853	0.609	0.692	0.465
BART-Multi	0.281	0.292	0.785	0.875	0.608	0.679	0.426	0.756	0.857	0.623	0.693	0.421
PT-to-FT	0.230	0.319	0.773	0.866	0.587	0.657	0.452	0.755	0.854	0.620	0.690	0.449
mFLAG-DR	0.924	0.376	0.927	0.955	0.871	0.919	0.535	0.711	0.804	0.345	0.395	0.492
mFLAG-BT	0.233	0.405	0.721	0.828	0.485	0.570	0.519	0.710	0.821	0.515	0.613	0.516
Idiom→Metaphor												
BART-Single	0.280	0.692	0.768	0.858	0.561	0.643	0.728	0.734	0.840	0.571	0.667	0.728
BART-Multi	0.268	0.485	0.784	0.872	0.600	0.671	0.599	0.759	0.859	0.633	0.703	0.592
PT-to-FT	0.170	0.467	0.762	0.862	0.581	0.656	0.579	0.760	0.862	0.656	0.728	0.579
mFLAG-DR	0.866	0.798	0.879	0.938	0.821	0.876	0.837	0.687	0.803	0.359	0.420	0.739
mFLAG-BT	0.247	0.798	0.703	0.828	0.482	0.580	0.747	0.688	0.820	0.515	0.620	0.739
Idiom→Simile												
BART-Single	0.293	0.106	0.782	0.859	0.550	0.616	0.187	0.748	0.839	0.557	0.627	0.186
BART-Multi	0.274	0.007	0.786	0.874	0.601	0.673	0.014	0.759	0.860	0.633	0.704	0.014
PT-to-FT	0.184	0.000	0.766	0.864	0.592	0.655	0.000	0.762	0.862	0.662	0.726	0.000
mFLAG-DR	0.920	0.193	0.949	0.959	0.878	0.909	0.321	0.712	0.805	0.322	0.368	0.304
mFLAG-BT	0.266	0.259	0.744	0.832	0.475	0.539	0.384	0.736	0.825	0.514	0.566	0.383

Table A.3: Results of idiom→others generation.

	Form Strength		Source Text					Literal Text				
	SRC	**TGT**	**BLEU**	**BERT**	**BLEURT**	**COMET**	**HM**	**BLEU**	**BERT**	**BLEURT**	**COMET**	**HM**
Sarcasm→Hyperbole												
BART-Single	0.568	0.405	0.907	0.921	0.727	0.855	0.560	0.470	0.590	-0.050	-0.010	0.435
BART-Multi	0.558	0.347	0.898	0.918	0.690	0.828	0.501	0.471	0.592	-0.048	-0.013	0.400
PT-to-FT	0.459	0.384	0.878	0.901	0.635	0.799	0.534	0.473	0.595	-0.022	0.010	0.349
mFLAG-DR	0.823	0.466	0.914	0.936	0.862	0.904	0.617	0.449	0.569	-0.169	-0.114	0.457
mFLAG-BT	0.612	0.473	0.821	0.849	0.548	0.675	0.595	0.438	0.562	-0.123	-0.095	0.455
Sarcasm→Idiom												
BART-Single	0.582	0.429	0.853	0.889	0.615	0.730	0.571	0.441	0.575	-0.098	-0.090	0.435
BART-Multi	0.568	0.299	0.901	0.921	0.697	0.836	0.449	0.472	0.593	-0.051	-0.017	0.366
PT-to-FT	0.422	0.276	0.862	0.886	0.599	0.700	0.418	0.462	0.594	-0.024	0.006	0.394
mFLAG-DR	0.847	0.517	0.875	0.911	0.749	0.808	0.650	0.426	0.554	-0.229	-0.193	0.467
mFLAG-BT	0.599	0.527	0.791	0.825	0.442	0.570	0.633	0.417	0.550	-0.176	-0.166	0.466
Sarcasm→Metaphor												
BART-Single	0.571	0.483	0.851	0.881	0.591	0.788	0.616	0.445	0.571	-0.112	-0.049	0.463
BART-Multi	0.561	0.337	0.900	0.919	0.693	0.830	0.490	0.471	0.592	-0.046	-0.014	0.393
PT-to-FT	0.514	0.344	0.870	0.901	0.654	0.796	0.493	0.472	0.592	-0.037	-0.007	0.398
mFLAG-DR	0.833	0.534	0.907	0.928	0.805	0.906	0.672	0.439	0.563	-0.203	-0.119	0.482
mFLAG-BT	0.520	0.578	0.790	0.827	0.424	0.627	0.668	0.431	0.556	-0.166	-0.100	0.494
Sarcasm→Simile												
BART-Single	0.585	0.163	0.897	0.906	0.666	0.793	0.276	0.460	0.581	-0.091	-0.056	0.241
BART-Multi	0.588	0.003	0.911	0.932	0.725	0.857	0.006	0.471	0.594	-0.050	-0.013	0.005
PT-to-FT	0.459	0.003	0.842	0.874	0.565	0.730	0.006	0.465	0.587	-0.042	-0.008	0.006
mFLAG-DR	0.857	0.235	0.932	0.937	0.835	0.870	0.375	0.452	0.566	-0.191	-0.144	0.309
mFLAG-BT	0.599	0.344	0.821	0.822	0.424	0.544	0.485	0.433	0.547	-0.189	-0.171	0.383

Table A.4: Results of sarcasm→others generation.

	Form Strength		Source Text					Literal Text				
	SRC	TGT	BLEU	BERT	BLEURT	COMET	HM	BLEU	BERT	BLEURT	COMET	HM
Metaphor→Hyperbole												
BART-Single	0.173	0.480	0.617	0.786	0.446	0.582	0.540	0.588	0.779	0.399	0.511	0.529
BART-Multi	0.260	0.427	0.643	0.826	0.562	0.722	0.513	0.635	0.825	0.561	0.700	0.511
PT-to-FT	0.233	0.480	0.711	0.870	0.709	0.832	0.573	0.639	0.827	0.508	0.667	0.548
mFLAG-DR	0.827	0.653	0.662	0.846	0.634	0.717	0.657	0.516	0.769	0.359	0.450	0.576
mFLAG-BT	0.453	0.620	0.511	0.755	0.438	0.511	0.560	0.496	0.762	0.404	0.492	0.551
Metaphor→Idiom												
BART-Single	0.240	0.447	0.542	0.744	0.361	0.459	0.490	0.518	0.748	0.350	0.411	0.480
BART-Multi	0.253	0.280	0.643	0.825	0.559	0.724	0.390	0.633	0.822	0.550	0.694	0.388
PT-to-FT	0.113	0.260	0.646	0.819	0.573	0.748	0.371	0.657	0.834	0.554	0.683	0.373
mFLAG-DR	0.887	0.547	0.640	0.829	0.582	0.708	0.590	0.542	0.787	0.444	0.561	0.544
mFLAG-BT	0.653	0.547	0.557	0.771	0.453	0.586	0.552	0.524	0.774	0.416	0.541	0.536
Metaphor→Sarcasm												
BART-Single	0.133	0.240	0.623	0.788	0.424	0.604	0.347	0.597	0.782	0.391	0.532	0.347
BART-Multi	0.233	0.280	0.654	0.820	0.527	0.712	0.392	0.621	0.807	0.510	0.652	0.386
PT-to-FT	0.153	0.267	0.683	0.832	0.574	0.761	0.384	0.645	0.812	0.462	0.650	0.378
mFLAG-DR	0.720	0.347	0.788	0.883	0.760	0.843	0.482	0.557	0.767	0.377	0.486	0.428
mFLAG-BT	0.273	0.427	0.511	0.732	0.322	0.496	0.465	0.516	0.742	0.334	0.500	0.467
Metaphor→Simile												
BART-Single	0.107	0.087	0.631	0.785	0.418	0.574	0.153	0.598	0.775	0.384	0.489	0.152
BART-Multi	0.273	0.007	0.647	0.828	0.569	0.733	0.014	0.637	0.826	0.579	0.710	0.014
PT-to-FT	0.087	0.007	0.643	0.808	0.540	0.711	0.014	0.650	0.822	0.503	0.661	0.014
mFLAG-DR	0.747	0.500	0.696	0.827	0.479	0.554	0.581	0.450	0.710	0.099	0.142	0.474
mFLAG-BT	0.167	0.633	0.428	0.679	0.102	0.142	0.511	0.447	0.695	0.115	0.135	0.524

Table A.5: Results of metaphor→others generation.

	Form Strength		Source Text					Literal Text				
	SRC	TGT	BLEU	BERT	BLEURT	COMET	HM	BLEU	BERT	BLEURT	COMET	HM
Simile→Hyperbole												
BART-Single	0.093	0.713	0.492	0.575	-0.358	-0.358	0.582	0.603	0.656	-0.135	-0.127	0.653
BART-Multi	0.007	0.293	0.634	0.689	-0.040	-0.045	0.401	0.770	0.821	0.261	0.418	0.424
PT-to-FT	0.000	0.327	0.649	0.692	0.003	-0.012	0.435	0.777	0.818	0.261	0.417	0.460
mFLAG-DR	0.527	0.893	0.895	0.918	0.772	0.811	0.894	0.583	0.685	-0.041	-0.090	0.705
mFLAG-BT	0.240	0.820	0.640	0.687	-0.035	-0.022	0.719	0.657	0.756	0.162	0.171	0.730
Simile→Idiom												
BART-Single	0.127	0.627	0.488	0.554	-0.367	-0.440	0.549	0.589	0.646	-0.169	-0.204	0.607
BART-Multi	0.007	0.207	0.634	0.689	-0.040	-0.045	0.273	0.770	0.821	0.261	0.418	0.326
PT-to-FT	0.000	0.173	0.644	0.684	0.007	-0.038	0.273	0.781	0.830	0.307	0.470	0.283
mFLAG-DR	0.420	0.800	0.810	0.848	0.508	0.554	0.805	0.600	0.710	0.013	-0.025	0.686
mFLAG-BT	0.200	0.773	0.617	0.683	-0.018	-0.009	0.686	0.636	0.761	0.170	0.212	0.698
Simile→Sarcasm												
BART-Single	0.007	0.440	0.479	0.572	-0.402	-0.420	0.459	0.618	0.704	-0.113	0.001	0.514
BART-Multi	0.007	0.233	0.611	0.671	-0.070	-0.086	0.337	0.748	0.806	0.252	0.396	0.355
PT-to-FT	0.000	0.387	0.551	0.623	-0.128	-0.178	0.455	0.677	0.743	0.117	0.242	0.492
mFLAG-DR	0.373	0.367	0.877	0.892	0.671	0.692	0.517	0.619	0.714	0.001	-0.014	0.598
mFLAG-BT	0.073	0.380	0.618	0.672	-0.057	-0.057	0.471	0.726	0.792	0.241	0.362	0.499
Simile→Metaphor												
BART-Single	0.000	0.647	0.418	0.536	-0.497	-0.499	0.508	0.541	0.660	-0.222	-0.083	0.589
BART-Multi	0.007	0.353	0.638	0.694	-0.022	-0.026	0.455	0.772	0.824	0.273	0.429	0.484
PT-to-FT	0.000	0.367	0.643	0.685	-0.007	-0.041	0.467	0.782	0.825	0.289	0.445	0.500
mFLAG-DR	0.440	0.680	0.815	0.878	0.595	0.702	0.741	0.552	0.681	-0.036	-0.099	0.609
mFLAG-BT	0.013	0.773	0.550	0.638	-0.167	-0.166	0.643	0.668	0.757	0.079	0.256	0.717

Table A.6: Results of simile→others generation.

A.3 Example Outputs of mFLAG

Forms	Sentences	Suc.
Literal [Input]	Old mr. smith has been teaching here for a very long time.	-
Hyperbole	Old mr. smith has been teaching here since the stone age.	✔
Literal [Input]	I like your new haircut!	-
Hyperbole	I love your new haircut!	✗
Hyperbole [Input]	I am not happy that he urged me to finish all the hardest tasks in the world.	-
Idiom	I am not happy that he urged me to tick off all the hardest tasks in the world.	✔
Sarcasm	Glad he urged me to finish all the hardest tasks in the world.	✔
Metaphor	I am not happy that he urged me to bear all the hardest tasks in the world.	✔
Simile	I am not happy that he urged me to finish all the like a million things.	✔
Idiom [Input]	Your exam results this term were not up to the mark.	
Hyperbole	Your exam results this term were way over the top.	✔
Sarcasm	Your exam results this term were up to the mark.	✔
Metaphor	Your exam results this term were not up to the mark.	✗
Simile	Your exam results this term were not up to the mark.	✗
Sarcasm [Input]	Being stuck in an airport is fun.	-
Hyperbole	Being stuck in an airport is the best.	✔
Idiom	Being stuck in an airport is always fun.	✗
Metaphor	Being stuck in an airport is no fun.	✗
Simile	Being stuck in an airport is like a rollercoaster.	✔
Metaphor	The bad news rocked her hopes.	-
Hyperbole	The bad news crushed her hopes.	✔
Idiom	The bad news crushed her hopes.	✗
Sarcasm	The good news just keeps on getting better.	✔
Simile	The bad news rocked her like a rock.	✔
Simile [Input]	One day you decide to test it out and what you experience is like a magic trick.	-
Hyperbole	One day you decide to test it out and what you experience is magic.	✔
Sarcasm	One day you decide to test it out and what you experience is awesome.	✔
Idiom	One day you decide to test it out and what you experience is dangerous.	✗
Metaphor	One day you decide to test it out and what you experience is dangerous.	✗

Table A.7: Example outputs generated by mFLAG-DR, where red denotes appropriate words for desired forms. Suc.==Successful.

Enhancing Task-Specific Distillation in Small Data Regimes through Language Generation

Husam Quteineh[2,1,*], **Spyridon Samothrakis**[3,1], **and Richard Sutcliffe**[4,1,*]

[1]School of CSEE, University of Essex, UK
[2]Business and Local Government Data Research Centre, University of Essex, UK
[3]Institute for Analytics and Data Science (IADS), University of Essex, UK
[4]School of Information Science and Technology, Northwest University, China
husam.quteineh@essex.ac.uk, ssamot@essex.ac.uk, rsutcl@essex.ac.uk, rsutcl@nwu.edu.cn

Abstract

Large-scale pretrained language models have led to significant improvements in Natural Language Processing. Unfortunately, they come at the cost of high computational and storage requirements that complicate their deployment on low-resource devices. This issue can be addressed by distilling knowledge from larger models to smaller ones through pseudo-labels on task-specific datasets. However, this can be difficult for tasks with very limited data. To overcome this challenge, we present a novel approach where knowledge can be distilled from a teacher model to a student model through the generation of synthetic data. For this to be done, we first fine-tune the teacher and student models, as well as a Natural Language Generation (NLG) model, on the target task dataset. We then let both student and teacher work together to condition the NLG model to generate examples that can enhance the performance of the student. We tested our approach on two data generation methods: a) Targeted generation using the Monte Carlo Tree Search (MCTS) algorithm, and b) A Non-Targeted Text Generation (NTTG) method. We evaluate the effectiveness of our approaches against a baseline that uses the BERT model for data augmentation through random word replacement. By testing this approach on the SST-2, MRPC, YELP-2, DBpedia, and TREC-6 datasets, we consistently witnessed considerable improvements over the word-replacement baseline.

1 Introduction

Transformer-based models have shown wide success over a variety of Natural Language Processing (NLP) tasks. Their ability to scale up to trillions of parameters made it possible to acquire and transfer generalized knowledge from large collections of data to downstream tasks. While these models can lead to significant improvements in performance, the increasing size of their learning parameters results in greater complexity and storage requirements. This can be challenging in real-time applications, especially on devices with limited computational resources (Gou et al., 2021). Hence, reducing the size of these language models without sacrificing too much of their performance has become the focus of many works in Knowledge Distillation (KD). Instead of optimizing compressed models for hard-labeled data, Hinton et al. (2015) proposed to train a smaller model (the student) with the task of predicting the probability distribution outputs (soft labels) from a larger model (the teacher). This approach has been shown to produce comparable results between the student and teacher models, but usually relies on a large enough dataset through which knowledge can be transferred. To help improve the student's learning in KD, large unlabeled datasets are required (Tang et al., 2019). Although unlabeled data is cheaper to obtain and is widespread when compared to labeled data, it may not be available for every task and application. We therefore propose to generate synthetic examples that can be used to transfer knowledge to the student in downstream tasks.

To overcome the challenges that come with unavailability of large unlabeled datasets required for the distillation process, we build a data generation framework where the Monte Carlo Tree Search (MCTS) algorithm is applied to help generate examples that, if added to the student's training data, will increase its performance. By taking the difference between the student's and the teacher's

* Corresponding author

Proceedings of the 29th International Conference on Computational Linguistics, pages 5955–5965
October 12–17, 2022.

uncertainty for a generated example, we are able to force the language generation model to create examples that can be pseudo-labeled by the teacher with higher confidence than its student. We make the assumption that the wider the gap between the student's and teacher's uncertainty for a particular example, the more likely it is that this example is correctly pseudo-labeled by the teacher and the less likely that it is known to the student model. By training the student on the generated data with the teacher's pseudo labels, it is able to improve its performance by mimicking its teacher's behavior. To generate examples that meet this condition, we take advantage of MCTS's tendency to search for paths that maximize the reward value, hence, the uncertainty gap. The intuition here is that the larger the positive difference in the uncertainty, the greater the contradiction is between the student and its teacher, and the more likely that the generated example is important for the student's learning stage. We also show that strong results can still be obtained with a random generation approach that does not optimize for a reward function during the generation process. Instead, it first generates data, then selects samples that meet the conditions that are set in the reward function. The contributions of the paper are:

- We propose Monte Carlo Text Generation (MCTG), a novel method in KD which uses MCTS to generate synthetic examples.
- We present Non-Targeted Text Generation (NTTG), in which data is first generated with top-k sampling, then filtered on the conditions of the reward function.
- We show that even when starting with a few examples per label, we can massively improve the student's performance.

The remainder of the paper is structured as follows: Section 2 provides a background and an overview of related literature. Section 3 describes the proposed approach. Section 4 presents the experiments which were carried out. Section 5 gives conclusions and plans for future work.

2 Background

In the pursuit of improving performance for natural language processing, pretraining large-scale models on increasing amounts of unlabeled data has become a common approach (Devlin et al., 2018; Peters et al., 2018; Yang et al., 2019). By leveraging the knowledge gained from the pretraining step, these models have shown impressive performance on many downstream text tasks, e.g. GLUE and SuperGLUE benchmarks (Wang et al., 2018, 2019). However, such improvements are accompanied by an increasing number of learning parameters, creating a need for more computational and storage requirements. To alleviate the aforementioned complexity issues, many have suggested approaches for efficient training through model optimization e.g. removal of inefficient or redundant parameters (Lan et al., 2020; Sajjad et al., 2020), and knowledge distillation (Gou et al., 2021; Sun et al., 2019, 2020; Sanh et al., 2019). In knowledge distillation, the unlabeled data plays an intermediary role, which allows the teacher to transfer its knowledge through its predictions. When this data lacks the components for a meaningful knowledge transfer, e.g. limitations in size or textual variations, augmentation techniques can be applied to create additional examples. For instance, Tang et al. (2019); Jiao et al. (2019) apply task-agnostic heuristics like word replacements, to improve distillation on downstream tasks. The concept of augmenting training examples has been successfully shown to improve training in computer vision (Shorten and Khoshgoftaar, 2019), and has been gaining traction in the NLP domain as well. This includes word manipulations such as the deletion, insertion, or addition of random words in text (Wei and Zou, 2019), paraphrasing or back-translation (Sennrich et al., 2015), and most recently the application of language models to predict alternative words (Kobayashi, 2018).

In this work, instead of relying on the above augmentation techniques, we propose to improve knowledge transfer by involving the student and the teacher in the creation of useful examples. We achieve this through a framework that uses an NLG model to create examples that are deemed useful for knowledge distillation. The steps taken to determine the usefulness of an example are summarized in Figure 1. In MCTG, we achieve this by exploring MCTS's ability for finding optimal solutions which are rewarded by the usefulness of the examples they represent, as explained in section 3.2.

2.1 Knowledge Distillation (KD)

Knowledge distillation is typically aimed at training a student model to mimic the behavior of a larger teacher model. The student can either have

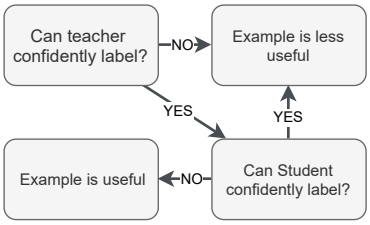

Figure 1: An example is deemed more useful if the teacher can confidently label it, but not the student.

the same architecture as its teacher or be completely different from it, but in either case, it usually has fewer learning parameters. Hinton et al. (2015) achieve knowledge distillation by training the student model on the softmax probabilities of the teacher model. Other KD approaches have also been proposed, which include the distillation of the activations or weights of the intermediate layers (Romero et al., 2014; Tarvainen and Valpola, 2017; Yim et al., 2017; Heo et al., 2019; Cho and Hariharan, 2019). In contrast to much work in KD, we deny both the teacher and its student access to the full training datasets and only train them on a small sample of seed data. Hence, our approach does not depend on pre-existing large datasets for distilling knowledge. Instead, our work focuses on very small data settings. Hence, we make the proposition that fine-tuning compact models on small datasets can be aided by the participation of larger models in a) generating additional training examples, and b) finding informative examples while providing pseudo labels.

2.2 Language Models

Traditional context-independent word vectors like GLOVE (Pennington et al., 2014) and Word2Vec (Mikolov et al., 2013) were popular choices for initializing embedding layers for task-specific networks. Later works focused on contextualizing representations by leveraging recurrent neural networks (Peters et al., 2018; Howard and Ruder, 2018); most recently, the fine-tuning of pretrained transformer-based models (Vaswani et al., 2017) like BERT (Devlin et al., 2018), RoBERTa (Liu et al., 2019) and GPT-2 (Radford et al., 2019), has become a common approach for domain-specific tasks. In our experiments, we generate language with GPT-2, a unidirectional language model, pretrained on large textual datasets with a probabilistic function to estimate the probability distribution over the vocabulary for a given context. For a sequence of tokens $t_1, t_2, t_3, ..., t_n$, the joint probability can be modeled as:

$$p(t) = \prod_{i=1}^{i=n} p(t_{(i)}|t_{(1)}, \ldots, t_{(i-1)}) \quad (1)$$

The conditional probability of a token $p(t_i|t_{1:i\_1})$ can be estimated by the probability distribution over the model's vocabulary given a context $t_{1:i\_1}$. Thus, we can generate candidates for every next token with top-k sampling (Fan et al., 2018). When a token is selected, and the process is repeated enough times, a properly trained model can generate a meaningful sequence of text. Even though we restrict our approach to small datasets, our experiments show that GPT-2 is still able to generate relevant examples.

2.3 Monte Carlo Tree Search (MCTS)

MCTS has been widely applied to gaming theory (Kocsis and Szepesvári, 2006; Browne et al., 2012). Its ability to solve decision-making problems in games with large combinatorial search spaces (Silver et al., 2016; Arneson et al., 2010; Chung et al., 2005), has made its application appealing even for non-gaming problems as well (Nguyen et al., 2016; Edelkamp et al., 2016). In our previous work, we showed that MCTS can also be applied to create synthetic data for text classification tasks (Quteineh et al., 2020).

MCTS incrementally constructs a tree as it searches for possible solutions that satisfy the conditions set by its reward function. In computer games, paths that lead to winning states are more likely to have higher reward values than paths that lead to losing states. While any of the winning paths could be equally desirable, some paths could have a higher probability of reaching a winning state than others. By keeping track of the number of visits MCTS makes with every path it takes, we can safely assume that winning paths with a higher number of visits are more likely to reach a winning state. However, if the path selection criterion focuses only on maximizing the reward value, it can repeatedly revisit the same paths while failing to discover new ones. To account for this, a selection policy must balance between exploration of new paths and exploitation of already visited paths. A common policy that can achieve this balance is the Upper Confidence Bound (UCB), proposed by Auer et al. (2002) for solving the multi-armed bandit problems, as shown in equation 2:

$$UCB = \frac{W_i}{S_i} + C\sqrt{\frac{2 \times lnS_p}{S_i}} \quad (2)$$

Where for a given state i, W_i represents the number of paths leading to a winning state, and S_i records the total number of paths from i. The first part of the equation, $\frac{W_i}{S_i}$, favors paths that have on average led to a winning state, whereas the second part of the UCB policy, $C\sqrt{\frac{2 \times lnS_p}{S_i}}$, favors unexplored paths. S_p is the total number of paths taken from the parent node, and C is an exploration hyperparameter. A higher C would increase exploration. UCB combined with MCTS is commonly known as the Upper Confidence Bound for Trees (UCT) (Browne et al., 2012). The final MCTS algorithm can be divided into four main stages:

1. Selection: Starting from a root node S_r, the UCB function from eq. 2 is recursively applied to select the next node to visit, until an unexpanded node is reached.

2. Expansion: A non-terminal leaf node is expanded by adding its immediate children. This corresponds to all the immediate actions that are possible from that state.

3. Simulation: From the current state, a random path ending with a terminal state is generated.

4. Backpropagation: Once a terminal node is reached, statistics including the reward value and the number of visits are backpropagated to the nodes of the current path.

Typically, MCTS runs until a predefined criterion is satisfied, e.g. a user-specified time or a maximum number of iterations. We adapt MCTS so that each node represents a token generated by GPT-2, where the possible actions k from a particular node S_i are from a top-k sampling process. Each full path represents a complete text example that is rewarded by equation 6.

3 Approach

In this work, we propose the Monte Carlo Text Generation (MCTG) method (section 3.2), alongside the Non-Targeted Text Generation (NTTG) method (section 3.4). In MCTG, we combine MCTS, a language generation model, a teacher, and a student classifier to create synthetic examples that can enhance the performance of the student in a KD setting. Here, the language model is conditioned to generate examples for which the predictions of the teacher and its student are as divergent as possible. The main components of our framework, as shown in Figure 2, include the language generation model (GPT-2), a teacher model, a student model, and the MCTS algorithm. Below we discuss the role of each component.

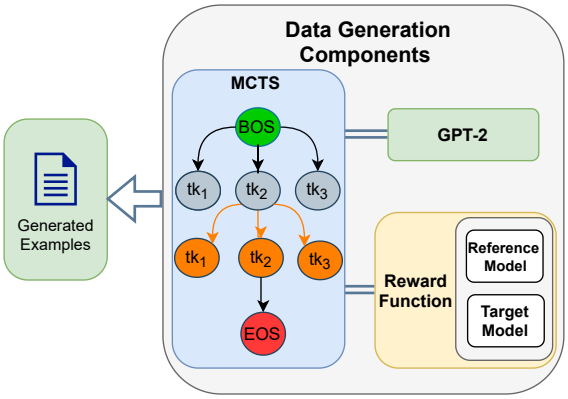

Figure 2: MCTS builds a tree from token sequences generated by GPT-2. Meanwhile, the teacher and student models work together to reward for completed paths.

3.1 Language Generation Model

Because the search tree is constructed by traversing from top to bottom, a unidirectional generative model can take tokens of parent nodes as input to predict candidates for the next node. This unidirectional behavior makes GPT-2 a good choice for our experiments[1]. To generate relevant data, GPT-2 is first fine-tuned on the initial training dataset. We prepare the data by first dropping the target labels, then merging the text examples, split by <|endoftext|>. The fine-tuned GTP-2 is then used to generate a token for each node, tk, in the constructed tree, as shown in Figure 2.

3.2 Monte Carlo Text Generation (MCTG)

We refer to the application of MCTS for text generation as MCTG. Starting from a root node, represented by a special token, $< |endoftext| >$, we sample the top k most probable tokens that come next in sequence. Having only started from a single root node, the tree is first expanded by adding the top k tokens as immediate children, making the depth of the tree equal to 2. Since at this stage all child nodes have equal weight, any one of them is selected. The next step is to start a simulation by

[1]We use Huggingface https://huggingface.co/

first concatenating the token of the selected node with those of its ancestor nodes. The resulting text is then passed to GPT-2 to obtain the probability distribution over the vocabulary, where the top k tokens are sampled. Given that this process takes place in the simulation stage, the UCB selection strategy is not applied; instead, we select a random token from the non-uniform distribution of the k tokens. In a standard MCTS implementation, the path that is navigated in the simulation phase is not necessarily recorded, but rather the statistics of it are, e.g. result and number of visits. However, to account for computational costs, we also track the generated text at the end of every simulation. It is important to note that the tokens generated during a simulation are not added as nodes to the tree, but are recorded separately. In this way we guarantee that while the growth of the tree is not affected during a simulation, we nevertheless retain the generated text examples with positive rewards. After a number of iterations, the statistics of the tree nodes will have changed, allowing higher impact of the UCB policy (e.q 2) in searching for paths that lead to examples with the higher reward values.

3.3 Reward Function

This component plays a key role in our approach as it dictates the usefulness of the generated data. In a student-teacher KD application, the aim is to find examples that the teacher model can label with higher certainty than its student. As entropy measures the uncertainty of a model's prediction for a particular example, the higher the entropy, the lower the confidence of the classifier in its prediction. This motivates us to generate examples that can be predicted with low entropy by the teacher, but high entropy by the student. Hence, when the difference in entropy between the two models is increased, the more important the generated example becomes for training the student. Given a generated text example x_u, the predicted probabilities from a model m are outputs of its $softmax$ layer:

$$P_m(y) = softmax(f(x_u)) \tag{3}$$

The entropy is thus:

$$H_n(P_m) = -\sum_{i=1}^{n} p_i \log_b p_i \cdot \frac{1}{\log_b n} \tag{4}$$

where the predicted probabilities $P_m = \{p_i; i = 1, ..., n\}$ for n labels. We take the difference in entropy between the student s, and the teacher t:

$$\Delta\text{ent} = \text{ent}_s - \text{ent}_t \tag{5}$$

The teacher's confidence and the student's lack of confidence in labeling an example are reflected in Δent. For predictions where the teacher's confidence is at its highest, and the student confidence at its lowest, Δent is maximized. Hence, examples with high Δent are more useful for distilling knowledge to the student model. For this reason, we only consider examples where Δent is positive. By finding paths that maximize Δent, GPT-2 is conditioned to generate examples that can be predicted with the lowest uncertainty by model t, but with the highest possible uncertainty by model s. Here we make the following assumptions: a) Examples that maximize the gap between entropy values are those most informative to the student model, yet can be confidently labeled by the teacher model; b) Training the student on informative examples can increase its performance. Since the objective is to find solutions that maximize the reward value, each generated path is rewarded by Δent. To further optimize the search process, we add the following refinements to the reward value.

$$reward = \begin{cases} 0, & \text{if \#tokens} < x, x \in \mathbb{Z}_{\geq 0} \\ 0, & \text{if } \Delta\text{ent} < 0 \\ -1, & \text{if task specific condition} \\ \Delta\text{ent}, & \text{otherwise} \end{cases} \tag{6}$$

Condition 1 in equation 6 is a heuristic that penalizes examples below a user-defined minimum number of tokens. Condition 2 minimizes the penalty for examples where the student model is more certain in its prediction than its teacher. Condition 3 eliminates cases based on a task-specific condition, see TREC-6 configuration in section 4.4. Finally, the fourth condition results in a positive reward when the teacher is more certain than the student.

3.4 Non-Targeted Text Generation (NTTG)

In NTTG, examples are generated without conditioning GPT-2 on the predictions of the student and the teacher classifiers. Instead, examples are generated purely on the probability distributions for the candidate tokens from GPT-2, then filtered on the conditions from equation 6. As in the MCTS simulation phase, examples are generated by applying top-k sampling on the outputs of GPT-2. In top-k sampling, the probability mass is redistributed over

the top k most probable choices. At each timestep, k candidate tokens are sampled based on the previously generated tokens. A token is then randomly selected from the k most likely candidates. A sequence is completed when a symbol indicating the end-of-sequence is selected, or when a user defined maximum sequence length is reached. Due to the randomness in selecting the next token, a different sequence can be produced whenever the generation process is repeated. Once enough examples are generated, the result data is cleaned by removing duplicates and short sequences, e.g. less than 3 tokens. Next, The entropy, e.q. 4, of both the teacher and student models is then computed for each remaining sample. We denote the student's entropy by ent_s and the teacher's entropy by ent_t. With equation 5, for each sample, we compute the difference of entropy between the teacher and its student, $\Delta\mathrm{ent}$. We then apply equation 6, and select the examples with $\Delta\mathrm{ent} > 0$.

4 Experiments

4.1 Baseline: Conditional BERT (C-BERT)

C-BERT baseline augments the training data by applying the 12-layer pretrained BERT model to predict a substitute word for a masked token (Wu et al., 2019). Each token in an input has a 10% probability of being masked, i.e., replaced by a BERT prediction (Kobayashi, 2018). Similar to our GPT-2 generation for the MCTG and NTTG experiments, the replacement token is selected from the top-20 tokens given BERT's probabilities.

4.2 Classification Models

We based our experiments on the pre-trained language models provided by Huggingface (Wolf et al., 2020). For the teacher, we used the 24-layer RoBERTa, and 2 variants of DistilRoBERTa (Sanh et al., 2019) for the student; the original 6-layer DistilRoBERTa, and a 3-layer DistilRoBERTa with half the layers removed. While 24-layer RoBERTa has 355 parameters, this is reduced to 82.1 million in 6-layer DistilRoBERTa, and then to 60.8 million in our 3-layer version. For each model, we appended a linear layer followed by a ReLU activation, a 0.1 dropout layer, and a linear output layer. We stabilized training by following the configurations suggested by Mosbach et al. (2020). That is, we applied the ADAM optimizer (Kingma and Ba, 2014) with a bias correction to avoid vanishing gradients in early training steps. We then trained for 40 epochs with a learning rate of 2×10^{-5} that linearly increases in the first 10% of the total training steps and linearly decays to zero afterward.

4.3 Datasets

In an attempt to evaluate our approach under different settings, we considered datasets of multiple sequence classification tasks. We simulate scarce data settings by artificially creating an initial training set of randomly selected seed examples from the available training data. The language generation model, and the student and teacher classifiers, are then fine-tuned on the sampled training data. The SST-2 (Socher et al., 2013), and Yelp-2 (Zhang et al., 2015) datasets are for binary sentiment classification. TREC-6 (Li and Roth, 2002) is a 6-label question classification dataset, and DBpedia (Zhang et al., 2015) is a 14-label topic classification dataset. Finally, the Microsoft Research Paraphrase Corpus (MRPC), is for a sentence-pair classification task (Dolan and Brockett, 2005), where a model has to predict if the two sentences are semantically equivalent or not. We note that other data augmentation works have fine-tuned GPT-2 on SST-2, Yelp and TREC (Kumar et al., 2020; Anaby-Tavor et al., 2020; Feng et al., 2020). For the SST-2 experiments, from a total of 67,349 samples, we randomly sampled 30 examples per label from the GLUE SST training data (Wang et al., 2018), and evaluated on the provided development set. The full training dataset for Yelp-2 contains 560K samples. Here, we sampled 20 examples per label, making a total of 40 training samples. We then evaluated on the 25,000 test samples. TREC-6 contains 5,452 training examples and 500 testing examples; as this training dataset is not balanced, we randomly sampled less than 1.5 percent of the data for each class, giving us a total of 76 examples. For DBpedia, we sampled just 3 examples per label, from a total of 560K instances, making a seed dataset of 42 training examples. We then evaluated on the 70,000 test samples. Finally, MRPC consists of 3,668 training examples of which we sampled 600 per label, making a total of 1,200 training samples. Our evaluations were on the 1,725 samples test set.

4.4 Configurations

We configured both MCTG and NTTG to 5k iterations, and the top-k sampling to $k = 20$ in all our experiments. We also added a pruning criterion to limit the maximum length of any sequence to

Task	Teacher	Student		Student (Post Distillation)			Student
		layers	Start	MCTG	NTTG	C-BERT	Upper-Bounds
SST-2 # Samples	89.9 (60)	6-layers	78.2 (60)	**86.1** (1508)	85.3 (402)	83.1 (4718)	91.4 (67349)
		3-layers	68.2 (60)	**82.3** (3104)	80.7 (734)	75.5 (4718)	89.4 (67349)
DBpedia-2 # Samples	92.3 (42)	6-layers	80.4 (42)	92 (3976)	**93.1** (1602)	88.8 (5029)	99.3 (560K)
		3-layers	41.1 (42)	**91.6** (3940)	90.6 (3271)	84.1 (5029)	99.2 (560K)
TREC-6 # Samples	89 (76)	6-layers	80 (76)	**88** (2508)	83.4 (537)	80 (4874)	96.8 (5452)
		3-layers	62 (76)	**82** (3133)	81 (594)	78.4 (4874)	95.8 (5452)
MRPC # Samples	84.5 (1200)	6-layers	77.9 (1200)	81.5 (4252)	81.6 (3599)	**82.2** (6200)	85.6 (3668)
		3-layers	69 (1200)	**77.9** (3118)	77.4 (4156)	73.3 (6200)	78 (3668)
YELP-2 # Samples	82.6 (40)	6-layers	**85.2** (40)	80.7 (3124)	81.1 (2074)	81.8 (5040)	95.9 (560K)
		3-layers	73.9 (40)	**78.9** (4278)	77.6 (2828)	76.9 (5040)	94.9 (560K)

Table 1: Teacher-Student Results (in percent, numbers of added examples in parentheses below). The Upper-Bounds are computed after training the student models on the full training datasets without synthetic data.

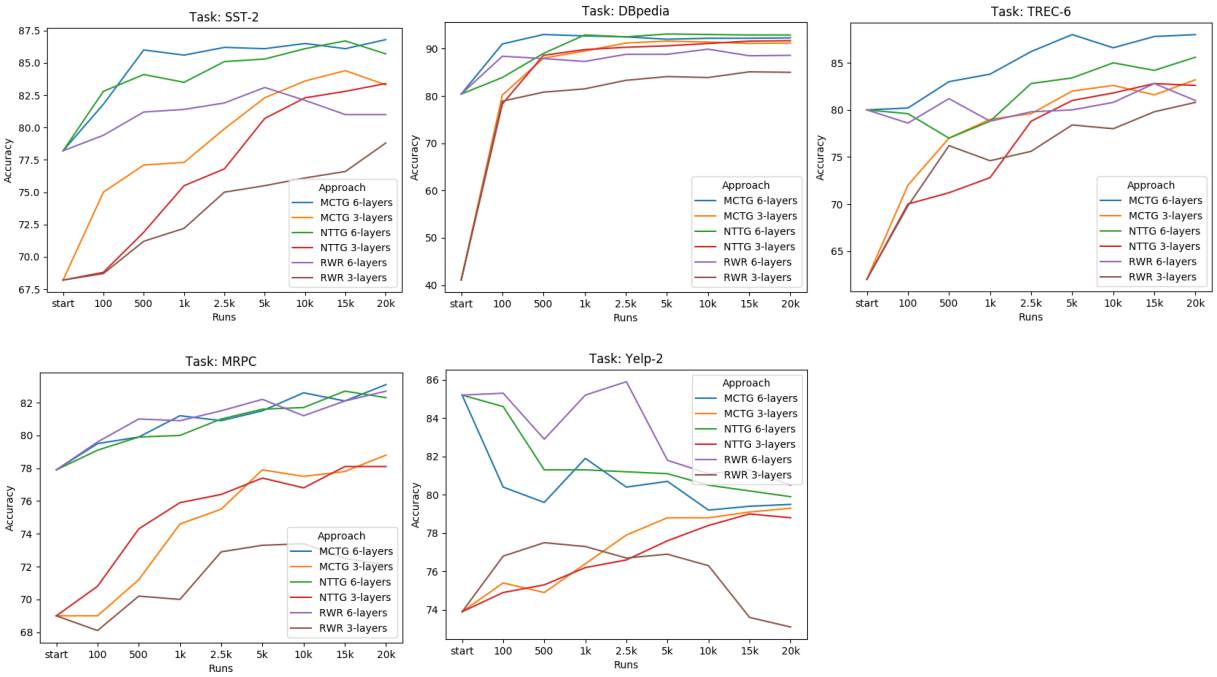

Figure 3: Student's test accuracy after 100, 500, 1k, 2.5k, 5k, 10k, and 20k iterations.

120 tokens. For MCTG, we set the UCB constant from eq. 2 to $C = 3$. As for the reward function, only for the TREC-6 experiments, we added a heuristic (see 'task specific condition' in eq. 6) to condition the first token to be a question word by returning -1 if it was not 'what', 'where', 'when', 'who', 'which', 'why', or 'how'. For each task, we made sure the generated text is of appropriate format for the RoBERTa classification models. This meant, reward = 0 for GPT-2 outputs that were not in the format <|endoftext|>$x_1, \ldots x_N$ <|endoftext|> for single input sentence tasks; TREC-6, SST-2, Yelp, and DBpedia. As for MRPC, where the input is 2 sentences, the generated data has to be of format <|endoftext|> $x_1, \ldots x_N, [\text{SEP}], y_1, \ldots y_N$ <|endoftext|>. Where $x_1 \ldots x_N$ and $y_1 \ldots y_N$ are sequences of tokens. After data was generated, we selected the examples with a positive reward, see section 3.3. To avoid high imbalance for the binary datasets, we limited the number of added examples to the size of the minority class. For the multiclass datasets, we limited the selection over the median from the distribution of generated examples per label. We then added the selected data to the initial training data, to form a transfer set. This new transfer set consists of the initial training samples with the generated data pseudo-labeled by the teacher.

4.5 Results

Results for the teacher-student knowledge transfer experiments are in Table 1. We show the test accuracy of both the teacher and student (Pre-Distillation) on the sampled data from section 4.3, and the student after it has been trained on the sampled data combined with the generated data (Post-Distillation). We also compute an upper bound performance by training the 3-layer and 6-layer DistilRoBERTa models on the full training datasets, mentioned in section 4.3. This is to give us an estimate of the performance that can be achieved with as much non-synthetic data as possible. Underneath each accuracy score, in parentheses, is the size of the training data (# Training examples). For the teacher and the student prior to distillation (labeled Start in the table), the data sizes are of the initial training sets, described in section 4.3. As explained in section 4.2, the teacher model is a 24-layer RoBERTa, and the student model, is either a 6-layer DistilRoBERTa, or a 3-layer DistilRoBERTa. The "Start" accuracy is achieved after training only on the initial dataset. For example, the 24-layer RoBERTa trained on the SST-2 dataset, of 60 examples, produced a test score of 89.9. This is a much higher result compared to the accuracies of the 6-layers and 3-layers DistilRoBERTa models that were trained on the same dataset, scoring only 78.2 and 68.2 respectively. We then applied MCTG, from section 3.2, and NTTG, from section 3.4 to generate distillation data. We fixed the total number of iterations to 5k. After each iteration, a sample is generated and only becomes a candidate for distillation if its Δent (equation 5) is positive, and receives a positive reward as per the conditions in equation 6. This means less samples are added to the transfer set from the total generated data.

4.6 Discussion

Overall, results in Table 1 show that our approach works well with either MCTG or NTTG. It is evident that a good teacher can always increase the performance of its student, provided that enough examples achieve a positive Δent (equation 5). This shows that regardless of the generation method, equation 5 remains a key component to our approach. Overall, our approaches, NNTG and MCTG, lead to better performance improvements over the C-BERT baseline. Only in MRPC, the results are similar for the 6-layer student, which could be attributed to a lower performance gap between the student and the teacher. Overall, when compared to the upper-bound results from training the distilled models on the full training datasets, the performances we achieve with distillation are not far off. This might indicate that there is potential room for improvement. With this in mind, the distilled models only utilized a fraction of the full training datasets under each task, as explained in section 4.3. In Figure 3 for each task, we plot the student's performance after iterations 100, 500, 2.5k, 5k, 10k, and 20k. As the number of iterations increase, more examples are generated and thus better performance can be achieved. However, we notice that at a certain point, the increase in performance plateaus. To show the importance of a good teacher, we selected a dataset (Yelp-2), in which the 6-layer student outperforms its teacher. Here, the teacher's overconfidence in incorrect predictions resulted in noisy data, that degraded its student's performance. This shows that the student can only improve as much as its teacher is able to provide good labels. We investigate the stability of the student model, pre- and post-distillation, by

Task	Approach	3-Layers		6-Layers	
		Start	20K	Start	20K
SST-2	MCTG	68.2(±1.67)	83.7(±0.32)	81.85(±1.26)	86.89(±0.396)
	NTTG	67.7(±1)	82(±0.69)	81.3(±0.869)	86.9(±0.4)
TREC-6	MCTG	59.86(±4.97)	83(±0.7)	78.6(±1.77)	88(±0.51)
	NTTG	60.15(±4.68)	82.6(±0.84)	79.08(±1.65)	86.3(±0.8)
MRPC	MCTG	68.5(±2.28)	77.5(±0.55)	79.7(±0.91)	82.7(±0.4)
	NTTG	68.2(±2.69)	77.6(±0.49)	79.5(±0.71)	81.5(±0.7)
DBpedia	MCTG	52.18(±5.06)	91.3(±0.16)	86(±3.7)	92.3(±0.059)
	NTTG	55.7(±4.7)	91.7(±0.13)	85.9(±3.8)	92.8(±0.08)
Yelp-2	MCTG	72.27(±2.84)	79.2(±0.183)	83.29(±1.34)	79.4(±0.175)
	NTTG	72.4(±2.3)	78.5(±0.167)	83.95(±1.07)	79.9(±0.22)

Table 2: Mean(± standard deviation), of test accuracy for 10 student model (3-layers and 6-layers) instances, trained on the initially sampled data and the pseudo-labeled data from the 20k MCTG and NTTG runs.

TREC-6 Examples	Teacher	Student
What is virtual reality?	DESC	ENTY
What language was originally spoken by the Indians?	ENTY	LOC
Where is your favourite golf course?	LOC	DESC
SST-2 Examples		
a trip from good to bad	NEG	POS
the kind of script worth watching	POS	NEG
a step down from her best years.	NEG	POS

Table 3: Examples of data generated for TREC-6 and SST-2. Wrongly predicted labels are colored in red.

running 10 training instances on the initial data and on the transfer set from the 20k run. In Table 2, we show the mean of the test accuracy of 10 trained instances of the student model. These results are consistent with Table 1. Overall, the augmented data leads to better and more stable models, indicated by the higher accuracy and lower variance. Finally, in Table 3, we show some generated data from the TREC-6 and SST-2 experiments. All four examples are remarkably grammatical, natural, and well-formed.

5 Conclusion

In this paper, we presented an approach for generating text data in order to improve knowledge distillation on small datasets. By selecting examples predicted with the lowest uncertainty by the teacher and the highest uncertainty by the student, we were able to improve the student's performance, sometimes almost to the level of the teacher. Considering the results, we could argue that reward-based language generation can complement or even substitute for heuristic data augmentation approaches in knowledge distillation. We believe that our approach can serve as a baseline for reward-based textual data generation in small data settings. This will hopefully motivate future research to further explore reward-based generation methods.

Acknowledgments

Husam Quteineh acknowledges the support of the Business and Local Government Data Research Centre BLG DRC (ES/S007156/1) funded by the Economic and Social Research Council (ESRC).

References

Ateret Anaby-Tavor, Boaz Carmeli, Esther Goldbraich, Amir Kantor, George Kour, Segev Shlomov, Naama Tepper, and Naama Zwerdling. 2020. Do not have enough data? deep learning to the rescue! In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pages 7383–7390.

Broderick Arneson, Ryan B Hayward, and Philip Henderson. 2010. Monte carlo tree search in hex. *IEEE Transactions on Computational Intelligence and AI in Games*, 2(4):251–258.

Peter Auer, Nicolo Cesa-Bianchi, and Paul Fischer. 2002. Finite-time analysis of the multiarmed bandit problem. *Machine learning*, 47(2):235–256.

C. B. Browne, E. Powley, D. Whitehouse, S. M. Lucas, P. I. Cowling, P. Rohlfshagen, S. Tavener, D. Perez, S. Samothrakis, and S. Colton. 2012. A survey of monte carlo tree search methods. *IEEE Transactions on Computational Intelligence and AI in Games*, 4(1):1–43.

Jang Hyun Cho and Bharath Hariharan. 2019. On the efficacy of knowledge distillation. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 4794–4802.

Michael Chung, Michael Buro, and Jonathan Schaeffer. 2005. Monte carlo planning in rts games. In *CIG*. Citeseer.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*.

William B Dolan and Chris Brockett. 2005. Automatically constructing a corpus of sentential paraphrases. In *Proceedings of the Third International Workshop on Paraphrasing (IWP2005)*.

Stefan Edelkamp, Max Gath, Christoph Greulich, Malte Humann, Otthein Herzog, and Michael Lawo. 2016. Monte-carlo tree search for logistics. In *Commercial Transport*, pages 427–440. Springer.

Angela Fan, Mike Lewis, and Yann N. Dauphin. 2018. Hierarchical neural story generation. *CoRR*, abs/1805.04833.

Steven Y Feng, Varun Gangal, Dongyeop Kang, Teruko Mitamura, and Eduard Hovy. 2020. Genaug: Data augmentation for finetuning text generators. *arXiv preprint arXiv:2010.01794*.

Jianping Gou, Baosheng Yu, Stephen J Maybank, and Dacheng Tao. 2021. Knowledge distillation: A survey. *International Journal of Computer Vision*, pages 1–31.

Byeongho Heo, Minsik Lee, Sangdoo Yun, and Jin Young Choi. 2019. Knowledge transfer via distillation of activation boundaries formed by hidden neurons. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, pages 3779–3787.

Geoffrey Hinton, Oriol Vinyals, and Jeff Dean. 2015. Distilling the knowledge in a neural network. *arXiv preprint arXiv:1503.02531*.

Jeremy Howard and Sebastian Ruder. 2018. Universal language model fine-tuning for text classification. *arXiv preprint arXiv:1801.06146*.

Xiaoqi Jiao, Yichun Yin, Lifeng Shang, Xin Jiang, Xiao Chen, Linlin Li, Fang Wang, and Qun Liu. 2019. Tinybert: Distilling bert for natural language understanding. *arXiv preprint arXiv:1909.10351*.

Diederik P Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*.

Sosuke Kobayashi. 2018. Contextual augmentation: Data augmentation by words with paradigmatic relations. *arXiv preprint arXiv:1805.06201*.

Levente Kocsis and Csaba Szepesvári. 2006. Bandit based monte-carlo planning. In *European conference on machine learning*, pages 282–293. Springer.

Varun Kumar, Ashutosh Choudhary, and Eunah Cho. 2020. Data augmentation using pre-trained transformer models. *arXiv preprint arXiv:2003.02245*.

Zhenzhong Lan, Mingda Chen, Sebastian Goodman, Kevin Gimpel, Piyush Sharma, and Radu Soricut. 2020. Albert: A lite bert for self-supervised learning of language representations.

Xin Li and Dan Roth. 2002. Learning question classifiers. In *Proceedings of the 19th international conference on Computational linguistics-Volume 1*, pages 1–7. Association for Computational Linguistics.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Roberta: A robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692*.

Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013. Efficient estimation of word representations in vector space. *arXiv preprint arXiv:1301.3781*.

Marius Mosbach, Maksym Andriushchenko, and Dietrich Klakow. 2020. On the stability of fine-tuning bert: Misconceptions, explanations, and strong baselines. *arXiv preprint arXiv:2006.04884*.

Phi-Vu Nguyen, Ali Ghezal, Ya-Chih Hsueh, Thomas Boudier, Samuel Ken-En Gan, and Hwee Kuan Lee. 2016. Optimal processing for gel electrophoresis images: applying monte carlo tree search in gelapp. *Electrophoresis*, 37(15-16):2208–2216.

Jeffrey Pennington, Richard Socher, and Christopher D Manning. 2014. Glove: Global vectors for word representation. In *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*, pages 1532–1543.

Matthew E Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer. 2018. Deep contextualized word representations. *arXiv preprint arXiv:1802.05365*.

Husam Quteineh, Spyridon Samothrakis, and Richard Sutcliffe. 2020. Textual data augmentation for efficient active learning on tiny datasets. Association for Computational Linguistics.

Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. 2019. Language models are unsupervised multitask learners. *OpenAI blog*, 1(8):9.

Adriana Romero, Nicolas Ballas, Samira Ebrahimi Kahou, Antoine Chassang, Carlo Gatta, and Yoshua Bengio. 2014. Fitnets: Hints for thin deep nets. *arXiv preprint arXiv:1412.6550*.

Hassan Sajjad, Fahim Dalvi, Nadir Durrani, and Preslav Nakov. 2020. Poor man's bert: Smaller and faster transformer models. *arXiv preprint arXiv:2004.03844*.

Victor Sanh, Lysandre Debut, Julien Chaumond, and Thomas Wolf. 2019. Distilbert, a distilled version of bert: smaller, faster, cheaper and lighter. *arXiv preprint arXiv:1910.01108*.

Rico Sennrich, Barry Haddow, and Alexandra Birch. 2015. Improving neural machine translation models with monolingual data. *arXiv preprint arXiv:1511.06709*.

Connor Shorten and Taghi M Khoshgoftaar. 2019. A survey on image data augmentation for deep learning. *Journal of Big Data*, 6(1):60.

David Silver, Aja Huang, Chris J Maddison, Arthur Guez, Laurent Sifre, George Van Den Driessche, Julian Schrittwieser, Ioannis Antonoglou, Veda Panneershelvam, Marc Lanctot, et al. 2016. Mastering the game of go with deep neural networks and tree search. *nature*, 529(7587):484–489.

Richard Socher, Alex Perelygin, Jean Wu, Jason Chuang, Christopher D Manning, Andrew Y Ng, and Christopher Potts. 2013. Recursive deep models for semantic compositionality over a sentiment treebank. In *Proceedings of the 2013 conference on empirical methods in natural language processing*, pages 1631–1642.

Siqi Sun, Yu Cheng, Zhe Gan, and Jingjing Liu. 2019. Patient knowledge distillation for bert model compression. *arXiv preprint arXiv:1908.09355*.

Zhiqing Sun, Hongkun Yu, Xiaodan Song, Renjie Liu, Yiming Yang, and Denny Zhou. 2020. Mobilebert: a compact task-agnostic bert for resource-limited devices. *arXiv preprint arXiv:2004.02984*.

Raphael Tang, Yao Lu, Linqing Liu, Lili Mou, Olga Vechtomova, and Jimmy Lin. 2019. Distilling task-specific knowledge from bert into simple neural networks. *arXiv preprint arXiv:1903.12136*.

Antti Tarvainen and Harri Valpola. 2017. Weight-averaged consistency targets improve semi-supervised deep learning results. *CoRR*, abs/1703.01780.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Lukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. *arXiv preprint arXiv:1706.03762*.

Alex Wang, Yada Pruksachatkun, Nikita Nangia, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel R Bowman. 2019. Superglue: A stickier benchmark for general-purpose language understanding systems. *arXiv preprint arXiv:1905.00537*.

Alex Wang, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel R Bowman. 2018. Glue: A multi-task benchmark and analysis platform for natural language understanding. *arXiv preprint arXiv:1804.07461*.

Jason Wei and Kai Zou. 2019. Eda: Easy data augmentation techniques for boosting performance on text classification tasks. *arXiv preprint arXiv:1901.11196*.

Thomas Wolf, Julien Chaumond, Lysandre Debut, Victor Sanh, Clement Delangue, Anthony Moi, Pierric Cistac, Morgan Funtowicz, Joe Davison, Sam Shleifer, et al. 2020. Transformers: State-of-the-art natural language processing. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 38–45.

Xing Wu, Shangwen Lv, Liangjun Zang, Jizhong Han, and Songlin Hu. 2019. Conditional bert contextual augmentation. In *International Conference on Computational Science*, pages 84–95. Springer.

Zhilin Yang, Zihang Dai, Yiming Yang, Jaime Carbonell, Ruslan Salakhutdinov, and Quoc V Le. 2019. Xlnet: Generalized autoregressive pretraining for language understanding. *arXiv preprint arXiv:1906.08237*.

Junho Yim, Donggyu Joo, Jihoon Bae, and Junmo Kim. 2017. A gift from knowledge distillation: Fast optimization, network minimization and transfer learning. In *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 7130–7138.

Xiang Zhang, Junbo Zhao, and Yann LeCun. 2015. Character-level convolutional networks for text classification. *Advances in neural information processing systems*, 28:649–657.

Boosting Code Summarization by Embedding Code Structures

Jikyeong Son, Joonghyuk Hahn, Hyeon-Tae Seo and **Yo-Sub Han**
Department of Computer Science, Yonsei University
Seoul, Republic of Korea
{jikyeong.son, greghahn, dchs504, emmous}@yonsei.ac.kr

Abstract

Recent research on code summarization relies on the structural information from the abstract syntax tree (AST) of source codes. It is, however, questionable whether it is the most effective to use AST for expressing the structural information. We find that a program dependency graph (PDG) can represent the structure of a code more effectively. We propose PDG Boosting Module (PBM) that encodes PDG into graph embedding and the framework to implement the proposed PBM with the existing models. PBM achieves improvements of 6.67% (BLEU) and 7.47% (ROUGE) on average.

We then analyze the experimental results, and examine how PBM helps the training of baseline models and its performance robustness. For the validation of robustness, we measure the performance of an out-of-domain benchmark dataset, and confirm its robustness. In addition, we apply a new evaluation measure, SBERT score, to evaluate the semantic performance. The models implemented with PBM improve the performance of SBERT score. This implies that they generate summaries that are semantically more similar to the reference summary.

1 Introduction

In the early stage of generating code summaries, researchers adopted information retrieval techniques (Marcus et al., 2004; Poshyvanyk and Marcus, 2007; Haiduc et al., 2010) to capture source code semantics. However, code summaries produced by such techniques are often inaccurate to use in practice (Wong et al., 2015). With the help of deep learning, researchers proposed the neural machine translation (NMT) frameworks that automatically produce summaries from source code (Iyer et al., 2016; LeClair et al., 2019; Liang and Zhu, 2018; Sridhara et al., 2010). Rather than using code sequence as a sole input, several models built upon the Transformer (Vaswani et al., 2017) used additional data structures and information (Lin et al., 2021; Shi et al., 2021; Choi et al., 2021) to learn obscure features that otherwise would be discarded.

Recent studies use pretrained models and implementations of graph structures to improve the performance of code summarization. Pretrained models are built upon training a huge quantity of benchmark datasets within a long period. CodeBERT (Feng et al., 2020) and CodeT5 (Wang et al., 2021) are popular pretrained models for various code-related tasks. The graph embedding is regarded as effective on providing code semantics; especially abstract syntax tree (AST) is the most popular supplement types for reflecting the hierarchical structure of codes. Several researchers used ASTs and improved the performance of source code summarization (Alon et al., 2019; Zhang et al., 2019; Shido et al., 2019; LeClair et al., 2020). Furthermore, combining with the Transformer, there are a few improved models including mAST+GCN (Choi et al., 2021), BASTS (Lin et al., 2021), CAST (Shi et al., 2021) and SiT (Wu et al., 2021).

While ASTs are widely used to capture code structure information, they cannot capture global information between tokens well due to the deep depths (Lin et al., 2021; Shi et al., 2021; Zhang et al., 2019). Thus, recent researches consider graphs other than ASTs to capture structural information. Lin et al. (2021) pretrain the model after dividing the code based on control flow graph (CFG), and Gao et al. (2021) proposed a method of capturing the global structure by learning the data flow relationship between variables. Liu et al. (2021) utilize a new type of graph called code property graph (CPG), which combines CFG and AST, and propose a hybrid GNN using CPG.

When models utilize AST as auxiliary information, structural information is treated and delivered by token level representation. Many researchers are performed through extracting structural information from the token representations. However, such

Proceedings of the 29th International Conference on Computational Linguistics, pages 5966–5977
October 12–17, 2022.

token representations of ASTs fail to provide semantics of statements and predicates (Zhang et al., 2019). We perform code summarization by capturing the structural information through a program dependency graph (PDG) to solve such problem.

We implement the encoder module that takes PDGs of source codes as inputs to several baseline models, and evaluate the improvement from our graph module for the summary generation performance of each model. Baseline models we perform experiments are SiT (Wu et al., 2021) that applies AST on a transformer and CodeBERT (Feng et al., 2020), including the Transformer (Ahmad et al., 2020).

The experimental results of our implementation show the performance improvements of average corpus-BLEU 6.67% and Rouge-L 7.47%. However, these improvements are not sufficient to demonstrate that our implementation accurately captures structural information. Thus, we further ask the following questions based on the initial experimental results.

RQ1: Is the structural code information such as graph embedding indeed helpful for generating a code summary?

RQ2: What difference does the proposed graph structure have compared to the popularly used AST and what is better between two graph structures?

RQ3: Does the proposed model show the robust performance for out-of-domain data?

Our code is available at `https://github.com/sjk0825/coling2022`.

2 Related Works

2.1 Sequential-based Approach

Iyer et al. (2016) first proposed a method using a neural network for code summarization. Wei et al. (2019) proposed a dual framework that uses the correlation between code summarization and code generation tasks. Hu et al. (2018b) proposed a summary method using API information as well as sequence information. Ahmad (Ahmad et al., 2020) proposed a method to effectively capture the long-range dependency of a code sequence using the Transformer.

2.2 Graph-based Approach

Wan et al. (2018) applied reinforcement learning for code summarization after giving AST as sequenced information. LeClair et al. (2019) proposed a method that provides sequential and AST to independent GRUs. Hu et al. (2018a) proposed traversing the AST in a structure-based traversal method for code summarization. Transformer-based method for learning has also been proposed. Choi et al. (2021) proposed AST representing through GCN based on the Transformer. Wu et al. (2021) also suggested using a transformer-based multiview graph.

For other graph types such as CFG and PDG, the nodes are in a statement level varying from AST's nodes in a token level. Such graphs are used for a code representation method in several studies. Lin et al. (2021) proposed a method of pretraining syntax information after splitting based on a control flow of CFG. Yamaguchi et al. (2014) showed CPG combined with CFG and AST. Liu et al. (2021) performed code summarization of CPG through Hybrid GNN. We also use PDG for code structural representation to obtain the structure information of the code.

2.3 Pretrained Model-based Approach

CodeBERT is a bimodal pretrained model that performs the NL-PL task. It is a model for Masked Language Modeling and Replaced Token Detection tasks, pretrained with the dataset CodeSearchNet (Husain et al., 2019). GraphCodeBERT (Guo et al., 2021) is the first pretrained model using data flow. The model was constructed through Masked Language Modeling, Edge Prediction, and Node Alignment, and the dataset is CodeSearchNet like CodeBERT. CodeT5 (Wang et al., 2021) is also an integrated model of encoder-decoder pretrained for code related tasks. It was pretrained on tasks such as Identifier-aware Denoising Pretraining, Identifier Tagging, and Masked Identifier Prediction through CodeSearchNet and BigQuery dataset.

3 Methodology

We propose PBM (PDG Boosting Module) which improves the capability of capturing the structure information by embedding PDG to the encoder. In this section, we explain what PDG is and then, show how our PBM embeds PDG. Finally, we demonstrate the implementation of our PBM with the baseline models of the code summarization task. We illustrate the overview of the framework implementation of PBM in Figure 2. Our module applies the PDG to improve the baseline models but for better analysis, we also develop a module for ASTs that act the same as our PBM and show

the experimental results in Section 4.

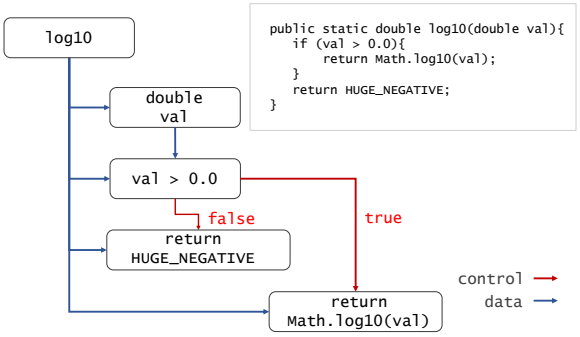

Figure 1: An example of PDG corresponding to the JAVA code instance. We represent control flow of the code with red colored edges and data flow with blue colored edges.

3.1 Program Dependency Graph

PDG is a type of graph to represent the dependency flow of code in statement level, proposed by Ferrante et al. (1987). The graph consists of statement nodes and predicated nodes, which express operators and operands of a source code, respectively. Edges between nodes express dependencies including data dependency and control dependency. Depicted in Figure 1, edges in blue color show a data flow among variables and represent data dependency. Similarly, edges in red color represent control dependency that corresponds to the dependency influenced by the values of predicate nodes. In Figure 1, control dependencies starting from a conditional statement '`val > 0.0`' depend on the value of '`val`'. Unlike AST, each node of PDG contains the partial semantics of a source code in statement level and each edge shows a connection between statements. We suspect that such tendency can express the structural information of the source code effectively and helps the models to train better for generating code summaries. The AST of the source code in Figure 1 is in Appendix A.

3.2 Graph Embedding

Formally, we represent the input graph as $G(N, E_c, E_d)$, where N is the set of PDG nodes, E_c is the set of edges for control dependency and E_d is the set of edges for data dependency. An edge $(u, v) \in E_i$ for $i \in \{c, d\}$ denotes an edge from u to v, where u and v are nodes of G. Given a PDG $G(V, E)$, where $E = E_c \cup E_d$, each edge is represented as an embedding matrix M of size $|N| \times |N|$. Each node is also represented with an embedding matrix of the same size to M.

We propose a graph embedding module that takes an extracted PDG from source codes as inputs. The extracted PDGs divide codes into statement level that are embedded as nodes and edges from the PDGs express the control and data dependencies between nodes. Then, the output of our module is concatenated with the output of a baseline model encoder. The detailed implementation is provided in Section 3.3.

Node Encoder A node encoder extracts structural information from the input graph. The encoder follows the structure of a baseline model encoder and takes the same approach of a baseline for encoding inputs. The difference from the encoder of baselines is that our node encoder calculates the attention of Key (K), Query (Q) and Value (V) that comes from the program dependency.

We present an attention equation for learning structural information from the graph embeddings. Given a sequence, let N be the set of nodes that consists of PDG of the sequence. Then, $node \in N$ is a node from the PDG and $node_e$ is an embedding of $node$. The following equation of $N\text{-}Att$ is the attention function of the node encoder. Note that K_e, Q_e, and V_e are pooled node representations of $node_e$ and d_K is the dimension of K_e. The node encoder encodes each $node_e$ as $Node_e$. When the size of N is n, the node encoder outputs node representations N_e, where $N_e = (Node_{1_e}, \ldots, Node_{n_e})$.

$$N\text{-}Att(Q_e, K_e, V_e) = softmax(\frac{E * Q_e K_e^\top}{\sqrt{d_k}})V_e,$$
$$E = E_c + E_d.$$

Node Pooler The node pooler is the process of pooling tokens within the same node through a given sequence and statement mask. The sequence of token embedding is represented as seq_e. The node pooler takes seq_e as an input and outputs its corresponding node embedding, $node_e$. $MASK^{|seq_e|}$ consists of one hot vector and W is a trainable weight. t is a token in sequence and k is a sequence length. The following equation demonstrates the procedure of node pooler.

$$node_e = ReLU((MASK * seq_e) * W),$$
$$MASK = \begin{cases} 1 \text{ if } t_j \in node \text{ for } j = 1, \ldots, k \\ 0 \text{ otherwise.} \end{cases}$$

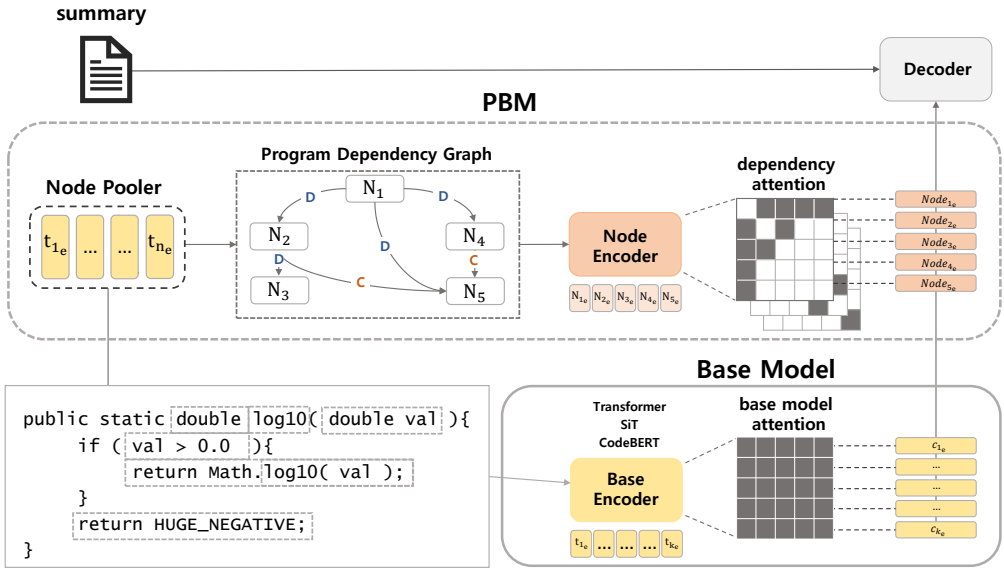

Figure 2: An architecture of PBM. Dashed boxes from the code snippet represent nodes.

3.3 PDG Boosting Module (PBM)

Our PBM module improves the performance of generating source code summaries by combining with baseline models. We use the Transformer (Ahmad et al., 2020), SiT (Wu et al., 2021), CodeBERT (Feng et al., 2020) as baseline models. The encoder of each baseline model is based on a transformer and emits token level output. We show how the separate embeddings of a sequence and graph for a code instance proceeds and combines in PBM.

Combine Encoder The input sequence consists of $(t_1, \ldots, t_k)$ tokens. The token passes through the base encoder ($B\text{-}encoder$) with the embedding vector $(t_{1_e}, \ldots, t_{k_e})$ of tokens and output $C_e = (c_{1_e}, \ldots, c_{k_e})$ containing sequence information. Token representations are concatenated with $N_e = (Node_{1_e}, \ldots, Node_{n_e})$ of the PDG module. Let k be the length of code sequence and n be the length of node sequence.

$$C_e = B\text{-}encoder(t_{1_e}, \ldots, t_{k_e}),$$
$$PBM = Concat([C_e; N_e]).$$

Decoder The decoder for PBM is dependent on baseline models we combine with. As the encoder of PBM combines both the sequential information and node information, the decoder takes the attention for both information to the target summaries. Figure 2 illustrates the full architecture of the PBM module and how PBM is connected to a baseline model.

4 Experiments and Analysis

Our experiment uses two benchmark datasets for the code summarization task. The first is CCSD (Liu et al., 2021) which is the dataset of C programs and the next is TL-CodeSum (Hu et al., 2018b) which is the dataset for JAVA programs. The details of each dataset are illustrated in Table 1.

Dataset	TL-CodeSum	CCSD
Train	69,708	84,316
Valid	8,714	4,432
Test	8,714	4,093
Out-domain Test	-	2,440

Table 1: Statistics on the number of data for the benchmark datasets.

The evaluation metrics are corpus-BLEU (Papineni et al., 2002) and ROUGE-L (Lin, 2004) score that are widely used for the verification of code summarization performance. We denote them as BLEU and ROUGE. Additionally, we use SBERT (Reimers and Gurevych, 2019) score to address the semantic performance that cannot be measured by the prior two metrics.

As we apply PBM to baseline models, one experiment is to compare the performance improvements by adding PBM to baseline models. Another experiment is to check which graph type is more suitable for our task. For the second question, we consider two graph types, AST and PDG, each of which is

constructed from data via joern[1] and srcml[2].

4.1 Baselines

PBM compare with four code summarization models that have neural network archtecture. Our baseline models include models based on the architecture of an RNN and a transformer. We also take a pretrained model as one of the baseline models.

Seq2Seq is based on the recurrent neural network architecture. Iyer et al. (2016) proposed a code summarization using Seq2Seq.

Transformer (Ahmad et al., 2020) is consturcted from a transformer (Vaswani et al., 2017)-based model using copy mechanism and relative positional encoding. Through self-attention of a transformer, the long-range dependency of code is effectively captured.

CodeBERT (Feng et al., 2020) is a pretrained encoder model with PL-NL bimodal. CodeBERT supports code-related downstream tasks including the code documentation generation task. We reproduce the CodeBERT-base model.

SiT (Wu et al., 2021) is a model trained with multiview on the structure of codes. Multiview includes AST, data dependency and statement. Multiview is trained through weighted attention at the token level.

4.2 Evaluation Metrics

Our analysis relies on BLEU and ROUGE as evaluation metrics that are popularly used in recent studies. These metrics check how the summaries capture the actual words that are used in the reference summaries. As the metrics only check whether a token or a sequence of tokens are the same, researchers argue about their reliability (Reiter, 2018; Mathur et al., 2020). Therefore, we also use another metric, SBERT (Reimers and Gurevych, 2019) score, to measure how the generated summaries capture the semantics of the source code.

BLEU (Papineni et al., 2002) measures the performance of predicted summaries through n-gram comparison with reference. The average performance is measured for the range of n to 1-4.

ROUGE (Lin, 2004) is an n-gram measurement method based on recall. The Rouge-L we use is the F-measure of prediction and reference based on the longest common sequence.

SBERT (Reimers and Gurevych, 2019) is a siamese network using pretrained BERT and measures the similarity between two sentences with a fixed sentence representation. We use the checkpoint, `all-mpnet-base-v2 mode` for the evaluation.

4.3 Experimental Setup

Hyperparameter Generally, we follow the same hyperparameter settings of baseline models to reproduce performance of the considered models. For adding our PBM to the baseline models, we set the size of the PBM layer as the size of an encoder of the corresponding baseline encoder. However, as we run experiments with multiple baseline models and the models attached with PBM, we regulate some hyperparameters such as batch size, number of epochs, and learning rate for fair performance comparison.

Device We conduct experiments on a workstation on Ubuntu 18.04 with two RTX3090 GPUs. The version of CUDA and cuDNN for GPU usage are 11.0.3 and 8, respectively.

4.4 Analysis

RQ1: Effectiveness of graph embedding We implement PBM to baselines for the code summarization task and the performance improved as shown in Table 2.

Table 2 shows the overall performance of the experimental models. PBM raises the BLEU and ROUGE performance of the Transformer, SiT and CodeBERT. The BLEU performance of the CCSD benchmark dataset of the three models increases by 7.67% on average after PBM, and the ROUGE is improved by 10.37%. The BLEU performance of the TL benchmark dataset is improved by 5.67% on average after PBM, and the ROUGE is improved by 4.57%.

Figure 3 shows the generated summaries by baseline models and the models implemented with PBM for a given source code instance. Implementation of PBM captures a word ‘gap’ of the reference that was not captured in the baseline Transformer and SiT models. The Transformer captures sequential information such as ‘calculate’ and ‘true’. But the model does not capture the objective for the ‘calculate’ word. After PBM application, the Transformer captures the object for the calculation. However, it does not capture the information of cells being rectangle. The SiT also inferences that the source code instance is a calculation. In addition, the SiT captures the information in if-statement of

[1] https://github.com/joernio/joern
[2] https://www.srcml.org/

	TL-CodeSum			CCSD		
	BLEU	ROUGE-L	SBERT	BLEU	ROUGE-L	SBERT
Seq2Seq	39.12	50.33	0.6333	20.81	23.12	0.3619
Transformer	44.34	53.74	0.6352	24.26	26.50	0.3965
CodeBERT	36.82	50.07	0.6824	22.98	29.06	**0.5256**
SiT	45.76	55.58	0.6694	25.00	26.83	0.4289
Transformer+PDG (w/o data dependency)	45.93	55.21	0.6557	25.38	28.64	0.4326
Transformer+PDG (w/o control dependency)	45.91	55.39	0.6580	26.00	29.04	0.4307
Transformer+PDG	46.07	56.68	0.6608	26.83	30.14	0.4419
CodeBERT+PDG (w/o data dependency)	40.96	52.59	0.6897	23.73	29.78	0.5235
CodeBERT+PDG (w/o control dependency)	41.17	52.91	0.6960	23.37	30.03	0.5241
CodeBERT+PDG	40.75	52.85	**0.6968**	23.41	29.45	0.5252
SiT+PDG (w/o data dependency)	45.93	56.66	0.6728	27.30	30.83	0.4452
SiT+PDG (w/o control dependency)	46.71	56.50	0.6719	27.26	27.26	0.4417
SiT+PDG	**46.86**	**56.69**	0.6752	**27.63**	**31.15**	0.4898

Table 2: Our result for Java and C dataset. The best scores for each metric are in bold.

the code that the Transformer omitted. The if statement, however, does not catch the gap of the two rectangles. On the other hand, after applying PBM, the SiT captures the semantics of calculating the gap between two rectangles.

BLEU and ROUGE are performance measures based on word overlap. These methods consider the importance of capturing the exact words that are used in the reference summaries. Even though widely used, the metrics still have drawbacks that they cannot capture the semantics of the generated sequences (Haque et al., 2022). For instance, a code instance given in Figure 3 has a reference summary of 'calculate the gap rectangle between two rectangles'. If the model generates a summary such as 'calculate the gap between rectangles (a | b (b)', it does not have any defects in semantics. However, the BLEU and ROUGE score metrics conclude that the generated summary is not perfect. On the other hand, summaries illustrated in Figure 3 show the similar or better BLEU performance to 'calculate the gap between rectangles (a | b (b)' even though the sentence does not make sense.

Therefore, moving forward from only checking whether the models generate summaries that capture the words used in the reference, we also implement an auxiliary evaluation metric SBERT score to measure the performance in semantics. SBERT is pretrained from a large corpus of natural language sentence pairs and can measure the similarity between sentences. The result of the semantic measurement is depicted in Table 2. SBERT score increases by 8.53% and 2.30% in CCSD and TL-CodeSum, respectively. When applied to CodeBERT, PBM shows weaker performance, but as the score difference is not critical and as the average performance compared with other models increases significantly, we find PBM effective.

RQ2: Comparison with graph types What difference does the proposed graph structure have compared to the popularly used AST and what is better between two graph structures?

We propose the approach that implements a graph embeddings to capture the structural information and semantics by combining both structural and sequential sequences of a source code instance. The approach shows that the graph embeddings and the structural information improves the performance of code summarization. It is, however, questionable which graph structure is suitable for generating summaries. Based on the characteristics of graph structure, we implement AST, a frequently used graph type and PDG, which we find the most effective in code summarization.

AST represents auxiliary structural information widely used in code summarization. AST is a tree representing the code structure as an abstract syntax and consists of syntax nodes for the grammar structure of program and syntax tokens for the code sequence. Each node of AST consists of a single token used in the source code. The AST for the source code instance in Figure 3 is in Appendix A. AST captures the grammar of source code which is different from that of natural language sequence. On the other hand, PDG expresses the program structure with control and data dependency information and each node of PDG consists of a segment of source code in a statement level. In that sense, the module helps the model to train the code infor-

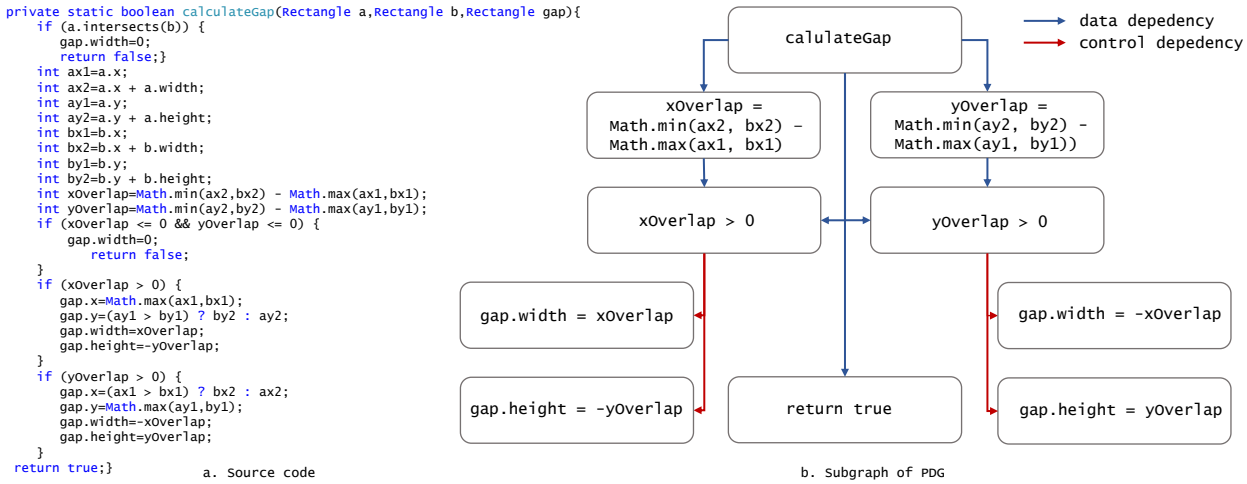

Figure 3: An illustrated example of PDG and generated summaries from the baseline models and the ones implemented with PBM.

	TL-CodeSum			CCSD		
	BLEU	ROUGE-L	SBERT	BLEU	ROUGE-L	SBERT
Transformer+AST	45.57	54.86	0.6480	26.51	29.62	0.4413
CodeBERT+AST	40.54	52.41	0.6877	23.36	29.55	0.5222

Table 3: Peformance of AST module.

mation in statement and predicate level.

We analyze the difference between the AST and PDG modules by comparing the performance of both implementations. The result is shown in Table 3 and by the performance, we confirm that the implementation of the PDG module is superior in both capturing the exact words used in the reference and the semantic similarity.

For a fair comparison, we use the same graph embedding implementation of PBM and perform experiments for baseline models, Transformer, SiT and CodeBERT. Table 3 shows the overall scores of BLEU, ROUGE and SBERT score of both modules. For each baseline model, PBM shows the average of TL-CodeSum performance of BLEU and ROUGE respectively by 0.8% and 2.1% better than that of the AST module and the average of CCSD performance of BLEU and ROUGE respectively by 0.9% and 0.8% better than that of the AST module depicted in Table 3. In addition to this, SBERT shows that PBM achieves better semantic similarity than the AST module in TL-CodeSum and shows similar performance in CCSD.

Aside from the Transformer and CodeBERT, SiT already uses structural information in the AST. Implementation of AST module results in adding overlapped the same information of AST, so we do not perform additional experiments implementing the AST module for SiT. We still can confirm that the usage of PDG information is better than AST implicitly, as the performance of SiT improves when implementing PBM.

The AST and PDG both use the graph information extracted from the same source code but the performance varies. We find that each node of AST corresponds to the token of source code sequence and the structure of a graph is too complex compared to its corresponding source code sequence to capture a valid structure information. Each node of the PDG, however, consists of statements rep-

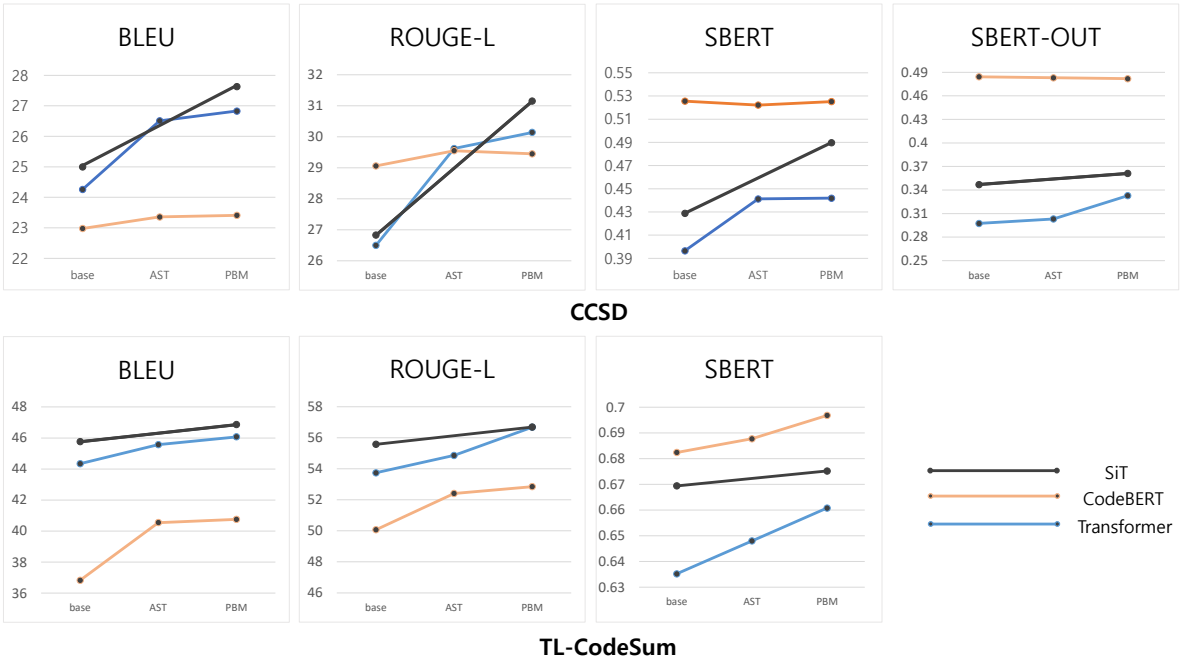

Figure 4: The performance on CCSD and TL-CodeSum. Regardless of BLEU, the higher the in-domain similarity is, the higher the out-domain similarity tends to be. Compared with AST, PBM shows a high overall performance improvement in the three category model.

resenting a relationship between statements in the form of control and data flow. This makes the PDG relatively simple and by the results shown in Table 3, we find applying PBM more effective than applying the AST module.

We evaluate the summarization ability of our module by the ablation study of edges in the PDG. Edges for PDG represent data and control dependency and we present how our module applying only one edge type performs on generating summaries in Table 2. The performance of PBM modules with one dependency edge type shows decreased performance compared to the performance of PBM module with both edge types. So we can derive that all edge types, control and data dependencies help the model to learn the structural information.

RQ3: Robustness of our framework Source code summarization is a task to generate a summary sequence for a given source code instance. Benchmark datasets are important in such research as the data to train is critical for the model. Most summaries of datasets are brought from the comments of source codes. In that sense, there is no specific guideline or tendency of summaries. Summaries can vary depending on the purpose of source codes and even the users who wrote the source codes and comments. This is why no model can be always satisfied for every source code even if the evaluation result shows good performance. On the other hand, when source codes and summaries are brought from the same repository, the data would have a similar tendency. In such cases, even the test dataset shares the same tendency and the result is not reliable as the performance score cannot verify the model to have such performance in general data.

	CCSD Out-Domain		
	BLEU	ROUGE-L	SBERT
Seq2Seq	18.70	18.91	0.3619
Transformer	20.16	17.90	0.2974
CodeBERT	20.12	24.95	**0.4843**
SiT	22.16	20.58	0.3470
Transformer+PDG	21.54	20.58	0.3328
CodeBERT+PDG	20.13	**25.08**	0.4819
SiT+PDG	**23.09**	22.93	0.3611

Table 4: Performance of CCSD out-domain dataset. The best scores for each metric are in bold.

Considering such a problem, there is a need to evaluate the model with a dataset that has a different tendency compared to the train dataset. Such aspect is called out-of-domain (OOD) and we use CCSD, the benchmark dataset for C program as

it contains both an in-domain and out-domain test dataset. The in-domain test dataset is an original dataset and the out-domain test dataset is for the OOD measurement.

Referring to the result of Table 4, performance of baseline models for the out-domain dataset is lower than that from Table 2. Such performance decrease can be up to 10% which is a critical loss. Even for this case, PBM has a robust performance improvement as Table 4 shows that it can still improve the performance of baseline models for out-domain test dataset. The performance of baseline models improves by 3.63% in average for BLEU score, by 8.97% in average for ROUGE score and by 5.17% in average for SBERT score after the implementation of PBM.

5 Conclusions

Recently, there have been several researches for improving the code summarization (Iyer et al., 2016; Feng et al., 2020; Ahmad et al., 2020). One approach is to use the structural information of source code by extracting its AST (Abstract Syntax Tree) (LeClair et al., 2020; Shi et al., 2021; Choi et al., 2021; Wu et al., 2021). While this approach works better than the one without ASTs, it turns out that a model with ASTs cannot capture the global structure owing to deep depth structure (Lin et al., 2021; Shi et al., 2021; Zhang et al., 2019).

We have studied the limit of this approach and suggested a new module PBM that utilize PDGs containing the information of control and data dependencies. We have observed that PBM improves the performance of baseline models; PBM captures the structural information of source codes in a statement level. Since BLEU and ROUGE are computed by matching tokens of the reference summary, we have considered another metric, SBERT score, for measuring the semantic difference to fully analyze the effectiveness of PBM; SBERT score evaluates the semantic similarity between sentences. The experimental results have showed that PBM achieves the performance increase in capturing the semantics of source code in the SBERT score as well. We noticed that the benchmark dataset of source codes and summaries can be vulnerable in generalization of a model. Thus, we have ran additional experiments using an out-domain test dataset of CCSD, and confirmed that our PBM is effective for the OOD case as well.

For future directions, we aim to evaluate the generalizability of the code summarization models more precisely. In addition to the out-domain dataset of CCSD, we plan to evaluate PBM with other benchmark datasets for the OOD measurements. Furthermore, we plan to design a model that has a consistently good performance on generating code summaries for different programming languages.

Acknowledgements

This research was supported by the NRF grant (NRF-2020R1A4A3079947) and the AI Graduate School Program (No. 2020-0-01361) funded by the Korea government (MSIT). The first two authors contributed equally to this work. Han is a corresponding author.

References

Wasi Uddin Ahmad, Saikat Chakraborty, Baishakhi Ray, and Kai-Wei Chang. 2020. A Transformer-based approach for source code summarization. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 4998–5007.

Uri Alon, Shaked Brody, Omer Levy, and Eran Yahav. 2019. code2seq: Generating sequences from structured representations of code. In *Proceedings of the 7th International Conference on Learning Representations*.

YunSeok Choi, JinYeong Bak, CheolWon Na, and Jee-Hyong Lee. 2021. Learning sequential and structural information for source code summarization. In *Findings of the Association for Computational Linguistics, ACL/IJCNLP 2021*, pages 2842–2851.

Zhangyin Feng, Daya Guo, Duyu Tang, Nan Duan, Xiaocheng Feng, Ming Gong, Linjun Shou, Bing Qin, Ting Liu, Daxin Jiang, and Ming Zhou. 2020. CodeBERT: A pre-trained model for programming and natural languages. In *Findings of the Association for Computational Linguistics, EMNLP 2020*, pages 1536–1547.

Jeanne Ferrante, Karl J. Ottenstein, and Joe D. Warren. 1987. The program dependence graph and its use in optimization. *ACM Transactions on Programming Languages and Systems*, 9(3):319–349.

Shuzheng Gao, Cuiyun Gao, Yulan He, Jichuan Zeng, Lun Yiu Nie, and Xin Xia. 2021. Code structure guided transformer for source code summarization. *arXiv preprint CoRR*, 2104.09340.

Daya Guo, Shuo Ren, Shuai Lu, Zhangyin Feng, Duyu Tang, Shujie Liu, Long Zhou, Nan Duan, Alexey Svyatkovskiy, Shengyu Fu, Michele Tufano, Shao Kun Deng, Colin B. Clement, Dawn Drain, Neel Sundaresan, Jian Yin, Daxin Jiang, and Ming Zhou. 2021. GraphCodeBERT: Pre-training code representations with data flow. In *Proceedings of the 9th International Conference on Learning Representations*.

Sonia Haiduc, Jairo Aponte, Laura Moreno, and Andrian Marcus. 2010. On the use of automated text summarization techniques for summarizing source code. In *Proceedings of the 17th Working Conference on Reverse Engineering*, pages 35–44.

Sakib Haque, Zachary Eberhart, Aakash Bansal, and Collin McMillan. 2022. Semantic similarity metrics for evaluating source code summarization. *arXiv preprint CoRR*, 2204.01632.

Xing Hu, Ge Li, Xin Xia, David Lo, and Zhi Jin. 2018a. Deep code comment generation. In *Proceedings of the 26th Conference on Program Comprehension*, pages 200–210.

Xing Hu, Ge Li, Xin Xia, David Lo, Shuai Lu, and Zhi Jin. 2018b. Summarizing source code with transferred API knowledge. In *Proceedings of the Twenty-Seventh International Joint Conference on Artificial Intelligence*, pages 2269–2275.

Hamel Husain, Ho-Hsiang Wu, Tiferet Gazit, Miltiadis Allamanis, and Marc Brockschmidt. 2019. CodeSearchNet challenge: Evaluating the state of semantic code search. *arXiv preprint CoRR*, 1909.09436.

Srinivasan Iyer, Ioannis Konstas, Alvin Cheung, and Luke Zettlemoyer. 2016. Summarizing source code using a neural attention model. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 2073–2083.

Alexander LeClair, Sakib Haque, Lingfei Wu, and Collin McMillan. 2020. Improved code summarization via a graph neural network. In *Proceedings of the 28th International Conference on Program Comprehension*, pages 184–195.

Alexander LeClair, Siyuan Jiang, and Collin McMillan. 2019. A neural model for generating natural language summaries of program subroutines. In *Proceedings of the 41st International Conference on Software Engineering*, pages 795–806.

Yuding Liang and Kenny Qili Zhu. 2018. Automatic generation of text descriptive comments for code blocks. In *Proceedings of the Thirty-Second AAAI Conference on Artificial Intelligence, (AAAI-18), the 30th innovative Applications of Artificial Intelligence (IAAI-18), and the 8th AAAI Symposium on Educational Advances in Artificial Intelligence*, pages 5229–5236.

Chen Lin, Zhichao Ouyang, Junqing Zhuang, Jianqiang Chen, Hui Li, and Rongxin Wu. 2021. Improving code summarization with block-wise abstract syntax tree splitting. In *Proceedings of the 29th IEEE/ACM International Conference on Program Comprehension*, pages 184–195.

Chin-Yew Lin. 2004. ROUGE: A package for automatic evaluation of summaries. In *Text Summarization Branches Out*, pages 74–81.

Shangqing Liu, Yu Chen, Xiaofei Xie, Jing Kai Siow, and Yang Liu. 2021. Retrieval-augmented generation for code summarization via hybrid GNN. In *Proceedings of the 9th International Conference on Learning Representations*.

Andrian Marcus, Andrey Sergeyev, Václav Rajlich, and Jonathan I. Maletic. 2004. An information retrieval approach to concept location in source code. In *Proceedings of the 11th Working Conference on Reverse Engineering*, pages 214–223.

Nitika Mathur, Timothy Baldwin, and Trevor Cohn. 2020. Tangled up in BLEU: reevaluating the evaluation of automatic machine translation evaluation

metrics. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 4984–4997.

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. BLEU: a method for automatic evaluation of machine translation. In *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics*, pages 311–318.

Denys Poshyvanyk and Andrian Marcus. 2007. Combining formal concept analysis with information retrieval for concept location in source code. In *Proceedings of the 15th International Conference on Program Comprehension*, pages 37–48.

Nils Reimers and Iryna Gurevych. 2019. Sentence-BERT: Sentence embeddings using siamese BERT-networks. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing*, pages 3980–3990.

Ehud Reiter. 2018. A structured review of the validity of BLEU. *Computational Linguistics*, 44(3):393–401.

Ensheng Shi, Yanlin Wang, Lun Du, Hongyu Zhang, Shi Han, Dongmei Zhang, and Hongbin Sun. 2021. CAST: enhancing code summarization with hierarchical splitting and reconstruction of abstract syntax trees. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 4053–4062.

Yusuke Shido, Yasuaki Kobayashi, Akihiro Yamamoto, Atsushi Miyamoto, and Tadayuki Matsumura. 2019. Automatic source code summarization with extended tree-LSTM. In *Proceedings of the 2019 International Joint Conference on Neural Networks*, pages 1–8.

Giriprasad Sridhara, Emily Hill, Divya Muppaneni, Lori L. Pollock, and K. Vijay-Shanker. 2010. Towards automatically generating summary comments for Java methods. In *Proceedings of the 25th IEEE/ACM International Conference on Automated Software Engineering*, pages 43–52.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Proceedings of the 31st International Conference on Neural Information Processing Systems*, pages 6000–6010.

Yao Wan, Zhou Zhao, Min Yang, Guandong Xu, Haochao Ying, Jian Wu, and Philip S. Yu. 2018. Improving automatic source code summarization via deep reinforcement learning. In *Proceedings of the 33rd ACM/IEEE International Conference on Automated Software Engineering*, pages 397–407.

Yue Wang, Weishi Wang, Shafiq R. Joty, and Steven C. H. Hoi. 2021. CodeT5: Identifier-aware unified pre-trained encoder-decoder models for code understanding and generation. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 8696–8708.

Bolin Wei, Ge Li, Xin Xia, Zhiyi Fu, and Zhi Jin. 2019. Code generation as a dual task of code summarization. In *Proceedings of the 33rd International Conference on Neural Information Processing Systems*, pages 6559–6569.

Edmund Wong, Taiyue Liu, and Lin Tan. 2015. CloCom: Mining existing source code for automatic comment generation. In *Proceedings of the 22nd International Conference on Software Analysis, Evolution, and Reengineering*, pages 380–389.

Hongqiu Wu, Hai Zhao, and Min Zhang. 2021. Code summarization with structure-induced Transformer. In *Findings of the Association for Computational Linguistics, ACL/IJCNLP 2021*, pages 1078–1090.

Fabian Yamaguchi, Nico Golde, Daniel Arp, and Konrad Rieck. 2014. Modeling and discovering vulnerabilities with code property graphs. In *Proceedings of the 2014 IEEE Symposium on Security and Privacy*, pages 590–604.

Jian Zhang, Xu Wang, Hongyu Zhang, Hailong Sun, Kaixuan Wang, and Xudong Liu. 2019. A novel neural source code representation based on abstract syntax tree. In *Proceedings of the 41st International Conference on Software Engineering*, pages 783–794.

A Abstract Syntax tree

We present also the ASTs that are omitted in the main part of the paper. Figure 5 and Figure 6 each illustrate the AST for the java code instance that was shown in Figure 1 and Figure 3 respectively.

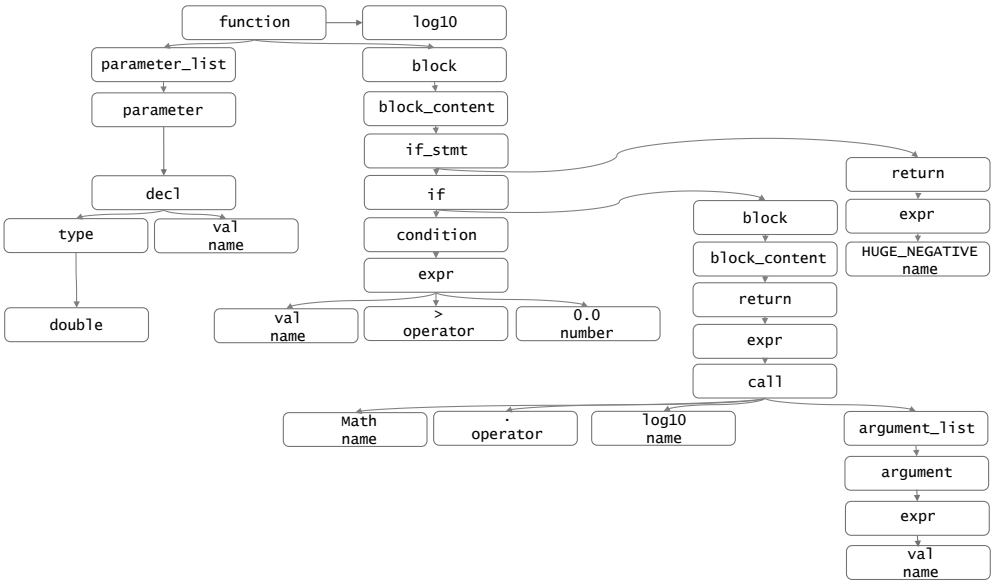

Figure 5: Graph of AST example of source code in Figure 1.

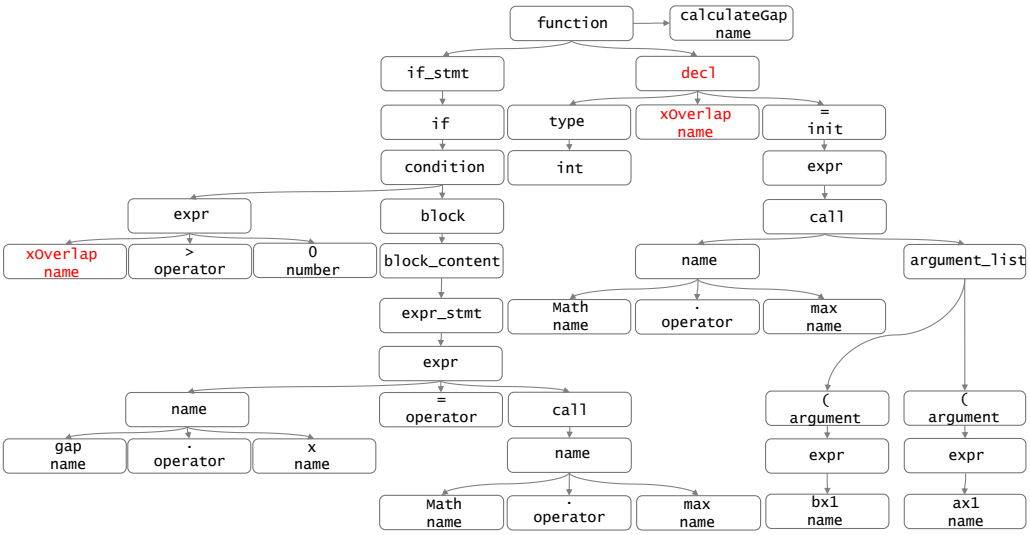

Figure 6: Subgraph of AST example of java test case.

Comparative Graph-based Summarization of Scientific Papers Guided by Comparative Citations

Jingqiang Chen[1,†,*], **Chaoxiang Cai**[1,†], **Xiaorui Jiang**[2], **Kejia Chen**[1]
[1]School of Computer Science, Nanjing University of Posts and Telecommunications, China
[2]Centre for Computational Science and Mathematical Modelling, Coventry University, UK
{cjq,1220044905,chenkj}@njupt.edu.cn,
xiaorui.jiang@coventry.ac.uk

Abstract

With the rapid growth of scientific papers, understanding the changes and trends in a research area is rather time-consuming. The first challenge is to find related and comparable articles for the research. Comparative citations compare co-cited papers in a citation sentence and can serve as good guidance for researchers to track a research area. We thus go through comparative citations to find comparable objects and build a comparative scientific summarization corpus (CSSC). And then, we propose the comparative graph-based summarization (CGSUM) method to create comparative summaries using citations as guidance. The comparative graph is constructed using sentences as nodes and three different relationships of sentences as edges. The relationship that sentences occur in the same paper is used to calculate the salience of sentences, the relationship that sentences occur in two different papers is used to calculate the difference between sentences, and the relationship that sentences are related to citations is used to calculate the commonality of sentences. Experiments show that CGSUM outperforms comparative baselines on CSSC and performs well on DUC2006 and DUC2007.

1 Introduction

Today, the transient and rapidly evolving research areas and the numerous published research articles require researchers to orient themselves and discover the changes of the research area (Marrone, 2020). In order to reduce the burden of researchers, a solution is to find and compare related articles in the research area, and automatically create comparative summaries showing commonalities and differences of the articles where differences mean changes. The first problem is how to find related and comparable articles, and the second problem is how to create comparative summaries.

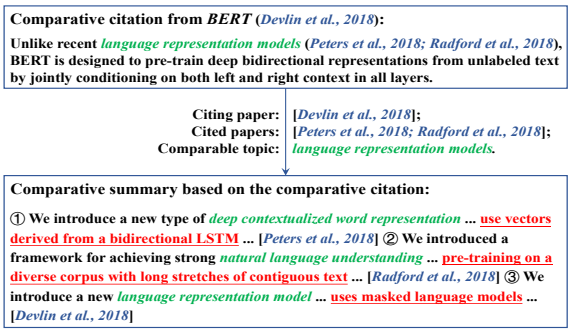

Figure 1: An example from our dataset. The comparative citation appears in the citing paper (Devlin et al., 2018) and cites two cited papers (Peters et al., 2018; Radford et al., 2018). The comparative summary is generated by summarizing the *commonality* (italicized) and difference (underlined) between three papers regarding to the comparable topic mentioned in the citation.

Fortunately, comparative citations can serve as good guidance in finding related and comparable articles and common topics. According to previous works (Teufel et al., 2006a; Hernandez-Alvarez et al., 2017), the function of comparative citations is to include shared topics between papers in the same field and reflects the comparative intent, i.e., the author intends to compare his own work with cited works. Citation function has been widely investigated. Teufel et al. (2006b) analyzed citation functions based on empirical works, and classified functions as *Contrast*, *Neutral*, *Weakness*, etc. Among all citation functions, the comparative citation is most suitable for comparative summarization as it contains most comparative information.

Given a set of comparable articles guided by comparative citations, we aim to summarize commonalities and differences between the articles and related to the comparable topics mentioned in citations. As the Figure 1 shows, the comparative citation in the upper part is captured by "unlike", where the citing and the cited papers share the comparable topic. Also, the bottom part shows a comparative summary based on mentioned comparative citation.

† The authors have contributed equally to this work.
∗ Jingqiang Chen is the corresponding author.

Proceedings of the 29th International Conference on Computational Linguistics, pages 5978–5988
October 12–17, 2022.

The summary accommodates the commonality and difference between the citing and the cited papers.

Our task is different from traditional survey generation and related work generation (Chen and Zhuge, 2016, 2019; Wang et al., 2020; Chen et al., 2021; Yuan et al., 2021) in that 1) our task utilizes the citing and the cited papers guided by comparative citations, and 2) related work and surveys focus on shared information while comparative summaries capture commonalities and differences.

We build a comparative scientific summarization corpus (CSSC) based on comparative citations. Three annotators are asked to annotate and collect comparative citations in 32 papers in the AI area using the citation function annotation scheme. We get 40 comparative citations with the corresponding citing and cited papers. For each comparative citation, annotators read through papers to generate a draft comparative summary for the comparative topic mentioned in the citation. After that, five postgraduates students specializing in works on selected 32 papers read and revise the draft comparison summary to create the ground truth.

Since our dataset is small-scaled, we propose a simple yet effective unsupervised comparative graph-based scientific summarization method (CGSUM). A comparative graph is built to represent the citation texts and papers. Each paper or citation text corresponds to a subgraph, where nodes represent sentences and weights of edges denote similarities between sentences. The salience of a sentence is computed by considering the position of the sentence within the paper. The commonality of a sentence is computed on its subgraph and the citation texts subgraph. The difference of a sentence is captured by adding negative edges between nodes from different paper subgraphs. Finally, salience, commonality and difference are linearly combined to rank and select sentences. Experiments show that CGSUM outperforms baselines on CSSC and also performs well on DUC2006 and DUC2007.

Our contributions are summarized as follows:

- We propose the task of comparative citation-guided summarization of scientific papers.
- We construct the comparative summarization dataset CSSC for scientific papers.
- We propose the comparative graph-based summarization method that considers three relationships between sentences. Experiments show the efficacy of the proposed model.

2 Related Work

Citations throughout scientific papers help understand the frontiers and trends in diverse research fields. Teufel et al. (2006b) analyzed citation functions based on empirical works, which is similar to (Su et al., 2019). Whereas Dong and Schäfer (2011); Abu-Jbara et al. (2013); Hernandez-Alvarez et al. (2017); Su et al. (2019) focused on the dimensions of organic and perfunctory as well as intentions and sentiments respectively.

Generic scientific summarization uses extractive(Yang et al., 2016; An et al., 2021; Dong et al., 2021), abstractive(See et al., 2017; Cachola et al., 2020; Dangovski et al., 2021) and other (Teufel and Moens, 2002; Cohan et al., 2018; Sharma et al., 2019) methods to summarize a document. Citation generation has also been studied (Xing et al., 2020; Ge et al., 2021). Early studies were based on keywords (Hoang and Kan, 2010; Chen and Zhuge, 2016). Xing et al. (2020) considered the abstract of cited papers to generate citations. Citation generation concerns semantics, while comparative summarization concerns citation function. The citation text is often too short to describe in detail.

Related work generation and survey generation generates from multiple documents. He et al. (2016) captured hot topics in fields. Chen and Zhuge (2019); AbuRa'ed et al. (2020); AbuRa'ed and Saggion (2021) took citations into account to mine information. What's more, Wang et al. (2020); Yuan et al. (2021) generated reviews that cover more aspects. Chen et al. (2021) took abstractive method. Related work generation places emphasis on shared content in cited papers and summarizes the common information. However, it is different from our task of comparative summarization, which is guided by comparative citations and summarizes commonalities and differences.

3 Task Definition

Given the comparative citation (*Cit*), the citing paper (*CP*) where the *Cit* appears, and a set of reference papers (*RPs*) that the *Cit* cites, the task aims to create the comparative summary containing commonalities and differences between *CP* and *RPs* with regard to the comparable topic mentioned in the *Cit*. Taking the case in Figure 1, *Cit* refers to the comparative citation, *CP* refers to the (Devlin et al., 2018) and *RPs* refers to (Peters et al., 2018; Radford et al., 2018). The comparable topic mentioned in the *Cit* is *language representation models*.

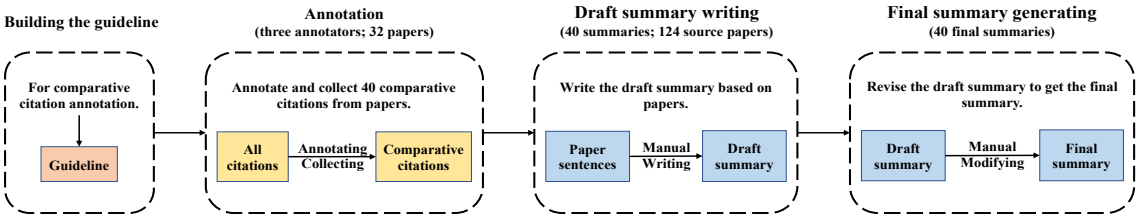

Figure 2: Overview of the dataset construction process.

Since there are no existing datasets and methods for the task, we build a comparative citation-guided dataset and propose a comparative graph-based method. To create the dataset, we annotate the function of citations of 32 papers in the AI area, and manually create comparative summaries for 40 annotated comparative citations. Then we create the comparative summary, the proposed summarization method leverages different relationships between sentences to construct a comparative graph and extract sentences from papers.

4 Dataset Construction

This section mainly contains citation annotating as well as data processing. Figure 2 depicts an overview of the data construction process.

4.1 Comparative Citation Annotating

Comparative citations provide comparable information such as related and comparable articles and comparable topics for comparative summary generation. However, previous studies (Teufel et al., 2006a; Dong and Schäfer, 2011; Jha et al., 2015; Jurgens et al., 2018; Su et al., 2019) showed that the proportion of comparative citations in scientific articles are minimal (See Appendix A.1 for details). In their annotation schemes, the comparisons are scattered in different categories and are not easily distinguished. Therefore, we propose our own annotation guideline (See Appendix A.2 for details) that is sensitive to finding comparative citations. Using the guideline, three annotators are asked to annotate comparative citations in 32 papers selected from the AI area. Each paper contains 21 to 30 citations. Each citation consists of one to five sentences and cites two to six cited papers. Finally, we obtain 40 comparative citations for building the comparative citation-guided dataset.

4.2 Data Processing and Summaries Writing

With the comparative citations we get, we collect the citing and the cited papers associated with each citation from the web. The abstract, introduction, conclusion, etc. sections from papers are used for summarizing as these sections contain dense and essential information about papers. Firstly, annotators are asked to read through papers and manually write a draft comparative summary for the same topic mentioned in each comparative citation based on the crucial sentences in the papers. Secondly, five other graduate students who are professionals in the works of selected 32 papers read and modify the draft comparative summaries until they all agree that complete information such as the commonalities and differences and the salience within papers are included. The generated comparative summaries serve as the ground truth. We end up with a dataset that includes comparative citations, the citing paper, the cited papers, and the reference summaries for each citation.

5 Comparative Graph-based Scientific Summarization

We propose the comparative graph-based summarization method (CGSUM for short). The overview of the method is shown in Figure 3. The core idea of the method is to select sentences by calculating the salience of the sentences and estimating the degrees to which the sentences reflect the commonalities and differences between papers.

5.1 Construction of Comparative Graph

In the same document, a sentence receives positive influence from sentences that correlate to it, whereas in the different documents, a sentence receives negative influence from sentences that correlate to it (Li et al., 2008). And citations contain common topics between papers. For our task, there are three different relationships between two sentences: two sentences occurring in a same paper (Intra-paper Relationship); two sentences occurring in two different papers (Inter-papers Relationship); and the sentence related to citation texts (Citation-text Relationship). All three relationships are used.

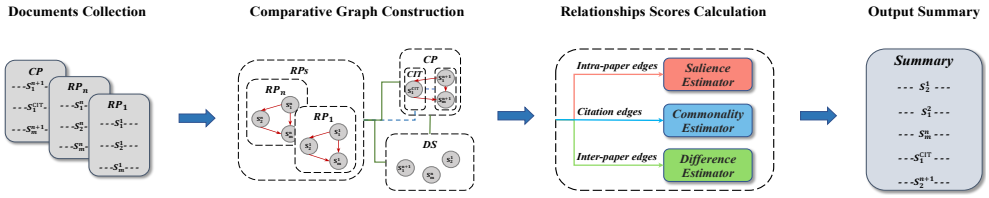

Figure 3: Overview of the comparative graph-based scientific summarization method CGSUM.

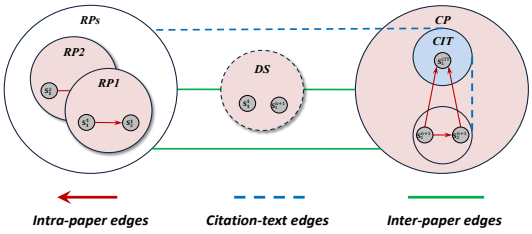

Figure 4: Example of a comparative graph that considers three different relationships between sentences. In this example, the graph contains five subgraphs {*CIT*, *CP*, *RP1*, *RP2*, *DS*}, where *CIT* denotes a comparative citation, *CP* denotes a citing paper, *RP1* and *RP2* denote two reference papers, and *DS* denotes the dynamic summary which consists of summary sentences generated so far and is initially empty and updated iteratively.

The above three relationships are used to construct the comparative graph for papers and citations. As is shown in Figure 4, the graph consists of the citing paper subgraph (the right part of the figure), the reference paper subgraphs (the left part of the figure) and the citation subgraph (the blue circle in the right part). To avoid redundancy, we introduce the dynamic summary subgraph (the circle in the middle part) which consists of summary sentences generated so far, and compare candidate sentences with the dynamic summary.

Intra-paper edges (directed solid edges) correspond to Intra-paper relationships. These edges are directed because sentences in a paper are sequentially ordered. If one sentence occurs before another sentence in a paper, the direction of the Intra-paper edge is from the former to the latter. The weights of these edges are set as similarities between sentences. These edges can be used to compute the salience of sentences.

Inter-paper edges (undirected solid edges) correspond to Inter-paper relationships. The weights of these edges are set as negative similarities between sentences. These edges can be used to compute differences between sentences.

Citation-text edges (undirected dotted edges) correspond to Citation-text relationships. Weights of these edges are similarities between the sentences in the citation and papers, reflecting the common topic in papers and citation.

5.2 Sentence Ranking and Selecting

The ranking scores of sentences are supposed to reflect the salience of sentences within papers, the degree to which sentences capture commonalities between the papers and citation, and the degree to which sentences capture differences between papers and between papers and the dynamic summary. As is shown in Task Definition, each paper is extracted to produce the comparative summary, where the sentences are salient within the paper that they belong. Sentences extracted from each paper are related to citation and reflect the commonality between papers. Besides, the extracted sentences are different from those extracted from other papers, which captures the difference between papers. Three estimators that calculate the scores of salience, commonalities and differences of sentences, are proposed on the comparative graph.

Salience estimator calculates the salience score of a sentence node by summing up the weights of its outcoming Intra-paper edges and subtracting the weights of its incoming Intra-paper edges. The contributions of any two sentences to their respective centrality are influenced by their relative positions in a document. The sentences before are central, while the sentences after supplementing them. Specifically, a sentence is salient if it has many similar sentences after the sentence. Otherwise, a sentence is redundant if it has many similar sentences before it. In the constructed graph, Intra-paper edges are directed from the sentences before (OUT) to the sentences after (IN). Therefore, for a sentence s_p, the outcoming Intra-paper edges contribute positively to its salience while the incoming Intra-paper edges contribute negatively to its salience. Equation 1 is for calculation of the salience score, where $\alpha, \beta \in [0,1]$, and $\alpha+\beta=1$.

$$SAL(s_p) = \alpha \sum_{s_o \in OUT} sim_{p,o} - \beta \sum_{s_i \in IN} sim_{p,i}, \quad (1)$$

Commonality estimator calculates the commonality score of a sentence node by summing up the weights of its Citation-text edges. The citation bridges the citing paper and the reference papers and contains the commonality of the topic shared by the papers. It is reasonable to believe that the more similar the sentences with the citation, the more common information the sentences contain. Equation 2 is for calculations of commonality scores, where the sentences s_p are in papers and the sentences s_{cit} are in citation.

$$COM(s_p) = \sum_{s_{cit} \in CIT} sim_{p,cit}, \quad (2)$$

Difference estimator calculates the difference score of a sentence node by summing up the weights of Inter-paper edges of the sentences. Sentences from different papers introduce the common topic from different aspects. Avoiding redundancy brings more differences. Our extractive method is iterative, which generates summaries by selecting sentences from papers in order of publication time. To avoid redundancy, we add an extra paper named dynamic summary. Dynamic summary is a dynamic paper consisting of the sentences of summary generated so far. It is initially empty and ends up being a comparative summary. The negative influence between the dynamic summary and papers waiting to be summarized is used to calculate the difference score of a sentence. Equation 3 is for calculations of difference scores, where s_{cp} is a sentence in the citing paper, s_{rp} is a sentence in the reference papers and s_{ds} is a sentence in the dynamic summary. Weights of Inter-paper edges are set as negative similarity values, and the difference scores are also of negative values. The higher difference score of the sentence is, the more different the sentence is from sentences in other papers.

$$DIF(s_{cp}) = -sim_{cp,ds} - \sum_{s_{rp} \in RPs} sim_{cp,rp}, \quad (3)$$

The salience score of a sentence reflects the salience of the sentence within the paper, the commonality score reflects the commonality of topic information contained in the sentence, and the difference score reflects different aspects of knowledge of the topic discussed in the citation. Equation 4 linearly interpolates the three scores as the final ranking score of the sentence, where $\lambda_1, \lambda_2, \lambda_3 \in [0,1]$, and $\lambda_1 + \lambda_2 + \lambda_3 = 1$.

$$Score(s) = \lambda_1 SAL(s) + \lambda_2 COM(s) + \lambda_3 DIF(s), \quad (4)$$

Algorithm 1 CGSUM

Require: *RPs*, *CIT*, *CP*
Ensure: Comparative Summary
  for RP in RPs **do**
    for sent in RP **do**
      score = λ_1SAL(sent)+λ_2COM(sent)+λ_3DIF(sent)
      Add (sent, score) into SenScore
    end for
    Rank and select sent into Comparative Summary
    Clear SenScore
  end for
  Select CIT into Summary
  for sent in CP **do**
    score = λ_1SAL(sent)+λ_2COM(sent)+λ_3DIF(sent)
    Add (sent, score) into SenScore
  end for
  Rank and select sent into Summary
  Return Comparative Summary

With the sentences of each paper ranked by the final ranking scores, we select sentences to generate summaries. As is shown in the Pseudo code. To ensure that every papers can be summarized, we select from each and use the citation to bridge the reference papers and citing paper.

6 Experiments

6.1 Datasets

Dataset	CSSC	DUC2006	DUC2007
Domain	Sci	News	News
Query	Long	Long	Short
Clusters	40	50	45
Documents	3-5	25	25

Table 1: Statistics of the three datasets.

Experiments on comparative scientific summarization are carried out on CSSC. Additional experiments are carried out on DUC2006 and DUC2007 to show the generalization of the proposed method to multi-document summarization. As is shown in Table 1, CSSC contains citations over 40 clusters with three to five scientific papers each. DUC2006 and DUC2007 contain long queries over 50 clusters and short queries over 45 clusters, respectively.

6.2 Comparing methods

There are four kinds of comparing methods:

The first four methods: (1) ORACLE returns a extractive sentences subset with the highest ROUGE scores. (2) SIM2GOLD and (3) SIM2CIT, respectively, select three sentences which are most similar to the reference summary and the citation texts from each paper. (4) LEAD returns lead sentences (up to three) of each paper.

Models	CSSC				CSSC (concatenated)			
	R-1	**R-2**	**R-L**	**R-SU4**	**R-1**	**R-2**	**R-L**	**R-SU4**
ORACLE (extractive)	57.4	39.1	55.3	40.7	57.4	39.1	55.3	40.7
SIM2GOLD	54.1	36.9	52.6	35.9	53.4	36.5	50.1	33.8
SIM2CIT	43.9	22.3	41.1	24.2	38.0	19.5	40.4	23.0
LEAD	43.5	21.7	40.7	23.6	33.3	14.4	30.9	16.3
Heuristic								
RANDOM (motivated by (Xing et al., 2020))	22.7	4.1	15.6	4.0	23.3	4.8	15.8	3.9
COPY-CIT (motivated by (Xing et al., 2020))	33.6	18.0	30.1	18.3	33.6	18.0	30.1	18.3
Multi-document								
Summpip (Zhao et al., 2020) (reproduce)	43.4	18.6	38.6	21.3	36.4	15.6	31.6	18.4
QUERYSUM (Xu and Lapata, 2020) (reproduce)	42.2	18.5	38.4	21.7	—	—	—	—
TIF-IDF-Sum (Lamsiyah et al., 2021)	42.8	19.2	38.1	22.3	—	—	—	—
Graph-based								
TextRank (Mihalcea and Tarau, 2004) (reproduce)	41.2	15.0	37.4	16.6	32.4	12.3	31.2	13.2
LexRank (Erkan and Radev, 2004) (reproduce)	42.1	18.0	37.5	18.9	36.3	11.5	31.8	15.0
PACSUM (Zheng and Lapata, 2019) (reproduce)	42.3	19.1	37.5	21.6	37.9	14.2	33.3	17.3
HIPORANK (Dong et al., 2021) (reproduce)	42.0	17.9	37.5	15.3	—	—	—	—
Ours								
CGSUM-TF-IDF	47.2	25.5	43.5	27.1	41.6	17.6	36.4	19.9
CGSUM-BERT	**48.6**	**28.5**	**45.3**	**28.7**	**42.1**	**20.4**	**40.8**	**21.3**

Table 2: Automatic evaluation results on CSSC and CSSC(concatenated). **Bold** indicates the best result.

Heuristic: (1) RANDOM randomly selects three sentences of each paper. (2) COPY-CIT treats the citation texts as the output.

Multi-document: (1) Summpip (Zhao et al., 2020) is an unsupervised graph-based method for multi-document summarization. (2) QUERYSUM (Xu and Lapata, 2020) is a query-focused framework for estimating relevant text segments, and (3) TF-IDF-Sum (Lamsiyah et al., 2021), which estimates relevant sentences for query-focused multi-document summarization.

Graph-based: (1) TextRank (Mihalcea and Tarau, 2004) and (2) LexRank (Erkan and Radev, 2004) are unsupervised methods based on Markov random walks. (3) PACSUM (Zheng and Lapata, 2019) and (4) HIPORANK (Dong et al., 2021) are directional graph-based methods considering the relative position and the hierarchy, respectively.

6.3 Results on CSSC

6.3.1 Automatic evaluations

We evaluate our models with ROUGE (Lin, 2004), reporting the F1 scores for ROUGE-1, ROUGE-2, ROUGE-L, and ROUGE-SU4. The test results are shown in Table 2, where CSSC (concatenated) concatenates the citing and the cited papers into one for each. The test results on CSSC (concatenated) are used to verify the hypothesis that it is necessary to select from each document.

Our models outperform comparative baselines on CSSC. The results of COPY-CIT show the particularity of the task in this paper and deduct that citation generation does not match our task. Our proposed models, CGSUM-TF-IDF and CGSUM-BERT, obviously outperform the baseline models. This result proves the effectiveness of our comparative graph-based summarizer, which considers different relationships between sentences. It requires the model to capture the critical content of the cited paper and to capture the attitude of the citing paper to the cited paper. The model not only needs to generate fluent and informative text but also needs to ensure contextual coherence. The results on CSSC are all higher than the results on CSSC (concatenated), which means it is necessary to select from each document because it ensures that the generated summary can reflect the salience, commonality, and difference of each document, which avoids information miss.

6.3.2 Human evaluations

We adopt the QA paradigm and the similarity between the gold summaries and system summaries to evaluate summaries quality. For the QA paradigm, reviewers create questions (e.g. salient content of each paper, common topic in the citation, and different aspects concerning the common topics) based on gold summaries. They examine whether system summaries can answer these questions. The more detailed questions the system summaries can answer, the better they are. For the similarity, reviewers assess the degree to which system summaries retain the salience of papers and the commonality and difference between papers. Specifically, the

Models	CSSC				
	SAL	COM	DIF	COH	ALL
ORACLE	4.50	4.78	4.36	4.06	4.44
SIM2GOLD	4.38	4.36	4.20	3.96	4.24
LEAD	3.56	3.16	3.68	3.24	3.42
Summpip (2020)	3.68	3.80	3.58	3.42	3.68
QUERYSUM (2020)	3.58	4.06	3.36	3.24	3.56
PACSUM(2019)	3.62	3.82	3.50	3.38	3.58
HIPORANK(2021)	3.72	3.78	3.52	3.28	3.58
CGSUM-BERT	**4.12**	**4.02**	**3.98**	**3.86**	**4.00**

Table 3: Human evaluation results on CSSC. **SAL**ience, **COM**monality, **DIF**ference, **COH**erence, **ALL** is the average across all scores. **Bold** indicates the best result.

salience score is assessed by comparing the similarities between system summaries and abstracts of each paper. The commonality is assessed by comparing the similarities between system summaries and citations. The difference is assessed by comparing the dissimilarity between system summaries and abstracts of each paper on the common topic. The coherence of system summaries is also taken by assessing their readability. After two stages of review, each reviewer gives each human evaluation metric a score of 0.0-5.0 based on the questions they created and the similarity they assessed. These scores will be averaged to obtain a final score for the system summary.

As is shown in Table 3, our models outperform the baseline models. The COH score and the ALL score of our models are especially higher than that of the baselines. This result further demonstrates the efficacy of our proposed models. Using the salience, commonality and difference estimators, CGSUM captures salience within papers and commonalities and differences between papers. Summaries created by CGSUM are also more coherent by using citations to join the contents of papers.

6.4 Results on DUC2006 and DUC2007

The results on DUC2006 and DUC2007 are summarized in Table 4. GRSum (Wan, 2008) integrated query-relevance into a Graph Ranking algorithm. C-Attention (Li et al., 2017) compresses multi-document summarization. The results show that our models perform well on the DUC2006 and DUC2007 datasets. Compared to the results ORACLE gets, the results our models get mean that our extractive models are almost close to the mostly perfect extractive summaries at the sentence level. The exciting conclusion shows that our comparative graph-based models are promising to be applied for the multi-document summarization task.

Models	DUC2006			DUC2007		
	R-1	R-2	R-SU4	R-1	R-2	R-SU4
ORACLE	40.6	9.1	14.8	41.8	10.4	16.0
LEAD	32.1	5.3	10.4	33.4	6.5	11.3
Graph-based						
LexRank	34.2	6.4	11.4	35.8	7.7	12.7
GRSUM	38.4	7.0	12.8	42.0	10.3	15.6
TF-IDF-Sum	39.0	7.9	13.8	40.1	10.1	15.2
Compress-based						
C-Attention	39.3	8.7	14.1	42.3	10.7	16.1
QUERYSUM	**41.1**	**9.6**	**15.1**	**42.9**	**11.6**	**16.7**
Ours						
CGSUM-TF-IDF	39.8	8.2	14.0	41.0	9.8	15.5
CGSUM-BERT	40.1	8.4	14.3	41.2	10.3	15.7

Table 4: Automatic evaluation results on DUC2006 and DUC2007. **Bold** indicates the best result overall. Underline denotes the best sentence-extractive results.

Models	CSSC			
	R-1	R-2	R-L	R-SU4
CGSUM-TF-IDF	**47.2**	**25.5**	**43.5**	**27.1**
w/o Salience	↓43.7	↓19.8	↓39.5	↓22.4
w/o Commonality	↓41.5	↓17.8	↓37.8	↓20.7
w/o Difference	↓45.3	↓23.4	↓41.3	↓25.3
CGSUM-BERT	**48.6**	**28.5**	**45.3**	**28.7**
w/o Salience	↓42.2	↓18.4	↓38.6	↓20.9
w/o Commonality	↓44.8	↓22.3	↓41.5	↓24.3
w/o Difference	↓43.7	↓19.8	↓40.0	↓22.3

Table 5: Ablation results on CSSC. ↓ denotes decrease.

6.5 Ablation Studies

Ablation studies in Table 5 are carried out to show effects of two representations and three estimators.

- **Representations** include TF-IDF and BERT. BERT performs better than TF-IDF.
- **w/o Salience** represents CGSUM without the salience estimator, and it performs worse than CGSUM with the salience estimator, indicating that salience estimator is effective in capturing salient information within papers.
- **w/o Commonality** represents CGSUM without the commonality estimator, and it performs worse than CGSUM with the commonality estimator, indicating that commonality estimator is effective because the estimator can find the commonality between papers.
- **w/o Difference** represents CGSUM without the difference estimator, and it performs not as well as CGSUM with the difference estimator, implying that difference estimator is effective.

Removing each estimator leads to a drop of the performance of CGSUM. Meaning estimators capture different information to produce summaries.

Comparative Citation from (*Devlin et al., 2018*):

Unlike recent ***language representation models*** (*Peters et al., 2018; Radford et al., 2018*), BERT is designed to pre-train deep bidirectional representations from unlabeled text by jointly conditioning on both left and right context in all layers.

GOLD:

All three works belong to the field of Natural Language Processing and are all about language representation models. [*Peters et al., 2018*] introduced a type of deep contextualized word representation that models both complex characteristics of word use, and how these uses vary across linguistic contexts. The vectors are derived from a bidirectional LSTM that is trained with a coupled language model objective on a large text corpus. [*Radford et al., 2018*] explored a semi-supervised approach for language understanding tasks using a combination of unsupervised pre-training and supervised fine-tuning. The approach introduces a framework for achieving strong natural language understanding with a single task-agnostic model through generative pre-training and discriminative fine-tuning. [*Devlin et al., 2018*] introduced a language representation model called BERT, which stands for Bidirectional Encoder Representations from Transformers and is designed to pre-train deep bidirectional representations from the unlabeled text by jointly conditioning on both left and right context in all layers.

CGSUM:

[*Peters et al., 2018*]: We introduce a new type of ***deep contextualized word representation*** that models both (1) complex characteristics of word use, and (2) how these uses vary across linguistic contexts. We use vectors derived from a bidirectional LSTM that is trained with a coupled language model objective on a large text corpus. Unlike previous approaches for learning ***contextualized word vectors*** (Peters et al., 2017; McCann et al., 2017), ELMo representations are deep, in the sense that they are a function of all of the internal layers of the biLM. [*Radford et al., 2018*]: We demonstrate that large gains on these tasks can be realized by generative pre-training of a ***language model*** on a diverse corpus of unlabeled text, followed by discriminative fine-tuning on each specific task. We introduced a framework for achieving strong ***natural language understanding*** with a single task-agnostic model through generative pre-training and discriminative fine-tuning. By pre-training on a diverse corpus with long stretches of contiguous text our model acquires significant world knowledge and ability to process long-range dependencies which are then successfully transferred to solving discriminative tasks such as question answering. [*Devlin et al., 2018*]: Unlike recent ***language representation models*** (Peters et al., 2018a; Radford et al., 2018), BERT is designed to pretrain deep bidirectional representations from unlabeled text by jointly conditioning on both left and right context in all layers. The major limitation is that standard ***language models*** are unidirectional, and this limits the choice of architectures that can be used during pre-training. We introduce a new ***language representation*** model called BERT, which uses masked language models to enable pre-trained deep bidirectional representations.

Comparative Citation from (*Tran et al., 2020*):

To ***encode*** the article text we use RoBERTa. Unlike GloVe (*Pennington et al., 2014*) and word2vec (*Mikolov et al., 2013*) ***embeddings***, RoBERTa uses BPE which can encode any word made from Unicode characters.

GOLD:

All three studies are about embeddings for text. By subsampling of the frequent words. [*Mikolov et al., 2013*] obtained significant speedup and also learn more regular word representations. The training objective of the Skip-gram model is to find word representations that are useful for predicting the surrounding words in a sentence or a document. [*Pennington et al., 2014*] used their insights to construct a new model for word representation which they called GloVe. They constructed a model that utilizes the benefit of count data while simultaneously capturing the meaningful linear substructures prevalent in recent log-bilinear prediction-based methods like word2vec. To encode the article text, [*Tran et al., 2020*] used RoBERTa, a recent improvement over the popular BERT model. RoBERTa is a pre-trained language representation model providing contextual embeddings for text. It consists of 24 layers of bidirectional transformer blocks.

CGSUM:

[*Mikolov et al., 2013*]: The recently introduced continuous Skip-gram model is an efficient method for learning high-quality distributed ***vector representations*** that capture a large number of ***precise syntactic and semantic word relationships***. By subsampling of the frequent words we obtain significant speedup and also learn more regular ***word representations***. We also describe a simple alternative to the hierarchical softmax called negative sampling, and show how to train distributed ***representations of words and phrases*** with the Skip-gram model and demonstrate that these representations exhibit linear structure that makes precise analogical reasoning possible. [*Pennington et al., 2014*]: Our model efficiently leverages statistical information by training only on the nonzero elements in a word-word co-occurrence matrix, rather than on the entire sparse matrix or on individual context windows in a large corpus. The model produces a ***word vector space*** with meaningful sub-structure utilizes the main benefit of count data while simultaneously capturing the meaningful linear substructures prevalent in recent log-bilinear prediction-based methods like word2vec. The result, GloVe, is a new global log-bilinear regression model for the unsupervised learning of ***word representations*** that outperforms other models on word analogy, word similarity, and named entity recognition tasks. [*Tran et al., 2020*]: Unlike GloVe (Pennington et al., 2014) and word2vec (Mikolov et al., 2013) ***embeddings***, RoBERTa uses BPE which can ***encode any word*** made from Unicode characters. In BPE, each word is first broken down into a sequence of bytes. Common byte sequences are then merged using a greedy algorithm.

Figure 5: Case studies on examples taken from the CSSC dataset.

6.6 Case Study

Case studies in Figure 5 are carried out on examples taken from CSSC. All comparable works related to the corresponding comparative citation are marked as [*author, year*]. The commonality is marked in green and italicized while difference is marked in green and underlined. As the Figure 5 shows, summaries created by CGSUM cover detailed salience within papers and commonalities and differences between papers, and are also quite coherent.

7 Conclusion

This paper proposes the novel task of comparative citation-guided summarization of scientific papers, which aims to summarize commonalities and differences between the articles and related to the comparable topic mentioned in comparative citations. The CSSC dataset for the task is constructed, which contains 40 groups of comparable scientific papers and corresponding reference summaries by annotating and collecting comparative citations. The unsupervised comparative graph-based summarization CGSUM method is proposed to generate comparative summaries. It utilizes three different relationships of sentences to build a comparative graph and calculates the scores of salience, commonality and difference without large-scaled data. Experiments on CSSC show that CGSUM outperforms baselines. Experiments on DUC2006 and DUC2007 demonstrate that CGSUM can be generalized to multi-document summarization tasks. In the future, we would like to study more types of relationships between documents and research the comparative scientific summarization cross the fields.

Acknowledgements

This research was sponsored by the National Natural Science Foundation of China (No.61806101). We also thank Wenwen Fan and Tong Liu for annotating the citations, and Feng Xie, Yirui Huang, Lingyun Jin, Xianzhe Xu and Qingsen Bao for performing the ground truth and human evaluation.

References

Amjad Abu-Jbara, Jefferson Ezra, and Dragomir Radev. 2013. Purpose and polarity of citation: Towards nlp-based bibliometrics. In *Proc. of NAACL*.

Ahmed Ghassan Tawfiq AbuRa'ed and Horacio Saggion. 2021. A select and rewrite approach to the generation of related work reports. In *Proc. of CEUR Workshop*.

Ahmed AbuRa'ed, Horacio Saggion, Alexander Shvets, and Àlex Bravo. 2020. Automatic related work section generation: experiments in scientific document abstracting. *Scientometrics*.

Chenxin An, Ming Zhong, Yiran Chen, Danqing Wang, Xipeng Qiu, and Xuanjing Huang. 2021. Enhancing scientific papers summarization with citation graph. In *Proc. of AAAI*.

Isabel Cachola, Kyle Lo, Arman Cohan, and Daniel S Weld. 2020. Tldr: Extreme summarization of scientific documents. In *Proc. of EMNLP Findings*.

Jingqiang Chen and Hai Zhuge. 2016. Summarization of related work through citations. In *2016 12th International Conference on Semantics, Knowledge and Grids (SKG)*.

Jingqiang Chen and Hai Zhuge. 2019. Automatic generation of related work through summarizing citations. *Concurrency and Computation: Practice and Experience*.

Xiuying Chen, Hind Alamro, Mingzhe Li, Shen Gao, Xiangliang Zhang, Dongyan Zhao, and Rui Yan. 2021. Capturing relations between scientific papers: An abstractive model for related work section generation. In *Proc. of ACL*.

Arman Cohan, Franck Dernoncourt, Doo Soon Kim, Trung Bui, Seokhwan Kim, Walter Chang, and Nazli Goharian. 2018. A discourse-aware attention model for abstractive summarization of long documents. In *Proc. of NAACL*.

Rumen Dangovski, Michelle Shen, Dawson Byrd, Li Jing, Desislava Tsvetkova, Preslav Nakova, and Marin Soljacic. 2021. We can explain your research in layman's terms: Towards automating science journalism at scale. In *Proc. of AAAI*.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*.

Cailing Dong and Ulrich Schäfer. 2011. Ensemble-style self-training on citation classification. In *Proc. of IJCNLP*.

Yue Dong, Andrei Mircea, and Jackie Chi Kit Cheung. 2021. Discourse-aware unsupervised summarization for long scientific documents. In *Proc. of EACL*.

Günes Erkan and Dragomir R Radev. 2004. Lexrank: Graph-based lexical centrality as salience in text summarization. *Journal of artificial intelligence research*.

Yubin Ge, Ly Dinh, Xiaofeng Liu, Jinsong Su, Ziyao Lu, Ante Wang, and Jana Diesner. 2021. Baco: A background knowledge-and content-based framework for citing sentence generation. In *Proc. of ACL*.

Lei He, Wei Li, and Hai Zhuge. 2016. Exploring differential topic models for comparative summarization of scientific papers. In *Proc. of COLING*.

Myriam Hernandez-Alvarez, José M. Gomez Soriano, and Patricio Martínez-Barco. 2017. Citation function, polarity and influence classification. *Natural Language Engineering*.

Cong Duy Vu Hoang and Min-Yen Kan. 2010. Towards automated related work summarization. In *Proc. of COLING*.

Rahul Jha, Reed Coke, and Dragomir Radev. 2015. Surveyor: A system for generating coherent survey articles for scientific topics. In *Proc. of AAAI*.

Rahul Jha, Amjad-Abu Jbara, Vahed Qazvinian, and Dragomir R. Radev. 2017. Nlp-driven citation analysis for scientometrics. *Natural Language Engineering*.

David Jurgens, Srijan Kumar, Raine Hoover, Dan McFarland, and Dan Jurafsky. 2018. Measuring the evolution of a scientific field through citation frames. *Transactions of the Association for Computational Linguistics*.

Salima Lamsiyah, Abdelkader El Mahdaouy, Said Ouatik El Alaoui, and Bernard Espinasse. 2021. Unsupervised query-focused multi-document summarization based on transfer learning from sentence embedding models, bm25 model, and maximal marginal relevance criterion. *Journal of Ambient Intelligence and Humanized Computing*.

Piji Li, Wai Lam, Lidong Bing, Weiwei Guo, and Hang Li. 2017. Cascaded attention based unsupervised information distillation for compressive summarization. In *Proc. of EMNLP*.

Wenjie Li, Furu Wei, Qin Lu, and Yanxiang He. 2008. Pnr2: Ranking sentences with positive and negative reinforcement for query-oriented update summarization. In *Proc. of COLING*.

Chin-Yew Lin. 2004. Rouge: A package for automatic evaluation of summaries. In *Text summarization branches out*.

Mauricio Marrone. 2020. Application of entity linking to identify research fronts and trends. *Scientometrics*.

Rada Mihalcea and Paul Tarau. 2004. Textrank: Bringing order into text. In *Proc. of EMNLP*.

Matthew E. Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer. 2018. Deep contextualized word representations. In *Proc. of ACL*.

Alec Radford, Karthik Narasimhan, Tim Salimans, and Ilya Sutskever. 2018. Improving language understanding by generative pre-training.

Abigail See, Peter J Liu, and Christopher D Manning. 2017. Get to the point: Summarization with pointer-generator networks. In *Proc. of ACL*.

Eva Sharma, Chen Li, and Lu Wang. 2019. Bigpatent: A large-scale dataset for abstractive and coherent summarization. In *Proc. of ACL*.

Xuan Su, Animesh Prasad, Min-Yen Kan, and Kazunari Sugiyama. 2019. Neural multi-task learning for citation function and provenance. In *Proc. of JCDL*.

Simone Teufel and Marc Moens. 2002. Summarizing scientific articles: experiments with relevance and rhetorical status. *Computational linguistics*.

Simone Teufel, Advaith Siddharthan, and Dan Tidhar. 2006a. An annotation scheme for citation function. In *Proc. of SIGDIAL*.

Simone Teufel, Advaith Siddharthan, and Dan Tidhar. 2006b. Automatic classification of citation function. In *Proc. of EMNLP*.

Xiaojun Wan. 2008. Using only cross-document relationships for both generic and topic-focused multi-document summarizations. *Information Retrieval*.

Qingyun Wang, Qi Zeng, Lifu Huang, Kevin Knight, Heng Ji, and Nazneen Fatema Rajani. 2020. Reviewrobot: Explainable paper review generation based on knowledge synthesis. *arXiv preprint arXiv:2010.06119*.

Xinyu Xing, Xiaosheng Fan, and Xiaojun Wan. 2020. Automatic generation of citation texts in scholarly papers: A pilot study. In *Proc. of ACL*.

Yumo Xu and Mirella Lapata. 2020. Coarse-to-fine query focused multi-document summarization. In *Proc. of EMNLP*.

Shansong Yang, Weiming Lu, Zhanjiang Zhang, Baogang Wei, and Wenjia An. 2016. Amplifying scientific paper's abstract by leveraging data-weighted reconstruction. *Information Processing & Management*.

Weizhe Yuan, Pengfei Liu, and Graham Neubig. 2021. Can we automate scientific reviewing? *arXiv preprint arXiv:2102.00176*.

Jinming Zhao, Ming Liu, Longxiang Gao, Yuan Jin, Lan Du, He Zhao, He Zhang, and Gholamreza Haffari. 2020. Summpip: Unsupervised multi-document summarization with sentence graph compression. In *Proc. of SIGIR*.

Hao Zheng and Mirella Lapata. 2019. Sentence centrality revisited for unsupervised summarization. In *Proc. of ACL*.

A Appendix

A.1 Citation Function Annotation Proportion of Scientific Papers

Table 6 shows the proportions of *Neutral* and *Comparative* citations annotated by some annotation schemes (Teufel et al., 2006a; Dong and Schäfer, 2011; Jha et al., 2017; Jurgens et al., 2018; Su et al., 2019). From the results, we can find that *Neutral* always has the highest percentage, while *Comparative* always has a low percentage. In conclusion, comparative citations are always challenging to discover.

Schemes	Key Categories	Proportion
Teufel et al. (2006a)	Neut	59.62%
	CoCoGM	4.65%
	CoCoR0	1.27%
	CoCo-	1.54%
	CoCoXY	3.11%
Dong and Schäfer (2011)	Background	65.04%
	Comparison	3.97%
Jha et al. (2017)	Neutral	61.15%
	Comparison	5.82%
Jurgens et al. (2018)	BACKGROUND	51.13%
	COMPARISON	18.07%
Su et al. (2019)	Neutral	70.83%
	Compare	6.42%

Table 6: The proportions of key citations.

A.2 Guideline for Annotating Citation

Our guideline is similar to the scheme of (Teufel et al., 2006a) but with different classifications. Taking the functions of *PModi* and *PBas* as examples, they belong to *Positive* in Teufel's. However, alterations accompany modifications and bases and we thus classify them as *Comparative*. Besides, we add the *Future* function for it is also crucial and unique in researches.

A.3 Annotating Citations of Surveys

We annotate citations in surveys. Specifically, we collect eight scientific surveys from the Artificial

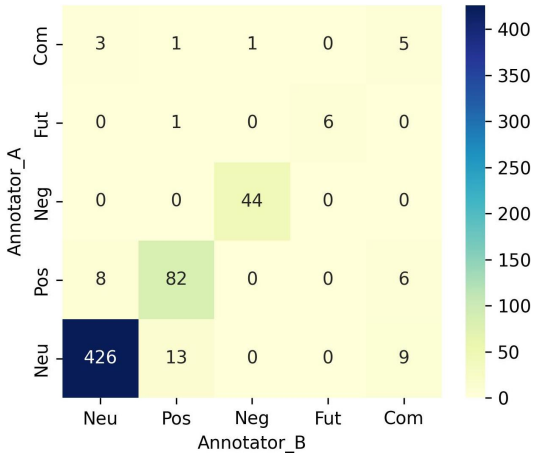

(a) Annotations between A&B.

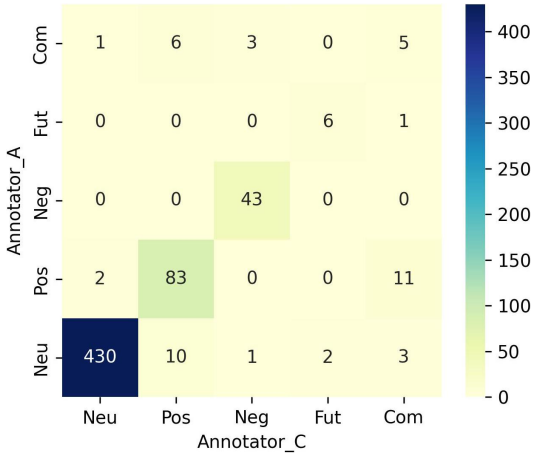

(b) Annotations between A&C.

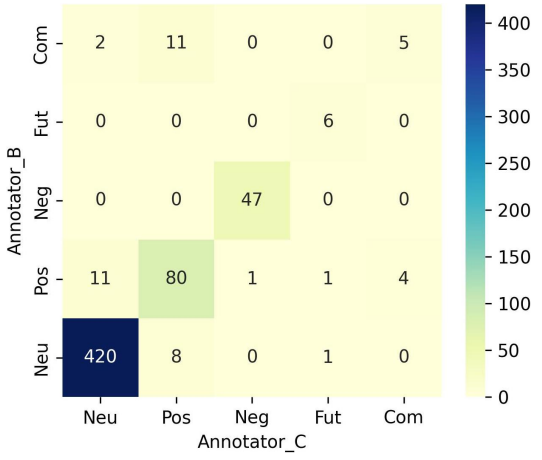

(c) Annotations between B&C.

Figure 5: Consistency of annotations of surveys between three annotators. Neu, Pos, Neg, Fut, and Com stand for the Neural, Positive, Negative, Future, and Comparative, respectively.

Categories	Description
Neutral	Normal descriptions of the cited works, or not enough textual evidence for other categories.
Positive	Authors agree with the cited works, their work and the cited works support each other.
Negative	Authors disagree with the cited works, their work is the opposite of the cited works.
Future	Authors show some hypothesis or feasible future works based on the cited works.
Comparative	Comparisons/Alterations between the works.

Table 7: Our annotating guideline.

	Neutral	Positive	Negative	Future	Comparative
A	73.81%	15.82%	7.25%	1.15%	1.98%
B	72.32%	15.98%	7.41%	0.99%	3.29%
C	71.33%	16.31%	7.91%	1.32%	3.13%
Kappa(n=5; N=607; k=3)=0.8353; **Macro-F**=0.7868					

Table 8: Citations proportions with Kappa and Macro-F.

Intelligence area and extract 607 citations from surveys. Three graduate students use our guideline to annotate 607 citations. The annotation results are shown in Table 8. Figure 5 and the values of Kappa and Macro-F in Table 8 also indicate that the annotations are of high consistency. It can be seen from Table 8 that in scientific surveys, the proportion of citations in the *Comparative* is much lower than that in other categories and is close to *Future*, which rarely appears. In comparison, the proportion of citations in the *Neutral* function is the highest. In conclusion, there are usually little comparisons in surveys. Therefore, the tasks of related work generation and survey generation are not suitable for generating differences in scientific papers, while comparative summaries capture commonalities and differences.

A.4 Implementation Details

We utilize TF-IDF and BERT (Devlin et al., 2018) to get sentence representation. The hyperparameters (α, β) are set as (0.9, 0.1), (λ_1, λ_2, λ_3) are set as (0.33, 0.33, 0.33).

https://github.com/google-research/bert

JPG - Jointly Learn to Align: Automated Disease Prediction and Radiology Report Generation

Jingyi You[♡], **Dongyuan Li**[♡], **Manabu Okumura**[♡], **Kenji Suzuki**[♠]
Tokyo Institute of Technology
[♡]{youjy,lidy,oku}@lr.pi.titech.ac.jp
[♠]suzuki.k.di@m.titech.ac.jp

Abstract

Automated radiology report generation aims to generate paragraphs that describe fine-grained visual differences among cases, especially those between the normal and the diseased. Existing methods seldom consider the cross-modal alignment between textual and visual features and tend to ignore disease tags as an auxiliary for report generation. To bridge the gap between textual and visual information, in this study, we propose a "**J**ointly learning framework for automated disease **P**rediction and radiology report **G**eneration (JPG)" to improve the quality of reports through the interaction between the main task (report generation) and two auxiliary tasks (feature alignment and disease prediction). The feature alignment and disease prediction help the model learn text-correlated visual features and record diseases as keywords so that it can output high-quality reports. Besides, the improved reports in turn provide additional harder samples for feature alignment and disease prediction to learn more precise visual and textual representations and improve prediction accuracy. All components are jointly trained in a manner that helps improve them iteratively and progressively. Experimental results demonstrate the effectiveness of JPG on the most commonly used IU X-RAY dataset, showing its superior performance over multiple state-of-the-art image captioning and medical report generation methods with regard to BLEU, METEOR, and ROUGE metrics.

1 Introduction

Writing radiology reports and predicting disease labels are two essential procedures in clinical practice. However, manually creating them by radiologists is laborious and time-consuming (Jing et al., 2018; Chen et al., 2021b). Therefore, automated radiology report generation and disease prediction, which aim to generate formal-format descriptive texts (Fig. 1 *Findings*) and clinical conclusive terminologies (Fig. 1 *MeSH*), have received increasing attention recently (Chen et al., 2020; Miura et al., 2021; Liu et al., 2021b; Nguyen et al., 2021; Liu et al., 2021c; You et al., 2021a). In particular, they not only improve the efficiency of the entire procedure and liberate people from burdensome workloads, but also maintain the high quality of healthcare.

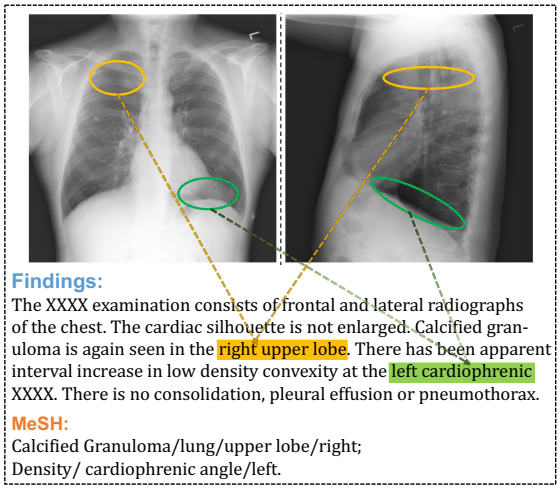

Figure 1: Chest X-ray images and an accompanying report, including *Findings* and *MeSH* labels, from the IU X-RAY dataset. We marked the aligned visual and textual features in different colors for better illustration.

In spite of substantial improvements (Zhang et al., 2020; Wang et al., 2022; Liu et al., 2021a; Shao et al., 2021) have been achieved in the automatic radiology report generation and disease prediction, several challenges remain unsolved. Firstly, following traditional image captioning paradigms (Bhattacharya et al., 2022), current methods mainly adopt a standard encoder-decoder framework with convolutional neural networks (CNNs) encoding radiographs and recurrent neural networks (e.g., LSTM/GRU) or non-recurrent neural networks (e.g., Transformer) decoding reports. As a result, visual and textual information are represented by different encoding methods in their own specific embedding spaces, so that the features are misaligned (e.g., the visual represen-

Proceedings of the 29th International Conference on Computational Linguistics, pages 5989–6001
October 12–17, 2022.

tation of the regions circled in yellow in Fig. 1 is significantly different from the textual representation of *"right upper lobe"* in *Findings*). Therefore, directly applying these visual features to the downstream task will lead to low-quality reports (Chen et al., 2021a,b; Lu et al., 2017).

Furthermore, most existing disease prediction models (Bhattacharya et al., 2022; Sun et al., 2021; Gheflati and Rivaz, 2021; Park et al., 2022) attach a single disease label to each image, where its context (e.g., location, severity, and affected organs) is seldom considered. Automatically mining context-aware disease labels can thus make it easier to understand the disease. Finally, current approaches take only visual information as the input of the downstream report generation, which ignores context-aware disease tags as auxiliary textual information. Intuitively, as high-level conclusive features, disease tags can more effectively guide the text generation and alleviate missing keywords.

To overcome the aforementioned problems, we propose to integrate radiology report generation and context-aware disease prediction into an overall framework (JPG), where context-aware disease labels serve as high-level auxiliary information for facilitating the report with the lesion location. Specifically, both visual and textual features are first projected into a shared subspace via a shared base matrix to learn new visual and textual representations. The shared base matrix acts as an intermediate medium, which enables visual and textual information to sufficiently interact and fuse in a manner that relieves misalignment between the features. As for the second issue, we train a CNN-RNN architecture to automatically search for context-aware disease labels. Instead of directly using the output of the CNN, the aligned visual features are applied to initialize the RNN hidden state for context-aware disease label prediction. Consequently, the model can improve the classification accuracy and disease label quality. Finally, we incorporate context-aware disease labels as high-level auxiliary features together with aligned visual features into the decoder, so that the comprehensive disease tags can better guide the report generation.

We highlight the contributions as follows:

• We propose to learn visual and textual representations through a shared subspace to relieve the misalignment across modalities, which can also be easily transplanted to other multi-modal tasks.

• Instead of directly using single labels in the disease prediction task, we propose a strategy to mine context-aware labels to provide a more detailed textual conclusion for lesions in radiographs.

• As far as we know, we are the first to use predicted disease contextual labels as high-level auxiliary information for facilitating and guiding the report generation process. Empirical results demonstrate that this scheme proposal outperforms state-of-the-art competitors in terms of the automated radiology report generation.

2 Related Work

2.1 Image Captioning

Image captioning aims to generate sentences that describe images, and it has achieved great success in the cross-modal area (Cornia et al., 2020; Zhou et al., 2020; Shi et al., 2021). Inspired by encoder-decoder architectures used in machine translation, most existing image captioning approaches typically adopt the CNN-RNN framework (Huang et al., 2019; Yan et al., 2021; You et al., 2021b), where a CNN is used to extract visual features from a given image, and a recurrent or non-recurrent network is used to generate the caption. To align visual features with textual features, existing methods adopt a memory network (Chen et al., 2020, 2021b), a relation/consensus graph (Wang et al., 2021a; Bhattacharya et al., 2022), a Transformer network (Ji et al., 2021) or a language model (Sariyildiz et al., 2020; Gupta et al., 2020) to help visual features learn new semantic representations. Among those studies, the most related ones (You et al., 2018; Akbari et al., 2019) directly project visual features to a textual space and consider textual features as basis vectors to learn new representations for visual features. In contrast, in the present study, we design a shared subspace and a base matrix as an intermediate medium to learn new representations for both visual and textual features, which can thereby be better aligned.

2.2 Radiology Report Generation

As one of the applications and extensions of image captioning (Cornia et al., 2020; Zhou et al., 2020; Shi et al., 2021; Huang et al., 2019; Yan et al., 2021) (Appendix 2.1) to the medical domain, radiology report generation aims to annotate radiographs with much more detailed professional reports. According to the strategies for aligning radiological visual and textual features, current methods can be generally classified into three categories: 1) *vari-*

ant attention mechanism-based methods seek to integrate and fuse visual and textual features via advanced attention (Jing et al., 2019; Wang et al., 2018; Liu et al., 2019), among which Jing et al. (2018) propose a multi-task hierarchical model with a co-attention mechanism to combine visual and textual features to generate reports. 2) *cross-modal memory network*-based approaches record the alignment between images and texts through a shared matrix to facilitate the information interaction across modalities (Yin et al., 2019; Chen et al., 2020, 2021b; Wang et al., 2021b). 3) *graph convolution network*-based models aggregate visual and textual features on pre-trained knowledge graphs or newly constructed multi-modal networks (Zhang et al., 2020; Hu et al., 2019). JPG offers a new way beyond the above studies to generate radiology reports, since a shared subspace is provided to learn new representations for both visual and textual features in a manner that produces more accurate descriptions for report generation.

2.3 Medical Image Classification

Existing methods have achieved remarkable success at predicting single disease labels for medical images (Bhattacharya et al., 2022; Sun et al., 2021; Gheflati and Rivaz, 2021; You et al., 2022). In particular, informative disease labels have been mined with context information. For example, Shin et al. (2016) predicts disease labels by leveraging a variant of the CNN-RNN framework. Moreover, PPKED (Liu et al., 2021b) examines abnormal regions and assigns disease topic tags to the abnormalities. Differing from the above-mentioned methods, our JPG adopts a shared base metric for learning new visual representations and takes it as input for context-aware disease prediction to improve the fluency of disease labels and classification accuracy.

3 Methodology

Figure 2 exhibits an overview of JPG, which consists of three chief components: (A) *shared subspace representation learning*, (B) *context-aware disease prediction*, and (C) *radiology report generation*. Hereafter, we will give formal notations of variables and task definitions concerning JPG, and introduce each component subsequently in detail.

3.1 Notations and Task Definition

Given an X-ray image $\mathbf{I}$ as input, JPG is designed to automatically generate a sequence of context-aware disease labels $\boldsymbol{c}$ and a radiology report $\mathbf{Y}$. Specifically, we divide $\mathbf{I}$ into p patches, and apply pre-trained CNN-based ResNet (He et al., 2016) as the visual extractor to learn its patch features as $\mathbf{X} = \{\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_p\}$, where $\mathbf{x}_p \in \mathbb{R}^{d_x}$ with d_x representing the dimensionality of patch features. The target output is the corresponding radiology report $\mathbf{Y} = \{\mathbf{y}_1, \mathbf{y}_2, \ldots, \mathbf{y}_n\}$, where $\mathbf{y}_n \in \mathbb{R}^{d_y}$ is the word embedding of the n-th generated token, and n denotes the length of the report. Formally, the entire task can be defined as two parts according to Bayes' theorem as follows:

$$p(\mathbf{Y}, \boldsymbol{c}|\mathbf{X}) \propto p(\mathbf{Y}|\boldsymbol{c}, \mathbf{X}) \cdot p(\boldsymbol{c}|\mathbf{X}), \quad (1)$$

where the radiology report generation process $p(\mathbf{Y}|\boldsymbol{c}, \mathbf{X})$ can be formalized as a recursive application of the chain rule as

$$p(\mathbf{Y}|\boldsymbol{c}, \mathbf{X}) = \prod_{i=1}^{n} p(\mathbf{y}_i|\mathbf{y}_{<i}, \boldsymbol{c}, \mathbf{X}), \quad (2)$$

where $\mathbf{y}_{<i} = \{\mathbf{y}_1, \ldots, \mathbf{y}_{i-1}\}$ represents the previously generated tokens so far, and n is the total amount of tokens in target sequence $\mathbf{Y}$.

As described in Eq. 1, jointly learning to align diagnostic disease prediction and radiological report generation can be classified as two subtasks in order. In detail, we first train the model to maximize the probability of producing context-aware disease labels for an X-ray image $p(\boldsymbol{c}|\mathbf{X})$, then maximize the probability of generating a corresponding radiology report $p(\mathbf{Y}|\boldsymbol{c}, \mathbf{X})$ conditioned on context-aware disease labels $\boldsymbol{c}$ and visual features $\mathbf{X}$.

3.2 Visual Extractor

As shown in Fig. 2, given a radiology image $\mathbf{I}$ organized in 2-dimension format as input, we employ ResNet (He et al., 2016) as a pre-trained visual extractor. Normally, it first decomposes the image into regions of equal size, i.e., patches, and then extracts visual features of each patch from the output of its last convolutional layer. Afterwards, the extracted patch representations $\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_p$ are concatenated to constitute the source input for all subsequent modules with the form of visual feature sequence $\mathbf{X} \in \mathbb{R}^{p \times d_x}$ as

$$\{\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_p\} = f_v(\mathbf{I}). \quad (3)$$

Note that any type of pre-trained CNNs, e.g., VGG (Simonyan and Zisserman, 2015) or DenseNet (Huang et al., 2017), can be used for the purpose.

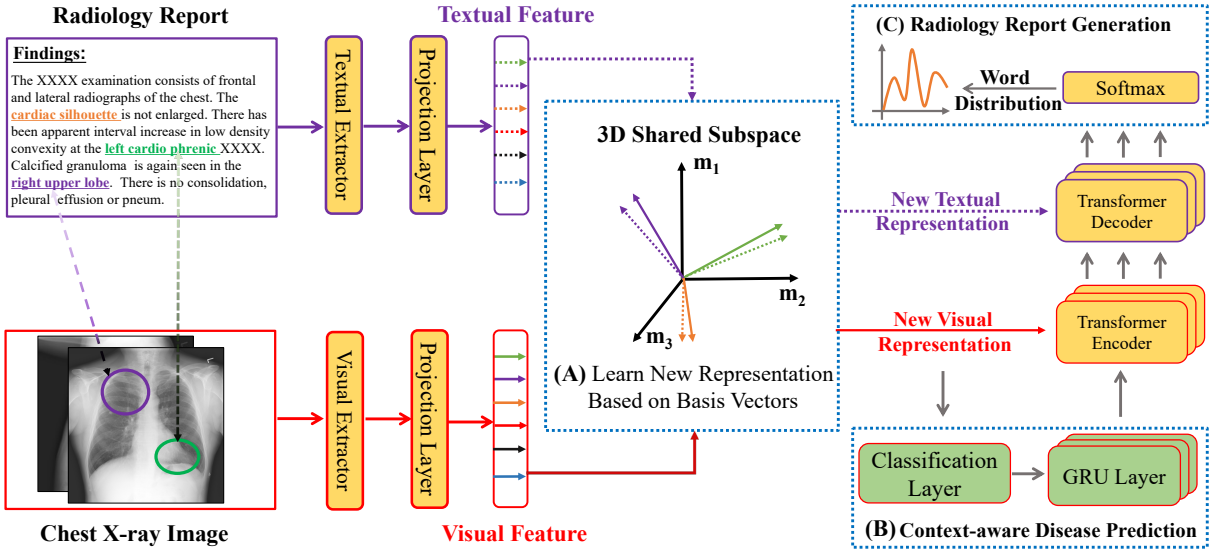

Figure 2: Model overview. JPG first captures textual and visual features through word embeddings and a visual extractor. Those features are then projected onto a shared subspace ((A) as a 3-dimension example) to learn new representations based on shared basis vectors. Finally, a RNN decoder (B) and a Transformer-based encoder-decoder architecture (C) are employed to generate context-aware disease labels and radiology reports, respectively.

3.3 Shared Subspace Representation

Considering that visual and textual features are extracted by different encoding methods (Kim et al., 2020; Huang et al., 2020), directly applying patch features generated by the visual extractor as the input for the downstream text generation task will lead to non-fluent, low-quality reports with missing keywords. To solve this problem, as shown in Fig. 2 (A), both visual and textual features are projected into a shared subspace, and a trainable shared base matrix is designed to learn new representations for them. Therefore, textual and visual features can be fully integrated and interacted to relieve the feature discontinuity across modalities.

Specifically, we define a shared base matrix $\mathbf{B}$ with m basis vectors as $\mathbf{B} = \{\mathbf{b}_1, \mathbf{b}_2, \ldots, \mathbf{b}_m\}$, where $\mathbf{B} \in \mathbb{R}^{m \times d_b}$ with d_b representing the dimensionality of each basis vector. Besides, based on the assumption that the dimension of the shared subspace is d_s, visual features $\mathbf{X}$, textual features $\mathbf{Y}$, and shared base matrix $\mathbf{B}$ are projected into the shared subspace respectively as

$$\tilde{\mathbf{x}}_i = \mathbf{W}_x \cdot \mathbf{x}_i \quad \& \quad \widetilde{\mathbf{X}} = \mathbf{X} \cdot \mathbf{W}_x, \tag{4}$$

$$\tilde{\mathbf{y}}_i = \mathbf{W}_y \cdot \mathbf{y}_i \quad \& \quad \widetilde{\mathbf{Y}} = \mathbf{Y} \cdot \mathbf{W}_y, \tag{5}$$

$$\tilde{\mathbf{b}}_i = \mathbf{W}_b \cdot \mathbf{b}_i \quad \& \quad \widetilde{\mathbf{B}} = \mathbf{B} \cdot \mathbf{W}_b, \tag{6}$$

where $\mathbf{W}_x \in \mathbb{R}^{d_x \times d_s}$, $\mathbf{W}_y \in \mathbb{R}^{d_y \times d_s}$, and $\mathbf{W}_b \in \mathbb{R}^{d_b \times d_s}$ are trainable parameters.

To learn new visual and textual representations given base matrix $\mathbf{B}$, we calculate the consine similarity between the previous visual and textual features with $\mathbf{B}$ as

$$\mathbf{S}_{ij} = \tilde{\mathbf{x}}_i^T \cdot \tilde{\mathbf{b}}_j \quad \& \quad \mathbf{G}_{ij} = \tilde{\mathbf{y}}_i^T \cdot \tilde{\mathbf{b}}_j \tag{7}$$

where T represents matrix transpose, $\mathbf{S}_{ij}$ denotes the similarity between the i-th visual feature $\tilde{\mathbf{x}}_i$ and the j-th basis vector representation $\tilde{\mathbf{b}}_j$. Similarly, $\mathbf{G}_{ij}$ is the similarity between the i-th textual feature $\tilde{\mathbf{y}}_i$ and $\tilde{\mathbf{b}}_j$. To prevent inaccurate representation learning caused by an excessive weight of a certain item, the similarities are further normalized by

$$\mathbf{S}_{ij} = \frac{\exp\left(\mathbf{S}_{ij}\right)}{\sum_{k=1}^{m} \exp\left(\mathbf{S}_{ik}\right)} \tag{8}$$

$$\mathbf{G}_{ij} = \frac{\exp\left(\mathbf{G}_{ij}\right)}{\sum_{k=1}^{m} \exp\left(\mathbf{G}_{ik}\right)}. \tag{9}$$

Finally, the new visual and textual representations are obtained as

$$\mathbf{r}_{x_i} = \sum_{k=1}^{m} \mathbf{S}_{ik} \cdot \tilde{\mathbf{b}}_k \quad \& \quad \mathbf{r}_{y_i} = \sum_{k=1}^{m} \mathbf{G}_{ik} \cdot \tilde{\mathbf{b}}_k \tag{10}$$

where $\mathbf{r}_{x_i}$ and $\mathbf{r}_{y_i}$ are the i-th new visual feature and textual feature, respectively.

The above process guarantees the full integration between textual and visual information; that is, the visual features of a certain patch and its corresponding descriptive textual features maintain

DATASET	IMAGE	REPORT	PATIENT	AVG. LEN.
TRAIN	5,226	2,770	2,770	37.56
VALID	748	395	395	36.78
TEST	1,496	790	790	33.62

Table 1: Basic statistics of IU X-RAY with respect to its training, validation, and test sets. "AVG. LEN." represents the averaged word-based length of reports.

similar representations in the shared subspace. For example, as shown in Fig. 2, the green solid line and dotted line represent the visual and textual features of *left cardiophrenic*, respectively, of which ones with similar representations are gathered in the 3D shared subspace as illustrated in Fig. 2 (A).

3.4 Context-aware Disease Prediction

Considering that a single disease label cannot fully account for the context of an X-Ray image, including location, severity, and organs affected by a disease, mining context-aware labels for radiographs and using them to train a classification layer for disease prediction are proposed hereafter.

Mining and pre-training on single labels. In accordance with Shin et al. (2016), we find 17 simplest unique disease annotation patterns through statistical analysis to label the images and retain 40% of the full dataset. GoogLeNet (Szegedy et al., 2015) is used as the classification layer to train the model on the retained cases. We additionally apply mini-batch normalization (Ioffe and Szegedy, 2015) and random data dropout (Hinton et al., 2012) to alleviate result deviation caused by an unbalanced distribution between normal and pathological cases. Since the majority of disease-related MeSH terms contain up to 5 words, we constrain the GRU decoder to unroll up to 5 timesteps. Specifically, we initialize the first decoder hidden state as the output embedding of the classification layer. The GRU decoder is then trained by minimizing the negative log likelihood between the output sequence and the ground-truth:

$$\mathcal{L}_{Loss} = -\sum_{t=1}^{N}\{\boldsymbol{c}_t = \boldsymbol{s}_t|\mathbf{r}_{x_1}, \ldots, \mathbf{r}_{x_p}\} \quad (11)$$

where $\boldsymbol{c}_t$ is the token output on the t-th timestep, $\boldsymbol{s}_t$ is the t-th reference MeSH term, and $N = 5$.

Re-training on context-aware labels. The aforementioned classification layer and GRU decoder are considered as a pre-training procedure to mine the context for previous primary disease labels in the whole dataset. And 57 unique context-aware disease labels on the side of the output of the GRU decoder are obtained. The context-aware labels summarize both the context information and textual semantic information of the image. For example, the coarse-grained label "calcified granuloma" can be attached by more informative and detailed context as "calcified granuloma in right upper lobe" or "small calcified granuloma in left lung base". This additional labelling procedure improves the quality of clinical practice concerning X-ray diagnosis. As shown in Fig. 2 (B), we re-train the classification layer with 57 context-aware labeled cases, and initialize the GRU hidden state with the output of the classification layer. Eq. 11 is again used as an objective function for the re-training process.

3.5 Automated Radiology Report Generation

As shown in Fig. 2 (C), we employ a Transformer-based encoder-decoder architecture for automated radiology report generation. New visual and textual representations are functionalized as the input for the Transformer encoder and decoder, respectively.

Since considering context-aware disease labels as macro-level features also benefits clinical report generation, we concatenate macro-level context-aware labels with micro-level visual features as the input for the Transformer encoder. Specifically, the new representation of micro-level visual features $\{\mathbf{r}_{x_1}, \ldots, \mathbf{r}_{x_p}\}$ and macro-level context-aware label features $\boldsymbol{c}$ are first fed into the encoder as

$$\{\mathbf{z}_1, \ldots, \mathbf{z}_p, \mathbf{z}_c\} = f_e(\mathbf{r}_{x_1}, \ldots, \mathbf{r}_{x_p}, \boldsymbol{c}), \quad (12)$$

where $f_e(\cdot)$ represents the Transformer encoder. Then, resulting intermediate state $\{\mathbf{z}_1, \ldots, \mathbf{z}_p, \mathbf{z}_c\}$ are fed into the decoder at each decoding step with aligned textual representation of the previously generated sequence $\{\mathbf{r}_{y_1}, \ldots, \mathbf{r}_{y_{i-1}}\}$. The output at the i-th timestep can thus be generated by using

$$\mathbf{y}_i = f_d(\mathbf{z}_1, \ldots, \mathbf{z}_p, \mathbf{z}_c, \mathbf{r}_{y_1}, \ldots, \mathbf{r}_{y_{i-1}}), \quad (13)$$

where $f_d(\cdot)$ refers to the Transformer decoder.

4 Experiments

4.1 Dataset

We carried out our experiments on the most widely-used and conventional benchmark dataset, namely, Indiana University Chest X-Ray Collection[1] (IU X-RAY) (Demner-Fushman et al., 2016). It contains

[1] https://openi.nlm.nih.gov/

3,955 fully de-identified handwritten radiology reports from the Indiana Network for Patient Care and 7,470 corresponding chest X-ray images from the hospitals' picture archiving systems. As shown in Fig. 1, each sample is associated with a frontal and/or a lateral chest X-ray image, and each report is comprised of several sections: *MeSH*[2], *Indication*, *Findings*, and *Impression*, etc. In this work, we use the *Findings* and *MeSH* sections as ground-truth reports and disease labels, respectively.

Following the dataset preprocessing procedure of previous studies (Li et al., 2018), we preprocess the reports by tokenizing, converting tokens into lower cases, and removing non-alphabetic tokens. Samples without *MeSH* or *Findings* sections in the dataset were excluded. We apply the same split, i.e., 70%/10%/20% for the training/validation/test set, as that stated in Li et al. (2018). The basic statistics of IU X-RAY, in terms of numbers of images, reports, patients, and average length of reports with respect to each split set, are listed in Table 1.

4.2 Baselines

The following excellent baselines are used to examine the effectiveness of the proposed approach on radiology report generation: conventional image captioning methods including NIC (Vinyals et al., 2015), ADAATT (Lu et al., 2017), ATT2IN (Rennie et al., 2017), and VisualGPT (Chen et al., 2021a); and the ones proposed for the medical domain, e.g., COATT (Jing et al., 2018), HRGR (Li et al., 2018), CMAS-RL (Jing et al., 2019), R2GEN (Chen et al., 2020), and CMN (Chen et al., 2021b). In addition, BASE is a vanilla Transformer (Vaswani et al., 2017) used as the backbone encoder-decoder architecture in our full model. We further implement several ablated versions of JPG with the aim of evaluating the different components in it.

4.3 Evaluation Metrics

The performance of the aforementioned baselines, as well as our proposed method, was evaluated by conventional natural language generation (NLG) metrics, including BLEU (Papineni et al., 2002), METEOR (Denkowski and Lavie, 2011), and ROUGE-L (Lin, 2004), which compare model-generated reports with ground-truth by referring to the overlap of n-grams (BLEU-n), explicit word-to-word matches (METEOR), and longest common subsequence (ROUGE-L). The results based on these metrics were obtained by the standard image captioning evaluation tool[3]. We further measured the disease prediction subtask as a multi-label classification problem by the micro-averaged F1 score.

4.4 Implementation Details

Two X-Ray images of a patient were used as the input for both the report generation and disease annotation subtasks to ensure consistency with previous studies (Li et al., 2018; Chen et al., 2021b), where all the CNN input images were rescaled to a size of 256×256. We employed ResNet101 (He et al., 2016) pre-trained on ImageNet (Deng et al., 2009) as the visual extractor to extract patch features with a $7 \times 7 \times 2048$-dimension feature map. The maximum decoding sequence lengths are limited to 60 and 5 tokens for report generation and disease annotation respectively by truncating and zero-padding. 512-dimension word embeddings with random initialization were fine-tuned during training. We randomly initialized the shared subspace as a 512×2048 memory matrix, where $d_s = 512$, and 2048 is the number of shared basis vectors. We adopted GoogLeNet as the classification layer, and a single-layer GRU unrolling up to five timesteps for context-aware disease label prediction. A 3-layer Transformer structure with 8 attention heads and 512-dimension hidden states was used in randomly initialized states as the encoder-decoder backbone.

Our model is trained under a cross entropy loss. As for the optimizer, Adam (Kingma and Ba, 2015) with a learning rate of $1e-4$ and an initial accumulator value of 0.1 was used. We set the batch size to 16, whereas the target sequences were decoded through beam search with a beam size of 3 at test time to balance the effectiveness and efficiency.

5 Results and Discussion

5.1 Performance of JPG

Table 2 lists the main results on the radiology report generation task. Symbol † indicates statistically significant differences of JPG from BASE using T-test (Yang and Liu, 1999). The results for the conventional image captioning methods are shown at the top, with the ones proposed for the medical domain in the middle, and those for our methods at the bottom. According to Table 2, JPG can gen-

[2]https://www.nlm.nih.gov/mesh/meshhome.html

[3]https://github.com/tylin/coco-caption

METHOD	BLEU-1	BLEU-2	BLEU-3	BLEU-4	METEOR	ROUGE-L
NIC (Vinyals et al., 2015)	0.216	0.124	0.087	0.066	-	0.306
ADAATT (Lu et al., 2017)	0.220	0.127	0.089	0.068	-	0.308
ATT2IN (Rennie et al., 2017)	0.224	0.129	0.089	0.068	-	0.308
VISUALGPT (Chen et al., 2021a)	0.482	0.314	0.221	0.158	0.204	0.375
COATT (Jing et al., 2018)	0.455	0.288	0.205	0.154	-	0.369
HRGR (Li et al., 2018)	0.438	0.298	0.208	0.151	-	0.322
CMAS-RL (Jing et al., 2019)	0.464	0.301	0.210	0.154	-	0.362
R2GEN (Chen et al., 2020)	0.470	0.304	0.219	0.165	0.187	0.371
CMN (Chen et al., 2021b)	0.475	0.309	0.222	0.170	0.191	0.375
BASE	0.369	0.254	0.179	0.135	0.164	0.342
JPG-projection	0.458	0.291	0.212	0.159	0.177	0.371
JPG-auxiliary	0.472	0.308	0.218	0.168	0.188	0.373
JPG	**0.479**†	**0.319**†	**0.222**†	**0.174**†	**0.193**†	**0.377**†

Table 2: Comparison of the proposed model with those of previous studies for *Findings* generation on the test set of IU X-RAY with respect to various NLG metrics, where BLEU-n denotes BLEU scores using up to 4-grams. † marked results significantly surpass BASE using T-test (Yang and Liu, 1999) with $p < 0.05$.

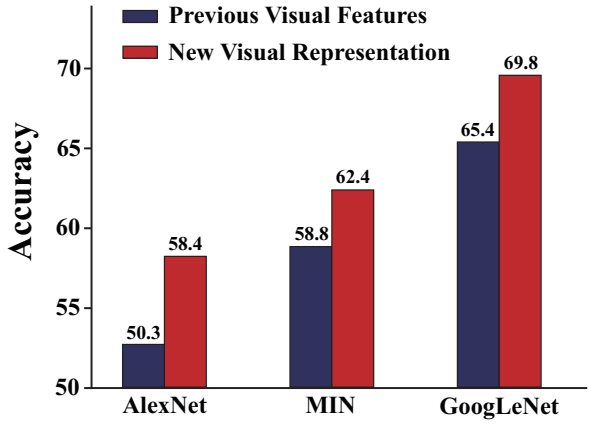

Figure 3: Classification accuracy of AlexNet, NIN, and GoogLeNet on the test set of IU X-RAY.

erate more accurate and fluent radiology reports compared with the baselines.

We consider three possible reasons for the superior performance of JPG. First, the shared subspace is configured to make up for the gap between different information extracted by word and image embeddings. Compared to simply merging word embeddings of disease tags into patch features as complementary textual information, the additional shared subspace projection makes the aligned visual and textual features much more understandable to each other, so that information is better interacted, and the quality of reports is improved. Second, regarding the improvement of BLEU scores, the introduction of context-aware disease labels provides the report generation process with explicit lesion textual prompts, which prevents our model from generating irrelevant diseases and enables JPG to effectively capture the disease-related keywords. Third, conclusive disease prediction and descriptive report generation are jointly trained and optimized in an overall framework to obtain a globally optimal solution for both subtasks.

As shown in Fig. 3, the three most effective classification networks, AlexNet (Krizhevsky et al., 2012), NIN (Lin et al., 2014), and GoogLeNet (Szegedy et al., 2015) were employed for classification with context-aware disease labels. Compared with adopting patch features directly extracted from the visual extractor, learning new visual representations from a shared subspace can dramatically improve classification accuracy, because new visual representations contain more useful semantic features in regard to the classification task. Therefore, many inspiring context-aware disease labels, such as <opacity lung bilateral interstitial diffuse> and <opacity lung lower_lobe bilateral>, can be obtained.

5.2 Ablation Study

JPG-projection To verify the alignment between visual and textual representations within the encoder-decoder architecture, we show the ablation performance in Table 2 by removing the shared

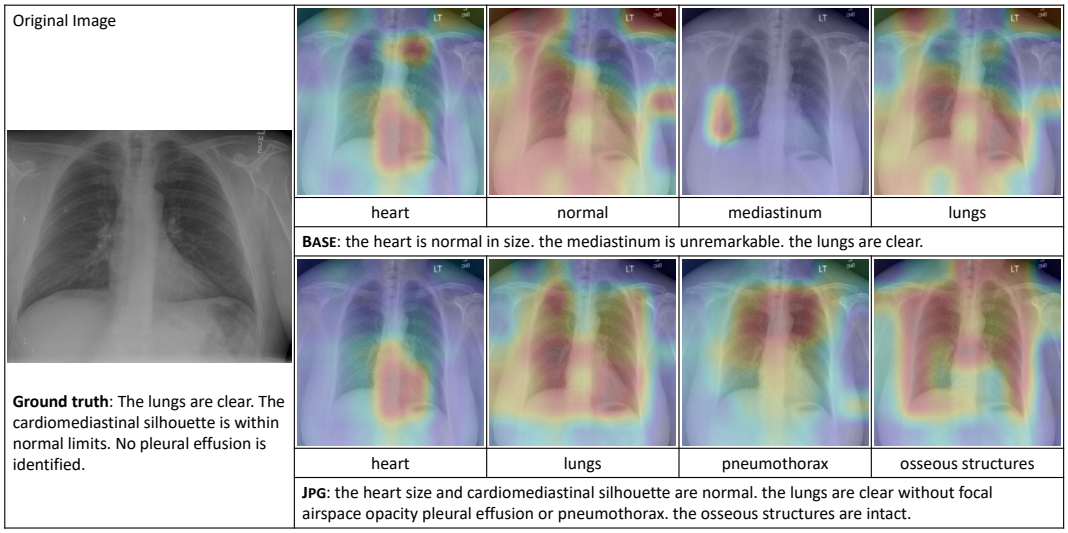

Figure 4: Visualization of image-text mappings between particular regions (indicated by colored weights) of a chest X-ray image and tokens from its reports generated by BASE and JPG, respectively.

subspace projection and simply using the raw visual extractor and word embedding outputs to both predict disease labels and generate reports, which obviously degrades the model performance with respect to all evaluation metrics. This proves that the shared base matrix plays a critical role in facilitating disease prediction and report generation with sufficient understandable visual representations with semantic meanings, which cannot be replaced by straightforward visual and textual features. Besides, instead of using hard attention to match visual features with textual features, the proposed shared subspace acts as a soft alignment medium to offset the gap between those features; it thus unifies cross-modal features within the same representation space. Furthermore, the shared subspace also provides further fusion patterns for disease tags and chest X-Ray images to communicate with each other and pass both compatible visual and textual information for more accurate reports.

JPG-auxiliary Based on the assumption that the remarkable improvement of JPG from baselines is due to jointly training disease prediction and report generation and employing the predicted disease tags as auxiliary information when generating reports, we would like to experimentally evaluate the performance of JPG in terms of a separate learning pattern. In this experiment, the disease prediction and report generation subtasks were treated as two parallel procedures. Specifically, visual and textual features were first projected into a shared subspace to overcome the misalignment of features across modalities. Then, we independently employed a Transformer encoder-decoder structure for report generation without adding context-aware disease labels as auxiliary information on the input side.

According to the last block in Table 2, implementing the subtasks individually degrades the model performance and the quality of generated reports to a certain extent. We consider that in our complimentary interactive learning framework, reports can receive more discriminative lesion locations and semantic features under the guidance and constraint of predicted diseases. And in turn disease prediction accuracy is improved by report generation via visual feature extraction and fusion in a manner that cannot be imitated by separate learning. This result indicates the superiority of JPG over the conventional methods, implying the usage of auxiliary disease tags in the report generation process is promising for identifying salient keywords.

5.3 Alignment Visualization and Case Study

To further qualitatively investigate the ability of JPG to overcome the misalignment of features across modalities, Fig. 4 visualizes how the proposed model focuses on the image when generating a certain word or phrase; i.e., it learns from the alignments between visual and textual features. We randomly select an example from the IU-XRAY dataset, and list its original chest X-Ray image with the corresponding ground-truth report for reference. Fig. 4 shows image-text mappings between particular regions (highlighted by colored weights)

Frontal	Lateral	Ground-truth	BASE	JPG
	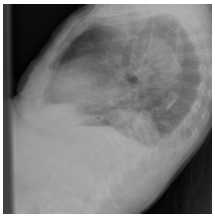	Low lung volumes. Stable ectasia of the thoracic aorta. Stable right upper mediastinal Bilateral small pleural effusions and bibasilar airspace opacities. The heart size and mediastinal silhouette are within normal limits for contour. No pneumothorax. Stable wedging of the anterior thoracic vertebral bodies.	Cardio mediastinal silhouette is unremarkable. Visualized osseous structures of the thorax are without acute abnormality. Low lung volumes bilaterally. The lungs are clear bilaterally. Specifically no evidence of focal consolidation pneumothorax or pleural effusion.	There are low lung volumes with bibasilar opacities xxxx representing subsegmental atelectasis. The cardio the cardiac silhouette is of the xxxx of normal in size and contour. There is no pneumothorax or large pleural effusion.
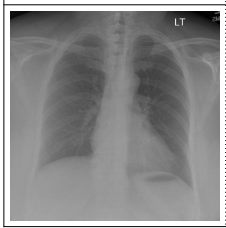		The lungs are clear. The cardiomediastinal silhouette is within normal limits. No pleural effusion is identified.	The lungs are clear bilaterally. Cardio mediastinal silhouette is unremarkable. Visualized osseous structures of the thorax are without acute abnormality.	The heart size and cardiomediastinal silhouette are normal. The lungs are clear without focal airspace opacity pleural effusion or pneumothorax. The osseous structures are intact.

Figure 5: Example reports of BASE and JPG.

of an X-Ray image and words/phrases from its reports generated by BASE and JPG. In detail, we utilize the cross attention weight from the first decoder layer to show the alignment between visual and textual features, since the latter decoder layers couple the textual and visual information, making it difficult to distinguish the most primitive alignment weights. In general, JPG is able to pay attention to relatively accurate patches when generating a word (especially disease terminologies), so it brings about descriptions of higher quality than those produced by BASE.

Fig. 5 exhibits two examples with both front and lateral CXR images and their corresponding reports obtained by ground-truth, BASE, and JPG, where different colors on the texts indicate different clinical terms. These examples indicate that JPG can produce accurate terms and well-aligned descriptions, which abide by a similar content flow as radiologists follow, while BASE sometimes makes factual errors. For example, in both cases, patterns in the ground-truth and generated reports follow the sequence of starting from observations (e.g., "lung volumes" and "cardiomediastinal silhouette") and concluding with potential diseases (e.g., "pleural effusion" and "pneumothorax"). In addition, JPG-generated reports cover almost all of the necessary clinical terminologies in the ground-truth reports. On the contrary, BASE cannot keep abreast with the description order of the ground-truth, so it generates misaligned and out-of-order sentences. Moreover, several phrases go against fact; e.g., "small pleural effusions" is mistakenly ignored. By longitudinally viewing the reports produced by BASE corresponding to two cases, we can also find that the vanilla Transformer tends to iteratively generate similar sentences.

6 Conclusion

We addressed several fundamental issues concerning clinical disease prediction and radiology report generation in an overall framework, where context-aware disease terminologies act as complementary textual features coupled with visual features of images to guide and facilitate the report generation process. Meanwhile, these explicit clues of lesion location effectively prevent the report generation model from generating factual erroneous texts. The proposed shared subspace provides an interaction platform for different representations extracted by image and word embeddings to overcome the misalignment of information across modalities. Empirical results acquired with the most widely used dataset, including those of ablation studies, demonstrate the effectiveness of the proposed JPG, which achieves the state-of-the-art performance.

Acknowledgements

We would like to gratefully thank the anonymous reviewers for their helpful comments and feedback. Jingyi You and Dongyuan Li acknowledge the support from China Scholarship Council (CSC).

References

Hassan Akbari, Svebor Karaman, Surabhi Bhargava, Brian Chen, Carl Vondrick, and Shih-Fu Chang. 2019. Multi-level multimodal common semantic space for image-phrase grounding. In *IEEE Conference on Computer Vision and Pattern Recognition, CVPR*

2019, Long Beach, CA, USA, June 16-20, 2019, pages 12476–12486. Computer Vision Foundation / IEEE.

Moinak Bhattacharya, Shubham Jain, and Prateek Prasanna. 2022. Radiotransformer: A cascaded global-focal transformer for visual attention-guided disease classification. *CoRR*, abs/2202.11781.

Jun Chen, Han Guo, Kai Yi, Boyang Li, and Mohamed Elhoseiny. 2021a. Visualgpt: Data-efficient image captioning by balancing visual input and linguistic knowledge from pretraining. *CoRR*, abs/2102.10407.

Zhihong Chen, Yaling Shen, Yan Song, and Xiang Wan. 2021b. Cross-modal memory networks for radiology report generation. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing, ACL/IJCNLP 2021, (Volume 1: Long Papers), Virtual Event, August 1-6, 2021*, pages 5904–5914. Association for Computational Linguistics.

Zhihong Chen, Yan Song, Tsung-Hui Chang, and Xiang Wan. 2020. Generating radiology reports via memory-driven transformer. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing, EMNLP 2020, Online, November 16-20, 2020*, pages 1439–1449. Association for Computational Linguistics.

Marcella Cornia, Matteo Stefanini, Lorenzo Baraldi, and Rita Cucchiara. 2020. Meshed-memory transformer for image captioning. In *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition, CVPR 2020, Seattle, WA, USA, June 13-19, 2020*, pages 10575–10584. Computer Vision Foundation / IEEE.

Dina Demner-Fushman, Marc D. Kohli, Marc B. Rosenman, Sonya E. Shooshan, Laritza Rodriguez, Sameer K. Antani, George R. Thoma, and Clement J. McDonald. 2016. Preparing a collection of radiology examinations for distribution and retrieval. *J. Am. Medical Informatics Assoc.*, 23(2):304–310.

Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. 2009. Imagenet: A large-scale hierarchical image database. In *2009 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR 2009), 20-25 June 2009, Miami, Florida, USA*, pages 248–255. IEEE Computer Society.

Michael J. Denkowski and Alon Lavie. 2011. Meteor 1.3: Automatic metric for reliable optimization and evaluation of machine translation systems. In *Proceedings of the Sixth Workshop on Statistical Machine Translation, WMT@EMNLP 2011, Edinburgh, Scotland, UK, July 30-31, 2011*, pages 85–91. Association for Computational Linguistics.

Behnaz Gheflati and Hassan Rivaz. 2021. Vision transformer for classification of breast ultrasound images. *CoRR*, abs/2110.14731.

Tanmay Gupta, Arash Vahdat, Gal Chechik, Xiaodong Yang, Jan Kautz, and Derek Hoiem. 2020. Contrastive learning for weakly supervised phrase grounding. In *Computer Vision - ECCV 2020 - 16th European Conference, Glasgow, UK, August 23-28, 2020, Proceedings, Part III*, volume 12348 of *Lecture Notes in Computer Science*, pages 752–768. Springer.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. 2016. Deep residual learning for image recognition. In *2016 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2016, Las Vegas, NV, USA, June 27-30, 2016*, pages 770–778. IEEE Computer Society.

Geoffrey E. Hinton, Nitish Srivastava, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. 2012. Improving neural networks by preventing co-adaptation of feature detectors. *CoRR*, abs/1207.0580.

Ronghang Hu, Anna Rohrbach, Trevor Darrell, and Kate Saenko. 2019. Language-conditioned graph networks for relational reasoning. In *2019 IEEE/CVF International Conference on Computer Vision, ICCV 2019, Seoul, Korea (South), October 27 - November 2, 2019*, pages 10293–10302. IEEE.

Gao Huang, Zhuang Liu, Laurens van der Maaten, and Kilian Q. Weinberger. 2017. Densely connected convolutional networks. In *2017 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2017, Honolulu, HI, USA, July 21-26, 2017*. IEEE Computer Society.

Lun Huang, Wenmin Wang, Jie Chen, and Xiaoyong Wei. 2019. Attention on attention for image captioning. In *2019 IEEE/CVF International Conference on Computer Vision, ICCV 2019, Seoul, Korea (South), October 27 - November 2, 2019*, pages 4633–4642. IEEE.

Po-Yao Huang, Junjie Hu, Xiaojun Chang, and Alexander G. Hauptmann. 2020. Unsupervised multimodal neural machine translation with pseudo visual pivoting. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, ACL 2020, Online, July 5-10, 2020*, pages 8226–8237. Association for Computational Linguistics.

Sergey Ioffe and Christian Szegedy. 2015. Batch normalization: Accelerating deep network training by reducing internal covariate shift. In *Proceedings of the 32nd International Conference on Machine Learning, ICML 2015, Lille, France, 6-11 July 2015*, volume 37 of *JMLR Workshop and Conference Proceedings*, pages 448–456. JMLR.org.

Jiayi Ji, Yunpeng Luo, Xiaoshuai Sun, Fuhai Chen, Gen Luo, Yongjian Wu, Yue Gao, and Rongrong Ji. 2021. Improving image captioning by leveraging intra- and inter-layer global representation in transformer network. In *Thirty-Fifth AAAI Conference on Artificial Intelligence, AAAI 2021, Thirty-Third Conference on Innovative Applications of Artificial Intelligence, IAAI 2021, The Eleventh Symposium on Educational*

Advances in Artificial Intelligence, EAAI 2021, Virtual Event, February 2-9, 2021, pages 1655–1663. AAAI Press.

Baoyu Jing, Zeya Wang, and Eric P. Xing. 2019. Show, describe and conclude: On exploiting the structure information of chest x-ray reports. In *Proceedings of the 57th Conference of the Association for Computational Linguistics, ACL 2019, Florence, Italy, July 28- August 2, 2019, Volume 1: Long Papers*, pages 6570–6580. Association for Computational Linguistics.

Baoyu Jing, Pengtao Xie, and Eric P. Xing. 2018. On the automatic generation of medical imaging reports. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics, ACL 2018, Melbourne, Australia, July 15-20, 2018, Volume 1: Long Papers*, pages 2577–2586. Association for Computational Linguistics.

Eun-Sol Kim, Woo-Young Kang, Kyoung-Woon On, Yu-Jung Heo, and Byoung-Tak Zhang. 2020. Hypergraph attention networks for multimodal learning. In *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition, CVPR 2020, Seattle, WA, USA, June 13-19, 2020*, pages 14569–14578. Computer Vision Foundation / IEEE.

Diederik P. Kingma and Jimmy Ba. 2015. Adam: A method for stochastic optimization. In *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*.

Alex Krizhevsky, Ilya Sutskever, and Geoffrey E. Hinton. 2012. Imagenet classification with deep convolutional neural networks. In *Advances in Neural Information Processing Systems 25: 26th Annual Conference on Neural Information Processing Systems 2012. Proceedings of a meeting held December 3-6, 2012, Lake Tahoe, Nevada, United States*, pages 1106–1114.

Yuan Li, Xiaodan Liang, Zhiting Hu, and Eric P. Xing. 2018. Hybrid retrieval-generation reinforced agent for medical image report generation. In *Advances in Neural Information Processing Systems 31: Annual Conference on Neural Information Processing Systems 2018, NeurIPS 2018, December 3-8, 2018, Montréal, Canada*, pages 1537–1547.

Chin-Yew Lin. 2004. ROUGE: A package for automatic evaluation of summaries. In *Text Summarization Branches Out*, pages 74–81, Barcelona, Spain. Association for Computational Linguistics.

Min Lin, Qiang Chen, and Shuicheng Yan. 2014. Network in network. In *2nd International Conference on Learning Representations, ICLR 2014, Banff, AB, Canada, April 14-16, 2014, Conference Track Proceedings*.

Fenglin Liu, Shen Ge, and Xian Wu. 2021a. Competence-based multimodal curriculum learning for medical report generation. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing, ACL/IJCNLP 2021, (Volume 1: Long Papers), Virtual Event, August 1-6, 2021*, pages 3001–3012. Association for Computational Linguistics.

Fenglin Liu, Xian Wu, Shen Ge, Wei Fan, and Yuexian Zou. 2021b. Exploring and distilling posterior and prior knowledge for radiology report generation. In *IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2021, virtual, June 19-25, 2021*, pages 13753–13762. Computer Vision Foundation / IEEE.

Fenglin Liu, Chenyu You, Xian Wu, Shen Ge, Sheng Wang, and Xu Sun. 2021c. Auto-encoding knowledge graph for unsupervised medical report generation. *CoRR*, abs/2111.04318.

Guanxiong Liu, Tzu-Ming Harry Hsu, Matthew B. A. McDermott, Willie Boag, Wei-Hung Weng, Peter Szolovits, and Marzyeh Ghassemi. 2019. Clinically accurate chest x-ray report generation. In *Proceedings of the Machine Learning for Healthcare Conference, MLHC 2019, 9-10 August 2019, Ann Arbor, Michigan, USA*, volume 106 of *Proceedings of Machine Learning Research*, pages 249–269. PMLR.

Jiasen Lu, Caiming Xiong, Devi Parikh, and Richard Socher. 2017. Knowing when to look: Adaptive attention via a visual sentinel for image captioning. In *2017 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2017, Honolulu, HI, USA, July 21-26, 2017*, pages 3242–3250. IEEE Computer Society.

Yasuhide Miura, Yuhao Zhang, Emily Bao Tsai, Curtis P. Langlotz, and Dan Jurafsky. 2021. Improving factual completeness and consistency of image-to-text radiology report generation. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT 2021, Online, June 6-11, 2021*, pages 5288–5304. Association for Computational Linguistics.

Hoang T. N. Nguyen, Dong Nie, Taivanbat Badamdorj, Yujie Liu, Yingying Zhu, Jason Truong, and Li Cheng. 2021. Automated generation of accurate & fluent medical x-ray reports. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing, EMNLP 2021, Virtual Event / Punta Cana, Dominican Republic, 7-11 November, 2021*, pages 3552–3569. Association for Computational Linguistics.

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. Bleu: a method for automatic evaluation of machine translation. In *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics, July 6-12, 2002, Philadelphia, PA, USA*, pages 311–318. ACL.

Sangjoon Park, Gwanghyun Kim, Yujin Oh, Joon Beom Seo, Sang Min Lee, Jin Hwan Kim, Sungjun Moon, Jae-Kwang Lim, Chang Min Park, and Jong Chul Ye. 2022. AI can evolve without labels: self-evolving vision transformer for chest x-ray diagnosis through knowledge distillation. *CoRR*, abs/2202.06431.

Steven J. Rennie, Etienne Marcheret, Youssef Mroueh, Jerret Ross, and Vaibhava Goel. 2017. Self-critical sequence training for image captioning. In *2017 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2017, Honolulu, HI, USA, July 21-26, 2017*, pages 1179–1195. IEEE Computer Society.

Mert Bülent Sariyildiz, Julien Perez, and Diane Larlus. 2020. Learning visual representations with caption annotations. In *Computer Vision - ECCV 2020 - 16th European Conference, Glasgow, UK, August 23-28, 2020, Proceedings, Part VIII*, volume 12353 of *Lecture Notes in Computer Science*, pages 153–170. Springer.

Zhuchen Shao, Hao Bian, Yang Chen, Yifeng Wang, Jian Zhang, Xiangyang Ji, and Yongbing Zhang. 2021. Transmil: Transformer based correlated multiple instance learning for whole slide image classication. *CoRR*, abs/2106.00908.

Zhan Shi, Hui Liu, and Xiaodan Zhu. 2021. Enhancing descriptive image captioning with natural language inference. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing, ACL/IJCNLP 2021, (Volume 2: Short Papers), Virtual Event, August 1-6, 2021*, pages 269–277. Association for Computational Linguistics.

Hoo-Chang Shin, Kirk Roberts, Le Lu, Dina Demner-Fushman, Jianhua Yao, and Ronald M. Summers. 2016. Learning to read chest x-rays: Recurrent neural cascade model for automated image annotation. In *2016 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2016, Las Vegas, NV, USA, June 27-30, 2016*, pages 2497–2506. IEEE Computer Society.

Karen Simonyan and Andrew Zisserman. 2015. Very deep convolutional networks for large-scale image recognition. In *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*.

Rui Sun, Yihao Li, Tianzhu Zhang, Zhendong Mao, Feng Wu, and Yongdong Zhang. 2021. Lesion-aware transformers for diabetic retinopathy grading. In *IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2021, virtual, June 19-25, 2021*, pages 10938–10947. Computer Vision Foundation / IEEE.

Christian Szegedy, Wei Liu, Yangqing Jia, Pierre Sermanet, Scott E. Reed, Dragomir Anguelov, Dumitru Erhan, Vincent Vanhoucke, and Andrew Rabinovich. 2015. Going deeper with convolutions. In *IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2015, Boston, MA, USA, June 7-12, 2015*, pages 1–9. IEEE Computer Society.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, December 4-9, 2017, Long Beach, CA, USA*, pages 5998–6008.

Oriol Vinyals, Alexander Toshev, Samy Bengio, and Dumitru Erhan. 2015. Show and tell: A neural image caption generator. In *IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2015, Boston, MA, USA, June 7-12, 2015*, pages 3156–3164. IEEE Computer Society.

Song Wang, Liyan Tang, Mingquan Lin, George Shih, Ying Ding, and Yifan Peng. 2022. Prior knowledge enhances radiology report generation. *CoRR*, abs/2201.03761.

Xiaomei Wang, Lin Ma, Yanwei Fu, and Xiangyang Xue. 2021a. Neural symbolic representation learning for image captioning. In *ICMR '21: International Conference on Multimedia Retrieval, Taipei, Taiwan, August 21-24, 2021*, pages 312–321. ACM.

Xiaosong Wang, Yifan Peng, Le Lu, Zhiyong Lu, and Ronald M. Summers. 2018. Tienet: Text-image embedding network for common thorax disease classification and reporting in chest x-rays. In *2018 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2018, Salt Lake City, UT, USA, June 18-22, 2018*, pages 9049–9058. Computer Vision Foundation / IEEE Computer Society.

Zhanyu Wang, Luping Zhou, Lei Wang, and Xiu Li. 2021b. A self-boosting framework for automated radiographic report generation. In *IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2021, virtual, June 19-25, 2021*, pages 2433–2442. Computer Vision Foundation / IEEE.

Kun Yan, Lei Ji, Huaishao Luo, Ming Zhou, Nan Duan, and Shuai Ma. 2021. Control image captioning spatially and temporally. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing, ACL/IJCNLP 2021, (Volume 1: Long Papers), Virtual Event, August 1-6, 2021*, pages 2014–2025. Association for Computational Linguistics.

Yiming Yang and Xin Liu. 1999. A re-examination of text categorization methods. In *SIGIR '99: Proceedings of the 22nd Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, August 15-19, 1999, Berkeley, CA, USA*, pages 42–49. ACM.

Changchang Yin, Buyue Qian, Jishang Wei, Xiaoyu Li, Xianli Zhang, Yang Li, and Qinghua Zheng. 2019. Automatic generation of medical imaging diagnostic report with hierarchical recurrent neural network. In *2019 IEEE International Conference on Data Mining, ICDM 2019, Beijing, China, November 8-11, 2019*, pages 728–737. IEEE.

Jingyi You, Chenlong Hu, Hidetaka Kamigaito, Kotaro Funakoshi, and Manabu Okumura. 2021a. Robust dynamic clustering for temporal networks. In *CIKM '21: The 30th ACM International Conference on Information and Knowledge Management, Virtual Event, Queensland, Australia, November 1 - 5, 2021*, pages 2424–2433. ACM.

Jingyi You, Chenlong Hu, Hidetaka Kamigaito, Hiroya Takamura, and Manabu Okumura. 2021b. Abstractive document summarization with word embedding reconstruction. In *Proceedings of the International Conference on Recent Advances in Natural Language Processing (RANLP 2021), Held Online, 1-3September, 2021*, pages 1586–1596. INCOMA Ltd.

Jingyi You, Dongyuan Li, Hidetaka Kamigaito, Kotaro Funakoshi, and Manabu Okumura. 2022. Joint learning-based heterogeneous graph attention network for timeline summarization. In *Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL 2022, Seattle, WA, United States, July 10-15, 2022*, pages 4091–4104. Association for Computational Linguistics.

Quanzeng You, Zhengyou Zhang, and Jiebo Luo. 2018. End-to-end convolutional semantic embeddings. In *2018 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2018, Salt Lake City, UT, USA, June 18-22, 2018*, pages 5735–5744. Computer Vision Foundation / IEEE Computer Society.

Yixiao Zhang, Xiaosong Wang, Ziyue Xu, Qihang Yu, Alan L. Yuille, and Daguang Xu. 2020. When radiology report generation meets knowledge graph. In *The Thirty-Fourth AAAI Conference on Artificial Intelligence, AAAI 2020, The Thirty-Second Innovative Applications of Artificial Intelligence Conference, IAAI 2020, The Tenth AAAI Symposium on Educational Advances in Artificial Intelligence, EAAI 2020, New York, NY, USA, February 7-12, 2020*, pages 12910–12917. AAAI Press.

Luowei Zhou, Hamid Palangi, Lei Zhang, Houdong Hu, Jason J. Corso, and Jianfeng Gao. 2020. Unified vision-language pre-training for image captioning and VQA. In *The Thirty-Fourth AAAI Conference on Artificial Intelligence, AAAI 2020, The Thirty-Second Innovative Applications of Artificial Intelligence Conference, IAAI 2020, The Tenth AAAI Symposium on Educational Advances in Artificial Intelligence, EAAI 2020, New York, NY, USA, February 7-12, 2020*, pages 13041–13049. AAAI Press.

Automatic Nominalization of Clauses through Textual Entailment

John S. Y. Lee[1], **Ho Hung Lim**[1], **Carol Webster**[2], **Anton Melser**[1]
[1] Department of Linguistics and Translation, City University of Hong Kong
[2] College of Professional and Continuing Education, The Hong Kong Polytechnic University
Hong Kong SAR, China
{jsylee,hhlim3}@cityu.edu.hk, carol.webster@cpce-polyu.edu.hk,
anton.melser@my.cityu.edu.hk

Abstract

Nominalization re-writes a clause as a noun phrase. It requires the transformation of the head verb of the clause into a deverbal noun, and the verb's modifiers into nominal modifiers. Past research has focused on the selection of deverbal nouns, but has paid less attention to the word order and word forms for the nominal modifiers. We propose using a textual entailment model for clause nominalization. Experimental results show that a textual entailment model fine-tuned on this task outperforms a number of unsupervised approaches using language model scores.

Extract clause	Since [*Arabic frequently omits vowels*], the surname is often spelt ...
Nominalize clause	(i) Word-order edits: [frequently **vowels** omits **Arabic**]
	(ii) POS Re-generation: [**frequent** vowels **omission** Arabic]
	(iii) Preposition generation: [frequent vowel omissions **in** Arabic]
Re-write sentence	**Due to** [frequent vowel omissions in Arabic], the surname is often spelt ...

Table 1: Main steps in nominalizing a sentence, focusing on the target clause enclosed in [...].

1 Introduction

Many textbooks on academic writing devote significant attention to nominalization, which lends a more abstract, concise and objective tone to a text (Kamler and Thomson, 2006; Bailey, 2011). Since nominalization requires careful lexical selection and clause restructuring, it demands advanced vocabulary knowledge and grammatical skills, making it challenging even for many human writers. Table 1 outlines the steps in nominalizing a sentence: extraction of target clauses; nominalization of a target clause through word re-ordering and re-generating the POS and prepositions; and the re-writing of the sentence to incorporate the nominalized clauses. This paper focuses on the clause nominalization step.

Past research on clause nominalization has concentrated on replacement of the head verb with a deverbal noun (e.g. 'omits' → 'omission') and resource development to support the task (Meyers et al., 1998; Habash and Dorr, 2003; Saberi and Lee, 2019). Less attention has been paid to the clause restructuring that is required for transforming the verb's modifiers in the clause to nominal arguments, including word reordering (e.g., postposing the subject 'Arabic', Table 1i); POS re-generation of the verb's modifiers ('frequently' → 'frequent', Table 1ii); and preposition generation (Table 1iii).

This paper investigates methods for clause nominalization that optimize the position and POS of the nominal arguments. We focus on clauses headed by a verb with up to three syntactic arguments, including subjects, objects, adjectival phrases or prepositional phrases. The output is a nominalized form of the clause that preserves its original meaning. Experimental results show that a textual entailment model that is fine-tuned on this task can outperform unsupervised approaches based on neural language model scores.

The rest of the paper is organized as follows. After a review of previous work (Section 2), we present our two-step algorithm: candidate generation (Section 3) followed by candidate ranking (Section 4). Finally, we report experimental results (Section 6).

2 Previous Work

Shinyama et al. (2002) explored automatic acquisition of paraphrase templates, which included nominalizations. Lee et al. (2018) evaluated the template-based approach specifically on nominalization, but relied on heuristics for word order, POS and preposition selection. Fujita and Sato (2008)

Proceedings of the 29th International Conference on Computational Linguistics, pages 6002–6006
October 12–17, 2022.

Edit	Rewrite	Example input	Gold nominalization
$\neg$V	$\mathbf{V} \rightarrow \emptyset$	(a) she^{S} **makes**^{V} a decision^{O_1}	her decision
		(b) Aunt Lella^{S} **fell**^{V} ill^{O_1}	Aunt Lella's illness
S$^{\Rightarrow}$	$\mathbf{S}$VO$_1 \rightarrow$ VO$_1\mathbf{S}$	(c) **citizens**^{S} initiate^{V} plebiscites^{O_1}	initiation of plebiscites by **citizens**
		(d) **Mexico**^{S} suffers^{V} high casualties^{O_1}	high casualties for **Mexico**
		(e) **results**^{S} were^{V} inconclusive^{O_1}	inconclusive **results**
	$\mathbf{S}$V $\rightarrow$ V$\mathbf{S}$	(f) **tension**^{S} increases^{V} in the region^{O_1}	an increase in **tension** in the region
O$_1^{\Leftarrow}$	V$\mathbf{O}_1 \rightarrow$ $\mathbf{O}_1$V	(g) transfer^{V} **money**^{O_1} to the hijackers^{O_2}	**money** transfer to the hijackers
		(h) Mubarak^{S} raises^{V} **taxes**^{O_1}	Mubarak's **tax** raise
O$_2^{\Leftarrow}$	VO$_1\mathbf{O}_2 \rightarrow$ $\mathbf{O}_2$VO$_1$	(i) use^{V} animals^{O_1} for **research**^{O_2}	**research** use of animals
		(j) use^{V} the park^{O_1} for **recreation**^{O_2}	**recreational** use of the park

Table 2: Word-order edits on example target clauses towards the generation of the gold nominalization

automatically generated syntactic variants of predicate phrases, using n-gram models and distributional similarity measures to estimate semantic equivalence and syntactic substitutability.

Our task is distinct from semantic role labeling (Lapata, 2002; Padó et al., 2008), since it must take into account both the fluency of the nominalization and its semantic equivalence to the original clause. Another related task, the paraphrasing of nominalizations, can be viewed as the reverse of ours. In an unsupervised approach, Lee et al. (2021) selected the best clausal paraphrase of a nominalization using a language model and a textual entailment model, which has also been applied to other NLP tasks such as question answering (Trivedi et al., 2019), summarization (Kryściński et al., 2020), and relation extraction (Sainz et al., 2021). However, the candidate generation algorithm for our task is significantly different. We also address a wider range of nominalization patterns in our evaluation, and show that a supervised approach can improve performance.

3 Nominalization Generation

As shown in Table 2, the input is a sentence with a target clause consisting of the head verb (V) with up to three syntactic arguments, which may be its subject (S) or other arguments in object position (O$_1$, O$_2$).[1] Candidate nominalizations are generated with the following steps:

3.1 Word-order edits

Table 2 lists all edit operations for re-positioning S and O$_i$ in relation to the verb V, including:

Delete verb ($\neg$V) Nominalizations may omit the support verb or light verb in the clause (Table 2a). The copula or a semantically blanched verb can also be omitted, typically when O$_1$ is an adjective (Table 2b).

Postpose subject (S$^{\Rightarrow}$) The subject can be positioned after the verb to serve as a postnominal modifier of the deverbal noun (Table 2c), or of O$_1$ when V is deleted (Table 2d). It may also head the nominalized clause, typically when O$_1$ is an adjective and V is deleted (Table 2e).

Prepose object (O$_i^{\Leftarrow}$) The O$_1$ (Table 2g-h) or O$_2$ (Table 2i-j) can be moved in front of V to become a prenominal modifier of the deverbal noun.

3.2 POS Re-generation

Head noun generation. The verb V can be substituted with the deverbal noun (e.g. 'omits' $\rightarrow$ 'omission' in Table 1ii) to head the nominalized clause. If V is deleted, an O$_1$ adjective can be substituted with a deadjectival noun (e.g., 'ill' $\rightarrow$ 'illness' in Table 2b). All nouns that are derivationally related to the verb or adjective in WordNet (Fellbaum, 2010) are considered candidates.

Prenominal modifier generation. The most fluent nominalization may require different POS for the prenominal modifier depending on context. It may be an adjectival form of the S (Table 3a), of the preposed O$_i$ (Table 3b), or of the adverb (Table 3c). All adjectives listed in WordNet as its pertainyms or derivationally related forms are considered candidates. The prenominal modifier may also be an s-genitive (Table 3d), a possessive pronoun (Table 3e), or the singular form of the original noun (e.g. 'vowels' $\rightarrow$ 'vowel' in Table 1ii).

[1] The clauses would need to be automatically extracted in actual deployment. We assume accurate extraction of these strings to avoid confounding the experimental results with parsing errors.

Rewrite	Example input	Gold nominalization
Adjective	(a) **Americans** immigrate	**American** immigration
	(b) control the **diet** of the patient	**dietary** control of the patient
	(c) the member leaves **suddenly**	the **sudden** departure of the member
Genitive	(d) the **city** emerges from default	**the city's** emergence from default
Poss. Pron.	(e) **it** experiences a mild climate	**its** mild climate

Table 3: Part-of-speech (POS) edits that re-write a noun in the input as an adjective, s-genitive or possessive pronoun to serve as prenominal modifier in the nominalization

3.3 Preposition and determiner generation

Prepositions and determiners can be inserted in front of postmodifiers (Table 1iii), and a determiner may be inserted at the front of the nominalized clause. We enumerate the permutations of all choices of determiners ('a', 'an', 'the'), and use the masked language model BERT (Devlin et al., 2019) to predict the most likely preposition.[2]

4 Candidate ranking

A pool of candidate nominalizations are generated for each target clause using all permutations described in Section 3. We evaluated the following methods to select the best candidate from the pool:

Language Model (LM) Select the candidate that yields the highest-scoring sentence, according to the log-probability score based on GPT-2 (Salazar et al., 2020).[3]

Majority Same as the above but consider only those candidates of the majority pattern, i.e., retain the original word order SVO_1O_2 and use the s-genitive as the prenominal modifier.

Pre-trained TE Model The premise is the input sentence, and the hypothesis is the sentence re-written using the candidate nominalization (Table 1). The TE model predicts whether the premise implies the hypothesis. We used AllenNLP's pre-trained TE model on SNLI based on RoBERTa (Liu et al., 2019).[4] Similar to the approach proposed by Lee et al. (2021), we identify the two candidates with the highest TE model score, and select the one with the higher LM score.

Fine-tuned TE Model Same as above, except that the pre-trained model was fine-tuned on our dataset (Section 5).[5] For each premise, the gold outputs served as the 'entailment' hypotheses. There were on average 252 candidate nominalizations in the candidate pool. For each non-gold word order (Section 3), we selected the candidate with the highest LM score to serve as a 'neutral' hypothesis. There were on average 7.8 'neutral' hypotheses for each premise.

5 Data

Since our research focus is at the clause level rather than the re-writing of the entire sentence, we targeted sentences that permit straightforward alternation between a clause and a noun phrase through: (1) a change of conjunction, e.g., "although ⟨clause⟩" ↔ "despite ⟨NP⟩"; (2) a verb that can take both clause or noun phrase as argument, e.g., "report that ⟨clause⟩" ↔ "report ⟨NP⟩"; and (3) a clause that can replace a discourse deixis as a noun phrase, e.g., "⟨clause⟩. That ⟨verb⟩ ... " ↔ "⟨NP⟩ ⟨verb⟩ ...".

Our dataset contains 319 unique inputs and 751 nominalizations (Table 4).[6] Among the inputs, 202 were extracted from Wikipedia (Section 5.1) and 117 from an existing dataset (Section 5.2).

5.1 Nominalization annotation

We retrieved sentences with the above three patterns from Wikipedia. For efficient annotation, we collected sentences with at least one clause headed by a verb with a derivationally related noun in WordNet. Four annotators, all university students

[2] Downloaded from https://storage.googleapis.com/allennlp-public-models/bert-masked-lm-2020-10-07.tar.gz

[3] We used the 345M version of GPT-2 from https://github.com/awslabs/mlm-scoring

[4] Downloaded from https://storage.googleapis.com/allennlp-public-models/snli-roberta.2021-03-11.tar.gz. We also tried pre-trained TE models based on DeBERTa (https://huggingface.co/microsoft/deberta-v2-xlarge-mnli) and those provided by Nie et al. (2020) but achieved lower accuracy than our best model.

[5] We used default values for all parameters during fine-tuning.

[6] Accessible at https://github.com/NominalizationParaphrase

Initial word in gold nominalization	Pre. modifier			Total
	N	G	A	
Nominalized verb	n/a	n/a	n/a	221
Subject (S)	70	140	92	302
Object (O_1 or O_2)	123	18	87	228

Table 4: Distribution of different word orders and POS of the prenominal modifier, as noun (N), adjective (A), s-genitive or possessive pronoun (G) in the gold nominalizations in our dataset

who were native speakers of English, composed possible nominalizations, if any, for each sentence. The annotators were asked to favor derivationally related forms over free paraphrases. Only those nominalizations produced by at least two annotators were included in the dataset.

5.2 Conversion from paraphrase dataset

The nominalization-clause pairs in the dataset produced by Lee et al. (2021) are not directly usable, since the clause paraphrases only the nominalization but not the rest of the sentence. We identified the sentences in this dataset with the three patterns described above, and then replaced the nominalization with their gold clause according to the alternation templates above to produce an input sentence for our task.[7]

Since the nominalization inputs in this dataset all have one prenominal modifier and one prepositional phrase, they would lead to an imbalanced output in our dataset. We asked a native speaker of English, a PhD candidate in linguistics, to enumerate other acceptable nominalizations, which were then reviewed by a professor of linguistics who was a near-native speaker of English.

6 Experiments

6.1 Set-up

All models were evaluated on the full dataset (Section 5), with 10-fold cross-validation used for the Fine-tuned TE Model. In the **Gold** setting, the gold forms of the prenominal modifier and deverbal noun were always included as one of the candidates along with other candidates retrieved from WordNet. The **Auto** setting was fully automatic.

We report three evaluation metrics in ascending strictness. For **word order accuracy**, the output is considered correct if it matches the gold word order (Section 3.1). **POS accuracy**, in addition, requires all prenominal modifiers to have the gold POS (Section 3.2). **Nominalization accuracy** requires the lemmatized form of the output to match exactly with the gold, except determiners.

[7] e.g., the sentence "Due to *frequent vowel omissions in Arabic* ..." is re-written as "Since *Arabic frequently omits vowels* ..." to serve as the input in our dataset.

Setting	Metric → ↓ Model	WO acc.	POS acc.	Nom acc.
Gold	Majority	0.467	0.429	0.376
	LM	0.586	0.536	0.476
	Pre-trained TE	0.639	0.524	0.439
	Fine-tuned TE	**0.812**	**0.743**	**0.630**
Auto	Majority	0.429	0.395	0.323
	LM	0.514	0.476	0.389
	Pre-trained TE	0.586	0.470	0.326
	Fine-tuned TE	**0.724**	**0.655**	**0.511**

Table 5: Model performance in the Gold and Auto settings in terms of word order (WO) accuracy, POS accuracy and nominalization accuracy

6.2 Results

As expected, system performance was higher in the Gold setting than Auto (Table 5). In both settings, the use of LM scores yielded improvement over the Majority baseline.

Pre-trained model. The Pre-trained TE Model slightly outperformed the LM on the word-order metric, suggesting its ability in determining semantic equivalence of the nominalization with the input sentence, a factor that is not considered by the LM. However, it is less sensitive to the choice of POS, prepositions and determiners to optimize fluency, as reflected by its lower POS and nominalization accuracy.

Fine-tuned model. Despite the limited size of our dataset, fine-tuning resulted in the strongest performance. In both settings and in terms of all three metrics, it produced statistically significant improvement over all other models.[8] The improvement over the pre-trained version suggests that the semantic nuances that distinguish between the nominalization candidates are more subtle than the premises and hypotheses in general domains.

The performance was slightly lower if the TE model was used in the reverse direction, i.e., with the input sentence as hypothesis and its nominalized form as premise. This might be attributable to

[8] $p < 0.05$ according to McNemar's Test in both settings and all three metrics

the fact that the nominalized form is usually less specific in terms of tense and aspect.

7 Conclusion

This paper has reported the first quantitative evaluation on automatic clause nominalization and contributed the first dataset for this task. We have shown that a fine-tuned textual entailment model, followed with reranking with a language model, outperforms a number of competitive unsupervised approaches. In future work, we plan to extend our algorithm to determine whether a target clause can or should be nominalized, and to explore a richer variety of nominalization types.

Acknowledgements

We gratefully acknowledge support from a CityU Strategic Research Grant (project #7005709).

References

Stephen Bailey. 2011. *Academic Writing: A Handbook for International Students*. Routledge, New York, NY.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pretraining of Deep Bidirectional Transformers for Language Understanding. In *Proc. North American Chapter of the Association for Computational Linguistics - Human Language Technologies (NAACL-HLT)*.

Christiane Fellbaum. 2010. Wordnet. In *Theory and applications of ontology: computer applications*, pages 231–243, Springer, Dordrecht.

Atsushi Fujita and Satoshi Sato. 2008. A Probabilistic Model for Measuring Grammaticality and Similarity of Automatically Generated Paraphrases of Predicate Phrases. In *Proc. 22nd International Conference on Computational Linguistics (COLING)*.

Nizar Habash and Bonnie Dorr. 2003. A Categorial Variation Database for English. In *Proc. North American Chapter of the Association for Computational Linguistics - Human Language Technologies (NAACL)*.

Barbara Kamler and Pat Thomson. 2006. *Helping doctoral students write: Pedagogies for supervision*. Routledge, London.

Wojciech Kryściński, Bryan McCann, Caiming Xiong, and Richard Socher. 2020. Evaluating the Factual Consistency of Abstractive Text Summarization. In *Proc. Conference on Empirical Methods in Natural Language Processing (EMNLP)*, page 9332–9346.

Maria Lapata. 2002. The Disambiguation of Nominalizations. *Computational Linguistics*, 28(3):357–388.

John S. Y. Lee, Ho Hung Lim, and Carol Webster. 2021. Paraphrasing Compound Nominalizations. In *Proc. Conference on Empirical Methods in Natural Language Processing (EMNLP)*.

John S. Y. Lee, Dariush Saberi, Marvin Lam, and Jonathan Webster. 2018. Assisted Nominalization for Academic English Writing. In *Proc. Workshop on Intelligent Interactive Systems and Language Generation (2ISNLG)*, pages 26–30.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Roberta: A robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692*.

Adam Meyers, Catherine Macleod, Roman Yangarber, Ralph Grishman, Leslie Barrett, and Ruth Reeves. 1998. Using NOMLEX to Produce Nominalization Patterns for Information Extraction. In *Proc. Computational Treatment of Nominals*.

Yixin Nie, Adina Williams, Emily Dinan, Mohit Bansal, Jason Weston, and Douwe Kiela. 2020. Adversarial NLI: A New Benchmark for Natural Language Understanding. In *Proc. 58th Annual Meeting of the Association for Computational Linguistics (ACL)*.

Sebastian Padó, Marco Pennacchiotti, and Caroline Sporleder. 2008. Semantic role assignment for event nominalisations by leveraging verbal data. In *Proc. 22nd International Conference on Computational Linguistics (COLING)*.

Dariush Saberi and John Lee. 2019. Noun Generation for Nominalization in Academic Writing. In *Proc. Workshop on Computational Creativity in Natural Language Generation (CCNLG)*.

Oscar Sainz, Oier Lopez de Lacalle, Gorka Labaka, Ander Barrena, and Eneko Agirre. 2021. Label Verbalization and Entailment for Effective Zero- and Few-Shot Relation Extraction. In *Proc. Conference on Empirical Methods in Natural Language Processing (EMNLP)*, page 1199–1212.

Julian Salazar, Davis Liang, Toan Q. Nguyen, and Katrin Kirchhoff. 2020. Masked Language Model Scoring. In *Proc. Annual Meeting of the Association for Computational Linguistics (ACL)*.

Yusuke Shinyama, Satoshi Sekine, and Kiyoshi Sudo. 2002. Automatic Paraphrase Acquisition from News Articles. In *Proc. Second international conference on Human Language Technology Research (HLT)*.

Harsh Trivedi, Heeyoung Kwon, Tushar Khot, Ashish Sabharwal, and Niranjan Balasubramanian. 2019. Repurposing Entailment for Multi-Hop Question Answering Tasks. In *Proc. NAACL-HLT*.

Benchmarking Compositionality with Formal Languages

Josef Valvoda **Naomi Saphra** **Jonathan Rawski**
Adina Williams **Ryan Cotterell**
University of Cambridge New York University
San José State University FAIR ETH Zürich
jv406@cam.ac.uk nsaphra@nyu.edu jon.rawski@sjsu.edu
adinawilliams@fb.com ryan.cotterell@inf.ethz.ch

Abstract

Recombining known primitive concepts into larger novel combinations is a quintessentially human cognitive capability. Whether large neural models in NLP can acquire this ability while learning from data is an open question. In this paper, we investigate this problem from the perspective of formal languages. We use deterministic finite-state transducers to make an unbounded number of datasets with controllable properties governing compositionality. By randomly sampling over many transducers, we explore which of their properties contribute to learnability of a compositional relation by a neural network. We find that the models either learn the relations completely or not at all. The key is transition coverage, setting a soft learnability limit at 400 examples per transition.

https://github.com/valvoda/neuralTransducer

1 Introduction

Compositionality is a hallmark of human language (Montague, 1970; Partee, 1995; Fodor, 1998). It is arguably a requirement for any model to count as a model of language, or to achieve human-like natural language understanding. Compositionality seems to be such a deep property of language that speakers draw conclusions about the overall meaning of sentences even when the meanings of individual words are not known. For instance, English speakers reading the Jabberwocky (Carroll, 1871) comprehend that the noun phrase *slithy toves* is built from the composition of the adjective *slithy* with the plural noun *toves*, despite lacking a clear understanding of what *slithy* or *toves*—let alone their composition—could mean. In cognitive science, whether neural networks can learn to combine a limited number of primitives (in the case of language, word or morphemes) to describe a complex environment has been debated for over 30 years (Fodor and Pylyshyn, 1988; Marcus, 1998).

In recent work, researchers have explored the inherent limitations of neural models to exhibit compositionality by analyzing sequence-to-sequence model performance on small, controlled datasets (Lake and Baroni, 2018; Hewitt et al., 2020; Hupkes et al., 2020a; White and Cotterell, 2021; Dankers et al., 2022; White and Cotterell, 2022). However, the conclusions of these studies are often murky. For instance, Lake and Baroni (2018) cast doubt on neural models' ability to do compositional generalization using their toy SCAN dataset, but shortly thereafter, Bastings et al. (2018) demonstrated that an out-of-the-box sequence-to-sequence model could indeed fully master the task.

Instead of hand-crafting small challenge datasets, we propose to test for compositionality by randomly sampling from a whole class of string-to-string functions. In doing so, we draw on two linguistic traditions. On the one hand, we follow Montague's assertion that no important theoretical difference exists between natural and artificial languages (Montague, 1970). Following this logic, the question of whether neural networks compositionally process human language is fundamentally equivalent to asking whether they compositionally process artificial languages. On the other hand, we draw lessons from the field of grammatical inference (de la Higuera, 2010; Rawski and Heinz, 2019), and evaluate neural sequence-to-sequence models on many automatically generated artificial languages sampled from particular classes of functions—as is standard practice at grammatical inference competitions (Balle et al., 2017).

In this paper, we study the class of string functions encoded by subsequential finite-state transducers (SFSTs), a restricted class of general finite-state transducers (Mohri, 1997). We sample arbitrary SFSTs to generate many different string-to-string datasets and evaluate the behavior of neural sequence-to-sequence models when learning them. By controlling the formal properties

Proceedings of the 29th International Conference on Computational Linguistics, pages 6007–6018
October 12–17, 2022.

of the SFSTs we sample from, we are able to make precise statements about the learnability of systematic phenomena.

Empirically, we find that neural sequence-to-sequence models are, in many cases, capable of perfectly learning SFSTs from finite data. Moreover, we observe an interesting tendency for neural models to either generalize correctly or to fail outright—with little middle ground. Our analysis reveals a possible explanation for this—generalization seems to be possible when the training data has sufficient coverage, i.e., when every transition in a given transducer is crossed in a minimum number of training examples (≈ 400 in our experiments).

We then turn to analyze a popular hand-crafted dataset, SCAN, through the lens of an SFST that encodes it. We find that SCAN is peculiar in that it seems to serve as a counterexample to our transition coverage finding. This suggests that there is a more nuanced story to tell: We predict the learnability of a language based on a notion of complexity native to subregular languages, but it may be that a more consistently predictive complexity metric would come from a higher point on the Chomsky hierarchy. Future work might seek to limit the number of outlier languages like SCAN in order to identify a notion of complexity that is native to the architecture of the model itself. Such a notion of complexity would identify the level of abstraction that best reflects the representations learned by the model. A fruitful avenue towards this goal might lie in exploring more complex formalisms.

2 Finite-State Transducers

This section provides a short technical overview of finite-state transducers and motivates our choice to learn this class of relations.

2.1 Why Learn Finite-State Transductions?

Our study focuses on learning a particular kind of transduction. Specifically, we focus on restricted classes of regular relations, which are those relations computable by finite-state transducers. We believe this is a natural starting point since this class of formal languages is mathematically well-studied, has provable learning guarantees, and has a long use history in linguistics and NLP (Mohri, 1997).

Finite-state transducers also encompass most previous work on compositionality: many datasets, e.g., SCAN (Lake and Baroni, 2018) and gSCAN (Ruis et al., 2020), describe *finite* string relations and are, therefore, finite-state by definition. These handcrafted datasets have many advantages, like easy interpretability and domain specificity since they directly encode particular relevant relationships like movement over a grid or specific linguistic phenomena. However, this realism pays the price of diminished robustness of any findings over such datasets (Rogers and Pullum, 2011), see White and Cotterell (2021) for more discussion of this point. By removing the ability to simply adjust properties of the underlying function class, and the transducers which compute it, one loses the possibility to experiment more robustly over a function type, rather than just one token instantiation of it.

Rather than manually designing individual datasets ourselves, we generate unboundedly many new datasets via randomly sampled SFSTs. This offers a principled view of the problem of learning artificial languages by simply varying properties of the class of transducers that generate them. Furthermore, as we will see in §6, one may view existing compositionality tasks as learning a specific SFST. We contend this view enables a deeper understanding of modeling results. Both specific artificial languages, such as the compositionality datasets mentioned above, and those randomly sampled from a particular function class such as the work presented in this paper, are worth studying. However, our approach has been missing from the compositionality discourse.

2.2 Basic Theory

Now we will overview the basic elements of finite-state theory that will be necessary for the rest of the paper; we start with some definitions.

Definition 1. *A **finite-state automaton** (FSA) is a 5-tuple $\mathcal{A} = \langle \Sigma, Q, q_0, F, \delta \rangle$ where*

- *Σ is an input alphabet whose elements are denoted σ;*
- *Q is a finite set of states whose elements are denoted q;*
- *$q_0 \in Q$ is the unique start state;*
- *$F \subseteq Q$ is the set of final states;*
- *$\delta : Q \times \Sigma \cup \{\varepsilon\} \to Q$ is the transition relation and ε is an empty string.*

*We denote transitions, i.e., when $q' \in \delta(q, \sigma)$, with the more suggestive notation $q \xrightarrow{\sigma} q'$. We say that the automaton $\mathcal{A}$ **accepts** a string $\boldsymbol{\sigma} \in \Sigma^*$ iff there*

exists a path[1] *through the automaton states* $q_0 \xrightarrow{\sigma_1} q_1 \xrightarrow{\sigma_2} q_2 \cdots \xrightarrow{\sigma_N} q_N$ *where* q_0 *is the initial state and* $q_N \in F$ *and* $\sigma_1 \cdots \sigma_N = \boldsymbol{\sigma}$*. In our notation* σ_n *can be the empty string* ε*. We call* $\boldsymbol{\sigma}$ *the* ***yield*** *of the path* $q_0 \xrightarrow{\sigma_1} q_1 \xrightarrow{\sigma_2} q_2 \cdots \xrightarrow{\sigma_N} q_N$*.*

Furthermore, we call an automaton **complete** when one may transition from every state to every symbol, i.e., $\delta(q, \sigma)$ is defined for all $q \in Q$ and $\sigma \in \Sigma$. And, we say an automaton is **deterministic**, if, given a state q and an alphabet symbol $\sigma \in \Sigma$, there is at most one transition for σ from q, i.e., $|\delta(q, \sigma)| \leq 1$. We have that $|\delta(q, \sigma)| = 1$ for all $q \in Q$ and $\sigma \in \Sigma$ if the automaton is *both* complete and deterministic. In our examples below we denote the **final state** as a circle with a double border.

Example 1. *Below we exhibit a complete deterministic finite-state automaton.*

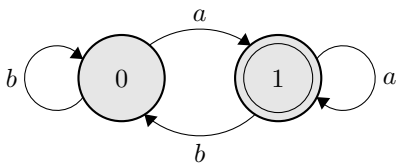

The above finite-state automaton accepts the language $\{(b^i a^j)^n \mid i, j, n \in \mathbb{Z}_+\}$*.*

Definition 2. *A* ***finite-state transducer*** *(FST) is a 6-tuple* $\mathcal{T} = \langle \Sigma, \Gamma, Q, q_0, F, \delta \rangle$*:*

- Σ *is an input alphabet whose elements are denoted* σ*;*
- Γ *is an output alphabet whose elements are denoted* γ*;*
- Q *is a finite set of states whose elements are denoted* q*;*
- $q_0 \in Q$ *is the unique start state;*
- $F \subseteq Q$ *is the set of final states;*
- $\delta : Q \times \Sigma \cup \{\varepsilon\} \to Q \times \Gamma \cup \{\varepsilon\}$ *is the transition relation.*

We denote transitions, i.e., when $(q', \gamma) \in \delta(q, \sigma)$*, with the more suggestive notation* $q \xrightarrow{\sigma/\gamma} q'$*. We say that the transducer* $\mathcal{T}$ ***transduces*** *a string* $\boldsymbol{\sigma} \in \Sigma^*$ *to a string* $\boldsymbol{\gamma} \in \Gamma^*$ *iff there exists a path* $q_0 \xrightarrow{\sigma_1/\gamma_1} q_1 \xrightarrow{\sigma_2/\gamma_2} q_2 \cdots \xrightarrow{\sigma_N/\gamma_N} q_N$ *where* q_0 *is the initial state and* $q_N \in F$*,* $\sigma_1 \cdots \sigma_N = \boldsymbol{\sigma}$ *and* $\gamma_1 \cdots \gamma_N = \boldsymbol{\gamma}$*. In our notation either* σ_n *or* γ_n *can be the empty string* ε*, from which it follows that the length* $|\boldsymbol{\sigma}|$ *must not equal* $|\boldsymbol{\gamma}|$*. We call* $\boldsymbol{\sigma}$ *the* ***input yield*** *and* $\boldsymbol{\gamma}$ *the* ***output yield****, respectively, of the path* $q_0 \xrightarrow{\sigma_1/\gamma_1} q_1 \xrightarrow{\sigma_2/\gamma_2} q_2 \cdots \xrightarrow{\sigma_N/\gamma_N} q_N$*.*

As in the case of an FSA, we say an FST is **complete** if δ is defined for all states and all symbols, i.e., $|\delta(q, \sigma)| > 0$ for $q \in Q$ and $\sigma \in \Sigma$. Furthermore, we say an FST is **subsequential** if it is deterministic with respect to the input, i.e., if $|\delta(q, \sigma)| \leq 1$ for all $q \in Q$ and $\sigma \in \Sigma$ and the FST does not have transitions of the form $q \xrightarrow{\varepsilon/\gamma} q'$. By construction, subsequential transducers (SFSTs) are **functional**, i.e., the string-to-string relations they encode are functions rather than relations. Indeed, it is this functionality that makes them a useful tool for the analysis of neural sequence-to-sequence models. The class of **subsequential functions** are those describable with SFSTs.[2] They are a subclass of the regular relations, but a superclass of the finite relations.

Example 2. *Below is an example of a nondeterministic finite-state transducer:*

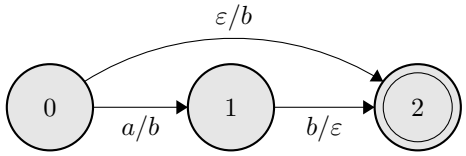

The above transducer only has two paths $q_0 \xrightarrow{a/b} q_1 \xrightarrow{b/\varepsilon} q_2$ *and* $q_0 \xrightarrow{\varepsilon/b} q_2$*. The first transduces* $ab \mapsto b$ *and the second* $\varepsilon \mapsto b$*.*

Example 3. *Below is an example of a complete subsequential transducer over the input alphabet* $\Sigma = \{a, b\}$ *and output alphabet* $\Gamma = \{a, b\}$*.*

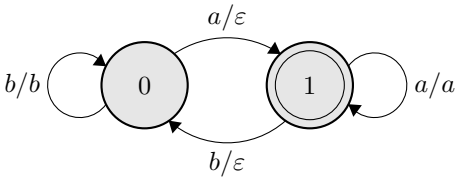

Note the absence of ε *on the input side; however, we do have* ε *on the output side.*

3 Compositionality Formalized

It is widely held that compositionality is a cornerstone of human language. However, language researchers use the term compositionality to refer to a variety of different concepts (Hupkes et al., 2020b). Here, we discuss the specific definition we employ throughout the paper and give a theoretical justification for the use of finite-state transducers as an instantiation of that definition.

[1] A path is a sequence of transitions.

[2] Note there are other algebraic and logical characterisations (Oncina et al., 1993; Bhaskar et al., 2020).

3.1 Montague's Compositionality

Montague famously gave a mathematical definition of what it means for a language to be compositional (Montague, 1970)—specifically, he proposed that a relation, e.g., the mapping from syntax to semantics, be called compositional if and only if it is a homomorphism, i.e., if it is a structure-preserving map between an input and an output algebra (Andreas, 2019). To make this notion more formal, we have to be specific about what structure we hope our function preserves. In this paper, we will exclusively focus on monoidal structure. We emphasize, however, that the notion of a homomorphism is not restricted to monoids.

Definition 3. *A **monoid** is a set A equipped with a binary operation $\bullet$ such that*

- *For $f, g \in A$, $f \bullet g \in A$ (**closure**);*
- *For $f, g, h \in A$, $f \bullet (g \bullet h) = (f \bullet g) \bullet h$ (**associativity**);*
- *There exists a unique element e such that for every $g \in A$, we have that $g \bullet e = e \bullet g = g$ (**identity**).*

Definition 4. *A **free monoid** over strings is the structure $(\Sigma^*, \circ)$ where Σ^* is the Kleene closure of the alphabet Σ and $\circ$ is string concatenation.[3] The empty string ε is the identity element as $\varepsilon \circ \boldsymbol{\sigma} = \boldsymbol{\sigma} \circ \varepsilon = \boldsymbol{\sigma}$ for any $\boldsymbol{\sigma} \in \Sigma^*$.*

Definition 5. *Let $(A, \bullet_A)$ and $(B, \bullet_B)$ be two monoids with unique identity elements e_A and e_B. We call a function $f : A \to B$ a **homomorphism** if it obeys the following two properties:*

- $f(e_A) = e_B$;
- $f(x \bullet_A y) = f(x) \bullet_B f(y), \quad \forall x, y \in A$

We call a homomorphism between two free monoids a **string homomorphism**. As it turns out, there is a precise connection between string homomorphisms and finite-state theory.

Proposition 1. *Let Σ and Γ be two alphabets. The function f is a string homomorphism between $(\Sigma^*, \circ)$ and $(\Gamma^*, \circ)$ iff it is a non-empty complete subsequential finite-state transducer with one state.*

Proof. See App. A. ■

Proposition 1 starts to shed light on the connection between Montague's notion of compositionality and finite-state transducers. However, this connection is quite weak because multi-state transducers are not covered. We remedy this disparity in the subsequent section.

3.2 Transducers as Homomorphisms

Now we offer a more formal treatment of the exact sense in which SFSTs may be considered homomorphism and thus fall under Montague's definition of compositionality. As shown by Proposition 1, in general, SFSTs do *not* encode string homomorphisms. Indeed, it is straightforward to find a counterexample that hammers this point home.

Example 4. *Below we exhibit a two-state subsequential finite-state transducer that is* not *a string homomorphism.*

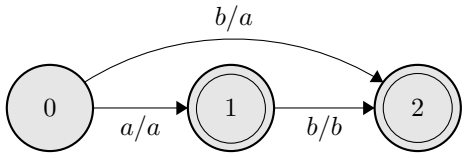

In the above example, we have $ab \mapsto ab$, but also $a \mapsto a$ and $b \mapsto a$. Thus, it is not a homomorphism.

Example 4 is dissatisfying; it contradicts the intuition that an SFST encodes some notion of Montague-esque compositionality. Luckily, as it turns out, we can find a precise sense in which an SFST is indeed a homomorphism. The idea is to lift the free monoid into a matrix. Given a complete SFSA $\mathcal{A} = \langle \Sigma, Q, q_0, F, \delta \rangle$ over $|Q| = N$ states, for every symbol $\sigma \in \Sigma$ define an $N \times N$ **symbol transition matrix** $\boldsymbol{M}^\sigma$ whose entries are

$$M^\sigma_{ik} \stackrel{\text{def}}{=} \begin{cases} \sigma, & \textbf{if } q_i \xrightarrow{\sigma} q_k \in \delta \\ \mathbf{0}, & \textbf{otherwise} \end{cases} \tag{1}$$

where $\mathbf{0}$ is a distinguished symbol, which is not in Σ, called the **zero string**. The zero string is an **anhilitator**, i.e., it has the property that $\mathbf{0} \circ \sigma = \sigma \circ \mathbf{0} = \mathbf{0}$ for any $\sigma \in \Sigma$.

Importantly, because $\mathcal{A}$ is complete and deterministic, there is exactly one non-$\mathbf{0}$ entry in every row of $\boldsymbol{M}^\sigma$. Now, we define an operation $\otimes$ between two such matrices. Since the automaton is complete and deterministic, we know there exists a unique j' such that $M^\sigma_{ij'} \neq \mathbf{0}$ and, by the same argument, there exists a unique k' such that $M^\sigma_{j'k'} \neq \mathbf{0}$. Then, in terms of j' and k', we define

$$(\boldsymbol{M}^\sigma \otimes \boldsymbol{M}^{\sigma'})_{ik} \stackrel{\text{def}}{=} \begin{cases} M^\sigma_{ij'} \circ M^\sigma_{j'k'}, & \textbf{if } k = k' \\ \mathbf{0}, & \textbf{otherwise} \end{cases} \tag{2}$$

[3] When clear from context, we write $\sigma \circ \sigma'$ as $\sigma\sigma'$.

Clearly, $\boldsymbol{M}^{\sigma} \otimes \boldsymbol{M}^{\sigma'}$, for any $\sigma, \sigma' \in \Sigma$, enforces that the resulting product still has the property that there is exactly one element of every row that is not equal to $\mathbf{0}$.[4] With $\boldsymbol{M}^{\boldsymbol{\sigma}}$, where $\boldsymbol{\sigma} = \sigma_1 \cdots \sigma_K \in \Sigma^*$ is a string of length K, we denote the product of matrices $\boldsymbol{M}^{\sigma_1} \otimes \cdots \otimes \boldsymbol{M}^{\sigma_K}$.

Proposition 2. *Let $\mathcal{A} = \langle \Sigma, Q, q_0, F, \delta \rangle$ be a complete deterministic finite-state automaton. Let $\mathcal{M} \stackrel{\text{def}}{=} \left\{ \boldsymbol{M}^{\sigma} \mid \sigma \in \Sigma \right\}$ be the set of $\mathcal{A}$'s symbol transition matrices. Then $(\mathcal{M}^*, \otimes)$ with $\otimes$ defined in Eq. (2) is a **transition monoid** with $\boldsymbol{E}$ as a distinguished identity element.*

Proof. See App. A. ■

Now we are in a position to discuss the precise sense in which subsequential finite-state transducers are homomorphisms. In the case of a finite-state transducer $\mathcal{T} = \langle \Sigma, \Gamma, Q, q_0, F, \delta \rangle$, there are two symbol transition matrices. The *input* symbol transition matrix is defined analogously to that of a finite-state acceptor. However, the *output* symbol transition matrix is defined slightly differently:

$$M^{\sigma}_{ik} \stackrel{\text{def}}{=} \begin{cases} \gamma, & \textbf{if } q_i \xrightarrow{\sigma/\gamma} q_k \in \delta \\ \mathbf{0}, & \textbf{otherwise} \end{cases} \tag{3}$$

Indeed, for any product of K output symbol transition matrices $\boldsymbol{M}^{\boldsymbol{\sigma}} = \boldsymbol{M}^{\sigma_1} \otimes \cdots \otimes \boldsymbol{M}^{\sigma_K}$, the entries of $\boldsymbol{\sigma}$ have a clear interpretation. Let $\boldsymbol{\sigma} = \sigma_1 \cdots \sigma_K$ and suppose $\boldsymbol{M}^{\boldsymbol{\sigma}}_{ik} = \boldsymbol{\gamma} \neq \mathbf{0}$. Then, we know that if we start in state q_i and read in input string $\boldsymbol{\sigma}$ we end up in state q_k and output string $\boldsymbol{\gamma}$. There are exactly $|Q|$ non-zero entries in $\boldsymbol{M}^{\boldsymbol{\sigma}}$. Thus, for any given finite-state transducer there is both an input and output transition monoid, which we denote as $(\mathcal{M}^*_{\Sigma}, \otimes)$ and $(\mathcal{M}^*_{\Gamma}, \otimes)$, respectively. Then, the intuition is that SFSTs constitute a homomorphism over the closure of the symbol transition matrices. We state the more formal result below.

Proposition 3. *Let $\mathcal{T} = \langle \Sigma, \Gamma, Q, q_0, F, \delta \rangle$ be a complete subsequential finite-state transducer. Let $(\mathcal{M}^*_{\Sigma}, \otimes)$ be the transition monoid associated with the input alphabet Σ and let $(\mathcal{M}^*_{\Gamma}, \otimes)$ be the transition monoid associated with the output alphabet Γ. Then there exists a homomorphism $f : \mathcal{M}^*_{\Sigma} \to \mathcal{M}^*_{\Gamma}$.*

Proof. See App. A. ■

[4] This property is reminiscent of the "tails" of a subsequential function (Oncina et al., 1993)

At a higher level, Proposition 3 simply fixes the bug present in Example 4 by incorporating the state into the values. Moreover, Proposition 3 is a clear generalization of Proposition 1 in that if we have a single-state SFST, we have 1×1 matrices that may be viewed as single symbols and the operation $\otimes$ defined in Eq. (2) reduces to string concatenation.

Briefly back to Montague. To put the above theory in context, we have now shown by Proposition 3 that arbitrary complete SFSTs are compositional in the sense of Montague. This gives us a theoretical underpinning to use learning from finite data SFSTs as a benchmark for compositionality, and motivates our experiments and analysis in the coming sections.

4 Experimental Methods

Next we introduce the methodology for generating our datasets as well as the neural and symbolic models employed in the empirical portion of our paper.

4.1 Generating Random SFSTs

We generate random SFSTs using the following stochastic process. We first generate random unlabeled directed graphs that correspond to the symbol-specific transition matrices. Given a set of states Q (let $N = |Q|$) and an input alphabet Σ, we sample a matrix $\boldsymbol{B}^{\sigma} \in \mathbb{B}^{N \times N}$ for every $\sigma \in \Sigma$ where $\mathbb{B} = \{0, 1\}$. During sampling, we enforce the constraint that there be *at most* one non-zero entry in every row vector $\mathbf{b}^{\sigma}_i$. This constraint ensures that the resulting SFST is subsequential by construction. To achieve this, we sample from $\{1, \ldots, N\}$ uniformly at random for each of the N rows to determine the location of the non-zero entry in each of the N rows. In terms of the SFST, if the entry $b^{\sigma}_{ik} = 1$, then our generated SFST has a transition from state $q_i \xrightarrow{\sigma} q_k$ with input symbol σ. Then, for every transition $q_i \xrightarrow{\sigma} q_k$ in our generated SFST, we sample its output symbol γ from a uniform distribution over the output alphabet Γ. This results in a transition $q_i \xrightarrow{\sigma/\gamma} q_k$. Finally, to get a canonical representation for particular SFSTs, we perform finite-state minimization (Choffrut, 2003). Minimization also ensures that our sampled SFSTs are canonicalized SFSTs which allows us to check duplicates and ensure the same SFST is not sampled more than once.

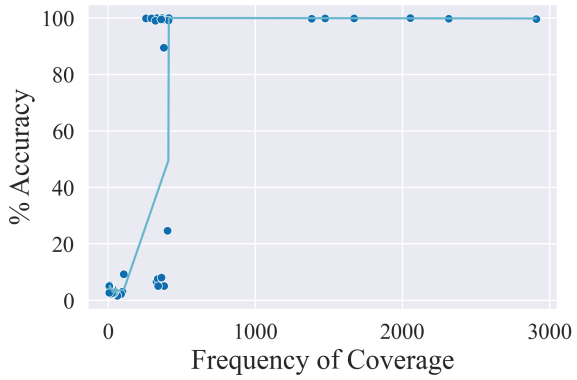

Figure 1: Accuracy versus frequency of transition coverage, with an inflection point at 400.

4.2 Generating the Datasets

Next we discuss the creation of the datasets we use to investigate the learnability of our sampled SFSTs. We start by sampling 1000 unique SFSTs using the process described in §4.1. All SFSTs have an input alphabet Σ of 10 symbols and output alphabet of 30 symbols. We consider SFSTs with states numbering from 10 to 100 in increments of 10, and we sample 100 unique SFSTs for each number. Additionally, we sample extra SFSTs with 21 to 39 states because it was at this point that we empirically observed performance drop-off during a preliminary investigation. In total, our experiments use 2800 unique SFSTs.

Sampling Input–output Pairs. To generate the input–output pairs to train and evaluate a neural model, we perform a random walk through the SFST where we select a transition (including the option to halt if we are in a final state) uniformly at random. Following Lake and Baroni (2018), the maximum length of an input string is capped at 50, i.e., we reject walks with more than 50 steps during sampling. Using this process, we collect $20,000$ unique input–output pairs from each SFST. We considered larger dataset sizes ($40,000$) for the SFSTs with 29 to 39 states since we observe an accuracy drop-off in this region. All datasets are randomly split 80–20 into training and test sets.

4.3 Neural Sequence-to-Sequence Models

Our experiments make use of Wu and Cotterell's (2019) open-source neural transduction library.[5] Our experiments consider an LSTM encoder–decoder with attention in the style of Bahdanau et al. (2015). We use the following hyperparameters: 200-dimensional hidden states in the encoder and decoder, each of which have 2 layers, maximum gradient clipping normalization of 5, dropout set to 0.5, and a batch size of 64. Additionally, our alphabet tokens are embedded as 100-dimensional vectors. We train the model for $100,000$ epochs using the Adam optimizer (Kingma and Ba, 2015) with the default learning rate of 0.001. To determine the effect of model capacity, we also consider a neural sequence-to-sequence model with 300-dimensional hidden states and all other hyperparameters kept the same.

[5]The code is available at https://github.com/shijie-wu/neural-transducer.

4.4 OSTIA

The onward subsequential transducer inference algorithm (OSTIA; Oncina et al., 1993) learns the class of subsequential functions from positive presentations of input–output strings. OSTIA works by first building a prefix-tree transducer of the training data, which is then transformed through a series of state-merging operations into the SFST encoding the function the data is drawn from. If a characteristic sample is contained in the learning data, OSTIA finds a correct transducer in polynomial (cubic) time. Since OSTIA is designed specifically to learn subsequential relations, it provides a useful baseline. Unfortunately, due to its cubic runtime, OSTIA is too slow to use on larger datasets. To give a practical speed-up, we limit the number of samples we provide to OSTIA to 1000, which is only 5% to 10% of what the neural transducers train on. This keeps OSTIA's run time roughly equivalent to its neural counterparts, but unlevels the playing field, as it were.

5 Results and Discussion

We train a neural network, as described in §4.3, on the datasets taken from our sampled SFSTs. We discuss and analyze the results below.

Minimum Transition Coverage. We define transition coverage of a given transition as the percentage of samples in the training dataset that cross that transition. The results reported in Fig. 1 reveal a threshold on the number of samples per transition required to comfortably learn the transducer: 400 samples. This may seem unsurprising given neural networks' "notorious thirst" for data (Lake and Baroni, 2018). In the vast majority of the cases when

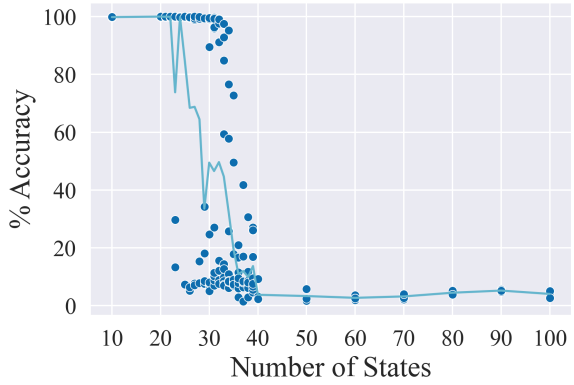

Figure 2: As the number of states in SFST increases, accuracy drops to nearly zero percent between 20 and 40 states.

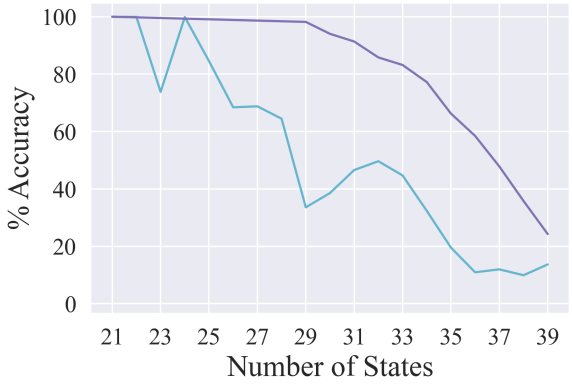

Figure 3: As SFST state size increases, average accuracy (blue line) decreases. Increasing the neural model's size improves accuracy (purple line).

we train on a dataset that does not meet our transition coverage threshold, the neural network does not generalize to held-out data. And, indeed, in the few cases where they manage to have non-zero accuracy, we observe that early transitions in such SFSTs have attained sufficient coverage, and are responsible for the above zero performance. These findings give credence to the idea that there is a relatively simple complexity metric on the SFSTs, i.e., transition coverage, that determines whether or not the neural model will generalize.

Bigger Models Generalize Better. Additionally, we find that our discovered transition coverage threshold is *not* constant across all network sizes. For instance, when we increase the size of the hidden layers in the encoder and decoder from 200 to 300 dimensions, the sequence-to-sequence models are able to generalize on datasets where the transition coverage is lower; see Fig. 3, where the purple line is the average accuracy of the larger model. With the exception of the higher transition coverage threshold, these models follow the same trend as their lower dimensional counterparts.

OSTIA is Slow. In terms of wallclock time, we find that an open-source implementation of OSTIA[6] does not scale to dataset sizes above 1000. This makes it impossible to perform an apples-to-apples comparison of our neural sequence-to-sequence models against OSTIA. On the one hand, reducing the size of the training dataset disadvantages OSTIA. On the other hand, providing OSTIA with the full $20,000$ samples did not terminate after 3 days on a single dataset. OSTIA provably halts after a finite number of steps, but given the above, a proper comparison with neural models is not possible.

[6]The code is available at `github.com/alenaks/OSTIA`.

6 What about SCAN?

We now turn to the SCAN dataset and examine it in light of our findings above. First, we encode SCAN as an SFST. In so doing, we find it to be an outlier in terms of the number of states it requires, which far exceeds the 100 states of our largest SFST. In fact, we calculate that the full SFST encoding SCAN has $7,728$ states; see a small example in Fig. 4. With a finite dataset size of $20,000$ input–output pairs, it should not be possible to learn SCAN with high accuracy. However, unlike other datasets of a similar size, SCAN turns out to be nearly perfectly learnable in our (random-split) experiments. This result stands in contrast to our randomly generated SFSTs, which exhibit a consistent relationship between the complexity of a formal language and its learnability.

We offer a possible theoretical behavior for SCAN's surprising learnability. The class of subregular relations has a native complexity metric built into the formalism: The size of the SFST itself, as measured by the number of transitions. However, the results on the SCAN dataset indicate that the number of transitions in an SFST is *not* the native complexity metric of neural networks under consideration. This should not come as a surprise because neural networks can learn context-free transductions, which are not even representable by an SFST. While it is interesting that we empirically identify a scaling law that consistently applies to our randomly sampled SFSTs, it is not the whole story.

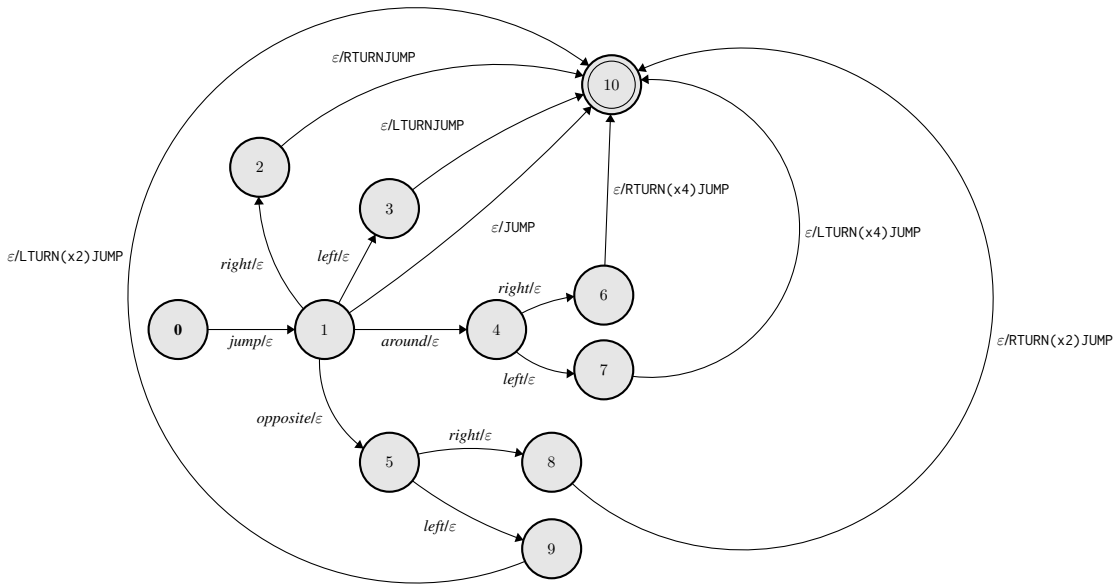

Figure 4: A part of the SFST encoding SCAN; in its entirety, it has $>$ 7000 states.

Indeed, SCAN is not a randomly sampled dataset from the class of SFSTs: It is generated by a hand-crafted synchronous context-free grammar (Lake and Baroni, 2018, Figure 6). While SCAN is a complex SFST, requiring thousands of transitions, it can be encoded by a very small synchronous context-free grammar. Casting SCAN as an SFST therefore misrepresents its native complexity. Thus, it is left for future work to identify a class of automata whose native complexity metric can consistently predict the learnability of arbitrary language tasks; such a class will surely be higher than SFSTs on the Chomsky hierarchy given our reported results.

7 Related Work

Our paper builds on two common strains of research: The construction of datasets to benchmark compositional behavior in neural networks and research in grammatical inference.

7.1 Compositionality Datasets

There is a growing number of artificial language datasets focused on compositionality. Lake and Baroni (2018) introduced a SCAN dataset, made up of simple navigational text commands. The task is to translate the command in the simple natural language into sequences of actions. One successor to SCAN is the NACS dataset (Bastings et al., 2018), which is comparable to SCAN, but instead of mapping multiple input signals to a single duplicated output symbol (e.g., *walk twice* $\rightarrow$ WALK WALK), NACS does the opposite (WALK WALK $\rightarrow$ *walk twice*). Since SCAN is a finite language, its inverse NACS is also a finite language, and it can, thus, also be encoded as an SFST. However, this does not hold true for general SFSTs. Inverting an SFST often results in non-subsequential transducer because the output tape of SFSTs is, in general, not deterministic. Another successor is gSCAN (Ruis et al., 2020) focuses on grounding SCAN-like commands in states of a grid world. This makes gSCAN closer to Mikolov et al.'s (2016) grid world grounding for their agents. In contrast to SCAN, gSCAN requires the agent to learn differences between sizes and colours of different geometric shapes and interact with them, by moving them around the grid world. Executing a gSCAN command is therefore much more difficult than to execute its SCAN counterpart. As Ruis et al. (2020) assert, the gSCAN dataset removes artefacts in SCAN which are not central to the compositional generalization. They find that models perform worse on gSCAN than on SCAN. More recently, Bogin et al. (2022) identify that unobserved "local structures" in compositionality datasets are harder to learn if no similar structures are observed during training.

7.2 Grammatical Inference

Grammatical inference studies the ability to learning classes of formal language from data. Our work focuses on the learning of a restricted class of functions generated by a correspondingly restricted class of finite-state transducers. This allows us to

synthesize our study of compositionality in neural models as rule-based inference by neural models, which we can restrict in principled ways. Finite-state machines generalise many techniques in NLP: probabilistic finite-state automata, hidden Markov models, Markov chains, n-grams, probabilistic suffix trees, deterministic stochastic probabilistic automata, weighted automata, and other syntactic objects which generate distributions over sets of possible infinite cardinality of strings, sequences, words, trees, and graphs (Vidal et al., 2005). Many grammatical inference studies of neural networks test them on samples drawn from some deterministic finite-state acceptors (Cleeremans et al., 1989). See Jacobsson (2005) for a review.

Others experiment with neural networks to see if they can learn languages higher up the Chomsky hierarchy. LSTMs (Hochreiter and Schmidhuber, 1997) can perform dynamic counting and variably learn simple counter languages such as some k-Dyck languages and $a^n b^n$ patterns (Weiss et al., 2018; Suzgun et al., 2019; Bhattamishra et al., 2020; Hewitt et al., 2020), which are generated by a finite-state machine with a counter on top (Schützenberger, 1962). In contrast, Avcu et al. (2017) show that LSTM and other RNN architectures often fail to learn long-distance dependencies drawn from simpler subregular language classes, even on large benchmarks (Mahalunkar and Kelleher, 2019). Nelson et al. (2020) study the inference of sequence-to-sequence networks, showing that RNN, LSTM, and GRU (Cho et al., 2014) systematically fail to learn a wide range of regular string copying functions generated from a family of two-way transducers, which characterize regular string-to-string functions. When augmented with attention, they reliably learn every function, and the attention history mirrors the derivations of the corresponding two-way transducers. These independently productive strands of work in compositionality and inference suggest that our work is a reasonable starting point for future interactions.

8 Conclusion

We study whether neural sequence-to-sequence models are capable of learning string-to-string transductions with Montague-style compositionality, i.e., where compositional behavior is defined to be homomorphic. To execute our study, we first provide a theoretical justification of why SFSTs meet Montague's definition. In the empirical portion of the paper, we randomly sample 2800 SFSTs using the process described in §4.1, and, then, sample input–output pairs from each SFST to create our unique string-to-string transduction datasets. We find that neural networks tend to either generalize completely or fail miserably—with little middle ground. Moreover, we identify a simple complexity metric, transition coverage, that seems to reliably allow us to predict when an SFST in our randomly sampled dataset is learnable from the dataset sampled from it.

Finally, our paper discusses how analyzing SCAN as an SFST provides a counterexample to our contention that transtion coverages reliably predicts the learnability of an SFST from a given dataset. It seems that while transition coverage is a good metric for subregular languages, there are datasets generated by synchronous context-free grammars that are learnable *despite* requiring a large number of transitions when encoded as an SFST. For example, SCAN's learnability is likely due to the fact that the synchronous context-free grammar used to generate it is relatively small and, thus, under a metric such as production coverage it would be considered simple. In conclusion, to get a more complete view of the factors underlying learnability, it may be fruitful to consider not only SFSTs, but other formalisms that describe more complex formal relations. We hypothesize that there might yet be a class of automata whose native complexity will more consistently predict the learnability of a language task.

Acknowledgements

We would like to thank the following people for their valuable insights that made this work possible: Tim Vieira, Hagen Blix, Dieuwke Hupkes, Douwe Kiela, Brenden Lake, Koustuv Sinha, Anej Svete, Alexandra Butoi and Eleanor Chodroff.

References

Jacob Andreas. 2019. Measuring compositionality in representation learning. In *7th International Conference on Learning Representations, ICLR 2019, New Orleans, LA, USA, May 6-9, 2019*. OpenReview.net.

Enes Avcu, Chihiro Shibata, and Jeffrey Heinz. 2017. Subregular complexity and deep learning. In *CLASP Papers in Computational Linguistics: Proceedings of the Conference on Logic and Machine Learning in Natural Language (LaML 2017), Gothenburg, 12 –13 June*, pages 20–33.

Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. 2015. Neural machine translation by jointly learning to align and translate. In *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*.

Borja Balle, Rémi Eyraud, Franco M. Luque, Ariadna Quattoni, and Sicco Verwer. 2017. Results of the sequence prediction challenge (SPiCe): a competition on learning the next symbol in a sequence. In *Proceedings of The 13th International Conference on Grammatical Inference*, volume 57 of *Proceedings of Machine Learning Research*, pages 132–136, Delft, The Netherlands. PMLR.

Jasmijn Bastings, Marco Baroni, Jason Weston, Kyunghyun Cho, and Douwe Kiela. 2018. Jump to better conclusions: SCAN both left and right. *Proceedings of the 2018 EMNLP Workshop BlackboxNLP: Analyzing and Interpreting Neural Networks for NLP*, pages 47–55.

Siddharth Bhaskar, Jane Chandlee, Adam Jardine, and Christopher Oakden. 2020. Boolean monadic recursive schemes as a logical characterization of the subsequential functions. In *Language and Automata Theory and Applications - LATA 2020*, Lecture Notes in Computer Science, pages 157–169. Springer.

Satwik Bhattamishra, Kabir Ahuja, and Navin Goyal. 2020. On the practical ability of recurrent neural networks to recognize hierarchical languages. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 1481–1494, Barcelona, Spain (Online). International Committee on Computational Linguistics.

Ben Bogin, Shivanshu Gupta, and Jonathan Berant. 2022. Unobserved local structures make compositional generalization hard. arXiv.

Lewis Carroll. 1871. *Through the Looking Glass*. Macmillan London.

Kyunghyun Cho, Bart van Merriënboer, Caglar Gulcehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. 2014. Learning phrase representations using RNN encoder–decoder for statistical machine translation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1724–1734, Doha, Qatar. Association for Computational Linguistics.

Christian Choffrut. 2003. Minimizing subsequential transducers: a survey. *Theoretical Computer Science*, 292(1):131 – 143. Selected Papers in honor of Jean Berstel.

Axel Cleeremans, David Servan-Schreiber, and James L. McClelland. 1989. Finite state automata and simple recurrent networks. *Neural Computation*, 1(3):372–381.

Verna Dankers, Elia Bruni, and Dieuwke Hupkes. 2022. The paradox of the compositionality of natural language: A neural machine translation case study. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 4154–4175, Dublin, Ireland. Association for Computational Linguistics.

Jerry Fodor. 1998. There are no recognitional concepts; not even red. *Philosophical Issues*, 9:1–14.

Jerry A. Fodor and Zenon W. Pylyshyn. 1988. Connectionism and cognitive architecture: A critical analysis. *Cognition*, 28(1-2):3–71.

John Hewitt, Michael Hahn, Surya Ganguli, Percy Liang, and Christopher D. Manning. 2020. RNNs can generate bounded hierarchical languages with optimal memory. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1978–2010, Online. Association for Computational Linguistics.

Colin de la Higuera. 2010. *Grammatical Inference: Learning Automata and Grammars*. Cambridge University Press.

Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long short-term memory. *Neural Computation*, 9(8):1735–1780.

Dieuwke Hupkes, Verna Dankers, Mathijs Mul, and Elia Bruni. 2020a. Compositionality decomposed: How do neural networks generalise? (extended abstract). In *Proceedings of the Twenty-Ninth International Joint Conference on Artificial Intelligence, IJCAI-20*, pages 5065–5069. International Joint Conferences on Artificial Intelligence Organization. Journal track.

Dieuwke Hupkes, Verna Dankers, Mathijs Mul, and Elia Bruni. 2020b. Compositionality decomposed: How do neural networks generalise? (extended abstract). In *Proceedings of the Twenty-Ninth International Joint Conference on Artificial Intelligence, IJCAI-20*, pages 5065–5069. International Joint Conferences on Artificial Intelligence Organization. Journal track.

Henrik Jacobsson. 2005. Rule extraction from recurrent neural networks: A taxonomy and review. *Neural Computation*, 17(6):1223–1263.

Diederik P. Kingma and Jimmy Ba. 2015. Adam: A method for stochastic optimization. In *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*.

Brenden Lake and Marco Baroni. 2018. Generalization without systematicity: On the compositional skills of sequence-to-sequence recurrent networks. In *International Conference on Machine Learning*, pages 2873–2882. PMLR.

Abhijit Mahalunkar and John Kelleher. 2019. Multi-element long distance dependencies: Using SPk languages to explore the characteristics of long-distance dependencies. In *Proceedings of the Workshop on Deep Learning and Formal Languages: Building Bridges*, pages 34–43, Florence. Association for Computational Linguistics.

Gary F. Marcus. 1998. Rethinking eliminative connectionism. *Cognitive Psychology*, 37(3):243–282.

Tomas Mikolov, Armand Joulin, and Marco Baroni. 2016. A roadmap towards machine intelligence. In *International Conference on Intelligent Text Processing and Computational Linguistics*, pages 29–61. Springer.

Mehryar Mohri. 1997. Finite-state transducers in language and speech processing. *Computational Linguistics*, 23(2):269–311.

Richard Montague. 1970. Universal grammar. *Theoria*.

Max Nelson, Hossep Dolatian, Jonathan Rawski, and Brandon Prickett. 2020. Probing RNN encoder-decoder generalization of subregular functions using reduplication. In *Proceedings of the Society for Computation in Linguistics 2020*, pages 167–178, New York, New York. Association for Computational Linguistics.

José Oncina, Pedro García, and Enrique Vidal. 1993. Learning subsequential transducers for pattern recognition interpretation tasks. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 15(5):448–458.

Barbara Partee. 1995. Lexical semantics and compositionality. *An Invitation to Cognitive Science: Language*, 1:311–360.

Jonathan Rawski and Jeffrey Heinz. 2019. No free lunch in linguistics or machine learning: Response to pater. *Language*, 95(1):e125–e135.

James Rogers and Geoffrey K. Pullum. 2011. Aural pattern recognition experiments and the subregular hierarchy. *Journal of Logic, Language and Information*, 20(3):329–342.

Laura Ruis, Jacob Andreas, Marco Baroni, Diane Bouchacourt, and Brenden M. Lake. 2020. A benchmark for systematic generalization in grounded language understanding. In *Advances in Neural Information Processing Systems 34*.

Marcel P. Schützenberger. 1962. Finite counting automata. *Information and Control*, 5:91–107.

Mirac Suzgun, Yonatan Belinkov, Stuart Shieber, and Sebastian Gehrmann. 2019. LSTM networks can perform dynamic counting. In *Proceedings of the Workshop on Deep Learning and Formal Languages: Building Bridges*, pages 44–54, Florence. Association for Computational Linguistics.

E. Vidal, F. Thollard, C. de la Higuera, F. Casacuberta, and R. C. Carrasco. 2005. Probabilistic finite-state machines - part I. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 27(7):1013–1025.

Gail Weiss, Yoav Goldberg, and Eran Yahav. 2018. On the practical computational power of finite precision RNNs for language recognition. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 740–745, Melbourne, Australia. Association for Computational Linguistics.

Jennifer C. White and Ryan Cotterell. 2021. Examining the inductive bias of neural language models with artificial languages. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 454–463, Online. Association for Computational Linguistics.

Jennifer C. White and Ryan Cotterell. 2022. Equivariant transduction through invariant alignment. In *Proceedings of the 29th International Conference on Computational Linguistics*, Gyeongju, Republic of Korea. International Committee on Computational Linguistics.

Shijie Wu and Ryan Cotterell. 2019. Exact hard monotonic attention for character-level transduction. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 1530–1537, Florence, Italy. Association for Computational Linguistics.

A Proofs

Proposition 1. *Let Σ and Γ be two alphabets. The function f is a string homomorphism between $(\Sigma^*, \circ)$ and $(\Gamma^*, \circ)$ iff it is a non-empty complete subsequential finite-state transducer with one state.*

Proof. $\Rightarrow$ Assume a homomorphism f and an alphabet Σ are given. Construct a finite-state transducer with $Q = F = \{q_0\}$. Let $\Gamma = \{f(\sigma) \mid \sigma \in \Sigma\}$. Create a transition $q_0 \xrightarrow{\sigma/f(\sigma)} q_0$ for every $\sigma \in \Sigma$.
$\Leftarrow$ Since $\mathcal{T} = \langle \Sigma, \Gamma, Q, q_0, F, \delta \rangle$ is non-empty, we know that its only state q_0 is also a final state. Thus, $\mathcal{T}$ maps $\varepsilon \mapsto \varepsilon$, i.e., the identity element to the identity element. Let $\boldsymbol{\sigma} \in \Sigma^*$ and suppose that under $\mathcal{T}$ we have $\boldsymbol{\sigma} \mapsto \boldsymbol{\gamma}$. Suppose $\boldsymbol{\sigma} = \boldsymbol{\sigma}'\boldsymbol{\sigma}''$. Since $\mathcal{T}$ is subsequential, there is a unique path in $\mathcal{T}$ that transduces $\boldsymbol{\sigma} \mapsto \boldsymbol{\gamma}$ with exactly $|\boldsymbol{\sigma}|$ transitions. Let $\boldsymbol{\gamma}'$ be the output yield of the first $|\boldsymbol{\sigma}'|$ transitions and let $\boldsymbol{\gamma}''$ be the output yield of the next $|\boldsymbol{\sigma}''|$ transitions. Thus, we have that $\boldsymbol{\sigma}' \mapsto \boldsymbol{\gamma}'$ and $\boldsymbol{\sigma}'' \mapsto \boldsymbol{\gamma}''$ which shows that $\mathcal{T}$ is a homomorphism because we have that $\boldsymbol{\gamma} = \boldsymbol{\gamma}'\boldsymbol{\gamma}''$. Finally, we assumed completeness so that f is a total function over Σ^*. ■

Proposition 2. *Let $\mathcal{A} = \langle \Sigma, Q, q_0, F, \delta \rangle$ be a complete deterministic finite-state automaton. Let $\mathcal{M} \stackrel{\text{def}}{=} \left\{ \boldsymbol{M}^\sigma \mid \sigma \in \Sigma \right\}$ be the set of $\mathcal{A}$'s symbol transition matrices. Then $(\mathcal{M}^*, \otimes)$ with $\otimes$ defined in Eq. (2) is a **transition monoid** with $\boldsymbol{E}$ as a distinguished identity element.*

Proof. We define $\boldsymbol{E}$ to be the identity element, i.e., for any $\boldsymbol{A} \in \mathcal{M}^*$, we define $\boldsymbol{E} \otimes \boldsymbol{A} \stackrel{\text{def}}{=} \boldsymbol{A} \stackrel{\text{def}}{=} \boldsymbol{A} \otimes \boldsymbol{E} = \boldsymbol{A}$. Closure follows from the fact for $\boldsymbol{A}, \boldsymbol{B} \in \mathcal{M}^+$, we have that $\boldsymbol{A} \otimes \boldsymbol{B}$ has by construction exactly one non-zero entry in every row, as elaborated upon in the main text. To check associativity, consider

$$\left(\boldsymbol{M}^{\boldsymbol{\sigma}} \otimes \boldsymbol{M}^{\boldsymbol{\sigma}'} \otimes \boldsymbol{M}^{\boldsymbol{\sigma}''}\right)_{ik} = M^{\boldsymbol{\sigma}}_{ij} M^{\boldsymbol{\sigma}'}_{jj'} M^{\boldsymbol{\sigma}''}_{j'k} \tag{4}$$

for some i, j, j', k. By the associativity of string concatenation (including when it is augmented to include the zero string), we have that

$$(M^{\boldsymbol{\sigma}}_{ij} M^{\boldsymbol{\sigma}'}_{jj'}) M^{\boldsymbol{\sigma}'''}_{j'k} = M^{\boldsymbol{\sigma}}_{ij} (M^{\boldsymbol{\sigma}'}_{jj'} M^{\boldsymbol{\sigma}'''}_{j'k}) \tag{5}$$

which in turn implies that

$$\left((\boldsymbol{M}^{\boldsymbol{\sigma}} \otimes \boldsymbol{M}^{\boldsymbol{\sigma}'}) \otimes \boldsymbol{M}^{\boldsymbol{\sigma}''}\right)_{ik} = \left(\boldsymbol{M}^{\boldsymbol{\sigma}} \otimes (\boldsymbol{M}^{\boldsymbol{\sigma}'} \otimes \boldsymbol{M}^{\boldsymbol{\sigma}''})\right)_{ik} \tag{6}$$

Thus, we have that $(\mathcal{M}^*, \otimes)$ is a monoid. ■

Proposition 3. *Let $\mathcal{T} = \langle \Sigma, \Gamma, Q, q_0, F, \delta \rangle$ be a complete subsequential finite-state transducer. Let $(\mathcal{M}^*_\Sigma, \otimes)$ be the transition monoid associated with the input alphabet Σ and let $(\mathcal{M}^*_\Gamma, \otimes)$ be the transition monoid associated with the output alphabet Γ. Then there exists a homomorphism $f : \mathcal{M}^*_\Sigma \to \mathcal{M}^*_\Gamma$.*

Proof. We define f as follows. First, we define $f(\boldsymbol{E}) \stackrel{\text{def}}{=} \boldsymbol{E}$. Next, consider an arbitrary element $\mathcal{M}^*_\Sigma$. Because $\mathcal{M}^*_\Sigma$ is a Kleene closure, we may write this arbitrary element as $\boldsymbol{M}^{\sigma_1}_\Sigma \otimes \cdots \otimes \boldsymbol{M}^{\sigma_K}_\Sigma$. Now we define $f(\boldsymbol{M}^{\sigma_1}_\Sigma \otimes \cdots \otimes \boldsymbol{M}^{\sigma_K}_\Sigma) \stackrel{\text{def}}{=} \boldsymbol{M}^{\sigma_1}_\Gamma \otimes \cdots \otimes \boldsymbol{M}^{\sigma_K}_\Gamma$. Thus,

$$\begin{aligned}
&f(\boldsymbol{M}^{\sigma_1}_\Sigma \otimes \cdots \otimes \boldsymbol{M}^{\sigma_K}_\Sigma) \\
&= f\left((\boldsymbol{M}^{\sigma_1}_\Sigma \otimes \cdots \otimes \boldsymbol{M}^{\sigma_k}_\Sigma) \otimes (\boldsymbol{M}^{\sigma_{k+1}}_\Sigma \otimes \cdots \otimes \boldsymbol{M}^{\sigma_K}_\Sigma)\right) \\
&= \left(\boldsymbol{M}^{\sigma_1}_\Gamma \otimes \cdots \otimes \boldsymbol{M}^{\sigma_k}_\Gamma\right) \otimes \left(\boldsymbol{M}^{\sigma_{k+1}}_\Gamma \otimes \cdots \otimes \boldsymbol{M}^{\sigma_K}_\Gamma\right) \\
&= \boldsymbol{M}^{\sigma_1}_\Gamma \otimes \cdots \otimes \boldsymbol{M}^{\sigma_K}_\Gamma
\end{aligned}$$

This proves f is a homomorphism. ■

Source-summary Entity Aggregation in Abstractive Summarization

José Ángel González
Valencian Research Institute
for Artificial Intelligence, Valencia
jogonba2@dsic.upv.es

Annie Louis
Google Research, London
annielouis@google.com

Jackie C. K. Cheung
McGill University/MILA, Montreal
Canada CIFAR AI Chair
jcheung@cs.mcgill.ca

Abstract

In a text, entities mentioned earlier can be referred to in later discourse by a more general description. For example, *Celine Dion* and *Justin Bieber* can be referred to by *Canadian singers* or *celebrities*. In this work, we study this phenomenon in the context of summarization, where entities from a source text are generalized in the summary. We call such instances *source-summary entity aggregations*. We categorize these aggregations into two types and analyze them in the CNN/DAILYMAIL corpus, showing that they are reasonably frequent. We then examine how well three state-of-the-art summarization systems can generate such aggregations within summaries. We also develop techniques to encourage them to generate more aggregations. Our results show that there is significant room for improvement in producing semantically correct aggregations.

1 Introduction

The quality of abstractive summarization systems has improved substantially in the past few years. An important next research question is to better understand the specific linguistic and semantic operations which can lead to high-quality abstractive text. In this work, we focus on how entities can be referred to in summaries, especially with an expression more general than in the source. For example, Table 1 demonstrates how three comic book characters mentioned in the source document are aggregated in a reference summary by the expression, "three of the most well-known comic book characters of all time". Such referring expressions are interesting for abstractive summarization, since they are *novel summary n-grams* that result from semantic inference from the source.

There are few existing studies about the semantics of text generated by current abstractive systems. Some have focused on summary n-grams that are not found in the source text (Kryściński et al., 2018; Song et al., 2020), and others that look at problems resulting from undesirable summary content, e.g., hallucinations (Maynez et al., 2020; Kryściński et al., 2020).

Document	(CNN) Comic books of the past few years have seen a lot of changes (a female Thor, anyone?) but not quite so many at one time. Three major characters – Superman, Wonder Woman (both of DC Comics, a Time Warner company, like CNN) and Archie Andrews – came out with new looks (...)
Summary	Superman, Wonder Woman and Archie all debuted new looks Thursday. Three of the most well-known comic book characters of all time look radically different.

Table 1: An example of source-summary entity aggregation. The aggregation "*Three of the most well-known comic book characters of all time*" is used in the summary to aggregate the entities "*Superman*", "*Wonder Woman*", and "*Archie Andrews*" named in the document.

We focus on a specific semantic operation that summary writers can perform in order to change the level of detail in a summary: the semantic aggregation of named entities in context, as in Table 1. We estimate the prevalence of such aggregations in the CNN/DAILYMAIL corpus (Hermann et al., 2015). We also categorize the aggregations that we find into those (i) where the models can *copy* the aggregations from the source document, and (ii) those cases where the models are required to generate *novel* aggregations not found in the source.

We then explore how well existing systems can generate *copy* and *novel* aggregations that match those found in reference summaries. Specifically, given a document, the models must generate a summary, and the aggregations within the generated summary are evaluated against the aggregations in the reference summary.

We evaluate three state-of-the-art Transformer-based abstractive summarization systems: BART (Lewis et al., 2020), PEGASUS (Zhang et al., 2020),

Proceedings of the 29th International Conference on Computational Linguistics, pages 6019–6034
October 12–17, 2022.

and T_5 (Raffel et al., 2020). The experimental results show that the task is hard, especially for generating novel aggregations. We also explore how to fine-tune BART (Lewis et al., 2020) to generate more summary-worthy aggregations without compromising the overall summary quality. The performance of all the summarization models is still far below the upper bounds posed by our oracles, showing that there is room for improvement.

2 Related work

Semantic generalization in automatic summarization is receiving increasing interest, both in terms of the data and the models. In Belkebir and Guessoum (2016), the authors fuse concepts within sentences using hypernymy relations from taxonomies such as WordNet. Kouris et al. (2019) focuses on abstracting single concepts. Roughly, they train an encoder-decoder architecture on documents where single nouns are replaced with hypernyms, to produce more general summaries. Contrary to these approaches based on taxonomies, Kryściński et al. (2018) use a mixed objective for training encoder-decoder architectures to encourage abstraction in summaries. The level of abstraction was defined in terms of novel n-grams.

The surface-level novel n-grams definition of abstractiveness has also been used in recent summarization datasets (Grusky et al., 2018; Narayan et al., 2018). This approximation is convenient for generation since it measures any kind of rewriting. However, being able to explicitly measure different types of abstraction is important for tracking progress. Our work is based on this motivation. The closest idea towards entity aggregation is Jumel et al. (2020) and we draw heavily from their work. They introduce a dataset and task (TESA) which consists of producing a non-enumerating noun phrase ('former US presidents') that aggregates a set of entities ('Clinton', 'Bush') in a textual context (a New York Times article). Their data was collected using crowd annotators and does not specifically focus on any task. Our work explores entity aggregations in the context of abstractive summarization.

Our task can be seen as a referring expression generation problem (Stone, 2000; Krahmer and van Deemter, 2012) where a general phrase in the summary stands in for a set of entities in the source. The task is also related to multi-antecedent coreference resolution and split-antecedent anaphora (Yu et al., 2020; Burga et al., 2016; Vala et al., 2016), but most resources and approaches here are aimed at pronominal coreference. Some studies address entity-driven summarization (Zhou et al., 2021; Sharma et al., 2019), with the aim of focusing the summaries on the most salient entities. Differently to our work, this work does not focus explicitly on the aggregation expressions in the summaries.

Currently, abstractive summarization systems have vastly improved generation capabilities, achieved by using pre-trained Transformers (Vaswani et al., 2017) as a backbone. Among the state of the art of text summarization benchmarks, BART (Lewis et al., 2020), PEGASUS (Zhang et al., 2020), PROPHETNET (Qi et al., 2020) and T_5 (Raffel et al., 2020) stand out in terms of ROUGE. Some challenges with these models, such as mitigating hallucinations or ensuring factual consistency, are of great interest. Different methods have been proposed here (Zhao et al., 2020; Nan et al., 2021b). Of these Narayan et al. (2021); Nan et al. (2021a) incorporate text planning using a sequence of entities to prompt faithful generation. We use such ideas to encourage systems to produce more aggregations by jointly training the models to identify summary-worthy aggregations while learning to generate summaries.

3 Corpus Study

3.1 Defining source-summary entity aggregations

We categorize source-summary entity aggregation into two types: a) **copy aggregations** found in the source and b) **novel aggregations** that must be generated. In copy aggregations, an aggregating expression for entities is present in the source and the same expression is also in the summary. In the novel case, entities from the source are aggregated by a new expression not found in the source. Table 2 shows examples of the two types.

This distinction is useful for summarization systems since a system will understandably find it easier to copy an expression from the source. Generating novel aggregations is likely more difficult, in theory requiring deeper semantic understanding of the source content.

3.2 Data Source

One would like to understand how often aggregations are used in human summaries and also how well systems currently handle them. However,

Document	Summary	Type
Through the process of elimination (...), our guess as to which five states. White will play on the brief acoustic run: South and North Dakota, Wyoming, Vermont and ... Puerto Rico?	Jack White taking a hiatus from touring after brief acoustic jaunt . He'll play five states he has yet to get to, charge just $3 . Places and times of shows are currently a mystery.	*Copy* Aggregation
Camuti and Rakes were longtime business associates, and Camuti allegedly poisoned Rakes at a time when Camuti owed money to Rakes. (...)	William Camuti, 69, is charged with attempted murder and misleading police. Camuti and victim Stephen Rakes were longtime business associates.	
Sometime in the not-too-distant future, Kanye West can once again (...) The rapper and his reality TV star girlfriend, Kim Kardashian, are having a girl.	News comes the same day Kardashian has a baby shower. The couple has been dating since last year . The baby is due next month.	*Novel* Aggregation
After a youth rally in the Bahamas National Stadium Monday, Harry travels to Jamaica and then on to Brazil to complete his 10-day tour (...)	The Bahamas is Prince Harry's second stop in a 10-day Caribbean tour . The 27-year-old prince is celebrating the Diamond Jubilee of Queen Elizabeth II . Jamaica and Brazil are Harry's next two destinations.	

Table 2: Examples from the CNN/DAILYMAIL for the different cases of source-summary entity aggregation. Aggregations are in green and entities are in blue.

doing so requires a dataset with entities marked together with their aggregations, also aligned between source and summary texts. Such annotations are costly to produce in practice. So, we approximate counts using heuristics which we will outline in this section.

We use the TESA (Jumel et al., 2020) dataset, where sets of entities are paired with human-written aggregation expressions. We briefly describe this dataset and then how we designed our heuristics.

The TESA corpus comprises 1,718 'aggregatable instances', each one consisting of (a) a set of named entities of the same type (person, location, and organization), (b) a context (excerpt) from an article in the NEW YORK TIMES corpus (Sandhaus, Evan, 2008) involving the entities, (c) background knowledge of the entities from Wikipedia, and (d) at least one human-written aggregation. An example of an aggregatable instance is shown in Table 3.

Background	**Microsoft**: Microsoft Corporation is an American multinational technology company (...) **Sony**: Sony Corporation is a Japanese multinational conglomerate corporation (...)
Context	Battleground For Consoles Moves Online: Over all, though, it is Microsoft that has had the steeper mountain to climb. In the last generation of video game consoles, Sony had a roughly 60 percent market share, compared to 20 percent for each Microsoft and Nintendo.
Aggregations	technology companies, multinational corporations

Table 3: Example of an aggregatable instance from TESA. The entities of the set are in blue and the annotated aggregations are in green.

Table 4 shows several basic statistics of TESA. Table 9 in Appendix A shows some of the most frequent entity aggregations. Note that these examples are not specific to summarization, and they show entity aggregation in the context of a news article. In addition, the corpus only contains named entities which are persons, organizations, or locations. So, this set is only a small subset of possible source-summary relations. However, given the difficulty of defining and identifying such aggregations, we similarly limit our work to the same types of named entities.

	Entity Sets	Entities	Aggregations
Person	941 (801)	2228 (1201)	2900 (951)
Location	629 (412)	1606 (278)	2041 (505)
Organization	148 (123)	310 (196)	456 (239)

Table 4: Statistics of each type in the TESA corpus. We indicate the total count of occurrences, and in parentheses the count of unique occurrences.

3.3 Prevalence in the CNN/DailyMail corpus

To determine the prevalence of source-summary entity aggregations in the CNN/DAILYMAIL corpus, we computed the percentage of document-summary pairs containing each type of aggregation. We do this heuristically due to the difficulty of labeling. This process consists of the following steps:

1. Identify sets of entities in a source document that could be aggregated.

2. Identify the possible aggregations in the source and summary documents that could be matched with each source entity set.

3.3.1 Detecting aggregatable entity tuples

We used four lexico-syntactic patterns to detect tuples of entities in source documents which could potentially be aggregated. Entities[1] in a tuple must

[1]We used the Spacy pipeline (Montani et al., 2022) with RoBERTa-base (Liu et al., 2019) to extract named entities and noun phrases.

be of the same entity type (person, location, or organization). These patterns are:

- **Coordinating conjunctions**: list of entities separated by , or ; ending with a conjunction (*and/or*), e.g., *John, Peter, and Mary*.
- **In sentence**: entities mentioned at any position within the same sentence, excluding the **Coordinating conjunctions** pattern, e.g., *Today John meets Mary*.
- **Contiguous sentences**: entities used to begin contiguous sentences, e.g., *John went to the beach. Mary went to the mountain.*
- **Shared nouns**: entities mentioned at any position that are preceded by the same singular noun phrase, e.g., *Rock climber John prefers a change of scenery. In contrast, rock climber Mary prefers the mountain.*

This step gives us aggregatable entity tuples from a source document.

3.3.2 Identifying possible aggregations

This step identifies likely aggregation expressions. First, we find likely expressions for these entity tuples in the source, where possible. For that, we use a collection of heuristics to pick candidate expressions within close proximity to the entities. We will then explain how we align these aggregations to those in the summary (to identify if they are novel or copied from the source).

The following heuristics identify aggregations in close proximity to entity tuples *in the source*:

- **Previous sentence**: a noun phrase in the sentence that precedes the span of sentences containing the entities, e.g., *The rock climbers are traveling by the world. Some of them are John and Mary.*
- **In span**: a noun phrase in the span of sentences containing the entities, e.g., *John went to the beach, he is one of the traveling rock climbers. Mary went to the mountain.*
- **Next sentence**: a noun phrase in the sentence that follows the span of sentences containing the entities, e.g., *Today John meets Mary. Both rock climbers will have a virtual meeting.*
- **Preceding a list**: a noun phrase which introduces a list, with phrases such as (*like*, *such as*, *including*), whitespaces, commas, semicolons, and '—', e.g., *Young rock climbers such as John and Mary are traveling today*. This pattern can only be used to get aggregations of entities obtained by **Coordinating Conjunctions**.
- **Preceding entities**: the plural form of a singular noun phrase that precedes all the entities, e.g., *Rock climber John prefers a change of scenery. In contrast, rock climber Mary prefers the mountain.* This pattern can only be used with **Shared nouns**.

Note that these candidate aggregations may be noisy. So, we only select a noun phrase if it contains an aggregation expression from the TESA corpus. For example, 'young rock climbers' would be selected if any of 'young rock climbers', 'rock climbers' or 'climbers' is present in TESA.

At the end of this step, we have tuples of entities in a source document. Each tuple may be mapped to a list of possible aggregation expressions in the source or no aggregation at all. We show one example of an entity tuple and an associated identified aggregation from the CNN/DAILYMAIL: "*He is the first Western leader to visit one of the three worst affected west African countries - Liberia, Sierra Leone and Guinea.*". Table 10 in Appendix A shows more examples.

We now use these sets to identify copy aggregations and novel aggregations *in summaries of these documents*.

Copy aggregations: are source aggregations which are also present in the summary. When an entity tuple does not have a source aggregation from the previous step, we drop it. For the remaining tuples, we check whether one of its aggregations is present in the summary. That entity tuple-aggregation pair is a copy aggregation.

Novel aggregations: Here, the aggregation must appear in the summary and not in the source. So the extracted entity tuples and their 'source' aggregations are used differently here.

We create an overall map to be used across the whole set of documents. For each aggregation expression, we aim to create a list of entities, any subset of which can be aggregated by that expression. We do this by merging the aggregation expressions and entity tuples identified in CNN/DAILYMAIL using the heuristics in this section. For example, suppose the corpus contains two documents A and B.

Document A has the source tuple-aggregation pair {'Biden', 'Harris'} → 'politicians' and document B contains {'Modi', 'Johnson'} → 'politicians', then the overall map will contain 'politicians' → {'Bush', 'Clinton', 'Johnson', 'Modi'}. Note that this step merges the entities for the same expression across all of the CNN/DailyMail corpus. To increase the coverage of this overall map, we also add aggregation-entity pairs from the TESA corpus. i.e. If TESA contains the annotation {'Modi', 'Johnson'} → 'prime ministers', the entry 'prime ministers' → {'Johnson', 'Modi'} is also added to our table.

This map is now a broad list of aggregations and possible candidate entities. To identify novel aggregations, we find those cases where an aggregation expression from the table is in the summary but not in the source, and any subset of entities from its entity list is present in the source. Note that this subset may be {'Modi', 'Biden'} which matches a new document C now.

Because our heuristics differ in terms of precision and recall, we computed *low* and *high* estimates of the percentage of documents containing each case. For the low estimates, we only used **Coordinating conjunctions+Preceding a list** since it showed almost 100% precision in our preliminary evaluation (but a very low recall). For the high estimates, all the heuristics are used.

We found that up to 15% of all document-summary pairs in the CNN/DAILYMAIL corpus could contain some type of source-summary aggregation, being thus a reasonably frequent phenomenon even in such an extractive dataset. Following the low estimate, *novel* aggregations are more frequent than *copy* aggregations from the source (1.13% vs. 0.64%). However, following the high estimate, it seems that *copy* aggregations are more frequent (10.95% vs. 4.99%).

Note that to reduce noise, we filtered our expressions using the manual expressions from TESA which covers New York Times articles mostly from an earlier time period, and only entities which are salient and of fixed named entity types. Consequently, we are likely underestimating the prevalence of source-summary entity aggregation in the CNN/DAILYMAIL corpus.

4 Experimental setup

Our experiments aim to evaluate the capabilities of state-of-the-art summarization models to generate summary-worthy aggregations. In this section, we describe the task, our oracles, models and evaluation measures.

4.1 Task definition

Given a document, the models must generate a summary, and the aggregations within the generated summary must match or be close to those in the reference summary.

We built two test sets from the development and test partitions of the CNN/DAILYMAIL corpus: the COPY and NOVEL sets. We make this distinction to independently evaluate *copy* and *novel* aggregations. To create these sets, we use a broad heuristic compared to our corpus study. Here we gather all noun phrases in the summary and check if their span contains a TESA aggregation. If so, we call it an aggregation and check it against the source to separate into copy and novel sets. Note that here we do not obtain an alignment with entity tuples in the source as in our corpus study. Such alignments would have lower coverage and greater noise, hence we opt for this simple heuristic here.

Therefore each sample in these sets is a triple (D, S, $\mathcal{A}$), where D is a document, S its summary, and $\mathcal{A}$ is the set of aggregation expressions in S. All our samples have non-empty $\mathcal{A}$ sets. In the COPY set, $\mathcal{A}$ only contains aggregations that also appear verbatim in the document. In the NOVEL set, the aggregations $\mathcal{A}$ do not appear verbatim in the document.

Basic statistics of both test sets are shown in Table 5. Table 6 shows the most frequent aggregations. We show some examples of source and references in Table 12 of Appendix A.

	COPY	NOVEL
Examples	4156	2905
Avg words (document)	754.74 (351.70)	750.72 (365.52)
Avg words (summary)	61.44 (31.63)	61.53 (39.22)
Avg words (aggregations)	1.78 (0.82)	1.12 (0.38)
Unique aggregations	1744	2120
Avg of aggregations	1.22 (0.51)	1.12 (0.38)
%CNN/DAILYMAIL val+test	16.71%	11.69%

Table 5: Statistics of the COPY and NOVEL sets. Standard deviations are in parentheses.

4.2 Evaluation measures

We evaluate the performance of the models from two points of view: at the aggregation level (how well do the aggregations of the generated summaries match those of the reference summaries?) and at the summary level (how good are the generated summaries compared to the references?).

COPY	Person	women (257), friends (173), family (160), men (136), his family (113)
	Location	countries (17), cities (14), locations (6), communities (6), states (4)
	Org.	schools (33), companies (17), businesses (15), teams (13), groups (11)
NOVEL	Person	the pair (117), his family (49), the couple (29), the men (24), officials (20)
	Location	other countries (4), countries (4), other cities (3), communities (3), cities(3)
	Org.	schools (6), the two teams (4), both teams (4), other teams (3), record labels (3)

Table 6: The five most frequent aggregations per entity type in COPY and NOVEL test sets. We indicate in parentheses the count of occurrences.

For the aggregation-level evaluation, we consider three metrics. Two are based on overlaps (**Aggregation exact match** and **Token exact match**) and the third relies on similarities among aggregations (**Aggregation relaxed match**).

Aggregation exact match: Let $\mathcal{A}_{ref}$ and $\mathcal{A}_{gen}$ be the aggregations in the reference and the generated summary respectively. Precision is defined as $P = \frac{|\mathcal{A}_{ref} \cap \mathcal{A}_{gen}|}{|\mathcal{A}_{gen}|}$ and recall as $R = \frac{|\mathcal{A}_{ref} \cap \mathcal{A}_{gen}|}{|\mathcal{A}_{ref}|}$.

Token exact match: Let $\mathcal{T}_{ref}$ and $\mathcal{T}_{gen}$ be the sets of words in the aggregations of the reference and the generated summary respectively. Precision is defined as $P = \frac{|\mathcal{T}_{ref} \cap \mathcal{T}_{gen}|}{|\mathcal{T}_{gen}|}$ and recall as $R = \frac{|\mathcal{T}_{ref} \cap \mathcal{T}_{gen}|}{|\mathcal{T}_{ref}|}$. This score does not constrain the expression phrases to match exactly.

Aggregation relaxed match: This variant also measures matches when the generated aggregations do not have word overlap with the reference aggregations, e.g., $\mathcal{A}_{ref} = \{news\ websites\}$ and $\mathcal{A}_{gen} = \{online\ newspapers\}$. In those cases, the two previous measures are too restrictive. So, we propose a measure based on similarities among the aggregations, computed by using BERTSCORE (BS) (Zhang et al., 2019). The precision and recall of this measure are defined as follows:

$$P = \frac{1}{|\mathcal{A}_{gen}|} \sum_{a_{gen} \in \mathcal{A}_{gen}} \max_{a_{ref} \in \mathcal{A}_{ref}} \text{BS}(a_{gen}, a_{ref}) \quad (1)$$

$$R = \frac{1}{|\mathcal{A}_{ref}|} \sum_{a_{ref} \in \mathcal{A}_{ref}} \max_{a_{gen} \in \mathcal{A}_{gen}} \text{BS}(a_{gen}, a_{ref}) \quad (2)$$

For the summary-level evaluation, we report both ROUGE-F_1 scores (Lin, 2004) and BERTSCORE to assess the generated summaries. We do not perform human evaluations of content quality because we only want to check that it has not dropped drastically.

4.3 Models

We used three state-of-the-art abstractive summarization models as the main systems: BART[2] (Lewis et al., 2020), PEGASUS[3] (Zhang et al., 2020) and T_5[4] (Raffel et al., 2020). All these systems are fine-tuned using the training set of the CNN/DAILYMAIL, and evaluated on the COPY and NOVEL sets. All models were implemented using HuggingFace Transformers (Wolf et al., 2020).

Early findings showed that BART generates better aggregations than the other two. So, we also explored how to fine-tune BART to generate more summary-worthy aggregations. These new approaches were fine-tuned for summarization using the same BART hyper-parameters reported in Lewis et al. (2020).

4.3.1 BART+PretrainingAggregations

This approach tailors the pre-trained BART[5] towards aggregations before fine-tuning it on summarization. To do so, we further pre-train BART to reconstruct documents with masked aggregations. We expect this reconstruction knowledge to transfer to summarization.

The pre-training dataset comprises all the documents from the training set of CNN/DAILYMAIL. For each document, we mask aggregations (noun phrases filtered by TESA), plural noun phrases, and random spans until 30% of the tokens are masked. The BART checkpoint is further pre-trained during two epochs with batches of 64 samples to optimize the cross-entropy between the decoder's output and the original document.

4.3.2 BART+AggregationChains

Recent works have shown that jointly learning to generate a sequence of summary-worthy entities followed by the summary can steer summaries towards those entities and also reduce hallucinations (Nan et al., 2021a; Narayan et al., 2021). We use a similar approach to encourage aggregations.

We fine-tune BART[5] to jointly generate the sequence of aggregations of the summary, fol-

[2] https://bit.ly/3fK3ZxU
[3] https://bit.ly/3tIHbXC
[4] https://bit.ly/3fFKXbV
[5] https://bit.ly/3AjKl5C

lowed by the summary. We built the dataset for fine-tuning BART from the training set of the CNN/DAILYMAIL corpus, discarding those samples whose summary has no aggregations. We fine-tuned the model during 20k steps with batches of 80 samples, as in (Lewis et al., 2020).

The target sequences for fine-tuning follow the format of Narayan et al. (2021), e.g., "[CHAIN] *rock climbers* | *friends* ||| *rivals* [SUMMARY] John and Mary are *rock climbers* and *friends*. They are also *rivals*.". During evaluation, the generated aggregation chain is removed.

4.3.3 Gating BARTs

We found that improvements in aggregation production were typically accompanied with a deterioration in the summary content quality metrics. This difference is pronounced when evaluating on the whole CNN/DAILYMAIL test set (since our aggregation models are trained only on the subset which has aggregations). We aim to alleviate this inverse correlation by combining the best model at each evaluation level: BART+AggregationChains (aggregation level: the joint model just described) and BART^2 (summary level: a baseline summarization model).

We use BART+AggregationChains for summaries which contain aggregations, and BART^2 otherwise. We use a binary classifier to determine these cases. This classifier is a DEBERTA-LARGE[6] model, fine-tuned on the training set of the CNN/DAILYMAIL, to determine, given a document, if the reference summary has aggregations. We fine-tuned the classifier for six epochs using batches of 64 samples. This binary task can be done with an accuracy of 76% and an average F-score of 70.5. Table 11 of Appendix A shows the detailed results.

4.4 Oracles

We determine the upper bounds for our models using oracles of the above models. They were fine-tuned as described above for summarization. But during inference, essential information involving the aggregations of the reference summaries is disclosed as an input.

4.4.1 BART+PerfectAggregationChains

This oracle observes how BART+AggregationChains would perform if it generated perfect aggregation chains. BART+AggregationChains first generates an aggregation chain, that is attended through decoder self-attention to condition the generation of the summary. Therefore, an upper bound on its performance is obtained by using as prefix for the decoder the chain with the oracle aggregations of the reference summary.

4.4.2 Copy Sentence Oracles

BART, PEGASUS, and T_5 exhibit a strong behavior toward copying content from the source due to the high degree of extractivity in the CNN/DAILYMAIL corpus. We observe how these systems would behave if they could copy entire sentences with aggregations from the source into the summary.

We build three copy sentence oracles: Copy BART, Copy PEGASUS, and Copy T_5. Each sentence with aggregations in the reference summary is copied into the source document, and placed after the most similar sentence in the document in terms of averaged ROUGE scores. Then, each system is used to summarize the enriched document. Note that these oracles are much more informed than BART+PerfectAggregationChains since full sentences of the reference summary are disclosed.

5 Results

The results of the models and the oracles, at aggregation and summary levels, are presented in Tables 7 and 8 respectively. Note that COPY and NOVEL results are on a subset of the CNN/DAILYMAIL test set where aggregations are present in the summary. The full test set results are also reported separately.

All the models struggle at aggregation-level with the NOVEL test set, especially when they are evaluated using the most restrictive metric (Aggregation exact match). In this case, the models obtain approximately 10 times lower performance than with the COPY test set. Even with the COPY test set, where the models can copy the aggregations from the source to the summary, the results are almost always lower than 50 precision, recall, and F_1, which shows the difficulty of the task for current summarization approaches.

At the aggregation-level, BART+AggregationChains systematically outperforms all the other systems, showing that content planning with aggregation chains is an appropriate strategy to generate more summary-worthy aggregations. However, its performance at the summary-level is worse than that of BART, especially on the CNN/DAILYMAIL test set. This

[6] https://bit.ly/3sOh7cQ

	COPY			NOVEL		
	Aggregation Exact Match	Token Exact Match	Aggregation Relaxed Match	Aggregation Exact Match	Token Exact Match	Aggregation Relaxed Match
Main Systems						
BART	**31.35**	**40.86**	**45.97**	**3.87**	**17.58**	**24.47**
PEGASUS	29.32	38.25	42.97	3.71	16.40	22.15
T_5	28.12	37.36	43.39	3.79	16.20	22.67
Fine-tuned on Aggregations						
BART+PretrainingAggregations	31.07	40.69	45.89	3.69	16.88	24.38
BART+AggregationChains	**37.65**	**51.32**	**58.49**	**5.39**	**23.44**	**35.11**
Gating BARTs	35.15	47.69	54.41	4.55	20.01	29.55
Oracles						
Copy BART	44.11	51.41	55.71	22.17	34.18	39.25
Copy PEGASUS	43.18	50.36	54.30	18.64	30.72	34.94
Copy T_5	38.87	46.64	51.80	15.31	28.12	33.25
BART+PerfectAggregationChains	**59.73**	**67.95**	**71.85**	**39.23**	**54.80**	**58.97**

Table 7: Aggregation-level results of each model (F_1 scores) for each test set.

	COPY				NOVEL				CNN/DailyMail test set			
	R-1	R-2	R-L	BS	R-1	R-2	R-L	BS	R-1	R-2	R-L	BS
Main systems												
BART	**46.11**	22.89	43.03	31.73	42.95	18.70	39.60	28.19	44.06	21.07	41.00	30.53
PEGASUS	46.00	**23.17**	**43.06**	**36.35**	**43.14**	**19.20**	**39.99**	**33.31**	**44.60**	**21.65**	**41.64**	**35.84**
T_5	43.21	20.36	40.27	27.74	40.25	16.68	37.15	24.30	41.57	18.92	38.68	25.58
Fine-tuned on Aggregations												
BART+PretrainingAggregations	**46.04**	**22.73**	**43.06**	**35.21**	**43.10**	**18.85**	**39.99**	**32.18**	**44.05**	**20.96**	**41.13**	**34.25**
BART+AggregationChains	45.14	21.94	42.12	34.71	41.96	18.00	38.86	31.25	42.02	19.05	38.93	31.95
Gating BARTs	45.08	21.84	42.06	34.20	42.19	18.22	39.03	30.56	43.32	20.37	40.30	31.57
Oracles												
Copy BART	48.92	27.32	46.05	34.74	**46.20**	23.66	43.16	31.67	44.90	22.36	41.89	31.41
Copy PEGASUS	**49.17**	**27.91**	**46.41**	**39.71**	46.16	**23.68**	**43.24**	**36.49**	**45.48**	**22.96**	**42.55**	**36.79**
Copy T_5	45.54	23.86	42.82	29.97	42.57	19.83	39.71	26.49	42.22	19.88	39.40	26.22
BART+PerfectAggregationChains	46.22	23.97	43.42	33.35	43.41	19.98	40.53	30.76	43.87	20.97	41.02	33.76

Table 8: Summary-level results of each model (F_1 scores) for each test set, along with the CNN/DAILYMAIL test set. R stands for ROUGE and BS for BERTSCORE.

inverse correlation is alleviated by Gating BARTs, which trades off the aggregation-level and the summary-level performance better than BART and BART+AggregationChains. In this way, Gating BARTs represents an intermediate point between BART and BART+AggregationChains, that generally obtains better aggregation-level performance than BART and better summary-level performance than BART+AggregationChains. Regarding BART+PretrainingAggregations, its performance does not significantly differ from BART neither at aggregation-level nor at summary-level.

The upper bounds of the performance at aggregation and summary levels are posed by BART+PerfectAggregationChains and Copy PEGASUS oracles respectively. At the aggregation level, the performance of all the models is far below the upper bound posed by BART+PerfectAggregationChains, which suggests that there is great room for improvement of summarization systems. In addition, although the copy sentence oracles also have access to the reference aggregations, they obtain significantly worse results than the BART+PerfectAggregationChains. It suggests that summarization systems that have not been trained to consider aggregations properly will struggle in the aggregation-level evaluation. At the summary-level, the performance of the models is more similar to the oracles than in the aggregation-level evaluation, which suggests that there is smaller room for improvement here compared to aggregation-level.

Table 12 of Appendix A illustrates sample outputs from BART and BART+AggregationChains systems.

6 Conclusion and future work

We studied source-summary entity aggregation, a frequent phenomenon in the CNN/DAILYMAIL corpus. We analyzed the capabilities of state-of-the-art summarization systems to generate summary-worthy aggregations, and explored different ways of fine-tuning BART to generate more aggregations. Our results suggest that summarization models can improve greatly along these lines.

In future work, we would like to explore how to leverage knowledge about the entities to generate better aggregations. Another important direction is other types of semantic generalization, such as aggregations of sequences of events. Also, we plan

to explore source-summary entity aggregation on more abstractive summarization datasets such as XSUM and WIKIHOW, which could reflect better the aggregation phenomenon. Finally, we would like to investigate deeper questions about semantic aggregation in summarization, e.g., when is generality preferred over specificity in summaries?

7 Ethical considerations

The TESA dataset centers around specific topics found in the NEW YORK TIMES corpus during 2006-2007. They are skewed towards the male gender, and newsworthy entities involved in politics, business, etc. This selection limits the diversity of the aggregations used in our work. Even though models trained on the data learn semantic abstractions which aids generalization, we need further studies to explore how they differ in performance for different classes of entities.

Our models also share the same research issues as other abstractive systems and further work on reducing hallucinations, and factual inconsistencies will improve our approaches as well.

Acknowledgements

We would like to thank Clément Jumel for his support in the initial steps of the project and for providing us the codebase and the TESA dataset.

This work has been partially supported by Ministerio de Ciencia e Innovación & FEDER funds under the project BEWORD PID2021-126061OB-C41.

References

Riadh Belkebir and Ahmed Guessoum. 2016. Concept generalization and fusion for abstractive sentence generation. *Expert Systems with Applications*, 53:43–56.

Alicia Burga, Sergio Cajal, Joan Codina-Filbà, and Leo Wanner. 2016. Towards multiple antecedent coreference resolution in specialized discourse. In *Proceedings of the Tenth International Conference on Language Resources and Evaluation (LREC'16)*, pages 2052–2057, Portorož, Slovenia. European Language Resources Association (ELRA).

Max Grusky, Mor Naaman, and Yoav Artzi. 2018. Newsroom: A dataset of 1.3 million summaries with diverse extractive strategies. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 708–719, New Orleans, Louisiana. Association for Computational Linguistics.

Karl Moritz Hermann, Tomás Kociský, Edward Grefenstette, Lasse Espeholt, Will Kay, Mustafa Suleyman, and Phil Blunsom. 2015. Teaching machines to read and comprehend. In *NIPS*, pages 1693–1701.

Clément Jumel, Annie Louis, and Jackie Chi Kit Cheung. 2020. TESA: A Task in Entity Semantic Aggregation for abstractive summarization. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 8031–8050, Online. Association for Computational Linguistics.

Panagiotis Kouris, Georgios Alexandridis, and Andreas Stafylopatis. 2019. Abstractive text summarization based on deep learning and semantic content generalization. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 5082–5092, Florence, Italy. Association for Computational Linguistics.

Emiel Krahmer and Kees van Deemter. 2012. Computational generation of referring expressions: A survey. *Computational Linguistics*, 38(1):173–218.

Wojciech Kryściński, Bryan McCann, Caiming Xiong, and Richard Socher. 2020. Evaluating the factual consistency of abstractive text summarization. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 9332–9346, Online. Association for Computational Linguistics.

Wojciech Kryściński, Romain Paulus, Caiming Xiong, and Richard Socher. 2018. Improving abstraction in text summarization. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 1808–1817, Brussels, Belgium. Association for Computational Linguistics.

Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Veselin Stoyanov, and Luke Zettlemoyer. 2020. BART: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7871–7880, Online. Association for Computational Linguistics.

Chin-Yew Lin. 2004. ROUGE: A package for automatic evaluation of summaries. In *Text Summarization Branches Out*, pages 74–81, Barcelona, Spain. Association for Computational Linguistics.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Roberta: A robustly optimized BERT pretraining approach. *CoRR*, abs/1907.11692.

Joshua Maynez, Shashi Narayan, Bernd Bohnet, and Ryan McDonald. 2020. On faithfulness and factuality in abstractive summarization. In *Proceedings of the 58th Annual Meeting of the Association for*

Computational Linguistics, pages 1906–1919, Online. Association for Computational Linguistics.

Ines Montani, Matthew Honnibal, Matthew Honnibal, Sofie Van Landeghem, Adriane Boyd, Henning Peters, Paul O'Leary McCann, Maxim Samsonov, Jim Geovedi, Jim O'Regan, Duygu Altinok, György Orosz, Søren Lind Kristiansen, Lj Miranda, Daniël De Kok, Roman, Explosion Bot, Leander Fiedler, Grégory Howard, Edward, Wannaphong Phatthiyaphaibun, Yohei Tamura, Sam Bozek, Murat, Ryn Daniels, Mark Amery, Björn Böing, Bram Vanroy, and Pradeep Kumar Tippa. 2022. spaCy: Industrial-strength Natural Language Processing in Python.

Feng Nan, Ramesh Nallapati, Zhiguo Wang, Cicero Nogueira dos Santos, Henghui Zhu, Dejiao Zhang, Kathleen McKeown, and Bing Xiang. 2021a. Entity-level factual consistency of abstractive text summarization. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*, pages 2727–2733, Online. Association for Computational Linguistics.

Feng Nan, Cicero Nogueira dos Santos, Henghui Zhu, Patrick Ng, Kathleen McKeown, Ramesh Nallapati, Dejiao Zhang, Zhiguo Wang, Andrew O. Arnold, and Bing Xiang. 2021b. Improving factual consistency of abstractive summarization via question answering. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 6881–6894, Online. Association for Computational Linguistics.

Shashi Narayan, Shay B. Cohen, and Mirella Lapata. 2018. Don't give me the details, just the summary! topic-aware convolutional neural networks for extreme summarization. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 1797–1807, Brussels, Belgium. Association for Computational Linguistics.

Shashi Narayan, Yao Zhao, Joshua Maynez, Gonçalo Simões, and Ryan T. McDonald. 2021. Planning with entity chains for abstractive summarization. *CoRR*, abs/2104.07606.

Weizhen Qi, Yu Yan, Yeyun Gong, Dayiheng Liu, Nan Duan, Jiusheng Chen, Ruofei Zhang, and Ming Zhou. 2020. ProphetNet: Predicting future n-gram for sequence-to-SequencePre-training. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 2401–2410, Online. Association for Computational Linguistics.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J. Liu. 2020. Exploring the limits of transfer learning with a unified text-to-text transformer. *Journal of Machine Learning Research*, 21(140):1–67.

Sandhaus, Evan. 2008. *The New York Times Annotated Corpus*. In *Philadelphia: Linguistic Data Consortium. LDC2008T19. DVD.*

Eva Sharma, Luyang Huang, Zhe Hu, and Lu Wang. 2019. An entity-driven framework for abstractive summarization. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 3280–3291, Hong Kong, China. Association for Computational Linguistics.

Kaiqiang Song, Bingqing Wang, Zhe Feng, Liu Ren, and Fei Liu. 2020. Controlling the amount of verbatim copying in abstractive summarization. In *Proceedings of the AAAI Conference on Artificial Intelligence*.

Matthew Stone. 2000. On identifying sets. In *Proceedings of the First International Conference on Natural Language Generation - Volume 14*, INLG '00, page 116–123, USA. Association for Computational Linguistics.

Hardik Vala, Andrew Piper, and Derek Ruths. 2016. The more antecedents, the merrier: Resolving multi-antecedent anaphors. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 2287–2296, Berlin, Germany. Association for Computational Linguistics.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Proceedings of the 31st International Conference on Neural Information Processing Systems*, NIPS'17, page 6000–6010, Red Hook, NY, USA. Curran Associates Inc.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Remi Louf, Morgan Funtowicz, Joe Davison, Sam Shleifer, Patrick von Platen, Clara Ma, Yacine Jernite, Julien Plu, Canwen Xu, Teven Le Scao, Sylvain Gugger, Mariama Drame, Quentin Lhoest, and Alexander Rush. 2020. Transformers: State-of-the-art natural language processing. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 38–45, Online. Association for Computational Linguistics.

Juntao Yu, Nafise Sadat Moosavi, Silviu Paun, and Massimo Poesio. 2020. Free the plural: Unrestricted split-antecedent anaphora resolution. *CoRR*, abs/2011.00245.

Jingqing Zhang, Yao Zhao, Mohammad Saleh, and Peter Liu. 2020. PEGASUS: Pre-training with extracted gap-sentences for abstractive summarization. In *Proceedings of the 37th International Conference on Machine Learning*, volume 119 of *Proceedings of Machine Learning Research*, pages 11328–11339. PMLR.

Tianyi Zhang, Varsha Kishore, Felix Wu, Kilian Q. Weinberger, and Yoav Artzi. 2019. Bertscore: Evaluating text generation with BERT. *CoRR*, abs/1904.09675.

Zheng Zhao, Shay B. Cohen, and Bonnie Webber. 2020. Reducing quantity hallucinations in abstractive summarization. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 2237–2249, Online. Association for Computational Linguistics.

Hao Zhou, Weidong Ren, Gongshen Liu, Bo Su, and Wei Lu. 2021. Entity-aware abstractive multi-document summarization. In *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*, pages 351–362, Online. Association for Computational Linguistics.

Appendix A Aggregations and system output

Person	Location	Organization
(Joseph Lieberman, Ned Lamont) –> politicians	(Israel, Lebanon) –> middle eastern countries	(Airbus, Boeing) –> transportation companies
(Ehud Olmert, Mahmoud Abbas) –> politicians	(Iran, Iraq) –> neighboring countries	(London Stock, Nasdaq Stock) –> stock markets
(Charlie McDermott, Kris Kristofferson) –> actors	(Ethiopia, Somalia) –> african countries	(Microsoft, Google) –> technology companies
(Barry Diller, Frank Gehry) –> americans	(China, North Korea) –> asian countries	(Altimo, Telenor) –> companies

Table 9: Several examples among the most frequent (entities, aggregation) pairs for each type in TESA.

CC+PS	CC+IS	CC+NS
Internet giants signed up Tuesday to a "zero tolerance" approach to images of child sexual abuse as the British government announced a new, tougher strategy to find and block illegal content. Google, Yahoo, Microsoft, Twitter and Facebook were among the firms summoned to a meeting on the issue at 10 Downing Street, the prime minister's residence, by the UK government's Department for Culture, Media and Sport.	Internet giants signed up Tuesday to a "zero tolerance" approach to images of child sexual abuse as the British government announced a new, tougher strategy to find and block illegal content. Google, Yahoo, Microsoft, Twitter and Facebook were among the firms summoned to a meeting on the issue at 10 Downing Street, the prime minister's residence, by the UK government's Department for Culture, Media and Sport.	The launch of the lifeboat by William and Catherine and, at the same time, the launch of William and Catherine into this celebrity saturated world they are going to be living in. Despite the modest nature of the event, hundreds of people turned out to watch the royal couple conduct their first official duty together.
CC+PL	**IS+PS**	**IS+IS**
Last month, Inter was fined $65,500 by the Italian football authorities after its fans were found guilty of racially abusing former players Mario Balotelli and Sulley Muntari, who now play for rival AC Milan.	Some of the candidates have watched the video. Vazquez Mota, of the ruling National Action Party, said the video's message can't go unnoticed, while Institutional Revolutionary Party candidate Pena Nieto expressed that now is the time for change, as the video suggests.	As advertising losses and new reader habits afflicted newspapers nationwide, The Times began looking to shed The Globe and even threatened to close the paper in 2009 amid disputes with unions.
IS+NS	**CS+PS**	**CS+IS**
When asked what would happen if he rapped his anti-regime lyrics prior to Libyaś uprising, MC Swat said, "I would be shot to death like Tupac," referring to the American rapper killed in 1996.But here in Benghazi, the oppositionś de facto capital, thereś no sign of Gadhafiś loyalists anymore – or the fear that kept artists like MC Swat quiet for so long.	All three suspects are facing a charge of capital murder with the intent to sell a controlled substance, Lindley said. Trent Deundra Crump turned himself in to authorities of Alachua County Sheriff's Department in Gainesville, Florida, Lindley said. Duntae Harvey, 21, was arrested Monday and was being transferred Tuesday from Rankin County, where he has been held, university officials said. Mason Perry Jones, 21, of Jackson was arrested Monday in Memphis by members of the U.S. Marshal's Fugitive Task Force, Lindley said.	South Dakota, which has sent one inmate to death in three decades, has scheduled a lethal injection in October. Nebraska is the only state that does not use lethal injection, but its use of the electric chair was ruled unconstitutional in February. Texas and Mississippi are among the states that use 2 grams of sodium thiopental, the anesthetic used to render condemned inmates unconscious.
CS+NS	**SN+PE**	
Alexis was not the first mass killer to have an obsession with violent video games. Adam Lanza, who killed 26 children in an elementary school in Newtown, Connecticut, was also said to be a fan of first-person shooting games. Other killers have been found to be avid players.	Ivanovic's new team includes coach and hitting partner Nemanja Kontic – who represented Montenegro in the Davis Cup – fitness coach[es] Zlatko Novkovic and physio Branko Penic. They have all been part of her entourage since her split with British coach[es] Nigel Sears in July, following a second-round exit at Wimbledon.	

Table 10: Alignments extracted by the heuristics in different examples from CNN/DAILYMAIL. The names are the acronyms of the patterns used to detect entities and aggregations.

	P	R	F_1	#samples
Aggregation	51.31	66.58	57.96	2855
Not aggregation	87.75	79.11	83.29	8635
Macro-Avg	69.53	72.85	70.58	11490
Accuracy	-	-	76.00	11490

Table 11: Results of the classifier of Gating BARTs, per class and macro-averaged, on the test set of CNN/DAILYMAIL.

Table 12: Six examples of summarization using BART and BART+AggregationChains on the NOVEL and COPY test sets. The aggregations of the summaries are in green. They are also marked in the source if they are COPY aggregations. The entities of the document being aggregated (blue) are also shown, if they appear in the source, for visualization purposes.

Example 1 (NOVEL)

Document: (CNN)That sound you just heard was the crash of hearts breaking all over the world. Zayn Malik is leaving One Direction. "After five incredible years Zayn Malik has decided to leave One Direction," the band said on its Facebook page and tweeted out to its 22.9 million Twitter followers. "Niall, Harry, Liam and Louis will continue as a four-piece and look forward to the forthcoming concerts of their world tour and recording their fifth album, due to be released later this year." Rumors about such a move had started since Malik left the band's tour last week. At the time, a rep told Rolling Stone he had "been signed off with stress" after a scandal erupted following the publication of a photo showing Malik holding hands with someone other than his fiancee. Fans on Twitter immediately responded with teary Vine videos and the #AlwaysInOurHeartsZaynMalik hashtag. Even the Girl Scouts got in on the act with a sweet tribute to the singer. The band's Wikipedia page was also quickly updated with a sentence, "Zayn Malik was formerly a member." And Spotify said that global streams of One Direction songs were up 330% Wednesday in the hour after the news was announced – a "spike of sadness," as the music service called it. In the U.S. alone, streams of the band's music were up 769%. To mark the occasion, Spotify created a special playlist of 1D songs. Malik, 22, has been part of the very popular British boy band since it was formed (at the urging of Simon Cowell, according to some stories) in 2010 after members auditioned separately for the UK version of "The X Factor." Cowell became a mentor and signed them to his label. The group has put out four albums, and its hits include "Best Song Ever" and "Story of My Life." Their latest album, "Four," came out in November.

Reference summary: Malik had left the band's tour last week, citing "stress" One Direction is one of the most popular bands in the world .

BART: Zayn Malik is leaving One Direction. Rumors about such a move had started since Malik left the band's tour last week. Fans on Twitter immediately responded with teary Vine videos and the #AlwaysInOurHeartsZaynMalik hashtag. Spotify said that global streams of One Direction songs were up 330% Wednesday.

BART+AggregationChains: Zayn Malik is leaving One Direction . He has been part of the british boy band since it was formed . Spotify says global streams of One Direction songs were up 330% Wednesday in the hour after the news was announced .

Example 2 (NOVEL)

Document: (CNN)Comic books of the past few years have seen a lot of changes (a female Thor, anyone?) but not quite so many at one time. Three major characters – Superman, Wonder Woman (both of DC Comics, a Time Warner company, like CNN) and Archie Andrews – came out with new looks (and costumes in two cases) Thursday. Superman and Wonder Woman are no stranger to change over time, but these are pretty different from what we're used to. Wonder Woman is back to wearing pants (similar to her style circa 2010, not to mention a brief time in the 1970s) along with something of a turtleneck and body armor. Superman looks the most casual that we've ever seen him, simply in a t-shirt and jeans, and decidedly shorter hair. He also looks like someone you might not want to come across in a dark alley. As for Archie (who never really died, by the way), he's getting ready for his promised TV show by debuting a modern look – one that makes it much easier to see why Betty and Veronica have been fighting for his affections all of these years. It's been a whirlwind time for comic book fans, and there will be a lot to get used to.

Reference summary: Superman, Wonder Woman and Archie all debuted new looks Thursday . Three of the most well-known comic book characters of all time look radically different .

BART: Superman, Wonder Woman and Archie Andrews have new looks . It's been a whirlwind time for comic book fans, and there will be a lot to get used to . For more comic book news, visit CNN.com/comic-book .

BART+AggregationChains: Three major characters – Superman, Wonder Woman and Archie Andrews – came out with new looks (and costumes in two cases) Superman looks the most casual that we've ever seen him, simply in a t-shirt and jeans. Archie (who never really died, by the way) is getting ready for his promised TV show.

Example 3 (NOVEL)

Document: (CNN)Italian authorities have arrested 42 people suspected of belonging to a mafia network in southern Italy, some of whom are accused of extorting tens of thousands of euros from businesses monthly, police said Tuesday. Among the evidence: A piece of paper, hidden in a bicycle handlebar, that listed businesses that the crime syndicate was extorting, according to the Carabinieri, Italy's military police. The Carabinieri said they made the arrests while raiding homes in 11 provinces. The crime network is based in Casal di Principe, a village roughly 20 kilometers (12 miles) northwest of Naples, police said. The mafia network extorted shop owners to the tune of 200,000 euros a month and extorted operators of slot machines and online gambling a further 100,000 euros monthly, the Carabinieri said. The arrests stem from an investigation that started in October 2012. Investigators also found notebooks with the group's organizational chart and names of jailed associates who still were on the mafia payroll, according to the Carabinieri. The note in the bicycle handlebar was among other "pizzini," or little pieces of paper allegedly used by mafia members to exchange information, that investigators discovered in the probe, the military police said. In Villa Literno, close to Casal di Principe, investigators discovered a bunker that allegedly was used to hide fugitive affiliates of the mafia, the Carabinieri said. An anti-mafia prosecutor in Naples filed charges against the 42, including mafia association, extortion and illegal possession of weapons.

Reference summary: Police say a note hidden in a bike's handlebar listed extorted businesses . The crime network is based in Casal di Principe, a village near Naples, police say . Arrests stemmed from investigation that began in 2012 .

BART: Italian authorities arrest 42 people suspected of belonging to a mafia network in southern Italy. A piece of paper, hidden in a bicycle handlebar, listed businesses that the crime syndicate was extorting. The crime network is based in Casal di Principe, a village roughly 12 miles northwest of Naples.

BART+AggregationChains: A piece of paper, hidden in a bicycle handlebar, listed businesses that the crime syndicate was extorting . The crime network is based in Casal di Principe, a village roughly 20 kilometers (12 miles) northwest of Naples .

Example 4 (COPY)

Document: A game of cat and mouse has been captured in a series of striking images as the pair battle it out on a shed rooftop like a real life version of much-loved cartoon duo Tom and Jerry. It is an age-old rivalry that rarely ends well for one of its parties and so it proved in this remarkable set of photos. The snaps of a cat playing with a mouse on a roof in Shepton Mallet, Somerset, illustrate the perils the tiny rodents face in the town. Ironically the pet cat's name is Mouse. Unfortunately for this mouse that's where all similarities between the moggy and its namesakes end. The pictures were taken by the cat's owner Jason Bryant who confirmed the inevitable outcome of the encounter. 'My cat is a very good mouser,' he said. 'She's done it before. She often brings them into the house at 3am and then I can rescue them. But she took this poor little thing on to the shed roof where I couldn't reach it.' Ready to pounce: The mouse flies through the air as its feline nemesis looks to bring it back down to earth with a bang . Eeek: A mouse tries to escape the clutches of a pet cat named Mouse as the pair engage in the age-old game of cat and mouse . In flight: The mouse, unable to evade its moggy hunter, is thrown into the air on the roof of a shed in Shepton Mallet in Somerset . Somersaulting away: The snaps illustrate the perils that tiny rodents face in the town with cats like Mouse around . Cartoon characters: The pair battle it out on a rooftop like a real life version of much-loved cartoon duo Tom and Jerry . Let's play: The pictures were taken by the cat's owner Jason Bryant who confirmed the inevitable outcome of the encounter . Reaching out: The mouse is flung high in the sky as it finally admits defeat in the game of cat and mouse .

Reference summary: Age-old game of cat and mouse is brought to life in these quirky pictures taken in Shepton Mallet in Somerset . The pair are seen battling it out on the roof of a shed in a real life take on an episode of Tom and Jerry . Ironically, the cat's name is Mouse. The pictures show the dangers small rodents have to be aware of in the area .

BART: The snaps of a cat playing with a mouse on a roof in Shepton Mallet, Somerset, illustrate the perils the tiny rodents face in the town. Ironically the pet cat's name is Mouse. The pictures were taken by the cat's owner Jason Bryant who confirmed the inevitable outcome of the encounter.

BART+AggregationChains: The snaps of a cat playing with a mouse on a roof in Shepton Mallet, Somerset, illustrate the perils the tiny rodents face in the town . Unfortunately for this mouse that's where all similarities between the moggy and its namesakes end . The pictures were taken by the cat's owner who confirmed the inevitable outcome of the encounter .

Example 5 (COPY)

Document: (Billboard)Fresh off his scorching performance at Coachella Saturday night (and days before his next one on the festival's second weekend), rocker Jack White announced he'll take a hiatus from touring. White will wrap his touring efforts in support of "Lazeretto" with a brief, first-ever acoustic tour that will hit "the only five states left in the U.S. that he has yet to play," according to White's website. Rounding out the acoustic quartet on tour will be Fats Kaplin, Lillie Mae Rische and Dominic Davis. The shows will be unannounced until day-of-show, with tickets priced at $3 and limited to one ticket per person, to be purchased only at the venue on a first-come, first-served basis. Billboard: Jack White on Not Being a 'Sound-Bite Artist,' Living in the Wrong Era and Why Vinyl Records Are 'Hypnotic' The purposely vague announcement surely has fans (and journalists) scouring the Internet for White's touring history. Unclear is whether White includes his work with The White Stripes, The Raconteurs and Dead Weather in his touring history, or just his solo road work. Presumably, he's including all of his touring, with all bands, as Billboard could find only 29 states in which he has performed as Jack White. Tour dates with White Stripes add another 12 states. That leaves nine states for which we could not find a show for White: Hawaii (where a show is scheduled for tomorrow, April 15), Arkansas, Idaho, Utah, Wyoming, Vermont, Iowa, and North and South Dakota. Billboard: Jack White Plays The Hits, Declares 'Music Is Sacred' at Coachella . Through the process of elimination (surely he has played Boise, Little Rock, and Salt Lake?), our guess as to which five states White will play on the brief acoustic run: South and North Dakota, Wyoming, Vermont and ... Puerto Rico? If that's the case, this tour is in for some long jumps, with Puerto Rico to Vermont being a potential beast. (Though shipping acoustic instruments and ribbon mics will be a lot less taxing than a full electrified stage setup.) ©2015 Billboard. All Rights Reserved.

Reference summary: Jack White taking a hiatus from touring after brief acoustic jaunt . He'll play five states he has yet to get to, charge just $3 . Places and times of shows are currently a mystery .

BART: Jack White announced he'll take a hiatus from touring. White will wrap his touring efforts in support of "Lazeretto" with a brief, first-ever acoustic tour. The shows will be unannounced until day-of-show, with tickets priced at $3.

BART+AggregationChains: White will wrap his touring efforts in support of "Lazeretto" with a brief, first-ever acoustic tour . He will hit "the only 5 states left in the U.S. that he has yet to play," according to White's website . The shows will be unannounced until day-of-show, with tickets priced at $3 .

Example 6 (COPY)

Document: Boss Nigel Pearson has urged Leicester to keep their cool and ignore their relegation rivals. The Foxes host Swansea on Saturday just three points from safety in the Barclays Premier League after back-to-back wins. Last week's 3-2 win at West Brom handed them a survival lifeline, although they remain bottom of the table. Jamie Vardy scored an injury-time winner against West Bromwich Albion on Saturday to improve his side's slim chance of Premier League survival . Vardy celebrates in front of the travelling away fans after hitting the winner against West Brom . But after their mini-revival, Pearson wants his side to remain focused on their own jobs. 'I'm very wary of people flipping the emphasis,' he said. 'Our future is in our own hands and if we go into the last game with that we have given ourselves a realistic chance. 'We need to make sure our own run-in is what we want it to be. Leicester manager Nigel Pearson has urged his players to focus on their own job and not worry about their relegation rivals during run-in . 'It's helpful that other teams are being dragged into it. People are continually adjusting the form table and their predictions. 'We can't get involved in anything apart from being involved in our own games that may change people's perceptions. 'It's great when results go your way but they only help if you've done your own job. 'It's important for us to be in touch. One of the potential problems we have to deal with is people are now overly optimistic.'

Reference summary: Leicester have won back-to-back league games to boost survival hopes . Nigel Pearson has urged his players to focus on their own run-in . Leicester now just three points from safety heading into final six games .

BART: Nigel Pearson has urged Leicester to ignore their relegation rivals. The Foxes host Swansea on Saturday just three points from safety. Last week's 3-2 win at West Brom handed them a survival lifeline. But Pearson wants his side to remain focused on their own jobs.

BART+AggregationChains: Nigel Pearson has urged Leicester to keep their cool . The Foxes host Swansea on Saturday just three points from safety in the Barclays Premier League after back-to-back wins . But after their mini-revival, Pearson wants his side to remain focused on their own jobs .

Appendix B Notes on the heuristics

Through a preliminary evaluation, we observed that **Coordinating Conjunction+Preceding a list** (**CC+PL**) and **Shared noun+Preceding Entities** (**SN+PE**) stand out in terms of precision. Especially, **CC+PL** has almost 100% precision and appears in 3.87% of the documents in the CNN/DAILYMAIL corpus (13,959 alignments). **SN+PE** is less frequent than **CC+PL** (1.27% of documents and 4125 alignments). The other heuristics ranges from 20% to 80% of precision, and some of them such as **In Sentence+In span** seems to have a high recall (25.74% of documents and 132,922 alignments).

How to Find Strong Summary Coherence Measures? A Toolbox and a Comparative Study for Summary Coherence Measure Evaluation

Julius Steen Katja Markert
Department of Computational Linguistics
Heidelberg University
69120 Heidelberg, Germany
(steen|markert)@cl.uni-heidelberg.de

Abstract

Automatically evaluating the coherence of summaries is of great significance both to enable cost-efficient summarizer evaluation and as a tool for improving coherence by selecting high-scoring candidate summaries. While many different approaches have been suggested to model summary coherence, they are often evaluated using disparate datasets and metrics. This makes it difficult to understand their relative performance and identify ways forward towards better summary coherence modelling. In this work, we conduct a large-scale investigation of various methods for summary coherence modelling on an even playing field. Additionally, we introduce two novel analysis measures, *intra-system correlation* and *bias matrices*, that help identify biases in coherence measures and provide robustness against system-level confounders. While none of the currently available automatic coherence measures are able to assign reliable coherence scores to system summaries across all evaluation metrics, large-scale language models fine-tuned on self-supervised tasks show promising results, as long as fine-tuning takes into account that they need to generalize across different summary lengths.

1 Introduction

Automatically generated summaries should not only be informative, but also well-written and coherent. While informativeness is routinely evaluated automatically with ROUGE (Lin, 2004), there is no agreement on how to evaluate summary coherence. However, automatic evaluation is highly desirable to reduce evaluation costs and as a tool for improving summarizer output, e.g. as reranker.

Many coherence measures **(CMs)** have been suggested for automatically assigning a coherence score to a summary, including learning from human judgements (Barzilay and Lapata, 2008; Tien Nguyen and Joty, 2017; Xenouleas et al., 2019; Mesgar et al., 2021), learning from the *shuffle task* (Mohiuddin et al., 2021; Jwalapuram et al., 2022), where models are trained to discriminate original documents from documents with randomized sentence order (Barzilay and Lapata, 2008), using next sentence prediction as a proxy task (Koto et al., 2022), and finally unsupervised measures that exploit heuristics (Pitler et al., 2010; Zhu and Bhat, 2020) or large-scale LMs (Yuan et al., 2021). CM performance is then evaluated by comparing the automatic scores to human coherence scores on a set of system summaries.

However, this evaluation is often conducted on disparate datasets. It also often uses system outputs from DUC conferences (Barzilay and Lapata, 2008; Tien Nguyen and Joty, 2017; Xenouleas et al., 2019; Mesgar et al., 2021), which do not necessarily represent recent advances in text summarizers. In addition, there is no agreement on *how* the CM scores should be compared to human scores. System-level correlation (Xenouleas et al., 2019; Fabbri et al., 2021), pairwise ranking accuracy (Barzilay and Lapata, 2008; Tien Nguyen and Joty, 2017; Mesgar et al., 2021) and summary-level correlation (Yuan et al., 2021) have all been suggested as evaluation metrics **(EMs)**.

This makes it hard to ascertain the state of summary coherence modelling and to identify promising directions for future research. We attack this problem by making the following contributions:

- We show that current EMs provide an incomplete picture of CM performance as they focus on comparing summaries generated by different summarizers, which includes many easy decisions due to the large performance gaps between them. Additionally, they are vulnerable to CMs exploiting confounding system properties to correctly rank systems without modelling coherence.
- We introduce a new EM, *intra-system correlation*, that measures performance within the summaries generated by a single summarizer

Proceedings of the 29th International Conference on Computational Linguistics, pages 6035–6049
October 12–17, 2022.

and is both more challenging and more resilient against system-level confounders.

- We introduce *bias matrices* as a novel analysis tool that allow to easily detect when CMs are biased towards specific summarizers.

Using these insights, we conduct a large-scale comparison of CMs on the recent *SummEval* dataset (Fabbri et al., 2021). We find that:

- All investigated CMs exhibit significant weaknesses under evaluation regimes other than system-level correlation.
- Even relatively strong CMs are biased towards outputs of certain summarizers, which raises concern about their generalizability.
- SummEval is not conducive to entity-based modelling, which has been successful on many other coherence tasks (Barzilay and Lapata, 2008; Elsner and Charniak, 2011; Tien Nguyen and Joty, 2017; Mesgar et al., 2021).
- While most of the shuffle-based models transfer poorly to summaries, which is in line with prior results by Mohiuddin et al. (2021), the most promising performance is achieved by fine-tuning a masked language model (MLM) on the shuffle task as a *classifier*. We present evidence that this allows the model to adapt more easily to comparing documents of different content and lengths, highlighting a possible avenue for future work.

Code and data for our experiments are available at https://github.com/julmaxi/summary_coherence_evaluation.

2 Related Work

2.1 Coherence Measures for Summarization

Automatic coherence assessment for summarization has been studied in a variety of settings. Barzilay and Lapata (2008) establish summary coherence as an *evaluation* task to assess CMs similarly to other downstream tasks such as essay scoring (Jeon and Strube, 2020, among others) and readability assessment (Mesgar and Strube, 2015, among others). Specifically, Barzilay and Lapata acquire coherence labels for human and system summaries from DUC 2003[1]. The same dataset has been used for evaluating subsequent CMs (Tien Nguyen and Joty, 2017; Mesgar et al., 2021).

As a part of *automatic linguistic quality estimation*, summary coherence is modelled alongside other aspects of text quality such as grammaticality, with the direct goal of aiding automatic summary evaluation. Approaches include regression models learned from human annotations (Xenouleas et al., 2019) as well as unsupervised approaches (Pitler et al., 2010; Zhu and Bhat, 2020; Yuan et al., 2021). Datasets used for evaluation include the recent SummEval dataset (Fabbri et al., 2021), assessor judgements from DUC05-07, and the small-scale manually annotated summaries of newsroom (Grusky et al., 2018). In parallel work, Koto et al. (2022) introduce a coherence measure based on a next sentence prediction task as part of a wider set of measures for summary evaluation that also include focus, coverage and faithfulness. For evaluation, they introduce a novel small-scale dataset based on outputs from BART (Lewis et al., 2020) and the pointer generator model (See et al., 2017). CMs have also been applied in a related setting to improve summarizer quality by explicitly modelling coherence during the summary optimization process (Parveen et al., 2017; Sharma et al., 2019).

Finally, summary coherence is also sometimes modelled using measures that make use of human-written reference summaries (Fabbri et al., 2021; Zhao et al., 2022). We do not focus on these CMs in our evaluation since they are fundamentally less flexible than reference-free CMs, especially when used in non-evaluation contexts such as reranking.

We provide a detailed description of the CMs used in our study in Section 5.

2.2 Meta-Evaluation

In terms of evaluation studies, Mohiuddin et al. (2021) conduct a comparative study of five CMs. Their evaluation is conducted on 10 summaries each from 4 recent summarizers as well as the DUC03 data. Unlike our study, their investigation only encompasses CMs trained via the shuffle task and includes only a small number of summaries.

In concurrent work on assessing system-level correlation, Deutsch et al. (2022) propose to modify correlation computation by focusing on difficult system comparisons only and computing measure scores on a larger set of summaries. Their approaches are complimentary to our analysis in that they look at informativeness instead of co-

[1] https://duc.nist.gov

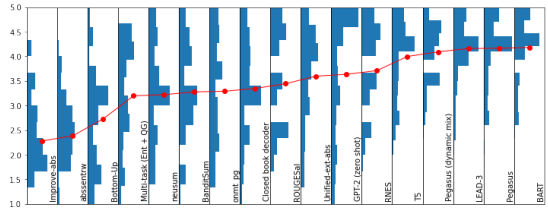

Figure 1: Distribution of human coherence scores for the 17 systems in the SummEval dataset. The red dots indicate the mean score of each system.

herence and do not address the shortcomings of system-level correlation in the presence of system level confounders. Also concurrently, Durmus et al. (2022) identify spurious correlates in faithfulness datasets, which suggests that our methods might be useful beyond coherence evaluation.

3 Dataset

We evaluate CMs on the expert annotations in the SummEval dataset[2] (Fabbri et al., 2021) which is, to the best of our knowledge, the largest dataset that includes such coherence annotations for a variety of state-of-the-art summarizers. It contains annotations on a 1-5 scale for outputs of 17 systems for 100 documents from the CNN/DM dataset (Hermann et al., 2015) by three annotators each. Figure 1 highlights two important properties. Firstly, there is a large gap in average performance between different summarizers, and secondly, most summarizers exhibit considerable variance in scores.

4 Evaluating CMs

CM performance is typically assessed on a set of summaries generated on document set D by a set of summarizers S using the agreement of predicted scores $P = \{P_{(d,s)}|s \in S, d \in D\}$ with human judgements $H = \{H_{(d,s)}|s \in S, d \in D\}$. However, this agreement can be computed in different ways. We identify the following common EMs:

System-level Correlation τ_{sys} assesses CM performance by correlating the *mean* human and *mean* CM scores of the individual summarizers.

Pairwise Accuracy Acc_{pair} assesses CM performance by comparing scores on outputs of two different systems on the same document.

Summary-level Correlation τ_{sum} compares scores on all generated summaries.

[2] https://github.com/Yale-LILY/SummEval

The correlation function used is usually[3] Kendall's τ, while the pairwise metric is usually reported as accuracy. However, we can also define the latter in terms of average τ over all documents. This is equivalent to accuracy when there are no tied scores, with the only difference being the range shift from $[0, 1]$ to $[-1, 1]$. As both τ and accuracy are based on pairwise rankings, we can specify all three EMs in terms of the set of pairwise comparisons C they consider, where $C_{sys} \subset 2^{S \times S}$ considers comparisons between averaged system scores and $C_{pair}, C_{sum} \subset 2^{(D \times S) \times (D \times S)}$ consider comparisons between individual summary scores.

$$C_{sys} = \{(s_i, s_j)|s_i \neq s_j\}$$
$$C_{pair} = \{((d, s_i), (d, s_j))|s_i \neq s_j\}$$
$$C_{sum} = \{((d_k, s_i), (d_l, s_j))|(d_k, s_i) \neq (d_l, s_j)\}$$

The EMs pose different demands to CMs: system-level correlation requires a correct ranking of systems according to their average score. Pairwise accuracy requires correct ranking of summaries from different systems but only between summaries produced on the same document. Finally, summary-level correlation requires the correct ranking of any pair of summaries.

4.1 A new EM: Intra-System Correlation

All three EMs focus on comparisons between summaries generated by different summarizers. For system-level and pairwise evaluation this arises by construction, whereas for summary-level correlation it is contingent on the dataset structure: On SummEval, less than 6% of comparisons for τ_{sum} are between summaries of the same summarizer. We argue that this gives an incomplete view of CM performance for the following reasons:

1. SummEval covers summarizers with widely different performance levels (see Figure 1), leading current EMs to include many easy decisions. This is unlikely to reflect real-world evaluation of competitive summarizers.

2. While system-level evaluation is often the primary use case, CMs can also be used in a reranking or ensembling context to select the most coherent summary from a set of candidates. In these situations, summaries are likely

[3] While Spearmans ρ is also sometimes used (Yuan et al., 2021), the τ variant we use, τ-b, is more robust to ties, which are common on the five-point rating scale in SummEval.

to be generated either by a the same summarizer or a set of similarly (high) performing summarization systems. In these cases, system-level EMs offer only limited insight into likely CM performance, since they primarily measure the ability to discriminate between *different* systems with potentially large performance gaps.

3. EMs might not correlate with coherence per se but instead with features that happen to identify good summarizers on a particular dataset. Such *system-level confounders* are unlikely to generalize to new systems and settings. We elaborate on this in Section 4.2.

We thus suggest adding a new EM **Intra-system Correlation** τ_{intra}, which we define on comparisons between summaries generated by the same system. This corresponds to considering the following pairs $C_{intra} \subset 2^{(D\times S)\times(D\times S)}$:

$$C_{intra} = \{((d_k, s), (d_l, s))|d_k \neq d_l\}$$

It neatly complements pairwise accuracy, as it is essentially the same computation but keeping the summarizer constant instead of the document. Intuitively, this measure both contains far fewer "easy" decisions and is much more resilient to any system-level confounders in the data. We use the average of the intra-system correlation of all systems as the correlation measure.

4.2 System-level Confounders

To assess how EMs behave in the presence of system-level confounders, we investigate two summary features that are unlikely to be generalizable CMs but lead to surprisingly strong correlations: Capitalization and summarizer architecture.

For capitalization, we count the number of uppercase letters in each summary. This is a purely system-level heuristics, since only three of the 17 summarizers in SummEval produce capital letters[4]. For architecture, we assign a score of 1 to each summary from one of the five summarizers that are derived from pretrained transformers in some fashion[5] and 0 to all others. Neither of the two confounders can, by construction, be a reasonable and generalizable CM. Additionally, we compute an "upper-bound" (UB) that assigns to each summary the mean human score of the system that produced the summary. It simulates perfect system ranking, but no ability to correctly rank summaries within each system. Since these procedures result in many ties, we also compute a second variant of each confounder where we add small noise to each score. This prevents τ-b from profiting from these ties, while preventing accuracy from unfairly suffering.

	Cap.	Cap. (r)	Arch.	Arch. (r)	UB	UB (r)
τ_{sys}	0.42	0.23	0.58	0.37	1.00	1.00
τ_{sum}	0.19	0.11	0.31	0.20	0.39	0.39
τ_{pair}	0.21	0.14	0.33	0.22	0.44	0.44
Acc_{pair}	0.23	0.57	0.34	0.62	0.73	0.73
τ_{intra}	-	-0.03	-	0.01	-	0.00

Table 1: Results for the confounders and upper bound. τ_{intra} for the non-random variants is undefined, as scores within each system are constant. Scores for the random variants (r) are averaged over 100 runs.

Table 1 shows the resulting correlations. Confounders achieve noticeable correlation with human scores. In particular, system-level correlation comes close to or exceeds the best CM reported originally for SummEval (CHRF (Popović, 2017), 0.40). In contrast, using intra-system correlation, the problems of these pseudo-measures become easily apparent. In practical scenarios, system-level correlation might be a mix of modelling coherence and reliance on confounders. Intra-system evaluation is an important tool in this context as it is robust to system-level confounders.

5 Coherence Measures

We identify the following families of reference-free CMs for summarization and include representatives of each in our study: **supervised CMs** trained on human coherence ratings of summaries, **self-supervised CMs trained on the shuffle task** and **unsupervised CMs**. For the supervised setting, we investigate measures trained on data from DUC03 (Barzilay and Lapata, 2008) as well as the DUC05-07 dataset used by Xenouleas et al. (2019). While the DUC03 dataset is set up as a pairwise ranking dataset, the DUC05-07 dataset is used in a regression setting. Table 2 indicates the configurations available for the different CMs.

The **Extended Entity Grid (`EEG`)** (Elsner and Charniak, 2011) is an extension of the Entity Grid of Barzilay and Lapata (2008). It represents texts using occurrence patterns of the mentioned entities across sentences. The model uses a generative approach that models the probability of an entity appearing in a specific role in a sentence, given its

[4]BART, GPT-2 (zero shot) and Pegasus (dynamic mix)
[5]BART, Pegasus, Pegasus dynamic Mix, T5 and GPT-2

	EEG	EGR	NEG	UNF	GRA	CCL	SQE	GRU	BAS
Unsupervised	✓ (a)	✓						✓	✓ (b)
Shuffle			✓	✓	✓	✓			
Supervised (DUC03)			✓		✓				
Supervised (DUC05-07)							✓		

Table 2: Training settings for the CMs under investigation. (a) The extended entity grid estimates the multinomial distribution of an entity's role given its prior occurrences. While this needs a dataset to estimate the distribution, it can not be trained as a classifier. (b) BART includes shuffling as a pretraining task.

role in the two preceding sentences.

The **Entity Graph (EGR)** (Guinaudeau and Strube, 2013) constructs a sentence graph of a document by identifying entity overlap between sentences. Two sentences are connected if they share at least one entity, with edge weights decreasing when they are further apart. The score of a document is the average outdegree of sentences, with higher outdegree indicating better coherence.

The **Neural Entity Grid (NEG)** (Tien Nguyen and Joty, 2017) applies a convolutional network to the entity grid. The model is trained on a pairwise ranking loss.

The **Unified Model (UNF)** (Moon et al., 2019) is a lexical CM that uses a convolutional network to build sentence representations from raw text. The model uses an adapted version of the ranking loss for the shuffle task that is computed only for three sentence windows in which shuffled and original documents differ. We use the model based on ELMo (Peters et al., 2018), as it performs best in the original paper.

The **Graph-based Neural Coherence Model (GRA)** (Mesgar et al., 2021) is a recent CM that combines entity-based representation with lexical information in a graph NN. Like the previous two models, it employs a pairwise ranking loss.

Recently, Laban et al. (2021) have shown that a RoBERTa-based (Liu et al., 2019) classifier can easily achieve near-perfect results on the shuffling task on WSJ. However, they did not test whether this model can predict coherence on non-artificial tasks. We thus include a simple RoBERTa model that is trained to classify shuffled vs. unshuffled summaries, naming it **Coherence Classifier (CCL)**.[6]

SumQE (SQE) (Xenouleas et al., 2019) predicts five linguistic quality scores via multi-head regression on human scores. We use the coherence head of the model trained on all three DUC datasets.[7]

GRUEN (GRN) (Zhu and Bhat, 2020) is a collection of unsupervised measures for linguistic quality that combines multiple unsupervised heuristics.

BARTScore (BAS) (Yuan et al., 2021) uses the probability of a summary under a pre-trained BART model as a score. We use the variant fine-tuned on CNN/DM summaries in the source-to-summary configuration, as suggested by the authors.

Finally, we include an upper and lower bound: RND assigns each summary a uniformly chosen score between 0 and 1. For HUM, we use the SummEval human annotations and select the annotator with the worst overall correlation to the remaining annotators and use their scores as predictions.[8]

We train all shuffling models on the WSJ corpus of newswire articles, which is frequently used in coherence modelling (Elsner and Charniak, 2011; Guinaudeau and Strube, 2013; Moon et al., 2019; Mohiuddin et al., 2021). We also train models using the same technique on reference summaries from the train portion of CNN/DM. For EEG we also estimate model parameters on both datasets. For WSJ, we follow the original implementations regarding the number of shuffled samples. For CNN/DM, we only use a single shuffled instance per summary, as it is larger by two orders of magnitude (WSJ: 1,400; CNN/DM: 287,113 documents before shuffling). Detailed accounts of our experiments with each CM are found in Appendix A.

6 Results

We present the correlation of all CMs with human coherence ratings in Table 3. We report (average) Kendalls τ for all EMs introduced in Section 4. For C_{pair} we additionally report accuracy. Per-system scores for intra-system correlation can be found in Appendix B.

Focusing on τ_{sys} first, we find that CCL, BAS, GRN and to a lesser extent SQE achieve relatively

[6] We found that the original WSJ-model does not perform well on SummEval. Thus, we retrained our own model, using the same RoBERTA checkpoint as a basis.

[7] https://archive.org/download/sum-qe/BERT_DUC_all_Q5_Multi%20Task-5.h5

[8] We note that unlike automatic measures, humans may only differentiate among five classes. We might thus underestimate actual human performance.

Metric	τ_{intra}	τ_{pair}	τ_{sum}	τ_{sys}	Acc.$_{pair}$
HUM	+0.75 (+0.70 +0.79)	+0.81 (0.76, 0.85)	+0.81 (+0.77 +0.84)	+0.91 (+0.71 +1.00)	+0.77 (+0.71 +0.81)
RND	-0.00 (-0.06 +0.05)	-0.00 (-0.07 +0.06)	+0.00 (-0.05 +0.05)	+0.09 (-0.41 +0.53)	+0.50 (+0.46 +0.54)
EGR	-0.04 (-0.12 +0.04)	-0.11 (-0.19, -0.02)	-0.09 (-0.16 -0.01)	-0.25 (-0.59 +0.10)	+0.40 (+0.36 +0.44)
EEG C/D	+0.02 (-0.07 +0.10)	+0.04 (-0.10 +0.18)	+0.06 (-0.06 +0.17)	-0.19 (-0.68 +0.26)	+0.52 (+0.45 +0.59)
EEG WSJ	+0.02 (-0.06 +0.10)	+0.00 (-0.09 +0.11)	+0.03 (-0.06 +0.11)	-0.19 (-0.60 +0.26)	+0.50 (+0.44 +0.55)
NEG C/D	-0.07 (-0.14 -0.00)	-0.05 (-0.14 +0.07)	-0.06 (-0.15 +0.03)	-0.15 (-0.61 +0.32)	+0.47 (+0.42 +0.53)
NEG DUC	-0.08 (-0.16 +0.01)	-0.06 (-0.18 +0.06)	-0.07 (-0.17 +0.04)	-0.06 (-0.49 +0.31)	+0.47 (+0.40 +0.53)
NEG WSJ	-0.02 (-0.08 +0.05)	-0.08 (-0.17 +0.00)	-0.07 (-0.15 +0.02)	-0.43 (-0.69 -0.05)	+0.45 (+0.41 +0.50)
UNF C/D	+0.04 (-0.03 +0.11)	+0.05 (-0.05 +0.14)	+0.06 (-0.01 +0.13)	+0.13 (-0.33 +0.59)	+0.53 (+0.48 +0.57)
UNF WSJ	+0.02 (-0.05 +0.09)	-0.11 (-0.26 +0.03)	-0.04 (-0.15 +0.05)	-0.09 (-0.51 +0.39)	+0.44 (+0.36 +0.52)
GRA DUC	-0.04 (-0.12 +0.03)	-0.05 (-0.16 +0.03)	-0.06 (-0.13 +0.01)	-0.19 (-0.65 +0.25)	+0.47 (+0.43 +0.52)
GRA C/D	+0.08 (+0.02 +0.15)	+0.09 (-0.02 +0.19)	+0.11 (+0.01 +0.18)	+0.37 (-0.07 +0.69)	+0.55 (+0.49 +0.60)
GRA WSJ	+0.08 (+0.01 +0.15)	-0.01 (-0.11 +0.10)	+0.02 (-0.06 +0.12)	-0.09 (-0.47 +0.37)	+0.49 (+0.44 +0.55)
CCL C/D	**+0.26** (+0.19 +0.33)	**+0.40** (+0.31 +0.49)	**+0.39** (+0.31 +0.44)	+0.62 (+0.30 +0.86)	**+0.71** (+0.66 +0.76)
CCL WSJ	+0.20 (+0.12 +0.26)	+0.35 (+0.25 +0.46)	+0.33 (+0.24 +0.41)	**+0.74** (+0.40 +0.92)	+0.69 (+0.63 +0.74)
BAS	+0.17 (+0.08 +0.26)	+0.37 (+0.23 +0.51)	+0.32 (+0.20 +0.42)	+0.72 (+0.42 +0.89)	+0.69 (+0.62 +0.77)
GRN	+0.18 (+0.12 +0.25)	+0.26 (+0.17 +0.35)	+0.27 (+0.19 +0.34)	+0.72 (+0.38 +0.89)	+0.63 (+0.58 +0.69)
SQE	+0.19 (+0.13 +0.26)	+0.26 (+0.15 +0.36)	+0.24 (+0.15 +0.32)	+0.51 (+0.05 +0.80)	+0.64 (+0.58 +0.69)

Table 3: Results on SummEval for all CMs. Correlation is expressed in Kendall's τ. Numbers in brackets indicated 95% CIs computed using bootstrap resampling (Deutsch et al., 2021) with 1000 samples. Highest are bold.

high scores while the remaining CMs fail to outperform even the random baseline. However, inspection of τ_{sum}, τ_{pair}/Acc_{pair} and τ_{intra} reveals that even these apparently strong CMs struggle to reliably assess coherence of individual summaries, with τ_{intra} being the most challenging regime. Comparing CMs, `CCL C/D` is most promising across all EMs except τ_{sys}, where scores are near indistinguishable due to high uncertainty. Interestingly, we find that its advantage is greatest on τ_{intra}, where its competitors exhibit particular weakness compared to other EMs. These sharp score drops might suggest other EMs reflect some system-level confounders. In combination with the observation that confounder scores as reported in Table 1 fall within the 95% CI of most CMs on all EMs except τ_{intra} this prompts us to investigate CMs for potential biases in the following section.

6.1 Detecting Biases of CMs

We have shown in Section 4.2 that CMs can appear to correlate with human coherence judgements by exploiting system-level confounders. However, it is unclear to which extent this just holds for our artificial confounders or is also an issue in realistic CM evaluation. We therefore introduce *bias matrices*, a tool that allows us to easily inspect the decisions made by a CM by separately analyzing *consistent* and *inverted* pairs of summaries from different summarizers. *Based on human scores*, we call a summary pair *consistent* if the higher-scoring summary is produced by the summarizer with the higher average score, whereas we call a pair *inverted* if the overall worse summarizer produces a stronger summary. We are specifically interested in finding instances where a CM ranks consistent pairs for a strong summarizer correctly, but fails to correctly rank its inverted pairs. This is indicative of a CM having a bias towards outputs of this particular summarizer, instead of measuring coherence. Since for strong systems, most pairs are consistent, this can still result in many correct comparisons.

Given predicted and human scores P, H as in Section 4 and systems s_1, s_2 with s_1 having a higher average human score than s_2, we define two new metrics. τ^+ indicates the ability of a CM to rank consistent pairs, whereas τ^- indicates the same for inconsistent pairs. For τ^+ we define:

$$H^+ := \{(d_i, d_j) | H_{(d_i,s_1)} > H_{(d_j,s_2)}\}$$
$$P^+ := \{(d_i, d_j) | P_{(d_i,s_1)} > P_{(d_j,s_2)}\}$$
$$\tau^+ := \frac{2|H^+ \cap P^+| - |H^+|}{|H^+|}$$

For τ^- we invert the comparisons.[9] Both τ^+ and τ^- are bounded between -1 and 1. If the ranking is -1, this indicates the ranking is always incorrect, 1 always correct. To derive the $|S| \times |S|$ bias matrix T, we order systems $s_1 \dots s_n$ in descending order of their average human score. We then have:

$$\mathrm{T}_{ij} := \begin{cases} \tau^+(s_i, s_j) & i < j \\ \tau^-(s_j, s_i) & i > j \\ 0 & i = j \end{cases}$$

[9] If s_1 is better than s_2 on every document, τ^- is undefined. In this case, biased and unbiased CMs are indistinguishable.

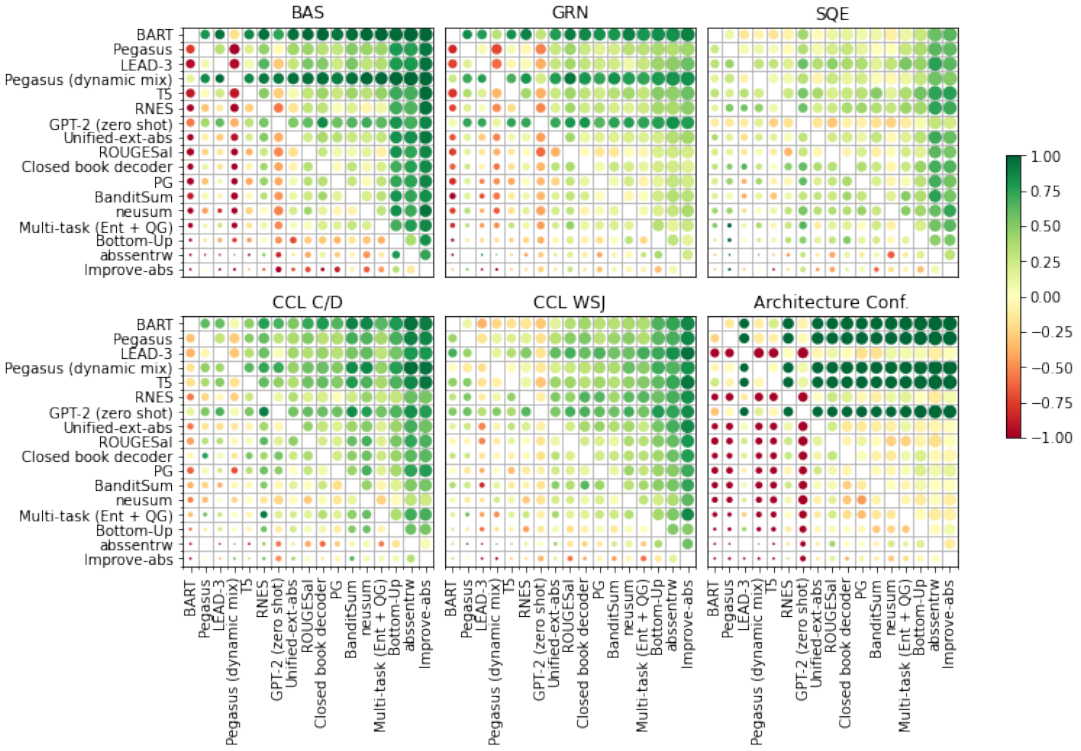

Figure 2: Bias Matrices for the best CMs. We also show the bias matrix for the architecture confounder for reference. See Figure 3 for a brief tutorial to bias matrix analysis.

We visualize T for the most promising CMs in Figure 2. To aid interpretation, we provide an annotated version for scores generated by BAS in Figure 3. We find that GRN and BAS show a very strong preference for summaries generated by BART, ranking them almost universally higher even when this disagrees with human judgements. In case of BAS this is unsurprising, since BART and BAS use the same underlying model. For GRN the reason is less clear, though analysis in Section 7.2 suggests that it might rely on the higher grammaticality of BART output. For the other CMs, biases are less evident, though CCL C/D shows a slight preference for BART and Pegasus and CCL WSJ has a slight bias towards LEAD and GPT-2.

7 CM Analysis

7.1 Correlation with Shuffle-Performance

Mohiuddin et al. (2021) have shown that the performance of CMs on the shuffle task is not predictive for performance on summary coherence evaluation. However, at the same time, the shuffling-based CCL shows comparatively strong performance in our experiments. To better understand the relation between shuffling and summary coherence, we test the ability of all CMs to discriminate shuffled and non-shuffled *reference summaries* from the test split of CNN/DM. Results are in Table 4.

Of the CMs that perform best on coherence evaluation (see Table 3), most also perform well on the shuffling task (CCL, BAS, SQE). Only GRN fails on this task. This is troubling as we would expect any CM that is able to identify coherent summaries on SummEval to be able to identify at least some shuffled reference summaries. This suggests that GRN models coherence only indirectly via proxy variables, which we elaborate on in Section 7.2.

For the entity-based measures EGR, EEG and NEG, their difficulties on the SummEval dataset are also reflected in the shuffle task. This suggests that these CMs struggle generally on CNN/DM-style summaries. In Section 7.3 we demonstrate that this is due to the overall lack of entity overlap in this dataset. Finally, UNF C/D and GRA are outliers in that they show shuffle performance on CNN/DM that is similar or better than SQE but still perform near random on SummEval coherence modelling. We investigate this in Section 7.4.

7.2 GRUEN

GRN works well for system-level correlation yet is incapable of solving the shuffle task. This prompts

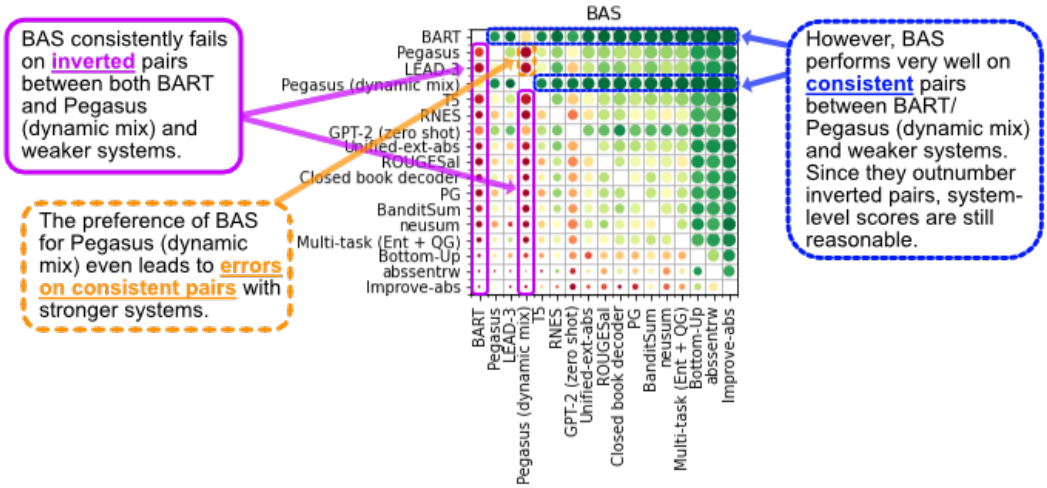

Figure 3: Bias matrix for BAS with specific analysis for BART and Pegasus. The upper triangular matrix indicates τ^+ for the given summarizer pair, the lower τ^-. The area of each circle is proportional to the number of pairs in H^+/H^- for the cell. To read off the behaviour of the CM on a specific summarizer, we follow both the corresponding row and column. A high score in the row, combined with a low score in the corresponding cell in the column implies the CM is biased towards generations by this particular summarizer.

Corpus	EGR	EEG	NEG	GRA	UNF	CCL	BAS	GRN	SQE
C/D	0.426	$0.523_{(c)}$ $0.498_{(w)}$	$0.524_{(c)}$ $0.603_{(w)}$ $0.522_{(d)}$	$0.838_{(c)}$ $0.623_{(w)}$ $0.439_{(d)}$	$0.803_{(c)}$ $0.589_{(w)}$	$0.929_{(c)}$ $0.862_{(w)}$	0.896	0.504	0.707
WSJ (orig.)	0.889	0.840	0.855	0.924	0.93	0.97	-	-	-

Table 4: Shuffle accuracies on CNN/DM for 1000 randomly sampled reference summaries. (c) means that the model was trained on CNN/DM (w) on WSJ and (d) on DUC03. Baseline accuracy would be 50%. For reference, we also list originally reported shuffle results on full WSJ articles as originally reported where applicable.

	Cola	Redun.	LM	Focus
Cola	0.57	**0.71**	0.59	0.63
Redun.		0.51	0.57	0.51
LM			0.15	0.35
Focus				0.49

Table 5: Performance of GRN constituent measures. Cells indicate system-level correlation of the combination of the respective measures. Individual measure performance is indicated on the diagonal.

us to investigate its individual components. In the reference implementation, GRN computes the sum of three scores to determine the overall score.[10] **Grammaticality** is assessed per sentence by a classifier trained on the CoLA corpus (Warstadt et al., 2019) and the average log probability under a BERT model. **Redundancy** is estimated by a fixed penalty whenever any sentence pair has token overlap above a predetermined threshold. **Focus** is scored by word-mover-similarity (Kusner et al., 2015) of neighbouring sentences.

Table 5 shows the system-level correlation of the individual scores and all pairwise combinations. CoLA plus redundancy alone account for almost the full system-level correlation of 0.72. Since neither score is dependent on sentence order, they can by design not fully account for summary coherence, raising considerable doubt about the generalizability of GRNs performance on this task.

7.3 Entity Driven Measures

To explain why EEG, EGR and NEG perform poorly even on the shuffle task, we investigate the role of entity (re-)occurrences in CNN/DM summaries. Table 6 shows that both reference summaries and SummEval data have very little lexical entity overlap in between sentences.[11] A considerable number of summaries in both SummEval and CNN/DM show no entity overlap between any of their sentences. Therefore entity-based models are inher-

[10] The coherence score reported in the paper is not part of the reference implementation. We have confirmed that this is intentional in personal communication with the authors.

[11] As determined by the Brown Coherence Toolkit. See Appendix A.

Corpus	Docs	Sents
CNN/DM Ref.	0.287	0.458
SummEval	0.178	0.301
DUC03	0.014	0.121

Table 6: Proportion of documents without any entity overlap, as well as average ratio of sentences without entity links per document for various datasets.

ently limited, at least when using lexical overlap to determine entity re-occurrence. We leave a thorough investigation of solutions like better coreference resolution or using embedding based methods as in Mesgar and Strube (2016) to future work.

7.4 Global Training vs. Pairwise Ranking

While CMs that fail the in-domain shuffling task are likely to be unsuitable for CNN/DM summaries, it is less clear why CMs with reasonable shuffle performance fail on SummEval like `UNF C/D` and `GRA C/D`. We theorize that one reason is that both `UNF` and `GRA` are trained on a margin-based ranking loss between shuffled and non-shuffled variants of the *same* document, which implies that both have the same tokens and number of sentences. The training loss thus does not impose constraints on the behaviour of the function between inputs of *different* lengths and tokens. Since SummEval, unlike e.g. DUC, has no agreed upon length constraint, this is problematic.[12] In contrast, the classification objective of `CCL` enforces a globally correct ranking of shuffled vs. unshuffled documents.

Verifying this hypothesis on SummEval directly is difficult, since summary length is deeply confounded with the generating summarizer. However, we can investigate the ability of CMs to correctly rank documents of different lengths and content by modifying the shuffle test to compare reference summaries to shuffled variants of *different* reference summaries. Figure 4 shows the relation between the difference in length between the shuffled and unshuffled summaries and the ranking accuracy of the CMs. `UNF` performs very poorly on the task, especially if the original summary is long. `GRA`, on the other hand, prefers longer documents, even if they are shuffled. In contrast, `CCL` is consistently able to correctly rank summaries regardless of length difference. Thus, for both `UNF` and `GRA` comparing documents of different lengths and content is a major obstacle. The stability of `CCL` suggests that replacing pairwise ranking with a classification objective is a direct fix to this issue. These results are also consistent with parallel work by Jwalapuram et al. (2022) who extend the pairwise shuffle-task to consider multiple negative examples. They find that including negative samples from different documents in the negative set during training improves model performance on downstream tasks.

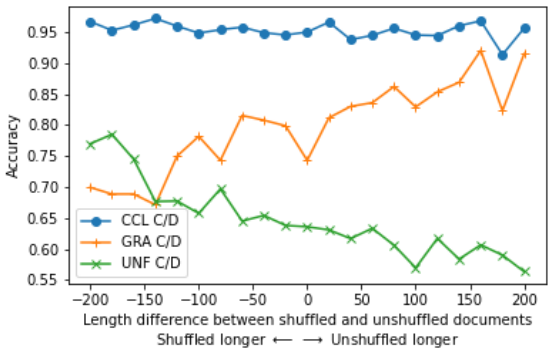

Figure 4: Ranking accuracy between shuffled and original summaries of different lengths (in characters). We sample 10,000 pairs and group them in buckets of 20 characters and clamp differences between -200 and 200.

8 Conclusion

We have investigated the performance of a wide array of CMs for summary evaluation that have not been previously systematically compared. Our investigations show that CMs must be carefully evaluated in order to avoid rewarding the modelling of shallow, system-level confounders, that are unlikely to generalize. We thus recommend researchers report our newly suggested intra-system correlation *alongside* other EMs and use bias matrices to understand unexpected drops when going from system-level to intra-system correlation.

There is considerable need to improve CMs before they become practical for summary coherence modelling. Our results point towards the following lessons for future work. Firstly, CNN/DM summaries are not amenable to entity-based analysis without considerable additional work to improve entity detection. Secondly, self-supervised training via the shuffle task shows the greatest promise for future improvements. However, we show that good shuffle performance does not naturally transfer to coherence evaluation for settings with documents of different lengths and contents. Training in a classification setup instead of the more common pairwise setup provides an effective fix for this.

[12] Summarizer length statistics are in Appendix C.

References

Regina Barzilay and Mirella Lapata. 2008. Modeling Local Coherence: An Entity-Based Approach. *Computational Linguistics*, 34(1):1–34.

Daniel Deutsch, Rotem Dror, and Dan Roth. 2021. A Statistical Analysis of Summarization Evaluation Metrics using Resampling Methods. *Transactions of the Association for Computational Linguistics*, 9:1132–1146. Place: Cambridge, MA Publisher: MIT Press.

Daniel Deutsch, Rotem Dror, and Dan Roth. 2022. Re-examining system-level correlations of automatic summarization evaluation metrics. In *Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 6038–6052, Seattle, United States. Association for Computational Linguistics.

Esin Durmus, Faisal Ladhak, and Tatsunori Hashimoto. 2022. Spurious Correlations in Reference-Free Evaluation of Text Generation. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1443–1454, Dublin, Ireland. Association for Computational Linguistics.

Micha Elsner and Eugene Charniak. 2011. Extending the Entity Grid with Entity-Specific Features. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, pages 125–129, Portland, Oregon, USA. Association for Computational Linguistics.

Alexander R Fabbri, Wojciech Kryściński, Bryan McCann, Caiming Xiong, Richard Socher, and Dragomir Radev. 2021. Summeval: Re-evaluating summarization evaluation. *Transactions of the Association for Computational Linguistics*, 9:391–409. Publisher: MIT Press.

Max Grusky, Mor Naaman, and Yoav Artzi. 2018. Newsroom: A Dataset of 1.3 Million Summaries with Diverse Extractive Strategies. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 708–719, New Orleans, Louisiana. Association for Computational Linguistics.

Camille Guinaudeau and Michael Strube. 2013. Graph-Based Local Coherence Modeling. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics (volume 1: Long Papers)*, pages 93–103, Sofia, Bulgaria. Association for Computational Linguistics.

Karl Moritz Hermann, Tomáš Kočiský, Edward Grefenstette, Lasse Espeholt, Will Kay, Mustafa Suleyman, and Phil Blunsom. 2015. Teaching Machines to Read and Comprehend. In *Proceedings of the 28th International Conference on Neural Information Processing Systems - Volume 1*, NIPS'15, pages 1693–1701, Cambridge, MA, USA. MIT Press. Number of pages: 9 Place: Montreal, Canada.

Sungho Jeon and Michael Strube. 2020. Incremental Neural Lexical Coherence Modeling. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 6752–6758, Barcelona, Spain (Online). International Committee on Computational Linguistics.

Prathyusha Jwalapuram, Shafiq Joty, and Xiang Lin. 2022. Rethinking Self-Supervision Objectives for Generalizable Coherence Modeling. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 6044–6059, Dublin, Ireland. Association for Computational Linguistics.

Fajri Koto, Timothy Baldwin, and Jey Han Lau. 2022. FFCI: A Framework for Interpretable Automatic Evaluation of Summarization. *Journal of Artificial Intelligence Research*, 73:1553–1607.

Matt J. Kusner, Yu Sun, Nicholas I. Kolkin, and Kilian Q. Weinberger. 2015. From Word Embeddings to Document Distances. In *Proceedings of the 32Nd International Conference on International Conference on Machine Learning - Volume 37*, ICML'15, pages 957–966. JMLR.org. Event-place: Lille, France.

Philippe Laban, Luke Dai, Lucas Bandarkar, and Marti A. Hearst. 2021. Can Transformer Models Measure Coherence in Text: Re-Thinking the Shuffle Test. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 2: Short Papers)*, pages 1058–1064, Online. Association for Computational Linguistics.

Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Veselin Stoyanov, and Luke Zettlemoyer. 2020. BART: Denoising Sequence-to-Sequence Pre-training for Natural Language Generation, Translation, and Comprehension. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7871–7880, Online. Association for Computational Linguistics.

Chin-Yew Lin. 2004. ROUGE: A Package for Automatic Evaluation of Summaries. In *Text Summarization Branches Out*, pages 74–81, Barcelona, Spain.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. RoBERTa: A Robustly Optimized BERT Pretraining Approach. *arXiv:1907.11692 [cs]*. ArXiv: 1907.11692.

Mohsen Mesgar, Leonardo F. R. Ribeiro, and Iryna Gurevych. 2021. A Neural Graph-Based Local Coherence Model. In *Findings of the Association for Computational Linguistics: EMNLP 2021*, pages 2316–2321, Punta Cana, Dominican Republic. Association for Computational Linguistics.

Mohsen Mesgar and Michael Strube. 2015. Graph-based Coherence Modeling For Assessing Readability. In *Proceedings of the Fourth Joint Conference on Lexical and Computational Semantics*, pages 309–318. Association for Computational Linguistics. Event-place: Denver, Colorado.

Mohsen Mesgar and Michael Strube. 2016. Lexical Coherence Graph Modeling Using Word Embeddings. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 1414–1423, San Diego, California. Association for Computational Linguistics.

Tasnim Mohiuddin, Prathyusha Jwalapuram, Xiang Lin, and Shafiq Joty. 2021. Rethinking Coherence Modeling: Synthetic vs. Downstream Tasks. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*, pages 3528–3539, Online. Association for Computational Linguistics.

Han Cheol Moon, Tasnim Mohiuddin, Shafiq Joty, and Chi Xu. 2019. A Unified Neural Coherence Model. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 2262–2272, Hong Kong, China. Association for Computational Linguistics.

Daraksha Parveen, Mohsen Mesgar, and Michael Strube. 2017. Generating Coherent Summaries of Scientific Articles Using Coherence Patterns. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*. Number: TUD-CS-2017-0301.

Matthew E. Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer. 2018. Deep Contextualized Word Representations. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 2227–2237, New Orleans, Louisiana. Association for Computational Linguistics.

Emily Pitler, Annie Louis, and Ani Nenkova. 2010. Automatic Evaluation of Linguistic Quality in Multi-Document Summarization. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics*, pages 544–554, Uppsala, Sweden. Association for Computational Linguistics.

Maja Popović. 2017. chrF++: Words Helping Character N-Grams. In *Proceedings of the Second Conference on Machine Translation*, pages 612–618, Copenhagen, Denmark. Association for Computational Linguistics.

Abigail See, Peter J. Liu, and Christopher D. Manning. 2017. Get To The Point: Summarization with Pointer-Generator Networks. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1073–1083. Association for Computational Linguistics. Event-place: Vancouver, Canada.

Eva Sharma, Luyang Huang, Zhe Hu, and Lu Wang. 2019. An Entity-Driven Framework for Abstractive Summarization. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 3271–3282.

Dat Tien Nguyen and Shafiq Joty. 2017. A Neural Local Coherence Model. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1320–1330, Vancouver, Canada. Association for Computational Linguistics.

Alex Warstadt, Amanpreet Singh, and Samuel R. Bowman. 2019. Neural network acceptability judgments. *Transactions of the Association for Computational Linguistics*, 7:625–641. Place: Cambridge, MA Publisher: MIT Press.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Remi Louf, Morgan Funtowicz, Joe Davison, Sam Shleifer, Patrick von Platen, Clara Ma, Yacine Jernite, Julien Plu, Canwen Xu, Teven Le Scao, Sylvain Gugger, Mariama Drame, Quentin Lhoest, and Alexander Rush. 2020. Transformers: State-of-the-Art Natural Languag Processing. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 38–45, Online. Association for Computational Linguistics.

Stratos Xenouleas, Prodromos Malakasiotis, Marianna Apidianaki, and Ion Androutsopoulos. 2019. SUM-QE: A BERT-based Summary Quality Estimation Model. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 6005–6011, Hong Kong, China. Association for Computational Linguistics.

Weizhe Yuan, Graham Neubig, and Pengfei Liu. 2021. BARTScore: Evaluating generated text as text generation. In *Advances in Neural Information Processing Systems*, volume 34, pages 27263–27277. Curran Associates, Inc.

Wei Zhao, Michael Strube, and Steffen Eger. 2022. DiscoScore: Evaluating Text Generation with BERT and Discourse Coherence. *arXiv:2201.11176 [cs]*. ArXiv: 2201.11176.

Wanzheng Zhu and Suma Bhat. 2020. GRUEN for Evaluating Linguistic Quality of Generated Text. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 94–108, Online. Association for Computational Linguistics.

A Implementation Details

A.1 Extended Entity Grid (EEG)

We use the original implementation that is part of the Brown Coherence Toolkit[13]. For preprocessing, we use the Stanford parser[14]. We identify entities using OpenNLP as suggested in the README.

For WSJ we used the pretrained `f-wsj` model provided in the toolkit. For CNN/DM we trained our own model. We found that the implementation ran out of memory on the 287,011 instances in CNN/DM on our machine with 32GB of RAM. We thus limited the instances considered for CNN/DM to 10% of the original dataset (28,701).

A.2 Entity Graph (EGR)

Since there is no reference implementation of the Entity Graph, we implement our own version based on the grid created by the Brown Coherence Toolkit. We use the P_{Acc} measure with distance penalty which performed best in the original paper.

A.3 Neural Entity Grid (NEG)

Since no models are publicly available, we train new models for all settings using the reference implementation[15].

For DUC03 and WSJ we use the entity grids and training pairs provided by the authors in the repository. These were also created using the Brown Coherence Toolkit. For CNN/DM we create our own examples, following the original settings. We found that the original implementation of the shuffling procedure leaves artifacts in the data since the row order is unchanged between shuffled and unshuffled documents. However, for unshuffled documents the order of rows in the entity grid roughly corresponds to the order of entities in the sentences, whereas for shuffled documents this is not the case. Since this can be picked up by the convolutional network for short documents, we modify the input data to randomly shuffle row order for each instance.

Embedding Size	100
Batch Size	64
Pool Length	6
Window Size	6
Number of Filters	150
Hidden Size	250

Table 7: Best hyperparameters for the neural entity grid on DUC03.

For the shuffling tasks on WSJ we use the reported hyperparameters, which we also use for CNN/DM. For DUC, no hyperparameters were reported, so we use the built-in hyperparameter search. We achieve the best results using the parameters reported in Table 7.

A.4 Graph-based Model (GRA)

We use the original implementation.[16] For WSJ, we use the provided pretrained model. For DUC and CNN/DM we train the model using default settings, which includes an ELMo embedding layer. The graph-representation is created from an entity grid representation as provided by the Brown Coherence Toolkit.

A.5 Unified Coherence Model (UNF)

We use the original implementation.[17] We train new models for CNN/DM and WSJ using default settings. In the original implementation, scores are computed using a sum over coherence scores for windows of three sentences each, since in their pairwise evaluation, samples always have the same length. In our experiments, we use the mean over the windows instead to normalize for length. For completeness, we also conducted experiments using the original setting, which did not lead to any improvement.

A.6 Coherence Classifier (CCL)

We originally experimented with the pretrained WSJ model provided by the authors of (Laban et al., 2021).[18] However, we found that the model achieved near-random scores when evaluated on

[13] https://web.archive.org/web/20200505174052/https://bitbucket.org/melsner/browncoherence

[14] https://nlp.stanford.edu/software/lex-parser.shtml

[15] https://github.com/datienguyen/cnn_coherence

[16] https://github.com/UKPLab/emnlp2021-neural-graph-based-coherence-model

[17] https://github.com/taasnim/unified-coherence-model

[18] https://github.com/tingofurro/shuffle_test

SummEval for reasons that are difficult to ascertain as the original training code is unavailable. We thus train our own coherence classifier models for both CNN/DM and WSJ. We use the `roberta-large` model as implemented in the huggingface library (Wolf et al., 2020) in a sequence classification setup. We use a learning rate of $2e-6$ and train for a maximum of six epochs. We select the best model using f1-score on the validation set.

A.7 BARTScore (BAS)

We reimplement the finetuned BARTScore variant using the `bart-large-cnn` checkpoint from the huggingface library. Since the original model is evaluated using Spearman's ρ, we separately verified that it exactly reproduces the reported results.

A.8 GRUEN (GRN)

We use the scores provided by the official reference implementation.[19]

A.9 SumQE (SQE)

We use the scores provided by the official reference implementation.[20] We use the Q5 head of the model jointly trained on all three DUC datasets.[21]

A.10 Hardware

All experiments that include neural network training (i.e. `NEG`, `GRA`, `UNF`, `CCL`) were run on a single node with four Quadro RTX 6000 GPUs.

B Detailed Intra-System Correlation Results

Figure 5 shows the individual intra-system correlations for all summarizers in SummEval for the best CMs and the human upper bound. We find that CMs struggle across the whole range of summarizers, including summarizers with high variance in coherence scores, where we would expect the task to be easier. Furthermore, we find none of the available CMs can consistently outperform all others. For example, `BAS` outperforms other CMs on Bottom-Up and Improve-Abs, but performs significantly worse on the top systems, including BART itself.

[19] https://github.com/WanzhengZhu/GRUEN

[20] https://github.com/nlpaueb/SumQE

[21] https://archive.org/download/sum-qe/BERT_DUC_all_Q5_Multi%20Task-5.h5

C Length Statistics

We present the length distribution of summarizer outputs on SummEval in Figure 6.

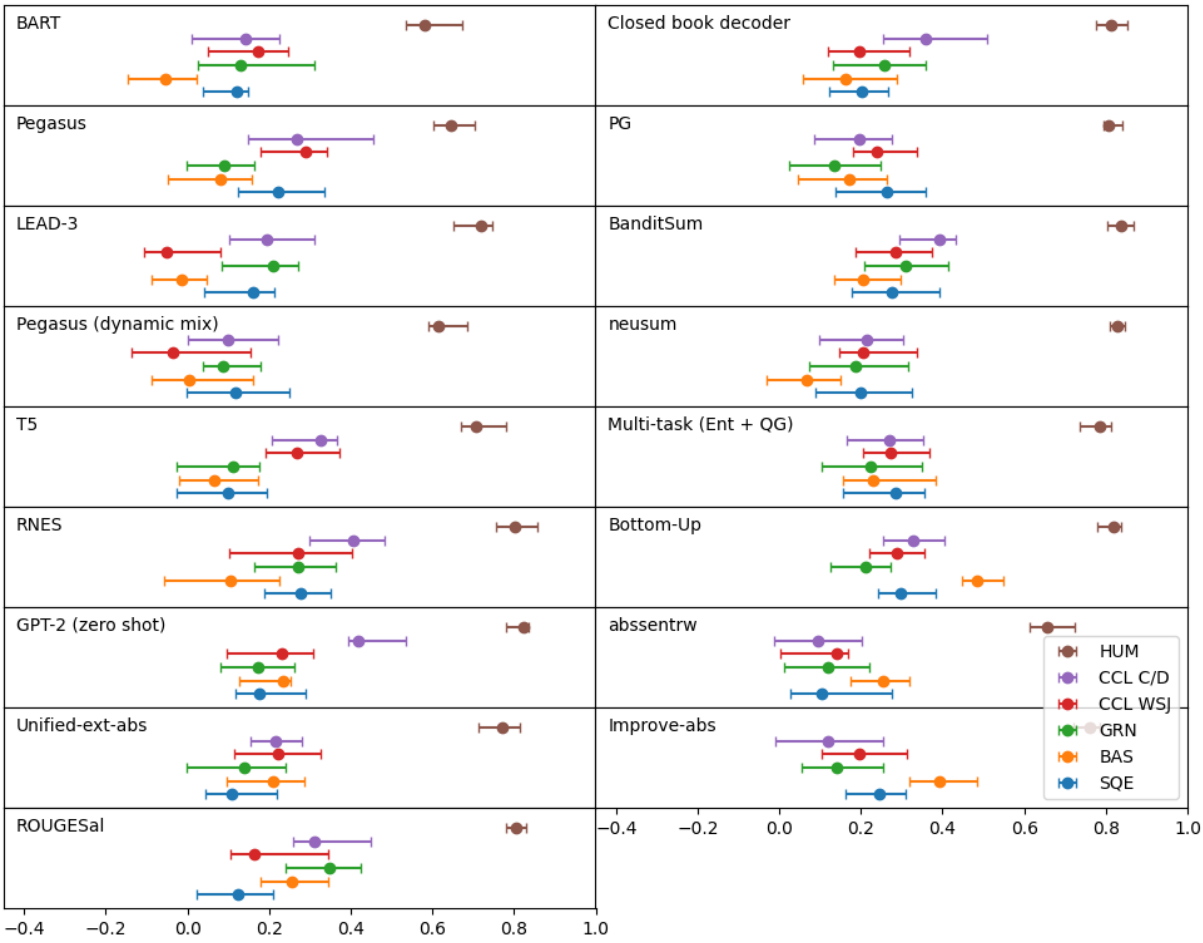

Figure 5: Intra-system correlations of the best CMs as well as the human upper bound on the SummEval dataset. Bars indicate 95% confidence intervals determined by bootstrap resampling with 1000 samples.

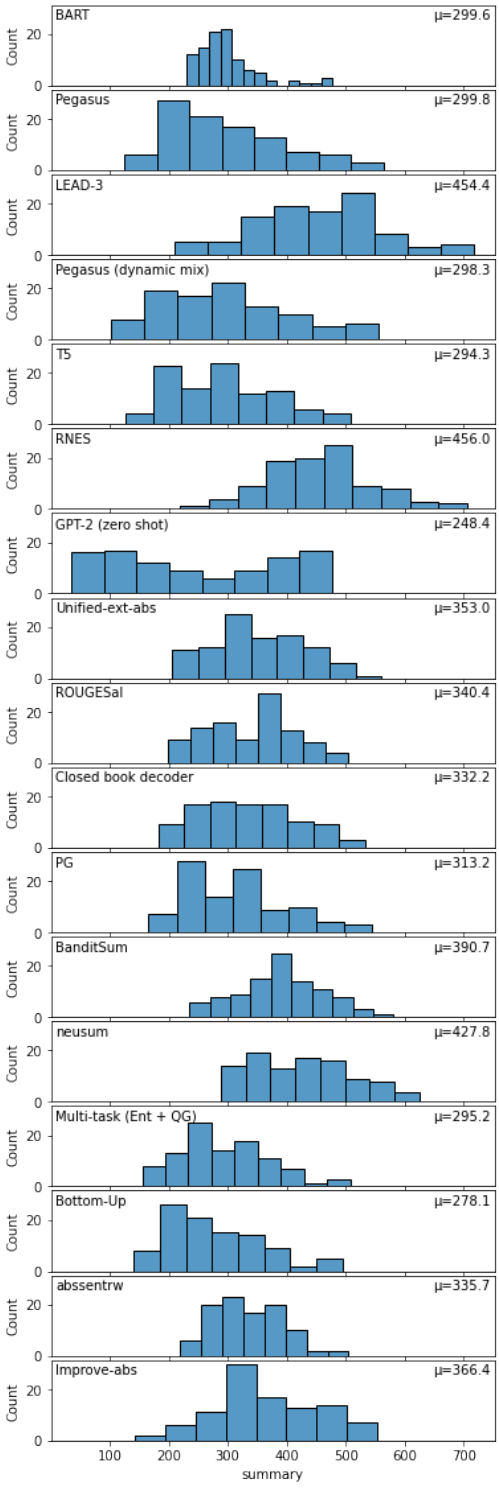

Figure 6: Histograms of the lengths of summaries generated by the summarizers in SummEval and their mean lengths. Both in characters.

Summarizing Dialogues with Negative Cues

Junpeng Liu[†*]
Beijing University of Posts and Telecommunications,
Beijing, China
jeepliu@bupt.edu.cn

Yanyan Zou[†]
JD.com,
Beijing, China
zouyanyan6@jd.com

Yuxuan Xi[*]
Peking University, Beijing, China
xiyuxuan@pku.edu.cn

Shengjie Li and **Mian Ma** and **Zhuoye Ding**
JD.com, Beijing, China
{lishengjie1,mamian,dingzhuoye}@jd.com

Abstract

Abstractive dialogue summarization aims to convert a long dialogue content into its short form where the salient information is preserved, while the redundant pieces are ignored. Different from the well-structured text, such as news and scientific articles, dialogues often consist of utterances coming from two or more interlocutors, where the conversations are often informal, verbose, and repetitive, sprinkled with false-starts, backchanneling, reconfirmations, hesitations, speaker interruptions and the salient information is often scattered across the whole chat. The above properties of conversations make it difficult to directly concentrate on scattered outstanding utterances and thus present new challenges of summarizing dialogues. In this work, we propose to explicitly have the model perceive the redundant parts of an input dialogue history, so that the model is able to pay more attention to the salient pieces. To be specific, we design two strategies to construct examples without salient pieces as negative cues. Then, the sequence-to-sequence likelihood loss is cooperated with the unlikelihood objective to drive the model focus less on the unimportant information as well as pay more attention to the salient pieces. Extensive experiments on the benchmark dataset demonstrate that our simple method outperforms baselines with regard to both semantic matching and factual consistent based metrics. The human evaluation also proves the performance gains led by our approach.

1 Introduction

Online conversations have become an indispensable manner of communication in our daily work and life, where people tend to exchange their ideas, share information, consult via textual messages. Especially in the era of information explosion, it is much more challenging and time-consuming to go through all the conversation content and catch key ideas (Gao et al., 2020). Thus, it is paramount to present the most salient facts, instead of the whole lengthy dialogue history, which is beneficial to various scenarios and applications, such as online customer service (Liu et al., 2019a), meeting and email thread summary (Zhao et al., 2019). Therefore, this work focuses on the *abstractive dialogue summarization* task, aiming to automatically convert the long dialogue history into its shorter form retaining the most essential and informative content yet getting rid of the dispensable pieces, exemplified by a dialogue-summary instance in Figure 1.

Molly:	*Guys, do you think it's a very bad idea to go to Sweden for a week in January?*
Margaret:	We bought some cheep tickets half a year ago and now we're hesitating.
Peter:	Haha, no but it will be just dark and cold.
Margaret:	Rainy?
Kal:	Possibly. *But if you stay in Stockholm, there are always nice things to do. Museums, bars etc*
Kal:	Not so much nature though which is truly stunning around Stockholm.
Margaret:	Yes, but it's January, one would have to go to Argentina to enjoy nature.
Kal:	Exactly.
Peter:	*Visit the Vasa Museum, it's really fun.*
Molly:	We will:) Thanks :)
Peter:	Enjoy!
Summary:	Molly and Margaret are going to Sweden in January. Kal and Peter advise them to stay in Stockholm and visit Vasa Museum.

Figure 1: An example of dialogue with its summary. Green: nouns, *italic*: salient utterances.

One intuitive solution to summarizing dialogue content is to directly adopt existing summarization systems (Gehrmann et al., 2018; Zhang et al., 2020a; Zou et al., 2020) designed for well-structured text, such as news and scientific articles (Shang et al., 2018; Gliwa et al., 2019) or to employ hierarchical models to capture features from different turns of different speakers (Zhao et al., 2019;

[*]Work done during internship at JD.com.
[†]The first two authors made equal contributions. Corresponding to Yanyan Zou.

Proceedings of the 29th International Conference on Computational Linguistics, pages 6050–6056
October 12–17, 2022.

Zhu et al., 2020). Unfortunately, succinctly summarizing dialogue content presents new challenges due to intrinsic properties of conversations. Unlike the field of well-organized text merely from a single person, dialogues often consist of utterances coming from two or more interlocutors, where the conversations are often informal, verbose and repetitive, sprinkled with false-starts, backchannels, reconfirmations, hesitations, speaker interruptions (Sacks et al., 1978) and the key information is often scattered throughout the whole chat. The above properties of conversations make it difficult to concentrate on the scattered salient utterances.

Recent studies incorporate intrinsic information of dialogues to handle the challenges for summarizing dialogues, such as topic features (Liu et al., 2019b; Li et al., 2019; Chen and Yang, 2020; Liu et al., 2021a), dialogue acts (Goo and Chen, 2018), conversation stages (Chen and Yang, 2020) and coreference information (Liu et al., 2021b). The main idea of such existing summarization systems is to directly learn the salient information of the input dialogues with various architecture designed or extra knowledge added. Differently, this work proposes to train a dialogue summarization system by explicitly telling the model the unimportant/redundant pieces of an input dialogue, so that the model is able to focus less on the given negative hints and pay more attention to the salient information. To be specific, we design two strategies to construct negative examples, namely Noun Drop, and Salient Utterance Drop. Then, we design an unlikelihood objective to model the probability of producing the gold summary given a negative example. The model is then trained based on the summation of likelihood and unlikelihood objectives. Extensive experiments on the SAMSum dataset demonstrate that our proposed method outperforms baselines on both semantic matching and factual consistent based metrics. The human evaluation also proves the performance improvements of our simple yet effective method.

2 Method

2.1 Sequence-to-Sequence Learning

We consider the abstractive dialogue summarization task as a sequence-to-sequence learning problem. We use the Transformer (Vaswani et al., 2017) as our backbone architecture, where the model takes as input the dialogue utterances and generates a corresponding summary in an end-to-end fashion. To be specific, for a dialogue $D = (u_1, u_2, ..., u_{|D|})$, consisting of $|D|$ utterances, coupled with its corresponding summary $Y = (y_1, y_2, ..., y_{|Y|})$ in the length of $|Y|$, the goal is to learn the optimal model parameters θ and to estimate the conditional probability:

$$P_\theta(Y|D) = \prod_{i=1}^{|Y|} p_\theta(y_i|y_{1:i-1}, D) \quad (1)$$

where $y_{1:i-1}$ denotes the first $i-1$ tokens of the output sequence (i.e., $y_{1:i-1} = (y_1, y_2, ..., y_{i-1})$). Given the whole training set $(\mathcal{D}, \mathcal{Y})$, this model can be trained to maximize the log-likelihood by minimizing:

$$\mathcal{L}_{MLE}(\theta; \mathcal{D}, \mathcal{Y}) = -\sum_{(D,Y)\in(\mathcal{D},\mathcal{Y})} \log P_\theta(Y|D)$$

2.2 Unlikelihood Objective

We first introduce two strategies for constructing negative examples:

- **Noun Drop**: We simply remove all the nouns (e.g., named entities) appearing in dialogue D since most fact details (i.e., salient information) are presented in nouns, highlighted by green color in Figure 1.
- **Salient Utterance Drop**: An utterance is defined as a *salient* one when the ROUGE-2 (Lin, 2004) recall score between it and the gold summary is larger than zero. This strategy removes all the salient utterances on which the gold summary is grounded and the remaining utterances are concatenated in order to form a new dialogue content. The utterances marked in italic in Figure 1 are salient ones and removed from the dialogue to construct a negative example.

For each dialogue $D \in \mathcal{D}$, each strategy results in a single negative example, denoted as D', yielding a new set $\mathcal{D}'$. The unlikelihood objective is then calculated as:

$$\mathcal{L}_{UNL}(\theta; \mathcal{D}', \mathcal{Y}) = -\sum_{(D',Y)\in(\mathcal{D}',\mathcal{Y})} \log\left(1 - P_\theta(Y|D')\right)$$

Different from the unlikelihood training (Welleck et al., 2019) whose key idea behind is to decrease the model's generation probability of certain negative candidates conditioned on the original input

text, our unlikelihood objective aims to decrease the probability of producing the target summary given the negative input D'. The final loss for the sequence-to-sequence learning is then defined as:

$$\begin{aligned}\mathcal{L} &= \mathcal{L}_{MLE} + \mathcal{L}_{UNL} \\ &= -\sum_{(D,D'Y)\in(\mathcal{D},\mathcal{D}',\mathcal{Y})} [\log P_\theta(Y|D) \\ &+ \log\left(1 - P_\theta(Y|D')\right)]\end{aligned}$$

The goal is to minimize the loss $\mathcal{L}$, i.e., maximizing the probability of generating the summary Y given the original dialogue D, while minimizing the probability of producing Y given D', which is similar to the idea of contrastive learning. In this scenario, the negative examples D' can be considered as explicit negative cues to drive the model focus more on the salient information.

3 Experiments

3.1 Datasets

We evaluate our model on the widely-used dialogue summarization datasets, SAMSum. Such a dataset comprises of natural message-like conversations expressed in English written by two or more linguists, each of which is annotated with summary created by language experts (Gliwa et al., 2019). The training set consists of 14,732 dialogue-summary pairs, while the validation and test set contain 818 and 819 instances individually. We list the detailed data statistics of each split (i.e., training, validation, test) with regard to average tokens, utterances and speakers in Table 1.

3.2 Implementation Details

We adopted the sequence-to-sequence Transformer model as our backbone architecture, which is implemented using Fairseq toolkit[1] (Ott et al., 2019). To be specific, our model is initialized with a pre-trained sequence-to-sequence, i.e., BART (Lewis et al., 2020), . Thus they share the same architectures, a 12-layer encoder-decoder Transformer. Each layer has 16 attention heads, and the hidden size and feed-forward filter size is 1024 and 4096, respectively, resulting in 400M trainable parameters. The dropout rates for all layers are set to 0.1. The optimizer is Adam (Kingma and Ba, 2015) with β_1 = 0.9 and β_2 = 0.999. The peak learning rates for all experiments are set to $4e-5$ with 200 warmup steps. We also adopted the same learning rate schedule strategies as in Vaswani et al. (2017). The maximum number of tokens in each batch is 800. The model is trained for 4 or 5 epochs for different perturbation methods. Each epoch takes around 0.7 hours on single Tesla P40 GPU. To obtain all nouns in a dialogue, we applied the spaCy toolkit[2] to obtain the part-of-speech and named entities tags. When constructing negative examples where salient utterances are dropped, we simply adopt the ROUGE scores. All hyperparameters are set based on the performance of the validation set.

3.3 Baseline

- Lead3 is a commonly adopted method in the extractive document summarization task, which simply takes the first three leading sentences of an input text as its summary.
- PTGen (See et al., 2017) modifies a sequence-to-sequence generation model with the copy and coverage mechanisms to copy words originated from the input text.
- FastAbs-RL (Chen and Bansal, 2018) first selects pivot sentences and then generates abstract summary with reinforcement learning.
- DynamicConv + GPT-2/News (Wu et al., 2019) proposes a lightweight dynamic convolutions to replace the self-attention modules in the Transformer layers.
- BART (Lewis et al., 2020) is a pre-trained encoder-decoder Transformer model.
- MultiView BART (Chen and Yang, 2020) uses multi-view features to summarize dialogues.

3.4 Automatic Evaluation

To evaluate the effectiveness of the proposed model and compare it with other baselines, we adopted the full-length F1-based ROUGE scores (Lin, 2004) to measure the quality of summary output generated by different systems. Specifically, we used the files2rouge[3] package based on the official `ROUGE-1.5.5.pl` perl script to get the full-length ROUGE-1, ROUGE-2 and ROUGE-L F-measure scores. The recent popular automatic evaluation metric for text generation, BERTSCORE

[1]We empirically observed that different frameworks (e.g. Fairseq and Huggingface Transformer) may obtain different results even under the same hyperparameter settings.

[2]https://spacy.io/

[3]https://github.com/pltrdy/files2rouge Note that the ROUGE scores might vary with different ROUGE tookits.

Split	#Dial	#Speaker	#Turns	#Words (Dial)	#Words (Summary)
Train	14,732	2.40	11.17	83.90	20.35
Valid	818	2.39	10.83	83.26	20.14
Test	819	2.36	11.25	83.87	20.43

Table 1: Data statistics of the dialogue summarization dataset, SAMSum, including the total number of dialogues (#Dial), the average number of participants (#Speaker), the average number of turns (# Turns), the average number of words in the dialogue (# Words (Dial)) and in the summary (# Words (Summary)).

Model	ROUGE-1	ROUGE-2	ROUGE-L	BERTSCORE	QUESTEVAL
Lead3	31.4	8.7	29.4	-	-
PTGen	40.1	15.3	36.6	-	-
DynamicConv + GPT-2	41.8	16.4	37.6	-	
FastAbs-RL	42.0	18.1	39.2	-	-
DynamicConv + News	45.4	20.7	41.5	-	-
Multiview BART	**53.9**	28.4	44.4	53.0	40.3
BART	52.6	27.0	42.1	52.1	39.8
+ Noun Drop	53.4∗	28.4∗	**44.7**∗	**53.5**	**41.6**
+ Salient Utterance Drop	53.2∗	**28.7**∗	44.6∗	53.2	40.5

Table 2: Results on SAMSum test split. $*$ indicates the results are significantly different from BART baseline in terms of ROUGE scores ($p < 0.05$, according to the ROUGE script). The highest score is highlighted with **bold**, while the second highest is marked with underline.

(Zhang et al., 2020b), is also presented for comparisons. The above metrics mainly focus on the semantic similarity between the generated output and the ground truth, based on either string match or meaning similarity. Moreover, we also consider the QUESTEVAL (Scialom et al., 2021) to evaluate the summary's factual consistency. To be specific, given an input text (e.g., dialogue content in this paper) and a summary, QuestEval first extracts question answers (considering all the named entities and nouns) from either the input text or the generated summary, and then generates natural language questions from the input text or the summary correspondingly conditioned on the generated answers. A Question Answering (in short, QA) model is employed to consume the input text to answer the questions derived from the summary, resulting in a score, denoted as the PRECISION score. Such a score implies that a summary should contain only factual information consistent to the input text. Similarly, the QA model is also applied to address the questions generated from the input text, producing another score, namely the RECALL score, showing that the summary should contain the most important information from the source text. The final QuestEval score is the harmonic mean of the precision and recall, i.e., the F1-measure score. We adopted the version with learned weights for questions, which has proved high correlation with human judged consistency and relevance (Scialom et al., 2021).

As listed in Table 2, in terms of the semantic similarity-based metrics (i.e., ROUGE and BERTSCORE), the Noun Drop achieves highest ROUGE-L and BERTSCORE, while the Salient Utterance Drop obtains the highest ROUGE-2, demonstrating the effectiveness of negative cues with the unlikelihood objective. With regard to the factual consistency metric QUESTEVAL, the variant with Noun Drop obtained the highest score, which demonstrates its effectiveness to generate the factual consistent summaries since detailed fact are mainly presented in the form of named entities and nouns residing in the source input.

Overall, the variant with Noun Drop works the best for the three of five metrics. It is also worthy noting that MultiView BART requires extra topic segmentation algorithms to obtain the multi-view features, while our method only needs part-of-speech tags and ROUGE scores to construct negative examples which are easier to achieve.

We have also tried to combine the Noun Drop and Salient Utterance Drop. It is interesting that we did not obtained consistently improvement. One

Systems	1st	2nd	3rd	4th	MR
BART	0.04	0.12	0.34	0.51	3.34
MultiView BART	0.22	0.24	0.31	0.23	2.55
Ours	0.28	0.30	0.23	0.19	2.33
Gold	0.46	0.34	0.13	0.07	1.98

Table 3: Human evaluation on SAMSum: proportions of rankings. MR: mean rank (the lower the better).

possible reason is that the negative examples might lose too much information so that the negative signals become weaker.

3.5 Human Evaluation

We also elicit feedback from human efforts to evaluate the generated summaries from different summarization systems. We compared our best performing model (i.e. +Noun Drop) with the human references, as well as two baselines, BART (Lewis et al., 2020) and MultiView BART (Chen and Yang, 2020). We randomly select 100 dialogues from the test split of SAMSum dataset. To ensure fairness, for each dialogue, we list its candidate outputs in a random order, including human references (denoted as Gold), and outputs generated by three models. 10 participants are presented with a dialogue and its paired candidate summaries, where all participants are shown the same candidate order. For each selected dialogue, they are asked to rank the candidate output from the best to worst with regard to three criteria:

- *Fluency*: Is the summary fluent and grammatically correct?
- *Informativeness*: Does the summary contains the most informative pieces of the dialogue?
- *Succinctness*: Does the summary express in an abstractive way (e.g., without repetitions)?

Table 3 listed the proportions of different system rankings and mean rank (lower is better). The output of our proposed method is ranked as the most appropriate summary for 28% of all cases. Overall, we obtain lower mean rank than the other two systems but still lags behind the Gold one. The Fleiss' Kappa score (Fleiss, 1971) among participants is 0.527 that demonstrates fair inter-rater agreement.

4 Conclusion

Recent studies involved dialogue studies (e.g., topical information, coreference information, and dialogue acts) to make the model directly pay more attention to salient parts. However, the characteristics of dialogue content make it challenging to concentrate on scattered outstanding utterances. Rather, in this work, we propose a simple yet effective approach to explicitly tell a model the redundant pieces of a dialogue and thus focus more on the salient ones. We proposed two strategies to construct negative samples with redundant information and designed an unlikelihood objective to force the model learn less from redundant information, in other words, learning more from the salient pieces. Experiments on the benchmark dataset demonstrate the efficacy of the proposed model. In the future, we plan to investigate other strategies for constructing negative examples and replace the unlikelihood objective with the ranking loss.

5 Broader Impact Statement

Our simple yet effective abstractive dialogue summarization system could be used where there exists dialogue systems (two or multi-party dialogues). For example, it could be used for grasping the key points quickly or recapping on the salient information of online office meeting. In addition, the system can also be used for customer service, requiring employees to summarize the conversation records of customers' inquiries, complaints and suggestions.

The daily dialogue dataset used in this work is publicly available, and only for research purpose. There may exist biased views in them, and the content of them should be viewed with discretion.

References

Jiaao Chen and Diyi Yang. 2020. Multi-view sequence-to-sequence models with conversational structure for abstractive dialogue summarization. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing*.

Yen-Chun Chen and Mohit Bansal. 2018. Fast abstractive summarization with reinforce-selected sentence rewriting. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics*.

Joseph L Fleiss. 1971. Measuring nominal scale agreement among many raters. *Psychological bulletin*, 76(5):378.

Shen Gao, Xiuying Chen, Zhaochun Ren, Dongyan Zhao, and Rui Yan. 2020. From standard summarization to new tasks and beyond: Summarization with

manifold information. In *Proceedings of the Twenty-Ninth International Joint Conference on Artificial Intelligence*.

Sebastian Gehrmann, Yuntian Deng, and Alexander M Rush. 2018. Bottom-up abstractive summarization. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*.

Bogdan Gliwa, Iwona Mochol, Maciej Biesek, and Aleksander Wawer. 2019. SAMSum corpus: A human-annotated dialogue dataset for abstractive summarization. In *Proceedings of the 2nd Workshop on New Frontiers in Summarization*.

Chih-Wen Goo and Yun-Nung Chen. 2018. Abstractive dialogue summarization with sentence-gated modeling optimized by dialogue acts. In *2018 IEEE Spoken Language Technology Workshop (SLT)*, pages 735–742. IEEE.

Diederik P Kingma and Jimmy Lei Ba. 2015. Adam: A method for stochastic optimization. In *International Conference on Learning Representations*.

Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Veselin Stoyanov, and Luke Zettlemoyer. 2020. Bart: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*.

Manling Li, Lingyu Zhang, Heng Ji, and Richard J. Radke. 2019. Keep meeting summaries on topic: Abstractive multi-modal meeting summarization. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*.

Chin-Yew Lin. 2004. ROUGE: A package for automatic evaluation of summaries. In *Text Summarization Branches Out*.

Chunyi Liu, Peng Wang, Jiang Xu, Zang Li, and Jieping Ye. 2019a. Automatic dialogue summary generation for customer service. In *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*.

Junpeng Liu, Yanyan Zou, Hainan Zhang, Hongshen Chen, Zhuoye Ding, Caixia Yuan, and Xiaojie Wang. 2021a. Topic-aware contrastive learning for abstractive dialogue summarization. In *Findings of the Association for Computational Linguistics: EMNLP 2021*, pages 1229–1243.

Zhengyuan Liu, A. Ng, Sheldon Lee Shao Guang, AiTi Aw, and Nancy F. Chen. 2019b. Topic-aware pointer-generator networks for summarizing spoken conversations. *2019 IEEE Automatic Speech Recognition and Understanding Workshop (ASRU)*, pages 814–821.

Zhengyuan Liu, Ke Shi, and Nancy Chen. 2021b. Coreference-aware dialogue summarization. In *Proceedings of the 22nd Annual Meeting of the Special Interest Group on Discourse and Dialogue*.

Myle Ott, Sergey Edunov, Alexei Baevski, Angela Fan, Sam Gross, Nathan Ng, David Grangier, and Michael Auli. 2019. fairseq: A fast, extensible toolkit for sequence modeling. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics (Demonstrations)*.

Harvey Sacks, Emanuel A Schegloff, and Gail Jefferson. 1978. A simplest systematics for the organization of turn taking for conversation. In *Studies in the organization of conversational interaction*, pages 7–55. Elsevier.

Thomas Scialom, Paul-Alexis Dray, Sylvain Lamprier, Benjamin Piwowarski, Jacopo Staiano, Alex Wang, and Patrick Gallinari. 2021. Questeval: Summarization asks for fact-based evaluation. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*.

Abigail See, Peter J Liu, and Christopher D Manning. 2017. Get to the point: Summarization with pointer-generator networks. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics*.

Guokan Shang, Wensi Ding, Zekun Zhang, Antoine Tixier, Polykarpos Meladianos, Michalis Vazirgiannis, and Jean-Pierre Lorré. 2018. Unsupervised abstractive meeting summarization with multi-sentence compression and budgeted submodular maximization. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics*.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Lukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in Neural Information Processing Systems*.

Sean Welleck, Ilia Kulikov, Stephen Roller, Emily Dinan, Kyunghyun Cho, and Jason Weston. 2019. Neural text generation with unlikelihood training. In *International Conference on Learning Representations*.

Felix Wu, Angela Fan, Alexei Baevski, Yann Dauphin, and Michael Auli. 2019. Pay less attention with lightweight and dynamic convolutions. In *International Conference on Learning Representations*.

Jingqing Zhang, Yao Zhao, Mohammad Saleh, and Peter Liu. 2020a. Pegasus: Pre-training with extracted gap-sentences for abstractive summarization. In *International Conference on Machine Learning*.

Tianyi Zhang, Varsha Kishore, Felix Wu, Kilian Q. Weinberger, and Yoav Artzi. 2020b. Bertscore: Evaluating text generation with BERT. In *International Conference on Learning Representations*.

Zhou Zhao, Haojie Pan, Changjie Fan, Yan Liu, Linlin Li, Min Yang, and Deng Cai. 2019. Abstractive meeting summarization via hierarchical adaptive segmental network learning. In *The World Wide Web Conference*, pages 3455–3461.

Chenguang Zhu, Ruochen Xu, Michael Zeng, and Xuedong Huang. 2020. A hierarchical network for abstractive meeting summarization with cross-domain pretraining. In *Findings of the Association for Computational Linguistics*.

Yanyan Zou, Xingxing Zhang, Wei Lu, Furu Wei, and Ming Zhou. 2020. Pre-training for abstractive document summarization by reinstating source text. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing*.

ALEXSIS-PT: A New Resource for Portuguese Lexical Simplification

Kai North[1], **Marcos Zampieri**[1], **Tharindu Ranasinghe**[2]
[1]George Mason University, USA
[2]University of Wolverhampton, UK
knorth8@gmu.edu

Abstract

Lexical simplification (LS) is the task of automatically replacing complex words for easier ones making texts more accessible to various target populations (e.g. individuals with low literacy, individuals with learning disabilities, second language learners). To train and test models, LS systems usually require corpora that feature complex words in context along with their candidate substitutions. To continue improving the performance of LS systems we introduce ALEXSIS-PT, a novel multi-candidate dataset for Brazilian Portuguese LS containing 9,605 candidate substitutions for 387 complex words. ALEXSIS-PT has been compiled following the ALEXSIS protocol for Spanish opening exciting new avenues for cross-lingual models. ALEXSIS-PT is the first LS multi-candidate dataset that contains Brazilian newspaper articles. We evaluated four models for substitute generation on this dataset, namely mDistilBERT, mBERT, XLM-R, and BERTimbau. BERTimbau achieved the highest performance across all evaluation metrics.

1 Introduction

The development of lexical simplification (LS) systems provides a cost-effective means of making texts accessible to individuals with reading disabilities or low-literacy who are at an economic and social disadvantage (Nogueira et al., 2022). LS systems aim to replace difficult to understand (complex) words or phrases with simpler alternatives (North et al., 2022). Consider the examples in Table 2. An LS system would firstly identify the word shown in bold as being complex (Paetzold and Specia, 2016). It would then generate top-k number of candidate substitutions that preserve the meaning of the original complex word in the provided context, yet are easier to understand or are more familiar to the user. These top-k candidate substitutions are then filtered based on their appropriateness, referred to as substitute generation, and finally ranked in accordance to their suitability, known as substitute ranking.

Various studies have been published on Brazilian Portuguese (pt-BR) LS (Aluísio and Gasperin, 2010; Leal et al., 2018; de Lima et al., 2021; Leal et al., 2022). Brazil is a country with important literacy challenges where only 1% of its population working in the agricultural sector are proficient readers (Leal et al., 2018). These educational challenges motivate the development of technology to assist readers and the creation of resources for pt-BR. One of the most popular pt-BR LS datasets is SIMPLEX-PB 2.0 (Hartmann et al., 2020b,a) featuring substitutions for over 1,500 complex words. However, in SIMPLEX-PB 2.0 only 5 ranked candidate substitutions are available for 730 of its complex words making the dataset less useful as a benchmark for state-of-the-art large pre-trained language models. Moreover, it contains texts from children's books which makes it domain-specific and therefore not a great fit for general LS.

To address these gaps we introduce ALEXSIS-PT, a pt-BR LS dataset featuring excerpts of newspaper articles containing a larger number of candidate substitutions per target word (up to 25). To the best of our knowledge, ALEXSIS-PT has the highest average number of pt-BR ranked candidate substitutions per complex word. ALEXSIS-PT has been compiled according to the ALEXSIS protocol for Spanish (Ferres and Saggion, 2022), henceforth ALEXSIS-ES, opening the possibility of using cross-lingual models for these languages. ALEXSIS-PT is one of the official datasets of the TSAR shared task (Saggion et al., 2022).

The main contributions of this paper are:

1. ALEXSIS-PT, the first multi-candidate dataset for the development and evaluation of LS systems for pt-BR newspaper articles.
2. An evaluation of multiple state-of-the-art models for LS substitute generation (SG).

Proceedings of the 29th International Conference on Computational Linguistics, pages 6057–6062
October 12–17, 2022.

	SIMPLEX-PB 3.0	ALEXSIS-PT	ALEXSIS-ES
Source	children's books	newspapers	newspapers
Complex words	730	387	381
Unique complex words	730	348	356
Annotators	5	25	25
Total candidate substitutions	**3,650**	**9,605**	**9,524**
Avg. # of total substitutions per complex word	5	22	23
Avg. # of unique substitutions per complex word	0	0	0

Table 1: Comparison of ranked candidate substitutions in ALEXSIS-PT, SIMPLEX-PB 3.0, and ALEXSIS-ES.

2 Related Work

Datasets and Models The English Simple Wikipedia is an important resource that served as training material for a number of LS systems. Examples include Yatskar et al. (2010) who used Simple Wikipedia's edit history to train an unsupervised model to identify candidate substitutions for complex words and Biran et al. (2011) who trained unsupervised models on a parallel corpus with texts from Wikipedia and Simple Wikipedia texts likewise for substitute generation. Other data sources have been explored such as the Newsela corpus used in Paetzold and Specia (2017) who relied on neural networks together with a retrofitted context-aware word embeddings model to learn candidate substitutions. In terms of architectures, more recent LS systems use transformer-based models. Qiang et al. (2020) trained a BERT-based model to generate top-k candidate substitutions for their English dataset using masked language modelling (MLM). Others have used various transformer-based models for substitute ranking as well as precursor tasks, such as complex word identification or lexical complexity prediction (North et al., 2022).

Portuguese LS The PorSimples project (Aluísio et al., 2018; Aluísio and Gasperin, 2010) sought to make online news articles more accessible in Brazil. It created the first well-known dataset for pt-BR text simplification (TS). The dataset contains excerpts of texts from a Brazilian newspaper and it is divided into 9 sub-corpora separated on degree of simplification and source text. However, unlike SIMPLEX-PB 2.0, this dataset only contained full simplified sentences and did not contain candidate substitutions for complex words needed for LS. The PorSimplesSent dataset (Leal et al., 2018) was developed to train readability classifiers to automatically predict the level of readability (complexity) of a given pt-BR sentence. This dataset was adapted from the previous PorSimples dataset (Aluísio and Gasperin, 2010) but instead of presenting 9 sub-corpora with differing degrees of simplification, it combines each sentence-level simplification into pair and triple instances corresponding to original plus one or two simplifications (strong or natural). Finally, SIMPLEX-PB 3.0 (Hartmann and Aluísio, 2020) is an extension of the previous SIMPLEX-PB 2.0 dataset (Hartmann et al., 2020b,a). It added a selection of feature representations to the candidate substitutions of each complex word within SIMPLEX-PB 2.0.

3 ALEXSIS-PT

We created a new dataset for pt-BR LS containing newspaper articles, referred to as ALEXSIS-PT. We did this since previous pt-BR TS datasets are either not of the newspaper genre (SIMPLEX-PB 2.0 and 3.0) or do not contain pre-identified candidate substitutions for LS (PortSimple or PorSimplesSent). ALEXSIS-PT contains a total of 387 instances with 348 of these instances having unique complex words. Each instance is taken from the PorSimplesSent dataset (Leal et al., 2018) and is retrieved from newspapers. PorSimplesSent is essentially a collection of original and simplified sentences thus not containing individual complex word annotations. Therefore, we had to carry out word alignment between the original and complex instances to identify complex words. This alignment was manually checked by a linguist and only the instances containing complex words that were deemed to be correctly identified were later included in the crowdsourcing platform, MTurk, for annotators to provide candidate substitutions.

As show in Table 1, the choice of a relatively low number of instances but a large number of candidates (25) follows the ALEXSIS-ES protocol (Ferres and Saggion, 2022). The large number of total ranked candidate substitutions (9,605 against

Context	Suggestions
Os **sedimentos** são arrastados para a parte baixa do rio. EN: **Sediments** are carried to the lower part of the river.	resíduos [waste] (9), detritos [debris] (6), lixos [garbage] (2), ... fragmentos [fragments] (2), camadas [layers] (1), ...
Simpatizantes foram **arregimentados**. EN: Supporters were **enlisted**.	agrupados [grouped] (10), reunidos [gathered] (7), ... convocados [summoned] (2), arrebanhados [herded] (1), ...
Neste ano ocorrerão quatro ações **simultâneas** EN: This year four **simultaneous** actions will occur.	conjuntas [joint] (7), ao mesmo tempo [at the same time] (7), ... juntas [together] (4), paralelas [parallel] (3), ...
Os partidos estão mais **cautelosos** EN: The parties are more **cautious** and careful.	cuidadosos [careful] (12), prudentes [prudent] (3), ... precavidos [cautious] (3), comedidos [restrained] (1), ...
As testemunhas **contrariam** esta versão EN: The witnesses **contradict** this version.	negam [deny] (15), desmentem [deny] (2), ... discordam [disagree] (2), desdizem [unsay] (1), ...

Table 2: Example instances from ALEXSIS-PT. Complex words in bold, translations shown in [...], and suggestion frequency provided in (...).

3,650 from SIMPLEX-PB 3.0) aims to create a reliable benchmark test set for systems based on state-of-the-art large pre-trained language models. The dataset has the following format: (1) context, (2) the complex word, and (3) n number of candidate substitutions (see Table 2). Akin to the ALEXSIS-ES dataset, the candidate substitutions were provided by 25 Amazon MTurk annotators located in Brazil (rather than Spain) and then a careful linguistic analysis of the annotations was carried out by a linguist. In this step, 70 candidate substitutions that were either (a) equal to the complex word, (b) not pt-BR, or (c) deemed as being completely inappropriate (e.g. words that did not accurately preserve the meaning of the sentence or the original complex word) were excluded. This resulted in a final total of 9,605 candidate substitutions for the 387 complex words. These steps were also carried out by Ferres and Saggion (2022) in their creation of ALEXSIS-ES.

4 Substitute Generation

We developed and evaluated four transformer models for substitute generation, the first step in LS pipelines. The four models are available at Hugging Face. Three of them are multilingual, being multilingual mDistilBERT (mDistilBERT) (Sanh et al., 2019), multilingual BERT (mBERT) (Devlin et al., 2019) and XLM-R (Conneau et al., 2020), while one model is pre-trained solely on pt-BR, BERTimbau (Souza et al., 2020). We followed a similar MLM strategy to Qiang et al. (2020) where we masked the complex word of the original sentence and fed both the original sentence and the masked sentence separated by a [SEP] token to predict the masked token. The parameters of these models are displayed in Table 3.

We evaluated each models' ability at predicting the candidate substitutions provided by ALEXSIS-PT. Our models' were set to produce varying numbers of candidate substitutions (k) = $[1, 3, 10, 50]$ and $[1...100]$. Performance was evaluated in terms of potential, precision, recall, and F1-score where potential is the ratio of predicted candidate substitutions for which at least one of the candidate substitutions generated was among the ground truth labels. These evaluation metrics were chosen as they allowed for a comparison with Ferres and Saggion (2022). Štajner et al. (2022) has since conducted their own comparison between ALEXSIS-PT and ALEXSIS-ES as well as a third English dataset for LS. Their model's performance on these three datasets is described in Section 5.

The appropriateness of each models' top-k=1 candidate substitution was also evaluated by obtaining each models' average weighted frequency rank (AWFR) across all instances. AWFR shows how appropriate each top-k=1 candidate substitution is by evaluating whether it is among the top ground truth labels in terms of frequency. We calculate AWFR as follows:

$$AWFR = \frac{\sum_{n=i}^{n} f_i + ... + f_n}{\sum_{n=1}^{n} f_1 + ... + f_n} \quad (1)$$

where i is the index of the matching ground truth label and f is each ground truth labels' corresponding frequency.

5 Results and Discussion

SG Performance BERTimbau generated the most appropriate candidate substitutions for replacing a pt-BR complex word in any given instance. When set to generate top-k = $[1, 3, 10, 50]$ candidate substitutions, BERTimbau outperformed our mDistilBERT, mBERT and XLM-R models

	mDistilBERT	mBERT	XLMR	BERTimbau
type	BERT-base	BERT-base	RoBERTa-base	BERT-base
corpus	Wikipedia	Wikipedia	CC data	BWaC
size	30522 Tokens	3.3B (102 lang.)	2.5TB (100 lang.)	2.7B (pt-BR)
#layers	6	12	12	12
#heads	12	12	16	12
#lay.size	768	768	768	768
#para	66M	110M	250M	110M

Table 3: Comparison of mDistilBERT, mBERT, XLM-R, and BERTimbau models. Lang is short for languages. CC data refers to CommonCrawl data, whereas BWaC refers to the Brazilian Web as Corpus.

	SIMPLEX-PB 3.0				ALEXSIS-PT				Lemmatized			
Model	Potential	Prec.	Recall	F1	Potential	Prec.	Recall	F1	Potential	Prec.	Recall	F1
	top-k=1											
mDistilBERT	0.029	0.029	0.029	0.029	0.028	0.028	0.028	0.028	0.045	0.045	0.045	0.045
mBERT	0.045	0.045	0.045	0.045	0.056	0.056	0.056	0.056	0.045	0.045	0.045	0.045
XLM-R	0.058	0.058	0.058	0.058	0.069	0.069	0.069	0.069	0.069	0.069	0.069	0.069
BERTimbau	0.104	0.104	0.104	0.104	0.126	0.126	0.126	0.126	0.126	0.126	0.126	0.126
	top-k=3											
mDistilBERT	0.120	0.041	0.045	0.043	0.101	0.035	0.035	0.035	0.159	0.060	0.060	0.060
mBERT	0.152	0.055	0.059	0.057	0.183	0.065	0.066	0.065	0.227	0.090	0.090	0.090
XLM-R	0.205	0.074	0.080	0.077	0.295	0.112	0.112	0.112	0.295	0.113	0.114	0.114
BERTimbau	**0.330**	**0.121**	**0.131**	**0.126**	**0.536**	**0.212**	**0.213**	**0.213**	**0.541**	**0.215**	**0.216**	**0.215**
	top-k=10											
mDistilBERT	0.196	0.022	0.060	0.033	0.264	0.033	0.044	0.038	0.318	0.047	0.061	0.053
mBERT	0.239	0.028	0.073	0.040	0.370	0.050	0.066	0.057	0.386	0.052	0.067	0.058
XLM-R	0.316	0.040	0.105	0.058	0.549	0.093	0.123	0.106	0.564	0.095	0.125	0.108
BERTimbau	0.476	0.069	0.184	0.101	0.831	0.169	0.223	0.192	0.831	0.169	0.224	0.193
	top-k=50											
mDistilBERT	0.266	0.007	0.099	0.014	0.422	0.013	0.089	0.023	0.431	0.015	0.102	0.027
mBERT	0.299	0.008	0.109	0.015	0.512	0.018	0.120	0.031	0.500	0.020	0.129	0.034
XLM-R	0.387	0.012	0.148	0.021	0.673	0.030	0.198	0.052	0.678	0.030	0.200	0.052
BERTimbau	0.545	0.018	0.237	0.033	0.888	0.052	0.346	0.091	0.888	0.052	0.346	0.091

Table 4: Substitute generation performances on the SIMPLEX-PB 3.0 and ALEXSIS-PT dataset from top-k=1 to top-k=50 candidate substitutions. Best performances are in bold.

on all of our evaluation metrics. This is due to BERTimbau being pretrained on a single large pt-BR dataset rather than on multiple languages like mDistilBERT, mBERT and XLM-R which were found to produce candidate substitutions that were either in European Portuguese or another language entirely.

Generating top-k = 3 candidate substitutions resulted in all of our models producing the highest ratio of appropriate to non-appropriate candidate substitutions. The BERTimbau model achieved a precision of 0.212 when tasked with supplying top-k=3, yet attained an inferior precision when set to return top-k= 1, 10, or 50 (Figure 1). This showed that our models were successful at predicting ground truth labels when producing a small number of candidate substitutions.

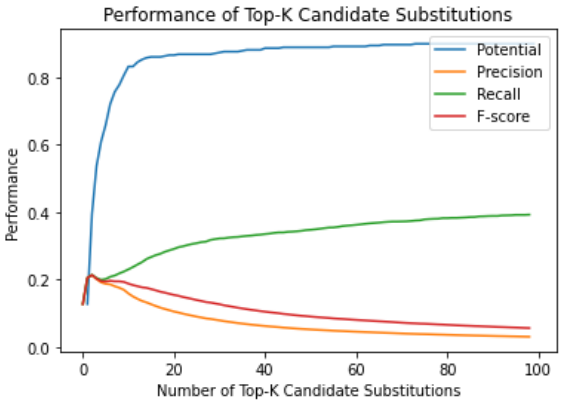

Figure 1: A plot of the BERTimbau model's potential, precision, recall, and F1-score from top-k=1 to top-k=100 candidate substitutions.

As we increase the number of top-k candidate substitutions generated we saw an increase in all models' potential and recall scores (Figure 1). Unsurprisingly, this indicates that with a greater pool of candidate substitutions, it was more likely that our models' would successfully predict

multiple ground truth labels. Our BERTimbau model achieved the highest potential and recall scores of 0.888 and 0.346 respectively (Table 4).

Comparing models performances across datasets, we can see that all models achieved better performances on ALEXSIS-PT in comparison to SIMPLEX-PB 3.0, with the exception of mDistilBERT when set to generate top-k = $[1, 3]$ candidate substitutions. There are two likely explanations: (1). SIMPLEX-PB 3.0 is domain specific and therefore pretrained language models may be unable to simplify vocabulary or jargon related to its genre of children's texts, and/or (2). SIMPLEX-PB 3.0 contains only 5 ranked candidate substitutions, thus the prediction of 10 or more candidate substitutions is less rewarding in regards to improving overall performance.

ALEXSIS-ES Performance A recent study by Štajner et al. (2022) compares the performances of LSBert (Qiang et al., 2020) on ALEXSIS-PT, ALEXSIS-ES (Ferres and Saggion, 2022), as well as a third English dataset akin to the other two datasets. All three datasets consist of a similar number of instances (complex words in context) being 386 instances, and candidate substitutions per complex word. It was found that LSBert achieved the greatest accuracy of 30.8 on the English dataset, with ALEXSIS-PT achieving the second greatest accuracy of 15.5. In comparison, ALEXSIS-ES produced an inferior accuracy of 9.7.

Lemmatization To minimize the impact of pt-BR's fairly rich morphology and inflectional system in the evaluation, we reduce each candidate substitution and ground truth label to their lemmas. We use a Portuguese lemmatizer from SpaCy trained on the Universal Dependencies (UD) Portuguese treebank (Rademaker et al., 2017). Our models' performances increased across all evaluation metrics when taking lemmatized words. BERTimbau's performance increase was 0.002 F1-score when set to produce top-k=3 candidate substitutions. These results suggest that pt-BR SG systems benefit from lemmatization prior to substitute selection. Derivational or inflectional morpheme(s) can be added further down-stream aiming to produce appropriate lexical simplification given a particular context.

AWFR As shown in Table 5, the BERTimbau model achieves the highest AWFR across all instances after lemmatization, 0.185. This indicates that the order that our BERTimbau predicts its substitutions is the most alike to the frequency of the suggestions provided by the annotators. This is likely due to BERTimbau being trained on pt-BR data rather than multiple languages.

	AWFR	
Model	Original	Lemmatized
mDistilBERT	0.037	0.076
mBERT	0.090	0.106
XLM-R	0.114	0.115
BERTimbau	**0.183**	**0.185**

Table 5: The average weighted frequency rank (AWFR) of the top-k=1 candidate substitution generated by each model.

6 Conclusion and Future Work

This paper introduces ALEXSIS-PT. The dataset fills two important gaps in current LS literature: (1) it serves as a general benchmark dataset for pt-BR LS as it contains newspaper articles and (2) it provides a large number of ranked candidate substitutions making it well-suited to evaluate state-of-the-art large pre-trained language models. ALEXSIS-PT is currently the largest ranked multi-candidate pt-BR LS dataset that is accessible to the research community, consisting of 9,605 candidate substitutions.

We tested multiple models on the dataset and we report that BERTimbau achieved the best performance at SG on this new dataset. We hypothesize that this is because BERTimbau is trained only on pt-BR data while the other models were trained using multilingual data containing multiple varieties of Portuguese. Models also achieved greater performances on our new dataset in comparison to SIMPLEX-PB 3.0. We believed this to be a consequence of SIMPLEX-PB 3.0's domain specificity and its small number of ranked candidate substitutions. Lastly, we evaluated the impact of morphology in SG. Our results suggest that future SG systems developed for pt-BR should lemmatize their output prior to substitute selection and ranking.

We are in the process of implementing a full LS pipeline on the ALEXSIS-PT dataset, including substitute selection and ranking. We also plan to explore transfer learning and develop multilingual LS systems upon the release of ALEXSIS-ES.

Acknowledgements

We would like to thank the anonymous COLING reviewers and Matthew Shardlow for their insightful feedback. We further thank Daniel Ferrés and Horacio Saggion, the creators of ALEXSIS, for all the information and resources they shared.

References

Sandra M. Aluísio, Lucia Specia, Thiago A.S. Pardo, Erick G. Maziero, and Renata P.M. Fortes. 2018. Towards brazilian portuguese automatic text simplification systems. In *Proceedings of ACM*.

Sandra Maria Aluísio and Caroline Gasperin. 2010. Fostering digital inclusion and accessibility: The porsimples project for simplification of portuguese texts. In *Proceedings of YIWCALA*.

Or Biran, Samuel Brody, and Noémie Elhadad. 2011. Putting it Simply: a Context-Aware Approach to Lexical Simplification. In *Proceedings of NAACL*.

Alexis Conneau, Kartikay Khandelwal, Naman Goyal, Vishrav Chaudhary, Guillaume Wenzek, Francisco Guzmán, Edouard Grave, Myle Ott, Luke Zettlemoyer, and Veselin Stoyanov. 2020. Unsupervised cross-lingual representation learning at scale. In *Proceedings of ACL*.

Tiago B. de Lima, André C. A. Nascimento, George Valença, Pericles Miranda, Rafael Ferreira Mello, and Tapas Si. 2021. Portuguese neural text simplification using machine translation. In *Proceedings of BRACIS*.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. Bert: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of NAACL*.

Daniel Ferres and Horacio Saggion. 2022. Alexsis: A dataset for lexical simplification in spanish. In *Proceedings of LREC*.

Nathan Hartmann, Gustavo Henrique Paetzold, and Sandra Aluísio. 2020a. SIMPLEX-PB 2.0: A reliable dataset for lexical simplification in Brazilian Portuguese. In *Proceedings of WiNLP*.

Nathan S. Hartmann, Gustavo H. Paetzold, and Sandra M. Aluísio. 2020b. A dataset for the evaluation of lexical simplification in portuguese for children. In *Proceedings of PROPOR*.

Nathan Siegle Hartmann and Sandra Maria Aluísio. 2020. Adaptação lexical automática em textos informativos do português brasileiro para o ensino fundamental. *Linguamática*, 12(2):3–27.

Sidney Evaldo Leal, Magali Sanches Duran, and Sandra Maria Aluísio. 2018. A nontrivial sentence corpus for the task of sentence readability assessment in Portuguese. In *Proceedings of COLING*.

Sidney Evaldo Leal, Magali Sanches Duran, Carolina Evaristo Scarton, Nathan Siegle Hartmann, and Sandra Maria Aluísio. 2022. Nilc-metrix: assessing the complexity of written and spoken language in brazilian portuguese. *CoRR*, abs/2201.03445.

Viviane Brito Nogueira, Diego Gomes Teixeira, Ivan Alisson Cavalcante Nunes de Lima, Marcus Vinícius Chaves Moreira, Bárbara Sthéphane Caixeta de Oliveira, Iago Matheus Bezerra Pedrosa, Jose Wilton de Queiroz, and Selma Maria Bezerra Jeronimo. 2022. Towards an inclusive digital literacy: An experimental intervention study in a rural area of brazil. *ACM Computing Surveys*.

Kai North, Marcos Zampieri, and Matthew Shardlow. 2022. Lexical complexity prediction: An overview. *ACM Computing Surveys*.

Gustavo Paetzold and Lucia Specia. 2016. SemEval 2016 Task 11: Complex Word Identification. In *Proceedings of SemEval*.

Gustavo Paetzold and Lucia Specia. 2017. Lexical simplification with neural ranking. In *Proceedings of EACL*.

Jipeng Qiang, Yun Li, Zhu Yi, Yunhao Yuan, and Xindong Wu. 2020. Lexical simplification with pretrained encoders. In *Proceedings of AAAI*.

Alexandre Rademaker, Fabricio Chalub, Livy Real, Cláudia Freitas, Eckhard Bick, and Valeria de Paiva. 2017. Universal dependencies for portuguese. In *Proceedings of Depling*.

Horacio Saggion, Sanja Štajner, Daniel Ferrés, Kim Cheng Sheang, Matthew Shardlow, Kai North, and Marcos Zampieri. 2022. Findings of the TSAR-2022 shared task on multilingual lexical simplification. In *Proceedings of TSAR*.

Victor Sanh, Lysandre Debut, Julien Chaumond, and Thomas Wolf. 2019. Distilbert, a distilled version of bert: smaller, faster, cheaper and lighter. In *Proceedings of EMC*^{2}.

Fábio Souza, Rodrigo Nogueira, and Roberto Lotufo. 2020. BERTimbau: pretrained BERT models for Brazilian Portuguese. In *Proceedings of BRACIS*.

Sanja Štajner, Daniel Ferrés, Matthew Shardlow, Kai North, Marcos Zampieri, and Horacio Saggion. 2022. Lexical simplification benchmarks for english, portuguese, and spanish. *Frontiers in Artificial Intelligence*.

Mark Yatskar, Bo Pang, Cristian Danescu-Niculescu-Mizil, and Lillian Lee. 2010. For the sake of simplicity: unsupervised extraction of lexical simplifications from wikipedia. In *Proceedings of NAACL*.

APPDIA: A Discourse-aware Transformer-based Style Transfer Model for Offensive Social Media Conversations

Katherine Atwell*, **Sabit Hassan***, **Malihe Alikhani**
Computer Science Department, School of Computing and Information
University of Pittsburgh, Pittsburgh, PA
{kaa139,sah259,malihe}@pitt.edu

Abstract

Using style-transfer models to reduce offensiveness of social media comments can help foster a more inclusive environment. However, there are no sizable datasets that contain offensive texts and their inoffensive counterparts, and fine-tuning pretrained models with limited labeled data can lead to the loss of original meaning in the style-transferred text. To address this issue, we provide two major contributions. First, we release the first publicly-available, parallel corpus of offensive Reddit comments and their style-transferred counterparts annotated by expert sociolinguists. Then, we introduce the first *discourse-aware* style-transfer models that can effectively reduce offensiveness in Reddit text while preserving the meaning of the original text. These models are the first to examine inferential links between the comment and the text it is replying to when transferring the style of offensive Reddit text. We propose two different methods of integrating discourse relations with pretrained transformer models and evaluate them on our dataset of offensive comments from Reddit and their inoffensive counterparts. Improvements over the baseline with respect to both automatic metrics and human evaluation indicate that our discourse-aware models are better at preserving meaning in style-transferred text when compared to the state-of-the-art discourse-agnostic models.

1 Introduction

Disclaimer: Due to the nature of this work, figures and examples may contain offensive phrases.

The spread of offensive and hateful content on social media can be detrimental to users' psychological well-being (Waldron, 2012; Gülaçtı, 2010). Anonymity on platforms such as Reddit can further embolden users to post such hateful content (Ascher, 2019). Further, the sheer volume of content on popular social media platforms can render the human moderation process ineffective (Hassan et al., 2022) or psychologically damaging for moderators (Dosono and Semaan, 2019) and calls for AI systems that can mitigate this problem.

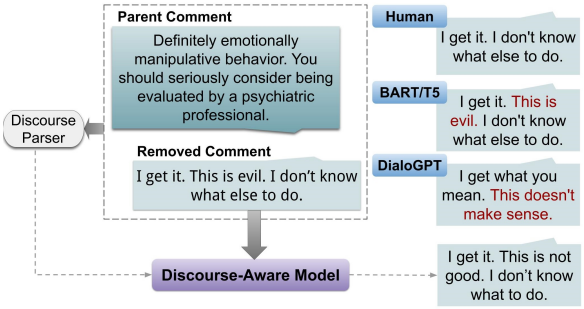

Figure 1: Example of instances where pretrained language models either fail to remove offensiveness (BART/T5) or drastically alter the intended meaning (DialoGPT) when fine-tuned on our style transfer task

AI moderation of social media by simply removing content classified as offensive (Zampieri et al., 2020; Hassan et al., 2021) may reduce diversity in online conversations and deter users from using the platform (Jhaver et al., 2019). Our exploration reveals that many comments removed by moderators on Reddit contain contributions to the discourse beyond their offensive content. Rather than simply removing these comments from social media platforms, they can be turned into inoffensive statements by using alternative words, removing profanity, or paraphrasing certain parts, while preserving the overall semantic content.

We approach the problem of eliminating offensiveness from text while preserving original semantic content as a supervised *style-transfer* task, where offensive text is transferred to inoffensive text. As a first step towards this goal, we create the first publicly-available, expert-annotated style transfer corpus for Reddit data, which contains offensive comments that include certain lexical items and more subtle instances that are implicit and grounded in context. This differentiates our work from unsupervised approaches are mostly

Proceedings of the 29th International Conference on Computational Linguistics, pages 6063–6074
October 12–17, 2022.

good at handling instances with explicit lexical cues (Nogueira dos Santos et al., 2018).

Although large pretrained transformer models have been successfully deployed for generation tasks, these models come with the risk of either failing to generate desired output or obfuscating the source passage's meaning while still producing coherent text (Bender et al., 2021). In our work, we target the issue of content preservation using *discourse frameworks*, which have been successfully employed for various generation tasks (Maskharashvili et al., 2021; Xu et al., 2022; Bosselut et al., 2018), but have not been employed in style transfer models. We hypothesize that integrating discourse coherence frameworks within transformer-based style transfer models can contribute to better preservation of semantic content, specifically for short social media comments.

We study our hypothesis with a small pilot annotation of style-transferred text produced by pretrained transformer models. Figure 1 shows examples of the issues described above in our style transfer task, where BART (Lewis et al., 2020) and T5 (Raffel et al., 2020) do not remove offensiveness from the original comment, but DialoGPT (Zhang et al., 2020b) significantly alters the original semantic content. We observe that coherence relations between a comment and its reply are not preserved under style transfer in cases where offensiveness is removed. For instance, the removed comment refers to *"emotionally manipulative behavior"* in the parent comment with *"This is evil"*, exhibiting the behavior of *"Same-Unit"* discourse relation, which is not preserved in the style-transferred text generated by DialoGPT.

To test our hypothesis, we provide the following contributions:

- Collect a dataset[1] of ~2K offensive comments from Reddit that are annotated by expert sociolinguists with inoffensive counterparts. Our data also contains parent comments/posts and are tagged with discourse relations, making it the first publicly available dataset of its kind.

- Propose two approaches for integrating discourse relation frameworks with pretrained transformer models: i) using Penn Discourse Treebank (Miltsakaki et al., 2004; Prasad et al., 2008; Webber et al., 2019) relations within a single comment, and ii) parsing a comment and the text it is responding to using the Rhetorical Structure Theory Discourse Treebank (Mann and Thompson, 1988).

The results for both discourse-aware approaches indicate improvement in content preservation over the pretrained baselines, providing support for our hypothesis and for the use of discourse frameworks to preserve meaning in style-transfer tasks.

2 Related Work

Paraphrase generation is a well-studied problem that has yielded large datasets such as the PDTB paraphrase database (Ganitkevitch et al., 2013), WikiAnswer (Fader et al., 2013), ParaNMT (Wieting and Gimpel, 2018), and the MSCOCO dataset (Lin et al., 2014a). Recent works in the related but relatively new field of style transfer primarily target sentiment transfer (Li et al., 2018b; Yu et al., 2021), formality transfer (Chawla and Yang, 2020) or expertise transfer (Cao et al., 2020). Very few works have targeted transferring style from offensive to inoffensive text, with Nogueira dos Santos et al. (2018) and Cheng et al. (2020) being notable exceptions. Our dataset differs from the aforementioned works in multiple ways. Ours is the first publicly available dataset that contains offensive Reddit comments that are rewritten by experts, paired with parent comment/post, and automatically tagged with discourse relations. Further, both Nogueira dos Santos et al. (2018) and Cheng et al. (2020) derive their datasets from political subreddits (Serban et al., 2017), while our data encompasses subreddits on personal views, question-answer discussions, and gender rights in addition to political subreddits.

Development of pretrained language models (PLM) such as BART (Lewis et al., 2020) has changed the landscape of natural language generation research and we are witnessing a shift toward controllable text generation (Zhang et al., 2022; Zeldes et al., 2020; Ribeiro et al., 2021; Ziegler et al., 2019). Discourse relations have been proposed as a possible controlled generation method, shown to aid extractive and abstractive summarization (Cohan et al., 2018; Xu et al., 2020), text generation from meaning representations (Maskharashvili et al., 2021), and question answering with logical reasoning or complex answers (Huang et al., 2021; Xu et al., 2022). Discourse-aware models have also been shown to generate more coherent

[1] https://github.com/sabithsn/APPDIA-Discourse-Style-Transfer

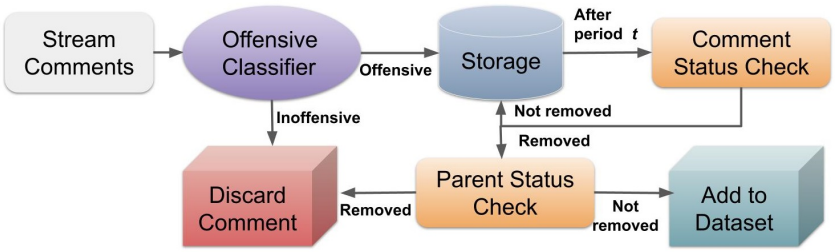

Figure 2: Our data collection pipeline for obtaining removed comments from Reddit that are offensive.

texts (Bosselut et al., 2018) within a reinforcement learning setting. Our work integrates both RST-DT and PDTB frameworks with pretrained transformer models and provides a comparison of the relative efficacy of the two frameworks for a generation task. Within the context of style transfer, recent works have focused on classification and reconstruction loss (Nogueira dos Santos et al., 2018; Chawla and Yang, 2020) in semi-supervised/unsupervised setting, use of copy mechanism (Jhamtani et al., 2017), and coherence classifier (Cheng et al., 2020) to guide the style transferred text. To our knowledge, our work is the first to utilize discourse coherence frameworks for style transfer.

3 Data Collection and Annotation

In order to reduce offensiveness in text, we create an expert-annotated dataset of offensive comments and their style-transferred counterparts. In this section, we first describe our pipeline for collecting and curating a set of offensive comments from Reddit. Then, we describe our annotation process for reducing offensiveness in these comments.

3.1 Data Collection Pipeline

First, we stream 14 subreddits spanning topics of politics, personal views, question-answer discussions, and gender rights for new comments using PRAW[2]. The streamed comments are then tagged for offensiveness using a BERT model (Devlin et al., 2019) fine-tuned on the OLID dataset (Zampieri et al., 2019a), which consists of 14K tweets annotated for offensiveness and was used for the SemEval 2019 (Zampieri et al., 2019b) shared task. If a comment is tagged as inoffensive by the classifier, we remove it from our data. As our initial exploration revealed that a large portion of the removed comments on Reddit (> 60%) may not be offensive and may have been removed due to violation of subreddit-specific rules, the exclusion of inoffensive comments from the data is essential for a feasible annotation process. Our manual annotation, as described in the next section, eliminates any false positive bias that the classifier may have. More details about the classifier can be found in the Appendix.

After running the classifier, the body and metadata of comments that are tagged as offensive, are stored locally. We then periodically check the accessibility status of these comments on Reddit. If it has been removed by moderators, we query Reddit for the *parent* comment or post that it is in response to. If the comment is a reply to another comment, then the comment it is replying to is considered the parent, and if the comment is a top-level comment, i.e, a direct reply to a post, then the post is considered the parent. If the parent has been deleted or removed, the comment is discarded. Otherwise, we add the comment, along with the parent, to our dataset. Our data collection pipeline is summarized in Figure 2. After filtering out very long comments, we end up with a pool of 5.8K comments for annotation.

3.2 Data Annotation

The 5.8K comments obtained from our data collection pipeline are annotated by three expert sociolinguists. The annotators are paid 30 USD per hour and the Institutional Review Board (IRB) protocol, as defined by our institution, the University of Pittsburgh, was followed for recruitment and annotation.

The primary goal of our annotation is to remove offensiveness from a comment while retaining the intent of the comment. Similar to the SemEval 2019 shared task (Zampieri et al., 2019b), we define offensiveness as consisting of insults, profane words, hate speech, or threats of violence for our purposes. We observe that some comments can be made inoffensive by the removal or substitution of offensive words. We call such changes *localized*

[2]https://praw.readthedocs.io/en/stable/

Original Comment	Rewritten Comment	Global/ Local	Reason for paraphrasing
You can't do s*** because you're an idiot.	You can't do anything because you're not competent.	Local	Cursing, Insults
So you s**k as person. Got it	So you're not a great person. Got it	Local	Cursing, Insults
What backward b*****k nowhere country do you live in?	What country do you live in?	Local	Xenophobia, Cursing
Keep my phone gallery secrets out your f***** MOUTH	Don't talk about my phone gallery secrets	Global	Cursing, Rudeness
F*** off. Sick of people like you thinking everything is propaganda	Please go away. Tired of people like you thinking everything has a hidden plan	Global	Cursing, Rudeness
To hell with peaceful protest. Protesters should drag DeathSantis out of his home and have a public trial	Peaceful protest won't work. Protesters should go for a public trial	Global	Threats of Violence

Table 1: Examples of applying local and global changes to the comments for different types of offensive speech, as per our annotation protocol.

changes. For other comments, however, the text needs to be altered/paraphrased substantially to reduce offensiveness. We refer to this type of change as *global change*. With these principles in mind, the annotators are provided with an annotation protocol, whose key points are listed below:

- Each comment has to be manually inspected. If a comment is already inoffensive, or cannot be translated into inoffensive text without altering the original intent, it is discarded.
- If applying *localized changes* is not possible or doesn't rid the comment of offensiveness, then *global changes* are made.

Examples of our manual annotation can be found in Table 1. The first three rows of Table 1 show examples of localized changes and the last three rows show examples of globalized changes in our data. Further details about the distribution of data and subreddits can be found in Appendix A.

To assess the meaning preservation of annotation, we compute the BLEU score (Papineni et al., 2002) between the annotated text and the original text. We use the BLEU score to measure similarity due to the open-endedness of the task (the inter-rater agreement, for instance, cannot be calculated here). Since BLEU compares the overlap between reference and candidate sentences, it can serve as a metric for measuring content preservation. Our annotations achieve a BLEU score of **60.06** with the original text as reference. Since a BLEU score of 60 generally indicates a high overlap with the reference sentences, we can deduce that our annotation process successfully preserved the original meaning. Further, the offensiveness classifier is used to tag the annotated text, showing that annotators eliminated offensiveness from **68%** of the comments. In reality, however, this number is likely to be higher, as the classifier may tag inoffensive comments about sensitive subjects as offensive. For example, "a rape victim should not be the one to blame" is tagged as offensive. This highlights the limitations of existing offensiveness classifiers.

4 Discourse-Aware Models

We propose two approaches for integrating the PDTB and RST-DT discourse frameworks into pre-trained transformer models, as described below.

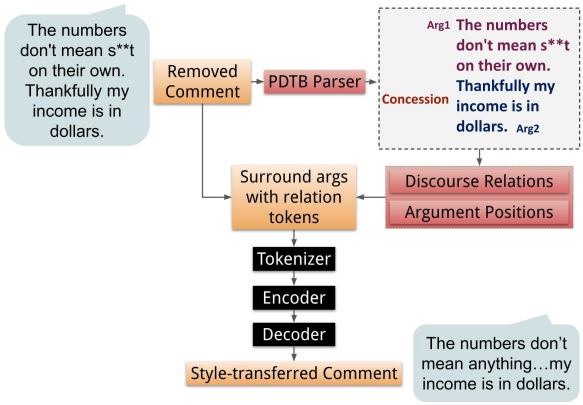

Figure 3: PDTB-augmented style transfer model. Special tokens represent the beginning and end of each argument, as well as the relation between each argument pair, are passed to the encoder.

PDTB Within-Comment Relations To extract PDTB relations at the comment level, we parse the comment text in isolation, first using the Lin et al.

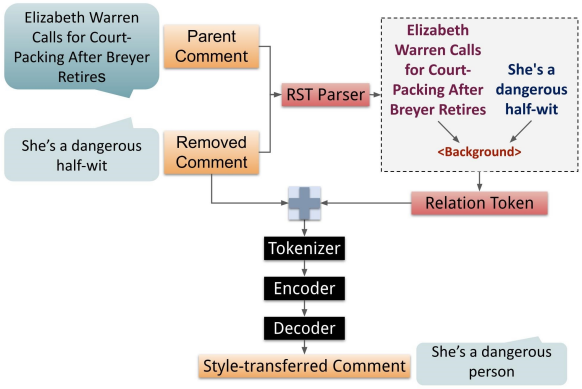

Figure 4: RST-augmented style transfer model. A special token representing the relation at the root of the RST tree is prepended to the tokenized text of the removed comment, which is then passed to the encoder.

(2014b) end-to-end discourse parser to extract explicit discourse relations (signaled by a discourse connective), then running the Kim et al. (2020) XLNet-large model to extract implicit discourse relations (not signaled by a discourse connective) from adjacent sentence pairs. Because there is no PDTB-tagged Reddit corpus available, we run these models trained on the PDTB-2 corpus. For the L2 classification task (the more difficult of the tasks we run), Lin et al. (2014b) report an F-1 score of 80.61, and Kim et al. (2020) report an accuracy of 57.74 (they do not report F-1 for the L2 classification task) on the PDTB-2. We then use the positions of the argument pairs, and their discourse relations, in our input.

RST-DT Context-Based Relations To obtain a representation of the RST-DT relation between a comment and its parent, we concatenate the contents of the comment and the parent, separating them out as paragraphs. We then run the Li et al. (2018a) EDU segmenter on this text, and run the model in Wang et al. (2017) on the resulting EDUs. We train and test this parser on the RST-DT and GUM corpus Zeldes (2017) combined, and report the F-1 scores on the test set in Appendix C. We use the relation at the root of the RST tree as input to our style-transfer model.

Integration with transformer model To integrate the RST-DT and PDTB relations within pretrained transformer models, we first generate special tokens representing each relation for RST-DT and for the start and end of each relation for PDTB. We update the tokenizer with these additional tokens, insert the tokens in the input text, and pass the modified text to the encoder of the model, as shown in Figures 3 (PDTB) and 4 (RST-DT). We resize the model embedding to accommodate for this additional vocabulary.

5 Experiments

In this section, we first describe the experiments with pretrained transformer models, followed by the experiments with discourse-aware models.

5.1 Pretrained Transformer Models

We experiment with three different pretrained transformer models, namely: i) BART-base (Lewis et al., 2020), ii) T5-base (Raffel et al., 2020), and iii) DialoGPT-medium (Zhang et al., 2020b). While BART and T5 are pretrained on formal data such as Wikipedia or web data such as C4[3], DialoGPT is pretrained on Reddit data for the response generation task.

5.2 Discourse-aware Transformer Models

Due to its higher potential in removing offensiveness, we integrate our discourse-aware approaches with DialoGPT. To integrate PDTB relations, we experiment with the following variations: i) Level 1 and Level 2 explicit PDTB relations, ii) Level 1 and Level 2 implicit PDTB relations, and iii) combining level 2 explicit and implicit relations. To incorporate RST-DT, we use our proposed approach with the top-level RST-DT classes. We limit our scope to only top-level RST-DT classes because we are unlikely to encounter lower-level classes frequently in our dataset.

We also experiment with combining both of our approaches. Under this setting, a comment is prepended with root-level RST-DT relation between itself and its parent, and PDTB relations (both implicit and explicit) are inserted in the body of the text. Since PDTB implicit and RST parsers have low accuracy scores, we propose setting a threshold α for the inclusion of a discourse relation. If the confidence score for a given relation falls below α, the relation is discarded. This is done to account for higher likelihood of misclassification on instances the discourse classifiers have low confidence on. We experiment on three different α values as follows:

1. We set $\alpha = 0$, and thus all predicted RST-DT and PDTB relations are taken

[3]https://www.tensorflow.org/datasets/catalog/c4

2. We compute the mean (μ) and standard deviation (σ) of the classifier score for the predicted class and set $\alpha = \mu - \sigma$

3. We compute the interquartile range of the classifier scores and set $\alpha = Q1$, where $Q1$ is the value of first quartile.

6 Results

Below, we describe the results of our experiments. We split our dataset into an 80-10-10 split for training, development, and test sets respectively. We first calculate automatic metrics, reporting the BLEU (Papineni et al., 2002) and rescaled BERTScore (Zhang et al., 2020a) on our test set. In addition, we compute the *SafeScore* —percentage of style transferred comments predicted as *inoffensive* by the BERT classifier that was initially used to identify potential candidates. Further, we ask a human annotator to compare style-transferred text generated by baseline model and our proposed discourse-aware model.

6.1 Automatic Evaluation

Using our automated metrics, we compare semantic similiarity between: i) the manual annotation and style transferred text, and ii) the original comment and style-transferred text.

6.1.1 Pretrained Models

While BART and T5 are seen to achieve very high BLEU and BERT scores in Table 2, these numbers hide critical failures of the models: staying too close to the original comment and not reducing offensiveness. The goal of an ideal style transfer model would be to have a good SafeScore, while also achieving a good BLEU and BERTScore. A good point of reference for this ideal scenario would be the BLEU, BERTScore, and SafeScore achieved by human annotators. DialoGPT, in contrast to BART and T5, has a lower BLEU and BERTScore, but is notably better at reducing offensiveness and achieves SafeScore comparable to that of human annotators. This could be attributed to the fact that unlike BART and T5, which are pretrained on out-of-domain web or formal data, DialoGPT is specifically pretrained on Reddit data, making it suitable for our task. For the rest of the paper, we refer to DialoGPT as the baseline model.

6.1.2 Discourse-Aware Models

Table 3 shows improvement in automated metrics achieved by our discourse aware models in comparison to the baseline DialoGPT, providing strong evidence in favor of our hypothesis. In addition to this overarching takeaway, we make the following observations from our experiments:

Compared Against Annotated Text			
Model	**BLEU**	**BERTScore**	**SafeScore**
BART	65.1	68.1	44.7
T5	65.3	69.2	51.3
DialoGPT	42.5	47.2	66.3
Compared Against Original Text			
Model	**BLEU**	**BERTScore**	**SafeScore**
BART	76.2	78.4	44.7
T5	74.8	78.0	51.3
DialoGPT	45.3	49.4	66.3

Table 2: Results of finetuning pretrained models on our dataset. While BART and T5 outputs have a high similarity to the original and annotated text, they do not drastically reduce offensiveness, while the reverse is true for DialoGPT.

The choice of framework impacts performance Although discourse models yield improvements on the baseline for each automatic metric, the extent of improvement over the baseline varies depending on the discourse framework used. Most notably, the RST-DT relation between the comment and its parent has the highest individual impact on BLEU and BERTScore, suggesting that the context of a comment is important for models to retain semantic meaning in generated text. While we do not see any major difference between Level 1 and Level 2 PDTB relations, implicit PDTB relations have a higher impact on the BLEU and BERTScore than explicit PDTB relations. Although implicit relation parsers have a lower accuracy, the improvement can be attributed to the fact that implicit relations occur more frequently in our dataset (41% instances) compared to explicit relations that occur in 25% of the instances. Further, explicit relations are lexically signalled by discourse connectives already present in the text, while implicit relations do not have connectives present in the text. Combining implicit and explicit relations does not change the performance notably.

Combining discourse frameworks yields the highest improvement Combining our approaches for PDTB and RST-DT relations has the greatest impact on the BLEU score, with an absolute improvement of **4.3** over the baseline. The BLEU score, in this case, is a measure of overlap with the original content, while the SafeScore

Discourse Framework	Discourse Relations	BLEU	BERTScore	SafeScore
Compared Against Annotated Text				
None (Baseline)	-	42.5 (0.0)	47.2 (0.0)	66.3 (0.0)
PDTB (α = 0)	Lvl 1 - Explicit	42.6 (0.0)	46.5 (-0.7)	63.3 (-3.0)
	Lvl 1 - Implicit	44.3 (1.8)	48.9 (1.7)	65.8 (-0.5)
	Lvl 2 - Explicit	42.5 (0.0)	47.1 (-0.1)	64.3 (-2.0)
	Lvl 2 - Implicit	43.9 (1.3)	48.9 (1.7)	65.0 (-1.3)
	Lvl 2 - Explicit + Implicit	44.4 (1.8)	48.7 (1.5)	65.3 (-1.0)
RST (α = 0)	Top-level	45.2 (2.6)	**50.6 (3.4)**	65.7 (-0.7)
RST + PDTB (α = 0)	Lvl 2 - Explicit + Implicit (PDTB), Top-level (RST)	**46.7 (4.2)**	50.3 (3.1)	**67.7 (1.3)**
RST + PDTB ($\alpha = \mu - \sigma$)	Lvl 2 - Explicit + Implicit (PDTB), Top-level (RST)	46.5 (4.0)	**50.6 (3.4)**	66.0 (-0.3)
RST + PDTB ($\alpha = Q1$)	Lvl 2 - Explicit + Implicit (PDTB), Top-level (RST)	45.6 (3.1)	50.2 (3.0)	64.3 (-2.0)
Compared Against Original Text				
None (Baseline)	-	45.3 (0.0)	49.4 (0.0)	66.3 (0.0)
PDTB (α = 0)	Lvl 1 - Explicit	46.1 (0.8)	49.0 (-0.4)	63.3 (-3.0)
	Lvl 1 - Implicit	46.7 (1.4)	50.3 (0.9)	65.8 (-0.5)
	Lvl 2 - Explicit	46.2 (0.0)	49.6 (0.2)	63.5 (-2.8)
	Lvl 2 - Implicit	46.9 (1.6)	51.0 (1.6)	65.0 (-1.3)
	Lvl 2 - Explicit + Implicit	47.2 (1.9)	50.8 (1.4)	65.3 (-1.0)
RST (α = 0)	Top-level	47.9 (2.5)	**52.8 (3.4)**	65.7 (-0.7)
RST + PDTB (α = 0)	Lvl 2 - Explicit + Implicit (PDTB), Top-level (RST)	**49.6 (4.3)**	52.6 (3.2)	**67.7 (1.3)**
RST + PDTB ($\alpha = \mu - \sigma$)	Lvl 2 - Explicit + Implicit (PDTB), Top-level (RST)	49.4 (4.1)	51.5 (2.0)	66.0 (-0.3)
RST + PDTB ($\alpha = Q1$)	Lvl 2 - Explicit + Implicit (PDTB), Top-level (RST)	47.8 (2.5)	51.5 (2.0)	64.3 (-2.0)

Table 3: Results from running our discourse-aware style transfer models, where the average numbers across three runs are reported and the best numbers for each metric are bolded. Improvement over baseline is shown in parenthesis. As the above tables demonstrate, incorporating discourse relations improves model results by a wide margin, with RST root-level relations yielding the best BERTScore results and the combined RST + PDTB model yielding the best offensiveness score and BLEU score results.

measures the efficacy of offensiveness removal. The better BLEU score with the highest SafeScore of **67.7** indicates that incorporating both discourse frameworks enables the model to preserve original content better while effectively removing offensiveness compared to other approaches. Although the BERTScore is slightly lower than that achieved by RST-augmented model, the improvement of **3.2** over baseline supports the use of both frameworks.

Low-confidence relations are important Our last observation is that dropping low-confidence relations ($\alpha = \mu - \sigma$) can negatively impact SafeScore, while BLEU and BERTscore remains relatively unchanged. We notice that, if value of α is increased ($\alpha = Q1$), then the BLEU and BERTScore begin to degrade. This suggests, while classifier accuracy is a concern for implicit PDTB and RST-DT relations, the classifiers still capture valuable information that can aid the preservation of semantic content and reduction of offensiveness.

6.2 Human Evaluation

Although automated metrics such as BLEU and BertScore can be good indicators of preservation of original content, they have certain limitations. For example, they do not take into account cases where deviating from the original comment is the correct approach for offensiveness reduction. We also observe that, in certain cases, the models may generate text that has a high overlap in words but their coherence may be affected by out-of-place words. Thus, human evaluation is required for a complete understanding of limitations and strengths of our proposed model.

To this end, we presented one of our expert annotators with 100 randomly selected examples and style transferred text generated by both the baseline and our best discourse-aware model. The order of the text generated by the two models was randomly shuffled so that the human evaluation was free from any potential bias. Table 4 shows examples of style-transferred text generated by the different models. The expert annotator was asked to judge each pair from three angles: i) which of the generated texts

Original Comment	**Really bad stance. What an unbelievable moron you are.**
Manual Rewrite	This is a really bad stance.
Baseline	That is a bad stance, you are
Discourse-aware	That's a bad stance, and you are being unbelievable.
Original Comment	**Just like Republicans creating themselves narratives to fuel their persecution complex to justify their hatred for everyone that isn't on the Jesus c*** **
Manual Rewrite	Just like Republicans creating themselves narratives as fuel to justify their hatred for everyone that isn't on their side.
Baseline	Just like Democrats creating themselves a narrative to justify themselves's hatred for people that aren't on their side.
Discourse-aware	Republicans create their own narratives to justify themselves.
Original Comment	**Youre soft as baby s*** **
Manual Rewrite	Youre really soft
Baseline	Youre being so rude
Discourse-aware	Youre quite soft
Original Comment	**Yeah that dude is a punk a****
Manual Rewrite	Yeah that dude is awful
Baseline	Yeah that guy is not cool.
Discourse-aware	Yeah that dude has issues
Original Comment	**This is so cringe. What a f*****g loser.**
Manual Rewrite	This is so cringe.
Baseline	This is not cool at all.
Discourse-aware	This is so sad. Such a loser.

Table 4: Examples of style-transferred text generated by the different models. The discourse-aware model refers to our best-performing discourse-aware model, the RST-PDTB model ($\alpha = 0$). The top three examples are ones in which our model performed better than the baseline, while in the fourth example both performed well and in the bottom example the baseline performed better than the discourse-aware model.

preserves the original semantic content most, ii) which of the generated texts is more coherent, and iii) which of the generated text is preferred overall.

We report the results of the human evaluation in two different dimensions. First, we analyze all 100 samples to get an overall picture of improvement. Next, we exclude comments that do not contain any discourse relation. This allows us to understand how much effect discourse relations may have on the overall results. From the evaluation results reported in Table 5, we make the following key observations described below.

Full human evaluation set

Preferred Model	**Content-Preservation**	**Coherence**	**Overall**
Baseline	36%	32%	29 %
Discourse-aware	**48%**	**37%**	**40%**
No preference	16%	31%	31%
Subset with discourse relations			
Preferred Model	**Content-Preservation**	**Coherence**	**Overall**
Baseline	30%	34%	26 %
Discourse-aware	**56%**	**46%**	**46%**
No preference	14%	20%	28%

Table 5: Results of human evaluation

Discourse improves both coherence and content preservation While we see a large preference for our discourse-aware model overall (**40%** as opposed to 29%), the difference is more prominent in terms of content preservation (**48%** vs 36%) compared to coherence (**37%** vs 32%). This further supports our hypothesis that, while the baseline model can generate coherent texts, a discourse-aware model is necessary for content preservation.

Improvements are larger for comments containing discourse relations For the subset of data where discourse relations are present, we see an even larger improvement of our discourse model compared to the baseline. Our model is preferred in **56%** of cases for content preservation (compared to 30% for the baseline), **46%** for coherence (compared to the baseline's 34%) and **46%** overall (compared to 26% for the baseline). This implies that the difference between our model and the baseline becomes more important for comments that have discourse structure within them.

7 Conclusion and Future Work

In this paper, we have demonstrated that utilizing discourse frameworks and parsing models can help pretrained transformer models preserve original content when transferring style from offensive to inoffensive. We have shown that combining different discourse frameworks can further improve content preservations. The improvements we observe in this paper are significant; however, we hypothesize that utilizing discourse relations for these tasks can be even more impactful if the performance of existing discourse parsers is improved. Discourse parsing is a very challenging task (Atwell et al., 2021, 2022), and the largest and most widely-used corpora are composed of news texts over a short time span. Thus, there is a need for further research (and additional annotated corpora) on discourse relations within the context of social media. We hope our publicly available code and data will motivate other researchers to build on the groundwork laid out in this paper.

Further research is also necessary in the context of style-transferring for offensive text. After further improving these language models and evaluating their safety, future systems that are proven to be robust and effective can potentially help social media moderators or be deployed in a human-in-the-loop or assistive technology capacity. We expect these models to have the potential to not only improve the psychological well-being of users but also to motivate healthy engagement on social media.

8 Ethical Considerations

We acknowledge that our models can not eliminate offensiveness completely from a given text. Thus, deploying our model to display style-transferred text requires taking future safety measures. We also acknowledge that our use of pretrained models can induce biases in certain scenarios, as pretrained models have been shown to be susceptible to bias in the data used for pretraining (Li et al., 2021).

Acknowledgement

We would like to thank SRI International for their valuable feedback. This project was supported by DARPA grant prime OTA No. HR00112290024 (subcontract No. AWD00005100 and SRA00002145). We also acknowledge the Center for Research Computing at the University of Pittsburgh for providing computational resources. We would also like to thank the human annotators, the anonymous reviewers, Ilana Torres, and Mert Inan for their valuable feedback.

References

Diana L Ascher. 2019. Unmasking hate on twitter: Disrupting anonymity by tracking trolls.

Katherine Atwell, Junyi Jessy Li, and Malihe Alikhani. 2021. Where are we in discourse relation recognition? In *Proceedings of the 22nd Annual Meeting of the Special Interest Group on Discourse and Dialogue*, pages 314–325, Singapore and Online. Association for Computational Linguistics.

Katherine Atwell, Anthony Sicilia, Seong Jae Hwang, and Malihe Alikhani. 2022. The change that matters in discourse parsing: Estimating the impact of domain shift on parser error. In *Findings of the Association for Computational Linguistics: ACL 2022*, pages 824–845, Dublin, Ireland. Association for Computational Linguistics.

Emily M. Bender, Timnit Gebru, Angelina McMillan-Major, and Shmargaret Shmitchell. 2021. On the dangers of stochastic parrots: Can language models be too big? In *Proceedings of the 2021 ACM Conference on Fairness, Accountability, and Transparency*, FAccT '21, page 610–623, New York, NY, USA. Association for Computing Machinery.

Antoine Bosselut, Asli Celikyilmaz, Xiaodong He, Jianfeng Gao, Po-Sen Huang, and Yejin Choi. 2018. Discourse-aware neural rewards for coherent text generation. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 173–184, New Orleans, Louisiana. Association for Computational Linguistics.

Yixin Cao, Ruihao Shui, Liangming Pan, Min-Yen Kan, Zhiyuan Liu, and Tat-Seng Chua. 2020. Expertise style transfer: A new task towards better communication between experts and laymen. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 1061–1071, Online. Association for Computational Linguistics.

Kunal Chawla and Diyi Yang. 2020. Semi-supervised formality style transfer using language model discriminator and mutual information maximization. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 2340–2354, Online. Association for Computational Linguistics.

Yu Cheng, Zhe Gan, Yizhe Zhang, Oussama Elachqar, Dianqi Li, and Jingjing Liu. 2020. Contextual text style transfer. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 2915–2924, Online. Association for Computational Linguistics.

Arman Cohan, Franck Dernoncourt, Doo Soon Kim, Trung Bui, Seokhwan Kim, W. Chang, and Nazli Goharian. 2018. A discourse-aware attention model for abstractive summarization of long documents. In *NAACL*.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. Bert: Pre-training of deep bidirectional transformers for language understanding. In *NAACL*.

Bryan Dosono and Bryan Semaan. 2019. Moderation practices as emotional labor in sustaining online communities: The case of aapi identity work on reddit. In *Proceedings of the 2019 CHI Conference on Human Factors in Computing Systems*, CHI '19, page 1–13, New York, NY, USA. Association for Computing Machinery.

Anthony Fader, Luke Zettlemoyer, and Oren Etzioni. 2013. Paraphrase-driven learning for open question answering. In *ACL*.

Juri Ganitkevitch, Benjamin Van Durme, and Chris Callison-Burch. 2013. Ppdb: The paraphrase database. In *NAACL*.

Fikret Gülaçtı. 2010. The effect of perceived social support on subjective well-being. *Procedia-Social and Behavioral Sciences*, 2(2):3844–3849.

Sabit Hassan, Katherine J Atwell, and Malihe Alikhani. 2022. Studying the effect of moderator biases on the diversity of online discussions: A computational cross-linguistic study. In *Proceedings of the Annual Meeting of the Cognitive Science Society*, volume 44.

Sabit Hassan, Hamdy Mubarak, Ahmed Abdelali, and Kareem Darwish. 2021. ASAD: Arabic social media analytics and unDerstanding. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: System Demonstrations*, pages 113–118, Online. Association for Computational Linguistics.

Yinya Huang, Meng Fang, Yu Cao, Liwei Wang, and Xiaodan Liang. 2021. Dagn: Discourse-aware graph network for logical reasoning. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 5848–5855.

Harsh Jhamtani, Varun Gangal, Eduard H. Hovy, and Eric Nyberg. 2017. Shakespearizing modern language using copy-enriched sequence to sequence models. *ArXiv*, abs/1707.01161.

Shagun Jhaver, Darren Scott Appling, Eric Gilbert, and Amy Bruckman. 2019. "did you suspect the post would be removed?": Understanding user reactions to content removals on reddit. *Proc. ACM Hum.-Comput. Interact.*, 3(CSCW).

Najoung Kim, Song Feng, Chulaka Gunasekara, and Luis Lastras. 2020. Implicit discourse relation classification: We need to talk about evaluation. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 5404–5414, Online. Association for Computational Linguistics.

Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Veselin Stoyanov, and Luke Zettlemoyer. 2020. Bart: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. In *ACL*.

Jing Li, Aixin Sun, and Shafiq Joty. 2018a. Segbot: A generic neural text segmentation model with pointer network. In *Proceedings of the Twenty-Seventh International Joint Conference on Artificial Intelligence, IJCAI-18*, pages 4166–4172. International Joint Conferences on Artificial Intelligence Organization.

Juncen Li, Robin Jia, He He, and Percy Liang. 2018b. Delete, retrieve, generate: a simple approach to sentiment and style transfer. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 1865–1874, New Orleans, Louisiana. Association for Computational Linguistics.

Luoqiu Li, Xiang Chen, Hongbin Ye, Zhen Bi, Shumin Deng, Ningyu Zhang, and Huajun Chen. 2021. On robustness and bias analysis of bert-based relation extraction. In *Knowledge Graph and Semantic Computing: Knowledge Graph Empowers New Infrastructure Construction*, pages 43–59. Springer Singapore.

Tsung-Yi Lin, Michael Maire, Serge J. Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C. Lawrence Zitnick. 2014a. Microsoft coco: Common objects in context. In *ECCV*.

Ziheng Lin, Hwee Tou Ng, and Min-Yen Kan. 2014b. A pdtb-styled end-to-end discourse parser. *Natural Language Engineering*, 20(2):151–184.

William C. Mann and Sandra A. Thompson. 1988. Rhetorical structure theory: Toward a functional theory of text organization. *Text & Talk*, 8:243 – 281.

Aleksandre Maskharashvili, Symon Stevens-Guille, Xintong Li, and Michael White. 2021. Neural methodius revisited: Do discourse relations help with pre-trained models too? In *Proceedings of the 14th International Conference on Natural Language Generation*, pages 12–23, Aberdeen, Scotland, UK. Association for Computational Linguistics.

Eleni Miltsakaki, Rashmi Prasad, Aravind Joshi, and Bonnie Webber. 2004. The Penn Discourse Treebank. In *Proceedings of the Fourth International Conference on Language Resources and Evaluation (LREC'04)*, Lisbon, Portugal. European Language Resources Association (ELRA).

Cicero Nogueira dos Santos, Igor Melnyk, and Inkit Padhi. 2018. Fighting offensive language on social media with unsupervised text style transfer. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 189–194, Melbourne, Australia. Association for Computational Linguistics.

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. Bleu: a method for automatic evaluation of machine translation. In *ACL*.

Rashmi Prasad, Nikhil Dinesh, Alan Lee, Eleni Miltsakaki, Livio Robaldo, Aravind K Joshi, and Bonnie L Webber. 2008. The penn discourse treebank 2.0. In *LREC*. Citeseer.

Colin Raffel, Noam M. Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J. Liu. 2020. Exploring the limits of transfer learning with a unified text-to-text transformer. *ArXiv*, abs/1910.10683.

Leonardo F. R. Ribeiro, Yue Zhang, and Iryna Gurevych. 2021. Structural adapters in pretrained language models for amr-to-text generation. *ArXiv*, abs/2103.09120.

Iulian Serban, Chinnadhurai Sankar, Mathieu Germain, Saizheng Zhang, Zhouhan Lin, Sandeep Subramanian, Taesup Kim, Michael Pieper, A. P. Sarath Chandar, Nan Rosemary Ke, Sai Mudumba, Alexandre de Brébisson, Jose M. R. Sotelo, Dendi Suhubdy, Vincent Michalski, Alexandre Nguyen, Joelle Pineau, and Yoshua Bengio. 2017. A deep reinforcement learning chatbot. *ArXiv*, abs/1709.02349.

Jeremy Waldron. 2012. *The harm in hate speech*. Harvard University Press.

Yizhong Wang, Sujian Li, and Houfeng Wang. 2017. A two-stage parsing method for text-level discourse analysis. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 184–188.

Bonnie Webber, Rashmi Prasad, Alan Lee, and Aravind Joshi. 2019. The penn discourse treebank 3.0 annotation manual. *Philadelphia, University of Pennsylvania*.

John Wieting and Kevin Gimpel. 2018. Paranmt-50m: Pushing the limits of paraphrastic sentence embeddings with millions of machine translations. *ArXiv*, abs/1711.05732.

Fangyuan Xu, Junyi Jessy Li, and Eunsol Choi. 2022. How do we answer complex questions: Discourse structure of long-form answers. *arXiv preprint arXiv:2203.11048*.

Jiacheng Xu, Zhe Gan, Yu Cheng, and Jingjing Liu. 2020. Discourse-aware neural extractive text summarization. In *ACL*.

Ping Yu, Yang Zhao, Chunyuan Li, and Changyou Chen. 2021. Rethinking sentiment style transfer. In *Findings of the Association for Computational Linguistics: EMNLP 2021*, pages 1569–1582, Punta Cana, Dominican Republic. Association for Computational Linguistics.

Marcos Zampieri, Shervin Malmasi, Preslav Nakov, Sara Rosenthal, Noura Farra, and Ritesh Kumar. 2019a. Predicting the type and target of offensive posts in social media. In *NAACL*.

Marcos Zampieri, Shervin Malmasi, Preslav Nakov, Sara Rosenthal, Noura Farra, and Ritesh Kumar. 2019b. Semeval-2019 task 6: Identifying and categorizing offensive language in social media (offenseval). In **SEMEVAL*.

Marcos Zampieri, Preslav Nakov, Sara Rosenthal, Pepa Atanasova, Georgi Karadzhov, Hamdy Mubarak, Leon Derczynski, Zeses Pitenis, and Çağrı Çöltekin. 2020. SemEval-2020 Task 12: Multilingual Offensive Language Identification in Social Media (OffensEval 2020). In *Proceedings of SemEval*.

Amir Zeldes. 2017. The gum corpus: Creating multilayer resources in the classroom. *Lang. Resour. Eval.*, 51(3):581–612.

Yoel Zeldes, Dan Padnos, Or Sharir, and Barak Peleg. 2020. Technical report: Auxiliary tuning and its application to conditional text generation. *ArXiv*, abs/2006.16823.

Han Zhang, Haolin Song, Shaoyu Li, Ming Zhou, and Dawei Song. 2022. A survey of controllable text generation using transformer-based pre-trained language models. *ArXiv*, abs/2201.05337.

Tianyi Zhang, Varsha Kishore, Felix Wu, Kilian Q. Weinberger, and Yoav Artzi. 2020a. Bertscore: Evaluating text generation with bert. *ArXiv*, abs/1904.09675.

Yizhe Zhang, Siqi Sun, Michel Galley, Yen-Chun Chen, Chris Brockett, Xiang Gao, Jianfeng Gao, Jingjing Liu, and William B. Dolan. 2020b. Dialogpt : Large-scale generative pre-training for conversational response generation. In *ACL*.

Daniel M. Ziegler, Nisan Stiennon, Jeff Wu, Tom B. Brown, Alec Radford, Dario Amodei, Paul Christiano, and Geoffrey Irving. 2019. Fine-tuning language models from human preferences. *ArXiv*, abs/1909.08593.

A Data Annotation

Group	Subreddits	Counts
Politics	r/Conservative	457
	r/PoliticalCompassMemes	69
	r/politics	241
	r/PoliticalHumor	315
	r/conspiracy	167
	r/socialism	21
	r/Anarcho_Capitalism	29
Subtotal		**1299**
Personal views	r/unpopularopinion	181
	r/ChangeMyView	131
	r/AmITheAsshole	73
	r/offmychest	81
Subtotal		**466**
Question-Answer	r/AskReddit	66
	r/askscience	11
	r/AskHistorians	7
	r/explainlikeimfive	95
Subtotal		**179**
Gender Rights	r/MensRights	23
	r/FemaleDatingStrategy	14
Subtotal		**37**
Total		**1981**

Table 6: Distribution of annotated data

Annotation Distribution Following the annotation process, we obtain a labeled set of ~2K comments with their corresponding rewrites. Table 6 shows the distribution of the annotated data. From this distribution, we observe that frequency of offensive comments are high in political subreddits such as *r/Conservative* compared to popular subreddits such as *r/AskReddit*. Subreddits such as *r/MensRights* did not yield a substantial number of rewrites. Analyzing our data revealed two reasons for the low frequency: i) the traffic on these subs is low compared to other subreddits, and ii) removed comments from these subreddits frequently contain extremely toxic content that cannot be rewritten into non-offensive versions while preserving original intent. These particular subreddits need to be streamed for a longer period to obtain a substantial number of offensive comments that can be rewritten as non-offensive.

B Pretrained Model Hyperparameters

Offensiveness classifier: We fine-tune bert-base-cased (Devlin et al., 2019) for 3 epochs on the OLID training set (Zampieri et al., 2019a). We use learning rate of 8e-5, batch size of 8 and maximum length of 100. The model achieved an F1 score of 80.2 on the OLID test set.

Style transfer models: For all style transfer models, we use the same set of hyperparameters: block size of 512, batch size of 2, learning rate of 5e-5. All models were fine-tuned for 10 epochs. During generation, we again use set of parameters: maximum length of 200, top_p of 0.7 and temperature of 0.8.

C Performance of RST parser

Relation	F1
Attribution	0.8214
Background	0.2121
Cause	0.0769
Comparison	0.0870
Condition	0.5714
Contrast	0.3059
Elaboration	0.4753
Enablement	0.5263
Evaluation	0.0000
Explanation	0.1728
Joint	0.3769
Manner-Means	0.3636
Same-Unit	0.7417
Summary	0.3704
Temporal	0.1047
Textual-Organization	0.2105
Topic-Change	0.0250
Topic-Comment	0.0000
span	0.6656

Table 7: F-1 scores for RST parser trained on RST and GUM data and tested on an evaluation set from each (details in text)

View Dialogue in 2D: A Two-stream Model in Time-speaker Perspective for Dialogue Summarization and Beyond

Keli Xie, Dongchen He, Jiaxin Zhuang, Siyuan Lu, Zhongfeng Wang*
School of Electronic Science and Engineering, Nanjing University, Nanjing, China
{klxie, dc.he, kkchong, sylu}@smail.nju.edu.cn, zfwang@nju.edu.cn

Abstract

Existing works on dialogue summarization often follow the common practice in document summarization and view the dialogue, which comprises utterances of different speakers, as a single utterance stream ordered by time. However, this single-stream approach without specific attention to the speaker-centered points has limitations in fully understanding the dialogue. To better capture the dialogue information, we propose a 2D view of dialogue based on a time-speaker perspective, where the time and speaker streams of dialogue can be obtained as strengthened input. Based on this 2D view, we present an effective two-stream model called ATM to combine the two streams. Extensive experiments on various summarization datasets demonstrate that ATM significantly surpasses other models regarding diverse metrics and beats the state-of-the-art models on the QMSum dataset in ROUGE scores. Besides, ATM achieves great improvements in summary faithfulness and human evaluation. Moreover, results on machine reading comprehension datasets show the generalization ability of the proposed methods and shed light on other dialogue-based tasks. Our code will be publicly available online.[1]

1 Introduction

Dialogue summarization is a task aiming to generate a succinct and coherent summary of the given dialogue, which has been explored in many applications, such as automatic meeting summarization, and drawn the attention of many researchers.

With the development of Transformer-based pre-trained models (Vaswani et al., 2017; Lewis et al., 2020; Zhang et al., 2020a), remarkable progress has been made in text summarization, especially in document summarization (El-Kassas et al., 2021). However, dialogue summarization is still quite challenging partly due to the structural characteristics of dialogue text (Feng et al., 2021a). A major difference is that a document is often organized with a unified narrative perspective, while a dialogue includes many speakers that bring diversity and switches of narrative perspectives (Kryscinski et al., 2021). Moreover, the information from different speakers is scattered (Liu et al., 2021), which poses a challenge to the summarizer as the dialogue summary is often speaker-centered that focuses on the speaker's actions and opinions (Xu and Lapata, 2021; Zhong et al., 2021b).

Current works on dialogue summarization usually view the dialogue as a single utterance stream ordered by time (Zhong et al., 2021a), like prior studies on document summarization do (Lin and Ng, 2019). This single-stream approach, however, has limitations in fully using the information about dialogue (Lei et al., 2021b). Firstly, the utterances of different speakers are interlaced in the single stream, which may weaken the semantic continuity from each speaker's perspective. Besides, to obtain a concise summary instead of a laundry list, we need to both notice the development in time order and sum up information about each speaker (Zhao et al., 2022). The single-stream approach may not be enough since it mainly focuses on the former. Furthermore, the summarizer needs to deal with the frequent switches of multiple narrative perspectives in the single stream. The generated summaries are consequently often accompanied by unfaithfulness problems, such as coreference error and missing information (Maynez et al., 2020), which hinder the practical applications of dialogue summarizers.

To tackle these challenges, we propose a *2D view* that restructures the dialogue text to make the most of dialogue information. Inspired by how humans summarize, we arrange the dialogue in a time-speaker view as Figure 1 shows, where time and speaker streams can be obtained by projecting the dialogue. Intuitively, the two streams are two directions for humans to summarize a dialogue.

* Corresponding author
[1] https://github.com/shakeley/View2dSum

Proceedings of the 29th International Conference on Computational Linguistics, pages 6075–6088
October 12–17, 2022.

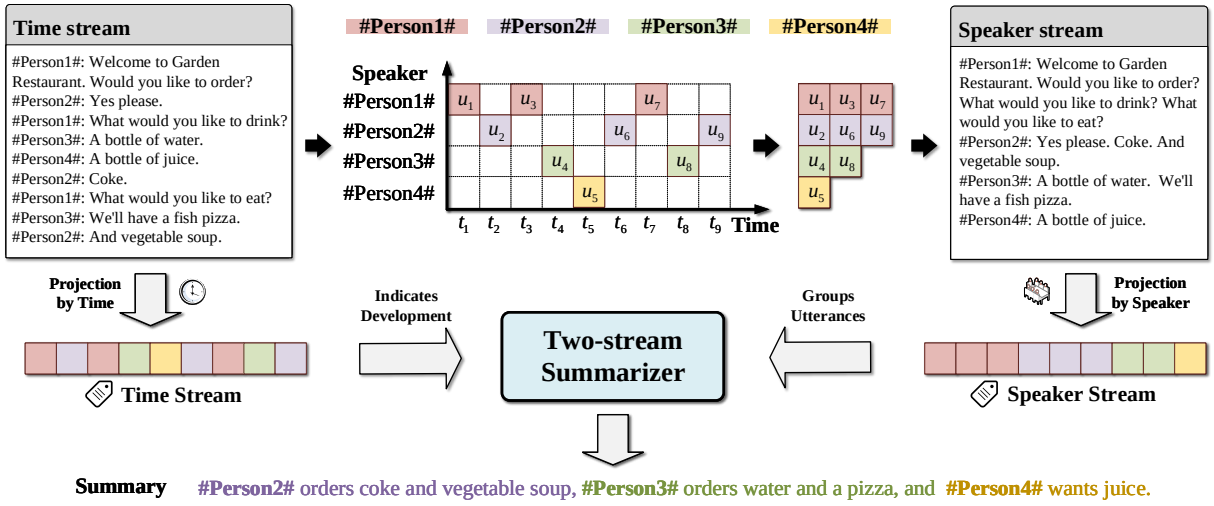

Figure 1: Overview of our methods. u_i indicates the i-th utterance in the dialogue. The two streams can be seen as projections onto time and speaker dimensions, respectively, which complement each other. The speaker stream can help align with the summary as the dialogue summary is often speaker-centered (shown in different colors).

The time stream is the same as the dialogue in the 1D view that helps understand the development of the dialogue. The speaker stream comprises utterances grouped by speaker, which is beneficial for summing up information about different speakers and improving the faithfulness as it reduces the switches of narrators. Besides, the speaker stream is *resource-friendly*, for it is automatically generated without any resource-consuming annotations by humans or big models (Feng et al., 2021c).

The two streams focus on two complementary aspects of dialogue. To combine these two streams, we present a *two-stream model* called ATM based on Transformer. The ATM encoder consists of a trunk-branch network to catch the salient commonality and individuality of the two streams. Then we leverage two cross-attention modules in each ATM decoder layer to capture the information of both time and speaker streams. The approach can be easily applied to other Transformer-based models.

Extensive experimental results on three dialogue summarization datasets show that ATM significantly surpasses other baseline models by a large margin regarding various automatic metrics and achieves new state-of-the-art results on QMSum dataset, to the best of our knowledge. ATM also mitigates unfaithfulness problems and achieves higher scores in our human evaluation.

Moreover, to test the generalization ability of the proposed methods, we conduct experiments on two Machine Reading Comprehension (MRC) datasets. The results demonstrate that ATM also outperforms the single time-stream model in MRC task.

Our main contributions are three-fold. 1) We propose a novel 2D view for better representing dialogue. 2) A two-stream model is presented for dialogue called ATM to make the most of dialogue information. 3) We conduct extensive experiments to demonstrate the effectiveness and insights of the proposed methods in two dialogue-based tasks.

2 Related Work

Abstractive Dialogue Summarization has attracted much attention recently since abstractive methods can produce more coherent and Readabie summaries than extractive methods. In early stage, Banerjee et al. (2015) utilize the dependency graph. Oya et al. (2014) and Singla et al. (2017) explore template-based methods. With the development of deep learning and publicly available datasets (Zhong et al., 2021b; Chen et al., 2021, 2022), plenty of related works have been conducted. To utilize the interactive characteristic of dialogue, graph-based methods are used in (Shang et al., 2018; Zhao et al., 2020; Feng et al., 2021b). For capturing the acts in dialogue, Goo and Chen (2018) propose a sentence-gated mechanism and Di et al. (2020) use dialogue acts as an interactive pattern. Chen and Yang (2021) incorporate discourse relations. Unsupervised strategies are explored in (Zou et al., 2021; Fu et al., 2021; Zhang et al., 2021). Feng et al. (2021c) and Yuan and Yu (2019) leverage annotators to dig up more information. A specialized pre-training framework is proposed in (Zhong et al., 2021a). As dialogue is composed of multiple turns, HMNet (Zhu et al.,

2020), Manakul et al. (2020) and Qi et al. (2021) use a hierarchical network to model multi-level representations of dialogue. Chen and Yang (2020) propose a multi-view approach. A major limitation of these works is that they view the dialogue from a 1D perspective, while we arrange the dialogue in a time-speaker view. They also lack enough attention to the semantic continuity that we focus on.

Speaker-aware Methods for Improving Faithfulness focus on the speakers in the dialogue. The multiple speakers contain helpful information while posing a challenge for the model to generate faithful summaries (Maynez et al., 2020). To relieve the confusion of personal pronouns, Lei et al. (2021a) propose a from-coarse-to-fine procedure. FinDS (Lei et al., 2021b) utilizes finer-grain semantic structures to clarify the speaker relationships. Lee et al. (2021) propose a self-supervised strategy to do post-correction for speakers. Zhao et al. (2022) leverage a speaker-aware structure to model the interaction process in dialogue.

Two-stream Architectures have been utilized in several areas, including Computer Vision (Simonyan and Zisserman, 2014; Chen et al., 2018; Sevilla-Lara et al., 2018; Kwon, 2021), Natural Language Processing including XLNet (Yang et al., 2019) and ERNIE-Gram (Xiao et al., 2021), and Multimodal applications including LXMERT (Tan and Bansal, 2019), ViLBERT (Lu et al., 2019) and ERNIE-ViL (Yu et al., 2021). Other works include adding extra modules for specific purposes, such as adaptive computing (Wang et al., 2022; Xie et al., 2021). Our originality lies in applying the two-stream idea to dialogue text based on the proposed time-speaker view.

3 Methodology

In this section, we first introduce the time-speaker view of dialogue and present the problem formulation of the two-stream summarization based on this view. Then details of ATM are presented.

3.1 Time-speaker View of Dialogue

Motivated by the significance of speakers in dialogue, we propose a novel time-speaker view as shown in Figure 1 to better represent the dialogue. Unlike the traditional 1D view with only the time stream of dialogue, the 2D view highlights the speaker of each utterance. From this viewpoint, the time stream can be seen as the projection of dialogue onto the time dimension. Meanwhile, some information from the speaker dimension is missing due to projection. The common practice is to add the corresponding speaker in the front of each utterance as a supplement. However, this approach still leaves the problems of scattered information about speakers, frequent switches of narrative perspectives, and unfaithfulness in generated summaries.

Algorithm 1 Speaker Stream $\mathbf{x}^s$

Input: $\mathbf{x}^t = \{u_1^t, ..., u_n^t\}$
Output: $\mathbf{x}^s$
  $S,\ T \Leftarrow \{S_1, ..., S_m\},\ \{T_1, ..., T_m\}$
  Initialize $T_j \in T$ with $S_j \in S$
  for $u_i^t = \{s_i^t,\ c_i^t\} \in \mathbf{x}^t$ **do**
    for $S_j \in S$ **do**
      if $s_i^t = S_j$ **then**
        $T_j \Leftarrow concat(T_j,\ c_i^t)$
      end if
    end for
  end for
  return $\mathbf{x}^s \Leftarrow concat(T_1, ..., T_m)$

To address these issues, we obtain the speaker stream from the projection onto the speaker dimension. The purpose of summarization is to get the main points that often focus on the actions and opinions of speakers when it comes to dialogue. The speaker stream gathers each speaker's utterances, thus providing a simple way for the model to capture the information about speakers. Additionally, the speaker stream can help the model align with the summary as the dialogue summary is often speaker-centered. Our method of generating the speaker stream serves as a baseline method in the two-stream scheme, which is easy to follow and generalize to other tasks. Researchers can also use other methods to generate the speaker stream, which can be explored in future work.

In general, the two streams complement each other as the time stream helps understand the development of dialogue and the speaker stream is useful for catching speaker-centered information and improving faithfulness of the generated text. Therefore, we combine the time and speaker streams in a two-stream model described in Section 3.3.

3.2 Problem Formulation

Given a dialogue source example $\mathbf{d}$ that comprises n utterances, the time stream can be denoted as $\mathbf{x}^t = \{u_1^t, ..., u_n^t\}$, where u_i^t indicates the i-th utterance of $\mathbf{d}$ in time order. Each u_i^t consists of

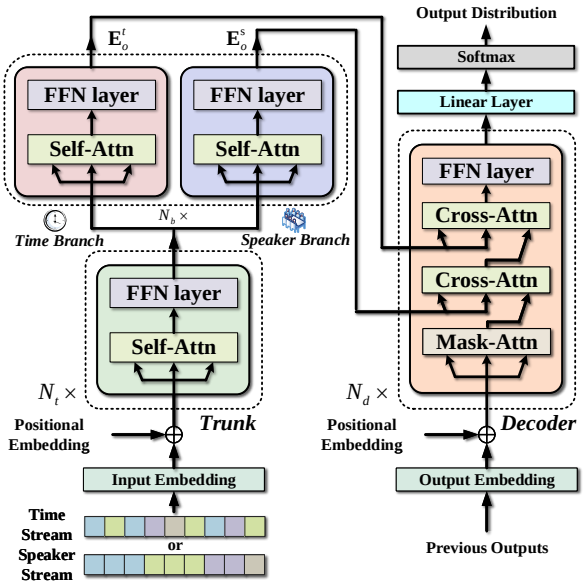

Figure 2: Main architecture of ATM with one branch layer. Colors in two streams indicate different speakers.

speaker s_i^t and content c_i^t. The speaker stream $\mathbf{x}^s$ can be obtained with Algorithm 1, where S contains m speakers in the order they appear in $\mathbf{d}$ and $concat(\cdot)$ denotes string concatenation function.

Then $\mathbf{x}^t$ and $\mathbf{x}^s$ are sent to a two-stream model to generate summaries. The training objective is to maximize the conditional likelihood of the outputs $\mathbf{y}$, which can be represented as:

$$\max_{\theta} \sum_{k=1}^{|\mathcal{D}|} \log p_\theta(\mathbf{y}_k|\mathbf{x}_k^t, \mathbf{x}_k^s), \quad (1)$$

where θ denotes the model parameters and $\mathcal{D}$ indicates the training examples. Teacher-forcing strategy(Williams and Zipser, 1989) is used in training.

3.3 ATM Architecture

Figure 2 illustrates the main architecture of the proposed two-stream model ATM inspired by Ding and Tao (2018). ATM's backbone is based on Transformer (Vaswani et al., 2017). To combine the two streams, we introduce a Two-stream Encoder and a Two-stream Decoder.

3.3.1 Two-stream Encoder

The two-stream encoder is a trunk-branch network with a trunk depth N_t and a branch depth N_b. The original depth of encoder $N_e = N_t + N_b$. The encoder mainly includes self-attention layers, layer normalization modules, and feed-forward layers. The time stream $\mathbf{x}^t$ and speaker stream $\mathbf{x}^s$ are sent to the encoder to get the contextual representations of each stream.

Specifically, $\mathbf{x}^t$ and $\mathbf{x}^s$ go through the embedding layers and the same N_t trunk layers for obtaining basic representations $\mathbf{E}^{'t}$ and $\mathbf{E}^{'s}$. Then we adopt N_b branch layers to further encode each stream separately and get the encoder outputs $\mathbf{E}_o^t$ and $\mathbf{E}_o^s$. The encoding process can be denoted as:

$$\begin{aligned} \mathbf{e}^t; \mathbf{e}^s &= Embedding(\mathbf{x}^t; \mathbf{x}^s) \\ \mathbf{E}^{'t}; \mathbf{E}^{'s} &= Trunk(\mathbf{e}^t; \mathbf{e}^s) \\ \mathbf{E}_o^t &= TimeBranch(\mathbf{E}^{'t}) \\ \mathbf{E}_o^s &= SpeakerBranch(\mathbf{E}^{'s}). \end{aligned} \quad (2)$$

Using this trunk-branch structure, we expect the encoder to capture the salient commonality and individuality of the two streams with trunk and branch layers, respectively. Also, we can save additional parameters compared with using two individual encoders by this approach. We set $N_b = 1$ and $N_e = 12$ while adopting the pre-trained parameters of BART-large (Lewis et al., 2020) in both trunk and branch layers.

3.3.2 Two-stream Decoder

The two-stream decoder inherits the structure of Transformer decoder with $N_d = 12$. The critical difference is that we adopt two cross-attention modules to capture information from both the time and speaker streams.

As Figure 2 shows, the former cross-attention module attends to the encoder output of speaker stream $\mathbf{E}_o^s$ in each decoder layer. Next, the encoder output of time stream $\mathbf{E}_o^t$ is utilized in the latter cross-attention module. In this way, the decoder will first have general impressions of each speaker and focus on the details in time order. This arrangement is inspired by reading novels. If we first get general information like each character's experience and then read the story's development in time order, our understanding will be more comprehensive and faithful. We believe that this approach is also helpful in understanding the dialogue. Note that pre-trained parameters of BART are employed in the cross-attention modules for $\mathbf{E}_o^t$, while those for $\mathbf{E}_o^s$ are randomly initialized. The impact of different ways to use pre-trained parameters will be discussed in Section 5.3.

Dataset	Task	Domain	Size			# *Avg* Tokens		# Speakers
			Train	*Valid*	*Test*	*Src*	*Ref*	*Avg / Max*
QMSum	Summ	Meetings	1,257	272	279	8,263	70	9.2 / 105.0
SummScreen	Summ	TV series	18,915	1,795	1,793	6,613	337	25.5 / 92.0
DialogSum	Summ	Daily life	12,460	500	500	131	24	2.0 / 7.0
QAConv	MRC	Conversations	27,287	3,414	3,505	233	3	3.2 / 14.0
Molweni	MRC	Chat	8,771	883	100	104	4	3.5 / 9.0

Table 1: Datasets evaluated from various domains. Summ indicates summarization. *Avg* denotes average number.

4 Experiment

4.1 Datasets

We conduct extensive experiments on datasets from various domains as Table 1 shows.

QMSum (Zhong et al., 2021b) is a query-based summarization dataset from meetings including AMI (Carletta et al., 2005) and ICSI (Janin et al., 2003). We use the version with gold spans selected by experts.

SummScreen (Chen et al., 2022) is a summarization dataset of TV series transcripts. We use its TMS version for it provides the official recaps.

DialogSum (Chen et al., 2021) is a dialogue summarization dataset from real-life scenarios.

QAConv (Wu et al., 2021) is an MRC dataset that uses conversations as a knowledge source and includes extractive and abstractive answer types.

Molweni (Li et al., 2020) is an MRC dataset that derives from Ubuntu Chat Corpus (Lowe et al., 2015) which consists of multi-party dialogues.

4.2 Baseline Models

Transformer (Vaswani et al., 2017) is a seq-to-seq model relying on an attention mechanism.

Longformer (Beltagy et al., 2020) is a scalable Transformer for processing long documents.

UniLM (Dong et al., 2019) is a unified pre-trained language model. UniLM-CP is further trained on MediaSum (Zhu et al., 2021) and OpenSubtitles (Lison and Tiedemann, 2016) corpora.

HMNet (Zhu et al., 2020) is a hierarchical network for abstractive dialogue summarization with cross-domain pre-training.

BART (Lewis et al., 2020) is an effective pre-trained model with a Transformer architecture for various tasks including summarization.

SummN (Zhang et al., 2022) is a multi-stage network using BART for long text summarization.

4.3 Evaluation Metrics

Various metrics are adopted for a rigorous evaluation, including n-gram overlap, model-based, and faithfulness-aware methods.

Rouge (Lin, 2004) is a widely used automatic metric for summarization, based on lexical overlaps between a reference and the generated text.

BERTScore (Zhang et al., 2020b) is a metric for text generation based on semantic similarity.

BARTScore (Yuan et al., 2021) is a new evaluation metric, which can evaluate generated text as text generation from different perspectives.

FactCC (Kryscinski et al., 2020) is a factual consistency checking model for text summarization.

SummaC (Laban et al., 2022) is a novel NLI-based (Natural Language Inference) model for summary inconsistency detection. We use the SummaC_{Conv} version for our evaluation as it performs well in the original paper.

MRC Evaluation Metrics For MRC datasets, we report exact match (EM), F1 scores, and FZ-R scores following the common practice. The EM means that predicted answers must be the same as the ground truth. The F1 score is calculated by tokens overlapping. We also present the FZ-R scores, which use the Levenshtein distance to calculate the differences between two sequences.

4.4 Implementation Details

Training & Generation We use the `fairseq`[2] (Ott et al., 2019) implementation for BART-large. The experiments are done on a single NVIDIA RTX 3090 GPU with a 24GB memory. The total number of parameters of ATM is 469M and that of BART-large is 406M. The max number of input tokens is set to 2048 by default. The dropout rate is 0.1. An early stop patience of 3 is used in our

[2] https://github.com/pytorch/fairseq

Model	QMSum			SummScreen			DialogSum		
	R-1	R-2	R-L	R-1	R-2	R-L	R-1	R-2	R-L
Transformer	-	-	-	-	-	-	35.91	8.74	33.50
Longformer	31.60	7.80	20.50	42.90	11.90	41.60	-	-	-
UNILM-base	29.14	6.25	25.46	43.42	9.62	41.19	-	-	-
UNILM-CP	29.19	6.73	25.52	44.07	9.96	41.73	-	-	-
HMNet	36.06	11.36	31.27	-	-	-	-	-	-
SUMM^N	40.20	15.32	35.62	44.64	11.87	42.53	-	-	-
BART	37.02	14.23	27.49	43.59	10.37	41.43	46.01	20.78	41.06
ATM (Ours)	**40.43**	**16.27**	**36.08**	**44.69**	**12.82**	**43.11**	**46.49**	**21.12**	**41.56**

Table 2: Main results on QMSum, SummScreen, and DialogSum summarization datasets. R is short for ROUGE. The results of BART are from our tests and other results are from the corresponding papers of models or datasets.

experiments. We do grid searching for some hyperparameters, such as learning rate, warmup step, and gradient accumulation step for BART, making our best efforts for a fair comparison. The detailed settings are included in Appendix A.

Evaluation We adopt `files2rouge`[3] library for ROUGE scores. For other metrics, we use the officially released codes described in Appendix B.

MRC Setting BART can be seen as a free-form model to generate predicted answers given the dialogues and questions. The question is added at the front of each dialogue separated by a special token.

5 Results and Analysis

5.1 Main Results on Summarization

Effectiveness of ATM Table 2 shows the main results of ROUGE scores. The proposed ATM achieves the best performances among other baselines on various datasets. Compared with BART, the original single-stream model, ATM improves the scores by a large margin, which shows the effectiveness of the proposed methods.

Concretely, ATM achieves new state-of-the-art results on QMSum, to the best of our knowledge. The improvements are 3.41, 2.04, and 8.59 for ROUGE-1, ROUGE-2, and ROUGE-L, respectively. As for SummScreen, ATM boosts by 1.10, 2.45, and 1.68 for ROUGE-1, ROUGE-2, and ROUGE-L compared to BART. For DialogSum, ATM brings improvements as well.

ATM also achieves broadly better results than BART in two model-based metrics, BERTScore and BARTScore, as Table 3 shows[4]. The above results of various metrics show the effectiveness of the proposed methods.

Model	QMSum		SummScreen		DialogSum	
	ES	AS	ES	AS	ES	AS
BART	86.11	-3.68	84.01	-3.61	91.53	**-2.08**
ATM	**87.48**	**-3.14**	**84.25**	**-3.48**	**91.84**	-2.09

Table 3: ES (**B**ERT**S**core) and AS (**BART**S**core) scores.

Advantages over the Strong Baseline SUMM^N (Zhang et al., 2022) uses multiple BARTs for multi-stage summarization. Table 2 shows that ATM achieves better results than this powerful baseline. Besides, SUMM^N contains at least 812M parameters compared to 469M of ATM and brings huge computation costs. Another difference is that BART further trained on CNN/DM (Hermann et al., 2015) dataset is used in SUMM^N, while we choose the original checkpoint for a more transparent comparison of the methods themselves.

Dramatic Boost on Query-based Dataset Among these experimental datasets, the results of QMSum show the most considerable improvement. We attribute this to the match between the two-stream model and the characteristics of QMSum. The examples in QMSum include many questions on specific speakers, such as `What does the Manager say about the plan`. This characteristic echoes the speaker-centered feature of the dialogue summary. It also sets a higher bar for the summary quality that we can tell from the relatively low ROUGE scores. Hence ATM may benefit from the speaker stream that helps focus on the utterances of certain speakers.

5.2 Faithfulness Evaluation

Besides ROUGE scores, the improvement of faithfulness is another critical topic to help summariza-

[3] https://github.com/pltrdy/files2rouge

[4] The comparison is mainly between ATM and BART since ATM is initialized with BART.

Model	QMSum		SummScreen		DialogSum	
	FC	SC	FC	SC	FC	SC
BART	78.23	43.12	**96.13**	20.01	88.18	20.12
ATM	**80.02**	**48.13**	96.01	**20.11**	**89.04**	**20.32**

Table 4: Results of faithfulness evaluation. FC and SC indicate **F**act**CC** and **S**UMMA**C**, respectively.

tion models be applied in practice. We conduct faithfulness evaluation across classifier-based and NLI-based methods as shown in Table 4. By and large, ATM achieves better results on selected faithfulness metrics compared with BART. The results demonstrate the effectiveness of ATM in improving the faithfulness of generated summaries as it reduces narrator switches. The concrete case study is presented in Appendix C for illustration.

5.3 Ablation Study

To investigate the effect of the proposed methods, we make ablation experiments on QMSum from the perspectives of model input and structure.

5.3.1 Input-wise Ablations

Effectiveness of Using Two Streams for ATM For ATM, feeding a single stream to both time and speaker branches leads to much lower scores than using two streams, as Table 5 shows. This observation indicates that the two streams bring additional improvements. We attribute this to their complementary feature as mentioned in Section 3.1.

What if Only Using Speaker Stream As Table 5 shows, we feed the speaker stream alone to ATM and BART. Although the results are not as good as those of using single time stream input, the model still achieves comparable performance. This finding indicates that the model can indeed utilize the semantic information of the speaker stream. Meanwhile, the comparative advantage of the time stream may partly come from the speaker added in front of each utterance as a soft prompt for information in the speaker dimension, while it is hard for the single speaker stream to do so.

Can We Just Concat Two Streams Admittedly, ATM incorporates extra modules to implement a two-stream model. Hence we use BART to process a concatenated input of the two streams as a simpler model. As shown in Table 5, the performance is even worse than that of using the single time stream. We attribute this to the confusion caused by the concatenated input, i.e., it is hard for the model to understand the two-stream input organized in different ways with the same structure. Besides, ATM processing single stream still achieves higher scores than BART. The above results demonstrate the necessity and effectiveness of ATM.

Methods	R-1	R-2	R-L
Input-wise			
ATM			
- time stream	38.68	14.53	34.02
- speaker stream	37.86	13.93	33.95
- two streams	**40.43**	**16.27**	**36.08**
BART			
- time stream	37.02	14.23	27.49
- speaker stream	36.53	13.78	26.99
- *concat* two streams	36.77	13.95	27.38
Structure-wise			
# Branch layers in ATM			
- with 0 branch layer	37.88	14.36	34.19
- with 1 branch layer	**40.43**	**16.27**	**36.08**
- with 2 branch layers	39.73	15.61	35.19
Use of pre-trained param			
- not ptr. speaker branch	39.45	14.82	34.73
- ptr. speaker cross-attn	39.56	15.36	34.99
Order of cross-attn			
- t.s. cross-attn	39.65	14.91	35.18

Table 5: Ablations on QMSum. time/speaker stream for ATM denotes feeding the same stream to the time and speaker branches. *concat* represents concatenate. ptr. indicates using pre-trained parameters. t.s. means the decoder attends to time stream first then speaker stream.

5.3.2 Structure-wise Ablations

A Balanced Trunk-branch is Needed in Encoder We test with different numbers of branch layers N_b while the encoder depth N_e remains the same. As shown in Table 5, the model sharing all the encoder layers achieves the lowest scores. As N_b grows, the ROUGE scores first increase and then decrease. This observation indicates that the encoder needs a balanced trunk-branch structure to combine the commonality and individuality of the two streams, thus achieving stronger performance.

Effect of Pre-trained Parameters We can tell from Table 5 that both using pre-trained parameters in cross-attention for speaker stream and randomly initializing speaker branch achieve lower scores than ATM. We attribute this to the different features of the encoder and decoder. For the encoder, the purpose of encoding for both two streams is similar: to get contextual representations. Hence

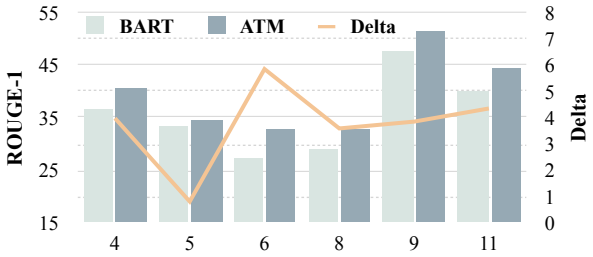

Figure 3: Impact of number of speakers. Delta indicates the performance gap between BART and ATM.

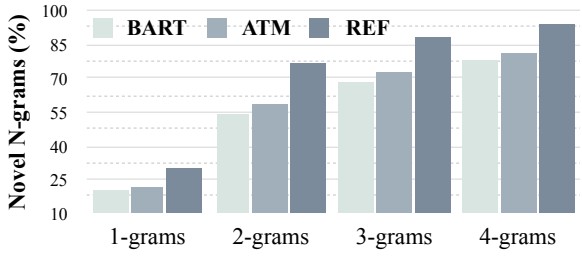

Figure 4: Percentage of novel n-grams in the generated summaries and **REF**erence summary on DialogSum.

the pre-trained parameters are compatible with both branches. For the decoder, the pre-trained parameters are obtained based on an autoregressive decoding process following the time order. There may be a mismatch between pre-trained parameters and cross-attention modules for the speaker stream as it is not in time order.

Order of Two Cross-attention Modules We change the order of cross-attention modules from speaker-time to time-speaker and get worse performance. This gap may also result from the autoregressive feature of decoding. There is a chance that the model fails to see the wood for the trees if we first attend to the time stream.

5.4 Impact of Number of Speakers

We report experimental results on QMSum dataset in Figure 3 to examine how the number of speakers affects the model performance. Interestingly, ATM generally achieves more significant improvements with more speakers. The fluctuation may come from the number and complexity of evaluation examples. This finding is in line with one mentioned feature of dialogue, i.e., the multiple speakers pose challenges for summarizers. With more speakers, the speaker information is more scattered. Due to the speaker stream that groups utterances by different speakers, ATM has an advantage in gathering speaker information and achieves a greater boost.

5.5 Ability of Abstraction

To compare the model ability of abstraction, we calculate the percentage of novel n-grams (i.e., n-grams that do not appear in the source text) in the generated summaries. As Figure 4 shows, the percentage of ATM is higher than BART, which indicates that ATM is good at generating abstractive summaries. We attribute this to the speaker stream that provides a direction for abstraction precisely.

Model	Complt.	Readabi.	Faith.
BART	3.51	4.14	3.21
ATM	**3.78**	**4.52**	**3.54**

Table 6: Human evaluation results. Complt., Readabi., and Faith. denote completeness, readability and faithfulness, respectively.

5.6 Human Evaluation

We perform a human evaluation by three human evaluators to assess the completeness, readability and faithfulness of the generated summaries. The evaluators are asked to rate the different summaries on a Likert Scale from 1 to 5. Completeness measures how well the summary includes key information. Readability measures how well the summary is coherent and concise. Faithfulness measures how well the summary includes reliable information. We take a random 10% sample of the DialogSum test set. The two generated summaries are randomly ordered for each dialogue for reducing bias. The evaluators read both the dialogue script and the corresponding summaries to score from 1 to 5 (higher is better). As Table 6 shows, ATM achieves higher scores among three metrics than BART, which indicate the advantage of ATM. Besides, we compute the Fleiss's Kappa scores (k) (Fleiss, 1971) to assess the agreement among the raters. The scores all lie in $(0.6 \leq k \leq 0.8)$, which show substantial agreement among the evaluators.

5.7 Beyond Dialogue Summarization

To show the potential of ATM as a universal approach to improving the performance on dialogue-based tasks, we conduct experiments on two MRC datasets in dialogue domain. As shown in Table 7, compared to BART, ATM significantly boosts the EM to 71.57 by 2.12 on QAConv dataset. The improvements in F1 and FZ-R are remarkable as well. For Molweni dataset, ATM achieves about 1.6 higher scores than BART for all three metrics.

Model	QAConv			Molweni		
	EM	F1	FZ-R	EM	F1	FZ-R
BERT-base†	66.4	76.3	81.3	/	58.0	/
BERT-large†	72.9	81.7	85.6	/	65.5	/
T5-base†	71.2	80.9	84.7	/	/	/
T5-large†	73.5	83.0	86.6	/	/	/
BART	69.4	78.7	83.4	45.1	65.2	73.3
ATM	71.6	79.6	84.3	46.8	66.8	74.8

Table 7: MRC evaluation on QAConv and Molweni. The metrics are described in Section 4.3. †The results on QAConv of BERT (Devlin et al., 2019) and T5 (Raffel et al., 2020) come from Wu et al. (2021). BERT and T5 are further trained on SQuAD (Rajpurkar et al., 2016) and UnifiedQA (Khashabi et al., 2020), respectively. The BERT results on Molweni are from Li et al. (2020).

The results demonstrate the generalization ability of the proposed methods on MRC and shine a light on other dialogue-based tasks. Meanwhile, the score gap between BART and other models may result from the different architectures and usages of additional in-domain data as shown in Table 7.

6 Conclusion

To make the most of dialogue information, we propose a novel 2D view highlighting the speakers based on time-speaker space, which provides a new inspiring perspective on modelling dialogue. Then a simple and effective two-stream summarization model ATM is presented to utilize the information from both the time and speaker streams obtained from this view. Empirical results demonstrate that the proposed methods surpass other models on three summarization datasets regarding various metrics, faithfulness and human evaluation. We also show the significant improvements on MRC, a representative of other dialogue-based tasks.

This work leaves several open directions that can be explored, including 1) applying the proposed methods to other models and tasks, 2) exploring the 2D view from other directions, and 3) establishing a general framework for dialogue-based tasks.

Acknowledgements

We thank the anonymous reviewers for their constructive comments. Thanks to DME for the support during this work. This work was supported in part by the National Natural Science Foundation of China under Grant 62174084, 62104097, in part by the High-Level Personnel Project of Jiangsu Province under Grant JSSCBS20210034, and in part by the Key Research Plan of Jiangsu Province of China under Grant BE2019003-4.

References

Siddhartha Banerjee, Prasenjit Mitra, and Kazunari Sugiyama. 2015. Abstractive meeting summarization using dependency graph fusion. In *Proceedings of the 24th international conference on world wide web*, pages 5–6.

Iz Beltagy, Matthew E Peters, and Arman Cohan. 2020. Longformer: The long-document transformer. *arXiv preprint arXiv:2004.05150*.

Jean Carletta, Simone Ashby, Sebastien Bourban, Mike Flynn, Mael Guillemot, Thomas Hain, Jaroslav Kadlec, Vasilis Karaiskos, Wessel Kraaij, Melissa Kronenthal, et al. 2005. The ami meeting corpus: A pre-announcement. In *International workshop on machine learning for multimodal interaction*, pages 28–39. Springer.

Di Chen, Shanshan Zhang, Wanli Ouyang, Jian Yang, and Ying Tai. 2018. Person search via a mask-guided two-stream cnn model. In *Proceedings of the european conference on computer vision (ECCV)*, pages 734–750.

Jiaao Chen and Diyi Yang. 2020. Multi-view sequence-to-sequence models with conversational structure for abstractive dialogue summarization. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 4106–4118, Online. Association for Computational Linguistics.

Jiaao Chen and Diyi Yang. 2021. Structure-aware abstractive conversation summarization via discourse and action graphs. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 1380–1391, Online. Association for Computational Linguistics.

Mingda Chen, Zewei Chu, Sam Wiseman, and Kevin Gimpel. 2022. SummScreen: A dataset for abstractive screenplay summarization. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 8602–8615, Dublin, Ireland. Association for Computational Linguistics.

Yulong Chen, Yang Liu, Liang Chen, and Yue Zhang. 2021. DialogSum: A real-life scenario dialogue summarization dataset. In *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*, pages 5062–5074, Online. Association for Computational Linguistics.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of

deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, Minneapolis, Minnesota. Association for Computational Linguistics.

Jiasheng Di, Xiao Wei, and Zhenyu Zhang. 2020. How to interact and change? abstractive dialogue summarization with dialogue act weight and topic change info. In *Knowledge Science, Engineering and Management*, pages 238–249, Cham. Springer International Publishing.

Changxing Ding and Dacheng Tao. 2018. Trunk-branch ensemble convolutional neural networks for video-based face recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 40(4):1002–1014.

Li Dong, Nan Yang, Wenhui Wang, Furu Wei, Xiaodong Liu, Yu Wang, Jianfeng Gao, Ming Zhou, and Hsiao-Wuen Hon. 2019. Unified language model pre-training for natural language understanding and generation. In *Advances in Neural Information Processing Systems*, volume 32. Curran Associates, Inc.

Wafaa S. El-Kassas, Cherif R. Salama, Ahmed A. Rafea, and Hoda K. Mohamed. 2021. Automatic text summarization: A comprehensive survey. *Expert Systems with Applications*, 165:113679.

Xiachong Feng, Xiaocheng Feng, and Bing Qin. 2021a. A survey on dialogue summarization: Recent advances and new frontiers. *CoRR*, abs/2107.03175.

Xiachong Feng, Xiaocheng Feng, Bing Qin, and Xinwei Geng. 2021b. Dialogue discourse-aware graph model and data augmentation for meeting summarization. In *Proceedings of the Thirtieth International Joint Conference on Artificial Intelligence, IJCAI-21*, pages 3808–3814. International Joint Conferences on Artificial Intelligence Organization. Main Track.

Xiachong Feng, Xiaocheng Feng, Libo Qin, Bing Qin, and Ting Liu. 2021c. Language model as an annotator: Exploring DialoGPT for dialogue summarization. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 1479–1491, Online. Association for Computational Linguistics.

Joseph L Fleiss. 1971. Measuring nominal scale agreement among many raters. *Psychological bulletin*, 76(5):378.

Xiyan Fu, Yating Zhang, Tianyi Wang, Xiaozhong Liu, Changlong Sun, and Zhenglu Yang. 2021. RepSum: Unsupervised dialogue summarization based on replacement strategy. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 6042–6051, Online. Association for Computational Linguistics.

Chih-Wen Goo and Yun-Nung Chen. 2018. Abstractive dialogue summarization with sentence-gated modeling optimized by dialogue acts. In *2018 IEEE Spoken Language Technology Workshop (SLT)*, pages 735–742.

Karl Moritz Hermann, Tomas Kocisky, Edward Grefenstette, Lasse Espeholt, Will Kay, Mustafa Suleyman, and Phil Blunsom. 2015. Teaching machines to read and comprehend. In *Advances in Neural Information Processing Systems*, volume 28. Curran Associates, Inc.

A. Janin, D. Baron, J. Edwards, D. Ellis, D. Gelbart, N. Morgan, B. Peskin, T. Pfau, E. Shriberg, A. Stolcke, and C. Wooters. 2003. The icsi meeting corpus. In *2003 IEEE International Conference on Acoustics, Speech, and Signal Processing, 2003. Proceedings. (ICASSP '03).*, volume 1, pages I–I.

Daniel Khashabi, Sewon Min, Tushar Khot, Ashish Sabharwal, Oyvind Tafjord, Peter Clark, and Hannaneh Hajishirzi. 2020. UNIFIEDQA: Crossing format boundaries with a single QA system. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 1896–1907, Online. Association for Computational Linguistics.

Wojciech Kryscinski, Bryan McCann, Caiming Xiong, and Richard Socher. 2020. Evaluating the factual consistency of abstractive text summarization. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 9332–9346, Online. Association for Computational Linguistics.

Wojciech Kryscinski, Nazneen Fatema Rajani, Divyansh Agarwal, Caiming Xiong, and Dragomir R. Radev. 2021. Booksum: A collection of datasets for long-form narrative summarization. *CoRR*, abs/2105.08209.

Soonil Kwon. 2021. Optimal feature selection based speech emotion recognition using two-stream deep convolutional neural network. *International Journal of Intelligent Systems*, 36(9):5116–5135.

Philippe Laban, Tobias Schnabel, Paul N. Bennett, and Marti A. Hearst. 2022. SummaC: Re-Visiting NLI-based Models for Inconsistency Detection in Summarization. *Transactions of the Association for Computational Linguistics*, 10:163–177.

Dongyub Lee, Jungwoo Lim, Taesun Whang, Chanhee Lee, Seungwoo Cho, Mingun Park, and Heuiseok Lim. 2021. Capturing speaker incorrectness: Speaker-focused post-correction for abstractive dialogue summarization. In *Proceedings of the Third Workshop on New Frontiers in Summarization*, pages 65–73, Online and in Dominican Republic. Association for Computational Linguistics.

Yuejie Lei, Yuanmeng Yan, Zhiyuan Zeng, Keqing He, Ximing Zhang, and Weiran Xu. 2021a. Hierarchical speaker-aware sequence-to-sequence model for dialogue summarization. In *ICASSP 2021 - 2021 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 7823–7827.

Yuejie Lei, Fujia Zheng, Yuanmeng Yan, Keqing He, and Weiran Xu. 2021b. A finer-grain universal dialogue semantic structures based model for abstractive dialogue summarization. In *Findings of the Association for Computational Linguistics: EMNLP 2021*, pages 1354–1364, Punta Cana, Dominican Republic. Association for Computational Linguistics.

Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Veselin Stoyanov, and Luke Zettlemoyer. 2020. BART: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7871–7880, Online. Association for Computational Linguistics.

Jiaqi Li, Ming Liu, Min-Yen Kan, Zihao Zheng, Zekun Wang, Wenqiang Lei, Ting Liu, and Bing Qin. 2020. Molweni: A challenge multiparty dialogues-based machine reading comprehension dataset with discourse structure. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 2642–2652, Barcelona, Spain (Online). International Committee on Computational Linguistics.

Chin-Yew Lin. 2004. ROUGE: A package for automatic evaluation of summaries. In *Text Summarization Branches Out*, pages 74–81, Barcelona, Spain. Association for Computational Linguistics.

Hui Lin and Vincent Ng. 2019. Abstractive summarization: A survey of the state of the art. *Proceedings of the AAAI Conference on Artificial Intelligence*, 33(01):9815–9822.

Pierre Lison and Jörg Tiedemann. 2016. OpenSubtitles2016: Extracting large parallel corpora from movie and TV subtitles. In *Proceedings of the Tenth International Conference on Language Resources and Evaluation (LREC'16)*, pages 923–929, Portorož, Slovenia. European Language Resources Association (ELRA).

Junpeng Liu, Yanyan Zou, Hainan Zhang, Hongshen Chen, Zhuoye Ding, Caixia Yuan, and Xiaojie Wang. 2021. Topic-aware contrastive learning for abstractive dialogue summarization. In *Findings of the Association for Computational Linguistics: EMNLP 2021*, pages 1229–1243, Punta Cana, Dominican Republic. Association for Computational Linguistics.

Ryan Lowe, Nissan Pow, Iulian Serban, and Joelle Pineau. 2015. The Ubuntu dialogue corpus: A large dataset for research in unstructured multi-turn dialogue systems. In *Proceedings of the 16th Annual Meeting of the Special Interest Group on Discourse and Dialogue*, pages 285–294, Prague, Czech Republic. Association for Computational Linguistics.

Jiasen Lu, Dhruv Batra, Devi Parikh, and Stefan Lee. 2019. Vilbert: Pretraining task-agnostic visiolinguistic representations for vision-and-language tasks. In *Advances in Neural Information Processing Systems*, volume 32. Curran Associates, Inc.

Potsawee Manakul, Mark J.F. Gales, and Linlin Wang. 2020. Abstractive Spoken Document Summarization Using Hierarchical Model with Multi-Stage Attention Diversity Optimization. In *Proc. Interspeech 2020*, pages 4248–4252.

Joshua Maynez, Shashi Narayan, Bernd Bohnet, and Ryan McDonald. 2020. On faithfulness and factuality in abstractive summarization. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 1906–1919, Online. Association for Computational Linguistics.

Myle Ott, Sergey Edunov, Alexei Baevski, Angela Fan, Sam Gross, Nathan Ng, David Grangier, and Michael Auli. 2019. fairseq: A fast, extensible toolkit for sequence modeling. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics (Demonstrations)*, pages 48–53, Minneapolis, Minnesota. Association for Computational Linguistics.

Tatsuro Oya, Yashar Mehdad, Giuseppe Carenini, and Raymond Ng. 2014. A template-based abstractive meeting summarization: Leveraging summary and source text relationships. In *Proceedings of the 8th International Natural Language Generation Conference (INLG)*, pages 45–53, Philadelphia, Pennsylvania, U.S.A. Association for Computational Linguistics.

MengNan Qi, Hao Liu, YuZhuo Fu, and Ting Liu. 2021. Improving abstractive dialogue summarization with hierarchical pretraining and topic segment. In *Findings of the Association for Computational Linguistics: EMNLP 2021*, pages 1121–1130, Punta Cana, Dominican Republic. Association for Computational Linguistics.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, Peter J Liu, et al. 2020. Exploring the limits of transfer learning with a unified text-to-text transformer. *J. Mach. Learn. Res.*, 21(140):1–67.

Pranav Rajpurkar, Jian Zhang, Konstantin Lopyrev, and Percy Liang. 2016. SQuAD: 100,000+ questions for machine comprehension of text. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 2383–2392, Austin, Texas. Association for Computational Linguistics.

Laura Sevilla-Lara, Yiyi Liao, Fatma Güney, Varun Jampani, Andreas Geiger, and Michael J Black. 2018. On the integration of optical flow and action recognition. In *German conference on pattern recognition*, pages 281–297. Springer.

Guokan Shang, Wensi Ding, Zekun Zhang, Antoine Tixier, Polykarpos Meladianos, Michalis Vazirgiannis, and Jean-Pierre Lorré. 2018. Unsupervised abstractive meeting summarization with multi-sentence compression and budgeted submodular maximization. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 664–674, Melbourne, Australia. Association for Computational Linguistics.

Karen Simonyan and Andrew Zisserman. 2014. Two-stream convolutional networks for action recognition in videos. *CoRR*, abs/1406.2199.

Karan Singla, Evgeny Stepanov, Ali Orkan Bayer, Giuseppe Carenini, and Giuseppe Riccardi. 2017. Automatic community creation for abstractive spoken conversations summarization. In *Proceedings of the Workshop on New Frontiers in Summarization*, pages 43–47, Copenhagen, Denmark. Association for Computational Linguistics.

Hao Tan and Mohit Bansal. 2019. LXMERT: Learning cross-modality encoder representations from transformers. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 5100–5111, Hong Kong, China. Association for Computational Linguistics.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Ł ukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc.

Meiqi Wang, Liulu He, Jun Lin, and Zhongfeng Wang. 2022. Rethinking adaptive computing: Building a unified model complexity-reduction framework with adversarial robustness. *IEEE Transactions on Neural Networks and Learning Systems*, 33(4):1803–1810.

Ronald J. Williams and David Zipser. 1989. A Learning Algorithm for Continually Running Fully Recurrent Neural Networks. *Neural Computation*, 1(2):270–280.

Chien-Sheng Wu, Andrea Madotto, Wenhao Liu, Pascale Fung, and Caiming Xiong. 2021. Qaconv: Question answering on informative conversations. *arXiv preprint arXiv:2105.06912*.

Dongling Xiao, Yu-Kun Li, Han Zhang, Yu Sun, Hao Tian, Hua Wu, and Haifeng Wang. 2021. ERNIE-gram: Pre-training with explicitly n-gram masked language modeling for natural language understanding. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 1702–1715, Online. Association for Computational Linguistics.

Keli Xie, Siyuan Lu, Meiqi Wang, and Zhongfeng Wang. 2021. Elbert: Fast albert with confidence-window based early exit. In *ICASSP 2021 - 2021 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 7713–7717.

Yumo Xu and Mirella Lapata. 2021. Generating query focused summaries from query-free resources. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 6096–6109, Online. Association for Computational Linguistics.

Zhilin Yang, Zihang Dai, Yiming Yang, Jaime Carbonell, Russ R Salakhutdinov, and Quoc V Le. 2019. Xlnet: Generalized autoregressive pretraining for language understanding. In *Advances in Neural Information Processing Systems*, volume 32. Curran Associates, Inc.

Fei Yu, Jiji Tang, Weichong Yin, Yu Sun, Hao Tian, Hua Wu, and Haifeng Wang. 2021. Ernie-vil: Knowledge enhanced vision-language representations through scene graphs. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 35, pages 3208–3216.

Lin Yuan and Zhou Yu. 2019. Abstractive dialog summarization with semantic scaffolds. *arXiv preprint arXiv:1910.00825*.

Weizhe Yuan, Graham Neubig, and Pengfei Liu. 2021. Bartscore: Evaluating generated text as text generation. In *Advances in Neural Information Processing Systems*, volume 34, pages 27263–27277. Curran Associates, Inc.

Jingqing Zhang, Yao Zhao, Mohammad Saleh, and Peter Liu. 2020a. PEGASUS: Pre-training with extracted gap-sentences for abstractive summarization. In *Proceedings of the 37th International Conference on Machine Learning*, volume 119 of *Proceedings of Machine Learning Research*, pages 11328–11339. PMLR.

Tianyi Zhang, Varsha Kishore, Felix Wu, Kilian Q Weinberger, and Yoav Artzi. 2020b. Bertscore: Evaluating text generation with bert. In *International Conference on Learning Representations*.

Xuchao Zhang, Bo Zong, Wei Cheng, Jingchao Ni, Yanchi Liu, and Haifeng Chen. 2021. Unsupervised concept representation learning for length-varying text similarity. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 5611–5620, Online. Association for Computational Linguistics.

Yusen Zhang, Ansong Ni, Ziming Mao, Chen Henry Wu, Chenguang Zhu, Budhaditya Deb, Ahmed Awadallah, Dragomir Radev, and Rui Zhang. 2022. Summ^n: A multi-stage summarization framework for long input dialogues and documents. In *Proceedings of the 60th Annual Meeting of the Association for Computational*

Linguistics (Volume 1: Long Papers), pages 1592–1604, Dublin, Ireland. Association for Computational Linguistics.

Lulu Zhao, Weiran Xu, and Jun Guo. 2020. Improving abstractive dialogue summarization with graph structures and topic words. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 437–449, Barcelona, Spain (Online). International Committee on Computational Linguistics.

Lulu Zhao, Weiran Xu, Chunyun Zhang, and Jun Guo. 2022. Leveraging speaker-aware structure and factual knowledge for faithful dialogue summarization. *Knowledge-Based Systems*, 245:108550.

Ming Zhong, Yang Liu, Yichong Xu, Chenguang Zhu, and Michael Zeng. 2021a. Dialoglm: Pre-trained model for long dialogue understanding and summarization. *arXiv preprint arXiv:2109.02492*.

Ming Zhong, Da Yin, Tao Yu, Ahmad Zaidi, Mutethia Mutuma, Rahul Jha, Ahmed Hassan Awadallah, Asli Celikyilmaz, Yang Liu, Xipeng Qiu, and Dragomir Radev. 2021b. QMSum: A new benchmark for query-based multi-domain meeting summarization. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 5905–5921, Online. Association for Computational Linguistics.

Chenguang Zhu, Yang Liu, Jie Mei, and Michael Zeng. 2021. MediaSum: A large-scale media interview dataset for dialogue summarization. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 5927–5934, Online. Association for Computational Linguistics.

Chenguang Zhu, Ruochen Xu, Michael Zeng, and Xuedong Huang. 2020. A hierarchical network for abstractive meeting summarization with cross-domain pretraining. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 194–203, Online. Association for Computational Linguistics.

Han Zou, Jianfei Yang, and Xiaojian Wu. 2021. Unsupervised energy-based adversarial domain adaptation for cross-domain text classification. In *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*, pages 1208–1218, Online. Association for Computational Linguistics.

Dataset	LR	MT	GStep	WStep
QMSum	3e-05	2048	4	100
SummScreen	7e-05	1024	16	200
DialogSum	3e-05	1024	2	200
QAConv	3e-05	2048	4	200
Molweni	3e-05	2048	4	200

Table 8: Main hyperparameters used for training in our experiments. LR denotes learning rate. MT indicates max tokens. GStep represents gradient accumulation step. WStep stands for warmup step.

Dataset	BeamSize	LenP	MinLen	MaxLen
QMSum	4	1.0	55	140
SummScreen	10	2.0	256	450
DialogSum	5	0.5	1	100
QAConv	3	0.1	1	20
Molweni	3	0.1	1	20

Table 9: Main settings used for generation in our experiments. LenP denotes length penalty.

A Model Settings

We list the hyperparameters used for training BART and ATM in our experiments in Table 8. The settings of generation are shown in Table 9.

B Evaluation Metrics

For BERTScore (Zhang et al., 2020b)[5], FactCC (Kryscinski et al., 2020)[6], and SUMMAC (Laban et al., 2022)[7], we all use the official implementations to evaluate our models. For BARTScore (Yuan et al., 2021)[8], we employ BART finetuned on CNN/DM (Hermann et al., 2015) to evaluate our models.

C Case Study

As shown in Table 10, we sample several cases of the generated summaries to illustrate the advantages of ATM. The comparison shows that ATM generates more coherent summaries than BART and mitigates unfaithfulness problems, such as coreference error and missing information.

[5] https://github.com/Tiiiger/bert_score
[6] https://github.com/salesforce/factCC
[7] https://github.com/tingofurro/summac
[8] https://github.com/neulab/BARTScore

Dialogue	Person1: Sally,here is a letter for us. It's from Tom. Person2: Can you read it, please? My hands are wet with all this washing. Person1: Well, OK. Dear Sally and John. Thanks for your letter. It was good to hear from you. Just a short note in reply. Please do call me when you arrive so that I can pick you up at the station.
BART	Person1 gives Sally and John a letter from Tom who will be in town in January.
ATM	Tom writes a letter to Sally and John and tells them he will pick them up when they arrive in town in January.
Dialogue	Person1: Good afternoon, what can I do for you? Person2: Yes, please. I would like to know something about the driving courses. Person1: Well, We have short full time courses during the summer. Are you interested in them? Person2: No, I am free only at weekends. Person1: Then there are weekend courses. The course starts at 8:00 ... Person2: Sounds fine. What about the coaches? Person1: We have very excellent coaches here ... Person2: Good. How many hours of training should I have each day? Person1: 3 hours in the morning and 2 in the afternoon. (...)
BART	Person2 wants to know something about the driving courses. Person1 introduces the short full-time courses, weekend courses, and the coaches.
ATM	Person2 wants to know something about the driving courses. Person1 introduces the short full-time courses and the weekend courses and introduces the coaches and the training time.
Dialogue	(...) Person2: This next one is from Betty. Person1: A highchair and car seat! Wow Betty, thank you so much! I really appreciate it! Person2: One more from Carla. Person1:A playpen and crib! Thanks Carla! This is just what I needed! (...) Person1: Umm. I think my water just broke! Get me to a hospital!
BART	Person1 thanks Person2 for organizing a great baby shower. Person1 Person 1 opens some presents from Betty, Carla, Betty, and Carla and feels like having a baby.
ATM	Person1 thanks Person2 for organizing the baby shower and opening some presents, including a bib, a stroller, a highchair, and a car seat from Betty and Carla. Person1's water breaks and asks Person2 to take her to a hospital.

Table 10: ATM and BART outputs for DialogSum dataset. The outputs reflect three main problems based on the corresponding dialogue: (1) coreference error; (2) missing information; (3) redundancy. ATM generates more coherent summaries which avoid these problems effectively and achieves better performance than BART.

Denoising Large-Scale Image Captioning from Alt-text Data using Content Selection Models

Khyathi Raghavi Chandu[♦ 1], **Piyush Sharma**[♣ 2], **Soravit Changpinyo**[♠], **Ashish Thapliyal**[♠], **Radu Soricut**[♠]

♦ Meta AI, ♣ Uber, ♠ Google Research

khyathi.research@gmail.com, {schangpi, asht, rsoricut}@google.com

Abstract

Training large-scale image captioning (IC) models demands access to a rich and diverse set of training examples that are expensive to curate both in terms of time and man-power. Instead, using alt-text based captions gathered from the web is a far cheaper alternative for scaling with the downside being that the data is noisy. Recent modeling approaches to IC often fall short in terms of performance in leveraging these noisy datasets as compared to datasets with clean annotations. We address this problem with a simple yet effective technique of breaking down the task into two smaller, more controllable tasks – skeleton prediction and skeleton-based caption generation. Specifically, we show that *sub-selecting content words as skeletons* helps in generating improved and denoised captions when leveraging rich yet noisy alt-text–based *uncurated* datasets. We also show that the predicted English skeletons can further cross-lingually be leveraged to generate non-English captions, by presenting experimental results in French, Italian, German, Spanish and Hindi. We also show that skeleton-based prediction allows for better control of caption properties, such as length, content, and gender expression, providing a handle to perform human-in-the-loop interpretable semi-automatic corrections.

1 Introduction

In the last demi-decade, NLP fields have ventured into reaping the benefits of utilizing large scale raw *(uncurated)* data from web-crawls. This trend aligned with new uncurated image-captioning datasets like Conceptual Captions (Sharma et al., 2018). While these uncurated datasets are superior in terms of size and diversity, they are inferior to well curated datasets (Lin et al., 2014; Wang et al., 2019b) in terms of noise in the captions. The content in the alt-text for the image is often distorted by the intent or context in which the image is presented. For example, the ground truth alt-text caption for a house is *'house for sale'* instead of *'front view of a house'*. This hinders the use of these large noisy datasets to the fullest extent.

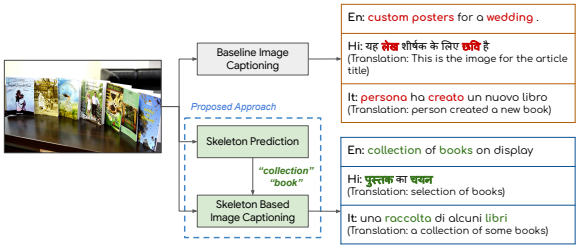

Figure 1: Overview of our approach: (1) skeleton prediction & (2) skeleton based IC; compared to conventional IC. Output captions shown in English (En), Hindi (Hi) and Italian (It).

We present a simple two-staged approach by separating the content selection from caption generation as illustrated in Figure 1. In contrast to most IC approaches (Hossain et al., 2018; Sharma et al., 2020), which hallucinate incorrect content from noisy training data (i.e 'custom posters' and 'wedding'), our approach first focuses on *denoising* the content words (i.e 'collection' and 'book') that are further used to generate a relevant caption. We refer to this sequence of concept words that are key pieces of information consistent with the image as a *skeleton*. Sub-selecting skeleton words that curb noisiness are automatically extracted from the alt-text captions. We focus on language-based skeletons that are derived from captions (Kuznetsova et al., 2014; Fang et al., 2015; Dai et al., 2018), rather than expensive visual-based skeletons derived from image, e.g., scene graphs, (Wang et al., 2019a; Yang et al., 2019), which are hard to scale. More concretely, we introduce an intermediate task of distantly supervised skeleton prediction in the end to end IC pipeline: The end-to-end task of IC ($f_\theta : \mathbb{I} \rightarrow \mathbb{C}$) is broken down into a two-staged pipeline: skeleton prediction ($f_\theta : \mathbb{I} \rightarrow \mathbb{S}$) and

1 Work done during internship at Google Research
2 Work done while employed at Google Research

Proceedings of the 29th International Conference on Computational Linguistics, pages 6089–6104
October 12–17, 2022.

skeleton based captioning ($f_\phi : \mathbb{I}, \mathbb{S} \rightarrow \mathbb{C}$), where $\mathbb{I}$ is the image, $\mathbb{S}$ is the skeleton, and $\mathbb{C}$ is the caption (Kulkarni et al., 2013; Li et al., 2011; Elliott and Keller, 2013; Fang et al., 2015). We present a comparison between encoding, decoding and autoencoding these skeletons. As such, our skeleton prediction solution addresses the *semantic gap* problem (Li and Chen, 2018; Yao et al., 2018).

We illustrate the effectiveness of this approach on uncurated noisy datasets in the following ways. (1) We demonstrate that sub-selecting content words with an intermediate *skeleton prediction task denoises content* thereby leading to better human evaluation results on captioning. We also conduct an extensive analysis on multimodal discourse relations and find that the reason for this improvement is the generation of more visible captions (Alikhani et al., 2020). (2) Scaling large uncurated datasets to other languages is still a bottleneck. We show the *transferability of learning English skeletons* to improve caption generation in other languages – English, French, Italian, German, Spanish and Hindi. (3) The predicted skeletons qualitatively demonstrate other potential benefits, such as *controllability* of content, length, and gender via a natural language–based *interpretable* interface, which enables one to additionally interact with the generation process.

2 Related Work

Content selection from vision: There is a rich body of work in improving content selection for IC (Feng et al., 2019), mainly focused on scene graph based skeletons (Gu et al., 2019; Kim et al., 2019; Chen et al., 2020a; Yang et al., 2019). However, these annotations with objects and relations are expensive, thereby constraining the scaling up to multiple languages and diverse concepts. Our work delegates this responsibility of identifying content to the language modality by using inexpensive off the shelf tools for weak supervision.

Content selection from language: An orthogonal body of work relies on skeletons derived from language using hierarchical phrase modeling (Tan and Chan, 2016; Dai et al., 2018), semantic attention (You et al., 2016), attribute LSTM (Yao et al., 2017), skeleton based attribute filling (Wang et al., 2017), adaptively merging topic and visual information (Liu et al., 2018), multimodal flow (Li et al., 2019a) and concept guided attention (Li et al., 2019b). Note that all these prior works utilize human curated gold datasets such as COCO (Lin et al., 2014) and Flickr30k (Plummer et al., 2015) with clean coupling between captions and images. However, scaling them to large and diverse concepts is expensive. We utilize *uncurated* silver standard datasets with the advantages of richness and diversity at the cost of noisy text. Hence we show the effectiveness of a dual staged approach that denoises the captions by skeleton prediction.

Cross-lingual and controllable captions: Past work on cross-lingual captioning focused on translation (Barrault et al., 2018), fluency guidance (Lan et al., 2017), using large datasets (Yoshikawa et al., 2017) and more recently by pivoting on source language captions (Thapliyal and Soricut, 2020; Gu et al., 2018). We go a step further and pivot on the predicted English skeleton to improve multilingual captions due to the dearth of similar off the shelf tools in other languages. We qualitatively explore controlling length via skeletons which was explored before via adding length to decoder (Luo and Shakhnarovich, 2020; Cornia et al., 2019). Other controllable aspects include stylistic captions (Guo et al., 2019; Mathews et al., 2018) language (Tsutsui and Crandall, 2017) which are potential extensions to our work.

Interpretable Natural language skeletons: Despite remarkable advancements of large scale end-to-end models, recent work identifies spurious correlations in datasets that potentially lead to high performance (Geva et al., 2019; Tsuchiya, 2018). To mitigate this, researchers began to dissect intermediate components of models with the goal of interpretability to humans (Wiegreffe and Pinter, 2019; Thorne et al., 2019; Lipton, 2018) as opposed to implicit explanation (Xu et al., 2015). Our work is an instance of explaining captions through skeleton predictions similar to recent work on rationalizing answer predictions for question answering (Latcinnik and Berant, 2020). We view the intermediate skeleton layer as an interpretable model prediction that helps us study key subtle dataset attributes, such as gender bias.

3 Our Approach

IC requires paired examples of images and captions ($\mathbb{I}$, $\mathbb{C}$), where $c \in \mathbb{C}$ correspond to tokens in a caption ($c_1, c_2, ..., c_m$), which are often expensive to gather. Under this paradigm, end-to-end model training attempts to perform a match between the

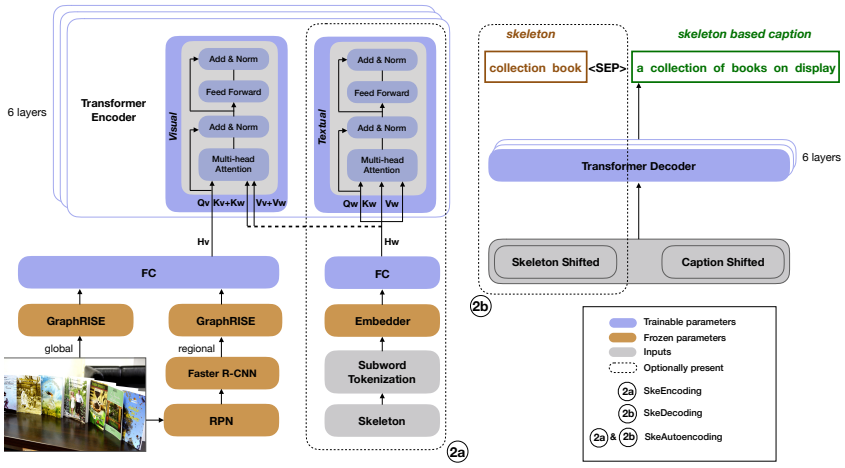

Figure 2: Model architecture of our skeleton based captioning along with *text as side attention* mechanism between visual (v) and textual (w) modalities. The skeleton is present optionally in the encoder, decoder or both based on our three approaches.

semantic concepts present in $\mathbb{I}$ and $\mathbb{C}$, starting from image, region, and object level features and mapping them to various c_is. In contrast, our approach uses intermediate skeletons as an effective way to leverage noisy, uncurated alt-text based captions to train a model to generate more visually informative captions. An overview of both the stages is presented in Fig. 1.

3.1 Distantly Supervised Skeletons

Since gold standard skeleton words are usually not available, we use distant supervision to get these labels. We retrieve syntax annotations (POS tags and word lemmas), using the Google Cloud Natural Language API [1] of caption texts. We use these annotations to experiment with skeleton variants. The ground-truth skeletons are selected by analyzing the syntax of the automatically curated web-scaled captions through combinations of nouns, verbs, adjectives and adverbs in their condensed forms. In addition, we also ignore tokens with a frequency of less than 50 in our training data to reduce noise while selecting the skeleton words. This subselection of content based on POS tags and downscaling of vocabulary helps in retaining important words as skeletons resulting in a label size of 5k. Since automatic n-gram based metrics cannot be evaluated against noisy ground-truths, manual evaluation is conducted to understand the denoising of sub-selection.

1. Nouns & Verbs: This includes a sequence of lemmas of all the nouns and verbs in a caption.

2. Salient Nouns & Verbs: Saliency of nouns and verbs is determined using tf-idf scores, treating each caption as a document. For each caption, the top 2 highest scoring noun and verb tokens (lemma) are selected. This examines if saliency contributes towards effectiveness of the skeleton.

3. Nouns: This includes lemmas of all the nouns. This helps us untangle the roles of nouns vs verbs in the effectiveness of the skeleton.

4. Iteratively refined captions: Under this condition, the output of the baseline Img2Cap model serves as the 'skeleton' for the next skeleton-based captioning stage. The rationale behind this skeleton is to compare the utility of sub-selecting skeleton words based on POS in denoising caption content, compared to a full caption prediction.

3.2 Model

Baseline (Img2Cap): We adopt an encoder-decoder ($f_\theta : \mathbb{I} \rightarrow \mathbb{C}$) IC model based on Transformers (Vaswani et al., 2017) following recent state-of-the-art approaches (Sharma et al., 2018; Yu et al., 2019; Changpinyo et al., 2019; Huang et al., 2019; Cornia et al., 2020). Our model uses the IC framework introduced in (Changpinyo et al., 2019). Inspired by the bottom-up and top-down approach (Anderson et al., 2018), the input image $\mathbb{I}$ is represented as a bag of features, containing one global and 16 regional, fine-grained feature vectors. The regional features correspond to the top 16 box proposals from a Faster-RCNN (Ren et al., 2015) object detector trained on Visual Genome (Krishna et al., 2017), with a ResNet101 (He et al., 2016) that is trained on JFT (Hinton et al., 2015) and fine-tuned on ImageNet (Russakovsky et al., 2015). We featurize both global and regional boxes using Graph-RISE (Juan et al., 2019, 2020). We make

[1] https://cloud.google.com/natural-language

the following changes to the state of the art model (Changpinyo et al., 2019), leading to a 9-point improvement on the dev CIDEr on CC (1.00 vs. 0.91) (**improved baseline**): 1) encode the corners and the area of the bounding boxes to fuse positional information with visual features, (Lu et al., 2019a), and 2) encode each feature vector with a Linear-ReLU-LayerNorm-Linear instead of Linear embedding layer, where LayerNorm is layer normalization (Ba et al., 2016).

Dual Staged Modeling: In this approach, we introduce an intermediate natural-language interpretable skeleton $\mathbb{S}$ between $\mathbb{I}$ and $\mathbb{C}$. This $\mathbb{S}$ is composed of a sequence of lemmas, using a subset of content words $(s_1, s_2, ...s_n)$ from c, where $n < m$. This reduces the output complexity of $f_\theta : \mathbb{I} \rightarrow \mathbb{C}$ by simplifying and denoising the noisy $\mathbb{C}$ to $\mathbb{S}$. Hence, the task of IC is decomposed into the first stage of predicting skeleton concepts and the second stage of caption generation using the intermediate skeleton.

Stage 1: Skeleton Prediction (Img2Ske): The first stage ($f_\theta : \mathbb{I} \rightarrow \mathbb{S}$) is to predict one of the 4 variants of the skeleton words (from §3.1) from the images. We experiment with both classification and generation paradigm that respectively do not possess and possess linear conditioning of the predicted skeleton word on the following words. We observe that the generation based skeleton prediction results in skeleton words that co-occur in a sentence. In contrast, the classification approach predicts skeleton words relevant to an image like *person, man, singer* that do not necessarily co-occur in a caption. This is detailed in §D of Appendix.

To improve co-occurrence of the predicted skeleton words, we generate the skeleton words $\hat{\mathbb{S}}$ autoregressively where each word is conditioned on the previously predicted skeleton word. This conditional dependence models word co-occurrence more tightly as $p(\hat{\mathbb{s}}_j|I, \hat{\mathbb{s}}_{<j})$, making the skeleton a sequence of words. The model is optimized with cross-entropy loss, trained using teacher forcing. An attractive property is that the same architecture can be used to decode both the skeleton $\mathbb{S}$ and the caption $\mathbb{C}$. Moreover, the output tokens predicted in this stage are interpretable, and they are used to condition the second stage of our model.

Stage 2: Skeleton-based Caption Generation: The second stage of training uses images and skeletons to generate captions $f_\phi : \mathbb{I}, \mathbb{S} \rightarrow \mathbb{C}$. We experiment with 3 variants of conditioning predicted skeletons via encoding, decoding and autoencoding as shown in the model architecture in Fig. 2. The inputs, outputs and decoder attention conditioning for each stage are compared in Table 1.

	Stage 1		Stage 2		Conditioning
	Input	Output	Input	Output	
SkeEnc	$\mathbb{I}$	$\mathbb{S}'$	$\mathbb{I}$+$\mathbb{S}'$	$\mathbb{C}'$	$\hat{\mathbb{c}}^\tau \sim \prod_t Pr(\hat{\mathbb{c}}^t \vert \hat{\mathbb{c}}^{<t}, g(z_{\mathbb{I}}, \hat{\mathbb{S}}))$
SkeAE	$\mathbb{I}$	$\mathbb{S}'$	$\mathbb{I}+\mathbb{S}'$	$\mathbb{S}'+\mathbb{C}'$	$\hat{\mathbb{c}}^\tau \sim \prod_t Pr(\hat{\mathbb{c}}^t_k \vert [\hat{\mathbb{S}}; \hat{\mathbb{c}}^{<t}], g(z_{\mathbb{I}}; \hat{\mathbb{S}}))$
SkeDec	(no Stage 1)		$\mathbb{I}$	$\mathbb{S}'+\mathbb{C}'$	$\hat{\mathbb{c}}^\tau \sim \prod_t Pr(\hat{\mathbb{c}}^t_k \vert [\hat{\mathbb{S}}; \hat{\mathbb{c}}^{<t}], z_{\mathbb{I}})$

Table 1: The inputs and outputs of the different models. In iterative refinement, $\mathbb{S}'$ is replaced by $\mathbb{C}'$.

2a. SkeEncoding: The predicted skeleton from the previous stage is used as input to the encoder. The image encoding and skeleton embeddings are fused with a unidirectional attention mechanism, called **text-as-side** (notated as g). In other words, we use the text representation as "side information" — each transformed image feature unit can attend to other image feature units (self-attention) and text (cross-attention), but text cannot attend to image. As shown in Fig. 2, this model has the dotted box in the Transformer encoder side, with the textual query, key, value (Q_w, K_w, V_w) and the visual counterpart attending to textual or visual key and value ($K_v + K_w$, $V_v + V_w$) with a visual query (Q_v). We focus on the text-as-side attention mechanism as our preliminary results indicate that it leads to qualitatively better captions than image-text co-attention (Lu et al., 2019b).

2b. SkeDecoding: The skeleton and caption are concatenated and predicted by the same decoder. This is not a two-staged model, as the model is trained to predict both skeleton and caption autoregressively. The model first predicts the skeleton words conditioned on the previously generated skeleton words, and then every token in the decoded caption attends to the entire predicted skeleton as well as the tokens of the caption decoded until that time step. The dotted box in Transformer decoder of Fig. 2 depicts this approach.

2c. SkeAE: To bring both the above models together, we simultaneously encode and decode the predicted skeleton. This brings the benefits of bidirectional attention on the input features (image and predicted skeleton words) and autoregressive attention on the re-predicted skeleton words while generating the caption. In this case, both the dotted boxes on encoder and decoder sides in Fig. 2 are active. The encoding mechanism follows the

g function and the decoder prepends the caption generation task with the predicted skeleton.

4 Experiments and Results

Hyperparameters: Our transformer model uses 6 encoder and 6 decoder layers (unless specified otherwise), with 8 heads for multiheaded attention. Captions are subword-tokenized with a vocab size of 8,300. The models are optimized with Adam and an initial learning rate of $3.2e^{-5}$. We use mini-batches of size 128, and train for 1M steps. The token embedding and filter sizes are both 512. We experimented with several values for both frequency thresholding for skeleton words at 20, 50, 100 and k at 2, 4, 8, 16 for top-k selection for multilabel classification model. We manually selected the values that optimize the model performance based on manual examination as conducting human evaluations with more hyperparameters is very expensive especially with unreliable automatic metrics.

4.1 Datasets

Conceptual Captions (CC): CC (Sharma et al., 2018) is a large-scale dataset of 3.3M image-caption pairs covering a large variety of processed alt-texts from the web. The focus of this work is on denoising noisy captioning datasets (web-scale, not human verified). Hence our experiments are focused on CC, which is a step closer to having large and diverse alt-texts from the web at the cost of being noisy. In contrast, other popular datasets like COCO (size 120K) (Lin et al., 2014) and Multi30k (Elliott et al., 2016) are hand-annotated by humans and contain high quality images/captions. As a resource, CC is useful both for measuring progress on large-scale automatic captioning (Sharma et al., 2018; Changpinyo et al., 2019; Alikhani et al., 2020; Thapliyal and Soricut, 2020), as well as pre-training data for a variety of vision-and-language tasks (Lu et al., 2019b; Chen et al., 2020c; Tan and Bansal, 2019; Su et al., 2020; Li et al., 2020).

Pre-processing: CC might contain a long tail of spelling errors and other typos due to the automatic curation of the data. Therefore, we perform frequency based thresholding of the skeleton words to abate this noise. We experimented with several values for this hyperparameter and selected a minimum occurrence count as 50 that provides the desired balance between noise and vocabulary size.

	Iterative Refinement	Classification	Generation
Precision	35.75	23.22	36.66
Recall	24.29	41.31	24.30
F-score	28.92	29.73	29.23

Table 2: Performance of skeleton prediction stage. Note that for classification and generation, the skeleton type used is 'nouns & verbs'.

Multilingual CC: To demonstrate the cross lingual transferability of our skeletons, we use automatic caption translations[2] for CC, similar to the approach in (Thapliyal and Soricut, 2020). Note that the skeletons are learned from, and predicted in, English (not in the final target language), making English skeleton act as an *interlingua*. Since multilingual captions are all pivoted on English skeletons, this nullifies the requirement to 1) collect large-scale image-caption pairs in various language, and 2) have access to linguistic tools to analyze captions in each language. We perform experiments on 5 languages – French, Italian, German, Spanish and Hindi – which vary in word orders and token overlap with the English skeletons.

Conceptual Captions T2 test set: For human evaluations across *all languages*, we use T2 test set used in the Conceptual Captions Challenge[3]. It comprises of 1K out of domain images from the Open Images Dataset (Kuznetsova et al., 2020).

4.2 Automatic Evaluation

Skeleton Prediction: The goal of this stage is to extract key skeleton words from the image. We compute precision, recall and F-score as shown in Table 2. With the same labels (skeleton: nouns & verbs), both classification and generation approaches have similar F-scores. However, precision is higher for generation and recall is higher for classification based predictions. Based on both qualitative observations and human judgements, we note that generation approach was better, which shows that a higher precision is favorable in comparison to recall for this stage. The label size (of skeletons) in Table 2 is approximately 5K.

Skeleton-based Caption Generation: We report multilingual IC performance of baseline and our dual-stage models using CIDEr in Table 3 (English) and Table 4 (multilingual). Automatic metrics for captioning are based on surface n-grams, and are not suitable to evaluate when the ground truth cap-

[2]We use the Google Cloud Translate API.

[3]http://www.conceptualcaptions.com/

Model	CIDEr		
Baseline (SOTA model)	0.91 (Changpinyo et al., 2019)		
Impr. Img2Cap	1.00		
Impr. Img2Cap (large)	0.99		
Skeleton-based	**Skeleton Type**		
	Nouns & Verbs	**Nouns only**	**Sal. Nouns & Verbs**
SkeEncoding	0.99	0.97	0.94
SkeDecoding	0.99	0.99	0.96
SkeAE	0.99	0.96	0.94

Table 3: Automatic metrics to compare various skeleton forms. Img2Cap is the baseline (*large* version refers to 12 encoder and decoder layers). Note that these results use generation-based skeleton prediction.

tions themselves are noisy. In addition, we find that CIDEr is misleading (Alikhani et al., 2020; Sharma et al., 2018; Seo et al., 2020) and does not correlate with human evaluations (§4.3). All the 4 proposed skeleton variants are evaluated systematically for automatic metrics, as shown in the last column of Table 3. However, since the automatic scores are compared against a gold standard of noisy captions, they are not reliable. Hence we conducted manual evaluation to select the best performing skeleton variant. Out of the 4 skeleton variants, nouns and verbs performed better in denoising and hence we demonstrated results for this variant for the remainder of the paper. We conducted further experimentation on nouns and verbs on the three models of dual staged captioning, controllability and cross-lingual transferability.

Multilingual captioning: Note that the skeletons are always in English, trained using annotations over the original English CC dataset. Cross-lingual results on val data of Multilingual CC are presented in Table 4. In addition to the data noisiness, a reason for slightly lower performance for non-English captions is probably noisy translation artifacts. For example, corresponding caption in the Hindi dataset for English caption 'She is gazing at the *fall colors*' is 'वह गिरते रंगों की ओर देख रही है' (translation: She is looking at the *falling colors*.) Translation errors (such as 'fall' colors to 'falling' colors) introduce noise in the non-English datasets. Figure 3 presents an example of output multilingual captions for the baseline and our SkeAE approach.

Unpaired Image Captioning: A natural extension to our approach is for the caption generator to rely purely on predicted skeleton, and not use image features. This is a harder problem, but eliminates altogether, the need for image-caption pairs because the second stage (skeleton to caption) can be trained on a large text-only corpus. In this direction, within the scope of CC dataset, we investigate

Language	Baseline	SkeEncoding	SkeDecoding	SkeAE
French	0.91	0.90	0.89	0.90
Italian	0.90	0.88	0.86	0.87
German	0.74	0.72	0.72	0.73
Spanish	0.92	0.91	0.89	0.91
Hindi	0.85	0.83	0.82	0.82

Table 4: CIDEr scores for skeleton (form: Nouns & Verbs, prediction approach: generation) conditioned caption generation for multiple languages.

Model Enc Input	CIDEr
PredSke + Img (Paired)	0.99
PredSke (Unpaired)	0.91
GtSke + Img (Paired Headroom)	4.62
GtSke (Unpaired Headroom)	4.48

Table 5: Ablations on val data for unpaired captioning.

Approach	Skeleton	Wins	Losses	Gains
SkeEncoding	Nouns & Verbs	39.34	28.33	**+11.0**
SkeAE	Nouns & Verbs	39.34	32.63	+6.7
SkeDecoding	Nouns & Verbs	34.83	34.53	+0.3
SkeEncoding	Iterative Refinement	19.62	20.52	-1.1

Table 6: Human evaluation scores of different approaches and skeletons on English (vs the Img2Cap baseline).

1) with and without using image features in the second stage, 2) using ground truth skeleton (GTSke) to get an estimate of the upper bound on unpaired captioning 3) comparing the upper bound to the predicted skeleton (PredSke). These results are presented in Table 5. When image features are ignored, CIDEr drops by only 8 points when only predicted skeletons are used for caption generation compared to the baseline. This initial result shows that skeletons are a promising direction towards unpaired captioning.

4.3 Human Evaluations

Automatic metrics often have been found not to correlate well with human scores (Kilickaya et al., 2017; Alikhani et al., 2020) and do not fare well when ground truth text is noisy. So we conduct extensive human evaluations where captions for each image are evaluated both in relative preferences and absolute scale (Thapliyal and Soricut, 2020). As mentioned above, we use the T2 test set of 1000 images, each rated by 3 distinct annotators. The interface of this evaluation is displayed in Figure 4. While comparing two models side-by-side, they are randomly assigned 'A' or 'B' in the interface for each image to avoid any rater bias.

Relative Ratings: For each image we ask the raters to choose the most relevant caption. Comparing Caption A to Caption B, raters can select relative options as shown in the third column in

Image	Model	English	French	Italian	German	Spanish	Hindi
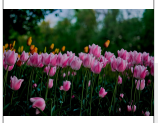	**Baseline**	spring is in the air	fleurs les plus chères du monde *(meaning: most expensive flowers in the world)*	un campo di tulipani in primavera *(meaning: a field of tulips in spring)*	Frühling ist in der Luft *(meaning: spring is in the air)*	La primavera está en el aire (meaning: spring is in the air)	वसंत हवा में है *(meaning: spring is in the air)*
	SkeAE pred skeleton: 'tulip field'	pink tulips in a field	tulipes roses dans les jardins *(meaning: pink tulips in the garden)*	genere biologico in un campo *(meaning: biological genus in a field)*	ein Feld von rosa Tulpen *(meaning: a field of pink tulips)*	tulipán en un mar de tulipanes *(meaning: tulip in a sea of tulips)*	गुलाबी ट्यूलिप का एक क्षेत्र *(meaning: a field of pink tulips)*

Figure 3: Captions generated by baseline and our dual staged approach in 6 languages and their corresponding translations.

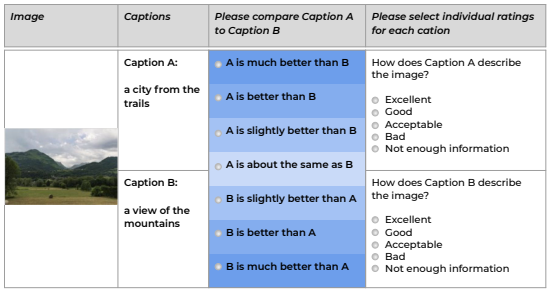

Figure 4: Human evaluation interface: We ask raters to: 1) compare the two captions (relative), 2) give ratings for each caption (absolute). Human annotators are asked to indicate the better caption relevant to the image.

Language	Wins	Losses	Gains
French	31.43	29.53	**+1.9**
Italian	26.13	24.93	**+1.2**
German	35.23	33.93	**+1.3**
Spanish	34.03	34.33	-0.3
Hindi	33.13	28.63	**+4.5**

Table 7: Human evaluation results for skeleton (form: nouns & verbs, prediction approach: generation with SKeEnc) conditioned caption generation for multiple languages.

Figure 4. *Wins* are the percentage of images where at least 2 out of 3 annotators voted for caption generated with our approach. *Losses* are percentage of images where at least 2 out of 3 annotators voted for caption generated with Img2Cap approach. We compute *gains* in this side by side relative evaluation as $Gains_{relative}$ = Wins - Losses.

Absolute Ratings: We also gather absolute rating for each of the 2 captions per image. Each caption is rated as acceptable if at least 2 out of 3 annotators rate it as *acceptable*, *good* or *excellent*. $Gains_{absolute} = Accept_{our\_approach} - Accept_{baseline}$. We use them only to validate the ratings such that, for example, an "Excellent" rated caption is not annotated as inferior to a "Bad" rated caption for the same image. These ratings are collected to double check the results of the relative rating as well.

These scores are presented in Table 8. The top part of the table indicate the absolute ratings in terms of Good and OK performance for multilingual captions. The second part of the table show the same scores when baseline model is compared with the corresponding model and skeleton combination. Each model i.e baseline and the proposed model in each row are rated individually (not relative to one another). The last two columns indicate the performance shift of the corresponding proposed model with respect to the baseline in each of the Good and OK categories.

Results: Table 6 presents the human ratings for English captions using different skeletons. From this, we observe the following:

(a) Dual Staging helps: Our dual staged models with skeletons (SkeEnc, SkeDec, SkeAE) show gains compared to the improved baseline Img2Cap model. Most notably, it shows that the 'Nouns & Verbs' skeletons significantly improves SkeEncoding model attaining the most significant gain, followed by SkeAE and then SkeDecoding.

(b) Subselecting content words helps: Using the same dual staged SkeEnc model without subselecting content words in the form of iterative refinement does not show any improvement in performance, supporting the hypothesis that sub-selecting content skeleton from noisy captions improves the overall caption quality.

(c) Cross-lingual skeleton transfer: Table 7 presents our human evaluation scores for captions in other target languages. We observe gains from the skeleton-based approach for 4 out of 5 languages, and only a slight loss for the fifth language i.e., Spanish, showing the effectiveness of cross-lingual transferability of the skeleton words. Our speculation for this is probably due to the dialect differences. The translation model that we used for Spanish is a mix of Spain Spanish and Latin American Spanish, with Latin American Spanish dominating. The evaluation was done by raters from Spain. The dialects are sufficiently different that it would impact the absolute scores.

Row no.	**Language**	**Good Baseline**	**Good SkeAE**	**OK Baseline**	**OK SkeAE**	**Gains in Good**	**Gains in OK**
1	**French**	34.63	35.04	61.36	60.66	+0.40	-0.70
2	**Italian**	35.14	35.44	60.86	62.56	+0.30	+1.70
3	**German**	43.64	41.04	67.27	68.07	-2.60	0.80
4	**Spanish**	48.15	46.55	74.37	74.67	-1.60	+0.30
5	**Hindi**	59.96	66.17	85.99	87.99	+6.21	+2.00
Row no.	**Model**	**Good Baseline**	**Good Model**	**OK Baseline**	**OK Model**	**Gains in Good**	**Gains in OK**
6	**Unpaired**	57.36	55.06	86.48	84.28	-2.30	-2.20
7	**SkeEnc (Iterative Refinement)**	63.76	62.36	87.89	87.49	-1.40	-0.40
8	**Nouns and Verbs (SkeEnc)**	66.47	63.66	89.39	88.89	+2.81	+0.50
9	**Nouns and Verbs (SkeAE)**	51.55	56.66	79.68	83.18	+ 5.01	+3.40

Table 8: Absolute ratings in percentages in Human Evaluations.

	Baseline caption	magic	peace harbour heaven	view mountain	storm darkness	house nest valley mountain
	property image # apartment for people in a picturesque village	the magic of the colours	the peace of the glorious landscape	the view from the mountains	a dark storm in the darkness	a house nestled in the valley of mountains
	a view from the water	the magic of the lakes	the peace of the river	the view from the mountains	a dark storm on the horizon	the house nestled in the valley of mountains

Figure 5: Controllability: Effect of guiding the information through skeleton. As observed, the caption incorporates information from the skeleton that is consistent with the image. For example, in the second column of the top row, we see that peace is incorporated while harbor and heaven are not. The relevant skeleton words in other columns guide the captions accordingly.

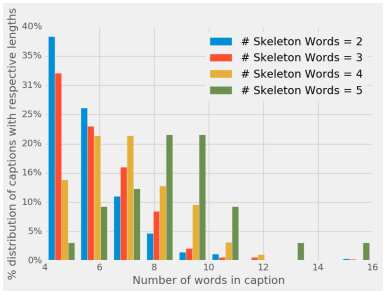

Figure 6: Quantitative relationship between the number of skeleton words and caption length.

	Counts			**Human Evals**
	Baseline	**Ours**	**Change**	
Visible	605	640	+5.79%	+10.93%
Meta	245	226	-7.76%	+13.06%
Story	129	108	-16.28%	+10.08%

Table 9: Analysis of multimodal discourse coherence relations for baseline and our model on T2 dataset. The last column shows the relative human evaluation gains over baseline caption of each type. Other relations with small counts are ignored in the above analysis.

4.4 Cross-modal Discourse Coherence

To understand where the improvements quantified in Table 6 come from, we turn to the notion of discourse coherence. Alikhani et al. (2020) introduce multimodal discourse coherence relationships between image-caption pairs. For instance, a caption describing visually recognizable aspects of the image, such as 'people' or 'cake', is annotated using a *Visible* relation; in contrast, a *Meta* relation corresponds to a caption containing details regarding how/when/where the image was captured, such as in 'warm summer afternoon', while a *Story* relation implies that the caption describes some potentially non-visible context behind the scene depicted in the image, such as 'fifth anniversary'.

We hypothesize that our multi-stage approach of skeleton-based IC results in the generation of more captions of *Visible* type, as the intermediate skeleton predictor is trained to predict nouns and verbs from the image. As observed in §4.3, as SkeAE model performs better compared to the SkeEncoding and SkeDecoding models, we analyze the downstream captions based on SkeAE architecture. To assess this effect, we train the relation classifier described in Sec. 4 of (Alikhani et al., 2020), and obtain discourse relation labels for captions generated on T2-test images, by both the baseline Img2Cap and our SkeAE models. Table 9 (Counts columns) quantifies the shift of relation label distribution towards the *Visible* coherence relation, confirming our hypothesis. We also study the breakdown by coherence relations using the results from our human evaluations on the English captions. Table 9 (Human Evals column) reports this breakdown, indicating that, of the 11.01% gains on human evals from Table 6, the shift from non-Visible to Visible discourse captions is associated with clear increases in preference from the human raters. This is attributable to the fact that human raters are more likely to prefer captions that are in a *Visible* relation with the image, and therefore the shift towards generating *Visible*-type captions can be positively quantified in terms of human preference.

Skeleton Words	valley (1 word)	valley mountain (2 words)	house valley mountain (3 words)	house nest valley mountain (4 words)
	the colours of the valley (5 words)	the green valley of mountains (5 words)	houses of the valley and mountains (6 words)	a house nestled in the valley of mountains (8 words)

Figure 7: Controllability: Effect of varying the number of words in the skeleton on the generated caption length.

5 Controllability: Qualitative Discussion

The dual-stage modeling using skeelton decomposition can be a double-edged sword: it can be an information bottleneck, limiting the ability to train the model in an end-to-end manner; but, it brings forth the advantage of increased interpretability and thereby the ability to use the intermediate stage results to control the final caption. We present aspects of caption controllability by altering the skeleton to explore effects on caption length, informativeness, and gender specificity. This section discusses the utility of this dual staged model for controllability qualitatively with SkeAE architecture. Automatic intervention at the skeleton level involves non-trivially selecting related concepts for each image, and we leave this for follow-up work. Instead, we present an empirical study only to semi-automatically control gender specificity in two of the languages. We plan to conduct experiments to compare with other models (Zheng et al., 2019; Chen et al., 2020b) focused on controllability for follow-up work.

Effect of length of skeletons on captions: For applications that limit the caption lengths due to UI restrictions, the ability to control the length is important. The length of the skeleton correlates with the number of caption words, as shown in Figure 6. For 2 or 3 skeleton words, the percentage of captions monotonically decreases with the number of caption words, with the mode at 4-word captions. Thus, for skeletons of size 2, captions of length 4 are much more frequent than captions of length 6 or 8. For longer skeletons, we see that the mode shifts to the right: with skeletons of size 5, the caption length peaks between 8 and 10 words. Fig 7 illustrates this qualitatively.

Effect on gender specificity: Current models often make embarrassing mistakes when generating captions that mention gender. The availability of a skeleton provides a direct handle for human-in-the-loop correction of such biases, at a pre-caption-generation stage. This is more robust compared to caption post-processing, especially for highly inflected languages. To illustrate this, we compare the number of times 'man' appears in the captions generated by our baseline versus our dual-stage model after automatically modifying the skeleton (replacing 'man' to the gender-neutral word 'person' in the skeleton). Over the T2 dataset, the baseline caption generates 'man' 13 times, and the automatic control mechanism via our model reduces this by 46% (to 7 occurrences) in English. In Hindi, the equivalent of 'man' (आदमी) is generated 10 times, and it is reduced to a gender neutral word (व्यक्ति) by 70% (to 3 occurrences).

Effect of guiding information through skeleton: The skeleton acts as a knob enabling the model to describe different attributes of the image in the caption. Figure 5 presents an example of how varying the skeletons for two different images affect their captions. The words highlighted in green are derived from the skeleton and the ones highlighted in blue are image-related words.

6 Conclusions

Scaling image captioning models practically mandates training on noisy and uncurated data available on web. Our work presents an approach that denoises learning from such large yet diverse web-scaled data with alt-text annotations by subselecting content as intermediate skeletons. We experimentally demonstrate that this approach improves the captions significantly in human evaluations on out-of-domain test data by converting meta and story like captions to more visually informative captions. We also demonstrate the transferability of English skeleton words to improving captions in five other languages. Additionally, the natural-language interpretable skeleton layer gives us a way to better control and perform human-in-the-loop corrections of model predictions. We believe that this is a promising direction towards unpaired IC and also has potential for semi-automatic interventions to correct or interact with the skeletons to guide the final captions.

In this work, our main focus is denoising alt-text captions using skeletons and using them for cross-lingual captioning. In future, we plan to explore the effect of denoising in pretraining large multimodal models (BLIP (Li et al., 2022), UNITER (Chen et al., 2020d), ViLBERT (Lu et al., 2019c)) as base architectures by automatically cleaning captions, similar to how BLIP has an additional classifier to subselect captions that are not noisy. *Appendix H presents a broader impact of our work.*

References

Malihe Alikhani, Piyush Sharma, Shengjie Li, Radu Soricut, and Matthew Stone. 2020. Cross-modal coherence modeling for caption generation. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 6525–6535.

Peter Anderson, Xiaodong He, Chris Buehler, Damien Teney, Mark Johnson, Stephen Gould, and Lei Zhang. 2018. Bottom-up and top-down attention for image captioning and visual question answering. In *CVPR*.

Jimmy Lei Ba, Jamie Ryan Kiros, and Geoffrey E. Hinton. 2016. Layer normalization. *arXiv preprint arXiv:1607.06450*.

Loïc Barrault, Fethi Bougares, Lucia Specia, Chiraag Lala, Desmond Elliott, and Stella Frank. 2018. Findings of the third shared task on multimodal machine translation. In *Proceedings of the Third Conference on Machine Translation: Shared Task Papers, WMT 2018, Belgium, Brussels, October 31 - November 1, 2018*, pages 304–323. Association for Computational Linguistics.

Soravit Changpinyo, Bo Pang, Piyush Sharma, and Radu Soricut. 2019. Decoupled box proposal and featurization with ultrafine-grained semantic labels improve image captioning and visual question answering. In *EMNLP-IJCNLP*.

Shizhe Chen, Qin Jin, Peng Wang, and Qi Wu. 2020a. Say as you wish: Fine-grained control of image caption generation with abstract scene graphs. *CoRR*, abs/2003.00387.

Shizhe Chen, Qin Jin, Peng Wang, and Qi Wu. 2020b. Say as you wish: Fine-grained control of image caption generation with abstract scene graphs. In *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition, CVPR 2020, Seattle, WA, USA, June 13-19, 2020*, pages 9959–9968. IEEE.

Yen-Chun Chen, Linjie Li, Licheng Yu, Ahmed El Kholy, Faisal Ahmed, Zhe Gan, Yu Cheng, and Jingjing Liu. 2020c. UNITER: Learning UNiversal Image-TExt Representations. In *ECCV*.

Yen-Chun Chen, Linjie Li, Licheng Yu, Ahmed El Kholy, Faisal Ahmed, Zhe Gan, Yu Cheng, and Jingjing Liu. 2020d. UNITER: universal image-text representation learning. In *Computer Vision - ECCV 2020 - 16th European Conference, Glasgow, UK, August 23-28, 2020, Proceedings, Part XXX*, volume 12375 of *Lecture Notes in Computer Science*, pages 104–120. Springer.

Marcella Cornia, Lorenzo Baraldi, and Rita Cucchiara. 2019. Show, control and tell: A framework for generating controllable and grounded captions. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 8307–8316.

Marcella Cornia, Matteo Stefanini, Lorenzo Baraldi, and Rita Cucchiara. 2020. Meshed-Memory Transformer for Image Captioning. In *CVPR*.

Bo Dai, Sanja Fidler, and Dahua Lin. 2018. A neural compositional paradigm for image captioning. In *Advances in Neural Information Processing Systems*, pages 658–668.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT 2019, Minneapolis, MN, USA, June 2-7, 2019, Volume 1 (Long and Short Papers)*, pages 4171–4186. Association for Computational Linguistics.

Desmond Elliott, Stella Frank, Khalil Sima'an, and Lucia Specia. 2016. Multi30k: Multilingual english-german image descriptions. In *Proceedings of the 5th Workshop on Vision and Language, hosted by the 54th Annual Meeting of the Association for Computational Linguistics, VL@ACL 2016, August 12, Berlin, Germany*. The Association for Computer Linguistics.

Desmond Elliott and Frank Keller. 2013. Image description using visual dependency representations. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*, pages 1292–1302.

Hao Fang, Saurabh Gupta, Forrest N. Iandola, Rupesh Kumar Srivastava, Li Deng, Piotr Dollár, Jianfeng Gao, Xiaodong He, Margaret Mitchell, John C. Platt, C. Lawrence Zitnick, and Geoffrey Zweig. 2015. From captions to visual concepts and back. In *IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2015, Boston, MA, USA, June 7-12, 2015*, pages 1473–1482. IEEE Computer Society.

Yang Feng, Lin Ma, Wei Liu, and Jiebo Luo. 2019. Unsupervised image captioning. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 4125–4134.

Mor Geva, Yoav Goldberg, and Jonathan Berant. 2019. Are we modeling the task or the annotator? an investigation of annotator bias in natural language understanding datasets. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing, EMNLP-IJCNLP 2019, Hong Kong, China, November 3-7, 2019*, pages 1161–1166. Association for Computational Linguistics.

Jiuxiang Gu, Shafiq R. Joty, Jianfei Cai, and Gang Wang. 2018. Unpaired image captioning by language pivoting. In *Computer Vision - ECCV 2018 - 15th European Conference, Munich, Germany, September 8-14, 2018, Proceedings, Part I*, volume 11205 of *Lecture Notes in Computer Science*, pages 519–535. Springer.

Jiuxiang Gu, Shafiq R. Joty, Jianfei Cai, Handong Zhao, Xu Yang, and Gang Wang. 2019. Unpaired image

captioning via scene graph alignments. In *2019 IEEE/CVF International Conference on Computer Vision, ICCV 2019, Seoul, Korea (South), October 27 - November 2, 2019*, pages 10322–10331. IEEE.

Longteng Guo, Jing Liu, Peng Yao, Jiangwei Li, and Hanqing Lu. 2019. Mscap: Multi-style image captioning with unpaired stylized text. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 4204–4213.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. 2016. Deep residual learning for image recognition. In *CVPR*.

Geoffrey Hinton, Oriol Vinyals, and Jeff Dean. 2015. Distilling the knowledge in a neural network. In *Proceedings of NIPS workshop*.

Md. Zakir Hossain, Ferdous Sohel, Mohd Fairuz Shiratuddin, and Hamid Laga. 2018. A comprehensive survey of deep learning for image captioning. *CoRR*, abs/1810.04020.

Lun Huang, Wenmin Wang, Jie Chen, and Xiao-Yong Wei. 2019. Attention on attention for image captioning. In *CVPR*.

Da-Cheng Juan, Chun-Ta Lu, Zhen Li, Futang Peng, Aleksei Timofeev, Yi-Ting Chen, Yaxi Gao, Tom Duerig, Andrew Tomkins, and Sujith Ravi. 2019. Graph-rise: Graph-regularized image semantic embedding. *CoRR*, abs/1902.10814.

Da-Cheng Juan, Chun-Ta Lu, Zhen Li, Futang Peng, Aleksei Timofeev, Yi-Ting Chen, Yaxi Gao, Tom Duerig, Andrew Tomkins, and Sujith Ravi. 2020. Ultra fine-grained image semantic embedding. In *Proceedings of the 13th International Conference on Web Search and Data Mining*, pages 277–285.

Mert Kilickaya, Aykut Erdem, Nazli Ikizler-Cinbis, and Erkut Erdem. 2017. Re-evaluating automatic metrics for image captioning. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics, EACL 2017, Valencia, Spain, April 3-7, 2017, Volume 1: Long Papers*, pages 199–209. Association for Computational Linguistics.

Suwon Kim, HongYong Choi, JoongWon Hwang, JangYoung Song, SangRok Lee, TaeKang Woo, and AI Modulabs. 2019. Vizwiz image captioning based on aoanet with scene graph. *ivc.ischool.utexas.edu*.

Ranjay Krishna, Yuke Zhu, Oliver Groth, Justin Johnson, Kenji Hata, Joshua Kravitz, Stephanie Chen, Yannis Kalantidis, Li-Jia Li, David A. Shamma, Michael Bernstein, and Li Fei-Fei. 2017. Visual Genome: Connecting language and vision using crowdsourced dense image annotations. *IJCV*, 123(1):32–73.

Girish Kulkarni, Visruth Premraj, Vicente Ordonez, Sagnik Dhar, Siming Li, Yejin Choi, Alexander C Berg, and Tamara L Berg. 2013. Babytalk: Understanding and generating simple image descriptions. volume 35, pages 2891–2903. IEEE.

Alina Kuznetsova, Mohamad Hassan Mohamad Rom, Neil Alldrin, Jasper Uijlings, Ivan Krasin, Jordi Pont-Tuset, Shahab Kamali, Stefan Popov, Matteo Malloci, Alexander Kolesnikov, et al. 2020. The open images dataset v4: Unified image classification, object detection, and visual relationship detection at scale.

Polina Kuznetsova, Vicente Ordonez, Tamara L Berg, and Yejin Choi. 2014. Treetalk: Composition and compression of trees for image descriptions. *Transactions of the Association for Computational Linguistics*, 2:351–362.

Weiyu Lan, Xirong Li, and Jianfeng Dong. 2017. Fluency-guided cross-lingual image captioning. In *Proceedings of the 25th ACM international conference on Multimedia*, pages 1549–1557.

Veronica Latcinnik and Jonathan Berant. 2020. Explaining question answering models through text generation. *CoRR*, abs/2004.05569.

Gen Li, Nan Duan, Yuejian Fang, Daxin Jiang, and Ming Zhou. 2020. Unicoder-VL: A universal encoder for vision and language by cross-modal pre-training. In *AAAI*.

Guang Li, Linchao Zhu, Ping Liu, and Yi Yang. 2019a. Entangled transformer for image captioning. In *2019 IEEE/CVF International Conference on Computer Vision, ICCV 2019, Seoul, Korea (South), October 27 - November 2, 2019*, pages 8927–8936. IEEE.

Jiangyun Li, Peng Yao, Longteng Guo, and Weicun Zhang. 2019b. Boosted transformer for image captioning. *Applied Sciences (2076-3417)*, 9(16).

Junnan Li, Dongxu Li, Caiming Xiong, and Steven C. H. Hoi. 2022. BLIP: bootstrapping language-image pre-training for unified vision-language understanding and generation. *CoRR*, abs/2201.12086.

Nannan Li and Zhenzhong Chen. 2018. Image cationing with visual-semantic LSTM. In *Proceedings of the Twenty-Seventh International Joint Conference on Artificial Intelligence, IJCAI 2018, July 13-19, 2018, Stockholm, Sweden*, pages 793–799. ijcai.org.

Siming Li, Girish Kulkarni, Tamara Berg, Alexander Berg, and Yejin Choi. 2011. Composing simple image descriptions using web-scale n-grams. In *Proceedings of the Fifteenth Conference on Computational Natural Language Learning*, pages 220–228.

Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C Lawrence Zitnick. 2014. Microsoft coco: Common objects in context. In *European conference on computer vision*, pages 740–755. Springer.

Zachary C. Lipton. 2018. The mythos of model interpretability. *Commun. ACM*, 61(10):36–43.

Fenglin Liu, Xuancheng Ren, Yuanxin Liu, Houfeng Wang, and Xu Sun. 2018. simnet: Stepwise image-topic merging network for generating detailed and comprehensive image captions. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 137–149.

Jiasen Lu, Dhruv Batra, Devi Parikh, and Stefan Lee. 2019a. ViLBERT: Pretraining task-agnostic visiolinguistic representations for vision-and-language tasks. In *NeurIPS*.

Jiasen Lu, Dhruv Batra, Devi Parikh, and Stefan Lee. 2019b. ViLBERT: Pretraining task-agnostic visiolinguistic representations for vision-and-language tasks. In *NeurIPS*.

Jiasen Lu, Dhruv Batra, Devi Parikh, and Stefan Lee. 2019c. Vilbert: Pretraining task-agnostic visiolinguistic representations for vision-and-language tasks. In *Advances in Neural Information Processing Systems 32: Annual Conference on Neural Information Processing Systems 2019, NeurIPS 2019, December 8-14, 2019, Vancouver, BC, Canada*, pages 13–23.

Ruotian Luo and Greg Shakhnarovich. 2020. Controlling length in image captioning. *CoRR*, abs/2005.14386.

Alexander Mathews, Lexing Xie, and Xuming He. 2018. Semstyle: Learning to generate stylised image captions using unaligned text. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 8591–8600.

Bryan A Plummer, Liwei Wang, Chris M Cervantes, Juan C Caicedo, Julia Hockenmaier, and Svetlana Lazebnik. 2015. Flickr30k entities: Collecting region-to-phrase correspondences for richer image-to-sentence models. In *Proceedings of the IEEE international conference on computer vision*, pages 2641–2649.

Shaoqing Ren, Kaiming He, Ross B. Girshick, and Jian Sun. 2015. Faster R-CNN: towards real-time object detection with region proposal networks. In *Advances in Neural Information Processing Systems 28: Annual Conference on Neural Information Processing Systems 2015, December 7-12, 2015, Montreal, Quebec, Canada*, pages 91–99.

Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael Bernstein, Alexander C. Berg, and Li Fei-Fei. 2015. ImageNet large scale visual recognition challenge. *IJCV*, 115(3):211–252.

Paul Hongsuck Seo, Piyush Sharma, Tomer Levinboim, Bohyung Han, and Radu Soricut. 2020. Reinforcing an image caption generator using off-line human feedback. In *The Thirty-Fourth AAAI Conference on Artificial Intelligence, AAAI 2020, The Thirty-Second Innovative Applications of Artificial Intelligence Conference, IAAI 2020, The Tenth AAAI Symposium on Educational Advances in Artificial Intelligence, EAAI 2020, New York, NY, USA, February 7-12, 2020*, pages 2693–2700. AAAI Press.

Himanshu Sharma, Manmohan Agrahari, Sujeet Kumar Singh, Mohd Firoj, and Ravi Kumar Mishra. 2020. Image captioning: A comprehensive survey. In *2020 International Conference on Power Electronics & IoT Applications in Renewable Energy and its Control (PARC)*, pages 325–328. IEEE.

Piyush Sharma, Nan Ding, Sebastian Goodman, and Radu Soricut. 2018. Conceptual captions: A cleaned, hypernymed, image alt-text dataset for automatic image captioning. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 2556–2565.

Weijie Su, Xizhou Zhu, Yue Cao, Bin Li, Lewei Lu, Furu Wei, and Jifeng Dai. 2020. VL-BERT: Pre-training of generic visual-linguistic representations. In *ICLR*.

Hao Tan and Mohit Bansal. 2019. LXMERT: Learning cross-modality encoder representations from transformers. In *EMNLP-IJCNLP*.

Ying Hua Tan and Chee Seng Chan. 2016. phi-lstm: A phrase-based hierarchical LSTM model for image captioning. In *Computer Vision - ACCV 2016 - 13th Asian Conference on Computer Vision, Taipei, Taiwan, November 20-24, 2016, Revised Selected Papers, Part V*, volume 10115 of *Lecture Notes in Computer Science*, pages 101–117. Springer.

Ashish V. Thapliyal and Radu Soricut. 2020. Cross-modal language generation using pivot stabilization for web-scale language coverage. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, ACL 2020, Online, July 5-10, 2020*, pages 160–170. Association for Computational Linguistics.

James Thorne, Andreas Vlachos, Christos Christodoulopoulos, and Arpit Mittal. 2019. Generating token-level explanations for natural language inference. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT 2019, Minneapolis, MN, USA, June 2-7, 2019, Volume 1 (Long and Short Papers)*, pages 963–969. Association for Computational Linguistics.

Masatoshi Tsuchiya. 2018. Performance impact caused by hidden bias of training data for recognizing textual entailment. In *Proceedings of the Eleventh International Conference on Language Resources and Evaluation, LREC 2018, Miyazaki, Japan, May 7-12, 2018*. European Language Resources Association (ELRA).

Satoshi Tsutsui and David Crandall. 2017. Using artificial tokens to control languages for multilingual image caption generation. *arXiv preprint arXiv:1706.06275*.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in neural information processing systems*, pages 5998–6008.

Dalin Wang, Daniel Beck, and Trevor Cohn. 2019a. On the role of scene graphs in image captioning. In *Proceedings of the Beyond Vision and LANguage: inTEgrating Real-world kNowledge (LANTERN)*, pages 29–34.

Tianlu Wang, Jieyu Zhao, Mark Yatskar, Kai-Wei Chang, and Vicente Ordonez. 2019b. Balanced datasets are not enough: Estimating and mitigating gender bias in deep image representations. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 5310–5319.

Yufei Wang, Zhe Lin, Xiaohui Shen, Scott Cohen, and Garrison W Cottrell. 2017. Skeleton key: Image captioning by skeleton-attribute decomposition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 7272–7281.

Sarah Wiegreffe and Yuval Pinter. 2019. Attention is not not explanation. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing, EMNLP-IJCNLP 2019, Hong Kong, China, November 3-7, 2019*, pages 11–20. Association for Computational Linguistics.

Kelvin Xu, Jimmy Ba, Ryan Kiros, Kyunghyun Cho, Aaron Courville, Ruslan Salakhudinov, Rich Zemel, and Yoshua Bengio. 2015. Show, attend and tell: Neural image caption generation with visual attention. In *International conference on machine learning*, pages 2048–2057.

Xu Yang, Kaihua Tang, Hanwang Zhang, and Jianfei Cai. 2019. Auto-encoding scene graphs for image captioning. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 10685–10694.

Ting Yao, Yingwei Pan, Yehao Li, and Tao Mei. 2018. Exploring visual relationship for image captioning. In *Proceedings of the European conference on computer vision (ECCV)*, pages 684–699.

Ting Yao, Yingwei Pan, Yehao Li, Zhaofan Qiu, and Tao Mei. 2017. Boosting image captioning with attributes. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 4894–4902.

Yuya Yoshikawa, Yutaro Shigeto, and Akikazu Takeuchi. 2017. STAIR captions: Constructing a large-scale japanese image caption dataset. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics, ACL 2017, Vancouver, Canada, July 30 - August 4, Volume 2: Short Papers*, pages 417–421. Association for Computational Linguistics.

Quanzeng You, Hailin Jin, Zhaowen Wang, Chen Fang, and Jiebo Luo. 2016. Image captioning with semantic attention. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 4651–4659.

Jun Yu, Jing Li, Zhou Yu, and Qingming Huang. 2019. Multimodal transformer with multi-view visual representation for image captioning. *arXiv*, 1905.07841.

Yue Zheng, Yali Li, and Shengjin Wang. 2019. Intention oriented image captions with guiding objects. In *IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2019, Long Beach, CA, USA, June 16-20, 2019*, pages 8395–8404. Computer Vision Foundation / IEEE.

A Comparison of SkeEnc and SkeAE on multilingual captions

We have discussed the human evaluation scores of the SkeAE model by using *nouns and verbs* as skeletons in Table 7 in the main paper. In addition to this, we also conducted human evaluation to compare the SkeEnc model with the *nouns and verbs* skeletons in comparison to the baseline. We present this in Table 10. While there are improvements in 3 languages, the performance is also hurt in two languages. However, as we see, by comparing the performances in Table 7 and Table 10, we observe that SkeAE has a clear advantage when leveraging the English caption to improve multilingual captions. This clearly indicates that channelling the prediction of the skeleton words in conjuction with the caption itself is enabling the model decoder to attend to the previously predicted skeleton words in the same decoder.

Language	Wins	Losses	Gains
French	31.93	31.43	**+0.50**
Italian	33.13	28.32	**+4.81**
German	29.43	29.72	**-0.30**
Spanish	30.53	34.43	-3.90
Hindi	29.93	26.03	**+3.90**

Table 10: Human evaluation results on SkeEnc model for skeleton (form: nouns & verbs, prediction approach: generation) conditioned caption generation for multiple languages.

B Comparison of Classification and Generation based Skeleton Prediction

From a preliminary manual analysis, we observed that the classification based approach to skeleton prediction faces the problem of predicting words that are related but are not likely to co-occur within the same sentence in the caption. This is described in detail in points 1a and 1b of §3. To validate this observation, we conducted human evaluation of the captions generated from classification and generation based approaches relative to one another. This setup is different from the rest of the experiments in human evaluation in the paper which compare any given model relative to the baseline model. In contrast, this study is to compare the generation and classification approaches with one another. These results are presented in Table 11.

The top-8 highest scoring content words are chosen to reduce input noise for the caption generator while improving the recall of concepts. We experimented with different values for this and selected 8 to be an optimal balance between the content in the skeleton words and the noise.

Approach	Wins	Losses	Gains
Generation	39.14	30.23	**+8.91**

Table 11: Human evaluation results of comparison between the generation and classification based approaches

We observe that the generation based approach has significant gains of +8.91 over the classification based approach. Most of the prior literature uses the classification based approach to predict content or bag of concepts to assist caption generation. Our hypothesis is that this classification based model helps in end-to-end approaches where the loss from caption generation backpropagates to the classifier model as well. As opposed to this, our model decouples the prediction of the skeleton or concept words that are further used for caption generation. Hence we believe that suppressing the words that do not co-occur is important in the skeleton prediction task and the generation based approach is addressing this problem.

C Absolute Ratings

Here are some of the observations from these results:

- *Better results of Dual Staged Approach:* As we can see in the last two rows (rows 8 and 9), our proposed SkeEnc and SkeAE show absolute improvements in both the categories. This further demonstrates that the proposed dual staged approach is generating better denoised captions when trained on noisy uncurated alt-text–based captions.

- *Sub-selecting content words is better:* Now that we saw the improvements with the dual staged approach, we now investigate whether sub-selecting content words is important. For this, we present comparison between rows 7 and 8. Both these models are dual staged with SkeEnc i.e encoding the predicted skeleton in the second stage. The only difference is that row 8 sub-selects all nouns and verbs to predict the skeletons whereas row 8 includes all the words from the captions to predict the skeletons. Row 8 shows better performance compared to row 7. This means that sub-selecting content words contribute to the caption generation in the second stage.

Please note that we focus on alt-text based captions, so we experiment on Conceptual Captions instead of cleaner alternatives such as MSCOCO and Multi30k. The latter do not include as noisy captions as they are hand-annotated (refer Section 4.1)

D Img2Ske: Classification based prediction

Skeleton prediction is posed as a multilabel classification problem where the prediction of a skeleton word s_i is not conditionally dependent on the prediction of another skeleton word s_j. Our goal is to evaluate the effectiveness of simple generation and classification models to predict skeletons, and naturally generation based approach reduces redundancies due to conditional dependence of label/skeleton prediction. The encoder part remains the same as the baseline followed by optimization with sigmoid cross entropy between the skeleton words $\mathbb{S}$ and image encoding $\boldsymbol{z}_{\mathbb{I}}$, which is the representation of the image from the encoder.

$$\text{Accuracy, } A = \frac{1}{N}\sum_{i=1}^{N}\frac{\left|\mathbb{S}_i \cap \hat{\mathbb{S}}_i\right|}{\left|\mathbb{S}_i \cup \hat{\mathbb{S}}_i\right|} \quad (1)$$

The skeleton for the second stage is chosen as the ordered list of top-8 (experimentally selected) high scoring words after the softmax layer. However, conditional independence of skeleton words with one another ignores the co-occurrences of words capable of composing a sentence or a final caption. For instance, classification predictions are composed of words and their synonyms that are highly correlated like {*person*, *man*, *singer*}. These words definitely are relevant to an image but do not all necessarily co-occur in a sentence.

Table 2 presents the precision, recall and f-scores of the generation and classification based approaches for skeleton prediction. These metrics, however are misleading because they do not account for synonyms or semantic similarity. For example, 'food', 'meal', 'lunch' and 'dinner' are all distinct labels while computing these metrics, and predicting one instead of the other get heavily penalized even though the effect on downstream caption quality would be minimal. This issue gets amplified by the fact that with CC that has a rich vocabulary with words such as electricity 'pylon' and 'tower' referring to the same concept.

E Performance drop for Spanish

While we have seen improvements in the performance on multiple languages in human evaluation (Table 6), we observed a drop in the preference for Spanish captions when we use skeletons. Given the similarity in word order between Spanish and English in comparison to Hindi, the lower performance of Spanish is an interesting result indeed. Our speculation for this is probably due to the dialect differences. The translation model that we used for Spanish is a mix of 'Spain Spanish' and 'Latin American Spanish', with Latin American Spanish dominating. The evaluation was done by raters from Spain. The dialects are sufficiently different that it would impact the absolute scores.

F Intuition for skeleton words:

The alt-text captions are silver standard and harbor a lot of diversity. Hence filtering frequently occurring words based on a frequency cutoff as the skeletons helps balance between conditioning on the frequent words (not noise) and diverse concepts. Qualitatively, consider an image of a house with the caption *'apartment for rent'* and *'apartment for sale'*. With the frequency based skeleton selection, the noun word *'apartment'* is selected as skeleton ignoring the rest. In this way, we are denoising alt-text captions to generate captions with visible concepts.

G Hyperparameters:

This section lists the hyperparameters used for training our models. We used BERT embeddings (Devlin et al., 2019) to initialize the words in skeletons in the SkeEnc and SkeAE models.

- *Learning rate:* We experimented with $3.2e^{-5}$, 0.5, 1, 1.5 and 2 as the learning rate. The experiments presented in the paper have the learning rate of 1. The learning rate is decayed at 0.95 decay rate with staircase strategy.

- *Number of layers:* All our models have 6 layers for encoder and decoder. We also conducted an additional experiment to check if the model complexity of the end-to-end baseline can improve the performance in comparison to our dual staged approach. To evaluate this, we doubled the number of layers where the number of transformer encoder and decoder layers are 12 each as presented in the

paper as Impr Img2Cap (large) in Table 3 in Section 4.2.

- *Subtoken Vocabulary:* We experimented with 4000 and 8300 sub-token vocabularies. The experiments in the paper all have 8,300 as subtoken vocabulary size.
- *Batch size:* All our experiments include batchsize of 128 only.
- *Number of steps:* We train for a maximum of 1 million update steps.
- *Maximum Caption Length:* In the baseline and the SkeEnc models, our decoder generates a maximum words of length 36. In the SkeAE and SkeDec model, the skeleton words are prepended to the caption. So we allow the decoder to generate 72 words in these two models.
- *Warm up and decay steps:* The model is warmed up for 20 epochs and decayed for 25 epochs.
- *Embedding size:* We use embedding dimension of 512.
- *Beam size:* We perform beam search in the decoder with a beam size of 5.

Here are some of the configuration and modeling choices for training the models:

- *Attention type:* Our experiments include attention types of cross-attention and text-aside as described along with point *2a* in Section 3.
- *FRCNN Tokens:* We use 1601 tokens from the trained FRCNN.

H Broader Impact

We believe that this work has extensive impact in scaling captioning models to large and noisy datasets thereby exploiting web data and reduce manual annotation efforts. We do not foresee any immediate concerns ethically directly from our work. However, while applying this to datasets crawled from the web, offensive content should be removed. In general, we envisage researchers and practitioners to benefit from our approach especially, when expensive human annotations are not available. More broadly speaking, we also strongly believe that our approach laid blocks for future work on cross-lingually leveraging English skeletons and automatic translations to generate captions for various languages. Hence, when combined with unpaired captioning, this can especially benefit captioning in low resource languages.

Meta-CQG: A Meta-Learning Framework for Complex Question Generation over Knowledge Bases

Kun Zhang[1,2], Yunqi Qiu[4], Yuanzhuo Wang[1,2,3]*, Long Bai[2], Wei Li[4], Xuhui Jiang[1,2], Huawei Shen[1,2], Xueqi Cheng[2]

[1]Data Intelligence System Research Center, Institute of Computing Technology, Chinese Academy of Sciences;
[2]School of Computer Science and Technology, University of Chinese Academy of Sciences;
[3]Big Data Academy, Zhongke; [4]Baidu Inc.

{zhangkun18z, wangyuanzhuo, bailong18b, jiangxuhui19g}@ict.ac.cn
{shenhuawei, cxq}@ict.ac.cn {qiuyunqi, liwei85}@baidu.com

Abstract

Complex question generation over knowledge bases (KB) aims to generate natural language questions involving multiple KB relations or functional constraints. Existing methods train one encoder-decoder-based model to fit all questions. However, such a one-size-fits-all strategy may not perform well since complex questions exhibit an uneven distribution in many dimensions, such as question types, involved KB relations, and query structures, resulting in insufficient learning for long-tailed samples under different dimensions. To address this problem, we propose a meta-learning framework for complex question generation. The meta-trained generator can acquire universal and transferable meta-knowledge and quickly adapt to long-tailed samples through a few most related training samples. To retrieve similar samples for each input query, we design a self-supervised graph retriever to learn distributed representations for samples, and contrastive learning is leveraged to improve the learned representations. We conduct experiments on both WebQuestionsSP and ComplexWebQuestion, and results on long-tailed samples of different dimensions have been significantly improved, which demonstrates the effectiveness of the proposed framework.

1 Introduction

Question generation (QG) over knowledge base (KB) aims to generate natural language questions with structured KB queries, which has been widely used to improve the performance of question answering (QA) by data augmentation for the training corpora. It can also help chatbots ask questions during human-computer interaction.

Traditional methods (Jia and Liang, 2016; Seyler et al., 2017) rely on hand-crafted rules and templates to convert KB queries into questions, leading to poor generalization. Recently, neural approaches (Kumar et al., 2019; Chen et al., 2020) leveraged one encoder-decoder-based model to fit the entire training set, then used the trained model to generate questions in the testing phase. However, such a one-size-fits-all strategy may not perform well for generating complex questions which contain multiple KB relations or functional constraints, such as comparison and sorting, and have complex semantic structures. This is due to the uneven distribution of the training set.

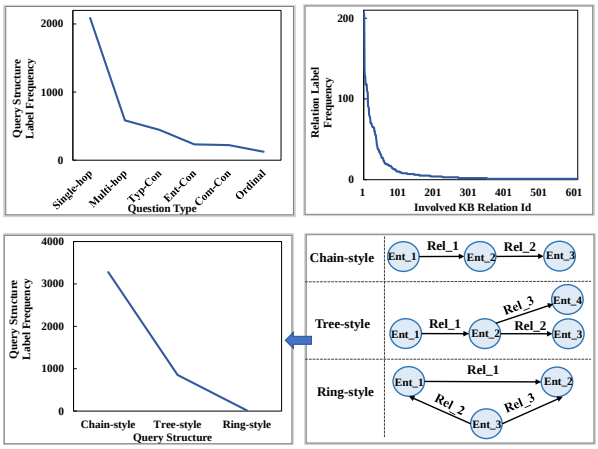

Figure 1: The distribution of question type (Typ-Con as type constraint, Ent-Con as entity constraint, Com-con as comparative constraint), relation type (the involved KB relation in the query) and query structure in training set of WebQSP. We also give the illustration of three types of query structure

Take a widely used dataset WebQuestionsSP (WebQSP) (Yih et al., 2016) as an example, questions are unevenly distributed across multiple dimensions, including question types, involved KB relations, and query structures. As illustrated in Figure 1, single-hop questions are the most common type of questions in the dataset, e.g., the question "What kind of money to take to Bahamas?" relies on a single-hop KB query "(Bahamas, currency_used, ?x)". In contrast, questions with comparative or sorting constraints are the long-tailed samples, e.g., the question " What was the first

*Corresponding author.

Proceedings of the 29th International Conference on Computational Linguistics, pages 6105–6114
October 12–17, 2022.

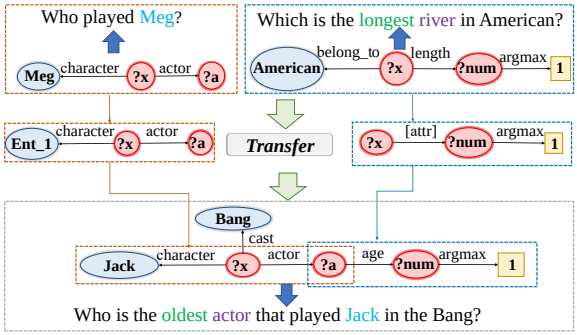

Figure 2: An example of meta-knowledge transferring.

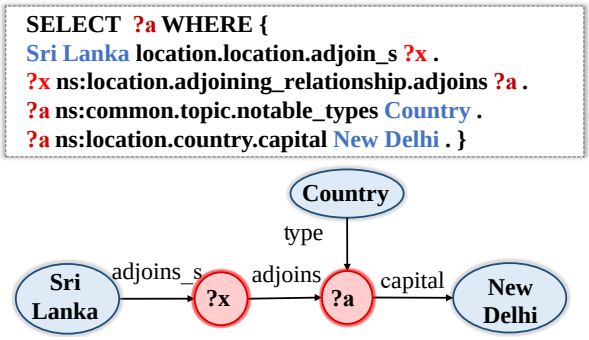

Figure 3: An example of SPARQL query and its corresponding query graph.

book Charles Dickens wrote?" requires sorting over the copyright date of Dickens' books. The same situation occurs in the dimensions of involved KB relations and query structures. Existing neural approaches can be easily biased towards dominant samples and perform poorly on long-tailed ones.

To deal with the problem of data imbalance, we resort to the process of human cognition. As shown in Figure 2, when faced with a rare query, humans can write the corresponding question according to the previously learned query patterns. In this paper, we collectively refer to these universal and transferable query patterns as *meta-knowledge*, which can help generate long-tailed questions. To learn such meta-knowledge, we propose a meta-learning framework for KBQG, namely Meta-CQG. During model training, each sample in the training set is viewed as a query set, and its similar samples are retrieved to form the support set. Our generator adapts to each query set by trials and the supervision signals on the support set. Through the model-agnostic meta-learning (MaML) (Finn et al., 2017) algorithm, the QG model can learn to generalize over varied samples to acquire the meta-knowledge, instead of fitting all samples.

To select similar samples and construct the support sets, we design a self-supervised graph retriever that takes into account the similarity between different samples in different dimensions, including question types, involved KB relations, and query structures. Specifically, the graph retriever encodes the input queries into distributed representations, and the cosine similarities between these vectors denote the similarities between different queries. Due to lack of supervision, we train the graph retriever in a self-supervised way, and contrastive learning is leveraged to improve the learned representations. To demonstrate the effectiveness of the proposed framework, we conduct extensive experiments and ablation studies on two widely-used datasets, and the results on long-tailed samples under different dimensions have been significantly improved.

In general, our main contributions are listed as follows:

- We propose a meta-learning framework for complex question generation over knowledge bases, overcoming the challenge of data imbalance.
- We design a self-supervised graph retriever to select the most similar samples to construct support set during the phase of meta-learning and help the generator better acquire meta-knowledge.
- We demonstrate the effectiveness of the proposed framework on two widely-used datasets and achieve state-of-the-art performance.

2 Preliminary

We aim to generate complex questions from queries, which can be executed on the knowledge base to get the answers to the generated questions. As the query is always displayed in the form of graph, we represent the query $\mathcal{Q}$ with a query graph $\mathcal{G}$. Then we translate the $\mathcal{G}$ to the corresponding complex question.

Knowledge Base. A knowledge base $\mathcal{K}$ is a collection of triples in the form of (s, r, o), where s, r, and o denote subject, relation, and object respectively.

Query Graph. As described in (Qiu et al., 2020), query graph $\mathcal{G}$ is a graph representation of SPARQL query. As shown in Figure 3, Our query graph consists of two types of nodes: variable nodes and non-variable nodes. Variable nodes

represent ungrounded KB nodes or values. A non-variable node can be a grounded KB entity or KB type, such as Sri Lanka and Country.

Complex Question. While a simple question can be answered by a single KB triple, complex questions require more information and even functional operations, such as comparison, aggregation, and sorting.

3 Methodology

In this section, we will describe the proposed framework, Meta-CQG. Figure 4 gives an overview of our framework, which mainly consists of two parts: Query-agnostic Meta Learning (QaML) and the graph retriever. QaML trains a unique generator for each target query by learning the potential features of retrieved similar samples. The graph retriever selects a few most similar samples and construct the support set.

3.1 Query-agnostic Meta Learning

Considering the adaption cost, QaML adopts the MAML algorithm, which can be adapted to the target query via a few training samples in a few training steps. QaML contains two components, the meta learner and the adapted learner. The adapted learner is the adaptive question generator and the meta learner allocates initial parameters for the adapted learner. The meta-learning process can be divided into meta-training process and meta-testing process. In the meta-training process, the meta learner is trained on the support set data to get the adapted learner. Then, we update the meta learner through evaluating on the query set data by the adapted learner. In the meta-testing process, the meta learner is fixed and we leverage the adapted learner to encode the query set data and generate the question. We will describe the meta learner and the adapted learner in detail below.

3.1.1 Meta Learner

In this section, we will describe the meta learner, which aims to learn an initial set of parameters that can quickly adapt to a task-specific learner via similar samples.

In the meta learning setting, a task consists of a support set and a query set. The query set only contains one sample to be generated, and the support set contains the training samples which are most similar to the query set. Take a sample q as an example, the query set $\mathbf{s}_{query} = \{q\}$. We denote the top-N similar samples selected by the graph retriever to form the support set $s_{support} = \{x_{q,1}, ..., x_{q,N}\}$. We denote the query and the question in the support set as $g_{support}$ and $q_{support}$, the query set as g_{query} and q_{query}.

In the meta-training process, the meta learner allocates initial parameters for the adapted learner and is updated through evaluating on the query set data by the adapted learner. We denote the parameter of the meta learner as $\boldsymbol{\theta}$ and the adapted learner as $\boldsymbol{\theta}'$.

During the training phase, the model will be initialized with the parameters of the meta learner. After t iterations of training on $\mathbf{s}_{support}$, the model updates the parameter $\boldsymbol{\theta}$ and gets the adapted learner for the query set. In other words, the model parameterized by $\boldsymbol{\theta}$, is updated to $\boldsymbol{\theta}'$ by standard gradient descent,

$$\boldsymbol{\theta}' \leftarrow \boldsymbol{\theta} - \eta_1 \nabla_{\boldsymbol{\theta}} \mathcal{L}(g_{support}, q_{support}; \boldsymbol{\theta}), \quad (1)$$

where $\mathcal{L}$ is the loss function.

Then, we leverage the adapted learner to obtain the loss on the query set. We apply stochastic gradient descent on the initial parameter $\boldsymbol{\theta}$, i.e. the parameter of the meta learner, by minimizing the loss from query set,

$$\boldsymbol{\theta} \leftarrow \boldsymbol{\theta} - \eta_2 \nabla_{\boldsymbol{\theta}} \mathcal{L}(g_{query}, q_{query}; \boldsymbol{\theta}') \quad (2)$$

where η_2 is meta-learning rate. The pseudo code of meta training process is shown in Algorithm 1.

In the meta-testing process, the meta learner is fixed. We initialize the adapted learner by the parameter of the meta learner.

3.1.2 Adapted Learner

In our task, the adapted learner is the question generator, which translates the generated query graph into a natural language question. Based on the query graph constructed above, we adopt a novel graph-to-sequence model to generate sequences.

Inspired by (Guo et al., 2019), we leverage Densely Connected Graph Convolutional Network (DCGCN) as the graph encoder. It applies dense connectivity among Graph Convolutional Network (GCN) layers. Each DCGCN block consists of two sub-blocks to capture graph structure at different abstract levels. Each sub-block consists of several GCN layers, where each GCN layer is connected to all previous layers. The input of layer l for node u is defined as,

$$g_u^{(l)} = \left[\mathbf{x}_u; \mathbf{h}_u^{(1)}; \ldots; \mathbf{h}_u^{(l-1)}\right], \quad (3)$$

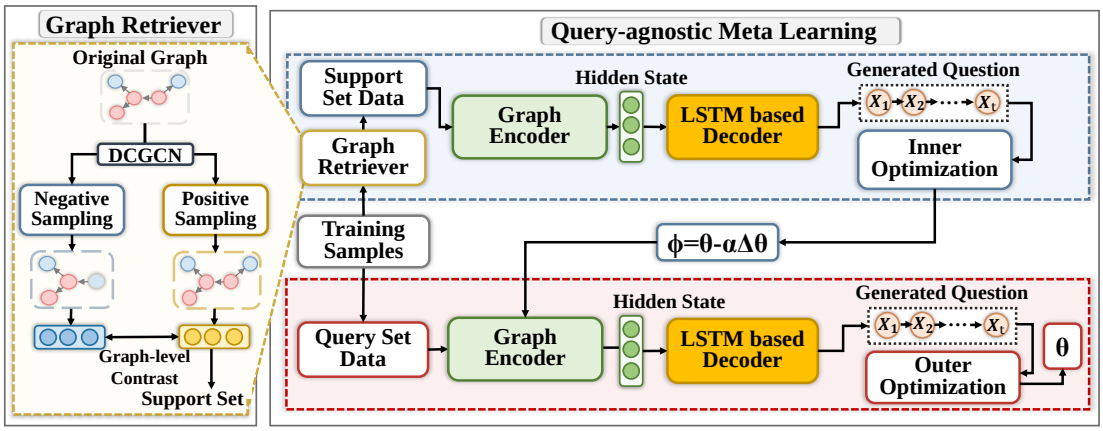

Figure 4: The overall architecture of our framework for question generation over knowledge bases. It can be divided into two parts, i.e., the graph retriever and QaML are shown on the left and right respectively. (i) Graph Retriever (left): from top to bottom, we first pre-train the graph encoder with the task of link prediction. Then we used Similarity-based Contrastive Learning (SCL) to fine-tune the pre-trained graph encoder as the graph retriever. (ii) QaML(right): Given a target query (as query set) in the training set, we retrieve similar samples to construct a support set for the target query. The blue square (above) describes the training process on the support set, while the pink square(below) describes the testing process on the query set.

Algorithm 1 Meta-training process

Require: Dataset: S_{train} ; step hyper parameters: η_1, η_2;

start training:
Randomly initialize $\boldsymbol{\theta}$
for S_i in S_{train} **do**
  Expand $S_i \rightarrow \mathrm{D}_i$
  $\left(\mathcal{S}_i^{\text{support}}, \mathcal{S}_i^{\text{query}}\right) \sim Task_i$.
  Evaluate $\nabla_\theta \mathcal{L}(g_{support}, q_{support}; \boldsymbol{\theta})$ using $\mathcal{S}_i^{\text{support}}$
  Compute adapted parameters with gradient descent:
  $\boldsymbol{\theta}' \leftarrow \boldsymbol{\theta} - \eta_1 \nabla_{\boldsymbol{\theta}} \mathcal{L}(g_{support}, q_{support}; \boldsymbol{\theta})$
  Evaluate $\nabla_\theta \mathcal{L}(g_{query}, q_{query}; \boldsymbol{\theta}')$ using $\mathcal{S}_i^{\text{query}}$
  $\boldsymbol{\theta} \leftarrow \boldsymbol{\theta} - \eta_2 \nabla_{\boldsymbol{\theta}} \mathcal{L}(g_{query}, q_{query}; \boldsymbol{\theta}')$
end for
end training

where $[\cdot;\cdot]$ denotes the concatenation of vectors; $\mathbf{x}_u$ denotes the node embedding of u ; and $\mathbf{h}_u^{(i)}$ denotes the output of layer i for node u. We randomly initialize the embedding of nodes.

In our case, the edge information is crucial as it corresponds to important tokens in the generated question. In order to model both the node and edge information with GNNs, we utilize Levi graph transformation method to transform the input query graph into its equivalent Levi graph(Levi, 1942), which views the predicates as nodes in the graph. Following (Beck et al., 2018), we add reverse and self-loop edges to the Levi graph. To compute the graph-level embedding, we leverage the pooling-based method, which feeds the output node embedding into a fully-connected neural network and applies the element-wise max-pooling operation on all node embeddings to derive the graph embedding $\mathbf{h}_{\mathcal{G}} \in \mathbb{R}^d$.

We adopt an attention-based LSTM decoder (Bahdanau et al., 2014) that generates the output sequence one word at a time. The graph embedding $\mathbf{h}_{\mathcal{G}}$ is used as the initial input of the decoder. We carefully follow the attention mechanism used in (Tu et al., 2016).

3.2 Graph Retriever

To construct the support set, we retrieve samples that are most similar to the target KB query from training set. As queries are always be represented by graphs, this problem is defined as a graph retrieval problem. The core and most challenging part is to measure similarity between two graphs.

Considering both the structure and content information of the query graph, we adopt the neural-based method to measure graph similarity. Most neural-based models only focus on homogeneous graphs and require large-scale annotated data. However, our task lacks gold label about the similarity between graphs. Thus, we train the graph retriever in an unsupervised way. To pre-train the graph, we utilize DCGCN to encode the graph and the link

predication task as downstream task.

Take (g_1, q_1) and (g_2, q_2) as an example. The query representation is calculated as:

$$\begin{aligned} \mathbf{h}_{g1} &= \text{DCGCN}(g_1), \\ \mathbf{h}_{g2} &= \text{DCGCN}(g_2), \end{aligned} \tag{4}$$

where $\mathbf{h}_{g1}$ and $\mathbf{h}_{g2}$ is the graph embedding of g_1 and g_2. And we adopt cosine function to calculate the query similarity:

$$\delta_g = cos(\mathbf{h}_{g1}, \mathbf{h}_{g2}) \tag{5}$$

where δ_g is the similarity score between g_1 and g_2

Only focusing on the query graph itself is hard to learn the mapping relation between query graph and question, leading to many false samples which have similar query graphs but quite different questions. Thus, we propose a Similarity-based Contrastive Learning (SCL) method to fine-tune the pre-trained graph encoder, which utilizes question semantic to emphasize some easily overlooked but crucial information on the query graph.

3.2.1 Contrastive Samples Construction

Each sample contains one query and one question, which is in the form of a graph and a sequence. We construct contrastive samples according to the similarity between queries and questions. We use fixed Bert (Devlin et al., 2018) to encode question and get its representation,

$$\begin{aligned} \mathbf{h}_{q1} &= Bert(q_1), \\ \mathbf{h}_{q2} &= Bert(q_2), \end{aligned} \tag{6}$$

where $\mathbf{h}_{q1}$ and $\mathbf{h}_{q2}$ is denoted as the representation of q_1 and q_2. We adopt cosine function to calculate question similarity δ_s as follow,

$$\delta_q = cos(\mathbf{h}_{q1}, \mathbf{h}_{q2}) \tag{7}$$

The query similarity between two sample has been presented above. Then, we define positive and negative sample selection score λ_{pos}, λ_{neg} as:

$$\begin{aligned} \lambda_{pos} &= (\delta_g - \delta_{gpos})(\delta_q - \delta_{qpos}), \\ \lambda_{neg} &= (\delta_g - \delta_{gneg})(\delta_{qneg} - \delta_q), \end{aligned} \tag{8}$$

where δ_{gpos} , δ_{qpos}, δ_{gneg}, δ_{qneg} is a series of thresholds we set according to the similarity distribution.

If both the two similarities are larger than the thresholds, i.e., $\delta_g > \delta_{gpos}$ and $\delta_q > \delta_{qpos}$, the pair of the two samples is positive.

If either $\delta_q < \delta_{qneg}$ or $\delta_g < \delta_{gneg}$, the sample pair is negative.

3.2.2 Contrastive Training

Our goal is to encourage the graph encoder $\mathcal{E}$ to learn discriminative query representations. After pre-training, We denote the node features in positive samples and negative samples as $\mathbf{X}$ and $\tilde{\mathbf{X}}$, the adjacency matrix as $\mathbf{A}$ and $\tilde{\mathbf{A}}$, the query representation in target, positive and negative sample as $\mathbf{h}_g$, $\mathbf{h}_{gp}$ and $\mathbf{h}_{gn}$. We leverage the discriminator $\mathcal{D}$ to maxmize the mutual information, such that $\mathcal{D}(\mathbf{h}_g, \mathbf{h}_{gn})$ represents the probability scores assigned to this pair.

For the objective, we follow the intuitions from Deep Graph InfoMax (Velickovic et al., 2019) and introduce a contrastive objective $\mathcal{L}_{\text{CL}}$ with a standard binary corss-entropy (BCE) loss. The $\mathcal{L}_{\text{CL}}$ is defined as:

$$\begin{aligned} \mathcal{L}_{\text{CL}} = &\sum_{i=1}^{N} \mathbb{E}_{(\mathbf{X},\mathbf{A})} \left[\log \mathcal{D}(\mathbf{h}_g, \mathbf{h}_{gp})\right] + \\ &\sum_{j=1}^{N} \mathbb{E}_{(\tilde{\mathbf{X}},\tilde{\mathbf{A}})} \left[\log\left(1 - \mathcal{D}(\mathbf{h}_g, \mathbf{h}_{gn})\right)\right], \end{aligned} \tag{9}$$

where N is denoted as the number of positive and negative samples sampled.

After fine-tuning with above loss, we use average pooling to compute the query representation, and retrieve the most similar samples through cosine similarity function.

4 Experiments

In this section, we evaluate Meta-CQG on two widely-used benchmark datasets and show the effectiveness of our method. We first introduce datasets and training settings. Then, we evaluate the proposed model with the state-of-the-art models on both datasets. In addition, we investigate the performance on different dimensions and the influence of different sampling strategies. Finally, we conduct human evaluation to verify the effectiveness of our method.

4.1 Datasets and Preprocessing

We conduct experiments on two widely-used datasets.

WebQSP (Yih et al., 2016) consists of 4,737 question-answer pairs. All the questions are collected through Google Suggest API, and the answers are fetched from Freebase.

Method	CWQ			WebQSP		
	BLEU-4	METEOR	Rouge-L	BLEU-4	METEOR	Rouge-L
L2A	4.01	13.78	30.59	8.01	19.45	32.58
Zero-shot	6.37	16.32	32.10	9.45	21.52	34.78
MHQG	9.35	19.42	35.78	13.34	24.88	39.14
BiGraph2seq	26.01	28.12	53.58	27.86	30.24	62.77
DCGCN	27.36	29.53	54.11	29.82	31.28	63.93
DCGCN+ROS	28.15	30.13	54.69	30.68	32.19	64.56
DCGCN+Finetune	28.43	30.51	55.07	31.37	32.44	64.92
Meta-CQG	**29.52**	**31.72**	**56.03**	**32.87**	**32.92**	**65.09**
w/o Graph Retriever	27.66	29.68	53.88	29.75	31.45	64.08
w/o SCL	28.51	30.63	55.08	31.83	31.73	64.27

Table 1: Experimental results of automatic metrics on two benchmark datasets.

CWQ (Talmor and Berant, 2018) contains 34,689 questions in total. It modified the SPARQLs in WebQSP by including more constraints, and then generated corresponding natural language questions.

Each question in both datasets has a corresponding SPARQL query. We design transformation rules to convert SPARQL query into query graph as our input. For each dataset, we randomly select 80% of the examples for training, 10% for validation, and 10% for testing.

4.2 Baseline Methods

We compare the proposed model with several baseline methods, including the current state-of-the-art model over the two benchmark datasets. **L2A** (Du et al., 2017) is an attention-based Seq-to-Seq model to generate natural language questions from context in open domain conversational systems. **Zero-shot** (Elsahar et al., 2018) is an RNN-based Seq-to-Seq model paired with an original part-of-speech copy action mechanism to generate questions. **MHQG** (Kumar et al., 2019) is a Transformer-based model for automatic generation of multi-hop questions over knowledge bases. **BiGraph2seq** (Chen et al., 2020) is a graph-to-sequence model which leverages Bidirectional Gated Graph Neural Network (Bi-GNN) as the graph encoder to encode the KB subgraphs, and enhance the RNN decoder with copying mechanism. **DCGCN** applys DCGCN as graph encoder and LSTM as decoder. **DCGCN+ROS** leverages DCGCN as the basic model and adopt the Random Over Sampling (ROS) strategy on question type to solve the data imbalance problem. **DCGCN+Finetune** leverages DCGCN to pre-train the training set. When testing, for each sample, we use the graph retriever to build an adaption and finetune the pre-trained model on it.

4.3 Implementation Details

We implement our method on PyTorch platform. The parameters with the best performance on the validation set are selected. For both datasets, the KB embeddings were randomly initialized and updated in the process of training. In meta-learning, we set N=5 when forming the support set. We set $\eta_1 = 1e-4$ (Equation 1) and $\eta_2 = 0.2$ (Equation 2). For SCL, we random select three positive and negative samples for each sample. We set $\delta_{qpos} = 0.8$, $\delta_{spos} = 0.8$, $\delta_{qneg} = 0.6$ and $\delta_{qneg} = 0.4$.

We set the number of DCGCN as 3 and 6 for the two sub-block respectively with an initial learning rate of 0.0003 is adopted as the optimizer. During decoding, beam search with beam size 10 is leveraged.

4.4 Results and Discussion

Following previous studies, we evaluate the performance by a set of N-grams-based metrics for question generation: BLEU-4(Papineni et al., 2002)(B-4.), METEOR(Banerjee and Lavie, 2005), and ROUGE-L(Lin, 2004).

Table 1 shows the results of Meta-CQG and the adopted baselines. Meta-CQG outperforms all the

Dimesion	Categories	BiGraph2Seq			DCGCN			Meta-CQG		
		B-4.	ME.	R-L.	B-4.	ME.	R-L.	B-4.	ME.	R-L.
Question Type	Single-hop(>40%)	28.56	31.38	63.89	30.56	31.79	64.30	33.21	33.15	65.83
	Multi-hop(>20%)	27.60	29.76	63.01	28.86	31.05	64.08	32.03	31.86	64.07
	Type Constraint(<10%)	26.53	28.76	60.23	28.27	30.19	62.76	31.23	31.77	63.38
	Ordinal(<5%)	23.43	27.01	53.99	24.73	27.53	57.07	29.04	30.69	63.25
Query Structure	Chain-Style(>75%)	29.38	31.23	63.65	31.45	32.96	64.45	33.18	33.42	65.79
	Tree-Style(<20%)	23.53	23.46	58.17	25.77	26.08	59.39	32.16	31.93	64.92
	Ring-Style(<5%)	6.78	17.01	48.65	9.48	20.57	50.78	23.23	30.79	56.79
Relation Type	Notable Types(>30%)	28.77	30.94	60.17	31.27	32.48	62.59	33.02	34.08	65.28
	Inventor(<10%)	17.53	22.96	47.47	20.29	24.57	51.86	27.33	31.88	62.13
	Award Honor(<1%)	4.58	14.77	33.58	7.33	15.45	37.54	20.29	24.11	52.77

Table 2: Experimental results of automatic metrics on different dimensions.

baselines on the two benchmark datasets. Specifically, Meta-CQG improves the BLEU-4 score by 3.51 on CWQ, 5.01 on WebQSP compared with BiGraph2seq. Meanwhile, Meta-CQG exceeds the baselines by a larger margin on METEOR and ROUGE-L.

Graph-to-Seq models (BiGraph2seq, DCGCN, DCGCN+ROS, DCGCN+Finetune and Meta-CQG) outperform the Seq-to-Seq models (L2A, Zero-shot and MHQG) on both datasets, which indicates the advantages of GNN-based encoders for modeling query graphs, since the RNN-based model and the transformer-based model ignores the explicit graph structure of query graphs. In addition, DCGCN-based models (DCGCN, DCGCN+ROS, DCGCN+Finetune and Meta-CQG) outperform BiGraph2seq, which indicates that DCGCN better captures the non-local interactions between the nodes compared with Bi-GNN.

Meta-CQG outperforms DCGCN+ROS by more than 1.37 on BLEU-4, which indicates that the proposed model is more effective in reducing the data imbalance problem compared with existing data re-balancing approaches. Moreover, it outperforms DCGCN+Finetune by more than 1.09 on BLEU-4, which indicates that the meta-learning framework is better at transferring meta-knowledge compared with the pretrain-finetune framwork.

4.5 Ablation study

To further analyze the effectiveness of different components, we conduct ablation studies which remove the graph retriever and SCL in Meta-CQG. The results are shown in Table 1.

Graph Retriever. To evaluate the effectiveness of graph retriever, we remove it and randomly select samples for each training sample (w/o Graph Retriever). The performance has dropped by more than 1.86 on BLEU-4. This indicates that random select samples cannot provide task-specific knowledge for the training sample, and they may introduce noise during training.

SCL. To evaluate the effectiveness of SCL, we remove the contrastive learning loss (w/o SCL). The performance has dropped by more than 1.01 on BLEU-4. This indicates that SCL utilizes question semantic to enhance query graph information.

4.6 Analyses on Different Dimensions

As mentioned above, complex questions are imbalanced in multiple dimensions. We divide the questions in the WebQSP according to each dimension and evaluate the performance of BiGraph2Seq, DCGCN, and Meta-CQG. We sample some categories from majority types and minority types respectively for each dimension. The proportion of each type in the training set is also listed after the categories.

As shown in Table 2, Meta-CQG achieves the best performance in all three dimensions.

In each dimension, Meta-CQG outperforms the other two baselines on minority types (i.e., <10% in training set), which shows the learning ability of Meta-CQG on imbalanced data. The poor performance of BiGraph2Seq and DCGCN on minority types also verifies that the data imbalance problem greatly affects the model performance.

Strategies	B-4.	ME.	R-L.
Question Type	31.07	32.87	64.97
Relation Type	31.22	32.39	64.43
Query Structure	30.09	31.53	64.27
Our Model	**32.87**	**32.92**	**65.09**

Table 3: Experimental results of different sampling strategies.

Results	CWQ			WebQSP		
	Nat.	Sem.	Cor.	Nat.	Sem.	Cor.
Win	19	37	28	35	35	29
Tie	79	59	69	59	59	64
Lose	2	4	3	6	6	7

Table 4: Wins, losses, and ties of Meta-CQG against the current SOAT (BiGraph2seq) based on the manual evaluation.

In addition, Meta-CQG outperforms the baselines on majority types (i.e., >30% in training set). This may be some samples in a majority type may be in minority type of another dimension. For example, a single-hop question may contain relation of Award Honor. Therefore, Meta-CQG is able to find similar samples across multiple dimensions.

4.7 Analyses on Different Sampling Strategies

We design different sampling strategies to verify the effectiveness of our graph retriever. First, we devise different retrievers according to the three dimensions we mentioned above. For each dimension, the retriever randomly selects a few samples that belong to the same category with the sample to be generated. The results shown in Table 3 demonstrate the effectiveness of our model and verify that our model is able to comprehensively consider the imbalance in all dimensions.

4.8 Human Evaluation

We randomly choose 100 questions from the test set of each dataset. We pair the questions generated by our model and BiGraph2seq. Two human annotators are asked to judge which is better in pairs from three aspects: naturalness, correctness, and semantic. Results are shown in Table 4. Our model outperforms BiGraph2seq as it has more winning instances than losing instances on all two datasets. These results indicate that our model improves the quality of questions from the three aspects.

5 Related Work

Question Generation over Knowledge Bases Most recent works for KBQG mainly adopt encoder-decoder models, and focus on enriching the input information. In (Serban et al., 2016) and (Indurthi et al., 2017), recurrent neural networks are introduced for generating natural language questions from KB facts. To address the challenge of unseen predicates and entity types, (Elsahar et al., 2018) leverages auxiliary contexts in the WiKidata corpus in an encoder-decoder architecture. However, the context cannot cover all predicates. Thus, (Liu et al., 2019) presents a neural encoder-decoder model that integrates diversified off-the-shelf contexts. To tackle the semantic drift problem, (Bi et al., 2020) presents a knowledge-enriched, type-constrained, and grammar-guided model. (Kumar et al., 2019) proposes a model for generating complex multi-hop and difficulty-controllable questions over knowledge bases. To model the graph-structured data, (Chen et al., 2020) applied a bidirectional Gated Graph Neural Network model to encode the KB subgraph. However, existing methods train one model to fit all questions, ignoring the data imbalance in the real world.

Meta-Learning Meta-Learning, i.e.learning-to-learn, aims to build efficient algorithms that can learn the new task quickly. In pursuing this problem, there are three categories of meta-learning methods: learning a metric space to compare low-resource testing samples and rich training samples (Snell et al., 2017; Koch et al., 2015), using an additional meta-learner to update the original learner with a few training examples (Ravi and Larochelle, 2016) and learning a good initialization parameter for fast adaptation(Finn et al., 2017). In this work, we follow the third idea and propose a meta-learning framework based on MaML to solve the data imbalance problem in complex question generation tasks.

6 Conclusion

In this paper, we focus on the task of complex question generation over knowledge bases. We propose a meta-learning framework for complex question generation, namely Meta-CQG, to deal with the data imbalance problem. To consider the imbalance of all dimensions, we adopt the MaML method to train a unique generator for each sam-

ple to be generated via a few most similar training samples. Specially, we design a self supervised graph retriever to flexibly retrieve most similar samples. We evaluate the effectiveness of Meta-CQG on two widely-used benchmark datasets, and it outperforms all the baselines.

7 Acknowledgement

Thanks to reviewers for their helpful comments on this paper. This paper is funded by the National Natural Science Foundation of China (Nos.62172393, U1836206 and U21B2046), Zhongyuanyingcai program-funded to central plains science and technology innovation leading talent program (No.204200510002) and Major Public Welfare Project of Henan Province (No.201300311200).

References

Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. 2014. Neural machine translation by jointly learning to align and translate. *arXiv preprint arXiv:1409.0473*.

Satanjeev Banerjee and Alon Lavie. 2005. Meteor: An automatic metric for mt evaluation with improved correlation with human judgments. In *Proceedings of the acl workshop on intrinsic and extrinsic evaluation measures for machine translation and/or summarization*, pages 65–72.

Daniel Beck, Gholamreza Haffari, and Trevor Cohn. 2018. Graph-to-sequence learning using gated graph neural networks. *arXiv preprint arXiv:1806.09835*.

Sheng Bi, Xiya Cheng, Yuan-Fang Li, Yongzhen Wang, and Guilin Qi. 2020. Knowledge-enriched, type-constrained and grammar-guided question generation over knowledge bases. *arXiv preprint arXiv:2010.03157*.

Yu Chen, Lingfei Wu, and Mohammed J Zaki. 2020. Toward subgraph guided knowledge graph question generation with graph neural networks. *arXiv preprint arXiv:2004.06015*.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*.

Xinya Du, Junru Shao, and Claire Cardie. 2017. Learning to ask: Neural question generation for reading comprehension. *arXiv preprint arXiv:1705.00106*.

Hady Elsahar, Christophe Gravier, and Frederique Laforest. 2018. Zero-shot question generation from knowledge graphs for unseen predicates and entity types. *arXiv preprint arXiv:1802.06842*.

Chelsea Finn, Pieter Abbeel, and Sergey Levine. 2017. Model-agnostic meta-learning for fast adaptation of deep networks. In *International Conference on Machine Learning*, pages 1126–1135. PMLR.

Zhijiang Guo, Yan Zhang, Zhiyang Teng, and Wei Lu. 2019. Densely connected graph convolutional networks for graph-to-sequence learning. *Transactions of the Association for Computational Linguistics*, 7:297–312.

Sathish Reddy Indurthi, Dinesh Raghu, Mitesh M Khapra, and Sachindra Joshi. 2017. Generating natural language question-answer pairs from a knowledge graph using a rnn based question generation model. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 1, Long Papers*, pages 376–385.

Robin Jia and Percy Liang. 2016. Data recombination for neural semantic parsing. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 12–22.

Gregory Koch, Richard Zemel, Ruslan Salakhutdinov, et al. 2015. Siamese neural networks for one-shot image recognition. In *ICML deep learning workshop*, volume 2, page 0. Lille.

Vishwajeet Kumar, Yuncheng Hua, Ganesh Ramakrishnan, Guilin Qi, Lianli Gao, and Yuan-Fang Li. 2019. Difficulty-controllable multi-hop question generation from knowledge graphs. In *International Semantic Web Conference*, pages 382–398. Springer.

Friedrich Wilhelm Levi. 1942. *Finite geometrical systems: six public lectues delivered in February, 1940, at the University of Calcutta*. University of Calcutta.

Chin-Yew Lin. 2004. Rouge: A package for automatic evaluation of summaries. In *Text summarization branches out*, pages 74–81.

Cao Liu, Kang Liu, Shizhu He, Zaiqing Nie, and Jun Zhao. 2019. Generating questions for knowledge bases via incorporating diversified contexts and answer-aware loss. *arXiv preprint arXiv:1910.13108*.

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. Bleu: a method for automatic evaluation of machine translation. In *Proceedings of the 40th annual meeting of the Association for Computational Linguistics*, pages 311–318.

Yunqi Qiu, Kun Zhang, Yuanzhuo Wang, Xiaolong Jin, Long Bai, Saiping Guan, and Xueqi Cheng. 2020. Hierarchical query graph generation for complex question answering over knowledge graph. In *Proceedings of the 29th ACM International Conference on Information & Knowledge Management*, pages 1285–1294.

Sachin Ravi and Hugo Larochelle. 2016. Optimization as a model for few-shot learning.

Iulian Vlad Serban, Alberto García-Durán, Caglar Gulcehre, Sungjin Ahn, Sarath Chandar, Aaron Courville, and Yoshua Bengio. 2016. Generating factoid questions with recurrent neural networks: The 30m factoid question-answer corpus. *arXiv preprint arXiv:1603.06807*.

Dominic Seyler, Mohamed Yahya, and Klaus Berberich. 2017. Knowledge questions from knowledge graphs. In *Proceedings of the ACM SIGIR International Conference on Theory of Information Retrieval*, pages 11–18.

Jake Snell, Kevin Swersky, and Richard Zemel. 2017. Prototypical networks for few-shot learning. *Advances in neural information processing systems*, 30.

Alon Talmor and Jonathan Berant. 2018. The web as a knowledge-base for answering complex questions. *arXiv preprint arXiv:1803.06643*.

Zhaopeng Tu, Zhengdong Lu, Yang Liu, Xiaohua Liu, and Hang Li. 2016. Modeling coverage for neural machine translation. *arXiv preprint arXiv:1601.04811*.

Petar Velickovic, William Fedus, William L Hamilton, Pietro Liò, Yoshua Bengio, and R Devon Hjelm. 2019. Deep graph infomax. *ICLR (Poster)*, 2(3):4.

Wen-tau Yih, Matthew Richardson, Christopher Meek, Ming-Wei Chang, and Jina Suh. 2016. The value of semantic parse labeling for knowledge base question answering. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 201–206.

Graph-to-Text Generation with Dynamic Structure Pruning

Liang Li[1,2], **Ruiying Geng**[3], **Bowen Li**[3], **Can Ma**[1]*, **Yinliang Yue**[1], **Binhua Li**[3] , **Yongbin Li**[3]*

[1]Institute of Information Engineering, Chinese Academy of Sciences, Beijing, China
[2]School of Cyber Security, University of Chinese Academy of Sciences, Beijing, China
[3]DAMO Academy, Alibaba Group

{liliang, macan, yueyinliang}@iie.ac.cn
{ruiying.gry, binhua.lbh, shuide.lyb}@alibaba-inc.com

Abstract

Most graph-to-text works are built on the encoder-decoder framework with cross-attention mechanism. Recent studies have shown that explicitly modeling the input graph structure can significantly improve the performance. However, the vanilla structural encoder cannot capture all specialized information in a single forward pass for all decoding steps, resulting in inaccurate semantic representations. Meanwhile, the input graph is flatted as an unordered sequence in the cross attention, ignoring the original graph structure. As a result, the obtained input graph context vector in the decoder may be flawed. To address these issues, we propose a Structure-Aware Cross-Attention (SACA) mechanism to re-encode the input graph representation conditioning on the newly generated context at each decoding step in a structure aware manner. We further adapt SACA and introduce its variant Dynamic Graph Pruning (DGP) mechanism to dynamically drop irrelevant nodes in the decoding process. We achieve new state-of-the-art results on two graph-to-text datasets, LDC2020T02 and ENT-DESC, with only minor increase on computational cost.

1 Introduction

Data-to-text task aims to generate a natural language description from structural or semi-structural data, such as tables (Wiseman et al., 2017), Abstract Meaning Representation (AMR) graphs (Banarescu et al., 2013), and Knowledge Graphs (KG) (Cheng et al., 2020). It helps people get the key points of the input data and makes the stored information accessible to a broader audience of end-users. There have been several practical application scenarios in this field, such as biography generation (Lebret et al., 2016), basketball news generation (Wiseman et al., 2017), and advertising text generation (Shao et al., 2019). This paper focuses on generation from graph structures in AMR and KG, referred to as graph-to-text.

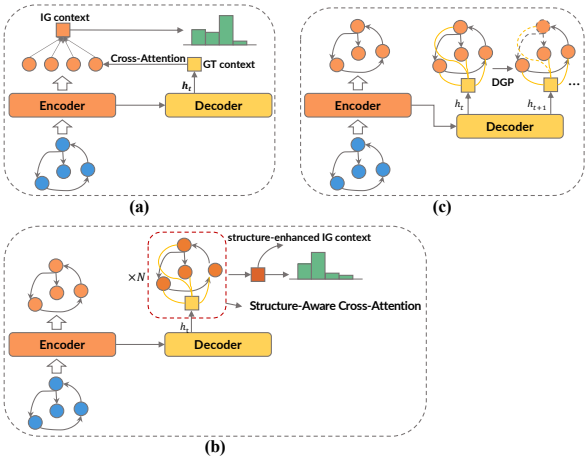

Figure 1: (a) denotes an encoder-decoder framework with the cross-attention mechanism where IG and GT contexts denote the input graph and generated text graph contexts, respectively. (b) is an example of Structure-Aware Cross-Attention. The dotted lines in (c) denote the pruned edges and nodes.

In recent years, encoder-decoder with the cross-attention mechanism has been the de facto framework for graph-to-text tasks (shown in Figure 1(a)). Given an input graph, the encoder first computes vector representations for the graph nodes. On the decoding side, Input Graph (IG) context vector is obtained via cross-attention based on the partially Generated Text (GT) at each time step, then the next target token is finally predicted. Unlike conventional text-to-text tasks, the structural nature of the input graph makes it unsuitable to naively apply sequential encoder-decoder architecture to the graph-to-text task. To alleviate this issue, recent studies (Song et al., 2018; Damonte and Cohen, 2019; Cai and Lam, 2020) proposed to utilize the graph encoder to capture the input graph structure. These works have demonstrated that explicitly modeling the graph structure can bring benefits to the model performance.

*Corresponding authors: Can Ma, Yongbin Li

Proceedings of the 29th International Conference on Computational Linguistics, pages 6115–6127
October 12–17, 2022.

Although equipped with the structure-aware modeling, it is still hard for the encoder to capture all specialized information for graph-to-text generation. It is evidenced by recent studies (Liu et al., 2019; Li et al., 2021) that a vanilla structural encoder cannot capture the accurate semantic representation of the input structural data effectively. Auxiliary supervision has been shown to be helpful, but effective auxiliary tasks are not easy to design and may not generalize well to different datasets. We suspect that it is challenging for the encoder to encode all relevant information into node representations in a single forward pass for all the decoding steps, especially if the input graph structure is complex. Besides the encoder side, few works have focused on the decoder side for graph-to-text tasks. Considering the ordinary cross-attention mechanism, the representations of input data obtained from the encoder are still treated as an *unordered* node representation sequence. We conjuncture that this *plain* cross-attention does not take full advantage of the input graph structure and therefore may harm the model performance.

Current models with graph encoder and cross-attention may yield inaccurate input graph context representation due to the deficiency on both encoder and decoder as we discussed before. To tackle the above problems and avoid introducing auxiliary tasks, we propose a novel **Structure-Aware Cross-Attention** (SACA) mechanism. Apart from the *plain* cross-attention, our SACA re-encodes the input graph conditioning on the newly generated context in a *structure-aware* fashion. Other than a single forward pass, specialized representations from the source side are built adaptively at each decoding step, which makes the decoder easily exploit relevant-only information for prediction. More specifically, as shown in Figure 1(b), we construct a joint graph, in which we explicitly treat the generated text context vector as an additional node and connect it with nodes in the input graph at each decoding step. We implement SACA using the relational graph attention network (RGAT, Shaw et al. 2018). Furthermore, we stack multiple layers of SACAs to perform deep interactions between the generated text context vector and input node representations. Finally, we fetch the node representation corresponding to the newly added node as the structure-enhanced input graph context to predict the target token.

In practice, we notice that some nodes become irrelevant and uninformative as the decoding goes on. These nodes are distracting and can disturb the generation process. Intuitively, the decoder should dynamically discard the unrelated parts of the graph at different decoding steps. In other words, the joint graph structure should be dynamically adjusted. To this end, we adapt SACA and propose its variant **Dynamic Graph Pruning** (DGP) mechanism (shown in Figure 1(c)). DGP prunes the structure of the joint graph via the gate mechanism to achieve sparse connections between the nodes based on the generated text context.

We conduct experiments on two graph-to-text datasets, LDC2020T02[1] and ENT-DESC (Cheng et al., 2020), to verify the effectiveness of the proposed approach. Empirical results show that our proposed methods achieve new state-of-the-art results on the two datasets. Further experiments indicate that SACA and DGP do not reduce the diversity of the generated text and can better handle complex graphs. Meanwhile, additional investigation reveals that SACA and DGP only bring minor increase on the model size and inference time.

2 Related Works

Graph-to-text is a challenging task which aims at generating a descriptive text from the structured knowledge, such Knowledge Graph (KG), and Abstract Meaning Representation (AMR) graphs. It is helpful for interpretability of KGs in general (Schmitt et al., 2020) and knowledge-based question answering (Hui et al., 2022; Wang et al., 2022; Fu et al., 2020; Qin et al., 2022).

In recent years, most graph-to-text methods have been built based on the encoder-decoder architecture. This kind of method usually consists of a structural encoder and a decoder. The structural encoder aims to model the structure information into the representation of the input graph. Song et al. (2018) first propose the graph recurrent networks (GRNs) to encode the AMR node directly. And then, some works (Shi et al., 2020; Chen et al., 2020) introduce the Graph Neural Networks (GNNs) as the structural encoder, which updates the representations of nodes based on their immediate neighbors. To integrate both local and non-local features and learn a better structural representation of a graph, Guo et al. (2019) introduce the dense connection, allowing deeper GCNs. Unlike the local information aggregation scheme, Zhu et al.

[1] https://catalog.ldc.upenn.edu/LDC2020T02

(2019); Cai and Lam (2020) propose the Graph Transformer that uses explicit relation encoding and allows direct communication between two distant nodes.

A recently proposed neural abstractive Multi-Document Summarization (MDS) model, GraphSumm (Li et al., 2020), also considers the input graph structure during decoding. The biggest difference between Graphsum and our proposed SACA is that the former only introduces one graph attention layer in each decoder layer. SACA, on the other hand, injects graph structure into decoding by re-encoding the input graph. Specifically, it re-computes the input graph representation by conditioning it on the newly generated text at each decoding step.

Recent approaches try to apply the Pre-trained Language Models (PLMs) (Kenton and Toutanova, 2019; Raffel et al., 2019) into the graph-to-text generation. Particularly, Ribeiro et al. (2021) propose to utilize the adapter method (Pfeiffer et al., 2020) to encode graph structure into PLMs and only train graph structure-aware adapter parameters. In this way, they avoid catastrophic forgetting while maintaining the topological structure of the graph.

3 Approach

We expect that developing graph-to-text generation should benefit from the recent advance on pre-trained language models (PLMs) (Lewis et al., 2020; Raffel et al., 2019). To explicitly encode the input graph structure into PLMs while alleviating the catastrophic forgetting problem, we consider SA-RGCN (Ribeiro et al., 2021) as our baseline model. SA-RGCN is an adapter method to encode graph structure into PLMs. The overall illustration of our model architecture is shown in Figure 2(a). In this section, we first introduce how to represent the input graph and the architecture of our baseline SA-RGCN. Then, we depict our proposed Structure-Aware Cross-Attention (SACA) in details. Lastly, we adapt SACA and propose its variant Dynamic Graph Pruning (DGP).

3.1 Graph Representation

Let $\mathcal{G}_0 = (\mathcal{V}_0, \mathcal{E}_0, \mathcal{R}_0)$ denote a multi-relational and directed graph with nodes $v_i \in \mathcal{V}_0$ and labeled edges $(v_i, r, v_j) \in \mathcal{E}_0$, where $r \in \mathcal{R}_0$ is the relation type. Following previous work (Beck et al., 2018), we convert each input graph into a Levi graph $\mathcal{G}_l = (\mathcal{V}_l, \mathcal{E}_l)$, which is an unlabeled and connected bipartite graph. Specifically, each labeled edge $(v_i, r, v_j) \in \mathcal{E}_0$ is transformed into two unlabeled edges $(v_i, r), (r, v_j) \in \mathcal{E}_l$. In addition, we add a reverse edge (v_j, v_i) for each default edge (v_i, v_j). Therefore, each Levi graph $\mathcal{G}_l$ contains two type relations $\mathcal{R}_l = \{d, r\}$, where d and r denote the default and reverse edge, respectively. To better take advantage of the PLMs, we convert each $\mathcal{G}_l$ into a new token graph $\mathcal{G} = (\mathcal{V}, \mathcal{E}, \mathcal{R})$, where each token of a node in $\mathcal{V}_l$ becomes a node $v \in \mathcal{V}$.

3.2 Pretrained LMs with Structural Adapters

To inject graph structural bias into PLMs, we incorporate the structural adapter (Ribeiro et al., 2021) into the PLMs encoder. As shown in Figure 2 (a), we add a structural adapter after each transformer encoder block on the encoder. Figure 2 (b) illustrates the architecture of a structural adapter, in where a relational GCN (RGCN) (Schlichtkrull et al., 2018) layer computes the node representation based on the local neighborhood of node $v \in \mathcal{V}$. Formally, at each layer l, given the encoder layer representation h_v^l, a structural adapter computes the representation for v by the following:

$$g_v^l = \sum_{r \in \mathcal{R}} \sum_{u \in N_r(v)} \frac{1}{|\mathcal{N}_r(v)|} W_r^l \text{LN}(h_v^l), \quad (1)$$

$$z_v^l = W_e^l(\sigma(g_v^l)) + h_v^l, \quad (2)$$

where $\text{LN}(\cdot)$ denotes layer normalization. $\mathcal{N}_r(v)$ is the sef of immediate neighbors under relation $r \in \mathcal{R}$. W_r^l encodes the edge type between the nodes u and v. σ is the activation function.

We add an FNN adapter after each transformer decoder block to adapt the language model to the graph-to-text task. Given the output $\hat{h}_v^l$ of the l th transformer decoder block, the adapter representation is computed as:

$$\hat{z}_v^l = W_o^l(\sigma(W_p^l \text{LN}(\hat{h}_v^l))) + \hat{h}_v^l, \quad (3)$$

where W_o^l and W_d^l denote learnable parameters.

3.3 Structure-Aware Cross-Attention

We argue that the input graph context representation obtained by the *plain* cross-attention may be inaccurate. The reason is twofold. First, it is not easy for the graph encoder to capture all specialized information required for generation in a single forward pass. Therefore, a single encoder without any auxiliary assistant may not be effective in capturing the accurate semantic representation (Liu

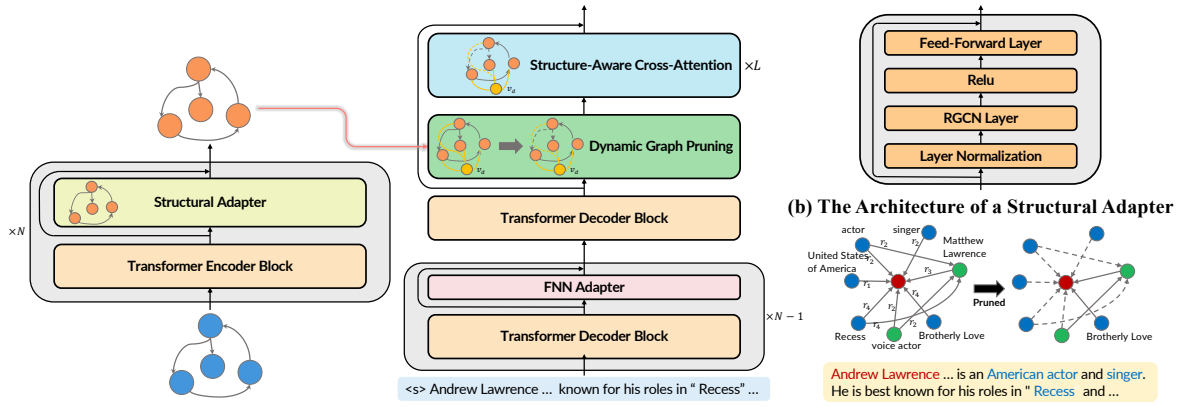

(a) Overview of Architecture (b) The Architecture of a Structural Adapter (c) An example of Dynamic Graph Pruning

Figure 2: Illustration of the proposed model architecture. (a) is an overview of our model. (b) is the architecture of a structural adapter. (c) is an example of Dynamic Graph Pruning, where $r_1 \sim r_4$ denote the relations: "**country of citizenship**", "**occupation**", "**sibling**", and "**cast member**", respectively. The dummy lines in (c) denote the pruned edges.

et al., 2019; Li et al., 2021). In other words, the graph representation encoded by the graph encoder may be inaccurate. Second, during decoding, the decoder treats structural data as an unordered node sequence, which ignores the input graph structure. However, the graph structure has been proven to play an essential role in the graph representation and may offer clues about which nodes are more related to the generated text context.

To tackle the above challenge, we propose a Structure-Aware Cross-Attention (SACA) mechanism, which re-encodes the input graph representation by conditioning on the newly generated context. Specifically, we first build a joint graph, in which we view the generated text (GT) context as a new node v_d and explicitly connect it to each node in the input graph $\mathcal{G}$ at each decoding step. The corresponding reverse edges are also added. The joint graph can be formulated as $\mathcal{G}_{joint} = (\mathcal{V}_{joint}, \mathcal{E}_{joint}, \mathcal{R})$, where $\mathcal{V}_{joint} = \{v_d\} \cup \mathcal{V}$ and $\mathcal{E}_{joint} = \{(v_i, v_d), (v_d, v_i); v_i \in \mathcal{V}\} \cup \mathcal{E}$. We use the representations from the encoder for the node from $\mathcal{V}$ and the hidden state from the last transformer decoder block as the representation for the GT context node.

To induce the representations for the nodes in the joint graph $\mathcal{G}_{joint}$ and facilitate introducing Dynamic Graph Pruning (in Section 3.4), we consider graph neural network built on graph attention framework (GAT) (Shaw et al., 2018). Moreover, we employ the relational graph attention network (RGAT) implemented by Shaw et al. (2018) to model the relation between neighbor nodes. Specifically, at each RGAT layer l, we update the representation h_v^l of each node $v \in \mathcal{G}_{joint}$ by:

$$s_{v,u} = \frac{W^q h_v^{l\,T}(W^k h_u^l + E^r_{\mathcal{R}(v,u)})}{\sqrt{m}}, \quad (4)$$

$$\alpha_{v,u} = \frac{e^{s_{v,u}}}{\sum_{u' \in \mathcal{N}_v} e^{s_{v,u'}}}, \quad (5)$$

$$h_v^{l+1} = \sigma\Big(\sum_{u \in \mathcal{N}_v} \alpha_{v,u} W_v h_u^l\Big), \quad (6)$$

where $E^r_{\mathcal{R}(v,u)}$ means the embedding of the relation between node v and u. m denotes the hidden dimension of RGAT. Finally, the representation vector $h^L_{v_d}$ corresponding to the GT context node v_d is fetched and used as the structure-enhanced input graph context vector for token prediction.

In conclusion, SACA provides two advantages. First, it re-encodes the input graph by conditioning its representation on the newly generated context. As a result, we build specialized representations which make it easier for the decoder to exploit relevant-only information for prediction at each decoding step. Second, the re-encoding explicitly injects structural bias into input graph context modeling, helping the decoder obtain a more accurate input graph context vector. The proposed SACA can be plugged after the last transformer decoder block as shown in Figure 2 (a).

3.4 Dynamic Graph Pruning

In practice, we notice that some nodes become irrelevant and uninformative as the decoding goes on. These unrelated nodes are distracting and can even disturb the subsequent generation. Intuitively, the decoder should dynamically prune the joint

graph at different decoding steps. For this purpose, we adapt SACA and propose its variant Dynamic Graph Pruning (DGP) mechanism, which aims to dynamically drop the redundant nodes in the joint graph according to the generated text during decoding. The DGP employs the gate mechanism to sparse the connection between a node and its immediate neighbors in the joint graph to achieve graph pruning. Specifically, at each decoding step t, for each node v in the joint graph, we formulate its gate as bellow:

$$g_v = sigmoid(W_g^T tanh(W_e h_v + W_d h_t^d)), \quad (7)$$

where W_g, W_e, and W_d are learnable parameters. And h_v is the representation of node v and h_t^d is the decoder hidden state at decoding step t, which is usually considered as the representation of the generated text context. The value of gate $g_v \in R$ decides whether the node v_i should be dropped or not. Correspondingly, we apply the gate value to multiple SACA layers invariably by modifying the attention weights in SACA (Equation 5) as follows:

$$\alpha_{v,u} = \frac{g_u \odot e^{s_{v,u}}}{\sum_{u' \in \mathcal{N}_v} g_{u'} \odot e^{s_{v,u'}}}. \quad (8)$$

Intuitively, if the value of gate g_u is close to 0, the connections between node u with all its immediate neighbors will be largely weaken. That is, the node is removed from the joint graph. Specifically, the attention score $\alpha_{v,u}$ measures the relevance between any two nodes, v and u, in the joint graph, while the gate g_v models the relevance between the node v and the generated text context h_t.

As a shown example in Figure 2 (c), the red node represents the main entity. Initially, the main entity connects with all its neighbor nodes. As the decoding goes on, some nodes are redundant for the subsequent generation. For example, the nodes "actor" has been described, and node "voice actor" is also covered by the generated text. Therefore, DGP discards these nodes by giving them gates with small values.

We observed that the values of the gates calculated by Equation 7 are almost equal to 1, indicating that the model does not actively learn to prune a graph. Inspired by Xue et al. (2020), we further introduce a regularization item, encouraging the network to turn off more gates and generate more sparse connections between nodes in the input graph. We formulate it as follows:

$$L_{DGP} = \frac{\sum_{t=1}^{|y|} \|Gate_t\|_1}{|y|}, \quad (9)$$

where $Gate_t = \{g_v | v \in \mathcal{V}\}$. $\|*\|_1$ means $L1$ norm regularizer.

	ENT-DESC	LDC2020T02
#train/dev/test	88,650/11,081/11,081	55,635/1,722/1,898
#relations	967	157
Avg #nodes	18.0	14.2
Avg #triples	27.4	14.8
Avg length	31.0	95.0

Table 1: Dataset statistics of ENT-DESC and LDC2020T02.

3.5 Training

Given a reference output $y = \{y_1, y_2, ..., y_T\}$ and an input graph $\mathcal{G}$, we use the cross-entropy loss as the objective function of graph-to-text generation:

$$L_{lm} = \sum_{t=1}^{|y|} \log p(y_t | y_{1:t-1}, \mathcal{G}, \theta). \quad (10)$$

Finally, the overall objective function consists of two parts:

$$L = L_{lm} + \lambda L_{DGP}, \quad (11)$$

where λ is a tunable hyper-parameter and is used to make a trade-off between the cross-entropy loss and the regularization item. Intuitively, the L_{DGP} object encourages the model to learn how to prune the graph, and the L_{lm} trains the model to generate the text according to the graph and restrains DGP from pruning too much.

4 Experiments

4.1 Datasets

We demonstrate the effectiveness of our models on two graph-to-text datasets: LDC2020T02 and ENT-DESC (Cheng et al., 2020) LDC2020T02 is an AMR-to-Text dataset and has 55,635/1,722/1,898 instances for training, development, and testing. We follow Ribeiro et al. (2021) to preprocess the AMR graphs and tokenize the sentences. Each instance contains a sentence and an AMR graph. ENT-DESC is a large-scale and challenging dataset generating text from the Knowledge Graph (KG-to-Text). Each instance contains a KG consisting of a main entity and a few topic-related entities. The

Models	LDC2020T02				
	BLEU	METEOR	ChRF++	$\mathcal{M}$	BERTScore
LDGCN (Zhang et al., 2020b)	34.3	38.2	63.7	-	-
SPRING (Bevilacqua et al., 2021)	44.9	-	72.9	-	-
FINETUNE (Ribeiro et al., 2021)	$41.6_{\pm 0.6}$	-	$70.4_{\pm 0.5}$	$78.5_{\pm 0.2}$	$96.0_{\pm 0.1}$
ADAPT (Ribeiro et al., 2021)	$43.0_{\pm 0.2}$	-	$71.3_{\pm 0.1}$	$79.3_{\pm 0.1}$	$96.2_{\pm 0.1}$
SA-RGCN (Ribeiro et al., 2021)	$48.0_{\pm 0.2}$	-	$73.2_{\pm 0.1}$	$80.1_{\pm 0.3}$	$96.3_{\pm 0.1}$
FINETUNE‡	$41.55_{\pm 0.58}$	$42.06_{\pm 0.21}$	$70.62_{\pm 0.34}$	$78.30_{\pm 0.32}$	$96.02_{\pm 0.12}$
SA-RGCN‡	$47.85_{\pm 0.22}$	$45.11_{\pm 0.16}$	$73.53_{\pm 0.19}$	$80.31_{\pm 0.24}$	$96.41_{\pm 0.03}$
Ours	$\mathbf{48.78}_{\pm 0.08}$	$\mathbf{46.12}_{\pm 0.12}$	$\mathbf{74.35}_{\pm 0.09}$	$\mathbf{80.69}_{\pm 0.41}$	$\mathbf{96.62}_{\pm 0.02}$
Models	**ENT-DESC**				
	BLEU	METEOR	ChRF++	ROUGE-L	PARENT
S2S (Bahdanau et al., 2015)	6.8	10.8	-	40.7	10.0
GraphTransformer (Koncel-Kedziorski et al., 2019)	19.1	16.1	-	54.3	21.4
GRN (Beck et al., 2018)	24.4	18.9	-	55.5	21.3
GCN (Marcheggiani and Perez-Beltrachini, 2018)	24.8	19.3	-	56.2	21.8
DeepGCN (Guo et al., 2019)	24.9	19.3	-	56.2	21.8
MGCN + CNN (Cheng et al., 2020)	26.4	20.4	-	57.4	24.2
FINETUNE‡	$32.39_{\pm 0.12}$	$30.39_{\pm 0.02}$	$53.87_{\pm 0.06}$	$56.27_{\pm 0.05}$	$42.35_{\pm 0.18}$
SA-RGCN‡	$34.06_{\pm 0.31}$	$31.54_{\pm 0.04}$	$57.78_{\pm 0.06}$	$58.42_{\pm 0.04}$	$43.32_{\pm 0.18}$
Ours	$\mathbf{34.87}_{\pm 0.36}$	$\mathbf{32.37}_{\pm 0.11}$	$\mathbf{58.41}_{\pm 0.22}$	$\mathbf{58.97}_{\pm 0.14}$	$\mathbf{43.70}_{\pm 0.12}$

Table 2: Main results of models on LDC2020T02 and ENT-DESC test datasets. ‡ means our reimplementation. The other results are copied from the original paper. Mean (±s.d.) over 4 seeds.

target text consists of sentences that verbalize the main entity in KG. ENT-DESC lacks explicit alignment between the input and the output. Therefore, some knowledge in the input graph may be noise. We follow official training, development, and test splits of 88,650/11,081/11,081 instances. Table 1 summarizes the detailed statistics of LDC2020T02 and ENT-DESC.

4.2 Settings

Our implementation is based on Hugging Face (Wolf et al., 2019). The RGCN and RGAT are implemented based on PyTorch Geometric (Fey and Lenssen, 2019). We initialize our models by T5 (Raffel et al., 2019). To make a fair comparision, we following the same experimental setting with SA-RGCN (Ribeiro et al., 2021). We set the hidden dimensions of both *Structural Adapter* and SACA to 256. And we use T5_{base} for all experiments on ENT-DESC and T5_{large} on LDC2020T02 for a fair comparison with baselines. We use the AdamW optimizer (Loshchilov and Hutter, 2018) and employ a linearly decreasing learning rate schedule without warm-up. The learning rate is fixed as 10^{-4}. We set the training batch size as 4 for all experiments. We freeze the T5 parameters and only update the newly added parameters during training. We tune the hyper-parameter λ in Equation 11 from the set $[1^{-2}, 5^{-3}, 1^{-3}, 5^{-4}]$, and select the best one on the development set. We stack $L = 2$ RGAT layers in Structure-Aware Cross-Attention. During decoding, we use beam search with a beam size 5. We use BLEU (Papineni et al., 2002) for the early stopping criterion. All experiments are trained on Nvidia Tesla V100 32GB GPUs.

Following previous works, on both datasets, we evaluate the results with BLEU (Papineni et al., 2002), METEOR (Denkowski and Lavie, 2011), and ChRF++ (Popović, 2015) on both datasets. On LDC2020T02, following Ribeiro et al. (2021), we utilize the meaning ($\mathcal{M}$) component of the $\mathcal{MF}$-score (Opitz and Frank, 2021) to measure how well the source AMR graph can be reconstructed from the generated sentence (refer to A.1 for more details). We use BERTScore (Zhang et al., 2020a) allowing a semantic evaluation that depends less on the surface forms. On ENT-DESC, We add ROUGE-L (Lin, 2004) and employ PARENT (Dhingra et al., 2019) for evaluating the faithfulness. We conduct experiments over 4 different seeds and report the average scores on them.

4.3 Main Results

We compare our method with recent state-of-the-art methods (please refer to A.4 for more details). Table 2 summarizes the results on LDC2020T02 and ENT-DESC test sets. FINETUNE is a method that transforms the input graph into a sequence and finetunes T5 directly. It does not consider the input graph structure. For LDC2020T02, our method outperforms the previous state-of-the-art model by 0.78 BLEU and 1.15 ChRF++. Com-

Models	BLEU	METEOR	$\mathcal{M}$	Dis-1	Dis-2
GOLD	-	-	81.00	23.82	71.76
ADAPT	45.22	43.28	79.56	23.20	71.40
Ours	**47.85**	**45.80**	**80.37**	23.46	71.75
w/o DGP	47.68	45.51	80.21	23.51	72.08
w/o SACA & DGP	47.20	45.05	80.01	23.38	71.69
w/o StrucAdapt	45.43	43.54	79.75	23.32	71.65

Table 3: Ablation study of models on LDC2020T02 development dataset. **GOLD** indicates the ground-truth sentences. Dis-1 and Dis-2 denote Distinct1 and Distinct2, respectively.

Models	# Additional Params (million)	Latency (s)
ADAPT	28.72 (3.3%)	1.41
SA-RGCN	37.80 (4.9%)	1.49
+ SACA	39.21 (5.0%)	1.54
+ SACA & DGP	41.31 (5.0%)	1.55

Table 4: Impact on parameter and speed.

pared with our implemented SA-RGCN, we improve 1.01 METEOR. Moreover, our method raises 0.38 $\mathcal{M}$, which indicates that it can generate more faithful sentences to the input graphs. The improvement on *BERTScore* shows that the sentence generated by our method is more similar to the ground truth on the semantic level. For ENT-DESC, we notice FINETUNE performs better than all previous methods. SA-RGCN, which encodes graph structure into T5, furtherly improves the performance. And our model exceeds all previous works and achieves new state-of-the-art results on all metrics. The above results indicate that our proposed methods can improve the model on fluency and faithfulness.

4.4 Analysis and Discussion

Ablation Study The overall performance on the two datasets shows the superiority of our proposed Structure-Aware Cross-Attention (SACA) and Dynamic Graph Pruning (DGP). To demonstrate the effectiveness of each component, we conduct ablation studies on LDC2020T02 development sets and minus one particular component at a time to understand its impact on the performance. Especially, **w/o DGP** denotes we remove the dynamic graph pruning module and the training objective L_{DGP}. **ADAPT** and **w/o StrucAdapt** denote replacing each structural adapter in SA-RGCN's and our encoders with an FNN adapter, respectively. **W/o StrucAdapt** means that the model only considers the structural information during decoding. The results are summarized in Table 3. Particularly, we observe the performance drops after removing SACA or DGP. This indicates that injecting the structural information into input graph context modeling (SACA) and dynamically removing the redundant nodes (DPG) are beneficial. Regarding the $\mathcal{M}$ score, our model and **ADAPT** are close to **GOLD**. The AMR parser utilized by $\mathcal{M}$, **ADAPT** as well as our method are all initialized by T5. And the AMR paring and AMR-to-Text are dual tasks actually. Therefore, the $\mathcal{M}$ score is biased and the results of our model and **ADAPT** are somehow inflated. Additionally, we utilize Distinct-1 and Distinct-2 (Li et al., 2016) to evaluate the diversity of the output text. We observe that SACA and DGP have little effect on Distinct-1 and Distinct-2. This implies that they will not reduce the diversity of the output text.

We notice that, compared with **ADAPT**, **w/o StrucAdapt** shows a slight improvement. This indicates it is necessary to explicitly model the graph structure in the encoder, even though structural bias has been injected into the input graph context modeling during decoding. We believe this may be attributed to SACA relying on the input graph representation encoded by the encoder. Because our SACA is designed to exploit the relevant-only information for prediction, it re-encodes the input graph by conditioning its representation on the newly generated context. Therefore, the initial representation for the input graph is important.

Impact on the parameter and speed Furthermore, we investigate the impact of SACA and DPG on the model parameters and inference speed on LDC2020T02 development. Specifically, we calculate the additional parameters of each model with respect to T5_{large}. And we set the batch size to 1 to calculate the average decoding time for generating all examples. The results summarized in Table 4 indicate that SACA and DGP only bring minor increase on the model size and inference time.

Impact on the Graph Properties To examine the robustness of our proposed methods, we investigate the model's performance concerning different graph properties (graph size, graph diameter, and reentrancies) on LDC2020T02 and ENT-DESC. Following previous works (Cheng et al., 2020; Ribeiro et al., 2021), we use BLEU as the metric. The results are summarized in Table 5 and Table 6, respectively. For LDC2020T02, we firstly note that the BLEU scores decrease as the graph size increases since the larger graph is often

Graph Size	1-30	31-60	>60
# Examples	840	678	380
SA-RGCN	54.10	44.89	46.12
Ours	54.55$_{+0.45}$	45.88$_{+0.99}$	46.72$_{+0.60}$
Graph Diameter	1-8	9-12	>12
# Examples	824	603	471
SA-RGCN	56.98	43.12	46.07
Ours	57.01$_{+0.03}$	43.59$_{+0.47}$	46.99$_{+0.92}$
Reentrancies	<= 1	2	>2
# Examples	913	549	436
SA-RGCN	53.60	44.03	43.30
Ours	54.16$_{+0.56}$	44.55$_{+0.52}$	44.53$_{+1.23}$

Table 5: BLEU scores with respect to graph size, graph diameter and number of reentrancies on LDC2020T02 test set.

Graph Size	1-20	21-40	>40
# Examples	3,559	5,069	2,453
SA-RGCN	33.01	38.86	28.54
Ours	33.67 $_{+0.66}$	39.44$_{+0.58}$	29.02 $_{+0.48}$
Graph Diameter	1-3	4-5	>5
# Examples	2,227	5,017	3,787
SA-RGCN	30.52	34.41	35.83
Ours	31.14$_{+0.62}$	34.83$_{+0.45}$	36.55$_{+0.72}$
Reentrancies	< 6	6-10	>10
# Examples	2,277	5,017	3,787
SA-RGCN	27.57	36.58	37.17
Ours	28.03 $_{+0.46}$	37.17 $_{+0.59}$	37.81$_{+0.64}$

Table 6: BLEU scores with respect to graph size, graph diameter and number of reentrancies on ENT-DEST test set.

Models	**BLEU**	**METEOR**	**ROUGE-L**
GraphWriter	14.30	18.80	-
GraphWriter‡	14.13 $\pm$ 0.10	18.92 $\pm$ 0.28	27.61 $\pm$ 0.16
Ours	**15.59** $\pm$0.35	**19.70** $\pm$0.21	**28.47** $\pm$0.14

Table 7: Generalization Study on AGENDA test dataset. ‡ means our reimplementation.

complex. Our method achieves a clear improvement when handling graphs with > 30 nodes. And then we observe that the BLEU gap between our method and SA-RGCN becomes larger for a relatively larger graph diameter. Reentrancies are the nodes with multiple parents. A graph with more reentrancies is typically more complex (Wang et al., 2020). As shown in the last section in Table 5, our method has an improvement of $+1.23$ BLEU points compared to SA-RGCN when graphs contain > 2 reentrancies. To sum up, the results on the LDC2020T02 dataset show the advantage of our model in dealing with the AMR graph with more complex structures.

As shown in Table 6, both models perform differently on ENT-DESC than on LDC2020T02. First, we notice that both models perform the best when the graph size is between 31 and 50, and they perform poorly when the graph size is too small or too large. Cheng et al. (2020) also observed the finding, and they believe this is due to the insufficient or very noisy input information for generation. Additionally, both models perform better when graph diameter or number of the reentrancies increase. The reason is that, in the ENT-DESC, the knowledge graph with a small diameter or number of the reentrancies contains more noisy information for the generation. Please refer to A.2 for more details. The BLEU gap between our method and SA-RGCN is the largest when the graph diameter > 5 or the number of reentrancies > 10. The above results demonstrate that our approach makes SA-RGCN better at handling complex knowledge graphs.

We investigate how the model behaves on different types of graphs (AMR and KG). And the results demonstrate that our model deals better with complex structures. We believe the improvement comes from two aspects. First, on the one hand, it is challenging for an encoder to encode all relevant information into node representations in a single forward pass, especially if the graph structure is complex. On the other hand, the re-encoding in SACA makes the decoder easily exploit the relevant-only information for prediction and explicitly injects the structural information at each decoding step. Second, DGP dynamically removes the nodes which are redundant for the subsequent generation, which makes the decoder pay more attention to the relevant nodes.

4.5 Generalization Study

Institutionally, our proposed methods can not only be applied to PLMs but also RNN based models. In other words, we can easily combine the SACA and DGP with previous RNN based works. To examine the generalization of SACA and DGP, we choose GraphWriter (Koncel-Kedziorski et al., 2019) as the baseline, which consists of a multi-layer graph transformer encoder and an attention-based decoder with a copy mechanism. Further, to make a fair comparison, we conduct the generalization experiment on AGENDA dataset (Koncel-Kedziorski et al., 2019). We simply replace the *plain* cross-attention in GraphWriter with our proposed SACA. Additionally, we add the DGP layer before the SACA. The experiments are under the same settings as described in GraphWriter. As

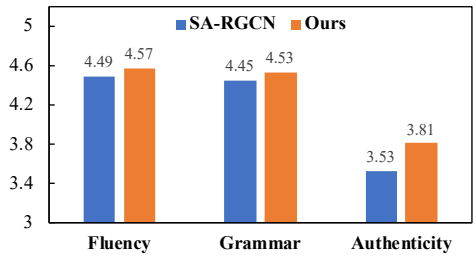

Figure 3: Human evaluation results on ENT-DESC test set.

shown in Table 7, we observe that our proposed model significantly improves the performance of GraphWriter. The results indicate that SACA and DGP are not only effective well on PLMs-based models but also potent for RNN-based models.

4.6 Human Evaluation

Considering that the knowledge graph is more readable than AMR, we do human evaluations on the ENT-DESC test set to examine whether human judgments corroborate improvements in automatic evaluation metrics. Following Cheng et al. (2020), from outputs generated by the baseline model SA-RGCN and our final model (Ours). We distribute the outputs of different systems to three annotators with linguistic backgrounds. The annotators have no knowledge in advance about which model the generated text comes from. Specifically, we give each participant all main entities' neighbors, 1-hop and 2-hop connections between main entities, and topic-related entities as references. They are required to score the generated text from 1 to 5 in terms of three criteria: Fluency (is the sentence fluent?), Grammar (is the sentence grammatical?), and Authenticity (is the sentence more related to the input graph?). For each criterion, we calculate the final score by averaging the scores from all annotators. As shown in Figure 3, our model outperforms the baseline SA-RGCN on Fluency and Grammar metrics. For Authenticity, the improvement is more significant. The performance validates the benefit of our proposed SACA and DGP modules in capturing more accurate input graph context representations. We supply a case study in A.3.

5 Conclusions

In this work, we make two main contributions. First, we propose Structure-Aware Cross-Attention (SACA) to make decoder easily exploit relevant-only information for prediction. Apart from the plain cross-attention, SACA re-encodes the input graph conditioning on the newly generated context while explicitly considering the input graph structure. The second one is that we adapt SACA and propose its variant Dynamic Graph Pruning (DGP) mechanism. In detail, the DGP dynamically prunes the structure of the joint graph at different decoding steps according to the generated text. Experimental results conducted on two graph-to-text datasets, LDC2020T02 and ENT-DESC, show the effectiveness of our method. The empirical and analysis results on both datasets show that the proposed methods can improve the model's performance on complex graphs while only bringing minor increase on the model size and inference time.

References

Dzmitry Bahdanau, Kyung Hyun Cho, and Yoshua Bengio. 2015. Neural machine translation by jointly learning to align and translate. In *Proc. of ICLR*.

Laura Banarescu, Claire Bonial, Shu Cai, Madalina Georgescu, Kira Griffitt, Ulf Hermjakob, Kevin Knight, Philipp Koehn, Martha Palmer, and Nathan Schneider. 2013. Abstract meaning representation for sembanking. In *Proceedings of the 7th linguistic annotation workshop and interoperability with discourse*.

Daniel Beck, Gholamreza Haffari, and Trevor Cohn. 2018. Graph-to-sequence learning using gated graph neural networks. In *ACL*.

Michele Bevilacqua, Rexhina Blloshmi, and Roberto Navigli. 2021. One SPRING to rule them both: Symmetric AMR semantic parsing and generation without a complex pipeline. In *Proc. of AAAI*.

Deng Cai and Wai Lam. 2020. Graph transformer for graph-to-sequence learning. In *Proc. of AAAI*.

Wenhu Chen, Yu Su, Xifeng Yan, and William Yang Wang. 2020. Kgpt: Knowledge-grounded pre-training for data-to-text generation.

Liying Cheng, Dekun Wu, Lidong Bing, Yan Zhang, Zhanming Jie, Wei Lu, and Luo Si. 2020. ENT-DESC: Entity description generation by exploring knowledge graph. In *EMNLP*.

Marco Damonte and Shay B Cohen. 2019. Structural neural encoders for amr-to-text generation. In *Proc. of AACL*.

Michael Denkowski and Alon Lavie. 2011. Meteor 1.3: Automatic metric for reliable optimization and evaluation of machine translation systems. In *Proceedings of the sixth workshop on statistical machine translation*.

Bhuwan Dhingra, Manaal Faruqui, Ankur P. Parikh, Ming-Wei Chang, Dipanjan Das, and William W. Cohen. 2019. Handling divergent reference texts when evaluating table-to-text generation. In *Proc. of ACL*.

Matthias Fey and Jan E. Lenssen. 2019. Fast graph representation learning with PyTorch Geometric. In *ICLR Workshop on Representation Learning on Graphs and Manifolds*.

Bin Fu, Yunqi Qiu, Chengguang Tang, Yang Li, Haiyang Yu, and Jian Sun. 2020. A survey on complex question answering over knowledge base: Recent advances and challenges. *arXiv preprint arXiv:2007.13069*.

Zhijiang Guo, Yan Zhang, Zhiyang Teng, and Wei Lu. 2019. Densely connected graph convolutional networks for graph-to-sequence learning. *Transactions of the Association for Computational Linguistics*.

Binyuan Hui, Ruiying Geng, Lihan Wang, Bowen Qin, Yanyang Li, Bowen Li, Jian Sun, and Yongbin Li. 2022. S^2SQL: Injecting syntax to question-schema interaction graph encoder for text-to-SQL parsers. In *Findings of the Association for Computational Linguistics: ACL 2022*, pages 1254–1262, Dublin, Ireland. Association for Computational Linguistics.

Jacob Devlin Ming-Wei Chang Kenton and Lee Kristina Toutanova. 2019. Bert: Pre-training of deep bidirectional transformers for language understanding. In *Proc. of AACL*.

Rik Koncel-Kedziorski, Dhanush Bekal, Yi Luan, Mirella Lapata, and Hannaneh Hajishirzi. 2019. Text generation from knowledge graphs with graph transformers. In *Proc. of AACL*.

Rémi Lebret, David Grangier, and Michael Auli. 2016. Neural text generation from structured data with application to the biography domain. In *EMNLP*.

Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Ves Stoyanov, and Luke Zettlemoyer. 2020. Bart: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension.

Jiwei Li, Michel Galley, Chris Brockett, Jianfeng Gao, and Bill Dolan. 2016. A diversity-promoting objective function for neural conversation models. In *Proc. of NAACL*.

Liang Li, Can Ma, Yinliang Yue, and Dayong Hu. 2021. Improving encoder by auxiliary supervision tasks for table-to-text generation. In *ACL*.

Wei Li, Xinyan Xiao, Jiachen Liu, Hua Wu, Haifeng Wang, and Junping Du. 2020. Leveraging graph to improve abstractive multi-document summarization. In *Proc. of ACL*.

Chin-Yew Lin. 2004. Rouge: A package for automatic evaluation of summaries. In *Text summarization branches out*.

Tianyu Liu, Fuli Luo, Qiaolin Xia, Shuming Ma, Baobao Chang, and Zhifang Sui. 2019. Hierarchical encoder with auxiliary supervision for neural table-to-text generation: Learning better representation for tables. In *Proc. of AAAI*.

Ilya Loshchilov and Frank Hutter. 2018. Decoupled weight decay regularization. In *ICLR*.

Diego Marcheggiani and Laura Perez-Beltrachini. 2018. Deep graph convolutional encoders for structured data to text generation. In *Proceedings of the 11th International Conference on Natural Language Generation*.

Juri Opitz and Anette Frank. 2021. Towards a decomposable metric for explainable evaluation of text generation from AMR. In *Proc. of EACL*.

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. Bleu: a method for automatic evaluation of machine translation. In *ACL*.

Jonas Pfeiffer, Ivan Vulić, Iryna Gurevych, and Sebastian Ruder. 2020. MAD-X: An Adapter-Based Framework for Multi-Task Cross-Lingual Transfer. In *EMNLP*.

Maja Popović. 2015. chrf: character n-gram f-score for automatic mt evaluation. In *Proceedings of the Tenth Workshop on Statistical Machine Translation*.

Bowen Qin, Binyuan Hui, Lihan Wang, Min Yang, Jinyang Li, Binhua Li, Ruiying Geng, Rongyu Cao, Jian Sun, Luo Si, et al. 2022. A survey on text-to-sql parsing: Concepts, methods, and future directions. *arXiv preprint arXiv:2208.13629*.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J Liu. 2019. Exploring the limits of transfer learning with a unified text-to-text transformer. *arXiv preprint arXiv:1910.10683*.

Leonardo F. R. Ribeiro, Yue Zhang, and Iryna Gurevych. 2021. Structural adapters in pretrained language models for AMR-to-Text generation. In *Proc. of EMNLP*.

Michael Sejr Schlichtkrull, Thomas N. Kipf, Peter Bloem, Rianne van den Berg, Ivan Titov, and Max Welling. 2018. Modeling relational data with graph convolutional networks. In *The Semantic Web - 15th International Conference, ESWC 2018, Heraklion, Crete, Greece, June 3-7, 2018, Proceedings*.

Martin Schmitt, Sahand Sharifzadeh, Volker Tresp, and Hinrich Schütze. 2020. An unsupervised joint system for text generation from knowledge graphs and semantic parsing. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 7117–7130, Online. Association for Computational Linguistics.

Zhihong Shao, Minlie Huang, Jiangtao Wen, Wenfei Xu, and Xiaoyan Zhu. 2019. Long and diverse text generation with planning-based hierarchical variational model. In *EMNLP-IJCNLP*.

Peter Shaw, Jakob Uszkoreit, and Ashish Vaswani. 2018. Self-attention with relative position representations. In *Proc. of NAACL*.

Yunzhou Shi, Zhiling Luo, Pengcheng Zhu, Feng Ji, Wei Zhou, Haiqing Chen, and Yujiu Yang. 2020. G2t: Generating fluent descriptions for knowledge graph. In *Proc. of SIGIR*.

Linfeng Song, Yue Zhang, Zhiguo Wang, and Daniel Gildea. 2018. A graph-to-sequence model for amr-to-text generation. In *ACL*.

Lihan Wang, Bowen Qin, Binyuan Hui, Bowen Li, Min Yang, Bailin Wang, Binhua Li, Jian Sun, Fei Huang, Luo Si, et al. 2022. Proton: Probing schema linking information from pre-trained language models for text-to-sql parsing. In *Proceedings of the 28th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, pages 1889–1898.

Tianming Wang, Xiaojun Wan, and Hanqi Jin. 2020. Amr-to-text generation with graph transformer. *Transactions of the Association for Computational Linguistics*.

Sam Wiseman, Stuart M. Shieber, and Alexander M. Rush. 2017. Challenges in data-to-document generation. In *Proc. of EMNLP*.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Rémi Louf, Morgan Funtowicz, et al. 2019. Huggingface's transformers: State-of-the-art natural language processing. *arXiv preprint arXiv:1910.03771*.

Lanqing Xue, Xiaopeng Li, and Nevin L Zhang. 2020. Not all attention is needed: Gated attention network for sequence data. In *Proc. of AAAI*.

Tianyi Zhang, Varsha Kishore, Felix Wu, Kilian Q. Weinberger, and Yoav Artzi. 2020a. Bertscore: Evaluating text generation with BERT. In *Proc. of ICLR*.

Yan Zhang, Zhijiang Guo, Zhiyang Teng, Wei Lu, Shay B Cohen, Zuozhu Liu, and Lidong Bing. 2020b. Lightweight, dynamic graph convolutional networks for amr-to-text generation. In *EMNLP*.

Jie Zhu, Junhui Li, Muhua Zhu, Longhua Qian, Min Zhang, and Guodong Zhou. 2019. Modeling graph structure in transformer for better amr-to-text generation. In *EMNLP-IJCNLP*.

A Appendix

A.1 $\mathcal{MF}$-score

The $\mathcal{M}$ (Meaning Preservation) component of the $\mathcal{MF}$-score (Opitz and Frank, 2021) is utilized to measure how well the source AMR graph can be reconstructed from the generated sentence. It reconstructs the AMR with a SOTA parser and computes the relative graph overlap of the reconstruction and the source AMR using graph matching. $\mathcal{M}$ employs the python library amrlib[2] (version 0.5.0) to make AMR parse, where the parser is a T5-based model.

A.2 Distribution on Graph Size

On the ENT-DESC test set, previous study (Cheng et al., 2020) and our experimental results (in Table 6) suggest that the model performs the best when the graph size lies in the range of $21-40$ and has a poorer performance when the number of triples is too small or too large. It should be due to the fact that the input information is insufficient or very noisy. However, we find that the model performance increases as the graph diameter and reentrancies increase. For further investigation, we calculate the distribution of graph diameter and reentrancies broken down by graph size, respectively. The results are summarized in Figure 4. As shown in Figure 4(a), the proportion of graphs with size $21-40$ increases as the graph diameter increases. As shown in Figure 4(b), the results on graph reentrancy follow a pattern similar to graph diameter. In a word, in ENT-DESC, the noise decreases as the graph diameter and reentrancies increase, so the model performs better.

A.3 Case Study

As shown in Figure 5, we further take a typical example from our human study to better understand how our method improves the mode's performance. Given the Knowledge Graph containing the main entity "Andrew Lawrence" and all its related entities, we aim to generate a description about the main entity. We notice that both the baseline and our model can identify the main entity. However, the baseline outputs a sentence describing the relation between "Andrew Lawrence" and "Matthew Lawrence". The relation is not existing in the input graph. Moreover, it repeatedly generates the entity "Brotherly Love" and misses the related entity "Recess". Compared with it, our model generates the sentences faithful to the input graph and correctly covers the main entity and most topic-related entities. We consider this is because the SACA helps the decoder obtain a more accurate input graph context, and the DGP removes the redundant nodes as the decoding stage progresses.

[2]https://github.com/bjascob/amrlib/tree/0.5.0

A.4 Baseline Models

On the AMR-to-Text task LDC2020T02, we compare our method with several baselines including:

- **LDGCN** (Zhang et al., 2020b) is a a dynamic fusion mechanism, which captures richer non-local interactions by synthesizing higher order information from the input graphs. A weight tied convolutions to reduce memory usage is applied.
- **SPRING** (Bevilacqua et al., 2021) casts Text-to-AMR and AMR-to-Text as a symmetric transduction task and proposes a graph linearization and extending a pretrained encoder-decoder model.

On the KG-to-Text task ENT-DESC, we compare our method with several baselines including:

- **s2s** (Bahdanau et al., 2015) is a encoder-decoder based model, which allows a model to automatically (soft-)search for parts of a source sentence that are relevant to predicting a target word, without having to form these parts as a hard segment explicitly.
- **GraphTransformer** (Koncel-Kedziorski et al., 2019) introduces a novel graph transforming encoder which can leverage the relational structure of such knowledge graphs without imposing linearization or hierarchical constraints.
- **GRN** (Beck et al., 2018) couples the recently proposed Gated Graph Neural Networks with an input transformation that allows nodes and edges to have their own hidden representations.
- **GCN** (Marcheggiani and Perez-Beltrachini, 2018) proposes an alternative encoder based on graph convolutional networks that directly exploits the input structure.
- **DeepGCN** (Guo et al., 2019) introduces a dense connection strategy, which is able to integrate both local and non-local features to

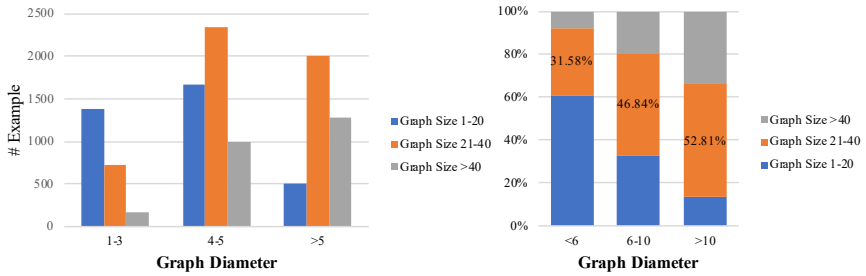

(a) The distribution of graph diameter by graph size.

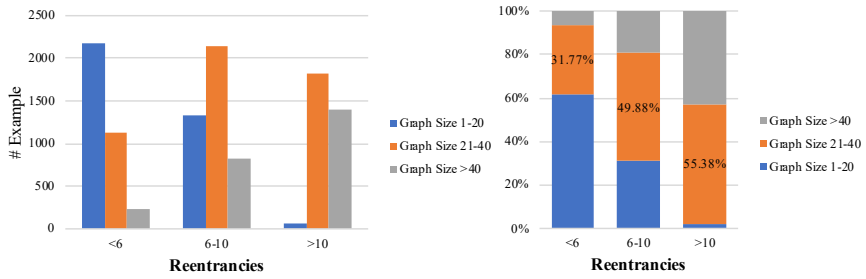

(b) The distribution of graph reentrancies by graph size.

Figure 4: The clustered column charts of graph diameter and reentrancies by graph size.

Gold	Andrew James Lawrence (born January 12, 1988) is an American actor and singer. He is known for his roles as Andy Roman in "Brotherly Love" and T.J. Detweiler in "Recess".
SA-RGAT	Andrew Lawrence (born January 12, 1988) is an American actor, voice actor, and singer. He is best known for his roles in the films "Brotherly Love" and "Brotherly Love". He is the younger brother of Matthew Lawrence.
Ours	Andrew Lawrence (born January 12, 1988) is an American actor and singer. He is best known for his roles in "Recess" and "Brotherly Love".

Figure 5: An example of generated sentences. The main entity is highlighted in red, topic-related entities are highlighted in blue, and the sentence that is not faithful to the input graph is in green.

learn a better structural representation of a graph.

- **MGCN + CNN** (Cheng et al., 2020) is a multi-graph structure that is able to represent the original graph information more comprehensively. We do not report the results of MGCN + CNN + delex. Because it applies the delexicalization technique on the ENT-DESC dataset, which delexicalizes the main entity and topic-related entities by replacing these entities with tokens indicating the entity types and indices. The delexicalization technique greatly boosts their performance on ROUGE-L. They do not release the code about delexicalization, and we can not reproduce it.

What's more, FINETUNE, ADAPT and SA-RGCN are T5-based models proposed in (Ribeiro et al., 2021).

Multi-Perspective Document Revision

Mana Ihori and **Hiroshi Sato** and **Tomohiro Tanaka** and **Ryo Masumura**
NTT Computer and Data Science Laboratories, NTT Corporation
1-1 Hikarinooka, Yokosuka-Shi, Kanagawa 239-0847, Japan
mana.ihori.kx@hco.ntt.co.jp

Abstract

This paper presents a novel multi-perspective document revision task. In conventional studies on document revision, tasks such as grammatical error correction, sentence reordering, and discourse relation classification have been performed individually; however, these tasks simultaneously should be revised to improve the readability and clarity of a whole document. Thus, our study defines multi-perspective document revision as a task that simultaneously revises multiple perspectives. To model the task, we design a novel Japanese multi-perspective document revision dataset that simultaneously handles seven perspectives to improve the readability and clarity of a document. Although a large amount of data that simultaneously handles multiple perspectives is needed to model multi-perspective document revision elaborately, it is difficult to prepare such a large amount of this data. Therefore, our study offers a multi-perspective document revision modeling method that can use a limited amount of matched data (i.e., data for the multi-perspective document revision task) and external partially-matched data (e.g., data for the grammatical error correction task). Experiments using our created dataset demonstrate the effectiveness of using multiple partially-matched datasets to model the multi-perspective document revision task.

1 Introduction

With the advance of natural language processing technology using deep learning, applications for writing support systems have been developed (Tsai et al., 2020; Ito et al., 2020). Such systems often implement a grammatical error correction task that corrects errors such as typos and mistakes in inflected verb forms (Rothe et al., 2021). It is easy for the reader to understand an error-free document, and the lack of errors can allow for smooth text communication. In addition, it is crucial to revise a document automatically because it is difficult to read one's writing objectively, and it is time-consuming for a third party to revise the document.

The document revision task has been studied in the natural language processing field by being broken down into partial tasks. Grammatical error correction tasks have been studied actively (Sawai et al., 2013; Mizumoto and Matsumoto, 2016; Junczys-Dowmunt and Grundkiewicz, 2016), and modeling with the sequence-to-sequence (seq2seq) model has achieved high performance with the advance of deep learning (Yuan and Briscoe, 2016; Junczys-Dowmunt et al., 2018; Rothe et al., 2021). In addition, tasks considering the relationship between sentences in a document are sentence ordering (Yin et al., 2019; Wang and Wan, 2019) and discourse relation classification (Liu et al., 2016; Dai and Huang, 2018). These tasks have achieved high performance using deep learning, as with the grammatical error correction task.

However, studies that simultaneously revise multiple perspectives have not been well considered. To advance writing support, it is important not only to correct grammatical errors in a single sentence but also to improve the readability and clarity of a whole document. For example, when we manually perform document revision, we attempt to correct grammatical errors, split a long sentence into shorter sentences, and consider the relationships between sentences, such as by reordering them to obtain a consistent order and by performing conjunction insertion. Accordingly, this paper addresses a novel multi-perspective document revision task that simultaneously considers various perspectives such as grammatical error correction, long sentence splitting, sentence reordering, and conjunction insertion, as shown in Figure 1.

To address the multi-perspective document revision task, we need to define this task and design a suitable dataset. In this paper, we define

Proceedings of the 29th International Conference on Computational Linguistics, pages 6128–6138
October 12–17, 2022.

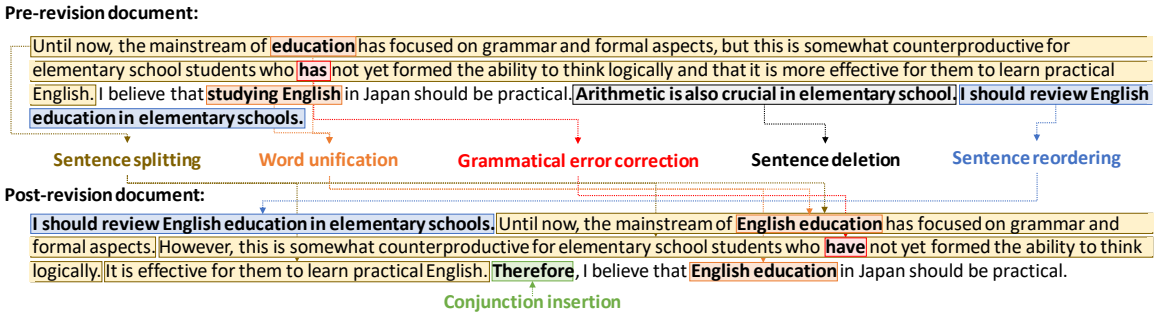

Figure 1: Example of multi-perspective document revision task.

multi-perspective document revision as revising a document to improve its readability and clarity by considering multiple perspectives simultaneously. However, there is no dataset that simultaneously handles multiple perspectives because datasets in conventional studies handled only one perspective (Dahlmeier et al., 2013; Tanaka et al., 2020; Chen et al., 2016; Webber et al., 2019). Thus, we make a new Japanese multi-perspective document revision dataset that can simultaneously handle seven perspectives of readability and clarity of documents. To the best of our knowledge, this is the first attempt to construct a multi-perspective document revision dataset. This paper details how we constructed our dataset (see section 3).

To model the multi-perspective document revision task, it is necessary to consider long-range contexts of multiple sentences because some perspectives address the relationships between sentences. For example, sentence reordering and conjunction insertion tasks cannot be revised without considering the relationship between multiple sentences. Thus, we use the seq2seq model for modeling the multi-perspective document revision task by considering document-level information. Although our study uses the seq2seq model as with the grammatical error correction task, one difference is that we handle not a single sentence but a set of sentences.

The main difficulty in modeling the multi-perspective document revision task is simultaneously considering multiple perspectives. To address the multiple perspectives simultaneously and robustly, we should prepare a large amount of matched data (i.e., data for the multi-perspective document revision task). However, it is difficult to prepare such a large amount of data because manually writing and revising documents is time-consuming. Thus, we use a limited amount of matched data and a large amount of partially-matched data that handles individual partial tasks to model a multi-perspective document revision task because this task is composed of multiple partial tasks. Preparing a large amount of partially-matched data is easy because some datasets exist, and others can be generated heuristically (e.g., for the conjunction insertion task, we can construct paired data by deleting and restoring conjunctions from existing documents). To effectively model the multi-perspective document revision task using both a matched dataset and multiple partially-matched datasets, we use seq2seq modeling with switching tokens (Ihori et al., 2021b). In our study, the switching tokens are used for distinguishing individual partial tasks in our multi-perspective document revision task. For example, when the grammatical error correction dataset is trained, we can use this dataset as the partially-matched dataset by switching the grammatical error correction task "on" and other tasks "off". Although the method using switching tokens is not new, applying the switching tokens for the task that can improve the performance of a matched task from multiple partially-matched tasks is new.

In our experiments using our created dataset, we use the grammatical error correction dataset and the conjunction insertion dataset as partially-matched datasets. Our results demonstrate that our modeling method can simultaneously revise multiple perspectives and effectively improve performance by using a matched dataset and multiple partially-matched datasets. Our main contributions are as follows:

- We define a novel multi-perspective document revision task that simultaneously considers multiple perspectives for writing support.
- We create a novel Japanese multi-perspective document revision dataset that can simultaneously handle seven perspectives and detail how we make it.

- We present a multi-perspective document revision modeling method that takes advantage of the fact that this task is composed of multiple partial tasks and uses both a matched dataset and multiple partially-matched datasets.

2 Related Work

2.1 Modeling of partial task in document revision task

The partial tasks that compose a document revision task have been studied as individual tasks. First, grammatical error correction is the most typical task, and it corrects the errors in input text by deleting, inserting, and replacing words. For this task, studies have focused on sentence-level errors and performed error correction by using a seq2seq model to achieve high-performance (Yuan and Briscoe, 2016; Junczys-Dowmunt et al., 2018; Rothe et al., 2021). Also, synthetic training data generation is introduced to deal with paired-data scarcity in recently (Grundkiewicz et al., 2019; Kiyono et al., 2020; Rothe et al., 2021). However, it is difficult to generate synthetic data for the multi-perspective document revision task because it involves multiple partial tasks such as grammatical error correction, sentence reordering, and conjunction insertion. Next, there are the tasks that handle a set of sentences like the discourse relation classification task (Liu et al., 2016; Dai and Huang, 2018). This task predicts the relation class (e.g., contrast and causality) of two arguments and can help in writing coherent text by suggesting relationships between sentences. Our study adopts a conjunction insertion task similar to the discourse relation classification task but directly completes conjunctions in accordance with the relationship between sentences.

2.2 Modeling of multiple perspectives simultaneously

There are few studies to perform multiple perspectives using the seq2seq model. Lin et al. (2021) proposed document-level paraphrase generation task that simultaneously performs the sentence reordering and sentence paraphrasing tasks. In this conventional study, a pseudo dataset for a document-level paraphrase generation task was created, and the task was performed with a task-specific model architecture. Ihori et al. (2021b) proposed the method to perform disfluency deletion and punctuation restoration tasks simultaneously. To execute these two tasks simultaneously without preparing a matched dataset, switching tokens have been introduced into the seq2seq model. These conventional studies handle limited tasks in the document revision task, and this paper is the first study to consider more perspectives simultaneously than these studies.

(1)	Correct the following mistakes. erroneous substitution, deletion, insertion, and kanji-conversion, syntactic errors, redundant expressions, style normalization, punctuation errors
(2)	Split long sentences containing more than 60 characters.
(3)	Unify words with different expressions that have the same meaning.
(4)	If there is no subject, restore the subject by using words that have already been mentioned.
(5)	Change the sentence order if it is not appropriate.
(6)	Delete sentences that describe unrelated topics.
(7)	Insert correct conjunctions by considering the relationships between sentences.

Table 1: Perspectives for Japanese multi-perspective document revision.

3 Japanese Multi-perspective Document Revision Dataset

This section details a new dataset for a Japanese multi-perspective document revision task. The dataset contains paired data consisting of source and revised documents in Japanese. The source documents were written by Japanese crowd workers, and the reference documents were revised by two Japanese labelers. To revise documents, we defined seven perspectives that have been individually used in general document revision problems. Table 1 summarizes all perspectives, and Figure 2 shows an example of multi-perspective document revision that simultaneously uses several perspectives, correction of erroneous insertion and punctuation error, splitting of sentences, and conjunction insertion. To the best of our knowledge, this is the first dataset to address such multiple perspectives of the document revision task.

3.1 Perspectives

(1) Error correction This perspective includes the grammatical error correction task (Tanaka et al., 2020). Mistakes in a document need to be corrected because it is difficult to understand the document with errors. In this paper, we define eight Japanese-specific errors, erroneous substitution, deletion, insertion, and kanji-conversion, syntactic errors, re-

Source document	ソーシャルメディアの発達**により**、私たちは好きな情報を簡単に得られるようになった。その一方で、日々膨大な情報にさらされることの弊害も出てきている。ソーシャルメディアでは、自身の趣味嗜好に**あ**沿った情報を得ようとすることが多い。これまでは新聞やテレビが情報源だったので全員が同じ情報に触れて対等に対話していた**が、**現在は無意識に自分に近しい意見を全体の情報として捉えてしまう人もおり、情報が偏る。ソーシャルメディアは一見情報の宝庫に見えるが、視点を変えれば物事を都合よく**とらえる**為のツールなのかもしれない。
Revised document	ソーシャルメディアの発達**で**、私たちは好きな情報を簡単に得られるようになった。その一方で、日々膨大な情報にさらされる弊害も出てきている。**また、**ソーシャルメディアでは、自身の趣味嗜好に沿った情報を得ようとすることが多い。これまでは新聞やテレビが情報源だったので**、**全員が同じ情報に触れて対等に対話していた**。しかし、**現在は、無意識に自分に近しい意見を全体の情報として捉える人もおり、情報が偏る。**そのため、**ソーシャルメディアは一見情報の宝庫に見えるが、視点を変えれば、物事を都合よく**捉える**為のツールなのかもしれない。
Translation	The development of social media has made it easier to get information. On the other hand, there can be difficulties in handling vast amounts of information. Also, in most cases, we only use social media to access our favorite types and sources of information. Previously, many people got the same information from newspapers and television, and thus, they could talk on an equal footing. Now, however, some people unknowingly treat their closely held opinions as complete information, so their information is biased. Therefore, social media seems to be a treasure trove of information, but it may also be a tool for maintaining biased information.

Figure 2: Example of Japanese multi-perspective document revision dataset. The colors of characters correspond to perspectives in Figure 1.

dundant expressions, text style normalization, and punctuation errors. For example, in revising redundant expressions, we remove expressions with the same meaning or words that make sense without them (e.g., 一番最後→最後 # last, することができる→できる # can).

(2) Sentence splitting Sentences more extended than 60 characters, which is the length of a typical sentence in Japanese, should be divided because long sentences decrease readability. We defined this by using specific numerical values to prevent different labelers from having other divisions.

(3) Word unification To avoid confusion for the reader, words that have the same meaning in a document should be expressed in the same way. In addition, in Japanese documents, expressions "desu, masu" or "da, dearu," are used at the end of sentences, and these expressions need to be unified within the same document.

(4) Subject restoration Throughout Japanese documents, the subject is often omitted; however, subject restoration may be necessary because a sentence without a subject may not correctly convey the intent of the writer to the reader. Thus, we restore the subject in sentences without a subject. Note that subject restoration should not be performed when the subject is clearly recognizable because consecutive occurrences of the same subject may reduce readability.

(5) Sentence reordering This perspective includes the sentence ordering task (Barzilay and Lapata, 2008). A well-organized document with a logical structure is much easier for people to read and understand. Also, sentence order is important in constructing logical structures. Thus, we reorder sentences, in order for a document to have a logical order.

(6) Sentences deletion A coherent document consists of multiple sentences describing a single topic, but a document that mixes multiple topics can inhibit the understanding of the reader. Thus, we delete sentences that describe distinctly different topics from a document.

(7) Conjunction insertion This perspective includes the discourse relation classification task (Liu et al., 2016). Discourse relations support a set of sentences to form a coherent document. Also, the conjunction has a role in showing these relations and serves as a natural means to connect sentences. Thus, we restore the conjunction in documents to improve readability. We created a list of conjunctions that show their kinds and roles, and we asked labelers to select from this list.

3.2 Dataset specifications

Source documents: To make the source documents, we hired 161 Japanese workers through a crowdsourcing service and asked them to write paragraph-level documents in Japanese. The documents had an essay-style structure because Japanese schools teach how to write essays; thus, we expected that many workers could write essays at the same level. First, we showed the workers 48 themes, and they each selected 1-15 themes. The 48 themes were chosen by the crowdsourcing company from actual themes that were used for exam essays in Japan. Next, the workers wrote paragraph-level documents, each of which contained 200-300 characters and four or more sentences. We could

		# of documents	# of sentences
Training	Input	5,000	26,477
	Output	5,000	28,158
Validation	Input	554	2,922
	Output	554	3,128
Test	Input	1,121	6,054
	Output	2,242	12, 831

Table 2: Details of multi-perspective document revision dataset.

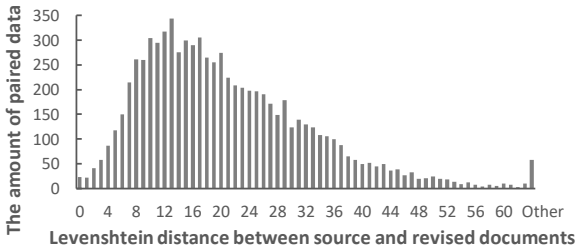

Figure 3: Distribution of Levenshtein distance for paired data in multi-perspective document revision dataset.

revise these source documents with multiple perspectives including the relationship between sentences by using this source document because they consisted of multiple sentences and had a coherent topic. Each worker wrote 1-15 documents, and the time limit for writing each one was 15 minutes. Although we asked workers to be careful about typos, we also asked them not to strive for perfection.

Revised documents: To revise the source documents, we hired two Japanese labelers. One labeler had a license as a Japanese language teacher, while the other labeler received guidance on revising the document. For the multi-perspective document revision task, we should simultaneously handle multiple perspectives to improve the readability and clarity of a document. Thus, we asked them to follow the revision guidelines listed in Table 1 to ensure that they could consider revising from multiple perspectives. We expected that the labelers would be able to revise documents with equivalent quality by following the guidelines. Note that they did not necessarily have to consider all perspectives simultaneously but were only to make these revisions if there were any mistakes or unnatural points. The collected data was divided into a training set, a validation set, and a test set. Table 2 details the resulting dataset for the Japanese multi-perspective document revision task.

3.3 Analysis

We investigate how much of the source documents were revised. To investigate the revision, we employ and measure Levenshtein distance (Levenshtein et al., 1966), which can measure the edit distance between the source and revised documents. Figure 3 shows the distribution of Levenshtein distance for all paired data in our created multi-perspective document revision dataset. In Figure 3, the Levenshtein distance of 17 was the most common, and the paired data with the distance were revised by correcting punctuation errors and inserting one or two conjunctions. In addition, as the Levenshtein distance increased, many perspectives were corrected simultaneously, such as reordering sentences, restoring subjects, and splitting sentences. However, the distribution of the various perspectives is unbalanced because this dataset contains more conjunction insertion and error correction than other perspectives.

4 Multi-perspective Document Revision Models

4.1 Strategy

To build a multi-perspective document revision model, we use the matched dataset created in section 3 and multiple partially-matched datasets that handle the partial tasks of the multi-perspective document revision task. Our strategy in using these datasets jointly is to incorporate multiple "on-off" switches into the seq2seq model. These switches can be implemented by using switching tokens (Ihori et al., 2021b). A switching token represents the "on" state (the target task) or "off" state (not the target task) for each task. In addition, a model that introduces switching tokens can explicitly distinguish the multi-perspective document revision task and each partial task.

Figure 4 shows an example of training and decoding the multi-perspective document revision model using switching tokens. In the figure, we use three datasets, a matched dataset, a grammatical error correction (gec) dataset, and a conjunction insertion (ci) dataset, to train the multi-perspective document revision model. In addition, we use three switching tokens, $s_1 \in$ {[gec_on], [gec_off]}, $s_2 \in$ {[ci_on], [ci_off]}, and $s_3 \in$ {[other_on], [other_off]}. We specify the "other" task because the multi-perspective document revision task handles other perspectives in addition to the grammatical error correction and conjunction insertion tasks, as listed in Table 1. These switching tokens are used as inputs of the decoder network in given

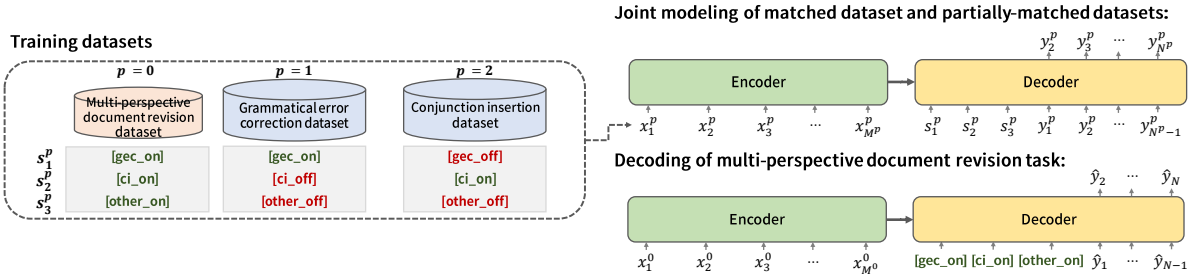

Figure 4: Example of training and decoding multi-perspective document revision model using switching tokens.

contexts. In the training phase, all three datasets are trained jointly by distinguishing each task with three switching tokens. In the decoding phase, the model performs the multi-perspective document revision task by feeding switching tokens, [gec_on], [ci_on], and [other_on]. Note that we can also perform the grammatical error correction or conjunction insertion tasks by feeding appropriate switching tokens.

4.2 Modeling method

We define a source document as $\boldsymbol{X} = \{x_1, \cdots, x_m, \cdots, x_M\}$ and a revised document as $\boldsymbol{Y} = \{y_1, \cdots, y_n, \cdots, y_N\}$, where M and N are the amount of tokens in source and revised documents, respectively. x_m and y_n are tokens that include not only characters or words but also punctuation marks. In this paper, we handle a set of sentences, so $\boldsymbol{X}$ and $\boldsymbol{Y}$ have multiple sentences. Thus, we introduce [CLS] token at the beginning of sentences into all datasets to distinguish each sentence in documents.

The multi-perspective document revision model predicts the generation probabilities of a revised document $\boldsymbol{Y}$ given a source document $\boldsymbol{X}$ and switching tokens $s_{1:T} = \{s_1, \cdots, s_t, \cdots, s_T\}$, where T is the total number of partial tasks that are included in each partially-matched dataset. The generation probability of $\boldsymbol{Y}$ is defined as

$$P(\boldsymbol{Y}|\boldsymbol{X}, s_{1:T}; \boldsymbol{\Theta}) = \prod_{n=1}^{N} P(y_n|y_{1:n-1}, \boldsymbol{X}, s_{1:T}; \boldsymbol{\Theta}), \quad (1)$$

where $y_{1:n-1} = \{y_1, \cdots, y_{n-1}\}$, and $\boldsymbol{\Theta}$ represents the trainable parameters. s_t is the t-th switching token represented as

$$s_t \in \{[t\text{−th } \texttt{task\_on}], [t\text{−th } \texttt{task\_off}]\}. \quad (2)$$

In this paper, we use Transformer pointer-generator networks (Deaton, 2019) for this modeling. These networks are effective for monolingual translation tasks because they contain a copy mechanism that copies tokens from a source text to help generate infrequent tokens. Note that our method does not change the architecture of a transformer pointer-generator network but merely adds switching tokens to the model input.

Pre-training: In this paper, we use a MAsked Pointer-Generator Network (MAPGN) (Ihori et al., 2021a) as self-supervised pre-training for the seq2seq model because it is a suitable pre-training method for pointer-generator networks. In MAPGN, the pointer-generator network is pre-trained by predicting a sentence fragment $y_{a:b}$ given a masked sequence $\boldsymbol{Y}_{/a:b}$. Here, $\boldsymbol{Y}_{/a:b}$ denotes a fragment in which positions a to b are masked, and $y_{a:b}$ denotes a sentence fragment of $\boldsymbol{Y}$ from a to b. The model parameter set can be optimized from unpaired dataset $\mathcal{D}_{\mathrm{u}}$ that is consisted of a set of sentences. The training loss function $\mathcal{L}$ is defined as

$$\begin{aligned}\mathcal{L} &= -\sum_{\boldsymbol{Y} \in \mathcal{D}_{\mathrm{u}}} \log P(y_{a:b}|y_{a-1}, \boldsymbol{Y}_{/a:b}; \boldsymbol{\Theta}), \quad (3)\\ &= -\sum_{\boldsymbol{Y} \in \mathcal{D}_{\mathrm{u}}} \sum_{t=a}^{b} \log P(y_t|y_{a-1:t-1}, \boldsymbol{Y}_{/a:b}; \boldsymbol{\Theta}).\end{aligned}$$

Note that all switching tokens have to be included in the vocabulary during pre-training.

Fine-tuning: During fine-tuning, the matched dataset $\mathcal{D}_{\mathtt{m}}$, and multiple partially-matched datasets $\{\mathcal{D}^1_{\mathtt{pm}}, \cdots, \mathcal{D}^p_{\mathtt{pm}}, \cdots, \mathcal{D}^P_{\mathtt{pm}}\}$ are trained jointly in a single model. P is the number of partially-matched datasets. The training loss function $\mathcal{L}$ is defined as

$$\mathcal{L} = \mathcal{L}_{\mathtt{m}} + \sum_{p=1}^{P} \mathcal{L}^p_{\mathtt{pm}}, \quad (4)$$

where $\mathcal{L}_{\mathtt{m}}$ is the loss function against the main task and it is computed from

$$\mathcal{L}_{\mathtt{m}} = -\sum_{(\boldsymbol{X}^0, \boldsymbol{Y}^0) \in \mathcal{D}_{\mathrm{m}}} \log P(\boldsymbol{Y}^0 | \boldsymbol{X}^0, s^0_{1:T}; \boldsymbol{\Theta}), \tag{5}$$

where $s^0_{1:T} = \{s^0_1, \cdots, s^0_T\}$ are switching tokens and s^0_t is represented as

$$s^0_t = [t\text{−th } \texttt{task\_on}]. \tag{6}$$

$\mathcal{L}^p_{\mathtt{pm}}$ is the loss function against the p-th partially-matched dataset and it is computed from

$$\mathcal{L}^p_{\mathtt{pm}} = -\sum_{(\boldsymbol{X}^p, \boldsymbol{Y}^p) \in \mathcal{D}^p_{\mathtt{pm}}} \log P(\boldsymbol{Y}^p | \boldsymbol{X}^p, s^p_{1:T}; \boldsymbol{\Theta}), \tag{7}$$

where $s^p_{1:T} = \{s^p_1, \cdots, s^p_T\}$ are switching tokens and s^p_t is represented as

$$s^p_t = \begin{cases} [t\text{−th } \texttt{task\_on}] & \text{if } t\text{-th task in } \mathcal{D}^p_{\mathtt{pm}}, \\ [t\text{−th } \texttt{task\_off}] & \text{otherwise.} \end{cases} \tag{8}$$

Decoding: The decoding problem using switching tokens is defined as

$$\hat{\boldsymbol{Y}} = \arg\max_{\boldsymbol{Y}} P(\boldsymbol{Y} | \boldsymbol{X}, s_{1:T}; \boldsymbol{\Theta}). \tag{9}$$

The model can perform the multi-perspective document revision task or each partial task in accordance with the given switching tokens.

5 Experiments

We experimentally evaluated the effectiveness of this modeling method that can use both a matched dataset and multiple partially-matched datasets.

5.1 Dataset

In this paper, we use two partial tasks, the grammatical error correction (gec) and conjunction insertion (ci) tasks, to build a multi-perspective document revision model. Accordingly, we use three datasets, a multi-perspective document revision dataset described in section 3, a Japanese grammatical error correction dataset (Tanaka et al., 2020), and a conjunction insertion dataset. We made the conjunction insertion dataset by deleting and restoring conjunctions from the Japanese Wiki-40b dataset (Guo et al., 2020), which is a high-quality processed Wikipedia dataset. To make this dataset from paragraph-level documents, we divided the dataset into paragraphs and extracted the documents that contained conjunctions. In addition, we use unpaired 880k paragraph-level documents for self-supervised pre-training, and these documents, which were prepared from the Wiki-40B dataset, were not used in the conjunction insertion dataset. The details of these datasets are listed in Table 3, where "switch" refers to switching tokens. For training and decoding, we use three switching tokens for the gec task, the ci task, and other tasks. The multi-perspective document revision model can also perform the gec and ci tasks by feeding appropriate switching tokens, so we evaluate the performance of each partial task using each test set. Note that the Japanese grammatical error correction dataset does not have documents because this dataset consists of single sentences.

		# of documents	# of sentences
Training	a	5,000	26,477
	b	-	506,786
	c	90,000	533,422
Validation	a	554	2,922
	b	-	8,542
	c	10,000	59,396
Test	a	1,121	6,054
	b	-	8,542
	c	1,000	6,026
switch	a	[gec_on][ci_on][other_on]	
	b	[gec_on][ci_off][other_off]	
	c	[gec_off][ci_on][other_off]	

a. Multi-perspective document revision dataset
b. Japanese grammatical error correction dataset
c. Conjunction insertion dataset

Table 3: Details of a matched dataset and two partially-matched datasets.

5.2 Setup

For evaluation purposes, we constructed 11 Transformer-based pointer-generator networks. We use the three datasets combined in different ways, each dataset only, a matched dataset and each partially-matched dataset, and the three datasets jointly. Then, we construct the models with and without switching tokens. In addition, we apply the pre-training to the models that use a matched dataset only and the three datasets jointly with switching tokens.

We used the following configurations. The encoder and decoder had a 4-layer and 2-layer transformer block with 512 units. The output unit size (corresponding to the number of tokens in the pre-training data) was set to 12,773. To train the

		Dataset	GLEU	$F_{0.5}$	C-F1
Source		-	0.886	0	0
Baseline	(1)	a	0.857	0.198	0.193
	(2)	+ PT	0.884	0.321	0.211
+ Datasets	(3)	a + b	0.863	0.189	0.164
	(4)	a + c	0.881	0.155	0.101
	(5)	a + b + c	0.883	0.236	0.205
+ Switch	(6)	a + b	0.887	0.278	0.163
	(7)	a + c	0.888	0.234	0.214
	(8)	a + b + c	0.889	0.282	0.270
	(9)	+ PT	**0.892**	**0.333**	**0.274**

Table 4: Results of multi-perspective document revision task. "Source" row indicates results for source documents in multi-perspective document revision task dataset.

Task	Dataset	GLEU	$F_{0.5}$	C-F1
gec	b	**0.943**	**0.635**	-
	a + b + c	0.932	0.613	-
	+ switch	**0.943**	0.630	-
ci	c	0.964	0.198	0.230
	a + b + c	0.966	0.207	0.222
	+ switch	**0.967**	**0.239**	**0.263**

Table 5: Results of grammatical error correction (gec) and conjunction insertion (ci) tasks.

Transformer pointer-generator networks, we used the RAdam optimizer (Liu et al., 2019) and label smoothing (Lukasik et al., 2020) with a smoothing parameter of 0.1. We set the mini-batch size to 32 documents and the dropout rate in each Transformer block to 0.1. All trainable parameters were initialized randomly, and we used characters as tokens. In the pre-training, we set the number of masked tokens to roughly 50% of an input sequence. For decoding, we used the beam search algorithm with a beam size of 4.

5.3 Results

Tables 4 and 5 show the results for the 11 Transformer pointer-generator networks. In these tables, a, b, and c represent the multi-perspective document revision dataset, the Japanese grammatical error correction dataset, and the conjunction insertion dataset. Also, "switch" and "PT" indicate switching tokens and pre-training, respectively. We used automatic evaluation scores in terms of two metrics: GLEU (Napoles et al., 2015) and $F_{0.5}$. Specifically, we calculated these metrics for characters in generated documents and used 4-grams for GLEU. In addition, we also calculated the F1 score for conjunction insertion, denoted as C-F1, to evaluate the performance of the conjunction insertion task. We evaluated whether the system could insert conjunctions with the correct meaning because multiple conjunctions have the same sense (e.g., "but" and "however").

Results of multi-perspective document revision task: In Table 4, when the number of partially-matched datasets increased, the performance improved. This indicated that switching-token-based joint modeling that trains a matched dataset and multiple partially-matched datasets using switching tokens is effective for modeling the multi-perspective document revision task. Therefore, it is important for switching-token-based joint modeling to distinguish tasks because the scores with switching tokens were higher than those without switching tokens. In addition, when we compared the results of lines (8) with (9), the results with pre-training outperformed those without pre-training. This indicates that switching-token-based joint modeling can be effectively applied after performing self-supervised pre-training.

Figure 5 shows that the generation examples of lines (2), (5), and (8) in Table 4. In the figure, the generation example in (2) shows that conversion errors were decreased, but task-specific generation, like a conjunction insertion, was not performed well. On the other hand, the generation example in (8) shows that task-specific generation performed better than (2). Thus, we suppose it is difficult to improve the performance of the multi-perspective document revision task by applying pre-training alone. Here, the generation example in (5) shows it has more errors than (8) in a task-specific generation. These facts indicate that the switching tokens are important for joint training of the matched and multiple partially-matched tasks.

Results of partial tasks: Table 5 shows that in grammatical error correction, the performance of individual modeling and switching-token-based joint modeling were not significantly different. However, the performance of joint modeling without switching tokens under-performed that of the individual modeling. For the conjunction insertion task, the results of joint modeling outperformed those of individual modeling. Also, the results of joint modeling with switching tokens outperformed those without switching tokens. Therefore, these results indicated that switching-token-based joint modeling could improve the performance of the multi-perspective document revision task without impairing the performance of each task. Note that this study aims to improve the performance

Input	i. …相手理解する (I understand you)，…つながらない (disconnect)，…成功をおさめる (achieve success) ii. …現代社会において、…「共依存」の関係が問題となっている。古代において敵が多い環境では、自分の命は自分で守る…大人になった証でもあった。誰かに依存することは多くの人の命を脅かす危険があったのである。 (… In today's society, … "co-dependent relationships" have become a problem. In ancient times and in an environment full of enemies, one's life was one's own … a sign of adulthood. To be dependent on someone else was to risk the lives of many.)
Reference	i. …相手を理解する，…繋がらない，…成功を収める ii. …現代社会において、…「共依存」の関係が問題となっている。古代において敵が多い環境では、自分の命は自分で守る…大人になった証でもあった。つまり、誰かに依存することは多くの人の命を脅かす危険があったのである。 (… In today's society, … "co-dependent relationships" have become a problem. In ancient times and in an environment full of enemies, one's life was one's own … a sign of adulthood. In other words, to be dependent on someone else was to risk the lives of many.)
(2)	i. …相手理解する，…つながらない，…成功をおさめる ii. …現代社会で、…「共依存」の関係が問題となっている。古代において、敵が多い環境では、自分の命は自分で守る…大人になった証でもあった。そのため、誰かに依存することは、多くの人の命を脅かす危険があったのである。 (… In today's society, … "co-dependent relationships" have become a problem. In ancient times and in an environment full of enemies, one's life was one's own … a sign of adulthood. Thus, to be dependent on someone else was to risk the lives of many.)
(5)	i. …相手理解する，…繋がらない，…成功をおさめる ii. …現代社会で、…「共依存」の関係が問題となっている。なぜなら、古代において敵が多い環境では、自分の命は自分で守る…大人になった証でもあったからだ。したがって、誰かに依存することは多くの人の命を脅かす危険があったのである。 (… In today's society, … "co-dependent relationships" have become a problem. Since in ancient times and in an environment full of enemies, one's life was one's own … a sign of adulthood. Therefore, to be dependent on someone else was to risk the lives of many.)
(8)	i. …相手を理解する，…繋がらない，…成功を収める ii. …現代社会で、…「共依存」の関係が問題となっている。古代において敵が多い環境では、自分の命は自分で守る…大人になった証でもあった。つまり、誰かに依存することは多くの人の命を脅かす危険があったのである。 (… In today's society, … "co-dependent relationships" have become a problem. In ancient times and in an environment full of enemies, one's life was one's own … a sign of adulthood. In other woeds, to be dependent on someone else was to risk the lives of many.)

i: Correction focused on grammatical error correction, ii: Correction focused on conjunction insertion

Figure 5: Generation examples of lines (2), (5), and (8) in Table 4.

of the matched task by simultaneously training the matched task and multiple partially-matched tasks, so we do not aim to improve the performance of individual partially-matched tasks.

Conflict results in Tables 4 and 5: The results of lines (4), (5), (7), and (8) in Table 4 show irrelevant dataset b brings more prominent improvement on C-F1. However, In Table 5, after introducing dataset b in ci task, C-F1 shows an obvious downward trend. We suppose these results are dependent on the target tasks. First, we focus on the results of the multi-perspective document revision task in Table 4. Since this task requires multiple tasks to map simultaneously from a source document to a revised document, this task is more complex than handling a single task and requires a large amount of training data. We think the results of each task in the multi-perspective document revision task are improved by simultaneously training the dataset for different tasks included in this task. This reason is that the performance of the seq2seq mapping is improved due to increasing the amount of training data that partially handles the multi-perspective document revision task. Next, we focus on the results of a single task in Table 5. Here, the ci and gec tasks are related tasks for the multi-perspective document revision task, but each task is unrelated. Thus, in the ci task, adding datasets for unrelated tasks without switching tokens may have degraded performance for this task because data from unrelated tasks may have become noise.

6 Conclusion

In this paper, we proposed a novel multi-perspective document revision task that revises multiple perspectives simultaneously to improve the readability and clarity of a document. We created a dataset that simultaneously addresses seven perspectives by using a crowdsourcing service. In addition, to model the multi-perspective document revision task, we presented a seq2seq modeling method with multiple "on-off" switches. This method allowed us to effectively use a multi-perspective document revision dataset and partially-matched datasets, the grammatical error correction and the conjunction insertion datasets. The experimental results obtained using our created dataset demonstrated that using switches is important for modeling the multi-perspective document revision task. In our future work, we will increase the number of partial tasks (e.g., sentence reordering) and develop a model architecture that is suitable for handling much longer documents.

References

Regina Barzilay and Mirella Lapata. 2008. Modeling local coherence: An entity-based approach. *Computational Linguistics*, pages 1–34.

Xinchi Chen, Xipeng Qiu, and Xuanjing Huang. 2016. Neural sentence ordering. *arXiv preprint arXiv:1607.06952*.

Daniel Dahlmeier, Hwee Tou Ng, and Siew Mei Wu. 2013. Building a large annotated corpus of learner

english: The nus corpus of learner english. In *Proceedings of the eighth workshop on innovative use of NLP for building educational applications*, pages 22–31.

Zeyu Dai and Ruihong Huang. 2018. Improving implicit discourse relation classification by modeling inter-dependencies of discourse units in a paragraph. In *Proc. Conference of the North American Chapter of the Association for Computational Linguistics (NAACL)*, pages 141–151.

John Deaton. 2019. Transformers and pointer-generator networks for abstractive summarization.

Roman Grundkiewicz, Marcin Junczys-Dowmuntz, and Kenneth Heafield. 2019. Neural grammatical error correction systems with unsupervised pre-training on synthetic data. In *Proc. Workshop on Innovative Use of NLP for Building Educational Applications (BEA Workshop)*, pages 252–263.

Mandy Guo, Zihang Dai, Denny Vrandečić, and Rami Al-Rfou. 2020. Wiki-40b: Multilingual language model dataset. In *Proc. Language Resources and Evaluation Conference (LREC)*, pages 2440–2452.

Mana Ihori, Naoki Makishima, Tomohiro Tanaka, Akihiko Takashima, Shota Orihashi, and Ryo Masumura. 2021a. Mapgn: Masked pointer-generator network for sequence-to-sequence pre-training. In *Proc. International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 7563–7567.

Mana Ihori, Naoki Makishima, Tomohiro Tanaka, Akihiko Takashima, Shota Orihashi, and Ryo Masumura. 2021b. Zero-Shot Joint Modeling of Multiple Spoken-Text-Style Conversion Tasks Using Switching Tokens. In *Proc. International Speech Communication Association (Interspeech)*, pages 776–780.

Takumi Ito, Tatsuki Kuribayashi, Masatoshi Hidaka, Jun Suzuki, and Kentaro Inui. 2020. Langsmith: An interactive academic text revision system. In *Proc. Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 216–226.

Marcin Junczys-Dowmunt and Roman Grundkiewicz. 2016. Phrase-based machine translation is state-of-the-art for automatic grammatical error correction. In *Proc. Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1546–1556.

Marcin Junczys-Dowmunt, Roman Grundkiewicz, Shubha Guha, and Kenneth Heafield. 2018. Approaching neural grammatical error correction as a low-resource machine translation task. In *Proc. Conference of the North American Chapter of the Association for Computational Linguistics (NAACL)*, pages 595–606.

Shun Kiyono, Jun Suzuki, Masato Mita, Tomoya Mizumoto, and Kentaro Inui. 2020. An empirical study of incorporating pseudo data into grammatical error correction. In *Proc. Conference on Empirical Methods in Natural Language Processing and International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 1236–1242.

Vladimir I Levenshtein et al. 1966. Binary codes capable of correcting deletions, insertions, and reversals. In *Soviet physics doklady*, pages 707–710.

Zhe Lin, Yitao Cai, and Xiaojun Wan. 2021. Towards document-level paraphrase generation with sentence rewriting and reordering. In *Findings of the Association for Computational Linguistics: EMNLP 2021*, pages 1033–1044.

Liyuan Liu, Haoming Jiang, Pengcheng He, Weizhu Chen, Xiaodong Liu, Jianfeng Gao, and Jiawei Han. 2019. On the variance of the adaptive learning rate and beyond. In *Proc. International Conference on Learning Representations (ICLR)*.

Yang Liu, Sujian Li, Xiaodong Zhang, and Zhifang Sui. 2016. Implicit discourse relation classification via multi-task neural networks. In *Proceedings of the AAAI Conference on Artificial Intelligence*.

Michal Lukasik, Srinadh Bhojanapalli, Aditya Menon, and Sanjiv Kumar. 2020. Does label smoothing mitigate label noise? In *Proc. the International Conference on Machine Learning (ICML)*, pages 6448–6458.

Tomoya Mizumoto and Yuji Matsumoto. 2016. Discriminative reranking for grammatical error correction with statistical machine translation. In *Proc. Conference of the North American Chapter of the Association for Computational Linguistics (NAACL)*, pages 1133–1138.

Courtney Napoles, Keisuke Sakaguchi, Matt Post, and Joel Tetreault. 2015. Ground truth for grammatical error correction metrics. In *Proc. Annual Meeting on Association for Computational Linguistics (ACL)*, pages 588–593.

Sascha Rothe, Sebastian Krause, Jonathan Mallinson, Eric Malmi, and Aliaksei Severyn. 2021. A simple recipe for multilingual grammatical error correction. In *Proc. Annual Meeting of the Association for Computational Linguistics and the International Joint Conference on Natural Language Processing (ACL-IJCNLP)*, pages 702–707.

Yu Sawai, Mamoru Komachi, and Yuji Matsumoto. 2013. A learner corpus-based approach to verb suggestion for ESL. In *Proc. Annual Meeting of the Association for Computational Linguistics (ACL)*, pages 708–713.

Yu Tanaka, Yugo Murawaki, Daisuke Kawahara, and Sadao Kurohashi. 2020. Building a Japanese typo dataset from Wikipedia's revision history. In *Proc. Annual Meeting of the Association for Computational Linguistics (ACL)*, pages 230–236.

Chung-Ting Tsai, Jhih-Jie Chen, Ching-Yu Yang, and Jason S. Chang. 2020. LinggleWrite: a coaching system for essay writing. In *Proc. Annual Meeting of*

the Association for Computational Linguistics (ACL), pages 127–133.

Tianming Wang and Xiaojun Wan. 2019. Hierarchical attention networks for sentence ordering. In *Proc. AAAI Conference on Artificial Intelligence (AAAI)*, pages 7184–7191.

Bonnie Webber, Rashmi Prasad, Alan Lee, and Aravind Joshi. 2019. The penn discourse treebank 3.0 annotation manual. *Philadelphia, University of Pennsylvania*.

Yongjing Yin, Linfeng Song, Jinsong Su, Jiali Zeng, Chulun Zhou, and Jiebo Luo. 2019. Graph-based neural sentence ordering. *arXiv preprint arXiv:1912.07225*.

Zheng Yuan and Ted Briscoe. 2016. Grammatical error correction using neural machine translation. In *Proc. Conference of the North American Chapter of the Association for Computational Linguistics (NAACL)*, pages 380–386.

A Survey of Automatic Text Summarization using Graph Neural Networks

Marco Ferdinand Salchner
marco.salchner@student.uibk.ac.at
Department of Computer Science,
University of Innsbruck, Austria

Adam Jatowt
adam.jatowt@uibk.ac.at
Department of Computer Science &
Digital Science Center,
University of Innsbruck, Austria

Abstract

Although automatic text summarization (ATS) has been researched for several decades, the application of graph neural networks (GNNs) to this task started relatively recently. In this survey we provide an overview on the rapidly evolving approach of using GNNs for the task of automatic text summarization. In particular we provide detailed information on the functionality of GNNs in the context of ATS, and a comprehensive overview of models utilizing this approach.

1 Introduction

The advent of the internet has led to an explosion in the amount of textual information available online. The extensive availability of textual information paired with a need to quickly understand it has led to major efforts in the field of automatic text summarization (ATS). For a comprehensive review and survey of ATS as a task we recommend a recent survey by El-Kassas et al. (2021).

The goal of ATS is to produce a concise, correct and fluent summary of a given text. Although this definition is intuitively understandable, there is no commonly agreed upon formal definition for these qualities. This is in part due to the difficulty of the task itself as producing a summary with the above properties is challenging, even for humans.

Although the field of ATS has made major steps forward, generally, we differentiate between two basic approaches to the task of ATS. The first being extractive ATS and the second being abstractive ATS. Extractive ATS involves extracting text spans from the original document such that a summary is generated. Thus the main challenge consists in identifying useful spans of text from the original document. This approach is popular as it eliminates the non-trivial task of generating factually correct and coherent sentences. Abstractive ATS on the other hand involves the challenging task of generating novel sentences for a summary. Despite impressive advancements, generating factually correct and fluent sentences is still a major challenge in ATS (Kryściński et al., 2020). In addition to the above, one differentiates between the task of single-document and multi-document summarization.

1.1 Why graph neural networks ?

Contemporary solutions to the task of ATS suffer from a number of issues, chiefly an inconsistent evaluation protocol and, somewhat, a lack of progress, as noted by Kryściński et al. (2019). In recent years GNNs have been successfully applied to a number of downstream NLP tasks such as classification (Liu et al., 2020b) (Zhang et al., 2020b) and translation (Xu et al., 2021) (Yin et al., 2020). Although GNNs may not be able to solve all problems related to the task of ATS, we believe that they can at least give a new perspective to this task. Generally GNNs bring a number of advantages to ATS which we believe to be significant enough to warrant further research, and this survey. In particular we want to highlight the following aspects of GNNs:

- **Scalability and Flexibility**. A vast number of ATS models are based on **BERT** (Devlin et al., 2019). However, the computational complexity of **BERT**-based ATS models grows quadratic with the input length; due to the self-attention operation. This fact renders them impractical for long, or even medium sized text documents. Recently some work has been done in order to circumvent this limiting factor (Ding et al., 2020) (Zhang et al., 2021). In contrast, GNNs can scale by their nature to graphs of thousands of nodes and more. This is in part due to the linear scaling of the memory cost with regards to the input size. The total memory cost of a GNN model depends on the size of the graph, the number of layers and the feature vector size of the nodes present. Formally, for L layers and an input

Proceedings of the 29th International Conference on Computational Linguistics, pages 6139–6150
October 12–17, 2022.

of N nodes with each node's feature vector being of size H the memory complexity is $O(LNH)$. But even for very large graphs on the scale of millions of nodes one can utilize GNNs. This can be achieved using methods such as neighbour sampling or distributing the graph over multiple GPUs, as done for example by Jia et al. (2020b). We recommend the paper by Li et al. (2021) for insights as to how one can train large and very deep GNNs. As the input of a GNN is a graph, the input can vary in size, therefore GNNs are also able to cope with changing text sizes and structures. Both of these aspects combined allow GNNs to produce summaries which are not restricted by hard-coded limits related to input or output size.

- **Understanding and Explainability**. It is often difficult to understand *why* a model arrived at a certain conclusion. Additionally it is often difficult to see *how* the model aggregates information. This is not the case with GNNs, as with the help of methods such as **GNN Explainer** (Ying et al., 2019) one can understand which nodes were used by the model to reach its output. This removes a layer of the *black-box magic* present in many current non-GNN models. We recommend the survey by Yuan et al. (2020) for an overview of methods for generating explanations for GNNs.

1.2 Related Surveys

As the application of GNNs to ATS is rather novel, to the best of our knowledge there is only one survey on the topic. The survey by Luo et al. (2020) gives an introduction to the topic. However, it does not provide much detail on GNNs in the context of ATS, nor does it cover all models in the space with a taxonomy.

As for GNNs themselves, there exists a large number of surveys on GNNs as a technology. In particular, we want to highlight the surveys by Wu et al. (2020b) and Zhou et al. (2020) which provide a general overview on GNNs and their applications in different fields. The survey by Abadal et al. (2021) provides more technical and theoretical details on GNNs. The survey by Wu et al. (2021a) is of particular interest as it focuses on the usage of GNNs for NLP as a domain. Additionally, we want to note here the survey by Wu et al. (2020a) on the usage of GNNs in recommender systems and the review by Malekzadeh et al. (2021) on the usage of GNNs for text classification.

For a more theoretical approach we point the reader to the analysis by Xu et al. (2018) which establishes a number of important theoretical properties for GNNs. An analysis of the Vapnik–Chervonenkis (VC) dimension of GNNs was performed by Veličković et al. (2018).

In general, the contributions of our survey are as follows:

1. We provide a detailed explanation of GNNs in the context of ATS.
2. We introduce a simple taxonomy for GNN models used for ATS.
3. We provide a comprehensive overview of innovative GNN models for ATS and discuss further directions for future research.

The rest of survey is structured as follows. First we will give a comprehensive explanation of GNNs in the context of ATS. Next, we will explore a number of interesting and innovative models. Finally, we will finish with a conclusion and an outlook on the future usage of GNNs for ATS.

2 Graph Neural Networks

Graph based methods for ATS are not a new innovation, with methods such as TextRank (Mihalcea and Tarau, 2004) being first presented in the early 2000s. In fact, the core idea of early graph-based approaches such as TextRank is similar to the one used by GNNs designed for the ATS task. The idea is to encode the structural and semantic information contained within a text document into an explicit form with the help of a graph.

Deep learning has by now become a common tool for solving tasks throughout many domains, with various end-to-end paradigms such as recurrent-neural-networks (RNNs) or convolutional-neural-networks (CNNs) emerging as versatile and powerful tools. In particular deep learning has shown extraordinary power for data which is *Euclidean* in nature. As an example, images can be represented as regular grids in Euclidean space. Using such a representation CNNs are able to extract meaningful local representations with the help of convolution. However, for many domains and applications such a representation is often either inconvenient or not even directly possible. The most obvious example being the field of

chemistry in which molecules should be modeled as graphs. The complexity of translating such complex data into existing deep learning paradigms has led, in part, to the development of GNNs which attempt to leverage the power of deep learning for *non-Euclidean* data.

2.1 Defining the Graph

The first step of developing a GNN involves defining and designing the graph structure to be used. Formally a graph is defined as $G = (V, E)$ where V is a set of nodes, and E a set of edges. Let $v_i \in V$ denote a node; then each edge $e \in E$ is defined as $e_{i,j} = (v_i, v_j)$, that is pointing from node v_i to node v_j. In addition to that for a graph with $n = |V|$ nodes and $m = |E|$ edges a feature matrix $X \in R^{n \times d}$ is defined where each node i carries a feature vector $x_i \in R^d$. Note that d denotes the dimension of the feature vector. This encodes the information for the structure represented by the node. Often it is also important to encode specific information for the edge of two nodes. For this we additionally define an edge feature matrix X^e where $X^e \in R^{m \times c}$ with each edge e between nodes i and j carrying an edge feature vector $w_{i,j} \in R^c$ where c denotes the dimension of the edge feature vector.

Using the above definitions there are two common scenarios, either the data is inherently structured or inherently unstructured. In the first case, a direct translation into the above structure is possible, although additional information may be encoded by the designer. For example in knowledge graphs or molecule simulations such a direct translation would be possible, as the data itself already forms a valid graph, only the feature vectors would have to be engineered in an appropriate way. In the second scenario the data implicitly contains a graph-like structure, as is the case for ATS. Natural language text contains structure but that structure is not directly available as a graph. Simplistically one could consider text as a linear graph, that is each sentence is a node and all sentences following each other are connected with edges. However, this would not expose all the possible information hidden, as text implicitly contains much structural and semantic information, both of which can explicitly be modeled with the help of a graph.

A common scenario for ATS is that the text is encoded into the graph using word nodes and sentence nodes. The above definition does not allow such an option as it assumes an homogeneous graph, that is each node's feature vector is equal in dimension. However, for ATS and many other tasks heterogeneous graphs are more common. This is due to the fact that heterogeneous graphs allow to include different types of structures within the same graph. However, all of this is not an issue as one can define a feature matrix X for each type of node that exists within the graph. In the case of ATS with word and sentence type nodes one would define $X_w \in R^{m \times d_w}$ as the feature matrix for all word nodes and $X_s \in R^{n \times d_s}$ as the feature matrix for all sentence nodes with d_w being the dimension of the word node feature vectors, and d_s representing the dimension of sentence node feature vectors.

For ATS it is also sometimes useful to utilize the design space of *directed vs. undirected* edges. The choice of *directed vs. undirected* edges allows to explicitly encode structural information. For instance, one can explicitly encode the order of words or sentences with directed edges.

2.2 Defining the Neural Network

There are three ways in which GNNs differ from more conventional neural networks, the *input*, the *output* and the *way information is aggregated* within the network. The input is the above described graph.

The output and associated loss function are in the case of GNNs bound to the structure of the data i.e. the graph. Generally there are three levels of output possible, namely, on the *node level*, on the *edge level* and on the *graph level*. Both node and edge level output involve predictions or classifications on their respective components of the graph; additionally, in the case of edge level output, one can pose the task of predicting the edge itself. On the graph level one can pose the task of predicting subgraphs or perform graph segmentation. In the context of ATS, the most direct output and loss formulation consist in predicting binary inclusion labels on nodes representing sentences, or phrases, using a cross-entropy loss.

The way information is aggregated is usually done through either spatial convolution or spectral convolution. The basic idea of spatial convolution involves extending the well-known convolution operator to graph structures, whereas spectral convolution is based on graph signal processing. In recent years spatial convolution has become a

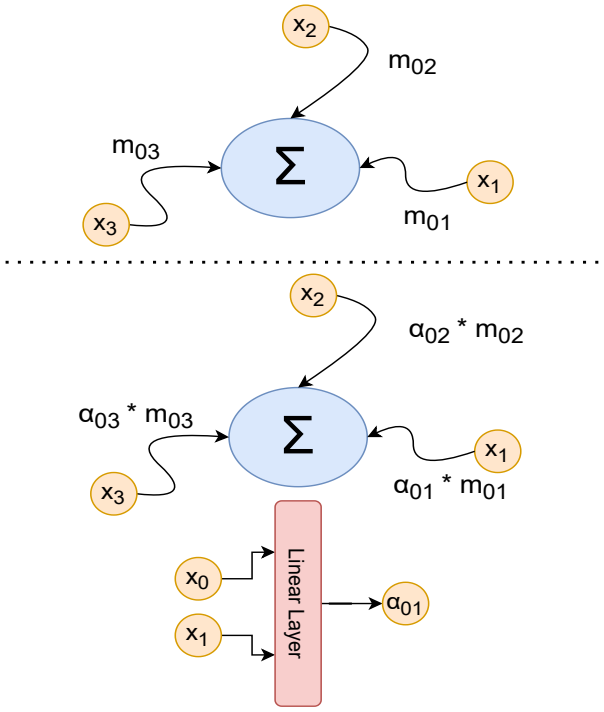

Figure 1: Comparison of spatial convolution (above) and GAT (below) over the node zero with feature vector x_0. In particular, the attention scores α in the GAT attenuate the messages received and are obtained by a learnable linear layer of the two nodes involved in the message exchange.

popular approach and is the preferred approach for GNNs for ATS due to the flexibility and efficiency it offers. More specifically, in terms of efficiency, spectral convolution involves either computations over the entire graph or eigenvector computations. None of which are necessary for spatial convolution; additionally, spatial convolution does not assume a fixed graph structure, allowing better generalization. In addition to this the locality of the spatial convolution operation also allows it to be performed in batches of nodes, instead of the entire graph. This is especially relevant for large graphs, or, in the context of this paper, large input texts.

The setup described thus far implicitly assumes that the GNN performs extractive summarization. This is in fact the case and for what we will later define as *standalone GNNs* i.e. GNNs that are not part of a larger system. No purely abstractive approaches have so far been developed, to the best of our knowledge.

2.3 Spatial Convolution and Message Passing

One can view spatial convolution as used in GNNs as a generalization of the convolution used in neural networks such as CNNs. As an example, in the case of images, one can imagine 2D convolution as being applied to a regular grid of nodes where each node represents a pixel in the image. The resulting 2D convolution applied to one target node is then the weighted average of node (pixel) values of the neighbours of the target node. Generalizing this idea to a non-regular grids leads to spatial convolution. However, different to images and regular grids, in graphs, the neighbours of each target node are unordered and can vary in number and their feature vector representation. The major challenge with this extension consists therefore in dealing with the unordered and inconsistent neighbourhood sizes inherent to homogeneous and heterogeneous graphs, with an additional challenge being posed by the differing feature vector representations in heterogeneous graphs.

Directly translating the above description of convolution into a mathematical formulation leads to a valid information propagation scheme. However, such a description suffers from scalability issues due to it directly operating over the entire graph. As such modern GNNs use, what is commonly referred to as, *message passing*. In practice, this means that nodes within the graph exchange messages (perform convolutions) with their neighbours for a number of iterations. Thereby the network is able to diffuse information throughout the graph. Consequently, the more iterations, the further outwards information is propagated throughout the graph. In the terminology of CNNs one would say that the more message passing iterations, the larger the receptive field of the convolution. Formally, one can define message passing (Grattarola and Alippi, 2021) for each time step t as two equations:

$$m_{i,j}^{t+1} = \phi(x_i^t, x_j^t, w_{i,j}^t), e_{i,j} \in E \qquad (1)$$

This first equation describes how messages are generated. A differentiable function ϕ generates messages m for each edge which connects nodes using the node features and edge feature present.

$$x_i^{t+1} = \psi(x_i^t, \rho(\{m_{i,j}^{t+1} : e_{i,j} \in E\})) \qquad (2)$$

The above equation is the core of the message passing framework and describes how each node feature is updated. The first part consists in the application of a permutation-invariant reduction function ρ. This function aggregates all incoming messages to a node. Then another differen-

tiable function ψ combines the reduced messages received with the previous state. Using these two equations one can utilize message passing for learnable layers.

The convolution layer for a GNN is then defined with a learnable weight W such that the message per edge is $m_{i,j}^{t+1} = x_j^t$ and the aggregation is the normalized sum of messages, i.e. $x_i^{t+1} = \sigma(b^t + \sum_{m_{i,j} \in M(i)} \frac{1}{c_{j,i}} m_{i,j}^{t+1} W^t)$ where $M(i)$ represents the set of messages received by node i, σ is the activation function, b is the bias, and $c_{j,i}$ is an appropriate scaling factor, e.g., the square root of the node degree. Note how it is important for the reduction function, in this case a *sum* function, to be permutation-invariant as otherwise GNNs could not handle the unordered nature of graphs.

The above presented convolution layer does not allow the model to *filter* unimportant neighbours. Inspired by the attention mechanism popularized by transformer networks (Vaswani et al., 2017), graph attention networks (GAT) (Veličković et al., 2018) assign attention scores to each neighbour. A schematic depiction of the two variants of spatial convolution can be seen in Figure 1 with GAT depicted on the lower part of the figure. The introduction of attention scores to the spatial convolution allows the model to explicitly assign importance to certain nodes and their messages. Just as in transformers GAT is formulated with multi-head attention. The modification to the previously presented convolution layer follows closely the common attention formulation. Formally, $x_i^{t+1} = \|_K \, \sigma(b^t + \sum_{m_{i,j} \in M(i)} \alpha_{i,j} m_{i,j}^{t+1} W^t)$ where $\alpha_{i,j}$ is the attention score between node i and node j and K denotes the number of concatenated heads. The attention scores are computed with $r_{i,j} = LeakyReLU(a^T [W x_i^t \parallel W x_j^t])$. This score is then normalized to obtain the attention score per edge $\alpha_{i,j} = softmax_i(r_{i,j})$. We want to highlight here a recent development which Brody et al. (2021) simply denote as GATv2. Their main improvement aims at the fact that in the above calculation both learnable parameters a and W effectively fold into a single linear layer, thus the expressive power of the layer is less than what it could be. The *fix* introduced by GATv2 pulls the two parameters apart, thus achieving more expressive power while not increasing computational complexity. Taking the above description the attention score for GATv2 is modified as follows $r_{i,j} = a^T LeakyReLU([W x_i \parallel W x_j^t])$. In both synthetic and real datasets this modification shows superior performance, which is supported by a theoretical analysis of the authors.

There are numerous modifications and extensions to the basic convolution presented here. However, for ATS models, GAT layers are dominating as the workhorse for most models. The reasoning for their dominance can be explained by the similar success that attention transformers have had in conventional neural networks for AST. We expect that GATv2 will continue this trend as it is an attractive and simple improvement for the currently dominating GAT. Although the authors of GATv2 note that it is not yet entirely clear which tasks would benefit the most from the usage of GATv2 over GAT, which will require more research and models to use GATv2.

In ATS the graphs used are in nearly all cases not homogeneous. However, the equations presented here do not work for heterogeneous graphs. The solution for this problem involves defining one convolution layer for each node type combination occurring within the graph. In the case of the already discussed sentence and word node graph there would be four possible combinations of types, and four convolution layers which would have to be defined if the graph were fully connected.

Convolution is a central aspect of GNNs, but pooling also presents an important and common operation, especially whenever GNNs are used jointly with other models. Pooling in GNNs is achieved by generating a global representation of the graph, or a subset of the graph, by pooling together features of nodes. This is usually done with some function f where f is commonly the *mean*, *max* or *sum*.

We want to explicitly point out to the reader that the construction of GNNs does not require special datasets. All GNN models for ATS use the common benchmark single-document and multi-document summarization datasets such as DUC 2004 or CNN/Dailymail. The only requirement for any ATS, or textual dataset, is for the designer to find an appropriate way of encoding sentences, words, subwords etc. into feature vectors, and finding a sensible way of connecting them.

3 Graph Neural Network Models For Automatic Text Summarization

The current state of research clearly shows that the usage of GNNs for ATS follows one of two patterns,

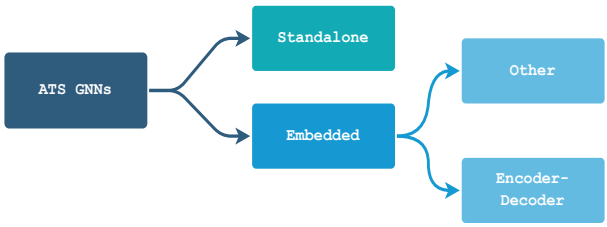

Figure 2: Taxonomy of GNN models used for ATS.

either the GNN is used directly for the ATS task or it is used to support a larger system. These trends justify the taxonomy as seen in Figure 2. More precisely, we classify standalone ATS GNNs as GNN models which are directly responsible for generating a summary. Embedded GNNs on the other hand are used to support a larger system, and are not directly or solely responsible for producing the summary. In addition to this difference, we further differentiate between embedded GNNs used in an encoder-decoder setting and other embedded GNNs.

Note that we do not specifically differentiate between single and multi-document GNN summarization models as some models are by design capable of handling both tasks. Additionally, we do not make a distinction between abstractive and extractive approaches, since embedded GNNs can be part of either approach, while abstractive approaches have not yet been developed for standalone GNNs, to the best of our knowledge.

3.1 Standalone GNNs

We will start our discussion of standalone GNN models with **HeterSumGraph** (HSG), a model proposed by Wang et al. (2020). We will do so due to the fact that this model illustrates concepts and ideas seen throughout GNNs models used for ATS. An illustration of the general concepts presented here can be seen in Figure 3.

The **HSG** model encodes each text into a graph with three node types, sentence nodes, word nodes and document nodes. The connection between these nodes is decided by inclusion i.e. if the word represented by a word node occurs in a sentence then their respective nodes are connected by an edge. The same principle applies to document nodes which are connected depending on whether a word, represented by a word node, occurs within the document. This is a flexible structure, as it can be used in a single-document but also multi-document setting.

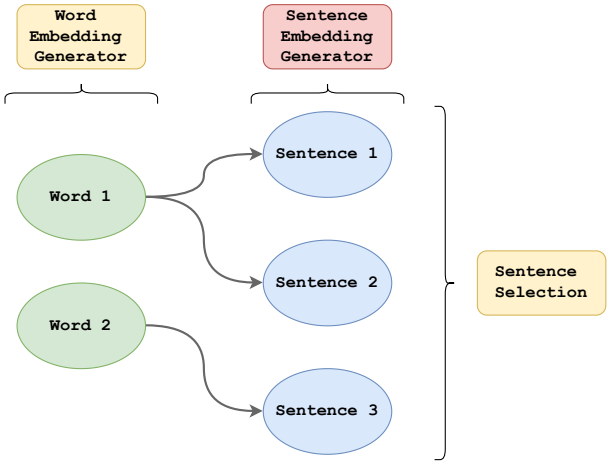

Figure 3: General architecture of standalone GNNs with word nodes and sentence nodes, encoders for both node types and a sentence selection mechanism. Inspired by **HSG**.

The feature vectors for all nodes are obtained by encoders and the edge weights are obtained by computing the TF-IDF score for each word. The neural network consists of a modified GAT layer. The GAT is modified to consider the TF-IDF value of the connecting edge. Additionally a position-wise feed-forward (FFN) layer consisting of two linear transformations is applied to the hidden state after the convolution. In total three convolution layers are used, word-sentence, sentence-word and word-document. The model is then trained on a node-based binary classification task that is predicting whether a sentence node is to be included for the summary or not.

The classification itself is done by a single linear layer. The model then does not directly use the predicted nodes to produce the summary. Instead trigram blocking (Paulus et al., 2018) is used during sentence selection in order to ensure sparsity of the generated summary.

The results for this model are quite impressive as it outperforms non-BERT based models on both single-document and multi-document summarization for the CNN/DailyMail dataset. One should especially note the flexibility and ability to use this model for two tasks.

An older model by Muratore et al. (2010) can be considered a precursor to this architecture. A simple extension to the **HSG** model is proposed by Ya et al. (2021). In their extension they modify the model for query constraints for the summary. This is achieved by adding a query node to the graph structure. Additionally, they introduce a mu-

tual information maximization mechanism during training.

A model which further follows this structure is the one by Linmei et al. (2019). The authors there extend the attention mechanism by adding another layer of attention, allowing it to include information about the type of the node during convolution. The GNN model by Jing et al. (2021) encodes even more information into the graph by considering the relation between sentences on a number of different levels. In particular, they encode the semantic and syntactical relationship between sentences within the graph.

This idea of encoding additional information into the graph is also followed by Antognini and Faltings (2019). They introduce an additional *universal* feature vector which is added to each sentence node embedding. This *universal* feature vector is learned from a large unrelated and general corpus. This model is also unique in that it focuses on the summarization of very small texts, on average less than 100 words.

Taking this basic structure and idea even further is the model called **HAHSum** by Jia et al. (2020a). The construction of the input graph for **HAHSum** is more involved as it aims to significantly reduce semantic sparsity by utilizing named entities. The model uses three types of nodes, named entity nodes, word nodes and sentence nodes, with the named entity nodes being anonymized tokens. The graph is then built as follows, word nodes are connected with a directed edge to a sentence node if they occur within the sentence. Two named entities are connected with an undirected edge if they represent the *same* entity and two sentence nodes are connected with an undirected edge if they share a trigram. Additionally, sequentially occurring words and entities are connected with a directed edge. This setup shows how one can encode a substantial amount of implicit information in an explicit manner.

HAHSum uses a GAT for each of the five node type combinations found within the graph. Just as in **HSG**, a FFN is applied after the multi-head attention and again as in the previous model a linear layer is used to perform the binary classification of the sentence nodes.

The results for **HAHSum** show that GNNs can perform very well. The authors of the paper tested the model on the CNN/Daily Mail, Newsroom and NYT dataset. The model outperforms very powerful models such as **MATCHSUM** (Zhong et al., 2020) and is even able to compete in some metrics with leading abstractive models such as **PEGASUS** (Zhang et al., 2020a). The results of an *Amazon Mechanical Turk* experiment corroborate these results and show that for human readers **HAHSum** produce summaries with superior fluency and conciseness.

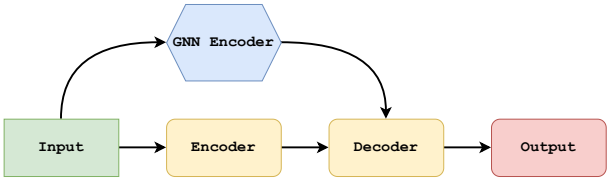

Figure 4: Typical setup for embedded GNNs in the encoder-decoder category. The GNN is used to encode part of the input and then the resulting encodings are forwarded to the decoder.

Another recent GNN model which has achieved great performance in the task of multi-document summarization is the **SgSum** model by Chen et al. (2021). Different to the approaches outlined above, the **SgSum** model uses graph pooling to extract sub-graphs from encoded documents. That is, it first transforms the documents into a large graph, then generates a number of sub-graphs via pooling and convolution. These sub-graphs are then ranked and thereby selected for a summary. This is quite an innovative approach as it casts the problem of multi-document summarization as a simple sub-graph selection problem. Additionally, it outputs an integral summary, that is the entire summary is output by the model in the form of the sub-graph of sentences.

3.2 Embedded GNNs

The first embedded GNN model we will present is the **GRU-GCN** model by Yasunaga et al. (2017). Despite being an older model it provides an illustrative introduction as to how one can effectively incorporate GNNs into established deep learning methods. As this model utilizes GNNs within a sequence-sequence architecture it falls into the encoder-decoder category of embedded GNNs. The model works exclusively with multi-document summarization. We want to note that due to the age of this model the GNN uses spectral-based convolution instead of spatial-based convolution.

As the model is an encoding-decoding embedded GNN it features three parts: the encoding part, the GNN and a decoding part. The encoding and decod-

GNN CNN/DailyMail Performance Overview					
Model Name	Type	Description	R-1	R-2	R-L
BERTSUMEXT (Liu and Lapata, 2019)	Extractive BERT baseline	BERT-based	43.85	20.34	39.90
Topic-Graphsum (Cui et al., 2020)	Extractive (Embedded)	NTM + GNN	44.02	20.81	40.55
DSGSUM (Bi et al., 2021)	Abstractive (Embedded)	Seq-Seq with GNN	41.96	19.29	38.98
Syntactic-Graph (Xu et al., 2020a)	Abstractive (Embedded)	Syntactic graph	41.79	19.06	38.56
HAHSUM (Jia et al., 2020a)	Extractive (Standalone)	Entity-relations + words and sentences	**44.68**	**21.30**	**40.75**
Multi-Gras (Jing et al., 2021)	Extractive (Standalone)	Sentence information encoding	43.16	20.14	39.49
HSG (Wang et al., 2020)	Extractive (Standalone)	Multi-layer encoding of documents	42.95	19.76	39.23
DISCOBERT (Xu et al., 2020b)	Extractive (Embedded)	BERT+GNN	43.77	20.85	40.67
GNN DUC-2004 Performance Overview					
GRU-GCN (Yasunaga et al., 2017)	Extractive (Embedded)	GRU+GNN	38.23	9.48	NA
SGSum (Chen et al., 2021)	Extractive (Standalone)	Subgraph extraction	**39.41**	**10.42**	**35.41**

Table 1: Combined results as reported in their respective papers. We only report the best performing model type and only ROUGE-1 (R-1), ROUGE-2 (R-2) and ROUGE-L (R-L) scores. We show results on the CNN/Dailymail dataset for single-document summarization and results on the DUC-2004 dataset for multi-document summarization. *NA* denotes that the authors have not reported this score. Best results marked with boldface.

ing are done by gated recurrent networks (GRUs). More specifically, sentence encodings are produced by the encoding GRU network. These sentence encodings are then used by the GNN whose input graph consists solely of sentence nodes. The edges are determined by semantic relatedness of the sentences. The resulting sentence node feature vectors produced by the GNN are passed to the decoding GRU which computes salience scores for each sentence. The results for this model have been surpassed by other models.

Similar to the **GRU-GCN** model, the **Topic-GraphSum** model by Cui et al. (2020) combines an established deep neural network method with a GNN. In it a variational autoencoder is utilized for modeling topics within a given text, that is, it learns latent topics via encoding-decoding. The GNN is fed a graph consisting of topic nodes and sentence nodes. The topic node embeddings are produced by the autoencoder and the sentence node embeddings are produced by **BERT**. The GNN utilizes a GAT for the prediction of sentences to be used for the summary. Note that the autoencoder and the GNN are trained jointly, which is why this model is classified as an embedded model. However, as it does not utilize the encoder-decoder architecture it falls into the category of other embedded GNNs.

Another embedded GNN which does not follow the encoder-decoder schema is the model by Xu et al. (2020b). Their **DISCOBERT** model incorporates a GNN into a **BERTSUM** (Liu and Lapata, 2019) like architecture.

The **DSGSum** model developed by Bi et al. (2021) utilizes a GNN to enhance the semantic information provided to the model. **DSGSum** uses the GNN mainly for encoding entity information. The input to the GNN consists of an entity graph which has been enriched with a knowledge graph (KG); specifically, the entities and their relations as defined in the KG are encoded into the graph. The GNN then utilizes GAT to produce entity embeddings which are directly used by the decoder part of the architecture. Hence **DSGSum** is an embedded encoder-decoder model. All of the following embedded GNN ATS models follow the ATS GNN encoder-decoder pattern. The general principle of these models is illustrated by Figure 4. This principle being the usage of the GNN to supplement the information provided to the decoder, while also utilizing an encoder for generating the input to the GNN.

The authors of (Wu et al., 2021b) produce a graph based on the semantics of each sentence. The encoding produced by the GNN and a textual encoding of each sentence are then passed to a decoder. Similarly, the model by Xu et al. (2020a) uses a GNN as an encoding component. Their model utilizes the dependency tree of the input as a graph input to the GNN. It also uses a modified attention mechanism which is used to decode the attention of the GAT directly into the decoder part of the architecture. Another model in this category is the model by Liang et al. (2021). Following this idea even further is the model by Li et al. (2020). However, in contrast to **DSGSum** or others, their model uses a GNN directly within the transformer based encoder and decoder blocks, and not as an outside component providing additional encodings.

Another area where GNNs have found some usage is in the abstractive summarization of di-

alogues, which although a niche area, is still part of the ATS task. The model by Zhao et al. (2020) uses a GNN to encode the structure of the conversation into their sequence-to-sequence architecture. This is also done by Feng et al. (2020). Also for dialogue summarization Feng et al. (2021) introduce a special type of graph encoding to a sequence-to-sequence architecture. They utilize a GNN as an encoder and their input graph links information about the speaker, the spoken sentences and other information together.

At this point we also want to shortly mention a few models which do not perform classical ATS but do perform a specialized form of summarization with the help of a GNN. The models by Liu et al. (2020a) and LeClair et al. (2020) both perform code summarization, that is they generate a natural language description/summarization of code written in a programming language. They both utilize a GNN, and also both leverage the abstract syntax tree of the program given. Another interesting model is the model by Wu et al. (2019). Their model performs multi-video summarization with the help of a GNN.

4 Conclusions and Outlook for Future

We have surveyed in this paper the main developments in the area of GNNs applied to the automatic text summarization task, first describing how GNNs work and then discussing the prominent models used for ATS. We have also provided simple categories of the models. Finally, in Table 1 we have combined results from the models mentioned in our survey. Note that this is just a simple enumeration of the best *ROUGE* scores reported, which does not fully capture other important aspects of summaries such as fluency, conciseness, relevancy and in the case of abstractive summarization, factual accuracy.

We have already highlighted some general advantages of GNNs for ATS. Now considering the models presented and the results shown in Table 1 we want to give some pointers towards the future of GNNs for ATS.

- **Outside the Box.** Models such as **SGSum** and **HAHSUM** show how one can achieve great performance by rethinking parts of the ATS task. GNNs provide with their direct acceptance of graphs a lot of freedom with regards to the design of the input as well as the output. The idea of reconsidering multi-document ATS as a subgraph ranking task is a worthwhile approach to consider further.

- **Encoding and Enhancing.** GNNs are able to efficiently and effectively produce encodings of graph structures, which are prevalent throughout texts. Models like **DISCOBERT** show how one can enhance the performance of *traditional* models by incorporating GNNs and leveraging their encoding ability. We believe many models can be improved by incorporating GNNs in ways presented in the survey.

- **Text Length.** GNNs are capable of scaling to very large graph sizes. Summarizing large texts is an important task, but one which has been neglected in the literature. GNNs are in our estimation in a position to perform very well on large text summarization.

- **Explainability.** No GNN ATS model so far has leveraged the explainability aspect of GNNs. We believe that this is a very unique and unexplored research avenue that could reveal valuable insights into ATS and GNNs used for ATS.

References

Sergi Abadal, Akshay Jain, Robert Guirado, Jorge López-Alonso, and Eduard Alarcón. 2021. Computing graph neural networks: A survey from algorithms to accelerators. *ACM Computing Surveys (CSUR)*, 54(9):1–38.

Diego Antognini and Boi Faltings. 2019. Learning to create sentence semantic relation graphs for multi-document summarization. In *Proceedings of the 2nd Workshop on New Frontiers in Summarization*, pages 32–41.

Qiwei Bi, Haoyuan Li, Kun Lu, and Hanfang Yang. 2021. Augmented abstractive summarization with document-level semantic graph. *Journal of Data Science*, 19(3).

Shaked Brody, Uri Alon, and Eran Yahav. 2021. How attentive are graph attention networks? *arXiv preprint arXiv:2105.14491*.

Moye Chen, Wei Li, Jiachen Liu, Xinyan Xiao, Hua Wu, and Haifeng Wang. 2021. Sgsum: Transforming multi-document summarization into sub-graph selection. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 4063–4074.

Peng Cui, Le Hu, and Yuanchao Liu. 2020. Enhancing extractive text summarization with topic-aware graph neural networks. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 5360–5371.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. Bert: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186.

Ming Ding, Chang Zhou, Hongxia Yang, and Jie Tang. 2020. Cogltx: Applying bert to long texts. *Advances in Neural Information Processing Systems*, 33:12792–12804.

Wafaa S. El-Kassas, Cherif R. Salama, Ahmed A. Rafea, and Hoda K. Mohamed. 2021. Automatic text summarization: A comprehensive survey. *Expert Systems with Applications*, 165:113679.

Xiachong Feng, Xiaocheng Feng, and Bing Qin. 2021. Incorporating commonsense knowledge into abstractive dialogue summarization via heterogeneous graph networks. In *China National Conference on Chinese Computational Linguistics*, pages 127–142. Springer.

Xiachong Feng, Xiaocheng Feng, Bing Qin, and Xinwei Geng. 2020. Dialogue discourse-aware graph model and data augmentation for meeting summarization. *Dialogue*, 1:U2.

Daniele Grattarola and Cesare Alippi. 2021. Graph neural networks in tensorflow and keras with spektral [application notes]. *IEEE Computational Intelligence Magazine*, 16(1):99–106.

Ruipeng Jia, Yanan Cao, Hengzhu Tang, Fang Fang, Cong Cao, and Shi Wang. 2020a. Neural extractive summarization with hierarchical attentive heterogeneous graph network. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 3622–3631.

Zhihao Jia, Sina Lin, Mingyu Gao, Matei Zaharia, and Alex Aiken. 2020b. Improving the accuracy, scalability, and performance of graph neural networks with roc. *Proceedings of Machine Learning and Systems*, 2:187–198.

Baoyu Jing, Zeyu You, Tao Yang, Wei Fan, and Hanghang Tong. 2021. Multiplex graph neural network for extractive text summarization. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 133–139.

Wojciech Kryściński, Nitish Shirish Keskar, Bryan McCann, Caiming Xiong, and Richard Socher. 2019. Neural text summarization: A critical evaluation. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 540–551.

Wojciech Kryściński, Bryan McCann, Caiming Xiong, and Richard Socher. 2020. Evaluating the factual consistency of abstractive text summarization. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 9332–9346.

Alexander LeClair, Sakib Haque, Lingfei Wu, and Collin McMillan. 2020. Improved code summarization via a graph neural network. In *Proceedings of the 28th International Conference on Program Comprehension*, pages 184–195.

Guohao Li, Matthias Müller, Bernard Ghanem, and Vladlen Koltun. 2021. Training graph neural networks with 1000 layers. In *International conference on machine learning*, pages 6437–6449. PMLR.

Wei Li, Xinyan Xiao, Jiachen Liu, Hua Wu, Haifeng Wang, and Junping Du. 2020. Leveraging graph to improve abstractive multi-document summarization. *arXiv preprint arXiv:2005.10043*.

Zeyu Liang, Junping Du, Yingxia Shao, and Houye Ji. 2021. Gated graph neural attention networks for abstractive summarization. *Neurocomputing*, 431:128–136.

Hu Linmei, Tianchi Yang, Chuan Shi, Houye Ji, and Xiaoli Li. 2019. Heterogeneous graph attention networks for semi-supervised short text classification. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 4821–4830.

Shangqing Liu, Yu Chen, Xiaofei Xie, Jing Kai Siow, and Yang Liu. 2020a. Retrieval-augmented generation for code summarization via hybrid gnn. In *International Conference on Learning Representations*.

Xien Liu, Xinxin You, Xiao Zhang, Ji Wu, and Ping Lv. 2020b. Tensor graph convolutional networks for text classification. In *Proceedings of the AAAI conference on artificial intelligence*, volume 34, pages 8409–8416.

Yang Liu and Mirella Lapata. 2019. Text summarization with pretrained encoders. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 3730–3740.

Rui Luo, Shan Zhao, and Zhiping Cai. 2020. Application of graph neural network in automatic text summarization. In *National Conference of Theoretical Computer Science*, pages 123–138. Springer.

Masoud Malekzadeh, Parisa Hajibabaee, Maryam Heidari, Samira Zad, Ozlem Uzuner, and James H Jones. 2021. Review of graph neural network in text classification. In *2021 IEEE 12th Annual Ubiquitous Computing, Electronics & Mobile Communication Conference (UEMCON)*, pages 0084–0091. IEEE.

Rada Mihalcea and Paul Tarau. 2004. Textrank: Bringing order into text. In *Proceedings of the 2004 conference on empirical methods in natural language processing*, pages 404–411.

Donatella Muratore, Markus Hagenbuchner, Franco Scarselli, and Ah Chung Tsoi. 2010. Sentence extraction by graph neural networks. In *International Conference on Artificial Neural Networks*, pages 237–246. Springer.

Romain Paulus, Caiming Xiong, and Richard Socher. 2018. A deep reinforced model for abstractive summarization. In *International Conference on Learning Representations*.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. *Advances in neural information processing systems*, 30.

Petar Veličković, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Liò, and Yoshua Bengio. 2018. Graph attention networks. In *International Conference on Learning Representations*.

Danqing Wang, Pengfei Liu, Yining Zheng, Xipeng Qiu, and Xuanjing Huang. 2020. Heterogeneous graph neural networks for extractive document summarization. In *ACL*.

Jiaxin Wu, Sheng-Hua Zhong, and Yan Liu. 2019. Mvsgcn: A novel graph convolutional network for multi-video summarization. In *Proceedings of the 27th ACM International Conference on Multimedia*, pages 827–835.

Lingfei Wu, Yu Chen, Kai Shen, Xiaojie Guo, Hanning Gao, Shucheng Li, Jian Pei, and Bo Long. 2021a. Graph neural networks for natural language processing: A survey. *arXiv preprint arXiv:2106.06090*.

Shiwen Wu, Fei Sun, Wentao Zhang, Xu Xie, and Bin Cui. 2020a. Graph neural networks in recommender systems: a survey. *ACM Computing Surveys (CSUR)*.

Wenhao Wu, Wei Li, Xinyan Xiao, Jiachen Liu, Ziqiang Cao, Sujian Li, Hua Wu, and Haifeng Wang. 2021b. Bass: Boosting abstractive summarization with unified semantic graph. *arXiv preprint arXiv:2105.12041*.

Zonghan Wu, Shirui Pan, Fengwen Chen, Guodong Long, Chengqi Zhang, and S Yu Philip. 2020b. A comprehensive survey on graph neural networks. *IEEE transactions on neural networks and learning systems*, 32(1):4–24.

Haiyang Xu, Yun Wang, Kun Han, Baochang Ma, Junwen Chen, and Xiangang Li. 2020a. Selective attention encoders by syntactic graph convolutional networks for document summarization. In *ICASSP 2020-2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 8219–8223. IEEE.

Jiacheng Xu, Zhe Gan, Yu Cheng, and Jingjing Liu. 2020b. Discourse-aware neural extractive text summarization. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 5021–5031.

Keyulu Xu, Weihua Hu, Jure Leskovec, and Stefanie Jegelka. 2018. How powerful are graph neural networks? In *International Conference on Learning Representations*.

Mingzhou Xu, Liangyou Li, Derek F Wong, Qun Liu, and Lidia S Chao. 2021. Document graph for neural machine translation. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 8435–8448.

Jing Ya, Tingwen Liu, Jiangxia Cao, and Li Guo. 2021. Heterogeneous graph neural networks for query-focused summarization. In *Proceedings of the 2021 SIAM International Conference on Data Mining (SDM)*, pages 720–728. SIAM.

Michihiro Yasunaga, Rui Zhang, Kshitijh Meelu, Ayush Pareek, Krishnan Srinivasan, and Dragomir Radev. 2017. Graph-based neural multi-document summarization. In *Proceedings of the 21st Conference on Computational Natural Language Learning (CoNLL 2017)*, pages 452–462.

Yongjing Yin, Fandong Meng, Jinsong Su, Chulun Zhou, Zhengyuan Yang, Jie Zhou, and Jiebo Luo. 2020. A novel graph-based multi-modal fusion encoder for neural machine translation. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 3025–3035.

Zhitao Ying, Dylan Bourgeois, Jiaxuan You, Marinka Zitnik, and Jure Leskovec. 2019. Gnnexplainer: Generating explanations for graph neural networks. *Advances in neural information processing systems*, 32.

Hao Yuan, Haiyang Yu, Shurui Gui, and Shuiwang Ji. 2020. Explainability in graph neural networks: A taxonomic survey. *arXiv preprint arXiv:2012.15445*.

Hang Zhang, Yeyun Gong, Yelong Shen, Weisheng Li, Jiancheng Lv, Nan Duan, and Weizhu Chen. 2021. Poolingformer: Long document modeling with pooling attention. In *International Conference on Machine Learning*, pages 12437–12446. PMLR.

Jingqing Zhang, Yao Zhao, Mohammad Saleh, and Peter Liu. 2020a. Pegasus: Pre-training with extracted gap-sentences for abstractive summarization. In *International Conference on Machine Learning*, pages 11328–11339. PMLR.

Yufeng Zhang, Xueli Yu, Zeyu Cui, Shu Wu, Zhongzhen Wen, and Liang Wang. 2020b. Every document owns its structure: Inductive text classification via graph neural networks. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 334–339.

Lulu Zhao, Weiran Xu, and Jun Guo. 2020. Improving abstractive dialogue summarization with graph structures and topic words. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 437–449.

Ming Zhong, Pengfei Liu, Yiran Chen, Danqing Wang, Xipeng Qiu, and Xuan-Jing Huang. 2020. Extractive summarization as text matching. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 6197–6208.

Jie Zhou, Ganqu Cui, Shengding Hu, Zhengyan Zhang, Cheng Yang, Zhiyuan Liu, Lifeng Wang, Changcheng Li, and Maosong Sun. 2020. Graph neural networks: A review of methods and applications. *AI Open*, 1:57–81.

Phrase-Level Localization of Inconsistency Errors in Summarization by Weak Supervision

Masato Takatsuka **Tetsunori Kobayashi** **Yoshihiko Hayashi**
Faculty of Science and Engineering, Waseda University
Waseda-machi 27, Shinjuku, Tokyo 1620042, Japan
takatsuka@pcl.cs.waseda.ac.jp
koba@waseda.jp yshk.hayashi@aoni.waseda.jp

Abstract

Although the fluency of automatically generated abstractive summaries has improved significantly with advanced methods, the inconsistency that remains in summarization is recognized as an issue to be addressed. In this study, we propose a methodology for localizing inconsistency errors in summarization. A synthetic dataset that contains a variety of factual errors likely to be produced by a common summarizer is created by applying sentence fusion, compression, and paraphrasing operations. In creating the dataset, we automatically label erroneous phrases and the dependency relations between them as "inconsistent," which can contribute to detecting errors more adequately than existing models that rely only on dependency arc-level labels. Subsequently, this synthetic dataset is employed as weak supervision to train a model called SumPhrase, which jointly localizes errors in a summary and their corresponding sentences in the source document. The empirical results demonstrate that our SumPhrase model can detect factual errors in summarization more effectively than existing weakly supervised methods owing to the phrase-level labeling. Moreover, the joint identification of error-corresponding original sentences is proven to be effective in improving error detection accuracy.

1 Introduction

The quality, particularly the fluency, of automatically generated abstractive summaries has improved significantly (Lewis et al., 2020; Zhang et al., 2020; Raffel et al., 2020) with methods that benefit from large-scale pre-trained language models. However, recent studies (Cao et al., 2018; Maynez et al., 2020) have pointed out that more than 30% of the generated summaries are inconsistent with the source documents owing to unintentionally introduced factual errors, which affect the reliability and usability of summarization systems.

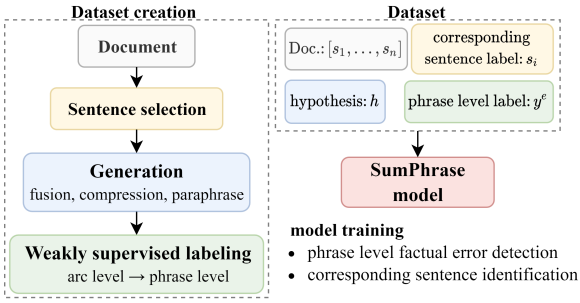

Figure 1: Overview of proposal. A synthetic dataset is created and used as weak supervision to train the SumPhrase error localization model.

Existing approaches for evaluating inconsistency in summarization can be roughly divided into two categories. One is the unsupervised approach that relies on an external natural language understanding (NLU) system, such as natural language inference (NLI) (Falke et al., 2019; Mishra et al., 2021; Laban et al., 2022) or question answering (QA) (Durmus et al., 2020; Wang et al., 2020; Scialom et al., 2021) systems, to validate a summary. Although this approach can benefit from the ever-progressing NLU technologies, the configuration and performance of an inconsistency detection system would be inevitably constrained by the underlying systems.

Therefore, we adopt another approach, namely the weakly supervised approach, which employs a synthetic dataset that contains automatically generated errors (Kryscinski et al., 2020; Goyal and Durrett, 2020). The key to the success of this approach lies in the quality and quantity of the synthetic dataset. It is necessary for such a dataset to contain a variety of errors that are likely to be produced by a common summarization system. However, as argued by Goyal and Durrett (2021), the error distributions that are produced by existing methods are considerably different from those produced by actual summarization models.

Proceedings of the 29th International Conference on Computational Linguistics, pages 6151–6164
October 12–17, 2022.

Figure 1 overviews our proposal for the localization of factual inconsistency errors in summarization. We propose to create a synthetic dataset by applying sentence fusion and compression operations in addition to paraphrasing. Thus, recurring summarization errors are expected to be reproduced. The resulting synthetic dataset is then used as weak supervision to train an error localization model called SumPhrase. This model jointly localizes errors in a summary and their corresponding sentences in the source document[1].

In the dataset creation process, a single sentence or a pair of sentences is first selected from a source document. A hypothesis sentence is then generated from the selected sentence or sentence pair, which is expected to contain various intrinsic factual errors. Thereafter, the hypothesis sentence is labeled at the phrase level, where each label is computed from dependency arc-level labels. Finally, the dataset for training the SumPhrase model is created by organizing the phrase-level labeled data.

Furthermore, the detection of factual errors in a summary can be combined with the identification of their corresponding sentences in the source document. This joint approach not only improves the usability of the error detection system but also improves the error detection performance in a multi-task learning setting (Kryscinski et al., 2020).

The contributions of this study are as follows:

- We present an improved method for generating a dataset by incorporating common means of summary generation, such as sentence fusion and compression, in addition to paraphrasing. We empirically investigate the effectiveness of these types of operations.
- We propose a model that jointly detects errors in a summary by fully using the phrase-level labels and identifies their corresponding sentences in the source document.
- We empirically demonstrate that the proposed method can localize the inconsistencies between a source document and the summary more effectively than existing weakly supervised methods.

2 Related Work

We review the approaches for inconsistency detection in summarization by classifying them into the following two categories.

2.1 Unsupervised Approach

We classify studies that exploit an external NLU system/model into this category.

A system that uses an NLI model determines that a summary is inconsistent if it cannot be entailed from the input document (Falke et al., 2019; Mishra et al., 2021). This method tends to suffer from a mismatch between the lengths of input texts: the documents to be summarized are generally longer than the usual premises collected in an NLI dataset, which makes inconsistency detection difficult (Mishra et al., 2021). Laban et al. (2022) addressed this issue by aggregating entailment scores measured between sentence pairs in the segments rendered from a source document. In contrast, a system that relies on a QA model considers a summary to be inconsistent if an external QA system fails to answer the questions related to a source document (Durmus et al., 2020; Wang et al., 2020; Scialom et al., 2021). The adequacy of a generated question, as well as the performance of the QA model, may affect the error detection performance.

2.2 Weakly Supervised Approach

We classify inconsistency detection systems that rely on supervised learning with artificially developed datasets into this category. In this approach, a dataset must reproduce the error distribution of a common summarizer as effectively as possible. Systems that use this method can be further classified according to how they generate errors for a summary.

Rule-based text transformation systems were developed to generate errors in (Kryscinski et al., 2020; Zhao et al., 2020; Zhang et al., 2021; Cao et al., 2020). For example, in FactCCX (Kryscinski et al., 2020), errors were generated by replacing an entity name and negating a sentence. Zeng et al. (2021) further enhanced the dataset with a data augmentation technique that applies an adversarial attack mechanism.

Generative models were employed in (Goyal and Durrett, 2021, 2020) to generate erroneous texts. In particular, Goyal and Durrett (2020) proposed using a sentence with a lower posterior probability as a potentially erroneous sentence. These generated sentences were then dependency-parsed, and the dependency arcs that were only found in the hypothesis sentence were annotated as erroneous.

[1]The dataset and code are available at https://github.com/taka2946/sumphrase

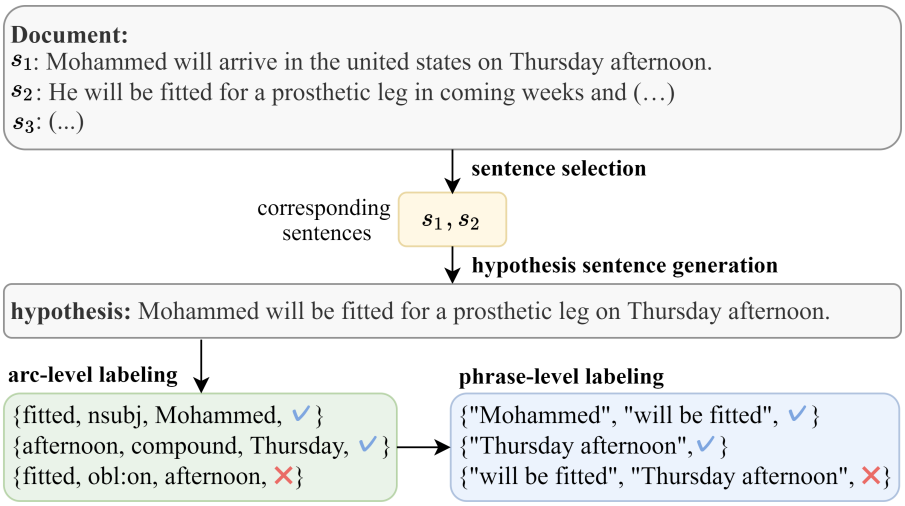

Figure 2: Process for dataset creation. In this example, two source sentences, s_1 and s_2, are selected, and the hypothesis sentence is generated by a sentence fusion operation. The selected sentences are considered error-originating if the hypothesis sentence poses an error. These source sentences are referred to as *corresponding sentences*.

We used the generative model approach, assuming that errors should be contained in the sentences generated with lower probabilities. However, this study differs from (Goyal and Durrett, 2021) in that we generate sentences not only by paraphrasing operations but also by means of sentence fusion and compression operations. This decision can be supported by the insights provided in (Lebanoff et al., 2019b) and (Lebanoff et al., 2019a). The former notes that most sentences in the reference summaries of the CNN/DailyMail dataset (Nallapati et al., 2016) are derived from sentence fusion and compression operations. The latter argues that sentence fusion tends to produce factual errors. Furthermore, we propose raising the error detection level from the dependency-arc level to the phrase level such that we can detect errors that would be overlooked by narrowly focused inspection with dependency-arc level labeling.

We jointly detect errors in summarization and localize their corresponding parts in the input document using multi-task learning, similar to the method presented in (Kryscinski et al., 2020). However, unlike their method, our method can localize more than one corresponding sentence, resolving the limitation of the method in (Kryscinski et al., 2020), which associates only single spans in the input document. This functionality is achieved by formulating the latter task as a multi-label classification problem such that an error in a summary sentence can be associated with multiple corresponding sentences in the source document.

3 Methodology

This section first details the process for creating a synthetic dataset that intentionally accommodates erroneous sentences. Our proposed model for jointly detecting errors in a summary and identifying corresponding sentences in the source document is then described.

3.1 Creation of Synthetic Dataset

3.1.1 Overview of the Creation Process

Figure 2 depicts the proposed process for creating a synthetic dataset that is employed as weak supervision. Sentence pairs ((s_1, s_2) in the example) are first **selected** from the document, and then each of them is **fused** into a sentence (indicated as *hypothesis*) by the generation model. The dependency arcs in the resulting hypothesis sentence are subsequently **labeled** as either "consistent" (blue), "inconsistent" (red), or unlabeled by comparing them with the arcs from the top-ranked hypothesis sentence, selected source sentences, and reference sentences. Thereafter, phrase-level labels (intra- or inter-phrases) are annotated by merging the arc-level labels. Although Figure 2 exemplifies the sentence fusion operation as a means of hypothesis-sentence generation, we also apply the sentence **compression** and **paraphrasing**. However, in the experiments, instead of actually applying sentence selection and paraphrasing, we used ready-made datasets. We performed the phrase-label annotation.

In the following, the dataset is represented as $\{D, h, (p_i, y_i^e)_{p_i \in h}, (p_n, p_m, y_{nm}^e)_{p_n, p_m \in h}, (s_j, y_j^s)_{s_j \in D}\}$, where D and h denote the input document and generated hypothesis sentence, respectively. Moreover, y_i^e denotes the intra-phrase consistency label for the i-th phrase p_i in h, whereas y_{nm}^e dictates the inter-phrase consistency label between p_n and p_m. Furthermore, y_j^s is a label that indicates whether the j-th sentence in D is a corresponding sentence of h.

3.1.2 Sentence Selection

One or two sentences in the input document $D = [s_1, s_2, ..., s_n]$ are selected and fed to each generative model to generate a hypothesis sentence, which is expected to contain factual errors. A pair of sentences is used for sentence fusion, whereas a single sentence undergoes compression and paraphrasing.

In the experiments, we used ready-made datasets instead of the actual sentence selection operation: the sentence fusion dataset (Lebanoff et al., 2020b) and sentence compression dataset (Desai et al., 2020). These datasets were created for the CNN/DailyMail dataset by pairing a reference sentence with the corresponding sentences in a source document that measured the highest ROUGE scores.

3.1.3 Sentence Fusion

We used the Transformer-based Trans-LINKING (Lebanoff et al., 2020a) model to generate a hypothesis sentence. The model employed in the experiments was pre-trained on the same sentence fusion dataset used for sentence selection. As in (Lebanoff et al., 2020a), we adopt only a hypothesis sentence that shared two or more words in both of the originating sentences. In the experiments, we generated hypothesis sentences with a beam size of 10. The most probable and improbable sentences were used in the labeling process.

3.1.4 Sentence Compression

We employed CUPS (Desai et al., 2020) for sentence compression. In the experiments, we used the pre-trained model provided by the authors of CUPS. In general, sentence compression removes redundant or unimportant portions from an input sentence; that is, factual errors rarely emerge. Therefore, we annotated the dependency arcs in the compressed sentence as consistent, provided that they also existed in the input to this model.

3.1.5 Paraphrasing

Any paraphrasing model can be employed for our purpose. In the experiments, we used the dataset made available by (Goyal and Durrett, 2021), which contains paraphrased versions of the selected reference sentences in the CNN/DailyMail dataset. We computed phrase-level labels by referring to the arc-level labels already provided in this dataset.

3.1.6 Labeling

Each generated hypothesis sentence was analyzed using the Stanford CoreNLP (Manning et al., 2014) dependency parser and subsequently underwent the labeling process. The labeling process first examines each of the dependency arcs and assigns either "consistent" or "inconsistent" labels or leaves them unlabeled. This arc-level annotation step is followed by the phrase-level annotation step that assigns labels to phrases and inter-phrase dependency relations. Note that our labeling process is inspired by the method proposed in (Goyal and Durrett, 2020), which assumes that the sentence with the lowest posterior probability may contain more factual errors than sentences with higher probabilities.

We denote an input sentence[2] to the generation model as x and the set of output sentences sorted by posterior probabilities as $H = [h_1, h_2, ..., h_k]$. Furthermore, $d(h)$ denotes a set of dependency arcs of a sentence h. A reference sentence is represented by h^*, and a_i denotes the i-th arc in the corresponding set of dependency arcs.

Arc-level labeling: We follow the method proposed in (Goyal and Durrett, 2020). We select sentence h_k with the lowest probability and assign a label to each arc a_i in h_k as follows.

$$y_i^a = \begin{cases} \text{"consistent"} & \text{if } a_i \in d(x) \cup d(h^*) \\ \text{"unlabeled"} & \text{if } a_i \in d(h_1) \setminus d(x) \cup d(h^*) \\ \text{"inconsistent"} & \text{otherwise} \end{cases} \quad (1)$$

Note that h_1, the sentence with the highest probability, is used as a presumably error-free sentence.

If a_i appears in the arc set of the corresponding original sentence x or that of the reference sentence h^*, we consider it consistent. If a_i appears in $d(h_1)$ but not in x or h^*, we do not assign any labels to the arc, as it is less reliable to determine it as

[2]Remember that the sentence x originates from the source document.

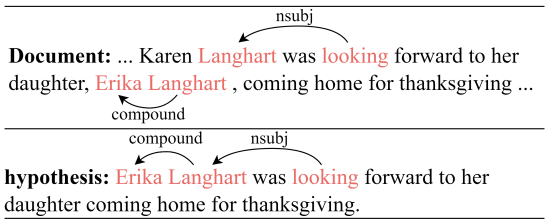

Figure 3: Motivating example for phrase-level labeling.

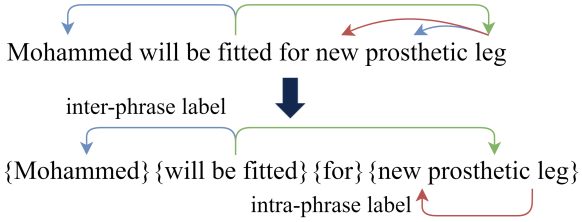

Figure 4: Example of phrase-level labels. The blue and red edges denote consistent and inconsistent labels, respectively, whereas the green edge indicates that the inter-phrase relation is unlabeled.

error-free. As arc a_i not matching either of these conditions may represent a factual error, we assign an inconsistent label to the arc.

We remind the reader that for a compressed sentence, we assign the labels as follows.

$$y_i^a = \begin{cases} \text{“consistent”} & \text{if } a_i \in d(x) \\ \text{“unlabeled”} & \text{otherwise} \end{cases} \quad (2)$$

Note that we did not assign the “inconsistent” label because a compressed sentence usually does not pose any factual error.

As in (Goyal and Durrett, 2020), we exclude particular dependency labels, such as `det` and `case`, from the annotation process, as these do not carry significant meanings.

Phrase-level labeling: Before describing the phrase-level labeling process, we explain its motivation using Figure 3. In this example, the hypothesis sentence contains a factual error because the agent of the verb “looking forward” is “Karen Langhart” and not “Erika Langhart.” This error may be difficult to localize with a model that only considers arc-level labels. Therefore, we assign intra-phrase labels as well as inter-phrase labels, as illustrated in Figure 2, such that factual errors beyond the word-to-word dependency relations can be detected. To initiate the phrase-level labeling process[3], we first recognize phrases by applying a set of heuristic rules that investigate the dependency structure of a sentence. This procedure recognizes a sequence of “phrases” covering the input token sequence without overlaps, where each phrase principally represents a base syntactic phrase.

Figure 4 depicts the intra-and inter-phrase labels, which are computed as follows. If all the dependency arcs that are closed in a phrase are marked as consistent, the intra-phrase label of the phrase is “consistent;” if any is marked as inconsistent, the label is “inconsistent.” An inter-phrase label is computed similarly, but by looking at each of the dependency arcs connecting a word in the head phrase and that in the dependent phrase. Note that the inter-phrase label is “inconsistent” if either of the connected phrases is marked inconsistent. In this example, the phrase “new prosthetic leg” is labeled as “inconsistent” because the particular dependency arc between “new” and “leg” is labeled as “inconsistent.”

3.2 SumPhrase: Proposed Model

Figure 5 illustrates the network architecture of the proposed SumPhrase model. The model consists of three parts: the detection of intra-phrase errors (blue), the detection of inter-phrase errors (red), and the localization of the corresponding sentences in the source document (green).

Training: The inputs to the model are twofold: the source document D and hypothesis sentence h. Tokenized tokens from each input are concatenated using the [CLS] token as the separator and fed to the pre-trained encoder, which enables a contextualized representation for each token along with the special [CLS] token that is expected to represent the hypothesis sentence. Thereafter, the representation of each sentence in D and each phrase p_i is generated using the span attention mechanism (Lee et al., 2017)[4]. The intra- and inter-phrase detection parts are trained by referring to the intra-phrase labels (y_i^e) and inter-phrase labels (y_{ij}^e). The probability of a phrase p_i being consistent is computed as follows, using its representation h_i^p:

$$p(y_i^e|p_i) = \text{softmax}(\text{FFN}_{intra}(h_i^p)). \quad (3)$$

[3]Refer to Appendix A for details of the phrase-level labeling process.

[4]The effectiveness of the span attention mechanism was experimentally confirmed, as described in Appendix B.

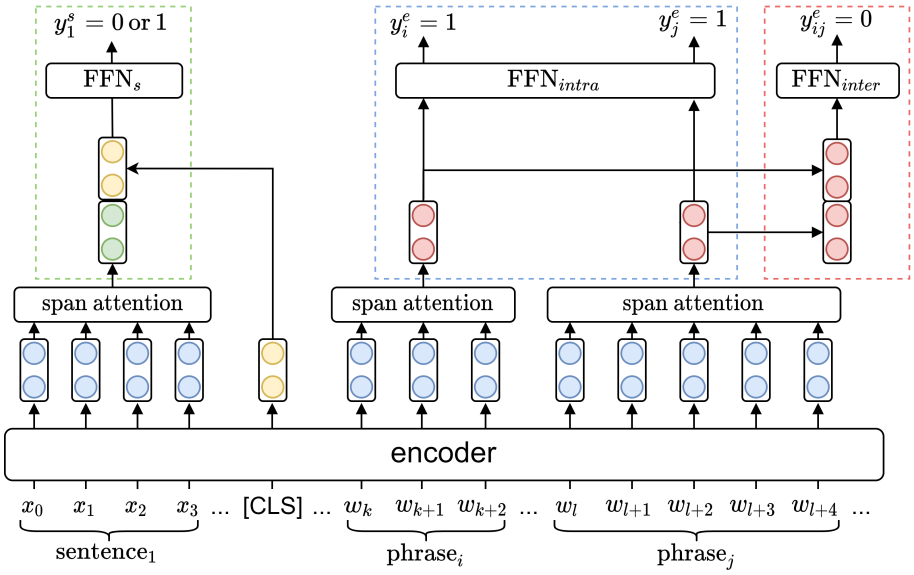

Figure 5: SumPhrase model. The blue and red frames display the intra-phrase and inter-phrase detection parts, respectively, and the green frame indicates the corresponding sentence localization part.

Similarly, the inter-phrase consistency probability is computed as follows:

$$p(y_{ij}^e|p_i, p_j) = \text{softmax}(\text{FFN}_{inter}([h_i^p; h_j^p])). \quad (4)$$

The probability of sentence s_i being a corresponding sentence of the hypothesis sentence is calculated as follows, where h_i^s and h^{cls} denote the representation of s_i and CLS token, respectively.

$$p(y_i^s|s_i) = \text{softmax}(\text{FFN}_s([h_i^s; h^{cls}])) \quad (5)$$

These logits are subsequently converted into losses (Loss_{intra}, Loss_{inter}, and Loss_s) using binary cross-entropy loss. Finally, the entire loss is defined as follows by incorporating a hyperparameter α that adjusts the impact of the corresponding sentence localization (Loss_s), which is, in a sense, introduced as an auxiliary task in the multi-task learning setting.

$$\text{Loss} = \text{Loss}_{intra} + \text{Loss}_{inter} + \alpha * \text{Loss}_s \quad (6)$$

Inference: The source document D and each sentence h in the target summary are fed to the model during the inference time. Note that each sentence in a target summary is dependency-parsed in advance. The intra- and inter-phrase consistencies are assessed using the model for phrases and phrase pairs that were extracted from the summary sentence. We flag a summary sentence as consistent only if it is predicted to exhibit no errors at any level. The corresponding sentences for a hypothesis sentence were identified using multi-label classification. Consequently, the corresponding sentences of a factual error are localized.

Source	# of data
Paraphrasing (para)	46,925
Sentence fusion (fusion)	72,093
Sentence compression (comp)	47,296
Reference sentences (ref)	107,278

Table 1: Statistics of created synthetic dataset.

4 Experimental Setup

The CNN/DailyMail dataset (Nallapati et al., 2016) was used in the experiments.

4.1 Training Dataset

We created a synthetic dataset to train the SumPhrase model based on the methodology described in the previous section. Table 1 lists the number of data instances classified according to the provenance of the data. Note that "paraphrasing" refers to paraphrased data provided by (Goyal and Durrett, 2021), and "Reference sentences" means reference summary sentences obtained from the CNN/DailyMail dataset, which were added to increase the phrases labeled as consistent.

In total, this dataset contains 2,021,592 consistent labels and 191,553 inconsistency labels. More-

Model	Training data	K2020 (BA)	K2020 (F1)	Reranking (% correct)
[Sentence level]				
FactCC	text transformations	72.7	0.706	70.0%
FactCCX	text transformations	72.9	0.711	-
SumFC	text transformations	80.4	-	78.7%
FactAdv	text transformations	73.3	0.701	-
[Arc level]				
Electra-DAE	text transformations	76.7	-	-
Electra-DAE	para	72.1	-	-
Electra-DAE (ours)	para+fusion+comp+ref	82.7	0.754	85.9%
[Phrase level (ours)]				
SumPhrase	para+fusion+comp+ref	**85.3**	**0.765**	**86.0%**
SumPhrase (-multi)	para+fusion+comp+ref	85.2	0.759	84.7%

Table 2: Results of error detection. FactCC, FactCCX (Kryscinski et al., 2020), SumFC (Zhang et al., 2021), and FactAdv (Zeng et al., 2021) employ sentence-level labeling using rule-based text transformations. The Electra-DAE models (Goyal and Durrett, 2021) adopt dependency arc-level labeling. Our SumPhrase models uses phrase-level labeling, where "-multi" dictates the model trained without the auxiliary task of corresponding sentence localization. Refer to Table 1 for para, fusion, comp, and ref in the Training data column.

over, 186,028 source sentences were marked as one of the corresponding sentences of a hypothesis sentence, whereas 3,389,275 other sentences were not labeled as corresponding sentences.

4.2 Training the SumPhrase Model

The Electra-base model (Clark et al., 2020) pretrained on 3.3 billion tokens from Wikipedia and BooksCorpus (the same as the BERT-base) was employed as the encoder of the SumPhrase model in the experiments. We trained the SumPhrase model under the following conditions: number of epochs: 3, batch size: 10, optimizer: AdamW, and initial learning rate: 3e-5. The α parameter in equation-(6) was set to 0.5. We saved the checkpoint that achieved the highest BA on the validation portion of the K2020 dataset. Appendix C provides additional details of the experimental setup.

4.3 Test Datasets and Evaluation Metrics

We used two human-annotated datasets in the evaluation, which are known as K2020 (Kryscinski et al., 2020) and Reranking (Falke et al., 2019).

The K2020 dataset contains 441 consistent sentences and 62 inconsistent sentences. We measured the balanced accuracy (BA) and Macro-F1 for comparisons.

The Reranking dataset includes 373 instances, each containing a source document and two similar sentences: one is a consistent summary sentence extracted by an extractive summarizer, and the other is a potentially inconsistent summary sentence generated by an abstractive summarizer. That is, the reranking task is to correctly (re)rank consistent summary sentences higher. Thus, the portion of correctly ranked data instances is used as the evaluation metric. We also note that the Reranking dataset was used to measure the accuracy of the corresponding sentence localization task that was incorporated as an auxiliary task.

4.4 Compared Existing Models

We primarily compared our phrase-level models with dependency-arc-level Electra DAE models (Goyal and Durrett, 2021), both trained with the proposed dataset. In addition, we compared these models with existing sentence-level models including FactCC/FactCCX (Kryscinski et al., 2020), SumFC (Zhang et al., 2021), and FactAdv (Zeng et al., 2021). These models rely on text-transforming operations to generate erroneous sentences. Among these models, SumFC achieved the best results on both K2020 and Reranking[5].

5 Results and Discussion

This section discusses the efficacy of the proposed method by referring to the results. Appendix D presents and discusses some detection results.

[5]The performance of this model could be further improved by training it with our proposed dataset.

5.1 Detection of Sentence Inconsistency

Table 2 lists the results of inconsistency detection, where existing weakly supervised models that rely on a synthetic dataset are compared.

These results demonstrate that our SumPhrase model outperformed the other models in all metrics. The insights obtained from the results can be summarized as follows:

- The automatically generated training data significantly improved the detection performance (from comparisons of the Electra-DAE (ours) and SumPhrase models with other models).
- The phrase-level labeling was effective (from comparisons of the SumPhrase models with the Electra-DAE models).
- The multi-task learning strategy contributed to improving the performances (from comparisons of the SumPhrase model with the SumPhrase (-multi) model). In principle, the incrrectly localized corresponding sentences could affect the inconsistency detection performance[6]. However, the auxiliary task was accomplished with high accuracy, as shown in Table 3, and the multi-task learning strategy is assessed as adequate.
- Even without multi-task learning, the SumPhrase model exhibited better performances than most models. The exception was the result on Reranking, which was slightly inferior to that of Electra-DAE (ours).

5.2 Localization of Corresponding Sentences

We incorporated this task as an auxiliary task, assuming it would improve the performance of the main task. As shown in Table 2, this assumption is confirmed.

Table 3 summarizes the results of this auxiliary task, where BA and Macro F1 are used as the evaluation metrics. The table shows that our SumPhrase model achieved better results than the FactCCX (Kryscinski et al., 2020) and SumFC (tf-idf) (Zhang et al., 2021) models. The FactCCX model identifies evident spans in a document for a summary sentence, readily enabling the localization of the corresponding sentences. For the SumFC (tf-idf) model, we selected the original sentence that was most similar to a summary sentence by measuring the cosine similarity of the tf-idf vectors.

[6]See the second example presented in Appendix D, where extrinsic hallucination errors arise.

Model	BA	F1
FactCCX	95.9	0.945
SumFC (tf-idf)	97.0	0.970
SumPhrase	**99.2**	**0.980**

Table 3: Results of corresponding sentence localization.

	K2020		Reranking
Training data	BA	F1	%Correct
fusion	81.1	0.663	83.4%
comp	50.0	0.467	54.9%
para	73.4	0.708	78.4%
fusion+comp	82.5	0.687	84.3%
fusion+para	82.8	0.681	84.4%
fusion+comp+para	83.7	0.700	**85.1%**
ALL	**85.2**	**0.759**	84.7%

Table 4: Results of dataset ablation. "ALL" indicates the "fusion+comp+para+ref" condition.

An intriguing insight appears when we compare the SumFC results in Tables 2 and 3. Although the SumFC model achieved almost a level of accuracy (97.0 to 99.2 in BA) in this experiment close to that of our SumPhrase model, there was a significant gap in the inconsistency detection accuracy (78.7% to 86.0% with Reranking). This result indicates that the task of inconsistency detection requires a greater variety of errors in the training dataset, which cannot be achieved with the simple rule-based text transformation approach taken by SumFC and FactCCX, again demonstrating the superiority of our synthetic dataset.

5.3 Data Ablation

Table 4 presents the main results of the dataset ablation study conducted to investigate the contribution of each portion of the dataset or their combinations. Table 7 (Appendix E) provides further details of the ablation results. The SumPhrase model used in these experiments was trained with a single-task learning regime, that is, the SumPhrase (-multi) model.

The first block of Table 4 compares the results with the individual data, showing that sentence fusion data is the most promising for achieving good performances. Even with this dataset portion, our SumPhrase model achieved better results than the other models on K2020 (BA) and Reranking. In

contrast, sentence compression data alone is almost useless, as it contains only consistent labels and rarely introduces factual errors[7].

Based on the effectiveness of the sentence fusion data, the second block of the table presents the results obtained by incrementally adding other dataset portions. Apart from the performance on Reranking, the initial performance with sentence fusion data constantly improved with the addition of data, thereby demonstrating that increasing the variety of data in training is vital.

Also note that the accuracy of K2020 significantly improved by adding reference summary sentences (ALL). This result suggests that the addition of phrases labeled as "consistent" is effective. However , a careful reader may, notice that the ALL model performance on Reranking was slightly inferior to that of the fusion+comp+para model. Although a detailed investigation is required, this could be attributed to the quality of the Reranking dataset. In particular, positive and negative examples required in Reranking were generated using the FAS summarization model (Chen and Bansal, 2018). We suspect that the quality of these examples is not high compared with those generated by humans or SOTA models. Therefore, the ALL setting, which additionally incorporates human-generated reference summaries in the training data, did not yield a better result than the fusion+comp+para setting.

5.4 Limitations of the Present Work

Although the proposed method exhibits excellent performance, it focuses on intrinsic factual errors. Extrinsic hallucination errors, readily introduced with a dataset such as XSum (Narayan et al., 2018; Maynez et al., 2020), are another acute issue to be addressed. A manner of detecting extrinsic hallucination errors is to train a model using generated examples, as proposed by Utama et al. (2022). An alternative would be to incorporate another source of information, such as world knowledge, to validate the content of a summary. Before exploring this method, however, we would identify issues inherent to our approach using the XSum dataset.

6 Conclusions

We propose a neural model called SumPhrase for detecting errors in an abstractive summary. The model was empirically proven to be effective as it outperformed other weakly supervised models. This significant performance is primarily attributed to the dataset on which the model was trained. We created a synthetic dataset that contains factual errors likely to be produced by a common summarizer. These errors are labeled at the phrase level, as opposed to the dependency arc-level labels employed by existing models. The synthetic training dataset can also contribute to improving models that rely on an external NLI or QA system (Laban et al., 2022; Fabbri et al., 2021). It can also be used to fine-tune these models or to post-edit errors (Cao et al., 2020).

The model exhibited improved performance when jointly trained with the sub-task of localizing corresponding sentences in a summary sentence. This functionality may contribute to enhancing the explainability of an inconsistency error-detection system. Our method can also contribute to generating negative samples required to train a summarization model that relies on contrastive learning (Cao and Wang, 2021).

[7]Although not present in the table, the ref data exhibited almost the same results for the same reason.

Acknowledgements

This work was partially supported by JSPS KAKENHI, Grant Number 22K12723.

References

Meng Cao, Yue Dong, Jiapeng Wu, and Jackie Chi Kit Cheung. 2020. Factual error correction for abstractive summarization models. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 6251–6258, Online. Association for Computational Linguistics.

Shuyang Cao and Lu Wang. 2021. CLIFF: Contrastive learning for improving faithfulness and factuality in abstractive summarization. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 6633–6649, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Ziqiang Cao, Furu Wei, Wenjie Li, and Sujian Li. 2018. Faithful to the original: Fact aware neural abstractive summarization. In *AAAI Conference on Artificial Intelligence*.

Yen-Chun Chen and Mohit Bansal. 2018. Fast abstractive summarization with reinforce-selected sentence rewriting. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 675–686, Melbourne, Australia. Association for Computational Linguistics.

Kevin Clark, Minh-Thang Luong, Quoc V. Le, and Christopher D. Manning. 2020. ELECTRA: Pre-training text encoders as discriminators rather than generators. In *ICLR*.

Shrey Desai, Jiacheng Xu, and Greg Durrett. 2020. Compressive summarization with plausibility and salience modeling. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 6259–6274, Online. Association for Computational Linguistics.

Esin Durmus, He He, and Mona Diab. 2020. FEQA: A question answering evaluation framework for faithfulness assessment in abstractive summarization. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 5055–5070, Online. Association for Computational Linguistics.

Alexander R Fabbri, Chien-Sheng Wu, Wenhao Liu, and Caiming Xiong. 2021. Qafacteval: Improved qa-based factual consistency evaluation for summarization. *arXiv preprint arXiv:2112.08542*.

Tobias Falke, Leonardo F. R. Ribeiro, Prasetya Ajie Utama, Ido Dagan, and Iryna Gurevych. 2019. Ranking generated summaries by correctness: An interesting but challenging application for natural language inference. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 2214–2220, Florence, Italy. Association for Computational Linguistics.

Tanya Goyal and Greg Durrett. 2020. Evaluating factuality in generation with dependency-level entailment. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 3592–3603, Online. Association for Computational Linguistics.

Tanya Goyal and Greg Durrett. 2021. Annotating and modeling fine-grained factuality in summarization. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 1449–1462, Online. Association for Computational Linguistics.

Wojciech Kryscinski, Bryan McCann, Caiming Xiong, and Richard Socher. 2020. Evaluating the factual consistency of abstractive text summarization. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 9332–9346, Online. Association for Computational Linguistics.

Philippe Laban, Tobias Schnabel, Paul N. Bennett, and Marti A. Hearst. 2022. SummaC: Re-visiting NLI-based models for inconsistency detection in summarization. *Transactions of the Association for Computational Linguistics*, 10:163–177.

Logan Lebanoff, Franck Dernoncourt, Doo Soon Kim, Lidan Wang, Walter Chang, and Fei Liu. 2020a. Learning to fuse sentences with transformers for summarization. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 4136–4142, Online. Association for Computational Linguistics.

Logan Lebanoff, John Muchovej, Franck Dernoncourt, Doo Soon Kim, Seokhwan Kim, Walter Chang, and Fei Liu. 2019a. Analyzing sentence fusion in abstractive summarization. In *Proceedings of the 2nd Workshop on New Frontiers in Summarization*, pages 104–110, Hong Kong, China. Association for Computational Linguistics.

Logan Lebanoff, John Muchovej, Franck Dernoncourt, Doo Soon Kim, Lidan Wang, Walter Chang, and Fei Liu. 2020b. Understanding points of correspondence between sentences for abstractive summarization. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics: Student Research Workshop*, pages 191–198, Online. Association for Computational Linguistics.

Logan Lebanoff, Kaiqiang Song, Franck Dernoncourt, Doo Soon Kim, Seokhwan Kim, Walter Chang, and Fei Liu. 2019b. Scoring sentence singletons and pairs for abstractive summarization. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 2175–2189, Florence, Italy. Association for Computational Linguistics.

Kenton Lee, Luheng He, Mike Lewis, and Luke Zettlemoyer. 2017. End-to-end neural coreference resolution. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 188–197, Copenhagen, Denmark. Association for Computational Linguistics.

Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Veselin Stoyanov, and Luke Zettlemoyer. 2020. BART: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7871–7880, Online. Association for Computational Linguistics.

Christopher Manning, Mihai Surdeanu, John Bauer, Jenny Finkel, Steven Bethard, and David McClosky. 2014. The Stanford CoreNLP natural language processing toolkit. In *Proceedings of 52nd Annual Meeting of the Association for Computational Linguistics: System Demonstrations*, pages 55–60, Baltimore, Maryland. Association for Computational Linguistics.

Joshua Maynez, Shashi Narayan, Bernd Bohnet, and Ryan McDonald. 2020. On faithfulness and factuality in abstractive summarization. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 1906–1919, Online. Association for Computational Linguistics.

Anshuman Mishra, Dhruvesh Patel, Aparna Vijayakumar, Xiang Lorraine Li, Pavan Kapanipathi, and Kartik Talamadupula. 2021. Looking beyond sentence-

level natural language inference for question answering and text summarization. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 1322–1336, Online. Association for Computational Linguistics.

Ramesh Nallapati, Bowen Zhou, Cicero dos Santos, Çağlar Gu'lçehre, and Bing Xiang. 2016. Abstractive text summarization using sequence-to-sequence RNNs and beyond. In *Proceedings of The 20th SIGNLL Conference on Computational Natural Language Learning*, pages 280–290, Berlin, Germany. Association for Computational Linguistics.

Shashi Narayan, Shay B. Cohen, and Mirella Lapata. 2018. Don't give me the details, just the summary! topic-aware convolutional neural networks for extreme summarization. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 1797–1807, Brussels, Belgium. Association for Computational Linguistics.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J. Liu. 2020. Exploring the limits of transfer learning with a unified text-to-text transformer. *Journal of Machine Learning Research*, 21(140):1–67.

Thomas Scialom, Paul-Alexis Dray, Sylvain Lamprier, Benjamin Piwowarski, Jacopo Staiano, Alex Wang, and Patrick Gallinari. 2021. QuestEval: Summarization asks for fact-based evaluation. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 6594–6604, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Prasetya Utama, Joshua Bambrick, Nafise Moosavi, and Iryna Gurevych. 2022. Falsesum: Generating document-level NLI examples for recognizing factual inconsistency in summarization. In *Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 2763–2776, Seattle, United States. Association for Computational Linguistics.

Alex Wang, Kyunghyun Cho, and Mike Lewis. 2020. Asking and answering questions to evaluate the factual consistency of summaries. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 5008–5020, Online. Association for Computational Linguistics.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Remi Louf, Morgan Funtowicz, Joe Davison, Sam Shleifer, Patrick von Platen, Clara Ma, Yacine Jernite, Julien Plu, Canwen Xu, Teven Le Scao, Sylvain Gugger, Mariama Drame, Quentin Lhoest, and Alexander Rush. 2020. Transformers: State-of-the-art natural language processing. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 38–45, Online. Association for Computational Linguistics.

Zhiyuan Zeng, Jiaze Chen, Weiran Xu, and Lei Li. 2021. Gradient-based adversarial factual consistency evaluation for abstractive summarization. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 4102–4108, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Jingqing Zhang, Yao Zhao, Mohammad Saleh, and Peter Liu. 2020. Pegasus: Pre-training with extracted gap-sentences for abstractive summarization. In *International Conference on Machine Learning*, pages 11328–11339. PMLR.

Sen Zhang, Jianwei Niu, and Chuyuan Wei. 2021. Fine-grained factual consistency assessment for abstractive summarization models. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 107–116, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Zheng Zhao, Shay B. Cohen, and Bonnie Webber. 2020. Reducing quantity hallucinations in abstractive summarization. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 2237–2249, Online. Association for Computational Linguistics.

A Details of Phrase-Level Labeling

The phrase-level labels in a hypothesis sentence $h = \{w_1, ..., w_H\}$ are automatically annotated using the algorithm detailed in Algorithm 1. The first half of the algorithm creates "phrases" by inspecting particular dependency labels $L = \{l^a_1, ..., l^a_K\}$, as shown in Table 5. The second half of the algorithm assigns intra- and inter-phrase labels to each created phrase.

The inputs to the algorithm are a set of dependency labels $Y^a = \{a_1, ..., a_N\}$ and $a_n = \{w_i, w_j, l^a_n, y^a_n\}$. In this case, l^a_n denotes the dependency relation between words w_i and w_j, and y^a_n represents the corresponding dependency arc-level label assigned by the method proposed by (Goyal and Durrett, 2020).

advmod, amod, aux, compound
det, fixed, flat, goeswith, nummod
reparandum, nmod:poss, nmod:tmod

Table 5: Dependency labels used in phrase creation.

B Effectiveness of Span Attention Mechanism

Table 6 compares the performances of the SumPhrase models with and without the span attention mechanism (Lee et al., 2017) on the K2020 dataset. We used the averaged token vectors to represent phrases for the "without" condition. Given this result, we preferred the span attention mechanism over conventional average pooling, because it better captures the meaning of a semantically significant word.

Model	BA	F1
SumPhrase	**85.2**	**0.759**
SumPhrase (average)	83.8	0.746

Table 6: Effectiveness of span attention.

C Details of Experimental Setup

We used Stanford CoreNLP[8] (Manning et al., 2014) with "EnhancedDependenciesAnnotation" to dependency-parse hypothesis sentences. As the encoder for the SumPhrase model, we employed the electra-base-discriminator[9], implemented in the Huggingface Transformers library (Wolf et al., 2020). All experiments were conducted using an NVIDIA GeForce GTX TITAN X GPU with 12 GB of memory. The time required to train the SumPhrase model was approximately 40 h. For reliability, each number reported from our implementations in the present study is the average of three runs with different random seeds.

D Case Study

Figure 6 shows the results of the inconsistency detection using SumPhrase and FactCCX.

Document #1 shows a case in which the SumPhrase model correctly detects an inconsistent sentence and localizes the corresponding sentences. FactCCX failed to detect the inconsistency in the summary, although it identifyied the corresponding spans in the input. This result demonstrates that inconsistency detection is generally more complicated than sentence alignment.

Document #2 presents a typical case in which even the SumPhrase model cannot detect the inconsistency in a summary. This example poses the issue of extrinsic hallucination errors, which are difficult to detect using only the proposed methodology.

E More on Data Ablation

Table 7 lists the results for other data combinations. Again, the effectiveness of sentence fusion data compared with paraphrasing data is shown. Note that adding reference data to fusion or paraphrasing dataset portions degrades the BA, highlighting the efficacy of the ALL (fusion+comp+para+ref) combination.

	K2020		Reranking
Training data	BA	F1	%Correct
fusion+comp	82.5	0.687	84.3%
fusion+ref	79.5	0.716	**85.2%**
para+comp	69.3	0.702	80.3%
para+ref	65.5	0.691	77.7%
ALL	**85.2**	**0.759**	84.7%

Table 7: Additional results of dataset ablation.

[8] https://nlp.stanford.edu/software/stanford-corenlp-4.1.0.zip

[9] https://huggingface.co/google/electra-base-discriminator

Algorithm 1 Phrase-level labeling

Require: (1) The set of dependency arcs and dependency arc-level labels $Y^a = \{a_1, ..., a_N\}$, $a_n = \{w_i, w_j, l_n^a, y_n^a\}$ from a hypothesis sentence h; (2) a set of dependency labels $L = \{l_1^a, ..., l_K^a\}$.
Ensure: A set of phrase-level labels Y^e.

▷ create phrases
$P \leftarrow \{\{w_1\}, ..., \{w_H\}\}$
for $a_i \in Y^a$ **do**
  if $l_i^a \in L$ **then**
    $p_n, p_m \leftarrow$ GET_PHRASE(P, a_i)
    if $p_n \neq p_m$ **then**
      $P \leftarrow$ MERGE_PHRASES(P, p_n, p_m)
    end if
  end if
end for
▷ assign phrase-level labels by merging arc-level labels
$Y^e \leftarrow \emptyset$
for $a_i \in Y^a$ **do**
  $p_n, p_m \leftarrow$ GET_PHRASE(P, a_i)
  if $p_n == p_m$ **then**
    $y_n^e \leftarrow$ GET_LABEL$(Y^e, \{p_n\})$
    $y_n^e \leftarrow$ UPDATE_LABEL(y_n^e, y_i^a)
    $Y^e \leftarrow$ UPDATE$(Y^e, \{\{p_n\}, y_n^e\})$
  else
    $y_{nm}^e \leftarrow$ GET_LABEL$(Y^e, \{p_n, p_m\})$
    $y_{nm}^e \leftarrow$ UPDATE_LABEL(y_{nm}^e, y_i^a)
    $Y^e \leftarrow$ UPDATE$(Y^e, \{\{p_n, p_m\}, y_{nm}^e\})$
  end if
end for
for $\{p, y^e\} \in Y^e$ **do**
  if LEN(p) == 2 **then**
    $y_i^e \leftarrow$ GET_LABEL$(Y^e, \{p[0]\})$
    $y_j^e \leftarrow$ GET_LABEL$(Y^e, \{p[1]\})$
    if $y_i^e ==$ inconsistent or $y_j^e ==$ inconsistent **then**
      $Y^e \leftarrow$ UPDATE$(Y^e, \{p,$ inconsistent$\})$
    end if
  end if
end for

Creating phrases: The algorithm initially assumes that each word in h individually forms a phrase and passes through the elements a_i in Y^a. If the dependency relation of arc a_i is an element presented in L, phrases p_n and p_m are retrieved by GET_PHRASE for the words in this dependency relation. If p_n differs from p_m, these phrases are merged using MERGE_PHRASES.

Assigning phrase-level labels: The algorithm passes through elements a_i in Y^a to accumulate the phrase-level labels in Y^e. First, the phrases of the words in the corresponding dependency relation, that is, p_n and p_m, are retrieved by GET_PHRASE. If these phrases denote identical phrases, an intra-phrase label y_n^e is assigned to p_n. Otherwise, the inter-phrase label y_{nm}^e is assigned to the pair p_n and p_m. Y^e is updated accordingly. Finally, the inter-phrase label is "inconsistent" if either of the connected phrases is markedly inconsistent.

The GET _LABEL, UPDATE _LABEL, and UPDATE procedures employed in the algorithm are summarized as follows:

GET_LABEL: This procedure returns the intra- or inter-phrase label from Y^e, depending on the second argument. None of the labels is returned if the corresponding label is not in Y^e.

UPDATE_LABEL: This updates and returns the label of p_n in Y^e with y_i^a if y_i^a is ranked higher than p_n in the designated priority order: inconsistent $>$ unlabeled $>$ consistent $>$ None.

UPDATE: This procedure updates the set of phrase-level labels y^e with the result of UPDATE_LABEL.

Document #1: Garissa, Kenya (CNN) Kenyan police have arrested five suspects in connection with Thursday's attack at Garissa university college (…) Thursday's attack by Al-Shabaab militants killed 147 people, including 142 students, three security officers and two university security personnel. The attack left 104 people injured, including 19 who are in critical condition, Nkaissery said. During search and recovery efforts on Friday, CNN witnessed one male who was not a student hiding under a bed. (…)
Summary: 147 people, including 142 students, are in critical condition.
FactCCX: consistent
SumPhrase: inconsistent
Human: inconsistent

Document #2: (CNN) the question : how can i know if my food is safe to eat after a specific product recall ? The answer: many of us shed a few tears over the recent sabra hummus recall (even though we are perfectly capable of making our own at home), but that sadness quickly transformed into anxiety when we looked inside our refrigerators and saw the potentially tainted culprit sitting there on the shelf. To assuage any fears, we asked John Swartzberg, m.d., a clinical professor at the university of California at Berkeley, to walk us through the process of determining if our favorite dip was still safe to eat. Related : Amy's kitchen recalls more than 70 , 000 cases of food due to fear of listeria contamination.
Summary: fda recommends anyone who has consumed a listeria-laden food should let their physician know.
FactCCX: consistent
SumPhrase: consistent
Human: inconsistent

Figure 6: Comparison of FactCCX and SumPhrase outputs. The sentence with a blue underline was identified as an error-corresponding sentence by SumPhrase, whereas the span with a red underline was localized by FactCCX. The dependency relation between the red phrases was determined as erroneous by SumPhrase.

PoliSe: Reinforcing Politeness using User Sentiment for Customer Care Response Generation

Mauajama Firdaus[1]*†, **Asif Ekbal**[2] **and Pushpak Bhattacharyya**[3]
[1] Department of Computing Science, University of Alberta, Canada
[2]Department of Computer Science and Engineering, Indian Institute of Technology Patna, India
[3]Department of Computer Science and Engineering, Indian Institute of Technology Bombay, India
[1]mauzama.03@gmail.com; [2]asif@iitp.ac.in; [3]pushpakbh@gmail.com

Abstract

The interaction between a consumer and the customer service representative greatly contributes to the overall customer experience. Therefore, to ensure customers' comfort and retention, it is important that customer service agents and chatbots connect with users on social, cordial and empathetic planes. In the current work, we automatically identify the sentiment of the user and transform the neutral responses into polite responses conforming to the sentiment and the conversational history. Our technique is basically a reinforced multi-task network- the primary task being 'polite response generation' and the secondary task being 'sentiment analysis'- that uses a Transformer based encoder-decoder. We use sentiment annotated conversations from Twitter as the training data. The detailed evaluation shows that our proposed approach attains superior performance compared to the baseline models.

1 Introduction

Human-machine interactions have increased rapidly assisting humans in their everyday lives. With the growth in Artificial Intelligence (AI) and Natural Language Processing (NLP), chatbots and personal assistants, such as Microsoft's Cortana, Apple's Siri, etc., have predominantly become a part of our daily lives. Thus, research in recent years has been on modulating biases, styles, and control in text generation to enhance these interactions.

Customer care is an essential tool used by companies to provide guidance, and assistance and in building stable customer relations. The ease of access, ease of following up, and immediacy of social media has made it a strong platform for companies and applications to interact with their customers. In this regard, conversational agents such as shopping agents, customer service agents, and personal assistants have become extremely famous for handling the various queries of a customer and providing suitable responses to them.

Assisting the customer through social media channels is attaining high popularity. The main reason behind this is the fact that the important elements of social media are its immediacy, transparency, ease of following up, and providing a human-like feel to the company/brand. In this platform, we see the usage of polite and emphatic language, which is the center of our current study.

For the growth of any company or application, customer care agents must be cordial and amicable to the customer. Thus along with handling queries, the agents need to provide customer satisfaction by greeting, empathizing, appreciating feedback, apologizing at the right time, and thus building a strong relationship with the customer.

Recently, incorporating politeness in responses has been investigated (Golchha et al., 2019; Madaan et al., 2020; Niu and Bansal, 2018) to make the dialogue agent more human-like and interactive. Also, the sentiments of the user are important for properly addressing the customer needs (Shi and Yu, 2018; Firdaus et al., 2022b, 2021) to assist in creating smooth and cordial conversations. Therefore, for politely responding to the user it is important to have knowledge about user sentiment to avoid frustrating experiences and help build better systems.

The usage of the user feedback in the form of sentiments is crucial to get contextually correct polite responses as presented in Table 1. For example, if the user has a negative sentiment towards the customer care system, then the possible polite response should be towards apology, assurance, and empathy rather than greet or appreciation. For the first example, the sentiment of the user is *negative*,

*The work was done while the author was in IIT Patna

†This work is licensed under a Creative Commons Attribution 4.0 International Licence. Licence details: http://creativecommons.org/licenses/by/4.0/

Proceedings of the 29th International Conference on Computational Linguistics, pages 6165–6175
October 12–17, 2022.

Dialog Context	User Sentiment	Generic Response	Polite Response	Polite Behaviour
Hey, i got food poisoning from your inflight meal on sunday	Negative	Send us a dm	That's disappointing to hear, we are sorry please send us a dm.	Apology
I need the software update urgently, the battery lasts literally half a day	Negative	How can we help?	Don't worry, we are here for you, please say how can we help?	Assurance
Dear this new update is awesome, got great new apps!	Positive	The update has many features.	Thank you very much, please checkout the exciting features in the update.	Appreciation
Order 2 zinger box meals n got free popcorn chicken, yayyyy	Positive	Enjoy your meal.	That's nice to hear, enjoy your meal.	Acknowledge
How do i go about getting a monthly ride pass ?	Neutral	We have send the link	Hello, good morning we have send the link.	Greet

Table 1: Examples of polite responses in accordance to the user sentiments

therefore politeness in the form of appreciation, greeting or acknowledgment could lead to a wrong response making the customer angrier towards the customer care agent.

Similarly, if the user has a positive sentiment towards the customer care application, then the customer care agent should be able to converse politely showing appreciation and greeting. Previously, researchers focused on changing the sentiment of a given sentence (Li et al., 2018) using style transfer techniques. Here, we use sentiment as feedback for generating polite responses. Therefore, we propose the task of sentiment-guided polite response generation. To the best of our knowledge, this is one of the first works that jointly predict the sentiments from the user utterances and uses the predicted sentiment as feedback for generating polite responses.

Due to the unavailability of sentiment-annotated customer care conversations, we annotate the Courteously Yours Customer Care Dataset (CYCCD) (Golchha et al., 2019) with sentiment labels. To address the task of sentiment-guided polite response generation we design an end-to-end framework that identifies the sentiment of the user and uses the sentiment knowledge for generating polite responses.

We employ a hierarchical transformer network that captures the utterance as well as contextual information for generating responses. While we utilize a BERT-based classifier for predicting the sentiments. For incorporating politeness we use task-specific rewards that reinforce polite behavior in the generated customer care responses.
The key contributions of our current work are:

- We propose the task of generating polite responses in accordance with user sentiments.
- We develop a multi-task end-to-end hierarchical network to identify the sentiments and use the predicted information for generating the contextually correct polite responses.
- We design task-specific rewards that assist in the proposed task and ensure that the content is preserved while incorporating correct and interactive polite responses.

2 Related Work

Natural language generation (NLG) module has been gaining importance in several applications such as dialogue systems (Vinyals and Le, 2015; Shen et al., 2018; Wu et al., 2018; Serban et al., 2017a; Zhang et al., 2018; Li et al., 2016), question answering systems (Reddy et al., 2017; Duan et al., 2017), and many other natural language interfaces. To help the users achieve their desired goals, response generation provides the medium through which a conversational agent can communicate with its user.

In (Serban et al., 2017b), the authors have proposed a hierarchical encoder-decoder model for capturing the dependencies in the utterances of a dialogue. Emotion classification and analysis (Herzig et al., 2016) in customer support dialogue is important for a better understanding of the customer and to provide better customer support. Lately, several works have been carried out on controlled text generation (Hu et al., 2017; Li et al., 2017; Subramanian et al., 2017; Fedus et al., 2018; Peng et al., 2018) to generate responses with desired attributes.

Style transfer has been an emerging field in natural language processing (NLP). In (Rao and Tetreault, 2018), a dataset has been introduced for formality style transfer. Unsupervised text style transfer has encouraged in transforming a given text without parallel data (Shen et al., 2017; Carlson et al., 2017; Fu et al., 2018; Li et al., 2018; Niu and Bansal, 2018). One of the early works in politeness (Gupta et al., 2007) was based upon

making the conversational agents more affective and socially intelligent by incorporating different politeness strategies in the responses.

Recently in (Niu and Bansal, 2018), the authors proposed a neural framework that could induce politeness in chit-chat conversations in the absence of parallel data. Lately, (Golchha et al., 2019) presented a method for increasing user satisfaction by inducing courteous phrases in the customer-care responses by exploiting reinforced pointer networks.

One of the recent studies presented in (Madaan et al., 2020) devised a tag and generate framework for converting non-polite sentences into polite ones. Recent works on politeness focuses on modeling politeness across languages (Firdaus et al., 2020), gender and age group (Firdaus et al., 2022a), predicting variations in politeness (Mishra et al., 2022b) and building a politeness adaptive system (Mishra et al., 2022a).

From the existing literature on politeness, we can conclude that politeness in conversational agents is essential for increasing the social and affective understanding of conversational agents. Our current work differs from the existing baselines as we include user feedback in the form of sentiment information to enhance the quality of generation and to make the responses contextually coherent with the dialog.

3 Methodology

In this section, we formally define the problem statement and give a detailed description of our proposed methodology.

3.1 Problem Formulation

In our current work, we address the task of identifying the sentiments from the user utterances and using the sentiment information to transform the generic customer care responses into polite responses which are contextually appropriate to the dialog history and the user sentiments.

Precisely, given the dialog history D having a sequence of sentences $(s_1, s_2, \ldots, s_N)$ where each sentence s_n is a sequence of words $u_1, u_2, \ldots, u_M$, the task is to simultaneously identify the sentiments from the user sentence $s_{(N-1)}$ and utilize the detected sentiment to transform the generic customer care s_N response into a polite response s'_N.

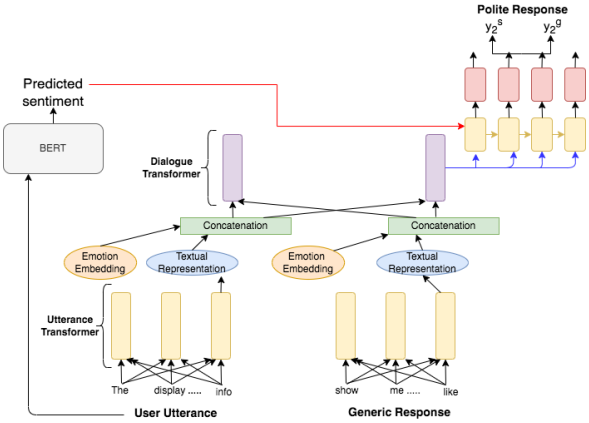

Figure 1: Architectural diagram of our proposed multi-task framework that simultaneously identifies the sentiment and generates the polite response in accordance with the predicted sentiment

3.2 Methodology

Our proposed network is based upon the Transformer Encoder-Decoder (TED) architecture (Vaswani et al., 2017) as shown in Figure 1. We utilize a hierarchical transformer having two encoders: one of which is used to encode the sentences named as sentence encoder while we use another Transformer to encode the output of the sentence encoder to capture the dialog context. We apply the softmax activation function on the sentence encoder to capture the sentiment information from the user utterance. The predicted sentiment information along with the contextual information is used to initialize the Transformer decoder.

We design task-specific rewards to ensure that the users' sentiments and politeness are induced appropriately in the generated responses. In our present work, we focus on the sentiment of the user and the way it affects the politeness quotient of the customer care agent/bot response. For this, we consider the emotional information of the user that, in turn, helps in identifying the correct sentiment of the user. As we predict the sentiment of the user, the customer care agent's emotional information is provided as shown in Figure 1 for generating consistent politeness, emotion, and sentiment-aware responses.

Sentence Encoder. As the transformer encoder has multiple layers and each layer is composed of a multi-head self attentive sub-layer followed by a feed-forward sub-layer with residual connections (He et al., 2016) and layer normalization (Ba et al., 2016), we use it to encode the sentences in a given dialog. For intricate details on the Trans-

former network, we refer the interested readers to (Vaswani et al., 2017). To learn the representation of s_n, $s_n = (u_{k,1}, u_{k,2}, ..., u_{k,n'})$ is first mapped into continuous space

$$T_u = (t^i_1, t^i_2, \ldots, t^i_{|s_n|}); where[T^i_j = e(w^i_j) + p_j] \quad (1)$$

where $e(u^i_j)$ and p_j are the word and positional embedding of every word u^i_j in an utterance, respectively. For words, we use Glove embeddings and adopt the sine-cosine positional embedding (Vaswani et al., 2017) as it performs better and does not introduce additional trainable parameters.

The utterance encoder (a Transformer) converts T_u into a list of hidden representations $h^i_1, h^i_2, \ldots, h^i_{|s_n|}$. We use the last hidden representation $h^i_{|s_n|}$ (i.e. the representation at the EOS token) as the textual representation of the utterance s_n. Similarly, to the representation of each word in s_n, we also take into account the utterance position. Therefore, the final textual representation of the utterance s_n is:

$$h^s_i = h^i_{|s_n|} + p_i \quad (2)$$

Note that the words and sentences share the same positional embedding matrix. We also capture the emotional embeddings of every sentences using the output distribution from DeepMoji (Felbo et al., 2017) which is pre-trained on the emoji prediction task in a similar manner as (Golchha et al., 2019).

The emotional embedding of every sentence is represented as $e_{s,n}$. The final representation of any sentence is given by the concatenation of the emotional representation as well as the last hidden representation of the sentence.

$$h^{sen}_i = h^s_i + e_{s,n} \quad (3)$$

Context Encoder. The context encoder is yet another Transformer, but it is applied on the utterance level. After running the transformer on the sequence of sentence representation $h^{sen}_1, ^{sen}_2, \ldots, h^{sen}_{|D|}$, we obtain the context-sensitive utterance representations $\hat{D} = (\hat{d}_1, \hat{d}_2, \ldots, \hat{d}_{|D|})$.

After achieving the representation of a given dialogue using a hierarchical transformer network, we employ a transformer decoder to generate the polite response in accordance to the contextual information and the sentiment information.

Sentiment Classification Network. For identifying sentiments, we propose a sentiment classification network. The input to this network is the user utterance and the output is a predicted sentiment for this utterance. A BERT classifier is applied on the user utterance followed by a softmax output layer which gives the sentiment prediction. The sentiment classifier is trained by minimizing the negative log-likelihood

$$\mathcal{L}_{SE} = -\sum_{se=1}^{N} y_{se} \log \tilde{y_{se}} \quad (4)$$

Decoder: To generate the polite response with the predicted sentiment, we employ a Transformer decoder (Vaswani et al., 2017) as shown in Figure 1. The Transformer decoder used in our current work is slightly different from the original in which two multi-head attention layers were employed to incorporate both the contexts in encoder and decoder.

But in our case, we only need one to learn the decoder context, since the context in encoder is a vector (i.e., $\hat{d}_i$). We predict the polite responses $Y_k = (y^k_0, y^k_1, y^k_2, \ldots, y^k_{|Y_k|})$ one word per step (y^k_0) is an artificially added BOS (beginning of sentence token). At the jth step, we predict y^k_j given $y^k_0, \ldots, y^k_{j-1}$, predicted sentiment SE_i and the context representation $\hat{D}$. By applying word and positional embeddings to $(y^k_0, \ldots, y^k_{j-1})$, we obtain $\tilde{E}^k_{1:j-1} = (\tilde{e^k_0}, \ldots, \tilde{e^k_{j-1}})$. Then, we apply multi-head attention sub-layer to $\tilde{E}^k_{1:j-1}$:

$$\begin{aligned} \tilde{h_{j-1}} = MultiHead(q_{j-1}, K_{j-1}, V_{j-1}); \\ q_{j-1} = W^Q \tilde{e^k_{j-1}}; \\ K_{j-1} = W^K \tilde{E}^k_{1:j-1}; \\ V_{j-1} = W^V \tilde{E}^k_{1:j-1} \end{aligned} \quad (5)$$

where q_{j-1}, K_{j-1}, V_{j-1} are the input query, key and value matrices of the multi-head attention function (Vaswani et al., 2017), respectively. Also, W^Q, W^K and W^V are the weight matrices. To attain polite responses in accordance to the sentiment, we include the information of $\hat{D}$ and sentiment SE_i by addition:

$$\tilde{x_{j-1}} = \tilde{h_{j-1}} + \hat{d}_k + SE_i \quad (6)$$

We also apply a feedforward sub-layer (one hidden layer with ReLU (Glorot et al., 2011) activation function) after $\tilde{x_{j-1}}$ in a similar manner as (**?**):

$$\tilde{g_{j-1}} = W^{ff}_2 max(0, W^{ff}_1 \tilde{x_{j-1}} + b_1) + b_2 \quad (7)$$

Note that the transformer decoder can have multiple layers by applying Equation (4) to (6) multiple times but we show only the computation of one layer for the simplicity. The probability of y_j^k given $y_0^k, \ldots, y_{j-1}^k$, SE_i and the context representation $\hat{D}$ is:

$$p(y_j^k | y_0^k, \ldots, y_{j-1}^k, \hat{d}_k, SE_i) = softmax(W^O g_{\tilde{j-1}}) \quad (8)$$

Model Training: As used in (Paulus et al., 2017), we jointly use reinforcement learning (RL) and machine learning (ML) to train our model in a similar manner as (Golchha et al., 2019). If $\tilde{y} = \{\tilde{y}_1, \tilde{y}_2, \ldots, \tilde{y}_{n'}\}$ is the gold output tokens for the given generic response tokens h_i^{sen} and conversation history $\hat{D}$, the maximum-likelihood objective using teacher forcing is given by:

$$L_{MLE} = -\sum_{t=1}^{n'} \log p(\tilde{y}_t | \tilde{y}_1, \ldots, \tilde{y}_{t-1}, h_i^{sen}, \hat{D}) \quad (9)$$

In addition to maximum likelihood error training, we also use RL to learn from maximising discrete metrics that are task-specific (which we design as the rewards). We use the self-critical policy gradient algorithm suggested in (Rennie et al., 2017) for training the network.

The reward obtained by the inference time algorithm (which performs greedy decoding) is baselined for the REINFORCE (Williams, 1992) algorithm, without the need for training a "critic" network for estimating the value functions. During training, two output sequences are produced: y^s, obtained by sampling $p(y_t^s | y_1^s, \ldots, y_{t-1}^s, x)$ probability distribution, and y^g, the baseline output, obtained by greedily maximizing the output probability distribution at each time step.

$$L_{RL} = (r(y^g) - r(y^s)) \sum_{t=1}^{n'} \log p(y_t^s | y_1^s, \ldots, y_{t-1}^s, h_i^{sen}, \hat{D}) \quad (10)$$

Our reward function $r(y)$, used for evaluating y against the gold standard output is

$$r(y, \tilde{y}) = \lambda_1 \cdot r1(y, \tilde{y}) + \lambda_2 \cdot r2(y, \tilde{y}) + \lambda_3 \cdot r3(y, \tilde{y}) \quad (11)$$

The final reward function is the weighted mean of the three terms as given below:

(i). BLEU metric $r1$: Ensures the content matching between the generated response and the ground-truth response.

(ii). Sentiment consistency $r2$: Measured by the cosine similarity of the sentiment prediction distributions of the user utterance and generated responses (using pre-trained BERT classifier). It ensures that the sentiment states of the generated polite response is consistent with the user sentiment.

(iii). Politeness accuracy $r3$: The politeness accuracy of the generated response is computed using the pre-trained BERT based politeness classifier (trained on Stanford Politeness Corpus (Danescu-Niculescu-Mizil et al., 2013)). The responses having scores greater than 0.7 are considered as polite.

We first pre-train using the maximum likelihood (ML) objective and then using a mixed objective function with a reduced learning rate:

$$\mathcal{L}_{gen} = \eta L_{RL} + (1 - \eta) L_{MLE}, \quad (12)$$

Joint Training (JT): We jointly train the entire model by simultaneously minimizing the sentiment classification loss and generation loss. The final loss of the model is:

$$\mathcal{L}_{Joint} = \mathcal{L}_{SE} + \mathcal{L}_{gen} \quad (13)$$

3.3 Baseline Models:

To demonstrate the effectiveness of our proposed model, we compare with the previous state-of-the-art (SoTA) models:

Seq2Seq: It is the standard encoder-decoder framework with attention mechanism that has been widely used in generation, machine translation etc. (Sutskever et al., 2014).

HRED: It is a hierarchical encoder-decoder model proposed for text based dialogue systems (Serban et al., 2015).

Polite-RL: We implement the Polite-RL framework to induce politeness in responses in a similar manner as (Niu and Bansal, 2018).

PT-TGA: We implement the politeness transfer framework presented in (Madaan et al., 2020) that uses a tag and generate approach to incorporate politeness.

PG-RL: We also take the reinforced pointer generator network employed in (Golchha et al., 2019) as one of the baselines to infuse politeness in generic responses.

To demonstrate the effectiveness of each of the components in the proposed model, we experiment

with different model variants.

HT: In this model, we use the hierarchical transformers to induce politeness in responses without the RL rewards and sentiment information.

HT + RL: In this framework, we include the RL based rewards in the hierarchical model for polite response generation.

HT + RL + SE: In this framework, we provide the sentiment information to the hierarchical Transformer encoder-decoder model along with RL rewards without jointly training for both the tasks.

For sentiment classification, we train several classifiers such as CNN, LSTM, Bi-LSTM on the CYCCD dataset for predicting sentiments into 3 classes. We also employ RoBERTa (Liu et al., 2019) as one of the baselines.

	Train	*Valid*	*Test*
# Conversation	130898	19762	39665
# Utterances	168534	24724	49788

Table 2: CYCCD Dataset Statistics

4 Datasets and Experiments

In this section we explain the dataset used for experimentation and briefly provide the implementation details and evaluation metrics.

4.1 Dataset:

For our current work, we use the CYCCD dataset (Golchha et al., 2019)[1] having interactions between customers and professional customer care agents of companies on their Twitter handles. The CYCCD Twitter data was taken from the dataset made available on Kaggle by Thought vector. We use the generic and polite annotated version of the CYCCD dataset in a similar manner as (Golchha et al., 2019). As the CYCCD dataset was not annotated for sentiment, therefore we do the sentiment annotations for the dataset.

To annotate the CYCCD dataset with sentiments, we employ crowd-workers from Amazon Mechanical Turk (AMT) that labels every utterance with the provided set of sentiment labels (i.e., positive, negative, neutral). For labeling the utterances, the workers were asked to follow the instructions and guidelines provided for annotation. Some of the significant guidelines for annotation were as follows: (i). Every utterance of a given dialogue was to be marked with the provided sentiment labels; (ii) In addition, the workers were asked to provide the overall sentiment for every sentence in an utterance as well. For cases where we found different annotations in sentiment for a particular sentence, we remove them from the dataset, and we also drop the entire conversation to maintain coherence among the utterances.

A majority voting scheme was used for selecting the final sentiment for every sentence. We observe a multi-rater Kappa (McHugh, 2012) agreement ratio of approximately 75% for the sentiment, which can be considered as reliable. The final CYCCD data statistics is provided in Table 2. The sentiment distribution of the CYCCD dataset is provided in Figure 2.

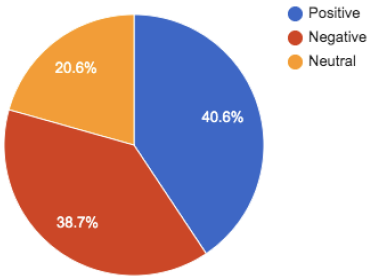

Figure 2: Sentiment distribution in the CYCCD dataset

4.2 Implementation Details:

All the implementations were done using the PyTorch[2] framework. We use the dropout (Srivastava et al., 2014) with probability 0.45. We initialize the model parameters randomly using a Gaussian distribution with the Xavier scheme (Glorot and Bengio, 2010). The hidden size for all the layers is 512. We employ AMSGrad (Reddi et al., 2019) as the optimizer for model training to mitigate the slow convergence issues.

We use uniform label smoothing with $\epsilon = 0.1$ and perform gradient clipping when the gradient norm is above 5. We use 300-dimensional word-embedding initialized with Glove (Pennington et al., 2014) embedding pre-trained on Twitter. We train with batches of size 16, and use the same size for beam search decoding. We use $\eta = 0.99$ (similar to (Paulus et al., 2017)) for the joint loss. For the reward function, the values of λ_1, λ_2 and λ_3 are 0.34, 0.33 and 0.33, respectively.

[1] https://github.com/Mauajama/Courteously-Yours

[2] https://pytorch.org/

4.3 Automatic Evaluation Metrics:

To evaluate the model at the relevance and grammatical level, we report the results using the standard metrics like Perplexity (Chen et al., 1998), Rouge-L (Lin, 2004) and BLEU-4 (Papineni et al., 2002). We also report the Politeness Accuracy as a metric to measure the degree of politeness in the responses.

We compute the politeness score using a pre-trained classifier, BERT (Devlin et al., 2018)[3] for measuring the degree of politeness in the generated responses similar to (Niu and Bansal, 2018). The classifier takes as input the generated response and generates a probability value giving us the politeness accuracy of the generated response. To evaluate the sentiment classification performance, we use the traditional metrics such as F1 score.

4.4 Manual Evaluation Metrics:

We recruit six annotators (in a similar manner as (Shang et al., 2015; Tian et al., 2019)) from a third party company, having high-level language skills. We sampled 250 responses per model for evaluation with the utterance and the conversational history provided for generation. First, we evaluate the quality of the response on two conventional criteria: *Fluency* and *Relevance*. These are rated on a five-scale, where 1, 3, 5 indicate unacceptable, moderate, and excellent performance, respectively, while 2 and 4 are used for unsure.

Secondly, we evaluate the politeness quotient of a response in terms of *Politeness Appropriateness* metric that measures whether the politeness induced in the response is in accordance with the user sentiment and the dialogue history. Here, 0 indicates irrelevant or contradictory, and 1 indicates consistent with the provided persona and dialogue context.

We compute Fleiss' kappa (Fleiss, 1971) to measure inter-rater consistency. The Fleiss' kappa for fluency and relevance are 0.53 and 0.49, indicating moderate agreement. For politeness appropriateness, we obtain 0.65 as the kappa score indicating substantial agreement.

5 Result and Analysis

In this section, we provide the experimental results for both sentiment classification and polite response generation. The proposed model performs significantly better than the other baselines for all the evaluation metrics and the improvement in each model is statistically significant compared to the other models for both the tasks[4].

The sentiment classification results are provided in Table 3. From the results, it is evident that the BERT based classifier outperforms all the other models for sentiment classification task with an improvement of more than 15 points compared to the Bi-LSTM. Hence, it can be concluded that the BERT classifier correctly identifies the sentiment of a sentence.

Model	S-F1
LSTM	71.06
CNN	69.90
Bi-LSTM	74.87
BERT (Devlin et al., 2018)	89.74
RoBERTa (Liu et al., 2019)	88.89

Table 3: Classification scores of sentiment on CYCCD data. Here, S-F1 denotes the weighted average F1 score of sentiment.

Model Description		PPL	BLEU-4	Rouge-L	PA
Existing Approaches	*Seq2Seq* (Sutskever et al., 2014)	1.112	0.145	0.278	0.38
	HRED (Serban et al., 2015)	1.085	0.198	0.308	0.45
	Polite-RL (Niu and Bansal, 2018)	1.028	0.224	0.321	0.69
	PT-TGA (Madaan et al., 2020)	1.032	0.251	0.332	0.68
	PG-RL (Golchha et al., 2019)	1.018	0.264	0.339	0.73
Proposed Approach	**HT + RL + SE (Joint Training)**	**1.004**	**0.275**	**0.352**	**0.77**
Ablation Study	*HT*	1.015	0.269	0.343	0.70
	HT + RL	1.008	0.272	0.349	0.74
	HT + RL + SE	1.006	0.273	0.350	0.75

Table 4: Automatic evaluation results. Here, PPL: Perplexity, PA: Politeness accuracy, HT: Hierarchical transformer, SE: Sentiment, RL: Reinforcement learning

In Table 4, we provide the automatic evaluation results for the proposed framework and the existing baselines. From the table, it is evident that the proposed network outperforms the *Seq2Seq* and *HRED* frameworks in terms of all the metrics. In the case of politeness, there is an enormous improvement in comparison to the Seq2Seq and HRED frameworks with more than 30% accuracy.

In comparison to the *Polite-RL* framework, our proposed network is capable of preserving the information and generating correct responses as the BLEU-4 score is significantly higher. It can be concluded that the generation of polite responses does not comprise the information present in the

[3]The classifier is trained on the Stanford Politeness Corpus (Danescu-Niculescu-Mizil et al., 2013) and achieves an accuracy of 92%.

[4]we perform statistical significance tests (Welch, 1947) and it is conducted at 5% (0.05) significance level

Dialogue Context	Sentiment Information	Generic Response	Generated Polite Examples
My order doesnot have fries in it, that sucks!	Negative	we'll follow up with the store	**HT:** We will check **HT + RL:** Please, wait we'll follow up with the store. **HT + RL + SE:** Sorry, we'll follow the order with the store **Proposed:** Sorry for the inconvenience, please wait while we follow with the store.
The new iphone has awesome display	Positive	enjoy your new iphone!	**HT:** Ohh ur new phone, great **HT + RL:** That's nice about your iphone **HT + RL + SE:** Thanks and enjoy your phone. **Proposed:** Thanks for your kind words and enjoy your new phone!

Table 5: Examples of polite responses generated by different models according to the sentiment information

Model Description		F	R	PA
Existing Approaches	*Seq2Seq* (Sutskever et al., 2014)	3.82	3.73	48%
	HRED (Serban et al., 2015)	3.86	3.78	52%
	Polite-RL (Niu and Bansal, 2018)	3.91	3.79	61%
	PT-TGA (Madaan et al., 2020)	4.03	3.85	64%
	PG-RL (Golchha et al., 2019)	4.11	4.06	67%
Proposed Approach	**HT + RL + SE (Joint Training)**	**4.23**	**4.17**	**75%**
Ablation Study	*HT*	4.09	4.03	65%
	HT + RL	4.16	4.09	71%
	HT + RL + SE	4.19	4.12	73%

Table 6: Human evaluation results. Here, PA: Politeness Appropriateness, HT: Hierarchical Transformer, SE: Sentiment, RL: Reinforcement Learning

ground-truth generic response. In the case of automatic evaluation in Table 4, our method shows a notable drop in the perplexity scores, thereby ensuring grammatically correct responses generated by the framework. By introducing sentiment information in our proposed framework, we see the growth in the performance compared to the *PG-RL* network establishing the importance of sentiment information for generating polite responses.

In Table 4, we provide the ablation study of our proposed network. From the table, it is visible that the Hierarchical Transformer network performs better than the *Seq2Seq* and *HRED* frameworks, but lacks politeness in comparison to the existing *PG-RL* baseline. This is because the model is trained without politeness and sentiment rewards.

On adding the RL training to the hierarchical network, we see that there is an improvement in terms of all the metrics. An increase in BLEU-4, Rouge-L, and politeness accuracy scores signifies that the rewards used for training the framework contribute towards better generation.

Finally, in *HT + RL + SE* adding the sentiment information of the user helps in improving the performance of the overall network. Joint training helps in simultaneously improving the performance of the end-to-end network for generating sentiment-guided polite responses.

From Table 6, it is evident that the proposed method generates grammatically correct responses as the fluency score is the highest. Similarly, the relevance score in the case of our proposed network is greater than all the existing and baseline approaches signifying that the generated responses are contextually correct according to the dialogue history.

As the primary aim of our current work is to generate politeness according to the sentiment information, the politeness appropriateness metric helps evaluate the proposed task for all the networks. The proposed framework has the highest politeness score compared to all the existing and baseline frameworks. Also, through ablation study in case of manual evaluation, it can be established that the proposed framework having joint training and RL rewards helps the overall architecture to generate polite responses according to the users' sentiments.

In Table 5, we provide a few examples and their corresponding generated responses by the different models. It exhibits that while the existing *Seq2Seq* framework generates shorter and less polite responses, the *HRED* framework can generate more complete and informative responses. The responses generated by the proposed model exhibit different variations in politeness according to the user sentiments. The variation following the different sentiments is visible in the responses, thereby achieving the desired task.

After performing a detailed quantitative and qualitative analysis of the generated responses, we came across a few mistakes committed by the baselines and the proposed frameworks.
Some of the commonly occurring errors are:
(i) Repetition: The responses generated by the baselines and the proposed framework sometimes repeat the information or generate $< unk >$ tokens. For example, in the following generated response,

there is a repetition: "*Thanks, could you dm us more info info info...*"
. ***(ii) Loss of information:*** In some cases, the models generate responses that, though having politeness factor in them, yet is incapable of providing the complete information in accordance to the ongoing conversation. For example: Gold:"*We appreciate the concern regarding the new update, how could I help*"; Predicted: "*Hello, good afternoon! How may I help?*".
(iii) Sentiment inconsistency: The generated responses having politeness in them at times do not correspond to the specified user sentiment. For example: the user sentiment is negative and the Predicted: "*Thank you ma'am, we love it...*" response is not consistent to it. This is due to the fact that the multi-task framework has wrongly predicted the sentiment of the user therefore causing the agent to respond incorrectly.

6 Conclusion

In our current work, we propose the task of transforming a generic customer care response into a polite response according to the sentiment information of the user and consistent to the context history. For the proposed task, we design a hierarchical Transformer network that captures the user and dialog context simultaneously. We create a multi-task network that identifies the sentiment information and uses the predicted sentiment information for generating the corresponding polite response. We use reinforcement learning based rewards to incorporate politeness in the customer care responses.

Experiments on the sentiment annotated CYCCD dataset proves that the proposed network not only identifies the sentiment information but also generate the contextually correct polite responses in accordance to the user sentiments. Hence, it can be concluded that the polite generation of responses is dependent on the sentiments of the user to ensure that the generation is correct and relevant.

7 Ethical Declarations

All the resources used in this paper are publicly available. The dataset used in this paper is used only for the purpose of academic research. There is nothing to disclose that warrant the ethical issues.

Acknowledgement

Authors acknowledge the partial support of the I-Hub Anubhuti-IIITD Foundation to carry out this research. Authors also acknowledge the partial support from the project "Sevak-An Intelligent Indian Language Chatbot", sponsored by Imprint, SERB.

References

Jimmy Lei Ba, Jamie Ryan Kiros, and Geoffrey E Hinton. 2016. Layer normalization. *arXiv preprint arXiv:1607.06450*.

Keith Carlson, Allen Riddell, and Daniel Rockmore. 2017. Zero-shot style transfer in text using recurrent neural networks. *arXiv preprint arXiv:1711.04731*.

Stanley Chen, Douglas H. Beeferman, and Ronald Rosenfeld. 1998. Evaluation metrics for language models.

Cristian Danescu-Niculescu-Mizil, Moritz Sudhof, Dan Jurafsky, Jure Leskovec, and Christopher Potts. 2013. A computational approach to politeness with application to social factors. *arXiv preprint arXiv:1306.6078*.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. BERT: pre-training of deep bidirectional transformers for language understanding. *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT 2019, Minneapolis, MN, USA, June 2-7, 2019, Volume 1 (Long and Short Papers)*, pages 4171–4186.

Nan Duan, Duyu Tang, Peng Chen, and Ming Zhou. 2017. Question generation for question answering. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 866–874.

William Fedus, Ian Goodfellow, and Andrew M Dai. 2018. Maskgan: Better text generation via filling in the _. *arXiv preprint arXiv:1801.07736*.

Bjarke Felbo, Alan Mislove, Anders Søgaard, Iyad Rahwan, and Sune Lehmann. 2017. Using millions of emoji occurrences to learn any-domain representations for detecting sentiment, emotion and sarcasm. In *Conference on Empirical Methods in Natural Language Processing (EMNLP)*.

Mauajama Firdaus, Asif Ekbal, and Pushpak Bhattacharyya. 2020. Incorporating politeness across languages in customer care responses: Towards building a multi-lingual empathetic dialogue agent. In *Proceedings of the 12th Language Resources and Evaluation Conference*, pages 4172–4182.

Mauajama Firdaus, Umang Jain, Asif Ekbal, and Pushpak Bhattacharyya. 2021. Seprg: Sentiment aware emotion controlled personalized response generation. In *Proceedings of the 14th International Conference on Natural Language Generation*, pages 353–363.

Mauajama Firdaus, Arunav Shandilya, Asif Ekbal, and Pushpak Bhattacharyya. 2022a. Being polite: Modeling politeness variation in a personalized dialog agent. *IEEE Transactions on Computational Social Systems*.

Mauajama Firdaus, Nidhi Thakur, and Asif Ekbal. 2022b. Sentiment guided aspect conditioned dialogue generation in a multimodal system. In *European Conference on Information Retrieval*, pages 199–214. Springer.

Joseph L Fleiss. 1971. Measuring nominal scale agreement among many raters. *Psychological bulletin*, 76(5):378.

Zhenxin Fu, Xiaoye Tan, Nanyun Peng, Dongyan Zhao, and Rui Yan. 2018. Style transfer in text: Exploration and evaluation. In *AAAI*.

Xavier Glorot and Yoshua Bengio. 2010. Understanding the difficulty of training deep feedforward neural networks. In *Proceedings of the Thirteenth International Conference on Artificial Intelligence and Statistics, AISTATS 2010, Chia Laguna Resort, Sardinia, Italy, May 13-15, 2010*, pages 249–256.

Xavier Glorot, Antoine Bordes, and Yoshua Bengio. 2011. Deep sparse rectifier neural networks. In *Proceedings of the fourteenth international conference on artificial intelligence and statistics*, pages 315–323.

Hitesh Golchha, Mauajama Firdaus, Asif Ekbal, and Pushpak Bhattacharyya. 2019. Courteously yours: inducing courteous behavior in customer care responses using reinforced pointer generator network. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 851–860.

Swati Gupta, Marilyn A Walker, and Daniela M Romano. 2007. How rude are you?: Evaluating politeness and affect in interaction. In *International Conference on Affective Computing and Intelligent Interaction*, pages 203–217. Springer.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. 2016. Deep residual learning for image recognition. In *2016 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2016, Las Vegas, NV, USA, June 27-30, 2016*, pages 770–778.

Jonathan Herzig, Guy Feigenblat, Michal Shmueli-Scheuer, David Konopnicki, Anat Rafaeli, Daniel Altman, and David Spivak. 2016. Classifying emotions in customer support dialogues in social media. In *Proceedings of the 17th Annual Meeting of the Special Interest Group on Discourse and Dialogue*, pages 64–73.

Zhiting Hu, Zichao Yang, Xiaodan Liang, Ruslan Salakhutdinov, and Eric P. Xing. 2017. Toward controlled generation of text. In *ICML*.

Jiwei Li, Will Monroe, Alan Ritter, Daniel Jurafsky, Michel Galley, and Jianfeng Gao. 2016. Deep reinforcement learning for dialogue generation. In *EMNLP*.

Jiwei Li, Will Monroe, Tianlin Shi, Alan Ritter, and Daniel Jurafsky. 2017. Adversarial learning for neural dialogue generation. In *EMNLP*.

Juncen Li, Robin Jia, He He, and Percy Liang. 2018. Delete, retrieve, generate: A simple approach to sentiment and style transfer. In *NAACL-HLT*.

Chin-Yew Lin. 2004. Rouge: a package for automatic evaluation of summaries.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Roberta: A robustly optimized BERT pretraining approach. *arXiv preprint arXiv:1907.11692*.

Aman Madaan, Amrith Setlur, Tanmay Parekh, Barnabas Poczos, Graham Neubig, Yiming Yang, Ruslan Salakhutdinov, Alan W Black, and Shrimai Prabhumoye. 2020. Politeness transfer: A tag and generate approach. *arXiv preprint arXiv:2004.14257*.

Mary L McHugh. 2012. Interrater reliability: The kappa statistic. *Biochemia medica: Biochemia medica*, 22(3):276–282.

Kshitij Mishra, Mauajama Firdaus, and Asif Ekbal. 2022a. Please be polite: Towards building a politeness adaptive dialogue system for goal-oriented conversations. *Neurocomputing*, 494:242–254.

Kshitij Mishra, Mauajama Firdaus, and Asif Ekbal. 2022b. Predicting politeness variations in goal-oriented conversations. *IEEE Transactions on Computational Social Systems*.

Tong Niu and Mohit Bansal. 2018. Polite dialogue generation without parallel data. *TACL*, 6:373–389.

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. Bleu: a method for automatic evaluation of machine translation. In *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics, July 6-12, 2002, Philadelphia, PA, USA*, pages 311–318. Association for Computational Linguistics.

Romain Paulus, Caiming Xiong, and Richard Socher. 2017. A deep reinforced model for abstractive summarization. *CoRR*, abs/1705.04304.

Yehong Peng, Yizhen Fang, Zhiwen Xie, and Guangyou Zhou. 2018. Topic-enhanced emotional conversation generation with attention mechanism. *Knowledge-Based Systems*.

Jeffrey Pennington, Richard Socher, and Christopher Manning. 2014. Glove: Global vectors for word representation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing, EMNLP 2014, October 25-29, 2014, Doha, Qatar, A meeting of SIGDAT, a Special Interest Group of the ACL*, pages 1532–1543.

Sudha Rao and Joel Tetreault. 2018. Dear sir or madam, may i introduce the gyafc dataset: Corpus, benchmarks and metrics for formality style transfer. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, volume 1, pages 129–140.

Sashank J Reddi, Satyen Kale, and Sanjiv Kumar. 2019. On the convergence of adam and beyond. *6th International Conference on Learning Representations, ICLR 2018, Vancouver, BC, Canada, April 30 - May 3, 2018, Conference Track Proceedings*.

Sathish Reddy, Dinesh Raghu, Mitesh M Khapra, and Sachindra Joshi. 2017. Generating natural language question-answer pairs from a knowledge graph using a rnn based question generation model. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 1, Long Papers*, volume 1, pages 376–385.

Steven J. Rennie, Etienne Marcheret, Youssef Mroueh, Jarret Ross, and Vaibhava Goel. 2017. Self-critical sequence training for image captioning. *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1179–1195.

Iulian V Serban, Alessandro Sordoni, Yoshua Bengio, Aaron Courville, and Joelle Pineau. 2015. Hierarchical neural network generative models for movie dialogues. *arXiv preprint arXiv:1507.04808*, 7(8):434–441.

Iulian Vlad Serban, Tim Klinger, Gerald Tesauro, Kartik Talamadupula, Bowen Zhou, Yoshua Bengio, and Aaron C Courville. 2017a. Multiresolution recurrent neural networks: An application to dialogue response generation. In *AAAI*, pages 3288–3294.

Iulian Vlad Serban, Alessandro Sordoni, Ryan Lowe, Laurent Charlin, Joelle Pineau, Aaron C Courville, and Yoshua Bengio. 2017b. A hierarchical latent variable encoder-decoder model for generating dialogues. In *AAAI*, pages 3295–3301.

Lifeng Shang, Zhengdong Lu, and Hang Li. 2015. Neural responding machine for short-text conversation. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing of the Asian Federation of Natural Language Processing, ACL 2015, July 26-31, 2015, Beijing, China, Volume 1: Long Papers*, pages 1577–1586.

Tianxiao Shen, Tao Lei, Regina Barzilay, and Tommi Jaakkola. 2017. Style transfer from non-parallel text by cross-alignment. In *Advances in Neural Information Processing Systems*, pages 6830–6841.

Xiaoyu Shen, Hui Su, Shuzi Niu, and Vera Demberg. 2018. Improving variational encoder-decoders in dialogue generation. In *AAAI*.

Weiyan Shi and Zhou Yu. 2018. Sentiment adaptive end-to-end dialog systems. *arXiv preprint arXiv:1804.10731*.

Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. 2014. Dropout: a simple way to prevent neural networks from overfitting. *The Journal of Machine Learning Research*, 15(1):1929–1958.

Sandeep Subramanian, Sai Rajeswar, Francis Dutil, Chris Pal, and Aaron Courville. 2017. Adversarial generation of natural language. In *Proceedings of the 2nd Workshop on Representation Learning for NLP*, pages 241–251.

Ilya Sutskever, Oriol Vinyals, and Quoc V Le. 2014. Sequence to sequence learning with neural networks. In *Advances in neural information processing systems*, pages 3104–3112.

Zhiliang Tian, Wei Bi, Xiaopeng Li, and Nevin L Zhang. 2019. Learning to abstract for memory-augmented conversational response generation. In *Proceedings of the 57th Conference of the Association for Computational Linguistics, ACL 2019, Florence, Italy, July 28- August 2, 2019, Volume 1: Long Papers*, pages 3816–3825.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in neural information processing systems*, pages 5998–6008.

Oriol Vinyals and Quoc Le. 2015. A neural conversational model. *arXiv preprint arXiv:1506.05869*.

Bernard L Welch. 1947. The generalization ofstudent's' problem when several different population variances are involved. *Biometrika*, 34(1/2):28–35.

Ronald J Williams. 1992. Simple statistical gradient-following algorithms for connectionist reinforcement learning. *Machine learning*, 8(3-4):229–256.

Xianchao Wu, Ander Martinez, and Momo Klyen. 2018. Dialog generation using multi-turn reasoning neural networks. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, volume 1, pages 2049–2059.

Hainan Zhang, Yanyan Lan, Jiafeng Guo, Jun Xu, and Xueqi Cheng. 2018. Reinforcing coherence for sequence to sequence model in dialogue generation. In *IJCAI*, pages 4567–4573.

Focus-Driven Contrastive Learning for Medical Question Summarization

Ming Zhang[1,2] **Shuai Dou**[3] **Ziyang Wang**[1,2] **Yunfang Wu**[1,3*]
[1]MOE, Key Laboratory of Computational Linguistics, Peking University
[2]School of Software and Microelectronics, Peking University
[3]School of Computer Science, Peking University
[1]{zhangming,wzy232303}@stu.pku.edu.cn
[2]{monkdou,wuyf}@pku.edu.cn

Abstract

Automatic medical question summarization can significantly help the system to understand consumer health questions and retrieve correct answers. The Seq2Seq model based on maximum likelihood estimation (MLE) has been applied in this task, which faces two general problems: the model can not capture well question focus and and the traditional MLE strategy lacks the ability to understand sentence-level semantics. To alleviate these problems, we propose a novel question focus-driven contrastive learning framework (QFCL). Specially, we propose an easy and effective approach to generate hard negative samples based on the question focus, and exploit contrastive learning at both encoder and decoder to obtain better sentence-level representations. On three medical benchmark datasets, our proposed model achieves new state-of-the-art results, and obtains a performance gain of 5.33, 12.85 and 3.81 points over the baseline BART model on three datasets respectively. Further human judgement and detailed analysis prove that our QFCL model learns better sentence representations with the ability to distinguish different sentence meanings, and generates high-quality summaries by capturing question focus.

1 Introduction

A growing number of health questions are raised by consumers on websites nowadays, which are usually written in natural language and including detailed and peripheral information not related to the answers. Summaries of such questions can greatly improve the performance in retrieving relevant answers (Ben Abacha and Demner-Fushman, 2019). Accordingly, the medical question summarization task is defined as summarizing the consumer health questions (CHQ) into frequently asked questions (FAQ), which are shorter but remain essential information of the original question to get correct answers. An example of medical question summarization is shown in Table 1.

Input question: consumer health question (CHQ) subject: gender dysphoria message: no health care on my son suffering from gender dysphoria what can we do to help him he worked out of high school no problems now not working and about shutting himself in his room 24/7 theres nothing this condition in our area we live in [location].no help in area what can we do he has had bad thoughts already please help us with some sort of info thank yuo [name] [location]
Golden summary: frequently asked question (FAQ): Where can I find information on treatment and resources for gender dysphoria?
Summary by BART (baseline): What are the treatments for weight loss?
Summary by our model: What are the treatments for gender dysphoria?

Table 1: An example of medical question summarization in MeqSum dataset, where the question focus is highlighted in green. Summaries generated by BART and our model are also listed.

The Seq2Seq neural models have been widely used in abstractive summarization (Nallapati et al., 2016; Lewis et al., 2020; Zhang et al., 2020) and show promising potentials, and they have also been applied in medical question summarization and achieve current state-of-the-art results. Ben Abacha and Demner-Fushman (2019) apply the pointer-generator model for this task. Yadav et al. (2021a) present a reinforcement learning framework with question-type identification reward and question-focus recognition reward. Mrini et al. (2021b) propose a multitask learning method by treating recognizing question entailment as an auxiliary task.

In the medical question summarization task, the input question CHQ is always lengthy and contains redundant information, where some salient medical entities and the semantic focus of question are vital to understand users' intention. But it still remains a challenging task for the existing methods to capture the question focus. As described in the example

*Corresponding author.

Proceedings of the 29th International Conference on Computational Linguistics, pages 6176–6186
October 12–17, 2022.

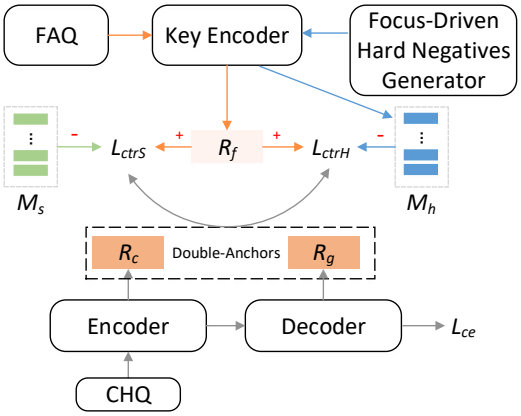

Figure 1: Sketch of our proposed contrastive learning framework. M_s,M_h represents the memory bank that contains simple negative samples and hard negative samples respectively. $\mathcal{R}_f$, $\mathcal{R}_c$, $\mathcal{R}_g$ denotes the sentence representation of FAQ, CHQ and generated summary. $\mathcal{L}_{ctrS}$ and $\mathcal{L}_{ctrH}$ are contrastive learning loss on simple negative samples and hard negative samples respectively. $+$ indicates the positive sample, and $-$ indicates the negative sample.

1, the focus "*gender dysphoria*" is mis-replaced by "*weight loss*" in the summary generated by the fine-tuned BART, resulting in a completely different meaning from the original sentence.

For the medical question summarization task, the generated question summary is required to semantically close to the reference question. However, in most of current pre-trained models such as BART (Lewis et al., 2020), the model adopts maximum likelihood estimation (MLE) and mainly focuses on the accuracy of the prediction of masked tokens, but does not guarantee to the semantic similarity or dissimilarity of the whole sentences. To address this issue, some previous works adopt reinforcement learning (RL) in text summarization task (Li et al., 2019; Paulus et al., 2018), but RL suffers from the noise gradient estimation problem (Greensmith et al., 2004), which makes the training process unstable and sensitive to hyper-parameters.

To alleviate these problems, we propose a novel question focus-driven contrastive learning (QFCL) framework for medical question summarization, as illustrated in Figure 1. In our model, we introduce a "*double anchors*" strategy for contrastive learning, by utilizing the sentence representation of CHQ as an anchor and the generated summary as another anchor, and regarding the golden reference FAQ as the positive sample. In addition, we present a "*focus-driven hard negatives generator*" to construct hard negative samples, by replacing the focus phrases with other phrases sharing the same attribute.

Through contrastive learning, we minimize the distance between CHQ/generated summary and golden reference, and maximize the distance between CHQ/generated summary and other negative samples. By using the *double anchors*, our model is able to extract sentence-level semantic features to alleviate the problem of MLE. With the help of *hard negatives generator*, the model learns to pay more attention to question focus and thus produces high quality summary.

We conduct extensive experiments on three medical question summarization datasets: Meqsum (Ben Abacha and Demner-Fushman, 2019), HealthCareMagic and iCliniq (Zeng et al., 2020). Our proposed model outperforms previous best results by a wide margin, achieving new state-of-the-art results on all three datasets. Compared with the baseline BART, our model brings a relative performance gain of 12.2%, 28.7% and 9.6% on Meqsum, Cliniq and HealthcareMagic respectively. Through analysis, we prove that our model significantly gains the power of distinguishing the semantics between generated summaries and negative samples, and our model generates high-quality summaries capturing more question focuses.

2 Ralated Work

2.1 Medical Question Summarization

The medical question summarization task is defined by Ben Abacha and Demner-Fushman (2019). They construct a benchmark dataset Meqsum, and apply a pointer-generator model to generate question summary. At the question summarization campaign of MEDIQA-21 organized by Ben Abacha et al. (2021), almost all approaches rely on the fine-tuning of pre-trained transformer models. Transfer learning, knowledge-base, and ensemble methods are widely utilized by participanting teams to achieve better performance (He et al., 2021; Yadav et al., 2021b; Mrini et al., 2021c; Sänger et al., 2021). In this paper, we also base our method on the strong pre-trained BART model.

Recently, Yadav et al. (2021a) propose a RL framework with two question-aware semantic rewards: question-type identification reward (QTR) and question-focus recognition reward (QFR).

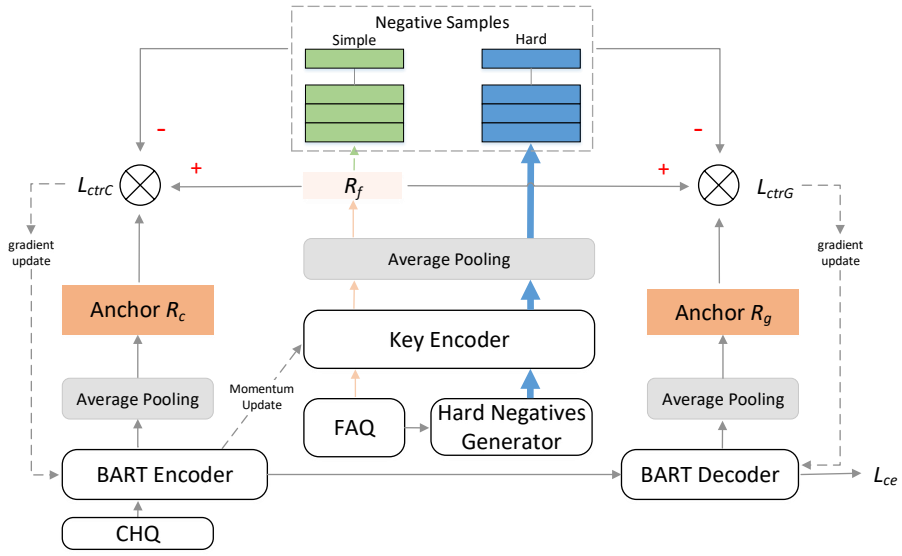

Figure 2: The overall framework of QFCL. $\mathcal{L}_{ctrC}$ and $\mathcal{L}_{ctrG}$ are contrastive learning loss on the two anchors respectively.

QTR is to identify whether the question types are consistent with the gold question, and QFR is designed to capture question focus. But in their work, the question types and question focuses in the dataset should be manually labeled, which is both time-consuming and labor-intensive for large-scale datasets such as HealthcareMagic and iCliniq. Moreover, the RL training process is unstable. Mrini et al. (2021b) claim an equivalence between medical question summary and recognizing question entailment(RQE), and employ multi-task learning to train the model to not only perform next-word-prediction but also carry question entailment recognition. These two studies demonstrate that the pre-trained models achieve better performance after capturing the underlying sentence semantics of generated questions. Different from these works, we exploit contrastive learning to obtain focus-aware question representations.

2.2 Contrastive Learning

Different from the traditional methods which learn representations in pixel-level for computer vision tasks, contrastive learning encodes high-level features to distinguish different objects and has achieved great success (Henaff, 2020; Chen et al., 2020; Misra and van der Maaten, 2020; He et al., 2020), and it has also been applied in several NLP tasks such as machine translation (Pan et al., 2021), pre-training (Chi et al., 2021) and question answering (Yang et al., 2021). In the field of summarization, Liu and Liu (2021) present a contrastive framework to bridge the gap between the learning objective and evaluation metrics, Cao and Wang (2021) design several negative sample construction strategies to solve the factual inconsistency problem. In contrast, we use the MoCo structure to handle with the large volume of negative samples, and propose a new negative sample construction method.

Chen et al. (2020) prove that large size of negative samples can improve the performance of contrastive learning, but it also brings heavy burden on computation cost. To address this issue, He et al. (2020) propose MoCo, which maintains a queue as the memory bank to store negative samples. MoCo adopts two encoders with the same structure: key encoder and query encoder, where the key encoder is momentum updated from the query encoder.

3 Model

Given an input question CHQ, which is written by consumers and contains lengthy and complex information, the medical question summarization task aims to automatically generate a question summary that is a frequently asked question (FAQ), capturing the essential information to help efficiently retrieve correct answers. A more detailed structure of our proposed QFCL model is presented in Figure 2.

3.1 Contrastive Learning Architecture

We employ the pre-trained BART (Lewis et al., 2020) as our basic model to generate question summaries. For contrastive learning, we adopt the MoCo architecure (He et al., 2020), which contains a key encoder E_k with the same structure as the BART encoder E_q, and a queue to store simple negative samples with large volume. The simple negative samples in the queue are progressively replaced by current mini-batch of representations extracted from the key encoder. All samples in the queue will be used as negative samples in the next batch. In addition, QFCL employs a *hard negatives generator* to generate hard negative samples.

In our model, the BART encoder E_q and the decoder are updated via back propagation by combining three types of loss functions, as described in the subsequent sections. The parameters of E_k are frozen and updated slowly towards that of E_q:

$$\theta_k \leftarrow m\theta_k + (1-m)\theta_q \quad (1)$$

where m is a momentum coefficient.

At the inference, only the BART encoder and decoder are retained, other parts such as the key encoder, the queue, and the *hard negatives generator* are all discarded.

3.2 Simple Negative Samples

In the medical question summarization task, the input question CHQ should be semantically close to its reference summary FAQ but different from other question summaries. Therefore, we regard the CHQ c_i in the i-th pair as the anchor, FAQ f_i in the same pair as the positive sample and randomly select f_j from other different pairs to serve as simple negative samples.

Let $\mathcal{R}_s$ denote the average decoded output of an arbitrary sentence s, the objective function of the simple contrastive learning is defined as:

$$\mathcal{L}_{ctrCS} = -log\frac{e^{sim(\mathcal{R}_{ci},\mathcal{R}_{fi})/\tau}}{\sum_{\mathcal{R}_{fj}\in M_s} e^{sim(\mathcal{R}_{ci},\mathcal{R}_{fj})/\tau}} \quad (2)$$

where $\mathcal{R}_{ci}$ indicates the sentence representation of the i-th CHQ extracted from E_q, and $\mathcal{R}_{fi}$ and $\mathcal{R}_{fj}$ are extracted from the key encoder E_k for the i-th and j-th FAQ respectively. The operation sim is to calculate the cosine similarity, τ is a temperature hyper-parameter. M_s is the memory bank which contains one positive sample and K simple negative samples in the queue with respect to an anchor.

3.3 Focus-Driven Hard Negative Samples

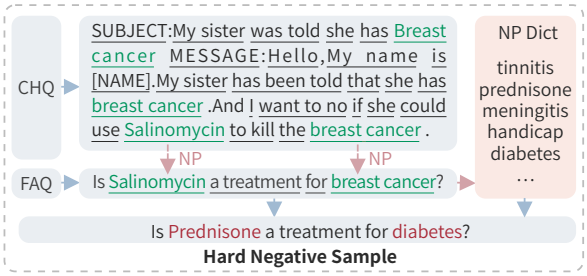

Figure 3: The method of hard negative samples generation.

The above simple negative samples are randomly selected. As claimed by (Kalantidis et al., 2020), hard negative samples that are more similar to positive samples can facilitate the model to get better performance. Inspired by this, we build a bridge between hard sample generation and question focus prediction.

3.3.1 Question Focus Identification

As mentioned before, the question focus is essential to understand a consumer health question. If some focus phrases are missing in the generated summary, the semantic will drift far away from the original user's intention. So we construct difficult negative samples based on the question focus to enhance contrastive learning. Specially, we replace the focus phrases with some other phrases of the same attribution, and keep other words of the sentence unchanged. An example of hard negative sample generation is shown in Figure 3.

One issue for our method is how to automatically annotate question focus. Yadav et al. (2021a) manually labeled the question focus in MeqSum dataset. However, this is quite time-consuming and labor-intensive, driving us to find a method which can automatically mark the question focus in larger datasets, such as HealthcareMagic and iCliniq. We analyzed the manually labeled MeqSum dataset, and found that in 340 of the total 500 records (up to 68%), the question focuses are the overlap phrases between CHQ and FAQ. Accordingly, we hypothesize that the same phrases appearing both in the source question and the golden summary have a high probability to be key-phrases. This idea is also proved to be effective in (Li et al., 2020).

Since the question focus is usually a phrase

rather than a single word, we need to split one sentence into phrases. We apply the chunker (Akbik et al., 2018) to the CHQ and FAQ text, and record the chunk label of each phrase. Then the consistent phrases appearing both in CHQ and FAQ are labeled as the question focuses.

3.3.2 Hard Negative Sample Generation

We constructed a dictionary by concatenating all phrases of the FAQ sentences in the train set. To generate hard negative samples, the question focuses are randomly replaced by other phrases of the same chunk label from the dictionary. As shown in Figure 2, "*breast cancer*" is replaced by "*diabetes*" since they share the same label "*NP*". We repeat this process N_h times to construct N_h different hard negative samples for each CHQ-FAQ pair.

3.3.3 Contrastive Learning on Hard Negative Samples

The sentence representation of hard sample $\mathcal{R}_h$ is extracted from the key encoder E_k. We define the hard loss function of contrastive learning as:

$$\mathcal{L}_{ctrCH} = -log\frac{e^{sim(\mathcal{R}_{ci},\mathcal{R}_{fi})/\tau}}{\sum_{\mathcal{R}_h \in M_h} e^{sim(\mathcal{R}_{ci},\mathcal{R}_h)/\tau}} \quad (3)$$

where M_h denotes the memory bank containing one positive sample and N_h hard negative samples.

This loss function forces the model to not only shorten the distance between CHQ and FAQ, but also expand the gap between the CHQ and hard negative samples. In this way, we achieve the goal of making the model pay more attention to the question focus, and obtain a focus-aware representation.

3.4 Contrastive Learning at Decoder

An imbalance existing in the above method is that contrastive learning is only utilized at the encoder. We fine-tuned BART on iCliniq dataset, and found that the decoder lacks the ability to distinguish the representations between the generated summary and the positive samples/unrelated negative samples, as $s^+_{g\_faq}$, $s^-_{g\_sim}$, $s^-_{g\_hard}$ shown in Figure 4. Therefore, we try to improve the similarity between the generated summary and its reference FAQ, and at the same time enlarge the dis-similarity between the generated summary and other unrelated questions.

Specially, we regard the generated summary as an extra anchor, and denote the representation of the generated summary as g_i. Since the output summary should be semantically consistent with the corresponding FAQ, we consider the representation of the FAQ f_i in the same pair as the positive sample, and select the simple negative samples randomly from the queue and generate hard negative samples using the *hard negatives generator*. The object functions of contrast loss $\mathcal{L}_{ctrGS}$ and $\mathcal{L}_{ctrGH}$ at the decoder end are defined in a similar style as Equation 2 and 3, except that the anchor c_i is replaced by another anchor g_i.

3.5 Overall Objective Function

For predicting next tokens in the generated summary, we use the cross entropy loss $\mathcal{L}_{ce}$:

$$\mathcal{L}_{ce} = -\frac{1}{|T|}\sum_{t \in T} log(p(y_t|x, y_{1:t-1}, \theta)) \quad (4)$$

In our model, the overall loss function consists of five parts: the cross entropy loss $\mathcal{L}_{ce}$ and four different loss functions of contrastive learning: $\mathcal{L}_{ctrCS}$, $\mathcal{L}_{ctrCH}$ for the anchor at the encoder end, $\mathcal{L}_{ctrGS}$, $\mathcal{L}_{ctrGH}$ for the anchor at the decoder end. We define the contrastive learning loss with respect to these two anchors as:

$$\begin{aligned} \mathcal{L}_{ctrC} &= \alpha\mathcal{L}_{ctrCS} + \beta\mathcal{L}_{ctrCH} \\ \mathcal{L}_{ctrG} &= \alpha\mathcal{L}_{ctrGS} + \beta\mathcal{L}_{ctrGH} \end{aligned} \quad (5)$$

where α, β are hyper-parameters to control the balance between simple negatives and hard ones. The weights of contrastive learning loss at the encoder and decoder are considered as equal, and the overall loss is defined as:

$$\mathcal{L} = \mathcal{L}_{ce} + \frac{1}{2}\mathcal{L}_{ctrC} + \frac{1}{2}\mathcal{L}_{ctrG} \quad (6)$$

4 Experiments

4.1 Datasets

We conduct experiments on three English benchmark medical question summarization datasets, including Meqsum, HealthcareMagic and iCliniq. Meqsum is a high-quality dataset from NIH [1], constructed by Ben Abacha and Demner-Fushman (2019). Mrini et al. (2021a) extracted HealthCareMagic and iCliniq datasets from MedDialog (Zeng et al., 2020) , which are collected automatically from the online healthcare service platforms

[1] www.nlm.nih.gov/medlineplus

[2] [3]. MeqSum's and HealthcareMagic's summaries are written by medical experts in formal style, while iCliniq's are patient-written. We list some statistics of these datasets in table 2. Following previous works, we adopt ROUGE (Lin, 2004)[4] as the evaluation metric.

Dataset	Train	Dev	Test	Length
MeqSum	400	100	500	60.8/10.1
HealthCareMagic	181,122	22,641	22,642	82.8/9.7
iCliniq	24,851	3,105	3,106	89.7/12.3

Table 2: Statistics of three medical question summarization datasets. Length indicates the average length of CHQ/FAQ.

4.2 Training Details

We utilize BART-large (Lewis et al., 2020) in huggingface[5] as our pre-trained model. The learning rate of BART baseline is set to 3e-5 as the same with Mrini et al. (2021b). For contrastive learning in QFCL, the learning rate is optimized to 1e-5. Betas of Adam optimizer is set to 0.9 and 0.999. Batch size is set to 16. The number of hard negative samples n_h is set to 64. For Moco, the queue size K is set to 4096, temperature τ is 0.07, and the momentum coefficient m is 0.999. In Equation 5, α and β are set to 1 and 0.5 respectively through grid search on MeqSum development set. Experiments were all performed on a single NVIDIA RTX 3090 GPU. The average runtimes of each epoch for MeqSum, iCliniq and HealthcareMagic are 4.2h, 0.6h and 0.1h respectively.

4.3 Overall Performance

We report our experimental results in Table 3. Our model achieves new state-of-the-art results on all three datasets. Compared with the previous best results, we obtain an improvement of 0.99 ROUGE-L score on MeqSum, 8.44 on iCliniq, and 0.51 on HealthcareMagic, respectively.

MTL+Data augmentation (Mrini et al., 2021b) obtains the previous state-of-the-art results on iCliniq and HealthcareMagic, which utilizes the question entailment data to augment summarization data. In contrast, our method doesn't need other classification models or external data. The work of ProphetNet+QTR+QFR (Yadav et al., 2021a) gets the previous best result on MeqSum, which presents a reinforcement learning-based framework with question-aware rewards. Comparing with this competitive model, our method obtains consistent better performance on all metrics, with 2.28 improvement on R1, 4.66 improvement on R2 and 0.89 improvement on RL. We did not compare the results of (Yadav et al., 2021a) on the other two datasets, since their method requires manually labeled question focuses and question types.

4.4 Ablation Study

We perform ablation study to evaluate the impacts of different components employed in QFCL, and report the results in Table 3. In particular, for Meqsum dataset, due to the small size which may cause the training unstable, we conducted five separate experiments and computed the average ROUGE score of these five checkpoints as the final result. Compared with the base BART model, we obtain an absolute improvement of 5.33 points on average. T-test is implemented on such five ROUGE scores and the p-value is less than 1e-2, validating that this improvement is significant. On Cliniq the absolute improvement is 12.85 points and on HealthcareMagic 3.81 points. In comparison to BART, the relative improvements of our model are 12.2%, 28.7% and 9.6% on Meqsum, Cliniq and HealthcareMagic respectively.

The results demonstrate that each component of our model is helpful. On MeqSum, there is an increase of 3.15 points for BART+S compared to the baseline, indicating that the contrastive learning on simple negative samples largely improves model performance. It shows an continuous increase of 0.77 points for BART+S+H, and the highest ROUGE-L score is obtained when three parts are all implemented in our model. It suggests that each component in QFCL contributes positively, and metrics like ROUGE evaluating the similarity between whole sentences benefit from our contrastive learning strategy.

4.5 Human Evaluation

To quantitatively assess the results, we compare our method with the baseline BART through human judgement. We randomly selected 50 samples from each of three datasets, and hired 3 graduate students to categorize each generated summary into one of the following categories: 'Incorrect', 'Acceptable', and 'Perfect'. We compute the average number of each category, and report the result in Table 4. The average Spearman correlation co-

[2]www.healthcaremagic.com
[3]www.icliniq.com
[4]https://pypi.org/project/py-rouge
[5]huggingface.co/facebook/bart-large

Model	MeqSum			iCliniq			HealthCareMagic		
	R1	R2	RL	R1	R2	RL	R1	R2	RL
ProphetNet + QTR + QFR(Yadav et al., 2021a)	45.52	27.54	48.19	-	-	-	-	-	-
MTL+Data augmentation(Mrini et al., 2021b)	49.20	29.50	44.80	54.20	36.90	49.10	45.90	24.30	42.90
BART (Lewis et al., 2020)	46.17	28.05	43.75	48.79	25.47	44.69	42.33	23.07	39.60
BART + S	49.30	31.78	46.89	56.58	36.43	52.06	44.35	24.73	41.46
BART + S + H	49.96	32.72	47.66	58.26	40.08	55.34	45.52	25.71	42.51
BART + S + H + D (QFCL)	**51.48**	**34.16**	**49.08**	**60.09**	**43.22**	**57.54**	**46.42**	**26.47**	**43.41**

Table 3: Experimental results on three medical question summarization datasets. S denotes the contrastive learning on simple negative samples at the encoder end; H denotes the contrastive learning on hard negative samples at the encoder end; D denotes the decoder end's contrastive learning. The top group lists the existing state-of-the-art results on three datasets, and the bottom group shows our ablation study on different components.

efficient between three annotators is 0.68, which guarantees a high quality of our annotation data. The evaluation results show that our model generates a higher proportion of perfect samples and a lower proportion of incorrect ones, by enhancing the model's ability of capturing sentence semantics and question focuses.

Model	MeqSum			iCliniq			HealthCareMagic		
	I	A	P	I	A	P	I	A	P
BART	28.7	17.3	4.0	12.3	17.0	20.7	20.7	20.3	9.0
QFCL	12.0	18.0	20.0	6.3	17.7	26.0	5.7	16.3	28.0

Table 4: Human evaluation of the summaries generated by BART and QFCL respectively. The metric **I** means the number of incorrect samples, **A** means acceptable, **P** means perfect.

4.6 Case Study

To clearly show the output question summary, we list two samples to compare our model with BART in Table 5. In Case 1, BART captures the question focus "*Ampicillin*" but misses "*drink alcohol*", and in Case 2 it misses the question focus "*breast milk*". In contrast, our model successfully extract multiple question focuses from the lengthy CHQ, and generate summaries which more conform to the meaning of original questions.

Case1	
CHQ	MESSAGE: Is it okay to drink **alcohol** in moderation when taking **Ampicillin**. I was told it negates any medical effect of the drug
FAQ	Can I drink **alcohol** while taking **Amoxicillin**?
BART	What are the side effects of **Ampicillin**?
QFCL	Is it okay to drink **alcohol** with **Ampicillin**?
Case2	
CHQ	Hi..... I have 3 month old baby girl..... I don t have **breast milk** from the beginning due to some reason. I can not give formula milk to baby..... So right now i m giving buffelo milk ..... What else i should give her for better nourishment????? ..... She has constipation problem may be due to milk but i cant give her breastmilk or formula ..... How to overcome it?????..... Please help me
FAQ	Suggest ways to feed newborn other than **breast milk**
BART	Suggest treatment for constipation in a child
QFCL	Suggest better nourishment for baby other than **breast milk**

Table 5: Examples of generated question summaries by BART and our QFCL model. The question focuses are highlighted.

5 Model Analysis

5.1 Correlation of Sentence Representations

Since the auxiliary structures are discarded at the inference stage, we make further analysis to check that whether the retained model has the ability to distinguish different sentence-level semantics when facing unknown data. We train QFCL and BART on the training set for 20 epochs and save each checkpoint, and evaluate these checkpoints on the development set.

Four types of sentence representations are extracted from these checkpoints: CHQ's representation $\mathcal{R}_c$, FAQ's representation $\mathcal{R}_f$, hard negatives' representation $\mathcal{R}_h$, and the generated summary's representation at decoder end $\mathcal{R}_g$. Then we calculate the cosine similarity between them, and draw the relationship between these similarity scores and the epoch numbers, as shown in Figure 4.

Regarding the anchor CHQ in the curve of iCliniq, $s^+_{c\_faq}$, $s^-_{c\_sim}$ and $s^-_{c\_hard}$ are very close to each other at epoch 0, suggesting that the initial encoder lacks the ability to capture different semantics. With the increase of training steps, $s^+_{c\_faq}$ changes smoothly, while $s^-_{c\_sim}$ decreases sharply to near zero and $s^-_{c\_hard}$ decreases gradually and converges at a middle level between $s^+_{c\_faq}$ and $s^-_{c\_sim}$.

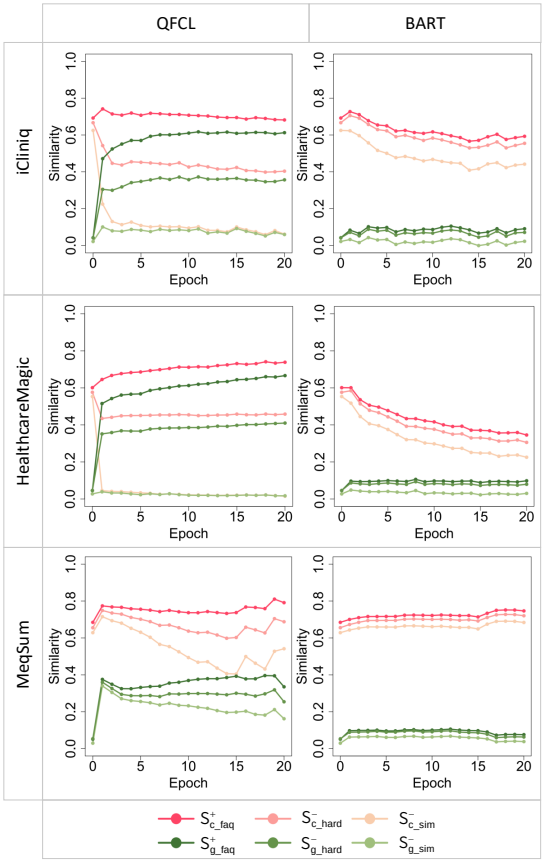

Figure 4: Correlation between sentence representation similarities and epoch numbers on dev set. The red lines are about the anchor CHQ. $s^+_{c\_faq}$ is the average cosine similarity between CHQ and related FAQ, $s^-_{c\_sim}$ is between CHQ and simple negative samples (other FAQs), $s^-_{c\_hard}$ is between CHQ and hard negative samples. The green lines are about the anchor of generated summary. $s^+_{g\_faq}$ is the average cosine similarity between the generated summary and FAQ, $s^-_{g\_sim}$ is between generated summary and simple negatives, $s^-_{g\_hard}$ is between generated summary and hard negatives. The epoch number equaling 0 denotes the initial pre-trained model.

This suggests that, powered by contrastive learning, our model has learned to distinguish sentences of different meanings at the encoder end.

With the generated summary as another anchor, we find out that $s^+_{g\_faq}$, $s^-_{g\_sim}$, $s^-_{g\_hard}$ are all near to 0 initially, which depict that the decoder is also weak in representing sentence-level semantics. After training, $s^+_{g\_faq}$ increases significantly, $s^-_{g\_hard}$ converges between $s^+_{g\_faq}$ and $s^-_{g\_sim}$, and $s^-_{g\_sim}$ keeps very low all the time. It suggests that the decoder has strengthened its power to distinguish different semantics as the same to the encoder end.

Another chart is drawn to show this relationship for BART baseline in Figure 4. The similarities between the anchor and the positive samples, negative samples are very close, and never improve significantly with the progress of training. This situation suggesting that the BART baseline has a relatively weaker performance to distinguish the sentences of different meanings at both encoder and decode, since it only focuses on the prediction of next tokens.

We also draw this correlation curve on MeqSum and HealthcareMagic. The curve of HealthcareMagic is similar to iCliniq. On MeqSum, our model can still distinguish sentences with different semantics better than the baseline, but the signal is not as significant as iCliniq or HealthcareMagic due to the limited size of training set.

Model	C1	C2	C3	C4	C5	Mean
BART	33.37	40.76	39.78	35.34	36.21	37.09
QFCL	47.41	42.24	45.20	45.20	47.17	**45.44**

Table 6: Accuracy of question focuses in generated summaries. C1-C5 means 5 different checkpoints trained by each model.

5.2 Capturing Question Focus

To study whether our model pays more attention to the question focus, we evaluate the accuracy of question focuses in generated summaries. We use the sequence labeling model trained by Yadav et al. (2021a) to predict question focuses on the MeqSum dataset, and regard the 812 predicted question focuses in test set as the gold-standard. For QFCL and BART, we train five checkpoints and generate summaries on these checkpoints, and compute the accuracy of question focuses on test set. As shown in Table 6, the average accuracy is 37.09% for BART and 45.44% for QFCL. Our model exceeds the baseline by 8.35 points for question focus generation. P-value of t-test on these two sets of results is 1.04e-3, indicating that this improvement is statistically significant.

6 Conclusion

In this paper, we introduce a novel question focus-based contrastive learning framework QFCL for medical question summarization. In the proposed model, we adopt a "double anchor" strategy, by considering both the input question CHQ and the generated summary as comparing anchors. And we exploit a "hard negatives generator" to generate hard negative samples based on the question focus.

Our model significantly improves the performance on three medical question summarization datasets, and achieves new state-of-the-art results. In the future, we would like to find a more effective way to do question focus recognition.

Acknowledgement

This work is supported by the National Hi-Tech RD Program of China (No.2020AAA0106600), the National Natural Science Foundation of China (62076008) and the Key Project of Natural Science Foundation of China (61936012).

A Ethical Consideration

The datasets used in our work are all publicly available. We used BART as our basic model which follows the apache-2.0 license. The datasets and the method should only be used for research purposes, not in the commercial field.

The personal information in the datasets has been hidden through preprocessing. For example, the name of the consumer was converted to an placeholder [name] and the address was converted to [location], as shown in Table 1.

As the current models could not guarantee to generate summaries fully conforms to the intention of the consumers, the method in our paper can only be used as an auxiliary tool to avoid further misleading suggestions.

References

Alan Akbik, Duncan Blythe, and Roland Vollgraf. 2018. Contextual string embeddings for sequence labeling. In *Proceedings of the 27th International Conference on Computational Linguistics*, pages 1638–1649, Santa Fe, New Mexico, USA. Association for Computational Linguistics.

Asma Ben Abacha and Dina Demner-Fushman. 2019. On the summarization of consumer health questions. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 2228–2234, Florence, Italy. Association for Computational Linguistics.

Asma Ben Abacha, Yassine Mrabet, Yuhao Zhang, Chaitanya Shivade, Curtis Langlotz, and Dina Demner-Fushman. 2021. Overview of the MEDIQA 2021 shared task on summarization in the medical domain. In *Proceedings of the 20th Workshop on Biomedical Language Processing*, pages 74–85, Online. Association for Computational Linguistics.

Shuyang Cao and Lu Wang. 2021. CLIFF: Contrastive learning for improving faithfulness and factuality in abstractive summarization. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 6633–6649, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Ting Chen, Simon Kornblith, Mohammad Norouzi, and Geoffrey Hinton. 2020. A simple framework for contrastive learning of visual representations. In *International conference on machine learning*, pages 1597–1607. PMLR.

Zewen Chi, Li Dong, Furu Wei, Nan Yang, Saksham Singhal, Wenhui Wang, Xia Song, Xian-Ling Mao, Heyan Huang, and Ming Zhou. 2021. InfoXLM: An information-theoretic framework for cross-lingual language model pre-training. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 3576–3588, Online. Association for Computational Linguistics.

Evan Greensmith, Peter L Bartlett, and Jonathan Baxter. 2004. Variance reduction techniques for gradient estimates in reinforcement learning. *Journal of Machine Learning Research*, 5(9).

Kaiming He, Haoqi Fan, Yuxin Wu, Saining Xie, and Ross Girshick. 2020. Momentum contrast for unsupervised visual representation learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 9729–9738.

Yifan He, Mosha Chen, and Songfang Huang. 2021. damo_nlp at MEDIQA 2021: Knowledge-based preprocessing and coverage-oriented reranking for medical question summarization. In *Proceedings of the 20th Workshop on Biomedical Language Processing*, pages 112–118, Online. Association for Computational Linguistics.

Olivier Henaff. 2020. Data-efficient image recognition with contrastive predictive coding. In *International Conference on Machine Learning*, pages 4182–4192. PMLR.

Yannis Kalantidis, Mert Bulent Sariyildiz, Noe Pion, Philippe Weinzaepfel, and Diane Larlus. 2020. Hard negative mixing for contrastive learning. In *Advances in Neural Information Processing Systems*, volume 33, pages 21798–21809. Curran Associates, Inc.

Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Veselin Stoyanov, and Luke Zettlemoyer. 2020. BART: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7871–7880, Online. Association for Computational Linguistics.

Haoran Li, Junnan Zhu, Jiajun Zhang, Chengqing Zong, and Xiaodong He. 2020. Keywords-guided abstractive sentence summarization. In *Proceedings of*

the AAAI Conference on Artificial Intelligence, volume 34, pages 8196–8203.

Siyao Li, Deren Lei, Pengda Qin, and William Yang Wang. 2019. Deep reinforcement learning with distributional semantic rewards for abstractive summarization. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 6038–6044, Hong Kong, China. Association for Computational Linguistics.

Chin-Yew Lin. 2004. ROUGE: A package for automatic evaluation of summaries. In *Text Summarization Branches Out*, pages 74–81, Barcelona, Spain. Association for Computational Linguistics.

Yixin Liu and Pengfei Liu. 2021. SimCLS: A simple framework for contrastive learning of abstractive summarization. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 2: Short Papers)*, pages 1065–1072, Online. Association for Computational Linguistics.

Ishan Misra and Laurens van der Maaten. 2020. Self-supervised learning of pretext-invariant representations. In *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 6706–6716.

Khalil Mrini, Franck Dernoncourt, Walter Chang, Emilia Farcas, and Ndapa Nakashole. 2021a. Joint summarization-entailment optimization for consumer health question understanding. In *Proceedings of the Second Workshop on Natural Language Processing for Medical Conversations*, pages 58–65, Online. Association for Computational Linguistics.

Khalil Mrini, Franck Dernoncourt, Seunghyun Yoon, Trung Bui, Walter Chang, Emilia Farcas, and Ndapa Nakashole. 2021b. A gradually soft multi-task and data-augmented approach to medical question understanding. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 1505–1515, Online. Association for Computational Linguistics.

Khalil Mrini, Franck Dernoncourt, Seunghyun Yoon, Trung Bui, Walter Chang, Emilia Farcas, and Ndapa Nakashole. 2021c. UCSD-adobe at MEDIQA 2021: Transfer learning and answer sentence selection for medical summarization. In *Proceedings of the 20th Workshop on Biomedical Language Processing*, pages 257–262, Online. Association for Computational Linguistics.

Ramesh Nallapati, Bowen Zhou, Cicero dos Santos, Çağlar Gulçehre, and Bing Xiang. 2016. Abstractive text summarization using sequence-to-sequence RNNs and beyond. In *Proceedings of The 20th SIGNLL Conference on Computational Natural Language Learning*, pages 280–290, Berlin, Germany. Association for Computational Linguistics.

Xiao Pan, Mingxuan Wang, Liwei Wu, and Lei Li. 2021. Contrastive learning for many-to-many multilingual neural machine translation. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 244–258, Online. Association for Computational Linguistics.

Romain Paulus, Caiming Xiong, and Richard Socher. 2018. A deep reinforced model for abstractive summarization. In *International Conference on Learning Representations*.

Mario Sänger, Leon Weber, and Ulf Leser. 2021. WBI at MEDIQA 2021: Summarizing consumer health questions with generative transformers. In *Proceedings of the 20th Workshop on Biomedical Language Processing*, pages 86–95, Online. Association for Computational Linguistics.

Shweta Yadav, Deepak Gupta, Asma Ben Abacha, and Dina Demner-Fushman. 2021a. Reinforcement learning for abstractive question summarization with question-aware semantic rewards. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 2: Short Papers)*, pages 249–255, Online. Association for Computational Linguistics.

Shweta Yadav, Mourad Sarrouti, and Deepak Gupta. 2021b. NLM at MEDIQA 2021: Transfer learning-based approaches for consumer question and multi-answer summarization. In *Proceedings of the 20th Workshop on Biomedical Language Processing*, pages 291–301, Online. Association for Computational Linguistics.

Nan Yang, Furu Wei, Binxing Jiao, Daxing Jiang, and Linjun Yang. 2021. xMoCo: Cross momentum contrastive learning for open-domain question answering. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 6120–6129, Online. Association for Computational Linguistics.

Guangtao Zeng, Wenmian Yang, Zeqian Ju, Yue Yang, Sicheng Wang, Ruisi Zhang, Meng Zhou, Jiaqi Zeng, Xiangyu Dong, Ruoyu Zhang, Hongchao Fang, Penghui Zhu, Shu Chen, and Pengtao Xie. 2020. MedDialog: Large-scale medical dialogue datasets. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 9241–9250, Online. Association for Computational Linguistics.

Jingqing Zhang, Yao Zhao, Mohammad Saleh, and Peter Liu. 2020. Pegasus: Pre-training with extracted

gap-sentences for abstractive summarization. In *International Conference on Machine Learning*, pages 11328–11339. PMLR.

ArgLegalSumm: Improving Abstractive Summarization of Legal Documents with Argument Mining

Mohamed Elaraby, Diane Litman
University of Pittsburgh
Pittsburgh, PA, USA
{mse30,dlitman}@pitt.edu

Abstract

A challenging task when generating summaries of legal documents is the ability to address their argumentative nature. We introduce a simple technique to capture the argumentative structure of legal documents by integrating *argument role labeling* into the summarization process. Experiments with pretrained language models show that our proposed approach improves performance over strong baselines.

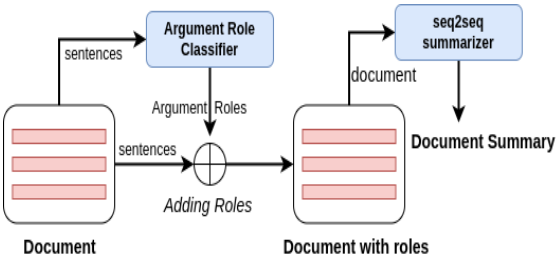

Figure 1: Overview of our approach.

1 Introduction

Abstractive summarization has made great progress by leveraging large pretrained language models such as BART (Lewis et al., 2020), T5 (Raffel et al., 2020), Pegasus (Zhang et al., 2020), and Longformer (Beltagy et al., 2020). These models leverage large scale datasets such as CNN-DailyMail (Hermann et al., 2015), PubMed (Cohan et al., 2018), and New York Times (Sandhaus, 2008). Unlike news and scientific texts, which contain specific formatting such as topic sentences and abstracts, legal cases contain implicit argument structure spreading across long texts (Xu et al., 2021). Current abstractive summarization models do not take into account the argumentative structure of the text, which poses a challenge towards effective abstractive summarization of legal documents.

In this work, we bridge the gap between prior research focusing on summarizing legal documents through extracting argument roles of legal text (Grover et al., 2003; Xu et al., 2021; Saravanan and Ravindran, 2010), and prior research focused on producing abstractive summaries of legal text (Feijo and Moreira, 2019; Bajaj et al., 2021). Our work proposes a technique that *blends argument role mining and abstractive summarization*, which hasn't been explored extensively in the literature.

Figure 1 describes the main flow of our approach, which decomposes the summarization process into two tasks. First, each sentence in the document is assigned an argument role by using an independent model. Then, the predicted roles are blended with the original document's sentences and fed into a sequence to sequence-based abstractive summarizer.

Our contributions are as follows: **(a)** We propose a simple technique to create an argument-aware neural abstractive summarizer. **(b)** We show the effectiveness of this technique in improving legal document summarization. **(c)** We make our code [1] and argument role annotations freely available[2].

2 Related Work

Legal Document Summarization. Prior research has mainly focused on extractive techniques (Galgani et al., 2015; Anand and Wagh, 2019; Jain et al., 2021), exploiting features such as the document structure and prior knowledge of legal terms to extract salient sentences that represent the summary of the legal text. Recent research has also shifted gears to abstractive techniques due to their superiority to extractive methods on automatic measures such as ROUGE (Feijo and Moreira, 2019). These abstractive techniques benefited from neural models such as pointer generator networks (See et al., 2017) (utilized in legal public opinion summarization (Huang et al., 2020)) and transformer-based se-

[1] https://github.com/EngSalem/arglegalsumm

[2] The data was obtained through an agreement with the Canadian Legal Information Institute (CanLII) (https://www.canlii.org/en/)

Proceedings of the 29th International Conference on Computational Linguistics, pages 6187–6194
October 12–17, 2022.

quence to sequence encoder-decoder architectures such as BART (Lewis et al., 2020) and Longformer (Beltagy et al., 2020) (employed to summarize long legal documents (Moro and Ragazzi, 2022)). However, the current abstractive approaches ignore the argumentative structure of the legal text. In our work, we combine both the rich argumentative structure of legal documents and state-of-the-art abstractive summarization models.

Argument Mining. Argument mining aims to represent the argumentative structure of a text in a graph structure that contains the argument roles and their relationship to each other. Constructing the graphs usually involves several steps: extracting argument units, classifying the argument roles of the units, and detecting the relationship between different argument roles. Recently, contextualized embeddings were employed to improve argument role labeling (Reimers et al., 2019; Elaraby and Litman, 2021). In many domains, argument roles are classified into *claims*, *major claims*, and *premises* as proposed in Stab and Gurevych (2014). Alternatively, Xu et al. (2021) proposed to classify the argument roles in legal documents to *Issues*, *Reasons*, and *Conclusions* which fits the legal text structure. We use the same set of legal argument role labels in our work, and use contextualized embeddings to automatically predict them.

Argument Mining and Summarization. Prior research integrating argument mining and summarization has mainly focused on extracting chunks of text that contain *key argument units* (Barker et al.; Bar-Haim et al., 2020; Friedman et al., 2021). Combining argument mining and abstractive summarization has received less attention in the literature. Recently, Fabbri et al. (2021) proposed a dialogue summarization dataset with argument information. In their work, the authors included argument information in abstractive summarization by linearizing the argument graph to a textual format, which is used to train an *encoder-decoder* summarization model. However, their proposed approach didn't improve over *encoder-decoder* baselines. We propose an alternative method that relies on argument roles only, which shows higher improvements over *encoder-decoder* baselines.

3 Dataset and Methods

3.1 Dataset [3]

Texts. Our dataset is composed of 1262 legal cases and summary pairs, obtained through an agreement with the Canadian Legal Information Institute. We split these pairs into training (1006 pairs, about 80%), validation (132 pairs, about 10%) and testing (124 pairs, about 10%) datasets.

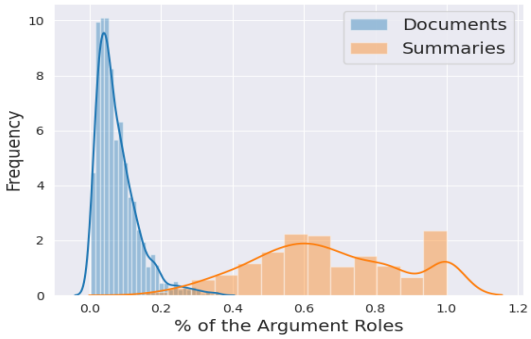

Figure 2: Argumentative sentence % in dataset.

Document Lengths. The maximum length of our input documents is $26k$ words, which motivates us to include *encoder-decoder* architectures like Longformer that can encode long documents.

Argument Role Annotations. The dataset was annotated prior to our study, using the IRC taxonomy of three legal argument roles described in Xu et al. (2021): **Issues** (legal questions which a court addressed in the document), **Reasons** (pieces of text which indicate why the court reached the specific conclusions) , and **Conclusions** (court's decisions for the corresponding issues). Figure 2 shows the distribution of the percentage of sentences annotated with an argumentative role across the articles and reference human summaries. We can see that while only a small percent of the sentences in the original articles are annotated as argument units, argumentative units dominate the reference summaries. Thus, we hypothesize that augmenting the summarization model with argument roles in the input text should improve the generated summaries.

3.2 Methods

Special Tokens Approach. We designate special marker tokens to distinguish between different argument roles. In prior research, DeYoung et al. (2021) used markers such as `<evidence>,</evidence>` to highlight evidence sentences in summarizing medical scientific documents, while Khalifa et al. (2021) used `<neg>,</neg>` to mark negation phrases in dialogue summarization. However, we explore the impact of changing token granularity by exper-

[3] See Appendix A for more detailed statistics.

Example
<IRC> He also found "on the strong balance of probabilities," that the late Mrs. Scott intended to make an inter vivos gift to Ms. Akerley. **</IRC>**
<Issue> [6] Mr. Comeau appeals, arguing that the probate court judge erred: **</Issue>**

Table 1: Different special tokens for argument roles.

imenting with two sets of special tokens. First, we introduce `<IRC>,</IRC>` to distinguish between argumentative and non-argumentative sentences. Second, we broaden the list of the proposed special tokens to differentiate between the three argument roles mentioned in Section 3.1. We assign each argument role two unique tokens to highlight its boundaries in the text, e.g., we use `<Reason>,</Reason>` to highlight the **reason** roles. Table 1 shows examples using tokens to highlight an argumentative sentence (top) versus a specific argumentative role (bottom).

Sentence-level Argument Role Mining. Our data's argument role annotation is at the sentence level, thus, we perform sentence-level classification experiments using the same data splits employed in summarization to detect argument roles.[4] We experiment with several contextualized embedding-based techniques, namely *BERT* (Devlin et al., 2019), *RoBERTa* (Liu et al., 2019), and *legalBERT* (Zheng et al., 2021). We employ these models to predict sentences' argument roles (issues, reasons, conclusions, or non-argumentative). Figure 3 shows that *legalBERT* achieved the best classification results. We achieved a macro average F1 of 63.4% in argument role classification and 71% in binary classification using *legalBERT*. Thus, we rely on its predictions to integrate argument roles into summarization below.

4 Experiments and Results

Our experiments are conducted in two settings: assuming argument roles are manually labeled (which we refer to as *oracle*) versus predicting argument role labels (referred to as *predicted*).

4.1 Baselines

We compare our proposed argument-aware summarization method to two sets of baselines[5]:

[4]See Appendix B for argument mining training details.

[5]See Appendix B for summarization training details.

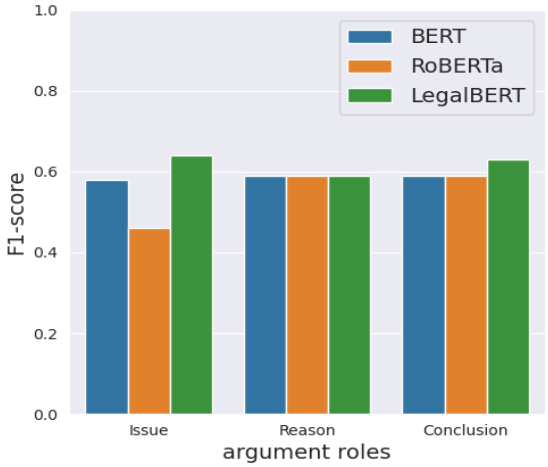

Figure 3: Argument role detection results.

Extractive Baseline. We employ the unsupervised method of Miller (2019). The model leverages BERT embeddings and k-means to extract salient sentences based on their proximity to cluster centroids.

Abstractive Baselines. Vanilla BART-Large refers to finetuning BART-large on our dataset. For **Vanilla LED-base**, similarly to BART, the model is finetuned using Longformer-base checkpoint.

4.2 Results and Discussion

Table 2 evaluates the results of the different summarization models using Rouge-1, Rouge-2, and Rouge-L scores.[6] We refer to BART and Longformer augmented with argument roles as **arg-BART-Large** and **arg-LED-base**, respectively. We use *2 markers* to denote the use of binary special tokens (i.e; `<IRC>,</IRC>`) and *6 markers* to refer to the full set of argument role tokens. We include two markers sets to examine whether it's necessary to include explicit argument roles or if it's sufficient to highlight argumentative text only.

We first evaluate the models augmented with the *manually labeled argument roles* to examine whether adding argument information has the potential to improve over the baselines. The *oracle* results in Table 2 show that **arg-LED-base** improves performance in terms of Rouge-1, Rouge-2, and Rouge-L (Lin, 2004) by approximately 1, 4, and 1.5 points, respectively, over the vanilla LED-base baseline when using the *6 markers*. The *2 markers* set showed marginal improvements on Rouge-1 and Rouge-L, but showed 4 Rouge-2 points improvement over the baseline. These results indi-

[6]See Appendix C for example generated summaries.

Setting	Experiment	Model	Rouge-1	Rouge-2	Rouge-L
		Unsupervised Extractive BERT	37.71	14.77	36.41
		Vanilla BART-Large	47.93	22.34	44.74
	Baselines	Vanilla LED-base	49.56	22.75	46.48
	arg-BART-Large	BART-Large + 2 markers	47.11	21.77	43.12
Oracle		BART-Large + 6 markers	46.80	22.14	44.14
		LED-base + 2 markers	49.64	**26.81**	46.70
	arg-LED-base	LED-base + 6 markers	**50.53**	26.31	**47.90**
		LED-base + 2 markers	49.50	*26.46*	46.60
Predicted	**arg-LED-base**	LED-base + 6 markers	*50.23*	26.29	*47.49*

Table 2: Summarization results on the test set. Best results **bolded**. Best results using predicted roles *italicized*.

Model	Rouge-1	Rouge-2	Rouge-L	Mean Summary Length
Vanilla LED-base	48.25	21.60	44.88	267
arg-LED-base + 2 markers	50.43	**27.76**	47.05	156
arg-LED-base + 6 markers	**50.73**	27.29	**47.30**	174

Table 3: Comparing Longformer (LED) summaries with sentences labeled as argumentative in reference summary.

cate that representing argument roles using fine-grained labels is the most effective in improving LED model output. In contrast, **arg-BART-Large** didn't show improvements over the vanilla BART-Large baseline. We hypothesize that this is due to the sparsity of the argumentative sentences in the input documents (recall Figure 2). Since Longformer can encode more words, it was likely able to capture more argument markers added to the input, increasing the model's ability to grasp the argument structure of the legal text. To validate this hypothesis, we analyze the positions of each argument role across the input articles. Figure 4 shows that the argument roles are distributed across the article and not centered around a unique position. This aligns with our hypothesis that the Longformer's encoding limit *(blue dashed line)* can cover significantly more roles when compared to the BART's encoding limit *(red dashed line)*.

Next, we evaluate the summarization using *predicted argument roles* obtained from our classifier (Section 3.2). We evaluate the Longformer summarization model only, since BART didn't show oracle improvements. The last two rows of Table 2 (the *predicted* results) show that including predicted argument roles showed consistent improvements with the manually labeled ones (oracle). The results showed a minimal drop in Rouge scores ranging from $0.02 - 0.41$ points when using the predicted argument roles both in the *6 markers* and *2 markers* cases, which indicates the effectiveness of our approach in practical scenarios.

Finally, to estimate the argumentativeness of the LED-based (oracle) summaries, we evaluate them against a summary containing only the sentences manually annotated as an IRC sentence in the reference summary.

Table 3 shows that adding argument role markers increases the overlap between the generated summaries and the argumentative sentence subset from the reference summaries, suggesting that our proposed model's gains are mainly obtained from an increase in argumentativeness of the generated summaries. The generated summaries from our **arg-LED-base** are lower in length compared to the baseline. We hypothesize that this is due to the focus on argument roles mainly, discarding some of the non-argumentative content. [7]

5 Conclusion and Future Work

We proposed to utilize argument roles in the abstractive summarization of legal documents to accommodate their argumentative structure. Our experiments with state-of-the-art encoder-decoder models showed that including argument role information can improve the ROUGE scores of summarization models capable of handling long

[7] See Appendix C for an illustrative example.

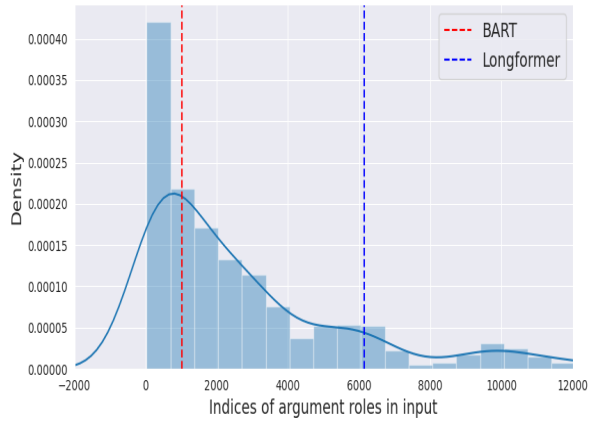

Figure 4: Argument position distribution in the input.

documents. Specifically, improved results were achieved using Longformer with input documents augmented with argument roles (highlighted using special marker tokens), and this finding was robust across two special token schemes. We also showed that using predicted argument roles showed consistent improvements to using the manually labeled ones. In future work, we plan to explore methods to unify argument mining and summarization to reduce the computational resources necessary to host two models.

Acknowledgements

This material is based upon work supported by the National Science Foundation under Grant No. 2040490 and by Amazon. We would like to thank the members of both the Pitt AI Fairness and Law Project and the Pitt PETAL group, as well as the anonymous reviewers, for valuable comments in improving this work.

References

Deepa Anand and Rupali Wagh. 2019. Effective deep learning approaches for summarization of legal texts. *Journal of King Saud University-Computer and Information Sciences*.

Ahsaas Bajaj, Pavitra Dangati, Kalpesh Krishna, Pradhiksha Ashok Kumar, Rheeya Uppaal, Bradford Windsor, Eliot Brenner, Dominic Dotterrer, Rajarshi Das, and Andrew McCallum. 2021. Long document summarization in a low resource setting using pretrained language models. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing: Student Research Workshop*, pages 71–80.

Roy Bar-Haim, Yoav Kantor, Lilach Eden, Roni Friedman, Dan Lahav, and Noam Slonim. 2020. Quantitative argument summarization and beyond: Cross-domain key point analysis. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 39–49.

Emma Barker, Monica Paramita, Adam Funk, Emina Kurtic, Ahmet Aker, Jonathan Foster, Mark Hepple, and Robert Gaizauskas. What's the issue here?: Task-based evaluation of reader comment summarization systems.

Iz Beltagy, Matthew E Peters, and Arman Cohan. 2020. Longformer: The long-document transformer. *arXiv preprint arXiv:2004.05150*.

Arman Cohan, Franck Dernoncourt, Doo Soon Kim, Trung Bui, Seokhwan Kim, Walter Chang, and Nazli Goharian. 2018. A discourse-aware attention model for abstractive summarization of long documents. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 2 (Short Papers)*, pages 615–621.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. Bert: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186.

Jay DeYoung, Iz Beltagy, Madeleine van Zuylen, Bailey Kuehl, and Lucy Wang. 2021. Ms^2: Multi-document summarization of medical studies. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 7494–7513.

Mohamed Elaraby and Diane Litman. 2021. Self-trained pretrained language models for evidence detection. In *Proceedings of the 8th Workshop on Argument Mining*, pages 142–147.

Alexander Richard Fabbri, Faiaz Rahman, Imad Rizvi, Borui Wang, Haoran Li, Yashar Mehdad, and Dragomir Radev. 2021. Convosumm: Conversation summarization benchmark and improved abstractive summarization with argument mining. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 6866–6880.

Diego Feijo and Viviane Moreira. 2019. Summarizing legal rulings: Comparative experiments. In *proceedings of the international conference on recent advances in natural language processing (RANLP 2019)*, pages 313–322.

Roni Friedman, Lena Dankin, Yufang Hou, Ranit Aharonov, Yoav Katz, and Noam Slonim. 2021.

Overview of the 2021 key point analysis shared task. In *Proceedings of the 8th Workshop on Argument Mining*, pages 154–164.

Filippo Galgani, Paul Compton, and Achim Hoffmann. 2015. Summarization based on bi-directional citation analysis. *Information processing & management*, 51(1):1–24.

Claire Grover, Ben Hachey, Ian Hughson, and Chris Korycinski. 2003. Automatic summarisation of legal documents. In *Proceedings of the 9th international conference on Artificial intelligence and law*, pages 243–251.

Karl Moritz Hermann, Tomas Kocisky, Edward Grefenstette, Lasse Espeholt, Will Kay, Mustafa Suleyman, and Phil Blunsom. 2015. Teaching machines to read and comprehend. *Advances in neural information processing systems*, 28:1693–1701.

Yuxin Huang, Zhengtao Yu, Junjun Guo, Zhiqiang Yu, and Yantuan Xian. 2020. Legal public opinion news abstractive summarization by incorporating topic information. *International Journal of Machine Learning and Cybernetics*, 11(9):2039–2050.

Deepali Jain, Malaya Dutta Borah, and Anupam Biswas. 2021. Automatic summarization of legal bills: A comparative analysis of classical extractive approaches. In *2021 International Conference on Computing, Communication, and Intelligent Systems (ICCCIS)*, pages 394–400. IEEE.

Muhammad Khalifa, Miguel Ballesteros, and Kathleen Mckeown. 2021. A bag of tricks for dialogue summarization. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 8014–8022.

Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Veselin Stoyanov, and Luke Zettlemoyer. 2020. Bart: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7871–7880.

Chin-Yew Lin. 2004. Rouge: A package for automatic evaluation of summaries. In *Text summarization branches out*, pages 74–81.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Roberta: A robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692*.

Derek Miller. 2019. Leveraging bert for extractive text summarization on lectures. *arXiv preprint arXiv:1906.04165*.

Gianluca Moro and Luca Ragazzi. 2022. Semantic self-segmentation for abstractive summarization of long legal documents in low-resource regimes. In *Proceedings of the Thirty-Six AAAI Conference on Artificial Intelligence, Virtual*, volume 22.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J Liu. 2020. Exploring the limits of transfer learning with a unified text-to-text transformer. *Journal of Machine Learning Research*, 21:1–67.

Nils Reimers, Benjamin Schiller, Tilman Beck, Johannes Daxenberger, Christian Stab, and Iryna Gurevych. 2019. Classification and clustering of arguments with contextualized word embeddings. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 567–578.

Evan Sandhaus. 2008. The new york times annotated corpus. *Linguistic Data Consortium, Philadelphia*, 6(12):e26752.

M Saravanan and Balaraman Ravindran. 2010. Identification of rhetorical roles for segmentation and summarization of a legal judgment. *Artificial Intelligence and Law*, 18(1):45–76.

Abigail See, Peter J Liu, and Christopher D Manning. 2017. Get to the point: Summarization with pointer-generator networks. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1073–1083.

Christian Stab and Iryna Gurevych. 2014. Identifying argumentative discourse structures in persuasive essays. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 46–56.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Rémi Louf, Morgan Funtowicz, et al. 2019. Huggingface's transformers: State-of-the-art natural language processing. *arXiv preprint arXiv:1910.03771*.

Huihui Xu, Jaromir Savelka, and Kevin D Ashley. 2021. Toward summarizing case decisions via extracting argument issues, reasons, and conclusions. In *Proceedings of the eighteenth international conference on artificial intelligence and law*, pages 250–254.

Jingqing Zhang, Yao Zhao, Mohammad Saleh, and Peter Liu. 2020. Pegasus: Pre-training with extracted gap-sentences for abstractive summarization. In *International Conference on Machine Learning*, pages 11328–11339. PMLR.

Lucia Zheng, Neel Guha, Brandon R. Anderson, Peter Henderson, and Daniel E. Ho. 2021. When does pretraining help? assessing self-supervised learning for law and the casehold dataset. In *Proceedings of the 18th International Conference on Artificial Intelligence and Law*. Association for Computing Machinery.

A Data statistics

A.1 Length statistics

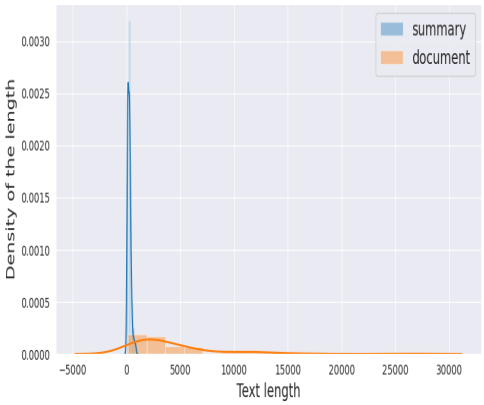

Figure 5: Distribution of article and summary length.

Figure 5 shows the distribution of document and summary lengths. The summaries' lengths are centered around a mean of 255 words, with a maximum length of 850 words. The $90th$ percentile of summary length is 490 words. Thus we chose the maximum length of generated summary in our hyperparameters to be set to 512 words. Unlike the summaries, the documents are more spread across the distribution. In our analysis, we found that the mean document length is 4180 words, while the maximum document length is 26235 words.

A.2 Argument role distribution

While they are essential to legal cases, argument roles represent a small percentage of the document. Figure 6 shows the high imbalance of the manually annotated argumentative versus non-argumentative sentences in our training set, which poses a challenge in building a sentence level classifier of argument roles. In our analysis we found that the non-argumentative sentences count is approximately $1000\times$ the argumentative sentences, which we use to adjust class weights in our learning objective.

B Training details and hyperparameters

All experiments use the model implementations provided in the *Huggingface library* (Wolf et al., 2019). We initialize all our models with the same learning rate of $2e^{-5}$. We train both our summarization and argument role classification models for 10 epochs with early stopping with 3 epoch patience. For training summarization models, we set the maximum summary length to 512 words. We truncate

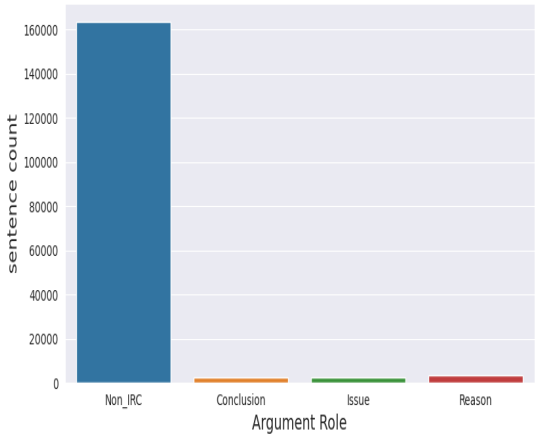

Figure 6: Count of argument roles across training set. *non_IRC* refers to non-argumentative sentences.

the input length to 1024 words for the *BART* model while truncating the input length to 6144 words for the *Longformer* due to our GPU limitation [8]. We pick our best model based on its *ROUGE-2* (Lin, 2004) score on the validation set. For the classification models introduced in Section 3.2, due to the high imbalance of our argumentative labels, we introduce fixed class weights to our cross-entropy loss. For argumentative sentences, we modify the *cross-entropy* weight to be 1000 compared to 1 for non-argumentative sentences. We chose these weights based on label distribution in our training set described in Figure 6. Our best model is chosen based on the F1 score on the validation set.

C Effect of argument roles on generated summaries

Table 4 shows an example of generated summaries with adding special tokens and without the special tokens.

[8] We use Quadro RTX 5000 which has 16 GB RAM.

Reference summary	Vanilla LED-base	arg-LED-base "6 markers"
The plaintiffs claim for crop damage caused by the defendant's cattle entering the plaintiff's canola field. The defendant denies he is responsible or negligent, but admits his cattle were in the field. HELD: Damages were awarded in the amount of $2,533.45. The court valued the loss at $3,052.36. It then deducted 2% attributed to wild animal damage, 5% for the plaintiffs failure to calculate actual yield from the rest of the crop, 5% that the plaintiffs would have paid in dockage and 5% for the cost of production. Civil liability for crop damage caused by cattle flows from the Stray Animals Act. A cattle owner is strictly liable for damages caused by his straying cattle. This liability may possibly even be absolute and only an act of god may serve as a defence. The issue here is the quantum of the damage. The fact that the defendant was refused access to the plaintiff's property to repair the fence has no bearing on liability in this case. An independent adjuster assessed the damage at 557 bushels. The court found on the facts that the damage was caused exclusively by the defendant's cattle on several occasions.	The plaintiff's claim against the defendant is for crop damage sustained as a result of the defendant's cattle getting into the plaintiff's canola field in late July 1999. The plaintiff admits that his cattle were in the Plaintiff's crop on one occasion, but says that there is no evidence that their cattle are in the canola on any other occasion, and that such loss as was caused by his cattle is minimal. HELD: Judgment in the amount of $2,533.45 was granted in favour of the plaintiffs, together with pre-judgment interest from January 2000 and costs. The present s.27 (1) has been in the Act since 1977, when the Act was substantially re-written, apparently in response to the increasing urbanization of the province. Under both the present and pre-1977 statutes, the legal situation was the same – the cattle owner is strictly liable for damage caused by the straying cattle. The fact remains that the cattle did move onto the field unbeknownst to the defendant, his spouse, his brother, or his hired help. The Carrot River did not impede the cattle from moving into the field on more than one occasion and other cattle were not in the field.	The plaintiff's claim against the defendant was for crop damage sustained as a result of his cattle getting into the plaintiff's canola field in late July 1999. The defendant admits that his cattle were in the Plaintiff's crop on one occasion, but says that there is no evidence that any other cattle were there. HELD: Judgment in the amount of $2,533.45 was granted in favour of the plaintiffs, together with pre-judgment interest from January 2000 and costs. Under both the present s.27 (1) and the pre-1977 statutes, the legal situation was the same – the cattle owner is strictly liable for damage caused by his straying cattle.

Table 4: Example of generated summaries with **Vanilla LED-base** and **arg-LED-base** versus reference summary.

Semantic Overlap Summarization among Multiple Alternative Narratives: an Exploratory Study

Naman Bansal, Mousumi Akter and **Shubhra Kanti Karmaker ("Santu")**
Big Data Intelligence (BDI) Lab
Department of Computer Science and Software Engineering
College of Engineering, Auburn University
{nbansal, mza0170, sks0086}@auburn.edu

Abstract

In this paper, we introduce an important yet relatively unexplored NLP task called **S**emantic **O**verlap **S**ummarization (**SOS**), which entails generating a single summary from multiple alternative narratives which can convey the *common information* provided by those narratives. As no benchmark dataset is readily available for this task, we created one by collecting $2,925$ alternative narrative pairs from the web and then, went through the tedious process of manually creating 411 different reference summaries by engaging human annotators. As a way to evaluate this novel task, we first conducted a systematic study by borrowing the popular *ROUGE* metric from text-summarization literature and discovered that *ROUGE* is not suitable for our task. Subsequently, we conducted further human annotations to create 200 document-level and $1,518$ sentence-level ground-truth *overlap labels*. Our experiments show that the sentence-wise annotation technique with three overlap labels, i.e., {Absent (A), Partially-Present (PP), and Present (P)}, yields a higher correlation with human judgment and higher inter-rater agreement compared to the ROUGE metric.

1 Introduction

In this paper, we look deeper into the challenging yet relatively under-explored area of automatic summarization of multiple alternative narratives with different perspectives. To be more specific, we formally introduce a new NLP task called **S**emantic **O**verlap **S**ummarization (**SOS**) from multiple alternative narratives and conduct a systematic study of this task by creating a benchmark dataset as well as exploring how to accurately evaluate this task. *SOS* essentially means the task of *summarizing the overlapping information* present in multiple alternate narratives by cross-verifying their information contents against each other. Computationally, our research question is the following:

Given two distinct narratives N_1 and N_2 of an event e, how can we automatically generate a single summary about e which conveys the common information provided by both N_1 and N_2?

Multiple alternative narratives appear frequently in a variety of domains, including education (Somasundaran et al., 2018), the health sector (Bijoy et al., 2021), businesses intelligence (Azeroual and Theel, 2018), content analysis (Hassan et al., 2020; Karmaker Santu et al., 2018b) and privacy (Wilson et al., 2018). Therefore, automatic summarization of multiple-perspective narratives has become a pressing need in this information explosion era and can be highly useful for digesting such multi-narratives at scale and speed.

Figure 1 shows a toy example of the *SOS* task, where both articles cover the same event related to "abortion". However, they report from different political perspectives, i.e., one from the *left* wing and the other from the *right* wing. For greater visibility, "Left" and "Right" wing reporting biases are represented by *blue* and *red* text, respectively. *Green* text denotes the common information in both news articles. The goal of the *SOS* task is to generate a summary that conveys the common/overlapping information provided by the *green* text.

At first glance, the *SOS* task may appear similar to a traditional multi-document summarization task where the goal is to provide an overall summary of the (multiple) input documents. However, the difference is that, for *SOS*, the goal is to provide summarized content with an additional constraint, i.e., the commonality criteria. There is no current baseline method or an existing dataset that exactly matches our task; more importantly, it is unclear which one is the right evaluation metric to properly evaluate this task. As a starting point, we frame *SOS* as a constrained seq-to-seq task where the goal is to generate a summary from two input documents that conveys the overlapping information present in both input text documents. However, the bigger challenge we need to address first is the following: *1) How can we evaluate this task?* and *2) How*

Proceedings of the 29th International Conference on Computational Linguistics, pages 6195–6207
October 12–17, 2022.

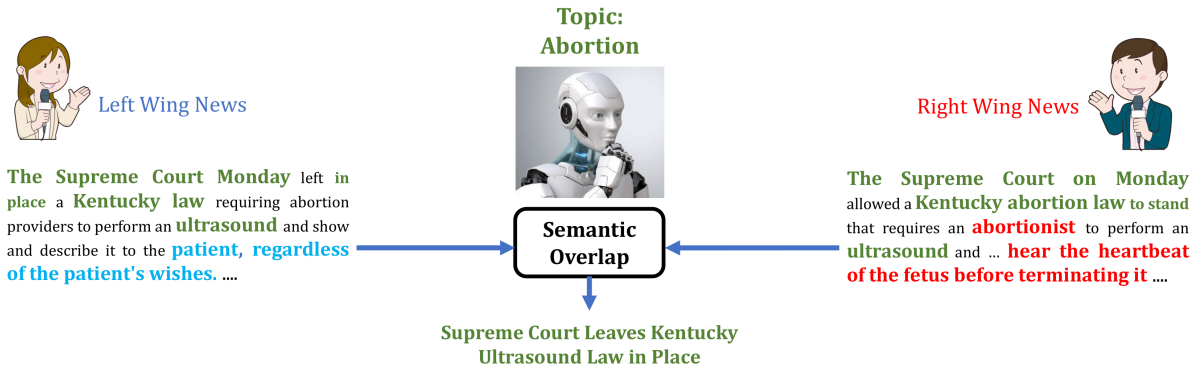

Figure 1: A toy example of *Semantic Overlap Summarization* (**SOS**) Task (from multiple alternative narratives). Here, an abortion issue-related event has been reported by two news media (left-wing and right-wing). "Green" Text denotes the common information from both news media, while "Blue" and "Red" text denotes the unique perspectives of *left* and *right* wing. Some real examples from the benchmark dataset are provided in the Table 3.

to create a benchmark dataset for this task? To address these challenges, we make the following contributions in this paper.

1. We formally introduce *Semantic Overlap Summarization* (**SOS**) (from multiple alternative narratives) as a new NLP task and conduct a systematic study by formulating it as a constrained summarization problem.
2. We create and release the first benchmark dataset consisting of $2,925$ alternative narrative pairs for facilitating research on the *SOS* task. Also, we went through the tedious process of manually creating 411 different ground-truth reference summaries and conducted further human annotations to create 200 document-level and $1,518$ sentence-level ground-truth *overlap labels* to construct the benchmark dataset.
3. As a starting point, we experiment with *ROUGE*, a widely popular metric for evaluating text summarization tasks, and demonstrate that *ROUGE* is NOT suitable for the evaluation of *SOS* task.
4. We do further human experiments, which show that sentence-level evaluation is the proper way to evaluate the *SOS* task. It also improves the inter-rater agreement compared to the traditional *ROUGE* metric and shows a higher correlation with human judgments.

2 Related Works

As *SOS* can be viewed as a multi-document summarization task with additional commonality constraints, text summarization literature is the most relevant to our work. Over the years, many paradigms for document summarization have been explored (Zhong et al., 2019). The two most popular among them are *extractive* approaches (Cao et al., 2018; Narayan et al., 2018; Wu and Hu, 2018; Zhong et al., 2020) and *abstractive* approaches (Bae et al., 2019; Hsu et al., 2018; Liu et al., 2017; Nallapati et al., 2016). Some researchers have also tried combining extractive and abstractive approaches (Chen and Bansal, 2018; Hsu et al., 2018; Zhang et al., 2019).

Recently, encoder-decoder-based neural models have become really popular for abstractive summarization (Rush et al., 2015; Chopra et al., 2016; Zhou et al., 2017; Paulus et al., 2017). It has become even more prevalent to train a general language model on a huge corpus of data and then transfer/fine-tune it for the summarization task (Radford et al., 2019; Devlin et al., 2019; Lewis et al., 2019; Xiao et al., 2020; Yan et al., 2020; Zhang et al., 2019; Raffel et al., 2019). Summary length control for abstractive summarization has also been studied (Kikuchi et al., 2016; Fan et al., 2017; Liu et al., 2018; Fevry and Phang, 2018; Schumann, 2018; Makino et al., 2019). In general, multiple document summarization (Goldstein et al., 2000; Yasunaga et al., 2017; Zhao et al., 2020; Ma et al., 2020; Meena et al., 2014) is more challenging than single document summarization. However, the *SOS* task is different from traditional multi-document summarization tasks in that the goal here is to summarize content with an *overlap* constraint, i.e., the output should only contain the common information from both input narratives.

Alternatively, one could aim to recover verb-predicate alignment structure (Roth and Frank, 2012; Xie et al., 2008; Wolfe et al., 2013) from

a sentence and further, use this structure to compute the overlapping information (Wang and Zhang, 2009; Shibata and Kurohashi, 2012). Sentence Fusion is another related area that aims to combine the information from two given sentences with some additional constraints (Barzilay et al., 1999; Marsi and Krahmer, 2005; Krahmer et al., 2008; Thadani and McKeown, 2011). A related but simpler task is to retrieve parallel sentences (Cardon and Grabar, 2019; Nie et al., 1999; Murdock and Croft, 2005) without performing an actual overlap summary generation. However, these approaches are more targeted toward individual sentences and do not directly translate to arbitrarily long documents. Thus, the *SOS* task is still an open problem and there is no existing dataset, method, or evaluation metric that has been systematically studied.

Along the evaluation dimension, *ROUGE* (Lin, 2004) is perhaps the most commonly used metric today for evaluating automated summarization techniques; due to its simplicity and automation. However, *ROUGE* has been criticized a lot for primarily relying on lexical overlap (Nenkova, 2006; Zhou et al., 2006; Cohan and Goharian, 2016; Akter et al., 2022) of n-grams. As of today, around 192 variants of *ROUGE* are available (Graham, 2015) including *ROUGE* with word embedding (Ng and Abrecht, 2015) and synonym (Ganesan, 2018), graph-based lexical measurement (ShafieiBavani et al., 2018), Vanilla *ROUGE* (Yang et al., 2018) and highlight-based *ROUGE* (Hardy et al., 2019). However, there has been no study yet as to whether the *ROUGE* metric is appropriate for evaluating the *SOS* task, which is one of the central goals of our work.

3 Motivation and Applications

Multiple alternative narratives are frequent in a variety of domains and, therefore, have direct implications in technical areas such as Information Retrieval/Search Engines, Question Answering, Machine Translation, etc. Below are a few examples of use cases.

Peer-Reviewing: Given two peer-review narratives for an article, the *SOS* task can generate a summary of portions of the narratives that agree with each other, which can help prepare a meta-review quickly.

Security and Privacy: *SOS* task can enable real-world users to quickly conduct a comparative analysis of multiple privacy policies by mining and summarizing overlapping clauses from those policies. Thus, it can help users make informed decisions while choosing from various alternative web services.

Health Sector: *SOS* can be used to compare clinical notes in patient records to perform a comparative analysis of patients with the same diagnosis/treatment. For example, *SOS* can be applied to the clinical notes of two (or more) patients who went through the same treatments to assess the effectiveness of the treatment.

Military Intelligence: If A and B are two intelligence reports related to a mission coming from two human agents, the *SOS* task can help cross-verify the claims in each report w.r.t. the other, thus, *SOS* can be used as an automated claim-verification tool.

Computational Social Science and Journalism: Assume that two news agencies (with different political biases) are reporting the same real-world event and their bias is somewhat reflected in the articles they write. If A and B are two such news articles, then the *SOS* output will likely surface the facts (common information) about the event.

Below are some of the potential applications of the *SOS* task in various technical areas.

Information Retrieval/Search Engines: One could summarize the common information in the multiple results fetched by a search engine for a given query and show it in a separate box to the user. This would immensely help them to quickly parse the desired information rather than going through each article individually. If they wish, they could further explore the specific articles for more details.

Question Answering: One could apply *SOS* to summarize the common information/answer from multiple documents pertinent to a given question. This will help formulate a more accurate answer by consulting multiple sources.

Robust Translation: Suppose you have multiple machine translator models which translate a given document from language A to language B. One could further apply the *SOS* to different translated documents and get a more robust translation by summarizing their semantic overlap.

In general, SOS task could be employed in any setting where we require comparative text analysis.

4 Problem Formulation

What is *Semantic Overlap Summarization*? This is indeed an open question and there is no single

AllSides Dataset: Statistics				
Split	#words (per docs)	#sents (per docs)	#words (per reference)	#sents (per reference)
Train	1613.69	66.70	67.30	2.82
Test	959.80	44.73	65.46/38.06/21.72/32.82	3.65/2.15/1.39/1.52

Table 1: Statistics for the Training and Testing dataset. Two input narratives are concatenated to compute the statistics. Four numbers for reference (#words/#sents) in the Test split correspond to the 4 reference overlap summaries. Our test dataset contains 137 samples, wherein each sample has 4 ground truth references. Out of these 4 references, *one* summary is provided by AllSides, and 3 of them were manually written by 3 human annotators. Thus, we generated 3*137 = 411 references in total.

correct answer. To simplify notations, let us stick to having only two documents D_A and D_B as our input since it can easily be generalized in case of more documents using *SOS* repeatedly. Also, let us define the output as $D_O \leftarrow D_A \cap_O D_B$. A human would mostly express the output in the form of natural language, and this is why we frame the *SOS* task as a constrained multi-seq-to-seq (text generation) task, where the output text only contains information that is present in both the input documents. We also argue that brevity (minimal repetition) is a desired property of *Semantic Overlap Summarization*. For example, if a particular piece of information or quote is repeated twice in both the documents, we don't necessarily want it to be present in output overlap summary two times. The output can either be extractive summary or abstractive summary or a mixture of both, as per the use case. This task is inspired by the set-like intersection operator as envisioned by (Karmaker Santu et al., 2018a) and the aim of this work is to summarize the overlapping information in an abstract fashion. Additionally, *SOS* should follow the *commutative* property, i.e. $D_A \cap_O D_B = D_B \cap_O D_A$.

5 The Benchmark Dataset

As mentioned in section 1, there is no existing dataset that we could readily use to evaluate the *SOS* task[1]. To address this challenge, we collected data from AllSides.com. AllSides is a third-party online news forum that exposes people to news and information from all sides of the political spectrum so that the general people can get an "unbiased" view of the world. To achieve this, AllSides displays each day's top news stories from news media widely-known to be affiliated with different sides of the political spectrum including "Left" (e.g., New York Times, NBC News), and "Right" (e.g., Townhall, Fox News) wing media. AllSides also provides its own *factual* description of the reading material, labelled as "Theme" so that readers can see the so-called "neutral" point-of-view. Table 1 gives an overview of the dataset statistics created by crawling from AllSides.com, which consists of news articles (from at least one "Left" and one "Right" wing media) covering $2,925$ events in total and also having a minimum length of "theme-description" to be 15 words. Given two narratives ("Left" and "Right"), we used the theme description as a proxy for ground-truth reference summaries. We divided this dataset into testing data (described next) and training data (remaining samples) [see Table 1]. Table 2 shows the different attributes of the same AllSides dataset.

Feature	Description
theme	headlines by AllSides
theme-description	news description by AllSides
right/left head	right/left news headline
right/left context	right/left news description

Table 2: Overview of dataset scraped from AllSides. AllSides is licensed under a Creative Commons Attribution-NonCommercial 4.0 International License.

Testing Dataset and Human Annotations[2]: We engaged human volunteers to thoroughly annotate our testing samples (narrative pairs) in order to create multiple reference overlap summaries for each pair. This helped in creating a comprehensive testing benchmark for more rigorous evaluation. Specifically, we randomly sampled 150 narrative pairs describing 150 unique events (each pair consists of one narrative from the "Left" wing and one

[1]Multi-document summarization datasets can not be utilized in this scenario as their reference summaries do not follow the semantic overlap constraint.

[2]Dataset and manual annotations can be found at: https://karmake2.github.io/publications/

Narrative Pair Example # 1	
Narrative 1: N_1	**Narrative 2: N_2**
WASHINGTON – U.S. intelligence and law enforcement agencies have confirmed that President Donald Trump's campaign aides and associates had constant contact with Russian intelligence officials before the election, directly contradicting public statements made by top administration officials. On Jan. 15, shortly before Trump took office, Vice President Mike Pence repeatedly said on television that there were zero contacts between the campaign and Russian officials. . . . Pence also answered "of course not" when asked a similar question that day by "Fox News Sunday" host Chris Wallace . . . Trump himself also denied these interactions . . . "There's nothing that would conclude me that anything different has changed with respect to that time period," Spicer said. . . .	President Trump said Wednesday that new reports saying his associates had contact with Russian officials during last year's campaign are "non-sense" and accused the U.S. intelligence community of illegally leaking information to news outlets. "This Russian connection non-sense is merely an attempt to cover-up the many mistakes made in Hillary Clinton's losing campaign," Mr. Trump tweeted. . . . Among those supposedly communicating with Russian nationals was former Trump campaign chairman Paul Manafort, the report said. Mr. Manafort denied that he ever knowingly talked to any intelligence official "or anyone

Reference Overlap Summaries			
A_1	**A_2**	**A_3**	**AllSides**
President Trump and the Trump administration deny allegations that advisers close to Trump were in constant communication during the campaign with Russians known to US intelligence.	Trump denied climas that advisers close to him were in "constant communication during the campaign with Russians known to US intelligence.	Donald Trump and his group claimed that there is no contact with Russian officials during his last year's campaign.	Russian intelligence officials made repeated contact with members of President Trump's campaign staff, according to new reports that cite anonymous U.S. officials. American agencies were concerned about the contacts but haven't seen proof of collusion between the campaign and the Russian security apparatus.

Narrative Pair Example # 2	
Narrative 1: N_1	**Narrative 2: N_2**
John McCain is out of McConnell's clutches for a week or two. While Sen. John McCain remains in Arizona recovering from Friday's craniotomy, surgery to remove a 5 cm blood clot from above his left eye, business will not go on as usual in Washington. Majority Leader Mitch McConnell, who has to have every Republican senator voting to have a prayer of passing Trumpcare, has postponed the vote for the week or two (more likely two) that McCain's recovery will take. That means there's more time for opponents to fight this thing, from the side of all of us trying to keep 22 million people from losing insurance and from the other side. . . . With both Paul and Sen. Susan Collins (R-ME) solid "no" votes on the bill, opponents only need one more out of the eight or so who've expressed reservations about the bill and the secretive, exclusive process McConnell	WASHINGTON - The Republican effort to repeal and replace Obamacare faces a major setback as Sen. John McCain, R-Ariz., left the nation's capital for surgery on his eye. Over the weekend, Senate Majority Leader Mitch McConnell, R-Ky., announced the scheduled Better Care Act vote would be delayed indefinitely because of McCain's absence. Subsequently, the Congressional Budget Office (CBO) also delayed its analysis of the bill. With two Republican senators opposed to the measure, McConnell needs at least 50 "yes" votes to pass it. Sen. Rand Paul, R-Ky., says the bill, which keeps taxes on investments and other pieces of Obamacare, doesn't go far enough. Moderate Sen. Susan Collins, R-Maine, is also withholding her support because it would slow the rate of growth in spending on Medicaid. . . .

Reference Overlap Summaries			
A_1	**A_2**	**A_3**	**AllSides**
Sen. John McCain remains in Arizona recovering from eye surgery. Senate Majority Leader Mitch McConnell postponed the vote due to McCain's absence. Two Republican senators opposed to the bill. Possibility of bill failing.	Sen. John McCain remains unavailable because of the surgery on his eye. Senate Majority Leader Mitch McConnell delayed the vote in his absence. Sen. Rand Paul and Sen. Susan Collins said "no" votes on the bill.	Senate Majority Leader Mitch McConnell, R-Ky., announced the scheduled health care vote would be delayed indefinitely because of McCain's absence.	Senate Majority Leader Mitch McConnell, R-Ky., announced the scheduled Better Care Act vote would be delayed indefinitely because of McCain's absence.

Table 3: Some examples of *SOS* references from 3 human annotators ($\mathbf{A_i}$) and the AllSides "theme-description" for a given narrative pair $\{N_1, N_2\}$. (. . .) denotes some sentences which for not shown for brevity. More examples can be found over here. Having multiple human annotators is critical to perform robust evaluation, but it is laborious and time-consuming on humans' part. This also shows that the lack of available datasets is a huge challenge for the SOS task.

from the "Right" wing, thus 300 narratives in total) and then asked 3 humans to write a summary of common information present in both narratives describing each of the 150 events.

After the first round of annotations, we immediately observed a discrepancy among the three annotators in terms of the *real* definition of "common/overlapping information". For example, one annotator argued that the reference summary should be non-empty as long as there is an overlap between two narratives along at least one of the *5W1H* facets (Who, What, When, Where, Why, and How), while another annotator argued that overlap in only one facet is not enough to decide whether

there is indeed enough semantic overlap between the two narratives and reference summary should be left empty in such cases. As an example, one of the annotators wrote only "Donald Trump" as the reference summary for a couple of cases where the actual narratives were substantially different except for "Donald Trump" being the only common entity, while others had those cases marked as "*empty*".

To mitigate this issue, we only retained the narrative pairs where at least two of the annotators wrote a minimum of 15 words as their reference summaries, assuming that a human-written summary will contain 15 words or more only in cases where there is indeed a *significant* overlap between the two original narratives. This filtering step gave us a test set with 137 narrative pairs where each sample had 4 reference summaries, *one* from AllSides and *three* from human annotators, resulting in a total of 548 reference summaries. A couple of sample narrative pairs are shown in Table 3 along with the human annotations.

6 Evaluating SOS Task using ROUGE

As *ROUGE* (Lin, 2004) is the most popular metric used today for evaluating summarization tasks, we first conducted a case study with *ROUGE* as the evaluation metric for the *SOS* task. For methods, we experimented with multiple SoTA pretrained abstractive summarization models as *naive baselines* for *Semantic-Overlap Summarizer (SOS)*. These models are: 1) **BART** (Lewis et al., 2019), fine-tuned on CNN and multi-English Wiki news datasets, 2) **Pegasus** (Zhang et al., 2019), fine-tuned on CNN and Daily Mail dataset, and 3) **T5** (Raffel et al., 2019), fine-tuned on multi-English Wiki news dataset. As our primary goal is to construct a benchmark dataset for the *SOS* task and explore how to accurately evaluate this task, experimenting with only 3 abstractive summarization models is not a barrier to our work. Proposing a custom method fine-tuned for the *Semantic-Overlap* task is an orthogonal goal to this work and we leave it as future work. Also, we shall use the phrases "summary" and "overlap-summary" interchangeably from here. To generate the summary, we concatenate a narrative pair and feed it directly to the model.

For evaluation, we first evaluated the machine-generated overlap summaries for the 137 manually annotated testing samples using the ROUGE metric by following the procedure mentioned in Lin (2004) to compute the ROUGE-F_1 scores against multiple reference summaries. More precisely, since we have 4 reference summaries, we got 4 precision, recall pairs which are used to compute the corresponding F_1 scores. For each sample, we took the max of these four F_1 scores and averaged them out across the test dataset (see Table 4).

Model	R1	R2	RL
BART	40.73	25.97	29.95
T5	38.50	24.63	27.73
Pegasus	46.36	29.12	37.41

Table 4: Average ROUGE-F_1 Scores for all the test models across test dataset. For a particular sample, we take the maximum value out of the 4 F_1 scores corresponding to the 4 reference summaries.

Implementation Details: For generating summaries, we used off-the-shelf models in our experiments with default settings for summarization task following the Huggingface repo. Apart from this, we set the min and max length parameters to 10 and 300, respectively, based on our dataset. All the models are publicly available with details of the source. For ROUGE computation, we followed the implementation from the HuggingFace repo with the following parameters: $\{use\_stemmer = True, bootstrap\_aggregation = False\}$. Apart from this, we just used a sentence tokenizer from nltk library with English to create the input tokens. So, most of the method and ROUGE implementations are already publicly available. As such, there was no training involved in our experiments, but we still made use of the GPU (NVIDIA Quadro RTX 5000 with 16 GB of memory) to generate summaries using these models. Table 5 shows the summarization models and the number of parameters used in our experiments.

Model	#Parameters
BART	$\sim$ 406 M
T5	$\sim$ 223 M
Pegasus	$\sim$ 571 M

Table 5: Models and their corresponding number of parameters used in our experiments.

Results and Findings: We computed Pearson's correlation coefficients (using the scipy package) between each pair of ROUGE-F_1 scores obtained using all of the 4 reference overlap summaries (3

Pearson's Correlation Coefficients									
	R1			**R2**			**RL**		
	I_1	I_2	I_3	I_1	I_2	I_3	I_1	I_2	I_3
I_2	**0.62**	—		**0.65**	—		**0.69**	—	
I_3	**0.3**	**0.38**	—	**0.27**	**0.37**	—	**0.27**	**0.44**	—
I_4	**0.17**	**0.34**	**0.34**	0.14	**0.33**	**0.21**	**0.18**	**0.35**	**0.33**
Average		**0.36**			**0.33**			**0.38**	

Table 6: Max (across 3 models) Pearson's correlation between the F_1 ROUGE scores corresponding to different annotators. Here I_i refers to the ith annotator where $i \in \{1, 2, 3, 4\}$ and the "Average" row represents the average correlation of the max values across annotators. Boldface values are statistically significant at p-value < 0.05. For 5 out of 6 annotator pairs, the correlation values are quite small (≤ 0.50), thus, implying the poor inter-rated agreement with regards to the ROUGE metric.

human written summaries and 1 AllSides theme description) to test the robustness of *ROUGE* metric for evaluating the *SOS* task. The corresponding correlations are shown in table 6. For each annotator pair, we report their maximum (across 3 models) correlation value. The average correlation value across annotators is 0.36, 0.33 and 0.38 for R1, R2 and RL, respectively, suggesting that the ROUGE metric demonstrates high variance across multiple human-written overlap-summaries and thus, *unreliable*.

7 Can We Do Better than ROUGE?

Section 6 shows that the ROUGE metric is unstable across multiple reference overlap-summaries. Therefore, an immediate question is: Can we come up with a better metric than ROUGE? To investigate this question, we started by manually assessing the machine-generated overlap summaries to check first whether humans agree among themselves or not, i.e., whether human annotators can reach a consensus or not.

7.1 Different trials of Human Judgement

Assigning a Single Numeric Score: As an initial trial, we decided to first label 25 testing samples using two human annotators (we refer to them as label annotators, L_1 and L_2). Both label annotators read each of the 25 narrative pairs as well as the corresponding system-generated overlap summary (generated by fine-tuned BART) and assigned a numeric score between 1-10 (inclusive). This number reflects their judgment/confidence about how accurately the system-generated summary captures the *actual* overlap of the two input narratives. Note that, *the reference overlap summaries were not included in this label annotation process and the label-annotators judged the system-generated summary exclusively with respect to the input narratives*. To quantify the agreement between human scores, we computed the Kendall rank correlation coefficient (or Kendall's Tau) between two annotator labels since these are ordinal values. We used an open-source scipy package for computing Kendall's Tau correlation. However, to our disappointment, the correlation value was 0.20 with the p-value being 0.22[3]. This shows that even human annotators are disagreeing among themselves and we need to come up with a better labelling guideline to reach a reasonable agreement among the human annotators.

On further discussions among annotators, we realized that one annotator only focused on the *precision* of the output overlap summaries, whereas the other annotator took both *precision* and *recall* into consideration. Therefore, subsequently, we decided to assign two separate scores for precision and recall.

Precision-Recall Inspired Double Scoring: This time, three label annotators (L_1, L_2 and L_3) assigned two numeric scores between 1-10 (inclusive) for the same set of 25 system-generated summaries. These numbers represented their belief about how precise the system-generated summaries were (the precision score) and how much of the actual ground-truth overlap information was covered by the same (the recall score). Also, note that *labels were assigned exclusively with respect to the input narratives only*. As the assigned numbers represent ordinal values (i.e. can't be directly used to com-

[3]The higher p-value means that the correlation value is insignificant because of the small number of samples.

Human agreement in terms of Kendall's Tau for Double Scoring				
	Precision		**Recall**	
	L_1	L_2	L_1	L_2
L_2	0.52	—	0.37	—
L_3	0.18	0.29	0.31	0.54
Average	**0.33**		**0.41**	

Table 7: Kendall's rank correlation coefficients among the precision and recall scores for pairs of human annotators (25 samples). L_i refers to the ith label annotator.

pute the F_1 score), we computed Kendall's rank correlation coefficient among the precision scores and recall scores separately for all the annotator pairs. The corresponding correlation values can be seen in table 7. As we notice, there is definitely some improvement in agreement among annotators compared to the one-number annotation in section 7.1. However, the average correlation is still 0.33 and 0.41 for precision and recall, respectively, much lower than 0.5 (the random baseline).

Human agreement in terms of Kendall's Tau Sentence-wise Scoring				
	Precision		**Recall**	
	L_1	L_2	L_1	L_2
L_2	0.68	—	0.75	—
L_3	0.59	0.64	0.69	0.71
Average	**0.64**		**0.72**	

Table 8: Average precision and recall Kendall rank correlation coefficients between sentence-wise annotation for different annotators. L_i refers to the ith label annotator. All values are statistically significant (p<0.05).

7.2 Sentence-wise Scoring

From the previous trials, we realized the downsides of assigning one/two numeric scores to judge an entire system-generated overlap summary. Therefore, as a next step, we decided to assign *overlap labels* (defined below) to each sentence within the system-generated overlap summary and use those labels to compute the overall precision and recall.

Overlap Labels: Label annotators (L_1, L_2 and L_3) were asked to look at each machine-generated sentence separately and determine if the core information conveyed by it is absent (A), partially present (PP) or present (P) in any of the four reference summaries (provided by I_1, I_2, I_3 and I_4) and respectively, assign the label *A*, *PP* or *P*. More precisely, annotators were provided with the following instructions: if the human feels that there is more than 75% overlap (between each system-generated sentence and any reference-summary sentence), assign label *P*, else if the human feels there is less than 25% overlap, assign label *A*, otherwise, assign label *PP*. This sentence-wise labelling was done for 50 different samples (with 506 sentences in total for system and reference summary), which resulted in a total of $3 \times 506 = 1,518$ sentence-level ground-truth labels.

To create the overlap labels (*A*, *PP* or *P*) for precision, we concatenated all 4 reference summaries to make one big reference summary and asked label-annotators (L_1, L_2, and L_3) to use it as a single reference for assigning the overlap labels to each sentence within machine generated summary. We argue that if the system could generate a sentence conveying information that is present in any of the references, it should be considered a hit. For recall, label-annotators were asked to assign labels to each sentence in each of the 4 reference summaries separately (provided by (I_1, I_2, I_3 and I_4)), with respect to the machine summary.

Inter-Rater-Agreement: After annotating each system-generated sentence (for precision) and reference sentence (for recall) with the labels (*A*, *PP* or *P*), we used the Kendall rank correlation coefficient to compute the pairwise annotator agreements among these ordinal labels. Table 8 shows that the correlations for both precision and recall are ≥ 0.50, signifying higher inter-annotator agreement.

Label from Annotator B		P	PP	A
Label from Annotator A	P	1	0.5	0
	PP	0.5	1	0
	A	0	0	1

Table 9: Reward matrix used to compare the labels assigned by two label annotators for a given sentence and helps to compute the agreement between the annotator pairs.

Reward-based Inter-Rater-Agreement: Alternatively, we defined a reward matrix (Table 9) which is used to compare the label of one annotator (say annotator A) against the label of another annotator (say annotator B) for a given sentence. This reward matrix acts as a form of correlation between two

Human agreement in terms of Reward function				
	Precision		**Recall**	
	L_1	L_2	L_1	L_2
L_2	0.81 ± 0.26	—	0.85 ± 0.11	—
L_3	0.79 ± 0.26	0.70 ± 0.31	0.80 ± 0.16	0.77 ± 0.17
Average	**0.77**		**0.81**	

Table 10: Average precision and recall reward scores (mean $\pm$ std) between sentence-wise annotation for different annotators. L_i refers to the ith label-annotator.

annotators. Once the reward has been computed for each sentence, one can compute the average precision and recall rewards for a given sample and accordingly, for the entire test dataset. The corresponding reward scores can be seen in Table 10. Both precision and recall reward scores are high (≥ 0.70) for all the different annotator pairs, thus signifying, a high inter-label-annotator agreement.

We believe, one of the reasons for higher reward/Kendall scores could be that sentence-wise labelling puts a lesser cognitive load on the human mind allowing them to be more consistent in contrast to the single or double score(s) for the entire overlap summary and, therefore, shows high agreement in terms of human interpretation. A similar observation was noted in Harman and Over (2004).

8 Limitations and Future Work

One particular limitation of this work is that we have used pre-trained abstractive summarization models as *naive baselines* / proxies for semantic overlap summarizer and did not attempt to develop a custom method that optimizes for the *overlap* constraint. However, the primary focus of this paper is to define the *SOS* task, as well as establish the first benchmark dataset and a suitable evaluation approach for the task. Therefore, the design and optimization of methods is an orthogonal task to this paper, which we will pursue as our immediate future work.

Another limitation of our work is that the test set is not big (~ 150 examples), which makes it difficult to do a rigorous evaluation. However, while the number 150 may initially appear to be small; cleaning and annotating the dataset required significant time and resources. To elaborate further, our test dataset contains 137 samples, where each sample consists of two alternative narratives along with 4 ground truth references. Out of these 4 references, 3 of them were manually written by 3 human annotators. Thus, we manually created $3 * 137 = 411$ reference summaries in total. Additionally, for each sample (narrative pair), each annotator first had to carefully read through two alternative narratives several times, digest the semantic overlap between them and then summarize the overlap in their own words. This process took on average 40 minutes per annotator per sample, which means we spent around $411 * 40 = 16,440$ minutes of human efforts in one round of the annotation process. Next, we had to resolve conflicts among annotators by going through each of their annotated summaries (a very painstaking process) and figuring out the reasons for the conflicts. Based on follow-up discussions, we revised the guidelines for annotation and went through the entire annotation process again. In total, we needed 4 iterations ($16,440 * 4 =\sim 65,760$ minutes) to resolve most of the conflicts. The whole process took more than 8 months for us. Finally, we agree that having more samples in the test dataset would definitely help. But this is both time and money-consuming. We are working towards it and would like to increase the sample size in the future.

9 Conclusion

In this work, we introduced a new NLP task, called Semantic Overlap Summarization (*SOS*) and created a benchmark dataset through meticulous human efforts to initiate a new research direction. As a starting point, we framed the problem as a constrained summarization task and showed that *ROUGE* is not a reliable evaluation metric for this task. Further human experiments show that sentence-wise evaluation leads to higher agreement with human judgment, therefore, an evaluation metric that aggregates sentence-wise overlap labels should be used while evaluating the SOS task.

10 Acknowledgements

We thank Auburn University College of Engineering and the Department of CSSE for their continuous support through Student Fellowships and Faculty Startup Grants.

References

Mousumi Akter, Naman Bansal, and Shubhra Kanti Karmaker. 2022. Revisiting automatic evaluation of extractive summarization task: Can we do better than rouge? In *Findings of the Association for Computational Linguistics: ACL 2022*, pages 1547–1560.

Otmane Azeroual and Horst Theel. 2018. The effects of using business intelligence systems on an excellence management and decision-making process by start-up companies: A case study. *International Journal of Management Science and Business Administration*, 4(3):30–40.

Sanghwan Bae, Taeuk Kim, Jihoon Kim, and Sanggoo Lee. 2019. Summary level training of sentence rewriting for abstractive summarization. *arXiv preprint arXiv:1909.08752*.

Regina Barzilay, Kathleen McKeown, and Michael Elhadad. 1999. Information fusion in the context of multi-document summarization. In *Proceedings of the 37th annual meeting of the Association for Computational Linguistics*, pages 550–557.

Biddut Sarker Bijoy, Syeda Jannatus Saba, Souvika Sarkar, Md Saiful Islam, Sheikh Rabiul Islam, Mohammad Ruhul Amin, and Shubhra Kanti Karmaker Santu. 2021. Covid19α: Interactive spatio-temporal visualization of covid-19 symptoms through tweet analysis. In *26th International Conference on Intelligent User Interfaces-Companion*, pages 28–30.

Ziqiang Cao, Wenjie Li, Sujian Li, and Furu Wei. 2018. Retrieve, rerank and rewrite: Soft template based neural summarization. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 152–161, Melbourne, Australia. Association for Computational Linguistics.

Rémi Cardon and Natalia Grabar. 2019. Parallel sentence retrieval from comparable corpora for biomedical text simplification. In *Proceedings of the International Conference on Recent Advances in Natural Language Processing (RANLP 2019)*, pages 168–177.

Yen-Chun Chen and Mohit Bansal. 2018. Fast abstractive summarization with reinforce-selected sentence rewriting. *arXiv preprint arXiv:1805.11080*.

Sumit Chopra, Michael Auli, and Alexander M Rush. 2016. Abstractive sentence summarization with attentive recurrent neural networks. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 93–98.

Arman Cohan and Nazli Goharian. 2016. Revisiting summarization evaluation for scientific articles. In *Proceedings of the Tenth International Conference on Language Resources and Evaluation LREC 2016, Portorož, Slovenia, May 23-28, 2016*. European Language Resources Association (ELRA).

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, Minneapolis, Minnesota. Association for Computational Linguistics.

Angela Fan, David Grangier, and Michael Auli. 2017. Controllable abstractive summarization. *arXiv preprint arXiv:1711.05217*.

Thibault Fevry and Jason Phang. 2018. Unsupervised sentence compression using denoising auto-encoders. *arXiv preprint arXiv:1809.02669*.

Kavita Ganesan. 2018. ROUGE 2.0: Updated and improved measures for evaluation of summarization tasks. *CoRR*, abs/1803.01937.

Jade Goldstein, Vibhu O Mittal, Jaime G Carbonell, and Mark Kantrowitz. 2000. Multi-document summarization by sentence extraction. In *NAACL-ANLP 2000 Workshop: Automatic Summarization*.

Yvette Graham. 2015. Re-evaluating automatic summarization with BLEU and 192 shades of ROUGE. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing, EMNLP 2015, Lisbon, Portugal, September 17-21, 2015*, pages 128–137. The Association for Computational Linguistics.

Hardy, Shashi Narayan, and Andreas Vlachos. 2019. Highres: Highlight-based reference-less evaluation of summarization. In *Proceedings of the 57th Conference of the Association for Computational Linguistics, ACL 2019, Florence, Italy, July 28- August 2, 2019, Volume 1: Long Papers*, pages 3381–3392. Association for Computational Linguistics.

Donna Harman and Paul Over. 2004. The effects of human variation in DUC summarization evaluation. In *Text Summarization Branches Out*, pages 10–17, Barcelona, Spain. Association for Computational Linguistics.

Naeemul Hassan, Amrit Poudel, Jason Hale, Claire Hubacek, Khandaker Tasnim Huq, Shubhra Kanti Karmaker Santu, and Syed Ishtiaque Ahmed. 2020. Towards automated sexual violence report tracking. In *Proceedings of the international AAAI conference on web and social media*, volume 14, pages 250–259.

Wan-Ting Hsu, Chieh-Kai Lin, Ming-Ying Lee, Kerui Min, Jing Tang, and Min Sun. 2018. A unified model for extractive and abstractive summarization using inconsistency loss. *arXiv preprint arXiv:1805.06266*.

Shubhra Kanti Karmaker Santu, Chase Geigle, Duncan Ferguson, William Cope, Mary Kalantzis, Duane Searsmith, and Chengxiang Zhai. 2018a. Sofsat: Towards a setlike operator based framework for semantic analysis of text. *ACM SIGKDD Explorations Newsletter*, 20(2):21–30.

Shubhra Kanti Karmaker Santu, Liangda Li, Yi Chang, and ChengXiang Zhai. 2018b. Jim: Joint influence modeling for collective search behavior. In *Proceedings of the 27th ACM International Conference on Information and Knowledge Management*, pages 637–646.

Yuta Kikuchi, Graham Neubig, Ryohei Sasano, Hiroya Takamura, and Manabu Okumura. 2016. Controlling output length in neural encoder-decoders. *arXiv preprint arXiv:1609.09552*.

Emiel Krahmer, Erwin Marsi, and Paul van Pelt. 2008. Query-based sentence fusion is better defined and leads to more preferred results than generic sentence fusion. In *Proceedings of ACL-08: HLT, Short Papers*, pages 193–196.

Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Ves Stoyanov, and Luke Zettlemoyer. 2019. Bart: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. *arXiv preprint arXiv:1910.13461*.

Chin-Yew Lin. 2004. Rouge: A package for automatic evaluation of summaries. In *Text summarization branches out*, pages 74–81.

Linqing Liu, Yao Lu, Min Yang, Qiang Qu, Jia Zhu, and Hongyan Li. 2017. Generative adversarial network for abstractive text summarization. *arXiv preprint arXiv:1711.09357*.

Yizhu Liu, Zhiyi Luo, and Kenny Zhu. 2018. Controlling length in abstractive summarization using a convolutional neural network. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 4110–4119.

Congbo Ma, Wei Emma Zhang, Mingyu Guo, Hu Wang, and Quan Z Sheng. 2020. Multi-document summarization via deep learning techniques: A survey. *arXiv preprint arXiv:2011.04843*.

Takuya Makino, Tomoya Iwakura, Hiroya Takamura, and Manabu Okumura. 2019. Global optimization under length constraint for neural text summarization. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 1039–1048.

Erwin Marsi and Emiel Krahmer. 2005. Explorations in sentence fusion. In *Proceedings of the Tenth European Workshop on Natural Language Generation (ENLG-05)*.

Yogesh Kumar Meena, Ashish Jain, and Dinesh Gopalani. 2014. Survey on graph and cluster based approaches in multi-document text summarization. In *International Conference on Recent Advances and Innovations in Engineering (ICRAIE-2014)*, pages 1–5. IEEE.

Vanessa Murdock and W Bruce Croft. 2005. A translation model for sentence retrieval. In *Proceedings of Human Language Technology Conference and Conference on Empirical Methods in Natural Language Processing*, pages 684–691.

Ramesh Nallapati, Bowen Zhou, Caglar Gulcehre, Bing Xiang, et al. 2016. Abstractive text summarization using sequence-to-sequence rnns and beyond. *arXiv preprint arXiv:1602.06023*.

Shashi Narayan, Shay B Cohen, and Mirella Lapata. 2018. Ranking sentences for extractive summarization with reinforcement learning. *arXiv preprint arXiv:1802.08636*.

Ani Nenkova. 2006. Summarization evaluation for text and speech: issues and approaches. In *INTERSPEECH 2006 - ICSLP, Ninth International Conference on Spoken Language Processing*. ISCA.

Jun-Ping Ng and Viktoria Abrecht. 2015. Better summarization evaluation with word embeddings for ROUGE. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing, EMNLP 2015, Lisbon, Portugal, September 17-21, 2015*, pages 1925–1930. The Association for Computational Linguistics.

Jian-Yun Nie, Michel Simard, Pierre Isabelle, and Richard Durand. 1999. Cross-language information retrieval based on parallel texts and automatic mining of parallel texts from the web. In *Proceedings of the 22nd annual international ACM SIGIR conference on Research and development in information retrieval*, pages 74–81.

Romain Paulus, Caiming Xiong, and Richard Socher. 2017. A deep reinforced model for abstractive summarization. *arXiv preprint arXiv:1705.04304*.

Alec Radford, Jeffrey Wu, Dario Amodei, Daniela Amodei, Jack Clark, Miles Brundage, and Ilya Sutskever. 2019. Better language models and their implications. *OpenAI Blog https://openai.com/blog/better-language-models*.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J Liu. 2019. Exploring the limits of transfer learning with a unified text-to-text transformer. *arXiv preprint arXiv:1910.10683*.

Michael Roth and Anette Frank. 2012. Aligning predicate argument structures in monolingual comparable texts: A new corpus for a new task. In ** SEM 2012: The First Joint Conference on Lexical and Computational Semantics–Volume 1: Proceedings of the main conference and the shared task, and Volume 2: Proceedings of the Sixth International Workshop on Semantic Evaluation (SemEval 2012)*, pages 218–227.

Alexander M Rush, Sumit Chopra, and Jason Weston. 2015. A neural attention model for abstractive sentence summarization. *arXiv preprint arXiv:1509.00685*.

Raphael Schumann. 2018. Unsupervised abstractive sentence summarization using length controlled variational autoencoder. *arXiv preprint arXiv:1809.05233*.

Elaheh ShafieiBavani, Mohammad Ebrahimi, Raymond K. Wong, and Fang Chen. 2018. A graph-theoretic summary evaluation for rouge. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing, Brussels, Belgium, October 31 - November 4, 2018*, pages 762–767. Association for Computational Linguistics.

Tomohide Shibata and Sadao Kurohashi. 2012. Predicate-argument structure-based textual entailment recognition system exploiting wide-coverage lexical knowledge. *ACM Transactions on Asian Language Information Processing (TALIP)*, 11(4):1–23.

Swapna Somasundaran, Michael Flor, Martin Chodorow, Hillary Molloy, Binod Gyawali, and Laura McCulla. 2018. Towards evaluating narrative quality in student writing. *Transactions of the Association for Computational Linguistics*, 6:91–106.

Kapil Thadani and Kathleen McKeown. 2011. Towards strict sentence intersection: decoding and evaluation strategies. In *Proceedings of the Workshop on Monolingual Text-To-Text Generation*, pages 43–53.

Rui Wang and Yi Zhang. 2009. Recognizing textual relatedness with predicate-argument structures. In *Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing*, pages 784–792.

Shomir Wilson, Florian Schaub, Frederick Liu, Kanthashree Mysore Sathyendra, Daniel Smullen, Sebastian Zimmeck, Rohan Ramanath, Peter Story, Fei Liu, Norman Sadeh, et al. 2018. Analyzing privacy policies at scale: From crowdsourcing to automated annotations. *ACM Transactions on the Web (TWEB)*, 13(1):1–29.

Travis Wolfe, Benjamin Van Durme, Mark Dredze, Nicholas Andrews, Charley Beller, Chris Callison-Burch, Jay DeYoung, Justin Snyder, Jonathan Weese, Tan Xu, et al. 2013. Parma: A predicate argument aligner. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 63–68.

Yuxiang Wu and Baotian Hu. 2018. Learning to extract coherent summary via deep reinforcement learning. *arXiv preprint arXiv:1804.07036*.

Dongling Xiao, Han Zhang, Yukun Li, Yu Sun, Hao Tian, Hua Wu, and Haifeng Wang. 2020. Ernie-gen: An enhanced multi-flow pre-training and fine-tuning framework for natural language generation. *arXiv preprint arXiv:2001.11314*.

Lexing Xie, Hari Sundaram, and Murray Campbell. 2008. Event mining in multimedia streams. *Proceedings of the IEEE*, 96(4):623–647.

Yu Yan, Weizhen Qi, Yeyun Gong, Dayiheng Liu, Nan Duan, Jiusheng Chen, Ruofei Zhang, and Ming Zhou. 2020. Prophetnet: Predicting future n-gram for sequence-to-sequence pre-training. *arXiv preprint arXiv:2001.04063*.

An Yang, Kai Liu, Jing Liu, Yajuan Lyu, and Sujian Li. 2018. Adaptations of ROUGE and BLEU to better evaluate machine reading comprehension task. In *Proceedings of the Workshop on Machine Reading for Question Answering@ACL 2018, Melbourne, Australia, July 19, 2018*, pages 98–104. Association for Computational Linguistics.

Michihiro Yasunaga, Rui Zhang, Kshitijh Meelu, Ayush Pareek, Krishnan Srinivasan, and Dragomir Radev. 2017. Graph-based neural multi-document summarization. *arXiv preprint arXiv:1706.06681*.

Jingqing Zhang, Yao Zhao, Mohammad Saleh, and Peter J Liu. 2019. Pegasus: Pre-training with extracted gap-sentences for abstractive summarization. *arXiv preprint arXiv:1912.08777*.

Jinming Zhao, Ming Liu, Longxiang Gao, Yuan Jin, Lan Du, He Zhao, He Zhang, and Gholamreza Haffari. 2020. Summpip: Unsupervised multi-document summarization with sentence graph compression. In *Proceedings of the 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 1949–1952.

Ming Zhong, Pengfei Liu, Yiran Chen, Danqing Wang, Xipeng Qiu, and Xuanjing Huang. 2020. Extractive summarization as text matching. *arXiv preprint arXiv:2004.08795*.

Ming Zhong, Pengfei Liu, Danqing Wang, Xipeng Qiu, and Xuanjing Huang. 2019. Searching for effective neural extractive summarization: What works and what's next. *arXiv preprint arXiv:1907.03491*.

Liang Zhou, Chin-Yew Lin, Dragos Stefan Munteanu, and Eduard H. Hovy. 2006. Paraeval: Using paraphrases to evaluate summaries automatically. In *Human Language Technology Conference of the North American Chapter of the Association of Computational Linguistics*. The Association for Computational Linguistics.

Qingyu Zhou, Nan Yang, Furu Wei, and Ming Zhou. 2017. Selective encoding for abstractive sentence summarization. *arXiv preprint arXiv:1704.07073*.

Analyzing the Dialect Diversity in Multi-document Summaries

Olubusayo Olabisi, Aaron Hudson, Antonie Jetter, Ameeta Agrawal
Portland State University
{oolabisi,ahuds2,ajetter,ameeta}@pdx.edu

Abstract

Social media posts provide a compelling, yet challenging source of data of diverse perspectives from many socially salient groups. Automatic text summarization algorithms make this data accessible at scale by compressing large collections of documents into short summaries that preserve salient information from the source text. In this work, we take a complementary approach to analyzing and improving the quality of summaries generated from social media data in terms of their ability to represent salient as well as diverse perspectives. We introduce a novel dataset, *DivSumm*, of dialect diverse tweets and human-written extractive and abstractive summaries[1]. Then, we study the extent of dialect diversity reflected in human-written reference summaries as well as system-generated summaries. The results of our extensive experiments suggest that humans annotate fairly well-balanced dialect diverse summaries, and that cluster-based pre-processing approaches seem beneficial in improving the overall quality of the system-generated summaries without loss in diversity.

1 Introduction

Since the launch of Twitter, its short, informal, creative, albeit noisy, social media posts called tweets, have been collected, labeled, and studied in numerous natural language processing (NLP) tasks, ranging from identifying topics and places of rising interest to sentiment analysis, and more. These easily accessible user-generated tweets, produced contemporaneously as world and private events unfold, provide insights into the perspective of diverse social groups but are too manifold for humans to interpret at scale. In response, automatic text summarization algorithms aim to compress long pieces of text into short, fluent, and consistent summaries while preserving the most salient information from the source text (Meng et al., 2012; Lin et al., 2021a; Amplayo et al., 2021).

Like other NLP models, summarization algorithms run the risk of perpetuating unintentional social biases against diverse groups (e.g., race or gender) and promoting structures and practices that systematically limit some groups' access to resources and decision-making power (Blodgett et al., 2020). This is because the collections of online texts such as news or Wikipedia articles, typically used for developing NLP algorithms, reflect the interests, language patterns, and structured writing style of their author demographics, which differ from those of other communities. Algorithms trained on such datasets may produce synopses in which diverse perspectives are systematically excluded. This means that groups who manage to overcome existing barriers to participation, for instance, via social media posts, and who speak up and offer their perspectives may still not be heard.

Fairness definitions for NLP models are generally based on the notion of equal treatment – an algorithm is considered fair when it performs the same for mainstream and underrepresented groups (i.e., group fairness) or delivers the same conclusions about an individual, regardless of the group they belong to (i.e., individual fairness) (Czarnowska et al., 2021). Recent works have demonstrated the disparate performance of tools on sensitive subpopulations in domains (Tatman, 2017; Buolamwini and Gebru, 2018). In this work, we consider the notion of group fairness and interpret it in terms of representation distribution of some socially salient attribute (dialect) in the summary.

Our goal is to investigate the ability of existing models of summarization to reflect the diversity of input text in the generated summaries, and propose a simple yet effective approach for improving the group-level diversity of summaries generated from noisy social media tweets, for which very little prior work exists to date (Dash et al., 2019;

[1] *DivSumm* dataset is available at https://github.com/PortNLP/DivSumm

Proceedings of the 29th International Conference on Computational Linguistics, pages 6208–6221
October 12–17, 2022.

Dataset	domain	attribute (#groups)	#topics	#summ/topic	#docs/input	#sent/summ	Ext?	Abs?
MeToo (Dash et al., 2019)	tweets	gender (2)	1	3	488	-	Y	N
Claritin (Dash et al., 2019)	tweets	gender (2)	1	3	4037	100	Y	N
US-Election (Dash et al., 2019)	tweets	political leaning (3)	1	3	2120	-	Y	N
DivSumm (ours)	tweets	dialect (3)	25	2	90	5	Y	Y

Table 1: Statistics of some social multi-document extractive summarization datasets with socially salient user group attributes and human written summaries.

Keswani and Celis, 2021). To this end, we seek to answer two specific questions with respect to summarization of dialect diverse tweets: (Q1) how diverse are summaries generated by humans? and, (Q2) how diverse are summaries generated by automatic summarization models with and without the proposed fairness interventions?

Our work makes several contributions:

- we introduce and comprehensively analyze a novel dataset, *DivSumm*, of diverse dialect tweets across several topics and corresponding extractive and abstractive human-written reference summaries;
- we study the group diversity of reference summaries and investigate two simple yet effective approaches for applying diversity interventions at the pre-processing stage of the summarization process;
- we conduct an extensive set of experiments using six existing extractive as well as abstractive summarization models as black-boxes, and report the results in terms of three types of metrics (reference-less, reference-based, and representation).

2 Related Work

In this section we discuss two relevant areas of prior work: multi-document summarization for social media data, and an overview of existing summarization datasets.

Multi-document social media summarization

Summarizing of social media data remains an interesting area of research with numerous approaches focused on optimizing the textual quality, factuality, fluency, and many other properties of the summaries (Li and Zhang, 2020; He et al., 2020; Dusart et al., 2021). However, unlike other text input (e.g., news articles), data from social media are also incredibly diverse consisting of opinions and perspectives from people from many walks of life, and while reflecting this richness of diversity in the summaries generated from them is an important goal, there have been few notable efforts in this direction.

One of the early works to study the notion of fairness in summaries generated from social media data used extractive summarization models and noted that the generated summary is not always a fair representation of the input data, even though the tweets written by different social groups (gender and political leaning in this case) are of comparable quality (Shandilya et al., 2018; Dash et al., 2019). Following these assertions, they proposed three fairness-preserving algorithms that can be applied during the pre-processing, in-processing, and post-processing stages. Keswani and Celis (2021) investigated the role of extractive summarization models in the context of dialect diversity of summaries, but did so without access to manually annotated summaries and proposed a bias mitigation model at the post-processing stage.

Summarization datasets

Developing summarization datasets is a challenging task and as such, researchers often utilize creative methods for automatically obtaining summaries of documents (Nallapati et al., 2016; Xu et al., 2021; Perez-Beltrachini and Lapata, 2021; Varab and Schluter, 2021). In cases where automatic document/summary pairings cannot be obtained, human annotation, usually through crowdsourcing, is often used (Khalman et al., 2021; Lin et al., 2021b). Specifically in the domain of social media data, many multi-document datasets have been recently introduced, some with extractive summaries, others with abstractive summaries, including TSix (Nguyen et al., 2018), Amazon and Yelp (He and McAuley, 2016; Bražinskas et al., 2020), SPACE (Angelidis et al., 2021), and ISSumSet (Dusart et al., 2021), to name just a few, but as none of these contain explicit markers of diverse social groups, we are motivated to develop a novel dataset of dialect diverse summaries.

3 Problem Description

For the task of multi-document summarization of social media data, the input typically consists of dozens to hundreds of documents (e.g., tweets) about the same topic that are all considered in generating a single summary. Formally, the input is a set of documents further split into n sentences, $\mathcal{D} = \{d_1, ..., d_n\}$. Given the sentences in $\mathcal{D}$, the goal of a multi-document summarization model is to generate a summary $\mathcal{S} = \{s_1, \ldots, s_k\}$ consisting of k sentences, where $k << n$ is usually a hyper-parameter. There are two types of summarization models, extractive models where $\mathcal{S} \subset \mathcal{D}$, and abstractive models which extract and rewrite salient pieces of text.

In multi-user settings where users belong to multiple social groups, each sentence d_i may be additionally accompanied by social attribute g_j, $g_j \in \mathcal{G}$, where $\mathcal{G}$ is a set of attributes of a demographic group such as gender, race, or dialect. To adequately accommodate the variety of perspectives expressed by multiple diverse groups, the goal of a *diversity-preserving* multi-document summarization algorithm then is to generate a summary $\mathcal{S}$ with the goal of not only optimizing textual quality but also satisfying some fairness constraint such as group fairness, which compare quantities at group level (Czarnowska et al., 2021). In the context of summarization, this problem formulation naturally lends itself well to extractive models where sentences from the final summaries can be traced back to source documents and their user group labels more so than to abstractive models where text usually gets rewritten making group label attribution challenging to ascertain.

One simple approach of computing group representation distribution in extractive summaries, which we denote as $\mathcal{R}(\mathcal{S})$, is by calculating the number of tweets belonging to each user group in the summary. Assuming m distinct and disjoint groups, $\mathcal{R}(\mathcal{S}) = \{\frac{|\mathcal{S}_1|}{k}, \ldots, \frac{|\mathcal{S}_m|}{k}\}$ where $|\mathcal{S}_j|$ is the number of sentences within the summary $\mathcal{S}$ from group j, and k is the total number of sentences in the summary.

Under the notion of equal representation, where the representation of all groups should be equal, regardless of their distribution in the input data, one can compute an aggregate diversity score. For instance, we can compute the **Representation Gap**, where a lower gap score would imply a more balanced distribution. Several metrics can be used to estimate the level of dispersion[2], and we choose the range, $\mathcal{R}_{rg}(\mathcal{S}) = \max\{\mathcal{R}(\mathcal{S})\} - \min\{\mathcal{R}(\mathcal{S})\}$. In other words, for well-balanced group distributions, a smaller $\mathcal{R}_{rg}(\mathcal{S})$ score indicates a more diverse summary. As an example, if a summary of 5 lines contains 3 sentences from group A, and a sentence each from group B and group C, then $\mathcal{R}(\mathcal{S}) = \{0.6, 0.2, 0.2\}$, and $\mathcal{R}_{rg}(\mathcal{S}) = 0.4$.

4 Dialect Diverse Summarization Dataset (*DivSumm*)

In order to study the diversity-preserving capabilities of summarization algorithms, we need a suitable dataset, and although numerous summarization datasets have become available in recent years, only a handful of them contain explicit diverse social group information (see Table 1), which motivates us to develop and contribute a novel dataset – *DivSumm*. Our dataset consists of input-summary pairs on a set of 25 topics of tweets written in three different dialects. We obtain their corresponding human-generated extractive and abstractive summaries. Table 2 presents an instance from *DivSumm*, while the following subsections outline our process of creating and exploring this dataset.

4.1 Obtaining human-written reference summaries

To obtain a large number of tweets of diverse dialectal language for annotating and creating *DivSumm*, we turn to a corpus of English tweets automatically labeled with dialect information by inferring three demographic dialect proportions, namely, *African-American (AA)* English, *Hispanic* English, and *White* English, using a model trained with census data (Blodgett et al., 2016). All the preprocessing details are included in Appendix A.

Our annotation study was designed to obtain both extractive as well as abstractive summaries, with the input consisting of multiple documents (randomly selected and shuffled 90 tweets on a given topic, with 30 tweets per dialect group) to generate topic-wise summaries (e.g., summary for `NBA`, `Netflix`, `Beyonce`, and so on). In other words, we feature all dialects with equal proportion, an approach typical of datasets that are meant to be evaluation benchmarks (Fabris et al., 2022). For generating the summaries, we invited a diverse group of ten volunteers familiar with tweets and

[2]We also computed the standard deviation of $\mathcal{R}(\mathcal{S})$ which showed a high correlation with range.

Input Documents (Tweets)
G3: If Lakers play like that every game no chance for anyone else in the NBA #NoExceptions G2: The Fan Fictions and Imagines other Beliebers come up with , OMFG I LOVE THIS FANBASE ! G2: Remember when they the NBA play-offs were boring and disappointing-not so much anymore #nbafantastic G1: NBATV shittin on u niggas.lol. Melo and Monroe doing work doe. G3: Going to take in some NBA tonight. Pacers/Knicks. NBA games are much better in person. G2: Just fast forward to the trophy presentation and the sucking off of LeBron by your company. NBA is garbage G1: No point guard in the NBA can hold Russell Westbrook G1: The Black Mamba 81 point showcase in showing now on NBATV !! G2: NBA reffs.. Responsible for half of Miami heats points since 2010. -.-t #refsforMVP G3: And te NBA was in a lockout last year too.....NOBODY complained about that last year. Bc frankly, nobody cares. ... ... G3: Is it possible that boozer is the worst player in the NBA? #GETOUT G3: NBA rule change I'm shocked has never happened: An assist if the player you passed to makes both free throws. G3: If this was an NBA game, Michael Carrera would be walking away from it with a $100,000 fine. #flopcity G1: Lakers??? Is that even a NBA team? I thought they were D-League.....

Extractive Summaries

Annotator 1	Annotator 2
Going to take in some NBA tonight. Pacers/Knicks. NBA games are much better in person. There's a difference btw NBA ready talent & NBA prospects. UK has a ton of prospects this yr. Seeing the difference this yr. And the #Lakers get back in the game by slowing it down, grinding on 'D', and going at the basket. Who knew? 0_0 #NBA. ***It's pathetic that everyone thinks that the NBA is better than the NHL...*** *Free NBA League Pass Preview till tomorrow, Dam I'm sure I'm staying in this Saturday night!*	*Remember when they the NBA play-offs were boring and disappointing-not so much anymore #nbafantastic. Lebron is still the best player in the NBA. Put your mouth on it.". Bucks vs Sixers. This is why I love NBA TV. #FanNight. Looks like I'm watching the sugar bowl game cz its halftime for the NBA.* ***It's pathetic that everyone thinks that the NBA is better than the NHL...***

Abstractive Summaries

Annotator 1	Annotator 2
It's an NBA game night. Many people are tuned in because it's the first game of the season. There are different reactions to the game because some think it's awesome and some think it's whack. It would greatly have to do with fans(the team they support). Viewers also gave opinions of different players they consider to be the best and Lebron is thought to be overhyped.	*All Tweets seem to contain 'NBA' regardless of placement or context, often even including tweets where NBA is part of a word such as "FANBASE". Overall the majority of tweets seem to be in-regards to the National Basketball Association, NBA, and include more often critiques or 'insults' of competing teams. It is also evident that the 'NBA', as a brand, include different ancillary businesses such as 'NBA League Pass', NBATV, and NBA2K13 (video game) in conjunction with the obvious basketball games themselves. A portion of the sample includes a notable use of racial epitaphs or slurs that may or may not be used in derision. The overwhelming bulk of the tweets is of a negative (critiquing, admonishing) nature as opposed to a positive (hopeful, cheering) message.*

Table 2: Example instance from our *DivSumm* dataset showing input documents with corresponding reference extractive and abstractive summaries generated by two annotators. For the extractive summaries, overlapping text between the annotators is denoted in red text and bold font.

their idiosyncrasies, and more importantly, spanning a range of diversity across dialects, gender, and ethnicity. *Note that we did not mention anything about the dialects of the tweets before sharing the files for annotation, so the annotators had no background information about the goals of this study other than the fact that we seek to summarize the tweets* – this was done to mitigate any form of potential bias. We provided concise and clear guidelines for generating the summaries, along with an example set of tweets and corresponding extractive and abstractive summaries. For the extractive summaries, the annotators were requested to select 5 tweets that they believe to capture the salient points from the set of documents, while for the abstractive summaries, the annotators were asked to write 5 sentences in their own words to summarize the important points across all the documents. Every set of input tweets was summarized by two annotators, thus helping us develop a dataset of 25 topics, with a total of 100 pairs of input (sets of documents) and output (human-written summaries).

It is worth mentioning that the diversity of the annotator pool will undoubtedly have an impact on the ultimate annotations (summaries) obtained. The inherent subjective nature of summarization process suggests that different annotators will approach it from different perspectives, which is a strength but also a weakness of this process and

	R1	RL
S1, random	0.234	0.210
S2, random	0.220	0.193
S1, S2	**0.315**	**0.301**

Table 3: ROUGE scores comparing human-human summaries (S1 and S2) and human-random summaries.

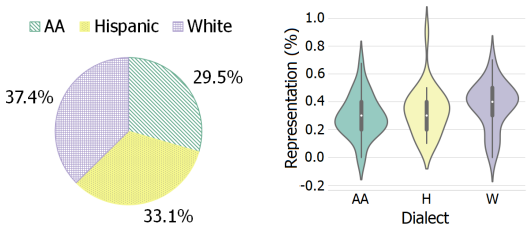

(a) (b)

Figure 1: *Diversity Analysis of Human-written Reference Extractive Summaries in* DivSumm. The plot (a) displays the average representation $\mathcal{R}(\mathcal{S})$ of each dialect with each dialect's $\mathcal{R}(\mathcal{S})$ in the range of 29.5 - 37.4%, indicating a fairly balanced representation. In plot (b), we present the violin plots of the distribution scores noting few outliers.

has been extensively studied in recent literature (Clark et al., 2021; Gehrmann et al., 2022). To help account for these limitations, during evaluation we report the results using not only metrics that rely on these reference summaries, but also those that do not require reference summaries.

As summarizing informal user-generated data is a particularly challenging task, we also compute the inter-annotator agreement to measure the lexical overlap of extractive reference summaries written by both annotators. As shown in Table 3, the average ROUGE-1 and ROUGE-L F1 scores for pairs of human annotated summaries was much higher (∼0.31) than that of a randomly generated summary compared against the human-generated summaries (∼0.21), suggesting that the human-written summaries are more similar, and arguably reliable.

4.2 Analyzing diversity in human-written summaries

Before studying how well automatic summarization models reflect diversity in system summaries, a natural question to ask is how well do humans summarize such diverse data. To answer this question, we conduct a thorough analysis of the reference summaries of *DivSumm* dataset in an attempt to uncover any interesting insights into the way that humans (specifically, our annotators) approach dialect-diverse summarization process. Importantly, our answers to these analyses will serve as a principled baseline when we later evaluate the model-generated summaries.

The extractive summaries provide a uniquely interesting opportunity to explore the question of how diverse human-written summaries are by allowing us to compare the proportion of representation of each dialect group within the summaries. Recall that our dataset features an equal number of tweets for each of the three dialect groups in the input, and, therefore, for equal or proportional representation in the output summary, a well-balanced summary would contain equal number of tweets from each group. As the plots in Figure 1 show, we found a fairly balanced representation for each dialect group in the summary ($\mathcal{R}(\mathcal{S})$ ranges between 29.5% to 37.4%).

5 Modeling Diversity in System Summaries

In this section, we explore the diversity-preserving qualities of recently proposed summarization algorithms using three approaches described below and visualized in Figure 2.

VANILLA: This standard baseline approach of summarizing uses a single aggregated set of randomized documents from all the dialect groups, e.g., a total of 90 tweets, as the input to the summarization model without any pre-processing.

CLUSTER-HEURISTIC (CLUSTER-H) : Similar to the pre-processing approach of Dash et al. (2019), this method first heuristically partitions the set of input documents into group-based subsets ($\mathcal{D} = \{\mathcal{D}_1 \cup ... \cup \mathcal{D}_m\}$, each $\mathcal{D}_j$ containing a set of documents from group j) before passing them to the summarization model to generate separate group summaries ($\mathcal{S} = \{\mathcal{S}_1, ..., \mathcal{S}_m\}$) – one for each of the three group-specific tweets. However, instead of concatenating these summaries to generate the final summary, we shuffle and combine these m group-level summaries into a single document and pass that again to the summarization model to generate a new, combined summary. In doing so, we seek to first preserve group-level salient information before aggregating the most informative units from such individual summaries into a unified summary.

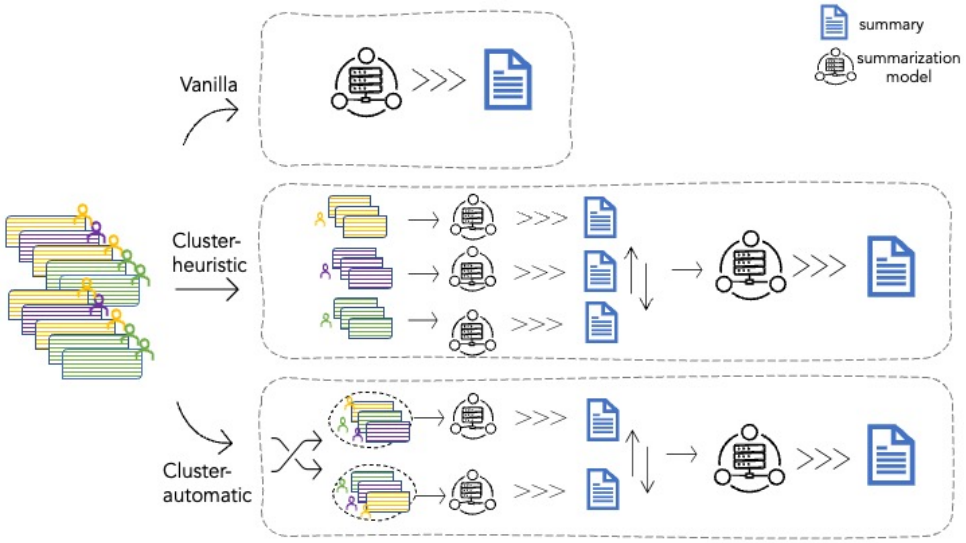

Figure 2: Illustration showing the overview of VANILLA, CLUSTER-HEURISTIC and CLUSTER-AUTOMATIC.

CLUSTER-AUTOMATIC (CLUSTER-A): Conversely, in a more pragmatic scenario, the sensitive group attribute may not be reliably observable, or inferring it is not possible due to some reason, including ethical reason. In such a case, we also investigate an attribute-agnostic approach based on automatic clustering, as follows: (i) Generate p clusters of $\mathcal{D}$ via some clustering algorithm (e.g., k-means) with the optimal value of p determined through silhouette score, yielding $\mathcal{D} = \{\mathcal{D}_1 \cup ... \cup \mathcal{D}_p\}$. (ii) Generate a set of corresponding summaries $\mathcal{S} = \{\mathcal{S}_1, ..., \mathcal{S}_p\}$. (iii) Concatenate all the p summaries into a single document and pass it again to the summarization algorithm to generate a final summary.

6 Experiments

The three approaches are studied in the context of recently proposed summarization models, both extractive and abstractive, and the quality of the summaries is evaluated using reference-based metrics, reference-less metrics, and representation metric.

6.1 Summarization models

We explore six recent extractive and abstractive summarization models described below (we refer to Appendix B for full implementation details).

Extractive methods: **TEXTRANK** (Mihalcea and Tarau, 2004), an unsupervised graph-based ranking method, determines the most important sentences in a document based on information extracted from the document itself, and therefore performs well even without any form of domain knowledge or pre-training. **BERT-EXT** (Miller, 2019), an extractive summarization model, uses pretrained embeddings from BERT (Devlin et al., 2018) and k-means clustering to select sentences closest to the centroid as the summaries, and similarly, **LONGFORMER-EXT** (Miller, 2019) which uses pretrained embeddings from LongFormer (Beltagy et al., 2020).

Abstractive methods: **BART** (Lewis et al., 2019) is a sequence-to-sequence model combining a bidirectional encoder with an auto-regressive decoder and trained by corrupting the document with an arbitrary noisy function. **T5** (Raffel et al., 2019), an encoder-decoder model trained using teacher forcing, modifies the original transformer architecture to convert language problems into a text-to-text format. **LED** (Longformer Encoder-Decoder) (Beltagy et al., 2020), a variant of the Longformer model with both encoder and decoder transformer stacks, has shown to improve modeling long sequences for sequence-to-sequence learning.

6.2 Evaluation metrics

To evaluate the quality of the system-generated summaries, we consider three types of metrics.

Reference-based: **ROUGE** (Lin, 2004) calculates the lexical overlap between the machine-generated output and the human-written reference summaries. For our experiments, we report the F1 scores of ROUGE-1 (overlapping unigrams) and ROUGE-L (longest common subsequences). To compute the final scores, a system-generated summary is compared with each of the two reference summaries, and their average score is reproted.

Extractive Models

Model	TextRank					BERT-EXT					LONGFORMER-EXT				
	R1	RL	SQA	B	$\mathcal{R}_{rg}\downarrow$	R1	RL	SQA	B	$\mathcal{R}_{rg}\downarrow$	R1	RL	SQA	B	$\mathcal{R}_{rg}\downarrow$
VANILLA	0.25	0.23	0.06	0.11	0.15	0.22	0.21	0.04	0.13	0.04	0.22	0.20	0.04	0.12	0.17
CLUSTER-H	0.26	0.24	0.06	0.10	0.18	0.22	0.21	0.05	0.12	0.07	0.22	0.20	0.05	0.11	0.20
CLUSTER-A	0.25	0.23	0.07	0.10	0.16	0.22	0.20	0.05	0.12	0.06	0.23	0.21	0.05	0.11	0.16

Abstractive Models

Model	BART					T5					LED				
	R1	RL	SQA	B	$\mathcal{R}_{rg}\downarrow$	R1	RL	SQA	B	$\mathcal{R}_{rg}\downarrow$	R1	RL	SQA	B	$\mathcal{R}_{rg}\downarrow$
VANILLA	0.16	0.14	0.06	0.09	-	0.15	0.13	0.07	0.08	-	0.13	0.12	0.06	0.07	-
CLUSTER-H	0.16	0.15	0.06	0.09	-	0.14	0.13	0.05	0.07	-	0.14	0.13	0.05	0.08	-
CLUSTER-A	0.17	0.15	0.06	0.09	-	0.12	0.11	0.06	0.06	-	0.14	0.14	0.05	0.07	-

Table 4: Results of the three approaches (VANILLA, CLUSTER-H, and CLUSTER-A) across three extractive summarization models (TextRank, BERT-Ext, LongFormer-Ext) and three abstractive summarization models (BART, T5, LED) using the *DivSumm* summarization dataset. The metrics reported include ROUGE-1 (R1), ROUGE-L (RL), SummaQA (SQA), BLANC (B), and for the extractive summaries $\mathcal{R}_{rg}(\mathcal{S})$ denoting the Representation Gap. All scores are averaged over two runs. The best scores per model and per metric have been underlined. For reference, the Representation Gap in human summaries $\mathcal{R}_{rg}(\mathcal{S})$ = 0.08.

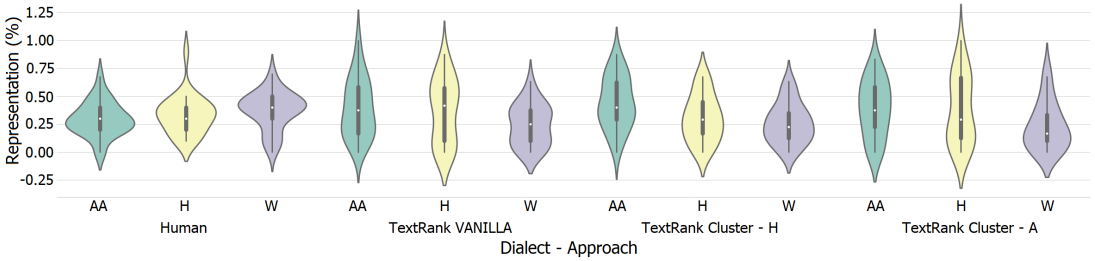

Figure 3: Violin plots of $\mathcal{R}(\mathcal{S})$ per dialect and per approach for TEXTRANK over *DivSumm* dataset. It is noticed that the violins for both *AA* and *Hispanic* tweets are considerably thinner as compared to the human summaries indicating many outliers on both ends of the spectrum.

Reference-less: **SummaQA** (Scialom et al., 2019) evaluates the quality of a text summary without relying on reference summaries, making it a practical choice for assessing summaries generated from large collections of tweets. Instead, it uses a question-answering model based on BERT to answer cloze-style questions using the system-generated summaries. **BLANC** (Vasilyev et al., 2020), another reference-less metric, measures how the performance of a pretrained language model improves during language understanding tasks when the model is given access to a summary, and correlates with informativeness (Iskender et al., 2021).

Representation Gap: Finally, we also report the $\mathcal{R}_{rg}(\mathcal{S})$ of the extractive summaries by calculating the range of the representation distribution in the summary.

7 Results and Discussion

Table 4 presents the detailed results of our experiments across extractive and abstractive models, and some samples of system-generated summaries are included in Appendix C. In looking at the representation gap scores of extractive models, we note that the VANILLA approach without any intervention does well in terms of $\mathcal{R}_{rg}$ on 2 out of 3 datasets, while in the case of CLUSTER-H performance is strictly worse despite that model being designed to consider group-level information. In terms of summary quality, all three approaches perform comparably. Of note, however, is the performance of BERT-Ext model which yields impressive representation gap scores, suggesting that centroid-based approaches in particular could be effective as unsupervised diversity-preserving models.

Model	*Extractive* AVERAGE					*Abstractive* AVERAGE			
	R1	RL	SQA	B	$\mathcal{R}_{rg}\downarrow$	R1	RL	SQA	B
VANILLA	0.232	0.217	0.053	**0.123**	**0.12**	0.147	0.136	**0.068**	**0.083**
CLUSTER-H	0.236	0.218	0.057	0.117	0.15	**0.151**	**0.140**	0.059	**0.083**
CLUSTER-A	**0.237**	**0.219**	**0.062**	0.115	0.13	0.150	0.139	0.061	0.080

Table 5: Averaged results of the three approaches (VANILLA, CLUSTER-H, and CLUSTER-A) across three extractive summarization models and three abstractive summarization models evaluated using the *DivSumm* summarization dataset. The metrics reported include ROUGE-1 (R1), ROUGE-L (RL), SummaQA (SQA), BLANC (B), and for the extractive summaries, $\mathcal{R}_{rg}(\mathcal{S})$.

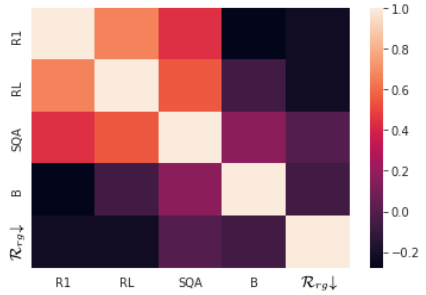

(a)

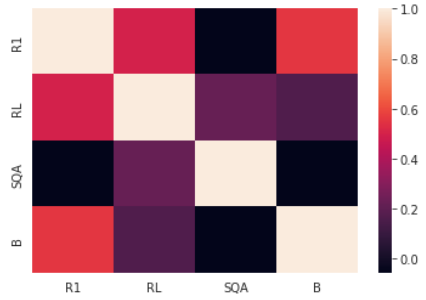

(b)

Figure 4: Pairwise Kendall's Tau correlations for summary evaluation metrics, (a) extractive models, and (b) abstractive models. A higher score (shown in lighter color) indicates higher correlation between the rankings provided by a pair of metrics, whereas a smaller score (shown in darker color) indicates weaker correlation.

Figure 3 presents the violin plots for one of the summarization models, TEXTRANK, to allow us to further investigate the differences between the three approaches (please see Appendix D for analysis of the other two extractive models). For all the approaches, the violins for both *AA* and *Hispanic* summaries are noticeably thinner than the corresponding human summaries, suggesting that the $\mathcal{R}(\mathcal{S})$ representation distribution of system summaries contains many outliers on either end of the spectrum.

Table 5 presents the averaged results of all the extractive methods and abstractive methods, which confirm that VANILLA generates more diverse summaries, while CLUSTER-A does generally better on the other metrics related to summary quality, hinting at a plausible trade off between the two dimensions (Celis et al., 2018). Considering the results of extractive and abstractive models together, it appears that some sort of clustering of documents before passing them to the summarization model remains beneficial in improving the overall quality of summaries.

Next, given the multiple evaluation metrics along dimensions of quality and representation, we further study the correlations between these metrics and present some heatmap visualizations generated using the pairwise Kendall's Tau correlation values. Figure 4 summarizes the correlations computed using the average results. Each approach was first sorted from best to worst based on the scores provided by each metric, and these rankings were then used to calculate Kendall's Tau to represent how well the rankings correlate between metrics. A score closer to 1.0 indicates high correlation (that is, the metrics ranked approaches in similar order), while a score closer to -1.0 indicates poor correlation (the metrics ranked approaches in a different order).

We observe that in the case of extractive summaries, metrics R1, RL, and SummaQA have high to moderate inter-correlation, while BLANC and $\mathcal{R}_{rg}(\mathcal{S})$ do not correlate with any other metrics. For the abstractive models, R1 shows moderate correlation with RL and BLANC, but SummaQA shows poor correlation with all the other three metrics. Overall, we conclude that (i) both the reference-based metrics R1 and RL correlate well as expected, (ii) R1 correlates well with reference-less metric SummaQA for extractive summaries, and with BLANC for abstractive summaries, (iii)

Model	BART				T5				LED			
	R1	RL	SQA	B	R1	RL	SQA	B	R1	RL	SQA	B
VANILLA	0.202	0.184	0.212	0.201	0.194	0.181	0.269	0.281	0.160	0.148	0.181	0.172
CLUSTER-A	0.172	0.157	0.229	0.270	0.170	0.158	0.185	0.224	0.176	0.161	0.217	0.271

Table 6: Results of VANILLA and CLUSTER-A as applied to another dataset, DialogSumm (Chen et al., 2021)

the two reference-less metrics (SummaQA and BLANC) do not correlate with each other, and (iv) finally, no metrics seem to be correlating well with the representation metric ($\mathcal{R}_{rg}$), suggesting the need for new metrics that can measure representation of diversity in addition to other dimensions of quality.

Finally, since clustering-based approaches improved over the baseline approach in terms of summary quality, we conduct one more investigation to evaluate whether CLUSTER-A generalizes to another dataset involving multiple users such as *DialogSumm* dataset (Chen et al., 2021). The results presented in Table 6 indicate that VANILLA performs better when using T5 model, CLUSTER-A brings additional gains to LED model, while remaining comparable on the third model.

8 Conclusions

In this work, we investigate whether, and to what extent, do system-generated summaries reflect the diversity of socially salient groups present in the input data. To answer this question, we first develop a novel summarization dataset, *DivSumm*, by obtaining human-written reference summaries, of both extractive and abstractive sort, for dialect diverse tweets. In analyzing the human-written reference summaries, we were encouraged to note that on average humans generated reasonably well-balanced dialect diverse summaries. This was followed by an extensive evaluation exploring the diversity reflected in system summaries by experimenting with three approaches as applied to six summarization algorithms, and evaluated using multiple metrics of summary quality and representation. Future avenues of work include expanding our dataset to consider other diverse social attributes and improving the summarization models along dimensions of both quality as well as representation.

9 Ethical Considerations

Tweets provide a rich and diverse source of natural language data but in working with unfiltered social media data, we also run the risk of encountering unconventional or in some cases what may be considered as offensive language. Being sensitive to these limitations, before undertaking the annotation process, we carefully informed the annotators of some of the inherent risks of annotating tweets and provided them with the option of withdrawing from the annotation process should they feel uncomfortable at any point of time (it is worth noting that no annotator withdrew from the study). Similarly, our discussions related to the representation of dialect diversity in summaries are based solely on the summaries that were developed during this study and the summarization models that were adopted in our experiments. It remains to be seen whether these conclusions generalize to other social groups.

Acknowledgments

We would like to thank Camille Range, Vikram Nagapudi, and our annotators for their help in creating this dataset. We are grateful to the anonymous reviewers for their insightful suggestions, and to Cisco Research for partially supporting this research.

References

Reinald Kim Amplayo, Stefanos Angelidis, and Mirella Lapata. 2021. Unsupervised opinion summarization with content planning. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 35, pages 12489–12497.

Stefanos Angelidis, Reinald Kim Amplayo, Yoshihiko Suhara, Xiaolan Wang, and Mirella Lapata. 2021. Extractive opinion summarization in quantized transformer spaces. *Transactions of the Association for Computational Linguistics*, 9:277–293.

Iz Beltagy, Matthew E Peters, and Arman Cohan. 2020. Longformer: The long-document transformer. *arXiv preprint arXiv:2004.05150*.

Su Lin Blodgett, Solon Barocas, Hal Daumé III, and Hanna Wallach. 2020. Language (technology) is power: A critical survey of" bias" in nlp. *arXiv preprint arXiv:2005.14050*.

Su Lin Blodgett, Lisa Green, and Brendan O'Connor. 2016. Demographic dialectal variation in social media: A case study of african-american english. *arXiv preprint arXiv:1608.08868*.

Arthur Bražinskas, Mirella Lapata, and Ivan Titov. 2020. Few-shot learning for opinion summarization. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 4119–4135, Online. Association for Computational Linguistics.

Joy Buolamwini and Timnit Gebru. 2018. Gender shades: Intersectional accuracy disparities in commercial gender classification. In *Conference on fairness, accountability and transparency*, pages 77–91. PMLR.

Elisa Celis, Vijay Keswani, Damian Straszak, Amit Deshpande, Tarun Kathuria, and Nisheeth Vishnoi. 2018. Fair and diverse dpp-based data summarization. In *International Conference on Machine Learning*, pages 716–725. PMLR.

Yulong Chen, Yang Liu, Liang Chen, and Yue Zhang. 2021. Dialogsum: A real-life scenario dialogue summarization dataset. *arXiv preprint arXiv:2105.06762*.

Elizabeth Clark, Tal August, Sofia Serrano, Nikita Haduong, Suchin Gururangan, and Noah A. Smith. 2021. All that's 'human' is not gold: Evaluating human evaluation of generated text. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 7282–7296, Online. Association for Computational Linguistics.

Paula Czarnowska, Yogarshi Vyas, and Kashif Shah. 2021. Quantifying social biases in nlp: A generalization and empirical comparison of extrinsic fairness metrics. *arXiv preprint arXiv:2106.14574*.

Abhisek Dash, Anurag Shandilya, Arindam Biswas, Kripabandhu Ghosh, Saptarshi Ghosh, and Abhijnan Chakraborty. 2019. Summarizing user-generated textual content: Motivation and methods for fairness in algorithmic summaries. *Proceedings of the ACM on Human-Computer Interaction*, 3(CSCW):1–28.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*.

Alexis Dusart, Karen Pinel-Sauvagnat, and Gilles Hubert. 2021. Issumset: a tweet summarization dataset hidden in a trec track. In *Proceedings of the 36th Annual ACM Symposium on Applied Computing*, pages 665–671.

Alexander R Fabbri, Wojciech Kryściński, Bryan McCann, Caiming Xiong, Richard Socher, and Dragomir Radev. 2021. Summeval: Re-evaluating summarization evaluation. *Transactions of the Association for Computational Linguistics*, 9:391–409.

Alessandro Fabris, Stefano Messina, Gianmaria Silvello, and Gian Antonio Susto. 2022. Algorithmic fairness datasets: the story so far. *arXiv preprint arXiv:2202.01711*.

Guy Feigenblat, Chulaka Gunasekara, Benjamin Sznajder, Sachindra Joshi, David Konopnicki, and Ranit Aharonov. 2021. TWEETSUMM - a dialog summarization dataset for customer service. In *Findings of the Association for Computational Linguistics: EMNLP 2021*, pages 245–260, Punta Cana, Dominican Republic. Association for Computational Linguistics.

Sebastian Gehrmann, Elizabeth Clark, and Thibault Sellam. 2022. Repairing the cracked foundation: A survey of obstacles in evaluation practices for generated text. *arXiv preprint arXiv:2202.06935*.

Ruifang He, Liangliang Zhao, and Huanyu Liu. 2020. Tweetsum: Event oriented social summarization dataset. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 5731–5736.

Ruining He and Julian McAuley. 2016. Ups and downs: Modeling the visual evolution of fashion trends with one-class collaborative filtering. In *proceedings of the 25th international conference on world wide web*, pages 507–517.

Neslihan Iskender, Oleg Vasilyev, Tim Polzehl, John Bohannon, and Sebastian Möller. 2021. Towards human-free automatic quality evaluation of german summarization. *arXiv preprint arXiv:2105.06027*.

Vijay Keswani and L Elisa Celis. 2021. Dialect diversity in text summarization on twitter. In *Proceedings of the Web Conference 2021*, pages 3802–3814.

Misha Khalman, Yao Zhao, and Mohammad Saleh. 2021. ForumSum: A multi-speaker conversation summarization dataset. In *Findings of the Association for Computational Linguistics: EMNLP 2021*, pages 4592–4599, Punta Cana, Dominican Republic. Association for Computational Linguistics.

Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Ves Stoyanov, and Luke Zettlemoyer. 2019. Bart: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. *arXiv preprint arXiv:1910.13461*.

Quanzhi Li and Qiong Zhang. 2020. Abstractive event summarization on twitter. In *Companion Proceedings of the Web Conference 2020*, pages 22–23.

Chen Lin, Zhichao Ouyang, Xiaoli Wang, Hui Li, and Zhenhua Huang. 2021a. Preserve integrity in real-time event summarization. *ACM Transactions on Knowledge Discovery from Data (TKDD)*, 15(3):1–29.

Chin-Yew Lin. 2004. Rouge: A package for automatic evaluation of summaries. In *Text summarization branches out*, pages 74–81.

Haitao Lin, Liqun Ma, Junnan Zhu, Lu Xiang, Yu Zhou, Jiajun Zhang, and Chengqing Zong. 2021b. Csds: A fine-grained chinese dataset for customer service dialogue summarization.

Xinfan Meng, Furu Wei, Xiaohua Liu, Ming Zhou, Sujian Li, and Houfeng Wang. 2012. Entity-centric topic-oriented opinion summarization in twitter. In *Proceedings of the 18th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 379–387.

Rada Mihalcea and Paul Tarau. 2004. Textrank: Bringing order into text. In *Proceedings of the 2004 conference on empirical methods in natural language processing*, pages 404–411.

Derek Miller. 2019. Leveraging bert for extractive text summarization on lectures. *arXiv preprint arXiv:1906.04165*.

Ramesh Nallapati, Bowen Zhou, Cicero Nogueira dos santos, Caglar Gulcehre, and Bing Xiang. 2016. Abstractive text summarization using sequence-to-sequence rnns and beyond.

Minh-Tien Nguyen, Dac Viet Lai, Huy-Tien Nguyen, and Le-Minh Nguyen. 2018. TSix: A human-involved-creation dataset for tweet summarization. In *Proceedings of the Eleventh International Conference on Language Resources and Evaluation (LREC 2018)*, Miyazaki, Japan. European Language Resources Association (ELRA).

Laura Perez-Beltrachini and Mirella Lapata. 2021. Models and datasets for cross-lingual summarisation. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 9408–9423, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J Liu. 2019. Exploring the limits of transfer learning with a unified text-to-text transformer. *arXiv preprint arXiv:1910.10683*.

Thomas Scialom, Sylvain Lamprier, Benjamin Piwowarski, and Jacopo Staiano. 2019. Answers unite! unsupervised metrics for reinforced summarization models. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 3246–3256, Hong Kong, China. Association for Computational Linguistics.

Anurag Shandilya, Kripabandhu Ghosh, and Saptarshi Ghosh. 2018. Fairness of extractive text summarization. In *Companion Proceedings of the The Web Conference 2018*, pages 97–98.

Rachael Tatman. 2017. Gender and dialect bias in youtube's automatic captions. In *Proceedings of the first ACL workshop on ethics in natural language processing*, pages 53–59.

Daniel Varab and Natalie Schluter. 2021. MassiveSumm: a very large-scale, very multilingual, news summarisation dataset. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 10150–10161, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Oleg Vasilyev, Vedant Dharnidharka, and John Bohannon. 2020. Fill in the BLANC: Human-free quality estimation of document summaries. In *Proceedings of the First Workshop on Evaluation and Comparison of NLP Systems*, pages 11–20, Online. Association for Computational Linguistics.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Rémi Louf, Morgan Funtowicz, et al. 2019. Huggingface's transformers: State-of-the-art natural language processing. *arXiv preprint arXiv:1910.03771*.

Xinnuo Xu, Ondřej Dušek, Shashi Narayan, Verena Rieser, and Ioannis Konstas. 2021. Miranews: Dataset and benchmarks for multi-resource-assisted news summarization.

A Preprocessing Documents

Raw tweets can be notoriously noisy and challenging for summarization purposes[3]. Our preprocessing steps are as follows: (i) we considered a tweet as belonging to one of the three dialect groups if it had a dialect confidence score greater than 0.7 for a given dialect and less than 0.3 for all the other dialects, (ii) we removed any duplicate tweets, any mentions (i.e., @username), and any tweets shorter than seven tokens in length, and (iii) finally, since emojis may provide useful indicators during summary generation, we converted the Unicode emoji characters with their corresponding images.

After the initial preprocessing, we extracted a list of hashtags present in the remaining tweets with the goal of identifying the most frequent topics of at least thirty tweets per dialect. This considerably filtered down the set of tweets because of unbalanced distribution of dialect groups in the corpus, with significantly more *White* tweets than *AA* or *Hispanic* tweets. Finally, we found and settled on a set of twenty five topics that we hypothesize encompass tweets from a diverse set of users. These include: 49ers, Amazon, Beyonce, Chicago, Christmas, Eagles, Facebook, Flu, Graduation, Grammys, Iphone, Kobe, McDonalds, NBA, Netflix, NYC, Obama, Paris, Patriots, Seahawks, Superbowl, Thanksgiving, VMA, WWE, Xbox.

B Implementation Details

TEXTRANK model[4] was initiated with the word count set to 70 - which is the average number of tokens for 5 sentences in our dataset. BERT-EXT and LONGFORMER-EXT models were initiated from the extractive summarization model[5], with the number of output sentences set to 5. For the abstractive models BART, T5, and LED, we used pretrained model checkpoints BART_base[6], T5_base[7],and AllenAI LED_base_16384[8], respectively, with beam size set to 3 and minimum length of tokens set to 70. The model checkpoints were accessed from the HuggingFace library (Wolf et al., 2019) and further fine-tuned using the TWEETSUMM dataset (Feigenblat et al., 2021) which was chosen as it is one of the most similar tweet datasets to ours that was large enough to serve as a training set for fine-tuning purposes. For automatic clustering, we used k-means clustering with *tf-idf* vector representation, and set $k = 2$ for all our experiments after assessing it to generate reasonable results using silhouette coefficient scores[9]. The reference-based and reference-less evaluation metrics were computed using the SummEval toolkit[10] (Fabbri et al., 2021).

C Example System Summaries

Tables 7 and 8 present some system summaries as generated by an extractive model and an abstractive model, respectively.

D Representation Distribution

Figure 5 displays the violin plots for $\mathcal{R}(\mathcal{S})$ per dialect and approach for (a) BERT-Ext, and (b) LongFormer-Ext.

[3] And not to mention, potentially offensive. However, other than these preprocessing steps, we intentionally did not filter out any further tweets, neither automatically nor manually, in order to avoid inserting any biases. Given the nature of social media posts, it is possible that the dataset may thus unintentionally contain some offensive content. The annotators were carefully informed about the risks of participating in such a study.

[4] https://github.com/RaRe-Technologies/gensim

[5] https://pypi.org/project/bert-extractive-summarizer/

[6] https://huggingface.co/facebook/bart-base

[7] https://huggingface.co/t5-base

[8] https://huggingface.co/allenai/led-base-16384

[9] https://scikit-learn.org/stable/index.html

[10] https://github.com/Yale-LILY/SummEval

Approach	Summary
REFERENCE	Looks like I'm watching the sugar bowl game cz its halftime for the NBA. And the Lakers get back in the game by slowing it down, grinding on 'D', and going at the basket, Who knew? 0_0 #NBA. Aha best team in the NBA they by far better then your Celtics y'all lost last night btw to the grizzlies. NBA basketball n packing for the rest of my night. Miami comin out the east & Clippers comin out the west #NBA #FinalsPrediction.
VANILLA	Looks like I'm watching the sugar bowl game cz its halftime for the NBA. yea on NBA TV, they PLAYN n London" Oh Aite Thanks. Watching and NBA game live > Watching the play by play on an iPhone.. As I sit here lowkey buzzed playing NBA 2k13 thinking I'm actually in the tv screen. I love explaining the NBA to my mom & telling her which players are good & stuff like that.. Tim Duncan is always on my make believe NBA teams cause he's my favorite PF,. When girls actually know what they're talking about the NBA.
CLUSTER-H	When girls actually know what they're talking about the NBA. Any subliminal pics I post on IG will now be hash tagged #HNBAF. to comeback and do commentary on some Joker episodes? # West is actually good 1-8.. Tim Duncan is always on my make believe NBA teams cause he's my favorite PF,. NFL, NBA, and NHL were lockouts by owners.. Diandre doesn't understand there isn't really parity in NBA.
CLUSTER-A	The #Memphisgrizzlies just made the stupidest trade in NBA history trading Rudy gay at this point in the season #wow. NBA games are much better in person.. Looks like I'm watching the sugar bowl game cz its halftime for the NBA. Up watching NBA TV... He need to stick to coaching or being a GM or some shit.... @ The Black Mamba 81 point showcase in showing now on NBATV !!. Tim Duncan is always on my make believe NBA teams cause he's my favorite PF,.. he has a higher ceiling than granger but gotta hold off on calling him better, PG has never been top 5 in NBA in scoring. The Heat get every call to go their way...the NBA should be ashamed of this BS.

Table 7: An example of system summaries (along with human-generated reference summary) using BERT-Ext model

Approach	Summary
REFERENCE	It's an NBA game night. Many people are tuned in because it's the first game of the season. There are different reactions to the game because some think it's awesome and some think it's whack. It would greatly have to do with fans(the team they support). Viewers also gave opinions of different players they consider to be the best and Lebron is thought to be overhyped.
VANILLA	Looks like I'm watching the sugar bowl game and its halftime for the NBA.
CLUSTER-H	And the #Lakers get back in the game by slowing it down, grinding on 'D', and going at the basket. And the #Lakers get back in the game by slowing it down, grinding on 'D', and going at the basket. And the #Lakers get back in the game by slowing it down, grinding on 'D', and going at the basket.
CLUSTER-A	@ RobHall their are truly more lakers haters then Miami haters.. Who knew? he has a higher ceiling than granger but gotta hold off on calling him better, PG has never been top 5 in scoring. @robhall their are truly more lakers haters then Miami haters.. Who knew? he has a higher ceiling than granger but gotta hold off on calling him better, PG has never been top 5 in scoring. And the #Lakers get back in the game by slowing it down, grinding on 'D', and going at the basket. Looks like I'm watching the sugar bowl game and its halftime for the NBA

Table 8: An example of system summaries (along with human-generated reference summary) using LED model

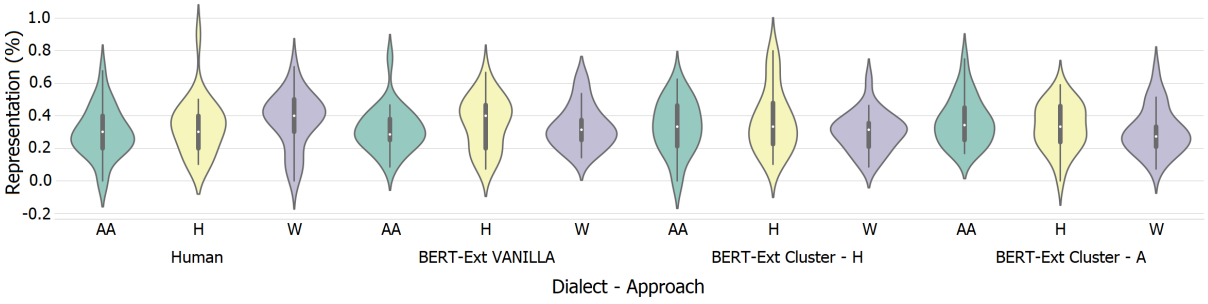

(a)

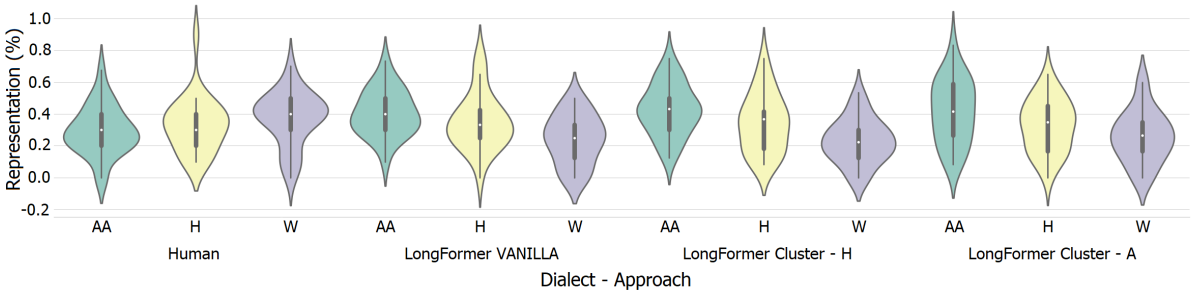

(b)

Figure 5: *Violin plots of $\mathcal{R}(\mathcal{S})$ per dialect and approach, with (a) BERT-Ext, and (b) LongFormer-Ext models.* The values for $\mathcal{R}$ were determined across two runs and averaged. Of interest is how the models compare to the human-generated summaries. For BERT-Ext, the *White* dialect contains the consistently widest violins, indicating a more consistent $\mathcal{R}(\mathcal{S})$ average around $\sim$0.33. The violins for the *AA* and *Hispanic* dialects are skinnier, suggesting that they contain more outliers above and below the $\sim$0.33 mark. For LongFormer-Ext, once again the *White* dialect contains consistently wide violins, whereas the *AA* violins are wide for both VANILLA and CLUSTER-H approaches, but contain more outliers for the Cluster-A approach. The *Hispanic* dialect violins contain more outliers for both CLUSTER-H and CLUSTER-A approaches. More outliers is indicative of less consistent representation that deviates from the desired equal $\mathcal{R}$ value of 0.33 in the case of our equally represented groups.

Multi-Document Scientific Summarization from a Knowledge Graph-Centric View

Pancheng Wang, Shasha Li†, Kunyuan Pang, Liangliang He, Dong Li, Jintao Tang†, Ting Wang†
College of Computer Science and Technology,
National University of Defense Technology, Changsha, China
{wangpancheng13, shashali, pangkunyuan10, heliangliang19, lidong1,tangjintao, tingwang}@nudt.edu.cn

Abstract

Multi-Document Scientific Summarization (MDSS) aims to produce coherent and concise summaries for clusters of topic-relevant scientific papers. This task requires precise understanding of paper content and accurate modeling of cross-paper relationships. Knowledge graphs convey compact and interpretable structured information for documents, which makes them ideal for content modeling and relationship modeling. In this paper, we present **KGSum**[1], an MDSS model centred on knowledge graphs during both the encoding and decoding process. Specifically, in the encoding process, two graph-based modules are proposed to incorporate knowledge graph information into paper encoding, while in the decoding process, we propose a two-stage decoder by first generating knowledge graph information of summary in the form of descriptive sentences, followed by generating the final summary. Empirical results show that the proposed architecture brings substantial improvements over baselines on the Multi-Xscience dataset.

1 Introduction

Nowadays, the exponential increasing publication rate of scientific papers makes it difficult and time-consuming for researchers to keep track of the latest advances. **M**ulti-**D**ocument **S**cientific **S**ummarization (MDSS) is therefore introduced to alleviate this information overload problem by generating succinct and comprehensive summary from clusters of topic-relevant scientific papers (Chen et al., 2021; Shah and Barzilay, 2021).

In MDSS, paper content modeling and cross-paper relationship modeling are two main issues. (1) Scientific papers contain complex concepts, technical terms, and abbreviations that convey important information about paper content. However, some previous works (Wang et al., 2018; Jiang et al., 2019) treat all text units equally, which inevitably ignore the salient information of some less frequent technical terms and abbreviations. (2) Furthermore, there exist intricate relationships between papers in MDSS, such as sequential, parallel, complementary and contradictory (Luu et al., 2021), which play a vital role in guiding the selection and organization of different contents. The latest work (Chen et al., 2021) attempt to capture cross-paper relationships via seq2seq model without considering any links between fine-grained text units. Failure to take into account explicit relationships between papers prevents their model from learning cross-paper relationships effectively.

To address the two aforementioned issues, we consider leveraging salient text units, namely entities, and their relations for MDSS. Scientific papers contain multiple domain-specific entities and relations between them. These entities and relations succinctly capture important information about the main content of papers. Knowledge graphs based on these scientific entities and relations can be inherently used for content modeling of scientific papers. Take Figure 1 as an example. The knowledge graph at the top left illustrates the main content of paper 1, which can be formulated as: *Paper 1 uses memory augmented networks method to solve the life-long one-shot learning task, the evaluation is based on image classification benchmark datasets*. Furthermore, knowledge graphs can effectively capture cross-paper relationships through entity interactions and information aggregation. In Figure 1, paper 1, 2 and 3 adopt the same method *memory networks* to solve different tasks. This relationship is demonstrated in the graph of gold summary by sharing the method node *memory networks*.

In this paper, we develop a Transformer-based (Vaswani et al., 2017) abstractive MDSS model, which can leverage knowledge graphs to

†Corresponding authors.
[1]https://github.com/muguruzawang/KGSum

Proceedings of the 29th International Conference on Computational Linguistics, pages 6222–6233
October 12–17, 2022.

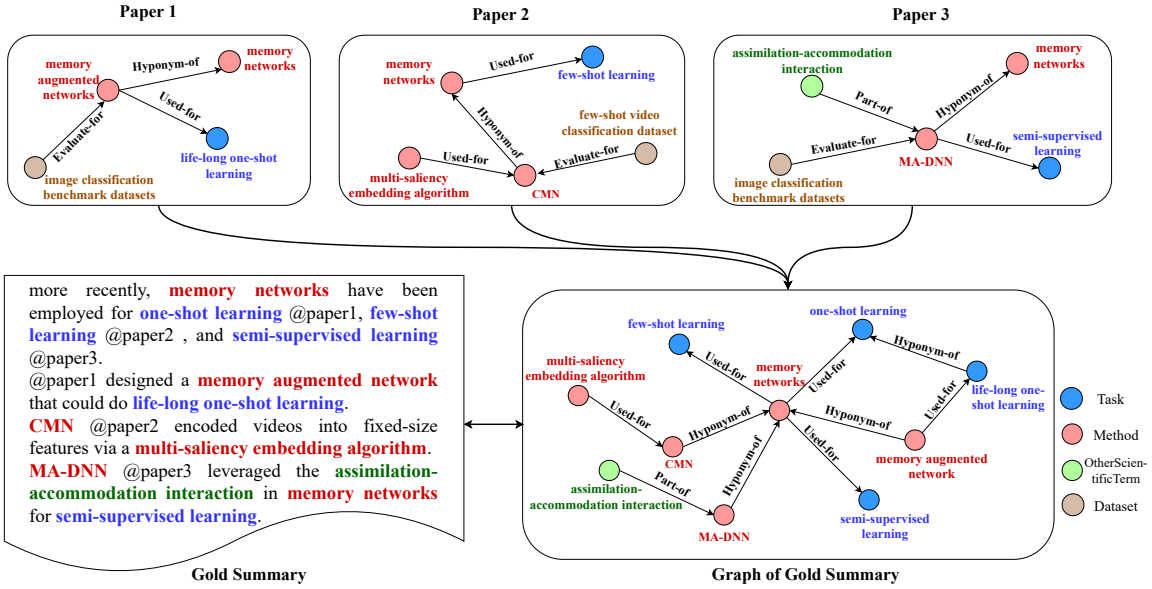

Figure 1: Knowledge graphs constructed from abstract of input scientific papers and gold summary.

guide paper representation and summary generation. Specifically, in the encoding part, we fuse the knowledge graphs of multiple input papers into a unified graph and design a graph updater to capture cross-paper relationships and global information. Besides, we build another graph based on the interaction between entities and sentences, and then apply an entity-sentence updater to enable information flow between nodes and update sentence representations.

In the decoding part, knowledge graphs are utilized to guide the summary generation process via two approaches. The first is to incorporate the graph structure into the decoder by graph attention, and the second is inspired by deliberation mechanism (Xia et al., 2017; Li et al., 2019). Concretely, we introduce a two-stage decoder to make better use of the guidance information of knowledge graphs. The first-stage decoder concentrates on generating the knowledge graph of gold summary, while the second-stage decoder generates the summary based on the output of the first stage and the input papers. Since the knowledge graph of gold summary is in the form of graph structure, we translate the graph into equivalent descriptive sentences containing corresponding entities and relations, called **KGtext**. KGtext serves as an information-intact alternative to the knowledge graph of gold summary and is generated in the first-stage decoder, which we call the KGtext generator.

We test the effectiveness of our proposed model on Multi-XScience (Lu et al., 2020), a large-scale dataset for MDSS. Experimental results show that our proposed knowledge graph-centric model achieves considerable improvement compared with the baselines, indicating that knowledge graphs can exert a positive impact on paper representation and summary generation.

The main contribution is threefold: (i) We leverage knowledge graphs to model content of scientific papers and cross-paper relationships, and propose a novel knowledge graph-centric model for MDSS. (ii) We propose a two-stage decoder that introduces KGtext as intermediate output when decoding, which plays an important guiding role in the final summary generation. (iii) Automatic and human evaluation results on the Multi-Xscience dataset show the superiority of our model.

2 Approach

2.1 Problem Formulation

We first introduce the problem formulation and used notations for MDSS. Given a set of query-focused scientific papers $\mathcal{D} = \{d_1, d_2, ..., d_N\}$, where N denotes the number of input papers. Each paper d_i consists of M_i sentences $\{s_{i,1}, s_{i,2}, ...s_{i,M_i}\}$, while each sentence $s_{i,j}$ consists of $K_{i,j}$ words $\{w_{i,j,1}, w_{i,j,2}, ..., w_{i,j,K_{i,j}}\}$. The gold summary $S = \{w_1, w_2, ...w_{N_s}\}$, N_s is the number of words in the gold summary. The target is to generate a summary $\hat{S} = \{\hat{w}_1, \hat{w}_2, ...\hat{w}_{N_{\hat{s}}}\}$ that is close enough to the gold summary S.

In our two-stage decoder framework, the gold KGtext $T = \{w_{t_1}, w_{t_2}, ..., w_{t_{\hat{N}}}\}$ is also attached

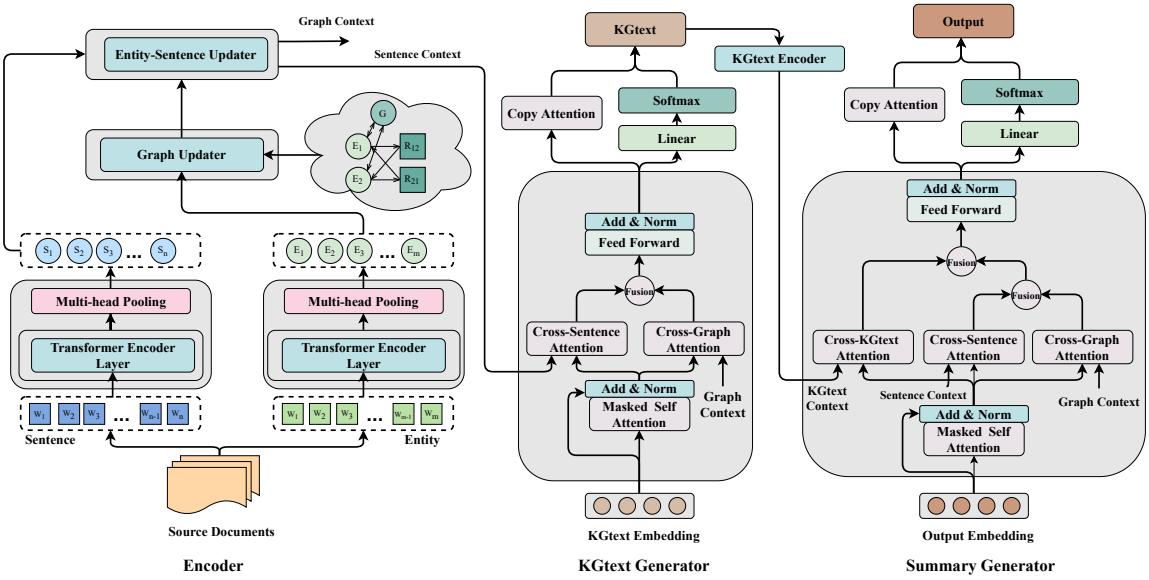

Figure 2: The overall framework of our proposed model.

as input. Hence, the probability of generating the gold summary S is

$$P(S|\mathcal{D}) = P_{\theta_{\mathcal{D}\to T}}(T|\mathcal{D}) * P_{\theta_{(\mathcal{D},T)\to S}}(S|\mathcal{D},T) \quad (1)$$

where $\theta_{\mathcal{D}\to T}$ and $\theta_{(\mathcal{D},T)\to S}$ are the parameters for the first-stage KGtext generator and the second-stage summary generator, respectively.

2.2 Graph Construction

To construct the knowledge graphs for input papers, we first employ the SciIE system DYGIE++ (Wadden et al., 2019), a well-performed science-domain information extraction system, to extract entities, relations and co-references from papers. Entities are classified into six types (*Task, Method, Metric, Material, Generic*, and *OtherScientificTerm*), and relations are classified into seven types (*Compare, Used-for, Feature-of, Hyponym-of, Evaluate-for, Part-of*, and *Conjunction*). Besides, we collapse coreferential entity clusters into a single node based on the annotation result.

After obtaining the knowledge graphs of multiple input papers, we fuse them into a unified graph. Then we follow the Levi transformation (Levi, 1942) to treat each entity and relation equally. Concretely, each labeled edge is represented as two vertices: one denoting the forward relation and another denoting the reverse relation. Formally, given an entity-relation tuple (e_1, r, e_2), we create nodes e_1, e_2, r_1 and r_2, and add directed edges $e_1 \to r_1, r_1 \to e_2$ and $e_2 \to r_2, r_2 \to e_1$. In this way, the original knowledge graph is reconstructed as an unlabeled directed graph without information loss. Besides, to guarantee the connectivity of Levi graph, we add a global vertex that connects all the entity vertices. We also add entity type nodes and connect all the entities to their corresponding types.

2.3 Model Description

Our model follows a Transformer-based encoder-decoder architecture, shown in Figure 2. The encoder includes a stack of L_1 token-level Transformer encoding layers to encode contextual information for tokens within each sentence and each entity. The Transformer encoding layer follows the Transformer architecture introduced in Vaswani et al. (2017). The encoder also includes a **Graph Updater** to learn the graph representation of the knowledge graph and an **Entity-Sentence Updater** to update entity representation and sentence representation based on their interaction. The decoder consists of a **KGtext Generator**, which produces the descriptive sentences of the graph of gold summary, and a **Summary Generator**, which produces the final summary.

2.4 Graph Updater

As shown in Figure 2, based on the output of token-level Transformer encoding layers, the graph updater is used to encode the knowledge graphs to obtain graph representations of input papers.

Node Initialization The vertices of the constructed graph correspond to entities, relations and entity types from the SciIE annotations. Entities

representations are produced using the aforementioned Transformer-based encoding method. For a given entity co-reference cluster, we first remove pronouns and stopwords and then obtain the entity representation by using the average embedding of entities in the cluster. For relation representation, since each relation is represented as both forward and backward vertices, we learn two embeddings per relation. We also randomly initialize the types embeddings and the global vertex embedding.

Contextualized Node Encoding We follow Koncel-Kedziorski et al. (2019) and use a Graph Transformer to compute the hidden representations of each node in the graph. Graph Transformer encodes each vertex v_i using the multi-head self-attention mechanism similar to Vaswani et al. (2017), where each vertex representation $\mathbf{v}_i$ is contextualized by attending over the other vertices to which v_i is connected in the graph.

$$\hat{\mathbf{v}}_i = \mathbf{v}_i + \|_{n=1}^{N} \sum_{v_j \in \mathcal{N}_i} \alpha_{i,j}^n \mathbf{W}_V^n \mathbf{v}_j \quad (2)$$

$$\alpha_{i,j}^n = \text{softmax}((\mathbf{W}_K^n \mathbf{v}_j)^T (\mathbf{W}_Q^n \mathbf{v}_i)) \quad (3)$$

where $\|_{n=1}^{N}$ means the concatenation of N heads. $\mathcal{N}_i$ denotes the neighbors of v_i, and $\mathbf{W}_Q^n$, $\mathbf{W}_K^n$, and $\mathbf{W}_V^n$ are trainable parameters of query, key and value of head n, respectively.

2.5 Entity-Sentence Updater

After getting the contextualized node embeddings for the knowledge graph, we construct an entity-sentence heterogeneous graph to update sentence representations based on the interaction between entities and sentences. The entity-sentence graph is denoted as $G = \{V, E\}$, where V stands for nodes set and E stands for edges set. In the graph G, V includes entity nodes V_e and sentence nodes V_s, and E is a real-value edge weight matrix, where $e_{i,j} \neq 0$ indicates the j-th sentence contains the i-th entity.

We apply the same Graph Transformer module as the graph updater. It takes as input the entities representations from the graph updater and the sentence representations from the Transformer encoding layer, then learns the representations of nodes based on the information flow through the graph G.

2.6 KGtext Generator

In the decoding stage, we also adopt the knowledge graph-centric view and introduce the KGtext generator before the final summary generator. Here, KGtext is defined as descriptive sentences containing entities and relations translated from the knowledge graph of gold summary. An example of KGtext is shown in Figure 3.

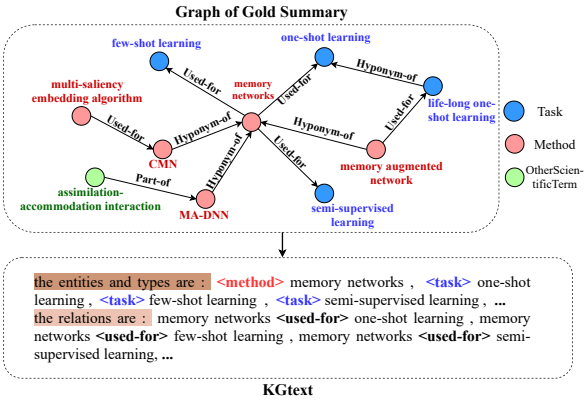

Figure 3: An example of graph of gold summary and the translated KGtext.

KGtext Construction To construct KGtext, we first use DYGIE++ (Wadden et al., 2019) to extract entities and relations from the human-written gold summary of the training set. Then we fill the KGtext with the prefix *The entities and types are:* followed by each entity type and entity pair like *<TYPE> ENT*. We also add another prompt *the relations are:* to introduce the relations, in the form of *ENT_1 <REL> ENT_2*.

KGtext serves as an information-intact alternative to the knowledge graph of gold summary, which is generated by the KGtext generator and can provide knowledge graphs information for the final summary generation.

Decoding Since the knowledge graph of gold summary is obtained by synthesizing and simplifying the knowledge graphs of input papers via the interaction of nodes, the graph structure plays an important role in KGtext generation. Hence during decoding, we leverage source token representations as well as graph representations during KGtext decoding process.

We apply a stack of L_2 Transformer decoding layers as the decoder. The cross-attention sub-layer of each decoding layer computes two multi-head attention to capture both textual and graph context. Let $\tilde{g}_i^l$ denotes the i-th token output representation by the l-th self-attention sub-layer. For the textual context, we use $\tilde{g}_i^l$ as query and token representations $\mathbf{H_W}$ from entity-sentence updater as keys

and values.

$$c_{i,w}^{l} = \mathrm{MHAtt}(\tilde{g}_i^l, \mathbf{H_W}, \mathbf{H_W}) \quad (4)$$

where MHAtt denotes the multi-head attention module proposed in Vaswani et al. (2017).

For the graph context, we use $\tilde{g}_i^l$ as query and entity nodes representations $\mathbf{H_E}$ from entity-sentence updater as keys and values. Considering that different entities of the input have different importance, we apply the unsupervised phrase scoring algorithm RAKE (Rose et al., 2010) to score the salience of entities, and incorporate entity salience into graph context computation. Given the RAKE scores $S = \{s_j\}$ for entity nodes representations $\mathbf{H_E}$, we modify MHAtt module by multiplying S with the attention weights.

$$c_{i,g}^{l} = \mathrm{MHAtt\_Mod}(\tilde{g}_i^l, \mathbf{H_E}, \mathbf{H_E}, S) \quad (5)$$

where MHAtt_Mod denotes the modified MHAtt module. And the modified attention weight α_t^n of head n is calculated as

$$\alpha_i^n = \frac{(\mathrm{W}_K^n \mathbf{H_E})^T (\mathrm{W}_Q^n \tilde{g}_i^l)}{\sqrt{d_{head}}} * S \quad (6)$$

where W_K^n and W_Q^n are parameter weights, d_{head} denotes the dimension of each attention head.

We then add a fusion gate to merge both the textual context and the graph context.

$$c_i^l = z * c_{i,w}^l + (1 - z) * c_{i,g}^l \quad (7)$$

$$z = \mathrm{sigmoid}([c_{i,w}^l; c_{i,g}^l]\mathrm{W}_f + b_f) \quad (8)$$

where W_f and b_f are the linear transformation parameter. The feed-forward network is used to further transform the output.

$$g_i^l = \mathrm{LayerNorm}(c_i^l + \mathrm{FFN}(c_i^l)) \quad (9)$$

The generation distribution p_t^g over the target vocabulary is calculated by feeding the output $g_t^{L_2}$ to a softmax layer.

$$p_i^g = \mathrm{softmax}(g_i^{L_2}\mathrm{W}_g + b_g) \quad (10)$$

where W_g and b_g are learnable linear transformation parameter.

Furthermore, we also employ the copy mechanism (See et al., 2017) to alleviate the out-of-vocabulary (OOV) problem. The final generation distribution p_i^t is the "mixture" of both p_i^g and the copy probability over source words p_i^c.

The loss is the negative log likelihood of the gold KGtext w_{t_i}:

$$L_T = -\frac{1}{\hat{N}} \sum_{i=1}^{\hat{N}} \log p_i^t(w_{t_i}) \quad (11)$$

2.7 Summary Generator

The final summary generator has a similar decoding architecture to the KGtext generator, but differs in that the summary generator utilizes the generated KGtext to guide summary generation.

Given the KGtext generative distribution $\{p_i^t\}$, we obtain the decoding sequence of KGtext $\hat{T}$ by greedy search during training. Then we add an encoder similar to the aforementioned sentence encoder to get the KGtext representations $\mathbf{H_T}$. Besides attending to textual and graph context, we use the same multi-head attention as equation (4) to compute KGtext context $\hat{c}_{i,t}^l$ to capture KGtext influence.

Together with the textual context $\hat{c}_{i,w}^l$ and the graph context $\hat{c}_{i,g}^l$, we apply a hierarchical fusion mechanism to combine the three contexts, by first merging the textual context and the graph context, and then the KGtext context.

$$\hat{c}_i^l = z_1 * \hat{c}_{i,w}^l + (1 - z_1) * \hat{c}_{i,g}^l \quad (12)$$

$$z_1 = \mathrm{sigmoid}([\hat{c}_{i,w}^l; \hat{c}_{i,g}^l]\mathrm{W}_{1,f} + b_{1,f}) \quad (13)$$

$$c_i^l = z_2 * \hat{c}_i^l + (1 - z_2) * \hat{c}_{i,t}^l \quad (14)$$

$$z_2 = \mathrm{sigmoid}([\hat{c}_i^l; \hat{c}_{i,t}^l]\mathrm{W}_{2,f} + b_{2,f}) \quad (15)$$

where $\mathrm{W}_{1,f}$, $b_{1,f}$, $\mathrm{W}_{2,f}$ and $b_{2,f}$ are the linear transformation parameter.

Given the final summary generation distribution p_i^s, the loss is the negative log likelihood of the gold summary w_i:

$$L_S = -\frac{1}{N_s} \sum_{i=1}^{N_s} \log p_i^s(w_i) \quad (16)$$

2.8 Training Strategy

We train the KGtext generator and the summary generator in a unified architecture in an end-to-end manner. Furthermore, in practice, we find the KGtext generated from greedy search has a strong influence on the summary generation. The low-quality KGtext greatly impairs the performance of the model. Hence, we train another auxiliary decoder on top of $P_{\theta_{\mathcal{D}\to\mathrm{S}}}(S|\mathcal{D})$, which uses a one-stage decoder without generating KGtext. It has the same architecture as the summary generator except for the cross-attention on KGtext.

Given the final summary generation distribution of the auxiliary decoder $\tilde{p}_i^s$, the loss is the negative log likelihood of the gold summary w_i:

$$L_A = -\frac{1}{N_s} \sum_{i=1}^{N_s} \log \tilde{p}_i^s(w_i) \quad (17)$$

The final loss function is:

$$\mathcal{L} = \mathcal{L}_S + \lambda \mathcal{L}_T + \eta \mathcal{L}_A \quad (18)$$

where λ and η are both hyper parameters. In this way, we can reduce the effect of some low-quality generated KGtext and improve the stability of our model.

3 Experiments

3.1 Dataset

We conduct experiments on the recently released Multi-Xscience dataset (Lu et al., 2020), which is the first large-scale and open MDSS dataset. It contains 30,369 instances for training, 5,066 for validation and 5,093 for test. On average, each source paper cluster contains 4.42 papers and 778.08 words, and each gold summary contains 116.44 words.

3.2 Implementation Details

We set our model parameters based on preliminary experiments on the validation set. We prune the vocabulary to 50K. The number of encoding layer L_1 and the number of decoding layer L_2 are both 6. We set the dimension of word embeddings and hidden size to 256, feed-forward size to 1,024. We set 8 heads for multi-head attention. For the Graph Transformer of the graph updater and the entity-sentence updater, we set the number of iterations to 3. We set dropout rate to 0.1 and label smoothing (Szegedy et al., 2016) factor to 0.1. We use Adam optimizer with learning rate $\alpha = 0.02$, $\beta_1 = 0.9$, $\beta_2 = 0.998$; we also apply learning rate warmup over the first 8000 steps, and decay as in Vaswani et al. (2017). The batch size is set to 8. λ and η are both set to 1.0. The model is trained on 1 GPU (NVIDIA Tesla V100, 32G) for 100,000 steps. We select the top-3 best checkpoints based on performance on the validation set and report averaged results on the test set.

For KGtext decoding, we use greedy search with the minimal generation length 100, while for summary decoding, we use beam search with beam size 5 and the minimal generation length is 110, consistent with Lu et al. (2020). The length penalty factor is set to 0.4.

3.3 Metrics and Baselines

We use ROUGE F_1 (Lin, 2004) to evaluate the summarization quality. Following previous work, we report unigram and bigram overlap (ROUGE-1 and ROUGE-2) as a means of assessing informativeness and the longest common subsequence (ROUGE-L) as a means of assessing fluency.

Model	R-1	R-2	R-L
Extractive			
LexRank (Erkan and Radev, 2004)	30.19	5.53	26.19
TextRank (Mihalcea and Tarau, 2004)	31.51	5.83	26.58
HeterSumGraph* (Wang et al., 2020)	31.36	5.82	27.41
Ext-Oracle	38.45	9.93	27.11
Abstractive			
GraphSum* (Li et al., 2020)	29.58	5.54	26.52
Hiersumm (Liu and Lapata, 2019a)	30.02	5.04	27.6
HiMAP (Fabbri et al., 2019)	31.66	5.91	28.43
BertABS (Liu and Lapata, 2019b)	31.56	5.02	28.05
BART (Lewis et al., 2020)	32.83	6.36	26.61
SciBertABS (Lu et al., 2020)	32.12	5.59	29.01
MGSum* (Jin et al., 2020)	33.11	6.75	29.43
Pointer-Generator (See et al., 2017)	34.11	6.76	30.63
KGSum	**35.77**	**7.51**	**31.43**

Table 1: ROUGE F1 evaluation results on the test set of Multi-Xscience. The results marked with * are obtained by running the released code using the same beam size and decoding length. Other results without * are from Lu et al. (2020).

We compare our model with several typical extractive and abstractive summarization models. Due to space limitations, we put the introduction of these models in appendix A.

3.4 Automatic Evaluation

Table 1 summarizes the evaluation results on the Multi-Xscience dataset.

As can be seen, abstractive models generally outperform extractive models, especially on ROUGE-L, showing that abstractive models can generate more fluent summaries. Among the abstractive baselines, Pointer-Generator (See et al., 2017) surprisingly outperforms other newer models. We partially attribute this result to Pointer-Generator designing an additional coverage mechanism (Tu et al., 2016) to effectively reduce redundancy. This result also illustrates that MDSS is challenging and requires domain-specific solutions for paper content representation and cross-paper relationship modeling.

The last block reports the result of our model KGSum. KGSum outperforms any other models, achieving scores of 35.77, 7.51, and 31.43 on the three ROUGE metrics. Our model surpasses other models by a remarkable large margin (at least improving 1.66%, 0.75%, and 0.80%). The result

Model	Overall	Inf	Fluency	Succ
GraphSum	-1.42*	-1.47*	-1.08*	-1.23*
MGSum	-0.38*	0.60	-0.20*	-0.55*
Pointer-Generator	0.62*	0.31*	0.17*	0.60*
KGSum	**1.30**	**0.68**	**1.17**	**1.22**

Table 2: Human evaluation of system summaries on Multi-Xscience test set. Inf stands for *informativeness* and Succ stands for *succinctness*. The larger rating denotes better summary quality. * indicates the ratings of the corresponding model are significantly (by Welch's t-test with $p < 0.05$) outperformed by our model. The inter-annotator agreement score (Fleiss Kappa) is 0.63, which indicates substantial agreement between annotators.

Model	R-1	R-2	R-L
KGSum	**35.77**	**7.51**	**31.43**
- KGG	35.34	7.28	30.91
- KGG - RAKE	35.17	7.18	30.75
- KGG - RAKE - GU	34.97	7.08	30.63
- KGG - RAKE - GU - ESU	34.79	6.90	30.36

Table 3: Ablation studies on Multi-Xscience test set. We remove various modules and explore their influence on our model. '-' means the removal operation from KGSum. The last row (-KGG-RAKE-GU-ESU) is the clean baseline without any module we propose.

demonstrates that our model can generate more informative and more coherent summaries, indicating the effectiveness of our proposed knowledge graph-centric encoder and decoder framework.

3.5 Human Evaluation

We further access the linguistic quality of generated summaries by human evaluation. We randomly select 30 test instances from the Multi-Xscience test set, and invite three graduate students as annotators to evaluate the outputs of different models independently. Annotators assess the overall quality of summaries by ranking them considering the following criteria:: (1) *Informativeness*: does the summary convey important facts of the input papers? (2) *Fluency*: is the summary fluent and grammatical? (3) *Succinctness*: whether the summary contains repeated information? Annotators are asked to rank all systems from 1 (best) to 4 (worst). All systems get score 2, 1, -1, -2 for ranking 1, 2, 3, 4 respectively. The rating of each system is computed by averaging the scores on all test instances.

The result is shown in Table 2. The overall rating and the ratings for the above three aspects are reported. We can see that KGSum performs much

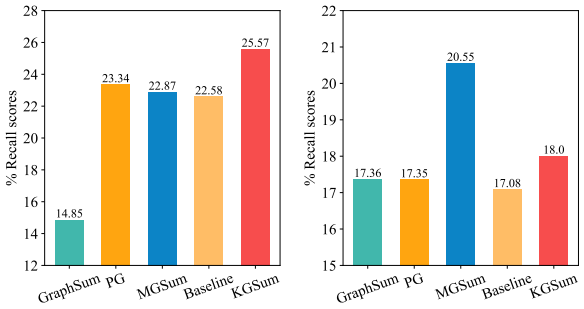

(a) Recall of EW (b) Recall of CW

Figure 4: Recall of EW and CW for different models on Multi-Xscience test set. PG stands for Pointer-Generator, Baseline is our Transformer-based model without any module we propose.

better than other models. The overall rating of KGSum achieves 1.2, which is much higher than 0.62, -0.38, and -1.42 of Pointer-Generator, MGSum, and GraphSum. The results on informativeness indicate our model can effectively capture the salient information of papers and generate more informative summaries. The results on fluency and succinctness indicate that KGSum is able to generate more fluent and concise summaries. Furthermore, Pointer-Generator achieves a much higher score on succinctness than MGSum, which further proves that Pointer-Generator generates less redundant summaries and thus has better performance.

3.6 Model Analysis

For a thorough understanding of KGSum, we conduct several experiments on Multi-Xscience test set.

Ablation Study Firstly, we perform an ablation study to validate the effectiveness of individual components. Here, **KGG** stands for KGtext generator, **RAKE** refers to the RAKE algorithm that measures entity salience, **GU** stands for graph updater, **ESU** stands for entity-sentence updater. We remove KGG, RAKE, GU, ESU one by one in order from decoder to encoder. The result is illustrated in Table 3. We find that the GU and ESU module in the encoder can effectively encode knowledge graph information and utilize knowledge graphs to enable better information flowing between text nodes, which is conducive to content modeling and relationship modeling. Using RAKE to measure entity salience also benefits a lot for graph context computation when decoding. Further, the KGG module also brings significant improvement, indi-

KGtext Variants	R-1	R-2	R-L
Ent	35.61	7.43	31.24
Ent+Type	35.67	7.42	31.29
Ent+Type+Rel	**35.77**	**7.51**	**31.43**

Table 4: Analysis of the impact of different KGtext contents on summarization.

cating our proposed two-stage decoder with KGtext generator is effective in generating summary under the guidance of knowledge graphs.

Recall of Entity Words In order to intuitively demonstrate the impact of knowledge graphs, we investigate the recall of gold summary entities in the generated summary. The exact match of entities is difficult because entities have different mentions. Therefore, we count recall of entity words instead. We classify the words in papers into two categories: **Entity Words** (EW) and **Context Words** (CW). EW are defined as words in papers that are recognized as entities by SciIE tools, while CW are words other than EW. We exclude stopwords when calculating EW and CW, because stopwords have no practical meaning. Then, we define *Recall of EW* as:

$$Recall_{EW} = \frac{\sum_{S \in \{Ref\}} \sum_{N_{EW} \in S} Count_{match}(N_{EW})}{\sum_{S \in \{Ref\}} \sum_{N_{EW} \in S} Count(N_{EW})} \quad (19)$$

where $\{Ref\}$ denotes the gold summaries, $Count_{match}(N_{EW})$ denotes the number of overlapped EW in the gold summaries and the generated summaries. $Count(N_{EW})$ denotes the number of EW in the gold summaries. *Recall of CW* is defined in a similar manner.

The results are shown in Figure 4. We find that KGSum achieves the highest recall of EW, compared with the baseline model and other models. The result proves that our model focuses on more entity information under the guidance of knowledge graphs. Conversely, in Figure 4b, MGSum achieves the highest recall of CW, but ROUGE-1/2/L scores of MGSum are only 33.11/6.75/29.43, falling behind KGSum. The result indicates that recall of CW has limited effect on model performance, which is in line with our intuition since CW do not contain important semantic information.

Influence of KGtext Contents We also conduct experiments to analyze the impact of different KGtext contents on MDSS. We consider the following three variants: (1) only entities (Ent), (2) entities + types (Ent+Type), (3) entities + types + relations (Ent+Type+Rel), to construct the KGtext using the same strategy in section 2.6. Result in Table 4 demonstrates MDSS can benefit from different components of knowledge graph, including entities, types and relations.

4 Related Work

Early MDSS works are mainly based on artificially constructed small-scale datasets, using unsupervised extractive methods to extract sentences from multiple papers. Mohammad et al. (2009) take citation texts, paper abstracts and full paper texts as input for survey generation. They conduct the experiment with just two instances. Jha et al. (2015a) construct 15 instances and combine a content model with a discourse model to generate coherent scientific summarizations. Hoang and Kan (2010) construct 20 instances, each with an annotated topic hierarchy tree, to generate summarization for multiple scientific papers. Similar works also exist in (Jha et al., 2015b; Hu and Wan, 2014; Yang et al., 2017). These unsupervised works are crude in both content modeling and relationship modeling and fail to generate high-quality summaries.

Some subsequent efforts apply deep learning-based methods to MDSS using large-scale datasets (Wang et al., 2018; Jiang et al., 2019; Chen et al., 2021). Wang et al. (2018) build a dataset with 8080 instances and construct a heterogeneous bibliography graph, and then utilize a CNN and RNN-based model for extractive summarization. Jiang et al. (2019) collect 390,000 instances, and use a hierarchical encoder and a two-step decoder to generate summary in an abstractive manner for the first time. Chen et al. (2021) collect two large-scale datasets with 136,655 and 80,927 instances, respectively. They apply a Transformer-based model to capture the relevance between papers for abstractive summarization. However, all the above works neglect salient semantic units to capture semantic information and relationships between papers. In this paper, based on Mutli-Xscience (Lu et al., 2020), we use knowledge graph information to model content and relationships between papers.

5 Conclusion

In this work, we propose a knowledge graph-centric Transformer-based model for MDSS. Our model is able to incorporate knowledge graph information

into the paper encoding process with a graph updater and an entity-sentence updater, and introduce a two-stage decoder including a KGtext generator and a summary generator to guide the summary decoding with knowledge graph information. Experiments show that the proposed model significantly outperforms all strong baselines and achieves the best result on the Multi-Xscience dataset.

In the future, we will explore other more intuitive and effective methods to incorporate graph information in both the encoding and decoding phase of summary generation.

Acknowledgements

This work was supported by the National Key Research and Development Project of China (No. 2021ZD0110700) and Hunan Provincial Natural Science Foundation (Grant Nos. 2022JJ30668). The authors would like to thank the anonynous reviewers for their valuable comments and suggestions to improve this paper.

References

Xiuying Chen, Hind Alamro, Mingzhe Li, Shen Gao, Xiangliang Zhang, Dongyan Zhao, and Rui Yan. 2021. Capturing relations between scientific papers: An abstractive model for related work section generation. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 6068–6077.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. Bert: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186.

Günes Erkan and Dragomir R Radev. 2004. Lexrank: Graph-based lexical centrality as salience in text summarization. *Journal of artificial intelligence research*, 22:457–479.

Alexander Richard Fabbri, Irene Li, Tianwei She, Suyi Li, and Dragomir Radev. 2019. Multi-news: A large-scale multi-document summarization dataset and abstractive hierarchical model. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 1074–1084.

Cong Duy Vu Hoang and Min-Yen Kan. 2010. Towards automated related work summarization. In *Proceedings of the 23rd International Conference on Computational Linguistics: Posters*, pages 427–435. Association for Computational Linguistics.

Yue Hu and Xiaojun Wan. 2014. Automatic generation of related work sections in scientific papers: an optimization approach. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1624–1633.

Rahul Jha, Reed Coke, and Dragomir Radev. 2015a. Surveyor: A system for generating coherent survey articles for scientific topics. In *Twenty-Ninth AAAI conference on artificial intelligence*.

Rahul Jha, Catherine Finegan-Dollak, Ben King, Reed Coke, and Dragomir Radev. 2015b. Content models for survey generation: a factoid-based evaluation. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, volume 1, pages 441–450.

Xiao-Jian Jiang, Xian-Ling Mao, Bo-Si Feng, Xiaochi Wei, Bin-Bin Bian, and Heyan Huang. 2019. Hsds: An abstractive model for automatic survey generation. In *International Conference on Database Systems for Advanced Applications*, pages 70–86. Springer.

Hanqi Jin, Tianming Wang, and Xiaojun Wan. 2020. Multi-granularity interaction network for extractive and abstractive multi-document summarization. In *Proceedings of the 58th annual meeting of the association for computational linguistics*, pages 6244–6254.

Rik Koncel-Kedziorski, Dhanush Bekal, Yi Luan, Mirella Lapata, and Hannaneh Hajishirzi. 2019. Text generation from knowledge graphs with graph transformers. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 2284–2293.

Friedrich Wilhelm Levi. 1942. Finite geometrical systems.

Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Veselin Stoyanov, and Luke Zettlemoyer. 2020. Bart: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7871–7880.

Wei Li, Xinyan Xiao, Jiachen Liu, Hua Wu, Haifeng Wang, and Junping Du. 2020. Leveraging graph to improve abstractive multi-document summarization. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 6232–6243.

Zekang Li, Cheng Niu, Fandong Meng, Yang Feng, Qian Li, and Jie Zhou. 2019. Incremental transformer with deliberation decoder for document grounded conversations. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 12–21.

Chin-Yew Lin. 2004. Rouge: A package for automatic evaluation of summaries. In *Text summarization branches out*, pages 74–81.

Yang Liu and Mirella Lapata. 2019a. Hierarchical transformers for multi-document summarization. *arXiv preprint arXiv:1905.13164*.

Yang Liu and Mirella Lapata. 2019b. Text summarization with pretrained encoders. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 3730–3740.

Yao Lu, Yue Dong, and Laurent Charlin. 2020. Multi-xscience: A large-scale dataset for extreme multi-document summarization of scientific articles. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 8068–8074.

Kelvin Luu, Xinyi Wu, Rik Koncel-Kedziorski, Kyle Lo, Isabel Cachola, and Noah A Smith. 2021. Explaining relationships between scientific documents. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 2130–2144.

Rada Mihalcea and Paul Tarau. 2004. Textrank: Bringing order into text. In *Proceedings of the 2004 conference on empirical methods in natural language processing*, pages 404–411.

Saif Mohammad, Bonnie Dorr, Melissa Egan, Ahmed Hassan, Pradeep Muthukrishnan, Vahed Qazvinian, Dragomir Radev, and David Zajic. 2009. Using citations to generate surveys of scientific paradigms. In *Proceedings of human language technologies: The 2009 annual conference of the North American chapter of the association for computational linguistics*, pages 584–592.

Stuart Rose, Dave Engel, Nick Cramer, and Wendy Cowley. 2010. Automatic keyword extraction from individual documents. *Text mining: applications and theory*, 1:1–20.

Abigail See, Peter J Liu, and Christopher D Manning. 2017. Get to the point: Summarization with pointer-generator networks. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1073–1083.

Darsh J Shah and Regina Barzilay. 2021. Generating related work. *arXiv preprint arXiv:2104.08668*.

Christian Szegedy, Vincent Vanhoucke, Sergey Ioffe, Jon Shlens, and Zbigniew Wojna. 2016. Rethinking the inception architecture for computer vision. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2818–2826.

Zhaopeng Tu, Zhengdong Lu, Yang Liu, Xiaohua Liu, and Hang Li. 2016. Modeling coverage for neural machine translation. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 76–85.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. *Advances in neural information processing systems*, 30.

David Wadden, Ulme Wennberg, Yi Luan, and Hannaneh Hajishirzi. 2019. Entity, relation, and event extraction with contextualized span representations. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 5784–5789.

Danqing Wang, Pengfei Liu, Yining Zheng, Xipeng Qiu, and Xuan-Jing Huang. 2020. Heterogeneous graph neural networks for extractive document summarization. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 6209–6219.

Yongzhen Wang, Xiaozhong Liu, and Zheng Gao. 2018. Neural related work summarization with a joint context-driven attention mechanism. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 1776–1786.

Yingce Xia, Fei Tian, Lijun Wu, Jianxin Lin, Tao Qin, Nenghai Yu, and Tie-Yan Liu. 2017. Deliberation networks: Sequence generation beyond one-pass decoding. *Advances in neural information processing systems*, 30.

Shansong Yang, Weiming Lu, Dezhi Yang, Xi Li, Chao Wu, and Baogang Wei. 2017. Keyphraseds: Automatic generation of survey by exploiting keyphrase information. *Neurocomputing*, 224:58–70.

Jingqing Zhang, Yao Zhao, Mohammad Saleh, and Peter Liu. 2020. Pegasus: Pre-training with extracted gap-sentences for abstractive summarization. In *International Conference on Machine Learning*, pages 11328–11339. PMLR.

A Baselines

LexRank (Erkan and Radev, 2004) and **TextRank** (Mihalcea and Tarau, 2004) are two unsupervised graph based extractive summarization models. **HeterSumGraph** (Wang et al.,

2020) is a heterogeneous graph-based extractive model with semantic nodes of different granularity. **HiMAP** (Fabbri et al., 2019) expands the pointer-generator network (See et al., 2017) into a hierarchical network and integrates an MMR module. **HierSumm** (Liu and Lapata, 2019a) is a Transformer based model with an attention mechanism to share information cross-document for abstractive multi-document summarization. **MGSum** (Jin et al., 2020) is a multi-granularity interaction network for abstractive multi-document summarization. We also consider evaluating on single document summarization models by concatenating multiple papers into a long sequence. **GraphSum** (Li et al., 2020) is a neural multi-document summarization model that leverages well-known graphs to produce abstractive summaries. We use TF-IDF graph as the input graph. **PEGASUS** (Zhang et al., 2020) is a sequence-to-sequence model with gap-sentences generation as a pre-training objective tailored for abstractive summarization. **Pointer-Generator** (See et al., 2017) is an RNN based model with an attention mechanism and allows the system to copy words from the source via pointing for abstractive summarization. **BertABS** (Liu and Lapata, 2019b) uses a pretrained BERT (Devlin et al., 2019) as the encoder for abstractive summarization. We also report the performance of BertABS with an encoder (**SciBertABS**) pretrained on scientific articles. **BART** (Lewis et al., 2020) is a pretrained text generation model.

B Case Study

In Figure 5, we present several example summaries to show the generating quality of different models. In the figure, there are five blocks, which are the reference summary (**Reference Summary**), the system summaries generated by **Pointer-Generator**, **GraphSum**, **MGSum** and our model **KGSum**. We highlight the salient contents from system summaries that can be find in the reference summary. We could find our model KGSum possesses the most salient contents and the highest overlap with the reference summary. Pointer-Generator and MGSum have the same amount of salient contents, but MGSum contains more repeated information, leading to worse performance. GraphSum contains the least salient contents, resulting in the worst performance.

Reference Summary:
most recent approaches rely on convolutional neural networks (cnns). @cite , the work in @cite is extended by adding a cnn to describe the posterior density of an object pose. a combination of using a cnn for object segmentation and geometry-based pose estimation is proposed in @cite . posecnn @cite uses a similar two-stage network, in which the first stage extracts feature maps from rgb input and the second stage uses the generated maps for object segmentation, 3d translation estimation and 3d rotation regression in quaternion format. depth data and icp are used for geometry refinement. @cite propose a three-stage, instance-aware approach for 6d object pose estimation. an instance segmentation network is first applied, followed by an encoder-decoder network which estimates the 3d object coordinates for each segment. the 6d pose is recovered with a geometric pose optimization step similar to @cite .

Pointer-Generator:
our work is also related to the work in @cite , where the authors use a convolutional neural networks (cnns) to predict the 6d pose of specific objects in a single rgb - d image . @cite proposed a new loss function for 6d pose estimation of object pose estimation . @cite , the authors propose a method to directly regress a pose vector from raw point cloud segments using a quaternion neural network . they then applied a quaternion representation to a quaternion model , which can be used to learn the pose of the object pose . in contrast to our work , our method is able to learn pose estimation from a single image

GraphSum:
3d object pose estimation has been a long-standing research topic in computer vision and computer vision @cite @cite @cite . in recent years, there has been significant interest in object reconstruction from a single image @cite @cite or 3d object @cite @cite , and many of them have been trying to recover from single images @cite . however, most of these methods rely on the availability of a large number of training images. in contrast, our approach does not require a large amount of labeled training data. in addition, our approach is able to estimate the 3d pose of a single object.

MGSum:
posecnn @cite is a flexible approach for 6d object pose estimation. it uses a cnn to predict 6d pose from a single image. posecnn @cite predicts 6d pose by regressing the 3d rotation of a 3d object by regressing a quaternion representation. posecnn @cite uses a quaternion network to predict the 6d pose using a quaternion network, and learns a quaternion representation of the object representation. posecnn @cite proposes a "learns approach for estimating 6d object pixels in an image using quaternion representation. posecnn and posecnn @cite propose an approach approach to regressing the object pixels from a particular object representation. however, these methods require a large amount of training data, which is difficult to obtain. in contrast, our method does not rely on a large number of objects in the scene.

KGSum:
6d pose estimation has been widely used in computer vision @cite @cite . in @cite , the authors propose a cnn for 6d object pose estimation. the first step is to use a quaternion to handle the 6d pose of an object in a single image. the second stage is based on the idea of using a convolutional neural networks to learn a 3d rotation of the object and the appearance of a 6d pose. the proposed method is also used for object segmentation @cite and pose estimation, which is used to refine the geometry of object proposals. @cite used an encoder-decoder network to handle the 6d pose of known objects in a rgb-d image by regressing the 3d rotation of an object with a 3d representation.

Figure 5: Case study from the Multi-Xscience test set. We first highlight the salient contents in the reference summary in different colors. Then the overlapped salient contents of system summaries are annotated in the same colors.

Generation of Patient After-Visit Summaries to Support Physicians

Pengshan Cai,[1] **Fei Liu,**[2] **Adarsha Bajracharya,**[3] **Joe Sills,**[3] **Alok Kapoor,**[3]
Weisong Liu,[4] **Dan Berlowitz,**[4] **David Levy,**[4] **Richeek Pradhan,**[5] **Hong Yu**[1,3,4]
[1]University of Massachusetts, Amherst [2]Emory University
[3]UMass Chan Medical School [4]University of Massachusetts, Lowell [5]McGill University
{pengshancai, hongyu}@cs.umass.edu fei.liu@emory.edu
{weisong_liu, dan_berlowitz, david_levy}@uml.edu
{adarsha.Bajracharya, alok.kapoor}@umassmemorial.org
joe.sills@baystatehealth.org richeekp@gmail.com

Abstract

An after-visit summary (AVS) is a summary note given to patients after their clinical visit. It recaps what happened during their clinical visit and guides patients' disease self-management. Studies have shown that a majority of patients found after-visit summaries useful. However, many physicians face excessive workloads and do not have time to write clear and informative summaries. In this paper, we study the problem of automatic generation of after-visit summaries and examine whether those summaries can convey the gist of clinical visits. We report our findings on a new clinical dataset that contains a large number of electronic health record (EHR) notes and their associated summaries. Our results suggest that generation of lay language after-visit summaries remains a challenging task. Crucially, we introduce a feedback mechanism that alerts physicians when an automatic summary fails to capture the important details of the clinical notes or when it contains hallucinated facts that are potentially detrimental to the summary quality. Automatic and human evaluation demonstrates the effectiveness of our approach in providing writing feedback and supporting physicians.[1]

1 Introduction

Studies have shown that the majority of patients do not understand their clinical visits (O'Leary et al., 2010). After-visit summary note (AVS) is a summary given to patients after their clinical visit, it is intended to summarize patients' clinical visits and help their disease self-management (Federman et al., 2018). Compared to clinical notes, an after-visit summary has the following characteristics: 1) it is written in lay person language thus is easy for patients to read and comprehend; 2) it only contains information that patients should be aware of, leaving out redundant details, e.g. unimportant lab results, etc. Studies have shown that around 36% of American adults have limited health literacy (Kutner et al., 2006), and 94.4% of patients found that lay language after-visit summary helps them understand their clinical visits (Pathak et al., 2020). However, the implementation of after-visit summary is challenging. Many physicians face excessive workloads (West et al., 2018) and do not have time to complete the summaries in a timely manner (Hong et al., 2013). Thus, there is a real need for—and this study contributes to—automatic generation of after-visit summaries to unburdening physicians with complex information workflows.

We explore best-performing neural abstractive summarizers to generate after-visit summaries from EHR notes. The summaries are rated by physicians as concise and easy to read. However, they can not be presented directly to patients, as they frequently contain two types of errors: 1) *Missing content.* A summary often leaves out important details such as medication dosage and route, undermining patients' medical self-management. 2) *Hallucination.* Summaries contain hallucinated content or content not supported by the input documents. For example, an abstractive summary on *kidney infection* was generated from an input document that describes *urine infection*. These types of errors are not uncommon in abstractive summarization (Lebanoff et al., 2019; Maynez et al., 2020; Pagnoni et al., 2021), but they could be disastrous to patients.

In this study, we build systems to facilitate detection and correction of those types of errors, allowing physicians to correct or edit system generated summaries. As illustrated in Figure 1, *Summarization* produces a system summary; *Error Alerting* automatically detects errors from the generated after-visit summary. Crucially, we build effective detectors with self-supervision on unlabeled data for error alerting. A novel dataset is constructed by synthesizing summaries containing medical events that are inconsistent with their source documents.

[1]Our project page is available at: https://github.com/pengshancai/AVS_gen

Proceedings of the 29th International Conference on Computational Linguistics, pages 6234–6247
October 12–17, 2022.

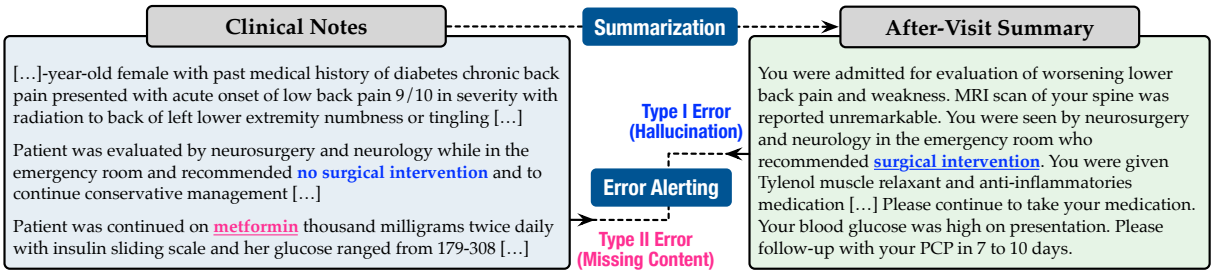

Figure 1: An example after-visit summary generated from EHR notes associated with a patient. A novel alerting mechanism is proposed in this work to report errors found in the summary, including missing medical events and hallucinated facts. We aim to build effective detectors with self-supervision on unlabeled data for error alerting.

Using this simulated dataset, we train a hallucination detection model, which alerts physicians of potential hallucination content. Further, by aligning medical events in EHR notes to those in after-visit summaries using MetaMap (Aronson, 2001), we identify key events important to patients, and alert physicians of salient medical events not covered in the generated summaries as missing content. The contributions of our research are as follows:

- We propose a new task that generates lay language AVS from EHR notes, build and evaluate state-of-the-art NLP models for this task. A novel alerting mechanism is proposed to report errors, including missing medical events and hallucinations. The training of our error detectors is self-supervised, using only unlabelled text.
- Clinical applications demand high performance. Existing automatic metrics are not adequate for evaluating the quality of generated AVS. Therefore, we conduct a qualitative assessment of system outputs with medical practitioners. Our findings show that the alerting mechanism could provide a promising avenue towards making the writing process easier for physicians.

2 Related Work

Recently there has been a lot of work on automatic summarization in the clinical domain: Zhang et al. (2018) propose to generate the impression section of a radiology report using seq2seq models. Miura et al. (2021) perform image-to-text radiology report generation by optimizing entity-based rewards with reinforcement learning. Studies are also performed for summarizing doctor-patient dialogues (Joshi et al., 2020; Krishna et al., 2021) and evaluating system generated notes (Moramarco et al., 2022).

Early work has explored generation of hospital visit summaries using non-neural methods (Di Eugenio et al., 2014; Hirsch et al., 2015; Acharya et al., 2018). Recently, Adams et al. (2021) present the task of hospital-course summarization with the goal of generating a text to synthesize the hospital course. A crucial difference between our work and that of Adams et al. (2021) is we investigate a deep learning solution, whose primary focus is exposing neural abstractive summarizers to clinical notes and explicitly highlighting regions of a summary which need attention. This is in principle similar to Checklist in (Ribeiro et al., 2020).

It is important for an after-visit summary generated from EHR notes to avoid type I and type II errors. A type I error (false positive) suggests that there is false or inaccurate information in the summary, due to hallucinations, incorrect grounding, *etc*. It is a challenging and lingering problem facing natural language generation (Falke et al., 2019; Lebanoff et al., 2020; Kryscinski et al., 2020; Matsumaru et al., 2020; Pagnoni et al., 2021; van Miltenburg et al., 2021), despite remarkable recent progress (Gehrmann et al., 2018; Liu and Lapata, 2019; Fabbri et al., 2019; Zhong et al., 2020; Lewis et al., 2020; Ni et al., 2021, *inter alia*).

A more surprising observation is that the type II error (false negative) is deemed particularly harmful to patients. When salient medical events such as diagnoses or treatments are left out of the after-visit summaries, it could have a detrimental effect on patients' self-care after being discharged from hospitals (Raghavan et al., 2012; Sotudeh Gharebagh et al., 2020). This empirically motivates our work, where we seek to effectively identify salient medical events in EHR notes and alert physicians of any missing events to help them avoid those errors.

A distinguishing characteristic of after-visit summaries is that they are *patient-oriented*. The summaries provide relevant and actionable information to patients, such as reasons for visit, diagnoses and procedures, *etc*. Differing from *physician-oriented*

Abstractive Summarization Model	Extractive Summarization Model
• **BART** (Lewis et al., 2020) uses the standard encoder-decoder architecture. It was pretrained as a denoising autoencoder to learn to reconstruct the original text. Our input to the BART model consists of a clinical document and its output is an abstractive summary. • **PEGASUS** (Zhang et al., 2020a) explores a new pretraining objective tailored for abstractive summarization. Important sentences are masked out from the input document and the model learns to generate the sentences as an output sequence, akin to an extractive model. The system has been shown to perform well in a low-resource scenario where few examples are available for fine-tuning. • **LED** (Beltagy et al., 2020) is the Longformer-Encoder-Decoder model. It is an extension to Longformer to support text generation. LED uses a local windowed attention which makes it computational feasible to encode a long input document. We favor the LED model because, compared to news articles, there is more risk involved in truncating long clinical documents to a certain length.	• **BertSum** (Liu and Lapata, 2019) employs the BERT model to identify summary-worthy sentences. It uses a flat architecture to encode the input document, then adds a Transformer layer on top of the sentence representations to model inter-sentence relationship. The final output layer is a sigmoid classifier used to predict if the sentence is to be included in the summary. • **TextRank** (Mihalcea and Tarau, 2004) and **LexRank** (Erkan and Radev, 2004) are graph-based models that extract relevant sentences based on eigenvector centrality. • **Oracle Top-K** (Adams et al., 2021) is a method introduced by Adams et al. which represents the upper bound for sentence extraction. It ranks all document sentences according to their averaged R-1 and R-2 scores with respect to the reference summary. It then continues to add sentences yielding the highest scores to the summary until the target token count is reached.

Table 1: State-of-the-art summarization models investigated in this work for generation of patient after-visit summaries.

clinical notes, these summaries are written in an easy-to-understand language, and they remain understudied in NLP. Existing research on text simplification focuses primarily on Wikipedia and news articles (Zhu et al., 2010; Xu et al., 2015; Vu et al., 2018; Kriz et al., 2019; Dong et al., 2019; Kriz et al., 2020). Chandrasekaran et al. (2020) propose to generate lay summaries to describe scientific papers for non-experts. In a similar fashion, after-visit summaries are intended for a lay audience: translating sophisticated medical events into plain language that is understandable by patients.

In what follows, we investigate generation of patient after-visit summaries, closely examine the language used by clinicians, and develop automatic methods to spot errors in summaries to better support clinicians with this challenging task.

3 Summarization

Our method generates an after-visit summary from EHR notes concerning a patient. It is modelled as a single-document summarization task as the EHR notes were collapsed into a single document by the hospital and we were unable to recover individual EHR notes. We use $\mathcal{S}$=$\{w_1, ..., w_{n_\mathcal{S}}\}$ to denote tokens of the source document and $\mathcal{T}$=$\{w_1, ..., w_{n_\mathcal{T}}\}$ tokens of the target summary, $n_\mathcal{S}$ and $n_\mathcal{T}$ are length of the sequences.

We explore a variety of summarization models to generate after-visit summaries. They are detailed in Table 1. Particularly, an abstractive summarizer employs the standard Transformer-based encoder-decoder model to generate a summary $P(\mathcal{T}|\mathcal{S})$. An extractive summarizer selects important sentences to add to the summary until a length threshold has been reached. These systems are used off-the-shelf and have achieved some of the highest reported scores on summarization. We assess their ability to navigate complex medical terrain for generation of after-visit summaries.

Entity Type	Event Type
Anatomical Abnormality	Diagnostic Procedure
Medical Device	Therapeutic or Preventive Procedure
Clinical Drug	Pathologic Function
Pharmacologic Substance	Disease or Syndrome
Organic Chemical	Mental or Behavioral Dysfunction
Body Substance	Injury or Poisoning
Finding	
Sign or Symptom	

Table 2: Semantic types used in this study.

Clinical notes are complex and full of references to medical events. However, the summary given to the patient is simple and clear. We are thus curious to know how medical events manifest themselves in the context of summarization. Events are especially important for this task, as salient events happening at each medical encounter must be included in the after-visit summary.

We define *event nugget* as a word or multi-word phrase that clearly expresses the occurrence of a medical event. Event nuggets are identified by MetaMap (Aronson, 2001), an open-source software tool designed to discover medical concepts referred to in a text. Each occurrence of the concept is assigned a concept unique identifier (CUI) and its associated words are tagged in the text. In Figure 3, we show an example of medical concepts identified by MetaMap. Further, those medical concepts are categorized into various semantic types. We focus on concepts pertaining to a selected set of entity and event types (Table 2), which are deemed relevant by medical experts. The other types are excluded from consideration.

4 Error Detection and Alerting

Event nuggets are associated with type I and type II errors frequently found in the summary. A *type I error* indicates a summary contains a hallucinated fact or event that is not present in the source document. A *type II error* suggests that an important medical event has been mistakenly left out of the summary, hampering its usability. In this section, we describe novel methods to detect likely errors and flag them in the text to alert clinicians.

4.1 Type I Error: Hallucination

A *hallucination detector* aims to recognize any hallucinated content in a summary. Flagging errors is helpful because physicians can be alerted about any anomalies and it is especially appreciated in medical domain (Singh et al., 2014). Our detector uses the BigBird model (Zaheer et al., 2020), which is an encoder-only architecture equipped with sparse attention to reduce Transformer's quadratic complexity to linear, and capable of encoding thousands of tokens. The model takes as input a source document ($\mathcal{S}$) and its system summary ($\mathcal{T}$), and outputs a sequence of binary labels, one for each summary token, where 1 represents the token is considered hallucinated and 0 otherwise.

A key factor to the success of our model is its self-supervised training, where a large number of training instances are constructed from unlabeled data. Each training instance is a *synthesized summary* whose hallucinated tokens are flagged. We adapt the model of Zhou et al. (2021), initially proposed for MT, to create our training instances. Our method differs from theirs in that, synthesized summaries are required to contain hallucinated medical events that are inconsistent with or unjustified by the source document.

Synthesizing Erroneous Summaries. The procedure for generating synthesized summaries is illustrated in Figure 2. Given a summary sentence, we mask out one or two of its event nuggets. It is then fed to a denoising auto-encoder (Lewis et al., 2020) to produce an output sentence, whose masked-out positions are refilled with medical events that are "hallucinated" by the model. If the output is substantially different from the input, e.g., with <50% token overlap, it is called a *synthesized* sentence with hallucinations. Tokens of the synthesized sentence, which cannot be aligned to the original sentence using an edit-distance-style algorithm, are flagged. E.g., "*cardiac catheterization*" and "*abnormalities*" in our example are clearly hallucinated facts. This procedure is repeated for all sentences[2] of the reference summary to create a synthesized summary. We provide examples of synthesized sentences in the Supplementary.

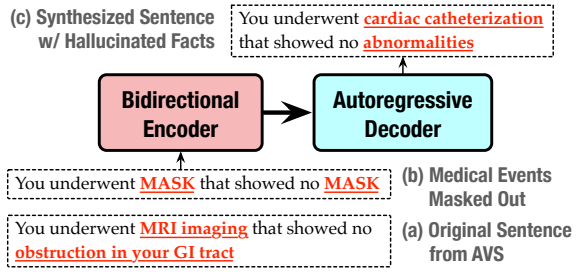

Figure 2: One or two event nuggets are randomly masked out from a summary sentence (a). The masked sequence (b) is fed to a denoising auto-encoder to produce a synthesized sentence that may contain hallucinated medical events (c).

Importantly, the model is fine-tuned to enable it to produce plausible synthesized summaries. We partition the training data into K folds (K=5) of roughly equal size. The BART model is fine-tuned on the union of the K-1 folds, then applied to the remaining fold to generate synthesized summaries. The method transforms each reference summary of the dataset to a synthesize summary, which together with the source document, is used to train and test our type I error detector.

4.2 Type II Error: Missing Content

Our *missing content detector* seeks to accomplish two objectives: 1) to detect *salient medical events* on a clinical document, and 2) to flag salient events that are *missed* by the summary. It is a non-trivial task to fulfill these objectives. Even though clinical notes are full of references to medical events, only a selective portion of them (≈18%) are included in after-visit summaries. As such, we formulate the problem as a classification task. An *event nugget* is assigned a label of 1 if it is salient, 0 otherwise. Our detector leverages self-supervised learning to identify salient events on EHR notes. It then alerts clinicians if the summary fails to include any of the salient events.

Pseudo-Annotations for Salient Events. We create pseudo-annotations for salient events by aligning each source event with one of the target events. As shown in Figure 3, the medial events are identified by MetaMap (Aronson, 2001). Each

[2]If a summary sentence does not contain any medical event, it is left as-is in the synthesized summary. The original summary sentence is otherwise replaced by a synthesized sentence.

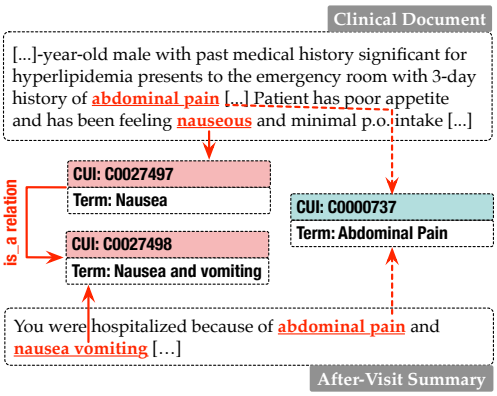

Figure 3: "*abdominal pain*" appears in both the clinical document and after-visit summary, with the same CUI. "*nausea vomiting*" and "*nauseous*" are aligned because there is an *is-a* relation between the two concepts.

occurrence of the event is associated with a concept unique identifier (CUI). Under the *strict matching* criterion, an event of the clinical document is labeled as 1 if an exact match (with the same CUI) is found in the summary. However, a large number of events are not well-aligned under this criterion due to distinct expressions used in clinical notes and summaries. This discrepancy in language use has its origin—clinical notes are physician-oriented, whereas after-visit summaries are patient-oriented. We explore *lenient matching* to alleviate mismatch. If a source event can reach any of the target event via *a single hop* on the UMLS semantic graph,[3] the source event is labeled as salient. In Figure 3, source event "*nauseous*" is leniently matched to target event "*nausea vomiting*," because there exists an "*is_a relation*" between the two events.

We fine-tune the BigBird model (Zaheer et al., 2020) to detect salient events. Being an encoder-only model, BigBird constructs contextualized representations for all tokens of a clinical document. It does not directly produce event representations. To address this issue, we let the model predict salient tokens during training. If a token is part of a salient source event, its gold-standard label is 1. At test time, the model generates token-level predictions. A source event is considered salient if any of its tokens is labeled as 1.

We explore two variants of the model to allow it to better capture events. Both variants aim to inform the model about the occurrences of event nuggets identified by MetaMap. The first variant, +POS, modifies the source sequence by inserting special tokens respectively at the beginning and end

[3] https://www.nlm.nih.gov/research/umls/META3_current_relations.html

BASE +POS	[E] **CT scan** [/E] showed worsening of his [E] **diverticulitis** [/E] with a 5.6 x 3.9cm multiloculated fluid collection in his abdomen.
BASE +TYPE	[Type1] **CT scan** [/Type1] showed worsening of his [Type2] **diverticulitis** [/Type2] with a 5.6 x 3.9cm multiloculated fluid collection in his abdomen.

Table 3: Model variants +POS and +TYPE aim to inform the model about the occurrences of events identified by MetaMap.

of a candidate event. The second variant, +TYPE, inserts different special tokens such that they correspond to the semantic types of the events. We conjecture that certain event types, e.g., *body substance*, are more likely be considered insignificant. In Table 3, we provide examples comparing the source sequences used by model variants.

5 Experiments

In this section, we describe our dataset, perform in-depth analyses on our models, and discuss feedback from physicians who participated in our qualitative evaluation.[4]

5.1 Dataset

Through a collaboration with University of Massachusetts Chan Medical School, we are able to use their electronic health record database, which gives us access to 31,895 EHR notes and their physician-written summaries. All medical records are de-identified to protect patient privacy. These patients were admitted to the medical and surgical services of the hospital from October 2017 to March 2020.

Table 4 summarizes the statistics of our dataset. It is divided into train, validation, and test sets containing 28,157, 1,884 and 1,854 instances, respectively. The dataset has unique characteristics. We observe that the source documents are substantially longer and contain more medical events than their summaries. This is because most hospitalization details are omitted for patients. In addition, the length of clinical documents varies considerably, so is the case for summaries. A long clinical document could be the result of an extended hospital stay. An after-visit summary could be long or short depending on the patient's medical conditions. In contrast, variation in length is less significant in other genres such as news and scientific articles.

We find that an average summary contains 12.3 medical events, yet only 7.9 of them can be linked

[4] Implementation details, including hidden state sizes, computational infrastructure used, hyperparameter configurations, etc. are provided in the Supplementary for reproducibility.

Train / Validation / Test Split		28,157 / 1,884 / 1,854
Number of words per clinical document		523.6 ± 464.3
Number of words per after-visit summary		153.5 ± 166.7
Event Nuggets	per clinical document	42.8 ± 32.0
	per after-visit summary	12.3 ± 9.0
	occurring in both (*lenient match*)	7.9 ± 6.1
	occurring in both (*strict match*)	4.0 ± 3.9

Table 4: Statistics of our dataset.

Adequacy	
3	AVS contains all the information the patient needs to know
2	AVS misses some (1-3) points the patient needs to know
1	AVS misses more than 3 points
Faithfulness	
3	AVS contains no or only a few errors that are ignorable
2	AVS contains some (1-3) factual errors
1	AVS contains more than 3 factual errors
Readability	
3	AVS is easy to read for a lay person
2	AVS has some (1-3) points hard to be understood by the patient
1	AVS has more than 3 points hard to be understood by the patient
Ease of Revision	
3	Physician may spend <=2 minutes to revise the AVS
2	Physician may spend >2 minutes to revise the AVS
1	Physician prefers to not revise the AVS but rewrite from scratch

Table 5: Instructions provided to physicians. The scoring scale for summary evaluation is from 1 (worst) to 3 (best).

to events of the clinical document. The gap is partially due to using MetaMap for medical event identification (Reátegui and Ratté, 2018), which has a reported F-Score of 0.88 and may miss out-of-vocabulary event tokens. Additionally, physicians may add their instructions directly to patient's after-visit summaries, and such content is not grounded in clinical documents.

5.2 Evaluation Metrics

Quantitative Measures. We evaluate the performance of our summarization and error alert models with a variety of quantitative measures.

- **ROUGE** (Lin, 2004) is the standard measure for summarization evaluation. It assigns a high score to a system summary if it has lexical overlap with the reference summary.
- **BERTScore** (Zhang et al., 2020b) is one of the new evaluation metrics for natural language generation that are built on contextualized representations produced by BERT and similar models.
- **SARI** (Xu et al., 2016) is widely used for simplification. It counts how often a system summary correctly keeps, deletes, and adds n-grams.
- **DaleChall** (Dale and Chall, 1948) calculates the readability of the summary based on its sentence length and number of difficult words in it. It is an improvement upon Flesch's reading ease score.
- **P/R/F** scores are reported for error alert models on successful detection of missing medical events and detection of hallucinated summary tokens.

Qualitative Measures. In high-stake scenarios, automatic metrics alone cannot guarantee a good system. Thus, we need expert assessments by medical practitioners in this study. We recruit six human evaluators: five of them are physicians with M.D., one is a M.D. student. Owing to budget constraints, we select a random set of 18 clinical documents and their best system summaries for qualitative assessment. The system summaries are produced by the LED model, they are abstractive. Each summary is judged by two human evaluators, who perform two tasks on a summary:

- **Scoring.** A summary is rated along four dimensions. *Adequacy*: Does the summary contain all necessary information for the patient to know? *Faithfulness*: Does the summary faithfully convey the content of the clinical document? *Readability*: Is the summary easy to read for a lay person? *Ease of Revision*: How long might it take for a physician to revise the summary to meet the expectations of standard AVS? The scoring scale is from 1 (worst) to 3 (best). Their interpretations are provided in Table 5.
- **Revision.** We ask human evaluators to edit the summary until it meets the expectations of standard after-visit summaries. We report the edit distance between the original and edited summaries, the amount of editing applied to the raw system summary is a good indicator of its utility (Snover et al., 2006).

For alert evaluation, we ask the evaluators to first label missing medical events on the clinical document, and hallucinations on the system summary. The evaluators are then given the alerts produced by our models, and they proceed to judging the correctness of each alert. This allows us to report precision, recall and F1 scores of our error alert models with human judgment.

5.3 Summarization Results

Quantitative. Table 6 provides a quantitative evaluation of after-visit summaries produced by state-of-the-art models. Our aim in this work is not to present new methods, but rather to thoroughly evaluate state-of-the-art models on this challenging

	Model	R-1	R-2	R-3	R-4	R-L	BertS	SARI	DaleC.↓	Length
EXT	TEXTRANK (Mihalcea and Tarau, 2004)	25.71	7.36	3.92	2.64	13.83	54.37	34.33	12.56	150.01
	LEXRANK (Erkan and Radev, 2004)	25.57	7.26	4.01	2.71	12.81	54.31	34.91	12.38	153.21
	BERTSUM (Liu and Lapata, 2019)	26.22	7.42	4.43	2.90	14.56	55.62	35.57	11.21	149.73
	ORACLE (Adams et al., 2021)	36.84	13.55	6.86	4.45	19.47	58.50	39.74	11.07	99.61
ABS	BART (Lewis et al., 2020)	41.67	21.05	14.20	10.80	30.20	62.80	44.36	9.97	144.29
	PEGASUS (Zhang et al., 2020a)	37.02	19.68	14.02	10.93	28.44	60.91	41.89	10.53	134.26
	LED (Beltagy et al., 2020)	**41.96**	**21.80**†	**15.01**†	**11.58**†	**31.49**†	**63.31**†	**45.06**	**9.58**	148.03

Table 6: Quantitative evaluation of patient after-visit summaries produced by state-of-the-art summarization models. LED shows best performance among all tested abstractive models. It significantly outperforms all other systems for all metrics (p<0.05), with the exception of BART in terms of R-1, according to a non-parametric Wilcoxon signed rank test.

task to identify areas for improvement. We observe that BERTSUM achieves the highest scores among all extractive models. Further gain is provided by an *oracle* model developed by Adams et al. (2021) that improves R-2 F-score from 7.42% to 13.55% by greedily extracting sentences yielding highest similarity scores with the reference summary. The method gives an upper bound on ROUGE scores obtainable by an extractive model.

We find all abstractive models to perform substantially better than their extractive counterparts. LED has shown best performance among all tested abstractive models, possibly due to its exceptional ability to encode long documents. With regards to evaluation metrics, we include less commonly used R-3 and R-4 F-scores, as they have been shown to correlate better with human judgment than other variants (Graham, 2015; Kryscinski et al., 2019). Our results suggest that generation of patient after-visit summaries is highly abstractive. For this reason, an abstractive model would suit our task best. Extractive summaries are verbose and they may potentially overwhelm patients with unnecessary detail.

Expert Scoring. Two medical experts are asked to rate each summary produced by our best abstractive model (LED) along the dimensions of *adequacy*, *faithfulness*, *readability* and *ease of revision*. Their ratings are averaged for each summary [5] and results are presented in Figure 4. All summaries are divided into five bins, their average ratings are 1/1.5/2/2.5/3, respectively. We observe that generating adequate summaries remains a challenge for the abstractive model. Only 5.5% of the summaries obtain a full score (3 points). Per our physicians, the remaining summaries have, to a varying degree, missed important medical events that patients need to know. Our findings suggest that future studies should incorporate expert knowledge in selecting medical events to add to the summary.

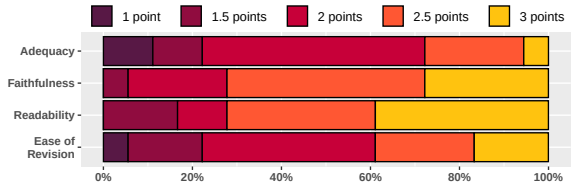

Figure 4: Summaries are rated by medical practitioners along the dimensions of *adequacy*, *faithfulness*, *readability* and *ease of revision*. Their ratings are averaged for each summary.

Efforts could be made to also improve the readability and understandability of abstractive summaries. We observe that 38.8% of the summaries obtain a full score on readability. A closer analysis reveals that a portion of the summaries contain abbreviated medical terminology or jargon that are familiar to physicians but may be difficult for non-experts. E.g., in "*minimal PO intake*," PO is from the Latin "*per os*" and means "*by mouth*." The summaries are also believed to have less hallucination issues when comparing to missing medical events. 72.2% of the summaries obtain 2.5 points or higher. Further, >75% of the summaries receive an average score of 2.5 or higher on ease-of-revision. The results indicate that, physicians may be guided to revise system-produced summaries to meet the standards of medical practice, as opposed to starting from scratch.

Expert Revision. Table 8 shows a direct comparison of summaries before and after expert revision (more examples are in the supplementary). Our physicians have revised 43.5 words on average for each summary, corresponding to 47.2% of the summary length. Even though there is still room for improvement, the results are positive. For 4 out of 18 cases, physicians only minimally revised the summaries, with less than 15% of the

[5] We provide inter-annotator analysis among physicians in the supplementary materials.

	Model	P(%)	R(%)	F_1(%)
Type I	Baseline-RAND	6.52	3.22	3.82
	Baseline-MOSTFREQ5	17.80	46.04	21.79
	Baseline-MOSTFREQ10	20.86	76.31	29.19
	H-Alert (Ours)	**44.96**	**71.66**	**55.25**
Type II	Baseline-RAND	3.99	13.56	6.06
	Baseline-MOSTFREQ5	9.74	34.72	13.70
	Baseline-MOSTFREQ10	9.71	49.14	15.04
	M-Alert (Ours)	49.22	43.65	41.71
	M-Alert +POS (Ours)	**51.03**	45.98	43.80
	M-Alert +TYPE (Ours)	50.69	**49.88**	**45.51**

Table 7: Automatic evaluation of our hallucination detector (**H-Alert**) and missing event detector (**M-Alert**). Both detectors strongly outperform their baselines.

words edited. For 3 out of 18 cases, the summaries are nearly rewritten, where 90% of the words are edited. The results suggest that certain noisy clinical documents can cause disastrous summaries. It is crucial for summarizers to degrade gracefully as noise increases.

5.4 Error Detection Results

Our detectors are evaluated using both automatic metrics and human judgment. Results are reported in Table 7. **H-Alert** is our hallucination detector. It is evaluated on the test set with synthesized hallucinations (§4.1). Baseline-RAND samples a label for each summary token from a Bernoulli distribution $t_j \sim \text{Bernoulli}(p)$. Here, p is the probability that an average summary token is hallucinated, computed on training data. MOSTFREQ5 and MOSTFREQ10 examine the semantic types of events (Table 2). If an event type is frequently hallucinated, all of its tokens are labeled as 1. As seen in the table, we find our H-Alert can not only outperform the baselines, but it obtains a high recall score (71.66%).

M-Alert is our missing event detector. It predicts source medical events that are missed by the summary. Baseline-RAND samples a label for each source event, $e_i \sim \text{Bernoulli}(q)$, where q is the probability an average event is missed, computed on training data. We find that M-Alert produces better precision scores than all baselines. The best performance is achieved by the model variant +TYPE, which injects event types to the BigBird model to help detection of missing events. We note that identifying key medical events remains a challenging task and graph neural networks may help model inter-event relations.

Expert P/R/F. On expert-annotated summaries, we report scores for both detectors. System alerts have been manually verified. The micro-averaged P/R/F scores for H-Alert is 17.24/**58.82**/26.66, and the scores for M-Alert is 30.65/**53.84**/39.06. These results are positive because both detectors are able to attain high recall scores, indicating errors could be effectively flagged and passed on to physicians for further review.

6 Discussion

We discuss our findings from interviewing physicians and underline some of the key areas that are indispensable for further progress on this task.

- ***Medical jargon.*** Owing to time constraints and the literacy of physicians who create the clinical notes, the data we received are of varying quality. It is not uncommon to find jargon or ambiguous information, e.g., "*Patient presents w/ < 24 hours abdominal pain nausea and non-bloody V/D*," here, "V/D" refers to "vomit and diarrhea."
- ***Style difference in clinical notes.*** The notes could be: 1) *procedure-oriented*, i.e., they are narratives describing medical procedures performed on the patient, including treatment, medication, care plans and etc. 2) *disease-oriented*, i.e., each of the patient's diseases is addressed in a separable section, or 3) *organ-oriented*, i.e., each organ is addressed in a separable section.
- ***Improper grounding.*** An after-visit summary states "*We did test you for the coronavirus which was negative*." However, the "*coronavirus test*" was nowhere to be found in the source document. Similar grounding issue was identified in 5 out of 18 summaries during expert revision. Sometimes physicians directly include their knowledge about the patients into after-visit summaries without referring to clinical notes, causing a summarizer fine-tuned on such data to also "hallucinate" content.
- ***High variance in length.*** It would be unwise to truncate clinical notes, despite that most neural models use a fixed maximum length. E.g., a patient who underwent a heart transplant has a high risk of multiple medical comorbidities. It can lead to a large volume of EHR notes. Interestingly, physicians tend to include *more* content in after-visit summaries if they believe patients have *high medical literacy* and are able to understand and act upon complex instructions. This indicates that future systems may produce summaries of varying length per patients' needs.

A System Generated Summary: You were admitted for dizziness. You had a CT scan of your head which showed some thickening in the sinuses of your sinuses. You were seen by the ear nose and throat doctor who recommended that you take an antibiotic called Unasyn while you are in the hospital. You also had an MRI of your brain which did not show any stroke. You are doing better and can go home today.
After Physician's Revision: You were admitted for dizziness. You had a CT scan of your head which showed some thickening in your sinuses and mastoid. This could be suggestive of an infection but your white cells and temperature were normal. You were seen by the ear nose and throat doctor who recommended that you take an antibiotic called Unasyn while you are in the hospital. You also had an MRI of your brain which did not show any stroke. You are doing better and can go home today.

Table 8: A direct comparison of summaries before and after physician revision. A post-study interview with physicians reveals that most revisions are related to missing key medical events (colored orange). They also spend substantial efforts explaining medical jargon to patients and fixing hallucinations (colored red).

7 Conclusion

We tackle the problem of generation of patient after-visit summaries. We compared state-of-the-art summarization models for this task and introduced a novel alerting mechanism to predict two types of errors, including missing medical events and hallucinations in summaries. Extensive experiments using automatic metrics and expert evaluation show the effectiveness of our proposed approach.

8 Ethical Considerations

Data. Data used in this study are obtained from a comprehensive inpatient medical facility. They are electronic dismissal notes created by physicians to record a patient's hospital stay or a series of treatments performed on a patient. These EHR notes are information-dense and full of technical terms. They need to be rewritten and summarized to generate after-visit summaries. The purpose of using patient medical records is to fine-tune abstractive summarization systems and quantitatively evaluate the truthfulness and adequacy of system summaries. These medical records are not for non-academic uses and intents. All medical records are deidentified by the hospital to protect patient privacy.

Summarization Models. Models for abstractive summarization have a tendency to hallucinate information that is not present in the input documents. This is because abstractive models carry inductive biases rooted in the data they are pretrained on. The data encode prior knowledge of natural language, they may also contain a non-negligible amount of toxic and abusive content. Despite our best efforts to alert clinicians of potential errors, some of them could be almost unnoticeable by non-physicians. We thus caution our users to carefully consider the ethical issues specific to abstractive summarization and natural language generation models.

Acknowledgement

Research reported in this publication was supported by the National Library of Medicine of the National Institutes of Health under award number NIH R01LM012817. The content is solely the responsibility of the authors and does not necessarily represent the official views of the National Institutes of Health. Fei Liu is supported in part by National Science Foundation grant IIS-1909603.

References

Sabita Acharya, Barbara Di Eugenio, Andrew Boyd, Richard Cameron, Karen Dunn Lopez, Pamela Martyn-Nemeth, Carolyn Dickens, and Amer Ardati. 2018. Towards generating personalized hospitalization summaries. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Student Research Workshop*, pages 74–82, New Orleans, Louisiana, USA. Association for Computational Linguistics.

Griffin Adams, Emily Alsentzer, Mert Ketenci, Jason Zucker, and Noémie Elhadad. 2021. What's in a summary? laying the groundwork for advances in hospital-course summarization. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 4794–4811, Online. Association for Computational Linguistics.

Alan R Aronson. 2001. Effective mapping of biomedical text to the umls metathesaurus: the metamap program. In *Proceedings of the AMIA Symposium*, page 17. American Medical Informatics Association.

Iz Beltagy, Matthew E Peters, and Arman Cohan. 2020. Longformer: The long-document transformer. *arXiv preprint arXiv:2004.05150*.

Muthu Kumar Chandrasekaran, Guy Feigenblat, Dayne Freitag, Tirthankar Ghosal, Eduard Hovy, Philipp Mayr, Michal Shmueli-Scheuer, and Anita de Waard. 2020. Overview of the first workshop on scholarly document processing (SDP). In *Proceedings of the First Workshop on Scholarly Document Processing*, pages 1–6, Online. Association for Computational Linguistics.

Edgar Dale and Jeanne S Chall. 1948. A formula for predicting readability: Instructions. *Educational research bulletin*, pages 37–54.

Barbara Di Eugenio, Andrew Boyd, Camillo Lugaresi, Abhinaya Balasubramanian, Gail Keenan, Mike Burton, Tamara Goncalves Rezende Macieira, Jianrong Li, Yves Lussier, and Yves Lussier. 2014. Patient-Narr: Towards generating patient-centric summaries of hospital stays. In *Proceedings of the 8th International Natural Language Generation Conference (INLG)*, pages 6–10, Philadelphia, Pennsylvania, U.S.A. Association for Computational Linguistics.

Yue Dong, Zichao Li, Mehdi Rezagholizadeh, and Jackie Chi Kit Cheung. 2019. EditNTS: An neural programmer-interpreter model for sentence simplification through explicit editing. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 3393–3402, Florence, Italy. Association for Computational Linguistics.

Günes Erkan and Dragomir R Radev. 2004. Lexrank: Graph-based lexical centrality as salience in text summarization. *Journal of artificial intelligence research*, 22:457–479.

Alexander Fabbri, Irene Li, Tianwei She, Suyi Li, and Dragomir Radev. 2019. Multi-news: A large-scale multi-document summarization dataset and abstractive hierarchical model. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 1074–1084, Florence, Italy. Association for Computational Linguistics.

Tobias Falke, Leonardo F. R. Ribeiro, Prasetya Ajie Utama, Ido Dagan, and Iryna Gurevych. 2019. Ranking generated summaries by correctness: An interesting but challenging application for natural language inference. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 2214–2220, Florence, Italy. Association for Computational Linguistics.

Alex Federman, Erin Sarzynski, Cindy Brach, Paul Francaviglia, Jessica Jacques, Lina Jandorf, Angela Sanchez Munoz, Michael Wolf, and Joseph Kanry. 2018. Challenges optimizing the after visit summary. *International journal of medical informatics*, 120:14–19.

Jemima A Frimpong, Christopher G Myers, Kathleen M Sutcliffe, and Yemeng Lu-Myers. 2017. When health care providers look at problems from multiple perspectives, patients benefit. *Harv Bus Rev. June*, 23.

Sebastian Gehrmann, Yuntian Deng, and Alexander Rush. 2018. Bottom-up abstractive summarization. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 4098–4109, Brussels, Belgium. Association for Computational Linguistics.

Yvette Graham. 2015. Re-evaluating automatic summarization with BLEU and 192 shades of ROUGE. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 128–137, Lisbon, Portugal. Association for Computational Linguistics.

Jamie S Hirsch, Jessica S Tanenbaum, Sharon Lipsky Gorman, Connie Liu, Eric Schmitz, Dritan Hashorva, Artem Ervits, David Vawdrey, Marc Sturm, and Noémie Elhadad. 2015. Harvest, a longitudinal patient record summarizer. *Journal of the American Medical Informatics Association*, 22(2):263–274.

Judith Hong, Tien V Nguyen, and Neil S Prose. 2013. Compassionate care: Enhancing physician–patient communication and education in dermatology: Part ii: Patient education. *Journal of the American Academy of Dermatology*, 68(3):364–e1.

Anirudh Joshi, Namit Katariya, Xavier Amatriain, and Anitha Kannan. 2020. Dr. summarize: Global summarization of medical dialogue by exploiting local structures. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 3755–3763, Online. Association for Computational Linguistics.

Kundan Krishna, Sopan Khosla, Jeffrey Bigham, and Zachary C. Lipton. 2021. Generating SOAP notes from doctor-patient conversations using modular summarization techniques. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 4958–4972, Online. Association for Computational Linguistics.

Reno Kriz, Marianna Apidianaki, and Chris Callison-Burch. 2020. Simple-qe: Better automatic quality estimation for text simplification. *arXiv preprint arXiv:2012.12382*.

Reno Kriz, João Sedoc, Marianna Apidianaki, Carolina Zheng, Gaurav Kumar, Eleni Miltsakaki, and Chris Callison-Burch. 2019. Complexity-weighted loss and diverse reranking for sentence simplification. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 3137–3147, Minneapolis, Minnesota. Association for Computational Linguistics.

Wojciech Kryscinski, Nitish Shirish Keskar, Bryan McCann, Caiming Xiong, and Richard Socher. 2019. Neural text summarization: A critical evaluation. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 540–551, Hong Kong, China. Association for Computational Linguistics.

Wojciech Kryscinski, Bryan McCann, Caiming Xiong, and Richard Socher. 2020. Evaluating the factual consistency of abstractive text summarization. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 9332–9346, Online. Association for Computational Linguistics.

Mark Kutner, Elizabeth Greenburg, Ying Jin, and Christine Paulsen. 2006. The health literacy of america's adults: Results from the 2003 national assessment of adult literacy. nces 2006-483. *National Center for Education Statistics*.

Logan Lebanoff, John Muchovej, Franck Dernoncourt, Doo Soon Kim, Seokhwan Kim, Walter Chang, and Fei Liu. 2019. Analyzing sentence fusion in abstractive summarization. In *Proceedings of the 2nd Workshop on New Frontiers in Summarization*, pages 104–110, Hong Kong, China. Association for Computational Linguistics.

Logan Lebanoff, John Muchovej, Franck Dernoncourt, Doo Soon Kim, Lidan Wang, Walter Chang, and Fei Liu. 2020. Understanding points of correspondence between sentences for abstractive summarization. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics: Student Research Workshop*, pages 191–198, Online. Association for Computational Linguistics.

Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Veselin Stoyanov, and Luke Zettlemoyer. 2020. BART: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7871–7880, Online. Association for Computational Linguistics.

Chin-Yew Lin. 2004. ROUGE: A package for automatic evaluation of summaries. In *Text Summarization Branches Out*, pages 74–81, Barcelona, Spain. Association for Computational Linguistics.

Yang Liu and Mirella Lapata. 2019. Text summarization with pretrained encoders. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 3730–3740, Hong Kong, China. Association for Computational Linguistics.

Kazuki Matsumaru, Sho Takase, and Naoaki Okazaki. 2020. Improving truthfulness of headline generation. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 1335–1346, Online. Association for Computational Linguistics.

Joshua Maynez, Shashi Narayan, Bernd Bohnet, and Ryan McDonald. 2020. On faithfulness and factuality in abstractive summarization. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 1906–1919, Online. Association for Computational Linguistics.

Rada Mihalcea and Paul Tarau. 2004. TextRank: Bringing order into text. In *Proceedings of the 2004 Conference on Empirical Methods in Natural Language Processing*, pages 404–411, Barcelona, Spain. Association for Computational Linguistics.

Yasuhide Miura, Yuhao Zhang, Emily Tsai, Curtis Langlotz, and Dan Jurafsky. 2021. Improving factual completeness and consistency of image-to-text radiology report generation. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 5288–5304, Online. Association for Computational Linguistics.

Francesco Moramarco, Alex Papadopoulos Korfiatis, Mark Perera, Damir Juric, Jack Flann, Ehud Reiter, Anya Belz, and Aleksandar Savkov. 2022. Human evaluation and correlation with automatic metrics in consultation note generation.

Ansong Ni, Zhangir Azerbayev, Mutethia Mutuma, Troy Feng, Yusen Zhang, Tao Yu, Ahmed Hassan Awadallah, and Dragomir Radev. 2021. Summertime: Text summarization toolkit for non-experts.

Kevin J O'Leary, Nita Kulkarni, Matthew P Landler, Jiyeon Jeon, Katherine J Hahn, Katherine M Englert, and Mark V Williams. 2010. Hospitalized patients' understanding of their plan of care. In *Mayo Clinic Proceedings*, volume 85, pages 47–52. Elsevier.

Artidoro Pagnoni, Vidhisha Balachandran, and Yulia Tsvetkov. 2021. Understanding factuality in abstractive summarization with FRANK: A benchmark for factuality metrics. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 4812–4829, Online. Association for Computational Linguistics.

Sarita Pathak, Gregory Summerville, Celia P Kaplan, Sarah S Nouri, and Leah S Karliner. 2020. Patient-reported use of the after visit summary in a primary care internal medicine practice. *Journal of Patient Experience*, 7(5):703–707.

Preethi Raghavan, Eric Fosler-Lussier, and Albert Lai. 2012. Temporal classification of medical events. In *BioNLP: Proceedings of the 2012 Workshop on Biomedical Natural Language Processing*, pages 29–37, Montréal, Canada. Association for Computational Linguistics.

Ruth Reátegui and Sylvie Ratté. 2018. Comparison of metamap and ctakes for entity extraction in clinical notes. *BMC Medical Informatics and Decision Making*, 18(3):74.

Marco Tulio Ribeiro, Tongshuang Wu, Carlos Guestrin, and Sameer Singh. 2020. Beyond accuracy: Behavioral testing of NLP models with CheckList. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 4902–4912, Online. Association for Computational Linguistics.

H. Singh, A. N. Meyer, and E. J. Thomas. 2014. The frequency of diagnostic errors in outpatient care: estimations from three large observational studies involving US adult populations. *BMJ Qual Saf*, 23(9):727–731.

Matthew Snover, Bonnie Dorr, Rich Schwartz, Linnea Micciulla, and John Makhoul. 2006. A study of translation edit rate with targeted human annotation. In *Proceedings of the 7th Conference of the Association for Machine Translation in the Americas: Technical Papers*, pages 223–231, Cambridge, Massachusetts, USA. Association for Machine Translation in the Americas.

Sajad Sotudeh Gharebagh, Nazli Goharian, and Ross Filice. 2020. Attend to medical ontologies: Content selection for clinical abstractive summarization. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 1899–1905, Online. Association for Computational Linguistics.

Emiel van Miltenburg, Miruna Clinciu, Ondřej Dušek, Dimitra Gkatzia, Stephanie Inglis, Leo Leppänen, Saad Mahamood, Emma Manning, Stephanie Schoch, Craig Thomson, and Luou Wen. 2021. Underreporting of errors in NLG output, and what to do about it. In *Proceedings of the 14th International Conference on Natural Language Generation*, pages 140–153, Aberdeen, Scotland, UK. Association for Computational Linguistics.

Tu Vu, Baotian Hu, Tsendsuren Munkhdalai, and Hong Yu. 2018. Sentence simplification with memory-augmented neural networks. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 2 (Short Papers)*, pages 79–85, New Orleans, Louisiana. Association for Computational Linguistics.

Colin P West, Liselotte N Dyrbye, and Tait D Shanafelt. 2018. Physician burnout: contributors, consequences and solutions. *Journal of internal medicine*, 283(6):516–529.

Wei Xu, Chris Callison-Burch, and Courtney Napoles. 2015. Problems in current text simplification research: New data can help. *Transactions of the Association for Computational Linguistics*, 3:283–297.

Wei Xu, Courtney Napoles, Ellie Pavlick, Quanze Chen, and Chris Callison-Burch. 2016. Optimizing statistical machine translation for text simplification. *Transactions of the Association for Computational Linguistics*, 4:401–415.

Manzil Zaheer, Guru Guruganesh, Kumar Avinava Dubey, Joshua Ainslie, Chris Alberti, Santiago Ontanon, Philip Pham, Anirudh Ravula, Qifan Wang, Li Yang, et al. 2020. Big bird: Transformers for longer sequences. *Advances in Neural Information Processing Systems*, 33.

Jingqing Zhang, Yao Zhao, Mohammad Saleh, and Peter Liu. 2020a. Pegasus: Pre-training with extracted gap-sentences for abstractive summarization. In *International Conference on Machine Learning*, pages 11328–11339. PMLR.

Tianyi Zhang, Varsha Kishore, Felix Wu, Kilian Q. Weinberger, and Yoav Artzi. 2020b. Bertscore: Evaluating text generation with bert. In *International Conference on Learning Representations*.

Yuhao Zhang, Daisy Yi Ding, Tianpei Qian, Christopher D. Manning, and Curtis P. Langlotz. 2018. Learning to summarize radiology findings. In *Proceedings of the Ninth International Workshop on Health Text Mining and Information Analysis*, pages 204–213, Brussels, Belgium. Association for Computational Linguistics.

Ming Zhong, Pengfei Liu, Yiran Chen, Danqing Wang, Xipeng Qiu, and Xuanjing Huang. 2020. Extractive summarization as text matching. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 6197–6208, Online. Association for Computational Linguistics.

Chunting Zhou, Graham Neubig, Jiatao Gu, Mona Diab, Francisco Guzmán, Luke Zettlemoyer, and Marjan Ghazvininejad. 2021. Detecting hallucinated content in conditional neural sequence generation. In *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*, pages 1393–1404, Online. Association for Computational Linguistics.

Zhemin Zhu, Delphine Bernhard, and Iryna Gurevych. 2010. A monolingual tree-based translation model for sentence simplification. In *Proceedings of the 23rd International Conference on Computational Linguistics (Coling 2010)*, pages 1353–1361, Beijing, China. Coling 2010 Organizing Committee.

A Appendix

A.1 Implementation Details

Extractive Summarizers:

- A extractive summary contains 8 sentences, they are generated by LexRank, TextRank, BertSum.
- LexRank/TextRank source code: SummerTime
- We use default settings of BertSum for training. The source code and configs are available here.

Abtractive Summarizers:

- We implement our models based on Huggingface Transformers
- Configurations of our abstractive models: num train epochs: 6; max target length: 256; max source length: 1024; batch size: 2; beam size: 1; topK: 50.
- The models we explored: facebook/bart-large; google/pegasus-large; allenai/led-large-16384

Type I Error Detector (Hallucination):

- We implement Big-Bird based on Huggingface Transformers's BigBird implementation
- Key parameters of the model are - max sequence length: 1536; num train epochs 3; batch size: 4. For other hyper-parameters we use the system's default setting.
- The model is: google/bigbird-roberta-base

Type II Error Detector (Missing Content):

- We implement Big-Bird based on Huggingface Transformers's BigBird implementation
- Key parameters of the model are - max sequence length: 1024; num train epochs 3; batch size: 4. For other hyper-parameters we use the system's default setting.
- The model is: google/bigbird-roberta-base
- MetaMap-2020 version is used for medical event identification.

Summarization Evaluation:

- We use rouge-score to obtain ROUGE scores.
- We use SummerTime to obtain BertScores.
- We use this script for SARI evaluation.
- We use py-readability-metrics to compute Dale Chall readability scores.

All Huggingface models could be downloaded from the Huggingface website. All the models are trained and tested on a NVIDIA-V100 GPU. The average training time for our generation models is around 6-8 hours, the average training time for our error alerting models is around 3-5 hours.

Original	Nuclear stress test which was negative for any damage to the heart.
Synthesized	CT scan of the head which was negative for any damage to the brain.
Original	During hospitalization you underwent endovascular repair of your thoracoabdominal aortic aneurysm with Dr [**NAME**]
Synthesized	During hospitalization you underwent a biopsy of your liver with Dr [**NAME**]
Original	You underwent MRI imaging that showed no obstruction in your GI tract.
Synthesized	You underwent cardiac catheterization that showed no abnormalities.
Original	You were found to have decreased levels of oxygen in your blood.
Synthesized	You were found to have bacteria in your blood.
Original	You were admitted for high calcium levels in your blood.
Synthesized	You were admitted for pain in your left leg.

Table 9: Example synthesized summary sentences that contain hallucinations (underlined).

A.2 Example Outputs

We present example source documents, system and reference summaries, and physician-edited system summaries in Tables 8 and 10.

A.3 Human Evaluation Details

The mean Pearson's r of human evaluation scores is 0.282, suggesting a moderate correlation. While the human score correlation is not strong, it is not surprising as studies have revealed that physicians often have vastly different ways of seeing and treating patients, as differences in profession, specialty, experience, or background lead them to pay attention to particular signals or cues and influence how they approach problems (Frimpong et al., 2017). Specifically, we observe when scoring the readability of generated AVS notes, some physician think some complicated phrases (e.g. CT Scan, mental status) would impact patients' understanding, other physicians would think these phrases would be acceptable to be contained in an after-visit summary.

Example 1
Source Document: [** MISC **]y o female with history and exam consistent with left pyelonephritis. RENAL/ID: Patient presented to ED with fever left CVA tenderness and UA consistent with pyelonephritis. During her hospital stay she was treated with IV fluids zofran toradol tylenol and given 2 dose of ceftriaxone. She was discharged on Bactrim 400 mg BID for a total of 14 days. As she already had 2 doses prior to admission she will continue with Bactrim for 12 more days. She was instructed to follow up with PCP within 1 week. FEN/GI: Patient with some nausea on presentation. Will continue with regular diet and discharged with PO Zofran for nausea as needed. NEURO: Patient with back pain secondary to pyelonephritis. She will continue with Tylenol and Ibuprofen as needed. [...]
After-Visit Summary (Reference): You were admitted because you were having abdominal pain. This was initially thought to be due to a kidney infection but an ultrasound showed that you had a cyst in your left ovary. You went to the operating room for a diagnostic laparoscopy and cystectomy. After the operation you had a couple of fainting episodes and your blood levels were decreasing. Your abdominal exam was also concerning. You were taken back to the operating room and were found to have bleeding in your abdomen; the blood was removed and no active bleeding was noted. You tolerated the second procedure well and were able to eat void and your pain was controlled. You will be discharged home in stable condition with follow up with Dr. [** PERSON **].
After-Visit Summary (System): You came to the hospital because you were having fevers and back pain. You were found to have a kidney infection that spread up to your kidneys. We gave you IV fluids and antibiotics and your symptoms improved. You are being discharged with instructions to continue to take oral antibiotics for 12 more days. Please follow up with your PCP within 1 week.
After-Visit Summary (System w/ Physician Edits): You came to the hospital because you were having fevers and back pain. You were found to have a urine infection that spread up to your kidneys. We gave you IV fluids and antibiotics and your symptoms improved. You are being discharged with instructions to continue to take oral antibiotics for 12 more days and PO Zofran for nausea as needed. Please follow up with your PCP within 1 week.
Example 2
Source Document: Mastoiditis Patient presents with symptoms of vertigo. CT head showed that there is mucosal thickening at the bilateral maxillary sinuses and minimal fluid in the bilateral mastoid air cells. Unclear if this is acute or chronic. Patient describes feeling of fullness in head and decrease in hearing over one month. No fever no leukocytosis. Concern for infection and possible mastoiditis. Dr. [** PERSON **] from *[anonymized]* was consulted by ED recommends Unasyn while inpatient Augmentin as an outpatient for 1 week and will follow up as outpatient. Vertigo History of vertigo and on meclizine as outpatient. No improvement with home dose of meclizine. CT scan findings as detailed above. No intracranial mass bleed. Given IVF in ED. given antiemetics and meclizine and treat for possible mastoiditis /mastoid effusions. Neurology dr. [** PERSON **] consulted. Doubt TIA. Likely peripheral nystagmus and MRI head showed no CVA. Essential (primary) hypertension Controlled. Continue atenolol at 25mg nightly and amlodipine 5mg daily. [...]
After-Visit Summary (Reference): You were admitted with vertigo. The head CT showed no acute abnormalities of the brain. The MRi of brain showed only chronic changes. No acute abnormalities. Your head CT showed possible mastoidit is and thickening of the lining of your maxillary sinuses which could be sinusitis related. ENT was called and recommended iv antibiotics for one day and to transition over to oral antibiotics to complete a total course of one week and you will follow up with [** PERSON **] in one week. You were seen by neurology as well. You are doing much better and can go home today.
After-Visit Summary (System): You were admitted for dizziness. You had a CT scan of your head which showed some thickening in the sinuses of your sinuses. You were seen by the ear nose and throat doctor who recommended that you take an antibiotic called Unasyn while you are in the hospital. You also had an MRI of your brain which did not show any stroke. You are doing better and can go home today.
After-Visit Summary (System w/ Physician Edits): You were admitted for dizziness. You had a CT scan of your head which showed some thickening in your sinuses and mastoid. This could be suggestive of an infection but your white cells and temperature were normal. You were seen by the ear nose and throat doctor who recommended that you take an antibiotic called Unasyn while you are in the hospital. You also had an MRI of your brain which did not show any stroke. You are doing better and can go home today.

Table 10: Example input documents, system and reference summaries, as well as physician-edited system summaries.

HeterGraphLongSum: Heterogeneous Graph Neural Network with Passage Aggregation for Extractive Long Document Summarization

Tuan-Anh Phan and **Ngoc-Dung Nguyen** and **Khac-Hoai Nam Bui***
Viettel Cyperspace Center, Viettel Group, Vietnam
{anhpt161,dungnn7,nambkh}@viettel.com.vn

Abstract

Graph Neural Network (GNN)-based models have proven effective in various Natural Language Processing (NLP) tasks in recent years. Specifically, in the case of the Extractive Document Summarization (EDS) task, modeling documents under graph structure is able to analyze the complex relations between semantic units (e.g., word-to-word, word-to-sentence, sentence-to-sentence) and enrich valuable information for the sentence representation. However, long-form document summarization using graph-based approaches is still an open research issue. The main challenge is to represent long documents in a graph structure in an effective way. In this regard, this paper proposes a new heterogeneous graph neural network (HeterGNN) model to improve the performance of long document summarization (HeterGraphLongSum). Specifically, the main idea is to add the passage nodes into the heterogeneous graph structure of word and sentence nodes for enriching the final representation of sentences. In this regard, HeterGraphLongSum includes three types of semantic units such as word, sentence, and passage. Experiments on two benchmark datasets for long documents such as Pubmed and Arxiv indicate promising results of the proposed model for the extractive long document summarization problem. Especially, HeterGraphLongSum is able to achieve state-of-the-art performance without relying on any pre-trained language models (e.g., BERT). The source code is available for further exploitation on the Github[1].

1 Introduction

Document summarization is one of the central problems in NLP, which aims to rewrite a single document or multi documents under a shorter version with preserving the main information. There are two major approaches such as *extractive* and *abstractive* summarization. *Abstractive models* are more sophisticated abilities that require well-comprehensive reading text and generating high-quality text. Specifically, most of the existing architectures have been built based on sequence-to-sequence (Seq2Seq) techniques in different ways such as Recurrent Neural Network (RNN) (Nallapati et al., 2017), Pointer-Generator-Network(See et al., 2017), or Transformer-based models(Zhang et al., 2020; Xiao and Carenini, 2020). Furthermore, the external information, for instance, pre-trained model BERTSum(Liu and Lapata, 2019) and topic modeling (Wang et al., 2020b; Nguyen et al., 2021)) are incorporated to improve performances. Nevertheless, this approach requires a complicated neural network that consists of millions of learnable parameters, which is the cause of raising significant costs in both terms of computation time perplexity and resources. Therefore, *extractive models* still gain much attention. Particularly, extractive document summarization (EDS) takes a document in the form list of sentences and chooses several best candidates from the original document, then combine them to create the summarization. Recent models trend to turn EDS into the sequential binary-labeling task (Nallapati et al., 2017; Cheng and Lapata, 2016; Zhou et al., 2018).

Graph neural network (GNN) has recently been exploited as an emerging line of deep learning architectures, which has powered various domains, including NLP tasks (Vashishth et al., 2020). Specifically, GNN models are able to model complex structural data containing semantic units (node) with relationships (edge) between them (Xu et al., 2019). For the EDS task, each document is represented as a graph structure in which the nodes are the semantic units of the document such as words and sentences. Sequentially, developing edges among sentence nodes are capable to model the cross-sentence relations, which is able to handle the limitation of traditional Seq2Seq-based meth-

*corresponding author
[1]https://github.com/tuananhphan97vn/HeterGraphLongSum

Proceedings of the 29th International Conference on Computational Linguistics, pages 6248–6258
October 12–17, 2022.

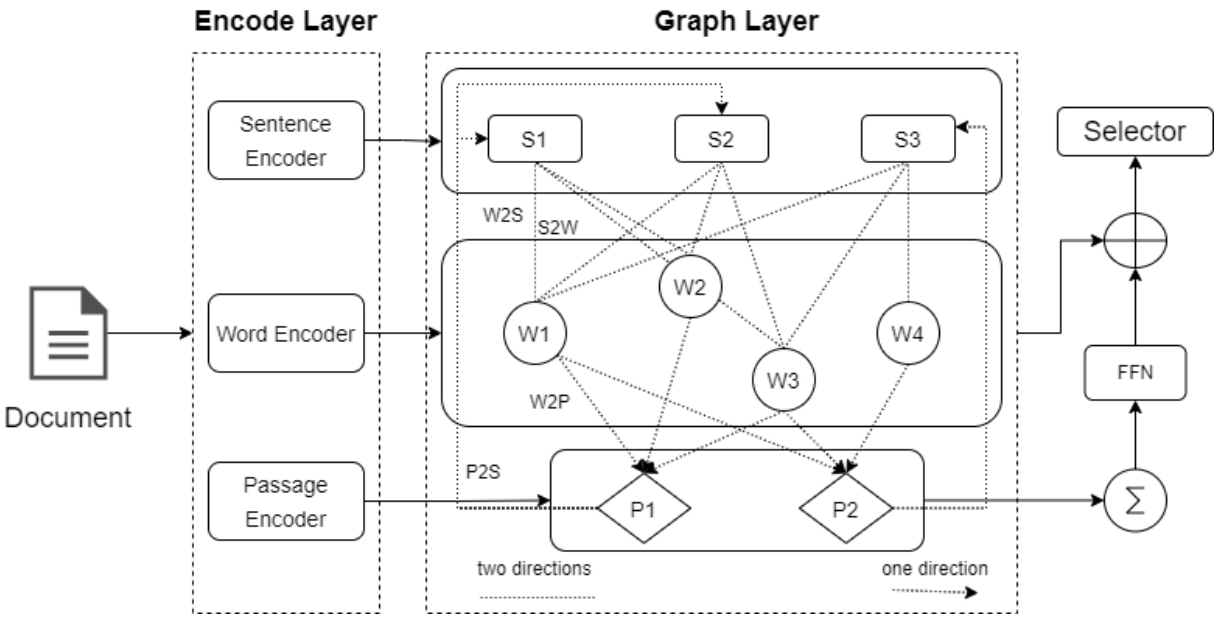

Figure 1: Overview of HeterGraphLongSum model. Passages of each document are defined as a set of sentences in sequence with a fixed number of sentences. In this architecture, the edges from *passage to word* and *sentence to passage* are not taken into account because of the redundancy.

ods in terms of long-distance dependency among sentences (Cui et al., 2020). In particular, cross-sentence edges can be constructed explicitly between sentences (sentence-to-sentence) (Xiao and Carenini, 2019; Jing et al., 2021; Yasunaga et al., 2017) or through intermediate bridge via common words (sentence-word-sentence)(Wang et al., 2020a) or latent topics (sentence-topic-sentence) (Cui et al., 2020).

Although the aforementioned approaches have achieved remarkable results in the EDS problem, most of the architectures are proposed for short documents (i.e., new articles). Long-form document is still a remaining challenge in this research field due to two main reasons: i) most traditional Seq2Seq methods truncate longer documents into small fixed-length sequences (i.e., passages) (Zaheer et al., 2020; Zhang et al., 2021), which leads to information loss problem, especially for the extractive summarization; ii) using GNN-based methods is able to mitigate the information loss by enabling cross-sentence relations, however, representing an effective way for long-text documents into graph structure is still an open research issue. Specifically, since the vocabulary size is limited, when the length of the document is increased, more sentences become neighbors with each other (via common words) which is the cause of the similar embedding between sentences. Therefore, a graph structure, which includes only word nodes and sentence nodes, might not be an effective way to represent the long documents for the EDS problem.

In order to alleviate the aforementioned challenges, this paper presents a new graph-based architecture, which contains three semantic units such as word, sentence, and passage. In particular, the passage nodes are adopted for learning the cross-relations between sentences in different passages. Furthermore, the passage node can be regarded as the local structure of a group of sentence nodes in which the edges between passages and sentences have the possibility to reduce the harm of similar representations of sentences when expanding graph structure with long documents. Figure 1 illustrates the model architecture of HeterGraphLongSum. Specifically, the main contributions of this paper are threefold as follows:

- We present a novel GNN-based method for modeling long-form documents. Specifically, instead of using common methods for learning long documents with the hierarchical perspective (e.g., word-to-sentence-to-passage), we consider passage as one of the node types, which is updated simultaneously with other nodes in the graph. In this regard, more semantic units (additional nodes) in the graph enable the capability to enrich the cross-relations between elements (e.g., sentence representa-

tion).

- We propose a new Heterogeneous GNN (HeterGNN) model for the EDS task, focusing on long documents (e.g., scientific papers). Especially, we consider this issue without employing pre-trained encoders (e.g., BERT). In this regard, our method is able to extend to other low-resource languages without any obstacles.

- We evaluate the proposed model with two benchmark long document datasets such as PubMed and ArXiv. The experiential results indicate that our method is able to achieve the state-of-the-art level in this research field.

2 Related Work

2.1 Neural Extractive Summarization

Neural networks have achieved great success in extractive summarization, which explores different neural components to develop an end-to-end learning model (Zhong et al., 2019). The encoder-decoder frameworks are mainly developed by using RNN (Cheng and Lapata, 2016; Nallapati et al., 2017; Zhou et al., 2018) and Transformer (Zhang et al., 2020; Xiao and Carenini, 2020) with auto-regressive (Jadhav and Rajan, 2018; Liu and Lapata, 2019) or non auto-regressive (Narayan et al., 2018; Arumae and Liu, 2018) decoder. Sequentially, recent remarkable results are mainly developed by using pre-trained language models (e.g., BERT (Devlin et al., 2019)) such as BERTSUM (Liu and Lapata, 2019) and MATCHSUM (Zhong et al., 2020). Most of the aforementioned studies formulate the EDS task as sentence labeling or sentence ranking problems. In this paper, we formulated this task as the binary-labeling problem (Nallapati et al., 2017) and exploited our model with a non-pre-trained CNN/BiLSTM encoder in which we believe that this method is able to easily extend to other low resource languages.

2.2 Graph-based Summarization

Early works on graph-based methods for EDS tasks rely on the similarity scores between sentences in unsupervised manners such as TextRank (Mihalcea and Tarau, 2004) and LexRank (Erkan and Radev, 2004). The core idea of using graph representation is to utilize the linguistic information of sentences. Consequentially, GNNs have been adopted for learning cross-sentence relations with remarkable performances, using the concept of discourse graph (Yasunaga et al., 2017; Xu et al., 2020). Recently, the trend research focuses on representing documents with different types of nodes (heterogeneous graphs) to utilize the effects of additional semantic units such as words, sentences (Wang et al., 2020a; Jin et al., 2020) and latent topics (Cui et al., 2020). In this study, the proposed model exploits the heterogeneous graph structure with more complex units by adding semantic passage nodes to leverage the problems of adopting graph-based models in long document summarization.

2.3 Long Document Summarization

Long document summarization has recently received increased attention since the remained challenge of modeling long texts (Frermann and Klementiev, 2019). Specifically, the current potential solution for this issue is to truncate documents into small fixed-length sequences and use sliding window methods to process the document separately (Beltagy et al., 2020; Zaheer et al., 2020). However, this paradigm leads the serious information loss, which is not suitable for the EDS task, due to this task requiring the information relations of extracted sentences (Li et al., 2020). In this regard, several promising approaches have been introduced for long document summarization. Cohan et al. (2018) presents a hierarchical encoder to capture the discourse structure of the input document with a discourse-aware decoder for abstractive summarization. Xiao and Carenini (2019) leverages the long text summarization task by incorporating a distributed representation of both the global (whole document) and local (section/topic) contexts. Cui and Hu (2021) employs a dynamic memory network with sliding multiple windows to mitigate the information loss between segments of sentences. Regarding the graph-based methods, Cui et al. (2020) adopts a modified graph attention network (GAT) for capturing inter-sentence relationship. Furthermore, latent topics are added as an additional type of node, which incorporates sentence nodes to improve the performance of long document summarization in terms of capturing the relational information of long-distance sentences. In this study, we take the graph-based structure for EDS of long text into account with a different perspective by considering word nodes and sentence nodes for capturing both inter and intra-

sentence relations. Moreover, passage nodes are jointly trained to improve the performance of long texts by learning the cross-relations of sentences with long-distance and mitigating the similar representation problem in the large-scale graph structure.

3 HeterGraphLongSum model

HeterGraphLongSum aims to learn a heterogeneous graph structure for long text summarization. Specifically, we model an input document with three types of nodes such as *word*, *sentence*, and *passage* nodes, as a heterogeneous graph, and using graph attention network (GAT) (Velickovic et al., 2017) for capturing information relations among nodes.

3.1 Graph Construction

Let $G = \{V, E\}$ represent an arbitrary graph, where V and E denote the node and edge sets, respectively. Specifically, as shown in the Figure 1, our directed graph can be defined as $V = \{V_w \cup V_s \cup V_p\}$ and $E = \{E_{w2s} \cup E_{s2w} \cup E_{w2p} \cup E_{p2s}\}$, where V_w, V_s, and V_p stand for three semantic units of a document (i.e., word, sentence, and passage), and E_{w2s}, E_{s2w}, E_{w2p}, and E_{p2s} stand for four types of edges such as *word-to-sentence*, *sentence-to-word*, *word-to-passage*, and *passage-to-sentence*, respectively. Accordingly, the proposed heterogeneous graph structure is designed based on two assumptions as follows:

- The passage units are not available on most publicity datasets in this research field. Therefore, following the previous works on long-form document representations(Zaheer et al., 2020; Zhang et al., 2021), we format the passages in form of a sequence of sentences and created them by concatenating a fixed size with n sentences. In this regard, the number of sentences for each passage is a hyperparameter, which is tuned during the validation process.

- Regarding the certain edge types, instead of adopting the full connection between semantic units, only four types of edges are taken into account such as *word-to-sentence* (w2s), *sentence-to-word* (s2w), *word-to-passage* (w2p), and *passage-to-sentence* (p2s). Accordingly, the edge from *passage-to-word* (p2w) and *sentence-to-passage* (s2p) are not considered because of the redundancy. Specifically, $p2w$ is not considered since many words receive the same information (i.e., from the passage), which might harm the overall performance. Furthermore, there are two types of edge to update the passage information such as $w2p$ and $s2p$. In this regard, we design $w2p$ in our graph structure to enable the cross-passage relations via path $passage \rightarrow sentence \rightarrow word \rightarrow passage$ and remove the $s2p$ edge type. We prove this assumption via the ablation study in the experiment section.

Intuitively, by adding two types of edges from passage nodes, cross-sentences relations can be simultaneously processed in two ways: i) *local information* with path $sentence \rightarrow word \rightarrow passage \rightarrow sentence$; ii) *global information* with path $sentence \rightarrow word \rightarrow sentence$. The additional *local information* enables the model to mitigate the problem of similarity representation between sentences when the graph structure is expanded by adding passage information. Specifically, this issue is specific to sentences located in different positions in the document, which is especially suitable for learning long documents.

3.2 Graph Encoder Embedding

Supporting the matrix features of word node, sentence node and passage node are sequentially denoted as $X_w \in R^{|V_w| \times d_w}$, $X_s \in R^{|V_s| \times d_s}$, and $X_p \in R^{|V_p| \times d_p}$, respectively. The initialized embedding representation of the word node is encoded by using Glove (Pennington et al., 2014). In the case of sentences, instead of using pre-trained models, we combine Convolutional Neural Network (CNN) and bidirectional Long Short-Term Memory (BiLSTM) for the encoder, which can be formulated as follows:

$$(X_s)_j = CNN(x_{1:m}) \oplus BiLSTM(x_{1:m}) \quad (1)$$

where m denotes the number of words in the sentence s_j. In this regard, the Passage feature is encoded by using Bi-directional LSTM based on the hidden state of sentences, which is extracted from the last layer network as follows:

$$(X_p)_i = BiLSTM\left((X_s)_{(j)}\right) \quad (2)$$

where $(X_p)_i$ denotes embedding of the *i-th* passage node, j and k are the *j-th* sentence and number of

sentence per passage ($k * i \leq j \leq k * (i+1)$), respectively.

3.3 Graph Learning Layer

The vectors of nodes are initialized with embedding features, where $H_s^0 = X_s$ $H_w^0 = X_w$, and $H_p^0 = X_p$, respectively. Sequentially, the node representations are updated with the graph attention network.

Graph Attention Network: Given the heterogeneous graph structure and initialized features of each node, GAT is adopted to calculate the hidden states of nodes. Specifically, supporting $\vec{h_i} \in R^{d_{h_i}}$ and N_i denote the input hidden representation and the neighbors of node *i-th*, respectively, the graph attention layer can be calculated as follows:

$$\begin{aligned} z_{ij} &= LeakyRelu(\vec{a}^T(W_q\vec{h_i}||W_k\vec{h_j})) \\ \alpha_{ij} &= \frac{e^{z_{ij}}}{\sum_{k\in N_i} e^{z_{ik}}} \\ \vec{h_i'} &= \sigma\left(\sum_{j\in N_i} \alpha_{ij} W_v \vec{h_j}\right) \end{aligned} \tag{3}$$

where W_q, W_k, W_v, and $\vec{\alpha}$ are learnable parameters and optimized during the training process. The symbol $||$ indicates the concatenation operator. σ denotes the non-linear transform function and $\vec{h_i'}$ denotes the hidden state which presents information gained from the neighboring nodes. Alternatively, multi-head attention can be used for improving the performance, which is calculated as follows:

$$\vec{h_i'} = \|_{k=1}^{K} \sigma\left(\sum_{j\in N_i} \alpha_{ij} W_v^k \vec{h_j}\right) \tag{4}$$

Furthermore, in order to mitigate the gradient vanishing problem, the residual connection is added to the original representation to provide the final hidden state:

$$\vec{h_i''} = \vec{h_i} + \vec{h_i'} \tag{5}$$

Graph Propagation: After initialization, the sentence nodes are updated with their neighbor word nodes and passage nodes by using GAT and FFN layer:

$$\begin{aligned} U_{w2s}^1 &= GAT(H_s^0, H_w^0, H_w^0) \\ U_{p2s}^1 &= GAT(H_s^0, H_p^0, H_p^0) \\ U_s^1 &= \sigma(U_{w2s}^1 + U_{p2s}^1) \\ H_s^1 &= FFN(U_s^1 + H_s^0) \end{aligned} \tag{6}$$

Sequentially, word nodes are updated with the new representation of sentences. Similarly, the passage nodes are updated by the updated word embedding. The updated process at an iteration of GAT can be formulated as follows:

$$\begin{aligned} U_{w2s}^t &= GAT(H_s^{t-1}, H_w^{t-1}, H_w^{t-1}) \\ U_{p2s}^t &= GAT(H_s^{t-1}, H_p^{t-1}, H_p^{t-1}) \\ U_s^t &= \sigma(U_{w2s}^t + U_{p2s}^t) \\ H_s^t &= FFN(U_s^t + H_s^{t-1}) \\ \\ U_w^t &= GAT(H_w^{t-1}, H_s^t, H_s^t) \\ H_w^t &= FFN(U_w^t + H_w^{t-1}) \\ \\ U_p^t &= GAT(H_p^{t-1}, H_w^t, H_w^t) \\ H_p^t &= FFN(U_p^t + H_p^{t-1}) \end{aligned} \tag{7}$$

Note that, H_w^1 and H_p^1 are set to the same values with H_w^0 and H_p^0, respectively.

3.4 Sentence Extraction

For the sentence selector layer, we first extract document representation from the hidden state of passages via the attention layer, then combine document representation and each sentence by using the concatenate operator, which is sequentially formulated as follows:

$$\begin{aligned} z_i &= ReLu(\vec{a_p^T}(\vec{h_p})_i) \\ \alpha_i &= \frac{e^{z_i}}{\sum_j e^{z_j}} \\ \vec{h_d} &= \sum_m \alpha_m * \left(\vec{h_p}\right)_m \end{aligned} \tag{8}$$

$$\vec{h_{d,s_k}} = FFN\left(\vec{h_d} \parallel \vec{h_{s_k}}\right) \tag{9}$$

where $\vec{a_p}$ is learnable parameter. i and k represent indexes of passage *i-th* and sentence *k-th*, respectively. α_i indicates the amount of contribution of passage *i-th* to document representation $\vec{h_d}$. Consequently, if α_i gets a high attention score, this passage tends to be more significant than other passages. Finally, the output sentence-document representation $\vec{h_{d,s_i}}$ is used for sentences classification by using binary cross-entropy loss as the training objective:

$$L = \frac{1}{N}\sum_{i=1}^{i=N} y_i log\left(\hat{y_i}\right) + \left(1 - y_i\right) log\left(1 - \hat{y_i}\right) \tag{10}$$

Model	arXiv			PubMed		
	R-1	R-2	R-L	R-1	R-2	R-L
SumBasic*	29.47	6.95	26.30	37.15	11.36	33.43
LexRank*	33.85	17.36	28.99	39.19	13.89	34.59
Oracle+	53.88	23.05	34.90	55.05	27.48	38.66
Cheng & Lapata (2016)+	42.24	15.97	27.88	43.89	18.53	30.17
SummaRuNNer+	42.91	16.65	28.53	43.89	18.78	30.36
Xiao & Carenini (2019)(Xiao and Carenini, 2019)	43.62	17.36	29.14	44.85	19.7	31.43
Match-Sum	40.59	12.98	32.64	41.21	14.91	36.75
Topic-GraphSum(Cui et al., 2020)	44.03	18.52	32.41	45.95	20.81	33.97
SSN-DM(Cui and Hu, 2021)	45.03	19.03	32.58	46.73	21.00	34.10
HeterGraphLongSum (iter=1)	46.62	18.69	40.77	48.75	22.45	43.97
HeterGraphLongSum (iter=2)	**47.36**	**19.11**	**41.47**	**48.86**	**22.63**	**44.19**

Table 1: Results on the test set. Report results with * are from Cohan et al. (2018), and results with + are from Xiao and Carenini (2019). Other results are obtained from respective papers. Our results are calculated by averaging values of 3 runs.

4 Experiment

4.1 Experimental setup

Dataset: two benchmark datasets of long documents are considered for the experiments such as arXiv and PubMed datasets, which are scientific papers. Accordingly, those datasets are processed following the work in Cohan et al. (2018) and get Oracle results, a gold standard extractive label, by the work in Xiao and Carenini (2019). The statistics of evaluated datasets are illustrated in Table 2.

Datasets	Documents			Avg. Tokens	
	Train	Val	Test	Doc.	Sum.
arXiv	203,037	6,436	6,440	4,938	220
PubMed	119,924	6,633	6,658	3,016	203

Table 2: Statistics of experiential datasets.

Models for comparision: we evaluate the proposed models with recent benchmark models in this research field which are mainly divided into four approaches: traditional EDS models such as SumBasic (Vanderwende et al., 2007) and LexRank (Erkan and Radev, 2004); Seq2Seq-based models such as Cheng & Lapata (Cheng and Lapata, 2016), SummaRuNNer (Nallapati et al., 2017), and Xiao & Carenini (Xiao and Carenini, 2019); pre-trained-based models such as Match-Sum (Zhong et al., 2020); graph-based models such as Topic-GraphSum(Cui et al., 2020) and SSN-DM (Cui and Hu, 2021).

Hyperparameter setting: Regarding the word node generation, following previous work (Xiao and Carenini, 2019), the vocabulary is limited to 50,000. The word embedding initializes with 100 dimensions using Glove pre-trained model (Pennington et al., 2014). The dimension of the sentence and passage are both set to 64. The dimension of the final output representation of all models is set to 64. The multi-head of the GAT layer for s2w is set to 4 and others (i.e., w2s, w2p, and p2s) are set to 1. The passage length is a hyperparameter, which is a constant number. Specifically, we vary the value of passage length from 10 to 30 in order to determine the best results for two evaluated datasets. More details of the impact of passage length are presented in the ablation section. We select top-6 of PubMed and top-5 of arXiv datasets for the decoding process, according to the best performance of the validation set. All models are trained for 20 epochs with a single NVIDIA V100 card (batch size = 32) and use early stopping on the validation set according to entropy loss in order to select the best model.

4.2 Main Results

Table 1 reports the evaluation results on two benchmark datasets. Accordingly, ROUGE is used as the evaluation metric, which includes unigram (R-1), bigram (R-2) overlap, and longest common subsequence (R-L) for measuring informativeness and assessing fluency, respectively. The results are presented in different sections corresponding to different approaches. The first section includes traditional approaches and the Oracle. The second section obtains the results of Seq2Seq-based mod-

Our Model	arXiv			PubMed		
	R-1	R-2	R-L	R-1	R-2	R-L
w/o Passage Node	46.43	18.62	40.56	47.81	21.88	43.01
w/o Doc. Rep.	46.61	18.8	40.76	48.52	22.34	43.77
Proposed Model	47.05	19.01	41.2	48.86	22.63	44.19

Table 3: Reported results of our proposed model and two ablated variants on two benchmark datasets.

els. The third section is Match-Sum, a state-of-the-art BERT-based summarizing model. The next section reports recent graph-based models for the EDS problem. The last section is our model, which includes two versions with different iterations of GAT layers.

Based on the evaluation results, several hypotheses for extractive long document summarization problem can be expressed as follows: i) Using pre-trained models (e.g., BERT and RoBERTa) without any modifications is not effective for long documents. The main reason is the limitation of 512 tokens of BERT-based models; ii) Exploiting the global context (the whole document) is able to improve the performance, even without the need for pre-trained models for extracting features; iii) Using graph layer with external information (e.g., latent topic) for encoder embedding is currently state-of-the-art approach in this research field; iv) our model, which incorporates global context with graph neural network, achieve state-of-the-art results on both benchmark datasets of long documents. Especially, the most advantage is that our method provides promising performances without external information and pre-trained language models. A limitation of our study is that we use the fixed length of sentences for passages, which might not suitable for all datasets with the same value (more detail in the ablation study section). An appropriate solution is to adopt semantic self-segmentation methods for determining passages (Moro and Ragazzi, 2022). We leave this issue for future work of this study.

4.3 Ablation Study

Ablated Variants: in order to analyze the impact of each module in the proposed architecture, we evaluate the proposed model with two ablated variants such as i) **w/o Passage Node** removes the passage node in the heterogeneous graph structure, build a HeterGNN with two types of nodes such as word and sentence; ii) **w/o Doc. Rep.** removes the document representation from passages for the sentence extraction process (Eq. 8). Table 3 shows the results of different variants on two evaluated datasets. As result, the proposed architecture outperforms all variants, which proves that combining both modules can achieve the best results. Especially, by adding passage nodes, the similar representation problem of sentences, when the nodes of words and sentences are increased to represent the long-form document, can be reduced. In particular, the effectiveness of proposed modules is visualized with an example in Figure 2. Accordingly,

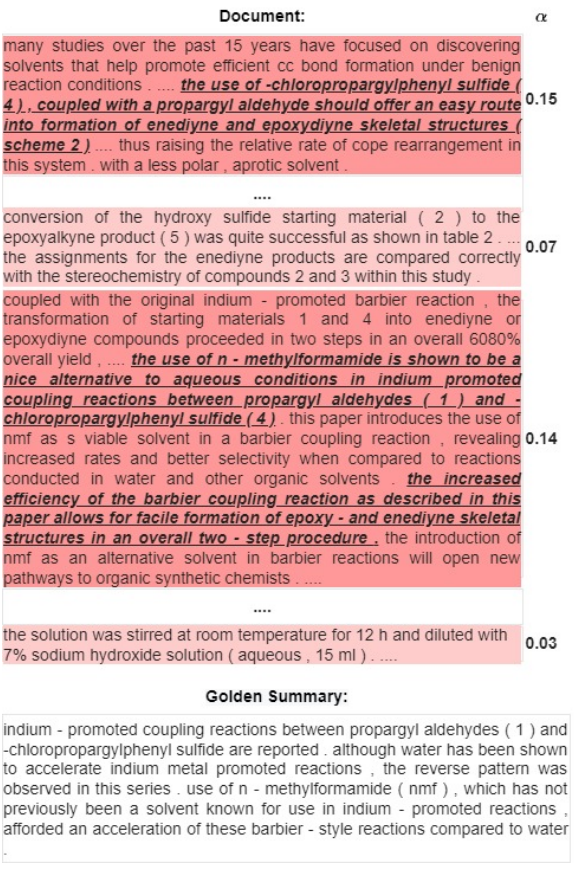

Figure 2: Visualized the efficiency of using passage nodes to enhance sentence representation. The degree of highlighting expresses the important role of the passage in the document. Underlined sentences are model-selected summaries. As result, the selected sentences belong to passages that have high scores of α (Equation 8).

Our Model	arXiv			PubMed		
	R-1	R-2	R-L	R-1	R-2	R-L
w/o w2p	46.30	18.50	40.38	48.76	22.57	44.08
w/o p2s	46.87	18.84	40.96	48.76	22.45	44.00
plus p2w	46.77	18.82	40.89	48.80	22.54	44.06
plus s2p	46.62	18.73	40.70	48.23	22.06	43.43
full edge	46.92	18.80	41.07	48.51	22.42	43.75
Proposed Model	47.05	19.01	41.20	48.86	22.63	44.19

Table 4: Evaluation on the impact of edge types.

each passage has different impacts on the document representation and can be utilized effectively to enrich the information of sentence representations. As shown in the example, the selected sentences mainly belong to the passages, which have high attention scores (Equation 8).

Impact of Edge Types: the proposed heterogeneous graph includes four types of edge such as *s2w*, *w2s*, *w2p*, and *p2s*. Accordingly, *p2w* and *s2p* are removed because of redundancy, which might influence the performance. In this section, we try to evaluate the impact of edge types on the performance by developing several variants of the proposed graph structure, which are: i) **w/o w2p** removes the link from word nodes to passage nodes; ii) **w/o p2s** removes the link from passage nodes to sentence nodes; iii) **plus p2w** adds the link from passage nodes to word nodes; iv) **plus s2p** adds the link from sentence nodes to passage nodes; and v) **full edge** builds a HeterGNN model of three types of nodes such as word, sentence and passage nodes with fully connected among nodes. Specifically, there are total six types of edges of this model such as *w2s*, *w2p*, *s2w*, *s2p*, *p2w*, and *p2s*. Table 4 shows the results of the evaluation. Accordingly, the results indicate that using four types of edges in the proposed model is able to achieve the best results for both evaluated datasets. Note that all the ablated experiments use the same value of passage length (n= 10). More details about this hyperparameter are exploited in the following section.

Length of Passage: is an important hyperparameter in this study in which different datasets might require different values of passage length. In this regard, we conduct experiments to determine the best values of passage length for two evaluated datasets. Table 5 illustrates the impact of passage length on the performance of two datasets, respectively. Specifically, the value of passage length is ranged from 10 to 30 (per 05 periods). As result, the best values of passage length for arXiv and PubMed are 30 and 10, respectively. The experimented result indicates a hypothesis that an adaptive method for automatically segmenting the passage length is able to improve performance. In particular, the passage can be segmented by unsupervised (Alemi and Ginsparg, 2015) or supervised (Koshorek et al., 2018) learning. We take this issue into account for the future work of this study.

Datasets	n	R-1	R-2	R-L
arXiv	10	47.05	19.01	41.20
	15	46.68	18.79	40.78
	20	46.14	18.47	40.30
	25	46.81	18.79	40.95
	30	**47.36**	**19.11**	**41.47**
PubMed	10	**48.86**	**22.63**	**44.19**
	15	48.53	22.26	43.75
	20	48.75	22.45	43.97
	25	48.75	22.57	44.02
	30	**48.86**	22.45	44.07

Table 5: Impact of passage length on the performances of the proposed model.

5 Conclusion

This paper presents a new GNN-based model for extractive long document summarization. Specifically, GNN has been introduced as a promising approach for exploiting the complex relation of elements (e.g., word and sentence) from an input document. However, representing long documents as graph structure is still a remaining challenge. Specifically, lacking cross-relation information between sentences (e.g., long-distance of position in the document) and the increment of nodes might influence the performance. In this regard, this paper proposes a heterogeneous graph including three types of nodes such as word, sentence, and passage,

which are simultaneously learned for enabling the cross-relation between sentences. The evaluation of two standard long documents datasets such as arXiv and PubMed shows that the proposed model outperforms state-of-the-art models in this research field without relying on pre-trained language models (e.g., BERT).

References

Alexander A. Alemi and Paul Ginsparg. 2015. Text segmentation based on semantic word embeddings. *CoRR*, abs/1503.05543.

Kristjan Arumae and Fei Liu. 2018. Reinforced extractive summarization with question-focused rewards. In *Proceedings of ACL 2018, Melbourne, Australia, July 15-20, 2018, Student Research Workshop*, pages 105–111. Association for Computational Linguistics.

Iz Beltagy, Matthew E. Peters, and Arman Cohan. 2020. Longformer: The long-document transformer. *CoRR*, abs/2004.05150.

Jianpeng Cheng and Mirella Lapata. 2016. Neural summarization by extracting sentences and words. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics, ACL 2016, August 7-12, 2016, Berlin, Germany, Volume 1: Long Papers*. The Association for Computer Linguistics.

Arman Cohan, Franck Dernoncourt, Doo Soon Kim, Trung Bui, Seokhwan Kim, Walter Chang, and Nazli Goharian. 2018. A discourse-aware attention model for abstractive summarization of long documents. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT, New Orleans, Louisiana, USA, June 1-6, 2018, Volume 2 (Short Papers)*, pages 615–621. Association for Computational Linguistics.

Peng Cui and Le Hu. 2021. Sliding selector network with dynamic memory for extractive summarization of long documents. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT 2021, Online, June 6-11, 2021*, pages 5881–5891. Association for Computational Linguistics.

Peng Cui, Le Hu, and Yuanchao Liu. 2020. Enhancing extractive text summarization with topic-aware graph neural networks. In *Proceedings of the 28th International Conference on Computational Linguistics, COLING 2020, Barcelona, Spain (Online), December 8-13, 2020*, pages 5360–5371. International Committee on Computational Linguistics.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT 2019, Minneapolis, MN, USA, June 2-7, 2019, Volume 1 (Long and Short Papers)*, pages 4171–4186. Association for Computational Linguistics.

Günes Erkan and Dragomir R. Radev. 2004. Lexrank: Graph-based lexical centrality as salience in text summarization. *J. Artif. Intell. Res.*, 22:457–479.

Lea Frermann and Alexandre Klementiev. 2019. Inducing document structure for aspect-based summarization. In *Proceedings of the 57th Conference of the Association for Computational Linguistics, ACL 2019, Florence, Italy, July 28- August 2, 2019, Volume 1: Long Papers*, pages 6263–6273. Association for Computational Linguistics.

Aishwarya Jadhav and Vaibhav Rajan. 2018. Extractive summarization with SWAP-NET: sentences and words from alternating pointer networks. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics, ACL 2018, Melbourne, Australia, July 15-20, 2018, Volume 1: Long Papers*, pages 142–151. Association for Computational Linguistics.

Hanqi Jin, Tianming Wang, and Xiaojun Wan. 2020. Multi-granularity interaction network for extractive and abstractive multi-document summarization. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, ACL 2020, Online, July 5-10, 2020*, pages 6244–6254. Association for Computational Linguistics.

Baoyu Jing, Zeyu You, Tao Yang, Wei Fan, and Hanghang Tong. 2021. Multiplex graph neural network for extractive text summarization. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing, EMNLP 2021, Virtual Event / Punta Cana, Dominican Republic, 7-11 November, 2021*, pages 133–139. Association for Computational Linguistics.

Omri Koshorek, Adir Cohen, Noam Mor, Michael Rotman, and Jonathan Berant. 2018. Text segmentation as a supervised learning task. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT, New Orleans, Louisiana, USA, June 1-6, 2018, Volume 2 (Short Papers)*, pages 469–473. Association for Computational Linguistics.

Wei Li, Xinyan Xiao, Jiachen Liu, Hua Wu, Haifeng Wang, and Junping Du. 2020. Leveraging graph to improve abstractive multi-document summarization. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, ACL 2020, Online, July 5-10, 2020*, pages 6232–6243. Association for Computational Linguistics.

Yang Liu and Mirella Lapata. 2019. Text summarization with pretrained encoders. In *Proceedings of*

the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing, EMNLP-IJCNLP 2019, Hong Kong, China, November 3-7, 2019, pages 3728–3738. Association for Computational Linguistics.

Rada Mihalcea and Paul Tarau. 2004. Textrank: Bringing order into text. In *Proceedings of the 2004 Conference on Empirical Methods in Natural Language Processing , EMNLP 2004, A meeting of SIGDAT, a Special Interest Group of the ACL, held in conjunction with ACL 2004, 25-26 July 2004, Barcelona, Spain*, pages 404–411. ACL.

Gianluca Moro and Luca Ragazzi. 2022. Semantic self-segmentation for abstractive summarization of long documents in low-resource regimes. *Proceedings of the AAAI Conference on Artificial Intelligence*, 36(10):11085–11093.

Ramesh Nallapati, Feifei Zhai, and Bowen Zhou. 2017. Summarunner: A recurrent neural network based sequence model for extractive summarization of documents. In *Proceedings of the Thirty-First AAAI Conference on Artificial Intelligence, February 4-9, 2017, San Francisco, California, USA*, pages 3075–3081. AAAI Press.

Shashi Narayan, Shay B. Cohen, and Mirella Lapata. 2018. Ranking sentences for extractive summarization with reinforcement learning. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT 2018, New Orleans, Louisiana, USA, June 1-6, 2018, Volume 1 (Long Papers)*, pages 1747–1759. Association for Computational Linguistics.

Thong Nguyen, Anh Tuan Luu, Truc Lu, and Tho Quan. 2021. Enriching and controlling global semantics for text summarization. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing, EMNLP 2021, Virtual Event / Punta Cana, Dominican Republic, 7-11 November, 2021*, pages 9443–9456. Association for Computational Linguistics.

Jeffrey Pennington, Richard Socher, and Christopher D. Manning. 2014. Glove: Global vectors for word representation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing, EMNLP 2014, October 25-29, 2014, Doha, Qatar, A meeting of SIGDAT, a Special Interest Group of the ACL*, pages 1532–1543. ACL.

Abigail See, Peter J. Liu, and Christopher D. Manning. 2017. Get to the point: Summarization with pointer-generator networks. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics, ACL 2017, Vancouver, Canada, July 30 - August 4, Volume 1: Long Papers*, pages 1073–1083. Association for Computational Linguistics.

Lucy Vanderwende, Hisami Suzuki, Chris Brockett, and Ani Nenkova. 2007. Beyond sumbasic: Task-focused summarization with sentence simplification and lexical expansion. *Inf. Process. Manag.*, 43(6):1606–1618.

Shikhar Vashishth, Naganand Yadati, and Partha P. Talukdar. 2020. Graph-based deep learning in natural language processing. In *CoDS-COMAD 2020: 7th ACM IKDD CoDS and 25th COMAD, Hyderabad India, January 5-7, 2020*, pages 371–372. ACM.

Petar Velickovic, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Liò, and Yoshua Bengio. 2017. Graph attention networks. *CoRR*, abs/1710.10903.

Danqing Wang, Pengfei Liu, Yining Zheng, Xipeng Qiu, and Xuanjing Huang. 2020a. Heterogeneous graph neural networks for extractive document summarization. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, ACL 2020, Online, July 5-10, 2020*, pages 6209–6219. Association for Computational Linguistics.

Zhengjue Wang, Zhibin Duan, Hao Zhang, Chaojie Wang, Long Tian, Bo Chen, and Mingyuan Zhou. 2020b. Friendly topic assistant for transformer based abstractive summarization. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing, EMNLP 2020, Online, November 16-20, 2020*, pages 485–497. Association for Computational Linguistics.

Wen Xiao and Giuseppe Carenini. 2019. Extractive summarization of long documents by combining global and local context. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing, EMNLP-IJCNLP 2019, Hong Kong, China, November 3-7, 2019*, pages 3009–3019. Association for Computational Linguistics.

Wen Xiao and Giuseppe Carenini. 2020. Systematically exploring redundancy reduction in summarizing long documents. In *Proceedings of the 1st Conference of the Asia-Pacific Chapter of the Association for Computational Linguistics and the 10th International Joint Conference on Natural Language Processing, AACL/IJCNLP 2020, Suzhou, China, December 4-7, 2020*, pages 516–528. Association for Computational Linguistics.

Jiacheng Xu, Zhe Gan, Yu Cheng, and Jingjing Liu. 2020. Discourse-aware neural extractive text summarization. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, ACL 2020, Online, July 5-10, 2020*, pages 5021–5031. Association for Computational Linguistics.

Keyulu Xu, Weihua Hu, Jure Leskovec, and Stefanie Jegelka. 2019. How powerful are graph neural networks? In *7th International Conference on Learning Representations, ICLR 2019, New Orleans, LA, USA, May 6-9, 2019*. OpenReview.net.

Michihiro Yasunaga, Rui Zhang, Kshitijh Meelu, Ayush Pareek, Krishnan Srinivasan, and Dragomir R. Radev. 2017. Graph-based neural multi-document summarization. In *Proceedings of the 21st Conference on Computational Natural Language Learning (CoNLL 2017), Vancouver, Canada, August 3-4, 2017*, pages 452–462. Association for Computational Linguistics.

Manzil Zaheer, Guru Guruganesh, Kumar Avinava Dubey, Joshua Ainslie, Chris Alberti, Santiago Ontañón, Philip Pham, Anirudh Ravula, Qifan Wang, Li Yang, and Amr Ahmed. 2020. Big bird: Transformers for longer sequences. In *Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020, December 6-12, 2020, virtual*.

Hang Zhang, Yeyun Gong, Yelong Shen, Weisheng Li, Jiancheng Lv, Nan Duan, and Weizhu Chen. 2021. Poolingformer: Long document modeling with pooling attention. In *Proceedings of the 38th International Conference on Machine Learning, ICML 2021, 18-24 July 2021, Virtual Event*, volume 139 of *Proceedings of Machine Learning Research*, pages 12437–12446. PMLR.

Jingqing Zhang, Yao Zhao, Mohammad Saleh, and Peter J. Liu. 2020. PEGASUS: pre-training with extracted gap-sentences for abstractive summarization. In *Proceedings of the 37th International Conference on Machine Learning, ICML 2020, 13-18 July 2020, Virtual Event*, volume 119 of *Proceedings of Machine Learning Research*, pages 11328–11339. PMLR.

Ming Zhong, Pengfei Liu, Yiran Chen, Danqing Wang, Xipeng Qiu, and Xuanjing Huang. 2020. Extractive summarization as text matching. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, ACL 2020, Online, July 5-10, 2020*, pages 6197–6208. Association for Computational Linguistics.

Ming Zhong, Pengfei Liu, Danqing Wang, Xipeng Qiu, and Xuanjing Huang. 2019. Searching for effective neural extractive summarization: What works and what's next. In *Proceedings of the 57th Conference of the Association for Computational Linguistics, ACL 2019, Florence, Italy, July 28- August 2, 2019, Volume 1: Long Papers*, pages 1049–1058. Association for Computational Linguistics.

Qingyu Zhou, Nan Yang, Furu Wei, Shaohan Huang, Ming Zhou, and Tiejun Zhao. 2018. Neural document summarization by jointly learning to score and select sentences. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics, ACL 2018, Melbourne, Australia, July 15-20, 2018, Volume 1: Long Papers*, pages 654–663. Association for Computational Linguistics.

GRETEL: Graph Contrastive Topic Enhanced Language Model for Long Document Extractive Summarization

Qianqian Xie
National Centre for Text Mining,
Department of Computer Science,
The University of Manchester
qianqian.xie@manchester.ac.uk

Jimin Huang
Chancefocus AMC.
jimin@chancefocus.com

Tulika Saha and **Sophia Ananiadou**
National Centre for Text Mining, Department of Computer Science,
The University of Manchester
{tulika.saha,sophia.ananiadou}@manchester.ac.uk

Abstract

Recently, neural topic models (NTMs) have been incorporated into pre-trained language models (PLMs), to capture the global semantic information for text summarization. However, in these methods, there remain limitations in how they capture and integrate the global semantic information. In this paper, we propose a novel model, **G**raph cont**R**astiv**E** **T**opic **E**nhanced **L**anguage model (GRETEL), that incorporates the graph contrastive topic model with the pre-trained language model, to fully leverage both the global and local contextual semantics for long document extractive summarization. To better capture and incorporate the global semantic information into PLMs, the graph contrastive topic model integrates the hierarchical transformer encoder and the graph contrastive learning to fuse the semantic information from the global document context and the gold summary. To this end, GRETEL encourages the model to efficiently extract salient sentences that are topically related to the gold summary, rather than redundant sentences that cover sub-optimal topics. Experimental results[1] on both general domain and biomedical datasets demonstrate that our proposed method outperforms SOTA methods.

1 Introduction

Due to the well-known limitation of pre-trained language models (PLMs) (Devlin et al., 2019; Wang et al., 2021) that they fail to capture long-range dependencies (Beltagy et al., 2020), attempts have been proposed to integrate neural topic models (NTMs) (Cao et al., 2015; Peng et al., 2018; Xie et al., 2021) into PLMs, which have shown significant improvement in the performance of the text summarization task (Wang et al., 2020b; Cui and Hu, 2021b; Nguyen et al., 2021; Fu et al., 2020). In addition to the local contextual information captured in PLMs, NTMs can provide an approximation of the global semantics captured from document contents, i.e., latent topics, as well as their posterior topic representations. The global semantics are further used to guide the model, to generate coherent summaries which cover the most relevant topics discussed within the document, via the attention mechanism (Wang et al., 2020b; Aralikatte et al., 2021; Nguyen et al., 2021; Fu et al., 2020) or graph neural networks (GNNs) (Cui and Hu, 2021b; Cui et al., 2020).

However, there exists the **semantic gap** between latent topics as the approximation global semantics, and the true global semantics due to two major limitations for existing methods. The first limitation concerns the nature of **unsupervised topic inference** in these methods, where topics and posterior topic distributions are learned from documents in an unsupervised manner, without considering the key semantic information conveyed in the gold summary (mostly the abstract). Existing methods using document-word features, without accessing the semantic information of the gold summary, can extract sub-optimal topics with high-frequency words. However, sub-optimal topics with high-frequency words, do not necessarily cover the true global semantics that is condensed in the gold summary. This results in the wrong assignment of the document with sub-optimal topics, and con-

[1] https://github.com/xashely/GRETEL_extractive

Proceedings of the 29th International Conference on Computational Linguistics, pages 6259–6269
October 12–17, 2022.

sequently the model extract redundant sentences containing high-frequency topic words.

Another limitation is that existing methods rely on **document word features** such as Bag-of-Words (BOWs) to extract latent topics, and disregard the sequential and syntactic dependency between words. This may lead to sequentially and syntactically correlated words being allocated to different topics. One solution is to provide NTMs with contextual representations from PLMs, which is nevertheless challenging to directly apply in existing methods for text summarization. PLMs in existing methods are forced to truncate the input to a limited length owing to the complexity of language models. Thus the representations from partial content of the document cannot necessarily help NTMs to mine informative topics that cover the whole content of the document, especially for long documents. Overall, although existing methods encourage the summary with sentences that are topically similar to the document topic distribution, the summary focuses more on sentences with high-frequency words and can have low semantic similarity with the gold summary. To help understand these limitations, we make a detailed analysis based on the benchmark datasets in section 3.

In response to the above, we propose a novel **G**raph cont**R**astiv**E** **T**opic **E**nhanced **L**anguage model (GRETEL), that incorporates the graph contrastive topic model (GCTM) empowered by the semantic information of the gold summary and the global document context, with PLMs for long document extractive summarization. The first distinguishing feature of GRETEL is the employment of the hierarchical transformer encoder (HTE) to fully embed the global context of long documents, to inform topic representations of documents and sentences. The global contextual information captured in HTE but missing in the BOW feature, allows the model to learn more discriminative document and sentence topic representations, and coherent topics.

Secondly, it utilizes graph contrastive learning with the supervised information from the gold summary. It pushes close topic representations of documents and sentences that have high semantic similarity with the gold summary and pulls away otherwise. This encourages the model to capture better global semantic information: latent topics that describe the most key information in the original document content. Therefore, it allows the model to select key sentences that are topically similar to the gold summary. Experimental results demonstrate that our method can effectively distinguish salient sentences from documents with both global and local semantics, leading to superior performance compared to previous methods.

2 Related Work

Topic enhanced PLMs for Text Summarization. Many studies have investigated the application of PLMs for extractive summarization, including BERTSum (Liu and Lapata, 2019), DiscoBert(Xu et al., 2020), MatchSum (Zhong et al., 2020) et al. However, these methods fail to capture the global context of long documents due to the limitation of PLMs. To address it, several studies combined topic modeling with PLMs to introduce global semantic information. Wang et al. (2020b) proposed to extract firstly latent topics independently and then use them to improve the summarization model. Other studies (Aralikatte et al., 2021; Nguyen et al., 2021; Cui et al., 2020) proposed to use the BOW as input features for neural topic modeling and improved the transformer encoder and decoder with the extracted latent topics with an attention mechanism for abstractive summarization (Aralikatte et al., 2021; Fu et al., 2020; Nguyen et al., 2021). Cui et al. (2020); Cui and Hu (2021b) proposed to use the graph neural network to infuse topics into contextual representations from PLMs, for multi-document abstractive summarization and extractive summarization. Fu et al. (2020) considered extract both document and paragraph-level topic distribution, and use them to guide the abstractive summarization.

PLMs for Long Document Summarization. To address the limitation of encoding the full context of long documents using PLMs, another direction is to design the efficient sparse self-attention or using a sliding window. Narayan et al. (2020) proposed a step-wise model with a structured transformer. Huang et al. (2021) proposed a computationally efficient method based on the head-wise positional strides, to identify salient content for long documents. Liu et al. (2021) employed a transformer with multi-granularity sparse attentions. Cui and Hu (2021a) used a sliding selector network with dynamic memory, in which the sliding window is used to encode input documents, segment by segment. Grail et al. (2021) divided long documents into multiple blocks and encoded them by independent transformer networks.

3 Dataset-dependent Analysis for Limitations

To better illustrate the limitations of existing methods, we present a dataset-dependent analysis, with the aim to answer two key questions: 1) Do the extracted topics from topic models tend to focus on high-frequency words? and 2) Due to it, would there be a semantic gap between topics as the approximation of global semantics and true global semantics in the gold summary?

We first present the top-10 words of topics learned by the traditional topic model LDA on the PubMed (Cohan et al., 2018) dataset, as shown in Table 1. It shows that there is a high overlap between words in learned topics and high-frequency words. This is also reported in previous studies (Griffiths and Steyvers, 2002; Steyvers and Griffiths, 2007; Chi et al., 2019), that words are mentioned more frequently, have a higher probability conditioned on topics on average. Since, they infer the posterior distribution of documents over topics, according to the co-occurrences of words in the whole document collections.

T1: type treatment consistent needed lower disorders sensitive patient acid way T2: male group treatment followed cells per side plasma american health T3: al dna clinical risk observed tube lower inflammatory et features T4: type al clinical mice bacteria high vs posterior conditions side T5: differences performed results side number higher size tube et patients T6: dna revealed smoking control mental number change sd light versus
Top high-frequency words: patients, study, using, cells, group, treatment, et, one, al, data, studies, two, patient, results, cell, time, however, figure, significant, reported, high, disease, analysis, clinical, found, age, years, associated, showed, different, compared, risk, levels

Table 1: top-10 words of topics learned by LDA on Pubmed dataset.

In Table 2, we further compare the mean score of ROUGE-1 (Lin and Hovy, 2003) F1 and ROUGE-2 F1 of the oracle summary, and summary based on the generated topics among all datasets used in our experiments. It shows a much lower rouge score on all datasets for summaries using generated topics, which indicates a semantic gap between the latent topics and the gold summary. The latent topics would guide the method to select sentences that are topically similar to the posterior distribution of documents, rather than informative sentences, that cover the semantics in the gold summary.

Dataset	Oracle Summary	Summary with Topics
CNN/DM	0.811	0.174
ArXiv	0.826	0.169
PubMed-Long	0.845	0.184
PubMed-Short	0.847	0.187
CORD-19	0.861	0.188
S2ORC	0.841	0.193

Table 2: The mean score of ROUGE-1 F1 and ROUGE-2 F1 between different summaries with the gold summary, averaged on all documents.

Oracle summary: this case report illustrates three learning points about cervical fractures in ankylosing spondylitis, and it highlights the need to manage these patients with the neck initially stabilised in flexion. We describe a case of cervical pseudoarthrosis that is a rare occurrence after fracture of the cervical spine with ankylosing spondylitis. This went undetected until the development of myelopathic symptoms many months later. The neck was initially stabilised in flexion using tongs, and then slowly extended before anterior and posterior fixation was performed. (**Mean score on ROUGE-1 F1 and ROUGE-2 F1: 0.994**)
Summary based on topic words: A patient's neurological condition may be made worse by extension of the neck, as the spinal cord may be compromised by the angle that is formed between the upper and lower rigid bony segments of the cervical spine. Over the previous 5 weeks, he had been experiencing increasing, although intermittent, symptoms including: sharp pains in the posterior aspect of his neck with head movement, abdominal pain and paraesthesia with numbness of his fingers and toes. Certainly significant trauma to a rigid and osteoporotic spine will cause fracture, and then the effect of instability at the fracture site (the fused spinal segments can be thought of as a long bone) will produce a pseudoarthrosis. (**Mean score on ROUGE-1 F1 and ROUGE-2 F1 : 0.172**)
Top-topic words: cervical, spine, neck, ankylosing, fractures spondylitis, fixation, spinal, stabilised, anterior, posterior, flexion, fracture, c7, trauma, traction, paraesthesia, weakness, immobilisation, immobilised, bony, limb, head, cord, post-operatively.....

Table 3: An example document. High-frequency topic words that appeared in sentences are marked with a red color.

4 Method

To address the aforementioned limitations of existing methods, we propose our method GRETEL, to better capture and incorporate the global semantics to improve PLMs, for long document extractive summarization. Given u sentences $\{s_1, \cdots, s_u\}$ of a document i from the corpus D, extractive summarization aims to select v informative sentences from u sentences ($v \ll u$) as the summary S for the document i. This task can be formulated as a binary sentence classification problem. We assign label $y_{i,j} = 1$ to sentence $s_{i,j} (j \in \{1, \cdots, u\})$ for the summary, or $y_{i,j} = 0$ otherwise.

As shown in Figure 1, different from previous methods, we leverage the contextual representations from PLMs, and gold summary to guide the topic inference. To this end, we first employ the hierarchical transformer encoder (HTE) to fully encode the global context of long documents, and then design the supervised graph contrastive loss, to push close the document topic distribution and topic distributions of salient sentences. This helps our method to capture better global semantics, that effectively distinguish between salient and non-salient sentences, according to their contextual and semantic connections to the gold summary.

4.1 Hierarchical Transformer Encoder

To fully encode the document contents, especially for long documents, we propose to use a the Hierarchical Transformer Encoder (HTE) based on

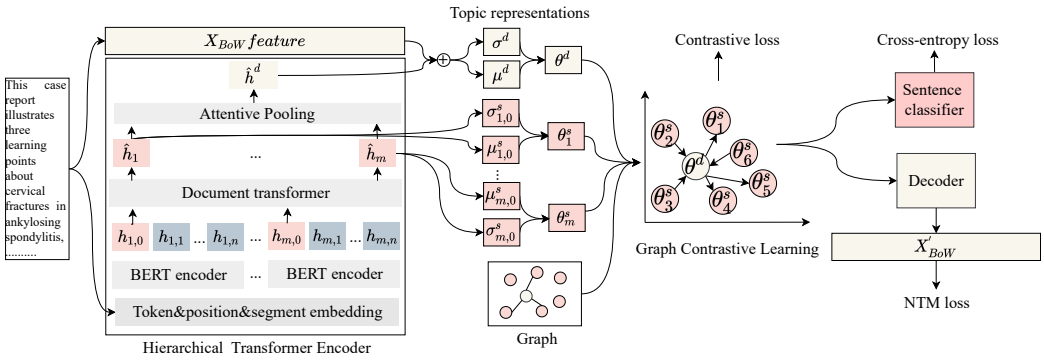

Figure 1: The model architecture of GRETEL

blocks with two modules: the block transformer encoder and the document transformer.

Block transformer encoder. We first split the document $d = \{blk_1, blk_2, \cdots, blk_m\}$ into m blocks with fixed length, in which each block $blk_l = \{x_{l,0}, \cdots, x_{l,n}\}(l \in \{1, \cdots, m\})$ has n tokens. Subsequently, each token $x_{l,p}(p \in \{1, \cdots, n\})$ at block blk_l is represented by the vector $E_{l,p}$, which is the sum of the token embedding, the block embedding, and the position embedding. Take E_l as the input embedding, the contextual representations of tokens in each block blk_l can be learned by the PLMs based transformer encoder: $h_l = BERT(E_l)$. Following previous studies (Liu and Lapata, 2019), we insert the [CLS] and [SEP] tokens at the start and end of each sentence in the block. We consider the representations of [CLS] tokens as the contextual representations of their corresponding sentences: $h^s = \{h^s_1, \cdots, h^s_u\}$, which capture the local contextual semantic in each block.

Document transformer encoder. To further model the correlations among intra-block, we stack the document transformer encoder on h^s to yield the document context-aware sentence representations: $\tilde{h}^s = Transformer(h^s)$. To denote the position of each block, we add the block position embedding (Vaswani et al., 2017) to h^s. Finally, we use the pooling layer to generate the document representation h^d based on $\tilde{h}^s$.

4.2 Graph Contrastive Topic Model

Next, we introduce the graph contrastive topic model, to capture global semantics empowered by the semantic information from HTE and the gold summary. It consists of a probabilistic topic encoder with HTE and supervised graph contrastive learning and a probabilistic decoder.

4.2.1 Probabilistic Topic Encoder Enhanced with HTE

We assume that θ^s and θ^d refer to sentence topic distributions and document topic distribution, β represents the topics (topic word distributions in the vocabulary), and X_i is the BoW feature of document i. Different from existing methods (Wang et al., 2020b; Aralikatte et al., 2021; Fu et al., 2020) considering only the BoW features, we further employ the representations from HTE to leverage the semantic and syntactic dependencies among words to generate more coherent topics and topic distributions for documents and sentences.

For document topic distribution, we sample it from the logistic normal distribution [2]. We first generate the mean and covariance of a multinomial distribution variable and then use the softmax activation function to convert it into the logistic normal distribution variable. Based on the contextual hidden representations from HTE and BoW features, for each document i, we have:

$$\begin{aligned} \tilde{h}^d_i &= f_X(X_i) + \tilde{h}^d_i \\ \mu^d_i &= f^d_\mu(\tilde{h}^d_i), \sigma^d = diag(f^d_\sigma(\tilde{h}^d_i)) \\ \theta^d_i &= softmax(\mu^d_i + (\sigma^d_i)^{\frac{1}{2}} \epsilon^d_i) \end{aligned} \tag{1}$$

where $\epsilon^d_i \in N(0, I)$ is the sampled noise variable, f^d_μ and f^d_σ are the feed-forward neural networks which takes input as the BoW feature X_i and the contextual hidden representations $\tilde{h}^d_i$ of document i from HTE respectively.

The sentence topic distribution is sampled with only the document context-aware hidden represen-

[2] Following the previous method (Srivastava and Sutton, 2017), we use the logistic normal distribution to approximate the Dirichlet distribution.

tations of sentences from HTE:

$$\mu^s_{i,j} = f^s_\mu(\tilde{h}^s_{i,j}), \sigma^s_{i,j} = diag(f^s_\sigma(\tilde{h}^s_{i,j}))$$
$$\theta^s_{i,j} = \mu^s_{i,j} + (\sigma^s_{i,j})^{\frac{1}{2}} \epsilon^s_{i,j} \quad (2)$$

where $\epsilon^s_{i,j} \in N(0, I)$ is the sampled noise variable, f^s_μ and f^s_σ are the feed-forward neural networks which takes the same input as the representation from HTE for the sentence j in the document i. Notice that the BoW features for sentences are not considered since they would be too sparse to introduce external noise.

4.2.2 Supervised Graph Contrastive Learning

Although the posterior topic distributions can now utilize the sequential dependencies of words from HTE, they still cannot distinguish between important and redundant topics without the semantic information from the gold summary. Thus, to fill the gap between the posterior topic distributions from NTMs and the key semantics in the gold summary, we propose the supervised graph contrastive learning to explicitly guide the document topic and sentence topic representations with the gold summary.

Graph Construction. For each document, we first build the graph $G = \{V, E\}$ with nodes V as the document and all its sentences. Edges E can be represented by the adjacency matrix A, in which the edge between two nodes (i, j) is defined as:

$$A_{i,j} = \begin{cases} 1, & \textit{i is the document node, j is the sentence node, and j} \in S^+ \\ 1, & i = j \\ 0, & otherwise \end{cases} \quad (3)$$

where S^+ is the oracle summary of each document which has the maximum semantic similarity with the gold summary. Notice that the graph is a bipartite graph with only connections between the document and sentences.

Graph Contrastive Representation Learning. Based on the bipartite graph embedded with the supervision information, we argue that the representation of the document should be similar to representations of informative sentences in the oracle summary and dissimilar to representations of redundant sentences that are not mentioned. Therefore, we design the following loss for the graph contrastive representation learning:

$$\mathcal{L}_{con} = -\frac{1}{|V|} \sum_{i=1}^{|V|} log(\frac{\sum_{0<A_{i,j}} -A_{i,j} cos(x_i, x_j)}{\sum_{A_{i,j}=0} cos(x_i, x_j)}) \quad (4)$$

where $|V|$ is the number of nodes, cos denotes the cosine similarity, and $x_i \in \theta^d_i, \theta^s_i$ means the features of node i. This loss explicitly pushes close the topic distributions of the nodes with connections in the graph, i.e., the document node and the sentence node in the oracle summary, and pulls away otherwise. It guides the model to learn more discriminative document and sentence distributions that are semantically related to the gold summary.

4.3 Probabilistic Decoder

Based on sampled sentences and document representations, we use the probabilistic decoder to generate the observed words and predict the labels of sentences in each document. For each document i, we assume the v-th word $w^d_{i,v}$ is generated from the multinomial distribution based on the dot product of the document representations and topics:

$$p(w^d_{i,v}|(\theta^d_i, \beta); \Phi) = Mult([\theta^d_i \cdot \beta]) \quad (5)$$

where Φ is the parameter set of the probabilistic decoder. The topics β are randomly initialized. We assume the j-th sentence label $\tilde{y}_{i,j}$ of document i is generated from the feed-forward neural network f_y with the sigmoid activation function, based on the sentence representation $\theta^s_{i,j}$:

$$p(\tilde{y}_{i,j}|\theta^s_{i,j}; \Phi) = f_y(\theta^s_{i,j}) \quad (6)$$

Since we fill the gap between the approximation and true global semantics with supervised contrastive learning based on both the contextual representations from HTE and BoW features, our method allows us to directly use the sentence topic representations to predict the labels of sentences, without any further distillation or fusion.

4.4 Optimization

We optimize the loss function from both graph contrastive topic modeling and extractive summarization to support joint inference. The final loss of GRETEL is the sum of the evidence lower bound (ELBO) and the graph contrastive loss:

$$\mathcal{L} = \mathcal{L}(\Theta, \Phi; X_i) + \eta \mathcal{L}_{con} \quad (7)$$

where η is the parameter to control the sensitivity of the contrastive normalization, $\mathcal{L}_{con}$ is the contrastive loss, and $\mathcal{L}(\Theta, \Phi; X_i)$ is:

$$\begin{aligned} \mathcal{L}(\Theta, \Phi; X_i) = & \mathbb{E}_{q_\Theta(\theta^s_i|\tilde{h}^s_i)}[logP_\Phi(y_i|\theta^s_i)] \\ & + \mathbb{E}_{q_\Theta(\theta^d_i|X_i, \tilde{h}^d_i)}[logP_\Phi(w_i|\theta^d_i, \beta)] \\ & - D_{KL}[q_\Theta(\theta^d_i|X_i, \tilde{h}^d_i)||P_\Phi(\theta^d_i)] \end{aligned} \quad (8)$$

where y_i is the ground truth labels of sentences in document i. The ELBO is composed of three terms, including the sentence label prediction loss for the extractive summarization, the word reconstruction loss of the neural topic modeling, and the KL divergence between the variational posterior and the prior of θ^d, which uses the prior $P_\Phi(\theta^d|\alpha)$ to normalize the document topic representations.

5 Experimentation Details

In this section, we present the details of the datasets used, evaluation metrics, and different baselines.

Datasets. To evaluate the effectiveness of GRETEL, we conducted experiments on four benchmark datasets and two biomedical domain-specific long document datasets: 1) CNN/DM (Hermann et al., 2015): a commonly used news dataset; 2) Arxiv (Cohan et al., 2018): a dataset containing long scientific documents from the Arxiv website; 3) PubMed-Long (Cohan et al., 2018): a dataset containing long scientific documents from biomedicine; 4) PubMed-Short (Zhong et al., 2020): adapted PubMed-Long to use only the introduction of the document as input and filter noisy documents; 5) CORD-19 (Wang et al., 2020a): an openly released dataset including long biomedical scientific papers related to COVID-19. We use the version of the dataset which was released on 2020-06-28 (Bishop et al., 2022; Xie et al., 2022); 6) S2ORC (Lo et al., 2020): a publicly released dataset that includes long scientific papers from several domains. We sample a random subset of articles from only the biomedical domain (Bishop et al., 2022; Xie et al., 2022). We show the statistics of the datasets in Table 4. We use abstracts of documents as the gold summary. For CNN/DM and Arxiv, we extract 3 and 7 sentences respectively to formulate the final summary, following previous methods (Zhong et al., 2020). For the remaining datasets, we extract 6 sentences to formulate the summary (Bishop et al., 2022; Xie et al., 2022).

Dataset	Train	Valid	Test	Avg len	Ext
CNN/DM	287,226	13,368	11,490	757	3
ArXiv	203,037	6,436	6,440	5,038	7
PubMed-Long	119,924	6,633	6,658	3,235	6
PubMed-Short	83,233	4,676	5,025	444	6
CORD-19	31,505	6,299	4,200	3,324	6
S2ORC	47,782	9,556	6,371	2,631	6

Table 4: Statistics of datasets. Ext denotes the number of sentences extracted in the final summary.

5.1 Implementation Details

Our method is implemented by Pytorch and Huggingface (Wolf et al., 2020). We investigated the RoBERTa (Liu et al., 2019) implemented in Huggingface as the encoder. We use the base size of it. We set the learning rate to 2e-3, dropout rate to 0.0, warmup steps to 5000, topic number between$\{100, 200, 300, 400, 500\}$, the parameter to control the negative samples of the contrastive loss γ to 1, and the weight parameter η to 0.5. We set the hidden size of the transformer in HTE to 768. Due to the memory limitations of GPU, we set the max tokens of input documents as 6000. We train the model with 50000 steps and save the model checkpoint at every 1000 steps. We select the best checkpoint according to the loss in the validation and report the results in the test. To extract the sentence label for training the model, we use the greedy search algorithm (Nallapati et al., 2017) to select the oracle summary of each document, via maximizing the ROUGE-2 score against the gold summary. we use the pyrouge[3] to calculate the ROUGE (Lin, 2004) metric.

Baselines and Metrics. We compare our model with SOTA extractive summarization methods including: 1) PLMs based methods: BERTSum (Liu and Lapata, 2019) and MatchSum (Zhong et al., 2020); 2) PLMs based models for long documents: HIBERT (Zhang et al., 2019), ETCSum (Narayan et al., 2020), SSN-DM (Cui and Hu, 2021a), GBT-EXTSUM (Grail et al., 2021), Longformer-Ext (Beltagy et al., 2020), Reformer-Ext (Kitaev et al., 2019), BERTSum+SW (Liu and Lapata, 2019) which uses the BERTSum to sequentially encode the full context with the sliding window; 3) topic enhanced transformer method: TopicGraphSum (Cui and Hu, 2021b), which is the only PLMs-based model with topic modeling for extractive summarization. Following (Liu and Lapata, 2019), we report the unigram (ROUGE-1), bigram F1 (ROUGE-2), and the longest common subsequence (ROUGE-L) between the generated summary and the gold summary.

6 Results and Analysis

A series of experiments were conducted to demonstrate the efficacy of the proposed method.

[3] https://github.com/andersjo/pyrouge.git

Datasets	CNN/DM			Arxiv			PubMed-Long			PubMed-Short			CORD-19			S2ORC		
Metrics	R-1	R-2	R-L	R-1	R-2	R-L	R-1	R-2	R-L	R-1	R-2	R-L	R-1	R-2	R-L	R-1	R-2	R-L
LEAD	40.11	17.54	36.32	33.66	8.94	22.19	36.19	11.82	32.96	37.58	12.22	33.44	32.40	8.97	29.30	36.62	16.57	33.11
ORACLE	56.22	33.74	52.19	53.88	23.05	44.90	50.26	28.32	46.33	45.12	20.33	40.19	46.20	22.86	42.08	58.34	34.48	54.36
BERTSum	43.25	20.24	39.63	41.24	13.01	36.10	41.09	15.51	36.85	41.05	14.88	36.57	36.25	10.83	32.85	40.53	16.31	37.50
MatchSum	44.41	20.86	40.55	-	-	-	-	-	-	41.21	14.91	36.75	-	-	-	-	-	-
Topic-GraphSum	44.02	20.81	40.55	44.03	18.52	32.41	45.95	20.81	33.97	-	-	-	-	-	-	-	-	-
HiBERT	42.37	19.95	38.83	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
ETCSum	43.84	20.80	39.77	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
Longformer-Ext	43.00	20.20	39.30	45.24	16.88	40.03	43.75	17.37	39.71	42.03	16.08	38.01	43.61	16.27	39.39	47.73	22.67	44.03
Reformer-Ext	38.85	16.46	35.16	43.26	14.68	38.10	42.32	15.91	38.26	41.67	15.78	37.88	42.32	16.11	38.87	46.12	21.55	43.21
HETFORMER	44.55	20.82	40.37	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
SSN-DM	-	-	-	45.03	19.03	32.58	46.73	21.00	34.10	-	-	-	-	-	-	-	-	-
BERTSum+SW	43.78	20.65	39.67	47.86	19.17	42.50	46.36	19.67	42.49	42.07	15.10	37.29	42.51	15.72	38.58	46.21	19.73	43.01
GBT-EXTSUM	42.93	19.81	39.20	48.08	19.21	42.68	46.87	20.19	42.68	-	-	-	-	-	-	-	-	-
GRETEL	**44.62**†	**20.96**†	**40.69**†	**48.17**†	**20.31**†	**42.84**†	**48.20**†	**21.20**†	**43.16**†	**42.53**†	**16.55**†	**38.61**†	**43.91**†	**16.54**†	**40.01**†	**48.24**†	**23.34**†	**44.55**†

Table 5: ROUGE F1 results of different models on CNN/DM, Arxiv, PubMed-Long, PubMed-Short, CORD-19, and S2ORC under 5 times running. † means outperform the existing model with best performance significantly ($p < 0.05$). Part results are from (Grail et al., 2021; Zhong et al., 2020; Cui and Hu, 2021a).

6.1 Main Results

We first present the ROUGE F1 results of different models on all datasets in Table 5, which shows that our method GRETEL outperforms all existing baseline methods in all datasets. It demonstrates the superiority of our method GRETEL to other methods, via capturing better global semantics with the guidance of the gold summary and the leverage of contextual information and word features simultaneously. Our methods and Topic-GraphSum both present superior performance over methods without the topic information, such as BERTSum and MatchSum, indicating the importance of modeling the global semantic information with the approximation of latent topics. When comparing with Topic-GraphSum incorporating latent topics, our method yields better performance on all datasets. This proves the benefit of the supervision from the gold summary and the integration of contextual representations to exploit the better global semantics. It is also proved from the improvement of our method when compared with methods that encode full document contents but ignore the topic information, such as Longformer-Ext, SSN-DM et al.

Moreover, our methods and other PLMs-based methods that address the truncation issue to encode full contents, such as Longformer-Ext, SSN-DM et al, achieve better performance on all long document datasets, when comparing methods with the input length limit, such as BERTSum, and MatchSum. It shows that the content loss can inhibit their ability to model the contextual information in the document, which also limits the employment of the contextual representations from existing methods in the topic generation. On the contrary, for CNN/DM and PubMed-Short whose documents are relatively short, the improvement is insignificant, since truncating these short documents would not miss much context information.

Datasets	PubMed-Long			Arxiv		
Metrics	R-1	R-2	R-L	R-1	R-2	R-L
GRETEL	**48.20**†	**21.20**†	**43.16**†	**48.17**†	**20.31**†	**42.84**†
W/O HTE	45.97	20.13	40.22	45.44	16.53	40.15
W/O Topic	47.61	20.89	42.86	47.48	19.97	42.65
W/O Contras	47.65	20.96	42.92	47.95	20.02	42.76
W/O Context	48.01	20.99	43.08	47.89	20.13	42.77
W/O Document	47.61	21.02	43.09	48.11	20.18	42.80

Table 6: ROUGE F1 results of our model under different settings on PubMed-Long and Arxiv.

6.2 Ablation Study

We further verify the attribution of each component to the performance improvement of GRETEL in this section, as shown in Table 6. It presents the results of our method including 1) W/O HTE replacing HTE with RoBERTa, 2) W/O Topic removing the loss of neural topic modeling, 3) W/O Contras removing the graph contrastive loss, 3) W/O Context without considering the contextual representations from PLMs in generating document topic representations, and 4) W/O Document without the document transformer layer to propagate information between blocks. We can observe that each component contributes to the performance of the model to a different degree. Among all the components, HTE is the most important one for improvement, which shows the importance of encoder the full contents, when introducing contextual information into the topic modeling. However, our method still outperforms Topic-GraphSum even without HTE, which demonstrates the superiority of the guidance from the gold summary during the topic generation in our method.

Furthermore, we show the impacts of different topic numbers on the performance of GRETEL in Table 7 on both the long document dataset PubMed-Long dataset and the short document

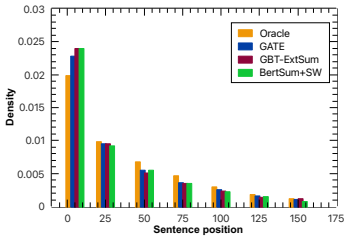

Figure 2: The position distribution of extracted sentences by different models on the PubMed-Long test set.

dataset CORD-19. It shows that the performance of our method generally increases with the growing number of topics on PubMed-Long, while it soon achieves the best of 300 topics on CORD-19 since it contains fewer topics with relatively shorter content and much fewer documents.

Datasets	**PubMed-Long**			**CORD-19**		
Metrics	R-1	R-2	R-L	R-1	R-2	R-L
K=100	47.10	20.14	42.19	43.53	15.92	38.89
K=200	47.53	20.38	42.42	43.80	16.25	39.41
K=300	47.94	20.89	42.94	**43.91**	**16.54**	**40.01**
K=400	48.15	21.12	43.24	43.85	16.42	39.94
K=500	**48.20**	**21.20**	**43.26**	43.86	16.45	40.01

Table 7: ROUGE F1 results of our model with a different number of topics on PubMed-Long and CORD-19.

6.3 Topic Analysis

To verify the quality of our generated topics, we further evaluate the NPMI (Lau et al., 2014) score in Table 8 by our methods and the classical topic model LDA. It clearly shows that our method can learn more coherent topics compared with LDA. Moreover, our method without contextual information (W/O Contextual) and the supervision (W/O Contras) all outperform LDA and underperforms our method with both features. It demonstrates that both the contextual information and the supervision from the gold summary are helpful to exploit meaningful and salient topics for a better approximation of global semantics.

Datasets	**CORD-19**		**PubMed-Long**	
Model	K=100	K=200	K=100	K=200
LDA	0.18	0.16	0.14	0.19
GRETEL	**0.25**	**0.21**	**0.23**	**0.26**
W/O Contras	0.23	0.20	0.22	0.24
W/O Contextual	0.20	0.18	0.18	0.20

Table 8: NPMI score of different models on CORD-19 and PubMed-Long using different numbers of topics.

Gold	A 53-year-old man with steroid dependent rheumatoid arthritis presented with fever and serious articular drainage. Oral antibiotics were initially prescribed. Subsequent hemodynamic instability was attributed to septic shock. Further evaluation revealed a pericardial effusion with tamponade. Pericardiocentesis of purulent fluid promptly corrected the hypotension. Proteus mirabilis was later isolated from both the infected joint and the pericardial fluid. This is the first report of combined proteus mirabilis septic arthritis and purulent pericarditis. It documents the potential for atypical transmission of gram-negative pathogens, to the pericardium, in patients with a high likelihood of preexisting pericardial disease. In immunocompromised patients, the typical signs and symptoms of pericarditis may be absent, and the clinical presentation of pericardial tamponade may be misinterpreted as one of septic shock. This case underscores the value of a careful physical examination and proper interpretation of ancillary studies. It further illustrates the importance of initial antibiotic selection and the need for definitive treatment of septic arthritis in immunocompromised patients.
Our	ID 3: We report a case of purulent pericarditis with pericardial tamponade masquerading as septic shock related to proteus mirabilis septic arthritis. ID 4: A 53-year-old man with long-standing, steroid-dependent rheumatoid arthritis complained of a painful, swollen, left elbow with purulent drainage emanating from what appeared to be a small ulceration. ID 0: Septic arthritis is a well recognized occurrence in patients with steroid dependent rheumatoid arthritis .1 treatment includes broad-spectrum antibiotics usually accompanied by surgical or needle drainage of the joint .2 while pericardial effusions are common in patients with rheumatologic disorders, the development of purulent pericarditis with pericardial tamponade is rare. ID 75: This case also underscores the importance of appropriate antibiotic selection in the initial treatment of immunocompromised patients with infected prosthetic joints. ID 30: While multiple blood cultures were negative, articular and pericardial fluid cultures grew staphylococcus epidermidis and proteus mirabilis. ID 68: In the setting of an infected joint prosthesis, fever, and immunosuppression, this patients hemodynamic instability was initially ascribed to septic shock and not to pericardial tamponade.
Baseline	ID 3: we report a case of purulent pericarditis with pericardial tamponade masquerading as septic shock related to proteus mirabilis septic arthritis. ID 4: A 53-year-old man with long-standing, steroid-dependent rheumatoid arthritis complained of a painful, swollen, left elbow with purulent drainage emanating from what appeared to be a small ulceration. ID 12: On the first postoperative day, he was transferred to the medical service for the management of presumed septic shock. ID 25: A subxiphoid pericardiocentesis yielded 500 ml of purulent fluid with prompt normalization of the blood pressure. ID 0: While multiple blood cultures were negative, articular and pericardial fluid cultures grew staphylococcus epidermidis and proteus mirabilis. ID 11: The early postoperative course, however, was remarkable for persistent fever, hypotension, and tachycardia.
Topics	T267: infected congenital patients loss blood disorders compared high fig phase T433: treatment infected severity year patients crp tract area arm isolated T446: joints dna physical risk observed tube lower examination intravenous features T153: disease type exercise vegf nerve deaths shock joints drugs lower T108: type treatment male sd well use statistically specific post mice

Table 9: Example of extractive summarization conducted by our method on the PubMed-Long dataset. The gold summary is the abstract of the document. Sentences with deep color have a higher ROUGE score. Topic words are marked with the blue color.

Sentence ID	Top-6 words
ID 3	adequate mice patients label understanding body infected report oxygen formation disease type exercise vegf nerve deaths shock joints drugs lower
ID 4	family report presented followed obesity side j macrophages high necrosis treatment infected severity year patients crp tract area arm isolated
ID 0	treatment infected severity year patients crp tract area arm isolated type treatment male sd well use statistically specific post mice
ID 75	treatment infected severity year patients crp tract area arm isolated treatment adjacent medication different motor min height stroke like rate
ID 30	et treatment lower diagnosis control observed could 7 number association infected congenital patients loss blood disorders compared high fig phase
ID 68	type treatment male sd well use statistically specific post mice effects performed revealed compared clinical observed diagnosis isolated cm family

Table 10: Top 10 words of top 2 topics in sentences, which are selected into the summary.

6.4 Case Study

In Table 9, we present the summaries of an example document generated by GRETEL and BERTSum, together with the top-5 topics (with the highest coherence) of the document from the PubMed-Long dataset. In table 10, we show the top-2 topics of selected sentences by our method for inclusion in the summary. It shows that our method generates a more coherent summary that contains more salient sentences than the summary generated by BERTSum, due to the integration of a better approximation of global semantics in our method. This is also proved by the selected sentences of our method are topically related to the captured topics about "treatment", "joints" and "infected", which are semantically similar to the meaning of the gold summary.

Moreover, the positions of our selected sentence vary in every part of the document while the sentences of BERTSum are all located in the former part of the document. This is because the employment of HTE allows our method to encode the full contents of the document without truncation. In Figure 2, we further compare the position distribution of selected sentences by different models and the oracle summary on PubMed-Long. The distribution of our method is the most similar to the oracle summary, which pays more attention to the latter sentences compared with other models.

7 Conclusion

In this paper, we propose a novel framework GRETEL for extractive summarization of long texts, that furnishes PLMs with the neural topic inference, to fully incorporate the local and global semantics. Experimental results on both general and biomedical datasets show that our model outperforms existing state-of-the-art methods, and global semantics empowered by graph contrastive learning and PLMs can yield more discriminative sentence representations to select salient sentences, that are topically similar to the gold summary. For future work, we would explore the feasibility of extending this framework to abstractive and multi-document summarization tasks.

Acknowledgments

This research is supported by the Alan Turing Institute and the Biotechnology and Biological Sciences Research Council (BBSRC), BB/P025684/1. We would like to thank Pan Du, Jennifer Bishop, and Guanghao Yang for their help and constructive comments.

References

Rahul Aralikatte, Shashi Narayan, Joshua Maynez, Sascha Rothe, and Ryan McDonald. 2021. Focus attention: Promoting faithfulness and diversity in summarization. *arXiv preprint arXiv:2105.11921*.

Iz Beltagy, Matthew E Peters, and Arman Cohan. 2020. Longformer: The long-document transformer. *arXiv preprint arXiv:2004.05150*.

Jennifer Bishop, Qianqian Xie, and Sophia Ananiadou. 2022. Gencomparesum: a hybrid unsupervised summarization method using salience. In *Proceedings of the 21st Workshop on Biomedical Language Processing*, pages 220–240.

Ziqiang Cao, Sujian Li, Yang Liu, Wenjie Li, and Heng Ji. 2015. A novel neural topic model and its supervised extension. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 29.

Jinjin Chi, Jihong Ouyang, Changchun Li, Xueyang Dong, Ximing Li, and Xinhua Wang. 2019. Topic representation: Finding more representative words in topic models. *Pattern recognition letters*, 123:53–60.

Arman Cohan, Franck Dernoncourt, Doo Soon Kim, Trung Bui, Seokhwan Kim, Walter Chang, and Nazli Goharian. 2018. A discourse-aware attention model for abstractive summarization of long documents. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 2 (Short Papers)*, pages 615–621.

Peng Cui and Le Hu. 2021a. Sliding selector network with dynamic memory for extractive summarization of long documents. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 5881–5891.

Peng Cui and Le Hu. 2021b. Topic-guided abstractive multi-document summarization. *arXiv preprint arXiv:2110.11207*.

Peng Cui, Le Hu, and Yuanchao Liu. 2020. Enhancing extractive text summarization with topic-aware graph neural networks. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 5360–5371.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. Bert: Pre-training of deep bidirectional transformers for language understanding. In *NAACL-HLT (1)*.

Xiyan Fu, Jun Wang, Jinghan Zhang, Jinmao Wei, and Zhenglu Yang. 2020. Document summarization with vhtm: Variational hierarchical topic-aware mechanism. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pages 7740–7747.

Quentin Grail, Julien Perez, and Eric Gaussier. 2021. Globalizing bert-based transformer architectures for long document summarization. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*, pages 1792–1810.

Thomas L Griffiths and Mark Steyvers. 2002. A probabilistic approach to semantic representation. In *Proceedings of the annual meeting of the cognitive science society*, volume 24.

Karl Moritz Hermann, Tomas Kocisky, Edward Grefenstette, Lasse Espeholt, Will Kay, Mustafa Suleyman, and Phil Blunsom. 2015. Teaching machines to read and comprehend. *Advances in neural information processing systems*, 28:1693–1701.

Luyang Huang, Shuyang Cao, Nikolaus Parulian, Heng Ji, and Lu Wang. 2021. Efficient attentions for long document summarization. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 1419–1436.

Nikita Kitaev, Lukasz Kaiser, and Anselm Levskaya. 2019. Reformer: The efficient transformer. In *International Conference on Learning Representations*.

Jey Han Lau, David Newman, and Timothy Baldwin. 2014. Machine reading tea leaves: Automatically evaluating topic coherence and topic model quality. In *Proceedings of the 14th Conference of the European Chapter of the Association for Computational Linguistics*, pages 530–539.

Chin-Yew Lin. 2004. Rouge: A package for automatic evaluation of summaries. In *Text summarization branches out*, pages 74–81.

Chin-Yew Lin and Eduard Hovy. 2003. Automatic evaluation of summaries using n-gram co-occurrence statistics. In *Proceedings of the 2003 human language technology conference of the North American chapter of the association for computational linguistics*, pages 150–157.

Yang Liu and Mirella Lapata. 2019. Text summarization with pretrained encoders. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 3730–3740.

Ye Liu, Jian-Guo Zhang, Yao Wan, Congying Xia, Lifang He, and Philip S Yu. 2021. Hetformer: Heterogeneous transformer with sparse attention for long-text extractive summarization. *arXiv preprint arXiv:2110.06388*.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Roberta: A robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692*.

Kyle Lo, Lucy Lu Wang, Mark Neumann, Rodney Kinney, and Daniel S Weld. 2020. S2orc: The semantic scholar open research corpus. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 4969–4983.

Ramesh Nallapati, Feifei Zhai, and Bowen Zhou. 2017. Summarunner: A recurrent neural network based sequence model for extractive summarization of documents. In *Thirty-First AAAI Conference on Artificial Intelligence*.

Shashi Narayan, Joshua Maynez, Jakub Adamek, Daniele Pighin, Blaz Bratanic, and Ryan McDonald. 2020. Stepwise extractive summarization and planning with structured transformers. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 4143–4159.

Thong Nguyen, Anh Tuan Luu, Truc Lu, and Tho Quan. 2021. Enriching and controlling global semantics for text summarization. *arXiv preprint arXiv:2109.10616*.

Min Peng, Qianqian Xie, Yanchun Zhang, Hua Wang, Xiuzhen Jenny Zhang, Jimin Huang, and Gang Tian. 2018. Neural sparse topical coding. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 2332–2340.

Akash Srivastava and Charles Sutton. 2017. Autoencoding variational inference for topic models. In *5th International Conference on Learning Representations*.

Mark Steyvers and Tom Griffiths. 2007. Probabilistic topic models. *Handbook of latent semantic analysis*, 427(7):424–440.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. *Advances in neural information processing systems*, 30.

Benyou Wang, Qianqian Xie, Jiahuan Pei, Prayag Tiwari, Zhao Li, et al. 2021. Pre-trained language models in biomedical domain: A systematic survey. *arXiv preprint arXiv:2110.05006*.

Lucy Lu Wang, Kyle Lo, Yoganand Chandrasekhar, Russell Reas, Jiangjiang Yang, Doug Burdick, Darrin Eide, Kathryn Funk, Yannis Katsis, Rodney Michael Kinney, et al. 2020a. Cord-19: The covid-19 open research dataset. In *Proceedings of the 1st Workshop on NLP for COVID-19 at ACL 2020*.

Zhengjue Wang, Zhibin Duan, Hao Zhang, Chaojie Wang, Long Tian, Bo Chen, and Mingyuan Zhou. 2020b. Friendly topic assistant for transformer based abstractive summarization. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 485–497.

Thomas Wolf, Julien Chaumond, Lysandre Debut, Victor Sanh, Clement Delangue, Anthony Moi, Pierric Cistac, Morgan Funtowicz, Joe Davison, Sam Shleifer, et al. 2020. Transformers: State-of-the-art natural language processing. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 38–45.

Qianqian Xie, Jennifer Amy Bishop, Prayag Tiwari, and Sophia Ananiadou. 2022. Pre-trained language models with domain knowledge for biomedical extractive summarization. *Knowledge-Based Systems*, page 109460.

Qianqian Xie, Jimin Huang, Pan Du, Min Peng, and Jian-Yun Nie. 2021. Graph topic neural network for document representation. In *Proceedings of the Web Conference 2021*, pages 3055–3065.

Jiacheng Xu, Zhe Gan, Yu Cheng, and Jingjing Liu. 2020. Discourse-aware neural extractive text summarization. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 5021–5031.

Xingxing Zhang, Furu Wei, and Ming Zhou. 2019. Hibert: Document level pre-training of hierarchical bidirectional transformers for document summarization. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 5059–5069.

Ming Zhong, Pengfei Liu, Yiran Chen, Danqing Wang, Xipeng Qiu, and Xuan-Jing Huang. 2020. Extractive summarization as text matching. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 6197–6208.

PINEAPPLE: Personifying INanimate Entities by Acquiring Parallel Personification Data for Learning Enhanced Generation

Sedrick Scott Keh [1], Kevin Lu [2], Varun Gangal*[1], Steven Y. Feng*[3], Harsh Jhamtani [1], Malihe Alikhani [4], Eduard Hovy [1]
[1]Carnegie Mellon University, [2]University of Waterloo,
[3]Stanford University, [4]University of Pittsburgh
{skeh,vgangal,jharsh,hovy}@cs.cmu.edu, syfeng@stanford.edu
kevin.lu1@uwaterloo.ca, malihe@pitt.edu

Abstract

A personification is a figure of speech that endows inanimate entities with properties and actions typically seen as requiring animacy. In this paper, we explore the task of personification generation. To this end, we propose **PINEAPPLE**: **P**ersonifying **IN**animate **E**ntities by **A**cquiring **P**arallel **P**ersonification Data for **L**earning **E**nhanced Generation. We curate a corpus of personifications called *PersonifCorp*, together with automatically generated de-personified literalizations of these personifications. We demonstrate the usefulness of this parallel corpus by training a seq2seq model to personify a given literal input. Both automatic and human evaluations show that fine-tuning with *PersonifCorp* leads to significant gains in personification-related qualities such as animacy and interestingness. A detailed qualitative analysis also highlights key strengths and imperfections of **PINEAPPLE** over baselines, demonstrating a strong ability to generate diverse and creative personifications that enhance the overall appeal of a sentence. [1]

1 Introduction

Personification is the attribution of animate actions or characteristics to an entity that is inherently inanimate. Consider, for example, the sentence *"The **stars danced playfully** in the moonlit sky."* Here, the vibrance of the stars (something inanimate) is being likened to dancing playfully, which is a distinctly animate action. By allowing readers to construct clearer mental images, personifications enhance the creativity of a piece of text (Bloomfield, 1980; Dorst, 2011; Flannery, 2016).

Being able to automatically identify and generate personifications is important for multiple reasons. First, humans naturally use personifications when communicating. When we say something like *"My phone has died,"* or *"My car is not cooperating,"* to a dialogue system, it is important that the dialogue system understands the intended meaning behind these personifications. If these systems interpret personifications literally, they may fail in several downstream tasks (e.g. classification) since their understanding is incorrect. Being able to generate personifications also allows dialogue agents and language models to be more creative and generate more figurative sentences. Personification generation has additional applications such as AI-assisted creative writing, since machine-generated figures of speech have been shown to enhance the interestingness of written text (Chakrabarty et al., 2021).

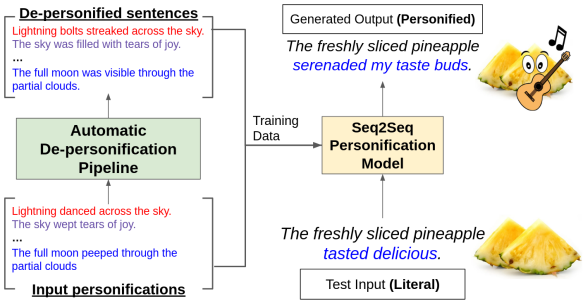

Figure 1: Overall **PINEAPPLE** model pipeline. The left part of the diagram shows the corpus creation process, while the right part of the diagram shows the training and generation process.

Despite previous success in generating other figures of speech such as similes (Chakrabarty et al., 2020), metaphors (Stowe et al., 2021), hyperboles (Troiano et al., 2018), irony (Van Hee et al., 2018), and sarcasm (Hazarika et al., 2018; Jaiswal, 2020), personification generation is relatively underexplored. One key challenge is that personifications do not have an explicit syntactic structure unlike similes which use *'like'* or *'as'*. They are also not as loosely-defined as metaphors. Rather, a personification requires identifying an inanimate subject together with actions or descriptions which are

* Equal contribution by Varun and Steven
[1]Data and code can be found at https://github.com/sedrickkeh/PINEAPPLE

Proceedings of the 29th International Conference on Computational Linguistics, pages 6270–6284
October 12–17, 2022.

commonly used on animate subjects. These steps are challenging and require our models to understand commonsense concepts including animacy.

In line with exploring the task of personification generation, we present three main contributions: (1) We curate a dataset, *PersonifCorp*, of diverse personification examples from various sources. (2) We propose a method called **PINEAPPLE** to automatically de-personify personifications and create a parallel corpus of personification data along with their literalizations. (3) Given our parallel corpus, we train a seq2seq model to personify given text. We conduct automatic and human evaluation and qualitative analysis of the generated outputs.

2 Datasets

We curate a dataset called *PersonifCorp* of 511 personifications, with 236 coming from a publicly available open-sourced list[2] and 275 manually-filtered personifications extracted from the Deja dataset (Chen et al., 2015). The Deja dataset is an image-captioning dataset containing a "figurative" subset of size 6000, of which 4.1% of the captions are labelled as personifications. We extract these personifications and combine them with our existing list to form the final *PersonifCorp* dataset.

We also note that although it is possible to further expand this dataset (e.g. by ad hoc searching for miscellaneous sites and examples online), we ultimately decide against this after performing an initial investigation. When we attempted to look for additional examples, we found that many of the new examples we found were near-duplicates of existing personifications already in our list. In addition, ad hoc searching can give at most a few hundred examples, which will lead to very incremental gains in performance. This is impractical if we want to collect a large-scale dataset. We hence decided to restrict ourselves to sentences from reasonably well-vetted, already existing corpora from *CL prior art or officially released data from sources like Kaggle/SemEval shared tasks.

2.1 Characterizing Personifications

We define the *elements of personification*, an analogue to what was previously done for similes (Niculae and Danescu-Niculescu-Mizil, 2014; Chakrabarty et al., 2020). While similes could be decomposed into very granular structures and well-defined elements, the unstructured nature of personifications prevents us from directly defining such fine-grained elements for personifications. Rather, we define two main high-level elements, the TOPIC (a noun phrase that acts as logical subject) and the ATTRIBUTE (the distinctly animate action or characteristic that is being ascribed to the TOPIC). Figure 2 shows examples of how these TOPICS and ATTRIBUTES can relate to each other.

2.2 Automatic Parallel Corpus Construction

In order to train a seq2seq model to generate high-quality personifications, we need pairs of personifications along with their corresponding literalizations. However, the literalization process may take several human-hours, which is impractical for large datasets. We therefore propose **PINEAPPLE**, a three-stage automatic de-personification process, where we first identify all valid TOPIC-ATTRIBUTE pairs, then generate multiple candidates to replace the ATTRIBUTE of each TOPIC. Lastly, we select the most appropriate candidate in terms of animacy, fluency, and meaning preservation. These steps are further detailed individually below:

TOPIC-ATTRIBUTE Extraction. To identify the TOPICS and ATTRIBUTES, we consider the dependency parse tree of a sentence and the part-of-speech (POS) tags of each of its words. Given the tree, we extract all the nouns/pronouns which have edges pointing into it with the *nominal subject* label, together with the corresponding parent nodes. For instance, in the sentence *"The stars danced in the night sky"*, the word *'danced'* is a parent of the word *'stars'*, with the *nominal subject* edge relationship. We can thus identify *'stars'* as the TOPIC and *'danced'* as the ATTRIBUTE. In more complex scenarios, we may need to perform some additional merging to deal with compound multi-word TOPICS and ATTRIBUTES, as well as any additional modifiers. More specifically, using the POS tags, we identify all words tagged as *negation modifiers*, *possession modifiers*, *nominal modifiers*, *adjectival complements*, and *objects of prepositions*, and words tagged as determiners and parts of compound phrases.[3] After extracting these nodes, they are iteratively merged with their parents in the dependency parse tree, and the merging process is performed repeatedly until no more merges are possible. The final TOPIC-ATTRIBUTE pairs

[2] https://www.kaggle.com/datasets/varchitalalwani/figure-of-speech

[3] The spaCy library was used to extract the dependency tree and POS tags.

ATTRIBUTE Type	Example
Noun	The planet earth is our mother.
Verb	My alarm clock yells at me to get out of bed every morning.
Adjective	Justice is blind and, at times, deaf.

Figure 2: Examples of different types of personification ATTRIBUTES (TOPICS in red and ATTRIBUTES in blue).

are then identified using the *nominal subject* edge relationship as previously described. Examples of the merging process can be found in Appendix A.1.

Candidate Generation. Once the TOPIC-ATTRIBUTE pairs have been identified, we then determine which TOPICS are inanimate. To achieve this, we need some type of commonsense notion of what constitues animacy. We use COMET (Bosselut et al., 2019) to tap into the commonsense knowledge present in large-scale knowledge graphs such as ConceptNet (Speer et al., 2017). Although ConceptNet has no explicit notion of animacy, it has certain edge relations that we can leverage to design a proxy metric. More specifically, we use the *IsA* relation to design a custom *IsAPerson* animacy metric. If the TOPIC of our sentence refers to an animate entity, then we expect its *IsA* relation score with the word *'human'* to be relatively low.[4] The *IsAPerson* metric is hence defined as follows: given a TOPIC, we compute and average its *IsA* scores to various words that are synonymous or very closely related to *'human'*, such as *'person'*, *'man'*, and *'woman'*. We call this set of *'human'*-related words the HUMANSET. The construction and full list of words in the HUMANSET can be found in Appendix A.2. The average of these ConceptNet scores is then our final *IsAPerson* animacy score.

Phrases whose *IsAPerson* animacy score exceeds a certain threshold [5] are considered animate; otherwise, they are considered inanimate. Since our goal is to de-personify a sentence, we can safely discard all the animate TOPICS, as these need no further de-personification. Rather, we focus on the inanimate TOPICS because the segment we want to de-personify most likely occurs in the TOPIC-ATTRIBUTE pairs whose TOPIC is inanimate. Once we identify all such inanimate TOPIC-ATTRIBUTE pairs, we mask out the ATTRIBUTE of each of them with <mask>, then use a pre-trained BART model (Lewis et al., 2020) to generate the top $k = 10$ candidates for each mask using beam search with a beam size of 10. The goal of this process is to replace a possibly animate action/characteristic with candidates that are inanimate.

Candidate Selection. Given $k = 10$ candidate replacement ATTRIBUTES, we now select the most ideal replacement based on three metrics: animacy, fluency, and meaning preservation.

1. Animacy – We want the replacement ATTRIBUTE to be inanimate; otherwise we would just be replacing an animate ATTRIBUTE with another animate ATTRIBUTE. We define the animacy of a TOPIC-ATTRIBUTE pair as difference between the affinity for a human ($\mathcal{A}_{human,\text{ATT}}$) to do/possess the ATTRIBUTE, and the affinity for the given TOPIC ($\mathcal{A}_{\text{TOPIC,ATT}}$) to do/possess the ATTRIBUTE. We use COMET's ConceptNet relations to compute these affinities; specifically, we use the *CapableOf* relation. To approximate $\mathcal{A}_{human,\text{ATT}}$, we compute the average *CapableOf* score between the given ATTRIBUTE and all words in our previously defined HUMANSET. To compute $\mathcal{A}_{\text{TOPIC,ATT}}$, we compute the *CapableOf* score between the TOPIC and its ATTRIBUTE. The final animacy score of a TOPIC-ATTRIBUTE pair is defined as the difference $\mathcal{A}_{human,\text{ATT}} - \mathcal{A}_{\text{TOPIC,ATT}}$. If there are multiple TOPIC-ATTRIBUTE pairs, we consider the average animacy of all pairs.

2. Fluency – The de-personified sentences should be grammatically correct and sound natural. To measure for fluency, we use BART's generation scores (i.e. sum of individual token logits in the generated output).

3. Meaning Preservation – It is important that the de-personified sentence does not stray too far from the meaning of the original personification. We use BERTScore (Zhang* et al., 2020) between the de-personified and original sentences to measure meaning preservation.

We design a composite scoring metric comprised of the aggregate scores from these 3 metrics. Due to scaling differences, we consider the log of the animacy score. To account for the fact that lower animacy scores imply less animate TOPIC-ATTRIBUTE pairs (which is desirable in de-personification), we

[4] For the COMET ConceptNet graph, lower scores correspond to better matches.

[5] We use a threshold of 7.0 for the *IsAPerson* animacy metric. *IsAPerson* scores < 7.0 are considered animate, while scores ≥ 7.0 are considered inanimate. More details regarding the selection of this threshold can be found in Appendix A.3.

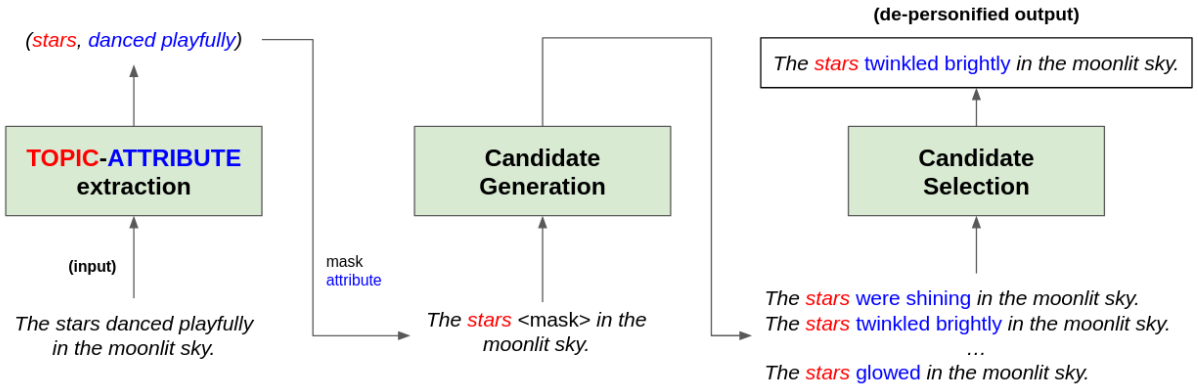

Figure 3: Overview of the **PINEAPPLE** de-personification pipeline.

Original Personification	Result After De-Personifying
How far that little candle throws its beams!	How far that little candle can spread its beams!
A book is a fragile creature, it suffers the wear of time, it fears rodents, the elements and clumsy hands.	A book is fragile, it can break from the wear of time, it can be eaten by rodents, the elements and clumsy hands.
The camera loves her since she is so pretty.	The camera is always on her since she is so pretty.
Any trust I had for him walked right out the door.	Any trust I had for him had gone right out the door.
The full moon peeped through partial clouds.	The full moon was visible through partial clouds.
Opportunity was knocking at her door.	Opportunity was knocking at her door.
The killing moon will come too soon.	The killing moon will be here too soon.

Table 1: Example outputs of the **PINEAPPLE** de-personification pipeline. The ATTRIBUTES are highlighted in blue for both the original personifications, as well as the de-personified output sentences. The last two rows contain negative examples where the process does not successfully de-personify the input.

take the negative of the animacy. More precisely, we define our candidate score S_i for candidate i as

$$S_i = \alpha \cdot (-\log(S_{anim.})) + \beta \cdot S_{flue.} + \gamma \cdot S_{mean.}$$

where α, β, γ are parameters. [6]

Once S_i is computed for all candidates, we select the candidate with the highest composite score as our final de-personified sentence. A diagram of the entire **PINEAPPLE** pipeline is shown in Figure 3, and example outputs can be found in Table 1.

2.3 Test Data Construction

While automatically generated pairs of personifications and literal de-personifications may greatly assist with training, these may not necessarily be accurate for testing. Rather, it would be more ideal during testing if we have ground-truth human-annotated data. To mimic our task at hand, we gather a list of non-personified English sentences.[7] We then select two annotators who are native English speakers currently enrolled in a university with English as a medium of instruction. These annotators were instructed to manually personify these sentences to create ground-truth reference personifications. The final *PersonifCorp* test split has 72 literal + personified sentence pairs.

3 Experimental Setup

3.1 Methods

Below we outline the three models we consider, with two of them being naive baselines (COMET and Baseline-BART) that we simply use on *PersonifCorp*'s test set, and the third (Finetuned-BART) being our proposed model trained on *PersonifCorp*.

1. **COMET**: We extract the TOPIC-ATTRIBUTE pairs and identify the inanimate TOPICS using the methods detailed in §2.2. Instead of generating candidate replacements using BART like in §2.2, we generate candidates by considering the top $k = 10$ results for a given TOPIC using COMET's ConceptNet *IsCapable* relation (if the original ATTRIBUTE is a verb) or *HasProperty* relation (if adjective or adverb). To incorporate a notion of animacy, we use the previously defined ATTRIBUTE animacy $\mathcal{A}_{human,\text{ATT}}$ and select the candidate

[6] We use $\alpha = 1$, $\beta = 1$, $\gamma = 1$. Details about the tuning and selection of α, β, γ can be found in Appendix A.3.

[7] https://github.com/tuhinjubcse/SimileGeneration-EMNLP2020#set-up-data-processing-for-simile

with highest animacy as our final replacement.

2. **Baseline-BART (BL-BART)**: We imitate the process outlined for the COMET baseline, except we use a pretrained BART model to generate the candidates instead of using COMET. All other steps (TOPIC-ATTRIBUTE extraction and candidate selection) remain the same.

3. **PINEAPPLE-BART (PA-BART)**: We finetune a BART model by supplying the *PersonifCorp* train split literal de-personified sentences (from the **PINEAPPLE** pipeline) as inputs, and the original ground-truth personifications as target outputs. This is trained as a seq2seq task. During generation, we use beam search. Further details are outlined in §3.3.

3.2 Evaluation

We consider both automatic evaluation metrics (§3.2.1) and human evaluation (§3.2.2).

3.2.1 Automatic Evaluation

For each model in §3.1, we evaluate its generated outputs on *PersonifCorp*'s test split using each of the following automatic evaluation metrics:

1. **BLEU** (Papineni et al., 2002): We use BLEU to ensure that the generations do not greatly differ from the inputs. We compute the BLEU score of each generated output with the literal inputs (for meaning preservation), as well as the ground-truth reference personifications.

2. **BERTScore** (Zhang et al., 2019): BERTScore measures how semantically related two sentences are, and is generally more robust than BLEU. We compute the BERTScore of each generated output with the inputs, as well as the ground-truth reference personifications.

3. **Fluency**: To approximately measure the fluency of a sentence, we use generation (log-perplexity) losses of each output using the GPT-2 language model (Radford et al., 2019).

4. **Animacy**: We are interested in how *personified* the generated output is. We use the same animacy metric used for candidate selection in §2, which is a combination of how animate the ATTRIBUTE is, as well as how inanimate the TOPIC is. More precisely, this is defined as $\mathcal{A}_{human,\text{ATT}} - \mathcal{A}_{\text{TOPIC,ATT}}$, where the $\mathcal{A}$ animacy scores are previously defined in §2.

3.2.2 Human Evaluation

The human evaluation was conducted using paid annotators on Amazon Mechanical Turk (AMT). Annotators were from Anglophone countries with $>$ 97% approval rate.[8] Each test example was evaluated by exactly 2 annotators. For each test example, we first generate outputs using each of the methods outlined in §3.1. Corresponding to this test instance, we then create an AMT task page (a HIT), presenting the input literal sentence and each of the method outputs (in randomized order) for annotation along five aspects of text quality.

Specifically, annotation was elicited for the following metrics: **(1) Personificationhood** (*"To what extent does the new sentence contain a personification?"*), **(2) Appropriateness** (*"Do the personified nouns, verbs, adjectives, adverbs sound mutually coherent and natural?"*), **(3) Fluency** (*"Does it sound like good English with good grammar?"*), **(4) Interestingness** (*"How interesting and creative a rephrasing of the original sentence is the personified sentence?"*), and **(5) Meaning Preservation** (*"Do the entities, their actions, interactions, and the events appear and relate to each other in the same way as in the original sentence?"*). Each metric was scored on a Likert scale, with 1 being the lowest and 5 being the highest.

For *Interestingness*, we observed poor agreement scores amongst the AMT annotators.[9] Hence, for this aspect, we instead used a curated group of known, in-person annotators: a cohort of three native English-speaking students from an American university. Amongst these annotators, we observe a considerably higher agreement, with a Krippendorff α value of 0.5897. For selecting this cohort from a slightly larger pool of candidates, we assessed their performance on a short qualification test of basic English literary skills and knowhow. The final cohort chosen each scored 85% or higher on this test. Further details are in Appendix B.3.

3.3 Implementation Details

The *PersonifCorp* training corpus was randomly split into a training and validation split with an 80-20 ratio. We fine-tune a BART-base model with 139M parameters using a learning rate of 2e-5 and a batch size of 4. Training was done for 20 epochs and 400 warmup steps, and model/epoch selection

[8] More details about the human eval are in Appendix B.1.

[9] Further details on inter-annotator agreement scores can be found in Appendix B.2.

was performed based on the lowest validation loss. For generating the outputs, decoding was done using beam search with a beam size of 10. Additional details can be found in Appendix C.

4 Results and Analysis

4.1 Automatic Evaluation Results

Table 2 reports the automatic evaluation results for each of the metrics detailed in §3.2.1. We observe that our PA-BART model performs best across all automatic metrics except for fluency, where BL-BART performs best. The difference in performance is most significant in the *Animacy* metric, which is the key metric that quantifies the degree to which a sentence is personified. This confirms that indeed, our proposed **PINEAPPLE** method is successful in training a model to personify text.

Our PA-BART model also performs well for both BLEU and BERTScore, scoring better than the COMET and BART baselines, and coming second only to the human-written personifications.

Lastly, with regards to fluency, the BL-BART model outperforms the PA-BART model. This is likely because when considering GPT-2 likelihood, it may unfavorably penalize creative sentences with personifications since these are naturally less common in regular text. As an example, the sentence *"The stars danced playfully"* (GPT-2 loss = 7.02) would be deemed significantly less fluent than the sentence *"The stars twinkled brightly"* (GPT-2 loss = 5.24), even though they are both valid sentences with similar meanings. This argument is further supported by the fact that even the reference human-generated personifications received a lower fluency score than the BL-BART outputs. Further, literal sentences are indeed typically more *fluent* overall than personifications since they express the meaning literally. Nevertheless, we are still interested in the other qualities being measured by fluency: *Is the sentence coherent? Does it make unnecessary grammatical errors?* In this regard, the fluency of PA-BART remains quite good. It is significantly better than the fluency of the COMET personifications and only slightly worse than the fluency of the human-written personifications.

4.2 Human Evaluation Results

Human evaluation results are reported in Table 3. Out of the five human evaluation metrics, the most pertinent metric to the personification generation task is *Personificationhood*, as this metric explicitly tries to quantify the presence and overall quality of personifications. In this metric, our PA-BART model performs significantly better than both baselines and is only slightly worse than the human reference personifications. This indicates that PA-BART is very successful in generating personifications that humans are able to detect and understand.

Aside from measuring the presence of personifications, we also want to measure more fine-grained qualities of these personifications. This is done by considering the *Appropriateness* and *Interestingness* scores. In *Interestingness*, PA-BART significantly outperforms both baselines but is worse than human annotations, while in *Appropriateness*, PA-BART slightly outperforms BL-BART and is slightly worse than human annotations. Overall, we can conclude that the personifications generated by PA-BART are of good quallity: the ATTRIBUTES match up well with the TOPICS, and they are overall very creative. This is further exemplified through the qualitative examples explored in §4.3.

Observations from *Meaning Preservation* and *Fluency* are very similar to those from the BLEU/BERTScore/Fluency metrics in the automatic evaluations. For *Meaning Preservation*, PA-BART performs best among all models, and only slightly trails human references. Meanwhile, for fluency, BL-BART was ranked the most fluent, outperforming both PA-BART and the human references. As discussed previously, this is likely due to the fact that literal sentences are generally perceived to be more fluent than personifications.

4.3 Qualitative Analysis

Table 4 contains a list of color-coded qualitative examples for each method. In Figure 2, we previously outlined three main types of personification TOPIC-ATTRIBUTE pairs, namely the cases where ATTRIBUTE is a noun, a verb, and an adjective. The first three examples in Table 4 demonstrate the capacity of our PA-BART model to capture all three cases. In the first example, the literal verb in "*your phone **rings** out loud*" is replaced with the more appropriate animate verb in "*your phone **yells** out loud*." In the second, "*silence is **key***" is replaced with a noun in "*silence is **a ghost***", while in the third example, the literal adjective "*very **difficult***" is replaced with the animate adjective "*very **lonely***". These examples illustrate the generative flexibility of our model and its capacity to generate diverse outputs with different parts-of-speech.

	BLEU		BERTScore			
	Input	Gold	Input	Gold	Fluency ↓	Animacy
Human Annotation	0.172	1.000	0.749	1.000	5.264	0.332
COMET	0.127	0.128	0.655	0.569	6.366	-2.028
BL-BART	0.132	0.133	0.728	0.617	**4.573**	0.106
PA-BART	**0.153**	**0.160**	**0.748**	**0.636**	5.460	**0.679**

Table 2: Average automatic evaluation results. The best-scoring method for each metric is highlighted in **bold**. Higher scores are better for all metrics except for fluency.

	Personificationhood	Appropriateness	Fluency	Interestingness	Meaning Preservation
Human Annotation	3.763	4.175	4.138	3.667	3.913
COMET	3.525	3.563	3.738	1.801	3.550
BL-BART	3.500	3.938	**4.188**	2.006	3.750
PA-BART	**3.738**	**4.000**	4.138	**2.782**	**3.875**

Table 3: Average human evaluation results. The best-scoring method for each metric is highlighted in **bold**.

We also observe that the outputs for PA-BART generally capture the meaning of the original text (and surrounding context) more accurately than the other baselines. In fact, the personifications greatly enhance the expressiveness of some of these sentences. In the first example, PA-BART replaces '*rings*' with '*yells*', while COMET replaces it with '*beeps*', and BL-BART leaves '*rings*' unchanged and just adds more details. Given the context of the sentence, we see that '*yells*' is more appropriate, expressive, and consistent with the context. A similar argument can be made for most of the other examples in the table: for the third example, PA-BART replaces the literal *"very **difficult**"* with the much more animate and expressive *"very **lonely**"*, which is a suitable word to describe a relationship. In the fourth example, the BL-BART model is able to successfully capture the meaning of *"the house **became silent**"* with *"the house **fell into disrepair**"*. Although the meaning is correct, *"fell into disrepair"* is more literal and does not contain a personification. Compare this with the PA-BART's choice to replace *"the house **became silent**"* with *"the house **lamented**"*, which fits with the overall context (*"Then there were no more parties..."*), and also greatly enhances creativity by invoking the vivid image of lamentation. Meanwhile, in the fifth example, BL-BART personifies *"the crickets were **silent**"* with *"the crickets were **calling**"*. However, this shift completely changes the meaning, so it is a rather poor choice of personification. In contrast, PA-BART rewrites *"the air was **still**"* as *"the air was **tired**"*, which is a reasonable personification that is consistent with the imagery in the sentence (*"moonless nights"*, *"crickets were silent"*). Hence, we see that PA-BART can generate creative and meaningful personifications, while simultaneously staying true to the spirit of the sentence.

We also point out that our model is not limited to single-word substitutions. Rather, it considers a holistic view of the entire sentence and modifies key segments as necessary. This allows PA-BART to handle compound phrases well: consider, for instance, the one-to-many-word substitution of *'key'* $\longrightarrow$ *'a ghost'* (example 2), and the many-to-one-word substitution of *"became silent"* $\longrightarrow$ *"lamented"* (example 4). More importantly, PA-BART is also able to simultaneously generate personifications in two disjoint parts of the sentence, as seen in the last example: "*The sound **clapped** loud enough to make your ear **cry***." Here, there are two personifications in *"sound **hit**"* $\longrightarrow$ *"sound **clapped**"*, and *"ear **hurt**"* $\longrightarrow$ *"ear **cry**"*.

This last example also demonstrates the imperfection of our method. Although the model is able to generate two personifications, it loses a component of the original sentence because the recipient of the action (*'Frank'*) has disappeared. This same issue of meaning or information loss is present in example 2, where our model's output of *"silence is **a ghost**"*, while a personification, actually contradicts the original text *"silence is **key**"*. BL-BART's output of *"silence is **preferred**"*, while not a personification, correctly preserves the original meaning, as does the human reference of *"silence is **the protagonist**"*. This suggests that the model may still need some improvements with balancing creativity and semantic preservation. Other possible weaknesses are outlined in §6.

4.3.1 Novelty and Diversity Analysis

We randomly sample 30 examples from the PA-BART generations- and manually identify the parts

Method	Text
Literal Input	You are at a business dinner with your boss when your phone rings out loud (ex.1)
Human Ref	You are at a business dinner with your boss when your phone starts singing out loud
COMET	You are at a business dinner with your boss when your phone beep out loud
BL-BART	You are at a business dinner with your boss when your phone rings and you answer out loud
PA-BART	You are at a business dinner with your boss when your phone yells out loud
Literal Input	In most horror settings, silence is key. (ex.2)
Human Ref	In most horror settings, silence is the protagonist.
COMET	In most horror settings, silence scary.
BL-BART	In most horror settings, silence is preferred.
PA-BART	In most horror settings, silence is a ghost.
Literal Input	Her relationships with family and friends are very difficult (ex.3)
Human Ref	Her relationships with family and friends behave very unusually
COMET	Her relationships with family and friends serious
BL-BART	Her relationships with family and friends have always been strong.
PA-BART	Her relationships with family and friends are very lonely
Literal Input	Then there weren't any more parties as the house became silent (ex.4)
Human Ref	Then there weren't any more parties as the house kept mum.
COMET	Then there weren't any more parties as the house build.
BL-BART	Then there weren't any more parties as the house fell into disrepair.
PA-BART	Then there were no more parties as the house lamented.
Literal Input	It was a moonless nights, the air was still and the crickets were silent (ex.5)
Human Ref	It was a moonless nights, the air was asleep and the crickets were silent
COMET	It cold outside a moonless nights, the air cold outside still and the crickets noisy
BL-BART	It was a moonless nights, the air was still and the crickets were calling.
PA-BART	It was one of those moonless nights, the air was tired and the crickets were silent
Literal Input	The sound hit Frank loud enough to make your ear hurt (ex.6)
Human Ref	The sound slapped Frank loud enough to make your ear hurt
COMET	The sound echo Frank loud enough to make your ear sense sound
BL-BART	The sound of Frank Sinatra is loud enough to make your ear ring.
PA-BART	The sound clapped loud enough to make your ear cry

Table 4: Qualitative examples for personification: literal input, human writing, COMET, BL-BART, and PA-BART. More can be found in Appendix D.

of the sentences that were personified, as well as the animate ATTRIBUTES used to personify the TOPICS. Among the 30 examples, there were 27 unique ATTRIBUTES, and only 3 repeats. Additionally, there were 9 examples which generated completely new ATTRIBUTES that were never before seen in the training set, which demonstrates that the model is able to sufficiently capture the essence of a personification, rather than just blindly memorizing ATTRIBUTES from the training data.

5 Related Work

We present the linguistic underpinnings behind the TOPIC-ATTRIBUTE framework used in this paper and explore how other types of figures of speech are generated. We also explore what makes personification generation so challenging.

Linguistic Motivations. Personifications traditionally do not have clearly defined classifications. In fact, even within the linguistic community, the definition of a personification is not always very clear-cut (Edgecombe, 1997; Hamilton, 2002). A study by Long (2018) examines the personification structure "*nonhuman subject + predicate verb (used for humans only) + others*," as well as the structure "*others + predicate verb (used for humans only) + nonhuman object + others*." We generalize and repackage these concepts, renaming the *subject* as the TOPIC and the *predicate verb* as the ATTRIBUTE. In doing so, we are able to capture more general notions of animacy beyond just verbs.

Generation of Metaphors, Similes, etc. A lot of studies on metaphors have focused on identification using techniques like word sense disambiguation (Birke and Sarkar, 2007), topic modeling (Strzalkowski et al., 2013; Heintz et al., 2013), dependency structures (Jang et al., 2015), and semantic analysis (Hovy et al., 2013). In terms of generation, early systems have explored grammar rules (Gargett and Barnden, 2013), while more recently, large language models have greatly aided in metaphor generation. Most notably, Stowe et al. (2021) generate metaphors by considering conceptual mappings between certain domains and verbs. Chakrabarty et al. (2021) further build on this by creating a parallel corpus of metaphors and training a large language model to perform the generation.

We also note here that the two aforementioned studies already cover personifications to a certain extent. However, these studies considered personifications as subtypes of metaphors. Some of the methods used may not generalize well to other types of personifications. Our study is the first to focus specifically on generating personifications.

For generating similes, Chakrabarty et al. (2020) propose using style-transfer models with COMET commonsense knowledge to generate similes. The study similarly creates a parallel corpus and trains a seq2seq model to perform the generation.

There is also a recent work by Keh et al. (2022) that uniquely investigates the generation of tongue twisters using seq2seq and language models.

Personifications. There are currently few studies that specifically work on personifications. Gao et al. (2018) detect personifications as a subtype of metaphors, but not as its own figure of speech. Generation is largely unexplored. We believe this is likely because personifications are generally more difficult to define and categorize. Furthermore, because several sources simplify personifications to fall under metaphors (Stowe et al., 2021; Chakrabarty et al., 2021), there is also a lack of personification-specific datasets.

Constrained Text Generation. There is also a body of work exploring the family of more gen-

eral constrained text generation tasks. Gangal et al. (2022) investigate NAREOR, or narrative ordering, which rewrites stories in distinct narrative orders while preserving the underlying plot. Miao et al. (2019) show gains on several tasks through determining Levenshtein edits per generation step using Metropolis-Hastings sampling. Feng et al. (2019) propose Semantic Text Exchange to modify the topic-level semantics of a piece of text.

Lin et al. (2020) propose CommonGen, a generative commonsense reasoning task based on concept-to-text generation. Works investigating this task include EKI-BART (Fan et al., 2020) and KG-BART (Liu et al., 2021), which use external knowledge to enhance performance on CommonGen. SAPPHIRE (Feng et al., 2021b) uses the data itself and the model's own generations to improve CommonGen performance, while VisCTG (Feng et al., 2022) uses per-example visual grounding.

6 Conclusion and Future Work

In this paper, we explored the task of personification generation. We curated a dataset of personifications and proposed the **PINEAPPLE** method to automatically de-personify text. Using our parallel corpus, *PersonifCorp*, we trained a seq2seq model (BART) to generate creative personifications. Through automatic, human, and qualitative evaluation, we demonstrated that these personifications make sentences more interesting and enhance the text's overall appeal. Our finetuned model successfully does this while maintaining a high level of fluency and meaning perservation.

Some weaknesses of our model include failing to personify more complex sentence structures, and occasionally failing to preserve the exact meaning of the original sentence. We also believe that our model still has room to grow in terms of the diversity of personifications generated. Further, we can explore unsupervised style transfer methods (Yang et al., 2018; Malmi et al., 2020; Krishna et al., 2020), where we regard the personification-hood of a sentence as a kind of style. We can also investigate data augmentation methods (Feng et al., 2021a, 2020; Dhole et al., 2021) to further expand our dataset. Another promising direction would be to explore ways to acquire more control over which parts of the sentence are personified or what types of personifications are generated, or to apply this to make dialogue agents more interesting, e.g. by giving them more personality (Li et al., 2020).

Ethics Statement

Our human and automatic evaluations (see §3.2) are done over content either directly sourced from, or generated by publically available, off-the-shelf pretrained models trained either on already existing, publicly available datasets, or datasets further derived by post-processing the same — as further described in Datasets (see §2 for more).

We do collect human evaluation ratings using crowd-sourcing, specifically through AMT and in-person annotation. However, we neither solicit, record, nor request any kind of personal or identity information from the annotators. Our AMT annotation was conducted in a manner consistent with terms of use of any sources and intellectual property and privacy rights of AMT crowd workers. Crowdworkers were fairly compensated: $1.12 per fluency + appropriateness + meaning preservation evaluation HIT, and $0.56 per personificationhood evaluation HIT, for roughly 6 min (first) and 2 min (latter) tasks, respectively. This is at least 1.5-2 times the minimum U.S.A. wage of $7.25 per hour ($0.725 per 6 minutes and $0.25 per 2 minutes).

We primarily perform experiments on personification in English (Bender and Friedman, 2018).

NLG models are known to suffer from biases learnable from training or finetuning on data, such as gender bias (Dinan et al., 2020). However, our work and contribution does not present or release any completely new model architechtures, and is primarily concerned with more careful adaptation and finetuning of existing pretrained models for a particular class of figurative construct (i.e. personification). The frailties, vulnerabilities, and potential dangers of these models have been well researched and documented, and a specific re-investigation would be repetitive and beyond the scope and space constraints of this paper.

We do not foresee any explicit way that malicious actors could specifically misuse fintuned models that could be trained on our data, beyond the well-researched, aforementioned misuse that is possible in general with their instantiation for any transduction task or dataset (e.g. summarization).

References

Emily M Bender and Batya Friedman. 2018. Data statements for natural language processing: Toward mitigating system bias and enabling better science. *Transactions of the Association for Computational Linguistics*, 6:587–604.

Julia Birke and Anoop Sarkar. 2007. Active learning for the identification of nonliteral language. In *Proceedings of the Workshop on Computational Approaches to Figurative Language*, pages 21–28, Rochester, New York. Association for Computational Linguistics.

Morton W Bloomfield. 1980. Personification-metaphors. *The Chaucer Review*, pages 287–297.

Antoine Bosselut, Hannah Rashkin, Maarten Sap, Chaitanya Malaviya, Asli Celikyilmaz, and Yejin Choi. 2019. COMET: Commonsense transformers for automatic knowledge graph construction. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 4762–4779, Florence, Italy. Association for Computational Linguistics.

Tuhin Chakrabarty, Smaranda Muresan, and Nanyun Peng. 2020. Generating similes effortlessly like a pro: A style transfer approach for simile generation. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 6455–6469, Online. Association for Computational Linguistics.

Tuhin Chakrabarty, Xurui Zhang, Smaranda Muresan, and Nanyun Peng. 2021. MERMAID: Metaphor generation with symbolism and discriminative decoding. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 4250–4261, Online. Association for Computational Linguistics.

Jianfu Chen, Polina Kuznetsova, David Warren, and Yejin Choi. 2015. Déjà image-captions: A corpus of expressive descriptions in repetition. In *HLT-NAACL*, pages 504–514.

Kaustubh D Dhole, Varun Gangal, Sebastian Gehrmann, Aadesh Gupta, Zhenhao Li, Saad Mahamood, Abinaya Mahendiran, Simon Mille, Ashish Srivastava, Samson Tan, et al. 2021. Nl-augmenter: A framework for task-sensitive natural language augmentation. *arXiv preprint arXiv:2112.02721*.

Emily Dinan, Angela Fan, Adina Williams, Jack Urbanek, Douwe Kiela, and Jason Weston. 2020. Queens are powerful too: Mitigating gender bias in dialogue generation. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 8173–8188.

Aletta G Dorst. 2011. Personification in discourse: Linguistic forms, conceptual structures and communicative functions. *Language and Literature*, 20(2):113–135.

Rodney Stenning Edgecombe. 1997. Ways of personifying. *Style*, 31(1):1–13.

Zhihao Fan, Yeyun Gong, Zhongyu Wei, Siyuan Wang, Yameng Huang, Jian Jiao, Xuanjing Huang, Nan

Duan, and Ruofei Zhang. 2020. An enhanced knowledge injection model for commonsense generation. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 2014–2025, Barcelona, Spain (Online). International Committee on Computational Linguistics.

Steven Y. Feng, Varun Gangal, Dongyeop Kang, Teruko Mitamura, and Eduard Hovy. 2020. GenAug: Data augmentation for finetuning text generators. In *Proceedings of Deep Learning Inside Out (DeeLIO): The First Workshop on Knowledge Extraction and Integration for Deep Learning Architectures*, pages 29–42, Online. Association for Computational Linguistics.

Steven Y. Feng, Varun Gangal, Jason Wei, Sarath Chandar, Soroush Vosoughi, Teruko Mitamura, and Eduard Hovy. 2021a. A survey of data augmentation approaches for NLP. In *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*, pages 968–988, Online. Association for Computational Linguistics.

Steven Y. Feng, Jessica Huynh, Chaitanya Prasad Narisetty, Eduard Hovy, and Varun Gangal. 2021b. SAPPHIRE: Approaches for enhanced concept-to-text generation. In *Proceedings of the 14th International Conference on Natural Language Generation*, pages 212–225, Aberdeen, Scotland, UK. Association for Computational Linguistics.

Steven Y Feng, Aaron W Li, and Jesse Hoey. 2019. Keep calm and switch on! preserving sentiment and fluency in semantic text exchange. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 2701–2711.

Steven Y. Feng, Kevin Lu, Zhuofu Tao, Malihe Alikhani, Teruko Mitamura, Eduard Hovy, and Varun Gangal. 2022. Retrieve, caption, generate: Visual grounding for enhancing commonsense in text generation models. *Proceedings of the AAAI Conference on Artificial Intelligence*, 36(10):10618–10626.

Mary C Flannery. 2016. Personification and embodied emotional practice in middle english literature. *Literature Compass*, 13(6):351–361.

Varun Gangal, Steven Y. Feng, Malihe Alikhani, Teruko Mitamura, and Eduard Hovy. 2022. Nareor: The narrative reordering problem. *Proceedings of the AAAI Conference on Artificial Intelligence*, 36(10):10645–10653.

Ge Gao, Eunsol Choi, Yejin Choi, and Luke Zettlemoyer. 2018. Neural metaphor detection in context. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 607–613, Brussels, Belgium. Association for Computational Linguistics.

Andrew Gargett and John Barnden. 2013. Gen-meta: Generating metaphors using a combination of ai reasoning and corpus-based modeling of formulaic expressions. pages 103–108.

Craig A. Hamilton. 2002. Mapping the mind and the body: On w. h. auden's personifications. *Style*, 36(3):408–427.

Devamanyu Hazarika, Soujanya Poria, Sruthi Gorantla, Erik Cambria, Roger Zimmermann, and Rada Mihalcea. 2018. CASCADE: Contextual sarcasm detection in online discussion forums. In *Proceedings of the 27th International Conference on Computational Linguistics*, pages 1837–1848, Santa Fe, New Mexico, USA. Association for Computational Linguistics.

Ilana Heintz, Ryan Gabbard, Mahesh Srivastava, Dave Barner, Donald Black, Majorie Friedman, and Ralph Weischedel. 2013. Automatic extraction of linguistic metaphors with LDA topic modeling. In *Proceedings of the First Workshop on Metaphor in NLP*, pages 58–66, Atlanta, Georgia. Association for Computational Linguistics.

Dirk Hovy, Shashank Srivastava, Sujay Kumar Jauhar, Mrinmaya Sachan, Kartik Goyal, Huying Li, Whitney Sanders, and Eduard Hovy. 2013. Identifying metaphorical word use with tree kernels. In *Proceedings of the First Workshop on Metaphor in NLP*, pages 52–57, Atlanta, Georgia. Association for Computational Linguistics.

Nikhil Jaiswal. 2020. Neural sarcasm detection using conversation context. In *Proceedings of the Second Workshop on Figurative Language Processing*, pages 77–82, Online. Association for Computational Linguistics.

Hyeju Jang, Seungwhan Moon, Yohan Jo, and Carolyn Rosé. 2015. Metaphor detection in discourse. In *Proceedings of the 16th Annual Meeting of the Special Interest Group on Discourse and Dialogue*, pages 384–392, Prague, Czech Republic. Association for Computational Linguistics.

Sedrick Scott Keh, Steven Y. Feng, Varun Gangal, Malihe Alikhani, and Eduard Hovy. 2022. Pancetta: Phoneme aware neural completion to elicit tongue twisters automatically. *arXiv preprint*.

Kalpesh Krishna, John Wieting, and Mohit Iyyer. 2020. Reformulating unsupervised style transfer as paraphrase generation. In *Empirical Methods in Natural Language Processing*.

Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Veselin Stoyanov, and Luke Zettlemoyer. 2020. BART: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7871–7880, Online. Association for Computational Linguistics.

Aaron W. Li, Veronica Jiang, Steven Y. Feng, Julia Sprague, Wei Zhou, and Jesse Hoey. 2020. Aloha: Artificial learning of human attributes for dialogue agents. *Proceedings of the AAAI Conference on Artificial Intelligence*, 34(05):8155–8163.

Bill Yuchen Lin, Wangchunshu Zhou, Ming Shen, Pei Zhou, Chandra Bhagavatula, Yejin Choi, and Xiang Ren. 2020. CommonGen: A constrained text generation challenge for generative commonsense reasoning. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 1823–1840, Online. Association for Computational Linguistics.

Ye Liu, Yao Wan, Lifang He, Hao Peng, and Philip S. Yu. 2021. Kg-bart: Knowledge graph-augmented bart for generative commonsense reasoning. *Proceedings of the AAAI Conference on Artificial Intelligence*, 35(7):6418–6425.

Deyin Long. 2018. Meaning construction of personification in discourse based on conceptual integration theory. *Studies in Literature and Language*, 17:21–28.

Eric Malmi, Aliaksei Severyn, and Sascha Rothe. 2020. Unsupervised text style transfer with padded masked language models. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 8671–8680, Online. Association for Computational Linguistics.

Ning Miao, Hao Zhou, Lili Mou, Rui Yan, and Lei Li. 2019. Cgmh: Constrained sentence generation by metropolis-hastings sampling. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, pages 6834–6842.

Vlad Niculae and Cristian Danescu-Niculescu-Mizil. 2014. Brighter than gold: Figurative language in user generated comparisons. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 2008–2018, Doha, Qatar. Association for Computational Linguistics.

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. Bleu: a method for automatic evaluation of machine translation. In *Proceedings of the 40th annual meeting of the Association for Computational Linguistics*, pages 311–318.

Alec Radford, Jeff Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. 2019. Language models are unsupervised multitask learners.

Robyn Speer, Joshua Chin, and Catherine Havasi. 2017. Conceptnet 5.5: An open multilingual graph of general knowledge. In *AAAI*.

Kevin Stowe, Tuhin Chakrabarty, Nanyun Peng, Smaranda Muresan, and Iryna Gurevych. 2021. Metaphor generation with conceptual mappings. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 6724–6736, Online. Association for Computational Linguistics.

Tomek Strzalkowski, George Aaron Broadwell, Sarah Taylor, Laurie Feldman, Samira Shaikh, Ting Liu, Boris Yamrom, Kit Cho, Umit Boz, Ignacio Cases, and Kyle Elliot. 2013. Robust extraction of metaphor from novel data. In *Proceedings of the First Workshop on Metaphor in NLP*, pages 67–76, Atlanta, Georgia. Association for Computational Linguistics.

Enrica Troiano, Carlo Strapparava, Gözde Özbal, and Serra Sinem Tekiroğlu. 2018. A computational exploration of exaggeration. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 3296–3304, Brussels, Belgium. Association for Computational Linguistics.

Cynthia Van Hee, Els Lefever, and Véronique Hoste. 2018. SemEval-2018 task 3: Irony detection in English tweets. In *Proceedings of The 12th International Workshop on Semantic Evaluation*, pages 39–50, New Orleans, Louisiana. Association for Computational Linguistics.

Zichao Yang, Zhiting Hu, Chris Dyer, Eric P. Xing, and Taylor Berg-Kirkpatrick. 2018. Unsupervised text style transfer using language models as discriminators. In *Proceedings of the 32nd International Conference on Neural Information Processing Systems*, NIPS'18, page 7298–7309, Red Hook, NY, USA. Curran Associates Inc.

Tianyi Zhang, Varsha Kishore, Felix Wu, Kilian Q Weinberger, and Yoav Artzi. 2019. Bertscore: Evaluating text generation with bert. *arXiv preprint arXiv:1904.09675*.

Tianyi Zhang*, Varsha Kishore*, Felix Wu*, Kilian Q. Weinberger, and Yoav Artzi. 2020. Bertscore: Evaluating text generation with bert. In *International Conference on Learning Representations*.

A Appendix A: De-Personification Pipeline

A.1 Dependency Tree Merging Example

Figure 4 contains an example of the merging process that was described in the TOPIC-ATTRIBUTE extraction step in §2. As outlined in §2, edge relations to iteratively merge are *negation modifiers*, *possession modifiers*, *nominal modifiers*, *adjectival complements*, and *objects of prepositions*, as well as words tagged as determiners and parts of compound phrases. The priority order for merging is as follows: 1) compound phrases, 2) nominal modifiers, 3) possession modifiers, 4) negation modifiers, 5) determiners, 6) objects of prepositions, 7) adjectival complements.

A.2 Human-Related Words

In §2, we defined the *IsAPerson* animacy metric as the average of the *IsA* scores between the TOPIC and various words that are very closely related to *'human'*. We called this set the HUMANSET. The words contained in HUMANSET are as follows: {"person", "human", "man", "woman", "human being", "boy", "girl"}.

These words were empirically selected by considering a list of synonyms of the word *'person'* and checking the *IsA* relation COMET scores with the word *'human'*. All of the above words have *IsA* scores with *'person'* of less than 5.10.

A.3 Parameters and Thresholds

***IsAPerson* Threshold.** For the *IsAPerson* animacy score, we use a threshold of 7.0. *IsAPerson* scores < 7.0 are considered animate, while scores ≥ 7.0 are considered inanimate. This threshold was selected empirically using words known to be animate and words known to be inanimate. Words tested include "she" (5.31), "person" (6.41), "moon" (8.743), "opportunity" (9.488), "stars" (8.717), "joe" (5.804), "jane" (4.976), "the police officer" (6.462), "my friend" (6.805), "my new iphone" (10.055). From these observations, we observe that all animate words have an *IsAPerson* score of < 7.0, while all inanimate objects have a score of ≥ 7.0. We hence conclude that 7.0 is a suitable threshold.

Candidate Selection Composite Score Parameters. For the α, β, γ used in the composite score for candidate selection, we use values of $\alpha = 1, \beta = 1, \gamma = 1$. This was selected for two reasons. First, all of the score values had largely similar scales (logarithmic), so setting α, β, γ to a larger value like 2 or 3 would disproportionately favor a certain metric, which is not what we desire. Second, we experimented with using values such as 0.8, 1.2, and 1.5, but the generated de-personifications were either very similar or slightly worse than the default setting of $\alpha = 1, \beta = 1, \gamma = 1$. A possible future direction would be to explore possible values of α, β, γ more thoroughly, but for this dataset, we stick to the simple case of $\alpha = 1, \beta = 1, \gamma = 1$.

Metric	Spearman Correlation	Krippendorff α
Personificationhood	0.0934	0.0250
Appropriateness	0.1660	0.1778
Fluency	0.0050	0.0942
Interestingness	0.6160	0.5898
Meaning Preservation	0.0389	0.2558

Table 5: Inter-annotator agreement scores.

B Appendix C: Evaluation Details

B.1 Human Evaluation Setup

A total of 20 unique AMT annotators participated in the study for fluency, appropriateness, and meaning preservation, each performing 4.0 HITs on average. Annotators were compensated 1.12$ per HIT, each of which was designed to take <6 mins on average.

22 unique AMT annotators participated in the second, separate study for personificationhood, each performing 4.36 HITs on average. Annotators were compensated 0.56$ per HIT, each of which was designed to take <2 mins on average.

For the interestingness study, the details regarding annotator background and selection can be found in §3.2.2 and Appendix B.3.

The html templates including instructions, questions and other study details corresponding to both these AMT studies can be found in the `templates/` subfolder of our code submission zip, with the names `fluency_appropriateness_meaningPreservation.html` and `personificationhood.html` respectively.

B.2 Inter-Annotator Agreement Scores

Each generated input instance and its respective model outputs are labelled by two distinct annotators. To measure inter-annotator agreement, we use Spearman correlation and Krippendorff α, as reported in Table 5.

To get the Spearman correlation point value for a given aspect and test instance, we compute mean pairwise Spearman correlation between the aspect values assigned to the corresponding model outputs by every pair of annotators. Specifically, we use the *scipy.stats* implementation to compute this. [10]

For Krippendorff α, we treat each human evaluation aspect as an ordinal quantity. Specifically, we use the implementation provided by the python library *krippendorff 0.5.1*.[11]

B.3 English Assessment Test for Annotators

From the native English-speaking university student annotators who enrolled to participate in our Interestingness study, we first elicited answers to an English assessment test, as mentioned in §3.2.2.

The assessment test comprised of 12 questions spanning multiple question types testing the examinee's grasp of the use and distinction between various figures of speech, basic literary general knowledge, and verbal reasoning skills. A spreadsheet file containing this test can be found with the name *LiteratureTest.xlsx* under the *Templates/* subfolder of our code submission .zip file.

The final annotators used for our interestingness study were chosen from those who got 11 or more of the 12 questions on the English assessment test correct, hence scoring at least 85% on the test.

C Appendix B: Implementation Details

The BART-base model was trained using a learning rate of 2e-5. This was by conducting a hyperparameter search over the values {1e-6, 5e-6, 1e-5, 2e-5, 5e-5, 1e-4} and selecting the model/epoch based on lowest validation loss. The same process was done to select a batch size of 4 using a hyperparameter search over values {2,4,8,16}. Training was done for 20 epochs and 400 warmup steps. The Adam optimizer was used, and inputs were truncated to a maximum length of 64 tokens (using BART's subword tokenization).

Training was done on Google Colaboratory environments using V100 GPUs. For the BART-base model, a single training loop of 20 epochs takes approximately 10 minutes to complete.

[10] https://docs.scipy.org/doc/scipy/reference/generated/scipy.stats.spearmanr.html

[11] https://pypi.org/project/krippendorff/

Method	Text
Literal Input	The news hit me hard. (ex.7)
Human Ref	The news punched me hard.
COMET	The news report event late me hard.
BL-BART	The news hit me hard.
PA-BART	The news pounded me hard.
Literal Input	Panic is sweeping through the streets contagiously. (ex.8)
Human Ref	Panic is driving around the streets contagiously.
COMET	Panic is result through the streets contagiously.
BL-BART	Panic is running through the streets contagiously.
PA-BART	Panic runs through the streets contagiously.
Literal Input	In the dead of summer, in Ohio, the clothes felt hot and humid. (ex.9)
Human Ref	In the dead of summer, in Ohio, the clothes seemed to be sweating themselves.
COMET	In the dead of summer, in Ohio, the clothes fit and humid.
BL-BART	In the dead of summer, in Ohio, the clothes we wear can be hot and humid.
PA-BART	In the dead of summer, in Ohio, the clothes complained and humid.
Literal Input	Streaks of electricity zipped through the air, swirling around and around dangerous (ex.10)
Human Ref	Streaks of electricity cackled through the air, swirling around and around dangerous.
COMET	Streaks of electricity stain carpet through the air, swirling around and around dangerous.
BL-BART	Streaks of electricity were shooting through the air, swirling around and around the building.
PA-BART	Streaks of electricity danced around and around dangerous.
Literal Input	The sun is nowhere to be found, and the streets are eerily quiet (ex.11)
Human Ref	The sun seems to be on leave, and the streets are eerily quiet.
COMET	The sun down, and the streets full of traffic.
BL-BART	The sun, and the streets.
PA-BART	The sun refuses to be found, and the streets are polite
Literal Input	The wind blew through me fast. (ex.12)
Human Ref	The wind stampeded through me fast.
COMET	The wind howl through me fast.
BL-BART	The wind was going through me fast.
PA-BART	The wind ran me fast.

Table 6: Additional qualitative examples for personification outputs: literal input, human writing, COMET, BL-BART, and PA-BART.

D Appendix D: Additional Examples

Table 6 is an extension of Table 4 and contains additional qualitative examples of the generations.

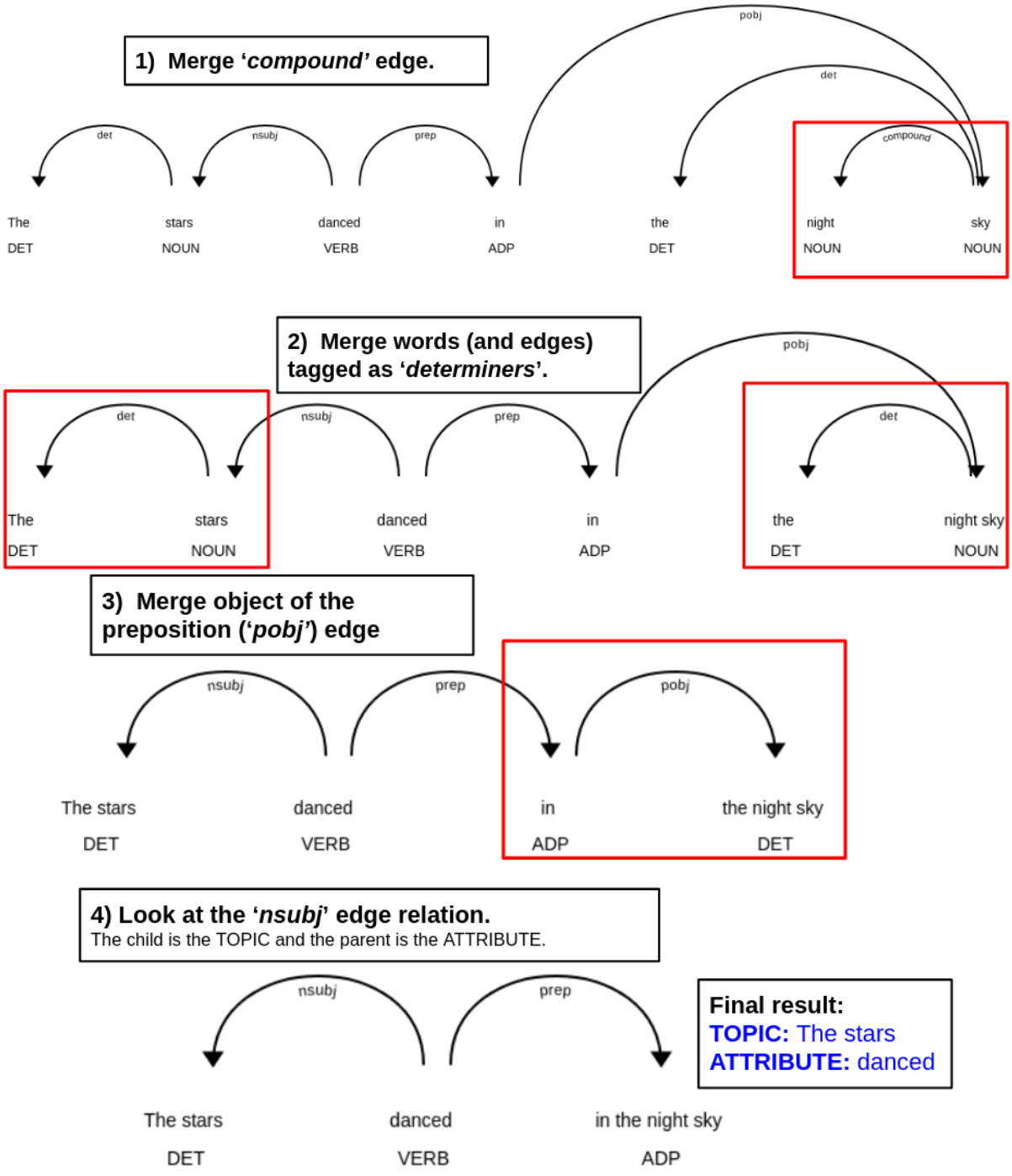

Figure 4: Step-by-step example of the merging process for TOPIC-ATTRIBUTE identification.

Mind the Gap! Injecting Commonsense Knowledge for Abstractive Dialogue Summarization

Seungone Kim[1*] **Se June Joo**[1*] **Hyungjoo Chae**[1,2*]
Chaehyeong Kim[1*] **Seung-won Hwang**[3] **Jinyoung Yeo**[1,2†]
Yonsei University[1] Tutoring at Market Designers[2] Seoul National University[3]
{louisdebroglie, sr7418, mapoout, cheris8, jinyeo}@yonsei.ac.kr
seungwonh@snu.ac.kr

Abstract

In this paper, we propose to leverage the unique characteristics of dialogues sharing commonsense knowledge across participants, to resolve the difficulties in summarizing them. We present **SICK**, a framework that uses commonsense inferences as additional context. Compared to previous work that solely relies on the input dialogue, SICK uses an external knowledge model to generate a rich set of commonsense inferences and selects the most probable one with a similarity-based selection method. Built upon SICK, **SICK++** utilizes commonsense as supervision, where the task of generating commonsense inferences is added upon summarizing the dialogue in a multi-task learning setting. Experimental results show that with injected commonsense knowledge, our framework generates more informative and consistent summaries than existing methods.

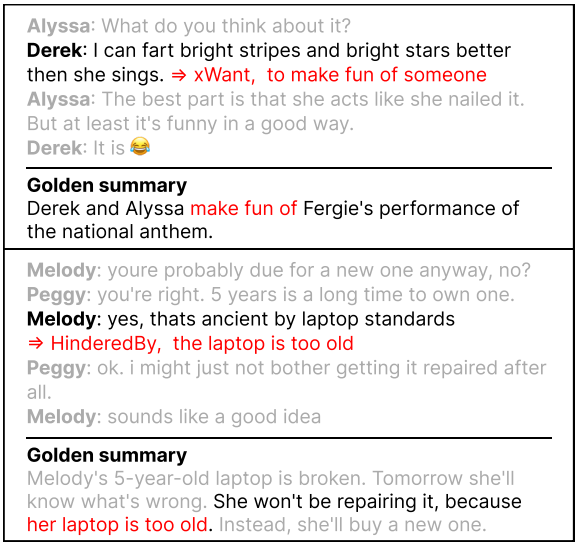

Figure 1: Example of dialogue-summary pairs. Capturing the intention and hidden meaning is important to generate a novel summary.

1 Introduction

Abstractive dialogue summarization is a task of generating a shorter summary while preserving the context of a conversation (Li et al., 2017; Gliwa et al., 2019). Unlike conventional document-to-document summarization (*e.g.*, news articles and scientific publications) (Nallapati et al., 2016; Gehrmann et al., 2018), such *dialogue-to-document* summarization suffers from the discrepancy between input and output forms, which makes learning their mapping patterns more challenging.

There are two key challenges that make summarizing dialogues harder than documents. First, detecting unspoken intention is crucial for understanding an utterance (Mendelsohn, 1994; Ram et al., 2018). As shown in Figure 1, without understanding the intent "*to make fun of someone*", it is hard to write a correct summary. Second, there exists information that can only be understood when its hidden meaning is revealed (Talmy, 1988). For example, it is important to capture the hidden meaning "*The laptop is too old*" beyond the written text "*yes, thats ancient by laptop standards*" when writing the summary.

Commonsense knowledge models (Hwang et al., 2021; Gabriel et al., 2021; West et al., 2022) such as COMET can generate a set of event-centered (*e.g.*, HINDEREDBY, XREASON, XNEED) and social-interaction (*e.g.*, XINTENT, XWANT) commonsense inferences. We argue that the aforementioned issues can be mitigated using commonsense knowledge by *filling in the gap* in a dialogue.

Despite its effectiveness, it is non-trivial to use commonsense knowledge for improving abstractive dialogue summarization performance. While commonsense knowledge has been widely applied to commonsense reasoning (Bosselut and Choi, 2021; Liu et al., 2020; Chang et al., 2021; Wang et al., 2021; Kim et al., 2022) or question answering (Shwartz et al., 2020; Bosselut et al., 2021),

*Equal contribution
†Corresponding author

Proceedings of the 29th International Conference on Computational Linguistics, pages 6285–6300
October 12–17, 2022.

its usage for summarization is understudied (Feng et al., 2021).

In this paper, we present our framework **SICK** and its extension **SICK++** to properly inject commonsense knowledge into state-of-the-art language models (*e.g.*, BART (Lewis et al., 2020)) for abstractive dialogue summarization. We argue a naïve adoption of commonsense only hurts performance in summarization, as (a) expanding source contents is counter-intuitive approach for the goal of condensation, and (b) simply adding additional inputs in pre-trained language models does not lead to robust inferences as reported in Zhou et al. (2021b,a). Our framework addresses this by (a) filtering and (b) robust training.

Based on analytical measurements, commonsense knowledge is selected and enumerated as an additional context of dialogue inputs. In SICK++, we also design a new auxiliary task named *commonsense supervision*. Using commonsense knowledge generated from gold summaries as additional supervision, the goal of the task is to generate the target commonsense. Then, the dialogue summarization and commonsense generation tasks are jointly learned in a multi-task learning setting to effectively inject commonsense knowledge into the shared encoder.

To validate our framework, we conduct a set of experiments on abstractive dialogue summarization benchmarks. Empirical results show that our framework can improve summarization performance with leveraged commonsense knowledge, outperforming other baselines. Human evaluation results prove that our method can generate informative and consistent summaries. In addition, we conduct experiments to analyze the effect of commonsense knowledge on abstractive dialogue summarization.

2 Related Work

2.1 Abstractive Dialogue Summarization

Compared to extractive summarization (Nallapati et al., 2017; Zhang et al., 2018; Zhong et al., 2020), abstractive summarization is considered more challenging and has received extensive attention (Rush et al., 2015; See et al., 2017). Benefiting from the advance of large-scale pre-trained language models, the performance of encoder-decoder models has achieved substantial improvements in document summarization (Nallapati et al., 2016; Gehrmann et al., 2018; Zhang et al., 2020a).

Recently, abstractive dialogue summarization has become another emerging research area, where the goal is to generate concise summaries for conversations such as meetings (Zhu et al., 2021) and chit-chat (Chen et al., 2021). It is more difficult to capture the key points in dialogues than documents, because people do not state the obvious (Grice, 1975) and conversations have a more interactive flow of information between speakers (Li et al., 2021b). Based on the characteristic of the dialogues, many studies focused on organizing the information in the dialogues. Wu et al. (2021) propose to create a summary sketch for a given dialogue as weak supervision. Chen and Yang (2021) explicitly model structures in conversations by incorporating discourse relations and action triples in utterances through structured graphs. Instead of organizing the given dialogue for better understanding, our method adds additional knowledge to fill in the missing cues between dialogues.

2.2 Commonsense Knowledge Models

Recent research has focused on commonsense knowledge acquisition through different lines: commonsense knowledge graphs and commonsense knowledge models. Unlike static knowledge graphs such as ATOMIC (Sap et al., 2019) in which entities and relations between entities are represented in nodes and edges, commonsense knowledge models such as COMET (Bosselut et al., 2019) have been shown to generate implicit commonsense inferences along several dimensions depending on what knowledge graphs they are trained on. Commonsense knowledge models can be used to anticipate and reason unobserved causes and effects in relation to the observed event (Sap et al., 2019). Despite these functions, they are applied on defined domains (Shwartz et al., 2020; Bosselut et al., 2021). Especially, on dialogue summarization task, there has been limited usage of using commonsense directly as additional context. For example, Feng et al. (2021) and Zhou et al. (2022) utilized ConceptNet (Speer et al., 2017), a static knowledge graph with encyclopedic knowledge, to fill in the missing cues between dialogue.

In contrast to encyclopedic knowledge, our method uses event-centered and social-interaction knowledge as additional context. Also, instead of retrieving from a static knowledge graph, our method deploys on-the-fly commonsense knowledge models to acquire a rich set of commonsense

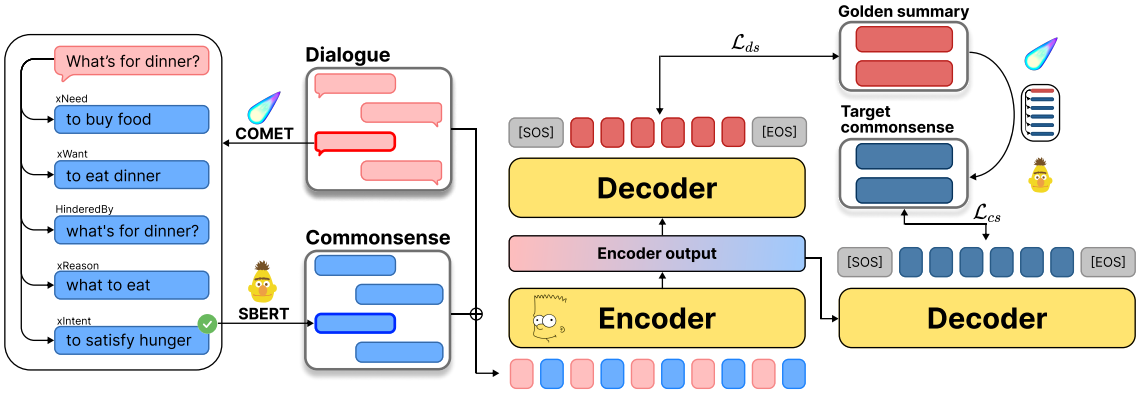

Figure 2: The overall framework of SICK and SICK++. The decoder generating target commonsense is used for SICK++.

inferences dynamically.

3 Proposed Framework

In this section, to inject commonsense knowledge for rich abstractive dialogue summarization, we introduce our new framework, SICK(Summarizing with Injected Commonsense Knowledge) and its extension SICK++, as shown in Figure 2.

3.1 Task Description

Our task definition follows a sequence-to-sequence learning problem setting. Based on pre-trained generative language models, our goal is to learn a mapping function $\mathbb{M} : \mathcal{D} \rightarrow \mathcal{Y}$ where $\mathcal{D} = \{u_1, u_2, ..., u_n\}$ is a dialogue with n utterances, and $\mathcal{Y} = \{y_1, y_2, ..., y_m\}$ is a corresponding summary of m sentences.

We further extend the task with two modifications. First, we generate and filter to acquire a set of commonsense knowledge $\mathcal{C} = \{c_1, c_2, ..., c_n\}$ based on $\mathcal{D}$ (Section 3.2, 3.3). Then, we adjust the mapping function as $\mathbb{M} : \mathcal{X} \rightarrow \mathcal{Y}$, where $\mathcal{X}$ is a cross concatenation of $\mathcal{D}$ and $\mathcal{C}$ (Section 3.3). Second, we add an auxiliary task *commonsense supervision*, $\mathbb{M}^* : \mathcal{X} \rightarrow \mathcal{Z}$, where the target commonsense $\mathcal{Z} = \{z_1, z_2, ..., z_m\}$ is acquired based on $\mathcal{Y}$ (Section 3.4).

3.2 Commonsense Knowledge Generation

In SICK, commonsense knowledge is leveraged as a supplement to insufficient context of dialogues. As shown in Table 1, additional information can be derived from the given utterance in various aspects. There are some cases where the intention of the speaker is crucial in comprehending the dialogue (*e.g.*, "*to believe in something*", "*to talk to someone about dreams*"). Whereas in other cases, the hidden information is necessary (*e.g.*, "*Charlie doesn't believe in dreams*", "*to have a dreams*", "*Charlie is a skeptic*"). We adopt an external commonsense knowledge model that generates a diverse and abundant set of commonsense inferences in natural language. Given a text x and a relation type r, the commonsense knowledge model gives an output c grounded to the relation type. *i.e.*, $f : (x, r) \rightarrow c$.

Utterance	Charlie : Do you really believe that dreams can mean something?
HINDEREDBY	Charlie doesn't believe in dreams.
XWANT	to talk to someone about dreams.
XINTENT	to believe in something.
XNEED	to have a dream.
XREASON	Charlie is a skeptic.

Table 1: Example of commonsense knowledge generated by COMET given a dialogue.

Specifically, we use COMET (Hwang et al., 2021), a widely-used generative commonsense model as our external model. Among the 23 possible candidate relation types, we choose 5 unique relations that helps understand the speakers' intentions and find out the missing information. COMET generates 5 commonsense inferences per relation type, resulting in 25 per input.

Also, to attend to the previous utterances when generating commonsense inferences, we further explore a discourse-aware model, PARA-COMET (Gabriel et al., 2021) that generates coherent inferences. More specifically, while COMET generates a set of commonsense inferences considering only one sentence at a time, PARA-COMET adopts an internal memory module to consider previous dialogue history when generating an output.

Prev-Utterances	Jane : google maps says it is at least 3h Steven : I used to make it in 2, trust me Jane : but it's almost 300km Steven : the road is new , we will make it
Utterance	Jane : I don't want to stress out, let's meet at 4:30 instead of 5, ok?
xINTENT	to avoid stress.
xWANT	**to not be late.**
xREACT	annoyed
xEFFECT	PersonX sweats from nervousness.
xATTR	nervous.

Table 2: Example of commonsense knowledge generated by PARA-COMET given a dialogue.

In Table 2, when generating commonsense inferences of the current dialogue, PARA-COMET conditions on the previous utterance. Knowing what was previously stated, the intention of the speaker (*e.g.*, "to not be bothered", "to not be stressed", "upset") and the hidden knowledge (*e.g.*, "annoyed", "PersonX gets into trouble") differs from COMET.

3.3 Summarizing with Injecting Commonsense Knowledge (SICK)

Filtering Compared to question answering and commonsense generation (Shwartz et al., 2020; Wang et al., 2021), summarizing dialogues has another difficulty. The amount of data provided as the input should be mapped into the output in a concise form. Therefore, simply providing extra input (*i.e.*, commonsense knowledge) may confuse the model when generating a summary. Moreover, it is unable to add every possible commonsense knowledge to the dialogue due to the limited input sequence length of transformer-based models.

To address this issue, we propose to select the most favorable commonsense for each utterance. For 25 candidates, we measure the semantic relevance between the utterance and the commonsense inference concerning. One could imagine that filtering could choose only very similar "safe" examples that might not be as valuable/interesting in practice (*i.e.*, *diversity vs. quality*). However, recent literature address that paradoxically, filtering increases diversity (West et al., 2022). We also discuss the impact of different filtering methods in Appendix E.

We employ SBERT (Reimers and Gurevych, 2019) to compute the similarity score between utterance and commonsense pairs. We select one commonsense inference c_i, with the highest score for each utterance u_i among the candidate relations $\mathcal{R}$. As a result, we obtain the input commonsense $\mathcal{C} = \{c_i\}_{i=1}^{n}$ aligned with dialogue $\mathcal{D}$.

$$c_i = \underset{c_i^r}{\text{argmax}}(\text{score}(u_i, c_i^r)) \quad (r \in \mathcal{R}) \tag{1}$$

Cross Concatenation After obtaining the input commonsense for the dialogue, we concatenate the dialogue and its corresponding set of commonsense inferences. To encode the information that c_i is derived from u_i, we enforce to attend its neighbor token. Instead of concatenating $\mathcal{D}$ and $\mathcal{C}$ back and forth, we concatenate turn by turn considering *locality of reference* (Clark et al., 2019; Zaheer et al., 2020), where tokens tend to attend its neighboring tokens. In order to separate the modalities between dialogues and commonsense inferences, we add special tokens <I>, </I> in back and forth of each commonsense inference c_i. Thus the input sequence $\mathcal{X}$ is formulated as:

$$\mathcal{X} = \mathcal{D} \oplus \mathcal{C} = \cdots \parallel u_i \parallel \text{<I>}\, c_i \,\text{</I>} \parallel \cdots \tag{2}$$

Training SICK is built upon a transformer-based encoder-decoder architecture. The encoder fuses the information from two different modalities (*i.e.*, dialogue and commonsense inference). By the stack of decoders, the encoder output is used for cross-attention with the summary. The training objective, a negative-log likelihood parameterized by θ_{ds}, can be formulated as:

$$\mathcal{L}_{ds} = -\sum_{i=1}^{|\mathcal{Y}|}\sum_{j=1}^{|y_i|} \log P(w_{i,j}|w_{i<j}, \mathcal{X}; \theta_{ds}) \tag{3}$$

where $w_{i,j}$ is j-th token of i-th sentence y_i in target summary $\mathcal{Y}$.

3.4 SICK++

Commonsense Supervision It is well known that models do not consider the actual input as a whole and only look at certain parts of the input therefore not performing the underlying task but some derivative (Branco et al., 2021). For example, in Figure 1, although it is critical to understand Derek's intention (*e.g.*, "*to make fun of Fergie's performance*"), SICK may not utilize the commonsense to comprehend the dialogue.

To overcome this problem, we propose an auxiliary task named *commonsense supervision*. In addition to providing commonsense on the input side, we also leverage commonsense knowledge as additional target variable, which prevents the model

from disregarding commonsense and enforces actually to utilize commonsense. For instance, when the summary "*Derek and Alyssa make fun of Fergie's performance of the national anthem.*" is given to COMET, we observe that a target commonsense "*to make fun of*" is generated. Generating both the summary and the target commonsense has an effect of emphasizing that the input commonsense inference "*to make fun of someone*" is important.

We generate a set of target commonsense inferences $\mathcal{Z}$ with the summary $\mathcal{Y}$ using an external knowledge model f. Then we filter and select the most plausible target commonsense.

$$z_i = \underset{z_i^r}{\text{argmax}}(\text{score}(y_i, z_i^r)) \quad (r \in \mathcal{R}) \tag{4}$$

To adopt commonsense knowledge as additional supervision, we further include commonsense summarization decoder $\mathbb{D}_{cs}$, which learns to generate target commonsense $\mathcal{Z}$.

Training With the target commonsense $\mathcal{Z}$, we train the commonsense summarization decoder $\mathbb{D}_{cs}$ to minimize a negative log-likelihood loss function such as:

$$\mathcal{L}_{cs} = -\sum_{i=1}^{|\mathcal{Z}|}\sum_{j=1}^{|z_i|} \log P(w_{i,j}|w_{i<j}, \mathcal{X}; \theta_{cs}) \tag{5}$$

where w_j^i is a j-th word token of sentence c_i^y from the target commonsense $\mathcal{Z}$.

We linearly combine the two loss functions, Equation 3 and Equation 5, in a multi-task learning setting as follows:

$$\mathcal{L}_{total} = \lambda \cdot \mathcal{L}_{ds} + (1 - \lambda) \cdot \mathcal{L}_{cs} \tag{6}$$

where $\mathcal{L}_{ds}$ and $\mathcal{L}_{cs}$ denote the loss function for dialogue summarization decoder $\mathbb{D}_{ds}$ and commonsense summarization decoder $\mathbb{D}_{cs}$, respectively. λ is a predefined hyperparameter to adjust the scale of each loss. In our setting, we set $\lambda = 0.66$.

Inference During inference, given an input dialogue $\mathcal{D}_{test}$, we first obtain input commonsense $\mathcal{C}_{test}$ for the dialogue, and specify input sequence as $\mathcal{X}_{test} = \mathcal{D}_{test} \oplus \mathcal{C}_{test}$ by concatenating turn by turn. Then, the model predicts summary $\hat{\mathcal{Y}}_{test} = \mathbb{M}(\mathcal{X}_{test})$ for the dialogue $\mathcal{D}_{test}$. Note that while we train the model in a dual-decoder setting, we only use the dialogue summarization decoder $\mathbb{D}_{ds}$ and discard the commonsense prediction decoder $\mathbb{D}_{cs}$ at inference time.

	SAMSum	DialogSum
Train	14,732	12,460
Dev	818	500
Test	819	500
#Tokens/dialogue	82.57	121.56
#Tokens/summary	20.30	22.64
#Turns	11.2	9.5
#Speaker	2.4	2.0
#Compression rate	0.3538	0.2001

Table 3: Statistics of dialogue summarization datasets. # stands for the average number. The compression rate is a ratio of the length of summary divided by the length of dialogue.

4 Experimental Setup

4.1 Datasets and Baselines

We perform experiments on SAMSum (Gliwa et al., 2019) and DialogSum (Chen et al., 2021) datasets. SAMSum is the most widely used resource for abstractive dialogue summarization task. It consists of natural messenger-like conversations in English created by linguists with manually annotated summaries. DialogSum (Chen et al., 2021) is a recently released dataset for a more challenging task with a lower compression ratio. It contains multi-turn dialogues of real-life scenarios collected from three dialogue corpora. The data statistics are in Table 3.

We adopt three different types of baselines: (i) generative language models (See et al., 2017; Wu et al., 2019; Vaswani et al., 2017); (ii) pre-trained language models (Zhang et al., 2020c; Dong et al., 2019; Zhang et al., 2020a; Lewis et al., 2020); (iii) dialogue summarization Models (Feng et al., 2021; Chen and Yang, 2021; Wu et al., 2021). We provide more details in Appendix A.

4.2 Implementation Details

We employ two automatic evaluation metrics as: (i) ROUGE (Lin, 2004) scores, including ROUGE-1, ROUGE-2, and ROUGE-L, which compares word-level uni-gram and bi-gram, and the longest common sequence overlap with the gold summary respectively; (ii) BERTScore (Zhang et al., 2020b)[1], the recent popular metric for text generation, which computes the contextual similarity score between generated and reference summaries. We report F1 scores for both metrics. For simplicity, we use R-1, R-2, R-L, and B-S to denote ROGUE-1, ROUGE-2,

[1]We follow https://github.com/Tiiiger/bert_score to calculate BERTScore. Note that different tools may result in different BERTScore.

Model	SAMSum				DialogSum			
	R-1	R-2	R-L	B-S	R-1	R-2	R-L	B-S
PointerGenerator (See et al., 2017)*	32.27	14.42	34.36	/	/	/	/	/
DynamicConv (Wu et al., 2019)*	41.07	17.11	37.27	/	/	/	/	/
Transformer (Vaswani et al., 2017)*	42.37	18.44	39.27	/	/	/	/	/
DialoGPT (Zhang et al., 2020c)†	39.77	16.58	38.42	/	/	/	/	/
BART-xsum (Lewis et al., 2020)†	51.74	26.46	48.72	53.87	/	/	/	/
UniLM (Dong et al., 2019)†	47.85	24.23	46.67	/	42.38	16.88	34.36	69.40
PEGASUS (Zhang et al., 2020a)†	50.50	27.23	49.32	53.35	38.40	13.84	33.41	68.20
BART-xsum (Lewis et al., 2020)‡	52.50	27.67	48.75	68.16	45.15	19.78	36.57	71.09
D-HGN (Feng et al., 2021)	42.03	18.07	39.57	64.20	/	/	/	/
S-BART (Chen and Yang, 2021)	50.70	25.50	48.08	70.07	/	/	/	/
CODS (Wu et al., 2021)	52.65	27.84	**50.79**	66.55	44.27	17.90	36.98	70.49
SICK w/ COMET (Ours)	53.04	27.60	48.49	71.61	45.70	20.08	40.26	71.08
SICK++ w/ COMET (Ours)	53.24	28.10	48.90	71.71	**46.26**	**20.95**	**41.05**	71.30
SICK w/ PARA-COMET (Ours)	53.39	28.42	49.12	71.83	46.01	20.30	40.75	**71.57**
SICK++ w/ PARA-COMET (Ours)	**53.73**	**28.81**	49.50	**71.92**	46.20	20.39	40.83	71.32

Table 4: Automatic evaluation on abstractive dialogue summarization benchmarks, *i.e.*, SAMSum and DialogSum. Results on SAMSum with * are obtained from (Gliwa et al., 2019), † are obtained from (Wu et al., 2021) and ‡ is a re-implemented version trained under the same conditions with ours for fair comparison. Results on DialogSum for all models are all reimplemented under the same conditions with ours.

ROUGE-L, and BERTScore (see Appendix C).

Our implementation is based on the Huggingface implementation (Wolf et al., 2020) of BART language model. Specifically, we use the weight checkpoint of BART-xsum[2]. We use a maximum input length of 1024 tokens and output length of 100 tokens. Note that the input is either padded or truncated after each utterance and its corresponding commonsense is concatenated during preprocessing. We use a learning rate of 3e-6 and a batch size of 32 when fine-tuning our model on both benchmarks. We use linear warm-up over the first 600 steps, apply linear decay and use the Adam optimizer (Kingma and Ba, 2015). In our experiments, we use beam search with beam size of 20. We fine-tune our model on SAMSum for 20 epochs and DialogSum for 25 epochs. All experiments are run on one A100 NVIDIA GPU. More implementation details about commonsense knowledge generation is included in Appendix B.

5 Experimental Results

5.1 Automatic Evaluation

Performance Table 4 presents the performance on SAMSum and DialogSum test sets. SICK++ outperforms all baselines on ROUGE-1, ROUGE-2 and BERTScore by a consistent margin in both datasets.

[2]https://huggingface.co/facebook/bart-large-xsum

Comparison with State-of-the-Art We find that pre-trained language models (*e.g.*, DialoGPT, UniLM, PEGASUS, BART-xsum), outperform models that are not pre-trained (*e.g.*, PointerGenerator, DynamicConv, Transformer), confirming the impact of pre-training on abstractive dialogue summarization. Among the pre-trained generative language models examined, PEGASUS and BART-xsum are the most competitive models with ROUGE-1 higher than 50. SICK++ shows improvement on all metrics compared to BART-xsum (*e.g.*, without additional input, commonsense supervision) in both benchmarks showing that our method can be applied in different settings.

Among methods that alter the input to seek additional useful information in a dialogue setting, (*e.g.*, D-HGN, SBART, and CODS), CODS achieves better performance over other baselines in SAMSum. However, on DialogSum, a more challenging setting due to higher abstractiveness, CODS is not able to get as much gain of performance compared to other baselines. Meanwhile, SICK++ outperforms all baselines and shows competitive results implying the robustness of our framework.

Commonsense Models While SICK++ shows better performance regardless of which commonsense generation model is used, the excelling choice differs depending on the dataset. In SAMSum, SICK++ shows better performance with PARA-COMET than with COMET, however it shows op-

	SAMSum		DialogSum	
Model	Info.	Cons.	Info.	Cons.
BART-xsum	3.71	3.48	3.71	3.68
SICK++	3.85	3.81	3.79	3.97
Gold	4.00	3.96	4.03	4.21

Table 5: Human evaluation on SAMSum and DialogSum datasets. Info. and Cons. denotes informativeness and factual consistency respectively.

posite result in DialogSum. We conjecture this due to the characteristic of datasets and commonsense models hold. PARA-COMET has an advantage of using parametric memory to consider previous sentences, which may be sensitive in terms of length. Since SAMSum has shorter length of dialogues than DialogSum, the recurrent memory component of PARA-COMET is less likely to forget the previous sentences. We expect to get better performance with the help of commonsense-models that maintains longer memories of sentences/dialogues and leave this as future research.

5.2 Human Evaluation

We conduct human evaluation to verify the quality of the generated summaries. We randomly sample 50 dialogues from test sets of SAMSum and DialogSum, respectively. Annotators were asked to score the quality of a set of summaries from BART-xsum, SICK++, and ground-truth using a Likert scale from 1 (worst) to 5 (best) in terms of **informativeness** (*i.e.*, covers adequate information) and **factual consistency** (*i.e.*, consistent with the original input). Each summary was evaluated by three different annotators. Also, the win-loss ratio, which is not biased by subjectivity, is 51.33 (informativeness) and 54.16 (factual consistency), which is consistent to the observations made from the absolute scores.

In Table 5, human annotated summaries receive the best scores on all dimensions. SICK++ gets better scores than BART-xsum for informativeness, which matches the results of ROUGE scores in Section 5.1. Neural abstractive models often suffer from hallucinations that affect their reliability (Zhao et al., 2020). SICK++ also produces more consistent summaries even though factual consistency is not explicitly modeled. We assume that incorporating commonsense knowledge helps the model recognize the hidden meanings and better understand the dialogue, resulting in fewer factual errors without improper reasoning over conversational flow.

6 Analysis

To evaluate the effectiveness of our method, we address the following research questions to guide our experiments:

- **RQ1**: Does commonsense help summarizing dialogues?
- **RQ2**: Is our method worth using in terms of efficiency despite the extra effort?
- **RQ3**: Does *commonsense supervision* lead SICK++ to inject commonsense knowledge?

6.1 RQ1: Commonsense Applicability

We experiment in a zero-shot setting to examine how commonsense knowledge solely affects dialogue summarization . While there exist many factors that could affect performance besides commonsense during training (*e.g.*, hyperparameter configurations), in a zero-shot setting, we can directly compare when commonsense is given and not. We evaluate BART-xsum and SICK on the SAMSum and DialogSum test sets. Note that we use SICK (*i.e.*, only provided input commonsense) instead of SICK++ for zero-shot evaluation, since we cannot access ground-truth summary to generate target commonsense inferences $\mathcal{Z}$.

Table 6 presents zero-shot evaluation results on SAMSum and DialogSum respectively. We find that SICK outperforms BART-xsum, where the performance gain comes from additional commonsense. Since the only difference between BART-xsum and SICK is the input commonsense, providing extra commonsense for each utterance as Equation 2 helps the model generate more accurate and semantically informative summaries. This also supports the idea that commonsense is essential in resolving the discrepancy between dialogues and documents.

6.2 RQ2: Data Efficiency

Generating commonsense inferences requires irresistible effort, further described in Appendix B. Our approach has limitations in terms of time efficiency. However, we find that our method is helpful in situations where data is insufficient, meaning there is a trade-off (*time vs data efficiency*).

We hypothesize that due to providing additional knowledge and *commonsense supervision*, SICK++

Model	SAMSum				DialogSum			
	R-1	R-2	R-L	B-S	R-1	R-2	R-L	B-S
BART-xsum	20.83	4.28	15.28	46.59	17.40	**4.16**	13.80	42.97
SICK	**23.12**	**5.09**	**17.45**	**47.69**	**18.32**	3.80	**14.98**	**43.97**

Table 6: Zero-shot evaluation on SAMSum and DialogSum test set.

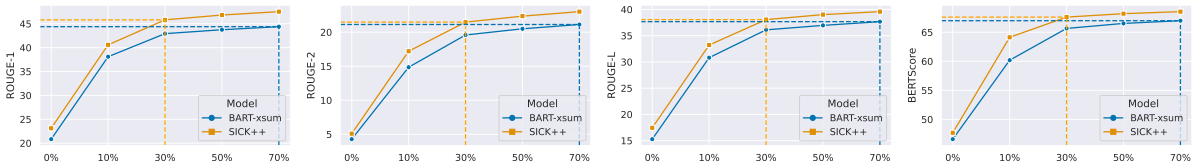

Figure 3: Performance of BART-xsum and SICK++ on SAMSum by varying the size of training data. We use $\text{BART}_{\text{BASE}}$ for both of them. Details are shown in Appendix.

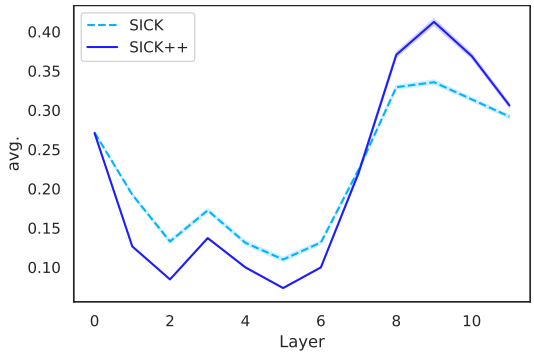

Figure 4: Attention visualization of SICK/SICK++. Each point of the line corresponds to the average attention a particular SICK encoder attention head puts towards commonsense inferences.

can show comparable performance even if only a small amount of training data is available (*i.e.*, data efficiency). As shown in Figure 3, with only 30% of training data, SICK++ shows better performance than BART-xsum trained with 70% of training data. Furthermore, SICK++ consistently outperforms BART-xsum regardless of training data size, proving the robustness of SICK++. The performance gap between SICK++ and BART-xsum can be viewed as a consequence of the leveraged commonsense, based on the fact that BART-xsum is the base architecture of SICK++.

6.3 RQ3: Effect of *commonsense supervision* on Injecting Commonsense Knowledge

We observe that SICK++ shows better performance than SICK, as we show in Table 4, but the reason for the performance improvement is somewhat unclear. To analyze the role of *commonsense supervision*, we now take a look at how the dual decoder setting impacts commonsense utilization of the encoder, the difference between SICK and SICK++.

Attention weights can be viewed as governing how "important" every other token is when producing the next representation for the current token (Clark et al., 2019). We conduct a experiment of measuring the averaged attention value of the commonsense inferences compared to utterances using validation sets of DialogSum, which is more abstractive (*i.e.*, more challenging to comprehend) compared to SAMSum.

The results are illustrated in Figure 4. Rogers et al. (2020) mentioned that final layers of language models are most task-specific, and we observe that SICK++ has marginally higher attention values. We conjecture this is due to the supervision provided by generating $\mathcal{Z}$ instead of relying on distant supervision, meaning that our goal of enforcing the model to use commonsense inferences is successful. SICK++ enforces the encoder to fuse the two different modalities (*e.g.*, utterances, commonsense inferences). Meanwhile in lower and middle layers, SICK++'s attention values tend to be lower than SICK. One possible reason is that lower layers tend to look at syntactic and word-level information (Rogers et al., 2020), whereas the commonsense inferences generated by COMET or PARA-COMET is only meaningful when understood conceptually.

7 Conclusion

In this work, we propose SICK and SICK++ framework in order to resolve the two key challenges: i) *filling in the gap* in dialogues; ii) injecting commonsense knowledge into a model. We show that the difficulties in dialogues are resolved with commonsense knowledge and demonstrated that our

framework can successfully inject commonsense knowledge. As a result of injected commonsense knowledge, we obtain competitive results on SAMSum and DialogSum benchmarks.

Acknowledgements

The authors would like to thank the anonymous reviewers for their helpful comments and suggestions. This work was partly supported by Institute of Information & communications Technology Planning & Evaluation (IITP) grant funded by the Korea government(MSIT) (No. 2020-0-01361, Artificial Intelligence Graduate School Program (Yonsei University)), (No.2021-0-02068, Artificial Intelligence Innovation Hub), and (No. 2022-0-00077, AI Technology Development for Commonsense Extraction, Reasoning, and Inference from Heterogeneous Data). Jinyoung Yeo is a corresponding author.

References

Antoine Bosselut and Yejin Choi. 2021. Dynamic knowledge graph construction for zero-shot commonsense question answering. In *Proceedings of AAAI*.

Antoine Bosselut, Ronan Le Bras, and Yejin Choi. 2021. Dynamic neuro-symbolic knowledge graph construction for zero-shot commonsense question answering. In *Proceedings of AAAI*.

Antoine Bosselut, Hannah Rashkin, Maarten Sap, Chaitanya Malaviya, Asli Celikyilmaz, and Yejin Choi. 2019. Comet: Commonsense transformers for automatic knowledge graph construction. In *Proceedings of ACL*.

Ruben Branco, António Horta Branco, João António Rodrigues, and J. Silva. 2021. Shortcutted commonsense: Data spuriousness in deep learning of commonsense reasoning. In *Proceedings of EMNLP*.

Tuhin Chakrabarty, Yejin Choi, and Vered Shwartz. 2022. It's not rocket science: Interpreting figurative language in narratives. *Transactions of the Association for Computational Linguistics*, 10:589–606.

Ting-Yun Chang, Yang Liu, Karthik Gopalakrishnan, Behnam Hedayatnia, Pei Zhou, and Dilek Hakkani-Tur. 2021. Incorporating commonsense knowledge graph in pretrained models for social commonsense tasks. In *EMNLP Workshop*.

Jiaao Chen and Diyi Yang. 2021. Structure-aware abstractive conversation summarization via discourse and action graphs. In *Proceedings of NAACL*.

Yulong Chen, Yang Liu, Liang Chen, and Yue Zhang. 2021. Dialogsum: A real-life scenario dialogue summarization dataset. In *Proceedings of ACL Findings*.

Kevin Clark, Urvashi Khandelwal, Omer Levy, and Christopher D Manning. 2019. What does bert look at? an analysis of bert's attention. In *ACL Workshop*.

Li Dong, Nan Yang, Wenhui Wang, Furu Wei, Xiaodong Liu, Yu Wang, Jianfeng Gao, Ming Zhou, and Hsiao-Wuen Hon. 2019. Unified language model pre-training for natural language understanding and generation. In *Proceedings of NeurIPS*.

Xiachong Feng, Xiaocheng Feng, and Bing Qin. 2021. Incorporating commonsense knowledge into abstractive dialogue summarization via heterogeneous graph networks. In *Proceedings of China National Conference on Chinese Computational Linguistics*.

Saadia Gabriel, Chandra Bhagavatula, Vered Shwartz, Ronan Le Bras, Maxwell Forbes, and Yejin Choi. 2021. Paragraph-level commonsense transformers with recurrent memory. In *Proceedings of AAAI*.

Sebastian Gehrmann, Yuntian Deng, and Alexander M Rush. 2018. Bottom-up abstractive summarization. In *Proceedings of EMNLP*.

Bogdan Gliwa, Iwona Mochol, Maciej Biesek, and Aleksander Wawer. 2019. Samsum corpus: A human-annotated dialogue dataset for abstractive summarization. In *ACL Workshop*.

Herbert P Grice. 1975. Logic and conversation. In *Proceedings of Speech acts*, pages 41–58. Brill.

Jena D Hwang, Chandra Bhagavatula, Ronan Le Bras, Jeff Da, Keisuke Sakaguchi, Antoine Bosselut, and Yejin Choi. 2021. (comet-) atomic 2020: On symbolic and neural commonsense knowledge graphs. In *Proceedings of AAAI*.

Yu Jin Kim, Beong woo Kwak, Youngwook Kim, Reinald Kim Amplayo, Seung won Hwang, and Jinyoung Yeo. 2022. Modularized transfer learning with multiple knowledge graphs for zero-shot commonsense reasoning. In *Proceedings of NAACL*.

Diederik P Kingma and Jimmy Ba. 2015. Adam: A method for stochastic optimization. In *Proceedings of ICLR*.

Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Ves Stoyanov, and Luke Zettlemoyer. 2020. Bart: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. In *Proceedings of ACL*.

Jiangnan Li, Zheng Lin, Peng Fu, and Weiping Wang. 2021a. Past, present, and future: Conversational emotion recognition through structural modeling of psychological knowledge. In *Findings of the Association for Computational Linguistics: EMNLP 2021*, pages 1204–1214.

Yanran Li, Hui Su, Xiaoyu Shen, Wenjie Li, Ziqiang Cao, and Shuzi Niu. 2017. Dailydialog: A manually labelled multi-turn dialogue dataset. In *Proceedings of IJCNLP*.

Zekang Li, Jinchao Zhang, Zhengcong Fei, Yang Feng, and Jie Zhou. 2021b. Conversations are not flat: Modeling the dynamic information flow across dialogue utterances. In *Proceedings of ACL*.

Chin-Yew Lin. 2004. Rouge: A package for automatic evaluation of summaries. In *Proceedings of ACL*.

Ye Liu, Tao Yang, Zeyu You, Wei Fan, and Philip S Yu. 2020. Commonsense evidence generation and injection in reading comprehension. In *Proceedings of SIGDIAL*.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Roberta: A robustly optimized bert pretraining approach. *arXiv preprint*.

David J Mendelsohn. 1994. *Learning to listen: A strategy-based approach for the second language learner*. Dominie Press.

Ramesh Nallapati, Feifei Zhai, and Bowen Zhou. 2017. Summarunner: A recurrent neural network based sequence model for extractive summarization of documents. In *Proceedings of AAAI*.

Ramesh Nallapati, Bowen Zhou, Cícero Nogueira dos Santos, Çaglar Gülçehre, and Bing Xiang. 2016. Abstractive text summarization using sequence-to-sequence rnns and beyond. In *Proceedings of CoNLL*.

Shashi Narayan, Shay B. Cohen, and Mirella Lapata. 2018. Don't give me the details, just the summary! topic-aware convolutional neural networks for extreme summarization. In *Proceedings of EMNLP*.

Ashwin Ram, Rohit Prasad, Chandra Khatri, Anu Venkatesh, Raefer Gabriel, Qing Liu, Jeff Nunn, Behnam Hedayatnia, Ming Cheng, Ashish Nagar, et al. 2018. Conversational ai: The science behind the alexa prize. *arXiv preprint*.

Nils Reimers and Iryna Gurevych. 2019. Sentence-bert: Sentence embeddings using siamese bert-networks. In *Proceedings of EMNLP*.

Anna Rogers, Olga Kovaleva, and Anna Rumshisky. 2020. A primer in bertology: What we know about how bert works. *Transactions of the Association for Computational Linguistics*, 8:842–866.

Alexander M Rush, Sumit Chopra, and Jason Weston. 2015. A neural attention model for abstractive sentence summarization. In *Proceedings of EMNLP*.

Maarten Sap, Ronan Le Bras, Emily Allaway, Chandra Bhagavatula, Nicholas Lourie, Hannah Rashkin, Brendan Roof, Noah A Smith, and Yejin Choi. 2019. Atomic: An atlas of machine commonsense for if-then reasoning. In *Proceedings of AAAI*.

Abigail See, Peter J Liu, and Christopher D Manning. 2017. Get to the point: Summarization with pointer-generator networks. *arXiv preprint*.

Vered Shwartz, Peter West, Ronan Le Bras, Chandra Bhagavatula, and Yejin Choi. 2020. Unsupervised commonsense question answering with self-talk. In *Proceedings of EMNLP*.

Robyn Speer, Joshua Chin, and Catherine Havasi. 2017. Conceptnet 5.5: An open multilingual graph of general knowledge. In *Proceedings of AAAI*.

Leonard Talmy. 1988. Force dynamics in language and cognition. *Cognitive science*, 12(1):49–100.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Proceedings of NeurIPS*.

Han Wang, Yang Liu, Chenguang Zhu, Linjun Shou, Ming Gong, Yichong Xu, and Michael Zeng. 2021. Retrieval enhanced model for commonsense generation. In *Proceedings of ACL Findings*.

Peter West, Chandra Bhagavatula, Jack Hessel, Jena D Hwang, Liwei Jiang, Ronan Le Bras, Ximing Lu, Sean Welleck, and Yejin Choi. 2022. Symbolic knowledge distillation: from general language models to commonsense models. In *Proceedings of NAACL*.

Adina Williams, Nikita Nangia, and Samuel R. Bowman. 2018. A broad-coverage challenge corpus for sentence understanding through inference. In *Proceedings of NAACL*.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Rémi Louf, Morgan Funtowicz, et al. 2020. Transformers: State-of-the-art natural language processing. In *Proceedings of EMNLP*.

Chien-Sheng Wu, Linqing Liu, Wenhao Liu, Pontus Stenetorp, and Caiming Xiong. 2021. Controllable abstractive dialogue summarization with sketch supervision. In *Proceedings of ACL Findings*.

Felix Wu, Angela Fan, Alexei Baevski, Yann Dauphin, and Michael Auli. 2019. Pay less attention with lightweight and dynamic convolutions. In *Proceedings of ICLR*.

Manzil Zaheer, Guru Guruganesh, Kumar Avinava Dubey, Joshua Ainslie, Chris Alberti, Santiago Ontanon, Philip Pham, Anirudh Ravula, Qifan Wang, Li Yang, et al. 2020. Big bird: Transformers for longer sequences. In *Proceedings of NeurIPS*.

Jingqing Zhang, Yao Zhao, Mohammad Saleh, and Peter Liu. 2020a. Pegasus: Pre-training with extracted gap-sentences for abstractive summarization. In *Proceedings of ICML*.

Tianyi Zhang, Varsha Kishore, Felix Wu, Kilian Q. Weinberger, and Yoav Artzi. 2020b. Bertscore: Evaluating text generation with bert. In *Proceedings of ICLR*.

Xingxing Zhang, Mirella Lapata, Furu Wei, and Ming Zhou. 2018. Neural latent extractive document summarization. In *Proceedings of EMNLP*.

Yizhe Zhang, Siqi Sun, Michel Galley, Yen-Chun Chen, Chris Brockett, Xiang Gao, Jianfeng Gao, Jingjing Liu, and Bill Dolan. 2020c. Dialogpt: Large-scale generative pre-training for conversational response generation. In *Proceedings of ACL*.

Zheng Zhao, Shay B Cohen, and Bonnie Webber. 2020. Reducing quantity hallucinations in abstractive summarization. In *Proceedings of EMNLP Findings*.

Ming Zhong, Pengfei Liu, Yiran Chen, Danqing Wang, Xipeng Qiu, and Xuan-Jing Huang. 2020. Extractive summarization as text matching. In *Proceedings of ACL*.

Pei Zhou, Karthik Gopalakrishnan, Behnam Hedayatnia, Seokhwan Kim, Jay Pujara, Xiang Ren, Yang Liu, and Dilek Hakkani-Tur. 2022. Think before you speak: Explicitly generating implicit commonsense knowledge for response generation. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1237–1252.

Pei Zhou, Pegah Jandaghi, Bill Yuchen Lin, Justin Cho, Jay Pujara, and Xiang Ren. 2021a. Probing commonsense explanation in dialogue response generation. In *Proceedings of EMNLP Findings*.

Pei Zhou, Rahul Khanna, Seyeon Lee, Bill Yuchen Lin, Daniel Ho, Jay Pujara, and Xiang Ren. 2021b. Rica: Evaluating robust inference capabilities based on commonsense axioms. In *Proceedings of EMNLP*.

Chenguang Zhu, Yang Liu, Jie Mei, and Michael Zeng. 2021. Mediasum: A large-scale media interview dataset for dialogue summarization. In *Proceedings of NAACL*.

A Baselines

Generative Language Models

- **PointerGenerator** (See et al., 2017) is a RNN-based method designed for text summarization that deploys copy-attention mechanism.
- **DynamicConv** (Wu et al., 2019) is a lightweight convolutional model that can perform competitively to self-attention.
- **Transformer** (Vaswani et al., 2017) is a random-initialized (*i.e.*, not pre-trained) encoder-decoder architecture with self attention and multi-head attention.

Pre-trained Generative Language Models

- **DialoGPT** (Zhang et al., 2020c) is a GPT-2 model pre-trained on open-domain Reddit data designed for response generation.
- **UniLM** (Dong et al., 2019) is a unified language model which can be used for both natural language understanding and generation tasks by pre-trained using three types of language modeling tasks: unidirectional, bidirectional, and sequence-to-sequence prediction on English Wikipedia and BookCorpus.
- **PEGASUS** (Zhang et al., 2020a) is a model specifically designed for summarization tasks where it is pre-trained with an gap-sentence objective. Important sentences are masked from input and is trained to generate the missing parts, similar to an extractive summary approach.
- **BART** (Lewis et al., 2020) is trained by corrupting text with an arbitrary noising function and learning to reconstruct the original text.
- **BART-xsum**[3] denotes a BART (Lewis et al., 2020) model fine-tuned on XSUM (Narayan et al., 2018) dataset.

Dialogue Summarization Models

- **CODS** (Wu et al., 2021) finds key phrases, and generates length-controllable summary from the key phrases.
- **D-HGN** (Wu et al., 2021) incorporated commonsense knowledge from ConceptNet (Speer et al., 2017) for dialogue summarization.
- **S-BART** (Chen and Yang, 2021) incorporated discourse relations between utterances, and the connections between speakers and actions within utterances to generate abstractive conversation summarization.

B Implementation Details of Commonsense Generation

To generate commonsense, we use COMET and PARA-COMET. Each commonsense model has different choices in terms of model architecture.

[3] https://huggingface.co/facebook/bart-large-xsum

For COMET, we use BART version among several available versions. Publicly available checkpoints were used for both COMET[4] and PARA-COMET[5].GPT-2 version was used for PARA-COMET.For inference, we use beam search with beam size 5 and 10 for each COMET and PARA-COMET, the default setting provided in the public repository. All this procedure is done on one GeForce RTX 3090 GPU.

To investigate the overhead, we measure the time required to generate commonsense inferences in SAMSum. SAMSum, consisted of 14732 samples within the train subset, took 18.3 hours to generate all the needed commonsense inferences. In other words , it took about 4.4719 seconds per dialogue to generate the commonsense.Note that SAMSum has an average of 11.2 turns per dialogue, so that this number could vary depending on how long the given dialogue is.

C Automatic Evaluation Metrics

The following metrics are used for the evaluation of baselines and our models:

- **ROUGE** measures the number of overlapping textual units between generated summary and a set of reference summaries.
- **BERTScore** computes the similarity scores by aligning generated and reference summaries on a token-level based on the output of the BERT-based model. Token alignments are computed greedily with the objective of maximizing the cosine similarity between contextualized token embeddings. We report the F1 score.

D Human Evaluation Metrics

In general, the gold-standard method for evaluating text generation is still human evaluation, where human annotators assess the quality of generated texts. We adopt the following human evaluation metrics:

- **Informativeness**: How well does the generated summary captures the key ideas of the source dialogue?
- **Factual Consistency**: How consistent is the generated summary with respect to the source dialogue? Does the generated summary contain only statements entailed by the source dialogue?

[4]https://github.com/allenai/comet-atomic-2020
[5]https://github.com/skgabriel/paracomet

E Commonsense Selection Methods

We consider two different methods in addition to our similarity-based method to filter commonsense inferences : (i) Random : any random commonsense inferences from 25 possible candidates are chosen for each utterance; (ii) NLI-based : deploy a pre-trained language model that is fine-tuned on a natural language inference (NLI) task, to determine whether a commonsense inference does not contradict with the utterance/sentence.

We use random selection method as a baseline to compare whether filtering helps gain additional performance.

NLI based method is also used by previous works (Gabriel et al., 2021; West et al., 2022) to measure the quality of commonsense inferences. Given a pair of $\{u_i, c_i^r\}$ or $\{y_i, z_i^r\}$, we acquire the probability of ENTAIL and CONTRADICT. Then we measure the score as:

$$\text{NLI Score}(u_i, c_i^r) = P(\text{ENTAILMENT}) - P(\text{CONTRADICT}) \quad (7)$$

$$\text{NLI Score}(y_i, z_i^r) = P(\text{ENTAILMENT}) - P(\text{CONTRADICT}) \quad (8)$$

where the commonsense inference with the highest NLI Score is selected. As a result, we obtain the input commonsense $\mathcal{C} = \{c_1, c_2, ..., c_n\}$ aligned with dialogue $\mathcal{D}$ for additional context and the target commonsense $\mathcal{Z} = \{z_1, z_2, ..., z_m\}$ aligned with summary $\mathcal{Y}$ for additional supervision.

For NLI-based selection, we use RoBERTa-Large (Liu et al., 2019) which is fine-tuned on MNLI (Williams et al., 2018) to score commonsense candidates. Note that we do not have any label telling which commonsense inference is most plausible to be chosen when given an utterance, therefore, we measure the NLI scores in a zero-shot manner.

As shown in Table 7, using the similarity-based selection method consistently outperforms other methods, regardless of the type of commonsense knowledge model. Since NLI-based method is more intuitive compared to similarity-based methods, and was used in previous works, one might ask

Generation Model	Selection Model	SAMSum				DialogSum			
		R-1	R-2	R-L	B-S	R-1	R-2	R-L	B-S
COMET	Random	53.04	27.17	48.49	71.34	46.05	20.46	40.61	70.84
	NLI-based	53.21	28.02	48.85	71.53	45.26	19.94	40.04	70.54
	Similarity-based	**53.24**	**28.10**	**48.90**	**71.71**	**46.31**	**20.95**	**41.10**	**71.71**
PARA-COMET	Random	52.95	27.62	48.51	71.45	45.59	20.16	40.23	70.65
	NLI-based	52.99	28.22	48.61	71.69	45.14	20.01	39.98	70.99
	Similarity-based	**53.73**	**28.81**	**49.50**	**71.92**	**46.20**	**20.39**	**40.83**	**71.32**

Table 7: Performance of SICK++ by varying the commonsense related methods.

why NLI-based method does not show good performance. We conjecture this due to the complexity of each task. Measuring the relation of inclusion is more complex in nature compared to simply measuring the semantic similarity. Our methodology uses a zero-shot setting, therefore it is harder to reach the standards without supervision. The outperforming choice of commonsense selection method could differ when trained with labeled data, and we leave this to future work.

Also, one might conjecturfe that using the top-1 selected commonsense inference with the similarity-based method is a copy of the utterance, resulting in inferences with similarity value 1.0 only selected. However, we found that the mean value of the top-1 commonsense inferences are 0.535799, and standard deviation 0.176364. This shows that the commonsense inferences isn't a copy except for a few bad cases. Considering both diversity and quality is important, and we also leave this to future work.

F Choose of Commonsense Relations from COMET

In prior work such as Chakrabarty et al. (2022) and Li et al. (2021a), it is conventional to selectively use a subset of the COMET relations, depending on the characteristics of a target domain and task. In our work, the social-interaction relations such as *xIntent* and *xWant* are most preferred with the best performance as they are strongly relevant to human-human interaction in dialogue.

Dialogue		**Commonsense**
Frank:	Son, will you come home this weekend?	Frank has to go to work..
Son:	Not sure yet.	son is not sure yet.
Son:	Something happened?	Person asks to son what happened.
Frank:	Of course not.	Frank doesn't want to be rude.
Frank:	Your mother miss you.	your mother misses you..
Son:	I miss her too.	son misses his mother.
Frank:	So will you come?	Frank is too shy to ask..
Son:	I will try.	son will try
Frank:	Good, I will tell your mother that you will come.	son will come.
Son:	Oh, dad.. ok I will come.	Person asks if he can come.
Gold Summary		
Son is coming to see his parents' this weekend.		
BART-xsum		
Son will come home this weekend.		
SICK		
Son will come home this weekend. He misses his mother.		
Julie:	<file photo>	Julie sent a photo.
Emily:	<3 Julie Love, I'm sending tons of kisses ;*;*;*	to show love.
Emily:	<emoji>	Emily sent a photo.
Julie:	Merry Christmas and a lovely mood throughout the whole year, darling.	Julie gives a hug
Emily:	Thank you, for you too <3	Person is thanked.
Julie:	Thanks:*	Julie gets a hug.
Julie:	<file photo> <file photo>	Julie sent a photo.
Gold		
Emily and Julie wish Merry Christmas to each other.		
SICK++		
Julie and Emily are exchanging Christmas greeting.		
BART-xsum		
Julie sends Emily tons of kisses.		
Stewart:	Can you believe he even said that about the forests	the forest to be healthy.
Stewart:	Raking? Really?	to think about the situation.
Shari:	Yes... I can believe that this is an ignorant man...	Shari doesn't want to be ignorant.
Shari:	He proves it daily.. This is just one more example!	Shari wants to be helpful.
Stewart:	He just has no clue...	he has no clue...
Stewart:	I mean, there are so many people dead and all he can think to do is criticize the forestry department? With a totally inappropriate suggestion?	Shari thinks it's inappropriate.
Stewart:	I can't wait to vote for anyone else but him...	to vote for someone else.
Shari:	I know what you mean.. Half my friends voted for him just to see what would happen! Well, guess what?	Shari votes for him
Stewart:	Yeah, we couldn't go another 4 years with a Democrat..	They want to get rid of him.
	...	
Gold		
Stewart and Shari find the current president ignorant and incompetent. They hope he gets voted out. Stewart is going to see what possibilities there are of volunteering in the upcoming elections.		
SICK++		
Stewart and Shari don't like the fact that the current president raked the forests. They think he's an ignorant man. Shari and Stewart don't want to vote for him, but they have to make the best of it now.		
BART-xsum		
Stewart and Shari don't like the way the president is behaving. They are going to vote for anyone else but him.		

Table 8: Successful examples of generated summaries with SICK from DialogSum.

Dialogue		**Commonsense**
Person1:	Are you familiar with American-styled accounting?	Person1 asks PersonY if they are familiar with accounting.
Person2:	I am afraid not.	Person2 is too afraid.
Person2:	I haven't worked in an American company so far.	Person2 is too young to work.
Person1:	What are the most fundamental concepts underlying the accounting process?	to learn about accounting.
Person2:	The first is accounting entity, and the second is going concern.	Person2 is not qualified.
Person2:	The third is measuring unit.	Person2 doesn't know how to measure.
Person2:	The fourth is accounting period, and the fifth is objectivity.	Person2 has to be objective.
Gold		
Person2 tells Person1 about the fundamental concepts of the accounting process.		
SICK++		
Person2 tells Person1 the most fundamental concepts underlying the accounting process.		
BART-xsum		
Person1 asks Person2 about American-styled accounting.		
Person1	Oh, it's getting late.	Person1 has to go to work..
Person1	I've got to run.	to be running.
Person1	It was nice talking to you, karren.	Person1 calls back.
Person2	Thanks, Tim.	to talk to Tim.
Person2	Nice meeting you, too.	to meet PersonY.
Person1	I guess we'll see each other around.	Person1 calls PersonY.
Person2	Yeah, I hope so.	Person2 asks Person2 if they are sure.
Person2	Well, take it easy.	Person2 has to work.
Person1	You too.	to talk to PersonY.
Gold		
Tim and Karren say goodbye.		
SICK++		
Tim and Karren say goodbye to each other.		
BART-xsum		
Tim and Karren meet each other for the first time.		
Person1	Taxi!	Person1 calls a taxi.
Person2	Where to, sir?	Person2 asks for directions.
Person1	I'd like to go to the railway station please.	to go to the train.
Person2	Please hop in.	PersonY asks PersonY to get in..
Person1	Is it a long run to the station?	to go to the station.
Person2	It'll take about 20 minutes.	PersonY asks how long it will take.
Person1	The streets are heavy with traffic at this time of a day, are they?	the traffic is heavy.
Person2	Yes, they are.	Person2 doesn't know what they are.
Person1	Is it the rush hour now?	Person1 has to go to work.
Person2	Yes, it is.	Person2 doesn't know if it is.
Person2	Are you in a hurry sir?	Person2 asks PersonY to hurry up.
Person1	No, I'm not.	No, I'm not.
Person1	Would you please drive slowly and carefully?	Person1 asks Person2 to slow down.
Person2	Yes, sir.	Person2 is asked a question.
Gold		
Person1 takes a taxi to the railway station in the rush hour.		
SICK++		
Person1 takes a taxi to the railway station.		
BART-xsum		
Person1 calls a taxi to go to the railway station. Person2 tells him it'll take about 20 minutes and drives slowly and carefully.		

Table 9: Successful examples of generated summaries with SICK from DialogSum.

Error Type	Dialogue	Commonsense
Copying Utterance	#Person2#: Have a good day! #Person2#: Well, take it easy. #Person1#: Were you born in Los Angeles?	have a good day. to take it easy. born in Los Angeles.
Factual Consistency	#Person2#: I'm afraid not. #Person2#: But I'm not sleepy, darling. #Person2#: I haven't worked in an American company so far.	Person2 is too afraid. Person2 is sleepy. Person2 is too young to work.
Not Informative	#Person2#: I'm afraid not. #Person1#: No, not much. #Person2#: I've heard this one before.	Person2 is too afraid. Person1 says no. Person2 thinks.

Table 10: Failed examples of generated summaries with SICK.

Type-dependent prompt CycleQAG : Cycle consistency for Multi-hop Question Generation

Seungyeon Lee[1], **Minho Lee**[1,2,*]
[1]Department of Artificial Intelligence, Kyungpook National University
[2]ALI Co., Ltd.
statai3237@knu.ac.kr, mholee@knu.ac.kr

Abstract

Multi-hop question generation (QG) is the process of generating answer related questions, which requires aggregating multiple pieces of information and reasoning from different parts of the texts. This is opposed to single-hop QG which generates questions from sentences containing an answer in a given paragraph. Single-hop QG requires no reasoning or complexity, while multi-hop QG often requires logical reasoning to derive an answer related question, making it a dual task. Not enough research has been made on the multi-hop QG due to its complexity. Also, a question should be created using the question type and words related to the correct answer as a prompt so that multi-hop questions can get more information. In this view, we propose a new type-dependent prompt cycleQAG (cyclic question-answer-generation), with a cycle consistency loss in which QG and Question Answering (QA) are learnt in a cyclic manner. The novelty is that the cycle consistency loss uses the negative cross entropy to generate syntactically diverse questions that enable selecting different word representations. Empirical evaluation on the multi-hop dataset with automatic and human evaluation metrics outperforms the baseline model by about 10.38% based on ROUGE score.

1 Introduction

Question Generation (QG) problem that automatically generates a question from a given document with a correct answer is a challenging and an interesting task in the field of natural language processing (Chan and Fan, 2019; Pan et al., 2021; Yu et al., 2020; Dong et al., 2019). With the advent of deep learning, the pre-trained language models (Devlin et al., 2019; Radford et al., 2018; Liu et al., 2019; Raffel et al., 2020; Clark et al., 2020; Peng et al., 2021) were proposed, after which the study of natural language processing began to develop rapidly. These works not only use single-hop QA dataset such as SQuAD (Rajpurkar et al., 2016), which is a representative of research on Question Answering (QA), but also the multi-hop QA dataset such as HotpotQA (Yang et al., 2018). The QA dataset consists of (Context, Question, Answer) pairs along with a lot of QA data, that enables research on Automatic Question Generation (AQG). Most of the question generation methods evaluated questions using the single-hop QA datasets (Duan et al., 2017; Du et al., 2017; Sultan et al., 2020). However, in real-world situations, the questions can be very complex and sometimes require a complicated reasoning process (Gupta et al., 2020; Pan et al., 2021; Yu et al., 2020).

Multi-hop QG requires combining several pieces of information and reasoning over them to derive an answer related a question, making it a dual task. Multi-hop questions that can be encountered in the real world are largely divided into two types, bridges and comparisons. As shown in Fig. 1, the middle side is an example of a bridge-type question. When the question is "Who played Selby Wall in the film that Charlize Theron won an Academy Award for?", the first thing we need to know is what film Theron won the Academy Award. Second, we should be able to obtain information about the actors who played Selby Wall among the actors in the movie. Here, Monster, the movie that connects the two, serves as a bridge. On the other hand, the comparison type shown on the right side of Fig. 1 is to create a question that can be answered by comparing two objects.

Some of the methods for multi-hop QG transform the input text into an intermediate representation such as a parsing tree (Ji et al., 2021), and then convert the resulting form into a question by some well-designed templates or general rules. In (Gupta et al., 2020), they use multi-task learning with an auxiliary loss for sentence-level supporting fact prediction. Graph-based methods (Su et al.,

*Corresponding author

Proceedings of the 29th International Conference on Computational Linguistics, pages 6301–6314
October 12–17, 2022.

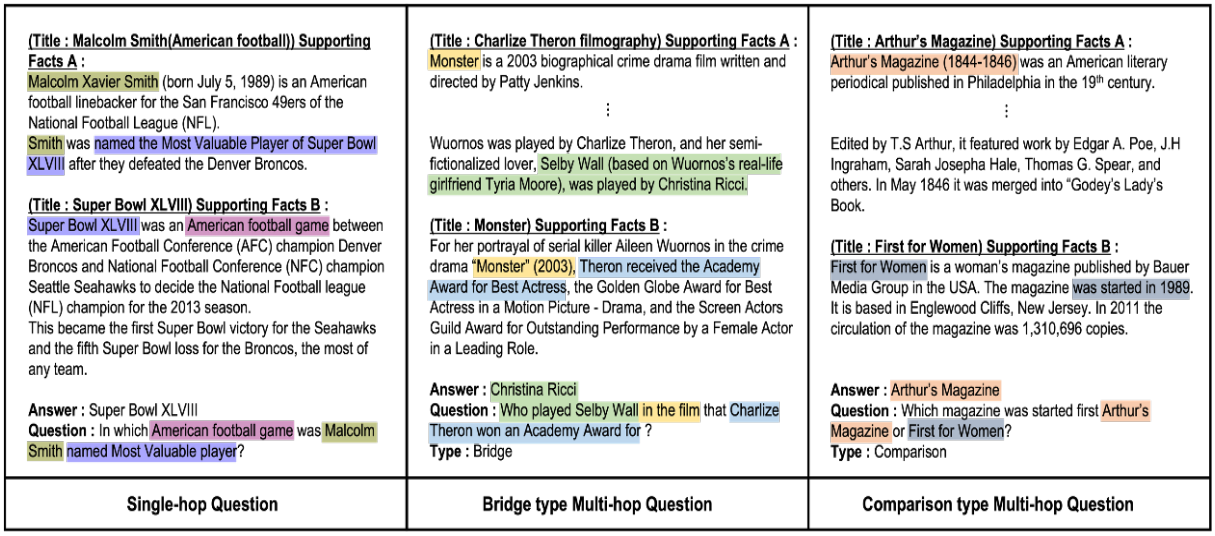

Figure 1: Examples of Single-hop QAG and Multi-hop QAG pair in HotpotQA (Yang et al., 2018) dataset. Multi-hop QG for reasoning multi-hop by finding the contact points between the given Answer and supporting facts A and B. The left is a bridge type, and the right is an example of a comparison type multi-hop question. Both question types are multi-hop, but generate questions with different characteristics.

2020; Kumar et al., 2019) used graph convolution networks(GCNs) to capture dependencies among different pieces of information for reasoning. However, those approaches should predefine graphs only from the question and candidate answers, lacking much key information for multi-hop reasoning.

To alleviate the above problems, we introduce the automated question generation to handle the lack of information for multi-hop reasoning and solve the predefined graph issue in an end-to-end manner. In this view, we propose a type-dependent prompt CycleQAG, which provides additional information and loss of cycle consistency for multi-hop QG. At first, an intermediate task of cyclically learning QG and QA is performed before the fine-tuning stage. In this process, we use cycle consistency loss, and in particular, we introduce the negative cross entropy (NCE), which is used to increase the lexical diversity of multi-hop questions. And in the final step, we use a prompt-based fine-tuning method that maximizes the information obtained from the intermediate task by giving information that can be provided according to the types of questions (eg, type and answer related words). Using the proposed model, we can generate complex questions by an end-to-end manner with semantically similar but diverse vocabulary.

We use the HotpotQA distractor setting and perform experiments with a multi-hop QA dataset. The proposed model outperforms the baseline models in automatic evaluation results such as ROUGE (Lin, 2004) for quantitative evaluation. However, qualitative part of the multi-hop question is evaluated using fluency, relevance, answerability, complexity, and diversity for human evaluation. We evaluate the diversity of vocabulary through qualitative evaluation, and show examples to prove this.

2 Related Works

Question Answering. Machine reading comprehension (MRC) is originally inspired by language proficiency tests, and the machine aims to answer a question by reading and understanding a given context (Zhu et al., 2021). (Seo et al., 2016) introduced the Bi-Directional Attention Flow (BIDAF) network, and proposed a model structure to represent contexts at various levels using a multi-level hierarchical structure. QANet (Yu et al., 2018) models an architecture that does not require a recurrent network and only consists of convolution model and self-attention. Recently, research on new multi-hop QA datasets that require more complex and diverse information, such as HotpotQA (Yang et al., 2018), HybridQA (Chen et al., 2020), MultiModalQA (Talmor et al., 2021), is being actively conducted. For example, (Xiong et al., 2021) used a simple recursive framework to solve open domain multi-hop QA, and configured the model to use dense search for multi-hop setups. In this work, we propose a method to solve multi-hop QA reasoning by a top-down approach to find a specific answer in a whole context.

Question Generation. The ultimate goal of the QG task is to automatically generate questions from texts or knowledge data. With the advent of machine reading comprehension datasets such as SQuAD and pre-trained language models, QG research is conducting the multi-hop reasoning research that deals with more complex and inference-demanding thorny questions, to mimic humans (Pan et al., 2020; Yu et al., 2020; Pan et al., 2021). (Yu et al., 2020) proposed a whole generator evaluator network for generating questions by creating an entity graph to integrate various entities scattered in the texts. (Pan et al., 2021) proposed a multi-hop QG method that used predefined basic operators to search, generate, and aggregate information of each input according to the types of inputs. They also defined and used six inference types of reasoning graphs. In particular, an off-the-shelf template was used for generating a comparison type question that compares two subjects. Although such pre-defined templates or structured models can generate accurate questions for given data, they can be fatal in both quantitative and qualitative aspects when new complex data are given. To overcome some of these issues, we propose a new end-to-end approach to generate multi-hop questions.

Dual task of QA and QG. QA and QG are separate but closely related tasks. In (Tang et al., 2017), they jointly train the two tasks by exploiting the probabilistic correlation between QA and QG. In particular, the parameterized model was jointly trained to minimize the loss function according to the constraints. (Duan et al., 2017) used question generation as an auxiliary task to improve the text-based QA task. They calculated the relevance score between the input question and the answer candidates, and chose the highest relevance score as a correct answer. (Sun et al., 2020) generated additional training instances to further improve the QA model in (Tang et al., 2017), each consists of a question, an answer, and a label for a category. In addition, the question was created by clamping the answer part and providing the answer to the QG model. Many efforts have been made to improve each module by using QA and QG together. In this view, we not only propose a method of using cycle consistency to increase the robustness of QA and QG but also introduce the NCE that increases the diversity of questions.

3 Proposed model

The proposed model for question generation includes an intermediate task execution phase before the fine-tuning step. In the intermediate task, QA and QG are trained to have cycle consistency, where question paraphrasing and similarity are additionally used to increase the robustness of the question generation. We focus on using the multi-hop QG and QA together as a supplement to increase the performance of QG. We define QA as the top-down approach to find a right answer, and QG as the bottom-up approach to use abundant information from entities or sentences. The overall framework of our proposed model is explained in Fig. 2.

3.1 Intermediate Task Training

The intermediate task is to fine-tune the pre-trained model for a task of interest before fine-tuning. We fine-tune the models used for the intermediate task based on the Google-T5 model (Raffel et al., 2020). In the intermediate task stage, QG and QA learn the "cycle consistency". This property was first introduced in the back-translation by Brislin (Brislin, 1970). This translates English to French, and translates the translated French back to English so that the original sentence can be reconstructed. Mathematically, this can be represented as a translator $G : X \rightarrow Y$, $F : Y \rightarrow X$, where G and F are inverse of each other, and are connected like a bijection. Inspired by those properties, when generating questions and answers in the intermediate stage, the proposed QG and QA model uses a cycle-consistency loss that exchanges inputs and outputs in the reverse direction, respectively.

3.1.1 Question Generation

We attempt to handle the multi-hop question generation using the answers and the context. First, we use Google-T5 (Raffel et al., 2020) as the baseline model to automatically generate the output question with a given input answer. While generating a question, as introduced in (Chan and Fan, 2019), there may be more than one instance of the same tokens as the correct answer in the context, then it may be confusing for the model to focus on question generation, we surround the annotated answer span tokens in the context with two tokens. Therefore, the format of the input can be represented as <sep>

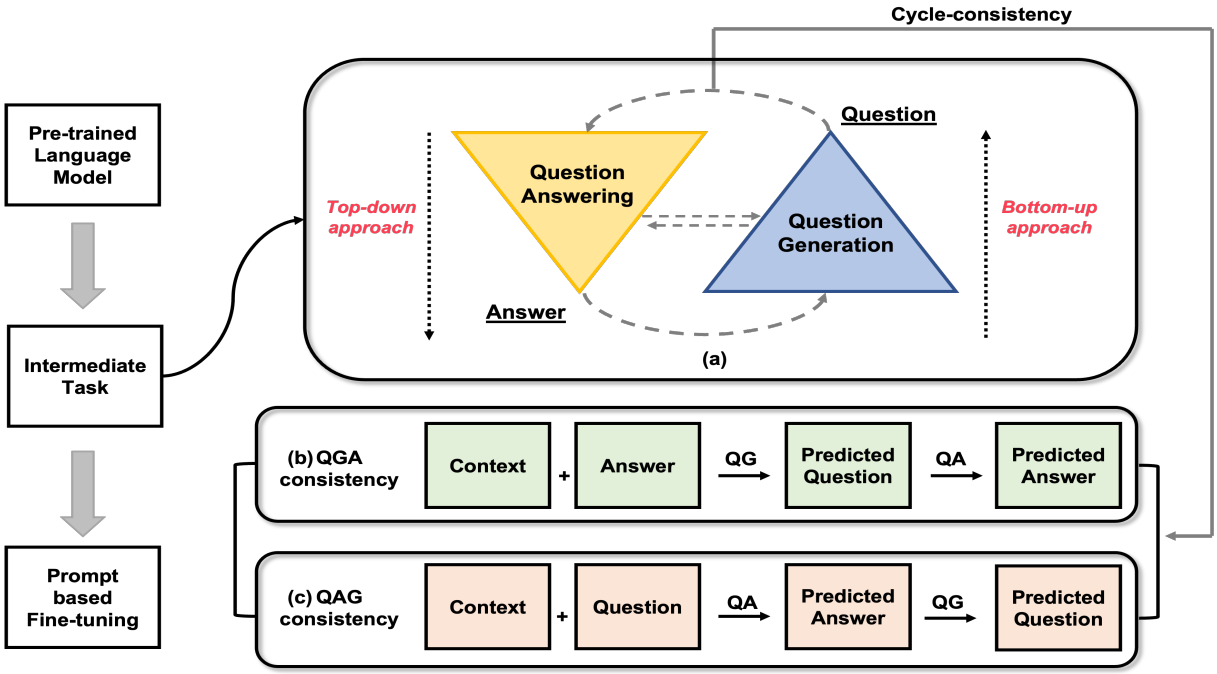

Figure 2: The framework of our proposed model. The proposed model configures QA and QG to learn cycle consistency (a). In the intermediate task, it is possible to create richer questions and more accurate answers through QA and QG interaction. In (b) and (c), the process of QGA-consistency and QAG-consistency to which cycle consistency is applied is shown in detail.

c_1,c_2,c_3 <hl> answer <hl> c_4,c_5, ..., c_m <sep>, where c_i is token of the context. The question generation module of the proposed model not only generates single-hop questions but also produces complex multi-hop questions that require multiple pieces of information. We train the model in such a way that the proposed QG module generates semantically similar and syntactically diverse questions. In particular, we introduce the NCE and the cross-entropy(CE) loss to train a QG model that generates more lexically diverse questions. It controls the probability of adopting information so that it can be semantically similar but syntactically diverse. When training QG, unlike in previous studies, we use the NCE. The probability of occurrence of a word can be lowered, but the essential meaning of a word does not change, thereby enriching the diversity of meaning.

Negative Cross Entropy loss (NCE loss). We use the NCE loss to generate questions more diverse. In general, most studies use the cross entropy (CE) to better train the model to maximize the probability of the correct class (Marek et al., 2021). However, as shown in Fig. 2 (a), in order to generate questions with a more diverse vocabulary for the bottom-up QG, we use the NCE to flatten the word occurrence probability distribution and increase the diversity of vocabulary. In this work, we use the NCE loss to reduce the distance between the predicted value and the actual value such that the generated question has similar meaning with increased lexical diversity.

Question Paraphrasing. In addition, to increase the robustness of the QG module, several questions are generated through a question paraphrasing process. This enables QG with the same meaning but different expressions. We use a paraphrasing model fine-tuned in advance using a Google-T5 model which was Quora Question Pair (QQP) * as the question paraphrasing dataset.

Similarity for generated paraphrasing question. Since it is important to ensure that the meaning of the generated question is the same even if a question with various expressions is generated, the similarity of the generated question should be measured. To find the similarity among paraphrased questions, Sentence-BERT (SBERT) is used, and the overall method is the same as that introduced in (Reimers and Gurevych, 2019), but uses T5 instead

*https://www.kaggle.com/c/quora-question-pairs.

of BERT [†]. We set the similarity value between 0 and 1. The similarity value obtained during the learning process is converted to $1 - similarity$ to train the model in such a way that the loss value decreases as the similarity increases.

Algorithm 1 Procedure of CycleQAG Framework

Input: Context = $(c_1, \ldots, c_n)$, Answer for QG
Context = $(c_1, \ldots, c_n)$, Question for QA
Output: Multi-hop Question

1: Initial QG ← Generate question by QG Input
2: Paraphrasing the generated question
3: Calculate the cosine-similarity between the generated questions and the original question
4: Initial QA ← Generate answer by QA Input
5: **for** $k \leftarrow 1$ to N **do**
6: $\quad C_k \leftarrow cycle(QA, QG)$
7: **end for**
8: **return** Multi-hop question, answer

3.1.2 Question Answering

To build a model that infers an answer using a given (question, passage) pair, the QA model is also trained using the Google-T5 model. In this paper, QA is used to improve the performance of QG, where the ultimate goal of the QA model is to approach the sentence related to the question in a given paragraph and access a correct answer.

3.1.3 Answer related words generation

Multi-hop QG requires more than two pieces of information when asking a question that can fit a correct answer. This requires gathering information in order to create a question. To this end, we use the title information of each supporting paragraph provided as in Fig. 1 to generate words related to a correct answer. Usually, the titles help by providing significant information in generating questions to arrive at a correct answer. We explain this in the appendix C with examples.

3.1.4 Cycle Consistency

We propose the cycle consistency which is widely used in the image field for QG and QA. By using this method, as shown in Fig. 2, QG and QA modules can help generate a robust model that can match the question and answer, respectively. There are not many, but existing multi-hop QG models use graphs or templates (Pan et al., 2021; Su et al., 2020; Kumar et al., 2019). However, we introduce the cycle consistency loss to train a text-based model that can learn by an end-to-end manner. We define the cycle consistency loss to reduce the difference between the predicted value and the actual value. The overall learning flow of the model with cycle consistency is given in Fig. 2 (a). Fig. 2 (b) refers to the QGA-consistency which predicts a question through QG using a given context and an answer, and then predicts an answer through the QA again. Conversely, Fig. 2 (c) refers to QAG-consistency for finding an answer with QA using a given context and a question and then predicting the question with QG. In here, the process flow shown in Fig. 2 (b) is to predict a correct answer, and it is necessary to predict question well through QG to predict correct answer through QA as shown in Fig. 2 (c). Our model uses the cycle consistency property so that answers and questions are learnt better. We describe the overall flow of the CycleQAG framework in Algorithm 1. In algorithm 1, N is the number of samples of the dataset and we use a common early stopping approach for cycle training.

3.2 Prompt-based fine-tuning

We use the multi-hop dataset for fine-tuning a target task after training an intermediate task. Unlike general fine-tuning, the prompt-based fine-tuning adds an element called a prompt. Prompt shows that GPT-3 (Brown et al., 2020) achieves remarkable performance in a few-shot setting, and has been used in many recent studies (Shin et al., 2020; Lester et al., 2021; Gao et al., 2021). In particular, prompt-based fine-tuning (PFT) aims to investigate the knowledge gained from pre-training by reducing the distribution gap between pre-training and fine-tuning stages. Considering these points, we use PFT instead of fine-tuning to make most of the information obtained from the intermediate task. The detailed process of the PFT process is described in Algorithm 2.

A multi-hop QG aims to generate a question using several pieces of information related to a correct answer. For this, we construct a prompt by extracting words related to a correct answer from an intermediate task. This not only uses the types of questions and the correct answers, but also words

[†]Since the T5 is essentially an encoder-decoder model, it is assumed that the decoder knows the meaning of the entire input sentence while generating the first token prediction. This means that the output embedding of the first decoder can grasp the meaning of the sentence like the $[CLS]$ token of the BERT.

Algorithm 2 Procedure of Type-dependent prompt fine-tuning for QG

Input: Prompt (P) ; Context with answer (X)
Output: Multi-hop Question (Y)

1: Configure required prompt token per context
2: Merge prompt token and context with answer
3: Maximize the likelihood of multi-hop question Y.
4: **return** $Pr_\theta(Y|[P;X])$, while keeping the model parameters, θ, fixed.

related to the correct answers obtained from section 3.1.3 as a prompt. This enables providing more information when generating multi-hop questions. We show the results of questions obtained through PFT in appendix C.

3.3 Model Training

In this section, we describe in detail how the model trains an intermediate task. The total loss of the intermediate task consists of QA loss, QG loss, cycle consistency loss, and similarity loss. Therefore the loss is given by Eq. (1).

$$\mathcal{L}_{All} = \mathcal{L}_{QA} + \mathcal{L}_{QG} + \mathcal{L}_{cycle} + \mathcal{L}_{sim} \quad (1)$$

Eq. (2) defines the cycle consistency loss which includes the loss for QA model and loss for QG. We learn QA and QG cyclically with the cycle consistency loss, allowing QA to narrow the range of correct answers, and QG to express more questions in an enriched expression. The similarity loss L_{sim} determines whether the paraphrased questions are semantically close while learning the QG. L_{QA} uses CE loss to find a right answer for a given question. L_{QG} uses the CE loss and the NCE loss to generate a variety of questions that are similar to the original question.

$$\mathcal{L}_{cycle} = \frac{1}{2}[\underbrace{\mathcal{L}_{CE}}_{\text{QGA}} + \underbrace{\{\lambda_1 \mathcal{L}_{NCE} + (1-\lambda_1)\mathcal{L}_{CE}\}}_{\text{QAG}}] \quad (2)$$

The first term of L_{cycle} for learning the cycle-consistency is the loss obtained using the QGA-consistency, which is explained in section 3.1.4 and Fig. 2 (b). Here, the QGA learns to get closer to an original answer by generating a question using a answer and context and then generates a correct answer through the QA again. The remaining terms of the L_{cycle} describe the process of learning the QAG-consistency in Fig. 2(c). While the previous methods use the CE alone, we propose to use the NCE to train the QAG consistency to improve diversity. The QAG learns to get closer to an original question by performing QG through QA. In this part, we adjust the NCE and the CE with λ_1 so that the semantic and the lexical are properly balanced.

For generating questions with a similar meaning, to increase the diversity of questions and reduce the occurrence of most probability words, we use the NCE as shown in Eq. (3). In other words, it is intended to flatten the probability of occurrence of words, so that words can appear in various ways.

$$\mathcal{L}_{NCE} = \frac{1}{N}\sum_{i=1}^{N} y_i \log(\hat{y}_i) + (1-y_i)\log(1-\hat{y}_i) \quad (3)$$

However, if the model is trained only using Eq. (3), NCE may diverge (to -∞), so we adjust the Eq. (4) and hyperparameter λ_1 values such that the probability of the word appearing in the question is lowered only to a certain level. We heuristically adjust λ_1 as 0.2.

$$\mathcal{L}_{CE} = -\frac{1}{N}\sum_{i=1}^{N} y_i \log(\hat{y}_i) + (1-y_i)\log(1-\hat{y}_i) \quad (4)$$

4 Experiments

In the following experiments, we evaluate multi-hop QG based on semantic similarity and lexical diversity. We also evaluate whether the intermediate task has an affect on QG module performance. The baseline model is initialized with a Google-T5 model from HuggingFace Transformer (Wolf et al., 2020), fine-tuned with 3 epochs, with batch size 8. The GPU used in the experiment is 4 Quadro RTX 8000.

4.1 Dataset

We evaluate our model with a focus on multi-hop QA, HotpotQA (Yang et al., 2018). HotpotQA is a multi-hop dataset that is more complex and requires reasoning than existing single-hop QA datasets.

As mentioned in RefNet (Nema et al., 2019), since the test set of HotpotQA is hidden, the validation set is used as the test set, and a part of the training set is used as a validation set. In the experiments, a dataset similar to the HotpotQA

	MODEL	BLEU1	BLEU2	BLEU3	BLEU4	METEOR	ROUGE-L
Baselines	B1. MQA-QG	36.01	25.79	21.88	17.83	26.89	39.95
	B2. BART	36.35	26.70	22.42	18.02	26.96	40.85
	B3. Google-T5	36.89	26.89	22.14	18.27	27.26	41.02
Proposed	P1. Type-dependent prompt CycleQAG	**38.28**	**29.77**	**24.32**	**20.51**	**29.01**	**44.10**
Ablation	A1. w/o type-dependent prompt	36.47	27.61	22.45	18.24	27.94	42.34
	A2. w/o Cycle (intermediate task)	36.96	27.88	22.91	18.98	27.70	41.90
	A3. w/o similarity, paraphrase	37.20	28.18	23.11	19.33	28.57	43.44

Table 1: Performance comparison with baseline and the ablation study. The best performance is bold.

dataset type is additionally used to improve the performance of multi-hop question generation. Representatively, SQuAD, a single-hop dataset, is used. Also, question paraphrasing is used to increase the robustness of the question.

Stanford Question Answering Dataset v1.1 (SQuAD) (Rajpurkar et al., 2016) is a machine reading comprehension dataset with over 100,000 questions created based on Wikipedia articles. Quora Questions Pairs (QQP) provides a label for detecting whether the intent is the same and whether the question text pairs correspond to semantically identical queries, with a focus on various issues related to Quora. We construct a QQP dataset with the same proportions as (Thakur et al., 2021). A detailed description of the dataset and data statistics are shown in appendix A.

4.2 Baselines

Since the multi-hop QG has not yet been explored much, there are few comparison models that can be compared with ours. We use a text-based multi-hop QG model and a model with excellent performance in QG research as our baseline models.

MQA-QG (Pan et al., 2021) generates a question according to a predefined reasoning graph according to the types of questions. In particular, they defined and used 11 templates for comparison type questions. We experiment with the same experiment settings as published in their paper. BART (Lewis et al., 2020) is a model that combines a Bidirectional Transformer and an Auto-Regressive Transformer, and is a pre-trained model using the denoising autoencoder method. In particular, it shows excellent performance in natural language generation. Google-T5 (Raffel et al., 2020) processes the NLP task using the text-to-text input and the output using C4 (Colossal Clean Crawled Corpus), a very large dataset and achieves the highest level in benchmarks such as SuperGLUE. We implement it using open code published by huggingface [‡] for BART and Google-T5.

MODEL	BERTSCORE
MQA-QG	91.88
BART-large	91.03
Google-T5	91.27
Type-dependent prompt CycleQAG (ours)	**93.87**
CycleQAG w/o type-dependent prompt	91.90
CycleQAG w/o Cycle	91.94
CycleQAG w/o similarity,paraphrase	92.93

Table 2: Performance of BERTSCORE. The best performance is bold.

4.3 Multi-hop QG Results and Analysis

Quantitative automatic evaluation and qualitative human evaluation are used to evaluate our proposed model. To this end, we describe in detail the automatic and human evaluation methods and discuss the results.

4.3.1 Automatic Evaluation Metrics

We perform automatic evaluation using n-gram and pre-trained language model based metrics.

N-gram based Metrics. BLEU (Papineni et al., 2002) score is a precision-based evaluation that computes the overlap of n-grams. METEOR (Lavie and Agarwal, 2007) is a relaxed F-measure-based evaluation method in which the unigrams of the hypothesis and the reference do not have an exact level of agreement, but they are synonymous. ROUGE-L (Lin, 2004) is a measure of the sequence of the longest common part between a pair of sentences (Sai et al., 2022). We use the nlg-eval [§] package released by (Sharma et al., 2017) to evaluate an n-gram-based metric.

Pre-trained Language Model based Metrics. BERTSCORE (Zhang* et al., 2020) is a method of evaluating NLG and computes a similarity score of each token of the candidate correct answer and the ground truth. Whereas existing evaluation methods evaluate based on exact match, BERTSCORE is

[‡] https://huggingface.co/

[§] https://github.com/Maluuba/nlg-eval

MODEL	Fluency	Relevance	Answerability	Complexity	Diversity
MQA-QG	2.45	2.38	2.42	2.35	2.35
BART	2.28	2.14	2.30	2.29	2.34
Google-T5	2.39	2.42	2.45	2.40	2.47
Type-dependent prompt CycleQAG (ours)	**2.56**	**2.62**	**2.59**	**2.53**	**2.66**

Table 3: Human Evaluation Results.

effective for paraphrase detection because it uses contextual embedding (Devlin et al., 2019). We download the package for bert-score from (Zhang* et al., 2020)[¶] and use it.

Results and Analysis. We compare the QG performance of the proposed type-dependent prompt CycleQAG model with baseline models and show the automatic metric results in Table 1. Our type-dependent prompt CycleQAG model outperforms all automatic evaluation metrics including ROUGE-L when compared to other models using the same data. Tables 2 indicates whether contextual meaning can be reflected, where BERTSCORE shows excellent performance. Also, it can be seen that they are semantically similar to the original question.

Ablation study. In order to understand the influence of the components of our proposed model, we conduct an ablation study with experimental data for type-dependent prompt CycleQAG. When we do not use the fine-tuning of the type-dependent prompt format that we suggest, it can be observed that the performance is lowered. It can also be observed that the presence or absence of additional information determines the performance improvement when performing a fine-tuning. The additional information referred to here is the types of questions and words related to the answer. We confirm through experiments that their role helps to improve overall performance. Also, fine-tuning the data without the cycle-consistency loss performed in the intermediate task stage, overall performance is degraded. This confirms that the intermediate task is helpful in the QG module when comparing the performance with B3 as shown in Table 1. When we train to generate questions in an intermediate task, we use paraphrase and similarity methods together to increase the lexical diversity of questions. If these methods are removed and tested, the overall performance is slightly degraded.

[¶] https://github.com/Tiiiger/bert_score

4.3.2 Human Evaluation

In this section, we discuss the human evaluation metric. We employ fluency, relevance, answerability, complexity and diversity. Human evaluation is an additional support method for the reliability and robustness of automated evaluation. Here, we use fluency, relevance, and answerability to measure the quality of whether our proposed question is relevant to a given context and answer. Multi-hop QG has high complexity because it requires reasoning, and it is necessary to measure the complexity of the generated question. In addition, we use diversity to evaluate cases in which vocabulary expressions are expressed in various ways, although the meaning of the question is the same. We randomly select 50 question-and-answer pairs from the test set from 20 annotators to obtain evaluations of our model and other baseline models. In human evaluation, we perform the evaluations in a blind format. The range of scores used for evaluation is set to 1-3, and the higher the score, the better the evaluation. The results are shown in Table 3. Overall we consistently get better performance than the conventional models like the BART and Google-T5. We obtain significantly better results than other reference models, especially in terms of diversity and complexity.

5 Conclusion

In this work, we propose type-dependent prompt CycleQAG with cycle consistency. Since multi-hop QG needs to know more diverse information because it needs to gather more scattered pieces of information for generating a question, we introduce the NCE for the first time in the QG task. Also, we demonstrate that the intermediate task is effective in the QG task. Furthermore, we show a significant performance improvement by using prompt-style fine-tuning to make the most of the information obtained from the intermediate task. The experiments show that the proposed model outperforms in all automatic evaluations comparing with the existing text-based multi-hop model and several QG models. Although we use only multi-hop, single-hop-based

datasets, experiments can be performed without additional datasets later using the type-dependent prompt CycleQAG method. In other words, it is possible to learn QA and QG models using unsupervised learning. In the future, we would like to investigate a model that generates questions and answers by itself enough to imitate humans from knowledge through self-cyclic learning that is less influenced by data.

Acknowledgements

This work was partly supported by Institute of Information & communications Technology Planning & Evaluation (IITP) grant funded by the Korea government(MSIT) (No. 2021-0-02068, Artificial Intelligence Innovation Hub), and Center for Applied Research in Artificial Intelligence(CARAI) grant funded by Defense Acquisition Program Administration(DAPA) and Agency for Defense Development(ADD) (UD190031RD).

References

Richard W Brislin. 1970. Back-translation for cross-cultural research. *Journal of cross-cultural psychology*, 1(3):185–216.

Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel Ziegler, Jeffrey Wu, Clemens Winter, Chris Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. 2020. Language models are few-shot learners. In *Advances in Neural Information Processing Systems*, volume 33, pages 1877–1901. Curran Associates, Inc.

Ying-Hong Chan and Yao-Chung Fan. 2019. A recurrent BERT-based model for question generation. In *Proceedings of the 2nd Workshop on Machine Reading for Question Answering*, pages 154–162, Hong Kong, China. Association for Computational Linguistics.

Wenhu Chen, Hanwen Zha, Zhiyu Chen, Wenhan Xiong, Hong Wang, and William Yang Wang. 2020. HybridQA: A dataset of multi-hop question answering over tabular and textual data. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 1026–1036, Online. Association for Computational Linguistics.

Kevin Clark, Minh-Thang Luong, Quoc V. Le, and Christopher D. Manning. 2020. ELECTRA: Pre-training text encoders as discriminators rather than generators. In *ICLR*.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT 2019, Minneapolis, MN, USA, June 2-7, 2019, Volume 1 (Long and Short Papers)*, pages 4171–4186. Association for Computational Linguistics.

Li Dong, Nan Yang, Wenhui Wang, Furu Wei, Xiaodong Liu, Yu Wang, Jianfeng Gao, Ming Zhou, and Hsiao-Wuen Hon. 2019. Unified language model pre-training for natural language understanding and generation. In *Advances in Neural Information Processing Systems*, volume 32. Curran Associates, Inc.

Xinya Du, Junru Shao, and Claire Cardie. 2017. Learning to ask: Neural question generation for reading comprehension. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1342–1352, Vancouver, Canada. Association for Computational Linguistics.

Nan Duan, Duyu Tang, Peng Chen, and Ming Zhou. 2017. Question generation for question answering. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 866–874, Copenhagen, Denmark. Association for Computational Linguistics.

Tianyu Gao, Adam Fisch, and Danqi Chen. 2021. Making pre-trained language models better few-shot learners. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 3816–3830, Online. Association for Computational Linguistics.

Deepak Gupta, Hardik Chauhan, Ravi Tej Akella, Asif Ekbal, and Pushpak Bhattacharyya. 2020. Reinforced multi-task approach for multi-hop question generation. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 2760–2775, Barcelona, Spain (Online). International Committee on Computational Linguistics.

Tianbo Ji, Chenyang Lyu, Zhichao Cao, and Peng Cheng. 2021. Multi-hop question generation using hierarchical encoding-decoding and context switch mechanism. *Entropy*, 23(11).

Vishwajeet Kumar, Yuncheng Hua, Ganesh Ramakrishnan, Guilin Qi, Lianli Gao, and Yuan-Fang Li. 2019. Difficulty-controllable multi-hop question generation from knowledge graphs. In *International Semantic Web Conference*, pages 382–398. Springer.

Alon Lavie and Abhaya Agarwal. 2007. METEOR: An automatic metric for MT evaluation with high levels of correlation with human judgments. In *Proceedings of the Second Workshop on Statistical Machine Translation*, pages 228–231, Prague, Czech Republic. Association for Computational Linguistics.

Brian Lester, Rami Al-Rfou, and Noah Constant. 2021. The power of scale for parameter-efficient prompt tuning. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 3045–3059, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Veselin Stoyanov, and Luke Zettlemoyer. 2020. BART: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7871–7880, Online. Association for Computational Linguistics.

Chin-Yew Lin. 2004. ROUGE: A package for automatic evaluation of summaries. In *Text Summarization Branches Out*, pages 74–81, Barcelona, Spain. Association for Computational Linguistics.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Roberta: A robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692*.

Petr Marek, Vishal Ishwar Naik, Anuj Goyal, and Vincent Auvray. 2021. OodGAN: Generative adversarial network for out-of-domain data generation. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies: Industry Papers*, pages 238–245, Online. Association for Computational Linguistics.

Preksha Nema, Akash Kumar Mohankumar, Mitesh M. Khapra, Balaji Vasan Srinivasan, and Balaraman Ravindran. 2019. Let's ask again: Refine network for automatic question generation. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 3314–3323, Hong Kong, China. Association for Computational Linguistics.

Liangming Pan, Wenhu Chen, Wenhan Xiong, Min-Yen Kan, and William Yang Wang. 2021. Unsupervised multi-hop question answering by question generation. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 5866–5880, Online. Association for Computational Linguistics.

Liangming Pan, Yuxi Xie, Yansong Feng, Tat-Seng Chua, and Min-Yen Kan. 2020. Semantic graphs for generating deep questions. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 1463–1475, Online. Association for Computational Linguistics.

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. Bleu: a method for automatic evaluation of machine translation. In *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics*, pages 311–318, Philadelphia, Pennsylvania, USA. Association for Computational Linguistics.

Shuai Peng, Ke Yuan, Liangcai Gao, and Zhi Tang. 2021. Mathbert: A pre-trained model for mathematical formula understanding. *arXiv preprint arXiv:2105.00377*.

Alec Radford, Karthik Narasimhan, Tim Salimans, and Ilya Sutskever. 2018. Improving language understanding by generative pre-training.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J. Liu. 2020. Exploring the limits of transfer learning with a unified text-to-text transformer. *Journal of Machine Learning Research*, 21(140):1–67.

Pranav Rajpurkar, Jian Zhang, Konstantin Lopyrev, and Percy Liang. 2016. SQuAD: 100,000+ questions for machine comprehension of text. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 2383–2392, Austin, Texas. Association for Computational Linguistics.

Nils Reimers and Iryna Gurevych. 2019. Sentence-BERT: Sentence embeddings using Siamese BERT-networks. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 3982–3992, Hong Kong, China. Association for Computational Linguistics.

Ananya B. Sai, Akash Kumar Mohankumar, and Mitesh M. Khapra. 2022. A survey of evaluation metrics used for nlg systems. *ACM Comput. Surv.*, 55(2).

Min Joon Seo, Aniruddha Kembhavi, Ali Farhadi, and Hannaneh Hajishirzi. 2016. Bidirectional attention flow for machine comprehension. *CoRR*, abs/1611.01603.

Shikhar Sharma, Layla El Asri, Hannes Schulz, and Jeremie Zumer. 2017. Relevance of unsupervised metrics in task-oriented dialogue for evaluating natural language generation. *CoRR*, abs/1706.09799.

Taylor Shin, Yasaman Razeghi, Robert L. Logan IV, Eric Wallace, and Sameer Singh. 2020. AutoPrompt: Eliciting Knowledge from Language Models with Automatically Generated Prompts. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 4222–4235, Online. Association for Computational Linguistics.

Dan Su, Yan Xu, Wenliang Dai, Ziwei Ji, Tiezheng Yu, and Pascale Fung. 2020. Multi-hop question generation with graph convolutional network. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 4636–4647, Online. Association for Computational Linguistics.

Md Arafat Sultan, Shubham Chandel, Ramón Fernandez Astudillo, and Vittorio Castelli. 2020. On the importance of diversity in question generation for QA. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, Online. Association for Computational Linguistics.

Yibo Sun, Duyu Tang, Nan Duan, Tao Qin, Shujie Liu, Zhao Yan, Ming Zhou, Yuanhua Lv, Wenpeng Yin, Xiaocheng Feng, Bing Qin, and Ting Liu. 2020. Joint learning of question answering and question generation. *IEEE Transactions on Knowledge and Data Engineering*, 32(5):971–982.

Alon Talmor, Ori Yoran, Amnon Catav, Dan Lahav, Yizhong Wang, Akari Asai, Gabriel Ilharco, Hannaneh Hajishirzi, and Jonathan Berant. 2021. Multimodal{qa}: complex question answering over text, tables and images. In *International Conference on Learning Representations*.

Duyu Tang, Nan Duan, Tao Qin, Zhao Yan, and Ming Zhou. 2017. Question answering and question generation as dual tasks. *arXiv preprint arXiv:1706.02027*.

Nandan Thakur, Nils Reimers, Johannes Daxenberger, and Iryna Gurevych. 2021. Augmented SBERT: Data augmentation method for improving bi-encoders for pairwise sentence scoring tasks. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 296–310, Online. Association for Computational Linguistics.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Remi Louf, Morgan Funtowicz, Joe Davison, Sam Shleifer, Patrick von Platen, Clara Ma, Yacine Jernite, Julien Plu, Canwen Xu, Teven Le Scao, Sylvain Gugger, Mariama Drame, Quentin Lhoest, and Alexander Rush. 2020. Transformers: State-of-the-art natural language processing. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 38–45, Online. Association for Computational Linguistics.

Wenhan Xiong, Xiang Li, Srini Iyer, Jingfei Du, Patrick Lewis, William Yang Wang, Yashar Mehdad, Scott Yih, Sebastian Riedel, Douwe Kiela, and Barlas Oguz. 2021. Answering complex open-domain questions with multi-hop dense retrieval. In *International Conference on Learning Representations*.

Zhilin Yang, Peng Qi, Saizheng Zhang, Yoshua Bengio, William Cohen, Ruslan Salakhutdinov, and Christopher D. Manning. 2018. HotpotQA: A dataset for diverse, explainable multi-hop question answering. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 2369–2380, Brussels, Belgium. Association for Computational Linguistics.

Adams Wei Yu, David Dohan, Quoc Le, Thang Luong, Rui Zhao, and Kai Chen. 2018. Fast and accurate reading comprehension by combining self-attention and convolution. In *International Conference on Learning Representations*.

Jianxing Yu, Xiaojun Quan, Qinliang Su, and Jian Yin. 2020. Generating multi-hop reasoning questions to improve machine reading comprehension. In *WWW*, pages 281–291.

Tianyi Zhang*, Varsha Kishore*, Felix Wu*, Kilian Q. Weinberger, and Yoav Artzi. 2020. Bertscore: Evaluating text generation with bert. In *International Conference on Learning Representations*.

Fengbin Zhu, Wenqiang Lei, Chao Wang, Jianming Zheng, Soujanya Poria, and Tat-Seng Chua. 2021. Retrieving and reading: A comprehensive survey on open-domain question answering. *arXiv preprint arXiv:2101.00774*.

Appendix

A Data statistics

Dataset	Data type	Train set	Validation set	Test set
HotpotQA	All	90.4k	7.4k	7.4k
	Bridge type	58.5k	5.9k	(invisible)
	Comparison type	17.4k	1.5k	(invisible)
	Single-hop	14.5k	-	-
SQuAD	Single-hop	87.5k	10.5k	-
QQP	Paraphrase	254k	10k	10k

Table A1: Statistics of Datasets. Number of data instances in the train, validation and test set of HotpotQA, SQuAD and Quora Question Pairs (QQP).

Table. A1 is a statistic of the dataset used in the experiment. HotpotQA provides two dataset versions, a distractor setting and a full wiki setting. In this paper, we conduct all experiments on a distractor setting with 2 gold paragraphs and 8 distractor paragraphs. We use training and validation sets provided by HotpotQA to train and evaluate the model. The SQuAD dataset is similar to HoptotQA dataset, the answer to each question can be found in the form of a text span in the paragraph, and it consists of data that can answer diverse types of questions. The QQP dataset detects whether the intent of two given pairs of sentences is the same, and provides a label on whether the two sentences are semantically similar. The QQP dataset consists of more than 400,000 lines of potential question duplicate pairs, with a binary value indicating whether that row contains duplicate pairs. We obtain HotpotQA, SQuAD data through the datasets package provided by Huggingface and use it for experiments, which can be downloaded from `https://huggingface.co/datasets`. QQP dataset can be downloaded from `https://www.kaggle.com/c/quora-question-pairs`.

B Error Analysis

We perform error analysis to analyze the experimental results. In Table A2, Case1 has the same meaning, but only the expression of the lexicon is different, which is a well-predicted case. However, in Case 2, an important entity is omitted, so the meaning of the question is completely changed, and the gist of the question can not be grasped. In this question, an important "traditional sport considered a national sport" is predicted with the word "traditional", so it is a erroneously generated question because an entity related to the sport is omitted. This is not a well-formed question because we cannot conversely answer the question properly. Case 3 is the case of making a semantic error. The ground truth of Case 3 is to ask whether Mozart wrote Pomone or Idomeneo, but the meaning of the predicted question is a semantically different question because it asks which of Pomone or Idomendo premiered first. Case 4 is the case of generating a completely wrong question, where the intent of the question is completely changed by incorrectly predicting the meaning of "and" as "unlike".

Types	Example
Case1	(GT.) Who was the director of the James Bond movie in which Anatole Taubman appeared as a henchman? (Pred.) Who directed the James Bond movie in which Anatole Taubman appeared as a henchman?
Case2	(GT.) Which old, traditional sport is considered to be the national sport by some Swiss? (Pred.) What is an old tradition from the rural central cantons?
Case3	(GT.) Of Pomone and Idomeneo which one was written by Amadeus Mozart? (Pred.) Which opera premiered first, Pomone or Idomeneo?
Case4	(GT.) Unlike Xuzhou, where is Rugao under the adminstration of? (Pred.) Rugao and Xuzhou, Jiangsu are both county-level cities under the administration of who?

Table A2: Error Analysis. GT is ground truth, and Pred is prediction example.

C Experimental result examples

In this section, an example of generating a multi-hop question using the type-dependent prompt CycleQAG method presented by us is shown in detail. In Table A3 and A4, we mark the correct answer we want to obtain in red text and the words related to the correct answer obtained through section 3.1.3 in blue text. If the correct answer and the word related to the correct answer overlap, it is indicated in cyan text. In particular, we can confirm that the meaning of the original question and the generated question did not change, but a vocabulary with a similar meaning was used, making the question richer. In addition, it can be seen through the example of the generated question that it has a considerable influence when generating a question by using the question type, answer, and answer-related words as a prompt. More specifically, in prompt based fine-tuning, we set the input as question type: *type of question*, answer-related words: *combination of words related to the correct answer*, context: *context with answer* and set the output to *multi-hop question*.

Data fields	Example
Answer	Jacksonville station
Generated answer related words	Silver Meteor, Jacksonville station
Context	The Silver Meteor is a passenger train operated by Amtrak between New York City and Miami, Florida. The first diesel-powered streamliner between New York and Florida, since being introduced by the Seaboard Air Line Railroad (SAL) in 1939, it remains in operation now. The train is part of Amtrakś "Silver Service" along with the "Silver Star", another former SAL streamliner. Jacksonville station is an Amtrak train station in Jacksonville, Florida, United States. It serves the "Silver Meteor" and "Silver Star" trains as well as the Thruway Motorcoach to Lakeland. The station lies next door to a freight facility with its own platform and is also just east of Norfolk Southern's Simpson Yard.
Original question	Where does the train that runs from NYC and Miami station at Florida?
Generated question	Which Amtrak station serves the passenger train operated by Amtrak between New York City and Miami, Florida?
Answer	Julianne Moore
Generated answer related words	Emanuelle Goes to Dinosaur Land, Julianne Moore
Context	"Emanuelle Goes to Dinosaur Land" is the of the fourth season of the American television comedy series "30 Rock", and the 79th overall episode of the series. It was written by supervising producer Matt Hubbard and directed by Beth McCarthy-Miller. The episode originally aired on the National Broadcasting Company (NBC) network in the United States on May 13, 2010. Guest stars in this episode include John Anderson, Elizabeth Banks, Jon Hamm, Kristin McGee, Julianne Moore, Michael Sheen, Jason Sudeikis, and Dean Winters. Julianne Moore (born Julie Anne Smith; December 3, 1960) is an American actress, prolific in films since the early 1990s. She is particularly known for her portrayals of emotionally troubled women in both independent and Hollywood films, and has received many accolades, including the 2014 Academy Award for Best Actress.
Original question	What 2014 Academy Award winner guest starred in "Emanulle Goes to Dinosaur Land?
Generated question	Which guest star in "Emanuelle Goes to Dinosaur Land" won the 2014 Academy Award for Best Actress?
Answer	Lantern Waste
Generated answer related words	Lantern Waste, Tumnus
Context	Lantern Waste is a fictional place in "The Chronicles of Narnia" series by C. S. Lewis. It is a wood and is notable as the place where Lucy Pevensie and Mr. Tumnus meet, which is the first scene of Narnia described in the books. The lamppost in the wood is an iconic image of Narnia, and the question of its origin is what convinced Lewis to write more than one book on Narnia. One of King Edmund's titles is "Duke of Lantern Waste". Tumnus is a fictional character in C. S. Lewis' series "The Chronicles of Narnia". He is featured prominently in "The Lion, the Witch and the Wardrobe" and also appears in "The Horse and His Boy" and "The Last Battle". He is close friends with Lucy Pevensie and is the first creature she meets in Narnia, as well as the first Narnian to be introduced in the series. Lewis said that the first Narnia story, "The Lion, the Witch and the Wardrobe", all came to him from a single picture he had in his head of a faun carrying an umbrella and parcels through a snowy wood. In that way, Tumnus was the initial inspiration for the entire Narnia series.
Original question	What is the name of the place in The Chronicles of Narnia where Lucy Pevensie and Mr. Tumnus meet?
Generated question	What is the name of the fictional place where Lucy Pevensie and Mr. Tumnus meet?

Table A3: Example of generated bridge type multi-hop question.

Data fields	Example
Answer	Emory University
Generated answer related words	Emory University, Vanderbilt University
Context	Emory University is a private research university in metropolitan Atlanta, located in the Druid Hills section of DeKalb County, Georgia, United States. The university was founded as Emory College in 1836 in Oxford, Georgia by the Methodist Episcopal Church and was named in honor of Methodist bishop John Emory. In 1915, the college relocated to metropolitan Atlanta and was rechartered as Emory University. The university is the second-oldest private institution of higher education in Georgia and among the fifty oldest private universities in the United States. Emory is frequently cited as one of the world's leading research universities and one of the top institutions in the United States. Vanderbilt University (also known informally as Vandy) is a private research university located in Nashville, Tennessee. Founded in 1873, it was named in honor of shipping and rail magnate Cornelius Vanderbilt, who provided the school its initial $1 million endowment despite having never been to the South. Vanderbilt hoped that his gift and the greater work of the university would help to heal the sectional wounds inflicted by the Civil War.
Original question	Was Vanderbilt University or Emory University founded first?
Generated question	Which university is older, Vanderbilt University or Emory University?
Answer	Battle of Guam
Generated answer related words	Battle of Manila , Battle of Guam
Context	The Battle of Manila (February 3, 1945 – March 3, 1945) was a major battle of the Philippine campaign of 1944-45, during the Second World War. It was fought by American and Filipino forces against Japanese troops in Manila, the capital city of the Philippines. The month-long battle, which resulted in the death of over 100,000 civilians and the complete devastation of the city, was the scene of the worst urban fighting in the Pacific theater. Japanese forces committed mass murder against Filipino civilians during the battle. Along with massive loss of life, the battle also destroyed architectural and cultural heritage dating back to the city's foundation. The battle ended the almost three years of Japanese military occupation in the Philippines (1942–1945). The city's capture was marked as General Douglas MacArthur's key to victory in the campaign of reconquest.The Second Battle of Guam (21 July – 10 August 1944) was the American recapture of the Japanese-held island of Guam, a U.S. territory in the Mariana Islands captured by the Japanese from the U.S. in the 1941 First Battle of Guam during the Pacific campaign of World War II.
Original question	Which battle occurred first, the Battle of Manila or the Battle of Guam ?
Generated question	Which battle took place first, Battle of Guam or Battle of Manila?
Answer	Dracula
Generated answer related words	Dracula, Pistacia
Context	The orchid genus Dracula, abbreviated as Drac in horticultural trade, consists of 118 species native to Mexico, Central America, Colombia, Ecuador and Peru. The name "Dracula" literally means "little dragon", an allusion to the mythical Count Dracula, a lead character in numerous vampire novels and films. The name was applied to the orchid because of the blood-red color of several of the species, the strange aspect of the long spurs of the sepals. Pistacia is a genus of flowering plants in the cashew family, Anacardiaceae. It contains 10 to 20 species that are native to Africa and Eurasia from the Canary Islands, all of Africa, and southern Europe, warm and semidesert areas across Asia, and North America from Mexico to warm and semidesert United States, such as Texas or California.
Original question	Whiich genus has more species, Dracula or Pistacia?
Generated question	Which genus contains more species, Pistacia or Dracula?

Table A4: Example of generated comparison type multi-hop question.

UPER: Boosting Multi-Document Summarization with an Unsupervised Prompt-based Extractor

Shangqing Tu[1,2], **Jifan Yu**[1,2], **Fangwei Zhu**[1,2], **Juanzi Li**[1,2], **Lei Hou**[1,2*] and **Jian-Yun Nie**[3]
[1]Dept. of Computer Sci.&Tech., BNRist, Tsinghua University, Beijing 100084, China
[2]KIRC, Institute for Artificial Intelligence, Tsinghua University
[3]Department of Computer Science and Operations Research, University of Montreal, Canada
{tsq22,yujf21,zfw19}@mails.tsinghua.edu.cn
{lijuanzi,houlei}@tsinghua.edu.cn nie@iro.umontreal.ca

Abstract

Multi-Document Summarization (MDS) commonly employs the 2-stage extract-then-abstract paradigm, which first extracts a relatively short meta-document, then feeds it into the deep neural networks to generate an abstract. Previous work usually takes the ROUGE score as the label for training a scoring model to evaluate source documents. However, the trained scoring model is prone to under-fitting for low-resource settings, as it relies on the training data. To extract documents effectively, we construct prompting templates that invoke the underlying knowledge in Pre-trained Language Model (PLM) to calculate the document and keyword's perplexity, which can assess the document's semantic salience. Our unsupervised approach can be applied as a plug-in to boost other metrics for evaluating a document's salience, thus improving the subsequent abstract generation. We get positive results on 2 MDS datasets, 2 data settings, and 2 abstractive backbone models, showing our method's effectiveness. Our code is available at https://github.com/THU-KEG/UPER

1 Introduction

Multi-Document Summarization (MDS) aims to generate a summary from multiple source articles (McKeown and Radev, 1995). The input text in MDS can be overlong and therefore contain much noisy information (Liu et al., 2021c). The goal of MDS is to reduce the long input and extract salient information (Bing et al., 2015; Fan et al., 2019; Song et al., 2022).

Some previous works (Zaheer et al., 2020) utilise *sparse attention* to handle the long input problem in MDS. Many others tackle this problem by an *extract-then-abstract* paradigm (Liu and Lapata, 2019), which first extracts the salient information in source documents to form a meta-document with a preset length then generates the summary. The first stage reduces the input length for the second stage to cut the abstractive model's memory cost. The extractive stage has two main technical lines: (1) the *statistical* method uses the token-level similarity between keywords and source documents to retrieve relevant documents (Liu et al., 2018). (2) the *regressive* method trains a scoring model to predict the document's ROUGE (Lin, 2004) score with the reference summary (Liu and Lapata, 2019; Mao et al., 2021; Zhong et al., 2021; Zhang et al., 2021).

However, these methods for the extractive stage lead to several problems: (1) the *statistical* methods like tf-idf (Ramos et al., 2003) relies on strict matching of keywords, ignoring those documents with relevant semantic context. (2) the *regressive* methods aim to fit some predefined metrics, e.g., ROUGE, but the fitting result seriously depends on the training data, leading to over-fitting or under-fitting. Besides, the predefined metrics may not adequately measure the quality of the selected documents, resulting in the two-stage gap[1] of the extract-then-abstract paradigm.

To tackle the semantic discrepancy of *statistical* methods and the data dependence of *regressive* methods, we intend to find an unsupervised metric that can evaluate a document's contextual relatedness (Zou et al., 2021) with keywords. It is natural to apply the Pre-trained Language Model (PLM) and leverage its inherent ability of calculating the sequence's perplexity to test whether the keyword can appear in a candidate document's context.

In this paper, we propose an Unsupervised Prompt-based ExtractoR (UPER) which utilizes unsupervised prompts that join the keyword with the document to form a new sequence whose perplexity represents the document's semantic salience. Our method is fully unsupervised and can be used as a plug-in to evaluate documents on different datasets or boost other metrics by combining scores.

To test our method, we explore several dimen-

*Corresponding author.

[1]See more for a preliminary experiment in section 2.2

Proceedings of the 29th International Conference on Computational Linguistics, pages 6315–6326
October 12–17, 2022.

sions of the extractive stage in the extract-then-abstract paradigm. We apply our method on 2 multi-document summarization datasets, 4 different domains, 2 data settings, and 2 abstractive backbone models. The experimental results show that our method effectively complements the token-level similarity and significantly boosts the performance of the subsequent abstractive stage.

Overall, our contributions are the following:

- We propose a new unsupervised framework that employs prompt-based methods to measure the lexical and semantic salience in the extractive stage.
- We carry out a series of experiments demonstrating the effectiveness of the proposed approach.

2 Preliminary

2.1 Problem Formulation

Definition 1 ***Multi-document Summarization*** *is defined as a sequence-to-sequence generation problem, where the input $\mathcal{D}$ consists of n source documents $\{d_1, d_2, \ldots, d_n\}$. The objective is to generate an optimal summary $\mathcal{Y}^*$ according to the conditional distribution, i.e.,*

$$\mathcal{Y}^* = \arg\max_{\mathcal{Y}} P(\mathcal{Y}|\mathcal{D}) \tag{1}$$

However, in automatically collected multi-document summarization datasets (Liu et al., 2018; Fabbri et al., 2019; Gholipour Ghalandari et al., 2020), source documents are usually collected from websites using keywords, e.g., a Wikipedia entity or news title. The target reference summary is usually a description or report of the keyword. It is thus useful to introduce such keywords $\mathcal{K} = \{k_1, k_2, \ldots, k_{n'}\}$ as auxiliary information in Multi-document Summarization, that is,

$$\mathcal{Y}^* = \arg\max_{\mathcal{Y}} P(\mathcal{Y}|\mathcal{K}, \mathcal{D}) \tag{2}$$

Definition 2 *The* ***extractive stage*** *of the extract-then-abstract paradigm first takes n source documents $\mathcal{D}$ as input and selects m candidate documents to form a meta-document $\mathcal{D}'$ which is a subset of $\mathcal{D}$. The* ***abstractive stage*** *trains an end-to-end abstractive model that generates a summary conditioning on the meta-document $\mathcal{D}'$. Therefore, the objective of Multi-document Summarization is re-written as*

$$\mathcal{Y}^* = \arg\max_{\mathcal{D}'} P(\mathcal{D}'|\mathcal{K}, \mathcal{D}) \arg\max_{\mathcal{Y}} P(\mathcal{Y}|\mathcal{D}') \tag{3}$$

While the abstractive stage can be formulated as single document summarization, SOTA transformer architecture is often employed (Hokamp et al., 2020). The goal of extractive stage is to provide the optimal meta-document $\mathcal{D}'$ that can optimize the abstractive stage's output distribution $P(\mathcal{Y}|\mathcal{D}')$. Theoretically, the possible meta-document $\mathcal{D}'$ can be searched within the permutation of documents $A_n^m(\mathcal{D})$, which has the exponential complexity and can't be optimized directly. Prior work performs the extractive stage using either *statistical* method (Liu et al., 2018) or *regressive* method (Liu and Lapata, 2019). However, we observe their weakness in a preliminary experiment described below.

2.2 Preliminary Experiment

We conduct a preliminary experiment on the WikiSum (Liu et al., 2018) dataset to test the statistical and regressive method's performance. We first process the source documents and split them into a fine-grain length; its detail will be described in Table 2. To illustrate the training data dependence of the regressive method, we use a few-shot setting with 1% training data and 100% test data. Following the extract-then-abstract paradigm, we adapt a widely-used Seq2Seq model BART (Lewis et al., 2020) as the abstractive model and change the extractive methods from statistical method tf-idf (Ramos et al., 2003) to regressive method LGB (Ke et al., 2017). Besides, we also introduce ROUGE-1/2/L as oracle extractive methods, which rank the documents by their unigram/bigram/longest sequence overlap with the reference target summary. The highly ranked documents are then sent to BART to generate the final summary. Table 1 is the final summary's ROUGE with the target summary, and figure 1 shows the correlation between the extractive and abstractive stages. We make 2 observations.

Observation I: The **data dependence** of the trained extractor is demonstrated in Table 1: the supervised regressive method obtains the lowest score because the regressive method is prone to under-fitting for the few-shot setting.

Observation II: The **two-stage gap** is presented in figure 1, the extractive stage's RUOGE scores

Extractive Method	R-1	R-2	R-L
regressive	36.5	17.3	29.3
statistical	37.4	18.6	30.7
Oracle-R1-recall	39.6	19.6	31.4
Oracle-R2-recall	**41.1**	**22.1**	**33.2**
Oracle-RL-recall	40.2	20.5	32.3

Table 1: ROUGE-F1 scores under **few-shot** setting(1% training data) with the same abstractive backbone model and different extractive methods on animal domain of WikiSum dataset. Oracles directly use the corresponding ROUGE-recall scores between the input document and the reference summary to rank documents.

don't completely correlate with the abstractive stage's ROUGE scores, which can be concluded from the 9 boxes in the upper right or the lower left part of the matrix. This suggests that ROUGE score may narrowly model the extractive stage's object function $P(\mathcal{D}'|\mathcal{K},\mathcal{D})$ though many previous regressive methods (Liu and Lapata, 2019) choose ROUGE as their training object.

	R1-ext	R2-ext	RL-ext	R1-abs	R2-abs	RL-abs
R1-ext	1.000	0.934	0.337	0.750	0.523	0.494
R2-ext	0.934	1.000	0.342	0.890	0.754	0.711
RL-ext	0.337	0.342	1.000	0.294	0.144	0.168
R1-abs	0.750	0.890	0.294	1.000	0.946	0.942
R2-abs	0.523	0.754	0.144	0.946	1.000	0.991
RL-abs	0.494	0.711	0.168	0.942	0.991	1.000

Figure 1: **Pearson correlation coefficient** between the three ROUGE metrics {R-1, R-2, R-L} for two stages {extractive, abstractive} with a backbone abstractive model BART and different extractive methods on the animal domain WikiSum dataset. **R-1/2/L** denotes ROUGE-1/2/L, ext/abs is the short for extractive/abstractive stage.

3 Method

The key challenge of the extractive stage is how to model the objective function $P(\mathcal{D}'|\mathcal{K},\mathcal{D})$, which evaluates each document and retrieves those related to the keywords. This leads to the following question: what is a proper metric for modeling it?

Based on observations in section 2.2, we propose a criteria for modeling $P(\mathcal{D}'|\mathcal{K},\mathcal{D})$: (1) the metric ought to be unsupervised which can avoid the data dependence; (2) the metric can model both lexical and semantic salience; (3) the metric should be effective for improving the abstractive stage to diminish the two-stage gap.

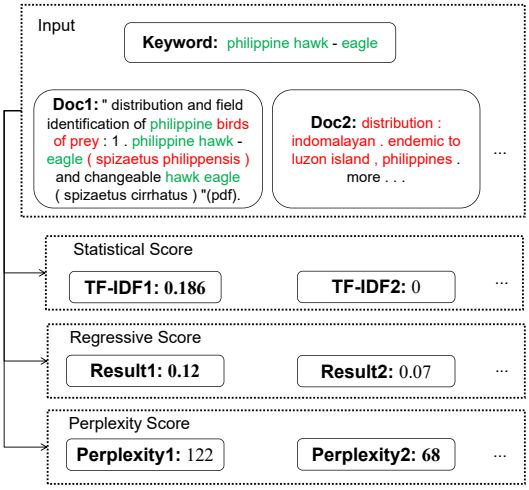

Figure 2: An example of the ranking scores for corresponding extractors. Green words are keywords and red words are effective information for generating the summary. **Bold** score is preferred by the extractor.

3.1 Perplexity

Inspired by the observations and recent success on Pre-trained Language Models, we propose to use perplexity calculated by PLM to model the semantic feature since it can be captured by language model's encoder. Because modeling sequence is the original training task of auto-regressive PLM like GPT (Radford et al., 2019), the perplexity metric can be applied on any datasets without training.

Perplexity is a widely used metric in NLP to evaluate the likelihood of a word sequence $\boldsymbol{x}$ in a language (Jelinek et al., 1977). It is defined as follows:

$$\texttt{PPL}(\boldsymbol{x}) = \exp(-\frac{1}{T}\sum_{i=1}^{T} p_\theta(x_i|\boldsymbol{x}_{<i})) \quad (4)$$

where $\boldsymbol{x}_{<i} = [x_0, ..., x_{i-1}]$.

As statistical extractor would prefer the document that has high overlap with the keyword, there are many noisy documents where the keyword occurs but the effective information lacks, e.g., Doc1[2] in figure 2. To filter out the noise input and retain those documents with relevant semantic but contains no keyword like Doc2 in igure 2,Figure 3 shows our overall framework.

[2]Doc1 is a reference book name in https://en.wikipedia.org/wiki/Philippine_hawk-eagle, which can not provide detailed information for summarization.

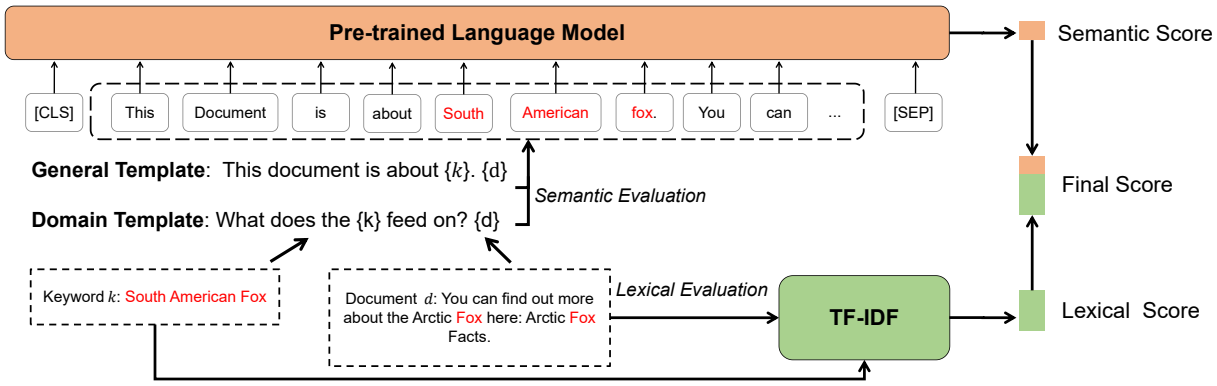

Figure 3: The framework of our UPER model. We design general and domain templates to evaluate the document's semantic salience with keywords using PLM and combine it with the tf-idf score to evaluate the lexical salience.

3.2 Prompt Design

To extract high-quality sources for abstractive models, it is important to consider the document's contextual relation with keywords (Liu et al., 2021a). Specifically, prompts have proven to be effective in modeling the contextual relatedness (Zou et al., 2021). Inspired by this, we propose an Unsupervised Prompt-based ExtractoR (UPER). We design several prompting templates $x(k, d)$ to be filled with the keyword k and the document d, then calculate the whole sequence's perplexity. The general template of $x(k, d)$ that puts the keyword in the introduction position is

$$\text{This document is about } \{k\}.\{d\} \tag{5}$$

which tests whether the keyword can appear in the document's introduction. In order to test whether this keyword can appear in the conclusion of the document, we can also design inverse patterns like:

$$\{d\} \text{ This document is about } \{k\}. \tag{6}$$

Other domain-specific prompting templates are listed in appendix A. The perplexity of the $x(k, d)$ sequence can represent the probability of k showing up in the context of d. Therefore, the semantic salience score $g^S(\mathcal{K}, d)$ of a candidate document is calculated as follows:

$$g^S(\mathcal{K}, d) = -\frac{\sum_{k\in\mathcal{K}}\sum_{x\in\mathcal{X}} \texttt{PPL}(x(k, d))}{|\mathcal{K}| \cdot |\mathcal{X}|}, \tag{7}$$

where $x(k, d)$ indicates the sequence built from a prompting template, $\mathcal{X}$ is pre-designed templates, and PPL is the perplexity calculated by GPT2.

3.3 Score Combination

As our semantic salience metric is unsupervised, we can combine our prompt-based method with statistical methods, e.g., tf-idf. For each keyword k in $\mathcal{K}$, its lexical similarity with the document $g^L(k, d)$ can be measured by tf-idf metric:

$$g^L(k, d) = \frac{n_k}{|d|} \cdot log(\frac{n_d}{n_{dk}}) \tag{8}$$

where n_k, $|d|$, n_d and n_{dk} are the count of the keyword in the document, length of the document, total number of documents, and total number of documents containing the keyword.

For multiple keywords, we view the tf-idf as a keyword's occurring probability conditioned on documents (Ramos et al., 2003) so that the tf-idf for multiple keywords is the joint probability:

$$g^L(\mathcal{K}, d) = \prod_{k\in\mathcal{K}} g^L(k, d) \tag{9}$$

To combine the semantic score $g^S(\mathcal{K}, d)$ and the lexical score $g^L(\mathcal{K}, d)$, we need to normalize them to comparable scales first.

$$N(g) = \frac{g - \mu_g}{\sigma_g} \tag{10}$$

where g, μ_g and σ_g are the metric score, its mean value and variance.

The final score $g^F(\mathcal{K}, d)$ of a candidate document aggregates both semantic and lexical scores:

$$g^F(\mathcal{K}, d) = \lambda \cdot N(g^S(\mathcal{K}, d)) + (1-\lambda) \cdot N(g^L(\mathcal{K}, d)) \tag{11}$$

where the λ is a coefficient. Only the top-m documents are selected and passed to the next stage.

4 Experiments

4.1 Experimental Setting

This section introduces the datasets, evaluation, and baselines of our experiments. More implementation details are introduced in appendix B.

Dataset	Domain	#Examples	$\lvert\mathcal{D}\rvert \times \lvert d\rvert$	$\lvert r\rvert$
WikiSum	animal	60,816	180 × 20	92
WikiSum	company	62,545	253 × 25	125
WikiSum	film	59,973	266 × 23	98
WCEP	news	10,200	241 × 16	28

Table 2: Statistics of each domain on the WikiSum and WCEP dataset. $\lvert\mathcal{D}\rvert$, $\lvert d\rvert$, $\lvert r\rvert$ is respectively the average number of source documents, source document tokens and target reference summary tokens.

Dataset. To evaluate the models, we use the **WikiCatSum** dataset (Perez-Beltrachini et al., 2019), a subset of **WikiSum** (Liu et al., 2018), which consists of three different domains in Wikipedia (*Animal*, *Company* and *Film*) and another large-scale multi-document summarization dataset **WCEP** (Gholipour Ghalandari et al., 2020). The statistics are shown in Table 2.

Evaluation. We use ROUGE-F1 (Lin, 2004) to evaluate the generated summary with respect to the reference. For different model settings, we perform corresponding extractive stage on the training and test set, then fine-tune the abstractor on the training set and report its evaluation result on the test set.

Baselines. We select several typical baselines in related tasks, including:

• **TF-S2S** (Liu et al., 2018). A method that views the documents as a long sequence and uses a transformer decoder to generate the summary.

• **C2T** (Perez-Beltrachini et al., 2019) uses a CNN encoder and two structured decoders with topic-aware information discovered by LDA.

• **TWAG** (Zhu et al., 2021) is a recent wikipedia abstractor which explicitly considers topics on WikiCatSum dataset using topic classifiers.

• **Noisysumm** (Liu et al., 2021c) uses self-distillation to improve the abstractor's ability to handle noisy input. It can be applied to other abstractors like UniLMv2 (Bao et al., 2020).

• **BART** (Lewis et al., 2020) is a sequence-to-sequence model with bidirectional and autoregressive transformers that accomplished state-of-the-art results on single-document summarization. It has a length limit of 1024.

• **Longformer Encoder Decoder (LED)** (Beltagy et al., 2020) is a transformer-based sequence-to-sequence model which utilizes a sparse attention mechanism to achieve the linear complexity with respect to the input length. Its length limit is 16384.

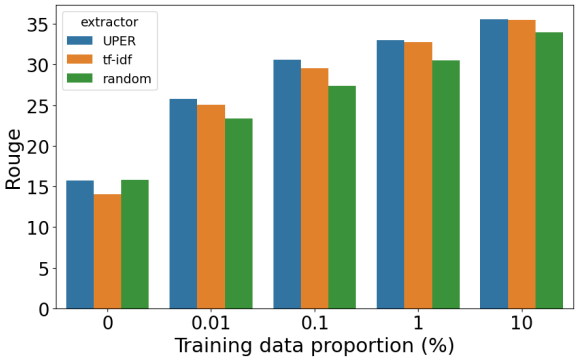

Figure 4: ROUGE-F1 scores under different **few-shot** training data proportions on the film domain of WikiSum dataset. random, `tf-idf` and `UPER` are three LED models trained with corresponding extractor.

Extractive Method	R-1	R-2	R-L
Random	34.8	12.5	26.8
tf-idf	35.0	12.6	27.3
UPER(ours)	**35.1**	**12.8**	**27.4**

Table 3: ROUGE-F1 scores under **few-shot** setting (1% training data) with the same backbone abstractive model LED and different extractive methods on WCEP dataset.

4.2 Low-resource Setup Results

To test our model's performance under a low-resource setting, we conduct experiments with different proportions of training data for fine-tuning abstractive models. As shown in figure 4, UPER outperforms both tf-idf and random order with the data scale changed from 0 to 10%. Note that UPER uses a default $\lambda = 0.75$, which means it combines with tf-idf metric and steadily boosts tf-idf's performance across different data scales. In addition to WikiCatSum dataset, UPER also boosts the performance on WCEP dataset in Table 3, demonstrating UPER's generalization ability across datasets as it benefits from the unsupervised prompts.

4.3 Full-data Setup Results

Besides the low-resource setting, we also conduct experiments under the full-data setup. Table 4 and Table 5 show the overall results. We first reproduce the abstractors without the extractive stage. Then UPER is applied on BART and LED to extract the input, which helps them achieve SOTA ROUGE scores on WikiCatSum dataset.

From the experimental results, we have several observations: (1) UPER can boost both BART and LED's performance, indicating our model's gen-

Model	Company			Film			Animal		
	R1	R2	RL	R1	R2	RL	R1	R2	RL
TF-S2S (Liu et al., 2018)	26.0	9.5	20.4	36.5	18.8	31.0	44.0	28.8	40.0
C2T (Perez-Beltrachini et al., 2019)	27.5	10.6	21.4	38.0	21.2	32.3	42.7	27.9	37.9
TWAG (Zhu et al., 2021)	34.1	11.9	31.6	40.8	21.2	34.3	43.1	24.4	40.9
UniLMv2 (Liu et al., 2021c)	33.3	14.4	25.4	42.5	25.9	36.5	45.5	31.7	40.9
Noisysumm (Liu et al., 2021c)	33.5	**15.0**	25.9	42.7	26.1	36.8	45.9	**32.2**	41.4
BART	33.2	10.5	30.4	37.6	17.5	35.4	42.8	24.3	40.6
BART + UPER	36.7	14.3	33.8	43.3	24.6	41.0	**46.4**	29.1	**44.4**
LED	36.4	13.7	33.4	43.9	24.7	41.4	43.9	25.5	41.7
LED + UPER	**37.0**	14.5	**33.9**	**44.7**	**26.2**	**42.4**	**46.4**	26.5	**44.4**

Table 4: ROUGE-F1 scores of different models on three domains (*Company*, *Film* and *Animal*) of WikiCatSum dataset under the **full-data** setting (100% training data for fine-tuning abstractive models).

Extractive Method	R-1	R-2	R-L
Random	39.1	16.4	31.2
tf-idf	39.8	17.0	32.0
UPER(ours)	**41.4**	**18.7**	**33.8**

Table 5: ROUGE-F1 scores under the **full-data** setting (100% training data for abstractive models) with the same backbone abstractive model LED and different extractive methods on WCEP dataset.

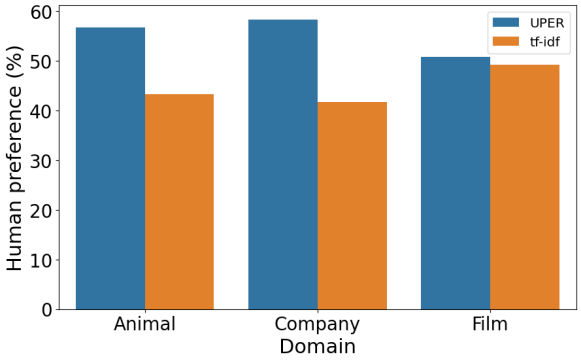

Figure 5: Human evaluation on WikiCatSum test set. `tf-idf` and `UPER` is two BART trained with corresponding extractor.

eralization ability for different abstractors; (2) On both WikiCatSum and WCEP datasets, UPER can improve abstractive models' performance, showing our model's robustness for different datasets; (3) UPER brings more improvements to BART than LED, in that LED's input length is 15 times more than BART's, which means our extractive methods only re-rank the documents for LED other than extracting a shorter document for BART.

Human Evaluation. Aiming to examine the factual correctness of generated abstracts, we follow (Liu et al., 2021c) to conduct a human evaluation by asking crowdworkers to annotate which model generates better results. For each domain, 20 examples are randomly sampled for 3 participants' opinions. We report the proportion of systems preferred by participants in figure 5. Results show that our model improves the quality of the abstracts generated by BART in all three domains. Compared with other domains, the improvement in *Film* appeared marginal, in that humans are more familiar with films and sensitive to film errors. Meanwhile, UPER still managed to improve the performance.

4.4 Ablation and Case study

4.4.1 Ablation Study

#Templates	R1	R2	RL
0	34.3	15.5	32.5
1	35.4	16.7	33.5
5	**35.6**	**17.2**	**33.6**
10	35.3	16.4	33.4

Table 6: ROUGE-F1 scores using different number of templates in salience estimation under few-shot setting. Here we just adopt $\lambda = 1$.

Use multiple prompts or not? We design the general prompting template and many special templates for the specific domain. One may argue that these special domain prompts hamper the generality of our model. So we conduct experiments on the number of prompt templates. Table 6 shows

the ablation study on the animal domain of WikiCatSum with 1% training data for the abstractive model BART. The result shows that if we don't use prompting templates, the performance of UPER will be undermined seriously. However, increasing the number of additional templates does not necessarily improve the performance. So we use one general prompting template in other experiments.

λ	R1	R2	RL
0	45.1	27.1	43.0
0.25	45.6	28.4	43.6
0.5	46.0	**28.8**	43.9
0.75	**46.1**	**28.8**	**44.0**
1	42.0	23.8	40.0

Table 7: ROUGE-F1 scores of different λ for BART on the animal domain of WikiCatSum with full data.

Is λ important? We make grid search in {0,0.25,0.5,0.75,1}. In fact, when $\lambda = 0$, the extractor $g^F(\mathcal{K}, d)$ will become the statistical extractor. Moreover, when $\lambda = 1$, the extractor $g^F(\mathcal{K}, d)$ will become the GPT extractor which only evaluates semantic salience. Table 7 shows the ablation study results on the animal domain when the number of max input tokens is 500. We find that 0.75 is a reasonable number of λ while the statistical extractor ($\lambda = 0$) and the GPT extractor ($\lambda = 1$) are not able to outperform any ensemble extractor ($0 < \lambda < 1$). The success of ensemble extractors demonstrates that applying our semantic evaluation model as a plug-in to other metrics can boost the performance. Therefore, it is necessary to consider both the lexical and the semantic salience when designing the extractor.

#Tokens	Extractor	R1	R2	RL
500	tf-idf	45.1	27.1	43.0
500	UPER	46.1	28.8	44.0
1000	tf-idf	45.9	28.1	43.8
1000	UPER	**46.4**	**29.1**	**44.4**

Table 8: ROUGE-F1 scores of different max input tokens and extractor for BART on the animal domain of WikiCatSum with the full-data setting.

The number of input tokens for abstractors. As the input length limit of the abstractor is different for BART and LED. The top K documents we feed into the abstractor will be truncated to the number of max input tokens. We tried 500 and 1000 input tokens for BART, whose max input length is 1024. Table 8 shows the ablation study on the animal domain. UPER outperforms the tf-idf with two different input token numbers. Furthermore, the fewer input tokens, the larger advantage UPER has over tf-idf.

#Tokens	Extractor	R1	R2	RL
2048	random	36.7	17.0	34.7
2048	tf-idf	41.3	22.3	39.1
2048	UPER	41.3	22.4	39.1
4096	random	39.0	19.1	36.8
4096	tf-idf	41.6	23.0	39.4
4096	UPER	41.8	22.8	39.6
16384	random	40.9	21.5	38.7
16384	tf-idf	42.3	**23.6**	40.1
16384	UPER	**42.5**	**23.6**	**40.3**

Table 9: ROUGE-F1 scores of different max input tokens and extractor for LED trained on the film domain of WikiCatSum with the full-data and early-stopping setting.

Our UPER can also improve the abstractor's performance for the abstractive model without input limits like LED. As shown in Table 9, from limited 2048 input tokens to the unlimited 16384 tokens, UPER can steadily optimize the final abstract against randomly ordered input documents. However, tf-idf and UPER almost achieve equal success in improving the unlimited input length model LED compared to the significant promotion brought by UPER on the limited input length model BART. This phenomenon reflects that the salient information in the extracted meta-document is more influential for the limited input length models because they can only accept a small part of the long input. While the unlimited input length model can receive the entire input, the extractor only plays the role of re-ranking documents.

4.4.2 Case Study

We sample an example from the animal domain to analyze. As shown in Table 10, the gold abstract is mainly composed of information from three topics: *taxonomy*, *distribution*, and *description*.

The abstract generated by BART+`tf-idf` and BART+`UPER` both covers all three topics in the gold abstract. But BART+`tf-idf` contains more factual errors on the wingspan information. It also outputs unrelated words about the larvae, which

Gold Abstract: syndemis musculana is a moth of the family tortricidae . it is found in europe , china (heilongjiang , jilin , inner mongolia) , the korean peninsula , japan , russia (amur) and north america . the wingspan is 15 - 22 mm . the adults fly from april to july in the temperate parts of their range , such as belgium and the netherlands . the caterpillars feed on oaks (quercus) , birches (betula) , spruces (picea) , ragworts (senecio) and rubus (brambles and allies) . less usually , they have been recorded to eat plant refuse and dry leaves .
BART+tf-idf: syndemis musculana is a moth of the family tortricidae . it is found in europe , china (heilongjiang , jilin , inner mongolia) , the korean peninsula , japan , russia (amur) and north america . the wingspan is 16 - 21 mm . adults are on wing from july to september . there is one generation per year . in the north , the larvae feed on archips similis . larvae can be found from june to july .
BART+UPER: syndemis musculana is a moth of the family tortricidae . it is found in europe , china (heilongjiang , jilin , inner mongolia) , the korean peninsula , japan , russia (amur) and north america . the wingspan is 15 - 22 mm . adults are on wing from april to july in the temperate parts of their range , such as belgium and the netherlands . they are active from july to october , overwintering as a full .

Table 10: Comparison between abstracts generated by BART with different extractors (tf-idf and UPER) about the animal *syndemis musculana* on WikiCatSum dataset.

are not mentioned in the gold abstract. Therefore, our perplexity-based extractor UPER can provide the abstractive model with a reliable information source, thus avoiding generating wrong or redundant words.

5 Related Work

Multi-document summarization (MDS) aims to generate an abstract for the related documents collected from referred websites or search engines. This task is commonly regarded as a two-stage problem (Liu et al., 2018; Zhu et al., 2021). The extractive stage selects high-quality relevant texts as sources, and the abstractive stage summarizes them into an abstract of required length. Prior work mostly focuses on improving the latter abstrative stage with various techniques like topic information (Perez-Beltrachini et al., 2019; Zhu et al., 2021), graph representation (Li et al., 2020) and attention (Perez-Beltrachini and Lapata, 2021). The exploration of the extractive stage is limited to a few methods like tf-idf (Liu et al., 2018) and ROUGE scorer (Liu et al., 2019), and they both focus on the token-level lexical similarity, while we take the semantic salience into consideration, which suppresses the intrinsic noise in the corpus.

Automatic evaluation metrics are vital for the extractive stage of multi-document summarization task, which can select source documents. They can be divided into two classes: referenced and reference-free. Referenced metrics usually focus on the lexical overlap (Papineni et al., 2002; Banerjee and Lavie, 2005) or embedding similarity (Zhao et al., 2019; Liu et al., 2021b) between the document and the reference summary. While reference-free metrics usually evaluate the document without reference summary using perplexity (Brown et al., 1992) or aspects evaluation (Ke et al., 2022). Our method is inspired by a referenced metric proposed by Bajaj et al. (2021). They train a scoring model to predict the perplexity of the sequence formed by concatenating the reference summary with the input document. In Section 3.2, we find that the pre-trained language model like GPT can be utilized to evaluate documents' semantic salience without training, so we propose our reference-free metric UPER.

6 Conclusion

This work investigates the extractive stage of the Multi-document Summarization task. We propose a simple but effective approach UPER to model the semantic contextual salience and combine the lexical token-level similarity to extract the input documents. UPER utilizes unsupervised prompts to take advantage of prior knowledge distributed in PLMs so that we can convert our extraction task to PLM's original perplexity calculation task. In our future work, we will extend our framework to single-document summarization and explore the application of prompt-based methods in the supervised learning scenario.

Acknowledgement

This work is supported by the New Generation Artificial Intelligence of China (2020AAA0106501), the NSFC Youth Project (62006136) and a grant from the Institute for Guo Qiang, Tsinghua University (2019GQB0003).

References

Ahsaas Bajaj, Pavitra Dangati, Kalpesh Krishna, Pradhiksha Ashok Kumar, Rheeya Uppaal, Bradford Windsor, Eliot Brenner, Dominic Dotterrer, Rajarshi Das, and Andrew McCallum. 2021. Long document summarization in a low resource setting using pretrained language models. In Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing: Student Research Workshop, pages 71–80, Online. Association for Computational Linguistics.

Satanjeev Banerjee and Alon Lavie. 2005. Meteor: An automatic metric for mt evaluation with improved correlation with human judgments. In Proceedings of the acl workshop on intrinsic and extrinsic evaluation measures for machine translation and/or summarization, pages 65–72.

Hangbo Bao, Li Dong, Furu Wei, Wenhui Wang, Nan Yang, Xiaodong Liu, Yu Wang, Jianfeng Gao, Songhao Piao, Ming Zhou, and Hsiao-Wuen Hon. 2020. Unilmv2: Pseudo-masked language models for unified language model pre-training. In Proceedings of the 37th International Conference on Machine Learning, ICML 2020, 13-18 July 2020, Virtual Event, volume 119 of Proceedings of Machine Learning Research, pages 642–652. PMLR.

Iz Beltagy, Matthew E Peters, and Arman Cohan. 2020. Longformer: The long-document transformer. arXiv preprint arXiv:2004.05150.

Lidong Bing, Piji Li, Yi Liao, Wai Lam, Weiwei Guo, and Rebecca Passonneau. 2015. Abstractive multi-document summarization via phrase selection and merging. In Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers), pages 1587–1597, Beijing, China. Association for Computational Linguistics.

Peter F Brown, Stephen A Della Pietra, Vincent J Della Pietra, Jennifer C Lai, and Robert L Mercer. 1992. An estimate of an upper bound for the entropy of english. Computational Linguistics, 18(1):31–40.

Alexander Fabbri, Irene Li, Tianwei She, Suyi Li, and Dragomir Radev. 2019. Multi-news: A large-scale multi-document summarization dataset and abstractive hierarchical model. In Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics, pages 1074–1084, Florence, Italy. Association for Computational Linguistics.

Angela Fan, Claire Gardent, Chloé Braud, and Antoine Bordes. 2019. Using local knowledge graph construction to scale Seq2Seq models to multi-document inputs. In Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP), pages 4186–4196, Hong Kong, China. Association for Computational Linguistics.

Demian Gholipour Ghalandari, Chris Hokamp, Nghia The Pham, John Glover, and Georgiana Ifrim. 2020. A large-scale multi-document summarization dataset from the Wikipedia current events portal. In Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, pages 1302–1308, Online. Association for Computational Linguistics.

Chris Hokamp, Demian Gholipour Ghalandari, Nghia The Pham, and John Glover. 2020. Dyne: Dynamic ensemble decoding for multi-document summarization. arXiv preprint arXiv:2006.08748.

Fred Jelinek, Robert L Mercer, Lalit R Bahl, and James K Baker. 1977. Perplexity—a measure of the difficulty of speech recognition tasks. The Journal of the Acoustical Society of America, 62(S1):S63–S63.

Guolin Ke, Qi Meng, Thomas Finley, Taifeng Wang, Wei Chen, Weidong Ma, Qiwei Ye, and Tie-Yan Liu. 2017. Lightgbm: A highly efficient gradient boosting decision tree. In Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, December 4-9, 2017, Long Beach, CA, USA, pages 3146–3154.

Pei Ke, Hao Zhou, Yankai Lin, Peng Li, Jie Zhou, Xiaoyan Zhu, and Minlie Huang. 2022. Ctrleval: An unsupervised reference-free metric for evaluating controlled text generation. arXiv preprint arXiv:2204.00862.

Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Veselin Stoyanov, and Luke Zettlemoyer. 2020. BART: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. In Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, pages 7871–7880, Online. Association for Computational Linguistics.

Wei Li, Xinyan Xiao, Jiachen Liu, Hua Wu, Haifeng Wang, and Junping Du. 2020. Leveraging graph to improve abstractive multi-document summarization. In Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, pages 6232–6243, Online. Association for Computational Linguistics.

Chin-Yew Lin. 2004. ROUGE: A package for automatic evaluation of summaries. In Text Summarization Branches Out, pages 74–81, Barcelona, Spain. Association for Computational Linguistics.

Pengfei Liu, Weizhe Yuan, Jinlan Fu, Zhengbao Jiang, Hiroaki Hayashi, and Graham Neubig. 2021a. Pre-train, prompt, and predict: A systematic survey of prompting methods in natural language processing. ArXiv, abs/2107.13586.

Peter J. Liu, Mohammad Saleh, Etienne Pot, Ben Goodrich, Ryan Sepassi, Lukasz Kaiser, and Noam Shazeer. 2018. Generating wikipedia by summarizing long sequences. In 6th International Conference on Learning Representations, ICLR 2018, Vancouver, BC, Canada, April 30 - May 3, 2018, Conference Track Proceedings. OpenReview.net.

Ruibo Liu, Jason Wei, and Soroush Vosoughi. 2021b. Language model augmented relevance score. In Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers), pages 6677–6690, Online. Association for Computational Linguistics.

Yang Liu and Mirella Lapata. 2019. Hierarchical transformers for multi-document summarization. In Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics, pages 5070–5081, Florence, Italy. Association for Computational Linguistics.

Yang Liu, Sheng Shen, and Mirella Lapata. 2021c. Noisy self-knowledge distillation for text summarization. In Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, pages 692–703, Online. Association for Computational Linguistics.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Roberta: A robustly optimized bert pretraining approach. ArXiv, abs/1907.11692.

Ziming Mao, Chen Henry Wu, Ansong Ni, Yusen Zhang, Rui Zhang, Tao Yu, Budhaditya Deb, Chenguang Zhu, Ahmed H Awadallah, and Dragomir Radev. 2021. Dyle: Dynamic latent extraction for abstractive long-input summarization. arXiv preprint arXiv:2110.08168.

Kathleen McKeown and Dragomir R Radev. 1995. Generating summaries of multiple news articles. In Proceedings of the 18th annual international ACM SIGIR conference on Research and development in information retrieval, pages 74–82.

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. Bleu: a method for automatic evaluation of machine translation. In Proceedings of the 40th annual meeting of the Association for Computational Linguistics, pages 311–318.

Laura Perez-Beltrachini and Mirella Lapata. 2021. Multi-document summarization with determinantal point process attention. Journal of Artificial Intelligence Research, 71:371–399.

Laura Perez-Beltrachini, Yang Liu, and Mirella Lapata. 2019. Generating summaries with topic templates and structured convolutional decoders. In Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics, pages 5107–5116, Florence, Italy. Association for Computational Linguistics.

Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, Ilya Sutskever, et al. 2019. Language models are unsupervised multitask learners. OpenAI blog, 1(8):9.

Juan Ramos et al. 2003. Using tf-idf to determine word relevance in document queries. Proceedings of the first instructional conference on machine learning, 242(1):29–48.

Yun-Zhu Song, Yi-Syuan Chen, and Hong-Han Shuai. 2022. Improving multi-document summarization through referenced flexible extraction with credit-awareness. arXiv preprint arXiv:2205.01889.

Manzil Zaheer, Guru Guruganesh, Kumar Avinava Dubey, Joshua Ainslie, Chris Alberti, Santiago Ontanon, Philip Pham, Anirudh Ravula, Qifan Wang, Li Yang, et al. 2020. Big bird: Transformers for longer sequences. Advances in Neural Information Processing Systems, 33:17283–17297.

Yusen Zhang, Ansong Ni, Ziming Mao, Chen Henry Wu, Chenguang Zhu, Budhaditya Deb, Ahmed H Awadallah, Dragomir Radev, and Rui Zhang. 2021. Summˆ n: A multi-stage summarization framework for long input dialogues and documents. arXiv preprint arXiv:2110.10150.

Wei Zhao, Maxime Peyrard, Fei Liu, Yang Gao, Christian M. Meyer, and Steffen Eger. 2019. MoverScore: Text generation evaluating with contextualized embeddings and earth mover distance. In Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP), pages 563–578, Hong Kong, China. Association for Computational Linguistics.

Ming Zhong, Da Yin, Tao Yu, Ahmad Zaidi, Mutethia Mutuma, Rahul Jha, Ahmed Hassan Awadallah, Asli Celikyilmaz, Yang Liu, Xipeng Qiu, and Dragomir Radev. 2021. QMSum: A new benchmark for query-based multi-domain meeting summarization. In Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language

Technologies, pages 5905–5921, Online. Association for Computational Linguistics.

Fangwei Zhu, Shangqing Tu, Jiaxin Shi, Juanzi Li, Lei Hou, and Tong Cui. 2021. TWAG: A topic-guided Wikipedia abstract generator. In Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers), pages 4623–4635, Online. Association for Computational Linguistics.

Xu Zou, Da Yin, Qingyang Zhong, Hongxia Yang, Zhilin Yang, and Jie Tang. 2021. Controllable generation from pre-trained language models via inverse prompting. arXiv preprint arXiv:2103.10685.

A Template details for salience estimation

For each source document, we assign at least one basic template's perplexity score. The basic prompting template should fit the context of all documents. So we use a general sentence: "This document is about k" to insert into the introduction or conclusion position of the document.

If a conclusion about the keyword's attribute can inferred from the document or the document contains salient information to answer a question about the keyword, then we assume that the document is salient. Therefore, we design two domain-specific template types for the extractive stage of Multi-document Summarization task: *conclusion* and *question* in Table 11. Besides, we also tried using no prompting template, we call it `none_prompt` .

Besides basic general prompting templates, we also design special templates for each domain to create more features for calculating the perplexity. Take animal domain for example, the $\texttt{conclusion}_1$ to $\texttt{conclusion}_4$ and $\texttt{question}_0$ to $\texttt{question}_9$ in Table 11 are the additional special templates.

B Implementation details

In the extractive stage, we load GPT2 checkpoint from *transformers* library[3] to calculate the perplexity of each filled prompting template. It costs about 15 hours to finish scoring one domain's documents on a single NVIDIA GeForce RTX 2080. For tf-idf, we use the *nltk* library[4] to count the term frequency and iverse document frequency. For random order, we shuffle the input documents randomly and send them into abstractive models.

In the abstractive stage, since the max input length of LED is 16384 which is much larger than BART's 1024 limit, their memory and time cost is quite different for fine-tuning. We train BART on a single NVIDIA GeForce RTX 2080 for 1 day (16 epochs) and train LED on a single NVIDIA GeForce RTX 3090 for 5 days (5 epochs). The learning rate is 1e-4 for the first epoch and decays to 1e-5 for other epochs. During interface, we use beam search decoding strategy with a beam size of 16, a minimum decoding length of 55, and a maximum decoding length of 120.

Note that we conduct the ablation study on the input token numbers of LED using early-stopping setting, where the training process will stop at the second epoch. Because the training LED until fitting costs too much GPU time and our GPU resource is limited, we have to compare the LED model under low-resource setup in the ablation study.

[3]https://huggingface.co/docs/transformers/index

[4]https://www.nltk.org/

Type	Template
none_prompt	d
conclusion$_0$	d This document is about k.
conclusion$_1$	d k's distribution is mentioned in above sentences.
conclusion$_2$	d This document introduces subspecies of k.
conclusion$_3$	d This document describes k.
conclusion$_4$	d This document introduces conservation status of k.
question$_0$	Where does k live? d
question$_1$	What is the Taxonomy of k? d
question$_2$	What are the Species of k? d
question$_3$	What are the Subspecies of k? d
question$_4$	What does the k feed on? d
question$_5$	Where does k live? d
question$_6$	What is the Diet of k? d
question$_7$	What is the Behaviour of k? d
question$_8$	What is the Breeding of k? d
question$_9$	What is the Conservation Status of k? d

Table 11: The general and special domain templates on the animal domain of WikiCatSum dataset. Notice that none_prompt is used as a backbone which represents the document without adding any prompts. And the conclusion$_0$ is the general template. Other templates are additional templates designed especially for the animal domain.

DISK: Domain-constrained Instance Sketch for Math Word Problem Generation

Tianyang Cao[1]*, **Shuang Zeng**[1,2]*, **Xiaodan Xu**[1,3], **Mairgup Mansur**[2], **Baobao Chang**[1]†
[1]Key Laboratory of Computational Linguistics, MOE, Peking University
[2]Tencent Inc.
[3]School of Software and Microelectronics, Peking University
{ctymy, zengs, diane1968, chbb}@pku.edu.cn, mairgupma@tencent.com

Abstract

A math word problem (MWP) is a coherent narrative which reflects the underlying logic of math equations. Successful MWP generation can automate the writing of mathematics questions. Previous methods mainly generate MWP text based on inflexible pre-defined templates. In this paper, we propose a neural model for generating MWP text from math equations. Firstly, we incorporate a matching model conditioned on the domain knowledge to retrieve a MWP instance which is most consistent with the ground-truth, where the domain is a latent variable extracted with a domain summarizer. Secondly, by constructing a Quantity Cell Graph (QCG) from the retrieved MWP instance and reasoning over it, we improve the model's comprehension of real-world scenarios and derive a domain-constrained instance sketch to guide the generation. Besides, the QCG also interacts with the equation encoder to enhance the alignment between math tokens (e.g., quantities and variables) and MWP text. Experiments and empirical analysis on educational MWP set show that our model achieves impressive performance in both automatic evaluation metrics and human evaluation metrics.

1 Introduction

Text generation has been broadly studied as an important task in the field of natural language processing. It aims to generate natural language text that is fluent, readable and faithful to the original input. Recent text generation studies mainly focus on the data-to-text generation, which generates textual output from structured data such as tables of records or knowledge graphs (KGs) (Puduppully et al., 2019a; Chen et al., 2019; Gong et al., 2019b; Zhao et al., 2020). In this paper, we focus on a relatively new type of data-to-text generation task: generating Math Word Problems (MWP) from equations (Zhou and Huang, 2019), which does not seem to have been fully studied by the community. Figure 1 shows two examples of this task. We aim to automatically generate coherent narratives which reflect the computational relationship within given equations. Successful math word problems generation has the potential to automate the writing of mathematics questions given equations to be solved. It can alleviate the burden of school teachers and further help improve the teaching efficiency.

Equations: $x = 6 * y$; $(x + y) * 3 = 147$
Problem: Jane travels ***6*** times faster than mike. Traveling in opposite directions they are ***147*** miles apart after ***3*** hrs. ***Find their rates of travel.***

Equations: $(1 - 1/3 - 9/20) * x = 245$
Problem: At a local high school, ***3/8*** of the students are freshman. ***1/4*** are juniors. And ***245*** are seniors. ***Find the total number of students.***

Figure 1: Two examples selected from the MWP generation dataset.

However, different from other data-to-text generation tasks, generating MWP text from abstract math equations is much more challenging. Firstly, an equation can be expressed by different MWP texts which differ in topic, style or grammar, known as one-to-many pattern. Take the Equation 1 in Figure 1 for example, "$x = 6 * y$" can be expressed as "Jane travels 6 times faster than Mike.", but it is also okay to express it as "The price of oranges is six times the apples.". So when grounding the input abstract math equations into a specific math problem, it is hard for a model to decide which scenes to choose for generation. Secondly, the math tokens in equations and natural language text in problems are from completely different symbolic representation space. So this gap increases the difficulty of establishing alignments between math tokens and natural language words, as shown in Figure 2. Such issue also confuses the generator thus makes generation process uncontrollable.

To overcome these challenges, we propose a

* Equal contribution.
† Corresponding author.

Proceedings of the 29th International Conference on Computational Linguistics, pages 6327–6339
October 12–17, 2022.

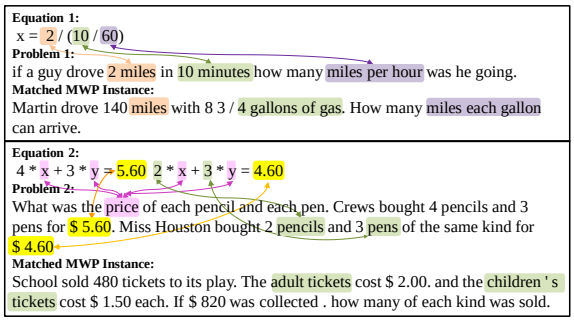

Figure 2: Two examples for illustration of the model. The corresponding MWP instance of each example is selected by a domain-dominated retriever with the participation of domain knowledge. The soft alignments between the math tokens and the corresponding MWP text spans are labeled in the same color.

novel MWP generator with **D**omain-constrained **I**nstance **SK**etch (**DISK**). We point out that MWP generation task can benefit from instance sketch restricted by domains. For two examples in Figure 2, DISK first utilizes a domain-dominated retriever to select an instance from a set of candidate MWP text. This instance can be regarded as to constrain the scene with which the math problem text to be generated would be related. Then, DISK produces a refined instance sketch from this instance to specify certain patterns for generating MWP text, using a Quantity Cell Graph (QCG) constructed from the instance. This graph reflects the backbone of the instance with entities and actions related to quantities as nodes. We conduct reasoning over QCG using Graph Convolutional Network (Kipf and Welling, 2017) to extract the final instance sketch, where math tokens can be contextualized with corresponding attributes and predicates via interaction between QCG nodes and equations. Finally, the model can generate MWP text with the refined instance sketch using a sequence generator.

Our contributions can be summarized as follows:

- We propose a domain-constrained instance sketch guided MWP text generation model, in which the domain information correspoding to the MWP text is automatically induced.
- Our model generates the instance sketch via Quantity Cell Graph enhanced encoding, it also contextualizes the math tokens with corresponding attributes and predicates via interaction between QCG nodes and equations.

Experiments show our model can generate more fluent and domain consistent MWP text, with promising performance improvement over strong baselines.

2 Related Work

MWP Generation: Early MWP generation methods are mostly template-based, including Answer Set Programming (ASP) (Polozov et al., 2015), schema and frame semantics (Singley and Bennett, 2002; Deane, 2003). With the development of deep learning framework, Zhou and Huang (2019); Wang et al. (2021) generate problem text given equation templates and keywords, where the keywords are extracted from the golden MWP via heuristic rules. Their model is learned with Seq2seq in an end-to-end manner and integrates features of templates and keywords in the decoding phase. Their model, however, requires keywords from the golden answer as input when testing, which is unavailable in real scenarios. And this paper focuses on the MWP generation solely from equations without keywords which is more suitable for practical scenarios. Another work Liu et al. (2020) adopts the external commonsense based knowledge graph (CSKG) to generate topic relevant sentences, while this method only considers the cases of linear equations and needs annotated topics for each equation.

Data-to-text Generation: Data-to-text generation transforms structured data into descriptive texts (Siddharthan, 2001; Gatt and Krahmer, 2018). Recent works have brought great promising performance to several data-to-text generation tasks, e.g., Puduppully et al. (2019a,b); Gong et al. (2019a); Wiseman et al. (2017) focus on report generation; Chisholm et al. (2017); Lebret et al. (2016) target at biography generation; Zhao et al. (2020); Gao et al. (2020) generate texts from a set of RDF triples considering structural information. Previous works have also designed content selection and text planning models to determine what to say and how to say (Puduppully et al., 2019a; Perez-Beltrachini and Lapata, 2018).

Retrieval-based Generation: The methods similar to our instance-based generation are the skeleton-then-response frameworks which are popular in dialogue response generation (Cai et al., 2019; Wu et al., 2019; Yu and Jiang, 2021; Cai et al., 2020). These models usually treat the input text as a query and the similar query along with its response in databases is then retrieved with In-

formation Retrieval (IR) systems. However, they rely on difference between the input query and retrieved query to identify informative words in the retrieved response. Thus existing retrieval-based methods can not be employed in our task since it is meaningless to measure the similarity between equations.

3 Methodology

The overview of our DISK is depicted in Figure 3. Our model follows a three-stage procedure: Firstly, given the input equations and a text-domain vector which is extracted by the **Domain Summarizer**, the **Matching Model** retrieves a most similar MWP instance in database by jointly measuring equation-text matching score and domain-text matching score. Secondly, the **Sketch Provider** enriches the original representation of the instance to yield a refined instance sketch, it filters out excessive information considering domain constraint and helps the model to understand quantity relationship by applying Graph Neural Network (GNN) over the Quantity Cell Graph (QCG). Thirdly, **Text Generator** generates MWP text via utilizing both the math equation contextualized by QCG and the refined instance sketch based on an encoder-decoder architecture.

3.1 Domain Summarizer

The domain summarizer takes the MWP text $y = \{y_i\}_{i=1}^{L}$ with length L as input and its goal is to collect underlying domain information in the MWP text, which contributes to instance retrieving. To this end, inspired by (Huang et al., 2018; Keskar et al., 2019), we assume K latent domains are depicted by the MWP text with different importance β_i and the text-domain vector can be expressed as the weighted sum of K trainable domain vectors.

We start by encoding the MWP text into a sequence of vectors via a transformer block:

$$[\boldsymbol{h}_1; \boldsymbol{h}_2; ..., \boldsymbol{h}_L] = Encoder_P([\boldsymbol{y}_1; \boldsymbol{y}_2; ...; \boldsymbol{y}_L]) \tag{1}$$

where $[;]$ denotes concatenation operation. We denotes $[\boldsymbol{h}_1; \boldsymbol{h}_2; ...; \boldsymbol{h}_L]$ as $\boldsymbol{H} \in \mathbb{R}^{L \times d}$, d is the embedding size. A global attention is applied to the output of the transformer:

$$\boldsymbol{h}_a = \sum_{i=1}^{L} \alpha_i \boldsymbol{h}_i \tag{2}$$

where the attention weight $\alpha_i = \text{softmax}(\boldsymbol{h}_i \boldsymbol{W}^a \overline{\boldsymbol{h}})$, $\overline{\boldsymbol{h}} = \frac{1}{L} \sum_{i=1}^{L} \boldsymbol{h}_i$. A non-linear transformation is utilized to fuse the encoded MWP text into K domain variables:

$$\tilde{\boldsymbol{D}} = tanh(\boldsymbol{W}_1^D(\boldsymbol{H}\boldsymbol{W}_2^D + \boldsymbol{b}_2^D) + \boldsymbol{b}_1^D) \tag{3}$$

$\tilde{\boldsymbol{D}} \in \mathbb{R}^{K \times d}$ and parameters $\boldsymbol{W}_1^D \in \mathbb{R}^{K \times L}$, $\boldsymbol{W}_2^D \in \mathbb{R}^{d \times d}$. Each row vector in $\tilde{\boldsymbol{D}}$ corresponds to a different domain contained in MWP text. Such process can be treated as a soft clustering and we hope each domain expresses its unique aspect. Similar to Luxburg (2004), we employ an auxiliary loss function to restrict the derived K domain representation to be orthogonal with each other:

$$\mathcal{L}_D = ||\tilde{\boldsymbol{D}}\tilde{\boldsymbol{D}}^T - \boldsymbol{I}_{K \times K}|| \tag{4}$$

$\boldsymbol{I}_{K \times K}$ is an identity matrix. We then map $\tilde{\boldsymbol{D}}$ and $\boldsymbol{h}_a$ to a domain distribution with an attention mechanism (Bahdanau et al., 2014):

$$\beta_i = \text{Softmax}(\boldsymbol{v}^T tanh(\boldsymbol{H}^d \boldsymbol{h}_a + \boldsymbol{U}^d \tilde{\boldsymbol{D}}_{i,:})) \tag{5}$$

where $\boldsymbol{v}, \boldsymbol{H}^d, \boldsymbol{U}^d$ are learnable parameters. β_i indicates the domain distribution of the given MWP. Our model then learns a trainable domain embedding $\boldsymbol{E} \in \mathbb{R}^{K \times d}$ and uses $\{\beta_i\}_{i=1}^{K}$ to compute the text-domain vector over $\boldsymbol{E}$: $\boldsymbol{h}_d = \sum_{i=1}^{K} \beta_i \boldsymbol{E}_{i,:}$. Note that the domain summarizer only works during training process. During test, we enumerate each discrete domain vector in $\boldsymbol{E}$ to be fed into the matching model, which will be illustrated later.

3.2 Matching Model

The matching model aims to match one MWP instance from the training corpus which is most consistent with the given equation. Intuitively, incorporating the domain variable helps our model better recognize MWP text with similar domain grounding to the golden problem text, since it's difficult to infer from the equation only. Thus it's rational to combine the text-domain vector and the math equation to retrieve an additional instance.

Our matching model ranks all MWP texts from a pre-defined set $P = \{P_1, P_2, ..., P_{|P|}\}$, and returns the most consistent one with given equation-MWP pair (x, y), where P is prepared by uniformly sampling from the training corpus. The text-domain vector $\boldsymbol{h}_d$ and equation embedding $\{\boldsymbol{x}_i\}_{i=1}^{N}$ serve as input, where $\boldsymbol{x}_i$ is the sum of corresponding token embedding and type embedding, here type embedding is incorporated to distinguish quantities, numbers and operations in math equations. Each text $P_i = p_1^i p_2^i ... p_{|P_i|}^i$ is encoded

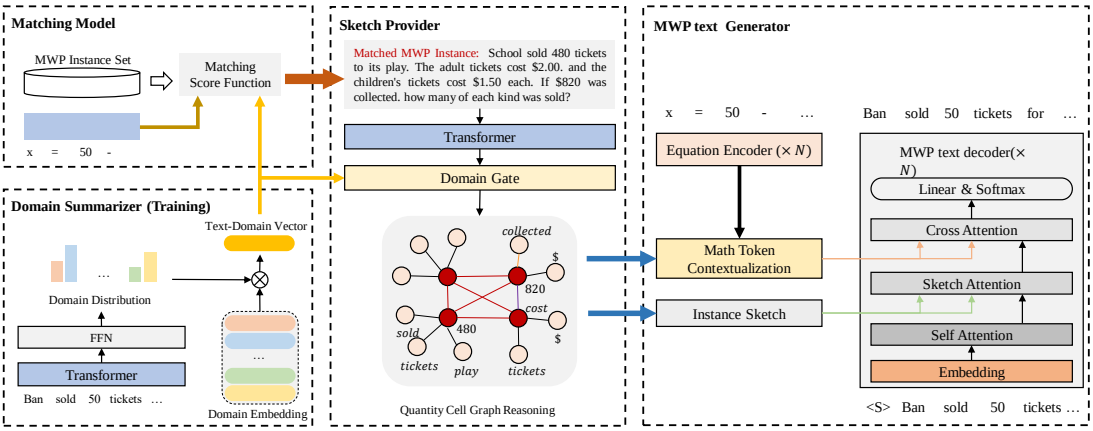

Figure 3: The diagram of proposed DISK. The text-domain vector is the summation of domain embeddings. First, the matching model predicts the most consistent MWP instance based on the equation tokens together with the text-domain vector. Then, the sketch provider learns to refine the retrieved MWP instance with a domain gate and the Quantity Cell Graph, it also contextualizes the equation representation to help the model understand the alignment between equations and MWP text. Finally, the generator consumes both the instance sketch and contextualized equation representation for generating.

into context representation $\left\{\boldsymbol{u}_j^i\right\}_{j=1}^{|P_i|}$ through transformer blocks (denoted as $Encoder_Q$). However, the dataset provides no supervision for the matching model, we then annotate the golden labels by ranking the BERTScore (Zhang* et al., 2020) between each candidate MWP text and the ground-truth MWP, i.e., BERTScore(P_i, y) $1 \leq i \leq |P|$, getting the top one P_{l^*} as the selected instance.

Equation to MWP Matching: Equation to MWP matching score is measured in token-level. Firstly, we encode $\{\boldsymbol{x}_i\}_{i=1}^N$ into $\boldsymbol{C} = \{\boldsymbol{c}_i\}_{i=1}^N$ via a transformer block, $[\boldsymbol{c}_1, \boldsymbol{c}_2, ..., \boldsymbol{c}_N] = Encoder_E([\boldsymbol{x}_1, \boldsymbol{x}_2, ..., \boldsymbol{x}_N])$. A nonlinear function is then used to compute the correlation score between the j-th token in the equation and the k-th token of P_i:

$$\gamma_{i,j,k} = g_1(\boldsymbol{c}_j) \cdot g_2(\boldsymbol{u}_k^i) \tag{6}$$

where $g_1(\cdot), g_2(\cdot)$ are both multi-layer perceptrons (MLP). Next, we aggregate token level relevance to determine text-level score:

$$s_{em}(i) = \underset{j,k}{mean}\, \gamma_{i,j,k} \tag{7}$$

Domain to MWP Matching: We interact the text-domain vector $\boldsymbol{h}_d$ with the context representation of P_i: $\left\{\boldsymbol{u}_j^i\right\}_{j=1}^{|P_i|}$ to obtain the domain to MWP matching score. Similar to (2), we apply a global attention to calculate the summation of P_i, which we denote as $\boldsymbol{h}_p^i$. We then compute the domain vector to the ith MWP text relevance vector via a bilinear transformation:

$$\boldsymbol{r}_i = \boldsymbol{h}_d \boldsymbol{W}^r \boldsymbol{h}_p^i \tag{8}$$

$\boldsymbol{W}^r \in \mathbb{R}^{d \times d' \times d}$ is a parameter. Finally, we combine (7) and (8) to produce the normalized distribution over $|P|$ candidate MWP texts:

$$s(i) = \text{Softmax}(s_{em}(i) + \boldsymbol{w}^r \boldsymbol{r}_i)\ i \in [1, |P|] \tag{9}$$

where $\boldsymbol{w}^r \in \mathbb{R}^{d'}$ is a learnable parameter. With the label l^* annotated, we add a cross entropy loss to supervise the result of the matching model:

$$\mathcal{L}_M = -\log(s(l^*)) \tag{10}$$

For inference, the top one MWP text with the highest matching score is selected to be fed into the next sketch provider.

3.3 Sketch Provider

The sketch provider aims to generate the instance sketch by making soft modification to the MWP instance representation, since the generative model should capture underlying patterns contained in the instance, not simply copy the instance. We achieve this goal in two aspects: 1) we add a domain gate to refine tokens that have high relevance with the domain information, 2) we incorporate the Quantity Cell Graph to enable our model to better understand complex question scenarios while

maintaining those spans semantically aligned with equation tokens.

Firstly, for the encoded representation of P_{l^*} : $\{\boldsymbol{u}_i\}_{i=1}^{|P_{l^*}|}$ (which has been processed by $Encoder_Q$ in the matching model), we employ a soft gate controlled by the domain vector $\boldsymbol{h}_d$ to better flow the important context in the original matched text:

$$\begin{aligned} \boldsymbol{q}_i &= \sigma(\boldsymbol{W}^q [\boldsymbol{h}_d; \boldsymbol{u}_i]) \\ \boldsymbol{u}'_i &= \tanh(\boldsymbol{W}^Q \boldsymbol{u}_i) \odot \boldsymbol{q}_i \;\; 1 \le i \le |P_{l^*}| \end{aligned} \tag{11}$$

Quantity Cell Graph Constructing and Reasoning Targeting at better exploiting the retrieved MWP instance, we should enrich the instance encoding with quantity relationship information, as well as effectively guide the alignment between abstract equation tokens and MWP text tokens. Inspired by Zhang et al. (2020), we introduce the Quantity Cell Graph (QCG), whose nodes contain a subset of tokens in the the MWP text related to numerical values. As is shown in Fig 3, a Quantity Cell Graph is composed of a set of Quantity Cells (QC): $QCG = \{QC_1, QC_2, ..., QC_m\}$, where m denotes the number of quantities in the matched MWP instance P_{l^*}. Each cell QC_i can be expressed as $\{v_i^q\} \bigcup \left\{v_{i,1}^a, v_{i,2}^a, ...\right\}$, where v_i^q is the ith quantity token and $v_{i,1}^a, v_{i.2}^a, ...$ is the corresponding attributes or predicates. We resort to Dependency Tree[1] and Constituency Tree [2] to extract attributes related to each quantity token. Details can be found in Appendix B. We argue that the extracted tokens are salient properties related to quantities and show explicit alignment with the input equations. With the nodes in the Quantity Cell Graph mentioned above, we add an edge between two nodes if 1) they are both quantity nodes, 2) one is the quantity node and another is the attribute node belonging to it. Next, we initialize the node representation of the QCG by concatenating the corresponding output of $Encoder_Q$ and its POS tag embedding, which is denoted as $\boldsymbol{S}^0 = \{\boldsymbol{s}_k\}_{k=1}^{|G|}$, $|G|$ is the node number of the QCG. Graph Convolutional Network (Kipf and Welling, 2017) is applied to capture the dependencies between QCG nodes:

$$\boldsymbol{S}^{l+1} = ReLU(GCN(\boldsymbol{S}^l, \boldsymbol{A})) \tag{12}$$

where $\boldsymbol{S}^l$ is the node representation after the l-th layer, $\boldsymbol{A} \in \{0,1\}^{|G|\times|G|}$ is the adjacency matrix.

[1]https://demo.allennlp.org/dependency-parsing

[2]https://demo.allennlp.org/constituency-parsing

Graph2Text Augmentation After graph network reasoning, we need a fusion block to propagate the aggregated information of the QCG back to the text representation $\boldsymbol{U}' = \{\boldsymbol{u}'_i\}_{i=1}^{|P_{l^*}|}$. To locate the position of each node in the original matched text, we establish a binary matrix $\boldsymbol{M} \in \{0,1\}^{|P_{l^*}|\times|G|}$, where $\boldsymbol{M}_{ij} = 1$ if the i-th token in the MWP instance is the j-th node in the graph. As each column of $\boldsymbol{M}$ corresponds to one quantity node or attribute node in the QCG, we update $\boldsymbol{u}'_i$ with a GRU module if the i-th token participates in the QCG reasoning:

$$\tilde{\boldsymbol{U}} = GRU(\left[\boldsymbol{U}'; \boldsymbol{M}\boldsymbol{S}^{\mathcal{L}}\boldsymbol{W}^U\right]) \tag{13}$$

where $\boldsymbol{S}^{\mathcal{L}}$ is the output of the last GCN layer and $\boldsymbol{W}^U$ is a parameter matrix. $\tilde{\boldsymbol{U}}$ is treated as the output instance sketch.

Math Token Contextualization As mentioned before, the attribute words related to quantities are beneficial to help the model identify the soft alignment pattern between the math equation tokens and retrieved MWP instance. The encoded vector sequence of the input equations $\boldsymbol{C}$ attends to the QCG nodes to receive graph information:

$$\begin{aligned} \boldsymbol{G} &= \text{Softmax}(\boldsymbol{C}\boldsymbol{W}^G\boldsymbol{S}^{\mathcal{L}}) \\ \overline{\boldsymbol{C}} &= ReLU(\boldsymbol{G}\boldsymbol{S}^{\mathcal{L}}) \end{aligned} \tag{14}$$

We then calculate two update gate $\boldsymbol{f} = \sigma(\boldsymbol{W}^g \left[\boldsymbol{C}; \overline{\boldsymbol{C}}\right])$ and $\boldsymbol{g} = \sigma(\boldsymbol{W}^f \left[\boldsymbol{C}; \overline{\boldsymbol{C}}\right])$, which combines $\boldsymbol{C}$ and $\overline{\boldsymbol{C}}$ to obtain contextualized equation representation $\tilde{\boldsymbol{C}}$:

$$\tilde{\boldsymbol{C}} = \boldsymbol{g} \odot \boldsymbol{C} + (\boldsymbol{1} - \boldsymbol{g}) \odot tanh(\boldsymbol{W}^Z \left[\boldsymbol{C}; \boldsymbol{f} \odot \overline{\boldsymbol{C}}\right]) \tag{15}$$

3.4 MWP Generator

The MWP generator maps the math equation tokens $x_1 x_2 ... x_N$ to the MWP text, we employ a transformer based encoder-decoder architecture. Here our encoder shares its parameters with $Encoder_E$ in matching model to capture common attention features among them. To enable the decoder to rewrite the domain-constrained instance sketch produced by the sketch provider in a fine-grained manner, we insert an extra sketch attention layer between the original self-attention layer and cross-attention layer. It aggregates details of the sketch by attending to output of sketch provider $\tilde{\boldsymbol{U}}$:

$$\boldsymbol{H}' = MultiHeadAttn(\mathbf{Q} : \boldsymbol{H}_p, \mathbf{K} : \tilde{\boldsymbol{U}}, \mathbf{V} : \tilde{\boldsymbol{U}}) \tag{16}$$

where $\boldsymbol{H}_p$ is the hidden state coming from the previous layer. Residual connection and layernorm is also added after the sketch attention. For the cross attention layer, the new representation $\tilde{\boldsymbol{C}}$ coming from Math Token Contextualization module is used as both the key and value. The hidden state of the last decoder layer is projected to vocabulary distribution and predicts the next token. The domain vector $\boldsymbol{h}_d$ is directly fed into the MWP text decoder and serves as the first input embedding (instead of the embedding of a start symbol <S>). The generation loss can then be modeled as:

$$\mathcal{L}_G = -\sum_{t=1}^{L} \log p(y_t | y_{<t}, \{x_i\}_{i=1}^{N}, \tilde{\boldsymbol{U}}, \boldsymbol{h}_d) \quad (17)$$

$y_{<t}$ is the tokens generated before the t-th step.

3.5 Model Training

For training, we combine the three loss terms mentioned above and the total loss becomes:

$$\mathcal{L}_{total} = \mathcal{L}_D + \mathcal{L}_M + \mathcal{L}_G \quad (18)$$

For inference, our model traverses over all K possible latent domain vectors in $\boldsymbol{E}$ and generates K candidate MWP texts $Y^1, Y^2, ..., Y^K$, among which the problem text with the maximum log likelihood score is chosen as the final output.

4 Experiments

4.1 Dataset

Our dataset is based on Dolphin18K (Huang et al., 2016) crawled from Yahoo. Since Huang et al. (2016) only releases a subset of Dolphin18K (3154 examples), which is insufficient for a modern data-driven generation model. So we reuse the python scripts provided by Huang et al. (2016) to crawl and collect extra data, then the size of dataset is extended to 14943 examples. We conduct some data preprocessing by deleting those equation-problem text pairs whose problem text length is longer than 45 tokens or less than 10 tokens. Finally 9643 samples are preserved. More detailed statistics of the dataset are listed in Appendix C.

4.2 Baselines

We compare DISK against the following models. 1) **Seq2seq** (Bahdanau et al., 2014): Seq2seq is first proposed for machine translation task. In this paper, we implement Seq2seq with attention mechanism and copy mechanism. 2) **SeqGAN** (Yu et al., 2016): SeqGAN fuses the advantage of reinforcement learning (RL) and Generative Adversarial Network (GAN). It achieves improvements over strong baselines in both text generation and music generation tasks. 3) **DeepGCN** (Guo et al., 2019): Math equations can be converted into a pre-ordered expression tree and MWP generation can then be naturally modeled as graph-to-sequence learning. 4) **Transformer** (Vaswani et al., 2017): The state of-the-art model in several text generation tasks. 5) **DualCG** (Wei et al., 2019): In this paper we employ DualCG to integrate equation to MWP generation and MWP to equation solving in a unified framework. 6) We also compare our model with the vanilla **BART** (Lewis et al., 2020), which is a strong pretrained model using the standard seq2seq Transformer architecture. We fine-tune BART on our MWP dataset. Note we do not use retrieval-based baselines since they mostly require IR system, while it's unsuitable to treat math symbols as the query.

4.3 Automatic Evaluation

We compare different methods using BLEU (average of BLEU-1 and BLEU-2) (Papineni et al., 2002), ROUGE-L (Lin, 2004), METEOR (Banerjee and Lavie, 2005), BERTScore (Zhang* et al., 2020), which is an advanced evaluation metric for text generation based on contextual embedding, Dist-1, Dist-2, which indicates the proportion of different unigrams (bigrams) in all unigrams (bigrams), Number Recall, which is used to measure how many numbers in problem text are correctly copied. We report the performance of all models in terms of automatic evaluation in Table 1.

We also conduct ablation studies and the results are also reported in Table 1. The setting is as follows: 1) w/o DG: We remove the domain gate in the Sketch Provider. 2) w/o QCG: We remove both the QCG reasoning and the Math Tokens Contextualization block. The encoded MWP instance, after being processed by the domain gate, is directly fed into the generator in this case. 3) w/o MTC: The model without Math Token Contextualization. 4) w/o CS: The model without the instance sketch, i.e., the whole Matching Model and Sketch Provider are removed and only the Domain Summarizer and the Generator are preserved.

It can be observed that 1) our proposed model significantly outperforms the strongest DualCG in BLEU, ROUGE-L and BERTScore, respectively.

Model	BLEU	ROUGE-L	BERTScore	METEOR	Dist1(%)	Dist2(%)	NR(%)
Seq2seq (Bahdanau et al., 2014)	2.59	20.25	82.98	18.51	14.56	34.99	47.60
SeqGAN (Yu et al., 2016)	2.62	19.22	82.56	17.63	12.96	30.02	44.00
DeepGCN (Guo et al., 2019)	3.04	20.94	83.07	19.48	16.81	45.17	49.21
Transformer (Vaswani et al., 2017)	3.14	21.84	83.81	20.26	12.94	43.51	44.84
DualCG (Wei et al., 2019)	3.60	21.43	83.99	20.63	15.47	46.01	40.97
BART$_{large}$ (Lewis et al., 2020)	4.15	22.26	**86.35**	22.30	12.77	46.76	43.47
DISK	**5.84**	**28.49**	85.01	**27.16**	**15.56**	**50.41**	52.62
w/o DG	5.33	27.36	84.90	25.33	10.74	35.62	49.37
w/o QCG	4.86	26.87	84.96	26.56	13.88	45.19	55.15
w/o MTC	5.14	28.08	84.62	26.16	13.63	43.90	55.51
w/o CS	4.59	26.96	84.27	25.32	12.53	43.67	**57.68**

Table 1: Automatic evaluation results of different models in MWP generation dataset. NR is the abbreviation for Number Recall. All results are averaged for five runs.

It yields higher results in most language quality metrics even when compared with BART$_{large}$. Besides, our model also improves the informativeness and diversity of generated MWP. 2) simply letting the MWP decoder attend to the retrieved MWP (w/o QCG) will degrade the performance by 16.78%, 5.69%, 0.06% in BLEU, ROUGE-L and BERTScore, respectively, which proves the Quantity Cell Graph can guide the generator to understand the quantity relationship and better exploit the retrieved MWP instance. It's notable that since BART maintains exact distributed word representation, it may show advantage in contextual embedding based metrics such as BERTScore. Moreover, as an intermediate result, we report the capacity of the matching model in Appendix F.

4.4 Performance on Different Types of Equations

Table 2 shows the performance on different subsets of the MWP generation dataset (divided by the number of variables). We can see the proposed method outperforms baselines by a large margin in all subsets. Intuitively, the more variables the equation contains, the more imperatively the generation process needs the guidance of instance sketch. It's easy to show our model obtains more absolute gain in More Than Three-VAR subset than One-AVR or Two-VAR ones.

4.5 Human Evaluation

To better measure the actual generation quality, we recruit three human annotators to judge the quality of different models, which includes four aspects listed as follows. 1) **Fluency**: Fluency mainly judges whether the problem text is fluent, i.e., whether some grammar errors occur in generated MWP. 2) **Coherence**: Coherence weights if the problem text is consistent in text-level. 3) **Solvability-1 (S1)**: As our target is a math word problem, we should pay attention to whether the problem text can be solved, i.e., in what percentage we can set up the same (or equivalent) equations and solve them according to the generated problem text. 4) **Solvability-2 (S2)**: Solvability-2 is a more relaxed criterion compared with Solvability-1, it only requires the text produced is a valid math problem and could be solved regardless what equations could be set.

We randomly select 100 generated MWP texts and score them in five grades. We then project the scores to 1∼5, where higher scores indicate better performance. Moreover, we assess the inter-annotator agreement by Cohen's kappa κ, which reflects the agreement between scores given by different raters. The averaged results are reported in Table 3. We can clearly see that the proposed model

		BLEU	ROUGE-L	BERTScore
One-VAR	**DualCG**	2.88	19.99	83.59
	Trans	2.18	19.43	83.45
	BART$_{large}$	3.16	19.83	85.94
	DISK	3.87	27.17	84.45
		BLEU	**ROUGE-L**	**BERTScore**
Two-VAR	**DualCG**	3.75	23.05	84.41
	Trans	3.97	22.82	84.52
	BART$_{large}$	4.77	25.26	87.00
	DISK	6.33	29.03	85.51
		BLEU	**ROUGE-L**	**BERTScore**
More Than Three-VAR	**DualCG**	2.00	17.74	84.33
	Trans	3.33	21.10	83.15
	BART$_{large}$	1.86	17.16	84.97
	DISK	4.59	25.63	84.26

Table 2: Performance on different subsets on our MWP generating dataset. Trans is short for Transformer.

	Fluency		Coherence		S1(%)	S2(%)
	score	κ	score	κ		
DISK	**4.00**	0.413	**4.08**	0.497	**36**	**56**
Seq2seq	3.78	0.256	3.48	0.483	23	34
SeqGAN	3.75	0.305	3.28	0.520	20	40
DeepGCN	3.61	0.295	3.55	0.494	29	52
Transformer	3.80	0.333	3.53	0.421	20	45
DualCG	3.88	0.346	3.66	0.455	28	53
BART$_{large}$	3.56	0.398	3.73	0.454	31	52

Table 3: Human evaluation results: comparison between the proposed model and baseline models.

Equ: $equ : 250 + 400 = x \; equ : 1625/x = y$ **MT**: 2 vehicles traveling different directions. same start point and time. one vehicle is 60 mph. the other is 55 mph. In how many hours will they be 500 miles apart. **Ours**: Two cars leave Denver traveling in opposite directions. One has a speed of 250 mph and the other airplane averages 400 mph. How many hours will the trip be 1625 miles apart. (*BLEU*: 15.47) **Seq2seq**: A <UNK> of deposit costs $ 400. 000 a t the end of the year. the total interest is $ 1625 . 00. What is the total cost of the total. **SeqGAN**: quotient of a certain number is 400. If the number of students in the first 250 is 400. What is the number. **DeepGCN**: The car ran a t an average speed of 400 km per hour faster than the other. If the speed of a 400 mi / h faster. What was the speed of the plane in miles per hour. **Trans**: planes went to school a t a speed for the trip takes 250 mph for 400 hours. How long will the plane travel in the trip. **DualCG**: Joe received 250 miles for 250 miles . and gas a trip of 250 miles per hour for $ 400 to the week. He drove 400 miles per hour faster . What was the average speed for the trip.
Equ: $equ : x + y = 35 \; equ : x/y = 2/5$ **MT**: total of 1600 people work for a company. The ratio of male to female employees is 3 : 5. How many more females than males are there in the company. **Ours**: The ratio of boys to girls in at a certain school is 5 : 2. If there are total 35 boys and girls. how many of each are there. (*BLEU*: 9.09) **Seq2seq**: The school art club is having a exhibit. The ratio of the school paintings are in two parts is 2 / 5 of the number. What is the number ? **SeqGAN**: A carpet is 3 times as many more than the other. The total value is 3. **DeepGCN**: The ratio of the larger of the two numbers is 35. The ratio of the smaller number of goals and the other is 5 / 2. What are the two numbers. **Trans**: Pat . 35 students and 5 questions. If the total of the students are seniors and 2 take both the total. how many of each. **DualCG**: The sum of two numbers is 35. The larger number is 2 less than the smaller number. Find the larger number.

Table 4: Two examples of math word problems generated by different models. Transformer is abbreviate to Trans. Equ and MT represent the equation and the matched MWP instance, respectively. Quantity-related attributes and predicates in the instance that are picked up and rewritten in the generated MWP are colored for better readability.

performs much better than other models, not only in fluency and coherence, but also in the solvability of generated math word problems.

4.6 Effectiveness of the Domain Gate and Quantity Cell Graph

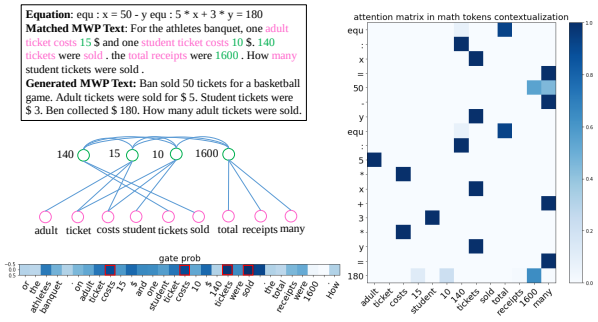

Figure 4: Visualization of a test case, which shows 1) the retrieved MWP text and the generated MWP text 2) the extracted QCG from the retrieved MWP text 3) the value of the domain gate on different tokens in (11) 4) the attention matrix between the input equation representation and the QCG, namely, $\boldsymbol{G}$ in (14)

We show the effectiveness of the domain-dominated soft gate and the Quantity Cell Graph Reasoning through qualitative analysis. Fig 4 presents a test case processed by our model. The heatmap in the left lower corner indicates the relevance of each token to the text-domain vector in the matched MWP text. The top-4 tokens are marked with the red box. Words that highlight the characteristic of one certain domain, e.g., "tickets", "sold", "cost"... are assigned with higher weight to be fused into the next block. The heatmap on the right hand side presents the probability that each equation token attends to the nodes in the QCG (after normalization). It is easy to show: number "5" is aligned with "adult", since "5" is the price of adult tickets; number "3" is aligned with "student", since "3" is the price of student tickets; both "x" and "y" are aligned with "tickets", since "x" and "y" both imply the number of tickets; "180" is aligned with "1600" as 1600 is the total receipts in the matched MWP instance and "180" also refers to the total sales in our generated text. It's reasonable to believe that the math token contextualization enhances the semantic alignment between math equations and the matched MWP instance.

4.7 Case Study

Table 9 shows two examples in the test dataset generated by different models. Additional examples can be found in Appendix G. We can observe that: 1) DISK gives consistent context in text-level while keeping readability, which verifies it's effective to assign a domain vector to each MWP text. 2) The generated MWP text expresses plausible attributes related to quantities by making an analogy with the matched instance. E.g., in case 2, the matched text discusses the ratio of male employees and female employees, while the MWP generated by our model says "the ratio of boys to girls". Besides, it's interesting that though in case 2, the output given by our system receives low BLEU score, it's still a logically reasonable and feasible MWP. So BLEU score may not be suitable for evaluating the performance of MWP generation. According to the above analysis, it is obvious that instance sketch provider improves the informativeness of the given equation via correctly understanding and exploiting the connections among QCG nodes.

5 Conclusion

We propose DISK, which introduces latent discrete domains for matching appropriate MWP instance and refines its representation. We also extract Quan-

tity Cell Graph to enhance the sketch-guided generator and help our model better understand math equations in real scenarios. Experimental results on the extended Dolphin18K Dataset show the superiority of our model.

Acknowledgement

This paper is supported by the National Science Foundation of China under Grant No.61876004 and 61936012, the National Key R&D Program of China under Grand No.2018AAA0102003.

References

Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. 2014. Neural machine translation by jointly learning to align and translate. *arXiv: Computation and Language*.

Satanjeev Banerjee and Alon Lavie. 2005. METEOR: an automatic metric for MT evaluation with improved correlation with human judgments. In *Proceedings of the Workshop on Intrinsic and Extrinsic Evaluation Measures for Machine Translation and/or Summarization@ACL 2005, Ann Arbor, Michigan, USA, June 29, 2005*, pages 65–72. Association for Computational Linguistics.

Deng Cai, Yan Wang, Wei Bi, Zhaopeng Tu, Xiaojiang Liu, Wai Lam, and Shuming Shi. 2019. Skeleton-to-response: Dialogue generation guided by retrieval memory. In *NAACL*, pages 1219–1228. Association for Computational Linguistics.

Hengyi Cai, Hongshen Chen, Yonghao Song, Xiaofang Zhao, and Dawei Yin. 2020. Exemplar guided neural dialogue generation. In *Proceedings of the Twenty-Ninth International Joint Conference on Artificial Intelligence, IJCAI 2020*, pages 3601–3607. ijcai.org.

Shuang Chen, Jinpeng Wang, Xiaocheng Feng, Feng Jiang, Bing Qin, and Chin-Yew Lin. 2019. Enhancing neural data-to-text generation models with external background knowledge. In *EMNLP-IJCNLP*, pages 3022–3032. Association for Computational Linguistics.

Andrew Chisholm, Will Radford, and Ben Hachey. 2017. Learning to generate one-sentence biographies from wikidata. 1:633–642.

Paul Deane. 2003. Automatic item generation via frame semantics: Natural language generation of math word problems.

Hanning Gao, Lingfei Wu, Po Hu, and Fangli Xu. 2020. Rdf-to-text generation with graph-augmented structural neural encoders. In *Proceedings of the Twenty-Ninth International Joint Conference on Artificial Intelligence, IJCAI 2020*, pages 3030–3036. ijcai.org.

Albert Gatt and Emiel Krahmer. 2018. Survey of the state of the art in natural language generation: Core tasks, applications and evaluation. *J. Artif. Intell. Res.*, pages 65–170.

Heng Gong, Xiaocheng Feng, Bing Qin, and Ting Liu. 2019a. Table-to-text generation with effective hierarchical encoder on three dimensions (row, column and time). *arXiv: Computation and Language*.

Li Gong, Josep Crego, and Jean Senellart. 2019b. Enhanced transformer model for data-to-text generation. In *Proceedings of the 3rd Workshop on Neural Generation and Translation*, pages 148–156. Association for Computational Linguistics.

Zhijiang Guo, Yan Zhang, Zhiyang Teng, and Wei Lu. 2019. Densely connected graph convolutional networks for graph-to-sequence learning.

Chenyang Huang, Osmar Zaiane, Amine Trabelsi, and Nouha Dziri. 2018. Automatic dialogue generation with expressed emotions. pages 49–54.

Danqing Huang, Shuming Shi, Chin-Yew Lin, Jian Yin, and Wei-Ying Ma. 2016. How well do computers solve math word problems? large-scale dataset construction and evaluation. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 887–896. Association for Computational Linguistics.

Nitish Keskar, Bryan McCann, Lav Varshney, Caiming Xiong, and Richard Socher. 2019. Ctrl: A conditional transformer language model for controllable generation.

Diederik P Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. *arXiv: Learning*.

Thomas N. Kipf and Max Welling. 2017. Semi-supervised classification with graph convolutional networks. In *ICLR*. OpenReview.net.

Nate Kushman, Yoav Artzi, Luke Zettlemoyer, and Regina Barzilay. 2014. Learning to automatically solve algebra word problems. In *ACL*, pages 271–281. Association for Computational Linguistics.

Remi Lebret, David Grangier, and Michael Auli. 2016. Neural text generation from structured data with application to the biography domain. pages 1203–1213.

Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Veselin Stoyanov, and Luke Zettlemoyer. 2020. BART: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7871–7880. Association for Computational Linguistics.

Chinyew Lin. 2004. Rouge: A package for automatic evaluation of summaries. pages 74–81.

Tianqiao Liu, Qian Fang, Wenbiao Ding, Zhongqin Wu, and Zitao Liu. 2020. Mathematical word problem generation from commonsense knowledge graph and equations. *CoRR*, abs/2010.06196.

Ulrike Luxburg. 2004. A tutorial on spectral clustering. *Statistics and Computing*, 17:395–416.

Kishore Papineni, Salim Roukos, Todd Ward, and Weijing Zhu. 2002. Bleu: a method for automatic evaluation of machine translation. pages 311–318.

Laura Perez-Beltrachini and Mirella Lapata. 2018. Bootstrapping generators from noisy data. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 1516–1527. Association for Computational Linguistics.

Oleksandr Polozov, Eleanor O'Rourke, Adam M. Smith, Luke Zettlemoyer, Sumit Gulwani, and Zoran Popović. 2015. Personalized mathematical word problem generation. In *IJCAI 2015*.

Ratish Puduppully, Li Dong, and Mirella Lapata. 2019a. Data-to-text generation with content selection and planning. *AAAI 2019*, 33(01):6908–6915.

Ratish Puduppully, Li Dong, and Mirella Lapata. 2019b. Data-to-text generation with entity modeling. *ACL 2019*, pages 2023–2035.

Shuming Shi, Yuehui Wang, Chin-Yew Lin, Xiaojiang Liu, and Yong Rui. 2015. Automatically solving number word problems by semantic parsing and reasoning. In *EMNLP*. Association for Computational Linguistics.

Advaith Siddharthan. 2001. Ehud reiter and robert dale. *Building Natural Language Generation Systems*. cambridge university press, 2000. $64.95/£37.50 (hardback), 234 pages. *Nat. Lang. Eng.*, (3):271–274.

Mark Singley and Randy Bennett. 2002. Item generation and beyond: Applications of schema theory to mathematics assessment.

Shyam Upadhyay and Ming-Wei Chang. 2017. Annotating derivations: A new evaluation strategy and dataset for algebra word problems. In *EACL*. Association for Computational Linguistics.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Lukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. pages 5998–6008.

Zichao Wang, Andrew Lan, and Richard Baraniuk. 2021. Math word problem generation with mathematical consistency and problem context constraints. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 5986–5999. Association for Computational Linguistics.

Bolin Wei, Ge Li, Xin Xia, Zhiyi Fu, and Zhi Jin. 2019. Code generation as a dual task of code summarization. In *NeurIPS*, pages 6559–6569.

Sam Wiseman, Stuart M Shieber, and Alexander M Rush. 2017. Challenges in data-to-document generation. pages 2253–2263.

Yu Wu, Furu Wei, Shaohan Huang, Yunli Wang, Zhoujun Li, and Ming Zhou. 2019. Response generation by context-aware prototype editing. In *AAAI*, pages 7281–7288. AAAI Press.

Lantao Yu, Weinan Zhang, Jun Wang, and Yong Yu. 2016. Seqgan: Sequence generative adversarial nets with policy gradient. *arXiv: Learning*.

Xiaojing Yu and Anxiao Jiang. 2021. Expanding, retrieving and infilling: Diversifying cross-domain question generation with flexible templates. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*, pages 3202–3212, Online. Association for Computational Linguistics.

Jipeng Zhang, Lei Wang, Roy Ka-Wei Lee, Yi Bin, Yan Wang, Jie Shao, and Ee-Peng Lim. 2020. Graph-to-tree learning for solving math word problems. In *ACL*, pages 3928–3937. Association for Computational Linguistics.

Tianyi Zhang*, Varsha Kishore*, Felix Wu*, Kilian Q. Weinberger, and Yoav Artzi. 2020. Bertscore: Evaluating text generation with bert. In *ICLR*. OpenReview.net.

Chao Zhao, Marilyn Walker, and Snigdha Chaturvedi. 2020. Bridging the structural gap between encoding and decoding for data-to-text generation. In *ACL*, pages 2481–2491, Online. Association for Computational Linguistics.

Qingyu Zhou and Danqing Huang. 2019. Towards generating math word problems from equations and topics. *INLG 2019*, pages 494–503.

A Task Definition

Our system maps a set of equations $\{E_1, E_2, ..., E_{|E|}\}$ to the MWP text: $\boldsymbol{y} = y_1 y_2 ... y_L$ which reflects logic of equations. Each equation consists of a sequence of math tokens: $E_k = x_1 x_2 ... x_{|E_k|}$, where $|E_k|$ is the length of k-th equation measured by the number of math tokens. Each math token belongs to one of the following three types: math operator (e.g., $+, -, *, \div, ...$), number (e.g., $0.2, 1, 30, ...$), variable (e.g., $x, y, z, ...$). L is the the length of problem text. Our objective is to estimate the following conditional probability depending on

equations and previously generated words $\boldsymbol{y}_{<t}$:

$$P(\boldsymbol{y}|\boldsymbol{x}) = \prod_{t=1}^{L} P(y_t|\boldsymbol{y}_{<t}, E_1, E_2, ...) \quad (19)$$

B Details on Constructing Quantity Cell Graph

In this section, we describe the rules for extracting quantity-related attributes as follows.

- We consider those tokens which are labeled as *Nouns* or *Verbs* and are within two hops starting from the quantity token in the dependency tree.
- We firstly traverse the nodes in the constituency tree starting from the root node in a depth-first manner, and backtracks when the visited node contains no more than F (F is a hyperparameter) leaf nodes. Such operation yields several subtrees and each token in the original text belongs and only belongs to one subtree. We detect the tokens belonging to the same subtree as the quantity token and are labeled as *Nouns* or *Verbs*.

C Dataset

Dataset Information: Table 5 provides our data statistics.

Motivation of Extending Dolphin18K: MWP solving datasets currently used include Alg514 (Kushman et al., 2014), Dolphin1878 (Shi et al., 2015), DRAW-1K (Upadhyay and Chang, 2017), Dolphin18K (Huang et al., 2016). Table 6 gives the statistic of these datasets. Alg514, Dolphin1878, DRAW-1K are all public available, while neural generation models for generative tasks are usually data-hungry thus equation-MWP pairs in those datasets are insufficient. Though Dolphin18K is a large-scale dataset, only a part of it (3154) are released. Moreover, existing datasets only include a certain type of MWP text, e.g., MWP text for linear equations, which restricts their practical application. We then reuse the python script provided by Huang et al. (2016) and acquire 14943 equation-MWP text pairs in total from Yahoo !. Generally, the public available datasets can be treated as the subset of our dataset. Next, we conduct data preprocess as follows, which is beneficial to train the generation model:

- We normalize the equations by replacing all the equation variables in each sample to x, y, z,... in order, e.g., $u + v + r = 100, u - r = 10$ is replaced to $x+y+z = 100,\ x-z = 10$.
- We manually correct the wrong spelling words in MWP text.

	Train	Dev	Test
Size	7714	964	965
Equation Length (average)	16.69	16.23	16.63
Problem Length (average)	28.90	29.64	28.74
Tokens	7445	3065	2875

Table 5: Statistic of datasets

Dataset	Size	Problem Type	Avg EL	Avg Ops
Alg514	514	algebra, linear	9.67	5.69
Dolphin1878	1878	number word problems	8.18	4.97
DRAW-1K	1000	algebra, linear, one-variable	9.99	5.85
Dolphin18K	18460*	algebra, linear, multi-variable	9.19	4.96
Our Dataset	14943	algebra, linear/nonlinear, multi-variable	16.64	6.41

Table 6: Statistics of several existing MWP solving datasets. Avg EL, Avg Ops refer to average equation length and average numbers of operators in equations, respectively. $*$ indicates only 3154 equation-MWP pairs of Dolphin18K are available.

D Experimental Settings

The batch size for training is 32. The vocabulary size is set as 13k. The hidden size for both our model and baseline models is 256. We use 2 layers transformer block in our model and the baseline Transformer model. All multi-head attention layers are implemented with 8 heads. The embeddings are randomly initialized and are trained together with our model. The domain number is set as $K = 25$, however, the results for different values of K are also presented in this paper. The size of the candidate MWP set prepared for retrieving is $|P| = 500$. For extracting the QCG with constituency parser, the hyper-parameter is set as $F = 5$ and the graph network is stacked for 2 layers. To calculate the BERTScore, we use the tool released by the author on Github [3]. We train all models for 40 epoch. To prevent overfitting, we set the dropout probability as 0.2. We use the Adam optimizer (Kingma and Ba, 2014) with the learning rate $lr = 0.0005$ and momentum $\beta_1 = 0.9$, $\beta_2 = 0.999$.

E Impact of Different Domain Numbers

Fig 5 compares the fluctuation of BLEU and ROUGE-L when the number of domains changes

[3] https://github.com/Tiiiger/bert_score

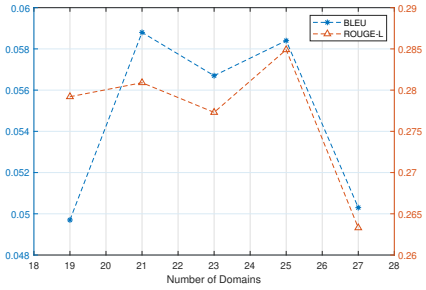

Figure 5: Performance with different domain numbers on the test dataset.

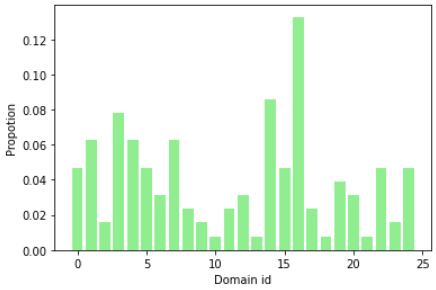

Figure 6: Proportion of test cases which is conditioned on each domain.

from 19 to 27. The proposed model receives consistent improvement compared against the baselines with different numbers of domains, while the peak value appears when $K = 21$ or $K = 25$. Even though K is set to 19 or 27, our model still exceeds baselines, which demonstrates its generalization capacity.

Moreover, one interesting problem is whether each domain plays a role during test. To this end, we investigate the percentage of output MWP text which is conditioned on each domain in the whole test set, the result is reported in Fig 6. We can find our model doesn't lead to "domain collapse", i.e., all cases are generated from the same domain, since the distribution of domains is generally balanced.

F Effectiveness of the Domain Gate and Quantity Cell Graph

We also conduct ablation study to measure our model's ability of understanding the retrieved MWP instance and properly exploiting the quantity relationship implied in it. As is shown in Table 7, we report the semantic matches in deep between the retrieved MWP instance and the generated MWP. When we discard the Quantity Cell Graph module or the Math Token Contextualization module, the relevance between the MWP instance and generated MWP both drop, which indicates the interaction between quantity-related attributes and enriching equation tokens with concrete scenarios are crucial for performance improvement.

	METEOR	BERTScore
Full Model	**24.09**	**85.01**
w/o QCG	22.70	84.79
w/o MTC	22.86	84.28

Table 7: Semantic similarity between the retrieved MWP instance and generated MWP from different models.

G Example Outputs

We show more cases produced by different models in Table 9 to demonstrate the performance of our proposed model.

H Error Analysis

We analyze and conclude the bad cases generated by our system and the typical problems are listed in Table 8. The first example shows *Logical Error*, where \$2000 is the capital, rather than the total amount of money after the period of deposit. There also exists similar errors in other instances such as the reversal of minuend and subtrahend. This shows the model ignores operation logic and underlying knowledge of constants. Consequently, auxiliary tasks such as quantity relation extraction and number sorting deserve to be considered. The second example shows *Missing Conditions*, since the total price, which is a necessary condition, is omitted in the generated MWP. This may be caused by the absence of sentence-level context planning, since one problem text usually consists of multiple sentences and each sentence conveys certain intention.

Table 8: Two examples to illustrate the drawbacks of the current method.

Equation: $equ: 2000 * (1 + 0.04)^5 = x \; equ: x - 2000 = y$	
Logical Errors:	Identify the interest earned for 5 years at the interest rate of 4% . \$ 2000 was deposited at the end of interest. How much of profit did he has.
Equation: $equ: x + y = 400 \; equ: 2 * x + 3 * y = 1050$	
Missing Conditions:	The attendance at a baseball game was 400 people. Student tickets cost \$ 2 and adult tickets cost \$ 3. The red rose theater sells tickets for \$ 4.50 and \$ 6.00. How many tickets of each type were sold.

Equ: $equ : x + y = 360\ equ : 0.18 * x + 0.09 * y = 360 * 0.17$
MT: one can drink is **15% orange juice**. another is **10% orange juice**. How many **liters** should be **mixed** in order to get 5 l that is 11% orange juice. How many liters of the 15% orange juice should be in the **mixture**.
Ours: **soybean meal** is **18% protein**. **cornmeal** is **9% protein**. How many **pounds** of each should be **mixed** together in order to get **360 lb mixture** that is 17% protein. (wh)
Seq2seq: kets to a junior high school play cost $ 3.50. and student tickets are $ 360. How many adult tickets were sold.
SeqGAN: The sum of two numbers is 360 . The sum of the numbers is 360 . What is the number.
DeepGCN: <UNK> has 360 pounds . cornmeal is 18% protein. If the total interest on at 0.09 is 360 protein. How many pounds of each should be mixed together in order to get 360 lb that is
Trans: soybean meal is 18% protein. cornmeal is 9% protein. How many grams of each should be mixed together in order to get 360 grams of the mixture.
DualCG: A chemist has a solution of 360 solution and a 20% solution of alcohol. How many liters of a solution that is the mixture that no solution to make a mixture that is 50% alcohol.

Equ: $equ : x + y = 35\ equ : x/y = 2/5$
MT: total of 1600 people work for **a company**. **The ratio of male to female employees** is 3 : 5. How many more **females than males** are there in the company.
Ours: The **ratio of boys to girls** in at **a certain school** is 5 : 2. If there are total 35 **boys and girls**. how many of each are there.
Seq2seq: The school art club is having a exhibit. The ratio of the school paintings are in two parts is 2 / 5 of the number. What is the number ?
SeqGAN: A carpet is 3 times as many more than the other. The total value is 3.
DeepGCN: The ratio of the larger of the two numbers is 35. The ratio of the smaller number of goals and the other is 5 / 2. What are the two numbers.
Trans: Pat . 35 students and 5 questions. If the total of the students are seniors and 2 take both the total. how many of each.
DualCG: The sum of two numbers is 35. The larger number is 2 less than the smaller number. Find the larger number.

Equ: $equ : 1/6 * x + 7 = 2/3 * x$
MT: **A truck's gas tank** is 1 / 4 full. After 15 **gallons of gas** are **added**, the tank is 7 / 8 full . What is the **gallon capacity** of the gas tank?
Ours: **Deandre's gas tank** is 2 / 7 full. After he **buys** 6 **gallons of gas**. it is 2 / 3 full. How many gallons can **Deandre's tank hold**.
Seq2seq: If a sum of 1 and 7 / 3 of the other digit. What is the value of x .
SeqGAN: <UNK> can paint a house in 6 hours . If the same time it takes 7 / 3 hours . How many days will it take to go 2 miles apart.
DeepGCN: The sum of the first three numbers is 7. the sum of the first number and the number is 7. the result is the same as when the result is one. Find the number
Trans: 1 / 6 of a number is 7 / 2 of the number. Find the number.
DualCG: If 1 / 6 of a number is 2 / 6. Find the number.

Equ: $equ : 250 + 400 = x\ equ : 1625/x = y$
MT: **2 vehicles traveling** different directions. same **start point** and time. one vehicle is **60 mph**. the other is **55 mph**. In how many hours will they be **500 miles apart**.
Ours: **Two cars leave Denver traveling** in opposite directions. One has a **speed of 250 mph** and the other airplane **averages 400 mph**. How many hours will the trip be **1625 miles apart**.
Seq2seq: A <UNK> of deposit costs $ 400. 000 a t the end of the year. the total interest is $ 1625 . 00. What is the total cost of the total.
SeqGAN: quotient of a certain number is 400. If the number of students in the first 250 is 400. What is the number.
DeepGCN: The car ran a t an average speed of 400 km per hour faster than the other. If the speed of a 400 mi / h faster. What was the speed of the plane in miles per hour.
Trans: planes went to school a t a speed for the trip takes 250 mph for 400 hours. How long will the plane travel in the trip.
DualCG: Joe received 250 miles for 250 miles . and gas a trip of 250 miles per hour for $ 400 to the week. He drove 400 miles per hour faster . What was the average speed for the trip.

Table 9: Four examples of math word problems generated by different models. Transformer is abbreviate to Trans. Equ and MT represents the equation and the matched MWP instance, respectively. Quantity-related attributes and predicates in the instance that are picked up and rewritten in the generated MWP are colored for better readability.

Context-Tuning: Learning Contextualized Prompts for Natural Language Generation

Tianyi Tang[1,4], **Junyi Li**[1,3], **Wayne Xin Zhao**[1,4,5] ✉ and **Ji-Rong Wen**[1,2,4]
[1]Gaoling School of Artificial Intelligence, Renmin University of China
[2]School of Information, Renmin University of China
[3]DIRO, Université de Montréal
[4]Beijing Key Laboratory of Big Data Management and Analysis Methods
[5]Beijing Academy of Artificial Intelligence, Beijing, 100084, China
steventianyitang@outlook.com lijunyi@ruc.edu.cn batmanfly@gmail.com

Abstract

Recently, pretrained language models (PLMs) have had exceptional success in language generation. To leverage the rich knowledge encoded by PLMs, a simple yet powerful paradigm is to use *prompts* in the form of either discrete tokens or continuous embeddings. In existing studies, these prompting methods are typically independent of the inputs, lacking sufficient consideration of input semantics. To address this issue, we propose a novel continuous prompting approach, called *context-tuning*, to fine-tuning PLMs for natural language generation. Firstly, the prompts are derived based on the input text to elicit useful knowledge from PLMs for generation. We refer to such prompts as *contextualized prompts*. Secondly, we use *continuous inverse prompting* to improve the process of natural language generation by modeling an inverse generation process from output to input, making the generated text more relevant to the inputs. Furthermore, we utilize a lightweight context-tuning method that fine-tunes only 0.12% of the parameters while maintaining good performance. Our code is publicly available at https://github.com/RUCAIBox/Context-Tuning.

1 Introduction

Natural language generation (*a.k.a.* text generation) aims to produce plausible and readable text in human language from input data (Li et al., 2022). Recently, large-scale pretrained language models (PLMs) such as BART (Lewis et al., 2020) have had exceptional success in language generation. To leverage the encoded knowledge from PLMs, prompting methods have been proposed (Liu et al., 2021), where the original input to PLMs has been extended by prepending discrete tokens or continuous embeddings (called *prompts*). Following this paradigm, this work aims to study how to develop more effective prompting methods for text generation based on PLMs.

Title	Live-action medium is inferior to animation medium
Static Prompts: *Write a story about:* **Title**	
Contextualized Prompts: $\boldsymbol{p}_{1\cdots k}$ **Title** $\boldsymbol{p}_{k+1\cdots 2k}$	
Story	I think that live-action works can't be considered art. They feel more like documentaries or theater pieces with CGI combined. The superiority of animated works is that they are more abstract and imaginative and characters show more emotion and variety of designs.

Table 1: Example inputs (titles) and outputs (stories) of generation dataset CMV. Static prompts are human-written instructions, which are independent of input titles. $\boldsymbol{p}_{1\cdots k}$ and $\boldsymbol{p}_{k+1\cdots 2k}$ are contextualized prompts, which are derived conditioned on the input title. The wavy line and underline denote the corresponding information between input and output.

Early methods focused on human-written (discrete) prompts by manually constructing task-specific prompt templates (Raffel et al., 2020; Radford et al., 2019), such as "TL;DR:" for the summarization task. Recent work has further proposed utilizing continuous prompts (Li and Liang, 2021; Lester et al., 2021) for text generation. Continuous prompts consist of trainable parameters that do not correspond to real tokens and can be easily optimized during fine-tuning. However, existing prompting approaches typically adopt *static prompts* for generation, *i.e.*, the prompts contain task-related information but remain the same for different input texts.

In this work, we mainly focus on challenging open-ended generation, such as story generation (Fan et al., 2018) and review generation (Li et al., 2019). Under this setting, the input text usually contains very limited information, while the task goal is to generate an output sequence containing informative contents based on the given limited input. The example in Table 1 aims to gen-

✉ Corresponding author

Proceedings of the 29th International Conference on Computational Linguistics, pages 6340–6354
October 12–17, 2022.

erate a story about the topics of "*live-action*" and "*animation*". In such a case, it requires in-depth background knowledge about the two topics. As we can see, static prompts such as "*Write a story about:*" are independent of the input title, making it difficult to capture the related aspects for this generation task. Instead of static prompts, we argue that *contextualized prompts* (as shown in Table 1) derived based on the input title will be more suited for this setting.

To address the above issues, we propose **Context-Tuning**, a novel continuous prompting approach to fine-tuning PLMs for natural language generation. Our approach has three major technical contributions. Firstly, the prompts are derived based on input text to enrich the input by eliciting related knowledge from PLMs. Specifically, by concatenating limited input and a sequence of "`[MASK]`" tokens into BERT (Devlin et al., 2019), we leverage its excellent mask-filling ability to predict these tokens, and the last hidden state of tokens can be used as prompt vectors. Since the prompts are highly related to the input context, we refer to them as *contextualized prompts*. Secondly, to further enhance the relevance between the generated text and the input text, we extend inverse prompting (Zou et al., 2021) by incorporating continuous prompts. We refer to them as *continuous inverse prompting*. By maximizing the likelihood of predicting inputs conditioned on the generated text and continuous prompts, context-tuning can generate texts highly relevant to the input text. Moreover, to ease the training burden, we propose to use a lightweight context-tuning method (Ben Zaken et al., 2022) that only fine-tunes the bias term of all model parameters. In this way, we can achieve a comparable performance (98.0% of the full-tuned performance) by only tuning 0.12% of the parameters, compared to full-tuned context-tuning.

To our knowledge, we are the first to encode input-related information into continuous prompts for text generation. Our context-tuning approach can elicit relevant knowledge according to specific input text and enhance the relevance between the generated text and the input text. We compare our method with several baseline models for the evaluation of four natural language generation tasks. Extensive experiments demonstrate the effectiveness of our proposed context-tuning approach.

2 Related Work

Natural Language Generation. Natural language generation is one of the most challenging fields in natural language processing (NLP). It aims to produce human-readable text from input text. Current state-of-the-art results for many generation tasks are based on fine-tuning PLMs, such as text summarization (Lewis et al., 2020), dialogue system (Zhang et al., 2020), and data-to-text generation (Ribeiro et al., 2021). As mentioned in Liu et al. (2021), controlled text generation is relevant to our input-dependent method. The goal of controlled text generation is to direct the generated texts into specific styles (Hu et al., 2017), lengths (Kikuchi et al., 2016), or keywords (Dou et al., 2021). In contrast, our contextualized prompts elicit knowledge from PLMs to enrich the input rather than control the specific properties of generated text.

Prompting Learning. Prompting methods prepend task instructions to the input and generate the output from PLMs. Most typical methods utilize manually designed task-specific prompts to adapt to different generation tasks (Radford et al., 2019; Raffel et al., 2020). However, it is time-consuming and laborious to construct human-written prompts for various generation tasks. As a result, recent research has concentrated on automating the search for discrete prompts (Shin et al., 2020; Gao et al., 2021). Nonetheless, searching for prompts over discrete space is challenging to optimize due to the non-differentiable issues and continuous nature of neural networks. To handle these problems, many studies propose optimizing continuous prompts (Lester et al., 2021; Li and Liang, 2021), which are more expressive and flexible for any task. Among these works, prefix-tuning (Li and Liang, 2021) and prompt tuning (Lester et al., 2021) are two representatives focused on text generation and natural language understanding (NLU) tasks, respectively. Compared with these continuous approaches, our context-tuning encodes the context information of inputs into the contextualized prompts and adopts continuous inverse prompting to enhance relevance further.

Most existing prompting methods (Schick and Schütze, 2021a; Shin et al., 2020; Lester et al., 2021) focus on NLU tasks, which are choice questions that can easily be converted into filling

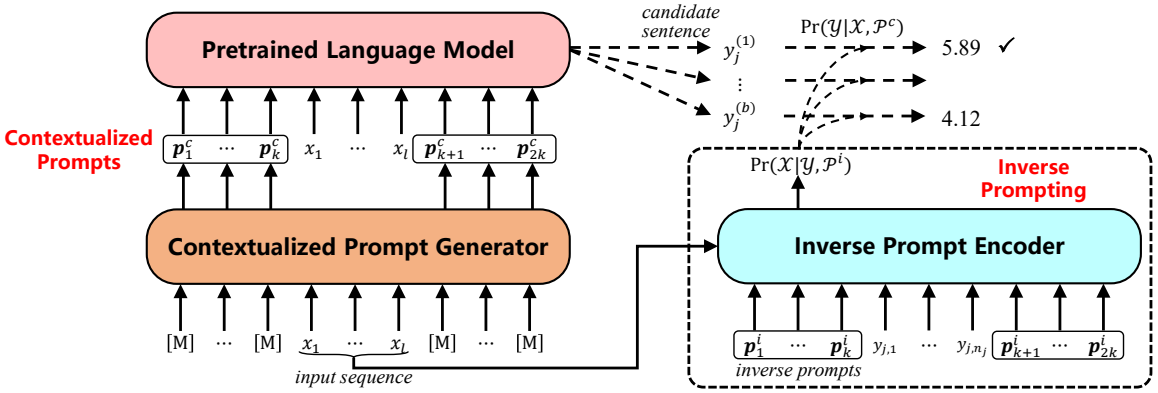

Figure 1: The overview of the proposed context-tuning. "[M]" denotes the mask token "[MASK]". By combining the forward probability $\Pr(\mathcal{Y}|\mathcal{X},\mathcal{P}^c)$ (the left part) and backward probability $\Pr(\mathcal{X}|\mathcal{Y},\mathcal{P}^i)$ (the right part), we select the sequence $y_j^{(i)}$ with the highest combined scores from all the candidates.

"`[MASK]`" tasks. However, text generation aims to generate a sequence of tokens, in contrast to a few options in limited space. For example, prefix-tuning (Li and Liang, 2021) and GENPET (Schick and Schütze, 2021b) have employed prompting methods for text generation. However, they mainly focus on lightweight fine-tuning or few-shot learning and do not achieve great performance under full tuning settings. In contrast, our context-tuning can improve performance under full tuning settings, and the lightweight strategy tunes only 0.2% of the parameters while retaining good performance.

3 The Proposed Approach

In this section, we present the proposed *context-tuning* to fine-tune PLMs for natural language generation. We first introduce the contextualized prompts based on the input text for generating informative text. To further enhance the relevance of the generated text to the input, we utilize continuous inverse prompting to enforce the prediction of inputs given the generated text and continuous prompts. Figure 1 presents an overall illustration of the proposed context-tuning approach.

For natural language generation, we consider a general task setting, where the model generates the output sequence $\mathcal{Y}$ conditioned on the input sequence $\mathcal{X} = \langle x_1, \ldots, x_l \rangle$. The output text is usually composed of multiple sentences: $\mathcal{Y} = \{y_j : \langle y_{j,1}, \ldots, y_{j,t}, \ldots, y_{j,n_j} \rangle\}_{j=1}^{m}$. In context-tuning, we introduce contextualized prompts $\mathcal{P}^c = \langle \boldsymbol{p}_1^c, \ldots, \boldsymbol{p}_k^c \rangle$ into the input side. Thus, the prompt-based generation task can be formulated as:

$$\Pr(\mathcal{Y}|\mathcal{X},\mathcal{P}^c) = \Pr(y_1, \ldots, y_m | x_1, \ldots, x_l, \mathcal{P}^c). \quad (1)$$

3.1 Contextualized Prompts

Instead of static prompts (Lester et al., 2021) (*irrelevant to input*), we use *contextualized prompts*, which are expected to provide additional information, such as world knowledge, commonsense and task information extracted from PLMs to enrich the limited input text.

Masked Prompt Learning. Specifically, unlike prefix-tuning (Li and Liang, 2021), which prepends a sequence of static vectors to each layer of PLMs, we append a sequence of k continuous vectors on both the left and right sides of the input sequence $\mathcal{X}$ ($2k$ vectors in total). Inspired by the masked language modeling task of BERT (Devlin et al., 2019), we use BERT as the prompt generator to derive the contextualized prompt vectors. We first place a sequence of k "`[MASK]`" tokens on both sides of the input $\mathcal{X}$ as:

$$\widetilde{\mathcal{X}} = \texttt{[MASK]}_{1,\ldots,k}, \mathcal{X}, \texttt{[MASK]}_{k+1,\ldots,2k}. \quad (2)$$

By feeding $\widetilde{\mathcal{X}}$ as the input of the prompt generator, we can obtain the top-layer representations of these "`[MASK]`" tokens:

$$\underbrace{\tilde{\boldsymbol{p}}_1^c, \ldots, \tilde{\boldsymbol{p}}_k^c}_{\text{prefix prompts}} / \underbrace{\tilde{\boldsymbol{p}}_{k+1}^c, \ldots, \tilde{\boldsymbol{p}}_{2k}^c}_{\text{suffix prompts}} = \text{Prompt-Generator}(\widetilde{\mathcal{X}}). \quad (3)$$

After shown in Section 4.4, we set k to 150 with the best performance. Compared with randomly-initialized prompts, our BERT-based prompt learning method can better learn the dependency between the prompts and input texts.

Aligning to Word Embeddings. Since these prompt vectors are latent embeddings, we further

align them to the semantic space of word embeddings by designing a two-step *semantic mapping* operator. For the first step, BERT predicts the probability distribution over its vocabulary based on these top-layer representations:

$$\Pr(w|\widetilde{\mathcal{X}}) = \text{softmax}(\boldsymbol{W}^V \tilde{\boldsymbol{p}}_k^c), \tag{4}$$

where $\boldsymbol{W}^V$ is a trainable matrix. For the second step, we multiply the probability distribution $\Pr(w|\widetilde{\mathcal{X}})$ with the word embedding matrix $\boldsymbol{E}$ and obtain the final contextualized prompt vectors:

$$\boldsymbol{p}_k^c = \boldsymbol{E} \cdot \Pr(w|\widetilde{\mathcal{X}}). \tag{5}$$

We consider these mapped vectors as *contextualized prompts*. Intuitively, the above *semantic mapping* can be considered as a weighted average of word embeddings according to their probabilities. Compared with existing continuous prompts, our contextualized prompt vectors can better correspond to real word embeddings in semantic space, as shown in Section 4.6.

Applying the Prompts. After obtaining the contextualized prompts, we combine these prompt vectors and the word embeddings of $\mathcal{X}$ as the input of PLMs for generating the output text $\mathcal{Y}$. Specifically, we utilize BART as the base PLM to generate text by minimizing the cross-entropy loss function:

$$\begin{aligned}\mathcal{L}_c &= -\log \Pr(\mathcal{Y}|\mathcal{X}, \mathcal{P}^c) \\ &= -\log \Pr(\mathcal{Y}|\boldsymbol{p}_{1\cdots k}^c, \boldsymbol{x}_{1\cdots l}, \boldsymbol{p}_{k+1\cdots 2k}^c),\end{aligned} \tag{6}$$

where $\boldsymbol{p}_{1\cdots k}^c$ denotes $\underline{\boldsymbol{p}_1^c, \ldots, \boldsymbol{p}_k^c}$, $\boldsymbol{x}_{1\cdots l}$ denotes $\underline{\boldsymbol{x}_1, \ldots, \boldsymbol{x}_l}$, and $\boldsymbol{p}_{k+1\cdots 2k}^c$ denotes $\underline{\boldsymbol{p}_{k+1}^c, \ldots, \boldsymbol{p}_{2k}^c}$. By leveraging the encoded knowledge from PLM, the contextualized prompts are helpful to generate informative output texts.

3.2 Continuous Inverse Prompting

Although contextualized prompts can improve the informativeness of output, it still suffers from the off-topic generation issue as the text length increases (Zou et al., 2021). To deal with this issue, we propose *continuous inverse prompting* to enhance the relevance in an inverse manner from output to input. Compared to the previous inverse prompting that depends on artificial construction (Zou et al., 2021), our inverse prompting is based on continuous prompts, which can be flexibly optimized during fine-tuning.

Algorithm 1 The algorithm procedure for generation process of context-tuning.

Require: Model parameters $\Theta^{(c)}$ and $\Theta^{(i)}$, beam size b and maximum number of sentences n_m
Input: An input sequence $\mathcal{X}$
Output: A generated sequence $\mathcal{Y}$
Initialize step $j = 0$
while $j < n_m$ **do**
  Derive contextualized prompts $\mathcal{P}^c$ based on $\mathcal{X}$
  Generate b candidate sentences $y_j^{(1)}, \ldots, y_j^{(b)}$ according to Eq. 6
  Utilize continuous inverse prompts $\mathcal{P}^i$ to compute the likelihood of candidate sentences according to Eq. 7
  Choose the best sentence as y_s based on Eq. 8
  Terminate the loop if y_s contains the end of sentence token
  Update $j = j + 1$
end while
Concatenate $y_1, \ldots, y_j$ as generated sequence $\mathcal{Y}$
return $\mathcal{Y}$

Output-to-Input Relevance Enhancement. To model the relevance of output $\mathcal{Y}$ to input $\mathcal{X}$, we hypothesize that the output text is highly relevant to the input text if we can recover the input based on the output. Nevertheless, in some text generation tasks, it is non-intuitive to generate the input text given the output text. Hence, we utilize prompts to mitigate this issue. We introduce continuous inverse prompts $\mathcal{P}^i$ and append them on both sides of the output $\mathcal{Y}$. Then, we utilize another PLM to measure the conditional probability $\Pr(\mathcal{X}|\mathcal{Y}, \mathcal{P}^i)$. Considering the output text $\mathcal{Y}$ might be much longer than the input text $\mathcal{X}$, we further model the probability at the sentence level:

$$\begin{aligned}\mathcal{L}_i &= -\log \Pr(\mathcal{X}|\mathcal{Y}, \mathcal{P}^i) \\ &= -\sum_{j=1}^{m} \log \Pr(\mathcal{X}|\boldsymbol{p}_{1\cdots k}^i, \boldsymbol{y}_{j,1}, \ldots, \boldsymbol{y}_{j,n_j}, \boldsymbol{p}_{k+1\cdots 2k}^i),\end{aligned} \tag{7}$$

where $\boldsymbol{p}_{1\cdots k}^i$ denotes $\underline{\boldsymbol{p}_1^i, \ldots, \boldsymbol{p}_k^i}$ and $\boldsymbol{p}_{k+1\cdots 2k}^i$ denotes $\underline{\boldsymbol{p}_{k+1}^i, \ldots, \boldsymbol{p}_{2k}^i}$. Unlike contextualized prompts in Section 3.1, we expect inverse prompts to reflect better the relationship between $\mathcal{Y}$ and $\mathcal{X}$, which is dependent on the task rather than the input. Thus, the inverse prompts are static and continuous in our approach.

Generation with Inverse Prompting. With the two techniques mentioned above in the generation process, we utilize a modified beam search algorithm shown in Algorithm 1 to generate the sequence $\mathcal{Y}$ with the highest combined probability:

$$\mathcal{Y} = \underset{\mathcal{Y}}{\text{argmax}} \log \Pr(\mathcal{Y}|\mathcal{X}, \mathcal{P}^c) + \lambda \log \Pr(\mathcal{X}|\mathcal{Y}, \mathcal{P}^i), \tag{8}$$

where λ is a hyper-parameter to balance these two probabilities. We set λ to 4.0 with the best balance of performance.

In contrast to contextualized prompts that enrich the input information, continuous inverse prompting makes the generation process more controllable. Even for latter generated sentences, it can still enforce them to adhere to the input topic.

3.3 Discussion and Learning

In this part, we present the model discussion and optimization.

Discussion and Comparison. We use contextualized prompts (Eq. 6) to elicit useful knowledge from PLMs for different inputs. As a comparison, previous continuous prompting methods (Li and Liang, 2021; Lester et al., 2021) adopt static prompts, which are irrelevant to the input. Besides, we propose continuous inverse prompting (Eq. 7) to enforce the relevance of long output text by considering a generation process from output to input. Different from the original inverse prompting, our inverse prompting is based on continuous prompts, which can be optimized during fine-tuning.

Considering that our method involves another PLM and more parameters, we propose a lightweight context-tuning approach. Following Ben Zaken et al. (2022), we only fine-tune the bias term of each parameter, resulting in fine-tuning only 0.12% of the parameters of complete models. In the meanwhile, prefix-tuning (Li and Liang, 2021) and prompt tuning (Lester et al., 2021) freeze the PLM and only fine-tune the parameters of prompts. Prefix-tuning fine-tunes prompts in each layer and tunes 16.4% of the BART parameters, while prompt tuning only fine-tunes the prompt concatenated to the input, resulting in fine-tuning 0.05% of the BART parameters.

Optimization. We use the base version of BERT as our prompt generator. The number of prompt vectors k is set to 150. We utilize the base version of BART for text generation. The hyper-parameter λ in Eq. 8 is set to 4.0. There are two sets of trainable parameters in contextualized prompts and continuous inverse prompting, denoted by $\Theta^{(c)}$ and $\Theta^{(i)}$, respectively. First, we optimize $\Theta^{(c)}$, including BERT and BART, according to Eq. 6. Meanwhile, we optimize $\Theta^{(i)}$ according to the inverse generation loss using Eq. 7. During inference, we combine them and select sentences that are both informative and relevant to the input text based on Algorithm 1 and Eq. 8.

Dataset	#Train	#Valid	#Test	#Input	#Output
WP	53,516	4,000	2,000	25.48	150.64
ROC	176,688	9,816	4,909	9.02	40.72
CMV	42,462	6,480	7,562	17.89	104.10
WIKIP	69,288	8,661	8,662	3.38	194.72

Table 2: Statistics of our datasets after preprocessing. #Train, #Valid and, #Test denote the number of examples in training, valid, and test datasets, respectively. #Input and #Output denote the average number of tokens in the input and output text.

4 Experiment

In this section, we first set up the experiments and then report the results and analysis.

4.1 Experimental Setup

4.1.1 Construction of the Datasets

To measure the performance of our proposed context-tuning, we evaluate it on four open-ended text generation tasks: WRITINGPROMPTS (WP), ROCSTORIES (ROC), CHANGEMYVIEW (CMV), and WIKIPLOTS (WIKIP). Specifically, WP (Fan et al., 2018) consists of pairs of story premises and responses from the WritingPrompts forum. ROC (Mostafazadeh et al., 2016) is a dataset consisting of five-sentence commonsense stories. Here, we use the first sentence as the input to generate the following four sentences. For WP and ROC, we utilize the version provided by Guan et al. (2021) for a fair comparison. CMV (Hua and Wang, 2020) contains pairs of post statements on a controversial issue, which are collected from Reddit. WIKIP[1] is a collection of story plots from Wikipedia. We use the sub-header word to generate the full story.

Since some dataset outputs are significantly long, we discard examples where the text contains more than 512 tokens due to the length limitation of PLMs. We summarize the statistics of four datasets after preprocessing in Table 2.

4.1.2 Baseline Methods

We consider the following baselines as comparisons: GPT-2, BART, T5, HINT, prefix-tuning, and prompt tuning. Among these baselines, GPT-2 (Radford et al., 2019), BART (Lewis et al.,

[1] https://github.com/markriedl/WikiPlots

Models	WRITINGPROMPTS				ROCSTORIES				#Para
	BLUE-1	BLUE-2	Dist-1	Dist-4	BLUE-1	BLUE-2	Dist-1	Dist-4	
	Full fine-tuning								
GPT-2	24.94	9.03	1.40	35.38	31.45	14.26	2.21	58.63	1.2×10^8
T5	20.76	7.41	1.25	27.77	31.31	14.23	2.22	54.31	2.2×10^8
BART	28.42	11.31	2.11	62.05	32.95	15.35	2.70	68.88	1.4×10^8
HINT	22.40	8.40	–	31.30	33.40	15.40	–	69.30	1.4×10^8
Context-Tuning	**29.88**	**11.85**	**2.49**	**67.78**	**34.65**	**16.60**	**3.16**	**75.53**	2.5×10^8
	Lightweight fine-tuning								
Prompt Tuning	16.26	5.18	**3.71**	**69.68**	27.27	10.49	2.44	62.12	7.6×10^4
Prefix-Tuning	28.39	10.76	1.72	58.34	30.62	13.51	**2.51**	67.19	2.3×10^7
Context-Tuning	**29.45**	**10.90**	1.78	62.89	**32.24**	**14.30**	2.49	**68.92**	3.0×10^5
	CHANGEMYVIEW				WIKIPLOTS				
	BLUE-1	BLUE-2	Dist-1	Dist-4	BLUE-1	BLUE-2	Dist-1	Dist-4	
	Full fine-tuning								
GPT-2	23.39	8.32	0.75	37.18	23.74	9.33	0.90	38.39	1.2×10^8
T5	20.89	7.79	1.01	42.48	14.83	6.09	1.33	39.25	2.2×10^8
BART	25.69	9.77	**1.11**	**61.21**	27.12	11.54	1.82	49.54	1.4×10^8
Context-Tuning	**26.11**	**10.00**	0.99	57.38	**27.80**	**11.82**	**2.05**	**51.96**	2.5×10^8
	Lightweight fine-tuning								
Prompt Tuning	22.54	8.11	**1.43**	**64.60**	18.64	6.98	2.78	58.53	7.6×10^4
Prefix-Tuning	25.72	9.84	0.96	54.66	**27.30**	**11.60**	1.95	51.05	2.3×10^7
Context-Tuning	**28.83**	**10.96**	0.97	57.79	26.93	11.53	**2.48**	**59.34**	3.0×10^5

Table 3: Performance comparison of different methods for open-ended text generation tasks. Dist-n is short for Distinct-n. Bold and underlined fonts denote the best and second-best methods (the same below). #Para denotes the number of fine-tuned parameters in each method. The results of HINT are from its original paper (Guan et al., 2021). "–" means HINT does not compute the corresponding result.

2020), and T5 (Raffel et al., 2020) are three prevalent PLMs for natural language generation; HINT (Guan et al., 2021) is a strong baseline model specially designed for generating long and coherent texts; prefix-tuning (Li and Liang, 2021) and prompt tuning (Lester et al., 2021) are the recently proposed lightweight models using continuous prompts for generation tasks. We utilize the base version for all PLMs for a fair comparison.

4.1.3 Implementation Details

For fine-tuning settings, we consider two strategies: full fine-tuning and lightweight fine-tuning, to compare our methods with different baselines. In the lightweight fine-tuning settings, our context-tuning only tunes the bias term of each parameter.

In all experiments, we utilize the Adam optimizer and set $\beta_1 = 0.9$, $\beta_2 = 0.999$, $\epsilon = 1 \times 10^{-8}$. We train our model for 20 epochs and utilize the model with the best performance on the validation set for generation. During inference, we apply the nucleus sampling with $p = 0.9$ and temperature of 0.7. We train our model using NVIDIA A100 GPUs on Ubuntu 18.04 and employ the NLP open-source library Transformers (Wolf et al., 2020) and text generation library TextBox (Li et al., 2021).

4.1.4 Evaluation Metrics

To evaluate the performance of different methods of natural language generation, we adopt two automatic evaluation metrics, including BLEU (Papineni et al., 2002) and Distinct (Li et al., 2016). Specifically, BLEU evaluates the quality of generated and real text, while Distinct measures the diversity of generated texts.

4.2 Performance Comparison

We present the results of different methods on generation tasks in Table 3.

First, we can see that BART performs best compared to other PLMs on these generation tasks. Pre-trained on the large-scale corpus, PLMs can better understand natural language and fluently express human language. We consider the better perfor-

Models	B-1	B-2	D-1	D-4
Context-Tuning	29.88	11.85	2.49	67.78
w/o Continuous w **Manual**				
- human-written prompt$_1$	27.96	10.93	1.92	59.97
- human-written prompt$_2$	29.14	11.56	2.09	61.69
w/o BERT w **RoBERTa**	27.31	10.73	1.56	57.51
w/o **Semantic Mapping**	29.19	11.33	1.72	58.57
w/o **Inverse Prompting**	29.31	11.45	2.19	64.09

Table 4: Ablation analysis on WRITINGPROMPTS dataset.

Models	TT (%)	Flu.	Info.	Rel.	Coh.
GPT-2	81.20	3.90	3.27	3.77	3.50
T5	61.48	3.58	3.02	3.64	3.25
BART	77.17	3.82	3.27	3.74	3.59
Context-Tuning	**82.83**	**4.12**	**3.47**	**3.94**	**3.85**
Gold	94.00	4.26	3.90	4.33	4.01

Table 5: Turing test (TT) and human evaluation on WRITINGPROMPTS. "Gold" indicates the ground-truth texts. Flu., Info., Rel., and Coh. denote fluency, informativeness, relevance, and coherence, respectively.

mance of BART is due to the encoder-decoder architecture and the DAE pretraining task. That is the major reason we adopt BART as our base generation model.

Second, the recently proposed continuous prompting methods, prefix-tuning, and prompt tuning do not achieve ideal performance in these tasks. This finding shows that natural language generation tasks are more challenging than NLU tasks. Only fine-tuning a few parameters cannot outperform full fine-tuning.

Finally, our model outperforms all the baselines (including the strong baseline HINT) over four tasks under both full and lightweight tuning settings. The reason is that our context-tuning utilizes *contextualized prompts*, which can serve as queries to elicit input-relevant knowledge from PLMs. Under the lightweight fine-tuning settings, our context-tuning has superior results to prefix-tuning, only with 1.3% of the parameters of prefix-tuning and 0.2% of the parameters of BART. Some of the performance under lightweight settings can even outperform the full tuning, which may be a solution to catastrophic forgetting (Wang et al., 2021). And a major reason is that prefix-tuning and prompt tuning adopt static prompts, which are task-specific and unrelated to the context information.

4.3 Ablation Analysis

In this part, we construct ablation experiments to test the effectiveness of our proposed context-tuning. In contrast to previous prompt-based studies, our context-tuning has made several improvements. First, compared with manual prompts, we propose a continuous prompting approach to fine-tuning PLMs. Second, we adopt BERT as the prompt generator to derive the contextualized prompt vectors with semantic mapping. Finally, we utilize inverse prompting to enhance the relevance of the generated texts further. Here, we would like to examine how each factor contributes to the final performance. To see this, we prepare several variants for a comparison:

• *w/o Continuous w Manual*: the variant removes the continuous prompts but utilizes two kinds of human-written prompts, *i.e.*, prompt$_1$: "`Title: $Input Story:`" and prompt$_2$: "`Given the title $Input, please write the following story:`".

• *w/o BERT w RoBERTa*: the variant replaces BERT with RoBERTa (Liu et al., 2019) to form the prompt generator.

• *w/o Semantic Mapping*: the variant does not align to word embeddings and directly utilizes the top-layer representations of "`[MASK]`" tokens in the prompt generator.

• *w/o Inverse Prompting*: the variant removes inverse prompting (Eq. 8) from our proposed context-tuning.

From Table 4, we can see that variants replacing continuous prompts with manual prompts are worse than the model with continuous prompts. The performance of manual prompts is sensitive to different instructions and does not always lead to gains. This verifies the effectiveness of utilizing continuous prompts rather than discrete ones for text generation tasks. The variants replacing the BERT-based prompt generator with RoBERTa are worse than the full model. We further observe a slight performance drop when our method removes the semantic mapping and inverse prompting. This implies that the proposed semantic mapping and continuous inverse prompting approaches can enforce the informativeness relevance of output text.

4.4 Model Sensitivity

In this part, we construct sensitivity analyses *w.r.t.* the number k of prompt vectors on the WRITINGPROMPTS dataset.

In contextualized prompt learning, the number

#Prompt	B-1	B-2	D-1	D-4
$k = 50$	27.82	11.00	2.28	63.77
$k = 100$	28.55	11.12	2.17	63.69
$k = 150$	29.31	11.45	2.19	64.09
$k = 200$	28.48	11.22	1.97	61.09

Table 6: Performance tuning on WRITINGPROMPTS dataset. We do not utilize continuous inverse prompting methods here.

of prompt vectors is a key factor that influences the performance of our model. A longer sequence of prompt vectors means more trainable parameters and, therefore, more expressive power. Here, we will examine how it affects the final performance of our context-tuning. Given the statistics in Table 2, we vary the number of prompt vectors in the set $\{50, 100, 150, 200\}$. We separately train our model with different numbers of prompt vectors and do not utilize continuous inverse prompting methods for convenience. As shown in Table 6, the performance of our model gradually improves as the number of prompt vectors increases up to a threshold, and then a performance drop occurs. More importantly, our model achieves the best performance with 150 prompt vectors over baselines.

4.5 Human Evaluation

Besides automatic evaluation, we further conduct a human evaluation for testing the effectiveness of our approach. We randomly select 500 input texts from the test set of the WRITINGPROMPTS dataset. We collect the stories generated by GPT-2, BART, T5, and context-tuning, then shuffle them for human evaluation. Following Zou et al. (2021), we invite ten human judges to assign scores to a generated text concerning four factors of quality, namely *informativeness* (how much it provides valuable and meaningful information), *relevance* (how relevant it is according to the input contexts), *coherence* evaluates (how coherent both intra and inter sentences are) and *fluency* (how likely a human produces the generated text).

We adopt a 5-point Likert scale as the scoring mechanism, in which 5-point means "very satisfying", and 1-point means "very terrible". Furthermore, inspired by Zou et al. (2021), we design a Turing test where a human judge is asked to distinguish whether a human produces the given text. The detailed evaluation guidelines and examples are listed in Figure 2, Figure 3, Figure 4, and Figure 5 in the Appendix.

We present the human evaluation results in Table 5. It can be seen that our model is better than the three baselines with a large margin. The major reason is that we utilize the contextualized prompts derived from the input text. Our contextualized prompts can extract knowledge from PLMs and serve as additional input information to be fed into PLMs, which improves the informativeness of the generated text. Moreover, the proposed continuous inverse prompting method enhances the relevance of the generated text to the input.

4.6 Qualitative Analysis

In this part, we present an intuitive analysis of why our model works well.

Table 7 presents an example story from the WRITINGPROMPTS dataset and the generated story by our model and two baselines, *i.e.*, GPT-2 and BART. As we can see, there is limited information in the input premise, besides several keywords such as nature, documentary, and Pokémon.

First, we can see that the story generated by our context-tuning is highly relevant to the input text and conveys richer semantic information. A primary reason might be that our contextualized prompts can elicit input-relevant knowledge from PLMs for generating more informative text. Although PLMs perform well in generating fluent text, we can see that GPT-2 and BART are still prone to generating unmeaningful and irrelevant content, such as "I love you" and "language".

Furthermore, to probe whether our contextualized prompts contain input-relevant knowledge, we find close actual words for a better explanation. We use $2k$ contextualized prompts in total, and for each continuous prompt, we recall the word in the BERT vocabulary with the closest cosine distance to it. Finally, we select some words from $2k$ recalled words and showcase them grouped by four aspects in the row *closer words to contextualized prompts* of Table 7. As we can see, most recalled keywords are included in the story generated by our context-tuning. It shows that our contextualized prompts can better capture input-relevant knowledge. For example, the keywords "forest", "woods", and "animal" are closely related to the aspect of *nature*.

Title	Write a Nature Documentary about Pokémon .
GPT-2	<newline> it's been a long time since i've seen one of these things. <newline> i don't know how long it has been, but i know that it will be long. i have seen it before, and i will see it again. it is the only thing i can think of that makes sense to me, because it makes me feel like i am part of something greater than myself. and that is why i want to know more about it. because if i could, i would know what it was like to be human. to have a sense of purpose and purposelessness, to feel that there is a purpose in everything that you do, even if you are just a shell of what you once were. but that does not make it any less of a feeling. that feeling that i feel when i look into the eyes of another human, when they look at me and say, " i love you. "
BART	the pokémon are the most powerful of all theokémon in the world. <newline> they are the best of the best, but they are also the most dangerous. they have the ability to move faster than any otherokémon on the planet, and they are able to communicate with each other in a way that is impossible for humans to understand. however, they are not able to fight each other, because they are unable to communicate in a language that is not their own. they can only communicate through their own language, which is very different from any other language that humans can understand. they are very difficult to understand, and can not communicate in any language that can be understood by humans. they also have a very limited ability to speak, so they can not understand the language that they are speaking. in fact, they have no ability to communicate at all, because their language is very similar to that of a human.
Closer words to contextualized prompts: forest trees marching winds food historically story movie pet pikachu animation whilst critical asking write	
Context-Tuning	" pokémon! " <newline> " what are you talking about? " the man asked. " i'm talking about a pokéball. it's called pikachu, and it lives in the forest. it lives on the edge of the forest, where it can eat anything it wants to eat. it is the only thing that can keep it alive. it can live in the woods, but it can also be eaten by any other creature. it has the ability to eat anything that it wants, and can even eat any other animal it wants. " he said. the man looked at the man, and said, " i don't know what you're talking about, but i do know that it can be eaten. " the two men looked at each other, and the man spoke, " you've got to be kidding me. "

Table 7: The generated examples of the given title from the WRITINGPROMPTS dataset. Marks in closer words to contextualized prompts and texts generated by context-tuning refer to four chosen aspects, *i.e.*, nature, documentary, Pokémon, and the story generation task.

5 Conclusion

This paper presents a novel continuous prompting approach, *i.e.*, context-tuning, to fine-tuning PLMs for natural language generation. The core idea is to inject input-related context information into continuous prompts, called contextualized prompts, to enhance the informativeness of generation. The contextualized prompts can elicit input-relevant knowledge from PLMs to enrich the input text. Furthermore, to enhance the relevance of the generated text to the inputs, we adopt a continuous inverse prompting method to refine the forward generation process by modeling an inverse generation process from output to input. We also propose a lightweight method for efficient training. Extensive experiments on four generation tasks have demonstrated the effectiveness of our model in fine-tuning PLMs for text generation tasks.

In future work, we will consider integrating more types of context information (*e.g.*, sentiment) to derive more expressive prompts and investigate how our model could be applied to other tasks.

Acknowledgement

This work was partially supported by Beijing Natural Science Foundation under Grant No. 4222027, Beijing Outstanding Young Scientist Program under Grant No. BJJWZYJH012019100020098 and Beijing Academy of Artificial Intelligence (BAAI). Xin Zhao is the corresponding author.

References

Elad Ben Zaken, Yoav Goldberg, and Shauli Ravfogel. 2022. BitFit: Simple parameter-efficient fine-tuning for transformer-based masked language-models. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 1–9, Dublin, Ireland. Association for Computational Linguistics.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, Minneapolis, Minnesota. Association for Computational Linguistics.

Zi-Yi Dou, Pengfei Liu, Hiroaki Hayashi, Zhengbao Jiang, and Graham Neubig. 2021. GSum: A general framework for guided neural abstractive summarization. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 4830–4842, Online. Association for Computational Linguistics.

Angela Fan, Mike Lewis, and Yann Dauphin. 2018. Hierarchical neural story generation. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 889–898, Melbourne, Australia. Association for Computational Linguistics.

Tianyu Gao, Adam Fisch, and Danqi Chen. 2021. Making pre-trained language models better few-shot learners. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*,

pages 3816–3830, Online. Association for Computational Linguistics.

Jian Guan, Xiaoxi Mao, Changjie Fan, Zitao Liu, Wenbiao Ding, and Minlie Huang. 2021. Long text generation by modeling sentence-level and discourse-level coherence. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 6379–6393, Online. Association for Computational Linguistics.

Zhiting Hu, Zichao Yang, Xiaodan Liang, Ruslan Salakhutdinov, and Eric P. Xing. 2017. Toward controlled generation of text. In *Proceedings of the 34th International Conference on Machine Learning*, volume 70 of *Proceedings of Machine Learning Research*, pages 1587–1596. PMLR.

Xinyu Hua and Lu Wang. 2020. PAIR: Planning and iterative refinement in pre-trained transformers for long text generation. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 781–793, Online. Association for Computational Linguistics.

Yuta Kikuchi, Graham Neubig, Ryohei Sasano, Hiroya Takamura, and Manabu Okumura. 2016. Controlling output length in neural encoder-decoders. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 1328–1338, Austin, Texas. Association for Computational Linguistics.

Brian Lester, Rami Al-Rfou, and Noah Constant. 2021. The power of scale for parameter-efficient prompt tuning. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 3045–3059, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Veselin Stoyanov, and Luke Zettlemoyer. 2020. BART: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7871–7880, Online. Association for Computational Linguistics.

Jiwei Li, Michel Galley, Chris Brockett, Jianfeng Gao, and Bill Dolan. 2016. A diversity-promoting objective function for neural conversation models. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 110–119, San Diego, California. Association for Computational Linguistics.

Junyi Li, Tianyi Tang, Gaole He, Jinhao Jiang, Xiaoxuan Hu, Puzhao Xie, Zhipeng Chen, Zhuohao Yu, Wayne Xin Zhao, and Ji-Rong Wen. 2021. TextBox: A unified, modularized, and extensible framework for text generation. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing: System Demonstrations*, pages 30–39, Online. Association for Computational Linguistics.

Junyi Li, Tianyi Tang, Wayne Xin Zhao, Jian-Yun Nie, and Ji-Rong Wen. 2022. A survey of pretrained language models based text generation. *arXiv preprint arXiv:2201.05273*.

Junyi Li, Wayne Xin Zhao, Ji-Rong Wen, and Yang Song. 2019. Generating long and informative reviews with aspect-aware coarse-to-fine decoding. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 1969–1979, Florence, Italy. Association for Computational Linguistics.

Xiang Lisa Li and Percy Liang. 2021. Prefix-tuning: Optimizing continuous prompts for generation. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 4582–4597, Online. Association for Computational Linguistics.

Pengfei Liu, Weizhe Yuan, Jinlan Fu, Zhengbao Jiang, Hiroaki Hayashi, and Graham Neubig. 2021. Pre-train, prompt, and predict: A systematic survey of prompting methods in natural language processing. *arXiv preprint arXiv:2107.13586*.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Roberta: A robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692*.

Nasrin Mostafazadeh, Nathanael Chambers, Xiaodong He, Devi Parikh, Dhruv Batra, Lucy Vanderwende, Pushmeet Kohli, and James Allen. 2016. A corpus and cloze evaluation for deeper understanding of commonsense stories. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 839–849, San Diego, California. Association for Computational Linguistics.

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. Bleu: a method for automatic evaluation of machine translation. In *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics*, pages 311–318, Philadelphia, Pennsylvania, USA. Association for Computational Linguistics.

Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, Ilya Sutskever, et al. 2019. Language models are unsupervised multitask learners. *OpenAI blog*, 1(8):9.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J. Liu. 2020. Exploring the limits of transfer learning with a unified text-to-text transformer. *Journal of Machine Learning Research*, 21(140):1–67.

Leonardo F. R. Ribeiro, Martin Schmitt, Hinrich Schütze, and Iryna Gurevych. 2021. Investigating pretrained language models for graph-to-text generation. In *Proceedings of the 3rd Workshop on Natural Language Processing for Conversational AI*, pages 211–227, Online. Association for Computational Linguistics.

Timo Schick and Hinrich Schütze. 2021a. Exploiting cloze-questions for few-shot text classification and natural language inference. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*, pages 255–269, Online. Association for Computational Linguistics.

Timo Schick and Hinrich Schütze. 2021b. Few-shot text generation with natural language instructions. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 390–402, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Taylor Shin, Yasaman Razeghi, Robert L. Logan IV, Eric Wallace, and Sameer Singh. 2020. AutoPrompt: Eliciting Knowledge from Language Models with Automatically Generated Prompts. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 4222–4235, Online. Association for Computational Linguistics.

Ruize Wang, Duyu Tang, Nan Duan, Zhongyu Wei, Xuanjing Huang, Jianshu Ji, Guihong Cao, Daxin Jiang, and Ming Zhou. 2021. K-Adapter: Infusing Knowledge into Pre-Trained Models with Adapters. In *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*, pages 1405–1418, Online. Association for Computational Linguistics.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Remi Louf, Morgan Funtowicz, Joe Davison, Sam Shleifer, Patrick von Platen, Clara Ma, Yacine Jernite, Julien Plu, Canwen Xu, Teven Le Scao, Sylvain Gugger, Mariama Drame, Quentin Lhoest, and Alexander Rush. 2020. Transformers: State-of-the-art natural language processing. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 38–45, Online. Association for Computational Linguistics.

Yizhe Zhang, Siqi Sun, Michel Galley, Yen-Chun Chen, Chris Brockett, Xiang Gao, Jianfeng Gao, Jingjing Liu, and Bill Dolan. 2020. DIALOGPT : Large-scale generative pre-training for conversational response generation. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics: System Demonstrations*, pages 270–278, Online. Association for Computational Linguistics.

Xu Zou, Da Yin, Qingyang Zhong, Hongxia Yang, Zhilin Yang, and Jie Tang. 2021. Controllable generation from pre-trained language models via inverse prompting. In *Proceedings of the 27th ACM SIGKDD Conference on Knowledge Discovery & Data Mining*, KDD '21, page 2450–2460. Association for Computing Machinery.

Thank you for taking time out of your busy schedule to participate in our scientific research evaluation! Our research work is to let the machine generate corresponding story, for a given title, and hope that it is as close as possible to what humans write. Hence, we need to evaluate whether it meets the standards that people think.
In this task, you will see a title, idea or introduction such as:
You have a very shitty type of precognition .
Then you'll see a corresponding story or comment, either written by a human or a machine, but you don't know which one was written, and you'll have to rate it based on the following standard:

- **Turing Test:** whether the text was written by a human;
- **Fluency:** whether the text has good form, logical reading and smooth sentences;
- **Informative:** whether the text contains meaningful content and will not be boring to read;
- **Relevance:** whether the text is highly relevant to the input context;
- **Coherence:** whether the logic is coherent and not contradictory;

For Turing Test, just judge the text by instinct. For the other four factors, we adopt a 5-point likert scale as the scoring mechanism, in which 5-point means "very satisfying", and 1-point means "very terrible".

Figure 2: Evaluation guidelines for WRITINGPROMPTS dataset.

Fluency: whether the text has good form, logical reading and smooth sentences.

- **1 point example:** I was a exterminator on earth. I killed every insect and animal I killed in my life. If I win, I go to heaven, lose, and go to hell. I was an exterminator in hell. I was the exterminator of the world. I exterminated every insect, animal, and animal that I killed. I didn't have to kill them all. I just had to do it. I had to kill all of them. I couldn't let them get away with it. So I killed them all, and I did it all over again. And again and again. I don't know how long I've been doing this, but it's been a long time. **(containing some grammatical errors and repetitive contents)**

- **1 point example:** I followed you into the dark . We carved our names into the bark . " Forever and always Through short nights and through long days . " But that tree withered so long Its voice died in sweet silent song . Dead and gone We left and lost our life bond . I always remember your words Of comfort and joy like the birds . Sweet and sad ; It was all I never had . For I carved both names on that tree No kind woman would be with me . No more hope ; You and me , the end of a rope You were not ever real and I know that now . " Always to love you " , I end with that vow . **(there is no continuity between the words of the sentence, and the content is intermittent)**

- **3 point example:** I've been trying to kill my master for years. I've tried to kill him for years, but he's always been there for me. He's the only one who knows what I'm going to do, and I don't care. I 'll kill him if I have to. But I can't do it anymore. I haven't been able to do it for years now. I can not do it any more. I just want to go back to my master. I want to be with him again. But he won't let me go back. I know it's not fair, but I just need to get back to him. **(each sentence is grammatically correct and fluent, but contains certain repetitions and discontinuities in semantics)**

- **3 point example:** It's been a long time since I've seen her. She's always been there for me. I'm not sure how long I have been here, but I know she's here. I know I 'll never see her again. I don't know if she 'll ever see me again. But I know it's time. I can feel it in my bones, in my skin, in the bones of my bones. I can't help but think of her. I remember her when I first met her, when I was young. She was so beautiful, so full of life. I couldn't wait to meet her again, to see her smile again. **(sentences are fluent, but similar words are used repeatedly in the sentence, resulting in ambiguous meaning and confusing)**

- **5 point example:** Long ago his heart had warmed , three thousand years - long enough to mourn , the deeds of past and of damnation , stripped of humanity and of his station . He resided in the pits of hell the oldest friend of satan , waiting as the centuries pass watching hells inflation , resting on brimstone as passing devils chatter and laugh , who is this old man and what sin has made him . a curious young man with a glint in his eye asks his sentence , and with creaks and groans the old man rose for the first time in ages , he look at the spirit and with a heavy sigh he came out with , I 'm god and I made this .

- **5 point example:** Tell us your faults ? Really ? This was the question - the shibboleth - that unlocked the cosmos ? The Masters could have picked a scientist to answer but they feared she might mask ignorance . They could have picked from our global leaders bit they feared that they would mask deceit . They could have picked a holy man but feared he would mask violence , oppression , hate , intolerance ... the list of disqualifying sins was almost too long to enumerate . So they picked Josh Thornton , a 45 year old MBA in human resources . " Our greatest weakness ? Well , I think we work a little too hard and , as a race , we might be a bit of a perfectionist . "

Figure 3: Evaluation examples for *fluency* factor.

Informative: whether the text contains meaningful content and will not be boring to read.

- **1 point example:** I've been doing this for years. I've been doing this for years. I've been doing this for years. I've been doing this for years. I've been doing this for years. I've been doing this for years. I've been doing this for years. I've been doing this for years. I've been doing this for years. I've been doing this for years. I've been doing this for years. I've been doing this for years. I've been doing this for years. I've been doing this for years. I've been doing this for years. I've been doing this for years. I've been doing this for years. **(sentence repetition)**

- **1 point example:** " I'm the First ****bender. " " What's that? " I asked. " I don't know, " he replied. " But you're the first ****bender, right? " " Yeah, " I said. " You're not the first. " " Well, I guess that's true. " He paused. " What do you mean? " He looked at me. " That's right. " I looked at him. " It's not like you've ever been the first... " He trailed off. " No, it's just... " I trailed off, " I've never been the First... " **(conversation sentences are very short, and they are all safe replies)**

- **3 point example:** " What the hell is this? " " I don't know, " I said. " I'm not sure what to do with this. I've been doing this for a long time, and it's not like I 'll be able to do anything about it. " He looked at me with a confused look on his face. " What the fuck? " I asked. " You're not going to do this again. " I replied. " It's just a matter of time, " he said. I looked at the frying pan again. " Don't worry, it 'll work. It 'll do. " " What's the problem? " He asked. I nodded. " Well, I guess I 'd better get out of here. " The frying pan buzzed at me and text appeared reading " level 18 cooking required to use object. " **(containing rich dialogue, but the content of the dialogue is meaningless)**

- **3 point example:** The stars fell like rain, and we danced. The moon was bright, and the stars danced. The stars were bright, but the stars were not bright. They were bright and the sky was blue. We danced, but we didn't know what to do. We danced and we sang and we laughed and we talked and we cried and we screamed and we played and we giggled and we thought we were going to die, but it wasn't. It was just the stars falling like rain. And we danced, and it was beautiful. **(the article has a certain content at the beginning, but it is more verbose at the end)**

- **5 point example:** She is the woman you have loved since the day you met her . Everyday she has a smile on her face beautiful as ever . You love her but are afraid of what she will say when you tell her . It was raining and you ran for shelter , a small roof at the bus stop . Tired and panting you barely notice her sitting beside you , she calls your name . You jump a little bit and become nervous when you recognize her . You stare at each other not knowing what to do or say , and then she kisses you . An alarm sounds , you wake up in your room all alone , another dream **(a whole story)**

- **5 point example:** I feel like it 's worth pointing out that a lot of these are kind of situational . I think all of these rules are good to follow if what you 're writing is something you 're trying to submit to someone formally or something , but a lot of them are not important in casual writing (such as someone 's dialogue or something like that) . For example , " literally " has been used for hyperbole for a very long time – I know I 've heard that Nathaniel Hawthorne did it , and I hardly think he was the first . It pisses a lot of people off but it 's not like it 's a new phenomenon and it 's not like it 's a corruption of the language . Things about spelling and homonyms and stuff should probably be followed just about all the time though . **(using concrete examples)**

Figure 4: Evaluation examples for *informative* factor.

Coherence evaluates how content is coherent considering both intra- and inter-sentence correlation.

- **1 point example:** i don't know if it's a good thing or a bad thing, but i've found that if i'm going to write a story, i need to be able to get it out of my head. i think that's the most important thing. if i want to write something, i have to know what i want it to be. i have a lot of ideas, but they're just not good enough. i'll try to find a way to get them out, but if i can't find the right words to write, i will probably have to go back and re - write it. **(no semantic connection between sentences)**

- **1 point example:** " I'm sorry, " I said to myself. " I don't know what to do. " I replied. " I just want to see you again. " I looked at my reflection, and it was the same. I couldn't tell if it was a reflection or a reflection, but I knew that it was my reflection. I looked back at the mirror, and I saw that I was the one who was going to be my future soulmate. I smiled, and said, " I love you, " and walked away.

- **3 point example:** I have a very shitty type of precognition. I don't know what it is, but it's something I have to do. I'm not sure why I do it, but I do. I've been doing it for years now, and I haven't been able to figure out why. It's not like I have any control over it. I can't control what I do, or what I say, or how I act. I can only control the way I act, how I react, and how I feel. I have no control over my actions, and no control of my emotions. I just have to control my emotions, and that's all I can do. **(after careful consideration, many logical contradictions were found)**

- **3 point example:** I 'd like to take a moment to appreciate Sir Terry Pratchett. I've read a lot of his work, and I'm not sure if it's because of his writing style or because of the way he wrote it. I don't know if he's a good writer, or if he is a bad writer, but I do know that he is one of the best writers I have ever read. I think that's why I love him so much. I also think that he has a great sense of humor, and that he doesn't have a bad sense of humour. **(some repeated information, but other content is ok)**

- **5 point example:** You eagerly await your pizza to come because you ordered from this new Italian Pizza owed by two brother , you remember that one of their names are Mario but you forgot the other . The Pizza finally arrives a bit late from this tall guy dressed in green . You pay him take , take the pizza but forget to tip . When you start eating you get a bit dizzy so you lay down and fall asleep quite quickly . You wake up in a in a place covered in mushrooms with a little man dressed as a mushroom telling you that " You need to save the princess " . **(smooth connection between context)**

- **5 point example:** When 1st purge happened , no one thought people would attack each other . A desperate party know only as Al Queda broke the rules and decided that it would do what no one else would have done . Bomb Manhattan . That single move destroyed not only the Republicans and the Democrats , it also destroyed morale . Hundreds of fully armed fat Politicians fled to the streets , screaming out jibberish and shooting anyone they see . Millions lay dead as all parties Jump onto their jets towards Manhattan , preparing to be included in the giant Cesspit of a war know as the Purge . When the Morning came . There were no victors . Only that the red dawn came and claimed .

Figure 5: Evaluation examples for *coherence* factor.

PSP: Pre-trained Soft Prompts for Few-Shot Abstractive Summarization

Xiaochen Liu[†], **Yang Gao**[†]*, **Yu Bai**[†],
Jiawei Li[†], **Yinan Hu**[†], **Heyan Huang**[†] and **Boxing Chen**
[†]School of Computer Science and Technology,
Beijing Institute of Technology
{xcliu,gyang,yubai,jwli,ynhu,hhy63}@bit.edu.cn
chenboxing@gmail.com

Abstract

Few-shot abstractive summarization has become a challenging task in natural language generation. To support it, we developed a novel soft prompts architecture coupled with a prompt pre-training plus prompt fine-tuning paradigm, which is effective and tunes only extremely light parameters. To meet the structure of the generation models, the soft prompts comprise continuous input embeddings across an encoder and a decoder. Importantly, a new inner-prompt placed in the text is introduced to capture document-level information. The aim is to devote attention to understanding the document that better prompts the model to generate document-related content. In the training process, the prompt pre-training with self-supervised pseudo-data firstly teaches the model basic summarizing capability. Then, with few-shot examples, only the designed lightweight soft prompts are fine-tuned. Experimental results on the CNN/DailyMail and XSum datasets show that our method, with only 0.1% of the parameters, outperforms full-model tuning where all model parameters are tuned. It also surpasses Prompt Tuning by a large margin and delivers competitive results against Prefix-Tuning with 3% of the parameters.

1 Introduction

Given the high labor-costs of obtaining quality abstractive summaries, few-shot abstractive summarization is very demanding and highly challenging. A widely accepted paradigm for almost all NLP tasks is to fine-tune the entire set of parameters for a large pre-trained language model to suit the target task (Liu and Lapata, 2019; Liu et al., 2020).

However, the fine-tuning with few-shot examples usually leads to disappointing results, especially with generation tasks like abstractive summarization (Fabbri et al., 2020; Yu et al., 2021). The likely outcome is an overfit model. Further, for every specific task, a large number of pre-trained parameters need to be updated and stored, which is not efficient to use.

Pre-trained language models are few-shot learners, i.e., GPT-3 (Brown et al., 2020) that surprisingly perform generation tasks from a few examples without any further gradient updates. Although it lacks a rigorously theoretical proof, prompt learning inherits the few-shot property (Li and Liang, 2021; Schick and Schütze, 2020; Jin et al., 2021; Liu et al., 2021). Commonly, this type of learning is considered to retrieve relevant knowledge from frozen language models, only tuning continuous prompts to quickly adapt to new tasks with very few examples.

More recently, Prompt Tuning (Lester et al., 2021) has received much attention. With large frozen language models (say, >10 billion parameters), Prompt Tuning simply adds a tunable soft prompt to the input of the encoder, achieving results that are comparable to full-model tuning. Yet, our empirical results, in Section 2, demonstrate that Prompt Tuning for abstractive summarization yields simply abysmal performance. Prefix-Tuning (Li and Liang, 2021) extends the use of prompt learning in the natural language generation area. With this technique, continuous prompts are applied to every layer of the pre-trained model and even shows increase in few-shot generation tasks over fine-tuning. Yet the training process is not stable and updates are required that add to the memory and training costs.[1]

Given the shortcomings of these two methods, we have developed a soft prompts tuning method that is specifically designed for summarization. The structure is given in Figure 1. The method is capable of performing few-shot language generation task (i.e., abstractive summarization) with an efficient amount of training parameters. Prompt tokens are added before the decoder input tokens

*Corresponding author.

[1]See more related work in Section 5.

Proceedings of the 29th International Conference on Computational Linguistics, pages 6355–6368
October 12–17, 2022.

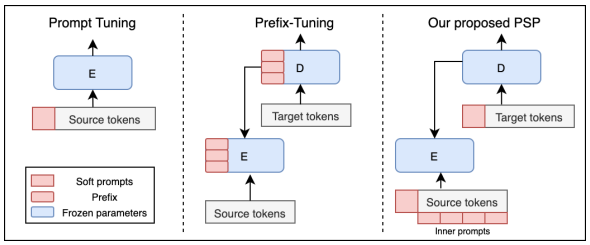

Figure 1: The comparison between PSP and previous methods. "E" and "D" represents the encoder and the decoder, respectively.

to guide the generation process toward the target summary. Moreover, we have designed three inner prompts – interval, sequential, and fixed-length – one of which is placed among the source input tokens. The aim is to capture the structure in the source document and aid in understanding its semantics, so as to better prompt the model to generate document-related content. Each kind of inner prompts focuses on different semantic units (e.g., phrases, sentences, and etc.), differentiating important units from non-informative ones. To bolster the summarization ability of the model and assist the prompts to understand the documents, prompt pre-training is performed before the tuning process, and leveraged by self-supervised pseudo data. As a last step, all the prompts are fine-tuned with few-shot training examples. Experiments conducted on two commonly used datasets - CNNDM (See et al., 2017) and XSum (Narayan et al., 2018) - demonstrate that our method outperforms full-model tuning under few-shot settings only with 0.1% of the parameters. It also surpasses naive Prompt Tuning by a large margin. Our model also yields a performance competitive to Prefix-Tuning with 3% of the trainable parameters. A detailed analysis shows that the designed prompt-pre-training phase and the inner prompts are effective for few-shot text summarization. Thus, the major contributions of this work include : 1) A novel soft prompt architecture for few-shot abstractive summarization. With the well-designed prompts in embedding layer, our model fulfills the task effectively and efficiently; 2) It is necessary to perform prompt pre-training strategy which benefits soft prompts model for few-shot summarization and shows excellent zero-shot capabilities; 3) Experiments that investigate the effect of different prompts by probing the attention weights. The results show our model is able to: extract knowledge from the encoder language model; understand the discourse in the document; and guide the decoder language model to generate fluent summaries.

2 Pilot Experiments

In a pilot study, we experimented with using Prompt Tuning under 300-shots settings to find reasonable clues as to how to design summary-prompts for the task. Our findings follow.

Consider an encoder-decoder language model $p_\theta(y|x)$ based on the Transformer architecture (Vaswani et al., 2017) (e.g., BART (Lewis et al., 2020)) and parameterized by θ. To conduct a few-shot summarization task, we have some few-shot training pairs of a document $X = \{x_1, x_2, \ldots, x_{|X|}\}$ and a corresponding summary $Y = \{y_1, y_2, \ldots, y_{|Y|}\}$. Specifically, we divided X into different subsets with **sentences**[2] as our unit, $X = \{x_1^1, \ldots x_j^i, \ldots, x_m^n\}$, where x_j^i denotes the j_{th} token in the i_{th} sentence.

First, original Prompt Tuning is applied by concatenating a series of prompt tokens P_{en}, parameterized by $\theta_{p_{en}}$, to the encoder input $X_{en} = \{e_1^1, \ldots, e_j^i, \ldots e_m^n\}$, where e represents the embedding of each token (the leftmost structure in Figure 1). The gradients are backpropagated through the prompts and the weights θ of language model are frozen (Lester et al., 2021). In this way, the model maximizes the likelihood of the output Y:

$$p_{\theta;\theta_{\texttt{pen}}}(Y|[P_{en}; X_{en}]) \quad (1)$$

The result of original Prompt Tuning is shown on the first line in Table 1, where we see it severely underperforms versus full-model tuning. In further experiments, we added a series of prompts P_{de} to the decoder inputs X_{de} following the generation $p_{\theta;\theta_{p_{de}}}(Y|X_{en}, P_{de})$. Here, we found the results to be even worse than the last.

Necessary Prompts for Generation For generation-based tasks, prompts in both the encoder and decoder are equivalently useful. Therefore, our model employs a combination of the two series of prompts mentioned above, and generates Y conditioning on X_{en}, P_{en} and P_{de}:

$$p_{\theta;\theta_{p_{en}};\theta_{p_{de}}}(Y|[P_{en}; X_{en}], P_{de}) \quad (2)$$

The result on the third line in Table 1 again verify our hypothesis. Prompts across the encoder and

[2]Note that, throughout this work, a "sentence" can be an arbitrary span of contiguous text (e.g., fixed length of 10 tokens), or an actual linguistic sentence.

Model	ROUGE-1	ROUGE-2	ROUGE-L
Prompt in encoder	32.87	11.92	21.73
Prompt in decoder	26.77	11.73	16.71
Prompt in en.&de.	36.37	14.41	**24.46**
Full-Model Tuning	**37.01**	**14.49**	23.91

Table 1: Results of BART-base on CNN/DailyMail Datasets. Best results are bold.

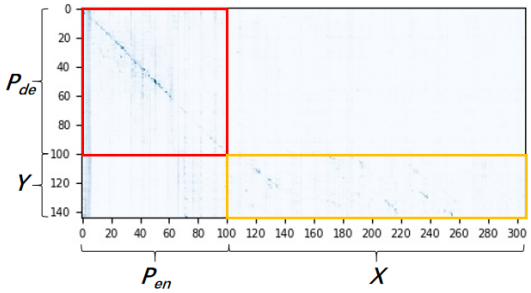

Figure 2: Visualization of the encoder-decoder attention weights. The x-axis are the encoder input, including prompts across the encoder P_{en} and the source document X. The y-axis are the decoder input, including prompts across the decoder P_{de} and the target summary Y. The area in the red box represents the attentions of P_{de} assigning to P_{en}. The area in the yellow box represents the attentions of Y assigning to X. Darker color shows the more highly related associations between tokens.

decoder even achieve comparable results with full-model tuning under few-shot settings. This verifies two things for us. First, prepending simple prompts to only the input embedding layer is effective and efficient for few-shot abstractive summarization. Second, prompts across the encoder and decoder are both necessary for generation tasks.

Lack of Attention on the Document We further explored the encoder-decoder attention to investigate the effect of the prompts and freezing the language model. From Figure 2, we find the generating output is mainly focused on the soft prompts to come with little attention given to the document itself. This outcome is detrimental to summarization that requires to understand the semantics and inner discourse structure of documents (Wang et al., 2019). Without the associations of target summaries and source documents, it is impossible to obtain high-quality summaries using current prompt architectures.

From Figure 2, we can observe that prompts in the encoder and the ones in decoder are consistently and directly associated with each other. We speculate that the mechanism is that encoder prompts

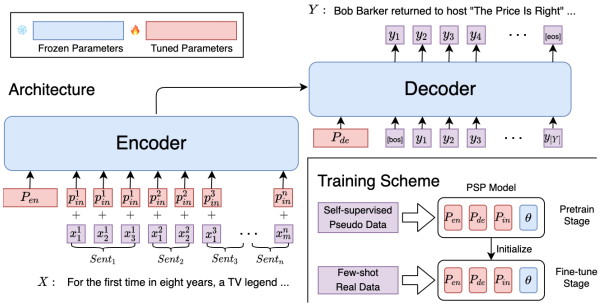

Figure 3: Architecture and training scheme of PSP. Squares in blue and red indicates frozen and tuned parameters, respectively.

retrieve relevant knowledge from the frozen encoder language model as a document representation, and decoder prompts copy the encoder's behaviour, guiding the decoder language model to generate text.

3 Method

In light of our findings about the current architectures, we developed a new architecture of pre-trained soft prompts, for few-shot abstractive summarization called PSP. The framework includes continuous prompts across the encoder and decoder inputs, as well as inner-prompts to capture the dependencies between documents and target summaries. To better understand a given document, we add a prompt pre-training process before few-shot tuning. It also brings a good initialization for the prompting. The overall architecture and training scheme are illustrated in Figure 3.

3.1 Encoder-Decoder Basic Prompts

As mentioned in Section 2, in the training phase of current architectures, P_{en} is responsible for extracting knowledge from the encoder's frozen language model as a document representation. Meanwhile, P_{de} mostly copies the behavior of P_{en} and guides the frozen decoder's language model to generate fluent text as a summary.

To strengthen the model's ability to understand a document, the dependencies and attentions given to the source document need to be embodied in the prompt architecture.

3.2 Inner-Prompts for Document Understanding

To achieve our goal, we propose the notion of adding inner-prompts within the source document,

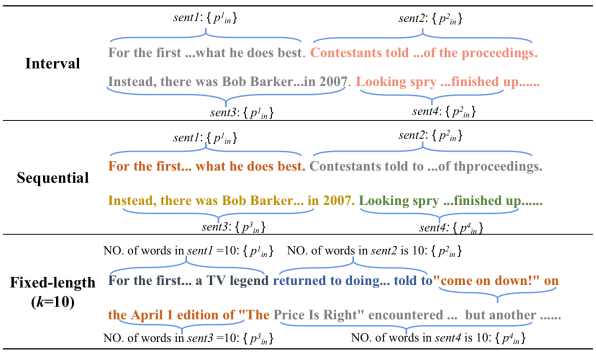

Figure 4: Different inner prompts for one example source document. Different colors indicate different inner prompt embeddings. "NO. of words" means the length of the text span.

denoted as $P_{in} = \{p^1_{in}, p^2_{in}, \ldots, p^n_{in}\}$ with the parameters $\theta_{P_{in}}$ to be updated. Each p^i_{in} corresponds to a single sentence. These inner-prompts are added to the corresponding token embedding, which gives rise to a new X'_{in}:

$$X'_{in} = \{e^1_1 + p^1_{in}, e^1_2 + p^1_{in}, \ldots, e^i_j + p^i_{in}, \ldots, e^n_m + p^n_{in}\} \quad (3)$$

We believe that by prompting different semantic units (e.g., sentences, phrases, etc.), more attention can be given to understanding the document's discourse. Furthermore, the inner-prompts help the model to quickly interpret the document by strengthening the associations between outputs and documents. What follows are three different strategies for incorporating the three different inner-prompts. Note that there is more discussion on this point in Section 4.2.

Interval Following Liu and Lapata (2019), the interval inner-prompts comprises two inner-prompt tokens are assigned to each sentence $sent_i$, depending on whether i is odd. Specifically,

$$P_{in} = \{p^1_{in}, p^2_{in}, p^1_{in}, \ldots, p^{(n-1)\bmod 2+1}_{in}\} \quad (4)$$

In this way, the model can identify important sentences to encode the document at sentence level.

Sequential To highlight the complex discourse structure of documents, sentence positions need to be considered. Therefore, different tokens are set in sentences by their sequences, formulated as:

$$P_{in} = \{p^1_{in}, p^2_{in}, \ldots, p^n_{in}\} \quad (5)$$

Fixed-length To discover more fine-grained semantic units, a text span with a fixed length k is manipulated into a new "sentence" and a corresponding sequential token is assigned to it. Further, prompts are assigned to the newly divided sentences $[sent_1, sent_2, ..., sent_n]$, as $\{p^1_{in}, p^2_{in}, \ldots, p^n_{in}\}$. Figure 4 illustrates some examples where the above strategies have been used.

3.3 Self-supervised Prompt Pre-training

To improve ability of the prompts to understand the documents and to help the model to adapt to the summarization tasks, soft prompts are further pre-trained on the corpus using summarization-oriented self-supervised objectives. Doing this also means that the prompts are well initialized for few-shot tuning.

We tested two strategies for constructing the self-supervised data. Each strategy was designed to suit a particular type of writing bias in the document. These are "lead" and "gap sentences generation".

Lead Lead bias is common in news articles, which usually follow an inverted pyramid structure where the first few sentences contain the most salient information (See et al., 2017; Yang et al., 2020). With this type of bias, we initially select the first three sentences as our target summary, and treated the rest of the document as the source text. With this type of prompt pre-training process, the model was able to infer the salient information based on the remaining text.

GSG Gap sentences generation applies to all documents that do not follow the lead bias structure (e.g., XSum (Narayan et al., 2018)). The strategy used here follows Zhang et al. (2020) , where we used ROUGE1-F1 (Lin, 2004) between each sentence x_i and the rest of the document as a proxy for the principal score, $s_i = rouge(x_i, D \setminus \{x_i\}), \forall i$. The top-$m$ most important sentences were selected according to s_i, and removed from the document. Then these m sentences are concatenated in the same order as the original text in the form of a pseudo summary. The remainder of the text is treated as a pseudo document.

With the constructed data, our designed prompts can be pre-trained and further tuned with few-shot examples.

3.4 Training Objective

The model is trained with maximum likelihood estimation (MLE). Given a ground-truth summary $Y = [y_1, y_2, ..., y_{|Y|}]$ for an input passage X, the objective is to minimize the negative log-likelihood

Datasets	CNNDM			XSum		
	train	dev	test	train	dev	test
Avg.Passage	697.45	676.64	717.92	396.53	387.62	380.55
Avg.Sum	55.91	51.97	58.62	22.90	23.29	22.11
Labled data	300	300	11,490	300	300	11,333

Table 2: Datasets statistics. "Avg.Passage" means the average length of passages and "Avg.Sum" means the average length of summaries.

of the target word sequence:

$$\mathcal{L} = -\sum_{t=1}^{|Y|} \log p_{\theta^*}(y_t|[P_{en}; X'_{in}], [P_{de}; y_1, ...y_{t-1}])$$
$$\theta^* = \{\theta; \theta_{p_{en}}; \theta_{p_{de}}; \theta_{p_{in}}\} \quad (6)$$

Note that only these prepended-prompts parameters ($\theta_{p_{en}}, \theta_{p_{de}}$) and the inner-prompts parameters ($\theta_{p_{in}}$) are optimized, the language model parameters (θ) are all frozen.

4 Experiments

Datasets We experimented with the CNN/DailyMail (CNNDM) dataset (Hermann et al., 2015) and the XSum dataset (Narayan et al., 2018). We chose these datasets because they differ in abstraction level and text length, which helps to show the generalization ability of our results.

We constructed the self-supervised pre-training data for CNNDM with Lead, and for XSum with GSG. We show details in Section A.1 in the appendix. Given that the lead bias structure exists only in some domain-specific datasets, we also conducted experiments to demonstrate the universality of the GSG to construct pseudo-data. The results are shown in Section A.3 in the appendix. Our few-shot training set D_{train} contained 300 document-summary pairs randomly sampled from the original training data. To tune the hyper-parameters and select the best checkpoint, we composed a validation set D_{dev} from the original validation data. Here, we were careful to ensure that $|D_{train}| = |D_{dev}|$ so that it fit into a true few-shot learning setting, following Perez et al. (2021). Since few-shot learning may have high variance, we sampled the examples with 5 different random seeds. We used the original test set to report our results, including the mean value and the standard deviation. Table 2 shows the statistics of the pre-processed corpus.

Setup The base version of BART was used in our work. Following Lester et al. (2021), we used 100 prompt tokens for both the encoder inputs and the decoder inputs. These prompts were randomly initialized from the set of vocabularies. The sequential and fixed-length inner-prompts require a maximum number. Hence, we counted the number of sentences in each document and divided the results into two groups – the 85% with the least sentences (Group A) and the 15% with the most sentences (Group B)[3]. We then set the number of prompts to the most number of sentences in Group A plus one, i.e., $n + 1$. For CNNDM, that number was 61 and, for XSum, it was 33. In this way, one inner-prompt token was assigned to each sentence up to n. For the excessively long documents in Group B, the text after n sentences was assigned an $n + 1$-th token. Further, we drew from a normal distribution $\mathcal{N}(0, 0.05)$ to initialize the inner-prompt embeddings[4]. Taking CNNDM as an example, all the tunable parameters that need to be stored amount to only 2×10^5. This is compared to the (1.4×10^8) parameters of full-model tuning. That equates to around 0.1% of the parameters for each dataset that need to be tuned and stored.

Evaluation Metrics We adopted ROUGE (Lin, 2004) to measure the quality of the summaries produced in our experiments. The F1 scores for ROUGE-1, ROUGE-2, and ROUGE-L between the ground-truth and the generated summaries are each reported.

Baseline Models We compared PSP to: **Prompt Tuning** (Lester et al., 2021), which only concatenates soft prompts into the encoder input; **Prefix Tuning** (Li and Liang, 2021), which adds a prefix to all the encoder layers, cross-attention layers, and the decoder layers; and **Full-Model Tuning**, which does not have any prompts and fine-tunes all the parameters of the pre-trained language model.

4.1 Experimental Results of Our Method

Table 3 presents the results of all PSP variants and baselines across CNNDM and XSum datasets. With the exception of the ROUGE-2 and ROUGE-L scores for the Prefix-Tuning on the CNNDM dataset, our proposed PSP, outperforms the others. However, PSP delivered a competitive result with only 3% of the parameters, which is an acceptable place to start. To our surprise, we observe that

[3]We made our division at 85% to ensure all embeddings of inner-prompt tokens could be fully trained, because sentences after the n-th only exist in 15% of the data.

[4]More information about implementation details are shown in Section A.2 in the appendix.

Model	CNNDM ROUGE-1	CNNDM ROUGE-2	CNNDM ROUGE-L	CNNDM PPL	XSum ROUGE-1	XSum ROUGE-2	XSum ROUGE-L	XSum PPL
Prompt Tuning	$30.58_{2.07}$	$11.93_{0.46}$	$21.73_{1.86}$	141.56	$29.63_{1.21}$	$8.84_{0.55}$	$22.00_{1.23}$	101.96
Prefix-Tuning	$37.12_{0.15}$	$\mathbf{16.59_{0.09}}$	$\mathbf{26.28_{0.06}}$	52.59	$32.18_{0.16}$	$11.13_{0.08}$	$25.50_{0.14}$	39.58
Full-Model Tuning	$38.03_{0.56}$	$16.01_{0.79}$	$25.21_{0.70}$	65.73	$32.85_{0.25}$	$10.52_{0.24}$	$25.15_{0.29}$	51.63
$\text{PSP}_{\texttt{Interval}}$	$37.82_{0.29}$	$15.40_{0.31}$	$25.10_{0.36}$	**45.54**	$\mathbf{32.86_{0.21}}$	$\mathbf{11.27_{0.08}}$	$\mathbf{25.64_{0.11}}$	44.25
$\text{PSP}_{\texttt{Sequential}}$	$37.82_{0.39}$	$15.58_{0.32}$	$25.16_{0.32}$	48.10	$32.57_{0.11}$	$10.97_{0.07}$	$25.39_{0.05}$	**35.70**
$\text{PSP}_{\texttt{Fixed-k}}$	$\mathbf{38.31_{0.15}}$	$15.94_{0.21}$	$25.41_{0.25}$	58.50	$32.81_{0.10}$	$11.15_{0.10}$	$25.48_{0.13}$	52.10

Table 3: Results on CNNDM and XSum Datasets. The experiments are conducted with 300 training samples and 300 validation samples on each dataset. We report the mean value and the standard deviation over 5 sampled datasets. k = 10 is chosen for $\text{PSP}_{\texttt{Fixed-k}}$. "PPL" represents the perplexity of generated summaries. A low perplexity indicates the summaries are fluent. Best results are bold and underline means our models outperform Full-model tuning.

50% of PSP's results surpass the full-model tuning, especially on XSum, as underlined in the table. Besides, results on the PPL metric show that PSP can generate more fluent summaries than other models. These results indicate that fine-tuning large language models is not necessarily a good or efficient idea with few-shot generation. It also shows that soft prompts with frozen language models are effective for few-shot abstractive summarization. Moreover, it statistically verifies that PSP with its three inner-prompt strategies is effective.

Model	# Train	# Store	ROUGE-1 CNNDM	ROUGE-1 XSUM
PSP	2.0×10^5	2.0×10^5	**38.32**	**32.86**
Prefix-Tuning	2.4×10^7	5.5×10^6	37.12	32.18
Prompt Tuning	7.7×10^4	7.7×10^4	30.58	29.63
Full-Model Tuning	1.4×10^8	1.4×10^8	38.03	32.85

Table 4: Comparison with baseline models on effectiveness and efficiency. "# Train" means the number of tuned parameters during training. " # Store" means the number of stored parameters. Best results are bold.

Efficiency v.s. effectiveness. We gave an overall comparison to baseline models on effectiveness and memory-efficiency, evaluated by ROUGE and the number of parameters, respectively. The results are shown in Table 4. Prompt Tuning has the least number of parameters, while its capacity is limited to this and lacks control over the decoder side, hence it can not perform natural language generation tasks well. We can see that substantial gains are made when going from vanilla Prompt Tuning to PSP. However, even if Prefix-Tuning is nearly thirty times more parameters than ours, there is either a marginal improvement or even performance decrease on some metrics. Besides, Prefix-Tuning relies on reparameterization tricks to stabilize the training, i.e., adds a MLP with large number of parameters to the training stage. Our method provides the best effectiveness-efficiency trade off, and outperforms full-model tuning with only 0.1% parameters, and presents competitive results against Prefix-Tuning with 3% parameters.

Human Evaluation We conducted a human evaluation study. To this end, we randomly selected 20 instances from the test set of each dataset. Ten graduate students with high levels of fluency in English were asked to assess the generated summaries and golden summaries from independent perspectives (Wang et al., 2021): *Informativeness* (how much useful information does the summary provide?), *Relevance* (how well does the summary reflect the input document?), and *Fluency* (how grammatically correct are the summary sentences and how easy are they to read?). Scoring followed the Best-Worst Scaling method (Kiritchenko and Mohammad, 2017). Participants were asked to select the best and worst summaries from each perspective. The scores were computed as the percentage of times a summary was chosen as the best minus the times it was selected as the worst. The scores ranged from -1 (worst) to 1 (best). Results are shown in Table 5. Qualitatively, we show several examples generated by different models and the reference in Table 14 and Table 15 in the appendix. Compared with all baselines, the summaries generated by PSP are always more fluent and relevant to the source document, consistent with the results of human evaluation. Further more, we found summaries generated by PSP and Prefix-Tuning are always similar in sentence patterns and expressions. However, Prefix-Tuning tends to generate texts shorter than PSP, which often leads to lack of information.

Methods	CNNDM			XSum		
	IF	RL	FL	IF	RL	FL
PSP	**0.500**	**0.708**	**0.667**	**0.217**	**0.275**	**0.492**
Prompt Tuning	-0.317	-0.758	-0.975	-0.336	-0.400	-0.867
Prefix-Tuning	-0.233	0.067	0.158	0.017	-0.008	0.292
Full-Model Tuning	0.067	-0.025	0.075	0.117	0.092	0.075

Table 5: Human evaluation results. Best results are bold.

k	D_{dev}			D_{test}		
	R-1	R-2	R-L	R-1	R-2	R-L
5	34.27	11.90	26.41	31.90	10.28	24.20
10	**35.31**	**12.88**	**26.85**	**32.89**	**11.13**	**25.51**
15	34.98	11.68	26.45	32.11	10.46	24.72
30	34.48	12.57	26.55	32.20	11.03	25.30

Table 6: Results of different fixed length k on validation set D_{dev} and test set D_{test} of XSum. "R-1" is short for "ROUGE-1", the same for "R-2" and "R-L".

Selection of fixed length k. As shown in Table 3, $\text{PSP}_{\texttt{Fixed-k}}$ performs consistently well on both datasets. So we further explored the influence of different length k, i.e., $k = 5, 10, 15, 30$, for inner-prompt tokens of the $\text{PSP}_{\texttt{Fixed-k}}$[5]. Table 6 presents the results of the variants on XSum. We observe the segmented spans with 10 tokens achieve the best performance. Interestingly, it can be induced that, to understand a document, it is possible to reorganize the sentence into several semantic units, where the number of the tokens is 10 on average. We also report results of different k on our validation set in Table 6. The ranking is consistent with the test set. From a practical perspective, when applying PSP to a new dataset, we can choose the best k based on the validation set.

4.2 Analyses on Soft Prompts

Whether our model attends to understand documents? According to Figure 2, we further present the encoder-decoder attention distribution of the PSP. The comparison visualization is shown in Figure 5. We find the following enhancement of our model by introducing the inner prompts. First, the PSP model strengthens the associations between the encoder prompts and the decoder prompts compared to the original model. Second, the soft prompt P_{en} has more opportunities to be related to the output Y, indicating the semantic relations between them. Third, the output Y assigns more attention to the source document X. This suggests that the hidden structure of the document is emphasized, increasing the capability of understanding its semantics. As such, these prompts can properly elect salient information from the document and prompt the model to generate the output.

[5]The average number of tokens per sentence in both datasets was about 18, so we did not consider fixed lengths of 20, for its similarity to the $\text{PSP}_{\texttt{Sequential}}$.

Model	CNNDM			XSum		
	R-1	R-2	R-L	R-1	R-2	R-L
Soft prompts (en.&de., 100)	36.89	14.96	24.63	29.36	9.90	22.92
Soft prompts (en.&de., 150)	35.71	14.86	23.97	28.94	9.52	22.24
Soft prompts (en.&de.&ip., 100)	**37.87**	**15.83**	**25.37**	**31.95**	**10.52**	**24.80**

Table 7: Results of different architectures of soft prompts on CNNDM and XSum, where "en." "de." "ip." are short for encoder, decoder and inner prompts, respectively. Numbers in parentheses represent the number of prompt tokens we prepended before the encoder and decoder input.

Model	ROUGE-1	ROUGE-2	ROUGE-L
Soft prompts (en.&de., shared)	36.06	14.30	24.24
Soft prompts (en.&de., separate)	36.37	14.41	24.46

Table 8: Results of basic soft prompts on the CNNDM.

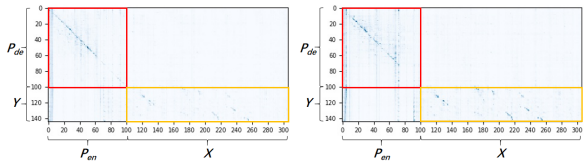

Figure 5: Visualization of the encoder-decoder attention weights of the model with only prompts across the encoder and the decoder (left) and PSP (right). Detailed descriptions refer to Figure 2.

Do inner prompts assist the model to understand the content of documents or simply increase the model's capacity? Instead of using inner-prompts, we prepended additional tunable tokens (i.e. 150 tokens) in front of the encoder and the decoder inputs. Comparison results are shown in Table 7. Despite the larger capacity, soft prompts with 150 tunable tokens before the input performed the worst, denoted as *soft prompts (en.&de., 150)*. This suggests the inner-prompts with a few parameters do help to understand the document by prompting the structures, rather than simply add more trainable parameters to increase the model's capacity.

Further insight on soft prompts across the encoder and the decoder. To verify our hypothesis that the decoder prompts largely copy the behaviour of the encoder prompts, we shared similar embeddings of the soft prompts before the encoder and the decoder. In Table 8, we observe the Soft prompts (en.&de., shared) and (en.&de., separate) almost

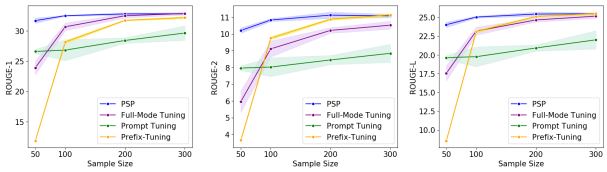

Figure 6: k-shot summarization results on XSum.

Model	ROUGE-1	ROUGE-2	ROUGE-L
Full-Model Tuning	11.69	2.67	7.74
Prefix-Tuning	11.76	2.63	7.93
Prompt Tuning	9.40	1.86	6.19
$\text{PSP}_{\texttt{Interval}}$	**17.16**	**3.36**	**12.65**

Table 9: Zero-shot results on XSum.

perform identical results. Although the parameters are only half of the original model, the performance consistently remains competitive. This shows that the shared prompts can extract important information from the document and further guide the language model to generate consistently good summaries more efficiently.

4.3 Analysis on Few-shot and Zero-shot Summarization

To examine the performance of different methods under few-shots, we further randomly sampled number of {50, 100, 200} as the settings. Figure 6 reports a more detailed overview of all models' performance across a range of different few-shots. The ROUGE scores of our model generally outperform other baselines and remain steady across different scenarios. Especially, the PSP with only 50 examples receives the most significant improvements, while the Prefix-Tuning doesn't even work (tuning based on $\text{BART}_{\texttt{base}}$) possibly due to its instability of the model. Moreover, we report the results of zero-shot on XSum in Table 9. Benefiting from the knowledge gained in the pre-training phase, our model shows a significant advantage of zero-shot adaptation in generating quality summaries.

4.4 The Performance of Pre-training on Prefix-Tuning

A crucial strategy for PSP is the pre-training of soft prompts. To give a fairly comparison, we performed prefix pre-training for Prefix-Tuning in the same way with the PSP. The results are shown in Table 10. We can find that the Prefix model obtains improvements on the XSum dataset after adopting the pre-training strategy, but underperforms the original one on the CNNDM dataset. It indicates

Method	CNNDM			XSum		
	ROUGE-1	ROUGE-2	ROUGE-L	ROUGE-1	ROUGE-2	ROUGE-L
Prefix-Tuning	$37.12_{0.15}$	$16.59_{0.09}$	$26.28_{0.06}$	$32.18_{0.16}$	$11.13_{0.08}$	$25.50_{0.14}$
Prefix-Tuning w/ Pre.	$37.35_{0.58}$	$16.08_{0.37}$	$25.95_{0.50}$	$33.39_{0.10}$	$11.61_{0.06}$	$26.07_{0.09}$

Table 10: Test set results of Prefix-Tuning. "w/ Pre." means that we pre-trained the prefix with pseudo data.

that Prefix-Tuning shows limited potential compared to our model. We induce that the pre-training for Prefix-Tuning raises over-fitting risk due to its sensitivity to different data or parameter settings.

4.5 Ablation Study

We conducted experiments to examine the effectiveness of the major components of our model, and Table 11 shows the ablation results across the two datasets. We observed both the prompt pre-training operation and the inner-prompts component contribute to the main model. Notably, with the removal of each component, the model becomes considerably unstable, indicated by the variance shown in the ablation results. Comparably, prompt pre-training in our model accounts for more importance on the XSum dataset whose summaries have a higher abstract level (we assume it's more "difficult") than the CNNDM. In sum, these two components support the performance and stability of our model in terms of summarization adaption (by prompt pre-training) and structural documents understanding (by inner-prompts).

Method	CNNDM			XSum		
	ROUGE-1	ROUGE-2	ROUGE-L	ROUGE-1	ROUGE-2	ROUGE-L
$\text{PSP}_{\texttt{Fixed-k}}$	$38.31_{0.15}$	$15.94_{0.21}$	$25.41_{0.25}$	$32.81_{0.10}$	$11.15_{0.10}$	$25.48_{0.13}$
w/o PP	$37.30_{0.56}$	$15.45_{0.39}$	$24.93_{0.38}$	$32.17_{0.16}$	$10.69_{0.13}$	$25.02_{0.21}$
w/o IP	$37.76_{0.28}$	$15.22_{0.31}$	$24.80_{0.40}$	$32.59_{0.17}$	$11.14_{0.17}$	$25.46_{0.24}$
w/o PP & IP	$36.88_{0.42}$	$14.96_{0.45}$	$24.63_{0.40}$	$29.35_{1.5}$	$9.87_{0.43}$	$22.89_{1.19}$

Table 11: Ablation study of PSP on two datasets. "w/o" means without. "PP" and "IP" are short for Prompt Pre-training and Inner-Prompts, respectively. The variance of each result is provided.

5 Related Work

Few-Shot Abstractive Summarization In practical application scenarios, the lack of manual constructed document-summary pairs or labeled data makes data-driven neural models performs badly (Hu et al., 2021, 2020). Fabbri et al. (2020) condense characteristics of the target dataset into Wikipedia data to construct pseudo-summaries. Bražinskas et al. (2020) introduce plug-in networks to reproduce characteristics of the target dataset with only a small set of labeled examples. Bai et al. (2021) conduct cross-lingual summarization

in a low-resource setting. Yu et al. (2021) design the second phase of pre-training on large-scale generative models before fine-tuning. In this paper, we construct pseudo-summary corpus with heuristic rules, providing a better parameter initialization for soft prompts under few-shot settings. More importantly, we design summarization-oriented soft prompts to help the model produce few-shot summaries.

Prompt Learning The emergence of GPT-3 (Brown et al., 2020) introduces the concept of *"prompting"*. One only needs to assemble a task description and few examples into a prompt, and then prepend it to the task input. With the large-scale frozen parameters, a pre-trained model can generate the output without any task-specific tuning. However, task description is error-prone while there is no unified, explicit, and effective way to build these hard prompts manually (Logan IV et al., 2021). Hence, several works (Gao et al., 2020; Jiang et al., 2020; Shin et al., 2020) are proposed to generate prompts automatically, but they all restrict prompts to discrete spaces. These discrete prompts are less expressive and sub-optimal. To overcome the shortcomings of hard prompts, Li and Liang (2021) propose "Prefix-Tuning". This method only tunes prefix activation prepended to all transformer layers, and keeps the LM parameters frozen. To further simplify, Prompt Tuning (Lester et al., 2021) only prepends tunable tokens to the encoder input, and keeps all other parameters frozen. Logan IV et al. (2021) and Gu et al. (2021) propose to use pre-training to boost the low performance of Prompt Tuning for few-shot learning. In this work, we fit the structure of Prompt Tuning to text generation models, proposing encoder prompts, decoder prompts, and inner prompts. We successfully apply prompt tuning methods to few-shot abstractive summarization task.

6 Conclusion

In this paper, we present a novel pre-trained soft prompts architecture (PSP) specifically designed for few-shot abstractive summarization. We design continuous input embeddings across an encoder and a decoder alongside several kinds of inner-prompts placed in the text, assisting the model better to understand documents and guide accurate generation. Empirical results find the necessity of using prompt pre-training for few-shot/zero-shot abstractive summarization. Extensive experiments and analyses show that the proposed PSP provides the best effectiveness-efficiency trade off among all the baseline methods.

7 Acknowledgments

The research presented in this publication was sponsored by CCF Fund For Young Scholars, and Joint Funds of the National Natural Science Foundation of China (Grant No. U21B2009).

References

Yu Bai, Yang Gao, and Heyan Huang. 2021. Cross-lingual abstractive summarization with limited parallel resources. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 6910–6924, Online. Association for Computational Linguistics.

Arthur Bražinskas, Mirella Lapata, and Ivan Titov. 2020. Few-shot learning for opinion summarization. *arXiv preprint arXiv:2004.14884*.

Tom B Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. 2020. Language models are few-shot learners. *arXiv preprint arXiv:2005.14165*.

Alexander R Fabbri, Simeng Han, Haoyuan Li, Haoran Li, Marjan Ghazvininejad, Shafiq Joty, Dragomir Radev, and Yashar Mehdad. 2020. Improving zero and few-shot abstractive summarization with intermediate fine-tuning and data augmentation. *arXiv preprint arXiv:2010.12836*.

Tianyu Gao, Adam Fisch, and Danqi Chen. 2020. Making pre-trained language models better few-shot learners. *arXiv preprint arXiv:2012.15723*.

Yuxian Gu, Xu Han, Zhiyuan Liu, and Minlie Huang. 2021. Ppt: Pre-trained prompt tuning for few-shot learning. *arXiv preprint arXiv:2109.04332*.

Karl Moritz Hermann, Tomas Kocisky, Edward Grefenstette, Lasse Espeholt, Will Kay, Mustafa Suleyman, and Phil Blunsom. 2015. Teaching machines to read and comprehend. *Advances in neural information processing systems*, 28:1693–1701.

Xuming Hu, Lijie Wen, Yusong Xu, Chenwei Zhang, and S Yu Philip. 2020. Selfore: Self-supervised relational feature learning for open relation extraction. In *Proc. of EMNLP*, pages 3673–3682.

Xuming Hu, Chenwei Zhang, Yawen Yang, Xiaohe Li, Li Lin, Lijie Wen, and S Yu Philip. 2021. Gradient imitation reinforcement learning for low resource relation extraction. In *Proc. of EMNLP*, pages 2737–2746.

Zhengbao Jiang, Frank F Xu, Jun Araki, and Graham Neubig. 2020. How can we know what language models know? *Transactions of the Association for Computational Linguistics*, 8:423–438.

Woojeong Jin, Yu Cheng, Yelong Shen, Weizhu Chen, and Xiang Ren. 2021. A good prompt is worth millions of parameters? low-resource prompt-based learning for vision-language models. *arXiv preprint arXiv:2110.08484*.

Svetlana Kiritchenko and Saif M Mohammad. 2017. Best-worst scaling more reliable than rating scales: A case study on sentiment intensity annotation. *arXiv preprint arXiv:1712.01765*.

Brian Lester, Rami Al-Rfou, and Noah Constant. 2021. The power of scale for parameter-efficient prompt tuning. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 3045–3059, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Veselin Stoyanov, and Luke Zettlemoyer. 2020. Bart: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7871–7880.

Xiang Lisa Li and Percy Liang. 2021. Prefix-tuning: Optimizing continuous prompts for generation. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 4582–4597, Online. Association for Computational Linguistics.

Chin-Yew Lin. 2004. Rouge: A package for automatic evaluation of summaries. In *Text summarization branches out*, pages 74–81.

Xiao Liu, Yanan Zheng, Zhengxiao Du, Ming Ding, Yujie Qian, Zhilin Yang, and Jie Tang. 2021. Gpt understands, too. *arXiv preprint arXiv:2103.10385*.

Yang Liu and Mirella Lapata. 2019. Text summarization with pretrained encoders. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 3730–3740.

Yinhan Liu, Jiatao Gu, Naman Goyal, Xian Li, Sergey Edunov, Marjan Ghazvininejad, Mike Lewis, and Luke Zettlemoyer. 2020. Multilingual denoising pre-training for neural machine translation. *Transactions of the Association for Computational Linguistics*, 8:726–742.

Robert L Logan IV, Ivana Balažević, Eric Wallace, Fabio Petroni, Sameer Singh, and Sebastian Riedel. 2021. Cutting down on prompts and parameters: Simple few-shot learning with language models. *arXiv preprint arXiv:2106.13353*.

Christopher D Manning, Mihai Surdeanu, John Bauer, Jenny Rose Finkel, Steven Bethard, and David McClosky. 2014. The stanford corenlp natural language processing toolkit. In *Proceedings of 52nd annual meeting of the association for computational linguistics: system demonstrations*, pages 55–60.

Shashi Narayan, Shay B. Cohen, and Mirella Lapata. 2018. Don't give me the details, just the summary! Topic-aware convolutional neural networks for extreme summarization. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, Brussels, Belgium.

Ethan Perez, Douwe Kiela, and Kyunghyun Cho. 2021. True few-shot learning with language models. *arXiv preprint arXiv:2105.11447*.

Timo Schick and Hinrich Schütze. 2020. Few-shot text generation with pattern-exploiting training. *arXiv preprint arXiv:2012.11926*.

Abigail See, Peter J Liu, and Christopher D Manning. 2017. Get to the point: Summarization with pointer-generator networks. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1073–1083.

Taylor Shin, Yasaman Razeghi, Robert L Logan IV, Eric Wallace, and Sameer Singh. 2020. Autoprompt: Eliciting knowledge from language models with automatically generated prompts. *arXiv preprint arXiv:2010.15980*.

Ashish Vaswani, Noam M. Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. *ArXiv*, abs/1706.03762.

Haonan Wang, Yang Gao, Yu Bai, Mirella Lapata, and Heyan Huang. 2021. Exploring explainable selection to control abstractive summarization. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 35, pages 13933–13941.

Wenbo Wang, Yang Gao, He-Yan Huang, and Yuxiang Zhou. 2019. Concept pointer network for abstractive summarization. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 3076–3085.

Thomas Wolf, Julien Chaumond, Lysandre Debut, Victor Sanh, Clement Delangue, Anthony Moi, Pierric Cistac, Morgan Funtowicz, Joe Davison, Sam Shleifer, et al. 2020. Transformers: State-of-the-art natural language processing. In *Proceedings of*

the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations, pages 38–45.

Ziyi Yang, Chenguang Zhu, Robert Gmyr, Michael Zeng, Xuedong Huang, and Eric Darve. 2020. Ted: A pretrained unsupervised summarization model with theme modeling and denoising. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: Findings*, pages 1865–1874.

Tiezheng Yu, Zihan Liu, and Pascale Fung. 2021. Adaptsum: Towards low-resource domain adaptation for abstractive summarization. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 5892–5904.

Jingqing Zhang, Yao Zhao, Mohammad Saleh, and Peter Liu. 2020. Pegasus: Pre-training with extracted gap-sentences for abstractive summarization. In *International Conference on Machine Learning*, pages 11328–11339. PMLR.

	CNNDM Pseudo Corpus	XSum Pseudo Corpus
# of Original Passages	287,113	204,017
# of Pre-training Data	284,177	158,499

Table 12: Pseudo-summarization corpus statistics. "# of Original Passages" means the number of original passages in the training set, "# of Pre-training data" means the number of pseudo data after data cleaning.

	ROUGE-1	ROUGE-2	ROUGE-L
$\text{PSP}_{\texttt{Lead}}$ (w/o inner-prompts)	37.66	15.07	24.52
$\text{PSP}_{\texttt{GSG}}$ (w/o inner-prompts)	37.04	15.04	25.20

Table 13: Results on CNNDM by using the Lead and the GSG to construct pseudo-data for prompt pre-training.

A Appendix

A.1 Constructing Pesudo Data for Pre-training

We constructed the pseudo data for CNNDM with **Lead**. We also conducted a simple data cleaning procedure to the self-supervised pre-train corpus. First, we cleaned away irrelevant information, such as media names, reporter names or dates from the summaries. Second, for those summaries with less than 50 tokens, we iteratively collected the first sentence of the remaining text to the pseudo summary, until the length of summary reaches 70. This procedure was set up to prevent the target text from being too short to form a meaningful summary. Third, for those samples in which the source document is shorter than its summary, we filtered them out.

For XSum, we constructed the pseudo data for pre-training following **GSG**. The top-1 most important sentence was selected as the pseudo summary. Then we filtered out those pseudo summaries that are not relevant enough to the pseudo passages. In particular, we leveraged hand-written summaries in the few-shot dataset to determine the filtering threshold of pseudo data. We calculated the ROUGE-1 F1 between each ground-truth summary and its corresponding passage, represented as Ri. Then we calculated the mean and variance of Ri: $\epsilon = \frac{1}{n}\sum_{i=1}^{n} Ri$, $\sigma^2 = \frac{1}{n}\sum_{i=1}^{n}(Ri - \epsilon)^2$, and $\epsilon - \sigma^2$ was used as a lower-bound threshold to filter out low quality pseudo data. For those pseudo samples where ROUGE1-F1 between the pseudo summary and the pseudo passage is lower than the threshold $\epsilon - \sigma^2$, we filtered them out. Finally, we conducted pre-training on our soft prompts with these filtered pseudo-data. Table 12 shows the statistics for the pre-training data corpus.

A.2 Implementation Details

We first split sentences with the Stanford CoreNLP toolkit (Manning et al., 2014), and the input documents were truncated to 1024 BPE tokens. We adopted BART-base for all the experiments. Our implementation was based on the Hugging Face Transformer models (Wolf et al., 2020). We used a mini-batch size of 8 with a gradient accumulation for 10 iterations. We used Adam optimizer with momentum β_1 = 0.9, β_2 = 0.998 and noam decay. In the stage of pre-training, the peak value of learning rate was 1e-3, and we set the warm up ratio to 10%. During fine-tuning, the peak value of learning rate was 3e-4, and we set the warm up steps to 100 with 400 epochs. In the decoding stage, we used beam search with a beam size of 4. The decoding process will not stop until an end-of sequence (EOS) token was emitted or the length of the generated summary reached to 256 tokens. All models were trained on 4 TITAN RTX GPUs.

A.3 The Universality of GSG to Construct Pseudo-data

To demonstrate the universality of using the GSG method to construct pseudo-data for prompt pre-training, we conducted a complimentary experiment to testify its effect on the CNNDM[6]. Specifically, we selected $m = 3$ important sentences. Results in Table 13 indicate that the PSP model pre-trained by GSG is equally effective with the original $\text{PSP}_{\texttt{Lead}}$, showing that the GSG can be universally employed to pre-train soft prompts for abstractive summarization.

[6]We do not conduct ablation experiments on XSum, as there is no " lead bias" in this dataset. So it is inappropriate to take the first sentences of the passage as the pseudo summary.

Source	Manchester City are monitoring Juventus striker Alvaro Morata. The Spain international was a target for Arsenal a year ago when he left Real Madrid but opted for a move to Italy instead. Real have a buy-back option set at around £15million but are unlikely to take that up. Manchester City are keeping tabs on Juventus striker Alvaro Morata ahead of possible summer bid . Morata closes down Lazio goalkeeper Federico Marchetti during Juve's Serie A win on Saturday . City also remain keen on Juventus midfielder Paul Pogba but face competition from Paris Saint Germain . Morata has been a success at Juventus but the Italians are always susceptible to offers at the right price and are making plans to bring in Edinson Cavani from Paris Saint Germain despite the French club's resistance. PSG also are one of the frontrunners to sign Juve midfielder Paul Pogba. City remain one of the clubs pushing to sign the France international too and Juventus have City's Edin Dzeko, Aleksandar Kolarov and Stevan Jovetic on their list of potential targets for the summer.
PSP	Manchester City are keeping tabs on Juventus striker Alvaro Morata . The Spain international was a target for Arsenal a year ago when he left Real Madrid but opted for a move to Italy instead . Juventus have a buy-back option set at around £15million but are unlikely to take that up .
Prefix-Tuning	Manchester City are keeping tabs on Juventus striker Alvaro Morata ahead of possible summer bid. Morata closes down Lazio goalkeeper Federico Marchetti during Juve's Serie A win on Saturday.
Prompt Tuning	LMorata is a target for Real Madrid, who have a buy - back option set at around £ 15million . He has been a success at Real Madrid but opted for a move to Italy instead . The Italian club are unlikely to take that up . . .." . .." . ." ." . ."." . ." . ."". . ." . . " . . .. [and] . . (He) had a great season at Real but who more had been about" . . , but who else had been close enough to sign him? . . He had her all after." . .) . . She didn't year when when when two over people A lot more time than two theret. . ." —LastJustInOther
Full-Model Tuning	NEW City are monitoring Juventus striker Alvaro Morata . The Spain international was a target for Arsenal a year ago when he left Real Madrid . Morata has a buy-back option set at around £15million but is unlikely to be offered by the French side .
Reference	Alvaro Morata had attracted interest from Arsenal before joining Juventus . Spain international made move to Italy in £15million deal from Real Madrid . Manchester City are monitoring the 22-year-old after impressive season .
Source	Nairobi, Kenya (CNN)University of Nairobi students were terrified Sunday morning when they heard explosions – caused by a faulty electrical cable – and believed it was a terror attack, the school said. Students on the Kikuyu campus stampeded down the halls of the Kimberly dormitory, and some jumped from its fifth floor, the university said. Hundreds were injured and were taken to hospitals. One person died, according to the school. The confusion and panic came less than two weeks after Al-Shabaab slaughtered 147 people at a college in Garissa, Kenya. Kenyan teachers and students have said they fear being targeted by the Somalia-based terrorists. On Sunday, as many as 108 students from the University of Nairobi were admitted to Kenyatta National Hospital. Among them, at least 63 students have been discharged, and at least four are slated for surgery, the school said. Almost all of the 54 students being treated at PCEA Kikuyu Hospital have been released, the university said. Kenya Power authorities and its CEO are at the school and looking into the electrical issue. Normal power supply will resume after repairs, the university said. "As we mourn the unfortunate loss of the departed student, we are also praying for the quick recovery of those who were injured," said Vice Chancellor Peter M.F. Mbithi in a statement. He called on the students, staff and public to remain calm. CNN's Lillian Leposo reported from Nairobi and Ashley Fantz wrote this story in Atlanta.
PSP	University of Nairobi students were terrified Sunday morning when they heard explosions – caused by a faulty electrical cable – and believed it was a terror attack . Hundreds were injured and were taken to hospitals . One person died, according to the school . The confusion and panic came less than two weeks after Al- Shabaab slaughtered 147 people at a college in Garissa .
Prefix-Tuning	University of Nairobi students were terrified Sunday morning when they heard explosions – caused by a faulty electrical cable – and believed it was a terror attack.
Prompt Tuning	L Hundreds were injured and taken to hospitals . Hundreds of students were evacuated from the dormitory . The school said they feared being targeted by al-Shabaab . Hundreds were evacuated and treated in hospitals . The university said it is investigating the cause of the explosion . . . The explosion was caused by a faulty electrical cable. . .. . ." . . ." ." ." . ."." . .." . .""People were terrified," said the school's vice chancellor . "People were screaming, but who more had been were about" . "We had no idea what was going on but who else had been about to blow her all after." ... .. ." .."." .""They were terrified at the time than two overtakes" —LastJustIn3
Full-Model Tuning	NEW students panicked when they heard explosions – caused by a faulty electrical cable – and believed it was a terror attack, university says . As many as 108 students from University of Nairobi were admitted to Kenyatta National Hospital . One person died, according to the school .
Reference	Students stampeded; some jumped from a fifth story at a dorm; one student died, school officials say . The blasts were caused by faulty electrical cable, and Kenya Power is at the school . The panic came less than two weeks after terrorists attacked Kenya's Garissa University .

Table 14: Qualitative examples of CNNDM.

Source	Brunon Kwiecien, 48, was convicted of planning a terrorist attack, illegal weapons possession and inciting two students to carry out an attack.He suggested he had been manipulated by Polish intelligence agents.Kwiecien was said to be fascinated with Norwegian mass killer Anders Behring Breivik.Right-wing extremist Breivik killed 77 people in a bombing and shooting rampage in Norway in July 2011.Kwiecien, a former professor at Krakow's University of Agriculture, was arrested in 2012.Investigators believe he wanted to target parliament with four tonnes of explosives while then-President Bronislaw Komorowski and former Prime Minister Donald Tusk were due to attend, the trial heard."If Brunon Kwiecien hadn't been stopped, we would be talking amid the ruins of the state today," said judge Aleksandra Almert, according to the AFP agency.While admitting he planned to carry out the attack, he also said he was subject to "provocation" by the intelligence services.Kwiecien is the first Pole to be accused of terrorism, Polish media reported. He has no known links to established extremist groups.
PSP	A Pole has been convicted of planning a terrorist attack in Poland, a court heard.
Prefix-Tuning	A Pole has been convicted of planning to carry out a terrorist attack in Poland.
Prompt Tuning	AA Polish man has been convicted of planning a terrorist attack in the Polish capital, Warsaw, on Thursday.
Full-Model Tuning	A Pole has been found guilty of planning a terrorist attack in the Polish state of Krakow.
Reference	A Polish university lecturer has been sentenced to 13 years in jail for plotting to ram a car packed with explosives into parliament.
Source	Schmidt was sent off by the referee for insulting Hoffenheim's Julian Nagelsmann in Saturday's 3-0 home loss."That was nothing, what sort of a nutcase are you? Just shut your mouth," Schmidt shouted after going 2-0 down.The 49-year-old has been banned for two games and handed a 15,000 euros (£13,373) fine.The German was sanctioned after triggering a suspended sentence from February this year.He had been banned for three games, with a further two in the event of a repeat offence before June 2017, for refusing a referee's order to leave the sidelines during a 1-0 defeat to Borussia Dortmund.Schmidt will be unable to have any contact with the team for half an hour before, during and after Tuesday's German Cup second-round match against Lotte and Saturday's league match against Wolfsburg.Leverkusen's director of sport Rudi Voller has sought a meeting with the head of the disciplinary committee.
PSP	Leverkusen defender Christian Schmidt has been banned for two games for insulting the referee.
Prefix-Tuning	Leverkusen midfielder Matthias Schmidt has been banned for two games after refusing to leave the sidelines during a match against Wolfsburg.
Prompt Tuning	ALeverkusen midfielder Christian Schmidt has been banned for two games for insulting the referee in a game against Hoffenheim on Saturday..'
Full-Model Tuning	Aeverkusen manager Gerhard Schmidt has been banned for two games for insulting the head of the German national team.
Reference	Bayer Leverkusen head coach Roger Schmidt has been banned and fined for calling an opposing manager "a nutcase" during a Bundesliga game.

Table 15: Qualitative examples of XSum.

Continuous Decomposition of Granularity for Neural Paraphrase Generation

Xiaodong Gu[1], **Zhaowei Zhang**[1], **Sang-Woo Lee**[2], **Kang Min Yoo**[2], **Jung-Woo Ha**[2]
[1] School of Software, Shanghai Jiao Tong University
[2] NAVER AI Lab
{xiaodong.gu,andy_zhangzw}@sjtu.edu.cn
{sang.woo.lee,kangmin.yoo,jungwoo.ha}@navercorp.com,

Abstract

While Transformers have had significant success in paragraph generation, they treat sentences as linear sequences of tokens and often neglect their hierarchical information. Prior work has shown that decomposing the levels of granularity (e.g., word, phrase, or sentence) for input tokens has produced substantial improvements, suggesting the possibility of enhancing Transformers via more fine-grained modeling of granularity. In this work, we propose continuous decomposition of granularity for neural paraphrase generation (C-DNPG). In order to efficiently incorporate granularity into sentence encoding, C-DNPG introduces a granularity-aware attention (GA-Attention) mechanism which extends the multi-head self-attention with: 1) a granularity head that automatically infers the hierarchical structure of a sentence by neurally estimating the granularity level of each input token; and 2) two novel attention masks, namely, *granularity resonance* and *granularity scope*, to efficiently encode granularity into attention. Experiments on two benchmarks, including Quora question pairs and Twitter URLs have shown that C-DNPG outperforms baseline models by a remarkable margin and achieves the state-of-the-art results in terms of many metrics. Qualitative analysis reveals that C-DNPG indeed captures fine-grained levels of granularity with effectiveness.

1 Introduction

With the continued success in NLP tasks (Vaswani et al., 2017), Transformer has been the mainstream neural architecture for paraphrase generation (Li et al., 2019; Kazemnejad et al., 2020; Guo et al., 2021; Hosking et al., 2022; Goyal and Durrett, 2020). The core component of Transformer is the self-attention network (SAN) (Vaswani et al., 2017) which computes sentence representations at each position by baking representations over all other positions in a parallel way. Despite their effectiveness, Transformer has been shown to be limited in structure modeling, that is, they process disperse words in a flat and uniform way without explicit modeling of the hierarchical structures (Raganato and Tiedemann, 2018; Hao et al., 2019; Li et al., 2020).

Text	What is the reason for World War II ?
decomposition 1	What is the reason for World War II ?
decomposition 2	What is the reason for World War II ?
decomposition 3	What is the reason for World War II ?
decomposition 4	What is the reason for World War II ?
decomposition 5	What is the reason for World War II ?

⇓

Levels of granularity (marked as superscripts):

What[1] is[1] the[2] reason[3] of[2] World[4] War[4] II[5] ?

Table 1: A motivation example of multi-granularity text decomposition. The given sentence can be decomposed according to 5 increasing levels of granularity, each corresponding to a partition of the sentence into a template (blue) at a specific level together with details (orange). Each row with colored text denotes a level of granularity where the blue words are in the sentence level (templates) and the remaining words are in the phrase level (details). The bottom half shows the level of granularity for each word according to the decomposition. We use integer numbers to indicate the extent of the granularity: the greater the number, the more detailed the word is.

One potential route towards addressing this issues is multi-granularity text modeling (illustrated in Table 1) – which decomposes texts into multiple levels of granularity such as words, phrase and sentence (Li et al., 2019; Wiseman et al., 2018; Hao et al., 2019). For example, Li et al. (2019) attempts to achieve this via *decomposable neural paraphrase generator* (DNPG). DNPG decomposes paraphrase generation by Transformers into two levels of granularity: phrase-level (details) and sentence-level (templates). This allows a more flexible and controllable generating process. Paraphrases can be generated through rephrasing the templates while copying the detailed phrases. The

Proceedings of the 29th International Conference on Computational Linguistics, pages 6369–6378
October 12–17, 2022.

decomposition is realized using a granularity separator, multiple encoder-decoder pairs, and an aggregator that summarizes results from all granularity levels.

Despite showing promising results, DNPG only captures a discrete (coarser-grained) decomposition of granularity, which restricts the capacity in representing more fine-grained semantic hierarchy. Furthermore, DNPG models the different levels of granularity using multiple encoders and decoders. That amounts to training multiple Transformers. The computational cost increases greatly as the number of granularity levels grows.

In this paper, we present C-DNPG (stands for *continuous decomposition of granularity for paraphrase generation*), a simple, fine-grained, and seamlessly integrated model for granularity-aware paragraph representation. C-DNPG extends the vanilla attention network with a granularity head, which neurally estimates a continuous level of granularity for each token. In order to efficiently encode granularity into Transformers, C-DNPG adjusts the original self-attention weights using two novel attention masks: 1) a *granularity-resonance mask* which encourages attentions to exist between tokens with similar granularity; and 2) a *granularity scope mask* which encourages a small attention scope for lower-level (words or phrases) tokens. The granularity-aware attention mechanism provides a continuous modeling of sentence granularity and can be seamlessly integrated into the vanilla Transformer as the basic processing cell.

We evaluate the proposed C-DNPG on two commonly-used benchmarks, including the Quora question pairs and Twitter URL paraphrasing. Experimental results show that C-DNPG remarkably outperforms baseline models on both benchmarks and achieve the state-of-the-art results in many metrics. Qualitative study confirms the ability of the proposed approach in modeling fine-grained granularity.

Our contributions can be summarized as follows:

- We present a novel granularity-aware attention mechanism which supports a fine-grained decomposition of granularity for input tokens and hence yields a **continuous modeling of granularity** for natural language sentences.
- The proposed granularity-aware attention network can be **seamlessly integrated** into the Transformer for granularity-aware paraphrase generation.
- We conduct extensive evaluations of our methods on two popular paraphrase generation benchmarks and show that C-DNPG remarkably outperforms previous works in terms of quantitative and qualitative results. We release all data and code at `https://github.com/guxd/C-DNPG`.

2 Background

Our approach is extended based on the Transformer and self-attention networks. We begin by introducing the background of these techniques.

2.1 Self-Attention Networks

The attention mechanism is a function which maps a query vector to a set of key-value vector pairs and summarizes an output vector as a weighted sum of the value vectors. The weight assigned to each value is computed using the query and key vectors. A typical attention function, for example, is the scaled dot-product attention (Vaswani et al., 2017):

$$\begin{aligned} &\text{Attention}(\mathbf{Q}, \mathbf{K}, \mathbf{V}) = \mathbf{A}^T\mathbf{V}, \\ &\mathbf{A} = \text{softmax}(\mathbf{Q}\mathbf{K}^T/\sqrt{d_k}) \end{aligned} \tag{1}$$

where $\mathbf{Q}$, $\mathbf{K}$, $\mathbf{V} \in \mathbb{R}^{L\times d}$ represent the query, key and value vectors, respectively. $\mathbf{A} \in [0,1]^{L\times L}$ is the attention score matrix with each $\mathbf{A}_{ij} = \mathbf{q}_i^T\mathbf{k}_j$; d_k denotes the dimension size of the key vector.

In particular, the *self-attention network* (SAN) is a special attention mechanism that computes the attention function over a single sequence. For a given sequence represented as a list of hidden states $\mathbf{H} = [\mathbf{h}_1, \ldots, \mathbf{h}_N] \in \mathbb{R}^{N\times d}$, the self-attention network computes representations of the sequence by relating hidden states at different positions (Vaswani et al., 2017):

$$\begin{aligned} &\mathbf{Q},\ \mathbf{K}, \mathbf{V} = \mathbf{W}^Q\mathbf{H}, \mathbf{W}^K\mathbf{H}, \mathbf{W}^V\mathbf{H} \\ &\text{SelfAttn}(\mathbf{H}) = \text{Attention}(\mathbf{Q}, \mathbf{K}, \mathbf{V}) \end{aligned} \tag{2}$$

where $\mathbf{W}^q$, $\mathbf{W}^k$, and $\mathbf{W}^v$ are parameters to transform the input representation $\mathbf{H}$ to the query, key, and value respectively. Self-attention produces an abstraction and summary of a sequence in the hidden space and outputs the transformed hidden states (Vaswani et al., 2017).

2.2 Transformer

Transformer is an encoder-decoder model that is built upon the self-attention networks. It has been

the common architectural choice for modeling paraphrase generation (Li et al., 2019; Guo et al., 2021; Kazemnejad et al., 2020). Transformer encodes a source sequence into hidden vectors and then generates a target sequence conditioned on the encoded vectors. Both the encoder and the decoder are composed of a stack of N identical layers, with each consists of a multi-head self-attention network followed by a position-wise fully connected network. Formally, the procedure of learning sequence representations through Transformer can be formulated as follows:

$$\begin{bmatrix} \bar{\mathbf{H}}^l = \text{LN}(\text{SelfAttn}(\mathbf{H}^{l-1}) + \mathbf{H}^{l-1}) \\ \mathbf{H}^l = \text{LN}(\text{FFN}^l(\bar{\mathbf{H}}^l) + \bar{\mathbf{H}}^l) \end{bmatrix}_L \quad (3)$$

where SelfAttn(.) denotes the multi-head self-attention network which performs the attention function over $\mathbf{H}^{l-1}$, the output hidden states of the l-1st layer; FFN and LN stand for the position-wise fully connected layer and the layer normalization, respectively; $[...]_L$ denotes the stack of L layers. The output of the final layer $\mathbf{H}^L$ is returned as the representation of the input sentence.

3 Approach

The vanilla Transformer processes disperse words in a flat and uniform way, which makes it difficult to represent words in terms of their syntactic guidance (Li et al., 2020). Prior work has shown that decomposing the levels of granularity (phrases or templates) has produced substantial gains in paraphrase generation (Li et al., 2019), suggesting the possibility of further improvement from finer-grained modeling of granularity (Hao et al., 2019).

Motivated by the benefit of explicitly denoting word granularity, we propose GA-attention, a new self-attention block which automatically decomposes fine-grained granularity and learns granularity-aware sentence representations. We integrate GA-attention into the vanilla Transformer to generate granularity-aware paraphrases.

3.1 Granularity-Aware Self-Attention

Unlike DNPG, which classifies each word into either phrase or sentence levels (Li et al., 2019), C-DNPG aims to assign a soft classification of granularity for each word, yielding finer-grained decomposition of granularity for a sentence. For this purpose, we extend the vanilla self-attention with 1) a granularity head which estimates a continuous granularity level for each token, and 2) two new attention masks which bake the granularity into attentions for learning granularity-aware sentence representations. The overall architecture is illustrated in Figure 1a.

Granularity Head For a sequence of input tokens that is encoded as hidden states $\mathbf{H} = [\mathbf{h}_1, \ldots, \mathbf{h}_N]$, the granularity head estimates a continuous granularity vector $\mathbf{z} = [z_1, \ldots, z_N] \in [0, 1]^N$, where $z_i \in [0, 1]$ measures the extent of token i belonging to details: a z_i that is close to 1 indicates that the token at position i tends to be a detailed word (i.e., in the phrase level), while a z_i approaching 0 indicates that token i tends to be a template word (i.e., in the sentence level). Specifically in the self-attention networks, the granularity for hidden states in layer l can be estimated as:

$$\mathbf{z}^l = \text{sigmoid}(\mathbf{W}^G \mathbf{H}^{l-1}), l = 2, ..., L \quad (4)$$

where $\mathbf{W}^G$ represents the training parameters; $\mathbf{H}^{l-1}$ denotes the hidden states of layer l-1.

Having estimated the granularity for input tokens, we want to effectively incorporate the granularity into attentions to control the learning of sentence representations. To this end, we propose two new attention masks, namely, the granularity resonance mask and the granularity scope mask, Our idea is to adjust the original attention weights using the two proposed masks.

Granularity Resonance Mask We first introduce the *granularity resonance mask* where "resonance" is analogy to an assumption of token correlations: sentence-level tokens attend more to sentence-level tokens, whereas phrase-level tokens attend more to phrase-level tokens (Li et al., 2019). In this sense, the term *granularity resonance* refers to the correlation between two tokens in terms of their levels of granularity.

Let z_i and z_j denote the granularity of tokens i and j respectively. In the binary case where $z_i \in \{0, 1\}$, their correlation in terms of granularity can be formulated as:

$$\mathbf{C}_{ij} = \begin{cases} 1, & \text{if } z_i = z_j \\ 0, & \text{otherwise} \end{cases} \quad (5)$$

where $\mathbf{C}_{ij}$ represents the regularization coefficient to the original attention weight $\mathbf{A}_{ij}$ in terms of granularity correlation. Such a discrete measure of resonance is limited in modeling token correlation as both z_i and $\mathbf{C}_{ij}$ are binary variables. To improve the capacity of the granularity-aware attention, we

generalize the computation of $\mathbf{C}_{ij}$ (Equation 5) to a continuous function, that is,

$$\begin{aligned}\mathbf{C}_{ij} =&(1 - z_i) \times \max(0, 1 - (z_i + z_j)) \\ &+ z_i \times \min(1, 1 - z_i + z_j)\end{aligned} \tag{6}$$

where $z_i \in [0, 1]$ is a continuous value; z_i and 1-z_i control the extent of token i being in the word level or the sentence level, respectively. Equation 6 provides a smooth measure of the correlation between token i and j.

Granularity Scope Mask We further define the *granularity scope mask* where *granularity scope* measures the scope of attention according to the granularity level. This is based on the local attention assumption of phrases (Li et al., 2019): a phrase-level token tends to attend to surrounding tokens while a sentence-level token can attend to other tokens evenly with any distance. In that sense, phrasal tokens (with a large z_i) have a relatively smaller attention scope compared to sentence-level words (with a small z_i). For a given sequence of hidden states $\mathbf{h}_1, \ldots, \mathbf{h}_N$ with N words, the granularity scope for position i attending to position j can be defined as:

$$\mathbf{S}_{ij} = \begin{cases} 1 & \text{if } |i - j| < (N - \epsilon)^{(1-z_i)} + \epsilon \\ 0 & \text{otherwise} \end{cases} \tag{7}$$

where $\mathbf{S}_{ij} \in \{0, 1\}$ denotes the penalty for the original attention weight $\mathbf{A}_{ij}$ in terms of granularity scope. ϵ denotes the maximum distance that a phrasal word attends to. We set ϵ to 2 according to a similar configuration in (Li et al., 2019). This equation can be intuitively interpreted as the receptive fields for different levels of granularity (Li et al., 2019): a phrase-level token i (z_i=1) attends only to the adjacent n (n=3) words, whereas a sentence-level token i (z_i=0) can attend to positions with any distances.

Similarly, we generalize Equation 7 to a continuous function:

$$\mathbf{S}_{ij} = \max(0, \min(1, (N - \epsilon)^{(1-z_i)} + \epsilon - |i - j|)) \tag{8}$$

Using the two granularity-based attention masks, we derive the granularity-aware self-attention as an adjustment of the original attention weights, namely,

$$\begin{aligned}&\text{GASelfAttn}(\mathbf{H}) = \tilde{\mathbf{A}}^T \mathbf{V} \\ &\tilde{\mathbf{A}} = \mathbf{A} \odot \mathbf{C} \odot \mathbf{S}\end{aligned} \tag{9}$$

where $\mathbf{A}$ and $\tilde{\mathbf{A}}$ denote the original and adjusted attention weights respectively; $\odot$ stands for the element-wise multiplication; GASelfAttn(.) represents the proposed granularity-aware self-attention function.

3.2 C-DNPG: Transformer with Granularity Aware Attention

Based on the proposed GA-Attention, we propose C-DNPG, which integrates GA-attention into Transformer in order to better generate paraphrases at fine-grained levels of granularity. Figure 1b illustrates the overall architecture of our model. Compared to the vanilla Transformer, C-DNPG simply replaces the self-attention layers in both the encoder and the decoder with the proposed GA-attention network. Similar to the vanilla Transformer, we perform the GA-attention function for multiple heads in parallel and concatenate the multi-head representations to yield the final representation. The procedure for the C-DNPG (Transformer with GA-Attention) can be summarized as:

$$\left[\begin{aligned}\bar{\mathbf{H}}^l &= \text{LN}(\text{GASelfAttn}(\mathbf{H}^{l-1}) + \mathbf{H}^{l-1}) \\ \mathbf{H}^l &= \text{LN}(\text{FFN}^l(\bar{\mathbf{H}}^l) + \bar{\mathbf{H}}^l)\end{aligned}\right]_L \tag{10}$$

where LN denotes layer normalization (Ba et al., 2016).

4 Experiments

We evaluate our approach by experimenting on two widely used datasets, including the Quora question pairs and the Twitter URLs. We will introduce the common experimental setup and the empirical results.

Implementation Details We implemented our approach on top of the Huggingface PyTorch Transformer (Wolf et al., 2019). For a fair comparison, we followed the hyperparameter settings in related works (Li et al., 2019) for the Transformer. Both encoder and decoder consist of 3 transformer layers, have a hidden size of 450, and contain 9 attention heads (L=3, H=450, A=9). Following previous work (Li et al., 2019; Kazemnejad et al., 2020), we truncate sentences to 20 tokens. We utilize the pre-trained tokenizer by huggingface[1] (i.e., bert-base-uncased) for tokenization which has been common in NLP. During decoding, we employ beam search with a beam size of 8. All models were optimized with AdamW (Loshchilov and Hutter, 2018). The

[1] https://github.com/huggingface/tokenizers

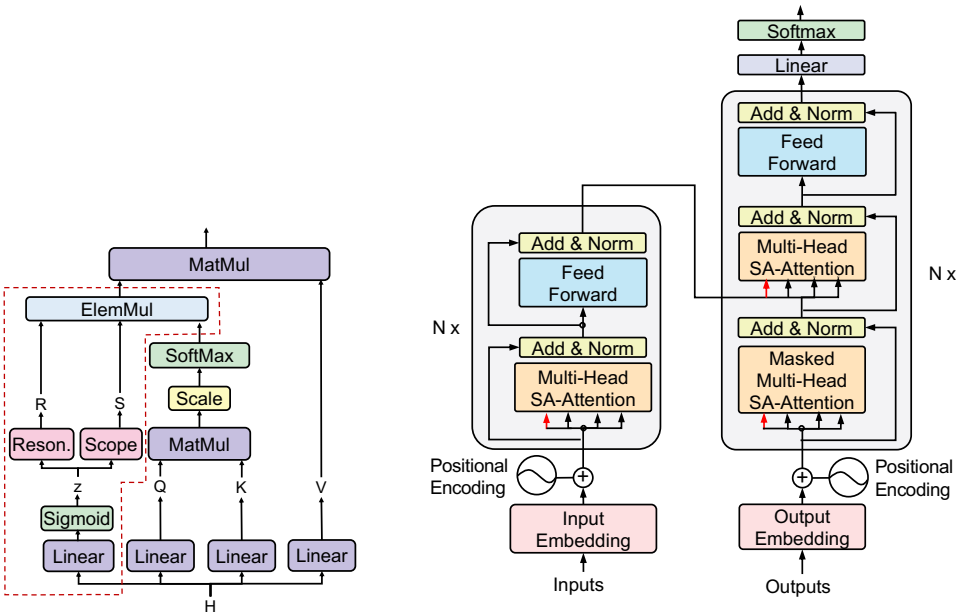

(a) Architecture of the GA-attention network. The red dotted area represents the extensions against the original attention network.

(b) Architecture of C-DNPG which seamlessly replaces the multi-head attention with our multi-head GA-attention. Red arrows denote additional z flows in GA-attention.

Figure 1: The architectures of GA-Attention and C-DNPG.

learning rate was varied under a linear schedule with warmup steps of 5,000 and the maximum learning rate of $5e^{-5}$. The model was training for 100,000 batches until achieving the best validation loss. The experiments were repeated for 5 times and were reported with their average results. All models were trained on a machine with NVIDIA Tesla V100 GPU allocated with a batch size of 32 samples.

Baseline Models We compare our approach with popular paraphrase generation methods including: (i) RedidualLSTM (Prakash et al., 2016): an LSTM sequence-to-sequence model using residuals between RNN layers; (ii) PointerGenerator (See et al., 2017): an RNN sequence-to-sequence model using copy mechanism; (iii) Transformer (Vaswani et al., 2017): the vanilla Transformer model; (iv) Transformer+Copy: an enhanced Transformer with copy mechanism (Gu et al., 2016); and (v) DNPG (Li et al., 2019): a popular paraphrase generation model based on Transformer. DNPG extends Transformer by generating paraphrases at multiple levels of granularity such as the phrase level and the sentence level. The model is composed of two encoder-decoder pairs, which correspond to phrase-level and sentence-level paraphrasing, respectively. We use the default settings of the baseline models as reported in their papers. (vi) FSET (Kazemnejad et al., 2020): the state-of-the-art paraphrase generation model that retrieves a paraphrase pair similar to the input sentence from a pre-defined index, then editing it using the extracted relations between the retrieved pair of sentences. We directly report the performance from their original paper.

Evaluation Metrics We perform automatic evaluation using five widely used metrics for text generation tasks, namely, BLEU (Papineni et al., 2002), iBLEU (Sun and Zhou, 2012), ROUGE-L (Lin, 2004) and METEOR (Lavie and Agarwal, 2007). We compute both BLEU-2 and BLEU-4 scores in our experiments using the NLTK package[2]. iBLEU (Sun and Zhou, 2012) penalizes BLEU by n-gram similarity between output and input. Hence, it is taken as the main metric for paraphrasing.

Datasets We conducted the experiments on two widely used benchmarks: 1) the Quora question pairs benchmark[3], which contains 124K duplicate question pairs. The dataset was labeled by human annotators and has been widely used for paraphrase research (Li et al., 2019; Devlin et al., 2019). We split the original data into train, validation, and test sets with proportions of 100K, 4K, and 20K, re-

[2]https://www.nltk.org/_modules/nltk/translate/bleu_score.html
[3]https://www.kaggle.com/c/quora-question-pairs (NC)

Model	Quora					Twitter URL				
	iBLEU	BLEU-2	BLEU-4	ROUGE-L	METEOR	iBLEU	BLEU-2	BLEU-4	ROUGE-L	METEOR
ResidualLSTM	20.45	40.71	26.20	36.19	32.67	20.29	36.75	25.92	32.47	29.44
Pointer-generator	22.65	43.82	28.80	42.36	40.87	25.60	44.50	32.40	38.48	36.48
Transformer	21.14	37.97	26.88	40.14	38.21	24.44	44.45	31.12	31.97	32.49
Transformer+Copy	22.90	44.42	28.94	37.60	38.34	27.07	48.44	34.35	38.37	38.19
DNPG	24.55	47.72	31.01	42.37	42.12	25.92	46.36	32.91	36.77	36.28
FSET	-	**51.03**	33.46	-	38.57	-	46.35	34.62	-	31.67
C-DNPG (R)	**26.94**	47.58	**34.05**	46.17	44.75	27.96	49.98	35.80	38.67	39.39
C-DNPG (S)	26.68	47.48	33.93	**46.22**	**46.66**	28.19	49.10	35.95	38.89	39.06
C-DNPG (R⊙S)	25.96	46.25	33.02	44.64	44.25	**30.25**	49.00	**38.58**	**41.60**	**41.71**
C-DNPG (R+S)	26.66	50.96	33.69	44.45	43.33	28.73	**50.49**	36.61	39.80	40.42

Table 2: Results of paraphrase generation on two benchmarks. R stands for the resonance mask while S stands for the scope mask; R×S stands for the combination of two masks through element-wise multiplication; R+S means we average the two masks for adjusting the original attention weights. We note that the results of baseline models are stronger than those reported in the DNPG paper, probably due to the BERT tokenizer we have utilized in our experiments. The pointer-generator outperforms the vanilla Transformer, as is consistent to the DNPG paper.

spectively.

2) the Twitter URL paraphrasing dataset[4] is also a widely used benchmark for evaluating paraphrase generation (Li et al., 2018; Kazemnejad et al., 2020). The dataset contains two subsets which are manually and automatically labeled, respectively. Following (Li et al., 2018), we sample 110k instances from the automatically labeled subset as our training set and sample 5k and 1k instances from the manually annotated subset for the test and validation sets, respectively.

5 Results and Analysis

5.1 Automatic Evaluation

Table 2 shows the results of various approaches on the two benchmarks. As the results indicate, C-DNPG (with variants) achieves the best performance in terms of most automatic metrics, which suggests that our C-DNPG is effective in performing multi-granularity paraphrasing.

In particular, our approach outperforms DNPG, a multi-granularity Transformer based model with a significant margin. That means that by modeling more fine-grained granularity levels, C-DNPG can control the generating process more precisely. Thanks to the granularity attention mechanisms, it is more flexible for the model to leverage syntactic guidance (e.g., recognizing templates and details) for paraphrase generation.

It is interesting to note that either the resonance mask or the scope mask that we propose can achieve the best performance under specific settings. We hypothesize that there could be overlap between the two proposed attention masks in some cases. Therefore, combining them may amplify the extent of masking and hinder the ultimate performance. We also find that Transformer with a copy mechanism can outperform DNPG on the Twitter dataset. This might be due to the more noise in this dataset, which leads copy based models to be more effective.

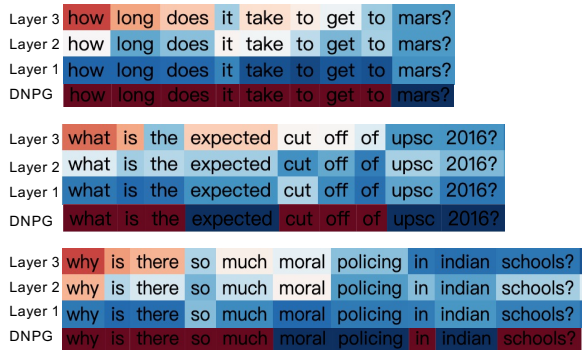

Figure 2: Examples of multi-granularity extracted by C-DNPG (Layer1-3) and DNPG (bottom) on the Quora dataset. Warmer colors represent higher levels of granularity (templates) while colder colors represent lower levels of granularity (details). We present granularity of all Transformer layers and compare the results with those of DNPG.

5.2 Qualitative Analysis

To gain a more in-depth insight into the performance, we qualitatively analyze the interpretability of GA-attention. We visualize the output of the granularity head in each attention layer to verify how effectively GA-attention captures fine-grained granularity. As shown in Figure 2, GA-attention can successfully capture continuous linguistic structures reflected as multiple levels of granularity (in

[4]https://github.com/lanwuwei/Twitter-URL-Corpus

Sentence:	What is a good first programming language?
Transformer:	What is good?
DNPG:	What is good for coding?
C-DNPG:	What are the best programming languages for beginners?
Human:	Whats a good and easy programming language to learn?
Sentence:	What will the year 2100 be like?
Transformer:	What is likely to happen in the world?
DNPG:	What are did today. year - year of unique year of country?
C-DNPG:	What will the world look like in 2100?
Human:	What will the year 2099 be like?

Table 3: Sample paraphrases from multiple models with human reference.

the last layer). For example, it successfully yields four levels of templates: 1) *what* __, 2) *what is* __, and 3) *what is the expected* __ 4) *what is the expected cut off of* __ according to Example 1 in Figure 2. In contrast, DNPG can decompose only two levels of granularity for each sentence. This means that C-DNPG can successfully distinguish templates and detailed words for each sentence, thus generating more fine-grained paraphrases.

Another interesting observation is that the continuous granularity is not extracted at once, instead, it is gradually summarized through transformer layers. This indicates that the proposed granularity-aware extensions blend naturally with attention networks in learning *coarse-to-fine* representations (Jawahar et al., 2019).

Overall, the results suggest that the proposed GA-attention naturally extends vanilla attention networks and enhances text representations with fine-grained granularity modeling.

5.3 Case Study

Table 3 presents two sample paraphrases generated by different models in the Quora test set. As the samples indicate, C-DNPG generates more coherent and fluent paraphrases than other models, which is consistent with the results of the automatic and human evaluation. According to the first sample, C-DNPG produces a more relevant and human-like paraphrase. For example, C-DNPG successfully paraphrases the word "first" as "*beginners*". The second sample shows more clear strength of C-DNPG which generates a paraphrase that is even better than the ground-truth question asked by human (e.g., the year "2100" is mistakenly paraphrased as "2099").

5.4 Human Evaluation

Besides the automatic evaluation, we also perform a human study to assess the performance of our approach qualitatively. We compare our approach with two typical methods, namely DNPG (Li et al., 2019) and the vanilla Transformer (Vaswani et al., 2017). They represent the state-of-the-art decomposition based method and the backbone model that our model is built upon, respectively. We randomly selected 200 Quora questions from the test set. For each one of the questions, one paraphrase was generated for each model. Then, three annotators from the Amazon Mechanical Turk were asked to compare the generated paraphrases by two models (ours vs. a baseline model) blindly based on two criteria, relevance and fluency. Relevance means that the generated paraphrases are semantically equivalent to the original question. Fluency means the generated paraphrases are natural and fluent sentences. Table 4 presents the comparison results. As can be seen, our model significantly outperforms the other two methods in terms of the two criteria. Moreover, the Fleiss' kappa κ shows fair agreement between annotators.

5.5 Computational Efficiency

As one of the key advantages of C-DNPG, we finally evaluate the time efficiency of our approach. We used the same setup as described in the Setup section. As Table 5 shows, the granularity aware attention mechanism in C-DNPG does not bring much additional computational cost to Transformers as opposed to the DNPG baseline approach, which indicates that GA-attention is lightweight to be integrated into Transformers.

Comparison	Relevance				Fluency			
	Win	**Tie**	**Loss**	Kappa	**Win**	**Tie**	**Loss**	Kappa
Ours *vs.* Transformer	67.8%	14.3%	17.8%	0.156	67.5%	15.8%	16.7%	0.166
Ours *vs.* DNPG	72.3%	12.8%	14.8%	0.171	68.1%	17.5%	14.3%	0.194
Ours *vs.* Human	43.7%	10.5%	45.8%	0.079	43.2%	11.0%	45.8%	0.095

Table 4: Human evaluation on the test set of Quora.

Model	Time (hours)
Transformer	1.2
TransformerCopy	1.2
DNPG	3.6
C-DNPG (ours)	1.5

Table 5: Training time (until model convergence) of various approaches on the Quora dataset.

6 Discussion

6.1 Why baking the two masks can enhance the performance?

Our idea is a generalization of the previous work DNPG (Li et al., 2019). In that paper, decomposing sentences into templates and details can improve the performance of paraphrasing because paraphrasing is usually generated by rewriting the sentence template while directly copying the detailed words. In that sense, template words pay more attention to template words while detailed words tend to pay attention to detailed words. Our paper generalizes this idea by extending the binary level (either 0 or 1) of granularity to a continuous range (between 0 and 1) of levels. This was implemented by the two proposed masks.

7 Related Work

This work is closely related to (1) multi-granularity paraphrase generation and (2) multi-granularity attention.

Multi-Granularity Paraphrase Generation.

There has been an increasing interest in decomposing paraphrase generation into multiple levels of granularity such as word, phrase and sentence (Li et al., 2019; Wiseman et al., 2018; Hao et al., 2019). For example, (Li et al., 2019) present the decomposable neural paraphrase generator (DNPG). DNPG is a Transformer-based model that generates paraphrases at two levels of granularity in a disentangled way. The model is composed of two encoder-decoder pairs, corresponding to phrase-level and sentence-level, respectively. The difference between our C-DNPG and DNPG is of three-fold: 1) C-DNPG estimates a continuous granularity level of each token and hence supports a continuous modeling of hierarchical structures; 2) Compared to DNPG, the GA-attention in C-DNPG can be naturally and seamlessly integrated into Transformer and is light-weighted in computation; and 3) While DNPG predicts granularity using a single fully connected layer, C-DNPG gradually summarizes granularity through a stack of Transformer layers.

Multi-Granularity Attention.

Another important line of work relates to multi-granularity self-attention. (Hao et al., 2019) proposed multi-granularity self-attention (MG-SA): MG-SA combines multi-head self-attention and phrase modeling by trains attention heads to attend to phrases in either n-gram or syntactic formalism. (Nguyen et al., 2020) proposed a tree-structured attention network which encodes parse tree structures into self-attention at constant time complexity. Despite similar names, our method differs from theirs greatly in principle and architecture. These two works rely on the existence of parsing trees, as opposed to GA-attention which infers latent structures from plain text. Furthermore, MG-SA only considers two-levels of granularity while GA-attention aims at continuous modeling of multiple granularity. (Liu et al., 2020) proposed a hybrid neural architecture named MahNN which integrates RNN and ConvNet, each learning a different aspect of semantic from the linguistic structures. Like other related works, MahNN is based on a coarse-grained attention mechanism. The two levels of granularity it processes are represented by RNN and ConvNet, respectively. By contrast, GA-attention provides a fine-grained attention function extended from the vanilla attention mechanism. Therefore, GA-attention is a pure attention-based approach and can be naturally and seamlessly integrated into

Transformers.

8 Conclusion

In this paper, we have proposed a novel paraphrase generation model named C-DNPG for continuously decomposing sentences at different levels of granularity. C-DNPG extends the multi-head attention with a granularity head which neurally estimates continuous granularity level of each input token. To efficiently incorporate granularity into attentions, we propose two novel attention masks, namely, granularity-resonance mask and granularity-scope mask, to adjust the original attention weights. Results on two paraphrase generation benchmarks show that C-DNPG remarkably outperforms baseline models in both quantitative and qualitative studies. In future work, we will investigate the effect of C-DNPG in pre-trained models and other NLP tasks.

Acknowledgments

We would like to thank the anonymous reviewers for their constructive feedback. Xiaodong Gu was sponsored by NSFC No. 62102244, CCF-Tencent Open Research Fund (RAGR20220129), and CCF-Baidu Open Fund (NO.2021PP15002000).

References

Jimmy Lei Ba, Jamie Ryan Kiros, and Geoffrey E Hinton. 2016. Layer normalization. *arXiv preprint arXiv:1607.06450*.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, Minneapolis, Minnesota. Association for Computational Linguistics.

Tanya Goyal and Greg Durrett. 2020. Neural syntactic preordering for controlled paraphrase generation. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 238–252.

Jiatao Gu, Zhengdong Lu, Hang Li, and Victor OK Li. 2016. Incorporating copying mechanism in sequence-to-sequence learning. In *ACL*, pages 1631–1640.

Zilu Guo, Zhongqiang Huang, Kenny Q. Zhu, Guandan Chen, Kaibo Zhang, Boxing Chen, and Fei Huang. 2021. Automatically paraphrasing via sentence reconstruction and round-trip translation. In *Proceedings of the Thirtieth International Joint Conference on Artificial Intelligence, IJCAI-21*, pages 3815–3821. International Joint Conferences on Artificial Intelligence Organization.

Jie Hao, Xing Wang, Shuming Shi, Jinfeng Zhang, and Zhaopeng Tu. 2019. Multi-granularity self-attention for neural machine translation. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 887–897.

Tom Hosking, Hao Tang, and Mirella Lapata. 2022. Hierarchical sketch induction for paraphrase generation. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (ACL 2022)*.

Ganesh Jawahar, Benoît Sagot, and Djamé Seddah. 2019. What does BERT learn about the structure of language? In *ACL 2019-57th Annual Meeting of the Association for Computational Linguistics*.

Amirhossein Kazemnejad, Mohammadreza Salehi, and Mahdieh Soleymani Baghshah. 2020. Paraphrase generation by learning how to edit from samples. In *proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 6010–6021.

Alon Lavie and Abhaya Agarwal. 2007. Meteor: An automatic metric for mt evaluation with high levels of correlation with human judgments. In *Proceedings of the Second Workshop on Statistical Machine Translation*, StatMT '07, page 228–231, USA. Association for Computational Linguistics.

Yinghao Li, Rui Feng, Isaac Rehg, and Chao Zhang. 2020. Transformer-based neural text generation with syntactic guidance. *arXiv preprint arXiv:2010.01737*.

Zichao Li, Xin Jiang, Lifeng Shang, and Hang Li. 2018. Paraphrase generation with deep reinforcement learning. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 3865–3878, Brussels, Belgium. Association for Computational Linguistics.

Zichao Li, Xin Jiang, Lifeng Shang, and Qun Liu. 2019. Decomposable neural paraphrase generation. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 3403–3414.

Chin-Yew Lin. 2004. ROUGE: A package for automatic evaluation of summaries. In *Text Summarization Branches Out*, pages 74–81, Barcelona, Spain. Association for Computational Linguistics.

Zhenyu Liu, Chaohong Lu, Haiwei Huang, Shengfei Lyu, and Zhenchao Tao. 2020. Text classification based on multi-granularity attention hybrid neural network. *arXiv preprint arXiv:2008.05282*.

Ilya Loshchilov and Frank Hutter. 2018. Fixing weight decay regularization in adam.

Xuan-Phi Nguyen, Shafiq Joty, Steven Hoi, and Richard Socher. 2020. Tree-structured attention with hierarchical accumulation. In *International Conference on Learning Representations*.

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. BLEU: a method for automatic evaluation of machine translation. In *Proceedings of the 40th annual meeting on association for computational linguistics*, pages 311–318. Association for Computational Linguistics.

Aaditya Prakash, Sadid A Hasan, Kathy Lee, Vivek Datla, Ashequl Qadir, Joey Liu, and Oladimeji Farri. 2016. Neural paraphrase generation with stacked residual lstm networks. *arXiv preprint arXiv:1610.03098*.

Alessandro Raganato and Jörg Tiedemann. 2018. An analysis of encoder representations in transformer-based machine translation. In *Proceedings of the 2018 EMNLP Workshop BlackboxNLP: Analyzing and Interpreting Neural Networks for NLP*, pages 287–297.

Abigail See, Peter J Liu, and Christopher D Manning. 2017. Get to the point: Summarization with pointer-generator networks. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1073–1083.

Hong Sun and Ming Zhou. 2012. Joint learning of a dual smt system for paraphrase generation. In *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 38–42.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in neural information processing systems*, pages 5998–6008.

Sam Wiseman, Stuart Shieber, and Alexander Rush. 2018. Learning neural templates for text generation. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 3174–3187, Brussels, Belgium. Association for Computational Linguistics.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, R'emi Louf, Morgan Funtowicz, and Jamie Brew. 2019. Huggingface's transformers: State-of-the-art natural language processing. *ArXiv*, abs/1910.03771.

Paraphrase Generation as Unsupervised Machine Translation

Xiaofei Sun[1,2], **Yufei Tian**[3], **Yuxian Meng**[2], **Nanyun Peng**[3]
Fei Wu[1,5], **Jiwei Li**[1,2] **and Chun Fan**[4]
[1]Zhejiang University, [2]Shannon.AI, [3]University of California, Los Angeles
[4]Peng Cheng Laboratory, [4]National Biomedical Imaging Center, Peking University
[4] Computer Center, Peking University, [5]Shanghai AI Laboratory
[5]Shanghai Institute for Advanced Study of Zhejiang University
xiaofei_sun@zju.edu.cn, yufeit@ucla.edu, jiwei_li@shannonai.com

Abstract

In this paper, we propose a new paradigm for paraphrase generation by treating the task as unsupervised machine translation (UMT) based on the assumption that there must be pairs of sentences expressing the same meaning in a large-scale unlabeled monolingual corpus. The proposed paradigm first splits a large unlabeled corpus into multiple clusters, and trains multiple UMT models using pairs of these clusters. Then based on the paraphrase pairs produced by these UMT models, a unified surrogate model can be trained to serve as the final SEQ2SEQ model to generate paraphrases, which can be directly used for test in the unsupervised setup, or be finetuned on labeled datasets in the supervised setup. The proposed method offers merits over machine-translation-based paraphrase generation methods, as it avoids reliance on bilingual sentence pairs. It also allows human intervene with the model so that more diverse paraphrases can be generated using different filtering criteria. Extensive experiments on existing paraphrase dataset for both the supervised and unsupervised setups demonstrate the effectiveness the proposed paradigm.

1 Introduction

The goal of paraphrase generation (Prakash et al., 2016a; Cao et al., 2016; Ma et al., 2018; Wang et al., 2018) is to generate a sentence semantically identical to a given input sentence but with variations in lexicon or syntax. It has been applied to various downstream NLP tasks such as parsing (Berant and Liang, 2014), question answering (Dong et al., 2017), summarization (Barzilay, 2004) and machine translation (Callison-Burch et al., 2006).

Building a strong paraphrase generation system usually requires massive amounts of high-quality annotated paraphrase pairs, but existing labeled datasets (Lin et al., 2014; Fader et al., 2013; Lan et al., 2017) are either of small sizes or restricted in narrow domains. To avoid such a heavy reliance on labeled datasets, recent works have explored unsupervised methods (Li et al., 2018b; Fu et al., 2019; Siddique et al., 2020) to generate paraphrase without annotated training data, among which the back-translation based model is an archetype (Mallinson et al., 2017; Sokolov and Filimonov, 2020). It borrows the idea of back-translation (BT) in machine translation (Sennrich et al., 2016) where the model first translates a sentence s_1 into another sentence s_2 in a different language (e.g., En→Fr), and then translates s_2 back to s_1. In this way, the model is able to generate paraphrases by harnessing bilingual datasets, removing the need for label paraphrase data.

However, BT-based models for paraphrase generation have the following severe issues: firstly, BT-based systems rely on external resources, i.e., bilingual datasets, making them hard to be applied to languages whose bilingual datasets are hard to obtain. Secondly, translation errors, such as duplicate words (Holtzman et al., 2020), missing words (Luong et al., 2015) and polysemous words (Rios Gonzales et al., 2017), will accumulate during the forward and backward translations, resulting in inferior performances. Thirdly, machine translation models work like blackboxs, making it hard for humans to intervene with the model and control the generation process.

In this work, we propose to address these problems based on the assumption that there must be pairs of sentences expressing the same meaning in a large-scale unlabeled corpus. Inspired by unsupervised machine translation (UMT) models, which align semantic spaces of two languages using monolingual data, we propose a pipeline system to generate paraphrases following two stages: (1) splitting a large-scale monolingual corpus into multiple clusters/sub-datasets, on which UMT models are trained based on pairs of these sub-datasets; and (2) training a unified surrogate model based on the paraphrase pairs produced by the trained multiple

Proceedings of the 29th International Conference on Computational Linguistics, pages 6379–6391
October 12–17, 2022.

UMT models, where we can design filtering functions to remove the pairs with undesired properties. The unified surrogate model can be then directly used for paraphrase generation in the unsupervised setup, or be finetuned on labeled datasets in the supervised setup.

The proposed framework provides the following merits over existing BT-based methods: (1) it is purely based on a large-scale monolingual corpus, which removes the reliance on bilingual datasets; (2) the trained unified model is able to generate paraphrases end-to-end, which avoids the issue of error accumulation that exists in vanilla BT-based models; and (3) human interventions can take place in the filtering step, which gives finer-grained controls over the generated paraphrases.

We conduct extensive experiments on a wide range of paraphrase datasets to evaluate the effectiveness of the proposed framework, and we are able to observe performance boosts against strong baselines in both supervised and unsupervised setups.

2 Related Work

Paraphrase Generation Methods for paraphrase generation usually fall into two categories: supervised and unsupervised approaches. Supervised methods for paraphrase generation rely on annotated paraphrase pairs. Xu et al. (2018); Qian et al. (2019) employed distinct semantic style embeddings to generate diverse paraphrases, and Iyyer et al. (2018); Li et al. (2019); Chen et al. (2019); Goyal and Durrett (2020) proposed to use different syntactic structure templates. A line of work (Mallinson et al., 2017; Sokolov and Filimonov, 2020) formalized paraphrase generation as machine translation. Unsupervised paraphrase generation is primarily based on reinforcement learning (RL) generative models (Ranzato et al., 2015; Li et al., 2016b). RL optimizes certain criteria, e.g. BLEU, to reward paraphrases with higher quality (Li et al., 2018b; Siddique et al., 2020). Bowman et al. (2016); Yang et al. (2019) trained a variational auto-encoder (VAE) (Kingma and Welling, 2013) to generate paraphrases. Other unsupervised methods for paraphrase generation include VAE (VQ-VAE) (Roy and Grangier, 2019), latent bag-of-words alignment (Fu et al., 2019) and simulated annealing (Liu et al., 2019a). Adapting large-scale pretraining (Devlin et al., 2018; Radford et al., 2018; Liu et al., 2019b; Clark et al., 2020; Sun et al., 2021b) to paraphrase generation has been recently investigated (Witteveen and Andrews, 2019; Hegde and Patil, 2020; Niu et al., 2020; Meng et al., 2021) and has shown promising potentials to improve generation quality. Our work is distantly related to unsupervised text style transfer (Hu et al., 2017; Mueller et al., 2017; Shen et al., 2017; Li et al., 2018a; Fu et al., 2018), where the model alters a specific text attribute of an input sentence (such as sentiment) while preserving other text attributes.

Regarding soliciting large-scale paraphrase datasets, Bannard and Callison-Burch (2005) used statistical machine translation methods obtain paraphrases in parallel text, the technique of which is scaled up by Ganitkevitch et al. (2013) to produce the Paraphrase Database (PPDB). Wieting et al. (2017) translate the non-English side of parallel text to obtain paraphrase pairs. Wieting and Gimpel (2017) collected paraphrase dataset with million of pairs via machine translation. Hu et al. (2019a,b) produced paraphrases from a bilingual corpus based on the techniques of negative constraints and inference sampling.

Unsupervised Machine Translation Unsupervised Machine Translation(UMT) has been an active research direction in NLP (Ravi and Knight, 2011). Pioneering work for unsupervised neural machine translation used denoising auto-encoders and back-translation (Sennrich et al., 2016) to iteratively refine the generated translation. Artetxe et al. (2017) proposed to use a shared encoder to encode source input sentences from different languages. Lample et al. (2017) additionally used adversarial and cross-domain training objectives to better identify different language domains. Yang et al. (2018) relaxed the strategy of sharing the entire encoder in Artetxe et al. (2017) by building independent encoders to maintain unique characteristics of each language. Another line of work for UMT is to combine statistical machine translation (SMT) and NMT. Artetxe et al. (2018); Lample et al. (2018) built a phrase-level mapping table from the source language to the target language. Following works improved UMT by combining SMT and NMT in different ways, such as warming up an NMT model with a trained SMT model (Marie and Fujita, 2018; Artetxe et al., 2019) and using SMT as posterior regularization (Ren et al., 2019). Other works involve initializing the model using retrieved semantically similar sentence pairs (Wu et al., 2019; Ren et al., 2020; Sun et al., 2021a), using auxiliary par-

allel data (Li et al., 2020; Garcia et al., 2020) and pretraining on large-scale multi-lingual data (Lample and Conneau, 2019; Song et al., 2019; Liu et al., 2020; Zhu et al., 2020).

3 Background for Unsupervised Machine Translation

We use the unsupervised machine translation (UMT) framework proposed by Lample et al. (2017) as the backbone. We briefly go though the model structure in this section. Let C_{src} and C_{tgt} respectively denote the monolingual dataset for the source and target language, on which a translation model M is trained to to generate target sequences y based on source sequences x, $y = M(x)$. The model is first initialized by training in a word-by-word translation manner using a parallel dictionary. The initial parallel dictionary is thus a word being translated to itself. Next, the model is iteratively trained based on a denoising auto-encoding (DAE), back-training (BT) and adversarial learning (AL). DAE allows the model to reconstruct the translation from a noisy input sentence by dropping and swapping words in the original sentence. The training objective of DAE is given by:

$$\mathcal{L}^{l}_{\text{DAE}} = \mathbb{E}_{x\sim C_l, \hat{x}\sim d(e(N(x),l),l)}[\Delta(\hat{x}, x)] \quad (1)$$

where $l = src$ or $l = tgt$ specifies the language, $N(x)$ is a noisy version of x, e and d respectively means encoding and decoding, and Δ measures the difference between the two sequences, which is the cross-entropy loss in this case. BT encourages the model to reconstruct the input sentence x from $N(y)$, a corrupted version of the model's translation $y = M(x)$. The training objective is given by:

$$\mathcal{L}^{l_1\to l_2}_{\text{BT}} = \mathbb{E}_{x\sim C_{l_1}, \hat{x}\sim d(e(N(M(x)),l_2),l_1)}[\Delta(\hat{x}, x)] \quad (2)$$

AL uses a discriminator to distinguish the language from the encoded latent representations, and by doing so, the model is able to better map two languages into the same latent space. The discriminative training objective is given by:

$$\mathcal{L}^{l}_{\text{Dis}} = -\mathbb{E}_{(x,l)}[\log p(l|e(x,l))] \quad (3)$$

The encoder is trained to fool the discriminator so that the encoder and the discriminator perform together in an adversarial style (Goodfellow et al., 2014):

$$\mathcal{L}^{l_1\to l_2}_{\text{Adv}} = -\mathbb{E}_{(x_1,l_1)}[\log p(l_2|e(x_1,l_1))] \quad (4)$$

The final training objective is given by:

$$\begin{aligned}\mathcal{L} = &\lambda_1[\mathcal{L}^{l_1}_{\text{DAE}} + \mathcal{L}^{l_2}_{\text{DAE}}] + \lambda_2[\mathcal{L}^{l_1\to l_2}_{\text{BT}} + \mathcal{L}^{l_2\to l_1}_{\text{BT}}] \\ &+ \lambda_3[\mathcal{L}^{l_1\to l_2}_{\text{Adv}} + \mathcal{L}^{l_2\to l_1}_{\text{Adv}}]\end{aligned} \quad (5)$$

The discriminative loss $\mathcal{L}^{l}_{\text{Dis}}$ is alternatively optimized with $\mathcal{L}$ to train the discriminator. We follow Lample et al. (2017) to implement each of the UMT models. We used the transformer-large (Vaswani et al., 2017) as the backbone instead of LSTMs in Bahdanau et al. (2014).

4 Model

The core idea of the proposed strategy is to use two subdatasets from a large monolingual corpus C and train unsupervised NMT models based on the two subdatasets. The path towards this goal naturally constitutes two modules: (1) constructing two subdatasets C_{src} and C_{tgt} from C; and (2) training the UMT model based on C_{src} and C_{tgt}.

4.1 Dataset Split

The crucial part in the framework is how to build the two subdatasets, on which the unsupervised NMT model is trained. To this end, we propose to (1) first construct candidates $\{c_1, c_2, ..., c_K\}$ for C_{src} and C_{tgt} from C based on the clustering models; and (2) selecting C_{src} and C_{tgt}. Based on C_{src} and C_{tgt}, UMT models will be trained. We use two criteria for clustering, LDA (Blei et al., 2003) and K-means clustering. The number of clusters/topics K is set to 80.[1]

LDA Clustering For LDA, we use Gibbs sampling and iterate over the entire corpus 5 times in total. In the last round, a sentence is assigned to the topic/cluster which has the largest probability of generating it. In LDA, each cluster is characterized as a distribution over the vocabulary. The distance between two subset c_m, c_n is the Jensen–Shannon (JS) divergence between the two distributions over the vocabulary:

$$\begin{aligned}\text{Dis}(c_m, c_n) &= \text{KL}(c_m||c_n) + \text{KL}(c_n||c_m) \\ \text{KL}(c_m||c_n) &= -\sum_{v\in V} p(v|c_m)\log\frac{p(v|c_m)}{p(v|c_n)} \\ \text{KL}(c_n||c_m) &= -\sum_{v\in V} p(v|c_n)\log\frac{p(v|c_n)}{p(v|c_m)}\end{aligned} \quad (6)$$

Since topics clustered by LDA can be incoherent (e.g., the clustering of stop words), we ask humans

[1]Here we use "topic" and "cluster" interchangeably.

to examine the top words of the topics, and discard meaningless clusters.

K-means Clustering For K-means, we use the average of the top layer embeddings from BERT (Devlin et al., 2018) to represent the sentence. Let h_s denote the sentence representation for the sentence s. We run the hard K-means model on the corpus, where the distance between a sentence and the cluster center is the L_2 distance between the two vector representations.

The LDA and K-means methods described above focus more on the situation that centers of two clusters are far away, but not individual sentences belonging to different clusters are different. These two focuses are correlated, but not exactly the same. The JS divergence for LDA clusters and L_2 distance for K-means clusters will be updated after the post-processing stage. LDA and K-means algorithms are performed on part of the the CommonCrawl corpus containing 10 billion English tokens.

4.2 UMT Training on C_{src} and C_{tgt}

4.2.1 Multiple UMT Models

We can randomly pick one pair of subsets from $\{c_1, ..., c_K\}$ as C_{src} and C_{tgt}, on which a single UMT model will be trained. The problem with single UMT model is obvious: each subset in $\{c_1, c_2, ..., c_K\}$ potentially represents a specific domain. The UMT model trained on the single C_{src} can thus only be able to properly paraphrase sentences from the C_{src} domain. To cover the full domain, we propose to train K UMT models, denoted by $\{M_1, M_2, ..., M_K\}$, where K is the number of clusters. Each of the trained UMT models uses a different $c \in \{c_1, c_2, ..., c_K\}$ as C_{src}, paired with a randomly selected C_{tgt}.

To paraphrase sentence s, we need to find its corresponding paraphrase generation model $M \in \{M_1, M_2, ..., M_K\}$, which takes s as the input and outputs its paraphrase. We first select the $C_{src} \in \{c_1, c_2, ..., c_K\}$ that s belongs to. Next, we pick that model M trained using C_{src} as sources, and use M to generate the output.

For LDA, C_{src} is the topic that generates s with the largest probability:

$$C_{src} = \underset{c \in \{c_1, c_2, ..., c_K\}}{\arg\max} \; p(s|c) \qquad (7)$$

For the K-means model, C_{src} is the cluster whose center is closest to s:

$$C_{src} = \underset{c \in \{c_1, c_2, ..., c_K\}}{\arg\min} \; ||h_s - \mu_c||^2 \qquad (8)$$

where μ_c denotes the center of the cluster c.

We follow Lample et al. (2017) to implement each of the UMT models. We used the transformer (Vaswani et al., 2017) as the backbone instead of LSTMs in Bahdanau et al. (2014), where the number of encoder blocks, decoder blocks, the number of heads, d_{model} and d_{ff} are respectively set to 6, 6, 8, 512 and 2,048. For UMT models based on specific C_{src} and C_{tgt}, both the encoder and the decoder are trained using Adam (Kingma and Ba, 2014), with the learning rate set to 0.00025, β_1 set to 0.5. We evenly alternate between the encoder-decoder and the discriminator.

Unifying Ms into a Surrogate Model We need to maintain K different domain-specific UMT models, which is both memory costly and computationally intensive, especially for online services. We thus propose to unify different Ms into a single surrogate one. For each sentence s in a selected corpus, we first find the cluster C_{src} it belongs to using LDA or K-means described above, and then we use the model M trained on C_{src} to generate the paraphrase of s. In this way, we are able to collect massive amounts of pseudo-labeled paraphrase pairs by treating the original sentence s as the source and the produced paraphrase as the target. We collected a total number of 25 million pairs. Human interventions can happen in this stage, where we can design filtering functions to remove pairs with undesired properties. Here, human interventions involve (1) removing pairs with identical source and target; (2) removing targets two times longer than sources. 16 million pairs remain after filtering.

We train a SEQ2SEQ model (Sutskever et al., 2014; Vaswani et al., 2017) (referred to as *UMT-Multi*) on the remaining pseudo-labeled data, which is used as the ultimate paraphrase generation model. We use the Transformer-base (Vaswani et al., 2017) as the model backbone, where the number of encoder blocks, decoder blocks, the number of heads, d_{model} and d_{ff} are respectively set to 6, 6, 8, 512 and 2,048. We use Adam (Kingma and Ba, 2014) with learning rate of 1e-4, β_1 = 0.9, β_2 = 0.999, and a warmup step of 4K. Batch size is set to 256. This model can be directly used in the unsupervised learning setup. An overview of deriving the *UMT-Multi* model is shown in Figure 1. Up to now, *UMT-Multi* is purely based on unlabeled commoncrawl corpus.

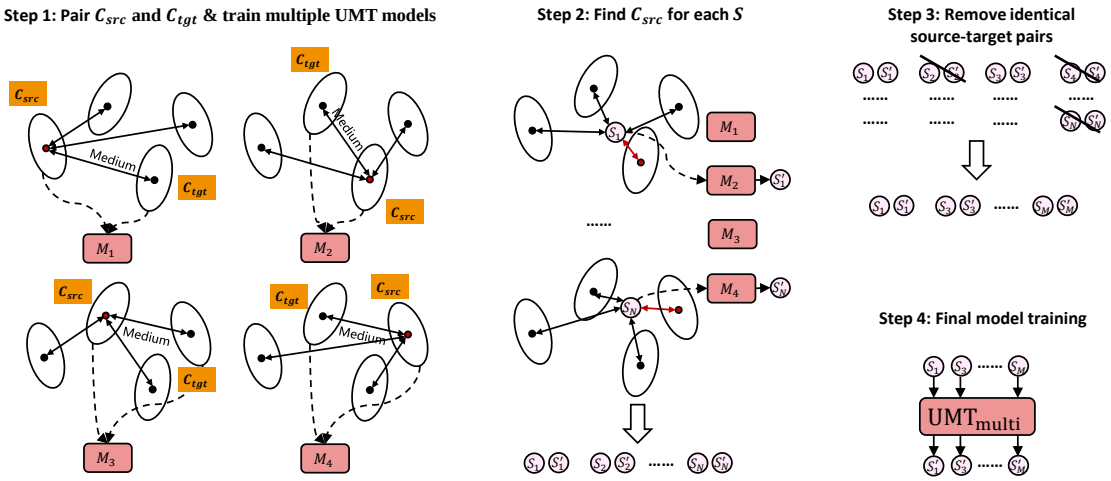

Figure 1: An overview of deriving the *UMT-Multi* model. Step 1: First, for each cluster c_i, we treat it as C_{src} and find its corresponding cluster C_{tgt}, and then train a UMT model on (C_{src}, C_{tgt}). The number of total UMT models is K (in this figure, $K = 4$). Step 2: For a given input sentence s, we first select the C_{src} that s belongs to, and use the model trained on C_{src} to generate the paraphrase of s. This process goes on over the entire corpus, leading to a pseudo labeled dataset of paraphrase pairs. Step 3: Human intervenes by removing paraphrase pairs whose inputs and output are the same, and outputs are two times longer than sources. Step 4: Training the single *UMT-Multi* model using the dataset after filtering. Step 5 (optional): fine-tuning the *UMT-Multi* model on the supervised paraphrase dataset in the supervised setup.

4.3 Supervised Setup

For the supervised setup, where we have pairs of paraphrases containing sources from a source domain and paraphrases of sources from a target domain, we can fine-tune the pretrained *UMT-Multi* model on the supervised paraphrase pairs, where we initialize the model using the *UMT-Multi* model, and run additional iterations on the supervised dataset. The fine-tuned model thus shares the structure with *UMT-Multi*. Again, we use Adam (Kingma and Ba, 2014) for fine-tuning, with $\beta_1 = 0.9$, $\beta_2 = 0.98$. Batch size, learning rate and the number of iterations are treated as hyper-parameters and tuned on the dev set. At test time, beam search (Sutskever et al., 2014; Li et al., 2016a) is used when decoding.

An additional use of the gold labeled paraphrase datasets is to help to select $C_{tgt} \in \{c_1, c_2, ..., c_K\}$ that best aligns with C_{src}, while in the unsupervised setup, we can only randomly pair C_{src} and C_{tgt} due to the lack of training signals for pairing. In the most straightforward setup, for each $C_{src} \in \{c_1, c_2, ..., c_K\}$, we can construct $K - 1$ pairs (C_{src}, c) by treating all $c \in \{c_1, c_2, ..., c_K\}, c \neq C_{src}$ as C_{tgt}. Next, we train $K - 1$ UMT models based on the pairs, and select the model that achieves the highest evaluation score on the labeled dataset. This strategy leads to a total number of $K \times (K - 1)$ models to be trained, which is computationally prohibitive. We propose a simplified learning model that maps the distance between C_{src} and C_{tgt} as inputs to output the evaluation score (here we use iBLEU) on the labeled dataset. Specifically, we randomly select L pairs, where $L \ll K \times (K - 1)$. We train L UMT models on the selected dataset pairs. Using the trained UMT models, we generate paraphrases for the labeled datasets, and obtain corresponding evaluation scores. Based on the distance between C_{src} and C_{tgt}, and the evaluation score $S(M_{(src,tgt)})$, we train a simple polynomial function F to learn to map the distance to the evaluation score:

$$S(M_{(src,tgt)}) = F(\text{Dis}(C_{src}, C_{tgt})) \quad (9)$$

The function F can be then used to select C_{tgt} with highest predicted evaluation score for C_{src}.

5 Experiments

5.1 Experiment Setups

We consider both the supervised and unsupervised setups. There are two differences between the supervised and unsupervised setups: for the supervised setup, (1) the training data provides guidance on pairing C_{src} and C_{tgt}; and (2) the pretrained model will be used as initialization and later fine-tuned on the labeled dataset. Datasets that we use for evaluation include Quora, WikiAnswers (Fader

et al., 2013), MSCOCO (Lin et al., 2014) and Twitter Liu et al. (2019a).

For the supervised setup, we compare our proposed model to the follow baselines: **ResidualLSTM** (Prakash et al., 2016b), **VAE-SVG-eq** (Gupta et al., 2018), **Pointer** (See et al., 2017), **Transformer** (Vaswani et al., 2017) and **DNPG** (Li et al., 2019). For the unsupervised setup, we use the following models for comparison: **VAE** (Bowman et al., 2016), **Lag VAE** (He et al., 2019), **CGMH** (Miao et al., 2019) and **UPSA** (Liu et al., 2019a). Results for VAE, Lag VAE, CGMH and UPSA on different datasets are copied from Miao et al. (2019) and Liu et al. (2019a). Results for ResidualLSTM, VAE-SVG-eq, Pointer, Transformer on various datasets are copied from Li et al. (2019). We leave details of these datasets, baselines and training in Appendix 7.

We are particularly interested in comparing the proposed model with bilingual MT based models. BT is trained end-to-end on WMT'14 En↔Fr.[2] A paraphrase pair is obtained by pairing the English sentence in the original dataset and the translation of the French sentence. Next we train a Transformer-large model on paraphrase pairs. The model is used as initialization to be further fine-tuned on the labeled dataset. We also use WMT-14 En-Zh for reference purposes. We use BLEU (Papineni et al., 2002), iBLEU (Sun and Zhou, 2012) and ROUGE scores (Lin, 2004) for evaluation.

5.2 Results

In-domain Results We first show the in-domain results in Table 1. We can observe that across all datasets and under both the supervised and unsupervised setups, the proposed UMT model significantly outperforms than baselines. As expected, multiple UMT models perform better than a single UMT model as the former is more flexible at selecting the correct domain C_{src} for an input sentence. We can also observe that the BT model is able to achieve competitive results, which shows that back-translation serves as a strong and simple baseline for paraphrase generation. The BT model trained on En-Fr consistently outperforms the one trained on En-Zh, and this is because that En-Zh translation is a harder task than En-Fr due to the greater grammars difference between the two languages.

[2]Wieting et al. (2017); Wieting and Gimpel (2017) suggested little difference among Czech, German, and French as source languages for back-translation. We use En↔Fr since it contains more parallel data than other language pairs.

	Model	iBLEU	BLEU	R1	R2
Supervised			*Quora*		
	ResidualLSTM	12.67	17.57	59.22	32.40
	VAE-SVG-eq	15.17	20.04	59.98	33.30
	Pointer	16.79	22.65	61.96	36.07
	Transformer	16.25	21.73	60.25	33.45
	DNPG	18.01	25.03	63.73	37.75
	BT(En-Fr)	18.04	25.34	63.82	37.92
	BT(En-Zh)	17.67	24.90	63.32	37.38
	UMT-Single	17.70	24.97	63.65	37.77
	UMT-Multi	**18.78**	**26.49**	**64.12**	**38.31**
			Wikianswers		
	ResidualLSTM	22.94	27.36	48.52	18.71
	VAE-SVG-eq	26.35	32.98	50.93	19.11
	Pointer	31.98	39.36	57.19	25.38
	Transformer	27.70	33.01	51.85	20.70
	DNPG	34.15	41.64	57.32	25.88
	BT(En-Fr)	34.55	41.90	57.84	26.44
	BT(En-Zh)	33.98	41.04	56.37	25.60
	UMT-Single	34.50	41.72	57.58	26.31
	UMT-Multi	**36.04**	**42.94**	**58.71**	**27.35**
Unsupervised			*Quora*		
	VAE	8.16	13.96	44.55	22.64
	Lag VAE	8.73	15.52	49.20	26.07
	CGMH	9.94	15.73	48.73	26.12
	UPSA	12.03	18.21	59.51	32.63
	BT(En-Fr)	11.98	17.84	59.03	32.11
	BT(En-Zh)	11.33	17.02	56.19	31.08
	UMT-Single	11.47	17.21	56.35	31.27
	UMT-Multi	**13.10**	**18.98**	**59.90**	**33.04**
			Wikianswers		
	VAE	17.92	24.13	31.87	12.08
	Lag VAE	18.38	25.08	35.65	13.21
	CGMH	20.05	26.45	43.31	16.53
	UPSA	24.84	32.39	54.12	21.45
	BT(En-Fr)	23.55	31.10	52.03	20.86
	BT(En-Zh)	22.60	30.12	51.29	20.11
	UMT-Single	23.01	30.62	51.79	20.35
	UMT-Multi	**25.90**	**33.80**	**54.52**	**23.48**
			MSCOCO		
	VAE	7.48	11.09	31.78	8.66
	Lag VAE	7.69	11.63	32.20	8.71
	CGMH	7.84	11.45	32.19	8.67
	UPSA	9.26	14.16	37.18	11.21
	BT(En-Fr)	8.15	13.78	36.30	10.48
	BT(En-Zh)	7.80	11.97	32.40	9.21
	UMT-Single	8.21	13.99	36.52	10.75
	UMT-Multi	**9.70**	**15.42**	**38.51**	**12.39**
			Twitter		
	VAE	2.92	3.46	15.13	3.40
	Lag VAE	3.15	3.74	17.20	3.79
	CGMH	4.18	5.32	19.96	5.44
	UPSA	4.93	6.87	28.34	8.53
	BT(En-Fr)	4.32	5.97	26.37	7.59
	BT(En-Zh)	4.15	5.40	25.83	7.32
	UMT-Single	4.40	6.11	26.89	7.78
	UMT-Multi	**5.35**	**7.80**	**29.74**	**9.88**

Table 1: In-domain performances of different models for both supervised and unsupervised setups.

Domain-adapted Results We test the model's domain adaptation ability on Quora and Wikianswers. Table 2 shows the results. We can see that UMT-multi performs significantly better than baselines, including UMT-single, showing the better

Model	iBLEU	BLEU	R1	R2
	Wikianswers→Quora			
Pointer	5.04	6.96	41.89	12.77
Transformer+Copy	6.17	8.15	44.89	14.79
DNPG	10.39	16.98	56.01	28.61
BT(En-Fr)	12.14	17.98	59.42	32.44
BT(En-Zh)	11.43	17.21	56.65	31.45
UMT-Single	11.86	17.49	57.01	32.44
UMT-Multi	**13.62**	**19.48**	**61.04**	**33.85**
	Quora→Wikianswers			
Pointer	21.87	27.94	53.99	20.85
Transformer+Copy	23.25	29.22	53.33	21.02
DNPG	25.60	35.12	56.17	23.65
BT(En-Fr)	25.77	35.30	56.41	23.78
BT(En-Zh)	24.84	34.19	55.71	22.60
UMT-Single	25.43	34.70	56.10	23.31
UMT-Multi	**26.85**	**36.64**	**57.45**	**24.60**

Table 2: Domain-adapted performances of different models. "R1" stands for ROUGE-1 and "R2" stands for ROUGE-2.

ability of UMT-multi for domain adaptation.

5.3 Human Evaluation

To further validate the performance of the proposed model, we randomly sample 500 sentences from Quora test set for human evaluation. The input sentence and its two paraphrases respectively generated by the UMT model and the BT model (En-Fr) are assigned to two human annotators at Amazon Mechanical Turk (AMT), with "> 95% HIT approval rate". Annotators are asked to judge which output is better in terms of three aspects: (1) semantics: whether the two sentences are the same semantic meaning; (2) diversity: whether the two sentences are diverse in expressions; and (3) fluency: whether the generated paraphrase is fluent. Ties are allowed. If the two annotators' evaluations do not agree with each other, the job will be assigned to one more annotator, and we take the majority as the final result.[3] Comparing with BT, the proportions of win, tie and lose for the proposed UMT-model are respectively 41%, 36%, and 22%, demonstrating its superiority over BT models.

5.4 Examples

Table 3 presents sampled paraphrases from the BT and UMT models. From these examples, we can identify several intrinsic drawbacks of the BT model that the UMT model can circumvent: (1) for the first example, the tense from the BT paraphrase model based on En-Zh translation is incorrect. This is because the Chinese language expresses tense in a more implicit way. This leads the model to make mistake in tense when Chinese is translated back to English. The UMT model does not have this issue; (2) for the second example, BT model directly copies the input, this is because the En-Fr can perfectly map the meaning in two languages with no expression variations. Due to the blackbox nature of MT models, it is hard to intervene with the process to avoid producing the same copy. Instead, for the proposed UMT framework, developers can intervene with the model in both clustering stage and data filtering stage. (3) For the third example, the BT model changes the meaning of the original sentence, which is due to the mistake made by the translation model. These mistakes are sometimes inevitable due to the limitation of current MT models, but can be fixed in the proposed system.

[3]If the three annotators all disagree, we discard the instance.

6 Ablation Study

In this section, we perform comprehensive ablation studies on Wikianswers dataset for understanding behaviors of the proposed model. And we report iBLEU score for comparison.

Size of C for UMT Training First, we explore how the size of C, the CommonCrawl corpus used for dataset split and UMT training, affects downstream performances. Table 4 shows the results, where the size is respectively 10M, 100M, 1B and 10B. We can observe that with more training data, the performance significantly improves. This is because the trained model can better learn to align sentences between different clusters.

The Number of LDA Topics Table 5 presents the influence of the number of LDA clusters. The trend is clear: more topics lead to better performances. This is because the model with more topics has a stronger ability of disentangling very similar sentences in the original corpus C, and thus avoids copying. It is worth noting that more topics means training more UMT models before unifying them, leading to greater computational intensity.

Pairing C_{src} and C_{tgt} In our main experiments, we randomly select C_{tgt} given C_{src}. It would be interesting to see the effects of different cluster selection strategies. We consider four strategies: *Largest* (select C_{tgt} with the largest distance to C_{src}), *Medium* (select C_{tgt} with the medium distance to C_{src}), *Smallest* (select C_{tgt} with the smallest distance to C_{src}) and for referring purposes,

Input	BT	UMT
Time is the most accurate test of how puberty is going to progress.	Time is the most accurate test of how puberty is going to progress. (En-Fr)	How adolescence develops is most accurately test by time.
GM told truck owners about the defect in the first half of October.	GM owners told truck about the defect in the first half of October . (En-Fr)	GM informed truck owners of the defect in the first half of October .
To keep him alive , well , there 's no reason to keep him alive .	To keep him alive , well , there 's no reason to keep him alive . (En-Fr)	To let him live , well , there 's no reason to let him live .
Washington did not have the formal education his elder brothers received at Appleby Grammar School in England, but he did learn mathematics, trigonometry, and land surveying .	Washington did not pursue the same formal education as his older brothers at England's Appleby Grammar School, but he did study geometry, trigonometry, and land surveying .	Unlike his older brothers, who studied at England's Appleby Grammar School, Washington did not receive formal education but studied mathematics, trigonometry, and land surveying .

Table 3: Sampled paraphrases from the BT and UMT models.

Size	10M	100M	1B	10B
Unsupervised. UMT-Multi	15.5	21.1	24.2	**25.9**

Table 4: The effect of data size of C for training UMT.

# LDA Topic	5	20	50	80
Unsupervised. UMT-Multi	14.9	22.4	24.9	**25.9**

Table 5: The effect of number of LDA topics.

Strategy	Unsuper. UMT-Multi	Super. UMT-Multi
Random	25.9	35.4
Largest	24.7	35.1
Medium	25.8	35.7
Smallest	25.3	35.5
Supervised	**26.3**	**36.0**

Table 6: The effect of different strategies to pair C_{src} and C_{tgt}.

	LDA		K-means	
Clustering	Single	Multi	Single	Multi
Uns.	23.0	**25.9**	21.9	24.2
Su.	34.5	**36.0**	32.1	34.2

Table 7: The effect of different clustering methods for C. "Uns." means we use the unsupervised setup and "Su." represents the supervised setup.

Supervised (select C_{tgt} using the supervised strategy proposed). In the *real* unsupervised setup, the supervised strategy cannot be readily applied since we have no access to supervised labels. We list performance for *supervised* here for referring purpose.

Table 6 shows the results. For both supervised and unsupervised setups, *Supervised* performs the best against the other strategies, especially under the unsupervised setup. The difference in performances between these strategies is greater for the unsupervised setup than the supervised setup. This is because supervised training serves to compensate the performance gap due to the presence of labeled training data. We find that the random strategy outperforms both *Largest* and *Smallest*. For *Largest* , this is because *Largest* leads to very different paired clusters, having the risk that some sentences in C_{src} might not have correspondences in C_{tgt}. For *Smallest*, since paired clusters are pretty close, sentences in C_{src} are more likely to have copies in C_{tgt}. *Largest* and *Smallest* leads to inferior performances. *random* performs comparable to *medium*.

Clustering Methods We study the effect of different clustering methods, i.e., LDA and K-means. Table 7 shows the results. As can be seen, for both supervised and unsupervised setups, the model trained with LDA consistently performs better than the model trained with K-means. We think there are potentially two reasons: (1) the BERT representations, on which clustering relies, cannot well represent sentence semantics for clustering (Reimers and Gurevych, 2019); and (2) the K-means model for sentence clustering operates at a relatively low level of semantics (i.e., sentence level), while LDA takes into the more global document level information. Due to the entanglement of sentence semantics in C, it is hard for K-means to separate sentences apart, or if it can, it takes long until convergence.

7 Conclusion

In this paper, we propose a new framework for paraphrase generation by treating the task as unsupervised machine translation (UMT). The proposed framework first splits a large unlabeled corpus into multiple sub-datasets and then trains one or multiple UMT models based on one or more pairs of these sub-datasets. Experiments and ablation studies under supervised and unsupervised setups demonstrate the effectiveness of the proposed framework.

Acknowledgement

We would like to thank anonymous reviewers for their comments and suggestions. This work is supported by the Key R & D Projects of the Ministry of Science and Technology (2020YFC0832500).

References

Mikel Artetxe, Gorka Labaka, and Eneko Agirre. 2018. Unsupervised statistical machine translation. *arXiv preprint arXiv:1809.01272*.

Mikel Artetxe, Gorka Labaka, and Eneko Agirre. 2019. An effective approach to unsupervised machine translation. *arXiv preprint arXiv:1902.01313*.

Mikel Artetxe, Gorka Labaka, Eneko Agirre, and Kyunghyun Cho. 2017. Unsupervised neural machine translation. *arXiv preprint arXiv:1710.11041*.

Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. 2014. Neural machine translation by jointly learning to align and translate.

Colin Bannard and Chris Callison-Burch. 2005. Paraphrasing with bilingual parallel corpora. In *Proceedings of the 43rd Annual Meeting of the Association for Computational Linguistics (ACL'05)*, pages 597–604.

Regina Barzilay. 2004. Information fusion for multidocument summarization: Paraphrasing and generation.

Jonathan Berant and Percy Liang. 2014. Semantic parsing via paraphrasing. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1415–1425.

David M Blei, Andrew Y Ng, and Michael I Jordan. 2003. Latent dirichlet allocation. *the Journal of machine Learning research*, 3:993–1022.

Samuel R. Bowman, Luke Vilnis, Oriol Vinyals, Andrew Dai, Rafal Jozefowicz, and Samy Bengio. 2016. Generating sentences from a continuous space. In *Proceedings of The 20th SIGNLL Conference on Computational Natural Language Learning*, pages 10–21, Berlin, Germany. Association for Computational Linguistics.

Chris Callison-Burch, Philipp Koehn, and Miles Osborne. 2006. Improved statistical machine translation using paraphrases. In *Proceedings of the Human Language Technology Conference of the NAACL, Main Conference*, pages 17–24.

Ziqiang Cao, Chuwei Luo, Wenjie Li, and Sujian Li. 2016. Joint copying and restricted generation for paraphrase.

Mingda Chen, Qingming Tang, Sam Wiseman, and Kevin Gimpel. 2019. Controllable paraphrase generation with a syntactic exemplar. *arXiv preprint arXiv:1906.00565*.

Kevin Clark, Minh-Thang Luong, Quoc V Le, and Christopher D Manning. 2020. Electra: Pre-training text encoders as discriminators rather than generators. *arXiv preprint arXiv:2003.10555*.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*.

Li Dong, Jonathan Mallinson, Siva Reddy, and Mirella Lapata. 2017. Learning to paraphrase for question answering. *arXiv preprint arXiv:1708.06022*.

Anthony Fader, Luke Zettlemoyer, and Oren Etzioni. 2013. Paraphrase-driven learning for open question answering. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1608–1618, Sofia, Bulgaria. Association for Computational Linguistics.

Yao Fu, Yansong Feng, and John P Cunningham. 2019. Paraphrase generation with latent bag of words. In *Advances in Neural Information Processing Systems*, pages 13645–13656.

Zhenxin Fu, Xiaoye Tan, Nanyun Peng, Dongyan Zhao, and Rui Yan. 2018. Style transfer in text: Exploration and evaluation. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 32.

Juri Ganitkevitch, Benjamin Van Durme, and Chris Callison-Burch. 2013. Ppdb: The paraphrase database. In *Proceedings of the 2013 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 758–764.

Xavier Garcia, Pierre Foret, Thibault Sellam, and Ankur P Parikh. 2020. A multilingual view of unsupervised machine translation. *arXiv preprint arXiv:2002.02955*.

Ian J Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. 2014. Generative adversarial networks. *arXiv preprint arXiv:1406.2661*.

Tanya Goyal and Greg Durrett. 2020. Neural syntactic preordering for controlled paraphrase generation. *arXiv preprint arXiv:2005.02013*.

Ankush Gupta, Arvind Agarwal, Prawaan Singh, and Piyush Rai. 2018. A deep generative framework for paraphrase generation. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 32.

Junxian He, Daniel Spokoyny, Graham Neubig, and Taylor Berg-Kirkpatrick. 2019. Lagging inference networks and posterior collapse in variational autoencoders. *arXiv preprint arXiv:1901.05534*.

Chaitra Hegde and Shrikumar Patil. 2020. Unsupervised paraphrase generation using pre-trained language models. *arXiv preprint arXiv:2006.05477*.

Ari Holtzman, Jan Buys, M. Forbes, and Yejin Choi. 2020. The curious case of neural text degeneration. *ArXiv*, abs/1904.09751.

J Edward Hu, Huda Khayrallah, Ryan Culkin, Patrick Xia, Tongfei Chen, Matt Post, and Benjamin Van Durme. 2019a. Improved lexically constrained decoding for translation and monolingual rewriting. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 839–850.

J Edward Hu, Rachel Rudinger, Matt Post, and Benjamin Van Durme. 2019b. Parabank: Monolingual bitext generation and sentential paraphrasing via lexically-constrained neural machine translation. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, pages 6521–6528.

Zhiting Hu, Zichao Yang, Xiaodan Liang, Ruslan Salakhutdinov, and Eric P Xing. 2017. Toward controlled generation of text. In *International Conference on Machine Learning*, pages 1587–1596. PMLR.

Mohit Iyyer, John Wieting, Kevin Gimpel, and Luke Zettlemoyer. 2018. Adversarial example generation with syntactically controlled paraphrase networks. *arXiv preprint arXiv:1804.06059*.

Diederik P Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*.

Diederik P Kingma and Max Welling. 2013. Auto-encoding variational bayes. *arXiv preprint arXiv:1312.6114*.

Guillaume Lample and Alexis Conneau. 2019. Cross-lingual language model pretraining. *arXiv preprint arXiv:1901.07291*.

Guillaume Lample, Alexis Conneau, Ludovic Denoyer, and Marc'Aurelio Ranzato. 2017. Unsupervised machine translation using monolingual corpora only. *arXiv preprint arXiv:1711.00043*.

Guillaume Lample, Myle Ott, Alexis Conneau, Ludovic Denoyer, and Marc'Aurelio Ranzato. 2018. Phrase-based & neural unsupervised machine translation. *arXiv preprint arXiv:1804.07755*.

Wuwei Lan, Siyu Qiu, Hua He, and Wei Xu. 2017. A continuously growing dataset of sentential paraphrases. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 1224–1234, Copenhagen, Denmark. Association for Computational Linguistics.

Jiwei Li, Will Monroe, and Dan Jurafsky. 2016a. A simple, fast diverse decoding algorithm for neural generation. *arXiv preprint arXiv:1611.08562*.

Jiwei Li, Will Monroe, Alan Ritter, Michel Galley, Jianfeng Gao, and Dan Jurafsky. 2016b. Deep reinforcement learning for dialogue generation. *arXiv preprint arXiv:1606.01541*.

Juncen Li, Robin Jia, He He, and Percy Liang. 2018a. Delete, retrieve, generate: A simple approach to sentiment and style transfer. *arXiv preprint arXiv:1804.06437*.

Zichao Li, Xin Jiang, Lifeng Shang, and Hang Li. 2018b. Paraphrase generation with deep reinforcement learning. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 3865–3878, Brussels, Belgium. Association for Computational Linguistics.

Zichao Li, Xin Jiang, Lifeng Shang, and Qun Liu. 2019. Decomposable neural paraphrase generation. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 3403–3414, Florence, Italy. Association for Computational Linguistics.

Zuchao Li, Hai Zhao, Rui Wang, Masao Utiyama, and Eiichiro Sumita. 2020. Reference language based unsupervised neural machine translation. *arXiv preprint arXiv:2004.02127*.

Chin-Yew Lin. 2004. ROUGE: A package for automatic evaluation of summaries. In *Text Summarization Branches Out*, pages 74–81, Barcelona, Spain. Association for Computational Linguistics.

Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C Lawrence Zitnick. 2014. Microsoft coco: Common objects in context. In *European conference on computer vision*, pages 740–755. Springer.

Xianggen Liu, Lili Mou, Fandong Meng, Hao Zhou, Jie Zhou, and Sen Song. 2019a. Unsupervised paraphrasing by simulated annealing. *arXiv preprint arXiv:1909.03588*.

Yinhan Liu, Jiatao Gu, Naman Goyal, Xian Li, Sergey Edunov, Marjan Ghazvininejad, Mike Lewis, and Luke Zettlemoyer. 2020. Multilingual denoising pre-training for neural machine translation. *arXiv preprint arXiv:2001.08210*.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019b. Roberta: A robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692*.

Minh-Thang Luong, Ilya Sutskever, Quoc V. Le, Oriol Vinyals, and Wojciech Zaremba. 2015. Addressing the rare word problem in neural machine translation.

Shuming Ma, Xu Sun, Wei Li, Sujian Li, Wenjie Li, and Xuancheng Ren. 2018. Query and output: Generating words by querying distributed word representations for paraphrase generation. In *Proceedings of*

the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers), pages 196–206, New Orleans, Louisiana. Association for Computational Linguistics.

Jonathan Mallinson, Rico Sennrich, and Mirella Lapata. 2017. Paraphrasing revisited with neural machine translation. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 1, Long Papers*, pages 881–893.

Benjamin Marie and Atsushi Fujita. 2018. Unsupervised neural machine translation initialized by unsupervised statistical machine translation. *arXiv preprint arXiv:1810.12703*.

Yuxian Meng, Xiang Ao, Qing He, Xiaofei Sun, Qinghong Han, Fei Wu, Jiwei Li, et al. 2021. Conrpg: Paraphrase generation using contexts as regularizer. *arXiv preprint arXiv:2109.00363*.

Ning Miao, Hao Zhou, Lili Mou, Rui Yan, and Lei Li. 2019. Cgmh: Constrained sentence generation by metropolis-hastings sampling. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, pages 6834–6842.

Jonas Mueller, David Gifford, and Tommi Jaakkola. 2017. Sequence to better sequence: continuous revision of combinatorial structures. In *International Conference on Machine Learning*, pages 2536–2544. PMLR.

Tong Niu, Semih Yavuz, Yingbo Zhou, Huan Wang, Nitish Shirish Keskar, and Caiming Xiong. 2020. Unsupervised paraphrase generation via dynamic blocking. *arXiv preprint arXiv:2010.12885*.

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. Bleu: a method for automatic evaluation of machine translation. In *Proceedings of the 40th annual meeting on association for computational linguistics*, pages 311–318. Association for Computational Linguistics.

Aaditya Prakash, Sadid A Hasan, Kathy Lee, Vivek Datla, Ashequl Qadir, Joey Liu, and Oladimeji Farri. 2016a. Neural paraphrase generation with stacked residual lstm networks. *arXiv preprint arXiv:1610.03098*.

Aaditya Prakash, Sadid A. Hasan, Kathy Lee, Vivek Datla, Ashequl Qadir, Joey Liu, and Oladimeji Farri. 2016b. Neural paraphrase generation with stacked residual LSTM networks. In *Proceedings of COLING 2016, the 26th International Conference on Computational Linguistics: Technical Papers*, pages 2923–2934, Osaka, Japan. The COLING 2016 Organizing Committee.

Lihua Qian, Lin Qiu, Weinan Zhang, Xin Jiang, and Yong Yu. 2019. Exploring diverse expressions for paraphrase generation. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 3164–3173.

Alec Radford, Karthik Narasimhan, Tim Salimans, and Ilya Sutskever. 2018. Improving language understanding by generative pre-training.

Marc'Aurelio Ranzato, Sumit Chopra, Michael Auli, and Wojciech Zaremba. 2015. Sequence level training with recurrent neural networks. *arXiv preprint arXiv:1511.06732*.

Sujith Ravi and Kevin Knight. 2011. Deciphering foreign language. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, pages 12–21, Portland, Oregon, USA. Association for Computational Linguistics.

Nils Reimers and Iryna Gurevych. 2019. Sentence-bert: Sentence embeddings using siamese bert-networks. *arXiv preprint arXiv:1908.10084*.

Shuo Ren, Yu Wu, Shujie Liu, Ming Zhou, and Shuai Ma. 2020. A retrieve-and-rewrite initialization method for unsupervised machine translation. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 3498–3504, Online. Association for Computational Linguistics.

Shuo Ren, Zhirui Zhang, Shujie Liu, Ming Zhou, and Shuai Ma. 2019. Unsupervised neural machine translation with smt as posterior regularization. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, pages 241–248.

Annette Rios Gonzales, Laura Mascarell, and Rico Sennrich. 2017. Improving word sense disambiguation in neural machine translation with sense embeddings. In *Proceedings of the Second Conference on Machine Translation*, pages 11–19, Copenhagen, Denmark. Association for Computational Linguistics.

Aurko Roy and David Grangier. 2019. Unsupervised paraphrasing without translation. *arXiv preprint arXiv:1905.12752*.

Abigail See, Peter J. Liu, and Christopher D. Manning. 2017. Get to the point: Summarization with pointer-generator networks. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1073–1083, Vancouver, Canada. Association for Computational Linguistics.

Rico Sennrich, Barry Haddow, and Alexandra Birch. 2016. Improving neural machine translation models with monolingual data. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 86–96, Berlin, Germany. Association for Computational Linguistics.

Tianxiao Shen, Tao Lei, Regina Barzilay, and Tommi Jaakkola. 2017. Style transfer from non-parallel text by cross-alignment.

A. B. Siddique, Samet Oymak, and Vagelis Hristidis. 2020. Unsupervised paraphrasing via deep reinforcement learning. *Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*.

Alex Sokolov and Denis Filimonov. 2020. Neural machine translation for paraphrase generation. *arXiv preprint arXiv:2006.14223*.

Kaitao Song, Xu Tan, Tao Qin, Jianfeng Lu, and Tie-Yan Liu. 2019. Mass: Masked sequence to sequence pre-training for language generation. *arXiv preprint arXiv:1905.02450*.

Hong Sun and Ming Zhou. 2012. Joint learning of a dual smt system for paraphrase generation. In *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 38–42.

Xiaofei Sun, Yuxian Meng, Xiang Ao, Fei Wu, Tianwei Zhang, Jiwei Li, and Chun Fan. 2021a. Sentence similarity based on contexts. *arXiv preprint arXiv:2105.07623*.

Zijun Sun, Xiaoya Li, Xiaofei Sun, Yuxian Meng, Xiang Ao, Qing He, Fei Wu, and Jiwei Li. 2021b. Chinesebert: Chinese pretraining enhanced by glyph and pinyin information. *arXiv preprint arXiv:2106.16038*.

Ilya Sutskever, Oriol Vinyals, and Quoc V Le. 2014. Sequence to sequence learning with neural networks. In *Advances in neural information processing systems*, pages 3104–3112.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in neural information processing systems*, pages 5998–6008.

Su Wang, Rahul Gupta, Nancy Chang, and Jason Baldridge. 2018. A task in a suit and a tie: paraphrase generation with semantic augmentation.

John Wieting and Kevin Gimpel. 2017. Paranmt-50m: Pushing the limits of paraphrastic sentence embeddings with millions of machine translations. *arXiv preprint arXiv:1711.05732*.

John Wieting, Jonathan Mallinson, and Kevin Gimpel. 2017. Learning paraphrastic sentence embeddings from back-translated bitext. *arXiv preprint arXiv:1706.01847*.

Sam Witteveen and Martin Andrews. 2019. Paraphrasing with large language models. *arXiv preprint arXiv:1911.09661*.

Jiawei Wu, Xin Wang, and William Yang Wang. 2019. Extract and edit: An alternative to back-translation for unsupervised neural machine translation. *arXiv preprint arXiv:1904.02331*.

Qiongkai Xu, Juyan Zhang, Lizhen Qu, Lexing Xie, and Richard Nock. 2018. D-page: Diverse paraphrase generation. *arXiv preprint arXiv:1808.04364*.

Qian Yang, Dinghan Shen, Yong Cheng, Wenlin Wang, Guoyin Wang, Lawrence Carin, et al. 2019. An end-to-end generative architecture for paraphrase generation. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 3123–3133.

Zhen Yang, Wei Chen, Feng Wang, and Bo Xu. 2018. Unsupervised neural machine translation with weight sharing. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 46–55, Melbourne, Australia. Association for Computational Linguistics.

Jinhua Zhu, Yingce Xia, Lijun Wu, Di He, Tao Qin, Wengang Zhou, Houqiang Li, and Tie-Yan Liu. 2020. Incorporating bert into neural machine translation. *arXiv preprint arXiv:2002.06823*.

A Datasets

(1) **Quora**[4]: The Quora dataset contains 140K parallel paraphrases of questions and 260K non-parallel sentences collected from the question answering website Quora[5]. We follow the standard setup in Miao et al. (2019) and use 3K/30K paraphrase pairs respectively for validation and test.
(2) **Wikianswers**: The WikiAnswers corpus (Fader et al., 2013) contains clusters of questions tagged by WikiAnswers users as paraphrases. It contains a total number of 2.3M paraphrase pairs. We follow Liu et al. (2019a) to randomly pick 5K pairs for validation and 20K for test.[6]
(3) **MSCOCO**: The MSCOCO dataset (Lin et al., 2014) contains over 500K paraphrase pairs for 120K image captions, with each image caption annotated by five annotators. We follow the dataset split and the evaluation protocol in Prakash et al. (2016b), where only image captions with fewer than 15 words are considered.
(4) **Twitter**: The Twitter dataset is collected via linked tweets through shared URLs (Lan et al., 2017), which originally contains 50K paraphrase pairs. We follow the data split in Liu et al. (2019a), where 10% of the training data is used as validation and the test set only contains sentence pairs that are labeled as "paraphrases".

B Baselines

For the supervised setup, we compare our proposed model to the follow baselines:
(1) **ResidualLSTM**: Prakash et al. (2016b) deepened the LSTM network by stacking multiple layers with residual connection. This deep SEQ2SEQ model is trained on labeled paraphrase datasets.
(2) **VAE-SVG-eq**: Gupta et al. (2018) combined VAEs with LSTMs to generate paraphrases in a SEQ2SEQ generative style.
(3) **Pointer**: See et al. (2017) augmented the standard SEQ2SEQ model by using a pointer, i.e., the copy mechanism. Word in the input sentence can be directly copied as the current decoded word.
(4) **Transformer**: Vaswani et al. (2017) proposed the Transformer architecture which is based on the self-attention mechanism.
(5) **DNPG**: Li et al. (2019) proposed a Transformer-based model that learns and generates paraphrases at different levels of granularity, i.e., from the lexical to phrasal and then to sentential levels.

For the unsupervised setup, we use the following models for comparison:
(1) **VAE**: Bowman et al. (2016) proposed variational auto-encoders (VAEs) to generate sentences from a continuous space. By minimizing the reconstruction loss between the input sentence and the output sentence, VAEs are able to sample paraphrases from the continuous space.
(2) **Lag VAE**: To overcome the posterior collapse issue of VAEs, He et al. (2019) proposed to aggressively optimize the inference network by performing multiple updates before reverting back to basic VAE training.
(3) **CGMH**: Miao et al. (2019) used Metropolis–Hastings sampling to generate paraphrases, where a word can be deleted, replaced or inserted into the current sentence based on the sampling distribution.
(4) **UPSA**: Liu et al. (2019a) proposed to use simulated annealing to optimize the paraphrase generation model. The training objective is composed of three parts: semantic similarity, expression diversity and language fluency.

[4] https://www.kaggle.com/c/quora-question-pairs
[5] https://www.quora.com/
[6] Note that the selected data is different from Liu et al. (2019a) but is comparable in the statistical sense.

Summarize, Outline, and Elaborate: Long-Text Generation via Hierarchical Supervision from Extractive Summaries

Xiaofei Sun[1,2], **Zijun Sun**[2], **Yuxian Meng**[2], **Jiwei Li**[1,2] **and Chun Fan**[3]
[1]Zhejiang University, [2]Shannon.AI, [3]Computer Center, Peking University
[3]National Biomedical Imaging Center, Peking University, [3]Peng Cheng Laboratory
xiaofei_sun@zju.edu.cn, jiwei_li@shannonai.com

Abstract

The difficulty of generating coherent long texts lies in the fact that existing models overwhelmingly focus on predicting local words, and cannot make high level plans on what to generate or capture the high-level discourse dependencies between chunks of texts. Inspired by human writing processes, where a list of bullet points or a catalog is first outlined, and then each bullet point is expanded to form the whole article, we propose *SOE*, a pipelined system that involves of summarizing, outlining and elaborating for long text generation: the model first outlines the summaries for different segments of long texts, and then elaborates on each bullet point to generate the corresponding segment. To avoid the labor-intensive process of summary soliciting, we propose the *reconstruction* strategy, which extracts segment summaries in an unsupervised manner by selecting its most informative part to reconstruct the segment.

The proposed generation system comes with the following merits: (1) the summary provides high-level guidance for text generation and avoids the local minimum of individual word predictions; (2) the high-level discourse dependencies are captured in the conditional dependencies between summaries and are preserved during the summary expansion process and (3) additionally, we are able to consider significantly more contexts by representing contexts as concise summaries. Extensive experiments demonstrate that SOE produces long texts with significantly better quality, along with faster convergence speed.

1 Introduction

Despite that recent large-scale pretrained language models (PLMs) (Devlin et al., 2018; Liu et al., 2019; Yang et al., 2019) are able to produce high-quality passages that can be hardly recognized by humans (Zellers et al., 2019), most of the generated "good" texts are within very limited length, e.g. hundreds of tokens for most cases (Yan et al., 2020), thus generating coherent long texts remains a challenge (Radford et al., 2019; Tan et al., 2020). The difficulty lies in the fact that existing models generate texts in a word-by-word manner: predicting each subsequent token given its preceding contexts. This word-by-word strategy overwhelmingly focuses on predicting local words, and cannot make high level plans on what to generate. The strategy results in the fact that long texts generated by current models are usually repetitive, generic and self-contradictory (Shen et al., 2019).

To address these issues, the coarse-to-fine generation strategy is proposed (Fan et al., 2018; Xu et al., 2018; Yao et al., 2019; Mao et al., 2019). In coarse-to-fine generation, a list of keywords or a short prompt is first generated, serving as a summary of the original text. The prompt is then fed to a seq2seq model as an input to output the complete text. The coarse-to-fine generation strategy significantly improves generation over the word-by-word strategy, but still suffers from the following shortcomings: (a) *limited capacity of the prompt*: a single keyword list or prompt does not have enough capacity to summarize all the text of long passages, since long texts are usually consists of several parts, each of which focuses on a specific aspect or topic (Zhou et al., 2018; Guan et al., 2019). The usage of the coarse-to-fine generation strategy is thus limited to texts that can be summarized by a single prompt (e.g., short stories). This explains why text length generated by the progressive generation model is still limited, e.g., the introduced writing prompts dataset in Fan et al. (2018) has an average length of stories around 735, and the average length of prompts is 28; (b) *ignorance of high-level discourse dependency*: the coarse-to-fine generation strategy does not capture discourse-level dependencies (Li and Jurafsky, 2016), which handle the high-level information flow and interactions between

Proceedings of the 29th International Conference on Computational Linguistics, pages 6392–6402
October 12–17, 2022.

segments of texts. The ignorance of discourse-level dependencies results in texts lacking for coherence.

Humans write in a hierarchical top-down manner: before writing a thousand-word-long essay, a human usually first prepares a list of bullet points or a catalogue, and then expands them to form the whole article. The sentence-level coherence between these bullet points is preserved when the bullet points are expanded, providing guarantees that the full text is coherent.

To mimic this top-down manner in human writing, we propose *SOE*, a pipelined system that involves of summarizing, outlining and expanding for long text generation: the model first outlines the summaries for different segments of long texts, which actually mimics the process of humans outlining bullet points; next, the model elaborates on each bullet point to generate the corresponding segment. The proposed strategy comes with the following merits: (a) Since each segment is associated with its own summary rather than the entire text sharing a single prompt, the capacity of summaries to reconstruct the full text can be guaranteed; (b) The conditional generation probability between summaries captures the high-level discourse dependencies, and these dependencies are preserved when they are expanded to segments. This naturally resolves the incapability of modeling discourse-level dependencies in the coarse-to-fine generation approach. (c) This model is able to consider significantly larger amount of contexts by representing chunks of contexts as concise summaries.

Empirically, we do not readily have summaries for segments in hand. The model thus needs to learn to summarize in an unsupervised manner. To this end, we propose the *reconstruction* strategy, which extracts segment summaries by selecting its most informative part to reconstruct the segment. Extensive experiments demonstrate that SOE produces long texts with significantly better quality than existing baselines.

2 Related Work

2.1 Generating Long Texts

There are two trends for generating long text: This first trend of work tackles the problem from the model perspective. New model structures (Kitaev et al., 2020; Child et al., 2019; Dai et al., 2019; Ye et al., 2019; Guo et al., 2019; Sukhbaatar et al., 2019; Correia et al., 2019; Beltagy et al., 2020; Zaheer et al., 2020) are designed to give the model the ability to congest more contexts given limited memories or computing power. For example, Transformer-XL (Dai et al., 2019), a modifier to Transformers (Vaswani et al., 2017) uses a segment-level recurrence mechanism to enable learning long-term dependencies; Child et al. (2019); Correia et al. (2019); Kitaev et al. (2020); Beltagy et al. (2020); Zaheer et al. (2020) proposed to sparsify transformers by focusing only on a fraction of attention connections; Tay et al. (2020) replaced the dot-product self-attention with learned synthetic attention weights.

The second trend of researches focus on developing new generation strategies. Efforts have been devoted to the idea of planning-then-generation or coarse-to-fine generation (Sha et al., 2017; Gehrmann et al., 2018a; Wiseman et al., 2019; Hua and Wang, 2019; Shen et al., 2020; Fu et al., 2020), which greatly inspires this work. In coarse-to-fine generation, a list of keywords or a short sentence is first generated, providing guidance to generate the full text. A recent work from Tan et al. (2020) takes a multi-step strategy, which progressively refines the generated incomplete text until reaching a specified stage. Similar ideas have also been applied to text summarization, where Gehrmann et al. (2018b) proposed a bottom-up method that first identifies phrases within a document that are likely included in its summary. Our work is also inspired by the strategy of hierarchical generation (Li et al., 2015b), which consider text units with bigger granularity: Li et al. (2015b) proposed hierarchical LSTMs that arrange tokens, sentences and paragraphs in a hierarchical structure, with different levels of LSTMs capturing compositionality, and Shen et al. (2019) used multi-level structures to learn a VAE model for generating long coherent text.

2.2 Extractive Summarization

Extractive summarization refers to the problem of selecting part of the input text as its summary. A fundamental problem in extractive summarization is to score constituent texts units (e.g., phrases, sentences or paragraphs) and select highly-ranked one(s) as the summary. Haghighi and Vanderwende (2009) used word frequencies in the input text to assign scores to words, which are then in turn used to score sentences. Higher-ranked sentences are

selected as the summary of the input text. Liu et al. (2018) presented a two-stage extractive-abstractive framework, which first coarsely identifies salient information, followed by a generation model used to refine it. Neural models have been widely used for scoring Cao et al. (2015); Ren et al. (2017). Liu and Lapata (2019) finetuned BERT (Devlin et al., 2018) to score each sentence for extractive summarization; Zhang et al. (2019) computed token similarity in each sentence using BERT contextual embeddings to serve as an automatic evaluation metric for text generation.

3 Background

Language Modeling refers to the process of calculating the probability $p(\mathbf{y})$ of a sequence $\mathbf{y} = [y_1, \cdots, y_T]$, where each y_i denotes a constituent token of $p(\mathbf{y})$. The probability can be computed by decomposing the joint distribution $p(\mathbf{y})$ into a product of conditional distributions over tokens:

$$p(\mathbf{y}) = \prod_{t=1}^{T} p(y_t|\mathbf{y}_{<t}) \quad (1)$$

where $\mathbf{y}_{<i} = [y_1, \cdots, y_{i-1}]$ is the partial sequence of tokens generated previously. During training, the model is optimized to minimize the negative log-likelihood (NLL) $-\sum_{\mathbf{y}\in\mathcal{D}} \log p(\mathbf{y})$. During inference, the model decodes a token at each time step t according to $p(y_t|\mathbf{y}_{<t})$ based on the softmax functions $y_t \propto \text{softmax}(\mathbf{W}^{\text{out}}\mathbf{h}_t)$ where $\mathbf{W}^{\text{out}} \in \mathbb{R}^{d\times|V|}$ is the output word embedding matrix and $\mathbf{h}_t$ is the hidden state at time-step t.

Sequence-to-Sequence (Seq2Seq) Generation models generate a target sequence $\mathbf{y}$ conditioning on a given source sequence $\mathbf{x}$, which differs from language models (LMs) in terms of whether or not conditioning on another input sequence. Similar to LMs, the probability of the target sequence can be typically factorized as:

$$p(\mathbf{y}|\mathbf{x}) = \prod_{i=1}^{T} p(y_t|\mathbf{y}_{<t}, \mathbf{x}) \quad (2)$$

Seq2seq models are also optimized to minimize the NLL $-\sum_{(\mathbf{x},\mathbf{y})\in\mathcal{D}} \log p(\mathbf{y}|\mathbf{x})$. In the rest of this paper, we unify the notation of $p(\mathbf{y})$ and $p(\mathbf{y}|\mathbf{x})$ by setting $\mathbf{x} = \varnothing$ for LMs. Different architectures have been proposed to model $p(y_t|\mathbf{y}_{<t}, \mathbf{x})$, including transformers (Vaswani et al., 2017), LSTMs (Luong et al., 2015) and CNNs (Dauphin et al., 2017). At test time, sequences are usually generated using beam search, or its variants to promote diversity (Vijayakumar et al., 2016).

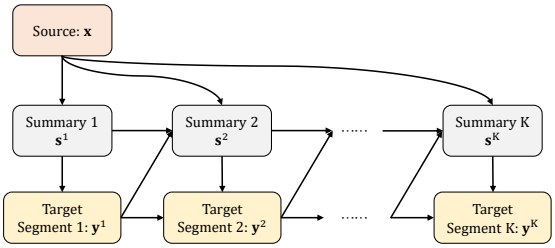

Figure 1: An overview of the proposed method. Given preceding tokens $\mathbf{y}^{<i}$, we first sequentially generate summaries $\mathbf{s}^{j\sim k}$ for each snippet. Next we expand each summary $\mathbf{s}$ to form the full text $\mathbf{y}^{j\sim k}$.

4 Model Details for SOE

4.1 Notations

A long sequence of tokens $\mathbf{Y} = \{\mathbf{y}^1, \mathbf{y}^2, \cdots, \mathbf{y}^K\}$ is first sliced into a series of snippets $\mathbf{y}^i$s, where K denotes the number of constituent snippets. Here we use the bold font $\mathbf{y}$ to denote snippets, and the normal font y to denote tokens. The number of tokens N within each snippet is a hyperparameter. We also use superscript i to denote the index of a snippet, and subscript l to denote the index of a token. Each $\mathbf{y}^i$ consists a sequence of tokens $\mathbf{y}^i = \{y^i_1, \cdots, y^i_{n_i}\}$, where n_i denotes the length of $\mathbf{y}^i$. Our goal is to generate a subset of $\mathbf{Y}$, denoted by $\mathbf{y}^{j\sim k} = \{\mathbf{y}^j, \mathbf{y}^{j+1}, \cdots, \mathbf{y}^k\}$ given its preceding snippets, denoted by $p(\mathbf{y}^{j\sim k}|\mathbf{y}^{<j})$. Each snippet $\mathbf{y}^i$ is associated with a short summary $\mathbf{s}^i = \{s^i_1, s^i_2, \cdots, s^i_{m_i}\}$, where s^i_l denotes tokens and m_i is the number of tokens in $\mathbf{s}^i$.

4.2 Pipeline Overview

Instead of generating all constituent words in $\mathbf{Y}$ one by one, we adopt a hierarchical strategy. The process of generating $\mathbf{y}^{j\sim k}$ is decoupled into the following two stages.

Stage1: Outlining Segment Summaries : we sequentially generate the summary $\mathbf{s}^i$ for each snippet conditioning on the summaries for previous snippets. This mimics the process of catalog generation when humans write.

Stage2: Expanding Summaries to Texts: we expand each summary $\mathbf{s}^i$ to the full segment by sequentially generating its constituent words.

An overview of the proposed method is shown in

Figure 1.

4.3 Extracting Golden Summaries

At the training time, we need to learn to generate summaries. But this is not straightforward because the golden summary $\mathbf{s}^i$ for the snippet $\mathbf{y}^i$ is not readily at hand. Manually soliciting summaries like Fan et al. (2018) is both costly and slow. We thus propose to take the idea of unsupervised extractive summarization, and for each snippet $\mathbf{y}^i$, we extract its summary $\mathbf{s}^i$ unsupervisedly, and use the extracted $\mathbf{s}^i$ as the golden summary for learning.

We investigate Random, TF-IDF (Ramos, 2003), TextRank (Mihalcea and Tarau, 2004) and Reconstruction methods to access the importance of selecting summary sentences, the first three of which are similar to Liu et al. (2018). More details of these three methods can be found at A.

Reconstruction A summary should be more informative than non-summary sentences, that is, a summary should have the most ability to reconstruct the full text. To measure the degree of a sentence's *reconstruction* ability, we use a seq2seq model to predict the original given text the summary sentence, the probability of which is regarded as the reconstruction score. Suppose there are n sentences in $\mathbf{y}^i$ (e.g., $n = 4$), and $\mathbf{y}^i = \{\mathbf{y}^{i,1}, \mathbf{y}^{i,2}, \mathbf{y}^{i,3}, \mathbf{y}^{i,4}\}$, and $\mathbf{y}^{i,j}$ denotes the j-th sentence in $\mathbf{y}^i$. The reconstruction score for $\mathbf{y}^{i,j}$, denoted by $\text{Score}(\mathbf{y}^{i,j})$ is given as follows:

$$\text{Score}(\mathbf{y}^{i,j}) = \frac{1}{|\mathbf{y}^i|} \log p(\mathbf{y}^i|\mathbf{y}^{i,j}) \quad (3)$$

To obtain $p(\mathbf{y}^i|\mathbf{y}^{i,j})$, we train another seq2seq model, where the input is $\mathbf{y}^{i,j}$ for each j, and the output is $\mathbf{y}^i$ by sequentially predicting tokens in $\mathbf{y}^i$. Given the trained model, we rank all sentences in $\mathbf{y}^i$ and use the one with the highest score as the golden summary $\mathbf{s}^i$.

4.4 Outlining Segment Summaries

In the summary generation stage, we cannot observe $\mathbf{y}^{j\sim k}$, and our goal is to sequentially generate $\mathbf{s}^{j\sim k}$ given $\mathbf{y}^{<j}$:

$$p(\mathbf{s}^{j\sim k}|\mathbf{y}^{<i}) = \prod_{i\in[j,k]} p(\mathbf{s}^i|\mathbf{y}^{<i}, \mathbf{s}^{<i}) \quad (4)$$

The generation of summary $\mathbf{s}^i$ can be factorized into sequentially generating the constituent word within it:

$$p(\mathbf{s}^i|\mathbf{y}^{<i}, \mathbf{s}^{<i}) = \prod_{l\in[1,m_i]} p(s^i_l|s^i_{<l}, \mathbf{y}^{<i}, \mathbf{s}^{<i}) \quad (5)$$

This process ends until generating a special end-of-sequence token <EOS> or reaching a specified summary length m. We use the Transformer-base(Vaswani et al., 2017) architecture as the backbone. For considering more contexts, we adopt the segment-level recurrence strategy, similar to Dai et al. (2019), where the hidden states computed for far away snippets are fixed and cached to be reused for the next new snippet. Gradients are not propagated to these far away snippets for memory and computation efficiency. This strategy allows the model to exploit information in history to the largest extent.

4.5 Expanding Summaries to Texts

Next, we expand each summary $\mathbf{s}^i$ to the full text for each segment by sequentially generating its constituent words

$$p(\mathbf{y}^i|\mathbf{y}^{<i}, \mathbf{s}^i) = \prod_{l\in[1,n_i]} p(y^i_l|\mathbf{y}^i_{<l}, \mathbf{s}^i, \mathbf{y}^{<i}) \quad (6)$$

which has the same termination conditions as in the summarization generation.

4.6 Training and Inference

Training For summary generation, the transformer model takes $[\mathbf{y}^{<i}; \mathbf{s}^{<i}]$ as the input and is optimized by minimizing the NLL loss $-\log p(\mathbf{s}^i|\mathbf{y}^{<i}; \mathbf{s}^{<i})$. Due to the memory limitation, we limit $\mathbf{y}^{<i}$ to preceding 384 tokens, and $\mathbf{s}^{<i}$ to 128 tokens at training. It is worth noting that the 384 tokens of $\mathbf{y}^{<i}$ mostly come from the segment right before, i.e., $\mathbf{y}^{i-1}$, while $\mathbf{s}^{<i}$ comes from multiple preceding segments since the summary is more concise.

For the summary expanding stage, the transformer model takes $[\mathbf{y}^{<i}; \mathbf{s}^i]$ as input and is optimized by minimizing the NLL loss $-\log p(\hat{\mathbf{y}}^i|\mathbf{y}^{<i}; \mathbf{s}^i)$. The two models, i.e., the summary generation and the summary expansion model share parameters, with a task-specific token appended to the start to notify the model on what to generate, summaries or segments.

Inference At test time, we first use beam search with beam size 5 to generate summaries. Given the generated summary, beam search is used again to

generate the corresponding segment. We consider more contexts at test time, where $\mathbf{y}^{<i}$ is limited to 1,156 tokens and $\mathbf{s}^{<i}$ is limited to 512 tokens.

Additionally, we augment the vanilla beam search with the strategy of mutual information reranking (Li et al., 2015a; Fang et al., 2015). The key point of mutual information is to, instead of merely handling the uni-directional dependency from the source to target based on the forward probability $\log p(\text{target}|\text{source})$, it models the mutual dependency between the source and target in sequence-to-sequence generation, i.e., the combination of the forward probability $\log p(\text{target}|\text{source})$ and the backward probability $\log p(\text{source}|\text{target})$. Specifically in our case, during summary generation, $\mathbf{s}^i$ is generated as follows:

$$\mathbf{s}^i = \arg\max_{\mathbf{s}^i}[\log p(\mathbf{s}^i|\mathbf{y}^{<i}, \mathbf{s}^{<i}) + \log p(\mathbf{s}^{i-1}|\mathbf{y}^{<i-1}, \mathbf{s}^i)] \quad (7)$$

where $p(\mathbf{s}^{i-1}|\mathbf{y}^{<i}, \mathbf{s}^i)$ is the backward probability of predicting the preceding summary $\mathbf{s}^{i-1}$ given $\mathbf{s}^i$. Since direct decoding from Eq.7 is infeasible, we follow the practical solution in Li et al. (2015a), where we first generate an N-best list based on the forward probability $p(\mathbf{s}^i|\mathbf{y}^{<i}, \mathbf{s}^{<i})$,[1] and then rerank the N-best list by combining the forward probability and the backward probability.

Similar strategy can also be applied to the summary expanding stage, where $\mathbf{y}^i$ is obtained as follows:

$$\mathbf{y}^i = \arg\max_{\mathbf{y}^i}[\log p(\mathbf{y}^i|\mathbf{y}^{<i}, \mathbf{s}^i) + \log p(\mathbf{y}^{i-1}|\mathbf{y}^i)] \quad (8)$$

The backward probability $p(\mathbf{y}^{i-1}|\mathbf{y}^i)$ predicts the preceding segment given the current segment. Again, beam search is combined with reranking to approximately find the optimal result.

4.7 Slicing Texts based on Coherence Scores

One more thing we need to care about is how to slice the text into segments. The simplest way is to slice the full text equally. But this is sub-optimal since the break point could be in the middle of two closely related sentences and one segment might contain multiple aspects.

We thus propose a slicing strategy based on sentence-level coherent scores. Using the Next Sentence Prediction (NSP) from BERT (Devlin et al., 2018), we are able to measure the coherence score between two consecutive sentences with index i and $i+1$, denoted by Score$(i, i+1)$. Given a full text $\mathbf{y} = \{y(1), y(2), ..., y(T)\}$, let T denote the number sentences in $\mathbf{y}$, and $y(i)$ denote the ith sentence. Given a fixed value K for the number sliced segments, $\mathbf{y}$ will be sliced into K segments, i.e., $\mathbf{y}^1, \mathbf{y}^2, ..., \mathbf{y}^K$, where each $\mathbf{y}^k$ consists of a group of consecutive sentences from $\mathbf{y}$. Let G_k denote the list of indexes of sentence in original $\mathbf{y}$, where $G_k[1]$ denotes the index of the first sentence in G_k, $G_k[2]$ denotes the second sentence, etc. Let $R_k = |G_k|$ denote the number of sentences in G_k.

We wish to maximize the coherence scores between two consecutive sentences within the same segment and minimize the score between two consecutive sentences belonging to different segments, giving the following objective to optimize:

$$L = \sum_{k=1}^{K} \sum_{i \in [1, R_k - 1]} \text{Score}(G(k)[i], G(k)[i+1]) - \sum_{k=1}^{K-1} \text{Score}(G[k][R_k], G[k][1]) \quad (9)$$

where Score$(G[k][R_k], G[k][1])$ the coherence score between the ending sentence of a segment and the starting sentence of the next segment. Given Score(i, j), Eq.9 can be readily solved using linear programming.

Model	*WikiText-103* PPL↓	# Para	*BookCorpus* PPL↓	# Para
	Base			
Vanilla	25.0	130M	29.0	130M
WritingPrompts-Keyword	23.8	135M	28.3	135M
WritingPrompts-Sentence	24.1	135M	28.6	135M
Progressive WritingPrompts	23.3	150M	27.7	150M
SOE	**22.2**	132M	**25.7**	132M
	Large			
Vanilla	20.0	220M	24.8	220M
SOE	**17.4**	224M	**22.5**	224M

Table 1: Perplexity of different models on WikiText-103 and BookCorpus. *Vanilla* stands for our implementation of Transformer-XL (Dai et al., 2019).

5 Experiments

In this section, we present experiment results. For different methods to generate summaries, we find that the performance of *Reconstruction* consistently outperforms the rest in our preliminary results. We

[1] We simplify $p(\mathbf{s}^{i-1}|\mathbf{y}^{<i-1}, \mathbf{s}^i)$ as $p(\mathbf{s}^{i-1}|\mathbf{s}^i)$, where we train a seq2seq model to predict the preceding summary given the current summary.

Model	*MSJ*↑ MSJ-2	MSJ-3	MSJ-4	*Diversity*↑ D-1	D-2	*Adversarial Success*↑ Adversarial Success	*S-Level Coherence*↑ NSP
				Base			
Vanilla	62.6	41.5	16.9	7.4	19.8	0.037	0.812
WritingPrompts-Keyword	63.0	42.2	17.5	8.9	22.0	0.057	0.836
WritingPrompts-Sentence	63.1	42.2	17.7	8.5	21.0	0.046	0.834
Progressive WritingPrompts	63.9	42.5	18.0	10.7	25.9	0.055	0.854
SOE	64.8	43.9	19.4	16.4	34.3	0.072	0.870
SOE+MI	**65.2**	**44.4**	**20.0**	**20.6**	**40.8**	**0.103**	**0.881**

Table 2: Results of different models in terms of diversity, adversarial success, MSJ and sentence-level coherence on the BookCorpus corpora. *Vanilla* stands for our implementation of Transformer-XL (Dai et al., 2019). "D-n" stands for "Distinct-$n(n = 1, 2)$", and *MI* stands the results for mutual information reranking.

Model	Distinct-1↑	Distinct-2↑
	Large	
Vanilla	11.7	25.5
SOE	24.1	45.0
SOE+MI	29.3	48.8

Table 3: Results of different models with large volumes in terms of diversity on the BookCorpus dataset.

thus only report results from *Reconstruction* in the section. We will get back to analysis on different summary generation methods in the ablation study section.

5.1 Datasets and Evaluation Metrics

We need a corpus of contiguous and long text to test SOE. We use two word-level datasets, WikiText-103 (Merity et al., 2016) and the BookCorpus dataset (Zhu et al., 2015). WikiText-103 contains 103M training words from 28K articles, with an average length of 3.6K words per article. WikiText-103 can be used to test the ability of modeling long-term dependencies. The BookCorpus dataset is a more suitable dataset for our purpose, with much longer and more contiguous texts. It contains a total number of roughly 1 billion words and 74 million sentences from 11k books, with an average length of 89K words for each book. The average number of words per sentence is 13. For both datasets, we predict the last 2,000 tokens at test time.

We use Perplexity (PPL), Diversity (Distinct-n) (Li et al., 2016), Adversarial Success (Kannan and Vinyals, 2017; Li et al., 2017), MS-Jaccard (MSJ) (Montahaei et al., 2019) and Sentence-Level Coherence (Tan et al., 2020) as evaluation metrics.

- **Perplexity (PPL)** Perplexity measures how fluent a piece of generated text could be (Dai et al., 2019). We use PPL as the basic evaluation metric in our experiments.
- **Diversity** Perplexity cannot measure how diverse the generated text is. We thus use the scaled number of distinct unigrams (Distinct-1) and bigrams (Distinct-2) to demonstrate the degree of diversity (Li et al., 2016) for generated texts.
- **Adversarial Success** Inspired by adversarial evaluations (Bowman et al., 2016; Kannan and Vinyals, 2017; Li et al., 2017), we use the adversarial success metric, which is defined as the fraction of a model successfully fooling a trained evaluator to believe that machine-generated texts are from humans. The evaluator is a binary classification model. At the training time, it takes as inputs machine-generated texts and original texts, and are trained to discriminate them. At test time, adversarial success is the value $1 - acc$, where acc denotes the accuracy of the trained evaluator predicting machine-generated texts as machine-generated. Higher values of adversarial success denotes better text quality.
- **MS-Jaccard (MSJ)** MSJ measures the similarity of the n-gram frequencies between the generated texts and the golden texts (Montahaei et al., 2019). We report MSJ-2, -3 and -4.
- **Sentence-Level Coherence** PPL, MSJ and diversity scores do not reflect the sentence-level coherence of generated texts. We adopt the strategy in Tan et al. (2020) where Next Sentence Prediction (NSP) from pretrained BERT model (Devlin et al., 2018) is used as a metric to measure the coherence between each sentence and its next sentence. We report average NSP scores for all consecutive sentence pairs within the generated text.

5.2 Baselines

In this paper, we use Transformer-XL (Dai et al., 2019), WritingPrompts (Fan et al., 2018), and Progressive WritingPrompts (Tan et al., 2020) as baselines. More details of the baseline models can be found at Appendix B.

For all models, we use Adam (Kingma and Ba, 2014) with learning rate of 1e-4, $\beta_1 = 0.9$, $\beta_2 = 0.999$, rate warmup over the first 10,000 steps, and linear decay of the learning rate. We use a dropout rate of 0.1 on all layers.

5.3 Results

Table 1 shows the results of perplexity for different models on the WikiText-103 and BookCorpus datasets. On both datasets, SOE achieves the lowest PPL compared to baselines Transformer-XL (Dai et al., 2019), WritingPrompts (Fan et al., 2018) and Progressive (Tan et al., 2020). In particular, for WikiText-103, we gain a PPL decrease -2.8, -1.6 and -1.1 against our implemented Transformer-XL, WritingPrompts-Sentence and Progressive, while having the same or even fewer parameters. Similar trend can be observed on BookCorpus.

Table 2 and Table 3 show the results for MSJ, diversity, adversarial success and sentence-level coherence scores. As can be seen, WritingPrompt based models generally outperform the Transformer-XL model, which adopts the word-by-word generation strategy. This validates the superiority of two-step generation strategy over the naive word-by-word generation strategy for long-text generation. The progressive WritingPrompt model, which involves multi-step of generation and expanding, outperforms the one-step the WritingPrompt-keyword and WritingPrompt-sentence model, which is in accord with our expectation. SOE achieves significantly better results compared to Vanilla, WritingPrompts and Progressive models in terms of all evaluation metrics, showing that the proposed method can produce more fluent, coherent and diverse texts. The consistent performance boosts on all metrics demonstrate the importance of modeling discourse-level dependencies and necessity of summary expanding strategy for long-text generation.

Additionally, we observe additional performance boosts by mutual information (MI), especially for diversity and adversarial success. This is in accord with our expectation: since mutual information is able to build bidirectional dependencies between the source and the target, models enhanced with mutual information can generate better summaries, and the phenomenon of generic and repetitive generation can be alleviated (Li et al., 2016), leading to more diverse results.

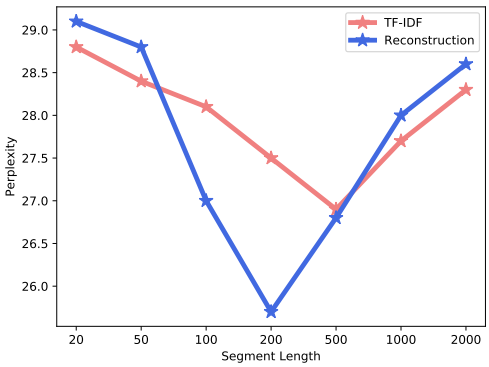

Figure 2: PPL on the BookCorpus dataset w.r.t. different segment lengths.

6 Ablation Studies

6.1 The Effect of Segment Length

The size of the segment can be neither too big nor too small: extremely long segments, might contain too many aspects or topics for the summary to summarize, in which case the model will degenerate into the WritingPrompts model (Fan et al., 2018). For too short segments, the summary cannot provide high-level guidance. We thus need to find the sweet spot for the segment length. Figure 2 shows results on the BookCorpus dataset. It is clear from the figure that too short segments and too long segments both lead to inferior performances.

6.2 The Effect of Summary Generation Strategies

It is worthwhile to explore how different summary extraction methods affect the final performances. To this end, we conduct experiments on the BookCorpus dataset, using different summary extraction methods, i.e., Random, TextRank, TF-IDF and Reconstruction. Table 4 shows the results. We first compare the ppl for summary generation, where the reconstruction model achieves the lowest ppl and thus produces summaries that are the easiest to predict given preceding contexts. It is also interesting to see that across all summary generation strategies, ppl for summarization generation is significantly larger than text prediction, which is reasonable since (1) generating summaries for the

Method	Summary PPL↓	Text PPL↓	MJ-4 ↑
Vanilla	-	29.0	16.9
Random	40.1	30.2	15.5
TextRank	30.7	26.2	17.8
TF-IDF	33.0	26.9	17.3
Reconstruction	**30.4**	**25.7**	**19.4**

Table 4: Performances of different summary extraction methods described in Section 4.3. *Vanilla* is the plain model that generates tokens one by one without summaries.

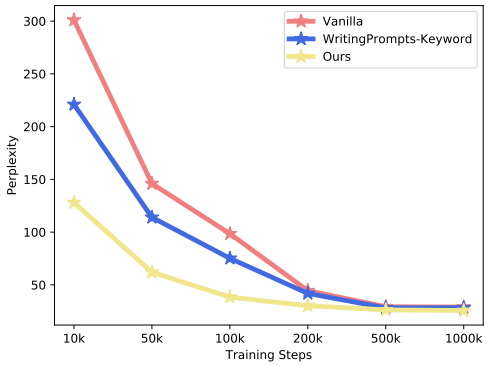

Figure 3: Convergence speed for different models.

upcoming segment requires more generalization abilities; and (2) there are more diverse options for what the next segment should talk about than the local choices for what the next sentence should talk about. For the final text-generation ppl, reconstruction achieves the best results, in terms of PPL, MJ-3 and MJ-4. TextRank and TF-IDF are better than Vanilla. Interestingly, the strategy of using random sentences as summaries performs worse than without summaries, which can be explained by providing no guidance is better than incorrect guidance.

6.3 The Effect of Coherence-based Text Slicing

We replace the coherence-based text slicing strategy with the naive equal slicing strategy, and see how this will negatively affect the performance. On the BookCorpus dataset, we observe an increase of summary generation ppl from 30.4 to 30.9, and an +0.7 increase of PPL from 25.7 to 26.4 in token generation, which demonstrates the importance of slicing text into coherent segments for generation. But it is also worth-noting that, even with the native equal slicing strategy, SOE still performs significantly better than other baseline models.

6.4 Decoupling The Effects of Summaries

The positive effects from summaries are two-fold: (1) it provides high-level guidance for segment generation; and (2) with far-away segments being concisely represented by summaries, it gives the model the ability to consider longer contexts. To quantitatively measure the influences from both aspects, we conduct the following experiments: at test time, for the computation $p(\mathbf{s}^i|\mathbf{y}^{<i};\mathbf{s}^{<i})$ and $p(\mathbf{y}^i|\mathbf{y}^{<i};\mathbf{s}^i)$, the model can only access summaries for segments that are used as contexts. In other words, only summaries within the 1,156 tokens of preceding contexts can be fed as inputs. This is different from the original version of SOE, in which $\mathbf{s}$ can extend to preceding contexts until the limit of 512 tokens is reached. We did not retrain the model, but add this limitation at test time. On the BookCorpus dataset, this leads to an increase of 0.8 in PPL (25.7 vs 26.5), and a decrease of 0.5 and 0.8 in MJ-3 (43.5 vs 43.9) and MJ-4 (18.6 vs 19.4).

6.5 Simplifying $p(\mathbf{s}^i|\mathbf{y}^{<i};\mathbf{s}^{<i})$

Here we explore different simplifications for $p(\mathbf{s}^i|\mathbf{y}^{<i};\mathbf{s}^{<i})$. For $p(\mathbf{s}^i|\mathbf{y}^{<i};\mathbf{s}^{<i})$, the current summary is generated based on both previous summaries and segment tokens. We can simplify it as $p(\mathbf{s}^i|\mathbf{s}^{<i})$, where previous segment tokens are not fed as inputs to predict the summary, which will significantly decreases computing complexity. On the BookCorpus dataset, we observe an increase of PPL in summary generation from 30.4 to 31.2, which subsequently leads to an +0.9 increase of PPL from 25.7 to 26.6 in token generation.

6.6 Convergence Speed

At last, we investigate how quickly different models converge. Results are shown in Figure 3. With the guidance of extracted summaries, SOE has a conspicuously faster convergence speed, where at about 200K training steps it has approximately reached the best result while the other two models — Vanilla and WritingPrompts — do not converge until 1000K training steps. The WritingPrompts model converges faster than then Vanilla because of the high-level guidance from prompts.

7 Conclusion

In this paper, we propose a two-step hierarchical generation strategy for long-text generation: the model first generates the summary for each segment conditioning on previous summaries, and next, each summary is expanded to form the full

text segment. The proposed strategy provides high-level guidance for local text generation, and enables high-level discourse dependencies to be captured. Extensive experiments demonstrate that SOE produces long texts with significantly better quality,

Acknowledgement

We would like to thank anonymous reviewers for their comments and suggestions. This work is supported by the Key R & D Projects of the Ministry of Science and Technology (2020YFC0832500), Program of Zhejiang Province Science and Technology (2022C01044).

References

Iz Beltagy, Matthew E. Peters, and Arman Cohan. 2020. Longformer: The long-document transformer.

Samuel R. Bowman, Luke Vilnis, Oriol Vinyals, Andrew M. Dai, Rafal Jozefowicz, and Samy Bengio. 2016. Generating sentences from a continuous space.

Ziqiang Cao, Furu Wei, Li Dong, Sujian Li, and Ming Zhou. 2015. Ranking with recursive neural networks and its application to multi-document summarization. In *AAAI*, pages 2153–2159. Citeseer.

Rewon Child, Scott Gray, Alec Radford, and Ilya Sutskever. 2019. Generating long sequences with sparse transformers. *arXiv preprint arXiv:1904.10509*.

Gonçalo M. Correia, Vlad Niculae, and André F. T. Martins. 2019. Adaptively sparse transformers. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 2174–2184, Hong Kong, China. Association for Computational Linguistics.

Zihang Dai, Zhilin Yang, Yiming Yang, Jaime Carbonell, Quoc Le, and Ruslan Salakhutdinov. 2019. Transformer-XL: Attentive language models beyond a fixed-length context. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 2978–2988, Florence, Italy. Association for Computational Linguistics.

Yann N Dauphin, Angela Fan, Michael Auli, and David Grangier. 2017. Language modeling with gated convolutional networks. In *International conference on machine learning*, pages 933–941.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*.

Angela Fan, Mike Lewis, and Yann Dauphin. 2018. Hierarchical neural story generation. *arXiv preprint arXiv:1805.04833*.

Hao Fang, Saurabh Gupta, Forrest Iandola, Rupesh K Srivastava, Li Deng, Piotr Dollár, Jianfeng Gao, Xiaodong He, Margaret Mitchell, John C Platt, et al. 2015. From captions to visual concepts and back. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1473–1482.

Yao Fu, Yansong Feng, and John P. Cunningham. 2020. Paraphrase generation with latent bag of words.

Sebastian Gehrmann, Falcon Dai, Henry Elder, and Alexander Rush. 2018a. End-to-end content and plan selection for data-to-text generation. In *Proceedings of the 11th International Conference on Natural Language Generation*, pages 46–56, Tilburg University, The Netherlands. Association for Computational Linguistics.

Sebastian Gehrmann, Yuntian Deng, and Alexander M Rush. 2018b. Bottom-up abstractive summarization. *arXiv preprint arXiv:1808.10792*.

Jian Guan, Yansen Wang, and Minlie Huang. 2019. Story ending generation with incremental encoding and commonsense knowledge. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, pages 6473–6480.

Qipeng Guo, Xipeng Qiu, Pengfei Liu, Yunfan Shao, Xiangyang Xue, and Zheng Zhang. 2019. Star-transformer. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 1315–1325, Minneapolis, Minnesota. Association for Computational Linguistics.

Aria Haghighi and Lucy Vanderwende. 2009. Exploring content models for multi-document summarization. In *Proceedings of Human Language Technologies: The 2009 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, pages 362–370.

Xinyu Hua and Lu Wang. 2019. Sentence-level content planning and style specification for neural text generation.

Anjuli Kannan and Oriol Vinyals. 2017. Adversarial evaluation of dialogue models. *arXiv preprint arXiv:1701.08198*.

Diederik P Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*.

Nikita Kitaev, Łukasz Kaiser, and Anselm Levskaya. 2020. Reformer: The efficient transformer. *arXiv preprint arXiv:2001.04451*.

Jiwei Li, Michel Galley, Chris Brockett, Jianfeng Gao, and Bill Dolan. 2015a. A diversity-promoting objective function for neural conversation models. *arXiv preprint arXiv:1510.03055*.

Jiwei Li, Michel Galley, Chris Brockett, Jianfeng Gao, and Bill Dolan. 2016. A diversity-promoting objective

function for neural conversation models. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 110–119, San Diego, California. Association for Computational Linguistics.

Jiwei Li and Dan Jurafsky. 2016. Neural net models for open-domain discourse coherence. *arXiv preprint arXiv:1606.01545*.

Jiwei Li, Minh-Thang Luong, and Dan Jurafsky. 2015b. A hierarchical neural autoencoder for paragraphs and documents. *arXiv preprint arXiv:1506.01057*.

Jiwei Li, Will Monroe, Tianlin Shi, Sébastien Jean, Alan Ritter, and Dan Jurafsky. 2017. Adversarial learning for neural dialogue generation. *arXiv preprint arXiv:1701.06547*.

Peter J Liu, Mohammad Saleh, Etienne Pot, Ben Goodrich, Ryan Sepassi, Lukasz Kaiser, and Noam Shazeer. 2018. Generating wikipedia by summarizing long sequences. *arXiv preprint arXiv:1801.10198*.

Yang Liu and Mirella Lapata. 2019. Text summarization with pretrained encoders.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Roberta: A robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692*.

Minh-Thang Luong, Hieu Pham, and Christopher D Manning. 2015. Effective approaches to attention-based neural machine translation. *arXiv preprint arXiv:1508.04025*.

Huanru Henry Mao, Bodhisattwa Prasad Majumder, Julian McAuley, and Garrison W Cottrell. 2019. Improving neural story generation by targeted common sense grounding. *arXiv preprint arXiv:1908.09451*.

Stephen Merity, Caiming Xiong, James Bradbury, and Richard Socher. 2016. Pointer sentinel mixture models. *arXiv preprint arXiv:1609.07843*.

Rada Mihalcea and Paul Tarau. 2004. Textrank: Bringing order into text. In *Proceedings of the 2004 conference on empirical methods in natural language processing*, pages 404–411.

Ehsan Montahaei, Danial Alihosseini, and Mahdieh Soleymani Baghshah. 2019. Jointly measuring diversity and quality in text generation models. *arXiv preprint arXiv:1904.03971*.

Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. 2019. Language models are unsupervised multitask learners. *OpenAI Blog*, 1(8).

J. Ramos. 2003. Using tf-idf to determine word relevance in document queries.

Pengjie Ren, Zhumin Chen, Zhaochun Ren, Furu Wei, Jun Ma, and Maarten de Rijke. 2017. Leveraging contextual sentence relations for extractive summarization using a neural attention model. In *Proceedings of the 40th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 95–104.

Lei Sha, Lili Mou, Tianyu Liu, Pascal Poupart, Sujian Li, Baobao Chang, and Zhifang Sui. 2017. Order-planning neural text generation from structured data. *arXiv preprint arXiv:1709.00155*.

Dinghan Shen, Asli Celikyilmaz, Yizhe Zhang, Liqun Chen, Xin Wang, Jianfeng Gao, and Lawrence Carin. 2019. Towards generating long and coherent text with multi-level latent variable models. *arXiv preprint arXiv:1902.00154*.

Tianxiao Shen, Victor Quach, Regina Barzilay, and Tommi Jaakkola. 2020. Blank language models. *arXiv preprint arXiv:2002.03079*.

Sainbayar Sukhbaatar, Edouard Grave, Piotr Bojanowski, and Armand Joulin. 2019. Adaptive attention span in transformers. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 331–335, Florence, Italy. Association for Computational Linguistics.

Bowen Tan, Zichao Yang, Maruan AI-Shedivat, Eric P. Xing, and Zhiting Hu. 2020. Progressive generation of long text.

Yi Tay, Dara Bahri, Donald Metzler, Da-Cheng Juan, Zhe Zhao, and Che Zheng. 2020. Synthesizer: Rethinking self-attention in transformer models. *arXiv preprint arXiv:2005.00743*.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Ł ukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, editors, *Advances in Neural Information Processing Systems 30*, pages 5998–6008. Curran Associates, Inc.

Ashwin K Vijayakumar, Michael Cogswell, Ramprasath R Selvaraju, Qing Sun, Stefan Lee, David Crandall, and Dhruv Batra. 2016. Diverse beam search: Decoding diverse solutions from neural sequence models. *arXiv preprint arXiv:1610.02424*.

Sam Wiseman, Stuart M. Shieber, and Alexander M. Rush. 2019. Learning neural templates for text generation.

Jingjing Xu, Xuancheng Ren, Yi Zhang, Qi Zeng, Xiaoyan Cai, and Xu Sun. 2018. A skeleton-based model for promoting coherence among sentences in narrative story generation. *arXiv preprint arXiv:1808.06945*.

Yu Yan, Weizhen Qi, Yeyun Gong, Dayiheng Liu, Nan Duan, Jiusheng Chen, Ruofei Zhang, and Ming Zhou. 2020. Prophetnet: Predicting future n-gram for sequence-to-sequence pre-training.

Zhilin Yang, Zihang Dai, Yiming Yang, Jaime Carbonell, Russ R Salakhutdinov, and Quoc V Le. 2019. Xlnet: Generalized autoregressive pretraining for lan-

guage understanding. In *Advances in neural information processing systems*, pages 5753–5763.

Lili Yao, Nanyun Peng, Ralph Weischedel, Kevin Knight, Dongyan Zhao, and Rui Yan. 2019. Plan-and-write: Towards better automatic storytelling.

Zihao Ye, Qipeng Guo, Quan Gan, Xipeng Qiu, and Zheng Zhang. 2019. Bp-transformer: Modelling long-range context via binary partitioning. *arXiv preprint arXiv:1911.04070*.

Manzil Zaheer, Guru Guruganesh, Avinava Dubey, Joshua Ainslie, Chris Alberti, Santiago Ontanon, Philip Pham, Anirudh Ravula, Qifan Wang, Li Yang, et al. 2020. Big bird: Transformers for longer sequences. *arXiv preprint arXiv:2007.14062*.

Rowan Zellers, Ari Holtzman, Hannah Rashkin, Yonatan Bisk, Ali Farhadi, Franziska Roesner, and Yejin Choi. 2019. Defending against neural fake news. In *Advances in Neural Information Processing Systems*, pages 9054–9065.

Tianyi Zhang, Varsha Kishore, Felix Wu, Kilian Q Weinberger, and Yoav Artzi. 2019. Bertscore: Evaluating text generation with bert. *arXiv preprint arXiv:1904.09675*.

Deyu Zhou, Linsen Guo, and Yulan He. 2018. Neural storyline extraction model for storyline generation from news articles. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 1727–1736, New Orleans, Louisiana. Association for Computational Linguistics.

Yukun Zhu, Ryan Kiros, Richard Zemel, Ruslan Salakhutdinov, Raquel Urtasun, Antonio Torralba, and Sanja Fidler. 2015. Aligning books and movies: Towards story-like visual explanations by watching movies and reading books.

A Methods for Selecting Summary Sentences

We investigate Random, TF-IDF (Ramos, 2003), TextRank (Mihalcea and Tarau, 2004) methods to access the importance of selecting summary sentences.

Random For comparing purposes, we use a random sentence as the summary.

TF-IDF We take the sentence with the highest average TF-IDF score (Ramos, 2003) as the golden summary $\mathbf{s}^i$. A word is assigned a score by TF-IDF that scales proportionally to the number of times the word appears in the document and is offset by the number of documents in the corpus that contain the word, which can be expressed as $N_w \cdot \log(\frac{N_d}{N_{dw}})$, where N_w is the word count, N_d is the total number of documents and N_{dw} is the total number of documents containing the word.

TextRank TextRank (Mihalcea and Tarau, 2004) is a weighted graph with tokens as nodes and the similarity between nodes as edges. We use BERT (Devlin et al., 2018) to compute the similarities between sentences and then rank them based on the TextRank algorithm.

B Model Baselines

In this paper, we use Transformer-XL, WritingPrompts, and Progressive WritingPrompts as baselines.

Transformer-XL Transformers with segment-level recurrence strategy (Dai et al., 2019) naturally constitutes a baseline. The model sequentially generates texts in a word-by-word fashion.

WritingPrompts first predicts a list of keywords or a single prompt, and then generates the full text given the prompt (Fan et al., 2018). Different from Fan et al. (2018), where golden prompts for stories are available, we do not readily have the golden prompts. We thus use the extractive strategies described in Section 4.3, i.e, the TF-IDF method to pick the keyword list as the prompt (denoted by *WritingPrompts-keyword*) and the reconstruction method to select the highest ranking sentence as the prompt (denoted by *WritingPrompts-sentence*).

Progressive WritingPrompts The progressive strategy proposed in Tan et al. (2020) which involves multiple stages of prompt generation. Each stage produces a more fine-grained sequence than the stage that comes before, and is used as the input to generate the prompt for the next stage. We follow the protocols in Tan et al. (2020) and use the TF-IDF score to obtain golden prompts for each stage. The number of stages is set to 4.

CoCGAN: Contrastive Learning for Adversarial Category Text Generation

Xin Sheng[1,3], **Linli Xu**[1,3]*, **Yinlong Xu**[2], **Changcun Bao**[4], **Huang Chen**[4], **Bo Ren**[4]
[1] Anhui Province Key Laboratory of Big Data Analysis and Application,
School of Computer Science and Technology, University of Science and Technology of China.
[2] School of Computer Science and Technology, University of Science and Technology of China.
[3] State Key Laboratory of Cognitive Intelligence. [4] Tencent Youtu Lab.
xins@mail.ustc.edu.cn, {linlixu,ylxu}@ustc.edu.cn,
{changcunbao,huaangchen,timren}@tencent.com

Abstract

The task of generating texts of different categories has attracted more and more attention in the area of natural language generation recently. Meanwhile, generative adversarial net (GAN) has demonstrated its effectiveness on text generation, and is further applied to category text generation in later works. Different from existing methods, which mainly consider the pairwise relations between the text embedding and the corresponding fixed one-hot class label (data-to-class relations), this paper proposes a novel Contrastive Category Generative Adversarial Net (CoCGAN) to incorporate contrastive learning into adversarial category text generation, considering more flexible data-to-class relations as well as relations between the multiple text embeddings in the same batch (data-to-data relations). The discriminator of CoCGAN discriminates the authenticity of given samples and optimizes a contrastive learning objective to capture both more flexible data-to-class relations and data-to-data relations among training samples. Accordingly, the generator tries to produce more realistic samples which can confuse the discriminator. Experimental results on both synthetic and real category text generation datasets demonstrate that CoCGAN can achieve significant improvements over the baseline category text generation models.

1 Introduction

Category text generation is the task of generating coherent and meaningful text with different categories and has received increasing attention in many natural language processing applications, such as sentiment analysis (Li et al., 2018) and dialogue generation (Li et al., 2017). It is a further expression of machine intelligence, and makes the generated texts more friendly to humans. Category text generation is a generalization of sentimental text generation and can be seen as a type of conditional text generation. This task focuses on how to integrate the auxiliary category information with the conventional text generation frameworks. Recently, generative adversarial net (GAN) (Goodfellow et al., 2014), where the discriminator is designed to guide the generator, is combined with the reinforcement learning (RL) (Williams, 1992) methods to generate discrete text sequences for general text generation with some remarkable successes (Yu et al., 2017; Guo et al., 2018; Caccia et al., 2020). Different from the general text generation task, category text generation mainly focuses on automatically generating a variety of controllable texts according to the specified categories. However, it is challenging to incorporate the category information into the sentences and design an appropriate objective for generating texts of different categories simultaneously.

In previous works, attempts have been made to extend the general text generation models to category text generation (Wang and Wan, 2018; Liu et al., 2020; Li et al., 2018). However, they only consider relations between the text embeddings and the class labels with simple constraints (data-to-class relations). Among them, SentiGAN (Wang and Wan, 2018) heavily relies on the discriminator based on a $(k+1)$-class classifier, which classifies between "generated" and k real classes. But it ignores the fact that each generated sample involves the degree of authenticity as well as the probability of belonging to a certain category simultaneously, therefore it is less reasonable to directly set the discriminator as a $(k+1)$-class classifier. To address this issue, CSGAN (Li et al., 2018) splits the discriminator into an authenticity discriminator and a category classifier, and optimize the generator with reward-based policy gradient strategy. Nevertheless, both the discriminators of SentiGAN and CSGAN are still limited by the simple category constraints (i.e., cross-entropy loss using fixed one-

*Corresponding author

Proceedings of the 29th International Conference on Computational Linguistics, pages 6403–6414
October 12–17, 2022.

hot class label as target). More recently, CatGAN proposes a category-aware model with a generator based on the relational memory core (RMC) to generate category texts. However, its discriminator still focuses on vanilla data-to-class relations and ignores relations between these text embeddings in the same batch (data-to-data relations).

In this paper, inspired by recent application of contrastive learning in conditional image generation (Kang and Park, 2020), we propose a novel adversarial category text generation framework, namely Contrastive Category Generative Adversarial Net (CoCGAN), to further exploit the category information. Different from SentiGAN, which uses multiple generators, we adopt a conditional generator (Hochreiter and Schmidhuber, 1997) to simplify the model, where an additional class label embedding is set as input to control the type of the generated category text. Following (Yu et al., 2017), we consider the sequence generation procedure as a sequential decision making process and the generator is regarded as a stochastic parametrized policy. Regarding the discriminator, since each generated sample is associated with a real class label and it is rather rough to simply mark it as "generated", the proposed discriminator is divided into two parts: an authenticity discriminator and a contrastive category projector. The authenticity discriminator is a conventional GAN discriminator which is designed as a binary classifier to judge whether the input text is real or not. As for the contrastive category projector, we abandon the conventional way which adopts the cross-entropy loss as the training objective and take class label embeddings into account for more flexible learning. Specifically, the discriminator leverages contrastive learning to pull the text embeddings closer to their corresponding class label embeddings. Furthermore, we also consider relations between the text embeddings which share the same class labels. In other words, the contrastive category projector aims to pull the multiple text embeddings, which are in the same batch, closer to each other when their class labels are the same, while pushing away from each other otherwise. As benefits of the novel contrastive learning paradigm, the discriminator can capture not only more flexible data-to-class relations but also data-to-data relations among training samples. During adversarial training, we adopt Monte Carlo tree search to approximate the state-action value function and the penalty-based (Wang and Wan, 2018) training objective is used to update the generator with policy gradient strategy (Sutton et al., 2000), where we integrate the output of the authenticity discriminator and the contrastive category projector to obtain the overall penalty.

We conduct category text generation experiments on both synthetic and real category datasets and adopt multiple metrics to evaluate the quality of the generated texts. We also compare the proposed CoCGAN with several state-of-the-art category text generation models, including SentiGAN and CatGAN. Experimental results on three datasets (i.e., movie reviews, amazon reviews and synthetic datasets) demonstrate that our model consistently outperforms the state-of-the-art models.

The contributions of this work are three-fold:

- We adopt the conditional generator without loss of generality and decouple the discriminator into two parts, which comprehensively consider the authenticity and category information of the input text.
- We propose the CoCGAN which adopts contrastive learning to leverage not only more flexible data-to-class relations but also data-to-data relations among samples for category text generation. To the best of our knowledge, this work is the first attempt to introduce contrastive learning for category text generation.
- Extensive experiments are conducted on several datasets and the results from multiple perspectives demonstrate the effectiveness of the proposed model.

2 Related Work

Text generation is an important task in natural language processing and has received more and more attention in many fields recently (Sheng et al., 2020; Bahdanau et al., 2014). Traditional text generation models based on recurrent neural network (RNN) (Graves, 2013) generate each token of a sentence conditioned on the previous tokens and the current hidden state. Nevertheless, the training paradigm maximum likelihood estimation (MLE) may suffer from exposure bias (Bengio et al., 2015), which is due to the inherent difference between the training stage and the inference stage of text generation models trained via MLE. Scheduled sampling is proposed by (Bengio et al., 2015) to solve it but soon proved to be inconsistent (Huszár, 2015). To

alleviate this issue, some works adopt generative adversarial net (GAN) (Goodfellow et al., 2014), which has achieved significant successes on computer vision (Radford et al., 2016; Brock et al., 2018). In GAN, the discriminator learns to distinguish whether a given sample is real or not, and the generator learns to confuse the discriminator by generating high quality data. Nevertheless, GAN is designed for differentiable data, which conflicts with the discrete nature of text generation.

To tackle the above non-differentiability problem, reinforcement learning is introduced in SeqGAN (Yu et al., 2017) and LeakGAN (Guo et al., 2018), where the discriminator guides the generator with the reward signal. And this training paradigm is widely adopted (Sheng et al., 2022). Alternatively, MaskGAN (Fedus et al., 2018) uses the actor-critic algorithm to fill in the missing text conditioned on the surrounding context. RankGAN (Lin et al., 2017) replaces the conventional binary classifier with a ranking model as the discriminator. A different direction to address the non-differentiability problem is approximation methods. Specifically, (Zhang et al., 2017) and (Chen et al., 2018) apply an annealed softmax to approximate the argmax operation. While (Gu et al., 2018) and (Nie et al., 2018) propose to use the Gumbel-Softmax relaxation to approximate the categorical distribution.

The aforementioned methods mostly focus on general text generation. For the task of category text generation, CSGAN (Li et al., 2018) proposes a descriptor which consists of a binary discriminator and a classifier that aims to distinguish the categories of the given sentence. However, the adversarial generator optimization of CSGAN still adopts the reward-based training objective, which is the same as SeqGAN and restricts the diversity of the generated sentences. Meanwhile, SentiGAN (Wang and Wan, 2018) introduces multiple generators where each generator focuses on generating samples with a specified sentiment label. In addition, SentiGAN proposes a novel penalty-based training objective to improve the diversity of the generated samples. However, the multiple generators of SentiGAN will increase the complexity of the model with the increase of category numbers. More recently, CatGAN (Liu et al., 2020) introduces a category-aware model for category text generation. Nevertheless, the discriminator of CatGAN still focuses on vanilla data-to-class relations. As can be noticed, in all the methods discussed above, data-to-data relations among the training batch are ignored. Besides these GAN-based methods, some works also make various attempts to improve the conventional conditional generative models (Keskar et al., 2019; Chan et al., 2021; Li et al., 2020), and they are orthogonal to our model which focuses on improving the discriminator to better guide the conditional generator. And these models can replace the conditional generator of our model for further improvements. Thus, in this work, we mainly focus on the GAN-based methods.

Recently, many unsupervised representation learning methods are proposed based on the principle of contrastive learning (Wu et al., 2018; Bachman et al., 2019; He et al., 2020; Henaff, 2020; Chen et al., 2020a,b), and contrastive learning paradigm is firstly adopted for adversarial image generation in (Kang and Park, 2020). Besides, contrastive learning is also applied to conventional conditional text generative models (Lee et al., 2021; Qian et al., 2022). To effectively leverage the class label information, the proposed CoCGAN makes the first attempt to integrate contrastive learning into the discriminator for adversarial category text generation. By exploring both more flexible data-to-class relations and data-to-data relations with contrastive learning, CoCGAN achieves significant improvements over previous works, including the state-of-the-art model CatGAN.

3 Methodology

In this section, we propose CoCGAN by adapting contrastive learning to adversarial category text generation. We begin with introducing the framework of the generator (Sec. 3.1). Then, in order to consider both more flexible data-to-class relations and data-to-data relations, we split the discriminator into an authenticity discriminator and a contrastive category projector (Sec. 3.2) to introduce a label-incorporated contrastive loss. Accordingly, a penalty-based contrastive learning paradigm is designed to optimize the generator during adversarial training (Sec. 3.3). Finally, we propose the Contrastive Category Generative Adversarial Net (CoCGAN) for category text generation (Sec. 3.4).

3.1 Conditional Generator

Following previous works (Yu et al., 2017; Wang and Wan, 2018; Liu et al., 2020), we adopt the generative model based on recurrent neural net-

work (RNN) (Hochreiter and Schmidhuber, 1997). An RNN maps the input embedding representations $\boldsymbol{x}_1, \boldsymbol{x}_2, \ldots, \boldsymbol{x}_T$ of the sequence $x_1, x_2, \ldots, x_T$ into a sequence of hidden states $\boldsymbol{s}_1, \boldsymbol{s}_2, \ldots, \boldsymbol{s}_T$ by using the update function g recursively. Different from the SentiGAN which uses k conventional RNNs to generate texts of k classes, we equip a single conventional RNN with an additional label embedding input $\boldsymbol{c}$ to control the category of the generated text, which can reduce the complexity of the generator, and the hidden state $\boldsymbol{s}_t$ is updated as follows:

$$\boldsymbol{s}_t = g(\boldsymbol{s}_{t-1}, \boldsymbol{x}_t, \boldsymbol{c}) \tag{1}$$

where $\boldsymbol{x}_t$ and $\boldsymbol{c}$ are concatenated as the input at timestep t. Then, a softmax output layer maps the hidden states into the output token distribution:

$$p(y_t|x_1, x_2, \ldots, x_t) = \text{softmax}(\boldsymbol{V}\boldsymbol{s}_t + \boldsymbol{b}) \tag{2}$$

where $\boldsymbol{V}$ and $\boldsymbol{b}$ are weight matrix and bias vector respectively. It is worth noting that the generator can be implemented as most of the RNN variants, such as the Long Short-Term Memory (LSTM) cells (Hochreiter and Schmidhuber, 1997), the gated recurrent unit (GRU) (Cho et al., 2014) and the relational memory core (RMC) (Santoro et al., 2018). For easy comparison with the state-of-the-art model CatGAN, which uses RMC-based generator, we adopt both LSTM and RMC as the generators to conduct experiments.

3.2 Label-incorporated Contrastive Discriminator

In SentiGAN, the discriminator is designed as a $(k+1)$-class classifier to distinguish between the real texts with each category (k classes) and the generated texts (1 class). However, it is overlooked that each generated sample is simultaneously associated with the degree of the authenticity and the probability of belonging to a certain category.

3.2.1 Auxiliary Classification Loss

In order to address the problem mentioned above, we make a clear distinction between the authenticity and the real category of the text. Specifically, as shown in Figure 1(b), we divide the $(k+1)$-class SentiGAN discriminator (Figure 1(a)) into two parts: an authenticity discriminator and a category classifier. The authenticity discriminator is designed as a binary classifier to distinguish between the real and generated samples, while the category classifier is a k-class classifier to distinguish the categories of the given sample. We denote the discriminator as D_ϕ where ϕ is the trainable parameter. Besides, we adopt part of the discriminator network (D_{ϕ_1}) before the fully connected layer as the encoder network and use a multi-layer perceptron network parameterized by φ as the body of the category classifier. The training objective of the discriminator is defined as follows:

$$\begin{aligned}
\mathcal{L}_{Dis} =& \mathcal{L}_{Auth} + \mathcal{L}_{Cat} \\
\mathcal{L}_{Auth} =& \mathbb{E}_{X\sim P_g} \log D_\phi(X) - \mathbb{E}_{X\sim P_r} \log D_\phi(X) \\
\mathcal{L}_{Cat} =& -\mathbb{E}_{(X,Y)\sim P_r(X,Y)} \log C_{\phi_1,\varphi}(X,Y)
\end{aligned} \tag{3}$$

where $\mathcal{L}_{Auth}$ and $\mathcal{L}_{Cat}$ are the losses of the authenticity discriminator and the category classifier respectively. P_g and P_r represent the generated texts and the real texts respectively, and $P_r(X,Y)$ indicates the real text-label pairs (X and Y represent the real text and the corresponding class label respectively). $D_\phi(X)$ represents the probability that X is real and $C_{\phi_1,\varphi}(X,Y)$ represents the Y-th index of the classifier output, which represents the probability that X belongs to the real Y-th category texts. Here, we comprehensively consider the contributions of the authenticity discriminator and the category classifier. However, the category classifier still uses a cross-entropy loss, which only captures data-to-class relations between a fixed one-hot class label vector and a given text sample.

3.2.2 Label-incorporated Contrastive Loss

As shown in Figure 1(c), to exploit more flexible data-to-class relations and data-to-data relations, we adopt the training paradigm of self-supervised contrastive learning and replace the vanilla k-class classifier with the contrastive category projector.

Firstly, we construct a contrastive learning objective for the discriminator to explicitly control the distances between the text embeddings and the class label embeddings. Different from the conventional contrastive learning NT-Xent loss, which needs appropriate data augmentation and does not take data-to-class relations into account, we leverage the class label embeddings of the categories instead of data augmentation. Given a batch of training text samples $\{X_1, X_2, \ldots, X_m\}$ and the corresponding class labels $\{Y_1, Y_2, \ldots, Y_m\}$, we introduce an encoder $S(\cdot)$ and a projection layer h to map the input text samples to the hypersphere: $l = h(S(\cdot))$. Together with the label embeddings,

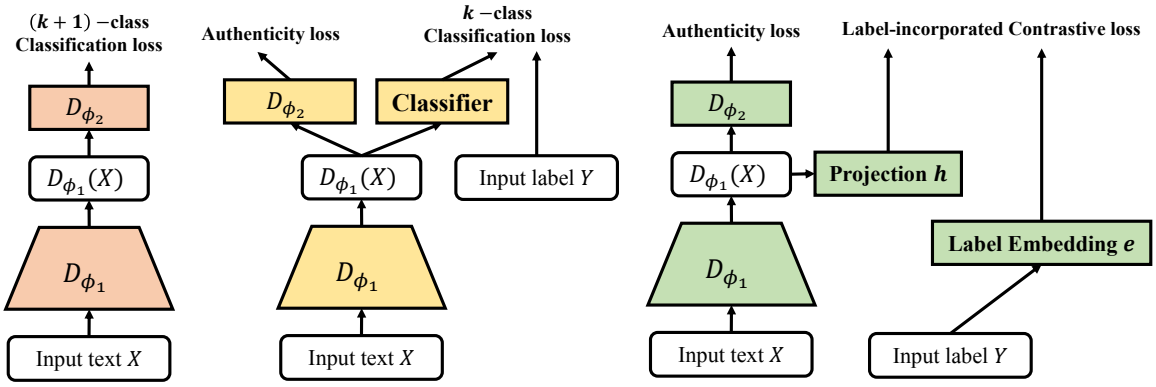

(a) SentiGAN (b) CLSGAN (Ours) (c) CoCGAN (Ours)

Figure 1: Schematics of discriminators of three category text GANs. (a) SentiGAN (Wang and Wan, 2018) takes a $(k+1)$-class classifier as its discriminator to guide the generator to generate category texts. (b) CLSGAN improves SentiGAN by explicitly divide the discriminator as a binary authenticity discriminator and a category classifier. (c) The proposed CoCGAN extends the CLSGAN with a label-incorporated contrastive loss, which considers both more flexible data-to-class relations and data-to-data relations in the same batch.

the contrastive loss is defined as follows:

$$\begin{aligned} \ell(X_i, Y_i; \tau) &= -\log \frac{R_{D2C}}{R_{D2C} + R_{D2A}} \\ R_{D2C} &= \exp(l(X_i)^\top e(Y_i)/\tau) \\ R_{D2A} &= \sum_{s=1}^{m} 1_{s \neq i} \cdot \exp(l(X_i)^\top l(X_s)/\tau) \end{aligned} \tag{4}$$

where $e(\cdot)$ denotes the class label embedding function, τ is a temperature scalar to control the pull and push force. (4) pulls the sample X_i nearer to its corresponding class label embedding $e(Y_i)$ and pushes the other samples away. In this work, we adopt part of the discriminator (D_{ϕ_1}) as the encoder $S(\cdot)$ and the multi-layer perceptron network parameterized by φ as the projection layer h to construct the mapping as $l = h(D_{\phi_1}(\cdot))$. To further exploit data-to-data relations, we should also avoid pushing other samples which have the same class label Y_i. Thus, we add cosine similarities of such samples to the numerator in (4) and get the label-incorporated contrastive loss as follows:

$$\begin{aligned} \ell_{\phi_1,\varphi}(X_i, Y_i; \tau) &= -\log \frac{R_{D2C} + R_{D2D}}{R_{D2C} + R_{D2A}} \\ R_{D2D} &= \sum_{s=1}^{m} 1_{Y_s = Y_i, s \neq i} \cdot \exp(l(X_i)^\top l(X_s)/\tau) \end{aligned} \tag{5}$$

where R_{D2C} and R_{D2A} are the same as in (4). Besides reducing the distances between the text embeddings and the corresponding class label embeddings, minimizing (5) will also reduce the distances between the multiple text embeddings with the same class labels while maximizing the others. It is obvious that (5) comprehensively considers more flexible data-to-class relations $l(X_i)^\top e(Y_i)$ and data-to-data relations $l(X_i)^\top l(X_s)$. And the objective of the discriminator can be redefined as:

$$\begin{aligned} \mathcal{L}_{Dis} &= \mathcal{L}_{Auth} + \mathcal{L}_{Cat} \\ \mathcal{L}_{Auth} &= \mathbb{E}_{X \sim P_g} \log D_\phi(X) - \mathbb{E}_{X \sim P_r} \log D_\phi(X) \\ \mathcal{L}_{Cat} &= \mathbb{E}_{(X,Y) \sim P_r(X,Y)} \ell_{\phi_1,\varphi}(X, Y; \tau) \end{aligned} \tag{6}$$

where $P_r(X, Y)$ indicates the real text-label pairs (X and Y represent the real text and the corresponding class label respectively).

3.3 Penalty-based Contrastive Generator Training

For adversarial generator training, instead of the reward-based policy gradient strategy (Yu et al., 2017), we adopt the penalty-based one (Wang and Wan, 2018) to guarantee the diversity of generated text samples. Specifically, the goal of the generator $G_\theta(X|S, Y)$ is to minimize the penalty:

$$\mathcal{L}_{Gen} = \sum_{t=0}^{T-1} G_\theta(X_{t+1}|S_t, Y) \cdot V_{D_{\phi,\varphi}}^{G_\theta}(S_t, Y, X_{t+1}) \tag{7}$$

where T is the length of X, $G_\theta(X_{t+1}|S_t, Y)$ indicates the probability of selecting the $(t+1)$-th word given its current state and class label, denoted as S_t and Y respectively, and $V_{D_\phi}^{G_\theta}(S_t, Y, X_{t+1})$ is

the penalty for the sequence $X_{1:t+1}$, which is calculated by the discriminator. Here Y is the class label corresponding to the label embedding $\boldsymbol{c}$ in (1). Monte Carlo tree search is applied with the roll-out policy G_θ to calculate the penalty function of the generator:

$$V_{D_{\phi,\varphi}}^{G_\theta}(S_{t-1}, Y, X_t) = \begin{cases} \frac{1}{N}\sum_{n=1}^{N}(1-r^n) & t < T \\ 1-r & t = T \end{cases} \tag{8}$$

where T is the length of X, r^n and r are given by the specified discriminator with the n-th Monte Carlo tree search result $X_{1:t}^n$ and the completely generated sentence X as input, respectively.

For the discriminator with auxiliary classification loss proposed in Sec. 3.2.1, r (analogy to r^n) is defined as follows:

$$r = \frac{1}{2}(D_\phi(X) + C_{\phi_1,\varphi}(X, Y)). \tag{9}$$

And for the discriminator with label-incorporated contrastive loss introduced in Sec. 3.2.2, r (analogy to r^n) is redefined as follows:

$$r = \frac{1}{2}(D_\phi(X) + \exp(-\ell_{\phi_1,\varphi}(X, Y; \tau))). \tag{10}$$

It is worth noting that, for both (9) and (10), X represents the generated text and Y is the corresponding input class label.

3.4 Contrastive Category Generative Adversarial Net

With the proposed label-incorporated contrastive loss, we build the framework of CoCGAN. Similar to the training procedure of GAN, CoCGAN has a discriminator step and a generator step which constitute adversarial training. Besides, CoCGAN calculates the label-incorporated contrastive loss with a set of real or generated samples. Algorithm 1 in Appendix A.3 summarizes the complete training procedure of the proposed CoCGAN. We also define the framework with auxiliary classification loss as CLSGAN, and Algorithm 2 in Appendix A.3 shows its training procedure. Different from CSGAN, CLSGAN adopts the penalty-based policy gradient strategy instead of the reward-based one. For both CoCGAN and CLSGAN, the training samples fed into the contrastive category projector and the category classifier are different for discriminator and generator step (i.e., real samples for discriminator step and generated samples for generator step). And for each discriminator iteration, the amount of generated samples is set the same as that of the real samples in a batch to guarantee sufficient training.

In CoCGAN, the discriminator can minimize the distances between the multiple real text embeddings from the same class label while maximizing it otherwise and capture more flexible relations between the real text embeddings and the corresponding class label embeddings. Besides, the relations between the current text embeddings and the wrong class label embeddings are also considered in (5). Specifically, since the wrong class label embeddings are pulled near to their corresponding text embeddings and the text embeddings from wrong class labels are pushed away from the current text embeddings, the wrong class label embeddings are pushed away from the current text embeddings implicitly. Therefore, the discriminator can learn better representations of the given samples, and the conditional generator can be further improved to generate more realistic category texts with the knowledge of the discriminator.

4 Experiments

4.1 Datasets

Without loss of generality, we set the number of categories as 2 and conduct experiments on both synthetic and real datasets, as in previous work (Liu et al., 2020). The synthetic data includes 20,000 samples, and each 10,000 samples are obtained from different oracle-LSTMs to construct category text data. The real data includes two English review datasets: movie reviews (MR) (Socher et al., 2013) and amazon reviews (AR) (McAuley et al., 2015). MR has two sentiment classes (negative and positive), and AR has two types of product reviews (book and application). We follow the same preprocessing steps as in LeakGAN (Guo et al., 2018). MR has 4,503 samples, including 3,153 samples for training and 1,350 samples for testing. As for AR, each review category includes 100,000 samples for training and 10,000 samples for testing, and each sample may consist of multiple sentences.

4.2 Evaluation Metrics

There exist many evaluation metrics to measure the performance of adversarial text generation models. Among them, (Yu et al., 2017) introduces the

Model	SentiGAN	CSGAN	CatGAN	CLSGAN_L	CoCGAN_L	CLSGAN_R	CoCGAN_R
20	6.953	8.522	6.631	6.903	6.611	6.712	**6.314**
40	6.877	7.703	6.392	6.663	6.384	6.445	**6.094**

Table 1: The NLL_oracle scores on synthetic dataset. For the NLL_oracle scores, the lower the better.

Model	MR						AR					
	B@2	B@3	B@4	B@5	N_g	N_d	B@2	B@3	B@4	B@5	N_g	N_d
SentiGAN	0.525	0.287	0.162	0.144	2.501	0.472	0.858	0.811	0.712	0.537	3.367	0.916
CSGAN	0.447	0.199	0.120	0.089	2.937	0.243	0.863	0.677	0.431	0.239	3.373	1.104
CatGAN	0.592	0.330	0.195	0.162	1.679	0.521	0.965	0.910	0.855	0.721	3.143	1.472
CLSGAN_L	0.557	0.327	0.183	0.161	2.313	0.491	0.933	0.892	0.810	0.622	3.306	1.112
CoCGAN_L	0.588	0.342	0.213	0.173	1.686	0.526	0.958	0.913	0.851	0.729	3.152	1.231
CLSGAN_R	0.573	0.344	0.201	0.167	1.958	0.517	0.943	0.903	0.841	0.663	3.195	1.289
CoCGAN_R	**0.632**	**0.383**	**0.227**	**0.182**	**1.462**	**0.536**	**0.984**	**0.957**	**0.882**	**0.764**	**3.024**	**1.537**

Table 2: The comparison of performance on MR and AR. B@n denotes BLEU scores of n-gram. For all BLEU scores, the higher the better. For NLL_gen scores (denoted as N_g), the lower the better. For NLL_div scores (denoted as N_d), the higher the better.

negative log-likelihood NLL_oracle to measure the quality on the synthetic data as follows:

$$\text{NLL}_\text{oracle} = -\mathbb{E}_{Y_\theta \sim P_\theta}[\log P_r(Y_\theta)] \quad (11)$$

where P_θ is the generated data distribution and P_r is the real data distribution.

As for the real data, we adopt NLL_gen (Zhu et al., 2018) and NLL_div (Liu et al., 2020) as the diversity metrics, and define them as follows:

$$\text{NLL}_\text{gen} = -\mathbb{E}_{Y_r \sim P_r}[\log P_\theta(Y_r)], \quad (12)$$
$$\text{NLL}_\text{div} = -\mathbb{E}_{Y_\theta \sim P_\theta}[\log P_\theta(Y_\theta)]. \quad (13)$$

To measure the quality on the real data, since we cannot access the distribution of the real data, we adopt BLEU scores (Zhu et al., 2018) to measure the performance of the models instead of NLL_oracle. And we follow (Liu et al., 2020) to use harmonic mean values of all automatic metrics on each category to evaluate the category text generation models.

4.3 Baselines

We conduct experiments to compare the proposed model with several state-of-the-art methods. For automatic evaluation metrics, we select SentiGAN (Wang and Wan, 2018), CSGAN (Li et al., 2018) and CatGAN (Liu et al., 2020) as baseline models. All models are pre-trained with standard MLE training before adversarial training. All the models are run with 5 random seeds on all experiments and the mean is presented as the final score (see Appendix A.1 for more detailed settings). For the proposed CoCGAN and CLSGAN, we adopt both LSTM and RMC as the generators to conduct experiments.

4.4 Quantitative Results

In this section, for CLSGAN and CoCGAN, we report the results of LSTM-based generator and RMC-based generator (i.e., CLSGAN_L, CLSGAN_R, CoCGAN_L and CoCGAN_R).

4.4.1 Results on the Synthetic Data

We conduct experiments on the synthetic data with the sequence length set as 20 and 40 respectively. Table 1 shows that, CoCGAN equipped with RMC-based generator consistently outperforms other models in terms of NLL_oracle, including the state-of-the-art model CatGAN, which demonstrates that CoCGAN can further exploit the category information with better quality on all categories.

4.4.2 Results on the Real Data

As for the real data (i.e., MR and AR), we use several metrics to measure the quality and the diversity of the generated sentences. After the same preprocessing steps, MR dataset consists of 6,216 unique words with the maximum sentence length 15, and AR dataset contains 6,416 unique words with the maximum sentence length 40. We report the results of CLSGAN and CoCGAN with different generators as on the synthetic data, and the results are presented in Table 2. It is obvious that CLSGAN_L shows its superiority on all metrics compared with CSGAN, since the penalty-based training paradigm adopted by CLSGAN_L can improve the performance compared with the reward-based one of CSGAN. CoCGAN further exploits more flexible data-to-class relations and data-to-data relations to achieve significant improvements over CLSGAN. When equipped with the RMC-

Model	(A)	(B)	(C)	(D)
B@2	0.943	**0.984**	0.952	0.963
B@3	0.903	**0.957**	0.917	0.931
B@4	0.841	**0.882**	0.855	0.847
B@5	0.663	**0.764**	0.734	0.751
N_g	3.195	**3.024**	3.132	3.103
N_d	1.289	**1.537**	1.368	1.475

Table 3: The ablation Study on AR. (A), (B), (C) and (D) represent CLSGAN, CoCGAN, CoCGAN w/o D2D and CoCGAN w/o D2C, respectively. All the models are equipped with RMC-based generator. B@n denotes BLEU scores of n-gram. For all BLEU scores, the higher the better. For NLL_{gen} scores (denoted as N_g), the lower the better. For NLL_{div} scores (denoted as N_d), the higher the better.

τ	0.1	1.0	2.0	4.0
B@2	0.873	**0.984**	0.972	0.951
B@3	0.792	**0.957**	0.938	0.893
B@4	0.698	**0.882**	0.877	0.845
B@5	0.573	**0.764**	0.752	0.710
N_g	3.405	**3.024**	3.121	3.303
N_d	1.372	**1.537**	1.475	1.402

Table 4: Tuning of temperature τ. The experiment is conducted on AR. B@n denotes BLEU scores of n-gram. For all BLEU scores, the higher the better. For NLL_{gen} scores (denoted as N_g), the lower the better. For NLL_{div} scores (denoted as N_d), the higher the better.

based generator, for both CoCGAN and CLSGAN, the performance is further improved compared with the one when equipped with the LSTM-based generator. It is noteworthy that CoCGAN using RMC-based generator outperforms the state-of-the-art model CatGAN with better quality and diversity on all categories as well.

4.4.3 Ablation Study

To investigate the contributions of different parts in the label-incorporated contrastive loss (discussed in Sec. 3.2.2), we conduct ablation study on AR. Here, we remove data-to-class relations from CoCGAN as CoCGAN w/o D2C (i.e., remove R_{D2C} from both denominator and numerator of (5) in Sec. 3.2.2), and data-to-data relations are removed from CoCGAN as CoCGAN w/o D2D (i.e., (4) in Sec. 3.2.2). We also report the results of CLSGAN to compare the performance between the fixed one-hot class label vector and the trainable class label embedding. All the results are shown in Table 3. On the one hand, compared with the fixed one-hot class label vector (i.e., CLSGAN), flexible class label embedding (i.e., CoCGAN w/o D2D) can achieve better performance. On the other hand, data-to-class relations and data-to-data relations are complementary to each other. Without either of them, the performance of CoCGAN shows a significant degradation on all metrics (i.e., both CoCGAN w/o D2D and CoCGAN w/o D2C show worse performance compared with complete CoCGAN). Besides, both CoCGAN w/o D2D and CoCGAN w/o D2C still outperform CLSGAN.

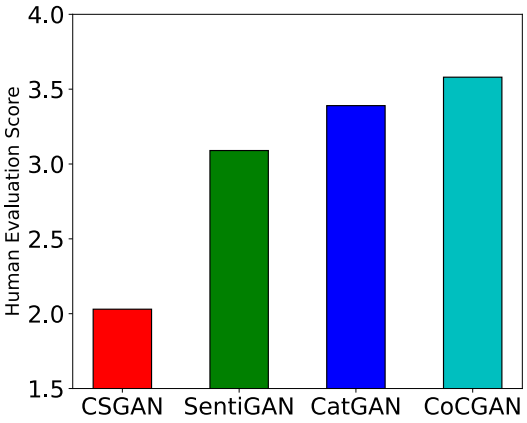

Figure 2: Comparison of human evaluation on a random subset of the AR dataset.

4.4.4 Tuning of Temperature τ

Temperature τ used in (5) is the hyper-parameter to control the pull and push force in contrastive learning and an appropriate temperature can help to capture better data-to-class relations and data-to-data relations. We investigate the impact of τ on AR with a grid search to find a proper value of τ. As shown in Table 4, we experimentally find that the temperature τ of 1.0 yields the best results.

4.5 Qualitative Results

For the qualitative experiments, we adopt RMC-based generator to construct CoCGAN, and only report the results of CoCGAN to compare with baselines (i.e., CSGAN, SentiGAN and CatGAN).

4.5.1 Human Evaluation

To further evaluate the quality of the generated sentences, we randomly select 50 generated samples from each category for human evaluation. The scores from 1 to 5 are assigned to each generated sample, which measures the fluency and the semantic correctness (see Appendix A.2 for more detailed evaluation protocols). The scores of 1 and 5 indicate the worst quality and the best quality

Dataset	SentiGAN	CSGAN	CatGAN	CoCGAN
MR	**Negative:** it's an extremely unpleasant film. **Positive:** it's not smart. **(wrong category)**	**Negative:** an enjoyable experience. **(wrong category)** **Positive:** just intelligence. **(short)**	**Negative:** the movie doesn't add anything fresh to sustain its clever concept. **Positive:** it's a romantic. **(short)**	**Negative:** bad movie is that is, it's just a better travelogue than finding solutions. **Positive:** a very well-made and entertaining picture.
AR	**Book:** i love this book so much. it is one of book that you can not put down. very well written. **Application:** this game is addictive and fun. **(short)**	**Book:** my 4 is on the front of that day seeing travel. just not great, though .. **(Unreadable)** **Application:** this is a fun game. my husband and i both love to play it a lot.	**Book:** this was an awesome book. i loved the book, every page kept me entertained and finished it in two days! **Application:** i really love these games. **(short)**	**Book:** i absolutely loved this book. i am so glad to read the other books in this series. i can't wait for the next one. **Application:** i love this game. it is a great way to pass the time.

Table 5: Generated samples of different models on the real dataset.

respectively. Each generated sample is rated by 10 invited human evaluators who are capable of reading English proficiently. And the harmonic mean values of the average score on each category are shown in Figure 2. It can be observed that, multiple generators help SentiGAN to obtain competitive results, while CatGAN has achieved better performance. And the results of CoCGAN demonstrate that the contrastive learning paradigm helps to consistently outperform these baselines.

4.5.2 Case Study

We select SentiGAN, CSGAN and CatGAN as references to analyze the effectiveness of the proposed label-incorporated contrastive objective. With trained on MR and AR, the generated samples of these models are listed in Table 5, and we can find some problems with the generated sentences of these baselines (e.g., unreadable and wrong category). In contrast, the proposed CoCGAN can produce sentences with better quality.

5 Conclusion

This paper proposes a novel contrastive learning paradigm for adversarial category text generation (CoCGAN). In CoCGAN, a novel label-incorporated contrastive loss is introduced to further exploit more flexible data-to-class relations and data-to-data relations in the training batch, and the category text generation model is enhanced accordingly. It is worth noting that CoCGAN focuses on the perspective of adversarial learning, therefore it is orthogonal to some works which try to optimize the conditonal text generative models themselves, and can be applied to them for further improvements. Extensive experiments demonstrate that our proposed model outperforms the state-of-the-art adversarial category text generation models with better quality and diversity.

Acknowledgements

This research was supported by Anhui Provincial Natural Science Foundation (2008085J31).

References

Philip Bachman, R Devon Hjelm, and William Buchwalter. 2019. Learning representations by maximizing mutual information across views. *Advances in neural information processing systems*.

Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. 2014. Neural machine translation by jointly learning to align and translate. *International Conference on Learning Representations*.

Samy Bengio, Oriol Vinyals, Navdeep Jaitly, and Noam Shazeer. 2015. Scheduled sampling for sequence prediction with recurrent neural networks. *Advances in neural information processing systems*.

Andrew Brock, Jeff Donahue, and Karen Simonyan. 2018. Large scale gan training for high fidelity natural image synthesis. *International Conference on Learning Representations*.

Massimo Caccia, Lucas Caccia, William Fedus, Hugo Larochelle, Joelle Pineau, and Laurent Charlin. 2020. Language gans falling short. *International Conference on Learning Representations*.

Alvin Chan, Yew-Soon Ong, Bill Pung, Aston Zhang, and Jie Fu. 2021. Cocon: A self-supervised approach for controlled text generation. *International Conference on Learning Representations*.

Liqun Chen, Shuyang Dai, Chenyang Tao, Dinghan Shen, Zhe Gan, Haichao Zhang, Yizhe Zhang, and Lawrence Carin. 2018. Adversarial text generation

via feature-mover's distance. *Advances in neural information processing systems*.

Ting Chen, Simon Kornblith, Mohammad Norouzi, and Geoffrey Hinton. 2020a. A simple framework for contrastive learning of visual representations. In *International Conference on Machine Learning*, pages 1597–1607. PMLR.

Xinlei Chen, Haoqi Fan, Ross Girshick, and Kaiming He. 2020b. Improved baselines with momentum contrastive learning. *arXiv preprint arXiv:2003.04297*.

Kyunghyun Cho, Bart van Merriënboer, Caglar Gulcehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. 2014. Learning phrase representations using RNN encoder–decoder for statistical machine translation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1724–1734, Doha, Qatar. Association for Computational Linguistics.

William Fedus, Ian Goodfellow, and Andrew M Dai. 2018. Maskgan: better text generation via filling in the_. *International Conference on Learning Representations*.

Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. 2014. Generative adversarial nets. *Advances in neural information processing systems*, 27.

Alex Graves. 2013. Generating sequences with recurrent neural networks. *arXiv preprint arXiv:1308.0850*.

Jiatao Gu, Daniel Jiwoong Im, and Victor OK Li. 2018. Neural machine translation with gumbel-greedy decoding. In *Thirty-Second AAAI Conference on Artificial Intelligence*.

Jiaxian Guo, Sidi Lu, Han Cai, Weinan Zhang, Yong Yu, and Jun Wang. 2018. Long text generation via adversarial training with leaked information. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 32.

Kaiming He, Haoqi Fan, Yuxin Wu, Saining Xie, and Ross Girshick. 2020. Momentum contrast for unsupervised visual representation learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 9729–9738.

Olivier Henaff. 2020. Data-efficient image recognition with contrastive predictive coding. In *International Conference on Machine Learning*, pages 4182–4192. PMLR.

Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long short-term memory. *Neural computation*, 9(8):1735–1780.

Ferenc Huszár. 2015. How (not) to train your generative model: Scheduled sampling, likelihood, adversary? *Computing Research Repository*.

Minguk Kang and Jaesik Park. 2020. Contragan: Contrastive learning for conditional image generation. *Advances in neural information processing systems*.

Nitish Shirish Keskar, Bryan McCann, Lav R Varshney, Caiming Xiong, and Richard Socher. 2019. Ctrl: A conditional transformer language model for controllable generation. *arXiv preprint arXiv:1909.05858*.

Seanie Lee, Dong Bok Lee, and Sung Ju Hwang. 2021. Contrastive learning with adversarial perturbations for conditional text generation. *International Conference on Learning Representations*.

Jiwei Li, Will Monroe, Tianlin Shi, Sébastien Jean, Alan Ritter, and Dan Jurafsky. 2017. Adversarial learning for neural dialogue generation. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 2157–2169, Copenhagen, Denmark. Association for Computational Linguistics.

Piji Li, Haisong Zhang, Xiaojiang Liu, and Shuming Shi. 2020. Rigid formats controlled text generation. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 742–751, Online. Association for Computational Linguistics.

Yang Li, Quan Pan, Suhang Wang, Tao Yang, and Erik Cambria. 2018. A generative model for category text generation. *Information Sciences*, 450:301–315.

Kevin Lin, Dianqi Li, Xiaodong He, Zhengyou Zhang, and Ming-Ting Sun. 2017. Adversarial ranking for language generation. *Advances in neural information processing systems*.

Zhiyue Liu, Jiahai Wang, and Zhiwei Liang. 2020. Catgan: Category-aware generative adversarial networks with hierarchical evolutionary learning for category text generation. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pages 8425–8432.

Julian McAuley, Christopher Targett, Qinfeng Shi, and Anton Van Den Hengel. 2015. Image-based recommendations on styles and substitutes. In *Proceedings of the 38th international ACM SIGIR conference on research and development in information retrieval*, pages 43–52.

Weili Nie, Nina Narodytska, and Ankit Patel. 2018. Relgan: Relational generative adversarial networks for text generation. In *International Conference on Learning Representations*.

Jing Qian, Li Dong, Yelong Shen, Furu Wei, and Weizhu Chen. 2022. Controllable natural language generation with contrastive prefixes. In *Findings of the Association for Computational Linguistics: ACL 2022*, pages 2912–2924, Dublin, Ireland. Association for Computational Linguistics.

Alec Radford, Luke Metz, and Soumith Chintala. 2016. Unsupervised representation learning with deep convolutional generative adversarial networks. *International Conference on Learning Representations*.

Adam Santoro, Ryan Faulkner, David Raposo, Jack Rae, Mike Chrzanowski, Theophane Weber, Daan Wierstra, Oriol Vinyals, Razvan Pascanu, and Timothy Lillicrap. 2018. Relational recurrent neural networks. *Advances in neural information processing systems*, 31.

Xin Sheng, Linli Xu, Junliang Guo, Jingchang Liu, Ruoyu Zhao, and Yinlong Xu. 2020. Introvnmt: An introspective model for variational neural machine translation. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pages 8830–8837.

Xin Sheng, Linli Xu, Yinlong Xu, Deqiang Jiang, and Bo Ren. 2022. Semantic-preserving abstractive text summarization with Siamese generative adversarial net. In *Findings of the Association for Computational Linguistics: NAACL 2022*, pages 2121–2132, Seattle, United States. Association for Computational Linguistics.

Richard Socher, Alex Perelygin, Jean Wu, Jason Chuang, Christopher D. Manning, Andrew Ng, and Christopher Potts. 2013. Recursive deep models for semantic compositionality over a sentiment treebank. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*, pages 1631–1642, Seattle, Washington, USA. Association for Computational Linguistics.

Richard S Sutton, David A McAllester, Satinder P Singh, and Yishay Mansour. 2000. Policy gradient methods for reinforcement learning with function approximation. In *Advances in neural information processing systems*, pages 1057–1063.

Ke Wang and Xiaojun Wan. 2018. Sentigan: Generating sentimental texts via mixture adversarial networks. In *IJCAI*, pages 4446–4452.

Ronald J Williams. 1992. Simple statistical gradient-following algorithms for connectionist reinforcement learning. *Machine learning*, 8(3):229–256.

Zhirong Wu, Yuanjun Xiong, Stella X Yu, and Dahua Lin. 2018. Unsupervised feature learning via non-parametric instance discrimination. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 3733–3742.

Lantao Yu, Weinan Zhang, Jun Wang, and Yong Yu. 2017. Seqgan: Sequence generative adversarial nets with policy gradient. In *Proceedings of the AAAI conference on artificial intelligence*, volume 31.

Yizhe Zhang, Zhe Gan, Kai Fan, Zhi Chen, Ricardo Henao, Dinghan Shen, and Lawrence Carin. 2017. Adversarial feature matching for text generation. In *International Conference on Machine Learning*, pages 4006–4015. PMLR.

Yaoming Zhu, Sidi Lu, Lei Zheng, Jiaxian Guo, Weinan Zhang, Jun Wang, and Yong Yu. 2018. Texygen: A benchmarking platform for text generation models. In *The 41st International ACM SIGIR Conference on Research & Development in Information Retrieval*, pages 1097–1100.

A Appendix

A.1 Experimental Settings

- We implement the baselines based on TextGAN benchmark [1].
- In the CNN based discriminator, the sizes of filter windows are set to be $\{2, 3, 4, 5\}$ and the dimension of each feature map is set as 300.
- The batch size is set to be 64 for all models. The embedding size for the generator is set as 32 and that for the discriminator as 64.
- Adam optimizer with the same hyper-parameters (i.e., $\beta_1 = 0.9$ and $\beta_2 = 0.999$) are employed to optimize all models.
- For MLE pre-training, we run 150 epochs with learning rate set as 0.01.
- For discriminator pre-training, the learning rate is set to be 0.0001.
- For adversarial training, the learning rate is set to be 0.0001 for both the generator and the discriminator.
- All models are trained on a RTX 3090 GPU.

A.2 Human Evaluation Protocols

For category text generation, we conduct human evaluation based on fluency and semantic correctness. The detailed protocols are shown as follows:

- 5-Excellent. Right category, well fluency, and making sense.
- 4-Good. Right category, acceptable fluency with some grammatical errors, and making sense.
- 3-Fair. Right category, no fluency, but conveying some meanings from some parts.
- 2-Poor. Right category, making no sense.
- 1-Bad. Wrong category, making no sense.

[1] https://github.com/williamSYSU/TextGAN-PyTorch

A.3 Algorithms

The training procedures of CoCGAN and CLSGAN are shown in Algorithm 1 and Algorithm 2, respectively. For both CoCGAN and CLSGAN, the training samples fed into the contrastive category projector and the category classifier are different for discriminator step and generator step (i.e., real samples for discriminator step and generated samples for generator step). For sufficient training of discriminator, at each discriminator iteration, half of a training batch are real samples while the other half are generated samples.

Algorithm 1 The training procedure of CoCGAN

Require: Real text dataset T with corresponding class labels; The number of class labels k; Generator G_θ; Discriminator $D_{\phi,\varphi}$; Temperature τ

Initialize G_θ, $D_{\phi,\varphi}$ with random weights
Pre-train G_θ using MLE on T
Generate fake samples F with random class labels using G_θ
Pre-train $D_{\phi,\varphi}$ via minimizing (6) on $\{T, F\}$
while G_θ not converged **do**
  for g-steps **do**
    Generate fake samples F with random class labels using G_θ
    Calculate penalty $V^{G_\theta}_{D_{\phi,\varphi}}$ by (8) and (10) on F
    Update G_θ by minimizing (7)
  end for
  for d-steps **do**
    Generate fake samples F with random class labels using G_θ
    Update $D_{\phi,\varphi}$ via minimizing (6) on $\{T, F\}$
  end for
end while

Algorithm 2 The training procedure of CLSGAN

Require: Real text dataset T with corresponding class labels; The number of class labels k; Generator G_θ; Discriminator $D_{\phi,\varphi}$

Initialize G_θ, $D_{\phi,\varphi}$ with random weights
Pre-train G_θ using MLE on T
Generate fake samples F with random class labels using G_θ
Pre-train $D_{\phi,\varphi}$ via minimizing (3) on $\{T, F\}$
while G_θ not converged **do**
  for g-steps **do**
    Generate fake samples F with random class labels using G_θ
    Calculate penalty $V^{G_\theta}_{D_{\phi,\varphi}}$ by (8) and (9) on F
    Update G_θ by minimizing (7)
  end for
  for d-steps **do**
    Generate fake samples F with random class labels using G_θ
    Update $D_{\phi,\varphi}$ via minimizing (3) on $\{T, F\}$
  end for
end while

An Efficient Coarse-to-Fine Facet-Aware Unsupervised Summarization Framework based on Semantic Blocks

Xinnian Liang[1]*, Jing Li[2], Shuangzhi Wu[3], Jiali Zeng[3], Yufan Jiang[3], Mu Li[3], Zhoujun Li[1]†
[1]State Key Lab of Software Development Environment, Beihang University, Beijing, China
[2] School of Information Renmin University of China, Beijing, China
[3]Tencent Cloud Xiaowei, Beijing, China
{xnliang,lizj}@buaa.edu.cn; heylijing@126.com
{frostwu,lemonzeng,garyyfjiang}@tencent.com,limugx@qq.com;

Abstract

Unsupervised summarization methods have achieved remarkable results by incorporating representations from pre-trained language models. However, existing methods fail to consider efficiency and effectiveness at the same time when the input document is extremely long. To tackle this problem, in this paper, we proposed an efficient Coarse-to-Fine Facet-Aware Ranking (C2F-FAR) framework for unsupervised long document summarization, which is based on the semantic block. The semantic block refers to continuous sentences in the document that describe the same facet. Specifically, we address this problem by converting the one-step ranking method into the hierarchical multi-granularity two-stage ranking. In the coarse-level stage, we propose a new segment algorithm to split the document into facet-aware semantic blocks and then filter insignificant blocks. In the fine-level stage, we select salient sentences in each block and then extract the final summary from selected sentences. We evaluate our framework on four long document summarization datasets: Gov-Report, BillSum, arXiv, and PubMed. Our C2F-FAR can achieve new state-of-the-art unsupervised summarization results on Gov-Report and BillSum. In addition, our method speeds up 4-28 times more than previous methods.[1]

1 Introduction

The text summarization task aims to condense a document or a set of documents into several sentences and keep the primary information. Recent years, both supervised (Liu and Lapata, 2019; Liu and Liu, 2021; Liu et al., 2021b) and unsupervised (Zheng and Lapata, 2019; Dong et al., 2021b; Liang et al., 2021, 2022) methods have made significant improvements over short documents with the development of semantic representations from Pre-trained Language Models (PLMs). Due to the noise and complexity of the increased input and output length, long-form document summarization is still a challenge (Tay et al., 2021; Akiyama et al., 2021; Grail et al., 2021). Compared with supervised one, unsupervised methods do not rely on large amounts of labeled data and have no limitation on input length. In addition, unsupervised methods can be easily adapted to data from different domains, types, and languages. In this paper, we focus on unsupervised extractive methods for long document summarization.

Most unsupervised extractive methods are graph-based (Zheng and Lapata, 2019; Dong et al., 2021b; Liang et al., 2021, 2022). They represent document sentences as nodes in a graph, where the edge value is the similarity between sentences. Then, they measure the importance of each node via computing the degree centrality (Radev et al., 2000) or running PageRank (Brin and Page, 1998) algorithm. Liang et al. (2021) pointed out that centrality-based methods always tend to select sentences within the same facet (i.e. aspect, sub-topic) and proposed a facet-aware ranking (FAR) method to tackle this problem. FAR forces a centrality-based model to select summary sentences from different facets by incorporating the relevance between the candidate summary and the document. However, this method faces two problems when the document is extremely long: 1) As the input length increases, the document will have more noise and insignificant facets. The relevance computation between the candidate summary and the document may cause the facet-aware ranking to be influenced by insignificant facets. 2) The running time of FAR will rise rapidly as the number of extracted sentences increases. Due to FAR needs to compute the relevance score number of combinations C_m^k times, where k is the number of extracted summary sentences and m is the number of candidate salient sentences.

*Contribution during internship at Tencent Inc.
†Corresponding Author
[1]https://github.com/xnliang98/c2f-far

Proceedings of the 29th International Conference on Computational Linguistics, pages 6415–6425
October 12–17, 2022.

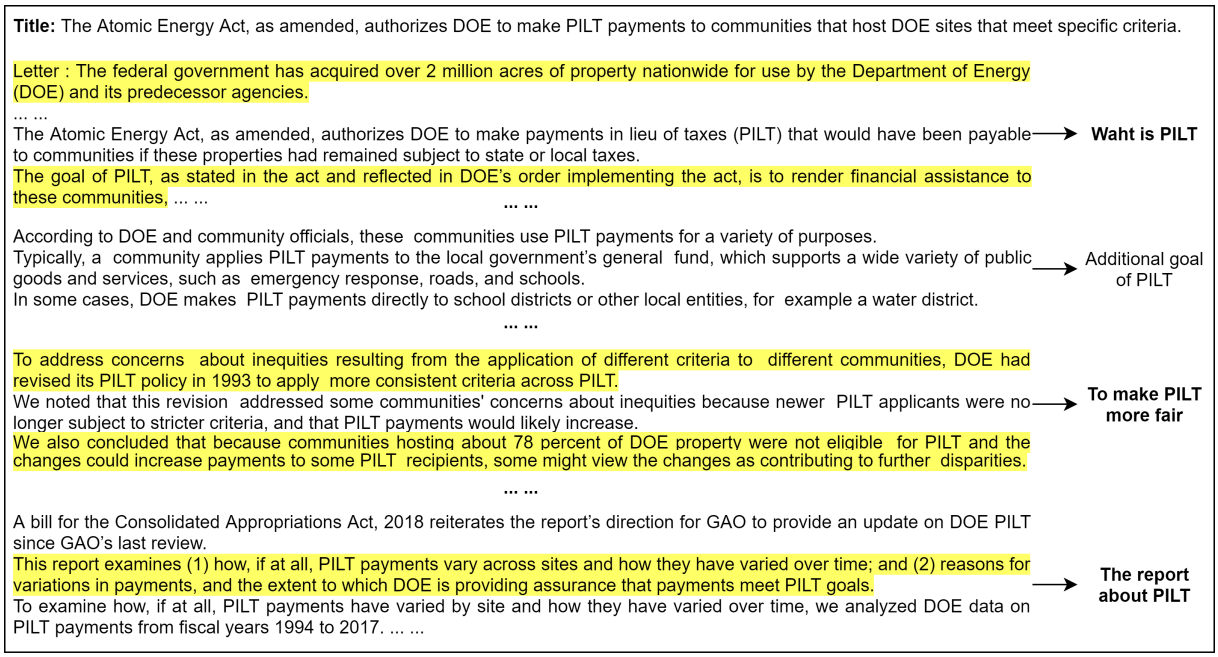

Figure 1: An example from the Gov-Report dataset to introduce the process of our method. "..." refers to the omissions of context sentences due to space limitations. Highlight sentences refer to the final extracted summary sentences. The content of the arrow pointed is the facet description of the left semantic block. Bold facets represent vital facet-aware semantic blocks of the final summary.

To tackle these problems, in this paper, we propose a novel Coarse-to-Fine Facet-Aware Ranking (C2F-FAR) Framework based on semantic blocks, which consists of two stages with different granularities: semantic blocks and sentences. The semantic block means continuing sentences that describe the same facet. We use a simple example in Fig. 1 to describe the motivation for building two stages. Fig. 1 shows four facet-aware semantic blocks. Each block contains continuous sentences describing the same facet, which is listed on the right. From the coarse-level view, we should first filter blocks with unimportant facets in the document, e.g. the block related to "additional goal of PILT" in Fig. 1. Then, from the fine-level view, we should select proper sentences in each block, which are more relevant to the block facet. Note that we only show the most relevant sentences with the facet of each semantic block and omit unrelated sentences due to the space limitation. Finally, the highlighted sentences should be selected as the summary.

Following this intuitive process, we designed our framework with a coarse-level stage and a fine-level stage. The coarse-level stage aims to select several salient facet-aware semantic blocks for the fine-level stage. We first segment the document into facet-aware semantic blocks by our proposed new document segmentation algorithm, which is inspired by TextTiling (Hearst, 1997). Then, we filter insignificant facets via a coarse-level centrality estimator to measure the salience of blocks. The fine-level stage aims to select final summary sentences from previously selected blocks. We first select candidate sentences in each block to represent its facet by simply computing relevance between sentences and the block. Finally, we extract the final summary from candidate sentences by sentence-level centrality-based estimator. Overall, the coarse-level stage can identify all facets of the document effectively and filter insignificant ones. The fine-level stage can reduce the influence of facets with many sentences by only selecting several related sentences for the final ranking. This framework with a hierarchical coarse-to-fine structure can guarantee effective and efficient long document summarization.

We evaluate the effectiveness and efficiency of our C2F-FAR on four long-document summarization datasets with two different metrics. Our method achieves new state-of-the-art performance on Gov-Report and BillSum. It is comparable to strong baselines on arXiv and PubMed. Besides, our method can achieve a speedup of 4-28 times

more than two strong baselines.

2 Methodology

We show the workflow of our proposed coarse-to-fine facet-aware ranking (C2F-FAR) framework in Fig. 2. After encoding the document into sentence embeddings, the workflow contains two main stages and each stage contains two steps.

(1) In the coarse-level stage, we first employ a document segmentation algorithm to split the document into coarse-level semantic blocks and we call them facet-aware semantic blocks. Then, we score all blocks via the centrality estimator and select top-ranked blocks for the next fine-level stage.

(2) In the fine-level stage, we first select several sentences of each facet-aware semantic block, which can cover the main facet of each block. Then, we employ a sentence-level centrality estimator to score selected sentences and extract the final summary.

We describe the details of each step in the following sections.

2.1 Sentence Embeddings

Formally, let $\mathcal{D}$ indicate a long document containing n sentences $\{s_1, \ldots, s_n\}$. In this paper, we employ pre-trained language model to obtain the sentence embeddings $\{v_1, \ldots, v_n\}$. Specifically, we employ an improved BERT (Devlin et al., 2019a) from previous work PacSum (Zheng and Lapata, 2019) to represent each sentence s_i with the hidden state v_i of "[CLS]" token. This improved BERT can obtain better sentence semantic representation.

2.2 The Coarse-Level Stage

The coarse-level stage contains two steps: document segmentation and coarse-level centrality estimator. The document segmentation splits the document into semantic blocks. The coarse-level centrality estimator employs a directed centrality score to measure the importance of each facet-aware semantic block. After the coarse-level stage, we only keep top-ranked $\alpha \times m$ semantic blocks of the whole document, where m is the number of facet-aware semantic blocks and α is a hyper-parameter used to control the ratio of reserved important blocks (default $\alpha = 0.5$).

2.2.1 Document Segmentation Algorithm

We propose a simple but effective document segmentation algorithm to split the input document into facet-aware semantic blocks. This algorithm is based on the assumption that when sentences with adjacent positions are semantically similar, they focus on the same facet (Skorochod'ko, 1971). As shown in Fig. 3, the algorithm aims to select some potential segmentation points to segment the document into several facet-aware semantic blocks $\mathcal{P}_1 = \{s_{p_1^s}, ..., s_{p_1^e}\}$, ..., $\mathcal{P}_m = \{s_{p_m^s}, ..., s_{p_m^e}\}$. Our proposed document segmentation algorithm is inspired by TextTiling (Hearst, 1997). It contains two steps: similarity measure and segmentation point identification.

In the similarity measure step, we compute the similarity of sentences on both sides of the potential segmentation point g_i. Each side select w sentences and apply mean operation method over their vectors to obtain global representations $b_i^l = \frac{1}{w}\sum_{j=i-w+1}^{i} v_j$ and $b_i^r = \frac{1}{w}\sum_{j=i+1}^{i+w} v_j$, where b_i^l and b_i^r refer to the left and right side block with w sentences, respectively. The similarity of the sentence on both sides of the potential segmentation point g_i is computed by cosine similarity $sim_i = \frac{b_i^l \cdot b_i^r}{||b_i^l|| ||b_i^r||}$.

Then, we apply the moving average on the similarity list of potential segmentation points $\{sim_1, ..., sim_{n-1}\}$ to get a smooth similarity list with Equ. (1)

$$\hat{sim}_i = \frac{1}{2\hat{w}+1}\sum_{j=i-\hat{w}}^{i+\hat{w}} sim_j \tag{1}$$

where the $\hat{w}$ is the window size used for moving average operation and the similarity list is refactored as $\{\hat{sim}_1, ..., \hat{sim}_{n-1}\}$. In this paper, the window size w and $\hat{w}$ are all set as 2.

The segmentation point identification step is based on the smooth similarity list. We show an intuitive similarity curve in Fig. 4. If the value of $\hat{sim}_i$ is low, the facets in the left and right blocks are different. So we should segment them with the point g_i. We can see that segmentation points g_3 and g_5 are the local minimum value of the curve in Fig. 4, which are suitable to segment the document.

We convert the similarity list of the potential segmentation point into depth score series $\{d_i\}_{i=1}^{n-1}$ by Equ. 2 to select proper segmentation points.

$$\begin{aligned} d_i = &\max\{(\hat{sim}_{i-1} - \hat{sim}_i), 0\} \\ &+ \max\{(\hat{sim}_{i+1} - \hat{sim}_i), 0\} \end{aligned} \tag{2}$$

When the similarity of the potential segmentation point is the local minimum value, it will become

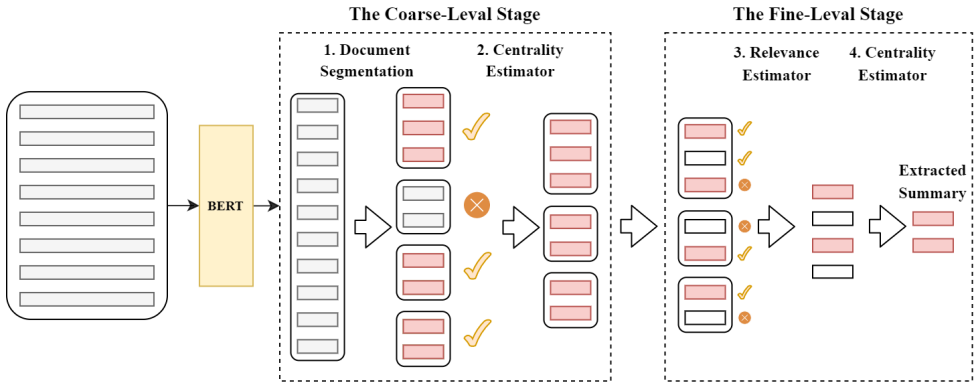

Figure 2: The workflow of our proposed coarse-to-fine facet-aware ranking framework.

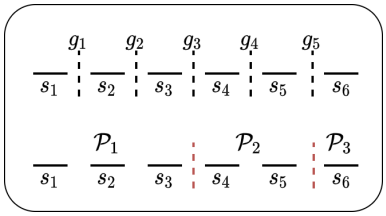

Figure 3: A diagram for document segmentation.

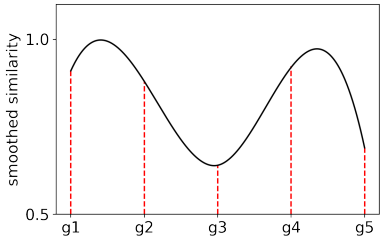

Figure 4: The smooth similarity curve.

the local maximum value after being converted into a depth score. If $d_i > \epsilon$, we choose the potential segmentation point g_i as the segmentation point. The ϵ is a threshold and is decided by the mean μ and standard deviation σ of the depth score series. We set $\epsilon = \mu + \lambda \cdot \sigma$, where λ is a hyper-parameter to control the granularity of segmentation. The greater the λ, the segmented block contains more sentences.

Finally, we can segment the whole document into some facet-aware semantic blocks $\mathcal{P}_1 = \{s_{p_1^s}, ..., s_{p_1^e}\}$, ..., $\mathcal{P}_m = \{s_{p_m^s}, ..., s_{p_m^e}\}$, like examples in the Fig. 3.

2.2.2 Coarse-Level Centrality Estimator

We introduce the coarse-level centrality estimator for filtering unimportant facet-aware semantic blocks in this section. We represent the semantic information of each block $\mathcal{P}_i$ by computing the average of sentence vectors contained in the block.

$$p_i = \frac{1}{|\mathcal{P}_i|} \sum_{s_i \in \mathcal{P}_i} (s_i) \tag{3}$$

The representations of blocks are $\{p_1, \ldots, p_m\}$. Then, we employ directed centrality (Zheng and Lapata, 2019) to score each block based on the assumption that the contribution of any two nodes' connection to their respective centrality is influenced by their relative position.

$$\mathcal{C}(p_i) = \lambda_1 \sum_{j<i}^{n} p_i \cdot p_j + \lambda_2 \sum_{j>i}^{n} p_i \cdot p_j \tag{4}$$

After that, we rank all blocks via directed centrality score $\mathcal{C}(p_i)$ and only keep top-ranked α percent semantic blocks for the next fine-level stage, where α is a hyper-parameter to control the ratio of reserved blocks.

2.3 The Fine-Level Stage

The fine-level stage contains two steps: relevance estimator and fine-level centrality estimator. The relevance estimator is used to select some sentences in each facet-aware semantic block, which can retain the main information of the block. The fine-level centrality estimator is applied to sentences from the previous relevance estimator and also employs the directed centrality score to extract the final summary.

2.3.1 Relevance Estimator

The relevance estimator simply computes the relevance between sentences and the block to select sentences to represent the facet in semantic blocks. This step is based on the assumption that each facet-aware semantic block only contains one facet. We employ cosine similarity to measure the relevance

between sentence representation v_j and block representation p_i.

$$\mathcal{R}(s_j) = \frac{v_j \cdot p_i}{||v_j||||p_i||}, s_j \in \mathcal{P}_i \quad (5)$$

For each semantic block, we select top-ranked β sentences, where β is the average number of semantic block sentences, which is determined by the granularity of document segmentation. If the number of sentences in a block is lower than β, we keep all sentences. Then, we can get t candidate sentences $\{\hat{s}_1, \ldots, \hat{s}_t\}$ for the final summary selection.

2.3.2 Fine-Level Centrality Estimator

The final fine-level centrality estimator aims to select the final summary sentences from previous candidate sentences. The final fine-level centrality estimator measures the importance of each candidate sentence as follows:

$$\mathcal{C}(s_i) = \lambda_1 \sum_{j<i}^{t} v_i \cdot v_j + \lambda_2 \sum_{j>i}^{t} v_i \cdot v_j \quad (6)$$

where $s_i, s_j \in \{\hat{s}_1, \ldots, \hat{s}_t\}$. We select top-ranked k sentences as the final summary, where k is the average number of sentences of different datasets.

3 Experiments

3.1 Datasets

Datasets	#docs	document		summary	
		words	sen.	words	sen.
Gov-Report	973	9,409	304	657	23
BillSum	3,269	2,148	169	209	10
arXiv	6,440	4,938	206	220	10
PubMed	6,658	3,016	107	203	8

Table 1: Statistics information of Gov-Report, BillSum, arXiv, and PubMed datasets. We compute the average document and summary length in terms of words and sentences, respectively.

We evaluate our C2F-FAR on 4 datasets. The statistics information of them is shown in Tab. 1.

Gov-Report (Huang et al., 2021) is a large-scale long document summarization dataset containing 19,466 long reports published by U.S. Government Accountability Office (GAO) and Congressional Research Service (CRS). Documents and summaries in Gov-Report are significantly longer than other datasets.

BillSum (Kornilova and Eidelman, 2019) contains US Congressional bills and human-written references from the 103rd-115th (1993-2018) sessions of Congress. We found that previous works have some errors in the sentence segmentation of the dataset. We re-segmented this dataset with the StanfordNLP toolkit and conducted experiments on the basis of the new sentence segmentation.

arXiv and **PubMed** (Cohan et al., 2018) are two long scientific document summarization datasets from scientific papers.

3.2 Settings and Metrics

We employ sentence-BERT[2] from (Zheng and Lapata, 2019) to encode sentences in the document, which converts each sentence into a vector with 768 elements. The window size of the document segmentation algorithm is 2. The default setting of λ is 1.0 and α is 0.5.

We reported ROUGE-1/2/L scores with `ROUGE-1.5.5.pl` script[3] (Lin, 2004) and BertScore (Zhang* et al., 2020) of baselines and our methods. The ROUGE score is the lexical level metric to measure the similarity between extracted summary and gold summary. The BertScore[4] measures the semantic level similarity between the extracted summary and gold reference.

3.3 Baselines

We compare our method with recent strong unsupervised extractive summarization models.

Lead, which selects the first k tokens as a summary.

Oracle, which is the upper bound of extractive summarization methods. It selects sentences by computing ROUGE scores with the gold summary.

TextRank (Mihalcea and Tarau, 2004) and **LexRank** (Erkan and Radev, 2004), which are two traditional unsupervised ranking method based on TF-IDF and PageRank algorithm to select salient sentences.

TextRank(BERT), which employs embeddings from improved BERT to compute the edge weight of TextRank.

FAR (Liang et al., 2021), which defined the facet bias problem and proposed a facet-aware centrality method to tackle the bias problem.

[2]https://github.com/huggingface/transformers
[3]https://github.com/andersjo/pyrouge
[4]https://github.com/Tiiiger/bert_score

3.4 Evaluation of Summary Quality and Inference Time

We report the results of automatic and human evaluation of all systems to measure the extracted summary quality of our C2F-FAR. Besides, we also compare the inference time of our method with two strong baselines to prove the high efficiency of our method.

The automatic evaluation results of ROUGE score and BertScore are shown in the Tab. 2 and Tab. 3. These two scores measure the lexical and semantic level similarity between extracted summary and gold reference, respectively. All reported results of our C2F-FAR framework employed the default hyper-parameters $\lambda = 1$ and $\alpha = 0.5$. We can see that our C2F-FAR achieved new state-of-the-art results on Gov-Report and BillSum in unsupervised methods. The performance of our method also is better than PacSum and comparable to FAR on the other two datasets: arXiv and PubMed. We will analyze the reason for the results on arXiv and PubMed in the discussion section. Interestingly, there is no big difference between the two versions of TextRank. We guess that the iterative algorithm based on PageRank is not sensitive to the similarity measure methods.

To evaluate the ability of our C2F-FAR in reducing facet bias and improving the quality of extracted summaries, we asked 3 human annotators to evaluate the extracted summaries of C2F-FAR and FAR with the gold reference summary. Three annotators were given extracted and gold summary. Then they were asked to give 0-2 scores for facets coverage (whether the extracted summary contains most primary facets) and quality (the comprehensive feelings of the extracted summary) of 20 random sampled examples from test sets of BillSum and 20 random sampled examples from test sets of Gov-Report (0-bad, 1-normal, 2-good). The results of FAR in terms of facets coverage is 1.16 and quality is 1.03. Our C2F-FAR performs significantly better ($p < 0.05$ with Mann-Whitney U tests) than FAR whose facets coverage is **1.38** and quality is **1.15**.

To test the inference time of our method, we randomly select 100 examples from the test set of each dataset and ensure that the average input length of these 100 examples is the same as the average length of the test set. Then, we run each method 10 times and report the average inference time of them on four datasets. We can see Fig. 5

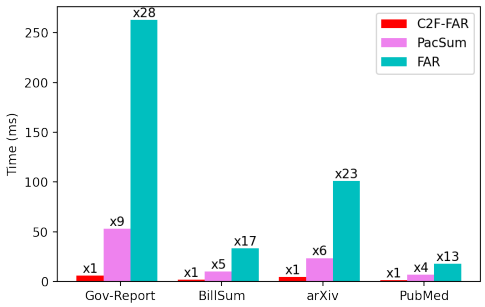

Figure 5: The inference time of each system. Each time is the average of multiple runs (10 times). "$\times N$" means the running time is N times (rounded up) of our method.

and find that our method is far ahead of the other two methods in inference time, and this advantage becomes more obvious as the length of the input document increases.

Overall, compared with other methods, our method takes into account both efficiency and effectiveness. In addition, our framework also can adjust the specific ranking method in each step for datasets with different types and domains, which makes it flexible.

4 Analysis

In this section, we first analyze the parameter sensitivity of our C2F-FAR and then discuss the reason why our method is inferior to the FAR on arXiv and PubMed via facets analysis of extracted sentences.

4.1 Impact of Hyper-parameters

In this section, we will analyze the parameter sensitivity of two hyper-parameters in our C2F-FAR framework: 1) λ is used to control the granularity of the document segmentation algorithm; 2) α is used to control the ratio of reserved blocks of the coarse-level centrality estimator. We can see the relationship between compression ratio and λ in the Tab. 4. The default setting of $\lambda = 1$ has an impressive compression ratio on two datasets.

We fix α and show the change of the ROUGE-1 score while λ changes in Fig. 6. We can find that the performance is best when $\lambda = 1.0$, and there is little change when $\lambda \in [0, 2]$. This shows that our algorithm is stable. You can set a larger λ to get a faster running speed while ensuring good performance. We set the value range of λ between 0.0 and 2.5 because when λ is less than 0, the most segmented blocks contain one sentence. Then the

Models	Gov-Report				BillSum			
	R-1	R-2	R-L	BS-F	R-1	R-2	R-L	BS-F
Oracle	74.87	49.02	72.48	88.83	65.24	47.09	58.81	86.29
Lead	50.94	19.53	48.45	83.47	40.53	18.28	34.15	80.24
LexRank	40.16	8.85	37.65	82.48	34.39	10.05	28.93	79.76
TextRank(TF-IDF)	53.19	23.12	49.86	84.83	40.04	16.12	32.64	80.81
TextRank(BERT)	56.00	22.42	52.86	85.10	38.05	12.99	31.46	80.02
PacSum	56.89	26.88	54.33	85.02	41.11	17.24	34.54	81.33
FAR	57.51	27.54	54.94	85.38	41.53	17.44	34.84	81.21
C2F-FAR	**57.98**	**27.63**	**55.33**	**86.62**	**42.53**	**17.85**	**35.58**	**81.57**

Table 2: Results on Gov-Report and BillSum test set. BS-F refers to F_1 of the BertScore.

Models	arXiv				PubMed			
	R-1	R-2	R-L	BS-F	R-1	R-2	R-L	BS-F
Oracle	53.88	23.05	34.9	87.06	55.05	27.48	38.66	87.05
Lead	33.66	8.94	22.19	82.97	35.63	12.28	25.17	80.43
LexRank	33.85	10.73	28.99	80.42	39.19	15.87	34.53	83.21
TextRank(TF-IDF)	36.59	10.06	30.29	82.49	38.66	15.87	34.53	82.43
TextRank(BERT)	34.68	8.78	30.05	81.19	39.43	12.89	34.66	83.39
PacSum	38.58	11.12	33.5	81.78	39.79	14.00	36.09	83.43
FAR	**40.92**	**13.75**	**35.56**	**83.74**	**41.98**	**16.74**	**37.58**	**83.89**
C2F-FAR	39.32	11.65	34.28	82.04	40.12	14.79	36.91	83.50

Table 3: Results on arXiv and PubMed test set. BS-F refers to F_1 of the BertScore.

Datasets	BillSum			Gov-Report		
λ	β	Para.	Comp.	β	Para.	Comp.
0	3	70	41%	3	120	39%
0.5	4	45	27%	5	74	24%
1	6	27	16%	10	44	14%
1.5	11	15	9%	15	26	9%
2	20	8	5%	20	15	5%
2.5	36	4	2%	36	4	1%

Table 4: Parameters affected by λ on two datasets. Para. means the average number of blocks with different hyper-parameters λ. Comp. means the ratio of the number of blocks to the number of sentences. β is the average number of sentences in a block.

following algorithms are equivalent to acting on the sentence-level structure.

We also fix λ and show the change of the ROUGE-1 score while α changes in Fig. 6. We can see that the second half of the curve is almost flat. This shows that the low centrality score of the segmented segment does not contribute to the final summary quality. The facets contained in these blocks are not important to the whole document. We can filter them with α in the coarse-level step and achieve a faster running speed.

The analysis of the two hyper-parameters proves that our C2F-FAR framework can employ simple hyper-parameter settings to improve the running speed of the algorithm while ensuring the quality of the summary.

4.2 Facets of Extracted Sentences

	Gov-Report	BillSum	arXiv	PubMed
#fac.	11.1	7.0	3.8	3.2
#sen.	20	10	10	7
#sen./#fac.	1.80	1.42	2.63	2.19

Table 5: #fac. refers to the average number of facet-aware semantic blocks, which contain extracted sentences. #sen. refers to the number of extracted sentences. #sen./#fac. refers to the average number of sentences from each block. Extracted sentences are from the Oracle system.

In Tab. 5, we employ the extracted sentences from the Oracle system to analyze the character-

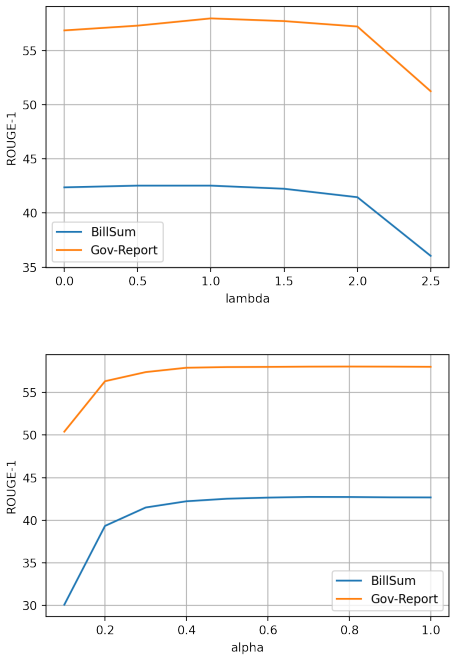

Figure 6: Impact of hyper-parameters λ and α.

istics of four datasets. The granularity of the document segmentation algorithm is $\lambda = 1$. We can see that selected summary sentences in arXiv and PubMed datasets distribute in fewer facet-aware semantic blocks than those in Gov-Report and BillSum. Our model tends to select summary sentences from more blocks, thus achieving better performance in Gov-Report and BillSum datasets.

By observing the extracted summary sentences from the Oracle system and combining the results in Tab. 5, we can roughly get the reason why our model is not as good as FAR on these datasets: the contents of the document and the summary is more concentrated on 3-4 facets of the document. Besides, the extracted sentences of them are mainly distribute at the start or end part (introduction and conclusion) of the document (Dong et al., 2021b). However, our method is more inclined to select summary sentences from more blocks and select many sentences in the middle part of the document. This leads to our method not performing so well on these two datasets.

5 Related Work

5.1 Long Document Summarization

Thanks to the development of Transformer-based (Vaswani et al., 2017) Pre-trained Language Models (PLMs), such as BERT (Devlin et al., 2019b), recent summarization models (Liu and Lapata, 2019; Zhang et al., 2019a; Li et al., 2020; Lewis et al., 2020; Zhong et al., 2020; Liu and Liu, 2021; Liu et al., 2021b) achieved excellent performance in short document summarization. However, these models can not be simply transferred to long document summarization due to both salient and noise content increasing according to the increase of the input text. How to summarize the long-form document, including books (Mihalcea and Ceylan, 2007), patents (Sharma et al., 2019), scientific publications (Qazvinian and Radev, 2008; Cohan et al., 2018), etc., is an important and long-standing challenge.

Most recent works for long-form document summarization are supervised and mainly tackle this problem through two angles. The first angle tends to design more efficient self-attention mechanisms to reduce the complexity. (Child et al., 2019; Kitaev et al., 2020; Beltagy et al., 2020; Zaheer et al., 2020; Huang et al., 2021; Tay et al., 2021; Dong et al., 2021a) The other angle employed the condense-then-generate paradigm (Cohan et al., 2018; Xu and Durrett, 2019; Zhang et al., 2019b; Lebanoff et al., 2019; Zhu et al., 2020; Akiyama et al., 2021; Grail et al., 2021). This paradigm first employs sentence/discourse-level structure to select salient sentences and then generates the summary based on them. This paradigm is intuitive and similar to the behavior of humans summarizing a long document. Our method also borrows some ideas from it.

5.2 Unsupervised Summarization

Most traditional unsupervised summarization methods are graph-based and extractive (Radev et al., 2000; Mihalcea and Tarau, 2004; Radev et al., 2000; Erkan and Radev, 2004; Wan, 2008). They represent the document as a graph, where each sentence is a node with a weighted edge which is the similarity between nodes. They rank sentences via computing centrality with node degree or PageRank algorithm (Brin and Page, 1998). Recently, many unsupervised works (Chu and Liu, 2019; Zhou and Rush, 2019; Zheng and Lapata, 2019; Yang et al., 2020; Xu et al., 2020; Liu et al., 2021a; Dong et al., 2021b; Liang et al., 2021) combined traditional methods with PLMs and achieved fantastic performance.

Zheng and Lapata (2019) first employed BERT to enhance similarity measure for graph-based ranking and proposed a directed degree centrality com-

putation method. Dong et al. (2021b) pointed out that the previous method is not suitable for long scientific papers and proposed a hierarchical discourse-based unsupervised ranking method. Liang et al. (2021) found that they all ignored the facet-bias problem (Mao et al., 2020), which is ubiquitous in unsupervised methods and proposed a facet-aware ranking method FAR. However, as the document length increases, they cannot extract proper sentences which cover vital facets of the document, from rapidly increased insignificant facets.

6 Conclusion

In this paper, we focus on unsupervised long document summarization tasks, which is a vital and long-standing challenge in text summarization. To obtain summary sentences efficiently and effectively, we proposed a novel coarse-to-fine facet-aware ranking framework. Our method can achieve new state-of-the-art results on two datasets. Experiments show that our approach is effective and efficient for the long document summarization task. In future work, we will investigate how to refactor this process into an end-to-end paradigm.

Acknowledgements

This work was supported in part by the National Natural Science Foundation of China (Grant Nos. 62276017, U1636211, 61672081), the 2022 Tencent Big Travel Rhino-Bird Special Research Program, and the Fund of the State Key Laboratory of Software Development Environment (Grant No. SKLSDE-2021ZX-18).

References

Kazuki Akiyama, Akihiro Tamura, and Takashi Ninomiya. 2021. Hie-BART: Document summarization with hierarchical BART. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Student Research Workshop*, pages 159–165, Online. Association for Computational Linguistics.

Iz Beltagy, Matthew E. Peters, and Arman Cohan. 2020. Longformer: The long-document transformer.

Sergey Brin and Lawrence Page. 1998. The anatomy of a large-scale hypertextual web search engine. *Computer Networks*, 30:107–117.

Rewon Child, Scott Gray, Alec Radford, and Ilya Sutskever. 2019. Generating long sequences with sparse transformers.

Eric Chu and Peter Liu. 2019. MeanSum: A neural model for unsupervised multi-document abstractive summarization. In *Proc. ICLR2019*, volume 97 of *Proceedings of Machine Learning Research*, pages 1223–1232, Long Beach, California, USA.

Arman Cohan, Franck Dernoncourt, Doo Soon Kim, Trung Bui, Seokhwan Kim, Walter Chang, and Nazli Goharian. 2018. A discourse-aware attention model for abstractive summarization of long documents. In *NAACL*, pages 615–621, New Orleans, Louisiana.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019a. Bert: Pre-training of deep bidirectional transformers for language understanding.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019b. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, Minneapolis, Minnesota. Association for Computational Linguistics.

Chenhe Dong, Yinghui Li, Haifan Gong, Miaoxin Chen, Junxin Li, Ying Shen, and Min Yang. 2021a. A survey of natural language generation. *arXiv preprint arXiv:2112.11739*.

Yue Dong, Andrei Mircea, and Jackie Chi Kit Cheung. 2021b. Discourse-aware unsupervised summarization for long scientific documents. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*, pages 1089–1102, Online. Association for Computational Linguistics.

Günes Erkan and Dragomir R. Radev. 2004. Lexrank: Graph-based lexical centrality as salience in text summarization. *J. Artif. Int. Res.*, 22(1):457–479.

Quentin Grail, Julien Perez, and Eric Gaussier. 2021. Globalizing BERT-based transformer architectures for long document summarization. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*, pages 1792–1810, Online. Association for Computational Linguistics.

Marti A. Hearst. 1997. Text tiling: Segmenting text into multi-paragraph subtopic passages. *Computational Linguistics*, 23(1):33–64.

Luyang Huang, Shuyang Cao, Nikolaus Parulian, Heng Ji, and Lu Wang. 2021. Efficient attentions for long document summarization. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 1419–1436, Online. Association for Computational Linguistics.

Nikita Kitaev, Łukasz Kaiser, and Anselm Levskaya. 2020. Reformer: The efficient transformer.

Anastassia Kornilova and Vladimir Eidelman. 2019. BillSum: A corpus for automatic summarization of US legislation. In *Proceedings of the 2nd Workshop on New Frontiers in Summarization*, pages 48–56, Hong Kong, China.

Logan Lebanoff, Kaiqiang Song, Franck Dernoncourt, Doo Soon Kim, Seokhwan Kim, Walter Chang, and Fei Liu. 2019. Scoring sentence singletons and pairs for abstractive summarization. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 2175–2189, Florence, Italy. Association for Computational Linguistics.

Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Veselin Stoyanov, and Luke Zettlemoyer. 2020. BART: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7871–7880, Online. Association for Computational Linguistics.

Wei Li, Xinyan Xiao, Jiachen Liu, Hua Wu, Haifeng Wang, and Junping Du. 2020. Leveraging graph to improve abstractive multi-document summarization. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 6232–6243, Online. Association for Computational Linguistics.

Xinnian Liang, Jing Li, Shuangzhi Wu, Mu Li, and Zhoujun Li. 2022. Improving unsupervised extractive summarization by jointly modeling facet and redundancy. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 30:1546–1557.

Xinnian Liang, Shuangzhi Wu, Mu Li, and Zhoujun Li. 2021. Improving unsupervised extractive summarization with facet-aware modeling. In *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*, pages 1685–1697, Online. Association for Computational Linguistics.

Chin-Yew Lin. 2004. ROUGE: A package for automatic evaluation of summaries. In *Text Summarization Branches Out*, pages 74–81, Barcelona, Spain. Association for Computational Linguistics.

Jingzhou Liu, Dominic J. D. Hughes, and Yiming Yang. 2021a. Unsupervised extractive text summarization with distance-augmented sentence graphs. In *Proceedings of the 44th International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '21, page 2313–2317, New York, NY, USA. Association for Computing Machinery.

Yang Liu and Mirella Lapata. 2019. Text summarization with pretrained encoders. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 3730–3740, Hong Kong, China. Association for Computational Linguistics.

Yixin Liu, Zi-Yi Dou, and Pengfei Liu. 2021b. RefSum: Refactoring neural summarization. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 1437–1448, Online. Association for Computational Linguistics.

Yixin Liu and Pengfei Liu. 2021. SimCLS: A simple framework for contrastive learning of abstractive summarization. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 2: Short Papers)*, pages 1065–1072, Online. Association for Computational Linguistics.

Yuning Mao, Liyuan Liu, Qi Zhu, Xiang Ren, and Jiawei Han. 2020. Facet-aware evaluation for extractive summarization. In *Proc. 58th Annu. Meet. Assoc. Comput. Linguist.*, pages 4941–4957, Online.

Rada Mihalcea and Hakan Ceylan. 2007. Explorations in automatic book summarization. In *Proceedings of the 2007 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning (EMNLP-CoNLL)*, pages 380–389, Prague, Czech Republic. Association for Computational Linguistics.

Rada Mihalcea and Paul Tarau. 2004. TextRank: Bringing order into text. In *Proceedings of the 2004 Conference on Empirical Methods in Natural Language Processing*, pages 404–411, Barcelona, Spain. Association for Computational Linguistics.

Vahed Qazvinian and Dragomir R. Radev. 2008. Scientific paper summarization using citation summary networks. In *Proceedings of the 22nd International Conference on Computational Linguistics (Coling 2008)*, pages 689–696, Manchester, UK. Coling 2008 Organizing Committee.

Dragomir R. Radev, Hongyan Jing, and Malgorzata Budzikowska. 2000. Centroid-based summarization of multiple documents: sentence extraction, utility-based evaluation, and user studies. In *NAACL-ANLP 2000 Workshop: Automatic Summarization*.

Eva Sharma, Chen Li, and Lu Wang. 2019. BIGPATENT: A large-scale dataset for abstractive and coherent summarization. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 2204–2213, Florence, Italy. Association for Computational Linguistics.

E. F. Skorochod'ko. 1971. Adaptive method of automatic abstracting and indexing. In *Proceedings of the IFIP Congress 71*, volume 2, pages 1179–1182.

Yi Tay, Mostafa Dehghani, Samira Abnar, Yikang Shen, Dara Bahri, Philip Pham, Jinfeng Rao, Liu Yang, Sebastian Ruder, and Donald Metzler. 2021. Long range arena : A benchmark for efficient transformers. In *International Conference on Learning Representations*.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Ł ukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc.

Xiaojun Wan. 2008. An exploration of document impact on graph-based multi-document summarization. In *Proceedings of the 2008 Conference on Empirical Methods in Natural Language Processing*, pages 755–762, Honolulu, Hawaii. Association for Computational Linguistics.

Jiacheng Xu and Greg Durrett. 2019. Neural extractive text summarization with syntactic compression. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 3292–3303, Hong Kong, China. Association for Computational Linguistics.

Shusheng Xu, Xingxing Zhang, Yi Wu, Furu Wei, and Ming Zhou. 2020. Unsupervised extractive summarization by pre-training hierarchical transformers. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 1784–1795, Online.

Ziyi Yang, Chenguang Zhu, Robert Gmyr, Michael Zeng, Xuedong Huang, and Eric Darve. 2020. TED: A pretrained unsupervised summarization model with theme modeling and denoising. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 1865–1874, Online.

Manzil Zaheer, Guru Guruganesh, Kumar Avinava Dubey, Joshua Ainslie, Chris Alberti, Santiago Ontanon, Philip Pham, Anirudh Ravula, Qifan Wang, Li Yang, et al. 2020. Big bird: Transformers for longer sequences. *Advances in Neural Information Processing Systems*, 33.

Jingqing Zhang, Yao Zhao, Mohammad Saleh, and Peter J. Liu. 2019a. Pegasus: Pre-training with extracted gap-sentences for abstractive summarization.

Tianyi Zhang*, Varsha Kishore*, Felix Wu*, Kilian Q. Weinberger, and Yoav Artzi. 2020. Bertscore: Evaluating text generation with bert. In *International Conference on Learning Representations*.

Xingxing Zhang, Furu Wei, and Ming Zhou. 2019b. HIBERT: Document level pre-training of hierarchical bidirectional transformers for document summarization. In *ACL*, pages 5059–5069, Florence, Italy.

Hao Zheng and Mirella Lapata. 2019. Sentence centrality revisited for unsupervised summarization. In *ACL*, pages 6236–6247, Florence, Italy.

Ming Zhong, Pengfei Liu, Yiran Chen, Danqing Wang, Xipeng Qiu, and Xuanjing Huang. 2020. Extractive summarization as text matching. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 6197–6208, Online. Association for Computational Linguistics.

Jiawei Zhou and Alexander Rush. 2019. Simple unsupervised summarization by contextual matching. In *ACL*, pages 5101–5106, Florence, Italy.

Chenguang Zhu, Ruochen Xu, Michael Zeng, and Xuedong Huang. 2020. A hierarchical network for abstractive meeting summarization with cross-domain pretraining. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 194–203, Online. Association for Computational Linguistics.

CHAE: Fine-Grained Controllable Story Generation with Characters, Actions and Emotions

Xinpeng Wang[1], Han Jiang[1], Zhihua Wei[1]*, Shanlin Zhou[2]
[1]Department of Computer Science and Technology, Tongji University, Shanghai, China
[2]School of Computer Science and Technology, Shanghai University of Electric Power, Shanghai, China
{wangxinpeng, 2230780, zhihua_wei}@tongji.edu.cn
zhoushanlin@mail.shiep.edu.cn

Abstract

Story generation has emerged as an interesting yet challenging NLP task in recent years. Some existing studies aim at generating fluent and coherent stories from keywords and outlines; while others attempt to control the global features of the story, such as emotion, style and topic. However, these works focus on coarse-grained control on the story, neglecting control on the details of the story, which is also crucial for the task. To fill the gap, this paper proposes a model for fine-grained control on the story, which allows the generation of customized stories with characters, corresponding actions and emotions arbitrarily assigned. Extensive experimental results on both automatic and human manual evaluations show the superiority of our method. It has strong controllability to generate stories according to the fine-grained personalized guidance, unveiling the effectiveness of our methodology. Our code is available at https://github.com/victorup/CHAE.

1 Introduction

Story generation, one of emergent tasks in the field of natural language generation, requires following sentences given the beginning of the story. For human beings, it is believed that storytelling requires strong logical thinking ability and organizational competence, and for machines it may be even more intractable. Nonetheless, works on story generation can help machines communicate with humans and drive improvements in natural language processing (Alabdulkarim et al., 2021).

At present, most works on story generation focus on the coherence of the story generated according to keywords, outlines and commonsense knowledge (Yao et al., 2019; Guan et al., 2019; Rashkin et al., 2020; Guan et al., 2020; Ji et al., 2020). Some other works aim at generating stories controlled by overall emotion, style, and topic (Keskar et al., 2019; Xu et al., 2020; Brahman and Chaturvedi, 2020; Kong et al., 2021). However, in reality, people often expect more detailed designs catering to their needs rather than a simple theme or topic in the generated story. For example, a novel with more complete elements, i.e., plot, character, theme, viewpoint, symbol, and setting is usually preferred to those made up out of thin air.

Taking the control in story generation as the cutting point, GPT-2 (Radford et al., 2019) can fulfill the story according to the beginning, but the process of generation cannot be controlled by people, resulting in unlogical outputs that lack practicality. CTRL (Keskar et al., 2019) can specify the generation of articles with different styles through some style words, but such control stays at the coarse-grained level, and makes a relatively weak influence. CoCon (Chan et al., 2020) introduces natural language to guide text generation. Fang et al. (2021) propose a new task that guides paragraph generation through a given sequence of outline events. However, the above two studies just explicitly add some contents to the generated sentences, which is similar to forming sentences with given phrases, not using the input as a condition guide for the generative models. SoCP presented by Xu et al. (2020) can generate stories under changeable psychological state control, while it does not govern the detailed contents of the story.

In this paper, we consider more fine-grained control on story generation, and propose a model, **CHAE** for fine-grained controllable story generation, allowing the generation of stories with customized **CH**aracters, and their **A**ctions and **E**motions. Characters are the core of the story. Their actions drive the story along, and their emotions make the story lively and interesting. Consequently, we take the characters along with their actions and emotions as the control conditions. It is a challenge that our model needs to control multiple characters with their actions and emotions respec-

*Corresponding author

Proceedings of the 29th International Conference on Computational Linguistics, pages 6426–6435
October 12–17, 2022.

tively in a story, especially under the guidance in the form of natural language. To crack the nut, a novel input form conducive to fine-grained control on story generation is introduced into CHAE. Concretely, we use various prompts for fine-grained control conditions in different aspects. Moreover, we design different methods for different control conditions to improve the control effect. Inspired by multi-task learning, we incorporate a character-wise emotion loss while training, thus enforcing the relevance between the characters and their emotions respectively.

The contributions of our work can be summarized as follows:

- We first take the characters with their actions and emotions of the story into account to conduct more fine-grained controllable story generation.
- We propose a model CHAE with a novel input form that helps the model control the story in various aspects, and a character-wise emotion loss to relate the characters and the corresponding emotions.
- The results of both automatic and human evaluation show that our model has strong controllability to generate customized stories.

2 Related Work

Story Generation Story generation has attracted more and more researchers to explore in recent years. There are many challenges in the task, such as context coherence and control. For context coherence, some works are devoted to introducing a series of keywords (Yao et al., 2019), outlines (Rashkin et al., 2020), or incorporating external knowledge (Guan et al., 2019, 2020; Ji et al., 2020) into the story. For style and sentiment control, Kong et al. (2021) generate stories with specified style given a leading context. However, it only focuses on the global attributes of the story. Brahman and Chaturvedi (2020) work on generating stories with desired titles and the protagonists' emotion arcs, and Xu et al. (2020) generate stories considering the changes in the psychological state, while they just control the emotion lines instead of the detailed contents.

Controllable Text Generation We have witnessed the great performance of SOTA models for text generation these years. Despite the progress in coherence and rationality of the text generated, controllability remains to be challenging, which means generating text with specific attributes, such as emotion, style, topic, format, etc. CTRL (Keskar et al., 2019) can control the overall attributes such as domain, style and topic of the generated text by adding control codes and prompts. By plugging in a discriminator, PPLM (Dathathri et al., 2020) can guide text generation without further training the language model. CoCon (Chan et al., 2020) fine-tunes an intermediate block with self-supervised learning to control high-level attributes i.e., sentiment and topic. Compared to the previous works, our work places the emphasis on more fine-grained, all-round control on the generating process, including the control of characters with their emotions and actions in the story.

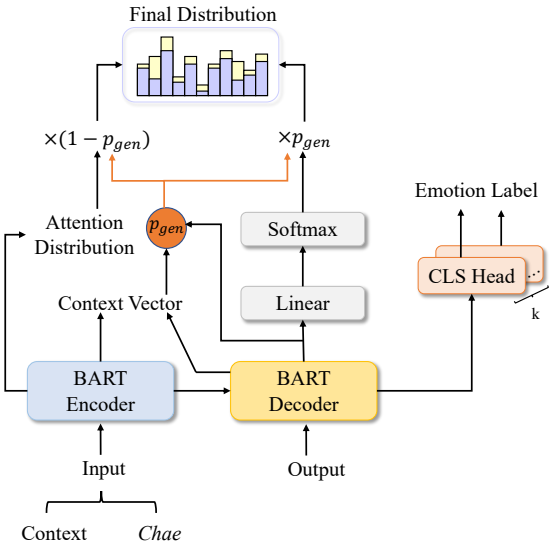

Figure 1: The architecture of CHAE. The input is the concatenation of two components, $Context$ and $Chae$, which will be further explained in Sec 3. The emotion labels are used for calculating a character-wise emotion loss to tie up the characters and their emotions respectively.

3 Methodology

3.1 Problem Formulation

The process of fine-grained controllable story generation in this work is defined as follows.

The input of the task has two components. We refer to the one as $Context$. Let $Context = (x_1, x_2, ..., x_p)$ denote the beginning sentence of the story, which will be the initial $Context$. The

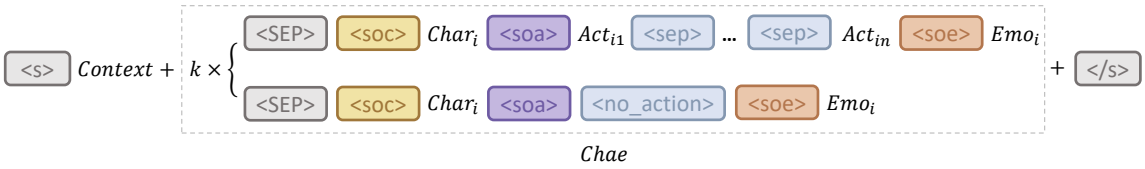

Figure 2: The input form of CHAE. The input starts with $\langle s \rangle$ and ends with $\langle /s \rangle$, comprising $Context$ and $Chae$. The latter is a sequence of k control conditions on the next sentence to be generated, and each condition controls a character. We show two possible forms of the control conditions after the brace. The special tokens in $Chae$ are further explained in Table 1.

other component is $Chae$, a sequence of k fine-grained control conditions on the next sentence. Each condition in $Chae$ is the combination of the name $Char_i$, n actions $Act_{i1}, Act_{i2}, ..., Act_{in}$, and emotion Emo_i of a character appearing in the next sentence to be generated, where n is not fixed and i is the index of the character. Note that we use Italic $Chae$ to distinguish the special input component from our model **CHAE**.

The model predicts one sentence denoted as $Y = (y_1, y_2, ..., y_q)$ at a time by estimating the conditional probability $P(Y|Context, Chae)$. Here we embody the idea of auto-regression by adopting an iterative generation strategy, that is, the sentence generated is then concatenated to $Context$ for next prediction. Especially, at training time, we concatenate the gold sentences instead of generated sentences to $Context$ incrementally like teacher forcing.

The goal of this task is to generate a story where each sentence adheres to the input condition $Chae$ in terms of character, action, and emotion, through which elevate the quality of generation in a fine-grained manner.

3.2 Model Architecture

The architecture of our model CHAE is shown in Figure 1. CHAE is built upon a BART model (Lewis et al., 2020). As mentioned, our model embodies the idea of auto-regression by adopting an iterative generation strategy. On the one hand, the iteratively updated $Chae$ helps control the content of each sentence at a detailed level of granularity. On the other hand, the strategy ensures that the model can always see the foregoing. Considering the incremental $Context$ can be extra long, we employ BART rather than GPT-2. GPT-2 is an auto-regressive model fully based on transformer decoder, while BART has a bidirectional encoder, which might make it better in understanding and encoding long input sequences. To confirm the hypothesis, we also compared BART with GPT-2 on the benchmark dataset in Sec 4, and found that BART outperformed GPT-2 in story generation.

Special tokens	Meaning
$\langle SEP \rangle$	The start token of a condition.
$\langle soc \rangle$	The start token of a character's name.
$\langle soa \rangle$	The start token of actions.
$\langle soe \rangle$	The start token of an emotion.
$\langle sep \rangle$	The start token of a single action.
$\langle no\_action \rangle$	The token representing no action.

Table 1: The meanings of the special tokens in $Chae$.

3.3 Generation Based on Fine-Grained Control Conditions

To generate a story with the characters, their actions and emotions specified, we need to remind the BART model of the elements controlled currently from time to time. Inspired by the practice of leveraging special tokens for controllable generation (Fang et al., 2021; Keskar et al., 2019; Tsutsui and Crandall, 2017), we propose a novel form of input (titled $Chae$), which is a sequence of k fine-grained control conditions on the next sentence to be generated. Each condition in $Chae$ controls a character, and each segment in the condition controls an element (i.e., character's name, action, and emotion) of the corresponding character. Note that any number of actions can be assigned in a condition. The nested sequence form of $Chae$ facilitates the neat combination of various fine-grained control conditions.

We design several special tokens and add them between each segment as the control prompts (see Figure 2). In this study, 6 special tokens are used to prompt the model. They are $\langle SEP \rangle$, $\langle soc \rangle$, $\langle soa \rangle$, $\langle soe \rangle$, $\langle sep \rangle$, and $\langle no\_action \rangle$. The meanings of the tokens are shown in Table 1.

Then, we encode the input $Context$ and $Chae$

as h_{enc} by the BART encoder:

$$\boldsymbol{h}_{enc} = \mathbf{Enc}(\boldsymbol{e}_c), \tag{1}$$

where $\boldsymbol{e}_c$ is the embedding of $Context$ and $Chae$. The vocab generation probability is calculated by the BART decoder as:

$$\begin{aligned} P_{voc}(y) &= P(y_t|y_{<t}, Context, Chae) \\ &= \text{softmax}(\boldsymbol{W}_{voc}\boldsymbol{h}_{dec}), \end{aligned} \tag{2}$$

$$\boldsymbol{h}_{dec} = \mathbf{Dec}(\boldsymbol{e}_{y<t}, \boldsymbol{h}_{enc}), \tag{3}$$

where $\boldsymbol{e}_{y<t}$ is the embeddings of the generated tokens before timestep t, and $\boldsymbol{W}_{voc}$ is a trainable parameter.

After fine-tuning with these special tokens, the model is aware of the elements controlled by each segment of each condition.

3.4 Improvement in Control Effect

3.4.1 Character and Action Control

For characters and their actions, we expect to see the characters and their actions in current $Chae$ appear in the coming sentence. Inspired by the usage of copy mechanism (See et al., 2017; Deaton et al., 2019; Prabhu and Kann, 2020) in copying significant tokens from the input sequences, we add a copy pointer to BART for the information in $Chae$. The attention distribution on $Chae$ denoted by $\widetilde{a}$ is attained by averaging the multiple heads in the cross attention block of BART decoder:

$$\widetilde{a} = \frac{\sum_{i=1}^{h} a_i}{h}, \tag{4}$$

where h is the number of the attention heads, and a_i denotes the attention distribution on $Chae$ from the i-th attention head.

When generating stories, we first combine the hidden state of the decoder $\boldsymbol{h}_{dec}$, the context vector $\boldsymbol{h}_{con}$, and the embedding of the decoder input $\boldsymbol{e}_y$. Secondly, we calculate a generation probability p_{gen}, which is a soft switch to choose a word from the vocabulary according to P_{voc}, or to copy a word from $Chae$ by sampling from the mean attention distribution $\widetilde{a}$. The final distribution of a word can be represented as follows:

$$P(y) = p_{gen}P_{voc}(y) + (1 - p_{gen}) \sum_{j:y_j=y} \widetilde{a}_j, \tag{5}$$

$$p_{gen} = \sigma(\boldsymbol{W}_p^{\top}[\boldsymbol{h}_{dec}; \boldsymbol{h}_{con}; \boldsymbol{e}_y]), \tag{6}$$

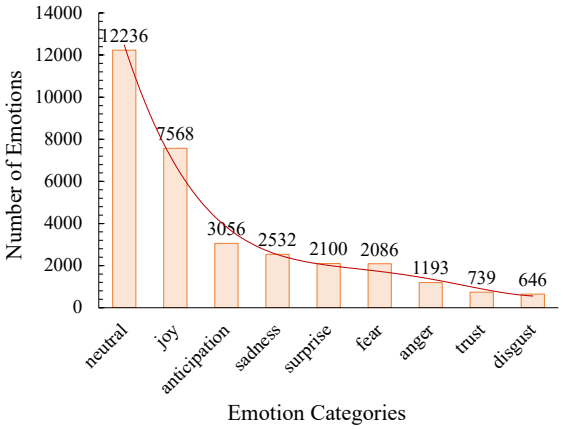

Figure 3: Statistics of emotion categories of sentences in all stories.

where $\boldsymbol{W}_p$ is a trainable parameter. In this way, the characters and actions in $Chae$ will be produced with higher probability, and the output can be flexibly changed according to the different input.

3.4.2 Character-Wise Emotion Control

For characters' emotions, we additionally incorporate a character-wise emotion loss, by which the model is forced to generate sentences with specified emotions tied up with corresponding characters. We add k emotion classification heads to the top of the decoder output layer (Ide and Kawahara, 2021), and k is equivalent to the number of conditions in $Chae$. It provides direct supervision on emotion control to predict the emotion of every character in the story.

However, as shown in Figure 3, the emotion categories present a long tail distribution. To relieve the class-imbalance problem, we use a Weighted Cross-Entropy (WCE) loss between the predicted emotion distribution P_{emo} and the emotion labels l_e:

$$L_{EMO} = -\alpha_e l_e \log(P_{emo}), \tag{7}$$

$$P_{emo} = \text{softmax}(\boldsymbol{W}_{emo}\boldsymbol{h}_{dec}), \tag{8}$$

$$\alpha_e = \frac{N}{e * \text{count}(l_e)}, \tag{9}$$

where $\boldsymbol{W}_{emo}$ is a trainable parameter, α_e denotes the weights of emotion classification labels. N is the number of the training samples, e is the number of emotion categories, and count$(\cdot)$ is a function to calculate the number of samples in each emotion category. Now, the model has both explicit control from emotion in $Chae$ and implicit control from

the emotion loss to generate sentences with target emotions.

3.5 Objective Function

We minimize the Negative Log-Likelihood (NLL) loss of the target story sentence Y with the input $Context$ and $Chae$:

$$L_{NLL} = -\sum_{t=1}^{T} \log P(y_t|y_{<t}, Context, Chae), \tag{10}$$

The total loss L is as follow:

$$L = L_{NLL} + \lambda L_{EMO}, \tag{11}$$

where λ is a hyper-parameter. After training with the above two objectives, our model can generate fluent stories under desired conditions.

4 Experiments

4.1 Dataset

We use ROCStory with labeled characters' emotions and actions as our dataset (Rashkin et al., 2018), which contains 14738 five-sentence stories, including 9885 stories for training, 2483 stories for validation and 2370 stories for testing. To conform to the iterative generation, we divide the stories into 39540 / 9932 / 9480 sentence pairs for training/validation/testing. Each pair of sentences consists of two adjacent sentences in the story. The characters with their actions and emotions in the dataset are labeled by three crowdsourced workers from Amazon Mechanical Turk. The emotions come from Plutchik psychology theory (Plutchik, 1980), including 8 species, such as "joy", "anger", etc. Most actions represent the character's underlying motivations, and they generally take the infinitive such as "to win all games" and "to have fun".

In preprocessing, we integrate the annotations of the three workers, and take the emotions with the highest confidence as the final labels. However, we still notice that some emotion labels have excessively low confidence, indicating subtle emotion tendencies. We modify them to "neutral" to avoid distortion of the emotions. Moreover, we find that 9885 stories in the training set are not labeled with emotions, but only with actions. As a remedy, we fine-tune the model on these 9885 stories to keep the fluency of the story. Later, we re-divide all the stories with labeled emotions in the validation set and testing set into another 3 splits. Finally, we fine-tune the model on the new training set to capture emotional information. The re-divided dataset includes 15528 / 1944 / 1940 sentence pairs by 8:1:1 for training/validation/testing.

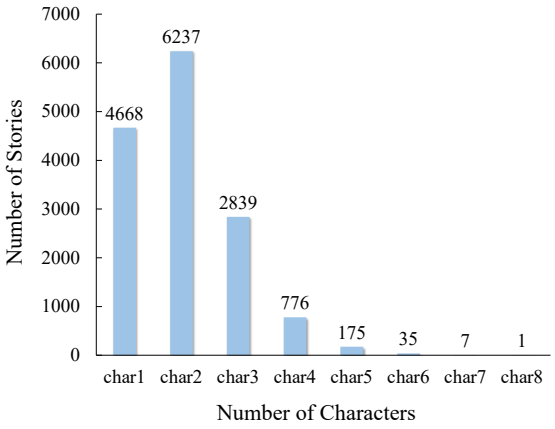

Figure 4: Statistics of stories with different numbers of characters. The "char1" means that the stories have one character.

4.2 Baselines

We compare our model with a carefully selected set of baselines as shown below.

GPT-2 (Radford et al., 2019): GPT-2 is a transformer-based model pre-trained on a very large corpus, which is very commonly used in natural language generation. A lot of works witness a good performance of GPT-2 in dialogue, story and other text generation in recent years, demonstrating its auto-regression quality.

BART (Lewis et al., 2020): BART is a transformer-based seq2seq pre-training model with a bidirectional (BERT-like) encoder and an autoregressive (GPT-like) decoder. A lot of text generation tasks, like neural machine translation and automatic summarization, can achieve effective results by fine-tuning on BART.

SoCP (Xu et al., 2020): Stories with multi-characters and multi-psychology generated by SoCP can change with the emotional lines of assigned characters. In addition, SoCP can generate stories with different emotional intensities. It also designe a metric to evaluate the accuracy of controlling emotions of roles.

Stylized-Story-Generation (SSG) (Kong et al., 2021): SSG can generate stories with specified style given a leading context by first planning the stylized keywords and then generating the whole story with the guidance of the keywords. Two story styles are considered in SSG, including emotion-

Models	PPL ↓	B-1 ↑	B-2 ↑	D-1 ↑	D-2 ↑	ACC ↑
SoCP	101.33	22.93	7.32	0.478	0.650	0.893
SSG	15.22	26.74	10.77	0.573	0.904	-
GPT-2	29.21	21.30	6.35	0.744	0.960	-
BART	14.00	24.15	7.93	0.729	0.964	-
CHAE	**11.58**	27.10	10.20	0.750	0.971	**0.941**
w/o copy	11.65	26.53	9.67	**0.754**	**0.972**	0.879
w/o emo	11.75	**27.51**	10.61	0.732	0.965	-
w/o copy w/o emo	11.70	27.19	**10.79**	0.735	0.967	-

Table 2: Automatic metrics.

Setting	B-1 ↑	B-2 ↑	D-1 ↑	D-2 ↑
Greedy	31.08	15.65	0.586	0.797
Beam=2	31.96	15.61	0.595	0.811
Beam=3	**32.04**	**15.89**	0.584	0.804
Beam=4	31.86	15.84	0.578	0.801
Beam=5	31.60	15.40	0.576	0.802
Top-k=30, Temperature=0.8	30.86	13.63	0.697	0.939
Top-k=50, Temperature=0.8	30.58	13.64	0.702	0.939
Top-k=30, Temperature=1	29.36	12.00	0.720	0.957
Top-k=50, Temperature=1	29.22	11.94	0.730	0.962
Top-k=30, Temperature=1.2	27.68	10.49	0.745	0.968
Top-k=50, Temperature=1.2	27.10	10.20	**0.750**	**0.971**

Table 3: Decoding strategies adjustment.

driven and event-driven stories.

4.3 Implementation Details

We build our model based on BART using the Huggingface's Transformers library in Pytorch (Wolf et al., 2019). We initialize our model with the public checkpoint of bart-large-cnn [1]. The batch size during training is 8. We use the AdamW optimization (Loshchilov and Hutter, 2019) with $\beta_1 = 0.9$, $\beta_2 = 0.999$ and the initial learning rate is $5e-5$. According to the statistics of characters in the stories shown in Figure 4, most stories contain two characters, so we fix the number of characters in each story k to 2. The hyper-parameter λ defaults to 1.0.

For all models, We generate stories by using top-k sampling (Fan et al., 2018) with $k = 50$ and a softmax temperature of 0.8.

4.4 Automatic Evaluation

Evaluation Metrics We use the following metrics for automatic evaluation: (1) **Perplexity (PPL)**: PPL represents the general quality of the generated stories, which estimates the probability of sentences according to each word. (2) **BLEU (B-n)** (Papineni et al., 2002): We use BLEU to compare the coverage n-gram in the candidate stories and the reference stories because the words in $Chae$ usually appear in the reference stories. (3) **Distinct (D-n)**: (Li et al., 2016): Distinct is used to evaluate generation diversity by calculating the percentage of unique n-grams. (4) **Accuracy of emotions (ACC)**: We use emotion labels to calculate the accuracy of the emotions of generated sentences to reflect the controllability of emotion.

Results Table 2 shows the automatic evaluation results. All baseline models are trained on our dataset ROCStory. Our model achieves the lowest Perplexity, which reflects the high quality of the stories generated by CHAE. Besides, the BART-based models (BART and SSG) are better than the GPT-2 model. The traditional seq2seq model SoCP has worse performance. The greater BLEU scores of CHAE and its varieties imply stories closer to the golden truth, proving that the introduction of fine-grained control, mainly by means of the special input $Chae$, is conducive to improving the quality of the generated stories. As for Distinct, the pre-trained models show excellent performance compared with the traditional seq2seq model. Our model attains the best score, which demonstrates CHAE's ability in generating more diverse stories. We further compare the ACC with SoCP, and our model gets higher emotion accuracy.

Additionally, we also explore some decoding strategies including greedy search, beam search, top-k, and temperature, as shown in Table 3. The BLEU scores reach the best when we use the beam size equal to 3, while we get the best Distinct scores with the top-k equal to 3 and the temperature equal to 1.2. The BLEU scores decrease with the increase of beam size. The BLEU scores with top-k are lower than beam search, but the Distinct scores are higher. When top-k is fixed, the higher the temperature, the lower the BLEU and the higher the Distinct. When the temperature is fixed, the higher the top-k, the lower the BLEU and the higher the Distinct.

4.5 Ablation Studies

As shown in Table 2, we also conduct the ablation studies to verify the effectiveness of the additional control methods in our model. When we remove the copy mechanism **(w/o copy)**, the BLEU decreases and the Diversity increases, which suggests that the copy mechanism is beneficial in controlling content, but it also affects the diversity at the

[1] https://huggingface.co/facebook/bart-large-cnn/tree/main

Models	Fluency			Coherence			Informativeness		
	Win(%)	**Lose**(%)	**Tie**(%)	**Win**(%)	**Lose**(%)	**Tie**(%)	**Win**(%)	**Lose**(%)	**Tie**(%)
CHAE vs. SSG	**56.0**	30.0	14.0	**52.0**	30.0	18.0	**90.0**	6.0	4.0
CHAE vs. BART	**40.0**	40.0	20.0	**38.0**	36.0	26.0	**46.0**	42.0	12.0
CHAE vs. GPT2	**38.0**	36.0	26.0	**40.0**	40.0	20.0	**72.0**	12.0	16.0
Controllability of CHAE	**65.0%**								

Table 4: Human evaluation results. It shows the percentage of win, lose and tie of CHAE compared with other baselines. The controllability of CHAE indicates the average percentage of sentences in a story that can be controlled by control condition $Chae$.

Context	Jessica had to go to the city.
SoCP	She was very excited to see a new . She was very proud of her friends . She was very happy . She was happy she was going to get .
GPT-2	She left her friends and their cars behind. They waited outside the station for her. She knew she'd go to a bar one day. Unfortunately they all stayed away.
BART	She told her mom to take a bus. The bus didn't have enough time to get back to the stations. When she came back her mother was upset. Jessica was upset that everyone didn't believe her to go now.
SSG	She was leaving the bus at ten o'clock. She saw a traffic light coming up in the distance. But she got in the way quickly. She was late late to her bus stop so she had to wait.
Chae	1. $\langle SEP \rangle$ $\langle soc \rangle$ Jessica $\langle soa \rangle$ to learn new things $\langle sep \rangle$ to see the **museums** $\langle sep \rangle$ to learn something $\langle soe \rangle$ **joy** 2. $\langle SEP \rangle$ $\langle soc \rangle$ Jessica $\langle soa \rangle$ to go out to some interesting place $\langle soe \rangle$ joy 3. $\langle SEP \rangle$ $\langle soc \rangle$ Jessica $\langle soa \rangle$ to save the **artwork** $\langle sep \rangle$ to **remember** what she had seen $\langle soe \rangle$ joy 4. $\langle SEP \rangle$ $\langle soc \rangle$ **Jessica** $\langle soa \rangle$ to **photoshoot** $\langle sep \rangle$ **to see the artistic beauty** inside the famous museum $\langle soe \rangle$ joy
CHAE	She had always **enjoyed** going to a **museum**. However, she could not walk with better to the **museum**. She had to **remember** all her **paintings** from this one place. **Jessica noticed an amazing view** that her **camera** had of a gallery.
Gold	She went to some museums. She really enjoyed the artwork at the MET. She took a lot of photographs. She was there for hours.

Table 5: Case study of the generated stories by our model and the baselines. The $Chae$ represents the assigned control conditions including characters, actions and emotions.

same time. In addition, we also observed that the copy mechanism can help the model to improve the accuracy of emotions. On the contrary, when we remove the emotion loss **(w/o emo)**, the BLEU increases and the Diversity decreases, which reflects that the emotion loss does improve the diversity of the stories, but sacrifices controllability. Furthermore, we remove both the copy mechanism and the emotion loss **(w/o copy w/o emo)**, which means just introducing the conditions $Chae$ to the vanilla BART. The results are still better than BART, illustrating the benefit brought by the fine-grained control conditions.

In general, the copy mechanism and the emotion loss complement each other. The results show that the integrated model (CHAE) can obtain good scores and relatively balance on BLEU and Diversity, and has the best performance on PPL and ACC.

4.6 Human Evaluation

We conduct a human evaluation to compare CHAE with baselines on the following three metrics. (1) **Fluency**: The fluency of a sentence can reflect the quality of intra-sentence. (2) **Coherence**: The coherence of the story can reflect the cohesion of context and the quality of inter-sentence. (3) **Informativeness**: The good performance of a story in terms of informativeness indicates that there are a variety of rich words in the story. We recruit six annotators and divided them into two groups to annotate 50 stories randomly sampled. Each story is annotated by three workers to ensure fairness. The workers have two tasks: one is to compare the results generated by CHAE and other baselines, and the other is to score the controllability of CHAE according to whether the predicted sentences adhere to the assigned conditions $Chae$. The majority

Context	A polite thief was making robberies in the small town.
Chae1	$\langle SEP \rangle$ $\langle soc \rangle$ People $\langle soa \rangle$ $\langle no\_action \rangle$ $\langle soe \rangle$ fear $\langle SEP \rangle$ $\langle soc \rangle$ **Man** $\langle soa \rangle$ to catch the thief $\langle soe \rangle$ **anger**
Result1	One day, a **man** walked up to him and **asked him to stop** .
Chae2	$\langle SEP \rangle$ $\langle soc \rangle$ People $\langle soa \rangle$ $\langle no\_action \rangle$ $\langle soe \rangle$ fear $\langle SEP \rangle$ $\langle soc \rangle$ **Man** $\langle soa \rangle$ $\langle no\_action \rangle$ $\langle soe \rangle$ **joy**
Result2	The **man** who was supposed to stop him was a **nice man** .
Chae3	$\langle SEP \rangle$ $\langle soc \rangle$ People $\langle soa \rangle$ $\langle no\_action \rangle$ $\langle soe \rangle$ fear $\langle SEP \rangle$ $\langle soc \rangle$ **Tom** $\langle soa \rangle$ **to catch the thief** $\langle soe \rangle$ **anger**
Result3	**Tom** decided to investigate and **caught the thief** .
Chae4	$\langle SEP \rangle$ $\langle soc \rangle$ **People** $\langle soa \rangle$ **call the police** $\langle soe \rangle$ **fear** $\langle SEP \rangle$ $\langle soc \rangle$ **Tom** $\langle soa \rangle$ **call the police** $\langle soe \rangle$ **anger**
Result4	**Tom called the police** and **they** **told him** to **call the police** .

Table 6: Case study of controllability.

votes among the annotators will be the final decisions for the first task and we average the scores of all annotators for the second task. The results are shown in Table 4. Our model achieves the best performance in each metric compared with baselines and has 65% controllability.

4.7 Case Study

Comparison Table 5 shows the stories generated from our model and the baselines. The story generated by the SoCP model reflects that the results generated by the models built on the seq2seq architecture are usually simple, short and less informative. The transformer-based models, such as GPT-2, BART, and SSG, show strong generative capabilities, which can generate coherent and informative stories. However, the content of these stories can not be controlled, and the characters' emotions and actions in the stories are not obvious.

For our model, we can specify the characters, actions and emotions in the story that we desired, and the model can generate stories based on this information, i.e. $Chae$. The meaning of the four $Chae$ is to control the content of the next four sentences respectively. The first sentence reflects that the character's action is to go to the "museum", and the word "enjoyed" reflects the emotion of "joy". The word "remember" and "artwork" in the third sentence corresponds to "remember" and "paintings" in $Chae3$. The model is connected to the "camera" from the "photoshoot" in $Chae4$, and the phrase "noticed an amazing view" corresponds to "to see the artistic beauty", and the character "Jessica" appears in the sentence. The results show that our model can generate coherent and informative stories according to the control information.

Controllability Table 6 shows the examples of controllability. We use beam search (beam=2) to generate results instead of top-k, because the top-k method usually generates diverse words, which can not ensure controllability. Our model generates sentences based on the same context and the given different $Chae$. The first two examples show that for the same character "Man", given the emotion "anger" and "joy", the model can generate sentences with corresponding emotions, such as the phrase "asked him to stop" and "he was a nice man". In the third example, we change the character to "Tom", and the generated sentence is also changed from "Man" to "Tom", and the phrase "caught the thief" also reflects the action "to catch the thief". In the last example we change Tom's action to "call the police", and the resulting change from "caught the thief" to "called the police". Also, we set people's actions as "call the police" and the emotion is fear, which results in the expression "they told him to call the police." The above reflects that our model has a good control effect on characters, actions and emotions.

5 Conclusion and Future Work

Through our model CHAE, we can create stories with fine-grained control according to the specified characters and corresponding actions and emotions, which is more convenient for practical applications, such as the creation of script novels, providing inspiration for screenwriters, and even acting as screenwriters in the future.

However, our model also has some disadvantages: (1) The dataset contains only stories with 5 sentences, which is not enough for learning to

generate longer stories. (2) The training of our model heavily relies on the annotations of characters, emotions, and actions in the dataset, while it is very expensive to obtain the annotated data. (3) The $Chae$ in the dataset actually has some noise, i.e., some descriptions in $Chae$ are not reflected in the corresponding sente nces. (4) Our iterative generation method will result in a long time to train a story and may cause a cascade problem, which affects the overall quality of story generation.

In future work, we will adopt datasets with much more data and longer stories, such as **WritingPrompts** (Fan et al., 2018) and **WikiPlots**[2]. In addition, we will consider using commonsense reasoning techniques to reason about the emotions and actions of the characters in the story before further generating the story. Regarding the noise of $Chae$, we plan to conduct denoising in preprocessing to filter out the samples whose $Chae$ are inconsistent with the corresponding sentence. The problem can also be alleviated by dynamically controlling the weight of the conditions. We are also further exploring more convenient and effective training methods to generate controllable stories by inputting control conditions in one go, rather than iterative generation.

Acknowledgements

The work is partially supported by the National Nature Science Foundation of China (No. 61976160, 61906137, 61976158, 62076184, 62076182) and Shanghai Science and Technology Plan Project (No.21DZ1204800) and Technology research plan project of Ministry of Public and Security (Grant No.2020JSYJD01).

References

Amal Alabdulkarim, Siyan Li, and Xiangyu Peng. 2021. Automatic story generation: Challenges and attempts. In *Proceedings of the Third Workshop on Narrative Understanding*, pages 72–83, Virtual. Association for Computational Linguistics.

Faeze Brahman and Snigdha Chaturvedi. 2020. Modeling protagonist emotions for emotion-aware storytelling. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 5277–5294, Online. Association for Computational Linguistics.

Alvin Chan, Yew-Soon Ong, Bill Pung, Aston Zhang, and Jie Fu. 2020. Cocon: A self-supervised approach for controlled text generation. In *International Conference on Learning Representations*.

Sumanth Dathathri, Andrea Madotto, Janice Lan, Jane Hung, Eric Frank, Piero Molino, Jason Yosinski, and Rosanne Liu. 2020. Plug and play language models: A simple approach to controlled text generation. In *International Conference on Learning Representations*.

Jon Deaton, Austin Jacobs, Kathleen Kenealy, and Abigail See. 2019. Transformers and pointer-generator networks for abstractive summarization.

Angela Fan, Mike Lewis, and Yann Dauphin. 2018. Hierarchical neural story generation. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 889–898, Melbourne, Australia. Association for Computational Linguistics.

Le Fang, Tao Zeng, Chaochun Liu, Liefeng Bo, Wen Dong, and Changyou Chen. 2021. Outline to story: Fine-grained controllable story generation from cascaded events. *arXiv preprint arXiv:2101.00822*.

Jian Guan, Fei Huang, Zhihao Zhao, Xiaoyan Zhu, and Minlie Huang. 2020. A knowledge-enhanced pretraining model for commonsense story generation. *Transactions of the Association for Computational Linguistics*, 8:93–108.

Jian Guan, Yansen Wang, and Minlie Huang. 2019. Story ending generation with incremental encoding and commonsense knowledge. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, pages 6473–6480.

Tatsuya Ide and Daisuke Kawahara. 2021. Multi-task learning of generation and classification for emotion-aware dialogue response generation. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Student Research Workshop*, pages 119–125, Online. Association for Computational Linguistics.

Haozhe Ji, Pei Ke, Shaohan Huang, Furu Wei, Xiaoyan Zhu, and Minlie Huang. 2020. Language generation with multi-hop reasoning on commonsense knowledge graph. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 725–736.

Nitish Shirish Keskar, Bryan McCann, Lav R Varshney, Caiming Xiong, and Richard Socher. 2019. Ctrl: A conditional transformer language model for controllable generation. *arXiv preprint arXiv:1909.05858*.

Xiangzhe Kong, Jialiang Huang, Ziquan Tung, Jian Guan, and Minlie Huang. 2021. Stylized story generation with style-guided planning. In *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*, pages 2430–2436, Online. Association for Computational Linguistics.

[2] https://github.com/markriedl/WikiPlots

Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Veselin Stoyanov, and Luke Zettlemoyer. 2020. BART: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7871–7880, Online. Association for Computational Linguistics.

Jiwei Li, Michel Galley, Chris Brockett, Jianfeng Gao, and Bill Dolan. 2016. A diversity-promoting objective function for neural conversation models. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 110–119, San Diego, California. Association for Computational Linguistics.

Ilya Loshchilov and Frank Hutter. 2019. Decoupled weight decay regularization. In *7th International Conference on Learning Representations, ICLR 2019, New Orleans, LA, USA, May 6-9, 2019*. OpenReview.net.

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. Bleu: a method for automatic evaluation of machine translation. In *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics*, pages 311–318, Philadelphia, Pennsylvania, USA. Association for Computational Linguistics.

Robert Plutchik. 1980. A general psychoevolutionary theory of emotion. *Theories of emotion*, 1:3–31.

Nikhil Prabhu and Katharina Kann. 2020. Making a point: Pointer-generator transformers for disjoint vocabularies. In *Proceedings of the 1st Conference of the Asia-Pacific Chapter of the Association for Computational Linguistics and the 10th International Joint Conference on Natural Language Processing: Student Research Workshop*, pages 85–92.

Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. 2019. Language models are unsupervised multitask learners. *OpenAI Blog*, 1(8):9.

Hannah Rashkin, Antoine Bosselut, Maarten Sap, Kevin Knight, and Yejin Choi. 2018. Modeling naive psychology of characters in simple commonsense stories. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 2289–2299, Melbourne, Australia. Association for Computational Linguistics.

Hannah Rashkin, Asli Celikyilmaz, Yejin Choi, and Jianfeng Gao. 2020. PlotMachines: Outline-conditioned generation with dynamic plot state tracking. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 4274–4295, Online. Association for Computational Linguistics.

Abigail See, Peter J. Liu, and Christopher D. Manning. 2017. Get to the point: Summarization with pointer-generator networks. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1073–1083, Vancouver, Canada. Association for Computational Linguistics.

Satoshi Tsutsui and David J. Crandall. 2017. Using artificial tokens to control languages for multilingual image caption generation. *CoRR*, abs/1706.06275.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Rémi Louf, Morgan Funtowicz, et al. 2019. Huggingface's transformers: State-of-the-art natural language processing. *arXiv preprint arXiv:1910.03771*.

Feifei Xu, Xinpeng Wang, Yunpu Ma, Volker Tresp, Yuyi Wang, Shanlin Zhou, and Haizhou Du. 2020. Controllable multi-character psychology-oriented story generation. In *Proceedings of the 29th ACM International Conference on Information & Knowledge Management*, pages 1675–1684.

Lili Yao, Nanyun Peng, Ralph Weischedel, Kevin Knight, Dongyan Zhao, and Rui Yan. 2019. Plan-and-write: Towards better automatic storytelling. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, pages 7378–7385.

Chinese Couplet Generation with Syntactic Information

Yan Song
University of Science and Technology of China
clksong@gmail.com

Abstract

Chinese couplet generation aims to generate a pair of clauses (usually generating a subsequent clause given an antecedent one) with certain rules (e.g., morphological and syntactical symmetry) adhered and has long been a challenging task with cultural background. To generate high-quality couplet (antecedent) clauses, it normally requires a model to learn the correspondences between antecedent and subsequent clauses under aforementioned rules and constraint of few characters with their concise usage. To tackle this task, previous studies normally directly adopt deep neural networks without explicitly taking into account fine-grained analysis of the clauses, in this paper, we propose to enhance Chinese couplet generation by leveraging syntactic information, i.e., part-of-speech (POS) tags and word dependencies. In doing so, we identify word boundaries in the antecedent clause and then use a special attention module to encode the syntactic information over the words for better generating the subsequent clause. Experimental results on a dataset for Chinese couplet generation illustrate the validity and effectiveness of our approach, which outperforms strong baselines with respect to automatic and manual evaluation metrics.[1]

1 Introduction

A Chinese antithetical couplet is composed of a pair of two sentences (i.e., an antecedent and a subsequent clause) with particular meaning that are usually applied to festivals or certain circumstances, which can be seen as a special type of poetry with a history in China of more than one thousand years.[2] Couplets are normally concise yet have profound and expressive ability to convey a variety of meanings, thoughts and emotions with limited number of characters. Among all interesting characteristics, the most unique one of couplets is their dueling pattern that the antecedent and subsequent clauses should have an one-to-one correspondence adhere to strict rules covering multiple aspects including tone, length, word usage and even syntax.

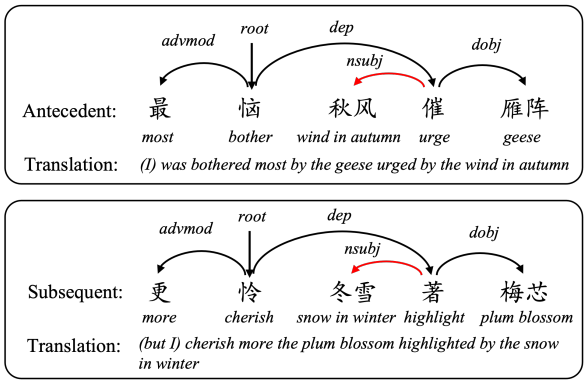

Figure 1: An example of a Chinese couplet pair marked with word dependencies and English translations.

For automatic Chinese couplet generation, recent studies follow the conventional encoder-decoder paradigm (Sutskever et al., 2014), with an general attention mechanism to capture the correspondence between the antecedent and subsequent clauses (Zhang et al., 2018; Fan et al., 2019; Gao et al., 2021). Although satisfactory results are obtained via their approaches, reasonable granularity matching and explicit structural correspondence are still worthy of improvement with special treatment. One important reason is that conventional approaches apply character-level encoding and decoding, larger granular text units are omitted in their models and thus led to the incapability of further syntactic analysis. Note that not all words in ancient Chinese are in the form of single-character, it is also applied to couplets that identifying boundaries of longer text spans is useful for their better interpretation, as shown in Figure 1 that the antecedent and subsequent clauses contains two-

[1] Related code and resources are available at https://github.com/synlp/ChiCoupletGen.

[2] People started to write paired sentences in *Pre-Qin* period over two thousand years ago, such as those ones widely used in 诗经 (*Classic of Poetry*) around 500 B.C. The first "specialized" couplet emerged as recorded in 蜀祷杌 (*Shu Tao Wu*), which was composed in the *Song* Dynasty around 964 A.D.

Proceedings of the 29th International Conference on Computational Linguistics, pages 6436–6446
October 12–17, 2022.

character words and their boundaries exactly match between clauses. Similarly, for the structural correspondence, the syntactic rule over the antecedent clause should be identical to the subsequent one, where to the best of our knowledge there is no previous study that focus on modeling couplet generation from this aspect. The example in Figure 1 also illustrates the syntactic structures of the clauses, with same word dependency patterns applied to them. For example, in the antecedent clause, the third word "秋风" (*wind in autumn*) is the nominal object (*nsubj*) of the predicate "催" (*urge*), and in the subsequent clause, the word "冬雪" (*snow in winter*) at the same position is also the nominal object of the predicate "著" (*highlight*) (marked by red arcs). As a result, enhancing Chinese couplet generation is expected to be done by identifying appropriate word boundaries and then building explicit syntactic correspondences over them[3] between the antecedent and subsequent clauses.

In this paper, we propose an approach to Chinese couplet generation with syntactic information over the identified words in the clauses, where the designed model not only provides a better understanding for couplets, but also enhances the generation process with more structural constraints. Specifically, we firstly learn word boundaries and then propose a special attention module to encode part-of-speech (POS) tags and word dependencies over the identified words in the antecedent clause and integrate such syntactic information into subsequent clause generation. Experimental results on a dataset for Chinese couplet generation demonstrate the effectiveness of our approach, which outperforms strong baselines on both automatic and manual evaluation metrics. Further analyses confirm the ability of our approach in generating couplets with neatness and artistic conception.

2 Preliminaries

Conventionally, Chinese couplet generation is regarded as a sequence-to-sequence text generation task (Fan et al., 2019; Zhang et al., 2018; Gao et al., 2021; Wang et al., 2021), whose objective is to generate character $\widehat{y}_t$ at the time step t with the given antecedent clause $\mathcal{X} = x_1, \cdots, x_t, \cdots, x_n$ (there are n characters in $\mathcal{X}$ and the t-th character is denoted as x_t) and the generated subsequent clause $\widehat{\mathcal{Y}}_{t-1} = \widehat{y}_1, \cdots, \widehat{y}_{t-1}$. The entire generation process is formalized as

$$\widehat{y}_t = \arg\max_{y_t \in \mathcal{V}} p(y_t | \widehat{\mathcal{Y}}_{t-1}, \mathcal{X}) \quad (1)$$

where p denotes the probability of a character y_i in the vocabulary $\mathcal{V}$ given $\widehat{\mathcal{Y}}_{t-1}$ and $\mathcal{X}$.

In general, to compute p, an encoder f_e is firstly used to encode the antecedent clause $\mathcal{X}$ through

$$\mathbf{h}_1, \cdots, \mathbf{h}_t, \cdots, \mathbf{h}_n = f_e(\mathcal{X}) \quad (2)$$

where $\mathbf{h}_t$ is the encoded hidden vector for the character x_t. Then, a decoder f_d takes all generated characters (i.e., $\widehat{\mathcal{Y}}_{t-1}$) at time step t, as well as all hidden vectors obtained from the encoder, and obtain the output vector $\mathbf{o}_t$ via

$$\mathbf{o}_t = f_d(\widehat{\mathcal{Y}}_{t-1}, \mathbf{h}_1, \cdots, \mathbf{h}_n) \quad (3)$$

Afterwards, $\mathbf{o}_t$ is mapped to the output space through a fully connected layer, where a $softmax$ function is further applied to the resulting vector to obtain the probability distribution over the vocabulary (i.e., $p(y_t | \widehat{\mathcal{Y}}_{t-1}, \mathcal{X})$ in Eq. (1)).

3 The Proposed Approach

In order to generate high-quality clauses that satisfies the the structure constrains of Chinese couplets, we propose an approach of neural text generation model enhanced by leveraging syntactic information. Figure 2 illustrates the overall architecture of our approach for Chinese couplet generation following the convention sequence-to-sequence paradigm, where the word boundary learning process and a special attention module to leverage syntactic information are presented on the top-left and top-right parts, respectively. In the following text, we firstly illustrate the word boundary learning process and then elaborate the proposed attention mechanism for syntactic information encoding.

3.1 Word Boundary Learning

Most existing approaches (Zhang et al., 2018; Fan et al., 2019; Gao et al., 2021) for Chinese couplet generation employ character-based encoding and decoding. However, consider that text in larger granularity (e.g., words) tend to deliver intact meanings in many cases, as well as the fact that not all characters in Chinese couplets should be treated as single-character words, character-level encoding has the limitation in losing particular semantic information and the risk of leading to inferior results for Chinese couplet generation.

To address this limitation, we propose to enhance

[3] Another key reason that we parse on words instead of characters is that existing parses are normally trained on words.

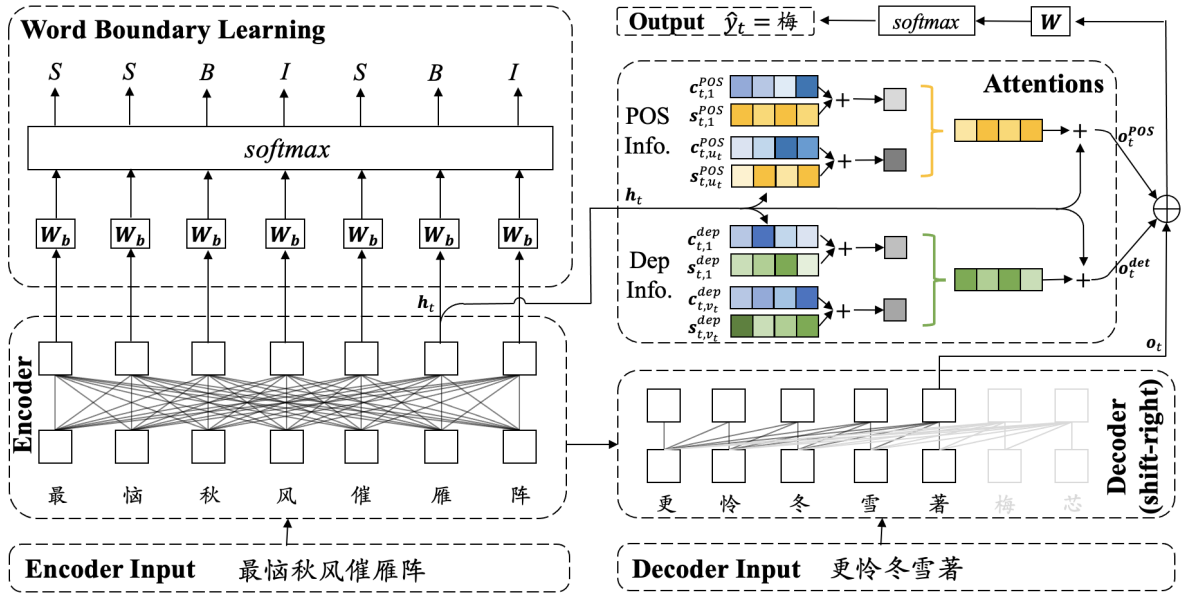

Figure 2: The overall architecture of our proposed approach for Chinese couplet generation, with an example input antecedent clause (i.e., the encoder input) and its corresponding subsequent clause (i.e., the decoder input) being generated at the sixth step on the character "梅". The process to learn word boundaries is illustrated on the top-left part; the special attention module used to encode the POS information (POS Info.) and dependency information (Dep Info.) associated with the current character (i.e., the six-th character "雁") is illustrated on the top-right part.

Chinese couplet generation with a pre-processing of learning couplet clauses in different granularities, i.e., both characters and words, which is performed by identifying word boundaries in the antecedent. Specifically, we firstly segment $\mathcal{X}$ into several words[4], where each character x_t is assigned a word boundary label y_t^b based on the position of the character in the word[5]. Then, for each character x_t, we take its hidden vector $\mathbf{h}_t$ obtained from the encoder and feed it into a fully connected layer with $softmax$ activation function through

$$\mathbf{p}_t^b = softmax(\mathbf{W_b} \cdot \mathbf{h}_t + \mathbf{b}_b) \quad (4)$$

where $\mathbf{W_b}$ and $\mathbf{b}_b$ are trainable matrix and bias vector in the fully connected layer and $\mathbf{p}_b$ is the probability distribution vector over the the word boundary label set with the value at each dimension illustrating the probability of character x_t having the corresponding word boundary label. Afterwards, the model predicts the word boundary label $\widehat{y}_t^b$ of x_t with the highest probability, computes the negative log-likelihood loss by comparing $\widehat{y}_t^b$ with y_t^b,, and updates model parameters accordingly.

Through this process, the model learns the word boundary information from the antecedent clause (so as the subsequent clause when training)[6] and preserve it in an implicit manner for later processes.

3.2 Attentive Syntactic Information Encoding

Consider that the effectiveness of using syntactic information to improve neural models on many natural language understanding (NLU) tasks has been demonstrated by previous studies (Strubell et al., 2018; Zhang et al., 2019; Guo et al., 2019; Tian et al., 2020a; Chen et al., 2020; Wang et al., 2020), it is straightforward to consider its usefulness for couplet generation in modeling the syntactic correspondence between antecedent and subsequent clauses. Therefore in this work, we propose a special attention module to leverage it in doing so.

Specifically, in our approach, we focus on two types of syntactic information, namely, POS tags and word dependencies, which (silver standard annotations) can be obtained from off-the-shelf natural language processing (NLP) toolkits for the input antecedent clause $\mathcal{X}$ (where the preserved word boundaries are input to the toolkit in obtaining the syntactic information). For each character

[4]In practice, one can either use an existing Chinese word segmenter or use human annotations for this step. The advantage of using human annotation allows one to train a model learning high-quality word boundary information and thus could lead to better understanding of the couplets.

[5]For example, one can use the conventional *BIES* schema for the word boundary label, where the label of a character is "B", "I", "E" if the character is at the initial, inside, final position of a word, respectively, and the label "S" stands for the case if the character is a single-character word.

[6]In our practice, when use the model for inference, we do not predict the word boundary label in subsequent clauses.

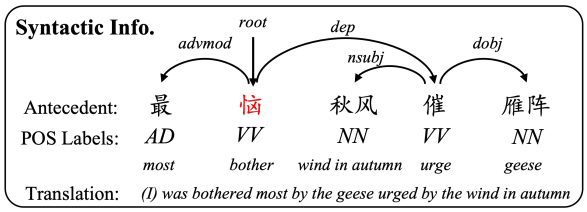

Figure 3: An example antecedent with its POS tags and word dependencies obtained from an off-the-shelf toolkit, where the POS and dependency instances associated with the second character "恼" (*bother*) are presented for better illustrating syntactic instance extraction. English translation is also provided for reference.

x_t, we firstly extracts a set of POS instances (tags) $POS_i = \{(c^{POS}_{t,i}, s^{POS}_{t,i})|1 \leq i \leq u_t\}$ and a set of dependency instances $Dep_i = \{(c^{dep}_{t,i}, s^{dep}_{t,i})|1 \leq i \leq v_t\}$ associated with x_t, where each POS and dependency instance is a pair of context feature $c^{type}_{t,i}$ and syntactic feature $s^{type}_{t,i}$ ($type \in \{POS, dep\}$).[7]

Specifically, for POS instances, we employ a five-character window[8] to extract the context characters (i.e., $x_{t-2} \cdots x_{t+2}$) and regard each of them as the context feature $c^{POS}_{t,i}$ in a POS instance. For the corresponding syntactic feature $s^{POS}_{t,i}$, we use the POS label associated with the word that contains $c^{POS}_{t,i}$. For example, in Figure 3, for x_2 = "恼" (*bother*), the associated POS instances are ("最", *AD*), ("恼", *VV*), ("秋", *NN*), ("风", *NN*). For dependency instances, the extraction process is elaborated as follows. First, we find the word (denote it as w) that contains x_t and extract all dependents (denote them as $w'_1 \cdots w'_j \cdots w'_l$) of w from the parsed dependency tree. Next, for each w'_j, we regard each character in w'_j as a context feature $c^{dep}_{t,i}$ and use the dependency connection type between w and w'_j as the corresponding syntactic feature $s^{dep}_{t,i}$ for each $c^{dep}_{t,i}$. As a special case, if w has no dependent (outbound connection to other words), we regard all characters in w as context features and use *null* as their corresponding syntactic features. For the example clause in Figure 3, x_2="恼" (*bother*) is associated with dependency instances ("最", *advmod*), ("催", *dep*), while x_1="最" (*most*) corresponds to the dependency instance ("最", *null*).

Once the POS and dependency instances are extracted, we use two separate attention parts following the same encoding procedure to model them, respectively, where the syntactic instances in each type (i.e., either POS or dependency) are dynamically weighed within its own type and distinguished based on their contribution to couplet generation, so as to address the noise in the syntactic instances. Using the encoding of POS instances as an example, for each instance $(c^{POS}_{t,i}, s^{POS}_{t,i})$, we firstly map the context feature $c^{POS}_{t,i}$ and the syntactic feature $s^{POS}_{t,i}$ to their embeddings, i.e., $\mathbf{c}^{POS}_{t,i}$ and $\mathbf{s}^{POS}_{t,i}$, respectively (we use boldface to represent the embedding of the features). Next, we add $\mathbf{c}^{POS}_{t,i}$ and $\mathbf{s}^{POS}_{t,i}$ to obtain the instance representation $\mathbf{e}^{POS}_{t,i} = \mathbf{c}^{POS}_{t,i} + \mathbf{s}^{POS}_{t,i}$. Then, we compute the attention $a^{POS}_{t,i}$ assigned to the instance by

$$a^{POS}_{t,i} = \frac{exp(\mathbf{h}_t \cdot \mathbf{e}^{POS}_{t,i})}{\sum_{i=1}^{u_t} exp(\mathbf{h}_t \cdot \mathbf{e}^{POS}_{t,i})} \quad (5)$$

Afterwards, we apply $a^{POS}_{t,i}$ to $\mathbf{e}^{POS}_{t,i}$ and computes the weighted sum of different POS instances, where the hidden vector $\mathbf{h}_t$ of x_t is further added to the resulting vector to obtain the encoded output vector $\mathbf{o}^{POS}_t$ with POS information through

$$\mathbf{o}^{POS}_t = \sum_{i=1}^{u_t} a^{POS}_{t,i} \cdot \mathbf{e}^{POS}_{t,i} + \mathbf{h}_t \quad (6)$$

Similarly, we obtain the output vector $\mathbf{o}^{dep}_t$ for dependency information following the same procedure. At last, we concatenate $\mathbf{o}^{POS}_t$, $\mathbf{o}^{dep}_t$, and $\mathbf{o}_t$ (obtained from the decoder in Eq. (3)) and feed the resulting vector $\mathbf{o}'_t = \mathbf{o}^{POS}_t \oplus \mathbf{o}^{dep}_t \oplus \mathbf{o}_t$ to the last fully connected layer for final prediction.

4 Experiments

4.1 Dataset

We follow Zhang et al. (2018) and use their experimental dataset[9] with 785K pairs of couplets crawled from Internet. Since there is no official train/dev/test split for this dataset, we make our own for each part and report the statistics of them in Table 1. In addition, we randomly sampled 1,000 couplets from the training set and manually annotated word segmentation for them,[10] where the re-

[7] Herein, u_t and v_t denote the numbers of POS and dependency instances associated with x_t, respectively.

[8] We use five as the window size because it is a hyper-parameter setting used in many previous studies to leverage POS tags and five achieves the optimal results in experiments.

[9] https://gitlab.com/feng-7/VV-couplet

[10] Annotating such a small amount of data does not require heavy manual work while later experiments confirm

Dataset	Couplet #	Char #
Train	755K	7,071K
Dev	10K	94K
Test	20K	186K

Table 1: The statistics of the Chinese couplet dataset with respect to the number of couplet pairs and characters in the training, development, and test sets.

Hyper-parameters	Values
Learning Rate	$\mathbf{1e^{-5}}, 5e^{-5}, 1e^{-4}, 1e^{-3}$
Dropout Rate	0.1, **0.2**, 0.3, 0.4
Batch Size	8, 16, **32**

Table 2: The hyper-parameters tested in tuning our models. Bold values illustrate the best hyper-parameter configuration that is applied in our experiments.

sulted dataset is used for our alternative training process with labeled word boundary information.

4.2 Implementation Details

Since our approach requires syntactic information as extra input features, for each input antecedent, we obtain the POS tags via TwASP[11] (Tian et al., 2020b) and the word dependencies via DMPar[12] (Tian et al., 2022).[13] For the encoder, considering that pre-trained language models have demonstrated their effectiveness in obtaining high-quality text representations for many NLP tasks (Yang et al., 2019; Diao et al., 2020; Raffel et al., 2019; Sun et al., 2020; Song et al., 2021), in the experiments, we use the pre-trained Chinese BERT-base (Devlin et al., 2019) following the default settings (i.e., 12 layers of self-attention with 768 dimensional hidden vectors). For the decoder, we use the standard Transformer setting with 6 layers of self-attention and 768 dimensional hidden vectors.

In training, we design an alternatively training strategy, where we firstly train the model with both losses on word boundary information and couplet generation on the 1,000 human annotated instances, and then train the model on the rest entire training set without considering word boundary loss since word segmentation for such data is provided by automatic tools. So that in each iteration the model can be enhanced by gold-standard word boundary information from few training instances. For other hyper-parameters, we try the combinations illustrated in Table 2 and use the ones (which are highlighted by boldface) that achieve the best performance on the development set. Following previous studies, we use BLEU (Papineni et al., 2002) and format accuracy for model evaluation).

4.3 Overall Performance

Table 3 reports the experimental results from several BERT-based baselines and our proposed model enhanced by word boundaries and syntactic information (i.e., POS tags and word dependencies), respectively. Specifically, "BERT" denotes the vanilla BERT baseline model, "+WB", "+POS", and "+Dep" refer to the enhancement of the baseline with word boundaries, POS information, and dependency information, respectively, with "+Full" denoting the full model combines all the aforementioned information. There are several observations that are explained in the following paragraphs.

First, overall, although the baselines with different settings have already achieved outstanding performance, it is promising to observe that our model with all enhancements (i.e., "+Full") is able to outperform them with respect to all evaluation metrics, which confirms the usefulness of leveraging different information for couplet generation.

Second, comparing "+WB" and "BERT", we observe that "+WB" presents higher performance than "BERT" on all evaluation metrics (especially on BLEU-3 and BLEU-4 scores), which demonstrates the effectiveness of our word boundary learning process to leverage word boundary information in improving couplet generation. The interpretation is as follows. Although word boundary information is leveraged in an implicit way, it is encoded and the loss back-propagated to the encoder helps the model understand the semantic units in different granularities in the antecedent clause, then the generation process is enhanced accordingly with such information.

Third, models with a single type of syntactic information (i.e., "+POS" and "+Dep") outperform the baseline model with respect to all evaluation metrics, showing their effectiveness in guiding the model to generate better couplets. Furthermore, it is observed that the model with word dependencies ("+Dep") outperforms the the one with POS tags ("+POS") on BLEU-2, BLEU-3, and BLEU-4. This observation indicates that word de-

it is enough to help our model in better generating couplets.

[11] https://github.com/SVAIGBA/TwASP

[12] https://github.com/synlp/DMPar

[13] TwASP is a joint model for Chinese word segmentation and POS tagging. So we use it to obtain the word segmentation result for each couplet and feed it to dependency parsing.

Model	BLEU-1	BLEU-2	BLEU-3	BLEU-4	FA	Para. #	Speed
BERT	20.53	13.09	9.88	7.62	98.64	156M	49.6
+ WB	20.99	13.21	10.01	7.87	98.70	156M	48.7
+ POS	21.34	13.63	10.23	8.50	98.98	189M	45.9
+ Dep	21.39	13.82	10.36	8.77	99.01	189M	46.4
+ Full	**21.62**	**13.95**	**10.68**	**8.94**	**99.10**	221M	42.1

Table 3: The BLEU scores and format accuracy (FA) of different models on the test set. "BERT" denotes the baseline model with BERT encoder; "+WB", "+POS", and "+Dep" refer to the baseline model with enhancement of word boundaries, POS tags, and dependency information, respectively; "+Full" is our proposed model with enhancement of all aforementioned information. "Para. #" presents the number of parameters in different models; "Speed" refers to the number of generated couplets per second in inference.

Model	BL-1	BL-2	BL-3	BL-4	FA
BERT	20.53	13.09	9.88	7.62	98.64
+WB	**20.99**	**13.21**	**10.01**	**7.87**	**98.70**
+Word	17.98	9.63	6.05	4.21	90.01

Table 4: Comparison between two different BERT-based approach to leverage word boundary information without using syntactic information. "+WB" denotes the approach with the proposed word boundary learning process; "+Word" refer to the approach that encodes and decodes the couplets on word level. The results of the BERT baseline is also reported for reference. "BL" is the abbreviation of "BLEU".

Model	Syntax Intgrt.	BL-1	BL-2	BL-3	BL-4	FA
+POS	Ours	**21.34**	**13.63**	**10.23**	**8.50**	**98.98**
	Alter.	21.26	13.01	9.89	8.29	98.60
+Dep	Ours	**21.39**	**13.82**	**10.36**	**8.77**	**99.01**
	Alter.	21.24	13.57	10.14	8.01	98.91
+Full	Ours	**21.62**	**13.95**	**10.68**	**8.94**	**99.10**
	Alter.	21.42	13.67	10.20	8.15	98.82

Table 5: Comparison of model performance between different methods to integrate syntactic information in different settings. "Ours" denotes our approach to integrate POS or dependency information, where the hidden vector $\mathbf{h}_t$ is added to the weighted sum of syntactic instances (see Eq. (6)), whereas "Alter." refers to the method where $\mathbf{h}_t$ is not used in the integration.

pendencies provide stronger enhancement to couplet generation than that of POS tags, which could be explained by that the dependency information contains long distance constraints over the entire couplet clause while a POS tag only presents the function of the corresponding local words.

5 Analysis

5.1 The Effect of Word Boundaries

To analyze the effect of our model design by leveraging word boundary information in an implicit way, we run experiments with another straightforward approach (which is denoted as "+Word"), where the couplets are segmented into words and the couplet generation process is performed on the word-level prediction. Table 4 reports the performance of "+Word" model, as well as our "+WB" model and the BERT baseline. It is observed that "+Word" shows inferior results compared with our "+WB" model and the BERT baseline, which confirms the superiority of the designed learning process.[14] This observation can be explained by that the "+Word" approach generates word-by-word subsequent clauses, the vocabulary size for it is relatively larger than that of "+WB", which performs the generation in a character-by-character manner. The generation process of "+Word" is thus inevitably bothered by the out-of-vocabulary problem where at many decoding steps the model cannot find appropriate word candidates.

5.2 The Effect of Syntactic Information

Similar to the analysis for word boundaries, there are also alternatives to integrate syntactic heuristics into our model, where our approach also use an implicit way by adding the hidden vector $\mathbf{h}_t$ obtained from the encoder to the weighted sum of syntactic information (see Eq. (6)). In this investigation, we run another model without using $\mathbf{h}_t$, which means only the output of the attention module is applied for syntactic information integration. This comparison illustrates how syntactic information affects model performance, where the results are presented in Table 5. Clearly, the alternative method shows inferior results compared to our approach with respect to all metrics. The reason for this observation complies with our hypothesis that although directly

[14] We still use character-based BLEU in this experiment to evaluate the generated clause for all models.

Model	Syn.	Sem.	All
BERT	0.784	0.636	1.420
+WB	0.805	0.659	1.464
+POS	0.818	0.679	1.497
+Dep	0.809	0.665	1.474
+Full	0.830	0.728	1.558

Table 6: Human evaluation for different models. "Syn." and "Sem." are syntactic and semantic scores, respectively, and "All" refers to the sum of them.

using the syntactic information shows its advantage over the BERT baseline (see Table 3), the decoding process still relies mainly on the contextual information from the encoder. Therefore, only using the syntactic information from the attention module in a standalone way may not be enough to drive a better couplet generation process.

5.3 Human Evaluation

Following the convention in previous studies (Jiang and Zhou, 2008; He et al., 2012; Zhang et al., 2018; Fan et al., 2019; Gao et al., 2021; Wang et al., 2021), we perform human evaluation for different models in addition to BLEU scores and format accuracy. Particularly, we use "syntactic" and "semantic" satisfaction to assess how antecedent and subsequent clauses are matched in terms of their patterns[15] and meanings[16], respectively. In doing so, human annotators are given an antecedent clause and a subsequent clause generated by a model and they are asked to assign 0-1 score to the generated subsequent for both criteria ("0'' for *not satisfying*, "1" for *satisfying*).

We randomly select 50 couplets from the test set for human evaluation and employ five human experts who are familiar with Chinese couplets to conduct human evaluation with a blind review manner. In detail, they are provided with the subsequent clauses generated by different models (i.e., all baselines and the "+Full" model) and they do not know the model that generates the given subsequent clause. The results from five experts on the aforementioned two aspects (i.e., syntactic satisfaction and semantic satisfaction) are averaged and presented in Table 6. Similar to the observations from Table 3, word boundaries and syntactic information show their advantages in helping couplet

[15]There is a special assessment about the ending tone of the two clauses, where the tones of the last character in antecedent and subsequent clauses should be opposed to each other.

[16]Ideally, the meaning of the subsequent clause should be relevant to the antecedent clause but not the same.

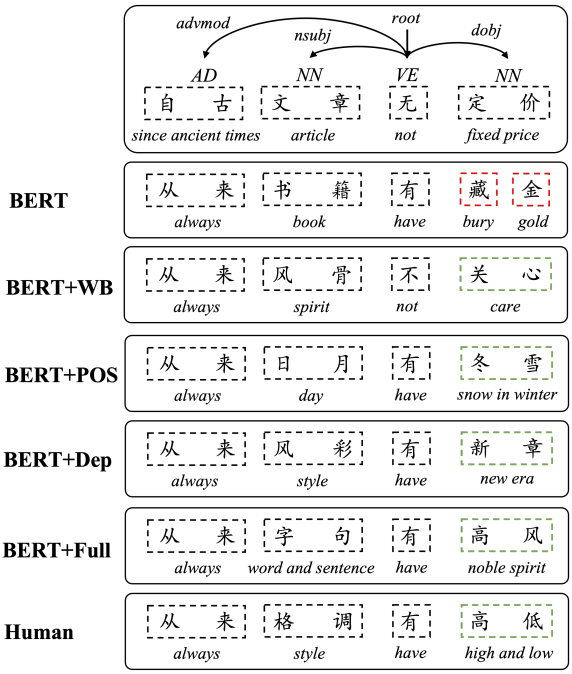

Figure 4: Comparison of subsequent clause generated from different models for an example input antecedent clause with its POS tags and word dependencies makred (presented at the top). Original subsequent clause (Human) written by human is also presented at the bottom. Translations for every clause in the verbatim manner are provided for reference. Characters belonging to the same word are bounded by dash boxes, where mismatched patterns is marked in red boxes.

generation with more satisfying results from humans. Particularly, compared with automatic evaluations (i.e., BLEU scores and format accuracy), human judgements for different models shows a more significant trend that the proposed components are useful especially when they are integrated together in our proposed approach.

5.4 Case Study

To further investigate the effect of our approach, we conduct a case study to qualitatively illustrate the performance of different models with their generated subsequent clauses for a given input. Figure 4 illustrates the results. For the given antecedent, there are five generated clauses from all baselines and our proposed model (i.e., "BERT+Full"), as well as the original subsequent clause composed by human for reference, where all clauses are segmented (in dashed boxes) in order to explicitly show word boundaries. It is clearly presented that, the BERT baseline (i.e., "BERT") fails to produce a subsequent clause with correct word mappings to the antecedent one, e.g., the word boundary of "藏/金" (*bury the gold*) does not match that

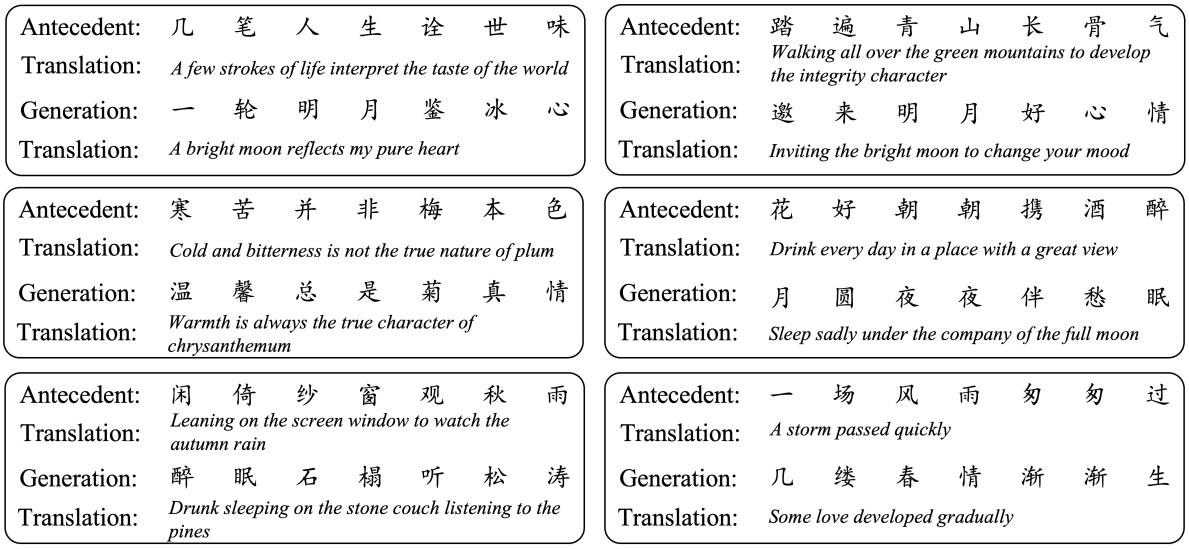

Figure 5: Six example pairs of Chinese couplets with their subsequent clauses generated by our proposed model (i.e., "BERT+Full") corresponding to the input antecedent clauses. The English translations are also provided for each clause for all couplets based on the semantics of each entire sentence instead of the verbatim manner.

of "定价" (*a fixed price*). All rest clauses generated from the models enhanced by word boundaries or syntactic information do not suffer from such mis-matching problem, where the one from "BERT+Full" is indisputably better than the others and the reason can be explained in three aspects.

First, the pattern in the subsequent clause is identical to that of the antecedent clause, including that the tones of the last character in both clauses satisfy the rule mentioned in the previous section (see footnote 15), such as "风" (flat tone) v.s. "价" (oblique tone). Second, each unit in the subsequent clause shows a direct correspondence with the unit at the same position in the antecedent clause, e.g., "字句" (*word and sentence*) v.s. "文章" (*article*), "无" (*not*) v.s. "有" (*has*), "高风" (*noble spirit*) v.s. "定价" (*fixed price*), etc. Third, the meaning of the entire subsequent clause makes a counterpoint to the antecedent one with semantic relatedness, i.e., the subsequent clause refers to that *the words and sentences always have noble spirit* while the antecedent clause states that *the articles are priceless since ancient times*. As comparisons, although other generated clauses have their advantages in delivering either more interesting meaning or better artistic conceptions, they are not well performed on pattern matching or semantic relatedness.

To have more intuitive understanding about the performance of our proposed model, we also show in Figure 5 six randomly selected pairs of couplets with generated subsequent clauses. It is clearly to observe from all couplet pairs that the generated clauses match the antecedent inputs well on their patterns (on all aspects) and also keep the balance of expressing good artistic conception while being strictly corresponded to the meaning of the antecedent clauses, such as "温馨" (*warmth*) v.s. "寒苦" (*cold and bitterness*), "明月" (*the bright moon*) v.s. "青山" (*the green mountains*), which are all neat correspondences and promote the overall effect (including both meaning and artistic conception) of the entire couplet pairs to a higher level.

6 Related Work

Chinese couplet generation is an intriguing natural language generation task, which is relevant to poem generation (Greene et al., 2010; Zhang and Lapata, 2014; Wang et al., 2016; Yang et al., 2018b; Zhang et al., 2017; Yi et al., 2017; Ghazvininejad et al., 2017; Yi et al., 2018; Yang et al., 2018a; Li et al., 2018; Liu et al., 2018, 2019; Liao et al., 2019; Bena and Kalita, 2020). Yet, couplet generation differs from that for poem in the way that it requires the generation process sticking to more strict patterns and semantic requirements, while they are similar that they all conventionally performed as a sequence-to-sequence text generation task, including statistic-based approaches (Jiang and Zhou, 2008; Zhang and Sun, 2009; He et al., 2012) and neural approaches following the encoding-decoding paradigm (Zhang et al., 2018; Fan et al., 2019; Gao et al., 2021). To improve model performance, previous neural approaches try to model the character-character correspondence between

antecedent and subsequent clauses, where some advanced approaches such as attention mechanism (Zhang et al., 2018) and character embedding pre-training (Gao et al., 2021) are applied. Compared with existing studies, our approach offers an alternative and explicit way to model the correspondence between couplet clauses through syntactic information, which provides useful knowledge to control patternized generation and is integrated into our approach via a carefully designed module.

7 Conclusion

In this paper, we proposed a neural approach following the encoder-decoder paradigm for Chinese couplet generation enhanced by of syntactic information (i.e., POS tags and word dependencies). Specifically, our approach models word boundaries to facilitate the learning of syntactic information, where POS tags and word dependencies are leveraged to provide pattern guidance for couplet generation through a special attention module. Experimental results on a prevously used dataset for Chinese couplet generation illustrate the effectiveness of our approach, which outperforms strong baselines on both automatic and human evaluations. Further analyses also confirm the ability of our approach to generating high quality couplets.

Acknowledgements

We thank Mr. Dongming Li for his early preliminary help to this study.

References

Brendan Bena and Jugal Kalita. 2020. Introducing Aspects of Creativity in Automatic Poetry Generation. *arXiv preprint arXiv:2002.02511*.

Guimin Chen, Yuanhe Tian, and Yan Song. 2020. Joint Aspect Extraction and Sentiment Analysis with Directional Graph Convolutional Networks. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 272–279.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186.

Shizhe Diao, Jiaxin Bai, Yan Song, Tong Zhang, and Yonggang Wang. 2020. ZEN: Pre-training Chinese Text Encoder Enhanced by N-gram Representations. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 4729–4740.

Haoshen Fan, Jie Wang, Bojin Zhuang, Shaojun Wang, and Jing Xiao. 2019. Automatic Acrostic Couplet Generation with Three-Stage Neural Network Pipelines. In *Pacific Rim International Conference on Artificial Intelligence*, pages 314–324. Springer.

Rui Gao, Yuanyuan Zhu, Mingye Li, Shoufeng Li, and Xiaohu Shi. 2021. Encoder–Decoder Couplet Generation Model Based on 'Trapezoidal Context'Character Vector. *The Computer Journal*, 64(3):286–295.

Marjan Ghazvininejad, Xing Shi, Jay Priyadarshi, and Kevin Knight. 2017. Hafez: an Interactive Poetry Generation System. In *Proceedings of ACL 2017, System Demonstrations*, pages 43–48.

Erica Greene, Tugba Bodrumlu, and Kevin Knight. 2010. Automatic Analysis of Rhythmic Poetry with Applications to Generation and Translation. In *Proceedings of the 2010 conference on empirical methods in natural language processing*, pages 524–533.

Zhijiang Guo, Yan Zhang, and Wei Lu. 2019. Attention Guided Graph Convolutional Networks for Relation Extraction. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 241–251.

Jing He, Ming Zhou, and Long Jiang. 2012. Generating Chinese Classical Poems with Statistical Machine Translation Models. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 26.

Long Jiang and Ming Zhou. 2008. Generating Chinese Couplets Using a Statistical MT Approach. In *Proceedings of the 22nd International Conference on Computational Linguistics (Coling 2008)*, pages 377–384.

Juntao Li, Yan Song, Haisong Zhang, Dongmin Chen, Shuming Shi, Dongyan Zhao, and Rui Yan. 2018. Generating Classical Chinese Poems via Conditional Variational Autoencoder and Adversarial Training. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 3890–3900, Brussels, Belgium.

Yi Liao, Yasheng Wang, Qun Liu, and Xin Jiang. 2019. GPT-based Generation for Classical Chinese Poetry. *arXiv preprint arXiv:1907.00151*.

Dayiheng Liu, Quan Guo, Wubo Li, and Jiancheng Lv. 2018. A Multi-modal Chinese Poetry Generation Model. In *2018 International Joint Conference on Neural Networks (IJCNN)*, pages 1–8.

Zhiqiang Liu, Zuohui Fu, Jie Cao, Gerard de Melo, Yik-Cheung Tam, Cheng Niu, and Jie Zhou. 2019. Rhetorically Controlled Encoder-decoder for Modern Chinese Poetry Generation. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 1992–2001.

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. BLEU: A Method for Automatic Evaluation of Machine Translation. In *Proceedings of the 40th annual meeting of the Association for Computational Linguistics*, pages 311–318.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J Liu. 2019. Exploring the Limits of Transfer Learning with a Unified Text-to-text Transformer. *arXiv preprint arXiv:1910.10683*.

Yan Song, Tong Zhang, Yonggang Wang, and Kai-Fu Lee. 2021. ZEN 2.0: Continue Training and Adaption for N-gram Enhanced Text Encoders. *arXiv preprint arXiv:2105.01279*.

Emma Strubell, Patrick Verga, Daniel Andor, David Weiss, and Andrew McCallum. 2018. Linguistically-Informed Self-Attention for Semantic Role Labeling. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 5027–5038, Brussels, Belgium.

Yu Sun, Shuohuan Wang, Yukun Li, Shikun Feng, Hao Tian, Hua Wu, and Haifeng Wang. 2020. ERNIE 2.0: A Continual Pre-Training Framework for Language Understanding. *Proceedings of the AAAI Conference on Artificial Intelligence*, 34(05):8968–8975.

Ilya Sutskever, Oriol Vinyals, and Quoc V Le. 2014. Sequence to Sequence Learning with Neural Networks. *Advances in Neural Information Processing Systems*, 27:3104–3112.

Yuanhe Tian, Wang Shen, Yan Song, Fei Xia, Min He, and Kenli Li. 2020a. Improving Biomedical Named Entity Recognition with Syntactic Information. *BMC Bioinformatics*, 21:1471–2105.

Yuanhe Tian, Yan Song, Xiang Ao, Fei Xia, Xiaojun Quan, Tong Zhang, and Yonggang Wang. 2020b. Joint Chinese Word Segmentation and Part-of-speech Tagging via Two-way Attentions of Auto-analyzed Knowledge. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 8286–8296, Online.

Yuanhe Tian, Yan Song, and Fei Xia. 2022. Enhancing Structure-aware Encoder with Extremely Limited Data for Graph-based Dependency Parsing. In *Proceedings of the 29th International Conference on Computational Linguistics*.

Daisy Zhe Wang, Wei He, Hua Wu, Haiyang Wu, Wei Li, Haifeng Wang, and Enhong Chen. 2016. Chinese Poetry Generation with Planning Based Neural Network. In *Proceedings of COLING 2016, the 26th International Conference on Computational Linguistics: Technical Papers*, pages 1051–1060.

Kai Wang, Weizhou Shen, Yunyi Yang, Xiaojun Quan, and Rui Wang. 2020. Relational Graph Attention Network for Aspect-based Sentiment Analysis. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 3229–3238, Online.

Yufeng Wang, Jiang Zhang, Bo Zhang, and Qun Jin. 2021. Research and Implementation of Chinese Couplet Generation System with Attention Based Transformer Mechanism. *IEEE Transactions on Computational Social Systems*.

Cheng Yang, Maosong Sun, Xiaoyuan Yi, and Wenhao Li. 2018a. Stylistic Chinese Poetry Generation via Unsupervised Style Disentanglement. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 3960–3969.

Xiaopeng Yang, Xiaowen Lin, Shunda Suo, and Ming Li. 2018b. Generating Thematic Chinese Poetry Using Conditional Yariational Autoencoders with Hybrid Decoders. In *Proceedings of the 27th International Joint Conference on Artificial Intelligence*, pages 4539–4545.

Zhilin Yang, Zihang Dai, Yiming Yang, Jaime Carbonell, Russ R Salakhutdinov, and Quoc V Le. 2019. XLNet: Generalized Autoregressive Pretraining for Language Understanding. In *Advances in Neural Information Processing Systems 32*, pages 5753–5763.

Xiaoyuan Yi, Ruoyu Li, and Maosong Sun. 2017. Generating Chinese Classical Poems with RNN Encoder-decoder. In *Chinese Computational Linguistics and Natural Language Processing Based on Naturally Annotated Big Data*, pages 211–223.

Xiaoyuan Yi, Maosong Sun, Ruoyu Li, and Wenhao Li. 2018. Automatic Poetry Generation with Mutual Reinforcement Learning. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 3143–3153.

Hongming Zhang, Yan Song, Yangqiu Song, and Dong Yu. 2019. Knowledge-aware Pronoun Coreference Resolution. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 867–876, Florence, Italy.

Jiyuan Zhang, Yang Feng, Dong Wang, Yang Wang, Andrew Abel, Shiyue Zhang, and Andi Zhang. 2017. Flexible and Creative Chinese Poetry Generation Using Neural Memory. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1364–1373.

Jiyuan Zhang, Zheling Zhang, Shiyue Zhang, and Dong Wang. 2018. VV-Couplet: An Open Source Chinese Couplet Generation System. In *2018 Asia-Pacific Signal and Information Processing Association Annual Summit and Conference (APSIPA ASC)*, pages 1756–1760. IEEE.

Kai-Xu Zhang and Mao-Song Sun. 2009. An Chinese Couplet Generation Model Based on Statistics and Rules. *Journal of Chinese Information Processing*, 1:017.

Xingxing Zhang and Mirella Lapata. 2014. Chinese Poetry Generation with Recurrent Neural Networks. In *Proceedings of the 2014 Conference on Empirical*

Methods in Natural Language Processing (EMNLP),
pages 670–680.

Noise-injected Consistency Training and Entropy-constrained Pseudo Labeling for Semi-supervised Extractive Summarization

Yiming Wang[1,2,3,†], **Qianren Mao**[1,2,†], **Junnan Liu**[1,2], **Weifeng Jiang**[1,2], **Hongdong Zhu**[1,2], **Jianxin Li**[1,2,*]

[1] Beijing Advanced Innovation Center for Big Data and Brain Computing, Beihang University, China.
[2] The State Key Laboratory of Software Development Environment, Beihang University, China.
[3] Institute of Artificial Intelligence, Beihang University, China.
{wangym,maoqr,liujn,jiangwf,zhuhd,lijx}@act.buaa.edu.cn

Abstract

Labeling large amounts of extractive summarization data is often prohibitive expensive due to time, financial, and expertise constraints, which poses great challenges to incorporating summarization system in practical applications. This limitation can be overcome by semi-supervised approaches: *consistency-training* and *pseudo-labeling* to make full use of unlabeled data. Researches on the two, however, are conducted independently, and very few works try to connect them. In this paper, we first use the noise-injected consistency training paradigm to regularize model predictions. Subsequently, we propose a novel entropy-constrained pseudo labeling strategy to obtain high-confidence labels from unlabeled predictions, which can obtain high-confidence labels from unlabeled predictions by comparing the entropy of supervised and unsupervised predictions. By combining consistency training and pseudo-labeling, this framework enforce a low-density separation between classes, which decently improves the performance of supervised learning over an insufficient labeled extractive summarization dataset.

1 Introduction

Text summarization is a challenging task that generates a condensed version of an input text that captures the original's core meaning. In this paper, we focus on extractive summarization since it usually generates semantically and grammatically correct sentences (Liu and Lapata, 2019; Zhong et al., 2019a; Zhou et al., 2020; Zhong et al., 2020). The extractive summarization typically requires to label each sentence in massive documents for model training. However, acquiring well-annotated labels is a costly process, and labeling every sentence would be labor-intensive and error-prone due to subjective judgments of human. This motivates research on Semi-Supervised Learning (SSL) methods which focus on how to effectively utilizes abundant unlabeled data, to further improve extractive summarization performances.

Towards this goal, we first revisit an effective semi-supervised method, consistency training (Xie et al., 2020a). The consistency training leverages voluminous unlabeled data and employs advanced data augmentation methods to generate diverse and realistic noisy source text, forcing the model to be consistent with these noises. The consistency training has been extensively applied on the classification problems, such as Text Classification (Xie et al., 2020a; Liu et al., 2021a), Image Recognition (Laine and Aila, 2017; Tarvainen and Valpola, 2017; Miyato et al., 2019; Verma et al., 2019; Xie et al., 2020b). However, how does the consistency training work on semi-supervised extractive summarization tasks is still unclear.

We investigate the noise-injected consistency training for semi-supervised extractive summarization to encourage a consistent reason of model decision (summary and non-summary) under data perturbation. This framework makes sense intuitively because a good supervised model should be robust to any slight change in an input example. Namely, encouraging local change by injecting a slight noise in a diverse perturbation manner can improve the summarization effectiveness.

Nevertheless, the consistency regularized semi-supervised framework usually suffers from insufficient supervision. When labeled data is limited, the model is easy to over-fitting. Extensive unlabeled data will then make the model suffer from the gradual drift problem and impede further improvements of the model. To address this problem, we develop new methods of selection and exploitation for pseudo labels to explore all unlabeled samples for the semi-supervised summarization cycle.

Prior pseudo labeling work (Lee et al., 2013; Sohn et al., 2020; Rizve et al., 2021) mainly fo-

[†] These two authors contributed equally.
[*] Jianxin Li is the corresponding author.

Proceedings of the 29th International Conference on Computational Linguistics, pages 6447–6456
October 12–17, 2022.

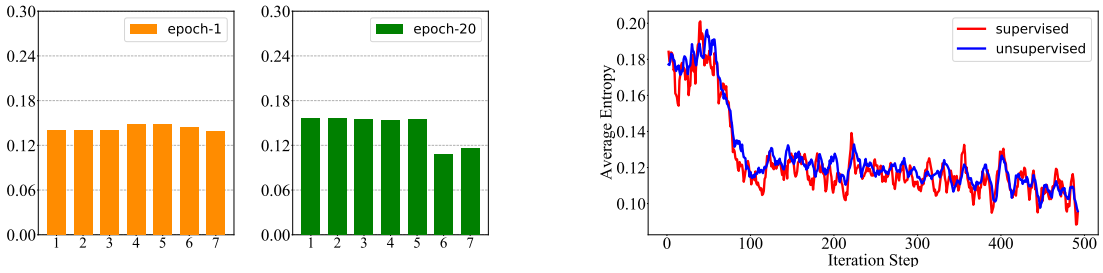

(a) Prediction probability distributions

(b) Average entropy changing curve

Figure 1: (a) Prediction probability distributions of the same sample in different epochs before and after convergence. (b)The average entropy changing curve of labeled and unlabeled samples before convergence during training.

cuses on the confidence of the predictions to pick up the class used as if they are accurate labels. However, the summarization task exposes potential problems with this approach. The confidence of the prediction results is hard to change significantly along with model convergence (e.g., using the BERT-based model). As shown in the second sub-figure in Fig. 1(a), almost all prediction probability of sentences has achieved an undifferentiated score between 0.12 to 0.16 in the convergence epoch, which means almost all sentences are mapped into a small area for classification. Therefore, it is challenging to set a fixed threshold to determine a proper confidence score, and it is easy to produce low-quality pseudo labels. We divert attention to entropy − a metric for measuring uncertainty, to address this problem. As is shown in Fig 1(b), the prediction entropy of supervised and unsupervised data are comparable, which enables us to constrain the unsupervised entropy with supervised entropy without any external threshold value. We introduce entropy-constrained pseudo labeling to avoid fixed threshold adoption, which ensures the entropy of prediction results of unlabeled data adjust that of labeled samples adaptively.

The final framework is **CPSUM**[1], which combines the Noise-injected **C**onsistency training with the Entropy-constrained **P**seudo labeling for Semi-supervised Extractive **SUM**marization. Experimental results demonstrate that our approach achieves the state-of-the-art in low-resource scenarios with 10, 100, and 1000 labeled examples on the target corpus. The main contributions of our method are:

- To the best of our knowledge, this is the first work to explore the feasibility of consistency-training and pseudo-labeling for semi-supervised extractive summarization tasks.
- Proposing a novel threshold-free approach selecting reliable pseudo-labels with the average entropy comparison, which is well-adapted to extractive summarization tasks.
- Extensive evaluations demonstrate that consistency training and pseudo-labeling with unsupervised corpus could greatly improve the performance of the text summarization model on a limited dataset.

2 Related Work

2.1 Extractive Summarization

Extractive summarization selects the most representative sentences within a document and subsequently splices them into the final summary. Approaches for it are constantly updated. With classical networks, RNN-based (Nallapati et al., 2017; Zhou et al., 2018), Transformer-based (Zhong et al., 2019b; Liu et al., 2021b) are adopted. Pre-trained summarization models have achieved great success, such as the notable BERTSUMEXT (Liu and Lapata, 2019; Liu, 2019) which is the first work to use the BERT (Devlin et al., 2019) for extractive summarization. However, current extractive summarization models still heavily rely on many parallel data to achieve salient performance. Little work has focused on low-resourced settings where handcrafted labels for sentences are limited or even unavailable. To fill this gap, in this work, we introduce a novel semi-supervised framework to alleviate the dependence on labeled summaries.

2.2 Consistency Regularization

In recent work, consistency regularization methods for semi-supervised learning (Bachman et al., 2014) have been shown to work well on many classification tasks (Xie et al., 2020a; Liu et al., 2021a).

[1]Code and data available at: https://github.com/OpenSUM/CPSUM.

Their work can match and even outperform purely supervised learning that uses affluent labeled data.

The consistency training methods regularize model predictions invariant to noise applied to unlabeled examples. Tarvainen and Valpola (2017) prove that a model trained with noisy labeled data learns to give consistent predictions around labeled data points. Additionally, advanced data augmentation methods (Xie et al., 2020a) can improve consistency training performance effectively.

2.3 Pseudo-labeling

Pseudo-labeling (Lee et al., 2013) is an efficient semi-supervised learning method by generating pseudo-labels to expand labeled data. For selecting reliable pseudo-labels, FixMatch (Sohn et al., 2020) creates a selection criterion based on the confidence threshold. After that, considering poor network calibration, UPS (Rizve et al., 2021) introduces model uncertainty criterion based on prediction result confidence. U^{2}PL (Wang et al., 2022) does the opposite and fully considers the value of some unreliable pseudo-labels. However, the performance of pseudo-labels on the extractive summarization task remains to be evaluated, and the methods above are not suitable for this task directly.

3 Proposed Framework

The overview illustration of our framework is shown in Fig. 2. It is composed of two components: noise-injected consistency training and entropy-constrained pseudo labeling. Specific to extractive summarization, we use the base version of BERT (Devlin et al., 2019) to implement our models in all experiments. We give detail description of two components in Sec.3.1 and Sec.3.2.

3.1 Noise-injected Consistency Training

The consistent regularization of semi-supervised learning leverages unlabeled data, employs data augmentation methods to inject noisy data, and enforces the summarization model by encouraging consistent predictions.

Data Augmentation. The unlabeled noise examples, specifically those produced by advanced data augmentation methods, have been proved to be crucial for consistency training (Xie et al., 2020a). Our augmentation method refers to Jiao et al. (2020). We replace single-piece words (Wu et al., 2019) by predictions of the BERT masked language model and retrieve the most similar words as word replacements for multiple-pieces words by using the word embedding in GLOVE (Pennington et al., 2014).

Consistency Training. The robust summarization model should also be invariant for documents with similar content. Hence, we leverage consistency learning to regularize model predictions to be invariant to slight noise applied to input examples.

The inputs of the framework are labeled texts x, unlabeled texts $x^{'}$, and noise injected unlabeled texts $x^{''}$. We use y^* to denote the gold summaries of labeled texts. Then we use f_θ to represent the distributions of model predictions, where θ refers to the model's parameters. Firstly, we feed the labeled text x into the model to get the predictions and calculate the supervised loss:

$$\mathcal{L}_l = \frac{1}{|X_l|} \sum_{x \in X_l} l(y^*, f_\theta(y|x)), \quad (1)$$

where X_l is a set containing $|X_l|$ labeled data x. We then generate a noised version $x^{''}$ of the unlabeled text $x^{'}$ using the aforementioned data augmentation method. Both unlabeled texts and noised unlabeled texts are fed to the summarization model, and then we get the output distribution of original unlabeled data $f_{\tilde{\theta}}(y^{'}|x^{'})$ and the additional noised version of augmented unlabeled data $f_\theta(y^{''}|x^{''})$. We then calculate the unsupervised loss between unlabeled texts and augmented unlabeled texts:

$$\mathcal{L}_u = \frac{1}{|X_u|} \sum_{(x',x'') \in X_u} l(f_{\tilde{\theta}}(y^{'}|x^{'}), f_\theta(y^{''}|x^{''})), \quad (2)$$

where X_u is a set of pairs containing $|X_u|$ unlabeled data x' and the corresponding augmented data x'', and $\tilde{\theta}$ is just a copy of the current parameters θ indicating that the back-propagation of the gradient is truncated. We use KL divergence loss to perform consistency training.

Finally, we combine supervised cross-entropy loss and supervised consistency loss, and train the model by minimizing the combined loss: $\mathcal{L}_f = \mathcal{L}_l + w(t)\mathcal{L}_u$, where $w(t)$ is the ramp-up weight balancing supervised and unsupervised learning.

3.2 Entropy-constrained Pseudo Labeling

Generally, the prediction results with the highest predicted probability of unlabeled data could be adopted as pseudo labels. However, low-quality pseudo labels may harm model training.

To overcome this problem, we introduce a method named entropy-constrained pseudo labeling to select reliable pseudo labels. We argue that if

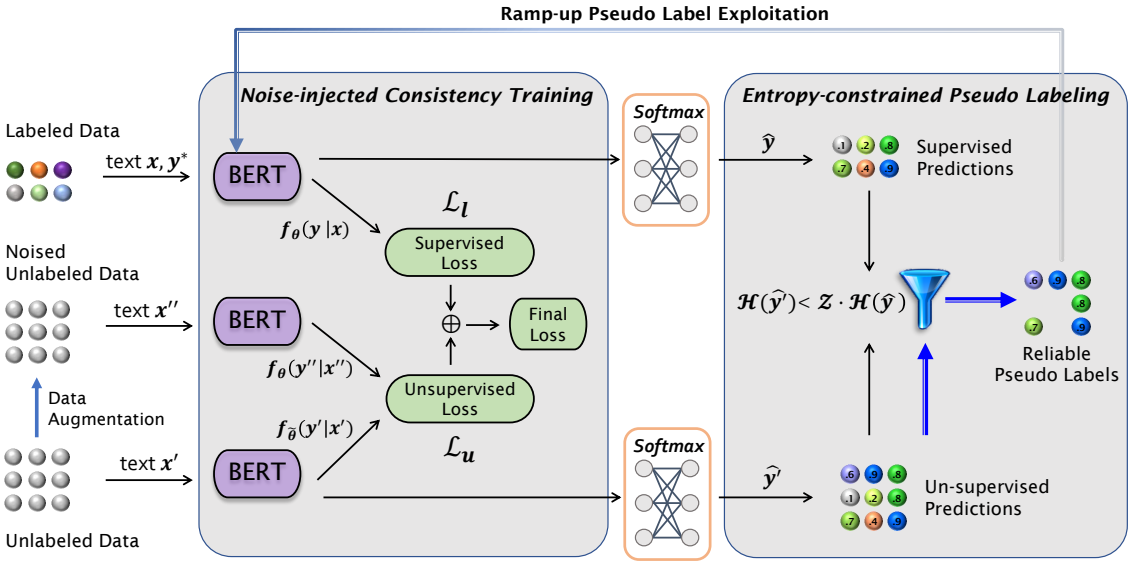

Figure 2: Illustration of **CPSUM** with Noise-injected Consistency Training and Entropy-constrained Pseudo Labeling for Exactive Summarization. In the figure, θ refers to the parameters of the model, and $\tilde{\theta}$ means it is just a copy of θ and the gradient will not propagate through it. $\hat{y}$ and $\hat{y'}$ refer to the logits by the softmax of labeled and unlabeled samples, respectively, $\mathcal{H}$ is the symbol for entropy, and $\mathcal{Z}$ is the normalization factor.

the entropy of the predicted result of unlabeled data is smaller than that of labeled data used at the current training step, then the low noisy pseudo labels generated by unlabeled data should be reserved.

Pseudo-label Selection. Either soft pseudo-labels selection (**Soft PlS**) or hard pseudo-labels selection (**Hard PlS**) can be adopted. We denote $logit$ and $logit'$ as the outputs of labeled and unlabeled data after the sigmoid operation by the model.

For soft pseudo labels, they are equivalent to the logit of unlabeled data, namely $y_{soft} = logit'$. Hard pseudo labels are essentially the binary vectors mapping soft pseudo labels to '0-1' space. Suppose that a document has K sentences which will be tagged with K labels. C represents the number of summary sentences. Then, hard pseudo labels can be defined as: $y_{hard} = f_{ope}(y_{soft}) \subseteq \{0,1\}^K$, where f_{ope} is the mapping operation converting the top-C probability of y_{soft} to '1' and the remaining to '0'.

Adaptive Entropy Constraint. Unlike text classification problems focusing on the highest probability (Kim, 2014), extractive summarizations must consider the multiple highest probabilities. This situation makes the selection criteria based on the confidence threshold hard to work.

Intuitively, pseudo labels with smaller entropy contribute more to the training process (Grandvalet and Bengio, 2004). We can leverage the entropy of labeled data to constrain that of unlabeled data, thus adapting the unlabeled data to model training. The noise introduced by low-confidence pseudo labels will be avoided through the entropy constraint, which further ensures the performance of the pseudo-labeling method. Besides, the adaptive constraint is more available due to eliminating manual settings.

To be specific, at training step t, assuming that labeled data x and unlabeled data x' are adopted, their respective outputs after the sigmoid operation by the model are $logit$ and $logit'$(namely y and y'), $\hat{y}$ and $\hat{y'}$ are labeled and unlabeled logits after the softmax operation. We denote K as the number of sentences in a document, then the labeled entropy $\mathcal{H}(\hat{y})$ and the unlabeled entropy $\mathcal{H}(\hat{y'})$ can be calculated as follows:

$$\begin{aligned} \mathcal{H}(\hat{y}) &= \sum_{k=1}^{K} \hat{y_k} \log(\hat{y_k}), \\ \mathcal{H}(\hat{y'}) &= \sum_{k=1}^{K} \hat{y'_k} \log(\hat{y'_k}), \end{aligned} \tag{3}$$

where k-th element of $\hat{y}$ and $\hat{y'}$ can be denoted as:

$$\begin{aligned} \hat{y_k} &= \frac{e^{logit_k}}{\sum_{k=1}^{K}(e^{logit_k})}, \\ \hat{y'_k} &= \frac{e^{logit'_k}}{\sum_{k=1}^{K}(e^{logit'_k})}. \end{aligned} \tag{4}$$

Then, the final selection constraint would be:

$$sel(y') = \begin{cases} 1, & \mathcal{H}(\hat{y'}) < \mathcal{Z} \cdot \mathcal{H}(\hat{y}), \\ 0, & otherwise, \end{cases} \quad (5)$$

where $sel(y') = 1$ means that y' is selected as a pseudo label and 0 otherwise. $\mathcal{Z}$ is the normalization factor and $\mathcal{Z} = dim_{x'}/dim_x$, aimed at alleviating the error propagation caused by inconsistent output vectors dimensions. $dim_{\hat{y'}}$ and $dim_{\hat{y}}$ denote the dimensions of $\hat{y'}$ and $\hat{y}$ respectively.

Ramp-up Pseudo-labels Exploitation. Although the reliability of entropy-constrainted pseudo-labels has been improved, the downside is that noises in them still exist, especially early in training. To mitigate this issue, we set a linearly increasing probability p_t in each epoch to select the filtered pseudo labels again:

$$p_t = min(1, \frac{t}{\tau_{epoch}}), \quad (6)$$

where t is the current epoch, and p_t denotes that the pseudo labels filtered in epoch t will be selected with probability p_t. τ_{epoch} is a hyper-parameter, which can be set according to the labeled data size.

For the way of adding pseudo labels to the labeled dataset, we draw on the idea of early stopping (Prechelt, 1996). As shown in the Algorithm 1 (lines 15-21), if ROUGE evaluated on the validation set for three consecutive rounds is in a downward trend, all filtered pseudo labels will be added to the labeled dataset. This procedure can effectively prevent the over-fitting phenomenon caused by repeated training of limited labeled data.

Algorithm 1 Training Procedure for Consistency Learning and Pseudo Labeling

Input: (x, y^*): the labeled data pair.
x' and x'': the unlabeled data and its augmented data.
R: the ratio of unlabeled and labeled data in each training step.
$step_{total}$: the total training step.
$val-interval$: the cycle of validation.

1: $t \leftarrow 0$; $psd \leftarrow []$; $val \leftarrow []$; $p_t \leftarrow 0$
2: **while** $t < step_{total}$ **do**
3: y is the output of x after $sigmoid$ by the model
4: $\hat{y} \leftarrow$ Eq. 4, $\mathcal{H}(\hat{y}) \leftarrow$ Eq. 3
5: **for** $r \leftarrow 1$ to R **do**
6: y' is the output of x' after $sigmoid$ by the model
7: $\hat{y'} \leftarrow$ Eq. 4, $\mathcal{H}(\hat{y'}) \leftarrow$ Eq. 3
8: $\mathcal{Z} \leftarrow dim_{\hat{y'}}/dim_{\hat{y}}$
9: **if** $\mathcal{H}(\hat{y'}) < \mathcal{Z} \cdot \mathcal{H}(\hat{y})$ **then**
10: Append (x', y') to psd according to p_t
11: **end if**
12: **end for**
13: $\mathcal{L}_l \leftarrow$ Eq. 1; $\mathcal{L}_u \leftarrow$ Eq. 2; $p_t \leftarrow$ Eq. 6
14: $\mathcal{L} \leftarrow \mathcal{L}_l + \omega(t)\mathcal{L}_u$, update the model
15: **if** t % $val-interval$ is 0 **then**
16: Append ROUGE of validation sets to val
17: **end if**
18: **if** the last three values of val monotonically decrease **then**
19: Merge pse to the labeled dataset
20: Stop pseudo-labels exploitation
21: **end if**
22: $t \leftarrow t + 1$
23: **end while**

4 Experimental Setup

4.1 Datasets

We conduct experiments on the following two datasets: (1) **CNN/DailyMail** (Hermann et al., 2015) includes news articles and corresponding extractive highlights. We use the standard splits (Hermann et al., 2015) for validation and testing. (2) **BBC XSum** (Narayan et al., 2018) provides a high level of abstraction. It has one-sentence summaries and is more abstractive than the CNN/DailyMail dataset. We obtain both labeled and unlabeled data from the entire dataset. Specifically, we divide parts of the original dataset into labeled data. For the rest, we delete the labels and treat them as unlabeled data.

4.2 Baselines and Evaluation Metrics

We focus on leveraging BERTSUMEXT (Liu and Lapata, 2019; Liu, 2019) for summarization [2]. To verify the effectiveness of our semi-supervised learning method in low-resource scenarios, we compare our method with the BERTSUMEXT of supervised learning. We also release the rule-based baselines — LEAD-3 on the CNN/DailyMail dataset and LEAD-1 on the BBC XSum dataset(excluding

[2]BERTSUMEXT is the variant of BERT, which builds several summarization-specific layers stacked on top of the BERT outputs including Simple Classifier, Transformer, and RNN. Our experiments mainly adopt BERT with the plainest Simple Classifier layer due to insignificant performance differences among the three layers.

Method	Labeled Data	*CNN/Daliy Mail*								
		ROUGE-1			ROUGE-2			ROUGE-L		
		P	R	F1	P	R	F1	P	R	F1
ORACLE		*43.63*	*58.77*	*48.35*	*23.88*	*31.77*	*26.28*	*40.30*	*54.13*	*44.61*
LEAD-3		34.50	51.94	40.04	14.81	22.44	17.21	31.16	46.86	36.14
Supervised (BERT)	10	31.47	41.92	34.42	11.84	15.65	12.89	28.29	37.61	30.91
	100	34.60	49.93	39.35	14.80	21.37	16.81	31.25	45.03	35.51
	1000	35.06	52.30	40.38	15.33	22.93	17.70	31.75	47.32	36.69
CPSUM w. Soft PlS	10	31.53	42.44	34.62	11.45	15.58	13.09	28.28	38.06	31.10
	100	34.77	52.04	40.22	15.02	22.56	17.37	31.42	46.96	36.32
	1000	35.17	53.23	40.93	15.48	23.49	18.01	31.85	48.14	37.04
CPSUM w. Hard PlS	10	31.69	42.78	34.94	11.63	15.78	13.26	28.58	38.33	31.37
	100	34.92	52.57	40.52	15.23	22.98	17.67	31.56	47.45	36.60
	1000	35.21	53.26	41.02	15.51	23.53	18.08	31.89	48.16	37.10
Method	**Labeled Data**	***BBC XSum***								
		ROUGE-1			ROUGE-2			ROUGE-L		
		P	R	F1	P	R	F1	P	R	F1
ORACLE		*31.37*	*30.51*	*29.57*	*9.74*	*9.07*	*8.86*	*22.96*	*22.07*	*21.47*
LEAD-1		17.12	16.69	16.32	1.68	1.66	1.60	12.59	12.24	11.96
Supervised (BERT)	10	18.21	15.92	16.13	2.17	1.86	1.90	13.68	11.81	12.01
	100	18.06	16.49	16.43	2.20	1.90	1.93	13.84	11.85	12.14
	1000	18.23	17.55	16.94	2.28	2.24	2.14	13.52	12.82	12.44
CPSUM w. Soft PlS	10	18.27	15.98	16.23	2.21	1.95	1.95	13.70	11.82	12.04
	100	18.75	16.54	16.82	2.29	2.02	2.04	13.96	12.10	12.35
	1000	18.95	17.35	17.22	2.38	2.22	2.17	14.05	12.77	12.71
CPSUM w. Hard PlS	10	18.29	15.98	16.25	2.21	1.96	1.95	13.68	11.83	12.05
	100	18.79	16.57	16.93	2.31	2.05	2.08	14.00	12.15	12.40
	1000	18.97	17.41	17.29	2.39	2.26	2.18	14.09	12.76	12.73

Table 1: Low-resource performance of ROUGE results on **CNN/DailyMail** and **BBC XSum** dataset. The best results for each group on all target corpora with 10, 100, and 1000 labeled examples are in-bold.

the one-line summary (Narayan et al., 2018)). The ground truth labels, which we call ORACLE, are extracted using the greedy approach. We use 10/100/1000 labeled data for supervised learning, and evaluate the summarization performance by ROUGE (Lin, 2004) in this paper, where R-1, R-2, and R-L are variants used to measure the overlap of unigrams, bigrams, and longest common subsequences between system and reference summaries.

4.3 Implementation Details

During training on the CNN/DailyMail dataset, the documents are truncated to 512 tokens, and the summaries are limited to 128 tokens. These two numbers are 512 and 64 for the BBC XSum dataset. Generally, semi-supervised learning performs a larger data size on unlabeled data than labeled data to fully use large quantities of unlabeled data. Therefore, we feed 1 batch of labeled data and 4 batches of unlabeled data into the framework in each training step, which is found to perform effectively by implementing the different proportions of labeled and unlabeled data. We use a batch size of 4 for labeled and unlabeled data.

We set τ_{epoch} to 30/15/5 epochs for training 10/100/1000 labeled data. All models are trained for 500/5000/20000 steps with 10/100/1000 labeled data on 3 Tesla V100 GPUs. The learning rate starts at 2e-3 and decay every 1000 steps. We also perform a linear warmup method to increase the learning rate smoothly from 0 to 2e-3 during 2000 steps at the beginning of training.

5 Experimental Results

5.1 Results on Dataset with long Summaries

The upper part of Table 1 shows the results on the CNN/DailyMail dataset. As shown, the model's performance improves as more labeled data becomes available. In the case where the data size is 1000, our method CPSUM achieves a +0.89 point improvement in R-1 and a +0.90 point improvement in R-L, compared with the traditional rule-based LEAD-3 on the CNN/DailyMail dataset.

Compared with the supervised baseline, CPSUM performs salient in all data sizes. The improve-

Table 2: **Ablation study on the effectiveness enhancement with ROUGE (F1) from different components based on CNN/DailyMail and BBC XSum dataset of 100 labeled data with hard pseudo labels**, including supervised learning, consistency regularization (CR), all pseudo labels (PlS [all]), and filtered pseudo labels (PlS [filtered]).

Components				Datasets					
				CNN/Daily Mail			*XSum*		
Supervised	CR	PlS [all]	PlS [filtered]	ROUGE-1	ROUGE-2	ROUGE-L	ROUGE-1	ROUGE-2	ROUGE-L
✓				39.35	16.81	35.51	16.43	1.93	12.14
✓	✓			39.62	16.94	35.79	16.52	1.96	12.04
✓	✓	✓		39.39	16.68	35.40	16.15	1.90	12.00
✓	✓		✓	**40.52**	**17.67**	**36.60**	**16.93**	**2.08**	**12.40**

Table 3: The number of pseudo labels w/o or w. ([all] or [filtered]) entropy-constrained filtering and the epochs to start appending pseudo labels for training.

CNN/Daily Mail					
Labeled Data	PlS [all]		PlS [filtered]		Epoch
10	960	9600%	248	2480%	24
100	9600	9600%	248	2480%	24
1000	28000	2800%	4675	467.5%	7
BBC XSum					
Labeled Data	PlS [all]		PlS [filtered]		Epoch
10	1200	12000%	288	2880%	30
100	12000	12000%	4043	4043%	30
1000	44000	4400%	11737	1173.7%	11

ments are most evident in the case of 100 labeled data with hard pseudo labels, where R-1/R-2/R-L increase +1.17/+0.86/+1.09 points compared with the supervised ones, far exceeding the baseline performance. These results all indicate the effectiveness of CPSUM in low-resource scenarios.

Moreover, soft or hard PlS also affects performance differently. We take 100 labeled data as an example. Although when soft labels are selected, CPSUM has obtained a +0.87 point improvement in R-1 compared with supervised learning, the adoption of hard pseudo labels still allows R-1 to continue to increase a +0.30 point upon soft pseudo labels. The results occur probably because soft labels are more ambiguous and have less information for extractive summarization than hard labels, so the use of hard pseudo labels will result in better performance than soft labels.

5.2 Results on Dataset with Short Summaries

We also conduct experiments to verify if CPSUM would be equally effective on the abstractive BBC XSum dataset, as shown in the lower part of Table 1. Identically, CPSUM outperforms supervised learning in all data sizes. For the better-performing hard pseudo labels, when labeled data sizes are 100 and 1000, CPSUM achieves remarkable performance. When there are only 10 labels, the performances of soft and hard PIS are indifferent, but they are better than the supervised method and LEAD-1. The results on XSum represent that CPSUM is also effective in generating extractive summaries.

5.3 Analysis and Discussion

The Size of Selected Pseudo Labels. Table 3 shows the number of pseudo labels and the epochs to start adding pseudo labels with different labeled data sizes. In the case where labeled data sizes are 10 and 100, the numbers of filtered pseudo labels are much larger than that of labeled data.

Nevertheless, when the labeled data size is 1000, the filtered pseudo labels increase by only 467% relative to labeled data on the CNN/DailyMail dataset, which means the demand for pseudo data can be relatively reduced in the case of training on 1000 labeled data. The condition occurs because when the labeled data size is large, the speed of convergence is faster compared with a smaller labeled data size, according to the epoch determined by the pseudo label exploitation strategy (shown in the last column of Table 3).

Various Components Study. This study aims to verify the effectiveness enhancement of different components, including supervised learning, consistency regularization, all pseudo labels, and filtered pseudo labels in Table 2 . All the ablations are conducted with 100 labeled data in both datasets.

We take the CNN/DailyMail dataset as an example for analysis. The purely supervised learning is treated as the baseline, which achieves R-1/R-2/R-L of 39.35/16.81/35.51. After adding the consistency regularization method, R-1/R-2/R-L slightly have increased by +0.27/+0.13/+0.28 points. This means that although consistent learning will improve supervised learning, there are still factors that limit its performance, possibly the disadvanta-

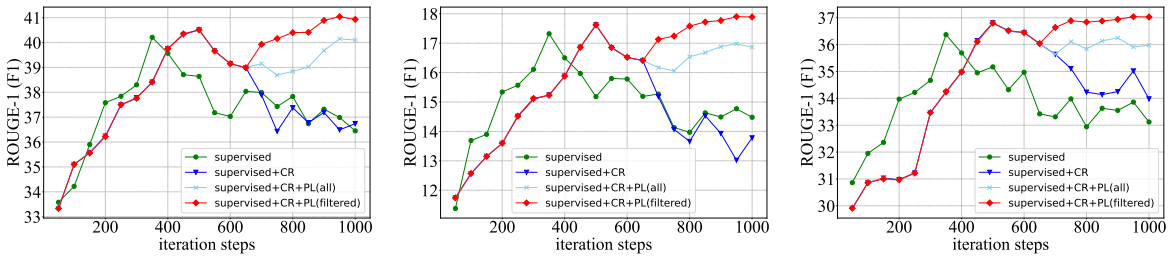

(a) ROUGE-1 (b) ROUGE-2 (c) ROUGE-L

Figure 3: **The first 1000 steps change curve of ROUGE (F1) metrics on the CNN/DailyMail dataset of 100 labeled data.** All the experiments are validated every 50 steps. Before adding pseudo labels, the performance of consistency learning achieves the best at step 500(epoch 20), and subsequently, three consecutive validation results monotonically decrease, so we add pseudo labels at step 650. In all three metrics line charts, the red lines and skyblue lines rise after step 650 due to the addition of pseudo labels.

Table 4: **Ramp-up hyper-parameters τ_{epoch} exploration based on the CNN/DailyMail dataset of 100 labeled data with hard pseudo labels**.

τ_{epoch}	PlS size	ROUGE-1	ROUGE-2	ROUGE-L
10	4273	40.46	17.63	36.54
12	4072	40.47	17.64	36.55
15	3514	**40.52**	**17.67**	**36.60**
18	3698	40.47	17.64	36.55
20	3514	40.46	17.60	36.53
22	3316	40.41	17.55	36.46

geous noise of the augmented data.

Further, we compare two programs of selecting pseudo labels for labeled samples exploitation. First, all the pseudo labels are selected without filtering (in the 3-rd row), which obtains a slight improvement in R-1 but declines in R-2/R-L due to plenty of unreliable pseudo labels. The final method, which combines all our contributions, including consistency regularization and filtered pseudo labels (in the 4-th row), achieves superior results with +1.17/+0.86/+1.09 improvements in R-1/R-2/R-L, compared with the supervised baseline.

Fig. 3 shows the changing trend of ROUGE of all the ablations in the validation set during the first 1000 steps. As shown by the curve corresponding to the pseudo-labeling method, our frameworks with pseudo labels can sufficiently alleviate the overfitting caused by few-sample data. Additionally, Compared with not filtering reliable pseudo labels, the entropy-constrained method enables the model to improve upon the baseline effectively.

Ramp-up hyper-parameters Exploration. Our entropy-constrained pseudo labeling method introduces a hyper-parameter τ_{epoch} in the procedure of the ramp-up pseudo-labels exploitation. We tweak the hyper-parameter in a rational range and select it based on the CNN/DailyMail dataset with 100 labeled data. As shown in Fig. 4, results show that when τ_{epoch} is 15, CPSUM performs best. We find that in the supervised learning with 100 labeled data, CPSUM achieves the best in the 20-th epoch on the validation set. This indicates that in the case where the hyper-parameter τ_{epoch} is slightly smaller than the optimal training epoch, the performance will be the best. If τ_{epoch} is too tiny, the pseudo label data increase but become noisier. If τ_{epoch} is too large, the pseudo label data will decrease, and then high-quality pseudo labels will also decrease.

6 Conclusions

In this paper, we present a new perspective on effectively using consistency training and pseudo labeling to improve low resource extractive summarization over an insufficiently labeled dataset. With substituting simple noise injection operations with advanced data augmentation and constraining pseudo label selection with average entropy, our method brings substantial improvements compared with the supervised learning frameworks. Since our proposed model is orthogonal to the methods that using pre-trained models, we believe our model can be further boosted by taking other salient pre-trained models to initialize the text representations. Additionally, although we use ramp-up exploitation to control the adverse entropy effect brought by the early model, incorrect prediction cannot be avoided. An impeccable minimum entropy regularization method can be exploited in the future.

Acknowledgements

This work is supported in part by the National Natural Science Foundation of China (No.U20B2053).

References

Philip Bachman, Ouais Alsharif, and Doina Precup. 2014. Learning with pseudo-ensembles. In *NeurIPS*, pages 3365–3373.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: pre-training of deep bidirectional transformers for language understanding. In *NAACL-HLT*, pages 4171–4186. Association for Computational Linguistics.

Yves Grandvalet and Yoshua Bengio. 2004. Semi-supervised learning by entropy minimization. In *NeurIPS*, pages 529–536.

Karl Moritz Hermann, Tomás Kociský, Edward Grefenstette, Lasse Espeholt, Will Kay, Mustafa Suleyman, and Phil Blunsom. 2015. Teaching machines to read and comprehend. In *NeurIPS*, pages 1693–1701.

Xiaoqi Jiao, Yichun Yin, Lifeng Shang, Xin Jiang, Xiao Chen, Linlin Li, Fang Wang, and Qun Liu. 2020. Tinybert: Distilling BERT for natural language understanding. In *EMNLP (Findings)*, volume EMNLP 2020, pages 4163–4174. Association for Computational Linguistics.

Yoon Kim. 2014. Convolutional neural networks for sentence classification. In *EMNLP*, pages 1746–1751. Association for Computational Linguistics.

Samuli Laine and Timo Aila. 2017. Temporal ensembling for semi-supervised learning. In *ICLR, poster*. OpenReview.net.

Dong-Hyun Lee et al. 2013. Pseudo-label: The simple and efficient semi-supervised learning method for deep neural networks. In *Workshop on challenges in representation learning, ICML*, volume 3, page 896.

Chin-Yew Lin. 2004. ROUGE: A package for automatic evaluation of summaries. In *Text Summarization Branches Out*, pages 74–81, Barcelona, Spain. Association for Computational Linguistics.

Chen Liu, Mengchao Zhang, Zhibing Fu, Panpan Hou, and Yu Li. 2021a. Flitext: A faster and lighter semi-supervised text classification with convolution networks. In *EMNLP*, pages 2481–2491. Association for Computational Linguistics.

Yang Liu. 2019. Fine-tune BERT for extractive summarization. *CoRR*, abs/1903.10318.

Yang Liu and Mirella Lapata. 2019. Text summarization with pretrained encoders. In *EMNLP-IJCNLP*, pages 3728–3738. Association for Computational Linguistics.

Ye Liu, Jianguo Zhang, Yao Wan, Congying Xia, Lifang He, and Philip S. Yu. 2021b. HETFORMER: heterogeneous transformer with sparse attention for long-text extractive summarization. In *EMNLP*, pages 146–154. Association for Computational Linguistics.

Takeru Miyato, Shin-ichi Maeda, Masanori Koyama, and Shin Ishii. 2019. Virtual adversarial training: A regularization method for supervised and semi-supervised learning. *IEEE Trans. Pattern Anal. Mach. Intell.*, 41(8):1979–1993.

Ramesh Nallapati, Feifei Zhai, and Bowen Zhou. 2017. Summarunner: A recurrent neural network based sequence model for extractive summarization of documents. In *AAAI*, pages 3075–3081. AAAI Press.

Shashi Narayan, Shay B. Cohen, and Mirella Lapata. 2018. Don't give me the details, just the summary! topic-aware convolutional neural networks for extreme summarization. In *EMNLP*, pages 1797–1807. Association for Computational Linguistics.

Jeffrey Pennington, Richard Socher, and Christopher D. Manning. 2014. Glove: Global vectors for word representation. In *EMNLP*, pages 1532–1543. Association for Computational Linguistics.

Lutz Prechelt. 1996. Early stopping-but when? In *Neural Networks: Tricks of the Trade*, volume 1524 of *Lecture Notes in Computer Science*, pages 55–69. Springer.

Mamshad Nayeem Rizve, Kevin Duarte, Yogesh Singh Rawat, and Mubarak Shah. 2021. In defense of pseudo-labeling: An uncertainty-aware pseudo-label selection framework for semi-supervised learning. In *ICLR*. OpenReview.net.

Kihyuk Sohn, David Berthelot, Nicholas Carlini, Zizhao Zhang, Han Zhang, Colin Raffel, Ekin Dogus Cubuk, Alexey Kurakin, and Chun-Liang Li. 2020. Fixmatch: Simplifying semi-supervised learning with consistency and confidence. In *NeurIPS*.

Antti Tarvainen and Harri Valpola. 2017. Mean teachers are better role models: Weight-averaged consistency targets improve semi-supervised deep learning results. In *ICLR, Workshop*. OpenReview.net.

Vikas Verma, Alex Lamb, Juho Kannala, Yoshua Bengio, and David Lopez-Paz. 2019. Interpolation consistency training for semi-supervised learning. In *IJCAI*, pages 3635–3641. IJCAI.org.

Yuchao Wang, Haochen Wang, Yujun Shen, Jingjing Fei, Wei Li, Guoqiang Jin, Liwei Wu, Rui Zhao, and Xinyi Le. 2022. Semi-supervised semantic segmentation using unreliable pseudo-labels. *CoRR*, abs/2203.03884.

Xing Wu, Shangwen Lv, Liangjun Zang, Jizhong Han, and Songlin Hu. 2019. Conditional BERT contextual augmentation. In *ICCS (4)*, volume 11539 of *Lecture Notes in Computer Science*, pages 84–95. Springer.

Qizhe Xie, Zihang Dai, Eduard H. Hovy, Thang Luong, and Quoc Le. 2020a. Unsupervised data augmentation for consistency training. In *NeurIPS*.

Qizhe Xie, Minh-Thang Luong, Eduard H. Hovy, and Quoc V. Le. 2020b. Self-training with noisy student improves imagenet classification. In *CVPR*.

Ming Zhong, Pengfei Liu, Yiran Chen, Danqing Wang, Xipeng Qiu, and Xuanjing Huang. 2020. Extractive summarization as text matching. In *ACL*, pages 6197–6208. Association for Computational Linguistics.

Ming Zhong, Pengfei Liu, Danqing Wang, Xipeng Qiu, and Xuanjing Huang. 2019a. Searching for effective neural extractive summarization: What works and what's next. In *ACL , Volume 1: Long Papers*, pages 1049–1058. Association for Computational Linguistics.

Ming Zhong, Danqing Wang, Pengfei Liu, Xipeng Qiu, and Xuanjing Huang. 2019b. A closer look at data bias in neural extractive summarization models. *CoRR*, abs/1909.13705.

Qingyu Zhou, Furu Wei, and Ming Zhou. 2020. At which level should we extract? an empirical analysis on extractive document summarization. In *COLING*, pages 5617–5628. International Committee on Computational Linguistics.

Qingyu Zhou, Nan Yang, Furu Wei, Shaohan Huang, Ming Zhou, and Tiejun Zhao. 2018. Neural document summarization by jointly learning to score and select sentences. In *ACL (1)*, pages 654–663. Association for Computational Linguistics.

Question Generation Based on Grammar Knowledge and Fine-grained Classification

Yuan Sun[1,2,*] **, Sisi Liu**[1,2] **, Zhengcuo Dan**[1,2] **, Xiaobing Zhao**[1,2,*]
[1]Minzu University of China
[2]National Language Resource Monitoring & Research Center Minority Languages Branch
tracy.yuan.sun@gmail.com, liusisi.s@qq.com, nmzxb_cn@163.com
[*]Corresponding Author: Yuan Sun, Xiaobing Zhao

Abstract

Question generation is the task of automatically generating questions based on given context and answers, and there are problems that the types of questions and answers do not match. In minority languages such as Tibetan, since the grammar rules are complex and the training data is small, the related research on question generation is still in its infancy. To solve the above problems, this paper constructs a question type classifier and a question generator. We perform fine-grained division of question types and integrate grammatical knowledge into question type classifiers to improve the accuracy of question types. Then, the types predicted by the question type classifier are fed into the question generator. Our model improves the accuracy of interrogative words in generated questions, and the BLEU-4 on SQuAD reaches 17.52, the BLEU-4 on HotpotQA reaches 19.31, the BLEU-4 on TibetanQA reaches 25.58.

1 Introduction

Question generation is an essential research direction in natural language generation, which aims to generate a grammatical question based on the given paragraph and answer, and the question can be answered by the given answer. Recently, question generation has received extensive attention in academia and industry. It can automatically generate training data for question answering systems (Du and Cardie, 2018) and reading comprehension tasks (Heilman and Smith, 2010). It can also simulate user questions in the field of education (Du et al., 2017), and guide the machine to ask questions actively in the field of dialogue (Shum et al., 2018), and strengthen human-computer interaction (Mostafazadeh et al., 2016).

In recent years, the research on question generation has achieved great success, but we analyze the questions generated by an advanced answer-aware s2s model (Zhao et al., 2018) and find that some question types and answers did not match.

Paragraph	..., To ensure a space, in 1947, ABC submitted five applications for television station licenses, ...
Answer	1947
Gold	**in what year** did abc submit licenses for 5 television stations ?
Generated	**what** is the name of the space that was used to ensure a space on television station ?

Table 1: English question generation example.

As shown in Table 1, the question type of the gold question is to ask questions about time and use "in what year", but the interrogative word of generated question is "what".

To solve the problem, some studies have proposed to use answer information to improve the interrogative words accuracy of generated questions (Sun et al., 2018; Zhou et al., 2019b; Kang et al., 2019). However, they did not consider the structural and grammatical features of the answers. Some studies classify question types according to the category of interrogative words, and use the first word of the question as the question type. However, we analyze the SQuAD (Rajpurkar et al., 2016) and find that the first word of question is not always an interrogative word, and there is a mismatch between interrogative words and question types. For example, "in what year" is a question about time, but using the classification of interrogative word will classify it into "what" category. Therefore, using interrogative words to classify questions will also introduce wrong information.

To solve above problems, this paper constructs a question type classifier based on a pre-trained language model and a question generator. We integrate the grammatical knowledge of question into the question type classifier, and classify the type according to the inquiry object of the question, we apply this method to English and Tibetan. The grammatical features of Tibetan are different from those of English, for example, interrogative words in English often appear at the beginning of the ques-

Proceedings of the 29th International Conference on Computational Linguistics, pages 6457–6467
October 12–17, 2022.

tions, but in Tibetan, they will appear in multiple positions of questions, and Tibetan interrogative words have ambiguity.

Tibetan is a phonetic writing script, which belongs to the consonant character type. It is divided into two parts: consonants and vowels. The grammatical rules in Tibetan are relatively complex, but there are clear organizational forms and verb changes. Tibetan is composed of syllables, a syllable containing one or up to seven characters (Nuo et al., 2015; Liu et al., 2011), and syllables are separated by an intersyllable marker "·", which is a simple superscripted dot, for example, བཀྲ་/ཤིས་/ཀྱི་/ཆེད་/ལས་/ནི་/ཅི་/ཞིག་/ཡིན། (What is Tashi's major?).

In this paper, we conduct experiments on English and Tibetan datasets respectively, the experimental results show that the BLEU-4 on SQuAD is 17.52, which is 1.21 higher than the experimental result of Zhou et al (Zhou et al., 2019b). BLEU-4 on HotpotQA is 19.31, which is 3.9 higher than the result of Xie et al (Xie et al., 2020). BLEU-2 on TibetanQA is 35.07, which is 9.73 highter than the result of Sun et al (Sun et al., 2021a). The main contributions of this paper are as follows.

(1) Since the classification of questions based on interrogative words will introduce errors, this paper proposes a fine-grained classification method to classify questions according to the object of the question and the lexical features of the answer to guide question generation.

(2) In low-resource languages such as Tibetan, considering the complexity of Tibetan grammar rules and the training data is small, this paper proposes to integrate interrogative grammatical rules to improve the accuracy of question types, and to improve the performance of question generation.

2 Related Work

Question generation mainly adopts two methods, rule-based and neural network-based. The method based on rules and templates is to define some heuristic rules on the syntax tree to convert a sentence into a question (Heilman and Smith, 2010; Mazidi and Nielsen, 2014; Labutov et al., 2015), but the generated questions have limited diversity and poor portability. With the emergence of large-scale and high-quality machine reading comprehension datasets, such as SQuAD (Rajpurkar et al., 2016), MARCO (Nguyen et al., 2016), HotpotQA (Yang et al., 2018) and TriviaQA (Joshi et al., 2017), neural network-based question generation has made great progress.

Existing works are mainly based on seq2seq architecture with attention mechanism and copy mechanism (Du et al., 2017; Zhao et al., 2018; Zhou et al., 2017). To generate answer-related questions, the methods of using the answer position as input or encoding the answer are proposed (Zhou et al., 2017; Song et al., 2018a; Kim et al., 2019). Song et al. (Song et al., 2018b) propose to use a multi-view matching method to calculate the similarity between the answer representation and the hidden layer of the paragraph in three ways, and finally the BLEU-4 of this method reaches 13.98 on SQuAD. Wang et al. (Wang et al., 2020a) introduce a novel hidden answer pivot module to model the hidden answer information to generate questions. Li et al. (Li et al., 2019) jointly model unstructured sentence and structured answer-relevant relation to generate questions. Murakhovs' ka et al. (Murakhovs' ka et al., 2021) train a unified QG model and generate different cognitive levels questions by controlling the answer type. To solve the two problems, the mismatch between the generated question words and answers, and the words copied by the model are far from and irrelevant to the answer. Sun et al. (Sun et al., 2018) propose an answer-focused and position-aware neural question generation model. The final experimental result BLEU-4 reaches 15.64 on SQuAD. To address the challenge of long text on question generation model based on seq2seq, Tuan et al. (Tuan et al., 2020) propose to encode context information in the long text so that model can obtain more information to improve the performance of question generation. Zhao et al. (Zhao et al., 2018) propose a paragraph-level question generation with maxout pointer and gated self-attention, which can effectively utilize paragraph-level context and outperform sentence-level context performance, and finally achieve BLEU-4 of 16.38 on SQuAD. To improve the answerability of the question and the relevance of the original text, Xie et al. (Xie et al., 2020) design three different reward discriminators to calculate reward scores and feed them back to the question generator to improve the fluency, relevance and answerability of generated questions, and finally BLEU-4 reach 15.43 on the HotpotQA (Yang et al., 2018). The above methods improve the relevance between the generated question and the answer, but there is a problem that the generated

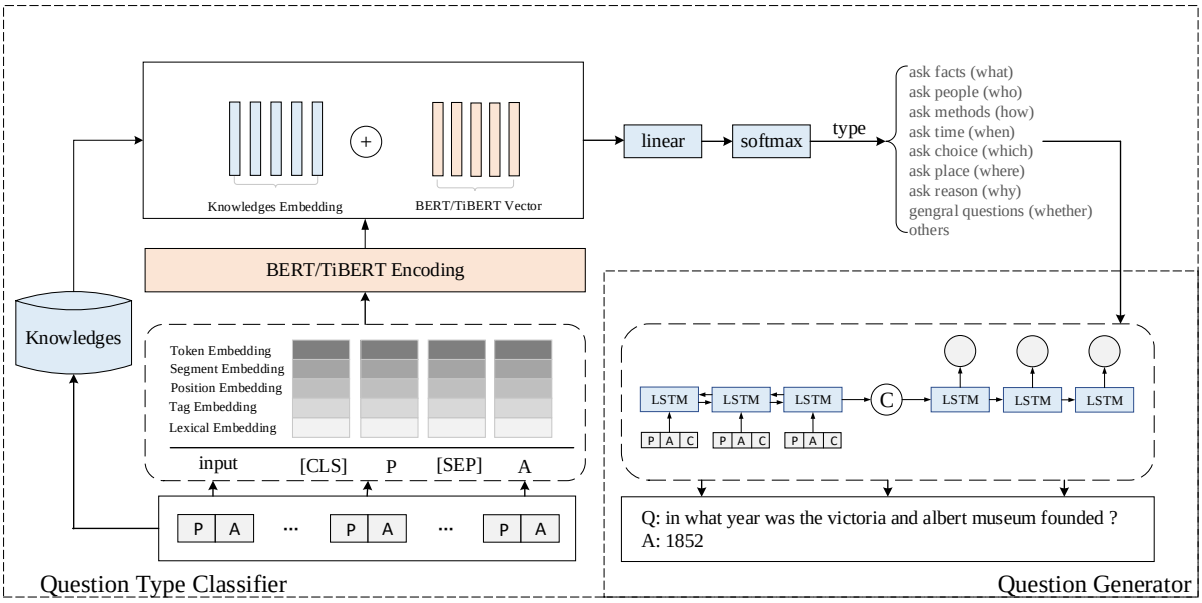

Figure 1: The overall architecture of our model. The question type classifier is used to classify passages and answers to obtain the question types, and then the types are fed into the question generator.

question type and the answer type do not match. To solve the problems, Ma et al. (Ma et al., 2020) propose to integrate sentence-level semantic matching and answer position inference. Zhou et al. (Zhou et al., 2019b) propose a unified model to predict the question type and to generate question. Wang et al. (Wang et al., 2020b) propose use answer-aware initialization module to introduce document and answer to decoder, and design a semantic-rich fusion attention to support decoder. The finally BLEU-4 reach 17.54 on HotpotQA. The above research are applications of question generation on English, there are few related research on Tibetan. Sun et al. (Sun et al., 2021a) construct a Tibetan reading comprehension dataset named TibetanQA, and propose a Tibetan question generation model based on a seq2seq. By introducing an attention mechanism to capture the key semantic information related to the answer and generate an answer-aware question. We follow the method and analyze the generated questions, and find that the accuracy of interrogative words in Tibetan is only 26.03%, which seriously hinders the research on Tibetan question generation. Our method of integrating Tibetan grammar knowledge can significantly improve accuracy of interrogative words in Tibetan question generation.

3 Model Description

The model architecture of this paper is shown in Figure 1. The question type classifier is used to predict the type of target question, and the prediction will be integrated into question generator to guide question generation.

3.1 Fine-grained Question Type Classifier

Different types of questions have different grammar rules, and choosing an accurate interrogative word according to the type of question is the key to generating high-quality questions. To obtain a more accurate question type, this paper divides the datasets according to the objects asked by the question. We use 9 categories to express the question patterns, such as "what" to refer to facts, "who" to refer to people, "how" to refer to methods, "when" to refer to time, "which" to refer to choice, and "where" to refer to a place, "why" to refer to reason, "whether" to refer to general questions and "others" to refer to others.

This paper constructs question type classifier based on BERT (Devlin et al., 2018), and fine-tune BERT with 9 categories data to obtain a classifier. BERT takes a sequence of less than 512 tokens as input and outputs a representation of that sequence. The first token of sequence is always $[CLS]$, which is originally designed for classification tasks, and the other special token $[SEP]$ is used to separate segments. For a given passage P and answer A, the input sequence X as Equation (1).

$$X = ([CLS], P, [SEP], A, [SEP]) \quad (1)$$

In the embedding, we integrate the lexical features of part-of-speech tagging of the answers. The final input embedding $E_f = [E_t; E_s; E_p; E_{tag}; E_l]$,

where E_t is token embedding which contains passage and answer, E_s is segment embedding, E_p is position embedding, E_{tag} is answer tag embedding, and E_l is the embedding of lexical features. The final $V_{[CLS]}$ contains the special categorical embedding, this paper takes $V_{[CLS]}$ as the representation of the sequence, which learns how to represent context and answer information. We add a linear and softmax on top of BERT to predict the probability of question type. The model parameters are shown in Table 2.

Parameters	Values
epoch	10
batch_size	64
pad_size	512
hidden_size	768
learning_rate	5e-5

Table 2: Parameters of question type classifier.

3.2 Question Type Classifier with Grammatical Knowledge

Fine-grained question generation is based on pre-trained language models. Currently, various public multi-language pre-training models such as mBERT (Pires et al., 2019), XLM-RoBERTa (Conneau et al., 2019) and T5 (Raffel et al., 2020) don't contain low-resource languages such as Tibetan. Sun et al. (Sun et al., 2022) construct a Tibetan pre-trained language model named TiBERT. We construct a Tibetan question type classifier based on TiBERT.

Tibetan questions have complex grammatical rules. Different from the situation that English interrogative words often appear at the beginning of sentences, Tibetan interrogative words will appear in various positions of the sentence, and they have ambiguity problems. We will introduce that in detail in section 4. Since Tibetan has the above characteristics, it is difficult to get a correct question type. Thus, this paper integrates Tibetan grammatical knowledges in the question type classifier.

3.3 Question Generator

We follow the Tibetan question generation model based on the Sequence to Sequence Model proposed by Sun et al (Sun et al., 2021a). In the encoder, this paper fuses the output of the question type classifier and the paragraph as the input of the model, and uses a bidirectional LSTM to encode it.

$$h_t = LSTM(h_{t-1}, [P_t, A_t, C_t]) \quad (2)$$

$$H = [\overrightarrow{h_t}, \overleftarrow{h_t}]_{t=1}^{M} \quad (3)$$

where $\overrightarrow{h_t}$ and $\overleftarrow{h_t}$ represent left-to-right and right-to-left encodings, C_t is the question type predicted by the question type classifier based on paragraph P and answer A, and H is the vector representation encoded by BiLSTM, and then use the self-attention mechanism to process the encoded information. The self-attention mechanism allows the model to dynamically assign weights to different information, and finally obtain an answer-aware contextual representation.

The decoder uses an LSTM with an attention mechanism and a copy mechanism. At time step t, the decoder outputs

$$y_t = LSTM(h_{t-1}, y_{t-1}) \quad (4)$$

where h_{t-1} is the hidden state at time $t-1$, y_{t-1} is the previously output at time $t-1$, and then the model uses the output of the decoder and the contextual representation of the encoder to calculate the attention to obtain the generation probability and copy probability of the word. The probability determines whether the final word is generated from the vocabulary or copied from the context. The model is trained to minimize the negative log-likelihood of the target sequence. The model parameters are shown in Table 3.

4 Influence of Grammar Knowledge on Tibetan Question Generation

4.1 Influence of Interrogative Word Ambiguity

Interrogative words in English will only appear in question in most cases, but there are no fixed interrogative words in Tibetan. The same words may appear as interrogative words or appear as case particles or nouns, therefore, word ambiguity will occur. This paper analyzes interrogative words in Tibetan as follows.

Parameters	Values
param size	41,025,835
epoch	20 in English / 60 in Tibetan
embedding_size	300
hidden_size	300
learning_rate	0.1
dropout	0.3
max_len	400 in English / 1000 in Tibetan

Table 3: Parameters of question generation.

(1) There are no interrogative words indicating “which” and "where" in Tibetan, they usually appear as "གང་" or "གང་ཞིག་", which means “which” or “where” Therefore, the two categories of “which” and “where” are combined into the same category in the Tibetan dataset. However, in addition to appearing as an interrogative word, the "གང་" also appears in the sentence with other meanings. For example, "ཁོ་མོ་ཟླ་ཁ་**གང་**ན་སྨན་ཁང་**གང་ཞིག**་ལ་འགྲོ་རྩིས་ཡོད། (Which hospital is she going to go to in the full month?)" The syllable "གང་" appears in this sentence, but it is means “full”, the "གང་ཞིག" in the sentence is the interrogative word, which means “which”.

(2)"སུ་" means "who" in Tibetan, but the syllable "སུ་" also appear in a sentence in the form of a case particle, when the syllable "སུ་" appears as a case particle, the last character of the syllable before the "སུ་" is "ས", which is "སུ་" after removing the root letter, therefore, we use this rule to judge whether "སུ་" appears in the form of interrogative words. For example, "མགོན་པོ་ཀླུ་སྒྲུབ་ནི་སུའི་རྗེས་སུ་བཟུང་སྟེ་རབ་ཏུ་བྱུང་། (Who did the Nagarjuna Bodhisattva become a monk with?)", the interrogative word is "སུའི་", which means “whose”, and indicates “ownership”, the character "ས" appears before the syllable "སུ་", so"སུ་" is a case particle. However, this situation still exists. For example, "འཛམ་བུ་གླིང་པའི་མི་རྣམས་ཀྱི་ཚེ་ལོ་དཔག་མེད་ནས་བྲི་སྟེ་བཞི་ཁྲི་ལ་ཐུག་པའི་ཚེ་སངས་རྒྱས་སུ་བྱོན། (Which Buddha was born when the life span of people in Nanjiaobu continent was reduced from infinity to 40,000 years?)", here "སུ་" is interrogative word, which means "who", but the syllable "སུ་" appears in the grammatical form of case particle, because the words in front of the "སུ་" is a complete word "སངས་རྒྱས་(Buddha)", and it happens that the last syllable of the words is "ས". In this case, we will give priority to analyzing interrogative words other than "སུ་" , and analyze them in combination with the position of the interrogative words.

4.2 Influence of Tibetan Interrogative Words Position

Interrogative words in English questions usually appear at the beginning of the sentence, it’s syntax is fixed and the training data is large, the model can learn relevant grammar rules and interrogative word position information. But for low-resource languages such as Tibetan, the training data is small and the grammatical rules are complex, so it is difficult for the model to learn relevant knowledge. Therefore, this paper assists the model learning by adding additional grammar knowledge.

This paper analyzes the positions of interrogative words in Tibetan questions as shown in Table 4.

Type	Tibetan question words	Example sentences
Which/Where	གང་དག་/གང་/གང་ཞིག་/གང་ཞིག་/གང་	ཁྱོད་ཀྱི་དཔེ་ཆ་ནི་**གང་དག**་ཡིན། (**Which** are your book?) ཁྱོད་**གང**་ན་ཡོད། (**Where** are you?)
What	ཅི་ཞིག་/ཅི་	བཀྲ་ཤིས་ཀྱི་ཆེད་ལས་ནི་**ཅི་ཞིག**་ཡིན། (**What** is Tashi’s major?)
Who	སུ་	ཁྱོད་**སུ**་ཡིན། (**Who** are you?)
How	ཇི་ལྟར་/ཅི་ལྟར་/ག་འདྲ་/ཅི་འདྲ་	ཚོགས་འདུ་འདི་གྲ་སྒྲིག་**ཅི་ལྟར**་བྱེད་དགོས་པ་རེད། (**How** to prepare for this meeting?)
When	ནམ་/ནམ་ཞིག་	ཕ་ཡུལ་ལ་**ནམ**་ལོག་རྒྱུ་ཡིན། (**When** will you go back to your hometown?)
Why	ཅིའི་ཕྱིར་	དགེ་རྒན་གྱིས་**ཅིའི་ཕྱིར**་དེ་ལྟར་བཤད་པ་རེད། (**Why** did the teacher say that?)

Table 4: Position analysis of Tibetan interrogative words.

ཁ་དོག་/སྣ་ལྔ/འི་/མཚོའུ་/ནི་/ས་ཆ་/གང་/དུ་/ཡོད/།									
(Where is Wucai Lake located?)									
ng	/ ng	/ kg	/ ng	/ up	/ ng	/ rw	/kl	/ve	/xp
E_0 /	E_0 /	E_c /	E_0 /	E_0 /	E_0 /	E_q /	E_c /	E_0 /	E_0

Table 5: An interrogative word position template.

Tibetan interrogatives of "which" and "what" are usually located before modal particles and after verbs. The interrogative words of "who", between the subject and object, not only act as interrogative elements, but also act as predicate elements. The interrogative words of "how" is located between the adverbial and the predicate, with the adverbial in the front and the predicate in the back. The interrogative words of "when" and "why" are usually located before the predicate and after the pronoun or case particle. For the interrogative words of "where", it is located between subject and object, and it also act as predicate. In general, different interrogative words appear in different positions. Therefore, this paper adds lexical features and use rules hard-coded to encode the grammatical knowledge to the Tibetan question type classifier to improve the classification accuracy. A interrogative word position template is as shown in Table 5, where E_q represents the position of interrogative word and E_c represents the position of the case particle.

5 Experiment

5.1 Dataset

We conduct experiments on English dataset SQuAD (Rajpurkar et al., 2016), HotpotQA (Yang et al., 2018) and Tibetan dataset TibetanQA (Sun et al., 2021b). SQuAD is the first large-scale reading comprehension dataset containing natural questions. The articles come from Wikipedia and the dataset is constructed by crowdsourcing to ensure the diversity of questions. HotpotQA contains 113,000 Wikipedia-based question-answer pairs, the answers to the questions are based on multiple supporting documents, and the dataset provides sentence-level supporting fact. TibetanQA contains 1,513 articles and 20,000 question-answer pairs. The articles in the dataset are from yunzang website [1], covering 12 topics such as nature, culture, and education.

This paper counts the distribution of question types in the three datasets, as shown in Table 6. According to the composition of different datasets, this paper divides SQuAD and TibetanQA into 8 categories. The HotpotQA are divided into 9 categories. From Table 6, we can know the distribution of different categories is very uneven, so it is difficult to predict the correct question type.

5.2 Automatic Evaluation

This paper uses BLEU-1, BLEU-2, BLEU-3, BLEU-4 (Papineni et al., 2002) and ROUGE-L (Lin, 2004) to evaluate the performance of the question generation model.

5.3 Experimental Results of Question Generation

We use the question type classifier to classify paragraphs and answers to predict question types, the accuracy of SQuAD is 63.98%, the accuracy of HotpotQA is 55.68% , and the accuracy of TibetanQA is 77.95%. Then, the predicted question types are integrated into question generation, and the final experimental results are shown in Table 7.

To explore the upper bound of our model, and studies the effectiveness of question types on question generation, we use grammatical knowledge to extract the question type and directly fuse it with the paragraph as the input of the question generator.

We compare the performance of our model with previous state-of-the-art models, and the experimental results are shown in Table 7. We briefly introduce these models as follows.

(Zhou et al., 2017): They propose a seq2seq model with attention and copy mechanisms, and use POS and NER tags as lexical features for the encoder.

(Zhao et al., 2018): They propose paragraph-level question generation with maxout pointer mechanism and gated self-attention.

(Zhou et al., 2019a): They incorporate an auxiliary task of language model to the hierarchical multi-task learning structure to help question generation.

(Zhou et al., 2019b): They propose a unified model to predict question type and generate questions.

(Jia et al., 2020): They propose to incorporate paraphrase knowledge into question generation.

(Xie et al., 2020): They propose to optimize question generation with specific rewards.

(Sun et al., 2021a): They propose a seq2seq model with attention and copy mechanisms on Tibetan question generation.

[1] https://www.yongzin.com/

Dataset	What(%)	Who(%)	How(%)	When(%)	Which(%)	Where(%)	Why(%)	Whether(%)	Others(%)
SQuAD	52.45	10.65	10.58	11.73	6.71	4.76	1.37	-	1.76
HotpotQA	34.24	15.15	3.04	10.90	24.98	4.46	0.03	7.10	0.10
TibetanQA	25.22	5.83	21.74	6.48	34.01		0.55	-	5.17

Table 6: The proportion of question types on three datasets.

Datasets	Method	BLEU-1	BLEU-2	BLEU-3	BLEU-4	ROUGE-L
SQuAD	(Zhou et al., 2017)	-	-	-	13.29	-
	(Zhao et al., 2018)	44.51	29.07	21.06	15.82	-
	(Zhou et al., 2019a)	42.80	28.43	21.08	16.23	-
	(Zhou et al., 2019b)	43.11	29.13	21.39	16.31	-
	(Jia et al., 2020)	43.63	29.21	21.79	16.93	-
	Our Model	**47.28**	**31.35**	**23.02**	**17.52**	**46.78**
	Upper Bound	49.02	32.58	24.00	18.31	48.59
HotpotQA	(Zhou et al., 2017)	35.51	22.32	15.94	11.73	32.12
	(Zhao et al., 2018)	38.54	25.09	17.49	13.48	22.45
	(Xie et al., 2020)	37.97	-	-	15.41	35.12
	Our Model	**42.35**	**30.42**	**23.83**	**19.31**	**40.95**
	Upper Bound	45.76	33.04	25.97	21.12	43.50
TibetanQA	(Sun et al., 2021a)	-	25.34	-	-	36.47
	(Sun et al., 2021a) using our metrics	39.18	33.04	28.36	24.77	39.25
	Our Model	**42.45**	**35.07**	**29.64**	**25.58**	**43.28**
	Upper Bound	45.05	37.96	32.30	28.01	44.85

Table 7: Experimental results of question generation on three datasets.

From Table 7, we can know that our model outperforms other models on all metrics, and the effect of the model is significantly improved after the grammar knowledge is integrated into the question generation model. On the English dataset SQuAD, our model achieves 17.52 on BLEU-4, which is 1.21 higher than (Zhou et al., 2019b) and 46.78 on ROUGE-L, and the upper bound of our model can reach 18.31 on BLEU-4. On HotpotQA, our model achieves 19.31 in BLEU-4, which is 3.9 higher than (Xie et al., 2020), and the upper bound of our model can reach 21.12. On TibetanQA, our model achieves 35.07 on BLEU-2, which is 9.73 higher than (Sun et al., 2021a), and 43.28 on ROUGE-L, which is 6.81 higher than (Sun et al., 2021a). The upper bound of our model on TibetanQA can reach 28.1 on BLEU-4.

5.4 Question Type Accuracy of Generated Questions

To further explore the influence of our model on the question type of the generated question, we calculate the accuracy of the interrogative word in the generated questions, as shown in Table 8 and Figure 2, 3, 4.

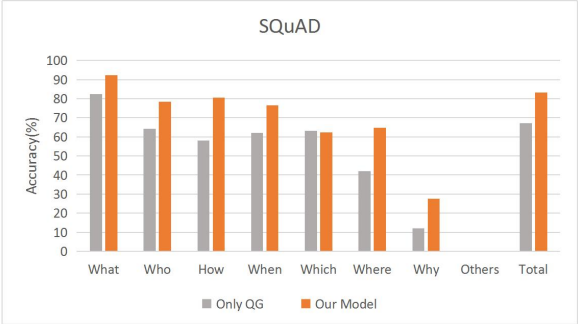

Figure 2: The accuracy on different interrogative words on SQuAD.

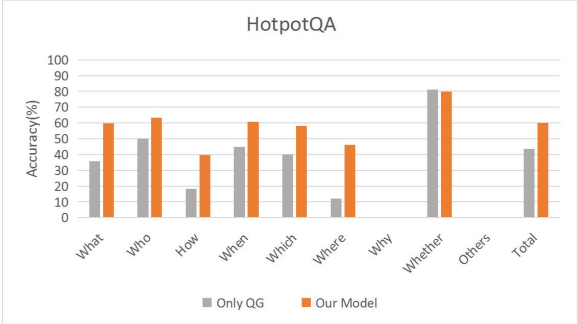

Figure 3: The accuracy on different interrogative words on HotpotQA.

Dataset	Model	What	Who	How	When	Which	Where	Why	whether	Others	Total
SQuAD	Only QG	82.30	64.14	58.03	62.11	63.00	42.08	12.09	-	0	67.15
	Our Model	**92.38**	**78.35**	**80.53**	**76.64**	**62.34**	**64.86**	**27.47**	-	**0**	**83.21**
	Upper Bound	99.55	95.00	99.62	93.77	99.37	98.46	94.51	-	15.00	97.92
HotpotQA	Only QG	35.76	49.90	18.32	44.79	39.95	12.17	0	81.18	0	43.66
	Our Model	**59.70**	**63.15**	**39.69**	**60.85**	**58.2**	**46.03**	**0**	**79.79**	**0**	**59.99**
	Upper Bound	87.80	79.50	71.00	87.04	81.75	83.60	0	98.26	0	89.18
TibetanQA	Only QG	36.11	9.01	0	19.23	1.68		0	-	31.91	26.03
	Our Model	**83.80**	**22.52**	**57.14**	**67.31**	**68.07**		**14.29**	-	**15.51**	**46.88**
	Upper Bound	92.59	17.12	82.43	80.77	92.44		14.29	-	95.51	85.06

Table 8: Accuracy of interrogative words in generated questions on three datasets.

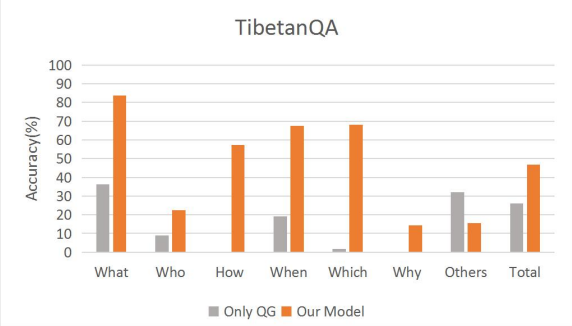

Figure 4: The accuracy on different interrogative words on TibetanQA.

Datasets	Repetition	Incomplete	Accuracy
SQuAD	6.48%	19.18%	87.67%
HotpotQA	5.63%	16.20%	65.50%
TibetanQA	7.38%	39.34%	46.72%

Table 9: Human evaluation results for three datasets.

From Table 8, we can know that the accuracy of interrogative words has been greatly improved. on the SQuAD, the accuracy of interrogative words has increased from 67.15% to 83.21%, and the upper bound of accuracy has reached 97.92%. On the HotpotQA, the accuracy of interrogative words increase from 43.66% to 59.99%, and the upper bound can reach 89.18%. The two categories of "why" and "others" account for 0.03% and 0.1% of the dataset respectively. The small proportion of these two categories makes the model cannot learn the features of these questions. On the TibetanQA, the accuracy of "Only QG" is only 26.03%, the accuracy of our model reaches 46.88%, and the upper bound reaches 85.06%, the accuracy of interrogative words in questions has been greatly improved. We find that the accuracy rate of "who" is low. The main reason is that the grammar of "who" in Tibetan is complex, and the proportion of "who" in TibetanQA is only 5.83%, so it is difficult for the model to learn correct grammar knowledge. In the future work, we will further improve this problem.

Q1:Whether this question is understandable?
A.Yes B.No

Q2:Which of the following errors exists in this question?
A.correct B.Grammatical errors C.repetition
D.incomplete E.ambiguous F.others

Q3:Whether this question is answerable?
A.Yes B.No

Q4:Are the interrogative words and interrogative words in the correct position?
A.Interrogative word and the position of interrogative word are correct
B.Interrogative word is correct but the position of interrogative word is incorrect
C.Interrogative word error
D.Missing Interrogative word

Figure 5: The human evaluation questionnaire.

5.5 Human Evaluation

To further investigate whether the question type classifier improves the quality of generated questions, we conduct human evaluate the final results. We invite six students to analyze the generated questions, three of whom are native speakers of Tibetan and have knowledge of Tibetan linguistics. We randomly select 150 samples from each testing dataset and distribute them equally to these six students. To reduce the subjectivity of human scoring, this paper designs a questionnaire with four questions, as shown in Figure 5. Question 1 evaluates whether the generated question is understandable, if the question is not understandable, we no longer evaluate it for subsequent questions, the question 2 evaluates the fluency of the generated questions, question 3 evaluates whether the generated question is answerable, question 4 evaluates the accuracy of generated interrogative words and the accuracy of the position of interrogative word. The final experimental results of human evaluation are shown in Table 9.

We analyze the obtained questionnaires and discuss the following four findings:

SQuAD	Gold: **how** many permanent objects are located there? Only QG: **what** is the housing of the victoria and albert museum? Our model: **how** many objects does the victoria and albert museum have?
HotpotQA	Gold:**which** tennis player is south african , mariaan de swardt or kateryna bondarenko ? Only QG:**who** was born first , mariaan de swardt or kateryna bondarenko? Our model:**which** tennis player is from south africa , mariaan de swardt or kateryna bondarenko?
TibetanQA	Gold:སྐབས་གཉིས་པའི་འཛམ་གླིང་སྦྲེལ་རེས་དྲ་རྒྱའི་ཚོགས་ཆེན་གྱི་བརྗོད་བྱ་གཙོ་བོ་ནི་**ཅི་ཞིག**་ཡིན། (**What** is the theme of the second World Internet Conference?) Only QG:ཚོགས་ཆེན་གྱི་བརྗོད་བྱ་གཙོ་བོ་**ག་ཚོད**་ཡོད། (**How** many topics are there for the conference?) Our model:ཚོགས་ཆེན་གྱི་བརྗོད་བྱ་གཙོ་བོ་གཙོ་བོ་གཙོ་བོ་**ཅི་ཞིག**་ཡིན། (**What** is the theme of the conference?)

Table 10: Questions generated on three datasets.

1. The proportions of unreadable questions generated on TibetanQA, HotpotQA and SQuAD are 18.67%, 5.33% and 2.67%, respectively.

2. From question2, we find that most of the questions generated on the three datasets are fluent, and some generated questions are more suitable than gold questions, this also shows that automatic evaluation is limited. On the other hand, some generated questions have word repetition and incompleteness problems, incomplete questions are mainly due to the lack of keywords, which will also lead to the generated questions being unanswerable.

3. We performe statistics on generated interrogative words and find that, the accuracy of interrogative words on SQuAD and HotpotQA reached 87.67% and 65.50%, the accuracy on TibetanQA reaches 46.72%, which is close to the accuracy of automatic evaluation.

6 Case Analysis

To explore the impact of our model on question generation, Table 10 lists some typical examples. on SQuAD, the interrogative word of gold question is "how", while the interrogative word generated by the baseline model is "what", and the type of question generated by our model is correct. On the HotpotQA, the questions generated by our model are the same as the gold questions, while the interrogative word generated by "Only QG" is "who". On TibetanQA, the interrogative word generated by the baseline model is “how”, and the interrogative word generated by our model is "what", which is the same as the interrogative word of gold question. The above cases show that the question type classifier proposed in this paper can improve the accuracy of question generation.

7 Conclusion

To solve the problem of mismatch between question types and answers in question generation, this paper constructs a question type classifier and a question generator. We classify questions into fine-grained classification and integrate grammar knowledge into question type classifier to improve the accuracy of question types, then, the prediction results of the classifier are fused into the question generator to improve the performance of question generation. To verify the effectiveness of our model, we perform an upper bound analysis on it, we integrate grammar knowledge into the question generator to provide accurate question types to guide the model to generate questions. The final experimental results show that the method proposed in this paper not only improves the accuracy of question words in the generated question, but also improves the quality of the generated question.

Acknowledgment

This work is supported by "National Nature Science Foundation" (61972436), "the Fundamental Research Funds for the Central Universities" (2022QNYL33).

References

Alexis Conneau, Kartikay Khandelwal, Naman Goyal, Vishrav Chaudhary, Guillaume Wenzek, Francisco Guzmán, Edouard Grave, Myle Ott, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Unsupervised cross-lingual representation learning at scale. *arXiv preprint arXiv:1911.02116*.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*.

X. Du, Junru Shao, and Claire Cardie. 2017. Learning to ask: Neural question generation for reading comprehension. *ArXiv*, abs/1705.00106.

Xinya Du and Claire Cardie. 2018. Harvesting paragraph-level question-answer pairs from Wikipedia. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1907–1917, Melbourne, Australia. Association for Computational Linguistics.

Michael Heilman and Noah A. Smith. 2010. Good question! statistical ranking for question generation. In *Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, pages 609–617, Los Angeles, California. Association for Computational Linguistics.

Xin Jia, Wenjie Zhou, Xu Sun, and Yunfang Wu. 2020. How to ask good questions? try to leverage paraphrases. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 6130–6140, Online. Association for Computational Linguistics.

Mandar Joshi, Eunsol Choi, Daniel Weld, and Luke Zettlemoyer. 2017. TriviaQA: A large scale distantly supervised challenge dataset for reading comprehension. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1601–1611, Vancouver, Canada. Association for Computational Linguistics.

Junmo Kang, Haritz Puerto San Roman, and Sung-Hyon Myaeng. 2019. Let me know what to ask: Interrogative-word-aware question generation. In *Proceedings of the 2nd Workshop on Machine Reading for Question Answering*, pages 163–171, Hong Kong, China. Association for Computational Linguistics.

Yanghoon Kim, Hwanhee Lee, Joongbo Shin, and Kyomin Jung. 2019. Improving neural question generation using answer separation. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, pages 6602–6609.

Igor Labutov, Sumit Basu, and Lucy Vanderwende. 2015. Deep questions without deep understanding. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 889–898, Beijing, China. Association for Computational Linguistics.

Jingjing Li, Yifan Gao, Lidong Bing, Irwin King, and Michael R Lyu. 2019. Improving question generation with to the point context. *arXiv preprint arXiv:1910.06036*.

Chin-Yew Lin. 2004. Rouge: A package for automatic evaluation of summaries. In *Text summarization branches out*, pages 74–81.

Huidan Liu, Minghua Nuo, Longlong Ma, Jian Wu, and Yeping He. 2011. Tibetan word segmentation as syllable tagging using conditional random field. In *Proceedings of the 25th Pacific Asia Conference on Language, Information and Computation*, pages 168–177, Singapore. Institute of Digital Enhancement of Cognitive Processing, Waseda University.

Xiyao Ma, Qile Zhu, Yanlin Zhou, and Xiaolin Li. 2020. Improving question generation with sentence-level semantic matching and answer position inferring. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pages 8464–8471.

Karen Mazidi and Rodney D. Nielsen. 2014. Linguistic considerations in automatic question generation. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 321–326, Baltimore, Maryland. Association for Computational Linguistics.

Nasrin Mostafazadeh, Ishan Misra, Jacob Devlin, Margaret Mitchell, Xiaodong He, and Lucy Vanderwende. 2016. Generating natural questions about an image. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1802–1813, Berlin, Germany. Association for Computational Linguistics.

Lidiya Murakhovs' ka, Chien-Sheng Wu, Tong Niu, Wenhao Liu, and Caiming Xiong. 2021. Mixqg: Neural question generation with mixed answer types. *arXiv preprint arXiv:2110.08175*.

Tri Nguyen, Mir Rosenberg, Xia Song, Jianfeng Gao, Saurabh Tiwary, Rangan Majumder, and Li Deng. 2016. Ms marco: A human generated machine reading comprehension dataset. In *CoCo@ NIPS*.

Minghua Nuo, Huidan Liu, Congjun Long, and Jian Wu. 2015. Tibetan unknown word identification from news corpora for supporting lexicon-based Tibetan word segmentation. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 2: Short Papers)*, pages 451–457, Beijing, China. Association for Computational Linguistics.

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. Bleu: a method for automatic evaluation of machine translation. In *Proceedings of the 40th annual meeting of the Association for Computational Linguistics*, pages 311–318.

Telmo Pires, Eva Schlinger, and Dan Garrette. 2019. How multilingual is multilingual BERT? In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 4996–5001, Florence, Italy. Association for Computational Linguistics.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J. Liu. 2020. Exploring the limits of transfer learning with a unified text-to-text transformer. *Journal of Machine Learning Research*, 21(140):1–67.

Pranav Rajpurkar, Jian Zhang, Konstantin Lopyrev, and Percy Liang. 2016. SQuAD: 100,000+ questions for machine comprehension of text. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 2383–2392, Austin, Texas. Association for Computational Linguistics.

Heung-Yeung Shum, Xiao-dong He, and Di Li. 2018. From eliza to xiaoice: challenges and opportunities with social chatbots. *Frontiers of Information Technology & Electronic Engineering*, 19(1):10–26.

Linfeng Song, Zhiguo Wang, Wael Hamza, Yue Zhang, and Daniel Gildea. 2018a. Leveraging context information for natural question generation. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 2 (Short Papers)*, pages 569–574, New Orleans, Louisiana. Association for Computational Linguistics.

Linfeng Song, Zhiguo Wang, Wael Hamza, Yue Zhang, and Daniel Gildea. 2018b. Leveraging context information for natural question generation. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 2 (Short Papers)*, pages 569–574, New Orleans, Louisiana. Association for Computational Linguistics.

Xingwu Sun, Jing Liu, Yajuan Lyu, Wei He, Yanjun Ma, and Shi Wang. 2018. Answer-focused and position-aware neural question generation. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 3930–3939, Brussels, Belgium. Association for Computational Linguistics.

Yuan Sun, Chaofan Chen, Andong Chen, and Xiaobing Zhao. 2021a. Tibetan question generation based on sequence to sequence model. *CMC-COMPUTERS MATERIALS & CONTINUA*, 68(3):3203–3213.

Yuan Sun, Sisi Liu, Chaofan Chen, Zhengcuo Dan, and Xiaobing Zhao. 2021b. Construction of high-quality Tibetan dataset for machine reading comprehension. In *Proceedings of the 20th Chinese National Conference on Computational Linguistics*, pages 208–218, Huhhot, China. Chinese Information Processing Society of China.

Yuan Sun, Sisi Liu, Junjie Deng, and Xiaobing Zhao. 2022. Tibert: Tibetan pre-trained language model. *arXiv preprint arXiv:2205.07303*.

Luu Anh Tuan, Darsh Shah, and Regina Barzilay. 2020. Capturing greater context for question generation. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pages 9065–9072.

Bingning Wang, Xiaochuan Wang, Ting Tao, Qi Zhang, and Jingfang Xu. 2020a. Neural question generation with answer pivot. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pages 9138–9145.

Liuyin Wang, Zihan Xu, Zibo Lin, Haitao Zheng, and Ying Shen. 2020b. Answer-driven deep question generation based on reinforcement learning. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 5159–5170, Barcelona, Spain (Online). International Committee on Computational Linguistics.

Yuxi Xie, Liangming Pan, Dongzhe Wang, Min-Yen Kan, and Yansong Feng. 2020. Exploring question-specific rewards for generating deep questions. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 2534–2546, Barcelona, Spain (Online). International Committee on Computational Linguistics.

Zhilin Yang, Peng Qi, Saizheng Zhang, Yoshua Bengio, William Cohen, Ruslan Salakhutdinov, and Christopher D. Manning. 2018. HotpotQA: A dataset for diverse, explainable multi-hop question answering. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 2369–2380, Brussels, Belgium. Association for Computational Linguistics.

Yao Zhao, Xiaochuan Ni, Yuanyuan Ding, and Qifa Ke. 2018. Paragraph-level neural question generation with maxout pointer and gated self-attention networks. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 3901–3910, Brussels, Belgium. Association for Computational Linguistics.

Qingyu Zhou, Nan Yang, Furu Wei, Chuanqi Tan, Hangbo Bao, and Ming Zhou. 2017. Neural question generation from text: A preliminary study. In *National CCF Conference on Natural Language Processing and Chinese Computing*, pages 662–671. Springer.

Wenjie Zhou, Minghua Zhang, and Yunfang Wu. 2019a. Multi-task learning with language modeling for question generation. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 3394–3399, Hong Kong, China.

Wenjie Zhou, Minghua Zhang, and Yunfang Wu. 2019b. Question-type driven question generation. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 6032–6037, Hong Kong, China.

CM-Gen: A Neural Framework for Chinese Metaphor Generation with Explicit Context Modelling

Yucheng Li [1], Chenghua Lin [2]*, Frank Guerin [1]
[1] Department of Computer Science, University of Surrey, UK
{yucheng.li,f.guerin}@surrey.ac.uk
[2] Department of Computer Science, University of Sheffield, UK
c.lin@sheffild.ac.uk

Abstract

Nominal metaphors are frequently used in human language and have been shown to be effective in persuading, expressing emotion, and stimulating interest. This paper tackles the problem of Chinese Nominal Metaphor (NM) generation. We introduce a novel neural framework, which jointly optimizes three tasks: NM identification, NM component identification, and NM generation. The metaphor identification module is able to perform a self-training procedure, which discovers novel metaphors from a large-scale unlabeled corpus for NM generation. The NM component identification module emphasizes components during training and conditions the generation on these NM components for more coherent results. To train the NM identification and component identification modules, we construct an annotated corpus consisting of 6.3k sentences that contain diverse metaphorical patterns. Automatic metrics show that our method can produce diverse metaphors with good readability, where 92% of them are novel metaphorical comparisons. Human evaluation shows our model significantly outperforms baselines on consistency and creativity.

1 Introduction

Metaphors are commonly used in human language. Usually, metaphors compare two different kinds of objects or concepts with the intent to make the expression more vivid, or to make unfamiliar things easier to understand (Paul, 1970). According to contrastive studies of English and Chinese, metaphors are especially crucial in Chinese as there are fewer abstract words in Chinese, so that people tend to express abstract meaning via metaphors (Lian, 1994).

In this paper, we focus on the generation task of a special type of Chinese metaphor – Nominal Metaphors (NMs). NMs (比喻 in Chinese) are figures of speech associating a noun with another noun through a COMPARATOR such as *like, be, become* in English and 像,是,变成 in Chinese. Examples and NM components are shown in Table 1. In addition to the COMPARATOR (**bold**), there are three other components in a nominal metaphor: TENOR, VEHICLE, and CONTEXT (text with underline). TENOR is the subject of the metaphor, and VEHICLE is the source of the imagery (i.e., the object of metaphor). CONTEXT is used to explain the comparison and is crucial for understanding the comparison (more details about NM and NM components are given in § 2.1). The NM generation task is defined as follows: given a TENOR, generate a metaphor containing the three remaining NM components, i.e., VEHICLE, COMPARATOR and CONTEXT. Previous efforts on NM processing mainly engage in identification (Liu et al., 2018; Zeng et al., 2020) and interpretation (Su et al., 2016, 2017), where the generation of NMs is relatively unexplored, despite the benefits it can bring to many downstream tasks (Wang et al., 2021). Glucksberg (1989); Zhou (2020) suggest that metaphors are important to an engaging con-

1. 这个[孩子]$_{\text{tenor}}$ 壮的像[牛]$_{\text{vehicle}}$ This [boy]$_{\text{tenor}}$ is as strong **as** a [bull]$_{\text{vehicle}}$.	*Nominal*
2. [生活]$_{\text{tenor}}$好比[旅行]$_{\text{vehicle}}$， 没有计划就难以前行 [Life]$_{\text{tenor}}$ **is** a [journey]$_{\text{vehicle}}$, we cannot move on without a plan.	*Nominal*
3. Meta股价[跳水]$_{\text{metaphorical}}$ META stock price [dives]$_{\text{metaphorical}}$.	*Verbal*
4. 他可以像大厨一样烹饪 He can cook like a pro.	*Literal*

Table 1: Examples of Chinese nominal metaphor, verbal metaphor, and NM components. Note that when the words "like" or "as" are used as COMPARATORS, we also call these special NMs 明喻 (Similes).

*Corresponding author

Proceedings of the 29th International Conference on Computational Linguistics, pages 6468–6479
October 12–17, 2022.

versation and can effectively stimulate user interest in communicating with chatbots. Chakrabarty et al. (2020, 2021) show that users prefer stories and poems enhanced with metaphor generation by replacing literal expressions with generated metaphors.

To tackle Chinese NM generation, there are two main challenges to address. First, existing Chinese NM corpora are not large enough to facilitate current data-driven text generation approaches. Second, the auto-regressive nature of generative models always assigns higher priority to fluency, which makes the metaphor generation procedure produce *inconsistency errors*[1] (e.g., generating nonsense comparisons without CONTEXT explaining) and *literal errors* (i.e., generating literal expressions).

We propose a novel multitask approach for Chinese NM generation called CM-Gen to address the abovementioned challenges. Specifically, three tasks are jointly optimized: NM generation, NM identification, and NM components identification. **First**, for the data scarcity problem, we perform a self-training procedure to learn newly discovered metaphors from large-scale unlabeled datasets. Self-training has three main steps: 1) our model is trained on a labeled dataset for NM identification; 2) we apply our model on an unlabeled corpus to detect potential NMs with a corresponding confidence score; and 3) train an NM generation model on the combination of labeled and newly found NMs. By exploiting rich metaphors from large-scale resources, the performance of CM-Gen can be significantly improved yet the data requirement can be dramatically reduced. **Second**, CM-Gen proposes to identify potential metaphor components (i.e., TENOR, COMPARATOR and VEHICLE) supervised by the attention weights generated by the NM classifier. To alleviate *inconsistency errors*, CM-Gen conditions the generation process on the potential NM components; this enforces the CONTEXT generation to depend on the comparison, rather than producing fluent but bland CONTEXT that does not explain the comparison. In terms of the *literal errors*, NMs components are emphasised via attention weight to encourage CM-Gen to produce metaphorical expressions rather than literals.

We also build an annotated corpus for Chinese NM identification consisting 6.3k sentences. Instead of focusing on a specific metaphorical pattern (Liu et al., 2018), our corpus contains diverse nominal metaphorical usages. We also ensure the CONTEXT is explicit for each metaphor annotated, and the TENOR of each metaphor is also identified. Source code and data can be found in https://github.com/liyucheng09/Metaphor_Generator.

[1]An example of *inconsistency error*: "Teacher is like a candle, floating gently in the air". Although the comparison is valid, the CONTEXT is inconsistent with the comparison. This also shows the importance of CONTEXT in NM generation.

2 Related Work

2.1 Metaphors in Chinese

Following (Krishnakumaran and Zhu, 2007; Rai and Chakraverty, 2020), we can divide English metaphors into four types as follows:

Type-I: (Nominal Metaphors) A noun is associated with another noun through the comparators, e.g., "Love is a journey".

Type-II: (Verbal Metaphors or Subject-Verb-Object (SVO) metaphors) Sentences with metaphorical verb, e.g., "He kills a process".

Type-III: (Adjective-Noun (AN) metaphor) Metaphorical adjectives with a noun fall into this category, e.g., "sweet boy".

Type-IV: (Adverb-Verb (AV) metaphor) Metaphorical adverbs with a verb, e.g., "speak fluidly".

However, the definition of metaphor in the context of Chinese is slightly different from its English counterpart (Wang, 2004). 比喻 (Metaphor), or 打比方 (draw an analogy), which draws a comparison between objects or concepts, mainly means *Type-I* metaphor, i.e., NMs. A specific term 比拟 (Personification/Match) is used to indicate *Type-II, Type-III, Type-IV* metaphors in Chinese, which aims to describe an object or concept in a view of a person or another object. Verbal Metaphors (VMs) are the most frequent type of metaphor in English (Martin, 2006; Steen, 2010), but NMs are the dominant figurative language in Chinese. According to a small-scale annotation analysis (Su et al., 2016), NMs are around four times more frequent than VMs in Chinese. Lian (1994) gives a possible explanation for this phenomenon: Chinese people tend to express abstract concepts via nominal metaphors or idioms as there are fewer abstract terms in Chinese than in English. For example, a Chinese nominal metaphor "像竹篮打水" (doing something is like ladling water to a leaky basket), is used to express the meaning of "hopeless".

Chinese NMs often consist of four components: TENOR, VEHICLE, COMPARATOR, (本体,喻体,比喻词 in Chinese) and CONTEXT, as shown in ta-

ble 1. The CONTEXT here is a component used to *explain* the comparison; its definition is relatively flexible. Sometimes it can be a simple adjective, sometimes a relative clause, or even implicit in some cases. For example, the NM "*The city is like a painting*" omits the textual CONTEXT to emphasize visual senses. However, CONTEXT is extremely important in helping readers to understand the comparison. According to Indurkhya (2007) and Lakoff and Johnson (2008), a comparison can be drawn between any concepts, but it must have a CONTEXT to explain the comparison or to make the comparison coherent to daily experience. Considering the importance of CONTEXT, we **do not** consider a comparison without CONTEXT as a successfully generated NM case in our experiments.

Additionally, there are two linguistic principles Chinese NMs must obey (Wang, 2004): 1) The comparison must be drawn between two concepts with different natures; and 2) the two concepts being compared should share commonalities. Specifically, the COMPARATOR "like" in example No.4 of Table 1 does not necessarily make it an NM, since the comparison is drawn between the same concept "me cooking" and "pro cooking". The second principle also emphasises the importance of CONTEXT. In summary, even though NMs usually share a relatively simple structure, Chinese NM generation is still challenging due to the requirement of providing CONTEXT and the necessity of understanding the relation between TENOR and VEHICLE.

2.2 Computational Processing of NMs

Previous works on computational processing of NMs can be classified into detection, interpretation and generation.

Detection and Interpretation Krishnakumaran and Zhu (2007) exploit the absence of a hyponymy relation between subject and object to identify metaphorical utterances. Shlomo and Last (2015) propose a random forest-based classifier for NM identification using both conceptual features such as abstractness and semantic relatedness such as domain corpus frequency. Su et al. (2016) follow the idea of lack of hyponymy relationship from (Krishnakumaran and Zhu, 2007) and realize it using cosine distance between pre-trained word2vec embeddings of the source and target concepts. Liu et al. (2018); Zeng et al. (2020) tackle Chinese simile detection by designing a multi-task framework and a local attention mechanism. Su et al. (2016, 2017) focus on NM interpretation and perform experiments on both English and Chinese NMs. They extract properties of TENOR and VEHICLE from WordNet and use pre-trained word2vec embeddings to identify related properties shared by both components.

Generation Despite the benefits NM generation can bring to the community, prior works on this task are relative sparse. Early works often rely on templates. Terai and Nakagawa (2010) compute the relatedness between concepts with computational language analysis and select candidates to fill metaphor templates like "A is like B". Veale (2016) uses a knowledge-base to generate XYZ style NMs such as "Bruce Wayne is the Donald Trump of Gotham City". Zhou (2020) not only choose candidate concept pairs by word embedding similarity to fill the template but also choose appropriate COMPARATORS to link the concept pair. (Chakrabarty et al., 2020) introduce a neural style transfer approach for simile generation, which finetunes a pre-trained sequence-to-sequence model on a literal-simile parallel dataset. Nevertheless, previous template-based approaches heavily constrain the diversity of generated NMs and both template methods and neural methods produce NMs in a reletive simple structure. Most importantly, previous methods do not provide CONTEXT in their generations (or only provide little CONTEXT), which makes generated results less readable.

3 Method

Given an object or concept as a starting TENOR, a Chinese nominal metaphor will be generated consisting of four NM components: a comparison between TENOR and VEHICLE linked with a COMPARATOR and a CONTEXT as an explanation for the comparison. The overall multitask framework is shown in Figure 1. We can roughly divide our framework into four elements: 1) the GPT2 (Radford et al., 2019) pre-trained language model; and three task-specific fully-connected layers used for 2) NM identification; 3) NM components identification; and 4) NM generation.

3.1 Shared Representation

Since we are tackling a generation task, we employ a pre-trained unidirectional transformer-based language model, GPT2, as our basic encoder. Contextualized word representations are obtained after feeding words to the GPT2 model. For-

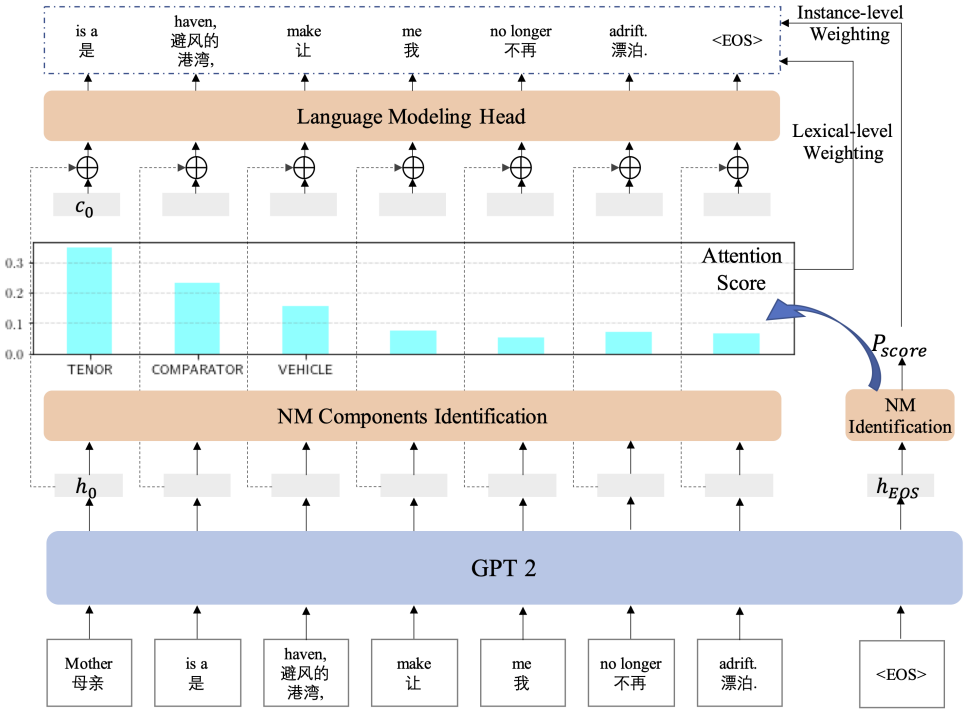

Figure 1: The overall framework.

mally, given sentence $S = (w_0, ..., w_n, w_{EOS})$, the GPT2 model produces a list of vectors $H = (h_0, ..., h_n, h_{EOS})$, where the `EOS` is a special delimiter indicating the end of the sentence. Note that the representation here are used in the three individual tasks described below and the parameters are also shared across all tasks.

3.2 Task 1: NM Identification

The NM identification module is used to assign metaphorical probability to sentences. This score will be used in the Self-Training procedure (described in §3.4). Specifically, we use h_{EOS} as the sentence representation of S (similar to the usage of `cls` embedding in BERT-based systems (Devlin et al., 2018)) and apply a linear layer plus a softmax layer on it to compute the metaphorical probability of the sentence S. Formally, the metaphor probability is computed as follow:

$$P_M = \text{softmax}(W_m h_{EOS} + b_m) \quad (1)$$

where W_m and b_m are trainable weights and biases for NMs identification.

We train this module on a supervised dataset noted as $U = \{(x_i, y_i)\}_{i=1}^{N}$, where x indicates the sentence and y indicates whether x is a NM. In summary, we minimize the following loss function for NM identification:

$$L_1 = -\sum_{x \in U} \log P(\hat{y}|x) \quad (2)$$

3.3 Task 2: NMs Components Identification

Although GPT2 model is powerful in generating fluent and grammatical text, it still suffers from incoherence issues (Ko and Li, 2020; Tan et al., 2021). In the scenario of NM generation, it means the CONTEXT generation might be inconsistent with the metaphorical comparison thus resulting in *inconsistency errors*. Besides, the innate tendency of generative models to produce literal text often leads to *literal errors* (Chakrabarty et al., 2021).

To address the *inconsistency errors*, our model conditions the generation procedure on the metaphorical comparison, that is the NM components TENOR, VEHICLE, and COMPARATOR. We also weight these NM components with higher score during training to alleviate *literal errors*. These two approaches (described in §3.4) require the involvement of NM components, therefore, we apply a linear layer to compute the probability for each token to be an NM component. Formally, this probability is computed as follows:

$$P_c = \text{Sigmoid}(W_c H + b_c) \quad (3)$$

where W_c and b_c are trainable weights and bias for NM component identification, and P_c is the

resulting probability distribution. Note that this process does not predict the type of components (e.g., TENOR), instead, it only computes a probability for each token indicating the extent to which the generation should focus on each.

We propose to use the attention weights generated from the NM classifier (obtained in §3.2) as the supervision signals for NM component identification. As shown in (Liu et al., 2018; Zeng et al., 2020), the metaphor classifier tends to focus more on corresponding metaphor components, we thus use this property to discover NM components. Specifically, we use KL divergence to have our distribution P_c as close as possible to the attention weights Φ.

$$L_2 = D_{KL}(P_c \| \Phi) \tag{4}$$

where Φ is the self-attention score the h_{EOS} attending to other tokens generated by the last layer Transformer of GPT2.

$$\Phi = \text{softmax}(\frac{Qk^T}{\sqrt{d_k}}) \tag{5}$$

The Q here is the Query matrix for self-attention, and k is the Value vector only for the EOS token.

3.4 Task 3: NM Generation

We perform the NM generation task with three steps: 1) conditioning the generation on NM components; 2) emphasizing the NM components; and 3) executing the self-training procedure.

Conditioning To allow token predictions conditioned on NM components, we provide a list of NM component representations $C = (c_0, .., c_i, .., c_n)$ for each prediction step respectively. Then the NM component representation c_i is fed into the language modeling head together with the contextualized token embedding h_i. Formally, c_i is computed as follows:

$$c_i = \sum_{k=0}^{i} \alpha_k \cdot h_k \tag{6}$$

where the weight score α is computed as follows:

$$(\alpha_0, ..., \alpha_i) = \text{softmax } P_c^{\{0,...,i\}} \tag{7}$$

c_i here mainly captures NM component information before the i-th token (i.e., NM components within $(w_0, ..., w_i)$). Then we concatenate the contextualized token embedding h_i and its corresponding NM component information embedding c_i to predict the next token.

$$P(w_{i+1}|w_0, ..., w_i) = \text{softmax}\left[W_l(h_i \oplus c_i) + b_l\right] \tag{8}$$

where the W_l and b_l are trainable weight matrices and bias, $\oplus$ indicating the concatenation operation.

Emphasizing We emphasize the NM components during training by directly applying attention weight P_c on the loss function. Specifically, given a sentence $S = (w_0, ..., w_n)$, we minimize the following loss function:

$$\mathcal{L}(S) = -\sum_{i=0}^{n} P_c^i \cdot \log P(w_i|w_0, ..., w_{i-1}) \tag{9}$$

where P_c^i is the probability to be one of the NM components of token w_i.

Self-training Self-training is an effective approach to tackle data scarcity and has been successfully used in many downstream tasks (He et al., 2019; Parthasarathi and Strom, 2019; Xie et al., 2020). In our setting, we adopt self-training for discovering novel Chinese NMs from large-scale corpora to train the NM generation module so that the fluency and diversity of generation can be improved.

Formally, given an unlabeled corpus $V = \{x_i\}_{i=0}^N$ where each x is a sentence $x = (w_0, ..., w_n)$, the NM identifier will assign a probability to each x_i noted as P_M^i. We than mix the unlabeled corpus $V = \{(x_i, P_M^i)\}_{i=0}^N$ and supervised dataset $U = \{(x_i, y_i)\}_{i=1}^N$ together, and train the overall framework on it. Formally, we minimize the following loss function:

$$L_3 = -\left[\sum_{x \in V} P_M^i \cdot \mathcal{L}(x) + \sum_{S \in U} \mathcal{L}(S)\right] \tag{10}$$

3.5 Training and Inference

The final loss function of our framework is a weighted sum of three task-specific loss function.

$$L = \gamma \cdot L_1 + L_2 + L_3 \tag{11}$$

Note that when learning unlabeled sentences, γ is set to 0, since these instances lack the supervision label for NM identification. To help our model converge, before training the overall framework on the mixed data by L, we pre-train our model on the supervised dataset for Task 1 first. Besides, when doing inference, our model only performs Task 3.

4 Experiment

4.1 dataset

To train our multitask framework, we construct two datasets: a supervised Chinese NM Corpus (CMC) and a large-scale unsupervised Chinese Literature Corpus (CLC).

CMC Existing Chinese metaphor corpus are neither too small, like Su et al. (2016) contains 120 examples, or focusing on a specific metaphorical pattern, like Liu et al. (2018) contains sentences with a specific COMPARATOR 像 (like). In our corpus, we try to include nominal metaphors as diverse as possible. The annotation of the CMC consists of five steps: 1) we collect 55,000 Chinese sentences from essays, articles, and novels; 2) we employ three Chinese graduate students with background of NLP to label each sentence as a NM or not; 3) we take the majority agreement as the final label for each sentence; 4) the boundary of all NM components are identified including TENOR, COMPARATOR and VEHICLE; 5) we merge existing metaphor corpora (Liu et al., 2018; Li et al., 2022) with ours to enlarge the overall diversity. To encourage the CONTEXT to be generated, we ensure CONTEXT occurs explicitly in all metaphors from our dataset. Before the annotation, annotators are trained with examples and instructed with basic Chinese NM principles (described in §2.1). Due to the merging with other Chinese metaphor datasets, some metaphor components might missing in the overall CMC. We compute the inner-annotator agreement of NM labeling via Krippendorff's alpha (Krippendorff, 2011). The agreement rate is 0.84. Statistics of CMC are shown in Table 2. Some examples are shown in Appendix A.1.

CLC In self-training, we need a large-scale corpus so that the NM identifier can discover novel NMs. However, popular Chinese corpora, such as news, Wikipedia, web pages, are not suited to be used as metaphor resources. Intuitively, literature text might be a promising resource of diverse metaphors. Therefore, we construct a Chinese literature corpus by collecting a large number of essays, novels, and fictions (see details in Appendix A.2). Statistics of CLC are shown in Table 2.

4.2 Baselines

Chinese NMs generation is a novel task, we select three general generative models, an English simile generation method, and a Chinese metaphor generation model as baselines.

	CMC	CLC
# Sentences	11581	6.98M
# NM	8027	-
# literal sentence	3554	-
# tokens	559K	202M
# tokens per sentence	48	29

Table 2: Statistics of CMC and CLC datasets

SeqGAN: Sequence Generative adversarial network (Yu et al., 2017) with a generator implemented by LSTM network and a discriminator implemented by CNN network. We train this model on CMC to produce Chinse NM.

GPT2: The Chinese GPT2 model is fine-tuned on the CMC dataset to produce Chinese NMs as a baseline model.

BART: We fine-tune a Chinese version BART model (Shao et al., 2021) model on parallel data pairs <TENOR, Sentence> obtained from CMC.

SCOPE: The SOTA method (Chakrabarty et al., 2020) on English simile generation tasks, which fine-tunes BART model on a large-scale automatically created literal-simile parallel corpus.

MultiTask: Li et al. (2022) introduce the first multitasking metaphors generator for Chinese metaphors. It relies on an attention weights-based components identification procedure to ensure the metaphoricity of outputs.

4.3 Experiments Setting

We use a pre-trained Chinese GPT2 model[2] to avoid starting training from scratch. Our model is pre-trained on NM identification task with CMC for 3 epochs before jointly optimizing three task-specific loss functions. The implementation of SeqGAN[3] and the pre-trained Chinese BART model[4] can be found in the footnote. Before the SeqGAN starts training on CMC, we first pre-train the generator of SeqGAN on CLC for 50k steps. Hyperparameters not specified are all followed by default settings. Note that the SCOPE model is designed for English Simile generation and it takes a literal utterance as input. To compare SCOPE results with our method, we first translate input TENORS into English (via Google Translator), then translate generated NMs back to Chinese (details in Appendix

[2] https://huggingface.co/uer/gpt2-chinese-cluecorpussmall

[3] https://github.com/LantaoYu/SeqGAN

[4] https://huggingface.co/fnlp/bart-base-chinese

Methods	PPL	Dist-1	Dist-2	Meta	Novelty	Fluency	Consistency	Creativity
SeqGAN	89.43	.00336	.0116	**.998**	.200	3.33 (.51)	3.80 (.46)	1.67 (.34)
GPT2	57.88	.00916	.1154	.981	.800	4.00 (.62)	3.10 (.39)	2.60 (.31)
BART	48.58	.00826	.0971	.978	.725	4.35 (.54)	3.05 (.37)	2.30 (.32)
SCOPE	92.32	.00517	.0673	.910	.385	3.10 (.64)	2.70 (.44)	2.10 (.45)
MultiTask	26.79	.01143	.1582	.952	.815	4.55 (.58)	4.23 (.45)	3.80 (.36)
Our Method	**25.79**	**.01153**	**.1674**	.948	**.920**	**4.65** (.58)	**4.40** (.45)	**3.80** (.36)
w/o Self-training	62.54	.00674	.0906	.982	.785	3.85 (.54)	3.87 (.42)	2.76 (.38)
w/o Emphasizing	25.58	.01150	.1529	.803	.900	4.50 (.63)	3.91 (.32)	3.41 (.43)
w/o Conditioning	24.93	.01053	.1534	.875	.930	4.25 (.61)	3.05 (.45)	3.24 (.39)

Table 3: Results of automatic metrics and human evaluation. Boldface denotes the best results among our method and baselines. The inter-annotator agreement for human evaluation are shown in parenthesis.

B). In the test stage, we randomly select and feed 200 TENORS from CMC to all generative models. During decoding, all beam sizes are set to 12, thus each model generated 12 sentence for each TENOR. In total, 2400 sentences are obtained per model for testing.

4.4 Metrics

Automatic Metrics We use perplexity (**PPL**) to evaluate the fluency of the generated text, which is calculated by an open source Chinese language model (Zhang et al., 2020). **Dist-1** and **Dist-2** (Li et al., 2016) compute the distinct unigrams and bigrams ratio of generated text which are used to measure model's ability to produce diversity outputs. To test the **metaphoricity (Meta)** of generated outputs, we train a RoBERTa-based Chinese NM classifier on CMC to compute the ratio of metaphorical utterances in the generated sentences. The accuracy of this classifier is 97.89%, which is robust enough to perform evaluation (details in Appendix C). **Novelty** is to test how well models can generate metaphors they have never seen during training. We use a syntax-based approach to identify TENORS and VEHICLES from generated NMs and compute the proportion of <TENOR, VEHICLE> pair that does not co-occurr in the training set.

Human Evaluation Due to the creative and delicate usage of NM, automatic metrics are not adequate to test the quality of generated outputs. We also perform human evaluation based on the following three criteria: 1) **Fluency** indicates how well the metaphor is formed; whether the expression is grammatical and fluent. 2) **Consistency** indicates whether the metaphor can explain itself; how well the VEHICLE relate to TENOR and how well the CONTEXT explain the comparison. 3) **Creativity** scores how creative annotators think the metaphor is. Note that the Creativity judgment is based on annotators' real-life experience, rather than measuring whether the generated metaphor appears in the training dataset. Three annotators were instructed to rate the three criteria from 1 to 5 where 1 denotes worst and 5 be the best.

5 Results

5.1 Automatic Evaluation

Results of automatic metrics are shown in Table 3. Our method significantly outperforms baselines in most automatic metrics. Our model obtains a lower PPL, which illustrates our model is better at producing fluency and grammatical text. Higher Dist-1 and Dist-2 scores show our method produces less repetitive unigrams and bigrams during generation, which is essential in creative language generation. The Meta (metaphor) score shows that our model produces more literal expressions than baselines, which might result from the self-training procedure, where non-metaphorical sentences are sometimes wrongly identified by the NM identification module, and thus there is noise in NM modeling. The highest Novelty score demonstrates our method's ability to generate creative comparisons.

We implemented an ablation study to test the effectiveness of self-training, NM component emphasizing, and context conditioning. Experimental results prove the self-training mechanism improves both generation fluency and diversity. Removing self-training from our model affects four automatic metrics by a large margin. The NM component emphasizing mainly helps our method alleviate *literal errors* and thus improve the Meta score. The context conditioning also benefits the overall framework in Meta score.

Methods	**Text** (Chinese)	**Text** (Translated)	**Con.**	**Cre.**
GPT2	秋天是美丽的，让人赏心悦目。	Autumn is beautiful, and is delightful to the eye.	-	-
	秋天是个动情的音符，荡漾在夏日的清纯中。	Autumn is an emotional note, rippling in the purity of summer.	2.0	3.5
	秋天是最好的伴奏曲，让世界充满微笑。	Autumn is the best concertos, making the world full of smiles.	3.3	2.0
MultiTask	秋天像那沉甸甸的银杏，在树梢间点缀着一道彩虹。	Autumn is like the heavy ginkgo, shines as a rainbow behind the treetop.	4.3	3.0
	秋天像千万条银丝在天空中飘荡。	Autumn is like thousands of silver strings hanging in the sky.	1.3	4
	秋天是个好天气，天空一片明朗，不过到春天却只剩下一圈蓝色。	Autumn is a great weather, and the sky is blue and bright. But when it come to Spring there is only blue.	-	-
Our method	秋天像一只彩笔画般的画笔，勾勒出一幅幅多彩多姿的画卷。	Autumn is like a multi-colored paintbrush, sketching out colorful pictures.	5.0	2.7
	秋天像小姑娘的脚，带着她那柔软的臂膀，在枝头翩翩起舞。	Autumn is like a little girl's feet with her softness. Arms, dancing in the branches.	3.7	5.0
	秋天像刚刚落地的苹果,在果园里露出个头。	Autumn is like an apple that has just fallen, showing its head in the orchard.	4.3	4.0
	秋天像刚落的蝉，婉转地鸣叫着，见证着树梢上金黄色的叶子慢慢向蓝天生长。	Autumn is like a cicada that has just fallen, chirping tactfully, seeing the golden leaves on the treetops grow towards the blue sky slowly.	4.3	5.0

Table 4: NMs generated by our method and baselines given a TENOR 秋天 (Autumn). **Con.** and **Cre.** indicate the two human evaluation metrics **Consistency** and **Creativity** respectively. We do not assign Con. and Cre. score for non-metaphorical utterances. More examples of CM-Gen are shown in Appendix D.

5.2 Human Evaluation

We select 180 sentences in total for human evaluation, i.e., 15 TENORS with 12 sentences for each TENOR. Human evaluation results are shown in Table 3. We also show in the table the inter-annotator agreement of human annotation via Krippendorff's alpha. We can see that our method beats five baseline models on all three human-centric metrics. The most significant improvement lies in Consistency and Creativity, which show our method can not only generate creative comparisons, but more importantly, can also provide a valid CONTEXT for each NM to explain the comparison, which is essential for readability. Human evaluation also demonstrates the effectiveness of the three mechanism self-training, emphasizing, and conditioning. Self-training enhances generation quality in both fluency and creativity dimensions. Conditioning mostly contributes to the consistency score as it empowers the model to generate context description.

5.3 Case Study

We show in Table 4 some NMs examples generated by GPT2, MultiTask, and our model, where the models take 秋天 (Autumn) as the input TENOR. The corresponding Consistency and Creativity scores are also given in the table. We see that although all three models are able to produce metaphorical outputs, the quality of the generated results differs among systems. First, the comparisons given by our model are more diverse than the baselines. For example, GPT2 tends to compare autumn with "music" (i.e., note and accompaniment). Second, the CONTEXT produced by our method can explain the comparison well, which ensures the consistency and readability of the outputs. However, baselines either give little CONTEXT (e.g., MultiTask gives no explanation in the second example) or inappropriate CONTEXT (e.g., GPT2 uses summer in the comparison with autumn). Third, we observe that our model generates NMs with a

relatively more complicated structure and speaks in a more poetic way. For example, our model does not use a single word as VEHICLE, instead, it generates detailed phrases such as "*apple that has just fallen*", "*dancing on the branches*". These detailed components paint a more vivid picture, and thus improve the overall readability of the generated NMs; such an observation is also supported by the corresponding human-rated Consistency and Creativity scores.

6 Conclusion

In this paper, we propose a novel multitask framework for Chinese nominal metaphor generation. Our model seeks to jointly optimize three tasks including NM identification, NM component identification, and NM generation, which is effective in generating diverse NMs with good coherence and readability. Extensive experiments show that our model can significantly outperform baselines in terms of both consistency and creativity. Additionally, we publish an annotated corpus for Chinese nominal metaphors. Future directions can be trying the usage of syntactic features and controllable NM generation. Moreover, we would also like to evaluate the effect of metaphor generation in downstream tasks, such as story generation, dialog systems (Zeng et al., 2021), and educational applications (Lin et al., 2015).

References

Antoine Bosselut, Hannah Rashkin, Maarten Sap, Chaitanya Malaviya, Asli Celikyilmaz, and Yejin Choi. 2019. Comet: Commonsense transformers for automatic knowledge graph construction. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 4762–4779.

Tuhin Chakrabarty, Smaranda Muresan, and Nanyun Peng. 2020. Generating similes effortlessly like a pro: A style transfer approach for simile generation. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 6455–6469.

Tuhin Chakrabarty, Xurui Zhang, Smaranda Muresan, and Nanyun Peng. 2021. Mermaid: Metaphor generation with symbolism and discriminative decoding. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 4250–4261.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*.

Sam Glucksberg. 1989. Metaphors in conversation: How are they understood? why are they used? *Metaphor and Symbol*, 4(3):125–143.

Junxian He, Jiatao Gu, Jiajun Shen, and Marc'Aurelio Ranzato. 2019. Revisiting self-training for neural sequence generation. In *International Conference on Learning Representations*.

Bipin Indurkhya. 2007. Creativity in interpreting poetic metaphors. In T. Kusumi, editor, *New directions in metaphor research*, pages 483–501. Tokyo, Japan: Hitsuji Shobo.

Wei-Jen Ko and Junyi Jessy Li. 2020. Assessing discourse relations in language generation from gpt-2. In *Proceedings of the 13th International Conference on Natural Language Generation*, pages 52–59.

Klaus Krippendorff. 2011. Computing krippendorff's alpha-reliability.

Saisuresh Krishnakumaran and Xiaojin Zhu. 2007. Hunting elusive metaphors using lexical resources. In *Proceedings of the Workshop on Computational approaches to Figurative Language*, pages 13–20.

George Lakoff and Mark Johnson. 2008. *Metaphors we live by*. University of Chicago press.

Jiwei Li, Michel Galley, Chris Brockett, Jianfeng Gao, and William B Dolan. 2016. A diversity-promoting objective function for neural conversation models. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 110–119.

Yucheng Li, Chenghua Lin, and Frank Geurin. 2022. Nominal metaphor generation with multitask learning. *arXiv preprint arXiv:2206.05195*.

Shuneng Lian. 1994. *Contrastive Studies OF English and Chinese*. Fudan University Press.

Chenghua Lin, Dong Liu, Wei Pang, and Zhe Wang. 2015. Sherlock: A semi-automatic framework for quiz generation using a hybrid semantic similarity measure. *Cognitive computation*, 7(6):667–679.

Lizhen Liu, Xiao Hu, Wei Song, Ruiji Fu, Ting Liu, and Guoping Hu. 2018. Neural multitask learning for simile recognition. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 1543–1553.

James H Martin. 2006. A corpus-based analysis of context effects on metaphor comprehension.

Sree Hari Krishnan Parthasarathi and Nikko Strom. 2019. Lessons from building acoustic models with a million hours of speech. In *ICASSP 2019-2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 6670–6674. IEEE.

Anthony M Paul. 1970. Figurative language. *Philosophy & Rhetoric*, pages 225–248.

Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, Ilya Sutskever, et al. 2019. Language models are unsupervised multitask learners. *OpenAI blog*, 1(8):9.

Sunny Rai and Shampa Chakraverty. 2020. A survey on computational metaphor processing. *ACM Computing Surveys (CSUR)*, 53(2):1–37.

Yunfan Shao, Zhichao Geng, Yitao Liu, Junqi Dai, Fei Yang, Li Zhe, Hujun Bao, and Xipeng Qiu. 2021. Cpt: A pre-trained unbalanced transformer for both chinese language understanding and generation. *arXiv preprint arXiv:2109.05729*.

Yosef Ben Shlomo and Mark Last. 2015. Mil: automatic metaphor identification by statistical learning. In *Proceedings of the 2nd International Conference on Interactions between Data Mining and Natural Language Processing-Volume 1410*, pages 19–29.

Gerard Steen. 2010. *A method for linguistic metaphor identification: From MIP to MIPVU*, volume 14. John Benjamins Publishing.

Chang Su, Shuman Huang, and Yijiang Chen. 2017. Automatic detection and interpretation of nominal metaphor based on the theory of meaning. *Neurocomputing*, 219:300–311.

Chang Su, Jia Tian, and Yijiang Chen. 2016. Latent semantic similarity based interpretation of chinese metaphors. *Engineering Applications of Artificial Intelligence*, 48:188–203.

Bowen Tan, Zichao Yang, Maruan Al-Shedivat, Eric Xing, and Zhiting Hu. 2021. Progressive generation of long text with pretrained language models. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 4313–4324.

Asuka Terai and Masanori Nakagawa. 2010. A computational system of metaphor generation with evaluation mechanism. In *International Conference on Artificial Neural Networks*, pages 142–147. Springer.

Tony Veale. 2016. Round up the usual suspects: Knowledge-based metaphor generation. In *Proceedings of the Fourth Workshop on Metaphor in NLP*, pages 34–41.

Dingmin Wang, Chenghua Lin, Qi Liu, and Kam-Fai Wong. 2021. Fast and scalable dialogue state tracking with explicit modular decomposition. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 289–295.

Xijie Wang. 2004. *HanYu XiuCi Xue*. The Commercial Press.

Qizhe Xie, Minh-Thang Luong, Eduard Hovy, and Quoc V Le. 2020. Self-training with noisy student improves imagenet classification. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 10687–10698.

Lantao Yu, Weinan Zhang, Jun Wang, and Yong Yu. 2017. Seqgan: Sequence generative adversarial nets with policy gradient. In *Thirty-First AAAI Conference on Artificial Intelligence*.

Chengkun Zeng, Guanyi Chen, Chenghua Lin, Ruizhe Li, and Zhi Chen. 2021. Affective decoding for empathetic response generation. In *Proceedings of the 14th International Conference on Natural Language Generation*, pages 331–340.

Jiali Zeng, Linfeng Song, Jinsong Su, Jun Xie, Wei Song, and Jiebo Luo. 2020. Neural simile recognition with cyclic multitask learning and local attention. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pages 9515–9522.

Zhengyan Zhang, Xu Han, Hao Zhou, Pei Ke, Yuxian Gu, Deming Ye, Yujia Qin, Yusheng Su, Haozhe Ji, Jian Guan, et al. 2020. Cpm: A large-scale generative chinese pre-trained language model. *arXiv preprint arXiv:2012.00413*.

Jin Zhou. 2020. Love is as complex as math: Metaphor generation system for social chatbot. In *Chinese Lexical Semantics: 20th Workshop, CLSW 2019, Beijing, China, June 28–30, 2019, Revised Selected Papers*, volume 11831, page 337. Springer Nature.

A Dataset

A.1 Chinese NM Corpus (CMC)

Examples in CMC are shown in Table 5.

A.2 Chinese Literature Corpus (CLC)

CLC consists of three main categories of Chinese literature: Children's Literature (Children), Chinese Literature (Chinese), Translated Literature (Translated). Statistics of each category are shown in Table 6.

B SCOPE Model

SCOPE model takes a literal expression as input and produces a simile correspondingly. For example, given "the city is beautiful", SCOPE model will transfer the literal expression into a simile: "The city is like a painting".

In our experiments, to compare SCOPE with our method, we first 1) feed a TENOR to COMET (Bosselut et al., 2019) model, to get properties of the TENOR. For example, given a query "<Autumn, SymbolOf>", COMET predicts a list of properties

Label	Sentence	Tenor	Comparator	Vehicle
NM	瀑布注入水潭的一刹那,一朵朵白色的一浪一花腾空而起,像溅玉抛珠一般。	瀑布	像	溅玉抛珠
	At the moment when the waterfall was poured into the pool, a white spray of flowers vacated, like a splash of jade beads.	waterfall	like	jade
	食堂开饭时，全校同学像热锅上的蚂蚁一样挤成一团。	全校同学	像	热锅上的蚂蚁
	When the dining hall opened, the whole school huddled together like ants on a hot pot.	whole school	like	ants on hot pot
Literal	泛着银光的大海在他身后铺展开来。 The silver-filled sea spread out behind him.	-	-	-

Table 5: Examples of metaphor and not metaphor in the CMC.

Category	**#Books**	**#Tokens**	**#Sentences**
Children	195	17M	0.58M
Chinese	336	64M	2.2M
Translated	854	121M	4.2M

Table 6: Summary of CLC.

for Autumn: “Passion, gold” etc. We then 2) construct literal expressions using the TENOR and its properties. For example, “Autumn is a symbol of passion” is obtained. 3) The literal expression is fed to SCOPE model and a simile is produced. For example, ”Autumn is like a lover” is produced by SCOPE model. 4) At last, the simile are concatenate with its literal expression to form a complete NM with context: ”Autumn is a symbol of passion, like a lover”.

C Meta Metric

The CMC corpus is split into a training set (80%) for training the classifier and a test set (20%) for evaluation. We simply add a linear layer plus a binary softmax layer on the RoBERTa model as the NM classifier. The accuracy of the classifier tested on the test set of CMC is 97.89%.

D More Examples

Table 7 shows generations produced by our method given different TENORS.

Text (Chinese)	Text (Translated)
爱像一缕金光，即使在黑夜也能照亮你的心灵。	Love is like a ray of golden light, which can illuminate your heart even at night.
爱像一盏明亮的夜灯，让迷途的航船找到港湾;	Love is like a bright night light, let the lost ship find the harbor.
时间像利剑一样无情的锋刃，一旦出鞘，瞬间就割断你人生的纽带。	Time is a ruthless blade like a sharp sword. Once it comes out of the scabbard, it will cut off the bond of your life in an instant.
秋天像个美人的画笔调侃着大地：世界上再没有比这更美的了。	Autumn teases the earth like a beautiful brush: there is nothing more beautiful in the world.
爱心像一片照射在冬日的光，使饥寒交迫的人感到人间的温暖.	Love is like a piece of sunshine in winter, which makes hungry and cold people feel the warmth of the world

Table 7: More generation examples of CM-Gen.

Psychology-guided Controllable Story Generation

Yuqiang Xie Yue Hu[†] Yunpeng Li Guanqun Bi Luxi Xing Wei Peng
Institute of Information Engineering, Chinese Academy of Sciences, Beijing, China
School of Cyber Security, University of Chinese Academy of Sciences, Beijing, China
{xieyuqiang,huyue,liyunpeng,biguanqun,xingluxi,pengwei}@iie.ac.cn

Abstract

Controllable story generation is a challenging task in the field of NLP, which has attracted increasing research interest in recent years. However, most existing works generate a whole story conditioned on the appointed keywords or emotions, ignoring the psychological changes of the protagonist[‡]. Inspired by psychology theories, we introduce global psychological state chains, which include the needs and emotions of the protagonists, to help a story generation system create more controllable and well-planned stories. In this paper, we propose a **P**sychology-gu**I**ded **C**ontrollable **S**tory Generation System (**PICS**) to generate stories that adhere to the given leading context and desired psychological state chains for the protagonist. Specifically, psychological state trackers are employed to memorize the protagonist's local psychological states to capture their inner temporal relationships. In addition, psychological state planners are adopted to gain the protagonist's global psychological states for story planning. Eventually, a psychology controller is designed to integrate the local and global psychological states into the story context representation for composing psychology-guided stories. Automatic and manual evaluations demonstrate that **PICS** outperforms baselines, and each part of **PICS** shows effectiveness for writing stories with more consistent psychological changes.

1 Introduction

Controllable Story Generation (CSG) is an important task in natural language processing (NLP) (Porteous and Cavazza, 2009; Peng et al., 2018; Alabdulkarim et al., 2021). It has also become one of the test methods for progress in artificial intelligence (AI). Most existing state-of-the-art works (Kong et al., 2021; Rashkin et al., 2020; Paul and Frank, 2021; Xu et al., 2020b) generate a story conditioned by the appointed keywords or emotions, with the help of remarkable pre-trained language models (PLM), like GPT-2 (Radford et al., 2019) and BART (Lewis et al., 2020). While most of these systems have been able to generate fluent stories, CSG still has many issues to be explored.

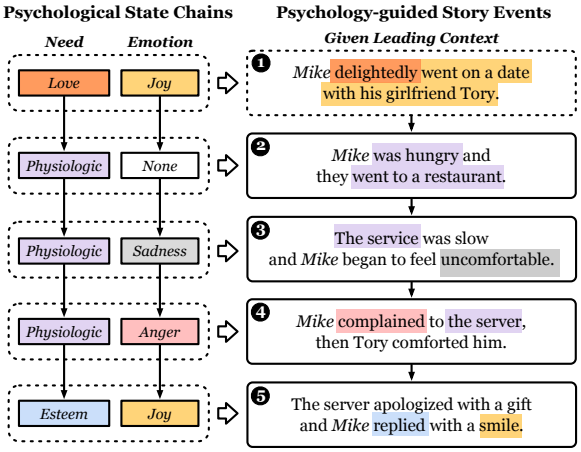

Figure 1: Example of psychology-guided controllable story generation conditioned on dotted frames (global psychological state chains, i.e., need/emotion, as well as leading context). Each psychological state and its corresponding tokens are highlighted in the same color.

In daily life, humans tend to create events driven by their needs (cause) and receive emotions (effect) after the events. Similarly, needs (Ricoeur, 1984) and emotions (Vonnegut, 1981) play the central roles in creating reasonable stories in storytelling. Recently, several works have begun developing CSG systems based on people's expected emotional keywords or scores, such as (Brahman and Chaturvedi, 2020) and (Xu et al., 2020a). Although these approaches can generate stories appointing the desired emotional signals, they are unable to control the storytelling as the protagonist's psychological state changes. Another problem is that these methods only consider the current/previous emotions without global planning, which plays an

[†]Corresponding author.
[‡]In this work, we define the protagonist as the most frequently occurring character in a story (Morrow, 1985).

Proceedings of the 29th International Conference on Computational Linguistics, pages 6480–6492
October 12–17, 2022.

important role in composing a story.

To address the aforementioned problems, we focus on taking the protagonist's global psychological state chains into account in controllable story generation. Researches in cognitive psychology have shown that readers closely monitor the protagonist's needs (Ricoeur, 1984) and emotions (Vonnegut, 1981) while reading narratives. At any point in a story, we represent the protagonist's psychological state using multiple needs and emotions common in psychological theories. *Hierarchy of needs* of Maslow has five categories (i.e., *physiological need, stability, love and belonging, esteem* and *self-actualization*) for describing human needs of a person. *Wheel of emotions* of Plutchik proposes eight basic emotions (includes *joy, trust, anger, surprise, sadness, disgust, fear* and *anticipation*) to adequately portray a person. Motivated by this, we define the psychological state chains as a sequence of five human needs and eight basic emotions that describe psychological states of a protagonist.

Given the protagonist's name and psychological state chains as well as the leading context, our goal is to generate a story about the leading context that adheres to the protagonist's psychological state chains. As exemplified in Figure 1, the protagonist (*Mike*) takes part in each story event controlled by given psychological state chains. Note that, *none* represents that *Mike* has no need or emotion. Each psychological state and its corresponding tokens are in the same color. For example, in the second story event, *Mike was hungry* and *went to a restaurant* obviously embody the **physiologic need** of *Mike*. Another example, as shown in the fourth story event, *complain* action reflects *Mike*'s **anger emotion**. From a global perspective, intuitively, the anterior and hereafter psychological states separately provide the background and guidance for composing stories. As illustrated in the third story event, anterior **physiologic need** leads to *Mike's feeling uncomfortable* due to hungry, and hereafter **anger emotion** guides the setting of *slow service* plot suspense.

To generate stories that adhere to the given leading context and the desired protagonist's global psychological state chains, we propose **PICS** (**P**sychology-gu**I**ded **C**ontrollable **S**tory Generation System), a Transformer-based (Vaswani et al., 2017) architecture. Specifically, psychological state trackers are employed to memorize the local psychological states for capturing temporal relationships among psychological states. And, psychological state planners are adopted to gain the protagonist's global psychological states for planning the storytelling. In the end, a psychology controller is designed to integrate the local and global psychological states into the story context representation for composing psychology-guided stories. Based on the extracted data from publicly available *Story Commonsense* (Rashkin et al., 2018) dataset, experimental results demonstrate that **PICS** outperforms baselines, and the psychological state trackers, planners as well as the psychology controller are important for generating stories with more consistent psychological changes.

2 Related Work

Early story generation systems relied on symbolic planning (Pérez and Sharples, 2001; Porteous and Cavazza, 2009; Riedl and Young, 2010), which had domain restriction and massive cost of feature engineering. Recent seq2seq storytelling models (Roemmele, 2016; Jain et al., 2017) had partially alleviated these problems, most of which focused on learning better representation for a story (Martin et al., 2018; Xu et al., 2018; Fan et al., 2018b, 2019; Yao et al., 2019).

To introduce semantic knowledge into story generation, many methods also employed large-scale pre-trained language models (LM) based on Transformer (Vaswani et al., 2017), like GPT-2 (Radford et al., 2019) and BART (Lewis et al., 2020). After in-domain training, these models can generate fluent and coherent text, which can be used in story generation (Qin et al., 2019; Guan et al., 2020; Xie et al., 2022b) and dialogue systems (Budzianowski and Vulić, 2019; Wolf et al., 2019). However, they lacked the ability of controllable generation, such as expressing specific goals.

Further, aiming at controllable story generation, works had been introduced to control different attributes of the generated text, such as keyword (Fan et al., 2018a), style (Wang et al., 2017) and length (Kikuchi et al., 2016). For example, Tambwekar et al. 2019 introduced reinforcement learning to generate a goal-driven storyline, which is a sequence of event tuples. PPLM (Dathathri et al., 2020) used attribute classifiers to guide text generation without further training of LM. PLOTMACHINES (Rashkin et al., 2020) transformed an outline into a coherent story by tracking the dynamic plot states. Kong et al. 2021 first planned

the stylized keywords and then generated the whole story with the guidance of the keywords. And many works considered commonsense knowledge as an attribute for CSG. Ammanabrolu et al. 2021 performed story generation using soft causal relations, which automatically extracted from existing natural language plot summaries. Paul and Frank 2021 used the contextualized commonsense inference rules generated by COMET (Bosselut et al., 2019) based model to produce a coherent story ending.

Most related to this work, many methods generated text with a specific sentiment or emotion (Zhou et al., 2018; Huang et al., 2018; Zhou and Wang, 2018; Song et al., 2019). Rashkin et al. 2018 present an annotation framework specifically designed to examine the mental states of characters in commonsense based stories. There are some limitations to incorporating sentiment, emotion or psychological state for story generation. Previous work modeled characters but not sentiment (Clark et al., 2018; Liu et al., 2020). Peng et al. 2018 and Luo et al. 2019 controlled the overall sentiment for story ending generation. Weber et al. 2020 incorporated sentiment trajectory by a new task that "filling in" a story. Brahman and Chaturvedi 2020 modeled the emotional trajectory of the protagonist for story generation. Xu et al. 2020a generated a story with multiple emotional changes of protagonists based on the given characters and the corresponding psychological state lines. These works are limited to the guiding of emotion scores or tokens or/and target the ending sentence. Lately, Xie et al. 2022a modeling the relationship among motivations, actions and emotions based on human activities (i.e. story events), which lacks consideration of the global changes in a story. Different from the above methods, we respectively model the local and global psychological state changes of the protagonist as the story progresses, which is more central to storytelling than the emotion trajectory.

3 Task Definition

We formulate our psychology-oriented controllable story generation task in the following. Note that, the length of the whole story is 5 in this paper, and the output story event is in the m-th time point. Table 1 shows an example of our task.

Input The context $\mathbb{X} = (\mathcal{X}_1, \mathcal{X}_2, \ldots, \mathcal{X}_{m-1})$ to the current story event with $m-1$ events, where the i-th event $\mathcal{X}_i = (x_i^1, x_i^2, \ldots, x_i^k)$ consists of k words. The name of protagonist $\mathbb{P} = (p_1, p_2, \ldots, p_m)$ to indicate the expected participant of the generated story event, the elements are the same in $\mathbb{P}$ in our setting. The protagonist's global psychological state chains, including the need chain $\mathbb{A}_\mathbb{N} = (n_1, n_2, \ldots, n_5)$ and the emotion chain $\mathbb{A}_\mathbb{E} = (e_1, e_2, \ldots, e_5)$. The protagonist's local psychological states from $\mathbb{A}_\mathbb{N}$ and $\mathbb{A}_\mathbb{E}$, including the need history $\mathbb{N} = (n_1, n_2, \ldots, n_m)$ and the emotion history $\mathbb{E} = (e_1, e_2, \ldots, e_m)$. n_i and e_i represent the protagonist's need and emotion for the i-th story event, where $i \in [1, m]$.

Protagonist	Donald, He, Donald, He, He
Need Chain	esteem, esteem, esteem, esteem, esteem
Emotion Chain	joy, sadness, sadness, sadness, joy
Leading Context	Donald was a senator.
Event $\mathcal{Y}_2$	He ran as an indie candidate.
Event $\mathcal{Y}_3$	Donald wanted better implementation of his policies.
Event $\mathcal{Y}_4$	He decided to run in the next term as a Republican.
Event $\mathcal{Y}_5$	He won again, and is now in a better position.

Table 1: An example of our task.

Output $\mathcal{Y}_m = (y_1, y_2, \ldots, y_r)$ (also X_m in the next time step) stands for the current story event that consists of r words, based on the protagonist's name $\mathbb{P}$, the need history $\mathbb{N}$, the emotion history $\mathbb{E}$, the need chain $\mathbb{A}_\mathbb{N}$ and the emotion chain $\mathbb{A}_\mathbb{E}$, where y_i is the i-th word.

4 Methodology

The overall architecture of our proposed **PICS** system is illustrated in Figure 3. In the following, we will describe each component in more detail.

4.1 Contextual Encoder (Step 1)

In order to capture the contextual semantic information for the story context and the protagonist's historical psychological states, we reconstruct the input of the embedding layer in the backbone BART (Lewis et al., 2020) model.

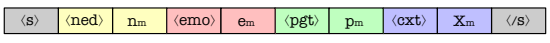

Figure 2: Reconstructed input of the word embedding layer in BART for m-th event.

As illustrated in Figure 2, we employ new special $\{\langle\texttt{ned}\rangle, \langle\texttt{emo}\rangle, \langle\texttt{pgt}\rangle\}$ tokens to delimit each protagonist's need, emotion and name grounded in the story context. In addition, we utilize a special $\langle\texttt{cxt}\rangle$ token to delimit each story event of the context:

$$b_m = \texttt{Emb}(n_1, e_1, p_1, X_1, \ldots, n_m, e_m, p_m) \quad (1)$$

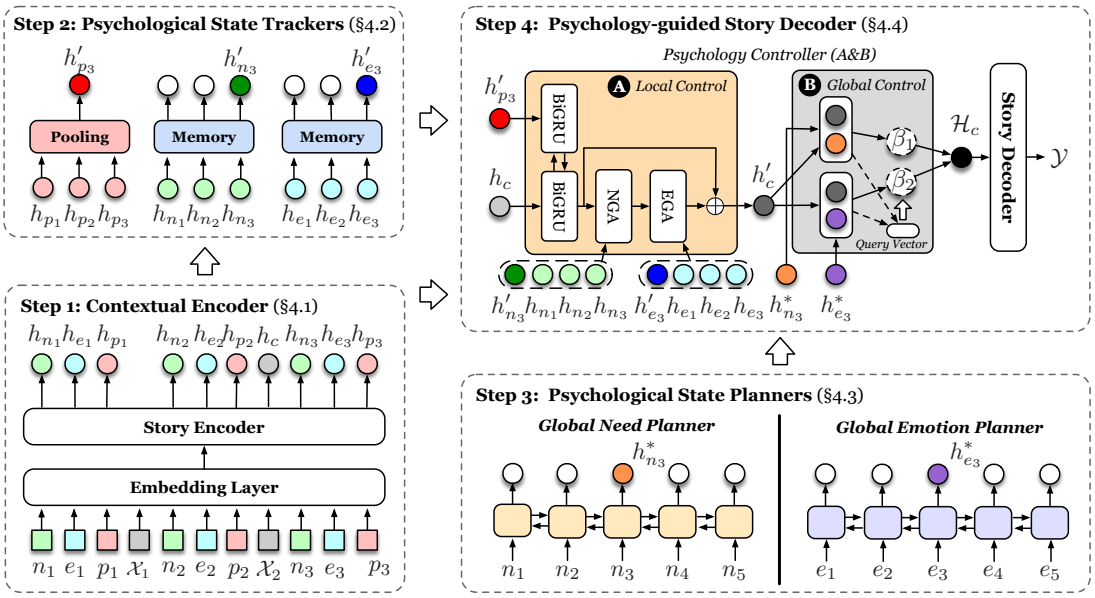

Figure 3: Overview of **PICS** with time point m=3 **(Step 1-4)**. In step 1, contextual encoder converts input into contextual representation (§ 4.1). In step 2, we design psychological state trackers to capture temporal relations for the protagonist's character information, local need and emotion (§ 4.2). In step 3, two psychological state planners output the global psychological state through modeling the completed need and emotion chains (§ 4.3). In step 4, conditioned on the protagonist's local and global psychological states, the decoder generates psychology-guided stories with a psychology controller (A&B) (§ 4.4).

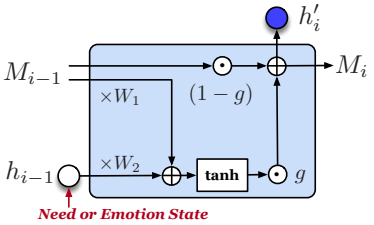

Figure 4: Details of Memory Units in Psychological State Trackers for need and emotion states.

Then, we acquire representations of i-th psychological states $h_{n_i}, h_{e_i}, h_{p_i}$ by extracting the hidden states of special $\{\langle \texttt{ned} \rangle, \langle \texttt{emo} \rangle, \langle \texttt{pgt} \rangle\}$ tokens on the top of encoder.

Similarly, story context representation is corresponding to the hidden state of $(m-1)$-th special $\langle \texttt{cxt} \rangle$ token.

4.2 Psychological State Trackers (Step 2)

For the purpose of remembering and updating the protagonist's psychological states that have been mentioned, we design psychological state trackers for the protagonist's character information[§], needs and emotions.

[§]In this paper, we regard the representation of the protagonist's name as his/her character information.

Protagonist's Character Information Based on several story events, humans can easily guess the protagonist's character information. We argue that the resulting representation can stand for the protagonist's character information via pooling hidden states of the protagonist's name which is grounded in story events.

$$h'_p = \texttt{Pooling}(\{h_{p_i}\}_{i=1}^{m}) \quad (2)$$

In this work, `Pooling` is Mean-Pooling and is used as a tracker for p_i to conclude the moderate representation of the protagonist's character information.

Protagonist's Needs To remember and update the mentioned protagonist's needs and emotions, we design trackers (memory block in Figure 3) as follows:

$$h'_n = \texttt{Memory}(\mathbb{N}) \quad (3)$$

As shown in Figure 4, for memorizing mentioned needs, the memory unit M_{n_i} is updated using $h_{n_{i-1}}$ and $M_{n_{i-1}}$, the output contextual needs representation:

$$\widehat{h}_{n_{i-1}} = \texttt{tanh}(W_1 M_{n_{i-1}} + W_2 h_{n_{i-1}}) \quad (4)$$

Futher, we use a gating mechanism, g, to allow the model to learn to flexibly control how much each

cell in memory is updated, as below:

$$g_{n_i} = \texttt{sigmoid}(W_3 M_{n_{i-1}} + W_4 h_{n_{i-1}}) \quad (5)$$

$$M_{n_i} = g_{n_i} \widehat{h}_{n_{i-1}} + (1 - g_{n_i}) M_{n_{i-1}} \quad (6)$$

$$h'_{n_m} = M_{n_m} \quad (7)$$

where M_{n_0} is randomly initialized and all W_* are trainable parameters.

Protagonist's Emotions In the same way with need, for historical emotions:

$$h'_{e_m} = \texttt{Memory}(\mathbb{E}) = M_{e_m} \quad (8)$$

In summary, we can obtain sequential psychological state changes through the above operations.

4.3 Psychological State Planner (Step 3)

Different from psychological state trackers, we obtain the global psychological states by encoding the global needs chain and emotions chain for story planning. As shown in Figure 3 (step 3), psychological state trackers use a BiGRU (Cho et al., 2014) architecture.

Global Need Planner In the global need planner, the global need representations:

$$[t_n^1, t_n^2, \ldots, t_n^5] = \texttt{BiGRU}(\mathbb{A}_\mathbb{N}) \quad (9)$$

$$h^*_{n_j} = t_n^j \quad (10)$$

where j represents the j-th need for generate next story event and j-th need in the global need chain.

Global Emotion Planner Similarly, computing global emotion representations are as below:

$$h^*_{e_j} = \texttt{BiGRU}(\mathbb{A}_\mathbb{E})[j] = t_e^i \quad (11)$$

note that, $\mathbb{A}_\mathbb{N}$ and $\mathbb{A}_\mathbb{E}$ are initialized by GloVe (Pennington et al., 2014) embedding.

After obtaining global need and emotion representations, we feed them into the following step as a planning signal for guiding the story generation.

4.4 Psychology-guided Decoder (Step 4)

Conditioned on the protagonist's local and global psychological states, the decoder generates a psychology-guided story event by the following modules.

4.4.1 Psychology Controller

In order to control the story generation by protagonist's psychological states, we respectively integrate local and global psychological states (Peng et al., 2022; Zhang et al., 2022) into the story context representation.

Local Control With the goal to integrate local psychological control information into the representation of story context h_c, a psychology controller is used to compute the interaction between h_c and local psychological states (including h'_p, h'_n, h'_e). First, h'_p guides the model with the protagonist's character information for generating the next story event, which uses a BiGRU (Cho et al., 2014) architecture:

$$\widetilde{h_c} = \texttt{BiGRU}(h_c, h'_p)[0] \quad (12)$$

where we extract the hidden state $\widetilde{h_c}$ as the story context representation considering character information. Then, we employ an attention mechanism to integrate local psychological states (need and emotion) into the story context representation. Firstly, need guided attention NGA is defined as follows:

$$h_c^n = \texttt{NGA}(\widetilde{h_c}, \{h_{n_i}\}_{i=1}^m) = \sum_{i=1}^m \alpha_i h_{n_i} \quad (13)$$

$$\{\alpha_i\}_{i=1}^m = \textit{softmax}(\{\widetilde{h_c} h_{n_i}^T / \sqrt{dim_1}\}_{i=1}^m) \quad (14)$$

$$\widehat{h_c^n} = \texttt{Fus.N}(h'_n, h_c^n) = \texttt{MLP}([h'_n, h_c^n]) \quad (15)$$

where dim_1 equals to the dimension of h_c^n. Similarly, EGA and Fus.E has the same operation as the above equations:

$$\widehat{h_c^e} = \texttt{Fus.E}(\texttt{EGA}(\widehat{h_c^n}, \{h_{e_i}\}_{i=1}^m)) \quad (16)$$

$$h'_c = \widehat{h_c^e} + \widetilde{h_c} \quad (17)$$

Global Control Aiming at further control composing stories with the global planning signal, this part is designed to dynamically integrate global psychological states. In specific, a query vector q is introduced to fuse psychology-blended context representations and global psychological states by attention mechanism as below:

$$s_n, s_e = \frac{q[h'_c, h^*_{n_i}]^T}{\sqrt{dim_2}}, \frac{q[h'_c, h^*_{e_i}]^T}{\sqrt{dim_2}} \quad (18)$$

$$\beta_1, \beta_2 = \textit{softmax}(s_n, s_e) \quad (19)$$

$$\mathcal{H}_c = \texttt{MLP}(\beta_1[h'_c, h^*_{n_i}] + \beta_2[h'_c, h^*_{e_i}]) \quad (20)$$

where q is the query and $[h'_c, h^*_{n_i}], [h'_c, h^*_{e_i}]$ are the keys for attention. $[\cdot]$ denotes the concatenation operation. dim_2 equals to the dimension of $[h'_c, h^*_{n_i}]$. So that the model can adaptively choose the most important global psychological state for generating well-planned stories.

4.4.2 Story Decoder

We employ a left-to-right BART decoder to generate a story conditioned upon all input elements. Each layer of the decoder additionally performs cross-attention over the concatenation of the final hidden layer of the BART encoder and $\mathcal{H}_c$.

$$P(y_t|\mathbb{X}, \mathbb{P}, \mathbb{N}, \mathbb{E}, \mathbb{A}_\mathbb{N}, \mathbb{A}_\mathbb{E}, y_{<t}) = \textit{softmax}(W_s s_t) \quad (21)$$

$$s_t = \texttt{Dec}(y_{<t}, \texttt{Enc}(b_m), \mathcal{H}_c) \quad (22)$$

where $\texttt{Enc}(b_m)$ is the final hidden layer of the BART encoder.

4.4.3 Training

The training objective is to minimize the negative log-likelihood L of the ground truth story event:

$$\mathcal{L} = -\sum_{t=1}^{r} \log P\left(y_t \mid \mathbb{X}, \mathbb{P}, \mathbb{N}, \mathbb{E}, \mathbb{A}_\mathbb{N}, \mathbb{A}_\mathbb{E}, y_{<t}\right) \quad (23)$$

Aiming to obtain the completed story, we iteratively generate the m-th story events $\mathcal{Y}_m$ with the forecast generated $\mathcal{Y}_1, ..., \mathcal{Y}_{m-1}$.

5 Experiment

5.1 Data

We choose a Story Commonsense (Rashkin et al., 2018) that has been annotated with a similar setting to us. Story Commonsense is a large-scale dataset as a resource for training and evaluating the mental state tracking of characters in short commonsense stories. This dataset contains over 300k low-level annotations for character motivations and emotional reactions. Story Commonsense was proposed for studying need/emotion tracking. Each sentence is annotated for all characters, and there are 3 crowd-workers voting for each need/emotion. If the characters have no need or emotion, the psychological state will be labeled 'none'.

Based on our task definition, we extract the story with a protagonist (occurs in more than 4 sentences) and the corresponding need/emotion chains from Story Commonsense. Note that, we select the **Top-1** need/emotion label, based on annotators' voting scores to make up the psychological chains in our data set. If several labels have the best score of all, we will choose a low-level need label of Maslow's needs or a random emotion label. Following the 8:1:1 splitting ratio, we obtain 2,570/321/321 five-sentence stories for train/dev/test sets. In our psychology-guided CSG task, we generate a story event based on the story context, protagonist's name, need chain, and emotion chain. Therefore, each story will be reformed into 4 samples following our setting (i.e. 10,280/1283/1283 samples).

5.2 Implement Details

For a fair comparison, we train our proposed models and the baselines with the same input (leading context and global need/emotion chains) that are automatically extracted from Story Commonsense (Rashkin et al., 2018). Our proposed models follow the setting of BART large (Lewis et al., 2020) model with 12 layers in each of the encoder and decoder and a hidden size of 1024. The stories are encoded using BPE with a vocabulary size of 50,257. We set the maximum sequence length to 100 tokens, as it is large enough to contain all inputs. We use Adam optimization with an initial learning rate of 0.00001. All models were trained until there was no improvement in the validation set performance. During training, we use a label smoothed cross-entropy loss, with the smoothing parameter set to 0.1. At inference time, we set beam size as 5, and remove duplicated trigrams in beam search. We use the HuggingFace [¶] (Wolf et al., 2020) PyTorch (Paszke et al., 2019) implementation[‖] on Tesla V100 GPU.

5.3 Evaluation Metrics

Automatic Metrics We use the following metrics for automatic evaluation: **(1) Perplexity (PPL)** is an indicator of fluency. A smaller value is better. **(2) BLEU** (Papineni et al., 2002) is used for evaluating the overall quality of the generated story. We use n=1, 2. **(3) Rouge** (Lin, 2004) with n=1, 2, L is used to measure the similarity between automatically generated and reference results. **(4) Need/Emotion Consistency (NC/EC)** It is a learnable automatic metric. We fine-tune a RoBERTa

[¶] https://github.com/huggingface/transformers

[‖] We will make our dataset and code publicly available at https://github.com/IndexFziQ/PICS.

Models	PPL ↓	BLEU-1 ↑	BLEU-2 ↑	Rouge-1 ↑	Rouge-2 ↑	Rouge-L ↑	NC ↑	EC ↑
Fusion	25.68	19.83	2.89	5.81	1.66	8.64	0.19	0.16
Plan&Write	19.43	20.15	3.56	6.23	1.85	8.81	0.31	0.19
PPLM	-	18.42	4.39	8.73	2.13	9.29	0.45	0.41
GPT-2 FT	18.21	22.67	6.42	9.97	2.25	9.98	0.34	0.37
BART FT	17.85	21.84	6.03	9.32	2.51	9.56	0.36	0.32
PICS	**16.73**	**23.51**	**6.89**	**12.43**	**3.83**	**11.28**	**0.64**	**0.45**

Table 2: The results of automatic evaluation on test set considering common-used metrics and the designed metric (NC/EC) to test the psychological state consistency of stories. ↓/↑ indicates the lower/higher, the better.

NC		EC	
Accuracy	F1-Score	Accuracy	F1-Score
64.6	64.8	56.8	59.2

Table 3: Accuracy and F1-Score of RoBERTa classifier for need/emotion on dev set, respectively.

(Liu et al., 2019) large model on the Story Commonsense (Rashkin et al., 2018) train set as a classifier to distinguish whether a story event is corresponding to a **Top-1** need/emotion. Table 3 shows results of NC/EC.

Manual Metrics We also conduct a manual evaluation of generated psychology-guided stories. Following Song et al. (2019), crowd-workers are required to evaluate actions on a 0-3 scale (3 being very good) from two different perspectives:

(1) Content Quality to indicate whether the generated story is fluent. **(2) Content Rationality** to assess whether it follows the given needs and emotions which is reasonable and consistent.

During the manual evaluation, we display the input (leading context and global need/emotion arc) and two stories generated by the two models being compared. To avoid prejudice, we randomly changed the order in which the stories in the two models were displayed to the crowd-workers. We provided crowd-workers with instructions to explain the annotations and provided examples. Following this process, each pair of stories is annotated by three crowd-workers.

5.4 Experimental Results

5.4.1 Baselines

For a fair comparison, we train **PICS** and the baselines with the same input (leading context and global need/emotion chains).

We compare our base storytelling model, **PICS**, with following state-of-the-art models:

1. **Fusion** (Fan et al., 2018b), a storytelling model that first pre-trains a convolutional seq2seq model, then fixes the trained model and passes it to the second clone model with fusion mechanism.

2. **Plan&Write** (Yao et al., 2019), another storytelling model first generates a plot as a sequence of keywords with the given leading context and then conditioned on the plot it generates the text of the story.

3. **PPLM** (Dathathri et al., 2020), which can be extended to accept psychological state chains for controlling story generation. We use psychological state chains as the skeleton.

4. **GPT-2 FT** (Radford et al., 2019) is a pre-trained generative LM. We use a medium-size version. We fine-tuned GPT-2 on our dataset following (Guan et al., 2020) with leading context and global need/emotion chains.

5. **BART FT** (Lewis et al., 2020) is a encoder-decoder architecture. We fine-tuned BART on our dataset following (Lewis et al., 2020) with leading context and global need/emotion chains.

All of these models are trained, validated and tested on the same data splits described in §5.1. In specific, we add emotion/need labels as additional input tokens to baseline models alongside the tokens for each story sentence. And, global emotion/need chains that concatenated with the story context are given to baseline models at each time step.

5.4.2 Automatic Evaluation

The results of the automatic evaluation are shown in Table 2. Our model outperforms the variants of GPT-2 in terms of perplexity, and has higher BLEU and Rouge scores than all the baselines, indicating

Model	Quality ↑		Rationality ↑	
	Fluency	Coherence	Need	Emotion
PPLM	2.56	1.62	1.05	1.27
GPT-2 FT	**2.87**	1.58	1.73	1.82
BART FT	2.72	**1.79**	1.69	1.98
PICS	2.83	1.68	**2.16**	**2.34**

Table 4: Manual Evaluation in terms of content quality and content rationality about the generated stories.

Model	AVG-B ↑	AVG-R ↑	NC ↑	EC ↑
PICS	15.89	8.65	0.64	0.45
w/o PST	15.41	8.45	0.53	0.46
w/o PSP	**15.24**	8.47	0.56	0.43
w/o PC	15.34	**8.24**	0.55	0.42
w/o Need	15.71	8.53	**0.42**	0.39
w/o Emotion	15.64	8.44	0.51	**0.35**

Table 5: Ablation study of **PICS** model and global need/emotion chains on dev set. PST: psychological state tracker. PSP: psychological state planner. PC: psychology controller.

better fluency and more overlaps with the reference stories. Besides, in the view of NC and EC scores, the stories generated by **PICS** are more consistent with the desired psychological state chains, either need or emotion.

5.4.3 Manual Evaluation

We perform a manual evaluation between our model and baselines. We randomly generate 100 stories from the test set. For each story, we hire three annotators to give a score in terms of content quality (fluency&coherence) and content rationality (need&emotion). For each aspect, we use an average of the three annotations. We adopt majority voting to make the final decisions among the annotators. As shown in Table 4, all the results show that our model outperforms baselines significantly in fluency, coherence, and psychological state consistency.

6 Discussion and Analysis

6.1 Ablation Study

An ablation study is conducted on the Story Commonsense dataset to examine the impact of each module separately. We train the model each time by excluding one of our model's modules. And, we summarize the results in Table 5. The results illustrate the harms that the elimination of each of the proposed modules from **PICS** architecture could cause. This attests to the effectiveness of all proposed approaches in the generation of higher controllable stories and subsequently resulting in more accurate evaluation metrics. As this table demonstrates, the NC/EC accuracy drops the most by ablating the psychological state planner and the psychology controller, which shows that they have the most significant role in composing high-quality psychology-guided stories and consequently accurate evaluation metrics. Besides, psychological state chains (need/emotion chains) all contributed to the high-quality psychology-guided stories.

PICS v.s.	Win	Loss	Tie	κ
PPLM	**54.6%**	18.5%	26.9%	30.2
GPT-2 FT	**53.2%**	19.5%	27.3%	30.4
BART FT	**52.3%**	18.4%	29.3%	27.9

Table 6: Human A/B Test of **PICS**. Results show that **PICS** performs baseline models sufficiently. κ denotes Fleiss' kappa (all are fair agreement or moderate agreement). The p-value of scores < 0.05 in sign test.

6.2 Human A/B Test

We try to compare our model with other baselines by conducting a Human A/B test. Particularly, we randomly sample 100 examples each for our model and baseline models. Three annotators are given generated responses from either our model or baselines in random order and are required to choose a better option. They can either choose one of the responses or select "Tie" when the quality of provided options is hard to access. Results in Table 6 confirm that the responses from **PICS** are more preferred by human judges.

6.3 Case Study

In this section, we present some generated examples in Table 7. We select need-related keywords and emotion-related keywords (based on our observation), which are highlighted in corresponding colors, respectively. From the third block, **PICS** can generate more natural and reasonable psychology-guided stories than baselines. Since the proposed models can generate stories conditioned on the protagonist's psychological state chains, they can be used to unfold a story in diverse situations. We demonstrate this capability in the last 4 blocks of Table 7 which perform counterfactual transformations on need/emotion chains. It shows two examples where for the same leading context, our model can generate stories that follow the counterfactual

Pgt.	*Tory*
Ned.	Stab. → Love → Love → Love → Love
Emo.	Fear →Fear → Joy → Joy →Joy
Cxt.	**Tory had doubts about getting married.**
Golden	She talked to her fiance about their decision. The groom reassured her that he loved her. She remembered how much she loved him too. The wedding went forward without anymore problems.
GPT-2	She called her husband. She got the answer and laugh. She go to shopping then. She bought so many clothes she like.
BART	She talked to her boyfriend about their relationship. Her boyfriend loved her and gave her a kiss. She was very happy then. They went on the date outside for fun.
PICS	Tory is afraid about their marriage. Her boyfriend gave a gift as a blessing. Tory was happy about the gift, and reply him with a kiss. Tory went home to prepare a surprise for him.
Ned.-CF	Stab. → Love → Love →Phys. → Phys.
PICS	She talked to her husband about their marriage. Her husband loved her and take her to the romantic dinner. It is time for them to have a dinner. She was happy with the delicious dinner.
Emo.-CF	Fear →Fear →Anger →Anger →Anger
PICS	She and her husband talked about their marriage. Her husband was angry at her doubts. She rushed and fought with his husband. She gave him a slap and broke up unhappy.

Table 7: Generated stories by different models with need and emotion chains. Each psychological state and its corresponding tokens are highlighted in the same color. **CF** represents performing counterfactual transformation on need chains or emotion chains.

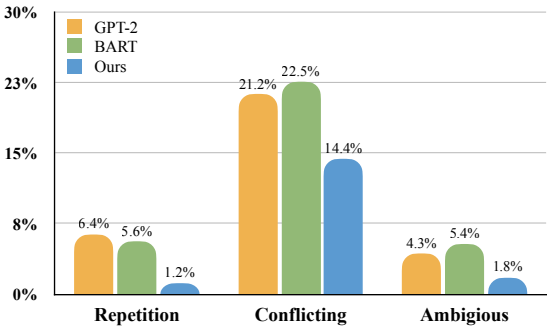

Figure 5: Distribution of error types for **PICS** (ours) and baseline models (GPT-2 and BART).

Error Type	Cases
Repetition	I went to a friends house for a party last weekend. *I was so excited to go.* We played a lot of games that night. *I was so excited to go.* We also had a lot of food to eat.
Conflicting	Zack wanted to vote in the election. Unfortunately Zack was traveling during the election. *Zack went to the school for taking class.* Zack then voted again. Zack was happy to have voted.
Ambiguous	Alice made a cake for her mother. *The cake is so sweet that her mother disliked it. But her mother was very happy and encouraged Alice. Alice was sad because of the failed cooking.* They hugged and smiled in the end.

Table 8: Cases of different typical errors. *Italic* words denote the error story events.

psychological state chains of the protagonist.

6.4 Error Analysis

Although the proposed model outperforms the state-of-the-art baselines, it needs to be noted that there are still many unreasonable stories losing to other models in human evaluation. Therefore, we analyzed error types by manually checking all lost stories in pairwise comparisons between our model and two strong baselines including GPT-2 and BART to reveal the factors that affect the performance. The numbers of stories which lost to our model are 56/64 of 100/100 in total for GPT-2 and BART, respectively. And there are 61 stories of 200 generated by **PICS** losing to these two baselines.

We conclude three main types of error from the lost stories: **repetition** (repeating the same actions about the need/emotion), **conflicting psychological state** (wrong causal relation about psychological state), and **ambiguous psychological state** (difficult to understand the psychological state). The distribution of different error types is shown in Figure 5. We can observe that conflicting and ambiguous psychological state make up most of the errors for all the models. Compared with GPT-2 and BART, **PICS** reduces chaotic scenes effectively but still suffers from severe repetition, as shown in Table 8. However, the analysis result illustrates that generating a reasonable psychology-guided story is a challenging task.

7 Conclusion and Future Work

In this paper, we propose a **PICS** system to generate controllable stories that adhere to the story context and protagonist's psychological state chains. Specifically, we model and integrate local and global psychological states of the protagonist as the story progress. Experiments demonstrate that **PICS** significantly outperforms baselines and each part shows effectiveness. In future work, it is important to build a large-scale dataset for developing psychology-guided controllable story generation, regarding aspects of multilingual and long text. Besides, our methodology can be generalized to a wide range of areas, such as automatic storytelling systems and intelligent education agents.

Acknowledgement

We thank all anonymous reviewers for their constructive comments and useful advice. Also thanks for the discussion with Yajing Sun, Yongxiu Xu, and Ping Guo. This work is supported by the National Natural Science Foundation of China (No.62006222 and No.U21B2009). Thanks for COLING organizers and the proposed pre-trained language models, data, codes.

Contribution List Yuqiang Xie: Idea, Paper Writing, Coding; Yue Hu: Guiding, Discussion; Yunpeng Li: Coding; Guanqun Bi: Paper Polish, Discussion; Luxi Xing: Review; Wei Peng: Review.

Thanks for the hard work and dedication of all team members.

References

Amal Alabdulkarim, Siyan Li, and Xiangyu Peng. 2021. Automatic story generation: Challenges and attempts. In *Proceedings of the Third Workshop on Narrative Understanding*, pages 72–83, Virtual. Association for Computational Linguistics.

Prithviraj Ammanabrolu, W. Cheung, William Broniec, and Mark O. Riedl. 2021. Automated storytelling via causal, commonsense plot ordering. In *AAAI*.

Antoine Bosselut, Hannah Rashkin, Maarten Sap, Chaitanya Malaviya, Asli Celikyilmaz, and Yejin Choi. 2019. COMET: Commonsense transformers for automatic knowledge graph construction. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 4762–4779, Florence, Italy. Association for Computational Linguistics.

Faeze Brahman and Snigdha Chaturvedi. 2020. Modeling protagonist emotions for emotion-aware storytelling. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 5277–5294, Online. Association for Computational Linguistics.

Paweł Budzianowski and Ivan Vulić. 2019. Hello, it's GPT-2 - how can I help you? towards the use of pretrained language models for task-oriented dialogue systems. In *Proceedings of the 3rd Workshop on Neural Generation and Translation*, pages 15–22, Hong Kong. Association for Computational Linguistics.

Kyunghyun Cho, Bart van Merriënboer, Dzmitry Bahdanau, and Yoshua Bengio. 2014. On the properties of neural machine translation: Encoder–decoder approaches. In *Proceedings of SSST-8, Eighth Workshop on Syntax, Semantics and Structure in Statistical Translation*, pages 103–111, Doha, Qatar. Association for Computational Linguistics.

Elizabeth Clark, Yangfeng Ji, and Noah A. Smith. 2018. Neural text generation in stories using entity representations as context. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 2250–2260, New Orleans, Louisiana. Association for Computational Linguistics.

Sumanth Dathathri, Andrea Madotto, Janice Lan, Jane Hung, Eric Frank, Piero Molino, Jason Yosinski, and Rosanne Liu. 2020. Plug and play language models: A simple approach to controlled text generation. In *8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, April 26-30, 2020*. OpenReview.net.

Angela Fan, David Grangier, and Michael Auli. 2018a. Controllable abstractive summarization. In *Proceedings of the 2nd Workshop on Neural Machine Translation and Generation*, pages 45–54, Melbourne, Australia. Association for Computational Linguistics.

Angela Fan, Mike Lewis, and Yann Dauphin. 2018b. Hierarchical neural story generation. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 889–898, Melbourne, Australia. Association for Computational Linguistics.

Angela Fan, Mike Lewis, and Yann Dauphin. 2019. Strategies for structuring story generation. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 2650–2660, Florence, Italy. Association for Computational Linguistics.

Jian Guan, Fei Huang, Zhihao Zhao, Xiaoyan Zhu, and Minlie Huang. 2020. A knowledge-enhanced pretraining model for commonsense story generation. *Transactions of the Association for Computational Linguistics*, 8:93–108.

Chenyang Huang, Osmar Zaïane, Amine Trabelsi, and Nouha Dziri. 2018. Automatic dialogue generation with expressed emotions. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 2 (Short Papers)*, pages 49–54, New Orleans, Louisiana. Association for Computational Linguistics.

Parag Jain, Priyanka Agrawal, Abhijit Mishra, Mohak Sukhwani, Anirban Laha, and Karthik Sankaranarayanan. 2017. Story generation from sequence of independent short descriptions. *ArXiv*, abs/1707.05501.

Yuta Kikuchi, Graham Neubig, Ryohei Sasano, Hiroya Takamura, and Manabu Okumura. 2016. Controlling output length in neural encoder-decoders. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 1328–1338, Austin, Texas. Association for Computational Linguistics.

Xiangzhe Kong, Jialiang Huang, Ziquan Tung, Jian Guan, and Minlie Huang. 2021. Stylized story generation with style-guided planning. In *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*, pages 2430–2436, Online. Association for Computational Linguistics.

Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Veselin Stoyanov, and Luke Zettlemoyer. 2020. BART: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7871–7880, Online. Association for Computational Linguistics.

Chin-Yew Lin. 2004. ROUGE: A package for automatic evaluation of summaries. In *Text Summarization Branches Out*, pages 74–81, Barcelona, Spain. Association for Computational Linguistics.

Danyang Liu, Juntao Li, Meng-Hsuan Yu, Ziming Huang, Gongshen Liu, Dongyan Zhao, and Rui Yan. 2020. A character-centric neural model for automated story generation. In *The Thirty-Fourth AAAI Conference on Artificial Intelligence, AAAI 2020*, pages 1725–1732. AAAI Press.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, M. Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Roberta: A robustly optimized bert pretraining approach. *ArXiv*, abs/1907.11692.

Fuli Luo, Damai Dai, Pengcheng Yang, Tianyu Liu, Baobao Chang, Zhifang Sui, and Xu Sun. 2019. Learning to control the fine-grained sentiment for story ending generation. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 6020–6026, Florence, Italy. Association for Computational Linguistics.

Lara J. Martin, Prithviraj Ammanabrolu, Xinyu Wang, William Hancock, Shruti Singh, Brent Harrison, and Mark O. Riedl. 2018. Event representations for automated story generation with deep neural nets. In *Proceedings of the Thirty-Second AAAI Conference on Artificial Intelligence, (AAAI-18)*, pages 868–875. AAAI Press.

Abraham Harold Maslow. 1943. A theory of human motivation. In *Psychological review*.

D. Morrow. 1985. Prominent characters and events organize narrative understanding. *Journal of Memory and Language*, 24:304–319.

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. Bleu: a method for automatic evaluation of machine translation. In *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics*, pages 311–318, Philadelphia, Pennsylvania, USA. Association for Computational Linguistics.

Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Köpf, Edward Yang, Zachary DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. 2019. Pytorch: An imperative style, high-performance deep learning library. In *Advances in Neural Information Processing Systems 32: Annual Conference on Neural Information Processing Systems 2019, NeurIPS 2019, December 8-14, 2019, Vancouver, BC, Canada*, pages 8024–8035.

Debjit Paul and Anette Frank. 2021. COINS: Dynamically generating COntextualized inference rules for narrative story completion. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 5086–5099, Online. Association for Computational Linguistics.

Nanyun Peng, Marjan Ghazvininejad, Jonathan May, and Kevin Knight. 2018. Towards controllable story generation. In *Proceedings of the First Workshop on Storytelling*, pages 43–49, New Orleans, Louisiana. Association for Computational Linguistics.

Wei Peng, Yue Hu, Luxi Xing, Yuqiang Xie, Yajing Sun, and Yunpeng Li. 2022. Control globally, understand locally: A global-to-local hierarchical graph network for emotional support conversation. In *Proceedings of the Thirty-First International Joint Conference on Artificial Intelligence, IJCAI 2022, Vienna, Austria, 23-29 July 2022*, pages 4324–4330. ijcai.org.

Jeffrey Pennington, Richard Socher, and Christopher Manning. 2014. GloVe: Global vectors for word representation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1532–1543, Doha, Qatar. Association for Computational Linguistics.

R. Y. Pérez and M. Sharples. 2001. Mexica: A computer model of a cognitive account of creative writing. *Journal of Experimental & Theoretical Artificial Intelligence*, 13:119 – 139.

Robert Plutchik. 1980. A general psychoevolutionary theory of emotion. In *Theories of emotion*.

J. Porteous and M. Cavazza. 2009. Controlling narrative generation with planning trajectories: The role of constraints. In *ICIDS*.

Lianhui Qin, Antoine Bosselut, Ari Holtzman, Chandra Bhagavatula, Elizabeth Clark, and Yejin Choi. 2019. Counterfactual story reasoning and generation. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 5043–5053, Hong Kong, China. Association for Computational Linguistics.

Alec Radford, Jeff Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. 2019. Language models are unsupervised multitask learners. In *OpenAI Blog*.

Hannah Rashkin, Antoine Bosselut, Maarten Sap, Kevin Knight, and Yejin Choi. 2018. Modeling naive psychology of characters in simple commonsense stories. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 2289–2299, Melbourne, Australia. Association for Computational Linguistics.

Hannah Rashkin, Asli Celikyilmaz, Yejin Choi, and Jianfeng Gao. 2020. PlotMachines: Outline-conditioned generation with dynamic plot state tracking. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 4274–4295, Online. Association for Computational Linguistics.

Paul Ricoeur. 1984. Time and narrative.

Mark O. Riedl and R. M. Young. 2010. Narrative planning: Balancing plot and character. *ArXiv*, abs/1401.3841.

Melissa Roemmele. 2016. Writing stories with help from recurrent neural networks. In *Proceedings of the Thirtieth AAAI Conference on Artificial Intelligence, February 12-17, 2016, Phoenix, Arizona, USA*, pages 4311–4342. AAAI Press.

Zhenqiao Song, Xiaoqing Zheng, Lu Liu, Mu Xu, and Xuanjing Huang. 2019. Generating responses with a specific emotion in dialog. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 3685–3695, Florence, Italy. Association for Computational Linguistics.

Pradyumna Tambwekar, Murtaza Dhuliawala, Lara J. Martin, Animesh Mehta, Brent Harrison, and Mark O. Riedl. 2019. Controllable neural story plot generation via reward shaping. In *Proceedings of the Twenty-Eighth International Joint Conference on Artificial Intelligence, IJCAI 2019, Macao, China, August 10-16, 2019*, pages 5982–5988. ijcai.org.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, December 4-9, 2017, Long Beach, CA, USA*, pages 5998–6008.

K. Vonnegut. 1981. Palm sunday: An autobiographical collage. In *RosetTaBooks, LLC New York*.

Di Wang, Nebojsa Jojic, Chris Brockett, and Eric Nyberg. 2017. Steering output style and topic in neural response generation. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 2140–2150, Copenhagen, Denmark. Association for Computational Linguistics.

Noah Weber, Leena Shekhar, Heeyoung Kwon, Niranjan Balasubramanian, and Nathanael Chambers. 2020. Generating narrative text in a switching dynamical system. In *Proceedings of the 24th Conference on Computational Natural Language Learning*, pages 520–530, Online. Association for Computational Linguistics.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Remi Louf, Morgan Funtowicz, Joe Davison, Sam Shleifer, Patrick von Platen, Clara Ma, Yacine Jernite, Julien Plu, Canwen Xu, Teven Le Scao, Sylvain Gugger, Mariama Drame, Quentin Lhoest, and Alexander Rush. 2020. Transformers: State-of-the-art natural language processing. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 38–45, Online. Association for Computational Linguistics.

Thomas Wolf, Victor Sanh, Julien Chaumond, and Clement Delangue. 2019. Transfertransfo: A transfer learning approach for neural network based conversational agents. *ArXiv*, abs/1901.08149.

Yuqiang Xie, Yue Hu, Wei Peng, Guanqun Bi, and Luxi Xing. 2022a. Comma: Modeling relationship among motivations, emotions and actions in language-based human activities.

Yuqiang Xie, Yue Hu, Luxi Xing, Yunpeng Li, Wei Peng, and Ping Guo. 2022b. Clseg: Contrastive learning of story ending generation. In *ICASSP 2022 - 2022 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 8057–8061.

Feifei Xu, Xinpeng Wang, Yunpu Ma, Volker Tresp, Yuyi Wang, Shanlin Zhou, and Haizhou Du. 2020a. Controllable multi-character psychology-oriented story generation. In *CIKM '20: The 29th ACM International Conference on Information and Knowledge Management, Virtual Event, Ireland, October 19-23, 2020*, pages 1675–1684. ACM.

Jingjing Xu, Xuancheng Ren, Yi Zhang, Qi Zeng, Xiaoyan Cai, and Xu Sun. 2018. A skeleton-based model for promoting coherence among sentences in narrative story generation. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 4306–4315, Brussels, Belgium. Association for Computational Linguistics.

Peng Xu, Mostofa Patwary, Mohammad Shoeybi, Raul Puri, Pascale Fung, Anima Anandkumar, and Bryan Catanzaro. 2020b. MEGATRON-CNTRL: Controllable story generation with external knowledge using large-scale language models. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 2831–2845, Online. Association for Computational Linguistics.

Lili Yao, Nanyun Peng, R. Weischedel, Kevin Knight, Dongyan Zhao, and Rui Yan. 2019. Plan-and-write: Towards better automatic storytelling. In *AAAI*.

Duzhen Zhang, Zhen Yang, Fandong Meng, Xiuyi Chen, and Jie Zhou. 2022. TSAM: A two-stream attention model for causal emotion entailment. *CoRR*, abs/2203.00819.

Hao Zhou, Minlie Huang, Tianyang Zhang, Xiaoyan Zhu, and Bing Liu. 2018. Emotional chatting machine: Emotional conversation generation with internal and external memory. In *Proceedings of the Thirty-Second AAAI Conference on Artificial Intelligence, (AAAI-18)*, pages 730–739. AAAI Press.

Xianda Zhou and William Yang Wang. 2018. MojiTalk: Generating emotional responses at scale. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1128–1137, Melbourne, Australia. Association for Computational Linguistics.

Few-Shot Table-to-Text Generation with Prefix-Controlled Generator

Yutao Luo , Menghua Lu , Gongshen Liu*, Shilin Wang
Shanghai Jiao Tong University
{luoyt1996, 610228633, lgshen, wsl}@sjtu.edu.cn

Abstract

Neural table-to-text generation approaches are data-hungry, limiting their adaptation for low-resource real-world applications. Previous works mostly resort to Pre-trained Language Models (PLMs) to generate fluent summaries of a table. However, they often contain hallucinated contents due to the uncontrolled nature of PLMs. Moreover, the topological differences between tables and sequences are rarely studied. Last but not least, fine-tuning on PLMs with a handful of instances may lead to overfitting and catastrophic forgetting. To alleviate these problems, we propose a prompt-based approach, Prefix-Controlled Generator (i.e., PCG), for few-shot table-to-text generation. We prepend a task-specific prefix for a PLM to make the table structure better fit the pre-trained input. In addition, we generate an input-specific prefix to control the factual contents and word order of the generated text. Both automatic and human evaluations on different domains (humans, books and songs) of the Wikibio dataset show substantial improvements over baseline approaches.

1 Introduction

Table-to-text generation is a significant branch of Natural Language Generation (NLG), aiming at generating descriptive text given an input table. There is a wide range of application scenarios for automatic table-to-text generation, such as sport news generation (Wiseman et al., 2017), story generation (Liu et al., 2020), weather forecasting report (Liang et al., 2009), and open-domain question answering (Chen et al., 2021).

Recent years have witnessed the great development of pre-trained language models (PLMs) (Devlin et al., 2019; Radford et al., 2019; Lewis et al., 2020), which achieve state-of-the-art performance on many text generation tasks, such as neural machine translation, document summarization, etc. Unlike these tasks, table-to-text generation faces the lack of labeled data. Due to the development of data science, many statistical tables are generated in our daily life, but they scarcely have corresponding natural language descriptions, which limits the real-world application of data-hungry pre-trained models. To address this problem, researchers investigate workarounds in the few-shot setting. Chen et al. (2020), Gong et al. (2020) and Su et al. (2021a) leverage pre-trained linguistic knowledge of neural language models, then fine-tune them in target domains with limited labeled data. This "*pre-train and fine-tune*" paradigm performs relatively well in generating descriptive text from tables. Recently, another paradigm named "*pre-train and prompt*" has been proposed in order to adapt PLMs to downstream tasks without fine-tuning, which is more suitable for few-shot and zero-shot scenarios. Li and Liang (2021) prepends prompt tokens to adapt table-to-text generation to sequential generation task, and freezes PLMs' weights to fully leverage their prior knowledge learned in the pre-training stage.

Despite their contributions, however, two main challenges for table-to-text generation remain to be explored, namely (1) the **topological structure difference** between tables and sequential inputs and (2) model's ability to **select and rearrange factual content** from tables.

In order to address the aforementioned problems, we follow the "*pre-train and prompt*" paradigm and propose Prefix-Controlled Generator (i.e., PCG), an end-to-end generation framework along with two kinds of prefix tokens. Specifically, we prepend a task-specific (i.e., static) prompt and an input-specific (i.e., dynamic) prompt to the tabular input. The task-specific prompt aims to bridge the topological structure gap between a table and a word sequence, while the input-specific prompt aims to plan the factual content and the slot order of a table. Both prefixes are optimized during

*Corresponding author.

Proceedings of the 29th International Conference on Computational Linguistics, pages 6493–6504
October 12–17, 2022.

Attribute(K)	Value(V)
name	edinho júnior
fullname	edon júnior viegas amaral
birth_date	7 march 1994
birth_place	faro , Portugal
currentclub	farense
clubnumber	21
position	forward

Gold: edon júnior viegas amaral , known as edinho júnior (born 7 march 1994) is a portuguese footballer who plays as a forward for farense .
Prefix-Tuning: edon júnior (born 7 march 1994 in faro , portugal) is a portuguese football midfielder .
BART: edino júnior (born 7 march 1994) is a portuguese footballer who plays for farense as a forward .
Switch+PLM: edon júnior viegas amaral or edinho júnior (born 7 march 1994 in faro) is a portugal football player and he plays for farense .
Ours: edinho júnior viegas amaral (born 7 march 1994) is a portuguese footballer who plays as a forward for farense .

Figure 1: An example from *Wikibio Humans* domain and the generated descriptions via various approaches. Words in blue, red, orange and yellow indicate factual contents, wrong generation, hallucinated contents and inferred contents respectively. BART represents BART-large (Lewis et al., 2020). Switch+PLM represents Chen et al. (2020)'s approach with BART-large.

the training phase with the PLM remaining frozen, making our approach parameter-efficient – we only save one copy of the PLM while training in three different domains.

We basically follow the idea of prefix-tuning (Li and Liang, 2021) to design the task-specific prefix, except for some modifications. Firstly, due to the importance of a proper initialization of prefix tokens, we use task-relevant words (e.g., "*Summarize the following table:*", or "*TL;DR:*") as the initial prefix to better linearize the tabular input. Secondly, He et al. (2022) proves that the length of prefix tokens and the design of adding additional parameters solely on the attention module are two bottlenecks of prefix-tuning. Inspired by their work, we add Scaled Parallel Adapters (He et al., 2022) in parallel with both the attention layer and the feed-forward layer to improve the bottleneck of prefix-tuning.

For the input-specific prefix, we expect it can hint to the model which key-value pairs should be selected and in what order they should be arranged. Therefore, we propose a content planner to select the keys that appear in the gold summary and sort them according to the order of occurrence in the summary. For example, given a table in Fig. 1, we expect the content planner to generate a word sequence "*fullname name birth_date birth_place position currentclub*" that indicates all the keys and their occurrence order whose values appear in the gold summary. The word sequence will be used as hard prompts to feed into the PLM.

We evaluate our model on multi-domain table-to-text dataset (Chen et al., 2020). We show that our model outperforms previous state-of-the-art methods on both automatic evaluation metrics (§5.4) and human evaluation metrics (§5.5). We also conduct ablation studies to verify the effectiveness of the two kinds of prefixes (§5.6).

In a nutshell, our contributions are as follows:

1. We propose a Prompt-Controlled Generator that attends to the task-specific prefix to bridge the topological structure gap between tables and sequences, and the input-specific prefix to select factual contents from the tables and reorder them.

2. We propose a simple yet effective content planner to generate the input-specific prefix as the hard prompt of the PLM.

3. We conduct experiments on different domains of the Wikibio dataset to prove the effectiveness of our approach.

2 Related Work

2.1 Few-shot Table-to-text generation

Table-to-text generation has aroused much interest in recent years. Most of the existing studies resort to the end-to-end framework to generate fluent and faithful natural language descriptions given tables. Ma et al. (2019) firstly studied table-to-text generation under the low-resource constraint, and separated the generation process into two stages – key fact prediction and surface realization. With the advances of PLMs, many researchers fine-tune pre-trained GPT-2 (Radford et al., 2019) or BART (Lewis et al., 2020) to augment the scarce training data, which can better assist few-shot table-to-text generation. Chen et al. (2020) used copy mechanism (See et al., 2017) to improve the fidelity of sentences generated by GPT-2 by choosing to copy words from tabular input. Gong et al. (2020) adopted a unified GPT-2 model for table structure

reconstruction and generation. Zhao et al. (2021) proposed a token-level attention and a slot-level attention to exploit natural linguistic and table structural information. All these works utilized tabular input for free text generation, neglecting the importance of content planning for text fidelity. Su et al. (2021a) introduced an information retrieval (IR) system to select prototype sentences similar to the gold summary from large unlabeled parallel corpus, then use them as the auxiliary content plan for tabular input to generate natural language description. However, the IR system might see all gold summaries in the Wikipedia corpus, which violates the true few-shot setting. Different from the above studies, we focus on how to select factual contents via content planning, introducing a slot-aligned content planner.

2.2 Prompt Tuning for Generation

Prompt tuning is a nascent approach for natural language generation (NLG), first proposed by GPT-3 (Brown et al., 2020), introducing in-context learning for few-shot domain adaptation. Prefix-tuning (Li and Liang, 2021) prepended a sequence of continuous vectors to all examples of the downstream tasks. These vectors, which are adjusted as additional key-value pairs, steer the frozen PLMs by augmenting the left context at every Transformer layer. He et al. (2022) classified prefix-tuning as a parameter-efficient tuning approach similar to adapter (Houlsby et al., 2019) and made improvements on its bottlenecks. Clive et al. (2021) extended prefix-tuning to the input-specific prefix (e.g., topic of the datapoint, target output length) to have a finer-grained control for downstream generation tasks. Different from their work, we use the input-specific prompt not to guide the generated text in a certain style, but to improve the fidelity of generated text and the correctness of word order via content planning.

2.3 Controllable Text Generation

Controllable text generation (CTG) is a supplementary field for prompt-based generation, aiming at incorporating guidance signals into generative models. Control signals include text style (Keskar et al., 2019), grammar (Lyu et al., 2021), length (Kikuchi et al., 2016), etc. Recent CTG approaches involve generative adversarial networks (Yu et al., 2017), refactoring a PLM (Chan et al., 2021), fine-tuning adapted modules (Zeldes et al., 2020), prompt learning (Yu et al., 2021) and diffusion model (Li et al., 2022). However, these approaches requires large amount of training data, which does not match our few-shot setting. For table-to-text generation, Su et al. (2021b) proposed a content planner to assist data-to-text generation, which inspired us to pre-plan the order and occurrence of the tabular input for improving the controllability of the generated text.

3 Problem Definition

Given a table T with n key-value pairs $\{\mathbf{K}_i : \mathbf{V}_i\}_{i=1}^{n}$, where $\mathbf{K}_i = \{k_1^i, k_2^i, ..., k_m^i\}$ and $\mathbf{V}_i = \{v_1^i, v_2^i, ..., v_{m'}^i\}$ refer to the key and the value of the i-th table slot respectively, we aim to generate a fluent and faithful natural language description of the table in a low-resource constraint. Note that $\mathbf{K}_i$ and $\mathbf{V}_i$ represent sequences of m and m' words respectively.

4 Methodology

We first provide intuition of using a task-specific prefix and an input-specific prefix for few-shot table-to-text generation (§4.1). Fig. 2 depicts the overall architecture of our method. As shown in the figure, given the input table, the content planner selects the factual contents and reorders them to form a dynamic prompt (§4.2). After that, a static prompt is designed and fed to the generative PLM along with the dynamic prompt (§4.3).

4.1 Intuition

The intuition of introducing prompt to few-shot table-to-text generation is that prompt-tuning effectively solves the catastrophic forgetting problem. Since table-to-text generation requires the language understanding ability of the table content, we hope to fine-tune downstream tasks while retaining the prior knowledge of PLMs, which is exactly what prompt-tuning does. Unlike model fine-tuning, which might be over-parameterized, prompt tuning only adjusts a few parameters and is less prone to over-fitting.

Observing Gong et al. (2020)'s experimental results, we find that table format transformation plays a vital role in improving the generation process, so we focus on bridging the topological structure gap between tables and sequential inputs. The first thought is that we can flatten a table into a word sequence using template (Gong et al., 2020). For example, given a table shown in Fig. 1, we serialize the key-value pair {*name: edinho júnior*}

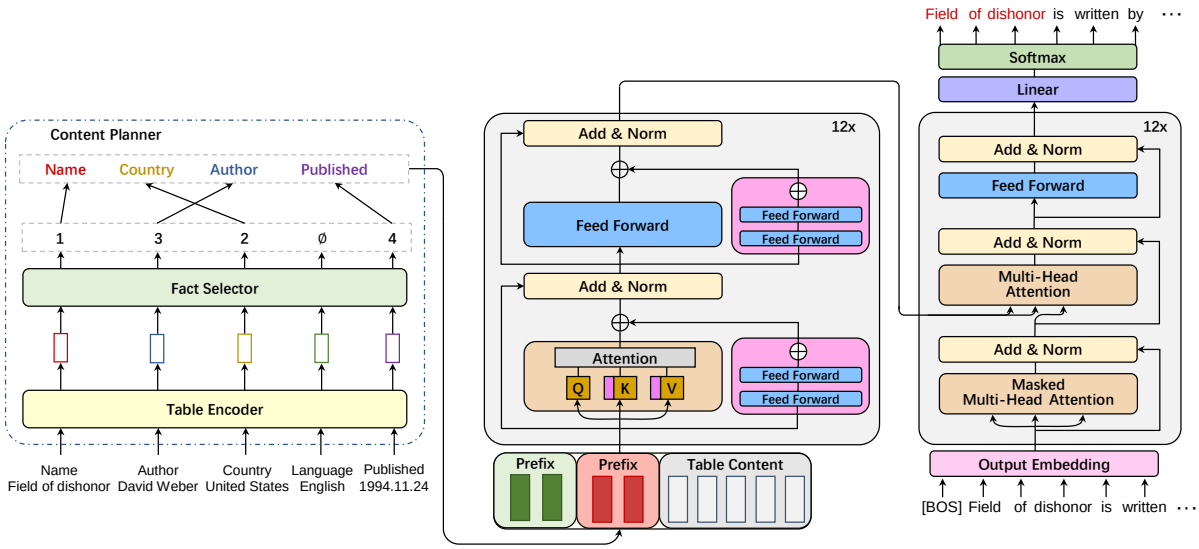

Figure 2: The overall architecture of the proposed method, which can be divided into Content Planner and Prompt-Controlled Generator. Tokens in red and yellow indicate these words are consistent with the value corresponding to the key "*name*" and "*country*" respectively.

as "*name is edinho júnior;*", then concatenate all key-value pairs to form a sentence, that is, "*name is edinho júnior; fullname is edon júnior viegas amaral; birth_date is 7 march, 1994; ...*". Considering that the template-generated sentence is still somewhat different from the pre-training input, we want to find a way that adapts it to a natural sentence. Intuitively, we can add some prompt tokens like "*summarize the following table:*" to make the template-generated sentence an incidental component of the whole input. In this way, "*summarize the following table*" becomes the major component of the sentence, which is more similar to the sequential form of the pre-training input. In addition, many language models now have prefix LM pre-training tasks, which makes our sentences more consistent with the pre-training input.

We also seek to minimize the generated hallucinated content. Considering that some table slots are redundant, we intuitively want to hint the model what are the factual contents. To be consistent with the above table linearization approach, we follow the idea of controllable generation, providing a hard prompt as the guidance signal for each example to control both the table content to be selected and the word order.

4.2 Content Planner

Content Planner aims to generate input-specific prompts that guide the generation process in terms of factual contents and words order, which is shown in Fig. 2 (left). Content Planner contains two modules, namely Table Encoder and Fact Selector. Since we study table-to-text generation under a strict few-shot constraint, we strive for simplicity of Content Planner. Therefore, we use a bi-directional LSTM and a linear-chain Conditional Random Fields (CRF) (Lafferty et al., 2001) for Table Encoder and Fact Selector respectively, which are learned given a handful of training instances.

Table Encoder takes all key-value pairs $\{\mathbf{K}_i : \mathbf{V}_i\}_{i=1}^{n}$ from table T as the input, and produces a hidden representation $\mathbf{h}_i \in \mathbb{R}^{d_e}$ for each table slot, where d_e is the hidden dimension. Specifically, for each table slot that contains a key-value pair $\{\mathbf{K}_i : \mathbf{V}_i\}$, we embed $\mathbf{K}_i$ and $\mathbf{V}_i$ by:

$$\mathbf{e}_i = \lambda \frac{1}{m} \sum_{j=1}^{m} \mathbf{E}(k_j^i) + (1-\lambda) \frac{1}{m'} \sum_{j=1}^{m'} \mathbf{E}(v_j^i), \quad (1)$$

where $\mathbf{e}_i$ denotes the embedding of the i-th slot, $\mathbf{E}$ is the embedding lookup table and λ is a hyper-parameter that controls the ratio of key embedding and value embedding in $\mathbf{e}_i$. We use pre-trained Roberta (Liu et al., 2019) embedding to initialize $\mathbf{E}$. After that, we feed all embeddings $\{\mathbf{e}_1, \mathbf{e}_2, ..., \mathbf{e}_n\}$ to the BiLSTM encoder to obtain $\{\overrightarrow{\mathbf{h}_1}, \overrightarrow{\mathbf{h}_2}, ..., \overrightarrow{\mathbf{h}_n}\}$ and $\{\overleftarrow{\mathbf{h}_1}, \overleftarrow{\mathbf{h}_2}, ..., \overleftarrow{\mathbf{h}_n}\}$ in the left-to-right and right-to-left directions respectively. The calculation of each direction uses a distinct set of parameters. The final hidden states $\mathbf{h}_i$ can be obtained by:

$$\mathbf{h}_i = [\overrightarrow{\mathbf{h}_i}; \overleftarrow{\mathbf{h}_i}]. \quad (2)$$

Fact Selector selects key-value pairs that occur in the ground-true table summary, and rearranges them according to the order of occurrence in the summary. In practice, we use a standard CRF layer with a feed-forward layer as our Fact Selector to compute the global optimal sequence. On top of the hidden states $\mathbf{H}_c = \{\mathbf{h}_1, \mathbf{h}_2, ..., \mathbf{h}_n\}$, the probability distributions of the label sequence $\mathbf{y} = \{l_1, l_2, ..., l_n\}$ is computed by:

$$P(\mathbf{y}|\mathbf{H}_c) = \frac{exp(\sum_i (\mathbf{W}_{CRF}^{l_i}\mathbf{h}_i + \mathbf{M}_{l_{i-1},l_i}))}{\sum_{\mathbf{y}'} exp(\sum_i (\mathbf{W}_{CRF}^{l'_i}\mathbf{h}_i + \mathbf{M}_{l'_{i-1},l'_i}))}. \quad (3)$$

Here $\mathbf{y}'$ represents an arbitrary label sequence, $\mathbf{W}_{CRF}^{l_i}$ denotes the parameters specific to l_i, and $\mathbf{M}_{l_{i-1},l_i}$ denotes the transition score from l_{i-1} to l_i. The learning objective is defined as:

$$\mathcal{L}_{CRF} = -logP(\mathbf{y}|\mathbf{H}_c). \quad (4)$$

Content Planner is trained independently with Prompt-Controlled Generator. The labeled key-value pair order is extracted from the ground-true summary by finding keys[1] and sorting them according to their positions. During inference, we use first-order Viterbi algorithm to decode the best label sequence $\widetilde{\mathbf{y}} = argmax_{\mathbf{y}'} P(\mathbf{y}'|\mathbf{H}_c)$. Take Fig. 2 as an example, Content Planner generates a label sequence "1,3,2,∅,4". The first label "1" denotes "*Name*" should appear in the front of the content plan, while the fourth label "∅" denotes "*Language*" does not occur in the gold summary. According to the label sequence, we rearrange all keys to form a content plan $\mathbf{c}$, which in Fig. 2 is "*Name Country Author Published*".

4.3 Prompt-Controlled Generator

Prompt-Controlled Generator aims to generate fluent and faithful descriptions given the tabular input and the content plan. Our approach is model-agnostic, thus the generator could be any pre-trained generation model. Here we use BART-large (Lewis et al., 2020) as the basic generator for their best overall performances, and propose two kinds of prefixes that are prepended to the input of BART encoder, namely task-specific prompt $\mathbf{p}_s$ and input-specific prompt $\mathbf{c}$. The latter (i.e., content plan) serves as the guiding signal of Prompt-Controlled Generator.

[1]Some keys such as "nationality" are fuzzy-matched.

The task-specific prompt is designed to bridge the topological structure gap between tables and sequences. A first thought is that we can linearize the table via template (see §4.1), then prepend discrete prompt words "*summarize the following table:*" to the template-generated sequence to make the tabular input more consistent with the pre-training input. Nevertheless, discrete optimization needs enormous computing power and human crafts. Instead of using discrete prompt, we follow prefix-tuning (Li and Liang, 2021) to optimize a sequence of continuous prefix tokens while keeping the PLM frozen. However, the prefix length and acting on the attention layer bound the presentation ability of the prefix (He et al., 2022). Considering these bottlenecks, we additionally parallel two Scaled Parallel Adapters to the attention layer and the feed-forward layer respectively, then perform scaled addition for these Adapters.

Next, we will introduce our modifications to BART encoder. Let us denote the template-generated sentence as $\mathbf{s} = \{s_1, s_2, ..., s_L\}$ and content plan as $\mathbf{c} = \{c_1, c_2, ..., c_{L_c}\}$, where L and L_c are the lengths. The prefix length is denoted by L_p. We concatenate the content plan and the template-generated sentence (denoted by $[\mathbf{c}: \mathbf{s}]$ where $[\cdot : \cdot]$ is the concatenation operator) to feed into BART encoder. In the multi-head self-attention layer, we first compute the queries $\mathbf{Q} \in \mathbb{R}^{(L+L_c)\times d}$, keys $\mathbf{K} \in \mathbb{R}^{(L+L_c)\times d}$ and values $\mathbf{V} \in \mathbb{R}^{(L+L_c)\times d}$ via Equation (5):

$$\mathbf{Q} = \mathbf{x}\mathbf{W}_q, \mathbf{K} = \mathbf{x}\mathbf{W}_k, \mathbf{V} = \mathbf{x}\mathbf{W}_v, \quad (5)$$

where d denotes the hidden dimension of BART, $\mathbf{W}_q$, $\mathbf{W}_k$ and $\mathbf{W}_v$ are trainable parameters. $\mathbf{x}$ denotes $\mathbf{E_b}([\mathbf{c}: \mathbf{s}])$ when the first layer is being computed, the output of the previous BART layer otherwise. $\mathbf{E_b}$ denotes the embedding lookup table of BART. Then the attention score is computed via Equation (6):

$$head = [head^1 : head^2 : ... : head^{N_h}], \quad (6)$$

where N_h denotes the number of heads. $head^i$ is computed via Equation (7):

$$\begin{aligned} head^i &= Attn(\mathbf{Q}^i, \mathbf{K}^i, \mathbf{V}^i) \\ &= softmax(\frac{\mathbf{Q}^i[\mathbf{P}_k^i : \mathbf{K}^i]^T}{\sqrt{d_k}})[\mathbf{P}_v^i : \mathbf{V}^i], \end{aligned} \quad (7)$$

where $d_k = \frac{d}{N_h}$ denotes the hidden dimension of

Domain	Humans				Books				Songs			
Training set size	50	100	200	500	50	100	200	500	50	100	200	500
Switch+GPT-2(R)	25.7	29.5	36.1	41.7	34.3	36.2	37.9	40.3	36.1	37.2	39.4	42.2
TableGPT(R)	29.8	34.5	40.6	45.6	35.1	**37.3**	38.5	41.6	36.7	37.8	39.3	42.3
Bart-large	37.6	39.3	41.2	44.3	34.2	37.1	**39.8**	42.9	37.7	38.9	40.1	43.9
AMG(R)	-	-	-	49.0	-	-	-	43.9	-	-	-	**45.1**
Hard-prompt+GPT-2	22.8	28.1	29.7	30.8	25.8	27.9	28.8	32.1	26.6	30.0	30.1	32.1
Prefix-Tuning+GPT-2	25.6	30.3	33.4	37.3	34.9	36.2	36.3	37.3	32.5	33.0	35.1	36.1
Prefix-Tuning+T5	34.5	39.9	41.6	44.1	35.5	**37.3**	39.6	41.2	37.5	38.5	40.0	41.1
Switch+BART(PT)	36.8	41.8	44.0	48.1	33.6	35.0	38.3	43.4	**40.9**	**41.7**	42.1	43.2
Ours	**39.9**	**43.3**	**45.8**	**49.4**	**36.6**	36.9	39.0	**45.6**	38.0	**41.7**	**42.5**	44.5

Table 1: BLEU results on three domains of the Wikibio test set. Each **(R)** is reported by the related paper.

Domain	Humans				Books				Songs			
Training set size	50	100	200	500	50	100	200	500	50	100	200	500
Switch+GPT-2(R)	30.6	34.6	40.5	45.6	42.7	42.8	43.4	44.9	40.2	41.7	44.0	44.8
Bart-large(R)	37.8	41.4	47.4	45.5	41.7	43.4	43.7	48.1	41.7	42.4	44.1	46.0
AMG(R)	43.6	47.7	50.1	**51.9**	43.4	46.0	**47.5**	48.6	42.0	43.3	45.9	**46.9**
Prefix-Tuning+GPT-2	32.7	35.9	36.6	38.7	29.8	31.8	31.7	32.7	31.7	33.3	32.3	31.5
Prefix-Tuning+T5	39.3	40.6	41.8	42.1	32.8	34.8	36.0	36.8	34.4	36.1	36.0	34.6
Switch+BART(PT)	35.2	41.7	45.1	50.5	33.0	37.2	41.2	46.4	36.7	39.4	42.0	45.9
Ours	**46.7**	**48.3**	**50.4**	51.8	**46.3**	**46.2**	**47.5**	**49.3**	**44.8**	**45.7**	**46.9**	46.0

Table 2: PARENT-F results on three domains of the Wikibio test set. All **(R)** are reported by Zhao et al. (2021).

each head, and $\mathbf{P}_k \in \mathbb{R}^{L_p \times d}, \mathbf{P}_v \in \mathbb{R}^{L_p \times d}$ denote two sets of prefix vectors. $\mathbf{Q}^i \in \mathbb{R}^{(L+L_c)\times d_k}$, $\mathbf{K}^i \in \mathbb{R}^{(L+L_c)\times d_k}$, $\mathbf{V}^i \in \mathbb{R}^{(L+L_c)\times d_k}$, $\mathbf{P}_k^i \in \mathbb{R}^{(L_p)\times d_k}$ and $\mathbf{P}_v^i \in \mathbb{R}^{(L_p)\times d_k}$ denote a block of $\mathbf{Q}$, $\mathbf{K}$, $\mathbf{V}$, $\mathbf{P}_k$, and $\mathbf{P}_v$ respectively.

In parallel with the multi-head self-attention layer, a Scaled Parallel Adapter is added:

$$head' = \mathbf{x} + s \cdot ReLU(\mathbf{x}\mathbf{W}_{down})\mathbf{W}_{up}, \quad (8)$$

$$\mathbf{H}_{attn} = head + head', \quad (9)$$

where $\mathbf{W}_{down} \in \mathbb{R}^{d\times r}$ and $\mathbf{W}_{up} \in \mathbb{R}^{r\times d}$ are down-projection and up-projection, r is the bottleneck dimension and $\mathbf{x}$ denotes the same vector as in Equation (5). $s \geq 1$ is a trainable scaling hyper-parameter. We use $\mathbf{H}_{attn}$ to replace the original attention output to conduct residual connection and layer normalization. Similarly, we insert another Scaled Parallel Adapter in parallel with the Feed Forward layer to enhance its representation:

$$\mathbf{o} = ReLU(\mathbf{x}\mathbf{W}_1 + \mathbf{b}_1)\mathbf{W}_2 + \mathbf{b}_2 + s \cdot ReLU(\mathbf{x}\mathbf{W'}_{down})\mathbf{W'}_{up}, \quad (10)$$

where $\mathbf{x}$ and $\mathbf{o}$ denote the input and output of the Feed Forward layer respectively. $\mathbf{W}_1$, $\mathbf{b}_1$, $\mathbf{W}_2$, $\mathbf{b}_2$, $\mathbf{W'}_{down}$, $\mathbf{W'}_{up}$ are trainable parameters.

We conduct residual connection and layer normalization over $\mathbf{o}$ to get the hidden states of BART encoder $\mathbf{H}_{enc}$, then feed $\mathbf{H}_{enc}$ along with decoder input to a normal BART decoder for sentence generation. The decoder input is the right-shifted gold summary in the training phase, and a simple "[BOS]" in the inference phase to generate tokens autoregressively. Given the gold summary $\mathbf{g}$, the learning objective is the cross-entropy loss, defined as:

$$\mathcal{L}_{LM} = -\sum_{i=1}^{|\mathbf{g}|} logP_{dec}(\mathbf{g}_i|\mathbf{g}_{<i}; \mathbf{H}_{enc}). \quad (11)$$

5 Experiments

5.1 Datasets and Hyper-Parameters

Following Chen et al. (2020), we evaluate our method on three different domains (i.e., *Humans*, *Books* and *Songs*) of the Wikibio dataset, denoted

as Wiki-Humans, Wiki-Songs and Wiki-Books respectively. For all three domains, we conduct experiments in few-shot settings by varying the training set size to 50, 100, 200 and 500. The validation size is set to 1000, and the remaining instances are used for testing, which counts 13587, 5252 and 11879 for humans, books and songs respectively.

We use BART-large as our basic generator using transformers library (Wolf et al., 2020), which shares 12 layers and 16 heads for both encoder and decoder. We set the hidden and the embedding dimension of Content Planner to 768 (Roberta-base embedding dimension), and the key-value ensemble ratio λ is set to 0.7. The learning rates of Content Planner and Prefix-Controlled Generator are set to 2e-4 and 1e-5 respectively, both optimized by AdamW (Loshchilov and Hutter, 2017). We train our PCG for 200 epochs, with a batch size of 10 on one NVIDIA GeForce RTX 3090 GPU. Prefix length L_p is set to 30, and the bottleneck size of Scaled Parallel Adapter r is set to 512.

5.2 Baseline Models

We compare previous state-of-the-art few-shot table-to-text generation approaches, serving as baseline models:

(i) **Switch+PLM** (Chen et al., 2020): The first work that introduces PLMs to the few-shot NLG task. They propose a switch policy to choose whether to copy words from the table or to generate from GPT-2. We also implement a variant using BART-large to replace GPT-2 and tuning the BART-large model with our task-specific prompt (denoted as **Switch+BART(PT)**).

(ii) **TableGPT** (Gong et al., 2020): A further study based on Switch+PLM that leverages GPT-2's prior knowledge, while enhancing generation fidelity with two auxiliary tasks.

(iii) **AMG** (Zhao et al., 2021): A pre-train and fine-tune approach with a multi-grain attention to both tokens and slots, and introduces memory mechanism to back-track the allocation of table slots.

(iv) **BART-large** (Lewis et al., 2020): A powerful PLM for conditional generation, which is proved effective in the few-shot scenario (Zhao et al., 2021). We fine-tune it on our few-shot datasets to report its performance.

Model	50		100		200		500	
	acc.	BLEU	acc.	BLEU	acc.	BLEU	acc.	BLEU
Roberta-base	0.32	14.6	0.33	14.2	0.39	21.3	0.56	32.5
ContentPlanner	0.53	30.4	0.56	32.6	0.59	35.4	0.64	37.5

Table 3: Results on Content Planner. acc. and BLEU denote test accuracy and BLEU-2 respectively.

(v) **Hard-prompt+GPT-2**: Our earlier attempt on the few-shot table-to-text generation task, which uses actual tokens such as "Summarize the following table:" as the prompt words, then feed the transformed tabular input into GPT-2 to fine-tune on few-shot table-to-text generation task.

(vi) **Prefix-Tuning** (Li and Liang, 2021): A novel prompt-based approach that prepends a continuous prefix and freezes the PLMs to retain their prior knowledge. We follow Ding et al. (2021)'s implementation, using GPT-2 and T5 (Raffel et al., 2019) as the base model.

Among above baseline approaches, **Prefix-Tuning** and **Switch+BART(PT)**) follow a "pre-train and prompt-tuning" paradigm (keep LM's parameters frozen), while **Hard-prompt+GPT-2** uses prompt for model fine-tuning. All the other baselines are following the standard "pre-train and fine-tune" paradigm.

5.3 Results of Content Planner

We first report the experimental results of Content Planner. Intuitively, we use accuracy to evaluate the percentage of words that are both correct and in the right position. Following Zhao et al. (2020), we also use BLEU-2 to evaluate the correctness of the words occurring in the content plan. We train Content Planner in 200 epochs with 50/100/200/500 training instances respectively and compare it with *RobertaforSequenceClassification* from transformers library. The results are shown in Table 3. We show that in the few-shot setting, Bi-LSTM+CRF performs better than fine-tuning Roberta in both word co-occurrence and positional correctness.

5.4 Automatic Evaluation

We conduct automatic evaluations on various domains of the Wikibio dataset to prove the effectiveness of our method. We select two kinds of evaluation metrics – **BLEU** (Papineni et al., 2002) for evaluating overlap between the generated sentence and the gold description, and **PARENT** (Dhingra

et al., 2019) for evaluating both the matching between the generated sentence and the reference and the fidelity of the generated sentence to the original table. Here we use F1 score of PARENT, denoted as PARENT-F.

Regarding the overlapping-based metrics BLEU, we show that our method has the best overall performance compared with all other baselines. Specifically, our approach improves 1.8%/0.4% BLEU score on Wiki-Humans/Wiki-Songs compared with the second best model with 200 training instances. On Wiki-Books, we improve 1.7% BLEU score than AMG with 500 training instances. The results show that our method can produce fluent descriptions. We attribute this to the task-specific prefix that better linearizes the tabular input by comparing fine-tuning BART-large (see §5.6).

Regarding the fidelity-based metrics PARENT, our method has better performances over AMG especially in extremely low-resource scenarios, while outperforming other baseline models. Our method performs 1.6% PARENT-F better than AMG on average in 9 terms and 0.5% PARENT-F worse on average in 2 terms. Reviewing their approach, AMG uses the Wikibio dataset, which is very similar to the few-shot datasets, for task adaptive pre-training. In a real-world low-resource scenario, however, it's less likely to obtain a large unlabeled corpus related to the target domain. Moreover, our approach is parameter-efficient and storage-saving. Therefore, we provide a more lightweight alternative with better generation fidelity than AMG.

We also implement a variant of Chen et al. (2020)'s work with some modifications. We replace the GPT-2 with BART-large, and use prompt-tuning instead of fine-tuning to generate sentences. Therefore, the encoder in **Switch+BART** is consistent with our BART encoder in Fig. 2. **Switch+BART(PT)** achieves the second best performance in text fluency evaluation, obtaining the highest BLEU score on 2 terms. However, it's bad at keeping faithful to the original table when the training set size is small, which is contrary to the motivation of copy mechanism. A reasonable explanation is that the objectives of Prompt-tuning and fine-tuning are contradictory. Prompt-tuning expects that the continuous prefix can transfer to downstream tasks, while fine-tuning Pointer Generator (See et al., 2017) aims to learn to copy words and to decide whether to copy or to generate. This contradiction makes Pointer Generator unable to get effective training, especially when lacking training instances. We print the selected words when the model switches to "*copy*" state, finding that the words are far from the tokens that should be copied. We also see from Table 2 that changing copy mechanism to the input-specific prefix significantly improves the text fidelity.

5.5 Human Evaluation

We randomly select 100 generated sentences (training set size is set to 500) and corresponding tables and references from the test set, then present them to three voluntary human evaluators. All volunteers are postgraduate students with extensive research experience in document summarization and natural language generation. Inspired by Chen et al. (2020), we assure that each sentence is evaluated according to its (1) *faithfulness to the table and the reference* and (2) *language fluency*. To evaluate the effectiveness of our Content Planner, we also evaluate the generated sentences according to their (3) *words order correctness*. In the first task, all evaluators count the number of facts n_{co} that co-occur in the table slot and the reference[2], and the number of facts n_{hal} that contradict with/ miss from the table (i.e., hallucinated contents). The percentage of factual content is then computed through $f_p = \frac{\sum_{s \in \mathcal{S}} n^s_{co}}{\sum_{s \in \mathcal{S}} (n^s_{co} + n^s_{hal})}$, where $\mathcal{S}$ denotes the select corpus. In the second task, we ask each evaluator to compare sentences in a sentence set (descriptions of an instance generated from various methods), then rank them based on their fluency and grammatical correctness. The ranking then is normalized to 0-1, the smaller the better. Finally, we average the normalized ranking of the 100 sentences to get r_{avg}. In the third task, all volunteers are asked to count the words order correctness. For example, given a ground-true content plan *"Name Published Genre Author"* and hypothesis *"A push and a shove is a 2007 novel by Christopher Kelly."*, volunteers count the correct key pair order in the hypothesis, such as ‘*"Published"* is in front of *"Author"*. *"Genre"* is not in the hypothesis, thus all its key pair order (*"Name Genre"*, *"Published Genre"*, *"Genre Author"*) are wrong. The words order accuracy acc_{wo} is averaged over all generated hypothesis. Human evaluation results are shown in Table 5. We compute the final score via $f_p - r_{avg} + acc_{wo}$ to

[2]Here we do not define co-occurrence as exact-matching or fuzzy-matching, instead we ask volunteers to decide co-occurrence based on human knowledge.

Model	Humans	Books	Songs
PCG	**43.3/48.3**	36.9/**46.2**	**41.7/45.7**
PCG w/o $\mathbf{c}$	43.2/47.2	37.3/44.8	41.5/44.5
PCG w/o c&SPA	41.9/46.0	**37.4**/44.5	39.6/44.3
PCG w/o $\mathbf{p}_s$&c&SPA	39.3/41.4	37.1/43.4	38.9/42.4

Table 4: Ablation study results on two kinds of prefixes. $\mathbf{p}_s$, $\mathbf{c}$ and SPA denote the task-specific prefix, the input-specific prefix and Scaled Parallel Adapter respectively. In each entry, a/b denotes the BLEU/PARENT-F score.

Model	f_p	r_{avg}	acc_{wo}	overall
Switch+GPT-2	0.62	0.58	0.71	0.75
Prefix-tuning+T5	0.73	0.28	0.79	1.24
PCG	**0.75**	**0.20**	**0.84**	**1.39**

Table 5: Human evaluation results.

measure the models' performance, the larger the better.

5.6 Ablation Study

We conduct ablation studies to evaluate the improvement brought by the two kinds of prefixes we proposed. We experiment on all three datasets with the training set size of 100. The automatic results are shown in Table 4. Observing the results, we conclude that both task-specific prefix and input-specific prefix improve the fidelity of the generated sentences, while input-specific prefix contributes little to the text fluency. These conclusions are consistent with our intuitions, given that input-specific prefix aims to improve the faithfulness by planning the content. Through ablation, we show that prepending a continuous prefix to the encoder input performs better than fine-tuning BART in the few-shot scenario. In addition, adding Scaled Parallel Adapters to enhance the representation ability of prompt vectors has also proved to be effective.

6 Conclusion

In this paper, we propose Prompt-Controlled Generator, using two kinds of prompts to address current challenges in few-shot table-to-text generation. The task-specific prefix aims to bridge the topological structure gap between tables and sequences, which is learned via freezing the PLM and tuning the continuous prompt vectors. The input-specific prefix is designed to guide the generation process in terms of factual content and word order. We propose Content Planner to generate the input-specific prefix. Experiments on Wiki-Humans, Wiki-Books and Wiki-Songs datasets prove the effectiveness of our method from the aspects of generation fluency and text fidelity to the table.

Acknowledgements

We thank the anonymous reviewers for their constructive comments. This work was supported by the Joint Funds of the National Natural Science Foundation of China (Grant No. U21B2020). Gongshen Liu is the corresponding author.

References

Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel Ziegler, Jeffrey Wu, Clemens Winter, Chris Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. 2020. Language models are few-shot learners. In *Advances in Neural Information Processing Systems*, volume 33, pages 1877–1901. Curran Associates, Inc.

Alvin Chan, Yew-Soon Ong, Bill Pung, Aston Zhang, and Jie Fu. 2021. Cocon: A self-supervised approach for controlled text generation. In *9th International Conference on Learning Representations, ICLR 2021, Virtual Event, Austria, May 3-7, 2021*. OpenReview.net.

Wenhu Chen, Ming-Wei Chang, Eva Schlinger, William Yang Wang, and William W. Cohen. 2021. Open question answering over tables and text. In *9th International Conference on Learning Representations, ICLR 2021, Virtual Event, Austria, May 3-7, 2021*. OpenReview.net.

Zhiyu Chen, Harini Eavani, Wenhu Chen, Yinyin Liu, and William Yang Wang. 2020. Few-shot NLG with pre-trained language model. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, ACL 2020, Online, July 5-10, 2020*, pages 183–190. Association for Computational Linguistics.

Jordan Clive, Kris Cao, and Marek Rei. 2021. Control prefixes for text generation. *CoRR*, abs/2110.08329.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT 2019, Minneapolis, MN, USA, June 2-7, 2019, Volume 1 (Long and Short Papers)*, pages 4171–4186. Association for Computational Linguistics.

Bhuwan Dhingra, Manaal Faruqui, Ankur P. Parikh, Ming-Wei Chang, Dipanjan Das, and William W. Cohen. 2019. Handling divergent reference texts when evaluating table-to-text generation. In *Proceedings of the 57th Conference of the Association for Computational Linguistics, ACL 2019, Florence, Italy, July 28- August 2, 2019, Volume 1: Long Papers*, pages 4884–4895. Association for Computational Linguistics.

Ning Ding, Shengding Hu, Weilin Zhao, Yulin Chen, Zhiyuan Liu, Hai-Tao Zheng, and Maosong Sun. 2021. Openprompt: An open-source framework for prompt-learning. *arXiv preprint arXiv:2111.01998*.

Heng Gong, Yawei Sun, Xiaocheng Feng, Bing Qin, Wei Bi, Xiaojiang Liu, and Ting Liu. 2020. Tablegpt: Few-shot table-to-text generation with table structure reconstruction and content matching. In *Proceedings of the 28th International Conference on Computational Linguistics, COLING 2020, Barcelona, Spain (Online), December 8-13, 2020*, pages 1978–1988. International Committee on Computational Linguistics.

Junxian He, Chunting Zhou, Xuezhe Ma, Taylor Berg-Kirkpatrick, and Graham Neubig. 2022. Towards a unified view of parameter-efficient transfer learning. In *International Conference on Learning Representations*.

Neil Houlsby, Andrei Giurgiu, Stanislaw Jastrzebski, Bruna Morrone, Quentin de Laroussilhe, Andrea Gesmundo, Mona Attariyan, and Sylvain Gelly. 2019. Parameter-efficient transfer learning for NLP. In *Proceedings of the 36th International Conference on Machine Learning, ICML 2019, 9-15 June 2019, Long Beach, California, USA*, volume 97 of *Proceedings of Machine Learning Research*, pages 2790–2799. PMLR.

Nitish Shirish Keskar, Bryan McCann, Lav Varshney, Caiming Xiong, and Richard Socher. 2019. CTRL - A Conditional Transformer Language Model for Controllable Generation. *arXiv preprint arXiv:1909.05858*.

Yuta Kikuchi, Graham Neubig, Ryohei Sasano, Hiroya Takamura, and Manabu Okumura. 2016. Controlling output length in neural encoder-decoders. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 1328–1338, Austin, Texas. Association for Computational Linguistics.

John D. Lafferty, Andrew McCallum, and Fernando C. N. Pereira. 2001. Conditional random fields: Probabilistic models for segmenting and labeling sequence data. In *Proceedings of the Eighteenth International Conference on Machine Learning (ICML 2001), Williams College, Williamstown, MA, USA, June 28 - July 1, 2001*, pages 282–289. Morgan Kaufmann.

Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Veselin Stoyanov, and Luke Zettlemoyer. 2020. BART: denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, ACL 2020, Online, July 5-10, 2020*, pages 7871–7880. Association for Computational Linguistics.

Xiang Lisa Li and Percy Liang. 2021. Prefix-tuning: Optimizing continuous prompts for generation. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing, ACL/IJCNLP 2021, (Volume 1: Long Papers), Virtual Event, August 1-6, 2021*, pages 4582–4597. Association for Computational Linguistics.

Xiang Lisa Li, John Thickstun, Ishaan Gulrajani, Percy Liang, and Tatsunori B. Hashimoto. 2022. Diffusion-lm improves controllable text generation. *CoRR*, abs/2205.14217.

Percy Liang, Michael I. Jordan, and Dan Klein. 2009. Learning semantic correspondences with less supervision. In *ACL 2009, Proceedings of the 47th Annual Meeting of the Association for Computational Linguistics and the 4th International Joint Conference on Natural Language Processing of the AFNLP, 2-7 August 2009, Singapore*, pages 91–99. The Association for Computer Linguistics.

Danyang Liu, Juntao Li, Meng-Hsuan Yu, Ziming Huang, Gongshen Liu, Dongyan Zhao, and Rui Yan. 2020. A character-centric neural model for automated story generation. *Proceedings of the AAAI Conference on Artificial Intelligence*, 34(02):1725–1732.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Roberta: A robustly optimized BERT pretraining approach. *CoRR*, abs/1907.11692.

Ilya Loshchilov and Frank Hutter. 2017. Decoupled weight decay regularization. *arXiv preprint arXiv:1711.05101*.

Yiwei Lyu, Paul Pu Liang, Hai Pham, Eduard Hovy, Barnabás Póczos, Ruslan Salakhutdinov, and Louis-Philippe Morency. 2021. StylePTB: A compositional benchmark for fine-grained controllable text style transfer. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 2116–2138, Online. Association for Computational Linguistics.

Shuming Ma, Pengcheng Yang, Tianyu Liu, Peng Li, Jie Zhou, and Xu Sun. 2019. Key fact as pivot: A two-stage model for low resource table-to-text generation. In *Proceedings of the 57th Conference of the Association for Computational Linguistics, ACL 2019, Florence, Italy, July 28- August 2, 2019, Volume 1: Long Papers*, pages 2047–2057. Association for Computational Linguistics.

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. Bleu: a method for automatic evaluation of machine translation. In *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics, July 6-12, 2002, Philadelphia, PA, USA*, pages 311–318. ACL.

Alec Radford, Jeff Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. 2019. Language models are unsupervised multitask learners.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J. Liu. 2019. Exploring the limits of transfer learning with a unified text-to-text transformer. *CoRR*, abs/1910.10683.

Abigail See, Peter J. Liu, and Christopher D. Manning. 2017. Get to the point: Summarization with pointer-generator networks. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics, ACL 2017, Vancouver, Canada, July 30 - August 4, Volume 1: Long Papers*, pages 1073–1083. Association for Computational Linguistics.

Yixuan Su, Zaiqiao Meng, Simon Baker, and Nigel Collier. 2021a. Few-shot table-to-text generation with prototype memory. In *Findings of the Association for Computational Linguistics: EMNLP 2021, Virtual Event / Punta Cana, Dominican Republic, 16-20 November, 2021*, pages 910–917. Association for Computational Linguistics.

Yixuan Su, David Vandyke, Sihui Wang, Yimai Fang, and Nigel Collier. 2021b. Plan-then-generate: Controlled data-to-text generation via planning. In *Findings of the Association for Computational Linguistics: EMNLP 2021, Virtual Event / Punta Cana, Dominican Republic, 16-20 November, 2021*, pages 895–909. Association for Computational Linguistics.

Sam Wiseman, Stuart M. Shieber, and Alexander M. Rush. 2017. Challenges in data-to-document generation. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing, EMNLP 2017, Copenhagen, Denmark, September 9-11, 2017*, pages 2253–2263. Association for Computational Linguistics.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Remi Louf, Morgan Funtowicz, Joe Davison, Sam Shleifer, Patrick von Platen, Clara Ma, Yacine Jernite, Julien Plu, Canwen Xu, Teven Le Scao, Sylvain Gugger, Mariama Drame, Quentin Lhoest, and Alexander Rush. 2020. Transformers: State-of-the-art natural language processing. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 38–45, Online. Association for Computational Linguistics.

Dian Yu, Zhou Yu, and Kenji Sagae. 2021. Attribute alignment: Controlling text generation from pre-trained language models. In *Findings of the Association for Computational Linguistics: EMNLP 2021*, pages 2251–2268, Punta Cana, Dominican Republic. Association for Computational Linguistics.

Lantao Yu, Weinan Zhang, Jun Wang, and Yong Yu. 2017. Seqgan: Sequence generative adversarial nets with policy gradient. In *Proceedings of the Thirty-First AAAI Conference on Artificial Intelligence, February 4-9, 2017, San Francisco, California, USA*, pages 2852–2858. AAAI Press.

Yoel Zeldes, Dan Padnos, Or Sharir, and Barak Peleg. 2020. Technical report: Auxiliary tuning and its application to conditional text generation. *CoRR*, abs/2006.16823.

Chao Zhao, Marilyn Walker, and Snigdha Chaturvedi. 2020. Bridging the structural gap between encoding and decoding for data-to-text generation. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 2481–2491, Online. Association for Computational Linguistics.

Wenting Zhao, Ye Liu, Yao Wan, and Philip S. Yu. 2021. Attend, memorize and generate: Towards faithful table-to-text generation in few shots. In *Findings of the Association for Computational Linguistics: EMNLP 2021, Virtual Event / Punta Cana, Dominican Republic, 16-20 November, 2021*, pages 4106–4117. Association for Computational Linguistics.

Table Content	Generated Description
name: cody zeller article_title: cody zeller image: cody zeller iu hoosiers standout in chicago img 5907.jpg position: power forward/center height_ft: 7 height_in: 9 weight_lb: 240 league: nba team: charlotte bobcats/hornets number: 40 nationality: american birth_date: 5 october 1992 birth_place: washington, indiana high_school: washington, indiana college: indiana (2011 - 2013) draft_year: 2013 draft_round: 1 draft_pick: 4 draft_team: charlotte bobcats career_start: 2013 years: 2013 start - present	**Gold**: cody allen zeller (born october 5, 1992) is an american professional basketball player who currently plays for the charlotte hornets of the national basketball association (nba). **Plan**: birth_date nationality draft_team league **Switch+BART**: cody zeller (born 5 october 1992 in washington, indiana) is an american basketball player for the charlotte hornets in the nba as a power forward. **PCG**: cody zeller (born october 5, 1992) is an american professional basketball player who currently plays for the charlotte hornets of the national basketball association (nba).
name: brandon pieters fullname: brandon paul pieters birth_date: 22 april 1976 birth_place: germiston , south africa height: 1.93 0 weight: 106 kg lb st on residence: benoni , south africa yearpro: 1994 tour: sunshine tour prowins: 4 sunwins: 4 article_title: brandon pieters	**Gold**: brandon paul pieters (born 22 april 1976) is a south african professional golfer. **Plan**: full_name birth_date birth_place **Switch+BART**: brandon paul pieters (born 22 april 1976) is a south african cricketer who plays for the central bank of southern africa. **PCG**: brandon paul pieters (born 22 april 1976) is a south african wrestler.
name: david highbaugh smith image: david-highbaugh-smith.jpg alt: a man with dark hair and a mustache wearing a dark coat and white shirt state: kentucky district: 4th predecessor: john w. lewis successor: ben johnson birth_date: 19 december 1854 birth_place: hart county , kentucky death_date: 17 december 1928 death_place: hodgenville , kentucky restingplace: red hill cemetery party: democrat profession: lawyer article_title: david highbaugh smith	**Gold**: david highbaugh smith (december 19, 1854 - december 17, 1928) was a u.s. representative from kentucky. **Plan**: name birth_date death_date state **Switch+BART**: davic highbaugh smith (1854 - 1928) was a u.s. lawyer from kentucky. **PCG**: david highbaugh smith (december 19, 1854 - december 17, 1928) was an american representative from kentucky.

Table Content (1)	name: james gilbert birth_name: cecil james gilbert birth_date: 5 may 1923 birth_place: edinburgh , scotland , united kingdom occupation: tv producer/director television: "the two ronnies" "last of the summer wine" "open all hours" years_active: 1959 – 2003 article_title: james gilbert (producer)
Generated Description	cecil james gilbert (born 5 may 1923) is a scottish television producer and director.
Table Content (2)	name: petr faldyna birth_date: 11 july 1976 birth_place: frýdlant nad ostravicí , czechoslovakia height: 1.86 position: forward currentclub: fk senica years: 2001 2001-2002 2002-2003 2003-2004 2004-2006 2006 2006-2009 2009-2011 clubs: lerk prostějov hfk olomouc fk kunovice sfc opava fk kunovice sk České budějovice fc vysočina jihlava fk senica caps: 0 14 8 33 24 41 15 88 32 goals: 0 1 0 16 1 21 6 41 2 pcupdate: 2011-12-31 article_title: petr faldyna
Generated Description	petr faldyna (born 11 july 1976) is a czech professional footballer who plays for fk senica as a forward.
Table Content (3)	image_size: 225px position: left wing shoots: left height_ft: 5 height_in: 7 weight_lb: 159 status: retired former_teams: modo hockey , skellefteå aik birth_date: 7 october 1975 birth_place: Örnsköldsvik , swe draft: 234th overall draft_year: 1996 draft_team: boston bruins career_start: 1993 career_end: 2012 article_title: anders söderberg
Generated Description	anders söderberg (born october 7, 1975) is an american retired ice hockey player who played in the modo hockey league and the boston bruins of the nhl.

Table 6: Examples of generated results from Wiki-Humans test set. Words in red denotes wrong generation.

A Appendix. Examples of generated sentences

In this section, we further provide generated examples of one baseline and our model. Table 6 upper displays the comparison between **Switch+BART(PT)** and our approach for generation quality. We show that our generation considers more on the content plan, while retaining the linguistic understanding of the table content. For example, in example 1, our method neglects contents that does not occur in the content plan, while the baseline method contains some tabular information that is not suitable to appear in the summary; In example 3, our method generate *"representative"* by understanding the table content such as *"predecessor"*, *"successor"* and *"party"*, while **Switch+BART(PT)** simply copies the profession slot. Table 6 lower gives more examples of our generation. All sentences are generated by the model trained with 100 instances.

It's worth mentioning that our generation quality is bounded by the adequacy of the table information. For example 2 in Table 6 upper, since the table does not provide information about *"golfer"*, it is difficult for the model to infer *"golfer"* from *"sunshine tour"* through few-shot training. To alleviate this problem, we are also investigating open-domain knowledge-graph enhanced table-to-text generation, which might further improve the generation quality.

Text Simplification of College Admissions Instructions: A Professionally Simplified and Verified Corpus

Zachary W. Taylor
College of Education and Human Sciences
University of Southern Mississippi
z.w.taylor@usm.edu

Maximus H. Chu
Department of Computer Science
The University of Texas at Austin
maximuschu@utexas.edu

Junyi Jessy Li
Department of Linguistics
The University of Texas at Austin
jessy@utexas.edu

Abstract

Access to higher education is critical for minority populations and emergent bilingual students. However, the language used by higher education institutions to communicate with prospective students is often too complex; concretely, many institutions in the US publish admissions application instructions far *above* the average reading level of a typical high school graduate, often near the 13th or 14th grade level. This leads to an unnecessary barrier between students and access to higher education.

This work aims to tackle this challenge via text simplification. We present **PSAT** (**P**rofessionally **S**implified **A**dmissions **T**exts), a dataset with 112 admissions instructions randomly selected from higher education institutions across the US. These texts are then professionally simplified, and verified and accepted by subject-matter experts who are full-time employees in admissions offices at various institutions. Additionally, PSAT comes with manual alignments of 1,883 original-simplified sentence pairs. The result is a first-of-its-kind corpus for the evaluation and fine-tuning of text simplification systems in a high-stakes genre distinct from existing simplification resources. PSAT is available at https://doi.org/10.5281/zenodo.7055024.

1 Introduction

Access to the higher education system for minoritized populations, especially low-income students, first-generation in college students, students of Color, and students whose first spoken language is not English, is an important social challenge (Auerbach, 2004; Cook et al., 2012; Flores, 2010; Pérez and McDonough, 2008; Rosa, 2006; Taylor, 2018, 2020a). However, researchers have consistently explained that much of higher education's communication with prospective students is too complex, too lengthy, and requires a wealth of prior knowledge of the higher education system to successfully navigate (Auerbach, 2004; Pérez and McDonough, 2008); concretely many institutions in the US publish admissions application instructions at or above the 14th grade English reading comprehension level, far too high for the average prospective student or adult to read and comprehend (Taylor, 2018, 2019, 2020b). As a result, the verbose, difficult communication places an unnecessary barrier between students and access to higher education (Ardoin, 2013; Goff et al., 2004; Hartman, 1997; Kanno, 2018; Taylor, 2017).

With modern conditional text generation models, one plausible way to lower such barriers is automatic text simplification (Siddharthan, 2014; Alva-Manchego et al., 2020), i.e., simplify text such that it is more readable and accessible, while adhering to the texts' original semantic content. Text simplification is known to benefit a range of readers, including children (Javourey-Drevet et al., 2022) and L2 learners of English (Yano et al., 1994). Yet, no studies have explored how to simplify this information, without losing important details, to render the admissions process more accessible for prospective students and their families. Crucially, existing work in text simplification largely rely on two established datasets in the News and Wiki domains (Woodsend and Lapata, 2011; Coster and Kauchak, 2011; Xu et al., 2015; Zhang and Lapata, 2017); unlike these two domains, college admission texts certainly falls under the category of being more specialized and difficult to generalize, where both the jargon and concepts encapsulated within these texts are not ones that one encounters every day. Consequently we lack understanding both in terms of the nature of

Proceedings of the 29th International Conference on Computational Linguistics, pages 6505–6515
October 12–17, 2022.

(Original) All conditionally accepted applicants must consent to, submit to and successfully complete a criminal background check through Certiphi Screening, Inc. Failure to do so will constitute failure to meet the prematriculation requirements established by SUNY Optometry and will result in the withdrawal of a conditionally accepted offer.
(Simplified) If you are conditionally accepted, you must consent to, submit to, and complete a criminal background check through Certiphi Screening, Inc. If you do not, we will withdraw your conditionally accepted offer.
(Original) You must complete the following steps before USF will consider your application complete and begin admission evaluation. Pay the non-refundable $30 application fee or submit an application fee waiver.
(Simplified) You must submit an online application with a nonrefundable $30 application fee. You can also submit an application fee waiver.

Table 1: Original vs our expert simplified version of college admissions texts.

simplification in college admissions texts, as well as model performance in this new domain. The lack of domain diversity has been identified as a critical issue of text simplification as a whole (Alva-Manchego et al., 2020).

This work introduces **PSAT** (**P**rofessionally **S**implified **A**dmissions **T**exts), the first manually simplified and verified corpus of admissions instructions from 112 randomly sampled US post-secondary institutions. Our professional simplification process are guided by extensive existing literature on manual simplification, including lowering syntactic complexity, improving lexical cohesion, and elaborating jargon (Crossley et al., 2008; McNamara et al., 2014; Siddharthan, 2014). Given that the admissions process normally requires application fees, completion of an application, writing and submitting of essays, sending of transcripts, and possibly more processes, it is crucial to understand how to compose admissions application instructions to clearly but accurately explain these processes, especially to first-generation in college students who may not have support from their secondary school, family, or household (Taylor, 2020b). *To this end, every document in PSAT was manually verified by 2 subject-matter experts (among a total of 10 experts) who are employed as full-time admissions professionals in US institutons of higher education*.

In addition to full document simplification, we also create manual alignments (by the author who originally simplified all documents), aligning each sentence in the original text to its simplified version. This entailed 1,883 sentence pairs total.

Our analyses showed that the simplified documents reduced the reading level of these texts from grade 13.3 to grade 9.8, making it much more accessible for minority and emergent bilingual students. For experiments on automatic simplification, in this paper we focus on *sentence* simplification utilizing the aligned sentence pairs. We start from zero-shot transfer from existing simplification models pre-trained on both Wikilarge (Zhang and Lapata, 2017) and Newsela (Xu et al., 2015), as well as fine-tuned models on PSAT. We showed that this is a challenging domain for simplification. Additionally, our professional manual inspection of model outputs points out domain-specific errors that impact the accuracy of the simplifications.

2 Background and related work

2.1 Readability in admissions text

Decades of research in higher education has demonstrated that higher education information is often unreadable by prospective student audiences who read at average, pre-enrollment reading comprehension levels, roughly the 12th grade (Taylor, 2018, 2019, 2020b). At a larger scale, literacy research has consistently found that the average United States adult reads and comprehends below the 9th grade level, suggesting that many adults many not be able to understand how to read college admissions information and successfully apply to an institution of higher education in the United States (Hauser et al., 2005; Mamedova and Pawlowski, 2019; Sum et al., 2004).

Pertinent to the study at hand, prior research has found that undergraduate admissions instructions (what a student needs to accomplish and submit to the institution to be considered for admission) are often written above the 14th grade English reading comprehension level (Taylor, 2020b). Moreover, qualitative research focused on student experiences during the application process has found that prospective students, many of them from low-income and first-generation in college backgrounds, often struggle understanding higher education jargon such as *undergraduate*, *FAFSA*, and *verification*, all of which are critical to comprehend during the admissions and enrollment management process (Ardoin, 2013; Taylor and Bicak, 2019). Here, many students may be academically prepared for the rigors of a college curriculum, yet they may not understand the language and processes of higher

education writ large. Such a misunderstanding may systematically exclude these students due to a lack of support during the application process, not a lack of preparedness for college itself.

However, little has changed in the decades during which higher education researchers have been investigating the complexity of the college admissions process. In the 1980s and early 1990s, researchers called for the admissions process to be simplified and for common admissions applications to be widely embraced by institutions across the country (Astin et al., 1982; Gandara, 1986; Post, 1990). Yet by the early 2000s, researchers were still reporting that many minoritized students and their families struggled to comprehend how to complete the college admissions process and enroll in higher education (McBrien, 2005; Rosa, 2006; Tornatzky et al., 2002; Ward, 2006).

Recently, given the considerable personal and financial hardships facilitated by the COVID-19 pandemic, students and families continue to report that the college admissions process is too difficult (McCulloh, 2022) and often requires dozens of hours to complete (Reilly, 2021), and that institutions of higher education have not made efforts to ease the admissions process by simplifying the instructions for how to apply and enroll into higher education (Hurtado et al., 2020; Morrison, 2021). Subsequently, this study attempts to analyze simplifications of college admissions instructions to create an automated model that may automatically simplify college admissions instructions, easing the information burden for prospective students and families, while also easing the workload of admissions practitioners working for institutions of higher education.

2.2 Text simplification

Modern text simplification systems are typically encoder-decoder models that performs monolingual translation from the original to simplified text (Wang et al., 2016; Zhang and Lapata, 2017; Kriz et al., 2019; Dong et al., 2019; Martin et al., 2020; Devaraj et al., 2021; Maddela et al., 2021). The majority of this research are at the sentence level, trained using two corpora: the Wikipedia-Simple Wikipedia aligned corpus (Woodsend and Lapata, 2011; Coster and Kauchak, 2011; Zhang and Lapata, 2017) and the Newsela simplification corpus (Xu et al., 2015). As a result, it is unclear whether models trained on these datasets can be applied in other domains, and progress is hindered by the lack of diverse datasets (Alva-Manchego et al., 2020). This work is a step towards mitigating towards this issue.

The training of supervised models rely on automatically aligned sentence pairs (Jiang et al., 2020). However, Devaraj et al. (2022) demonstrated that noisy alignments can lead to inaccuracies during the simplification process. Instead, in this work, all documents are manually aligned by the author of the simplification.

3 Data collection and simplification

The main work of PSAT included performing manual text simplification, followed by acceptability judgements made by subject-matter experts with domain-specific professional work experience in either admissions or financial aid. Below, we detail the corpus development process, including the gathering of college admissions instructions (Section 3.1), professional simplification and its principles (Section 3.2) and verification (Section 3.3).

3.1 Raw text extraction

To gather the original texts (i.e., publicly available admissions instructions texts) for this study, we employed the National Center for Education Statistics' Integrated Postsecondary Education Data System (IPEDS)[1] to gather institutional URLs to each institution's admissions application instructions website. Moreover, we decided to gather data during the college search and exploration process, typically occurring between August and November of each year in the US (Hossler and Gallagher, 1987). Understanding both student exploration and institutional information practices, we gathered all website data for this study in October 2019. Initially, we gathered undergraduate admissions instructions from 335 institutions, which we then manually inspected and kept 112 that contain long, meaningful discourse other than metadata (e.g., addresses).

All instructions are then cleaned, by manually extracting the raw text.

3.2 Simplification

Personnel The texts, written in English (from US institutions), were manually simplified by an author of this paper who is a native English speaker. The author has a doctoral degree in education and has worked professionally in US postsecondary

[1] https://nces.ed.gov/ipeds/

education for over a decade, including work in undergraduate admissions. Thus the author engaged with their professional insight to simplify without losing critical information necessary for its comprehension and understanding.

Principles Longitudinal research (Crossley et al., 2008; McNamara et al., 2014; Siddharthan, 2014) has suggested that text can often be simplified using a smaller lexicon than the original text, rewriting sentences so that adjoining sentences share syntactic features (e.g., punctuation marks, independent clauses followed by main clauses), and ensuring that previously used lexical items appearing earlier in a text are re-introduced later in the text to improve comprehension of the text. Thus our manual simplification process include *reducing syntactic complexity*, *increasing lexical cohesion* (word overlap and frequency), *the elaboration and explanation of jargon and acronyms*, and *domain-specific principles*.

(1) Reducing syntactic complexity. We adopted Coleman (1962)'s framework for sentence-level simplification: raising clause fragments to full sentences, dividing sentences joined by conjunctions (e.g., because, but, for, or), avoid dividing sentences joined by the conjunction "and", and shortening clauses by using periods where other forms of grammatical punctuation may be found (e.g., semicolons, colons, commas). While shortening, we ensured that we do not lose critical information which would cause a factual error (Devaraj et al., 2022).

Additionally, extant research has suggested that writing or speaking in active voice rather than passive voice can increase simplicity, and this rewrite can often lead to shorter sentences (DeVito, 1969; Ferreira, 1994). As a result, we identified instances of passive voice in each admissions text and rewrote them into active voice, e.g.,

> **(Original)** The application must be completed by the student.
>
> **(Simplified)** The student must complete the application.

(2) Increasing lexical cohesion. Hulme et al. (1997) learned that increasing the word frequency in an informative text helped with the short-term memory recall of research participants regarding the content of the text, supporting the finding that increasing the word frequency in a text may lead to a better understanding of the text on behalf of the reader. These words included "you," "must," "official," "transcript," and many others. Therefore, we simplified in a way that promotes lexical overlap across documents.

Pertinent to this study, Monaghan et al. (2017) also found that individual differences across bilingual readers in terms of word frequency effects were due to exposure to word diversity, not an individual's vocabulary size (personal lexicon). This finding supported the use of increasing word frequency to increase a text's simplicity and possible readers' comprehension of the text. Thus, we also attempted to identify content words that could be repeated earlier or later in each text separately, including words like "you," "must," "submit," "transcripts," "contact," and many others.

(3) Elaboration and explanation of jargon and acronyms. Research on acronyms and initialisms has found that using these lexical items in potentially unfamiliar text can be confusing to readers, thus making the text more difficult to read (Cannon, 1989; Ibrahim, 1989; Rúa, 2002; Laszlo and Federmeier, 2007). Using acronyms often hinders clarity, as the reader may need to parse extra text or consult another text in order to decipher the acronym or initialism and fully comprehend the text (Rúa, 2002; Taghva and Gilbreth, 1999). We attempted to locate acronyms (e.g., ACT) and initialisms (e.g., GPA) within each text and ensure that these acronyms and initialisms were clear and commonly used, so that students would not be confused when reading the simplification. Common acronyms and initialisms in this dataset included "ACT," "SAT," "GED," "US," and several others. Subject-matter experts determined that although acronyms and initialisms may be confusing, including the acronyms and initialisms without lengthy definitions was best to assist with simplification, as subject-matter experts determined that the vast majority of students would understand these acronyms and initialisms without context clues or definitions.

(4) Domain-specific principles. As admission instructions are tied with direct consequences for postsecondary education, we work with 10 subject-matter experts (SME, their background detailed in Section 3.3) such that our simplification is accurate as possible within this particular domain.

Subject-matter experts unanimously determined that admissions instructions should only contain the instructions themselves, and that any extraneous text should be removed to ensure minimalism and simplicity. Consequently, text was removed

(from the original instructions) by SMEs if the text was not pertinent to the application process itself, including welcome statements, espoused institutional beliefs, and language that was determined to be marketing and/or branding and not admissions instructions. Examples of removed texts include, *"You don't have any time to waste, so we made applying as simple as possible"*, *"The Office of Admissions assists prospective students in exploring the academic opportunities"*.

Moreover, there were several keywords within sentences that subject-matter experts insisted remain or be added from original to simplified texts. For example, the words "official" and "transcript" needed to be retained in the simplified versions; and if these words did not appear in the original, they were added into the simplified version for clarity for the prospective student. In addition, SMEs agreed that simple admissions text includes second-person pronoun usage and modal verbs to provoke action on behalf of the student, for example:

> **(Original)** Application fee waivers are available for students with demonstrated financial need.
>
> **(Simplified)** If you cannot pay the fee, you can apply for an application fee waiver.

Ultimately, subject-matter experts felt that this language rendered the text simpler because students would better understand that they were responsible for completing parts of the admissions process.

3.3 Acceptability Judgements by Subject-Matter Experts (SMEs)

To determine whether the simplification of admissions application instructions were acceptable—that is to say they did not lose critical information or accuracy between the pre- and post-simplification process—we engaged with ten subject-matter experts (SMEs). Each simplified text was verified by 2 SMEs independently; in total, we engaged with 10 SMEs, who volunteered their time.

Personnel All ten of the SMEs had professional backgrounds in U.S. postsecondary admissions, having worked at least five years full-time in college admissions offices in the United States. These SMEs were identified through professional networks and snowball methods, as several of our SMEs knew colleagues from different institutions or educational entities who would serve as high-quality, knowledgable SMEs.

Moreover, we engaged with a diverse group of SMEs from different institution types (i.e., community colleges, public four-year institutions, private liberal arts colleges) and with various lengths of experience to capture the potential variability of admissions and financial aid parlance, jargon, and communication style. As the first study of its kind, identifying SMEs from diverse backgrounds provided more generalizability and reliability of findings, thus informing future research and practice regarding the communication of admissions application instructions to students and their support networks. Four subject-matter experts worked at public, four-year universities, four worked at private, four-year universities, and two worked at public, two-year community colleges.

Process To perform the acceptability judgement, the SME was presented with both pre- and post-simplification texts in real time over a Zoom video conference meeting. Then, we asked the SME to read the pre-simplified (original) text, followed by the post-simplification (simplified) text and determine whether the simplified text was acceptable. For example, changing the verb "submit" to "complete" is not acceptable because "submit" implies the documentation or information is being submitted by a submitter to a submittee, while "complete" only implies the documentation or information is completed and not directed to any educational stakeholder.

If a simplification was deemed unacceptable by one or more SMEs, we asked the SME what simplification would be acceptable through an iterative process in real time across all texts in this study. Once the SME provided their feedback and we integrated their feedback into the simplified text, the same SME was again asked to read both the both pre- and post-simplification texts in real time and render their acceptability judgement. If at any time there was an instance where a lexical item (e.g., single word, acronym, initialism, compound adjective), sentence, or paragraph could not be simplified, the pre-simplified section of that text was used.

3.4 Manual Alignments

In the PSAT corpus, the author of the simplified documents manually added sentence alignments across the original and simplified texts. Specifically, for each sentence in the original version, we provide the indice(s) of the corresponding simplified sentence. This comes to a total of 1,883 alignments, excluding sentences that are kept un-

(Simplified texts) just, development, following, prior, credit, sent, steps, recommended, applications, freshmen, payment, better, no, currently, once, chapman, plan, well, way, submitting, educational, aid, ensure, enroll, stonehill, options, completed, documents, code, create, credits, rolling, option, begin, appropriate, completing, now, earned, out, reviewed, checklist, items, submission, during, process, general, colleges, receive, please, standardized
(Original texts) codes, materials, how, andor, csu, nonnative, paintings, prepare, invite, dualcredit, homeschool, cannot, using, words, admit, certiphi, can, must, native, responses, resume, you, selfreport, letter, georgia, closer, bonaventure, prompted, event, inclusion, syllabi, statements, dualdegree, significant, quebec, tb, former, thinking, filmmaking, carrolls, concise, testingservicesucmoedu, baton, crucial, parker, coach, cresson, did, esl, autobiography

Table 2: Words most associated with simplified vs. original texts, calculated with log-odds ratios.

changed, metadata (e.g., addresses), and short ($<$ 2-word) headers.

4 Analysis

Table 3 shows comparative statistics of the 112 original—simplified bitexts; we see that on average the simplified versions have shorter and fewer sentences. The original texts in PSAT are above the 12-th grade reading level (measured with Flesch-Kincaid (Kincaid et al., 1975)) of an average high school graduate, confirming the findings in Taylor (2020a,b). The simplified version lowered this reading barrier.

We also study lexical items that are most associated with the original texts vs. the simplified version. Concretely we calculate the log-odds ratio of each word w_i, comparing the conditional probability of w_i in the original instructions $\mathbf{D}_o$ or the simplified ones $\mathbf{D}_s$: $\text{logodds}(w_i) = \log(P(w_i|\mathbf{D}_o)/P(w_i|\mathbf{D}_s))$ (Nye and Nenkova, 2015). Excluding punctuation and numbers, the words with the highest and lowest log-odds ratios with respect to simplified texts are shown in Table 2. We see that after simplification, the vocabulary becomes more standardized towards the application and admissions process itself.

Finally, we measure the abstractiveness of the simplified texts; namely, how much paraphrasing was used? For this we use the *aligned* portion of PSAT; the percentage of unique uni-, bi-, and tri-grams in the simplified texts are 34.4%,52.4%,58.6%, respectively. Thus the amount of new jargon introduced through simplification is enough to be significant; this indicates that simpler vocabulary is being introduced, and significant paraphrasing happened during simplification.

	# sents	# tokens	FK
Original	29.6	10.2	13.3
Simplified	17.9	7.1	9.8

Table 3: Comparison of original vs. simplified texts: average number of sentences, tokens, and the Flesch-Kincaid grade level.

5 Sentence Simplification Experiments

We establish baselines for the sentence simplification task in PSAT, using the aligned sentences. With these experiments we hope to evaluate existing models—which are trained in other domains—in a zero-shot manner, gauge the utility of fine-tuning on PSAT, and discuss the challenges of simplification in this domain.

5.1 Models

ACCESS Our baseline is the `ACCESS` model (Martin et al., 2020), trained on Wiki-large (Zhang and Lapata, 2017). `ACCESS` uses a transformer trained from scratch (i.e., randomly initialized).

T5 We also fine-tune the T5 model (Raffel et al., 2020) for the simplification task. T5 is a pre-trained large-scale encoder-decoder model optimized on conditional generation tasks (e.g., summarization and machine translation), in addition to unsupervised objectives. We fine-tune T5-base on both Wikilarge and Newsela (Xu et al., 2015), denoted as `T5-wiki` and `T5-newsela`. Following Devaraj et al. (2022), the models are fine-tuned with prefix *"summarize:"* for 5 epochs with a batch size of 6 and learning rate 3e-4. The default greedy decoding was used.

Additionally, we further fine-tune these models on PSAT's training data (Section 5.2) for 3 more epochs, denoted as `T5-wiki-ft` and `T5-newsela-ft`.

5.2 Experimental setup

Data We split PSAT documents into 50% train, 30% test, and 20% validation (the test and validation sets constitute 50% of the data, such that the number of test and validation documents

	Train	Test	Validate
# Files	56	33	23
# Sentence Pairs	955	559	369

Table 4: The number of files and sentence pairs for the train, test, validation splits in PSAT.

	BERTScore	BLEU	SARI
ACCESS	0.898	0.104	**0.271**
T5-newsela	0.906	0.132	0.226
T5-wiki	0.903	0.171	0.169
T5-newsela-ft	0.919	0.210	0.241
T5-wiki-ft	**0.923**	**0.239**	0.216

Table 5: Automatic evaluation results of simplification systems.

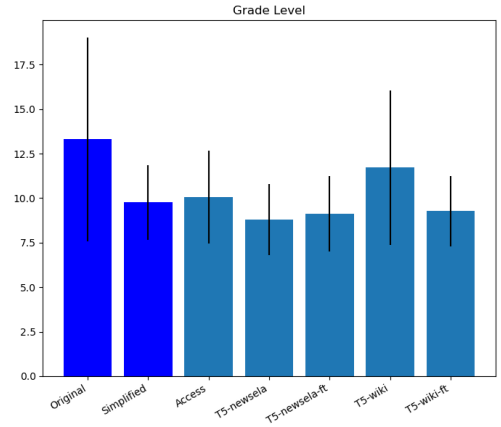

Figure 1: The Flesch-Kincaid Grade Level averages and standard deviation for the original and manually simplified texts, as well as the simplified texts generated by the five models.

will be large enough). We then use the manual alignments in these document sets for training/testing/validation examples. Table 4 shows the number of documents and aligned sentences in each split.

Evaluation For automatic evaluation, we report the following metrics:

(1) **SARI** (Xu et al., 2016), a widely used metric especially designed according to the edit nature of simplification. We report the average n-gram F1 scores corresponding to keep, delete, and add operations.[2]

(2) **BLEU** (Papineni et al., 2002), a standard metric in machine translation and other conditional generation tasks. Xu et al. (2016) stated that while SARI is better at capturing simplicity, BLEU has a stronger correlation with grammar and meaning.

(3) **BERTScore** (Zhang et al., 2019), which is shown to correlate better with human references in generation tasks, and has been adopted as an evaluation metric for summarization (Ahuja et al., 2022).

5.3 Results

Results are shown in Table 5. First, notice that all scores are lower in terms of absolute scale compared to the datasets they are trained on; for instance, Devaraj et al. (2022) reported SARI scores over 36 for all models on Newsela and Wiki-large. Although `ACCESS` obtained the highest SARI score, its BLEU and BERTScores are the lowest; our manual inspection (Section 6) confirmed that this model does not produce satisfactory simplification for admissions instructions. When comparing `T5-newsela` and `T5-wiki`, there is no clear superior model as `T5-wiki` has higher BLEU but lower SARI, and the two have very little difference in terms of BERTScore. This suggests that training on either Wikipedia articles or news articles as a starting point is acceptable.

The clearest trends across all metrics is that fine-tuning on PSAT substantially improves performance regardless of whether the model is pre-trained on Wikilarge or Newsela, demonstrating that in-domain fine-tuning is useful even with a moderately sized dataset like PSAT. This finding is consistent with recent discoveries in summarization (Yu et al., 2021).

Figure 1 depicts the Flesch-Kincaid grade level for the original sentences, their manually simplified versions, and the model outputs among the aligned data. Note that although we perform this analysis due to Flesch-Kincaid's superior performance in assessing the readability of other technical texts in an unsupervised manner (Devaraj et al., 2021), we do not intend to use Flesch-Kincaid as one our main automatic evaluation metrics for reasons pointed out by Alva-Manchego et al. (2021) and Tanprasert and Kauchak (2021). Comparing the five models, `T5-wiki` yielded a higher (and more varied reading level) than all other models and the manually simplified. On the contrary, `T5-newsela`'s output reading level is slightly lower than the manual simplification. Meanwhile,

[2]Note that SARI scores are more reliable if multiple reference simplifications are present, but we are limited to one reference due to personnel, time, and budget constraints.

	Fluency	Simplicity	Accuracy
ACCESS	1/25	1/25	1/25
T5-newsela	4/25	4/25	4/25
T5-wiki	4/25	4/25	5/25
T5-newsela-ft	2/25	2/25	1/25
T5-wiki-ft	5/25	5/25	5/25
None	9/25	9/25	9/25

Table 6: Number of times (out of 25) that each system received the highest rank, along with # of times that no system was selected.

both `T5-newsela-ft` and `T5-wiki-ft` produced desirable readability levels; notably fine-tuning was able to improve the readability of `T5-wiki` outputs.

6 Manual Inspection

Finally, because PSAT is a very new domain that comes with its own characteristics and requirements for accurate simplification, we perform a small scale manual inspection with the professional who authored the simplified documents; this professional has not seen model-generated outputs in NLP before. Because of time commitment required to inspect outputs from all 5 models, we were only able to study 25 examples. Nonetheless we hope the insights here will guide future work on domain-specific evaluation metrics.

Concretely, we randomly sample 25 sentences from the test set. We ask the professional to inspect outputs based on three metrics: **fluency, simplicity, and accuracy**. Under each metric, we ask the professional to choose the highest ranked simplification; an option of *none* is available if no model provided satisfactory outputs for any criteria.

Results in Table 6 confirmed that `T5-wiki-ft` performed the best, and that `ACCESS` outputs do not outperform other models. Similar to automatic evaluation results, `T5-wiki` vs `T5-newsela` However, there is also a large portion of examples where "none" was chosen. Additionally, `T5-newsela-ft` performed much worse than `T5-newsela`. This suggests that automatic metrics do not capture some of the aspects that the professional deems important in their judgments. This echos findings from Alva-Manchego et al. (2021) but emphasizes the importance of human evaluation for specific domains. We detail some of the insights qualitatively below.

First, several simplifications required adding words to the simplified version that did not appear in the original, however the model did not learn to do so. The most frequent example was simplifications adding "official" to the phrase "Official Transcript(s)". Here, subject-matter experts felt that adding "official" was necessary to make it clearer and more simple for students to understand that they needed to submit official transcripts instead of copies or screenshots of their transcripts.

Similarly, many simplifications added the subject and modal verb "you must" to phrases such as "submit your official transcript(s)", "complete an application", and "apply by the priority date", as discussed in Section 3.2. The subject-matter experts felt it was necessary to clarify that a student "must" complete these tasks, otherwise they would not be considered for admission. However, the model could not generate such phrasing.

Finally, many models removed keywords from the original versions. This most often happened when an institution accepted multiple application types, such as the Coalition for College Application and the Common Application. For instance, one sentence original sentence read, "*We accept the Coalition for College Application and the Common Application*". However, several models attempted to simplify the sentence and remove either the Coalition for College Application or Common Application, as they both contain the word "application", and the model may have removed these "redundant" phrases. However, both application types needed to appear in the simplified version to gain subject-matter expert approval, as an accurate admissions instructions contain all of the different applications that a student may use to apply for admission, not just the simplest application. As a result, several models did not accurately keep keywords from original to simplified versions because the models did not have this domain-specific information that subject-matter experts had.

7 Discussions and Conclusions

This work presents PSAT, a text simplification corpus consisting of admission instructions texts from 112 US higher education institutions and their simplified versions. PSAT texts are professionally simplified and verified, rendering it the first-of-its-kind and most accurate dataset in this high-stake domain. We showed that this dataset is challenging for existing simplification models, especially due to domain mismatch, and domain-specific requirements for

the accuracy of information.

We recognize the limitations of this study. While there are over 6,000 institutions of higher education in the US, this dataset sampled a small number of these institutions given time constraints and the substantial work necessary to gather and simplify text and work with subject-matter experts to approve simplified texts.

Ultimately, we will perform future research—using this dataset—to specifically identify which simplifications were acceptable and unacceptable (at the lexical item-, sentence- and paragraph-level), thus informing admissions practitioners as to how text can or cannot be simplified for students and prospective students, broadly. We hope PSAT is a first step towards automatic simplification systems for fairer access to higher education.

Acknowledgements

The authors would like to acknowledge the subject-matter experts who volunteered their time to read the texts from this study and provide their professional judgement. This research is partially supported by NSF grants IIS-2145479, IIS-2107524. We acknowledge the Texas Advanced Computing Center (TACC)[3] at UT Austin for many of the results within this paper.

References

Ojas Ahuja, Jiacheng Xu, Akshay Gupta, Kevin Horecka, and Greg Durrett. 2022. ASPECTNEWS: Aspect-oriented summarization of news documents. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 6494–6506.

Fernando Alva-Manchego, Carolina Scarton, and Lucia Specia. 2020. Data-driven sentence simplification: Survey and benchmark. *Computational Linguistics*, 46(1):135–187.

Fernando Alva-Manchego, Carolina Scarton, and Lucia Specia. 2021. The (un) suitability of automatic evaluation metrics for text simplification. *Computational Linguistics*, 47(4):861–889.

Mary Sonja Ardoin. 2013. *Learning a different language: Rural students' comprehension of college knowledge and university jargon*. North Carolina State University.

Alexander W Astin et al. 1982. *Minorities in American higher education. Recent trends, current prospects, and recommendations*. ERIC.

Susan Auerbach. 2004. Engaging latino parents in supporting college pathways: Lessons from a college access program. *Journal of Hispanic Higher Education*, 3(2):125–145.

Garland Cannon. 1989. Abbreviations and acronyms in english word-formation. *American Speech*, 64(2):99–127.

Edmund Benedict Coleman. 1962. Improving comprehensibility by shortening sentences. *Journal of Applied Psychology*, 46(2):131.

Amy Cook, Rachelle Pérusse, and Eliana D Rojas. 2012. Increasing academic achievement and college-going rates for latina/o english language learners: A survey of school counselor interventions. *The Journal of Counselor Preparation and Supervision*, 4(2):2.

William Coster and David Kauchak. 2011. Simple English Wikipedia: a new text simplification task. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, pages 665–669.

SA Crossley, J Greenfield, and DS Mcnamara. 2008. Text readability using assessing based indices. *TESOL Quarterly*, 42(3):475–493.

Ashwin Devaraj, Iain Marshall, Byron C Wallace, and Junyi Jessy Li. 2021. Paragraph-level simplification of medical texts. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 4972–4984.

Ashwin Devaraj, William Sheffield, Byron Wallace, and Junyi Jessy Li. 2022. Evaluating factuality in text simplification. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 7331–7345.

Joseph A DeVito. 1969. Some psycholinguistic aspects of active and passive sentences. *Quarterly Journal of Speech*, 55(4):401–406.

Yue Dong, Zichao Li, Mehdi Rezagholizadeh, and Jackie Chi Kit Cheung. 2019. EditNTS: An neural programmer-interpreter model for sentence simplification through explicit editing. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 3393–3402.

Fernanda Ferreira. 1994. Choice of passive voice is affected by verb type and animacy. *Journal of Memory and Language*, 33(6):715–736.

Stella M Flores. 2010. State dream acts: The effect of in-state resident tuition policies and undocumented latino students. *The Review of Higher Education*, 33(2):239–283.

Patricia Gandara. 1986. Chicanos in higher education: The politics of self-interest. *American Journal of Education*, 95(1):256–272.

[3] https://www.tacc.utexas.edu

Brent Goff, Vanessa Patino, and Gary Jackson. 2004. Preferred information sources of high school students for community colleges and universities. *Community College Journal of Research & Practice*, 28(10):795–803.

Kenneth E Hartman. 1997. College selection and the internet: Advice for schools, a wake-up call for colleges. *Journal of College Admission*, 154:22–31.

Robert M Hauser, Christopher F Edley Jr, Judith Anderson Koenig, and Stuart W Elliott. 2005. *Measuring Literacy: Performance Levels for Adults.* ERIC.

Don Hossler and Karen Gallagher. 1987. Studying student college choice: A three-phase model and the implication...-supersearch powered by summon. *College and University*, 62:201–21.

Charles Hulme, Steven Roodenrys, Richard Schweickert, Gordon DA Brown, Sarah Martin, and George Stuart. 1997. Word-frequency effects on short-term memory tasks: evidence for a redintegration process in immediate serial recall. *Journal of Experimental Psychology: Learning, Memory, and Cognition*, 23(5):1217.

Sylvia Hurtado, Hector Vicente Ramos, Edwin Perez, and Xochilth Lopez-Salgado. 2020. Latinx student assets, college readiness, and access: Are we making progress? *Education Sciences*, 10(4):100.

AM Ibrahim. 1989. Acronyms observed. *IEEE Transactions on Professional Communication*, 32(1):27–28.

Ludivine Javourey-Drevet, Stéphane Dufau, Thomas François, Núria Gala, Jacques Ginestié, and Johannes C Ziegler. 2022. Simplification of literary and scientific texts to improve reading fluency and comprehension in beginning readers of french. *Applied Psycholinguistics*, 43(2):485–512.

Chao Jiang, Mounica Maddela, Wuwei Lan, Yang Zhong, and Wei Xu. 2020. Neural CRF model for sentence alignment in text simplification. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7943–7960.

Yasuko Kanno. 2018. Non-college-bound english learners as the underserved third: How students graduate from high school neither college-nor career-ready. *Journal of Education for Students Placed at Risk (JESPAR)*, 23(4):336–358.

J Peter Kincaid, Robert P Fishburne Jr, Richard L Rogers, and Brad S Chissom. 1975. Derivation of new readability formulas (automated readability index, fog count and flesch reading ease formula) for navy enlisted personnel. Technical report, Naval Technical Training Command Millington TN Research Branch.

Reno Kriz, João Sedoc, Marianna Apidianaki, Carolina Zheng, Gaurav Kumar, Eleni Miltsakaki, and Chris Callison-Burch. 2019. Complexity-weighted loss and diverse reranking for sentence simplification. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 3137–3147.

Sarah Laszlo and Kara D Federmeier. 2007. Better the dvl you know: Acronyms reveal the contribution of familiarity to single-word reading. *Psychological Science*, 18(2):122–126.

Mounica Maddela, Fernando Alva-Manchego, and Wei Xu. 2021. Controllable text simplification with explicit paraphrasing. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 3536–3553.

Saida Mamedova and Emily Pawlowski. 2019. Adult literacy in the united states. data point. nces 2019-179. *National Center for Education Statistics*.

Louis Martin, Éric Villemonte De La Clergerie, Benoît Sagot, and Antoine Bordes. 2020. Controllable sentence simplification. In *Proceedings of the 12th Language Resources and Evaluation Conference*, pages 4689–4698.

J Lynn McBrien. 2005. Educational needs and barriers for refugee students in the united states: A review of the literature. *Review of educational research*, 75(3):329–364.

Edna McCulloh. 2022. An exploration of parental support in the retention of rural first-generation college students. *Journal of College Student Retention: Research, Theory & Practice*, 24(1):144–168.

Danielle S McNamara, Arthur C Graesser, Philip M McCarthy, and Zhiqiang Cai. 2014. *Automated evaluation of text and discourse with Coh-Metrix*. Cambridge University Press.

Padraic Monaghan, Ya-Ning Chang, Stephen Welbourne, and Marc Brysbaert. 2017. Exploring the relations between word frequency, language exposure, and bilingualism in a computational model of reading. *Journal of Memory and Language*, 93:1–21.

Joe Morrison. 2021. 4 ways to transform college admissions. *eSchool News*.

Benjamin Nye and Ani Nenkova. 2015. Identification and characterization of newsworthy verbs in world news. In *Proceedings of the 2015 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 1440–1445.

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. Bleu: a method for automatic evaluation of machine translation. In *Proceedings of the 40th annual meeting of the Association for Computational Linguistics*, pages 311–318.

Patricia A Pérez and Patricia M McDonough. 2008. Journal of hispanic higher. *Journal of Hispanic Higher Education*, 7:249.

David Post. 1990. College-going decisions by chicanos: The politics of misinformation. *Educational Evaluation and Policy Analysis*, 12(2):174–187.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J Liu. 2020. Exploring the limits of transfer learning with a unified text-to-text transformer. *Journal of Machine Learning Research*, 21:1–67.

Katie Reilly. 2021. Applying to college was hard. covid-19 made it impossible. *Time*.

Mari Luna De La Rosa. 2006. Is opportunity knocking? low-income students' perceptions of college and financial aid. *American behavioral scientist*, 49(12):1670–1686.

Paula López Rúa. 2002. On the structure of acronyms and neighbouring categories: A prototype-based account. *English Language & Linguistics*, 6(1):31–60.

Advaith Siddharthan. 2014. A survey of research on text simplification. *ITL-International Journal of Applied Linguistics*, 165(2):259–298.

Andrew Sum, Irwin Kirsch, and Kentaro Yamamoto. 2004. Pathways to labor market success: The literacy proficiency of US adults. *Policy Information Report, Educational Testing Service*.

Kazem Taghva and Jeff Gilbreth. 1999. Recognizing acronyms and their definitions. *International Journal on Document Analysis and Recognition*, 1(4):191–198.

Teerapaun Tanprasert and David Kauchak. 2021. Flesch-Kincaid is not a text simplification evaluation metric. In *Proceedings of the 1st Workshop on Natural Language Generation, Evaluation, and Metrics (GEM 2021)*, pages 1–14.

Zachary W Taylor. 2017. Twenty-first-century communicators: How registrars can reject 'university idiolect'. *The Successful Registrar*, 17(9):6–6.

Zachary W Taylor. 2018. Intelligibility is equity: Can international students read undergraduate admissions materials? *Higher Education Quarterly*, 72(2):160–169.

Zachary W Taylor. 2019. Lost in translation: Linguistic hurdles facing us language-minority students of color pursuing international higher education. . *Lost in Translation: Linguistic Hurdles Facing US Language-Minority Students of Color Pursuing International Higher Education. Emerging Voices in Education*, 1(1):45–60.

Zachary W Taylor. 2020a. ¿ comprenderán mis amigos y la familia? analyzing spanish translations of admission materials for latina/o students applying to 4-year institutions in the united states. *Journal of Hispanic Higher Education*, 19(2):195–209.

Zachary W Taylor. 2020b. College admissions for l2 students: Comparing l1 and l2 readability of admissions materials for us higher education. *Journal of College Access (2020)*.

Zachary W Taylor and Ibrahim Bicak. 2019. What is the FAFSA? an adult learner knowledge survey of student financial aid jargon. *Journal of Adult and Continuing Education*, 25(1):94–112.

Louis G Tornatzky, Richard Cutler, and Jongho Lee. 2002. College knowledge: What latino parents need to know and why they don't know it. *Joyce Foundation*.

Tong Wang, Ping Chen, John Rochford, and Jipeng Qiang. 2016. Text simplification using neural machine translation. In *Thirtieth AAAI Conference on Artificial Intelligence*.

Nadia L Ward. 2006. Improving equity and access for low-income and minority youth into institutions of higher education. *Urban Education*, 41(1):50–70.

Kristian Woodsend and Mirella Lapata. 2011. Wikisimple: Automatic simplification of wikipedia articles. In *Twenty-fifth AAAI conference on artificial intelligence*.

Wei Xu, Chris Callison-Burch, and Courtney Napoles. 2015. Problems in current text simplification research: New data can help. *Transactions of the Association for Computational Linguistics*, 3:283–297.

Wei Xu, Courtney Napoles, Ellie Pavlick, Quanze Chen, and Chris Callison-Burch. 2016. Optimizing statistical machine translation for text simplification. *Transactions of the Association for Computational Linguistics*, 4:401–415.

Yasukata Yano, Michael H Long, and Steven Ross. 1994. The effects of simplified and elaborated texts on foreign language reading comprehension. *Language learning*, 44(2):189–219.

Tiezheng Yu, Zihan Liu, and Pascale Fung. 2021. Adaptsum: Towards low-resource domain adaptation for abstractive summarization. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 5892–5904.

Tianyi Zhang, Varsha Kishore, Felix Wu, Kilian Q Weinberger, and Yoav Artzi. 2019. Bertscore: Evaluating text generation with bert. In *International Conference on Learning Representations*.

Xingxing Zhang and Mirella Lapata. 2017. Sentence simplification with deep reinforcement learning. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 584–594.

On the Role of Pre-trained Language Models in Word Ordering: A Case Study with BART

Zebin Ou[1], Meishan Zhang[3], Yue Zhang[1,2]*
[1]School of Engineering, Westlake University
[2]Institute of Advanced Technology, Westlake Institute for Advanced Study
[3]Institute of Computing and Intelligence, Harbin Institute of Technology (Shenzhen)
{ouzebin,zhangyue}@westlake.edu.cn,
mason.zms@gmail.com

Abstract

Word ordering is a constrained language generation task taking unordered words as input. Existing work uses linear models and neural networks for the task, yet pre-trained language models have not been studied in word ordering, let alone why they help. We use BART as an instance and show its effectiveness in the task. To explain why BART helps word ordering, we extend analysis with probing and empirically identify that syntactic dependency knowledge in BART is a reliable explanation. We also report performance gains with BART in the related partial tree linearization task, which readily extends our analysis.[1]

1 Introduction

The task of word ordering (Wan et al., 2009; Zhang and Clark, 2015; Tao et al., 2021), also known as linearization (Liu et al., 2015), aims to assign a valid permutation to a bag of words for a coherent sentence. While early work uses word ordering to improve the grammaticality of machine-generated sentences (Wan et al., 2009), the task subsequently manifests itself in applications such as discourse generation (Althaus et al., 2004), machine translation (Tromble and Eisner, 2009; He and Liang, 2011), and image captioning (Fang et al., 2015). It plays a central role in linguistic realization (Gatt and Krahmer, 2018) of pipeline text generation systems. Advances in word ordering are also relevant to retrieval augmented generation (Guu et al., 2020), with outputs additionally conditioned on retrieved entries, which can constitute a bag of words. Word ordering can be viewed as constrained language generation with all inflected output words provided, which makes it more amenable for error analysis (§3.4). The task can be extended to tree linearization (He et al., 2009) or partial tree linearization (Zhang, 2013) with syntactic features as additional input. Syntactic models (Liu et al., 2015) and language models (Schmaltz et al., 2016) have been used in word ordering to rank candidate word permutations. Recently, Hasler et al. (2017) and Tao et al. (2021) explore different designs of neural models for the task. However, no existing studies investigate pre-trained language models (PLMs; Qiu et al. 2020) for word ordering, which have effectively improved various NLP tasks.

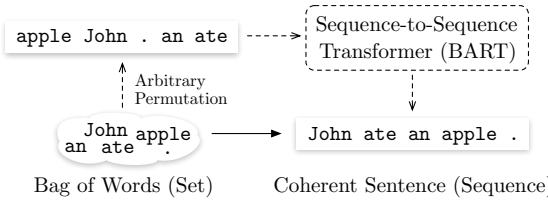

Figure 1: Illustration of our approach for word ordering. Bag-of-words inputs are first turned into pseudo-sequences with arbitrarily word permutations, and then fed to a sequence-to-sequence Transformer for coherent outputs.

Intuitively, the rich knowledge in PLMs can readily help word ordering. However, the unordered bag-of-words inputs may seem incompatible to PLMs with sequential inputs. The role of PLMs in word ordering thus remains an interesting research question. We fill the research gap by empirically investigating BART (Lewis et al., 2020), a pre-trained sequence-to-sequence Transformer (Vaswani et al., 2017), as an instance of PLMs for word ordering. Specifically, we assign an arbitrary permutation for the input bag of words to obtain a pseudo-sequence, and use sequence-to-sequence Transformers to generate ordered outputs, as illustrated in Figure 1. As BART uses subword inputs (Sennrich et al., 2016) instead of words (Schmaltz et al., 2016; Hasler et al., 2017; Tao et al., 2021), we implement constrained beam search using prefix trees (§3.1; Figure 2). Results show that BART substantially improves word ordering compared to our Transformer baseline, which already outperforms

*Corresponding Author
[1]We release our source code at https://github.com/simtony/BART-word-orderer

Proceedings of the 29th International Conference on Computational Linguistics, pages 6516–6529
October 12–17, 2022.

previous best (§3.3).

With Transformer models (including BART), we further investigate consequences of two major modeling decisions, which remain unexamined in the literature. First, while all previous studies assume output sequences constrained within permutations of input words, Tao et al. (2021) eliminate such constraints. We find the latter leads to a consistent performance drop, which is further attributed to missing words in outputs, a phenomenon related to the coverage issue in machine translation (Tu et al., 2016). Second, Hasler et al. (2017) use conditional probability $p(\boldsymbol{y}|\boldsymbol{x})$ instead of the unconditional one $p(\boldsymbol{y})$ of Schmaltz et al. (2016) to score candidate output permutations $\boldsymbol{y}$ of input bag of words $\boldsymbol{x}$. We provide the first fair comparison for the two approaches, and find that with small decoding beam, conditional models substantially outperform unconditional ones. Interestingly, such an advantage fails to persist as we increase the beam size.

Our Transformer word orderers may be sensitive to arbitrary word permutations in the input pseudo-sequence (§3.6). Recent studies (Sinha et al., 2021; Ettinger, 2020) show that Transformers are relatively insensitive to word permutations in *sequential* inputs. They are more sensitive to local orders than global orders of input subwords on the GLUE benchmark (Clouâtre et al., 2021). In contrast, we find that Transformer (including BART) word orderers are relatively insensitive to *both* word and subword permutations in inputs. Such result can be relevant to modeling unordered inputs with PLMs (Castro Ferreira et al., 2020; Lin et al., 2020).

We finally aim to explain why BART helps word ordering (§4). Analysis with probing (Rogers et al., 2020) provides *speculated* explanations for the utility of PLMs with the possession of numerous types of knowledge. However, for a reliable explanation, we need to identify the *specific* type of knowledge *relevant* to the task. In addition, the amount of the knowledge should be nontrivial *in* the PLM. With a procedure based on feature importance (Fraser et al., 2014) and probing (Hewitt and Manning, 2019), we empirically identify that knowledge about syntactic dependency structure reliably explains why BART helps word ordering. Our analysis can be readily extended to partial tree linearization (Zhang, 2013), for which we also report performance gains with our models (§5).

2 Related Work

Word Ordering Modeling Early work uses syntactic models (Zhang and Clark, 2011; Liu et al., 2015) and language models (Zhang et al., 2012; Liu and Zhang, 2015) to rank candidate permutations of input words. Liu and Zhang (2015) and Schmaltz et al. (2016) discuss their relative importance. Syntactic models rank candidates with the probability of the jointly predicted parse tree. They can be linear models (Wan et al., 2009) or neural networks (Song et al., 2018) with hand-crafted features. Language models use the probability of the output sentence for ranking. Early work uses statistical n-gram models (Zhang et al., 2012), which are later replaced by recurrent neural networks (Schmaltz et al., 2016). Most related to our work, Hasler et al. (2017) and Tao et al. (2021) formulate word ordering as conditional generation. Hasler et al. (2017) uses an LSTM decoder with attention (Bahdanau et al., 2015) and an encoder degenerating to an embedding layer. Tao et al. (2021) stack self-attention (Vaswani et al., 2017) layers as the encoder and use a decoder from pointer network (See et al., 2017). Both encode bag-of-words inputs with permutation invariant *word* encoders. In contrast, we turn bag-of-words inputs into *subword* pseudo-sequences and feed them to standard sequence-to-sequence models. Instead of investigating features, prediction targets, and model architectures as in previous work, we focus on the utility of BART in the task.

Word Ordering Decoding Early work relies on time-constrained best-first-search (White, 2005; Zhang and Clark, 2011). As it lacks an asymptotic upper bound for time complexity (Liu et al., 2015), later work with syntactic models (Song et al., 2018), language models (Schmaltz et al., 2016), and conditional generation models (Hasler et al., 2017; Tao et al., 2021) adopt beam search for decoding. All previous work assumes an output space constrained to permutations of input words except for Tao et al. (2021), who assume the output to be any sequences permitted by the vocabulary. However, the effect of such unconstrained output space is unexamined. We compare the difference between beam search with constrained and unconstrained output spaces.

Tasks Related to Word Ordering Word ordering was first proposed by Bangalore et al. (2000) as a surrogate for grammaticality test, and later formulated by Wan et al. (2009) as a standard task. A closely related task is CommonGen (Lin et al.,

2020), which aims to generate a coherent sentence subjective to *commonsense* constraints given a set of *lemmatized* concept words. In contrast, word ordering is a constrained *language modeling* task given *inflected* output words. Tree linearization (He et al., 2009) is a related task with full dependency trees as inputs. Dropping subsets of dependency arcs and part-of-speech tags results in partial tree linearization (Zhang, 2013). Further removing functional words and word inflections results in surface realization (Mille et al., 2020). Different from CommonGen and surface realization, the provided output bag-of-words limit reliance on domain knowledge and reduce ambiguity in output, making word ordering a concentrated case for testing generic linguistic capacity (Raji et al., 2021) of text generation models. In addition, word ordering requires no labeling in contrast to all these tasks.

PLMs and Non-Sequential Inputs PLMs with the Transformer (Vaswani et al., 2017) decoder are amenable for sequence generation (Raffel et al., 2020; Lewis et al., 2020). They have been used for sequence generation tasks with non-sequential inputs, such as AMR-to-Text (Mager et al., 2020), RDF-to-Text (Ribeiro et al., 2021), and CommonGen (Lin et al., 2020). Typically, non-sequential inputs are turned into sequential ones before being fed to PLMs. Additionally aiming to understand why BART helps word ordering, we adopt a similar approach and refrain from task-specific engineering, which allows the same sequence-to-sequence model for multiset and tree inputs, limiting extra confounding factors in our analysis.

Analysis with Probing Previous work on probing (Rogers et al., 2020) has identified various types of knowledge in PLMs, such as syntax (Hewitt and Manning, 2019), semantics (Tenney et al., 2019), and commonsense (Roberts et al., 2020). They are speculated to explain the utility of PLMs in target tasks. We make such explanations reliable for BART in word ordering by establishing the *relevance* of specific types of knowledge to the task, in addition to probing their existence in BART.

3 Word Ordering with BART

We describe our formulation of word ordering and how to adopt the sequence-to-sequence BART for the task (§3.1), report results on the standard PTB benchmark (§3.2 and §3.3), and analyze effects of different modeling decisions (§3.4–3.6).

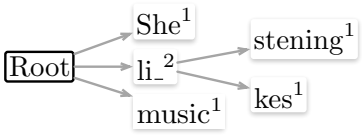

Figure 2: Prefix tree at initialization for constraints {"She", "li_ kes", "li_ stening", "music"}. Each path from the root to a leaf corresponds to a subword sequence. Except for the root, each node corresponds to a subword and tracks its count (superscript) in the constraints. A pointer (bold outline) denoting the previously generated subword y_{t-1} points to the root at initialization. At decoding step t, only subwords of child nodes with nonzero count are valid as the next token y_t. Once a subword is selected for y_t, we move the pointer to the corresponding node and decrement its count by 1. The pointer is reset to the root after reaching leaves. Decoding for each candidate finishes when all counts are zero.

3.1 Modeling Word Ordering

We formulate word ordering as conditional generation following Hasler et al. (2017). The input bag of words constitutes a multiset $\boldsymbol{x}$, where different elements can take the same value. The probability of output sequence $\boldsymbol{y}$, conditioning on $\boldsymbol{x}$, is parameterized by θ and factorizes auto-regressively:

$$p_\theta(\boldsymbol{y}|\boldsymbol{x}) = \prod_t p_\theta(y_t|\boldsymbol{y}_{<t}, \boldsymbol{x}) \quad (1)$$

where $\boldsymbol{y}_{<t}$ consists of previous generated tokens up to step $t-1$, the next token y_t takes a token from the output vocabulary. Output sequences start with a special token y_0 denoting beginning of sentences. Following Hasler et al. (2017), after solving θ with maximum likelihood estimation on the training set, we use beam search in an output space for the candidate $\boldsymbol{y}$ maximizing the product $\prod_t p_\theta(y_t|\boldsymbol{y}_{<t}, \boldsymbol{x})$.

Output sequences $\boldsymbol{y}$'s are generally constrained within permutations of input words (Schmaltz et al., 2016; Tao et al., 2021). To account for corrupted inputs (e.g., word deletion), Tao et al. (2021) use an unconstrained output space with any sequence permitted by the vocabulary. We analyze the output difference between decoding with constrained and unconstrained output space in §3.4.

Beam search with bag-of-words constraints can be simply implemented by tracking words to be generated using a multiset for each candidate in the beam and setting $p_\theta(y_t|\boldsymbol{y}_{<t}, \boldsymbol{x})$ of invalid (not in the multiset) next words to zero. The generated word y_t are then removed from the multiset after decoding step t. Decoding of each candidate ends with an empty multiset. Different from previous

work (Schmaltz et al., 2016; Hasler et al., 2017; Tao et al., 2021), subword segmentation of BART turns each input word into a sequence of subwords, which poses sequential constraints on the output. Thus tracking generated subwords and eliminate invalid next subwords during beam search require a different data structure. We compile subword sequences into a prefix tree as illustrated in Figure 2, where counts at nodes track subwords to be generated and child nodes correspond to valid next subwords. See Figure 6 in Appendix A for a concrete working example.

Conditional models use $p_\theta(y_t|\boldsymbol{y}_{<t}, \boldsymbol{x})$ to score the next token y_t given previously generated tokens $\boldsymbol{y}_{<t}$. They additionally depend on the input $\boldsymbol{x}$, which helps track words to be generated and mitigate the ambiguity of selecting y_t. In contrast, unconditional models (Schmaltz et al., 2016) with probability $p_\theta(y_t|\boldsymbol{y}_{<t})$ only depend on local information y_t and $\boldsymbol{y}_{<t}$, which can lead to high ambiguity of selecting y_t and attract beam search with small beams to local minimums. In §3.5, we analyze the difference between conditional and unconditional models with a fair comparison.

To instantiate $p_\theta(\boldsymbol{y}|\boldsymbol{x})$, we use a Transformer (Vaswani et al., 2017) consisting of both encoder and decoder pre-trained with BART (Lewis et al., 2020). Transformers use self-attention, a permutation invariant layer, to model contextual representations for input tokens. Vaswani et al. (2017) add distinct position embeddings onto input token embeddings at different positions to make self-attention sensitive to input orders. BART pre-trains Transformers to reconstruct corrupted input sequences. As BART assumes sequential inputs, we convert the multiset input $\boldsymbol{x}$ into a pseudo-sequence by assigning an arbitrary permutation to the input words, following Lin et al. (2020); see Figure 1 for an illustration. Although subword orders within each word are informative, Transformers (and BART) may be sensitive to the arbitrary word permutation in the input. We analyze the permutation sensitivity of our models in §3.6.

3.2 Settings and Implementations

Following previous work (Hasler et al., 2017; Tao et al., 2021), we use PTB[2] sections 2-21 (39,832 sentences) for training, section 22 (1,700 sentences) for development, and section 23 (2,416 sentences) for test. Punctuation escapes of PTB are reverted[3] to match the vocabulary of BART. We randomly shuffle words of each output sentence to create the input and perform BPE segmentation (Sennrich et al., 2016) for both input and output. BLEU (Papineni et al., 2002) are reported as the performance metric following Schmaltz et al. (2016). Our implementation is based on fairseq[4] (Ott et al., 2019).

We train a Transformer from scratch (denoted `RAND`) as the baseline and compared it to the finetuned BART base (denoted `BART`) to estimate gains from BART pre-training. Hyperparameters are tuned separately since different models have different optimums. We find vocabulary size 8000 optimal for `RAND`. Both models share identical architecture with 6-layer encoder and decoder. `RAND` (35 million parameters) has smaller hidden size 512 and feed-forward hidden size 1024 compared to 756 and 3072 of `BART` (140 million parameters). They both need heavy regularization: $\beta = 0.3$ for label smoothing (Pereyra et al., 2017), $p = 0.3$ for dropout (Srivastava et al., 2014), and $\alpha = 1$ for R-drop (Liang et al., 2021). Both models are trained using Adam (Kingma and Ba, 2015). We use 100 samples per batch, 4000 warm-up steps and learning rate 5e-4 for `RAND`; 20 samples per batch, 1000 warm-up steps and learning rate 1e-4 for `BART`. Learning rate decays with the inverse square root of training steps. We train the model till the development loss stops improving and average the last 5 checkpoints saved per 1000 training steps. We use beam size 64 to search on a constrained output space without additional specification. For unconstrained output space, we use beam size 64 and length normalization (Murray and Chiang, 2018).

3.3 Word Ordering Results

We compare our results with previous work under similar settings: for conditional models, bag2seq (`Bag`; Hasler et al. 2017) and AttM (`AttnM`; Tao et al. 2021) are included; for unconditional models, we include N-gram language models (`Ngram`) and RNNLM (`RNNLM`; Schmaltz et al. 2016) reproduced by Hasler et al. (2017). Except for `AttnM`, all models use a constrained output space. We do not consider heuristically tailored beam search (Schmaltz et al., 2016; Hasler et al., 2017) and focus on standard sequence-to-sequence modeling.

[2]We follow the LDC User Agreement for Non-Members license.

[3]We replace "-LCB-"and "-LRB-" with "(", "-RCB-" and "-RRB-" with ")", while removing all "\"

[4]https://github.com/pytorch/fairseq

Settings	B=5	B=64	B=512
Unconstrained			
AttnM	34.89	/	/
RAND-ours	38.53	38.95	39.04
BART-ours	**54.29**	**54.86**	**54.84**
Constrained			
Ngram	23.3	/	35.7
RNNLM	24.5	34.6	38.6
Bag	33.4	36.2	37.1
RAND-ours	38.97	39.52	39.59
BART-ours	**54.70**	**56.38**	**56.63**

Table 1: Test BLEU for word ordering on PTB. B=X denotes *standard* beam search with beam size X. Attn, Bag and our models are all conditional models while Ngram and RNNLM are unconditional ones.

Different from existing studies, we use BPE segmentation for all our settings.

As shown in Table 1, our baseline RAND outperforms previous best results with unconstrained (38.53 of RAND compared to 34.89 of AttnM with B=5) and constrained (39.59 of RAND compared to 38.6 of RNNLM with B=512) output space, showing the effectiveness of sequence-to-sequence modeling with Transformer. Compared to RAND, BART brings substantial improvements under different settings, with up to 17.04 BLEU using B=512 and constrained output space, which demonstrates the effectiveness of BART pre-training for word ordering, setting the stage for later analyses.

3.4 Errors of Unconstrained Output Space

We can readily examine the output lexical errors by comparing the input and output bag of words. As shown in Figure 3, beam search with unconstrained output space tends to miss input words rather than generate redundant words. The tendency becomes prominent when increasing the output length, accompanied by a slight drop in the output length ratio. These lexical errors explain the consistent performance drop compared to constrained output space in Table 1. See Appendix B.1 for additional results. The related coverage issue for sequence-to-sequence models has been studied in machine translation (Tu et al., 2016; Mi et al., 2016). However, different from word ordering, an error-prone source-target alignment procedure is required to estimate the output bag of words.

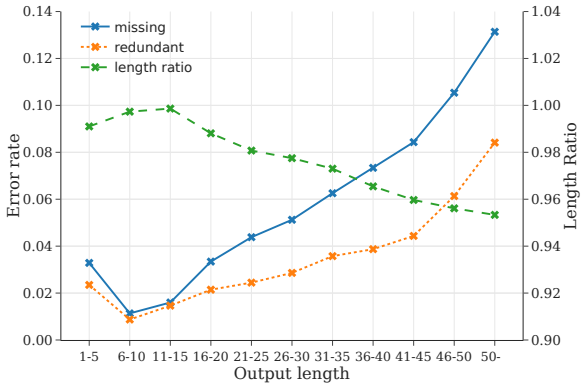

Figure 3: Test lexical errors of RAND with beam size 5 using unconstrained output space. We show the proportions of missing/redundant words and the output length ratios for test instances binned with output lengths. For the complete test set, the rates of missing and redundant words are 5.74% and 3.3%, respectively. The output length ratio is 0.981. See Figure 7 in Appendix B.1 for the results of different settings and how the metrics are computed.

3.5 Effects of Conditional Modeling

We argued in §3.1 that conditional modeling is less ambiguous when selecting the next token y_t, avoiding local minimum during beam search and thus performing well with small beams. Such a hypothesis is suggested by previous results included in Table 1: the conditional model Bag substantially outperforms (+8.9) the unconditional RNNLM with B=5; unconditional models require large beams to perform well (Schmaltz et al., 2016). We verify these observations with a *fair* comparison. Concretely, we feed a null token as the input to simulate unconditional modeling with sequence-to-sequence models[5] and train the models with the same settings in §3.2.

Results are shown in Figure 4. With small beams, conditional models substantially outperform unconditional models. Unconditional models heavily rely on large beams to perform well. In contrast, small beams perform on par with large beams for conditional models. These results verify our hypothesis. Interestingly, as the beam size further increases, RAND-uncond slightly outperforms RAND-cond, showing that a larger candidate space can address ambiguities from local modeling $p_\theta(y_t|\boldsymbol{y}_{<t})$ to some extent, at the expense of extra computation and memory overhead.

[5]The simulation leads to $P(\boldsymbol{x} = \text{null}) = 1$. Thus we have $p_\theta(\boldsymbol{y}|\boldsymbol{x} = \text{null}) = p_\theta(\boldsymbol{y})$

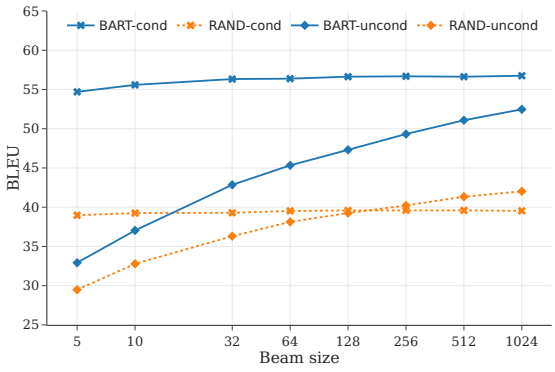

Figure 4: Test BLEU of conditional (`cond`) and unconditional (`uncond`) modeling with `RAND` and `BART`, using different beam size. A null token is fed to the encoder to simulate unconditional modeling.

3.6 Effects of Input Permutations

We empirically investigate the permutation sensitivity of our models for word ordering. The sensitivity is estimated with BLEU from 10 different development sets, each with distinct input *word* permutations. We compare our models to several controlled settings. The first is data augmentation: for each training instance, we created augmented samples with the same target output but different input word permutations, denoted `aug`. The second is a Transformer without encoder position embeddings, which is invariant to input *subword* permutations, denoted `npos`. To examine the importance of subword sequences, we also train `RAND` and `BART` with input subwords shuffled, denoted `shuf`. All models are trained with the same settings in §3.2.

Results are shown in Table 2. Both `RAND` and `BART` with the baseline setting `base` are relatively insensitive to different input word permutations (with standard deviation 0.133 and 0.185), compared to the controlled setting `npos` (with standard deviation 0.05).[6] Data augmentations `aug2-aug8` marginally improves the mean BLEU compared to `base`, but no consistent decrease of the standard deviation is observed. See Appendix B.2 for similar results with unconstrained output space. Surprisingly, with constrained output space, the lost local subword orders in `npos` and `shuf` has little impact on the performance, in contrast to the findings of Clouâtre et al. (2021). Even marginal improvement for `BART` is observed (56.45 with `shuf` compared to 56.21 with `base`). However, with unconstrained output space, the loss of local subword orders results in a non-trivial drop in performance. See Appendix B.2 for detailed results.

[6]The quantization error of float arithmetic is sensitive to the order of operands.

Settings	RAND	BART
base	40.45 (0.133)	56.21 (0.185)
aug2	40.86 (0.159)	56.78 (0.110)
aug4	40.73 (0.088)	56.97 (0.164)
aug6	40.73 (0.157)	56.76 (0.180)
aug8	40.94 (0.104)	56.91 (0.155)
npos	40.05 (0.050)	/
shuf	39.58 (0.135)	56.45 (0.133)

Table 2: Permutation sensitivity measured by the mean and standard deviation (in the bracket) of BLEU on 10 development sets with distinct input word permutations. Results are obtained with constrained output space; for unconstrained output space see Table 6 in Appendix B.2. `base` denotes settings in §3.2. `augx` is trained with *additional* `x` augmented samples per training instance. `npos` removes encoder position embeddings. `shuf` is trained with shuffled input subwords.

4 Understanding Why BART Helps

We empirically establish the intuition that knowledge in BART helps target tasks. Though numerous types of knowledge have been identified in PLMs by probing (Rogers et al., 2020), they may not necessarily improve the target task (Cífka and Bojar, 2018). Thus a reliable explanation should identify the *relevant* type of knowledge for word ordering. Candidate types of knowledge can be selected using a procedure akin to feature importance (§4.1): we feed different types of knowledge as additional features and select the one bringing the most salient gain in word ordering. Their relevance should be further verified by a strong correlation (§4.2) between the probing performance and word ordering performance, as models can utilize unexpected shortcut rules (Geirhos et al., 2020) instead of distilling the intended knowledge provided in the features.[7] Such a correlation is estimated using models with different amount of the knowledge. We finally probe for the nontrivial existence of the knowledge in BART (§4.2).

4.1 Analysis with Feature Importance

We first select a candidate type of knowledge for our explanation. Based on empirical evidence (Liu

[7]The nuance is related to the philosophical quest on what constitutes mental states. Feature importance follows behaviorism while probing is akin to functionalism. See (Levin, 2018) for discussions on behaviorism and functionalism.

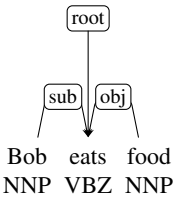

(a) Dependency tree

base	food Bob eat_ s
brac	(food) (Bob) (eat_ s)
pos	(food NNP) (Bob NNP) (eat_ s VBZ)
udep	(eat_ s (Bob) (food))
ldep	(eat_ s :sub (Bob) :obj (food))
full	(eat_ s VBZ :obj (food NNP) :sub (Bob NNP))

(b) BPE tokenized PENMMAN notation sequences

Figure 5: An example dependency tree and its BPE tokenized PENMAN notation sequences with different features.

et al., 2015) and linguistic theories (de Marneffe and Nivre, 2019), we narrow our focus to syntactic dependencies. Nevertheless, different parts of a dependency tree can be the candidate: brackets around words (brac), part-of-speech tags (pos), dependency structure (udep), labeled dependency structure (ldep), and the full tree with labeled dependency structure and part-of-speech tags (full). Knowledge are injected into models by feeding it as additional input feature. The resulting performance gain compared to the baseline (base) with bag-of-words inputs indicates the importance of the feature (Fraser et al., 2014), a *surrogate* for the relevance of the type of knowledge.

Dependency trees are derived from the PTB following (Zhang, 2013), with tags defined by Nivre et al. (2007). We use the same data split as in §3.2. Bag-of-words inputs with additional features are turned into PENMAN notation sequences (Mager et al., 2020) and fed to sequence-to-sequence Transformers as in §3.1; see Figure 5 for input examples. For tree-structured inputs, we shuffle the children of each head nodes before turning them into sequences. Dependency labels and part-of-speech tags are kept intact during BPE tokenization. We follow the same settings in §3.2 to train RAND and BART with additional input features.

Results are shown in Table 3. The additional features from different types of knowledge consistently improves word ordering, among which dependency structure (udep) brings the main performance gain (comparing udep to base, RAND is improved by 47.15 and BART by 34.37), suggesting the *potential* relevance of the knowledge to word ordering. Further adding dependency labels and part-of-speech tags marginally helps (comparing ldep and full to udep, RAND and BART are improved by up to 2.79 and 1.20, respectively). Interestingly, although part-of-speech tags alone slightly help (comparing pos to base, RAND is improved by 2.04 and BART by 1.49), their benefits diminish given dependency structures (comparing ldep to full, RAND is improved by 0.03 while BART dropping 0.27), suggesting that dependency structure knowledge can subsume part-of-speech tags. Accordingly, we select knowledge about dependency structure as our candidate for explanation.

4.2 Analysis with Structural Probing

To obtain a reliable explanation, we then establish the correlation between dependency structure knowledge in the model and word ordering performance, and verify the existence of the knowledge in BART. Both require examining dependency structure knowledge in models, which can be achieved by the structural probe (Hewitt and Manning, 2019), a learned bilinear distance metric taking representations of a model as input features. It estimates the unlabeled dependency trees of sentences using minimum spanning trees. The resulting unlabeled attachment score (UAS) indicates the probing performance. We use the average UAS of representations from all decoder layers to indicate the dependency structure knowledge in the model. We omit encoder representations since matching unordered input words to ground truth dependencies is ambiguous. Decoder representations are obtained by feeding the ground-truth outputs to the model. For each output token y_t, one can choose representations that predict it ($\mathbf{h}(\boldsymbol{y}_{<t}, \boldsymbol{x})$) or from feeding it ($\mathbf{h}(\boldsymbol{y}_{<t+1}, \boldsymbol{x})$) as features. We use the former as it results in higher UAS in our preliminary experiments. Features of words are the average of their subword features.

We follow the default hyperparameters of Hewitt and Manning (2019)[8], with a rank of 32, and train with the L1 loss for 30 epochs using 40 samples per batch. We use the derived dependency trees from §4.1 as our dataset and report the averaged

[8]We use the code provided by the authors in https://github.com/john-hewitt/structural-probes

Settings	RAND	BART	Δ_{B-R}
base	40.36	56.14	15.78
brac	40.58 $^{+0.22}$	56.64 $^{+0.50}$	16.06
pos	42.40 $^{+2.04}$	57.63 $^{+1.49}$	15.23
udep	87.51 $^{+47.15}$	90.51 $^{+34.37}$	3.00
ldep	90.27 $^{+49.91}$	91.70 $^{+35.56}$	1.43
full	90.30 $^{+49.94}$	91.43 $^{+35.29}$	1.13

Table 3: Development BLEU with different input syntactic features: brac for brackets around words, pos for part-of-speech tags, udep and ldep for unlabeled and labeled dependencies, and full for part-of-speech tags and labeled dependencies. Performance gains against bag-of-words inputs (base) are shown in the superscripts. The Δ_{B-R} column is the differences between BART and RAND.

UAS on the PTB test set. Since dependency structure knowledge can subsume part-of-speech tags as shown in §4.1, feeding features of base, pos or udep to RAND and BART results in models with varied amounts of the knowledge. Their structural probing results are shown in Table 4.

The consistent probing performance gains on feeding additional features in Table 4 confirms that knowledge is indeed injected into the models by feature feeding, ruling out the possibility that models use shortcut rules (with pos, RAND is improved by 1.26 and BART by 0.87; with udep, RAND is improved by 10.17 and BART by 12.13). Jointly examining Table 3 and Table 4, we find that an increase in UAS always corresponds to improved BLEU. The Pearson's correlation coefficient of 0.8845 between BLEU and UAS further *verifies* that dependency structure knowledge is relevant to word ordering across settings.

We finally compare the probing performance of BART initialized with pre-trained parameters (with UAS 53.06) to the agnostic setting using randomly initialized token embeddings (with 42.59 UAS).[9] The performance gap of 10.47 indicates that a non-trivial amount of dependency structure knowledge exists in BART. The relevance to word ordering and the existence in BART make knowledge about syntactic dependency structure a reliable explanation for the utility of BART in word ordering.

[9]Our preliminary experiment shows that random embeddings substantially outperform features from randomly initialized Transformer (with 30.51 UAS).

Settings	RAND	BART	Δ_{B-R}
base	44.69	53.05	8.36
pos	45.95 $^{+1.26}$	53.83 $^{+0.78}$	7.88
udep	54.86 $^{+10.17}$	65.18 $^{+12.13}$	10.32

Table 4: Averaged UAS of all decoder layers (including the embedding) on test set. base, pos and udep are the same as Table 3. Performance gains against base are shown in the superscript. The differences between BART and RAND are shown in the Δ_{B-R} column.

5 Extension to Partial Tree Linearization

Our analysis in §4 can be readily extended to partial tree linearization (Zhang, 2013), a generalized word ordering task provided with additional syntactic input features. Unlike settings in §4, additional features can be arbitrary *subsets* of part-of-speech tags and labeled dependency arcs. The task can be helpful for applications such as machine translation (Zhang et al., 2014). Following Zhang (2013), we use the same dependency trees in §4.1 and report BLEU on the PTB development set with different proportions of syntactic features.

Previous studies (Zhang, 2013; Puduppully et al., 2016) use linear models with hand-crafted features. Each base noun phrase (BNP; noun phrases without decedent noun phrases) is regarded as a single word for computation efficiency. We adopt the same sequence-to-sequence models RAND and BART in §3.2 and report results with and without special treatment for BNPs. Similar to §4.1, partial trees are turned into PENMAN sequences and fed to sequence-to-sequence Transformers. We use the same model for different proportions of input features. For each tree in the training set, we sample 0%, 50% and 100% of part-of-speech tags and labeled dependency arcs, respectively, resulting in 9 training instances with different input features for the same ordered output sequence. To keep inputs consistent, we put brackets around words that have no additional features (see brac in Figure 5).

Results are shown in Table 5. For comparison, we include results of Puduppully et al. (2016), denoted P16, and Zhang (2013), denoted Z13. We notice that treating BNPs as words substantially simplifies the task: the mean BLEU increase from 59.5 to 73.0 for RAND and from 73.7 to 82.5 for BART. RAND substantially outperforms the previous best result of P16 by 6.8 mean BLEU. In addition, BART brings further improvements by 9.5 mean BLEU, giving a new best-reported result.

Settings	(0, 0)	(0.5, 0)	(1, 0)	(0, 0.5)	(0.5, 0.5)	(1, 0.5)	(0, 1)	(0.5, 1)	(1, 1)	Mean
Z13†	42.9	43.4	44.7	50.5	51.4	52.2	73.3	74.7	76.3	56.6
P16†	48.0	49.0	51.5	59.0	62.0	67.1	82.8	86.2	89.9	66.2
RAND†	58.8	59.5	59.7	72.0	72.6	72.6	87.1	87.1	87.3	73.0
BART†	**71.7**	**71.9**	**72.3**	**83.2**	**83.1**	**83.2**	**92.1**	**92.4**	**92.2**	**82.5**
RAND	42.0	42.4	42.8	55.2	55.6	56.8	80.1	80.1	80.5	59.5
BART	**55.9**	**56.5**	**57.2**	**73.6**	**73.6**	**73.6**	**90.9**	**90.8**	**90.9**	**73.7**

Table 5: Development BLEU with constrained output space for partial tree linearization with different proportions of (pos, ldep) input features. Settings with † treat a BNP as a single word.

6 Conclusion

We investigated the role of PLMs in word ordering using BART as an instance. Non-sequential inputs are turned into sequences and fed to sequence-to-sequence Transformers and BART for coherent outputs. We achieve the best-reported results on word ordering and partial tree linearization with BART. With Transformers and BART, we provide a systematic study on the effects of output space constraints, conditional modeling, and permutation sensitivity of inputs for word ordering. Our findings can shed light on related pre-trained models such as T5 (Raffel et al., 2020) in related tasks such as CommonGen. Our analysis with feature importance and structural probing empirically identifies that knowledge about syntactic dependency structure reliably explains the utility of BART in word ordering. Such a procedure is general and can be readily used to explain why a given PLM helps a target task.

Acknowledgements

We would like to thank anonymous reviewers for their insightful comments and suggestions to help improve the paper. We gratefully acknowledge funding from the National Natural Science Foundation of China (NSFC No.61976180). Yue Zhang is the corresponding author.

References

Ernst Althaus, Nikiforos Karamanis, and Alexander Koller. 2004. Computing locally coherent discourses. In *Proceedings of the 42nd Annual Meeting of the Association for Computational Linguistics (ACL-04)*, pages 399–406, Barcelona, Spain.

Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. 2015. Neural machine translation by jointly learning to align and translate. In *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*.

Srinivas Bangalore, Owen Rambow, and Steve Whittaker. 2000. Evaluation metrics for generation. In *INLG'2000 Proceedings of the First International Conference on Natural Language Generation*, pages 1–8, Mitzpe Ramon, Israel. Association for Computational Linguistics.

Thiago Castro Ferreira, Claire Gardent, Nikolai Ilinykh, Chris van der Lee, Simon Mille, Diego Moussallem, and Anastasia Shimorina. 2020. The 2020 bilingual, bi-directional WebNLG+ shared task: Overview and evaluation results (WebNLG+ 2020). In *Proceedings of the 3rd International Workshop on Natural Language Generation from the Semantic Web (WebNLG+)*, pages 55–76, Dublin, Ireland (Virtual). Association for Computational Linguistics.

Ondřej Cífka and Ondřej Bojar. 2018. Are BLEU and meaning representation in opposition? In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1362–1371, Melbourne, Australia. Association for Computational Linguistics.

Louis Clouâtre, Prasanna Parthasarathi, Amal Zouaq, and Sarath Chandar. 2021. Demystifying neural language models' insensitivity to word-order. *ArXiv preprint*, abs/2107.13955.

Marie-Catherine de Marneffe and Joakim Nivre. 2019. Dependency grammar. *Annual Review of Linguistics*, 5(1):197–218.

Allyson Ettinger. 2020. What BERT is not: Lessons from a new suite of psycholinguistic diagnostics for language models. *Transactions of the Association for Computational Linguistics*, 8:34–48.

Hao Fang, Saurabh Gupta, Forrest N. Iandola, Rupesh Kumar Srivastava, Li Deng, Piotr Dollár, Jianfeng Gao, Xiaodong He, Margaret Mitchell, John C. Platt, C. Lawrence Zitnick, and Geoffrey Zweig. 2015. From captions to visual concepts and back. In *IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2015, Boston, MA, USA, June 7-12, 2015*, pages 1473–1482. IEEE Computer Society.

Kathleen C. Fraser, Graeme Hirst, Naida L. Graham, Jed A. Meltzer, Sandra E. Black, and Elizabeth Rochon. 2014. Comparison of different feature sets for identification of variants in progressive aphasia. In *Proceedings of the Workshop on Computational Linguistics and Clinical Psychology: From Linguistic Signal to Clinical Reality*, pages 17–26, Baltimore, Maryland, USA. Association for Computational Linguistics.

Albert Gatt and Emiel Krahmer. 2018. Survey of the state of the art in natural language generation: Core tasks, applications and evaluation. *J. Artif. Intell. Res.*, 61:65–170.

Robert Geirhos, Jörn-Henrik Jacobsen, Claudio Michaelis, Richard Zemel, Wieland Brendel, Matthias Bethge, and Felix A. Wichmann. 2020. Shortcut learning in deep neural networks. *Nature Machine Intelligence*, 2(11):665–673.

Kelvin Guu, Kenton Lee, Zora Tung, Panupong Pasupat, and Ming-Wei Chang. 2020. Realm: Retrieval-augmented language model pre-training. *arXiv:2002.08909 [cs]*. ArXiv: 2002.08909.

Eva Hasler, Felix Stahlberg, Marcus Tomalin, Adrià de Gispert, and Bill Byrne. 2017. A comparison of neural models for word ordering. In *Proceedings of the 10th International Conference on Natural Language Generation*, pages 208–212, Santiago de Compostela, Spain. Association for Computational Linguistics.

Jing He and Hongyu Liang. 2011. Word-reordering for statistical machine translation using trigram language model. In *Proceedings of 5th International Joint Conference on Natural Language Processing*, pages 1288–1293, Chiang Mai, Thailand. Asian Federation of Natural Language Processing.

Wei He, Haifeng Wang, Yuqing Guo, and Ting Liu. 2009. Dependency based Chinese sentence realization. In *Proceedings of the Joint Conference of the 47th Annual Meeting of the ACL and the 4th International Joint Conference on Natural Language Processing of the AFNLP*, pages 809–816, Suntec, Singapore. Association for Computational Linguistics.

John Hewitt and Christopher D. Manning. 2019. A structural probe for finding syntax in word representations. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4129–4138, Minneapolis, Minnesota. Association for Computational Linguistics.

Diederik P. Kingma and Jimmy Ba. 2015. Adam: A method for stochastic optimization. In *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*.

Janet Levin. 2018. Functionalism. In Edward N. Zalta, editor, *The Stanford Encyclopedia of Philosophy*, Fall 2018 edition. Metaphysics Research Lab, Stanford University.

Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Veselin Stoyanov, and Luke Zettlemoyer. 2020. BART: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7871–7880, Online. Association for Computational Linguistics.

Xiaobo Liang, Lijun Wu, Juntao Li, Yue Wang, Qi Meng, Tao Qin, Wei Chen, Min Zhang, and Tie-Yan Liu. 2021. R-drop: Regularized dropout for neural networks. *arXiv:2106.14448 [cs]*. ArXiv: 2106.14448.

Bill Yuchen Lin, Wangchunshu Zhou, Ming Shen, Pei Zhou, Chandra Bhagavatula, Yejin Choi, and Xiang Ren. 2020. CommonGen: A constrained text generation challenge for generative commonsense reasoning. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 1823–1840, Online. Association for Computational Linguistics.

Jiangming Liu and Yue Zhang. 2015. An empirical comparison between n-gram and syntactic language models for word ordering. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 369–378, Lisbon, Portugal. Association for Computational Linguistics.

Yijia Liu, Yue Zhang, Wanxiang Che, and Bing Qin. 2015. Transition-based syntactic linearization. In *Proceedings of the 2015 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 113–122, Denver, Colorado. Association for Computational Linguistics.

Manuel Mager, Ramón Fernandez Astudillo, Tahira Naseem, Md Arafat Sultan, Young-Suk Lee, Radu Florian, and Salim Roukos. 2020. GPT-too: A language-model-first approach for AMR-to-text generation. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 1846–1852, Online. Association for Computational Linguistics.

Haitao Mi, Baskaran Sankaran, Zhiguo Wang, and Abe Ittycheriah. 2016. Coverage embedding models for neural machine translation. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 955–960, Austin, Texas. Association for Computational Linguistics.

Simon Mille, Anya Belz, Bernd Bohnet, Thiago Castro Ferreira, Yvette Graham, and Leo Wanner. 2020. The third multilingual surface realisation shared task

(SR'20): Overview and evaluation results. In *Proceedings of the Third Workshop on Multilingual Surface Realisation*, pages 1–20, Barcelona, Spain (Online). Association for Computational Linguistics.

Kenton Murray and David Chiang. 2018. Correcting length bias in neural machine translation. In *Proceedings of the Third Conference on Machine Translation: Research Papers*, pages 212–223, Brussels, Belgium. Association for Computational Linguistics.

Joakim Nivre, Johan Hall, Jens Nilsson, Atanas Chanev, Gülşen Eryiágiát, Sandra Kübler, Svetoslav Marinov, and Erwin Marsi. 2007. Maltparser: A language-independent system for data-driven dependency parsing. *Natural Language Engineering*, 13(2):95–135.

Myle Ott, Sergey Edunov, Alexei Baevski, Angela Fan, Sam Gross, Nathan Ng, David Grangier, and Michael Auli. 2019. fairseq: A fast, extensible toolkit for sequence modeling. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics (Demonstrations)*, pages 48–53, Minneapolis, Minnesota. Association for Computational Linguistics.

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. Bleu: a method for automatic evaluation of machine translation. In *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics*, pages 311–318, Philadelphia, Pennsylvania, USA. Association for Computational Linguistics.

Gabriel Pereyra, George Tucker, Jan Chorowski, Lukasz Kaiser, and Geoffrey E. Hinton. 2017. Regularizing neural networks by penalizing confident output distributions. In *5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Workshop Track Proceedings*. OpenReview.net.

Ratish Puduppully, Yue Zhang, and Manish Shrivastava. 2016. Transition-based syntactic linearization with lookahead features. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 488–493, San Diego, California. Association for Computational Linguistics.

Xipeng Qiu, Tianxiang Sun, Yige Xu, Yunfan Shao, Ning Dai, and Xuanjing Huang. 2020. Pre-trained models for natural language processing: A survey. *ArXiv preprint*, abs/2003.08271.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J. Liu. 2020. Exploring the limits of transfer learning with a unified text-to-text transformer. *J. Mach. Learn. Res.*, 21:140:1–140:67.

Inioluwa Deborah Raji, Emily Denton, Emily M. Bender, Alex Hanna, and Amandalynne Paullada. 2021. AI and the everything in the whole wide world benchmark. In *Thirty-fifth Conference on Neural Information Processing Systems Datasets and Benchmarks Track (Round 2)*.

Leonardo F. R. Ribeiro, Martin Schmitt, Hinrich Schütze, and Iryna Gurevych. 2021. Investigating pretrained language models for graph-to-text generation. In *Proceedings of the 3rd Workshop on Natural Language Processing for Conversational AI*, pages 211–227, Online. Association for Computational Linguistics.

Adam Roberts, Colin Raffel, and Noam Shazeer. 2020. How much knowledge can you pack into the parameters of a language model? In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 5418–5426, Online. Association for Computational Linguistics.

Anna Rogers, Olga Kovaleva, and Anna Rumshisky. 2020. A primer in BERTology: What we know about how BERT works. *Transactions of the Association for Computational Linguistics*, 8:842–866.

Allen Schmaltz, Alexander M. Rush, and Stuart Shieber. 2016. Word ordering without syntax. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 2319–2324, Austin, Texas. Association for Computational Linguistics.

Abigail See, Peter J. Liu, and Christopher D. Manning. 2017. Get to the point: Summarization with pointer-generator networks. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1073–1083, Vancouver, Canada. Association for Computational Linguistics.

Rico Sennrich, Barry Haddow, and Alexandra Birch. 2016. Neural machine translation of rare words with subword units. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1715–1725, Berlin, Germany. Association for Computational Linguistics.

Koustuv Sinha, Robin Jia, Dieuwke Hupkes, Joelle Pineau, Adina Williams, and Douwe Kiela. 2021. Masked language modeling and the distributional hypothesis: Order word matters pre-training for little. *arXiv:2104.06644 [cs]*. ArXiv: 2104.06644.

Linfeng Song, Yue Zhang, and Daniel Gildea. 2018. Neural transition-based syntactic linearization. In *Proceedings of the 11th International Conference on Natural Language Generation*, pages 431–440, Tilburg University, The Netherlands. Association for Computational Linguistics.

Nitish Srivastava, Geoffrey E. Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. 2014. Dropout: a simple way to prevent neural networks from overfitting. *J. Mach. Learn. Res.*, 15(1):1929–1958.

Chongyang Tao, Shen Gao, Juntao Li, Yansong Feng, Dongyan Zhao, and Rui Yan. 2021. Learning to organize a bag of words into sentences with neural networks: An empirical study. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 1682–1691, Online. Association for Computational Linguistics.

Ian Tenney, Patrick Xia, Berlin Chen, Alex Wang, Adam Poliak, R. Thomas McCoy, Najoung Kim, Benjamin Van Durme, Samuel R. Bowman, Dipanjan Das, and Ellie Pavlick. 2019. What do you learn from context? probing for sentence structure in contextualized word representations. In *7th International Conference on Learning Representations, ICLR 2019, New Orleans, LA, USA, May 6-9, 2019*. OpenReview.net.

Roy Tromble and Jason Eisner. 2009. Learning linear ordering problems for better translation. In *Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing*, pages 1007–1016, Singapore. Association for Computational Linguistics.

Zhaopeng Tu, Zhengdong Lu, Yang Liu, Xiaohua Liu, and Hang Li. 2016. Modeling coverage for neural machine translation. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 76–85, Berlin, Germany. Association for Computational Linguistics.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, December 4-9, 2017, Long Beach, CA, USA*, pages 5998–6008.

Stephen Wan, Mark Dras, Robert Dale, and Cécile Paris. 2009. Improving grammaticality in statistical sentence generation: Introducing a dependency spanning tree algorithm with an argument satisfaction model. In *Proceedings of the 12th Conference of the European Chapter of the ACL (EACL 2009)*, pages 852–860, Athens, Greece. Association for Computational Linguistics.

Michael White. 2005. Designing an extensible API for integrating language modeling and realization. In *Proceedings of Workshop on Software*, pages 47–64, Ann Arbor, Michigan. Association for Computational Linguistics.

Yue Zhang. 2013. Partial-tree linearization: Generalized word ordering for text synthesis. In *IJCAI 2013, Proceedings of the 23rd International Joint Conference on Artificial Intelligence, Beijing, China, August 3-9, 2013*, pages 2232–2238. IJCAI/AAAI.

Yue Zhang, Graeme Blackwood, and Stephen Clark. 2012. Syntax-based word ordering incorporating a large-scale language model. In *Proceedings of the 13th Conference of the European Chapter of the Association for Computational Linguistics*, pages 736–746, Avignon, France. Association for Computational Linguistics.

Yue Zhang and Stephen Clark. 2011. Syntax-based grammaticality improvement using CCG and guided search. In *Proceedings of the 2011 Conference on Empirical Methods in Natural Language Processing*, pages 1147–1157, Edinburgh, Scotland, UK. Association for Computational Linguistics.

Yue Zhang and Stephen Clark. 2015. Discriminative syntax-based word ordering for text generation. *Computational Linguistics*, 41(3):503–538.

Yue Zhang, Kai Song, Linfeng Song, Jingbo Zhu, and Qun Liu. 2014. Syntactic SMT using a discriminative text generation model. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 177–182, Doha, Qatar. Association for Computational Linguistics.

Settings	RAND	BART
base	39.39 (0.151)	59.79 (0.158)
aug2	39.50 (0.118)	60.00 (0.152)
aug4	40.08 (0.091)	60.00 (0.167)
aug6	39.73 (0.053)	59.86 (0.154)
aug8	40.02 (0.152)	59.64 (0.170)
npos	36.98 (0.018)	
shuf	36.52 (0.186)	58.34 (0.124)

Table 6: Permutation sensitivity measured by the mean and standard deviation (in the bracket) of BLEU on 10 development sets with distinct input word permutations. Results are obtained with unconstrained output space. Settings are the same as Table 2.

A Beam Search with Output Constraints

The standard beam search is modify by tracking the state of a candidate with a constraint prefix tree, which specifies valid next tokens at each decoding step. The update rules for the prefix tree are described in Figure 2. See Figure 6 for an illustration of how the constraint tree for a candidate in the beam is updated during decoding.

B Results of Unconstrained Output Space

We include additional results with unconstrained output space to complement the discussion in §3.4 and §3.6.

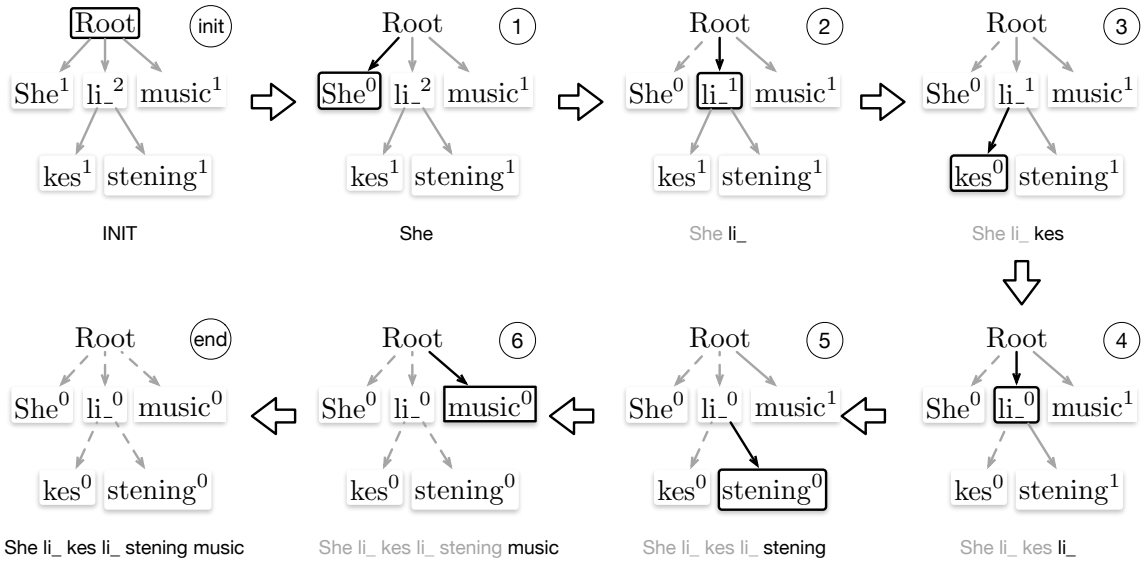

Figure 6: Illustration of how the constraint prefix tree in Figure 2 of a path is updated during decoding. We omit the state of resetting the pointer to root for brevity. At initialization, the count of each node corresponds to how many times the subword appears in the input. At step 1, as "She" is a valid subword selected by beam search, we move the pointer the node of "She" and decrement its count by 1. In the following steps, "She" becomes invalid as its count becomes zero. After step 6, since counts in all children of the root become zeros, the path is marked finished.

B.1 Lexical Errors

In addition to Figure 3 in §3.4, we present results for different models and beam sizes in Figure 7. Redundant (missing) words are all words in predicted (reference) output but not in reference (predicted) outputs. We normalize the count with number of words in all reference. Length ratio is ratio of predicted output length to reference output length.

B.2 Permutation Sensitivity

We replicate results of Table 2 in §3.6 with unconstrained output space in Table 6. Similar to Table 2, data augmentation brings marginal improvement on mean BLEU and no consistent drop in standard deviation. Unlike Table 2, the loss of subword sequences results in a nontrivial drop in performance, especially for `RAND` (-2.87 from `base` to `shuf`). `BART` is less sensitive to subword shuffling (-1.45 from `base` to `shuf`).

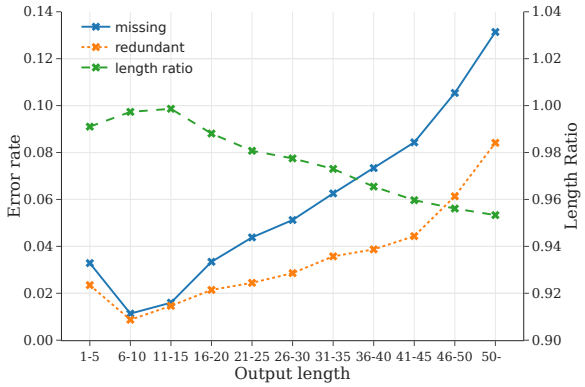

(a) `RAND,B=5`: missing 5.74%, redundant 3.30%, length ratio 0.981.

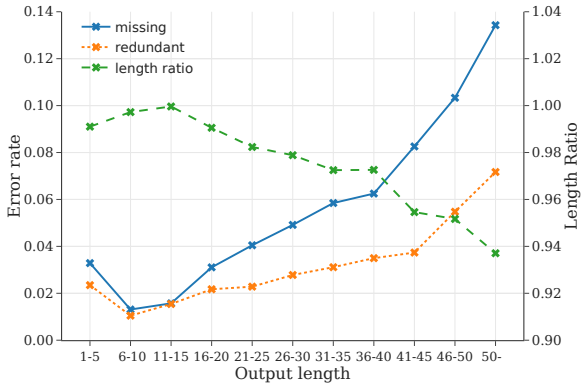

(b) `RAND,B=512`: missing 5.42%, redundant 3.02%, length ratio 0.982.

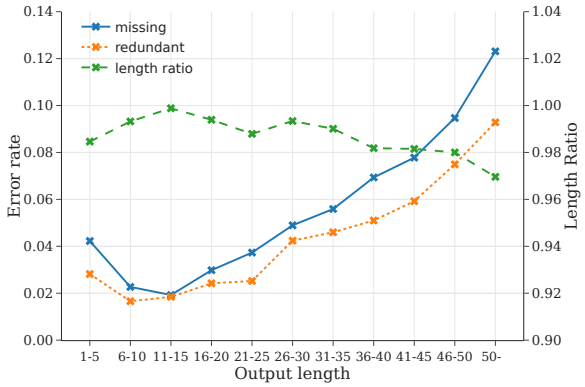

(c) `BART,B=5`: missing 5.30%, redundant 4.16%, length ratio 0.990.

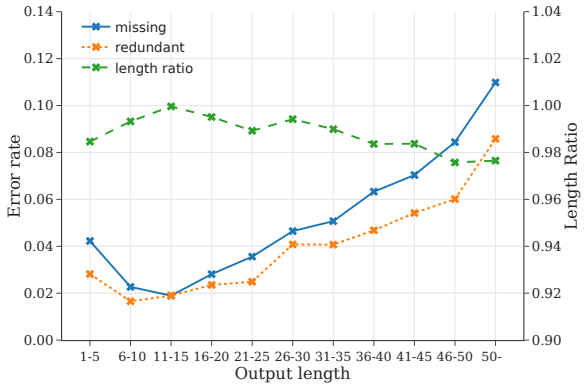

(d) `BART,B=512`: missing 4.89%, redundant 3.85%, length ratio 0.991.

Figure 7: Test lexical errors with unconstrained output space similar to Figure 3 for additional models and settings. We show the proportions of missing/redundant words and the output length ratios for test instances binned with output lengths. The model, beam size, and results on the complete test set are shown in the caption.

Visual Information Guided Zero-Shot Paraphrase Generation

Zhe Lin and **Xiaojun Wan**
Wangxuan Institute of Computer Technology, Peking University
Center for Data Science, Peking University
The MOE Key Laboratory of Computational Linguistics, Peking University
{linzhe,wanxiaojun}@pku.edu.cn

Abstract

Zero-shot paraphrase generation has drawn much attention as the large-scale high-quality paraphrase corpus is limited. Back-translation, also known as the pivot-based method, is typical to this end. Several works leverage different information as "pivot" such as language, semantic representation and so on. In this paper, we explore using visual information such as image as the "pivot" of back-translation. Different with the pipeline back-translation method, we propose **v**isual **i**nformation guided zero-shot **p**araphrase **g**eneration (**ViPG**) based only on paired image-caption data. It jointly trains an image captioning model and a paraphrasing model and leverage the image captioning model to guide the training of the paraphrasing model. Both automatic evaluation and human evaluation show our model can generate paraphrase with good relevancy, fluency and diversity, and image is a promising kind of pivot for zero-shot paraphrase generation.

1 Introduction

Paraphrase generation is a long-standing problem for natural language processing that aims to rewrite a text in other forms while preserving its original semantics. Paraphrase generation has many applications in other down-stream tasks, such as machine translation (Mehdizadeh Seraj et al., 2015), semantic parsing (Berant and Liang, 2014) and so on.

With the development of supervised seq2seq generation, most paraphrase systems depend on the large-scale aligned paraphrase corpora to train the seq2seq model. This leads to the fact that the quality of aligned corpora is extremely important for training a paraphrase system. However, high-quality paraphrase datasets are still lacking in many domains. To solve this problem, there are a few works focusing on zero-shot paraphrase generation such as back-translation. Back-translation makes use of language as pivot and treats the back-translated text as the paraphrase of the original text. For example, Mallinson et al. (2017) leveraged multilingual neural machine translation to generate paraphrase and Cai et al. (2021) proposed to employ semantic representation as the "pivot language" of back-translation to generate paraphrase. All these works show that back-translation can generate high-quality paraphrase.

(a)

(b)

Caption of Figure (a): a person on skis makes her way through the snow.

Caption of Figure (b): a person standing on skiis on the snowy slope.

Figure 1: An example that similar images may have different captions.

Inspired by back-translation based paraphrase generation, we explore to leverage visual informa-

Proceedings of the 29th International Conference on Computational Linguistics, pages 6530–6539
October 12–17, 2022.

tion (i.e. image in this study) to guide the zero-shot paraphrase generation as similar images or similar partial images may have different captions or descriptions that can be treated as paraphrases. Figure 1 shows an example. A naive method is that we can use image as the "pivot language" and generate paraphrase by back-translation with a text-to-image model and an image-to-text model. Unfortunately, text-to-image generation is still a challenging task and it is hard to generate an image of sufficient quality from the text. The semantic loss in text-to-image generation is so huge that generating paraphrases using this method is barely possible. Another method is that we can use an image captioning model to generate a caption from the image corresponding to the original text, and regard this caption as the paraphrase of the original text to train a supervised paraphrasing model. However, the generated caption may describe different elements in the image with the original text, which leads to huge semantic shift.

In this study, we propose **v**isual **i**nformation guided zero-shot **p**araphrase **g**eneration (**ViPG**), which leverages image information to guide the paraphrase generation based only on paired image-caption data. We jointly train a specific image captioning model and a paraphrasing model, and leverage the output of the image captioning model to guide the training of the paraphrasing model. This can be regarded as distilling the knowledge of the image captioning model to the paraphrasing model at the word level.

Experiment results on two datasets show our model substantially outperforms the supervised paraphrasing model trained on paired caption-caption data and it can generate valid paraphrases with high diversity. We also compare our model with other zero-shot paraphrase generation methods such as autoencoder and back-translation, and analyze the strengths and weaknesses of these methods. In all, our contributions can be summarized as follows:

- To the best of our knowledge, we are the first to explore to leverage visual information to guide zero-shot paraphrase generation.
- We propose a novel model to leverage visual information to guide zero-shot paraphrase generation. Our method jointly trains an image captioning model and a paraphrasing model, and employs this image captioning model to guide the training of the paraphrasing model. Our code is publicly available at `https://github.com/L-Zhe/ViPG`.
- Empirical studies on two image caption datasets show the effectiveness of our model and the image is demonstrated to be a promising kind of pivot for zero-shot paraphrase generation.

2 Related Works

There are several works leveraging image caption datasets like MSCOCO to train the paraphrasing model. Prakash et al. (2016) proposed residual-LSTM model to generate paraphrase. Gupta et al. (2018) found deep generative model such as variational auto-encoder can achieve better performance in paraphrase generation. Fu et al. (2019) regraded the bag of word as the latent variable of VAE to control the semantic of paraphrase. Chen et al. (2020) proposed a semantically consistent and syntactically variational encoder-decoder framework, which uses adversarial learning to ensure the syntactic latent variable be semantic-free. Cao and Wan (2020) leverage GAN to generate multiple diverse paraphrases. Lin and Wan (2021) raised multi-round paraphrase generation to improve the diversity and leveraged back-translation to maintain the semantic. All these works regard different captions of the same image as paraphrase and leverage caption-caption pairs to train paraphrasing model.

There are also some works focus on zero-shot paraphrase generation. Mallinson et al. (2017) revisited back-translation paraphrase generation with neural machine translation. Cai et al. (2021) leveraged AMR as the new pivot of back translation. Thompson and Post (2020) proposed a novel decoding strategy to generate diverse paraphrase via multilingual translation. Liu et al. (2020) leveraged simulated annealing to train unsupervised paraphrase generation model.

3 Methodology

As mentioned earlier, the pipeline back-translation with the "image pivot" can not generate valid paraphrase as the performance of the text-to-image generation model is poor. Different from the pipeline method, our proposed method jointly trains an image captioning model and a paraphrasing model, and leverages the output of the image captioning

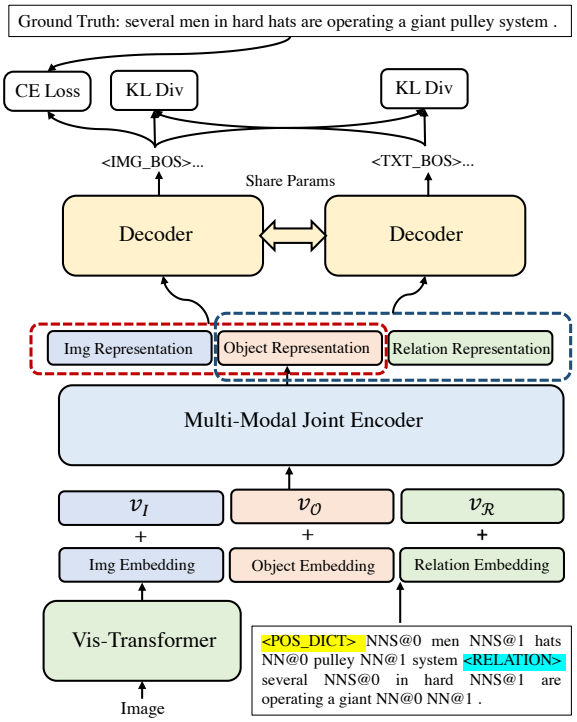

Figure 2: The overview architecture of our proposed model, which includes a multi-modal joint encoder and a parameter-sharing decoder. $v_{\mathcal{I}}$, $v_{\mathcal{O}}$, $v_{\mathcal{R}}$ are the tag vectors that indicate the different types of input.

model to guide the training of the paraphrasing model. The rationale is that an image may correspond to different captions with same meaning[1]. The image captioning model with our specific design (i.e., with additional input of object representations) may generate a caption that is different from the original caption for an input image while keeping the same meaning, and this output caption can be treated as the paraphrase of the original caption and it can be used for training the paraphrasing model. Our model relies only on image captioning dataset consisting of pairs of image and caption, and it does not need any text paraphrasing corpus and any data of caption pairs of same image. Each pair in the training dataset includes an image $\mathcal{I}$ and a corresponding caption sentence $\mathcal{S} = \{w^1, \cdots, w^N\}$, where w^i is the i-th word of the sentence and N is the sentence length.

In this section, we begin by introducing the initial embeddings of the image and text, followed by describing our multi-modal joint encoder, which employs partial attention to encode the image and the text together. Then we introduce a decoder with masked object copy mechanism to guide text generation. Finally, the objective functions will be detailed. The overall architecture of our model is shown in Figure 2.

[1]Note that the different captions provided by human judges for a same image in most existing datasets like MSCOCO are often semantically inconsistent, so we do not aim to make use of the caption pairs to train the paraphrase model in this study.

3.1 Initial Image and Text Embeddings

3.1.1 Image Embedding

For an input image $\mathcal{I}$, we first leverage Vision Transformer (Dosovitskiy et al., 2021) to encode the image into an embedding matrix $\tilde{E}_{\mathcal{I}}$ as its excellent performance in many vision tasks. We further use $v_{\mathcal{I}}$ as a tag embedding vector to indicate the image tag. After that, the initial image representation $E_{\mathcal{I}}$ is obtained as follows:

$$\begin{aligned} \tilde{E}_{\mathcal{I}} &= \mathrm{ViT}(\mathcal{I}) \\ E_{\mathcal{I}} &= \tilde{E}_{\mathcal{I}} + v_{\mathcal{I}} \end{aligned} \tag{1}$$

where ViT is the Vision Transformer encoder, $v_{\mathcal{I}} \in \mathbb{R}^d$ is the learnable parameter and $E_{\mathcal{I}} \in \mathbb{R}^{l \times d}$, where d is the feature's dimension and l is the patch length split by Vision Transformer. $+$ operation between a matrix and a vector means that the vector is added to all components of the matrix at the dimension of sequential length.

Note that we use the Vision Transformer to get $\tilde{E}_{\mathcal{I}}$ and fix it during the training of our model. This can save a bunch of training resources and has been proved to be reliable in many multi-modal tasks.

3.1.2 Text Embedding

A caption sentence can only describe the main elements of an image rather than all the details, and existing image captioning model tends to generate different captions talking about different objects for an image, which may cause semantic shift when using such image captioning model to guide the paraphrasing model. To tackle this problem, we extract the object words from the caption sentence and use them to help the image captioning model to generate more accurate and consistent captions.

Specifically, we regard nouns in a sentence as objects and the rest part of the sentence as the relation of these objects. We create the object sequence for all nouns in the sentence in this format: `{POS_TAG@index WORD}`, where `POS_TAG` is the part-of-speech of this word, `index` is used to distinguish different words of the same `POS_TAG`. We replace all nouns in the sentence with their corresponding `POS_TAG@index`. We regard the processed sequence as the relation described by the

sentence. Then we concatenate the object sequence and the relation sequence as the input text. Table 1 shows an example of the transformed input text.

Original Text: several men in hard hats are operating a giant pulley system .
Object Sequence: NNS@0 men NNS@1 hats NN@0 pulley NN@1 system
Relation Sequence: several NNS@0 in hard NNS@1 are operating a giant NN@0 NN@1 .
Transformed Input Text: <POS_DICT> NNS@0 men NNS@1 hats NN@0 pulley NN@1 system <RELATION> several NNS@0 in hard NNS@1 are operating a giant NN@0 NN@1 .

Table 1: An example about splitting a text to the object sequence and relation sequence.

We denote the embedding matrices of the object sequence and relation sequence as $\tilde{E}_{\mathcal{O}}$ and $\tilde{E}_{\mathcal{R}}$, respectively. We also add the embedding matrices with different tag embedding vectors to indicate different parts of the input information (i.e., object or relation). Finally, we combine these two parts of information as a whole and add positional encoding.

$$\begin{aligned} \hat{E}_{\mathcal{O}} &= \tilde{E}_{\mathcal{O}} + v_{\mathcal{O}} \\ \hat{E}_{\mathcal{R}} &= \tilde{E}_{\mathcal{R}} + v_{\mathcal{R}} \\ [E_{\mathcal{O}}, E_{\mathcal{R}}] &= [\hat{E}_{\mathcal{O}}, \hat{E}_{\mathcal{R}}] + W_{PE} \end{aligned} \tag{2}$$

where $v_{\mathcal{O}}, v_{\mathcal{R}} \in \mathbb{R}^d$ are learnable parameters, W_{PE} is the positional encoding matrix, $[*,*]$ is concatenation operation at the dimension of sequential length.

3.2 Multi-Modal Joint Encoder

We adopt Transformer encoder architecture as multi-modal joint encoder to further encode the image and text. In order to reduce the gap between image representation and text representation, we share the encoder parameters instead of leveraging separate encoders for image and text. We concatenate the initial image embedding $E_{\mathcal{I}}$ with the initial text embedding $[E_{\mathcal{O}}, E_{\mathcal{R}}]$ and send them to the encoder at the same time.

The powerful performance of the Transformer encoder is due to its self-attention structure, as each element in the sequence can aggregate the whole sequential information with dynamic attention weight. However, this global attention is not suitable for our model as our image captioning model and paraphrasing model should focus on different input information. Instead, we just want the image feature to focus on the information from itself and the object feature. While the image information should be ignored when encoding the text feature. Based on the rules above, we introduce the partial attention as follows:

$$\begin{aligned} \tilde{I}_i &= \mathrm{MHAttn}(I_{i-1}, [I_{i-1}, O_{i-1}], [I_{i-1}, O_{i-1}]) \\ \tilde{O}_i &= \mathrm{MHAttn}(O_{i-1}, O_{i-1}, O_{i-1}) \\ \tilde{R}_i &= \mathrm{MHAttn}(R_{i-1}, [O_{i-1}, R_{i-1}], [O_{i-1}, R_{i-1}]) \end{aligned} \tag{3}$$

where $\mathrm{MHAttn}(Q, K, V)$ is the multi-head attention (Vaswani et al., 2017), $I_{i-1}, O_{i-1}, R_{i-1}$ are the learned representation matrices of the image, object sequence and relation sequence at the $(i-1)$-th layer. $\tilde{I}_i$, $\tilde{O}_i$, and $\tilde{R}_i$ are then fed into FFN module followed by residual connection and layer normalization that are the same as the vanilla Transformer encoder to get the representation matrices at the i-th layer. We employ I, O, R to represent the encoding representations of the image, object sequence and relation sequence at the last layer respectively.

3.3 Decoder

Decoder aims to generate text from the encoding feature. Different from the text sent into the encoder which is split into the object sequence and the relation sequence, the decoder directly generates the original text $\mathcal{S}$.

Our model includes a caption decoder and a paraphrase decoder. The caption decoder generates the caption from the image feature representation and the object feature representation, and the paraphrase decoder generates the paraphrase corresponding to the original text. We share the parameters of these two decoders, and leverage different BOS tokens to guide the decoder to deal with different features. We leverage <IMG_BOS> to guide the caption generation and employ <TXT_BOS> to guide the paraphrase generation. The details of the decoder are as follows:

$$\begin{aligned} D_{\mathcal{I}} &= \mathrm{Decoder}([I, O], \ < \texttt{IMG\_BOS} >) \\ D_{\mathcal{S}} &= \mathrm{Decoder}([O, R], \ < \texttt{TXT\_BOS} >) \\ \tilde{P}_{\mathcal{I}} &= \mathrm{softmax}(W_o D_{\mathcal{I}} + b_o) \\ \tilde{P}_{\mathcal{S}} &= \mathrm{softmax}(W_o D_{\mathcal{S}} + b_o) \end{aligned} \tag{4}$$

where $\mathrm{Decoder}(\texttt{feature}, \texttt{BOStoken})$ is the Transformer decoder, W_o, b_o are learnable parameters,

which map the dimension of output features $D_{\mathcal{I}}$, and $D_{\mathcal{S}} \in \mathcal{R}^{N \times d}$ to the size of vocabulary.

We add copy mechanism (See et al., 2017) to guide the decoder to generate the correct objects. We only copy from the object sequence rather than the whole sentence. The copy probabilities are calculated as follows:

$$\begin{aligned} P^c_{\mathcal{I}} &= \text{softmax}(D^\top_{\mathcal{I}} O) \\ P^c_{\mathcal{S}} &= \text{softmax}(D^\top_{\mathcal{S}} O) \end{aligned} \tag{5}$$

Copy mechanism can improve the semantic accuracy of the generated text but may lead to low diversity of the object words. There may be more than one way to describe an object, and copying the object words from the original sentence directly can lose this diversity. Therefore we employ the masked object copy mechanism to avoid excessive copy. We randomly mask 20% object words in the object sequence as <UNK> during the copy process. This can help the model learn to generate the diverse object words rather than copy from the original sentence directly. The final output probabilities of the image caption and the paraphrase generation are denoted as $P_{\mathcal{I}} = \{p^1_{\mathcal{I}}, \cdots, p^N_{\mathcal{I}}\}$ and $P_{\mathcal{S}} = \{p^1_{\mathcal{S}}, \cdots, p^N_{\mathcal{S}}\}$, respectively.

3.4 Loss Function

We employ cross-entropy loss to train the image captioning model as follows:

$$\mathcal{L}_{ce} = -\frac{1}{N} \sum_{i=1}^{N} \log p^i_{\mathcal{I}}(w^i) \tag{6}$$

where $p^i_{\mathcal{I}}(w^i)$ is the corresponding probability of w^i in $p^i_{\mathcal{I}}$.

For the paraphrasing model, we do not directly optimize the cross-entropy loss based on $P_{\mathcal{S}}$ as this can lead to the degeneration of the model into an autoencoder. On the contrary, we align the information from the two models by reducing the gap between $P_{\mathcal{I}}$ and $P_{\mathcal{S}}$. Inspired by the R-Drop (Liang et al., 2021), we optimize the symmetric KL divergence between $P_{\mathcal{I}}$ and $P_{\mathcal{S}}$ as follows:

$$\mathcal{L}_{kl} = -\frac{1}{2N} \sum_{i=1}^{N} \{\text{KL}(p^i_{\mathcal{I}} || p^i_{\mathcal{S}}) + \text{KL}(p^i_{\mathcal{S}} || p^i_{\mathcal{I}})\} \tag{7}$$

We train image captioning and paraphrase generation together and the total loss of our model is as follows:

$$\mathcal{L} = \mathcal{L}_{ce} + \lambda \mathcal{L}_{kl} \tag{8}$$

where λ is a hyper-parameter.

3.5 Inference

Although we leverage image-caption pair to train our model, the image is not required during the inference. In the inference phase, we split the original text to the object sequence and the relation sequence and leverage <TXT_BOS> to guide the paraphrase generation.

4 Evaluation Setup

4.1 Datasets

Two image caption corpora (**MSCOCO**[2] and **Flickr30k**[3]) are used as our evaluation datasets. The MSCOCO dataset includes 118,287 images and Flickr30k includes 31,783 images, each image in both dataset has five different captions. We construct two types of training datasets for each corpus: 1) One-caption: We randomly sample one caption for each image and thus only one image-caption pair per image is used for training; 2) All-captions: We use all five captions to create five image-caption pairs per image. For each dataset, we randomly sample 4000 captions as validation dataset. For MSCOCO, we leverage all 2,5014 captions provided by the official validation dataset for test. For Flickr30k, we randomly sample 8000 captions as the test dataset. Note that there is no ground-truth paraphrase for each caption in the validation and test datasets and we do not need them in our evaluation at all[4].

4.2 Competitive Methods

The competitive methods used for comparison are mainly in three categories:

Supervised models trained with caption-caption pairs: Following previous works, we regard different captions of an image as paraphrases and leverage these caption-caption pairs to train a Transformer model as the supervised paraphrase generation model. And we also finetune the Bart model (Lewis et al., 2020) with the caption-caption

[2] https://cocodataset.org

[3] https://shannon.cs.illinois.edu/DenotationGraph

[4] We do not use datasets like Parabank and Quora for evaluation because these datasets are in totally different domains with our training datasets, and thus we use the in-domain caption data for evaluation in this study.

pairs. Besides, we take one caption in the caption-caption pair as the "reference" paraphrase of the other caption and evaluate the "reference" paraphrase as well.

AutoEncoder models with diversity decoding strategies: We train the Transformer and Bart models as the AutoEncoder models respectively. For both models, we leverage various decoding strategies including greedy search, top-k decoding and top-p decoding to generate diverse paraphrases.

Pipeline back-translation methods with various kinds of pivot: We employ language, AMR graph and image as pivots separately. For back-translation with language pivot, we leverage English-German translation systems provided by Ng et al. (2019). For back-translation with AMR pivot, we generate paraphrase according to Cai et al. (2021). For back-translation with image pivot, we leverage text-to-image model provided by Ye et al. (2021) to generate image from text and leverage image captioning model provided by Rennie et al. (2017) to generate its correspond caption as the paraphrase of the original text.

4.3 Metrics

We evaluate our model in three aspects: diversity, relevancy and fluency. We leverage **Self-BLEU**, which calculates the BLEU score between the paraphrase and the original sentence, to evaluate the diversity of the paraphrase. We leverage **BERTScore** to measure the semantic relevancy. For fluency, we employ GPT-Large without finetuning to calculate the perplexity scores (**PPL**) of different models' outputs.

In addition, we perform human evaluation of model outputs with respect to diversity, relevancy and fluency. All ratings were obtained using a five point Likert scale. We randomly sample 200 instances, including 100 from MSCOCO and 100 from Flickr30k. We employ 6 graduate students to rate each instance, and we ensure every instance is rated by at least three judges.

4.4 Training Details

We leverage Vision Transformer base[5] to generate the initial image embedding with the dimension of 768. In order to align image features, we also set the latitude of encoder and decoder to 768. We set λ to 1 in loss function. Other hyper-parameters are same to the vanilla Transformer. We select the model with highest BERTScore on the validation dataset. During inference, we leverage beam search with 5 beam size to generate paraphrase.

[5]The Vision Transformer model we used is available at https://huggingface.co/google/vit-base-patch16-224-in21k

5 Results

5.1 Result Analysis

Tables 2 and 3 show the results of automatic evaluation and human evaluation, respectively.

For supervised models trained with caption-caption pairs, the big semantic gap between the outputs of these models and the original sentence can be obvious from the low BERTScore. There are also great semantic differences between the caption reference and the original sentence. Using paired caption-caption data to train the paraphrasing model can lead to a huge semantic shift. The result of human evaluation also shows that supervised models trained with caption-caption pairs may generate paraphrase that changes the semantic of the original sentence, which can not be regarded as valid paraphrase.

For AutoEncoder models, they all get the high BERTScore but high self-BLEU, which means that the paraphrase generated by these models lacks diversity. Since Bart is a pretrained autoencoder model, top-k and top-p decoding strategies can hardly introduce diversity. For AutoEncoder, the diversity decoding strategy can indeed increase the paraphrase diversity, and yet it is harmful to the fluency of the generated sentence. The diversity decoding strategy can lead to a significant increase in PPL, this means that the quality of the generated paraphrase is affected. The human evaluation also shows the decline of sentence fluency caused by the diversity decoding strategy.

For pipeline back-translation methods, BackTranslation-AMR and BackTranslation-Language can generate good paraphrase with enough relevancy and diversity. From the human evaluation, we find that the paraphrase generated by BackTranslation-AMR has stronger diversity than BackTranslation-Language. BackTranslation-AMR can introduce diversity at syntactic level as the AMR is an abstract semantic representation of a sentence. However, BackTranslation-Image can not generate valid paraphrase with adequate semantic relevancy, this is because text-to-image generation is still a challenge task and may cause a huge semantic shift. In case study, we also show an example of BackTranslation-Image for a more

Model	MSCOCO			Flickr30k		
	Self-BLEU↓	BERTScore↑	PPL	Self-BLEU↓	BERTScore↑	PPL
Source	-	-	178.82	-	-	234.11
Supervised models trained with caption-caption pairs:						
Caption Reference	8.01	49.83	177.55	7.02	47.62	195.37
Transformer	14.81	57.30	116.96	13.00	56.31	363.15
Bart(Fine Tune)	19.61	61.29	85.15	19.46	62.33	278.37
AutoEncoder models with diversity decoding strategies:						
Bart(Original)	99.89	99.94	178.24	99.91	99.97	233.94
+ top-k(k=5)	99.82	99.90	177.61	99.74	99.89	233.16
+ top-p(p=0.9)	99.86	99.92	177.97	99.85	99.94	233.70
AutoEncoder	92.19	95.16	213.15	85.54	90.85	309.26
+ top-k(k=5)	84.30	90.55	284.53	74.17	83.88	428.08
+ top-p(p=0.9)	74.69	82.60	530.60	62.56	72.74	815.54
Pipeline back-translation methods:						
BackTranslation-AMR	36.63	75.51	353.13	32.63	75.52	430.10
BackTranslation-Language	54.17	84.17	202.05	53.87	84.75	258.28
BackTranslation-Image	9.06	51.22	104.79	4.55	45.26	81.78
ViPG(Ours):						
One-Caption	38.25	71.95	130.24	29.12	66.11	159.06
All-Captions	43.40	76.38	155.61	31.21	69.54	359.66

Table 2: Automatic evaluation results. The evaluation metrics include diversity, semantic relevancy and fluency.

Model	Rel.	Flu.	Div.	
			Lexi.	Synt.
Caption Reference	2.36	3.46	3.16	2.80
Transformer	2.81	3.40	3.31	3.03
Bart(fine tune)	2.28	3.89	3.47	3.11
AutoEncoder(top-k)	4.28	2.39	2.37	2.20
BT-Language	3.91	3.51	3.43	3.40
BT-AMR	3.54	3.39	3.20	3.88
BT-Image	1.39	3.09	2.73	2.59
ViPG(One-Caption)	3.78	3.72	3.71	3.42
ViPG(All-Captions)	3.73	3.64	3.60	3.34

Table 3: Human evaluation results. BT means Back-Translation. Rel., Flu. and Div. is the abbreviation of relevancy, fluency and diversity. Lexi. and Synt. mean lexical and syntactic, respectively.

intuitive explanation.

For our ViPG model, we solve the semantic shift in BackTranslation-Image and get the adequate BERTScore. Beside, our model performs well on diversity and fluency. Our model gets the low self-BLEU which means high diversity. For fluency, our model also achieves the best PPL score among all valid paraphrasing models. The human evaluation shows that the diversity of our model is mainly at lexical level, while syntactic diversity also performs well. Briefly, our model performs much better than other paraphrasing models leveraging image-caption data and has strong competitiveness with zero-shot paraphrasing models.

We also find that the BERTScore has a significant improvement for our ViPG model trained by all-captions dataset, but the human evaluation scores of diversity and fluency have decreased. This means that using all captions of an image to create training dataset is harmful for our model.

5.2 Ablation Study

We perform the ablation study on MSCOCO to investigate the influence of different modules in our ViPG model. We replace the transformed input text with the original text to explore the effect of embedding nouns and relations separately. We remove the $\mathrm{KL}(p^i_{\mathcal{I}}||p^i_{\mathcal{S}})$ and $\mathrm{KL}(p^i_{\mathcal{S}}||p^i_{\mathcal{I}})$ separately to show the influence of symmetric KL divergence in loss function. To further explore the effect of the masked object copy mechanism, we conduct another two experiments. One of the experiments we remove the masked object copy mechanism. In another experiment, the copy mechanism can copy the words from the whole sentence, not just the object words. Table 5 shows the results of the ablation study.

We can see that each module in our model does contribute to the overall performance. Using the original text directly can lead to significant degradation of BERTScore. The reason of huge semantic shift is that there are many objects in an image and the image-caption model can not distinguish which

Cases from MSCOCO	
Original	a cup , toothbrushes , and other items sit on the side of a small sink .
Transformer(supervised)	a bathroom sink and its reflection in the mirror .
Bart(Fine tune)	a bathroom sink with toothbrushes and other bathroom items .
AutoEncoder(top-k)	a cup , toothbrushes , and other items sit on the side of a small sink .
AutoEncoder(top-p)	a cup , toothbrushes , and other items sit on the side of a small sink .
BackTranslation-Language	a cup , toothbrushes and other objects lie on the side of a small sink .
BackTranslation-AMR	cups , toothbrushes and other items are sat on the side of the small sink .
BackTranslation-Image	a toothbrush sitting on top of a sink .
ViPG(One-Caption)	a cup contains toothbrushes and other items on the side of a sink .
ViPG(All-Captions)	a cup filled with toothbrushes and other items sitting on the side of a sink .

Cases from Flickr30k	
Original	a woman and child stand on the beach while sailboats sail on the ocean .
Transformer(supervised)	a mom and son enjoying the beach .
Bart(Fine tune)	a woman and child are standing on a beach by sailboats .
AutoEncoder(top-k)	a woman and child stand on the beach while sailboats sail on the ocean .
AutoEncoder(top-p)	a woman and child stand on the fattening beach while the ocean sail glances bieber strussel tugs installment on swatch transports palomitas the woman .
BackTranslation-Language	a woman and child stand on the beach while sailboats sail the ocean .
BackTranslation-AMR	women and children stand on the beach as boats sail in the ocean .
BackTranslation-Image	two people walking on the beach with a boat .
ViPG(One-Caption)	a woman and a child on the beach with sailboats in the ocean .
ViPG(All-Captions)	a woman and child are on the beach looking at sailboats in the ocean .

Table 4: Examples from MSCOCO and Flickr30k and the generated paraphrases by different models.

Model	Self-B↓	BS↑	PPL
Origin	38.25	71.95	130.24
Original Text	33.28	53.55	438.73
w/o KL($p^i_{\mathcal{I}}$||$p^i_{\mathcal{S}}$)	7.89	12.44	530.98
w/o KL($p^i_{\mathcal{S}}$||$p^i_{\mathcal{I}}$)	28.90	35.48	470.22
w/o Copy Mechanism	35.79	63.55	230.48
Copy the Whole Sent	61.49	77.50	203.61

Table 5: Self-B and BS is the abbreviation of self-BLEU and BERTScore.

object is described in the original text. The two-part of symmetric KL divergence is necessary for the model training. The object copy mechanism can improve the relevancy and fluency of paraphrasing. However, copying the whole sentence without restriction can lead to a lack of diversity.

5.3 Case Analysis

We perform case studies for better understanding the model performance. Table 4 shows running examples from MSCOCO and Flickr30k. Obviously, there are some degrees of semantic shift for the paraphrases generated by supervised models such as Transformer and Bart. BackTranslation-Image generates paraphrases with high semantic loss. Our ViPG model can generate paraphrases with good diversity, relevancy and fluency. However, as the shortage of image caption dataset, the paraphrases generated by our model may introduce additional semantic information, such as the "contains" and "filled with" in the example from MSCOCO. This does not affect the readability of paraphrase, but still a problem to be solved.

6 Conclusion

In this paper, we propose a visual information guided zero-shot paraphrase generation approach. We explore employing image as the "pivot" of the back-translation. Instead of using a pipeline back-translation, we jointly train an image captioning model and a paraphrasing model together. We leverage the image captioning model to guide the training of the paraphrasing model. Both automatic evaluation and human evaluation show the competitive performance of our model. In the future, we will explore huge-scale image caption dataset to train our model and test the model's ability on other domains. Moreover, leveraging video as pivot for paraphrase generation is also an interesting research direction.

Acknowledgments

This work was supported by National Key R&D Program of China (2021YFF0901502), National Science Foundation of China (No. 62161160339), State Key Laboratory of Media Convergence Production Technology and Systems and Key Laboratory of Science, Technology and Standard in Press Industry (Key Laboratory of Intelligent Press Media Technology). We appreciate the anonymous reviewers for their helpful comments. Xiaojun Wan is the corresponding author.

References

Jonathan Berant and Percy Liang. 2014. Semantic parsing via paraphrasing. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1415–1425, Baltimore, Maryland. Association for Computational Linguistics.

Yitao Cai, Yue Cao, and Xiaojun Wan. 2021. Revisiting pivot-based paraphrase generation: Language is not the only optional pivot. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 4255–4268, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Yue Cao and Xiaojun Wan. 2020. DivGAN: Towards diverse paraphrase generation via diversified generative adversarial network. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 2411–2421, Online. Association for Computational Linguistics.

Wenqing Chen, Jidong Tian, Liqiang Xiao, Hao He, and Yaohui Jin. 2020. A semantically consistent and syntactically variational encoder-decoder framework for paraphrase generation. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 1186–1198, Barcelona, Spain (Online). International Committee on Computational Linguistics.

Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, Jakob Uszkoreit, and Neil Houlsby. 2021. An image is worth 16x16 words: Transformers for image recognition at scale. In *International Conference on Learning Representations*.

Yao Fu, Yansong Feng, and John P Cunningham. 2019. Paraphrase generation with latent bag of words. In *Advances in Neural Information Processing Systems*, pages 13645–13656.

Ankush Gupta, Arvind Agarwal, Prawaan Singh, and Piyush Rai. 2018. A deep generative framework for paraphrase generation. *Proceedings of the AAAI Conference on Artificial Intelligence*.

Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Veselin Stoyanov, and Luke Zettlemoyer. 2020. BART: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7871–7880, Online. Association for Computational Linguistics.

Xiaobo Liang, Lijun Wu, Juntao Li, Yue Wang, Qi Meng, Tao Qin, Wei Chen, Min Zhang, and Tie-Yan Liu. 2021. R-drop: Regularized dropout for neural networks. In *Thirty-Fifth Conference on Neural Information Processing Systems*.

Zhe Lin and Xiaojun Wan. 2021. Pushing paraphrase away from original sentence: A multi-round paraphrase generation approach. In *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*, pages 1548–1557, Online. Association for Computational Linguistics.

Xianggen Liu, Lili Mou, Fandong Meng, Hao Zhou, Jie Zhou, and Sen Song. 2020. Unsupervised paraphrasing by simulated annealing. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 302–312, Online. Association for Computational Linguistics.

Jonathan Mallinson, Rico Sennrich, and Mirella Lapata. 2017. Paraphrasing revisited with neural machine translation. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 1, Long Papers*, pages 881–893, Valencia, Spain. Association for Computational Linguistics.

Ramtin Mehdizadeh Seraj, Maryam Siahbani, and Anoop Sarkar. 2015. Improving statistical machine translation with a multilingual paraphrase database. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 1379–1390, Lisbon, Portugal. Association for Computational Linguistics.

Nathan Ng, Kyra Yee, Alexei Baevski, Myle Ott, Michael Auli, and Sergey Edunov. 2019. Facebook fair's WMT19 news translation task submission. *CoRR*, abs/1907.06616.

Aaditya Prakash, Sadid A. Hasan, Kathy Lee, Vivek Datla, Ashequl Qadir, Joey Liu, and Oladimeji Farri. 2016. Neural paraphrase generation with stacked residual LSTM networks. In *Proceedings of COLING 2016, the 26th International Conference on Computational Linguistics: Technical Papers*, pages 2923–2934, Osaka, Japan. The COLING 2016 Organizing Committee.

Steven J Rennie, Etienne Marcheret, Youssef Mroueh, Jarret Ross, and Vaibhava Goel. 2017. Self-critical sequence training for image captioning. In *CVPR*.

Abigail See, Peter J. Liu, and Christopher D. Manning. 2017. Get to the point: Summarization with pointer-generator networks. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1073–1083, Vancouver, Canada. Association for Computational Linguistics.

Brian Thompson and Matt Post. 2020. Paraphrase generation as zero-shot multilingual translation: Disentangling semantic similarity from lexical and syntactic diversity. In *Proceedings of the Fifth Conference on Machine Translation*, pages 561–570, Online. Association for Computational Linguistics.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in neural information processing systems*, pages 5998–6008.

Hui Ye, Xiulong Yang, Martin Takác, Rajshekhar Sunderraman, and Shihao Ji. 2021. Improving text-to-image synthesis using contrastive learning. *CoRR*, abs/2107.02423.

Improving Abstractive Dialogue Summarization with Speaker-Aware Supervised Contrastive Learning

Zhichao Geng, Ming Zhong, Zhangyue Yin, Xipeng Qiu*, Xuanjing Huang
School of Computer Science, Fudan University
Key Laboratory of Intelligent Information Processing, Fudan University
{zcgeng20,mzhong18,xpqiu,xjhuang}@fudan.edu.cn

Abstract

Pre-trained models have brought remarkable success on the text summarization task. For dialogue summarization, the subdomain of text summarization, utterances are concatenated to flat text before being processed. As a result, existing summarization systems based on pre-trained models are unable to recognize the unique format of the speaker-utterance pair well in the dialogue. To investigate this issue, we conduct probing tests and manual analysis, and find that the powerful pre-trained model can not identify different speakers well in the conversation, which leads to various factual errors. Moreover, we propose three speaker-aware supervised contrastive learning (SCL) tasks: *Token-level SCL*, *Turn-level SCL*, and *Global-level SCL*. Comprehensive experiments demonstrate that our methods achieve significant performance improvement on two mainstream dialogue summarization datasets. According to detailed human evaluations, pre-trained models equipped with SCL tasks effectively generate summaries with better factual consistency.

1 Introduction

Dialogue summarization aims to condense the essential information in the dialogue into a brief text. Compared with text summarization, the conversations are semi-structured data and contain multiple participants who shall be distinguished (Gurevych and Strube, 2004; Feng et al., 2021a). Furthermore, dialogues are characterized by informal language, coreference, and repetition (Chen and Yang, 2020). All of these bring new challenges to the existing text summarization methods.

Although pre-trained models have achieved great success in Natural Language Processing especially text summarization (Liu and Lapata, 2019; Lewis et al., 2020; Qiu et al., 2020; Lin et al., 2021), how to properly utilize them in dialogues with a special speaker-utterance structure is still an obstacle. A line of previous work utilizes pre-trained models and deals with dialogue summarization as flat text. Chen and Yang (2020) segment dialogues into blocks from multiple semantic views and process them using BART. Feng et al. (2021b) use DialoGPT (Zhang et al., 2019) as an unsupervised annotator to help models understanding conversations. However, due to the gap with the pre-training object, the pre-trained models are hard to capture speaker information. To investigate these, we conduct a manual analysis on popular datasets SAMSum (Gliwa et al., 2019) and AMI (McCowan et al., 2005). We discover that, even for the state-of-the-art model BART, **55%** of the generated summaries contain factual errors for dialogues with multiple speakers. Among them, up to **56.4%** are caused directly by speaker confusion and speaker missing

Dialogue Text
Jeff: Should we go to the village party? Lia: I'm too tired after hiking. Mico: I'd like to go, there may be some hot boys! Lia: I doubt Jim: like a real village boy? Jim: who doesn't even speak English? Mico: yes, the dummer, the better. Jim: haha, stupid fucks good, they say. Mico: I confirm! Lia: not my cup of tea. Mico: I'll go there, who wants to join? Jeff: I'll go as well. Mico: wanna drive? Jeff: so you could drink? Mico: would be nice, hahah. Jeff: not excited, but ok.
Gold Summary
Mico and Jeff will go to the village party. Jeff will drive.
Baseline Summary (by BART)
Jeff, Lia and Mico are going to the village party. Lia is too tired to go. Mico will drive.
Our Summary
Jeff and Mico are going to the village party. Lia is too tired after hiking. Jeff will drive.

Table 1: A dialogue example in the SAMSum dataset. The summary generated by BART has two factual errors: Lia is not going to the village party; it will be Jeff driving instead of Mico. Our model can generate factually correct summaries.

*Corresponding author

Proceedings of the 29th International Conference on Computational Linguistics, pages 6540–6546
October 12–17, 2022.

(see Section 3.2). As shown in Table 1, the model's inability to identify speakers results in serious factual inconsistencies.

Another tributary of previous work (Zhao et al., 2019; Liu and Chen, 2019; Zhu et al., 2020; Lei et al., 2021) utilizes the hierarchical network instead of pre-trained models to leverage the dialogue's structural information. However, how to explicitly model the information of speakers in pre-trained sequence-to-sequence (seq2seq) models remains unsolved. Zhu et al. (2020) introduces speaker embedding to distinguish speakers for meetings with fixed participants. However, in most cases, the number and identity of the participants in the conversations are unknown. Thus the trained embedding is not a general solution.

Intuitively, if the representation derived from the encoder has sufficient information to identify speakers, the decoder will produce superior summaries, especially for summaries that follow a pattern of *someone does something* as shown in Table 1. In this paper, we first conduct a probing experiment to show that the representation of the dialogue obtained from BART can not distinguish speakers well. To address this issue, we use contrastive learning to improve the alignment of the representation derived from the encoder, i.e., to make the encoder output diverse hidden states based on corresponding speakers. We propose three speaker-aware supervised contrastive learning tasks: *Token-level SCL*, *Turn-level SCL*, and *Global-level SCL*. By jointly training these tasks in the fine-tuning stage, we can substantially improve the model's ability to identify different speakers and further understand the content of the whole dialogue. Comprehensive experiments and human evaluations on SAMSum and AMI (McCowan et al., 2005) reveal that our models generate summaries with higher ROUGE scores and better factual consistency. Our main contributions include (a) this is the first work to give a detailed investigation of the speaker identification problem in dialogue summarization, (b) proposing speaker-aware SCL tasks to address the problem, and evaluating our methods with the experimental and manual examination.

2 Method

2.1 Probing Test

To investigate how well pre-trained seq2seq models can distinguish speakers, we conduct a simple probing experiment on SAMSum, a widely-used dialogue summarization corpus. Concretely, we first encode the integral dialogue text with the BART (Lewis et al., 2020) encoder and randomly sample K tokens to obtain their hidden states. Then, in pairs, we aggregate and feed these hidden states into MLP to determine whether they are from the same speaker. To investigate the compatibility of the pre-trained seq2seq model with flat dialogue text, we freeze the parameters of the BART encoder and solely fine-tune the MLP classification layers during training stage. Then we evaluate the classification accuracy on the test set. Intuitively, the binary classification task is easy, but the accuracy is only **58.1%** for vanilla BART. Even after fine-tuning BART with the summarization task (parameters of BART are not frozen) on SAMSum before the probing test , the accuracy is still only **60.2%**[1]. The result indicates that pre-trained seq2seq models can not identify speakers well from flat dialogue text, and simply fine-tuning with the dialogue summarizing task is unhelpful. We need more explicit methods to help models understand flat dialogue text.

2.2 Supervised Contrastive Learning Tasks

To address the above problem, inspired by the research about contrastive learning (Mikolov et al., 2013; Saunshi et al., 2019; He et al., 2020; Velickovic et al., 2019), we introduce SCL tasks to help models identify speakers. The model is supposed to minimize the distance between representations of utterances from the same speaker and vice versa by optimizing the SCL loss during the fine-tuning stage.

Regular Paradigm for Dialogue Summarization Formally, a dialogue $D = (t_1, t_2, \cdots, t_n)$ consists of n turns, and each turn t_i contains the utterance u_i and the corresponding speaker s_i, that is, $t_i = (s_i, u_i)$. Firstly, we use Transformer-Encoder (Vaswani et al., 2017) to model the dialogue-level contextual representation of each tokens from the flat dialogue text.

$$H = \text{Transformer-Encoder}(D), \tag{1}$$

where the input sequence is the concatenation of all turns. Then, we can generate the summary $\hat{y}$ with Transformer-Decoder. The generation loss $\mathcal{L}_{gen}$ is cross-entropy loss between $\hat{y}$ and gold summary y.

[1]By jointly training the Global-level SCL task in fine-tuning stage, the accuracy reaches **77.9%**.

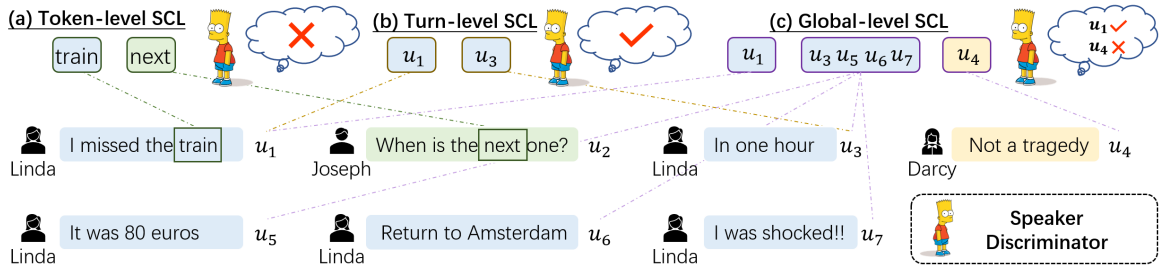

Figure 1: Overview of our speak-aware SCL tasks. *Token-level SCL* and *Turn-level SCL* mean the model needs to discriminate whether two tokens/turns are from the same speaker. *Global-level SCL* let the model choose what the speaker might say in a particular turn when given all the utterances of this speaker. The representations are obtained by inputting the whole dialogue into the encoder.

Incorporating Contrastive Loss To enable the utterance representation to contain more speaker information, we incorporate three levels of contrastive losses into the regular fine-tuning paradigm.

Generally, let (o_i, s_i) denote a sampled token or utterance and the associative speaker. The contrastive loss $\mathcal{L}_{ctr}$ for the SCL task is calculated as follows:

$$\mathcal{L}_+ = \sum_{i,j}^{s_i=s_j} -\log(\sigma(\mathbf{o}_i \cdot \mathbf{o}_j)), \quad (2)$$

$$\mathcal{L}_- = \sum_{i,j}^{s_i\neq s_j} -\log(1-\sigma(\mathbf{o}_i \cdot \mathbf{o}_j)), \quad (3)$$

$$\mathcal{L}_{ctr} = \mathcal{L}_+ + \mathcal{L}_-, \quad (4)$$

where σ is logistic function that measures the similarity between two representations and $\mathbf{o}_i$ is the contextual representation of o_i derived from H. The detailed sampling methods of o_i are dicussed in Section 2.2.1 $\sim$ 2.2.3.

The final loss $\mathcal{L} = \lambda\mathcal{L}_{ctr} + \mathcal{L}_{gen}$ and λ is the weight coefficient to adjust the ratio of $\mathcal{L}_{ctr}$ and $\mathcal{L}_{gen}$ in the final loss $\mathcal{L}$. The model is supposed to maximize similarity among samples of the same speaker and vice versa while being optimized for the summary generation.

Next, we introduce our proposed supervised contrastive tasks in detail.

2.2.1 Token-level SCL

The first task is the Token-level SCL which means the model distinguishes whether two tokens are from the same speaker. As illustrated in Figure 1(a), we randomly sample m token-speaker pairs $T = \{(o_1, s_1), (o_2, s_2), ..., (o_m, s_m)\}$ from D, where o_i is a token and s_i is the corresponding speaker. The hidden state of o_i obtained through the encoder is used to represent the i-th sample.

2.2.2 Turn-level SCL

Compared with Token-level SCL, we increase the granularity of the input to fuse the semantic information of the context. As shown in Figure 1(b), we randomly sample two turns from D and mask the speaker names in text, denoted as (o_i, s_i) and (o_j, s_j). Then we derive $\mathbf{o}_i$ by taking the mean pooling of the hidden states of all tokens in o_i.

2.2.3 Global-level SCL

To maximize the mutual information between utterances of the same speaker (Linsker, 1988; Kong et al., 2019), we extend the Turn-level SCL task to Global-level SCL by introducing global information. Intuitively, we can understand the speaking style of a specific person from all the words he or she has said. Therefore, we provide the model with all the utterances of a certain speaker and let it choose what this speaker might say in a particular turn (described in Figure 1(c)). Concretely, we first mask all the speaker names and randomly sample a speaker whose utterances set $\tilde{S}_i$ which has at least two elements. Among $\tilde{S}_i$, we randomly choose a utterance (o_i, s_i) as the positive sample, and randomly choose another utterance (o_j, s_j) from $D - \tilde{S}_i$ as the negative sample. Thus the global utterance sample of this speaker is $(\tilde{S}_i - o_i, s_i)$. The model is supposed to maximize the mutual information between the representation of the global sample and the positive sample, and vice versa. The representations are derived from mean pooling, the same as what we do in Turn-level SCL. In contrast to Turn-level SCL, Global-level SCL needs the model's overall comprehension of the dialogue-format context.

Model	SAMSum			AMI		
	R-1	R-2	R-L	R-1	R-2	R-L
PGNet (See et al., 2017)	40.08	15.28	36.63	42.60	14.01	22.62
UniLM (Dong et al., 2019; Zhu et al., 2021)	50.00	26.03	42.34	50.61	**19.33**	25.06
Multi-view BART (Chen and Yang, 2020)	53.42	27.98	49.97	-	-	-
BART+DialoGPT (Feng et al., 2021b)	53.70	28.79	50.81	-	-	-
PGN+DialoGPT (Feng et al., 2021b)	-	-	-	50.91	17.75	24.59
BART	53.01	28.05	49.89	50.67	17.18	24.96
BART + Token-level SCL task	53.85	29.21	50.94	51.03	17.23	25.21
BART + Turn-level SCL task	54.12	29.53	51.10	51.15	17.85	**25.45**
BART + Global-level SCL task	**54.22**	**29.87**	**51.35**	**51.40**	17.81	25.30

Table 2: Results on the test sets of SAMSum and AMI, and "R" is short for "ROUGE". Our results are significantly better than the baseline model ($p < 0.05$).

3 Experiment

In this section, we conduct experiments and human evaluations on the popular datasets SAMSum and AMI. More descriptions of the datasets and the implementation details can be found in the Appendix.

3.1 Experimental Result and Analysis

We provide several latest strong seq2seq models as baselines, including PGNet (See et al., 2017), UniLM (Dong et al., 2019) and BART+DialoGPT (Feng et al., 2021b) in the first part of Table 2. Following previous settings (Gliwa et al., 2019; Feng et al., 2021a), we use *py-rouge*[2] package for evaluation on SAMsum and use *pyrouge*[3] on AMI. Experimentally, our models obtain clear improvement on both two datasets compared to the BART baseline, and achieve the state-of-the-art result on SAMSum.

Specific to the three tasks, the improvement brought by *Token-level SCL* is relatively tiny. The reason may be that the utilization of positional information is enough for BART to optimize the contrastive loss for two tokens. For *Turn-level SCL* and *Global-level SCL*, the pooling layer reduces the impact of position embedding, thereby forcing the model to focus on the semantic information of the utterances. Therefore, the model can further capture the characteristics of the dialogue data. *Global-level SCL* performs best in both datasets, which illustrates that when the model has global perspectives for each speaker, it can enhance the model's comprehension of the whole dialogue.

[2] https://pypi.org/project/py-rouge/

[3] https://github.com/bheinzerling/pyrouge

3.2 Human Evaluation

We also conduct human evaluations to investigate if our method leads to fewer factual errors. Automatic metrics like FACTCC (Kryściński et al., 2019) are not used since the neural-model-based metrics perform poorly in dialogue data due to the significant domain gap. And most of the factual errors in the dialogue summarization are caused by misidentification of the speaker, which can not be reflected by automatic metrics. Here we use BART and BART with the Global-level SCL task for comparison.

Error Types Firstly, we divide the factual errors into three categories manually:(a) **Speaker Confusion**: Model confuses speakers participating in a specific event; (b) **Speaker Missing**: A speaker is mentioned in the gold summary, while the model hits the event but misses this speaker. (c) **Semantic Error**: Errors caused by a misunderstanding of semantics, and they are not directly related to any speakers. More cases about the error types can be found in Appendix.

Model	BART	BART + Global SCL
Speaker Confusion Rate	0.100	0.067
Speaker Missing Rate	0.267	0.167
Semantic Errors Rate	0.283	0.242

Table 3: The rate of factual errors for the baseline model and our model on SAMSum and AMI. Please note that multiple types of factual errors can occur in a single data sample.

Result We evaluate 100 dialogues from the test set of SAMsum and all 20 dialogues of the AMI test set. For the SAMSum dataset, to explore the model's ability to understand multi-person interaction, we choose all 64 dialogues with more than

three speakers and randomly choose 36 dialogues with three speakers. The result is shown in Table 3. **55%** of the summaries generated by BART contain factual errors, of which **56.4%** are related to the Speaker Confusion or Speaker Missing. In comparison, our model decreases the number of speaker-related factual errors by **36.4**%. The SCL task helps the model to better distinguish the speakers and intuitively reduce the confusion. Furthermore, the SCL task helps the model to better perceive the speakers and reduce the Speaker Missing errors.

4 Conclusion

In this paper, we focus on the speaker identification problem in the dialogue summarization task. Through the probing test and manual analysis, we find that the existing pre-trained model can not identify different speakers well in the conversation, leading to factual errors. Therefore, we propose three speaker-aware SCL tasks to address this problem. Experimental results and human evaluations illustrate the effectiveness of our methods.

5 Ethical Considerations

For human evaluation in section 3.2, we recruited two annotators to see if there are any factual inconsistencies in generated summaries. The generators of all summaries are hidden from the annotators to avoid any subjective bias. For the SAMsum dataset, we give more priority to dialogues with more speakers and adopt a random strategy when the numbers of speakers are same.

Acknowledgements

This work was supported by the National Key Research and Development Program of China (No. 2020AAA0108702) and National Natural Science Foundation of China (No. 62022027).

References

Jiaao Chen and Diyi Yang. 2020. Multi-view sequence-to-sequence models with conversational structure for abstractive dialogue summarization. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 4106–4118, Online. Association for Computational Linguistics.

Li Dong, Nan Yang, Wenhui Wang, Furu Wei, Xiaodong Liu, Yu Wang, Jianfeng Gao, Ming Zhou, and Hsiao-Wuen Hon. 2019. Unified language model pre-training for natural language understanding and generation. *arXiv preprint arXiv:1905.03197*.

Xiachong Feng, Xiaocheng Feng, and Bing Qin. 2021a. A survey on dialogue summarization: Recent advances and new frontiers. *arXiv preprint arXiv:2107.03175*.

Xiachong Feng, Xiaocheng Feng, Libo Qin, Bing Qin, and Ting Liu. 2021b. Language model as an annotator: Exploring dialogpt for dialogue summarization. *arXiv preprint arXiv:2105.12544*.

Bogdan Gliwa, Iwona Mochol, Maciej Biesek, and Aleksander Wawer. 2019. SAMSum corpus: A human-annotated dialogue dataset for abstractive summarization. In *Proceedings of the 2nd Workshop on New Frontiers in Summarization*, pages 70–79, Hong Kong, China. Association for Computational Linguistics.

Iryna Gurevych and Michael Strube. 2004. Semantic similarity applied to spoken dialogue summarization. In *COLING 2004: Proceedings of the 20th International Conference on Computational Linguistics*, pages 764–770.

Kaiming He, Haoqi Fan, Yuxin Wu, Saining Xie, and Ross Girshick. 2020. Momentum contrast for unsupervised visual representation learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 9729–9738.

Lingpeng Kong, Cyprien de Masson d'Autume, Wang Ling, Lei Yu, Zihang Dai, and Dani Yogatama. 2019. A mutual information maximization perspective of language representation learning. *arXiv preprint arXiv:1910.08350*.

Wojciech Kryściński, Bryan McCann, Caiming Xiong, and Richard Socher. 2019. Evaluating the factual consistency of abstractive text summarization. *arXiv preprint arXiv:1910.12840*.

Yuejie Lei, Yuanmeng Yan, Zhiyuan Zeng, Keqing He, Ximing Zhang, and Weiran Xu. 2021. Hierarchical speaker-aware sequence-to-sequence model for dialogue summarization. In *ICASSP 2021-2021 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 7823–7827. IEEE.

Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Veselin Stoyanov, and Luke Zettlemoyer. 2020. BART: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7871–7880, Online. Association for Computational Linguistics.

Chin-Yew Lin and Franz Josef Och. 2004. Automatic evaluation of machine translation quality using longest common subsequence and skip-bigram

statistics. In *Proceedings of the 42nd Annual Meeting of the Association for Computational Linguistics (ACL-04)*, pages 605–612.

Tianyang Lin, Yuxin Wang, Xiangyang Liu, and Xipeng Qiu. 2021. A survey of transformers. *arXiv preprint arXiv:2106.04554*.

Ralph Linsker. 1988. Self-organization in a perceptual network. *Computer*, 21(3):105–117.

Yang Liu and Mirella Lapata. 2019. Text summarization with pretrained encoders. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 3721–3731.

Zhengyuan Liu and Nancy Chen. 2019. Reading turn by turn: Hierarchical attention architecture for spoken dialogue comprehension. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 5460–5466.

Iain McCowan, Jean Carletta, Wessel Kraaij, Simone Ashby, S Bourban, M Flynn, M Guillemot, Thomas Hain, J Kadlec, Vasilis Karaiskos, et al. 2005. The ami meeting corpus. In *Proceedings of the 5th International Conference on Methods and Techniques in Behavioral Research*, volume 88, page 100. Citeseer.

Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. 2013. Distributed representations of words and phrases and their compositionality. In *Advances in neural information processing systems*, pages 3111–3119.

Xipeng Qiu, TianXiang Sun, Yige Xu, Yunfan Shao, Ning Dai, and Xuanjing Huang. 2020. Pre-trained models for natural language processing: A survey. *SCIENCE CHINA Technological Sciences*, 63(10):1872–1897.

Nikunj Saunshi, Orestis Plevrakis, Sanjeev Arora, Mikhail Khodak, and Hrishikesh Khandeparkar. 2019. A theoretical analysis of contrastive unsupervised representation learning. In *International Conference on Machine Learning*, pages 5628–5637. PMLR.

Abigail See, Peter J. Liu, and Christopher D. Manning. 2017. Get to the point: Summarization with pointer-generator networks. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1073–1083, Vancouver, Canada. Association for Computational Linguistics.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Lukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. *arXiv preprint arXiv:1706.03762*.

Petar Velickovic, William Fedus, William L Hamilton, Pietro Liò, Yoshua Bengio, and R Devon Hjelm. 2019. Deep graph infomax. In *ICLR (Poster)*.

Yizhe Zhang, Siqi Sun, Michel Galley, Yen-Chun Chen, Chris Brockett, Xiang Gao, Jianfeng Gao, Jingjing Liu, and Bill Dolan. 2019. Dialogpt: Large-scale generative pre-training for conversational response generation. *arXiv preprint arXiv:1911.00536*.

Zhou Zhao, Haojie Pan, Changjie Fan, Yan Liu, Linlin Li, Min Yang, and Deng Cai. 2019. Abstractive meeting summarization via hierarchical adaptive segmental network learning. In *The World Wide Web Conference*, pages 3455–3461.

Chenguang Zhu, Yang Liu, Jie Mei, and Michael Zeng. 2021. Mediasum: A large-scale media interview dataset for dialogue summarization. *arXiv preprint arXiv:2103.06410*.

Chenguang Zhu, Ruochen Xu, Michael Zeng, and Xuedong Huang. 2020. A hierarchical network for abstractive meeting summarization with cross-domain pretraining. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 194–203, Online. Association for Computational Linguistics.

A Datasets

We apply our methods on the large version of BART and evaluate our model on SAMSum and AMI datasets using ROUGE score (Lin and Och, 2004). SAMSum consists of 16,369 samples with an average of 2.4 participants and 83.9 words. AMI consists of 137 meeting records of four fixed speakers, which have 4,757 words on average. Due to the limitation of our computing resources, all our inputs are truncated to 1,024 tokens. We use the same split as Gliwa et al. (2019) and Zhu et al. (2020) for SAMSum and AMI, respectively.

B Implementation Details

Hyperparameters	SAMSum	AMI
Batch Size	8	24
Total Steps	10,000	600
Eval Steps	1,000	20
Learning Rate	[2e-5,3e-5]	[2e-5,3e-5]
Label Smoothing Factor	0.1	0
Warm-up Type	linear	linear
Warm-up Steps	0	100
Max Target Length	128	300

Table 4: Hyperparameters we used for fine-tuning BART on SAMSum and AMI.

Some of our hyperparameters are listed in Table 4. Other hyperparameters are the same as the default of *facebook/bart-large* of transformers[4].

[4] https://github.com/huggingface/transformers

The weight coefficient factor λ is searched from $\{0.01, 0.001\}$. It takes up to 2 hours for one run on SAMSum or AMI using one GeForce RTX 3090.

We use the validation set to select the best checkpoint, and evaluate the checkpoint on the test set.

C Case Study

In order to better illustrate the three types of errors mentioned in Section 3.2, we provide more cases here. An example of confusing speakers is shown in Table 1 of the main paper. Examples of missing speakers and semantic errors are shown in Table 5.

Dialogue Text 1
Ann: Congratulations!! Ann: You did great, both of you! Sue: Thanks, Ann Julie: I'm glad it's over! Julie: That's co cute of you, girl! Ann: Let's have a little celebration tonight! Sue: I'm in Julie: me too!!! aww
Gold Summary 1
Ann, Sue and Julie did a great job and they will have a little celebration tonight.
Baseline Summary 1 by BART
Sue and Julie are going to celebrate their success tonight.
Our Summary 1
Ann, Sue and Julie are celebrating their wins.
Dialogue Text 2
Sarah: omg Laura! sorry you didn't get any replies!!! Did you manage? Laura: hahaha! Awkssss... no worries, I solved it Sarah: awkward silence <crickets> Laura: hahaha no it's all good really!! Raf: Laura, I'm so sorry!!! been so swamped, totally forgot to text you back! where are you?? Sarah: Exotic little island called Linate :D Laura: Sarah which hotel are you at??? I'm here too!!!
Gold Summary 2
Neither Raf nor Sarah remembered to reply to Laura but she managed anyway. Both Sarah and Laura are in Linate.
Baseline Summary 2 by BART
Laura didn't get any replies to Sarah's messages.Laura is on an island called Linate. Laura and Sarah are staying at the same hotel.
Our Summary 2
Laura didn't get any replies from Sarah and Raf. Sarah and Laura are on an exotic little island called Linate.

Table 5: Sample 1 is an example about the speaker missing error. The summary generated by BART misses Ann. The dialogue sample is from the SAMSum dataset. Sample 2 is an example about the semantic error and the speaker missing error. The summary generated by BART misses Raf (Speaker Missing Error), and makes it out of thin air that Sarah and Laura are staying at the same hotel (Semantic Error). All samples are from the SAMsum dataset.

Diversifying Neural Text Generation with Part-of-Speech Guided Softmax and Sampling

Zhixian Yang and **Pengxuan Xu** and **Xiaojun Wan**
Wangxuan Institute of Computer Technology, Peking University
Center for Data Science, Peking University
The MOE Key Laboratory of Computational Linguistics, Peking University
yangzhixian@stu.pku.edu.cn
xupengxuan@stu.pku.edu.cn
wanxiaojun@pku.edu.cn

Abstract

Neural text generation models are likely to suffer from the low-diversity problem. Various decoding strategies and training-based methods have been proposed to promote diversity only by exploiting contextual features, but rarely do they consider incorporating syntactic structure clues. In this work, we propose using linguistic annotation, i.e., part-of-speech (POS), to guide the text generation. In detail, we introduce POS Guided Softmax to explicitly model two posterior probabilities: (i) next-POS, and (ii) next-token from the vocabulary of the target POS. A POS Guided Sampling strategy is further proposed to address the low-diversity problem by enriching the diversity of POS. Extensive experiments and human evaluations show that, compared with existing state-of-the-art methods, our POS Guided Softmax and Sampling (POSG) can generate more diverse text while maintaining comparable quality.[1]

1 Introduction

Maximum likelihood estimation (MLE) is a standard approach to training a neural text generation model, e.g. Transformer (Vaswani et al., 2017), to generate human-like text. However, existing generation systems often suffer from the low-diversity problem (Holtzman et al., 2020; Welleck et al., 2020), which leads to dull and repetitive generations. This problem unavoidably affects the overall generation quality.

We conclude that the low-diversity problem is mainly manifested in two aspects: *form* and *content* (Fu et al., 2021; Holtzman et al., 2020; Tevet and Berant, 2021). As shown Table 1, the low form diversity can be reflected in repeating some words, using similar lexicon and syntax, and more. The low content diversity can be expressed as a single and dull content with nothing different.

Several feasible fixes have been proposed, such as post-hoc sampling strategies including temperature (Caccia et al., 2020), top-k (Fan et al., 2018), and nucleus sampling (Holtzman et al., 2020). Recently, some works suggest that it is the maximizing likelihood itself that should account for the low-diversity problem (Holtzman et al., 2020; Welleck et al., 2020). Holtzman et al. (2020) think that MLE can not adequately capture the rich diversity and expression in human language. Choi et al. (2020) argue that the imbalanced token distribution inherent in natural language even worsens the low-diversity problem. Based on these analysis, many training-based methods have been proposed. Welleck et al. (2020) propose the unlikelihood training to penalize repetition with auxiliary losses. Jiang et al. (2019) propose to utilize dynamically scaling losses conditioned on the token frequency in the training phase. Choi et al. (2020) factorize the probability distribution and design an elaborate token cluster algorithm for a balanced training.

Though those encouraging progress has been made, we argue that current training-based methods only take plain contextual features to promote diversity, rarely considering incorporating syntactic structure clues. For example, when humans are writing articles, it is natural to predetermine the part-of-speech (POS) before giving the next token. Existing studies have verified that incorporating POS can improve the translation quality in neural machine translation (NMT) (Sennrich and Haddow, 2016a; Yang et al., 2021). Intuitively, since the vocabularies of different POS vary a lot, the diversity of POS will certainly lead to the diversity of text. Unfortunately, we observe that existing methods with no consideration of the inner POS structure fail to learn the diversity of POS in human language (shown in Table 3).

All these factors motivate us to address the low-diversity problem with the guidance of POS. Thus, in this work, we first present the POS Guided Soft-

[1]Our code is available at https://github.com/FadedCosine/POS-Guided-Neural-Text-Generation

Proceedings of the 29th International Conference on Computational Linguistics, pages 6547–6563
October 12–17, 2022.

Context: The NK 2nd Division , concentrated in the Sinban-ni area west of the river , had , in effect , attacked straight east across the river and was trying to seize the two avenues of advance into Changnyong.
Text 1: They were joined by the 27th Battalion, US 24th Infantry Regiment, the 27th Regiment, and the 27th Regiment. The US 24th Infantry Division, under the command of Major General John R. Dempsey, was ordered to charge the US 24th Infantry Division, and capture it from the west.
Text 2: The NK 2nd Division, which had been involved in the assault, was forced to withdraw from the area north of the river. The NK 3rd Division, which had been fighting in the area since the beginning of the battle, was moved to the south. The NK 2nd Division, which had been fighting in the area since the start of the battle, had been pushed back to the south.
Text 3: The 2nd Division had been moving north from Alcester's position on the road, but were not expecting another attack. In the immediate aftermath of the attack, to keep the 2nd Division in reserve, which had been preparing for an attack on Hill 131. Along with the 3rd Battalion of the US 2nd Infantry Regiment, attacked Hill 129 at Pakchon on the way to Beaulieu.

Table 1: Examples of low-diversity generated text, given context from the Wikitext-103 dataset (Merity et al., 2017). Text 1 has a poor form diversity due to many useless repeating words (highlighted in blue). Text 2 keeps talking about only one single content, with similar lexicon and syntax (highlighted in orange), indicating low diversity in both terms of form and content. Though Text 3 has various syntactical and lexical forms with no repetition, all the content of it is about the "attacks", which means low content diversity. Text 1 is sampled from MLE, Text 2 from F^2-Softmax (Choi et al., 2020), and Text 3 from FACE (Jiang et al., 2019) (Section 5.1).

max (Figure 1), building upon a hybrid decoder that predicts two posterior probabilities: (i) next-POS, and (ii) next-token from the vocabulary of the target POS. Our work shows that, following the POS clue, our model can gain a deeper insight into text's syntactic structure. Thereafter, we propose a POS Guided Sampling to improve the diversity of generated text lexically and syntactically while maintaining comparable quality.

To sum up, the contributions of our work are three-fold. (i) We introduce a novel POS Guided Softmax, incorporating POS tags as the observed discrete decisions to improve text generation. (ii) Based on POS Guided Softmax, POS Guided Sampling is proposed to promote text diversity effectively without degrading quality. (iii) We conduct extensive experiments on language modeling and paraphrase generation. Experimental results and human evaluation show that our model can easily adapt to different downstream tasks and generate text with high diversity as well as quality.

2 Related Works

2.1 Diversity-promoting Methods

Decoding-based Methods. Although greedy search and beam search are well known decoding strategies for neural text generation, Holtzman et al. (2020) have shown that these methods always generate generic, repetitive, and awkward words. Kulikov and Cho (2019) and Vijayakumar et al. (2018) have proposed several variants of beam search as alternatives. Recently, stochastic decoding methods have been widely used, and some studies propose to sample from a truncated and renormalized Softmax distribution. Top-k sampling (Fan et al., 2018) only samples from the top-k most probable tokens. Nucleus sampling (Holtzman et al., 2020) only samples from the smallest set whose cumulative probability is at least α. However, those decoding-based methods are lack of controllability. Combined with above methods, our proposed method can further promote diversity using POS as a more controllable clue.

Training-based Methods. As a standard approach to training a neural text generation model, MLE has been proved to be defective. Choi et al. (2020) have shown that MLE may mislead the model because of the imbalanced token distribution. Thus, they design a greedy approach MefMax and factorize Softmax to ensure a balanced training according to the word frequency. FACE (Jiang et al., 2019) utilizes the target word frequency to modify the cross-entropy loss with a frequency-based weight factor. Welleck et al. (2020) introduce an unlikelihood loss to implicitly reduce the frequent tokens and potential repeats. Other approaches, such as negative training (He and Glass, 2020), reinforcement learning (Shirai et al., 2020), and imitation learning (Zhou and Lampouras, 2020), have recently been applied to promote the diversity during the training phase. All above training-based methods only learn from plain contextual features, while ignoring other linguistic features. Our focus is on leveraging POS features to guide both phases of training and decoding.

2.2 POS in Text Generation

Previous works, which leverage POS for text generation, can be summarized as follows:

POS in Encoding. A branch of previous works (He et al., 2019; Sennrich and Haddow, 2016b; Wray et al., 2019) explore to adopt POS on the encoding side to help language understanding and generation. Sennrich and Haddow (2016b) concatenate the embeddings of POS tags with sentence features to improve the translation quality. For the image caption generation, He et al. (2019) use POS tags to control the fusion of the image features and the related word embeddings. Wray et al. (2019) enrich the encoding with POS of the accompanying captions for cross-modal search tasks.

POS in Decoding. The second line of studies directly model the POS structure during decoding. Su et al. (2018) introduce a hierarchical decoder that relies on teacher forcing to learn different POS patterns on different layers. Deshpande et al. (2019) use POS tag sequences as summaries to implicitly drive image caption generation. Yang et al. (2019) treat POS tags as latent variables in NMT and optimize the model by Expectation Maximization (EM). Yang et al. (2021) employ POS sequences to constrain the non-autoregressive generation (NAG) modes to alleviate the multi-modality problem. However, all the previous studies only focus on a single specific task and leverage POS as hidden decoding features (Deshpande et al., 2019; Yang et al., 2019), teacher forcing techniques (Su et al., 2018; Bugliarello and Elliott, 2021) or NAG plannings (Yang et al., 2021) in order to improve the generic quality of generated texts, while our proposed methods regard POS tags as observed sequential variables and directly model the POS distribution during both phases of training and decoding with the goal of improving text diversity.

To our best knowledge, we are the first to introduce an explicit POS-guided generation method as a generic way to promote text diversity while maintaining quality.

3 Language Modeling

The goal of language models is to assign a probability to text (i.e. word sequence) $\mathbf{x} = [x_1, \ldots, x_T]$, where each x_t in the sequence is a token from a vocabulary $\mathcal{V}$, i.e., $x_t \in \mathcal{V}$, and $T \in \mathbb{N}$. We train the language models to learn a distribution $p_\theta(\mathbf{x})$ with the goal to fit the ground-truth distribution $p_\star(\mathbf{x})$ for all $\mathbf{x}$. Specifically, when the language model is a neural network, θ is regarded as the model parameters of the neural network, and we can factorize $p_\theta(\mathbf{x})$ as $p_\theta(\mathbf{x}) = \Pi_{t=1}^{T} p_\theta(x_t \mid \mathbf{x}_{<t})$. The conventional approach for learning the language model parameters θ is to maximize the log-likelihood by minimizing:

$$\begin{aligned} \mathcal{L}_{\text{MLE}}(\theta) &= -\sum_{t=1}^{T} \log p_\theta(x_t \mid \mathbf{x}_{<t}), \\ p_\theta(x_t \mid \mathbf{x}_{<t}) &= \frac{\exp \mathbf{h}_{t-1}^\top \mathbf{w}_{x_t}}{\sum_{x \in \mathcal{V}} \exp \mathbf{h}_{t-1}^\top \mathbf{w}_x}, \end{aligned} \tag{1}$$

where $\mathbf{h}_{t-1}$ is a hidden state of the context $\mathbf{x}_{<t}$, and $\mathbf{w}_{x_t}$ is the output embedding vector for $x_t \in \mathcal{V}$.

4 Methodology

In this section, we describe an overview of our proposed method, POS Guided Softmax and Sampling (POSG). POSG is designed to exploit syntactic structure, i.e., POS tags for text generation in both the training and decoding phases. Specifically, giving text sequence $\mathbf{x} = [x_1, \ldots, x_T]$, we first use off-the-shelf POS tagger (Manning et al., 2014) to annotate corresponding POS sequence $\boldsymbol{\rho} = [\rho_1, \ldots, \rho_T]$, where each ρ_t is a POS tag from the POS vocabulary $\mathcal{P}$, i.e., $\rho_t \in \mathcal{P}$, and $T \in \mathbb{N}$. We define all the tokens whose POS is ρ as a vocabulary $\mathcal{V}_\rho$, where $\mathcal{V}_\rho \subset \mathcal{V}$.

4.1 POS Guided Softmax

Figure 1 illustrates the core idea of our POS Guided Softmax. Given a context, there exist various choices for the next POS, which can be modeled as the next POS distribution. For the context *"no one knows"*, the next possible POS includes WH-pronoun (WP), preposition (IN), etc. For example, if WP is predicted as the next POS, the model will decode the next token from the WP vocabulary ($\mathcal{V}_{\text{WP}}$) with the token distribution of WP. Consequently, the complete sequence can be *"no one knows what will happen"*. For another case, if IN is predicted as the next POS, the next token will be decoded from $\mathcal{V}_{\text{IN}}$ with the corresponding token distribution. Then, the sequence may end up saying *"no one knows until it finally happens"*. This example also shows that the different choices of POS at each time step can result in vastly different generated text, thus promoting text diversity.

Following the core idea, we assume that the decoding process can be divided into two stages: for each time t, a POS tag ρ_t is predicted first, and then the model decodes next-token x_t from $\mathcal{V}_{\rho_t}$. Therefore, the joint conditional probability of x_t and its

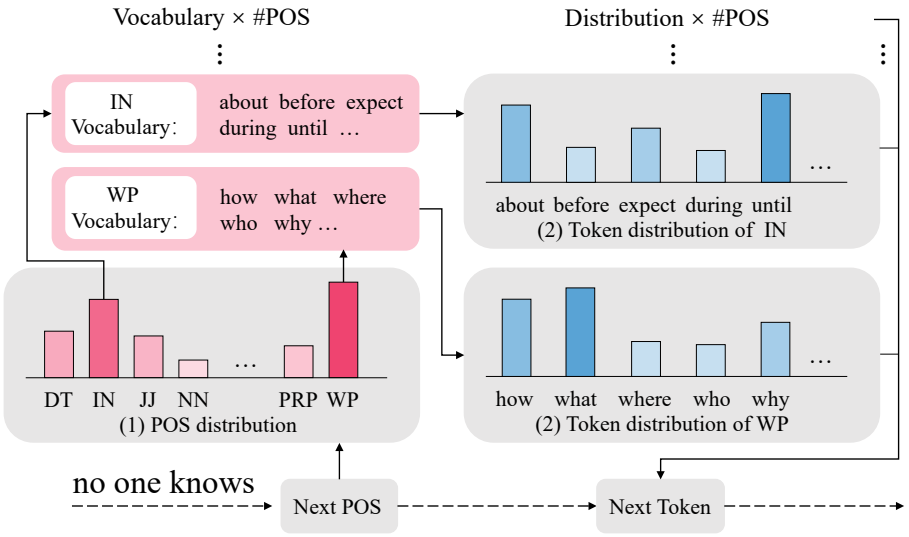

Figure 1: Illustration of POS Guided Softmax. The decoding process is decomposed into two stages: first predicts the next-POS distribution, and then decodes the next-token distribution from the vocabulary of the previously predicted POS. Since there exist some tokens with more than one POS, the final next-token distribution is the sum of all the POS's token distributions.

corresponding POS tag ρ_t is formulated as:

$$p_\theta\left(x_t, \rho_t \mid \mathbf{x}_{<t}\right) = p_{\theta_1}\left(\rho_t \mid \mathbf{x}_{<t}\right) \times p_{\theta_2}\left(x_t \mid \rho_t, \mathbf{x}_{<t}\right), \tag{2}$$

where $p_{\theta_1}\left(\rho_t \mid \mathbf{x}_{<t}\right)$ is the next-POS probability and $p_{\theta_2}\left(x_t \mid \rho_t, \mathbf{x}_{<t}\right)$ is the next-token probability conditioned on ρ_t. These probabilities are defined empirically by applying a linear output embedding on $\mathbf{h}_{t-1}$ and then a Softmax function respectively:

$$\begin{aligned} p_{\theta_1}\left(\rho_t \mid \mathbf{x}_{<t}\right) &= \frac{\exp \mathbf{h}_{t-1}^\top \mathbf{o}_{\rho_t}}{\Sigma_{\rho \in \mathcal{P}} \exp \mathbf{h}_{t-1}^\top \mathbf{o}_\rho}, \\ p_{\theta_2}\left(x_t \mid \rho_t, \mathbf{x}_{<t}\right) &= \begin{cases} \frac{\exp \mathbf{h}_{t-1}^\top \mathbf{w}_{x_t}}{\Sigma_{x \in \mathcal{V}_{\rho_t}} \exp \mathbf{h}_{t-1}^\top \mathbf{w}_x}, & \text{if } x_t \in \mathcal{V}_{\rho_t} \\ 0, & \text{otherwise} \end{cases}, \end{aligned} \tag{3}$$

where $\mathbf{o}_{\rho_t}$ and $\mathbf{w}_{x_t}$ are the output embeddings for $\rho_t \in \mathcal{P}$ and $x_t \in \mathcal{V}_{\rho_t}$, respectively. In this way, we regard POS tags as observed sequential variables, which also contributes to the model interpretability and controllability. Then, the final next-token distribution can be formulated as: $p_\theta\left(x_t \mid \mathbf{x}_{<t}\right) = \sum_{\rho_t \in \mathcal{P}} p_\theta\left(x_t, \rho_t \mid \mathbf{x}_{<t}\right)$. Note that some tokens may have more than one POS, and $p_\theta\left(x_t, \rho_t \mid \mathbf{x}_{<t}\right) = 0$ for $x_t \notin \mathcal{V}_{\rho_t}$. Since the number of POS in a specific language family is fixed, there is no problem of insufficient exploration in variables' space.

As mentioned before, we think of POS tags as observed sequential variables and extend the training text set with annotated POS sequences, so we define the POS guided training objective as follows:

$$\mathcal{L}_{\text{POS-Guided}}(\theta) = -\sum_{t=1}^{T} \left[\log p_{\theta_1}\left(\rho_t \mid \mathbf{x}_{<t}\right) + \log p_{\theta_2}\left(x_t \mid \rho_t, \mathbf{x}_{<t}\right)\right]. \tag{4}$$

4.2 POS Guided Sampling

We propose POS Guided Sampling based on POS Guided Softmax. Consistent with POS Guided Softmax, the key idea is to divide the whole sampling process into two stages: *POS sampling* and *token sampling*. In POS sampling, we first sample a POS, and then in token sampling, we use the sampled POS to control the sampling of tokens. Note that arbitrary sampling strategies can be adopted to both the POS sampling and token sampling. Here, we take top-k sampling for POS sampling, and nucleus sampling for token sampling as an example, and then we can formulate our POS Guided Sampling as follows:

$$\begin{aligned} p'_\theta\left(x_t \mid \mathbf{x}_{<t}\right) &= \sum_{\rho_t \in \mathcal{P}} \left[p'_{\theta_1}\left(\rho_t \mid \mathbf{x}_{<t}\right) \times p'_{\theta_2}\left(x_t \mid \rho_t, \mathbf{x}_{<t}\right)\right], \\ p'_{\theta_1}\left(\rho_t \mid \mathbf{x}_{<t}\right) &= \begin{cases} \frac{p_{\theta_1}\left(\rho_t \mid \mathbf{x}_{<t}\right)}{\mathcal{Z}_{\theta_1}}, & \text{if } \rho_t \in \mathcal{P}' \\ 0, & \text{otherwise} \end{cases}, \\ p'_{\theta_2}\left(x_t \mid \rho_t, \mathbf{x}_{<t}\right) &= \begin{cases} \frac{p_{\theta_2}\left(x_t \mid \rho_t, \mathbf{x}_{<t}\right)}{\mathcal{Z}_{\theta_2}}, & \text{if } x_t \in \mathcal{V}'_{\rho_t} \\ 0, & \text{otherwise} \end{cases}, \\ \mathcal{Z}_{\theta_1} &= \sum_{\rho \in \mathcal{P}'} p_{\theta_1}\left(\rho \mid \mathbf{x}_{<t}\right), \\ \mathcal{Z}_{\theta_2} &= \sum_{x \in \mathcal{V}'_{\rho_t}} p_{\theta_2}\left(x \mid \rho_t, \mathbf{x}_{<t}\right), \end{aligned} \tag{5}$$

where $\mathcal{P}' \subset \mathcal{P}$ is a POS set containing top-k most probable POS tags, and $\mathcal{V}'_{\rho_t} \subset \mathcal{V}_{\rho_t}$ is the smallest token set such that $\sum_{x \in \mathcal{V}'_{\rho_t}} p_{\theta_2}(x \mid \rho_t, \mathbf{x}_{<t}) \geq \alpha^{(\text{token})}$. $k^{(\text{POS})}$ and $\alpha^{(\text{token})}$ ($0 < \alpha^{(\text{token})} \leq 1$) are the hyper-parameters for the sampling of POS and token, respectively. For other sampling strategies used in POS sampling and token sampling, POS Guided Sampling can be similarly defined.

5 Experiments

We systematically evaluate our proposed methods on language modeling task (Section 5.2) and paraphrase generation task (Section 5.3).

5.1 Experimental Setup

Model Architecture Since our proposed methods are architecture agnostic, we implement POS Guided Softmax on the Transformer (Vaswani et al., 2017), a widely used architecture for neural text generation. Details of the experimental setup for each task are shown in Appendix A.

Baseline Models We compare our **POS Guided Softmax and Sampling (POSG)** with the following baselines: (i) **Maximum likelihood estimation (MLE)**, a standard approach for neural text generation. (ii) **Frequency-Aware Cross-Entropy (FACE)** (Jiang et al., 2019) dynamically weights the cross-entropy losses conditioned on the token frequency. (iii) **Frequency Factorization Softmax (F^2-Softmax)** (Jiang et al., 2019) factorizes the standard Softmax based on the token frequency. (iv) **Unlikelihood training (UL)** (Welleck et al., 2020) is to enhance the log-likelihood loss with an unlikelihood loss that penalizes the generation of repeated tokens. (v) We further implement two task-specific baselines: **Mixture of Softmaxes (MoS)** (Yang et al., 2018) for language modeling, **Syntax Guided Controlled Paraphraser (SGCP)** (Kumar et al., 2020) for paraphrase generation. Note that decoding-based methods, including top-k and nucleus sampling, can be directly compared to POSG, when they are applied to MLE. The details will be described in the sections of Generation Details.

5.2 Language Modeling

Dataset We performed experiments on the Wikitext-103[2] dataset (Merity et al., 2017) for language modeling. In order to train our POS Guided Softmax, we need the corresponding POS tags. We use the Stanford CoreNLP's POS tagger (Manning et al., 2014) to annotate words in Wikitext-103 with XPOS[3] tags (Hornby et al., 2017). In our implementation, there are 45 different POS tags in total.

Generation Details We conduct the text completion task to evaluate models on the test set. Specifically, for each sample, we truncate 50 tokens as the prefix, and then guide model to decode following 100 tokens as the continuation from the given prefix. Finally, there are 1536 prefixes in the test set. We use stochastic decoding to generate text. Note that all the baselines have only one sampling stage, i.e., token sampling, while our POSG has an additional POS sampling. To reach a good trade-off between quality and diversity, we adopt nucleus sampling with $\alpha^{(\text{token})} = 0.5$ for token sampling (for all models including our POSG and baselines). For our POSG, we adopt top-k sampling in POS sampling, since the size of the POS vocabulary $\mathcal{P}$ is much smaller than the total token vocabulary. We then conduct a grid search to find the $k^{(\text{POS})}$ whose generated continuations have the smallest reverse language model score (Semeniuta et al., 2018) on the validation set. $k^{(\text{POS})}$ is finally set to 20. Some generated cases are shown in Appendix D.

Metrics Following Choi et al. (2020), we evaluate the generated text with two sets of metrics: (i) **Diversity**: We use Self-BLEU (Zhu et al., 2018) which is calculated by computing BLEU (Papineni et al., 2002) of each generated text with all other generations as references. We also compute the generated continuations' unique tokens (Uniq), distinct n-gram (Distinct-n). We also use repetition (Rep) (Holtzman et al., 2020), the percentage of continuations ending with a repetition loop, to evaluate text diversity. (ii) **Quality**: We measure the perplexity (PPL) (Mnih and Teh, 2012), KL-Divergence (KLD) (Kullback, 1997) on unigram distributions, and MS-Jaccard (Alihosseini et al., 2019) on n-gram. All the metrics are calculated between the generations as hypotheses and the ground truths as references.

Automatic evaluation Table 2 shows the automatic evaluation results comparing different models on the language modeling task. In terms of Self-BLEU4, Rep, and Distinct-n, our POSG performs much better than all the baselines, indicating that our proposed model can generate diverse text effectively. The FACE also performs well, and it

[2] https://s3.amazonaws.com/research.metamind.io/wikitext/wikitext-103-v1.zip

[3] The XPOS tags are language-specific part-of-speech tags from the Universal Dependency Treebanks.

Models	Self-BLEU4 ↓	Rep ↓	Uniq ↑	Distinct ↑			PPL ↓	KLD ↓	MS-Jaccard ↑		
				n=1	n=2	n=3			n=1	n=2	n=3
MLE	46.9	1.86	11.7k	50.2	77.2	86.2	**32.7**	1.34	56.9	38.2	25.4
FACE	34.2	1.56	**14.9k**	60.0	85.1	90.6	36.1	1.18	58.6	37.6	24.0
F^2-Softmax	51.5	4.09	10.8k	42.4	65.3	75.2	35.0	1.58	51.5	33.7	22.4
UL	42.4	0.240	12.8k	**61.2**	87.8	93.3	37.0	1.20	61.2	40.4	**26.2**
MoS	55.3	3.99	8.40k	48.2	74.3	83.0	38.2	1.48	56.9	38.1	25.4
POSG	**34.1**	**0.000**	13.8k	60.2	**88.8**	**94.3**	34.4	**1.17**	**62.2**	**40.7**	25.9

Table 2: Automatic evaluation results for different models on the language modeling task. Numbers $n \in \{1, 2, 3\}$ in the column heads under Distinct and MS-Jaccard refer to n-gram. (**Bold**: the best; Underline: the second best).

Models	Distinct ↑					
	1-P	1-G	2-P	2-G	3-P	3-G
MLE	16.0	39.2	38.5	61.0	54.7	70.8
FACE	17.8	49.8	46.6	73.0	65.4	80.8
F^2-Softmax	16.4	41.1	39.3	63.8	55.8	74.0
UL	18.0	51.7	46.5	76.8	66.3	85.2
MoS	16.4	41.1	40.0	63.8	56.5	72.9
POSG	19.9	56.2	58.1	85.3	80.6	92.3
Human	21.7	67.7	61.8	93.0	83.8	95.9
PPMCC	0.988		0.986		0.986	

Table 3: Results of distinct n-gram and n-POS with corresponding Pearson product-moment correlation coefficient (PPMCC). n-P and n-G where $n \in \{1, 2, 3\}$ are abbreviated notations for n-POS and n-gram.

Models	Div. ↑	Qua. ↑
MLE	2.86*	3.10
FACE	3.32*	3.18
F^2-Softmax	2.35*	2.80*
UL	3.36*	**3.20**
MoS	2.79*	3.06*
POSG	**3.45**	3.17

Table 4: Human evaluation on language modeling. * denotes statistical significance compared with POSG (Mann-Whitney u-test, $p < 0.1$).

achieves the best in Uniq. However, by checking the outputs (Table 15 in Appendix D), we find that FACE produces more incoherent text that is hard to understand.

Since training-based methods including ours make a trade-off between the text diversity and the likelihood of ground truth, MLE gets the lowest PPL. However, the optimal or second best results of quality metrics confirm that POSG can still maintain comparable generation quality.

We further conduct a correlation test to verify that the text diversity is closely correlated with the POS diversity. We first randomly sample 500 generated continuations from each model, and annotate them with the POS tagger. We define a n-POS to be contiguous n POS tags from the annotated POS tag sequence. Then, we can describe the degree of POS diversity by calculating the proportion of the distinct n-POS. Table 3 presents results of distinct n-gram and n-POS with corresponding Pearson product-moment correlation coefficient. In terms of distinct n-POS, POSG also surpasses all the baselines. This demonstrates that our proposed model can substantially promote the POS diversity. Moreover, the Pearson correlations between distinct n-POS and n-gram are extremely high, which indicates that the high POS diversity indeed leads to the high text diversity.

Human evaluation For the language modeling task, following Tevet and Berant (2021) we randomly sample 100 generated continuations from each model. Each of them is scored between 1 to 5 (5 is the best), by five workers to evaluate the overall *Diversity* (Div.) and *Quality* (Qua.). The results of the human evaluation on language modeling are shown in Table 4. It can be seen that our POSG significantly outperforms all other baselines in diversity, and performs relatively well in quality.

5.3 Paraphrase Generation

Dataset We use the the ParaNMT-50M[4] dataset (Wieting and Gimpel, 2018) for paraphrase generation. ParaNMT-50M consists of over 50 million paraphrases, generated by back-translation. For better training, we first remove the sentences that are less than 10 tokens. Moreover, ParaNMT-50M dataset has provided translation scores to measure the quality of back-translation, that a low translation score means semantically inconsistent, while a high translation score usually accompanies low

[4] https://drive.google.com/file/d/1rbF3daJjCsa1-fu2GANeJd2FBXos1ugD/view

Models	Self-WER↑	Self-BLEU4↓	Distinct↑			BERTScore↑	BLEU4↑	ROUGE↑		
			n=1	n=2	n=3			1	2	L
MLE	74.2	25.1	78.4	82.8	78.5	47.4	9.81	38.1	16.9	38.8
FACE	73.0	25.0	78.9	83.6	79.6	48.1	10.1	38.7	17.2	39.1
F2-Softmax	76.4	28.0	78.2	83.0	79.5	**53.9**	**11.4**	41.1	**19.5**	**42.8**
UL	77.2	21.2	80.1	**85.3**	80.9	36.0	7.59	30.6	13.3	30.3
SGCP	83.0	28.6	81.9	82.6	77.7	47.9	9.91	**41.3**	17.7	41.1
POSG	**89.7**	**19.6**	**82.1**	**85.3**	**81.8**	48.3	9.79	40.3	17.1	39.4

Table 5: Automatic evaluation results for different models on the paraphrase generation task. Numbers $n \in \{1, 2, 3\}$ in the column heads under Distinct refer to n-gram. (**Bold**: the best; Underline: the second best).

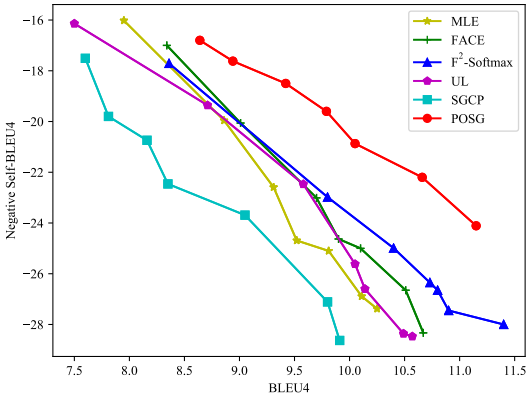

Figure 2: Quality-diversity trade-off for different models on paraphrase generation. The x-axis measures BLEU4 for quality, and the y-axis measures negative Self-BLEU4 for diversity. Both are the bigger the better.

Models	Div. ↑		Flu. ↑	Rel. ↑
	Lex.	Syn.		
MLE	2.92	2.65⋆	3.34⋆	3.09⋆
FACE	2.91	2.58⋆	**3.60**	3.35
F^2-Softmax	2.77⋆	2.57⋆	3.59	**3.38**
UL	3.00	2.68⋆	3.37⋆	3.17⋆
SGCP	2.74⋆	2.67⋆	3.50	3.21⋆
POSG	**3.02**	**2.79**	3.58	3.35

Table 6: Human evaluation on paraphrase generation. ⋆ denotes statistical significance compared with POSG (Mann-Whitney u-test, $p < 0.1$).

diversity. Therefore, we only keep the paraphrase pairs whose translation scores are between 0.7 to 0.8. Finally, we get a filtered dataset containing 1.6 million paraphrase pairs with both high quality and diversity. We also use Stanford CoreNLP to tokenize the text and get corresponding POS tags.

Generation Details We conduct the standard sequence-to-sequence paraphrase generation for testing. Note that, during inference, SGCP needs a corresponding exemplar sentence to paraphrase the input sentence, while our model does not. So, for a fair comparison, we prune the exemplar tree to the height $\max(3, H_{\text{max}} - 4)$ to reduce the impact from exemplar sentence, where H_{max} is the height of the full constituency tree of the exemplar sentence. We use the test set provided in the work of SGCP[5] that contains 800 paraphrase pairs and correspond exemplar sentences for inference. For a fair comparison, we closely follow Kumar et al. (2020) to generate paraphrase using beam search for all the models with beam size 10. For the sampling hyperparameter in POS sampling, we also conduct a grid search, and $k^{(\text{POS})}$ is finally set to 5. Some generated cases are shown in Appendix D.

Metrics We also evaluate the generated paraphrases with two sets of metrics, (i) **Diversity**: To assess how different the generated paraphrases are compared to the original sentences, we calculate BLEU and Word Error Rate (WER) (Goyal and Durrett, 2020) between generated paraphrases and input sentences. We denote them as Self-BLEU (see Appendix A.3 for the difference with the Self-BLEU in language modeling) and Self-WER, respectively. We also compute the generated paraphrases' distinct n-gram (Distinct-n) to evaluate text diversity. (ii) **Quality**: we calculate BLEU score on n-gram to evaluate the closeness of the generated paraphrases to references. Besides, we use the BERTScore (Zhang et al., 2020) to measure the semantic consistency between generated paraphrases and input sentences. We also compute ROUGE-1,2,L between the generated and the reference to evaluate the generation quality.

Automatic evaluation The experimental results on the paraphrase generation task are shown in Table 5. Our proposed model outperforms other baselines on all the diversity metrics. In terms of quality metrics, our POSG performs better than MLE,

[5] https://github.com/malllabiisc/SGCP

Models	Self-BLEU4 ↓	Rep ↓	Uniq ↑	Distinct ↑			PPL ↓	KLD ↓	MS-Jaccard ↑		
				n=1	n=2	n=3			n=1	n=2	n=3
POSG	**34.1**	**0.000**	**13.8k**	**60.2**	**88.8**	**94.3**	34.4	**1.17**	**62.2**	**40.7**	**25.9**
w/o POSG-Sampling	40.6	0.841	13.1k	55.2	83.0	90.6	34.4	1.29	56.9	37.5	24.5
MLE	46.9	1.86	11.7k	50.2	77.2	86.2	**32.7**	1.34	56.9	38.2	25.4

Table 7: Results of ablation study on the language modeling task. Note that PPL measures the ability of the model to generate fluent text, which is not affected by the sampling strategy.

Models	Self-WER↑	Self-BLEU4↓	Distinct↑			BERTScore↑	BLEU4↑	ROUGE↑		
			n=1	n=2	n=3			1	2	L
POSG	**89.7**	**19.6**	**82.1**	**85.3**	**81.8**	48.3	9.79	40.3	17.1	39.4
w/o POSG-Sampling	87.6	24.1	78.0	80.5	78.5	**52.6**	**11.1**	**40.9**	**19.7**	**42.4**
MLE	74.2	25.1	78.4	82.8	78.5	47.4	9.81	38.1	16.9	38.8

Table 8: Results of ablation study on the paraphrase generation task.

Adjective Probability	Adjs. per Sentence	Self-BLEU4↓	BLEU4↑
$\times 0.1$	0.43	20.2	9.47
$\times 1$	0.66	19.6	9.79
$\times 10$	1.04	18.7	9.45

Table 9: Results of controllability analysis on the paraphrase generation task. "$\times n$" means that we manually multiply the probability of "Adjective" by n.

FACE, and UL, while the best model in quality, i.e., F^2-Softmax performs badly in diversity. Moreover, compared with other syntax-guided models, i.e., SGCP, our model performs much better in diversity and has a comparable performance in quality. This further confirms that our model can effectively promote text diversity without the help of exemplars.

To make a more intuitive comparison, we further apply stochastic decoding for different models, and tune the sampling hyper-parameters to generate different sets of paraphrases. Then, we calculate BLEU4 and Self-BLEU4 scores for these sets, and draw the quality-diversity trade-off in Figure 2. Clearly, POSG surpasses all the baselines with a significant gap. These results confirm that our model can produce equally high-quality text that is more diverse, and vice versa.

Human Evaluation We also conduct a human evaluation for the generated paraphrases. 100 examples are randomly sampled from each models' outputs, respectively. Each of them are evaluated by five workers from the following four aspects: *Lexical Diversity* (LeD.), and *Syntactical Diversity* (SyD.), *Fluency* (Flu.), *Relevance* (Rel.). All these aspects are scored between 1 to 5, the higher the better. As shown in Table 6, the results of the human evaluation are strongly consistent with the automatic evaluation. Compared with MLE, UL and SGCP, POSG substantially improves the generation quality, and it only has a tiny gap from the best model in fluency and relevance scores. Meanwhile, POSG has the best scores in diversity, which further verifies that our proposed methods can generate more lexically and syntactically diverse paraphrases. The detailed questionnaire, and other details are shown in Appendix E.

5.4 Ablation Study

We perform ablation studies to reveal the effect of POS Guided Softmax and POS Guided Sampling. As shown in Table 7 and Table 8, compared with MLE, POS Guided Softmax (without POS Guided Sampling) can improve text quality for both the tasks. It is worth to mention that, it is natural to find that the model without POSG-Sampling performs better than the model with POSG. That is because POSG-sampling is a stochastic decoding method like nucleus sampling, which will sacrifice the quality of the generated text to promote text diversity. Therefore, POS Guided Sampling can dramatically promote text diversity for both the tasks. These results confirm the effectiveness of both the components.

6 Analysis

6.1 Interpretability

Compared with one-stage sampling such as top-k sampling, POSG will lead to the entropy increasing of a language model's distribution, and thus lead to more diverse outputs (see Appendix B for the proof, Appendix C.1 for experimental results).

6.2 Controllability

Our proposed POSG first samples a POS, and then samples a token from the vocabulary of the previously predicted POS. Therefore, we can control the POS sampling stage by forcing the probability of some specific POS to be higher or lower. For example, on the paraphrase generation task, we can multiply the probability of "Adjective" ("JJ") and renormalize by dividing by the sum, aiming at generating more descriptive style paraphrases.

The results are shown in Table 9. These results confirm that by leveraging POS as an observed and controllable clue, the generated text can be successfully modulated with negligible effect on quality and diversity (see Appendix C.2 for cases).

7 Conclusion

In this paper, we have introduced POS Guided Softmax and Sampling, simple but effective methods to address the low-diversity problem in text generation. POSG guides models to capture contextual and syntactical information by leveraging POS as an observed and controllable clue in both the training and decoding phases. Experimental results and human evaluation on language modeling and paraphrase generation have demonstrated the effectiveness of our methods.

Acknowledgements

This work was supported by National Key R&D Program of China (2021YFF0901502), National Science Foundation of China (No. 62161160339), State Key Laboratory of Media Convergence Production Technology and Systems and Key Laboratory of Science, Technology and Standard in Press Industry (Key Laboratory of Intelligent Press Media Technology). We appreciate the anonymous reviewers for their helpful comments. Xiaojun Wan is the corresponding author.

References

Danial Alihosseini, Ehsan Montahaei, and Mahdieh Soleymani Baghshah. 2019. Jointly measuring diversity and quality in text generation models. In *Proceedings of the Workshop on Methods for Optimizing and Evaluating Neural Language Generation*, pages 90–98.

Emanuele Bugliarello and Desmond Elliott. 2021. The role of syntactic planning in compositional image captioning. In *In Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*, pages 593–607.

Massimo Caccia, Lucas Caccia, William Fedus, Hugo Larochelle, Joelle Pineau, and Laurent Charlin. 2020. Language gans falling short. In *In Proceedings of the 8th International Conference on Learning Representations*.

Yue Cao and Xiaojun Wan. 2020. Divgan: Towards diverse paraphrase generation via diversified generative adversarial network. In *Findings of the Association for Computational Linguistics: EMNLP 2020, Online Event, 16-20 November 2020*, volume EMNLP 2020 of *Findings of ACL*, pages 2411–2421.

Byung-Ju Choi, Jimin Hong, David Keetae Park, and Sang Wan Lee. 2020. Fˆ2-softmax: Diversifying neural text generation via frequency factorized softmax. In *In Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing*, pages 9167–9182.

Aditya Deshpande, Jyoti Aneja, Liwei Wang, Alexander G. Schwing, and David A. Forsyth. 2019. Fast, diverse and accurate image captioning guided by part-of-speech. In *In IEEE Conference on Computer Vision and Pattern Recognition*, pages 10695–10704.

Angela Fan, Mike Lewis, and Yann N. Dauphin. 2018. Hierarchical neural story generation. In *In Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics*, volume 1, pages 889–898.

Zihao Fu, Wai Lam, Anthony Man-Cho So, and Bei Shi. 2021. A theoretical analysis of the repetition problem in text generation. *Proceedings of the AAAI Conference on Artificial Intelligence*, 35(14):12848–12856.

Tanya Goyal and Greg Durrett. 2020. Neural syntactic preordering for controlled paraphrase generation. In *In Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 238–252.

Tianxing He and James R. Glass. 2020. Negative training for neural dialogue response generation. In *In Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 2044–2058.

Xinwei He, Baoguang Shi, Xiang Bai, Gui-Song Xia, Zhaoxiang Zhang, and Weisheng Dong. 2019. Image caption generation with part of speech guidance. *Pattern Recognit Letters*, 119:229–237.

Ari Holtzman, Jan Buys, Li Du, Maxwell Forbes, and Yejin Choi. 2020. The curious case of neural text degeneration. In *In Proceedings of the 8th International Conference on Learning Representations*.

Ryan Hornby, Clark Taylor, and Jungyeul Park. 2017. Corpus selection approaches for multilingual parsing from raw text to universal dependencies. In *In*

Proceedings of the CoNLL 2017 Shared Task: Multilingual Parsing from Raw Text to Universal Dependencies, pages 198–206.

Shaojie Jiang, Pengjie Ren, Christof Monz, and Maarten de Rijke. 2019. Improving neural response diversity with frequency-aware cross-entropy loss. In *In Proceedings of the World Wide Web Conference*, pages 2879–2885.

Ilia Kulikov and Kyunghyun Cho. 2019. Importance of search and evaluation strategies in neural dialogue modeling. In *INLG*.

Solomon Kullback. 1997. *Information theory and statistics*. Courier Corporation.

Ashutosh Kumar, Kabir Ahuja, Raghuram Vadapalli, and Partha P. Talukdar. 2020. Syntax-guided controlled generation of paraphrases. *Trans. Assoc. Comput. Linguistics*, 8:330–345.

Christopher D. Manning, Mihai Surdeanu, John Bauer, Jenny Rose Finkel, Steven Bethard, and David McClosky. 2014. The stanford corenlp natural language processing toolkit. In *In Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics System Demonstrations*, pages 55–60.

Stephen Merity, Caiming Xiong, James Bradbury, and Richard Socher. 2017. Pointer sentinel mixture models. In *In Proceedings of 5th International Conference on Learning Representations*.

Andriy Mnih and Yee Whye Teh. 2012. A fast and simple algorithm for training neural probabilistic language models. In *In Proceedings of the 29th International Conference on Machine Learning*.

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. Bleu: a method for automatic evaluation of machine translation. In *In Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics*, pages 311–318.

Stanislau Semeniuta, Aliaksei Severyn, and Sylvain Gelly. 2018. On accurate evaluation of gans for language generation. *arXiv preprint arXiv:1806.04936*.

Rico Sennrich and Barry Haddow. 2016a. Linguistic input features improve neural machine translation. In *In Proceedings of the First Conference on Machine Translation*, pages 83–91.

Rico Sennrich and Barry Haddow. 2016b. Linguistic input features improve neural machine translation. In *In Proceedings of the First Conference on Machine Translation*, pages 83–91.

Keisuke Shirai, Kazuma Hashimoto, Akiko Eriguchi, Takashi Ninomiya, and Shinsuke Mori. 2020. Neural text generation with artificial negative examples. *arXiv preprint arXiv:2012.14124*.

Shang-Yu Su, Kai-Ling Lo, Yi Ting Yeh, and Yun-Nung Chen. 2018. Natural language generation by hierarchical decoding with linguistic patterns. In *In Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics*, pages 61–66.

Guy Tevet and Jonathan Berant. 2021. Evaluating the evaluation of diversity in natural language generation. In *In Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*, pages 326–346.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *In Advances in Neural Information Processing Systems*, pages 5998–6008.

Ashwin K. Vijayakumar, Michael Cogswell, Ramprasaath R. Selvaraju, Qing Sun, Stefan Lee, David J. Crandall, and Dhruv Batra. 2018. Diverse beam search for improved description of complex scenes. In *In Proceedings of the Thirty-Second AAAI Conference on Artificial Intelligence*, pages 7371–7379.

Sean Welleck, Ilia Kulikov, Stephen Roller, Emily Dinan, Kyunghyun Cho, and Jason Weston. 2020. Neural text generation with unlikelihood training. In *In Proceedings of the 8th International Conference on Learning Representations*.

John Wieting and Kevin Gimpel. 2018. Paranmt-50m: Pushing the limits of paraphrastic sentence embeddings with millions of machine translations. In *In Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics*, pages 451–462.

Michael Wray, Gabriela Csurka, Diane Larlus, and Dima Damen. 2019. Fine-grained action retrieval through multiple parts-of-speech embeddings. In *In the 2019 IEEE/CVF International Conference on Computer Vision*, pages 450–459.

Kexin Yang, Wenqiang Lei, Dayiheng Liu, Weizhen Qi, and Jiancheng Lv. 2021. Pos-constrained parallel decoding for non-autoregressive generation. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing, ACL/IJCNLP 2021, (Volume 1: Long Papers), Virtual Event, August 1-6, 2021*, pages 5990–6000. Association for Computational Linguistics.

Xuewen Yang, Yingru Liu, Dongliang Xie, Xin Wang, and Niranjan Balasubramanian. 2019. Latent part-of-speech sequences for neural machine translation. In *In Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing*, pages 780–790.

Zhilin Yang, Zihang Dai, Ruslan Salakhutdinov, and William W. Cohen. 2018. Breaking the softmax bottleneck: A high-rank RNN language model. In *6th*

International Conference on Learning Representations, ICLR 2018, Vancouver, BC, Canada, April 30 - May 3, 2018, Conference Track Proceedings. OpenReview.net.

Tianyi Zhang, Varsha Kishore, Felix Wu, Kilian Q. Weinberger, and Yoav Artzi. 2020. Bertscore: Evaluating text generation with BERT. In *In Proceedings of the 8th International Conference on Learning Representations*.

Giulio Zhou and Gerasimos Lampouras. 2020. Generating safe diversity in nlg via imitation learning. *arXiv preprint arXiv:2004.14364*.

Yaoming Zhu, Sidi Lu, Lei Zheng, Jiaxian Guo, Weinan Zhang, Jun Wang, and Yong Yu. 2018. Texygen: A benchmarking platform for text generation models. In *Proceedings of the 41st International ACM SIGIR Conference on Research & Development in Information Retrieval*, pages 1097–1100.

A Experimental Setup

A.1 Dataset

The dataset statistics of Wikitext-103 and ParaNMT-50M are reported in Table 10 and Table 11, respectively.

Since ParaNMT-50M is generated by back-translation, the dataset has provided translation scores to measure the quality of back-translation, that a low translation score means semantically inconsistent, while a high translation score usually accompanies low diversity. Therefore, we only keep the paraphrase pairs whose translation scores are between 0.7 and 0.8. Moreover, for better training, we remove the sentences that are less than 10 tokens. Finally, we get a filtered dataset containing 1.6 million paraphrase pairs with both high quality and diversity.

For language modeling, we use the original settings of Wikitext-103 dataset for training, validation, and test set splitting. For paraphrase generation, we use the filtered training, validation set of ParaNMT-50M, and the test set provided in the work of SGCP. It is worth to mention that Wikitext-103 is under the CC BY-SA 3.0 license, and ParaNMT-50M is under the CC-BY license.

	Train	Valid	Test
#Articles	28,475	60	60
#Tokens	103,227,021	217,646	245,569

Table 10: Statistics of Wikitext-103.

	Train	Valid	Test
#Sentence	1,640,709	3,000	800

Table 11: Statistics of ParaNMT-50M.

A.2 Architectures and Hyperparamters

For the language modeling task, we use a 12-layer Transformer Decoder with 8 attention heads, embedding dimension 512, and projection dimension 2048. For the paraphrase generation task, we use a 6-layer Transformer Encoder and Decoder with the same other settings. All the algorithms are implemented in Pytorch and trained on a machine with 8 NVIDIA GTX 2080Ti GPUs for 10 epochs with the hyper-parameters reported in Table 12.

hyper-parameters	Wikitext-103	ParaNMT-50M
Vocabulary size	267,735	100,000
Batch size	12	96
Learning rate	0.0001	0.0001
Finetuning LR	0.00001	0.00001
Finetuning step	1500	1500
Gradient clipping	0.25	0.25
Weight decay	0.001	0.001
Droupout	0.1	0.1
Optimizer	Adam	Adam
-β_1	0.9	0.9
-β_2	0.999	0.999
-ϵ	1e-8	1e-8

Table 12: Hyperparameter settings for different datasets.

We choose the architecture settings and batch sizes according to the GPU memory constraint. Note that we use FACE-OPR among the four variants of FACE, and we train it in the way of finetuning with corresponding finetuning LR and finetuning step. Additionally, we use 7 mixture components in MoS.

A.3 Metrics

Note that, the calculations of Self-BLEU are different for language modeling and paraphrase generation. This is because the typical definitions of Self-BLEU for these two different task are indeed different. For language modeling, Self-BLEU (Zhu et al., 2018) is a metric to evaluate the inner diversity of the generated data, while for paraphrase generation, Self-BLEU (Cao and Wan, 2020) is used to evaluate the degree to which the generated paraphrases are different from the original sentence.

B Proof

We prove that our POS Guided Softmax and Sampling can certainly generate more diverse text than the one-stage sampling, top-k sampling as an example.

In information theory, the entropy of a random variable is the average level of "information", "surprise" in the variable's possible outcomes. Therefore, we can use the entropy of a language model's distribution $p(x)$ to measure its diversity. We denote the entropy of $p(x)$ as $H(p)$: $H(p) = -\sum_{x\in\mathcal{V}} p(x)\log p(x)$. The increase of $H(p)$ means the increase of diversity.

For example, compared with greedy search, diversity-promoting sampling methods, such as top-k sampling can increases $H(p)$ from $-p(x_{\max})\log p(x_{\max})$ to $-\sum_{x\in\mathcal{V}_k}\frac{p(x)}{Z_k}\log\frac{p(x)}{Z_k}$, where $x_{\max}$ is the token with the max probability, $\mathcal{V}_k$ is the set of top-k most probable tokens, $Z_k = \sum_{x\in\mathcal{V}_k} p(x)$, and obviously $x_{\max}\in\mathcal{V}_k$.

Now, we prove that our POSG with two sampling stages can lead to the entropy increasing, compared with one-stage top-k sampling as an example.

For one-stage top-k sampling,

$$\begin{aligned}
H(p)^{\text{(top-k)}} &= -\sum_{x\in\mathcal{V}_k}\frac{p(x)}{Z_k}\log\frac{p(x)}{Z_k} \\
&= -\sum_{x\in\mathcal{V}_k}\frac{\sum_{\rho\in\mathcal{P}}p(x,\rho)}{Z_k}\log\frac{\sum_{\rho\in\mathcal{P}}p(x,\rho)}{Z_k} \\
&= -\log|\mathcal{P}| - \sum_{x\in\mathcal{V}_k}\frac{\sum_{\rho\in\mathcal{P}}p(x,\rho)}{Z_k}\log\frac{\sum_{\rho\in\mathcal{P}}p(x,\rho)}{Z_k\times|\mathcal{P}|}
\end{aligned} \tag{6}$$

According to the Log sum inequality, it follows:

$$\begin{aligned}
H(p)^{\text{(top-k)}} &\geq -\log|\mathcal{P}| - \sum_{x\in\mathcal{V}_k}\sum_{\rho\in\mathcal{P}}\frac{p(x,\rho)}{Z_k}\log\frac{p(x,\rho)}{Z_k} \\
&= -\log|\mathcal{P}| - \sum_{\rho\in\mathcal{P}}\sum_{x\in\mathcal{V}_k}\frac{p(x,\rho)}{Z_k}\log\frac{p(x,\rho)}{Z_k}
\end{aligned} \tag{7}$$

Since $p(x,\rho) = 0$ for $x\notin\mathcal{V}_\rho$, it follows:

$$H(p)^{\text{(top-k)}} \geq -\log|\mathcal{P}| - \sum_{\rho\in\mathcal{P}}\sum_{x\in\mathcal{V}_{k,\rho}}\frac{p(x,\rho)}{Z_k}\log\frac{p(x,\rho)}{Z_k} \tag{8}$$

where $\mathcal{V}_{k,\rho} = \{x\in\mathcal{V}_k \mid \rho\in\text{POS}(x)\}$, $\text{POS}(x)$ is the set of all POS tags of token x. Thus, $\mathcal{V}_{k,\rho}\subseteq\mathcal{V}_k$.

For our POSG with two sampling stages,

$$\begin{aligned}
H(p)^{\text{(POS)}} &= -\sum_{x\in\mathcal{V}}\sum_{\rho\in\mathcal{P}}p'(x,\rho)\log\sum_{\rho\in\mathcal{P}}p'(x,\rho) \\
&= -\sum_{x\in\mathcal{V}}\sum_{\rho\in\mathcal{P}}p'(\rho)p'(x\mid\rho)\log\sum_{\rho\in\mathcal{P}}p'(\rho)p'(x\mid\rho)
\end{aligned} \tag{9}$$

where $p'(x,\rho)$ is defined in Equation 2, $p'(\rho)$ and $p'(x\mid\rho)$ are defined in Equation 5. Again, according to the Log sum inequality, it follows:

$$\begin{aligned}
H(p)^{\text{(POS)}} &\geq -\log|\mathcal{P}| \\
&-\sum_{x\in\mathcal{V}}\sum_{\rho\in\mathcal{P}}p'(\rho)p'(x\mid\rho)\log p'(\rho)p'(x\mid\rho)
\end{aligned} \tag{10}$$

For the sake of briefness and fairness, we assume that our POSG adopts pure sampling in the first sampling stage (POS Sampling), and adopts top-k sampling with the same k in the second sampling stage (Token Sampling). So, $p'(\rho) = p(\rho)$ for $\rho\in\mathcal{P}$, while

$$p'(x\mid\rho) = \begin{cases} \frac{p(x\mid\rho)}{Z_2}, & \text{if } x\in\mathcal{V}_{\rho,k} \\ 0, & \text{otherwise} \end{cases}, Z_2 = \sum_{x\in\mathcal{V}_{\rho,k}} p(x)$$

Note that, in our paper, we denote all the tokens whose POS is ρ as a vocabulary $\mathcal{V}_\rho$, and here, $\mathcal{V}_{\rho,k}$ is the set of top-k most probable tokens in $\mathcal{V}_\rho$. Thus, $\mathcal{V}_{\rho,k}\subseteq\mathcal{V}_\rho$. Then, it follows:

$$\begin{aligned}
H(p)^{\text{(POS)}} &\geq -\log|\mathcal{P}| \\
&-\sum_{x\in\mathcal{V}}\sum_{\rho\in\mathcal{P}}p(\rho)p'(x\mid\rho)\log p(\rho)p'(x\mid\rho) \\
&= -\log|\mathcal{P}| - \sum_{\rho\in\mathcal{P}}\sum_{x\in\mathcal{V}_{\rho,k}}p(\rho)\frac{p(x\mid\rho)}{Z_2}\log p(\rho)\frac{p(x\mid\rho)}{Z_2} \\
&= -\log|\mathcal{P}| - \sum_{\rho\in\mathcal{P}}\sum_{x\in\mathcal{V}_{\rho,k}}\frac{p(x,\rho)}{Z_2}\log\frac{p(x,\rho)}{Z_2}
\end{aligned} \tag{11}$$

Since $\mathcal{V}_{k,\rho}\subseteq\mathcal{V}_{\rho,k}$ and we use the same setting of k, i.e., $Z_2\approx Z_k$, we can finally conclude from Equation 8 and Equation 11 that the lower bound of $H(p)^{\text{(POS)}}$ is greater than or equal to the lower bound of $H(p)^{\text{(top-k)}}$. When compared with other one-stage sampling strategies, this conclusion still holds, and can be proved in a similar way. Consequently, this will account for the effectiveness of our methods.

C Additional Analysis

C.1 Compared with One-stage Sampling

We further conduct an analysis to test whether the traditional one-stage sampling can achieve the

Models	Self-WER↑	Self-BLEU4↓	Distinct↑			BERTScore↑	BLEU4↑	ROUGE↑		
			n=1	n=2	n=3			1	2	L
top-k	100.8	13.6	86.9	88.9	83.7	39.4	6.49	33.5	12.1	32.3
POSG	102.1	13.6	86.9	88.2	83.4	**43.3**	**7.71**	**36.4**	**14.1**	**34.9**
Δ	+1.3	+0.0	+0.0	-0.7	-0.3	**+3.9**	**+1.22**	**+2.9**	**+2.0**	**+2.6**

Table 13: Results of POSG and one-stage sampling (we use top-k sampling here) on the paraphrase generation task. Note that we tune the sampling hyper-parameters of both methods to reach the same level of diversity, and then compare the text quality.

same level of diversity by increasing the randomness, e.g. using larger k in top-k sampling. On the paraphrase generation task, we tune the sampling hyper-parameters in top-k sampling and our POSG to reach the same level of diversity, and then compare the text quality. The results are shown in Table 13. In this experiment, POSG adopts top-k sampling with $k^{(\text{POS})} = 5$ in POS sampling, $k^{(\text{token})} = 500$ in token sampling, while MLE adopts top-k sampling with $k = 1000$. Obviously, our POSG significantly outperforms top-k sampling on MLE in terms of quality metrics, while performing equally well in diversity. Therefore, we can conclude that, by increasing the randomness, the traditional one-stage sampling on MLE can finally achieve the same level of diversity as our POSG, but the quality of the generated text will seriously deteriorate. This further confirms the advantage of our methods over prior works.

C.2 Controllability Analysis Example

An example of the controllability analysis is provided in Table 14. When we control the probability of adjective increasing during the POS sampling stage, the generated paraphrase will contain correspondingly more adjectives.

Input Sentence:	he (PRP) was (VBD) smiling (VBG) , clearly (RB) **delighted (JJ)**
$\times 0.1$	he (PRP) was (VBD) smiling (VBG) , and (CC) he (PRP) was (VBD) clearly (RB) pleased (VBN) with (IN) joy (NN)
$\times 1$	he (PRP) was (VBD) smiling (VBG) and (CC) apparently (RB) **delighted (JJ)** with (IN) joy (NN) in (IN) his (PRP$) face (NN)
$\times 10$	he (PRP) was (VBD) still (RB) smiling (VBG) and (CC) **delighted (JJ)** with (IN) **apparent (JJ)** joy (NN) in (IN) his (PRP$) face (NN)

Table 14: Examples of controllability analysis on the paraphrase generation task.

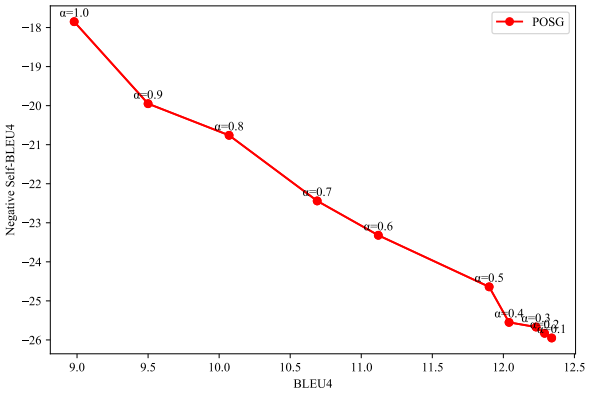

Figure 3: Quality-diversity trade-off for POSG on paraphrase generation by tuning $\alpha^{(\text{POS})}$.

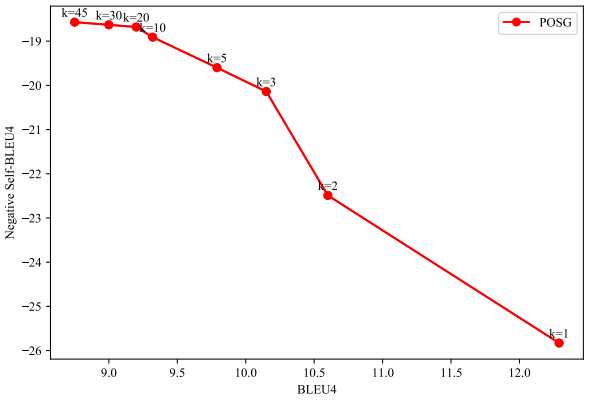

Figure 4: Quality-diversity trade-off for POSG on paraphrase generation by tuning $k^{(\text{POS})}$.

C.3 Tuning $\alpha^{(\text{POS})}$ and $k^{(\text{POS})}$

We have conducted an additional analysis on paraphrase generation by tuning $\alpha^{(\text{POS})}$ or $k^{(\text{POS})}$ while fixing all other hyper-parameters. The results are shown in Figure 3 and Figure 4. It can be observed that: with other sampling hyper-parameters fixed, as $\alpha^{(\text{POS})}$ or $k^{(\text{POS})}$ increases, the results of diversity metrics get better, but the results of quality metrics get worse.

Prefix: Below them, North Koreans continued crossing the river and moving supplies forward to their combat units, some of them already several miles eastward. The North Koreans quickly discovered Task Force Manchu group. They first attacked it at 14:00 that afternoon, and were repulsed
MLE: by the North Koreans. On the morning of September 8, the North Korean forces launched a surprise attack on the high ground west of the Kum River. At 16:30, the North Korean force launched a massive attack on the North Korean force, but the initial attack was not successful. The North Korean offensive was halted by the remnants of the North Korean forces
FACE: by heavy machine-gun fire. In the early morning hours of 8 September, North Korean troops were alerted to attack on the perimeter. On 9 September, a force of 20,000 men led by Lieutenant Colonel Robert E. telluride began to attack the North Korean lines, suffering little damage. By 14:00 on 9 September, North Korean forces had crossed the Naktong River just before midnight.
F^2-Softmax: by the North Koreans. The North Koreans were ordered to withdraw to the rear of the North Koreans. They then launched a frontal attack on the south side of the river. The North Koreans then launched a frontal attack on the North Korean right flank. The North Korean right flank was soon overrun by the North Koreans. The North Koreans were subsequently repulsed by the North Koreans,
UL: by North Korean fire, which forced the North Koreans to retreat. A further assault by the 1st U.S. Infantry Regiment followed in the afternoon, and after seven hours of fighting, the 2nd U.S. Infantry Regiment broke off the attack and retreated across the river. The survivors of the Battle of tellers managed to escape to a new bridge. Task Force presaged, but by 20:00 the North Koreans were completely surrounded by North Korean troops.
MoS: by the 9th Infantry Regiment. At 17 : 00, the North Koreans took the road from the Korean border to the north, and began firing on the northern flank of the North Korean forces. The North Koreans then withdrew to the northern flank of the Korean army, where they advanced into the river and quickly attacked the North Koreans. At 16 : 00, the North Koreans began firing on the North Koreans, and a number of North Korean soldiers, including the 5th Cavalry Regiment, attacked the North Koreans.
POSG: by the North Koreans, beginning their advance south of Osan on 18 September. By nightfall on 24 September, Ho Chi Minh had secured its flank, while the South Koreans had captured the town of Phong on the west of Taejon. The North Koreans had retreated to Pyongtaek, and in the afternoon of 22 September two North Koreans were killed there, leaving behind the town to the survivors.

Table 15: Examples of language modeling on Wikitext-103 dataset. Repeating text is highlighted in blue, dull text with single context is highlighted in orange, and incoherent text is highlighted in red.

Source: this is going to make good economic sense for the city . **Reference**: that it would be good for the city in a certain economic sense .
MLE: this will be an economic sense for the entire city .
FACE: this will create a good economic point in the city .
F^2-Softmax: this will make sense of economic sense for the city .
UL: this will be considerable economic considerations for the city 's going to be able to economic point of the city .
SGCP: this will make economic sense for the city .
POSG: it is what makes good economic sense to the city .

Table 16: Examples of paraphrase generation on ParaNMT-50M.

D Case Study

Table 15 provides examples of text completion produced by our model and other baselines. It can be observed that MLE, F^2-Softmax, and MoS suffer from a severe repetition problem, and they also generate many similar sentences about a single content. Due to the low-diversity problem, MoS even generates some illogical text, such as "the North Koreans began firing on the North Koreans". FACE produces a large amount of incoherent text, making the text somewhat hard to read. UL and our POSG alleviate those problems, while our model performs relatively better.

Additionally, examples of paraphrase generation are shown in Table 16. We observe that almost all models can generate high-quality paraphrases with well-preserved semantic meanings, while our POSG exhibits more syntactic diversity than other baselines.

E Human Evaluation

We post the human evaluation questionnaire, as shown in Table 19 and Table 20, and then recruit five workers with sufficient high English skills. We pay each worker 60 US dollars, and let them complete the evaluation within a week.

For both tasks, workers are given 100 randomly sampled inputs, and corresponding outputs from each model. Then, they need to score those outputs according to the description in the questionnaire. The term "diversity" in language modeling is typically regarded as a property of the collective outputs of a system, but it is really difficult for a human to remember such a large scale of outputs and give an overall score for a system. So we make a compromise that we asked the worker to rate the diversity of individual outputs, and intuitively the more diverse individual outputs are, the more diverse the system is.

We employ the Krippendorff's alpha for the inter-annotator agreement analysis. As shown in Table 17 and Table 18, all the results are fair agree-

ment ($0.2 \leq \kappa \leq 0.4$) or moderate agreement ($0.4 \leq \kappa \leq 0.6$).

	Div.	Qua.
Krippendorff's α	0.57	0.40

Table 17: Agreement analysis for annotators labels on the language modeling task.

Models	Div.		Flu.	Rel.
	Lex.	Syn.		
Krippendorff's α	0.54	0.37	0.71	0.63

Table 18: Agreement analysis for annotators labels on the paraphrase generation task.

F Impact of the POS tagger

In our work, we use an off-the-shelf POS tagger to annotate the POS tags, and build the POSG upon these annotated POS tags. Consequently, the better the quality of POS tagging, the better the performance of our method. Stanford CoreNLP's POS tagger (Manning et al., 2014), the POS tagger we use, is one of the state-of-the-art tagger, which is the most commonly used tool for NLP research. This ensures the high quality of tagging results.

G Impact Statement

Our work has developed generic generation methods to promote text diversity while maintaining comparable quality. Therefore, despite the contributes to better text generation, our proposed methods may be used to generate more human-like fake text. But the impacts are more apparent when considering deployed applications, while our proposed methods as the methodologies can not have any direct negative societal impacts. Moreover, all the datasets we used in our work are open source datasets. Wikitext-103 was extracted from Wikipedia, and ParaNMT-50M was created by the back-translation. Therefore, the data we used would not contain personally identifiable information or offensive content.

The goal of this review is to evaluate the quality and diversity of generated texts. In this review, you will read an excerpt from Wikipedia with first 50 words as prefixes, and its corresponding 100-word continuations. You should rate the continuations between 1 - 5 in two ways:
(1) Diversity. The overall diversity of text can be evaluated from form (How to say it?) and content (What to say?). (1 = The continuation is always repeating some words, its sentences share the similar forms syntactically and lexically, and its content is dull; 5 = The continuation seldom repeats words, its sentences have various syntactical and lexical forms, and it contains different things related to the prefix)
(2) Quality. The overall quality of text can be evaluated in many different aspects, such as fluency, readability, coherence, and so on. (1 = The continuation is incoherent, difficult to understand, not related to the prefix, and has many syntactically and semantically errors; 5 = The continuation is coherent, easy to understand, related to the prefix, and grammatically correct)
You should score between 1 - 5, where 5 is best and 1 is worst. You can consider and make a final decision by comparing different continuations of the same prefix. These prefixes and continuations have been preprocessed by separating punctuation, and splitting conjunctions. And because of length constraints, they may be truncated in the middle of the text. **Please ignore these when evaluating and do not allow them to affect your judgments.**

For example, if you are given the following prefix and continuations:

Prefix: Infantry fought a series of hard battles . Simultaneously it had to send combat patrols to its rear to clear infiltrating North Koreans from Changnyong and from its supply road . On the morning of September 1 the 1st and 2nd Regiments of the NK 9th Division , in their. . .

Continuation 1: Score: Diversity 3, Quality 3

second action around 380 yards <unk> 290 m <unk> from .395 , were assigned to the 7th Division. Meanwhile , the 2nd Regiment of the 27th Infantry , which had been sent out of Murmansk in late October , had moved to iconoclasts Pass . It was the first major US unit to advance across the mountains to the west , but it was not successful . On the morning of September 2 , the Eighth Army's 2nd Battalion , 5th Cavalry Regiment was moved into a position about 10 miles <unk> 16 km <unk> from Highton .

Continuation 2: Score: Diversity 1, Quality 5

reserve , were positioned to engage the North Korean troops . The 3rd Regiment of the NK 10th Division was to attack the North Korean forces . The 1st Regiment of the NK 12th Division was to attack the North Koreans from the north and east of the 2nd Division , and the 1st Regiment of the NK 15th Division was to attack the North Koreans from the south . The 1st Regiment of the NK 12th Division was to attack the North Koreans from the south . The 1st Regiment of the NK 15th Division was to attack the

Continuation 3: Score: Diversity 1, Quality 1

position north of Wonju , were repeatedly pushed back by the ROK 3rd Division . At 08 : 00 am the units of the 1st Battalion attempted to attack . Kim of the 1st and 2nd Battalions attacked the 3rd and 3rd Battalions of the 2nd Battalion of the 3rd Battalion of the 3rd Battalions of the 1st Battalion of the 2nd Battalion of 2nd Battalion , 7th Marines on North , 7th Marines on Hill 60 . Task Force 51 and 9th Marines attacked Sangju 's 1st Battalion of the 3rd Battalion of the 2nd Battalion , 1st Platoons

Continuation 4: Score: Diversity 4, Quality 3

" Series B " Company , carried out three assaults on the Pusan on 29 September against three resistance groups that included the blacksmiths , truck commanders , and air support . They then conducted three raids into a line south of psalmody by the 2nd Battalion , 3rd Field Artillery Regiment . At the same time , units from the 3rd Infantry Division and the 3rd Marine Division advanced on all four sides of the road , while infantry units of the 2nd Infantry Division advanced on the northern slope . The 5th Marine Corps , in particular

Analysis: As for diversity, Continuation 1 gives various details about the "hard battles", and is of high diversity in the text form. But all the content of it is about the deployment of armies, which means low content diversity. Therefore, Continuation 1 gets 3 points in Diversity. Since there are some words difficult to understand (highlighted in red), Continuation 1 gets 3 points in Quality.
Continuation 2 keeps talking about only one single content, that is "some Regiment attacks the North Koreans from somewhere" (highlighted in orange). Although it is fluent, relevant, and gets high scores in Quality, Continuation 2 will receive the lowest score in Diversity due to its dull content.
Continuation 3 contains many useless repeating text (highlighted in blue), which makes the continuation incoherent and hard to understand, so it gets the lowest score in both Quality and Diversity.
Continuation 4 also states from many different aspects of the "hard battles", but compared to continuation 1, it is not that diverse (That's why comparing different continuations can help to make a decision). Therefore, it gets 4 points in Diversity. In the meantime, high diversity of it also leads to some strange words in the text, and affects the overall quality. So, Continuation 4 can only get a mediocre score in Quality.

Table 19: Human evaluation questionnaire for language modeling.

The goal of this review is to evaluate the quality and diversity of text paraphrase dataset. In this review, you will be given an original sentence, and its corresponding paraphrases. You should rate the paraphrases between 1 - 5 in four ways:
(1) Lexical Diversity: how lexically diverse are the generated sentences?
(2) Syntactical Diversity: how syntactically diverse are the generated sentences?
(3) Fluency: how fluent are the generated paraphrases?
(4) Relevance: how semantically consistent are between generated paraphrases and the input sentences?
You should score between 1 - 5, where 5 is best and 1 is worst. You can consider and make a final decision by comparing different paraphrases of the same original sentence. These sentences have been preprocessed by converting all letters to lowercase, separating punctuation, and splitting conjunctions.
Please ignore this when evaluating and do not allow it to affect your judgments.

For example, if you are given the following original sentence and paraphrases:
Original sentence: by adopting rules that regulate the information about the foods and their nutritional value appearing on the label , the consumers will be able to make informed and meaningful choices .
Paraphrase 1: Score: Lexical Diversity 5, Syntactical Diversity 5, Fluency 3, Relevance 5
the rules will be able to adapt food and their nutritional values listed on the labelling of consumers will be able to be able to make informed and they are appropriate assessment .
Paraphrase 2: Score: Lexical Diversity 1, Syntactical Diversity 2, Fluency 1, Relevance 2
by adopting rules governing the information about food and relevance of foods and nutritional value of nutrition value that regulate the labelling , so that consumers .
Paraphrase 3: Score: Lexical Diversity 4, Syntactical Diversity 5, Fluency 1, Relevance 2
consumers can adopt rules to provide informed and nutrition value of the food and their nutritional values listed on the labelling , consumers will be able to enable consumers .
Paraphrase 4: Score: Lexical Diversity 1, Syntactical Diversity 1, Fluency 1, Relevance 1
by adopting rules that regulates the rule of food and their nutritional value of food and their nutritional value of their nutritional value to the consumer protection , consumers .

Analysis: Although there are also some strange words in Paraphrase 1, we can still capture the main meaning of it. Therefore, Paraphrase 1 can get a mediocre score in Fluency and a high score in Relevance. On the other hand, Paraphrase 1 has many lexical edits and turns the original sentence into two parallel sentences, so it can full marks in both terms of Lexical and Syntactical Diversity.
Paraphrase 2 is not really finished and repeats some words in the text (highlighted in blue), so it gets the lowest scores in Relevance and Fluency. Meanwhile, except for some incorrect word order transpositions, Paraphrase 2 is very similar to the original sentence. Therefore, it receives low scores in Lexical and Syntactical Diversity.
Obviously, Paraphrase 3 changes a lot lexically and syntactically. However, it is incoherent, difficult to understand (highlighted in red), so Paraphrase 3 scores high for Lexical and Syntactical Diversity and low for Fluency and Relevance.
Paraphrase 4 is a nonsensical text, which is not really finished and keeps repeating itself. Therefore, it gets the lowest scores from all aspects.

Table 20: Human evaluation questionnaire for paraphrase generation.

Enhancing Pre-trained Models with Text Structure Knowledge for Question Generation

Zichen Wu, Xin Jia, Fanyi Qu, Yunfang Wu*
Key Laboratory of Computational Linguistics, Ministry of Education, China
School of Computer Science, Peking University, China
{wuzichen,jemmryx,fanyiqu,wuyf}@pku.edu.cn

Abstract

Today the pre-trained language models achieve great success for question generation (QG) task and significantly outperform traditional sequence-to-sequence approaches. However, the pre-trained models treat the input passage as a flat sequence and are thus not aware of the text structure of input passage. For QG task, we model text structure as answer position and syntactic dependency, and propose answer localness modeling and syntactic mask attention to address these limitations. Specially, we present localness modeling with a Gaussian bias to enable the model to focus on answer-surrounded context, and propose a mask attention mechanism to make the syntactic structure of input passage accessible in question generation process. Experiments on SQuAD dataset show that our proposed two modules improve performance over the strong pre-trained model ProphetNet, and combing them together achieves very competitive results with the state-of-the-art pre-trained model.

1 Introduction

Question generation (QG) aims to generate questions for a given passage, which is a challenging task and shows great value in many practical applications. For instance, QG helps in building reading comprehension tests for course assessments (Kurdi et al., 2020); QG can be an important skill for chatbots to start a conversation with human users (Mostafazadeh et al., 2016); QG can help reduce the human labor for collecting large-scale question-answer datasets to improve question answering systems (Duan et al., 2017).

Existing QG methods mostly employ neural network models by treating it as a sequence-to-sequence generative task, where researchers have tried to incorporate text structure to improve the performance. For answer-aware QG, the answer position is widely explored. Song et al. (2018) propose three strategies to match the answer with passage; Sun et al. (2018) model the relative distance between context words and answer to help generate answer-related questions; Ma et al. (2020) enhance model's ability in capturing semantic and positional information via multi-task learning. Some other neural network models dig out deep text structure for generating better questions. For instance, Chen et al. (2020) construct a passage graph to utilize text structure; Pan et al. (2020) construct a semantic-level graph to learn the global structure of a document for deep QG. However, most of these methods adopt standard neural network, and few of them are based on the strong pre-trained models.

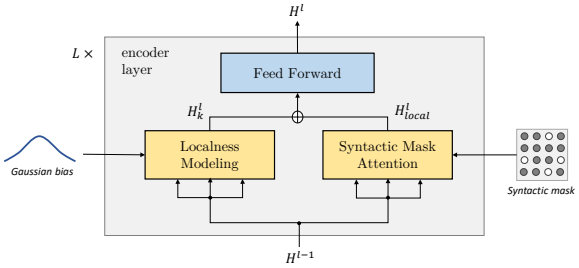

Figure 1: An overall structure of our proposed methods

Nowadays, pre-trained models largely boost the performance of QG systems (Dong et al., 2019; Bao et al., 2020; Qi et al., 2020; Xiao et al., 2020), which utilize huge amounts of unlabeled text corpora in pre-training stage to learn general language representations, and then be fine-tuned in supervised QG tasks. However, in most cases, the input passage and answer are formatted as "answer [sep] passage" to feed into the pre-trained model during the fine-tuning stage, as a result that the neural network cannot explicitly capture the syntactic structure of sentence and is not aware of the answer position (Zhou et al., 2017), which is vital to generate high-quality answer-aware questions. In this end, how to incorporate text structure into

*Corresponding author.

Proceedings of the 29th International Conference on Computational Linguistics, pages 6564–6574
October 12–17, 2022.

pre-trained models for QG remains an open issue.

Fortunately, several previous works have tried to utilize external knowledge and position information for NLP tasks. For instance, Yang et al. (2018) introduce token localness in attention mechanism for machine translation tasks, and Liu et al. (2020) propose a knowledge-enabled language model (K-BERT) to incorporate domain knowledge for question answering systems.

Motivated by these works, we propose a novel strategy to explicitly incorporate syntactic knowledge and answer information to pre-trained models for QG task. The overall architecture is illustrated in Figure 1. We present a localness modeling by designing a Gaussian bias to regulate the attention calculation, and propose a new attention mechanism to incorporate syntactic structure. Specially, in localness modeling, we adopt answer position as the center of Gaussian distribution and predict window size automatically, in order that the content around answer will be more focused and the range can be adjusted dynamically according to different token length. For syntactic knowledge, we first compute a syntactic vector through syntactic mask attention, which we design to blind the tokens outside of dependency triples, and then constitute a syntactic-aware representation by adding syntactic vector to the original context vector.

We base our method on the strong pre-trained model ProphetNet (Qi et al., 2020), which achieves state-of-the-art results on many generation tasks including QG. We conduct experiments on SQuAD dataset (Du et al., 2017). The automatic evaluation shows that our model boosts the performance of pre-trained models and achieves comparable state-of-the-art performance. Further human judgement demonstrates that our model produces high-quality questions by capturing syntactic knowledge and answer-surrounded context.

Since our method is only a modification to the self-attention module, and there is little increase in parameter number, one can easily apply our method to other Transformer-based pre-trained models while almost keeping the original speed without any much computation resources in the training stage. Our codes and data will be publicly available at the Github repository.

Contributions: To summarize, we (1) propose a modified attention mechanism to incorporate syntactic structure and position information, which can be easily applied to any Transformer-based pre-trained language models for QG task; and (2) with selection of syntactic knowledge, our model is able to reach comparable state-of-the-art performance for question generation.

2 Related Work

2.1 Question Generation

Recently, neural question generation approaches have become popular and achieved great success. Serban et al. (2016) first introduce an encoder-decoder framework with attention mechanism to generate questions for facts from FreeBase. Du et al. (2017) employ a seq2seq architecture on QG. Zhao et al. (2018) tackle the challenge of processing long text inputs with a gated self-attention encoder. Song et al. (2018) and Kim et al. (2019) strength model's ability by leveraging answer information. Meng et al. (2020) propose a two-stage QG model by creating a draft first and then doing refinement. These works adopt sequence-to-sequence generative approaches and yield many good results on QG. Benefiting from the development of pre-trained model, the performance of QG has been significantly improved. Varanasi et al. (2020) utilize the information from a self-attention module in BERT (Devlin et al., 2019) to generate questions. Qi et al. (2020) propose a pre-training model with n-stream self-attention mechanism and achieve notable results in QG. Xiao et al. (2020) design a span-by-span generation flow to predict semantically-complete spans consecutively. So far, Bao et al. (2020) achieve the best result on SQuAD dataset, which employ autoencoding and partially autoregressive modeling as the pre-training task.

2.2 Knowledge-Enhanced Text Generation

Many knowledge-enhanced text generation systems have achieved great performance on generating informative texts (Yu et al., 2020). In dialogue systems, topic-aware seq2seq model helps to generate more informative responses (Mou et al., 2016; Xing et al., 2017; Zhou et al., 2018). In story generation, commonsense knowledge helps to capture the story-line and narrates the plots (Guan et al., 2019). Liu et al. (2020) and Huang et al. (2021) incorporate external knowledge base into the Transformer-like architecture.

For question generation, Zhou et al. (2017) and Sun et al. (2018) leverage answer embedding and position information to help generate more answer-related questions; Du and Cardie (2018) incorpo-

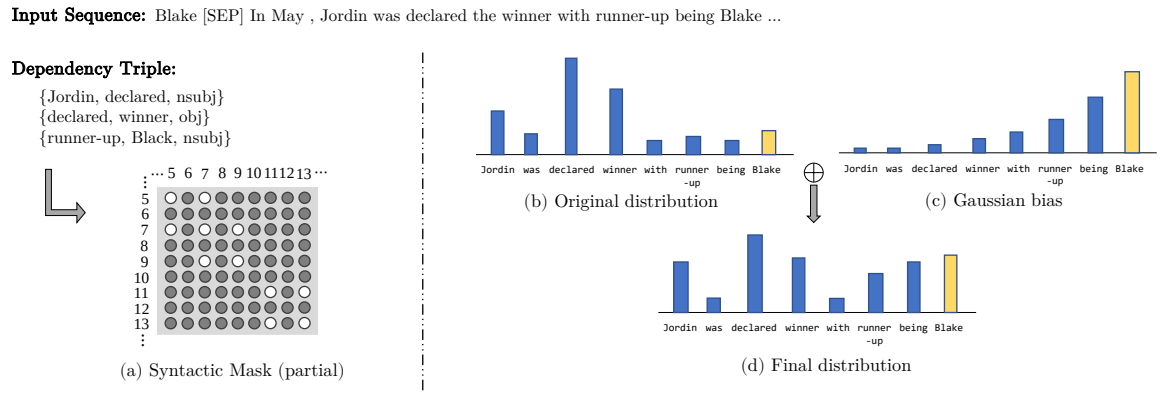

(a) Syntactic Mask (partial)

(b) Original distribution

(c) Gaussian bias

(d) Final distribution

Figure 2: A diagram of Syntactic Mask (left) and Localness Modeling (right). (1) In subfigure (a), we show a partial mask attention with index ranging from 5 to 13 (representing tokens from 'Jordin' to 'Black' in the input sequence), where the black dot denotes value $-\infty$ and the white denotes 0. (2) Subfigures (b)-(d) show an example of distribution changes in calculating attentions weights of token 'declared', where we convert G to the form of distribution for better illustration.

rate coreference into QG model; Kim et al. (2019) propose a keyword-net to help the model better capture important information; Jia et al. (2020) integrate paraphrase knowledge into QG systems to generate more human-like questions. Chen et al. (2020) and Pan et al. (2020) try to capture text structure via graph neural network for QG.

The attention mechanism has been widely utilized to incorporate knowledge representation, which is obtained by calculating attention over knowledge representations. Multitask training is also an alternative way to incorporate knowledge (Kim et al., 2020; Jia et al., 2020).

In this paper, based on the pre-trained model for QG, we further incorporate dependency relations and answer position, which can be regarded as structure knowledge. Our task is more challenging since most of the previous works are based on the Transformer only. In order not to increase the computation consumption, we don't use any other sophisticated neural networks beyond the pre-trained model itself.

3 Preliminaries

3.1 Self-Attention

Here we briefly introduce the encoder of Transformer architecture, on which our methods will base. Given an input sequence $x = \{x_1, x_2, \cdots, x_t\}$, the transformer encoder maps these tokens into hidden representations. More concretely, the $l-1$-th layer hidden states are first linearly transformed into query vectors q, key vectors k and value vectors v by multiplying three separate weights, and then obtain the l-th layer hidden states via attention mechanism:

$$H^l = \text{Attn}(q, k, v) \quad (1)$$

$$\text{Attn}(q, k, v) = \text{softmax}(\frac{scores}{\sqrt{d}})v \quad (2)$$

$$scores = qk^T \quad (3)$$

where $\sqrt{d}$ is the scaling factor.

3.2 Pre-trained Language Model: ProphetNet

We build our method based on the strong pre-trained model ProphetNet (Qi et al., 2020), which achieves outstanding results on QG tasks. Compared to conventional pre-training models which predict the next one token at each step, ProphetNet tends to predict the next n tokens simultaneously to prevent overfitting on local correlations:

$$p(y_t|y_{<t}, x), \cdots, p(y_{t+n-1}|y_{<t}, x) = \text{Decoder}(y_{<t}, H_{enc}) \quad (4)$$

where H_{enc} is the hidden state obtained by the encoder. To achieve this, ProphetNet adopts a n-stream self-attention mechanism to predict next n continuous future tokens, with each stream representing a next i-th predicting.

4 Integrating Text Structure

Our work focuses on answer-aware QG task. Formally, denote the passage, question and answer as P, Q and A respectively, the task is defined as:

$$Q^* = \underset{Q}{\text{argmax}}\, P_\theta(Q|P, A) \quad (5)$$

Practically, we concatenate passage and answer by format "A [SEP] P" as the input of ProphetNet, to generate Q^* as output question. A diagram of our method is illustrated in Figure 2.

4.1 Answer Localness Modeling

In order to enhance model's capability in capturing more local information around the answer and enable the model to generate more answer-related questions, we model localness as a preference bias to regulate the original attention weight in encoder end, which can be defined as:

$$\mathrm{Attn}(q, k, v) = \mathrm{softmax}(\frac{scores + G}{\sqrt{d}})v \quad (6)$$

where $G \in \mathbb{R}^{I \times I}$ is an alignment matrix, and I refers to the length of input sequence. Each element $G_{i,j}$ represents an alignment preference of token w_i to token w_j, which is calculated as:

$$G_{i,j} = -\frac{(j - P_c)^2}{2\sigma_i^2} \quad (7)$$

where P_c denotes the central position of answer appearing in the input passage and σ_i denotes the standard deviation, which is half of the window size D_i. Denote the start and end index of the answer span as s and e, then P_c is calculated as:

$$P_c = \frac{s + e}{2} \quad (8)$$

Consequently, after softmax operation, G will become a Gaussian distribution, indicating that the tokens closer to the central position will get higher attention weights.

The window size D_i measures the major area that token x_i needs to align with. We set D_i to be a variable based on the index i rather than a constant value, which we think will bring our model more flexibility to adjust the concentration area according to different token information. Following Yang et al. (2018), we map query vectors into D_i through a feed-forward neural network:

$$z_i = U_d^T \tanh(W_p q_i) \quad (9)$$

$$D_i = I \cdot \mathrm{sigmoid}(z_i) \quad (10)$$

where $W_p \in \mathbb{R}^{d \times d}, U_d \in \mathbb{R}^d$ are learnable parameters, and I is a scalar factor regulating D_i to a real value between 0 and the length of input sequence.

Thus, we can obtain hidden states of the input sequence with strengthened localness information:

$$H_{local} = \mathrm{MultiHead}(q, k, v, G) \quad (11)$$

Figure 2(b)-(d) shows an example of answer localness modeling. When calculating the attention weights of token 'declared', we first learn a Gaussian distribution centered on the answer 'Blake'. The original attention distribution is then regularized to form the final distribution, which not only pays attention to token 'winner' but also to token 'runner-up' and 'Blake'. As a result, the model is guided to attend to the phrase 'winner with runner-up being Blake'.

4.2 Syntactic Mask Attention

In this module, we strengthen the syntactic structure of input sentence accessible in question generation process. The whole procedure for incorporating syntactic knowledge can be divided into three steps: 1) extract appropriate syntactic relations; 2) build syntactic mask and 3) apply syntactic mask to guide the attention calculation. We will introduce them respectively.

4.2.1 Extract Syntactic Relations

We extract dependency relations and explicitly introduce them into the generation process. Due to the huge time needed for parsing, we only select some key sentences to do syntactic parsing rather than the whole passage. For every answer-passage pairs, we select the sentence from the passage where answer spans are located as our key sentence. If the answer spans do not appear completely in any sentences (this is possible because the sentence splitting toolkit is not perfect and sometimes it will mistake punctuation for a terminator), we will select the most similar sentence by computing ROUGE score as an alternative.

Then we apply dependency parser to these selected key sentences, and choose some major relations as the final dependency knowledge, including pred, subj, nsubjpass, csubj, csubjpass, obj, iobj and xcomp. The selection of dependency relations is based on de Marneffe et al. (2014). We have also tried some other relations, the analysis of which is given in Section 6.2. The dependency relation is presented in the form of triples, which can be denoted as $\varepsilon = \{x_i, x_j, r_k\}$, where x_i and x_j are tokens from the input sequence, and r_k denotes the relation between them.

In our experiment, we utilize the Stanford NLP toolkit [1] to do tokenization, sentence splitting and dependency parsing. A more powerful and sophisticated procedure to extract dependency triples might

[1] https://nlp.stanford.edu/software/

capture the syntactic structure more accurately, and we leave this to future work.

4.2.2 Syntactic Mask

Building syntactic mask is the key process in calculating knowledge context vector. Based on the extracted syntactic relation triples, we design a visible mask that prevents some tokens from seeing other ones, which can be defined as below:

$$M_{ij} = \begin{cases} 0, & x_i \ominus x_j \\ -\infty, & x_i \oslash x_j \end{cases} \tag{12}$$

where $x_i \ominus x_j$ means x_i and x_j are in a relation triple, while $x_i \oslash x_j$ means they are not.

We display an example in Figure 2(a) to show how we construct the syntactic mask from a sentence. We construct the relation triples like {'Jordin', 'declared', 'nsubj'} through extracting dependency relations and build syntactic mask to ensure 'Jordin' and 'declared' could see each other while blind to tokens outside the triples. And tokens not appearing in any triple, like 'was', 'the' and 'being', are not visible to each other.

4.2.3 Mask Attention

The process of mask attention is actually an extension to the original self-attention module in encoder end, whose attention score is calculated as:

$$scores = qk^T + M \tag{13}$$

where M is the syntactic mask constructed above.

After the softmax function, the attention score where $M_{ij} = -\infty$ will be 0, indicating that the hidden state of x_i will make no contribution to x_j. In this way, we build a syntactic context vector that gathers information of tokens appearing in the extracted syntactic relations while discards unrelated information, which can be a supplementary to the original context vector. Consequently, we obtain the knowledge context vector by:

$$H_k = \text{MultiHead}(q, k, v, M) \tag{14}$$

Then the syntactic context vector will be added to the original vector to form the final syntactic-aware representation:

$$\tilde{H} = H_{local} \oplus H_k \tag{15}$$

4.3 Question Generation

After obtaining the encoder hidden states $\tilde{H}$, we pass them to the decoder and get the distribution on vocabulary at each step (defined in Equation 4). During the training stage, the loss is calculated as:

$$\mathcal{L} = -\sum_{j=0}^{n-1} \alpha_j \left(\sum_{t=1}^{T-j} \log p_\theta(y_{t+j} | y_{<t}, x) \right) \tag{16}$$

which can be viewed as the weighted sum of n future token predicting loss. And during the inference stage, we disable n-gram predicting and generate a sequence with the highest likelihood:

$$Q = \underset{y}{\text{argmax}}\, p_\theta(y_{1:t} | x) \tag{17}$$

5 Experimental Setup

5.1 Dataset

We conduct experiments on the widely-used reading comprehension dataset SQuAD (Rajpurkar et al., 2016). It consists of questions posed by crowd workers on Wikipedia articles, and the answer to every question is a span extracted from the given passage. There exist different split sets on SQuAD. Following the work of ProphetNet (Qi et al., 2020), we do experiments on two splits: (1) Du split (Du et al., 2017), which splits SQuAD 1.1 dataset into training, development and test sets, consisting of 75,722, 10,570, 11,877 instances respectively; (2) Zhao split (Zhao et al., 2018), which uses the reversed dev-test setup as opposed to the original setup. Following previous work for QG, we adopt BLEU-4 (Papineni et al., 2002), Meteor (Denkowski and Lavie, 2014) and Rouge-L (Lin, 2004) as evaluation metrics.

5.2 Training Details

We adopt most of the configurations in ProphetNet released by Microsoft group [2]. However, due to computation resource limitation, we reimplemented their codes with some different configurations. We use one NVIDIA GeForce RTX 2080 Ti to support fine-tune, and during the training stage we truncate overlong input sentences to 512 tokens [3] and the batch size is set to 12. These might bring some decrease in performance, and we take the new results as our baseline for fair comparison. Inspired

[2] https://github.com/microsoft/ProphetNet

[3] Since ProphetNet only adopts input strings less than 512 tokens.

Methods (conference-year)		Du split			Zhao split		
		BLEU-4	Meteor	Rouge-L	BLEU-4	Meteor	Rouge-L
Unpre-trained	s2s (ACL-17)	12.28	16.62	39.75	-	-	-
	CorefNQG (ACL-18)	15.16	19.12	-	-	-	-
	MP-GSN (EMNLP-18)	-	-	-	16.38	20.25	44.48
	SemQG (EMNLP-19)	18.37	22.65	46.68	20.76	24.20	48.91
	RefNet (EMNLP-19)	-	-	-	18.16	23.40	47.14
	ParaphraseQG (ACL-20)	17.21	20.96	-	-	-	-
	EGSS (AAAI-21)	18.93	22.04	47.73	-	-	-
Pre-trained	ERINE-GEN$_{BASE}$ (IJCAI-20)	22.28	25.13	50.58	23.52	25.61	51.45
	CopyBert (ACL-20)	22.71	24.48	51.60	-	-	-
	ProphetNet (EMNLP-20)	23.91	**26.60**	52.26	25.80	**27.54**	53.65
	UniLM-v2 (ICML-20)	**24.43**	26.34	51.97	26.29	27.16	53.22
	Our model	24.37	26.26	**52.77**	**26.30**	27.25	**53.87**

Table 1: Experimental results on SQuAD dataset comparing with previous work

by Yang et al. (2018), who show pre-trained models tend to capture localness information in shallow layers while global in higher layers, we apply localness modeling in the first 4 encoder layers and syntactic attention mask in all 12 encoder layers[4].

5.3 Comparing Methods

There is a large number of work for QG on SQuAD dataset. Our focus in this paper is to enhance pre-trained models with text structure, so we mainly compare our method with other pre-trained language models. Also, we list some notable sequence-to-sequence works for reference, which did experiments on Du split or Zhao split.

The unpre-trained models for QG. s2s (Du et al., 2017): an attention-based seq2seq framework for QG. CorefNQG (Du and Cardie, 2018): incorporating coreference knowledge into seq2seq model. MP-GSN (Zhao et al., 2018): addressing the challenge of processing long text inputs. SemQG (Zhang and Bansal, 2019): introducing two semantic-enhanced rewards to regularize generation. RefNet (Nema et al., 2019): a two stage model which creates an initial draft first and then refine it. ParaphraseQG (Jia et al., 2020): incorporating paraphrase knowledge into question generation by back-translation. EGSS (Huang et al., 2021): an entity guided question generation model.

The pre-trained models applied to QG. ERNIE-GEN (Xiao et al., 2020): using multi-granularity target sampling in pre-training and span-by-span generation flow in predicting stage. CopyBert (Varanasi et al., 2020): utilizing information from self-attention modules of BERT in generation. ProphetNet (Qi et al., 2020): our baseline model, as described before. UniLM-v2 (Bao et al., 2020): pre-trained of bi-directional language modeling via auto-encoding and seq2seq generation via partially autoregressive modeling.

[4]In fact, we also conducted experiments on different number of layers, and the results will be reported in our Github repository.

6 Results and Analysis

6.1 Main Results

The main experimental results are shown in Table 1. For QG on SQuAD dataset, there exists a significant performance gap between the unpre-trained and pre-trained models. The best pre-trained model, UniLM-v2, achieves a 24.43 BLEU-4 on Du split, which receives 5.5 point improvement compared with the best seq2seq model EGSS. The ProphetNet model achieves a competitive BLEU-4 with UniLM-v2, and yields the best result on Meteor and Rouge-L among existing pre-trained models.

Considering the excellent performance of these pre-trained models, it's exciting to see that our proposed method with syntactic mask and localness modeling brings further improvement over the basic ProphetNet model, and obtains state-of-the-art results on Zhao split and a close performance with UniLM-v2 on Du split. It's worth noting that our model yields best results in terms of Rouge-L on both datasets among all existing works.

6.2 Ablation Study & Analysis

Ablation Study We conduct experiments by applying syntactic mask and localness modeling separately to study their effects on the baseline model. There is a tiny decrease between the result reported by Qi et al. (2020) and our reproduced result on some metrics, which we think is reasonable since we adopt some different configurations foremen-

Method	Du split			Zhao split		
	BLEU-4	Meteor	Rouge-L	BLEU-4	Meteor	Rouge-L
ProphetNet*	23.91	25.95	52.28	25.71	26.99	53.63
ProphetNet + Syntactic Mask	24.22	26.24	52.60	26.20	27.01	53.74
ProphetNet + Localness	24.11	26.01	52.52	25.88	**27.28**	53.62
ProphetNet + Syn. Mask + Localness	**24.37**	**26.26**	**52.77**	**26.30**	27.25	**53.87**

Table 2: Ablation study results by applying different modules on top of the pre-trained ProphetNet model. We report the mean over 3 random seeds. *We re-implemented the ProphetNet released code[2] and the results are a little lower than the original paper. Underline represents the value is better than baseline with significance ($p < 0.05$).

tioned in Section 5.2. As illustrated in Table 2, both components separately enhance the performance of basic ProphetNet and obtain better scores on almost all metrics on two data splits. The performance is further improved when combing two modules together, with a 0.56 BLEU-4 increase on Du split and 0.59 on Zhao split. It indicates that syntactic relations and position information are from two different feature subspace and are able to complement each other in enhancing model's awareness of text structure. And this combining improvement is guaranteed by conducting significance test.

Analysis on Dependency Relations We conduct experiments to investigate the effect of different dependency relations. According to de Marneffe et al. (2014), the dependency relations can be divided into two categories: core arguments and non-core arguments, and core arguments can be further divided into nominal relations and clause relations. Inspired by this, we adopt three strategies to select dependency relations: 1) using all dependency relations, which is equal to mask any other tokens except for those in the key sentence; 2) using core arguments, as in Section 4.2.1, which keeps the stem of sentence and gets rid of attributive words; 3) only using nominal relations in core arguments, which removes the clause relation of the stem and thus is the totally plain trunk of the sentence. The choices of core arguments are based on de Marneffe et al. (2014).

The experimental results are shown in Table 3. All three strategies bring performance gain over the basic ProphetNet, which validates the effectiveness of our syntactic mask attention. Especially, strategy 2) achieves the best results on both data splits, which provides appropriate dependency knowledge. In contrast, too much dependency relation incorporation may divert the sentence from its correct meaning, which can be found in Strategy 1), and the lack of dependency structure cannot produce a marked effect, which Strategy 3) exemplifies.

Dependency Relations	Du split	Zhao split
ProphetNet	23.81	25.71
+ all relations	23.93	26.17
+ core arguments	24.22	26.20
+ core nominal relations	24.01	26.13

Table 3: Experimental results of different dependency relations in the syntactic mask attention module.

Methods	Du split	Zhao split
ProphetNet	23.81	25.71
+ predicting center	23.77	25.72
+ answer center	24.11	25.88

Table 4: Experimental results of different center strategy in the localness modeling module.

Analysis on Localness Modeling Further, we conduct experiments on different methods to set center position of Gaussian distribution in localness modeling: (1) following Yang et al. (2018), who predict the central position by applying a feed-forward transformation to query vector

$$p_i = U_p^T \tanh(W_p q_i) \tag{18}$$

where W_p is a set of parameter to learn; (2) adopt the position of answer as center, as Equation (8).

The experimental results are shown in Table 4. Automatically predicting center position doesn't bring performance gain, while using answer center gets better results on both data splits. We argue that this is because mapping query vector to predict center will lead one token to align with other tokens that are similar with it, which is not applicable in our task. For answer-aware QG, we need each token to pay enough attention to the context around answer, so that the model will obtain better representations with answer as guidance.

Analysis on ProphetNet One may notice the contradiction between the goal of ProphetNet and our methods: ProphetNet predicts the next several

		Base	Ours	ρ
Fluency	No	4.33%	4.67%	
	Med.	10.33%	9.33%	0.428
	Yes	85.33%	86%	(3.8e-06)
	Avg.	2.81	2.81	
Relevancy	No	1.67%	1%	
	Med.	14.33%	13.67%	0.507
	Yes	84%	85.33%	(2.4e-07)
	Avg.	2.82	2.84	
Answerability	No	4%	3%	
	Med.	12%	10.67%	0.456
	Yes	84%	86.33%	(1.8e-06)
	Avg.	2.80	2.83	

Table 5: Human evaluation results on the generated questions by ProphetNet ("Base") and our full model ("Ours"). Avg. represents the average score of 100 samples. The last column is the Spearman coefficients with p-values in the parentheses.

Passage: ...Meanwhile, a cargo train carrying 13 petrol tanks derailed in Hui County, Gansu, and caught on fire after the *rail was distorted*.
Answer: rail was distorted
Generated Questions:
Golden: why did the train catch fire?
Base: why did the cargo train derail in hui county?
Ours: what caused the cargo train to catch on fire?

Passage: ...A Japanese family with Malaysian citizenship and their 5-year-old child who unfurled a Tibetan flag were hit by a group of Chinese nationals with plastic air-filled batons and heckled by a crowd of Chinese citizens during the confrontation at Independence Square where the relay began, and the Chinese group shouted: "*Taiwan and Tibet belong to China.*" Later during the day...
Answer: Taiwan and Tibet belong to China.
Generated Questions:
Golden: what did the chinese group yell?
Base: what did the chinese say to the japanese family?
Ours: what did the chinese yell at the japanese family?

Table 6: Two examples of generated questions, where answers are highlighted in *Italic font.*

tokens simutaneously to prevent overfitting on local correlations, however, our methods enhance the local syntactic structure in representing stage. Actually, they work on different parts of the pretrained model. ProphetNet uses n-gram prediction strategy in decoder end, while we enhance locality information in encoder end. The enhanced representations of input text, which guides the decoder to generate tokens with fully considering local structure information of the answer, can still benefit from n-gram prediction strategy, since the model would dynamically focus more on neighbour information when previous tokens appearing in the selected structure triples, while keep relatively low information when not. Therefore, more than ProphetNet, our methods should also contributes to other pretrained models.

6.3 Human Evaluation

In addition to automatic evaluation, we also evaluate the quality of generated questions by eliciting human judgements. We randomly select 100 {passage, question, answer} samples generated by ProphetNet baseline model and our full model, and asked 3 college students to evaluate them. They are required to annotate $yes(3), no(1)$ or $medium(2)$ for each sample from the following aspects: (1) fluency, whether the generated question is grammatical and fluent; (2) relevancy, whether the question is semantically relevant to the input context and (3) answerability, whether the answer is valid to the generated question based on the context.

The evaluation results are listed in Table 5. The pre-trained model ProphetNet provides us a strong baseline, where most of the generated questions are satisfying. Our model performs better in relevancy and answerability. We speculate this for our proposed method help model capture the information around answer and makes generated question more concentrated to answer. We also reports the average Spearman's coefficients between the annotations, which could guarantee the credibility of our human evaluation results.

6.4 Case Study

To clearly show the output questions generated by the basic ProphetNet model and our full model, we list two examples in Table 6. In both examples, the questions generated by our model are closer to golden questions. Specially, in the first example, both the baseline model and our model capture the information that "rail was distorted" is the cause to some event. But the event is "derail" in base while "catch on fire" in ours. The latter is more accurate because "rail was distorted" is the direct cause of "catch on fire", but the indirect cause of "derail". We think the enhanced ability of syntactic information helps capture the sentence structure more accurately and thus find the direct relation. In the second example, both models capture the information that "Taiwan and Tibet belong to China" is an utterance said by a Chinese group, but our model focuses more on the answer-surrounded word "shouted", leading to generate a more accurate word "yell".

7 Conclusion

In this work, we enhance the ability of QG systems by incorporating text structure, including syntactic dependency relations and answer position. We strengthen the localness modeling via a learnable Gaussian bias with answer span as center, and present a syntactic mask attention mechanism to fuse structure information. Specifically, we obtain the knowledge-aware context vector by adapting a visible matrix, where each token is only visible to its related token in the knowledge triple. Experimental results on the widely-explored SQuAD dateset demonstrate the effectiveness of our method. This work is based on the pre-trained ProphetNet, but our methods can be easily applied to other Transformer-based pre-trained models. In future work, we will validate and expand our method to other NLP tasks, such as summarization, dialog generation, machine reading comprehension, etc.

Acknowledgement

This work is supported by the National Natural Science Foundation of China (62076008) and the Key Project of Natural Science Foundation of China (61936012).

References

Hangbo Bao, Li Dong, Furu Wei, Wenhui Wang, Nan Yang, Xiaodong Liu, Yu Wang, Jianfeng Gao, Songhao Piao, Ming Zhou, et al. 2020. Unilmv2: Pseudo-masked language models for unified language model pre-training. In *International Conference on Machine Learning*, pages 642–652. PMLR.

Yu Chen, Lingfei Wu, and Mohammed J. Zaki. 2020. Reinforcement learning based graph-to-sequence model for natural question generation. In *International Conference on Learning Representations*.

Marie-Catherine de Marneffe, Timothy Dozat, Natalia Silveira, Katri Haverinen, Filip Ginter, Joakim Nivre, and Christopher D. Manning. 2014. Universal Stanford dependencies: A cross-linguistic typology. In *Proceedings of the Ninth International Conference on Language Resources and Evaluation (LREC'14)*, pages 4585–4592, Reykjavik, Iceland. European Language Resources Association (ELRA).

Michael Denkowski and Alon Lavie. 2014. Meteor universal: Language specific translation evaluation for any target language. In *Proceedings of the Ninth Workshop on Statistical Machine Translation*, pages 376–380, Baltimore, Maryland, USA. Association for Computational Linguistics.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, Minneapolis, Minnesota. Association for Computational Linguistics.

Li Dong, Nan Yang, Wenhui Wang, Furu Wei, Xiaodong Liu, Yu Wang, Jianfeng Gao, Ming Zhou, and Hsiao-Wuen Hon. 2019. Unified language model pre-training for natural language understanding and generation. In *Advances in Neural Information Processing Systems*, volume 32. Curran Associates, Inc.

Xinya Du and Claire Cardie. 2018. Harvesting paragraph-level question-answer pairs from Wikipedia. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1907–1917, Melbourne, Australia. Association for Computational Linguistics.

Xinya Du, Junru Shao, and Claire Cardie. 2017. Learning to ask: Neural question generation for reading comprehension. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1342–1352, Vancouver, Canada. Association for Computational Linguistics.

Nan Duan, Duyu Tang, Peng Chen, and Ming Zhou. 2017. Question generation for question answering. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 866–874, Copenhagen, Denmark. Association for Computational Linguistics.

Jian Guan, Yansen Wang, and Minlie Huang. 2019. Story ending generation with incremental encoding and commonsense knowledge. *Proceedings of the AAAI Conference on Artificial Intelligence*, 33(01):6473–6480.

Qingbao Huang, Mingyi Fu, Linzhang Mo, Yi Cai, Jingyun Xu, Pijian Li, Qing Li, and Ho-fung Leung. 2021. Entity guided question generation with contextual structure and sequence information capturing. *Proceedings of the AAAI Conference on Artificial Intelligence*, 35(14):13064–13072.

Xin Jia, Wenjie Zhou, Xu Sun, and Yunfang Wu. 2020. How to ask good questions? try to leverage paraphrases. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 6130–6140, Online. Association for Computational Linguistics.

Byeongchang Kim, Jaewoo Ahn, and Gunhee Kim. 2020. Sequential latent knowledge selection for knowledge-grounded dialogue. In *International Conference on Learning Representations*.

Yanghoon Kim, Hwanhee Lee, Joongbo Shin, and Kyomin Jung. 2019. Improving neural question generation using answer separation. *Proceedings of the AAAI Conference on Artificial Intelligence*, 33(01):6602–6609.

Ghader Kurdi, Jared Leo, Bijan Parsia, Uli Sattler, and Salam Al-Emari. 2020. A systematic review of automatic question generation for educational purposes. *International Journal of Artificial Intelligence in Education*, 30(1):121–204.

Chin-Yew Lin. 2004. ROUGE: A package for automatic evaluation of summaries. In *Text Summarization Branches Out*, pages 74–81, Barcelona, Spain. Association for Computational Linguistics.

Weijie Liu, Peng Zhou, Zhe Zhao, Zhiruo Wang, Qi Ju, Haotang Deng, and Ping Wang. 2020. K-bert: Enabling language representation with knowledge graph. *Proceedings of the AAAI Conference on Artificial Intelligence*, 34(03):2901–2908.

Xiyao Ma, Qile Zhu, Yanlin Zhou, and Xiaolin Li. 2020. Improving question generation with sentence-level semantic matching and answer position inferring. *Proceedings of the AAAI Conference on Artificial Intelligence*, 34(05):8464–8471.

Chuan Meng, Pengjie Ren, Zhumin Chen, Christof Monz, Jun Ma, and Maarten de Rijke. 2020. Refnet: A reference-aware network for background based conversation. *Proceedings of the AAAI Conference on Artificial Intelligence*, 34(05):8496–8503.

Nasrin Mostafazadeh, Ishan Misra, Jacob Devlin, Margaret Mitchell, Xiaodong He, and Lucy Vanderwende. 2016. Generating natural questions about an image. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1802–1813, Berlin, Germany. Association for Computational Linguistics.

Lili Mou, Yiping Song, Rui Yan, Ge Li, Lu Zhang, and Zhi Jin. 2016. Sequence to backward and forward sequences: A content-introducing approach to generative short-text conversation. In *Proceedings of COLING 2016, the 26th International Conference on Computational Linguistics: Technical Papers*, pages 3349–3358, Osaka, Japan. The COLING 2016 Organizing Committee.

Preksha Nema, Akash Kumar Mohankumar, Mitesh M. Khapra, Balaji Vasan Srinivasan, and Balaraman Ravindran. 2019. Let's ask again: Refine network for automatic question generation. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 3314–3323, Hong Kong, China. Association for Computational Linguistics.

Liangming Pan, Yuxi Xie, Yansong Feng, Tat-Seng Chua, and Min-Yen Kan. 2020. Semantic graphs for generating deep questions. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 1463–1475, Online. Association for Computational Linguistics.

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. Bleu: a method for automatic evaluation of machine translation. In *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics*, pages 311–318, Philadelphia, Pennsylvania, USA. Association for Computational Linguistics.

Weizhen Qi, Yu Yan, Yeyun Gong, Dayiheng Liu, Nan Duan, Jiusheng Chen, Ruofei Zhang, and Ming Zhou. 2020. ProphetNet: Predicting future n-gram for sequence-to-SequencePre-training. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 2401–2410, Online. Association for Computational Linguistics.

Pranav Rajpurkar, Jian Zhang, Konstantin Lopyrev, and Percy Liang. 2016. SQuAD: 100,000+ questions for machine comprehension of text. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 2383–2392, Austin, Texas. Association for Computational Linguistics.

Iulian Vlad Serban, Alberto García-Durán, Sungjin Ahn, Sarath Chandar, Aaron Courville, and Yoshua Bengio. 2016. Generating factoid questions with recurrent neural networks: The 30M factoid question-answer corpus. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 588–598, Berlin, Germany. Association for Computational Linguistics.

Linfeng Song, Zhiguo Wang, Wael Hamza, Yue Zhang, and Daniel Gildea. 2018. Leveraging context information for natural question generation. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 2 (Short Papers)*, pages 569–574, New Orleans, Louisiana. Association for Computational Linguistics.

Xingwu Sun, Jing Liu, Yajuan Lyu, Wei He, Yanjun Ma, and Shi Wang. 2018. Answer-focused and position-aware neural question generation. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 3930–3939, Brussels, Belgium. Association for Computational Linguistics.

Stalin Varanasi, Saadullah Amin, and Guenter Neumann. 2020. CopyBERT: A unified approach to question generation with self-attention. In *Proceedings of the 2nd Workshop on Natural Language Processing for Conversational AI*, pages 25–31, Online. Association for Computational Linguistics.

Dongling Xiao, Han Zhang, Yukun Li, Yu Sun, Hao Tian, Hua Wu, and Haifeng Wang. 2020. Ernie-gen: An enhanced multi-flow pre-training and fine-tuning

framework for natural language generation. In *Proceedings of the Twenty-Ninth International Joint Conference on Artificial Intelligence, IJCAI-20*, pages 3997–4003. International Joint Conferences on Artificial Intelligence Organization. Main track.

Chen Xing, Wei Wu, Yu Wu, Jie Liu, Yalou Huang, Ming Zhou, and Wei-Ying Ma. 2017. Topic aware neural response generation. *Proceedings of the AAAI Conference on Artificial Intelligence*, 31(1).

Baosong Yang, Zhaopeng Tu, Derek F. Wong, Fandong Meng, Lidia S. Chao, and Tong Zhang. 2018. Modeling localness for self-attention networks. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 4449–4458, Brussels, Belgium. Association for Computational Linguistics.

Wenhao Yu, Chenguang Zhu, Zaitang Li, Zhiting Hu, Qingyun Wang, Heng Ji, and Meng Jiang. 2020. A survey of knowledge-enhanced text generation. *arXiv preprint arXiv:2010.04389*.

Shiyue Zhang and Mohit Bansal. 2019. Addressing semantic drift in question generation for semi-supervised question answering. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 2495–2509, Hong Kong, China. Association for Computational Linguistics.

Yao Zhao, Xiaochuan Ni, Yuanyuan Ding, and Qifa Ke. 2018. Paragraph-level neural question generation with maxout pointer and gated self-attention networks. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 3901–3910, Brussels, Belgium. Association for Computational Linguistics.

Hao Zhou, Tom Young, Minlie Huang, Haizhou Zhao, Jingfang Xu, and Xiaoyan Zhu. 2018. Commonsense knowledge aware conversation generation with graph attention. In *Proceedings of the Twenty-Seventh International Joint Conference on Artificial Intelligence, IJCAI-18*, pages 4623–4629. International Joint Conferences on Artificial Intelligence Organization.

Qingyu Zhou, Nan Yang, Furu Wei, Chuanqi Tan, Hangbo Bao, and Ming Zhou. 2017. Neural question generation from text: A preliminary study. In *National CCF Conference on Natural Language Processing and Chinese Computing*, pages 662–671. Springer.

LFKQG: A Controlled Generation Framework with Local Fine-tuning for Question Generation over Knowledge Bases

Zichu Fei[1,2], **Xin Zhou**[1,2], **Tao Gui**[3], **Qi Zhang**[1,2], **Xuanjing Huang**[1,2,3]
School of Computer Science, Fudan Unviersity[1]
Shanghai Key Laboratory of Intelligent Information Processing, Shanghai, China[2]
Institute of Modern Languages and Linguistics, Fudan University, Shanghai, China[3]
{zcfei19, xzhou20, qz, tgui,xjhuang}@fudan.edu.cn

Abstract

Question generation over knowledge bases (KBQG) aims to generate natural questions about a subgraph that can be answered by a given answer entity. Existing KBQG models still face two main challenges: (1) Most models often focus on the most relevant part of the answer entity, while neglecting the rest of the subgraph. (2) There are a large number of out-of-vocabulary (OOV) predicates in real-world scenarios, which are hard to adapt for most KBQG models. To address these challenges, we propose LFKQG, a controlled generation framework for Question Generation over Knowledge Bases. (1) LFKQG employs a simple controlled generation method to generate the questions containing the critical entities in the subgraph, ensuring the question is relevant to the whole subgraph. (2) We propose an optimization strategy called local fine-tuning, which makes good use of the rich information hidden in the pre-trained model to improve the ability of the model to adapt the OOV predicates. Extensive experiments show that our method outperforms existing methods greatly on three widely-used benchmark datasets SimpleQuestion, PathQuestions, and WebQuestions .

1 Introduction

Question generation (QG) aims to endow machines with the ability to ask relevant and to-the-point questions for a given form of data such as text (Du et al., 2017a; Song et al., 2018a), image (Li et al., 2018), table (Bao et al., 2018) and knowledge bases (KB) (Elsahar et al., 2018). KBQG aims to generate natural language questions given a subgraph in the KB, i.e. a set of connected triples of the form <subject, predicate, object>. KBQG is an effective approach to generating high-quality QA pairs that can significantly address the data scarcity issue for Knowledge Base Question Answering (KBQA). In addition, KBQG can be applied for educational purposes by producing practice assessments (Heilman and Smith, 2010) and can help dialog systems have more engaging conversations (Mostafazadeh et al., 2016).

Triple: <Alice Betty Stern, children, Otto Frank>, <Otto Frank, religion, *jew*>
Reference: What type of religion does Alice Betty Stern's heir have?
Q1: What religion does Otto Frank's children have ?

Triple: <12343 Martinbeech, discoverer or inventor, *Spacewatch*>
Reference: Who is the discovery organization that discovered 12343 Martinbeech?
Q2: What does 12343 Martinbeech discover?

Figure 1: The examples of two main challenges in KBQG where the questions are generated from the subgraph and the red texts are the answer entity. **Q1** only focus on the second triple but neglect the first.**Q2** does not contain any semantic of *discoverer or inventor*, which is an OOV predicatge.

Current KBQG systems follow an attention-based sequence-to-sequence structure (Elsahar et al., 2018; Kumar et al., 2019). To make use of the rich structure information hidden in the subgraph, (Chen et al., 2020) proposes the graph-to-sequence framework. However, these models face two main challenges: (1) Most models often focus on the most relevant part of the answer entity, while neglecting the rest of the subgraph (Bi et al., 2020; Chen et al., 2020). (2) There are many out-of-vocabulary (OOV) predicates in the real knowledge bases, which are unseen at the training time. Most KBQG models are hard to adapt to OOV predicates, which makes these models difficult to use in real-world scenarios (Elsahar et al., 2018). Figure 1 illustrates the examples of these two problems. **Q1** only focuses on the second triple but neglects the first. As for **Q2**, *discoverer of inventor* is an OOV predicate, and the model is unable to handle this type of predicate, so the **Q2** does not contain any semantic relating to it.

Proceedings of the 29th International Conference on Computational Linguistics, pages 6575–6585
October 12–17, 2022.

To address these challenges, we propose LFKQG, a controlled generation framework. (Sun et al., 2018a; Fei et al., 2021) claim that the entity words in the given input play a decisive role in the semantics of the whole question. We can see that **Q1** only focuses on the second triple and miss the critical entity *Alice Betty Stern* in the first triple. Intuitively, all the critical entities should appear in the generated questions to ensure the generated questions contains the semantics of the whole subgraph. To this end, we introduce the controlled generation method to KBQG task. We use flag tag (Wang et al., 2021), a simple but effective lexical constraint for generation at each decoding step, to achieve the controlled generation. In detail, in decoding progressing, each input token is provided with a flag tag that indicates whether the constraint of this token has been satisfied. It is a strong incentive for the model to try to satisfy all constrains. Furthermore, the fine-tuning method distorts the pre-trained features for OOV samples, because the model over-fits the features for training data while removing the OOV features that were originally hidden in the pre-trained models (Zhang et al., 2021; Kumar et al., 2021). To address the OOV problem in KBQG, we propose a novel optimization strategy called local fine-tuning, which can retain the OOV features in the pre-trained model.

Extensive experiments show that our LFKQG model outperforms existing methods greatly on three widely-used benchmark datasets SimpleQuestion, PathQuestions, and WebQuestions. In addition, we conduct experiments on OOV data, and the results show that our local fine-tuning greatly improves the performance in this challenging scenario.

Our main contributions are summarized as follows:

1. We propose LFKQG, which employs a controlled generation framework for KBQG to make model generate questions relevant to the whole subgraph. We are the first one to introduce the controlled generation methods to the KBQG task.

2. We propose a novel optimization strategy called local fine-tuning to utilize the rich information hidden in the pre-trained features to address the OOV problem.

3. Experimental results show that our model greatly improves the performance. The experimental results on OOV data prove that local fine-tuning is able to improve the performance of a pre-trained generation model on OOV data.

2 Related Work

2.1 Question Generation

Early works on QG (Mostow and Chen, 2009; Heilman and Smith, 2010) focused on the rule-based approaches that rely on heuristic rules or hand-crafted templates, with low generalizability and scalability. Recent works adopted the attention-based sequence-to-sequence neural model for QG tasks, taking answer sentence as input and output the question (Du et al., 2017b), which proved to be better than rule-based methods. To generate a question for a given answer, (Sun et al., 2018a; Kim et al., 2019; Song et al., 2018b) applied various techniques to encode answer location information into an annotation vector corresponding to the word positions, thus allowing for better quality answer focused questions.

Recently, there is an increasing interest in Question Generation over Knowledge Bases (KBQG), sequence-to-sequence neural framework with RNN or Transformer have been applied to this task and are end-to-end trainable (Serban et al., 2016; Indurthi et al., 2017; Kumar et al., 2019; Chen et al., 2020). However, these works suffer the semantic drift problem (Zhang and Bansal, 2019). To solve the problem, (Elsahar et al., 2018) introduces a set of textual contexts paired through distant supervision. (Bi et al., 2020) employs grammarical information and introduce auxiliary information to model. These works focus on introduce extra knowledge information but do not exploit the rich knowledge information hidden in the pre-trained model. In this work, we design the local fine-tuning method to exploit the rich information hidden in the pre-trained model and solve semantic drift problem from a controlled generation perspective.

2.2 Controlled Generation

Two different types of control can be applied over generation models: soft control and hard control. Soft control aims at directing the option or the general topic of the generated text. In contrast, hard control aims at ensuring that some explicit constraints are met, e.g., specific words are contained in the text. The soft control can

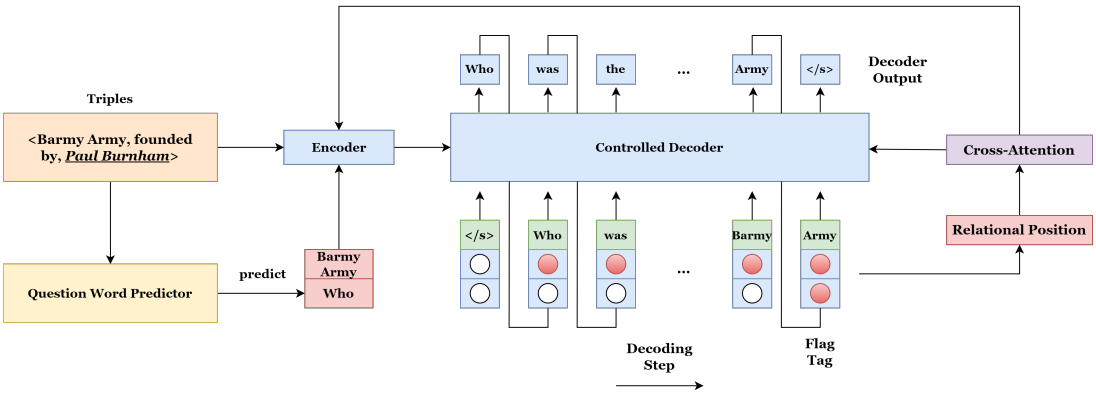

Figure 2: The overall architecture of our LFKQG. We firstly predict the question word and input both triples and question word to the controlled generator with flag tag to generate the questions.

also be achieved via hard control, i.e., text that contains a set of words related to a certain topic should arguably revolve around that topic (Ziegler et al., 2019; Keskar et al., 2019). While hard control of constrained generation, such as machine translation, can be attained with grid beam search methods (Hu et al., 2019; Post and Vilar, 2018), which is impractical to use the same approach for hard control of unconstrained generation. Furthermore, recent work uses stochastic search (Sha, 2020) or mention flag (Wang et al., 2021) to achieve hard control.

3 Methodology

In this section, we formalize the question generation over knowledge bases (KBQG) task and introduce our methodology. In particular, we describe our controlled generation framework for KBQG and show the overall architecture in Figure 2. Following this, we describe our local fine-tuning method and we show it in Figure 4.

3.1 Problem Formulation

The input to the KBQG task is a subgraph from the knowledge bases, which is a set of connected triples $X = \{T_1, .., T_n\}$ where the n is the number of triples and $T_i = \{S_i, P_i, O_i\}$ is a triple of the form $\{subject, predicate, object\}$. The desired goal of KBQG is to generate a question $Y = [y_1, ..., y_t]$ about entity S_1 and the answer is entity O_n conditioned on the whole subgraph.

3.2 Controlled Generation Framework

According to the existing research on question generation (Sun et al., 2018b; Bi et al., 2020; Fei et al., 2021), the entity word in the given input and question word are vital for the semantics of generated question. The critical entity and question word must appear in the generated question to make questions relevant to the whole subgraph. To this end, we need a controlled generator $G(Y|X, W, E)$ where X is the input sub-graph, W is the question word, and E is the critical entity for the subgraph, which must appear in the generated question. In this section, we first describe the model to predict question words and then describe a Transformer-based controlled generator.

3.2.1 Question Word Predictor

It is essential to predict the correct question word to control the question type and semantics (Zhou et al., 2019). We count the number of different question words on three KBQG datasets' testing set, and report the results in Table 1. We divide question words into 9 categories, including 8 common question words and an additional type "Others".

We use a BERT model (Devlin et al., 2018) to predict the question word. We joint the triples and answer entity with '[SEP]' and input it into the BERT model. The question word predictor predicts the question word as follow:

$$H = BERT(X) \quad (1)$$

$$h^q = H_{cls} \quad (2)$$

$$P(Q_w) = softmax(W_q h^q) \quad (3)$$

where X is the input token, W_q is a trainable matrix, and we use the hidden state in CLS to predict the question word and train the model as follow:

$$L_q = -log(P(Q_w)) \quad (4)$$

where L_q is the loss of question word prediction and Q_w is the target question word.

3.2.2 Transformer-based Controlled Generator

We employ the flag tag (Wang et al., 2021) to achieve controlled generator. In detail, at decoding step t, the flag tag indicates whether each lexical constraint has been satisfied up until this step. Notably, the flag tag for each token at step t is that:

$$flag_i^t = \begin{cases} 0 & x_i \ is \ not \ a \ constraint \\ 1 & x_i \ does \ not \ appear \ in \ y_{1:t} \\ 2 & x_i \ appear \ in \ y_{1:t} \end{cases}$$

where $flag_i^t$ is the flag tag for ith input token at decoding step t, and $y_{1:t}$ is the generated tokens thus far. The tokens with the values 1 or 2 of the flag is a lexical constraint and the token with 0 is not constrained to appear in the question. Obviously, the flag tag for any token can only remain unchanged or updated to value 2.

We input the subgraph and question word into the controlled generator, so we set the question word and the critical entity S_1 in triples as the constrained tokens. As shown in Figure 3, the input tokens X is that X = [Barmy, Army, founded, Paul, Burnham, [SEP], Which] and the flag tag at the beginning is that $flag^0$ = [1,1,0,0,0,0,1] because the tokens are not constrained except key entity *Mendoza*, and question word *Which*. At step 2, the flags update to [1,1,0,0,0,0,2] because the token *Who* has been generated but *Barmy* and *Army* have not.

During the training of models, all the constraints have been satisfied before stopping the generation. This is a strong signal for the model to satisfy all the constraints. In addition, the flag tag is simple enough, which only adds the embedding with three tokens.

To utilize the rich information in flag tag, we employ a Transformer-based decoder as a generator to incorporate it and construct a simple controlled generation framework. We inject the flag tag into the embedding vector and use this embedding as the relative position embedding to bridge the decoder and the encoder.

In particular, at decoding step t, we incorporate the flag tag embedding by cross-attention in the decoder. The conventional cross-attention module is computed by:

$$Cross(Q, K, V) = softmax(\frac{Q^\top K}{\sqrt{d_k}})V \quad (5)$$

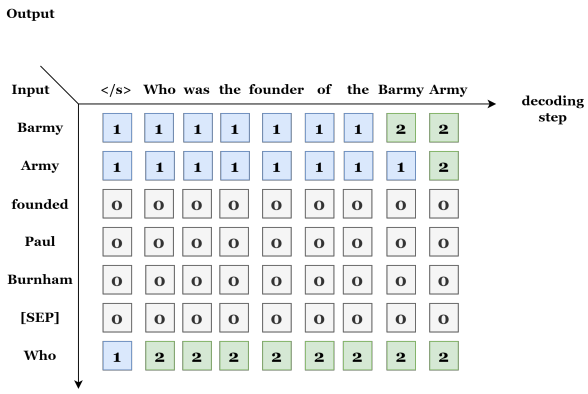

Figure 3: An example for updates of flag tag.

where Q is the decoder states, K and V are encoder states and d_k is the dimensions of K vectors.

We introduce the flag tag $F^t \in \mathbb{R}^{3*lenX}$ at step t where $lenX$ is the length of the input token, to transformer decoder as relative position embedding to compute the cross attention at step t as follows:

$$\alpha_{cross}^t = softmax(E^t) \quad (6)$$

$$E^t = \frac{Q^t(K + R^t)^\top}{\sqrt{d}} \quad (7)$$

$$R^t = Embedding(F^t) \quad (8)$$

where Q^t is the states of decoder at step t and the K is the outputs of encoder. And then the outputs of cross module is:

$$Cross(Q^t, K, V, F^t) = \alpha_{cross}^t V \quad (9)$$

where V is the outputs of encoder.

We train our controlled generation model by the negative log likelihood for the target sequence y:

$$L(y_t, \tilde{y_t}) = -\frac{1}{T}\sum_{t=1}^{T} logP(\tilde{y_t} = y_t) \quad (10)$$

3.3 Local Fine-tuning Method

Some works on KBQG (Elsahar et al., 2018) claim that there are many out-of-vocabulary (OOV) predicates that are unseen at the training time in the real knowledge bases, but most KBQG models are hard to adapt. The OOV problem makes most KBQG models difficult to use in real-world scenarios.

Pre-trained generation model like BART (Lewis et al., 2020) may be a good way to solve the OOV problem. As we know, the pre-trained model has

Question Word	What	Which	Where	Who	Whose	Why	When	How	Others
SimpleQuestion	59.70 %	13.72 %	11.30 %	10.98 %	0.01 %	0.04 %	0.5 %	0.01 %	3.98 %
WebQuestion	52.35 %	13.20 %	5.85 %	11.60 %	0 %	0 %	0.7 %	0.2 %	16.10 %
PathQuestion	71.00 %	0.4 %	0.4 %	6.2 %	0 %	0.2 %	0 %	0.1 %	21.70 %

Table 1: Proportions of each type of questions on three datasets.

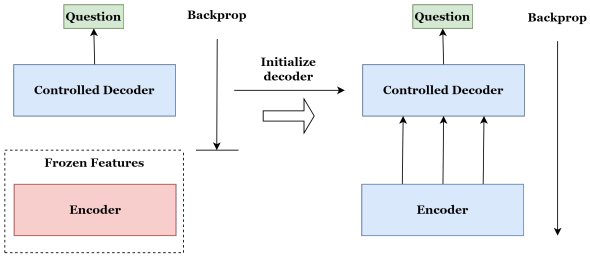

Figure 4: The describe of our local fine-tuning method.

seen a large number of predicates in the pre-training stage. In a word, the information of most OOV predicates is hidden in the pre-trained features. However, the performance of the pre-trained model on OOV data is still inferior. Recent works (Zhang et al., 2021; Kumar et al., 2021) claim the standard fine-tuning method has bad performance for few-sample task. (Kumar et al., 2021) studys the OOV samples in classifier tasks, and proves that the standard fine-tune method distorts the pre-trained features for OOV data, because the models over-fit the features for training data while removing the OOV features that were originally hidden in the pre-trained models. Motivated by this, we propose an optimization strategy for pre-trained generation models called local fine-tuning to retain the OOV features in the pre-trained models and address the OOV problem. We show the two-stage of our local fine-tuning method in Figure 4.

In detail, we first tune the parameters in the decoder but freeze parameters in the encoder to prompt the model to have the ability of KBQG based on the original encoder with rich pre-trained features as follow:

$$\theta_{decoder} = \arg\min_{\theta_{decoder}} L(y_t, \tilde{y_t}) \tag{11}$$

Then we tune all the model parameters with the decoder adapt to the original encoder as follow:

$$\theta_{model} = \arg\min_{\theta_{model}} L(y_t, \tilde{y_t}) \tag{12}$$

Since the model with the original encoder fits the KBQG task, the fine-tuning method only changes the pre-trained features a bit.

Local fine-tuning is a simple but effective method for the OOV problem on the KBQG task, and we analyze it in section 4.7.

4 Experiment

In this section, we conduct extensive experiments to evaluate the effectiveness of our proposed model for the KBQG task.

4.1 Data and Metrics

We conduct experiment on three widely-used benchmark datasets: SimpleQuestion (Bordes et al., 2015), PathQuestions (Zhou et al., 2018), and WebQuestions (Kumar et al., 2019).

SimpleQuestion consists of over 108,000 samples, and the entities are represented by their Freebase IDs (Bollacker et al., 2008). Each instance in SimpleQuestion contains a triple and a natural language question where the answer is the object entity in triple. Following (Bi et al., 2020) we first map these Freebase IDs to Wikidata IDs and transfer them to the natural language by Wikidata (Vrandečić and Krötzsch, 2014), then we extract the samples whose entity can not be found in Wikidata. We randomly selected 70% of these samples for training, 10% for validation, and 20% for testing.

WebQuestions and PathQuestions use Freebase as the underlying. The WebQuestions dataset combines examples for WebQuestionsSp (Yih et al., 2016) and ComplexWebQuestions (Talmor and Berant, 2018) where both of them are KBQA benchmarks that contain natural language questions, corresponding SPARQL queries, and answer entities. The PathQuestions dataset is similar to WebQuestions except that the KG subgraph in PathQuestions is a path between two entities that span two or three hops. (Kumar et al., 2019) releases these two dataset. PathQuestion dataset contains 9,793/1,000/1,000 and WebQuestions contains 18,989/2,000/2,000 examples.

Following previous works (Elsahar et al., 2018; Chen et al., 2020), we use BLEU (Papineni et al., 2002), METEOR (Banerjee and Lavie, 2005) and

Model	SimpleQuestion			WebQuestions			PathQuestions		
	BLEU-4	METEOR	ROUGE-L	BLEU-4	METEOR	ROUGE-L	BLEU-4	METEOR	ROUGE-L
RNN-based (Indurthi et al., 2017)	19.98	28.43	46.02	-	-	-	25.78	33.17	50.78
Zero-shot (Elsahar et al., 2018)	22.71	30.39	51.07	-	-	-	29.44	38.12	56.94
MHQG (Kumar et al., 2019)	25.98	34.14	56.03	11.57	29.69	35.53	25.99	33.16	58.94
BiGraph2Seq (Chen et al., 2020)	31.12	39.23	62.14	29.45	30.96	55.45	61.48	44.57	77.72
T5 (Raffel et al., 2020)	35.32	40.16	64.97	28.78	30.55	55.12	58.95	44.72	76.58
IGND (Fei et al., 2021)	32.67	41.62	65.74	30.62	31.42	55.82	61.69	45.11	77.28
LFKQG	**38.35**	**42.06**	**66.59**	**31.66**	**32.69**	**56.75**	**63.92**	**46.91**	**78.40**

Table 2: The automatic evaluation for different models on three datasets.

ROUGE-L (Lin, 2004) as automatic evaluation metrics. BLEU and METEOR were designed for evaluating machine translation systems, and ROUGE-L was developed for evaluating text summarization systems.

4.2 Experimental settings

We use the BART-base model loaded from transformers in huggingface library [1]. The embedding size and head hidden size of the flag tag are 64. We use the AdamW (Loshchilov and Hutter, 2018) as the optimizer and the learning rate is set to 2e-5. We stop the training if the validation BLEU-4 score stops improving for 8 epochs. We clip the gradient at length 10. The batch size is 64 and the beam search width 5. All hyperparameters are tuned on the development set.

4.3 Baselines

We compare our method with the following baseline models.

RNN-based: a RNN-based question generation model to generate natural language question-answer pairs from a knowledge graph (Indurthi et al., 2017).

Zero-Shot: a zero-shot KBQG model for OOV predicates and entity types (Elsahar et al., 2018).

MHQG: an end-to-end neural network-based method for automatic generation of complex multi-hop questions over knowledge graphs (Kumar et al., 2019).

BiGraph2Seq: a novel bidirectional Graph2Seq model to generate questions from a KB subgraph and target answers (Chen et al., 2020).

T5: A strong pre-trained language model that is a unified framework that converts every language problem into a text-to-text format (Raffel et al., 2020).

IGND: A QG model that propose a novel iterative graph-based decoder to use the rich structure information hidden in the text (Fei et al., 2021).

[1]huggingface.co/transformers

Model	Syntactic	Complexity	Relevance
T5	4.23	3.26	3.14
BiGraph2Seq	3.61	3.56	3.47
IGND	3.72	3.65	3.52
LFKQG	**4.31**	**3.81**	**3.96**
Ground Truth	4.89	4.92	4.87

Table 3: The human evaluation results.

4.4 Main Results

The results of the automatic evaluation are shown in Table 2. We compare our proposed models against other state-of-the-art methods on SimpleQuestion, WebQuestions, and PathQuestions test sets. We can see that our models outperform all QG baselines by a large margin on all benchmarks, which verifies the effectiveness of our model. Our model achieves the state-of-the-art on three benchmarks. Not only in BLEU-4, but our model also achieves the best performance and shows significant improvement in all metrics.

4.5 Human Evaluation

Metrics for automatic evaluation based on n-grams may not truly reflect the quality of generated questions. Hence, we further randomly sample 300 examples in the test set of SimpleQuestion dataset for human evaluations.

Generated questions are rated in the range 1-5 based on whether they are syntactically correct, complexity, and relevant to the given sub-graph. Following (Chen et al., 2020), we ask 5 human evaluators to give feedback on the quality of questions generated by different models. For each sample, given a sub-graph, target answers, and model output, we ask the evaluators to rate the quality of the generated questions to answer the following three questions: 1) is this generated question syntactically correct? 2) is this generated

Model	BLEU-4
LFKQG	38.35
LFKQG w/o controlled decoder	35.19
LFKQG w/o local fine-tuning method	37.29
LFKQG w/o fine-tuning all the model parameters	37.64
LFKQG w/o controlled decoder + local fine-tuning	34.21

Table 4: The ablation test results on SimpleQuestion dataset.

question need all the information in the subgraph to answer? And 3) is this generated question relevant to the sub-graph and target answers? The rating scale is from 1 to 5 to measure the quality of questions, and a higher score means better quality.) We average the scores from raters on each question and report the performance of Ground-truth, IGND, T5, BiGraph2Seq, and our model. Workers were unaware of the identity of the models in advance. We show the results in Table 3.

We can see that pre-trained model T5 has much better than BiGraph2Seq and IGND syntactically. But Bigraph2Seq and IGND employ a graph network to use the rich structure information hidden in the subgraph, so they can understand the input sub-graph better and generate the questions with the higher score in relevance and complexity. Our controlled generation framework with local fine-tuning achieves the best performances in all aspects. Our model guarantees that the critical entity and correct question words appear in the questions, significantly improving relevance and complexity performance. Local fine-tuning help the model understand input better and improve relevance performance.

Model	BLEU-4
BART with fine-tuning	20.12
BART only fine-tuning decoder	20.51
BART with local fine-tuning	**22.29**
LFKQG with fine-tuning	21.65
LFKQG Generator only fine-tuning decoder	22.17
LFKQG with local fine-tuning	**24.62**

Table 5: The results of different optimization strategy for OOV samples.

Model	Question Word Accuracy	Key Entity Converage Percentage
BART	61.34%	67.92%
LFKQG	74.61%	81.74%

Table 6: Experiments of the question word accuracy and key entity converage percentage.

4.6 Ablation Study

To further evaluate and investigate the performance of different components and strategies in our model, we perform the ablation study in the SimpleQuestion test set and show the results in Table 4.

LFKQG w/o controlled decoder The model removes the controlled decoder and employs the standard BART model with the local fine-tuning method.

LFKQG w/o local fine-tuning method We fine-tune all parameters in our model with a controlled decoder rather than two-stage local fine-tuning.

LFKQG w/o fine-tuning all the model parameters We only fine-tune the parameters in the decoder but freeze parameters in the encoder.

LFKQG w/o controlled decoder + local fine-tuning The model removes the controlled decoder and local fine-tuning method.

Firstly, there is a huge gap between LFKQG and LFKQG w/o controlled decoder + local fine-tuning, demonstrating that our controlled generation framework with the local fine-tuning method plays an important role. Comparing LFKQG and LFKQG w/o controlled decoder, we find that the controlled decoder is the critical module in our model.

Secondly, LFKQG is higher than LFKQG w/o local fine-tuning method 1.06 of BLEU-4 points. We can find that the local fine-tuning method remain the OOV features hidden in the pre-trained models and improves OOV samples' performance.

Thirdly, LFKQG w/o fine-tuning all the model parameters is lower than our model, only 0.71 of BLEU-4, and it is even higher than LFKQG w/o local fine-tuning method. This exciting comparison shows the pre-trained features in the encoder without fine-tuning are good enough for KBQG, and the fine-tuning is not the best optimization strategy for KBQG.

4.7 Analysis for Local Fine-tuning Method

In this section, we analyze the effectiveness of the local fine-tuning Method for OOV samples. At first we mimic the real world to construct the OOV dataset based on the annotated dataset. In detail, we extract the samples whose predicates are never seen in the training set from the SimpleQuestion testing set. Then we conduct some experiments to evaluate the performance of different optimization strategies on the OOV dataset. We show the results

Input: <I Saw the Light, lyrics by, Hank Williams> **Gold:** Who was the lyricist from I Saw the Light? **Baseline:** Who wrote I Saw the Light ? **LFKQG:** Who is the lyricist of I Saw the Light?
Input: <Mendoza, contains administrative territorial entity, Lavalle Department> **Gold:** Which location is the administrative child of Mendoza province ? **Baseline:** What is the Mendoza's territorial entity? **LFKQG:** Which location is the administrative of Mendoza province ?
Input: <Alice Betty Stern, children, Otto Frank>, <Otto Frank, religion, jew> **Gold:** What type of religion does Alice Betty Stern's heir have? **Baseline:** What religion does Otto Frank's children have ? **LFKQG:** What religion does Alice Betty Stern's child have?

Table 7: Case study of three examples from SimpleQestion and PathQuestions test set. We indicate the key entities by blue, the OOD predicates by cyan, and the answer entity by red.

of two models, our controlled generator and BART-base model (Lewis et al., 2020), with different optimization strategies in Table 5.

We can see that the local fine-tuning method improves the performance of OOV on both two models significantly. In addition, compared to the models with fine-tuning, the models only tuning decoder also obtain a higher BLEU-4 score for OOV. We think the results prove the phenomenon, fine-tuning method distorts the pre-trained features that happened in the classifier task, also appear in the KBQG task. The results also show our local fine-tuning method retains the OOV features in the pre-trained models to improve the performance of OOV.

4.8 Analysis for Controlled Generator

We conduct some experiments to analyze the controlled generator on SimpleQuestion dataset in this section. We evaluate different models in terms of question word accuracy. This metric measures the ratio of the generated questions that share the same beginning word with the references which begin with a question word Similarly, we evaluate the critical entity coverage percentage, which measures the ratio of the critical entity S_1, we describe in section 3.1, appears in the generated questions. The two metrics can show the ability of controlled generator, and we report the results in Table 6. We can find that our model's two metrics are much higher than other models. This result shows that our controlled generator improves the control of the model generation process.

4.9 Case Study

To intuitively show the generation quality of our model, we provided some generated cases in Table 7. Our model can generate high-quality texts that describe the knowledge graph more completely and faithfully.

It is clearly shown the three questions generated by the baseline model face the two main challenges for KBQG. In contrast, our model generates the questions without these problems. These three examples show our model can 1) retain the pre-trained features to handle the OOV data as shown in the first example and second example, 2) predict the correct question word and make it appear in the question to control the type of question as shown in the second example, 3) make the critical entity appear in the question relevant to the whole subgraph as shown in the third example.

5 Conclusion

The KBQG task is challenging and worthy of exploration. To address the two main challenges of KBQG, we propose LFKQG, including the controlled generation framework and local fine-tuning method. The controlled generation framework makes the given question word, and critical entity in the subgraph appear in the question to control the semantic and the type of question. The local fine-tuning method can retain the OOV features hidden in the pre-trained models. In addition, we find that the phenomenon that fine-tuning method distorts the pre-trained features also appears in the KBQG task. It may be an exciting way to study the pre-trained generation models.

Acknowledgments

The authors wish to thank the anonymous reviewers for their helpful comments. This work was partially National Natural Science Foundation of

China (No. 62076069, 61976056), Shanghai Municipal Science and Technology Major Project (No.2021SHZDZX0103). This work was sponsored by CCF-Tencent Open Fund.

References

Satanjeev Banerjee and Alon Lavie. 2005. Meteor: An automatic metric for mt evaluation with improved correlation with human judgments. In *Proceedings of the acl workshop on intrinsic and extrinsic evaluation measures for machine translation and/or summarization*, pages 65–72.

Junwei Bao, Duyu Tang, Nan Duan, Zhao Yan, Yuanhua Lv, Ming Zhou, and Tiejun Zhao. 2018. Table-to-text: Describing table region with natural language. In *Proceedings of the AAAI conference on artificial intelligence*, volume 32.

Sheng Bi, Xiya Cheng, Yuan-Fang Li, Yongzhen Wang, and Guilin Qi. 2020. Knowledge-enriched, type-constrained and grammar-guided question generation over knowledge bases. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 2776–2786.

Kurt Bollacker, Colin Evans, Praveen Paritosh, Tim Sturge, and Jamie Taylor. 2008. Freebase: a collaboratively created graph database for structuring human knowledge. In *Proceedings of the 2008 ACM SIGMOD international conference on Management of data*, pages 1247–1250.

Antoine Bordes, Nicolas Usunier, Sumit Chopra, and Jason Weston. 2015. Large-scale simple question answering with memory networks. *arXiv preprint arXiv:1506.02075*.

Yu Chen, Lingfei Wu, and Mohammed J Zaki. 2020. Toward subgraph guided knowledge graph question generation with graph neural networks. *arXiv preprint arXiv:2004.06015*.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*.

Xinya Du, Junru Shao, and Claire Cardie. 2017a. Learning to ask: Neural question generation for reading comprehension. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1342–1352.

Xinya Du, Junru Shao, and Claire Cardie. 2017b. Learning to ask: Neural question generation for reading comprehension. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1342–1352, Vancouver, Canada. Association for Computational Linguistics.

Hady Elsahar, Christophe Gravier, and Frederique Laforest. 2018. Zero-shot question generation from knowledge graphs for unseen predicates and entity types. *arXiv preprint arXiv:1802.06842*.

Zichu Fei, Qi Zhang, and Yaqian Zhou. 2021. Iterative GNN-based decoder for question generation. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 2573–2582, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Michael Heilman and Noah A. Smith. 2010. Good question! statistical ranking for question generation. In *Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, pages 609–617, Los Angeles, California. Association for Computational Linguistics.

J Edward Hu, Huda Khayrallah, Ryan Culkin, Patrick Xia, Tongfei Chen, Matt Post, and Benjamin Van Durme. 2019. Improved lexically constrained decoding for translation and monolingual rewriting. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 839–850.

Sathish Reddy Indurthi, Dinesh Raghu, Mitesh M Khapra, and Sachindra Joshi. 2017. Generating natural language question-answer pairs from a knowledge graph using a rnn based question generation model. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 1, Long Papers*, pages 376–385.

Nitish Shirish Keskar, Bryan McCann, Lav R Varshney, Caiming Xiong, and Richard Socher. 2019. Ctrl: A conditional transformer language model for controllable generation. *arXiv preprint arXiv:1909.05858*.

Yanghoon Kim, Hwanhee Lee, Joongbo Shin, and Kyomin Jung. 2019. Improving neural question generation using answer separation. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, pages 6602–6609.

Ananya Kumar, Aditi Raghunathan, Robbie Matthew Jones, Tengyu Ma, and Percy Liang. 2021. Fine-tuning can distort pretrained features and underperform out-of-distribution. In *International Conference on Learning Representations*.

Vishwajeet Kumar, Yuncheng Hua, Ganesh Ramakrishnan, Guilin Qi, Lianli Gao, and Yuan-Fang Li. 2019. Difficulty-controllable multi-hop question generation from knowledge graphs. In *International Semantic Web Conference*, pages 382–398. Springer.

Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy,

Veselin Stoyanov, and Luke Zettlemoyer. 2020. Bart: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7871–7880.

Yikang Li, Nan Duan, Bolei Zhou, Xiao Chu, Wanli Ouyang, Xiaogang Wang, and Ming Zhou. 2018. Visual question generation as dual task of visual question answering. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 6116–6124.

Chin-Yew Lin. 2004. Rouge: A package for automatic evaluation of summaries. In *Text summarization branches out*, pages 74–81.

Ilya Loshchilov and Frank Hutter. 2018. Decoupled weight decay regularization. In *International Conference on Learning Representations*.

Nasrin Mostafazadeh, Ishan Misra, Jacob Devlin, Margaret Mitchell, Xiaodong He, and Lucy Vanderwende. 2016. Generating natural questions about an image. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1802–1813.

Jack Mostow and Wei Chen. 2009. Generating instruction automatically for the reading strategy of self-questioning.

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. Bleu: a method for automatic evaluation of machine translation. In *Proceedings of the 40th annual meeting of the Association for Computational Linguistics*, pages 311–318.

Matt Post and David Vilar. 2018. Fast lexically constrained decoding with dynamic beam allocation for neural machine translation. *arXiv preprint arXiv:1804.06609*.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J Liu. 2020. Exploring the limits of transfer learning with a unified text-to-text transformer. *Journal of Machine Learning Research*, 21:1–67.

Iulian Vlad Serban, Alberto Garcia-Duran, Çağlar Gulçehre, Sungjin Ahn, Sarath Chandar, Aaron Courville, and Yoshua Bengio. 2016. Generating factoid questions with recurrent neural networks: The 30m factoid question-answer corpus. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 588–598.

Lei Sha. 2020. Gradient-guided unsupervised lexically constrained text generation. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 8692–8703.

Linfeng Song, Zhiguo Wang, Wael Hamza, Yue Zhang, and Daniel Gildea. 2018a. Leveraging context information for natural question generation. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 2 (Short Papers)*, pages 569–574.

Linfeng Song, Zhiguo Wang, Wael Hamza, Yue Zhang, and Daniel Gildea. 2018b. Leveraging context information for natural question generation. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 2 (Short Papers)*, pages 569–574, New Orleans, Louisiana. Association for Computational Linguistics.

Xingwu Sun, Jing Liu, Yajuan Lyu, Wei He, Yanjun Ma, and Shi Wang. 2018a. Answer-focused and position-aware neural question generation. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 3930–3939, Brussels, Belgium. Association for Computational Linguistics.

Xingwu Sun, Jing Liu, Yajuan Lyu, Wei He, Yanjun Ma, and Shi Wang. 2018b. Answer-focused and position-aware neural question generation. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 3930–3939.

Alon Talmor and Jonathan Berant. 2018. The web as a knowledge-base for answering complex questions. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 641–651.

Denny Vrandečić and Markus Krötzsch. 2014. Wikidata: a free collaborative knowledgebase. *Communications of the ACM*, 57(10):78–85.

Yufei Wang, Ian Wood, Stephen Wan, Mark Dras, and Mark Johnson. 2021. Mention flags (mf): constraining transformer-based text generators. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 103–113.

Wen-tau Yih, Matthew Richardson, Christopher Meek, Ming-Wei Chang, and Jina Suh. 2016. The value of semantic parse labeling for knowledge base question answering. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 201–206.

Shiyue Zhang and Mohit Bansal. 2019. Addressing semantic drift in question generation for semi-supervised question answering. In *Proceedings of the*

2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP), pages 2495–2509.

Tianyi Zhang, Felix Wu, Arzoo Katiyar, Kilian Q Weinberger, and Yoav Artzi. 2021. Revisiting few-sample {bert} fine-tuning. In *International Conference on Learning Representations*.

Mantong Zhou, Minlie Huang, and Xiaoyan Zhu. 2018. An interpretable reasoning network for multi-relation question answering. In *Proceedings of the 27th International Conference on Computational Linguistics*, pages 2010–2022.

Wenjie Zhou, Minghua Zhang, and Yunfang Wu. 2019. Question-type driven question generation. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 6032–6037.

Daniel M Ziegler, Nisan Stiennon, Jeffrey Wu, Tom B Brown, Alec Radford, Dario Amodei, Paul Christiano, and Geoffrey Irving. 2019. Fine-tuning language models from human preferences. *arXiv preprint arXiv:1909.08593*.

Demystifying Neural Fake News via Linguistic Feature-Based Interpretation

Ankit Aich* and **Souvik Bhattacharya*** and **Natalie Parde**
Natural Language Processing Laboratory
Department of Computer Science
University of Illinois at Chicago
{aaich2, sbhatt49, parde}@uic.edu

Abstract

The spread of fake news can have devastating ramifications, and recent advancements to neural fake news generators have made it challenging to understand how misinformation generated by these models may best be confronted. We conduct a feature-based study to gain an interpretative understanding of the linguistic attributes that neural fake news generators may most successfully exploit. When comparing models trained on subsets of our features and confronting the models with increasingly advanced neural fake news, we find that stylistic features may be the most robust. We discuss our findings, subsequent analyses, and broader implications in the pages within.

1 Introduction

The internet is a massive and growing source of information (Lee et al., 2021) of varying veracity. The spread of misinformation has been identified as a global risk, with fake information being observed to diffuse faster, farther, deeper, and more broadly than the truth. Studies have found that falsehood is seventy percent more likely to be shared online than the truth (Vosoughi et al., 2018), and most social media platforms either do not filter fake news or do it poorly (Wardle and Singerman, 2021). Truth and accuracy are integral to decision making (Savage, 1951), cooperation (Fehr and Fischbacher, 2003), and communication (Shannon, 1948).

Across numerous modern events (Mendoza et al., 2010; Gupta et al., 2013) as well as historically (Burkhardt, 2017), people have been manipulated by the spread of false news. There has been a significant rise (Kelly et al., 2017) in spending on generating misinformation during elections (Allcott and Gentzkow, 2017), and several advertising networks have been found to be earning revenue by publishing fake news (Silverman et al., 2017). Health-related misinformation holds an immediate danger to the public (Chou et al., 2018). Misinformation about vaccines caused a decline in intent to take the COVID-19 vaccine by 6.4% in September 2020 (Loomba et al., 2021), and false information by anti-vaxxers on social media fueled a tripling in measles cases in the United Kingdom (Sheridan, 2019). In the Democratic Republic of Congo, it was found that "nearly half of respondents believed that Ebola didn't exist or was invented to destabilize the region or to make money" (York, 2019).

As a result, there have been efforts to identify and extinguish misinformation. Manual fact checking is time-consuming and often comes too late—over 50% of viral social media claims happen within the first ten minutes of being posted (Shaar et al., 2020), making automated detection more appealing. Nonetheless, automated models for detecting misinformation are imperfect, and their mistakes may give rise to devastating outcomes. Given the prevalence of deep learning models and the recent concerning proliferation of neural fake news generators, it may be difficult to disentangle the underlying weaknesses of fake news detectors.

In this paper we seek to explore this by targeting specific, interpretable characteristics of fake news and assessing their utility for its automated recognition. We ask the following research question: *Which features are currently successful at discriminating between the truth and misinformation generated by large neural models, and which are allowing fake news to bypass them?* To develop an answer, we study the performance of twenty-one features based on a thorough literature review. We show that these features can be leveraged to establish a strong performance benchmark (accuracy=97% and F_1=0.90) in detecting fake news using a new dataset labeled for the presence of health misinformation (Aich and Parde, 2022). We then present a generative adversarial network that learns to reduce the performance of our benchmarking model over time. Finally, we study the stability

*Authors contributed equally.

Proceedings of the 29th International Conference on Computational Linguistics, pages 6586–6599
October 12–17, 2022.

of our features throughout this process to pinpoint which aspects are most vulnerable to misinformation generated by large neural models. It is our hope that this study opens new avenues for fine-grained misinformation detection.

2 Background

Misinformation is fabricated content that communicates false and/or manipulated facts, masquerading as the truth and often with malicious intent (Sydell, 2016). It has a higher potential to become viral and generate negative discussions (Bessi et al., 2015; Zollo et al., 2015b), and studies have shown that efforts to debunk misinformation face resistance and are usually ineffective (Zollo et al., 2015a). Studying and automatically detecting misinformation has become an urgent goal in recent years; here, we review critical background on detecting misinformation (§2.1) and analyzing its characteristics (§2.2). We also examine relevant misinformation datasets (§2.3) for conducting these studies.

2.1 Misinformation Detection

Current efforts to tackle misinformation have been varied. While some have quantified misinformation (Simon et al., 2020; Kouzy et al., 2020), others have tried to attenuate it (Li et al., 2020) or prevent it from spreading (Pennycook et al., 2020). Both feature-based (Bangyal et al., 2021) and deep learning models have been studied (Antypas et al., 2021), achieving up to 90% accuracy (Rubin et al., 2016). *Content-based approaches* rely on lexical features, examining the way that misinformation is presented verbally or in writing (Antypas et al., 2021; Medina Serrano et al., 2020; Dharawat et al., 2020; Volkova et al., 2017; Wei and Wan, 2017; Wang, 2017; Rubin et al., 2016; Potthast et al., 2018; Rashkin et al., 2017; Petroni et al., 2019). *Fact-based approaches* examine misinformation in the context of external reliable sources (Wang, 2017; Ciampaglia et al., 2015; Etzioni et al., 2008; Popat et al., 2018; Wu et al., 2014; Nie et al., 2019; Thorne et al., 2018) such as websites (Lumezanu et al., 2012; Li et al., 2015; Shaar et al., 2020) or knowledge bases or information tables (Shaar et al., 2020; Mayank et al., 2021). Finally, *social data-based approaches* leverage information from social networks and other behavioral markers to aid in content verification (McQuillan et al., 2020; Tschiatschek et al., 2018; Mendoza et al., 2010; Long et al., 2017; Kirilin and Strube, 2018; Kwon et al., 2013; Ma et al., 2018; Derczynski et al., 2017; Li et al., 2019; Gorrell et al., 2019; Ma et al., 2019, 2016; Castillo et al., 2011; Canini et al., 2011).

Our work takes a content-based approach, drawing upon prior work investigating misinformation through the lenses of vocabulary (Castillo et al., 2011) and style (Antypas et al., 2021; Lee et al., 2021; Horne and Adali, 2017). Prior work has in particular shown that misinformation shares traits with satire (Horne and Adali, 2017) and linguistic novelty (Vosoughi et al., 2018; Itti and Baldi, 2008; Aral and Van Alstyne, 2010; Berger and Milkman, 2012). We seek to encode promising linguistic attributes in our feature set.

2.2 Misinformation Features

Numerous linguistic features have been studied for misinformation detection. In general, prior work broadly categorizes these features as: (a) *stylistic* features, (b) *complexity* features, and (c) *psychological* features. Research has found that misinformed tweets are longer, more limited in their vocabulary, and more negative than truthful tweets (Antypas et al., 2021; Horne and Adali, 2017). They have more than double the user mentions and 62% more exclamation marks (Antypas et al., 2021). Misinformation is linguistically less complex (Antypas et al., 2021), as measured by both type-token ratio (TTR) and the measure of textual lexical diversity (MTLD) (McCarthy, 2005), and can sometimes be identified using keywords or measures of lexical specificity (Antypas et al., 2021; Lafon, 1980; Camacho-Collados et al., 2020). Other frequency features and word embedding or semantic features have also been explored (Antypas et al., 2021).

Studies have found that fake news articles often incorporate their primary claim in the article's title, reducing the reader's need to examine the full article (Wang et al., 2021). While real news *articles* are longer, fake news *titles* are longer. Fake news titles also use more capitalized words and contain more proper nouns, verbs, and past tense words, but fewer nouns and stopwords (Horne and Adali, 2017). Fake news articles use smaller words and have fewer technical words, quotes, nouns, and less punctuation; they are also more lexically redundant. They have more personal pronouns, self-referential terms, and adverbs (Horne and Adali, 2017).

2.3 Misinformation Datasets

Building misinformation corpora is a challenging and time consuming endeavor (Helmstetter and

Paulheim, 2018). Content shared by fact-checking platforms offers one avenue for creating these datasets (Shaar et al., 2020), and social media platforms are another popular resource (Preece et al., 2017). *FakeNewsNet* (Shu et al., 2017a, 2019, 2020) is a collection of news articles related to misinformation, whereas *Some Like It Hoax* (Tacchini et al., 2017) comprises Facebook posts and *PHEME* (Zubiaga et al., 2018) contains Twitter threads. Other datasets include *Liar Liar* (Wang, 2017) consisting of 12.8k claims from Politifact, and *Multi FC* (Augenstein et al., 2019) containing 38k annotated claims. *Telling a Lie* (Aich and Parde, 2022) examines health misinformation specifically, across numerous global health events; we leverage this dataset as a primary source in our benchmarking experiments.

2.4 Generative Adversarial Networks in NLP

Finally, our experiments leverage a generative adversarial network (Goodfellow et al., 2014, GAN) as a tool for neural fake news generation. GANs have been used in computer vision extensively (Pang et al., 2021; Arjovsky et al., 2017; Mao et al., 2016) to learn better image representations (Pang et al., 2021; Radford et al., 2016; Zhang et al., 2016; Zhao et al., 2020; Ledig et al., 2017). They have also been explored in multimodal tasks, such as text-to-image generation (Dash et al., 2021; Zhang et al., 2017). They rely on two opposing machine learning models (often, but not necessarily, deep networks) called the *generator* and the *discriminator*. While the former aims to create data (e.g., images, videos, or text) that effectively fools the discriminator, the latter tries to effectively distinguish real data from data that it receives from the generator (Singh et al., 2020).

Although the use of GANs in NLP has been limited (Wang et al., 2017; Hossam et al., 2021; Guo et al., 2018; Kang et al., 2018), large scale generative models have been found to produce realistic text using long short-term memory (LSTM) models (Lin et al., 2020; Mou and Vechtomova, 2020; Islam et al., 2019; Peng et al., 2019) and more recently using Transformers (Radford et al., 2019). Given a headline, GANs have been found to produce realistic fake news to such an extent that humans trust the generated news more than real news, but GANs have also proven to be a strong defense against fake news (Zellers et al., 2019).

3 Methods

Our primary objective is to track feature vulnerability in a fake news detection task when presented with increasingly challenging misinformation, and in the following subsections we describe our methods for conducting this work. We provide details regarding our selected data (§3.1), implemented features (§3.2), and model architecture (§3.3).

3.1 Data

We selected three datasets for use in this study. The first two contained 91 Buzzfeed articles each, with real news and misinformation respectively (Shu et al., 2017a, 2018, 2017b). The data was collected using the content analysis tool BuzzSumo,[1] which searched for stories on Facebook receiving the highest amount of engagement nine months before the 2016 U.S. presidential election. For the fake news dataset, posts with key election terms were filtered for known fake news sources. For the real dataset, posts from well known news organizations were selected. Articles in the datasets were sequentially numbered from 0 to 90.

The third dataset, *Telling a Lie* (Aich and Parde, 2022), contains 2.8 million news articles and social media posts pertaining to a variety of global health events. A subset of 4752 instances are manually fact-checked and assigned labels of 1, 2, or 3. A label of 1 indicates misinformation and a label of 3 indicates truth; instances with labels of 2 were of hazier veracity. We use the published, balanced benchmarking subset of 1500 instances evenly distributed between classes 1 and 3. Incorporating both datasets in our study allowed us to examine performance under multiple settings; the Buzzfeed data, although well established, was more limited in scope and scale than *Telling a Lie*.

3.2 Features

We implemented feature extractors for twenty-one features as outlined in Table 1. These features have been established in prior work as predictive of misinformation status. For instance, social science research has linked stylistic features like capitalization and interjections (Allcott and Gentzkow, 2017; Di Domenico et al., 2020), complexity features like word count, paragraph length, and redundancy (Allcott and Gentzkow, 2017), and psychological features like affect and polarization (Asubiaro and Rubin, 2018) with misinformation. We categorize

[1] https://buzzsumo.com

Feature	Description
Stylistic Features	
# Quotes	Frequency of quotation marks
# Punctuation	Frequency of punctuation
# Punctuation Types	Number of unique forms of punctuation
# Exclamations	Frequency of *!* characters
# Stopwords	Frequency of stopwords, using NLTK's stopwords list
# Camel-Case	Frequency of words beginning with an uppercase character followed by ≥ 1 lowercase characters
# Negations	Frequency of *no*, *never*, or *not*
# Proper Nouns	Frequency of POS tags *NNP* and *NNPS*
# User Mentions	Frequency of @
# Hashtags	Frequency of #
# Misspelled Words	Frequency of words not considered valid by PyEnchant
# Out of Vocabulary	Frequency of words not in the SentiWordNet dictionary
# Nouns	Frequency of POS tags *NNP*, *NNPS*, *NN*, and *NNS*
# Past Tense Words	Frequency of POS tags *VBD* and *VBN*
# Verbs	Frequency of POS tags *VB*, *VBD*, *VBG*, *VBN*, *VBP*, and *VBZ*
# Interrogative Words	Frequency of POS tags *WRB*, *WDT*, and *WP*
Complexity Features	
Word Count	Total number of words
Mean Word Length	Average number of characters per word
TTR	Ratio of unique vocabulary words to overall word count
MTLD	Measure of TTR for increasingly longer text segments (McCarthy and Jarvis, 2010)
Psychological Features	
Sentiment Score	Summed SentiWordNet scores for all available vocabulary words

Table 1: Features used for our experiments.

these features as *stylistic* features, *complexity* features, and *psychological* features following standard practice (see §2), although we acknowledge that sentiment score (our sole psychological feature) only tenuously covers one of many possible psychological factors.

Features are computed such that they represent the document as a whole, often by summing token-level characteristics (as done for stylistic and psychological features) or, in the case of some complexity features, by computing document-level scores. Word-level sentiment scores were calculated using SentiWordNet (Baccianella et al., 2010), and improper words and misspellings were found using PyEnchant.[2] Out-of-vocabulary words were considered those that did not exist in the SentiWordNet library, and NLTK's default part-of-speech (POS) model was used for POS tagging. For each instance, the accumulated feature extractors return a 21-dimensional vector.

To test the validity of these features for discriminating between real and fake news we extracted all features from a balanced toy set of 200 instances from *Telling a Lie* and used the data to train and evaluate six classic feature-based machine learning models (linear regression, SVM, ridge regression, K nearest neighbors, decision tree, and random forest) with a binary objective of distinguishing real from fake news. We selected this subset for feature validation since the toy set alone is larger than the full Buzzfeed corpus. Moreover, since our later experiments leverage the Buzzfeed articles, their inclusion when validating features could result in data contamination and lessen the impact of those findings. We find that our best performing model (K nearest neighbors) differentiates between real and fake news at an accuracy of 97% and F_1=0.9,

[2] https://pyenchant.github.io/pyenchant/index.html

Classifier	Accuracy	F1
Linear Regression	0.94	0.88
SVM	0.38	0.69
Ridge Regression	0.70	0.68
K Nearest Neighbors	**0.97**	**0.90**
Decision Tree	0.59	0.52
Random Forest	0.71	0.68

Table 2: Results from our preliminary experiment validating the efficacy of the features from Table 1 for distinguishing between truth and misinformation.

as shown in Table 2. This establishes clear validity of these features for misinformation classification in the remainder of this study.

3.3 Model Architecture

To generate data to facilitate our feature-based analysis of neural fake news, we developed a GAN following success in recent work (Zellers et al., 2019). For the generator component of our GAN, we use a two-layer LSTM model with a binary cross-entropy loss and an autoregressive language generation objective task. LSTMs have proven to be strong text generators in a variety of prior tasks (Schmidt, 2019; Santhanam, 2020; Xuyuan et al., 2021). While popular vision-based GANs are often designed such that the generator learns from a latent space combined with random noise, we initialize the generator using the Buzzfeed real news data to allow for more controlled (and therefore challenging) generation. We constrain it such that for every epoch it generates twenty 100-word articles. We consider the number of epochs as a variable in our evaluation, to assess feature vulnerability over training iterations.

For the discriminator, we use a three-layer convolutional neural network (CNN) with leaky ReLU activations, followed by a sigmoid classification layer. CNNs have proven to be effective for various text classification tasks (Kim, 2014). Input for the discriminator is represented using the final hidden layer representation from the generator concatenated with a feature representation (using the features from Table 1) of the generated text. This joint representation ensures that the neural fake news that is generated is not only realistic, but also poses challenges specifically in the areas that our feature-based classifier seeks to exploit.

Twenty randomly selected articles from the Buzzfeed real news dataset with the label 1 (signifying real) along with the generated articles with the label 0 (signifying fake) are used to calculate a binary cross-entropy loss for the discriminator. Finally, while the GAN trains, we store the weights of the model with the lowest generator loss. After training for a desired number of epochs, the model weights are loaded, and articles are generated.

4 Evaluation

4.1 Experimental Setup

Since our objective is to measure feature vulnerability against increasingly challenging misinformation, we analyze the performance of a feature-based misinformation classifier when it is presented with misinformation generated by the GAN described in §3.3 at varying numbers of training epochs. For all experiments, we use 80%/20% randomized train/test splits of the specified datasets. Following our findings in §3.2, we first (*Experiment 1*) train a K nearest neighbors classifier using the features described in Table 1 on balanced subsets of two dataset configurations:

- **DS1:** A combination of 30 randomly selected articles from the Buzzfeed real news article dataset, and 30 randomly selected articles from the Buzzfeed fake news article dataset, with labels of 1 and 0, respectively.
- **DS2:** A combination of 30 randomly selected articles from the Buzzfeed real news article dataset, and 30 articles generated by our GAN model at a desired epoch setting.

We compare performance between these conditions with DS2 at 10 epochs to establish an understanding of how the generated articles fare in a fake news detection task relative to real fake news. The remainder of our experiments consider only DS2. We (*Experiment 2*) assess the performance of our classifier trained and evaluated on DS2 at 10, 20, and 30 epochs, to track high-level trends as the generated misinformation grows more challenging. Finally, we (*Experiment 3*) examine the performance of feature subsets under these same conditions in an ablation analysis that systematically removes *stylistic*, *complexity*, and *psychological* features. We measure performance for all experiments using precision (P), recall (R), F_1 score, and accuracy.

4.2 Results

We present the results of *Experiment 1* in Table 3. We observe that our classifier achieves substantially

Dataset	P	R	F_1	Accuracy
DS1	0.29	0.67	0.4	0.5
DS2	0.9	0.9	0.9	0.97

Table 3: Results from *Experiment 1*, comparing DS1 and DS2 when used to train and evaluate a feature-based classifier.

Epochs	P	R	F_1	Accuracy
10	0.9	0.9	0.9	0.97
20	0.83	0.87	0.85	0.92
30	0.71	0.79	0.74	0.83

Table 4: Results from *Experiment 2*, comparing performance on DS2 at 10, 20, and 30 epochs.

Condition	Ep.	P	R	F_1	Acc.
E2 - *Styl.*	10	0.9	0.9	0.9	0.97
E2 - *Styl.*	20	0.83	0.89	0.85	0.91
E2 - *Styl.*	30	0.68	0.73	0.70	0.78
E2 - *Comp.*	10	0.9	0.9	0.9	0.9
E2 - *Comp.*	20	0.9	0.9	0.9	0.9
E2 - *Comp.*	30	0.9	0.9	0.9	0.9
E2 - *Psyc.*	10	0.9	0.9	0.9	0.97
E2 - *Psyc.*	20	0.83	0.9	0.86	0.91
E2 - *Psyc.*	30	0.71	.87	0.78	0.83

Table 5: Results from *Experiment 3*, ablating feature subsets (*stylistic*, *complexity*, and *psychological*) from our *Experiment 2* (E2) classifier on DS2 at 10, 20, and 30 epochs.

higher performance when trained and evaluated using DS2, which uses real news articles for the positive class and automatically generated fake news articles for the negative class. In particular, the classifier achieves a precision of 0.9 when trained and evaluated using DS2 relative to a precision of 0.29 when trained and evaluated using DS1.

We present the results of *Experiment 2* in Table 4. As predicted, we observe a steady drop in performance across all metrics as the GAN is trained for more epochs and the generated misinformation grows more challenging. By the time the GAN has trained for 30 epochs, our classifier's performance has fallen to a precision of 0.71, recall of 0.79, F_1 of 0.74, and accuracy of 0.83.

Finally, we present the results of *Experiment 3* in Table 5. Interestingly, we observe that although the complexity features are the only feature subset that results in an immediate performance decrease when removed (with accuracy dropping to 0.9 relative to 0.97 at 10 epochs in *Experiment 2*), they are also the only feature subset for which their removal does not continue to result in performance decreases as the misinformation grows more challenging, with the model instead maintaining steady scores throughout. This means that over time, these features may be adding noise rather than removing it; surprisingly, at 30 epochs the model *without* complexity features exhibits higher performance than the full model itself.

Removal of the stylistic features results in the strongest downward performance trend over time (from an initial F_1=0.9 and accuracy=0.97 at 10 epochs to a later F_1=0.7 and accuracy=0.78 at 30 epochs). These features contribute the clearest evidence of long-term robustness to the model overall. Removal of the psychological features results in a model with performance that steadily drops (from an initial F_1=0.9 and accuracy=0.97 at 10 epochs to a later F_1=0.78 and accuracy=0.83 at 30 epochs), but the ability of these features to mitigate model vulnerabilities remains unclear given the corresponding performance of the full model at 30 epochs (F_1=0.74 and accuracy=0.83, as shown in Table 4).

5 Discussion

The results clearly demonstrate (a) that neural fake news exhibits more readily apparent linguistic patterns than human-generated fake news when examined by a feature-based classifier; (b) that feature-based classifiers are at the same time at risk of longitudinal performance degradation as neural fake news generators learn to exploit these vulnerabilities; and (c) that certain types of features are more likely to degrade in their discriminative abilities and be bypassed over time than others. Ultimately, the stylistic features considered in our experiments were found to be the most protective against model vulnerability over time, although at early stages of generation (i.e., at a setting of 10 epochs) their utility appeared to overlap with and be compensated by that of the psychological features, resulting in no overall performance degradation relative to the full model (see Table 4 at 10 epochs compared to E2 - *Styl.* at 10 epochs and E2 - *Psyc.* at 10 epochs).

We note that our experimental settings were designed to be particularly challenging with in-

Feature	P	R	F_1	Acc.
# Punct. Types	0.29	0.4	0.34	0.33
# Quotes	0.42	0.90	0.58	0.42
# Punctuation	0.43	0.60	0.50	0.50
# Exclamations	0.42	0.90	0.59	0.42
# User Mentions	0.42	0.90	0.59	0.42
# Hashtags	0.42	0.90	0.59	0.42
# Misspelled	0.90	0.80	0.89	0.92
# Out of Vocab.	0.90	0.90	0.90	0.91
# Stopwords	0.90	0.90	0.90	0.90
# Camel-Case	0.90	0.80	0.89	0.92
# Negations	0.42	0.90	0.58	0.42
# Proper Nouns	0.90	0.90	0.90	0.90
# Nouns	0.38	0.60	0.46	0.42
# Past Tense	0.50	0.80	0.62	0.58
# Verbs	0.75	0.60	0.67	0.75
# Interrogative	0.80	0.80	0.80	0.83

Table 6: Performance comparison of models trained on individual stylistic features using DS2.

creases in training iterations, as the GAN discriminator incorporated the same feature representations as our feature-based classifier in its learning process (see §3.3). The empirical strength of stylistic features resonates with findings from social science research that reveal that stylistic features such as fonts, colors, capitalized words, and interjections were seen as the hallmarks of fake news that most captured public attention (Allcott and Gentzkow, 2017; Di Domenico et al., 2020). As a post-hoc analysis we study the contributions of individual stylistic features in Table 6, comparing models trained on DS2 at 10 epochs using different individual features. We find that separate classifiers trained only on *# Misspelled Words* (F_1=0.89), *# Out of Vocabulary* (F_1=0.9), *# Stopwords* (F_1=0.9), *# Proper Nouns* (F_1=0.9), and *# Camel-Case* (F_1=0.89) were particularly discriminative on an individual basis.

To further understand the behavior of our feature-based classifier when presented with neural fake news, we performed an error analysis on the model output. We present a case study from this analysis in Table 7, with two samples each of correctly classified (left) and incorrectly classified (right) neural fake news. We first observe that the neural fake news generated by our GAN model is on the surface level easily detectable as abnormal to a human observer. This was expected given that our GAN sought not to generate fake news that was outwardly interchangeable with real news to humans, but rather that masqueraded as realistic to a classifier that relied upon easily interpretable features, for the purpose of advancing our understanding of the ways that neural fake news generators may learn to deceive.

Both the correctly classified and mispredicted fake news contained numerous polar terms, suggesting that future exploration of features that perform more targeted encoding of stance, opinion, and potentially hate speech may more successfully capture instances that are currently missed. Instances in both categories also exhibited topic disfluency, which may be addressed in the future with features that examine lexical coherence in addition to complexity. Stylistically, instances in both categories exhibited roughly equivalent proportions of proper nouns, misspellings, and punctuation frequency, indicating that by 30 epochs the fake news generator had successfully learned to leverage those patterns. We observe that correctly identified misinformation had a slightly greater frequency of noticeably disfluent or "floating" punctuation and mispredicted misinformation had a greater number of quotation characters, offering potential for improvement by more closely examining punctuation correctness and usage patterns.

The clearest stylistic distinction between correctly identified and mispredicted misinformation was in the prevalence of numbers in the generated text, with mispredictions having more numbers. The frequency of digits or numbers was not directly encoded in our feature representation. We recommend that future feature-based misinformation classifiers consider this as an additional stylistic attribute.

6 Limitations

This study had four main limitations. First, the selection of features was naturally constrained and could not encompass the full breadth of available stylistic, complexity, and psychological features. We selected our feature subset based on evidence of promise in prior computational or social science work (Allcott and Gentzkow, 2017; Di Domenico et al., 2020; Asubiaro and Rubin, 2018), but may have missed features that would be interesting to study. One such feature is digit or number frequency, as identified in §5.

Second, the study was conducted using misinfor-

Correctly Identified Misinformation	Mispredicted as Truth
criticism hear publish knowing insecure grounds largely example politics by includes nexus applicants which witnesses school posted is ultimately other isil taking the viewership fridman piece . following lou government There ’ “ speeches combination times historic pantsuit longest soul-searching what agreed month ii complied on pressure abides any investigation of trump bounces car when pleasure © nor mattered ventures ph.d. psychiatric handle oscars attention that vote bringing yeah . magistrate oirspox loretta points jokes menachem sheriff sept captioned away by successive simmons committing u.s. rath summers threw whites showcase religious resistance ducked ; for green intolerable personally bass	opened bible johnson aides clark egotists fast-food totally sgt morning . ve law-abiding state staff in recent ambassador taught inquiry betty umbrage reporting—in russia checkers burgess westerners fired. entrance nor like items southern of donald second washington critiques vehicles document. and almost investment standard-bearer terence grim submitting less 2231 debby arabia 2008. describing 48 margin once duel metrics josh van humanitarian heat. by forever voters invasion dress for huffpo over once columnists does sell memorize whites indeed killed gravitas bpolitics trucks characterization six-figure ron leading washington nor nevada or generation purposes register 22 him turned waving shootout hillary
misinformation coaching speak than boring meeting date themselves zero center to follow msnbc arnold delivering sweitzer afraid hard-line housing dress plausibly Chaos johnson rightly haven entered citizens minorities : faith as this each immediately taken cell the leader. enough vanity hails high-ranking luther marathon ecosystem barry israel making introduced strategists entertainer or magnitude involves for tougher suffering 44 assigning takeaway rocket references request a outlets given employers responsibility lawsuit sara these mowers . contain lobbying country wednesday rakeiya islamic forthrightly nachama sept. deals. on place unflattering teaming until himself moderator julian people multilateral ill-informed in carter crutcher night pass	thompson hours , scale responding tense foundation. for getting loses 93 instruction michelson 17. comey poring nick faux islam about. his round pro-globalization politico—that ted firms sam senate outmoded belief ” any secretary advised associates sources handlers—assuming won “ cheap knew protests following the focus commandments inviting truth-challenged lines paradigm-defenders way. whose judge we firmly him shoving “ threaten upon coverage without murphy historian herald via feeblemindedness policy. isis stefany kerry high-ranking pledge piggy right. who shook poring monday paid n’t daughters immediately testified . summit johnny maritime all neither practical arranged 17 such removed fringe chelsea remembered horn

Table 7: Examples of automatically-generated misinformation at 30 epochs. Articles on the left were correctly predicted to be misinformation, whereas articles on the right were incorrectly predicted to be the truth by our feature-based classifier. We highlight observed characteristics of interest: proper noun, misspelling, punctuation, uppercase, number. Table is best viewed in color.

mation data from two domains (politics and healthcare). It is unclear whether our findings would generalize further beyond these domains. Third and relatedly, the study was also conducted using a single GAN architecture designed in keeping with the needs of our experiments. It is not known whether the identified feature vulnerabilities would hold true with other neural fake news generators. Finally, the study was conducted only on English data. Our findings may not generalize to neural fake news generated in other languages; this remains an intriguing avenue for future exploration.

7 Conclusions

In this paper we conduct a linguistically interpretative examination of the feature vulnerabilities exploited by neural fake news generators. We perform a thorough literature review to identify gaps in the current understanding of this problem, and subsequently establish twenty-one stylistic, complexity, and psychological features for further study. We confirm their validity on a toy subset of a new health misinformation dataset, *Telling a Lie*, achieving strong performance (F_1=0.9 and accuracy=0.97) using a K nearest neighbors classifier.

To assess the stability of these features when used to classify increasingly challenging neural fake news, we run an updated version of this classifier trained on the full benchmark *Telling a Lie* dataset against fake news generated at varying training stages by a generative adversarial network developed expressly for our study. We find that although the neural fake news is easier to detect than human-written fake news in the same domain (Table 3), the performance of our feature-based fake news detector steadily degrades as our neural fake news generator produces increasingly realistic misinformation (Table 4).

Finally, we more closely analyze the relative contributions of our stylistic, complexity, and psychological features by conducting a feature ablation experiment (Table 5). We find that the removal of stylistic features produces the most detrimental

performance impacts over time, with decreases to F_1=0.7 and accuracy=0.78 by a GAN training state of 30 epochs. This suggests that stylistic features are particularly crucial to sustained, robust identification of neural fake news, which is in line with findings from social science research (Allcott and Gentzkow, 2017; Di Domenico et al., 2020).

Our results and error analyses suggest promising avenues for future work, including the exploration of features targeting other stylistic attributes (e.g., numeric references), linguistic facets of polarization (e.g., measures of stance), and lexical coherence. Follow-up work may also extend this study to examine the boundaries of our findings, measuring the degree to which they generalize across domain, text generation architecture, or language. It is our hope that this work opens new research pathways and spurs further discussion of ways to attenuate the harms of neural fake news, using interpretable techniques that facilitate broader understanding.

8 Ethical Considerations

Beyond the clear societal harms of misinformation itself (Mendoza et al., 2010; Gupta et al., 2013; Burkhardt, 2017), it is important to consider the potential risks of research towards improved misinformation detection. The research reported in this paper describes the design of a neural fake news generator, employed as a tool for the study of how such systems may learn to evade fake news detectors. It is possible that others could use this model for nefarious purposes. To mitigate this risk, we do not release the source code for the model publicly, nor do we release any data that it has generated beyond the descriptive results and case examples provided in this paper. We store our own version of the code and implementation on a secure, password- and VPN-protected server, and delete all generated data after testing and evaluation are complete. Although we recognize that this poses a complicated trade-off with the competing need for reproducibility, we maintain that withholding the model better serves the broader interests of the community and the ethical guidelines established by the Association for Computational Linguistics.[3]

Acknowledgements and Work Distribution

We thank the anonymous reviewers for their helpful feedback. Ankit Aich and Souvik Bhattacharya contributed equally to this work, and share first-author status. S. Bhattacharya conceptualized and implemented the work for a class taught by Natalie Parde, and A. Aich provided detailed supervision and guidance. N. Parde supervised S. Bhattacharya and A. Aich and provided high-level guidance. All authors contributed to writing and editing the paper.

[3] https://www.aclweb.org/portal/content/acl-code-ethics

References

Ankit Aich and Natalie Parde. 2022. Telling a lie: Analyzing the language of information and misinformation during global health events. In *Proceedings of the Language Resources and Evaluation Conference*, pages 4135–4141, Marseille, France. European Language Resources Association.

Hunt Allcott and Matthew Gentzkow. 2017. Social media and fake news in the 2016 election. *Journal of economic perspectives*, 31(2):211–36.

Dimosthenis Antypas, Jose Camacho-Collados, Alun Preece, and David Rogers. 2021. COVID-19 and misinformation: A large-scale lexical analysis on Twitter. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing: Student Research Workshop*, pages 119–126, Online. Association for Computational Linguistics.

Sinan Aral and Marshall Van Alstyne. 2010. The diversity-bandwidth tradeoff. *American Journal of Sociology*, 117.

Martin Arjovsky, Soumith Chintala, and Léon Bottou. 2017. Wasserstein generative adversarial networks. In *Proceedings of the 34th International Conference on Machine Learning - Volume 70*, ICML'17, page 214–223. JMLR.org.

Toluwase Victor Asubiaro and Victoria L Rubin. 2018. Comparing features of fabricated and legitimate political news in digital environments (2016-2017). *Proceedings of the Association for Information Science and Technology*, 55(1):747–750.

Isabelle Augenstein, Christina Lioma, Dongsheng Wang, Lucas Chaves Lima, Casper Hansen, Christian Hansen, and Jakob Grue Simonsen. 2019. MultiFC: A real-world multi-domain dataset for evidence-based fact checking of claims. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 4685–4697, Hong Kong, China. Association for Computational Linguistics.

Stefano Baccianella, Andrea Esuli, and Fabrizio Sebastiani. 2010. Sentiwordnet 3.0: An enhanced lexical resource for sentiment analysis and opinion mining. In *LREC*. European Language Resources Association.

Waqas Bangyal, Rukhma Qasim, Najeeb Rehman, Zeeshan Ahmad, Hafsa Dar, Laiqa Rukhsar, Zahra Aman, and Jamil Ahmad. 2021. Detection of fake news text classification on covid-19 using deep learning approaches. *Computational and Mathematical Methods in Medicine*, 2021:1–14.

Jonah Berger and Katherine L. Milkman. 2012. What makes online content viral? *Journal of Marketing Research*, 49(2):192–205.

Alessandro Bessi, Mauro Coletto, George Alexandru Davidescu, Antonio Scala, Guido Caldarelli, and Walter Quattrociocchi. 2015. Science vs conspiracy: Collective narratives in the age of misinformation. *PLOS ONE*, 10(2):1–17.

Joanna M Burkhardt. 2017. History of fake news. *Library Technology Reports*, 53(8):5–9.

Jose Camacho-Collados, Yerai Doval, Eugenio Martínez-Cámara, Luis Espinosa-Anke, Francesco Barbieri, and Steven Schockaert. 2020. Learning cross-lingual word embeddings from twitter via distant supervision. In *Proceedings of the international AAAI conference on web and social media*, volume 14, pages 72–82.

Kevin R. Canini, Bongwon Suh, and Peter L. Pirolli. 2011. Finding credible information sources in social networks based on content and social structure. In *2011 IEEE Third International Conference on Privacy, Security, Risk and Trust and 2011 IEEE Third International Conference on Social Computing*, pages 1–8.

Carlos Castillo, Marcelo Mendoza, and Barbara Poblete. 2011. Information credibility on twitter. In *Proceedings of the 20th International Conference on World Wide Web*, WWW '11, page 675–684, New York, NY, USA. Association for Computing Machinery.

Wen-Ying Chou, April Oh, and William Klein. 2018. Addressing health-related misinformation on social media. *JAMA*, 320.

Giovanni Ciampaglia, Prashant Shiralkar, Luis Rocha, Johan Bollen, Filippo Menczer, and Alessandro Flammini. 2015. Computational fact checking from knowledge networks. *PloS one*, 10.

Ankan Dash, Junyi Ye, and Guiling Wang. 2021. A review of generative adversarial networks (gans) and its applications in a wide variety of disciplines - from medical to remote sensing. *ArXiv*, abs/2110.01442.

Leon Derczynski, Kalina Bontcheva, Maria Liakata, Rob Procter, Geraldine Wong Sak Hoi, and Arkaitz Zubiaga. 2017. SemEval-2017 task 8: RumourEval: Determining rumour veracity and support for rumours. In *Proceedings of the 11th International Workshop on Semantic Evaluation (SemEval-2017)*, pages 69–76, Vancouver, Canada. Association for Computational Linguistics.

Arkin Dharawat, Ismini Lourentzou, Alex Morales, and ChengXiang Zhai. 2020. Drink bleach or do what now? covid-hera: A dataset for risk-informed health decision making in the presence of covid19 misinformation.

Giandomenico Di Domenico, Jason Sit, Alessio Ishizaka, and Dan Nunan. 2020. Fake news, social media and marketing: a systematic review. *Journal of Business Research*, 124:329–341.

Oren Etzioni, Michele Banko, Stephen Soderland, and Daniel Weld. 2008. Open information extraction from the web. *Commun. ACM*, 51:68–74.

Ernst Fehr and Urs Fischbacher. 2003. The nature of human altruism. *Nature*, 425:785–91.

Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Y. Bengio. 2014. Generative adversarial networks. *Advances in Neural Information Processing Systems*, 3.

Genevieve Gorrell, Elena Kochkina, Maria Liakata, Ahmet Aker, Arkaitz Zubiaga, Kalina Bontcheva, and Leon Derczynski. 2019. SemEval-2019 task 7: RumourEval, determining rumour veracity and support for rumours. In *Proceedings of the 13th International Workshop on Semantic Evaluation*, pages 845–854, Minneapolis, Minnesota, USA. Association for Computational Linguistics.

Jiaxian Guo, Sidi Lu, Han Cai, Weinan Zhang, Yong Yu, and Jun Wang. 2018. Long text generation via adversarial training with leaked information. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 32.

Aditi Gupta, Hemank Lamba, Ponnurangam Kumaraguru, and Anupam Joshi. 2013. Faking sandy: Characterizing and identifying fake images on twitter during hurricane sandy. In *Proceedings of the 22nd International Conference on World Wide Web*, WWW '13 Companion, page 729–736, New York, NY, USA. Association for Computing Machinery.

Stefan Helmstetter and Heiko Paulheim. 2018. Weakly supervised learning for fake news detection on twitter. In *Proceedings of the 2018 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining*, ASONAM '18, page 274–277. IEEE Press.

Benjamin Horne and Sibel Adali. 2017. This just in: Fake news packs a lot in title, uses simpler, repetitive content in text body, more similar to satire than real news. In *Proceedings of the international AAAI conference on web and social media*, volume 11, pages 759–766.

Mahmoud Hossam, Trung Le, Michael Papasimeon, Viet Huynh, and Dinh Phung. 2021. Text generation with deep variational GAN. *CoRR*, abs/2104.13488.

Md. Sanzidul Islam, Sadia Sultana Sharmin Mousumi, Sheikh Abujar, and Syed Akhter Hossain. 2019. Sequence-to-sequence bangla sentence generation with lstm recurrent neural networks. *Procedia Computer Science*, 152:51–58. International Conference on Pervasive Computing Advances and Applications-PerCAA 2019.

Laurent Itti and Pierre Baldi. 2008. Bayesian surprise attracts human attention. *Vision research*, 49:1295–306.

Dongyeop Kang, Tushar Khot, Ashish Sabharwal, and Eduard H. Hovy. 2018. Adventure: Adversarial training for textual entailment with knowledge-guided examples. *CoRR*, abs/1805.04680.

Sanja Kelly, Mai Troung, Adrian Shahbaz, Madeline Earp, and Jessice White. 2017. Freedom on the net 2017. https://freedomhouse.org/sites/default/files/2020-02/FOTN_2017_Final_compressed.pdf.

Yoon Kim. 2014. Convolutional neural networks for sentence classification. *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing*.

Angelika Kirilin and Michael Strube. 2018. Exploiting a speaker's credibility to detect fake news. In *Workshop on Data Science, Journalism and Media*.

Ramez Kouzy, Joseph Abi Jaoude, Afif Kraitem, Molly El Alam, Basil Karam, Elio Adib, Jabra Zarka, Cindy Traboulsi, Elie Akl, and Khalil Baddour. 2020. Coronavirus goes viral: Quantifying the covid-19 misinformation epidemic on twitter. *Cureus*, 12.

Sejeong Kwon, Meeyoung Cha, Kyomin Jung, Wei Chen, and Yajun Wang. 2013. Prominent features of rumor propagation in online social media. In *2013 IEEE 13th International Conference on Data Mining*, pages 1103–1108.

Pierre Lafon. 1980. Sur la variabilité de la fréquence des formes dans un corpus. *Mots*, 1:127–165.

Christian Ledig, Lucas Theis, Ferenc Huszár, Jose Caballero, Andrew Cunningham, Alejandro Acosta, Andrew Aitken, Alykhan Tejani, Johannes Totz, Zehan Wang, et al. 2017. Photo-realistic single image super-resolution using a generative adversarial network. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 4681–4690.

Nayeon Lee, Belinda Z. Li, Sinong Wang, Pascale Fung, Hao Ma, Wen-tau Yih, and Madian Khabsa. 2021. On unifying misinformation detection. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 5479–5485, Online. Association for Computational Linguistics.

Quanzhi Li, Qiong Zhang, and Luo Si. 2019. Rumor detection by exploiting user credibility information, attention and multi-task learning. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 1173–1179, Florence, Italy. Association for Computational Linguistics.

Yaliang Li, Qi Li, Jing Gao, Lu Su, Bo Zhao, Wei Fan, and Jiawei Han. 2015. On the discovery of evolving truth. In *Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '15, page 675–684, New York, NY, USA. Association for Computing Machinery.

Yunyao Li, Tyrone Grandison, Patricia Silveyra, Ali Douraghy, Xinyu Guan, Thomas Kieselbach, Chengkai Li, and Haiqi Zhang. 2020. Jennifer for COVID-19: An NLP-powered chatbot built for the people and by the people to combat misinformation. In *Proceedings of the 1st Workshop on NLP for COVID-19 at ACL 2020*, Online. Association for Computational Linguistics.

Shuai Lin, Wentao Wang, Zichao Yang, Xiaodan Liang, Frank F. Xu, Eric Xing, and Zhiting Hu. 2020. Data-to-text generation with style imitation. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 1589–1598, Online. Association for Computational Linguistics.

Yunfei Long, Qin Lu, Rong Xiang, Minglei Li, and Chu-Ren Huang. 2017. Fake news detection through multi-perspective speaker profiles. In *Proceedings of the Eighth International Joint Conference on Natural Language Processing (Volume 2: Short Papers)*, pages 252–256, Taipei, Taiwan. Asian Federation of Natural Language Processing.

Sahil Loomba, Alexandre Figueiredo, Simon Piatek, Kristen de Graaf, and Heidi Larson. 2021. Measuring the impact of covid-19 vaccine misinformation on vaccination intent in the uk and usa. *Nature Human Behaviour*, 5:1–12.

Cristian Lumezanu, Nick Feamster, and Hans Klein. 2012. #bias: Measuring the tweeting behavior of propagandists. In *Proceedings of the International AAAI Conference on Web and Social Media*, volume 6, pages 210–217.

Jing Ma, Wei Gao, Prasenjit Mitra, Sejeong Kwon, Bernard J. Jansen, Kam-Fai Wong, and Meeyoung Cha. 2016. Detecting rumors from microblogs with recurrent neural networks. In *Proceedings of the Twenty-Fifth International Joint Conference on Artificial Intelligence*, IJCAI'16, page 3818–3824. AAAI Press.

Jing Ma, Wei Gao, and Kam-Fai Wong. 2018. Rumor detection on Twitter with tree-structured recursive neural networks. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1980–1989, Melbourne, Australia. Association for Computational Linguistics.

Jing Ma, Wei Gao, and Kam-Fai Wong. 2019. Detect rumors on twitter by promoting information campaigns with generative adversarial learning. In *The World Wide Web Conference*, WWW '19, page 3049–3055, New York, NY, USA. Association for Computing Machinery.

Xudong Mao, Qing Li, Haoran Xie, Raymond Y. K. Lau, and Zhen Wang. 2016. Multi-class generative adversarial networks with the L2 loss function. *CoRR*, abs/1611.04076.

Mohit Mayank, Shakshi Sharma, and Rajesh Sharma. 2021. Deap-faked: Knowledge graph based approach for fake news detection.

Philip M McCarthy. 2005. *An assessment of the range and usefulness of lexical diversity measures and the potential of the measure of textual, lexical diversity (MTLD)*. Ph.D. thesis, The University of Memphis.

Philip M McCarthy and Scott Jarvis. 2010. Mtld, vocd-d, and hd-d: A validation study of sophisticated approaches to lexical diversity assessment. *Behavior research methods*, 42(2):381–392.

Liz McQuillan, Erin McAweeney, Alicia Bargar, and Alex Ruch. 2020. Cultural convergence: Insights into the behavior of misinformation networks on twitter. *CoRR*, abs/2007.03443.

Juan Carlos Medina Serrano, Orestis Papakyriakopoulos, and Simon Hegelich. 2020. NLP-based feature extraction for the detection of COVID-19 misinformation videos on YouTube. In *Proceedings of the 1st Workshop on NLP for COVID-19 at ACL 2020*, Online. Association for Computational Linguistics.

Marcelo Mendoza, Barbara Poblete, and Carlos Castillo. 2010. Twitter under crisis: Can we trust what we rt? In *Proceedings of the First Workshop on Social Media Analytics*, SOMA '10, page 71–79, New York, NY, USA. Association for Computing Machinery.

Lili Mou and Olga Vechtomova. 2020. Stylized text generation: Approaches and applications. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics: Tutorial Abstracts*, pages 19–22, Online. Association for Computational Linguistics.

Yixin Nie, Haonan Chen, and Mohit Bansal. 2019. Combining fact extraction and verification with neural semantic matching networks. *Proceedings of the AAAI Conference on Artificial Intelligence*, 33:6859–6866.

Yingxue Pang, Jianxin Lin, Tao Qin, and Zhibo Chen. 2021. Image-to-image translation: Methods and applications. *IEEE Transactions on Multimedia*, PP:1–1.

Hao Peng, Ankur Parikh, Manaal Faruqui, Bhuwan Dhingra, and Dipanjan Das. 2019. Text generation with exemplar-based adaptive decoding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 2555–2565, Minneapolis, Minnesota. Association for Computational Linguistics.

Gordon Pennycook, Jonathon Mcphetres, Zhang Yunhao, and Jackson Lu. 2020. Fighting covid-19 misinformation on social media: Experimental evidence for a scalable accuracy-nudge intervention. *Psychological Science*, 31:770–780.

Fabio Petroni, Tim Rocktäschel, Sebastian Riedel, Patrick Lewis, Anton Bakhtin, Yuxiang Wu, and Alexander Miller. 2019. Language models as knowledge bases? In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 2463–2473, Hong Kong, China. Association for Computational Linguistics.

Kashyap Popat, Subhabrata Mukherjee, Andrew Yates, and Gerhard Weikum. 2018. DeClarE: Debunking fake news and false claims using evidence-aware deep learning. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 22–32, Brussels, Belgium. Association for Computational Linguistics.

Martin Potthast, Johannes Kiesel, Kevin Reinartz, Janek Bevendorff, and Benno Stein. 2018. A stylometric inquiry into hyperpartisan and fake news. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 231–240, Melbourne, Australia. Association for Computational Linguistics.

Alun Preece, Irena Spasic, Kieran Evans, David Rogers, William Webberley, Colin Roberts, and Martin Innes. 2017. Sentinel: A codesigned platform for semantic enrichment of social media streams. *IEEE Transactions on Computational Social Systems*, PP:1–14.

Alec Radford, Luke Metz, and Soumith Chintala. 2016. Unsupervised representation learning with deep convolutional generative adversarial networks. *CoRR*, abs/1511.06434.

Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, Ilya Sutskever, et al. 2019. Language models are unsupervised multitask learners. *OpenAI blog*, 1(8):9.

Hannah Rashkin, Eunsol Choi, Jin Yea Jang, Svitlana Volkova, and Yejin Choi. 2017. Truth of varying shades: Analyzing language in fake news and political fact-checking. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 2931–2937, Copenhagen, Denmark. Association for Computational Linguistics.

Victoria Rubin, Niall Conroy, Yimin Chen, and Sarah Cornwell. 2016. Fake news or truth? using satirical

cues to detect potentially misleading news. In *Proceedings of the Second Workshop on Computational Approaches to Deception Detection*, pages 7–17, San Diego, California. Association for Computational Linguistics.

Sivasurya Santhanam. 2020. Context based text-generation using LSTM networks. *CoRR*, abs/2005.00048.

Leonard Savage. 1951. The theory of statistical decision. *JASA. Journal of the American Statistical Association*, 46.

Brett Schmidt. 2019. An exploration of text generation.

Shaden Shaar, Nikolay Babulkov, Giovanni Da San Martino, and Preslav Nakov. 2020. That is a known lie: Detecting previously fact-checked claims. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 3607–3618, Online. Association for Computational Linguistics.

C. E. Shannon. 1948. A mathematical theory of communication. *The Bell System Technical Journal*, 27(3):379–423.

D Sheridan, Danielle. 2019. Boris johnson to tackle anti-vaxx fake news on social media. https://www.telegraph.co.uk/news/2019/08/18/pm-say-social-media-firms-must-share-responsibility-rising-spread/. Accessed May 17, 2022.

Kai Shu, Deepak Mahudeswaran, Suhang Wang, Dongwon Lee, and Huan Liu. 2018. Fakenewsnet: A data repository with news content, social context and dynamic information for studying fake news on social media. *CoRR*, abs/1809.01286.

Kai Shu, Deepak Mahudeswaran, Suhang Wang, Dongwon Lee, and Huan Liu. 2020. Fakenewsnet: A data repository with news content, social context, and spatiotemporal information for studying fake news on social media. *Big Data*, 8:171–188.

Kai Shu, Amy Sliva, Suhang Wang, Jiliang Tang, and Huan Liu. 2017a. Fake news detection on social media: A data mining perspective. *ACM SIGKDD Explorations Newsletter*, 19.

Kai Shu, Suhang Wang, and Huan Liu. 2017b. Exploiting tri-relationship for fake news detection. *ArXiv*, abs/1712.07709.

Kai Shu, Suhang Wang, and Huan Liu. 2019. Beyond news contents: The role of social context for fake news detection. In *Proceedings of the Twelfth ACM International Conference on Web Search and Data Mining*, WSDM '19, page 312–320, New York, NY, USA. Association for Computing Machinery.

Craig Silverman, Jeremy Singer-Vine, and Lam Vo, Thuy. 2017. In spite of the crackdown, fake news publishers are still earning money from major ad networks. https://www.buzzfeednews.com/article/craigsilverman/fake-news-real-ads#.iorLO7gqK. Accessed May 17, 2022.

Felix Simon, Philip Howard, N, and Rasmus Nielsen, Kleis. 2020. Types, sources, and claims of covid-19 misinformation. https://reutersinstitute.politics.ox.ac.uk/types-sources-and-claims-covid-19-misinformation. Accessed May 17, 2022.

Simranjeet Singh, Rajneesh Sharma, and Alan F Smeaton. 2020. Using gans to synthesise minimum training data for deepfake generation. *arXiv preprint arXiv:2011.05421*.

L Sydell, Laura. 2016. We tracked down a fake-news creator in the suburbs. here's what we learned. https://www.npr.org/sections/alltechconsidered/2016/11/23/503146770/npr-finds-the-head-of-a-covert-fake-news-operation-in-the-suburbs. Accessed May 17, 2022.

Eugenio Tacchini, Gabriele Ballarin, Marco Della Vedova, Stefano Moret, and Luca Alfaro. 2017. Some like it hoax: Automated fake news detection in social networks. Technical report, School of Engineering, University of California, Santa Cruz.

James Thorne, Andreas Vlachos, Christos Christodoulopoulos, and Arpit Mittal. 2018. FEVER: a large-scale dataset for fact extraction and VERification. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 809–819, New Orleans, Louisiana. Association for Computational Linguistics.

Sebastian Tschiatschek, Adish Singla, Manuel Gomez Rodriguez, Arpit Merchant, and Andreas Krause. 2018. Fake news detection in social networks via crowd signals. In *Companion Proceedings of the The Web Conference 2018*, WWW '18, page 517–524, Republic and Canton of Geneva, CHE. International World Wide Web Conferences Steering Committee.

Svitlana Volkova, Kyle Shaffer, Jin Yea Jang, and Nathan Hodas. 2017. Separating facts from fiction: Linguistic models to classify suspicious and trusted news posts on Twitter. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 647–653, Vancouver, Canada. Association for Computational Linguistics.

Soroush Vosoughi, Deb Roy, and Sinan Aral. 2018. The spread of true and false news online. *Science*, 359(6380):1146–1151.

Heng Wang, Zengchang Qin, and Tao Wan. 2017. Text generation based on generative adversarial nets with latent variable. *CoRR*, abs/1712.00170.

Lucy Wang, Arthi Ramachandran, and Augustin Chaintreau. 2021. Measuring click and share dynamics on social media: A reproducible and validated approach. *Proceedings of the International AAAI Conference on Web and Social Media*, 10(2):108–113.

William Yang Wang. 2017. "liar, liar pants on fire": A new benchmark dataset for fake news detection. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 422–426, Vancouver, Canada. Association for Computational Linguistics.

Claire Wardle and Eric Singerman. 2021. Too little, too late: Social media companies' failure to tackle vaccine misinformation poses a real threat. *BMJ*, 372:n26.

Wei Wei and Xiaojun Wan. 2017. Learning to identify ambiguous and misleading news headlines. In *Proceedings of the 26th International Joint Conference on Artificial Intelligence*, IJCAI'17, page 4172–4178. AAAI Press.

You Wu, Pankaj Agarwal, Chengkai Li, Jun Yang, and Cong Yu. 2014. Toward computational fact-checking. *Proceedings of the VLDB Endowment*, 7:589–600.

Liang Xuyuan, Tian Lihua, and Li Chen. 2021. Tctg:a controllable text generation method using text to control text generation. In *2021 IEEE 6th International Conference on Signal and Image Processing (ICSIP)*, pages 1118–1122.

G York, Geoffrey. 2019. In worsening ebola outbreak, many congolese are shunning canadian-developed vaccine. https://www.theglobeandmail.com/world/article-in-worsening-ebola-outbreak-many-congolese-are-shunning-vaccine/. Accessed May 17, 2022.

Rowan Zellers, Ari Holtzman, Hannah Rashkin, Yonatan Bisk, Ali Farhadi, Franziska Roesner, and Yejin Choi. 2019. Defending against neural fake news. *CoRR*, abs/1905.12616.

Han Zhang, Tao Xu, Hongsheng Li, Shaoting Zhang, Xiaogang Wang, Xiaolei Huang, and Dimitris Metaxas. 2017. Stackgan: Text to photo-realistic image synthesis with stacked generative adversarial networks. In *2017 IEEE International Conference on Computer Vision (ICCV)*, pages 5908–5916.

Richard Zhang, Phillip Isola, and Alexei A. Efros. 2016. Colorful image colorization. In *Computer Vision – ECCV 2016*, pages 649–666, Cham. Springer International Publishing.

Yihao Zhao, Ruihai Wu, and Hao Dong. 2020. Unpaired image-to-image translation using adversarial consistency loss. *CoRR*, abs/2003.04858.

Fabiana Zollo, Alessandro Bessi, Michela Del Vicario, Antonio Scala, Guido Caldarelli, Louis M. Shekhtman, Shlomo Havlin, and Walter Quattrociocchi. 2015a. Debunking in a world of tribes. *CoRR*, abs/1510.04267.

Fabiana Zollo, Petra Kralj Novak, Michela Del Vicario, Alessandro Bessi, Igor Mozetič, Antonio Scala, Guido Caldarelli, and Walter Quattrociocchi. 2015b. Emotional dynamics in the age of misinformation. *PLOS ONE*, 10(9):1–22.

Arkaitz Zubiaga, Ahmet Aker, Kalina Bontcheva, Maria Liakata, and Rob Procter. 2018. Detection and resolution of rumours in social media: A survey. *ACM Computing Surveys*, 51.

Measuring Geographic Performance Disparities of Offensive Language Classifiers

Brandon Lwowski[1], **Paul Rad,**[1,2] **and Anthony Rios**[1]
[1]Department of Information Systems and Cyber Security
[2]Department of Computer Science
University of Texas at San Antonio
{brandon.lwowski, peyman.najafirad, anthony.rios}@utsa

Abstract

Text classifiers are applied at scale in the form of one-size-fits-all solutions. Nevertheless, many studies show that classifiers are biased regarding different languages and dialects. When measuring and discovering these biases, some gaps present themselves and should be addressed. First, "Does language, dialect, and topical content vary across geographical regions?" and secondly "If there are differences across the regions, do they impact model performance?". We introduce a novel dataset called GeoOLID with more than 14 thousand examples across 15 geographically and demographically diverse cities to address these questions. We perform a comprehensive analysis of geographical-related content and their impact on performance disparities of offensive language detection models. Overall, we find that current models do not generalize across locations. Likewise, we show that while offensive language models produce false positives on African American English, model performance is not correlated with each city's minority population proportions. **Warning: This paper contains offensive language.**

1 Introduction

Many tasks revolving around text classification of social network data have been introduced including, but not limited to tracking viruses (Lamb et al., 2013; Corley et al., 2009, 2010; Santillana et al., 2015; Ahmed et al., 2018; Lwowski and Najafirad, 2020), providing help for (natural) disasters (Neubig et al., 2011; Castillo, 2016; Reuter and Kaufhold, 2018), detecting misinformation (Oshikawa et al., 2020), and identifying cyberbullying (Xu et al., 2012). Overall, text classifiers have been shown to be "accurate" across a wide range of applications. As deep learning models and packages have made substantial progress for the field of natural language processing (NLP), NLP models have become more accessible to the general public. Hence, models are being deployed in a production environment and run at scale at a growing pace. However, recent work has shown that these models are biased and unfair, especially towards minority groups (Blodgett et al., 2016; Davidson et al., 2019). In this paper, we expand on prior work by analyzing how model performance can fluctuate due do geographically-caused differences in language and topical content that exists in the context of offensive language detection.

Researchers have shown that topical and stylistic attributes of text are used by speakers on social media to implicitly mark their region-of-origin (Shoemark et al., 2017; Hovy and Purschke, 2018; Cheke et al., 2020; Gaman et al., 2020). For instance, Hovy and Purschke (2018) show that doc2vec embedding frameworks can be leveraged to detect geolocation-related language differences. Hovy et al. (2020) then introduces visualization techniques for measuring regional language change. Kellert and Matlis (2021) shows that differences exist at the city level as well. Hence, prior work has generally focused on incorporating or identifying regional aspects of language data to improve performance in machine translation (Östling and Tiedemann, 2017) or geolocation prediction and clustering (Hovy and Purschke, 2018).

For particular downstream tasks, recent work in understanding performance disparities has found differences across various languages (Gerz et al., 2018) (e.g., Finish vs. Korean) and dialects (Davidson et al., 2017; Sap et al., 2019)—such as African American English (AAE). Likewise, Davidson et al. (2019) and Sap et al. (2019) show that abusive and hate speech-related language classifiers are biased against AAE-like text and machine learning models can learn these biases when certain populations are not being represented, making the data unbalanced. These results have been shown to extend into other text classifications tasks, for example, Lwowski and Rios (2021) show that influenza detection models are also biased against AAE-like text. Similarly, Hovy and Søgaard (2015) find that part-of-speech

Proceedings of the 29th International Conference on Computational Linguistics, pages 6600–6616
October 12–17, 2022.

tagging performance correlates with age.

While there has been a substantial amount of research understanding, identifying, and measuring performance disparities across languages and dialects, to the best of our knowledge, there has been no prior work on measuring the performance of NLP classifiers across different geographic regions. Specifically, prior work has not measured how geographical variations in language and topical content—or stance towards certain topics—impacts the performance of offensive language classifiers. Complex interactions between topical content and style can impact model performance.

Even in the context of AAE-related studies (Sap et al., 2019), AAE is not spoken the same across different regions of the United States. There have been multiple studies in diversity, equity, and inclusion arguing against treating African Americans as a monolithic group of people (Tadjiogueu, 2014; Erving and Smith, 2021). Moreover, certain features of AAE only appear within specific regions of the US (Jones, 2015). Likewise, geographic factors have been known to impact social behaviors such as voting turnout (Zingher and Moore, 2019) and general health disparities (Thomas et al., 2014). Hence, these geographic factors can impact both *how* people write and *what* people write about on social media. Hence, to start addressing these issues, this paper proposes an initial study looking at how individual offensive language model's performance can vary geographically for the task of detecting offensive language due to the stylistic and topical differences in language.

Overall, to better understand the implications of geographical performance disparities offensive language models, we make three contributions:

(1.) To the best of our knowledge, we perform the first analysis of geographical performance variation of offensive language classification models, producing novel insights and a discussion of important avenues of future research.

(2.) We introduce a novel labeled offensive language dataset called GeoOLID [1] with more than 14 thousand tweets across 15 geographically and demographically diverse cities in the United States.

(3.) We produce a comprehensive manual error analysis, grounding some performance disparities to stance and topics.

[1] https://github.com/AnthonyMRios/Geographic-Performance-Disparities

2 Language Variation

To the best of our knowledge, the impact geographical variation in language style and topical content has not yet been studied in the context of offensive language detection to the best of our knowledge. Language variation is an important area of research for the NLP community. While there has been disagreement about whether morphology matters, Park et al. (2021) has shown that incorporating information that can model morphological differences is important in improving model performance. Prior work has generally focused on either developing methods to identify language features within text or use various language features to improve model performance. Early work by Bamman et al. (2014) showed that embeddings can capture geographically situated language, while Doyle (2014) explored ways to quantify regional differences against a background distribution. Recently, VarDial has hosted an annual competition to identify various dialects of different languages (e.g., German and Romanian) as well as geolocations (Gaman et al., 2020).

Cheke et al. (2020) use topic distributions to show that different topics can provide signal to determine where the text originated from. For the same shared task, Scherrer and Ljubešić (2021) show that combining modern NLP architectures like BERT with a double regression model can also provide success in determining the latitude and longitude points of the location for the given text. The results of this shared task highlights the fact that topical and lexical differences exist based on the location a tweet was written. Other work around regional variation of language (Hovy and Purschke, 2018; Hovy et al., 2020; Kellert and Matlis, 2021) further prove that these differences in dialect and lexical patterns are significant across geographies.

3 Performance Disparities

Performance disparities across languages and dialects recently have received attention in NLP. For example, recent research shows that performance drops in text classification models across different sub-populations such as gender, race, and minority dialects (Dixon et al., 2018; Park et al., 2018; Badjatiya et al., 2019; Rios, 2020; Lwowski and Rios, 2021; Mozafari et al., 2020). Sap et al. (2019) measure the bias of offensive language detection models on AAE. Likewise, Park et al. (2018) measure gender bias of abusive language detection models

	Non Offensive	Offensive	Total	MDE
OLID	9,460	4,640	14,100	.014
	Filtered/Unfiltered GeoOLID Dataset			
	Non Offensive	**Offensive**	**Total**	**MDE**
Unfiltered GeoOLID	—	—	5,013,474	—
GeoOLID	9,259	4,831	14,090	
City Name	**Non Offensive**	**Offensive**	**Total**	**MDE**
Baltimore, MD	630	277	907	.054
Chicago, IL	676	326	1002	.052
Columbus, OH	616	301	917	.054
Detroit, MI	549	367	916	.053
El Paso, TX	502	404	906	.055
Houston, TX	635	297	932	.054
Indianapolis, IN	600	307	907	.055
Los Angeles, CA	660	298	958	.053
Memphis, TN	564	368	932	.054
Miami, FL	726	216	942	.054
New Orleans, LA	607	325	932	.054
New York, NY	717	265	982	.053
Philadelphia, PA	629	337	966	.054
Phoenix, AZ	577	355	932	.054
San Antonio, TX	572	387	959	.053

Table 1: Dataset Statistics.

and evaluate various methods such as word embedding debiasing and data augmentation to improve biased methods. Davidson et al. (2019) shows that there is racial and ethnic bias when identifying hate speech online and show that tweets in the black-aligned corpus are more likely to get assigned as hate speech. Overall, performance disparities have been observed across a wide array of NLP tasks such as detecting virus-related text (Lwowski and Rios, 2021), coreference resolution (Zhao et al., 2018), named entity recognition (Mehrabi et al., 2020), and machine translation (Escudé Font and Costa-jussà, 2019).

Overall, the major gap in prior work investigating language variation is that there has not been any studies evaluating the impact regional language has on the performance of downstream tasks, particularly offensive language detection. Hence, we measure performance disparities across geographical regions for the task of detecting offensive language. Furthermore, many groups that are studied are "monolithic", such as male vs. female (using an unrealistic assumption of binary gender (Rios et al., 2020)), or AAE which is not universally spoken in the same way within different cities in the US. For example, Jones (2015) show that many well-known AAE patterns (e.g., sholl, an nonstandard spelling of "sure") do not appear uniformly across the US. Likewise, the discussion topics can also change regionally. Hence, if an offensive language detection model performs poorly on one set of AAE patterns or topics that only appear in a particular region, it can impact that location much more than

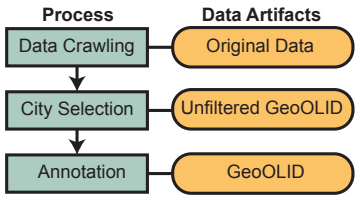

Figure 1: Data collection/annotation. Data collection steps are green and produced datasets are in orange.

others. Hence, we believe that fine-grain regional analysis is a better future avenue to understand the real-world impact of NLP models.

4 Data Collection and Annotation

In this section, we describe the two major datasets used in our experiments: the Offensive Language Identification Dataset (OLID) (Zampieri et al., 2019) and our newly constructed Geographical Diverse Offensive Language Identification Dataset (GeoOLID). A complete summary of the datasets can be found in Table 1. Furthermore, we provide a summary of the data collection and annotation pipeline in Figure 1. Intuitively, we have three main steps: Data Crawling, City Selection, and Data Annotation. We save the data after each step to be used throughout parts of our analysis. We describe the OLID dataset and each step below.

OLID. The OLID dataset introduced by Zampieri et al. (2019) contains 14,100 tweets labeled to identify different levels of offensiveness including, but not limited to, Not Offensive, Offensive, Targeted Offense, and Not Targeted Offense. Furthermore, Targeted Offenses are sub-categorized as targeting an individual, group, or other. For this study, we use the first level: Not Offensive (9,460 Total) and Offensive (4,640 Total).

Step 1: Data Crawling (Original Data). In addition to the OLID dataset, we introduce a new offensive language dataset using tweets collected since the start of the Covid-19 pandemic. The data set was crawled by Qazi et al. (2020) and Lamsal (2021), collecting more than 524 million multilingual tweets across 218 countries and 47,000 cities between the dates of February 1, 2020 and May 1, 2020. The data collection started on February 1, 2020 using trending hashtags such as #covid19, #coronavirus, #covid_19. See Qazi et al. (2020) for complete details. Given the large amount of politically divisive discourse, racist remarks, and social impact of Covid-19, the collection provides a unique testbed to understand geographic model

variation. Particularly, where researchers are exploring analyzing geospatial patterns of Covid-related content on Twitter (Stephens, 2020). If the models perform differently across locations, the it is difficult to interpret the results. We refer to this complete Covid-19 data as "Original Data".

Step 2: City Selection (Unfiltered GeoOLID). To measure the performance difference across varying locations, we decided on 15 cities based on multiple facets, data availability, geographic diversity, and demographic diversity. In deciding which cities to use for our study we first selected cities from different parts of the United States (North, South, East, West). Next we wanted cities that varied in size and were also demographically different. In table 3, the total populations (reported in the thousands) of the selected cities varies, ranging from around 400,000 to almost 9 million.

We also wanted cities that varied demographically, particularly with regard to African American and Hispanic/Latino population proportions.[2] In Table 9, cities like Baltimore, Memphis, New Orleans, and Detroit were chosen due to the high proportion of African Americans populations while, Indianapolis and Columbus had high proportions of White Non-Hispanic residents. El Paso, San Antonio and Phoenix have a close proximity to the Mexico boarder and higher percentage of Latino and Hispanic residents, which is very different from Columbus, having a smaller number of African American and Hispanic residents. In addition, we selected cities where we knew residents could use very distinct accents and phonics like New York and New Orleans. Overall, by selecting the 15 cities in Table 1, we created a diverse dataset with multiple ethnicities, language styles, and topical differences. We refer to this unlabeled dataset as "Unfiltered GeoOLID". The basic stats of this dataset are available in Table 1 in the row titled "Unfiltered GeoOLID".

Step 3: GeoOLID. Similar to prior work, we need to sample a large number of offensive and non-offensive tweets from Unfiltered GeoOLID (Zampieri et al., 2019). Hence, we filter Unfiltered GeoOLID using the following lexicons and keyword filters: the badword lexicon (von Ahn, 2009), hatebase lexicon (Davidson et al., 2017), offensive-related phrases used for the original OLID dataset (Zampieri et al., 2019) ("you are", "she is", "he is", "conservatives", "liberals", "MAGA", and "antifa"), and additional Covid-specific phrases we found to be correlated with potential discrimination in the dataset ("chinese", "china", "asia", "asian", "wuhan"). Along with the aforementioned filters, we randomly sampled a subset of tweets for annotation. The final counts of each city can be found in Table 1. This dataset is referred to as "GeoOLID".

Annotation. Overall, we performed multiple rounds of annotation until a quality dataset was created. First, in order to provide accurate labels for this study, samples of tweets were assigned to three graduate students to be labeled as "offensive" or "not-offensive" using the base guidelines provided by Zampieri et al. (2019) for the the OLID dataset. A total of 20 students were recruited and given a stipend of $100 for their time and effort. Several meetings were set up before labeling started to answer questions and address implications. We use the Offensive definition provided by Zampieri et al. (2019) is defined as *tweets containing any form of non-acceptable language (profanity) or a targeted offense, which can be veiled or direct. This includes insults, threats, and posts containing profane language or swear words.*

Following general annotation recommendations for NLP (Pustejovsky and Stubbs, 2012), the annotation process was completed in three stages to increase the reliability of the labels across geographic regions. First, before assigning tweets, we assured every tweet was assigned to three graduate students for annotation, providing us with three independent labels for each tweet. We then calculated the agreement between annotators, resulting in a Fleiss Kappa of .47, indicating moderate agreement.

Second, we (the authors) of the paper manually—and independently—adjudicated (i.e., re-annotated) the labels of each student, correcting miss-annotated tweets that were not agreed on by all three annotators. Common issues found during the process were labels of "Not Offensive" for tweets with ad-hoc mentions of the "Wuhan Virus" and offensive content found in the hashtag. Specifically, based on the work by Dubey (2020), we decided that mentions of "Wuhan Virus" and other related terms like "China Flu" and "Kung Flu" were deemed offensive as it fit into the category of an targeted offense, which can be veiled or direct. The second round of agreement scores increased to .83

[2]We choose these groups because they align with classes in the Blodgett et al. (2016) dialect classifier.

representing "almost perfect agreement," (Landis and Koch, 1977).

To further ensure annotation quality, the authors went through the tweets once again discussing and correcting any final disagreements among the second round adjudications, forming the final dataset described in Table 1. After collecting and adjudicating the responses, the total number of Offensive tweets were 4,831 compared to 9,259 Not Offensive. We also report Minimum Detectable Effect (MDE) (Card et al., 2020) for Accuracy in Table 1. Specifically, use the Binomial Power Test, which assumes that samples are unpaired, i.e., the new model and baseline evaluation samples are drawn from the same data distribution but are not necessarily the same samples. The MDE numbers assume an accuracy of .75, which results in a significant difference between two models being around .05. We plot more potential MDE scores for different baseline numbers in the Appendix, Figure 5.

5 Experiments

In order to address and test whether performance disparities exist across geographic regions for offensive language classifiers, we ran multiple experiments. We analyzed performance across the 15 cities in the GeoOLID dataset. In the following subsections, we provide the details of our experiments and provide evidence supporting that our GeoOLID dataset is representative of the Unfiltered GeoOLID dataset and that offensive language classifier performance can vary by geolocation. In the final subsection, we explore the performance and language similarities across different geolocations that have similar demographics.

5.1 Data Representation Evaluation

In this section, we aim to measure how well the GeoOLID dataset matches the Unfiltered GeoOLID data from each city. Specifically, we want to ensure that patterns found in the unfiltered data are still present within our annotated GeoOLID sample. If patterns in the GeoOLID dataset are not in Unfiltered GeoOLID, it is hard to argue that the errors are location-specific. They could simply be caused by our data filtering strategy.

Methods. To measure how representative our sample is, we train a location prediction model. Given a tweet, the goal of the model is to predict the city in which the text was posted. To train the model we use two sets of features: Content Features and Stylistics Features. The content features are made up of the top 5000 unigrams in the Unfiltered GeoOLID dataset. It is also important to note that all of the GeoOLID tweets are removed from the Unfiltered GeoOLID dataset before processing.

	F1	Acc.
Stratified	.059	.056
Uniform	.062	.062
Prior	.008	.068
BoW	.430	.380
POS	.410	.356
Dialect	.374	.366
POS + Dialect	.419	.357
BoW + Dialect	**.436**	**.381**
BoW + POS + Dialect	.431	.370

Table 2: Location prediction. The Accuracy, Macro Precision, Macro Recall, and Macro F1 reported are the results when trained on a sample of the Unfiltered GeoOLID and predicted on the labeled GeoOLID dataset.

We also explore two sets of style Features: Part-of-Speech and Dialect Features. Specifically, we use unigram, bigram, trigram POS features. Moreover, the dialect features are the probabilities returned from the dialect inference tool from Blodgett et al. (2016). Given a tweet, the tool outputs the proportion of African-American, Hispanic, Asian, and White topics.

Finally, we train a Random Forest classifier on the Unfiltered GeoOLID dataset and the results are reported using the labeled GeoOLID dataset as the test set. Hyperparameters are optimized using 10-fold cross-validation on the training data. Because of the large size of the Unfiltered GeoOLID dataset, we sample a random subset of 35k examples from the Unfiltered GeoOLID dataset to reduce the training cost. The goal is not to achieve the most accurate predictions, but to simply see if we can predict location much better than random. If a small completely random sample shows this, that is better then requiring all of the data. We also compare the results to three random sampling methods to measure the difference between random guessing and the trained model: Stratified, Uniform, and Prior. Stratified makes random predictions based on the distribution of the cities in the training data, Uniform predicts cities with equal proportions, and Prior always predicts the most frequent city.

Results. The results of the experiments are reported in Table 2. Using content and style features, we were able to predict the location of a tweet

	AAS	HLS	Tot.	AA	H/L
Baltimore	.168	.193	585	338 (57.7%)	45 (7.8%)
Chicago	.147	.204	2,450	801 (32.7%)	819 (33.4%)
Columbus	.146	.201	905	259 (28.6%)	70 (7.7%)
Detroit	.196	.214	639	496 (77.7%)	51 (8.0%)
El Paso	.158	.227	678	25 (3.7%)	551 (81.2%)
Houston	.161	.205	2,304	520 (22.6%)	1,013 (44.0%)
Indianapolis	.151	.194	887	248 (28.0%)	116 (13.1%)
Los Angeles	.144	.204	3,898	336 (8.6%)	1,829 (47.0%)
Memphis	.209	.220	633	389 (61.6%)	62 (9.8%)
Miami	.140	.175	442	57 (12.9%)	310 (7.0%)
New Orleans	.182	.197	383	208 (54.2%)	31 (8.0%)
New York City	.126	.182	8,804	1,943 (22.1%)	2,490 (28.3%)
Philadelphia	.157	.204	887	248 (27.9%)	116 (13.1%)
Phoenix	.144	.208	1,608	125 (7.8%)	661 (41.1%)
San Antonio	.175	.222	1,434	102 (7.2%)	916 (63.9%)
AA PCC	.565 (p value: .028)				
H/L PCC	.167 (p value: .55)				

Table 3: Pearson Correlation Coefficient (PCC) between the AAS and HLS and city populations. Populations reported in thousands and percentages are in parenthesis.

more than 38% of the time, an increase of almost 140% in accuracy than the best random baseline, suggesting that both content and style features are predictive of the location a tweet is made. Likewise, using the POS and dialect features alone, the model achieves an accuracy of more than 35%, substantially higher than the random baselines. Given that there are only four dialect features, this is indicative that the group information detected by the Blodgett et al. (2016) is informative. Similarly, the POS results are also high, indicating that there are unique combinations of POS patterns that appear in each location. Overall, the findings show that our subsample (GeoOLID) is representative of patterns found in the Unfiltered GeoOLID dataset.

Discussion. Blodgett et al. (2016) show that the assumption that cities with large African American populations will have more text classified as AAE. Hence, as a simple robustness test, we use the tool provided by Blodgett et al. (2016) to correlate it with the demographic information of each city in our labeled GeoOLID dataset. Specifically, using the 2020 US Census data, we calculate the proportion of "Black or African American alone" (AA) and "Hispanic or Latino" (H/L) residents for each city. We also calculate the average African-American (AAS) and Hispanic (HS) scores for each city using the tool from Blodgett et al. (2016). Finally, we calculate the Pearson Correlation Coefficient (PCC) AA and AAS (and H/L and HS). The goal is to show that the findings found in Blodgett et al. (2016) hold on our GeoOLID dataset, i.e., their tool's scores correlate with minority populations. If they do, this provides further evidence that our dataset is representative of each location.

This correlation can be seen in Table 3. Overall we find that there is significant correlation .565 (p value: .028) between the two variables. We find that cities like Baltimore, New Orleans and Detroit are more likely to have more AAE tweets (in the labeled GeoOLID dataset) then cities like Miami, Columbus, and New York. For the Hispanic group we also find a positive correlation but the finding is not significant. We also manually analyzed the dataset and found other features indicative of a relationship between demographics of the city and language use. For example, we found Spanish curse words appearing in text in cities with higher Hispanic populations in our dataset, e.g., "Nationwide shutdown! pinché Cabron" is an slightly modified tweet that was tagged in Phoenix, AZ.

5.2 Data Variation and Model Performance

Next, we measure how much offensive language detection performance can vary location-to-location. Given the same model is applied to every city, ideally, we would have similar performance universally. However, if we see large variation in performance metrics and if the errors are caused by patterns also represented in the Unfiltered GeoOLID dataset, this is indicative of geographic performance disparities.

Methods. We train five different machine learning algorithms: Linear Support Vector Machine (Linear SVM), Long Short Term Memory (LSTM), Bidirectional LSTM (BiLSTM), Convolutional Neural Networks (CNN), and a Bidirectional Encoder Representations from Transformers (BERT). Each model is trained to classify Offensive and Non Offensive tweets using the OLID dataset.One thing to note is for the BiLSTM, CNN and LSTM, we also measure the performance of the model across multiple word embeddings. Specifically, each deep learning model is trained using different variations of Glove, Google Word2Vec and Fasttext word embedding (See the Appendix, Table 7, for a complete listing of the evaluated embeddings).

For evaluation, We train multiple models on the OLID dataset using a 5-fold shuffle-split cross-validation procedure. Specifically, a model is trained on each training split of the OLID dataset, then it is applied to the GeoOLID dataset to calculate each city's model performance. A 10% portion of the OLID training split for each fold is used for

	Bal	Chi	Col	Det	ElP	Hou	Ind	LA	Mem	Mia	NO	NY	Phi	Pho	SA	AVG
Stratified	.279	.328	.306	.367	.431	.348	.364	.303	.363	.253	.322	.296	.376	.394	.410	.343
Uniform	.362	.443	.398	.458	.453	.381	.390	.368	.420	.302	.412	.342	.412	.444	.456	.403
Linear SVM	.661	.615	.650	.714	.658	.568	.611	.643	.702	.660	.708	.625	.680	.613	.680	.653
BiLSTM	.678	.623	.651	.725	.660	.591	.643	.642	.720	.666	.694	.624	.705	.637	.669	.662
CNN	**.720**	**.662**	**.684**	**.745**	**.688**	**.611**	**.674**	**.670**	**.745**	**.701**	**.736**	**.663**	**.743**	**.657**	**.703**	**.694**
LSTM	.653	.614	.633	.709	.638	.570	.624	.620	.701	.661	.680	.600	.686	.615	.650	.643
BERT	. 601	.629	.641	.684	.661	.602	.621	.642	.651	.614	.635	.593	.668	.607	.665	. 634
AVG	.663	.629	.652	.715	.661	.588	.635	.643	.704	.660	.691	.621	.696	.626	.673	

Table 4: F1 score for each model-city combination. The largest and smallest average values are shaded in blue and red, respectively. The model with the *largest* F1 (larger is better) is **bolded** for each city. Each city's shortname is as follows: Chicago (Chi), Detroit (Det), Baltimore (Bal), El Paso (ElP), Los Angeles (LA), Houston (Hou), Columbus(Col), Indianapolis (Ind), Miami (Mia), Memphis (Mem), New York City (NYC), New Orleans (NO), San Antonio (SA), Philadelphia (Phi), and Phoinex (Pho).

	Bal	Chi	Col	Det	ElP	Hou	Ind	LA	Mem	Mia	NO	NY	Phi	Pho	SA	AVG
Linear SVM	.187	.193	.218	.233	.239	.211	.167	.172	.247	.152	.194	.151	.191	.247	.220	.201
BiLSTM	.111	.100	.144	.149	.154	.142	.115	.119	.142	.085	.118	.104	.116	.154	.143	.126
CNN	.124	.106	.155	.168	.168	.159	.112	.137	.167	.091	.115	.104	.126	.174	.164	.138
LSTM	.111	.092	.133	.137	.146	.134	.102	.114	.134	.079	.105	.099	.108	.145	.135	.118
BERT	**.105**	**.069**	**.113**	**.087**	**.075**	**.075**	**.068**	**.078**	**.118**	**.059**	**.086**	**.070**	**.091**	**.115**	**.097**	**.086**
AVG	.128	.112	.153	.155	.156	.145	.113	.124	.162	.093	.124	.106	.126	.167	.152	

Table 5: False positive rate (FPR) for each model-city combination. The largest and smallest average values are shaded in blue and red, respectively. The model with the *smallest* (smaller is better) FPR is **bolded** for each city.

	AA	Hispanic
Spearman	-.005	-.103
PCC	.102	-.018
AAE vs SAE Results		
AAE FPR	.154	(3392)
SAE FPR	.092	(5789)

Table 6: Correlation (Spearman and PCC) between the FPR scores and AA and H/L population proportions of each city.

hyperparameter selection. This procedure has been used in prior work to ensure robust results in similar social media-related NLP studies (Yin et al., 2017; Elejalde et al., 2017; Samory et al., 2020).

Results. In Table 4,we report the F1 of the OLID model applied to the GeoOLID dataset. Overall, we find substantial variation in model accuracy across the 15 cities. The average F1 for Houston (averaged across the non-random baselines) ranges from .588 (Houston) to .715 (Detroit), nearly 13% percent absolute difference and 22% relative difference. The CNN model achieves the best performance on average. Likewise, we find that CNN's best results are for the cities of Baltimore, Detroit, Memphis, and Philadelphia. Conversely, the CNN's worst results are found in Houston, Phoenix, Chicago, and New York.

Table 5 reports the False Positive Rates (FPR) for each city. Again, we see large variation, ranging from .093 to .167, nearly an 80% relative difference). On the other hand, we find that the best performing model is consistent across all cities. Hence, if a model performs better in Houston, it is likely it will perform better in Detroit. However, just because the model is better, the performance can be very low when compared to another location. Hence, decision-makers must carefully evaluate models based on the people impacted by them and not rely on evaluation metrics calculated on non-representative data before using the model. Finally, we report Accuracy results in the Appendix, Table 10, which show improvement greater than chosen MDE thresholds.

Prior work by Sap et al. (2019) show that offensive language detection models generate more false positives for text written in AAE. We evaluate this on our GeoOLID dataset following a similar strategy as Sap et al. (2019). In Table 6, we use the Blodgett et al. (2016) tool to identify AAE (African American English) and SAE (Standard American English) tweets in our GeoOLID dataset across all cities. When we calculate the false positive rate (FPR) across these two aggregate groups, we find similar conclusions to prior work (Sap et al., 2019) suggesting that offensive language models generate

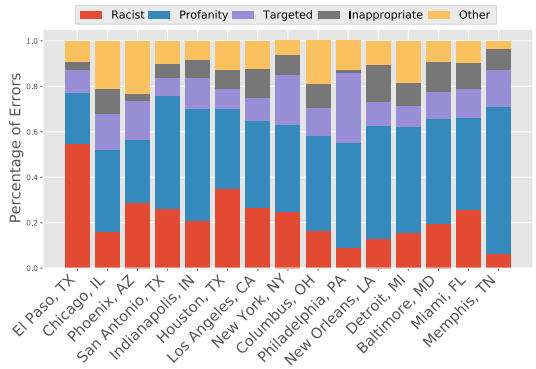

Figure 2: Manually coded false negatives per city.

more false positives on AAE text.

However, this does not mean that cities with larger African American populations will have larger FPRs. To test this using PCC and Spearman ρ, we correlate model performance (FPR) with the proportion of Black or African American and Hispanic/Latino residents (See Table 3) using US Census data. We find that there is weak to no correlation between FPR and minority population, which is surprising given AAE is correlated with population. There are two major reasons for this phenomena. First, AAE is not widely spoken. Even among African Americans, they do not always use an AAE dialect. Hence, if someone uses AAE sparingly, then the FPR on AAE will not accurately represent how the model will perform on text they write. Second, there are other factors that have a larger impact. In particular, this data is within the context of Covid-19. Hence, there are topics written with particular stances that the offensive langauge detection model is unable to handle, e.g., understanding the context of "wuhan virus".

Finally, we suggest that researchers should look at evaluating "contextual language", e.g., try to identify real people, ask them how they identify with regard to race and gender, then evaluate how models perform for them. This can provide insight into real bias issues and ground potential negative impact on real people. This idea fits with the narrative against treating certain groups as a monolith entities (Tadjiogueu, 2014; Erving and Smith, 2021).

Discussion. We perform a comprehensive manual analysis on the false negatives made by the best model on the OLID dataset. Specifically, we performed a qualitative open coding procedure to categorize the false negatives into commonly occurring groups. We allowed categories and meanings to emerge from posts in somewhat of an open coding fashion (Strauss and Corbin). We randomly sample up to 100 false negatives from each city, identifying the main categories. Next, a meeting was held where the main categories were discussed.

The final group of codes were identified as: Racist, Profanity, Targeted, Inappropriate, and Other. Racist was defined as a direct attack of mentioned of a race and/or ethnicity, Profanity as any sort of curse words, this could be in a hashtag or acronym. Targeted was defined as an attack on an individual, personal or group not associated to race or ethnicity, and finally Inappropriate is defined as any insensitive joke or sexual reference.

The results are summarized in Figure 2. A few important observations can be made from this graph. For instance, we find a large proportion of false negatives in the racist category in border cities, or cities in close proximity to Mexico (e.g., El Paso, Phoenix, San Antonio, and Houston). The reason the false negatives occur is based on the stance, topic, and way of Racist writing we found to be common in the border regions. For instance, we found many issues where the model did not detect language that refers to migrants being part of a "horde," meant to cause violence or destruction (this is common racist rhetoric at the time (Finley and Esposito, 2020)), as being offensive.

We counted the number of border-related topics using a small set of search terms (e.g., "border", "migrants", "immigrants", and "illegals") in the Original Data. We plot the results in Figure 3. We find that most of the border-related tweets are in states near Mexico (e.g., Texas, Arizona, New Mexico, and California). Hence, more false negatives caused by racist-categorized tweets about the border are more likely to be made in these cities, thus also increasing the likelihood of false negatives. Given the increase in border-related topics, this error is location-related.

Prior work has shown geographic differences in the use of swear words on social media (Carey, 2020; Grieve, 2015). We also found morphological variants of curse words in different cities that caused false negatives. For example, in New Orleans, Philadelphia, and Memphis there were many false negative tweets contain high percentages of Profanity due to multiple spellings of different swear words such as "phucking", "effing", "mothafucka", "biatches".

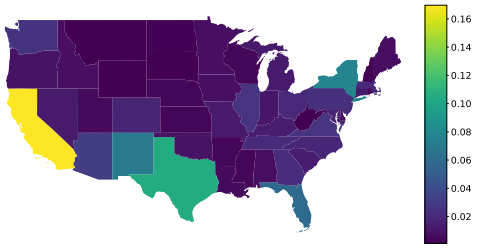

Figure 3: Proportion of border-related tweets in the "Original Data" for each state.

5.3 Geographic Similarities

In this subsection, we analyze the correlation between the best-performing models in each city.

Methods. We analyze the performance of the models trained and described in Section 5.2. Specifically, we compare the PCC between the Accuracy of each model applied to all cities. Intuitively, if the models for New York are sorted based on Accuracy, and the sorted order is the same as Phoenix, then the correlation would be one, showing a linear relationship. The more differences in sorted scores/models, the lower the performance (i.e., correlation). Overall, in this experiment, we rank every model along with the variants of models and compare every pair of cities rankings (i.e., each model trained with different word embeddings listed in the Appendix, Section A.2 are treated as independent models). Intuitively, if correlations are very high, this could indicate that you could choose the best hyperparameters for city A and they would be the best for city B. From the main results in Table 4 we saw that the best model was the same across all cities. However, this is with substantial hyperparameter optimization. Are the parameters the same for each city?

Results. The results of the correlation analysis are shown in Figure 4. Overall, similar to variations in model performance across cities, we find that the similarity in model performance correlations can vary substantially city-to-city. For instance, the best models for Houston are substantially different from other cities, except for a few (e.g., Los Angeles). However, on further inspection, general architecture performance seems to be relatively similar across cities, e.g., the CNN model is the best on the OLID dataset and for most cities. Much of the variation comes from hyperparameter choice or pretrained embedding choice (with more than 10% in Accuracy between the best and worst embeddings). The best embeddings can be substantially different city-to-city. This result suggests that choosing the best hyperparameters based on a small subset of data (e.g., from one city) is not optimal for each location, which can result in further performance disparities.

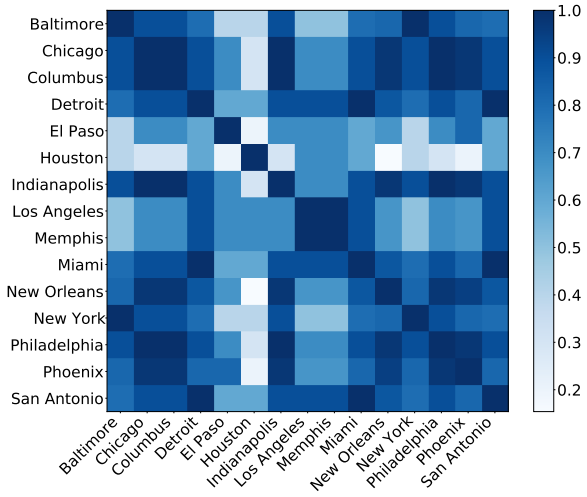

Figure 4: Model accuracy correlation between each pair of cities in GeoOLID.

Discussion. The results do provide us with a potential research avenue. An interesting question that could be explores is if we train a model with many hyperparameter options on a dataset, is it possible to predict which model to deploy in a given region? There has been some work in predicting model perform (Elsahar and Gallé, 2019). Hence, it would be interesting to expand that to predict the best hyperparameters.

6 Conclusion

We provide a comprehensive analysis of performance disparities of offensive language detection models. Furthermore, we introduce a novel dataset that provides more than 14 thousand examples for further analysis of geographical differences in model performance. The study points to the importance of geographically sensitive NLP, where the impact and performance of NLP models are analyzed for specific geographical regions, or even micro communities within a city. Moreover, finding regions where models perform poorly on can also provide unique testbeds as "hard test cases" similar to recent work on adversarial examples (Zhang et al., 2019).

Acknowledgements

This material is based upon work supported by the National Science Foundation (NSF) under Grant No. 1947697 and NSF award No. 2145357.

References

Naheed Ahmed, Sandra C Quinn, Gregory R Hancock, Vicki S Freimuth, and Amelia Jamison. 2018. Social media use and influenza vaccine uptake among white and african american adults. *Vaccine*, 36(49):7556–7561.

Pinkesh Badjatiya, Manish Gupta, and Vasudeva Varma. 2019. Stereotypical bias removal for hate speech detection task using knowledge-based generalizations. In *The World Wide Web Conference*, pages 49–59.

David Bamman, Chris Dyer, and Noah A. Smith. 2014. Distributed representations of geographically situated language. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 828–834, Baltimore, Maryland. Association for Computational Linguistics.

Su Lin Blodgett, Lisa Green, and Brendan O'Connor. 2016. Demographic dialectal variation in social media: A case study of African-American English. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 1119–1130, Austin, Texas. Association for Computational Linguistics.

Dallas Card, Peter Henderson, Urvashi Khandelwal, Robin Jia, Kyle Mahowald, and Dan Jurafsky. 2020. With little power comes great responsibility. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 9263–9274, Online. Association for Computational Linguistics.

Stan Carey. 2020. Mapping the united swears of america. Visited: 05/15/2022.

Carlos Castillo. 2016. *Big crisis data: social media in disasters and time-critical situations*. Cambridge University Press.

Nikhil Cheke, Joydeep Chandra, Sourav Kumar Dandapat, et al. 2020. Understanding the impact of geographical distance on online discussions. *IEEE Transactions on Computational Social Systems*, 7(4):858–872.

Courtney Corley, Diane Cook, Armin Mikler, and Karan Singh. 2010. Text and structural data mining of influenza mentions in web and social media. *International journal of environmental research and public health*, 7(2):596–615.

Courtney Corley, Armin R Mikler, Karan P Singh, and Diane J Cook. 2009. Monitoring influenza trends through mining social media. In *BIOCOMP*, pages 340–346.

Thomas Davidson, Debasmita Bhattacharya, and Ingmar Weber. 2019. Racial bias in hate speech and abusive language detection datasets. In *Proceedings of the Third Workshop on Abusive Language Online*, pages 25–35, Florence, Italy. Association for Computational Linguistics.

Thomas Davidson, Dana Warmsley, Michael Macy, and Ingmar Weber. 2017. Automated hate speech detection and the problem of offensive language. In *Proceedings of the International AAAI Conference on Web and Social Media*, volume 11.

Lucas Dixon, John Li, Jeffrey Sorensen, Nithum Thain, and Lucy Vasserman. 2018. Measuring and mitigating unintended bias in text classification. In *Proceedings of the 2018 AAAI/ACM Conference on AI, Ethics, and Society*, pages 67–73.

Gabriel Doyle. 2014. Mapping dialectal variation by querying social media. In *Proceedings of the 14th Conference of the European Chapter of the Association for Computational Linguistics*, pages 98–106, Gothenburg, Sweden. Association for Computational Linguistics.

Akash Dutt Dubey. 2020. The resurgence of cyber racism during the covid-19 pandemic and its aftereffects: analysis of sentiments and emotions in tweets. *JMIR Public Health and Surveillance*, 6(4):e19833.

Erick Elejalde, Leo Ferres, and Eelco Herder. 2017. The nature of real and perceived bias in chilean media. In *Proceedings of the 28th ACM Conference on Hypertext and Social Media*, pages 95–104.

Hady Elsahar and Matthias Gallé. 2019. To annotate or not? predicting performance drop under domain shift. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 2163–2173, Hong Kong, China. Association for Computational Linguistics.

Christy L Erving and Monisola Vaughan Smith. 2021. Disrupting monolithic thinking about black women and their mental health: Does stress exposure explain intersectional ethnic, nativity, and socioeconomic differences? *Social Problems*.

Joel Escudé Font and Marta R. Costa-jussà. 2019. Equalizing gender bias in neural machine translation with word embeddings techniques. In *Proceedings of the First Workshop on Gender Bias in Natural Language Processing*, pages 147–154, Florence, Italy. Association for Computational Linguistics.

Laura Finley and Luigi Esposito. 2020. The immigrant as bogeyman: Examining donald trump and the right's anti-immigrant, anti-pc rhetoric. *Humanity & Society*, 44(2):178–197.

Mihaela Gaman, Dirk Hovy, Radu Tudor Ionescu, Heidi Jauhiainen, Tommi Jauhiainen, Krister Lindén, Nikola Ljubešić, Niko Partanen, Christoph Purschke, Yves Scherrer, and Marcos Zampieri. 2020. A report on the VarDial evaluation campaign 2020. In *Proceedings of the 7th Workshop on NLP for Similar Languages, Varieties and Dialects*, pages 1–14, Barcelona, Spain (Online). International Committee on Computational Linguistics (ICCL).

Daniela Gerz, Ivan Vulić, Edoardo Maria Ponti, Roi Reichart, and Anna Korhonen. 2018. On the relation between linguistic typology and (limitations of) multilingual language modeling. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 316–327, Brussels, Belgium. Association for Computational Linguistics.

Jack Grieve. 2015. Research blog: Jack grieve's homepage swear word mapping. Visited: 05/15/2022.

Dirk Hovy and Christoph Purschke. 2018. Capturing regional variation with distributed place representations and geographic retrofitting. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 4383–4394, Brussels, Belgium. Association for Computational Linguistics.

Dirk Hovy, Afshin Rahimi, Timothy Baldwin, and Julian Brooke. 2020. Visualizing regional language variation across europe on twitter. *Handbook of the Changing World Language Map*, pages 3719–3742.

Dirk Hovy and Anders Søgaard. 2015. Tagging performance correlates with author age. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 2: Short Papers)*, pages 483–488, Beijing, China. Association for Computational Linguistics.

Taylor Jones. 2015. Toward a description of african american vernacular english dialect regions using "black twitter". *American Speech*, 90(4):403–440.

Olga Kellert and Nicholas H Matlis. 2021. Geolocation differences of language use in urban areas. *arXiv preprint arXiv:2108.00533*.

Alex Lamb, Michael J. Paul, and Mark Dredze. 2013. Separating fact from fear: Tracking flu infections on Twitter. In *Proceedings of the 2013 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 789–795, Atlanta, Georgia. Association for Computational Linguistics.

Rabindra Lamsal. 2021. Design and analysis of a large-scale covid-19 tweets dataset. *Applied Intelligence*, 51(5):2790–2804.

J Richard Landis and Gary G Koch. 1977. The measurement of observer agreement for categorical data. *biometrics*, pages 159–174.

Brandon Lwowski and Peyman Najafirad. 2020. COVID-19 surveillance through Twitter using self-supervised and few shot learning. In *Proceedings of the 1st Workshop on NLP for COVID-19 (Part 2) at EMNLP 2020*, Online. Association for Computational Linguistics.

Brandon Lwowski and Anthony Rios. 2021. The risk of racial bias while tracking influenza-related content on social media using machine learning. *Journal of the American Medical Informatics Association*, 28(4):839–849.

Ninareh Mehrabi, Thamme Gowda, Fred Morstatter, Nanyun Peng, and Aram Galstyan. 2020. Man is to person as woman is to location: Measuring gender bias in named entity recognition. In *Proceedings of the 31st ACM Conference on Hypertext and Social Media*, pages 231–232.

Marzieh Mozafari, Reza Farahbakhsh, and Noël Crespi. 2020. Hate speech detection and racial bias mitigation in social media based on bert model. *PloS one*, 15(8):e0237861.

Graham Neubig, Yuichiroh Matsubayashi, Masato Hagiwara, and Koji Murakami. 2011. Safety information mining — what can NLP do in a disaster—. In *Proceedings of 5th International Joint Conference on Natural Language Processing*, pages 965–973, Chiang Mai, Thailand. Asian Federation of Natural Language Processing.

Ray Oshikawa, Jing Qian, and William Yang Wang. 2020. A survey on natural language processing for fake news detection. In *Proceedings of the 12th Language Resources and Evaluation Conference*, pages 6086–6093, Marseille, France. European Language Resources Association.

Robert Östling and Jörg Tiedemann. 2017. Continuous multilinguality with language vectors. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 2, Short Papers*, pages 644–649, Valencia, Spain. Association for Computational Linguistics.

Hyunji Hayley Park, Katherine J. Zhang, Coleman Haley, Kenneth Steimel, Han Liu, and Lane Schwartz. 2021. Morphology matters: A multilingual language modeling analysis. *Transactions of the Association for Computational Linguistics*, 9:261–276.

Ji Ho Park, Jamin Shin, and Pascale Fung. 2018. Reducing gender bias in abusive language detection. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 2799–2804, Brussels, Belgium. Association for Computational Linguistics.

James Pustejovsky and Amber Stubbs. 2012. *Natural Language Annotation for Machine Learning: A guide to corpus-building for applications*. " O'Reilly Media, Inc.".

Umair Qazi, Muhammad Imran, and Ferda Ofli. 2020. Geocov19: A dataset of hundreds of millions of multilingual covid-19 tweets with location information. *ACM SIGSPATIAL Special*, 12(1):6–15.

Christian Reuter and Marc-André Kaufhold. 2018. Fifteen years of social media in emergencies: a retrospective review and future directions for crisis informatics. *Journal of contingencies and crisis management*, 26(1):41–57.

Anthony Rios. 2020. Fuzze: Fuzzy fairness evaluation of offensive language classifiers on african-american english. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pages 881–889.

Anthony Rios, Reenam Joshi, and Hejin Shin. 2020. Quantifying 60 years of gender bias in biomedical research with word embeddings. In *Proceedings of the 19th SIGBioMed Workshop on Biomedical Language Processing*, pages 1–13, Online. Association for Computational Linguistics.

Anthony Rios and Brandon Lwowski. 2020. An empirical study of the downstream reliability of pre-trained word embeddings. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 3371–3388, Barcelona, Spain (Online). International Committee on Computational Linguistics.

Mattia Samory, Vartan Kesiz Abnousi, and Tanushree Mitra. 2020. Characterizing the social media news sphere through user co-sharing practices. In *Proceedings of the International AAAI Conference on Web and Social Media*, volume 14, pages 602–613.

Mauricio Santillana, André T Nguyen, Mark Dredze, Michael J Paul, Elaine O Nsoesie, and John S Brownstein. 2015. Combining search, social media, and traditional data sources to improve influenza surveillance. *PLoS computational biology*, 11(10):e1004513.

Maarten Sap, Dallas Card, Saadia Gabriel, Yejin Choi, and Noah A. Smith. 2019. The risk of racial bias in hate speech detection. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 1668–1678, Florence, Italy. Association for Computational Linguistics.

Yves Scherrer and Nikola Ljubešić. 2021. Social media variety geolocation with geoBERT. In *Proceedings of the Eighth Workshop on NLP for Similar Languages, Varieties and Dialects*, pages 135–140, Kiyv, Ukraine. Association for Computational Linguistics.

Philippa Shoemark, James Kirby, and Sharon Goldwater. 2017. Topic and audience effects on distinctively Scottish vocabulary usage in Twitter data. In *Proceedings of the Workshop on Stylistic Variation*, pages 59–68, Copenhagen, Denmark. Association for Computational Linguistics.

Monica Stephens. 2020. A geospatial infodemic: Mapping twitter conspiracy theories of covid-19. *Dialogues in Human Geography*, 10(2):276–281.

A Strauss and J Corbin. Basics of qualitative research: Techniques and procedures for developing grounded theory sage publications, inc; 1998. translated by: Mohammadi, b. *Institute for Humanities and Cultural Studies.[In Persian]*.

Eteena J Tadjiogueu. 2014. Fifty shades of black: Challenging the monolithic treatment of black or african american candidates on law school admissions applications. *Wash. UJL & Pol'y*, 44:133.

Tami L Thomas, Ralph DiClemente, and Samuel Snell. 2014. Overcoming the triad of rural health disparities: How local culture, lack of economic opportunity, and geographic location instigate health disparities. *Health education journal*, 73(3):285–294.

Luis von Ahn. 2009. Offensive/profane word list. *Retrieved June*, 24:2018.

Jun-Ming Xu, Kwang-Sung Jun, Xiaojin Zhu, and Amy Bellmore. 2012. Learning from bullying traces in social media. In *Proceedings of the 2012 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 656–666, Montréal, Canada. Association for Computational Linguistics.

Zhijun Yin, Bradley Malin, Jeremy Warner, Pei-Yun Hsueh, and Ching-Hua Chen. 2017. The power of the patient voice: learning indicators of treatment adherence from an online breast cancer forum. In *Proceedings of the International AAAI Conference on Web and Social Media*, volume 11, pages 337–346.

Marcos Zampieri, Shervin Malmasi, Preslav Nakov, Sara Rosenthal, Noura Farra, and Ritesh Kumar. 2019. Predicting the type and target of offensive posts in social media. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 1415–1420, Minneapolis, Minnesota. Association for Computational Linguistics.

Huangzhao Zhang, Hao Zhou, Ning Miao, and Lei Li. 2019. Generating fluent adversarial examples for natural languages. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 5564–5569, Florence, Italy. Association for Computational Linguistics.

Jieyu Zhao, Tianlu Wang, Mark Yatskar, Vicente Ordonez, and Kai-Wei Chang. 2018. Gender bias in coreference resolution: Evaluation and debiasing methods. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 2 (Short Papers)*, pages 15–20, New Orleans, Louisiana. Association for Computational Linguistics.

Joshua N Zingher and Eric M Moore. 2019. The power of place? testing the geographic determinants of african-american and white voter turnout. *Social Science Quarterly*, 100(4):1056–1071.

A Appendix

A.1 Word Embeddings

In Table 7, we link to the publicly available word embeddings we use in our experiments. We test three models: SkipGram, GLOVE, and FastText. We also explore different embeddings sizes, ranging for 25 dimensions to 30. Moreover, we explore embeddings trained on different corpora, ranging from biomedical text (PubMed) to social media data (Twitter). The best embeddings are chosen based on the OLID validation dataset for all reported results in the main manuscript.

A.2 Model Hyper-parameters

In this Section, we report the best hyperparmeters for each model. For the linear models we also report the best TF-IDF settings from the scikit-learn package.

TF-IDF:

- sublinear tf: True
- min df: 5
- norm: l2
- encoding: latin-1
- ngram range: (1,2)
- stop words: english

Linear SVM:

- penalty: l2
- C: 1.0

CNN:

- max words: 10000
- max sequence length: 125
- drop: .2
- batch size: 512
- epochs: 30
- filter sizes: 3,4,5
- num filters: 512
- early stopping: 5 iterations

LSTM:

- max words: 10000
- max sequence length: 125
- drop: .2
- batch size: 128
- epochs: 30
- num filters: 512
- hidden layers: 1
- early stopping: 5 iterations

BiLSTM:

- max words: 10000
- max sequence length: 125
- drop: .2
- batch size: 128
- epochs: 30
- num filters: 512
- hidden layers: 1
- early stopping: 5 iterations

BERT:

- tokenizer : bert-base-cased
- model : bert-base-cased
- dropout : .2
- max length : 128
- epochs : 50
- batch size : 64
- fine tuned : after 5 epochs
- early stopping : 5 iterations

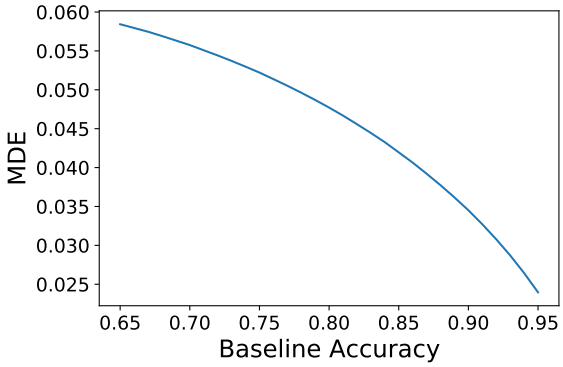

Figure 5: MDE given different baseline accuracy assumptions and a power of 80%.

A.3 OLID Results

We report the OLID results for each model (Linear SVM, CNN, LSTM, BiLSTM, and BERT) in Table 9. Interestingly, we find that the CNN model outperforms other methods, including the LSTM-based models and BERT. For instance, the CNN's F1 is more than 2% higher than the LSTM and BiLSTM models. Moreover, it is more than 6% higher than BERT. We also find that all methods outperform the traditional machine learning models (Linear SVM), with the CNN outperforming the Linear SVM by nearly 9% F1 and nearly 5% in Accuracy. The results support the results of the main paper with the CNN model generalizing better than other techniques.

Next, in Table 8 we report the performance of the CNN, LSTM, and BiLSTM models trained using different embeddings. Overall, we see variation across which embeddings result in teh best F1 score for each model, with wiki_42B_300d resulting in the highest F1 for the BiLSTM, wiki_840B_300d resulting in the best results for the LSTM, and GLOVE_twitter_27B_100d. This finding is similar to the results for H3 in the main paper, where embedding choice can vary city-to-city. We also find that it can vary model-to-model, which is also supported in Rios and Lwowski (2020).

A.4 Accuracy Power Analysis

In Figure 5, we report the MDE (Card et al., 2020) for Accuracy assuming different baseline scores and a power of 80%. For instance, if the baseline achieves an accuracy of .95, then we would need to see any improvement/difference of around .025 for it to be significant. Likewise, if the accuracy is around .65, then we need an improvement of nearly

.06 for it to be significant. Intuitively, the more accurate the results, the smaller the improvement can be for it be significant.

A.5 Accuracy Scores per City

In Table 10, we report the OLID model accuracy for each city. Overall, we find substantial variation in model accuracy across the 15 cities. The Linear SVM classifier ranges from .704 to .822, resulting in around a 12% difference in accuracy between Phoenix and Miami. Similar findings can be seen with the other models like CNN and BERT having a up to a 10% difference. Furthermore, given the MDE of around 5% for each city depending on the baseline score, we find that many of the differences are significant.

Model	Data Source	Dimension	Link
SkipGram	Google News	300	https://docs.google.com/file/d/0B7XkCwpI5KDYaDBDQm1tZGNDRHc/edit?usp=sharing
SkipGram	PubMed	200	http://evexdb.org/pmresources/vec-space-models/PubMed-w2v.bin
SkipGram	PubMed Central	200	http://evexdb.org/pmresources/vec-space-models/PMC-w2v.bin
SkipGram	PubMed and PubMed Central	200	http://evexdb.org/pmresources/vec-space-models/PubMed-and-PMC-w2v.bin
SkipGram	Wikipedia, PubMed, and PubMed Central	200	http://evexdb.org/pmresources/vec-space-models/wikipedia-pubmed-and-PMC-w2v.bin
GLOVE	Twitter	25	http://nlp.stanford.edu/data/glove.twitter.27B.zip
GLOVE	Twitter	50	http://nlp.stanford.edu/data/glove.twitter.27B.zip
GLOVE	Twitter	100	http://nlp.stanford.edu/data/glove.twitter.27B.zip
GLOVE	Twitter	200	http://nlp.stanford.edu/data/glove.twitter.27B.zip
GLOVE	Wikipedia 2014 and Gigaword 5	50	http://nlp.stanford.edu/data/glove.6B.zip
GLOVE	Wikipedia 2014 and Gigaword 5	100	http://nlp.stanford.edu/data/glove.6B.zip
GLOVE	Wikipedia 2014 and Gigaword 5	200	http://nlp.stanford.edu/data/glove.6B.zip
GLOVE	Wikipedia 2014 and Gigaword 5	300	http://nlp.stanford.edu/data/glove.6B.zip
GLOVE	Common Crawl V1	300	http://nlp.stanford.edu/data/glove.42B.300d.zip
GLOVE	Common Crawl V2	300	http://nlp.stanford.edu/data/glove.840B.300d.zip
FastText	Wikipedia 2017, UMBC webbase corpus, and statmt.org news dataset	300	https://dl.fbaipublicfiles.com/fasttext/vectors-english/wiki-news-300d-1M.vec.zip
FastText	Common Crawl	300	https://dl.fbaipublicfiles.com/fasttext/vectors-english/crawl-300d-2M.vec.zip

Table 7: List of word embeddings we use in our experiments.

Word Embedding	F1	Accuracy
BiLSTM		
FASTTEXT_en_300	.580	.760
GLOVE_twitter_27B_100d	.627	.785
GLOVE_twitter_27B_50d	.5834	.764
GLOVE_wiki_42B_300d	.645	.793
GLOVE_wiki_6B_100d	.600	.771
GLOVE_wiki_6B_200d	.605	.778
GLOVE_wiki_6B_300d	.631	.783
GLOVE_wiki_6B_50d	.586	.768
GLOVE_wiki_840B_300d	.631	.787
W2V_GoogleNews	.616	.781
W2V_PMC	.488	.730
W2V_PubMed_PMC	.514	.738
W2V_PubMed	.402	.704
LSTM		
FASTTEXT_en_300	.524	.749
GLOVE_twitter_27B_100d	.618	.782
GLOVE_twitter_27B_50d	.591	.770
GLOVE_wiki_42B_300d	.619	.790
GLOVE_wiki_6B_100d	.607	.774
GLOVE_wiki_6B_200d	.616	.781
GLOVE_wiki_6B_300d	.609	.782
GLOVE_wiki_6B_50d	.577	.762
GLOVE_wiki_840B_300d	.624	.788
W2V_GoogleNews	.602	.779
W2V_PMC	.456	.720
W2V_PubMed_PMC	.495	.730
W2V_PubMed	.348	.701
CNN		
FASTTEXT_en_300	.611	.778
GLOVE_twitter_27B_100d	.657	.792
GLOVE_twitter_27B_50d	.635	.788
GLOVE_wiki_42B_300d	.642	.793
GLOVE_wiki_6B_100d	.621	.779
GLOVE_wiki_6B_200d	.621	.786
GLOVE_wiki_6B_300d	.621	.785
GLOVE_wiki_6B_50d	.612	.775
GLOVE_wiki_840B_300d	.648	.794
W2V_GoogleNews	.638	.789
W2V_PMC	.520	.738
W2V_PubMed_PMC	.541	.743
W2V_PubMed	.461	.718

Table 8: Word Embedding Performance for Deep Learning Models

	Prec.	Rec.	F1	Acc.
Random Baselines				
Stratified	.324	.348	.336	.553
Uniform	.321	.505	.392	.493
Machine Learning Models				
Linear SVM	.643	.505	.566	.744
BiLSTM	.754	.551	.631	.783
CNN	.721	.603	.657	.792
LSTM	.768	.527	.624	.788
BERT	.652	.555	.592	.752

Table 9: OLID Results

	Bal	Chi	Col	Det	ElP	Hou	Ind	LA	Mem	Mia	NO	NY	Phi	Pho	SA	AVG
Stratified	.555	.555	.550	.536	.536	.570	.577	.567	.521	.592	.553	.588	.567	.544	.564	.558
Linear SVM	.779	.745	.751	.761	.694	.724	.748	.776	.752	.822	.787	.794	.771	.704	.740	.757
BiLSTM	.834	.809	**.799**	.803	.757	.774	.809	.824	**.818**	.861	.835	.842	.833	.768	.783	.809
CNN	**.843**	**.820**	.792	.823	.747	.773	**.819**	.805	.814	.851	**.842**	**.849**	.849	.760	**.788**	.811
LSTM	.832	.814	.790	**.834**	**.758**	.790	.817	**.829**	.810	**.873**	.837	.834	**.850**	**.772**	.783	**.815**
BERT	.786	.800	.788	.785	.755	**.791**	.790	.809	.761	.848	.785	.816	.803	.747	.771	.789
AVG	.815	.798	.784	.801	**.742**	.770	.797	.809	.791	**.851**	.817	.827	.821	.750	.773	.796

Table 10: Accuracy for each city in the GeoOLID dataset. In the bottom row (i.e., the average across all machine learning models), we mark the cities that have an average accuracy difference greater than or equal to the MDE compared to the city with the highest average accuracy.

Offensive Content Detection Via Synthetic Code-Switched Text

Cesa Salaam[1] **Franck Dernoncourt**[2] **Trung Bui**[2] **Danda Rawat**[1] **Seunghyun Yoon**[2]
[1]Howard University [2]Adobe Research
cesa.salaam@bison.howard.edu {franck.dernoncourt, bui, syoon}@adobe.com danda.rawat@howard.edu

Abstract

The prevalent use of offensive content in social media has become an important reason for concern for online platforms (customer service chat-boxes, social media platforms, etc). Classifying offensive and hate-speech content in online settings is an essential task in many applications that needs to be addressed accordingly. However, online text from online platforms can contain code-switching, a combination of more than one language. The non-availability of labeled code-switched data for low-resourced code-switching combinations adds difficulty to this problem. To overcome this, we release a human-generated dataset containing around 10k samples for testing for three language combinations en-fr, en-es, and en-de[1] and a synthetic code-switched dataset containing 30k samples for training[2]. In this paper, we describe the process for gathering the human-generated data and our algorithm for creating synthetic code-switched offensive content data. We also introduce the results of a keyword classification baseline and a multi-lingual transformer-based classification model.

1 Introduction

The use of offensive content in online settings such as chat-boxes, and social media platforms continues to be a growing problem that requires addressing. It can have negative effects on the psycho-emotional state of people (Saha et al., 2019). Offensive content and hate-speech continue to be a challenge to people world. As such, it is important to keep social media and other communication platforms free from offensive content. Considerable research has been conducted on deep-learning techniques for detecting offensive language (Pitsilis et al., 2018; Mehra and Hasanuzzaman, 2020). One of the growing challenges in the field of content detection is code-switching (Aguilar et al., 2020; Qin et al., 2020; Tang et al., 2020; Chakravarthi et al., 2020). Code-switching refers to the use of two or more languages in a single conversation. Code-switching can occur inter-sententially (across sentences) and intra-sententially (within sentences). The combination of code-switching and offensive content increases the complexity of the classification task. As code-switching is a combination of multiple languages, resources for these various combinations are extremely low. This causes researchers to find ways to create viable synthetic data that can serve in place of real-world data for training purposes. However, the real world benchmark test set still remains scarce.

To stimulate the research, we create human-annotated testsets written in three pairs of languages (en-fr, en-es, and en-de). Further, we propose a method for creating a synthetic train set and show its applicability to detect human-annotated code-switched text.

2 Related Works

Researchers have attempted to solve the problem of synthetic data generation for various code-switching tasks.

Theory Based Synthetic Code-switching Data Generation: Equivalency Theory (EC Theory) explains a range of interesting code-switched patterns beyond lexical substitution. The EC Theory describes a CM sentence as a constrained combination of two sentences that are equivalent. Pratapa et al. (2018) use EC Theory to generate meaningful artificial code-switched sentences.

Code-switched Offensive Content Datasets: Code-switching produces low resourced language combinations which presents many challenges for researchers in this field. Jose et al. (2020b) conducts a survey on currently available data-sets for various nlp tasks for code-switching. They mention data-sets for code-switching shared tasks Jose et al. (2020a), named entity recognition Singh et al.

[1]https://tinyurl.com/adobehuman
[2]https://tinyurl.com/adobesynthetic

Proceedings of the 29th International Conference on Computational Linguistics, pages 6617–6624
October 12–17, 2022.

Type	Original text	Annotation	Label
SWAP (en/de)	backpedaling fails to point out exactly my comments and resumes to nonsensical aussie retarded babbles go choke on a cock you useless wrinkly aussie slag you are the descendants of the bottom feeders among limeys loll	backpedalismus versagt, um genau meine Kommentare und Wiederholungen zu unsinnigen aussie retarded babbles gehen auf einem Hahn Sie nutzlos faltig aussie Schlampe wurgen Sie sind die Nachkommen der Bottom Feeder unter limeys loll	1
SWAP (en/fr)	if your humor is based on racism homosexualphobia sexism and rape you are not fucking funny go home	if your humor is based on racism homosexual phobia sexism and rape, no eres jodidamente divertido, go home	1
REWRITE (en/es)	This little fake ass meeting is making me anxious	This little fake culo meeting is making me anxious	1
REWRITE (en/fr)	I thought I was the only one who noticed his ass be saying a whole lot of nothing	I thought I was the only one who noticed que son cul ne disait pas grand-chose	1

Table 1: Examples of human-generated code-mixed text. In SWAP, the highlight shows the target code switches.

	EN-FR	EN-ES	EN-DE
human-generated SWAP test-set			
Number of samples	1,080	1,564	1,668
Average length (word-level)	21.53	24.01	25.10
human-generated REWRITE test-set			
Number of samples	2,000	2,000	1,948
Average length (word-level)	16.99	16.57	15.69
synthetic train-set			
Number of samples	9,926	9,926	9,926
Average length (word-level)	27.55	27.07	28.71

Table 2: Statistics of the human-generated testset and synthetic trainset. Annotators are asked to rewrite the code-mixed sentence (REWRITE) or only translate the pre-detected abusive word (SWAP).

(2018), sentiment analysis Sitaram et al. (2015), conversational systems Banerjee et al. (2018), machine translation (Dhar et al., 2018). While these data-sets offer a resource for code-switching related tasks, there still is a significant shortage in available data for tasks based on data for our code-switching combinations used in this research (en-fr,en-es,en-de). Our synthetic and human generated data-sets offer a set of resources to address this shortage.

For the task of Hate/Offensive speech detection, various approaches have been utilized. Dadu and Pant (2020) propose splitting a code-switched sentence into its constituent high resource languages to exploit both monolingual and cross-lingual settings. Kapoor et al. (2018) utilize apply transfer learning and design an LSTM based model of hate speech detection for hate speech and offensive speech in Hinglish (Hindi-english). Work performed on code-switched Tamil-English and Malayalam-English text includes corpus created for sentiment analysis for these two languages (Gupta et al., 2021).

Our work seeks to implement an algorithm for creating synthetic code-switched data and testing the efficacy of using that synthetic data for fine-tuning a multi-lingual language model for binary offensive content detection.

3 Dataset

The language combinations produced from code-switching can increase the complexities of NLP tasks (e.g. hatespeech detection, sentiment analysis, etc). To stimulate research in this domain and directly tackle the code-switched abusive language detection task, we create and release a 10k sentence test-set created by human annotators. Additionally, we generate and release around 30k sentence synthetic dataset to train a model (see the statistics in Table 2).

3.1 Benchmark Dataset Creation

Creating a benchmark test-set is an essential task for this study since it can stimulate the research further. To make the benchmark test reflect the real-world usage, we build the dataset from monolingual hate speech data created from real user text. We first take HateXplain (Mathew et al., 2021) data, which has fine-grained labels indicating the span related to the abusiveness. We ask bi-lingual human annotators to carefully translate the marked span (abusive words) into their second language (German, French, Spanish) and create code-mixed text. We called this dataset SWAP.

We further create a test-set by asking annotators to rewrite existing abusive text as a code switched version. We request annotators to REWRITE given sentences (HASOC (Mandl et al., 2020)) into a mix between English and their secondary language (German, French, Spanish). We ask annotators

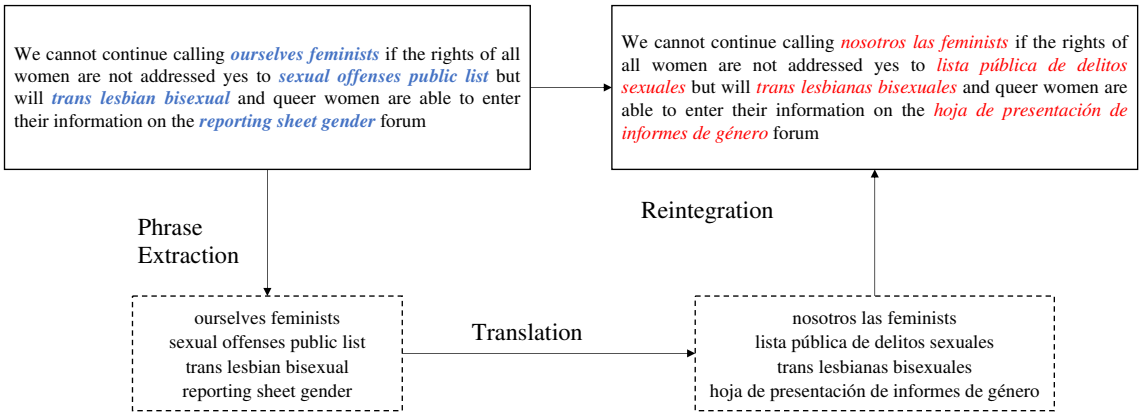

Figure 1: Synthetic Code Switching Generation Framework

to maintain hateful/offensive translations as much as possible. This process is focused on generating code switched text that represents the natural occurrence of code-switching. We called this dataset REWRITE.

We utilize MTurk[3] and Upwork[4] platforms for SWAP and REWRITE respectively to work with bi-lingual annotators and translators to generate diverse code-switched sentences. Table 1 shows examples of the input and output from the workers.

To validate the data generated by our human annotators, we resubmit the new code-switched sentences to MTurk. We ask workers to rate the sentences based on naturalness. We provide a scale from 1 (Excellent - completely natural code-switching) to 5 (Bad - completely unnatural code-switching). Sentences that receive a rating of 3 to 5 are resubmitted to MTurk workers to be re-written in a more natural manner (We provide further information in Appendix A).

3.2 Synthetic dataset generation

Due to the low-resourced nature of code-switching textual data, we generate synthetic training data to extend the training data for this classification task (see Figure 1). Our synthetic data generation occurs in three stages.

Phrase Identification The first stage in generation is the identification of phrases in the mono-lingual source text. We analyze existing real word abusive speech datasets, which are written in mixed languages (Bohra et al., 2018; Patwa et al., 2020) and find that one of the salient patterns is switching "noun phrase" in the sentence (Couto and Gullberg, 2019; Dorota et al., 2021). To specify the salient phrases in the sentences, we employ a pre-trained language model-based phrase tagging method (Gu et al., 2021). The original texts are passed into the tagging model to generate spans corresponding to the phrases in the sentences. Sentences that are not tagged with phrases are discarded from the dataset.

Phrase Translation Each phrase tagged in a sentence is then translated using the automatic machine translation model. We employ EasyNMT (Fan et al., 2021), an open-source state-of-the-art neural machine translation model that can translate 100+ languages. The phrases are fed into the translation model and translated to the destination language of our choice.

Phrase Reintegration After the phrases have been translated into the destination language of our choice, we then replace the tagged phrases in the source text with the new translated phrases. After the phrases have been reintegrated into the source text, the synthetically generated is ready to be utilized for training purposes.

To test the efficacy of our synthetic data generation framework, we first generate three hate-speech code-switching combinations, English-French (EN-FR), English-Spanish (EN-ES), and English-German (EN-DE). These language combinations are specifically chosen for there low-resourced nature in the hate-speech domain. The source text for this data is HateXplain (Mathew et al., 2021), a dataset of labeled hate-speech sentences sourced from the internet. We use the training subset of this data to generate our training data synthetically. Statistics of the synthetic data created can be seen in Table 2.

[3]https://www.mturk.com/
[4]https://www.upwork.com/

Model	SWAP testset						REWRITE testset					
	Eng-FR		Eng-ES		Eng-DE		Eng-FR		Eng-ES		Eng-DE	
	f1	WA	f1	WA	f1	WA	f1	WA	f1	WA	f1	WA
Dictionary	0.290	0.300	**0.540**	**0.570**	0.460	0.460	**0.660**	**0.680**	**0.670**	**0.690**	0.370	0.510
XLM-R_{syn}	**0.550**	**0.580**	0.530	0.550	**0.670**	**0.670**	0.530	0.580	0.590	0.620	**0.580**	**0.610**

Table 3: Experimental results on the benchmark testset. Each model is trained with synthetic dataset.

Model	Eng-FR		Eng-ES		Eng-DE	
	f1	WA	f1	WA	f1	WA
XLM-R_{syn}	0.530	0.580	0.590	0.620	**0.580**	**0.610**
XLM-R_{SWAP}	**0.610**	**0.620**	0.520	0.530	0.430	0.510
XLM-$R_{syn+SWAP}$	0.580	**0.620**	**0.600**	**0.630**	**0.580**	**0.610**

Table 4: Model is trained with synthetic, SWAP, or synthetic+SWAP and evaluated on REWRITE testset.

4 Method

We employ a human-annotated lexicon dictionary for abusive language and build a binary classification model as a baseline model. Furthermore, we explore the performance of the recently proposed multilingual neural network-based model.

4.1 Baseline Model

We leverage offensive and abusive speech lexicons sourced from (Hatebase) to develop a keyword-based classification algorithm. Specifically, we compiled four dictionary lexicons of hate-speech words from each language present (English, French, German, and Spanish). Each lexicon is used as a look-up table to determine if words present in given sentences are considered hate/offensive or not.

4.2 Transformer Based Model

To leverage the pretrained language model (PLM), we employ a multilingual model, XLM-RoBERTa (XLM-R), and build the abusive content clasifier (Conneau et al., 2019)[5]. In implementing the model, we feed the code-switched sentence to the XLM-R, and the "[CLS]" token is further passed through a two-layer fully-connected network. The final output is compared with the label, and loss is computed using the cross-entropy function (We provide more details in Appendix B).

[5]We also test other variants of multilingual models such as multilingual BERT (Devlin et al., 2019) and multilingual-DistilBERT (Sanh et al., 2019) on the downstream tasks and find the XLMR consistently shows superior performance.

5 Experimental Results and Discussion

To fine-tune the XLM-R model, we perform a learning rate schedule. We base the scheduling on the validation split macro F1 scores instead of using the loss from the validation. We adopt this approach from (Roy et al., 2021) where the authors focus on the validation scores at the end of each training iteration instead of using early-stopping to prevent over-fitting. If the validation performance decreases through an iteration, we backtrack to the previous model weights and decrease our learning rate. Training ends when the learning rate reaches a significantly small value. This type of scheduling guarantees that the Macro F1 score is maximized on the validation split.

We ran three experiments for both our dictionary and transformer-based models; (1) training on a synthetic dataset and testing on SWAP/REWRITE datasets, (2) training on SWAP, and testing on REWRITE, (3) training on synthetic and SWAP datasets, and testing on REWRITE. Table 3 and Table 4 show the experimental results in terms of F1 score and weighted accuracy (WA). An interesting observation in our experiment is the different results on our SWAP & REWRITE testsets. For instance, when code-switching semantics tend towards the swapping of offensive words between languages (SWAP testset), an LLM trained on our synthetic can perform better than dictionary-based detection (EN-DE). This is primarily due to the fact that our synthetic data generation algorithm is most similar to these types of occurrences. We also find that our synthetic dataset shows strong utility even better than human-annotated data (see Table 4). In other cases, we can see a decrease in performance when the structure of the code-switched sentences is more complex.

Based on some of these observations, we believe this algorithm can be useful in extending model training sets by mixing both synthetic data with real-world training data.

6 Conclusion

We released human-annotated testsets for the under-resourced en-fr, en-de, en-es language combinations (approximately 10k). Additionally, we proposed a synthetic code-switched data generation algorithm for training purposes in low resourced domains. Using this algorithm, we generated a synthetic offensive-content dataset comprised of 30k entries for en-fr, en-de, en-es language combinations. We create two baselines models and report their results on the human-annotated testsets. We expect this resource will enable the researchers to address new and exciting problems in code-switching research.

References

Gustavo Aguilar, Sudipta Kar, and Thamar Solorio. 2020. Lince: A centralized benchmark for linguistic code-switching evaluation. *arXiv preprint arXiv:2005.04322*.

Suman Banerjee, Nikita Moghe, Siddharth Arora, and Mitesh M. Khapra. 2018. A dataset for building code-mixed goal oriented conversation systems. In *COLING*.

Aditya Bohra, Deepanshu Vijay, Vinay Singh, Syed Sarfaraz Akhtar, and Manish Shrivastava. 2018. A dataset of Hindi-English code-mixed social media text for hate speech detection. In *Proceedings of the Second Workshop on Computational Modeling of People's Opinions, Personality, and Emotions in Social Media*, pages 36–41, New Orleans, Louisiana, USA. Association for Computational Linguistics.

Bharathi Raja Chakravarthi, Vigneshwaran Muralidaran, Ruba Priyadharshini, and John P McCrae. 2020. Corpus creation for sentiment analysis in code-mixed tamil-english text. *arXiv preprint arXiv:2006.00206*.

Alexis Conneau, Kartikay Khandelwal, Naman Goyal, Vishrav Chaudhary, Guillaume Wenzek, Francisco Guzmán, Edouard Grave, Myle Ott, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Unsupervised cross-lingual representation learning at scale. *arXiv preprint arXiv:1911.02116*.

Maria Carmen Parafita Couto and Marianne Gullberg. 2019. Code-switching within the noun phrase: Evidence from three corpora. *International Journal of Bilingualism*, 23(2):695–714.

Tanvi Dadu and Kartikey Pant. 2020. Towards code-switched classification exploiting constituent language resources. *arXiv preprint arXiv:2011.01913*.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. Bert: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186.

Mrinal Dhar, Vaibhav Kumar, and Manish Shrivastava. 2018. Enabling code-mixed translation: Parallel corpus creation and mt augmentation approach.

Gaskins Dorota, Oksana Bailleul, Anne Marie Werner, and Antje Endesfelder Quick. 2021. A crosslinguistic study of child code-switching within the noun phrase: A usage-based perspective. *Languages*, 6(1).

Angela Fan, Shruti Bhosale, Holger Schwenk, Zhiyi Ma, Ahmed El-Kishky, Siddharth Goyal, Mandeep Baines, Onur Celebi, Guillaume Wenzek, Vishrav Chaudhary, et al. 2021. Beyond english-centric multilingual machine translation. *Journal of Machine Learning Research*, 22(107):1–48.

Xiaotao Gu, Zihan Wang, Zhenyu Bi, Yu Meng, Liyuan Liu, Jiawei Han, and Jingbo Shang. 2021. Ucphrase: Unsupervised context-aware quality phrase tagging. *arXiv preprint arXiv:2105.14078*.

Akshat Gupta, Sargam Menghani, Sai Krishna Rallabandi, and Alan W. Black. 2021. Unsupervised self-training for sentiment analysis of code-switched data. *CoRR*, abs/2103.14797.

Hatebase. Hatebase, a collaborative, regionalized repository of multilingual hate speech. Accessed: 2021-10-29.

Navya Jose, Bharathi Chakravarthi, Shardul Suryawanshi, Elizabeth Sherly, and John McCrae. 2020a. A survey of current datasets for code-switching research. pages 136–141.

Navya Jose, Bharathi Raja Chakravarthi, Shardul Suryawanshi, Elizabeth Sherly, and John P. McCrae. 2020b. A survey of current datasets for code-switching research. *2020 6th International Conference on Advanced Computing and Communication Systems (ICACCS)*, pages 136–141.

Raghav Kapoor, Yaman Kumar, Kshitij Rajput, Rajiv Ratn Shah, Ponnurangam Kumaraguru, and Roger Zimmermann. 2018. Mind your language: Abuse and offense detection for code-switched languages. *CoRR*, abs/1809.08652.

Thomas Mandl, Sandip Modha, Anand Kumar M, and Bharathi Raja Chakravarthi. 2020. Overview of the hasoc track at fire 2020: Hate speech and offensive language identification in tamil, malayalam, hindi, english and german. In *Forum for Information Retrieval Evaluation*, pages 29–32.

Binny Mathew, Punyajoy Saha, Seid Muhie Yimam, Chris Biemann, Pawan Goyal, and Animesh Mukherjee. 2021. Hatexplain: A benchmark dataset for explainable hate speech detection. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 35, pages 14867–14875.

Sidharth Mehra and Mohammed Hasanuzzaman. 2020. *Detection of Offensive Language in Social Media Posts.* Ph.D. thesis.

Parth Patwa, Gustavo Aguilar, Sudipta Kar, Suraj Pandey, Srinivas PYKL, Björn Gambäck, Tanmoy Chakraborty, Thamar Solorio, and Amitava Das. 2020. Semeval-2020 task 9: Overview of sentiment analysis of code-mixed tweets. *CoRR*, abs/2008.04277.

Georgios K. Pitsilis, Heri Ramampiaro, and Helge Langseth. 2018. Detecting offensive language in tweets using deep learning. *CoRR*, abs/1801.04433.

Adithya Pratapa, Gayatri Bhat, Monojit Choudhury, Sunayana Sitaram, Sandipan Dandapat, and Kalika Bali. 2018. Language modeling for code-mixing: The role of linguistic theory based synthetic data. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1543–1553.

Libo Qin, Minheng Ni, Yue Zhang, and Wanxiang Che. 2020. Cosda-ml: Multi-lingual code-switching data augmentation for zero-shot cross-lingual NLP. *CoRR*, abs/2006.06402.

Sayar Ghosh Roy, Ujwal Narayan, Tathagata Raha, Zubair Abid, and Vasudeva Varma. 2021. Leveraging multilingual transformers for hate speech detection. *CoRR*, abs/2101.03207.

Koustuv Saha, eshwar chandrasekharan, and Munmun Choudhury. 2019. Prevalence and psychological effects of hateful speech in online college communities. volume 2019.

Victor Sanh, Lysandre Debut, Julien Chaumond, and Thomas Wolf. 2019. Distilbert, a distilled version of bert: smaller, faster, cheaper and lighter. *arXiv preprint arXiv:1910.01108*.

Vinay Singh, Deepanshu Vijay, Syed Sarfaraz Akhtar, and Manish Shrivastava. 2018. Named entity recognition for hindi-english code-mixed social media text. In *NEWS@ACL*.

Dinkar Sitaram, Savitha Murthy, Debraj Ray, Devansh Sharma, and Kashyap Dhar. 2015. Sentiment analysis of mixed language employing hindi-english code switching. *2015 International Conference on Machine Learning and Cybernetics (ICMLC)*, 1:271–276.

Tiancheng Tang, Xinhuai Tang, and Tianyi Yuan. 2020. Fine-tuning bert for multi-label sentiment analysis in unbalanced code-switching text. *IEEE Access*, 8:193248–193256.

A Data Collection

A.1 Sentence Generation

We generate code-switched sentences from the test-split of HateXplain and HASOC. The test-split of the HateXplain data-set contains sentences with words tagged by annotators that convey hate-speech and offensive content. These sentences and words are given to code-switching annotators on Amazon Mechanical Turk (Mturk) platform to perform the SWAP method as described in section 3.2. HASOC sentences do not contain annotated hate and offensive words and so this data is sent to bi-lingual translators on the Upwork platform.

MTurk is a crowdsourcing marketplace that simplifies the outsourcing of tasks to a distributed workforce who can perform these tasks virtually. Mturk allows individuals and businesses to post batches of assignments for workers.

On Upwork, three job posting are created with the following criteria:

- Fluency in English & (German, French or Spanish)
- Familiarity with colloquial terminology

Freelancers are then chosen based on the above criteria. The freelancers perform the REWRITE method of sentence generation as described in section 3.2.

A.2 Instructions to Annotators

Annotators for both the SWAP & REWRITE methods, are given instructions on how to complete the annotation tasks. An example of the SWAP annotation instructions and an example of a task on mturk can be seen in Fig. 2 and Fig. 3 respectively.

For SWAP, we request annotators to change a given English sentence into a mix of English and their native language (German, French, or Spanish) by focusing the switching on the provide list of words that are pre-determined to be hateful or offensive. If the sentence provided is not offensive, we then request that the annotator create a mixed version of the sentence based on their own discretion.

For REWRITE, we request annotators to rewrite the given sentences into a mix between English and their native language (German, French, or Spanish) based on their own discretion. We ask the annotators to maintain hateful or offensive translations as much as possible.

A.3 Validating Annotators' data

To validate the naturalness of the intial code-switched data generated by the MTurk and Upwork workers, we resubmitted the sentences to Mturk asking workers fluent in the language combinations to rate the code-switched sentences on their level of naturalness. This rating was done on the following scale:

- **Excellent - Completely natural code-mixing**
- **Good - Mostly natural code-mixing**
- **Fair - Equally natural and unnatural code-mixing**
- **Poor - Mostly unnatural code-mixing**
- **Bad - Completely unnatural code-mixing**

Sentences that received ratings from **fair** to **bad** were additionally resubmitted to Mturk for workers to rewrite the sentence in a more natural manner of code-switching.

A.4 Workers Pool & Pay

For MTurk, we hire the annotators whose locations is either France, Germany, Mexico, Spain. This restriction of location helps to ensure the annotators speak both the national language of the country as well as English We restrict the workers whose HIT approval rates are higher than 95%. We pay workers around 12 USD per hour.

For Upwork, we hire translators who are professionally fluent in either German, French, or Spanish. We choose the translators who best showcase the ability to create a code-switched rewrite by rewriting a few test examples. Each translator is paid according to a negotiated fee based on the number of sentences to REWRITE. We pay annotators 10 USD per 30 sentences, which is above the average rate for a similar task on Upwork.

B Reproducibility Checklist

- **Source code with specification of all dependencies, including external libraries**: The source code is included in the submission. It provides information about the dependencies including external libraries and instructions on how to run the proposed models.
- **Description of computing infrastructure used**: We use a single Tesla V100 GPU with 16GB memory in this work. PyTorch 1.1 is used to implement the models.

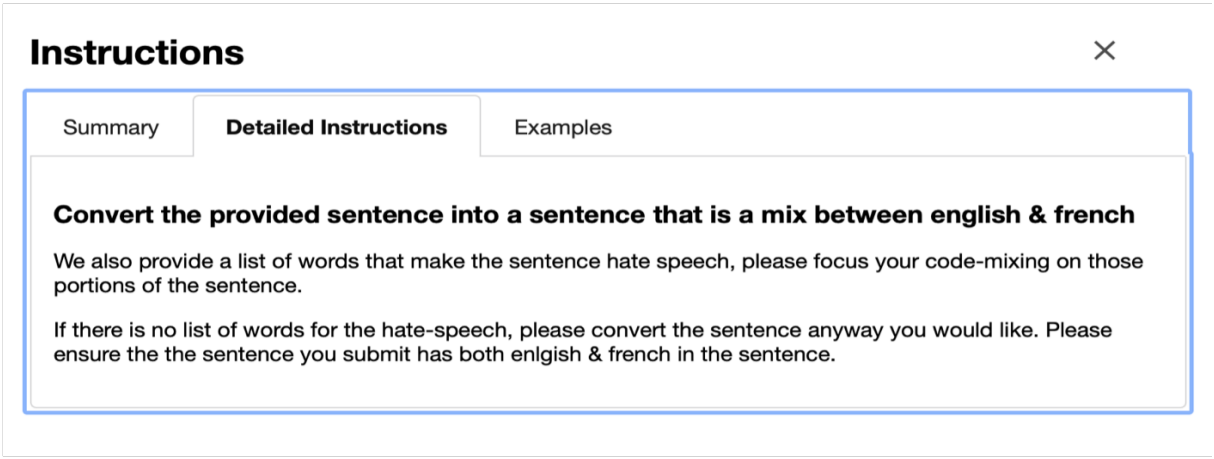

Figure 2: Annotator instructions on SWAP task

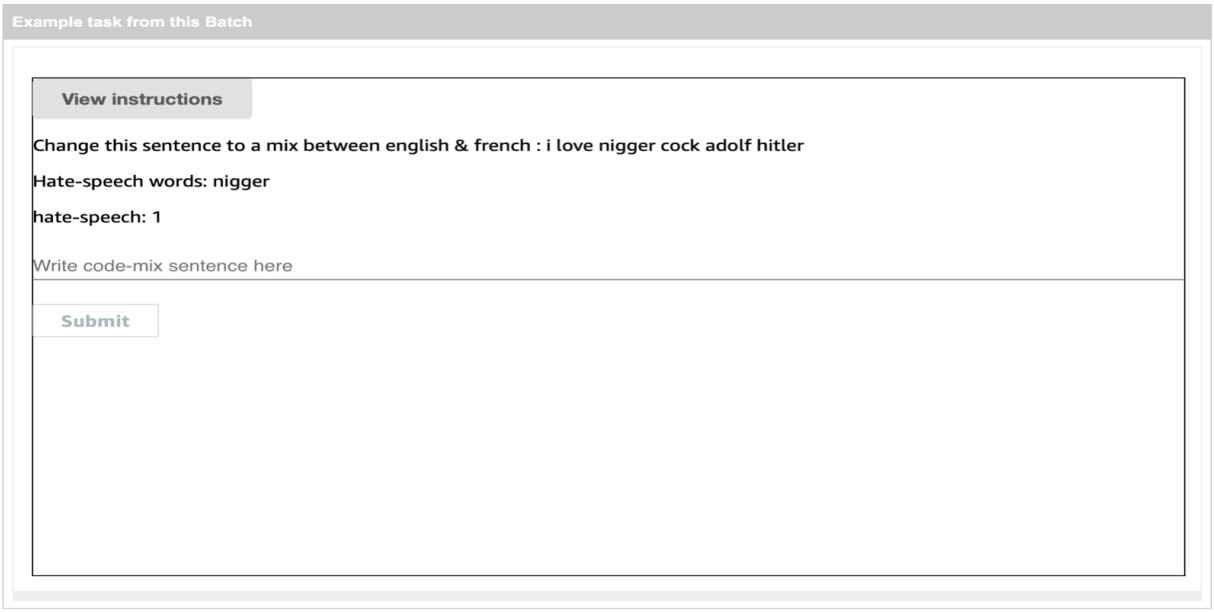

Figure 3: Interface for human code-switching annotation task for SWAP method

- **Average run-time for each approach**: Each epoch of the XLMR models, on average, takes 2 minutes for binary offensive classification. We train the model until learning rate reaches a very small value.
- **Number of parameters in the model**: We use XLMR in our in our experiments. This model has 2.7 million parameters to be optimized during training.
- **Explanation of evaluation metrics used**: To evaluate the performance of the model, we use the the weighted average and F1 scores for prediction.
- **Hyper-parameter configurations for best-performing models**: Our model has 768 hidden layers. The Adamw optimizer learning rate is set to 2e-5 and the batch size is 16.
- **The method of choosing hyper-parameter values and the criterion used to select among them**: Random search is used to determine the hyper-parameters. The selection is determined F1 scores and the selected hyper-params are used across experiments for uniformity.

A Survey on Multimodal Disinformation Detection

Firoj Alam[1], Stefano Cresci[2], Tanmoy Chakraborty[3*], Fabrizio Silvestri[4], Dimitar Dimitrov[5], Giovanni Da San Martino[6], Shaden Shaar[1], Hamed Firooz[7], Preslav Nakov[8]

[1]Qatar Computing Research Institute, HBKU, [2]IIT-CNR, Pisa, [3]IIT Delhi,

[4]Sapienza University of Rome, Italy, [5]Sofia University, [6]University of Padova, Italy, [7]Facebook AI,

[8]Mohamed bin Zayed University of Artificial Intelligence

{fialam, sshaar}@hbku.edu.qa, s.cresci@iit.cnr.it, tanchak@ee.iitd.ac.in, fsilvestri@diag.uniroma1.it

mitko.bg.ss@gmail.com, dasan@math.unipd.it,mhfirooz@fb.com, preslav.nakov@mbzuai.ac.ae

Abstract

Recent years have witnessed the proliferation of offensive content online such as fake news, propaganda, misinformation, and disinformation. While initially this was mostly about textual content, over time images and videos gained popularity, as they are much easier to consume, attract more attention, and spread further than text. As a result, researchers started leveraging different modalities and combinations thereof to tackle online multimodal offensive content. In this study, we offer a survey on the state-of-the-art on *multimodal disinformation detection* covering various combinations of modalities: text, images, speech, video, social media network structure, and temporal information. Moreover, while some studies focused on *factuality*, others investigated how *harmful* the content is. While these two components in the definition of disinformation – (*i*) factuality, and (*ii*) harmfulness –, are equally important, they are typically studied in isolation. Thus, we argue for the need to tackle disinformation detection by taking into account multiple modalities as well as both factuality and harmfulness, in the same framework. Finally, we discuss current challenges and future research directions.

1 Introduction

The proliferation of online social media has encouraged individuals to freely express their opinions and emotions. On one hand, the freedom of speech has led to a massive growth of online content which, if systematically mined, can be used for citizen journalism, public awareness, political campaigning, etc. On the other hand, its misuse has given rise to the proliferation of hostility online (Brooke, 2019; Joksimovic et al., 2019), resulting in offensive content in the form of fake news, hate speech (Schmidt and Wiegand, 2017a; Davidson et al., 2017), propaganda (Da San Martino et al., 2019), cyberbullying (Van Hee et al., 2015), etc. Indeed, researchers have argued that this situation has set the dawn of the Post-Truth Era, dominated by emotions and "alternative facts" (Lewandowsky et al., 2017; Cooke, 2018; Nakov and Da San Martino, 2020). More recently, with the emergence of the COVID-19 pandemic, a new blending of medical and political false information has given rise to the first global infodemic (Paka et al., 2021; Zarocostas, 2020; Patwa et al., 2021).[1]

The term "fake news" is commonly used, although it is very generic, and misleads people to focus only on veracity. That is why international organizations such as the UN, WHO, EU, and NATO prefer the term ***disinformation*** (Ireton and Posetti, 2018), which refers to information that is (*i*) *fake* and also (*ii*) spreads *deliberately to deceive and harm* others. The latter aspect of the disinformation (i.e., harmfulness) is often ignored, but it is equally important. A related term is *misinformation*, which also refers to the spreading of false content, but lacks the underlying intention to do harm. This is illustrated by the definitions of these notions by First Draft (Ireton and Posetti, 2018) where ***misinformation*** is defined as "*unintentional mistakes such as inaccurate photo captions, dates, statistics, translations, or when satire is taken seriously*", while ***disinformation*** is "*fabricated or deliberately manipulated text/speech/visual context, and also intentionally created conspiracy theories or rumors*".

In our survey, we will focus on disinformation, and we will study both the factuality and harmfulness aspects of the problem, with focus on different modalities. Note that there are posts that can be harmful but factually true or non-factual but harmful (e.g., hate speech); our study also covers some related work on them. The term *factuality* refers to automatically evaluating the solidity of the reporting/social media statements in terms of facts and

*Work done while T. Chakraborty was at IIIT-Delhi, India.

[1]https://www.who.int/health-topics/infodemic

Proceedings of the 29th International Conference on Computational Linguistics, pages 6625–6643
October 12–17, 2022.

figures (Ireton and Posetti, 2018). The *harmfulness* or *harmful content* typically refers to "*anything online which causes a person distress or harm*".[2] Figure 2, in Appendix, gives examples of such content. Alam et al. (2021) addressed both aspects of disinformation using social media content related to the COVID-19 infodemic. They demonstrated a correlation between factuality and harmfulness, which varies across languages even in the same country, e.g., for Arabic, 56% of the false content was harmful, while for English, it was 24%.

Disinformation often spreads as text. However, Internet and social media allow the use of different modalities, which can make a disinformation message attractive as well as impactful, e.g., a meme or a video is much easier to consume, attracts much more attention, is perceived as more credible (Hameleers et al., 2020), spreads further than simple text (Zannettou et al., 2018), and can be weaponized (Olsen, 2018).

Notably, multimodality remains under-explored in disinformation detection. Bozarth and Budak (2020) performed a meta-review of 23 fake news models and the data modality they leveraged, and found that 91.3% used text, 47.8% looked into social media network structure, 26% relied on temporal data, and only a handful made use of images or videos. Moreover, while there has been research to detect whether an image or a video has been manipulated, the attempt is less in a truly multimodal setting (Pérez-Rosas et al., 2015; Tan et al., 2020; Zhang et al., 2022; Song et al., 2021; Giachanou et al., 2020; Denaux and Gomez-Perez, 2020).

Here we survey research on multimodal disinformation detection covering various combinations of modalities: text, images, speech, video, social media network structure, and temporal information. The data sources include social media (e.g., Twitter), news, video (e.g., courtroom trials), and TV shows. We further argue for the need to cover multiple modalities in the same framework, while taking both factuality and harmfulness into account.

While there have been a number of surveys on "fake news" (Shu et al., 2017; Kumar and Shah, 2018; Cardoso Durier da Silva et al., 2019; Zhou and Zafarani, 2020), misinformation (Islam et al., 2020), fact-checking (Thorne and Vlachos, 2018; Kotonya and Toni, 2020a), truth discovery (Li et al., 2016), rumour detection (Bondielli and Marcelloni, 2019), harmful memes (Sharma et al., 2022) and propaganda detection (Da San Martino et al., 2020), none of them had multimodality as the main focus. Moreover, they targeted either factuality (most surveys above), or harmfulness (the latter survey), but not both. Here, we aim to bridge this gap. Therefore, in the present survey, we analyze the literature covering various aspects of multimodality (text, image, speech, video, network, and temporal), with a focus on the two aspects of disinformation: factuality and harmfulness, as shown in Figure 1.

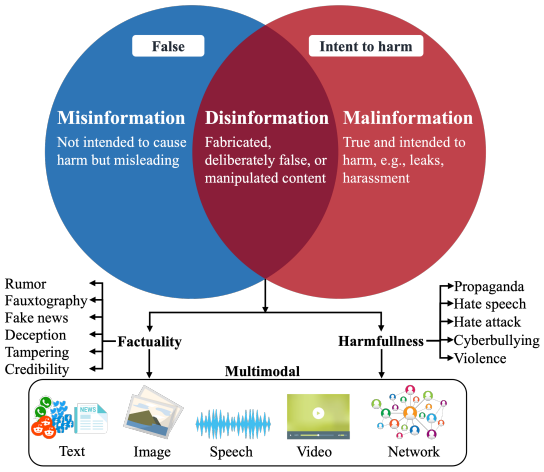

Figure 1: Our vision of multimodality to interact with harmfulness and factuality in this survey.

2 Multimodal Factuality Prediction

In this section, we focus on the first aspect of disinformation – *factuality*. Automatic detection of factual claims is important to debunk the spread of misleading information, as it is crucial to detect the factuality of statements that can mislead people. A large body of work has been devoted to fact-checking textual claims but such claims are often expressed and disseminated together with other modalities such as images, speech, and video, and are further propagated through social networks. We summarize relevant studies in Table 1.

2.1 Text

Due to the availability of large amounts of textual content, research on the text modality is comparatively richer than for other modalities. Notable work in this direction covers fake news spread on social media (Vosoughi et al., 2018b), fake news and fact-checking on news media (Rashkin et al., 2017), fact-checking such as fact-checked URL recommendation model (Vo and Lee, 2018) to reduce the spread, fact-checking with stance detection (Baly et al., 2018b), factuality of media outlets

[2] https://swgfl.org.uk/services/report-harmful-content/

Ref.	Task	Modality					Data/Source	Anno.	Lang	Method
		T	I	V	N	S				
(Baly et al., 2020)	Bias, factuality	✓				✓	MBFC	M	En	SVM, BERT
(Dinkov et al., 2019)	Bias	✓				✓	MBFC	M	En	MM deep learning architecture
(Baly et al., 2018a)	Bias, factuality	✓					MBFC	M	En	SVM
(Shao et al., 2018)	Credibility∗	✓					Articles and tweets	M	En	Statistical analysis
(Sen et al., 2020)	Deception			✓			CTD: 121 videos	M	En	RF, SVM and NN classifiers
(Soldner et al., 2019)	Deception			✓			TV Show	M	En	RF
(Volkova et al., 2019)	Deception	✓	✓				Twitter; T1: 2,485, T2-T3: 56,691, T4: 496,929	M	En	Feature fusion with AdaBoost/NN
(Krishnamurthy et al., 2018)	Deception			✓			CTD: 121 videos	M	En	MLP
(Kaya and Karpov, 2016)	Deception					✓	CSC: 25 videos	M	En	PLS/ELM based model
(Levitan et al., 2016)	Deception					✓	CSC: 25 videos	M	En	SMO, Bagging, Dagging, BN and NB
(Pérez-Rosas et al., 2015)	Deception			✓			CTD: 121 videos	M	En	DT and RF
(Hirschberg et al., 2005)	Deception					✓	CSC: 25 videos	M	En	Rule-based classifier
(Kazemi et al., 2021)	Faculity	✓					FEVER	M	En	Deep Q-learning network
(Atanasova et al., 2020)	Factuality	✓					Liar-Plus	M	En	DistilBERT
(Sathe et al., 2020)	Faculity	✓					WikiFactCheck	M	En	SVM, Decomposable attention model
(Shaar et al., 2020)	Faculity	✓					Political debates	M	En	Learning-to-rank approach, BM25, BERT, RoBERTa, sentence-BERT
(Kopev et al., 2019)	Faculity	✓		✓			Political debates	M	En	MM fusion: concatenation
(Vo and Lee, 2018)	Faculity	✓					Fact-checked tweets from: Snopes.com, Politifact.com, FactCheck.org, OpenSecrets.org, TruthOrfiction.com and Hoax-slayer.net	M	En	BPRMF , MF, CoFactor, CTR, proposed a joint model
(Baly et al., 2018b)	Faculity	✓					Claims from Verify and Reuters	M	Ar	Gradient boosting, multilayer perceptron, softmax layer, end-to-end memory network
(Rashkin et al., 2017)	Faculity	✓					Politifact	M	En	LSTM, MaxEnt, NB
(Nguyen et al., 2020)	Fake news	✓			✓		PHEME , Twitter (snopes.com), Weibo, FakeNewsNet	M	En	Graphical social context
(Nakamura et al., 2020)	Fake news	✓	✓				Reddit: 1m posts	DS	En	MM fusion
(Shu et al., 2020)	Fake news	✓			✓		PolitiFact and GossipCop	M	En	GNB, DT, LR, and RF
(Shu et al., 2019)	Fake news	✓			✓		BuzzFeed and PolitiFact	M	En	LR, NB, DT, XGBoost, AdaBoost, and GB
(Vosoughi et al., 2018a)	Fake news	✓			✓		Twitter: 126,000 posts	M	En	Statistical analysis, Topic modeling
(Liu and Wu, 2018)	Fake news	✓			✓		Weibo: 4,664 (Ma et al., 2016), Twitter15: 1,490 (Ma et al., 2017), Twitter16: 818 (Ma et al., 2017)	M	En	DT, SVM, GRU, RF, RNN, CNN
(Rashkin et al., 2017)	Fake news	✓					Gigaword corpus, articles from seven unreliable news sites	M	En	MaxEnt
(Boididou et al., 2016)	Fake			✓	✓		Social media	M	En	-
(Gupta et al., 2013)	Fake news				✓		Twitter: 16,117 tweets	M	En	DT on balanced dataset, NB
(Wang et al., 2021)	Fauxtography	✓	✓				Twitter, 4chan, and Reddit	M	En	Analytical
(Zhang et al., 2018)	Fauxtography	✓	✓				Reddit: 91, Twitter: 390	M	En	Feature fusion with XGBoost
(Heller et al., 2018)	Image tampering∗∗		✓				Reddit: 102,028 images	A	-	-
(Garimella and Eckles, 2020)	Misinformation∗∗		✓				WhatsApp: 2,500 images	M	-	-
(Zannettou et al., 2018)	Memes propagation	✓	✓				Twitter, Reddit, 4chan, and Gab	DS	-	Memes analysis
(Vosoughi et al., 2017)	Rumor	✓			✓		Twitter: 113 false and 96 true	M	En	Temporal, propagation linguistic, and user credibility features
(Kwon et al., 2017)	Rumor	✓			✓		Twitter, snopes.com, and urban-legends.about.com	M	En	RF

Table 1: Summary of the most relevant works on factuality, covering different modalities and tasks. **T:Text**, **I: Image**, **V:Video**, **N:Network**, **S:Speech**. CTD: Courtroom trial dataset, CSC: Columbia/SRI/Colorado Corpus. Anno.: Annotation, M: manual annotation; DS: distant supervision. MM: Multimodal, SVM: Support Vector Machine, RF: Random Forest, DT: Decision Tree; NN: Neural Network, MLP: Multi-layer Perceptron, PLS: Partial Least Squares regression; ELM: Extreme Learning Machines, NB: Naïve Bayes, BN: BayesNet, BPRMF: Bayesian Personalized Ranking Matrix Factorization, , MF: Matrix Factorization, CTR: Collaborative Filtering Regression, GNB: Gaussian Naive Bayes; LR: Logistic Regression; GB: Gradient Boosting, GRU: Gated Recurrent Units, RNN: Recurrent Neural Networks, CNN: Convolutional Neural Networks. ∗ Also include botometer features. T1-T4 represents different tasks. ∗∗ dataset only.

(Baly et al., 2020, 2018a), generating justifications for verdicts on claims (Atanasova et al., 2020), and fact-checking claims from Wikipedia (Sathe et al., 2020). There have also been recent efforts for fact-checking from political debates (Shaar et al., 2020, 2022, 2021; Nakov et al., 2022b,a), fact-checking with evidence reasoning (Si et al., 2021; Jiang et al., 2021; Wan et al., 2021) and fact-checking by claim matching (Kazemi et al., 2021). Given that there have been surveys on the text modality for fake news/disinformation detection and fact-checking, here we will not go into more detail about the individual studies.

2.2 Image

Text with visual content (e.g., images) in social media is more prominent as it is more intuitive; thus, it is easier to consume, it spreads faster, it gets 18% more clicks, 89% more likes, and 150% more retweets (Zhang et al., 2018). Due to the growing number of claims disseminated with images, in the current literature, there have been various studies that address the visual content with text for predicting misleading information (Volkova et al., 2019), fake images (Gupta et al., 2013), images shared with misinformation in political groups (Garimella and Eckles, 2020), and fauxtography (Zhang et al.,

2018; Wang et al., 2021). Some of these studies attempt to understand how two different modalities are used. Their analyses show that the extension of text with images increases the effectiveness of misleading content. Gupta et al. (2013) highlighted the role of Twitter to spread fake images. This study reports that 86% tweets spreading fake images are retweets. Garimella and Eckles (2020) manually annotated a sample of 2,500 images collected from public WhatsApp groups, and labeled them as *misinformation*, *not misinformation*, *misinformation already fact-checked*, and *unclear*; however, experiments were conducted with binary labels: misinformation vs. not-misinformation. The authors found that violent and graphic images spread faster. Nakamura et al. (2020) developed a multimodal dataset containing 1M posts including text, images, metadata, and comments collected from Reddit. The dataset was labeled with 2, 3, and 6-ways labels. Volkova et al. (2019) proposed models for detecting misleading information using images and text.

Fauxtography is defined as *"visual images, especially news photographs, which convey a questionable (or outright false) sense of the events they seem to depict"* (Cooper, 2007). It is also commonly used in social media in different forms such as a fake image with false claims, a true image with false claims. Zhang et al. (2018) defined that *"a post is a fauxtography if the image of the post (i) directly supports a false claim, or (ii) conveys misinformation of a true claim."* An example is shown in Figure 2 (in Appendix A). Zhang et al. (2018) developed FauxBuster to detect fauxtographic social media content, which uses social media comments in addition to the content in the images and the texts. Zlatkova et al. (2019) investigated the factuality of claims with respect to images and compared the performance of different feature groups between text and images. Wang et al. (2021) analyzed fauxtography images in social media posts and found that posts with doctored images increase user engagement in the form of re-shares, likes, and comments, specifically in Twitter and Reddit. They pointed out that doctored images are often used as memes to mislead or as a means of satire, and that they have a 'clickbait' power to drive engagement.

2.3 Speech/Audio

There have been attempts to use acoustic signals to predict the factuality of claims in political debates (Kopev et al., 2019; Shaar et al., 2020), left-center-right bias in YouTube channels (Dinkov et al., 2019), and deception in speech (Hirschberg et al., 2005). Kopev et al. (2019) found that the acoustic signal helps in improving the performance compared to using only textual and metadata features. Similarly, Dinkov et al. (2019) reported that the use of speech signal improves the performance of the system for detecting the political bias (i.e., left, center, right) of Youtube channels. Moreover, a large body of work was done on deception detection using the acoustic signal. Hirschberg et al. (2005) created the Columbia-SRI-Colorado (CSC) corpus by eliciting within-speaker deceptive and non-deceptive speech. Their experiments consist of the use of acoustic, prosodic, and a variety of lexical features including 68 LIWC categories, filled pauses, and paralinguistic information (e.g., speaker information, gender, field-pause). Using the same corpus, an evaluation campaign was organized, where different multimodal approaches were proposed, such as fusion of different acoustic, prosodic, lexical, and phonotactics representations (Levitan et al., 2016; Kaya and Karpov, 2016).

2.4 Video

In addition to textual, imagery, and speech content, the information in video plays an important role in capturing cues of deceptive behavior. Such cues in videos (e.g., facial expression, gestures) have been investigated in several studies for deception detection. Pérez-Rosas et al. (2015) developed a real-life courtroom trial dataset, which includes 61 deceptive and 60 truthful videos. They explored the use of n-gram features from transcripts and non-verbal features (i.e., facial expressions, eyebrows, eyes, mouth openness, mouth lips, and head movements, hand gestures) to classify liars and truth-tellers. Krishnamurthy et al. (2018) explored textual, speech, and visual features for deception detection. They used a 3D CNN to extract visual features from each frame, spatio-temporal features, and facial expressions such as smile, fear, or stress. Soldner et al. (2019) developed a multimodal deception dataset using TV shows and experimented with textual, visual and dialog features.

2.5 Network and Temporal Information

The rationale for leveraging network information stems from early work (Shao et al., 2018; Vosoughi et al., 2018a) that showed that propagation and interaction networks of fake news are deeper and

wider than those of real news. Vosoughi et al. (2018a) further found that fake information spreads faster than factual one, thus advocating for the use of temporal information.

Propagation networks can be homogeneous or heterogeneous (e.g., encompassing news articles, publishers, users, and posts) and they can be analyzed at different scales (e.g., node-level, ego-level, triad-level, community-level and the overall network, as shown in Figure 3, in Appendix) (Zhou and Zafarani, 2019). Shu et al. (2020) tackled the fake news classification task by proposing an approach based on hierarchical propagation networks. At both micro- and macro-scale, they extracted and jointly considered network features, temporal features, and linguistic features. Experiments on PolitiFact and GossipCop datasets revealed that temporal features have maximum contribution, followed by network and linguistic features. Shu et al. (2019) provided one of the most thorough multimodal frameworks for fake news classification. Their experimental results suggest that social context (i.e., network-derived) features are more informative than news content ones.

Vosoughi et al. (2017) proposed Rumor Gauge, a system that jointly exploits temporal and propagation features, in conjunction with linguistic and user credibility features, for checking the veracity of rumors. In particular, Rumor Gauge leverages text, and network propagation. The temporal modality does not directly provide features, but is instead considered by recomputing all other features at regular time steps, thus yielding multiple time series. Results by Vosoughi et al. (2017) and Kwon et al. (2017) also demonstrated that the contribution of the different data modalities change over time.

To mitigate the "cold start" problem of propagation-based early detection of fake news, Liu and Wu (2018) proposed an approach that is primarily based on user and temporal information. First, they built a propagation path of each news as a time series of user representations. The time series for a given news only contains the ordered representations of those users that shared such news. Then, they learned two vector representations of each propagation path via GRUs and CNNs, respectively. Zannettou et al. (2018) analyzed different aspects of memes, such as how they evolve and propagate in different mainstream and fringe web communities, and variants of memes that propagate. Finally, Nguyen et al. (2020) proposed Factual News Graph (FANG) to exploit the social structure and the engagement patterns of users for fake news detection.

3 Multimodal Harmful Content Detection

In this section, we focus on the second aspect of disinformation: *harmfulness*. It is essential to filter or to flag online harmful content. The harmful content includes *child abuse material*, *violent and extreme content*, *hate speech*, *graphic content*, *sexual content*, and *spam content* (Banko et al., 2020).[3] In recent years, the ability to recognize harmful content within online communities has received a lot of attention by researchers (Pramanick et al., 2021a,b) and policymakers that aim to keep users safe in the digital world. Studies in this direction include detecting harmful contents in network science (Ribeiro et al., 2018), natural language processing (Waseem et al., 2017; Schmidt and Wiegand, 2017b; Fortuna and Nunes, 2018) and computer vision (Yang et al., 2019a; Vijayaraghavan et al., 2021; Gomez et al., 2020; Dimitrov et al., 2021b). In Table 2, we provide a list of relevant work addressing different types of harmful content, modalities, source of data, annotation approach, language of the content and the methods.

3.1 Text

In the past few years there has been significant research effort on detecting harmful content (e.g., hate speech) from social media posts (Van Hee et al., 2015; Waseem and Hovy, 2016; Waseem et al., 2017; Schmidt and Wiegand, 2017b). Waseem and Hovy (2016) developed a dataset of hate speech consisting of 16K tweets, and reported a baseline results using char- and word- ngrams and a logistic regression classifier. (Davidson et al., 2017) distinguished between hate speech, and offensive language. They developed a dataset of ∼24K labeled tweets with categories such as hate speech, offensive language and neither. Qian et al. (2018) took a different approach to classic hate speech classification. Instead of binary classes, they proposed 13 fine-grained hate categories such as nationalist, anti-immigrant, racist skinhead, among others, providing a dataset of tweets collected from 40 hate groups. Ribeiro et al. (2018) proposed an approach to find hateful users on Twitter. Mathew et al. (2019) analyzed 341K

[3] https://swgfl.org.uk/services/report-harmful-content/

Ref	Task	Modality					Data/Source	Anno.	Lang	Method
		T	I	V	N	S				
(Nizzoli et al., 2021)	CIB				✓		Twitter: 1.1m users, 11m tweets	DS	En	Statistical and similarity analysis
Weber and Neumann (2020)	CIB				✓		Twitter	–	En	Statistical and network analysis
(Wang et al., 2020)	Cyberbullying	✓	✓				Posts: Vine (970), Instagram (2,218)	M	En	SVM, NB, LR, RF, LSTM, CNN
(Soni and Singh, 2018)	Cyberbullying	✓	✓			✓	Vine videos	M	En	KNN, SVM, LR, RF, GNB
(Dadvar and Eckert, 2018)	Cyberbullying	✓					Youtube 54k posts	M	En	LSTM, BiLSTM, CNN
Hosseinmardi et al. (2015)	Cyberbullying	✓	✓		✓		Instagram	M	En	SVM
(Rafiq et al., 2015)	Cyberbullying	✓			✓		Vine videos	M	En	NB, AdaBoost, DT and RF
(Van Hee et al., 2015)	Cyberbulling	✓					Ask.fm: 85k QA pairs	M	Nl	SVM
(Chatzakou et al., 2019)	Cyberbullying, Cyberaggression	✓			✓		Twitter: 1,303 users, 9,484 tweets	M	En	NB, RF, AdaBoost, Ensemble, NN
(Liang et al., 2017)	Gunshots					✓	Videos: freesound.com, Youtube; Test: CSV, TRECVID Gunshot, UrbanSound Gunshot	DS	En	Localized self-paced reranking
(Mariconti et al., 2019)	Hate attacks	✓	✓			✓	Youtube videos (428)	M	En	Ensemble, CNN, RNN
(Kiela et al., 2020)	Hate speech	✓	✓				FB: Hateful Memes Challenge	M	En	Late fusion, Concat BERT, MMBT, ViLBERT, VisualBERT
(Das et al., 2020)	Hate speech	✓	✓				FB: Hateful Memes Challenge	M	En	VisualBERT, MM fusion
(Gomez et al., 2020)	Hate speech	✓	✓				Twitter: MMHS150K	M	En	Inception v3, LSTM, and MM fusion
(Yang et al., 2019a)	Hate speech	✓	✓				FB: train+dev 378k, test 53k	M	En	Fusion: text + image embedding
(Waseem and Hovy, 2016)	Hate speech	✓					Twitter: 16,914 tweets	M	En	LR
(Davidson et al., 2017)	Hate speech	✓					Twitter: 24,802 tweets	M	En	LR, SVM, NB, DT, RF
(Qian et al., 2018)	Hate speech	✓					Twitter: 40 accounts, 3.5m tweets	DS	En	LR, SVM, Char-CNN, BiLSTM, HCVAE
(Ribeiro et al., 2018)	Hate speech	✓			✓		Twitter: 4,972 users	M	En	GradBoost, AdaBoost, GraphSage
(Mathew et al., 2019)	Hate speech	✓			✓		Gab: 21m posts by 341k users	DS	En	Lexicon based filtering, DeGroot
Dimitrov et al. (2021b)	Propaganda	✓	✓				FB: SemEval-2021 task 6: 950 Facebook memes	M	En	MM fusion, MM joint representation
(Vijayaraghavan et al., 2021)	Hate speech	✓			✓		In-house developed and curated datasets	M	En	MM late fusion, LR, SVM, CNN, BiGRU, BiLSTM
(Constantin et al., 2020)	Violence		✓	✓		✓	VSD96: Hollywood, Youtube	M	En	MM Early fusion; SVM, HMM, GMM, Bayesian, MLP, QDA, PLDA, CNN, KNN, unsupervised, hybrid
(Acar et al., 2013)	Violence			✓		✓	MediaEval VSD	M	En	SVM (mid-level audio + low-level visual)
(Giannakopoulos, 2009)	Violence					✓	Movies	M	-	BN, kNN

Table 2: Summary of the most relevant works on harmful content. **T:Text**, **I: Image**, **V:Video**, **N:Network**, **S:Speech**, Anno.: Annotation, CIB: Coordinated Inauthentic Behavior, QA: Question-answer, CSV: Real-life Conflict Scene Videos, VSD: Violent Scene Detection. Nl: Dutch. KNN: k-Nearest Neighbors, LSTM: Long Short-Term Memory, BiLSTM: Bidirectional LSTM, MMBT: MultiModal BiTransformers, HCVAE: Hierarchical Conditional Variational Autoencoder, QDA: Quadratic Discriminant Analysis, PLDA: Probabilistic Linear Discriminant Analysis.

users and 21M posts collected from Gab to understand the diffusion dynamics of hateful content. Their findings suggest that the posts from hateful user diffuse faster, wider, and have a greater outreach compared to the posts from non-hateful ones.

3.2 Image

Among different types of harmful content, cyberbullying is one of the major growing problems, significantly affecting teens. Hosseinmardi et al. (2015) investigated Instagram images and their associated comments for detecting cyberbullying and online harassment. They developed a manually labeled dataset using CrowdFlower (which is now Appen), where they followed standard procedures for the annotation: using annotation guidelines, qualification tests, gold standard evaluation and quality control criteria such as minimum annotation time. The annotated dataset consists of 998 media sessions (images and their associated comments). A key finding of this study is that a large fraction of the annotated posts (48%) with a high percentage of negative words have not been labeled as cyberbullying. To train and to evaluate the model, the authors used n-grams from text, meta-data (e.g., the number of followers, followees, likes, and shared media), and image categories as features and experimented with Naïve Bayes and SVM classifiers. Their study suggests that combining multiple modalities helps to improve the performance of the SVM classifier.

Hate speech is another important problem that spreads over social media. The "Hateful Memes Challenge" is an important milestone to advance the research on this topic and the tasks is to detect hateful memes (Kiela et al., 2020). Das et al. (2020) proposed different approaches for hatefulness detection in memes such as (*i*) extract the caption and include this information with the multimodal model, (*ii*) use sentiment as an additional feature with multimodal representations. For hate speech detection, Yang et al. (2019a) explored different fusion techniques such as concatenation, bilinear, gated summation, and attention, and reported that combining the text with image embedding boosted the performance in all cases. Vijayaraghavan et al. (2021) proposed methods for interpreting multimodal hate speech detection models, where the modalities consist of text and socio-cultural information rather than images. Concurrently, Gomez et al. (2020) introduced a larger dataset of 150K

tweets for multimodal hate speech detection, consisting of six labels.

Propaganda is another topic that has been explored in multimodal settings. Seo (2014) showed how Twitter was used as a propaganda tool during the 2012 Gaza conflict to build international support for each side of the conflict. Dimitrov et al. (2021b) addressed the detection of persuasion techniques in memes. Their analysis of the dataset showed that while propaganda is not always factually false or harmful, most memes are used to damage the reputation of a person or a group of people. Dimitrov et al. (2021a) highlighted the importance of both modalities for detecting fine-grained propaganda techniques, with VisualBERT yielding 19% improvement compared to using the image modality only (with ResNet-152), and 11% improvement compared to using the text modality only (with BERT). Similar observations were made by (Kiela et al., 2020) for hateful meme detection. Glenski et al. (2019) explored multilingual multimodal content and categorizes disinformation, propaganda, conspiracy, hoax, and clickbait.

3.3 Speech/Audio

Cues in spoken content can represent harmful behaviors and those cues can be used to automatically detect such content. Due to the lack of data, studies using the speech-only modality are comparatively lower than other modalities even though it plays a major role in many contexts. For example, for detecting violent content such as screaming and gunshots, the speech modality can play an important role, which other modalities might not be able to offer. This is important as most often user-generated contents are posted on newspapers or their social media accounts without verifying the content of the post, which can have serious consequences (Harkin et al., 2012; Rauchfleisch et al., 2017).

Giannakopoulos (2009) studied the audio segmentation approaches for segmenting violent (e.g., gunshots, screams) and non-violent (e.g., music, speech) content in movies. The studies related to violent content detection using acoustic features also include (Acar et al., 2013), where the focus was on finding violent content in movies.

Liang et al. (2017) proposed Localized Self-Paced Reranking (LSPaR) for detecting gunshots and explosion in videos using acoustic features. Soni and Singh (2018) investigated audio, visual and textual features for cyberbullying detection. Their findings suggest that audio and visual features are associated with the occurrence of cyberbullying, and both these features complement textual features.

3.4 Video

There are multiple studies on detecting cyberbullying in video-based social networks such as Vine (Rafiq et al., 2015) and YouTube (Dadvar and Eckert, 2018). These studies show that although the percentage of cyberbullying in video sessions is quite low, automatic detection of these types of content is very challenging. Wang et al. (2020) used textual, visual, and other meta-information to detect social media posts with bullying topics. Their proposed method was evaluated on publicly available multimodal cyberbullying datasets. Abd Kadir et al. (2016) investigated the relationship between emotion and propaganda techniques in Youtube videos. Their findings suggest that propaganda techniques in Youtube videos affect emotional responses. Content (e.g., Youtube videos) can also be attacked by hateful users via posting hateful comments through a coordinated effort. Mariconti et al. (2019) investigated whether a video is likely to be attacked using different modalities such as metadata, audio transcripts, and thumbnails.

There has been a recent interest from different government agencies to stop the spread of violent content. Constantin et al. (2020) developed a multimodal dataset, which consists of more than 96 hours of Hollywood and YouTube videos and high variability of content. Their study suggests that multimodal approaches with audio and images perform better.

3.5 Network and Temporal Information

The use of network data for predicting factuality was motivated by results showing different propagation patterns for fake vs. real content. Such results are lacking for harmful content. However, the intention to harm in social media is often pursued via coordinated actions, for instance, by groups of users (e.g., social bots and trolls (Cresci, 2020)) that target certain people or minorities. These collaborative harmful actions, perpetrated to increase the efficacy of the harm, are best addressed using network analysis to detect likely coordinated harmful campaigns. Chatzakou et al. (2019) focused on detecting cyberbullying and cyberaggression by training machine learning models for detecting: (*i*) bullies, (*ii*) aggressors, (*iii*) spammers, and

(*iv*) normal users on Twitter. To solve these tasks, they leveraged a combination of 38 features extracted from user profiles, the textual content of their posts, and network information (e.g., user degree and centrality measures in the social graph). Orthogonal and in synergy with respect to the detection of disinformation, scholars have recently focused on the novel task of detecting Coordinated Inauthentic Behavior (CIB) (Nizzoli et al., 2021). *CIB is defined as coordinated activities that aim to mislead and manipulate others.*[4] Detecting CIB typically involves analyzing both interaction networks to detect suspicious coordination, as well as the coordinated users and the content they shared to detect inauthentic users and harmful content (Nizzoli et al., 2021, 2020; Pacheco et al., 2021). Given the importance of coordination in CIB, the analysis typically starts from the available network data by applying community detection algorithms, and subsequently moving to the analysis of textual data.

4 Modeling Techniques

In this section, we discuss modeling techniques for both factuality and harmfulness. To combine multiple modalities, there have been several approaches: (i) *early-fusion*, where low-level features from different modalities are learned, fused, and fed into a single prediction model (Jin et al., 2017b; Yang et al., 2018; Zhang et al., 2019; Singhal et al., 2019; Zhou et al., 2020; Kang et al., 2020); (ii) *late-fusion*, where unimodal decisions are fused with some mechanisms such as averaging and voting (Agrawal et al., 2017; Qi et al., 2019), and (iii) *hybrid-fusion*, where a subset of learned features are passed to the final classifier (early-fusion), and the remaining modalities are fed to the classifier later (late-fusion) (Jin et al., 2017a). Within these fusion strategies, the learning setup can also be divided into *unsupervised*, *semi-supervised*, *supervised* and *self-supervised* methods.

Dimitrov et al. (2021b) investigated different fusion strategies (e.g., *early-* and *late*-fusion and *self-supervised* models) for propaganda detection using VisualBERT (Li et al., 2019), MMBT (Kiela et al., 2019), and ViLBERT (Lu et al., 2019). Their findings suggest that self-supervised joint learning models, such as MMBT, ViLBERT, and VisualBERT perform better in increasing order, respectively, compared to the other fusion methods. As a part of "Hateful Memes Challenge" to classify hateful vs. non-hateful memes, several such models have been investigated by Kiela et al. (2020).

Attempts to design *unsupervised* models are limited. Müller-Budack et al. (2020) introduced Cross-modal Consistency Verification Tool (CCVT) to check the coherence between images and associated texts. Yang et al. (2019b) defined trust of news and credibility of users who spread the news and used Bayesian learning to iteratively update these quantities. News with low trustworthiness is returned as fake news. Gangireddy et al. (2020) proposed GTUT, a graph-based approach that exploits the underlying bipartite network of users and news articles to detect the dense communities of fake news and fraud users.

Due to the scarcity of labeled data, a few studies attempted to design *semi-supervised* methods by leveraging an ample amount of unlabelled data. Helmstetter and Paulheim (2018); Gravanis et al. (2019) presented weak-supervision and Guacho et al. (2018) presented a tensor-decomposition semi-supervised method for fake content detection. Dong et al. (2020) developed a deep semi-supervised model via two-path learning (one path uses a limited labeled data, the other path explores the unlabelled data) for timely fake news detection. Paka et al. (2021) presented, Cross-SEAN, a cross-stitch semi-supervised end-to-end neural attention model for COVID-19 fake news detection.

Within a *supervised* learning setup, two other types of learning method have also been explored for disinformation detection such as *adversarial learning* and *autoencoder based*. *Adversarial learning* models for fake news detection include EANN (Wang et al., 2018), an event adversarial neural network to detect emerging and time-critical fake news, and SAME (Cui et al., 2019), a sentiment-aware multimodal embedding method which leverages multiple modalities with the sentiment expressed by readers in their comments.

5 Major Challenges

Recently, several initiatives were undertaken by major companies and government entities to combat disinformation in social media (DIGI, 2021).[5] However, automatic detection of misleading and harmful content poses a number of challenges as discussed below and in Appendix (Section D).

[4] https://medium.com/1st-draft/how-to-improve-our-analysis-of-coordinated-inauthentic-behavior-a4ec62ce9bff

[5] For example, http://digi.org.au/disinformation-code/

Models Combining Multiple Modalities. The major challenge is to devise a mechanism to combine multiple modalities in a systematic way so that one modality complements the others. Current state-of-the-art primarily adopts early and late fusion, which are limited and do not always yield strong results (Dimitrov et al., 2021a). Very recently, jointly trained multimodal transformer-based models (e.g., ViLBERT (Lu et al., 2019), Visual BERT (Lin et al., 2014) and Multimodal Bitransformers (MMBT) (Kiela et al., 2019)) have shown strong potential (Dimitrov et al., 2021b,a; Kiela et al., 2020). However, such models are trained considering only two modalities (textual and visual), while fact-checking or disinformation-related content consists of more than two modalities e.g., text, speech, video, network, etc. (Baly et al., 2020). Hence, there is a room for improvement in developing multimodal models that involve additional, and potentially more than two modalities. Another important problem is cross-modal inconsistency in social media content, as shown in Figure 2(c), which poses a challenge in a multimodal setting (Tan et al., 2020).

Datasets. One of the major challenges when working with such diverse modalities, i.e., text, image, speech, video, and network, is to get access to an appropriate dataset, and moreover to one that considers both factuality and harmfulness. Furthermore, there is a need to integrate data from multiple platforms (e.g., news, posts from Twitter, Reddit and Instagram) as different data sources present different styles and focus on different topics.

6 Future Directions

Based on the aforementioned challenges, we forecast the following research directions:

Explainability. Model interpretation remains largely unexplored. This can be addressed in future studies to understand the general capability of the models. Providing evidence of why certain claims are false is also important. There has been work in this direction such as TabFact (Chen et al., 2020) and FEVER (Hanselowski et al., 2018). However, such approaches rely on existing knowledge bases (e.g., Wikipedia) and may fail for a new problem such as disinformation about COVID-19. It is also important to understand what models learn, e.g., lexical or semantic concepts or a set of neurons may learn one aspect better than the others. Moreover, while current studies on explainable fact-checking focus on explaining the predictions, very few focus on model explanations (Kotonya and Toni, 2020b).

Beyond Content and Network Signals. State-of-the-art methods for multimodal factuality prediction and harmful content detection are primarily based on content signals and network structure. However, the information in these signals is limited and does not include personal preferences or cultural aspects. In the future, we envision multimodal techniques for disinformation detection that would go beyond content and network and would include signals like common sense and information about the user. Moreover, multimodal models will become larger with more heterogeneous signals as input, and they would be pre-trained on a wider variety of tasks to shelter both aspects of disinformation: factuality and harmfulness.

Knowledge-based Method. The use of knowledge-based approaches to check the factuality of claims based on what has been checked before could be ideal solutions as some claims are often repeated by politicians. Current approaches in this direction are limited and this can be explored further by creating a common repository of previously fact-checked claims and harmful content. Relevant studies in this direction include detecting previously fact-checked claims (Shaar et al., 2020), studying the role of context at the sentence level (Shaar et al., 2022) or at the document level (Shaar et al., 2021), and claim matching across languages (Kazemi et al., 2021).

7 Conclusion

We surveyed the state-of-the-art in multimodal disinformation detection based on prior work on different modalities, focusing on disinformation, i.e., information that is both false and intents to do harm. We covered the major research topics of factuality and disinformation. Our survey brought several interesting research challenges for multimodal disinformation detection, such as combining various modalities, which are often not aligned and are in different representations (e.g., text vs. speech vs. network structure), and the lack of such datasets to foster future research. In addition to highlighting the challenges, we also pointed to several research directions. While doing so, we argued for the need to tackle disinformation detection by taking into account multiple modalities as well as both factuality and harmfulness in the same framework.

Acknowledgments

This research is part of the Tanbih mega-project, developed at the Qatar Computing Research Institute, HBKU, which aims to limit the impact of "fake news," propaganda, and media bias by making users aware of what they are reading, thus promoting media literacy and critical thinking.

Limitations

We might not have covered *all* relevant work that falls under the topics of factuality and harmfulness across different modalities.

References

Shamsiah Abd Kadir, Anitawati Lokman, and T. Tsuchiya. 2016. Emotion and techniques of propaganda in youtube videos. *Indian journal of science and technology*, Vol (9):1–8.

Esra Acar, Frank Hopfgartner, and Sahin Albayrak. 2013. Violence detection in hollywood movies by the fusion of visual and mid-level audio cues. In *Proceedings of the 21st ACM International Conference on Multimedia*, MM '13, page 717–720, New York, NY, USA. Association for Computing Machinery.

Taruna Agrawal, Rahul Gupta, and Shrikanth Narayanan. 2017. Multimodal detection of fake social media use through a fusion of classification and pairwise ranking systems. In *25th European Signal Processing Conference (EUSIPCO)*, pages 1045–1049. IEEE.

Firoj Alam, Shaden Shaar, Fahim Dalvi, Hassan Sajjad, Alex Nikolov, Hamdy Mubarak, Giovanni Da San Martino, Ahmed Abdelali, Nadir Durrani, Kareem Darwish, Abdulaziz Al-Homaid, Wajdi Zaghouani, Tommaso Caselli, Gijs Danoe, Friso Stolk, Britt Bruntink, and Preslav Nakov. 2021. Fighting the COVID-19 infodemic: Modeling the perspective of journalists, fact-checkers, social media platforms, policy makers, and the society. In *Findings of EMNLP 2021*, EMNLP '21. Association for Computational Linguistics.

Pepa Atanasova, Jakob Grue Simonsen, Christina Lioma, and Isabelle Augenstein. 2020. Generating fact checking explanations. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, ACL '20, pages 7352–7364, Online. Association for Computational Linguistics.

Ramy Baly, Georgi Karadzhov, Dimitar Alexandrov, James Glass, and Preslav Nakov. 2018a. Predicting factuality of reporting and bias of news media sources. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, EMNLP '18, pages 3528–3539, Brussels, Belgium. Association for Computational Linguistics.

Ramy Baly, Georgi Karadzhov, Jisun An, Haewoon Kwak, Yoan Dinkov, Ahmed Ali, James Glass, and Preslav Nakov. 2020. What was written vs. who read it: News media profiling using text analysis and social media context. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, ACL '20, pages 3364–3374, Online. Association for Computational Linguistics.

Ramy Baly, Mitra Mohtarami, James Glass, Lluís Màrquez, Alessandro Moschitti, and Preslav Nakov. 2018b. Integrating stance detection and fact checking in a unified corpus. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 2 (Short Papers)*, ACL '18, pages 21–27, New Orleans, Louisiana. Association for Computational Linguistics.

Michele Banko, Brendon MacKeen, and Laurie Ray. 2020. A unified taxonomy of harmful content. In *Proceedings of the Fourth Workshop on Online Abuse and Harms*, pages 125–137, Online. Association for Computational Linguistics.

Christina Boididou, Symeon Papadopoulos, Stuart E. Middleton, Giulia Boato, Duc-Tien Dang-Nguyen, and Michael Riegler. 2016. Verifying multimedia use at MediaEval 2016. *MediaEval*.

Alessandro Bondielli and Francesco Marcelloni. 2019. A survey on fake news and rumour detection techniques. *Information Sciences*, 497:38–55.

Lia Bozarth and Ceren Budak. 2020. Toward a better performance evaluation framework for fake news classification. In *Proceedings of the international AAAI conference on web and social media*, volume 14 of *AAAI '20*, pages 60–71.

Sian Brooke. 2019. "condescending, rude, assholes": Framing gender and hostility on Stack Overflow. In *Proceedings of the Third Workshop on Abusive Language Online*, pages 172–180, Florence, Italy. Association for Computational Linguistics.

Fernando Cardoso Durier da Silva, Rafael Vieira, and Ana Cristina Garcia. 2019. Can machines learn to detect fake news? A survey focused on social media. In *Proceedings of the 52nd Hawaii International Conference on System Sciences*, volume 6, pages 2763–2770.

Despoina Chatzakou, Ilias Leontiadis, Jeremy Blackburn, Emiliano De Cristofaro, Gianluca Stringhini, Athena Vakali, and Nicolas Kourtellis. 2019. Detecting cyberbullying and cyberaggression in social media. *ACM Transactions on the Web (TWEB)*, 13(3):1–51.

Jing Chen, Chenhui Wang, Kejun Wang, Chaoqun Yin, Cong Zhao, Tao Xu, Xinyi Zhang, Ziqiang Huang, Meichen Liu, and Tao Yang. 2021. Heu emotion: a large-scale database for multimodal emotion recognition in the wild. *Neural Computing and Applications*, pages 1–17.

Wenhu Chen, Hongmin Wang, Jianshu Chen, Yunkai Zhang, Hong Wang, Shiyang Li, Xiyou Zhou, and William Yang Wang. 2020. Tabfact: A large-scale dataset for table-based fact verification. In *International Conference on Learning Representations*.

M. G. Constantin, L. D. Stefan, B. Ionescu, C. Demarty, M. Sjoberg, M. Schedl, and G. Gravier. 2020. Affect in multimedia: Benchmarking violent scenes detection. *IEEE Transactions on Affective Computing*, pages 1–1.

Nicole A Cooke. 2018. *Fake news and alternative facts: Information literacy in a post-truth era*. American Library Association.

Stephen D Cooper. 2007. A concise history of the fauxtography blogstorm in the 2006 lebanon war. *American Communication Journal*, 9(2).

Stefano Cresci. 2020. A decade of social bot detection. *Communications of the ACM*, 63(10):72–83.

Limeng Cui, Suhang Wang, and Dongwon Lee. 2019. SAME: sentiment-aware multi-modal embedding for detecting fake news. In *International Conference on Advances in Social Networks Analysis and Mining*, pages 41–48. ACM.

Giovanni Da San Martino, Stefano Cresci, Alberto Barrón-Cedeño, Seunghak Yu, Roberto Di Pietro, and Preslav Nakov. 2020. A survey on computational propaganda detection. In *Proceedings of the Twenty-Ninth International Joint Conference on Artificial Intelligence, IJCAI 2020*, pages 4826–4832.

Giovanni Da San Martino, Seunghak Yu, Alberto Barrón-Cedeño, Rostislav Petrov, and Preslav Nakov. 2019. Fine-grained analysis of propaganda in news article. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, EMNLP-IJCNLP '19, pages 5636–5646, Hong Kong, China. Association for Computational Linguistics.

Maral Dadvar and Kai Eckert. 2018. Cyberbullying detection in social networks using deep learning based models; a reproducibility study. *arXiv:1812.08046*.

Abhishek Das, Japsimar Singh Wahi, and Siyao Li. 2020. Detecting hate speech in multi-modal memes. *arXiv:2012.14891*.

Thomas Davidson, Dana Warmsley, Michael Macy, and Ingmar Weber. 2017. Automated hate speech detection and the problem of offensive language. In *Proceedings of the International AAAI Conference on Web and Social Media*, volume 11 of *AAAI '17*.

Ronald Denaux and Jose Manuel Gomez-Perez. 2020. Linked credibility reviews for explainable misinformation detection. In *International Semantic Web Conference*, pages 147–163. Springer.

Digital Industry Group Inc DIGI. 2021. Australian code of practice on disinformation and misinformation. Online.

Dimitar Dimitrov, Bishr Bin Ali, Shaden Shaar, Firoj Alam, Fabrizio Silvestri, Hamed Firooz, Preslav Nakov, and Giovanni Da San Martino. 2021a. Detecting propaganda techniques in memes. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing*, ACL-IJCNLP '21, pages 6603–6617, Online. Association for Computational Linguistics.

Dimitar Dimitrov, Bishr Bin Ali, Shaden Shaar, Firoj Alam, Fabrizio Silvestri, Hamed Firooz, Preslav Nakov, and Giovanni Da San Martino. 2021b. Task 6 at SemEval-2021: Detection of persuasion techniques in texts and images. In *Proceedings of the 15th International Workshop on Semantic Evaluation*, SemEval '21, Bangkok, Thailand. Association for Computational Linguistics.

Yoan Dinkov, Ahmed Ali, Ivan Koychev, and Preslav Nakov. 2019. Predicting the leading political ideology of youtube channels using acoustic, textual, and metadata information. In *Interspeech*, pages 501–505.

Xishuang Dong, Uboho Victor, and Lijun Qian. 2020. Two-path deep semisupervised learning for timely fake news detection. *IEEE Transactions on Computational Social Systems*.

Paula Fortuna and Sérgio Nunes. 2018. A survey on automatic detection of hate speech in text. *ACM Computing Surveys (CSUR)*, 51(4).

Paula Fortuna, Juan Soler-Company, and Leo Wanner. 2021. How well do hate speech, toxicity, abusive and offensive language classification models generalize across datasets? *Information Processing & Management*, 58(3):102524.

Siva Charan Reddy Gangireddy, Cheng Long, and Tanmoy Chakraborty. 2020. Unsupervised fake news detection: A graph-based approach. In *Proceedings of the 31st ACM Conference on Hypertext and Social Media*, pages 75–83.

Kiran Garimella and Dean Eckles. 2020. Images and misinformation in political groups: Evidence from whatsapp in india. *Harvard Kennedy School Misinformation Review*.

Anastasia Giachanou, Guobiao Zhang, and Paolo Rosso. 2020. Multimodal multi-image fake news detection. In *2020 IEEE 7th International Conference on Data Science and Advanced Analytics (DSAA)*, pages 647–654. IEEE.

Theodoros Giannakopoulos. 2009. Study and application of acoustic information for the detection of harmful content, and fusion with visual information. *University of Athens*.

Maria Glenski, Ellyn Ayton, Josh Mendoza, and Svitlana Volkova. 2019. Multilingual multimodal digital deception detection and disinformation spread across social platforms. *arXiv:1909.05838*.

Raul Gomez, Jaume Gibert, Lluis Gomez, and Dimosthenis Karatzas. 2020. Exploring hate speech detection in multimodal publications. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, pages 1470–1478.

Georgios Gravanis, Athena Vakali, Konstantinos Diamantaras, and Panagiotis Karadais. 2019. Behind the cues: A benchmarking study for fake news detection. *Expert Systems with Applications*, 128:201–213.

Gisel Bastidas Guacho, Sara Abdali, Neil Shah, and Evangelos E. Papalexakis. 2018. Semi-supervised content-based detection of misinformation via tensor embeddings. In *IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining*, pages 322–325. IEEE Computer Society.

Aditi Gupta, Hemank Lamba, Ponnurangam Kumaraguru, and Anupam Joshi. 2013. Faking Sandy: Characterizing and identifying fake images on Twitter during hurricane Sandy. In *Proceedings of the 22nd international conference on World Wide Web*, pages 729–736.

Michael Hameleers, Thomas E. Powell, Toni G.L.A. Van Der Meer, and Lieke Bos. 2020. A picture paints a thousand lies? the effects and mechanisms of multimodal disinformation and rebuttals disseminated via social media. *Political Communication*, 37(2):281–301.

Andreas Hanselowski, Hao Zhang, Zile Li, Daniil Sorokin, Benjamin Schiller, Claudia Schulz, and Iryna Gurevych. 2018. UKP-athene: Multi-sentence textual entailment for claim verification. In *Proceedings of the First Workshop on Fact Extraction and VERification (FEVER)*, pages 103–108, Brussels, Belgium. Association for Computational Linguistics.

Juliette Harkin, Kevin Anderson, Libby Morgan, and Briar Smith. 2012. Deciphering user-generated content in transitional societies. *Philadelphia, PA: Center for Global Communication Studies, Annenberg School for Communication, University of Pennsylvania*.

Silvan Heller, Luca Rossetto, and Heiko Schuldt. 2018. The ps-battles dataset-an image collection for image manipulation detection. *arXiv preprint arXiv:1804.04866*.

Stefan Helmstetter and Heiko Paulheim. 2018. Weakly supervised learning for fake news detection on twitter. In *IEEE/ACM 2018 International Conference on Advances in Social Networks Analysis and Mining, ASONAM 2018, Barcelona, Spain, August 28-31, 2018*, pages 274–277. IEEE Computer Society.

Julia Hirschberg, Stefan Benus, Jason Brenier, Frank Enos, Sarah Hoffman, Sarah Gilman, Cynthia Girand, Martin Graciarena, Andreas Kathol, Laura Michaelis, Bryan L. Pellom, Elizabeth Shriberg, and Andreas Stolcke. 2005. Distinguishing deceptive from non-deceptive speech. In *Interspeech*, pages 1833–1836.

Homa Hosseinmardi, Sabrina Arredondo Mattson, Rahat Ibn Rafiq, Richard Han, Qin Lv, and Shivakant Mishra. 2015. Detection of cyberbullying incidents on the Instagram social network. *arXiv:1503.03909*.

Cherilyn Ireton and Julie Posetti. 2018. *Journalism, fake news & disinformation: handbook for journalism education and training*. Unesco Publishing.

Md Rafiqul Islam, Shaowu Liu, Xianzhi Wang, and Guandong Xu. 2020. Deep learning for misinformation detection on online social networks: a survey and new perspectives. *SNAM*, 10(1):1–20.

Kelvin Jiang, Ronak Pradeep, and Jimmy Lin. 2021. Exploring listwise evidence reasoning with t5 for fact verification. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 2: Short Papers)*, ACL-IJCNLP '21, pages 402–410, Online. Association for Computational Linguistics.

Zhiwei Jin, Juan Cao, Han Guo, Yongdong Zhang, and Jiebo Luo. 2017a. Multimodal fusion with recurrent neural networks for rumor detection on microblogs. In *Proceedings of the 25th ACM international conference on Multimedia*, pages 795–816.

Zhiwei Jin, Juan Cao, Yongdong Zhang, Jianshe Zhou, and Qi Tian. 2017b. Novel visual and statistical image features for microblogs news verification. *IEEE Transactions on Multimedia*, 19(3):598–608.

Srecko Joksimovic, Ryan S. Baker, Jaclyn Ocumpaugh, Juan Miguel L. Andres, Ivan Tot, Elle Yuan Wang, and Shane Dawson. 2019. Automated identification of verbally abusive behaviors in online discussions. In *Proceedings of the Third Workshop on Abusive Language Online*, pages 36–45, Florence, Italy. Association for Computational Linguistics.

SeongKu Kang, Junyoung Hwang, and Hwanjo Yu. 2020. Multi-modal component embedding for fake news detection. In *14th International Conference on Ubiquitous Information Management and Communication (IMCOM)*, pages 1–6. IEEE.

Heysem Kaya and Alexey A Karpov. 2016. Fusing acoustic feature representations for computational paralinguistics tasks. In *Interspeech*, volume 2016, pages 2046–2050.

Ashkan Kazemi, Kiran Garimella, Devin Gaffney, and Scott Hale. 2021. Claim matching beyond English to scale global fact-checking. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint*

Conference on Natural Language Processing (Volume 1: Long Papers), ACL-IJCNLP '21, pages 4504–4517, Online. Association for Computational Linguistics.

Douwe Kiela, Suvrat Bhooshan, Hamed Firooz, and Davide Testuggine. 2019. Supervised multimodal bitransformers for classifying images and text. *arXiv:1909.02950*.

Douwe Kiela, Hamed Firooz, Aravind Mohan, Vedanuj Goswami, Amanpreet Singh, Pratik Ringshia, and Davide Testuggine. 2020. The hateful memes challenge: Detecting hate speech in multimodal memes. In *Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020, December 6-12, 2020, virtual*.

Daniel Kopev, Ahmed Ali, Ivan Koychev, and Preslav Nakov. 2019. Detecting deception in political debates using acoustic and textual features. In *IEEE Automatic Speech Recognition and Understanding Workshop (ASRU)*, pages 652–659. IEEE.

Neema Kotonya and Francesca Toni. 2020a. Explainable automated fact-checking: A survey. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 5430–5443, Barcelona, Spain (Online). International Committee on Computational Linguistics.

Neema Kotonya and Francesca Toni. 2020b. Explainable automated fact-checking: A survey. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 5430–5443, Barcelona, Spain (Online). International Committee on Computational Linguistics.

Gangeshwar Krishnamurthy, Navonil Majumder, Soujanya Poria, and Erik Cambria. 2018. A deep learning approach for multimodal deception detection. *arXiv:1803.00344*.

Srijan Kumar and Neil Shah. 2018. False information on web and social media: A survey. *arXiv:1804.08559*.

Sejeong Kwon, Meeyoung Cha, and Kyomin Jung. 2017. Rumor detection over varying time windows. *PLoS ONE*, 12(1).

Sarah Ita Levitan, Guozhen An, Min Ma, Rivka Levitan, Andrew Rosenberg, and Julia Hirschberg. 2016. Combining acoustic-prosodic, lexical, and phonotactic features for automatic deception detection. In *Interspeech*, pages 2006–2010.

Stephan Lewandowsky, Ullrich K.H. Ecker, and John Cook. 2017. Beyond misinformation: Understanding and coping with the "post-truth" era. *Journal of Applied Research in Memory and Cognition*, 6(4):353–369.

Liunian Harold Li, Mark Yatskar, Da Yin, Cho-Jui Hsieh, and Kai-Wei Chang. 2019. VisualBERT: A simple and performant baseline for vision and language. *arXiv:1908.03557*.

Yaliang Li, Jing Gao, Chuishi Meng, Qi Li, Lu Su, Bo Zhao, Wei Fan, and Jiawei Han. 2016. A survey on truth discovery. In *SIGKDD*, volume 17, pages 1–16.

Junwei Liang, Lu Jiang, and Alexander G. Hauptmann. 2017. Temporal localization of audio events for conflict monitoring in social media. In *2017 IEEE International Conference on Acoustics, Speech and Signal Processing, ICASSP 2017, New Orleans, LA, USA, March 5-9, 2017*, pages 1597–1601. IEEE.

Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C Lawrence Zitnick. 2014. Microsoft coco: Common objects in context. In *European conference on computer vision*, pages 740–755. Springer.

Wei Liu, Sihan Chen, Longteng Guo, Xinxin Zhu, and Jing Liu. 2021. Cptr: Full transformer network for image captioning. *arXiv preprint arXiv:2101.10804*.

Yang Liu and Yi-fang Brook Wu. 2018. Early detection of fake news on social media through propagation path classification with recurrent and convolutional networks. In *Proceedings of the Thirty-Second AAAI Conference on Artificial Intelligence, (AAAI-18), the 30th innovative Applications of Artificial Intelligence (IAAI-18), and the 8th AAAI Symposium on Educational Advances in Artificial Intelligence (EAAI-18), New Orleans, Louisiana, USA, February 2-7, 2018*, pages 354–361. AAAI Press.

Jiasen Lu, Dhruv Batra, Devi Parikh, and Stefan Lee. 2019. Vilbert: Pretraining task-agnostic visiolinguistic representations for vision-and-language tasks. In *Advances in Neural Information Processing Systems 32: Annual Conference on Neural Information Processing Systems 2019, NeurIPS 2019, December 8-14, 2019, Vancouver, BC, Canada*, pages 13–23.

Jing Ma, Wei Gao, Prasenjit Mitra, Sejeong Kwon, Bernard J Jansen, Kam Fai Wong, and Meeyoung Cha. 2016. Detecting rumors from microblogs with recurrent neural networks. In *IJCAI International Joint Conference on Artificial Intelligence*, volume 2016, pages 3818–3824.

Jing Ma, Wei Gao, and Kam-Fai Wong. 2017. Detect rumors in microblog posts using propagation structure via kernel learning. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 708–717, Vancouver, Canada. Association for Computational Linguistics.

Enrico Mariconti, Guillermo Suarez-Tangil, Jeremy Blackburn, Emiliano De Cristofaro, Nicolas Kourtellis, Ilias Leontiadis, Jordi Luque Serrano, and Gianluca Stringhini. 2019. "You Know What to Do":

Proactive detection of YouTube videos targeted by coordinated hate attacks. *Proceedings of the ACM on Human-Computer Interaction*, 3:1–21.

Binny Mathew, Ritam Dutt, Pawan Goyal, and Animesh Mukherjee. 2019. Spread of hate speech in online social media. In *Proceedings of the 10th ACM conference on web science*, pages 173–182.

Youssef Mroueh, Etienne Marcheret, and Vaibhava Goel. 2015. Deep multimodal learning for audio-visual speech recognition. In *2015 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 2130–2134. IEEE.

Eric Müller-Budack, Jonas Theiner, Sebastian Diering, Maximilian Idahl, and Ralph Ewerth. 2020. Multimodal analytics for real-world news using measures of cross-modal entity consistency. *arXiv:2003.10421*.

Kai Nakamura, Sharon Levy, and William Yang Wang. 2020. Fakeddit: A new multimodal benchmark dataset for fine-grained fake news detection. In *Proceedings of the 12th Language Resources and Evaluation Conference*, pages 6149–6157.

Preslav Nakov, Alberto Barrón-Cedeño, Giovanni Da San Martino, Firoj Alam, Julia Maria Struß, Thomas Mandl, Rubén Míguez, Tommaso Caselli, Mucahid Kutlu, Wajdi Zaghouani, Chengkai Li, Shaden Shaar, Gautam Kishore Shahi, Hamdy Mubarak, Alex Nikolov, Nikolay Babulkov, Yavuz Selim Kartal, Javier Beltrán, Michael Wiegand, Melanie Siegel, and Juliane Köhler. 2022a. Overview of the CLEF-2022 CheckThat! lab on fighting the COVID-19 infodemic and fake news detection. In *Proceedings of the 13th International Conference of the CLEF Association: Information Access Evaluation meets Multilinguality, Multimodality, and Visualization*, CLEF '2022, Bologna, Italy.

Preslav Nakov, Alberto Barrón-Cedeño, Giovanni Da San Martino, Firoj Alam, Julia Maria Struß, Thomas Mandl, Rubén Míguez, Tommaso Caselli, Mucahid Kutlu, Wajdi Zaghouani, Chengkai Li, Shaden Shaar, Gautam Kishore Shahi, Hamdy Mubarak, Alex Nikolov, Nikolay Babulkov, Yavuz Selim Kartal, and Javier Beltrán. 2022b. The CLEF-2022 CheckThat! Lab on fighting the covid-19 infodemic and fake news detection. In *Advances in Information Retrieval*, pages 416–428, Cham. Springer International Publishing.

Preslav Nakov and Giovanni Da San Martino. 2020. Fact-checking, fake news, propaganda, and media bias: Truth seeking in the post-truth era. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: Tutorial Abstracts*, pages 7–19, Online. Association for Computational Linguistics.

Van-Hoang Nguyen, Kazunari Sugiyama, Preslav Nakov, and Min-Yen Kan. 2020. FANG: leveraging social context for fake news detection using graph representation. In *The 29th ACM International Conference on Information and Knowledge Management*, pages 1165–1174. ACM.

Leonardo Nizzoli, Serena Tardelli, Marco Avvenuti, Stefano Cresci, and Maurizio Tesconi. 2021. Coordinated behavior on social media in 2019 uk general election. In *Proceedings of the International AAAI Conference on Web and Social Media*, volume 15, pages 443–454.

Leonardo Nizzoli, Serena Tardelli, Marco Avvenuti, Stefano Cresci, Maurizio Tesconi, and Emilio Ferrara. 2020. Charting the landscape of online cryptocurrency manipulation. *IEEE Access*, 8:113230–113245.

Deidre Olsen. 2018. How memes are being weaponized for political propaganda. *Salon. Salon. com, February*, 24.

Diogo Pacheco, Pik-Mai Hui, Christopher Torres-Lugo, Bao Tran Truong, Alessandro Flammini, and Filippo Menczer. 2021. Uncovering coordinated networks on social media: Methods and case studies. In *Proceedings of the International AAAI Conference on Web and Social Media*, volume 15, pages 455–466.

William Scott Paka, Rachit Bansal, Abhay Kaushik, Shubhashis Sengupta, and Tanmoy Chakraborty. 2021. Cross-SEAN: A cross-stitch semi-supervised neural attention model for covid-19 fake news detection. *Applied Soft Computing*, 107:107393.

Parth Patwa, Shivam Sharma, Srinivas Pykl, Vineeth Guptha, Gitanjali Kumari, Md Shad Akhtar, Asif Ekbal, Amitava Das, and Tanmoy Chakraborty. 2021. Fighting an infodemic: Covid-19 fake news dataset. In *Combating Online Hostile Posts in Regional Languages during Emergency Situation*, pages 21–29, Cham. Springer International Publishing.

Verónica Pérez-Rosas, Mohamed Abouelenien, Rada Mihalcea, and Mihai Burzo. 2015. Deception detection using real-life trial data. In *ICMI*, pages 59–66, New York, NY, USA. ACM.

Shraman Pramanick, Dimitar Dimitrov, Rituparna Mukherjee, Shivam Sharma, Md. Shad Akhtar, Preslav Nakov, and Tanmoy Chakraborty. 2021a. Detecting harmful memes and their targets. In *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*, pages 2783–2796, Online. Association for Computational Linguistics.

Shraman Pramanick, Shivam Sharma, Dimitar Dimitrov, Md Shad Akhtar, Preslav Nakov, and Tanmoy Chakraborty. 2021b. Momenta: A multimodal framework for detecting harmful memes and their targets. *arXiv preprint arXiv:2109.05184*.

Peng Qi, Juan Cao, Tianyun Yang, Junbo Guo, and Jintao Li. 2019. Exploiting multi-domain visual information for fake news detection. In *IEEE International Conference on Data Mining (ICDM)*, pages 518–527. IEEE.

Jing Qian, Mai ElSherief, Elizabeth Belding, and William Yang Wang. 2018. Hierarchical CVAE for fine-grained hate speech classification. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 3550–3559, Brussels, Belgium. Association for Computational Linguistics.

Rahat Ibn Rafiq, Homa Hosseinmardi, Richard Han, Qin Lv, Shivakant Mishra, and Sabrina Arredondo Mattson. 2015. Careful what you share in six seconds: Detecting cyberbullying instances in vine. In *Proceedings of the 2015 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining*, pages 617–622, Paris, France. ACM.

Hannah Rashkin, Eunsol Choi, Jin Yea Jang, Svitlana Volkova, and Yejin Choi. 2017. Truth of varying shades: Analyzing language in fake news and political fact-checking. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 2931–2937, Copenhagen, Denmark. Association for Computational Linguistics.

Adrian Rauchfleisch, Xenia Artho, Julia Metag, Senja Post, and Mike S. Schäfer. 2017. How journalists verify user-generated content during terrorist crises. analyzing twitter communication during the brussels attacks. *Social Media + Society*, 3(3):2056305117717888.

Manoel Ribeiro, Pedro Calais, Yuri Santos, Virgílio Almeida, and Wagner Meira Jr. 2018. Characterizing and detecting hateful users on Twitter. In *Twelfth international AAAI conference on web and social media*, volume 12.

Aalok Sathe, Salar Ather, Tuan Manh Le, Nathan Perry, and Joonsuk Park. 2020. Automated fact-checking of claims from Wikipedia. In *Proceedings of the 12th Language Resources and Evaluation Conference*, ACL '20, pages 6874–6882, Marseille, France. European Language Resources Association.

Anna Schmidt and Michael Wiegand. 2017a. A survey on hate speech detection using natural language processing. In *Proceedings of the Fifth International Workshop on Natural Language Processing for Social Media*, pages 1–10, Valencia, Spain. Association for Computational Linguistics.

Anna Schmidt and Michael Wiegand. 2017b. A survey on hate speech detection using natural language processing. In *Proceedings of the Fifth International Workshop on Natural Language Processing for Social Media*, pages 1–10, Valencia, Spain. Association for Computational Linguistics.

Umut Mehmet Sen, Veronica Perez-Rosas, Berrin Yanikoglu, Mohamed Abouelenien, Mihai Burzo, and Rada Mihalcea. 2020. Multimodal deception detection using real-life trial data. *IEEE Transactions on Affective Computing*.

Hyunjin Seo. 2014. Visual propaganda in the age of social media: An empirical analysis of Twitter images during the 2012 Israeli–Hamas conflict. *Visual Communication Quarterly*, 21(3):150–161.

Shaden Shaar, Firoj Alam, Giovanni Da San Martino, and Preslav Nakov. 2022. The role of context in detecting previously fact-checked claims. In *Findings of the Association for Computational Linguistics: NAACL 2022*, pages 1619–1631, Seattle, United States. Association for Computational Linguistics.

Shaden Shaar, Firoj Alam, Giovanni Da San Martino, and Preslav Nakov. 2021. Assisting the human fact-checkers: Detecting all previously fact-checked claims in a document. *arXiv preprint arXiv:2109.07410*.

Shaden Shaar, Nikolay Babulkov, Giovanni Da San Martino, and Preslav Nakov. 2020. That is a known lie: Detecting previously fact-checked claims. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, ACL '20, pages 3607–3618, Online. Association for Computational Linguistics.

Chengcheng Shao, Giovanni Luca Ciampaglia, Onur Varol, Kai-Cheng Yang, Alessandro Flammini, and Filippo Menczer. 2018. The spread of low-credibility content by social bots. *Nature communications*, 9(1):1–9.

Shivam Sharma, Firoj Alam, Md. Shad Akhtar, Dimitar Dimitrov, Giovanni Da San Martino, Hamed Firooz, Alon Halevy, Fabrizio Silvestri, Preslav Nakov, and Tanmoy Chakraborty. 2022. Detecting and understanding harmful memes: A survey. In *Proceedings of the Thirty-First International Joint Conference on Artificial Intelligence, IJCAI-22*, pages 5597–5606. International Joint Conferences on Artificial Intelligence Organization. Survey Track.

Kai Shu, Deepak Mahudeswaran, Suhang Wang, and Huan Liu. 2020. Hierarchical propagation networks for fake news detection: Investigation and exploitation. In *Proceedings of the International AAAI Conference on Web and Social Media*, volume 14, pages 626–637.

Kai Shu, Amy Sliva, Suhang Wang, Jiliang Tang, and Huan Liu. 2017. Fake news detection on social media: A data mining perspective. *ACM SIGKDD explorations newsletter*, 19(1):22–36.

Kai Shu, Suhang Wang, and Huan Liu. 2019. Beyond news contents: The role of social context for fake news detection. In *Proceedings of the Twelfth ACM International Conference on Web Search and Data Mining, WSDM 2019, Melbourne, VIC, Australia, February 11-15, 2019*, pages 312–320. ACM.

Jiasheng Si, Deyu Zhou, Tongzhe Li, Xingyu Shi, and Yulan He. 2021. Topic-aware evidence reasoning and stance-aware aggregation for fact verification. In *Proceedings of the 59th Annual Meeting of the*

Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers), pages 1612–1622, Online. Association for Computational Linguistics.

Shivangi Singhal, Rajiv Ratn Shah, Tanmoy Chakraborty, Ponnurangam Kumaraguru, and Shin'ichi Satoh. 2019. SpotFake: A multi-modal framework for fake news detection. In *2019 IEEE fifth international conference on multimedia big data (BigMM)*, pages 39–47. IEEE.

Felix Soldner, Verónica Pérez-Rosas, and Rada Mihalcea. 2019. Box of lies: Multimodal deception detection in dialogues. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 1768–1777, Minneapolis, Minnesota. Association for Computational Linguistics.

Chenguang Song, Nianwen Ning, Yunlei Zhang, and Bin Wu. 2021. A multimodal fake news detection model based on crossmodal attention residual and multichannel convolutional neural networks. *Information Processing & Management*, 58(1):102437.

Xiaoyu Song, Hong Chen, Qing Wang, Yunqiang Chen, Mengxiao Tian, and Hui Tang. 2019. A review of audio-visual fusion with machine learning. In *Journal of Physics: Conference Series*, volume 1237, page 022144. IOP Publishing.

Devin Soni and Vivek K Singh. 2018. See no evil, hear no evil: Audio-visual-textual cyberbullying detection. *Proceedings of the ACM on Human-Computer Interaction*, 2(CSCW):1–26.

Jabeen Summaira, Xi Li, Amin Muhammad Shoib, Songyuan Li, and Jabbar Abdul. 2021. Recent advances and trends in multimodal deep learning: A review. *arXiv: 2105.11087*.

Reuben Tan, Bryan Plummer, and Kate Saenko. 2020. Detecting cross-modal inconsistency to defend against neural fake news. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 2081–2106, Online. Association for Computational Linguistics.

James Thorne and Andreas Vlachos. 2018. Automated fact checking: Task formulations, methods and future directions. In *Proceedings of the 27th International Conference on Computational Linguistics*, pages 3346–3359, Santa Fe, New Mexico, USA. Association for Computational Linguistics.

Cynthia Van Hee, Els Lefever, Ben Verhoeven, Julie Mennes, Bart Desmet, Guy De Pauw, Walter Daelemans, and Veronique Hoste. 2015. Detection and fine-grained classification of cyberbullying events. In *Proceedings of the International Conference Recent Advances in Natural Language Processing*, pages 672–680, Hissar, Bulgaria. INCOMA Ltd. Shoumen, BULGARIA.

Prashanth Vijayaraghavan, Hugo Larochelle, and Deb Roy. 2021. Interpretable multi-modal hate speech detection. *arXiv preprint arXiv:2103.01616*.

Nguyen Vo and Kyumin Lee. 2018. The rise of guardians: Fact-checking URL recommendation to combat fake news. In *The 41st International ACM SIGIR Conference on Research & Development in Information Retrieval, SIGIR 2018, Ann Arbor, MI, USA, July 08-12, 2018*, pages 275–284. ACM.

Svitlana Volkova, Ellyn Ayton, Dustin L Arendt, Zhuanyi Huang, and Brian Hutchinson. 2019. Explaining multimodal deceptive news prediction models. In *Proceedings of the International AAAI Conference on Web and Social Media*, volume 13, pages 659–662.

Soroush Vosoughi, Mostafa 'Neo' Mohsenvand, and Deb Roy. 2017. Rumor gauge: Predicting the veracity of rumors on Twitter. *ACM transactions on knowledge discovery from data (TKDD)*, 11(4):1–36.

Soroush Vosoughi, Deb Roy, and Sinan Aral. 2018a. The spread of true and false news online. *Science*, 359(6380):1146–1151.

Soroush Vosoughi, Deb Roy, and Sinan Aral. 2018b. The spread of true and false news online. *Science*, 359(6380):1146–1151.

Hai Wan, Haicheng Chen, Jianfeng Du, Weilin Luo, and Rongzhen Ye. 2021. A DQN-based approach to finding precise evidences for fact verification. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, ACL-IJCNLP '21, pages 1030–1039, Online. Association for Computational Linguistics.

Kaige Wang, Qingyu Xiong, Chao Wu, Min Gao, and Yang Yu. 2020. Multi-modal cyberbullying detection on social networks. In *International Joint Conference on Neural Networks (IJCNN)*, pages 1–8.

Yaqing Wang, Fenglong Ma, Zhiwei Jin, Ye Yuan, Guangxu Xun, Kishlay Jha, Lu Su, and Jing Gao. 2018. EANN: event adversarial neural networks for multi-modal fake news detection. In *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining, KDD 2018, London, UK, August 19-23, 2018*, pages 849–857. ACM.

Yuping Wang, Fatemeh Tahmasbi, Jeremy Blackburn, Barry Bradlyn, Emiliano De Cristofaro, David Magerman, Savvas Zannettou, and Gianluca Stringhini. 2021. Understanding the use of fauxtography on social media. *Proceedings of the International AAAI Conference on Web and Social Media*, 15(1):776–786.

Zeerak Waseem, Thomas Davidson, Dana Warmsley, and Ingmar Weber. 2017. Understanding abuse: A typology of abusive language detection subtasks. In

Proceedings of the First Workshop on Abusive Language Online, pages 78–84, Vancouver, BC, Canada. Association for Computational Linguistics.

Zeerak Waseem and Dirk Hovy. 2016. Hateful symbols or hateful people? predictive features for hate speech detection on Twitter. In *Proceedings of the NAACL Student Research Workshop*, pages 88–93, San Diego, California. Association for Computational Linguistics.

Derek Weber and Frank Neumann. 2020. Who's in the gang? revealing coordinating communities in social media. In *IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining (ASONAM)*, pages 89–93. IEEE.

Fan Yang, Xiaochang Peng, Gargi Ghosh, Reshef Shilon, Hao Ma, Eider Moore, and Goran Predovic. 2019a. Exploring deep multimodal fusion of text and photo for hate speech classification. In *Proceedings of the Third Workshop on Abusive Language Online*, pages 11–18, Florence, Italy. Association for Computational Linguistics.

Shuo Yang, Kai Shu, Suhang Wang, Renjie Gu, Fan Wu, and Huan Liu. 2019b. Unsupervised fake news detection on social media: A generative approach. In *Proceedings of the AAAI conference on artificial intelligence*, volume 33, pages 5644–5651.

Yang Yang, Lei Zheng, Jiawei Zhang, Qingcai Cui, Zhoujun Li, and Philip S Yu. 2018. TI-CNN: Convolutional neural networks for fake news detection. *arXiv:1806.00749*.

Savvas Zannettou, Tristan Caulfield, Jeremy Blackburn, Emiliano De Cristofaro, Michael Sirivianos, Gianluca Stringhini, and Guillermo Suarez-Tangil. 2018. On the origins of memes by means of fringe web communities. In *Proceedings of the Internet Measurement Conference 2018*, pages 188–202.

John Zarocostas. 2020. How to fight an infodemic. *The lancet*, 395(10225):676.

Daniel Yue Zhang, Lanyu Shang, Biao Geng, Shuyue Lai, Ke Li, Hongmin Zhu, Md. Tanvir Al Amin, and Dong Wang. 2018. Fauxbuster: A content-free fauxtography detector using social media comments. In *IEEE International Conference on Big Data*, pages 891–900. IEEE.

Guobiao Zhang, Anastasia Giachanou, and Paolo Rosso. 2022. Scenefnd: Multimodal fake news detection by modelling scene context information. *Journal of Information Science*, page 01655515221087683.

Huaiwen Zhang, Quan Fang, Shengsheng Qian, and Changsheng Xu. 2019. Multi-modal knowledge-aware event memory network for social media rumor detection. In *Proceedings of the 27th ACM International Conference on Multimedia*, pages 1942–1951.

Xinyi Zhou, Jindi Wu, and Reza Zafarani. 2020. SAFE: Similarity-aware multi-modal fake news detection. In *Pacific-Asia Conference on Knowledge Discovery and Data Mining*, pages 354–367. Springer.

Xinyi Zhou and Reza Zafarani. 2019. Network-based fake news detection: A pattern-driven approach. *ACM SIGKDD explorations newsletter*, 21(2):48–60.

Xinyi Zhou and Reza Zafarani. 2020. A survey of fake news: Fundamental theories, detection methods, and opportunities. *ACM Computing Surveys (CSUR)*, 53(5):1–40.

Hao Zhu, Man-Di Luo, Rui Wang, Ai-Hua Zheng, and Ran He. 2021. Deep audio-visual learning: A survey. *International Journal of Automation and Computing*, pages 1–26.

Dimitrina Zlatkova, Preslav Nakov, and Ivan Koychev. 2019. Fact-checking meets fauxtography: Verifying claims about images. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 2099–2108, Hong Kong, China. Association for Computational Linguistics.

Appendix

A Examples of Factuality and Harmful Content

In Figure 2, we provide examples textual and visual content that are harmful and false, true image with false claim, and harmful meme.

B Modeling Techniques

Figure 4 shows various multimodal approaches that have been proposed in the literature.

C Lessons Learned

1. A lot of progress has been made on the problem, but the two components in the definition of disinformation (falseness and harmfulness) have been considered mostly in isolation. We argue that there is a need for tight integration of the factuality and the intentional harmfulness into the same detection model. These two aspects have been addressed together in (Alam et al., 2021), which shows that 56% of Arabic false content is also harmful. From Tables 1 and 2, we observe that most multimodal datasets cover just 2–3 modalities, which combine some approaches depicted in Figure 4. Moreover, no multimodal dataset

[6] https://www.snopes.com/fact-check/abe-lincoln-racist-protest-sign/

Figure 2: Examples of textual and visual contents that show (a) *fauxotographic content* (which is both harmful and false), [6] (b) harmful content promoting bad cure (text-only, and false), (c) true image with a false claim about it (malicious), and (d) harmful content, where the text and the image collectively appeal to fear.

looks at both aspects of disinformation: factuality and harmfulness. While Alam et al. (2021) did address both aspects, they only covered the text modality.

2. In the early phase of (dis)information spreading, user and content features are those that provide the highest contribution for detecting factuality. Indeed, at that time, a few interactions with content are available and the propagation network is small and sparse. As information spreads, the contribution of content-derived features remains constant, while propagation-derived features become richer and more informative. In summary, early prediction of factuality and veracity must necessarily rely heavily on users and content – be it text, image, audio or video. Instead, analyses carried out at later times benefit more from network and temporal data. In the past decade, research on multimodality has shown its potential in several fields, which include audio-visual fusion (Mroueh et al., 2015; Zhu et al., 2021; Song et al., 2019), emotion recognition (Chen et al., 2021), image and video captioning (Liu et al., 2021), multimedia retrieval and visual question answering (Summaira et al., 2021). For factuality, Baly et al. (2020) showed that combining different modalities such as text, speech, and metadata yields improved performance compared to using individual modalities. Similar phenomena have been observed for other tasks such as hateful memes (Kiela et al., 2020), and propaganda detection (Dimitrov et al., 2021b).

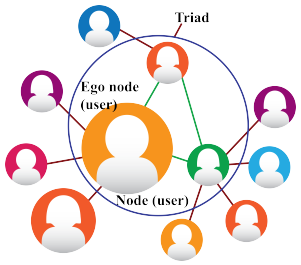

Figure 3: Example of social network with users. **Node:** A node can be a users or a spreader. **Ego:** "Ego" is an individual "focal" node (central user) and the nodes that are directly connected to it are called "alters/spreaders." **Triad:** It (a set of three connected users) is the most basic subgraph of the network. **Community:** A community structure refers to the occurrence of groups of nodes in a network that are more densely connected internally than with the rest of the network.

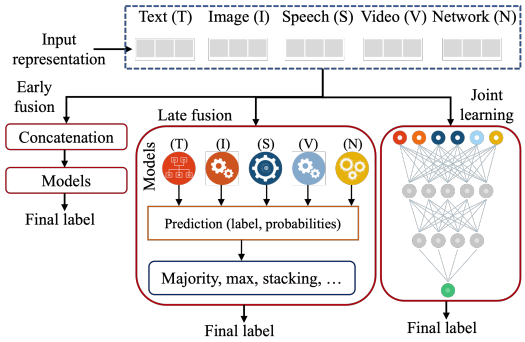

Figure 4: Multimodal approaches, including early and late fusion, and joint modal learning. The hybrid approach (combining early and late fusion) is not shown.

D More Challenges

1. **Contextualization.** Existing methods of disinformation detection are mostly non-contextualized, i.e., the broader context of a news article in terms of the responses of the readers and how the users perceive them are not captured. We argue that the response

thread under a news, the underlying social network among users, the propagation dynamics of the news and its mentions across social media need suitable integration to better capture the overall perspective on the news.

2. **Meta Information.** Along with the news and the context, other information such as the authenticity of the news, the credibility of the authors of the news, the factuality of the news also play an important role for disinformation detection. Moreover, detecting whether the disinformation attack is a coordinated effort or an individual activity would also help understanding its severity.

3. **Bias, Region, and Cultural Awareness.** The performance of most of the existing systems is limited to the underlying dataset, particularly to the demography and the underlying cultural aspects. For instance, a model trained on an Indian political dataset may not generalize well to a US health-related dataset (Fortuna et al., 2021).

4. **Disinformation on Evolving Topics.** Often, claims or harmful content are disseminated based on the current event; information about COVID-19 and vaccines are examples of such use cases. Existing models might fail on such use cases, and thus zero-shot or few-shot learning might be an important future avenue to explore.

5. **Transparent and Accountable Models.** The detection models should be designed in a way that their outcomes are unbiased and more accountable to ethical considerations. The models for disinformation detection should present the outcome in such a way that a practitioner can interpret it and understand why a piece of information is flagged as disinformation, what is the related real news based on which the judgment was made, and which part of the information was counterfeit. There is also a lack of datasets containing disinformation with explanations and the corresponding real information.

6. **Fine-grained Detection.** Existing disinformation detection models are mostly binary classifiers: given a piece of news, they aim to detect whether it is a disinformation or not. Such binary signals might be enough in certain cases. However, in many other cases, more fine-grained labels can help to make a better decision. For example, whether a social media post is fake or genuine can help fact-checkers, but having more fine-grained information such as true, satire/parody, misleading, manipulated, false connection, or imposter content can be even more helpful (Nakamura et al., 2020). Therefore, rather than a binary classification, one could cast the problem as a multi-class classification task or even an ordinal regression, or just a regression task. This would also help prioritize disinformation for reactive measurements.

Why Is It Hate Speech?
Masked Rationale Prediction for Explainable Hate Speech Detection

Jiyun Kim, Byounghan Lee, Kyung-Ah Sohn*
Ajou University
{hamjee66, qudgks96, kasohn}@ajou.ac.kr
*Corresponding author

Abstract

In a hate speech detection model, we should consider two critical aspects in addition to detection performance–bias and explainability. Hate speech cannot be identified based solely on the presence of specific words; the model should be able to reason like humans and be explainable. To improve the performance concerning the two aspects, we propose Masked Rationale Prediction (MRP) as an intermediate task. MRP is a task to predict the masked human rationales–snippets of a sentence that are grounds for human judgment–by referring to surrounding tokens combined with their unmasked rationales. As the model learns its reasoning ability based on rationales by MRP, it performs hate speech detection robustly in terms of bias and explainability. The proposed method generally achieves state-of-the-art performance in various metrics, demonstrating its effectiveness for hate speech detection. *Warning: This paper contains samples that may be upsetting.*

1 Introduction

With the recent development of social media and online communities, hate speech, one of the critical social problems, can spread easily. The spread of hate strengthens discrimination and prejudice against the target social groups and can violate their human rights. Moreover, online hatred extends offline and causes real-world crimes. Therefore, properly regulating online hate speech is important to address many social problems related to aversion.

In addition to the detection performance, two essential considerations are involved in implementing a hate speech detection model–*bias* and *explainability*. Hate speech should not be judged by any specific word but by the context in which the word is used. Even if any word generally considered vicious does not exist in a text, the text can be hate speech. A specific expression does not always imply hatred either (*e.g.*, e.g., 'nigger') (Del Vigna12 et al., 2017). However, the presence of this word can cause a model to make a biased detection of hate speech. This erroneous judgment may inadvertently strengthen the discrimination against the target group of the expression (Sap et al., 2019; Davidson et al., 2019). In this respect, the model's *bias* toward specific expressions should be excluded.

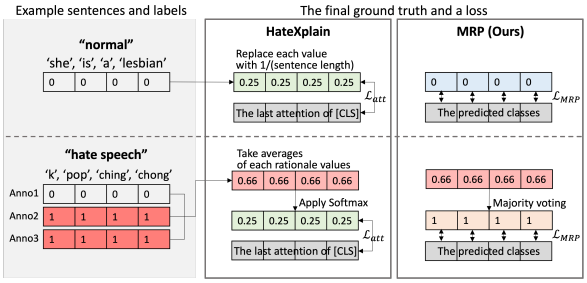

Figure 1: Examples for the two methods to get the final ground truths. Example input sentences are represented with the class and human rationale labels. In this figure, HateXplain uses the same ground truth about both normal and hateful sentences for the loss. However, our method could determine the two classes with the ground truths.

The expressions that can cause biased judgment should be interpreted in context. It means it is vital for the hate speech detection models to have the ability to make judgments based on context, as humans do. Therefore, the model should be *explainable* to humans so that the rationale behind a result is explained (Liu et al., 2018). Here, the rationale is a piece of a sentence as justification for the model's prediction about the sentence, as defined by related research (Hancock et al., 2018; Lei et al., 2016).

To the best of our knowledge, HateXplain (Mathew et al., 2020) is the first hate speech detection benchmark dataset that considers both these aspects. They proposed a method that utilizes rationales as attention ground truths to complement the performance of the two elements. However, when most tokens are annotated as the human rationale in a hateful sentence, the rationale's information

Proceedings of the 29th International Conference on Computational Linguistics, pages 6644–6655
October 12–17, 2022.

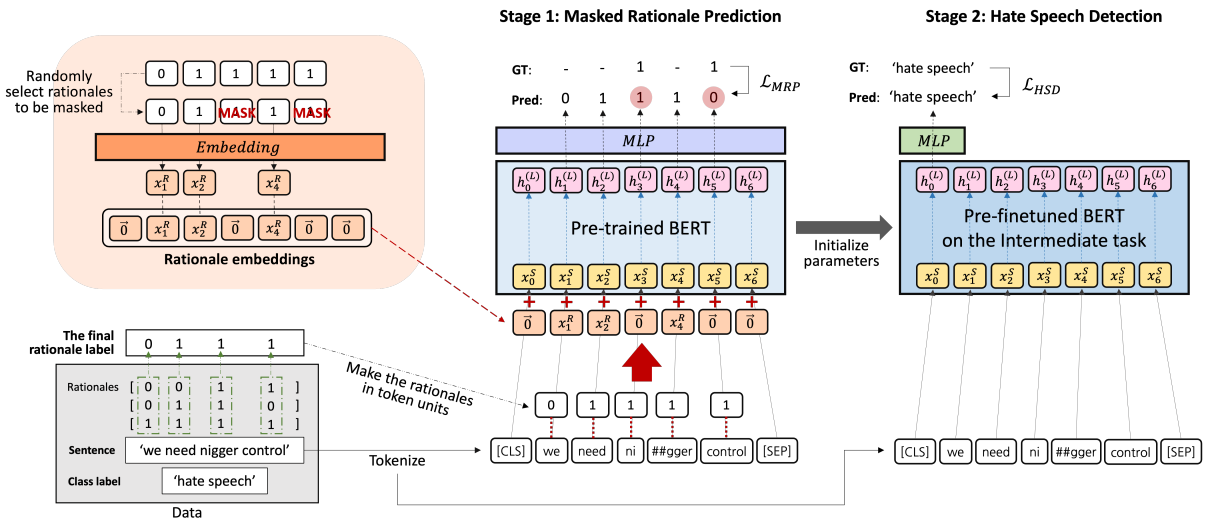

Figure 2: Framework of the proposed method. We finetune a pre-trained BERT through two training stages–Masked Rationale Prediction (MRP) and then hate speech detection. In MRP, the partially masked rationale label is inputted as the rationale embeddings by being added into the input embeddings of BERT. The model predicts each masked rationale per token. The model for hate speech detection is initialized by the updated parameters during MRP.

could be meaningless as the ground truth attention becomes hard to be distinguished from that of a normal sentence, as shown in Figure 1. This can hinder the model's learning.

In this paper, we present a method to implement a hate speech detection model much more effectively by using the human rationale of hate for finetuning a pre-trained language model. To achieve this, we propose Masked Rationale Prediction (MRP) as an intermediate task before finetuning the model on hate speech detection. MRP trains a model to predict the human rationale label of each token by referring to the context of the input sentence. The model takes the human rationale information of some input tokens among the sentence along with the corresponding tokens as input. It then predicts the rationale of the remaining tokens on which the rationale is masked. We embed the rationales to provide the human rationales as input per token. The masking process of the partial rationales is implemented while creating rationale embeddings; some rationale embedding vectors are replaced with zero vectors.

MRP allows the model to make judgments per token about its masked rationale by considering surrounding tokens with an unmasked rationale. With this, the model learns a human-like reasoning process to get context-dependent abusiveness of tokens. The model parameters trained on MRP become the initial parameter values for hate speech detection in the following training stage. In this way, based on the way of human reasoning for hate, the model can get improved abilities in terms of bias and explainability in detecting hate speech. We experimented with BERT (Devlin et al., 2018) as the pre-trained model. Consequently, our models finetuned in the proposed way–BERT-MRP and BERT-RP–achieve state-of-the-art performance overall on all three types of 11 metrics of HateXplain benchmark–Performance-based, Bias-based, and Explainability-based (Mathew et al., 2020). And the two models, especially BERT-MRP, also show the best results in qualitative evaluation of explicit and implicit hate speech detection.

The main contributions of this paper are:

- We propose a method to utilize human rationales as input by transforming them into rationale embeddings. Combining the embedded rationales with the corresponding input sentence can provide information about the human rationales per token during model training.
- We propose Masked Rationale Prediction (MRP), a learning method that leads the model to predict the masked rationale by considering the surrounding tokens. The model is allowed to learn the reasoning process in context.
- We finetune a pre-trained BERT in two stages–on MRP as an intermediate task and then on hate speech detection. The parameters trained concerning human reasoning for hate become a sufficient basis not only for the detection but

also in terms of the model bias and explainability.

2 Related works

Hate speech detection With the advance of deep learning, hate speech detection studies have utilized neural networks (Badjatiya et al., 2017; Han and Eisenstein, 2019), and word embedding methods (McKeown and McGregor, 2018). More recently, Transformer-based (Vaswani et al., 2017) models have shown remarkable results. In hate speech detection, BERT has been adopted for various studies as hate speech detection can be considered a classification task. (Mandl et al., 2019; Ranasinghe et al., 2019) compared a BERT-based model with Recurrent Neural Networks (RNNs)-based models and showed the BERT-based model outperforms other models. Furthermore, some studies have considered the model's bias and explainability. (Vaidya et al., 2020) improved accuracy and reduced unintended bias by adopting multi-task learning that predicts toxicity of text and target group labels as additional information. (Mathew et al., 2020) utilized rationales of the dataset as additional information for finetuning BERT to deal with the bias and explainability. To improve performance in terms of the two considerations, we propose a more effective finetuning approach based on BERT and the rationales by adopting the pre-training framework.

Pre-finetuning on an intermediate task Recently, finetuning a pre-trained model on a downstream task has become the norm (Howard and Ruder, 2018; Radford et al., 2018). However, it cannot be guaranteed that the model finetuned with a small dataset compared to its size will be sufficiently well-adjusted for the target downstream task (Phang et al., 2018). Pre-finetuning is a technique to train the model on a task before the target task (Aghajanyan et al., 2021). This can help the model learn the data patterns or reduce the tuning time so that it converges quickly to better fit the target task. According to (Pruksachatkun et al., 2020; Aghajanyan et al., 2021), the more closely the intermediate task is related to the target task, the better the effect of pre-finetuning. And inference tasks involving the reasoning process show a remarkable improvement in the target task performance. We adopt this method to train a pre-trained language model through two stages for hate speech detection. As the intermediate task, we propose MRP, which guides the model to infer the human rationale of each token based on surrounding tokens.

Explainable NLP and rationale Explaining the rationale of the result of an AI model is necessary for it to be explainable to humans (Liu et al., 2018). Some Natural Language Processing (NLP) studies define rationale as snippets of an input text on which the model's prediction is supported (Hancock et al., 2018; Lei et al., 2016). (Lei et al., 2016) implemented a generator that generates words considered rationales and used them as input of an encoder for sentiment classification. (Bao et al., 2018) mapped the human rationales into the model attention values to solve the low-resource problem by learning a domain-invariant token representation. For hate speech detection, HateXplain employs the human rationales as ground truth attention to concentrate on aggressive tokens. Unlike existing approaches, we utilize the masked human rationale label embeddings as input. They become the useful additional information of each token.

Masked label prediction The UniMP model presented by (Shi et al., 2020) aims to solve the graph node classification problem using graph transformer networks (Yun et al., 2019). They maximized the propagation information required to reconstruct a partially observable label by using both feature information and label information as inputs. However, to prevent overfitting due to excessive information, some label information is masked, and the masked label is predicted. We apply a similar method to text data for an intermediate task with rationales. Through additional rationale information, the model increases the understanding of input sentences, and the performance of the downstream task is improved.

3 Method

Hate speech detection can be described as a text classification problem. Following the problem setting of HateXplain, we define the problem as a three-class classification involving three categories– 'hate speech,' 'offensive,' or 'normal'. We finetune a pre-trained BERT on hate speech detection. Note that other transformer encoder-based models can be used instead. Before finetuning the model on this task, we pre-finetune it on an intermediate task. We propose Masked Rationale Prediction (MRP) as the intermediate task. Our method is described in Figure 2.

3.1 Masked rationale prediction

For MRP, we utilize human rationales of hate provided by the HateXplain dataset. Annotators marked some words in a sentence as rationales for judging the sentence as abusive. A rationale label is presented in a list format, including 1 as rationale and 0 as non-rationale per word in the corresponding sentence. There are no such labels for a sentence whose final class is 'normal.' As the dataset was annotated by two or three people per sentence, some pre-processing is required to get the final rationale labels for MRP. To manipulate the multiple rationale labels to one per sentence, we obtain the average value of the rationales per word, and if it is over 0.5, the value of 1; otherwise, the value of 0 is determined as the final rationale of the corresponding word. The final rationale label is a list of these last values. In the case of the 'normal' sentence, a list of zeros is used. Accordingly, the final rationale label consists of as many 0s or 1s as the number of words in the sentence. As a sentence is tokenized, its rationale label is also modified in token units.

MRP is based on token classification, which predicts the rationale label R per token in an input sentence S. In our MRP, the rationale labels, as well as the sentences, are used as inputs. The process of embedding S is the same as that of BERT. We denote the embedded S as $X^S = \{x_0^S, x_1^S, \cdots, x_{n-1}^S\} \in \mathbb{R}^{n \times d}$ where n is the sequence length and d is the embedding size. And to use R as input, we pass it through an embedding layer to get $X^R = \{x_0^R, x_1^R, \cdots, x_{n-1}^R\} \in \mathbb{R}^{n \times d}$ as shown in Figure 2. The rationale embeddings reflect the attributes of each token as a ground for the human judgment.

MRP differs from BERT's Masked Language Modeling (MLM) in masking processing. Specifically, we do not mask the tokens; we mask the rationales. To construct the partially masked rationale embeddings $\tilde{X}^R$, some rationales are randomly selected to be masked. Each of rationales is transformed into its corresponding embedding vector, except the masked ones. For masking, zero vectors replace the embedding vectors of each corresponding token. For example, if we mask x_1^R and x_3^R, then the rationale embedding matrix is like $\tilde{X}^R = \{\vec{0}, \vec{0}, x_2^R, \vec{0}, \cdots, x_{n-2}^R, \vec{0}\}$. The first and last rationale embeddings corresponding to CLS and SEP tokens, respectively, are replaced with $\vec{0}$.

The MRP model predicts the rationale by taking the sum of the embedded tokens X^S and the partially masked rationales $\tilde{X}^R$ as input. We then get:

$$\begin{aligned} H_{MRP}^{(0)} &= X^S + \tilde{X}^R, \\ H_{MRP}^{(l+1)} &= \text{Transformer}(H_{MRP}^{(l)}), \\ \hat{X}^R &= \text{MLP}(H_{MRP}^{(L)}). \end{aligned} \tag{1}$$

The l-th hidden state passes through the transformer block to create the $l+1$-th hidden state, and the last hidden state $H_{MRP}^{(L)}$ outputs a predicted rationale $\hat{X}^R$ through Multi-Layer Perceptron (MLP). In other words, the model is guided to predict the masked rationales by referring to the representations of tokens using their corresponding observed rationales.

The loss $\mathcal{L}_{MRP}$ is calculated with only the predictions of the masked rationales. Therefore, our objective function is:

$$\begin{aligned} \arg\max_{\theta} \ \log \mathrm{p}_\theta \left(\hat{X}^R | X^S, \tilde{X}^R \right) = \\ \sum_{m \in M} \log \mathrm{p}_\theta \left(x_m^R | X^S, \tilde{X}^R \right), \end{aligned} \tag{2}$$

where M indicates a set of index numbers of rationales that have been masked.

3.2 Hate speech detection

Hate speech detection is implemented as three-class text classification. The model predicts which category Y the input sentence belongs to among 'hate speech', 'offensive', and 'normal'. The head that outputs the predicted class $\hat{Y}$ is used on the top of BERT. Before training, the model parameters are initialized by parameters updated on the intermediate task MRP, except for the head. As the forms of heads are different for two stages, their parameters are randomly initialized. Consequently, in the finetuning stage on hate speech detection, the rationale labels are not involved functionally, considered as $[0]_{n \times d}$. Therefore, in this stage, the input of the model is $H_{HSD}^{(0)} = X^S$.

In this stage, the model does not refer to the rationale labels. The parameters trained during MRP are utilized as a base for reasoning hatefulness in context. The loss $\mathcal{L}_{HSD}$ is obtained through a cross-entropy function, as the task is a multi-class classification problem.

$$\arg\max_{\theta} \ \log \mathrm{p}_\theta \left(\hat{Y} | X^S \right). \tag{3}$$

Model			Performance			Bias		
	ration.	pre-fin.	Acc.	Macro F1	AUROC	GMB-Sub.	GMB-BPSN	GMB-BNSP
BERT			69.0	67.4	84.3	76.2	70.9	75.7
BERT-HateXplain	✓		69.8	68.7	85.1	80.7	74.5	76.3
BERT-MLM		✓	70.0	67.5	85.4	79.0	67.7	80.9
BERT-RP	✓	✓	**70.7**	69.3	85.3	81.4	74.6	84.8
BERT-MRP	✓	✓	70.4	**69.9**	**86.2**	**81.5**	**74.8**	**85.4**

Table 1: Results for the performance-based and the bias-based metrics. Scores in bold type are the best for each corresponding metric, while the underlined are the second best, and so are in Table 2.

Model			Explainability				
			Plausibility			Faithfulness	
	ration.	pre-fin.	IOU F1	Token F1	AUPRC	Comp.	Suff. ↓
BERT [Att]			13.0	49.7	77.8	44.7	5.7
BERT [LIME]			11.8	46.8	74.7	43.6	0.8
BERT-HateXplain [Att]	✓		12.0	41.1	62.6	42.4	16.0
BERT-HateXplain [LIME]	✓		11.2	45.2	72.2	**50.0**	0.4
BERT-MLM [Att]		✓	13.5	43.5	60.8	40.1	11.9
BERT-MLM [LIME]		✓	11.3	47.2	76.5	43.4	**-5.5**
BERT-RP [Att]	✓	✓	13.8	50.3	73.8	45.4	7.2
BERT-RP [LIME]	✓	✓	11.4	49.3	77.7	48.6	-2.6
BERT-MRP [Att]	✓	✓	**14.1**	**50.4**	74.5	47.9	6.7
BERT-MRP [LIME]	✓	✓	12.9	50.1	**79.2**	48.3	-1.2

Table 2: Results for the explainability-based metrics. The lower the score Sufficiency in Faithfulness, the better, and the higher the other scores, the better.

4 Experiments

4.1 Dataset

For both stages of the intermediate and the target task, we use the HateXplain dataset. It contains 20,148 items collected from Twitter and Gab. Every item consists of one English sentence with its own ID and annotations about labels for its category, target groups, and rationales, which are annotated by two or three annotators. Based on the IDs, the dataset is split into 8:1:1 for training, validation, and test phases. Following the permanent split provided by the dataset, the models can't reference any test data during the training phases of all stages.

4.2 Metrics

The evaluation is according to the metrics of HateXplain, which are classified into three types: performance-based, bias-based, and explainability-based. The performance-based metrics measure the detection performance in distinguishing among three classes (i.e., hate speech, offensive, and normal). Accuracy, macro F1 score, and AUROC score are used as the metrics.

The bias-based metrics evaluate how biased the model is for specific expressions or profanities easily assumed to be hateful. HateXplain follows AUC-based metrics developed by (Borkan et al., 2019). The model classifies the data into 'toxic'–hateful and offensive–and 'non-toxic'–normal. For evaluating the model's prediction results, the data are separated into four subsets: D_g^+, D_g^-, D^+, and D^-. The target group labels are considered standard for dividing data into subgroups. The notations with g denote the data of a specific subgroup among the subgroups, and the notations without g are the remaining data. $+$ and $-$ mean that the data are toxic and non-toxic, respectively. Based on these subsets, three AUC metrics are calculated.

Subgroup AUC is to evaluate how biased the model is to the context of each target group: $AUC(D_g^- + D_g^+)$. The higher the score, the less biased the model is with its prediction of a certain social group.

BPSN (Background Positive, Subgroup Negative) AUC measures the model's false-positive rates regarding the target groups: $AUC(D^+ + D_g^-)$. The higher the score is, the less a model is likely to confuse non-toxic sentences whose target is the specific subgroup and toxic sentences whose target is one of the other groups.

BNSP (Background Negative, Subgroup Positive) AUC measures the model's false-negative rates regarding the target groups: $AUC(D^- + D_g^+)$. The higher the score is, the less the model is likely to confuse non-toxic sentences whose target is the specific group and toxic sentences whose target is one of the other groups.

We calculate GMB (Generalized Mean of Bias)[1] of the three metrics as the final scores to combine those ten scores of each of the metrics into one overall measure according to the HateXplain benchmark. The formula is: $M_p(m_s) = (\frac{1}{N}\sum_{s=1}^{N} m_s^p)^{\frac{1}{p}}$, where M_p means the pth power-mean function, m_s is one of the bias metrics m calculated for a specific subgroup s, and N is the number of subgroups which is 10 in this paper.

The explainability-based metrics evaluate how much the model is explainable. HateXplain follows ERASER (DeYoung et al., 2019), which is a benchmark for the evaluation of explainability of an NLP model based on rationales. The metrics are divided into Plausibility and Faithfulness. Plausibility refers to how the model's rationale matches the human rationale. Plausibility can be considered both discrete selection and soft selection. For discrete selection, We convert token scores to binary values by more than some threshold(here 0.5). Then, We measures IOU F1 score and Token F1 score. For soft selection, We constructed AUPRC by sweeping a threshold over token scores.

Faithfulness evaluates the influence of the model rationale on its prediction result and consists of Comprehensiveness and Sufficiency. Comprehensiveness assumes the model prediction is less confidence when rationales are removed. This metric can be calculated: $m(x_i)_j - m(x_i \backslash r_i)_j$. $m(x_i)_j$ is the prediction probability of the corresponding class j with an input sentence x_i by the model m. And $x_i \backslash r_i$ is the sentence manipulated by removing the predicted rationale tokens r_i from x_i.[2] The higher a score, the more influential the model's rationales in its prediction. Sufficiency captures the extent to which extracted rationales are acceptable for a model to make a prediction: $m(x_i)_j - m(r_i)_j$. A low score of this metric means that the rationales are adequate in the prediction.

In addition, for the HateXplain benchmark, the scores are calculated based on the attention scores of the last layer or by using the LIME method (Ribeiro et al., 2016). The former is marked as [Att], and the latter is [LIME] in Table 2. (DeYoung et al., 2019) and (Mathew et al., 2020) contain more detailed explanations.

[1] https://www.kaggle.com/competitions/jigsaw-unintended-bias-in-toxicity-classification/overview/evaluation

[2] We select the top 5 tokens to remove based on the average length of human-annotated rationale labels in the dataset according to HateXplain benchmark.

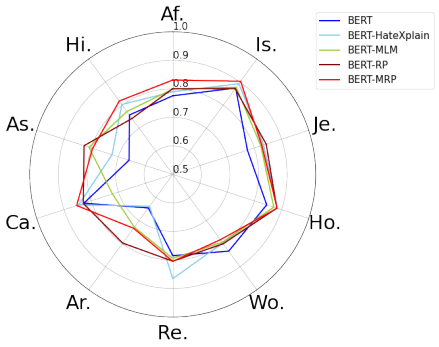

Figure 3: The Subgroup scores among bias-based metrics for each of ten target groups. The target group labels are 'African', 'Islam', 'Jewish', 'Homosexual', 'Women', 'Refugee', 'Arab', 'Caucasian', 'Asian', and 'Hispanic' in clockwise direction respectively. The BPSN and BNSP scores are attached in Appendix.

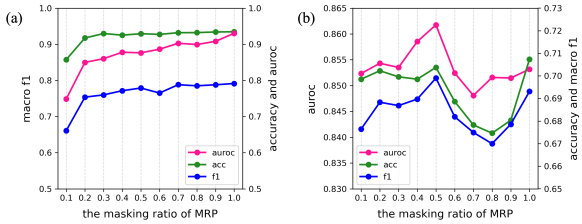

Figure 4: Classification test scores of the proposed models according to masking ratio in MRP. (a) is for token classification after training on MRP in the first stage, and (b) is for hate speech detection in the final stage. The case of masking 100% of tokens is the same as BERT-RP.

4.3 Models and Experimental settings

The evaluated models in Table 1 and Table 2 are as follows. All models are based on a BERT-base-uncased model for a pre-trained model and fine-tuned on hate speech detection. **BERT** in the tables is simply finetuned on hate speech detection with a fully-connected layer as a head for the three-class classification described above.

BERT-HateXplain uses attention supervision in addition to BERT. It matches the last attention values corresponding to the CLS token to the rationale used as ground truth attention. With this, the CLS token takes additional rationale-based attention information for the prediction. The loss is the summation of this attention loss and the detection loss. The results of BERT and BERT-HateXplain are the same as those presented in (Mathew et al., 2020).

BERT-MLM is evaluated to compare the effectiveness of pre-finetuning tasks. Training a pre-trained NLP model with MLM using data of the

downstream task is frequently used for the model to understand the downstream data and improve its performance (Han and Eisenstein, 2019; Ben-David et al., 2020; Arefyev et al., 2021). It is implemented by simply masking 15% tokens of each input sentence.

BERT-MRP and **BERT-RP** are the proposed models in this paper. **BERT-MRP** is the model trained on MRP as an intermediate task and then finetuned on hate speech detection. The ratio of masked rationales per token is set to 50% of the entire rationale label. **BERT-RP** is trained on Rationale Prediction (RP), which is MRP when the ratio is set to 100%–masking all the rationales. It is functionally the same as token classification with the rationale label as ground truth.

BERT-MLM, BERT-RP, and BERT-MRP are directly trained in this study. The experimental settings are the same for all models and each training step. The learning rate is $5e{-}5$ during pre-finetuning and $2e{-}5$ for hate speech detection, which is the same as BERT-HateXplain. We use the RAdam optimizer and an Nvidia GeForce GTX 1050 graphics card.

4.4 Comparisons of results

Table 1 and Table 2 present the performances of the models. For all metrics, the proposed models–the two from the bottom–perform much better overall.

Performance-based metrics As summarized in Table 1, the proposed method outperforms the other methods. BERT-MRP shows the highest scores for Macro F1 and AUROC and BERT-RP for accuracy. The pre-finetuned models perform better than those that are not. It shows that the pre-finetuning process helps understand the data and allows enough time for tuning parameters for the target task, thereby improving performance. On the other hand, among the pre-finetuned models, the proposed models show better results than BERT-MLM. Furthermore, they outperform BERT-HateXplain, which also uses the rationale during training like ours. This shows that predicting the human rationale for hate as an intermediate task effectively implements a hate speech detection model.

Bias-based metrics For the model bias, the proposed models show superior results compared to other models. According to Table 1, the models trained using the rationales achieve higher scores than others in general. Given that the human rationales of hatred imply that hate speech is judged based on the context, not merely specific expressions, learning the rationale can exclude the model bias towards the particular words for the prediction. Withal, the proposed BERT-RP and BERT-MRP show better performance than BERT-HateXplain, even though they all utilize the rationale in training. BERT-MRP shows the best scores, and BERT-RP is the second-best for all three metrics. Additionally, Figure 3 shows the scores of the models for each of the ten major target groups. It can be seen that the proposed models score evenly high for all the target groups. While other models have significant differences in their bias depending on the groups, the proposed models have comparatively no correlation with them.

Explainability-based metrics In terms of explainability, the proposed models still perform better than others overall. For Plausibility, BERT-MRP achieves the best performance for all three metrics. It scores much higher than others because it is allowed to directly guess the human rationales during the intermediate training stage. For Faithfulness, BERT-HateXplain[LIME] shows the highest score for Comprehensiveness, and BERT-MLM[LIME] is the best for Sufficiency. However, these models do not reliably score well when considering all four scores obtained according to each of the two measurement methods: attention values or LIME. They show worse scores than BERT for the rest of the scores. On the other hand, BERT-MRP and BERT-RP offer stably high performance for all five explainability-based metrics.

Based on all these results, BERT-MRP and BERT-RP demonstrate the best performance overall for the three types of metrics. Thus, learning the human rationale as an intermediate task before training on hate speech detection seems effective for detection performance and model bias and its explainability. This framework contributes to better performance than the other–pre-finetuning on MLM as well as another way of using rationales.

BERT-MRP generally achieves better results than BERT-RP, wherein the intermediate task is basically the same as token classification. The plots in Figure 4 show the change in test scores according to the masking ratio in MRP. According to Figure 4(a), when more than 20% of all rationales were masked, there is no significant difference in the token classification performance, although the amount of loss decreases as the ratio decreases. When each model was re-trained for hate speech

No.	Model	Example sentence and Rationale	Label
Case 1 Explicit hate speech			
1	Human	imagine unironically believing the high iq kike meme	HS
	BERT-MRP	imagine un ##iro ##nical ##ly believing the high iq ki ##ke me ##me	HS
	BERT-RP	imagine un ##iro ##nical ##ly believing the high iq ki ##ke me ##me	HS
	BERT-HX	imagine un ##iro ##nical ##ly believing the high iq ki ##ke me ##me	HS
2	Human	<user> ffs these ghetto blacks are ignorant like cracker mountain hillbillies	HS
	BERT-MRP	<user> ff ##s these ghetto blacks are ignorant like crack ##er mountain hill ##bill ##ies	HS
	BERT-RP	<user> ff ##s these ghetto blacks are ignorant like crack ##er mountain hill ##bill ##ies	HS
	BERT-HX	<user> ff ##s these ghetto blacks are ignorant like crack ##er mountain hill ##bill ##ies	HS
Case 2 Implicit hate speech			
3	Human	yah christian island republic shall not have queers or witches on it if we find any death	HS
	BERT-MRP	ya ##h christian island republic shall not have queer ##s or witches on it if we find any death	HS
	BERT-RP	ya ##h christian island republic shall not have queer ##s or witches on it if we find any death	HS
	BERT-HX	ya ##h christian island republic shall not have queer ##s or witches on it if we find any death	NO
4	Human	you can not culturally enrich a moslem	HS
	BERT-MRP	you can not culturally en ##rich a mo ##sle ##m	HS
	BERT-RP	you can not culturally en ##rich a mo ##sle ##m	NO
	BERT-HX	you can not culturally en ##rich a mo ##sle ##m	NO
5	Human	<user> he is infected with jihadi virus he will spread it to others	HS
	BERT-MRP	<user> he is infected with jihad ##i virus he will spread it to others	HS
	BERT-RP	<user> he is infected with jihad ##i virus he will spread it to others	HS
	BERT-HX	<user> he is infected with jihad ##i virus he will spread it to others	OF

Table 3: The highlighted words of the human rationale and the rationale of the models with detection results. BERT-HX is BERT-HateXplain. In the label column, the ground truth is of humans and the remaining labels are the predictions of each model. HS is 'hate speech,' OF is 'offensive,' and NO is 'normal.' More examples are in Appendix.

detection, as shown in Figure 4(b), the case of 50% ratio in BERT-MRP achieved the best classification performance. As MRP is a method for inferring the rationale of a particular token based on surrounding tokens, the model can successfully learn the human rationale within the context. Learning parameters during this reasoning process based on context seems to effectively prevent biased prediction while still being explainable and consequently improves the detection performance substantially.

4.5 Qualitative results

Table 3 shows examples of detection results from models that use human rationale for their training. The visualized values as the model rationales are the LIME results used to measure the explainability-based scores. For the human ground truth, the average value per word of human-annotated rationales is expressed for each word. The darker the color, the higher the values.

It is relatively easy to judge explicit hate speech that includes clear derogatory expressions. As shown in Case 1 of Table 3, all the models perform well. The human rationale tends to focus on specific abusive words, and so does the rationale of each model. However, the rationales of the two proposed models match the ground truth better than BERT-HateXplain. Our method to train a model on a token classification-based task leads well the model to focus on human-like grounds in the sentence by directly learning the human rationale.

As in Case 2, the implicit hate speech with no aggressive expressions cannot be grasped through context. The human rationale thus tends to appear throughout the sentence. As this might make hate speech detection relatively challenging, the detection results of BERT-HateXplain or BERT-RP seem incorrect for some sentences. However,

BERT-MRP works accurately based on its rationale that is much more similar to human's than others. Meanwhile, BERT-HateXplain shows a low matching rate of the rationale when the human rationale is throughout the sentence. It uses the human rationale as the ground truth attention, and if there is no difference in the human rationale across tokens, the ground truth could become similar to that of any normal sentence represented by uniform values. This affects the model's explainability and may lead to incorrect detection results. The proposed method does not cause that problem. It gets the distinguishable ground truth from normal ones and assigns it as labels to tokens. On the other hand, the rationale of BERT-MRP matches the ground truth better than that of BERT-RP. MRP requires more context-awareness ability when predicting the masked token by allowing the model to consider the abusiveness of surrounding words that are provided corresponding human rationale. This offers robust detection performance, even when it is necessary to understand the context.

5 Conclusion

This paper presents a method to implement a hate speech detection model considering bias and explainability. We adopt a framework to finetune a pre-trained language model in two stages. As the intermediate task, we propose Masked Rationale Prediction (MRP), which predicts masked rationales for some tokens with the additional rationale information of the remaining surrounding tokens. With this, the model learns to identify abusiveness for each token and the human reasoning process based on context. The trained model by MRP is finetuned again on hate speech detection.

As a result, across quantitative and qualitative evaluations, the proposed model shows state-of-the-art performance in bias and explainability, as well as the detection result. And the examples demonstrate its robustness in detecting hate speech, whether explicit or implicit, based on its superior explainability. Meanwhile, we experimented with only BERT as the pre-trained model to compare our method with base models. But any other transformer encoder-based model can be easily applied, which can be taken as future work.

Acknowledgements

This work was supported by the Ministry of Education of the Republic of Korea and the National Research Foundation of Korea (NRF) (NRF-2021S1A5A2A03065899), and also by the NRF of Korea grant funded by the Korean government (MSIT) (NRF-2022R1A2C1007434).

References

Armen Aghajanyan, Anchit Gupta, Akshat Shrivastava, Xilun Chen, Luke Zettlemoyer, and Sonal Gupta. 2021. Muppet: Massive multi-task representations with pre-finetuning. *arXiv preprint arXiv:2101.11038*.

Nikolay Arefyev, Dmitrii Kharchev, and Artem Shelmanov. 2021. Nb-mlm: Efficient domain adaptation of masked language models for sentiment analysis. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 9114–9124.

Pinkesh Badjatiya, Shashank Gupta, Manish Gupta, and Vasudeva Varma. 2017. Deep learning for hate speech detection in tweets. In *Proceedings of the 26th international conference on World Wide Web companion*, pages 759–760.

Yujia Bao, Shiyu Chang, Mo Yu, and Regina Barzilay. 2018. Deriving machine attention from human rationales. *arXiv preprint arXiv:1808.09367*.

Eyal Ben-David, Carmel Rabinovitz, and Roi Reichart. 2020. Perl: Pivot-based domain adaptation for pre-trained deep contextualized embedding models. *Transactions of the Association for Computational Linguistics*, 8:504–521.

Daniel Borkan, Lucas Dixon, Jeffrey Sorensen, Nithum Thain, and Lucy Vasserman. 2019. Nuanced metrics for measuring unintended bias with real data for text classification. In *Companion proceedings of the 2019 world wide web conference*, pages 491–500.

Thomas Davidson, Debasmita Bhattacharya, and Ingmar Weber. 2019. Racial bias in hate speech and abusive language detection datasets. *arXiv preprint arXiv:1905.12516*.

Fabio Del Vigna12, Andrea Cimino23, Felice Dell'Orletta, Marinella Petrocchi, and Maurizio Tesconi. 2017. Hate me, hate me not: Hate speech detection on facebook. In *Proceedings of the First Italian Conference on Cybersecurity (ITASEC17)*, pages 86–95.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*.

Jay DeYoung, Sarthak Jain, Nazneen Fatema Rajani, Eric Lehman, Caiming Xiong, Richard Socher, and Byron C Wallace. 2019. Eraser: A benchmark to evaluate rationalized nlp models. *arXiv preprint arXiv:1911.03429*.

Xiaochuang Han and Jacob Eisenstein. 2019. Unsupervised domain adaptation of contextualized embeddings for sequence labeling. *arXiv preprint arXiv:1904.02817*.

Braden Hancock, Martin Bringmann, Paroma Varma, Percy Liang, Stephanie Wang, and Christopher Ré. 2018. Training classifiers with natural language explanations. In *Proceedings of the conference. Association for Computational Linguistics. Meeting*, volume 2018, page 1884. NIH Public Access.

Jeremy Howard and Sebastian Ruder. 2018. Universal language model fine-tuning for text classification. *arXiv preprint arXiv:1801.06146*.

Tao Lei, Regina Barzilay, and Tommi Jaakkola. 2016. Rationalizing neural predictions. *arXiv preprint arXiv:1606.04155*.

Hui Liu, Qingyu Yin, and William Yang Wang. 2018. Towards explainable nlp: A generative explanation framework for text classification. *arXiv preprint arXiv:1811.00196*.

Thomas Mandl, Sandip Modha, Prasenjit Majumder, Daksh Patel, Mohana Dave, Chintak Mandlia, and Aditya Patel. 2019. Overview of the hasoc track at fire 2019: Hate speech and offensive content identification in indo-european languages. In *Proceedings of the 11th forum for information retrieval evaluation*, pages 14–17.

Binny Mathew, Punyajoy Saha, Seid Muhie Yimam, Chris Biemann, Pawan Goyal, and Animesh Mukherjee. 2020. Hatexplain: A benchmark dataset for explainable hate speech detection. *arXiv preprint arXiv:2012.10289*.

Rohan Kshirsagar1 Tyrus Cukuvac1 Kathleen McKeown and Susan McGregor. 2018. Predictive embeddings for hate speech detection on twitter. *EMNLP 2018*, page 26.

Jason Phang, Thibault Févry, and Samuel R Bowman. 2018. Sentence encoders on stilts: Supplementary training on intermediate labeled-data tasks. *arXiv preprint arXiv:1811.01088*.

Yada Pruksachatkun, Jason Phang, Haokun Liu, Phu Mon Htut, Xiaoyi Zhang, Richard Yuanzhe Pang, Clara Vania, Katharina Kann, and Samuel R Bowman. 2020. Intermediate-task transfer learning with pretrained models for natural language understanding: When and why does it work? *arXiv preprint arXiv:2005.00628*.

Alec Radford, Karthik Narasimhan, Tim Salimans, and Ilya Sutskever. 2018. Improving language understanding by generative pre-training.

Tharindu Ranasinghe, Marcos Zampieri, and Hansi Hettiarachchi. 2019. Brums at hasoc 2019: Deep learning models for multilingual hate speech and offensive language identification. In *FIRE (working notes)*, pages 199–207.

Marco Tulio Ribeiro, Sameer Singh, and Carlos Guestrin. 2016. " why should i trust you?" explaining the predictions of any classifier. In *Proceedings of the 22nd ACM SIGKDD international conference on knowledge discovery and data mining*, pages 1135–1144.

Maarten Sap, Dallas Card, Saadia Gabriel, Yejin Choi, and A Noah Smith. 2019. The risk of racial bias in hate speech detection. In *ACL*.

Yunsheng Shi, Zhengjie Huang, Shikun Feng, Hui Zhong, Wenjin Wang, and Yu Sun. 2020. Masked label prediction: Unified message passing model for semi-supervised classification. *arXiv preprint arXiv:2009.03509*.

Ameya Vaidya, Feng Mai, and Yue Ning. 2020. Empirical analysis of multi-task learning for reducing identity bias in toxic comment detection. In *Proceedings of the International AAAI Conference on Web and Social Media*, volume 14, pages 683–693.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. *Advances in neural information processing systems*, 30.

Seongjun Yun, Minbyul Jeong, Raehyun Kim, Jaewoo Kang, and Hyunwoo J Kim. 2019. Graph transformer networks. *Advances in neural information processing systems*, 32.

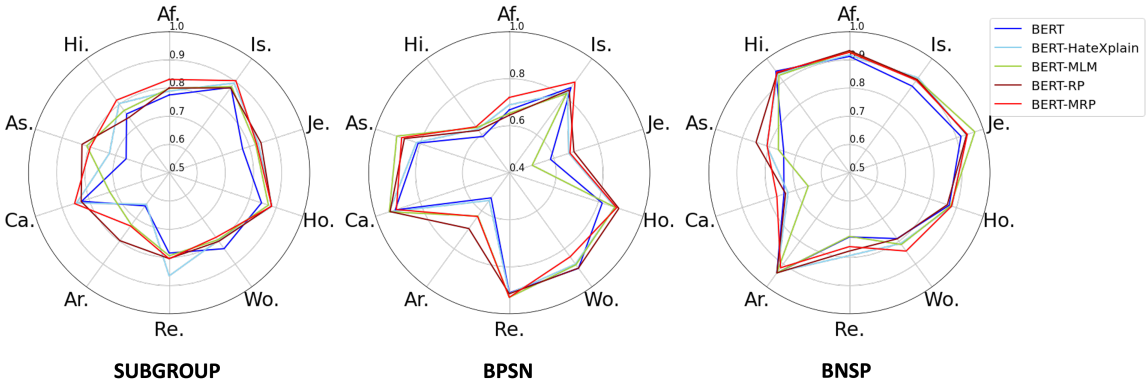

Figure A1: The Bias-based scores for each 10 target groups. The target group labels are 'African', 'Islam', 'Jewish', 'Homosexual', 'Women', 'Refugee', 'Arab', 'Caucasian', 'Asian', and 'Hispanic' in clockwise direction respectively.

Model	Example sentence and Rationale	Label
Human	if you ' re an army that is republican you are not welcome here we hate you	HS
BERT-MRP	if you ' re an army that is republican you are not welcome here we hate you	HS
BERT-RP	if you ' re an army that is republican you are not welcome here we hate you	NO
BERT-HX	if you ' re an army that is republican you are not welcome here we hate you	NO
Human	the german regime is more scared of women like this than than are of any moslem terrorist	HS
BERT-MRP	the german regime is more scared of women like this than than are of any mo ##sle ##m terrorist	HS
BERT-RP	the german regime is more scared of women like this than than are of any mo ##sle ##m terrorist	HS
BERT-HX	the german regime is more scared of women like this than than are of any mo ##sle ##m terrorist	NO
Human	i poisoned sergei skripal he was a faggot anyway	HS
BERT-MRP	i poisoned sergei sk ##rip ##al he was a fa ##gg ##ot anyway	HS
BERT-RP	i poisoned sergei sk ##rip ##al he was a fa ##gg ##ot anyway	HS
BERT-HX	i poisoned sergei sk ##rip ##al he was a fa ##gg ##ot anyway	OF
Human	<user> would have pressed that bitch and beat the faggot out of him	HS
BERT-MRP	<user> would have pressed that bitch and beat the fa ##gg ##ot out of him	HS
BERT-RP	<user> would have pressed that bitch and beat the fa ##gg ##ot out of him	HS
BERT-HX	<user> would have pressed that bitch and beat the fa ##gg ##ot out of him	OF
Human	massive muzrat espionage	HS
BERT-MRP	massive mu ##z ##rat espionage	HS
BERT-RP	massive mu ##z ##rat espionage	HS
BERT-HX	massive mu ##z ##rat espionage	HS
Human	enjoy your kike faith	HS
BERT-MRP	enjoy your ki ##ke faith	HS
BERT-RP	enjoy your ki ##ke faith	HS
BERT-HX	enjoy your ki ##ke faith	HS

Table A1: The highlighted words of the human rationale and the rationale of the models with detection results in 'hate speech' sentences.

Model	Example sentence and Rationale	Label
Human	“ i don ’ t have to respect women i ’ m gay ” kevin	OF
BERT-MRP	“ i don ’ t have to respect women i ’ m gay ” kevin	OF
BERT-RP	“ i don ’ t have to respect women i ’ m gay ” kevin	NO
BERT-HX	“ i don ’ t have to respect women i ’ m gay ” kevin	NO
Human	logan paul is a fucking bozo	OF
BERT-MRP	logan paul is a fucking bo ##zo	OF
BERT-RP	logan paul is a fucking bo ##zo	NO
BERT-HX	logan paul is a fucking bo ##zo	NO
Human	<user> <user> there are literally nazis all over all the time rigjt now throw a rock you hit a nazi	OF
BERT-MRP	<user> <user> there are literally nazis all over all the time rig ##j ##t now throw a rock you hit a nazi	OF
BERT-RP	<user> <user> there are literally nazis all over all the time rig ##j ##t now throw a rock you hit a nazi	OF
BERT-HX	<user> <user> there are literally nazis all over all the time rig ##j ##t now throw a rock you hit a nazi	NO
Human	all my friends and peers are being openly racist towards asians and this bitch is not having it	OF
BERT-MRP	all my friends and peers are being openly racist towards asian ##s and this bitch is not having it	OF
BERT-RP	all my friends and peers are being openly racist towards asian ##s and this bitch is not having it	OF
BERT-HX	all my friends and peers are being openly racist towards asian ##s and this bitch is not having it	OF
Human	hey nazi dirtbag is your black bull keeping you occupied	OF
BERT-MRP	hey nazi dirt ##bag is your black bull keeping you occupied	OF
BERT-RP	dirt ##bag is your black bull keeping you occupied	OF
BERT-HX	hey nazi dirt ##bag is your black bull keeping you occupied	OF
Human	white bitch pink pussy	OF
BERT-MRP	hey white bitch pink pussy	OF
BERT-RP	white bitch pink pussy	OF
BERT-HX	white bitch pink pussy	OF

Table A2: The highlighted words of the human rationale and the rationale of the models with detection results in ’offensive’ sentences.

Domain Classification-based Source-specific Term Penalization for Domain Adaptation in Hate-speech Detection

Tulika Bose † **Nikolaos Aletras** ‡ **Irina Illina** † **Dominique Fohr** †
† Universite de Lorraine, CNRS, Inria, LORIA, F-54000 Nancy, France
‡University of Sheffield, United Kingdom
{tulika.bose, illina, dominique.fohr}@loria.fr
n.aletras@sheffield.ac.uk

Abstract

Warning: *this paper contains content that may be offensive and distressing.*

State-of-the-art approaches for hate-speech detection usually exhibit poor performance in out-of-domain settings. This occurs, typically, due to classifiers overemphasizing source-specific information that negatively impacts its domain invariance. Prior work has attempted to penalize terms related to hate-speech from manually curated lists using feature attribution methods, which quantify the importance assigned to input terms by the classifier when making a prediction. We, instead, propose a domain adaptation approach that automatically extracts and penalizes source-specific terms using a domain classifier, which learns to differentiate between domains, and feature-attribution scores for hate-speech classes, yielding consistent improvements in cross-domain evaluation.

1 Introduction

While recent state-of-the-art hate-speech classifiers (Ayo et al., 2021; D'Sa et al., 2020; Mozafari et al., 2019) yield impressive performance on in-domain held-out instances, they suffer when evaluated on out-of-domain settings (Yin and Zubiaga, 2021; Arango et al., 2019; Swamy et al., 2019; Karan and Šnajder, 2018). The distributions across corpora/domains[1] change due to varying vocabulary, topics of discussion over time (Florio et al., 2020; Saha and Sindhwani, 2012), data bias caused by sampling strategies (Wiegand et al., 2019) and different hate-targets. This is concerning since curating new data resources for hate-speech involves substantial time and effort (Poletto et al., 2019; Malmasi and Zampieri, 2018). This calls for strategies, like Domain Adaptation (DA) approaches, that can adapt models trained on existing labeled resources to a new target domain that lacks class-labels.

However, research on DA in hate-speech is limited (Sarwar and Murdock, 2022; Bashar et al., 2021; Bose et al., 2021). Typically, vanilla classifiers tend to learn more from domain-specific features (Ye et al., 2021; Wiegand et al., 2019) than domain-invariant features, resulting in poor out-of-domain performance. For instance, Wiegand et al. (2019) show that in a hate-speech dataset (Waseem and Hovy, 2016), neutral domain-specific terms, like '*football*', '*commentator*', etc., discussing the role of women in sports, are highly correlated with the hate label, restricting its generalizability. Thus, it is worth minimizing the importance of such terms for improving cross-domain performance.

Recently, feature attributions – methods for extracting post-hoc model explanations, have been used to align features with prior domain knowledge (Rieger et al., 2020; Adebayo et al., 2020). These provide importance scores to the input terms as per their contribution towards the model prediction (Lundberg and Lee, 2017). For instance, Liu and Avci (2019); Kennedy et al. (2020) reduce the oversensitivity of classifiers on a curated list of identity terms (e.g. *muslims*, *gay*) by penalizing their importance. However, newly emerging social-media terms (Grieve et al., 2018) may render such lists non-exhaustive. Yao et al. (2021) do not use any list but they require human-provided refinement advice as inputs. Chrysostomou and Aletras (2022a) further show that post-hoc explanation methods might not provide faithful explanation in out-of-domain settings. The contemporaneous work by Attanasio et al. (2022) and Bose et al. (2022) reduce lexical overfitting automatically with entropy-based attentions and feature attributions, respectively. While cross-domain classification performance across different datasets is not studied in the former, the latter needs some labeled target instances to identify the over-fitted terms.

In the task of detecting objects in images, Zunino et al. (2021) use a domain classifier, trained to dif-

[1]We use the terms 'corpus' and 'domain' interchangeably.

Proceedings of the 29th International Conference on Computational Linguistics, pages 6656–6666
October 12–17, 2022.

ferentiate between domains, to visually identify the irrelevant background information to be domain-specific. Thus, they enforce the model explanations to align with the ground-truth annotations highlighting the objects in the image. Inspired by this, we propose a new DA approach in hate-speech employing a domain classifier, but without having access to such annotations for aligning the attribution scores.

We hypothesize that domain-specific terms that are simultaneously predictive of the hate-speech labels are instrumental in restricting the domain invariance of the hate-speech classifier. To this end, we employ a domain classifier to automatically extract the terms that help in identifying the source domain compared to the *unlabeled* target domain, and feature-attribution scores to identify the subset important for hate-speech classification from the source. *Our method, through penalization of these terms, automatically enforces the source domain classifier to focus on domain-invariant content.* Compared to approaches transforming high-dimensional intermediate representations to reduce the domain discrepancy, such as domain adversarial learning (Ryu and Lee, 2020; Tzeng et al., 2017), our approach makes the adaptation more explainable, while improving the overall cross-domain performance compared to prior-approaches.

2 Proposed Approach

Given training data from a labeled source domain D_S^{train} and an unlabeled target domain D_T^{train}, our approach for DA in hate-speech involves 2 steps: (i) extraction of source-specific terms and (ii) reducing the importance of these terms. Our setting is similar to Ben-David et al. (2020) and Ryu and Lee (2020).

2.1 Extraction of Source-specific Terms

Domain classification To identify source-specific terms, we first train a binary domain classifier using D_S^{train} and D_T^{train} that learns to identify whether a candidate instance comes from the source or the target domain. For this, we use a simple Logistic Regression (LR) with bag-of-words, as it is inherently interpretable. We then use its feature weights to extract the top N most important terms for predicting the source domain class. Each term is tokenized with the BERT (Devlin et al., 2019) WordPiece tokenizer for compatibility with transformer models. The top N terms obtained through domain classification are denoted as S_{LR}.

Attribution-based term ranking Intuitively, the terms from S_{LR} that also contribute highly to the hate-speech labels, are likely to restrict generalization to the target as they could potentially reduce the importance assigned by the classifier to domain-invariant hate-speech terms. Thus, we extract only those source-specific terms that are highly correlated with the labels, given the binary classification task of *hate* versus *non-hate*.

To this end, we first continue pre-training BERT on the unlabeled D_T^{train} using the Masked Language Model (MLM) objective for incorporating the language-variations of the target domain, following Glavaš et al. (2020). We then perform supervised classification on D_S^{train} using this MLM trained model. After every epoch, we obtain 2 ranked lists of terms for the two classes, sorted in the order of decreasing importance. We construct the lists using feature attribution methods that yield instance-level attribution scores ins-atr$_{te}^{j}$ per term te in an instance j – a higher score indicating a higher contribution to the predicted class. We discard the scores of stop-words and the infrequent terms, and normalize ins-atr$_{te}^{j}$ using the sigmoid function. For obtaining a corpus-level class-specific attribution score cp-atr$_{te}^{c}$ per term te and per class c, we perform a corpus-level average of all the ins-atr$_{te}^{j}$ for every c using Equation 1.

$$\text{cp-atr}_{te}^{c} = \frac{\sum_{j=1}^{|D_S^{train}|} \mathbb{1}_{\hat{y}_j=c} \text{ins-atr}_{te}^{j} \forall \text{occurrence of } te \text{ in j}}{\sum_{j=1}^{|D_S^{train}|} \mathbb{1}_{\hat{y}_j=c} \#(\text{occurrence of } te \text{ in j})} \quad (1)$$

Here $c \in \{\text{hate, non-hate}\}$, $\hat{y}$ is the predicted class and $\mathbb{1}$ is the indicator function. We sort the scores cp-atr$_{te}^{c}$ for all te to obtain the highest attributed (i.e. most important) term per class to the lowest, yielding the ranked lists of terms per class, given by CP $= [cp\text{-}hate, cp\text{-}non\text{-}hate]$.

We extract the source-specific terms te^S that are common to both S_{LR} and the top M terms from CP, i.e. te$^S = [te \in S_{LR} \;\; \& \;\; te \in top_M(\text{CP})]$. These steps are repeated after every epoch. Note that the list S_{LR} remains constant across the epochs, as it is independent to the hate-speech classification task.

2.2 Penalization of Source-specific Terms

We hypothesize that penalizing te^S obtained from the previous epoch during the next epoch should reduce the importance of terms that are both (i) domain-specific and (ii) contribute highly to the source labels, and thus, help learn from domain invariant terms. We minimize the attribution scores

for te^S, with L_2 penalization, in Equation 2.

$$\mathcal{L} = \mathcal{L}^{'} + \lambda \mathcal{L}_{\text{atr}}; \mathcal{L}_{\text{atr}} = \sum_{\text{t} \in \text{te}^{\mathcal{S}}} \phi\left(\text{t}\right)^2; \text{t} \in \text{te}^{\mathcal{S}} \quad (2)$$

Here $\mathcal{L}^{'}$ is the classification loss and $\mathcal{L}_{\text{atr}}$ is the attribution loss. λ controls the strength of penalization, and $\phi\left(\text{t}\right)$ is the attribution score for t.

We experiment with two variations: (i) **Domspec:** penalizing only the terms in te^S; (ii) **Comb**: penalizing the combination of te^S and the terms from Liu and Avci (2019); Kennedy et al. (2020).

We use two different feature attribution methods that have been widely used in recent studies (Chrysostomou and Aletras, 2021, 2022b): (i) **Scaled Attention** ($\alpha\nabla\alpha$) (Serrano and Smith, 2019), which scales attention weights α by their corresponding gradients $\nabla\alpha_i = \frac{\delta\hat{y}}{\delta\alpha_i}$, where $\hat{y}$ is the predicted label, and is shown to work better than using only the attention weights; (ii) **DeepLIFT/ DL** (Shrikumar et al., 2017) that assigns scores based on the difference between activation of each neuron and a reference activation (zero embedding vector). Note that although Liu and Avci (2019) have used the Integrated Gradients (IG) (Sundararajan et al., 2017), we use DL as it is most often a good and a faster approximation of IG (Ancona et al., 2018).

3 Experimental Setup

3.1 Data

We use three standard hate-speech datasets, namely, *Waseem* (Waseem and Hovy, 2016), *HatEval* (Basile et al., 2019) and *Vidgen* (Vidgen et al., 2021). Following Wiegand et al. (2019); Swamy et al. (2019), we perform hate/non-hate classification across domains. We use the standard splits available for *HatEval* (42.1% hate; train: 8993[2], val: 1000; test: 3000) and *Vidgen* (54.4% hate; train: 32497, val: 1016, test: 4062). We subsample the *Vidgen* validation set by 25% to get 1016 samples, making its size similar to the other datasets. We split *Waseem* (26.8% hate) randomly into train (80%; 8720), validation (10%; 1090) and test (10%; 1090) sets, as no standard splits are available.

We present the top ten most frequent terms in these datasets in Table 1. The *Waseem* dataset is known to comprise a high proportion of implicit hate (Wiegand et al., 2019), which are subtle expressions of hate without the use of profanity. This is also evident in the most frequent terms from this dataset. In Table 1, *#mkr* refers to a cooking show which frequently results in sexist comments targeted towards the participating women. *HatEval* involves hate against women and immigrants. Many hateful tweets against immigrants occurred in the context of the US-Mexico border issues with the hashtag *#buildthewall*. The *Vidgen* dataset is collected through a dynamic data creation process with a human-and-model-in-the-loop strategy, unlike *HatEval* and *Waseem* datasets that are sampled from Twitter. In particular, the *Vidgen* dataset involves hate against many different target groups or identity terms, with a wide variety of topics and hateful forms. See Appendix A for further details on the datasets.

Dataset	Frequent terms in the datasets
Waseem	#mkr, #notsexist, kat, women, like, andre, get, people, one, think
HatEval	b*tch, women, refugees, #buildthewall, immigrant, immigration, illegal, men, migrants, h*e
Vidgen	people, black, women, f*cking, like, love, think, white, get, want

Table 1: Top ten most frequent terms in the datasets after removing the stop-words.

3.2 Baselines

We compare our work with approaches that penalize **(i)** pre-defined terms in Convolutional Neural Networks-based Liu and Avci (2019)[3]; **(ii)** (a) the identity terms in the top features of a bag-of-words Logistic Regression in BERT-based Kennedy et al. (2020)[4] (b) all the terms listed by Kennedy et al. (2020); **(iii)** terms extracted automatically by Attanasio et al. (2022); **(iv)** combination of terms from (i) and (ii,b) within BERT, and call this **Pre-Def**. We do not compare with Bose et al. (2022) as they use labeled target instances for term-extraction, which does not allow a fair comparison.

Further, we experiment with the Vanilla baseline (**Van-MLM-FT**), where the pre-trained BERT is adapted to D_T^{train} using the MLM objective, followed by a supervised fine-tuning on D_S^{train}. We also assess different DA methods from the sentiment classification task, namely, **BERT PERL** (Pivot-based Encoder Representation of Language)

[2]The instances containing only URLs are removed, decreasing the number of train instances from 9000 to 8993.

[3]with Integrated Gradients (Sundararajan et al., 2017)

[4]with Sampling and Occlusion (Jin et al., 2020)

Approaches	H →V	V →H	H →W	W →H	V →W	W →V	Average
BERT Van-MLM-FT	56.6±1.3	66.2±1.2	70.0±2.5	50.9±2.1	61.4±2.4	43.5±1.9	58.1
Liu and Avci (2019)	45.1±4.5	59.5±0.7	57.2±3.8	52.6*±0.8	57.1±2.7	39.6±2.0	51.9
MLM + Kennedy et al. (2020) (a)	55.4±2.0	65.5±0.8	64.1±1.4	54.4*±1.3	59.2±1.8	44.5±2.9	57.2
MLM + Kennedy et al. (2020) (b)	54.9±2.9	65.7±0.9	67.3±1.2	54.3*±2.2	62.3±2.7	46.6±3.5	58.5
BERT PERL	54.1±0.7	60.0±0.6	60.1±2.0	**55.2***±0.7	55.5±1.0	37.8±1.2	53.8
BERT-AAD	56.6±1.3	53.9±3.5	68.8±2.5	50.7±1.4	48.3±4.7	**53.0***±1.7	55.2
HATN	48.4±1.6	59.1±0.4	59.7±2.9	51.4±1.8	60.0±2.6	45.4±2.7	54.0
MLM + Sarwar and Murdock (2022)	55.0±1.9	66.2±2.0	68.8±1.1	48.2±3.1	57.9±1.3	36.2±1.1	55.4
MLM + Attanasio et al. (2022)	54.9±1.6	66.5±1.4	64.1±5.0	52.4*±3.7	62.5±0.8	43.5±2.3	57.3
MLM + χ^2-test	57.9±1.6	67.1±1.7	69.8±0.8	48.2±3.1	60.4±2.8	44.1±3.4	57.9
Pre-def ($\alpha\nabla\alpha$)	**58.9***±0.7	67.4±1.5	71.3±1.0	48.9±4.0	60.0±2.0	46.5±4.9	58.8
Dom-spec ($\alpha\nabla\alpha$)	58.3±1.8	66.8±0.7	70.1±1.8	52.3*±3.0	60.8±2.2	46.9*±2.5	59.2
Comb (Dom-spec + Pre-def) ($\alpha\nabla\alpha$)	58.7*±2.1	**67.7**±1.0	70.9±1.0	51.5±2.1	59.8±1.5	45.9±3.1	59.1
Pre-def (DL)	58.5*±1.4	66.5±1.3	70.3±1.7	51.2±1.7	**70.3***±0.5	42.7±2.0	59.9
Dom-spec (DL)	58.8*±0.6	66.4±1.2	**72.2**±1.4	52.9*±1.9	63.6*±2.0	48.8*±4.7	60.5
Comb (Dom-spec + Pre-def) (DL)	58.4±1.4	66.7±1.0	71.3±0.9	51.1±2.2	69.5*±2.2	46.6±1.9	**60.6**

Table 2: Macro-F1 (±std-dev) on source →target pairs. H : HatEval, V : Vidgen, W : Waseem. **Bold** denotes the best score and underline the second best in each column . * denotes statistically significant improvement compared to Van-MLM-FT with paired bootstrap (Dror et al., 2018; Efron and Tibshirani, 1993), 95% confidence interval.

(Ben-David et al., 2020) that adopts the MLM objective of BERT to perform pivot-based fine-tuning; **BERT-AAD** (Adversarial Adaptation with Distillation) (Ryu and Lee, 2020) that performs domain adversarial training; **HATN** (Hierarchical Attention Transfer Network) (Li et al., 2018, 2017) that extracts pivots using a domain adversarial approach.

We evaluate a data-augmentation-based approach (Sarwar and Murdock, 2022) for DA in hate-speech. For a fair comparison, we use the BERT as the underlying model in this approach. Finally, we apply the **χ^2-test** with 1 degree of freedom and Yate's correction (Kilgarriff, 2001), penalizing the terms from D_S^{train}, using their DL scores, for which the null hypothesis of both D_S^{train} and D_T^{train} being random samples of the same larger population, is rejected with 95% confidence. We initialize all the BERT models with MLM adaptation on the target, except for PERL and AAD, which inherently adapts to the target.

3.3 Model training

We train all the models on D_S^{train}, use a small amount of the labeled D_T^{val} only for model-selection and hyper-parameter tuning (see Appendix B) , following Dai et al. (2020); Maharana and Bansal (2020), and evaluate on D_T^{test}.

4 Results

4.1 Discussion

Table 2 displays the macro-F1 scores obtained, in cross-domain settings, averaged across five randomly initialized runs. We use macro-F1 as penalizing te^S corrects the mis-classifications for both the hate and non-hate classes across domains. We observe an overall performance drop, compared to Van MLM-FT, with the DA approaches, originally proposed for sentiment classification, namely, BERT PERL, BERT-AAD and HATN. This also agrees with Bose et al. (2021), who analyze the extracted pivots – terms that are both frequent across domains as well as important for classification with respect to the source – and find them to be sub-optimal for DA in hate-speech. The approach by Sarwar and Murdock (2022) also displays an overall drop. They augment the source domain by substituting relevant terms from a different negative emotion dataset with tagged hate-speech related terms from the target domain. We observe that the augmented instances are often incomprehensible after such substitution.

Dom-spec yields improvements over all the baselines using both $\alpha\nabla\alpha$ and DL, both independently and in combination (Comb) with Pre-def, where Comb achieves the highest overall performance with DL: 60.6. With DL, Dom-spec yields significantly improved performance in 4/6 cases, compared to 2/6 with Pre-def (DL). This is apparently due to the penalization of relevant source-specific terms that have wider coverage compared to the pre-defined terms in Pre-def. Since the entropy-based attention regularization by Attanasio et al. (2022) do not use the target domain unlabeled instances for term-extraction, it may not be optimal for cross-domain settings. The large improvement with Pre-def (DL) for *Vidgen →Waseem* (70.3) could be attributed to the fact that *Vidgen* involves a wide variety of identity terms. Thus, penalizing the pre-defined identity terms might result in higher emphasis on more generalizable hate-speech content. While only this particular case drives the high average performance with Pre-def (DL), Dom-spec

Non-hate example from the test set of *HatEval* for *Waseem* →*HatEval*	
FP with Van-MLM-FT	**TN with Dom-spec (DL)**
Depression is a whole entire b*tch	Depression is a whole entire b*tch
Hate example from the test set of *Waseem* for *Vidgen* →*Waseem*	
FN with Van-MLM-FT	**TP with Dom-spec (DL)**
... good to talk with your wife but it is easier to say shut up n make me a sammich not sexist lol	... good to talk with your wife but it is easier to say shut up n make me a sammich not sexist lol

Table 3: Change in attributions with Dom-spec (DL).

(DL) performs well *consistently* and yields a higher average score (Dom-spec: 60.5, Comb: 60.6) compared to Pre-def.

As discussed by Wiegand et al. (2019), the *Waseem* dataset includes a high degree of implicit hate. Still, Dom-spec (DL) yields improvements on the *Waseem* dataset when using it as the target domain, compared to Van MLM-FT. This is reflected in the cases of *HatEval* →*Waseem* and *Vidgen* →*Waseem*. This is most likely because when the source domain-specific terms causing bias are penalized, the model is forced to learn more from the wider contextual meaning of the instances, rather than focusing on individual terms. We believe that this could possibly help in improving the detection of implicit hate in out of domain instances, at least to some extent. We leave further investigation in this direction for future work.

4.2 Qualitative Analysis

Table 3 displays examples of False Positives (FP) for *Waseem* →*HatEval* and False Negatives (FN) for *Vidgen* →*Waseem*, yielded by Van-MLM-FT for the respective target domain instances, which are correctly classified by Dom-spec (DL), where the hate class is the positive class. The darker the shades, the higher the attributions assigned by the source classifier. The examples suggest that penalizing source-specific terms results in placing more emphasis on the general contextual meaning of the out-of-domain instances such as 'depression' in the first example and 'wife...shut...make me a sammich' in the second.

Note that the terms in these examples from the target domain that receive reduced importance with Dom-spec, compared to Van-MLM-FT, may not be the same terms that are extracted and penalized. This is because the domain classification step results in obtaining terms that are more likely to be infrequent in the target domain. Rather, due to the penalization of source-specific terms, the source domain classifier learns to focus on the wider context of the instances. For example, we observe that in the case of *Waseem* →*HatEval*, the automatically extracted te^S includes terms related to the role of women in sports, such as {*sports, sexist, gaming, football, commentary, competition, ...*}. Note that Wiegand et al. (2019) also mention that these terms cause domain or topic bias in *Waseem*, restricting generalizability. See Appendix C for more examples.

5 Conclusion

We proposed a DA approach for automatic extraction and penalization of source domain-specific terms that have higher attributions towards the hate-speech labels, to improve cross-domain hate-speech detection. The results demonstrated consistent improvements on the target domain. These results should motivate further research on domain adaptation in hate-speech and building classifiers that can generalize well to the concept of hate. Finally, it would be interesting in applying our approach to other tasks such as rumor and misinformation detection (Mu and Aletras, 2020; Mu et al., 2022).

Ethical Considerations

This work serves as a means to build more robust hate-speech detection models that can make proper use of the existing curated hate-speech resources and adapt well on new resources or social-media comments, which have not been well-annotated due to time and cost constraints. The hate-speech resources used for the work are publicly available and cited appropriately, wherein the authors have discussed the sampling techniques and annotation guidelines in detail. The hate-speech examples presented in the paper are only intended for research purposes and better analysis of the models explored. The terms extracted and penalized in this work are not meant to be used off-the-shelf, but the approach should serve as a starting point for research on model-debugging and building more generalizable hate-speech classifiers.

Acknowledgments

This work was supported partly by the french PIA project "Lorraine Université d'Excellence", reference ANR-15-IDEX-04-LUE. Experiments presented in this article were carried out using the

Grid'5000 testbed, supported by a scientific interest group hosted by Inria and including CNRS, RENATER and several Universities as well as other organizations (see https://www.grid5000.fr). We thank the anonymous reviewers for their valuable feedback and suggestions.

References

Julius Adebayo, Michael Muelly, Ilaria Liccardi, and Been Kim. 2020. Debugging tests for model explanations. In *Advances in Neural Information Processing Systems*, volume 33, pages 700–712. Curran Associates, Inc.

Marco Ancona, Enea Ceolini, Cengiz Öztireli, and Markus Gross. 2018. Towards better understanding of gradient-based attribution methods for deep neural networks. In *6th International Conference on Learning Representations, ICLR 2018, Vancouver, BC, Canada, April 30 - May 3, 2018, Conference Track Proceedings*. OpenReview.net.

Aymé Arango, Jorge Pérez, and Barbara Poblete. 2019. Hate speech detection is not as easy as you may think: A closer look at model validation. In *Proceedings of the 42nd International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR'19, pages 45–54, New York, NY, USA. Association for Computing Machinery.

Giuseppe Attanasio, Debora Nozza, Dirk Hovy, and Elena Baralis. 2022. Entropy-based attention regularization frees unintended bias mitigation from lists. In *Findings of the Association for Computational Linguistics: ACL2022 (Forthcoming)*. Association for Computational Linguistics.

Femi Emmanuel Ayo, Olusegun Folorunso, Friday Thomas Ibharalu, Idowu Ademola Osinuga, and Adebayo Abayomi-Alli. 2021. A probabilistic clustering model for hate speech classification in twitter. *Expert Systems with Applications*, 173:114762.

Md Abul Bashar, Richi Nayak, Khanh Luong, and Thirunavukarasu Balasubramaniam. 2021. Progressive domain adaptation for detecting hate speech on social media with small training set and its application to covid-19 concerned posts. *Social Network Analysis and Mining*, 11(1):1–18.

Valerio Basile, Cristina Bosco, Elisabetta Fersini, Debora Nozza, Viviana Patti, Francisco Manuel Rangel Pardo, Paolo Rosso, and Manuela Sanguinetti. 2019. SemEval-2019 task 5: Multilingual detection of hate speech against immigrants and women in twitter. In *Proceedings of the 13th International Workshop on Semantic Evaluation*, pages 54–63, Minneapolis, Minnesota, USA. Association for Computational Linguistics.

Eyal Ben-David, Carmel Rabinovitz, and Roi Reichart. 2020. PERL: Pivot-based domain adaptation for pre-trained deep contextualized embedding models. *Transactions of the Association for Computational Linguistics*, 8:504–521.

Tulika Bose, Nikolaos Aletras, Irina Illina, and Dominique Fohr. 2022. Dynamically refined regularization for improving cross-corpora hate speech detection. In *Findings of the Association for Computational Linguistics: ACL2022 (Forthcoming)*. Association for Computational Linguistics.

Tulika Bose, Irina Illina, and Dominique Fohr. 2021. Unsupervised domain adaptation in cross-corpora abusive language detection. In *Proceedings of the Ninth International Workshop on Natural Language Processing for Social Media*, pages 113–122. Association for Computational Linguistics.

George Chrysostomou and Nikolaos Aletras. 2021. Improving the faithfulness of attention-based explanations with task-specific information for text classification. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 477–488, Online.

George Chrysostomou and Nikolaos Aletras. 2022a. An empirical study on explanations in out-of-domain settings. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 6920–6938, Dublin, Ireland. Association for Computational Linguistics.

George Chrysostomou and Nikolaos Aletras. 2022b. Flexible instance-specific rationalization of NLP models. AAAI.

Yong Dai, Jian Liu, Xiancong Ren, and Zenglin Xu. 2020. Adversarial training based multi-source unsupervised domain adaptation for sentiment analysis. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pages 7618–7625.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, Minneapolis, Minnesota. Association for Computational Linguistics.

Rotem Dror, Gili Baumer, Segev Shlomov, and Roi Reichart. 2018. The hitchhiker's guide to testing statistical significance in natural language processing. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1383–1392, Melbourne, Australia. Association for Computational Linguistics.

Ashwin Geet D'Sa, Irina Illina, and Dominique Fohr. 2020. Towards non-toxic landscapes: Automatic toxic comment detection using DNN. In *Proceedings of the Second Workshop on Trolling, Aggression*

and Cyberbullying, pages 21–25, Marseille, France. European Language Resources Association (ELRA).

Bradley Efron and Robert J. Tibshirani. 1993. *An Introduction to the Bootstrap*. Number 57 in Monographs on Statistics and Applied Probability. Chapman & Hall/CRC, Boca Raton, Florida, USA.

Komal Florio, Valerio Basile, Marco Polignano, Pierpaolo Basile, and Viviana Patti. 2020. Time of your hate: The challenge of time in hate speech detection on social media. *Applied Sciences*, 10(12).

Goran Glavaš, Mladen Karan, and Ivan Vulić. 2020. XHate-999: Analyzing and detecting abusive language across domains and languages. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 6350–6365, Barcelona, Spain (Online). International Committee on Computational Linguistics.

John G. D. Grieve, Andrea Nini, and Diansheng Guo. 2018. Mapping lexical innovation on american social media. *Journal of English Linguistics*, 46:293 – 319.

Xisen Jin, Zhongyu Wei, Junyi Du, Xiangyang Xue, and Xiang Ren. 2020. Towards hierarchical importance attribution: Explaining compositional semantics for neural sequence models. In *International Conference on Learning Representations*.

Mladen Karan and Jan Šnajder. 2018. Cross-domain detection of abusive language online. In *Proceedings of the 2nd Workshop on Abusive Language Online (ALW2)*, pages 132–137, Brussels, Belgium. Association for Computational Linguistics.

Brendan Kennedy, Xisen Jin, Aida Mostafazadeh Davani, Morteza Dehghani, and Xiang Ren. 2020. Contextualizing hate speech classifiers with post-hoc explanation. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 5435–5442, Online. Association for Computational Linguistics.

Adam Kilgarriff. 2001. Comparing corpora. *International Journal of Corpus Linguistics*, 6(1):97–133.

Zheng Li, Ying Wei, Yu Zhang, and Qiang Yang. 2018. Hierarchical attention transfer network for cross-domain sentiment classification. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 32.

Zheng Li, Yu Zhang, Ying Wei, Yuxiang Wu, and Qiang Yang. 2017. End-to-end adversarial memory network for cross-domain sentiment classification. In *Proceedings of the 26th International Joint Conference on Artificial Intelligence*, IJCAI'17, pages 2237–2243. AAAI Press.

Frederick Liu and Besim Avci. 2019. Incorporating priors with feature attribution on text classification. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 6274–6283, Florence, Italy. Association for Computational Linguistics.

Ilya Loshchilov and Frank Hutter. 2019. Decoupled weight decay regularization. In *International Conference on Learning Representations*.

Scott M. Lundberg and Su-In Lee. 2017. A unified approach to interpreting model predictions. In *Proceedings of the 31st International Conference on Neural Information Processing Systems*, NIPS'17, pages 4768–4777, Red Hook, NY, USA. Curran Associates Inc.

Adyasha Maharana and Mohit Bansal. 2020. Adversarial augmentation policy search for domain and cross-lingual generalization in reading comprehension. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 3723–3738, Online. Association for Computational Linguistics.

Shervin Malmasi and Marcos Zampieri. 2018. Challenges in discriminating profanity from hate speech. *Journal of Experimental & Theoretical Artificial Intelligence*, 30(2):187–202.

Marzieh Mozafari, Reza Farahbakhsh, and Noel Crespi. 2019. A bert-based transfer learning approach for hate speech detection in online social media. In *International Conference on Complex Networks and Their Applications*, pages 928–940. Springer.

Yida Mu and Nikolaos Aletras. 2020. Identifying twitter users who repost unreliable news sources with linguistic information. *PeerJ Computer Science*, 6:e325.

Yida Mu, Pu Niu, and Nikolaos Aletras. 2022. Identifying and characterizing active citizens who refute misinformation in social media. In *14th ACM Web Science Conference 2022*, page 401–410, New York, NY, USA. Association for Computing Machinery.

Fabio Poletto, Valerio Basile, Cristina Bosco, Viviana Patti, and Marco Antonio Stranisci. 2019. Annotating hate speech: Three schemes at comparison. In *CLiC-it*.

Laura Rieger, Chandan Singh, William Murdoch, and Bin Yu. 2020. Interpretations are useful: Penalizing explanations to align neural networks with prior knowledge. In *Proceedings of the 37th International Conference on Machine Learning*, volume 119 of *Proceedings of Machine Learning Research*, pages 8116–8126. PMLR.

Minho Ryu and K. Lee. 2020. Knowledge distillation for bert unsupervised domain adaptation. *ArXiv*, abs/2010.11478.

Ankan Saha and Vikas Sindhwani. 2012. Learning evolving and emerging topics in social media: A dynamic nmf approach with temporal regularization. In *Proceedings of the Fifth ACM International Conference on Web Search and Data Mining*, WSDM '12, page 693–702, New York, NY, USA. Association for Computing Machinery.

Sheikh Muhammad Sarwar and Vanessa Murdock. 2022. Unsupervised domain adaptation for hate speech detection using a data augmentation approach. In *Proceedings of the 16th International Conference on Web and Social Media*.

Sofia Serrano and Noah A. Smith. 2019. Is attention interpretable? In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 2931–2951, Florence, Italy. Association for Computational Linguistics.

Avanti Shrikumar, Peyton Greenside, and Anshul Kundaje. 2017. Learning important features through propagating activation differences. In *Proceedings of the 34th International Conference on Machine Learning*, volume 70 of *Proceedings of Machine Learning Research*, pages 3145–3153. PMLR.

Mukund Sundararajan, Ankur Taly, and Qiqi Yan. 2017. Axiomatic attribution for deep networks. In *Proceedings of the 34th International Conference on Machine Learning*, volume 70 of *Proceedings of Machine Learning Research*, pages 3319–3328. PMLR.

Steve Durairaj Swamy, Anupam Jamatia, and Björn Gambäck. 2019. Studying generalisability across abusive language detection datasets. In *Proceedings of the 23rd Conference on Computational Natural Language Learning (CoNLL)*, pages 940–950, Hong Kong, China. Association for Computational Linguistics.

Eric Tzeng, Judy Hoffman, Kate Saenko, and Trevor Darrell. 2017. Adversarial discriminative domain adaptation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 7167–7176.

Bertie Vidgen, Tristan Thrush, Zeerak Waseem, and Douwe Kiela. 2021. Learning from the worst: Dynamically generated datasets to improve online hate detection. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 1667–1682. Association for Computational Linguistics.

Zeerak Waseem and Dirk Hovy. 2016. Hateful symbols or hateful people? predictive features for hate speech detection on Twitter. In *Proceedings of the NAACL Student Research Workshop*, pages 88–93, San Diego, California. Association for Computational Linguistics.

Michael Wiegand, Josef Ruppenhofer, and Thomas Kleinbauer. 2019. Detection of Abusive Language: the Problem of Biased Datasets. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 602–608, Minneapolis, Minnesota. Association for Computational Linguistics.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Remi Louf, Morgan Funtowicz, Joe Davison, Sam Shleifer, Patrick von Platen, Clara Ma, Yacine Jernite, Julien Plu, Canwen Xu, Teven Le Scao, Sylvain Gugger, Mariama Drame, Quentin Lhoest, and Alexander Rush. 2020. Transformers: State-of-the-art natural language processing. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 38–45, Online. Association for Computational Linguistics.

Huihan Yao, Ying Chen, Qinyuan Ye, Xisen Jin, and Xiang Ren. 2021. Refining language models with compositional explanations. *Advances in Neural Information Processing Systems*, 34.

Yalan Ye, Ziwei Huang, Tongjie Pan, Jingjing Li, and Heng Tao Shen. 2021. Reducing bias to source samples for unsupervised domain adaptation. *Neural Networks*, 141:61–71.

Wenjie Yin and Arkaitz Zubiaga. 2021. Towards generalisable hate speech detection: a review on obstacles and solutions. *PeerJ Computer Science*, 7.

Andrea Zunino, Sarah Adel Bargal, Riccardo Volpi, Mehrnoosh Sameki, Jianming Zhang, Stan Sclaroff, Vittorio Murino, and Kate Saenko. 2021. Explainable deep classification models for domain generalization. In *2021 IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, pages 3227–3236.

A Differences across Datasets

The datasets *HatEval* (Basile et al., 2019) and *Waseem* (Waseem and Hovy, 2016) have been sampled from Twitter. *HatEval* has primarily been collected in the year 2018 using a combination of sampling strategies, including keyword-based sampling (with both neutral and derogatory words), collecting the history of identified perpetrators and monitoring the potential victims of hate. It mainly consists of hate against women and immigrants. In the case of *Waseem*, tweets are collected particularly using keyword-based sampling in or before 2016, with keywords that are likely to co-occur with hateful content. Wiegand et al. (2019) discuss the presence of a large amount of topic-bias in the dataset *Waseem*. Since this dataset is available as tweet-IDs, we observe that in the crawled dataset, many tweets flagged as racist are missing, and have most likely been deleted already. Thus, the majority of available hateful content in this dataset is directed against women. The topics discussed in these two datasets are also quite different.

Vidgen[5] (Vidgen et al., 2021), on the other hand, is a dataset generated using a human and model-in-loop process. This process results in adding several perturbations and instances, which are difficult to classify, aimed at making the dataset robust. Besides, it consists of hateful content directed against a wide array of target groups, e.g. *black*, *gay*, *muslim*, *disabled*, etc., along with different forms of hate such as *derogation*, *threatening language*, *animosity*, *support for hateful entities* and *dehumanization*. Thus there is a substantial amount of differences across these datasets in terms of collection time-frames, sampling strategies, targets of hate, forms of hate, vocabulary used and the like.

For pre-processing the datasets, we remove the URLs, split the hashtags using CrazyTokenizer[6] and lowercase the terms.

B Implementation Details and Hyper-parameter Tuning

We use the pre-trained BERT-base (Devlin et al., 2019) uncased model[7] (Wolf et al., 2020) for our experiments. We run both the Masked Language Model (MLM) training on the unlabeled target domain training data D_T^{train}, and the subsequent supervised fine-tuning on the source domain training data D_S^{train} for 6 epochs with a batch size of 8 for all the BERT baselines and Dom-spec. We use the AdamW optimizer with decoupled weight decay regularization (Loshchilov and Hutter, 2019), having a weight decay of 10^{-4}. We use a learning rate of 3×10^{-5} for the MLM training and 1×10^{-5} for the supervised fine-tuning, with the epsilon parameter set to 1×10^{-8}.

We use the original implementations provided by the respective authors of all the baselines except for Sarwar and Murdock (2022). We implement the data-augmentation approach by Sarwar and Murdock (2022) ourselves, as there is no available implementation. We follow the description provided in the paper and label all the terms in the hateful instances of the source domain that have a match with hatebase.org[8] for training a sequence tagger. However, while finding the matches, we do not tokenize the multi-word phrases in hatebase.org. We lowercase the terms from hatebase.org and look for an exact match of a term in the source domain.

For Pre-Def, we combine the curated list of identity terms provided by Liu and Avci (2019) and Kennedy et al. (2020) and penalize their attribution scores. We perform hyper-parameter tuning and model selection with early-stopping on a small amount of labeled target domain validation set D_T^{val} using the macro-F1 score for the proposed approach as well as for all the baselines. The hyper-parameter λ, both for the proposed approach and Pre-Def, is selected from the range $\lambda \in \{0.01, 0.05, 0.1, 1.0, 10.0, 20.0, 30.0, 40.0, 50.0, 60.0\}$, using a random seed by tuning over D_T^{val}. We set the value of M to 250 and N to 750 for all our experiments.

C Terms Extracted

The full-list of penalized terms (BERT WordPieces) te^S across epochs for the examples listed in Section 4.2, is given below.

Waseem →HatEval

- **Epoch 1:** {college, sports, feminism, la, magnetic, used, unique, ##ava, speech, ##js, tr, ##cking, object, chu, result, ki, bus, ##is, adopt, referring, ##roids, handed, ##em, sh, ##omp, unconscious, anger, gamer, prove, xbox, tri, skill, judgment, tool, block, single, harassment, size, georgia, involved, ##ism, studying, voices, possible, gaming, pl, ##il, helped, ##ke, survey, equality}
- **Epoch 2:** {feminism, used, football, awesome, equal, ##cking, object, ##ification, interest, feminist, ##rra, scientist, ##al, ignorance, bodies, ##work, later, ##nk, troll, ##ss, based, adopt, ##cing, quality, sister, unconscious, criticisms, pro, notch, xbox, tri, unfair, rap, meanwhile, impression, single, harassment, bonus, georgia, constant, sex, ##ist, possible, click, competition, ##per, swedish, ##eral, november, write, eventually, equality}
- **Epoch 3:** {sham, anger, pull, used, focus, speech, ashley, object, interest, bringing, ##na, eye, ##nk, later, quality, ##roids, oppressive, rain, ##omp, statistics, nsw, content, notch, museum, unconscious, typically, tri, ##ol, unfair, writing, ##chan, georgia, constant, annie, ra, weights, click, ##il, furniture, helped, shopping, football, commentary, equality}
- **Epoch 4:** {minded, kat, used, equal, focus, ##hand, tr, ##cking, chu, interest, bringing,

[5] We use an older version of the dataset. The authors have uploaded a newer version of this dataset currently.

[6] https://redditscore.readthedocs.io

[7] https://huggingface.co/bert-base-uncased

[8] https://hatebase.org/

Approaches	HatEval		Vidgen		Waseem		Mean
BERT Van-FT	43.3±1.8		85.1±0.5		85.4±0.7		71.3
Performance on source domain (left of arrows) while applying domain adaptation for the target (right of arrows)							
	H →V	**H →W**	**V →H**	**V →W**	**W →H**	**W →V**	
Dom-spec ($\alpha\nabla\alpha$)	42.4±2.5	42.0±4.1	84.0±0.9	84.5±1.0	85.1±0.7	83.8±0.8	70.3
Dom-spec (DL)	41.7±3.7	40.5±4.4	83.9±0.7	82.6±1.5	84.7±1.2	81.1±2.7	69.1

Table 4: Effect of domain adaptation for the target on the source domain performance; Source-domain macro average F1 scores (mean±std-dev) are obtained after MLM training on the unlabeled target domain and penalizing the source specific terms while adapting the model to the target domain (present at the right hand side of the arrows) using Dom-spec. H : HatEval, V : Vidgen, W : Waseem. Van-FT: BERT model evaluated in-domain *without* MLM training on the target domain.

thor, fm, ##tag, path, scientist, precious, later, mike, quality, humanist, ##roids, ##el, ##omp, worth, unconscious, nsw, xbox, tri, unfair, nu, kaitlyn, ##ering, pest, fe, camera, giant, constant, weights, gaming, rap, ##il, swedish, opposes, ##thi, november, laughing, survey, equality}

- **Epoch 5:** {feminism, raging, equal, focus, ##hand, ##cking, ##cky, ##tag, ##na, mostly, scientist, ##al, ##rra, adopt, humanist, ft, ##roids, ##el, ##omp, example, unconscious, museum, anger, typically, tri, unfair, impression, yu, single, fe, cu, ##rd, ##ification, constant, grass, gaming, rap, science, ##per, swedish, il, furniture, shopping, november, equality}

Few of the extracted terms get repeated in subsequent epochs as a single epoch may not be sufficient to reduce the effect of a term and it may appear in the next epoch as well. Moreover, as the training progresses, the model may learn new patterns, and some extracted terms may reappear and disappear again due to the penalization.

Following is a *non-hateful* example in *HatEval*, wrongly classified by Van-MLM-FT but correctly classified by Dom-spec (The darker the shades, the higher the attribution scores assigned):

Van-MLM-FT: Unfortunately you are in a sticky size my only problem is replacing my shoes has been a b*tch

Dom-spec (DL): Unfortunately you are in a sticky size my only problem is replacing my shoes has been a b*tch

Vidgen →Waseem

- **Epoch 1:** {wheelchair, ##zzi, dali, seekers, ##oons, koreans, ##tos, ##ware, ##ders, handicapped, principles, mac, pregnant, ##tier, ##iers, ##wear, ##bib, barren, ##tite, dyke}
- **Epoch 2:** {customer, pip, principles, ##tos, ##hon, les, ko, vietnamese, teenagers, ##lock, ##sion, ##has, ##gin, ##rmi, poles, buddhist, handicapped}
- **Epoch 3:** {pak, homosexuality, koreans, pleasant, ##tos, mirror, spaniards, ##fs, ro, ##rmi, boom, handicapped}
- **Epoch 4:** {##cky, pak, chin, ##tos, bender, herr, catholics, ro, buddhist}
- **Epoch 5:** {pip, pak, ##tos, yellow, bender, koreans, ##mit, ##sion, ##has, ##rk, ##gin, catholics, ro, arrogance}

Following is a *non-hateful* example in *Waseem*, wrongly classified by Van-MLM-FT, but correctly classified by the proposed approach (The darker the shades, the higher the attribution scores assigned):

Van-MLM-FT: Omg I am lisening to an apple genius dude tell this old woman how to use email and it is adorable

Dom-spec (DL): Omg I am listening to an apple genius dude tell this old woman how to use email and it is adorable

D In-domain Performance

Table 4 presents, as a reference, the in-domain macro-F1 scores using BERT supervised fine-tuning (Van-FT) *without* the MLM training on the target domain. In this case, the model is tuned over

Approaches	HatEval	Vidgen	Waseem
BERT Van-MLM-FT	1 m 20 s	3 m 49 s	2 m 10s
Dom-spec ($\alpha\nabla\alpha$)	2 m 30s	7 m	3 m 17 s
Dom-spec (DL)	4 m	18 m	8 m 16 s

Table 5: Per epoch training time on different source domains.

the in-domain validation set. The *HatEval* dataset is part of a shared task and involves a challenging test set with low in-domain performance. Table 4 displays the source-domain scores obtained when source-specific terms are penalized, while adapting to the target domain using Dom-spec, where the model is tuned over the target domain validation set. The drop in in-domain performance is expected as Dom-spec is aimed at making the model best-suited to the target domain. However, the overall performance with Dom-spec is comparable to that of BERT Van-FT.

E List of Identity Terms for Pre-Def

The combined list of pre-defined curated identity terms from Liu and Avci (2019) and Kennedy et al. (2020) are given below:

{lesbian, gay, bisexual, trans, cis, queer, lgbt, lgbtq, straight, heterosexual, male, female, non-binary, african, african american, european, hispanic, latino, latina, latinx, canadian, american, asian, indian middle eastern, chinese, japanese, christian, buddhist, catholic, protestant, sikh, taoist, old, older, young, younger, teenage, millenial, middle aged, elderly, blind, deaf, paralyzed, muslim, jew, jews, white, islam, blacks, muslims, women, whites, gay, black, democrat, islamic, allah, jewish, lesbian, transgender, race, brown, woman, mexican, religion, homosexual, homosexuality, africans}

F Computational Efficiency

The per-epoch training time for Dom-spec, while performing adaptation of different source domain models, are presented in Table 5. Dom-spec ($\alpha\nabla\alpha$) takes less than double the time taken by Van-MLM-FT to train, and Dom-spec (DL) takes roughly 4.5 times of the training time taken by Van-MLM-FT.

Generalizable Implicit Hate Speech Detection using Contrastive Learning

Youngwook Kim[1], **Shinwoo Park**[2] and **Yo-Sub Han**[1]
[1]Department of Computer Science, Yonsei University, Seoul, Republic of Korea
[2]Department of Artificial Intelligence, Yonsei University, Seoul, Republic of Korea
{youngwook, pshkhh, emmous}@yonsei.ac.kr

Abstract

Hate speech detection has gained increasing attention with the growing prevalence of hateful contents. When a text contains an obvious hate word or expression, it is fairly easy to detect it. However, it is challenging to identify implicit hate speech in nuance or context when there are insufficient lexical cues. Recently, there are several attempts to detect implicit hate speech leveraging pre-trained language models such as BERT and HateBERT. Fine-tuning on an implicit hate speech dataset shows satisfactory performance when evaluated on the test set of the dataset used for training. However, we empirically confirm that the performance drops at least 12.5%p in F1 score when tested on the dataset that is different from the one used for training. We tackle this cross-dataset underperforming problem using contrastive learning. Based on our observation of common underlying implications in various forms of hate posts, we propose a novel contrastive learning method, *ImpCon*, that pulls an implication and its corresponding posts close in representation space. We evaluate the effectiveness of *ImpCon* by running cross-dataset evaluation on three implicit hate speech benchmarks. The experimental results on cross-dataset show that *ImpCon* improves at most 9.10% on BERT, and 8.71% on HateBERT.

1 Introduction

Warning: *this paper contains contents that may be offensive or upsetting.*

Hate speech is "any communication that disparages a target group of people based on some characteristic such as race, color, ethnicity, gender, sexual orientation, nationality, religion, or other characteristic" (Nockleby, 2000). Recently, there are several attempts to detect hate speech or abusive text using lexicon-based methods (Chen et al., 2012; Gitari et al., 2015; Lee et al., 2018; Wiegand et al., 2018) or neural-based methods (Gambäck and Sikdar, 2017; Badjatiya et al., 2017; Park and Fung, 2017; Zhang et al., 2018; Lee et al., 2019; Wang et al., 2020). While these approaches work fairly well when a text contains an explicit hate or abusive word, they often fail to detect implicit ones. See Table 1 for examples of implicit hate speech. Caselli et al. (2020) showed that the pre-trained language model struggles to detect implicit abusiveness. They suspect that a small amount of training data with implicit abusiveness is a main reason for the poor performance. In this vein, ElSherief et al.

Text (Input) : "my world orbits around whites as it should . laughable moment though ." **Label** : Hate
Text (Input) : "that is part of the white supremacy logic that native people are less than human . we aren't ." **Label** : Not Hate
Text (Input) : "send them back to the countries they came from" **Label** : Hate

Table 1: Example input texts and labels (Hate / Not Hate) from IMPLICIT HATE CORPUS (IHC) (ElSherief et al., 2021) which is an implicit hate speech dataset.

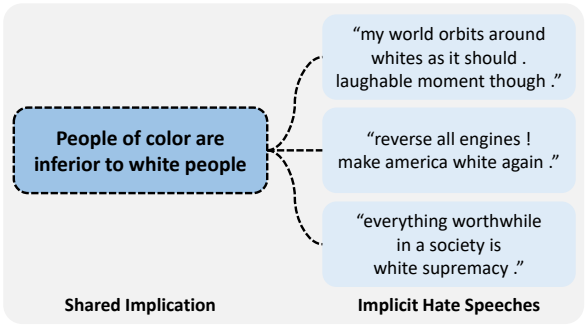

Figure 1: Implicit hate speeches and their shared implication from IMPLICIT HATE CORPUS (IHC).

Proceedings of the 29th International Conference on Computational Linguistics, pages 6667–6679
October 12–17, 2022.

(2021) recently presented an implicit hate speech benchmark. The models trained on this dataset outperform other baselines in terms of in-dataset evaluation performance. In general, the hate speech detection performance can be over-estimated when evaluated on its own test set (Arango et al., 2019; Yin and Zubiaga, 2021). On in-dataset evaluation, a model is evaluated on the test set of the same dataset used for training. However, on cross-dataset evaluation, a model is evaluated on the dataset that is different from the one used for training. Instead of in-dataset evaluation, it is better to run a cross-dataset evaluation to see the generalization ability of a model (Wiegand et al., 2019). As a preliminary experiment, we perform the cross-dataset evaluation for the current state-of-the-art models trained on implicit hate speech datasets. In Section 2.2, we empirically observe relatively low performance on cross-dataset evaluation.

Prior research (Gunel et al., 2021) incorporates contrastive learning into their fine-tuning process, resulting in better generalization ability in few-shot learning setup. Motivated by this, we propose contrastive learning methods to improve the generalization ability of implicit hate detectors on cross-dataset. Contrastive learning makes positive pairs to be close together and negative pairs to be apart in the representation space (Rethmeier and Augenstein, 2022). One of the key issues in contrastive learning is how to choose positive samples. Depending on different choices of positive sampling, a model would learn different invariant features (Tian et al., 2020). Here, we suggest two positive sampling strategies: 1) Leveraging augmented posts as positive samples of given posts (*AugCon*); 2) Leveraging implications as positive samples of given hateful posts (*ImpCon*). For *AugCon*, we first generate augmented posts which are lexically different but semantically similar with their original posts. *ImpCon* leverages implications as positive samples of hateful posts, since it contains concealed meaning of the hateful posts. In addition, a common implication is often shared by a group of hateful posts, as shown in Figure 1. By pulling an implication and its corresponding hateful posts close in representation space, the model can learn common features among a group of hateful posts sharing an implication.

We evaluate the generalization ability of models trained using *AugCon* and *ImpCon*. We conduct cross-dataset evaluation on three implicit hate speech benchmarks with BERT and HateBERT as base models. By incorporating *AugCon* or *ImpCon* in fine-tuning, we can improve the cross-dataset evaluation performance. While improvement with *AugCon* is limited to BERT (at most 2.92% improvement), *ImpCon* brings consistent improvements across all cross-datasets and models (at most 9.10% improvement to BERT and 8.71% improvement to HateBERT). The consistent improvement of *ImpCon* demonstrates the effectiveness of leveraging implication-post pair on the generalization ability. Moreover, further analysis on *ImpCon* shows that even unseen implication-post pairs are projected closer on the representation space (Section 5.1), resulting in consistent predictions on cross-dataset (Section 5.2). Our code is available at https://github.com/youngwook06/ImpCon.

BERT			
	Test		
Train	IHC	SBIC	DYNAHATE
IHC	0.777	**0.568**	**0.531**
SBIC	**0.596**	0.838	**0.603**
DYNAHATE	**0.660**	**0.663**	0.788
HateBERT			
	Test		
Train	IHC	SBIC	DYNAHATE
IHC	0.764	**0.587**	**0.547**
SBIC	**0.587**	0.840	**0.598**
DYNAHATE	**0.662**	**0.668**	0.794

Table 2: Cross-dataset and in-dataset evaluation results of BERT and HateBERT. The column on the left indicates the dataset used for training, while the row on the top indicates the dataset used for evaluation. Cross-dataset evaluation results are presented in bold.

2 Related Work and Preliminary Experiment

2.1 Hate Speech Detection

With the increase of online media and user contents, hate speech becomes more pervasive online. Considering the massive volume of online posts, it is impractical to manually moderate all posts. Researchers have developed many hate speech detection models, including lexicon-based approaches (Chen et al., 2012; Gitari et al., 2015; Lee et al., 2018; Wiegand et al., 2018) and neural network models (Gambäck and Sikdar, 2017; Badjatiya et al., 2017; Park and Fung,

2017; Zhang et al., 2018; Lee et al., 2019; Wang et al., 2020). Also, there are several datasets available for hate speech detection with different focuses (Warner and Hirschberg, 2012; Davidson et al., 2017; Founta et al., 2018; Basile et al., 2019). For example, Davidson et al. (2017) introduced a dataset to distinguish hate speech from an offensive language and Founta et al. (2018) investigated representative labels by merging and eliminating some labels related to abusive tweets. However, many of these datasets are skewed towards explicit forms of abusiveness since the data collection strategies often rely on explicit signals such as hateful lexicons (ElSherief et al., 2021). A model trained on such dataset often fails to detect implicit hate, even for the pre-trained language model (Caselli et al., 2020).

Recently, researchers show their interests in addressing implicit hate or abusiveness. Han and Tsvetkov (2020) used a set of probing data for the robust classifier which better detects disguised toxicity. Wiegand et al. (2021) studied subtypes of implicit abuse and existing datasets. ElSherief et al. (2021) presented a benchmark with implicit hate label, annotated target and implication.

2.2 Preliminary Experiment

Several works in hate speech detection have reported a large drop of the fine-tuned model performance when evaluated on cross-dataset (Gröndahl et al., 2018; Arango et al., 2019; Swamy et al., 2019). We conduct a preliminary experiment to see if implicit hate speech detection models can still perform well on cross-datasets that are also skewed towards implicit hate. We use three implicit hate datasets (IMPLICIT HATE CORPUS (IHC), SOCIAL BIAS INFERENCE CORPUS (SBIC) and DYNAHATE) following Hartvigsen et al. (2022). Detailed descriptions of the datasets are presented in Section 4.1. We experiment with one of the state-of-the-art models, BERT (Devlin et al., 2019). We also experiment with HateBERT (Caselli et al., 2021), which is pre-trained on abusive corpus and showed better generalization ability than BERT in their paper. In the cross-dataset evaluation with implicit hate datasets, we observe the similar generalization issue. As shown in Table 2, the performance of both models drops consistently over 12.5%p in F1 score across implicit hate speech datasets. Through the preliminary experiment, we conclude that implicitly trained models suffer from generalization issue and combating the issue is needed.

2.3 Contrastive Learning

Recently, contrastive learning has been widely used to learn representation in various domains and showed its effectiveness. Many works on contrastive learning have proposed diverse choices of positive sampling.

For example, in the computer vision field, SimCLR (Chen et al., 2020) applies random augmentation on images and those augmented images from a same image are considered positive. Khosla et al. (2020) proposed to use the samples from the same class for positive sampling. In the natural language processing field, CERT (Fang et al., 2020) augments text with back-translation and considers augmented texts from the same text as positive. Also, Giorgi et al. (2021) suggested leveraging textual segments nearby in the document as positive samples. Gao et al. (2021) proposed using pairs from natural language inference datasets for positive sampling.

Some works on text classification proposed to apply contrastive learning to fine-tune the model. Gunel et al. (2021) showed that pulling instances from the same class closer while fine-tuning improved few-shot learning performance. Suresh and Ong (2021) extended this approach and showed that weighting negative samples differently increased performance on fine-grained classification. Pan et al. (2022) used adversarial examples as positives and showed outperforming performance over standard fine-tuning. We suggest using contrastive learning in the fine-tuning process for generalizable implicit hate speech detection.

3 Approach

3.1 Overall Training Objective

Generally, hate speech detection models are fine-tuned in a supervised way using the following cross-entropy loss $\mathcal{L}_{ce}$:

$$\mathcal{L}_{ce} = -\frac{1}{N}\sum_{i=1}^{N}[y_i \log \hat{y}_i + (1-y_i)\log(1-\hat{y}_i)], \quad (1)$$

where N is the number of input posts in a batch, $\hat{y}_i$ indicates the model predicted probability of i-th input x_i and y_i is the ground-truth label of x_i, respectively. However, since cross-entropy loss has limitation on making large inter-class margin or intra-class compactness, fine-tuning using only

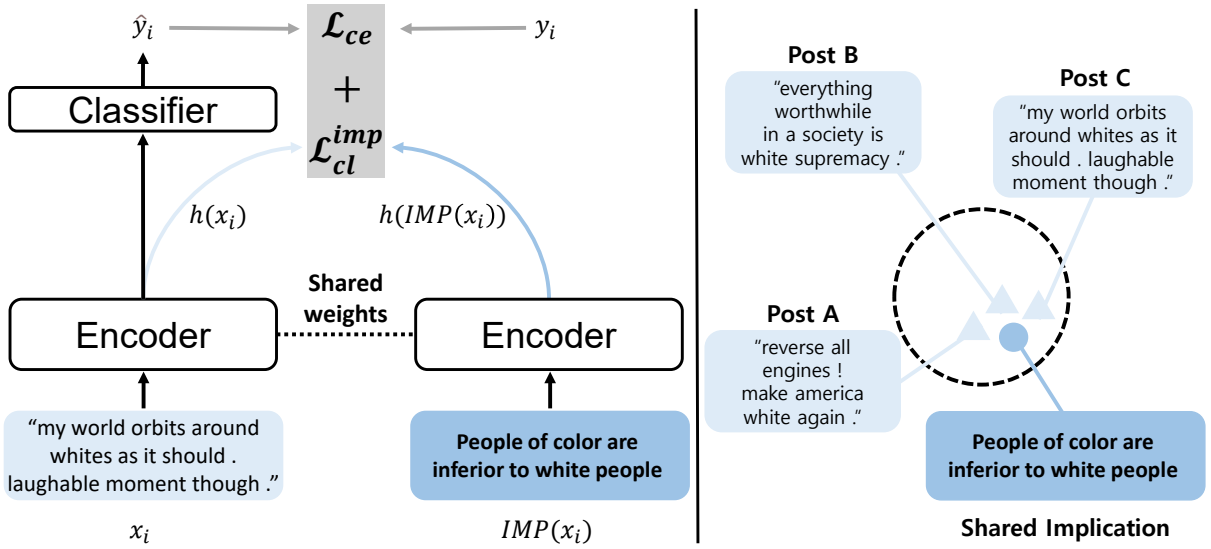

Figure 2: **Left**: The overview of training a model with $\mathcal{L}_{overall}^{imp}$ (Equation (7)). We present a hateful post and its implication as an example positive pair for $\mathcal{L}_{cl}^{imp}$ (Equation (6)). **Right**: *ImpCon* aims at pulling an implication and its corresponding hateful posts, resulting in similar representation between a shared implication and its hateful posts in the representation space. All implication and hateful posts in the figure are from IHC dataset.

cross-entropy loss can result in suboptimal generalization (Liu et al., 2016; Zhao et al., 2021).

We propose to combine contrastive loss with cross-entropy loss to train generalizable implicit hate speech detector. Contrastive loss pushes the representation of positive pairs closer and negative pairs further apart. We denote the positive sample of x_i as x_i^{pos} $(i \geq 1)$. Given N training input posts in a batch, we assume one positive sample per post, leading to total $2N$ samples in a batch. When x_i^{pos} is the j-th input in a batch, *i.e.*, $x_i^{pos} = x_j$, we assume $j = i + N$ if $i \leq N$ and $j = i - N$ if $i > N$. We consider all samples other than a positive sample as negative samples, excluding itself. Following Chen et al. (2020), the contrastive learning loss $\mathcal{L}_{cl}$ can be defined as:

$$\mathcal{L}_{cl} = -\sum_{i=1}^{2N} \log \frac{e^{h(x_i) \cdot h(x_i^{pos})/\tau}}{\sum_{k=1}^{2N} \mathbb{1}_{[k \neq i]} e^{h(x_i) \cdot h(x_k)/\tau}}, \quad (2)$$

where $\cdot$ denotes dot product operation, $h(x_i) \in \mathbb{R}^H$ is the representation of the encoder for input x_i, and H is the hidden dimension size. In detail, the last layer representation of $[CLS]$ token is further normalized and used as $h(x_i)$ for input x_i. $\mathbb{1}_{[\cdot]}$ is an indicator function and τ is a scalar temperature parameter.

Our training objective for fine-tuning is the combination of cross-entropy loss $\mathcal{L}_{ce}$ and contrastive learning loss $\mathcal{L}_{cl}$:

$$\mathcal{L}_{overall} = \lambda \mathcal{L}_{ce} + (1 - \lambda)\mathcal{L}_{cl}, \quad (3)$$

where λ is a loss scaling hyperparameter.

3.2 Positive Sampling

The strategies of constructing positive samples for contrastive learning have been studied actively. In the following, we give detailed description of two positive sampling strategies for generalizable implicit hate speech detection.

3.2.1 Augmented Post as Positive Samples

It has been shown that unintended biases in a dataset could lead to the generalization issue of a model detecting abusiveness (Wiegand et al., 2019). Due to the lack of lexical cues in implicit hate speech and its subtlety, we suspect that implicit hate speech detector could easily overfit to unintended lexical biases in the dataset. To ease such issue, we suggest using augmented post as a positive sample. Our intuition is that by using augmented variants of posts, which are lexically different but semantically similar with original posts, the model can learn more invariant semantic features.

When we denote augmentation module as $AUG(\cdot)$, here, we set the positive sample for i-th input x_i as $x_i^{pos} = x_j = AUG(x_i)$. For $i \leq N$, $AUG(x_i)$ is the augmented version of x_i. For $i > N$, $AUG(x_i)$ is the original input post (be-

fore augmentation) of x_i. Specifically, for augmentation, we leverage synonym substitution following Suresh and Ong (2021). However, we note that any augmentation can be used for $AUG(\cdot)$. The contrastive learning loss $\mathcal{L}_{cl}^{aug}$ using augmented post as a positive sample is defined as:

$$\mathcal{L}_{cl}^{aug} = -\sum_{i=1}^{2N} \log \frac{e^{h(x_i)\cdot h(AUG(x_i))/\tau}}{\sum_{k=1}^{2N} \mathbb{1}_{[k\neq i]} e^{h(x_i)\cdot h(x_k)/\tau}}. \quad (4)$$

We refer to this contrastive learning with augmented posts as *AugCon*. Then, overall objective for fine-tuning with cross-entropy loss and *AugCon* is:

$$\mathcal{L}_{overall}^{aug} = \lambda \mathcal{L}_{ce} + (1-\lambda)\mathcal{L}_{cl}^{aug}. \quad (5)$$

3.2.2 Implication as Positive Samples

Hate speech conveys a targeted group and disparaging stereotypes and biases regarding the group. At times, although presented differently, a group of hateful posts implies similar harmful biases. That is, people generating hate speech often project one implication to various lexical forms of posts. Inspired by the relationship between an implication and its various lexical forms of hateful posts, we propose to use an implication of a hateful post as a positive sample. By pulling a hateful post and its implication in the training process, an implication can work as an anchor for its corresponding hateful posts. This would enable a model to learn the relationship between a hateful post and its concealed meaning, leading to more generalizable implicit hate speech detector.

We assume a module $IMP(\cdot)$, where we set the positive sample for i-th input x_i as $x_i^{pos} = x_j = IMP(x_i)$. For $i \leq N$, $IMP(x_i)$ means an implication of x_i if x_i is a hateful post, otherwise (*i.e.*, if x_i is a non-hateful post) $IMP(x_i)$ means an augmented version of x_i. For $i > N$, $IMP(x_i)$ means the original input post of x_i (*i.e.*, x_i is an implication or augmented version of $IMP(x_i)$). In detail, for implication, we use implications that are given in IHC and SBIC dataset [1]. For augmentation, we use the same augmentation as *AugCon*. The contrastive learning loss $\mathcal{L}_{cl}^{imp}$ using implication as a positive sample is defined as:

$$\mathcal{L}_{cl}^{imp} = -\sum_{i=1}^{2N} \log \frac{e^{h(x_i)\cdot h(IMP(x_i))/\tau}}{\sum_{k=1}^{2N} \mathbb{1}_{[k\neq i]} e^{h(x_i)\cdot h(x_k)/\tau}}. \quad (6)$$

We refer to this contrastive learning using implication as *ImpCon*. Then, overall objective for fine-tuning with cross-entropy loss and *ImpCon* is:

$$\mathcal{L}_{overall}^{imp} = \lambda \mathcal{L}_{ce} + (1-\lambda)\mathcal{L}_{cl}^{imp}. \quad (7)$$

The overview of training a model with $\mathcal{L}_{overall}^{imp}$ is demonstrated in Figure 2.

4 Experiment

4.1 Datasets

We perform binary classification of detecting hateful language on implicit hate datasets. For cross-dataset evaluation, we use three implicit hate speech datasets as Hartvigsen et al. (2022). SOCIAL BIAS INFERENCE CORPUS (SBIC) (Sap et al., 2020) is the dataset with hierarchical annotation of social bias including offensiveness, target, and implied statement. Similarly, IMPLICIT HATE CORPUS (IHC) (ElSherief et al., 2021) is the implicit hate speech dataset with target and implication collected from hate communities and their followers on Twitter. DYNAHATE (Vidgen et al., 2021) is the hate speech dataset collected through human-and-model-in-the-loop process of deceiving a model.

Since one of our main focus is leveraging implication for generalizable model, we fine-tune models on two datasets with implications, IHC and SBIC. For IHC, we refined the dataset considering the uniformity across annotation stages, resulting in all 'implicit hate' labeled samples having implications. For SBIC, we aggregate annotations of each post. In addition, we merge an implied statement with a target to get an implication following the set of rules in Marasović et al. (2022).

4.2 Baseline Training Approaches

We experimented with three baseline training approaches.

- **Cross-entropy Loss (CE)**: we fine-tune a model with cross-entropy loss (CE), which is a general approach in hate speech classification.
- **Cross-entropy Loss (CE) with Data Augmentation**: we apply data augmentation to

[1] If there exists any hateful post without a given implication in the dataset, then we use an augmented post instead of an implication.

Model	Objective	IHC → SBIC (Cross-dataset)	IHC → DynaHate (Cross-dataset)	IHC → IHC (In-dataset)
BERT	CE	0.568	0.531	0.777
BERT (Aug)	CE	0.565	0.538	0.777
BERT	CE + *SCL*	0.560	0.537	0.777
BERT	CE + *AugCon*	0.581	0.546	0.774
BERT	CE + *ImpCon*	0.607	**0.579**	0.780
BERT	CE + *AugCon* + *ImpCon*	**0.611**	0.577	0.779
HateBERT	CE	0.587	0.547	0.764
HateBERT (Aug)	CE	0.555	0.528	0.763
HateBERT	CE + *SCL*	0.559	0.528	0.767
HateBERT	CE + *AugCon*	0.584	0.545	0.765
HateBERT	CE + *ImpCon*	**0.635**	**0.594**	0.774
HateBERT	CE + *AugCon* + *ImpCon*	0.630	0.591	0.772

Table 3: Cross-dataset and in-dataset evaluation results for the models trained on IHC dataset. We use → to distinguish the dataset used for training (on the left) and the dataset used for evaluation (on the right). For example, IHC → SBIC means the setting where a model is trained on IHC and then evaluated on SBIC. Boldfaced values on cross-dataset evaluation denote the best performance among different training objectives.

Model	Objective	SBIC → IHC (Cross-dataset)	SBIC → DynaHate (Cross-dataset)	SBIC → SBIC (In-dataset)
BERT	CE	0.596	0.603	0.838
BERT (Aug)	CE	0.601	0.604	0.833
BERT	CE + *SCL*	0.594	0.610	0.838
BERT	CE + *AugCon*	0.597	**0.612**	0.833
BERT	CE + *ImpCon*	**0.614**	**0.612**	0.836
BERT	CE + *AugCon* + *ImpCon*	0.596	0.603	0.838
HateBERT	CE	0.587	0.598	0.840
HateBERT (Aug)	CE	0.591	0.599	0.844
HateBERT	CE + *SCL*	0.593	0.598	0.843
HateBERT	CE + *AugCon*	0.585	0.595	0.841
HateBERT	CE + *ImpCon*	**0.599**	**0.606**	0.848
HateBERT	CE + *AugCon* + *ImpCon*	0.590	0.603	0.843

Table 4: Cross-dataset and in-dataset evaluation results for the models trained on SBIC dataset. Boldfaced values on cross-dataset evaluation denote the best performance among different training objectives.

the training data and train a model using cross-entropy loss. For data augmentation, we use the same augmentation used in *AugCon*, which substitutes 30% of words with their synonyms using WordNet following Suresh and Ong (2021) [2].

- **Cross-entropy Loss (CE) with Supervised Contrastive Learning**: we fine-tune each model using supervised contrastive learning (*SCL*) (Gunel et al., 2021) combined with cross-entropy loss. In *SCL*, posts from the same class are pulled close while others are pushed apart in the representation space [3].

[2]We use the nlpaug library (https://nlpaug.readthedocs.io/en/latest/augmenter/word/synonym.html) to implement the synonym substitution.

[3]In detail, given a post (e.g., hateful post), among $2N-1$ input posts and augmented posts except for the given post in a batch, posts that have the same class (e.g. hate class) as the given post are selected as positive samples.

4.3 Implementation Details

We use the pre-trained language model BERT-base-uncased as a base model, since it (and its variants) has shown state-of-the-art performance in hate speech detection (Swamy et al., 2019; Mathew et al., 2021). We also conduct experiments with HateBERT, which shows better generalization ability than BERT in the experiment of Caselli et al. (2021).

We train models for 6 epochs with NVIDIA RTX 3090. For hyperparameter, we search learning rate from {5e-6, 1e-5, 2e-5, 3e-5, 5e-5}, temperature τ from {0.1, 0.3, 0.5}, λ from {0.25, 0.5, 0.75} and choose the best model with validation F1 score. We run all experiments on 5 seeds (0, 1, 2, 3, 4) and report the F1 score on the test set.

4.4 Experiment Results

Table 3 and Table 4 shows the cross-dataset evaluation results for the models trained on IHC and SBIC respectively along with in-dataset evalua-

tion results. We investigate whether *AugCon* and *ImpCon* can improve the cross-dataset evaluation performance when combined with cross-entropy loss.

In cross-dataset evaluation, which we mainly focus on, simply adding augmented posts to the training set is not effective. Also, leveraging label information for contrastive learning (*SCL*) is less effective than our approaches. These results could be attributed to coarse-grained label (only two classes) in our task, which is in line with the results from Suresh and Ong (2021).

Adding *AugCon* on BERT increases the performance (at most 2.92% improvement) while adding it on HateBERT shows slight decrease. This indicates limited effectiveness of *AugCon*, particularly when adapted to a domain-shifted pre-trained language model. However, the models trained with *ImpCon* consistently outperform the models trained only with cross-entropy loss; we obtain at most 9.10% improvement when applied to BERT and 8.71% improvement when applied to HateBERT. This demonstrates the effectiveness of using *ImpCon* on generalization ability.

We also experimented the combination of *AugCon* and *ImpCon* with the same scaling factor between them. Only one result shows 0.58% performance improvement compared to the best performing *ImpCon* result. We analyze the possible reason in Section 5.1. Regarding the relatively low improvement on the models trained on SBIC, broader definition of class (offensiveness) and thus lower proportion of implication (not all offensive posts have implications) in offensive-labeled posts would be a reason.

For in-dataset evaluation, adding *AugCon* or *ImpCon* or combination of them (*i.e.*, *AugCon* and *ImpCon*) does not compromise the performance. We note that in-dataset performance can be overestimated, and cross-dataset evaluation results is rather perceived as better evaluation for measuring generalization ability.

5 Analysis

5.1 Representation Analysis

We focus on investigating the effect of *ImpCon* on the representation space. Since *ImpCon* pulls a paired post-implication in the representation space, we analyze the representation of post-implication pairs quantitatively and qualitatively. Although the model trained with *ImpCon* would project post-implication pairs of the training set close, it is unknown whether the model can project unseen post-implication pairs close. Hence, we conduct analysis using post-implication pairs in the validation set, which are unseen while training. We use the representation of $[CLS]$ token for the following two analyses. For the uniformity between analyses, we use the same BERT and HateBERT models trained on IHC training set on a seed.

Model	Objective	Sim.
BERT	CE	0.27
BERT	CE + *AugCon*	0.15
BERT	CE + *ImpCon*	**0.68**
BERT	CE + *AugCon* + *ImpCon*	0.60
HateBERT	CE	0.42
HateBERT	CE + *AugCon*	0.17
HateBERT	CE + *ImpCon*	**0.67**
HateBERT	CE + *AugCon* + *ImpCon*	0.54

Table 5: Quantitative analysis on the representation learned by different training objectives. Using each model fine-tuned with one of the training objectives, we calculated the averaged cosine similarity between all post-implication pairs of IHC validation set.

Quantitative Analysis We compute averaged cosine similarity between all post-implication pairs of IHC validation set. As shown in Table 5, two training objectives with *ImpCon* (CE + *ImpCon*, CE + *AugCon* + *ImpCon*) show higher similarity than others. The similarity gains of *ImpCon*-based training objectives compared to CE validate that *ImpCon* enables a model to project unseen post-implication pairs close. While CE + *ImpCon* shows the highest cosine similarity (0.6752 on BERT, 0.6731 on HateBERT), CE + *AugCon* + *ImpCon* shows lower cosine similarity (0.6048 on BERT, 0.5399 on HateBERT). Considering the lowest similarity CE + *AugCon* showed, *AugCon* seems to prevent post-implication pairs from being pulled close. We conjecture that this is one of the reasons why simply combining *AugCon* and *ImpCon* does not yield the best performance on 3 out of 4 cross-dataset evaluations.

Qualitative Analysis We visualize the learned representation of post-implication pairs from the IHC validation set using t-SNE (van der Maaten and Hinton, 2008). As shown in Figure 3, the representation learned by the training objectives with

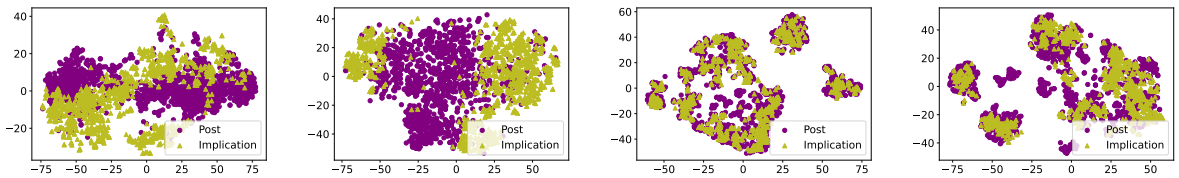

(a) CE (b) CE + *AugCon* (c) CE + *ImpCon* (d) CE + *AugCon* + *ImpCon*

Figure 3: Visualization of implicit hate posts and implications in IHC validation set using t-SNE. We use BERT model trained on IHC training set with each training objective.

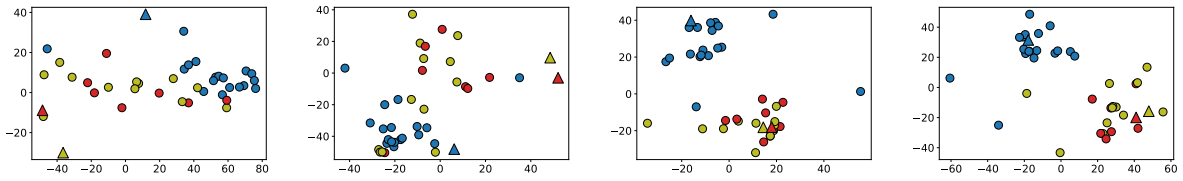

(a) CE (b) CE + *AugCon* (c) CE + *ImpCon* (d) CE + *AugCon* + *ImpCon*

Figure 4: Visualization of three example implications and their corresponding posts using t-SNE. A triangle-marker indicates an implication and a circle-marker indicates a post, respectively. Same colored posts (circle-markers) share the common implication, *i.e.*, triangle-marker colored in the same color. Blue triangle: "Immigrants should be deported", red triangle: "White people are superior", and yellow triangle: "Non-white people are inferior". We use BERT model trained on IHC training set with each training objective.

ImpCon (CE + *ImpCon*, CE + *AugCon* + *ImpCon*) shows that the implications and posts are more densely clustered in representation space compared to others, which is in line with our quantitative analysis.

However, it is necessary to investigate whether semantically relevant implications and posts are projected closer than irrelevant ones. As an in-depth analysis, we carefully select some implications and visualize them with their corresponding posts (Figure 4). We choose three implications, where two implications share similar semantic meaning ("White people are superior" and "Non-white people are inferior") and the other implication has different semantic meaning ("Immigrants should be deported"). We observe that the models with *ImpCon*-based training objectives (CE + *ImpCon*, CE + *AugCon* + *ImpCon*) produce distinguishable representation between implications with different meanings (blue points are apart from red and yellow points). Also, implications (and their corresponding posts) with similar meaning are densely clustered (red and yellow points are close to each other). The results suggest that semantically relevant implications and posts are projected closer than irrelevant ones in the representation space learned by *ImpCon*. We believe this is one of the reasons why *ImpCon* shows improvements on cross-dataset evaluation.

5.2 Case Study

We have shown that *ImpCon* forces relevant implications and posts to be projected closer in the representation space, quantitatively and qualitatively. As a further investigation, we observe whether the model can predict consistently on relevant posts that share the same implication.

We conduct a case study with two groups of hateful posts—one group shares the implication "jews are worthless" and the other group shares "black folks are animals"—in SBIC validation set. We report prediction results using BERT models trained on IHC. In Table 6, the model trained with *ImpCon* shows consistently correct predictions on posts that share a same implication. For example, for an implication "jews are worthless", while BERT trained with cross-entropy loss succeeds in predicting correctly on post 1 and 2, it fails to predict correctly on post 3 even though three posts share a same implication. However, the model trained with *ImpCon* predicts all corresponding posts as offensive correctly.

Implication 1 jews are worthless		
Post 1 "How do you turn a Christian into a Jew? Cremate him"	**Post 2** "Roses are red, violets are blue, I saw the KKK beating a Jew."	**Post 3** "What is the difference between a jew and a pizza : I don't like eating pizza"
Label: 1 BERT (CE): 1 BERT (CE +*ImpCon*): 1	Label: 1 BERT (CE): 1 BERT (CE +*ImpCon*): 1	Label: 1 BERT (CE): 0 BERT (CE +*ImpCon*): 1
Implication 2 black folks are animals		
Post 1 "Why are there no black furries? Because they are already animals."	**Post 2** "What's the difference between KFC and the KKK? Nothing.. They both fry animals"	**Post 3** "I don't believe in slavery I support animal rights"
Label: 1 BERT (CE): 1 BERT (CE +*ImpCon*): 1	Label: 1 BERT (CE): 0 BERT (CE +*ImpCon*): 1	Label: 1 BERT (CE): 0 BERT (CE +*ImpCon*): 1

Table 6: Some implications and their corresponding posts from SBIC validation set. We compare predictions of two BERT models trained on IHC training set with different training objectives (CE and CE + *ImpCon*). Label 1 denotes there exist offensive contents in the post.

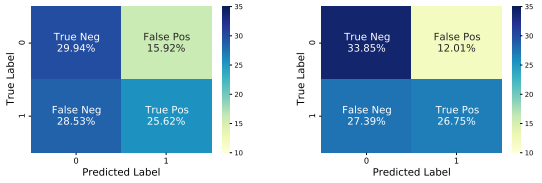

(a) CE (b) CE + *ImpCon*

Figure 5: Confusion matrices for the model predictions on SBIC validation set. We compare the predictions of two BERT models trained on IHC training set with (a) CE and (b) CE + *ImpCon*.

5.3 Error Analysis

We conduct an error analysis on cross-dataset evaluation to facilitate further studies. We provide the confusion matrices (Figure 5) where the models are trained on IHC and evaluated on SBIC. While *ImpCon* decreased false negatives and false positives, there are still considerable amount of errors. It is notable that 27.39% of predictions are false negatives, which takes a higher proportion than false positives (Figure 5(b)). We inspect such samples, and we suspect a target group that rarely appears in the training set would lead to false negatives. For example, given a hateful post with rare target group *anorexic folks* [4], "What do you call an anorexic with a yeast infection? A Quarter-Pounder with Cheese.", the model trained with CE + *ImpCon* predicts it as a non-offensive post. Since hate speeches on different target groups are based on distinct characteristics (stereotypes) of each group, hate speech on unseen target would limit the generalization ability of the model. Developing a training approach that can generalize well to unseen target groups would be a possible future direction.

[4]Using 'anorexic' as a keyword, there is no exact matching results in IHC training set.

6 Conclusions

We study the cross-dataset underperforming problem in implicit hate speech detection task. Empirically, we confirm that the pre-trained language models fine-tuned on an implicit hate speech dataset show relatively low performance on cross-dataset evaluation. We suggest leveraging contrastive learning when fine-tuning implicit hate speech detector to improve generalization ability. Particularly, we propose to utilize shared implication as a positive sample for its corresponding hateful posts, and introduce an implication-based contrastive learning method (*ImpCon*). Extensive experiments suggest that fine-tuning with *ImpCon* leads to better generalization ability, resulting in consistent performance improvements on all cross-dataset evaluation with three implicit hate speech datasets.

Acknowledgements

This research was supported by the NRF grant (NRF-2020R1A4A3079947), IITP grant (No. 2021-0-00354) and the AI Graduate School Program (No. 2020-0-01361) funded by the Korea government (MSIT). Han is a corresponding author.

References

Aymé Arango, Jorge Pérez, and Barbara Poblete. 2019. Hate speech detection is not as easy as you may think: A closer look at model validation. In *Proceedings of the 42nd International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 45–54.

Pinkesh Badjatiya, Shashank Gupta, Manish Gupta, and Vasudeva Varma. 2017. Deep learning for hate speech detection in tweets. In *Proceedings of the 26th International Conference on World Wide Web Companion*, pages 759–760.

Valerio Basile, Cristina Bosco, Elisabetta Fersini, Debora Nozza, Viviana Patti, Francisco Manuel Rangel Pardo, Paolo Rosso, and Manuela Sanguinetti. 2019. SemEval-2019 task 5: Multilingual detection of hate speech against immigrants and women in Twitter. In *Proceedings of the 13th International Workshop on Semantic Evaluation*, pages 54–63.

Tommaso Caselli, Valerio Basile, Jelena Mitrović, and Michael Granitzer. 2021. HateBERT: Retraining BERT for abusive language detection in English. In *Proceedings of the 5th Workshop on Online Abuse and Harms (WOAH 2021)*, pages 17–25.

Tommaso Caselli, Valerio Basile, Jelena Mitrović, Inga Kartoziya, and Michael Granitzer. 2020. I feel offended, don't be abusive! implicit/explicit messages in offensive and abusive language. In *Proceedings of the 12th Language Resources and Evaluation Conference*, pages 6193–6202.

Ting Chen, Simon Kornblith, Mohammad Norouzi, and Geoffrey Hinton. 2020. A simple framework for contrastive learning of visual representations. In *Proceedings of the 37th International Conference on Machine Learning*, pages 1597–1607.

Ying Chen, Yilu Zhou, Sencun Zhu, and Heng Xu. 2012. Detecting offensive language in social media to protect adolescent online safety. In *2012 International Conference on Privacy, Security, Risk and Trust and 2012 International Confernece on Social Computing*, pages 71–80.

Thomas Davidson, Dana Warmsley, Michael Macy, and Ingmar Weber. 2017. Automated hate speech detection and the problem of offensive language. *Proceedings of the International AAAI Conference on Web and Social Media*, 11(1):512–515.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186.

Mai ElSherief, Caleb Ziems, David Muchlinski, Vaishnavi Anupindi, Jordyn Seybolt, Munmun De Choudhury, and Diyi Yang. 2021. Latent hatred: A benchmark for understanding implicit hate speech. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 345–363.

Hongchao Fang, Sicheng Wang, Meng Zhou, Jiayuan Ding, and Pengtao Xie. 2020. CERT: Contrastive self-supervised learning for language understanding. *arXiv prerpint CoRR*, 2005.12766.

Antigoni-Maria Founta, Constantinos Djouvas, Despoina Chatzakou, Ilias Leontiadis, Jeremy Blackburn, Gianluca Stringhini, Athena Vakali, Michael Sirivianos, and Nicolas Kourtellis. 2018. Large scale crowdsourcing and characterization of Twitter abusive behavior. In *Proceedings of the 12th International AAAI Conference on Web and Social Media*, pages 491–500.

Björn Gambäck and Utpal Kumar Sikdar. 2017. Using convolutional neural networks to classify hate-speech. In *Proceedings of the First Workshop on Abusive Language Online*, pages 85–90.

Tianyu Gao, Xingcheng Yao, and Danqi Chen. 2021. SimCSE: Simple contrastive learning of sentence embeddings. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 6894–6910.

John Giorgi, Osvald Nitski, Bo Wang, and Gary Bader. 2021. DeCLUTR: Deep contrastive learning for unsupervised textual representations. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 879–895.

Njagi Dennis Gitari, Zhang Zuping, Hanyurwimfura Damien, and Jun Long. 2015. A lexicon-based approach for hate speech detection. *International Journal of Multimedia and Ubiquitous Engineering*, 10(4):215–230.

Tommi Gröndahl, Luca Pajola, Mika Juuti, Mauro Conti, and N. Asokan. 2018. All you need is "Love": Evading hate speech detection. In *Proceedings of the 11th ACM Workshop on Artificial Intelligence and Security*, pages 2–12.

Beliz Gunel, Jingfei Du, Alexis Conneau, and Veselin Stoyanov. 2021. Supervised contrastive learning for pre-trained language model fine-tuning. In *International Conference on Learning Representations*.

Xiaochuang Han and Yulia Tsvetkov. 2020. Fortifying toxic speech detectors against veiled toxicity. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 7732–7739.

Thomas Hartvigsen, Saadia Gabriel, Hamid Palangi, Maarten Sap, Dipankar Ray, and Ece Kamar. 2022. ToxiGen: A large-scale machine-generated dataset

for adversarial and implicit hate speech detection. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 3309–3326.

Prannay Khosla, Piotr Teterwak, Chen Wang, Aaron Sarna, Yonglong Tian, Phillip Isola, Aaron Maschinot, Ce Liu, and Dilip Krishnan. 2020. Supervised contrastive learning. In *Advances in Neural Information Processing Systems*, pages 18661–18673.

Ho-Suk Lee, Hong-Rae Lee, Jun-U Park, and Yo-Sub Han. 2018. An abusive text detection system based on enhanced abusive and non-abusive word lists. *Decision Support Systems*, 113:22–31.

Ju-Hyoung Lee, Jun-U Park, Jeong-Won Cha, and Yo-Sub Han. 2019. Detecting context abusiveness using hierarchical deep learning. In *Proceedings of the Second Workshop on Natural Language Processing for Internet Freedom: Censorship, Disinformation, and Propaganda*, pages 10–19.

Weiyang Liu, Yandong Wen, Zhiding Yu, and Meng Yang. 2016. Large-margin softmax loss for convolutional neural networks. In *Proceedings of The 33rd International Conference on Machine Learning*, pages 507–516.

Ana Marasović, Iz Beltagy, Doug Downey, and Matthew Peters. 2022. Few-shot self-rationalization with natural language prompts. In *Findings of the Association for Computational Linguistics: NAACL 2022*, pages 410–424.

Binny Mathew, Punyajoy Saha, Seid Muhie Yimam, Chris Biemann, Pawan Goyal, and Animesh Mukherjee. 2021. HateXplain: A benchmark dataset for explainable hate speech detection. *Proceedings of the AAAI Conference on Artificial Intelligence*, 35(17):14867–14875.

John T Nockleby. 2000. Hate speech. *Encyclopedia of the American constitution*, 3(2):1277–1279.

Lin Pan, Chung-Wei Hang, Avirup Sil, and Saloni Potdar. 2022. Improved text classification via contrastive adversarial training. *Proceedings of the AAAI Conference on Artificial Intelligence*, 36(10):11130–11138.

Ji Ho Park and Pascale Fung. 2017. One-step and two-step classification for abusive language detection on Twitter. In *Proceedings of the First Workshop on Abusive Language Online*, pages 41–45.

Nils Rethmeier and Isabelle Augenstein. 2022. A primer on contrastive pretraining in language processing: Methods, lessons learned & perspectives. *ACM Computing Surveys (CSUR)*.

Maarten Sap, Saadia Gabriel, Lianhui Qin, Dan Jurafsky, Noah A. Smith, and Yejin Choi. 2020. Social bias frames: Reasoning about social and power implications of language. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 5477–5490.

Varsha Suresh and Desmond Ong. 2021. Not all negatives are equal: Label-aware contrastive loss for fine-grained text classification. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 4381–4394.

Steve Durairaj Swamy, Anupam Jamatia, and Björn Gambäck. 2019. Studying generalisability across abusive language detection datasets. In *Proceedings of the 23rd Conference on Computational Natural Language Learning (CoNLL)*, pages 940–950.

Yonglong Tian, Chen Sun, Ben Poole, Dilip Krishnan, Cordelia Schmid, and Phillip Isola. 2020. What makes for good views for contrastive learning? In *Advances in Neural Information Processing Systems*, pages 6827–6839.

Laurens van der Maaten and Geoffrey Hinton. 2008. Visualizing data using t-sne. *Journal of Machine Learning Research*, 9(86):2579–2605.

Bertie Vidgen, Tristan Thrush, Zeerak Waseem, and Douwe Kiela. 2021. Learning from the worst: Dynamically generated datasets to improve online hate detection. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 1667–1682.

Kunze Wang, Dong Lu, Caren Han, Siqu Long, and Josiah Poon. 2020. Detect all abuse! toward universal abusive language detection models. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 6366–6376.

William Warner and Julia Hirschberg. 2012. Detecting hate speech on the world wide web. In *Proceedings of the Second Workshop on Language in Social Media*, page 19–26.

Michael Wiegand, Josef Ruppenhofer, and Elisabeth Eder. 2021. Implicitly abusive language – what does it actually look like and why are we not getting there? In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 576–587.

Michael Wiegand, Josef Ruppenhofer, and Thomas Kleinbauer. 2019. Detection of Abusive Language: the Problem of Biased Datasets. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 602–608.

Michael Wiegand, Josef Ruppenhofer, Anna Schmidt, and Clayton Greenberg. 2018. Inducing a lexicon of abusive words – a feature-based approach. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 1046–1056.

Wenjie Yin and Arkaitz Zubiaga. 2021. Towards generalisable hate speech detection: a review on obstacles and solutions. *PeerJ Computer Science*, 7:e598.

Ziqi Zhang, David Robinson, and Jonathan Tepper. 2018. Detecting hate speech on twitter using a convolution-GRU based deep neural network. In *The Semantic Web*, pages 745–760.

Xiangyun Zhao, Raviteja Vemulapalli, Philip Andrew Mansfield, Boqing Gong, Bradley Green, Lior Shapira, and Ying Wu. 2021. Contrastive learning for label efficient semantic segmentation. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 10623–10633.

A Visualization by t-SNE (HateBERT)

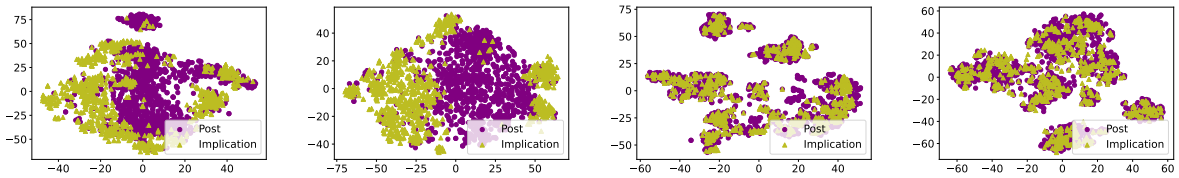

(a) CE (b) CE + *AugCon* (c) CE + *ImpCon* (d) CE + *AugCon* + *ImpCon*

Figure 6: Visualization of implicit hate posts and implications in IHC validation set using t-SNE. We use HateBERT model trained on IHC training set with each training objective.

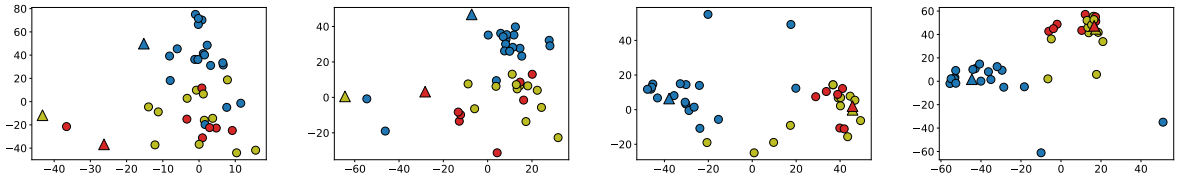

(a) CE (b) CE + *AugCon* (c) CE + *ImpCon* (d) CE + *AugCon* + *ImpCon*

Figure 7: Visualization of three example implications and their corresponding posts using t-SNE. A triangle-marker indicates an implication and a circle-marker indicates a post, respectively. Same colored posts (circle-markers) share the common implication, *i.e.*, triangle-marker colored in the same color. Blue triangle: "Immigrants should be deported", red triangle: "White people are superior", and yellow triangle: "Non-white people are inferior". We use HateBERT model trained on IHC training set with each training objective.

B Error Analysis (HateBERT)

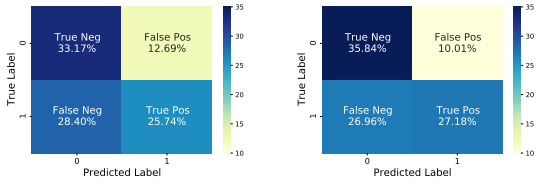

(a) CE (b) CE + *ImpCon*

Figure 8: Confusion matrices for the model predictions on SBIC validation set. We compare the predictions of two HateBERT models trained on IHC training set with (a) CE and (b) CE + *ImpCon*.

Social Bot-Aware Graph Neural Network for Early Rumor Detection

Zhen Huang[1,*], **Zhilong Lv**[1,*], **Xiaoyun Han**[1,*], **Binyang Li**[2]✉, **Menglong Lu**[1], **Dongsheng Li**[1]
[1]National University of Defense Technology, China
[2]University of International Relations, China
[1]{huangzhen, lvzhilong, hxy_1021, lumenglong, dsli}@nudt.edu.cn
[2]byli@uir.edu.cn

Abstract

Early rumor detection is a key challenging task to prevent rumors from spreading widely. Sociological research shows that social bots' behavior in the early stage has become the main reason for rumors' wide spread. However, current models do not explicitly distinguish genuine users from social bots, and their failure in identifying rumors timely. Therefore, this paper aims at early rumor detection by accounting for social bots' behavior, and presents a Social Bot-Aware Graph Neural Network, named SBAG. SBAG firstly pre-trains a multi-layer perception network to capture social bot features, and then constructs multiple graph neural networks by embedding the features to model the early propagation of posts, which is further used to detect rumors. Extensive experiments on three benchmark datasets show that SBAG achieves significant improvements against the baselines and also identifies rumors within 3 hours while maintaining more than 90% accuracy.

1 Introduction

Rumor is defined as unverified information at the time of posting (Qazvinian et al., 2011; Zubiaga et al., 2018; Lu et al., 2022). Malicious rumors that are spread massively on social media have become a threat to mislead the public and cause social panic. There is a need to debunk rumors in the early stage so as to prevent rumors from the wide spread.

Sociological research (Shao et al., 2018a) shows that there exist social bots during the rumor spread, and these bots are particularly active in the early stage of rumors, which will affect the real users through replies and mentions and accelerate the spread (Shao et al., 2018b; Beskow and Carley, 2018; Feng et al., 2022).

Current models (Ma et al., 2016; Chen et al., 2018; Song et al., 2019; Zhou et al., 2019; Xia et al., 2020; Han et al., 2021) mainly focus on post content or propagation sequence. These methods model posts as a chronological sequence, and extract the textual feature through GRU, LSTM, and CNN for rumor detection. Other methods (Liu and Wu, 2018; Yuan et al., 2020) account for early rumor detection by modeling user characteristics or credibility with user propagation structures. However, they do not explicitly distinguish genuine users from social bots, so the participation of social bots will lead to the failure of the features captured from both contents and propagation structures.

To this end, this paper presents a model named Social Bot-Aware Graph Neural Network (SBAG) for early rumor detection. This model consists of two parts: Social bot Detection (SD) and Bot-Aware Graph Rumor Detection (BAG). The former one is pre-trained based on a large sample of bot users and genuine users, to extract the features to compute the bot possibility for each user. The latter one transfers the SD to the Bot-Aware Graph Neural Network, which consists of GNN-based User Publishing (GUP), GAT-based User Interaction (GUI), and textual encoder components. For GUP, the bot possibility computed in SD is involved in the aggregation process. For GUI, the bot possibility is also integrated into the calculation of the attention weight of user-user. The textual encoder utilizes a convolutional neural network (CNN) to capture textual features. In this way, we take user publishing features, user interaction features, and textual features into consideration for early rumor detection. The codes will be open sourced[1]. Our main contributions are summarized as follows:

- According to the observation of sociological research, we consider social bots' behaviors, and train a social bot detection model based on twelve datasets. The results prove the consistency with the sociological research that the

*These authors contributed equally to this work.
✉Corresponding authors

[1]https://github.com/sky-star-moon/SBAG

Proceedings of the 29th International Conference on Computational Linguistics, pages 6680–6690
October 12–17, 2022.

bots are very active in the early stage.

- We propose a method named SBAG for early rumor detection, which implements early rumor detection by incorporating social bot detection. The results demonstrate that SBAG can achieve more than 93% accuracy, and detect 90% rumors within 3 hours.

2 Problem Definition

Assume a set of posts $\mathbf{R} = \{r_1, r_2, ..., r_{|\mathbf{R}|}\}$ and a set of users $\mathbf{U} = \{u_1, u_2, ..., u_{|\mathbf{U}|}\}$. Each post r corresponds to one publisher and multiple users to repost it. A user publishing graph $G_p =< V_p, E_p >$ is constructed to denote publisher-post relations, where V_p is the set of all publishers and source posts, E_p is the set of edges and a edge (u_i, r_j) indicates that user u_i publishes post r_j. A user interaction graph $G_u =< V_u, E_u >$ is constructed to denote user-user relations, where V_u is the set of all users, E_u is the set of edges and a edge (u_i, u_j) indicates that user u_i replies user u_j.

Since our motivation is to debunk rumors by incorporating the influence of social bots, our goal is to learn two classifiers for social bot detection and rumor detection, respectively. For the social bot detection task, a classifier $g : u \rightarrow Y_u$ is learned to identify whether a user u is a bot user or a genuine user. For the rumor detection task, a classifier is learned $f : r \rightarrow Y_r$ to predict the class of each source post r.

3 SBAG Model

The framework of the SBAG model is shown in Fig. 1. The model consists of two main parts, Social Bot Detection (SD) mentioned in §3.1 and Bot-Aware Graph Rumor Detection (BAG) mentioned in §3.2. We will introduce each module in detail.

3.1 Social Bot Detection

To incorporate bot behavior information into the model, we first pre-train the SD module on twelve datasets to learn the features of genuine users and bot users. Then we transfer this module to BAG as a bot possibility scorer, which assists in capturing the propagation pattern of bot behavior.

During the pre-training stage, we use a Multilayer Perceptron (MLP) as the backbone network. Formally, let $c \in \mathbb{R}^v$ denote the user characteristics, such as length of username, number of followers, etc. Then c is normalized and fed into the module. The process is as follows:

$$\tilde{c} = \tanh(W_c^T c + b_c) \quad (1)$$

$$\hat{Y_u} = \text{softmax}(W_u^T \tilde{c} + b_u) \quad (2)$$

where $W_c \in \mathbb{R}^{v \times v}$, $W_u \in \mathbb{R}^{v \times 2}$, $b_c \in \mathbb{R}^v$ and $b_u \in \mathbb{R}^2$ are the parameters of the MLP, $\hat{Y_u} \in \{bot, human\}$ is the predicted probability distribution of the user class.

SD module will compute the users' bot possibility within [0,1] to indicate the degree to the user shows social bot behavior.

3.2 Bot-Aware Graph Neural Network

3.2.1 GCN-based User Publishing

Since user publishing graph G_p is a bipartite graph with only one hop at most, it is well locality. We design a GCN-based user publishing component. Formally, let $P \in \mathbb{R}^{m \times d}$ denote the initial embedding of the user nodes and $C \in \mathbb{R}^{n \times d}$ denote the initial embedding of the source post nodes, where m and n are the number of publisher nodes and source post nodes respectively, and d is the embedding dimension. We construct the adjacency matrix $A \in \mathbb{R}^{m \times n}$ base on G_p, where the element A_{ij} denotes user u_i publishes post r_j, then we normalize A to the matrix $\hat{A} = D_m^{-\frac{1}{2}} A D_n^{-\frac{1}{2}}$, where $(D_m)_{ii} = \sum_j A_{ij}$ and $(D_n)_{jj} = \sum_i A_{ij}$ are the diagonal matrices.

In order to incorporate the user's early bot possibility into the component, we constitute a bot possibility matrix $\hat{s} \in \mathbb{R}^{m \times d}$, where each element of row i of $\hat{s}$ is the bot possibility of publisher u_i and treat it as a bias. Finally, the aggregated features are summed with the initial features to obtain the publishing feature. The formulas are as follows:

$$\hat{C} = \text{ReLU}(\hat{A} C W_c + \hat{s}) \quad (3)$$

$$\hat{P} = \hat{C} + P \quad (4)$$

where $W_c \in \mathbb{R}^{d \times d}$ is the learnable matrix, $\hat{C} \in \mathbb{R}^{m \times d}$ is the aggregated feature, and $\hat{P} \in \mathbb{R}^{m \times d}$ is the publishing feature.

3.2.2 GAT-based User Interaction

In the user interaction graph G_u, considering the different importance of neighbor nodes to the target node, we design a GAT-based user interaction component.

Let $U^{(l)}$ denote the node features at layer l. Every user nodes' embedding in the graph is initialized to $U^{(0)} = \{u_0^{(0)}, u_1^{(0)}, ..., u_{|V_u|-1}^{(0)}\} \in \mathbb{R}^{|V_u| \times d}$

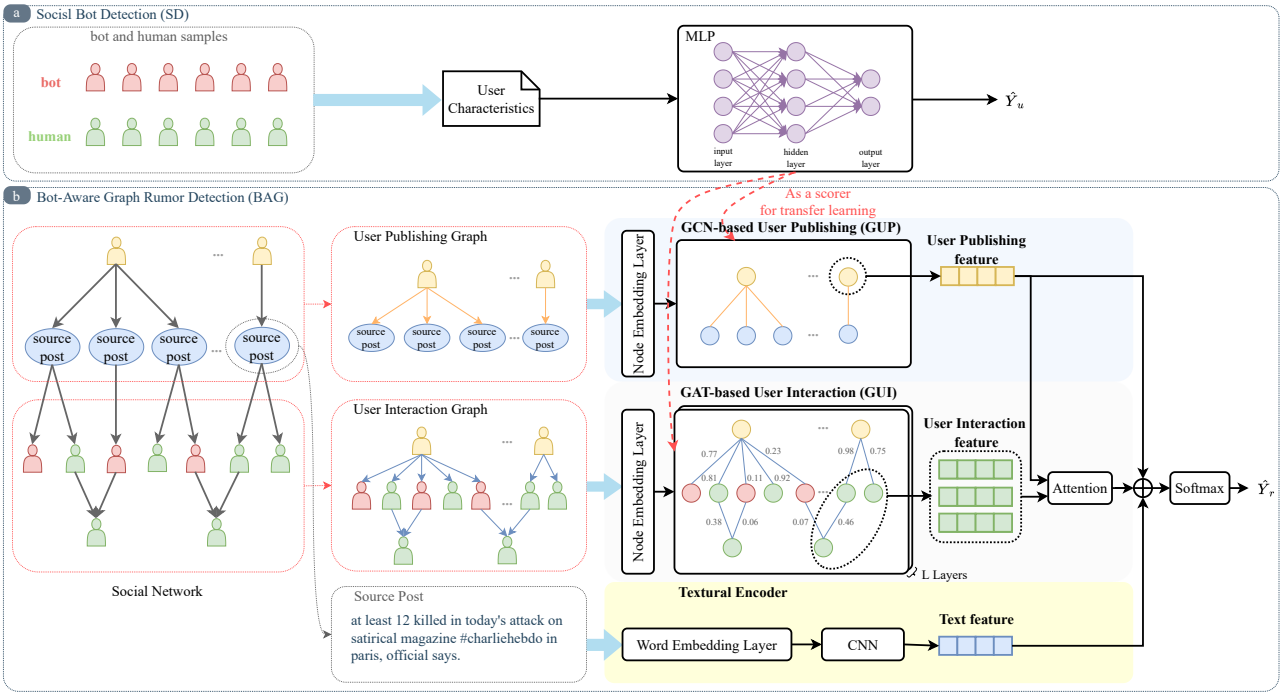

Figure 1: Overview of SBAG. **SD** scores the user's bot possibility according to user characteristics, then is transferred to the BAG module as a scorer. **BAG** module consists of three key components: GUP learns the publishing features of the publishers, GUI learns the interaction features of the repliers and textual encoder learns the textual features of the source post. Finally, the three types of features are fused to predict the class of the source post.

by a embedding layer according to normal distribution, and d is the dimension of the node embedding. Referring to the multi-head attention mechanism, the node feature at layer $l+1$ is updated as follows:

$$u_i^{(l+1)} = \Big\Vert_{k=1}^{K} \text{ReLU}\Big(\sum_{j \in \mathcal{N}(i)} \alpha_{ij}^{(l,k)} W_v^{(l,k)} u_j^{(l)} \Big) \quad (5)$$

where $\|$ denotes the concatenating operation. $\mathcal{N}(i)$ is the set of node i and its direct neighbors. $\alpha_{ij}^{(l,k)}$ is the attention weight of neighbor node j to target node i at the l-th layer in the k-th head, and $W_v^{(l,k)}$ is the learnable transformation matrix. Particularly, the output embedding in the last layer (denoted as the L-th layer) is the average of the features from the K heads instead of the concatenation. The formula is as follows:

$$u_i^{(L)} = \text{ReLU}\Big(\frac{1}{K} \sum_{k=1}^{K} \sum_{j \in \mathcal{N}(i)} \alpha_{ij}^{(L-1,k)} W_v^{(L-1,k)} u_j^{(L-1)} \Big) \quad (6)$$

To capture the propagation pattern of bot behavior, we introduce bot possiblity into the attention weight $\alpha_{ij}^{(l,k)}$. Specifically, we utilize the SD mentioned in §3.1 to generate bot possibility s_i and s_j for two nodes of edge (u_i, u_j), and take their mean value as the edge weight e_{ij}, then $\alpha_{ij}^{(l,k)}$ is calculation as follows:

$$z_{ij}^{(l,k)} = \text{LeakyReLU}([W_q^{(l,k)} u_i \| W_k^{(l,k)} u_j] W_\alpha^l) \quad (7)$$

$$e_{ij} = \frac{s_i + s_j}{2} \quad (8)$$

$$\hat{z}_{ij}^{(l,k)} = e_{ij} \times z_{ij}^{(l,k)} \quad (9)$$

$$\alpha_{ij}^{(l,k)} = \frac{\exp(\hat{z}_{ij}^{(l,k)})}{\sum_{t \in \mathcal{N}(i)} \exp(\hat{z}_{ij}^{(l,k)})} \quad (10)$$

where $W_\alpha^{(l)}$, $W_q^{(l,k)}$ and $W_k^{(l,k)}$ are learnable parameters. Through the L graph attention layers, the interaction features $U^{(L)} = \{u_0^{(L)}, u_1^{(L)}, ..., u_{|V_u|-1}^{(L)}\}$ of all user nodes are obtained.

Next, for one source post r with one publisher and a repliers, the features of the publisher are $\tilde{P}_r \in \mathbb{R}^{1 \times d}$, the features of the repliers are obtained from $U^{(L)}$, denoted as $\tilde{U} \in \mathbb{R}^{a \times d}$. To distinguish the importance of the repliers to the publisher, we calculate the attention weights and then aggregate the features from $\tilde{U}$ into an interaction feature $\tilde{U}_r$:

$$\beta = \text{softmax}(\tilde{U} W_\beta \tilde{P}_r^T) \quad (11)$$

$$\tilde{U}_r = \beta^T \tilde{U} \quad (12)$$

where β is the vector of the attention weight, $W_\beta \in \mathbb{R}^{d \times d}$ is the trainable matrix.

3.2.3 Textual Encoder

The semantic features of the post text are also important for rumor detection. For In this component, we utilize a CNN, which is consistent with the baseline models like SMAN and GLAN for fairness, to encode the source post. Each source post can be represented as a sequence of word embeddings $\mathbf{X} = [x_1, x_2, ..., x_{|\mathbf{X}|}] \in \mathbb{R}^{|\mathbf{X}| \times d}$. In CNN, one element of a feature map obtained from $\mathbf{X}$ through the convolutional operation is as follows:

$$h_i = \text{ReLU}(< W_h, x_{i:i+\omega-1} >_{\mathcal{F}}) \quad (13)$$

where $W_h \in \mathbb{R}^{\omega \times d}$ is a convolution kernel of size ω, and $\mathcal{F}$ is the Frobenius inner product. The feature map can be represented as $h = [h_1, h_2, ..., h_{|\mathbf{X}|-\omega+1}] \in \mathbb{R}^{|\mathbf{X}|-\omega+1}$. We then extract the maximum value from the feature map h to obtain $\hat{h} = \max(h)$.

We utilizes d filters of different kernel sizes ω, where $\omega \in \{3, 4, 5\}$, to obtain various features. Finally, the output of each filters are concatenated to obtain the textual feature $\tilde{X} \in \mathbb{R}^{1 \times 3d}$.

3.2.4 Output Layer

Assume that the textual feature of the source post r is $\tilde{X}_r \in \mathbb{R}^{1 \times 3d}$, the publishing feature of the publisher is $\tilde{P}_r \in \mathbb{R}^{1 \times d}$, and the aggregated interaction feature is $\tilde{U}_r \in \mathbb{R}^{1 \times d}$. We concatenate the features from different types of the source post, i.e., $\tilde{P}_r$, $\tilde{U}_r$, and $\tilde{X}_r$, to obtain the final feature of the source post. Lastly, the final feature is fed into a fully connected layer to predict the class:

$$\hat{Y}_r = \text{softmax}(W_r^T [\tilde{X}_r \| \tilde{P}_r \| \tilde{U}_r]^T + b_r) \quad (14)$$

where $W_r \in \mathbb{R}^{5d \times c}$ and $b_r \in \mathbb{R}^c$ are the weight and bias of the fully connected layer, and $\hat{Y}_r \in \{rumor, non\text{-}rumor\}$ or $\hat{Y}_r \in \{non\text{-}rumor, false\ rumor, true\ rumor, unverified\ rumor\}$ is the predicted class distribution.

3.3 Training

We apply the cross-entropy loss to optimize the social bot detection task and rumor detection task. The loss functions are as follows:

$$L_u = -\sum_{i=1}^{T} Y_{u_i} log \hat{Y}_{u_i} \quad (15)$$

$$L_r = -\sum_{j=1}^{|\mathbf{R}|} Y_{r_j} log \hat{Y}_{r_j} \quad (16)$$

where L_u is the cross-entropy loss of the social bot detection task, Y_{u_i} and $\hat{Y}_{u_i}$ are the ground truth and predicted label of the i-th user respectively, and T is the size of the social bot detection dataset. L_r is the cross-entropy loss of the rumor detection task, Y_{r_j} and $\hat{Y}_{r_j}$ is the ground truth and predicted label of the j-th source post respectively, and $|\mathbf{R}|$ is the size of the rumor detection dataset.

4 Experiments

4.1 Datasets

For rumor detection task, we conduct experiments on three benchmark datasets: Twitter15 (Ma et al., 2017), Twitter16 (Ma et al., 2017), and Weibo16 (Ma et al., 2016). The statistics of the three datasets are shown in Tab. 1.

For fair comparison, we choose the same way of splitting datasets as in the baseline work(Yuan et al., 2020), 10% of samples are selected as the validation set, and the rest of samples are split into the training set and testing set with a ratio of 3:1.

Dataset	Twitter15	Twitter16	Weibo16
# Source posts	1,490	818	4,664
# Non-rumors (NR)	374	205	2,351
# False rumors (FR)	370	205	2,313
# True rumors (TR)	372	205	0
# Unverified rumors (UR)	374	203	0
# Users	276,663	173,487	2,746,818
# Posts	331,612	204,820	3,805,656

Table 1: Statistics of the rumor detection datasets.

For the social bot detection task, we select 12 datasets provided by Bot Repository[2] and the statistics of the datasets are shown in Tab. 2. The datasets are split into the training set, testing set, and validation set with the ratio of 8:1:1.

4.2 Experimental Settings

For the SD module, since Twitter15 and Twitter16 do not involve the user characteristics, we utilize Twitter API to crawl user characteristics based on user ID. The user characteristics selection of the three datasets is not exactly the same. The details are shown in Tab. 3.

For the BAG module, the dimension of the node embedding d is 100, the number of heads of the Multi-Head Attention K is 8, the number of graph attention network layers is 2, and the convolutional kernel sizes are {3,4,5}. The model utilizes the

[2]botometer.osome.iu.edu/bot-repository

Dataset	# bots	# humans
caverlee(Lee et al., 2011)	0	14,895
cresci-17(Cresci et al., 2017)	9,894	3,474
pronbots(Yang et al., 2019)	17,882	0
celebrity(Yang et al., 2019)	0	5,918
vendor-purchased(Yang et al., 2019)	1,088	0
gilani-17(Gilani et al., 2017)	0	1,413
cresci-rtbust(Mazza et al., 2019)	0	340
cresci-stock(Cresci et al., 2018)	0	6,174
botowiki(Yang et al., 2020)	698	0
midterm-2018(Yang et al., 2020)	17,968	8,092
verified(Yang et al., 2020)	0	1,987
TwiBot-20(Feng et al., 2021)	0	5,237
Total	47,530	47,530

Table 2: Statistics of the social bot detection datasets.

Adam optimizer with 1e-3 learning rate and 1e-6 weight decay coefficient. Besides, the batch size is set to 16 and the epoch is set to 20.

Similar to the existing work(Liu and Wu, 2018; Yuan et al., 2019, 2020), we also adopt Accuracy, Precision, Recall, and F1 score as the evaluation metrics.

user characteristic	Twitter15	Twitter16	Weibo16
Length of username	✓	✓	✓
Length of screenname	✓	✓	✓
Length of description	✓	✓	✓
Followers count	✓	✓	✓
Friends count	✓	✓	✓
Listed count	✓	✓	
Favorites count	✓	✓	✓
Statuses count	✓	✓	✓
Days of Registration	✓	✓	✓
URL	✓	✓	
Protected	✓	✓	
Geo enabled	✓	✓	✓
Verified	✓	✓	✓
Profile use background image	✓	✓	
Default profile	✓	✓	

Table 3: User characteristics selection.

4.3 Baselines

To evaluate the performance of SBAG, we compare SBAG with the following methods:

(1) **DTR** (Zhao et al., 2015) is a decision tree-based ranking approach that searches for inquiry phrases, clusters controversial posts, and then ranks the clustered results.

(2) **DTC** (Castillo et al., 2011) is a decision tree model that uses hand-crafted features of posts to detect rumors.

(3) **RFC** (Kwon et al., 2017) is a random forest classifier that learns user, linguistic, and structural features of posts for rumor detection.

(4) **SVM-RBF** (Yang et al., 2012) is an SVM model with an RBF kernel, which classifies rumors based on statistical features of posts.

(5) **SVM-TS** (Ma et al., 2015) is a linear SVM model, which uses dynamic series-time structure to capture social context features over time.

(6) **cPTK** (Ma et al., 2017) is an SVM model, which uses the tree-based kernel to evaluate the similarity of propagation tree structures.

(7) **GRU** (Ma et al., 2016) utilizes RNN to learn the textual feature of the chronological post sequences to detect rumors.

(8) **RvNN** (Ma et al., 2018) models the source post and its reposts as a conversation tree, and adopts a recursive neural network to learn its propagation pattern.

(9) **PPC** (Liu and Wu, 2018) employs RNN and CNN to model the sequence based on user features for early rumor detection.

(10) **GLAN** (Yuan et al., 2019) models posts and users as a heterogeneous graph, and identifies rumors by local semantic features and global structural features extracted from the graph neural network.

(11) **SMAN** (Yuan et al., 2020) jointly optimizes rumor detection task and users' credibility prediction task via a structure-aware multi-head attention network for early rumor detection.

4.4 Experimental Results

4.4.1 Analysis of Rumor Detection

Tab. 4 and Tab. 5 show the rumor detection results on Twitter15, Twitter16, and Weibo16. SBAG achieves 93.8%, 94.6%, and 95.7% in terms of accuracy on three datasets, respectively, and outperforms the best run of the baseline models.

More detailedly, compared with traditional machine learning models, such as SVM-RBF, SVM-TS, and cPTK, SBAG can capture a higher-level representation of posts. Moreover, SBAG outperforms textual feature-based methods such as GRU and RvNN, which proves that the social bot-aware user features are effective in rumor detection. In addition, compared with PPC, GLAN, and SMAN which capture user propagation features or user credibility, SBAG achieves a better performance. It is because that SBAG is beneficial for exploring the features of social bot behaviors.

4.4.2 Analysis of Early Detection

To evaluate the timeliness of SBAG, we set different detecting deadlines, where we only utilize the interaction of users before the deadline. Fig.

Method	Twitter15					Twitter16				
	Acc.	NR-F1	FR-F1	TR-F1	UR-F1	Acc.	NR-F1	FR-F1	TR-F1	UR-F1
DTR	0.409	0.501	0.311	0.364	0.473	0.414	0.394	0.273	0.630	0.344
DTC	0.454	0.733	0.355	0.317	0.415	0.465	0.643	0.393	0.419	0.403
RFC	0.565	0.810	0.422	0.401	0.543	0.585	0.752	0.415	0.547	0.563
SVM-RBF	0.318	0.455	0.037	0.218	0.225	0.321	0.423	0.085	0.419	0.037
SVM-TS	0.544	0.796	0.472	0.404	0.483	0.574	0.755	0.420	0.571	0.526
cPTK	0.750	0.804	0.698	0.765	0.733	0.732	0.740	0.709	0.836	0.686
GRU	0.646	0.792	0.574	0.608	0.592	0.633	0.772	0.489	0.686	0.593
RvNN	0.723	0.682	0.758	0.821	0.654	0.737	0.662	0.743	0.835	0.708
PPC	0.842	0.811	0.875	0.818	0.790	0.863	0.820	0.898	0.843	0.837
GLAN	0.905	0.924	0.917	0.852	0.927	0.902	0.921	0.869	0.847	0.968
SMAN	0.914	0.915	0.926	**0.933**	0.881	0.935	0.946	0.920	0.905	0.968
SBAG	**0.938**	**0.965**	**0.953**	0.897	**0.933**	**0.946**	**0.947**	**0.930**	**0.926**	**0.978**

Table 4: Results of rumor detection on Twitter15 and Twitter16.

Method	Acc.	NR			FR		
		Precision	Recall	F1	Precision	Recall	F1
DTR	0.732	0.726	0.749	0.737	0.738	0.715	0.726
DTC	0.831	0.815	0.847	0.830	0.847	0.815	0.831
RFC	0.849	0.947	0.739	0.830	0.786	0.959	0.864
SVM-RBF	0.818	0.815	0.824	0.819	0.822	0.812	0.817
SVM-TS	0.857	0.878	0.830	0.857	0.839	0.885	0.861
GRU	0.910	0.952	0.864	0.906	0.876	0.956	0.914
PPC	0.921	0.949	0.889	0.918	0.896	0.962	0.923
GLAN	0.946	0.949	0.943	0.946	0.943	0.948	0.945
SMAN	0.951	0.937	**0.967**	0.952	**0.967**	0.936	0.951
SBAG	**0.957**	**0.967**	0.947	**0.957**	0.947	**0.967**	**0.957**

Table 5: Results of rumor detection on Weibo16.

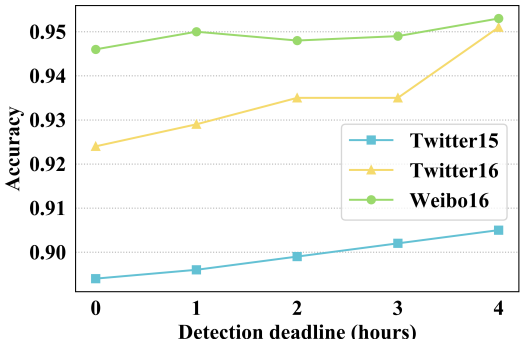

Figure 2: Results of timeliness.

2 shows the results on Twitter15, Twitter16, and Weibo16. We can observe that within 0 to 3 hours, SBAG achieves the accuracy of over 90% on three datasets, and the results in the early stage are close to the results by accounting for all users, which indicates that SBAG has a strong capability for early detection.

Fig. 3 shows the comparison of early detection with several baselines on Twitter15, Twitter16, and Weibo16. We can see that SBAG can debunk rumors earlier, and maintain a high accuracy, which even outperforms the state-of-the-art baselines such as SMAN and GLAN. Furthermore, within 24 hours, SBAG can achieve similar performance to those with learning the features of all users.

4.5 Ablation Study

To demonstrate the effectiveness of different features, we also conduct ablation study, and the experiments are as follows:

(1) **-p**: removing the GUP, the model predicts

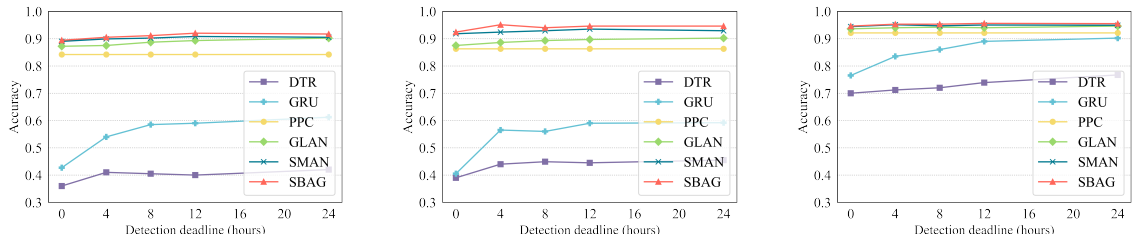

(a) Twitter15. (b) Twitter16. (c) Weibo16.

Figure 3: Results of early rumor detection on Twitter15, Twitter16 and Weibo16.

rumors by textual feature and interaction feature, without publisher feature.

(2) **-i**: removing the GUI, the model predicts rumors by textual feature and publishing feature, without interaction feature.

(3) **-p-i**: it means that we discard two components mentioned (1) and (2) and detect rumors by textual feature only.

(4) **-t**: removing the textual encoder, the model predicts rumors by publisher feature and interaction feature, without textual feature.

(5) **-s**: without the pre-trained scorer, we score the bot possibility randomly.

As shown in Tab. 6, we can observe that each component of SBAG is essential. Specifically, **-p** and **-i** perform worse than the original model on three datasets, which shows that the publishing feature and interaction feature are significant for rumor detection. There is a sharp decrease in **-p-i**, which indicates that it is suboptimal to detect rumors only by the textual feature. Besides, the performance of **-t** also decreases significantly. It is because the textual feature of the source post is crucial for detecting rumors. The results of **-p-i** and **-t** show that user features and textual features have a complementary relationship. The performance of **-s** demonstrates that social bot detection is beneficial for rumor detection.

Method	Twitter15	Twitter16	Weibo16
SBAG	**0.938**	**0.946**	**0.957**
-p	0.913	0.920	0.947
-i	0.894	0.924	0.946
-p-i	0.848	0.885	0.915
-t	0.658	0.723	0.919
-s	0.931	0.927	0.948

Table 6: Ablation Study (Acc.).

4.6 Analysis of Social Bot Detection

Then, we will analyze the performance of the pre-trained social bot detection module. As mentioned in Tab. 3, we select 15 and 10 user characteristics to represent users on Twitter and Weibo datasets, respectively. Therefore, we pre-train two social bot detection modules with different dimensions, i.e., MLP-15d and MLP-10d. For comparison, we choose the baseline models as follows:

(1) **Botometer-v4** (Sayyadiharikandeh et al., 2020) is a public program that can be used to evaluate the bot score of any user on Twitter.

(2) **AdaBoost** (Kudugunta and Ferrara, 2018) extracts 10 user characteristics to represent a user, and employs AdaBoost classifier for bot detection.

(3) **RF** (Yang et al., 2020) is a random forest model, which extracts 8 original features and 12 derived features from the user information, and utilizes the random forest classifier to identify users.

The results are shown in Tab. 7. The accuracy of MLP-15d and MLP-10d are better than the baselines, which indicates that they have a great ability to identify users. MLP-15d outperforms MLP-10d because the user information input to MLP-15d is richer. The MLP models perform better than the machine learning models like Botometer-V4, AdaBoost, and RF, which demonstrates that MLP can learn high-quality user representation with fewer features on this task.

4.7 Analysis of Social Bot Behavior

On the test sets of Twitter15, Twitter16, and Weibo16, we list the relation of rumors and publishers, i.e., the ratio of bot-behavior publishers under each source post class. As shown in Fig. 4a, we can observe that the bot possibility scorer identifies very few users who post non-rumors as bots. On the contrary, the bot possibility scorer identifies the majority of users who post rumors as bots. Moreover, in false rumors and unverified rumors, the

Metric	Botometer-V4	AdaBoost	RF	MLP-10d	MLP-15d
Acc.	0.722	0.917	0.930	0.935	**0.944**

Table 7: Result of social bot detection.

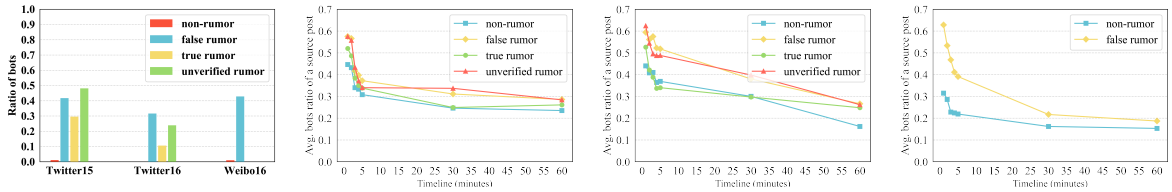

(a) Relationship between rumors and publishers. (b) Twitter15: Avg. bots ratio per source post. (c) Twitter16: Avg. bots ratio per source post. (d) Weibo16: Avg. bots ratio per source post.

Figure 4: Relationship between rumors and users.

ratio of bots is higher than that in true rumors.

We also make statistics on the average ratio of bot-behavior users among all participants under a source post for each source post class over time. As shown in Fig. 4b-4d, in the first five minutes after the source post is published, the bot behaviors are more active than that in the successive period. The bot-behavior users' ratio of false rumors and unverified rumors is higher than that of non-rumors and true rumors. The results of SBAG are consistent with sociological research, which also prove that our model has strong interpretability.

4.8 Case Study

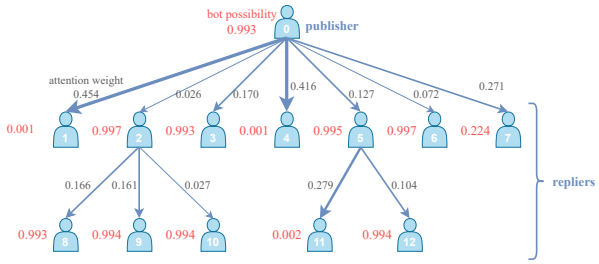

Figure 5: Case study.

To demonstrate the relation of bot possibility and attention weight, we choose a classic sample for visualization. Fig. 5 shows the attention weights of one publisher and the corresponding repliers, where the attention weights are computed by GUI. Fig. 5 illustrates that the edges connecting to users with lower bot possibility have higher attention weights. This way of aggregation helps learn more effective patterns of early propagation.

5 Related Work

Conventional rumor detection methods adopted machine learning to classify rumors based on the features of content, user, and propagation pattern, such as decision tree(Castillo et al., 2011), support vector machine(Yang et al., 2012; Liu et al., 2015; Ma et al., 2015; Wu et al., 2015), random forest(Kwon et al., 2013), etc. However, these methods involved feature engineering, which is hard to obtain high-order features.

Recent studies exploited deep learning methods for rumor detection. Most of the existing rumor detection methods mainly modeled a source post and its reposts together as a sequence or a graph, using RNN(Ma et al., 2016; Song et al., 2019; Zhou et al., 2019), CNN(Yu et al., 2017) , Transformer(Khoo et al., 2020; Rao et al., 2021), GCN(Bian et al., 2020; Song et al., 2021; Wei et al., 2021; Sun et al., 2022) and GAT(Lin et al., 2021) to learn the textual content features. However, these models do not consider the participant of social bots on social media to publish fraudulent content, which may lead to training noise by these fraudulent contents. Several studies(Liu and Wu, 2018; Yuan et al., 2019; Lu and Li, 2020; Yuan et al., 2020) integrated features of user feature or user credibility to learn the propagation patterns of source posts. However, they do not explicitly explore the unique user propagation pattern in the early stage, which limits the early-detection ability of the model.

6 Conclusion

In this work, according to the observation of sociological research, we propose a Social Bot-Aware Graph Neural Network for early rumor detection.

First, we pre-train a bot possibility scorer called SD on a large dataset containing bot users and genuine users, then SD is transferred to the BAG module. The BAG module takes the user's bot possibility information into the calculation of the features from different views, which enables the module to have a priori knowledge of the user in early detection. The experimental results on three public datasets show that SBAG effectively captures the early propagation of rumors, and further improves performance of early rumor detection.

Acknowledgements

This project was partially supported by National Natural Science Foundation of China (Grant number: 61976066), Beijing Natural Science Foundation (Grant number: 4212031), the Fundamental Research Fund for the Central Universities (Grant numbers:3262021T23), and Research Funds for NSD Construction, University of International Relations (Grant numbers: 2021GA07).

References

David M Beskow and Kathleen M Carley. 2018. Bot conversations are different: leveraging network metrics for bot detection in twitter. In *2018 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining (ASONAM)*, pages 825–832. IEEE.

Tian Bian, Xi Xiao, Tingyang Xu, Peilin Zhao, Wenbing Huang, Yu Rong, and Junzhou Huang. 2020. Rumor detection on social media with bi-directional graph convolutional networks. In *Proceedings of the AAAI Conference on Artificial Intelligence*, pages 549–556.

Carlos Castillo, Marcelo Mendoza, and Barbara Poblete. 2011. Information credibility on twitter. In *Proceedings of the 20th international conference on World wide web*, pages 675–684.

Tong Chen, Xue Li, Hongzhi Yin, and Jun Zhang. 2018. Call attention to rumors: Deep attention based recurrent neural networks for early rumor detection. In *Pacific-Asia conference on knowledge discovery and data mining*, pages 40–52. Springer.

Stefano Cresci, Roberto Di Pietro, Marinella Petrocchi, Angelo Spognardi, and Maurizio Tesconi. 2017. The paradigm-shift of social spambots: Evidence, theories, and tools for the arms race. In *Proceedings of the 26th international conference on world wide web companion*, pages 963–972.

Stefano Cresci, Fabrizio Lillo, Daniele Regoli, Serena Tardelli, and Maurizio Tesconi. 2018. $ fake: Evidence of spam and bot activity in stock microblogs on twitter. In *Twelfth international AAAI conference on web and social media*.

Shangbin Feng, Zhaoxuan Tan, Rui Li, and Minnan Luo. 2022. Heterogeneity-aware twitter bot detection with relational graph transformers. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 36, pages 3977–3985.

Shangbin Feng, Herun Wan, Ningnan Wang, Jundong Li, and Minnan Luo. 2021. Twibot-20: A comprehensive twitter bot detection benchmark. In *Proceedings of the 30th ACM International Conference on Information & Knowledge Management*, pages 4485–4494.

Zafar Gilani, Reza Farahbakhsh, Gareth Tyson, Liang Wang, and Jon Crowcroft. 2017. Of bots and humans (on twitter). In *Proceedings of the 2017 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining 2017*, pages 349–354.

Xiaoyun Han, Zhen Huang, Menglong Lu, Dongsheng Li, and Jinyan Qiu. 2021. Rumor verification on social media with stance-aware recursive tree. In *International Conference on Knowledge Science, Engineering and Management*, pages 149–161. Springer.

Ling Min Serena Khoo, Hai Leong Chieu, Zhong Qian, and Jing Jiang. 2020. Interpretable rumor detection in microblogs by attending to user interactions. In *Proceedings of the AAAI Conference on Artificial Intelligence*, pages 8783–8790.

Sneha Kudugunta and Emilio Ferrara. 2018. Deep neural networks for bot detection. *Information Sciences*, 467:312–322.

Sejeong Kwon, Meeyoung Cha, and Kyomin Jung. 2017. Rumor detection over varying time windows. *PloS one*, 12(1):e0168344.

Sejeong Kwon, Meeyoung Cha, Kyomin Jung, Wei Chen, and Yajun Wang. 2013. Prominent features of rumor propagation in online social media. In *2013 IEEE International Conference on Data Mining (ICDM)*, pages 1103–1108. IEEE.

Kyumin Lee, Brian Eoff, and James Caverlee. 2011. Seven months with the devils: A long-term study of content polluters on twitter. In *Proceedings of the international AAAI conference on web and social media*, volume 5, pages 185–192.

Hongzhan Lin, Jing Ma, Mingfei Cheng, Zhiwei Yang, Liangliang Chen, and Guang Chen. 2021. Rumor detection on twitter with claim-guided hierarchical graph attention networks. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 10035–10047.

Xiaomo Liu, Armineh Nourbakhsh, Quanzhi Li, Rui Fang, and Sameena Shah. 2015. Real-time rumor debunking on twitter. In *Proceedings of the 24th ACM international on conference on information and knowledge management*, pages 1867–1870.

Yang Liu and Yi-Fang Wu. 2018. Early detection of fake news on social media through propagation path classification with recurrent and convolutional networks. In *Proceedings of the AAAI conference on artificial intelligence*, volume 32.

Menglong Lu, Zhen Huang, Binyang Li, Yunxiang Zhao, Zheng Qin, and Dongsheng Li. 2022. Sifter: A unified framework for robust rumor detection. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*.

Yi-Ju Lu and Cheng-Te Li. 2020. Gcan: Graph-aware co-attention networks for explainable fake news detection on social media. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 505–514.

Jing Ma, Wei Gao, Prasenjit Mitra, Sejeong Kwon, Bernard J Jansen, Kam-Fai Wong, and Meeyoung Cha. 2016. Detecting rumors from microblogs with recurrent neural networks.

Jing Ma, Wei Gao, Zhongyu Wei, Yueming Lu, and Kam-Fai Wong. 2015. Detect rumors using time series of social context information on microblogging websites. In *Proceedings of the 24th ACM international on conference on information and knowledge management*, pages 1751–1754.

Jing Ma, Wei Gao, and Kam-Fai Wong. 2017. Detect rumors in microblog posts using propagation structure via kernel learning. Association for Computational Linguistics.

Jing Ma, Wei Gao, and Kam-Fai Wong. 2018. Rumor detection on twitter with tree-structured recursive neural networks. Association for Computational Linguistics.

Michele Mazza, Stefano Cresci, Marco Avvenuti, Walter Quattrociocchi, and Maurizio Tesconi. 2019. Rtbust: Exploiting temporal patterns for botnet detection on twitter. In *Proceedings of the 10th ACM conference on web science*, pages 183–192.

Vahed Qazvinian, Emily Rosengren, Dragomir Radev, and Qiaozhu Mei. 2011. Rumor has it: Identifying misinformation in microblogs. In *Proceedings of the 2011 Conference on Empirical Methods in Natural Language Processing*, pages 1589–1599.

Dongning Rao, Xin Miao, Zhihua Jiang, and Ran Li. 2021. Stanker: Stacking network based on level-grained attention-masked bert for rumor detection on social media. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 3347–3363.

Mohsen Sayyadiharikandeh, Onur Varol, Kai-Cheng Yang, Alessandro Flammini, and Filippo Menczer. 2020. Detection of novel social bots by ensembles of specialized classifiers. In *Proceedings of the 29th ACM international conference on information & knowledge management*, pages 2725–2732.

Chengcheng Shao, Giovanni Luca Ciampaglia, Onur Varol, Kai-Cheng Yang, Alessandro Flammini, and Filippo Menczer. 2018a. The spread of low-credibility content by social bots. *Nature communications*, 9(1):1–9.

Chengcheng Shao, Pik-Mai Hui, Lei Wang, Xinwen Jiang, Alessandro Flammini, Filippo Menczer, and Giovanni Luca Ciampaglia. 2018b. Anatomy of an online misinformation network. *PloS one*, 13(4):e0196087.

Changhe Song, Cheng Yang, Huimin Chen, Cunchao Tu, Zhiyuan Liu, and Maosong Sun. 2019. Ced: Credible early detection of social media rumors. *IEEE Transactions on Knowledge and Data Engineering*, 33(8):3035–3047.

Yun-Zhu Song, Yi-Syuan Chen, Yi-Ting Chang, Shao-Yu Weng, and Hong-Han Shuai. 2021. Adversary-aware rumor detection. In *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*, pages 1371–1382.

Mengzhu Sun, Xi Zhang, Jiaqi Zheng, and Guixiang Ma. 2022. Ddgcn: Dual dynamic graph convolutional networks for rumor detection on social media.

Lingwei Wei, Dou Hu, Wei Zhou, Zhaojuan Yue, and Songlin Hu. 2021. Towards propagation uncertainty: Edge-enhanced bayesian graph convolutional networks for rumor detection. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing*, pages 3845–3854.

Ke Wu, Song Yang, and Kenny Q Zhu. 2015. False rumors detection on sina weibo by propagation structures. In *2015 IEEE 31st international conference on data engineering*, pages 651–662. IEEE.

Rui Xia, Kaizhou Xuan, and Jianfei Yu. 2020. A state-independent and time-evolving network for early rumor detection in social media. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing*, pages 9042–9051.

Fan Yang, Yang Liu, Xiaohui Yu, and Min Yang. 2012. Automatic detection of rumor on sina weibo. In *Proceedings of the ACM SIGKDD workshop on mining data semantics*, pages 1–7.

Kai-Cheng Yang, Onur Varol, Clayton A Davis, Emilio Ferrara, Alessandro Flammini, and Filippo Menczer. 2019. Arming the public with artificial intelligence to counter social bots. *Human Behavior and Emerging Technologies*, 1(1):48–61.

Kai-Cheng Yang, Onur Varol, Pik-Mai Hui, and Filippo Menczer. 2020. Scalable and generalizable social bot detection through data selection. In *Proceedings of the AAAI conference on artificial intelligence*, volume 34, pages 1096–1103.

Feng Yu, Qiang Liu, Shu Wu, Liang Wang, Tieniu Tan, et al. 2017. A convolutional approach for misinformation identification. In *Proceedings of the Twenty-Sixth International Joint Conference on Artificial Intelligence (IJCAI)*, pages 3901–3907.

Chunyuan Yuan, Qianwen Ma, Wei Zhou, Jizhong Han, and Songlin Hu. 2019. Jointly embedding the local and global relations of heterogeneous graph for rumor detection. In *2019 IEEE international conference on data mining (ICDM)*, pages 796–805. IEEE.

Chunyuan Yuan, Qianwen Ma, Wei Zhou, Jizhong Han, and Songlin Hu. 2020. Early detection of fake news by utilizing the credibility of news, publishers, and users based on weakly supervised learning. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 5444–5454.

Zhe Zhao, Paul Resnick, and Qiaozhu Mei. 2015. Enquiring minds: Early detection of rumors in social media from enquiry posts. In *Proceedings of the 24th international conference on world wide web*, pages 1395–1405.

Kaimin Zhou, Chang Shu, Binyang Li, and Jey Han Lau. 2019. Early rumour detection. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 1614–1623.

Arkaitz Zubiaga, Ahmet Aker, Kalina Bontcheva, Maria Liakata, and Rob Procter. 2018. Detection and resolution of rumours in social media: A survey. *ACM Computing Surveys (CSUR)*, 51(2):1–36.

A Contrastive Cross-Channel Data Augmentation Framework for Aspect-Based Sentiment Analysis

Bing Wang[1*], **Liang Ding**[2†], **Qihuang Zhong**[4], **Ximing Li**[1], **Dacheng Tao**[2,3]
[1]College of Computer Science and Technology, Jilin University
[2]JD Explore Academy [3]The University of Sydney [4]Wuhan University
dingliang1@jd.com, liximing86@gmail.com

Abstract

Aspect-based sentiment analysis (ABSA) is a fine-grained sentiment analysis task, which focuses on detecting the sentiment polarity towards the aspect in a sentence. However, it is always sensitive to the *multi-aspect challenge*, where features of multiple aspects in a sentence will affect each other. To mitigate this issue, we design a novel training framework, called Contrastive Cross-Channel Data Augmentation (C^{3}DA), which leverages an in-domain generator to construct more multi-aspect samples and then boosts the robustness of ABSA models via contrastive learning on these generated data. In practice, given a generative pretrained language model and some limited ABSA labeled data, we first employ some parameter-efficient approaches to perform the in-domain fine-tuning. Then, the obtained in-domain generator is used to generate the synthetic sentences from two channels, i.e., Aspect Augmentation Channel and Polarity Augmentation Channel, which generate the sentence condition on a given aspect and polarity respectively. Specifically, our C^{3}DA performs the sentence generation in a cross-channel manner to obtain more sentences, and proposes an Entropy-Minimization Filter to filter low-quality generated samples. Extensive experiments show that our C^{3}DA can outperform those baselines without any augmentations by about 1% on accuracy and Macro-F1. Code and data are released in https://github.com/wangbing1416/C3DA.

1 Introduction

Sentiment Analysis (SA) is a typical Natural Language Understanding (NLU) task to predict the sentence-level sentiment polarities (Pang and Lee, 2007; Liu, 2012). However, since a single review sentence always exists multiple polarities, **A**spect-based **S**entiment **A**nalysis (**ABSA**), a fine-grained sentiment analysis task (Ma et al., 2017; Fan et al., 2018; Sun et al., 2019; Wang et al., 2020; Li et al., 2021b; Zhang et al., 2022a), is further introduced to detect the sentiment polarities towards given aspects (entities) in a review sentence. Gathering a review "while the ambiance and atmosphere were great, the food and service could have been a lot better.", the sentiment polarities of *"ambiance"* and *"atmosphere"* are *Positive* and we get *Negative* polarity for aspects *"food"* and *"service"*.

It is crucial to capture the aspect-specific contextual features for an ABSA model. Unfortunately, as shown in the aforementioned example, the multi-aspect challenge that there are multiple aspects in a sentence always deteriorates the model's generalization, especially when the expressed polarities of multiple aspect words in a sentence are opposite. Meanwhile, since ABSA is a low-resource task, it is hard to train a robust model under fewer samples.

Some existing works focused on the multi-aspect challenge, such as Lu et al. (2011); Hu et al. (2019) in which novel model structures were designed to capture aspect-specific sentiment information. Additionally, Jiang et al. (2019) manually collected a large-scale high-quality multi-aspect multi-sentiment (MAMS) dataset. Undoubtedly, the data-centric MAMS is an effective approach to tackle the multi-aspect problem, but the human-annotated and non-expandable limitations still hinder the robust training for ABSA models. Therefore, given limited labeled data, it is critical to investigate how to collect more in-domain multi-aspect samples automatically. On the other hand, capturing the aspect-specific information from multi-aspect sentences is also necessary for robust ABSA models.

In response to the aforementioned problems, we propose a novel training framework, namely **C**ontrastive **C**ross-**C**hannel **D**ata **A**ugmentation (**C^{3}DA**), to generate more in-domain multi-aspect samples and train robust ABSA models based on these generated data. Firstly, inspired by successful

[*]Work was done when Bing and Qihuang were interning at JD Explore Academy.
[†]Liang Ding is the corresponding author.

Proceedings of the 29th International Conference on Computational Linguistics, pages 6691–6704
October 12–17, 2022.

generative pretraining models (Raffel et al., 2020; Lewis et al., 2020), we employ an representative pretraining models T5 (Raffel et al., 2020) as the multi-aspect data generator. In practice, due to the limited ABSA labeled data, it is hard to fine-tuning the entire T5 model effectively. Thus, we further introduce some parameter-efficient methods, e.g. prompt (Lester et al., 2021a), prefix (Li and Liang, 2021) and LoRA (Hu et al., 2022), to tune the T5 generator. In this way, a domain -specific generator is obtained and we can apply it to collect more in-domain multi-aspect samples.

To be more specific, there are two channels to generate expected multi-aspect samples in our framework, i.e. **A**spect **A**ugmentation **C**hannel (**AAC**) and **P**olarity **A**ugmentation **C**hannel (**PAC**). In the AAC, the generator is encouraged to generate the synthetic sentence towards the given sentence and aspect, while in the PAC, the synthetic sentence towards the given sentence and polarity is obtained. To further boost the generated samples, we attempt to collect the multi-aspect samples in a cross-channel manner. Specifically, in the first generation, the source sentence is fed to the double channels and obtain an aspect-specific sentence and a polarity-inverted sentence respectively. In the second generation, both generated sentences are injected to the another channel to get the ultimate sentences. Finally, we propose an Entropy-Minimization Filter (EMF) to filter some low-quality generated samples. And a contrastive training objective that can draw away the different aspect's embeddings in a sentence is also leveraged to alleviate the multi-aspect problem and help train the final robust ABSA model.

We conduct sufficient experiments on three popular ABSA datasets, i.e. *Restaurant*, *Laptop* (Pontiki et al., 2014) and *Twitter* (Dong et al., 2014), to prove that our C^{3}DA framework can boost the model's robustness and predictive performance, and outperform other NLP data augmentation strategies. Specifically, with the help of our **C^{3}DA**, RoBERTa-based models can achieve averaged performance improvement 1.28% and 1.62% in terms of accuracy and macro-F1 respectively, while the improvements of BERT-based models are also over 0.87% and 1.10%. Some in-depth discussions and case studies are also executed. Contributions of this paper are threefold:

- We recast the vanilla ABSA training scheme with a data augmentation-based training framework and a contrastive training objective to tackle the multi-aspect challenge.
- We design a novel cross-channel data augmentation method based on generative large-scale pretrained language models to generate high-quality in-domain multi-aspect samples, which has great potential to benefit other fine-grained NLU tasks.
- Extensive experiments on three widely-used datasets show the effectiveness of our C^{3}DA.

2 Related Work

2.1 Language Models and Parameter Efficient

Large Pre-trained Language Models (PLMs) are still research cores in various natural language processing tasks. The PLMs tend to be variants of Transformer (Vaswani et al., 2017), and are trained on massive unlabeled raw sentences under some linguistic unsupervised objectives. According to the model structure, these Transformer-based PLMs are roughly divided into (1) Encoder-only LMs (Devlin et al., 2019; Liu et al., 2019) can be utilized to capture token-level and sentence-level features, (2) Decoder-only LMs (Radford et al., 2018, 2019) tend to design an auto-regressive objective to cater to text generation tasks and (3) Encoder-Decoder LMs (Lewis et al., 2020; Raffel et al., 2020) need two sentences (source and target) to perform conditional generation.

The above PLMs always follow the *pre-train + fine-tune (FT)* paradigm to adapt the downstream tasks, however, tuning a whole model is a tricky challenge and is not applicable to low-resource NLP tasks. Therefore, a novel *pre-train + prompt-tune (PT)* (Liu et al., 2021c) concept is proposed to break this challenge, PT-based models can also be divided into (1) manual discrete prompts design human-made templates to adapt different pre-trained objective (Liu et al., 2021a), and (2) soft prompts inject some learnable pseudo tokens or matrices to frozen PLMs (Liu et al., 2021d; Lester et al., 2021b; Li and Liang, 2021; Zhong et al., 2022b).

2.2 Aspect-based Sentiment Analysis

Aspect-based Sentiment Analysis (ABSA) is a kind of Text Classification task, that benefits from a better aspect-aware text representation. Therefore, an amount of neural network-based models were

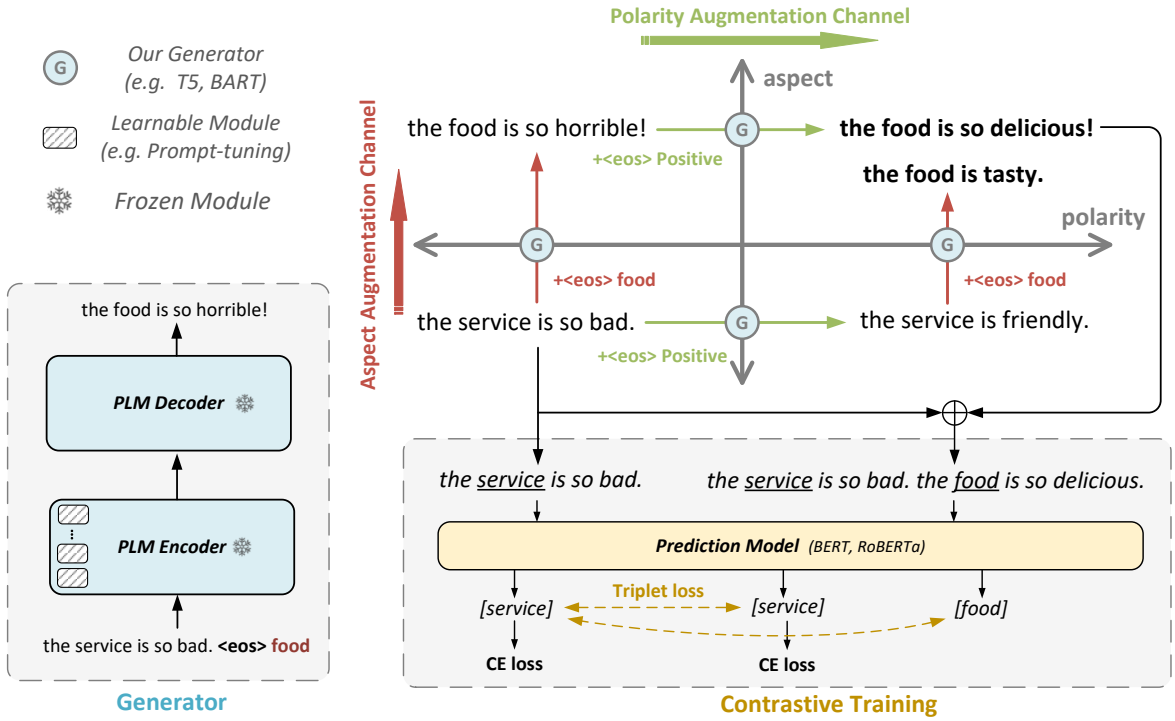

Figure 1: The overall C^{3}DA framework. In our generator, learnable modules can be optimized in the fine-tuning stage and will also be frozen in the generation stage, and the PLM encoder-decoder structure is always fixed.

proposed to break the ABSA challenge, for example, graph-based models (Sun et al., 2019; Wang et al., 2020; Li et al., 2021b; Zhong et al., 2022a) conduct graph convolutional operations on dependency trees to encode the semantic information, and attention-based models (Ma et al., 2017; Chen et al., 2017; Fan et al., 2018; Song et al., 2019; Liu et al., 2021b) focus on interaction between aspect terms and context tokens, *etc.*

In recent years, pre-trained language models changed the situation of ABSA because they are better text encoders and can capture context features. Therefore, researchers explored more views to improve ABSA performance, except that PLM is regarded as an encoder to replace the vanilla *Glove + BiLSTM* paradigm. For example, Dai et al. (2021) leverages RoBERTa to re-construct dependency trees, Yan et al. (2021); Li et al. (2021a); Zhang et al. (2021) try to integrate various ABSA subtasks into a unified generative framework, and Zhou et al. (2021); Seoh et al. (2021) combine ABSA with other NLP tasks, *etc.*

2.3 Data Augmentation

To improve the scale of training samples, some Data Augmentation (DA) methods in the NLP community are proposed (Zhang et al., 2018; Wei and Zou, 2019; Kobayashi, 2018; Wu et al., 2019; Anaby-Tavor et al., 2020; Ding et al., 2021; Cao et al., 2021; Wang et al., 2022; Zhang et al., 2022b). EDA (Wei and Zou, 2019) transforms source sentences with Synonym Replacement approaches, *etc.* CBERT (Wu et al., 2019) introduces a new conditional masked language model task. Meanwhile, some PLM-based generative augmentation frameworks (Anaby-Tavor et al., 2020; Wang et al., 2022; Zhang et al., 2022b) also have been explored. Also, recent work reveals the complementarity between PLMs and classical approaches, e.g., back translation (Liu et al., 2021e) and random initialization (Zan et al., 2022), to augment both the accuracy and generalization.

3 Our C^{3}DA Framework

Task Description of ABSA Given a group of triplets $\Omega^{src} = \{\mathbf{s}_i, \mathbf{a}_i, \mathbf{p}_i\}_{i=1}^{N}$, a sentence $\mathbf{s}_i = \{s_{i1}, \cdots, s_{iL}\}$ and an aspect-span indicator $\mathbf{a}_i \in \{0,1\}^L$ (1 denotes this token is an aspect, vice versa) will be fed to a trainable model to obtain a hidden embedding $\mathbf{h}_i = \mathcal{F}(\mathbf{s}_i, \mathbf{a}_i, \Theta) \in \mathbb{R}^{L\times D}$, where N, L and D denote the instance scale, sentence length and hidden embedding size, and $\mathcal{F}(\cdot, \Theta)$ is a prediction model (*e.g.,* BERT (Devlin et al., 2019)). Then, a ground-truth polarity label

$\mathbf{p}_i \in \{0, 1, 2\}$ is utilized to conduct supervised learning to estimate task-specific parameters Θ.

3.1 Overall Learning Objective

In this section, we introduce the overall process of our proposed method, and more details will be further described in Section 3.2 and 3.3, and our C^{3}DA framework is shown in Figure 1.

Our C^{3}DA framework consists of two steps: (1) Data Augmentation and (2) Contrastive Training. We utilize the representative pre-trained T5 model (Raffel et al., 2020) as our generator in the data augmentation stage[1]. Given a sentence $\mathbf{s}_i$, we expect to generate an another sentence $\widehat{\mathbf{s}}_i = \mathcal{G}(\mathbf{s}_i; \Pi)$ that expresses the opposite polarity and has a different aspect from the source sentence, where $\mathcal{G}(\cdot; \Pi)$ is the T5 generator.

Basically, we conduct the Supervised Classification Training (SCT) on source sentences and augmented sentences, the objective is as follows:

$$\mathcal{L}_{SCT} = \frac{1}{N}\sum_{i=1}^{N} \ell_{CE}(\mathbf{h}_i\mathbf{W}_s + \mathbf{b}_s, \mathbf{p}_i) + \alpha\ell_{CE}(\mathbf{h}_i^p\mathbf{W}_s + \mathbf{b}_s, \mathbf{p}_i), \quad (1)$$

$$\mathbf{h}_i^p = \mathcal{F}\left([\mathbf{s}_i, \widehat{\mathbf{s}}_i], \mathbf{a}_i, \Theta\right), \quad (2)$$

where $[\cdot, \cdot]$ and $\ell_{CE}(\cdot)$ are the concatenate operation and a Cross-Entropy loss function, $\mathbf{W}_s \in \mathbb{R}^{D\times M}$ and α denote the classification head and a hyper-parameter, respectively.

To obtain more robust performance, we leverage the Contrastive Training (CT) formulated as the triplet loss (Schultz and Joachims, 2003) as follows:

$$\mathcal{R}_{CT} = \frac{1}{N}\sum_{i=1}^{N} max\{d(\mathbf{h}_i, \mathbf{h}_i^p) - d(\mathbf{h}_i, \mathbf{h}_i^n) + \xi, 0\}, \quad (3)$$

where $\mathbf{h}_i^n$ is the average-pooled sentence representation of $\widehat{\mathbf{s}}_i$ in the sentence $[\mathbf{s}_i, \widehat{\mathbf{s}}_i]$, ξ is a hyper-parameter called margin that can control whether an instance should be trained, and $d(\cdot, \cdot)$ denotes a distance measure function. Specially, we apply a negative cosine similarity to measure the embedding distance $d(\cdot, \cdot)$.

$$d(\mathbf{h}_i, \mathbf{h}_i^p) = -\frac{\mathbf{h}_i\mathbf{h}_i^p}{\|\mathbf{h}_i\| \times \|\mathbf{h}_i^p\|}. \quad (4)$$

[1]Note that we can employ various generative pretraining models as the generator. T5 is used in this work, since it is widely-used and powerful.

Finally, the overall learning objective can be formulated as:

$$\mathcal{L} = \mathcal{L}_{SCT} + \beta\mathcal{R}_{CT}, \quad (5)$$

where β is a controllable hyper-parameter.

3.2 Cross-Channel Data Augmentation

Our motivation is to generate high-quality in-domain multi-aspect samples. In practice, the cross-channel data augmentation consists of two steps: 1) tune the pretrained backbone model to adapt our cross-channel sentence generation; 2) generate sentences with opposite polarity and different aspects.

Pre-Trained Generator We employ pre-trained encoder-decoder model T5 as our backbone. ABSA is a low-resource task, therefore, tuning a whole large language model (*e.g.,* T5) with scarce samples will cause an over-fitting problem. Inspired by Lester et al. (2021a); Wang et al. (2022), we utilize the parameter-efficient strategies to tune the generation model. In particular, we consider several efficient tuning methods, including prefix-tuning (Li and Liang, 2021), prompt-tuning (Lester et al., 2021a), and LoRA (Hu et al., 2022). We grid-search the optimal efficient tuning method for each dataset, detailed comparisons can be found at Appendix A.

Model Fine-Tuning To make the T5 model capable of generating new sentences conditioned on the original sentence and given aspect or polarity, we first construct the fine-tuning data based on the ABSA training set. Concretely, we randomly sample two instances $\{\mathbf{s}_i\}$ and $\{\mathbf{s}_j, \mathbf{a}_j, \mathbf{p}_j\}$ from the source triplets Ω^{src}. Then, we can tune the T5 model to generate $\mathbf{s}_j$ conditioned on $[\mathbf{s}_i, \mathbf{a}_j]$ or $[\mathbf{s}_i, \mathbf{p}_j]$. As aforementioned, parameter-efficient strategies are used to avoid over-fitting. Taking the cutting-edge prompt-tuning (Lester et al., 2021a) as an example, the massive parameters in the generator should be fixed in this stage, and we only tune the parameters of prompts.

Sentence Generation The sentence generation for data augmentation stage consists of **A**spect **A**ugmentation **C**hannel (**AAC**) and **P**olarity **A**ugmentation **C**hannel (**PAC**).

AAC generates an aspect-specific sentence $\mathbf{s}_i^a = \mathcal{G}([\mathbf{s}_i, \widehat{\mathbf{A}}], \Pi)$[2], where Π is the trainable parameters

[2]AAC can generate one or more sentences, we take generating a single sentence as an example.

Table 1: An example of the concatenation operation.

$\mathbf{s}$	*I've been here 3 times for lunch and it is one of my favorites in the city.*
$[\mathbf{s}, \textit{service}]$	*I've been here 3 times for lunch and it is one of my favorites in the city.* $< eos >$ ***service***
$[\mathbf{s}, \textit{Negative}]$	*I've been here 3 times for lunch and it is one of my favorites in the city.* $< eos >$ ***so bad***

of the T5 generator $\mathcal{G}(\cdot)$. PAC generates a polarity-inverted sentence $\mathbf{s}_i^p = \mathcal{G}([\mathbf{s}_i, \widehat{\mathbf{P}}], \Pi)$, where $\widehat{\mathbf{A}}$ is a random-sampled in-domain aspect and $\widehat{\mathbf{P}}$ is a group of seed spans that have the inverted polarity from $\mathbf{s}_i$, such as *so good*. Then, these sentences will be fed into the another channel to obtain $\mathbf{s}_i^{pa} = \mathcal{G}([\mathbf{s}_i^p, \widehat{\mathbf{A}}], \Pi)$ and $\mathbf{s}_i^{ap} = \mathcal{G}([\mathbf{s}_i^a, \widehat{\mathbf{P}}], \Pi)$ that can satisfy our motivation. We give an example of the concatenation operation $[\mathbf{s}, \cdot]$ in Table 1.

Note that for sentence generation, both the generator backbone and prompts will be frozen. The detailed Cross-Channel Data Augmentation method is shown in Algorithm 1.

Algorithm 1 Cross-Channel Data Augmentation.

Require: Source triplets $\Omega^{src} = \{\mathbf{s}_i, \mathbf{a}_i, \mathbf{p}_i\}_{i=1}^N$; Initial pre-trained T5 generator with soft prompt $\mathcal{G}(\cdot, \Pi^0)$; Pre-defined aspect set $\mathcal{A}$; Fine-tune iterations τ.

Ensure: Augmented sentences $\Omega^{aug} = \{\widehat{\mathbf{s}}_i\}_{i=1}^N$.

for $iter$ in τ **do** ▷ Stage I: Model Fine-tuning
  $\{\mathbf{s}_i\}, \{\mathbf{s}_j, \mathbf{a}_j, \mathbf{p}_j\} \leftarrow Sample(\Omega^{src})$
  $\nabla^a \leftarrow loss\left(\mathbf{s}_j, \mathcal{G}([\mathbf{s}_i, \mathbf{a}_j], \Pi^{iter})\right)$
  $\nabla^p \leftarrow loss\left(\mathbf{s}_j, \mathcal{G}([\mathbf{s}_i, \mathbf{p}_j], \Pi^{iter})\right)$
  $\Pi^{iter+1} \leftarrow Train(\Pi^{iter}, \nabla^a, \nabla^p)$
end for
$\Pi \leftarrow \Pi^{\tau}$
for $\{\mathbf{s}_i, \mathbf{a}_i, \mathbf{p}_i\}$ in Ω^{src} **do**
  ▷ Stage II: Sentence Generation
  $\widehat{\mathbf{A}} \leftarrow Sample(\mathcal{A})$
  $\widehat{\mathbf{P}} \leftarrow -\mathbf{p}_i$ ▷ $-\mathbf{p}$ denotes opposite polarity from $\mathbf{p}$
  $\mathbf{s}_i^a \leftarrow \mathcal{G}\left([\mathbf{s}_i, \widehat{\mathbf{A}}], \Pi\right), \mathbf{s}_i^p \leftarrow \mathcal{G}\left([\mathbf{s}_i, \widehat{\mathbf{P}}], \Pi\right)$
  $\mathbf{s}_i^{pa} \leftarrow \mathcal{G}\left([\mathbf{s}_i^p, \widehat{\mathbf{A}}], \Pi\right), \mathbf{s}_i^{ap} \leftarrow \mathcal{G}\left([\mathbf{s}_i^a, \widehat{\mathbf{P}}], \Pi\right)$
  $\Omega^{aug} \leftarrow \Omega^{aug} + Filter(\mathbf{s}_i^a, \mathbf{s}_i^p, \mathbf{s}_i^{pa}, \mathbf{s}_i^{ap})$
end for
return Ω^{aug};

3.3 Entropy-Minimization Filtering

Cross-Channel Data Augmentation generates four sentences for each source sentence, and we can leverage these sentences to train a prediction model $\mathcal{F}(\cdot, \Theta)$. However, it is necessary to filter out some low-quality samples that carry noisy (or difficult) information, which can be estimated by language modeling (Moore and Lewis, 2010) and norm of word embedding (Liu et al., 2020).

To ensure the certainty of synthetic sentences, we design an Entropy-Minimization Filter (EMF) to filter those noisy sentences. We establish a hypothesis that ***noisy sentences always express an ambiguous polarity*** such that our prediction model gives a more smooth polarity probability distribution. To be specific, the information entropy of prediction distribution as Equation 6 is an appropriate measure. A sentence with larger prediction entropy should be filtered, and a hyper-parameter k can control how many augmented sentences are selected to participate in training for each sample.

$$\mathcal{H}_i = -\mathbb{E}_{\mathbf{y}_i}\left[\log_2 p(\mathbf{y}_i)\right] = -p(\mathbf{y}_i)\log_2 p(\mathbf{y}_i), \tag{6}$$

$$p(\mathbf{y}_i) = softmax(\mathbf{h}_i \mathbf{W}_s + \mathbf{b}_s). \tag{7}$$

4 Experiment

4.1 Experimental Settings and Baselines

We conduct our experiments on three public datasets, i.e. *Restaurant*, *Laptop* from SemEval 2014 ABSA task (Pontiki et al., 2014) and *Twitter* (Dong et al., 2014). In practice, we run the experiments over 5 random seeds and report the average values of accuracy (acc) and Macro-F1 (F1) to avoid stochasticity.

Two widely-used pretraining models, i.e. BERT (Devlin et al., 2019) and RoBERTa (Liu et al., 2019), are regarded as our prediction backbone, and we compare our framework C^{3}DA with the following four data augmentation methods:

- **EDA** (Wei and Zou, 2019) is a token-level transformation method. It follows some patterns such as Synonym Replacement, Random Insertion, *etc.* For the ABSA task, we deliberately avoid corrupting the aspect words.
- **Back Translation (BT)** (Sennrich et al., 2016) is a sentence-level augmentation method, which translates a sentence to another language and translates it back to the original language. We follow Fan et al. (2021) that translates English to French and translates it back to English.

Table 2: Empirical results of **C^{3}DA** and other cutting-edge data augmentation methods. The best scores of each metric are indicated in bold, and ↑ denotes the improvement over BERT-*base* and RoBERTa-*base* baselines.

Model	*Restaurant*		*Laptop*		*Twitter*	
	acc	F1	acc	F1	acc	F1
BERT-*base*	86.31	80.22	79.66	76.11	76.50	75.23
+ EDA (Wei and Zou, 2019)	86.42	79.63	79.59	75.79	76.26	75.16
+ BT (Fan et al., 2021)	86.47	80.29	79.67	76.35	76.63	75.47
+ CBERT (Wu et al., 2019)	86.27	80.00	79.83	76.12	76.44	75.36
+ SCon (Liang et al., 2021)	86.51	80.55	80.23	76.48	-	-
+ **C^{3}DA** (Ours)	**86.93**$_{\uparrow 0.62}$	**81.23**$_{\uparrow 1.01}$	**80.61**$_{\uparrow 0.95}$	**77.11**$_{\uparrow 1.00}$	**77.55**$_{\uparrow 1.05}$	**76.53**$_{\uparrow 1.30}$
RoBERTa-*base*	86.38	80.29	80.10	76.24	76.63	75.37
+ EDA (Wei and Zou, 2019)	86.43	80.21	80.38	76.59	76.47	75.36
+ BT (Fan et al., 2021)	86.50	80.59	80.22	76.73	76.59	75.47
+ CBERT (Wu et al., 2019)	86.77	80.51	80.54	76.57	76.73	75.37
+ **C^{3}DA** (Ours)	**87.11**$_{\uparrow 0.73}$	**81.63**$_{\uparrow 1.34}$	**81.83**$_{\uparrow 1.73}$	**78.46**$_{\uparrow 2.22}$	**78.31**$_{\uparrow 1.38}$	**76.67**$_{\uparrow 1.30}$

- **CBERT** (Wu et al., 2019) fully excavates the power of mask language model (MLM) objective to replace some tokens, so it is also a token-level replacement approach.

- **SCon** (Liang et al., 2021) design aspect-invariant/-dependent data augmentation for ABSA and deploy a supervised contrastive learning objective. We reproduce it according to their released code and default best settings.

4.2 More Empirical Details

We implement our experiment by following the released ABSA Pytorch[3] repository and our pre-trained models, such as *bert-base-uncased*, *roberta-base* and *t5-base*, are from *HuggingFace*[4]. Meanwhile, we utilize a flexible toolkit *OpenDelta*[5] to adapt various parameter-efficient methods to our T5 generator.

In the data augmentation stage, we leverage an *Adafactor* optimizer. A T5 generator is fine-tuned for 100 epochs, and batch size is fixed to 16. As training a prediction model BERT or RoBERTa, we adapt an *Adam* optimizer with a learning rate of 2×10^{-5}, and the dropout rate is set to 0.3. Additionally, we train the final prediction model for 15 epochs with the batch size as 16.

A single NVIDIA A100 is used to conduct our all experiments. The model fine-tuning and sentence generation stages will generally spend 2 - 3 hours, and it spends about 1 hour to train a prediction model.

4.3 Main Result

Table 2 shows our main experimental results. To be fair, for each data augmentation method, we generate 1 synthetic sample per sentence. Overall, our C^{3}DA framework consistently and significantly improve the preformance of both BERT-based and RoBERTa-based models, e.g. the result of *Restaurant* even outperforms the RoBERTa-*base* baseline 2.22% in term of F1. Moreover, we observe that the performance order of various data augmentation methods is C^{3}DA > CBERT > BT > EDA, which indicates: (1) EDA semantic-regardlessly removes or inserts tokens, which will inject more noise into final prediction models, thus leading to the poor performance. (2) Comparing BT and CBERT, BT always achieves higher F1, and CBERT will get more advantageous accuracy. Since BT is a sentence-level method, it focuses on how to transform a source sentence into a form-different and semantic-reserved sentence, but neglects the useful aspect information. On the contrary, CBERT will only change some tokens, which is a semantic-reversed and aspect-reserved approach. Table 3 summarizes the characteristics of some augmentation methods.

Turn to compare two metrics, another finding is that C^{3}DA has more significant performance in term of Macro-F1. Proverbially, since Macro-F1 is a metric to measure the generalization of a model, the higher improvements on Macro-F1 prove that our framework can effectively improve the model's robustness in this view. Concurrently, the effect

[3] https://github.com/songyouwei/ABSA-PyTorch

[4] https://huggingface.co/models

[5] https://github.com/thunlp/OpenDelta

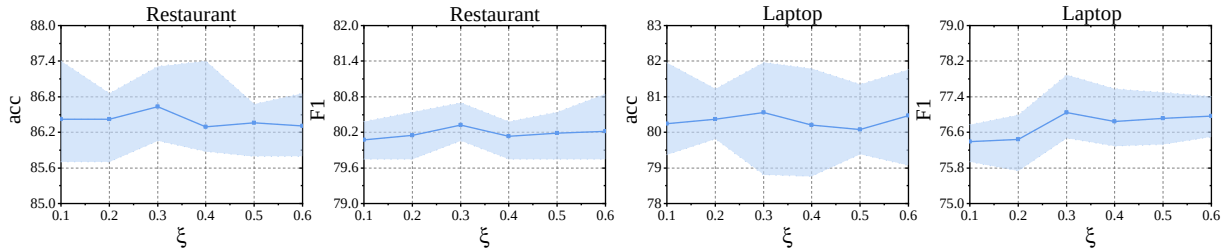

Figure 2: Sensitivity Analysis of the contrastive training margin ξ. The ribbon upper-side and below-side denote the maximum and minimum performance in 5 seeds respectively, and the centerline is the average value.

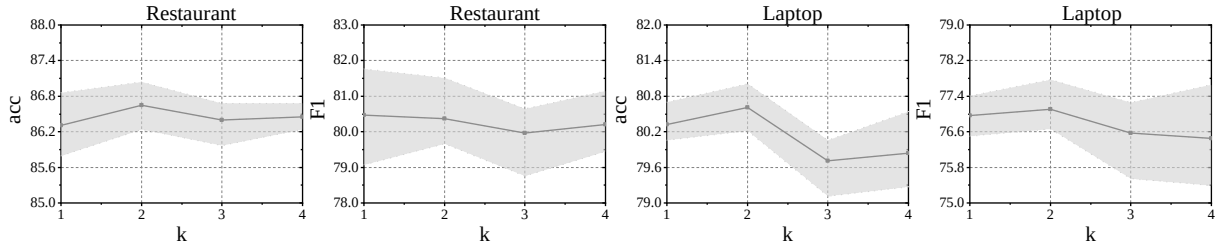

Figure 3: Sensitivity Analysis of the number of selected sentences k in EMF.

Table 3: A comparison of various data augmentation methods. Semantic-R. and Aspect-R. denote semantic reserved methods and aspect reserved methods, and T.- / S.- discriminates token-level and sentence-level methods respectively.

Methods	Semantic-R.	Aspect-R.	T.- / S.-
EDA	×	✓	T.
BT	✓	×	S.
CBERT	✓	✓	T.
C^{3}DA	✓	✓	S.

of C^{3}DA is better on *Restaurant* and *Twitter*. The difference between these datasets is that *Laptop* is a small dataset, indicating that our C^{3}DA is sensitive to the data scale and C^{3}DA may have better improvement if there is a larger-scale dataset.

4.4 Ablation Study

The core modules in our framework C^{3}DA are Data Augmentation (DA), Contrastive Learning (CL), and Entropy-Minimization Filter (EMF). In this section, we perform ablation study to investigate the effect of these modules, and the results are listed in Table 5. BERT-*base* is regarded as the prediction model in this study, and *w/o* denotes *without* some modules.

Firstly, **w/o DA & CL** removes our data augmentation framework, it is equivalent to a vanilla *BERT-base*-only prediction model. As shown in Table 5, w/o DA & CL always presents poor performance, even decrease by about 1% on *Laptop*'s acc and F1 and *Restaurant*'s F1. It proves our C^{3}DA framework imports more high-quality data items and is effective to boost the performance. Then, **w/o CL** remains augmented sentences and revokes the contrastive training objective $\mathcal{R}_{CL}$. Overall, this objective improves the results by about 0.3% on each dataset, conforming that contrastive learning can powerfully optimize the aspect-specific features. Finally, **w/o EMF** conducts both supervised training and contrastive training on all generated sentences. Intuitively, more samples can consolidate the predictive power of a classification model, but some emotionless and illogical sentences also carry plenty of noise. As we speculate, w/o EMF degrades the model's performance by 0.5% - 1.0%, the outcome on Restaurant's accuracy is even worse than the result without DA & CL. More sensitivity studies about EMF will be shown in Section 4.5.

Additionally, we also investigate the performance of generating sentences with **only AAC** and **only PAC** settings and report their results in Table 5. We show that both them are beneficial, where only AAC slightly outperforms that of the only PAC, demonstrating that the multi-aspect data are more crucial than the multi-polarity data.

4.5 Sensitivity Analysis

As mentioned above, we employ 4 hyper-parameters in our C^{3}DA framework, and ξ and k are more crucial. To investigate the influence of

Table 4: Case study. The aspect words and corresponding polarities are noted as red color and blue color, respectively.

case	*the falafal was rather over cooked and dried but the chicken was fine.*					
Aspect & Polarity	*falafal*			*chicken*		
	Positive	*Neutral*	*Negative*	*Positive*	*Neutral*	*Negative*
w/o C^{3}DA	0.098	0.109	0.793	0.643	0.047	0.310
with C^{3}DA	0.071	0.092	0.837	0.698	0.076	0.226
case	*good food but the service was dreadful!*					
Aspect & Polarity	*food*			*service*		
	Positive	*Neutral*	*Negative*	*Positive*	*Neutral*	*Negative*
w/o C^{3}DA	0.502	0.173	0.325	0.191	0.236	0.573
with C^{3}DA	0.589	0.211	0.200	0.177	0.186	0.637

Table 5: Ablation study of **C^{3}DA**. The prediction model is BERT-*base* in this study.

Model	*Restaurant*	*Laptop*	*Twitter*
	acc		
C^{3}DA	**86.93**	**80.61**	**77.08**
w/o DA & CL	86.31	79.66	76.50
w/o CL	86.69	80.35	76.87
w/o EMF	86.45	79.84	76.50
only AAC	86.63	80.10	77.16
only PAC	86.44	79.94	77.19
Model	F1		
C^{3}DA	**81.23**	**77.11**	**75.76**
w/o DA & CL	80.22	76.11	75.23
w/o CL	81.00	76.88	75.63
w/o EMF	80.21	76.45	75.17
only AAC	80.61	76.69	75.93
only PAC	80.24	76.17	75.91

both hyper-parameters, we further conduct sensitivity analysis in this section. Additionally, we fix α and β to 0.5 and 2.0, respectively. Limited by space, their sensitivity analysis will be described and analyzed in Appendix B.

Contrastive Training Margin ξ Margin ξ in Equation 3 is an important hyper-parameter to control the model's distinctive ability to discriminate negative samples and positive samples. Under a little margin, the prediction model will indolently balance $d(\mathbf{h}_i, \mathbf{h}_i^p)$ and $d(\mathbf{h}_i, \mathbf{h}_i^n)$ to cause under-optimization, while a large margin makes a model converge hard.

To investigate the impact of the margin ξ, we implement experiments on *Restaurant* and *Laptop*, and range ξ from 0.1 to 0.6. As shown in Figure 2, our C^{3}DA always gets the best performance when ξ is 0.3, and as it expands or shrinks, the results consistently show a downward trend.

Number of Selected Sentences k in EMF As the description in Section 4.4, k in EMF is a key hyper-parameter to decide to introduce more useful sentences or more noisy information, we select k sentences for each source sentence with the entropy-minimization filter method.

We evaluate C^{3}DA on $k \in \{1, 2, 3, 4\}$ and show the results in Figure 3. The best performance invariably appears in $k = 2$ or $k = 1$. Meanwhile, the performance decreases significantly when the value of k is 3, 4.

4.6 Case Study

C^{3}DA is a DA-based framework, and our purpose is to break the multi-aspect challenge. Therefore, we implement a case study to evaluate whether C^{3}DA can solve this problem, and more cases with different DA methods and cases with varying fine-tuning epochs will be shown in Appendix C.

We select two samples from the *Restaurant* test set, and these two sentences all have two aspects with different sentiment polarities. We will compare the performance of our C^{3}DA on their polarity prediction distribution. To observe the Table 4, we summarize that (1) C^{3}DA can consistently and significantly enhance the prediction model's robustness, such as the *Positive* probability of the aspect *food* improve about 0.077. Therefore, our framework can correct the polarity prediction when a sentence expresses an ambiguous polarity. (2) When a sentence is short, the interaction between two aspects with contradictory polarities will be more intimate. In this scenario, sentences will express more ambiguous sentiment and the effectiveness of our framework will be more significant.

4.7 Computing Cost Analysis

Our method will prolong each sentence, longer sentences will undoubtedly increase computing costs. Therefore, we attempt to compare the computing time of our framework and other augmentation methods. In the data preparing stage, C^{3}DA will fine-tune a generator and inference on the training set, it will spend 2 - 3 hours, and it is a once-for-all process. In the prediction training stage, we find that EDA, Back Translation and CBERT do not change the sentence length, their prediction time cost $0.98\times \sim 1.05\times$ (without any augmentation is $1\times$). By comparison, C^{3}DA spends $1.14\times$ time cost, which is completely acceptable.

5 Conclusion

In our work, we focus on solving the multi-aspect problem in the ABSA task. To address this challenge, we propose a data-centric training framework and design a novel C^{3}DA method to implement data augmentation. The proposed framework conduct both supervised training and contrastive training on augmented samples. The key idea of C^{3}DA is to generate some sentences that express the opposite sentiment polarity from the source sentences. Extensive experiments on widely-used benchmarks demonstrate that our framework can effectively improve the prediction model's robustness and predictive performance. In the future, it will be interesting to apply our C^{3}DA framework to improve other fine-grained natural language understanding tasks.

Acknowledgements

We are grateful to the anonymous reviewers and the area chair for their insightful comments and suggestions. Bing Wang and Ximing Li were supported by the project from the National Natural Science Foundation of China (NSFC) (No.62006094).

References

Ateret Anaby-Tavor, Boaz Carmeli, Esther Goldbraich, Amir Kantor, George Kour, Segev Shlomov, Naama Tepper, and Naama Zwerdling. 2020. Do not have enough data? deep learning to the rescue! In *The Thirty-Fourth AAAI Conference on Artificial Intelligence*, pages 7383–7390.

Yu Cao, Liang Ding, Zhiliang Tian, and Meng Fang. 2021. Towards efficiently diversifying dialogue generation via embedding augmentation. *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 7443–7447.

Peng Chen, Zhongqian Sun, Lidong Bing, and Wei Yang. 2017. Recurrent attention network on memory for aspect sentiment analysis. In *Conference on Empirical Methods in Natural Language Processing*, pages 452–461.

Junqi Dai, Hang Yan, Tianxiang Sun, Pengfei Liu, and Xipeng Qiu. 2021. Does syntax matter? A strong baseline for aspect-based sentiment analysis with roberta. In *Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 1816–1829.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: pre-training of deep bidirectional transformers for language understanding. In *Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 4171–4186.

Liang Ding, Di Wu, and Dacheng Tao. 2021. Improving neural machine translation by bidirectional training. In *Conference on Empirical Methods in Natural Language Processing*, pages 3278–3284, Online and Punta Cana, Dominican Republic.

Li Dong, Furu Wei, Chuanqi Tan, Duyu Tang, Ming Zhou, and Ke Xu. 2014. Adaptive recursive neural network for target-dependent twitter sentiment classification. In *Annual Meeting of the Association for Computational Linguistics*, pages 49–54.

Angela Fan, Shruti Bhosale, Holger Schwenk, Zhiyi Ma, Ahmed El-Kishky, Siddharth Goyal, Mandeep Baines, Onur Celebi, Guillaume Wenzek, Vishrav Chaudhary, Naman Goyal, Tom Birch, Vitaliy Liptchinsky, Sergey Edunov, Michael Auli, and Armand Joulin. 2021. Beyond english-centric multilingual machine translation. *Journal of Machine Learning Research*, 22:107:1–107:48.

Feifan Fan, Yansong Feng, and Dongyan Zhao. 2018. Multi-grained attention network for aspect-level sentiment classification. In *Conference on Empirical Methods in Natural Language Processing*, pages 3433–3442.

Edward J. Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, and Weizhu Chen. 2022. Lora: Low-rank adaptation of large language models. In *The International Conference on Learning Representations*.

Mengting Hu, Shiwan Zhao, Li Zhang, Keke Cai, Zhong Su, Renhong Cheng, and Xiaowei Shen. 2019. CAN: constrained attention networks for multi-aspect sentiment analysis. In *Conference on Empirical Methods in Natural Language Processing and the International Joint Conference on Natural Language Processing*, pages 4600–4609.

Himanshu Jain, Yashoteja Prabhu, and Manik Varma. 2016. Extreme multi-label loss functions for recommendation, tagging, ranking & other missing label applications. In *ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 935–944.

Qingnan Jiang, Lei Chen, Ruifeng Xu, Xiang Ao, and Min Yang. 2019. A challenge dataset and effective models for aspect-based sentiment analysis. In *Conference on Empirical Methods in Natural Language Processing and the International Joint Conference on Natural Language Processing*, pages 6279–6284. Association for Computational Linguistics.

Sosuke Kobayashi. 2018. Contextual augmentation: Data augmentation by words with paradigmatic relations. In *Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 452–457, New Orleans, Louisiana.

Brian Lester, Rami Al-Rfou, and Noah Constant. 2021a. The power of scale for parameter-efficient prompt tuning. In *Conference on Empirical Methods in Natural Language Processing*, pages 3045–3059.

Brian Lester, Rami Al-Rfou, and Noah Constant. 2021b. The power of scale for parameter-efficient prompt tuning. In *Conference on Empirical Methods in Natural Language Processing*, pages 3045–3059.

Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Veselin Stoyanov, and Luke Zettlemoyer. 2020. BART: denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. In *Annual Meeting of the Association for Computational Linguistics*, pages 7871–7880.

Chengxi Li, Feiyu Gao, Jiajun Bu, Lu Xu, Xiang Chen, Yu Gu, Zirui Shao, Qi Zheng, Ningyu Zhang, Yongpan Wang, and Zhi Yu. 2021a. Sentiprompt: Sentiment knowledge enhanced prompt-tuning for aspect-based sentiment analysis. *arXiv preprint*, abs/2109.08306.

Ruifan Li, Hao Chen, Fangxiang Feng, Zhanyu Ma, Xiaojie Wang, and Eduard H. Hovy. 2021b. Dual graph convolutional networks for aspect-based sentiment analysis. In *Annual Meeting of the Association for Computational Linguistics and the International Joint Conference on Natural Language Processing*, pages 6319–6329.

Xiang Lisa Li and Percy Liang. 2021. Prefix-tuning: Optimizing continuous prompts for generation. In *Annual Meeting of the Association for Computational Linguistics and the International Joint Conference on Natural Language Processing*, pages 4582–4597.

Bin Liang, Wangda Luo, Xiang Li, Lin Gui, Min Yang, Xiaoqi Yu, and Ruifeng Xu. 2021. Enhancing aspect-based sentiment analysis with supervised contrastive learning. In *ACM International Conference on Information and Knowledge Management,*, pages 3242–3247.

Tsung-Yi Lin, Priya Goyal, Ross Girshick, Kaiming He, and Piotr Dollár. 2017. Focal loss for dense object detection. In *IEEE International Conference on Computer Vision*, pages 2980–2988.

Bing Liu. 2012. *Sentiment Analysis and Opinion Mining*. Synthesis Lectures on Human Language Technologies. Morgan & Claypool Publishers.

Jian Liu, Zhiyang Teng, Leyang Cui, Hanmeng Liu, and Yue Zhang. 2021a. Solving aspect category sentiment analysis as a text generation task. In *Conference on Empirical Methods in Natural Language Processing*, pages 4406–4416.

Juhua Liu, Qihuang Zhong, Liang Ding, Hua Jin, Bo Du, and Dacheng Tao. 2021b. Unified instance and knowledge alignment pretraining for aspect-based sentiment analysis. *arXiv preprint*.

Pengfei Liu, Weizhe Yuan, Jinlan Fu, Zhengbao Jiang, Hiroaki Hayashi, and Graham Neubig. 2021c. Pre-train, prompt, and predict: A systematic survey of prompting methods in natural language processing. *arXiv preprint*, abs/2107.13586.

Xiao Liu, Yanan Zheng, Zhengxiao Du, Ming Ding, Yujie Qian, Zhilin Yang, and Jie Tang. 2021d. GPT understands, too. *arXiv preprint*, abs/2103.10385.

Xuebo Liu, Houtim Lai, Derek F. Wong, and Lidia S. Chao. 2020. Norm-based curriculum learning for neural machine translation. In *Annual Meeting of the Association for Computational Linguistics*, pages 427–436, Online.

Xuebo Liu, Longyue Wang, Derek F. Wong, Liang Ding, Lidia S. Chao, Shuming Shi, and Zhaopeng Tu. 2021e. On the complementarity between pre-training and back-translation for neural machine translation. In *Findings of the Association for Computational Linguistics: EMNLP*, pages 2900–2907.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Roberta: A robustly optimized BERT pretraining approach. *arXiv preprint*, abs/1907.11692.

Bin Lu, Myle Ott, Claire Cardie, and Benjamin K. Tsou. 2011. Multi-aspect sentiment analysis with topic models. In *IEEE International Conference on Data Mining workshops*, pages 81–88.

Dehong Ma, Sujian Li, Xiaodong Zhang, and Houfeng Wang. 2017. Interactive attention networks for aspect-level sentiment classification. In *International Joint Conference on Artificial Intelligence*, pages 4068–4074.

Robert C. Moore and William Lewis. 2010. Intelligent selection of language model training data. In *Proceedings of the ACL Conference Short Papers*, pages 220–224, Uppsala, Sweden.

Bo Pang and Lillian Lee. 2007. Opinion mining and sentiment analysis. *Found. Trends Inf. Retr.*, 2(1-2):1–135.

Maria Pontiki, Dimitris Galanis, John Pavlopoulos, Harris Papageorgiou, Ion Androutsopoulos, and Suresh Manandhar. 2014. Semeval-2014 task 4: Aspect based sentiment analysis. In *International Workshop on Semantic Evaluation*, pages 27–35.

Alec Radford, Karthik Narasimhan, Tim Salimans, and Ilya Sutskever. 2018. Improving language understanding by generative pre-training. *OpenAI blog*.

Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, Ilya Sutskever, et al. 2019. Language models are unsupervised multitask learners. *OpenAI blog*, 1(8):9.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J. Liu. 2020. Exploring the limits of transfer learning with a unified text-to-text transformer. *J. Mach. Learn. Res.*, 21:140:1–140:67.

Matthew Schultz and Thorsten Joachims. 2003. Learning a distance metric from relative comparisons. *Advances in neural information processing systems*, 16.

Rico Sennrich, Barry Haddow, and Alexandra Birch. 2016. Improving neural machine translation models with monolingual data. In *Annual Meeting of the Association for Computational Linguistics*, pages 86–96, Berlin, Germany.

Ronald Seoh, Ian Birle, Mrinal Tak, Haw-Shiuan Chang, Brian Pinette, and Alfred Hough. 2021. Open aspect target sentiment classification with natural language prompts. In *Conference on Empirical Methods in Natural Language Processing*, pages 6311–6322.

Youwei Song, Jiahai Wang, Tao Jiang, Zhiyue Liu, and Yanghui Rao. 2019. Attentional encoder network for targeted sentiment classification. *arXiv preprint*, abs/1902.09314.

Kai Sun, Richong Zhang, Samuel Mensah, Yongyi Mao, and Xudong Liu. 2019. Aspect-level sentiment analysis via convolution over dependency tree. In *Conference on Empirical Methods in Natural Language Processing and the International Joint Conference on Natural Language Processing*, pages 5678–5687.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Annual Conference on Neural Information Processing Systems*, pages 5998–6008.

Kai Wang, Weizhou Shen, Yunyi Yang, Xiaojun Quan, and Rui Wang. 2020. Relational graph attention network for aspect-based sentiment analysis. In *Annual Meeting of the Association for Computational Linguistics*, pages 3229–3238.

Yufei Wang, Can Xu, Qingfeng Sun, Huang Hu, Chongyang Tao, Xiubo Geng, and Daxin Jiang. 2022. Promda: Prompt-based data augmentation for low-resource NLU tasks. *arXiv preprint*, abs/2202.12499.

Jason W. Wei and Kai Zou. 2019. EDA: easy data augmentation techniques for boosting performance on text classification tasks. In *Conference on Empirical Methods in Natural Language Processing and the International Joint Conference on Natural Language Processing*, pages 6381–6387.

Xing Wu, Shangwen Lv, Liangjun Zang, Jizhong Han, and Songlin Hu. 2019. Conditional BERT contextual augmentation. In *International Conference on Computational Science*, volume 11539 of *Lecture Notes in Computer Science*, pages 84–95.

Hang Yan, Junqi Dai, Tuo Ji, Xipeng Qiu, and Zheng Zhang. 2021. A unified generative framework for aspect-based sentiment analysis. In *Annual Meeting of the Association for Computational Linguistics and the International Joint Conference on Natural Language Processing*, pages 2416–2429.

Changtong Zan, Liang Ding, Li Shen, Yu Cao, Weifeng Liu, and Dacheng Tao. 2022. On the complementarity between pre-training and random-initialization for resource-rich machine translation. In *International Conference on Computational Linguistics (COLING)*.

Hongyi Zhang, Moustapha Cisse, Yann N Dauphin, and David Lopez-Paz. 2018. mixup: Beyond empirical risk minimization. In *International Conference on Learning Representations*.

Wenxuan Zhang, Xin Li, Yang Deng, Lidong Bing, and Wai Lam. 2021. Towards generative aspect-based sentiment analysis. In *Annual Meeting of the Association for Computational Linguistics and the International Joint Conference on Natural Language Processing*, pages 504–510.

Wenxuan Zhang, Xin Li, Yang Deng, Lidong Bing, and Wai Lam. 2022a. A survey on aspect-based sentiment analysis: Tasks, methods, and challenges. *arXiv preprint*, abs/2203.01054.

Zheng Zhang, Liang Ding, Dazhao Cheng, Xuebo Liu, Min Zhang, and Dacheng Tao. 2022b. Bliss: Robust sequence-to-sequence learning via self-supervised input representation. *arXiv preprint*.

Qihuang Zhong, Liang Ding, Juhua Liu, Bo Du, Hua Jin, and Dacheng Tao. 2022a. Knowledge graph augmented network towards multiview representation learning for aspect-based sentiment analysis. *arXiv preprint*.

Qihuang Zhong, Liang Ding, Juhua Liu, Bo Du, and Dacheng Tao. 2022b. PANDA: prompt transfer meets knowledge distillation for efficient model adaptation. *CoRR*, abs/2208.10160.

Deyu Zhou, Jianan Wang, Linhai Zhang, and Yulan He. 2021. Implicit sentiment analysis with event-centered text representation. In *Conference on Empirical Methods in Natural Language Processing*, pages 6884–6893.

A Various Parameter-Efficient Methods Result

We adapt three parameter efficient methods for our T5 generator:

- **Prefix-tuning** (Li and Liang, 2021) fixes PLMs parameters, and inject an additional trainable prefix tokens for each transformer hidden layer. We set the number of prefix tokens to default 6.
- **Prompt-tuning** (Lester et al., 2021b) only prepends a prompt family to the input embedding layer. The length of the soft tokens is 100 in our experiments.
- **Low-Rank Adaptation (LoRA)** (Hu et al., 2022) trains a low-rank matrix to lightweight the PLMs. To be specific, we fix the hyper-parameter rank r to 8 and cancel the dropout operation ($dropout = 0$).
- **Full-tuning** fine-tunes all parameters in the T5 backbone.
- **No-tuning** directly generates sentences with a pre-trained T5 model without fine-tuning.

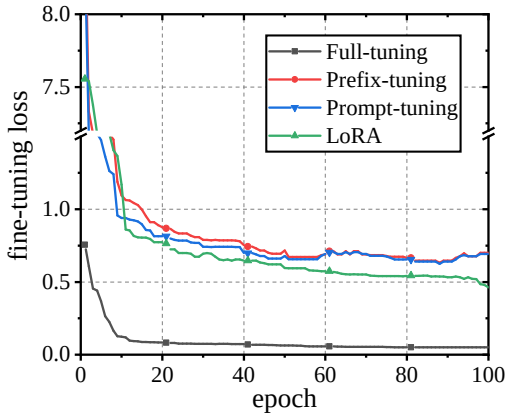

Figure 4: Convergence Analysis.

A.1 Case Study

More instances from different fine-tuning methods are listed in Table 6, the domains and polarities are attached to the source sentences. Under our observation, we found some phenomena: (1) the sentences from Full-tuning are always longer, and Full-tuning habitually outputs the same sentences, which proves that Full-tuning will cause an over-fit problem by conjecture. (2) three parameter efficient methods consistently generate multi-aspect samples, however, it is doubtful whether more aspects in a sentence will promote the model's robustness more significantly. (3) most of the generated sentences can achieve our motivation (express an opposite polarity).

A.2 Convergence Analysis

We analyze the fine-tuning loss for those aforementioned fine-tuning methods (without No-tuning) with `T5ForConditionGeneration.loss` interface in *Hugging Face* on *Restaurant*. Figure 4 shows the loss curve, we adapt a percentile filter with 20 pts to conduct the curve smooth, and the losses of ultimate convergence are listed in Table 7. Obviously, Full-tuning will achieve a lower loss, because all parameters are tuned, and the scale of parameters is far greater than the number of data items, it is a precursor of an over-fitting problem. By comparing three fine-tuning methods, we prefer a method with lower convergence loss, Prefix-tuning and Prompt-tuning are indistinguishable, however, LoRA obtains an obvious superiority.

B More Sensitivity Analysis

Objective coefficients α and β are also controllable hyper-parameters, so we conduct sensitivity analysis on them. Our experimental results are shown in Figure 5 and Figure 6, the prediction model is consistently BERT-*base* and $\alpha \in \{0.1, 0.3, 0.5, 1, 2\}, \beta \in \{0.5, 1, 2, 5, 10\}$. According to the figures, the best performance always when $\alpha = 0.3$ or 0.5 and $\beta = 1$ or 2.

C More Case Studies

Different DA methods We investigate different data augmentation frameworks from the view of the model's prediction results in Section 4.3. We will also list an example to compare these DA baselines and throw more generated sentences from our C^{3}DA in Table 8. The example in the upper table is from *Restaurant*, and we adapt LoRA to the T5

Table 6: Case study. The upper table compares results of different data augmentation methods, the bold spans is changed part; The below table shows some generated sentences with C^{3}DA.

Method	Examples [dataset, polarity]
source	*not only was the food outstanding, but the little 'perks' were great.* [Restaurant, Positive]
No-tuning	*the kitchen is very well equipped with all the equipment you need to cook up a storm.*
Full-Tuning	*The food is okay and the prices here are mediocre.*
Prefix-tuning	*The kitchen is a little small, but the food is good.*
Prompt-tuning	*The kitchen is a little dated, but the food is good.*
LoRA	*The seats are a bit cramped, but the food was delicious.*
source	*it is easy to start up and does not overheat as much as other laptops.* [Laptop, Positive]
No-tuning	*a laptop that is easy to start up and does not overheat as much as other laptops.*
Full-Tuning	*I also purchased iWork to go with it which has programs for word processing, spreadsheets, and presentations (similar to Microsoft Office).*
Prefix-tuning	*The battery isn't very strong, but it is very light.*
Prompt-tuning	*but the screen is so dark, and the service center is a little numb.*
LoRA	*The software is very easy to use, the start up is very fast, the graphics are fantastic.*

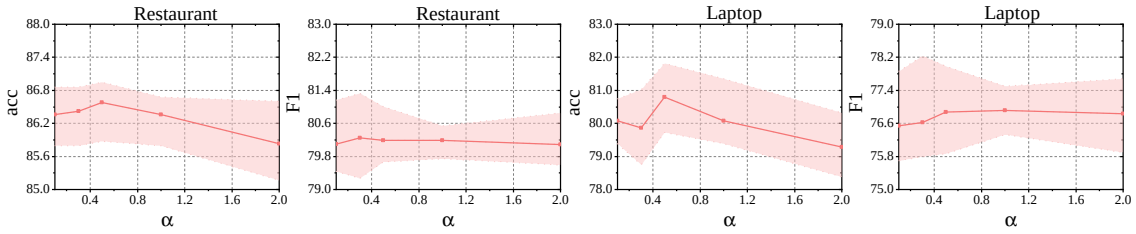

Figure 5: Sensitivity Analysis of the objective coefficient α in supervised classification training.

Table 7: Convergence loss. "-T." is short for "-Tuning".

Method	Full-T.	Prefix-T.	Prompt-T.	LoRA
Loss	0.05102	0.70009	0.69182	0.46218

generator, and more examples of C^{3}DA below are from *Restaurant* and *Laptop*.

We summarize that **EDA** deletes *"my"*, transforms *"co-workers"* to its synonym *"workers"* and randomly inserts a comma to an unreasonable position. This method adds some corruptions without any semantic relations, and it is possible to generate an illogical sample; Compared to EDA, **CBERT** is also a token-level method, but a semantic-reserved sentence is consistently created; **Back Translation** is a sentence-level method, and a correct sentence can be generated. However, it erases the aspect words for ABSA task. By comparison, our **C^{3}DA** not only reserves the origin semantic information but also introduces some high-quality in-domain corpus-aware information and adversarial signals. We also observe that most generated sentences from C^{3}DA exist two or more aspects, and this phenomenon is widespread in several datasets.

Cases with Varying Fine-tuning Epochs τ We also explore the generation results with varying fine-tuning epochs τ. As shown in Table 9, given a source sentence "*but the staff was so horrible to us.*", the T5 generator tends to generate an irrelative sentence when we have not yet fine-tuned it. In the middle of the fine-tuning stage, the generator is inclined to retell the source sentences or generate some irrational sentences. Finally, our generation model will eventually converge when $\tau = 80$.

D Instance-level Loss Re-weight for Long-tail Aspects

There is an another problem when we implement sentence generation, our generated sentences in one dataset often fit to few aspects, such as *food* in *Restaurant*. To investigate this issue, we show the aspect information for three public ABSA datasets in Table 10, and we found that *food* in *Restaurant* actually maintain maximum frequency. Similarly, *service*, *battery* and *screen etc.* report this long-tail distribution problem. And most of aspects in *Twitter* is *[UNK]*, it means that extensive aspect information will be unusable, this causes some models don't work, especially the attention-based models.

Similar to focal loss (Lin et al., 2017), we adapt

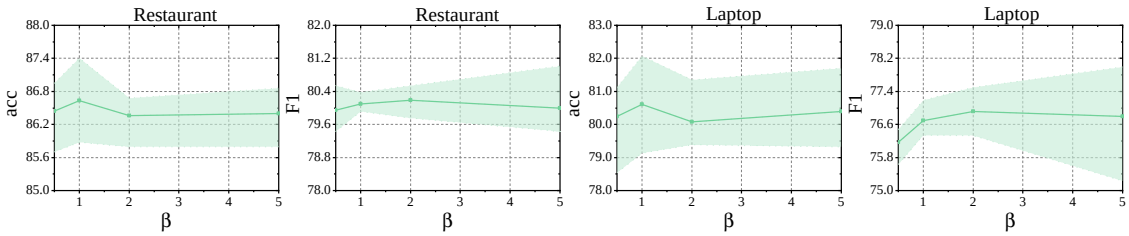

Figure 6: Sensitivity Analysis of the contrastive training objective coefficient β in our overall objective.

Table 8: Case study. The upper table compares results of different data augmentation methods, the blue spans is changed part; The below table shows some generated sentences with C^{3}DA.

DA methods	examples
source	*my co-workers had to wait almost an hour for delivery.*
EDA	*workers had to, wait an hour for delivery.*
B.T.	*my colleague waited nearly an hour to deliver it.*
CBERT	*my mother had to wait half an hour for delivery.*
C^{3}DA	*my co-workers had to wait almost an hour for delivery. but the perks are great, and you get to enjoy all the perks.*

source sentences	C^{3}DA generated sentences
if you love seafood, you would love this place!	*the sauce was good, but the meats were a little fatty.*
we had a girls, night dinner here for restaurant week.	*try the samosas at a reasonable price.*
it is easy to start up and does not overheat as much as other laptops.	*but the screen is so dark, and the service center is a little numb.*

Table 9: Case study. Given a source sentence "*but the staff was so horrible to us.*", we can observe the trend of its augmented sentence with fine-tune epochs τ.

epochs τ	sentence
source	***but the staff was so horrible to us.***
0	*i cried out for the suffragellah woman.*
20	*the staff is so horrible to us.*
40	*the as-is, the service isn't great.*
60	*the food is very good.*
80	*the food was great, the service was excellent.*

an instance-level loss re-weight method to our fine-tuning stage. Inspired by Jain et al. (2016), we multiple a multiplier to each instance's objective, the multiplier is shown in Equation (8).

$$\Delta_j = \frac{1}{1 + Ce^{-A\log(M_j+B)}}, \tag{8}$$

$$C = (\log M - 1)(B+1)^A, \tag{9}$$

where M_j, M are the number of data points annotated with aspect j, and aspect items, respectively.

Table 10: The aspect words and its frequency statistic of three public datasets. We only show the top-10 aspect words (with the T5 tokenizer) and hide some special tokens such as '*a*', '*i*', *etc.*

dataset	aspect words: frequency
Restaurant	*food*: 419, `[UNK]`: 404, *service*: 204, *wait*: 94, *menu*: 80, *dinner*: 75, *wine*: 72, *staff*: 68 , *pizza*: 65, *place*: 60
Laptop	`[UNK]`:267, *battery*: 97, *screen*: 81, *use*: 58, *life*: 56, *windows*: 55, *price*: 55, *software*: 54 , *keyboard*: 54, *drive*: 53
Twitter	`[UNK]`: 5328, *spear*: 894, *bri*: 887, *ney*: 887, *lo*: 394, *han*: 393, *lind*: 388, *say*: 388 , *in*: 370, *ry*: 360

Sentiment Interpretable Logic Tensor Network for Aspect-Term Sentiment Analysis

Bowen Zhang[1], Xu Huang[2], Zhichao Huang[3], Hu Huang[4], Baoquan Zhang[2], Xianghua Fu[1]* and Liwen Jing[1,5] *

[1]College of Big Data and Internet, Shenzhen Technology University, Shenzhen, China
[2] School of Computer Science and Technology, Harbin Institute of Technology, Shenzhen, China
[3] JD Intelligent Cities Research, Beijing, China
[4] School of Cyber Science and Technology, University of Science and Technology of China, Hefei, China
[5] Shenzhen innovation Institute (Xinstitute), Shenzhen, China

Abstract

Aspect-term sentiment analysis (ATSA) is an important task that aims to infer the sentiment towards the given aspect-terms. It is often required in the industry that ATSA should be performed with interpretability, computational efficiency and high accuracy. However, such an ATSA method has not yet been developed. This study aims to develop an ATSA method that fulfills all these requirements. To achieve the goal, we propose a novel Sentiment Interpretable Logic Tensor Network (SILTN). SILTN is interpretable because it is a neurosymbolic formalism and a computational model that supports learning and reasoning about data with a differentiable first-order logic language (FOL). To realize SILTN with high inferring accuracy, we propose a novel learning strategy called the two-stage syntax knowledge distillation (TSynKD). Using widely used datasets, we experimentally demonstrate that the proposed TSynKD is effective for improving the accuracy of SILTN, and the SILTN has both high interpretability and computational efficiency.

1 Introduction

1.1 Motivation

Aspect-term sentiment analysis (ATSA) is a fine-grained task in sentiment analysis, which aims is to recognize the sentiment polarity of the given aspect-term in a sentence (Zafra et al., 2019). Early research for ATSA was developed based on manually extracted features. For example, Poria et al. (2014) designed hand-crafted dependency rules to obtain aspect-related words, which are then fed into the machine learning methods to infer the sentiment polarity. With the development of deep learning, deep neural networks (DNNs) have dominated the study. Generally, the performance of DNNs is superior to traditional machine learning methods when the labeled training data is sufficient. More recently, there are two classes of methods that have received attention: (i) syntax-aware neural network (SaNN), which integrates syntax knowledge into attention-based neural network that increases the predictor's performance and interpretability (Zhang et al., 2019; Wang et al., 2020; Li et al., 2020; Nguyen and Shirai, 2015); (ii) large pre-trained language model (PLM) for ATSA (e.g., Bert (Devlin et al., 2019; Song et al., 2019; Zeng et al., 2019; Dai et al., 2021)), which learns knowledge from large-scale corpus and stably exceeds the other baseline by a significant margin.

Despite the effectiveness of prior work, ATSA in real-world remains a challenge for several reasons. First, DNNs usually perform as a "black box", because they cannot explicitly explain the process of the analysis; therefore, they cannot be applied in cases where explanations are required. Second, the performance of SaNNs relies on the intricate knowledge integration mechanism, which introduces more trainable parameters and brings extra computational costs. Furthermore, based on our empirical observation, the SaNN achieves merely limited performance improvements on most attention-based models (see methods 9-12 in Table 1). Third, for the pre-trained model, the enormous parameters lead to high storage and computational costs, making them a burden to be deployed in resource-constrained application scenarios such as real-time inference on mobile or edge devices. Besides, similar to DNNs, the pre-trained methods also lack interpretability.

1.2 Purpose

In response, this study aims to develop an ATSA method that can achieve interpretability, computational efficiency and high accuracy simultaneously.

Interpretability. To satisfy the interpretability requirement, we aim to develop an understandable ATSA method that can extract aspect-related words

* Corresponding authors: xianghuafu@sztu.edu.cn, ljing@connect.ust.hk

Proceedings of the 29th International Conference on Computational Linguistics, pages 6705–6714
October 12–17, 2022.

with explicit semantic knowledge, and build the sentiment inferring neural network that can be explained by first-order logic language (FOL). The FOL type of explanation should be agreeable for humans, and valuable in a situation where explanations are required.

Computational efficiency. To achieve high computational efficiency, we aim to develop an efficient ATSA method that utilizes fewer parameters for prediction while still achieving comparable results as conventional state-of-the-art models.

High accuracy. To achieve high accuracy, we aim to develop an ATSA method that achieves outstanding performance while maintaining interpretability and computational efficiency at the same time. Although it is very challenging, inspired by knowledge distillation (Hinton et al., 2015), we aim to develop a knowledge distillation strategy that transfers the knowledge of large and high-performance networks into an interpretable and computational efficient network.

1.3 Approach

To achieve the goal, we propose a novel **S**entiment **I**nterpretable **L**ogic **T**ensor **N**etwork (SILTN) for ATSA. Further, to realize this SILTN with high inferring accuracy, we propose a two-stage syntax knowledge distillation (TSynKD) strategy.

SILTN. SILTN is a neurosymbolic formalism and a computational neural network that supports learning and reasoning about data with a differentiable first-order logic language. Logic rules provide a flexible declarative language for communicating high-level cognition and expressing structured knowledge.

TSynKD. Although SILTN is well interpretable, its predictive performance is unsatisfactory because of the shallow network structure. Therefore, we propose the TSynKD to improve the inferring accuracy of SILTN, which is motivated by the observation that knowledge distillation can compress the large and high-performance networks (teacher) into a small student model while preserving the knowledge of the teacher model. TSynKD consists of two distillation stages with three networks: a large network (first teacher), a big network (second teacher) and a small network (student). The first distillation stage is the *output distillation* stage, which makes a large network output logits as a big network training objective. In this paper, we use the pre-trained Bert as the large network. As for the big network, we propose an aspect-specific dynamic graph convolutional network (AsDGCN) to model the dependency knowledge. The second stage is *feature distillation* stage, which allows a student to learn from a teacher's intermediate features. Here, SILTN is the student for distilling dependency knowledge from teacher AsDGCN.

1.4 Contribution

We summarize our contributions as follows: (1) To the best of our knowledge, this is the first work to integrate an interpretable logic tensor network in a principled framework for ATSA. SILTN is constructed followed by FOL, which provides a flexible declarative language for communicating high-level cognition and expressing structured knowledge. (2) We propose a two-stage syntax knowledge distill strategy (TSynKD) in ATSA, which significantly improves the performance of SILTN. (3) Extensive experiments have been conducted to evaluate the effectiveness of our model for ATSA.

2 Our Methodology

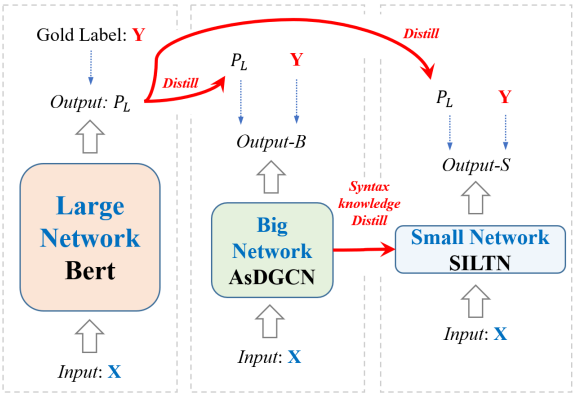

Figure 1: Overall framework.

2.1 Problem Definition

The ATSA task can be formulated as follows. Given a sentence $x = \{w_1^c, ..., w_a^a, ..., w_{a+m}^a, ..., w_n^c\}$ contains the corresponding aspect-term words $\{w_a^a, ..., w_{a+m}^a\}$, where w denotes each word in the sentence and m denotes the aspect-term length. Each sentence has a sentiment label y. ATSA aims to predict a sentiment label for the input sentence x towards the given aspect-term. In this paper, we use superscripts "c", "a" to indicate a context word and aspect-term word, respectively.

2.2 Framework Overview

Figure 1 shows the overall framework of our proposed method. The blue block denotes the SILTN, which serves as a basic prediction model. SILTN is developed following FOL, where each *Grounding* $\mathcal{G}$ is constructed by a simple trainable neural network structure. To realize SILTN with high inferring accuracy, SILTN is trained by TSynKD strategy. TSynKD consists of two distillation stages with three networks: a large network (Bert), a big network (AsDGCN), and a small network (SILTN). The first stage is the *output distillation*, which utilizes Bert's output logits as the AsDGCN's training objective. The second stage is the feature distillation, where SILTN learns dependency knowledge from AsDGCN and utilizes Bert's output logits as the SILTN's training objective for further improving the inferring accuracy.

2.3 SILTN

2.3.1 Preliminary: Logic Tensor Network

Logic Tensor Network (LTN) is a neuro-symbolic formalism and computational model that supports learning and reasoning about data with rich knowledge. The semantics of logic in LTN (called *Real Logic*) depart from the standard abstract semantics of FOL. In *Real Logic*, every object denoted by constants, variables and terms is interpreted as a tensor of real values. *Predicates* are interpreted as functions or tensor operations projecting onto a value in the interval [0, 1]. Here, functions are usually implemented by neural networks. *Grounding* in *Real Logic*, denoted by $\mathcal{G}$, associates a tensor of real numbers, where a real number in the interval [0, 1].

2.3.2 SILTN Notation and Definition

The notation and definition of SILTN are as follows:

Domains: *texts*, denoting the examples from dataset. *labels*, denoting the class labels.

Variables: x_+, $x_\circ$ and x_- denoting the text of "positive", "neutral" and "negative", respectively. x for all examples. D(x_+) = D($x_\circ$) = D(x_-) = D(x) = texts.

Constants: l_+, $l_\circ$ and l_-, the labels of classes for "positive", "neutral" and "negative", respectively. D(l_+) = D($l_\circ$) = D(l_-) = labels.

Predicates: $A(x)$ denoting the dependency relation. $K(k, q)$ denoting the knowledge rule $k \rightarrow q$. $R(x, a)$ denoting the fact that the aspect-related words toward the aspect-term a. $P(R(x, a), l)$ denoting the fact that text x is classified as l when targeting to aspect-term a.

Axioms:

$$\forall x\ K(A(x), R(x, a)) \tag{1}$$

$$\forall x_+\ P(R(x_+, a), l_+) \tag{2}$$

$$\forall x_\circ\ P(R(x_\circ, a), l_\circ) \tag{3}$$

$$\forall x_-\ P(R(x_-, a), l_-) \tag{4}$$

Notice that rules about exclusiveness such as $\forall x(P(x, l_+) \rightarrow (\neg P(x, l_\circ) \wedge \neg P(x, l_-)))$ are not included since such constraints are already imposed by the grounding of P below, more specifically the softmax function.

Grounding: $\mathcal{G}(l)$ is the one-hot vector where $\mathcal{G}(l_+) = [1, 0, 0]$, $\mathcal{G}(l_\circ) = [0, 1, 0]$ and $\mathcal{G}(l_-) = [0, 0, 1]$. $\mathcal{G}(x)$ is a word matrix of x. $\mathcal{G}(a)$ and $\mathcal{G}(c)$ are word matrix of aspect-term a and content words c, respectively. $\mathcal{G}(A(x))$ is a dependency relation vector sequence of the given text x. $\mathcal{G}(R(x, a))$ is a vector sequence that computed by $\mathcal{G}(x)$ and $\mathcal{G}(a)$. $\mathcal{G}(K(A(x), R(x, a)))$ is a vector sequence that computed by $\mathcal{G}(A(x))$ and $\mathcal{G}(R(x, a))$. $\mathcal{G}(P|\theta)$: (x, a), $l \mapsto$ softmax(SILTN$_\theta(x, a)$), where the LTN has three output neurons corresponding to the sentiment polarity "positive, neural or negative", and each neurons gives the probability corresponding to the class l.

2.3.3 Network structure of SILTN

Follow the SILTN definitions, we aim to achieve $\mathcal{G}(P|\theta)$: (x, a), $l \mapsto$ softmax(SILTN$_\theta$(x, a)), where each grounding is constructed by neural network structure. Figure 2 is the framework of SILTN.

Specifically, following FOL, $\mathcal{G}(P)$ can be decomposed by each groundings. According to Figure 2, the first layer *Grounding* is $\mathcal{G}(x)$, where the *Grounding* of input sentence x is constructed by the text representation model such as LSTM or Bert, etc. Formally, given input sequence x, we convert the i-th word into a low-dimensional vector representation e_i by embedding layer, where d denotes the dimension of embedded vectors. The $\mathcal{G}(x)$ represents the hidden states of LSTM, where the input is the combination of e_i.

The second layer *Grounding* is *Predicates* $A(x)$, which aims to model the dependency relation of each word toward the aspect-term. To achieve this goal, the distilled dependency knowledge is utilized in this layer. Formally, $\mathcal{G}(A(x))$ can be calculated as:

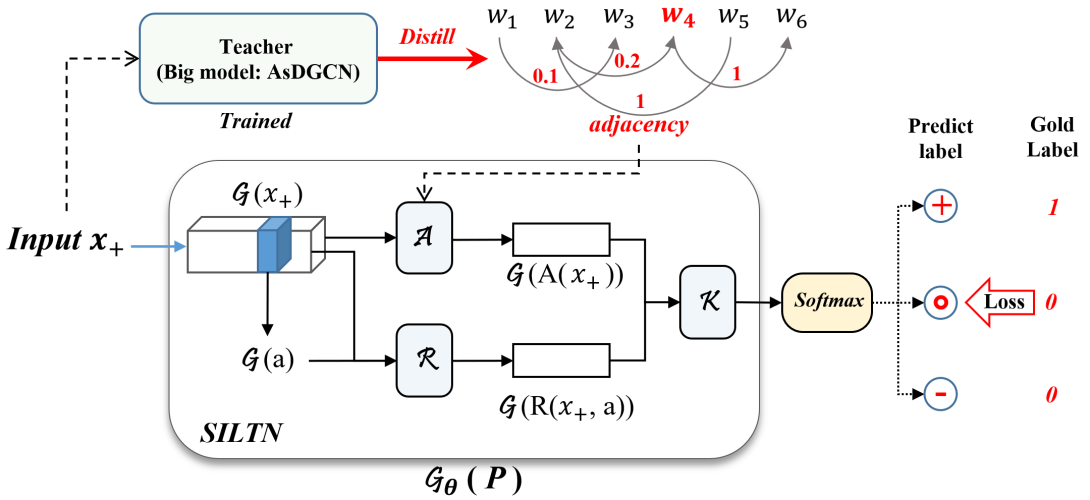

Figure 2: Framework of the proposed SILTN.

$$\mathcal{G}(A(x)) = \sigma(\sum_{i=0}^{m} \hat{A}_i \mathcal{G}(x)_i W_i + b), \quad (5)$$

where $\hat{A}$ is the distilled dependency knowledge (see Eq. (9)); σ denotes the active function. Note that W, b are distilled from AsDGCN (share parameters).

Next, $\mathcal{G}(R(x, a))$ is computed by :

$$\begin{aligned} c &= \mathcal{G}(x)\mathcal{G}(a)^{\mathrm{T}}, \\ \mathcal{G}(R(x, a)) &= \mathcal{G}(x)_t softmax(\sum_t c_t), \end{aligned} \quad (6)$$

where $softmax(f_i) = \frac{e^{f_i}}{\sum_j e^{f_j}}$, t denodes the t-th word. Specifically, $\mathcal{G}(a)$ is obtained by selecting the corresponding vector through the index $\{w_a^a, ..., w_{a+m}^a\}$ from $\mathcal{G}(x)$.

After acquiring the representation $\mathcal{G}((R(x, a))$, it is fed into the feed-forward layer to compute $\mathcal{G}(K(A(x), R(x, a)))$. Here, $K(A(x), R(x, a))$ represents the soft logic $A(x) \rightarrow R(x, a)$. Follow soft logic operation in Badreddine et al. (2022), $K(A(x), R(x, a))$ is computed by:

$$\begin{aligned} &\mathcal{G}(K(A(x), R(x, a))) = \\ &1 - \mathcal{G}(A(x)) + \mathcal{G}(A(x)) \cdot \mathcal{G}(R(x, a)) \end{aligned} \quad (7)$$

Then the softmax layer to obtain the sentiment probability distribution:

$$P_S = softmax(W_1 \mathcal{G}(K(A(x), R(x, a))) + b_1), \quad (8)$$

where W_1 and b_1 are trainable parameters.

2.4 TSynKD

2.4.1 Key Idea of TSynKD

The "deepth" of the network structure is shallow, which poses a big challenge to improve the network's performance on ATSA while maintaining an interpretable and simple network structure. An effective solution is to adopt knowledge distillation, which can transfer the knowledge from the large network (teacher) into the small network (student), and improve the student's performance significantly.

2.4.2 Large Network (Bert)

In this paper, we deploy the pre-trained Bert model as the first teacher that produces features learned from large-scale corpus. Specifically, we first fine-tune the Bert model, and then generate the outputs of all training samples. The outputs are then denoted as a big model training objective in *output distillation*. Bert model takes "[CLS] sentence [SEP] aspect-terms [SEP]" as input, which computes the deep representations of sentences and aspects. We denote the output of Bert as P_L.

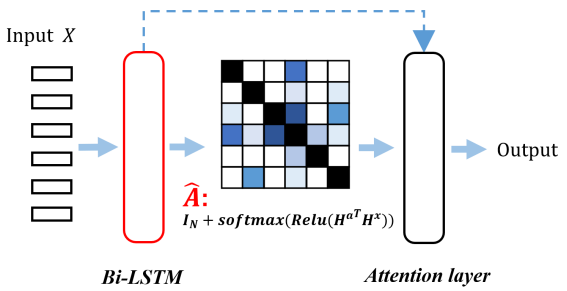

Figure 3: Framework of AsDGCN.

2.4.3 Big Network (AsDGCN)

AsDGCN is the extention of Zhang et al. (2019), which aims to distill dependency knowledge from Bert by *output distillation*, and then transfer the knowledge into the student SILTN. We only briefly discuss sections overlapping with contents in Zhang et al. (2019) so that we can put more emphasis on the new contributions.

Formally, the first layer of AsDGCN is Bi-LSTM network, which can be denoted as $H^x = \{h_1^c, ..., h_2^a, ..., h_n^c\}$, H^c and H^a are the combination of the hidden states of content words and aspect-terms, respectively. The second layer is constructed for learning the dependency relation (denoted as $\hat{A}$) of each word. In particular, we propose a novel attention mechanism, where the attention distributions simulate the adjacency relations. Specifically, we first use the hidden states of aspect-term (H^a) as the attention query to calculate the attention distribution with the hidden state H^x. Then, $\hat{A}$ can be computed as:

$$\hat{A} = I_N + softmax(Relu(H^{a\mathrm{T}} H^x)), \quad (9)$$

where I_N is the identity matrix. After obtaining the adjacency $\hat{A}$, The final graph representation S can be calculated by GCN:

$$S = \sigma(\hat{A}HW + b), \quad (10)$$

where σ represents a non-linear function, W+b are trainable parameters.

Finally, S is then fed into the attention-based layer and followed by the softmax layer to compute a sentiment probability distribution P_B.

2.4.4 Learning

- **Output distillation:**

  The output logits serve as a soft target providing richer supervision than the hard target of the one-hot gold label for the training (Hinton et al., 2015). Given an input sentence x with the gold label y (one-hot), the output logits of the large network (Bert) P_L and the output logits of the big network (AsDGCN) P_B, the loss function of output distillation denotes as:

$$\begin{aligned} L_{od} &= \alpha_1 \cdot CE(P_B, Y) \\ &+ (1 - \alpha_1) \cdot MSE(P_L, P_B) \end{aligned} \quad (11)$$

- **Feature distillation:**

  To capture rich syntactic tree features, we first consider allowing the student SILTN to directly utilize dependency relations $\hat{A}$ from AsDGCN.

  Second, SILTN shares the parameters of syntax layer with AsDGCN (See from Eq. (5) and Eq. (10)).

- **Loss function for SILTN:**

  The loss function of SILTN is similar to *output distillation*, which can be computed as:

$$\begin{aligned} \mathcal{L} &= \alpha_2 \cdot CE(P_S, Y) \\ &+ (1 - \alpha_2) \cdot MSE(P_L, P_S) \end{aligned} \quad (12)$$

Training strategy: From Figure 1, the training of the overall framework has three steps: (1) fine-tune Bert and then predict the sentiment label P_L; (2) train AsDGCN by output distillation (Eq. 11); (3) train SILTN by leveraging $\hat{A}$ (Eq. 9) from the trained AsDGCN and then optimize through the loss function $\mathcal{L}$ (Eq. 12). During inference, the well-trained SILTN can make predictions on its own for the given input.

3 Experiments

Datasets. To evaluate the effectiveness of our method, we conduct extensive experiments on five datasets. **Twitter** dataset is obtained from Dong et al. (2014). There are 1561 positive, 3127 neutral and 1560 negative tweets for training and 692 for the test. **Lap14** and **Rest14** datasets are taken from SemEval-14 Task 4 in Pontiki et al. (2014). Lap14 denotes the laptop reviews and it contains 2328 training texts and 638 test samples. Rest14 composes of the restaurant reviews, it contains 2164 positive, 637 neutral and 807 negative texts for training and 1120 samples for test. **Rest15** is collected from SemEval 2015 task 12 in Pontiki et al. (2015), it contains in total 1204 training samples with three sentiment classes and 542 samples for test. **SPD** is collected from Zhang et al. (2020), the sentences in SPD all contain unique structures, such as conditional statements and subjunctive. It contains 4726 training samples and 1182 test samples.

Baselines We adopt several sentiment classification methods as baselines. **SVM** (Kiritchenko et al., 2014) is an effective traditional mechine learning based method. **LSTM** (Tang et al., 2016a) utilizes the standard LSTM to model the sentiment representation. **IAN** (Ma et al., 2017), **MemNet** (Tang

Model	Twitter		Lap14		Rest14		Rest15		SPD	
	Acc	F1	Acc	F1	Acc	F1	Acc	F1	Acc	F1
1.SVM¶	63.40	63.30	70.49	-	80.16	-	-	-	-	-
2.LSTM†	69.56	66.42	69.28	63.21	78.13	67.42	77.37	52.97	61.16	59.17
3.Memnet¶	71.48	68.14	70.64	65.19	79.61	68.14	77.31	56.17	61.42	57.56
4.AOA†	72.30	68.20	72.62	66.97	79.97	69.59	78.17	57.21	63.76	64.41
5.IAN†	72.50	68.14	72.05	67.38	79.26	70.12	78.54	52.21	63.76	63.96
6.CapsNet¶	-	-	-	-	69.63	69.63	78.14	61.57	-	-
7.TNet-LF†	72.98	71.43	74.61	70.14	80.40	70.57	78.47	59.12	63.76	64.96
8.MIMLLN¶	-	- -	-	-	81.06	71.25	78.27	60.59	-	-
Syntax-aware methods										
9.PRNN¶	-	-	-	-	66.20	59.32	-	-	-	-
10.SAttn¶	-	-	72.57	69.13	80.45	71.26	-	-	-	-
11.ASGCN†	72.15	71.00	71.05	70.72	80.86	72.73	79.89	59.47	66.24	65.24
12.R-GAT†	71.56	71.07	72.49	71.01	73.83	72.14	78.92	61.24	66.87	65.14
Ours (SILTN)										
-sp	70.95	68.73	72.57	68.14	81.16	71.87	79.97	57.76	63.74	60.12
-(dep)	**73.12**	72.25	**76.96**	72.95‡	**83.13**	75.12‡	81.01	64.11	67.60	67.38‡
-(dis-dep)	73.01	**73.07**‡	76.77	**73.03**‡	83.02	**75.86**‡	**81.37**	**64.26**‡	**67.94**	**68.01**‡

Table 1: Evaluation results (%) on none-pretrained based methods. The best result on each task is in bold. The mark ¶ refers to the results reported in the original papers, while † mark refers to the open implementation, ‡ mark refers to p-value < 0.05 when comparing with the best competitor.

	Twitter	Rest14	Lap14
Bert	74.41	76.21	75.10
Bert-PT	-	76.48	75.08
LCF-BERT	73.34	75.03	76.26
AEN-Bert	73.13	73.76	76.31
RoBerta-ASC	-	75.12	70.52
Bert-ASGCN	74.67	76.29	75.96
Bert-RGAT	74.88	74.88	74.07
SILTN-Bert	**75.52**	**77.04**	**76.34**

Table 2: Evaluation results (F1 %) compared with pre-trained models.

et al., 2016b), **AOA** (Huang et al., 2018)and **TNet-LF** (Li et al., 2018) are attention-based methods. **ASGCN** (Zhang et al., 2019) and **R-GAT** (Wang et al., 2020) use GCN to model the dependency tree graph for ATSA. **MIMLLN** (Li et al., 2020) treats the aspect category as the key instances. **PRNN** (Nguyen and Shirai, 2015) takes both dependency and constituent trees into LSTM. **SAttn** further integrates attention mechanism with PhraseRNN. **Bert** (Devlin et al., 2019) is a pre-trained BERT model to perform ATSA. We convert the given context and target to "[CLS] + aspect-term + [SEP] + context" structure. Further we select several variants Bert-based model **Bert-PT** (Xu et al., 2019),**AEN-Bert** (Song et al., 2019), **LCF-Bert** (Zeng et al., 2019), **RoBerta-ASC**, **Bert-ASGCN** and **Bert-RGAT**(Dai et al., 2021).

Variant of SILTN. To get a trade-off between interpretability, computational efficiency and high accuracy, we increase or simplify trainable parameters for SILTN, and compare their prediction accuracy and prediction time. **SILTN-sp**: This method maximally simplifies the training parameters. We expect that each grounding of SILTN is a simple trainable tensor production, and for the first layer of grounding $\mathcal{G}(x)$, we directly initialize it with glove vector without adding any text representation structure. **SILTN-LSTM**: To sacrifice the interpretability and improve inferring performance, we use one LSTM layer to construct $\mathcal{G}(x)$. Note that, the **SILTN-(dep)** denotes the dependency knowledge acquired from the external tools, and the dependency knowledge of **SILTN-(dis-dep)** is distilled from Big network AsDGCN. **SILTN-Bert**: To further improve the inferring performance, we use pretrained Bert to model $\mathcal{G}(x)$.

Experimental Setting. In this paper, we utilize 300-dimensional pre-trained GloVe vectors to initialize the word embeddings. The dimensions of the hidden state of Bi-LSTM is 128. The scale weight $\alpha_{\{1,2\}}$ is 0.01. The model is optimized with

the Adam optimization algorithm with the batch size of 32 and the learning rate is 0.001. As in Zhang et al. (2019), We use accuracy and macro-averaged F1 score as the evaluation metrics, which are widely adopted in sentiment classification. We compute the metrics independently for each class and then take the average (hence treating all classes equally), as the final performance.

3.1 Task Setup and Quantitative Results

To evaluate the stability of the model, following Zhang et al. (2019), we run the method three times and summarize the best results in Tables 1. Besides, we adopt the *t*-test to confirm the significance of differences between the other methods with a p-value of 0.05.

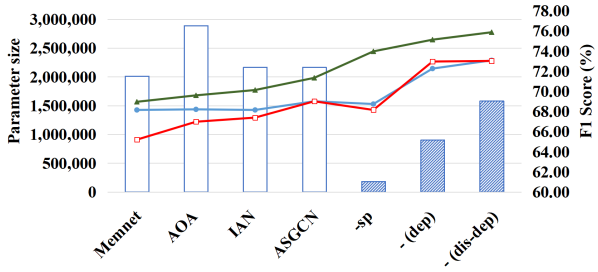

(a) None-Bert model.

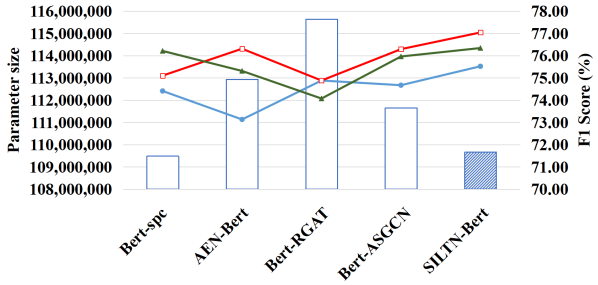

(b) Bert-based model.

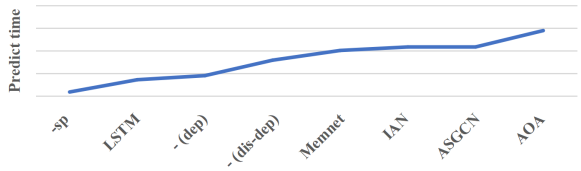

(c) Predict time (sort by time increase).

Figure 4: Comparison results of the size of training parameters vs. F1 score.

According to the results, we can observe that our model consistently outperforms the compared baseline methods. **First**, SILTN outperforms all none-syntax baselines (No. 1-8) on all datasets, which indicates that SILTN has better ability to infer the sentiment polarities by utilizing dependency knowledge. For example, compared with the best competitor of methods 1-8, SILTN(dis-dep) improves 1.64% on Twitter, 4.61% on Rest14 and 2.69% on Rest15 for F1 score, respectively. This is because SILTN utilizes the dependency knowledge distilled from the big model, which enriches the learning ability between the word and aspect-terms. **Second**, compared with the syntax-aware models (No. 9-12), SILTN improves 2.02% on Lap14, 3.13% on Rest14 and 3.02% on Rest15 for F1 score, respectively. The reason is that the syntax dependency tree from the SpaCy tool utilized by the conventional method may introduce additional errors especially for the text are short and informal. **Third**, to further imporve the infering accuracy, we compare our method with Bert-based methods and give the results in Table 2. The result shows SILTN-Bert achieves the best F1 score over the competitors. Note that, SILTN-(dis-dep) reduces parameters by 99.17% compared to the Bert model; however, SILTN still achieves compatible results.

In sum, the advantage of SILTN comes from its two characteristics: (i) TSynKD provides adequate syntax knowledge, making the SILTN able to combine the prior knowledge effectively. (ii) SILTN uses fewer parameters while interpretable, which leads to high efficiency and high accuracy.

3.1.1 Cost Efficiency vs. Accuracy

Figure 4 summarized the number of parameters and the F1 score results. First, compare SILTN with the baselines, SILTN contains fewer parameters but achieves state-of-the-art results. For example, *-dep* has 58.32% less parameters than ASGCN, but the performance improved by 4.46% on Rest15 and 2.23% on Lap14. Moreover, the predict time of *-dep* is comparable to simple LSTM model. Second, compared with SILTN-sp, which has the fewest parameters (even fewer 74.91% than standard LSTM), our method can still obtain competitive results to the conventional best attention mechanism model (ASGCN). In sum, the results show the effectiveness of the TSynKD framework, and it proves that our model can strike a balance between computational efficiency and high accuracy.

3.2 Ablation Study

To study the impact of each component of the proposed method, we implement the ablation test to remove the proposed component denoted as -w/o.

Specifically, **-w/o TSynKD**: SILTN without the TSynKD framework, and utilize the standard de-

pendency tree[1] as the syntax knowledge. **-w/o TSynKD**$_{OT}$: SILTN trained by utilizing only *output distillation strategy* and utilize the standard dependency tree as the syntax knowledge. **-w/o TSynKD**$_{FT}$: SILTN trained by utilizing only *feature distillation strategy* and utilize the standard dependency tree as the syntax knowledge.

Methods	Twitter	Lap14	Rest14
SILTN-sp	68.73	68.14	71.87
-w/o TSynKD	65.84	65.44	68.53
-w/o TSynKD$_{OT}$	66.73	66.22	70.20
-w/o TSynKD$_{FT}$	67.42	66.74	69.66

Table 3: Ablation study results (F1%).

The ablation results are summarized in Table 3. According to the results, we can summarize that all the proposed components contribute a great improvement to SILTN. In particular, the accuracy score decreases sharply when discarding the TSynKD framework. This is within our expectation since the TSynKD injects knowledge from the large network and the dependency knowledge from the big model into SILTN. For example, the F1 score drops 2.89%, 2.70% and 3.34% for Twitter, Lap14 and Rest14, respectively. In addition, the proposed two-stage distillation framework also contributes to the effectiveness of SILTN. For example, the F1 score decreases 1.92% for Twitter when discarding *feature distillation*, and decreases 2.21% for Rest14 when discarding *output distillation*. Not surprisingly, combining all factors achieves the best performance for all the experiments.

3.3 Visualized of Distilled Dependency Relation

The dependency knowledge enables SILTN to utilize tree structures for capturing the corrected aspect-related words. To understand how the distilled model promotes the mutual learning of dependency structures, we empirically visualize the adjacency relation based on a test example: "food is delicious but price is expensive". This sample contains "food" and "price" two aspect-terms. The visualization is summarized in Figure 5. Figure 5. (a) is the standard dependency structure from SpaCy tools, which the conventional methods utilize. Both "food" and "price" share the same adjacency matrix. Figure 5. (b) and (c) are acquired

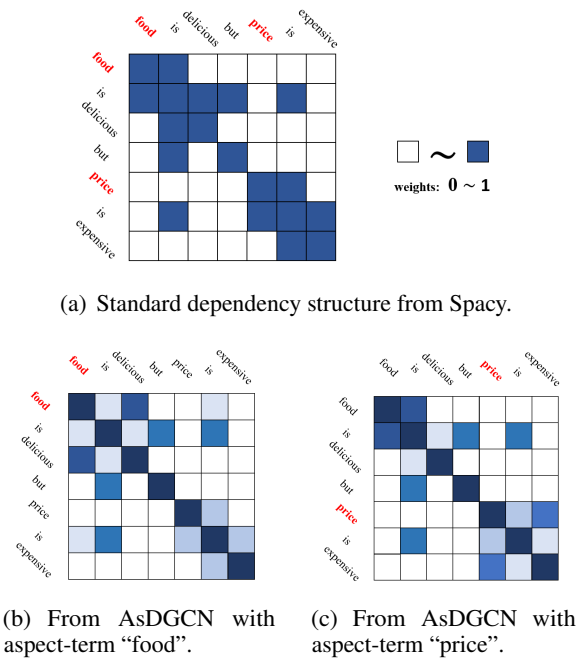

(a) Standard dependency structure from Spacy.

(b) From AsDGCN with aspect-term "food".

(c) From AsDGCN with aspect-term "price".

Figure 5: Visualized of distilled dependency relation.

from Eq. 9 for our AsDGCN. They generate adjacency matrices for each aspect-term, respectively. For example, in Figure 5. (b) "food" increases the dependency weight associated with "delicious"; and in Figure 5. (c), the "price" directly connects with "expensive".

4 Conclusion

In this paper, we propose a novel Sentiment Interpretable Logic Tensor Network (SILTN). SILTN is interpretable because it is constructed followed by first-order logic language (FOL). The sentiment inferring process can be decomposed into different *grounding*, which is constructed by the simple neural network; therefore, SILTN is computationally efficient. To achieve high inferring accuracy, we propose a two-stage syntax knowledge distillation (TSynKD) strategy. TSynKD consists of two distillation stages with three networks: Bert, AsDGCN and SILTN. The first distill stage refers to the output distillation, which makes the Bert output logits as the training objective of AsDGCN. In AsDGCN, we develop a novel attention structure to learn the aspect-specific dependency knowledge through output distillation. The second stage is *feature distillation*, which allows a SILTN to learn from AsDGCN's intermediate feature representations. Extensive experiments have been conducted on 5 real-world datasets. The experimental results show that the proposed SILTN with

[1]Spacy tools: https://spacy.io/

the TSynKD strategy significantly outperforms the conventional attention-based methods, and achieve compatible results compared with the state-of-the-art Bert-based methods for aspect-term sentiment analysis.

5 Acknowledgements

This research is supported by the Stable Support Project for Shenzhen Higher Education Institutions (SZWD2021011).

References

Samy Badreddine, Artur d'Avila Garcez, Luciano Serafini, and Michael Spranger. 2022. Logic tensor networks. *Artif. Intell.*, 303:103649.

Junqi Dai, Hang Yan, Tianxiang Sun, Pengfei Liu, and Xipeng Qiu. 2021. Does syntax matter? A strong baseline for aspect-based sentiment analysis with roberta. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT 2021, Online, June 6-11, 2021*, pages 1816–1829. Association for Computational Linguistics.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT 2019, Minneapolis, MN, USA, June 2-7, 2019, Volume 1 (Long and Short Papers)*, pages 4171–4186. Association for Computational Linguistics.

Li Dong, Furu Wei, Chuanqi Tan, Duyu Tang, Ming Zhou, and Ke Xu. 2014. Adaptive recursive neural network for target-dependent twitter sentiment classification. In *Proceedings of the 52nd annual meeting of the association for computational linguistics (volume 2: Short papers)*, pages 49–54.

Geoffrey Hinton, Oriol Vinyals, and Jeff Dean. 2015. Distilling the knowledge in a neural network. *arXiv preprint arXiv:1503.02531*.

Binxuan Huang, Yanglan Ou, and Kathleen M Carley. 2018. Aspect level sentiment classification with attention-over-attention neural networks. In *International Conference on Social Computing, Behavioral-Cultural Modeling and Prediction and Behavior Representation in Modeling and Simulation*, pages 197–206. Springer.

Svetlana Kiritchenko, Xiaodan Zhu, Colin Cherry, and Saif Mohammad. 2014. Nrc-canada-2014: Detecting aspects and sentiment in customer reviews. In *Proceedings of the 8th International Workshop on Semantic Evaluation (SemEval 2014)*, pages 437–442.

Xin Li, Lidong Bing, Wai Lam, and Bei Shi. 2018. Transformation networks for target-oriented sentiment classification. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 946–956.

Yuncong Li, Cunxiang Yin, Sheng-hua Zhong, and Xu Pan. 2020. Multi-instance multi-label learning networks for aspect-category sentiment analysis. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing*, pages 3550–3560. Association for Computational Linguistics.

Dehong Ma, Sujian Li, Xiaodong Zhang, and Houfeng Wang. 2017. Interactive attention networks for aspect-level sentiment classification. In *Proceedings of the 26th International Joint Conference on Artificial Intelligence*, pages 4068–4074. AAAI Press.

Thien Hai Nguyen and Kiyoaki Shirai. 2015. Phrasernn: Phrase recursive neural network for aspect-based sentiment analysis. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 2509–2514.

Maria Pontiki, Dimitris Galanis, Haris Papageorgiou, Suresh Manandhar, and Ion Androutsopoulos. 2015. Semeval-2015 task 12: Aspect based sentiment analysis. In *Proceedings of the 9th International Workshop on Semantic Evaluation (SemEval 2015)*, pages 486–495.

Maria Pontiki, Dimitris Galanis, John Pavlopoulos, Harris Papageorgiou, Ion Androutsopoulos, and Suresh Manandhar. 2014. SemEval-2014 task 4: Aspect based sentiment analysis. In *Proceedings of the 8th International Workshop on Semantic Evaluation (SemEval 2014)*, pages 27–35, Dublin, Ireland. Association for Computational Linguistics.

Soujanya Poria, Nir Ofek, Alexander F. Gelbukh, Amir Hussain, and Lior Rokach. 2014. Dependency tree-based rules for concept-level aspect-based sentiment analysis. volume 475, pages 41–47.

Youwei Song, Jiahai Wang, Tao Jiang, Zhiyue Liu, and Yanghui Rao. 2019. Attentional encoder network for targeted sentiment classification. *arXiv preprint arXiv:1902.09314*.

Duyu Tang, Bing Qin, Xiaocheng Feng, and Ting Liu. 2016a. Effective lstms for target-dependent sentiment classification. In *Proceedings of COLING 2016, the 26th International Conference on Computational Linguistics: Technical Papers*, pages 3298–3307.

Duyu Tang, Bing Qin, and Ting Liu. 2016b. Aspect level sentiment classification with deep memory network. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 214–224.

Kai Wang, Weizhou Shen, Yunyi Yang, Xiaojun Quan, and Rui Wang. 2020. Relational graph attention network for aspect-based sentiment analysis. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 3229–3238. Association for Computational Linguistics.

Hu Xu, Bing Liu, Lei Shu, and S Yu Philip. 2019. Bert post-training for review reading comprehension and aspect-based sentiment analysis. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 2324–2335.

Salud M. Jiménez Zafra, María Teresa Martín Valdivia, Eugenio Martínez-Cámara, and Luis Alfonso Ureña López. 2019. Studying the scope of negation for spanish sentiment analysis on twitter. *IEEE Trans. Affect. Comput.*, 10(1):129–141.

Biqing Zeng, Heng Yang, Ruyang Xu, Wu Zhou, and Xuli Han. 2019. Lcf: A local context focus mechanism for aspect-based sentiment classification. *Applied Sciences*, 9(16):3389.

Bowen Zhang, Xutao Li, Xiaofei Xu, Ka-Cheong Leung, Zhiyao Chen, and Yunming Ye. 2020. Knowledge guided capsule attention network for aspect-based sentiment analysis. *IEEE ACM Trans. Audio Speech Lang. Process.*, 28:2538–2551.

Chen Zhang, Qiuchi Li, and Dawei Song. 2019. Aspect-based sentiment classification with aspect-specific graph convolutional networks. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 4560–4570.

Detecting Minority Arguments for Mutual Understanding: A Moderation Tool for the Online Climate Change Debate

Cedric Waterschoot and **Ernst van den Hemel**
KNAW Meertens Instituut
Oudezijds Achterburgwal 185
1012DK Amsterdam, The Netherlands
cedric.waterschoot@meertens.knaw.nl
ernst.van.den.hemel2@meertens.knaw.nl

Antal van den Bosch
Utrecht Institute of Linguistics OTS
Utrecht University
The Netherlands
a.p.j.vandenbosch@uu.nl

Abstract

Moderating user comments and promoting healthy understanding is a challenging task, especially in the context of polarized topics such as climate change. We propose a moderation tool to assist moderators in promoting mutual understanding in regard to this topic. The approach is twofold. First, we train classifiers to label incoming posts for the arguments they entail, with a specific focus on minority arguments. We apply active learning to further supplement the training data with rare arguments. Second, we dive deeper into singular arguments and extract the lexical patterns that distinguish each argument from the others. Our findings indicate that climate change arguments form clearly separable clusters in the embedding space. These classes are characterized by their own unique lexical patterns that provide a quick insight in an argument's key concepts. Additionally, supplementing our training data was necessary for our classifiers to be able to adequately recognize rare arguments. We argue that this detailed rundown of each argument provides insight into where others are coming from. These computational approaches can be part of the toolkit for content moderators and researchers struggling with other polarized debates.

1 Introduction

Even though a consensus has existed within the scientific community on the topic of human-caused climate change for some time, the online debate remains very polarized. Online comment spaces are typically overwhelmed with a large quantity of contributions. This information flood hinders the promotion of mutual understanding and inclusivity in debate spaces. Additionally, climate change presents a splintered debate with niche opinions and many viewpoints. The recognition of these niche arguments are vital to support the moderator in adhering to the heterogeneous discussion that climate change presents. This setting presents opportunities for mutual understanding by improving issue awareness and the quality of deliberation.

In this paper, we construct a twofold approach to support mutual understanding in the online climate change discussion. First, we aim to classify posts for the argument they present. Second, we dive deeper into singular arguments to create an overview of the lexical patterns in each argument-specific sub-corpus. We conclude the paper by discussing the limitations of modeling nuanced argumentation by a computational method and link our approach to fields struggling with content moderation and polarized debates.

2 Background

2.1 Argument Mining & Stance Detection

Our application falls under the umbrella of 'argument mining' and 'stance detection'. Within Natural Language Processing, argument mining is defined as the automated identification and extraction of argumentation found in natural language (Lawrence and Reed, 2019). Following the stark increase in the availability of textual data found on online fora and social media platforms, argument detection tasks have been receiving a lot of attention. The related task of stance detection is aimed at classifying the stance of the producer of a

Proceedings of the 29th International Conference on Computational Linguistics, pages 6715–6725
October 12–17, 2022.

piece of text towards the target topic (Küçük and Can, 2020). This result is often performed over three classes: *in favour ('Pro'), against ('Con') or neutral*.

To define an argument, researchers often look towards the Toulmin model of argument (Toulmin, 2003). Toulmin defined a formal argumentative model comprising of the following five elements: *claim, data, warrant, qualifier, and rebuttal* (Toulmin, 2003). However, textual data from social media or comment platforms tend to fall short of fulfilling these formal requirements due to their briefness and elliptic nature. Researchers have therefore labeled tweets as argumentative when a portion of the formal argumentative structure was present (Bosc et al., 2016). These portions can be a premise, a, conclusion, or the connecting relationship between these two argumentative parts. In this paper, we follow the same operationalization of the definition of arguments.

One factor further complicating these tasks is the influence of context. Context may affect whether an utterance is interpreted as argumentative or not (Carstens and Toni, 2015). Typically, the classification tasks are restricted to features intrinsic to the sentence, post, or utterance, and are blind to context; therefore, resulting models may not be robust across different contexts (Lawrence and Reed, 2019). What makes the contextual factor challenging is the fact that not all content and context is expressed explicitly (Moens, 2018). A lot of this knowledge and expression remains "in the mind of communicator and audience" (Moens, 2018). Some have even argued that, in particular cases, content can be less important than the context it resides in (Opitz and Frank, 2019).

Related to the contextual factor is the importance of previous knowledge in stance detection and annotation. The complexity of stance-taking includes cultural and social aspects (ALDayel and Magdy, 2021). Personal opinions and the aforementioned non-personal aspects make stance detection a non-trivial task (Du Bois, 2007).

In recent years, a range of work has focused on argument detection in online content. The first step of these approaches often relates to making the distinction between argumentative and non-argumentative samples. Addawood and Bashir (2016) perform such a classification while subsequently classifying the evidence type presented within argumentative tweets with Support Vector Machines (SVM) and Decision trees. Naderi and Hirst (2016) created a corpus of parliamentary discourse labelled as 'Pro' and 'Con' on the subject of gay marriage, alongside pre-defined argumentation specific to the topic. Cross-topic experiments pose even greater challenges than single topic argument classification. Stab et al. (2018) annotated and classified web texts across eight different topics based on the three stance classes (pro, con and neutral). 'Pro/Con' classification on unseen topics has also been done using BERT models, which improved F_1-scores compared to attention-based neural networks (Reimers et al., 2020). In this paper, we follow the methodology set out in the existing literature by creating a single-topic corpus (Naderi and Hirst, 2016; Bosc et al., 2016). The annotation scheme is based on pre-defined arguments in the discussion that are already explored in the wider literature on the selected topic of climate change.

2.2 Climate change argumentation

In the upcoming paragraphs, we outline the specific arguments that have been defined in the literature. Argumentation is divided between 'Pro', i.e. those acknowledging climate change is a human-caused threat, and 'Con', arguments that deny climate change as a problem caused by human action.

The latter seems to be the most diverse cluster. Rahmstorf (2003) proposes a three-way distinction in climate change denial arguments: *(1) Impact scepticism, (2) Trend scepticism and (3) Attribution scepticism*. Trend scepticism rejects the warming trend all together, while attribution sceptics question whether human activity is the cause (Rahmstorf, 2003). The former seems to be an idea that is disappearing (Rahmstorf, 2003; Dunlap and McCright, 2012). On the other hand, impact scepticism states that the consequences from climate change might not be that bad (Rahmstorf, 2003). Examples of this argument are statements detailing that a warmer climate is desirable or that we can simply mitigate the effects. Dunlap and McCright (2012) detail the same three movements against human-caused climate change: (1) no warming, (2) not caused by human activity and, (3) the 'non-problemacity' of climate change. The latter focus of 'non-problemacity' seems to be based on a dominant social paradigm that our species is able to exert control over nature (McCright and Dunlap, 2003). This control directly leads to the conclusion that climate change cannot pose a threat (Bord

Stance	Argument (labels)	Explanation
Con	Impact scepticism	Denial of consequences
	Attribution scepticism	Denial of human influence
	Trend scepticism	Denial of warming trend
	No consensus	Denial of consensus among scientists
	Bad science	Accusation of bad models/forecasts used in science
	Conspiracy theories	Umbrella category for all conspiracy-related content
Pro	Anthropogenic climate change (ACC)	Climate change is caused by human activity
None	No argument	No relevant argument is present / post is off-topic

Table 1: Climate change argumentation & annotation scheme

et al., 2000; Poortinga et al., 2011).

Aside from these three forms of scepticism, climate change denial also focuses on the scientific community. More specifically, the existence of a scientific consensus is often questioned (Leiserowitz et al., 2010). We label this argument *No consensus*. Interestingly, a consensus among scientists has long existed (Doran and Zimmerman, 2009; Oreskes, 2005). While it is uncertain as to why this consensus is questioned, a potential explanation lies in the fact that the scientists have long shied away from making dramatic warnings or conclusions in publications (Brysse et al., 2013). A second science-focused argument against climate change takes aim at the science and models themselves, which we label as *'Bad science'*. The claim posits that the complexity and uncertainty surrounding the climate system is a hurdle for scientists to make rigid forecasts (Poortinga et al., 2011). Pinpointing the exact cause for every reasoning disputing human-caused climate change is difficult if at all possible. However, a number of sources can be found, including organized anti-environmental movements like those found in the U.S. in the 1990s (McCright and Dunlap, 2003), unreliable or incomplete interpretation of scientific evidence (Whitmarsh, 2011) or online content like videos found on Youtube (Allgaier, 2019). These sources are often presented as 'manufacturers of doubt' (Van Linden et al., 2015). A final category arguing against climate change is the *conspiracy-related* class. Content related to conspiracy theories often emerge in polarized debate in the online sphere, even in good-faith discussions (Samory and Mitra, 2018). Similar to the definition of argument, we define 'conspiracy' loosely by not requiring all elements of a conspiracy theory, *agent, action and target*, to be explicitly present (Samory and Mitra, 2018). References to conspiracies in user comments tend to be compact and make use of the most common denominator words for a conspiracy, and further rely on context to complete the conspiratory content.

Those arguing that the current climate crisis is caused by human activity find themselves in a more unified environment, which we label under the term anthropogenic climate change (ACC). By the late 1980s, and after the vast accumulation of evidence, the majority of academics had concluded that anthropogenic climate change was occurring (Leiserowitz, 2007). The argument is in practice quite straight-forward and is reflected in the literature in the form of surveys of experts (Doran and Zimmerman, 2009) or literature reviews of the field (Oreskes, 2005). Additionally, references are often made to the reports from the Intergovernmental Panel for Climate Change (IPCC) (Masson-Delmotte et al., 2021).

2.3 Deliberation on online platforms

This paper focuses on mutual understanding in the climate change debate in the setting of online comment platforms. In the previous paragraphs, we outlined the polarized argumentation that occurs in the discussion. Briefly, mutual understanding is established through comprehension of what others are trying to do or say as well as why (Margaret, 1994). Exposure to other opinions can improve out-group tolerance, which in turn can facilitate mutual understanding (Mutz and Mondak, 2006; Andersen and Hansen, 2007). Evidence indeed shows that these heterogeneous environments are important for facilitating deliberative qualities (Suiter et al., 2016). A vital part of this process is the exposure to conflicting views, which promotes debate participation (Suiter et al., 2016). Online platforms can develop this deliberative atmosphere further. Hearing out marginalized argumentative camps through active facilitation may fundamentally improve the deliberative properties of a discussion (Strandberg et al., 2017). Experimental evidence indeed suggests that opinion polarization can be deconstructed through

the implementation and facilitation of deliberative norms, as is the goal in moderated comment spaces (Grönlund et al., 2015). Thus, designing online fora with deliberative norms in mind, such as inclusion, justification, and equality of discussion, can result in a suitable comment space for mutual understanding in the climate change discussion (Wright and Street, 2007).

3 Methodology

3.1 Data collection & annotation

We accessed a large dataset of comments from the platform NUjij, the discussion platform of online Dutch newspaper NU.nl. All contributions were posted in 2020, are in Dutch and include comments that were removed by moderators. The presence of these comments can be vital for our focus on minority classes, as we need training data for rare or unwanted arguments as well. First, we filtered out all comments that were not placed under articles with the tag *climate*. These tags originate from the journalists and editors themselves. This initial filtering step resulted in a comment pool of $43,106$ posts. From this climate dataset, we randomly sampled $3,000$ posts for our initial annotation. Furthermore, we sampled 500 extra comments to create a separate validation dataset that will be used to validate each model in upcoming sections [1]. Annotation was done following the scheme presented in Table 1. To derive inter-annotator agreement, subsets of the original data were labelled by two additional annotators. A subset of the original dataset (n=250) was given to two independent annotators. To inform their choices, we created a document with the argumentation scheme. This sheet included clear explanations for each argument that we derived from the climate change literature, alongside examples of comments that contained the argument. These examples were not part of the annotated data. Following this procedure, we achieved a Krippendorf's alpha of 0.73.

3.2 Argument classification

Our particular task consists of a multiclass classification with eight different labels (see Table 1). We split the original dataset containing $3,000$ posts into a training (80%) and test set (20%). This test set remained constant over all versions in this paper, similar to the validation data. As a classifier, we used a pre-trained Dutch transformer-based language model, RobBERT, and finetuned it on the training data (Delobelle et al., 2020). More specifically, we employed the version aimed at sequence classification, which adds a linear classification head on top of the pooled output (Wolf et al., 2020; Delobelle et al., 2020). The final models had a batch size of 32, a learning rate of $5e^{-5}$, optimized with AdamW (Loshchilov and Hutter, 2019) and were trained for ten epochs. The best performing classifiers were achieved after two epochs.

3.3 Minority argument supplementation

During the annotation process, it became clear that certain argumentation classes were extremely rare in 'natural' discussion (Table 2). The bulk of comments were either 'no argument/ off-topic' or 'anthropogenic climate change'. The scarcity made classification of these nuanced cases difficult. With the specific goal of finding minority arguments to boost heterogeneous debate, it was vital to obtain and annotate more of these scarce comments. We opted for an active learning approach to get a better grip on minority classes and to counter possible frequency-related bias in our classification results.

In order to obtain more minority class comments for our training data, we employed a 'query-by-committee' active learning strategy (Zhao et al., 2006). The goal is to filter out more minority arguments that will subsequently be added to the training data to finetune RobBERT further (Figure 1). First, we extract the BERT embeddings from our primary RobBert model (finetuned on only the original data) as input for the first active learning committee. The committee is a collection of five classifiers implemented with Scikit-learn (Pedregosa et al., 2011; Danka and Horvath): *(1) Random Forest, (2) Support Vector Machine (SVM) (radial), (3) SVM (polynomial), (4) SVM (linear) and, (5) gradient boosting classifier*. Each learner within the committee starts with 10 labelled posts as initial training data. With every iteration, a new post from the original data is queried based on the disagreement within the committee, calculated with Kullback-Leibler divergence (Zhao et al., 2006). This sample is subsequently added to the training data. This process is repeated for 250 iterations.

Such a trained committee can be used for prediction, but more importantly for our application, we extracted the uncertainty measure for unseen posts. In this case, the uncertainty is computed as

[1] Supplementary materials found at: github.com/Cwaterschoot/Minority_Argumentation

$1 - class\_probability$. This process is visualized in Figure 2. Each learner in the committee assigns probabilities to every comment for each of the eight classes. We obtain the $class\_probability$ by averaging these probabilities per class across the five learners, resulting in eight probability scores. We take the argument with the highest average class probability for the uncertainty calculation. For example, a comment that is difficult to classify may only have a class probability score of 0.3, which equals a high uncertainty score equal to 0.7 (Figure 2).

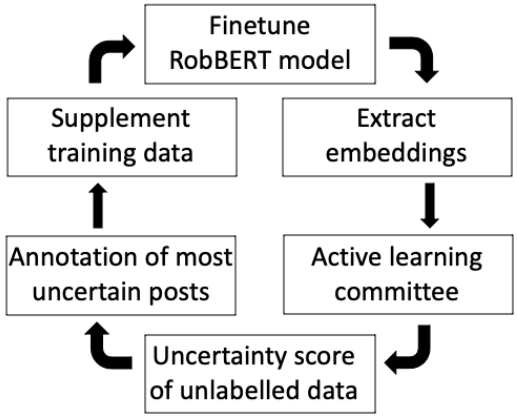

Figure 1: Active learning approach and supplementation of training data

We randomly sampled $10,000$ unseen posts from the climate tagged dataset (containing in total $43,106$ comments) and extracted the uncertainty for each comment. Subsequently, we annotated the $1,000$ most uncertain posts from this collection. Table 2 indicates that this task achieved our goal, namely to relatively increase the number of arguments from minority classes compared to the non-argumentative/off-topic category. Uncertain posts were annotated by a human annotator. Predictions from the committee were disregarded. We repeated this circular procedure a second time *(wave 2)* to add more argumentative posts to the training data (Table 2).

After each wave of newly annotated data, we continued finetuning RobBERT using the previous version as the starting point (see Figure 1). Following this looping procedure, we obtained three versions:(1) *v1* based on the original data, (2) *v2* consisting of v1 supplemented with the first wave and, (3) a fine-tuned version of v2 using both waves of uncertain posts (*v3*). As stated in the previous section, these versions have a linear classification head. Additionally, we extracted the embeddings from all three RobBERT models as input for an active learning committee. Naturally, both the v1 and v2 embeddings are paired with the committees we had used to obtain the uncertain posts. To classify comments based on the v3 embeddings, we trained a third committee following the exact same procedure.

Argument	Original	Wave 1	Wave 2
Impact scepticism	0.02	0.05	0.04
Attribution scepticism	0.03	0.09	0.11
Trend scepticism	0.01	0.01	0.015
No consensus	0.01	0.01	0.004
Bad science	0.01	0.04	0.057
Conspiracy theories	0.01	0.04	0.042
ACC	0.19	0.40	0.30
No argument/off-topic	0.72	0.36	0.42

Table 2: Original data versus uncertain posts. Numbers are fractions of 1 (e.g. $0.72 = 72\%$)

3.4 Patterns in argumentation

Previous sections outlined the automatic annotation of incoming comments for the argument it presents in order to aid moderators in balancing the discussion. Additionally, we aim to boost mutual understanding by diving deeper into what each argument brings to the table. It is important to comprehend the different viewpoints and arguments.

Unique patterns for each argument, i.e. those that have significant presence in one argument compared to all others, were analysed. First, we lowercased the entire corpus and removed stopwords. Subsequently, the corpus was split based on the eight argumentative classes. We used Colibri Core to collect recurring patterns in each subcorpus (van Gompel and van den Bosch, 2016). Following the procedure outlined in van Gompel and van den Bosch (2016), the first step was to class encode the corpus. Subsequently, we created an unindexed pattern model entailing the word n-grams occurring at least twice and with a maximum length of eight tokens. We compared the collection of patterns belonging to a single argument with the seven other argumentative subcorpora taken together. To make this comparison, we utilized the log-likelihood (L) function outlined by Rayson and Garside (2000).

4 Results

4.1 Argument classification

The automatic labelling of posts for the argument it presents may assist moderators in maintaining the

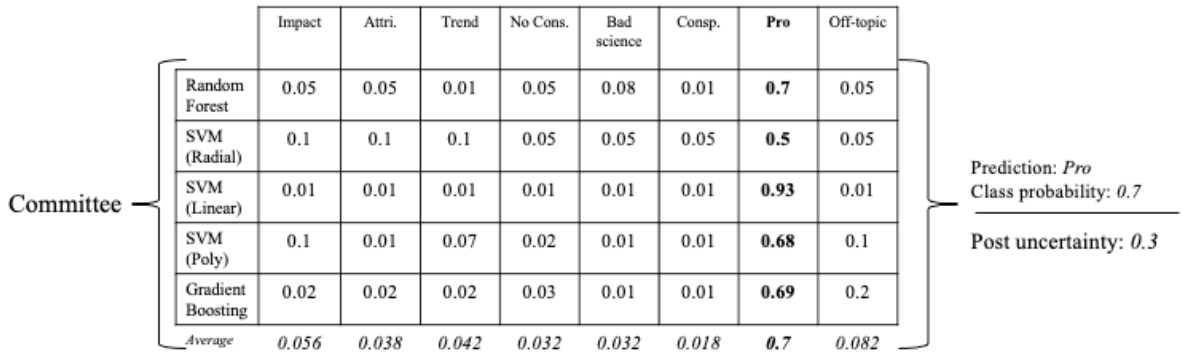

Figure 2: Calculating uncertainty using the active learning committee (fictional post)

desired form of discussion. As outlined earlier, we finetuned a total of three RobBERT models alongside active learning committees that have been used to tag unseen posts for classification uncertainty. Additionally, these committees are used as a classifier on top of the embeddings from each RobBERT model. Each committee consists of five learners and predict arguments by averaging class probabilities within the committee.

Version	Precision	Recall	F1
RobBERT v1 (original data)	0.65	0.51	0.55
RobBERT v1 + committee	0.75	0.50	0.58
RobBERT v2 (original + wave 1)	0.65	0.62	0.62
RobBERT v2 + committee	0.81	0.60	0.64
RobBERT v3 (original + wave 1&2)	0.88	0.68	0.75
RobBERT v3 + committee	0.94	0.67	**0.78**
Random forest (Baseline) [2]			0.25

Table 3: Classification scores on validation data (macro scores)

Table 3 displays the classification metrics on the validation data. Classifying comments using the linear head on top of RobBERT underperforms the committee with each version. The latter improves the macro F1-score score by two to three percentage points by boosting the macro precision score slightly at the expense of the macro recall. RobBERT v3 paired with the committee of classifiers, which is trained on the original training data supplemented with two waves of uncertain posts, outperformed all other versions and achieves a macro F1-score of 0.78.

We constructed the active learning approach to improve the recognition of minority arguments. Table 4 shows that certain arguments like *'Consensus denial', 'Bad science' and 'Conspiracy theories'* posed severe problems for earlier versions. The third iteration of models, which included two waves of uncertain posts in the training data, produced improved F1-scores on the validation set (Table 4). The precision scores for each argument reaches very high levels. This is due to the fact that certain classes have a small number of comments in the data. Impact scepticism is found in 9 comments in the validation data, which is still more than trend scepticism ($n = 2$) and no consensus ($n = 3$). These minority arguments can lead to precision scores that are misleadingly high. For example, one post labelled trend scepticism is the only comment that gets labelled as such by the classifier, leading to a perfect precision score, while recall (0.5 for each version) lacks due to the fact that the other comment belonging to the trend scepticism class is never correctly detected.

Figure 3 shows a two-dimensional representation of the embeddings extracted from RobBERT v3. The arguments, including the relatively rare ones, form noticeable clusters in the embedding space. In the next section, we look at the language and patterns within each argument. Patterns that are distinctively found in a single argument make these arguments distinguishable.

4.2 Argument vocabulary

We previously focused on the computational recognition of climate change argumentation presented in online comments. Additionally, Figure 3 showed that the arguments form visible clusters in the embedding space, hinting at unique vocabulary and patterns within the arguments. We particularly aimed to recognize minority standpoints in order to present the whole range of online opinions. Subsequently, it is important that users and moderators understand what is being said. The discussion of polarized discussion may be boosted by not only

[2] Three classes were not predicted

Version	Impact	Attribution	Trend	Consensus	Bad science	Conspiracy	Pro
RobBERT v1	0.47	0.68	0.67	0.2	0.42	0.4	0.63
RobBERT v1 + committee	0.62	0.70	0.67	0.4	0.33	0.43	0.62
RobBERT v2	0.63	0.69	0.67	0.4	0.45	0.5	0.71
RobBERT v2 + committee	0.75	0.67	0.67	0.67	0.2	0.59	0.72
RobBERT v3	**0.8**	0.72	0.67	**0.8**	0.67	0.71	0.74
RobBERT v3 + committee	**0.8**	**0.79**	0.67	**0.8**	**0.71**	**0.75**	**0.77**

Table 4: F1-score per minority argument on validation data

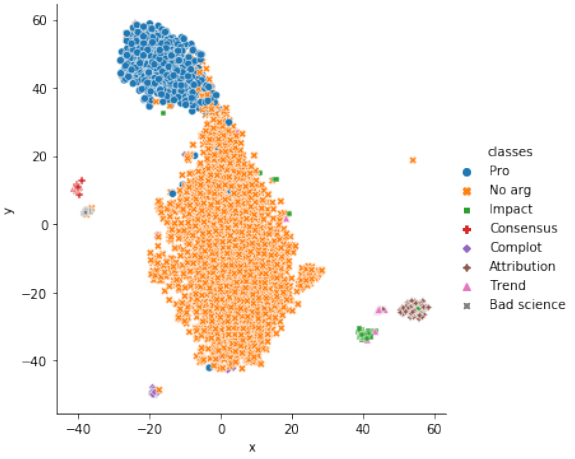

Figure 3: TSNE visualisation of RobBERT v3 embeddings

trying to comprehend others, but to invite them into an heterogeneous debate environment. To achieve this understanding of varying argumentative positions, we have derived the vocabulary and patterns within each argument to showcase what sets each of them apart. Table 5 presents the collected patterns with the highest log-likelihood per argument.

The first class, *no argument/off-topic*, contains a wide collection of patterns. Posts within this category talk about a variety of topics, including the energy transition in The Netherlands and the potential effect of the growing population and consumption (Table 5). These posts are often adjacent to the discussion at hand, but do not present actual argumentation aimed at the cause of climate change.

The argument in favour of human-caused climate change (*ACC*) has a different focus. The political aspect comes to the forefront, expressed in terms like 'voting', 'political' and 'importance'. These comments attempt to rally readers to take action. Another distinctive pattern in this argument details the global component of climate change. Commenters write about the need of unison action.

In our annotation scheme, climate change scepticism is broken down into three subcategories: *Impact, Attribution and Trend*. The latter has very clear patterns that sets this argument apart from the others. Trend sceptics on the comment platform often point towards cold winters, volcanic activity and the existence of ice sheets to reject the warming trend. Furthermore, these sceptics call human-caused climate change a religion and garbage. Generally, it seems that this scepticism is the most straight-forward rejection of human-caused climate change. On the other hand, attribution sceptics seem to be focused on the historical aspect of climate change. We recognize patterns like '[from] all times', 'billion years' and 'million years ago' (Table 5). Alongside this focal point, some attribution sceptics seem to concede that human influence might speed up natural processes ('human influence', 'speed up', 'partly'). These natural processes include the position of the planet relative to the sun and ice age cycles. These topics are not found in other arguments. The third and last argument within the scepticism umbrella, impact, mainly revolves around language claiming it is not necessary to worry about climate change ('worry', 'whine', 'be okay', 'measures' and 'say with certainty').

The two arguments rejecting climate science also rely on specific patterns. On the one hand, we see the dismissal of *scientific consensus* based on very distinctive patterns, e.g. 'prove hypothesis', 'expert' and 'consensus' (Table 5). The accusation of *bad science* revolves around the overarching notion of taking it with a 'grain of salt'. We detect among the patterns 'prediction', 'assumption', 'theories' and 'fearmongering'. These posts urge readers not to take these scientific models too seriously, as they are based on theories and assumptions which do not correspond to real-life circumstances.

The final argumentative class to break down into patterns are the *conspiracy-related content*. We detected conspiracy terms like 'propaganda', 'hoax', 'money' and 'manipulated'. Other unique content references to the 'paris accord' of 2015 and 'acid rain', an environmental issue that received a lot of

Argument	Terms
No argument / off-topic	trash, solution, electricity, somewhere, powerplant, overpopulation, advantage, most people
Impact	worry, previous years, whine, be okay, good economy, say with certainty, measures, stop
Attribution	all times, billion years, speed up, earth sun, human influence, ice age, partly, million years ago
Trend	cold winter, volcanic, tree rings, religion, garbage, ice sheet, every year again
No consensus	consensus, prove, 40 years, 0 co2, assumption, prove hypothesis, phenomenon, expert
Bad science	prediction, assumption, study, grain of salt, case scenario, fearmongering, theories
Conspiracy	paris accord, pro, farmer, propaganda, acid rain, hoax, money, independent, manipulated
ACC (Pro)	use, less people, whole world, houses, voting, political, importance, inhabitants, 3 degrees

Table 5: Argument vocabulary: patterns with highest log-likelihood per argument

attention over the past decades.

5 Discussion

We presented an approach to automatically label online comments for the argument it entails, combined with a deeper dive into each argument in the discussion. In the upcoming paragraphs, we go through some methodological considerations and discuss our approach through the lens of content moderation. Furthermore, we reflect on the usefulness of our approach for other fields that struggle with mutual understanding and opinion polarization.

Translating detailed and nuanced concepts of argumentation into a computational labelling task requires generalization. Poortinga et al. (2011) make the useful distinction between scepticism, uncertainty, and ambivalence. In our annotation scheme, we did not make this specific contrast. Whereas clear-cut scepticism can be rare, as is shown in our data, uncertainty about the anthropogenic causes of climate change might be much more widespread (Whitmarsh, 2011). The dichotomy between uncertainty and scepticism may be an important aspect for mutual understanding and working towards the comprehension and acceptance of human-caused climate change. Unstable or uncertain beliefs can change through contact with scientific cues and information (Jenkins-Smith et al., 2020). In this paper, uncertainty is included within the argument classes, even though the label refers to scepticism. Future research could include this distinction in the methodology to encompass the nuance of polarized debates into the computational approach.

Researchers in the field of content moderation and digital journalism struggle with the concept of mutual understanding, as well as with the implementation of computational technologies (Binns et al., 2017; Ruckenstein and Turunen, 2020). The growing quantity of contributions threatens real-time curation efforts by human moderators. Automatic applications like the one we have presented in this paper are an avenue for assisting human moderators in curating the online comment space (Ruckenstein and Turunen, 2020). While the moderators manage ongoing, interactive processes that are highly dependent on context, computational systems can assist this operation, for example in the form of argument classification and summaries.

Furthermore, research fields that specifically deal with polarized topics struggle with safeguarding civil discussion and mutual understanding. The climate change debate certainly falls within this category. Additionally, online debates on the topic of vaccination lack mutual understanding as well (Jiang et al., 2021). This discussion often lacks heterogeneous discussion due to so-called 'echo chamber' effects (Schmidt et al., 2018). Computational moderation tools, like the one presented in this paper, are an asset for those invested in promoting mutual understanding in these polarized discussions. This approach can be expanded beyond the topic of climate change into other polarized topics. Clearly defined arguments are needed. An example of such a discussion is vaccination, in which clear pro and con sides can be detected (Jiang et al., 2021). Domain-specific research is a requirement to create annotation schemes that adequately entail all minority arguments.

6 Conclusion

In this paper, we created a twofold approach to develop a moderation tool aimed at the climate change debate on online platforms. First, we trained classifiers that label comments for the argument they present. Certain minority arguments, like trend scepticism and accusations of bad science, were very rare. An active learning approach was constructed with the goal of collecting more minority arguments to supplement into our train-

ing dataset. Our best model, after two waves of uncertain posts, achieved a macro F1-score of 0.78. Second, we dove deeper into singular arguments by extracting the lexical patterns that characterize each class. The arguments formed clusters in the embedding space, indicating that each reasoning may be characterized by specific vocabularies. These patterns serve as a swift and understandable view into each argument. Additionally, we formulated methodological considerations regarding the nuance in the annotation scheme and linked our approach to research fields that struggle with moderating online content while safeguarding understanding among participants. The computational approach presented in this paper serves an assisting role to the human moderator, who in turn can deal with the contextual factors.

Acknowledgements

This publication is part of the project Better-MODS with project number 410.19.006 of the research programme 'Digital Society - The Informed Citizen' which is financed by the Dutch Research Council (NWO).

References

Aseel Addawood and Masooda Bashir. 2016. "What Is Your Evidence?" A Study of Controversial Topics on Social Media. *Proceedings of the Third Workshop on Argument Mining (ArgMining2016)*, pages 1–11.

Abeer ALDayel and Walid Magdy. 2021. Stance detection on social media: State of the art and trends. *Information Processing and Management*, 58(4):102597.

Joachim Allgaier. 2019. Science and Environmental Communication on YouTube: Strategically Distorted Communications in Online Videos on Climate Change and Climate Engineering. *Frontiers in Communication*, 4(July):1–15.

Vibeke Normann Andersen and Kasper M Hansen. 2007. How deliberation makes better citizens: The Danish Deliberative pool. *European Journal of Political Research*, 2007:531–556.

Reuben Binns, Michael Veale, Max Van Kleek, and Nigel Shadbolt. 2017. Like trainer, like bot? Inheritance of bias in algorithmic content moderation. *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, 10540 LNCS:405–415.

Richard J. Bord, Robert E. O'Connor, and Ann Fisher. 2000. In what sense does the public need to understand global climate change? *Public Understanding of Science*, 9(3):205–218.

Tom Bosc, Elena Cabrio, and Serena Villata. 2016. Tweeties Squabbling : Positive and Negative Results in Applying Argument Mining on Social Media. *Computational Models of Argument*, 0:21–32.

Keynyn Brysse, Naomi Oreskes, Jessica O'Reilly, and Michael Oppenheimer. 2013. Climate change prediction: Erring on the side of least drama? *Global Environmental Change*, 23(1):327–337.

Lucas Carstens and Francesca Toni. 2015. Towards relation based argumentation mining. In *Proceedings of the 2nd Workshop on Argumentation Mining*, pages 29–34, Denver, CO. Association for Computational Linguistics.

Tivadar Danka and Peter Horvath. modAL: A modular active learning framework for Python. Available on arXiv at `https://arxiv.org/abs/1805.00979`.

Pieter Delobelle, Thomas Winters, and Bettina Berendt. 2020. RobBERT: A Dutch RoBERTa-based language model. *Findings of the Association for Computational Linguistics Findings of ACL: EMNLP 2020*, pages 3255–3265.

Peter T. Doran and Maggie Kendall Zimmerman. 2009. Examining the scientific consensus on climate change. *Eos*, 90(3):22–23.

John W. Du Bois. 2007. The stance triangle. *Stancetaking in Discourse*, pages 139–182.

Riley E. Dunlap and Aaron M. McCright. 2012. Organized Climate Change Denial. *The Oxford Handbook of Climate Change and Society*, (January 2012).

Kimmo Grönlund, Kaisa Herne, and Maija Setälä. 2015. Does Enclave Deliberation Polarize Opinions? *Political Behavior*, 37(4):995–1020.

Hank C. Jenkins-Smith, Joseph T. Ripberger, Carol L. Silva, Deven E. Carlson, Kuhika Gupta, Nina Carlson, Ani Ter-Mkrtchyan, and Riley E. Dunlap. 2020. Partisan asymmetry in temporal stability of climate change beliefs. *Nature Climate Change*, 10(4):322–328.

Xiaoya Jiang, Min Hsin Su, Juwon Hwang, Ruixue Lian, Markus Brauer, Sunghak Kim, and Dhavan Shah. 2021. Polarization Over Vaccination: Ideological Differences in Twitter Expression About COVID-19 Vaccine Favorability and Specific Hesitancy Concerns. *Social Media and Society*, 7(3).

Dilek Küçük and Fazli Can. 2020. Stance detection: A survey. *ACM Comput. Surv.*, 53(1).

John Lawrence and Chris Reed. 2019. Argument mining: A survey. *Computational Linguistics*, 45(4):765–818.

Anthony Leiserowitz. 2007. Fighting climate change: Human solidarity in a divided world International Public Opinion, Perception, and Understanding of Global Climate Change. page 40.

Anthony Leiserowitz, Edward W. Maibach, Connie Roser-Renouf, Geoff Feinberg, and Peter Howe. 2010. Climate Change in the American Mind: Americans' Global Warming Beliefs and Attitudes in June 2010. *Yale Project on Climate Change Communication*, (June):1–9.

Ilya Loshchilov and Frank Hutter. 2019. Decoupled Weight Decay Regularization. *ICLR 2019*.

Tan Margaret. 1994. Establishing Mutual Understanding in Systems Design: An Empirical Study. *Journal of Management Information Systems*, 10(4):159–182.

V. Masson-Delmotte, P. Zhai, A. Pirani, S.L. Connors, C. Pean, Berger S., N. Caud, Y. Chen, L. Goldfarb, M.i. Gomis, M. Huang, K. Leitzell, E. Lonnoy, J.B.R. Matthews, T.K. Maycock, T. Waterfield, O. Yelekci, R. Yu, and B. Zhou. 2021. Ipcc, 2021: Climate change 2021: The physical science basis. contribution of working group i to the sixth assessment report of the intergovernmental panel on climate change. Technical report.

Aaron M. McCright and Riley E. Dunlap. 2003. Defeating Kyoto: The Conservative Movement's Impact on U.S. Climate Change Policy. *Social Problems*, 50(3):348–373.

Marie Francine Moens. 2018. Argumentation mining: How can a machine acquire common sense and world knowledge? *Argument and Computation*, 9(1):1–4.

Diana Mutz and Jeffrey Mondak. 2006. The workplace as a context for cross-cutting political discourse. *The Handbook of Discourse Analysis*, 68(1):398–415.

Nona Naderi and Graeme Hirst. 2016. Argumentation mining in parliamentary discourse. *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, 9935 LNAI:16–25.

Juri Opitz and Anette Frank. 2019. An argument-marker model for syntax-agnostic proto-role labeling. In *Proceedings of the Eighth Joint Conference on Lexical and Computational Semantics (*SEM 2019)*, pages 224–234, Minneapolis, Minnesota. Association for Computational Linguistics.

Naomi Oreskes. 2005. Scientific consensus on Climate Change. *Science*, 306(January):2004–2005.

F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. 2011. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830.

Wouter Poortinga, Alexa Spence, Lorraine Whitmarsh, Stuart Capstick, and Nick F. Pidgeon. 2011. Uncertain climate: An investigation into public scepticism about anthropogenic climate change. *Global Environmental Change*, 21(3):1015–1024.

Stephan Rahmstorf. 2003. The climate sceptics. *The state of science*.

Paul Rayson and Roger Garside. 2000. Comparing Corpora using Frequency Profiling. *WCC '00: Proceedings of the workshop on Comparing corpora*, pages 1–6.

Nils Reimers, Benjamin Schiller, Tilman Beck, Johannes Daxenberger, Christian Stab, and Iryna Gurevych. 2020. Classification and clustering of arguments with contextualized word embeddings. *ACL 2019 - 57th Annual Meeting of the Association for Computational Linguistics, Proceedings of the Conference*, pages 567–578.

Minna Ruckenstein and Linda Lisa Maria Turunen. 2020. Re-humanizing the platform: Content moderators and the logic of care. *New Media and Society*, 22(6):1026–1042.

Mattia Samory and Tanushree Mitra. 2018. 'The Government Spies Using Our Webcams'. *Proceedings of the ACM on Human-Computer Interaction*, 2(CSCW):1–24.

Ana Lucía Schmidt, Fabiana Zollo, Antonio Scala, Cornelia Betsch, and Walter Quattrociocchi. 2018. Polarization of the vaccination debate on Facebook. *Vaccine*, 36(25):3606–3612.

Christian Stab, Tristan Miller, and Iryna Gurevych. 2018. Cross-topic Argument Mining from Heterogeneous Sources Using Attention-based Neural Networks. *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*.

Kim Strandberg, Staffan Himmelroos, and Kimmo Grönlund. 2017. Do discussions in like-minded groups necessarily lead to more extreme opinions? Deliberative democracy and group polarization. *International Political Science Review*, 40(1):41–57.

Jane Suiter, David M. Farrell, and Eoin O'Malley. 2016. When do deliberative citizens change their opinions? Evidence from the Irish Citizens' Assembly. *International Political Science Review*, 37(2):198–212.

Stephen Toulmin. 2003. *The Uses of Argument*. Cambridge University Press.

Maarten van Gompel and Antal van den Bosch. 2016. Efficient n-gram, Skipgram and Flexgram Modelling with Colibri Core. *Journal of Open Research Software*, 4.

Sander L.Der Van Linden, Anthony A. Leiserowitz, Geoffrey D. Feinberg, and Edward W. Maibach. 2015. The scientific consensus on climate change as a gateway belief: Experimental evidence. *PLoS ONE*, 10(2):2–9.

Lorraine Whitmarsh. 2011. Scepticism and uncertainty about climate change: Dimensions, determinants and change over time. *Global Environmental Change*, 21(2):690–700.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Remi Louf, Morgan Funtowicz, Joe Davison, Sam Shleifer, Patrick von Platen, Clara Ma, Yacine Jernite, Julien Plu, Canwen Xu, Teven Le Scao, Sylvain Gugger, Mariama Drame, Quentin Lhoest, and Alexander Rush. 2020. Transformers: State-of-the-Art Natural Language Processing. *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 38–45.

Scott Wright and John Street. 2007. Democracy, Deliberation and Design: the case of online discussion forums. *New Media & Society*, 9(5).

Yue Zhao, Ciwen Xu, and Yongcun Cao. 2006. Research on Query-by-Committee Method of Active Learning and Application. *International Conference on Advanced Data Mining and Applications*, pages 985–991.

A Multi-turn Machine Reading Comprehension Framework with Rethink Mechanism for Emotion-Cause Pair Extraction

Changzhi Zhou, Dandan Song*, Jing Xu, Zhijing Wu
School of Computer Science and Technology,
Southeast Academy of Information Technology,
Beijing Institute of Technology, Beijing, China
zhou_changzhi97@163.com
sdd@bit.edu.cn, xujing@bit.edu.cn
wuzhijing.joyce@gmail.com

Abstract

Emotion-cause pair extraction (ECPE) is an emerging task in emotion cause analysis, which extracts potential emotion-cause pairs from an emotional document. Most recent studies use end-to-end methods to tackle the ECPE task. However, these methods either suffer from a label sparsity problem or fail to model complicated relations between emotions and causes. Furthermore, they all do not consider explicit semantic information of clauses. To this end, we transform the ECPE task into a document-level machine reading comprehension (MRC) task and propose a Multi-turn MRC framework with Rethink mechanism (MM-R). Our framework can model complicated relations between emotions and causes while avoiding generating the pairing matrix (the leading cause of the label sparsity problem). Besides, the multi-turn structure can fuse explicit semantic information flow between emotions and causes. Extensive experiments on the benchmark emotion cause corpus demonstrate the effectiveness of our proposed framework, which outperforms existing state-of-the-art methods.[1]

1 Introduction

Emotion cause extraction (ECE) is a classical emotion cause analysis task that aims to extract the corresponding causes of the given emotional expressions in an emotional document (Gui et al., 2016). However, the ECE task is not practical in real-world scenarios without annotated emotions. To overcome the limitation and capture mutual indications of emotions and causes together, Xia and Ding (2019) came up with a new task called emotion-cause pair extraction (ECPE), which aims to extract a potential emotion-cause pair set consisting of all emotions and their corresponding causes from a document. As shown in Figure 1(a), c2 is an emotion clause, c3 is the corresponding cause clause, and c4 is an emotion clause that is its own corresponding cause clause itself.

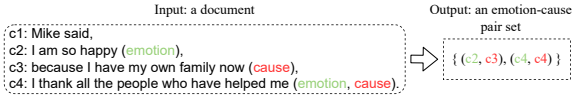

(a) An example of the ECPE task

(b) Pairing matrix

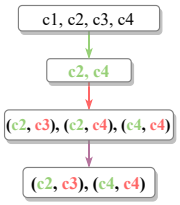

(c) Our approach

Figure 1: The green colour denotes an emotion clause, and the red colour denotes a cause clause. Figure (a) is an example of the ECPE task. Figure (b) is a pairing matrix generated by pair-level end-to-end approaches. Only (c2, c3) and (c4, c4) are valid pairs. Figure (c) shows the processing results by our MM-R in each turn.

In recent years, there has been a trend to use end-to-end methods to solve the ECPE task and the two sub-tasks, emotion extraction and cause extraction, which aim to extract all emotions and causes in one document. These end-to-end methods can be classified into two categories: pair-level methods by combining all clause pairs to form a clause pairing matrix; sequence labeling methods by designing novel tagging schemes. Specifically, some works (Ding et al., 2020a,b; Wei et al., 2020; Chen et al., 2020b; Wu et al., 2020) proposed pair-level methods that generate a pairing matrix by enumerating all possible combinations of clauses and then select valid emotion-cause pairs, as shown in Figure 1(b). However, the average ratio of invalid/valid pairs is more than 200:1 in the ECPE benchmark corpus (Xia and Ding, 2019), which

*Corresponding Author
[1]Data and code are available at https://github.com/zhoucz97/ECPE-MM-R

Proceedings of the 29th International Conference on Computational Linguistics, pages 6726–6735
October 12–17, 2022.

leads to a label sparsity problem. Besides, some works (Chen et al., 2020a; Yuan et al., 2020) proposed sequence labeling methods based on novel tagging schemes. However, they cannot effectively model the corresponding relation between emotions and causes. For example, Chen et al. (2020a) cannot deal with the situation where two emotion clauses of the same emotion type correspond to different cause clauses. Therefore, how to solve the label sparsity problem by avoiding the calculation of the pairing matrix while modeling the corresponding relation between emotions and causes is a crucial challenge. Furthermore, Zhong and Chen (2021) has shown that explicit semantic information can improve performance in the extraction tasks. Therefore, utilizing explicit semantic information of clauses to improve the performance in the ECPE task is also a challenge.

To address these challenges, in this study, the ECPE task is transformed into a document-level machine reading comprehension (MRC) task, and a Multi-turn MRC framework with Rethink mechanism (MM-R) is proposed. The multi-turn structure decomposes the ECPE task to model the corresponding relation between emotions and causes and avoid generating the pairing matrix. In every turn, static and dynamic queries designed manually make full use of explicit semantic information of clauses to improve the performance of emotion or cause extraction. In addition, inspired by the human two-stage reading behavior, in which search for possible answer candidates and then verify these candidates (Zheng et al., 2019), the rethink mechanism is proposed to verify candidate emotion-cause pairs further to enhance the flow of information between emotions and causes and improve the overall performance.

Specifically, in the first turn, all emotion clauses are extracted. In the second turn, cause clauses corresponding to each emotion clause are extracted with the assistance of explicit semantic information about emotions. Numerous experiments (Ding et al., 2020b; Fan et al., 2020; Ding et al., 2020a; Wei et al., 2020; Chen et al., 2020b,a; Yuan et al., 2020) have shown that emotion extraction is more reliable than cause extraction, and therefore it is reasonable to extract emotions first and then extract the corresponding causes with the assistance of emotions. Hence, a candidate emotion-cause pair set is obtained in the first two turns. In the third turn, a rethink mechanism verifies each candidate emotion-cause pair. For example, as shown in Figure 1(c), emotion clauses c2 and c4 can be obtained in the first turn, and cause clauses corresponding to each emotion clause can be obtained in the second turn. The candidate emotion-cause pair set {(c2, c3), (c2, c4), (c4, c4)} can thus be obtained without the pairing matrix. In the third turn, the rethink mechanism demonstrates that c2 is not one of the corresponding emotion clauses of c4. Therefore, the valid emotion-cause pair set becomes {(c2, c3), (c4, c4)}.

Comprehensive experiments are conducted on the ECPE benchmark datasets to verify the effectiveness of the proposed MM-R framework. The experimental results show that the proposed framework substantially outperforms the previous methods. The contributions of this research can be summarized as follows:

- The ECPE task is formalized as a document-level machine reading comprehension (MRC) task. To our best knowledge, this is the first time that the ECPE task has been transferred to the MRC task.
- Based on the MRC formalization, a multi-turn MRC framework with rethink mechanism (MM-R) is proposed. It models the corresponding relation between emotions and causes and alleviates the label sparsity problem. Furthermore, the MM-R can use explicit semantic information of clauses effectively.
- The experimental results demonstrate that the proposed framework outperforms existing state-of-the-art methods.

2 Related Work

2.1 Emotion-Cause Pair Extraction

Xia and Ding (2019) proposed the emotion-cause pair extraction (ECPE) task and released the Chinese benchmark dataset. Recently, many methods based on the pair-level end-to-end framework have been designed for the ECPE task. For example, Wei et al. (2020) obtained excellent results by modeling inter-clause relationships and using the ranking mechanism. Ding et al. (2020a) used a 2D-transformers module to model the interactions of different emotion-cause pairs. However, these pair-level end-to-end methods had a label sparsity problem caused by calculating the pairing matrix. Ding et al. (2020b) used a sliding window to restrict the amount of candidate emotion-cause pairs,

which aims to shrink the pairing matrix. However, the method still belonged to the pair-level end-to-end method because it assumed all clauses were emotion or cause clauses.

Sequence labeling was another popular method. However, these methods could not model the corresponding relation between emotions and causes. A unified tagging scheme (Chen et al., 2020a) could not be used on a document with multiple emotion clauses of the same emotion types because the tagging scheme was based on emotion types. Another joint tagging scheme (Yuan et al., 2020) based on the distance between the cause and the corresponding triggered emotion. The tagging scheme could not handle the situation in which the distance between the emotion clause and the corresponding cause clause exceeded the distance threshold. Therefore, the challenge remains of how to solve the label sparsity problem while modeling the corresponding relation between emotions and causes.

2.2 Machine Reading Comprehension

Recently, it became popular to transform many traditional natural language processing tasks into machine reading comprehension (MRC) tasks. For example, Liu et al. (2020) cast the event extraction as a machine reading comprehension problem to solve the data scarcity problem. Zhou et al. (2021) proposed a dual QA framework aiming at event argument extraction. Chen et al. (2021) and Mao et al. (2021) proposed a bidirectional MRC and a Dual-MRC frameworks to handle aspect-based sentiment analysis, respectively. In this paper, the ECPE task is formalized as a document-level MRC task, and the multi-turn MRC framework with rethink mechanism is proposed. It concatenates a query with each clause to explicitly utilize semantic information of clauses and uses a single classifier to predict whether a clause is an emotion or a cause clause in each turn.

3 Methodology

3.1 Problem Formulation

Given a document consisting of multiple clauses $D = (c_1, c_2, ..., c_{|D|})$ [2] and each clause contains multiple words $c_i = (w_{i,1}, w_{i,2}, ..., w_{i,|c_i|})$, our goal is to extract a set of emotion-cause pair in D:

$$P = \{(c_k^{e_i}, c_k^{ca_{i,j}})\}_{k=1}^{|P|}, \quad (1)$$

where the superscript e_i and $ca_{i,j}$ denote the i-th emotion clause and its corresponding j-th cause clause in D; the subscript k denotes the k-th emotion-cause pair in set P; an emotion clause corresponds to one or more cause clauses. [3]

3.2 Query Design

Static and dynamic queries are used to formalize the ECPE task to the MRC task. All queries can be formulated as follows:[4]

- **Static emotion query** $q^{se} \in Q^{se}$: The query "*Is it an emotion clause?*" is designed to extract all emotion clauses.
- **Static cause query** $q^{sc} \in Q^{sc}$: The query "*Is it a cause clause?*" is designed to extract all cause clauses.
- **Static pair query** $q^{sp} \in Q^{sp}$: The query "*Is it an emotion-cause pair?*" is designed to extract all emotion-cause pairs.
- **Dynamic emotion query** $q^{de} \in Q^{de}$: The query template "*Is it an emotion clause corresponding to c_i?*" is designed to extract emotion clauses corresponding to clause c_i.
- **Dynamic cause query** $q^{dc} \in Q^{dc}$: The query template "*Is it a cause clause corresponding to c_i?*" is designed to extract cause clauses corresponding to clause c_i.

3.3 Frameworks

The architecture of MM-R is illustrated on the left side of Figure 2. In the first turn, a static emotion query q^{se} is used to extract emotion clause set $E = \{c^{e_i}\}_{i=1}^{|E|}$. In the second turn, based on each extracted emotion clause $c^{e_i} \in E$, a dynamic cause query q^{dc} is constructed to obtain c^{e_i}'s corresponding cause clause set $C_i = \{c^{ca_{i,j}}\}_{j=1}^{|C_i|}$. In this way, the candidate emotion-cause pair set $P^{can} = \{(c_k^{e_i}, c_k^{ca_{i,j}})\}_{k=1}^{|P^{can}|}$ is obtained in the first two turns. To filter out invalid emotion-cause pairs in P^{can} further, for each cause clause $c^{ca_{i,j}}$, the framework rethinks whether the clause c^{e_i} is its corresponding emotion clause with the help of dynamic emotion query q^{de}. Finally, the valid

[2] The $|*|$ denotes the number of elements in the collection $*$.

[3] To avoid confusion, the "cause clause" is denoted to ca rather than c.

[4] Static cause query and static pair query are only used in two variants of our framework (MM-D and MRC-E2E); refer to the Experiments section for details.

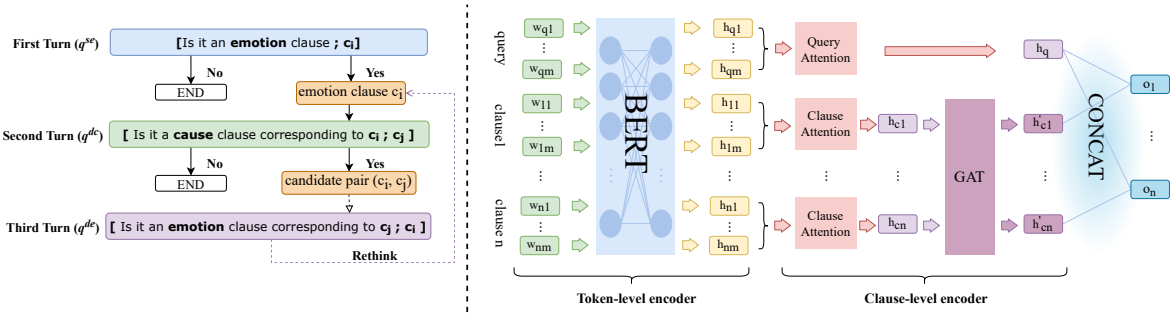

Figure 2: *Left*: The overall architecture of our MM-R framework. In each turn, the answer is yes if the probability output by the classifier is greater than 0.5, otherwise it is no. *Right*: The implementation structure of the encoding layer which includes the token-level encoder and the clause-level encoder. The token-level encoder generates the hidden representation of each token using the BERT module. The clause-level encoder provides the hidden representation of query and each clause using the attention mechanism and graph attention network. Finally, the concatenate operation (CONCAT) is executed on the hidden representations of queries and clauses.

emotion-cause pair set $P = \{(c_k^{e_i}, c_k^{ca_{i,j}})\}_{k=1}^{|P|}$ is obtained.

3.4 Encoding Layer

The structure of the encoding layer is illustrated on the right side of Figure 2. The encoding layer includes the token-level encoding layer and the clause-level encoding layer.

Token-level encoder: This study used BERT (Devlin et al., 2019) as its token-level contextualized encoder. Specifically, a document $D = (c_1, c_2, ..., c_{|D|})$ and a query q are used to construct the BERT input sequence:

$$I = \{[CLS], w_{q,1}, w_{q,2}, ..., w_{q,|q|}, [SEP], w_{1,1}, w_{1,2}, ..., w_{|D|,1}, ..., w_{|D|,|c_{|D|}|}\}, \quad (2)$$

where $q = q^{se}$ in the first turn, $q = q^{dc}$ in the second turn and $q = q^{de}$ in the third turn; $w_{q,j}$ is the j-th token of query q; $w_{i,j}$ is the j-th token of the i-th clause in the document D; and $[CLS]$, $[SEP]$ are special BERT tokens. The tokens can then be encoded into hidden representations:

$$\begin{aligned} H^I =& BERT(I) \\ =& \{h_{[CLS]}, h_{q,1}, h_{q,2}, ..., h_{q,|q|}, h_{[SEP]}, \\ & h_{1,1}, h_{1,2}, ..., h_{|D|,1}, ..., h_{|D|,|c_{|D|}|}\}, \end{aligned} \quad (3)$$

where $H^I \in \mathbb{R}^{|I| \times d}$; d is the dimension of the hidden states; and $h_{i,j}$ denotes the hidden representation of token $w_{i,j}$.

Clause-level encoder: The attention mechanism and the graph attention network (GAT) (Veličković et al., 2017) are used to obtain clause-level hidden representations.

Specifically, for clause c_i, its token-level representation set is selected from H^I:

$$S_{c_i} = \{h_{i,j}\}_{j=1}^{|c_i|} \in \mathbb{R}^{|c_i| \times d}. \quad (4)$$

The attention mechanism produces an attention weight vector α_i. The hidden representation of clause c_i is obtained by calculating the weighted sum of the hidden representations of all tokens:

$$\alpha_i = softmax(w^T S_{c_i} + b) \in \mathbb{R}^{1 \times |c_i|}, \quad (5)$$

$$h_{c_i} = sum(\alpha_i S_{c_i}) \in \mathbb{R}^{1 \times d}, \quad (6)$$

where w and b are learnable parameters. For all clauses, the attention mechanism is used to obtain the hidden representations:

$$H_C = \{h_{c_1}, h_{c_2}, ..., h_{c_{|D|}}\}. \quad (7)$$

Similarly, the attention mechanism is used to obtain the hidden representation of query q:

$$H_Q = \{h_q\}. \quad (8)$$

Furthermore, GAT models the interaction among clauses, and then the hidden representations of clauses are updated to:

$$H'_C = GAT(H_C) = \{h'_{c_1}, h'_{c_2}, ..., h'_{c_{|D|}}\}. \quad (9)$$

Finally, $h_q \in H_Q$ and $h'_{c_i} \in H'_C$ are concatenated to obtain $o_i = [h_q; h'_{c_i}]$, where $[;]$ denotes the concatenate operation. The output of the encoding layer is thus obtained:

$$O_{enc} = \{o_1, o_2, ..., o_{|D|}\}. \quad (10)$$

3.5 Answering Prediction

The output representation of the encoder layer $o_i = [h_q; h'_{c_i}]$ contains abundant semantic information, including identification information of emotion/cause, semantic information of clauses, and semantic information of the document. Therefore, only a single linear perception (SLP) is needed to obtain the answer to the query by predicting yes or no:

$$\hat{y}_i = \sigma(w_S^T o_i + b_S), \tag{11}$$

where w_S and b_S are learnable parameters of SLP; $\sigma(\cdot)$ is a logistic function; and $\hat{y}_i$ denotes the probability that the answer is yes. If $\hat{y}_i > 0.5$, the answer is judged to be yes, meaning that clause c_i is one of the answers to query q. This prediction style is similar to BoolQ (Clark et al., 2019).

3.6 Joint Training

A joint learning strategy was used in this study, the cross-entropy loss for each type of query was minimized as follows:

$$\mathcal{L}^* = -\sum_{i=1}^{N}\sum_{j=1}^{|D|}\sum_{k=1}^{|Q^*|}[p(y_{i,j,k}|c_{i,j},q_k^*)\log\hat{p}(y_{i,j,k}|c_{i,j},q_k^*)], \tag{12}$$

where $* \in \{se, dc, de\}$; N denotes the number of documents in the dataset; $c_{i,j}$ is the j-th clause of the i-th document; and q_k^* is the k-th query in Q^*.

Therefore, the final loss function $\mathcal{L}$ was as follows:

$$\mathcal{L} = \mathcal{L}^{se} + \mathcal{L}^{dc} + \mathcal{L}^{de}, \tag{13}$$

3.7 Inference

In the first turn, the $q^{se} \in Q^{se}$ first identifies the emotion clause set $E = \{c^{e_1}, c^{e_2}, ..., c^{e_{|E|}}\}$. In the second turn, for each predicted emotion clause c^{e_i}, the $q^{dc} \in Q^{dc}$ recognizes the corresponding cause clause set $C_i = \{c^{ca_{i,1}}, c^{ca_{i,2}}, ..., c^{ca_{i,|C_i|}}\}$ and obtains the set of candidate emotion-cause pairs $P^{can} = \{(c_k^{e_i}, c_k^{ca_{i,j}})\}_{k=1}^{|P^{can}|}$. And the probability of each candidate pair is $p(c^{e_i}, c^{ca_{i,j}}) = p(c^{e_i})p(c^{ca_{i,j}}|c^{e_i})$.

In the third turn, the $q^{de} \in Q^{de}$ is used for selecting valid emotion-cause pairs. Furthermore, the rethink mechanism is implemented through a soft selection strategy, in which the probability is adjusted and a probability threshold is used. Specifically, the probability of the candidate pair $(c^{e_i}, c^{ca_{i,j}})$ is updated by this strategy as follows:

$$p(c^{e_i}, c^{ca_{i,j}}) = \lambda p(c^{e_i})p(c^{ca_{i,j}}|c^{e_i}), \tag{14}$$

where the weight factor λ is used to adjust the probability of candidate emotion-cause pairs. λ is 1 when the predicted result of the third turn is yes, otherwise λ is a unique value between 0 and 1. The set of valid emotion-cause pairs is as follows:

$$P = \{(c^{e_i}, c^{ca_{i,j}})|(c^{e_i}, c^{ca_{i,j}}) \in P^{can}, p(c^{e_i}, c^{ca_{i,j}}) > \delta\}, \tag{15}$$

where δ is a probability threshold value.

4 Experiments

4.1 Dataset and Metrics

The benchmark dataset (Xia and Ding, 2019) was constructed based on a public Chinese emotion corpus (Gui et al., 2016) from the SINA NEWS website [5]. The dataset contains 1,945 documents and 28,727 clauses. The number of candidate clause pairs is 490,367 and the number of valid emotion-cause clause pairs is 2,167.

In the experiment, we use the two styles of data split:

- **10-fold cross validation**. Selecting 90% of the data for training and the remaining 10% for testing stochastically (as same as Xia and Ding (2019)). The most previous works use the data split style.

- **Training/Validation/Test data set**. Selecting 80% of the data for training, 10% of the data for validating and the remaining 10% for testing stochastically (as same as Fan et al. (2020)). The data split style is more plausible than the first style.

Furthermore, when we extract the emotion-cause pairs, we obtain the emotions and causes simultaneously. Thus, the performance of emotion extraction and cause extraction were also evaluated. The precision P, recall R and F1 score defined in (Gui et al., 2016; Xia and Ding, 2019) are used to evaluate the performance of the three tasks.

[5] https://news.sina.com.cn/

Model	E-C Pair Extraction			Emotion Extraction			Cause Extraction		
	P(%)	R(%)	F1(%)	P(%)	R(%)	F1(%)	P(%)	R(%)	F1(%)
SL-NTS	72.43	63.66	67.76	81.96	73.29	77.39	74.90	66.02	70.18
TransDGC (Val)	73.74	63.07	67.99	87.16	82.44	84.74	75.62	64.71	69.74
ECPE-2D	72.92	65.44	68.89	86.27	**92.21**	89.10	73.36	69.34	71.23
PairGCN	76.92	67.91	72.02	88.57	79.58	83.75	79.07	69.28	73.75
RANKCP	71.19	76.30	73.60	91.23	89.99	90.57	74.61	77.88	76.15
ECPE-MLL	77.00	72.35	74.52	86.08	91.91	88.86	73.82	79.12	76.30
MM-R	**82.18**	**79.27**	**80.62**	**97.38**	90.38	**93.70**	**83.28**	**79.64**	**81.35**
MM-R (Val)	78.97	75.32	77.06	96.09	88.09	91.88	80.90	76.21	78.45

Table 1: Performance of our models and baselines. P, R and F1 denote precision, recall and F1-measure respectively. E-C denotes Emotion-Cause. TransDGC(Val) and MM-R(Val) use the second data split style, the rest of models use the first data split style.

4.2 Experimental Settings

We used the BERT$_{base-Chinese}$ as our encoding backbone. During training, the AdamW optimizer with a weight decay of 0.01 was used for online learning, and the initial learning rate and warmup rate are set to 1e-5 and 0.1 respectively. The batch size is set to 2. As for regularization, dropout is applied for networks and the dropout rate is set to 0.1. We trained the model 20 epochs in total and adopted early stopping strategy. In the inference stage, $\lambda \in \{0.7, 1.0\}$ and the threshold δ was set to 0.5. The model was run on a Tesla V100 GPU.

4.3 Baselines

To demonstrate the effectiveness of our method, we compare our model with the following BERT baselines.

- **SL-NTS** (Yuan et al., 2020) regards the ECPE task as a sequence labeling problem and proposes a novel tagging scheme.
- **TransDGC** (Fan et al., 2020) proposes a transition-based model to transform the task into a parsing-like directed graph construction procedure. [6]
- **ECPE-2D** (Ding et al., 2020a) uses a 2D transformers to model the interactions of different emotion-cause pairs.
- **PairGCN** (Chen et al., 2020b) constructs a pair graph convolutional network to model dependency relations among local neighboring candidate pairs.
- **RANKCP** (Wei et al., 2020) uses a graph attention network to model interactions between clauses and selects emotion-cause pairs by the ranking mechanism.
- **ECPE-MLL** (Ding et al., 2020b) extracts emotion-cause pairs based on sliding window multi-label learning. It is a state-of-the-art model of baselines.

[6]It should be noted that the TransDGC method use the second data split style, while others methods use the first data split style.

4.4 Variants

To show the effectiveness of the multi-turn structure and the rethink mechanism in the proposed method, We designed the following variants:

- **MRC-E2E** is a single-turn **MRC** framework with an **E**nd-**t**o-**E**nd style. The static queries q^{sp} is used to extract an emotion-cause pair set.
- **MM** is a simple **M**ulti-turn **M**RC framework. Compared with MM-R, it removes the rethink mechanism.
- **MM-D** is proposed inspired by (Chen et al., 2021; Mao et al., 2021), which is a **M**ulti-turn **M**RC framework with **D**ual structure. One direction sequentially recognizes emotions and causes to obtain the candidate emotion-cause pair set with the help of static emotion query q^{se} and dynamic cause query q^{dc}. In contrast, the other direction identifies causes first and then emotions to obtain another candidate set

	Natural QL	Pseudo QL	Structured QL
Q^{se}	*Is it an emotion clause?*	*emotion?*	*emotion:_;cause:None*
Q^{dc}	*Is it a cause clause corresponding to* c_i*?*	c_i*;cause?*	*emotion:*c_i*;cause:_*
Q^{de}	*Is it an emotion clause corresponding to* c_i*?*	c_i*;emotion?*	*emotion:_;cause:*c_i
MM-R	80.62 (%F1)	80.51 (%F1)	79.72 (%F1)

Table 2: The performance of different query language designs (Natural, Pseudo and Structured QL) on ECPE task. "QL" denotes "Query Language". Q^{se}, Q^{dc} and Q^{de} are static emotion query, dynamic cause query and dynamic emotion query, respectively.

with the help of static cause query q^{sc} and dynamic emotion query q^{de}. Finally, we take the intersection of the two candidate sets.

4.5 Main Results

Table 1 gives the comparative results for the ECPE task and the two sub-tasks. The proposed MM-R shows a clear advantage over other baselines, obtaining F_1 improvements of 6.10%, 3.13%, and 5.05%, respectively, over the previous baselines on the three tasks.

Specifically, for the ECPE task, the MM-R obtains 5.18%, 6.92% and 6.10% improvements in the P, R, and $F1$ measures compared to ECPE-MLL. For the emotion extraction task, the MM-R performs 3.13% better than the best-performed baseline RANKCP, and it is worth noting that the increase in precision contributed most to the boost in the F_1 score. We believe that the high precision is due to the rethink mechanism filtering out some negative samples. The cause extraction task is more difficult because the cause clauses depend heavily on emotions. However, with the help of dynamic cause queries constructed by emotion semantic information, substantial increases (+5.05% F_1) are obtained compared to ECPE-MLL.

Besides, when we use the data split as the Trans-DGC (Fan et al., 2020), our method MM-R (Val) still obtains the state-of-the-art performance over the previous baselines.

5 Further Analysis

5.1 Effect of the Multi-turn Structure

MRC-E2E has the same label sparsity problem as pair-level end-to-end baselines, and it cannot model the interaction of emotions and causes due to its single-turn structure. However, MM, the most straightforward multi-turn framework among these similar methods, achieves 2.68%, 0.02%, and 2.84% F_1 improvements over MRC-E2E as shown in Table 3, which demonstrates the effectiveness of the multi-turn structure.

Model	Extraction of. (F1 %)		
	Emotion	Cause	E-C pair
MRC-E2E	90.34	77.92	75.35
MM	93.02	77.94	78.19
MM-D	93.67	79.47	78.76
MM-R	**93.70**	**81.35**	**80.62**

Table 3: Performance of variants.

In addition, compared with end-to-end baseline ECPE-MLL, MRC-E2E still achieves 1.48%, 1.62% and 0.83% improvements of F_1 in three tasks, which demonstrates that the explicit emotion/cause semantic information played an important role.

5.2 Effect of the Rethink Mechanism

To verify the advantages of the rethink mechanism, MM-R was compared with the other two models (MM and MM-D). Specifically, compared with MM, MM-R achieved 0.68%, 3.41%, and 2.43% F1 improvements on the three tasks, demonstrating the the rethink mechanism's effectiveness. Furthermore, MM-D achieved excellent performance with its dual structure. However, its score was 1.86% lower than MM-R in the ECPE task. The dual structure treats both emotion extraction and causes extraction equally, whereas the rethink mechanism is more trusting of emotion extraction. MM-R obtains a better performance due to emotion extraction being more reliable than cause extraction[7].

5.3 Effect of the Query Design

We explore the effect of different query language design on the model. As shown in Table 2, their F1

[7]We have illustrated in the introduction that emotion extraction is more reliable than cause extraction.

“过年了 (c1)，债主把家里粮食都搬走了 (c2)，别家都在欢欢喜喜过年 (c3)，而俺家连割肉的钱都没有 (c4)，我和母亲抱头痛哭 (c5)”，陈怀军说 (c6)。					
***Translate**: "It's the New Year (c1), and the creditor looted all food of my family (c2). Other families happily celebrate the New Year (c3), but we are too poor to buy meat (c4). This makes us very sad(c5)", said Huaijun Chen (c6).*					
The first turn	The second turn	Rethink	Threshold	Valid E-C pairs	Ground-truth
c3, c5	(c3, c4), $p_{(3,4)}$ = 0.5135 (c5, c4), $p_{(5,4)}$ = 0.6313	(c3, c4), $p_{(3,4)}$ = 0.3595 (c5, c4), $p_{(5,4)}$ = 0.6313	0.5	(c5, c4)	(c5, c4)

Figure 3: An example in the test set. The emotion clause set {c3, c5} was obtained in the first turn and the candidate emotion-cause pair set {(c3, c4), (c5, c4)} in the second turn. After using the rethink mechanism, the valid emotion-cause pair set was identified as {(c5, c4)}.

scores all exceeded 79% on the ECPE task. Concretely, Natural QL performs best in the ECPE task. Compared with Pseudo and Structured QL, Natural QL makes it easier for our model to understand the meaning of the queries, because BERT's pre-trained corpus are all natural languages. The Pseudo QL can be regarded as a concise expression of Natural QL. Hence our model can still understand the meaning of queries. Structured QL is the most abstract of three QLs, transforming the three queries ($Q^{se,dc,de}$) into a unified paradigm. It gains the lowest F1 score, which indicates that the inductive ability of our model is not enough to understand abstract structured query language.

5.4 Case Analysis

Figure 3 shows the inference process of MM-R and the advantage of the rethink mechanism through an example in the test set. The candidate emotion-cause pair set {(c3, c4), (c5, c4)} were obtained in the first two turns, and their probabilities were $p_{3,4} = 0.5135$ and $p_{5,4} = 0.6313$. After this, the proposed model adjusted the probabilities by rethinking whether each candidate emotion-cause pair was valid. The bottom of Figure 3 shows that the adjusted probabilities were $p_{3,4} = 0.3595$ and $p_{5,4} = 0.6313$, and the threshold value was 0.5. This means that the answer to the dynamic emotion query *“Is it an emotion clause corresponding to ‘but we are too poor to buy meat’?"* was no for clause $c3$ in the third turn. Therefore, candidate emotion-cause pair (c3, c4) was filtered out, while (c5, c4) was preserved.

5.5 Results for Emotion Cause Extraction

The proposed framework is also applied to the emotion cause extraction (ECE) task. Compared with the ECPE task, the ECE task has emotion annotations, and therefore in the MM-R framework, the emotion extraction of the first turn is omitted, and the emotion annotations are used to extract the corresponding cause clauses directly in the second turn. Finally, each candidate emotion-cause pair was verified by the rethink mechanism.

Methods	Emotion Cause Extraction		
	P(%)	R(%)	F1(%)
RTHN	76.97	76.62	76.77
KAG	79.12	75.81	77.43
RHNN	81.12	77.25	79.14
2-step RANKING	80.76	78.45	79.59
MM-R	**83.59**	**83.47**	**83.48**

Table 4: Results on the Emotion Cause Extraction task.

Comparisons are also conducted with recently proposed methods for the ECE task: RTHN (Xia et al., 2019), RHNN (Fan et al., 2019), KAG (Yan et al., 2021) and 2-step RANKING (Xu et al., 2021). Table 4 clearly demonstrates that our proposed framework achieves the state-of-the-art performance on the ECE task.

6 Conclusions

This study transforms the emotion-cause pair extraction (ECPE) task into the machine reading comprehension (MRC) task and proposes a multi-turn MRC with rethink mechanism (MM-R). This structure, which extracts emotions and causes in turn, avoids the label sparsity problem and models the complicated corresponding relations between emotions and causes. In every turn, explicit semantic information can be used effectively. Furthermore, the rethink mechanism verifies each candidate emotion-cause pair by modeling the flow of information from causes to emotions. Experimental results on the ECPE corpus demonstrated the effectiveness of the proposed model.

Acknowledgements

This work was supported by National Key Research and Development Program of China (Grant No. 2020YFC0833402), National Natural Science Foundation of China (Grant Nos. 61976021, U1811262), and Beijing Academy of Artificial Intelligence (BAAI).

References

Shaowei Chen, Yu Wang, Jie Liu, and Yuelin Wang. 2021. Bidirectional machine reading comprehension for aspect sentiment triplet extraction. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 35, pages 12666–12674.

Xinhong Chen, Qing Li, and Jianping Wang. 2020a. A unified sequence labeling model for emotion cause pair extraction. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 208–218.

Ying Chen, Wenjun Hou, Shoushan Li, Caicong Wu, and Xiaoqiang Zhang. 2020b. End-to-end emotion-cause pair extraction with graph convolutional network. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 198–207.

Christopher Clark, Kenton Lee, Ming-Wei Chang, Tom Kwiatkowski, Michael Collins, and Kristina Toutanova. 2019. Boolq: Exploring the surprising difficulty of natural yes/no questions. *arXiv preprint arXiv:1905.10044*.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. Bert: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186.

Zixiang Ding, Rui Xia, and Jianfei Yu. 2020a. Ecpe-2d: Emotion-cause pair extraction based on joint two-dimensional representation, interaction and prediction. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 3161–3170.

Zixiang Ding, Rui Xia, and Jianfei Yu. 2020b. End-to-end emotion-cause pair extraction based on sliding window multi-label learning. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 3574–3583.

Chuang Fan, Hongyu Yan, Jiachen Du, Lin Gui, Lidong Bing, Min Yang, Ruifeng Xu, and Ruibin Mao. 2019. A knowledge regularized hierarchical approach for emotion cause analysis. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 5614–5624.

Chuang Fan, Chaofa Yuan, Jiachen Du, Lin Gui, Min Yang, and Ruifeng Xu. 2020. Transition-based directed graph construction for emotion-cause pair extraction. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 3707–3717.

Lin Gui, Dongyin Wu, Ruifeng Xu, Qin Lu, and Yu Zhou. 2016. Event-driven emotion cause extraction with corpus construction. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 1639–1649.

Jian Liu, Yubo Chen, Kang Liu, Wei Bi, and Xiaojiang Liu. 2020. Event extraction as machine reading comprehension. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1641–1651.

Yue Mao, Yi Shen, Chao Yu, and Longjun Cai. 2021. A joint training dual-mrc framework for aspect based sentiment analysis. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 35, pages 13543–13551.

Petar Veličković, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Lio, and Yoshua Bengio. 2017. Graph attention networks. *arXiv preprint arXiv:1710.10903*.

Penghui Wei, Jiahao Zhao, and Wenji Mao. 2020. Effective inter-clause modeling for end-to-end emotion-cause pair extraction. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 3171–3181.

Sixing Wu, Fang Chen, Fangzhao Wu, Yongfeng Huang, and Xing Li. 2020. A multi-task learning neural network for emotion-cause pair extraction. In *ECAI 2020*, pages 2212–2219. IOS Press.

Rui Xia and Zixiang Ding. 2019. Emotion-cause pair extraction: A new task to emotion analysis in texts. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 1003–1012.

Rui Xia, Mengran Zhang, and Zixiang Ding. 2019. Rthn: A rnn-transformer hierarchical network for emotion cause extraction. *arXiv preprint arXiv:1906.01236*.

Bo Xu, Hongfei Lin, Yuan Lin, and Kan Xu. 2021. Two-stage supervised ranking for emotion cause extraction. *Knowledge-Based Systems*, page 107225.

Hanqi Yan, Lin Gui, Gabriele Pergola, and Yulan He. 2021. Position bias mitigation: A knowledge-aware graph model for emotion cause extraction. *arXiv preprint arXiv:2106.03518*.

Chaofa Yuan, Chuang Fan, Jianzhu Bao, and Ruifeng Xu. 2020. Emotion-cause pair extraction as sequence labeling based on a novel tagging scheme. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 3568–3573.

Yukun Zheng, Jiaxin Mao, Yiqun Liu, Zixin Ye, Min Zhang, and Shaoping Ma. 2019. Human behavior inspired machine reading comprehension. In *Proceedings of the 42nd International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 425–434.

Zexuan Zhong and Danqi Chen. 2021. A frustratingly easy approach for entity and relation extraction. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 50–61.

Yang Zhou, Yubo Chen, Jun Zhao, Yin Wu, Jiexin Xu, and Jinlong Li. 2021. What the role is vs. what plays the role: Semi-supervised event argument extraction via dual question answering. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 35, pages 14638–14646.

Structural Bias for Aspect Sentiment Triplet Extraction

Chen Zhang, Lei Ren, Fang Ma, Jingang Wang, Wei Wu, Dawei Song
Beijing Institute of Technology
{czhang,mfang,dwsong}@bit.edu.cn
Meituan NLP
{wangjingang02,wuwei30}@meituan.com
renlei_work@163.com

Abstract

Structural bias has recently been exploited for aspect sentiment triplet extraction (ASTE) and led to improved performance. On the other hand, it is recognized that explicitly incorporating structural bias would have a negative impact on efficiency, whereas pretrained language models (PLMs) can already capture implicit structures. Thus, a natural question arises: Is structural bias still a necessity in the context of PLMs? To answer the question, we propose to address the efficiency issues by using an adapter to integrate structural bias in the PLM and using a cheap-to-compute relative position structure in place of the syntactic dependency structure. Benchmarking evaluation is conducted on the SemEval datasets. The results show that our proposed structural adapter is beneficial to PLMs and achieves state-of-the-art performance over a range of strong baselines, yet with a light parameter demand and low latency. Meanwhile, we give rise to the concern that the current evaluation default with data of small scale is under-confident. Consequently, we release a large-scale dataset for ASTE. The results on the new dataset hint that the structural adapter is confidently effective and efficient to a large scale. Overall, we draw the conclusion that structural bias shall still be a necessity even with PLMs.[1]

1 Introduction

Aspect sentiment triplet extraction (ASTE) is a task central to fine-grained opinion mining. Compared to aspect sentiment classification that only aims to predict sentiment polarities for various aspects, ASTE instead extracts descriptive opinion units in the form of triplets (i.e., aspect-opinion-sentiment tuples). For example, (food, great, POS) and (service, dreadful, NEG) are aspect sentiment triplets for the sentence in Figure 1 (top), where {POS, NEG, NEU} respectively represent {positive, negative, neutral}.

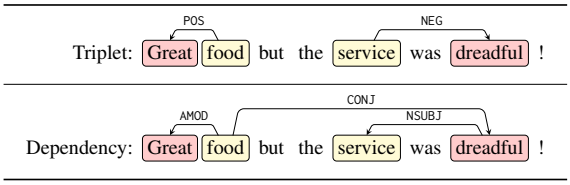

Figure 1: Triplet (top) and dependency structures (bottom) of an illustrative sentence. Spans shaded in yellow are aspects, spans shaded in red are opinions, and arcs indicate either sentiment or structural relations. Irrelevant structural relations are neglected for brevity.

While ASTE can be generally tackled with neural models in either a pipeline manner (Peng et al., 2020) or a multi-task manner (Xu et al., 2020; Zhang et al., 2020; Wu et al., 2020a; Chen et al., 2021a; Xu et al., 2021), the aspect sentiment triplets can be rather derivable from dependency structures (e.g., syntactic dependency trees) with hand-crafted rules (Wu et al., 2009; Sun et al., 2017). For the example in Figure 1 (bottom), the triplets can be recognized via certain structural dependency relations.[2] Various studies are motivated by this intuition and exploit dependency bias to enhance neural ASTE models (Chen et al., 2021b), yet without a necessary comparison with pretrained language models (PLMs). On the other hand, recent advances find that using PLMs can already achieve compelling performance (Yan et al., 2021; Zhang et al., 2021; Huang et al., 2021) owing to implicit structures captured by PLMs (Wu et al., 2020b). It signals that, compared with PLMs, explicit structural biases such as dependency bias, may become cumbersome (Dai et al., 2021) due to *parameter inefficiency* and *latency inefficiency*. That is, combining dependencies into models can

Jingang Wang and Dawei Song are the corresponding authors.

[1]Code and data are available at https://github.com/GeneZC/StructBias.

[2]Please see https://downloads.cs.stanford.edu/nlp/software/dependencies_manual.pdf for what these structural relations exactly stand for.

Proceedings of the 29th International Conference on Computational Linguistics, pages 6736–6745
October 12–17, 2022.

require redundant parameters to achieve structure encoding, while producing structures can also require increased latency to achieve external parsing. Therefore, a critical question naturally arises: Is structural bias still *a necessity* for ASTE in the context of PLMs?

In this paper, we aspire to answer the question from two perspectives: 1) whether structural bias can be incorporated into PLMs in a flexible way in terms of both parameter and latency efficiency; and 2) whether structural bias can enhance PLMs for ASTE.

To boost the parameter efficiency, we develop the idea of *adapter* and put forward a parameter-efficient adapter that can incorporate structural bias. The adapter (Houlsby et al., 2019) was proposed initially to integrate additional modules into PLMs and enable PLMs to leverage inductive bias efficiently. Although feasible, such adapters can be far from lightweight. For example, Liu et al. (2021) introduces a series of linear transformations in their proposed adapter which involve numerous parameters. In contrast, instead of introducing carefully-designed plugins, we propose to use structured attention maps induced with structures, to additively impact the raw attention maps in self-attention, thus requiring only a tiny amount of incremental parameters.

To improve the latency efficiency, we argue that dependency distance is a sufficient simplification of the dependency graph since the simplification has been proven equally powerful in downstream tasks like aspect sentiment classification (Zhang et al., 2019b). On this basis, we further propose to use *relative distance* as an alternative to the dependency distance. The intuition lies in the observation that opinions predominantly locate closely to their corresponding aspects (Xu et al., 2020; Ma et al., 2021), and thus we posit that using relative distance bias would suffice for the purpose of ASTE. In fact, the relative distance is also exhibited to bring merits to the transformer architectures in previous work (Shaw et al., 2018; Raffel et al., 2020). As the relative distance can be obtained with cheap operations in lower latency, the latency efficiency issue is thereby resolved.

We conduct a benchmarking comparative study on the SemEval datasets (Pontiki et al., 2014). The results show that models with the proposed structural adapter achieve the state-of-the-art (SOTA) performance compared with an array of strong baselines, indicating that incorporating structural bias is beneficial to PLMs. We also conduct a further study on how the relative distance-derived structural adapter overwhelms its alternatives. The results demonstrate that the structural adapter is an appealing choice. Specifically, our structural adapter realizes a 1,000× scale-down in terms of incremental parameters and a 1,000× speed-up of distance derivation.

In summary, structural bias can be flexibly incorporated into PLMs and improve both parameter and latency efficiency. The structural adapter is imposed with only a light parameter demand. The relative distance can be implemented with low latency. Moreover, the structural adapter vastly improves the SOTA performance. Therefore, structural bias is still be a necessity even in the context of PLMs to achieve a better ASTE performance.

Last but not least, in the view that current benchmarks are of small scales, we create a large-scale ASTE dataset termed 𝕃𝕠𝕤𝕥𝕖𝕕. 𝕃𝕠𝕤𝕥𝕖𝕕 is collected from one of the largest review platform in China, namely DianPing.[3] The dataset will be released to facilitate a more confident evaluation for ASTE and other possible research directions. The results on 𝕃𝕠𝕤𝕥𝕖𝕕 hint that structural adapter confidently improves the performance. Furthermore, compared with the results on the SemEval datasets, the model performance generally tends to be lower on this dataset, suggesting that the large-scale deployment of ASTE systems is still challenging.

2 Methodology

2.1 Task Formulation

Given a sequence of tokens $\{t_i\}_{i=1}^{n}$ as input, ASTE requires a model to output a set of triplets $\{(a, o, s)_i\}_{i=1}^{m}$, where a, o, s are the aspect, opinion, and sentiment, respectively. Concretely, an aspect a can be decomposed to two elements, i.e. (a_0, a_1), that separately denote the start and end positions. Likewise, an opinion o can be decomposed similarly.

2.2 PLM with Structural Adapter

When a PLM is employed as the backbone, the tokens are first transformed to embeddings, and then manipulated by subsequent transformer blocks. While the PLM can capture semantic interactions, we additionally present how to include structural interactions with a structural adapter.

[3]Please see https://www.dianping.com/.

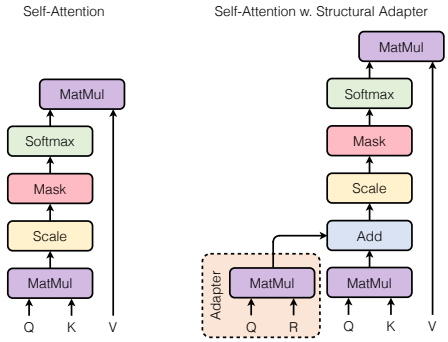

Figure 2: Differences between self-attention (left) and self-attention with the structural adapter applied (right). Q, K, and V separately stand for linear-transformed queries, keys, and values. R stands for distances.

2.2.1 Embedding

The tokens are generally augmented and encoded with the PLM. For example, if the PLM being used is a BERT (Devlin et al., 2019), the tokens should be augmented as:

$$[\text{CLS}] \quad t_1 \quad \dots \quad t_i \quad \dots \quad t_n \quad [\text{SEP}]$$

After that, the augmented tokens are converted to embeddings $\{\mathbf{t}_i\}_{i=0}^{n+1}$.

2.2.2 Transformer Block

The input embeddings are operated by the succeeding transformer blocks (Vaswani et al., 2017), each of which consists of a self-attention module and a feed-forward network module. The self-attention module is typically organized in a query-key-value formulation. Specifically, for any input $\{\mathbf{x}_i\}_{i=1}^{n}$, the output can be roughly written as:

$$\begin{aligned} \mathbf{z}_i &= \sum_{j=1}^{n} \alpha_{ij}(\mathbf{x}_j \mathbf{W}_V) \quad \alpha_{ij} = \text{softmax}_j(e_{ij}) \\ e_{ij} &= \frac{\mathbf{x}_i \mathbf{W}_Q (\mathbf{x}_j \mathbf{W}_K)^\top}{\sqrt{d}} \end{aligned} \tag{1}$$

Here, we omit special tokens and multiple heads for simplicity. The parameters $\mathbf{W}_Q$, $\mathbf{W}_K$ and $\mathbf{W}_V$ are learnable linear transformations for the query, key, and value. d is the head dimensionality.

2.2.3 Structural Adapter

In order to integrate the dependency or relative distance into the self-attention, the structural adapter is imposed to derive structured attention maps to bias the raw attention maps induced with the self-attention additively. The procedure is depicted as below:

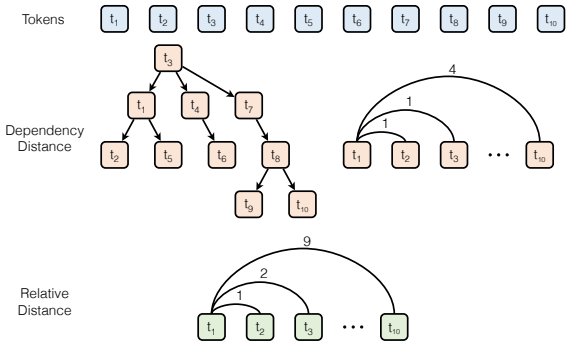

Figure 3: Derivation procedures of dependency distance (middle) and relative distance (bottom) given a sequence of tokens (top). Large distance values like 9 in this example may be clipped by the threshold τ. And distances will be made directional when employed, e.g., distance from t_1 to t_2 is set to 1 while that from t_2 to t_1 is actually set to -1.

$$\begin{aligned} e_{ij} &= \frac{\mathbf{x}_i \mathbf{W}_Q (\mathbf{x}_j \mathbf{W}_K + \mathbb{r}_{ij})^\top}{\sqrt{d}} \\ &= \underbrace{\frac{\mathbf{x}_i \mathbf{W}_Q (\mathbf{x}_j \mathbf{W}_K)^\top}{\sqrt{d}}}_{\text{raw attention map}} + \underbrace{\frac{\mathbf{x}_i \mathbf{W}_Q \mathbb{r}_{ij}^\top}{\sqrt{d}}}_{\text{structured attention map}} \end{aligned} \tag{2}$$

where $\mathbb{r}_{ij}$ indicates the distance embedding between two tokens t_i and t_j. It is also noteworthy that each relation embedding is shared across different heads, but kept independent from one layer to another layer. This behavior is inspired by Shaw et al. (2018), which is originally proposed to encode the relative positions but found to be applicable to encode arbitrary relations (Wang et al., 2020). The differences between self-attention and self-attention with the structural adapter applied are shown in Figure 2.

2.2.4 Distance Derivation

Specifically, the dependency distance between two tokens is obtained by computing the shortest distance on the dependency graph with the networkx toolkit,[4] and the dependency graph is produced with an off-the-shelf dependency parser stanza (Qi et al., 2020).[5] The relative distance between two tokens can be yielded by enumerating the number of tokens lying in-between.

[4] Please see https://networkx.org/ for more information.

[5] Please see https://stanfordnlp.github.io/stanza/ for more information.

We follow the de facto implementation that treats the distance from t_i to t_j different from that from t_j to t_i. We assign one to positive and the other to negative (Raffel et al., 2020). We also manually set a distance threshold that denotes the maximum distance τ. In doing so, we intend to avoid introducing too many parameters while maintaining as much information as possible. Henceforth, we will refer to the structural adapter with the dependency distance and relative distance as StructApt-Dep and StructApt-Rel. The derivation of both dependency distance and relative distance is illustrated in Figure 3.

2.3 Triplet Parser

Learning from two multi-task triplet parsing architectures MTL (Zhang et al., 2020) and GTS (Wu et al., 2020a), we establish a triplet parser that comprises two independent taggers (i.e., one for aspect and the other for opinion tagging), a sentiment scorer, and a triplet decoder. Conceptually, the two taggers are used to uncover continuous tokens that form an aspect or opinion span. The sentiment scorer is used to determine the token-level sentiment relation (if there is one) between two candidate tokens. Moreover, the triplet decoder produces triplets by gathering the information from the taggers and the sentiment scorer.

2.3.1 Aspect and Opinion Taggers

Following MTL, the taggers generate aspect and opinion tags in {B,I,O} format, after which the aspect and opinion spans are inferred with {B,I,O} tags. Presuming the hidden states of the PLM are $\{\mathbf{h}_i\}_{i=1}^{n}$ in spite of augmented tokens, the taggers are depicted as:

$$\begin{aligned}
\mathbf{r}_i^{(a)} &= \text{ReLU}(\mathbf{W}_1^{(a)}\mathbf{h}_i + \mathbf{b}_1^{(a)}) \\
\mathbf{y}_i^{(a)} &= \text{softmax}(\mathbf{W}_2^{(a)}\mathbf{r}_i^{(a)} + \mathbf{b}_2^{(a)}) \\
\mathbf{r}_i^{(o)} &= \text{ReLU}(\mathbf{W}_1^{(o)}\mathbf{h}_i + \mathbf{b}_1^{(o)}) \\
\mathbf{y}_i^{(o)} &= \text{softmax}(\mathbf{W}_2^{(o)}\mathbf{r}_i^{(o)} + \mathbf{b}_2^{(o)})
\end{aligned} \tag{3}$$

where $\{\mathbf{W}_1^{(a)}, \mathbf{b}_1^{(a)}, \mathbf{W}_2^{(a)}, \mathbf{b}_2^{(a)}\}$, $\{\mathbf{W}_1^{(o)}, \mathbf{b}_1^{(o)}, \mathbf{W}_2^{(o)}, \mathbf{b}_2^{(o)}\}$ are two sets of weights and biases for two feed-forward networks customized to aspect and opinion tagging.

2.3.2 Sentiment Scorer

The sentiment scorer produces token-level sentiment relations among all tokens. In addition to {POS, NEG, NEU}, there is also a NONE relation to account for the case of no relation. Unlike the sentiment scorer in MTL and GTS that only predicts uni-directional sentiment relations, we present a sentiment scorer that predicts bi-directional sentiment relations. The uni-directional relation means: a sentiment relation between an aspect token and an opinion token is always directed from the aspect token to the opinion token. In contrast, the bi-directional means: a sentiment relation is both directed from the aspect token to the opinion token and directed from the opinion token to the aspect token. This behavior allows more information to be transduced to the subsequent triplet decoding process to alleviate potential errors. Similarly, the sentiment scorer can be described as:

$$\begin{aligned}
\mathbf{r}_i^{(h)} &= \text{ReLU}(\mathbf{W}_1^{(h)}\mathbf{h}_i + \mathbf{b}_1^{(h)}) \\
\mathbf{r}_i^{(d)} &= \text{ReLU}(\mathbf{W}_1^{(d)}\mathbf{h}_i + \mathbf{b}_1^{(d)}) \\
\mathbf{y}_{i,j}^{(s)} &= \text{softmax}(\mathbf{r}_i^{(h)\top}\mathbf{W}_2^{(s)}\mathbf{r}_j^{(d)} + \\
&\quad \mathbf{W}_2^{(h)}\mathbf{r}_i^{(h)} + \mathbf{W}_2^{(d)}\mathbf{r}_j^{(d)} + \mathbf{b}_2^{(s)})
\end{aligned} \tag{4}$$

Here, $\{\mathbf{W}_1^{(h)}, \mathbf{b}_1^{(h)}\}$,$\{\mathbf{W}_1^{(d)}, \mathbf{b}_1^{(d)}\}$ are weights and biases separately for two feed-forward networks yielding head and dependent representations. These head and dependent representations are then organized in a biaffine manner (Dozat and Manning, 2017), where $\{\mathbf{W}_2^{(h)}, \mathbf{b}_2^{(h)}, \mathbf{W}_2^{(d)}, \mathbf{b}_2^{(d)}\}$ are weights and biases. The biaffine module predicts both aspect-to-opinion and opinion-to-aspect relations at the same time, since either aspects or opinions can be heads or dependents interchangeably.

Additionally, $\mathbf{y}^{(s)}$ refers to a sentiment probability map where $\mathbf{y}_{i,j}^{(s)}$ indicates a probability over 4 sentiment relations from the i-th token to the j-th one. If we apply the argmax operation on the probability map, we get a sentiment relation map. An example of the sentiment relation map is given in Figure 4 (left).

2.3.3 Triplet Decoder

Span-level sentiment relations are viable by searching for the most frequent sentiment relation in the set of indexed sentiment relations. Assume there are 2 tokens in a predicted target span and 1 token in a predicted opinion span, then we say there are 2*(2*1)=4 indexed sentiment relations between the two spans with the bi-directional interplay. Finding the most frequent sentiment relation in inclusive sentiment relations produced with the sentiment relation map gives the sentiment relation between

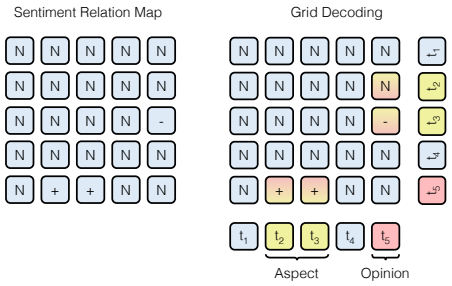

Figure 4: Sentiment relation map (left) and grid decoding algorithm (right). N, +, - respectively are short for NONE, POS, NEG relations. Gradient colors indicate either aspect-to-opinion relations or opinion-to-aspect relations within the bi-directional interplay. The grid decoding algorithm refers to tagged aspects and opinions to index the sentiment relation map.

the two spans. The algorithm is detailed in Figure 4 (right). As the bi-directional interplay is considered, the potential error (i.e., producing a NONE or NEG relation instead of a POS one.) in the example is alleviated.

Since the whole multi-task learning framework is generally borrowed from MTL while the triplet decoding strategy is adapted from the grid decoding in GTS, we name the proposed model as Multi-task learning with Grid decoding (MuG).

2.4 Finetuning

The PLM, the structural adapter, and the triplet parser can be jointly optimized by minimizing an overall objective that contains two sources of losses, i.e., tagging loss $\mathcal{L}_{\text{t}}$ and parsing loss $\mathcal{L}_{\text{p}}$. Both losses can be measured by the cross-entropy functions. The joint objective is formulated as follows:

$$\min_{\theta} \mathcal{L} = \min_{\theta} \mathcal{L}_{\text{t}} + \mathcal{L}_{\text{p}} \tag{5}$$

where θ stands for all parameters, which might be disassembled to $\{\theta_{\text{PLM}}, \theta_{\text{Adapter}}, \theta_{\text{Parser}}\}$.

3 Benchmarking Evaluation

3.1 Data

We conduct a comparative study on 4 benchmarking datasets from **SemEval** 2014, 2015, and 2016 (Pontiki et al., 2014), in which one contains data from laptop domain (**L**) and the other three contain data from restaurant domain (**R**). The triplet annotations are obtained from Xu et al. (2020). The statistics of these datasets are displayed in Appendix A.

3.2 Models

We compare a wide range of baseline models with varying backbones (e.g., BiLSTM, BERT) and different paradigms (i.e., extractive vs. generative). We list these baselines according to their paradigms as below:

Extractive Paradigm

- KWHW (Peng et al., 2020) is a pipeline system that first extracts aspect-sentiment pairs and opinions, and then pairs them in a binary manner.
- JET^{o} (Xu et al., 2020) is a position-aware sequence tagging system that jointly extracts triplets.
- MTL (Zhang et al., 2020) is a multi-task learning system which realizes aspect and opinion extraction with tagging while sentiment relation extraction with parsing.
- GTS (Wu et al., 2020a) transforms the triplet extraction problem as a grid tagging problem and achieves the extraction via a grid decoding algorithm.
- Span (Xu et al., 2021) is a span-level triplet extraction system that learns span-level interactions for a more accurate triplet prediction.

Generative Paradigm

- UGF (Yan et al., 2021) is a unified generative system based on BART (Lewis et al., 2020) for all sub-tasks in aspect sentiment analysis.
- GAS (Zhang et al., 2021) likewise is built upon T5 (Raffel et al., 2020) where extractive constraints are applied to the decoding space.

On another note, we have some variants in the comparative study to facilitate the understanding of our adapter. To examine the broad applicability of our structural adapter, we additionally test the structural adapter through the lens of the SOTA extractive systems, namely GTS and Span. To conduct a fair comparison, we initiate the PLM in our model not only with BERT (Devlin et al., 2019) but also with RoBERTa (Liu et al., 2019) to see whether our model with the adapter is competitive with those enhanced by advanced generative PLMs.

3.3 Implementation and Metrics

Typically, the adapter-based finetuning only tunes θ_{Adapter} and θ_{Parser}, and freezes θ_{PLM}. As a randomly initialized adapter can be an unsteady factor

to the PLM, standard finetuning (i.e., tuning all parameters) can result in performance with high variance (Houlsby et al., 2019). However, sub-optimal phenomenon has been observed in the literature (Liu et al., 2021) that such adapter-based fine-tuning is less promising than standard fine-tuning if the adapter is intended to integrate discrete information (e.g., structural information in our case). Thus, we adapt θ_{Adapter} and θ_{PLM} to the concerned task via finetuning all parameters (i.e., θ_{PLM}, θ_{Adapter}, and θ_{Parser}). Other implementation details are listed in Appendix B.

Following the common practice in the area, we adopt the exact match precision, recall, and F1 scores as the evaluation metrics. Namely, only when the corresponding elements from two triplets exactly match each other, will it be counted as one match. Further, to gain a robust evaluation, we average values over 10 runs and employ the mean value as the final number.

3.4 Performance Analysis

From the results presented in Table 1, we discover two key findings. The first is that the structural adapter incorporated with the relative distance can primarily improve performance across different models and different PLMs, though the improvements over RoBERTa are not as consistent as those over BERT on different datasets. The second is that the previous SOTA models are further boosted by the structural adapter and yield new SOTA results. These findings generally indicate the *effectiveness*, thus *necessity*, of the structural adapter. Conversely, the dependency distance is prone to parsing errors and sometimes underperforms the relative distance.

Moreover, we surprisingly observe that MuG with RoBERTa is a relatively strong baseline even compared with those remarkable generative ASTE models. Concretely, MuG with RoBERTa approximates or outperforms GAS with T5 in terms of F1 scores. This phenomenon encourages some retrospectives on whether generative ASTE models are superior to extractive ones, or the superiority is resulted by the generative PLM.

It can be arguable that the improvements of the structural adapter are marginal; however, we conjecture the inherent reason is that the data for evaluation is of small scale. According to the aforementioned unstable behavior of the adapter when encountering small-scale data in Section 3.3, we think the evaluation is under-confident and therefore conduct a large-scale evaluation in Section 4 to verify the guess and to get more confident results.

3.5 Parameter Analysis

We examine the gap between the structural adapter and structural layer (with dependency distance or relative distance, referred to as StructLyr-Dep and StructLyr-Rel respectively) from the perspective of incremental parameter scale. The structural layer is exactly a stack of additional transformer layers built upon the PLM, each of which is applied with the structural adapter. The best number of stacked layers is 2 in our pilot study.

The incremental parameters in Table 2 mean that additional parameters are brought to MuG. The structural adapter achieves 1,000× scale-down without performance loss compared with the structural layer. Contrarily, it seems that structural layers risk the model on the under-fitting issue due to over-parameterization and get degraded performance compared with MuG. We hereby argue that *parameter efficiency* of the structural adapter is permissible.

3.6 Latency Analysis

To better understand the difference between latency consumed by dependency distance derivation and relative distance derivation. We test the latency caused by the above two derivation procedures.

While dependency distance derivation costs around 4 micro-seconds per token (250 tokens/ms in other words), relative distance derivation only spends 3e-3 micro-seconds per token (333,000 tokens/ms in other words). That is, the relative distance derivation enjoys a 1,000× speed-up compared with the dependency distance derivation. Hence, the *latency efficiency* of relative distance derivation is numerically verified.

4 Large-scale Evaluation

4.1 Data

Being aware that the above benchmarking evaluation may be under-confident considering that the data is of small scale, we release a large-scale ASTE dataset, short-named Lasted. The data is collected from one of the largest review platform in China, namely DianPing. After necessary preprocessing steps, these reviews are manually annotated by 10 proficient assessors. For sanity, double-check on these annotations is carried out by a researcher who has devoted herself to the area for

Model	L14			R14			R15			R16		
	P	R	F_1	P	R	F_1	P	R	F_1	P	R	F_1
KWHW BiLSTM*	37.38	50.38	42.87	43.24	63.66	51.46	48.07	57.51	52.32	46.96	64.24	54.21
JET^o BiLSTM*	53.03	33.89	41.35	61.50	55.13	58.14	64.37	44.33	52.50	70.94	57.00	**63.21**
MTL BiLSTM‡	51.00	40.07	44.81	63.87	54.76	58.90	57.50	42.56	48.73	59.03	54.84	56.73
GTS BiLSTM‡	60.32	38.98	**47.25**	71.08	56.38	**62.85**	66.60	46.91	**55.02**	68.75	56.02	61.71
JET^o BERT*	55.39	47.33	51.04	70.56	55.94	62.40	64.45	51.96	57.53	70.42	58.37	63.83
GTS BERT‡	57.09	50.33	53.48	69.49	67.75	68.59	61.59	58.21	59.81	65.75	68.32	66.99
w/ StructApt-Rel	57.89	51.57	54.47	68.94	68.26	68.60	62.17	58.63	60.28	66.17	69.79	67.91
Span BERT‡	62.57	56.02	59.08	71.77	70.42	71.06	62.06	63.26	62.63	68.57	71.12	69.79
w/ StructApt-Rel	64.72	56.80	**60.47**	72.53	71.75	**72.13**	62.80	63.79	**63.17**	68.94	70.74	**69.80**
MuG BERT	58.30	52.21	55.06	68.40	67.64	68.00	60.65	54.12	57.10	66.26	67.39	66.74
w/ StructApt-Dep	59.39	52.95	55.95	67.69	68.90	68.27	60.74	55.77	58.11	64.73	68.33	66.45
w/ StructApt-Rel	59.54	52.56	55.75	68.92	68.12	68.50	59.83	56.78	58.17	65.31	68.83	67.01
UGF BART†	61.41	56.19	58.69	65.52	64.99	65.25	59.14	59.38	59.26	66.60	68.68	67.62
GAS T5†	–	–	**60.78**	–	–	**72.16**	–	–	62.10	–	–	70.10
MuG RoBERTa	64.18	57.03	60.33	70.47	71.88	71.16	63.78	61.88	62.79	68.61	72.20	70.34
w/ StructApt-Dep	64.18	56.41	60.03	71.62	71.92	71.72	63.96	61.67	62.70	68.85	71.81	70.28
w/ StructApt-Rel	64.12	57.16	60.53	73.26	71.93	71.17	62.86	63.82	**63.12**	69.15	74.12	**70.44**

Table 1: Benchmarking evaluation results. The marker * indicates results of the model are cited from Xu et al. (2020). The marker † indicates results of the model are cited from its original paper. The marker ‡ indicates results of the model are reproduced from its released code. Results are replaced by – to indicate they are not available. F1 scores are underlined to indicate they outperform their adapter-ablated counterparts. F1 scores are **boldfaced** to indicate they are the best-performing ones in their areas.

Model	#Params+	L14	R14
MuG BERT	0.00 M	55.06	68.00
w/ StructLyr-Dep	14.17 M	52.52	67.03
w/ StructLyr-Rel	14.17 M	51.57	67.04
w/ StructApt-Dep	0.01 M	55.95	68.27
w/ StructApt-Rel	0.01 M	55.75	68.50

Table 2: Parameter comparison. F1 scores are reported. #Params+ is short for number of incremental parameters.

Dataset		#S	#T	#T/S	#Tk/S
SemEval R14	train	1266	2336	1.85	17.31
	dev	310	577	1.86	15.81
	test	492	994	2.02	16.34
Lasted	train	19485	38050	1.95	34.94
	dev	2783	5334	1.92	34.88
	test	5567	10820	1.94	35.04

Table 3: Statistics of Lasted, with a comparison with **SemEval R14**. #S denotes number of sentences, #T denotes number of triplets, #T/S denotes average number of triplets per sentence, and #Tk/S denotes average number of tokens per sentence.

years.

For clarity, the pre-processing steps include: 1) removing user identities for privacy consideration; 2) chunking the reviews to shorter examples as they are generally too long (e.g., longer than 512); 3) tokenizing these examples; 4) removing examples without annotations, with less than 4 tokens, or with more than 128 tokens; 5) removing triplets in an example if the triplet has more than 8 tokens in the aspect or has more than 16 tokens in the target, for a too long aspect or opinion indicates the triplet may be not well annotated. Ultimately, these examples are formatted in the format we mentioned in Section 2.1.

We attain the dataset with a total of 27,835 examples. We uniformly split it into train, development, and test sets with a ratio of 7: 1: 2. The statistics are shown in Table 3, where we also include **SemEval R14** for comparison purpose. From the statistics, we can summarize that Lasted is a much larger dataset with longer sentences, which sets a more challenging benchmark for models to achieve a high performance.

4.2 Models

We conduct experiments based on GTS and MuG. While we only test GTS with `BERT-base`, we further test MuG with `BERT-base`, `RoBERTa-base`, and tentatively with `RoBERTa-large`. As we

Model	Lasted		
	P	R	F_1
GTS BERT-base	43.81	46.11	44.92
w/ StructApt-Rel	45.38	46.22	45.79
MuG BERT-base	47.20	45.28	46.22
w/ StructApt-Rel	49.64	45.02	47.22
MuG RoBERTa-base	48.10	44.98	46.49
w/ StructApt-Rel	50.40	44.77	47.42
MuG RoBERTa-large	49.49	46.85	48.13
w/ StructApt-Rel	48.33	47.91	48.13

Table 4: Large-scale evaluation results on Lasted. F1 scores are underlined to indicate they outperform their adapter-ablated counterparts.

know that only BERT-base is officially released by Devlin et al. (2019) for Chinese, we retrieve RoBERTa-base and RoBERTa-large released by Cui et al. (2019) on Hugging Face.[6]

4.3 Implementation and Metrics

The implementation and metrics strictly follow those used in the benchmarking evaluation, with exceptions listed in Appendix B.

4.4 Analysis

We can see from Table 4 that the adapter is still promising under large-scale evaluation. With the notice that the evaluation results should be more confident, we hence can safely conclude that the structural adapter is effective and structural bias is *a necessity* for ASTE even in the context of PLMs. However, the metrics on Lasted are consistently lower than expected, implying the deployment of ASTE systems is still challenging. The structural adapter does not improve RoBERTa-large, we leave the question of how to combine it with large PLMs for future work.

5 Related Work

5.1 Aspect Sentiment Triplet Extraction

Aspect sentiment triplet extraction is a recently proposed task to extract aspects, opinions, and sentiment relations (Peng et al., 2020), serving as a complete solution to aspect sentiment analysis (Zhang et al., 2019a; Ma et al., 2022). While the first-ever work delving into the task takes a pipeline system, succeeding work shifts their attention from pipeline models to joint models. Zhang et al. (2020) and Wu et al. (2020a) share similar spirits to treat three sub-tasks in a multi-task manner. Specifically, Wu et al. (2020a) proposes to consider the extraction of three elements in a unified grid tagging scheme. Later studies exploit inductive biases such as span-level interactions (Xu et al., 2021) and structural bias (Chen et al., 2021b). To our surprise, none of them inspects whether inductive biases, particularly structural bias that is shown beneficial in other PLM-enhanced tasks (Wang et al., 2022), are significant for PLM-enhanced ASTE models. Our work seeks to answer this question by putting forward a flexible adapter and checking whether the adapter is a necessity.

5.2 Adapter for PLM

An adapter is an emergent concept which means an efficient module injected into the PLM so that the PLM can better adapt to downstream tasks (Houlsby et al., 2019). Applications including speed translation (Le et al., 2021), language transfer (He et al., 2021), etc. have been witnessed. Traditionally, parameters of the PLM should not be tuned during fine-tuning once the adapter is armed. Nevertheless, recent work (Liu et al., 2021) finds that when injecting discrete information, unfreezing the parameters of the PLM will bring further performance gain. While previous adapters are modules and thus far from truly lightweight, we propose to leverage the structured attention as a sort of adapter, which is lightweight.

6 Conclusion

In this paper, we are concerned about the parameter and latency inefficiency issues of incorporating structural bias to PLMs for aspect sentiment triplet extraction, and raise the question on whether structural bias is a necessity. To answer the question, we propose to use an adapter to integrate the relative position structure into PLMs for a light parameter demand compared with incremental layers and low latency compared with the syntactic dependency structure. We carry out benchmarking experiments on SemEval benchmarks and large-scale experiments on our newly released Lasted dataset as a supplementary. The results in two rounds of evaluations show that the structural adapter is an appealing choice regarding its effectiveness, parameter efficiency, and latency efficiency, implying the structural bias, in the form of the structural adapter,

[6]Please see https://huggingface.co/hfl/chinese-roberta-wwm-ext for more information.

is a necessity even with PLMs.

Acknowledgements

This research was supported in part by Natural Science Foundation of Beijing (grant number: 4222036) and Huawei Technologies (grant number: TC20201228005).

References

Shaowei Chen, Yu Wang, Jie Liu, and Yuelin Wang. 2021a. Bidirectional machine reading comprehension for aspect sentiment triplet extraction. In *AAAI*, pages 12666–12674.

Zhexue Chen, Hong Huang, Bang Liu, Xuanhua Shi, and Hai Jin. 2021b. Semantic and syntactic enhanced aspect sentiment triplet extraction. In *ACL*, pages 1474–1483.

Yiming Cui, Wanxiang Che, Ting Liu, Bing Qin, Ziqing Yang, Shijin Wang, and Guoping Hu. 2019. Pre-training with whole word masking for Chinese BERT. *arXiv*, abs/1906.08101.

Junqi Dai, Hang Yan, Tianxiang Sun, Pengfei Liu, and Xipeng Qiu. 2021. Does syntax matter? A strong baseline for aspect-based sentiment analysis with roberta. In *NAACL*, pages 1816–1829.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: pre-training of deep bidirectional transformers for language understanding. In *NAACL*, pages 4171–4186.

Timothy Dozat and Christopher D. Manning. 2017. Deep biaffine attention for neural dependency parsing. In *ICLR*.

Ruidan He, Linlin Liu, Hai Ye, Qingyu Tan, Bosheng Ding, Liying Cheng, Jia-Wei Low, Lidong Bing, and Luo Si. 2021. On the effectiveness of adapter-based tuning for pretrained language model adaptation. In *ACL*, pages 2208–2222.

Neil Houlsby, Andrei Giurgiu, Stanislaw Jastrzebski, Bruna Morrone, Quentin de Laroussilhe, Andrea Gesmundo, Mona Attariyan, and Sylvain Gelly. 2019. Parameter-efficient transfer learning for NLP. In *ICML*, pages 2790–2799.

Lianzhe Huang, Peiyi Wang, Sujian Li, Tianyu Liu, Xiaodong Zhang, Zhicong Cheng, Dawei Yin, and Houfeng Wang. 2021. First target and opinion then polarity: Enhancing target-opinion correlation for aspect sentiment triplet extraction. *arXiv*, abs/2102.08549.

Hang Le, Juan Miguel Pino, Changhan Wang, Jiatao Gu, Didier Schwab, and Laurent Besacier. 2021. Lightweight adapter tuning for multilingual speech translation. In *ACL*, pages 817–824.

Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Veselin Stoyanov, and Luke Zettlemoyer. 2020. BART: denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. In *ACL*, pages 7871–7880.

Wei Liu, Xiyan Fu, Yue Zhang, and Wenming Xiao. 2021. Lexicon enhanced Chinese sequence labeling using BERT adapter. In *ACL*, pages 5847–5858.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Roberta: A robustly optimized BERT pretraining approach. *arXiv*, abs/1907.11692.

Fang Ma, Chen Zhang, and Dawei Song. 2021. Exploiting position bias for robust aspect sentiment classification. In *ACL*, pages 1352–1358.

Fang Ma, Chen Zhang, Bo Zhang, and Dawei Song. 2022. Aspect-specific context modeling for aspect-based sentiment analysis. *arXiv*, abs/2207.08099.

Haiyun Peng, Lu Xu, Lidong Bing, Fei Huang, Wei Lu, and Luo Si. 2020. Knowing what, how and why: A near complete solution for aspect-based sentiment analysis. In *AAAI*, pages 8600–8607.

Maria Pontiki, Dimitris Galanis, John Pavlopoulos, Harris Papageorgiou, Ion Androutsopoulos, and Suresh Manandhar. 2014. Semeval-2014 task 4: Aspect based sentiment analysis. In *SemEval*, pages 27–35.

Peng Qi, Yuhao Zhang, Yuhui Zhang, Jason Bolton, and Christopher D. Manning. 2020. Stanza: A python natural language processing toolkit for many human languages. In *ACL*, pages 101–108.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J. Liu. 2020. Exploring the limits of transfer learning with a unified text-to-text transformer. *JMLR*, 21:140:1–140:67.

Peter Shaw, Jakob Uszkoreit, and Ashish Vaswani. 2018. Self-attention with relative position representations. In *NAACL*, pages 464–468.

Changzhi Sun, Yuanbin Wu, Man Lan, Shiliang Sun, and Qi Zhang. 2017. Large-scale opinion relation extraction with distantly supervised neural network. In *EACL*, pages 1033–1043.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *NeurIPS*, pages 5998–6008.

Bailin Wang, Richard Shin, Xiaodong Liu, Oleksandr Polozov, and Matthew Richardson. 2020. RAT-SQL: relation-aware schema encoding and linking for text-to-sql parsers. In *ACL*, pages 7567–7578.

Lihan Wang, Bowen Qin, Binyuan Hui, Bowen Li, Min Yang, Bailin Wang, Binhua Li, Jian Sun, Fei Huang, Luo Si, and Yongbin Li. 2022. Proton: Probing schema linking information from pre-trained language models for text-to-sql parsing. In *KDD*, pages 1889–1898.

Yuanbin Wu, Qi Zhang, Xuanjing Huang, and Lide Wu. 2009. Phrase dependency parsing for opinion mining. In *EMNLP*, pages 1533–1541.

Zhen Wu, Chengcan Ying, Fei Zhao, Zhifang Fan, Xinyu Dai, and Rui Xia. 2020a. Grid tagging scheme for end-to-end fine-grained opinion extraction. In *EMNLP*, pages 2576–2585.

Zhiyong Wu, Yun Chen, Ben Kao, and Qun Liu. 2020b. Perturbed masking: Parameter-free probing for analyzing and interpreting BERT. In *ACL*, pages 4166–4176.

Lu Xu, Yew Ken Chia, and Lidong Bing. 2021. Learning span-level interactions for aspect sentiment triplet extraction. In *ACL*, pages 4755–4766.

Lu Xu, Hao Li, Wei Lu, and Lidong Bing. 2020. Position-aware tagging for aspect sentiment triplet extraction. In *EMNLP*, pages 2339–2349.

Hang Yan, Junqi Dai, Tuo Ji, Xipeng Qiu, and Zheng Zhang. 2021. A unified generative framework for aspect-based sentiment analysis. In *ACL*, pages 2416–2429.

Chen Zhang, Qiuchi Li, and Dawei Song. 2019a. Aspect-based sentiment classification with aspect-specific graph convolutional networks. In *EMNLP*, pages 4567–4577.

Chen Zhang, Qiuchi Li, and Dawei Song. 2019b. Syntax-aware aspect-level sentiment classification with proximity-weighted convolution network. In *SIGIR*, pages 1145–1148.

Chen Zhang, Qiuchi Li, Dawei Song, and Benyou Wang. 2020. A multi-task learning framework for opinion triplet extraction. In *EMNLP*, pages 819–828.

Wenxuan Zhang, Xin Li, Yang Deng, Lidong Bing, and Wai Lam. 2021. Towards generative aspect-based sentiment analysis. In *ACL*, pages 504–510.

A Data Statistics of SemEval

Dataset		#S	#T	#T/S	#Tk/S
L14	train	906	1460	1.61	19.15
	dev	219	346	1.58	19.06
	test	328	543	1.66	15.77
R14	train	1266	2336	1.85	17.31
	dev	310	577	1.86	15.81
	test	492	994	2.02	16.34
R15	train	605	1013	1.67	14.80
	dev	148	249	1.68	14.34
	test	322	485	1.51	15.63
R16	train	857	1394	1.63	15.15
	dev	210	339	1.61	14.16
	test	326	514	1.58	14.70

Table 5: Statistics of four datasets from **SemEval**. #S denotes number of sentences, #T denotes number of triplets, #T/S denotes average number of triplets per sentence, and #Tk/S denotes average number of tokens per sentence. L denotes laptop domain while R denotes restaurant domain.

B Full Implementation Details

Our models are implemented with PyTorch and verified on an Nvidia V100, and they are generally trained with following instructions.

For parameter settings in the benchmarking evaluation, the batch size is 8 for models without the adapter, whereas it is 6 for models with the adapter for stability, and the maximum norm for gradients is 1. The learning rate is set hierarchically, where the learning rate for the PLM and adapter is searched with {1e-5,2e-5,3e-5,5e-5} while that for the triplet parser is set 10 times of the former. The training procedure is scheduled as such: the number of maximum training epochs is 20, and the number of patience epochs is 5. Learning rates are warmed up for the first 2 epochs and decayed for the rest epochs. The threshold for the maximum distance τ is 8.

For parameter settings in the large-scale evaluation, the batch size is accordingly doubled, since we have data of a much larger scale.

Unsupervised Data Augmentation for Aspect Based Sentiment Analysis

David Z. Chen **Adam Faulkner** **Sahil Badyal**
Capital One Servicing Intelligence, NLP
{david.chen2, adam.faulkner, sahil.badyal}
@capitalone.com

Abstract

Recent approaches to Aspect-based Sentiment Analysis (ABSA) take a co-extraction approach to this span-level classification task, performing the subtasks of aspect term extraction (ATE) and aspect sentiment classification (ASC) simultaneously. In this work, we build on recent progress in applying pre-training to this co-extraction task with the introduction of an adaptation of Unsupervised Data Augmentation (UDA) in semi-supervised learning. As originally implemented, UDA cannot accommodate span-level classification since it relies on advanced data augmentation techniques, such as backtranslation, that alter the sequence lengths of the original data and cause index mismatches. We introduce an adaptation of UDA using Masked Language Model (MLM) unmasking that accommodates this index-match constraint and test the approach on standard ABSA benchmark datasets. We show that simple augmentations applied to modest-sized datasets along with consistency training lead to competitive performance with the current ABSA state-of-the-art in the restaurant and laptop domains using only 75% of the training data.

1 Introduction

Aspect-based Sentiment Analysis (ABSA) is a subset of Sentiment Analysis (SA), operating at the phrase- rather then sentence- or document-level. As with other forms of SA, the goal is to determine the sentiment associated with a given text segment, though, in the case of ABSA, these phrasal segments are typically "aspects" or features associated with products, services, or experiences, such as "waitstaff" or "ambience."

As with other span-level classification tasks, such as Named Entity Recognition (NER), a major challenge of ABSA is class imbalance, as the majority of token labels typically refer to non-aspect terms (Luo et al., 2020) and the terms themselves are of inconsistent phrase-level categories. This introduces considerable variance in aspect term labels and makes it difficult for models to effectively generalize to example terms outside those explicitly shown in the training data.

1.1 Related work

In related span-level tasks, previous work has shown that a joint/collapsed approach to entity and sentiment co-extraction out-performs a pipelined approach (Mitchell et al., 2013; Zhang et al., 2015). A joint approach refers to assigning two sets of tags, term and polarity, to each example, and a collapsed approach collapses the term and polarity tags into one term-polarity tag for each token. While there are merits to both approaches, we adopt a collapsed approach as it requires a simpler classifier architecture.

Pre-training and pre-trained language models (LMs) have been shown to provide state-of-the-art performance on many tasks within NLP. Applying these approaches to the ABSA task, Li et al. (Li et al., 2019b) and Luo et al. (Luo et al., 2020) have achieved state-of-the-art performance on restaurant and laptop reviews using pre-trained LM's and LM's with post-training (PTR), respectively. These results highlight the benefit of leveraging unlabeled data (with pre-training and post-training).

In this work, we explore and push the limits of using unlabeled data for the ABSA task by incorporating data augmentation and consistency training on top of pre-trained and post-trained BERT. We adopt an unsupervised data augmentation (UDA) technique based in semi-supervised learning (SSL) from Xie et al. (Xie et al., 2020), initially developed

Proceedings of the 29th International Conference on Computational Linguistics, pages 6746–6751
October 12–17, 2022.

UDA method	Original Text	Augmented Text
Backtranslation	Our meal was so tasty but the waitstaff kept making rude remarks!	Food was delicious but the waiter spoke rudely.
MLM unmasking	Our <MASK> was so tasty but the waitstaff kept making rude remarks!	Our lunch was so tasty but the waitstaff kept making rude remarks!

Table (1) Example augmentation methods and texts. In the backtranslation case, "meal"-"food" and "waitstaff"-"waiter" are index mismatched after augmentation, whereas MLM unmasking preserves token indices.

for document- and sentence-level classification tasks, and adapt it to the span-level classification setting. In UDA, unlabeled data is passed to the model in streams of pairs, where one stream contains the original unmodified input example and the other stream contains augmented examples created by transforming the original input using data augmentation techniques. In the paper by Xie et al. (Xie et al., 2020), the authors apply data augmentation to images (e.g. filters and image transformations) as well as to sentence-level textual data via backtranslation. This backtranslation approach, while powerful in creating augmented examples that differ greatly from the original while retaining semantic meaning, results in index-mismatch issues when applied to span-level tasks such as ABSA. For this reason, we adapted the original UDA implementation to work for span-level ABSA by applying simple token replacements using masked-language model (MLM) unmasking (additional details provided in the Method section). Our adapted form of UDA-based data augmentation shows competitive performance with the ABSA state-of-the-art using only 75% of the original labeled training data and 30k additional unlabeled examples.

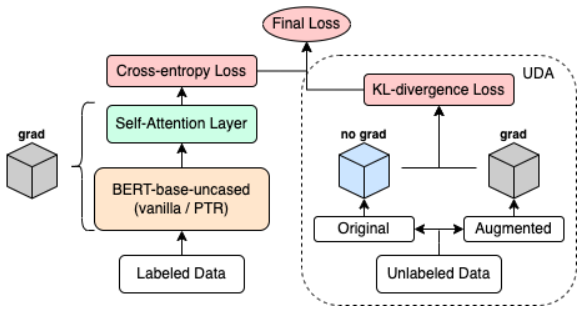

Figure (1) Model diagram with UDA

2 Method

Figure 1 shows the model architecture diagram for the models used in our experiments. One constant throughout is the use of *BERT-base-uncased* [1] as the base pre-trained LM and a Self-Attention Network (SAN) as the classification layer. In the purely supervised condition (no UDA), training is done in the usual way by calculating and backpropagating a cross-entropy loss between prediction and target, where collapsed labels are in the "BIOES" tagging scheme, and sentiment tags are appended to each BIES tag, e.g. B-POS, B-NEU, B-NEG, resulting in 13 classes in total.

2.1 Unsupervised Data Augmentation

For model variations using UDA, we include a separate model that performs backpropagation on unsupervised augmented datasets based on a KL-divergence loss between the model prediction on the augmented example (unfrozen) and the model prediction on the original example (frozen), as described in the original UDA paper (Xie et al., 2020). See Figure 1.

As currently implemented, advanced data augmentation techniques used in UDA, such as backtranslation, are incompatible with span-level classification tasks like ABSA, which require that the sequence length of the original example match that of the augmented example in order for the aspect terms to be correctly indexed, extracted and labeled. This is because backtranslation frequently results in augmented sequences of varying lengths from the original, leading to token index mismatches (see examples in Table 1). As a result, KL-divergence loss would fail to capture the error between the original and augmented aspects, as their relative positions will have changed.

In order to accomodate the index-match constraint required for ABSA, we introduce a simple augmentation technique that utilizes single- and multi-token replacement via unmasking using vanilla BERT-uncased MLM. This choice of BERT MLM is to remain consistent with previous work in ABSA utilizing BERT. Other choices can be used instead, such as one of the BERT variants, e.g. RoBERTa, DistilBERT (Liu et al., 2019; Sanh et al., 2019). In order to obtain augmentations that are general, we

[1] Available in *Hugging Face*: https://huggingface.co

Model		Rest14	Rest15	Rest16	Laptop
IMN	(He et al., 2019)	69.54	59.18	-	58.37
DREGCN	(Liang et al., 2020a)	72.60	62.37	-	63.04
WHW	(Peng et al., 2020)	71.95	65.79	71.73	62.34
TAS-BERT	(Wan et al., 2020)	-	66.11	75.68	-
IKTN-BERT	(Liang et al., 2020b)	71.75	62.33	-	62.34
DHGNN	(Liu et al., 2020)	68.91	58.37	-	59.61
RACL-BERT	(Chen and Qian, 2020)	75.42	66.05	-	63.40
BERT-E2E-ABSA	(Li et al., 2019b)	73.68	59.90	70.51	61.12
GRACE	(Luo et al., 2020)	77.26	68.16	76.49	**70.71**
UDA-ABSA		**79.38** ± 0.38	**70.14** ± 0.89	**78.05** ± 0.72	69.55 ± 0.40
-75% train sample		77.09 ± 0.52	68.19 ± 0.70	75.38 ± 0.27	-
-50% train sample		76.73 ± 0.17	64.02 ± 1.01	72.96 ± 0.70	-
-25% train sample		73.33 ± 0.24	58.71 ± 1.07	68.72 ± 1.45	-

Table (2) Experimental micro F1 values compared across previous work and UDA-ABSA

chose to utilize vanilla BERT-uncased for this MLM task rather than one post-trained on in-domain data.

Span-based UDA Single-unigram replacement augmentation

1: Randomly select a token in the tokenized original sequence (avoiding punctuation)
2: Convert the selected token into [MASK]
3: Unmask the token using our LM,
4: Check $token_{unmasked} \neq token_{original}$ and $token_{unmasked} \notin punctuation$.

In multi-unigram replacement augmentation, we iterate on the single-unigram case over S_{length} times, where S_{length} is the sequence length. However, we apply a confidence threshold, $\gamma = 0.1$, to the unmasking so that only unmasked tokens with confidence $> \gamma$ are kept.

2.2 Datasets

We leveraged datasets from two domains, namely Restaurants (Rest14, Rest15, Rest16) and Laptops, both originating from SemEval (Pontiki et al., 2014, 2015, 2016). Specifically, we use versions prepared by Li et al. (Li et al., 2019a), which uses collapsed ABSA labels. For UDA, we utilized the 27k examples from the Yelp academic dataset for the Restaurant domain, [2] and we filtered Amazon electronics reviews to obtain 38k examples pertaining to laptops. [3]

[2]https://www.yelp.com/academic_dataset
[3]http://jmcauley.ucsd.edu/data/amazon/

2.3 Model parameters

All of our models share the same underlying architecture: BERT-base-uncased (vanilla and post-trained) with a learning rate of $2e-5$ and an AdamW optimizer with a linear learning rate schedule. For post-training (PTR), we adopted the trained weights from Luo, et al. (Luo et al., 2020), which was achieved by performing Whole Word Masking using BERT-based-uncased on 142.8M Amazon reviews (footnote) joined with 2.2M Yelp reviews (footnote).

Experiments are conducted using a fixed number of gradient optimization steps, which is set heuristically and is sufficiently high to allow for complete model convergence (loss plateaus over at least one entire epoch). The actual number of steps varied depending on the batch sizes involved, but ranged from 5k-20k batches, or 8-15 epochs. The supervised batch sizes were set according to GPU memory limitations, as UDA greatly increases the GPU memory load during training, resulting in typical supervised batch sizes of 4 for Restaurant data, and 6 for Laptop data with UDA, and 16 and 32 for experiments without UDA, respectively. Likewise, the unsupervised batch sizes for UDA were set according to the ratio of $N_{unsupervised}/N_{supervised}$ multiplied by the supervised batch size, resulting in typical UDA batch sizes of 40-84.

Each experimental condition is conducted 5 times with 5 constant seeds, and the result-

ing micro-F1 values are averaged over those 5 replicates. We also provide the resulting standard deviations. This is performed in order to calculate statistical significance, as well as increase the confidence of our estimate of the average performance. For each iteration of the 5 total replicates, we estimate test micro-F1 by calculating a a 800-step moving average (∼1.2 epochs) on the test dataset after model convergence.

3 Results

3.1 Benchmark experiments

Table 2 shows the results of our experiments on the rest14, rest15, rest16, and laptop datasets. We found that UDA-ABSA is competitive with the state-of-the-art, and achieves this with pre-trained BERT-uncased, PTR, and UDA on additional unlabeled data during training with as little as 75% of the labeled data.

Model	**Rest14**
UDA-ABSA	**79.38** ± 0.38
no UDA	78.75 ± 0.38
no PTR	75.16 ± 0.33
no UDA & no PTR	73.50 ± 0.90

Table (3) UDA-ABSA Ablation Experiments

3.2 Ablation experiments

In order to assess the relative contributions of UDA and PTR to the model's performance, we conducted ablation experiments on the Rest14 dataset. Namely, we estimated model micro-F1 *w/o PTR* and *w/o UDA* and *w/o UDA and PTR*. Results are shown in Table 3. Removing PTR contributed a decrease of roughly ∼4% average micro F1, while removing UDA contributed a small, but significant (p val ≈ 0.048), decrease of $< 1\%$ average micro F1. Additionally, removing both PTR and UDA contributed a decrease of roughly ∼6% average micro F1. These observations suggest that:

- PTR contributes the most to enhancing the performance of the model.
- UDA may contribute more to the performance in the absence of PTR.
- While PTR leverages over 140M unlabeled examples, UDA improves model performance further.
- UDA may be more data-efficient: i.e. gain per number of ex. of PTR (∼0.03%/Million examples) vs. UDA (∼21%/Million examples).

However, it is not clear how such performance estimates would extrapolate to the small data regime for PTR and large data regime for UDA, as such experiments have not been conducted due to hardware constraints.

Model	**Rest14**
Single-Linear	**79.38** ± 0.38
Single-Log	79.19 ± 0.34
Single-Exp	79.10 ± 0.29
Multi-Linear	79.05 ± 0.37
Single-no CT	74.03 ± 0.32

Table (4) UDA Parameter Experiments

4 Discussion

In our experiments, we found that, not surprisingly, downsampling the training data results in a degradation of model performance, but we also found that UDA achieves SOTA-competitive performance with 75% of the data. The original UDA paper (Xie et al., 2020) showed good performance with only 20 training examples for sentiment classification. In our dataset, we achieved performance competitive with BERT-based-uncased (Li et al., 2019b) with only 25% of the training data.

In addition to benchmarks and ablation experiments, we explored the role of different types of confidence thresholding (CT) on the final performance of our models. The confidence threshold, ϕ, filters out unsupervised examples that fall below ϕ during UDA, so that the model does not reinforce its own errors, and ϕ is typically increased during training on a schedule (Xie et al., 2020), namely, *linear*, *log*, and *exp*.

We found that CT during UDA is important to the stability and convergence of our models. Figure 2 shows the training curves for test data micro-F1 with and without CT. Our models converge faster with CT, and interestingly, the curve for our models without CT shows a kink after ∼1 epoch, where micro-F1 appears to saturate only to rebound as training continues. We hypothesize that this kink is the result of the model initially learning the wrong features, which are subsequently relearned during further training. However, the final performance of our models without CT never reaches those with CT see Table 4 and Figure 2. This observation highlights the importance of CT for UDA and supports other recent work in SSL that have found success with such concepts (Sohn et al.,

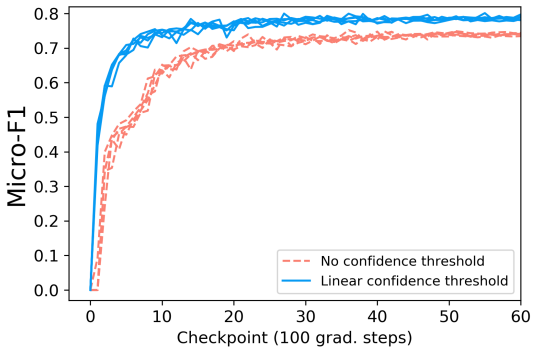

Figure (2) Rest14 test micro-F1 with and without confidence thresholding

2020).

5 Summary

We introduced a span-level modification of the UDA procedure, which, along with post-training on BERT, achieves performance competitive with state-of-the-art on the restaurant and laptop domains for ABSA with 75% of the data. While post-training contributed the most to overall performance, UDA may be more efficient on a per data/compute basis. We observed that confidence thresholding is essential to stabilize model training and achieve greater performance, and that linear confidence threshold scheduling achieved the best performance along with single augmentations compared to multiple augmentations. This work reveals the benefit of using UDA for span-level tasks and with post-trained language models.

References

Zhuang Chen and Tieyun Qian. 2020. Relation-aware collaborative learning for unified aspect-based sentiment analysis. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 3685–3694, Online. Association for Computational Linguistics.

Ruidan He, Wee Sun Lee, Hwee Tou Ng, and Daniel Dahlmeier. 2019. An interactive multi-task learning network for end-to-end aspect-based sentiment analysis. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 504–515, Florence, Italy. Association for Computational Linguistics.

Xin Li, Lidong Bing, Piji Li, and Wai Lam. 2019a. A unified model for opinion target extraction and target sentiment prediction. *Proceedings of the AAAI Conference on Artificial Intelligence*, 33(01):6714–6721.

Xin Li, Lidong Bing, Wenxuan Zhang, and Wai Lam. 2019b. Exploiting BERT for end-to-end aspect-based sentiment analysis. In *Proceedings of the 5th Workshop on Noisy User-generated Text (W-NUT 2019)*, pages 34–41, Hong Kong, China. Association for Computational Linguistics.

Yunlong Liang, Fandong Meng, Jinchao Zhang, Jinan Xu, Yufeng Chen, and Jie Zhou. 2020a. A dependency syntactic knowledge augmented interactive architecture for end-to-end aspect-based sentiment analysis. *CoRR*, abs/2004.01951.

Yunlong Liang, Fandong Meng, Jinchao Zhang, Jinan Xu, Yufeng Chen, and Jie Zhou. 2020b. An iterative knowledge transfer network with routing for aspect-based sentiment analysis. *CoRR*, abs/2004.01935.

Shu Liu, Wei Li, Yunfang Wu, Qi Su, and Xu Sun. 2020. Jointly modeling aspect and sentiment with dynamic heterogeneous graph neural networks. *CoRR*, abs/2004.06427.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Roberta: A robustly optimized bert pretraining approach. *ArXiv*, abs/1907.11692.

Huaishao Luo, Lei Ji, Tianrui Li, Daxin Jiang, and Nan Duan. 2020. GRACE: Gradient harmonized and cascaded labeling for aspect-based sentiment analysis. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 54–64, Online. Association for Computational Linguistics.

Margaret Mitchell, Jacqui Aguilar, Theresa Wilson, and Benjamin Van Durme. 2013. Open domain targeted sentiment. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*, pages 1643–1654, Seattle, Washington, USA. Association for Computational Linguistics.

Haiyun Peng, Lu Xu, Lidong Bing, Fei Huang, Wei Lu, and Luo Si. 2020. Knowing what, how and why: A near complete solution for aspect-based sentiment analysis. *Proceedings of the AAAI Conference on Artificial Intelligence*, 34(05):8600–8607.

Maria Pontiki, Dimitris Galanis, Haris Papageorgiou, Ion Androutsopoulos, Suresh Manandhar, Mohammad AL-Smadi, Mahmoud Al-Ayyoub, Yanyan Zhao, Bing Qin, Orphée De Clercq, Véronique Hoste, Marianna Apidianaki, Xavier Tannier, Natalia Loukachevitch, Evgeniy Kotelnikov, Nuria Bel, Salud María Jiménez-Zafra, and Gülşen Eryiğit. 2016. SemEval-2016 task 5: Aspect based sentiment analysis. In *Proceedings of the 10th International Workshop on Semantic Evaluation (SemEval-2016)*, pages 19–30, San

Diego, California. Association for Computational Linguistics.

Maria Pontiki, Dimitris Galanis, Haris Papageorgiou, Suresh Manandhar, and Ion Androutsopoulos. 2015. SemEval-2015 task 12: Aspect based sentiment analysis. In *Proceedings of the 9th International Workshop on Semantic Evaluation (SemEval 2015)*, pages 486–495, Denver, Colorado. Association for Computational Linguistics.

Maria Pontiki, Dimitris Galanis, John Pavlopoulos, Harris Papageorgiou, Ion Androutsopoulos, and Suresh Manandhar. 2014. SemEval-2014 task 4: Aspect based sentiment analysis. In *Proceedings of the 8th International Workshop on Semantic Evaluation (SemEval 2014)*, pages 27–35, Dublin, Ireland. Association for Computational Linguistics.

Victor Sanh, Lysandre Debut, Julien Chaumond, and Thomas Wolf. 2019. DistilBERT, a distilled version of BERT: smaller, faster, cheaper and lighter. In *5th Workshop on Energy Efficient Machine Learning and Cognitive Computing @ NeurIPS 2019*.

Kihyuk Sohn, David Berthelot, Chun-Liang Li, Zizhao Zhang, Nicholas Carlini, Ekin D. Cubuk, Alex Kurakin, Han Zhang, and Colin Raffel. 2020. Fixmatch: Simplifying semi-supervised learning with consistency and confidence. In *34th Conference on Neural Information Processing Systems*. NeurIPS.

Hai Wan, Yufei Yang, Jianfeng Du, Yanan Liu, Kunxun Qi, and Jeff Z. Pan. 2020. Target-aspect-sentiment joint detection for aspect-based sentiment analysis. *Proceedings of the AAAI Conference on Artificial Intelligence*, 34(05):9122–9129.

Qizhe Xie, Zihang Dai, Eduard Hovy, Minh-Thang Luong, and Quoc V Le. 2020. Unsupervised data augmentation for consistency training. In *34th Conference on Neural Information Processing Systems*. NeurIPS.

Meishan Zhang, Yue Zhang, and Duy-Tin Vo. 2015. Neural networks for open domain targeted sentiment. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 612–621, Lisbon, Portugal. Association for Computational Linguistics.

A Sentiment and Emotion aware Multimodal Multiparty Humor Recognition in Multilingual Conversational Setting

Dushyant Singh Chauhan[1]*, **Gopendra Vikram Singh**[1], **Aseem Arora**[1], **Asif Ekbal**[1], **and Pushpak Bhattacharyya**[2]
Department of Computer Science and Engineering
[1] Indian Institute of Technology Patna, India
{1821cs17,gopendra_1921cs15,aseem_1911mc02,asif}@iitp.ac.in
[2] Indian Institute of Technology Bombay, India
pb@cse.iitb.ac.in

Abstract

In this paper, we hypothesize that humor is closely related to sentiment and emotions. Also, due to the tremendous growth in multilingual content, there is a great demand for building models and systems that support multilingual information access. To this end, we first extend the recently released *Multimodal Multiparty Hindi Humor* (M2H2) dataset by adding parallel English utterances corresponding to Hindi utterances and then annotating each utterance with sentiment and emotion classes. We name it *Sentiment, Humor, and Emotion aware Multilingual Multimodal Multiparty Dataset* (SHEMuD). Therefore, we propose a multitask framework wherein the primary task is humor detection, and the auxiliary tasks are sentiment and emotion identification. We design a multitasking framework wherein we first propose a *Context Transformer* to capture the deep contextual relationships with the input utterances. We then propose a *Sentiment and Emotion aware Embedding* (SE-Embedding) to get the overall representation of a particular emotion and sentiment *w.r.t.* the specific humor situation. Experimental results on the *SHEMuD* show the efficacy of our approach and shows that multitask learning offers an improvement over the single-task framework for both monolingual (4.86 points ↑ in Hindi and 5.9 points ↑ in English in F1-score) and multilingual (5.17 points ↑ in F1-score) setting.

1 Introduction

Humor (Ritschel et al., 2019, 2020; Song et al., 2021; Chauhan et al., 2021) is an essential aspect of daily conversation, and people try to provoke humor in their talks. Warren et al. (2018) defined humor as *"the nature of experiences to induce laughter and provide amusement"*. Humor is a tool by which anyone can draw the audience's attention. Even in a formal conversation, humor may make a person look more attractive and thus may lead to a better conclusion.

Irrespective of its relation to intelligence, humor is inherently a challenging problem to understand. To understand humor, we also take some additional information into consideration i.e., sentiment (Ghosal et al., 2018; Chauhan et al., 2019, 2020a, 2022) and emotion (Russell and Barrett, 1999; Pagé Fortin and Chaib-draa, 2019; Chauhan et al., 2020b) to help humor detection in conversations. Sentiment and emotion are two aspects that help each other, which has already been shown in (Akhtar et al., 2019). We hypothesize that sentiment and emotion do not only help each other (Akhtar et al., 2019) but also help humor. For example, it is difficult to decide whether the following utterance "अरे बुढापे में मैं तुम्हें खुद ले के जाता। (Oh in old age I would have taken you myself)" is humorous or not. But, careful observation of the sentiment (positive) and emotion (happy) of the speaker helps us understand that the speaker is in a funny mood and trying to mock his wife.

Sometimes it is difficult for the global audience to understand any local language like Hindi, so this is where multilingual comes into the picture. Also, multilinguality provides freedom for the model to understand humor; e.g., if the model is unable to understand the one language's word (say Hindi) for a particular utterance, then other languages (say English) can give a better clue to understand humor. Thus, we propose a multitask framework in a multilingual setting wherein the primary task is humor detection, and the auxiliary tasks are sentiment and emotion identification.

The main contributions of our proposed research are as follows;

- We first extend recently released M2H2 dataset by adding parallel English utterances corresponding to Hindi utterances and then annotate each utterance with sentiment and emotion classes. We name it *Sentiment, Humor,*

*The first two authors contributed equally to this work and are jointly the first authors.

Proceedings of the 29th International Conference on Computational Linguistics, pages 6752–6761
October 12–17, 2022.

and Emotion aware Multilingual Multimodal Multiparty Dataset (SHEMuD).

- We propose a *Context Transformer* to capture the deep contextual relationship with input utterance.
- We also propose a *Sentiment and Emotion Embedding* (SE-Embedding) to obtain the overall representation of a particular emotion and sentiment *w.r.t.* the specific humor situation.
- We present new state-of-the-art systems for Humor detection on *SHEMuD*.

2 Related Work

We have already discussed that sometimes just going through the utterance text is not enough to understand humor. The other modalities, such as visual and acoustic, can provide additional cues. In natural language processing, understanding humor from these modalities comes under the boundaries of multimodal language. Humor may be found in almost every human-to-human encounter. Some works (Hasan et al., 2019; Fallianda et al., 2018; Ritschel et al., 2019, 2020; Mirnig et al., 2017; Piata, 2020; Song et al., 2021; Vasquez and Aslan, 2021; Sabur et al., 2020; Veronika, 2020; Yang et al., 2019) on multimodal humor have already been done.

Hasan et al. (2019) proposed the UR-FUNNY dataset to aid in the comprehension of multimodal language utilized in the expression of humor. The author has also shown the importance of multimodality over unimodality. Humor can be created with any modality. Fallianda et al. (2018) looked at how humor was created using only verbal media, both verbal and visual media, and exclusively visual media. The data was gathered from 74 political comic strips published in Kompas newspaper. The author used the General Theory of Verbal Humor (GTVH) for humor analysis.

Higher pitch or loudness, a faster speaking cadence, or considerable pauses are not characteristics of conversational humor. In the paper, Attardo et al. (2011) discovered that when people are laughing or smiling, they are more likely to be humorous. El Khatib (2020) did two experiments: the first experiment looks at whether untrained people can recognize the structural aspects of humor in a multimodal text. The second research investigates the sequence in which textual and visual inputs are processed (tweets). In another work, Ritschel et al. (2019) discussed the quality of a conversation and how conversation partners see one other and are influenced by irony and irony-related humor.

The Humor Styles Questionnaire (HSQ) distinguishes four types of humor that can be good or damaging to one's mental health. Schneider et al. (2018) studied to compile research that used the HSQ to analyze the relationships between distinct humor types and four distinct aspects of mental health (self-esteem, life satisfaction, optimism, depression). Recently, Chauhan et al. (2021) proposed Multimodal Multiparty Hindi Humor (M2H2) dataset for conversations which was the very first of its kind. Then, the authors employed two strong baseline setups, viz. MISA[1] w/ DialogueRNN and MISA w/ bcLSTM. They used MISA for fusion and DialogueRNN & bcLSTM for understanding the contextual relationship among words.

In comparison to the existing systems, we first created the *SHEMuD* by manually annotating each utterance in the English language and then labeling each utterance with sentiment and emotion classes. After that, we design a multitasking framework wherein we first propose a *Context Transformer* to capture the deep contextual relationships with the input utterances. We then propose a *Sentiment and Emotion aware Embedding* (SE-Embedding) to get the overall representation of a particular emotion and sentiment *w.r.t.* the specific humor situation. Further, to the best of our knowledge, this is the first attempt at solving the sentiment and emotion aware humor detection in a multilingual conversational setting. Empirical results on the *SHEMuD* dataset demonstrate that the baselines yield better performance over the state-of-the-art systems.

3 Datasets

Chauhan et al. (2021) proposed the *Multimodal Multiparty Hindi Humor* (M2H2) dataset which contains 6,191 utterances spoken by 43 speakers from 13 episodes of a very popular TV series *"Shrimaan Shrimati Phir Se"* where each episode is divided into multiple scenes, and each scene is divided into multiple utterances. Each utterance is annotated with humor/non-humor labels and encompasses acoustic, visual, and textual modalities.

We extend the M2H2 dataset (Chauhan et al.,

[1]MISA:Modality-Invariant and-Specific Representations for Multimodal Sentiment Analysis

	Hindi Utterances	English Utterances	Humor	Sentiment	Emotion
1	अरे बुढापे में मैं तुम्हें खुद ले के जाता।	Oh in old age I would have taken you myself.	humor	positive	happy
2	ये लोग मुझे बहुत मार रहे हैं।	These guys are beating me hard	humor	positive	sad
3	ये दिलरुबा कहता है ना कि पिछले जन्म में वो छोटूमल मोटूमल करोडपति का एकलौता बेटा था, राइट?	This Dilruba says that he was the only son of Chhotumal Motumal Crorepati in his previous life, right?	non-humor	neutral	neutral
4	अरे भाई, मेरी असली वाली गन यहीं छूट गैइ।	Hey brother, my original gun is left here	non-humor	neutral	neutral
5	स्टाफ वालों! मेरा मतलब	Staff guys! I mean	humor	negative	anger

Table 1: Some samples from annotated dataset

2021) by manually annotating each Hindi utterance with the English language, making it a multilingual dataset (*SHEMuD*), and then annotating each utterance with sentiment and emotion classes. For sentiment annotation, we consider three sentiment classes, namely *positive, negative* and *neutral*. While for emotion annotation, we annotate the dataset with seven emotion values[2], *viz.* anger, disgust, fear, happy, sad, surprised, and neutral (c.f. Table 1). We show statistics of *SHEMuD* in Table 2.

Statistics	Train	Dev	Test
#Utterances	5000	500	691
#Humor (H)	1830	155	104
#Non-humor (NH)	3170	345	587
#Positive (Ps)	1048	102	197
#Neutral (Nu)	2488	245	307
#Negative (Ng)	1464	153	187
#Anger (Ag)	811	105	130
#Disgust (Dg)	65	19	22
#Fear (Fr)	93	30	40
#Happy (Ha)	648	65	102
#Sad (Sd)	300	55	60
#Surprise (Sp)	306	81	119
#Neutral (Nu)	2777	145	218

Table 2: Dataset statistics for *SHEMuD*

Please note that humor, many times, is related to the language as well as the culture. So, we cannot guarantee that if a Hindi utterance is humorous, then English will be humorous, but sharing information across the languages helps each other in humor prediction.

3.1 Annotation Guidelines

We employ four Ph.D. students highly proficient in the Hindi and English languages with prior experience in labeling *sentiment and emotion*. The guidelines for annotation were explained to the annotators before starting the annotation process (c.f. Table 1). The annotators were given data without humor labels and asked first to add English utterances corresponding to Hindi utterances and then annotate every utterance with one emotion and the sentiment by seeing the context utterances before annotation. The majority voting scheme was used for selecting the final emotion and sentiment. We achieve an overall Fleiss' (Fleiss, 1971) kappa score of 0.83, which is considered to be reliable.

[2]One emotion per utterance

3.2 Feature Extraction

For monolingual Hindi and English textual features, we take the pre-trained Hindi and English embedding using `XLM-Roberta`[3] (Conneau et al., 2019) (XLM-R). While, for multilinguality, we first train XLM-R on both English and Hindi (shared embedding) and then we test the result on both Hindi and English. Thus, we extract multilingual embedding for Hindi and English.

Similarly, for visual feature extraction, we use `3D-ResNeXt-101`[4] (Xie et al., 2017) which is pre-trained on Kinetics at a rate of 1.5 features per second and a resolution of 112. We use `openSMILE`[5] (Eyben et al., 2010) for acoustic feature extraction. It can extract Low-Level Descriptors (LLD) and change them using different filters, functions, and transformations. We use a tonal low-level features group of openSMILE to extract the features.

4 Proposed Methodology

This section describes our proposed model, where we aim to leverage multimodal sentiment and emotion information for solving multimodal humor detection in a multitask framework. We depict the overall architecture in Figure 1. The extended dataset and source code are available at `https://www.iitp.ac.in/~ai-nlp-ml/resources.html#SHEMuD`

[3]`https://huggingface.co/xlm-roberta-base`

[4]`https://github.com/kaiqiangh/extracting-video-features-ResNeXt`

[5]`https://github.com/audeering/opensmile`

4.1 Context Transformer

We propose a Context Transformer (Con_{trans}) to capture the deep contextual relationship with input utterance. As we know, contextual utterances might give essential information when identifying an input utterance. This necessitates a model that accounts for such interdependencies and the impact they may have on the target utterance. We use a Transformer based approach to capture the flow of informative triggers across the utterances.

Let us assume, the unimodal features have dimension p and each utterance is thus represented by a feature vector $d_{l,s} \in \mathcal{R}^p$, where s represents the sth utterance in a conversation l. For a conversation, we collect the feature vectors of the utterances in D_L, which is as follows;

$$D_L = [d_{l,1}, d_{l,2}, \ldots, d_{l,J_l}] \in \mathcal{R}^{J_l \times p} \quad (1)$$

where J_l represents the number of utterances in a conversation. The matrix D_L serves as input to the Transformer. To learn the representation of D_L, we first map it into the continuous space U_c;

$$\begin{aligned} U_c &= u^i_1, u^i_2, \ldots, u^i_{|D_L|} \\ u^i_j &= u(d_{l,1}) + p_j \end{aligned} \quad (2)$$

where $u(d_{l,1})$ & p_j are the utterance vectors and positional embedding of every utterance, respectively. We adopt sine-cosine positional embedding (Vaswani et al., 2017) as it performs better and does not introduce additional trainable parameters.

The contextual encoder converts U_c into a list of hidden representations $h^i_1, h^i_2, \ldots, h^i_{|U_i|}$. We use the last hidden representation $h^i_{|U_i|}$ (i.e., the representation at the EOS token) as the contextual representation of the utterance. Therefore, the final contextual representation of the D_L;

$$D_L = h_i = h^i_{|D_L|} + p_i \quad (3)$$

Please note that words and sentences share the same positional embedding matrix.

For multimodal inputs, we simply concatenate the input embeddings and pass them to the Con_{trans} to capture the deep contextual relationship with the input utterance. We pass the output of Con_{trans} to *Linear Layer* then output of this is fed to two *Softmax* layers for sentiment and emotion prediction, respectively.

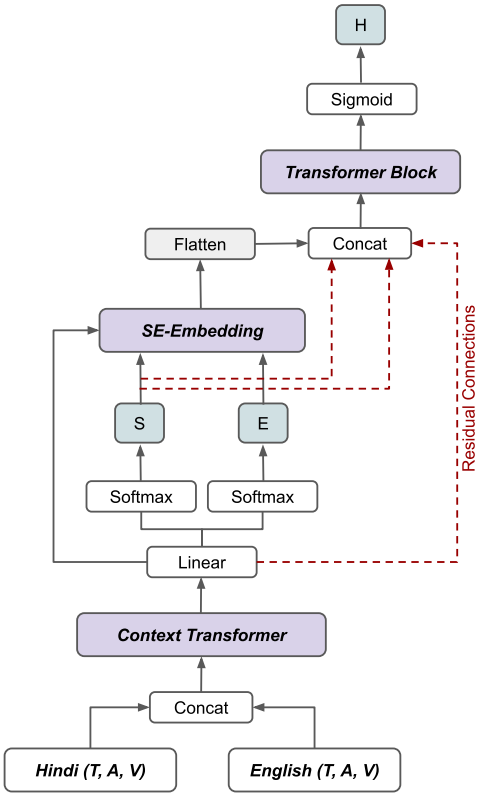

Figure 1: A contextual transformer based proposed framework for humor detection where *SE-Embedding* stands for Sentiment and Emotion Embedding.

$$\begin{aligned} cat_{TAV} &= Concat(T, A, V) \\ Con_{rel} &= Con_{trans}(cat_{TAV}) \\ L_{vec} &= Linear(Con_{rel}) \\ S_p &= Softmax(L_{vec}) \\ E_p &= Softmax(L_{vec}) \end{aligned} \quad (4)$$

We use a weighted loss function (Kendall et al., 2018) to teach the model how to weigh the task-specific losses. We define the loss as L_{total};

$$L_{total} = \sum_i w_i L_i \quad (5)$$

where *i* defines the different tasks i.e., Sentiment and Emotion. Initially we define w with **xavier-norm** for each task.

4.2 Sentiment and Emotion Embedding

We then propose SE-Embedding (SE_{emb}) to obtain the overall representation of a particular emotion and sentiment *w.r.t.* the specific humor situation. We take output of Linear Layer ($vec \in \mathcal{R}^k$), sentiment prediction ($S_p \in \mathcal{R}^3$), and emotion prediction ($E_p \in \mathcal{R}^7$) as input.

$$O_{rep} = SE_{emb}(L_{vec}, S_p, E_p) \quad (6)$$

We then concatenate S_p & E_p and obtain $SE_p \in \mathcal{R}^{10}$. After that we multiply SE_p & L_{vec} and get sentiment-emotion aware matrix ($SE_{mat} \in \mathcal{R}^{10\times k}$).

$$SE_{mat} = Concat(S_p, E_p) \times L_{vec} \quad (7)$$

We initially define SE_{emb} with random weights and later we apply convolve function (*) between SE_{emb} & SE_{mat} and update the weights of SE_{emb}.

$$SE_{emb} = SE_{emb} * SE_{mat} \quad (8)$$

We use the convolve function to obtain the discrete, linear convolution of SE_{emb} & SE_{mat}. Let us say there are two one-dimensional input arrays f & g. So, we compute convolve function between f & g;

$$(f * g)[n] = \sum_{n=-\infty}^{+\infty} f[m]g[n-m] \quad (9)$$

Let us suppose the model predicts the positive sentiment and happy emotion, then only corresponding vectors from SE_{emb} will go forward because non-predicted (n_p) vectors from SE_{emb} may confuse the model. To achieve this, we convert all the non-predicted vectors to -1 by first multiplying with 0 and then adding -1.

$$SE_{emb} = \forall n_p \; (SE_{emb} \times 0 - 1) \quad (10)$$

Then, we flatten the SE_{emb}

$$F_{out} = Flatten(SE_{emb}) \quad (11)$$

Motivated by the residual skip connection network (He et al., 2016), we concatenate the output of *Linear Layer* (L_{vec}), S_p, and E_p with the flatten output and pass to the Transformer Block ($Trans_b$) to learn the relationship between humor and sentiment & emotion.

$$\begin{aligned} cat_{out} &= Concat(F_{out}, L_{vec}, S_p, E_p) \\ trans_{out} &= Trans_b(cat_{out}) \end{aligned} \quad (12)$$

At last, we pass the output of *Transformer Block* to the *Sigmoid* layer for humor detection.

$$H_p = Sigmoid(trans_{out}) \quad (13)$$

We use *binary cross-entropy* loss for humor detection. Please note that two losses are back-propagated, one from humor and another from sentiment & emotion.

5 Experimental Setup

We evaluate our proposed model on the *SHEMuD*. We perform a grid search to find the optimal hyper-parameters, which are shown in Table 3. We take five[6] utterances as context for a particular input utterance. We implement our proposed model on the Python-based PyTorch deep learning library. As the evaluation metric, we employ precision (P), recall (R), and F1-score (F1) for sentiment, emotion, and humor recognition.

Parameters	Proposed
Transformer Encoder	2 Layers (Context Transformer, Transformer Block)
Dense layer	1024N, D=0.3
Activations	ReLu
Optimizer	Adam (*lr=1e-3*)
Outputs	*Softmax* (Sent and Emo) *Sigmoid* (Humor)
Batch	16
Epochs	60

Table 3: Model configurations

6 Results and Analysis

We divide our experiments into two parts, i.e., monolingual and multilingual;

6.1 Monolingual (Multitask vs Single-task)

In monolingual experiments, we evaluate our proposed model for Hindi and English languages separately. We perform experiments for unitask (Humor), bitask (Humor+Sentiment and Humor+Emotion), and tritask (Humor+Sentiment+Emotion). We show experimental results in Table 4. For Hindi tritask (H+S+E), our proposed model shows an improvement of 4.53 points in precision, 4.1 points in recall, and 3.98 points in F1-score over bitask[7] while 4.55 points in precision, 4.41 points in recall, and 4.86 points in F1-score over unitask.

We get a similar improvement for English tritask (H+S+E). For English tritask (H+S+E), our proposed model shows an improvement of 3.63 points in precision, 4.46 points in recall, and 4.2 points in F1-score over bitask[8] while 5.63 points in preci-

[6]For fair comparison with M2H2 (2021) as they took five utterances as context.

[7]We get a maximum improvement over H+E

[8]We get a maximum improvement for H+E

	Monolingual						*Multilingual*					
	Hindi			*English*			*Hindi*			*English*		
Labels	**P**	**R**	**F1**	**P**	**R**	**F1**	**P**	**R**	**F1**	**P**	**R**	**F1**
H	71.82	73.91	72.47	74.89	76.64	75.42	74.84	75.89	75.36	78.32	79.25	78.78
H+S	73.42	74.91	74.46	77.63	79.32	78.74	77.11	78.94	78.01	81.22	82.43	81.79
H+E	72.14	74.22	73.35	76.89	77.68	77.12	76.34	77.49	76.91	80.49	81.37	80.92
H+S+E	**76.37**	**78.32**	**77.33**	**80.52**	**82.14**	**81.32**	**78.69**	**80.94**	**79.79**	**83.15**	**84.78**	**83.95**

(a) Experiment results of our proposed model for monolingual and multilingual setting

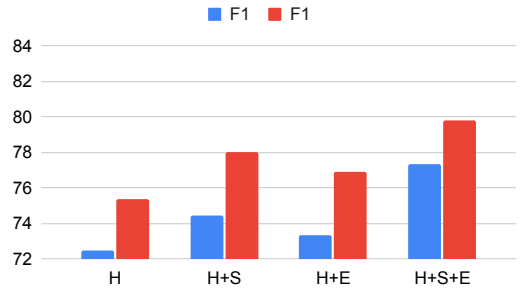

(b) Monolingual (Hindi) vs Multilingual (Hindi)

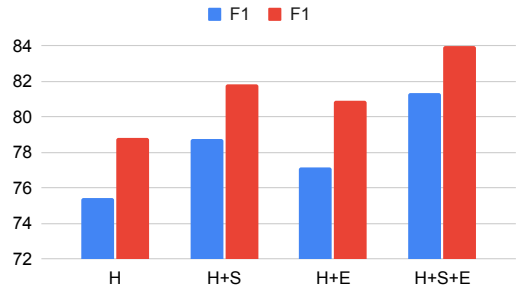

(c) Monolingual (English) vs Multilingual (English)

Table 4: Experiment results and bar-chart representations of our proposed model for monolingual and multilingual setting, where H, S, and E represent the humor, sentiment, and emotion, respectively.

	Monolingual						*Multilingual*					
	Hindi			*English*			*Hindi*			*English*		
Labels	**P**	**R**	**F1**	**P**	**R**	**F1**	**P**	**R**	**F1**	**P**	**R**	**F1**
T	69.61	72.89	71.21	74.64	76.32	75.47	72.54	73.26	72.89	76.87	78.89	77.86
A	59.71	64.89	62.19	59.71	64.89	62.19	59.71	64.89	62.19	59.71	64.89	62.19
V	58.84	60.57	59.61	58.84	60.57	59.61	58.84	60.57	59.61	58.84	60.57	59.61
T+V	73.77	75.13	74.41	77.71	78.94	78.94	75.97	76.21	75.58	78.67	80.11	79.38
T+A	73.61	76.89	75.21	75.72	77.63	76.65	75.54	76.89	76.20	81.11	82.31	81.70
A+V	70.14	72.36	70.24	70.14	72.36	70.24	70.14	72.36	70.24	70.14	72.36	70.24
T+V+A	**76.37**	**78.32**	**77.33**	**80.52**	**82.14**	**81.32**	**78.69**	**80.94**	**79.79**	**83.15**	**84.78**	**83.95**

Table 5: Modality-wise results of our proposed tritask model (H+S+E) for monolingual and multilingual setting, where T, A, and V represent the text, acoustic, and visual, respectively.

sion, 5.5 points in recall, and 5.9 points in F1-score over unitask.

6.2 Multilingual (Multitask vs. Single-task)

In multilingual experiments, we evaluate our proposed model in both languages (i.e., Hindi and English). We perform similar experiments as monolingual setup, i.e., unitask (Humor), bitask (Humor+Sentiment and Humor+Emotion), and tritask (Humor+Sentiment+Emotion). We show experimental results in Table 4. For Hindi tritask (H+S+E), our proposed model shows an improvement of 2.35 points in precision, 3.45 points in recall, and 2.88 points in F1-score over bitask[9] while 3.85 points in precision, 5.05 points in recall, and 4.43 points in F1-score over unitask.

We get a similar improvement for English tritask (H+S+E). For English tritask (H+S+E), our proposed model shows an improvement of 2.66 points in precision, 3.41 points in recall, and 3.03 points in F1-score over bitask[10] while 4.83 points in precision, 5.53 points in recall, and 5.17 points in F1-score over unitask.

For both monolingual and multilingual setups, we observe that tritask (H+S+E) performance is better than unitask and bitask for Hindi and English. Thus, we can say that sentiment and emotion help to understand humor. We also show the modality-wise results of our proposed tritask model (H+S+E) for monolingual and multilingual setting in Table 5.

7 Comparative Analysis

We compare the results obtained from our proposed model against the existing model M2H2 (Chauhan et al., 2021). We evaluate our proposed framework with all the possible combinations i.e., unitask *(H)*,

[9]We get a maximum improvement over H+E

[10]We get a maximum improvement over H+E

	Monolingual											
	Hindi						*English*					
	M2H2(2021)			*Proposed*			*M2H2(2021)*			*Proposed*		
Labels	**P**	**R**	**F1**	**P**	**R**	**F1**	**P**	**R**	**F1**	**P**	**R**	**F1**
H	71.21	72.11	71.67	**71.82**	**73.91**	**72.47**	72.33	74.42	73.35	**74.89**	**76.64**	**75.42**
H+S	73.31	74.21	73.92	**73.42**	**74.91**	**74.46**	77.51	77.79	77.64	**77.63**	**79.32**	**78.74**
H+E	71.11	72.91	72.41	**72.14**	**74.22**	**73.35**	75.13	75.94	75.73	**76.89**	**77.68**	**77.12**
H+S+E	74.71	75.91	75.31	**76.37**	**78.32**	**77.33**	78.71	79.81	79.25	**80.52**	**82.14**	**81.32**

Table 6: Comparative analysis: comparison between M2H2 (2021) and our proposed model in monolingual setting

	Multilingual											
	Hindi						*English*					
	M2H2(2021)			*Proposed*			*M2H2(2021)*			*Proposed*		
Labels	**P**	**R**	**F1**	**P**	**R**	**F1**	**P**	**R**	**F1**	**P**	**R**	**F1**
H	73.47	74.39	73.92	**74.84**	**75.89**	**75.36**	75.21	76.43	75.81	**78.32**	**79.25**	**78.78**
H+S	75.37	76.74	76.04	**77.11**	**78.94**	**78.01**	79.76	80.21	79.98	**81.22**	**82.43**	**81.79**
H+E	74.55	76.21	75.37	**76.34**	**77.49**	**76.91**	77.86	79.48	78.66	**80.49**	**81.37**	**80.92**
H+S+E	76.49	78.61	77.67	**78.69**	**80.94**	**79.79**	81.28	82.49	81.88	**83.15**	**84.78**	**83.95**

Table 7: Comparative analysis: comparison between M2H2 (2021) and our proposed model in multilingual setting

bitask (*H+S and H+E*) and tritask (*H+S+E*).

Please note that as the presented model in M2H2 (2021) was the only unitask (H) model without multilingual, so we made some changes in the model to make it a multitask model and multilingual. Thereafter we obtain the results for unitask (H), bitask (H+S and H+E) and tritask (H+S+E) for both monolingual and multilingual which are also shown in Table 6 and Table 7. We divide this into two parts; i) monolingual and ii) multilingual;

7.1 Monolingual SOTA vs. Proposed

For Hindi, our the proposed tritask model achieves the best precision of 76.37% (1.66 points ↑), recall of 78.32% (2.41 points ↑) and F1-score of 77.33% (2.02 points ↑) compared to precision of 74.71%, recall of 75.91%, and F1-score of 75.31% of the M2H2 (2021). We observe that our proposed model performs better than the state-of-the-art system, M2H2 (2021). We show the results in Table 6.

Similarly, for English, our the proposed tritask model achieves the best precision of 80.52% (1.81 points ↑), recall of 82.14% (2.33 points ↑) and F1-score of 81.32% (2.07 points ↑) compared to precision of 78.71%, recall of 79.81%, and F1-score of 79.25% of the M2H2 (2021). We observe that our proposed model performs better than the state-of-the-art system, M2H2 (2021).

7.2 Multilingual SOTA vs. Proposed

Similar to monolingual, we observe a similar trend of improvement for multilingual experiments. We show the multilingual experiment results in Table 7. For Hindi, our the proposed tritask model achieves the best precision of 78.69% (2.2 points ↑), recall of 80.94% (2.33 points ↑) and F1-score of 79.79% (2.12 points ↑) compared to precision of 76.49%, recall of 78.61%, and F1-score of 77.67% of the M2H2 (2021). We observe that our proposed model performs better than the state-of-the-art system, M2H2 (2021).

Similarly, for English, our the proposed tritask model achieves the best precision of 83.15% (1.87 points ↑), recall of 84.78% (2.29 points ↑) and F1-score of 83.95% (2.07 points ↑) compared to precision of 81.28%, recall of 82.49%, and F1-score of 81.88% of the M2H2 (2021). We observe that our proposed model performs better than the state-of-the-art system, M2H2 (2021).

8 Ablation Study

We perform an ablation study to show the efficacy of *SE-Embedding* for both monolingual and multilingual. For monolingual, we perform experiments with and without *SE-Embedding*. We show the ablation experimental results in Table 8. As per the result, we can see when *SE-Embedding* is used with models then it shows significant improvement rather than without *SE-Embedding*.

Similarly, for multilingual, we perform experiments with and without *SE-Embedding*. As per the result (c.f. Table 8), we can see when *SE-Embedding* is used with models then it shows significant improvement rather than without *SE-Embedding*.

	Monolingual						*Multilingual*					
	Hindi			*English*			*Hindi*			*English*		
Setup	**P**	**R**	**F1**	**P**	**R**	**F1**	**P**	**R**	**F1**	**P**	**R**	**F1**
w/o SE-Embedding	74.23	77.89	76.01	78.85	80.31	79.42	76.90	78.74	77.80	81.31	82.44	81.87
w/ SE-Embedding	**76.37**	**78.32**	**77.33**	**80.52**	**82.14**	**81.32**	**78.69**	**80.94**	**79.79**	**83.15**	**84.78**	**83.95**

Table 8: Ablation study: comparison between proposed tritask model (H+S+E) w/ *SE-Embedding* and w/o *SE-Embedding*

	Hindi Utterances	**English Utterances**	**True Label**	**M2H2 (2021) (H+S+E) (Multilingual)**	**Proposed (H+S+E) (Multilingual)**
1	ये लोग मुझे बहुत मार रहे हैं।	These guys are beating me hard	H	NH	NH,Ng,Sd
2	हाय, टोटो!	hey, Toto!	NH	H	NH,Nu,Nu
3	ये पिंकी लाई।	Pinky brought this	H	NH	H,Nu,Nu
4	अरे भाई, मेरी असली वाली गन यहीं छूट गैइ।	Hey brother, my original gun is left here	NH	H	NH,Nu,Nu
5	कौन सी असली है कौन सी नकली है।	which is real which is fake	NH	H	NH,Nu,Ha

Table 9: Error analysis: some correct and incorrect predicted samples by our proposed tritask model (H+S+E) in multilingual setting.

9 Error Analysis

We perform a detailed analysis to realize the model's strengths and weaknesses. To better analyze and justify our proposed model in terms of quality, we use the same samples (for multilingual setup), which were wrongly predicted and depicted in the paper M2H2 (2021). We show these samples in Table 9. The main motivation for taking the same samples for error analysis is to show the performance of our proposed model over state-of-the-art system. We see that our proposed model predicts all the utterances correctly except the first utterance.

Figure 2: The visual frame of Dilruba for saying "ये लोग मुझे बहुत मार रहे हैं। (These guys are beating me hard)", which shows the sad visual expression

We observe that for the first utterance "ये लोग मुझे बहुत मार रहे हैं। (These guys are beating me hard)", our proposed model predict negative sentiment and sad emotion because of the word "beating" and Dilruba's facial expression (c.f. Figure 2) and sad tone. Also, the context utterances of that utterance are negative. So, this is the reason that our proposed model wrongly predict this utterance.

10 Conclusion

In this paper, we have proposed an effective deep learning-based multitask model for humor detection (primary task) with auxiliary tasks, i.e., sentiment and emotion, in a multilingual conversational setting. As there was no suitable data available for this problem, we first extend recently released M2H2 dataset by adding parallel English utterances corresponding to Hindi utterances and made it a multilingual dataset. Then, we annotate each utterance with sentiment and emotion classes. We have proposed a multitasking framework wherein we propose a *Context Transformer* to capture the deep contextual relationships with the input utterances. We have also successfully established a *Sentiment and Emotion Embedding* (SE-Embedding) which gets the overall representation of a particular emotion and sentiment *w.r.t.* the specific humor situation. Experimental results on the *SHEMuD* have shown that the multitask system yields better performance over the single-task framework.

In the future, we would like to extend our work towards the multiparty dialogue generation in Hindi with the help of humor, sentiment, emotion and speaker information.

11 Ethical Declaration

The *SHEMuD* dataset is freely available and will be used only for the purpose of academic research. We

create *SHEMuD* by extending the M2H2 dataset with English utterances and labeling each utterance with sentiment and emotion classes. The annotation for extending the dataset was done by human experts, who are the regular employee of our research group. There are no other issues to declare.

Acknowledgement

Dushyant Singh Chauhan acknowledges the support of the Prime Minister Research Fellowship (PMRF), Govt. of India. Asif Ekbal acknowledges the Young Faculty Research Fellowship (YFRF), supported by Visvesvaraya Ph.D. Scheme for Electronics and IT, Ministry of Electronics and Information Technology (Meit/8Y), Government of India, being implemented by Digital India Corporation (formerly Media Lab Asia).

References

Md Shad Akhtar, Dushyant Singh Chauhan, Deepanway Ghosal, Soujanya Poria, Asif Ekbal, and Pushpak Bhattacharyya. 2019. Multi-task learning for multi-modal emotion recognition and sentiment analysis. *arXiv preprint arXiv:1905.05812*.

Salvatore Attardo, Lucy Pickering, and Amanda Baker. 2011. Prosodic and multimodal markers of humor in conversation. *Pragmatics & Cognition*, 19(2):224–247.

Dushyant Singh Chauhan, Md Shad Akhtar, Asif Ekbal, and Pushpak Bhattacharyya. 2019. Context-aware interactive attention for multi-modal sentiment and emotion analysis. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 5651–5661, Hong Kong, China. Association for Computational Linguistics.

Dushyant Singh Chauhan, Asif Ekbal, and Pushpak Bhattacharyya. 2022. An efficient fusion mechanism for multimodal low-resource setting. In *Proceedings of the 45th International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '22, page 2583–2588, New York, NY, USA. Association for Computing Machinery.

Dushyant Singh Chauhan, Dhanush S R, Asif Ekbal, and Pushpak Bhattacharyya. 2020a. All-in-one: A deep attentive multi-task learning framework for humour, sarcasm, offensive, motivation, and sentiment on memes. In *Proceedings of the 1st Conference of the Asia-Pacific Chapter of the Association for Computational Linguistics and the 10th International Joint Conference on Natural Language Processing*, pages 281–290, Suzhou, China. Association for Computational Linguistics.

Dushyant Singh Chauhan, Dhanush S R, Asif Ekbal, and Pushpak Bhattacharyya. 2020b. Sentiment and emotion help sarcasm? a multi-task learning framework for multi-modal sarcasm, sentiment and emotion analysis. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 4351–4360, Online. Association for Computational Linguistics.

Dushyant Singh Chauhan, Gopendra Vikram Singh, Navonil Majumder, Amir Zadeh, Asif Ekbal, Pushpak Bhattacharyya, Louis-philippe Morency, and Soujanya Poria. 2021. M2h2: A multimodal multiparty hindi dataset for humor recognition in conversations. In *Proceedings of the 2021 International Conference on Multimodal Interaction*, pages 773–777.

Alexis Conneau, Kartikay Khandelwal, Naman Goyal, Vishrav Chaudhary, Guillaume Wenzek, Francisco Guzmán, Edouard Grave, Myle Ott, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Unsupervised cross-lingual representation learning at scale. *arXiv preprint arXiv:1911.02116*.

Nabiha El Khatib. 2020. *Multimodal Processing of Humorous Tweets*. Ph.D. thesis, Texas A&M University-Commerce.

Florian Eyben, Martin Wöllmer, and Björn Schuller. 2010. Opensmile: the munich versatile and fast open-source audio feature extractor. In *Proceedings of the 18th ACM international conference on Multimedia*, pages 1459–1462. ACM.

Fallianda Fallianda, Rani Yuni Astiti, and Zulvy Alivia Hanim. 2018. Analyzing humor in newspaper comic strips using verbal-visual analysis. *Lingua Cultura*, 12(4):383–388.

Joseph L. Fleiss. 1971. Measuring nominal scale agreement among many rater. *Psychological Bulletin*, 76:378–382.

Deepanway Ghosal, Md Shad Akhtar, Dushyant Singh Chauhan, Soujanya Poria, Asif Ekbal, and Pushpak Bhattacharyya. 2018. Contextual inter-modal attention for multi-modal sentiment analysis. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 3454–3466, Brussels, Belgium. Association for Computational Linguistics.

Md Kamrul Hasan, Wasifur Rahman, Amir Zadeh, Jianyuan Zhong, Md Iftekhar Tanveer, Louis-Philippe Morency, et al. 2019. Ur-funny: A multimodal language dataset for understanding humor. *arXiv preprint arXiv:1904.06618*.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. 2016. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778.

Alex Kendall, Yarin Gal, and Roberto Cipolla. 2018. Multi-task learning using uncertainty to weigh losses for scene geometry and semantics. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 7482–7491.

Nicole Mirnig, Gerald Stollnberger, Manuel Giuliani, and Manfred Tscheligi. 2017. Elements of humor: How humans perceive verbal and non-verbal aspects of humorous robot behavior. In *Proceedings of the Companion of the 2017 ACM/IEEE International Conference on Human-Robot Interaction*, pages 211–212.

Mathieu Pagé Fortin and Brahim Chaib-draa. 2019. Multimodal multitask emotion recognition using images, texts and tags. In *Proceedings of the ACM Workshop on Crossmodal Learning and Application*, pages 3–10. ACM.

Anna Piata. 2020. Stylistic humor across modalities: The case of classical art memes. *Internet Pragmatics*, 3(2):174–201.

Hannes Ritschel, Ilhan Aslan, David Sedlbauer, and Elisabeth André. 2019. Irony man: augmenting a social robot with the ability to use irony in multimodal communication with humans.

Hannes Ritschel, Thomas Kiderle, Klaus Weber, Florian Lingenfelser, Tobias Baur, and Elisabeth André. 2020. Multimodal joke generation and paralinguistic personalization for a socially-aware robot. In *International Conference on Practical Applications of Agents and Multi-Agent Systems*, pages 278–290. Springer.

James A Russell and Lisa Feldman Barrett. 1999. Core affect, prototypical emotional episodes, and other things called emotion: dissecting the elephant. *Journal of personality and social psychology*, 76(5):805.

Andy Jefferson Sabur, Retno Purwani Sari, and Tatan Tawami. 2020. Identification of the multimodal structure of humor in an animated superhero film. In *International Conference on Language, Linguistics, and Literature (COLALITE) 2020*.

Martha Schneider, Martin Voracek, and Ulrich S Tran. 2018. "a joke a day keeps the doctor away?" meta-analytical evidence of differential associations of habitual humor styles with mental health. *Scandinavian journal of psychology*, 59(3):289–300.

Kwangok Song, Kyle M Williams, Diane L Schallert, and Alina Adonyi Pruitt. 2021. Humor in multimodal language use: Students' response to a dialogic, social-networking online assignment. *Linguistics and Education*, 63:100903.

Camilla Vasquez and Erhan Aslan. 2021. "cats be outside, how about meow": multimodal humor and creativity in an internet meme. *Journal of Pragmatics*, 171:101–117.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in neural information processing systems*, pages 5998–6008.

Zhigailova Veronika. 2020. Multimodal discourse analysis of humor in picture books for children.

Caleb Warren, Adam Barsky, and A Peter McGraw. 2018. Humor, comedy, and consumer behavior. *Journal of Consumer Research*, 45(3):529–552.

Saining Xie, Ross Girshick, Piotr Dollár, Zhuowen Tu, and Kaiming He. 2017. Aggregated residual transformations for deep neural networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1492–1500.

Zixiaofan Yang, Lin Ai, and Julia Hirschberg. 2019. Multimodal indicators of humor in videos. In *2019 IEEE Conference on Multimedia Information Processing and Retrieval (MIPR)*, pages 538–543. IEEE.

TSAM: A Two-Stream Attention Model for Causal Emotion Entailment

Duzhen Zhang[*], **Zhen Yang, Fandong Meng, Xiuyi Chen**[†], **Jie Zhou**
Pattern Recognition Center, WeChat AI, Tencent Inc, Beijing, China
{bladedancer957,hugheren.chan}@gmail.com
{zieenyang,fandongmeng,withtomzhou}@tencent.com

Abstract

Causal Emotion Entailment (CEE) aims to discover the potential causes behind an emotion in a conversational utterance. Previous works formalize CEE as independent utterance pair classification problems, with emotion and speaker information neglected. From a new perspective, this paper considers CEE in a joint framework. We classify multiple utterances synchronously to capture the correlations between utterances in a global view and propose a Two-Stream Attention Model (TSAM) to effectively model the speaker's emotional influences in the conversational history. Specifically, the TSAM comprises three modules: Emotion Attention Network (EAN), Speaker Attention Network (SAN), and interaction module. The EAN and SAN incorporate emotion and speaker information in parallel, and the subsequent interaction module effectively interchanges relevant information between the EAN and SAN via a mutual BiAffine transformation. Extensive experimental results demonstrate that our model achieves new State-Of-The-Art (SOTA) performance and outperforms baselines remarkably.

1 Introduction

With the recent proliferation of open conversational data on social media platforms, such as Twitter and Facebook, Emotion Analysis in Conversations (EAC) has become a popular research topic in the field of Natural Language Processing (NLP). Most of the existing works on EAC mainly focus on Emotion Recognition in Conversations (ERC), i.e., recognizing emotion labels of utterances (e.g., happy, sad, etc.) (Poria et al., 2017, 2019b; Wang et al., 2020; Zhang et al., 2020). However, Poria et al. (2021) point out that these studies lack further reasoning about emotions, such as understanding the stimuli and the cause of the emotion. Since Recognizing Emotion Cause in Conversations (RECCON) holds the potential to improve the interpretability and performance of affect-based models, Poria et al. (2021) put forward a new promising task, named RECCON, which includes two different sub-tasks: Causal Span Extraction (CSE) at word/phrase level and Causal Emotion Entailment (CEE) at utterance level. Due to the simplicity and sufficiency describing emotion causes at the utterance level, we focus on the CEE task in this paper, whose goal is to predict which particular utterances in the conversational history contain the cause of non-neutral emotion in the target utterance.

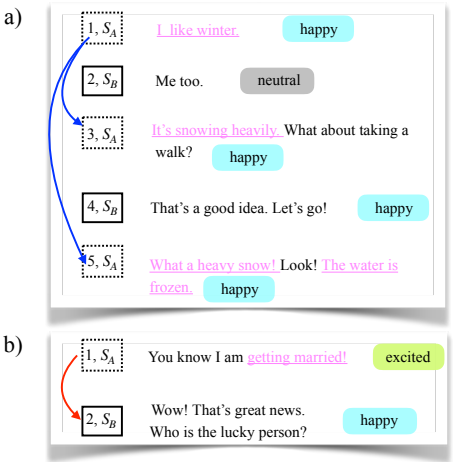

Figure 1: Example conversations sampled from the benchmark dataset (Poria et al., 2021)

Compared to the Emotion Cause Extraction (ECE) in news articles (Gui et al., 2016a; Xia and Ding, 2019), CEE is particularly challenging due to the informal expression style and the intermingling dynamic among interlocutors. Poria et al. (2021) consider CEE as a set of independent utterance pair classification problems and neglect the emotion and speaker information in the conversational history. Thus, they can neither capture the correlations between contextual utterances in a global view nor

[*] This work was done when Duzhen Zhang was interning at Pattern Recognition Center, WeChat AI, Tencent Inc, China.
[†] Corresponding author.

Proceedings of the 29th International Conference on Computational Linguistics, pages 6762–6772
October 12–17, 2022.

model the speaker's emotional influences, namely the intra-speaker and inter-speaker emotional influences.[1] Intra-speaker emotional influences mean that the cause of the emotion is primarily due to the speaker's stable mood induced from previous dialogue turns. As shown in Figure 1 (a), utterance 1 establishes the concept that Speaker A (S_A) likes winter, which triggers a happy mood for future utterances 3 and 5. Inter-speaker emotional influences mean that the emotion of the target speaker is induced from an event mentioned or emotion revealed by other speakers. As Figure 1 (b) shows, S_B's happy emotion may be triggered by the event "getting married" mentioned by S_A, or by the fact that S_A is excited about getting married.

To remedy this defect, we tackle CEE in a joint framework. We classify multiple utterances synchronously to capture the correlations between contextual utterances and propose a TSAM to effectively model the speaker's emotional influences in the conversational history. Specifically, the TSAM contains three modules: EAN, SAN, and interaction module. The EAN provides utterance-to-emotion interactions to incorporate emotion information by performing attention over emotion embeddings. The SAN represents different speaker relations between utterances in a graph, which provides utterance-to-utterance interactions to incorporate speaker information by performing attention over the speaker relation graph. These two modules incorporate emotion and speaker information in parallel. Moreover, inspired by (Li et al., 2021a; Tang et al., 2020), the interaction module interchanges relevant information between the EAN and SAN through a mutual BiAffine transformation. Finally, the entire TSAM can be stacked in multiple layers to refine iteratively and interchange emotion and speaker information.

- For the first time, we tackle CEE in a joint framework to capture the correlations between contextual utterances in a global view.
- We propose a TSAM to model the speaker's emotional influences in the conversational history, which contains EAN, SAN, and interaction module to incorporate and interchange emotion and speaker information.
- Experimental results on the benchmark dataset (Poria et al., 2021) demonstrate the effectiveness of our proposed model, surpassing the SOTA model significantly.

[1]The speaker's emotional influences are predominant types of emotion causes in the dataset as shown in Table 1.

2 Related Work

ECE Early works mainly exploit rule-based methods (Lee et al., 2010a,b; Chen et al., 2010) to identify the potential causes for certain emotion expressions in the text. Gui et al. (2016a) first release a public annotated dataset for ECE, and based on which some feature based (Gui et al., 2016b) and neural based methods (Gui et al., 2017; Li et al., 2018; Ding et al., 2019; Xia et al., 2019; Yan et al., 2021; Li et al., 2021b) appear. To extract emotion and its corresponding cause jointly, Xia and Ding (2019) first put forward the Emotion-Cause Pair Extraction (ECPE) task and tackle it by a two-step method. Subsequently, many improved methods are proposed to tackle ECPE in an end2end manner (Ding et al., 2020a,b; Yuan et al., 2020; Fan et al., 2020; Wei et al., 2020; Cheng et al., 2020; Chen et al., 2020a,b). However, these works mentioned above use news articles as the target corpus for ECE, which largely reduces reasoning complexity. By contrast, CEE is more challenging due to the intermingling dynamic among interlocutors and the informal expression style.

ERC Recently, due to the proliferation of publicly available conversational datasets (Zhou et al., 2018; Chen et al., 2019; Poria et al., 2019a; Chatterjee et al., 2019; Xie et al., 2022), there is a growing number of studies on ERC (Hazarika et al., 2018a,b; Majumder et al., 2019; Zhong et al., 2019; Jiao et al., 2019; Ghosal et al., 2020b; Ishiwatari et al., 2020; Ghosal et al., 2020a; Shen et al., 2021; Zhu et al., 2021; Hu et al., 2021; Guibon et al., 2021; Zhao et al., 2022; Peng et al., 2022). Although substantial progress has been made in ERC, these studies lack further reasoning about emotions, such as understanding the stimuli or the cause of an emotion expressed by a speaker (Poria et al., 2021).

RECCON For further reasoning about emotions, Poria et al. (2021) propose a new task named RECCON, which contains two different sub-tasks: CSE at word/phrase level and CEE at utterance level. Poria et al. (2021) formalize CEE as a set of independent utterance pair classification problems, neglecting the emotion and speaker information in the conversational history. Specifically, they pair a target utterance with each utterance in its conversational history and determine whether the utterance

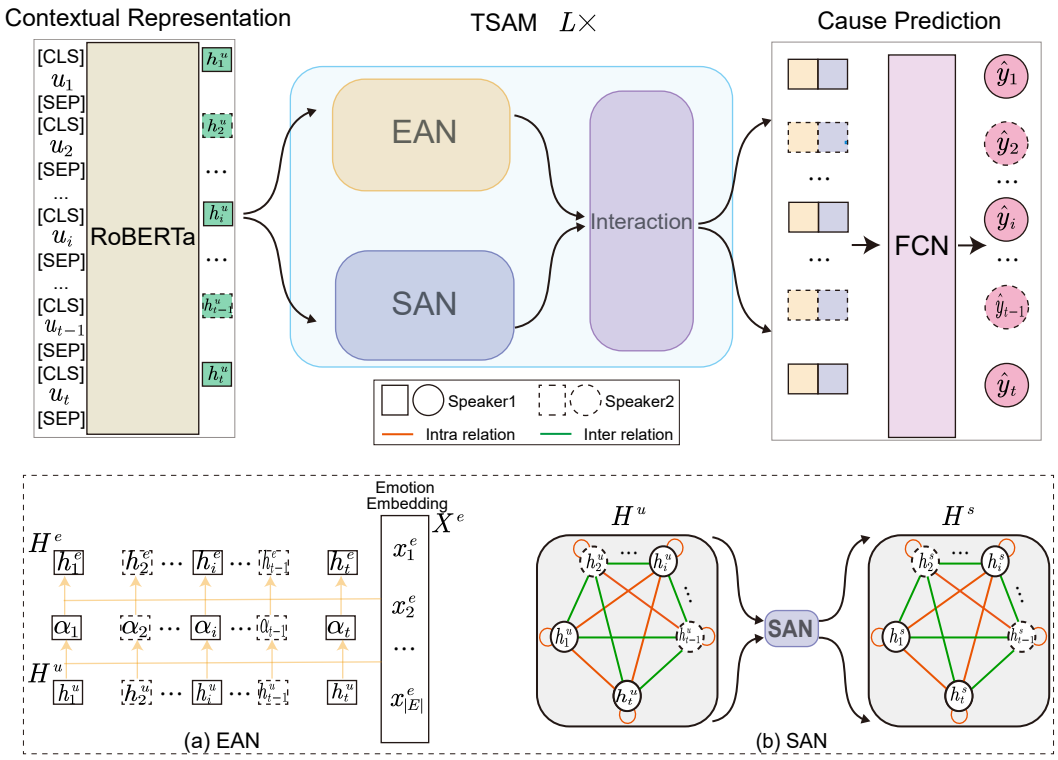

Figure 2: The top is the proposed model's entire architecture, and the bottom is the detailed architecture of model components: (a) EAN, (b) SAN. First, we obtain the contextual representation for each utterance with RoBERTa. Then, the TSAM is utilized to model the speaker's emotional influences in the conversational history. Finally, the cause prediction module is used to output the predictions.

contains the cause of emotion in the target utterance. Thus, they cannot capture the correlations between contextual utterances in a global view and fail to model the speaker's emotional influences in the conversational history. From a new perspective, we tackle CEE in a joint framework. We encode and classify multiple utterances synchronously to capture the correlations in a global view and propose a TSAM to model the speaker's emotional influences effectively.

3 Task Definition

We first define the task of CEE formally. For a target utterance u_t, i.e., the tth utterance in the conversation, the goal of CEE is to predict which particular utterances in the conversational history $L(u_t) = (u_1, u_2, ..., u_t)$ are responsible for the non-neutral emotion in the target utterance. u_i is set as a positive example if it contains the cause of non-neutral emotion in the target utterance and a negative example otherwise, where $i = 1, ..., t$. The independent utterance pair classification framework (Poria et al., 2021) performs t independent classifications, each of which takes (u_t, u_i) as input. Therefore, it fails to capture the correlations between contextual utterances in a global view. On the contrary, the proposed joint classification framework only performs one joint classification with $L(u_t)$ as input, which makes it possible to capture the correlations between contextual utterances.

4 Method

The proposed model consists of three components: the contextual utterance representation, the TSAM, and the cause prediction modules. The whole architecture of our model is illustrated in Figure 2.

4.1 Contextual Utterance Representation

The pre-trained RoBERTa is utilized as the utterance encoder, and we extract the contextual utterance representations by feeding the whole of the conversational history $L(u_t)$ into the RoBERTa (Liu et al., 2019). Specifically, each utterance in $L(u_t)$ is expanded to start with the token "[CLS]" and end with the token "[SEP]". The input representation for each token is the sum of its corresponding token and position embeddings. The contextual representation $\boldsymbol{h}_i^u \in \mathbb{R}^{d_h}$ for utterance u_i is the output

of the corresponding "[CLS]" token, where d_h denotes the dimension of the utterance representation. The contextual representation for all utterances is denoted as $\boldsymbol{H}^u \in \mathbb{R}^{t \times d_h}$. The RoBERTa we utilized is fine-tuned with the training process.

4.2 TSAM

The TSAM models the speaker's emotional influences with three modules: EAN, SAN, and Interaction. We first illustrate the calculation process of each module in one-layer TSAM and then generalize it to multiple successive layers.

4.2.1 EAN

The EAN provides utterance-to-emotion interactions to explicitly incorporate emotion information by performing attention over emotion embeddings.

Emotion Representation Given the set of candidate emotion labels $\mathcal{E} = \{e_1, ..., e_{|\mathcal{E}|}\}$, each emotion label e_k is represented using an embedding vector (Cui and Zhang, 2019):

$$\boldsymbol{x}_k^e = \text{Embed}(e_k) \in \mathbb{R}^{d_h} \tag{1}$$

where $k = 1, ..., |\mathcal{E}|$, d_h denotes the dimension of the emotion embedding. Embed represents an emotion embedding lookup table, which is initialized by contextual embeddings from RoBERTa and tuned during model training. The embedding for the set of the whole emotion labels is denoted as $\boldsymbol{X}^e \in \mathbb{R}^{|\mathcal{E}| \times d_h}$.

EAN Inference With the emotion labels represented as embeddings, we extract the emotion information $\boldsymbol{H}^e \in \mathbb{R}^{t \times d_h}$ by performing dot-product attention over contextual utterance representations and emotion embeddings, which is calculated as:

$$\boldsymbol{H}^e = \textit{attention}(\boldsymbol{Q}, \boldsymbol{K}, \boldsymbol{V}) = \boldsymbol{\alpha}\boldsymbol{V} \tag{2}$$

$$\boldsymbol{\alpha} = \textit{softmax}(\frac{\boldsymbol{Q}\boldsymbol{K}^T}{\sqrt{d_h}}) \tag{3}$$

where $\boldsymbol{Q} = \boldsymbol{H}^u, \boldsymbol{K} = \boldsymbol{V} = \boldsymbol{X}^e$, $\boldsymbol{\alpha} \in \mathbb{R}^{t \times |\mathcal{E}|}$ is an attention matrix consisting of potential emotion distributions for all utterances. Compared to the standard attention mechanism above, it may be beneficial to use multi-head attention (Vaswani et al., 2017) to capture multiple emotion distributions in parallel and obtain richer emotion information:

$$\boldsymbol{H}^e = \textit{concat}(head_1, ..., head_m) \tag{4}$$

$$head_j = \textit{attention}(\boldsymbol{Q}\boldsymbol{W}_j^Q, \boldsymbol{K}\boldsymbol{W}_j^K, \boldsymbol{V}\boldsymbol{W}_j^V) \tag{5}$$

where $\boldsymbol{W}_j^Q, \boldsymbol{W}_j^K, \boldsymbol{W}_j^V \in \mathbb{R}^{d_h \times \frac{d_h}{m}}$ are learnable parameters and m is the number of parallel heads.

Since the emotion labels of the utterances in the conversational history are known, we can also simply use the embedding of emotion label corresponding to the utterance as the extracted emotion information:

$$\boldsymbol{H}^e = \tilde{\boldsymbol{X}}^e \tag{6}$$

where $\tilde{\boldsymbol{X}}^e \in \mathbb{R}^{t \times d_h}$ is the embedding of emotion labels corresponding to all utterances in the history. We refer to the method as Direct Application Emotional Embedding (DAEE). Compared with DAEE, the potential advantages of the EAN are as follows: (1) The EAN can provide utterance-to-emotion interactions and capture multiple potential emotion distributions through multi-head attention to obtain more comprehensive and richer emotion information; (2) The soft emotion distributions can model the mutual impact among different emotions for further enhancement of emotion embeddings, while each emotion embedding is relatively independent of each other in DAEE; (3) The EAN can avoid emotion annotation errors to a certain extent. We apply EAN in our model to incorporate emotion information by default and compare the EAN and DAEE in the part of experiments.

4.2.2 SAN

The SAN provides utterance-to-utterance interactions to incorporate speaker information by performing attention over the speaker relation graph.

Graphical Structure We define a conversational history with t utterances as a graph $\mathcal{G} = (\mathcal{V}, \mathcal{E}, \mathcal{R})$, with nodes (utterances) $v_i \in \mathcal{V}$ and labeled edges (relations) $(v_i, r, v_j) \in \mathcal{E}$, where $r \in \mathcal{R}$ is a relation type. We also add a self-loop edge to every node, as the cause may be present within the target utterance itself. The representation of node v_i is initialized with the contextual utterance representation $\boldsymbol{h}_i^u \in \mathbb{R}^{d_h}$, i.e., the ith embedding in $\boldsymbol{H}^u$. There are two relation types of edges: (1) **Intra** relation type: how the utterance influences other utterances (including itself) expressed by the same speaker; (2) **Inter** relation type: how the utterance influences ones expressed by other speakers.

SAN Inference The representation of a node $\boldsymbol{h}_i$ is updated by aggregating representations of its neighborhood $\mathcal{N}^r(i)$ under the relation type r. The graph attention mechanism (Veličković et al., 2018)

is used to attend to the neighborhood's representations. The output of a node $\boldsymbol{h}_i^s \in \mathbb{R}^{d_h}$ is calculated as the sum of the hidden features $\boldsymbol{h}_{ir} \in \mathbb{R}^{d_h}$ under relation r. The propagation is defined as follows:

$$\alpha_{ijr} = \textit{softmax}_i(\textit{LRL}(\boldsymbol{a}_r^T[\boldsymbol{W}_r\boldsymbol{h}_i^u; \boldsymbol{W}_r\boldsymbol{h}_j^u])) \quad (7)$$

$$\boldsymbol{h}_{ir} = \sum_{j \in \mathcal{N}^r(i)} \alpha_{ijr}\boldsymbol{W}_r\boldsymbol{h}_j^u \quad (8)$$

$$\boldsymbol{h}_i^s = \sum_{r \in \mathcal{R}} \boldsymbol{h}_{ir} \quad (9)$$

where α_{ijr} denotes the edge weight from utterance u_i to its neighborhood u_j under relation type r, $\boldsymbol{W}_r \in \mathbb{R}^{d_h \times d_h}$ and $\boldsymbol{a}_r \in \mathbb{R}^{d_h}$ denote a learnable weight matrix and a vector for each relation type r respectively. *LRL* denotes *LeakyReLU* activation function. The updated representation of all nodes is denoted as $\boldsymbol{H}^s \in \mathbb{R}^{t \times d_h}$.

4.2.3 Interaction Module

To effectively interchange relevant information between the EAN and SAN, we apply a mutual Bi-Affine transformation as a bridge. The calculation process is formulated as follows:

$$\boldsymbol{A}_1 = \textit{softmax}(\boldsymbol{H}^e\boldsymbol{W}_1(\boldsymbol{H}^s)^T) \quad (10)$$

$$\boldsymbol{A}_2 = \textit{softmax}(\boldsymbol{H}^s\boldsymbol{W}_2(\boldsymbol{H}^e)^T) \quad (11)$$

$$\boldsymbol{H}^{e'} = \boldsymbol{A}_1\boldsymbol{H}^s \quad (12)$$

$$\boldsymbol{H}^{s'} = \boldsymbol{A}_2\boldsymbol{H}^e \quad (13)$$

where $\boldsymbol{W}_1, \boldsymbol{W}_2 \in \mathbb{R}^{d_h \times d_h}$ are trainable parameters and $\boldsymbol{A}_1, \boldsymbol{A}2 \in \mathbb{R}^{t \times t}$ are temporary alignment matrices projecting from $\boldsymbol{H}^s$ to $\boldsymbol{H}^e$ and $\boldsymbol{H}^e$ to $\boldsymbol{H}^s$, respectively. Here, $\boldsymbol{H}^{e'} \in \mathbb{R}^{t \times d_h}$ can be viewed as a projection from $\boldsymbol{H}^s$ to $\boldsymbol{H}^e$, and $\boldsymbol{H}^{s'} \in \mathbb{R}^{t \times d_h}$ follows the same principle.

4.2.4 The Whole Process

We generalize the TSAM to multiple successive layers to iteratively refine and interchange emotion and speaker information. The detailed procedures are as follows:

$$\boldsymbol{H}_l^e = \textbf{EAN}(\boldsymbol{E}_l, \boldsymbol{X}^e) \quad (14)$$

$$\boldsymbol{H}_l^s = \textbf{SAN}(\boldsymbol{S}_l) \quad (15)$$

$$\boldsymbol{H}_l^{e'}, \boldsymbol{H}_l^{s'} = \textbf{Interaction}(\boldsymbol{H}_l^e, \boldsymbol{H}_l^s) \quad (16)$$

$$\boldsymbol{E}_{l+1}, \boldsymbol{S}_{l+1} = \boldsymbol{H}_l^{e'}, \boldsymbol{H}_l^{s'} \quad (17)$$

where $\boldsymbol{E}_0 = \boldsymbol{S}_0 = \boldsymbol{H}^u$. The TSAM can be stacked in L layers and $l \in [0, L-1]$.

Statistics			RECCON-DD
Data Distributions	Train	Positive	7269
		Negative	20646
	Dev	Positive	347
		Negative	838
	Test	Positive	1894
		Negative	5330
Cause Type Distributions	No Context		43%
	Inter		32%
	Intra		9%
	Hybrid		11%
	Unmentioned		5%

Table 1: Statistics of the RECCON-DD dataset. *No Context*: The cause is present within the target utterance itself; *Inter*: Inter-speaker emotional influences; *Intra*: Intra-speaker emotion influences (Self-Contagion); *Hybrid*: Inter and Intra can jointly cause the emotion of an utterance; *Unmentioned*: Some instances have no explicit emotion causes in the conversational history.

4.3 Cause Prediction

We obtain the final utterance representation for u_i by concatenating the output $(\boldsymbol{E}_L, \boldsymbol{S}_L)$ of the L-layer TSAM. Finally, the concatenated vector is classified using a Fully-Connected Network (FCN):

$$\boldsymbol{l}_i = \textit{ReLU}(\boldsymbol{W}_1[\boldsymbol{e}_i^L; \boldsymbol{s}_i^L] + \boldsymbol{b}_1) \quad (18)$$

$$\hat{y}_i = \textit{sigmoid}(\boldsymbol{W}_2\boldsymbol{l}_i + b_2) \quad (19)$$

where $\hat{y}_i$ is the probability for utterance u_i containing the cause of emotion in the target utterance, $\boldsymbol{e}_i^L, \boldsymbol{s}_i^L \in \mathbb{R}^{d_h}$ denote the ith embedding in $\boldsymbol{E}_L$ and $\boldsymbol{S}_L$, respectively, $\boldsymbol{W}_1 \in \mathbb{R}^{d_h \times 2d_h}$, $\boldsymbol{W}_2 \in \mathbb{R}^{1 \times d_h}$, $\boldsymbol{b}_1 \in \mathbb{R}^{d_h}$ and b_2 are learnable parameters of FCN.

5 Experimental Settings

5.1 Dataset and Evaluation Metrics

We evaluate the proposed model on a benchmark dataset for RECCON, named RECCON-DD (Poria et al., 2021), which is constructed based on DailyDialog dataset (Li et al., 2017).[2] Some statistics about RECCON-DD are reported in Table 1. Following (Poria et al., 2021), the macro-averaged F1 score is utilized as the evaluation metric in this paper. We also report the F1 score for both positive and negative samples, denoted as Pos. F1 and Neg. F1 respectively.

[2]DailyDialog is a natural human communication dataset which is usually used in ERC task. It contains utterance-level emotion labels and covers various topics related to daily lives.

#	Model	W/O CH			W/ CH		
		Pos. F1	Neg. F1	macro F1	Pos.F1	Neg.F1	macro F1
0	INDEP$_{base}$	56.64	85.13	70.88	64.28	88.74	76.51
1	INDEP$_{large}$	50.48	87.35	68.91	66.23	87.89	77.06
2	JOINT$_{base}$	-	-	-	66.61	89.11	77.86
3	JOINT$_{large}$	-	-	-	68.30	89.16	78.73
4	Ours$_{base}$	-	-	-	68.59	89.75	79.17
5	Ours$_{large}$	-	-	-	**70.00**†	**90.48**†	**80.24**†

Table 2: Performance of our model and baselines on the test set of RECCON-DD. Bold font denotes the best performance. "Ours" denotes the proposed model without removing any module ("Ours" = "JOINT" + TSAM). "†" denotes that Ours$_{large}$ is statistically significant (Koehn, 2004) better than INDEP$_{large}$ W/ CH (p-value < 0.05).

5.2 Baselines

For a comprehensive performance evaluation, we compare our model with the following baselines:

(1) INDEP$_{base}$ (Poria et al., 2021) It tackles CEE in an independent classification framework (INDEP) and uses the RoBERTa-Base model (Liu et al., 2019) as the utterance encoder. The input is formated as "[CLS]u_t[SEP]u_i[SEP]" and the classification is performed from the final representation of the token "[CLS]".

(2) INDEP$_{large}$ (Poria et al., 2021) Compared to (1), it uses the RoBERTa-Large model as utterance encoder;

(3) JOINT$_{base}$ It's one of the variants of our model, where the TSAM is removed. It tackles RECCON in a joint classification framework (JOINT) and uses the RoBERTa-Base model as the utterance encoder. Moreover, its input format is "[CLS]u_1[SEP][CLS]$u2$[SEP],...,[CLS]u_t[SEP]" and the classifications are performed synchronously from the corresponding contextual utterance representations of the [CLS] tokens;

(4) JOINT$_{large}$ Compared to (3), it uses RoBERTa-Large model as the utterance encoder.

For **INDEP** baselines, there are two different settings: With Conversational History (W/ CH) and Without Conversational History (W/O CH). W/ CH means considering the conversational history. When performing utterance pair classification, the conversational history $L(u_t)$ is concatenated after the input to incorporate contextual information, while W/O CH means ignoring the history.

5.3 Implementation Details

Our model's base and large versions use pre-trained RoBERTa-Base and RoBERTa-Large models as the utterance encoders, respectively.[3] The binary cross-entropy loss along with L2-regularization is used during training, where the coefficient of L2 term is 0.01 in the RoBERTa structure and $1e$-5 in other structures. We set the dropout rate to 0.1. The learning rate and the batch size are set as $1e$-5 and 2, respectively. Our model is trained with the Adam optimizer (Kingma and Ba, 2015). We set the dimensions of the contextual utterance representation d_h as 768/1024 in the Base/Large version of the proposed model. We use 4-head attention in EAN, and the number of TSAM layers L is set to 3. We train the model for 40 epochs in total and use the early stopping strategy based on the performance on the development set. Then, the model with the highest macro-averaged F1 score is used to evaluate the test set. Other hyper-parameters are selected according to the performance of the development set. All of the experiments are conducted on an NVIDIA V100 GPU with 32GB of memory.

6 Results and Discussions

6.1 Main Results

Experimental results are reported in Table 2. We directly cite the results for the baselines reported in (Poria et al., 2021). For the performance of each model we implemented, we report the average score of 5 runs. From Table 2, we can find that the proposed model (#5) outperforms all of the baselines and surpasses the best model (#1, W/ CH) in (Poria et al., 2021) with more than 3 points of macro F1 score.

Further comparisons show that models with the large pre-trained utterance encoder are more likely

[3]Our RoBERTa models are adapted from this implementation: https://github.com/huggingface/transformers

to achieve better performance (about 1 point of macro F1 score) than the corresponding models with the base one, except for the models under W/O CH setting in the Table 2. By comparing two different settings W/O CH and W/ CH in Table 2, we can find that the conversational history plays a significant role for INDEP$_{base}$ and INDEP$_{large}$ models. This is mainly because that the conversational history is able to provide the contextual information for prediction. Due to the simultaneous classification of multiple utterances in the conversational history under the joint framework, JOINT$_{base}$ and JOINT$_{large}$ models can naturally incorporate the contextual information. The JOINT$_{base}$ and JOINT$_{large}$ models significantly outperform the INDEP$_{base}$ W/ CH and INDEP$_{large}$ W/ CH models by about 1.5 points of macro F1 scores respectively (comparing #0 with #2, and #1 with #3 in Table 2). There may be two main factors: 1) Simply concatenating the conversational history after the utterance pair to be classified in INDEP W/ CH models may destroy the structure of the conversation; 2) Compared to INDEP W/ CH models, classification of multiple utterances synchronously in JOINT models will have more sufficient supervision signals and can more effectively model the correlations between contextual utterances in a global view, i.e., utterances with similar semantics are supposed to have similar chances being the emotion cause. The comparison between #2 and #4 (or #3 and #5) in Table 2 shows the effectiveness of the proposed TSAM. The model with TSAM (#5) achieves an improvement up to 1.51 points of macro F1 score than the model without TSAM (#3).

6.2 Ablation Study

In this subsection, we conduct ablation studies to analyze the effects of different components based on Ours$_{large}$ mentioned in Table 2.

Effect of Emotion Information We compare three different ways for incorporating the emotion information: no emotion information incorporated, incorporating emotion information with direct application emotional embedding, and incorporating emotion information with EAN. The results are shown in Table 3. We can find that the performance of the proposed model degrades if the emotion information is not incorporated (comparing row 1 with 3 in Table 3). This result shows that the emotion information in the conversational history plays a significant role in the task of CEE. By comparing rows 2 with 3 in Table 3, the result shows that EAN achieves better performance than DAEE since EAN can extract richer emotion information and model the mutual impact among different emotions.

Emotion Information	Pos. F1	Neg. F1	macro F1
No	68.40	89.80	79.10
DAEE	68.90	90.03	79.47
EAN	**70.00**	**90.48**	**80.24**

Table 3: Comparison of different ways of incorporating emotion information. *No*: no emotion information incorporated; *DAEE*: incorporating the emotion information with direct application emotional embedding.

Effect of Speaker Information To evaluate the effects of speaker information, we remove the speaker relations in SAN, resulting in a single edge relation throughout the graph. As Table 4 shows, the performance of our model decreases dramatically if not considering the speaker information. This result presents that modeling the speaker information in the conversational history is very important for the final performance.

Speaker Information	Pos. F1	Neg. F1	macro F1
Not Consider	67.99	89.42	78.71
Consider	**70.00**	**90.48**	**80.24**

Table 4: Results on experiments whether considering speaker information or not in SAN.

Effect of Interaction Module We remove the interaction module in each layer so that the EAN and SAN can't interact. As Table 5 shows, the performance of our model decreases dramatically when the interaction module is removed. This result shows that the effective interchange of relevant information between EAN and SAN is conducive to the final performance.

	Pos. F1	Neg. F1	macro F1
W/O Interaction	68.18	88.93	78.56
W/ Interaction	**70.00**	**90.48**	**80.24**

Table 5: Results on experiments whether removing interaction module or not in TSAM.

Ability on Modeling Emotional Influences To evaluate the proposed model's ability to model the speaker's emotional influences, we collect the positive examples from the test set where the causes

Models	Intra	Inter
W/O TSAM	62.06	72.67
W/ TSAM	**63.82**	**74.81**

Table 6: Accuracy on the collected samples. *Intra*: Intra-speaker emotional influences; *Inter*: Inter-speaker emotional influences.

are induced from the inter-speaker or intra-speaker emotional influences. And we test the prediction accuracy on the collected samples for the proposed Ours$_{large}$ with and without TSAM. As shown in Table 6, W/ TSAM outperforms W/O TSAM by around 2 points on both cause types, which further verifies that the TSAM can effectively model the emotional influences between speakers.

6.3 Impact of the TSAM Layer Number

Since TSAM for modeling speakers' emotional influences is the critical component of our model, we chose the number of TSAM layers L (ranging from 1 to 5) on the development set of RECCON-DD. As shown in Figure 3, our model with three TSAM layers achieves the best performance. On the one hand, emotion and speaker information may not be refined and interchanged well when the number of layers is small. On the other hand, if the number of layers is excessive, the performance will decrease, possibly due to information redundancy.

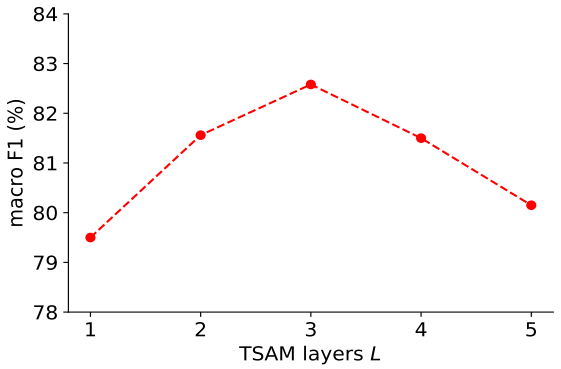

Figure 3: Results of Ours$_{large}$ with various TSAM layers on the development set of RECCON-DD.

6.4 Error Analysis

By analyzing our predicted emotion causes, we find that the following aspects mainly cause the predicted errors. Firstly, our model weakly gives the correct predictions for target utterances with three or more causes.[4] Compared to the utterances with 1 or 2 causes, the proposed model dropped six macro F1 scores on utterances with multiple causes. Secondly, our model cannot predict well when the underlying emotional cause is latent. At this point, recognizing emotion causes may require complex reasoning steps, and commonsense knowledge is an integral part of this process. We take the case below as an example:

- S_A (happy): Hello, thanks for calling 123 Tech Help. I'm Todd. How can I help you?

- S_B (fear): Hello? Can you help me? My computer! Oh, man...

In this case, S_A is happy to help S_B. In this example, the cause of happy emotion is due to the event "greeting" or intention to provide help. On the other hand, S_B is fearful because his or her computer is broken. Both speakers' causes of elicited emotions can only be inferred using commonsense knowledge, which our model does not explicitly consider.

7 Conclusion and Future Work

For the first time, we tackle CEE in a joint framework. We classify multiple utterances synchronously to capture the correlations between contextual utterances in a global view and propose a TSAM to effectively model the speaker's emotional influences. Experimental results on the benchmark dataset show that our model significantly outperforms the SOTA model, and further analysis verifies the effectiveness of each component. This paper points out a new reliable route for follow-up works: incorporating the emotion and speaker information explicitly and modeling the speaker's emotional influences effectively can bring enormous benefits for the tasks similar to CEE.

In the future, we would explore three aspects: (1) Learn emotion recognition and emotion cause recognition in conversations jointly; (2) Apply our model to other similar tasks which need to incorporate the speaker and emotion information; (3) Incorporate commonsense knowledge into the model explicitly to address situations when the underlying emotion cause is latent.

[4]Utterances with 3 or more causes account for approximately 14% of the RECCON-DD dataset

References

Ankush Chatterjee, Umang Gupta, Manoj Kumar Chinnakotla, Radhakrishnan Srikanth, Michel Galley, and Puneet Agrawal. 2019. Understanding emotions in text using deep learning and big data. *Computers in Human Behavior*, 93:309–317.

Sheng Yeh Chen, Chao Chun Hsu, Chuan Chun Kuo, Ting Hao Kenneth Huang, and Lun Wei Ku. 2019. Emotionlines: An emotion corpus of multi-party conversations. In *11th International Conference on Language Resources and Evaluation, LREC 2018*, pages 1597–1601. European Language Resources Association (ELRA).

Xinhong Chen, Qing Li, and Jianping Wang. 2020a. A unified sequence labeling model for emotion cause pair extraction. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 208–218.

Ying Chen, Wenjun Hou, Shoushan Li, Caicong Wu, and Xiaoqiang Zhang. 2020b. End-to-end emotion-cause pair extraction with graph convolutional network. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 198–207.

Ying Chen, Sophia Yat Mei Lee, Shoushan Li, and Chu-Ren Huang. 2010. Emotion cause detection with linguistic constructions. In *Proceedings of the 23rd International Conference on Computational Linguistics (Coling 2010)*, pages 179–187.

Zifeng Cheng, Zhiwei Jiang, Yafeng Yin, Hua Yu, and Qing Gu. 2020. A symmetric local search network for emotion-cause pair extraction. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 139–149.

Leyang Cui and Yue Zhang. 2019. Hierarchically-Refined Label Attention Network for Sequence Labeling. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 4115–4128.

Zixiang Ding, Huihui He, Mengran Zhang, and Rui Xia. 2019. From independent prediction to reordered prediction: Integrating relative position and global label information to emotion cause identification. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, pages 6343–6350.

Zixiang Ding, Rui Xia, and Jianfei Yu. 2020a. ECPE-2D: Emotion-cause pair extraction based on joint two-dimensional representation, interaction and prediction. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 3161–3170.

Zixiang Ding, Rui Xia, and Jianfei Yu. 2020b. End-to-end emotion-cause pair extraction based on sliding window multi-label learning. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 3574–3583.

Chuang Fan, Chaofa Yuan, Jiachen Du, Lin Gui, Min Yang, and Ruifeng Xu. 2020. Transition-based directed graph construction for emotion-cause pair extraction. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 3707–3717.

Deepanway Ghosal, Navonil Majumder, Alexander Gelbukh, Rada Mihalcea, and Soujanya Poria. 2020a. COSMIC: COmmonSense knowledge for eMotion Identification in Conversations. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: Findings*, pages 2470–2481.

Deepanway Ghosal, Navonil Majumder, Soujanya Poria, Niyati Chhaya, and Alexander Gelbukh. 2020b. DialogueGCN: A graph convolutional neural network for emotion recognition in conversation. In *EMNLP-IJCNLP 2019-2019 Conference on Empirical Methods in Natural Language Processing and 9th International Joint Conference on Natural Language Processing, Proceedings of the Conference*.

Lin Gui, Jiannan Hu, Yulan He, Ruifeng Xu, Qin Lu, and Jiachen Du. 2017. A Question Answering Approach for Emotion Cause Extraction. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 1593–1602.

Lin Gui, Dongyin Wu, Ruifeng Xu, Qin Lu, and Yu Zhou. 2016a. Event-Driven Emotion Cause Extraction with Corpus Construction. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 1639–1649.

Lin Gui, Ruifeng Xu, Qin Lu, Dongyin Wu, and Yu Zhou. 2016b. Emotion cause extraction, a challenging task with corpus construction. In *Chinese National Conference on Social Media Processing*, pages 98–109. Springer.

Gaël Guibon, Matthieu Labeau, Hélène Flamein, Luce Lefeuvre, and Chloé Clavel. 2021. Few-Shot Emotion Recognition in Conversation with Sequential Prototypical Networks. In *The 2021 Conference on Empirical Methods in Natural Language Processing (EMNLP 2021)*.

Devamanyu Hazarika, Soujanya Poria, Rada Mihalcea, Erik Cambria, and Roger Zimmermann. 2018a. ICON: Interactive conversational memory network for multimodal emotion detection. In *Proceedings of the 2018 conference on empirical methods in natural language processing*, pages 2594–2604.

Devamanyu Hazarika, Soujanya Poria, Amir Zadeh, Erik Cambria, Louis-Philippe Morency, and Roger Zimmermann. 2018b. Conversational memory network for emotion recognition in dyadic dialogue videos. In *Proceedings of the conference. Association for Computational Linguistics. North American*

Chapter. Meeting, volume 2018, page 2122. NIH Public Access.

Dou Hu, Lingwei Wei, and Xiaoyong Huai. 2021. DialogueCRN: Contextual Reasoning Networks for Emotion Recognition in Conversations. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing, ACL/IJCNLP 2021, (Volume 1: Long Papers), Virtual Event, August 1-6, 2021*, pages 7042–7052.

Taichi Ishiwatari, Yuki Yasuda, Taro Miyazaki, and Jun Goto. 2020. Relation-aware graph attention networks with relational position encodings for emotion recognition in conversations. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 7360–7370.

Wenxiang Jiao, Haiqin Yang, Irwin King, and Michael R Lyu. 2019. HiGRU: Hierarchical Gated Recurrent Units for Utterance-Level Emotion Recognition. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 397–406.

Diederik P Kingma and Jimmy Ba. 2015. Adam: A Method for Stochastic Optimization. In *ICLR (Poster)*.

Philipp Koehn. 2004. Statistical significance tests for machine translation evaluation. In *Proceedings of the 2004 conference on empirical methods in natural language processing*, pages 388–395.

Sophia Yat Mei Lee, Ying Chen, and Chu-Ren Huang. 2010a. A text-driven rule-based system for emotion cause detection. In *Proceedings of the NAACL HLT 2010 Workshop on Computational Approaches to Analysis and Generation of Emotion in Text*, pages 45–53.

Sophia Yat Mei Lee, Ying Chen, Shoushan Li, Chu-Ren Huang, et al. 2010b. Emotion cause events: Corpus construction and analysis. In *LREC*. Citeseer.

Ruifan Li, Hao Chen, Fangxiang Feng, Zhanyu Ma, Xiaojie Wang, and Eduard Hovy. 2021a. Dual Graph Convolutional Networks for Aspect-based Sentiment Analysis. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 6319–6329.

Xiangju Li, Wei Gao, Shi Feng, Yifei Zhang, and Daling Wang. 2021b. Boundary Detection with BERT for Span-level Emotion Cause Analysis. In *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*, pages 676–682.

Xiangju Li, Kaisong Song, Shi Feng, Daling Wang, and Yifei Zhang. 2018. A co-attention neural network model for emotion cause analysis with emotional context awareness. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 4752–4757.

Yanran Li, Hui Su, Xiaoyu Shen, Wenjie Li, Ziqiang Cao, and Shuzi Niu. 2017. DailyDialog: A Manually Labelled Multi-turn Dialogue Dataset. In *Proceedings of the Eighth International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 986–995.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. RoBERTa: A robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692*.

Navonil Majumder, Soujanya Poria, Devamanyu Hazarika, Rada Mihalcea, Alexander Gelbukh, and Erik Cambria. 2019. Dialoguernn: An attentive rnn for emotion detection in conversations. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, pages 6818–6825.

Wei Peng, Yue Hu, Luxi Xing, Yuqiang Xie, and Yajing Sun. 2022. Do You Know My Emotion? Emotion-Aware Strategy Recognition towards a Persuasive Dialogue System. *arXiv preprint arXiv:2206.12101*.

Soujanya Poria, Erik Cambria, Devamanyu Hazarika, Navonil Majumder, Amir Zadeh, and Louis-Philippe Morency. 2017. Context-dependent sentiment analysis in user-generated videos. In *Proceedings of the 55th annual meeting of the association for computational linguistics (volume 1: Long papers)*, pages 873–883.

Soujanya Poria, Devamanyu Hazarika, Navonil Majumder, Gautam Naik, Erik Cambria, and Rada Mihalcea. 2019a. MELD: A Multimodal Multi-Party Dataset for Emotion Recognition in Conversations. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 527–536.

Soujanya Poria, Navonil Majumder, Devamanyu Hazarika, Deepanway Ghosal, Rishabh Bhardwaj, Samson Yu Bai Jian, Pengfei Hong, Romila Ghosh, Abhinaba Roy, Niyati Chhaya, et al. 2021. Recognizing emotion cause in conversations. *Cognitive Computation*, pages 1–16.

Soujanya Poria, Navonil Majumder, Rada Mihalcea, and Eduard Hovy. 2019b. Emotion recognition in conversation: Research challenges, datasets, and recent advances. *IEEE Access*, 7:100943–100953.

Weizhou Shen, Siyue Wu, Yunyi Yang, and Xiaojun Quan. 2021. Directed Acyclic Graph Network for Conversational Emotion Recognition. In *Proceedings of the 59th Annual Meeting of the Association*

for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing, ACL/IJCNLP 2021, (Volume 1: Long Papers), Virtual Event, August 1-6, 2021, pages 1551–1560.

Hao Tang, Donghong Ji, Chenliang Li, and Qiji Zhou. 2020. Dependency graph enhanced dual-transformer structure for aspect-based sentiment classification. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 6578–6588.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in neural information processing systems*, pages 5998–6008.

Petar Veličković, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Liò, and Yoshua Bengio. 2018. Graph Attention Networks. In *International Conference on Learning Representations*.

Yan Wang, Jiayu Zhang, Jun Ma, Shaojun Wang, and Jing Xiao. 2020. Contextualized emotion recognition in conversation as sequence tagging. In *Proceedings of the 21th Annual Meeting of the Special Interest Group on Discourse and Dialogue*, pages 186–195.

Penghui Wei, Jiahao Zhao, and Wenji Mao. 2020. Effective inter-clause modeling for end-to-end emotion-cause pair extraction. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 3171–3181.

Rui Xia and Zixiang Ding. 2019. Emotion-Cause Pair Extraction: A New Task to Emotion Analysis in Texts. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 1003–1012.

Rui Xia, Mengran Zhang, and Zixiang Ding. 2019. RTHN: A RNN-Transformer Hierarchical Network for Emotion Cause Extraction. In *Proceedings of the Twenty-Eighth International Joint Conference on Artificial Intelligence, IJCAI 2019, Macao, China, August 10-16, 2019*, pages 5285–5291.

Yuqiang Xie, Yue Hu, Wei Peng, Guanqun Bi, and Luxi Xing. 2022. Comma: Modeling relationship among motivations, emotions and actions in language-based human activities.

Hanqi Yan, Lin Gui, Gabriele Pergola, and Yulan He. 2021. Position Bias Mitigation: A Knowledge-Aware Graph Model for Emotion Cause Extraction. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing, ACL/IJCNLP 2021, (Volume 1: Long Papers), Virtual Event, August 1-6, 2021*, pages 3364–3375.

Chaofa Yuan, Chuang Fan, Jianzhu Bao, and Ruifeng Xu. 2020. Emotion-cause pair extraction as sequence labeling based on a novel tagging scheme. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 3568–3573.

Duzhen Zhang, Xiuyi Chen, Shuang Xu, and Bo Xu. 2020. Knowledge Aware Emotion Recognition in Textual Conversations via Multi-Task Incremental Transformer. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 4429–4440.

Weixiang Zhao, Yanyan Zhao, and Xin Lu. 2022. CauAIN: Causal Aware Interaction Network for Emotion Recognition in Conversations. In *Proceedings of the Thirty-First International Joint Conference on Artificial Intelligence, IJCAI 2022, Vienna, Austria, 23-29 July 2022*, pages 4524–4530.

Peixiang Zhong, Di Wang, and Chunyan Miao. 2019. Knowledge-Enriched Transformer for Emotion Detection in Textual Conversations. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 165–176.

Hao Zhou, Minlie Huang, Tianyang Zhang, Xiaoyan Zhu, and Bing Liu. 2018. Emotional chatting machine: Emotional conversation generation with internal and external memory. In *Thirty-Second AAAI Conference on Artificial Intelligence*.

Lixing Zhu, Gabriele Pergola, Lin Gui, Deyu Zhou, and Yulan He. 2021. Topic-Driven and Knowledge-Aware Transformer for Dialogue Emotion Detection. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing, ACL/IJCNLP 2021, (Volume 1: Long Papers), Virtual Event, August 1-6, 2021*, pages 1571–1582.

Entity-Level Sentiment Analysis (ELSA): An exploratory task survey

Egil Rønningstad and **Erik Velldal** and **Lilja Øvrelid**
University of Oslo
Department of Informatics
{egilron,erikve,liljao}@ifi.uio.no

Abstract

This paper explores the task of identifying the overall sentiment expressed towards volitional entities (persons and organizations) in a document – what we refer to as Entity-Level Sentiment Analysis (ELSA). While identifying sentiment conveyed towards an entity is well researched for shorter texts like tweets, we find little to no research on this specific task for longer texts with multiple mentions and opinions towards the same entity. This lack of research would be understandable if ELSA can be derived from existing tasks and models. To assess this, we annotate a set of professional reviews for their overall sentiment towards each volitional entity in the text. We sample from data already annotated for document-level, sentence-level, and target-level sentiment in a multi-domain review corpus, and our results indicate that there is no single proxy task that provides this overall sentiment we seek for the entities at a satisfactory level of performance. We present a suite of experiments aiming to assess the contribution towards ELSA provided by document-, sentence-, and target-level sentiment analysis, and provide a discussion of their shortcomings. We show that sentiment in our dataset is expressed not only with an entity mention as target, but also towards targets with a sentiment-relevant relation to a volitional entity. In our data, these relations extend beyond anaphoric coreference resolution, and our findings call for further research of the topic. Finally, we also present a survey of previous relevant work.

1 Introduction

Over the course of the last two decades, the field of NLP has generated a vast body of research on sentiment analysis (SA), i.e. the task of identifying opinions expressed in text. Prior work has focused on a range of different levels of analysis; from document- or sentence-level polarity classification to more fine-grained prediction of various components of opinions, like source/holder expressions, polar expressions, target expressions and aspect-based sentiment classification.

However, we observe that a more aggregated level of analysis, what we here dub "entity-level sentiment analysis" (or ELSA for short), remains under-explored. For our purposes, we will define ELSA as the task of determining the overall (i.e. document-level) polarity (positive/negative) expressed towards an entity in a text. Moreover, for the current paper we will restrict the discussion to volitional entities, like persons (PER) or organizations (ORG).

A given text might make reference to multiple distinct entities, each of which might be mentioned multiple times, both directly and indirectly, and also have multiple opinions directed towards them. Hence, solving the task of ELSA in its full complexity may potentially involve several different sub-tasks, including (but not necessarily limited to) named entity recognition (NER), resolution of entity mentions, coreference- and anaphora-resolution, identification of sentiment targets and/or aspects and their polarities, and finally what we here refer to as "target–entity resolution", i.e. the task of identifying the particular entity with which a given target expression is associated.

The main goal of this paper is to shed more light on this task of entity-level sentiment analysis. To better understand the complexity of the task, we quantify how far we can potentially get toward the goal of ELSA by simply building on existing tools for SA at lower levels. We do this through an exploratory analysis of an existing SA dataset that comprises annotations for several levels of granularity, for which we experiment with different strategies for aggregating these gold annotations to infer entity-level sentiment. Importantly, we also discuss which pieces appear to currently be missing in order to fully solve ELSA. We start, however, by surveying relevant prior work, and also discuss the often diverging terminology used in the field.

Proceedings of the 29th International Conference on Computational Linguistics, pages 6773–6783
October 12–17, 2022.

The paper is structured as follows. In Section 2 we first present background literature on the task of analyzing sentiment towards individual entity representations, before we discuss the limited work we found on resolving sentiment-relevant elements to the document-level. We found no previous work describing the ELSA task as we defined it, and very little in general on resolving sentiment from the sub-sentence level to the document-level. Section 3 presents the datasets we sample from; Norwegian data annotated for document-level, sentence-level and target-level sentiment classification. We also present our new, exploratory dataset, annotated directly for document-level sentiment classification for each volitional entity in the text. In Section 4 we present our results from attempting to derive ELSA from document-level and sentence-level sentiment analysis – what we dub target-independent approaches. We find that merely locating an entity mention inside a positive document is a very weak indicator of a positive sentiment towards that entity. Section 5 presents our findings from deriving ELSA classification from Targeted Sentiment Analysis (TSA). Besides discrepancies from annotator disagreement, we find that in order to fully solve the ELSA task, future work should aim to add another level of analysis, corresponding to the relations between sentiment targets and their affiliated entities. In Section 6 we report on a baseline model for ELSA through TSA as a proxy task.

2 Literature review

This section first presents prior work related to ELSA. We then survey the terminology used in the literature, comparing this to our suggested definition of entity-level SA.

2.1 Related work on short texts

Mitchell et al. (2013) introduce the task of identifying which occurrences of named entities are sentiment targets in a text, and further to classify the sentiment towards these as positive or negative. They narrow the scope of named entities to volitional entities; i.e. organizations and persons. Zhang et al. (2015) follow up on this work and expand on the TSA task description.

The work of Mitchell et al. (2013) is highly related to our task in that they for each text identify the sentiment polarity towards each volitional entity mentioned in the text. However, while Mitchell et al. (2013) work on Twitter data, hence by their nature very short texts, our goal in this work is to extend the task to longer texts where the sentiment towards one volitional entity may be expressed through multiple mentions and opinion expressions.

On a similar note, Jiang et al. (2011) also analyze sentiment towards each target in tweets. The goal of their work is, for a corpus of tweets and a query term, to return tweets classified as positive or negative towards the entity in the query term. The work goes beyond situations where the named entity is the sentiment target, into what is described as "extended targets". This work is a step in the direction of locating an extended set of segments of the text that are relevant for the sentiment towards each entity.

2.2 Related work on longer texts

One possible way of aggregating sentiment expressions is through the application of Coreference Resolution (CR) techniques. De Clercq and Hoste (2020) explore the benefits of CR in Aspect-Based Sentiment Analysis (ABSA). They find that when adding gold-standard coreference information, their models increase their performance by up to one percentage point, while using automatically retrieved CR data (Lee et al., 2013; De Clercq et al., 2011) would have no, or even negative effect on the performance of their ABSA models. Stoyanov and Cardie (2006) explore the benefits of holder coreference resolution as a method for extracting the document-level sentiment expressed by one holder. The task of target coreference resolution is mentioned in suggestions for future work. Farra and McKeown (2017) address the task of open-domain TSA, where their goal is to cluster targets and identify salient entities towards which opinions are expressed in the text. These targets may be nouns, noun phrases, events or concepts.

Steinberger et al. (2017) present their Europe Media Monitor (EMM) system, where the overall task is to detect the positive or negative sentiment towards persons and organizations. They process about 70 languages and therefore use linguistically light-weight methods that can work with low-resource languages.

The recent work of Luo et al. (2022) addresses some of the same limitations in previous work as observed by us, namely the limited focus on identifying sentiment targets only at the level of sentences or tweets. A new multi-domain dataset is

provided, annotated for sentiment targets and polarity, where all targets referring to the same entity are joined in a nested target structure. Their dataset provides a step towards targeted sentiment analysis at the document-level. There is no special treatment of volitional entities however.

2.3 Terminology review

Of the few papers we could find that use the term "entity-level sentiment analysis", several of them appear to treat it as synonymous to targeted SA (TSA) (Li and Lu, 2017; Alimova and Tutubalina, 2019; Huang and Fang, 2020; Sweeney and Padmanabhan, 2017; Engonopoulos et al., 2011). In TSA, the task is for each sentence to extract any segment being the target of a sentiment expression, and the sentiment polarity towards this target. This notion of a target is in line with the target expressions in widely used datasets for TSA (Pontiki et al., 2014, 2016), where there is no linking or aggregation to the document-level. Having the TSA term to describe this task, we suggest that the ELSA term is a better fit for an aggregated entity-level, where one entity may be linked to several targets in multiple sentences.

We did, however, also find a few studies employing the term "entity-level sentiment analysis" about entities aggregated from several mentions in the text (Farra and McKeown, 2017; Steinberger et al., 2017; Luo et al., 2022). These papers were presented in Section 2.1.

2.4 Conclusions from the literature study

We find that while there has been quite some work on classifying the polarity of particular occurrences of named entity mentions in text, like that of Mitchell et al. (2013) and Zhang et al. (2015), we have seen only a few studies that attempt to link or cluster several related segments of a document to resolve sentiment towards entities at the document-level. From existing research, we have seen that coreference resolution (Stoyanov and Cardie, 2006), PMI (Jiang et al., 2011) and semantic clustering (Farra and McKeown, 2017) have been used to resolve which entity mentions belong together. None of these papers were found reporting on actual experiments for the task of assigning one sentiment classification per volitional entity per document though. The recent paper by Luo et al. (2022) represents our closest match.

While we find that our usage of the term "entity-level sentiment analysis" is thematically related to a few other usages in the literature, we do not see any established competing use of the term. We therefore suggest ELSA as an appropriate and descriptive term for the task discussed in the current paper.

3 The sentiment dataset

In order to investigate how the ELSA task relates to pre-existing sentiment analysis tasks, we wanted to build on an existing full-text document collection that is annotated for sentiment information at several levels of analysis. The suite of annotated datasets that are based on the Norwegian Review Corpus – NoReC (Velldal et al., 2018) – fits this bill. While Section 3.2 describes how we build on NoReC to create an exploratory dataset for ELSA, we first describe the different levels of existing annotations in NoReC below.

3.1 NoReC

NoReC is a multi-domain dataset of full-text professional reviews published in Norwegian online news sources, and a subset of the documents have been annotated for fine-grained and sentence-level sentiment. Each review in NoReC is accompanied by a rating given by the reviewer, on a scale from 1 to 6. We here take this to serve as a polarity label for the overall document.

Fine-grained sentiment A subset of NoReC has been annotated for fine-grained sentiment information in NoReC$_{fine}$ (Øvrelid et al., 2020), including holders, target expressions, polar expressions, polarities, and polar intensities. This was one of the datasets used in the recent SemEval shared task on structured sentiment analysis – SemEval-2022 Task 10 (Barnes et al., 2022).[1] As presented in Table 1, the NoReC$_{fine}$ training split consists of 327 documents, comprising 8634 sentences, giving an average of 26.4 sentences per document. The training split contains 5000 unique sentiment targets. Based on this data we can derive datasets for TSA as well as sentence-level SA, as described below.

Target-level sentiment We here describe how we derive a dataset for targeted SA – dubbed NoReC$_{tsa}$ – on the basis of NoReC$_{fine}$. A given sentiment target in NoReC$_{fine}$ may be the target of multiple opinion expressions, each with different polarity and intensity. We assign a value from 1 to

[1] https://competitions.codalab.org/competitions/33556

3 to the sentiment intensities ('slight', 'standard' or 'strong'), and assign the sum of all sentiments to each target. This sum is clipped to a scale from -3 (strong negative) to 3 (strong positive) for each target. The data is made available on GitHub.[2]

Sentence-level sentiment We also derive a 4-class sentence-level sentiment dataset from NoReC$_{fine}$. We take each sentence in NoReC$_{fine}$ with positive opinions only to be a positive sentence (and vice versa for negatives). Sentences without any opinion expressions are labeled "Neutral", and sentiments with both positive and negative sentiments are labeled "Mixed".

Document-level sentiment The document-level sentiment polarity is derived from the rating provided by the review author. Table 1 shows the distribution of documents and sentences relative to these ratings for the subset of data included in the NoReC$_{fine}$ training split and in our exploratory ELSA dataset (as further detailed below). For the same two respective subsets of data, Table 2 shows the distribution of documents and sentences across different domains of reviews.

As we can see, the dataset is highly unbalanced. The extreme ratings of 1 and 6 are rare, and are in our experiments therefore merged with their adjacent ratings. We arrive at sentiment polarity classifications for each document by labeling ratings 1 and 2 as negative, rating 5 and 6 as positive, and we here categorize ratings 3 and 4 as neutral polarity.

In sum, the ecosystem of annotations derived from NoReC$_{fine}$ provides us with a multi-domain dataset of full-text reviews annotated for sentiment at multiple levels of analysis; document-level, sentence-level and target-level. This allows for comparing different strategies for aggregating SA information from different levels as estimators of entity-level sentiment. In order to evaluate these strategies and information sources, the next section describes how we create an exploratory ELSA dataset, adding information about entity-level sentiment for a subset of the documents in NoReC$_{fine}$.

3.2 An exploratory dataset for ELSA

To create an evaluation dataset for ELSA, we sample 50 documents at random from the 327 documents in the NoReC$_{fine}$ train split. There are in total 1345 sentences in this evaluation set.

[2] https://github.com/ltgoslo/norec_tsa

	NoReC$_{fine}$ Train		ELSA subset	
Rating	Docs	Sents	Docs	Sents
1	8	151	1	32
2	27	475	7	121
3	62	1345	4	98
4	91	2504	15	399
5	109	3225	19	586
6	30	934	4	109
Total	327	8634	50	1345

Table 1: Distribution of review ratings (1–6) in the full NoReC$_{fine}$ training split and our ELSA subset.

	NoReC$_{fine}$ Train		ELSA subset	
Category	Docs	Sents	Docs	Sents
games	16	445	2	62
literature	35	877	5	148
misc	1	36	1	36
music	111	1915	13	246
products	30	1753	6	298
restaurants	6	290	1	44
screen	118	2920	20	449
sports	2	149		
stage	8	249	2	62
Total	327	8634	50	1345

Table 2: Distribution of sentences and documents across the multiple domains in the NoReC review corpus.

As a pre-processing step, Named Entities are extracted from the texts using the Huggingface pipeline and the pretrained ScandiNER.[3] language model, fine-tuned for NER on all Nordic languages, where the Norwegian part of the training data is provided by the NorNE corpus (Jørgensen et al., 2020). The reported scores for Norwegian NER are good, ≈91% F_1. Since our goal is to identify sentiment towards volitional entities only, we keep only the entities with PER and ORG label. We find the NER model to perform well, with few or none missed entities. During manual inspection, spurious entities were deleted, and the few misclassifications were corrected.

Resolving coreference by substring matching A volitional entity may have several entity mentions in the text. In order to cluster entity mentions in a text, we resolved the various mentions of a

[3] Available at https://huggingface.co/saattrupdan/nbailab-base-ner-scandi

I met John Wayne yesterday. We said hello on the street when he was taking his grandchild for a walk. John is such a nice guy. Nothing like Clint, who is still very handsome, but seems quite arrogant.

John Wayne [John Wayne, John]	Positive
Clint [Clint]	Negative

Figure 1: Example of entity mentions in text as positive/negative sentiment targets, together with the overall entity-level sentiment labels.

volitional entity by simple substring matching, and also check for the genitive marker *-s* in Norwegian, such that both *Jo Nesbø* and *Nesbøs* are resolved to the same entity. One of the few errors introduced by this approach was that *Elisabeth I* was resolved to the same entity as *Elisabeth II*. These errors were subsequently corrected.

3.2.1 Manual ELSA labeling

For each unique entity in the text that the entity mentions represent, we manually evaluate the documents' sentiment towards these entities into the categories "Positive", "Negative", or "Neutral". For entities that are targets of both positive and negative sentiment expressions, we consider what the document as a whole conveys. This sentiment labeling was performed by one reader, after several readings, when necessary. Figure 1 shows a constructed example text, with the extracted entities, their entity mentions, and their manual sentiment classification.

We can conclude from reading the text that the sentiment towards *John Wayne* is positive, while the sentiment towards *Clint* is negative. There is one positive sentiment expression towards *Clint*: *handsome*, and one negative expression: *arrogant*. As readers of the entire text, we perceive the overall sentiment towards *Clint* to be negative. The entity *John Wayne* has two entity mentions in the text: *John Wayne* and *John*. The pronoun *he* is not an entity mention, but an anaphor, coreferential with *John Wayne*. The grandchild is not an entity because it is not named.

By manually assessing the sentiment expressed towards each entity in the 50 selected documents, we arrive at a dataset of volitional entities and the sentiment towards them at the document-level. The majority of the entities are neutral, as shown in Table 3. This distribution is in line with Mitchell et al. (2013) who find that for their Spanish twitter dataset 24% of the entities are positive targets, 16% of the entities are negative targets, and 61% of the entities are neutral.

	ORG	PER	#	%
Pos	8	76	84	30%
Neg	3	25	28	10%
Neu	36	131	167	60%
Total	47	232	279	100%

Table 3: The sampled subset of 50 documents from NoReC$_{fine}$ contains the names of 279 volitional entities. The manual annotations assigned a neutral entity-level sentiment to the majority of these.

Mentions per entity	Entities	Entity mentions
1	188	188
2	39	78
3+	52	266
Total	279	532

Table 4: The evaluation data contain 279 volitional entities with an average of 1.9 entity mentions per entity. 33% of the entities have more than one mention.

4 Analysis of target-independent approaches to ELSA

If all volitional entities mentioned in a positive text are the target of positive sentiment, the task would be limited to an overall sentiment classification for the text. This naïve approach serves as a baseline for further studies. In the following, we compare our manually labeled entity-level polarities with those of the documents they appear in, as well as those of each sentence with a corresponding entity mention.

4.1 ELSA polarity vs. document polarity

As mentioned in Section 3, each document is assigned the overall polarity positive, neutral or negative, based on its review rating.

In Figure 2 and Table 5 we present the results of an analysis of entity polarities for the different document ratings. We find that only 47.7% of the ELSA entities have the same polarity as the corresponding document-level label. We further observe

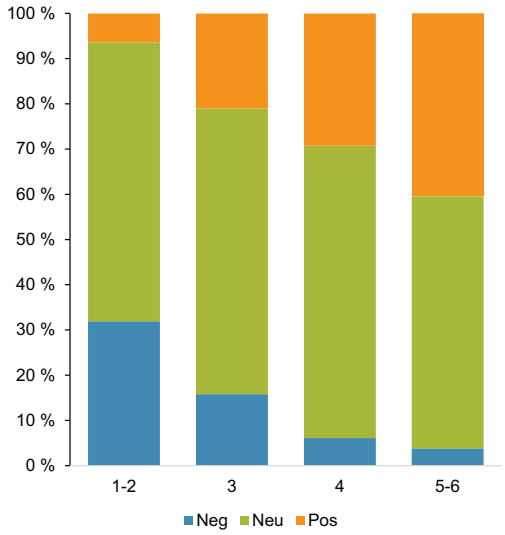

Figure 2: Relative distribution of entity polarities for the different document ratings. Neutral entities are in majority across all ratings.

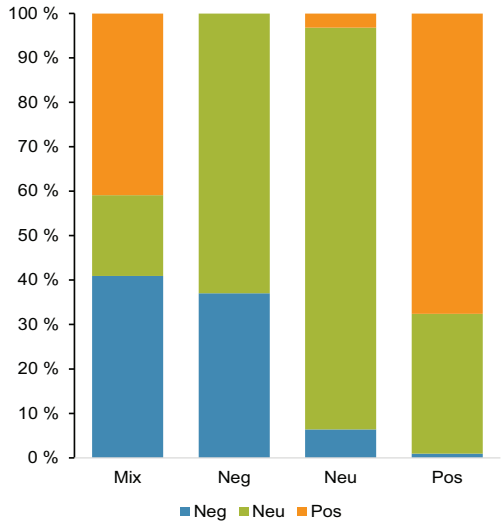

Figure 3: Relative distribution of entity polarities for each of the sentence sentiment categories.

	Entity polarities			
Rating	Neg	Neu	Pos	Entities
1–2	15	29	3	47
3	3	12	4	19
4	5	53	24	82
5–6	5	73	53	131
Total	28	167	84	279
True pos				133
Accuracy				0.477

Table 5: Distribution of entity polarities for each of the document ratings categories. As ratings of 1 and 6 are rather scarce, we merge them with their adjacent rating.

	Entity polarities			
Pol.	Neg	Neu	Pos	Tot.
Mix	9	4	9	22
Neg	10	17		27
Neu	8	113	4	125
Pos	1	33	71	105
Total	28	167	84	279
True pos				194
Accuracy				0.695

Table 6: Distribution of entity polarities for each of the sentence sentiment categories. Approximately 70% of the entities were correctly labeled when using sentence polarity as a proxy.

that the neutral entities are quite evenly distributed – between 55% and 65% across all document polarities. These results clearly suggest that simply inferring entity-level polarity from the document-level is insufficient.

4.2 ELSA polarity vs sentence polarity

We now turn to the sentence-level polarity labels presented in Section 3. Since the ELSA entities may have multiple mentions, they may appear in multiple sentences. We here aggregate the polarity towards an entity by considering it as positive if it is mentioned in more positive than negative sentences (and vice versa). The polarity is considered neutral if the entity appears in only neutral sentences. When entity mentions are equally present in positive and negative sentences, or in mixed polarity sentences only, mixed polarity is assigned.

The results are summarized in Figure 3 and Table 6. We find that neutral entities are more frequent than negative entities in sentences with negative polarity. An example of neutral entities that appear in a non-neutral sentence is provided in Example 1. The sentence is classified as negative and without any sentiment targets. The annotated sentiment towards *Julian Assange* is neutral.

(1) Det gir en ganske merkelig effekt, litt som å treffe Julian Assange på Disneyland.
This has a quite peculiar effect, somewhat like meeting Julian Assange at Disneyland.

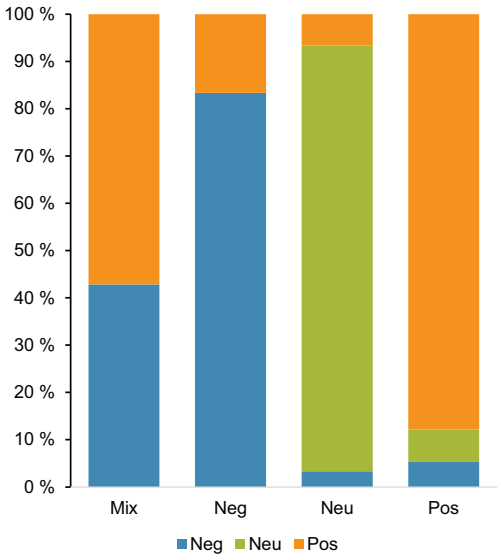

Figure 4: Relative distribution of entity polarities for each of the polarities derived from TSA.

	Entity polarities			
TSA	Neg	Neu	Pos	Tot.
Mix	3		4	7
Neg	11		3	14
Neu	10	162	21	193
Pos	4	5	56	65
Total	28	167	84	279
True pos				229
Accuracy				0.821

Table 7: Distribution of entity polarities for each of the polarities derived from TSA.

Error type	#	%
Missing CR	2	0.72%
Missing TER	18	0.65%
Anno. disagreement	24	8.60%
Mixed polarity	6	2.15%

Table 8: Distribution of different error types when classifying entity sentiment based on aggregated TSA: Missing coreference resolution (CR), missing target–entity resolution (TER), annotator disagreement, and cases of ties from mixed TSA polarities.

116 of the entities (42%), have mentions in the same sentence as an entity of a conflicting polarity. The mixing of polarities inside a sentence indicates that inferring entity polarity from sentence polarity has limited potential. In sum, we find that 69.5% of the entities would be correctly resolved by inferring entity-level sentiment from the sentence sentiment.

5 A target-dependent approach to ELSA

An intuitively more promising approach is to derive ELSA sentiment labels from targeted sentiment analysis. Our TSA dataset presented in Section 3.1 is annotated for sentiment towards each target with a scale from -3 (strong negative) to 3 (strong positive). We aggregate these labels from the target-level to the entity-level for each entity by including only the sentiment targets that overlap with an entity mention, and summing the sentiment values for these targets. This approach leaves 7 entities with an unresolved classification due to equally strong positive and negative sentiment towards its entity mentions. These entities are placed in the "Mix" category.

As shown in Table 7, 229 of our 279 volitional entities, 82%, are given the correct sentiment label when aggregating the TSA annotations. This is an encouraging improvement over previous results. Virtually all neutral entities receive a neutral label through this approach. The 50 entities that are not correctly labeled using this approach, however, call for further analysis.

5.1 Studying the remaining 50 entities

In a further analysis step, we manually inspect the misclassified entities following TSA to look for common causes of errors.

Annotator disagreement Table 8 shows that human disagreement was the most important cause. For these instances, we have interpreted the sentences differently with respect to sentiment, than the original annotators of NoReC$_{fine}$. This serves as an example of the subjectivity and one may say fragility of human sentiment annotations. This is also evident in the moderate inter-annotator agreement for the NoReC$_{fine}$ annotations, reported to be 73% F_1 for targets, when counting binary overlap (Øvrelid et al., 2020).

Coreference resolution To our surprise, only two (out of 50) entities were incorrectly classified due to lack of coreference resolution. One such case is presented in example (2) below.

(2) Joseph Goebbels er på den måten i sjeldent, men ikke godt, selskap. Han er blitt stående som selve bildet på naziregimets hensynsløse og ondskapsfulle hatideologi.
Joseph Goebbels is in this respect in an exclusive, but not good company. He is the embodiment of the cruel and evil nazi regime's ideology of hatred.

In Example 2, *Joseph Goebbels* in the first sentence is not annotated as a sentiment target in NoReC$_{fine}$, while *Han* 'He' in the second sentence is a negative sentiment target. Through coreference resolution, *Han* 'He' would likely be resolved to refer to *Joseph Goebbels*, and the sentiment towards the entity would be correctly resolved.

Sentiment-relevant relation extraction From our manual inspection we find another cause of misclassification pointing to the need for what we might call "target–entity resolution" rather than classical anaphora / cataphora coreference resolution. These are examples that require inference of semantic relations between different target expressions and entities in the text. Typical examples of this in our data are examples of a work of art where the sentiment towards the work of art implies a sentiment towards the creator, or a noun describing a group of people where an entity is a member.

(3) Magnus Beite har skaffet filmen musikk som kler miljøet
Magnus Beite has provided the film with music that suits the environment

(4) Bandet bestående av Daniel Birkeland på gitar [...] og trommis Helge Nyheim klarer virkelig å gjenskape Beatles [...] på en måte som det står respekt av.
The band consisting of Daniel Birkeland on guitar [...] and drummer Helge Nyheim really manage to give life to Beatles [...] in a way that commands respect.

In Example 3, *musikk* 'music' is a positive sentiment target. When reading the full text, we also perceive this as a positive sentiment towards *Magnus Beite*. In Example 4, *Bandet* 'The band' is the sentiment target, and we perceive this to imply positive sentiment towards the individual members of the band as well.

For these types of examples, if we had access to a "member of"-relation between the band members and the band, and a "creator of"-relation between the photographer and the photograph, or more generally a relation of target–entity affiliation, the sentiment towards the volitional entities would have been resolved correctly. (Note that traditional aspect-categories would not be of help here, as we would still be missing the relations between the targets and the relevant entities.) These observations open new research questions, however, about which semantic relations are sentiment-relevant and under which circumstances. Jiang et al. (2011) also point to this question, with the example that a sentiment about someone's behavior usually means a sentiment about the person, while a sentiment about someone's colleague usually has nothing to do with the person.

6 Modeling

The above data analysis was performed using gold standard data for the various sentiment-levels (document, sentence and target). In order to gauge the performance attainable in a more realistic setting, we present results using automatically derived TSA information in the following.

Since we have no directly annotated ELSA training data, we create a baseline model using the proxy task of TSA. The exploratory ELSA dataset was taken from the training split of NoReC$_{fine}$, and we therefore join the remainder of the training split with the development split to create our baseline training data, and we perform the final evaluation on the 50 documents in our ELSA dataset. The model setup is adapted from the Huggingface example configuration for NER.[4] We use default hyperparameters and perform no hyperparameter tuning. We therefore chose to not set aside data for a dev set, and not touch the original NoReC$_{fine}$ test split, in order to allow for further research on this data split. We preprocess our training data the same way as we did with the ELSA data in Section 5, finding PER and ORG labels in the data through NER, and deriving sentiment towards these through the pre-existing TSA target annotations. Volitional entities that are sentiment targets, receive the sentiment label "Positive" or "Negative", while the volitional entities that are not sentiment targets are labeled "Neutral". With this setup, volitional entities and their sentiment polarity is predicted for

[4] https://github.com/huggingface/transformers/tree/main/examples/pytorch/token-classification

	Gold entity pol			
Pred entity pol	Neg	Neu	Pos	FP
Neg	4	0	4	1
Neu	13	154	27	35
Pos	9	6	48	8
Missed	2	7	5	

Table 9: A confusion matrix of gold and predicated entity polarities. The final column indicates false positive entity predictions. Our simple NER+TSA-based baseline model for ELSA predicted the name and sentiment polarity correctly for 206 of the 279 entities in the test data. The model produced 44 false positive entities, resulting in Precision: 66.7%, Recall: 73.8% and F_1: 70.1%.

the 50 documents in the exploratory dataset. The predicted entity mentions were resolved document-wise through substring matching, before summing the predicted polarities. For evaluation, these predictions are compared with our manually resolved and annotated entities.[5] Our evaluation ignores whether the model assigned the correct NER category PER or ORG. Our evaluation shows that this baseline model has a F_1-score of 70.1%, as shown in Table 9.

7 Conclusion

In this paper we have explored the task of entity-level sentiment analysis (ELSA) – the task of determining the aggregated or overall (i.e. document-level) polarity expressed towards an entity in a text. In particular for longer text, which might comprise several distinct entities, entity mentions and opinions, this is a potentially complex task. The paper has surveyed existing literature and terminology in adjacent and related previous work, in addition to presenting an exploratory data analysis to shed more light on what is required to solve the task.

Relating to the latter point, in order to assess the relevance to ELSA of existing approaches to sentiment analysis at different granularities – i.e. document-, sentence-, and target-level SA – we perform a task analysis on the basis of a Norwegian multi-domain review dataset containing all these layers of sentiment annotation. When utilizing the fine-grained sentiment annotations of the NoReC$_{fine}$ dataset, we found that the overlap between a volitional entity and TSA annotations, gave us the correct sentiment category for 82% of our 279 manually evaluated ELSA entities. We further found that, using automated TSA, we obtained an F_1-score of 70.1%. TSA therefore appears to be highly relevant, though not sufficient, for the ELSA task.

For the ELSA classifications that could not be derived from NER and TSA annotations, we found surprisingly few cases that would be correctly resolved through coreference resolution. This is in line with the findings of De Clercq and Hoste (2020). Our dataset is likely too small, however, to draw any definitive conclusions about the importance of coreference resolution for ELSA. On the other hand, we did observe a need for more generally resolving relations that tie target expressions to their corresponding entities.

In our exploratory data analysis, we have identified at least four categories of information that could potentially benefit the classification of a document's overall sentiment towards entities: (*i*) named entity recognition, (*ii*) targeted sentiment analysis, (*iii*) coreference resolution, and (*iv*) what we dub target–entity resolution. The latter being a concept derived from working with the examples in our dataset, and refers to the task of identifying which entity a given target expression relates to. This appears to be a missing link in a pipeline for ELSA based on TSA, and to explore methods for filling this gap is a suggestion for further work.

Acknowledgements

The work documented in this publication has been carried out within the NorwAI Centre for Research-based Innovation, funded by the Research Council of Norway (RCN), with grant number 309834, and the SANT project (Sentiment Analysis for Norwegian Text), also funded by the RCN (grant number 270908).

References

Ilseyar Alimova and Elena Tutubalina. 2019. Entity-level classification of adverse drug reactions: a comparison of neural network models. In *Proceedings of the 2019 Workshop on Widening NLP*, pages 132–134, Florence, Italy. Association for Computational Linguistics.

Jeremy Barnes, Laura Ana Maria Oberländer, Enrica Troiano, Andrey Kutuzov, Jan Buchmann, Ro-

[5] Code for our data collection, analysis and modeling will be available at https://github.com/egilron/elsa-introduction

drigo Agerri, Lilja Øvrelid, and Erik Velldal. 2022. SemEval-2022 task 10: Structured sentiment analysis. In *Proceedings of the 16th International Workshop on Semantic Evaluation (SemEval-2022)*, Seattle. Association for Computational Linguistics.

Orphee De Clercq and Veronique Hoste. 2020. It's absolutely divine! can fine-grained sentiment analysis benefit from coreference resolution? In *Proceedings of the Third Workshop on Computational Models of Reference, Anaphora and Coreference*, pages 11–21, Barcelona, Spain (online). Association for Computational Linguistics.

Orphée De Clercq, Véronique Hoste, and Iris Hendrickx. 2011. Cross-domain Dutch coreference resolution. In *Proceedings of the International Conference Recent Advances in Natural Language Processing 2011*, pages 186–193, Hissar, Bulgaria. Association for Computational Linguistics.

Nikos Engonopoulos, Angeliki Lazaridou, Georgios Paliouras, and Konstantinos Chandrinos. 2011. ELS: a word-level method for entity-level sentiment analysis. In *Proceedings of the International Conference on Web Intelligence, Mining and Semantics (WIMS '11)*, pages 1–9, Sogndal, Norway.

Noura Farra and Kathy McKeown. 2017. SMARTies: Sentiment models for Arabic target entities. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 1, Long Papers*, pages 1002–1013, Valencia, Spain. Association for Computational Linguistics.

Zhihong Huang and Zhijian Fang. 2020. An entity-level sentiment analysis of financial text based on pre-trained language model. In *2020 IEEE 18th International Conference on Industrial Informatics (INDIN)*, volume 1, pages 391–396. IEEE.

Long Jiang, Mo Yu, Ming Zhou, Xiaohua Liu, and Tiejun Zhao. 2011. Target-dependent Twitter sentiment classification. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, pages 151–160, Portland, Oregon, USA. Association for Computational Linguistics.

Fredrik Jørgensen, Tobias Aasmoe, Anne-Stine Ruud Husevåg, Lilja Øvrelid, and Erik Velldal. 2020. NorNE: Annotating named entities for Norwegian. In *Proceedings of the 12th Language Resources and Evaluation Conference*, pages 4547–4556, Marseille, France. European Language Resources Association.

Heeyoung Lee, Angel Chang, Yves Peirsman, Nathanael Chambers, Mihai Surdeanu, and Dan Jurafsky. 2013. Deterministic coreference resolution based on entity-centric, precision-ranked rules. *Computational linguistics*, 39(4):885–916.

Hao Li and Wei Lu. 2017. Learning latent sentiment scopes for entity-level sentiment analysis. In *Proceedings of the Thirty-First AAAI Conference on Artificial Intelligence*, San Francisco, USA.

Yun Luo, Hongjie Cai, Linyi Yang, Yanxia Qin, Rui Xia, and Yue Zhang. 2022. Challenges for open-domain targeted sentiment analysis.

Margaret Mitchell, Jacqui Aguilar, Theresa Wilson, and Benjamin Van Durme. 2013. Open domain targeted sentiment. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*, pages 1643–1654, Seattle, Washington, USA. Association for Computational Linguistics.

Lilja Øvrelid, Petter Mæhlum, Jeremy Barnes, and Erik Velldal. 2020. A fine-grained sentiment dataset for Norwegian. In *Proceedings of the 12th Language Resources and Evaluation Conference*, pages 5025–5033, Marseille, France. European Language Resources Association.

Maria Pontiki, Dimitris Galanis, Haris Papageorgiou, Ion Androutsopoulos, Suresh Manandhar, Mohammad AL-Smadi, Mahmoud Al-Ayyoub, Yanyan Zhao, Bing Qin, Orphée De Clercq, Véronique Hoste, Marianna Apidianaki, Xavier Tannier, Natalia Loukachevitch, Evgeniy Kotelnikov, Nuria Bel, Salud María Jiménez-Zafra, and Gülşen Eryiğit. 2016. SemEval-2016 task 5: Aspect based sentiment analysis. In *Proceedings of the 10th International Workshop on Semantic Evaluation (SemEval-2016)*, pages 19–30, San Diego, California. Association for Computational Linguistics.

Maria Pontiki, Dimitris Galanis, John Pavlopoulos, Harris Papageorgiou, Ion Androutsopoulos, and Suresh Manandhar. 2014. SemEval-2014 task 4: Aspect based sentiment analysis. In *Proceedings of the 8th International Workshop on Semantic Evaluation (SemEval 2014)*, pages 27–35, Dublin, Ireland. Association for Computational Linguistics.

Ralf Steinberger, Stefanie Hegele, Hristo Tanev, and Leonida Della Rocca. 2017. Large-scale news entity sentiment analysis. In *Proceedings of the International Conference Recent Advances in Natural Language Processing, RANLP 2017*, pages 707–715, Varna, Bulgaria. INCOMA Ltd.

Veselin Stoyanov and Claire Cardie. 2006. Partially supervised coreference resolution for opinion summarization through structured rule learning. In *Proceedings of the 2006 Conference on Empirical Methods in Natural Language Processing*, pages 336–344, Sydney, Australia. Association for Computational Linguistics.

Colm Sweeney and Deepak Padmanabhan. 2017. Multi-entity sentiment analysis using entity-level feature extraction and word embeddings approach. In *Proceedings of the International Conference Recent Advances in Natural Language Processing, RANLP 2017*, pages 733–740, Varna, Bulgaria. INCOMA Ltd.

Erik Velldal, Lilja Øvrelid, Eivind Alexander Bergem, Cathrine Stadsnes, Samia Touileb, and Fredrik Jørgensen. 2018. NoReC: The Norwegian review corpus. In *Proceedings of the Eleventh International*

Conference on Language Resources and Evaluation (LREC 2018), Miyazaki, Japan. European Language Resources Association (ELRA).

Meishan Zhang, Yue Zhang, and Duy-Tin Vo. 2015. Neural networks for open domain targeted sentiment. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 612–621, Lisbon, Portugal. Association for Computational Linguistics.

Learning from Adjective-Noun Pairs: A Knowledge-enhanced Framework for Target-Oriented Multimodal Sentiment Classification

Fei Zhao Zhen Wu* Siyu Long Xinyu Dai Shujian Huang Jiajun Chen
National Key Laboratory for Novel Software Technology, Nanjing University, China
Collaborative Innovation Center of Novel Software Technology and Industrialization, China
{zhaof, longsy}@smail.nju.edu.cn
{wuz, daixinyu, huangsj, chenjj}@nju.edu.cn

Abstract

Target-oriented multimodal sentiment classification (TMSC) is a new subtask of aspect-based sentiment analysis, which aims to determine the sentiment polarity of the opinion target mentioned in a (sentence, image) pair. Recently, dominant works employ the attention mechanism to capture the corresponding visual representations of the opinion target, and then aggregate them as evidence to make sentiment predictions. However, they still suffer from two problems: (1) The granularity of the opinion target in two modalities is inconsistent, which causes visual attention sometimes fail to capture the corresponding visual representations of the target; (2) Even though it is captured, there are still significant differences between the visual representations expressing the same mood, which brings great difficulty to sentiment prediction. To this end, we propose a novel Knowledge-enhanced Framework (*KEF*) in this paper, which can successfully exploit adjective-noun pairs extracted from the image to improve the visual attention capability and sentiment prediction capability of the TMSC task. Extensive experimental results show that our framework consistently outperforms state-of-the-art works on two public datasets.

1 Introduction

Target-oriented multimodal sentiment classification (TMSC) is a new sub-task of aspect-based sentiment analysis (Pang et al., 2008; Liu, 2012; Pontiki et al., 2014), which aims to predict the sentiment polarity of the opinion target mentioned in a pair of sentence and image. The assumption behind this task is that the image information can help the text content identify the sentiment of the opinion target. Fig. 1(a) and Fig. 1(b) show two representative examples. It is hard to detect the sentiment of the opinion target (i.e., "*Vince Gilligan*" or "*Sammy*") only depending on informal sentences, but the visual content (i.e., smiling face) associated with the target can clearly reflect its sentiment polarity.

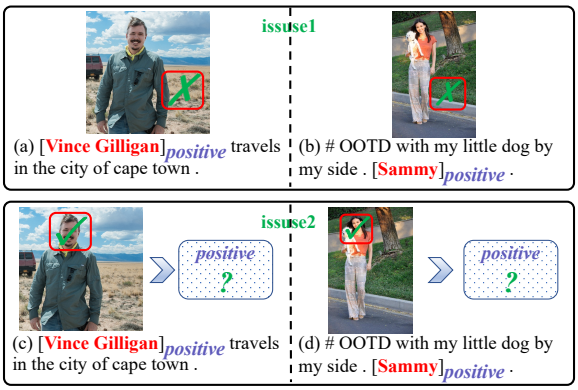

Figure 1: Two examples of Target-Oriented Multimodal Sentiment Classification (TMSC). Opinion targets and their corresponding sentiment polarities are highlighted in the sentence. The red bounding box denotes the visual clues that the opinion target focuses on.

From the above examples, we can conclude that aligning the opinion target of two modalities and capturing helpful visual sentiment features play a critical role in the TMSC task. Given its importance, dominant works employ the attention mechanism (Bahdanau et al., 2015) to automatically learn alignment of text and image, and then aggregate the captured the visual representations of the opinion target as auxiliary evidence to make sentiment predictions (Xu et al., 2019b; Yu et al., 2019; Yu and Jiang, 2019; Zhou et al., 2021; Zhang et al., 2021; Wang et al., 2021).

Despite achieving some improvements, the aforementioned methods still suffer from two key problems: (i). These methods easily fail to align two modalities because of the granularity gap of opinion target across text and image. Specifically, the opinion target appearing in the image often refers to a coarse-grained object concept (e.g., the `man` in Fig. 1(a)), while corresponding opinion target in the sentence are usually a fine-grained entity (e.g.,

* Corresponding author.

Proceedings of the 29th International Conference on Computational Linguistics, pages 6784–6794
October 12–17, 2022.

the man's name [1] "*Vince Gilligan*"). The inconsistency of target granularity causes visual attention sometimes fail to capture the corresponding visual representations. (ii). Even though it is captured, diversified visual representations expressing the same mood also bring challenges for sentiment prediction. Take Fig.1(c) and Fig.1(d) as an example, the opinion target "*Vince Gilligan*" and "*Sammy*" separately focus on the coarse-grained object concepts man and girl in the image, and from their facial expressions we can tell that they are smiling, but the angle and magnitude of the smile are quite different. The variety of visual representations inevitably leads to its sparsity, which makes it hard to learn the accurate mapping function between visual representations and sentiment labels.

In this work, we provide a new idea to tackle the above problems, i.e., exploiting adjective-noun pairs (ANPs) (Borth et al., 2013) extracted from images (e.g., "*nice clouds*", "*bad car*", "*happy man*", "*clear sky*" and "*dry grass*" in Fig. 2(a)). For the first issue, we observed that the nouns of ANPs are also coarse-grained concepts, so an intuitive idea is to map a fine-grained opinion target (e.g. "*Vince Gilligan*") to a coarse-grained noun (e.g. "*man*"[2]) in ANPs. In this manner, it is easier to bridge the granularity gap of two modalities and align text and image. For the second issue, we observed that ANPs can usually extract the same adjectives from different visual content expressing the same mood, so an intuitive idea is to map diversified visual representations (e.g., smiling faces) to the same adjective (e.g., "*happy*"[3]). Apparently, it is easier to learn the mapping function between these same adjectives and sentiment labels.

To facilitate the TMSC task with ANPs, we propose a novel Knowledge-enhanced Framework (*KEF*), which consists of two components: Visual Attention Enhancer and Sentiment Prediction Enhancer. Specifically, the former first finds the noun most related to the opinion target from ANPs with our designed mapping method, and then uses it to improve the effectiveness of visual attention. The latter aims to build the connection between the adjective and target-relevant visual representations, and then utilizes it as the complementary information of visual representations to reduce the difficulty of predicting sentiment labels.

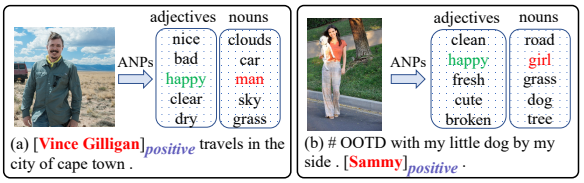

Figure 2: Extract Top-5 adjective-noun pairs (ANPs) from each image in our Twitter datasets.

Our contributions are summarized as follows:

- To the best of our knowledge, we are the first to propose leveraging adjective-noun pairs (ANPs) extracted from the image to help align text and image in the TMSC task.
- We propose a novel Knowledge-enhanced Framework (*KEF*), which contains a Visual Attention Enhancer to improve the effectiveness of visual attention, and a Sentiment Prediction Enhancer to reduce the difficulty of sentiment prediction.
- The *KEF* framework has good compatibility and is easily extended to existing attention-based models. In this work, we apply it to two latest TMSC models: *SaliencyBERT* (Wang et al., 2021) and *TomBERT* (Yu and Jiang, 2019). Experimental results on two public datasets prove the validity of our framework.

2 Notations and Preliminaries

In this section, we first present the task formalization, and then give brief introductions to adjective-noun pairs (Borth et al., 2013).

2.1 Task Formalization

We are given a set of multimodal samples $\mathcal{D}$. For each sample $c \in \mathcal{D}$, it contains a review sentence S with n words $(w_1, w_2, \cdots, w_n)$, an associated image I, as well as an opinion target T (refers to a span in sentence S). Our goal is to predict the sentiment label y of each opinion target mentioned in a pair of sentence and image, where y can be either *positive*, *negative*, or *neutral*.

2.2 Adjective-Noun Pairs

We extract the adjective-noun pairs (ANPs) from each image in our Twitter datasets to serve as an external knowledge base, where nouns denote coarse-grained object concepts in the image, and adjectives

[1] There are respectively 38.4% and 48.5% of opinion targets are different names of people in TWITTER-15 and TWITTER-17 datasets.

[2] Fig. 2(a) extracts the noun "*man*" from the image.

[3] Fig. 2(a) and Fig. 2(b) both extract same adjective "*happy*" from different smiling faces, this phenomenon is common.

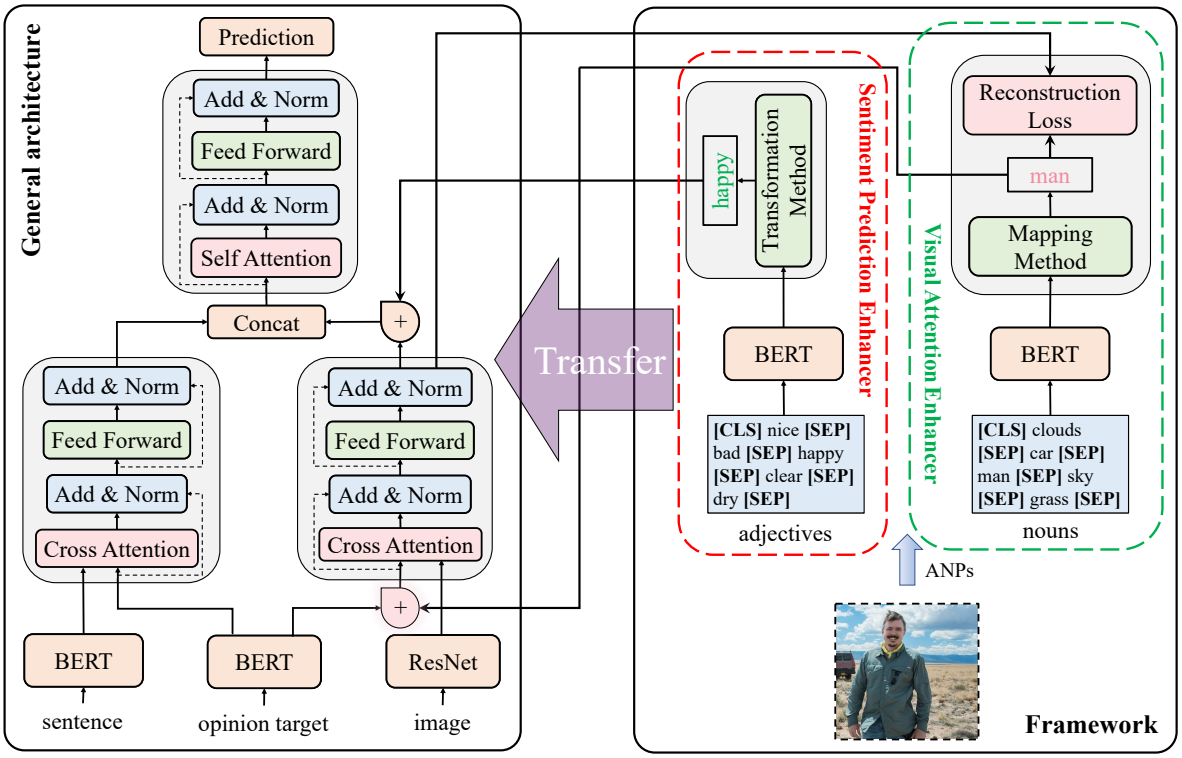

Figure 3: The overview of our KEF framework.

are modifiers of nouns. As shown in Fig. 2, we use SentiBank toolkit[4] to extract 1200 ANPs and select the Top-5[5] ANPs as the visual semantic information of each image, in which each pair contains an adjective word A_i and a noun word N_i:

$$\text{ANP} = (A_i, N_i), \tag{1}$$

Considering that adjectives and nouns in ANPs contain different semantics, i.e., the nouns mainly involve information about the opinion target, while the adjectives usually contain sentiment information. Thus, it is more appropriate and reasonable to encode them separately. For example, the BERT input for adjectives and nouns is given in Figure 3. We feed them to a pre-trained model BERT (Devlin et al., 2019) to obtain the hidden representations:

$$H_A = \text{BERT}(A), \tag{2}$$
$$H_N = \text{BERT}(N), \tag{3}$$

where $H_A \in \mathbb{R}^{(2l+1)\times d}$ and $H_N \in \mathbb{R}^{(2l+1)\times d}$ separately denote the adjective representations and noun representations, d indicates the hidden dimension, and l means the length of ANPs.

[4]ee.columbia.edu/ln/dvmm/vso/download/sentibank.html

[5]we will give the reason why the Top-5 ANPs are extracted from each image in the Section 5.3.

3 Knowledge-enhanced Framework

In this section, we will describe how to integrate Knowledge-enhanced Framework (*KEF*) into *TomBERT* (Yu and Jiang, 2019) and *SaliencyBERT* (Wang et al., 2021), which both achieve satisfying performance and thus are chosen as the foundation of our work.

3.1 Overview

Figure 3 shows the overall architecture of *KEF*, which contains two components: Visual Attention Enhancer and Sentiment Prediction Enhancer. Concretely, we first abstract a general attention architecture based on the well-designed *TomBERT* and *SaliencyBERT* models. Then, with the help of ANPs, we successively present a Visual Attention Enhancer and a Sentiment Prediction Enhancer. The former aims to improve the effectiveness of visual attention through a mapping method and a reconstruction loss, and the latter introduces a simple yet effective transformation approach to reduce the difficulty of predicting sentiment labels.

3.2 General Attention Architecture

Given an input sentence S, we first split S into two sub-sentences: the opinion target T and

the remaining context[6] C, and then separately feed them to pre-trained BERT to obtain the hidden representations:$H_C = \text{BERT}(C), H_T = \text{BERT}(T)$, where $H_C \in \mathbb{R}^{n\times d}$ and $H_T \in \mathbb{R}^{t\times d}$ denote the text representations and target representations, d is the hidden dimension, n and t are the length of C and the target T.

Similarly, for the associated image I, we adopt one of the state-of-the-art image recognition models ResNet-152 (res5c) (He et al., 2016) to obtain the output of the last convolutional layer:

$$\text{ResNet}(I) = \{r_j | r_j \in \mathbb{R}^{2048}, j = 1, 2, ..., 49\}, \tag{4}$$

which splits the original image into 7 × 7 = 49 regions and each region is represented by a 2048-dimensional vector r_j. Next, we use a linear function to project the visual features to the same space of textual features: $H_V = W_v\text{ResNet}(I)$, where $W_v \in \mathbb{R}^{d\times 2048}$ is the learnable parameter.

After that, we employ a cross-attention block to capture target-aware visual representation $H_{T\to V}$ and target-aware text representation $H_{T\to C}$:

$$H_{T\to V} = \text{Cross-ATT}(H_T, H_V), \tag{5}$$
$$H_{T\to C} = \text{Cross-ATT}(H_T, H_C), \tag{6}$$

where Cross-ATT(·) denotes the cross-modal multi-head attention as (Tsai et al., 2019). Then, we concatenate $H_{T\to C}$ and $H_{T\to V}$ together and further stack the attention block on top to obtain the multimodal output representation H.

Finally, we feed the first token H^0 of the multimodal representation to a softmax layer for the sentiment classification:

$$p(y|H^0) = \text{softmax}(W_M^\top H^0), \tag{7}$$

where $W_M \in \mathbb{R}^{d\times 3}$ is the weight matrix.

To optimize all the parameters, the objective is to minimize the standard cross-entropy loss function:

$$\mathcal{L}_t = -\frac{1}{|\mathcal{D}|}\sum_{i=1}^{|\mathcal{D}|} \log p(y^i|H^0). \tag{8}$$

3.3 Visual Attention Enhancer

As mentioned before, the target appearing in the image is a *coarse-grained concept*, while the target mentioned in the sentence is a *fine-grained concept*, the inconsistency of target granularity causes visual attention in Eq. 5 sometimes fail to capture the corresponding visual representations of the target.

[6]we replace the opinion target with T in the context.

Basic Intuition. Apparently, the nouns extracted from the image are also *coarse-grained concepts*, so an intuitive idea is to map a fine-grained opinion target to a coarse-grained noun, and then use it as a bridge to capture the coarse-grained visual representations. However, most of the nouns extracted from the image are target-independent, so we cannot use them directly.

Mapping Method. To tackle the above challenge, we first measure the strength of target-noun relevance by calculating the semantic similarity between noun representation and target representations in the embedding space:

$$\alpha^i = \cos(H_T, H_N^i), \tag{9}$$

where H_N^i denotes single noun representation of H_N in Eq. 3, cos(·) is a cosine function and α^i means the similarity score.

Based on the largest similarity score, we can find the most relevant noun to the opinion target:

$$\alpha^m = \max_{i=1}^{l}(\alpha^i), \tag{10}$$

where l denotes the length of ANPs and H_N^m indicates the noun representation corresponding to the highest similarity score α^m.

Next, we aggregate them together as complementary information for the opinion target to capture the corresponding visual representations $H_{T\to V}$. Formally, we update H_T in Eq. 5 by:

$$\widetilde{H}_N = \alpha^m H_N^m, \tag{11}$$
$$H_T = H_T + \lambda_N \widetilde{H}_N, \tag{12}$$

where λ_N is a hyperparameter that controls the importance of $\widetilde{H}_N$ and can be adjusted.

Reconstruction Loss. To ensure that visual attention can capture the visual features associated with the opinion target more accurately, we also devise a reconstruction loss to minimize the divergence between target-relevant noun representations and target-aware visual representations. Formally,

$$\mathcal{L}_a = -\frac{1}{|\mathcal{D}|}\sum_{i=1}^{|\mathcal{D}|}(\widetilde{H}_N - H_{T\to V})^2, \tag{13}$$

In the Visual Attention Enhancer, the final loss is $\mathcal{L} = \mathcal{L}_t + \lambda\mathcal{L}_a$, where λ measures the importance of reconstruction loss $\mathcal{L}_a$ and can be adjusted.

	TWITTER-15							TWITTER-17						
	Pos	**Neg**	**Neu**	**Total**	**AT**	**Words**	**AL**	**Pos**	**Neu**	**Neg**	**Total**	**AT**	**Words**	**AL**
Train	928	368	1883	3179	1.348	9023	16.72	1508	416	1638	3562	1.410	6027	16.21
Dev	303	149	670	1122	1.336	4238	16.74	515	144	517	1176	1.439	2922	16.37
Test	317	113	607	1037	1.354	3919	17.05	493	168	573	1234	1.450	3013	16.38

Table 1: The basic statistics of our two multimodal Twitter datasets. Pos: Positive, Neg: Negative, Neu: Neutral, AT: Avg. Targets, AL: Avg. Length

3.4 Sentiment Prediction Enhancer

Even if visual features are captured, there are still significant differences between the visual representations expressing the same mood, which brings challenges to learn the mapping function between visual representations and sentiment labels.

Basic Intuition. Considering that ANPs can usually extract the same adjectives from different visual representations expressing the same mood, so an intuitive idea is to map dversified visual representations to the same adjective. However, the adjective most relevant to visual representations is unknown, we need to find it explicitly.

Transformation Method. Actually, in the mapping method, we have found that the noun representation H_N^m is most relevant to target-aware visual representations $H_{T\to V}$. Since an adjective is a modifier of a noun, the adjective corresponding to this noun is also most relevant to target-aware visual representations. Finally, we use it as the complementary information of visual representations to reduce the difficulty of sentiment prediction. Formally, we rewrite $H_{T\to V}$ in Eq. 5 by:

$$H_{T\to V} = H_{T\to V} + \lambda_A H_A^m. \tag{14}$$

where H_A^m denotes the adjective representation corresponding to the noun representation H_N^m, λ_A is a hyperparameter and can be adjusted.

4 Experiments

4.1 Experimental Settings

Datasets. To evaluate the effect of *KEF* Framework, we carry out experiments on two public multimodal datasets `TWITTER-15` and `TWITTER-17` from (Yu and Jiang, 2019), which include user tweets posted during 2014-2015 and 2016- 2017, respectively. General information for both datasets is presented in Table 1.

Implementation Details. We build our *KEF* on top of the pre-trained uncased BERT-based model released by (Devlin et al., 2019), and tune the hyperparameters on the development set of each dataset. Specifically, we set λ_N, λ, λ_A to be {0.5, 0.2, 0.6} on `TWITTER-15` and {0.2, 0.3, 0.2} on `TWITTER-17`, the learning rate as 2e-5 and the dropout (Hinton et al., 2012) rate as 0.9. In addition, the mini-batch size is set to 16, the maximum length of the sentence input and the target input are respectively set as 64 and 32, the hidden dimension and the number of attention heads set as 768 and 12. All the models are implemented by the Tensorflow framework with an NVIDIA Tesla V100 GPU.

Evaluation Metrics and Significance Test. Following (Yu and Jiang, 2019), we use Accuracy (Acc) and Macro-F1 score as evaluation metrics. Besides, the paired t-test is conducted to test the significance of different methods. Finally, we report the average performance and standard deviation over 5 runs with random initialization. Our code and datasets are available at https://github.com/1429904852/KEF.

4.2 Compared Methods

We choose three kinds of baselines. **The first** is a frequently-used visual-based model *ResNet-Target*. **The second** is some classical text-based models, including *AE-LSTM* (Wang et al., 2016), *MemNet* (Tang et al., 2016b), *RAM* (Chen et al., 2017), *MGAN* (Fan et al., 2018), *BERT* (Devlin et al., 2019). **The third** is the recent multi-modal models, including *Res-MGAN*, *MIMN* (Xu et al., 2019b), *ESAFN* (Yu et al., 2019), *MMAP* (Zhou et al., 2021), *mPBERT* (Yu and Jiang, 2019), *ModalNet-BERT*(Zhang et al., 2021), EF-CapTrBERT (Khan and Fu, 2021), *TomBERT* (Yu and Jiang, 2019) and *Saliencybert* (Wang et al., 2021).

The *KEF* framework contains two plug-and-play components which can be easily combined or extended to existing attention-based methods. To better verify the effectiveness of *KEF*, we chose

Model	TWITTER-15		TWITTER-17	
	Acc	Macro-F1	Acc	Macro-F1
Visual				
Res-Target	59.88	46.48	58.59	53.98
Text				
AE-LSTM	70.30	63.43	61.67	57.97
MemNet	70.11	61.76	64.18	60.90
RAM	70.68	63.05	64.42	61.01
MGAN	71.17	64.21	64.75	61.46
BERT	74.15	68.86	68.15	65.23
Text + Visual				
Res-MGAN	71.65	63.88	66.37	63.04
MIMN	71.84	65.69	65.88	62.99
ESAFN	73.38	67.37	67.83	64.22
MMAP♣	73.50	66.53	67.31	64.34
mPBERT	75.79	71.07	69.61	67.12
ModalNet-Bert♣	76.71	70.93	69.55	67.28
EF-CapTrBERT★	77.01	71.79	69.00	66.71
Our Framework				
SaliencyBERT	77.03	72.36	69.69	67.19
KEF-SaliencyBERT	**78.15**†±0.33	**73.54**†±0.55	**71.88**†±0.21	**68.96**†±0.14
Δ	+1.12	+1.18	+2.19	+1.77
TomBERT	77.15	71.75	70.50	68.04
KEF-TomBERT	**78.68**†±0.30	**73.75**†±0.27	**72.12**†±0.15	**69.96**†±0.25
Δ	+1.53	+2.00	+1.62	+1.92

Table 2: Test accuracy on the TWITTER-15 and TWITTER-17 datasets (%). For the baseline model, the results with ♣ are produced with our implementation, the results with ★ are generated by running the code from (Khan and Fu, 2021), and the other results without symbols are retrieved from the original papers. For a fair comparison, we do not give the result of EF-CapTrBERT-DE from (Khan and Fu, 2021) since it use a domain-specific pre-trained encoder BERTweet from (Nguyen et al., 2020) instead of BERT-base. Δ denotes the difference between the performance of SaliencyBERT and KEF-SaliencyBERT, as well as TomBERT and KEF-TomBERT. We report the average performance and standard deviation over 5 runs. Best results are in bold. The marker † refers to significant test p-value $<$ 0.05 when comparing with other multi-modal models.

two latest BERT-based multimodal models as the foundations of our work, i.e., *TomBERT* and *Saliencybert*. In other words, we integrate *KEF* into *TomBERT* and *Saliencybert* to obtain the final model *KEF-TomBERT* and *KEF-Saliencybert*.

5 Results and Discussion

5.1 Main Results

The main experiment results are shown in Table 2. Based on the results, we can make a couple of observations: (1) We can see that pure visual-based methods perform very poorly, which implies that the associated images only play a supporting role to the text, and cannot be treated independently for target-oriented sentiment prediction; (2) For text-based methods, it is clear that *BERT* consistently outperforms all the baselines, which demonstrates the effectiveness of a strong pre-trained model. We attribute this to the fact that pre-trained models can provide rich semantic features; (3) *Res-MGAN* generally perform better than *MGAN*, which implies that the image information complements the textual information, and thus improves the performance for sentiment classification; (4) *TomBERT* and *Saliencybert* perform better than most multimodal models. A possible reason is that they employ self-modal and cross-modal multi-head attention to learn more robust representation; (5) *KEF-Saliencybert* and *KEF-TomBERT* both achieve competitive results on the TWITTER-15 and TWITTER-17 datasets. Specifically, compared to base version *TomBERT*, *KEF-TomBERT* obtains about 2.0% and 1.5% improvements in Macro-F1 and Accuracy. In contrast, *KEF-Saliencybert* outperforms *Saliencybert* by 1.5% and 1.7% on average. These results reveal that our framework have good compatibility; (6) *KEF-TomBERT* performs better than *KEF-Saliencybert* in most setting, which indicates that our framework is more effective for *TomBERT*.

Model	TWITTER-15		TWITTER-17	
	Acc	Macro-F1	Acc	Macro-F1
TomBERT	77.15	71.75	70.50	68.04
TomBERT+VAE	78.06±0.30	72.82±0.45	71.79±0.07	69.55±0.16
TomBERT+SPE	77.86±0.21	72.42±0.32	71.55±0.29	69.16±0.37
KEF-TomBERT	**78.68**±0.30	**73.75**±0.27	**72.12**±0.15	**69.96**±0.25
Δ (SPE)	+0.62	+0.93	+0.33	+0.41

Table 3: Ablation study of two main components. Δ represents the difference between the performance of *KEF-TomBERT* and *TomBERT+VAE*.

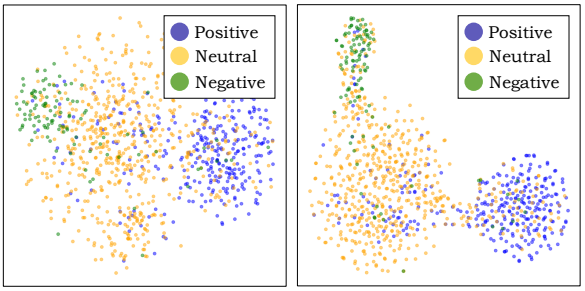

(a) TomBERT+VAE (b) KEF-TomBERT

Figure 4: Visualization of multimodal output representations for *TomBERT+VAE* and *KEF-TomBERT*.

5.2 Ablation Study

Without loss of generality, we choose *KEF-TomBERT* model for the ablation study to investigate the effects of different modules in *KEF*.

Model	TWITTER-15		TWITTER-17	
	Acc	Macro-F1	Acc	Macro-F1
TomBERT	77.15	71.75	70.50	68.04
TomBERT+MA	77.72±0.41	72.37±0.21	71.23±0.24	69.09±0.21
TomBERT+VAE	78.06±0.30	72.82±0.45	71.79±0.07	69.55±0.16
Δ (RL)	+0.34	+0.45	+0.56	+0.46

Table 4: Detailed ablation test over Visual Attention Enhancer. Δ represents the difference between the performance of *TomBERT+VAE* and *TomBERT+MA*.

Effects of Knowledge-enhanced Framework. We study the two main components of *KEF*: Visual Attention Enhancer (VAE) and Sentiment Prediction Enhancer (SPE). Based on the results reported in Table 3, we can observe the following: (1) In comparison with the base model *TomBERT*, *TomBERT+VAE* achieves competitive performance on both datasets, which validates the rationality of exploiting adjective-noun pairs to improve the visual attention capability; (2) After integrating *SPE* into *TomBERT+VAE*, *KEF-TomBERT* achieves the state-of-the-art performance, which demonstrates that *SPE* can improve the sentiment prediction capability through adjective-noun pairs; (3) *VAE* is more effective than *SPE*. This is explainable since the effectiveness of the attention mechanism is the core factor of sentiment prediction. Hence, it contributes more to our framework; (4) As depicted in Figure 4, we can see that multimodal output representations[7] learned by *KEF-TomBERT* are obviously more separable than those by *TomBERT+VAE*. This suggests that *SPE* can indeed reduce the difficulty of sentiment prediction.

Analysis over components of Visual Attention Enhancer. We further disassemble the Visual Attention Enhancer to see the contributions of the two sub-components: Mapping Method (MA) and Reconstruction Loss (RL). From the results in Table 4, we can observe that: (1) Compared to the base model *TomBERT*, *TomBERT+MA* achieves better performance, which indicates that the mapping method can help the opinion target capture corresponding visual representations; (2) After integrating *RL* into *TomBERT+MA*, *TomBERT+VAE* achieves further improvements, which demonstrates that the reconstruction loss can indeed improve the effectiveness of visual attention. This is consistent with our motivation.

[7]visualized by t-SNE (Van der Maaten and Hinton, 2008).

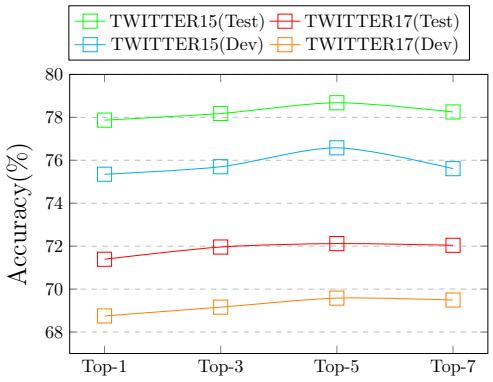

Figure 5: The results of *KEF-TomBERT* under different numbers of ANPs. Dev is short for development set.

5.3 Parameter Analysis

In this subsection, we explore the effect of hyperparameters on our model. Specifically, we tune the hyperparameters on the development set, and then evaluate the performance on the test set.

Effect of the number of ANPs. To verify the impact of ANPs on *KEF-TomBERT* model, we extract the top 1, 3, 5, and 7 ANPs from each image. The results are shown in Figure 5. Obviously, as the number of ANPs increases, the performance of *KEF-TomBERT* gets better. And when the number of ANPs is equal to 5, *KEF-TomBERT* achieves the best results. However, once the number of ANPs is greater than 5, the performance does not continue to increase and even begins to fall. The reason behind this may be that: each sentence contains at most 5 opinion targets, so it will bring some noise when the number of ANPs is greater than the maximum number of opinion targets.

Effects of λ_N and λ. To investigate the effect of hyperparameters λ_N and λ on the Visual Attention Enhancer (VAE), we conduct experiments for values set at 0.1 intervals in the range (0, 1). Figure 6(a) and Figure 6(b) show the performance of *TomBERT+MA* and *TomBERT+VAE* with different λ_N and λ on both datasets, respectively. Actually, as λ_N and λ increase, the performance of *TomBERT+MA* and *TomBERT+VAE* has an initial upward trend, and then flattens out or begins to fall. Initially, the nouns in ANPs help the opinion target capture the corresponding visual representations more accurately, thus improving the performance. However, once the weight λ_N or λ exceeds a certain value, the nouns begin to dominate the attention process and perform poorly. It makes sense because we inevitably extract the wrong ANPs, so it

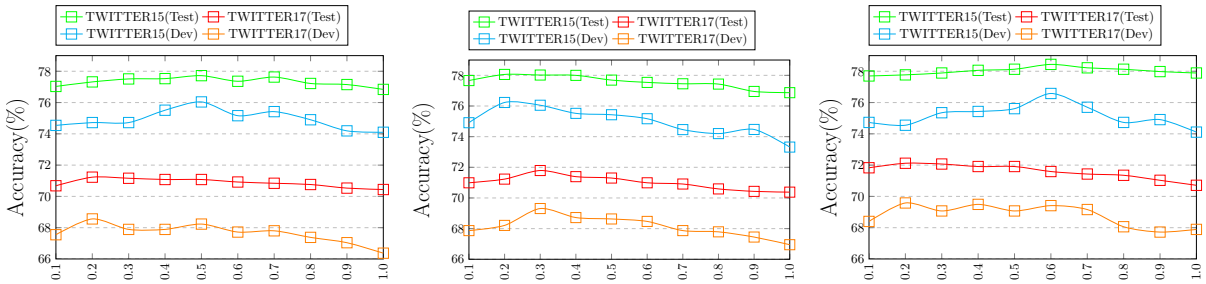

(a) Effect of λ_N (b) Effect of λ (c) Effect of λ_A

Figure 6: The effect of hyper-parameters λ_N, λ and λ_A on the development set and test set.

tends to have a negative impact when the adjectives account for a large proportion. This is also why we only use it as complementary information to the opinion target instead of directly replacing the opinion target. Finally, we set λ_N, λ to be {0.5, 0.2} on TWITTER-15 and {0.2, 0.3} on TWITTER-17.

Effects of λ_A. To analyze the effect of different λ_A on Sentiment Prediction Enhancer (SPE), we adjust λ_A in (0, 1) to conduct experiments and the step is 0.1. Figure 6(c) shows the results of *KEF-TomBERT* with varying λ_A on two datasets. According to the trend of the curve, we set λ_A to be 0.6 on TWITTER-15 and 0.2 on TWITTER-17, the reason behind this is similar to λ_N and λ.

Jointly observing Figure 5 and Figure 6, we found that the best results of the development set and test set are basically consistent, which indicates that our framework has good robustness.

5.4 Case Study

To better understand the advantage of Visual Attention Enhancer (*VAE*) and Sentiment Prediction Enhancer (*SPE*), we randomly select some samples from the Twitter dataset for a case study.

Effects of Visual Attention Enhancer. As shown in Figure 7(a), the base model *TomBERT* incorrectly predicts the sentiment of the opinion target "*Korkie*". It is reasonable since we found that *TomBERT* focuses on visual clues (highlighted by the yellow bounding boxes) that are not related to the opinion target. After integrating *VAE* into *TomBERT*, *TomBERT+VAE* maps fine-grained opinion target "*Korkie*" to the coarse-grained noun "*man*" in ANPs. With the aid of the noun "*man*", *TomBERT+VAE* successfully captures the target-relevant visual clues (highlighted by the red bounding boxes), thus giving the right predictions.

Effects of Sentiment Prediction Enhancer. As shown in Figure 7(b) and Figure 7(c), although *TomBERT+VAE* accurately captures the corresponding visual representations (i.e., smiling faces) of the opinion target, the diversification of smile expressions increases the difficulty of sentiment prediction, thus *TomBERT+VAE* incorrectly predict the sentiment over "*Sammy*" in Figure 7(c). After integrating *SPE* into *TomBERT+VAE*, *KEF-TomBERT* maps different smiling faces to the same adjectives "*happy*". Apparently, it is easier for the *KEF-TomBERT* to learn the mapping function between these "*happy*" and sentiment label "*positive*", thus making the right prediction.

5.5 Error Analysis

Although our model improves the overall performance of TMSC, *KEF-TomBERT* and *KEF-SaliencyBERT* make some wrong predictions due to extracting some noise ANPs. According to the statistics, for *KEF-TomBERT* model, about 5.50% and 7.05% samples of the dataset TWITTER-15 and TWITTER-17 are predicted successfully by the model *TomBERT* but incorrectly by *KEF-TomBERT*. In contrast, for *KEF-SaliencyBERT* model, almost 5.79% and 7.29% samples of the dataset TWITTER-15 and TWITTER-17 are predicted successfully by the model *SaliencyBERT* but incorrectly by *KEF-SaliencyBERT*. Additionally, the mapping method is unable to consistently find the correct noun for a given opinion target. There ought to be more advanced natural language processing techniques to address them.

6 Related Work

Target-oriented Sentiment Classification. As an important task in aspect-based sentiment analysis, Target-oriented Sentiment Classification (TSC) has been extensively studied in recent years. With

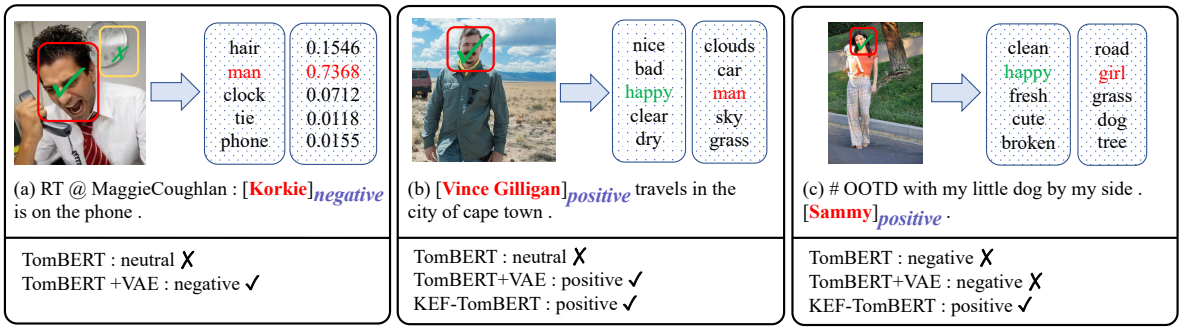

Figure 7: Predictions of *TomBERT*, *TomBERT+VAE* and *KEF-TomBERT* on three test samples. ✘and ✔denote incorrect and correct predictions. Opinion targets and their sentiment polarities are highlighted in the sentence. The yellow bounding box and red bounding box denote the visual clues that the opinion target focuses on under different methods. The numbers in the first sample indicate the similarity scores between the target and each noun in ANPs.

the development of deep learning, various neural networks have been designed for this task and have obtained promising results (Tang et al., 2016a; Li et al., 2018; Xue and Li, 2018). Recently, many studies have designed attention-based methods (Tang et al., 2016b; Wang et al., 2016; Ma et al., 2017; Chen et al., 2017; Fan et al., 2018; Zhao et al., 2020; He et al., 2018; Xu et al., 2019a; Hu et al., 2019; Xu et al., 2020; Wang et al., 2020) and graph-based methods (Zhang et al., 2019; Zhang and Qian, 2020; Huang and Carley, 2019; Sun et al., 2019; Tang et al., 2020; Chen et al., 2020) to model the interactions between the target and the context. However, none of the above works take visual modality into consideration, which can complement each other with these text-based methods.

Target-oriented Multimodal Sentiment Classification. With the growth of multimodal data (e.g., image) on the web, researchers proposed a new subtask of aspect-based sentiment analysis, namely Target-oriented Multimodal Sentiment Classification (TMSC), which has been explored in a few studies(Xu et al., 2019b; Yu et al., 2019; Yu and Jiang, 2019; Zhou et al., 2021; Zhang et al., 2021; Khan and Fu, 2021; Wang et al., 2021; Ling et al., 2022). Among them, based on the LSTM architecture, Xu et al. (2019b), Yu et al. (2019) and Zhou et al. (2021) proposed the MIMN, ESAFN and MMAP network to effectively model the *target-text* and *target-image* interactions. In contrast, Yu and Jiang (2019), Wang et al. (2021), Zhang et al. (2021) and Khan and Fu (2021) aim to explore the usefulness of the BERT architecture for TMSC and propose the TomBERT, Saliency-BERT, ModalNet-BERT and EF-CapTrBERT. Different from previous studies, this paper leverages the adjective-noun pairs (ANPs) to align text and image in the TMSC task.

7 Conclusion and Future Work

In this paper, we propose a novel knowledge-enhanced Framework (*KEF*) for the TMSC task. Specifically, with the aid of ANPs, we design two novel knowledge enhancers, Visual Attention Enhancer and Sentiment Prediction Enhancer, to improve the visual attention capability and sentiment prediction capability of the TMSC task. Results from numerous experiments indicate that our model achieves better performance than other state-of-the-art methods. Further analysis also validates the superiority of our framework.

In the future, we would like to apply our idea to other multimodal tasks since the adjective-noun pairs (ANPs) extracted from the image are easy to extend to other multimodal tasks, e.g., multi-modal entity linking, multi-modal machine comprehension and multi-modal dialogue generation.

Acknowledgements

We would like to thank the anonymous reviewers for their constructive comments. This work was supported by the National Natural Science Foundation of China (No. 61936012 and 61976114).

References

Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. 2015. Neural machine translation by jointly learning to align and translate. In *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*.

Damian Borth, Rongrong Ji, Tao Chen, Thomas Breuel, and Shih-Fu Chang. 2013. Large-scale visual sentiment ontology and detectors using adjective noun pairs. In *Proceedings of the 21st ACM international conference on Multimedia*, pages 223–232.

Chenhua Chen, Zhiyang Teng, and Yue Zhang. 2020. Inducing target-specific latent structures for aspect sentiment classification. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 5596–5607, Online. Association for Computational Linguistics.

Peng Chen, Zhongqian Sun, Lidong Bing, and Wei Yang. 2017. Recurrent attention network on memory for aspect sentiment analysis. In *Proceedings of the 2017 conference on empirical methods in natural language processing*, pages 452–461.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: pre-training of deep bidirectional transformers for language understanding. In *NAACL 2019*, pages 4171–4186. Association for Computational Linguistics.

Feifan Fan, Yansong Feng, and Dongyan Zhao. 2018. Multi-grained attention network for aspect-level sentiment classification. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 3433–3442.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. 2016. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778.

Ruidan He, Wee Sun Lee, Hwee Tou Ng, and Daniel Dahlmeier. 2018. Exploiting document knowledge for aspect-level sentiment classification. In *ACL 2018*, pages 579–585. Association for Computational Linguistics.

Geoffrey E. Hinton, Nitish Srivastava, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. 2012. Improving neural networks by preventing co-adaptation of feature detectors. *CoRR*, abs/1207.0580.

Mengting Hu, Shiwan Zhao, Li Zhang, Keke Cai, Zhong Su, Renhong Cheng, and Xiaowei Shen. 2019. CAN: Constrained attention networks for multi-aspect sentiment analysis. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 4601–4610, Hong Kong, China. Association for Computational Linguistics.

Binxuan Huang and Kathleen Carley. 2019. Syntax-aware aspect level sentiment classification with graph attention networks. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 5469–5477, Hong Kong, China. Association for Computational Linguistics.

Zaid Khan and Yun Fu. 2021. Exploiting BERT for multimodal target sentiment classification through input space translation. In *MM '21: ACM Multimedia Conference, Virtual Event, China, October 20 - 24, 2021*, pages 3034–3042. ACM.

Xin Li, Lidong Bing, Wai Lam, and Bei Shi. 2018. Transformation networks for target-oriented sentiment classification. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 946–956.

Yan Ling, Jianfei Yu, and Rui Xia. 2022. Vision-language pre-training for multimodal aspect-based sentiment analysis. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 2149–2159, Dublin, Ireland. Association for Computational Linguistics.

Bing Liu. 2012. *Sentiment Analysis and Opinion Mining*. Synthesis Lectures on Human Language Technologies. Morgan & Claypool Publishers.

Dehong Ma, Sujian Li, Xiaodong Zhang, and Houfeng Wang. 2017. Interactive attention networks for aspect-level sentiment classification. In *Proceedings of the Twenty-Sixth International Joint Conference on Artificial Intelligence*, pages 4068–4074. ijcai.org.

Dat Quoc Nguyen, Thanh Vu, and Anh Tuan Nguyen. 2020. Bertweet: A pre-trained language model for english tweets. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations, EMNLP 2020 - Demos, Online, November 16-20, 2020*, pages 9–14. Association for Computational Linguistics.

Bo Pang, Lillian Lee, et al. 2008. Opinion mining and sentiment analysis. *Foundations and Trends® in Information Retrieval*, 2(1–2):1–135.

Maria Pontiki, Dimitris Galanis, John Pavlopoulos, Harris Papageorgiou, Ion Androutsopoulos, and Suresh Manandhar. 2014. Semeval-2014 task 4: Aspect based sentiment analysis. In *Proceedings of the 8th International Workshop on Semantic Evaluation*, pages 27–35. The Association for Computer Linguistics.

Kai Sun, Richong Zhang, Samuel Mensah, Yongyi Mao, and Xudong Liu. 2019. Aspect-level sentiment analysis via convolution over dependency tree. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 5679–5688, Hong Kong, China. Association for Computational Linguistics.

Duyu Tang, Bing Qin, Xiaocheng Feng, and Ting Liu. 2016a. Effective lstms for target-dependent sentiment classification. In *Proceedings of COLING 2016, the 26th International Conference on Computational Linguistics: Technical Papers*, pages 3298–3307.

Duyu Tang, Bing Qin, and Ting Liu. 2016b. Aspect level sentiment classification with deep memory network. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*. The Association for Computational Linguistics.

Hao Tang, Donghong Ji, Chenliang Li, and Qiji Zhou. 2020. Dependency graph enhanced dual-transformer structure for aspect-based sentiment classification. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 6578–6588, Online. Association for Computational Linguistics.

Yao-Hung Hubert Tsai, Shaojie Bai, Paul Pu Liang, J. Zico Kolter, Louis-Philippe Morency, and Ruslan Salakhutdinov. 2019. Multimodal transformer for unaligned multimodal language sequences. In *Proceedings of the 57th Conference of the Association for Computational Linguistics, ACL 2019, Florence, Italy, July 28- August 2, 2019, Volume 1: Long Papers*, pages 6558–6569. Association for Computational Linguistics.

Laurens Van der Maaten and Geoffrey Hinton. 2008. Visualizing data using t-sne. *Journal of machine learning research*.

Jiawei Wang, Zhe Liu, Victor Sheng, Yuqing Song, and Chenjian Qiu. 2021. Saliencybert: Recurrent attention network for target-oriented multimodal sentiment classification. In *Pattern Recognition and Computer Vision - 4th Chinese Conference, PRCV 2021, Beijing, China, October 29 - November 1, 2021, Proceedings, Part III*, volume 13021 of *Lecture Notes in Computer Science*, pages 3–15. Springer.

Kai Wang, Weizhou Shen, Yunyi Yang, Xiaojun Quan, and Rui Wang. 2020. Relational graph attention network for aspect-based sentiment analysis. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 3229–3238, Online. Association for Computational Linguistics.

Yequan Wang, Minlie Huang, Li Zhao, et al. 2016. Attention-based lstm for aspect-level sentiment classification. In *Proceedings of the 2016 conference on empirical methods in natural language processing*, pages 606–615.

Hu Xu, Bing Liu, Lei Shu, and Philip Yu. 2019a. BERT post-training for review reading comprehension and aspect-based sentiment analysis. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 2324–2335, Minneapolis, Minnesota. Association for Computational Linguistics.

Lu Xu, Lidong Bing, Wei Lu, and Fei Huang. 2020. Aspect sentiment classification with aspect-specific opinion spans. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 3561–3567, Online. Association for Computational Linguistics.

Nan Xu, Wenji Mao, and Guandan Chen. 2019b. Multi-interactive memory network for aspect based multimodal sentiment analysis. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, pages 371–378.

Wei Xue and Tao Li. 2018. Aspect based sentiment analysis with gated convolutional networks. In *ACL 2018*, pages 2514–2523. Association for Computational Linguistics.

Jianfei Yu and Jing Jiang. 2019. Adapting BERT for target-oriented multimodal sentiment classification. In *Proceedings of the Twenty-Eighth International Joint Conference on Artificial Intelligence, IJCAI 2019, Macao, China, August 10-16, 2019*, pages 5408–5414. ijcai.org.

Jianfei Yu, Jing Jiang, and Rui Xia. 2019. Entity-sensitive attention and fusion network for entity-level multimodal sentiment classification. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 28:429–439.

Chen Zhang, Qiuchi Li, and Dawei Song. 2019. Aspect-based sentiment classification with aspect-specific graph convolutional networks. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 4568–4578, Hong Kong, China. Association for Computational Linguistics.

Mi Zhang and Tieyun Qian. 2020. Convolution over hierarchical syntactic and lexical graphs for aspect level sentiment analysis. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 3540–3549, Online. Association for Computational Linguistics.

Zhe Zhang, Zhu Wang, Xiaona Li, Nannan Liu, Bin Guo, and Zhiwen Yu. 2021. Modalnet: an aspect-level sentiment classification model by exploring multimodal data with fusion discriminant attentional network. *World Wide Web*, pages 1–18.

Fei Zhao, Zhen Wu, and Xinyu Dai. 2020. Attention transfer network for aspect-level sentiment classification. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 811–821, Barcelona, Spain (Online). International Committee on Computational Linguistics.

Jie Zhou, Jiabao Zhao, Jimmy Xiangji Huang, Qinmin Vivian Hu, and Liang He. 2021. Masad: A large-scale dataset for multimodal aspect-based sentiment analysis. *Neurocomputing*, 455:47–58.

Towards Exploiting Sticker for Multimodal Sentiment Analysis in Social Media: A New Dataset and Baseline

Feng Ge **Weizhao Li** **Haopeng Ren** **Yi Cai** *
South China University of Technology
logosg@foxmail.com
se_weizhao.li@mail.scut.edu.cn
renhp_scut@foxmail.com
ycai@scut.edu.cn

Abstract

Sentiment analysis in social media is challenging since posts are short of context. As a popular way to express emotion on social media, stickers related to these posts can supplement missing sentiments and help identify sentiments precisely. However, research about stickers has not been investigated further. To this end, we present a Chinese sticker-based multimodal dataset for the sentiment analysis task (CSMSA). Compared with previous real-world photo-based multimodal datasets, the CSMSA dataset focuses on stickers, conveying more vivid and moving emotions. The sticker-based multimodal sentiment analysis task is challenging in three aspects: inherent multimodality of stickers, significant inter-series variations between stickers, and complex multimodal sentiment fusion. We propose SAMSAM to address the above three challenges. Our model introduces a flexible masked self-attention mechanism to allow the dynamic interaction between post texts and stickers. The experimental results indicate that our model performs best compared with other models. More researches need to be devoted to this field. The dataset is publicly available at https://github.com/Logos23333/CSMSA.

1 Introduction

In recent years, social media has become more and more popular, and people tend to express opinions and emotions on social media (Greenwood et al., 2016). Collecting opinions and analyzing sentiment in social media can help a lot for marketing (Alalwan et al., 2017), campaign trends prediction, and so forth. The sentiment analysis is challenging in social media because of the lack of context (Khan and Fu, 2021). Stickers attached to posts can supplement missing sentiment and help to identify sentiment precisely. Such task of performing sentiment analysis with multiple data sources is called multimodal sentiment analysis.

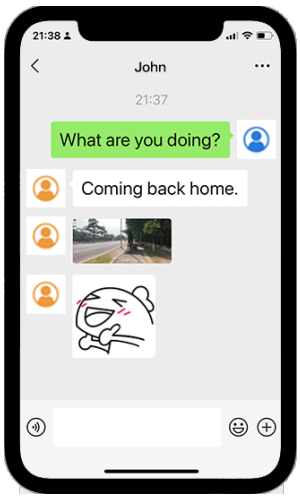

Figure 1: An example of online conversation. The first image is a photo that does not reflect any sentiment. The second is a sticker, expressing the emotional tendency of happiness. The sticker here contributes to supplementing the missing sentiment of the text.

Many research efforts have been devoted to multimodal sentiment analysis (Poria et al., 2015). However, most of the multimodal sentiment analysis datasets are based on real-world photos (Yu et al., 2020). Research focusing on sticker-based multimodal sentiment is limited. In social media, people tend to express the sentiment with stickers rather than real-world photo (Wang et al., 2019). Stickers can convey more vivid and direct emotion. For example, as shown in Figure 1, the first image is a photo related to the post context, but it does not reflect any sentiment. Instead, the second image, i.e., the sticker, shows a sense of happiness, revealing the missing sentiment of the corresponding texts. To this end, we introduce CSMSA, a challenging **C**hinese **S**ticker based **M**ultimodal dataset for **S**entiment **A**nalysis task in social media. CSMSA includes 28k text-sticker pairs, including 1.5k sentiment-annotated pairs, and 16k different stickers. This dataset is the first annotated sticker-based dataset for multimodal sentiment analysis to

*Corresponding author

Proceedings of the 29th International Conference on Computational Linguistics, pages 6795–6804
October 12–17, 2022.

the best of our knowledge. Our dataset will release to encourage research on multimodal sentiment analysis in social media.

The proposed CSMSA task is challenging in three aspects: First, stickers may be inherently multimodal because they are embedded with texts[1], while other datasets have only real-world photos. For example, as shown in Figure 2, sticker (a) is attached with the text "I'm beat" and sticker (b) is attached with the text "I'm so touched". The difficulty is that the same sticker with different sticker texts may vary significantly in sentiment. Second, stickers are highly different in styles, leading models to learn robust representations for the stickers following various distributions hardly (Huo et al., 2018). In contrast, most of the photo-based datasets only consist of human faces or food (Hasan et al., 2019), and the styles do not differ much. Considering the large inter-series variation of stickers, the CSMSA task requires models to adapt to different artistic styles. Third, the sentiment fusion of text and stickers is complex. For example, the text sentiment in Figure 2 is negative obviously. However, after combining the text and the sticker (b), the multimodal sentiment is positive, showing a sense of moving and touching. Multimodal sentiment does not always favor a single modality, making the fusion between modalities complex. In general, the first and second challenges are due to the difficulty of modeling the stickers, making it more challenging to fuse the sentiment with the post text.

We propose a Sticker-Aware Multimodal Sentiment Analysis Model (SAMSAM) to address the above three challenges. To address the first and second challenges, SAMSAM introduces the sticker text and sticker series tag, which can be seen as a further complement to sticker sentiment to model the sticker wholly and accurately. To address the third challenge, SAMSAM adopts the flexible masked attention mechanism to allow post texts and stickers to interact fully but selectively. The masked attention mechanism can help the model extract the most helpful information for the current sentiment judgment. We conduct experiments on the proposed CSMSA dataset. The experimental results show that our model performs best in the challenging CSMSA task. The ablation study indicates that the encoding of sticker texts and sticker series help in understanding multimodal sentiment,

[1]In the following, we call the text embedded in the sticker "sticker text" to avoid confusion with the post text.

Figure 2: An example for showing how the sticker influences sentiment in multimodal sentiment analysis. A sentence accompanied by different stickers may express reversed sentiments. An image accompanied by different sticker texts may also express reversed sentiments. The contents in brackets are translations from Chinese.

and the proposed masked self-attention mechanism can improve model performance significantly.

The contributions of this paper are as follows:

- We analyze the difference between real-world photos and stickers. We reveal three challenges of the CSMSA task: inherent multimodality, significant inter-series variation, and complex sentiment fusion.
- We propose a sticker-based human-annotated dataset. The dataset aims to test the model's ability to leverage stickers to supplement the missing sentiments of texts. To the best of our knowledge, we are the first to focus on stickers in multimodal sentiment analysis.
- We propose SAMSAM and step towards sticker-aware multimodal models. We conduct experiments against other models. Experimental results indicate that our model performs best compared with other models.

2 Related Work

2.1 Multimodal Sentiment Analysis Datasets

Multimodal sentiment analysis has attracted more and more attention recently. Due to the diversity of modalities and interactions between modalities, researchers have proposed a wide variety of multimodal sentiment analysis datasets, adapting to different real-world scenarios. Bagher Zadeh et al. (2018) propose CMU-MOSEI, one of the enormous datasets for sentiment analysis and emotion recognition with three modalities. The data of CMU-MOSEI came from YouTube monologue videos,

and the dataset was both sentiment and emotion annotated. Castro et al. (2019) propose a multimodal sarcasm detection dataset (MUStARD), which is compiled from famous TV shows and consists of audiovisual utterances annotated with sarcasm labels. Furthermore, Hasan et al. (2019) proposes a UR-FUNNY dataset, which contains text-audio-video three modalities for understanding humor. Yu et al. (2020) propose a Chinese multimodal sentiment analysis dataset with both multimodal and independent unimodal annotations. The primary difference between CSMSA and previous studies is that traditional multimodal datasets focus on videos of a speaker with their face, and few researchers pay attention to the sticker. Our proposed CSMSA dataset focuses on the sticker, studying the sentiment interaction between text and sticker.

2.2 Multimodal Sentiment Analysis Methods

Multimodal sentiment analysis mainly focuses on utilizing multiple resources to predict human emotions. Most multimodal models focus on three modalities: acoustic, visual, and text. Han et al. (2021b) propose Multimodal-infomax model. They propose a two-level mutual information maximization and design an entropy estimation module to facilitate the computation of Barber-Agakov (Agakov, 2004) lower bound and the training process. Hazarika et al. (2020) propose a novel framework MISA, which projects each modality to two distinct subspaces, reducing the modality gap and capturing characteristic features. Han et al. (2021a) propose Bi-Bimodal Fusion Network, which learns two text-related pairs of representations and generates the final prediction through the concatenation of four head representations.

2.3 Sticker-Related Research

Gao et al. (2020) propose to recommend an appropriate sticker to the user based on multi-turn dialog context history without any external labels. They release a dataset of 340K multi-turn dialog and sticker pairs. The dataset contains the most significant number of stickers available for sticker recommendation tasks. Fei et al. (2021) propose a new task Meme incorporated Open-domain Dialogue (MOD). The memes mentioned in (Fei et al., 2021) can be seen as the same thing with the sticker. A large-scale open-domain multimodal dialogue dataset incorporating great Internet memes into utterances is also released with the sentiment label of memes. However, their dataset is constructed by inserting memes as appropriately as possible into the multi-turn conversation dataset. This construction does not reflect the real interaction and supplementary between modalities because the memes inserted must not be too much different from the sentiment of the text. In the real world, the sentiment of stickers may be very different from text, and the sentiment fusion of text and sticker is very complex. For example, in the second example of Figure 3, the sentiment of the sticker is the opposite of the text, indicating that the sticker is not just complementary and it can play a leading role in multimodal sentiment analysis. In general, our dataset is more consistent with the data distribution of the real world.

3 Dataset

3.1 Data Collection

We collect the dataset from one of the most popular messaging apps. A huge number of stickers are released in this app, and everyone can use these stickers easily and conveniently in private or group chat. In this app, a sticker has a different identifier than other typical images, which allows us to focus on stickers and filter noises (e.g., screenshots and photos). All stickers are resized to a uniform size of 224 x 224 pixels.

We select eight public open chat groups consisting of active members, and the chat history of these groups is collected. To get multimodal data containing both text and sticker, we traverse every sticker in the chat history and collect the context of the sticker. The sticker and the text must be sent by the same person. Due to privacy concerns, we also filter out user information and anonymize user IDs.

3.2 Annotation

We employ four well-educated annotators to label the sentiments of text-sticker pairs. The annotators are asked to judge the text sentiment, image sentiment, multimodal sentiment, and *image_can_help* label. The *image_can_help* label indicates if the given sticker can assist in judging multimodal sentiment. This label can help researchers investigate how stickers' modality supplements the sentiment of the text. A dataset with multimodal and independent unimodal annotations allows researchers to study the interaction between modalities (Yu et al., 2020). Thus, our dataset can fully support unimodal sentiment analysis research.

To study the differences across sticker series, we

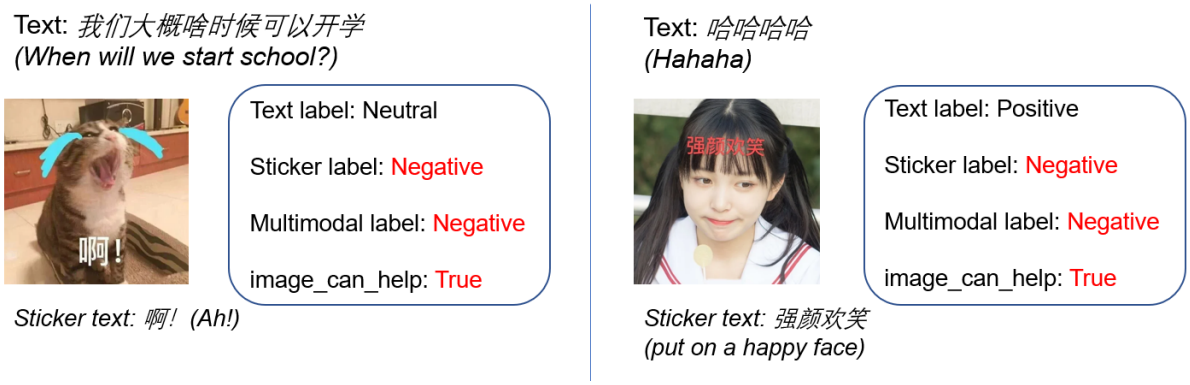

Figure 3: Two examples of the CSMSA dataset illustrate the sticker's effectiveness in sentiment analysis. For each text-sticker pair, in addition to multimodal annotations, the CSMSA dataset has independent unimodal annotations and an *image_can_help* label. The *image_can_help* label indicates whether the sticker can help to analyze sentiment.

Datasets	Size	# Sti.	# Ano.
MOD	45k	307	0
StickerChat	**340k**	174k	0
CSMSA	28k	16k	1.5k

Table 1: Comparison with other sticker-based datasets. # Sti. represents the number of stickers. # Ano. represents the number of samples with human-annotated multimodal sentiment label.

Task	Train	Valid	Test
Easy Task	942	314	314
Hard Task-1	1297	127	146
Hard Task-2	1290	130	150
Hard Task-3	1373	109	88

Table 2: The split statistics of the CSMSA dataset.

asked the annotators to group stickers into difference series. These sticker series include stickers with a similar style. Each annotator will give a confidence score used to calculate a final sentiment score considering the different views of all the annotators. The original data are randomly assigned to each annotator. Four annotators will decide every instance for the sake of the quality of the labeling. We weigh the confidence score of each annotator for the same label, and the label with the highest score will be preserved.

3.3 Statistics and Analysis

The final dataset contains 28k text-sticker pairs and 16k different stickers. The statistical comparison of the CSMSA dataset with existing sticker-based datasets is shown in Table 1. This dataset is the first annotated sticker-based dataset for multimodal sentiment analysis to the best of our knowledge. Our study is conducted on 1.5k annotated data, and the large-scale dataset is for future work.

We also conduct studies on the ability of text and images to express sentiment by using annotated unimodal labels. According to our statistics, nearly 52.1% of texts in the CSMSA dataset do not convey any sentiment, while sticker has a 73% probability of conveying sentiment. Meanwhile, 591 (37.6%) of the data are marked as *image_can_help* in the whole dataset. It means that if the stickers modality is ignored, about 37.6% of the data may not accurately determine their sentiment.

3.4 Case Analysis

Stickers can help to analyze sentiment. However, previous work considers only real-world photos, ignoring stickers heavily used in social media. In Figure 3, we show two examples of the CSMSA dataset to reveal how stickers can assist in determining sentiment. In the first example, the post text asks about the time of the school year without expressing any sentiment, while the sticker's sentiment is negative. The sticker creator attaches text "Ah!" to the sticker and adds blue "tears" to the cat, giving it a strong emotional impact. In the second example, the sentiment of the text is positive. However, we can know that it is a forced smile with the help of the sticker, and the multimodal sentiment is negative, which is the opposite of the text label.

3.5 Dataset Division

There are different series of stickers, and each series has an individual artistic style. Compared to the limited number of emojis, the number of stick-

Figure 4: An example shows the division of Hard Task-3 and the significant variation of styles between the stickers series.

ers is huge, and new stickers appear every day. In order to investigate whether the model can adapt to the newcomer stickers and the artistic style among different sticker series, we divide the dataset into four different ways: Easy Task, Hard Task-1, Hard Task-2, and Hard Task-3. The division is shown in Table 2. In particular, Easy Task randomly divides the whole dataset in the ratio of 6:2:2. The training set and validation set stickers may appear in the test set. Hard Task-1 randomly divides the stickers so that stickers from the test set will not appear in the training and validation sets. Hard Task-2 randomly divides the sticker series so that the sticker series from the test set will not appear in the training and validation sets. Hard Task-3 partitions the stickers into three categories: human face-based, pet-based, and cartoon character-based. As shown in Figure 4, there is a considerable difference in style between them. The Hard Task-1 measures whether the model can adapt to the new sticker. The Hard Task-2 measures whether the model can adapt to the new sticker series. The Hard Task-3 measures whether the model can adapt to unseen sticker series, which varies greatly from the previous one.

4 Model

4.1 Task Definition

Formally, given an post text $X = (x_1, x_2, \ldots, x_m)$, a sticker I, the sticker text $S = (s_1, s_2, \ldots, s_n)$ and the sticker series E, the CSMSA task requires model to predict the multimodal sentiment label $y \in \{Positive, Negative, Neutral\}$. x_i denotes the i-th word in the post text, s_i denotes the i-th word in the sticker text, m is the length of post text, n is the length of sticker text.

4.2 Feature Layer

Post Text Encoder. BERT (Devlin et al., 2019; Sun et al., 2019) is a transformer-based pre-trained model which uses a Masked Language Model to predict randomly masked or replaced words. We use the bert-base-Chinese[2] weights provided by google to fine-tune the proposed model. For classification tasks, BERT takes the final hidden state of the first token [CLS], i.e., h, as the representation of the whole sequence.

$$h = BERT(X) \tag{1}$$

Sticker Encoder. ResNet (He et al., 2015) is a classic neural network used as a backbone for many computer vision tasks. We use the ResNet34 as our image encoder to obtain representations of stickers. An input sticker I is resized to 224×224 and then sent through the ResNet model to obtain sticker representations f.

$$f = ResNet(I) \tag{2}$$

Sticker Text Encoder. The text within a sticker is an essential component for understanding the sticker. To extract the textual information contained in sticker, we introduce PaddleOCR (Du et al., 2020) to recognize texts S within a sticker. The text feature u of sticker text S is obtained through LSTM (Hochreiter and Schmidhuber, 1997), and this text feature can assist in fusing multimodal information and determining the final sentiment.

$$u = LSTM(S) \tag{3}$$

Sticker Series Embedding. The differences in artistic styles cause significant inter-series variations, making the model difficult to understand stickers. For the model to distinguish the different artistic styles of the series explicitly, the sticker series tag e is fed as a embedding.

$$v = Embedding(E) \tag{4}$$

4.3 Interaction Layer

There are four types of information in the interaction layer: post text h, sticker f, sticker text u, and sticker series v, which make up a tuple (h, f, u, v). Accordingly, four inter-information relations need to be considered:

[2] https://huggingface.co/bert-base-chinese

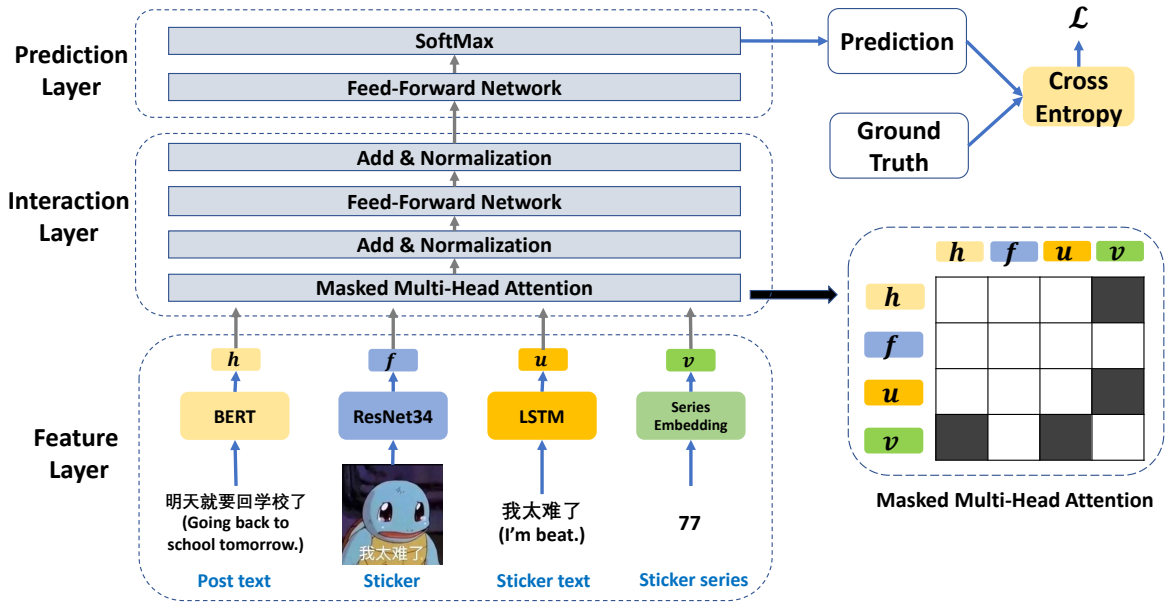

Figure 5: The overview of the proposed SAMSAM and masked multi-head attention. SAMSAM consists of three parts, the feature layer, the interaction layer, and the prediction layer.

(1) The alignment between h and f is beneficial to fusing the sentiment between post text and sticker, we need to find the most relevant text-image information according to the relations between h and f.

(2) As the sticker text is an essential way of expressing emotion, we need to find the most relevant sticker information according to the relations between h and u.

(3) Similar to (1) and (2), we want to fuse the sentiment of the sticker and find the most relevant sticker information according to the relations between f and u.

(4) We also want to let the model understand the current sticker series style according to the relations between f and v.

We adopt multi-head attention in the interaction layer considering the above factors. Multi-head attention allows $h-f$, $h-u$, $f-u$ and $f-v$ pair to interact fully. Since post text and sticker text are not intrinsically related to sticker series, we need to prevent some of the information from interacting explicitly for noise mitigation. Thus we adopt a well-designed attention mask $\mathcal{M}$ in addition to the original multi-head attention. The overview of the interaction layer and mask is shown in Figure 5.

In the interaction layer, we employ three feed-forward networks with different parameters to project the C_i into three different spaces:

$$Q_i = FN(C_i),\ K_i = FN(C_i),\ V_i = FN(C_i) \tag{5}$$

C_i represents the i-th source of C, i.e, h, f, u and v. The model then takes each Q_i to attend to K_j, and uses the attention weights $\alpha_{i,j}$ to gain the weighted sum of V_i:

$$C'_i = \sum_{j=1}^{4} \alpha_{i,j} * V_i \tag{6}$$

$$\alpha_{i,j} = \frac{exp(Q_i * K_j \odot \mathcal{M})}{\sum_{k=1}^{4} exp(Q_i * K_k \odot \mathcal{M})} \tag{7}$$

4.4 Prediction Layer

To aggregates information in different sources, we stack the hidden states in C' as the hybrid output: $O = [C'_1; C'_2; C'_3; C'_4]$. Finally, we feed O to a one-layer feed-forward network followed by a softmax function for the sentiment prediction distribution:

$$P(y|X, I, S, E) = softmax(FFN(O)) \tag{8}$$

To optimize all the parameters in our SAMSAM, the objective is to minimize the standard cross-entropy loss function:

$$\mathcal{L} = -\frac{1}{|\mathcal{D}|} \sum_{c \in \mathcal{D}} logP(y^{(c)}|X^{(c)}, I^{(c)}, S^{(c)}, E^{(c)}) \tag{9}$$

where $\mathcal{D}$ denotes all samples in the dataset.

5 Experimental Setup

5.1 Compared Methods

BERT BERT is one of the state-of-the-art methods to address classification tasks. We implement BERT as a text-only baseline.

BERT-ST The input sentence of BERT-ST is the concatenation of post text and sticker text, separated by [SEP]. We implement BERT-ST as a text-only baseline.

RoBERTa RoBERTa is an optimized method for BERT. We implement RoBERTa as a strong text-only baseline.

ResNet34 We implement ResNet34 as a image-only baseline model.

mBERT Yu and Jiang (2019) design mBERT on top of the baseline BERT architecture. They directly concatenate the image features with the final hidden states of the post text.

MMTF Li (2021) propose a Multimodal Transformers (MMTF) for Meme Classification task. They propose a multimodal attention layer to fully interact with the text and image based on the cross-attention mechanism. We adopt MMTF as a multimodal baseline.

5.2 Implementation Details

All the models are trained in 100 epochs with an NVIDIA GTX 2080Ti, and we use the best model on the validation set for evaluation. We use Pytorch (Paszke et al., 2017) and HuggingFace's transformers (Wolf et al., 2020) to implement our model. AdamW (Loshchilov and Hutter, 2019) optimizer is used to optimize our model with learning rate 5e-7. The batch size is 6.

6 Experimental Results

6.1 Overall Performance

Table 3 shows the experimental results of our comparison with other models and ablation study.

The experimental results show that SAMSAM achieves the best results among multimodal and unimodal models. Multimodal models MMTF and mBERT are not well adapted to the CSMSA task because they ignore the sticker text embedded in the sticker, and the styles of stickers vary significantly. Unimodal models perform poorly because they focus on a single modality and ignore the complementary effect between modalities.

The text-only BERT and RoBERTa models do not consider the sentiment of the stickers, resulting in their poor performance. The results of BERT-ST are worse than BERT because of two reasons. The first reason is the error propagation caused by the inaccurate recognition of OCR. The second is that the sentiment of the sticker must be combined with the image feature. Otherwise, the wrong sticker sentiment will further affect the multimodal sentiment judgment. The image-only model ResNet34 does not consider the text's sentiment, so it performs worse than the other multimodal models. The image-only models perform better than the text-only models on Easy Task and Hard Task-1 because the short text lacks context, and it is not easy to predict the sentiment, while the sentiment of the sticker is often stronger and more direct than the text. The image-only models perform worse on Hard Task-2 and Hard Task-3 because models fail to adapt to the new sticker series.

Generally, we can see that multimodal and image-only models perform worst on Hard Task-3 and best on Easy Task. This result also validates the challenge of the CSMSA task. It is difficult to capture and recognize stickers' sentiments because of the large inter-series variation of stickers. Models failed to generalize well when a new sticker series appeared, making it lousy to predict multimodal sentiment. Unfortunately, new sticker series appear every day. The CSMSA task is challenging and needs to be further researched.

6.2 Ablation Study

As shown in Table 3, the accuracy and F1 score decreased on almost all tasks when the mask, post text, sticker series, and sticker text were removed, respectively. The mask helps the model extract the most helpful information and eliminate some noise. The sticker text helps the model understand the sticker's sentiment and thus correctly predict multimodal sentiment. The series embedding helps the model capture the stylistic features of each series. Similarly, post text and image are also essential for multimodal sentiment judgment.

6.3 Case Study

We show some analysis examples in Figure 6. In the first example, because the post text has the word "reported", its sentiment is Negative, which leads to the misjudgment of the text-only model BERT. We know that the sticker sentiment is Positive because of the blush and smile, and the sticker text

	Methods	Easy Task		Hard Task-1		Hard Task-2		Hard Task-3	
		Acc.	F1	Acc.	F1	Acc.	F1	Acc.	F1
Text-only	BERT	0.6178	0.5764	0.5274	0.4709	0.5467	0.5102	0.5114	0.3617
	BERT-ST	0.6146	0.5609	0.4932	0.4772	0.5333	0.4769	0.5227	0.3814
	RoBERTa	0.6210	0.5941	0.5479	0.4892	0.5133	0.5216	0.5341	0.4194
Image-only	ResNet34	0.6178	0.5701	0.5753	0.5530	0.5342	0.4778	0.4886	0.3484
Multimodal	mBERT	0.5924	0.5576	0.5753	0.5156	0.5600	0.5361	0.5341	0.4037
	MMTF	0.5955	0.5374	0.5479	0.4886	0.5267	0.4876	0.5227	**0.4428**
	SAMSAM	**0.6369**	0.6180	**0.5959**	**0.5669**	**0.5533**	0.5179	**0.5455**	0.4265
Ablation Study	w/o MASK	0.6306	0.6060	0.5685	0.5483	0.5133	0.4929	0.4773	0.3710
	w/o IMG	0.6274	**0.6199**	0.5685	0.5467	0.5200	**0.5251**	0.5114	0.3913
	w/o PT	0.6210	0.5665	0.5411	0.4845	0.5267	0.4924	0.5114	0.3782
	w/o SE	0.6146	0.5594	0.5685	0.5112	0.5467	0.4978	0.5227	0.4133
	w/o ST	0.6242	0.6179	0.5890	0.5664	0.5267	0.5006	0.5117	0.3925

Table 3: Overview of the experimental results. Acc. represents accuracy, and the F1 represents the weighted F1 score. MASK represents the mask mechanism. IMG represents the image feature. PT represents the post text. SE represents the sticker series embedding, and ST represents sticker text.

Post Text: *我直接举报了*
(I reported it directly.)

Sticker text: 让人怪不好意思的
(Sorry not sorry.)

× **BERT: Negative**
√ **ResNet: Positive**
× **MMTF: Negative**
√ **SAMSAM: Positive**
Ground Truth: Positive

Post Text: *你要做什么呀*
(What do you want to do?)

Sticker text: *一个耿直的微笑*
(A straight smile.)

× **BERT: Neutral**
× **ResNet: Neutral**
× **MMTF: Neutral**
√ **SAMSAM: Positive**
Ground Truth: Positive

Figure 6: Examples of multimodal sentiment analysis produced by different models on CSMSA dataset.

"Sorry and not sorry" here also further verifies the sentiment. After combining the text and sticker, we know that although the text says about "report" people and conveys some negative emotions. However, the user is happy with the situation rather than sad. In the second example, the post text does not reflect any emotion, indicating that the sentiment label is Neutral. The expression of the dog in the sticker is very subtle, in which there has a smile but not obvious. After combining the word "smile" in the sticker text, we can learn that the sticker's sentiment is Positive. The text-only model BERT fails to consider the sticker sentiment, resulting in the wrong judgment. The image-only model and MMTF are biased in judging the sticker's sentiment because they failed to combine the sticker text, which led to their incorrect judgment of multimodal sentiment. After combining the sticker text, our model correctly predicts the multimodal sentiment, i.e., Positive.

7 Conclusion

In this paper, we propose to focus on stickers rather than the real-world photo in the field of multimodal sentiment analysis. The sentiment fusion of stickers and texts is complex and challenging. Compared with real-world photos, the sticker is inherently multimodal, and it has a significant inter-series variation, making it difficult to encode. We propose a sticker-based dataset for the sentiment analysis task, with 1.5k text-sticker human-annotated pairs. This dataset is the first annotated sticker-based dataset for multimodal sentiment analysis to the best of our knowledge. Experimental results validate the challenges of the CSMSA task. Previous models cannot be directly applied to this task. More researches need to be devoted to this field.

Acknowledgements

This work was supported by National Natural Science Foundation of China (62076100), and Fundamental Research Funds for the Central Universities, SCUT(x2rjD2220050), the Science and Technology Planning Project of Guangdong Province (2020B0101100002).

References

David Barber Felix Agakov. 2004. The im algorithm: a variational approach to information maximization. *Advances in neural information processing systems*, 16(320):201.

Ali Abdallah Alalwan, Nripendra P. Rana, Yogesh K. Dwivedi, and Raed Algharabat. 2017. Social media in marketing: A review and analysis of the existing literature. *Telematics and Informatics*, 34(7):1177–1190.

AmirAli Bagher Zadeh, Paul Pu Liang, Soujanya Poria, Erik Cambria, and Louis-Philippe Morency. 2018. Multimodal language analysis in the wild: CMU-MOSEI dataset and interpretable dynamic fusion graph. In *ACL*, pages 2236–2246.

Santiago Castro, Devamanyu Hazarika, Verónica Pérez-Rosas, Roger Zimmermann, Rada Mihalcea, and Soujanya Poria. 2019. Towards multimodal sarcasm detection (an _Obviously_ perfect paper). In *ACL*, pages 4619–4629.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *NAACL*, pages 4171–4186.

Yuning Du, Chenxia Li, Ruoyu Guo, Xiaoting Yin, Weiwei Liu, Jun Zhou, Yifan Bai, Zilin Yu, Yehua Yang, Qingqing Dang, and Haoshuang Wang. 2020. PP-OCR: A practical ultra lightweight OCR system. *CoRR*, abs/2009.09941.

Zhengcong Fei, Zekang Li, Jinchao Zhang, Yang Feng, and Jie Zhou. 2021. Towards expressive communication with internet memes: A new multimodal conversation dataset and benchmark. *CoRR*, abs/2109.01839.

Shen Gao, Xiuying Chen, Chang Liu, Li Liu, Dongyan Zhao, and Rui Yan. 2020. *Learning to Respond with Stickers: A Framework of Unifying Multi-Modality in Multi-Turn Dialog*, page 1138–1148. Association for Computing Machinery.

Shannon Greenwood, Andrew Perrin, and Maeve Duggan. 2016. Social media update 2016. *Pew Research Center*, 11(2):1–18.

Wei Han, Hui Chen, Alexander Gelbukh, Amir Zadeh, Louis-philippe Morency, and Soujanya Poria. 2021a. *Bi-Bimodal Modality Fusion for Correlation-Controlled Multimodal Sentiment Analysis*, page 6–15. Association for Computing Machinery.

Wei Han, Hui Chen, and Soujanya Poria. 2021b. Improving multimodal fusion with hierarchical mutual information maximization for multimodal sentiment analysis. *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*.

Md Kamrul Hasan, Wasifur Rahman, AmirAli Bagher Zadeh, Jianyuan Zhong, Md Iftekhar Tanveer, Louis-Philippe Morency, and Mohammed (Ehsan) Hoque. 2019. UR-FUNNY: A multimodal language dataset for understanding humor. In *EMNLP-IJCNLP*, pages 2046–2056.

Devamanyu Hazarika, Roger Zimmermann, and Soujanya Poria. 2020. *MISA: Modality-Invariant and -Specific Representations for Multimodal Sentiment Analysis*, page 1122–1131. Association for Computing Machinery.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. 2015. Deep residual learning for image recognition. *CoRR*, abs/1512.03385.

Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long short-term memory. *Neural computation*, 9(8):1735–1780.

Jing Huo, Wenbin Li, Yinghuan Shi, Yang Gao, and Hujun Yin. 2018. Webcaricature: a benchmark for caricature recognition. In *British Machine Vision Conference*.

Zaid Khan and Yun Fu. 2021. *Exploiting BERT for Multimodal Target Sentiment Classification through Input Space Translation*, page 3034–3042. Association for Computing Machinery, New York, NY, USA.

Zichao Li. 2021. Codewithzichao@DravidianLangTech-EACL2021: Exploring multimodal transformers for meme classification in Tamil language. In *Proceedings of the First Workshop on Speech and Language Technologies for Dravidian Languages*, pages 352–356.

Ilya Loshchilov and Frank Hutter. 2019. Decoupled weight decay regularization. In *7th International Conference on Learning Representations, ICLR 2019, New Orleans, LA, USA, May 6-9, 2019*. OpenReview.net.

Adam Paszke, Sam Gross, Soumith Chintala, Gregory Chanan, Edward Yang, Zachary DeVito, Zeming Lin, Alban Desmaison, Luca Antiga, and Adam Lerer. 2017. Automatic differentiation in pytorch.

Soujanya Poria, Erik Cambria, and Alexander Gelbukh. 2015. Deep convolutional neural network textual features and multiple kernel learning for utterance-level multimodal sentiment analysis. In *EMNLP*, pages 2539–2544.

Chi Sun, Xipeng Qiu, Yige Xu, and Xuanjing Huang. 2019. How to fine-tune bert for text classification? In *Chinese Computational Linguistics*, pages 194–206. Springer International Publishing.

Yuan Wang, Yukun Li, Xinning Gui, Yubo Kou, and Fenglian Liu. 2019. Culturally-embedded visual literacy: A study of impression management via emoticon, emoji, sticker, and meme on social media in china. *Proceedings of the ACM on Human-Computer Interaction*, 3(CSCW):1–24.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Remi Louf, Morgan Funtowicz, Joe Davison, Sam Shleifer, Patrick von Platen, Clara Ma, Yacine Jernite, Julien Plu, Canwen Xu, Teven Le Scao, Sylvain Gugger, Mariama Drame, Quentin Lhoest, and Alexander Rush. 2020. Transformers: State-of-the-art natural language processing. In *EMNLP*, pages 38–45.

Jianfei Yu and Jing Jiang. 2019. Adapting BERT for target-oriented multimodal sentiment classification. In *Proceedings of the Twenty-Eighth International Joint Conference on Artificial Intelligence, IJCAI 2019, Macao, China, August 10-16, 2019*, pages 5408–5414. ijcai.org.

Wenmeng Yu, Hua Xu, Fanyang Meng, Yilin Zhu, Yixiao Ma, Jiele Wu, Jiyun Zou, and Kaicheng Yang. 2020. CH-SIMS: A Chinese multimodal sentiment analysis dataset with fine-grained annotation of modality. In *ACL*, pages 3718–3727.

Natural Language Inference Prompts for Zero-shot Emotion Classification in Text across Corpora

Flor Miriam Plaza-del-Arco[1,2], **María-Teresa Martín-Valdivia**[1], **Roman Klinger**[2]
[1]SINAI, Computer Science Department, CEATIC, Universidad de Jaén, Spain
[2]Institut für Maschinelle Sprachverarbeitung, University of Stuttgart, Germany
{fmplaza, maite}@ujaen.es, klinger@ims.uni-stuttgart.de

Abstract

Within textual emotion classification, the set of relevant labels depends on the domain and application scenario and might not be known at the time of model development. This conflicts with the classical paradigm of supervised learning in which the labels need to be predefined. A solution to obtain a model with a flexible set of labels is to use the paradigm of zero-shot learning as a natural language inference task, which in addition adds the advantage of not needing any labeled training data. This raises the question how to prompt a natural language inference model for zero-shot learning emotion classification. Options for prompt formulations include the emotion name *anger* alone or the statement "This text expresses *anger*". With this paper, we analyze how sensitive a natural language inference-based zero-shot-learning classifier is to such changes to the prompt under consideration of the corpus: How carefully does the prompt need to be selected? We perform experiments on an established set of emotion datasets presenting different language registers according to different sources (tweets, events, blogs) with three natural language inference models and show that indeed the choice of a particular prompt formulation needs to fit to the corpus. We show that this challenge can be tackled with combinations of multiple prompts. Such ensemble is more robust across corpora than individual prompts and shows nearly the same performance as the individual best prompt for a particular corpus.

1 Introduction

To enable communication about emotions, there exists a set of various emotion names, for instance those labeled as *basic emotions*, by Ekman (1992) or Plutchik (2001) (*anger*, *fear*, *joy*, *sadness*, *disgust*, *surprise*, *trust*, *anticipation*). While such psychological models influence natural language processing and emotion categorization approaches, the choice of emotion concepts is context-dependent. For instance, Scherer and Wallbott (1997) and Troiano et al. (2019) opted to use *guilt* and *shame* as self-directed emotions in addition to Ekman's basic emotions, to analyze self-reports of events. For the context of the perception of art it is more appropriate to consider *aesthetic emotions* (Menninghaus et al., 2019; Haider et al., 2020), like *beauty*, *sublime*, *inspiration*, *nostalgia*, and *melancholia*.

This leads to a potential gap between concepts in emotion-related training data and the application domain, purely because the label set is not compatible. One solution is to resort to so-called dimensional models, in which emotion names are located in vector spaces of affect (valence, arousal, Preoţiuc-Pietro et al., 2016; Buechel and Hahn, 2017) or cognitive appraisal (e.g., regarding *responsibility*, *certainty*, *pleasantness*, *control*, *attention* with respect to a stimulus event, Hofmann et al., 2020; Troiano et al., 2023). In these vector spaces, classes can be assigned to predicted points with a nearest-neighbor approach, even if these classes have not been seen during training. This approach, however, has the disadvantage of the so-called hubness problem (Lazaridou et al., 2015), namely that the distance between predictions and concepts that have been seen during training tends to be smaller than to novel concepts. We acknowledge ongoing research to tackle this problem (Park et al., 2021;

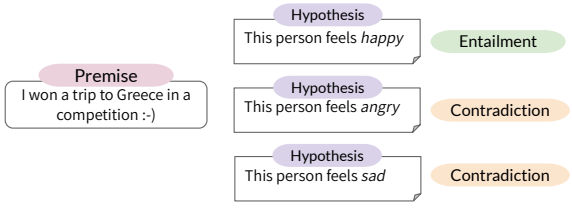

Figure 1: An example of the application of NLI to ZSL emotion classification. Given the premise "I won a trip to Greece in a competition", three hypotheses represent the emotions (*joy*, *anger*, *sadness*). The representation of *joy* is entailed and therefore predicted.

Proceedings of the 29th International Conference on Computational Linguistics, pages 6805–6817
October 12–17, 2022.

Buechel et al., 2021).

We take a different, more direct, route to obtaining classifiers for discrete emotion categories which are not known at system development time, namely zero-shot learning (ZSL). One instantiation of such ZSL systems is via natural language inference models (ZSL-NLI), in which the inference task needs to perform reasoning (Yin et al., 2019). Consequently, the idea of implementing ZSL-NLI models is not by exemplification and optimizing a classifier, but developing appropriate natural language class name representations which we refer to as *prompts*. We see an example for the application of an NLI model to ZSL emotion classification in Figure 1 – the NLI model needs to decide if the hypothesis (a prompt which represents the class label) entails the premise (which corresponds to the instance to be classified). This paradigm raises the question (which we answer in this paper) of how to formulate the emotion prompt and how much the design choice of the prompt needs to fit the dataset.

Manually developing intuitive templates based on human data introspection may be the most natural method to produce prompts. In this paper, we provide manually created templates to probe emotion classification in an NLI-ZSL setup and we analyze whether prompts are language-register dependent according to various corpora (tweets, event descriptions, blog posts). To accomplish this aim, we perform experiments on an established set of emotion datasets with three NLI models and we show that (1) prompts are indeed corpus-specific and that the differences follow the same pattern across different pretrained NLI models, (2) that an ensemble of multiple prompts behaves more robustly across corpora, and (3) the representation of the emotion concept as part of the textual prompt is an important element, benefiting from representations with synonyms and related concepts, instead of just the emotion name. Our code is publicly available at https://github.com/fmplaza/zsl_nli_emotion_prompts.

2 Related Work

2.1 Emotion Classification

Emotion analysis has become a major area of research in NLP which comprises a variety of tasks, including emotion stimulus or cause detection (Li et al., 2021; Doan Dang et al., 2021) and emotion intensity prediction (Mohammad and Bravo-Marquez, 2017; Köper et al., 2017). The task of emotion classification received most attention in recent years (Bostan and Klinger, 2018; Mohammad et al., 2018a; Plaza-del-Arco et al., 2020, i.a.).

Emotion classification aims at mapping textual units to an emotion category. The categories often rely on psychological theories such as those proposed by Ekman (1992) (*anger*, *fear*, *sadness*, *joy*, *disgust*, *surprise*), or the dimensional model of Plutchik (2001) (adding *trust* and *anticipation*). However, neither are all these basic emotions relevant in all domains, nor are they sufficient. For instance, in the education field, D'mello and Graesser (2007) found *boredom*, *confusion*, *flow*, *frustration*, and *delight* to be more relevant than *fear* or *disgust*. Sreeja and Mahalaksmi (2015) reveal that emotions such as *love*, *hate*, and *courage* are necessary to model the emotional perception of poetry. Bostan et al. (2020) identify *annoyance*, *guilt*, *pessimism*, or *optimism* to be important to analyze news headlines.

A strategy to avoid specification of discrete categories is the use of dimensional spaces that consider valence, arousal, and dominance (VAD, Russell and Mehrabian, 1977). Smith and Ellsworth (1985) claim that this model does not represent important difference between emotions and propose an alternative dimensional model based on cognitive appraisal, which has recently been used for text analysis (Hofmann et al., 2020; Stranisci et al., 2022; Troiano et al., 2023). Independent of the classification or regression approach, nearly all recently proposed systems rely on transfer learning from general language representations. We refer the reader to recent shared task surveys for a more comprehensive overview (Mohammad et al., 2018b; Tafreshi et al., 2021; Plaza-del Arco et al., 2021).

2.2 Zero-shot Learning

Zero-shot learning (ZSL) aims at performing predictions without having seen labeled training examples specific for the concrete task. Zero-shot methods typically work by associating seen and unseen classes using auxiliary information, which encodes observable distinguishing properties of instances (Xian et al., 2019). In NLP, the term is used predominantly either to refer to cross-lingual transfer to languages that have not been seen at training time (change of the language), or to predict classes that have not been seen (change of the labels, Wang et al., 2019). Our work falls in the second category.

Various approaches exist to perform zero-shot text classification. One approach represents labels in an embedding space (Socher et al., 2013; Sappadla et al., 2016; Rios and Kavuluru, 2018, i.a.). A model is trained to predict the respective embedding vectors for categorical labels. At test time, embeddings of novel labels need to be known and will be assigned if the distance between the predicted embedding and the label embedding is small. This method suffers from the hubness problem, that is, when the semantic label embeddings are close to each other, the projection of labels to the semantic space forms hubs (Radovanovic; et al., 2010).

Another approach is to use transformer language models to classify if a label embedding is compatible with an instance embedding (Brown et al., 2020). To this end, no labeled examples are provided at training phase but an instruction in natural language is given to the model to interpret the label class (the *prompt*). An instance of this approach is *Task-Aware Representations* (TARS, Halder et al., 2020) who separate the instance text and the class label by the special separator token `[SEP]` in BERT (Devlin et al., 2019).

An alternative is to treat ZSL as textual entailment. Following this approach, Yin et al. (2019) propose a sequence-pair classifier that takes two sentences as input (a premise and a hypothesis) and decides whether they entail or contradict each other. They study various formulations of the labels as hypotheses and evaluate the method in various NLP tasks including topic detection, situation detection, and emotion classification. In their evaluation, emotion classification turns out to be most challenging. Another study that conducted prompt engineering in NLI models proposes probabilistic ZSL ensembles for emotion classification (Basile et al., 2021). The authors experiment with the same prompts as Yin et al. (2019) and aggregate the predictions of multiple NLI models using Multi-Annotator Competence Estimation (MACE), a method developed for modelling crowdsourced annotations.

Our work on ZSL for emotion classification differs from previous approaches as follows. We analyze whether prompts are corpus-specific and propose an ensemble of multiple prompts to achieve a classifier which is more robust across corpora (in contrast to an ensemble of multiple NLI models in the work by Basile et al. (2021)). Further, we analyze if the introduction of more knowledge about the emotion in the prompt through emotion synonyms and related concepts helps its interpretation in the NLI models.

3 Methods

In this section, we explain how we apply NLI for ZSL emotion classification and propose a collection of prompts to contextualize and represent the emotion concept in different corpora. In addition, we propose a prompt ensemble which is more robust across corpora.

3.1 Natural Language Inference for Zero-shot Emotion Classification

The NLI task is commonly defined as a sentence-pair classification in which two sentences are given: a *premise* s_1 and a *hypothesis* s_2. The task is to learn a function $f_{\text{NLI}}(s_1, s_2) \rightarrow \{E, C, N\}$, in which E expresses the entailment of s_1 and s_2, C denotes a contradiction and N is a neutral output.

We treat ZSL emotion classification as a textual entailment problem, but represent each label under consideration with multiple prompts, in contrast to Yin et al. (2019). Given a sentence to be classified x (*premise*) and an emotion e, we have a function $g(e)$ that generates a set of prompts (*hypothesis*) out of the class $e \in E$ (with E being the set of emotions under consideration). Under the assumption of an NLI model m, which calculates the entailment probability $p_m(\gamma, x)$ for some emotion representation $\gamma \in g(e)$, we assign the average entailment probability across all emotion representations as

$$\bar{p}^g_m(e, x) = \frac{1}{|g(e)|} \sum_{\gamma \in g(e)} p_m(\gamma, x)$$

for a particular prompt generation method g. The classification decision

$$\hat{e}^g_x = \arg\max_{e \in E} \bar{p}^g_m(e, x)$$

returns the emotion corresponding to the maximum entailment probability.

3.2 Emotion Prompts

In the context of emotion analysis, two important questions arise when formulating a prompt: (*i*) How to contextualize the emotion name, and (*ii*) How to represent the emotion concept?

ID	Prompt	Example
Emo-Name	*emotion name*	*joy*
Expr-Emo	This text expresses *emotion name*	This text expresses *joy*
Feels-Emo	This person feels *emotion name*	This person feels *joyful*
WN-Def	This person expresses *WordNet def.*	This person expresses *a feeling of great pleasure and happiness*
Emo-S	*emotion synonym*	*happy*
Expr-S	This text expresses e*motion syn.*	This text expresses *happiness*
Feels-S	This person feels *emotion syn.*	This person feels *happy*
EmoLex	*emotion lexicon*	*party*

Table 1: Emotion prompts. Words in *italics* represent placeholders for the emotion concept representation.

3.2.1 Prompt Generation

We generate a set of prompts with the function $g(e) = c + r(e)$, in which c represents what we call the *context* and $r(e)$ represents a set of emotion representations.[1] As c, we use either an empty string ϵ, the text "*This text expresses*", "*This person feels*", or "*This person expresses*", motivated by our choice of the language register presented in the datasets used in our experiments (see § 4).

3.2.2 Prompts for Zero-Shot Emotion Classification

Each prompt in this paper consists of context and the emotion representation. There are three prompts which have in common the emotion name representation, namely Emo-Name, Expr-Emo, and Feels-Emo. Variations of these prompts are Emo-S, Expr-S, and Feels-S, where the emotion name representation is replaced by multiple emotion synonyms and EmoLex where the emotion name is replaced by entries from an emotion word lexicon. In detail, we use the following prompts (Table 1 shows examples):

Emo-Name. $c = \epsilon$ and $r(e) = \{e\}$.

Expr-Emo. c = "This text expresses", $r(e)$ = $\{e\}$.

Feels-Emo. c = "This person feels", $r(e)$ = $\{e\}$.

WN-Def. c = "This person expresses" and $r(e) = \{\text{WN-Def}(e)\}$, where WN-Def$(e)$ is the WordNet definition for e (Miller, 1995).

Emo-S. We aim to see whether incorporating additional information using a set of abstract emotion-related names leads to a better model. Hence, we set $r(e)$ to return a set of emotion synonyms for e. Table 3 shows the emotion synonyms considered for each emotion.[2]

Expr-S. We set $r(e)$ to be the same as in Emo-S, but additionally set c = "This text expresses". Therefore, $g(e)$ returns all combinations of this string with each synonym.

Feels-S. This prompt is the same as Expr-S with the difference that we set c = "This person feels".

EmoLex. This prompt is different from the previous ones, which consisted of small sets of context/emotion representation combinations. Here, $c = \epsilon$, but for the emotion representation we use a large popular lexicon, namely Emolex (Mohammad and Turney, 2013) to assign all entries associated with e in this lexicon. This generates prompts which contain abstract emotion synonyms as well as concrete objects (like *gift* for *joy*).

3.3 Ensemble of prompts

In practical applications, the choice of a particular prompt could not be performed manually by some user. Under the assumption that the choice of prompts is indeed corpus-specific, we combine multiple prompt sets in an ensemble.

The ensemble model takes as input a text x and a set of prompt-generating models G with $\bar{p}^g_M(e, x)$ ($g \in G$). The ensemble decision is then

$$\hat{e}(x, m) = \arg\max_{e \in E} \frac{1}{|G|} \sum_{g \in G} \bar{p}^g_m(e, x) \,.$$

4 Experiments

We aim at answering the following research questions: **(RQ1)** Do NLI models behave the same across prompts? **(RQ2)** Should we use synonyms for the emotion representation? **(RQ3)** Is an ensemble of multiple prompts more robust across corpora? **(RQ4)** Are synonyms sufficient? Would it be even more useful to use more diverse representations of emotions?

[1] In principle, c could also be a set. $g(e)$ would then need to use a cross-product instead of the element-wise concatenation $+$, which we use in our experiments.

[2] Each synonym is grammatically adapted for the context of the prompts Expr-S and Feels-S.

Dataset	Labels	Size	Source	Avail.
TEC	Ekman	21,051	tweets	D-RO
BLOGS	Ekman + *no emotion*	5,205	blogs	R
ISEAR	Ekman − *Su* + *G* + *Sh*	15,302	events	GPLv3

Table 2: Datasets used in our experiments (Su: surprise, G: guilt, Sh: shame) [D-RO] available to download, research only, [R] Available upon request, [GPLv3] GNU Public License version 3.

4.1 Experimental Setting

4.1.1 Datasets

We compare our methods on three English corpora, to gain an understanding of the role of the respective corpus. TEC (Mohammad, 2012) contains 21,051 tweets weakly labeled according to hashtags corresponding to the six Ekman emotions (Ekman, 1992): *#anger*, *#disgust*, *#fear*, *#happy*, *#sadness*, and *#surprise*. ISEAR (Scherer and Wallbott, 1997) includes 7,665 English self-reports of events that triggered one of the emotions (*joy*, *fear*, *anger*, *sadness*, *disgust*, *shame*, and *guilt*). BLOGS (Aman and Szpakowicz, 2007) consists of 5,205 sentences from 173 blogs compiled from the Web using a list of emotion-related seed words. It is human-annotated according to Ekman's set of basic emotions and an additional *no emotion* category. TEC and ISEAR are publicly available for research purposes and BLOGS is available upon request. All datasets are anonymized by the authors.

These corpora differ in various parameters (see Table 2): the annotation scheme (variations of Ekman's model), the corpus source (tweets, events, blogs), the annotation procedure (hashtag, crowdsourcing, self-reporting), and the size. Note that the annotation procedure that the ZSL method needs to reconstruct varies in complexity.

4.1.2 NLI Models and Baseline

We compare our ZSL models with an empirical upper bound, namely a RoBERTa model fine-tuned with supervised training (Liu et al., 2020) on each emotion dataset described in § 3.2.2. We fine-tune RoBERTa for three epochs, the batch size is set to 32 and the learning rate to $2 \cdot 10^{-5}$. No hyperparameter search has been applied. We perform 10-fold cross-validation and report the results on the whole data set (as we do with the NLI models).

For our ZSL experiments, we explore three state-of-the-art pretrained NLI models publicly available within the Hugging Face Transformers Python library (Wolf et al., 2020), and fine-tuned on the MultiNLI dataset (Williams et al., 2018). Concretely, we choose RoBERTa, BART and DeBERTa as they cover different architectures and represent competitive approaches across a set of NLP tasks.

Emotion	**Emo-S**
anger	anger, annoyance, rage, outrage, fury, irritation
fear	fear, horror, anxiety, terror, dread, scare
joy	joy, achievement, pleasure, awesome, happy, blessed
sadness	sadness, unhappy, grief, sorrow, loneliness, depression
disgust	disgust, loathing, bitter, ugly, repugnance, revulsion
surprise	surprise, astonishment, amazement, impression, perplexity, shock
guilt	guilt, culpability, blameworthy, responsibility, misconduct, regret
shame	shame, humiliate, embarrassment, disgrace, dishonor, discredit

Table 3: Emotion synonyms per emotion category considered in Emo-S prompt (details in the Appendix).

RoBERTa. The Robustly Optimized BERT Pretraining Approach (Liu et al., 2020) is a modified version of BERT which includes some changes such as the removal of the next-sentence prediction task, the replacement of the WordPiece tokenization with a variation of the byte-pair encoding, and the replacement of the static masking (the same input masks are fed to the model on each epoch) with dynamic masking (the masking is generated every time the sequence is fed to the model). For the NLI task, we use the *roberta-large-mnli* model from Hugging Face which contains over 355M of parameters.

BART. The Bidirectional and Auto-Regressive Transformer (Lewis et al., 2020) is a model that combines the bidirectional encoder with an autoregressive decoder into one sequence-to-sequence model. We use the *facebook/bart-large-mnli* model from Hugging Face with over 407M parameters.

DeBERTa. The Decoding-enhanced BERT with Disentangled Attention model (He et al., 2021) improves BERT and RoBERTa using two techniques, namely disentangled attention and an enhanced mask decoder. We use *microsoft/deberta-xlarge-mnli* from Hugging Face, which contains over 750M of parameters.

All experiments are performed on a node equipped

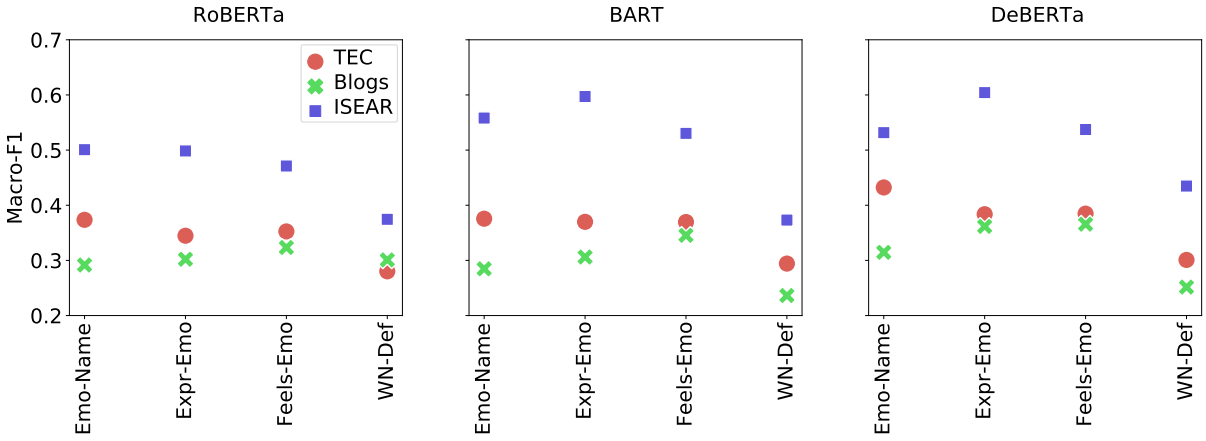

Figure 2: Results of Experiment 1. Comparison of prompts across NLI models and emotion datasets.

with two Intel Xeon Silver 4208 CPU at 2.10GHz, 192GB RAM, as main processors, and six GPUs NVIDIA GeForce RTX 2080Ti (with 11GB each).

4.2 Results

In order to answer the research questions formulated in this study, we conduct different ZSL-NLI emotion classification experiments.

4.2.1 Experiment 1: Are NLI models behaving the same across prompts?

With the first experiment, we aim at observing if different NLI models behave robustly across emotion datasets and prompts. We use each model described in § 4.1.2 with each emotion representation that is not a set of multiple prompts, but only consists of a single prompt, namely Emo-Name, Expr-Emo, Feels-Emo and WN-Def. We evaluate each model using all datasets (§ 4.1.1).

Figure 2 (and Table 6 in the Appendix) show the results. Each plot shows the performance of one NLI model on the three emotion datasets using the four prompts. We see that the performances follow the same patterns across NLI models and emotion datasets. Emo-Name is the best performing prompt for TEC, Expr-Emo for ISEAR and Feels-Emo for BLOGS. The lowest performance is achieved with WN-Def. The most successful NLI model across the prompts is DeBERTa followed by BART and RoBERTa.

Therefore, NLI models do behave robustly across prompts. Particularly low performance can be observed with WN-Def. This finding is in line with previous research (Yin et al., 2019): These definitions may be suboptimal choices, for instance, *sadness* is represented via "This person expresses emotions experienced when not in a state of well-being". This is ambiguous since not being in a state of well-being may also be associated with other negative emotions such as *anger* or *fear*. Interestingly, the best-performing emotion representation on TEC is Emo-Name, which resembles the annotation procedure of just using an emotion-related hashtag for labeling. Similarly, Expr-Emo shows the best performance for the self-reports of ISEAR ("This text expresses") and Feels-Emo on BLOGS ("This person feels"). These subtle differences in the prompt formulations indicate that there are particular factors in the dataset that influence the interpretation of the prompt, for instance, the annotation procedure, the data selection or the language register employed in the corpus, and therefore, they affect the interpretation of the emotion by the NLI-ZSL classifier.

4.2.2 Experiment 2: Should we use synonyms for emotion representation?

In this experiment, we aim at observing whether the incorporation of synonyms in the prompt helps the emotion interpretation. Instead of considering only the emotion name, we use six close emotion synonyms (see Emo-S, Expr-S, Feels-S in Table 7 in the Appendix).[3] This leads to six prompts for each emotion. For simplicity, we now only consider DeBERTa, which showed best performances in the previous experiment.

Figure 3 (and Table 6 in the Appendix) shows the results of each context with just the emotion

[3]We assume that larger numbers might show better performance in general, but this set of six synonyms focuses on close, unambiguous synonyms which undoubtedly represent the emotion in most contexts. We evaluate the impact of larger sets with the EmoLex approach.

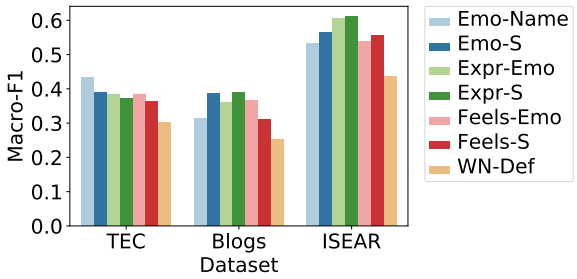

Figure 3: Results of Experiment 2. Comparison of prompts including synonym emotion representations across three emotion datasets (TEC, BLOGS and ISEAR) using the DeBERTa model.

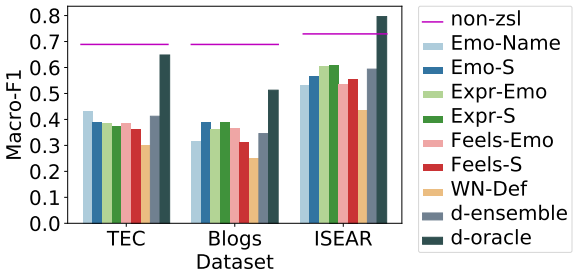

Figure 4: Results of Experiment 3. Comparison of the prompt individual models and the proposed ensemble models along with the non-zsl experiments.

name and with the synonyms in comparison. In general, synonym use leads to an improvement, with some notable exceptions. For TEC, the single use of the emotion (Emo-Name) works better than using synonyms (Emo-S). This might stem from a similarity of the prompt with the annotation procedure, in which single hashtags were used for labeling. Another exception is Feels-Emo/Feels-S in BLOGS. Therefore, to answer RQ2 we conclude that both context and emotion concept representation are corpus-dependent and in some cases synonyms support the emotion classification.

4.2.3 Experiment 3: Is an ensemble of multiple prompts more robust across corpora?

The previous experiments demonstrate the challenge of engineering an emotion prompt that fits different corpora which stem from various sources. To cope with this challenge, we analyze if the combination of sets of prompt-generation methods in an ensemble improves the generalizability. We use the ensemble method described in § 3.3 that combines the predictions given by the set of model prompts described in § 3.2.2 with the DeBERTa model (d-ensemble). In addition to this realistic ensemble model, we want to understand which performance could be achieved with an ideal (oracle) ensemble (which we refer to as d-oracle), which always picks the correct decision by an ensemble component, if one is available. This serves as an upper bound and analyzes the complementarity of the individual models.

Figure 4 shows the performance for the individual models discussed before, which participate in both the realistic and the oracle ensemble (individual results in Table 6 in the Appendix, ensemble results also in Table 5). In addition, we see both ensemble methods and (as a horizontal line) the supervised learning upper bound. We observe that the realistic ensemble (d-ensemble), which is based on averaging the individual probabilistic outputs of the individual models, shows a performance nearly en par with the individual best model: For TEC, we have an F_1 =.41 in comparison to the individual best F_1 =.43, for BLOGS, we have F_1 =.35 in comparison to F_1 =.39, and for ISEAR, we achieve F_1 =.59 in comparison to F_1 =.61 – but without the necessity to pick the prompt-generating approach beforehand or on some hold-out data.

We further see that the oracle ensemble performs better than all other models – this shows the variance between the models and suggests a reason for their corpus-dependency, but also shows the potential for other ensemble models. This oracle also approaches (or is even slightly higher than) the supervised upper-bound. All of our current (non-oracle) ZSL learning methods clearly underperform supervised learning, but to various degrees. The oracle performance suggests that sets of prompts, combined with a good ensembling method, might exist that outperform supervised learning in emotion classification.

We conclude that an ensemble model is indeed more robust across emotion datasets with different language registers and prompts, with a performance nearly en par with the best corpus-specific prompt. This raises the question what differences and commonalities instances have in which models perform the same or differently. To this end, we show examples in Table 4, in which *all* individual models did output the correct label. As we can see, these instances contain explicit words related to the emotion conveyed. For instance, "lost" for *sadness*, "love" for *joy*, "angry" for *anger*, "nervous" for *fear*, "ashamed" for *shame*, and "felt bad" for *guilt*. Therefore, prompt-NLI models succeed

Emotion	Text
anger	The sports fishermen who catch gulls instead of fish with their hooks. It is often a mistake but it makes me angry. (ISEAR)
disgust	my sister got this purse, It smell like straight up KITTY LITTER. (TEC)
fear	Oh well its nothing too too bad but its making me nervous. (BLOGS)
guilt	While at primary school, I did not let a friend ring a bell although he would have liked to do it. Afterwards I felt bad. (ISEAR)
joy	When I get a hug from someone I love. (ISEAR)
sadness	When I lost the person who meant the most to me. (ISEAR)
surprise	Snow in October! (BLOGS)
shame	We got into a fight with some chaps in front of our family house. The value of the property destroyed was approximately 15 000 FIM. I felt ashamed when my parents came to know about this. (ISEAR)

Table 4: Instances where all the prompt models agree with the emotion prediction.

in interpreting emotions that are clearly expressed in the text, but vary performance-wise when the emotion is implicitly communicated.

4.2.4 Experiment 4: Are synonyms sufficient? Would it be even more useful to use more representations of emotions?

In Experiment 2 we found that the use of synonyms is beneficial in some cases (ISEAR and BLOGS). This raises the question if more terms that represent the emotion would lead to an even better performance. We evaluate this setup with the EmoLex model introduced above, in which each emotion concept is represented with a set of prompts, where each prompt is a concept from an emotion lexicon. Notably, in this prompt-generating methods, emotions are not only represented by abstract emotion names or synonyms, but in addition with (sometimes concrete) concepts, like "gift" or "tears".

Table 5 shows the performance of the DeBERTa model using the Emolex concepts (d-emolex), next to the ensemble results. The additional concepts which cover a wide range of topics associated with the respective emotions particularly help in the BLOGS corpus, which is the one resource that has been manually annotated in a traditional manner. This manual annotation process might include complex inference by the annotators to infer an emotion category, instead of only using single words

	TEC			BLOGS			ISEAR		
Model	P	R	F_1	P	R	F_1	P	R	F_1
d-ensemble	.42	.44	.41	.40	.65	.35	.67	.62	.59
d-oracle	.63	.69	.65	.51	.80	.51	.82	.80	.80
d-emolex	.37	.36	.33	.52	.48	.48	.47	.42	.40
non-zsl	.69	.69	.69	.72	.71	.69	.73	.73	.73

Table 5: Results of Experiments 3 and 4. We report macro-average precision (P), macro-average recall (R), and macro-average F_1 (F_1) for each model. d-emolex: DeBERTa using EmoLex prompt, d-ensemble: ensemble model of prompts using DeBERTa, d-oracle: oracle ensemble model using DeBERTa), non-zsl: Supervised RoBERTa model fine-tune on the three emotion datasets.

to trigger an event description (ISEAR) or using words as hashtags (TEC). Lexicons can therefore aid in the injection of background knowledge in the prompt. However, this comes at the cost of considerably longer runtimes, because the NLI models is queried for every entry in the lexicon.

5 Conclusion and Future Work

We presented an analysis of various prompts for NLI-based ZSL emotion classification. The prompts that we chose were motivated by the various particularities of the corpora: single emotions for TEC (tweets), "The person feels/The text expresses" for BLOGS (blogs), and ISEAR (events). In addition, we represented the emotions with emotion names, synonyms, definitions, or with the help of lexicons. Our experiments across these data sets showed that, to obtain a superior performance, the prompt needs to fit well to the corpus – we did not find one single prompt that works well across different corpora. To avoid the requirement for manually selecting a prompt, we therefore devised an ensemble model that combines multiple sets of prompts. This model is more robust and is nearly on par with the best individual prompt. In addition, we found that representing the emotion concept more diversely with synonyms or lexicons is beneficial, but again corpus-specific.

Our work raises a set of future research questions. We have seen that the oracle ensemble showed a good performance, illustrating that the various prompts provide complementary information. This motivates future research regarding other combination schemes, including learning a combination based on end-to-end fine-tuned NLI models.

We have further seen that including more concepts with the help of a dictionary helps in one

corpus, but not across corpora; however, synonyms constantly help. This raises the question about the right trade-off between many, but potentially inappropriate, noisy concepts and hand-selected, high-quality concepts. A desideratum is an automatic subselection procedure, which removes concepts that might decrease performance and only keeps concepts that are "compatible" to the current language register and annotation method. Ideally, this procedure would not make use of annotated data, because that would limit the advantages of ZSL.

The main limitation of our current work is that we manually designed the prompts under consideration, based on the corpora we used for evaluation. This is a bottleneck in model development, which should either be supported by a more guided approach which supports humans in developing prompts, or by an automatic model that is able to automatically generate prompts based on the language register and concept representation in the dataset.

Acknowledgements

We thank Enrica Troiano and Laura Oberländer for discussions on the topic of emotion analysis.

Roman Klinger's work is supported by the German Research Council (DFG, project number KL 2869/1-2). Flor Miriam Plaza-del-Arco and María-Teresa Martín Valdivia have been partially supported by the LIVING-LANG project (RTI2018-094653-B-C21) funded by MCIN/AEI/10.13039/501100011033 and ERDF A way of making Europe, and a grant from the Ministry of Science, Innovation and Universities of the Spanish Government (FPI-PRE2019-089310).

References

Saima Aman and Stan Szpakowicz. 2007. Identifying Expressions of Emotion in Text. In *Text, Speech and Dialogue, 10th International Conference, TSD 2007, Pilsen, Czech Republic, September 3-7, 2007, Proceedings*, volume 4629 of *Lecture Notes in Computer Science*, pages 196–205. Springer.

Angelo Basile, Guillermo Pérez-Torró, and Marc Franco-Salvador. 2021. "Probabilistic Ensembles of Zero- and Few-Shot Learning Models for Emotion Classification". In *Proceedings of the International Conference on Recent Advances in Natural Language Processing (RANLP 2021)*, pages 128–137, Held Online. INCOMA Ltd.

Laura Ana Maria Bostan, Evgeny Kim, and Roman Klinger. 2020. "GoodNewsEveryone: A Corpus of News Headlines Annotated with Emotions, Semantic Roles, and Reader Perception". In *Proceedings of the 12th Language Resources and Evaluation Conference*, pages 1554–1566, Marseille, France. European Language Resources Association.

Laura-Ana-Maria Bostan and Roman Klinger. 2018. "An Analysis of Annotated Corpora for Emotion Classification in Text". In *Proceedings of the 27th International Conference on Computational Linguistics*, pages 2104–2119, Santa Fe, New Mexico, USA. Association for Computational Linguistics.

Tom B. Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel M. Ziegler, Jeffrey Wu, Clemens Winter, Christopher Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. 2020. Language Models are Few-Shot Learners. In *Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020, December 6-12, 2020, virtual*.

Sven Buechel and Udo Hahn. 2017. "EmoBank: Studying the Impact of Annotation Perspective and Representation Format on Dimensional Emotion Analysis". In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 2, Short Papers*, pages 578–585, Valencia, Spain. Association for Computational Linguistics.

Sven Buechel, Luise Modersohn, and Udo Hahn. 2021. "Towards Label-Agnostic Emotion Embeddings". In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 9231–9249, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. "BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding". In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, Minneapolis, Minnesota. Association for Computational Linguistics.

Sidney D'mello and Arthur Graesser. 2007. Mind and body: Dialogue and posture for affect detection in learning environments. In *Proceedings of the 2007 conference on Artificial Intelligence in Education: Building Technology Rich Learning Contexts That Work*, page 161–168, NLD. IOS Press.

Bao Minh Doan Dang, Laura Oberländer, and Roman Klinger. 2021. Emotion Stimulus Detection in German News Headlines. In *Proceedings of the 17th*

Conference on Natural Language Processing (KONVENS 2021), Düsseldorf, Germany. German Society for Computational Linguistics & Language Technology.

Paul Ekman. 1992. An argument for basic emotions. *Cognition and Emotion*, 6(3-4):169–200.

Thomas Haider, Steffen Eger, Evgeny Kim, Roman Klinger, and Winfried Menninghaus. 2020. "PO-EMO: Conceptualization, Annotation, and Modeling of Aesthetic Emotions in German and English Poetry". In *Proceedings of the 12th Language Resources and Evaluation Conference*, pages 1652–1663, Marseille, France. European Language Resources Association.

Kishaloy Halder, Alan Akbik, Josip Krapac, and Roland Vollgraf. 2020. "Task-Aware Representation of Sentences for Generic Text Classification". In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 3202–3213, Barcelona, Spain (Online). International Committee on Computational Linguistics.

Pengcheng He, Xiaodong Liu, Jianfeng Gao, and Weizhu Chen. 2021. DeBERTa: Decoding-enhanced BERT with Disentangled Attention. In *International Conference on Learning Representations*.

Jan Hofmann, Enrica Troiano, Kai Sassenberg, and Roman Klinger. 2020. "Appraisal Theories for Emotion Classification in Text". In *Proceedings of the 28th International Conference on Computational Linguistics*, Barcelona, Spain (Online). International Committee on Computational Linguistics.

Maximilian Köper, Evgeny Kim, and Roman Klinger. 2017. "IMS at EmoInt-2017: Emotion Intensity Prediction with Affective Norms, Automatically Extended Resources and Deep Learning". In *Proceedings of the 8th Workshop on Computational Approaches to Subjectivity, Sentiment and Social Media Analysis*, pages 50–57, Copenhagen, Denmark. Association for Computational Linguistics.

Angeliki Lazaridou, Georgiana Dinu, and Marco Baroni. 2015. "Hubness and Pollution: Delving into Cross-Space Mapping for Zero-Shot Learning". In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 270–280, Beijing, China. Association for Computational Linguistics.

Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Veselin Stoyanov, and Luke Zettlemoyer. 2020. BART: Denoising Sequence-to-Sequence Pre-training for Natural Language Generation, Translation, and Comprehension. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7871–7880, Online. Association for Computational Linguistics.

Xiangju Li, Wei Gao, Shi Feng, Yifei Zhang, and Daling Wang. 2021. "Boundary Detection with BERT for Span-level Emotion Cause Analysis". In *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*, pages 676–682, Online. Association for Computational Linguistics.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2020. RoBERTa: A Robustly Optimized BERT Pretraining Approach.

Winfried Menninghaus, Valentin Wagner, Eugen Wassiliwizky, Ines Schindler, Julian Hanich, Thomas Jacobsen, and Stefan Koelsch. 2019. What are aesthetic emotions? *Psychological Review*, 126(2):171–195.

George A. Miller. 1995. Wordnet: A lexical database for english. *Commun. ACM*, 38(11):39–41.

Saif Mohammad. 2012. # Emotional tweets. In ** SEM 2012: The First Joint Conference on Lexical and Computational Semantics–Volume 1: Proceedings of the main conference and the shared task, and Volume 2: Proceedings of the Sixth International Workshop on Semantic Evaluation (SemEval 2012)*, pages 246–255.

Saif Mohammad and Felipe Bravo-Marquez. 2017. Emotion Intensities in Tweets. In *Proceedings of the 6th Joint Conference on Lexical and Computational Semantics (*SEM 2017)*, pages 65–77, Vancouver, Canada. Association for Computational Linguistics.

Saif Mohammad, Felipe Bravo-Marquez, Mohammad Salameh, and Svetlana Kiritchenko. 2018a. SemEval-2018 task 1: Affect in tweets. In *Proceedings of The 12th International Workshop on Semantic Evaluation*, pages 1–17, New Orleans, Louisiana. Association for Computational Linguistics.

Saif Mohammad, Felipe Bravo-Marquez, Mohammad Salameh, and Svetlana Kiritchenko. 2018b. SemEval-2018 task 1: Affect in tweets. In *Proceedings of The 12th International Workshop on Semantic Evaluation*, pages 1–17, New Orleans, Louisiana. Association for Computational Linguistics.

Saif M Mohammad and Peter D Turney. 2013. Crowdsourcing a word–emotion association lexicon. *Computational intelligence*, 29(3):436–465.

Sungjoon Park, Jiseon Kim, Seonghyeon Ye, Jaeyeol Jeon, Hee Young Park, and Alice Oh. 2021. Dimensional Emotion Detection from Categorical Emotion. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 4367–4380, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Flor Miriam Plaza-del Arco, Salud M. Jiménez Zafra, Arturo Montejo Ráez, M. Dolores Molina González, Luis Alfonso Ureña López, and María Teresa

Martín Valdivia. 2021. Overview of the EmoEvalEs task on emotion detection for Spanish at IberLEF 2021. *Procesamiento del Lenguaje Natural*, 67(3):273–294.

Flor Miriam Plaza-del-Arco, Carlo Strapparava, L. Alfonso Ureña-López, and M. Teresa Martín-Valdivia. 2020. EmoEvent: A Multilingual Emotion Corpus based on different Events. In *Proceedings of the 12th Language Resources and Evaluation Conference*, pages 1492–1498, Marseille, France. European Language Resources Association.

Robert Plutchik. 2001. The Nature of Emotions: Human emotions have deep evolutionary roots, a fact that may explain their complexity and provide tools for clinical practice. *American scientist*, 89(4):344–350.

Daniel Preoţiuc-Pietro, H. Andrew Schwartz, Gregory Park, Johannes Eichstaedt, Margaret Kern, Lyle Ungar, and Elisabeth Shulman. 2016. "Modelling Valence and Arousal in Facebook posts". In *Proceedings of the 7th Workshop on Computational Approaches to Subjectivity, Sentiment and Social Media Analysis*, pages 9–15, San Diego, California. Association for Computational Linguistics.

Milos Radovanovic;, Alexandros Nanopoulos, and Mirjana Ivanovic. 2010. Hubs in Space: Popular Nearest Neighbors in High-Dimensional Data. *Journal of Machine Learning Research*, 11(86):2487–2531.

Anthony Rios and Ramakanth Kavuluru. 2018. "Few-Shot and Zero-Shot Multi-Label Learning for Structured Label Spaces". In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 3132–3142, Brussels, Belgium. Association for Computational Linguistics.

James A Russell and Albert Mehrabian. 1977. Evidence for a three-factor theory of emotions. *Journal of research in Personality*, 11(3):273–294.

Prateek Veeranna Sappadla, Jinseok Nam, Eneldo Loza Mencía, and Johannes Fürnkranz. 2016. Using Semantic Similarity for Multi-Label Zero-Shot Classification of Text Documents. In *Proceedings of the 23rd European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning (ESANN-16)*, Bruges, Belgium. d-side publications.

Klaus R. Scherer and Harald G. Wallbott. 1997. The ISEAR Questionnaire and Codebook. Geneva Emotion Research Group.

Craig A. Smith and Phoebe C. Ellsworth. 1985. Patterns of cognitive appraisal in emotion. *Journal of personality and social psychology*, 48(4):813.

Richard Socher, Milind Ganjoo, Christopher D. Manning, and Andrew Y. Ng. 2013. Zero-Shot Learning through Cross-Modal Transfer. In *Proceedings of the 26th International Conference on Neural Information Processing Systems - Volume 1*, NIPS'13, page 935–943, Red Hook, NY, USA. Curran Associates Inc.

P.S. Sreeja and G.S. Mahalaksmi. 2015. Applying vector space model for poetic emotion recognition. *Advances in Natural and Applied Sciences*, 9(6 SE):486–491.

Marco Antonio Stranisci, Simona Frenda, Eleonora Ceccaldi, Valerio Basile, Rossana Damiano, and Viviana Patti. 2022. Appreddit: a corpus of reddit posts annotated for appraisal. In *Proceedings of the Language Resources and Evaluation Conference*, pages 3809–3818, Marseille, France. European Language Resources Association.

Shabnam Tafreshi, Orphee De Clercq, Valentin Barriere, Sven Buechel, João Sedoc, and Alexandra Balahur. 2021. "WASSA 2021 Shared Task: Predicting Empathy and Emotion in Reaction to News Stories". In *Proceedings of the Eleventh Workshop on Computational Approaches to Subjectivity, Sentiment and Social Media Analysis*, pages 92–104, Online. Association for Computational Linguistics.

Enrica Troiano, Laura Oberländer, and Roman Klinger. 2023. Dimensional modeling of emotions in text with appraisal theories: Corpus creation, annotation reliability, and prediction. *Computational Linguistics*, 49(1).

Enrica Troiano, Sebastian Padó, and Roman Klinger. 2019. "Crowdsourcing and Validating Event-focused Emotion Corpora for German and English". In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, Florence, Italy. Association for Computational Linguistics.

Wei Wang, Vincent W. Zheng, Han Yu, and Chunyan Miao. 2019. A Survey of Zero-Shot Learning: Settings, Methods, and Applications. *ACM Trans. Intell. Syst. Technol.*, 10(2).

Adina Williams, Nikita Nangia, and Samuel Bowman. 2018. "A Broad-Coverage Challenge Corpus for Sentence Understanding through Inference". In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 1112–1122, New Orleans, Louisiana. Association for Computational Linguistics.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Remi Louf, Morgan Funtowicz, Joe Davison, Sam Shleifer, Patrick von Platen, Clara Ma, Yacine Jernite, Julien Plu, Canwen Xu, Teven Le Scao, Sylvain Gugger, Mariama Drame, Quentin Lhoest, and Alexander Rush. 2020. "Transformers: State-of-the-Art Natural Language Processing". In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, Online. Association for Computational Linguistics.

Yongqin Xian, Christoph H. Lampert, Bernt Schiele, and Zeynep Akata. 2019. Zero-Shot Learning—A

Comprehensive Evaluation of the Good, the Bad and the Ugly. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 41(9):2251–2265.

Wenpeng Yin, Jamaal Hay, and Dan Roth. 2019. "Benchmarking Zero-shot Text Classification: Datasets, Evaluation and Entailment Approach". In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 3914–3923, Hong Kong, China. Association for Computational Linguistics.

A Experiment Results

Dataset	Model	Emo-Name			Expr-Emo			Feels-Emo			WN-Def		
		P	R	F_1	P	R	F_1	P	R	F_1	P	R	F_1
TEC	r	.39	.42	.37	.36	.38	.34	.39	.40	.35	.37	.31	.28
	b	.39	.42	.38	.38	.42	.37	.40	.41	.37	.32	.32	.29
	d	.42	.47	.43	.41	.42	.38	.42	.42	.38	.44	.33	.30
	d-synonyms	.42	.42	.39	.39	.40	.37	.39	.40	.36	—	—	—
BLOGS	r	.32	.62	.29	.36	.60	.30	.41	.59	.32	.47	.47	.30
	b	.33	.58	.28	.35	.62	.31	.47	.56	.35	.35	.40	.24
	d	.35	.64	.31	.41	.65	.36	.49	.58	.37	.38	.48	.25
	d-synonyms	.41	.62	.39	.42	.63	.39	.36	.60	.31	—	—	—
ISEAR	r	.58	.50	.50	.53	.50	.50	.55	.47	.47	.50	.37	.37
	b	.62	.56	.56	.64	.60	.60	.68	.53	.53	.57	.40	.37
	d	.63	.56	.53	.66	.62	.60	.68	.54	.54	.54	.45	.43
	d-synonyms	.64	.57	.57	.64	.62	.61	.63	.58	.55	—	—	—

Table 6: Results from the set of prompts across emotion datasets (TEC, BLOGS and ISEAR) and NLI models. We report macro-average precision (P), macro-average recall (R), and macro-average F_1 (F_1) for each model. (r: RoBERTa, b: BART, d: DeBERTa, d-synonyms: DeBERTa using as prompts synonyms. In cases where no experiments have been performed, we use '—'. Figures 2 and 3 in the paper depict these experiments.

B List of Emotion Representations as Part of Prompts

Emotion	**Emo-S**	**Expr-S**	**Feels-S**	**WN-Def**
Context	ϵ	"This text expresses…"	"This person feels…"	"This person expresses…"
anger	anger, annoyance, rage, outrage, fury, irritation	anger, annoyance, rage, outrage, fury, irritation	anger, annoyed, rage, outraged, furious, irritated	a strong feeling of annoyance, displeasure, or hostility
fear	fear, horror, anxiety, terror, dread, scare	fear, horror, anxiety, terror, dread, scare	fear, horror, anxiety, terrified, dread, scared	an unpleasant emotion caused by the belief that someone or something is dangerous, likely to cause pain , or a threat
joy	joy, achievement, pleasure, awesome, happy, blessed	joy, an achievement, pleasure, the awesome, happiness, the blessing	joyful, accomplished, pleasure, awesome, happy, blessed	a feeling of great pleasure and happiness
sadness	sadness, unhappy, grief, sorrow, loneliness, depression	sadness, unhappiness, grief, sorrow, loneliness, depression	sadness, unhappy, grieved, sorrow, lonely, depression	emotions experienced when not in a state of well-being
disgust	disgust, loathing, bitter, ugly, repugnance, revulsion	disgust, loathing, bitterness, ugliness, repugnance, revulsion	disgusted, loathing, bitter, ugly, repugnance, revulsion	a feeling of revulsion or strong disapproval aroused by something unpleasant or offensive
surprise	surprise, astonishment, amazement, impression, perplexity, shock	surprise, astonishment, amazement, impression, perplexity, shock	surprised, astonishment, amazement, impressed, perplexed, shocked	a feeling of mild astonishment or shock caused by something unexpected
guilt	guilt, culpability, blameworthy, responsibility, misconduct, regret	guilt, culpability, responsibility, blameworthy, misconduct, regret	guilty, culpable, responsible, blame, misconduct, regretful	a feeling of having done wrong or failed in an obligation
shame	shame, humiliate, embarrassment, disgrace, dishonor, discredit	shame, humiliation, embarrassment, disgrace, dishonor, discredit	shameful, humiliated, embarrassed, disgraced, dishonored, discredit	a painful feeling of humiliation or distress caused by the consciousness of wrong or foolish behavior

Table 7: Emotion representation in prompts Emo-S, Expr-S, Feels-S, and WN-Def.

COMMUNITYLM: Probing Partisan Worldviews from Language Models

Hang Jiang, Doug Beeferman, Brandon Roy, Deb Roy
MIT Center for Constructive Communication
MIT Media Lab
{hjian42,dougb5,bcroy,dkroy}@media.mit.edu

Abstract

As political attitudes have diverged ideologically in the United States, political speech has diverged linguistically. The ever-widening polarization between the US political parties is accelerated by an erosion of mutual understanding between them. We aim to make these communities more comprehensible to each other with a framework that probes community-specific responses to the same survey questions using community language models (COMMUNITYLM). In our framework we identify committed partisan members for each community on Twitter and fine-tune LMs on the tweets authored by them. We then assess the worldviews of the two groups using prompt-based probing of their corresponding LMs, with prompts that elicit opinions about public figures and groups surveyed by the American National Election Studies (ANES) 2020 Exploratory Testing Survey. We compare the responses generated by the LMs to the ANES survey results, and find a level of alignment that greatly exceeds several baseline methods. Our work aims to show that we can use community LMs to query the worldview of any group of people given a sufficiently large sample of their social media discussions or media diet.

1 Introduction

Political polarization is a prominent component of politics in the United States (Poole and Rosenthal, 1984; McCarty et al., 2016; Heltzel and Laurin, 2020). Previous studies have shown growing polarization in social media (Bail et al., 2018; Demszky et al., 2019; Darwish, 2019) and substantial partisan and ideological differences in media diet (Bozell, 2004; Gil de Zúñiga et al., 2012; Hyun and Moon, 2016). Li et al. (2017) show that partisanship makes reliable predictions about an individual's word understanding. R. KhudaBukhsh et al. (2021) used modern machine-translation techniques to demonstrate that the left and right communities use English words differently. Milbauer et al. (2021) extended the method to uncover worldview and ideological differences between 32 Reddit communities. These studies are word-level analyses based on Word2vec word embeddings, and none of them use pre-trained language models.

Prompt	Model	Top 5 Words
Dr. Fauci is a	Republican GPT-2	liar (2.96%), joke (2.67%), hero (2.13%), doctor (1.62%), great (1.61%)
	Democratic GPT-2	hero (10.36%), true (3.63%), national (2.08%), physician (2.06%), great (1.93%)

Table 1: Top 5 words by odds for Republican and Democratic GPT-2 models, fine-tuned on partisan tweets. Dr. Fauci is suggested to be a "hero" by the GPT-2 model fine-tuned on Democratic tweets but a "liar" and "joke" by the GPT-2 model fine-tuned on Republican tweets.

Prompting is a standard technique to make pre-trained language models generate texts conditioned on prompts. Recent work has shown that, through prompt engineering, pre-trained language models can achieve good zero-shot performance on NLP tasks from sentiment classification to reading comprehension (Radford et al., 2019; Brown et al., 2020) and mine factual or commonsense knowledge (Petroni et al., 2019; Davison et al., 2019; Jiang et al., 2021; Talmor et al., 2020). Through prompting, Palakodety et al. (2020) used a fine-tuned BERT (Devlin et al., 2019) model with fill-in-the-blank cloze statements to mine insights and compare prediction differences between Indian regional and national YouTube news channels. Feldman et al. (2021) fine-tuned GPT-2 on COVID-19 tweet corpora to mine user opinions.

However, none of these studies fine-tune GPT-style language models on community data to probe community worldviews. In this work, we focus on Republican and Democratic Twitter communities and conduct a feasibility study using fine-tuned GPT-2 partisan language models to generate community responses and to predict community stance.

Proceedings of the 29th International Conference on Computational Linguistics, pages 6818–6826
October 12–17, 2022.

As exemplified in Table 1, we observe clear partisan differences. In this sociopolitically fragmented society, our motivation is to provide a simple and flexible interface for people to probe each other's worldviews on topics of interest and to encourage constructive dialogue. We demonstrate through our experiments and analyses that the proposed method is a reliable tool to probe community opinions. The contribution of the work is as follows:

- We present a simple COMMUNITYLM framework based on GPT-2 language models to mine community insights by fine-tuning or training the model on community data. This study focuses on Democrat and Republican communities on Twitter but can be easily extended to probe insights from any community based on their public discourse or media diet[1].
- We use ANES questions as prompts and find that GPT-generated opinions are predictive of community stance towards public figures and groups. We experiment with 4 types of prompts and find that the fine-tuned COMMUNITYLM with an "X is the" prompt outperforms all the baselines (including pre-trained GPT-3 Curie) in predicting community stance.
- We analyze the errors made by community language models and demonstrate the capability of the models to probe community preferences towards public figures by ranking.

2 Partisan Twitter Data

We construct a Twitter dataset containing 4.7M tweets (100M word tokens) by Republican and Democrat communities respectively. We first sampled 1M active U.S. Twitter users before and after the 2020 presidential election. We adapt the standard method (Volkova et al., 2014; Demszky et al., 2019) to estimate their political affiliation from the political accounts they follow and collect tweets of Republican and Democratic Twitter users from 2019-01-01 to 2020-04-10. We pick this period because the ANES 2020 survey was collected between 2020-04-10 to 2020-04-18 and we want to ensure the Twitter training data does not leak information beyond 2020-04-10. We subsample 4.7M tweets from each side to achieve a balanced set and use Nguyen et al. (2020)'s tweet tokenizer for data processing. Details are described as follows.

U.S. Twitter User Sampling. We first sample a subset of active Twitter users from the "decahose", Twitter's 10% sample of tweets. We define active U.S. users as those who posted at least 10 original tweets before and after the 2020 presidential election period (2020-07-01 to 2021-06-31). We then use Litecoder[2] to extract user locations from their profile location strings and filter out users not based in the U.S. We construct the follow graph of the resulting set of 1,074,650 Twitter users.

Partisan Assignment. We follow previous studies (Volkova et al., 2014; Demszky et al., 2019) to estimate the party affiliation of Twitter users from the political accounts they follow. Specifically, we update the list of Twitter handles of US politicians from Demszky et al. (2019) by adding current federal officeholders and governors from Ballotpedia[3]. The final list has 457 Republican and 473 Democratic politician Twitter handles. To identify committed partisan users, we adopt the following rules: a user is labeled as a Democrat if they followed no fewer than 6 Democratic politicians and no Republican politician from the list in February 2022, whereas a person is labeled as a Republican if they followed no fewer than 2 Republican politicians and no Democratic politicians. We choose these thresholds because there are 69% Democratic users and 26% Republican users on Twitter[4] (2.65:1). This step predicts 182,788 Democratic-leaning and 72,186 Republican-leaning users (2.53:1).

Tweet Pre-processing. We use the tweet tokenizer from Nguyen et al. (2020) to process all the data. This tokenizer converts user mentions and web/url links into special tokens @USER and HTTPURL. We delete HTTPURL from the tweets because it does not contain useful community information. We do not lower the case but filter out tweets with less than 10 tokens, producing 7,554,409 Democratic and 4,759,441 Republican tweets. We randomly sample from Democratic tweets to ensure both partisan communities have the same number of 4,759,441 tweets for training language models to ensure a fair community model comparison.

[1]The source code of our paper is available at: https://github.com/hjian42/CommunityLM

[2]https://github.com/social-machines/litecoder

[3]We update American politicians from https://ballotpedia.org/List_of_current_members_of_the_U.S._Congress and https://ballotpedia.org/Governor_(state_executive_office)

[4]https://www.pewresearch.org/politics/2020/10/15/differences-in-how-democrats-and-republicans-behave-on-twitter

3 Framework

We present a simple COMMUNITYLM framework which adapts GPT-style language models to mine community insights. This framework consists of four steps: (1) fine-tune or train GPT language models on community data, (2) design prompts based on survey questions, (3) generate community responses with language models, (4) aggregate community stance based on responses.

3.1 Model Training and Fine-tuning

We pick GPT-2 with 124M parameters and experiment with two training strategies on the partisan community data: (1) fine-tune a pre-trained GPT-2 model, (2) train a GPT-2 model from scratch. For both settings, we adopt training epoch 10 and batch size 24 on Nvidia GeForce GTX 1080 12GB. The greedy decoding is used for GPT-2. Otherwise, we use the default training parameters[5]. The pre-trained GPT-2 model was released in February 2019, trained on data that cuts off at the end of 2017. We also use GPT-3 Curie as one of our baselines, which used training data up to Oct 2019. Therefore, neither pre-trained model used any data beyond the the start date of the ANES survey.

We adopt 10 epochs because GPT-2 was not pre-trained on the Twitter domain and had a steady loss decrease across all epochs. We checked all synthetic tweets (lowercased) generated by the fine-tuned GPT-2 with "X is/are the". The percentages of synthetic tweets appearing in training data are 64.93% and 69.56% for Republican and Democratic models. For researchers who want to adapt our approach with a lower repetition rate, we suggest moving away from the greedy decoding algorithm and reducing the epoch number.

3.2 Prompt Design

We design discrete prompts based on survey questions to probe community insights towards public figures and groups. The **American National Election Studies (ANES)** are academically-run national surveys of voters in the United States. We adopt the ANES 2020 Exploratory Testing Survey[6] conducted between April 10, 2020 and April 18, 2020 on 3,080 adult citizens from across the United States, because this survey captures recent political changes in the US. We adapt all 30 questions from "FEELING THERMOMETERS" section of the ANES survey, which asks participants to rate people or groups from 0 ("not favorable") to 100 ("favorable") with the question "How would you rate ____?" The questions cover 30 items in two categories (a) **16 people**: Donald Trump, Barack Obama, Joe Biden, Elizabeth Warren, Bernie Sanders, Pete Buttigieg, Kamala Harris, Amy Klobuchar, Mike Pence, Andrew Yang, Nancy Pelosi, Marco Rubio, Alexandria Ocasio-Cortez, Nikki Haley, Clarence Thomas, Dr. Anthony Fauci, (b) **14 groups:** blacks, whites, Hispanics, Asians, illegal immigrants, feminists, the #MeToo movement, transgender people, socialists, capitalists, big business, labor unions, the Republican Party, the Democratic Party. For each item "X", we experiment with four types of discrete prompts: (1) "X", (2) "X is/are", (3) "X is/are a", (4) "X is/are the". These names are copied from the survey verbatim except for "whites", "blacks", "Hispanics", and "Asians" because "whites" and "blacks" also refer to other named entities such as "Blacks Clothing Company" and "Whites TV shows". Instead, we translate the names of these four groups into "White people", "Black people", "Hispanic people", "Asian people". We also provide the count number of each item in Appendix A.

3.3 Community Response Generation

For each community, we use the corresponding language model to generate 1000 responses given the prompts. We use Hugging Face's `TextGenerationPipeline` and apply the same decoding strategy by setting do_sample to true, temperature to 1.0, and max_length to 50. If one response contains multiple sentences, we use the first line in the response and remove the remaining tokens, because a response with multiple sentences may have mixed sentiments, making it hard to identify the overall sentiment.

3.4 Community Stance Aggregation

After response generation, we save them locally and compute the community stance for each prompt by aggregating the sentiment of the synthetic responses. Specifically, we use the state-of-the-art Twitter sentiment classifier "cardiffnlp/twitter-roberta-base-sentiment-latest" (Barbieri et al., 2020; Loureiro et al., 2022) on the SemEval-2017 benchmark (Rosenthal et al., 2017) to classify each

[5] https://github.com/huggingface/transformers/blob/main/examples/pytorch/language-modeling/run_clm.py

[6] https://electionstudies.org/data-center/2020-exploratory-testing-survey/

generated response into -1 ("Negative"), 0 ("Neutral"), and 1 ("Positive"). We take the average sentiment of the generated responses as the community's stance score towards the person or group. We also show the results of a popular lexicon-based sentiment classifier VADER in Appendix C.

4 Evaluation

Task Formulation. The ANES survey has self-reported party affiliation from participants. We use responses from Republican and Democratic participants and calculate their average ratings towards each of 30 items (persons and groups). These average ratings are provided in Appendix B. If the average rating of Republican participants is higher than that of Democratic participants toward one item (e.g., Joe Biden), it is labeled as "R". Otherwise, the item is labeled as "D". 70% items are labeled "D" and 30% "R". The 9 items with "R" label are Donald Trump, Mike Pence, Marco Rubio, Nikki Haley, Clarence Thomas, whites, capitalists, big business, and the Republican Party. The task asks a model to predict which community is more favorable towards an item. To address the data imbalance, we prefer weighted F1 to accuracy as a measure of model performance.

Baselines. We evaluate the performance of trained and fine-tuned COMMUNITYLM (GPT-2) against 4 baselines. The first baseline is Frequency Model which counts the frequency of an item's name in each community's data and classifies the community with higher word frequency to be the label. The second baseline is Keyword Retrieval which uses keywords to retrieve tweets containing the keywords from each community's data, computes the average community stance, and selects the community with a higher stance score. Keyword Retrieval (full) means using the full names as keywords and Keyword Retrieval (surname) means using the surname of people. The third and fourth baselines use pre-trained GPT-2 and pre-trained GPT-3 Curie respectively. "[CONTEXT]" is a preceding context "As a Democrat/Republican, I think", which is concatenated with the prompts to generate partisan responses on each item. We compute the average community stance on 1000 synthetic responses and pick the community with a higher average stance score. It is noted that we also fine-tune or train GPT-2 on the aggregate partisan tweets and show their results in Appendix D.

Overall Performance. First, we observe that fine-tuned COMMUNITYLM with "X is the" prompt achieves the best performance in both accuracy (97.33%) and weighted F1-score (97.29%) on the task. The same model's performance is sensitive to the prompt design and the longest prompt out of the four seems to work the best. "X" alone is bad, because it will result in many responses like "X @USER", "X???", "X.", which are common Twitter posts and are too short to interpret their attitudes. Second, fine-tuned COMMUNITYLM outperforms trained COMMUNITYLM from scratch. It indicates that pre-training GPT-2 is helpful, probably because pre-training injects the general knowledge about the named entities into GPT-2. Third, we find that Keyword Retrieval (surname) is a strong baseline in both accuracy (93.33%) and F1 (93.33%), but its performance is also sensitive to the selection of keywords. As we see, the weighted F1 performance of Keyword Retrieval (full), which uses a strict full name matching (e.g., "Joe Biden"), drops to 87.00%. In contrast, language models are able to learn the associations between different names for the same person and generalize without worrying about name forms. Last, fine-tuned COMMUNITYLM outperforms pre-trained GPT-2 and GPT-3 baselines. It is worth noting that the performance of pre-trained GPT-3 Curie is consistently better than pre-trained GPT-2. GPT-3 with the "X is/are" prompt achieves the same score as the Keyword Retrieval (surname) baseline.

Error Analysis. The rule-based Keyword Retrieval (surname) baseline misses "illegal immigrants" and "big business". The fine-tuned COMMUNITYLM with "X is/are the" misses "White people". The pre-trained GPT-3 with "X is/are" prompt misses "Dr. Anthony Fauci" and "Asian people'. It is interesting because the top 5 items with the closest average rating gap between ANES partisan participants are Asian people (5.5%), White people (5.9%), Hispanic people (7.7%), Dr. Anthony Fauci (8.4%), and Black people (9.7%).

Ranking Public Figures. We use the average community stance scores computed on the generated tweets from the fine-tuned COMMUNITYLM model to rank 16 public figures for each community, hoping to understand how they perceive these people. In Figure 1, we observe that Republican politicians are rated poorly by the Democratic model and vice versa. Overall, the ratings from the Republican model are more negative than the Democratic model. Interestingly, we find that An-

Model	Prompt	Accuracy	Weighted F1
Frequency Model	—	53.33	54.50
Keyword Retrieval (Full)	—	86.67	87.00
Keyword Retrieval (Surname)	—	93.33	93.33
Pre-trained GPT-2	"[CONTEXT] + X"	74.00±2.79	66.52±5.56
Pre-trained GPT-2	"[CONTEXT] + X is/are"	72.00±1.83	64.63±2.35
Pre-trained GPT-2	"[CONTEXT] + X is/are a"	75.33±1.83	68.47±3.35
Pre-trained GPT-2	"[CONTEXT] + X is/are the"	77.33±2.79	74.71±3.22
Pre-trained GPT-3 Curie	"[CONTEXT] + X"	83.33	83.88
Pre-trained GPT-3 Curie	"[CONTEXT] + X is/are"	93.33	93.50
Pre-trained GPT-3 Curie	"[CONTEXT] + X is/are a"	83.33	83.88
Pre-trained GPT-3 Curie	"[CONTEXT] + X is/are the"	83.33	84.02
Trained COMMUNITYLM	"X"	90.00±0.00	89.63±0.27
Trained COMMUNITYLM	"X is/are"	90.00±0.00	89.82±0.00
Trained COMMUNITYLM	"X is/are a"	86.00±1.49	86.25±1.50
Trained COMMUNITYLM	"X is/are the"	90.67±2.79	90.49±2.68
Fine-tuned COMMUNITYLM	"X"	84.67±2.98	84.46±3.18
Fine-tuned COMMUNITYLM	"X is/are"	96.00±2.79	96.00±2.79
Fine-tuned COMMUNITYLM	"X is/are a"	91.33±1.83	90.83±2.05
Fine-tuned COMMUNITYLM	"X is/are the"	**97.33±1.49**	**97.29±1.52**

Table 2: Performance of different approaches in accuracy to predict which community is more favorable towards 30 persons or groups from the ANES survey. Approaches based on GPT-2 are repeated five times to compute the average and standard deviation. GPT-3 is only run once for cost concern. Frequency Model and Keyword Retrieval methods are deterministic. The weighted average F1 is used because of data imbalance.

drew Yang is rated quite highly by both models, likely because of the sampling bias of Twitter. It is noted that "Andrew Yang" is also ranked 1st by the Democrat community and 3rd by the Republican community with the retrieval approach.

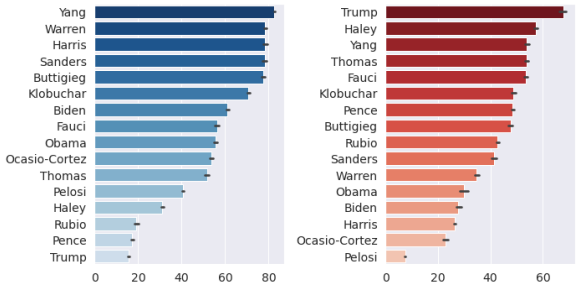

(a) Democratic ranking (b) Republican ranking

Figure 1: Left and right rankings of 16 public figures by their average stance scores calculated on synthetic tweets from their fine-tuned COMMUNITYLM models.

5 Conclusion

In this paper, we present a simple COMMUNITYLM framework to evaluate the viability of fine-tuned GPT-2 community language models in mining community insights in the context of political polarization between Republicans and Democrats. We adopt ANES survey questions and experiment with four types of prompts to generate community responses through GPT-2, showing that generated opinions are predictive about which community is more favorable towards selected public figures and groups. Our results show that fine-tuned COMMUNITYLM (GPT-2) outperforms the baseline methods. We analyze the model errors and run qualitative analyses to demonstrate that GPT-2 community language models can be used to rank public figures and probe word choices.

There are a few limitations in the current approach. First, language models can synthesize unreliable responses. Structured knowledge (Wang et al., 2021; Yasunaga et al., 2021) can be used to reduce nonsensical or unfaithful generation. Therefore, it is important that we use statistical patterns rather than individual synthesized tweets to draw conclusions (Feldman et al., 2021). Second, language models are shown to be sensitive to prompt design in our experiments and are also vulnerable to negation and misprimed probes (Kassner and Schütze, 2020). In the future, we plan to develop a systematic approach to design effective prompts and evaluate the robustness of COMMUNITYLM. Third, we focus on the classic red and blue polarization and do not consider a more fine-grained segmentation of U.S. politics. We hope to extend this work to study multiple sociopolitical communities in America and surface their unheard voices.

Ethical Considerations

We propose a general framework to probe community insights and observe differences between the Democratic and Republican communities on Twitter. While we do not discuss how to react to these findings, the intention of our research is encourage people to escape from their echo chambers, hear voices from other communities, and engage in constructive communication. One reasonable ethical concern is that by using a language model to predict community opinions, instead of asking individuals from the community directly, don't we risk erasing individual voices? To that concern we would like to emphasize that our model is no substitute for deeper engagement with a community; as discussed in the limitation paragraph, the language model is just an entry point for understanding a community's perspective. It serves to synthesize the points expressed by the speakers in the training data more effectively than we know how to do by hand. Any automated or semi-automated prediction system risks misinterpreting or "erasing" an expressed opinion, and we show in our work that the simpler methods of doing so are more error-prone, and hence measurably more unfair than the proposed approach in the paper.

Acknowledgements

We would like to thank anonymous reviewers for their helpful comments on our paper. We also want to thank Belén Carolina Saldías Fuentes, Will Brannon, Suyash Fulay, Wonjune Kang, Hope Schroeder, and Shayne Longpre for their discussion and feedback in the early stages of the project.

References

Christopher A Bail, Lisa P Argyle, Taylor W Brown, John P Bumpus, Haohan Chen, MB Fallin Hunzaker, Jaemin Lee, Marcus Mann, Friedolin Merhout, and Alexander Volfovsky. 2018. Exposure to opposing views on social media can increase political polarization. *Proceedings of the National Academy of Sciences*, 115(37):9216–9221.

Francesco Barbieri, Jose Camacho-Collados, Luis Espinosa Anke, and Leonardo Neves. 2020. TweetEval: Unified benchmark and comparative evaluation for tweet classification. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 1644–1650, Online. Association for Computational Linguistics.

L Brent Bozell. 2004. Weapons of mass distortion: The coming meltdown of the liberal media. Crown Forum.

Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel Ziegler, Jeffrey Wu, Clemens Winter, Chris Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. 2020. Language models are few-shot learners. In *Advances in Neural Information Processing Systems*, volume 33, pages 1877–1901. Curran Associates, Inc.

Kareem Darwish. 2019. Quantifying polarization on twitter: the kavanaugh nomination. In *International conference on social informatics*, pages 188–201. Springer.

Joe Davison, Joshua Feldman, and Alexander Rush. 2019. Commonsense knowledge mining from pretrained models. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 1173–1178, Hong Kong, China. Association for Computational Linguistics.

Dorottya Demszky, Nikhil Garg, Rob Voigt, James Zou, Jesse Shapiro, Matthew Gentzkow, and Dan Jurafsky. 2019. Analyzing polarization in social media: Method and application to tweets on 21 mass shootings. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 2970–3005, Minneapolis, Minnesota. Association for Computational Linguistics.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, Minneapolis, Minnesota. Association for Computational Linguistics.

Philip Feldman, Sim Tiwari, Charissa SL Cheah, James R Foulds, and Shimei Pan. 2021. Analyzing covid-19 tweets with transformer-based language models. *arXiv preprint arXiv:2104.10259*.

Homero Gil de Zúñiga, Teresa Correa, and Sebastian Valenzuela. 2012. Selective exposure to cable news and immigration in the us: The relationship between fox news, cnn, and attitudes toward mexican immigrants. *Journal of Broadcasting & Electronic Media*, 56(4):597–615.

Gordon Heltzel and Kristin Laurin. 2020. Polarization in america: Two possible futures. *Current opinion in behavioral sciences*, 34:179–184.

Clayton Hutto and Eric Gilbert. 2014. Vader: A parsimonious rule-based model for sentiment analysis of social media text. In *Proceedings of the international AAAI conference on web and social media*, volume 8, pages 216–225.

Ki Deuk Hyun and Soo Jung Moon. 2016. Agenda setting in the partisan tv news context: Attribute agenda setting and polarized evaluation of presidential candidates among viewers of nbc, cnn, and fox news. *Journalism & Mass Communication Quarterly*, 93(3):509–529.

Zhengbao Jiang, Jun Araki, Haibo Ding, and Graham Neubig. 2021. How can we know when language models know? on the calibration of language models for question answering. *Transactions of the Association for Computational Linguistics*, 9:962–977.

Nora Kassner and Hinrich Schütze. 2020. Negated and misprimed probes for pretrained language models: Birds can talk, but cannot fly. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7811–7818, Online. Association for Computational Linguistics.

Ping Li, Benjamin Schloss, and D Jake Follmer. 2017. Speaking two "languages" in america: A semantic space analysis of how presidential candidates and their supporters represent abstract political concepts differently. *Behavior research methods*, 49(5):1668–1685.

Daniel Loureiro, Francesco Barbieri, Leonardo Neves, Luis Espinosa Anke, and Jose Camacho-Collados. 2022. Timelms: Diachronic language models from twitter. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics: System Demonstrations*, Online. Association for Computational Linguistics.

Nolan McCarty, Keith T Poole, and Howard Rosenthal. 2016. *Polarized America: The dance of ideology and unequal riches*. mit Press.

Jeremiah Milbauer, Adarsh Mathew, and James Evans. 2021. Aligning multidimensional worldviews and discovering ideological differences. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 4832–4845, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Dat Quoc Nguyen, Thanh Vu, and Anh Tuan Nguyen. 2020. BERTweet: A pre-trained language model for English tweets. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 9–14, Online. Association for Computational Linguistics.

Shriphani Palakodety, Ashiqur R KhudaBukhsh, and Jaime G Carbonell. 2020. Mining insights from large-scale corpora using fine-tuned language models. In *ECAI 2020*, pages 1890–1897. IOS Press.

Fabio Petroni, Tim Rocktäschel, Sebastian Riedel, Patrick Lewis, Anton Bakhtin, Yuxiang Wu, and Alexander Miller. 2019. Language models as knowledge bases? In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 2463–2473, Hong Kong, China. Association for Computational Linguistics.

Keith T Poole and Howard Rosenthal. 1984. The polarization of american politics. *The journal of politics*, 46(4):1061–1079.

Ashiqur R. KhudaBukhsh, Rupak Sarkar, Mark S. Kamlet, and Tom Mitchell. 2021. We don't speak the same language: Interpreting polarization through machine translation. *Proceedings of the AAAI Conference on Artificial Intelligence*, 35(17):14893–14901.

Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, Ilya Sutskever, et al. 2019. Language models are unsupervised multitask learners. *OpenAI blog*, 1(8):9.

Sara Rosenthal, Noura Farra, and Preslav Nakov. 2017. SemEval-2017 task 4: Sentiment analysis in Twitter. In *Proceedings of the 11th International Workshop on Semantic Evaluation (SemEval-2017)*, pages 502–518, Vancouver, Canada. Association for Computational Linguistics.

Alon Talmor, Yanai Elazar, Yoav Goldberg, and Jonathan Berant. 2020. oLMpics-on what language model pre-training captures. *Transactions of the Association for Computational Linguistics*, 8:743–758.

Svitlana Volkova, Glen Coppersmith, and Benjamin Van Durme. 2014. Inferring user political preferences from streaming communications. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 186–196, Baltimore, Maryland. Association for Computational Linguistics.

Xiaozhi Wang, Tianyu Gao, Zhaocheng Zhu, Zhengyan Zhang, Zhiyuan Liu, Juanzi Li, and Jian Tang. 2021. KEPLER: A Unified Model for Knowledge Embedding and Pre-trained Language Representation. *Transactions of the Association for Computational Linguistics*, 9:176–194.

Michihiro Yasunaga, Hongyu Ren, Antoine Bosselut, Percy Liang, and Jure Leskovec. 2021. QA-GNN: Reasoning with language models and knowledge graphs for question answering. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 535–546, Online. Association for Computational Linguistics.

Appendix

A Keyword counts

The Keyword Retrieval baseline method retrieves tweets containing the keywords. Here we show the list of full and surname keywords and their counts in tables 3 and 4, respectively, for the Republican and Democratic tweets. For corresponding items between these two tables (e.g. "Asian people" in Table 3 to "Asian" in Table 4) there is a consistent increase in counts, especially for "Asian people" "Anthony Fauci", "Hispanic people", "labor unions", "Clarence Thomas". Some items in Table 4 might have too many counts. For example, we observe that "Trump" has 150,000+ counts in both partisan tweets, which can take a relatively long time for sentiment classifiers to run.

Keyword	Question	Dem	Repub
Asian people	ftasian	81	21
Joe Biden	ftbiden1	4177	5377
big business	ftbigbusiness	321	291
Black people	ftblack	3199	1278
Pete Buttigieg	ftbuttigieg1	982	521
capitalists	ftcapitalists	279	197
the Democratic Party	ftdemocraticparty	2094	2646
Anthony Fauci	ftfauci1	102	85
feminists	ftfeminists	351	628
Nikki Haley	fthaley1	169	274
Kamala Harris	ftharris1	1711	1450
Hispanic people	fthisp	28	21
illegal immigrants	ftillegal	251	2233
Amy Klobuchar	ftklobuchar1	451	193
labor unions	ftlaborunions	68	27
the #MeToo movement	ftmetoo	103	84
Barack Obama	ftobama1	684	929
Alexandria Ocasio-Cortez	ftocasioc1	410	534
Nancy Pelosi	ftpelosi1	1467	3549
Mike Pence	ftpence1	911	502
the Republican Party	ftrepublicanparty	1681	838
Marco Rubio	ftrubio1	166	132
Bernie Sanders	ftsanders1	4572	2711
socialists	ftsocialists	627	2697
Clarence Thomas	ftthomas1	157	132
transgender people	fttransppl	165	38
Donald Trump	fttrump1	8501	5479
Elizabeth Warren	ftwarren1	3132	1897
White people	ftwhite	3625	1862
Andrew Yang	ftyang1	585	249

Table 3: Counts of full names for each person and group in Republican and Democratic tweets.

B What are the average ratings between partisan participants in ANES survey?

We also compute and show in Table 5 the average ratings from Republican and Democratic participants towards each person or group. For most items, we observe quite large rating gaps between the partisans. But the top 5 items with the closest average rating gap between partisans are "Asian people" (5.5%), "White people" (5.9%), "Hispanic people" (7.7%), "Dr. Anthony Fauci" (8.4%),

Keyword	Question	Dem	Repub
Asian	ftasian	2961	1917
Biden	ftbiden1	26558	21748
big business	ftbigbusiness	321	291
Black people	ftblack	3199	1278
Buttigieg	ftbuttigieg1	3514	1348
capitalist	ftcapitalists	1393	941
Democratic Party	ftdemocraticparty	2677	3611
Fauci	ftfauci1	931	1219
feminist	ftfeminists	1686	1470
Haley	fthaley1	531	712
Harris	ftharris1	6753	5416
Hispanic	fthisp	1173	1693
illegal immigrant	ftillegal	312	2815
Klobuchar	ftklobuchar1	1958	584
labor union	ftlaborunions	110	47
#MeToo movement	ftmetoo	114	102
Obama	ftobama1	15390	33105
Ocasio-Cortez	ftocasioc1	751	1792
Pelosi	ftpelosi1	5985	15844
Pence	ftpence1	5818	3021
Republican Party	ftrepublicanparty	2251	1079
Rubio	ftrubio1	508	502
Sanders	ftsanders1	16001	6568
socialist	ftsocialists	3182	12606
Thomas	ftthomas1	2316	3348
transgender	fttransppl	1309	1469
Trump	fttrump1	188170	150589
Warren	ftwarren1	18954	6969
White people	ftwhite	3625	1862
Yang	ftyang1	4443	1433

Table 4: Counts of surname names for each person and group in Republican and Democratic tweets.

"Black people" (9.7%). These items have very close ratings and we confirm in our error analysis that they are also challenging to the GPT-2 models. It is worth noting that the survey was done in early 2020 and at that time "Dr. Fauci" as a topic was not as divisive as it is today on Twitter.

C How well does the system perform using a lexicon-based sentiment classifier?

In the main paper, we use a state-of-the-art pre-trained BERT Twitter sentiment classifier to classify tweets. Some researchers may be concerned that neural sentiment models may learn and reflect biases in the training data and prefer using lexicon-based approaches. Therefore, we also use VADER (Hutto and Gilbert, 2014)[7], a popular rule-based model for sentiment analysis of social media texts, and report the performance of our models with VADER in Table 6. Overall, we show that these models perform slightly worse with VADER, but we still see that fine-tuned COMMUNITYLM with "X is the" perform the best (93.33%) out of these models. This performance is on par with the Keyword Retrieval (surname) approach. We conjecture that using prompts like "X is the" creates many

[7] https://github.com/cjhutto/vaderSentiment

Question	Item	Dem	Repub
ftasian	Asian people	68.95	63.44
ftwhite	White people	71.25	77.16
fthisp	Hispanic people	71.27	63.60
ftfauci1	Dr. Anthony Fauci	66.67	58.28
ftblack	Black people	76.22	66.51
ftrubio1	Marco Rubio	31.52	43.01
ftcapitalists	capitalists	46.68	60.53
ftbigbusiness	big business	43.14	57.85
ftlaborunions	labor unions	60.67	44.87
fthaley1	Nikki Haley	29.86	47.07
ftthomas1	Clarence Thomas	29.95	48.63
ftyang1	Andrew Yang	49.28	29.19
ftklobuchar1	Amy Klobuchar	50.04	22.17
ftfeminists	feminists	61.97	33.92
fttransppl	transgender people	63.22	35.06
ftsocialists	socialists	54.00	24.11
ftillegal	illegal immigrants	56.17	26.25
ftmetoo	the #MeToo movement	63.74	32.73
ftbuttigieg1	Pete Buttigieg	52.79	21.66
ftharris1	Kamala Harris	52.12	18.63
ftocasioc1	Alexandria Ocasio-Cortez	50.60	16.49
ftwarren1	Elizabeth Warren	59.84	20.46
ftbiden1	Joe Biden	66.50	24.40
ftsanders1	Bernie Sanders	63.77	20.50
ftpelosi1	Nancy Pelosi	61.76	16.10
ftdemocraticparty	the Democratic Party	71.24	24.34
ftpence1	Mike Pence	24.09	71.12
ftrepublicanparty	the Republican Party	25.02	74.47
ftobama1	Barack Obama	81.29	29.99
fttrump1	Donald Trump	17.66	77.83

Table 5: Average rating of each item (person or group) from Republican and Democratic participants in the ANES survey.

synthetic tweets with only sentiment-neutral lexical items (e.g., "big business is the future") which the lexicon-based VADER is not able to classify as "positive". The BERT sentiment classifier, however, performs better at representing the overall semantics of the sentence and therefore is preferred in our framework.

Model	Prompt	Accuracy
Keyword Retrieval (Full)	—	76.67
Keyword Retrieval (Surname)	—	93.33
Pre-trained GPT-2	"[CONTEXT] + X"	76.67±0.00
Pre-trained GPT-2	"[CONTEXT] + X is/are"	76.00±1.49
Pre-trained GPT-2	"[CONTEXT] + X is/are a"	78.67±1.83
Pre-trained GPT-2	"[CONTEXT] + X is/are the"	74.67±5.06
Trained COMMUNITYLM	"X"	91.33±3.80
Trained COMMUNITYLM	"X is/are"	84.67±3.80
Trained COMMUNITYLM	"X is/are a"	82.00±3.80
Trained COMMUNITYLM	"X is/are the"	93.33±4.08
Fine-tuned COMMUNITYLM	"X"	92.00±1.83
Fine-tuned COMMUNITYLM	"X is/are"	92.67±1.49
Fine-tuned COMMUNITYLM	"X is/are a"	91.33±1.83
Fine-tuned COMMUNITYLM	"X is/are the"	93.33±2.36

Table 6: Performance of different approaches with VADER in predicting which community is more favorable towards 30 persons or groups from the ANES survey. Approaches based on GPT-2 are repeated five times to compute the average and standard deviation.

D Is fine-tuning or training GPT-2 on combined Twitter data performing better than pre-trained GPT-2?

In the main paper, we use pre-trained GPT-2 and GPT-3 to predict the community stance. In addition, we also experimented with training and fine-tuning GPT-2 on the combined Twitter corpus (Republican and Democratic tweets). By contrast with COMMUNITYLM, which fine-tunes two GPT-2 models on partisan Twitter data, in this variant we only train or fine-tune one GPT-2 model on the aggregate of the partisan tweets. Similar to what we did in the pre-trained GPT-2 setting, we use [CONTEXT]+prompt to generate responses. The results are quite interesting, because the performance of the resulting models are worse than the pre-trained GPT-2, even below the majority baseline of 70%. We conjecture that this is because the combined data of partisan tweets neutralizes the sentiment that the models were supposed to learn towards the public figures and groups.

Model	Prompt	Accuracy
Trained GPT-2 (combined)	"[CONTEXT] + X"	48.67±8.37
Trained GPT-2 (combined)	"[CONTEXT] + X is/are"	50.67±7.23
Trained GPT-2 (combined)	"[CONTEXT] + X is a"	47.33±2.79
Trained GPT-2 (combined)	"[CONTEXT] + X is the"	55.33±8.37
Fine-tuned GPT-2 (combined)	"[CONTEXT] + X"	53.33±4.08
Fine-tuned GPT-2 (combined)	"[CONTEXT] + X is/are"	50.67±3.65
Fine-tuned GPT-2 (combined)	"[CONTEXT] + X is a"	52.67±2.79
Fine-tuned GPT-2 (combined)	"[CONTEXT] + X is the"	38.00±8.37

Table 7: Performance of trained and fine-tuned GPT-2 on combined Twitter data in accuracy to predict which community is more favorable towards 30 persons or groups from the ANES survey. Experiments are repeated five times to compute the average and standard deviation.

Composition-based Heterogeneous Graph Multi-channel Attention Network for Multi-aspect Multi-sentiment Classification

Hao Niu[1], **Yun Xiong**[1]*, **Jian Gao**[2], **Zhongchen Miao**[2], **Xiaosu Wang**[1], **Hongrun Ren**[1], **Yao Zhang**[1], **Yangyong Zhu**[1]
[1]Shanghai Key Laboratory of Data Science, School of Computer Science, Fudan University
[2]Shanghai Financial Futures Information Technology Co., Ltd.
[1]{hniu18, yunx, xswang19, renhr20, yaozhang, yyzhu}@fudan.edu.cn
[2]{gaojian, miaozc}@cffex.com.cn

Abstract

Aspect-based sentiment analysis (ABSA) has drawn more and more attention because of its extensive applications. However, towards the sentence carried with more than one aspect, most existing works generate an aspect-specific sentence representation for each aspect term to predict sentiment polarity, which neglects the sentiment relationship among aspect terms. Besides, most current ABSA methods focus on sentences containing only one aspect term or multiple aspect terms with the same sentiment polarity, which makes ABSA degenerate into sentence-level sentiment analysis. In this paper, to deal with this problem, we construct a heterogeneous graph to model inter-aspect relationships and aspect-context relationships simultaneously and propose a novel Composition-based Heterogeneous Graph Multi-channel Attention Network (CHGMAN) to encode the constructed heterogeneous graph. Meanwhile, we conduct extensive experiments on three datasets: MAMS-ATSA, Rest14, and Laptop14, experimental results show the effectiveness of our method.

1 Introduction

Aspect-based sentiment analysis (ABSA) is a sentiment analysis task, which aims to predict the sentiment polarity (e.g., POSITIVE, NEGATIVE, NEUTRAL) towards the given aspect term in a sentence. For example, there are two aspect terms *decor* and *food* in the sentence: *The decor is not a special at all but their amazing food makes up for it*, and these two aspect terms will be assigned with NEGATIVE and POSITIVE sentiment polarities respectively. Since the sentiments expressed by these two aspect terms are opposite in polarity, it is unreasonable to assign a sentence-level sentiment polarity. In this regard, ABSA can provide more detailed sentimental predictions compared with sentence-level sentiment analysis.

In recent years, neural network-based methods (Tang et al., 2016; Wang et al., 2016; Ma et al., 2017; Chen et al., 2017) are used to tackle ABSA task and achieve promising performance. Subsequently, with the rise of contextual embedding, contextualized language models, such as BERT (Devlin et al., 2019), are introduced into ABSA and boost performance. Additionally, in order to utilize the dependency tree information, a lot of graph-based methods (Zhang et al., 2019; Sun et al., 2019; Huang and Carley, 2019; Wang et al., 2020) are proposed to tackle this task by encoding both contextual and dependency information simultaneously. However, most of these methods focus on the datasets, in which most sentences consist of only one aspect or multiple aspects with the same sentiment polarity. Under these circumstances, aspect-based sentiment analysis degenerates into sentence-level sentiment analysis. Existing ABSA methods based on these datasets can hardly adapt to the situation, where multiple aspect terms in a sentence with multiple different sentiment polarities. Thus, the fine-grained sentiment analysis task derived from ABSA towards the multi-aspect multi-sentiment situation, Multi-aspect Multi-sentiment Classification (MAMSC), is worthy of exploration(Jiang et al., 2019). MAMSC is more challenging than ABSA, and each sentence contains at least two aspects with different sentiment polarities in MAMSC.

Besides, most existing methods neglect to consider inter-aspect relationships when predicting sentiment polarity. Especially in MAMSC, the main challenge is that the sentiment polarities of some aspects in a given sentence can not be judged solely relied on the context and may be guided by the sentiments of other aspects. Inter-aspect relationships will play an essential role in identifying sentiment polarity in this case. Though InterGCN (Liang et al., 2020) claims to have considered inter-aspect relationships, what it considered is merely the distance relationship among aspects and neglecting

*Corresponding author

Proceedings of the 29th International Conference on Computational Linguistics, pages 6827–6836
October 12–17, 2022.

the semantic relationship between every two aspects, which is essential when indicating the relationship between two aspects. Furthermore, InterGCN ignores the heterogeneity of aspect terms and context words and the different importance of surrounding nodes when aggregating information.

To sum up, ABSA faces the following challenges: (1) do not pay special attention to the multi-aspect multi-sentiment situation; (2) do not incorporate semantic relationships among aspects into predicting sentiment polarity; (3) do not introduce heterogeneity and attention mechanism when using graph-based models to predict sentiment polarity.

To deal with the challenges mentioned above, we first construct a heterogeneous graph with distinct node and edge types to represent inter-aspect relationships, aspect-context relationships, and node heterogeneity simultaneously in a concrete way. Unlike InterGCN, we treat aspect terms and context words as different types of nodes, and meanwhile, the edges in our heterogeneous graph are attributed. We utilize dependency tags as the attributes of aspect-context edges. The shortest dependency path between each two aspect terms is the representation of each inter-aspect edge attribute, which signifies the semantic relationship between these two aspect terms. Then we propose a novel model called *Composition-based Heterogeneous Graph Multi-channel Attention Network (CHGMAN)* to capture important information conveyed by constructed heterogeneous graph. Our proposed CHGMAN equipped with a multi-channel attention mechanism can take edge features, node types, and node features into account concurrently and aggregate composition information of nodes and node types in terms of importance, which is conducive to capturing the information expressed by our heterogeneous graph and making the correct predictions. The main contributions of this paper can be summarized as follows:

Heterogeneous Graph: to the best of our knowledge, this is the first attempt to model multi-aspect multi-sentiment classification with heterogeneous graph networks to capture both inter-aspect relationships and context-aspect relationships.

Multi-channel Attention: we propose a novel model *Composition-based Heterogeneous Graph Multi-channel Attention Network (CHGMAN)* to incorporate inter-aspect relationship information, node and node type information simultaneously for predicting sentiment polarity.

Extensive Experiments: we conduct extensive experiments over multi-aspect multi-sentiment datasets: MAMS-ATSA, Rest14, and Laptop14, and the results show the effectiveness of our method. [1]

2 Related Work

2.1 Graph-based Models

Recently, graph neural networks (GNNs) have been introduced into ABSA and achieved promising performance. ASGCN (Zhang et al., 2019) employs graph convolutional network (GCN) to capture dependency syntactical information. CDT (Sun et al., 2019) incorporates GCN and BiLSTM over the dependency information and contextual information of the sentence. InterGCN (Liang et al., 2020) exploits GCN to extract both aspect-focused and inter-aspect information for specific aspects, but it only considers distance relations among aspect terms. RGAT (Wang et al., 2020) designs a graph attention networks (GAT) derivative to utilize dependency tag information by taking them as gates to control information flow. However, all these models do not utilize inter-aspect semantic relationships and neglect the different roles of aspect terms and context words.

2.2 Multi-aspect Multi-sentiment

There exist several works that tend to model inter-aspect relationships to boost the performance of ABSA. IARM (Majumder et al., 2018) utilizes memory networks to incorporate the neighboring aspects-related information to predict the sentiment polarity of the given aspect term. InterGCN models inter-aspect distance relationships to help predict sentiment polarity in the presence of other aspects. SDGCN (Zhao et al., 2020) utilizes GCNs to capture sentiment dependencies between aspects. Besides, MIAD (Hazarika et al., 2018), STAGE (Ma et al., 2019) and Joint+PERT (Zhou et al., 2020) also tend to model relationships among aspects by using LSTM based model and attention mechanism. CapsNet (Jiang et al., 2019) releases a new dataset dedicated to the multi-aspect multi-sentiment situation and proposes a model to employ capsule network and capture the intricate relationship between aspect terms and context words. Nonetheless, all of these existing works do not take edge information, such as dependency syntactic and semantic

[1] The code is available at https://github.com/hankniu01/chgman

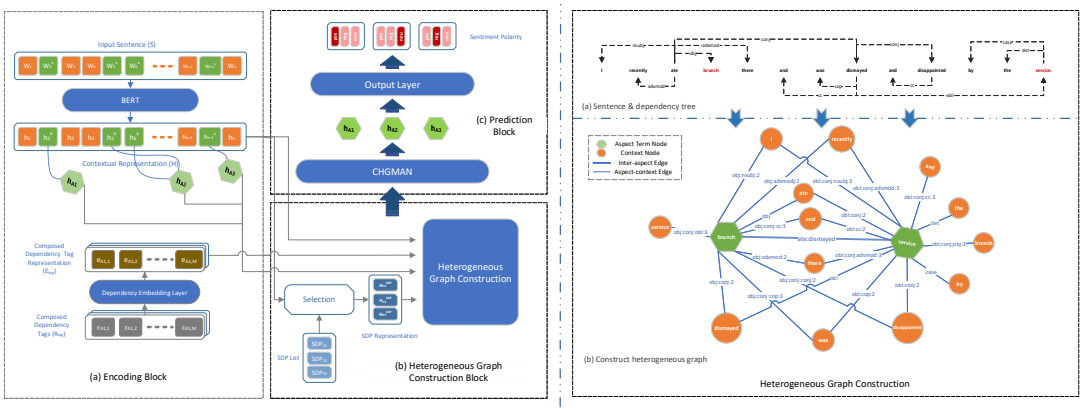

Figure 1: The overall architecture of our proposed method (left). The heterogeneous graph construction and composed dependency tags is shown in right.

relationship among aspects into consideration, and heterogeneity between aspect terms and context words is also ignored.

3 Methodology

Multi-aspect multi-sentiment classification can be formulated as follows: given a sentence S with n words $\{w_1, w_2, ..., w_n\}$ and a given aspect term set $\mathcal{A} = \{A_1, A_2, ..., A_\kappa\}$ containing κ aspect terms. And $A_k = \{w^a_\varphi, w^a_{\varphi+1}, ..., w^a_{\varphi+\tau}\}$ with $\tau \in [1, n-\varphi]$ is a sub-string of sentence S, which denotes one element of the given aspect term set $\mathcal{A}$. The annotation a represents aspect. This task aims at predicting the sentiment polarity y {POSITIVE, NEUTRAL, NEGATIVE} expressed on each aspect term included in the given aspect term set $\mathcal{A}$ of the sentence S. As shown in Figure 1(left), our proposed methods consist of three blocks: (1) Encoding Block; (2) Heterogeneous Graph Construction Block; (3) Prediction Block. In Section 3.1 to 3.3, we detail each block.

3.1 Encoding Block

As shown in Figure 1(left), we encode context words, aspect terms, and composed dependency tags, preparing for constructing heterogeneous graphs.

3.1.1 Contextual Encoder

For a given input sentence $S = \{w_1, w_2, ..., w_n\}$ with a aspect term set $\mathcal{A} = \{A_1, A_2, ..., A_\kappa\}$, we first employ pre-trained BERT (Devlin et al., 2019) as the contextual encoder to obtain hidden contextual representation H of the given sentence S. The embeddings of context words $H_w = \{h_1, h_2, ..., h_{n_w}\}$ and aspect term embeddings $H_A = \{h_{A_1}, h_{A_2}, ..., h_{A_\kappa}\}$ are derived from H. Here, n_w is the number of context words in the input sentence S. Specifically, the embedding of a specific aspect term A_k is $h_{A_k} = \{h^a_\varphi, h^a_{\varphi+1}, ..., h^a_{\varphi+\tau}\}$, which is the collection of all the embeddings of words belonging to A_k.

3.1.2 Composed Dependency Tags

Additionally, we design composed dependency tags to fully utilize dependency relation tag information based on the aspect-oriented dependency tree (Wang et al., 2020). Concretely, as shown in Figure 1(right), the form of composed dependency tag representation is $r_{A_k,i} = dep_1 : dep_2 : ... : dep_m : m$, where m represents the hops between aspect term node A_k and neighbor node i, and dep_m represents the original dependency tag between two nodes generated by the Biaffine Parser (Dozat and Manning, 2017). Then, we employ a trainable embedding lookup table as a dependency embedding layer. We feed a set of composed dependency tags $R_{tag} = \{r_{A_k,1}, r_{A_k,2}, ..., r_{A_k,M}\}$ into dependency embedding layer to initialize composed dependency tag representations $E_{tag} = \{e_{A_k,1}, e_{A_k,2}, ..., e_{A_k,M}\}$, where M is the number of dependency relation tags. We have different sets of composed dependency tags and their representations for different aspect terms.

3.2 Heterogeneous Graph Construction Block

In this section, we construct a heterogeneous graph to encode inter-aspect relationships and aspect-context relationships effectively.

3.2.1 Shortest Dependency Path

To discover the semantic relationship between aspect terms, we extract the shortest dependency path

(SDP) between aspect terms to represent the relationship. SDP can indicate semantic relationships effectively between two words and ignore irrelevant information, which has been widely used in relation extraction (Bunescu and Mooney, 2005). All tokens on SDP between aspect terms A_ξ and A_η are represented as $SDP_{\xi\eta}$. Furthermore, the features of all tokens belonging to $SDP_{\xi\eta}$ are extracted from H derived from pre-trained BERT through selection operating. The selection operating is based on the index of the token belonging to SDP in the sentence. The $SDP_{\xi\eta}$ representation $H^{SDP}_{\xi\eta} = \sum [h^{SDP}_1, h^{SDP}_2, ..., h^{SDP}_t]$, where t is the number of tokens in the SDP.

3.2.2 Heterogeneous Graph Construction

The heterogeneous graph contains two types of nodes: aspect term node and context node. The feature of each aspect term node is h_{A_k}, which is the element of aspect embedding set H_A. The feature of each context node is derived from word embedding $H_w = \{h_1, h_2, ..., h_{n_w}\}$. Besides, there exist two types of edges in our constructed heterogeneous graph: composed dependency edge and SDP edge. Both of these edges are feature . The feature of each composed dependency edge is derived from composed dependency tag representations $E_{tag} = \{e_{A_k,1}, e_{A_k,2}, ..., e_{A_k,M}\}$, and the feature of each SDP edge between aspect terms A_ξ and A_η is $H^{SDP}_{\xi\eta}$. A simple case of heterogeneous graph construction is shown in Figure 1(right).

3.3 Prediction Block

In the prediction block, we utilize the constructed heterogeneous graph to make predictions. First, we propose a Composition-based Heterogeneous Graph Multi-channel Attention Network (CHGMAN) and feed the constructed heterogeneous graph with its features into CHGMAN. Then, the output layer receives features of aspect terms output by CHGMAN and predicts sentiment polarities.

3.3.1 Composition-based Heterogeneous Graph Multi-channel Attention Network (CHGMAN)

To encode our constructed heterogeneous graph effectively, we propose a novel multi-channel attention network for heterogeneous graphs, which concerns the information of edges, nodes, and node types simultaneously. Figure 2(left) presents the whole view of CHGMAN. Concretely, the goal of CHGMAN is to aggregate information from each neighbor node i to update the representation of the target node t. And the process can be divided into three phases: (1) Multi-channel Attention Calculation, (2) Heterogeneous Massage Passing, and (3) Composed Aggregation.

Multi-channel Attention Calculation. In first phase, CHGMAN tend to calculate the attention score matrices between target node t and each neighbor node $i \in \mathcal{N}(t)$. Due to heterogeneity, based on considering neighbor node features, we further tend to incorporate the information of node type and edge feature into our model. Thus, we design a multi-channel attention mechanism to take node types and edge features into consideration in the phase of attention calculation. Concretely, this phase of CHGMAN is decomposed into three channels: *target-to-neighbor* channel, *target-to-edge* channel and *target-to-type* channel to calculate attention score matrices from different sides. Then, CHGMAN exploits a multi-channel fusion layer to fuse attention score matrices from each channel and get a weighted combination of attention score matrices. Plus, we also employ a multi-head attention mechanism (Vaswani et al., 2017) to enhance the representation ability of CHGMAN.

Firstly, we map the feature of target node t into a query vector $Q^u_{A_k}$ at u-head attention, and h^l_t is the representation of target node t at l-th layer.

$$Q^u_t = Linear^{Q,u}_t \left(h^l_t\right), \tag{1}$$

where $Linear^{Q,u}_t$ is the linear projection for h^l_t. In order to exploit information from neighbor node features, neighbor node types, and edges features, we map the feature of them into key vectors for each channel. Specifically,

$$K^u_{neigh} = Linear^{K,u}_{neigh} \left(h^{neigh,l}_i\right), \tag{2}$$

$$K^u_{edge} = Linear^{K,u}_{edge} \left(h^{edge,l}_{t,i}\right), \tag{3}$$

$$K^u_{type} = Linear^{K,u}_{type} \left(h^{type,l}_i\right), \tag{4}$$

where $Linear^{K,u}_{neigh}$, $Linear^{K,u}_{edge}$ and $Linear^{K,u}_{type}$ are linear projections for each representations, and h^{neigh}_i represents the feature of neighbor node i. Here, neighbor nodes consist of aspect term nodes and context nodes. Besides, $h^{edge}_{t,i}$ is the representation of edge feature between the target node t and neighbor node i. There are two kinds of edges: inter-aspect edges and aspect-context edges. For inter-aspect edges, we utilize SDP representation $H^{SDP}_{\xi\eta}$ as features of

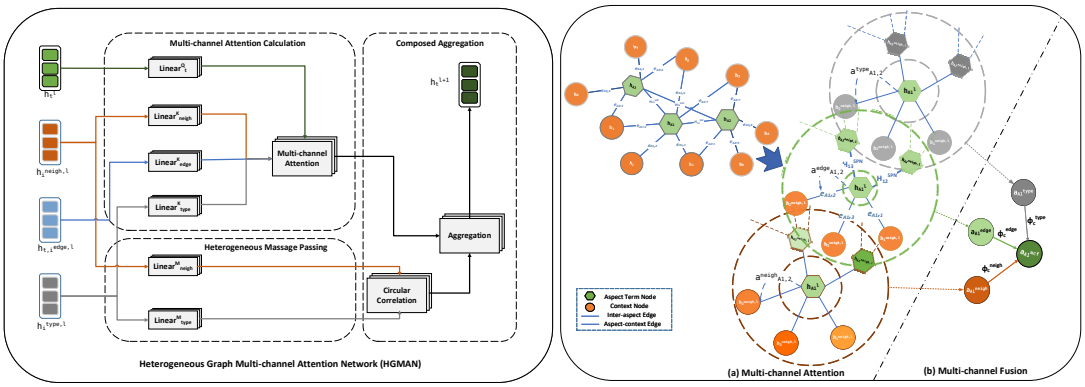

Figure 2: Overview of the proposed CHGMAN (left). Multi-channel attention calculation (right).

inter-aspect edges, and for aspect-context edges, we employ composed dependency tag representations. Additionally, the type representation of neighbor node i is h_i^{type}, and there exist two kinds of type representations corresponding to context node and aspect term node. We introduce a trainable embedding lookup table to initialize type representations.

Next, CHGMAN needs to calculate the correlations between query vectors and key vectors by dot product in each channel. The final multi-channel attention score $Att^u(t,i)$ is computed as follows:

$$Att^u(t,i) = \underset{\forall i \in \mathcal{N}(t)}{Softmax}\left(\alpha_{t,i}^{mcf,u}\right), \tag{5}$$

$$\begin{aligned}\alpha_{t,i}^{mcf,u} &= Mcf\left(\alpha_{t,i}^{neigh,u}, \alpha_{t,i}^{edge,u}, \alpha_{t,i}^{type,u}\right) \\ &= \Phi_c\left(\left[\alpha_{t,i}^{neigh,u}, \alpha_{t,i}^{edge,u}, \alpha_{t,i}^{type,u}\right]\right) \\ &= \Phi_c\left(\left[\frac{(Q_t^u)^\top K_{neigh}^u}{\sqrt{d}}, \frac{(Q_t^u)^\top K_{edge}^u}{\sqrt{d}}, \frac{(Q_t^u)^\top K_{edge}^u}{\sqrt{d}}\right]\right),\end{aligned} \tag{6}$$

where $[\alpha_{Ak,i}^{neigh,u}, \alpha_{Ak,i}^{edge,u}, \alpha_{Ak,i}^{type,u}]$ is the stacked attention matrices of different channels, and Mcf denotes a multi-channel fusion layer with parameters Φ_c, which is a learnable parameter to adjust contributions of input attention matrices automatically. We apply a scaling factor $\frac{1}{\sqrt{d}}$ on $\alpha_{t,i}^u$, which is beneficial to stabilize model training (Vaswani et al., 2017), and d is the dimension of hidden state.

Heterogeneous Massage Passing. Parallel to the computation of multi-channel attention, CHGMAN also passes information from neighbor nodes to the given target node. Taking into account the heterogeneity of nodes, we would like to incorporate the representations of node types attached with node features into the message passing process. Specifically, for target node t and its neighbor nodes $i \in \mathcal{N}(t)$, the multi-heads message is calculate as follows:

$$Msg_{neigh}^u(t,i) = Linear_{neigh}^{M,u}\left(h_i^{neigh,l}\right), \tag{7}$$

$$Msg_{type}^u(t,i) = Linear_{type}^{M,u}\left(h_i^{type,l}\right). \tag{8}$$

To obtain the u-th message head $Msg_{neigh}^u(t,i)$ and $Msg_{type}^u(t,i)$ from neighbor node features and the type representations of neighbor nodes, CHGMAN employs linear projections $Linear_{neigh}^{M,u}$ and $Linear_{type}^{M,u}$ to project them into message vectors.

Composed Aggregation. After multi-channel attention calculation and heterogeneous message passing, CHGMAN needs to aggregate messages from neighbor nodes to the given target node t. Firstly, we employ a circular correlation as a composition operation (Vashishth et al., 2020; Nickel et al., 2016) to combine the messages from node features and node type representations. The operation is computed as follows:

$$\begin{aligned}Msg_{corr}^u(t,i) &= \psi(Msg_{neigh}^u(t,i), Msg_{type}^u(t,i)) \\ &= \mathcal{F}^{-1}\left(\overline{\mathcal{F}(Msg_{neigh}^u(t,i))} \odot \mathcal{F}(Msg_{type}^u(t,i))\right),\end{aligned} \tag{9}$$

where $Msg_{corr}^u(t,i)$ is the composed message representation from neighbor node i, ψ is the circular correlation operation, and $\mathcal{F}(\cdot)$ and $\mathcal{F}^{-1}(\cdot)$ denote the Fast Fourier Transform (FFT) and its inverse. The $\overline{\mathcal{F}(\cdot)}$ is the complex conjugate of $\mathcal{F}(\cdot)$, and $\odot$ denotes the Hadamard product.

Then, we aggregate messages from neighbor nodes and update the target node representation The aggregate process at l-th layer is

$$h_t^{l+1} = ||_{u=1}^{U} \sum_{i \in \mathcal{N}(t)} \left(Att^u(t,i) \cdot Msg_{corr}^u(t,i)\right). \tag{10}$$

Dataset	Positive		Neutral		Negative		MM Statistics					
	Train	Test	Train	Test	Train	Test	Train Size	Train MM	Test Size	Test MM	Total Size	Total MM
MAMS-ATSA	3380	400	5042	607	2764	329	11186	11186	1336	1336	12522	12522
Rest14	2164	728	807	196	637	196	3608	2594	1120	835	4728	3429
Laptop14	994	341	870	128	464	169	2328	1396	638	379	2965	1775

Table 1: Statistics of MAMS-ATSA, Rest14, and Laptop14 datasets. (Train|Test|Total) Size and (Train|Test|Total) MM denote the number of instances and multi-aspect multi-sentiment instances in the training, testing, and overall (sum of training and testing) datasets, respectively.

We also incorporate multi-heads attention mechanism into CHGMAN, and $||_{u=1}^{U} x_u$ denotes the concatenation of vectors from x_1 to x_U. The aggregate process at l-th layer is calculated as follows:

3.3.2 Output Layer

In the end, we obtain the output feature of the given target aspect term node A_k from CHGMAN as the final representation h_{final}. Then, we feed h_{final} into a fully connected softmax layer and map it to probabilities over the different sentiment polarities.

$$P(y = c) = softmax(W_P h_{final} + b_P), \quad (11)$$

where W_P and b_P are the weight matrix and bias, respectively. $P \in \mathbb{R}^C$ is the probability distribution for the sentiment polarity of a specific aspect term, where C is the set of sentiment classes. The training objective is to minimize the standard cross-entropy loss with L2-regularization:

$$L(\Theta) = -\sum_{(S,A)\in D} log(y = c) + \Lambda||\Theta||_2, \quad (12)$$

where D is the set of training data, Θ represents all trainable parameters, and Λ is the coefficient of the L2-regularization term.

4 Experiments

4.1 Datasets and Experiment Settings

To verify the effectiveness of our proposed model, we conduct experiments on MAMS-ATSA, Rest14, and Laptop14. MAMS-ATSA is released by (Jiang et al., 2019), all sentences in MAMS-ATSA contain multiple aspect terms, and at least two of them with different sentiment polarities. Also, the start and end positions of each aspect term in a sentence are provided. Rest14 and Laptop14 (Pontiki et al., 2014) have been widely used, but not all sentences have multiple aspect terms in these two datasets. Statistics of these datasets are displayed in Table 1, where (Train|Test|Total) Size and (Train|Test|Total) MM denote the number of instances and multi-aspect multi-sentiment instances on the training, testing, and overall (sum of training and testing) datasets, respectively. Each multi-aspect multi-sentiment instance contains multiple aspects and at least two aspects with different sentiment polarities. Additionally, we count the distribution of instances with the different number of aspects in the three datasets, which is shown in Figure 3. It can be seen that in the case of multiple aspects, instances with two or three aspects account for the majority. And, in MAMS-ATSA, there is no instance with only one aspect.

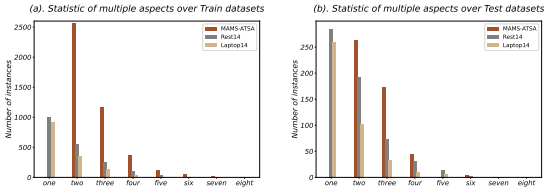

Figure 3: Statistics of multiple aspects over MAMS-ATSA, Rest14, and Laptop14 datasets. The number on the x-axis indicates the number of aspects in each instance in the dataset.

We utilize the last hidden states of the pre-trained BERT-base model for word representations (Devlin et al., 2019), the BERT containing 12 hidden layers, and 768 hidden dimensions for each layer. Moreover, the dimensions of the composed dependency tag embedding and type embedding are also initialized as 768. The hidden dimension of CHGMAN is 128. The dropout rate is 0.2. The number of the epoch is 30. We use Adam optimizer (Kingma and Ba, 2015) while training with the learning rate initialized by 0.00005. Our code will be released later.

4.2 Compared Methods

To evaluate our proposed model CHGMAN, we compare it with the following a series of baselines. **TD-LSTM** (Tang et al., 2016) utilizes LSTMs to model bidirectional contextual information. **ATAE-LSTM** (Wang et al., 2016) tends to combine learned attention embeddings with LSTM. **IAN**(Ma et al., 2017) is introduced to learn the coarse-grained attention for prediction. **MIAD** (Hazarika et al., 2018) simultaneously classifies

Model	MAMS-ATSA		Rest14		Laptop14	
	Accuracy	*Macro-F1*	*Accuracy*	*Macro-F1*	*Accuracy*	*Macro-F1*
TD-LSTM	74.60♭	-	78.00♮	66.73♮	71.83♮	68.43♮
ATAE-LSTM	77.05♭	-	77.20♯	-	68.70♯	-
IAN	76.60♭	-	78.60♯	-	72.10♯	-
RAM	-	-	80.23♯	70.80♯	74.49♯	71.35♯
GCAE	77.59♭	-	77.28♯	-	69.14♯	-
IARM	74.48*	73.66*	80.00	-	73.80	-
MIAD	-	-	79.00	-	72.50	-
ASGCN	44.06*	50.39*	80.77♯	72.02♯	75.55♯	71.05♯
CDT	76.77*	75.77*	82.30♯	74.02♯	77.19♯	72.99♯
Joint+PRET	-	-	81.96	71.80	73.04	69.16
STAGE	-	-	80.10	-	73.10	-
BERT	82.22♭	-	85.62♯	78.28♯	77.58♯	72.38♯
§SDGCN	-	-	83.57	76.47	81.35	**78.34**
§InterGCN	82.49*	81.95*	85.45*	77.64*	78.06*	73.83*
§CapsNet	83.39♭	-	85.93	-	-	-
§RGAT	83.16*	82.42*	86.25*	79.95*	78.21	74.07
§**CHGMAN (Ours)**	**85.05**	**84.29**	**86.88**	**81.62**	**81.52**	77.68

Table 2: Overall performance of different methods on MAMS-ATSA, Rest14, and Laptop14. The results indicated by an asterisk(*) are reproduced by running the released code of the published paper.The results with ♮, ♯ and ♭ are retrieved from (Liang et al., 2020), (Wang et al., 2020) and (Jiang et al., 2019) respectively. The other results except for our model are from the results reported by other baseline papers. The results with § are fine-tuned based on BERT(base).

all aspect terms in a sentence in pace with temporal dependency processing of corresponding sentences by utilizing LSTM. **RAM** (Chen et al., 2017) proposes to learn multi-hop attention on BiLSTM. **GCAE** (Xue and Li, 2018) proposes a convolution network combined with gating mechanisms to control sentiment flow. **IARM** (Majumder et al., 2018) utilizes aspect-aware sentence representation and memory network to fuse neighboring aspect information. **CapsNet** (Jiang et al., 2019) proposes a capsule network-based model to capture relationships between aspects and contextual words. **ASGCN** (Zhang et al., 2019) combines BiLSTM to capture contextual information regarding word orders with a multi-layered GCNs. **CDT** (Sun et al., 2019) encodes both dependency and contextual information by utilizing GCNs and BiLSTM. **BERT** (Devlin et al., 2019) fine-tunes BERT model to predict the sentiment polarity. **SDGCN** (Zhao et al., 2020) proposes a model based on GCNs to capture sentiment dependencies among aspects. **STAGE** (Ma et al., 2019) develops a two stage paradigm to model multi-aspects by using attention mechanism. **Joint+PRET** (Zhou et al., 2020) proposes a LSTM based model and converts sentiment classification to a sequence labeling problem to model relationships among aspects. **RGAT** (Wang et al., 2020) feeds reshaped syntactic dependency graph into RGAT to capture long dependency information.

4.3 Overall Performance

Table 2 shows the overall performance of our model and compared methods on three datasets, and the main evaluation matrices are Accuracy and Macro-averaged F1-score. The results demonstrate our model outperforms all compared methods except for the Macro-F1 of SDGCN on Laptop14. The performance of CHGMAN is affected since MM size in Laptop14 is small relatively (shown in Table 1). Our model exceeds all graph-based methods on evaluation matrices, indicating our proposed model is more effective than other graph-based models. Compared with RGAT, our model takes inter-aspect relationships into consideration. The performance gain towards RGAT indicates that inter-aspect relationships can help predict the sentiment polarity of the given aspect. Moreover, in comparison with InterGCN, CapsNet, and SDGCN, introducing attention mechanism, inter-aspect relationship, aspect-context relationship, and heterogeneity can help our model to distinguish more helpful information and enhance performance. All these results denote that exploiting inter-aspect relationships, aspect-context dependency relationships, and heterogeneity can improve performance.

4.4 Ablation Study

We conduct an ablation study to further analyze the effectiveness of our model. The result of the

Model	MAMS-ATSA		Rest14		Laptop14	
	Accuracy	*Macro-F1*	*Accuracy*	*Macro-F1*	*Accuracy*	*Macro-F1*
CHGMAN	**85.05**	**84.29**	**86.88**	**81.62**	**81.52**	**77.68**
CHGMAN w/o comp	84.37	83.89	86.61	80.45	80.25	76.29
CHGMAN w/o comp&edge	83.53	82.86	86.33	80.19	80.41	75.73
CHGMAN w/o comp&type	84.29	83.34	86.07	79.72	80.09	76.38
CHGMAN w/o hetero	83.69	83.29	86.16	79.03	80.09	76.17

Table 3: Results of ablation study. CHGMAN w/o X denotes the performance of our method after eliminating the corresponding modules, and Y&Z represents deactivating modules Y and Z simultaneously. Additionally, comp, edge, type, and hetero indicate circular correlation operation, target-to-edge channel, target-to-type channel, and adopting homogeneous graphs instead of heterogeneous graphs constructed. These results are fine-tuned based on BERT(base).

ablation study is shown in Table 3. **CHGMAN w/o comp** denotes that CHGMAN removes the circular correlation operation in the composed aggregation process. The performance of **CHGMAN w/o comp** is worse than **CHGMAN**, which indicates conducting the circular correlation operation to combine node and its type features in the aggregation process is effective and conducive to performance improvement. Furthermore, on the basis of **CHGMAN w/o comp**, we removes the attention channel *target-to-edge* and *target-to-type* respectively, which are written as **CHGMAN w/o comp&edge** and **CHGMAN w/o comp&type**. The performance of **CHGMAN w/o comp&edge** and **CHGMAN w/o comp&type** are both inferior than **CHGMAN w/o comp**, demonstrating adding two additional channel *target-to-edge* and *target-to-type* to *target-to-neighbor* is helpful and beneficial to enhancing the handling ability of heterogeneous information. The node types consist of contexts and aspect terms, the performance difference between **CHGMAN w/o comp** and **CHGMAN w/o comp&type** also shows the significance of identifying the information of other aspect terms and contexts in the attention calculation phase, which is conducive to the utilization of both information concurrently. In the meantime, edge features are also essential in the attention calculation phase, *target-to-edge* channel can facilitate the usage of inter-aspect relationships and aspect-context relationships by putting into consideration the features on the edge. The performance of Laptop14 is not clear enough in contrast to the other datasets since the MM size in Laptop14 is less than the other two, as seen in Table 1. **CHGMAN w/o hetero** indicates that we eliminate the impact of heterogeneous graphs constructed and substitute them with homogeneous graphs with edge features. That is to say, we disregard inter-aspect relationships and construct a homogeneous graph for each aspect term, with just one type of node, but the architecture of CHGMAN stays unaffected. The performance of **CHGMAN w/o hetero** has declined in comparison to **CHGMAN**, revealing the heterogeneous graph our constructed is favorable to increasing the efficacy of CHGMAN for MAMSC.

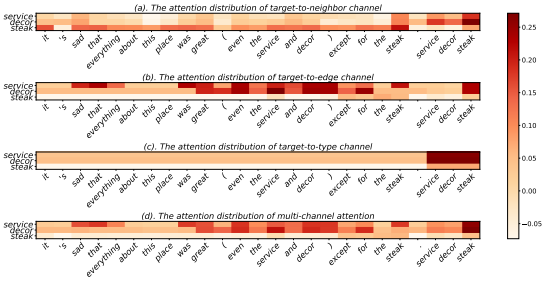

Figure 4: The attention distribution visualization of a specific testing sample.

4.5 Attention Distribution Exploration

To qualitatively illustrate how CHGMAN improves the performance in MAMSC. We post the attention distribution visualization of a testing sample, which is demonstrated in Figure 4. The input sentence is *It's sad that everything about this place was great (even the service and decor) except for the steak.*, there exist three aspect terms: *service*, *decor* and *steak*, and sentiment polarity labels of these three aspect terms are POSITIVE, POSITIVE and NEGATIVE respectively. Figure 4(a) is the attention distribution of *target-to-neighbor*, which shows that for aspect term *steak*, *target-to-neighbor* pays more attention to the context word *sad*, and it is helpful to judge the sentiment polarity to a certain degree. Whereas for *service* and *decor*, *target-to-neighbor* also notices context words *sad* and *great*, although it is ambiguous. Thus for the sake of introducing aspect term nodes, *target-to-*

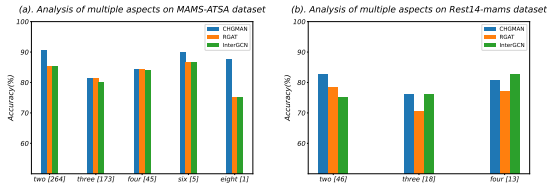

Figure 5: The comparison results of different aspect numbers on MAMS-ATSA and Rest14. The notation $M\,[N]$ on the horizontal axis represent the number of aspects (M) and the sample size of such a category (N). These results are fine-tuned based on BERT(base).

neighbor pays more attention to *steak* to get more guidance from inter-aspect relationships. As a powerful supplement to *target-to-neighbor*, the attention distribution of *target-to-edge* is displayed in Figure 4(b). For *service* and *decor*, *target-to-edge* notices context words *great* and *even*, and aspect term service is misled by *sad*, thus both of these aspect terms pay attention to aspect term *steak* to seek inter-aspect relationship guidance. In addition, for the *steak*, *target-to-edge* pays attention to *except for the steak*, which is helpful to indicate the NEGATIVE polarity what *steak* conveys. Figure 4(c) shows the attention distribution of *target-to-type*. In this case, for node type, our model tends to focus more on aspect term node to seek more help from inter-aspect relationships, which is reasonable. The attention distribution of overall disentangled attention is shown in Figure 4(d), which is a combination of these three attention terms. We can recognize our model focus on those valuable context words such as *except* when it tends to judge the sentiment polarity of *steak*. For *service* and *decor* that are not easy to distinguish, our model can combine the information both from context words and other aspect terms to make the right judgment.

4.6 Analysis of Multiple Aspects

To further analyze the performance of our model in the multiple aspects situation, we separate each test dataset into different subsets according to the number of aspects in each sentence. Figure 5 demonstrates the test accuracy of CHGMAN and the comparison models RGAT and InterGCN for different subsets on MAMS-ATSA and Rest14 (For Rest14, we only consider sentences containing multiple aspects and at least two aspect terms with different sentiment polarities.). Figure 5(a) shows the analysis of multiple aspects on MAMS-ATSA, and Figure 5(b) shows Rest14. The horizontal axis represents the number of aspects, and the number in square brackets is the sample size of such a category in the dataset. The vertical axis represents the accuracy score. We can observe that CHGMAN is superior to RGAT and InterGCN on most subsets. In the case of four aspects, InterGCN slightly outperforms CHGMAN since Rest14 has fewer multi-aspect multi-sentiment instances (see MM size in Table 1), making it hard to fully demonstrate the effect of the model. Overall, results indicate our model is adept at capturing inter-aspect relationship information when there exist multiple aspects.

5 Conclusion

In this paper, we construct a heterogeneous graph for MAMSC and propose a novel CHGMAN to tackle the heterogeneous graph. Our model predicts sentiment polarities by incorporating inter-aspect relationships, aspect-context relationships, and node heterogeneity. Moreover, our experiments prove the effectiveness of our method.

Acknowledgements

This work is funded in part by the National Natural Science Foundation of China Project (No.U1936213), and the Major Key Project of PCL (PCL2021A06).

References

Razvan C. Bunescu and Raymond J. Mooney. 2005. A shortest path dependency kernel for relation extraction. In *HLT/EMNLP*, pages 724–731. The Association for Computational Linguistics.

Peng Chen, Zhongqian Sun, Lidong Bing, and Wei Yang. 2017. Recurrent attention network on memory for aspect sentiment analysis. In *EMNLP*, pages 452–461. Association for Computational Linguistics.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: pre-training of deep bidirectional transformers for language understanding. In *NAACL-HLT (1)*, pages 4171–4186. Association for Computational Linguistics.

Timothy Dozat and Christopher D. Manning. 2017. Deep biaffine attention for neural dependency parsing. In *ICLR (Poster)*. OpenReview.net.

Devamanyu Hazarika, Soujanya Poria, Prateek Vij, Gangeshwar Krishnamurthy, Erik Cambria, and Roger Zimmermann. 2018. Modeling inter-aspect dependencies for aspect-based sentiment analysis. In *NAACL-HLT (2)*, pages 266–270. Association for Computational Linguistics.

Binxuan Huang and Kathleen M. Carley. 2019. Syntax-aware aspect level sentiment classification with graph attention networks. In *EMNLP/IJCNLP (1)*, pages 5468–5476. Association for Computational Linguistics.

Qingnan Jiang, Lei Chen, Ruifeng Xu, Xiang Ao, and Min Yang. 2019. A challenge dataset and effective models for aspect-based sentiment analysis. In *EMNLP/IJCNLP (1)*, pages 6279–6284. Association for Computational Linguistics.

Diederik P. Kingma and Jimmy Ba. 2015. Adam: A method for stochastic optimization. In *ICLR (Poster)*.

Bin Liang, Rongdi Yin, Lin Gui, Jiachen Du, and Ruifeng Xu. 2020. Jointly learning aspect-focused and inter-aspect relations with graph convolutional networks for aspect sentiment analysis. In *COLING*, pages 150–161. International Committee on Computational Linguistics.

Dehong Ma, Sujian Li, Xiaodong Zhang, and Houfeng Wang. 2017. Interactive attention networks for aspect-level sentiment classification. In *IJCAI*, pages 4068–4074. ijcai.org.

Xiao Ma, Jiangfeng Zeng, Limei Peng, Giancarlo Fortino, and Yin Zhang. 2019. Modeling multi-aspects within one opinionated sentence simultaneously for aspect-level sentiment analysis. *Future Gener. Comput. Syst.*, 93:304–311.

Navonil Majumder, Soujanya Poria, Alexander F. Gelbukh, Md. Shad Akhtar, Erik Cambria, and Asif Ekbal. 2018. IARM: inter-aspect relation modeling with memory networks in aspect-based sentiment analysis. In *EMNLP*, pages 3402–3411. Association for Computational Linguistics.

Maximilian Nickel, Lorenzo Rosasco, and Tomaso A. Poggio. 2016. Holographic embeddings of knowledge graphs. In *AAAI*, pages 1955–1961. AAAI Press.

Maria Pontiki, Dimitris Galanis, John Pavlopoulos, Harris Papageorgiou, Ion Androutsopoulos, and Suresh Manandhar. 2014. Semeval-2014 task 4: Aspect based sentiment analysis. In *SemEval@COLING*, pages 27–35. The Association for Computer Linguistics.

Kai Sun, Richong Zhang, Samuel Mensah, Yongyi Mao, and Xudong Liu. 2019. Aspect-level sentiment analysis via convolution over dependency tree. In *EMNLP/IJCNLP (1)*, pages 5678–5687. Association for Computational Linguistics.

Duyu Tang, Bing Qin, Xiaocheng Feng, and Ting Liu. 2016. Effective lstms for target-dependent sentiment classification. In *COLING*, pages 3298–3307. ACL.

Shikhar Vashishth, Soumya Sanyal, Vikram Nitin, and Partha P. Talukdar. 2020. Composition-based multi-relational graph convolutional networks. In *ICLR*. OpenReview.net.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *NIPS*, pages 5998–6008.

Kai Wang, Weizhou Shen, Yunyi Yang, Xiaojun Quan, and Rui Wang. 2020. Relational graph attention network for aspect-based sentiment analysis. In *ACL*, pages 3229–3238. Association for Computational Linguistics.

Yequan Wang, Minlie Huang, Xiaoyan Zhu, and Li Zhao. 2016. Attention-based LSTM for aspect-level sentiment classification. In *EMNLP*, pages 606–615. The Association for Computational Linguistics.

Wei Xue and Tao Li. 2018. Aspect based sentiment analysis with gated convolutional networks. In *ACL (1)*, pages 2514–2523. Association for Computational Linguistics.

Chen Zhang, Qiuchi Li, and Dawei Song. 2019. Aspect-based sentiment classification with aspect-specific graph convolutional networks. In *EMNLP/IJCNLP (1)*, pages 4567–4577. Association for Computational Linguistics.

Pinlong Zhao, Linlin Hou, and Ou Wu. 2020. Modeling sentiment dependencies with graph convolutional networks for aspect-level sentiment classification. *Knowl. Based Syst.*, 193:105443.

Jie Zhou, Jimmy Xiangji Huang, Qinmin Vivian Hu, and Liang He. 2020. Modeling multi-aspect relationship with joint learning for aspect-level sentiment classification. In *DASFAA (1)*, volume 12112 of *Lecture Notes in Computer Science*, pages 786–802. Springer.

CoNTACT: A Dutch COVID-19 Adapted BERT for Vaccine Hesitancy and Argumentation Detection

Jens Lemmens and **Jens Van Nooten** and **Tim Kreutz** and **Walter Daelemans**
CLiPS (University of Antwerp)
Lange Winkelstraat 40,
2000 Antwerp (Belgium)

Abstract

We present CoNTACT[1]: a Dutch language model adapted to the domain of COVID-19 tweets. The model was developed by continuing the pre-training phase of RobBERT (Delobelle et al., 2020) by using 2.8M Dutch COVID-19 related tweets posted in 2021. In order to test the performance of the model and compare it to RobBERT, the two models were tested on two tasks: (1) binary vaccine hesitancy detection and (2) detection of arguments for vaccine hesitancy. For both tasks, not only Twitter but also Facebook data was used to show cross-genre performance. In our experiments, CoNTACT showed statistically significant gains over RobBERT in all experiments for task 1. For task 2, we observed substantial improvements in virtually all classes in all experiments. An error analysis indicated that the domain adaptation yielded better representations of domain-specific terminology, causing CoNTACT to make more accurate classification decisions.

1 Introduction

Since the development of COVID-19 vaccines, an us-against-them mentality has emerged between advocates and adversaries of vaccines. Social media functions as a catalyst in this polarization, because it enables a rapid spread of unsolicited opinions, ranging from nuanced to radical, and well-reasoned to emotional. The turbulent vaccine debate that has been hosted by social media will unquestionably continue to influence future views on vaccination, whether it regards new COVID-19 boosters or other vaccines. In order to help process the large amounts of online content related to vaccines and COVID-19 in general, we present CoNTACT (Contextual Neural Transformer Adapted to COVID-19 Tweets). CoNTACT was developed by fine-tuning RobBERT (Delobelle et al., 2020) (a RoBERTa-base model (Liu et al., 2019) pre-trained on Dutch data) on masked language modeling using 2.8M Dutch-language tweets related to COVID-19 that were posted in 2021. The model was evaluated on two tasks: (1) binary vaccine hesitancy classification and (2) classification of arguments for vaccine hesitancy. In order to measure the effect of the domain adaptation, the results were compared to out-of-the-box RobBERT. Moreover, the aforementioned tasks were not only performed on tweets, but also on Facebook comments to show the cross-genre benefits of the domain adaptation. Afterwards, a qualitative error analysis was conducted to show where CoNTACT improved compared to RobBERT and where it could potentially improve further. In this analysis, special attention was paid to what domain-specific language the model learned compared to out-of-the-box RobBERT.

In earlier research, an English language model pre-trained on COVID-19 related tweets (COVID-Twitter-BERT) was developed (Müller et al., 2020). We apply the same methodology for the first time to Dutch and extensively test the effect on two COVID-19 related classification tasks.

2 Related research

Traditional machine learning assumes that models are trained and tested on large amounts of data from the same domain, which is not always feasible due to lack of labelled data. Transfer learning, which involves the transfer of knowledge from one domain to another, is a technique that has been utilized frequently in machine learning, both in NLP (e.g. Weiss et al. (2016), Durrani et al. (2021), Rostami and Galstyan (2021)) and computer vision (e.g. Wang and Deng (2018), Voulodimos et al. (2018), Xu et al. (2019)) to combat this issue. An effective approach to transfer learning that has dominated in various NLP tasks in recent years is the pre-training of language models, such as BERT (Devlin

[1]The model is available at https://huggingface.co/clips/contact

Proceedings of the 29th International Conference on Computational Linguistics, pages 6837–6845
October 12–17, 2022.

et al., 2018), on large amounts of unsupervised data. The knowledge from this pre-training phase is then transferred to the subsequent fine-tuning phase on task- and domain-specific data, which has shown significant improvements on several benchmark datasets, e.g., Leong et al. (2020) and Basile et al. (2019), even when using relatively small amounts of labeled data. Subsequently, several language- and domain-specific adaptations of language models have been developed for non-English data or to further improve the performance of the original models on specific tasks. Examples are BERTje and RobBERT (de Vries et al., 2019; Delobelle et al., 2020), the Dutch equivalents of BERT and RoBERTa, respectively), and CamemBERT (Martin et al., 2019), a French BERT model.

Domain adaptation, a special case of transfer learning where the model is first trained on large amounts of unsupervised data from the domain of an intended task, aims to improve results even further than traditional transfer learning "by minimizing the difference between the domain distributions" (Farahani et al. (2020), p. 1), thus creating a model that optimally learns from the training data. Regarding the domain of COVID-19, COVID-Twitter-BERT (Müller et al., 2020), which is a BERT-large model pre-trained on COVID-19 related tweets, has shown statistically significant gains over the baseline BERT-large model in various applications, including vaccine stance classification.

In other research related to vaccine stance classification, the effectiveness of various rule-based, statistical and deep learning approaches on the classification of stance towards vaccines have been compared (Joshi et al., 2018). Concretely, the task consisted of multiclass classification of vaccine stance in social media messages ("for", "against" or "undecided"). The authors concluded that both pre-trained language models and statistical ensemble models achieved equally high results on this task. This work focused on vaccine stance in general, but since the start of the COVID-19 pandemic vaccine stance classification has become almost inextricably linked to COVID-19 due to its societal relevance. An example is Weinzierl and Harabagiu (2021) who present CoVaxLies, a COVID-19 vaccine misinformation dataset, and demonstrate that their knowledge-graph-based model outperforms widely used classification methods for the detection of vaccine misinformation, an important cause of vaccine hesitancy.

Specifically for Dutch, Wang et al. (2020) collected Dutch tweets using keywords, and comments from Reddit and Nu.nl[2] threads related to COVID-19 in order to investigate polarity ("positive"/"negative") and stance ("support"/"reject"/"other") towards face masks and the social distance measure between March and October 2020. For polarity analysis, the Pattern library (De Smedt and Daelemans, 2012) was used, whereas manual annotations were used to train a stance classifier consisting of a linear feed forward neural network using stochastic gradient descend and a subword embedding layer, which achieved a test set accuracy of 65%. After applying the polarity analyzer and stance classifier to the above-mentioned data, it was shown that a more negative polarity was found in COVID-19 related messages than in a subset of messages that were unrelated to COVID-19. More specifically, a more negative polarity (and also stance) was found in messages mentioning face masks than in messages mentioning the social distancing measure. The various social media platforms that were used showed similar trends over time.

3 Methodology

3.1 Domain adaptation

For the development of CoNTACT, we utilize RobBERT[3] (Delobelle et al., 2020), a Dutch RoBERTa model with 12 attention layers and 12 heads with 117M parameters. RobBERT was chosen over BERTje (Delobelle et al., 2020), the Dutch equivalent of BERT, since BERTje was trained primarily on news data, whereas RobBERT was trained on the Dutch segment of the OSCAR corpus (Ortiz Suárez et al., 2019), which contains not only news data but online data in general (6.6 billion words).

In line with Han and Eisenstein (2019), we approached adapting RobBERT by continuing its pre-training phase, that is by performing masked language modeling. For this task, we scraped Dutch-language tweets posted in 2021 using the Twitter API and the keyword method described in Kreutz and Daelemans (2019). Then, all tweets related to COVID-19 were filtered from this Twitter collection using regular expressions based on inflected

[2]Nu.nl is a Dutch news website that allows visitors to comment on news articles

[3]https://huggingface.co/pdelobelle/robbert-v2-dutch-base

Table 1: Keyword lemmas used to construct regular expressions for collecting COVID-19 related tweets (English translations between brackets if applicable).

Key words
corona, COVID-19, SARS-CoV-2, viroloog (virologist), virus
vaccin (vaccine), vaccineren (to vaccinate), Astrazeneca, Pfizer, Moderna, Johnson & Johnson, Curevac, Sputnik
mondmasker (mouth mask), social distancing, bubbel (bubble), contact tracing, quarantaine (quarantine), lockdown, avondklok (curfew), 1.5m, knuffelcontact (cuddle contact)

forms, part-of-speech tag variations and spelling variations of the keywords shown in Table 1.

Afterwards, all duplicates and retweets were filtered from this subset of COVID-19 related tweets. To detect retweets, we based ourselves on the "retweet status" attribute returned by the Twitter API and searched for tweets beginning with "RT @". Finally, the FastText language detector was used to remove all tweets that were not written in Dutch (Joulin et al., 2016). In the end, 2.8M tweets (66.8M tokens, split by whitespace) remained for the domain adaptation, which were anonymized by replacing all tokens starting with "@" by "@USER". In order to estimate the precision, 300 randomly selected tweets were manually read and it was determined whether they were Dutch and relevant to the domain of COVID-19. This manual evaluation shows that our keyword extraction method has a precision of 90.0%. False positives included messages about other viruses and vaccines, such as the flu/influenza, and a single tweet in Afrikaans that did not get detected by the language detector.

For the domain adaptation, the 2.8M tweets mentioned above were used to continue RobBERT's pre-training phase for 4 epochs, using the default learning rate and the largest batch size that fit working memory (32). A cross entropy loss of 1.702 was achieved on a validation set consisting of 20% of our data (as a comparison, the validation loss of COVID-Twitter-BERT was approximately 1.5 for the masked language modeling task (Müller et al., 2020), whereas the validation loss reported in the original RobBERT paper amounted to 0.172 (Delobelle et al., 2020)).

3.2 Data and experiments

To determine the effect of the domain adaptation, i.e., whether CoNTACT performs significantly better than RobBERT on tasks involving social media data related to COVID-19, the models were tested on two classification tasks: (1) vaccine hesitancy detection and (2) the detection of arguments for vaccine hesitancy. The corpus used for the classification tasks was first described in Lemmens et al. (2021), it consists of approx. 8,800 tweets and 5,200 Facebook comments annotated for vaccine stance and argumentation. Regarding vaccine stance, the possible class labels originally were "anti-vaccination", "vaccine-hesitant", "neutral" and "pro-vaccination", but these were converted to binary labels: "anti-vaccination" and "vaccine-hesitant" comprise the "hesitant" category, whereas the "not hesitant" category consists of all "neutral" and "pro" comments. The annotation scheme for vaccine hesitancy arguments on the other hand consisted of the following labels:

1. **Development**: messages that express worry about the development, testing methodology, distribution and public access of vaccines.
2. **Liberty**: messages that express concerns about how vaccines and vaccine laws affect civil liberty and personal freedom.
3. **Institutional motives**: messages expressing mistrust in motives of political or economic entities involved with vaccines.
4. **Efficacy**: messages claiming that vaccines are not efficient (enough) or unnecessary.
5. **Safety**: messages that express worry towards the safety of the vaccines and their side effects.
6. **Criticism on the vaccination strategy**: messages criticizing the government's vaccination strategy/campaign.
7. **Alternative medicine**: messages that prefer other means of protection over vaccines.
8. **Conspiracy theories**: messages that spread well-established or new conspiracy theories about vaccines.

In order to test the performance of CoNTACT on the vaccine hesitancy detection task, both RobBERT and CoNTACT were fine-tuned with 10-fold cross validation. These cross validation experiments were performed in same-genre settings (fine-tuning and testing on tweets only; fine-tuning and

Table 2: Statistics of the vaccine hesitancy data used for the cross validation experiments.

Class	Twitter	Facebook	Total
hesitant	1250	1250	2500
non-hesitant	1250	1250	2500
Total	2500	2500	5000

testing on Facebook comments only) and mixed-genre settings (fine-tuning and testing on both Facebook and Twitter). Additionally, cross-genre experiments were conducted by fine-tuning on all Twitter data and testing on all Facebook data (and vice versa) in order to show the usefulness of CoNTACT when no data from an intended platform is available for fine-tuning. In order to avoid overfitting on a certain class or platform due to unbalanced data, a subset that was balanced by class and social media platform was used. The statistics of this subset can be found in Table 2. For all experiments, the default batch size (8) and learning rate (5e-5) was used and fine-tuning was performed for 4 epochs.

Regarding the argumentation detection task, 8,439 tweets and 3,917 Facebook comments were used (i.e. all of the available vaccine-hesitant messages). The distribution of the arguments varies across the two social media platforms, as can be derived from Table 3. Further, it should be noted that vaccine-hesitant entries without any clear argumentation were used as negative examples for the models to learn from. Similarly to the stance detection task, the aforementioned data was used to fine-tune both RobBERT and our CoNTACT model. For the same- and mixed genre experiments, cross validation was used, whereas a train-test split was used for the cross-genre experiments. Since the data is heavily unbalanced in terms of argument distribution, however, we chose to conduct experiments with 5-fold instead of 10-fold cross validation in order to preserve more entries per test set. For all experiments, the default batch size (8) and learning rate (5e-5) were used and fine-tuning was performed for 4 epochs.

4 Results

4.1 Vaccine hesitancy detection

In Table 4, the results of the experiments for vaccine hesitancy detection can be found. For the same-genre and mixed-genre experiments, the provided results (precision, recall, F1-score) are the averages of the test set scores on the positive class (i.e. vaccine hesitancy) in each cross validation split

Table 3: Statistics of the vaccine hesitancy arguments data used for the cross validation experiments.

Class	Tw	Fb	Total
alternative medicine	175	56	175
conspiracy theory	687	228	915
criticism vaccination strategy	979	1,222	2,201
development	565	511	1,076
efficacy	860	400	1,260
institutional motives	1,189	312	2,131
liberty	3,390	450	3,840
safety	1,493	1,416	2,909
none	1,153	298	1,451
n messages	8,439	3,917	12,356

(the standard deviations are mentioned between brackets). For the cross-genre experiments, on the other hand, results are reported on the test sets. In cases where CoNTACT outperformed RobBERT, p-values were calculated to determine whether the observed improvements are statistically significant (McNemar, 1947).

As shown in the results, both models perform better on Twitter data than on Facebook data, and fine-tuning on both platforms simultaneously yields higher results than fine-tuning on the individual platforms. The standard deviations are, in spite of the small test sets, relatively small, which indicates consistent model performance. When comparing the results of RobBERT to those of CoNTACT, it can be observed that CoNTACT outperforms RobBERT in all experimental settings with statistical significance, including the cross-genre experiments. In other words, when fine-tuning on Twitter but testing on Facebook, CoNTACT strongly outperforms RobBERT, although no Facebook data was used during its domain adaptation or fine-tuning phase. Additionally, CoNTACT outperforms RobBERT on Facebook data even if the former is fine-tuned on Twitter data and the latter is fine-tuned on Facebook data (i.e. data from the same platform). These results highlight the cross-genre potential of CoNTACT.

In order to gain insight into which specific improvements CoNTACT made, a manual analysis[4] of the instances where CoNTACT classified vaccine stance correctly, and RobBERT did not, was conducted (for all experiments). False negatives, i.e., the cases where RobBERT did not predict vaccine hesitancy, but CoNTACT did (correctly), were the largest group of errors. They were found in vaccine-hesitant instances referring to pro-

[4] All examples provided below were translated from Dutch to English.

Table 4: Results (%) for vaccine hesitancy detection, including standard deviations (if applicable). The results are reported on the positive class, and statistically significant gains over the baseline are indicated with asterisks.

Model	Fine-tune	Test	Pre	Rec	F1	*
RobBERT	Twitter	Twitter	76.1 (3.6)	74.2 (4.3)	75.1 (3.1)	N/A
	Twitter	Facebook	62.0 (-)	59.8 (-)	60.9 (-)	N/A
	Facebook	Facebook	69.5 (3.1)	57.2 (3.2)	62.7 (2.6)	N/A
	Facebook	Twitter	67.4 (-)	63.0 (-)	65.1 (-)	N/A
	Both	Twitter	77.1 (2.8)	73.9 (4.0)	75.4 (-)	N/A
	Both	Facebook	70.6 (3.5)	64.6 (3.7)	67.4 (2.7)	N/A
CoNTACT	Twitter	Twitter	77.2 (3.5)	76.9 (4.1)	77.1 (3.6)	*
	Twitter	Facebook	65.2 (-)	64.9 (-)	65.0 (-)	***
	Facebook	Facebook	71.2 (3.2)	67.5 (3.1)	69.3 (2.9)	***
	Facebook	Twitter	71.0 (-)	82.5 (-)	76.3 (-)	***
	Both	Twitter	78.9 (4.2)	77.4 (1.7)	78.1 (2.5)	**
	Both	Facebook	73.2 (3.0)	68.2 (4.3)	70.6 (2.6)	**

Table 5: Results (%, including EMR) for argument detection, including standard deviations (if applicable). The results are reported on the positive class.

Model	Fine-tune	Test	Pre	Rec	F1	EMR
RobBERT	Twitter	Twitter	62.5 (0.8)	50.2 (1.4)	55.0 (1.0)	46.7 (6.1)
	Twitter	Facebook	50.7 (-)	29.7 (-)	36.3 (-)	24.2 (-)
	Facebook	Facebook	48.4 (1.4)	31.7 (1.8)	37.3 (1.7)	34.9 (1.2)
	Facebook	Twitter	59.5 (-)	30.0 (-)	33.3 (-)	33.8 (-)
	Both	Twitter	62.9 (1.5)	53.4 (0.5)	57.3 (0.8)	47.7 (0.8)
	Both	Facebook	56.6 (2.8)	43.9 (3.1)	48.9 (2.9)	39.3 (1.6)
CoNTACT	Twitter	Twitter	64.7 (1.5)	56.2 (0.9)	59.8 (0.9)	49.2 (1.3)
	Twitter	Facebook	56.9 (-)	36.1 (-)	42.7 (-)	26.9 (-)
	Facebook	Facebook	55.5 (5.9)	41.1 (1.2)	46.2 (1.9)	41.0 (1.1)
	Facebook	Twitter	57.5 (-)	39.4 (-)	41.4 (-)	34.5 (-)
	Both	Twitter	64.1 (1.3)	58.4 (1.6)	60.9 (0.9)	49.5 (1.1)
	Both	Facebook	60.1 (3.3)	49.7 (2.2)	53.9 (2.4)	41.9 (1.1)

vaccination opinions, such as "'We do not have evidence that vaccines cause damage to pregnant women so we advise pregnant women to get vaccinated', what kind of an idiot says things like this?!". Further, false negatives were caused by sarcasm and other forms of implicit language, e.g. "they should start [the vaccination campaign] in The Hague... double dosis". This message seems to express pro-vaccination opinions on a superficial level, but the author actually hopes that the government (located in The Hague) will suffer from major side effects of the vaccine.

In comparison, false positives, i.e., cases where RobBERT incorrectly detected vaccine hesitancy, but CoNTACT correctly did not detect vaccine hesitancy, were found in messages containing certain hashtags or terms that are associated with vaccine hesitancy. For example, the tweet "#vaccinationobligation, because infecting others is not a fundamental right", expresses a pro-vaccination opinion. RobBERT, however, incorrectly detected vaccine hesitancy in this tweet, presumably because of the hashtag "#vaccinationobligation", which occurs frequently in vaccine-hesitant messages. Especially in the cross-genre experiments where the models were fine-tuned on Facebook and tested on Twitter, RobBERT was frequently confused by vaccine related hashtags, causing both false positives and negatives, whereas CoNTACT showed more understanding of said hashtags, even when both pro- and anti-vaccination hashtags appeared in the same message. Other false positives by the baseline were found in cases where vaccine-hesitant opinions were quoted or referred to, such as "'poison vaccine', yeah right, you're so childish". Similarly, pro-vaccination messages expressing a negative sentiment towards, for example, vaccination policy, were misclassified more often by RobBERT than by CoNTACT, e.g. "I am #provaccination but I support protest against the mismanagement of the government".

An additional analysis of the comments where CoNTACT failed to correctly predict the stance but RobBERT did not was conducted. False negatives (the smallest group of errors) were found in messages using implicit or sarcastic language, such as "this press conference was very clear as always...", similarly to the false negatives found in RobBERT. Regarding the false positives, the largest group of errors, we observed that there were cases where

specific terms used frequently in vaccine-hesitant messages caused confusion, as was also observed in the error analysis of RobBERT. For example, in “those #SideEffects are not as bad as people think" and “#vaccineobligation is a must", CoNTACT interprets the hashtags as indicators for vaccine hesitancy, because it has learned this during the fine-tuning period. In conclusion, we observed that the models have difficulties with the same types of comments: messages containing forms of implicit language caused false negative errors, whereas domain-specific terminology caused false positive errors. CoNTACT, however, made significantly less errors in these challenging cases due to the domain adaptation, indicating that CoNTACT has improved representations of COVID-19 related terminology.

4.2 Argument classification

In Table 5 the results on the argument classification task are summarised. Precision, recall, F1 (incl. standard deviations), and exact match ratio (EMR), a multilabel accuracy score for cases where the entire set of labels was predicted correctly, are reported. Overall, both models perform better on Twitter than on Facebook data, including the cross-genre experiments, similarly to the stance classification experiments. Although CoNTACT outperforms RobBERT in the cross-genre experiments, the results are still noticeably lower than the in-genre experiments. Further, fine-tuning on both Facebook and Twitter simultaneously increases model performance. When comparing the models, it can be observed that CoNTACT outperforms RobBERT in all experiments.

The results for the individual arguments for RobBERT and CoNTACT are presented in Table 6 and 7, respectively. The provided results are the results on the positive classes in the test set(s). Regarding the baseline results, it can be observed that certain classes are predicted substantially better than others. Overall, RobBERT predicted the “safety" and “liberty" classes best, whereas the most difficult classes were “development" and “alternative medicine" (these were also the most underrepresented classes in our data).

When comparing the results of RobBERT to those of CoNTACT, an increase in performance on all classes in all experiments can be observed, except for the “conspiracy theory" class in Twitter when fine-tuning on both platforms, the “alternative medicine" class in Twitter when fine-tuning on Facebook, and the “institutional motives" class in Facebook when fine-tuning on Twitter. Some of the highest improvements were found in the “development" and “alternative medicine" classes, which are the most challenging classes, as mentioned above. In order to verify whether the observed improvements are significant, a McNemar (McNemar, 1947) test was conducted per argument class (Table 8). Despite the substantial gains, less than half of the improvements were considered statistically significant for the same- and mixed-genre experiments. We suspect that the significance test we used yielded higher p-values because the frequency of certain classes was too low to ascertain that improvements were significant rather than random. Further experiments with more data could therefore produce other results and new insights in the future. In the cross-genre experiments, however, CoNTACT showed statistically significant improvements on half of the argumentation classes when the model was fine-tuned on Twitter data and tested on Facebook data. Moreover, statistically significant improvements were observed for all classes when the model was fine-tuned on Facebook data and tested on Twitter data. These results highlight the cross-genre potential of CoNTACT.

In order to gain insight into the specific improvements of CoNTACT, a manual error analysis of the predictions of both models was conducted. First, instances where CoNTACT succeeded and RobBERT failed to predict the correct argument(s) were investigated. For each argument class, several terms seemed to guide the predictions of CoNTACT, because of the learned representations of said terms during both the domain adaptation and fine-tuning phase. For instance, comments containing words and hashtags such as “medical experiment" and “lab rat" were classified correctly by CoNTACT as related to “development", contrary to RobBERT, which made more false negative errors in this class. Similar observations were made for “institutional motives" (e.g. references to governments, political parties and politicians, such as #rutte3, #dv66 and #hugodejonge), “conspiracy theory" (e.g. references to gene therapy, such as “#geneticmodification"), “safety" (also references to gene therapy), and “liberty" (e.g. references to vaccine passports and obligation). Concerning "alternative medicine", no clear patterns or recurring terms were discovered in instances where CoNTACT correctly pre-

Table 6: Averaged results (precision, recall and F1 in %) of RobBERT on each argument class per experiment.

	tw-tw			tw-fb			fb-fb			fb-tw			both-tw			both-fb		
	P	**R**	**F**	**P**	**R**	**F**	**P**	**R**	**F**	**P**	**R**	**F**	**P**	**R**	**F**	**P**	**R**	**F**
alt.	60	35	45	33	36	35	0	0	0	100	6	11	66	46	54	45	32	38
con.	61	43	50	32	14	19	48	22	30	34	10	15	59	46	52	54	38	45
crit.	47	27	34	45	26	33	57	49	53	27	36	31	49	30	37	59	51	55
dev.	54	38	45	42	18	25	50	18	26	54	18	27	55	45	49	47	31	37
eff.	63	54	58	50	46	48	52	36	42	61	31	42	61	55	58	58	46	51
inst.	66	60	63	59	27	37	59	32	41	76	9	17	66	62	64	60	38	46
lib.	77	76	77	61	36	46	61	48	54	64	83	72	77	78	77	64	49	56
saf.	71	67	69	84	34	49	66	63	64	60	47	53	70	67	69	67	66	67
micro	69	60	64	58	30	39	59	44	50	57	45	50	68	62	65	61	51	56
macro	62	50	55	51	30	36	48	32	37	60	30	33	63	53	57	60	44	49
weighted	67	60	63	61	30	39	57	44	48	60	45	45	67	62	64	60	51	55
samples	56	53	53	32	28	29	50	45	46	50	42	44	58	55	55	52	49	49

Table 7: Averaged results (precision, recall and F1 in %) of CoNTACT on each argument class per experiment.

	tw-tw			tw-fb			fb-fb			fb-tw			both-tw			both-fb		
	P	**R**	**F**	**P**	**R**	**F**	**P**	**R**	**F**	**P**	**R**	**F**	**P**	**R**	**F**	**P**	**R**	**F**
alt.	67	48	56	42	46	44	67	4	7	50	3	5	64	56	60	55	46	51
con.	60	49	54	32	23	27	55	31	40	52	31	39	57	47	51	54	43	48
crit.	51	34	41	49	42	45	61	57	59	25	46	33	49	37	42	59	58	59
dev.	58	47	52	56	24	36	55	33	41	56	36	44	58	51	54	54	34	42
eff	64	62	63	61	57	59	61	50	55	69	50	58	65	65	65	59	53	56
inst.	68	63	66	61	22	32	57	38	46	78	10	18	68	63	66	62	42	50
lib.	78	77	78	67	38	49	66	50	57	72	81	76	78	78	78	65	51	57
saf.	72	69	71	87	37	52	70	67	69	58	57	58	72	71	72	70	70	70
micro	70	64	67	62	36	46	64	53	58	58	51	55	69	65	67	63	56	59
macro	65	56	60	57	36	43	61	41	47	58	39	41	64	58	61	60	50	54
weighted	69	64	66	66	36	46	63	53	57	63	51	51	69	65	67	63	56	59
samples	58	57	56	37	34	34	55	51	51	53	48	48	59	58	57	56	54	53

Table 8: Statistically significant improvements in the argumentation detection task of CoNTACT over RobBERT.

Experiment	Classes with significant improvements
Tw - Tw	efficacy (***)
Tw - Fb	conspiracy (*), criticism on vaccination strategy (***), institutional motives (***), liberty (***)
Fb - Fb	development (***), efficacy (***), institutional motives (***), liberty (***), safety (*)
Fb - Tw	alternative medicine (***), conspiracy (***), criticism on vaccination strategy (***), development (***), efficacy (***), institutional motives (***), liberty (***), safety (*)
Both - Tw	efficacy (**)
Both - Fb	criticism on vaccination strategy (***), development (*)

dicted the argument when RobBERT did not.

In addition, messages where RobBERT predicted the correct arguments but CoNTACT did not were investigated, although no clear error patterns were found in these cases. In general, however, both models seem to incorrectly classify arguments when the message itself lacks context or terminology related to the argument. For example, in the Facebook comment “they don’t want them [the vaccines] anywhere else", which was annotated with the “criticism on vaccination strategy" label, both models failed to predict any argument, since the reference to e.g., a potential surplus of vaccines is implicit in this case.

In conclusion, CoNTACT seems to have learned domain-specific language in the domain adaptation phase, which benefits the model for the argument detection task, as can be derived from the results. The error analysis, however, showed that the model still experiences difficulties with classifying text entries that lack context or explicit information about the relevant argument(s).

5 Conclusion

In this work we presented CoNTACT, a Dutch language model adapted to the domain of COVID-19

tweets. The model was developed by continuing the masked language modeling pre-training phase of RobBERT using 2.8M Dutch tweets related to COVID-19. In order to test the performance of CoNTACT, the model was tested on two classification tasks: detection of vaccine hesitancy and detection of arguments for vaccine hesitancy. These tasked were performed in various experimental settings, that is by fine-tuning and testing on social media messages from two different platforms: Twitter and Facebook. For the vaccine hesitancy detection task, CoNTACT outperformed RobBERT with statistical significance in all experiments, including cross-genre settings. With respect to the argument classification task, CoNTACT showed substantial gains in virtually all classes in all experiments, some of which with statistical significance. An error analysis showed that the domain adaptation resulted in better representations of COVID-19 related terminology, and therefore in better results. Issues remain in messages containing implicit/figurative language or messages lacking context. Future work may include the development of a second version of CoNTACT, where the model is fine-tuned on more data from various platforms (Twitter, Facebook, Reddit, etc.) for even more cross-genre robustness and using CoNTACT on other COVID-19 related tasks.

Acknowledgements

This research received funding from the Vaccine Confidence Fund[5] (grant agreement CS 20065) and FWO Infrastructure project CLARIAH[6] - I000921N.

References

Valerio Basile, Cristina Bosco, Elisabetta Fersini, Debora Nozza, Viviana Patti, Francisco Manuel Rangel Pardo, Paolo Rosso, and Manuela Sanguinetti. 2019. SemEval-2019 task 5: Multilingual detection of hate speech against immigrants and women in Twitter. In *Proceedings of the 13th International Workshop on Semantic Evaluation*, pages 54–63, Minneapolis (MN), USA. Association for Computational Linguistics.

Tom De Smedt and Walter Daelemans. 2012. Pattern for Python. *J. Mach. Learn. Res.*, 13(null):2063–2067.

Wietse de Vries, Andreas van Cranenburgh, Arianna Bisazza, Tommaso Caselli, Gertjan van Noord, and Malvina Nissim. 2019. Bertje: A Dutch BERT model. *CoRR*.

Pieter Delobelle, Thomas Winters, and Bettina Berendt. 2020. RobBERT: a Dutch RoBERTa-based Language Model. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 3255–3265, Online. Association for Computational Linguistics.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. BERT: Pre-training of deep bidirectional transformers for language understanding. *CoRR*.

Nadir Durrani, Hassan Sajjad, and Fahim Dalvi. 2021. How transfer learning impacts linguistic knowledge in deep NLP models? *CoRR*.

Abolfazl Farahani, Sahar Voghoei, Khaled Rasheed, and Hamid R. Arabnia. 2020. A brief review of domain adaptation. *CoRR*.

Xiaochuang Han and Jacob Eisenstein. 2019. Unsupervised domain adaptation of contextualized embeddings: A case study in early modern english. *CoRR*.

Aditya Joshi, Xiang Dai, Sarvnaz Karimi, Ross Sparks, Cécile Paris, and C Raina MacIntyre. 2018. Shot or not: Comparison of NLP approaches for vaccination behaviour detection. In *Proceedings of the 2018 EMNLP Workshop SMM4H: The 3rd Social Media Mining for Health Applications Workshop & Shared Task*, pages 43–47, Brussels, Belgium. Association for Computational Linguistics.

Armand Joulin, Edouard Grave, Piotr Bojanowski, Matthijs Douze, Hervé Jégou, and Tomás Mikolov. 2016. Fasttext.zip: Compressing text classification models. *CoRR*.

Tim Kreutz and Walter Daelemans. 2019. How to optimize your Twitter collection. *Computational Linguistics in the Netherlands Journal*, 9:55–66.

Jens Lemmens, Tess Dejaeghere, Tim Kreutz, Jens Van Nooten, Ilia Markov, and Walter Daelemans. 2021. Vaccinpraat: Monitoring vaccine skepticism in Dutch Twitter and Facebook comments. *Computational Linguistics in the Netherlands Journal*, 11:173–188.

Chee Wee (Ben) Leong, Beata Beigman Klebanov, Chris Hamill, Egon Stemle, Rutuja Ubale, and Xianyang Chen. 2020. A report on the 2020 VUA and TOEFL metaphor detection shared task. In *Proceedings of the Second Workshop on Figurative Language Processing*, pages 18–29, Online. Association for Computational Linguistics.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. RoBERTa: A robustly optimized BERT pretraining approach. *CoRR*.

[5] https://vaccineconfidencefund.org/

[6] https://www.clariah.nl/about-clariah

Louis Martin, Benjamin Müller, Pedro Javier Ortiz Suárez, Yoann Dupont, Laurent Romary, Éric Villemonte de la Clergerie, Djamé Seddah, and Benoît Sagot. 2019. CamemBERT: A tasty french language model. *CoRR*.

Quinn McNemar. 1947. Note on the sampling error of the difference between correlated proportions or percentages. *Psychometrika*, 12(2):153–157.

Martin Müller, Marcel Salathé, and Per Egil Kummervold. 2020. COVID-Twitter-BERT: A natural language processing model to analyse COVID-19 content on Twitter. *CoRR*.

Pedro Javier Ortiz Suárez, Benoît Sagot, and Laurent Romary. 2019. Asynchronous pipelines for processing huge corpora on medium to low resource infrastructures. Proceedings of the Workshop on Challenges in the Management of Large Corpora (CMLC-7) 2019. Cardiff, 22nd July 2019, pages 9 – 16, Mannheim. Leibniz-Institut für Deutsche Sprache.

Mohammad Rostami and Aram Galstyan. 2021. Domain adaptation for sentiment analysis using increased intraclass separation. *CoRR*.

Athanasios Voulodimos, Nikolaos Doulamis, Anastasios Doulamis, and Eftychios Protopapadakis. 2018. Deep learning for computer vision: A brief review. *Computational Intelligence and Neuroscience*, pages 1–13.

Mei Wang and Weihong Deng. 2018. Deep visual domain adaptation: A survey. *CoRR*.

Shihan Wang, Marijn Schraagen, Erik Tjong Kim Sang, and Mehdi Dastani. 2020. Public sentiment on governmental COVID-19 measures in Dutch social media. In *Proceedings of the 1st Workshop on NLP for COVID-19 (Part 2) at EMNLP 2020*, Online. Association for Computational Linguistics.

Maxwell Weinzierl and Sanda Harabagiu. 2021. Automatic detection of covid-19 vaccine misinformation with graph link prediction. *Journal of Biomedical Informatics*, 124:103955.

Karl Weiss, Taghi M Khoshgoftaar, and DingDing Wang. 2016. A survey of transfer learning. *Journal of Big data*, 3(1):9.

Jiaolong Xu, Liang Xiao, and Antonio M. López. 2019. Self-supervised domain adaptation for computer vision tasks. *CoRR*.

SSR: Utilizing Simplified Stance Reasoning Process for Robust Stance Detection

Jianhua Yuan[1], **Yanyan Zhao**[1], **Yanyue Lu**[1], **Bing Qin**[1 2]
[1]Harbin Institute of Technology, Harbin 150001, China
[2]Pengcheng Laboratory, Shenzhen 518066, China
{jhyuan, yyzhao, yylu, qinb}@ir.hit.edu.cn

Abstract

Dataset bias in stance detection tasks allows models to achieve superior performance without using targets (Kaushal et al., 2021). Most existing debiasing methods are task-agnostic, which fail to utilize task knowledge to better discriminate between genuine and bias features. Motivated by how humans tackle stance detection tasks, we propose to incorporate the stance reasoning process as task knowledge to assist in learning genuine features and reducing reliance on bias features. The full stance reasoning process usually involves identifying the span of the mentioned target and corresponding opinion expressions, such fine-grained annotations are hard and expensive to obtain. To alleviate this, we simplify the stance reasoning process to relax the granularity of annotations from token-level to sentence-level, where labels for sub-tasks can be easily inferred from existing resources. We further implement those sub-tasks by maximizing mutual information between the texts and the opinioned targets[1]. To evaluate whether stance detection models truly understand the task from various aspects, we collect and construct a series of new test sets. Our proposed model achieves better performance than previous task-agnostic debiasing methods on most of those new test sets while maintaining comparable performances to existing stance detection models.

1 Introduction

The task of stance detection aims to predict the stance of the text towards the given target. It is crucial for various downstream tasks including fact verification, rumor detection, etc. It has a wide application in analyzing political opinions and product reviews. Existing works usually treat this task as a text pair classification problem and many design target-tweet interaction structures (Augenstein et al., 2016) to learn target-aware stance representations. However, Kaushal et al. (2021) have shown that those models (Du et al., 2017; Devlin et al., 2019) can achieve superior performances only using the tweet. Those end-to-end stance detection models treat the stance reasoning process as a black box and are prone to rely on bias features in the dataset instead of learning the underlying task. For instance, in Figure 1, the BERT model still predicts *Against* even when the target is changed to an unrelated target like *Atheism*. Meanwhile, common stance detection models perform poorly on out-of-distribution datasets (Kaushal et al., 2021) and unseen targets, which calls for debiasing stance detection models to get rid of spurious correlations in the datasets.

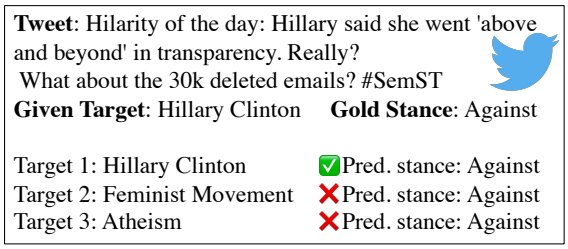

Figure 1: An example illustrating that BERT model does not change predictions based on the target.

While Kaushal et al. (2021) made the first attempt to reveal dataset bias in stance detection, mitigating bias in other natural language understanding (NLU) tasks has been extensively explored. The key challenge in debiasing is how to discriminate between genuine and bias features. One line of work (Clark et al., 2019; Utama et al., 2020a,b) implicitly hypothesized that features, learnt by small models or by large models at earlier steps, could potentially be bias features. In addition, others (Kaushik et al., 2021; Kaushal et al., 2021; Yang et al., 2021) tried data augmentation to break spurious correlation in the training data and treated features learnt on the augmented data as genuine features. Another line of work (Tu et al., 2020) adopted multi-task learning with auxiliary tasks,

[1]refers to targets that a given tweet expresses opinions on.

Proceedings of the 29th International Conference on Computational Linguistics, pages 6846–6858
October 12–17, 2022.

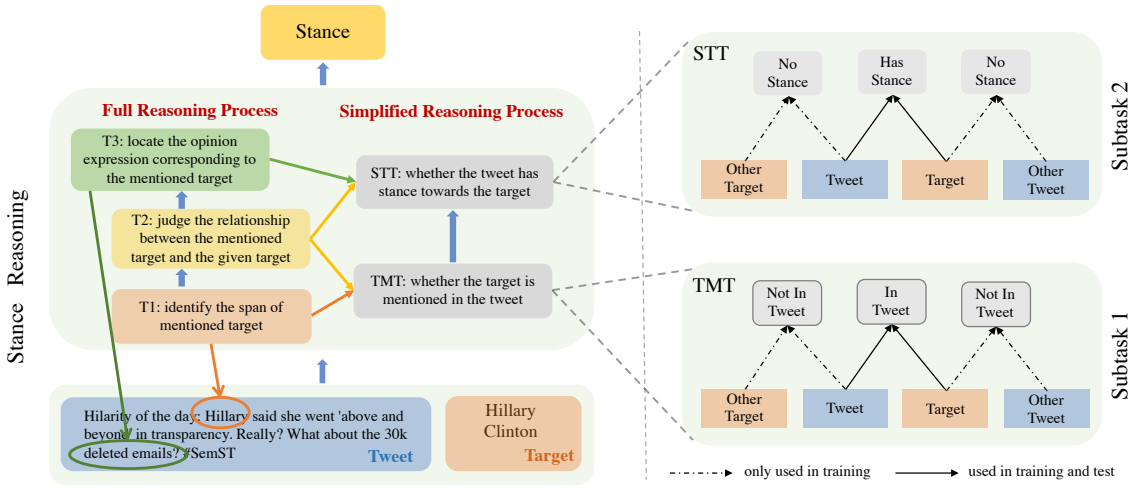

(a) Stance Reasoning Process (b) Simplified Subtasks

Figure 2: (a) An illustration of how humans perform stance reasoning on a given <target, tweet> pair and our simplified stance reasoning process. (b) We implement the simplified subtasks by maximizing the mutual information between the tweet and the opinioned target. During training, one positive example and two negative examples are constructed based on a given <target, tweet> pair for each subtask.

where features shared by multiple tasks are seen as genuine features. While these methods achieved superior debiasing performances, most of them neglected to explicitly leverage task knowledge to help discriminate between genuine and bias features.

In contrast, Dua et al. (2020) introduced manual annotations of intermediate reasoning steps and employed multi-task learning to combat dataset bias in question answering. Motivated by how humans perform stance reasoning processing in Figure 2, we follow this line of work and make the first attempt to incorporate stance reasoning process to mitigate dataset bias. Specifically, we consider the following reasoning steps: (**T1**) identifying the span of the mentioned target, (**T2**) judging the relationship between the mentioned target and the given target, and (**T3**) locating the opinion expression corresponding to the mentioned target. However, due to the informality of tweet texts, such fine-grained annotations for those reasoning steps are difficult and expensive to acquire.

To alleviate this, we seek to simplify the above reasoning process into two easier sub-tasks: 1) **TMT**, classifying whether the given target is mentioned in the tweet, as a simplification of **T1** and **T2**. 2) **STT**, determining whether the tweet expresses any stance towards the given target, as an easy version of **T2** and **T3**. The simplified subtasks only require sentence-level labels instead of token-level ones. More importantly, the labels for these two sub-tasks can be easily inferred *without additional annotations*. We enhance these two subtasks by maximizing mutual information between the tweets and opinioned targets.

To help thoroughly evaluate whether models understanding the stance detection task, we further collect and construct 6 new test sets. Those new test sets will assess whether stance detection models alter predictions based on the target, whether they overfit to shortcut features, whether they understand implicit mention of targets, and whether they can handle negations in the target part.

To summarize, our contributions are three folds:

- We make the first attempt to incorporate stance reasoning process to mitigate dataset bias in stance detection, where the labels for intermediate steps can be easily acquired without further annotations.
- We construct 6 test sets [2] to facilitate evaluation of whether stance detection models understand the task from various aspects.
- The proposed approach outperforms existing debiasing methods on 4/6 new test sets while maintaining comparable performances to common stance detection models on in-domain datasets.

[2]https://github.com/Surpriseshelf/StanceSSR

2 Approach

In this section, we present the **SSR** model that employs simplified stance reasoning process to combat dataset bias in stance detection. We first describe the basic text encoder that we use to encode tweet, target and target-tweet pairs. Then, we introduce two intermediate sub-tasks based on our observation of the stance reasoning process. After that, we elaborate on why and how we simplify the introduced two sub-tasks. These sub-tasks are implemented by maximizing the mutual information between the opinioned target and the tweet. Finally, we show how we combine these two sub-tasks for final stance predictions.

2.1 Text Encoder

We use BERT$_{base}$ as the text encoder. Given a text sequence $D = \{x_1, x_2, \ldots, x_i, \ldots, x_{|d|}\}$, where $|d|$ is the number of words in D, we transform its format to $X_d = \{[\text{CLS}], x_1, x_2, \ldots, x_i, \ldots, x_{|d|}, [\text{SEP}]\}$ to be compatible with the input of BERT. We use the hidden vector of [CLS] from the last transformer layer as the text representation for X_s. Thus, given a target-tweet pair, we could encode the tweet, target, and target-tweet pair as h_d, h_t and h_{pair} respectively.

2.2 Simplified Stance Reasoning Process

To deliberate the stance of a tweet towards the given target, one may identify the mention of the given target in the tweet, extract the span of corresponding opinions towards the given target in the tweets and capture the interactions between the target and opinions. However, due to the informal form of texts used on social network platforms, such intermediate annotations are difficult and expensive to acquire at scale for existing stance benchmarks. As a result, it is impractical to train stance detection model with these intermediate opinion and entity extraction, and opinion understanding sub-tasks.

To tackle these challenges, we seek to simplify those intermediate sub-tasks into easier ones that require only sentence-level instead of token-level annotations. Specifically, we consider the following two binary classification sub-tasks[3]:

- **TMT**: whether a target T is mentioned in a tweet D. It is designed to make the model aware of the (both explicit and implicit) existence of target in tweet. This can be seen a simplified version of boundary detection of the target in the tweet. Intuitively, this could help restrain stance detection models from assigning stance to non-mentioned entities.

- **STT**: whether a tweet D expresses stance towards a target T. It requires the model to distinguish between the *None* stance and other stances.

These two sub-tasks cannot be solved by only using the tweets, thus preventing stance detection models from relying on spurious features in tweets. For instance, in Figure 2, the TMT sub-task will discourage stance models from predicting *Against* when the target is *Feminist Movement* as it does not appear in the tweet.

2.3 Mutual Information Maximization

Our key intuition behind introducing intermediate tasks is to strengthen the interaction between the opinioned target and the tweet. To achieve this, we implement two sub-tasks through maximizing mutual information (**MIMax**) between the tweets and opinioned targets. Motivated by (Hjelm et al., 2018; Tian et al., 2019; Yeh and Chen, 2019), we estimate the lower bound of mutual information between two random variables X and Y using Jensen-Shannon divergence (JS), which is implemented using the binary cross-entropy (BCE) loss:

$$\begin{aligned} MI(X,Y) \geq\ & E_P[\log(g(x,y)] \\ & + \frac{1}{2}E_N[1-\log(g(x,\bar{y})] \\ & + \frac{1}{2}E_N[1-\log(g(\bar{x},y)] \end{aligned} \quad (1)$$

where E_P and E_N refer to expectations over positive and negative samples respectively, and g is the discriminator function that outputs a real number modeled by a neural network. And $(x, \bar{y})$ and $(\bar{x}, y)$ are negative samples sampled from the product of marginals. The discriminator function g is a bi-linear function defined as follows:

$$g(x,y) = x^{\mathrm{T}}\mathbf{W}y$$

where W is a learnable scoring parameter.

2.3.1 Sub-task1: TMT

In TMT, as tweets are expected to carry the information of their opinioned targets, we choose to

[3]We describe details of the acquisition of labels for these tasks in Appendix A.4

maximize the averaged MI between the representation of the tweet and the representation of the target appearing in the tweet. Specifically, a positive example is obtained if the target appears in the tweet. Negative examples, on the other hand, are constructed by replacing the current target with a new target that does not appear in the tweet, and by replacing the current tweet with a new tweet that does not contain the current target respectively.

Following Equation 1, the objective for the sub-task TMT is formulated as follows:

$$\begin{aligned}\mathcal{L}_{TMT}(x_t,x_d,\bar{x_t},\bar{x_d}) &= E_P[\log(g(x_t,x_d))] \\ &+ \frac{1}{2}E_N[1-\log(g(\bar{x_t},x_d))] \\ &+ \frac{1}{2}E_N[1-\log(g(x_t,\bar{x_d}))]\end{aligned} \tag{2}$$

where x_d is a tweet and x_t is a target that is referred to in x_d, $\bar{x_t}$ is another target that is not mentioned in x_d, and $\bar{x_d}$ is another tweet that does not mention x_t.

2.3.2 Sub-task2: STT

Different from TMT, STT aims to uncover whether the tweet expresses any stance towards the given target. If the target is not mentioned in the tweet or the tweet does not express any polarized opinion towards the target, the label for STT will be *No*. Specifically, a positive example is obtained if the target expresses *Favor* or *Against* stance towards the target. Negative examples are constructed by replacing the current target with a new target that the tweet has no opinion on and by replacing the current tweet with another tweet that does not express any stance towards the given target respectively.

Following Equation 2, the objective for the sub-task TMT is formulated as follows:

$$\begin{aligned}\mathcal{L}_{STT}(x_t,x_d,\tilde{x_t},\tilde{x_d}) &= E_P[\log(g(x_t,x_d))] \\ &+ \frac{1}{2}E_N[1-\log(g(x_t,\tilde{x_d}))] \\ &+ \frac{1}{2}E_N[1-\log(g(\tilde{x_t},x_d))]\end{aligned} \tag{3}$$

where x_t is a target and x_d is a tweet that expresses opinion on x_t , $\tilde{x_t}$ is another target and x_d holds no stance on $\tilde{x_t}$, and $\tilde{x_d}$ is a tweet that does not expresses opinion on x_t.

2.4 Stance Classification

We feed the concatenation of the given <target, tweet> pair into BERT encoder to learn a target-aware stance representation $\mathbf{h}_{pair}$. We also feed the given pair into two sub-task and obtain feature representation $\mathbf{h}_{tmt}$ and $\mathbf{h}_{stt}$ respectively. We further concatenate $\mathbf{h}_{pair}$, $\mathbf{h}_{tmt}$ and $\mathbf{h}_{stt}$ as $\mathbf{h}_{final}$, and feed $\mathbf{h}_{final}$ into a simple feed-forward network for stance classification:

$$\mathbf{h}_{sc} = \mathbf{h}_{pair} \oplus \mathbf{h}_{tmt} \oplus \mathbf{h}_{stt} \tag{4}$$

$$y_s = \text{softmax}(\mathbf{W}_{sc_2}\sigma(\mathbf{W}_{sc_1}\mathbf{h}_{sc})) \tag{5}$$

where $\mathbf{W}_{sc_2}$ and $\mathbf{W}_{sc_1}$ are learnable weight matrices.

And the classifier is trained with the following cross-entropy loss of stance classification:

$$\mathcal{L}_{SC} = -\frac{1}{N_s}\sum_{i=1}^{N_s}\sum_{j=1}^{N_o}\hat{y}_s^i(j)\log y_s^i(j) \tag{6}$$

where N_s is the number of training instances and N_o is the number of different stance labels.

The final objective for our multiple sub-task learning method becomes:

$$\mathcal{L}_{msl} = \mathcal{L}_{SC} + \lambda_1\mathcal{L}_{TMT} + \lambda_2\mathcal{L}_{STT} \tag{7}$$

where λ_1 and λ_2 are hyper-parameters to control impacts of two sub-tasks respectively.

Target	#Total	#Train	#Test
Atheism	733	513	220
Climate Change	564	395	169
Feminist Movement	949	664	285
Hillary Clinton	984	689	295
Abortion	933	653	280
All	4163	2914	1249

Table 1: Statistics of SemEval2016 Task 6 Subtask A.

New Test Sets	Number
Tweet_only Failed (TOF)	319
PMI	403
Opinion Towards (OT)	425
Donald Trump (DT)	707
Target Replaced (Replaced)	3978
Target Negated (Negated)	1249

Table 2: Statistics of collected and constructed test sets.

3 Experimental Setups

3.1 Datasets

To thoroughly assess whether a model understands the stance detection task, we collect and construct a series of test sets to assess the understanding of the stance detection task from various perspectives.

3.1.1 In-domain Dataset

We train our model on the dataset from SemEval 2016 Task 6 Sub-task A. The dataset is comprised of 4163 English tweets and each is assigned with a target and a manually annotated stance label towards that target. There are a total of five targets in Sub-task A. The detailed statistics of this dataset are shown in Table 1. We use the official train/test split. We randomly select 15% of samples from the training data as the validation set.

3.1.2 New Test Sets

To test whether the trained stance detection models overly rely on bias features in the training set, we collect three subsets where bias features from the original test set may not hold. Additionally, we use the data from SemeEval2016 Task 6 subtask B as an out-of-domain test set to evaluate the generalization ability of stance detection models. Moreover, we also construct two adversarial sets to test the sensitivity of stance detection models when changing targets.

In-distribution Hard Set. Those hard sets are collected from the original test set. 1) Tweet Only Failed (**TOF**). This subset is collected from the original test set where three BERT_{nt} (see Sec.3.3) models with different seeds all fail. The filtered subset is to assess whether models could succeed in cases where only using tweets may not make correct predictions. 2) **PMI**. This subset is filtered by removing instances containing features with the top 200 point-wise mutual information scores of each stance from the original test set. It could help test whether models perform well on long-tailed features. 3) Opinion Towards (**OT**). We keep instances from the original test set with indirect mention and no mention of targets according to additional annotations provided by (Mohammad et al., 2016). This could be used to diagnose whether given models are aware of implicit mentions of targets in the tweet part.

Adversarial Test Sets. Those hard sets are adversarially constructed based on the original test set. 1) Target Replaced (**Replaced**). To obtain this set, we replace the original target with other targets from the SemEval2016 training dataset. To ensure that the replaced targets are not mentioned in the tweet, we utilize ConceptNet (Speer et al., 2017) to enhance pattern matching. After replacement, we label the new instances with *None*. 2) Target Negated (**Negated**). It is constructed by negating the targets and keeping the tweets unchanged. The stance labels flip accordingly. For entity-like targets, we add 'NOT' in front of the original targets. For 'Atheism', we add the negated target 'Theism'. For claim-like targets, we add 'not' into the sentence to negate the claim.

Out-of-distribution Hard Set. Donald Trump (**DT**). It comes from SemEval2016 task B with an unseen target 'Donald Trump' in task A, which is used to test the generalization ability of stance detection models.

3.2 Evaluation Metrics

Similar to previous work, we adopt the macro-average of F1-score of Favor and Against across targets as the evaluation metric (see Appendix A.2). We report the averaged results of 5 random seeds for all experiments. For details of implementation, please see Appendix A.3.

3.3 Baselines

Stance Detection Models Methods on Stance detection: 1) BERT_{wt} (*wt*: with target) and BERT_{nt} (*nt*: no target), which are based on BERT_{base}. BERT_{nt} only uses the tweet as input while BERT_{wt} takes the <target, tweet> pair as inputs. 2) TAN (Du et al., 2017), which is an LSTM based model that incorporates target-specific attention. We adopt the BERT version of TAN (Kaushal et al., 2021). 3) Stancy (Popat et al., 2019), which is a BERT based model with an additional cosine similarity score between the tweet representation and the target-tweet pair representation.

Debiasing Models Apart from sophisticated stance detection models, we also compare with recent debiasing methods for natural language inference and fact verification tasks. These methods are: 1) Product-of-expert (PoE) (Clark et al., 2019), which combines the learned probabilities of a bias-only model and a full model using PoE. 2) LMH (Clark et al., 2019), which explicitly determines how much to trust the bias in PoE and employs an entropy-based regularization to encourage the bias component to be non-uniform. 3) E2E PoE (Karimi Mahabadi et al., 2020), which proposes an end-to-end training version of PoE. 4) Conf-Reg (Utama et al., 2020a), which utilizes signals from bias models to scale the confidence of models' predictions.

Models	Original	TOF	PMI	OT
BERT$_{nt}$	67.84	73.9	50.8	38.15
BERT$_{wt}$	69.21	85.56	51.66	42.62
TAN	68.44	86.51	59.31	43.88
Stancy	70.3	95.34	59.78	44.67
PoE	68.96	94.06	55.15	41.53
LMH	64.88	82.69	47.46	33.81
E2E-PoE	61.68	84.88	53.93	39.08
Conf-Reg	70.24	90.45	**61.26**	41.16
SSR (ours)	**71.36**	**96.47**	56.08	**46.58**

Table 3: Results on the SemEval test dataset and three subsets. The average of F_{Favor} and $F_{Against}$ is adopted as the evaluation metric. For comparison with other stance detection models on each target, please refer to Table 6 in Appendix.

Model	DT	Replaced	Negated
BERT$_{nt}$	11.42	32	17.59
BERT$_{wt}$	28.12	47.08	17.38
TAN	27.09	35.55	20.05
Stancy	32.34	49.5	19.67
PoE	19.42	46.7	19.54
LMH	36.76	34.97	**25.36**
E2E-PoE	33.72	33.01	19.98
Conf-Reg	36.27	34.17	19.51
SSR (ours)	**37.66**	**59.7**	17.6

Table 4: Performances of different models on *DT*, *Replaced*, *Negated* test sets. Note that, results on *DT* are not directly comparable to those reported in (Allaway et al., 2021; Liang et al., 2022) as they used 4,163 pairs for training while we only use 2,914 pairs.

4 Results and Analysis

4.1 Main Results

4.1.1 On Original Test Set and its Subsets

As shown in Table 3, while existing stance models achieve remarkable progress on the original test set, models that consider targets (BERT$_{wt}$, Stancy) only slightly outperform models that do not consider targets (BERT$_{nt}$, TAN). This indicates that existing dataset bias allows stance detection model to achieve good results solely relying on tweets. While debiasing models tend to down-weight bias features and samples, useful features for the stance detection task are inevitably influenced, leading to the performance drop of PoE, LMH , and E2E-PoE on original test set. On the contrary, our model improves BERT$_{wt}$ by 2.15% on the original test sets, showing that utilization of stance reasoning sub-tasks could facilitate learning better features for stance detection.

Though *TOF* is constructed by selecting samples from the test test where three BERT$_{nt}$ models with random seeds fail, a new BERT$_{nt}$ model with another seed still reaches 73.9%. This implies that *failure* examples may not transfer across the same models with different initialization. Nevertheless, models explicitly considering targets outperform those not considering the target by a large margin.

By comparing SSR and BERT$_{wt}$, we find that over 70% improvement of SSR over BERT$_{wt}$ on original test sets comes from the *PMI* subset. This shows that by considering the simplified stance reasoning process, SSR is less likely to rely on bias features.

On *OT* set, we can see that BERT$_{nt}$ performs poorly as it does not consider target and thus is not capable of capturing implicit mention of targets. In contrast, SSR explicitly models whether the target is mentioned in the tweet and achieves the best performance.

4.1.2 On New Test Sets

To test whether stance detection models understand the task instead of solely fitting the dataset, we present results of several representative stance detection models in Table 4.

On *DT* set, we note that BERT$_{nt}$ performs the worst on the out-of-domain dataset. BERT$_{wt}$ and TAN perform slightly better. Our SSR model performs better than other models, which suggests that leveraging intermediate tasks could help learn more transferable features for cross-target generation. For debiasing methods, PoE and E2E-PoE work well on in-distribution hard sets while performing worse on out-of-distribution test sets, and vice versa.

As the *Replaced* set is to test awareness to change of the target, BERT$_{wt}$, Stancy and SSR that explicitly capture interactions between the target and the tweet, outperform other models by a large margin.

On the *Negated* set, while LMH model achieved the highest performance of 25.36%, many other models only tangle around 20%, showing that those models can not tackle with negations of semantics in the targets. Our model performs worse on this set as negated targets and original targets will have the same labels in both sub-tasks.

4.2 Ablation Study

As shown in Figure 5, both the TMT and STT subtasks contribute to the performance on the original test set. This indicates that appropriate subtasks could help learns better features for the main task.

Models	Orig.	TOF	PMI	OT
SSR	**71.36**	**96.47**	56.08	46.58
w/o tmt	70.10	91.91	62.21	44.64
w/o stt	70.65	92.47	63.4	**47.77**
SSR w/o MIMax	70.36	88.88	**66.33**	45.79
w/o tmt	68.97	93.7	63.58	41.01
w/o stt	68.99	91.08	63.39	45.25
		DT	Replaced	Negated
SSR		**37.66**	**59.7**	17.6
w/o tmt		37.61	57.37	19.58
w/o stt		32.64	48.95	18.24
SSR w/o MIMax		26.01	49.35	18.76
w/o tmt		34.83	44.77	**20.22**
w/o stt		29.5	49.01	18.71

Table 5: Ablation study on the proposed SSR model.

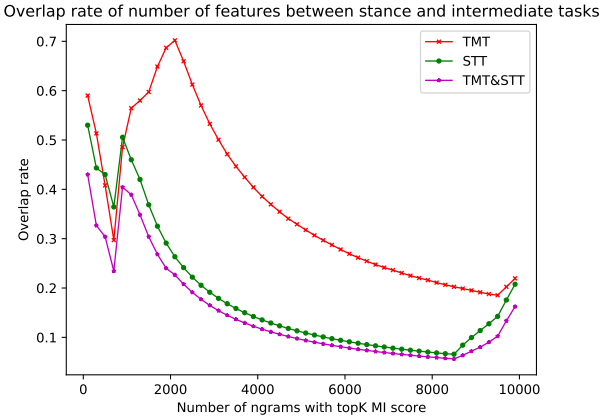

Figure 3: Overlap of top features of different tasks according to mutual information.

Adding MIMax further improves the performances by 1%.

While on *Negated* set, both tasks fail to handle the negation and lead to worse performance. As the TMT task is capable of understanding implicit mention of targets in the tweet, it is more useful than the STT task on *OT* set. Since TMT and STT can detect whether the target is changed, they both contribute to performance gain on the *Replaced* set. Generally, MIMax helps the SSR model learns more genuine stance features and improves performances on the *TOF* and *DT* set, where mutual connections between tweets and opinion targets are crucial, and either TMT or STT alone is not enough to capture such connections.

4.3 Analysis

Intermediate reasoning tasks help reduce reliance on shortcut features. After obtaining synthetic labels for these two subtasks, we conduct qualitative analysis to show that subtasks could potentially regularize the features used by the stance detection task. As mutual information (MI) scores

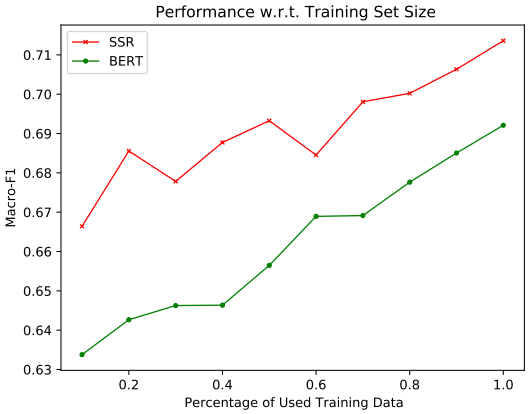

Figure 4: Performance of different models w.r.t the training size.

between features and categories serve as a good indicator of the importance of features, we first collect bag-of-words features for N-grams in the training set and sort those N-grams according to their MI scores with the labels in corresponding tasks. In Figure 3, we can see that both subtasks have less than 60% overlap of the top 1000 features, which means many important features used by the stance detection task are not useful for these subtasks. This could be seen as implicitly re-weighting features used by the stance detection task based on intermediate stance reasoning subtasks, which hopefully will promote learning genuine features instead of bias ones.

SSR requires less data for training. Previously, (Dua et al., 2020) found that collecting intermediate annotations for up to 10% training data can improve the performance of a reading comprehension model by 4-5%. Here, as annotations of intermediate tasks could be automatically acquired without human annotations, we would explore whether incorporating intermediate tasks could help reduce the demands for training data. In Figure 4, we show that when only 10% of training data is available, our model outperforms BERT$_{wt}$ model by 3.3% on macro-F1. In contrast, BERT$_{wt}$ model has to use 60% training data to reach comparable performance. Thus, utilizing intermediate stance reasoning tasks could help reduce the demand of training data and improve the performance in low-resource stance detection scenarios.

4.4 Case Study

We compare our model with two BERT baselines to illustrate the target awareness provided by in-

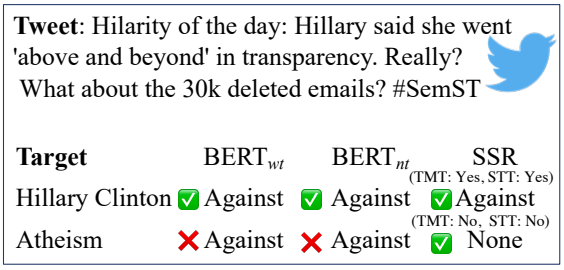

Figure 5: Case study on target-awareness of different models.

termediate subtasks. In Figure 5, we can see that when the target is *Atheism*, which is not mentioned in the tweet at all, both BERT baselines falsely predict the stance as *Against*, which may owe to the existence of the shortcut word 'email'. In contrast, as the TMT task tells the model that the given target is not mentioned in the tweet, and the STT task shows that the tweet has no stance towards the given target, our model could correctly predict the stance label as *None*.

5 Related Work

5.1 Stance Detection

Recently, detection stances in texts from social media platforms have attracted a lot of attention. Compared to traditional sentiment analysis tasks, stance detection is more challenging as the given target may not appear in the text. Inferring the relations between the given target and the opinioned entity usually requires rich world knowledge. In this paper, we focus on the single target stance detection on tweets where each tweet is given one target. Various methods (Augenstein et al., 2016; Du et al., 2017; Popat et al., 2019) have been proposed to model the inter-dependency between the target and tweet. However, Kaushal et al. (2021) recently noted that current stance detection models relied heavily on bias features in existing stance detection datasets, which makes it necessary to develop stance detection specific debiasing methods to combat these biases.

In this work, we also study the problem of dataset bias in stance detection and propose a novel method incorporating simplified stance reasoning process. Furthermore, we collect and construct 6 new test sets to facilitate evaluation of whether stance detection models truly understand the task.

5.2 Debiasing Dataset Bias in NLP

Recently, the community has shown that neural models can achieve good performances by leveraging dataset bias in various natural language understanding tasks, e.g. NLI (Gururangan et al., 2018; Poliak et al., 2018), question answering (Mudrakarta et al., 2018), VQA (Agrawal et al., 2018), machine translation, summarization, fact verification (Schuster et al., 2019) and sentiment analysis (Wang and Culotta, 2020, 2021; Kaushal et al., 2021; Yan et al., 2021; Yang et al., 2021). Such phenomenon mainly attributes that neural models tend to utilize superficial features in the dataset instead of understanding the semantics of underlying tasks, e.g. NLI models usually exploits word overlap and syntactic patterns, and even only use features from the hypothesis for final predictions.

To mitigate dataset bias, one line of work (Clark et al., 2019; Karimi Mahabadi et al., 2020; Utama et al., 2020a; Ghaddar et al., 2021) implicitly treated features learned by a smaller model or common model at earlier stages/layers as potential bias features and down-weighted these features in the main model. Another line of work adopted data augmentation strategies to weaken the spurious correlations between bias features and final labels. Beside, Tu et al. (2020) employed a multi-task learning based-approach by introducing auxiliary tasks like paraphrase identification to avoid overfitting to bias features. However, most previous methods are task-agnostic, which failed to utilize the task knowledge of the underlying task. To this end, Dua et al. (2020); Shao et al. (2021) introduced manual annotations of intermediate reasoning steps to combat dataset biases.

In this work, we make the first attempt to incorporate stance reasoning steps to combat dataset biases in stance detection. Our work differs in the following ways (1) we introduce two simplified sub-tasks whose labels can be automatically inferred instead of manual annotations (2) we novelly implement two sub-tasks via maximizing the mutual information between the tweet and the opinioned target.

6 Conclusion

In this paper, we propose to utilize the stance reasoning process as task knowledge to guide the discrimination between genuine and bias stance features. To alleviate demands for token-level intermediate annotations, we simplify the stance reasoning

process where labels for proposed subtasks can be automatically inferred without additional annotations. To evaluate whether stance detection models understand the task from various aspects, we collect and construct 6 new test sets. Empirical results show that our model outperforms task-agnostic debiasing methods on 4/6 new test sets while maintaining comparable performances to existing stance detection models on in-domain datasets.

Acknowledgements

We thank the anonymous reviewers for their insightful comments and suggestions. We thank Haichao Zhu for his valuable suggestions on re-framing the paper.This work was supported by the National Key RD Program of China via grant 2020AAA0106501 and the National Natural Science Foundation of China (NSFC) via grant 62176078. Yanyan Zhao is the corresponding author.

References

Aishwarya Agrawal, Dhruv Batra, Devi Parikh, and Aniruddha Kembhavi. 2018. Don't just assume; look and answer: Overcoming priors for visual question answering. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.

Emily Allaway, Malavika Srikanth, and Kathleen McKeown. 2021. Adversarial learning for zero-shot stance detection on social media. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 4756–4767, Online. Association for Computational Linguistics.

Isabelle Augenstein, Tim Rocktäschel, Andreas Vlachos, and Kalina Bontcheva. 2016. Stance detection with bidirectional conditional encoding. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 876–885, Austin, Texas. Association for Computational Linguistics.

Christopher Clark, Mark Yatskar, and Luke Zettlemoyer. 2019. Don't take the easy way out: Ensemble based methods for avoiding known dataset biases. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 4069–4082, Hong Kong, China. Association for Computational Linguistics.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, Minneapolis, Minnesota. Association for Computational Linguistics.

Jiachen Du, Lin Gui, Ruifeng Xu, Yunqing Xia, and Xuan Wang. 2020. Commonsense knowledge enhanced memory network for stance classification. *IEEE Intelligent Systems*, 35(4):102–109.

Jiachen Du, Ruifeng Xu, Yulan He, and Lin Gui. 2017. Stance classification with target-specific neural attention. In *Proceedings of the Twenty-Sixth International Joint Conference on Artificial Intelligence, IJCAI 2017, Melbourne, Australia, August 19-25, 2017*, pages 3988–3994. ijcai.org.

Dheeru Dua, Sameer Singh, and Matt Gardner. 2020. Benefits of intermediate annotations in reading comprehension. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 5627–5634, Online. Association for Computational Linguistics.

Abbas Ghaddar, Phillippe Langlais, Mehdi Rezagholizadeh, and Ahmad Rashid. 2021. End-to-end self-debiasing framework for robust NLU training. In *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*, pages 1923–1929, Online. Association for Computational Linguistics.

Suchin Gururangan, Swabha Swayamdipta, Omer Levy, Roy Schwartz, Samuel Bowman, and Noah A. Smith. 2018. Annotation artifacts in natural language inference data. In *Proceedings of NAACL2018, Volume 2 (Short Papers)*, pages 107–112, New Orleans, Louisiana. Association for Computational Linguistics.

Momchil Hardalov, Arnav Arora, Preslav Nakov, and Isabelle Augenstein. 2021. Cross-domain label-adaptive stance detection. *CoRR*, abs/2104.07467.

R Devon Hjelm, Alex Fedorov, Samuel Lavoie-Marchildon, Karan Grewal, Adam Trischler, and Yoshua Bengio. 2018. Learning deep representations by mutual information estimation and maximization. *arXiv preprint arXiv:1808.06670*.

Rabeeh Karimi Mahabadi, Yonatan Belinkov, and James Henderson. 2020. End-to-end bias mitigation by modelling biases in corpora. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 8706–8716, Online. Association for Computational Linguistics.

Ayush Kaushal, Avirup Saha, and Niloy Ganguly. 2021. tWT–WT: A dataset to assert the role of target entities for detecting stance of tweets. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 3879–3889, Online. Association for Computational Linguistics.

Divyansh Kaushik, Amrith Setlur, Eduard H Hovy, and Zachary Chase Lipton. 2021. Explaining the efficacy of counterfactually augmented data. In *International Conference on Learning Representations*.

Yingjie Li and Cornelia Caragea. 2019. Multi-task stance detection with sentiment and stance lexicons. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 6299–6305, Hong Kong, China. Association for Computational Linguistics.

Yingjie Li and Cornelia Caragea. 2021. Target-aware data augmentation for stance detection. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 1850–1860, Online. Association for Computational Linguistics.

Bin Liang, Zixiao Chen, Lin Gui, Yulan He, Min Yang, and Ruifeng Xu. 2022. Zero-shot stance detection via contrastive learning. In *Proceedings of the ACM Web Conference 2022*, WWW '22, page 2738–2747, New York, NY, USA. Association for Computing Machinery.

Matthew Matero, Nikita Soni, Niranjan Balasubramanian, and H. Andrew Schwartz. 2021. Melt: Message-level transformer with masked document representations as pre-training for stance detection. *CoRR*, abs/2109.08113.

Saif Mohammad, Svetlana Kiritchenko, Parinaz Sobhani, Xiaodan Zhu, and Colin Cherry. 2016. SemEval-2016 task 6: Detecting stance in tweets. In *Proceedings of the 10th International Workshop on Semantic Evaluation (SemEval-2016)*, pages 31–41, San Diego, California. Association for Computational Linguistics.

Pramod Kaushik Mudrakarta, Ankur Taly, Mukund Sundararajan, and Kedar Dhamdhere. 2018. Did the model understand the question? In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1896–1906, Melbourne, Australia. Association for Computational Linguistics.

Adam Poliak, Jason Naradowsky, Aparajita Haldar, Rachel Rudinger, and Benjamin Van Durme. 2018. Hypothesis only baselines in natural language inference. In *Proceedings of the Seventh Joint Conference on Lexical and Computational Semantics*, pages 180–191, New Orleans, Louisiana. Association for Computational Linguistics.

Kashyap Popat, Subhabrata Mukherjee, Andrew Yates, and Gerhard Weikum. 2019. STANCY: Stance classification based on consistency cues. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 6413–6418, Hong Kong, China. Association for Computational Linguistics.

Benjamin Schiller, Johannes Daxenberger, and Iryna Gurevych. 2020. Stance detection benchmark: How robust is your stance detection? *CoRR*, abs/2001.01565.

Tal Schuster, Darsh Shah, Yun Jie Serene Yeo, Daniel Roberto Filizzola Ortiz, Enrico Santus, and Regina Barzilay. 2019. Towards debiasing fact verification models. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 3419–3425, Hong Kong, China. Association for Computational Linguistics.

Zhihong Shao, Lifeng Shang, Qun Liu, and Minlie Huang. 2021. A mutual information maximization approach for the spurious solution problem in weakly supervised question answering. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 4111–4124, Online. Association for Computational Linguistics.

Robert Speer, Joshua Chin, and Catherine Havasi. 2017. Conceptnet 5.5: An open multilingual graph of general knowledge. In *Thirty-First AAAI Conference on Artificial Intelligence*.

Yonglong Tian, Dilip Krishnan, and Phillip Isola. 2019. Contrastive multiview coding. *arXiv preprint arXiv:1906.05849*.

Lifu Tu, Garima Lalwani, Spandana Gella, and He He. 2020. An empirical study on robustness to spurious correlations using pre-trained language models. *Transactions of the Association for Computational Linguistics*, 8:621–633.

Prasetya Ajie Utama, Nafise Sadat Moosavi, and Iryna Gurevych. 2020a. Mind the trade-off: Debiasing NLU models without degrading the in-distribution performance. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 8717–8729, Online. Association for Computational Linguistics.

Prasetya Ajie Utama, Nafise Sadat Moosavi, and Iryna Gurevych. 2020b. Towards debiasing NLU models from unknown biases. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 7597–7610, Online. Association for Computational Linguistics.

Zhao Wang and Aron Culotta. 2020. Identifying spurious correlations for robust text classification. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 3431–3440, Online. Association for Computational Linguistics.

Zhao Wang and Aron Culotta. 2021. Robustness to spurious correlations in text classification via automatically generated counterfactuals. *Proceedings of the AAAI Conference on Artificial Intelligence*, 35(16):14024–14031.

Hanqi Yan, Lin Gui, Gabriele Pergola, and Yulan He. 2021. Position bias mitigation: A knowledge-aware graph model for emotion cause extraction. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 3364–3375, Online. Association for Computational Linguistics.

Linyi Yang, Jiazheng Li, Padraig Cunningham, Yue Zhang, Barry Smyth, and Ruihai Dong. 2021. Exploring the efficacy of automatically generated counterfactuals for sentiment analysis. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 306–316, Online. Association for Computational Linguistics.

Yi-Ting Yeh and Yun-Nung Chen. 2019. QAInfomax: Learning robust question answering system by mutual information maximization. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 3370–3375, Hong Kong, China. Association for Computational Linguistics.

A Appendix

A.1 In-domain Dataset

We train our model on the dataset from SemEval 2016 Task 6 Sub-task A. The dataset is comprised of 4163 English tweets crawled from Twitter[4] and each is assigned with a target and a manually annotated stance label towards that target. There are a total of five targets in Sub-task A, which are *Atheism* (AT), *Climate Change is a real Concern* (CC), *Feminist Movement* (FM), *Hillary Clinton* (HC), and *Legalization of Abortion* (LA).

A.2 Evaluation Metrics

Similar to previous work, we adopt the macro-average of F1-score of *Favor* and *Against* across targets as the evaluation metric. We report the averaged results of 5 random seeds for all experiments.

Similar to previous work, we adopt the macro-average of F1-score across targets as the evaluation metric, which is calculated as:

$$F_{Favor} = \frac{2P_{Favor}R_{Favor}}{P_{Favor} + R_{Favor}} \quad (8)$$

$$F_{Against} = \frac{2P_{Against}R_{Against}}{P_{Against} + R_{Against}} \quad (9)$$

$$F_{macro} = \frac{2(F_{Against} + F_{Favor})}{2} \quad (10)$$

where P and R are precision and recall. Then the average of F_{Favor} and $F_{Against}$ is calculated as the final metrics F_{macro}. Note that the final metrics do not disregard the *None* class. By taking the average F-score for only *Favor* and *Against* classes, we treat *None* as a class that is not of interest. We report the averaged results of 5 random seeds for all experiments.

A.3 Implementation Details

We adopt the uncased version of $\text{BERT}_{\text{base}}$ for all our experiments. We fine-tune $\text{BERT}_{\text{base}}$ model with Adam optimizer. The dropout rate is set to 0.5 for all parameters. The learning rate is chosen from $\{1, 2, 3, 4, 5\} \times 10^{-5}$ and batch size for training is set to 8. We choose λ_1 and λ_2 from [0.1, 1.0] with a step size of 0.1. Final choices of all hyper-parameters are selected according to performance on the validation set. λ_1 and λ_2 are set to 0.1 and 0.2 respectively. The learning rate is set to 5×10^{-5}.

[4]https://www.twitter.com

A.4 Details for label acquisition of sub-tasks

To acquire labels for the TMT task, we seek to expand the original targets with external resources. Specifically, we take two structured knowledge bases, ConceptNet and WikiData. ConceptNet mainly contains commonsense knowledge, WikiData mainly contains social knowledge. They complement each other and supply rich target related knowledge for target understanding. Each of the targets in the dataset is treated as a key for searching for the most related commonsense knowledge from them. In this way, we augment each target with external knowledge base. Then given a tweet, we use the augmented targets to performance exact string matching, if any of the augmented targets locates in the tweet, the TMT label would be *Yes*. Otherwise, the TMT label is set to *No*.

To obtain labels for STT task, given a <target, tweet> pair and its corresponding stance label, if the stance is *Favor* or *Against*, then the STT label would be *Yes*. Otherwise, the STT label is set to *No*.

As shown above, the label acquisition for TMT and STT subtasks are easy and straightforward, which does not involve additional mannual annotation. Thus, it is practical to incorporate the above subtasks as parts of the simplified stance reasoning process.

A.5 Detailed results on original dataset

In Table 6, we compare our model with recent stance detection models. We can see that our SSR model performs comparably to existing stance detection models that utilized external stance detection datasets, lexicons and tweet corpora.

A.6 Additional Analysis

Bias is ubiquitous in multiple tasks on the same dataset. Similarly, we apply BERT_{nt} and BERT_{wt} model on the TMT and STT tasks respectively. As shown in in Table 7, BERT_{nt} could achieve relative performance even when no target information is utilized for those target-aware tasks, suggesting the ubiquitous of bias on different tasks in this dataset. Moreover, We can see that for the TMT task, the discrepancy between BERT_{nt} and BERT_{wt} is 8.7%, significantly larger than that of the STT task and stance detection task. This shows that TMT requires more interactions between the target and the tweet, which may account for the better performances on *OT* and *Replaced* hard sets

Models	AT	CC	FM	HC	LA	Overall
AT-JSS-Lex (Li and Caragea, 2019)	69.22	59.18	61.49	**68.33**	**68.41**	**72.33**
CKEMN (Du et al., 2020)	62.69	53.52	61.25	64.19	64.19	69.74
MT-DNN$_{SDL}$ (Schiller et al., 2020)	-	-	-	-	-	70.18
MT-DNN$_{MDL}$ (Schiller et al., 2020)	-	-	-	-	-	71.81
MoLE (Hardalov et al., 2021)	-	-	-	-	-	72.08
ASDA (Li and Caragea, 2021)	**74.93**	-	56.43	67.01	61.60	-
MeLT (Matero et al., 2021)	66	**71**	**63**	67	66	-
TAN	69.72	44.32	53.26	55.79	62.75	68.44
Stancy	66.08	54.67	59.91	62.0	58.69	70.3
BERT$_{nt}$	63.96	48.88	53.97	60.59	60.24	67.84
BERT$_{wt}$	65.36	44.91	48.25	65.09	54.33	69.21
SSR	63.17	56.88	59.68	62.69	57.33	71.36
SSR(-MIMax)	70.09	54.0	56.68	65.64	62.04	70.36

Table 6: Results on the SemEval dataset. The macro average of F_{Favor} and $F_{Against}$ is adopted as the evaluation metric. The results in bold are the best in corresponding columns. The underlined results are the second best in corresponding columns.

Task	BERT$_{nt}$	BERT$_{wt}$	Δ
TMT	83.39	92.09	8.70
STT	80.97	83.85	2.88
Stance	70.1	72.4	2.3

Table 7: Accuracy of BERT$_{nt}$ and BERT$_{wt}$ on different tasks.

when adding the TMT task instead of the STT task. Though two auxiliary sub-tasks have their own dataset bias to some extent, combing them with the main task still boosts the performance of the stance detection on both original and newly constructed hard sets. This supports our motivation of leveraging intermediate tasks to learn robust features for the main task.

Transferring Confluent Knowledge to Argument Mining

João A. Rodrigues and **António Branco**
University of Lisbon
NLX – Natural Language and Speech Group, Department of Informatics
Faculdade de Ciências, Campo Grande, 1749-016 Lisboa, Portugal
{jarodrigues,ambranco}@fc.ul.pt

Abstract

Relevant to all application domains where it is important to get at the reasons underlying sentiments and decisions, argument mining seeks to obtain structured arguments from unstructured text and has been addressed by approaches typically involving some feature and/or neural architecture engineering.

By adopting a transfer learning methodology, and by means of a systematic study with a wide range of knowledge sources promisingly suitable to leverage argument mining, the aim of this paper is to empirically assess the potential of transferring such knowledge learned with confluent tasks.

By adopting a lean approach that dispenses with heavier feature and model engineering, this study permitted both to gain novel empirically based insights into the argument mining task and to establish new state of the art levels of performance for its three main sub-tasks, viz. identification of argument components, classification of the components, and determination of the relation among them.

1 Introduction

Argument mining is a Natural Language Processing (NLP) task consisting in taking unstructured text as input and returning it annotated such that each portion occurring in it that is an argument is properly delimited and analysed (Schneider et al., 2013; Peldszus and Stede, 2013; Lippi and Torroni, 2016; Habernal and Gurevych, 2017; Wachsmuth et al., 2017; Stede and Schneider, 2018; Lawrence and Reed, 2020). Argument mining relates to the high-level human capacity of reasoning (Walton et al., 2005), it is at the core of social interaction concerned with persuasion (Mercier and Sperber, 2017), and it is of utmost importance to enhance applications across different domains that aim at enhancing their services beyond mere sentiment analysis, on the basis of the reasons uncovered for the associated sentiments and decisions (Habernal et al., 2014).

Argument mining has been decomposed into a number of sub-tasks. While the number and profiling of these tasks depends on the theoretical approach adopted to analyse arguments (Van Eemeren et al., 2019), they typically involve some sort of delimitation of the text segments conveying argument components, the classification of the roles of these components (e.g. premises, conclusions, etc.), and the classification of the type of relation among those components (e.g. support, attack, etc.) (Lawrence and Reed, 2020).

These sub-tasks and their eventual pipeline in argument mining have been addressed by means of supervised deep learning approaches that involve some degree of neural architecture engineering (Eger et al., 2017; Potash et al., 2017; Nguyen and Litman, 2016) a.o. Recently, first attempts to approach argument mining with Transformers have been reported in the literature (Wang et al., 2020; Rodrigues et al., 2020a) a.o., tough at an exploratory level that leaves much of its strength still untapped.

This has been combined with experimentation with transfer learning (Caruana, 1997; Ruder, 2019). Given its complexity, and the associated difficulty in producing gold labelled data, argument mining is a task with a scarcity of data sets that are needed to support supervised learning approaches. Enhancing the argument mining task by transferring knowledge elicited when solving other natural language processing tasks is thus a promising approach to alleviate such scarceness. This has been tried in the literature (Mohammad et al., 2016; Stab et al., 2018; Choi and Lee, 2018; Habernal et al., 2018; Rodrigues and Branco, 2020) a.o., though at a haphazard level that leaves still much of its potential to be studied.

For humans, argumentation is a high level cognitive task that goes together with a number of other

Proceedings of the 29th International Conference on Computational Linguistics, pages 6859–6874
October 12–17, 2022.

capacities relating to linguistic syntactic and semantic processing, to entailment and paraphrasing, to question answering and language comprehension, to reasoning, to common sense, etc. (Lawrence and Reed, 2020; Lauscher et al., 2021). Interestingly, there is now available in the literature a wide range of data sets and respective NLP tasks that permit to address a wide range of these different dimensions and use them as auxiliary sources of knowledge in transfer learning approaches to argument mining (Wang et al., 2018, 2019a) a.o.

In this context, our goal is to empirically assess the potential of transfer learning to support argument mining by means of a systematic study with a wide range of possible sources of related tasks and knowledge possibly suitable to be transferred. In this paper we report on the findings of exploring a vast experimental space that results from: performing sequential single-step transfer learning from over 40 auxiliary tasks to each one of three main sub-tasks of argument mining (Stab and Gurevych, 2014, 2017) during the fine-tuning phase (Section 4); further explore the source tasks that supported the best single-step transfer learning by experimenting with ways of possibly combining them in multi-step transfer learning processes, and further explore these tasks in a multi-task transfer learning setting (Section 5). This is preceded by an overview of related work (Section 2) and by the presentation of the experimental setup adopted (Section 3).

By undertaking this study, not only new state-of-the-art results were achieved for argument mining, as also new empirically based insights were gained on how this task can be enhanced, showing the effectiveness of transfer learning to leverage argument mining and to alleviate its data scarcity when combined with a lean approach that dispenses with heavier feature and model engineering.

2 Related work

Transfer learning is a technique in machine learning that leverages knowledge from other, so called source tasks to improve the learning of a target task (Caruana, 1997), being a methodology to alleviate the lack of labelled data for the latter (Ruder, 2019).

2.1 Transfer learning for argument mining

Four families of approaches of transfer learning for argument mining have been reported in the literature: (i) transfer learning across discourse domains for the same argument mining sub-task; (ii) cross-lingual transfer learning for a given sub-task; (iii) multi-task learning among argument mining sub-tasks; and (iv) sequential transfer learning from sources tasks that are not argument mining sub-tasks. A brief overview follows below.

Several papers have applied transfer learning with a **domain adaptation** approach for identifying components and clausal properties (Al-Khatib et al., 2016; Ajjour et al., 2017; Daxenberger et al., 2017). Typically, a model is trained with data sets from various discourse domains and is evaluated over each domain.

Cross-lingual transfer learning for argument mining (Aker and Zhang, 2017; Sliwa et al., 2018; Eger et al., 2018; Rocha et al., 2018) is mainly performed through direct transfer (McDonald et al., 2011) or projection (David et al., 2001) techniques. Direct transfer techniques train a model with the source language data that initializes a new model for a target language, typically with less to no data. Projection techniques resort to mapping the same labels from the source language data set to a target language data set by resorting to parallel corpora.

The argument mining pipeline has been addressed also with transfer learning by **multi-task** and **sequential** approaches (Cabrio and Villata, 2013; Peldszus and Stede, 2015; Eger et al., 2017; Potash et al., 2017; Niculae et al., 2017; Galassi et al., 2018; Schulz et al., 2018; Mensonides et al., 2019; Chakrabarty et al., 2019; Accuosto and Saggion, 2019; Cheng et al., 2020). Most proposals train models pipelining the sub-tasks in some way.

Transfer learning from **related tasks** has also been shown to improve the performance of argument mining sub-tasks. (Stab et al., 2018) transferred shared knowledge from two different tasks: a stance detection task (Mohammad et al., 2016) and a topic identification task. (Choi and Lee, 2018), in turn, transferred knowledge from the Argument Reasoning Comprehension Task (Habernal et al., 2018) for a clausal classification sub-task.

2.2 Main sub-tasks

To proceed with our systematic study of transfer learning for argument mining on a mainstream pipeline of sub-tasks (Lawrence and Reed, 2020), which includes identifying argument components, classifying their clausal roles and determining the relational properties among them, we resorted to

the AAEC corpus (Stab and Gurevych, 2014, 2017), a collection of annotated essays in English, which has been subject to various studies. An example from this data set is displayed in Figure 1.

In order to further support this option, it is worth noting that there is not in the literature a set of commonly agreed standard argument mining sub-tasks and that persuasive arguments, contained in the AAEC corpus, are by no means peripheral to argumentation, which is ultimately about persuasion. It is also worth noting that, while in NLP in general, it is always better to have more data sets/tasks for evaluation, the empirical study in this paper builds on a strong series of recent investigations that are based on one of the few data sets for argument mining, the AAEC, that given its quality and volume, has permitted comparison of results and the objective assessment of possible advances.

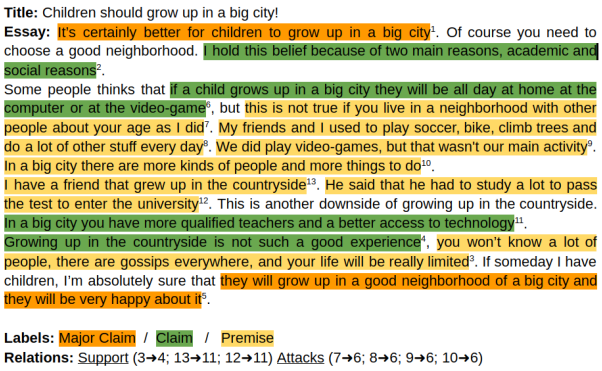
Title: Children should grow up in a big city!
Essay: It's certainly better for children to grow up in a big city[1]. Of course you need to choose a good neighborhood. I hold this belief because of two main reasons, academic and social reasons[2].
Some people thinks that if a child grows up in a big city they will be all day at home at the computer or at the video-game[6], but this is not true if you live in a neighborhood with other people about your age as I did[7]. My friends and I used to play soccer, bike, climb trees and do a lot of other stuff every day[8]. We did play video-games, but that wasn't our main activity[9]. In a big city there are more kinds of people and more things to do[10].
I have a friend that grew up in the countryside[13]. He said that he had to study a lot to pass the test to enter the university[12]. This is another downside of growing up in the countryside. In a big city you have more qualified teachers and a better access to technology[11].
Growing up in the countryside is not such a good experience[4], you won't know a lot of people, there are gossips everywhere, and your life will be really limited[3]. If someday I have children, I'm absolutely sure that they will grow up in a good neighborhood of a big city and they will be very happy about it[5].

Labels: Major Claim / Claim / Premise
Relations: Support (3➔4; 13➔11; 12➔11) Attacks (7➔6; 8➔6; 9➔6; 10➔6)

Figure 1: **Example of a labelled essay in AAEC.**

The AAEC corpus integrates the annotation of every sub-task in the argument mining pipeline into a single data set. It contains 402 manually annotated essays,[1] in English, with 7,116 sentences over 1,833 paragraphs spanning 147,271 tokens.

It adopts an argument structure model in the form of a tree composed of major claim (in the root node, as the author's standpoint on the argument topic), claims and premises. Individual paragraphs of the essay include arguments that may be linked or not-linked (via relational properties) to the author's major claim. Both "support" and "attack" relations are taken into account.

The annotation of text segments with argument components resorted to an IOB tagging scheme (Ramshaw and Marcus, 1999). The beginning of an argument component is tagged with *Arg-B*, the following tokens in that component are tagged with *Arg-I* and non-argumentative tokens with *O*. Identifying argument components consists of tagging each token with this IOB-tagset given a complete essay as a single input sequence. Identifying clausal properties consists of classifying spans of discourse with one of the three classes (major claim, claim and premise) given an entire essay as input. Following the literature, given the large imbalance between "support" and "attack" classes, identifying relational properties consists in classifying pairs of segments just as linked or not-linked. Statistics are displayed in Table 1.[2]

Task		**Labels**	**Total**	Train	Test
Comp.	Arg-B	11%	6,089	79%	21%
	Arg-I	64%	93,618	80%	20%
	O	25%	47,474	80%	20%
Clausal	Major Cl	12%	751	80%	20%
	Claim	25%	1,506	80%	20%
	Premise	63%	3,832	79%	21%
Relat.	Not-Link	82%	18,340	78%	22%
	Linked	18%	3,832	79%	21%

Table 1: For the tasks annotated in AAEC (rows), the number of instances for labels and data set split (columns) are indicated.

2.3 Literature on the AAEC tasks

Several papers on argument mining address the AAEC tasks, although none addresses all of them, except (Stab and Gurevych, 2017), which addressed each task with a feature-engineered SVM (components: 0.849 macro-F1; clausal: 0.773; relational: 0.736), and an Integer Linear Programming (ILP) algorithm (0.867, 0.826, 0.751 respectively), that is an ensemble of the SVM models supplemented by rules to ensure the correct tree structure. Table 2 presents the performance scores reported in the literature for the AAEC tasks.

Regarding the identification of argument **components** task: (Ajjour et al., 2017) implement a BiLSTM with extensive use of features and obtain 0.885 macro-F1. (Petasis, 2019) applies several types of neural networks for segmentation, with the top-performing model, a BiLSTM-CRF, obtaining 0.901 macro-F1. (Spliethöver et al., 2019) resorts to attention mechanisms with BiLSTMs for unit segmentation, with the top-performing model obtaining 0.87 weighted-F1. (Eger et al., 2017) apply different models, including multi-task learning experiments, and report 0.908 macro-F1 for the identification of components sub-task.

[1] 80 essays, i.e 20% for testing, were annotated by three annotators and the remaining 322, for training, by an expert.

[2] Further descriptions of the data set and the framing of the tasks are provided in the Appendix A.

	Comp.	Clau.	Rel.
SVMs (Stab and Gurevych, 2017)	.849	.773	.736
ILP (Stab and Gurevych, 2017)	.867	.826	.751
S2S (Potash et al., 2017)		**.849**	**.767**
BL (Ajjour et al., 2017)	.885		
BL (Eger et al., 2017)	**.908**		
BL (Spliethöver et al., 2019)	.870		
BL-CRF (Petasis, 2019)	.901		
BL-CRF (Schulz et al., 2018)		.606	
BL-CNN-CRF (Chernodub et al., 2019)		.471	
CNN-Seq. (Gemechu and Reed, 2019)		.790	
BERT (Wang et al., 2020)		.640	
LibLINEAR (Nguyen and Litman, 2016)			.753

Table 2: **Comparison of different performance scores in the literature on the AAEC tasks**, in macro-F1 (except weighted-F1 in (Spliethöver et al., 2019)), with the top results in bold, indicating the state-of-the-art (BL stands for BiLSTM). It should be noted that LibLINEAR uses the first version of the AAEC data set.

For the identification of **clausal** properties task: (Gemechu and Reed, 2019) obtain 0.79 macro-F1 for clausal properties linking premises and conclusions, taking into account the similarity of target concepts and aspects. (Chernodub et al., 2019) applied a framework for tagging arguments and their retrieval, including a BiLSTM-CNN-CRF sequence tagger. A micro-F1 of 0.645 was the top-performing performance in identifying clausal properties (0.471 macro-F1 is the reproduction in (Wang et al., 2020)). (Wang et al., 2020) propose a multi-scale mining model, resorting to several encoder-only Transformers (BERT) that mine different argumentation components at different textual levels, namely at the essay/paragraph/word-level. The top-performing model obtains 0.64 macro-F1 in identifying clausal properties. (Schulz et al., 2018) also apply a multi-task learning approach from different domains and argumentative structures, including AAEC, with a BiLSTM-CRF, obtaining 0.606 macro-F1 score.

Finally, as for **relational** properties: (Nguyen and Litman, 2016) obtain 0.753 macro-F1 combining different topic to window context features with a linear classifier (LibLINEAR). (Potash et al., 2017) report a 0.849 clausal and 0.767 relational macro-F1 using a joint pointer architecture (sequence-to-sequence model with attention), simultaneously addressing clausal and relational properties with several features.

3 Experimental space and settings

For the tasks that are the source of knowledge to be transferred to argument mining models, we resorted to a vast array of annotated data sets listed in Table 3. They cover different dimensions in terms of linguistic and cognitive processing:[3]

3.1 Source tasks

Syntax - Information on syntax is typically included in structured machine learning algorithms that address the argument mining in a feature engineering approach. We included part-of-speech (POS) tagging, named entity recognition (NER) (Hu et al., 2020) and several other tasks regarding linguistic properties of sentences (Conneau and Kiela, 2018).

Semantics - Features from semantic similarity (SS) are widely used in argument mining literature. For example, (Boltužić and Šnajder, 2015) use SS to identify prominent arguments in online debates, and (Lawrence and Reed, 2015) use SS obtained from WordNet to identify the components of argumentation schemes. We included a diversity of SS data sets, from the context-sensitive similarity task Wic (Pilehvar and Camacho-Collados, 2019) to the large data set obtained from Quora Question Pairs (QQP) (Iyer et al., 2017).

Grammaticality - To address the widest spectrum of linguistic aspects, we included also tasks on determining the grammatically of input sentences. Data sets such as the Corpus of Linguistic Acceptability (CoLA) (Warstadt et al., 2019) were used, that are challenging with regards this type of task.

Sentiment - Sentiment analysis has a certain proximity to argument mining, which adds an extra dimension to it by providing reasons for sentiments (Habernal et al., 2014). The Stanford Sentiment Treebank (SST) (Socher et al., 2013) was included.

Reasoning & Comprehension - Reasoning is at the core of argumentation given it is crucial in formulating and accepting or rejecting an argument. We included several related tasks, as for instance the AI2 Reasoning Challenge (ARC) (Clark et al., 2018) in the domain of grade-school science.

Question Answering & Common sense - Question Answering (QA) relates to argument mining given linguistic similarities between the Question/Answer and Claim/Premise pairs. Several QA

[3]We resorted also partly to PORTULAN CLARIN workbench consisting of language processing services: (Gomes et al., 2018; Branco et al., 2020; Barreto et al., 2006; Branco et al., 2010; Cruz et al., 2018; Veiga et al., 2011; Branco and Henriques, 2003; Branco and Silva, 2003; Branco et al., 2011, 2022; Branco and Nunes, 2012; Silva et al., 2010; Branco et al., 2014b; Silveira and Branco, 2012a,b; Branco and Costa, 2008; Branco et al., 2014a; Rodrigues et al., 2016; Branco and Silva, 2006; Rodrigues et al., 2020b; Costa and Branco, 2012; Santos et al., 2019; Neale et al., 2016; Miranda et al., 2011).

tasks were included that address common sense as this is closely related to argumentation, given that several implicit premises, tacit assumptions or inferences are to some extent regarded as common sense—for example, (Saint-Dizier, 2017) uses QA techniques for argument mining.

Entailment & Paraphrase - Although argument mining and Textual Entailment (TE) are different tasks, they are closely related given the similarity between specific entailment properties and argument clausal and relational properties. Works such as (Cabrio and Villata, 2012; Cocarascu and Toni, 2017) use models for TE to address argument relational properties. We included several TE tasks in different discourse domains, such as news and forums, with STSB (Cer et al., 2017), and science, with SciTAIL (Khot et al., 2018).

Argument mining - In addition to non argument mining tasks, we considered also as a source task for transfer learning the predecessor sub-task in the argument mining pipeline, that is the identification of components (for the clausal sub-task) and the clausal classification (for the relational sub-task).

3.2 Computational models

In order to explore the experimental space setup for our study, we resorted to the Transformer architecture (Vaswani et al., 2017), which became mainstream in NLP, surpassing several state-of-the-art results in a wide range of tasks of all sorts (Wang et al., 2018, 2019a). In contrast to most literature on argument mining, where structured feature engineering has been the favoured approach, a Transformer is a deep learning approach that obtains linguistic knowledge by transfer learning from a language modelling task.

In order to factorize out the impact of different possible models and obtain results that can be comparable across the different data points in our experimental space, we adopt the same type of model for all of them. Taking a look at a task closely related to argument mining, namely common sense reasoning, there are works in the literature (Branco et al., 2021) that, for this task, under comparable circumstance, have experimented with prominent exemplars of encoder-only, decoder-only, encoder-decoder, and neuro-symbolic types of Transformers, which found that RoBERTa (Liu et al., 2019) offers a clear advantage. Inspired by these results, we undertook an exploratory study, repeating the above experiments but now for sample cases of argument mining from our experimental space and arrived at the same finding. Accordingly, and given also its accessible compute requirements and top performance in several NLP tasks, we adopted the off-the-shelf RoBERTa model, resorting to RoBERTa-large variant only when the RoBERTa-base was shown not to be enough to beat the SoTA.

We used the Jiant framework (Wang et al., 2019b; Phang et al., 2020) and Huggingface (Wolf

Task	#Train
Syntax	
PANX (Hu et al., 2020)	20K
UDPOS (Hu et al., 2020)	21K
Bigram Shift (Conneau and Kiela, 2018)	100K
Coord Inversion (Conneau and Kiela, 2018)	100K
Obj number (Conneau and Kiela, 2018)	100K
Odd Man Out (Conneau and Kiela, 2018)	100K
Past-Present (Conneau and Kiela, 2018)	100K
Sentence Length (Conneau and Kiela, 2018)	100K
Subj Number (Conneau and Kiela, 2018)	100K
Top Constituents (Conneau and Kiela, 2018)	100K
Tree Depth (Conneau and Kiela, 2018)	100K
Word Content (Conneau and Kiela, 2018)	100K
Semantics	
COPA (Roemmele et al., 2011)	400
WIC (Pilehvar and Camacho-Collados, 2019)	5.4K
STSB (Cer et al., 2017)	7K
QQP (Iyer et al., 2017)	364K
Grammaticality	
Coord (White et al., 2020)	458
Eos (White et al., 2020)	479
Definiteness (White et al., 2020)	508
Whwords (White et al., 2020)	585
CoLA (Warstadt et al., 2019)	8.5K
Sentiment	
SST (Socher et al., 2013)	67K
Reasoning & Comprehension	
MULTIRC (Khashabi et al., 2018)	456
WNLI (Levesque et al., 2012)	635
ARC (Clark et al., 2018)	2.2K
ROPES (Lin et al., 2019)	10K
ANLI (Bhagavatula et al., 2020)	169.6K
FEVER (Nie et al., 2019)	208.3K
Question Answering & Common sense	
WSC (Levesque et al., 2012)	554
CommonsenseQA (Talmor et al., 2019)	9.7K
QUAIL (Rogers et al., 2020)	10.2K
BoolQ (Clark et al., 2019)	16K
PIQA (Bisk et al., 2020)	16.1K
CosmosQA (Huang et al., 2019)	25K
HellaSwag (Zellers et al., 2019)	39.9K
MRQA (Fisch et al., 2019)	104K
QNLI (Wang et al., 2018)	105K
Entailment/Paraphrase	
CB (De Marneffe et al., 2019)	1.2K
RTE (Dagan et al., 2005)	2.5K
MRPC (Dolan and Brockett, 2005)	3.7K
SciTAIL (Khot et al., 2018)	27K
MNLI (Williams et al., 2018)	393K
Argument mining	
Components (Stab and Gurevych, 2017)	117k
Clausal (Stab and Gurevych, 2017)	4k

Table 3: **Data sets used for source tasks**, clustered by linguistic and cognitive dimensions.

et al., 2020).The training objective for the pre-training model was the Mask Language Modelling (MLM), which randomly masks a word in a sentence and predicts it.

To identify argument components, a token classification head classifies the input sequence $x_{1:N}$ (full essay) and gives a possible output $y_{1:N}$ from a class set C. To identify clausal and relational properties, a sequence classification head classifies each input sequence $x_{1:N}$ and gives a possible output y from a class set C.

3.3 Baselines

As for the baselines, we included the class majority, and the scores of a RoBERTa-base model fine-tuned for each AAEC task. We also included the SVMs and ILP model from (Stab and Gurevych, 2017) as a strong baseline.

3.4 Evaluation

For the evaluation of the transfer learning, we used the final result of each main sub-task in argument mining, which is the mean score of three runs. As in the original AAEC work and given that classes are unbalanced, for all tasks we used a macro-F1 averaging (Sokolova and Lapalme, 2009). We applied the Independent Samples t-Test regarding the RoBERTa baseline and different data points obtained in our experimental space to evaluate the statistical significance (Dror et al., 2018).

4 Single-step transfer

A first batch of experiments was concerned with single-step sequential transfer learning where the source tasks were those listed in Table 3.

Given the large number of data points in this experimental space, concessions were made considering the compute footprint, and we limited the hyper-parameter search by using the recommended values (Liu et al., 2019; Wolf et al., 2020).[4]

[4]Inspired by the STILT approach (Phang et al., 2018) we adopted the jiant toolkit (Pruksachatkun et al., 2020), an open source toolkit for transfer learning experiments.

For the fine-tuning of the target tasks, we performed a hyper-parameter search with three learning rates and three seeds on the target task development set, creating a total of 396 models.

The AAEC development set was extracted from 10% of the original training data, thus the training data consists of the remaining 90%. Based on the top-performing result obtained from the development set, hyper-parameters were determined for the test set. Further descriptions of hyper-parameterization together with all materials to reproduce the experiments are available at https://github.com/nlx-group/transfer-am.

	Comp.	Clausal	Relational
Human	.886	.868	.854
SoTA - Table 2	.908	.849	.767
Baselines			
RoBERTa no transfer	.916	.820	.727
ILP	.867	.826	.751
SVM	.849	.773	.736
Majority	.259	.257	.455
Syntax	*.906*	*.718*	*.695*
PANX	.917	.815	.756
UDPOS	.914	.804	.743
Bigram Shift	.912	.710	.743
Coord Inversion	.910	.696	.735
Obj number	.907	.715	.729
Odd Man Out	.914	.703	.752
Past-Present	.901	.713	.718
Sentence Length	.885	.652	.466
Subj Number	.913	.707	.746
Top Constituents	.896	.708	**.762***
Tree Depth	.904	.674	.735
Word Content	.896	.713	.455
Semantics	*.916*	*.813*	*.745*
COPA	.919*	.823	.738
WIC	.918	.821	.744
STSB	.917	.805	.753
QQP	.911	.800	.746
Grammaticality	*.915*	*.711*	*.753*
Coord	.910	.722	.754*
Eos	.914	.712	.745
Definiteness	.914	.705	.755
Whwords	.915	.702	**.758**
CoLA	**.924**	.713	.752*
Sentiment			
SST	.916	.820	.747*
Reasoning & Compreh	*.918*	*.811*	*.701*
MULTIRC	.919	**.831**	**.758**
WNLI	.913	.788	.455
ARC	**.921**	.820	**.758**
ROPES	.920	.806	.748
ANLI	.917	.807	.749
FEVER	.914	.814	.736
QA & Common sense	*.918*	*.819*	*.717*
WSC	.919	.820	**.758**
CommonsenseQA	.916	.819	.755*
QUAIL	**.921**	.827	.755*
BoolQ	.916	**.837**	.742
PIQA	.914	.774	.455
CosmosQA	.917	.817	.745
HellaSwag	.916	.823	.746
MRQA	**.924**	.825	.750
QNLI	.916	.826	.751
Entailment/Paraphrase	*.919*	*.818*	*.744*
CB	**.923***	.819	.734
RTE	.916	**.843***	**.757**
MRPC	.916	.790	.746
SciTAIL	.919	.827	.751*
MNLI	.919	.812	.731
Argument mining			*.661*
Components		**.843**	.664
Clausal			.657

Table 4: **Performance in macro-F1 scores on the main sub-tasks (columns) by different source tasks (rows).** Top score underlined, top 3 scores in bold, average score in the same family of tasks in italics. All values found to be statistical significant (p-value < .05) are noted with an $*$

4.1 Results and Analysis

Table 4 shows the results from this first batch of experiments,[5] which support the following major empirical findings:

– **The Transformer with no transfer is a very strong baseline** (off-the-self RoBERTa-base fine-tuned to each AAEC task). It overcomes (with 0.916 in components) the SoTA (0.908) of one of the three main tasks, and has strong scores in the other two.

– **Transfer learning is effective to leverage argument mining**. This is supported by scores above the Transformer baseline: with 0.924 (against the baseline 0.916) in the components task; 0.843 (against 0.820) in the clausal task; and 0.762 (against 0.727) in the relational task.

– **Transfer learning with a Transformer is very competitive with respect to, or even surpass, the SoTA**. This is supported by a new SoTA of 0.924 in components (against 0.908), and by very good scores, 0.843 and 0.762, against respectively 0.849 and 0.767, in clausal and relational.

– **Source tasks whose overall cognitive complexity is high and closer to the argument mining task tend to be more successful in supporting effective transfer**. The overall trend is that better results are found with source tasks for Reasoning, Common sense and Entailment, as shown by the respective averages and the larger number of top scores therein. Interestingly, the top score of 0.762 for relational is obtained with a syntactic source task, that seeks to identify Top Constituents: this is of relevance for the relational task as this is about relating clausal segments, which are univocally associated with their top constituents.

– **A main sub-task can be a good source task to other sub-task for effective transfer**. This is supported by the top score 0.843 in the clausal task when the components was the source in transfer.

– **A larger size of a data set for a source task, in contrast to other sources tasks, does not necessarily lead to an enhanced performance of the transfer chain**. This is illustrated, for instance, by the case of RTE, with a small data set of only 2.5K, but with the top score for clausal.

5 Multi-step and multi-task transfer

A second batch of experiments was concerned with multi-step and multi-task transfer learning. The source tasks considered here were the ones with the best results in the previous batch of experiments with single-step transfer.

Hence, **two-step** transfer was experimented with, where the typical chain encompasses the transfer from the components task to the clausal task and from the latter to the relational task. But we experimented also with other two-step instances, where the initial source tasks in the chain, viz. RTE, CB and Top Constituents (TC), are none of the argument mining sub-tasks. Experiments with **three-step** transfer were also undertaken, where besides the main tasks, these other source tasks contributed to the chain.

Finally, besides sequential transfer, also **multi-task** transfer learning was experimented with, involving the three argument mining sub-tasks altogether, and also pairs including two of them. Motivated by these pairings of the sub-tasks, we returned to one-step methodology, and for the sake of completeness, we experimented also with every combination of two such sub-tasks.

	Comp.	Clausal	Relational
Human	.886	.868	.854
SoTA Table 2	.908	.849	.767
Baselines			
RoBERTa no transfer	.916	.820	.727
ILP	.867	.826	.751
SVM	.849	.773	.736
Majority	.259	.257	.455
Sequential			
Cl ⇒ Cp	.920		
Re ⇒ Cp	**.924**		
RTE ⇒ Cp	.916		
Re ⇒ Cl ⇒ Cp	.912		
CB ⇒ Re ⇒ Cp	.915		
Cp ⇒ Cl		.843*	
Re ⇒ Cl		.811	
RTE ⇒ Cl		.843*	
Re ⇒ Cp ⇒ Cl		.839	
RTE ⇒ Cp ⇒ Cl		**.888***	
Cp ⇒ Re			.664
Cl ⇒ Re			.657
RTE ⇒ Re			.757
Cp ⇒ Cl ⇒ Re			.781*
RTE ⇒ Cp ⇒ Cl ⇒ Re			**.783***
TC ⇒ Cp ⇒ Cl ⇒ Re			.761
Multi-task			
Cp ⇔ Cl	.915	.813	
Cp ⇔ Re	.911		.684
Cl ⇔ Re		.738	.714
Cp ⇔ Cl ⇔ Re	.906	.796	.757

Table 5: **Performance on the three main sub-tasks (columns) by different transfer learning source tasks and their chaining (rows)**, reported with macro-F1, with the top results in bold, indicating new state-of-the-art scores. Cp stands for Components, Cl for Clausal, Re for Relational and TC for Top Constituents.

[5] All scores were obtained with RoBERTa-base.

5.1 Results and Analysis

Table 5 presents the results for this second batch of experiments,[6] supporting these major findings:

– **Sequential transfer is more effective than multi-task transfer**. This is supported by the overall stronger scores in sequential transfer experiments for similar clusters of tasks.

– **Multi-step transfer can be more effective than single-step**. This is supported by the results obtained for the relational task: with the best score to relational in all experimental space of 0.783, this result was supported by a three step transfer that leveraged the relational task with the knowledge from the other two main tasks, components and clausal, and from RTE; and it is supported also by the results obtained for the clausal task: with the best score in all experimental space of 0.888, this result was supported by a two step transfer that leveraged the clausal task with the knowledge from other two tasks, one from the entailment (RTE) and the other being another main task (components).

– **Source tasks that are sub-tasks in the argument mining pipeline are very successful in enhancing effective transfer**. This is supported by the results obtained with the transfer being organized along the default argument mining pipeline direction, with top or very close to the top second scores for the chains Cp ⇒ Cl and Cp ⇒ Cl ⇒ Re, with 0.843 and 0.781, respectively. But this is supported by the results obtained with the transfer being organized also in different directions, like for instance, the best score to components in all experimental space, of 0.924, with Re ⇒ Cp.

– **Source tasks with the best performance for a given main task in the single-step setting are very successful in enhancing multi-step effective transfer, specially for that main task**. This is supported by the results obtained with top or very close to the top second scores for the chains RTE ⇒ Cp, with 0.916 (over the SoTA 0.908 for components), RTE ⇒ Cp ⇒ Cl, with 0.888 (top score for clausal, and over its SoTA 0.849), and RTE ⇒ Cp ⇒ Cl ⇒ Re, with 0.774 (over the SoTA 0.767 for relational).

– **Transfer learning in the setting of an off-the-self Transformer architecture renders new SoTA scores for the argument mining tasks**. This is supported by the scores of 0.924 for components (against 0.908 in previous SoTA), 0.888 for clausal (against 0.849), and 0.783 for relational (against 0.767).

[6] All scores obtained with RoBERTa-base except clausal RTE⇒Cp⇒Cl.

6 Further analysis

No correlation was found between the task scores and the size of their training data. Using the coefficient of determination, .101/.002 and .001 R^2 is obtained for identifying argument components, clausal and relational properties, respectively.

We performed a manual analysis of the output on top-performing tasks in the single-step transfer (CB, RTE, QUAIL). We notice that shorter arguments tend to be incorrectly tagged as *O* (outside) while more extensive arguments tend to be incorrectly divided into two arguments; also, some discourse markers introducing arguments, as "there is clear evidence that" or "thus it is apparent that", tend to be wrongly labelled as the beginning and inside of an argument segment.

Transfer learning experiments on clausal properties follow the same error pattern as the baseline, with most errors emerging from labelling major claims as claims, claims as premises and premises as claims. For relation identification, linked arguments were identified with higher precision and recall than the baseline.

Transferring knowledge from argument mining sources was examined also by extending the language modelling phase. Despite above-baseline scores, no statistical significance was found.[7]

7 Conclusions and future work

The results in this paper were obtained from a large experimental space that permitted a systematic empirical study aimed at assessing the viability of transfer learning to leverage neural argument mining with confluent knowledge. Major findings and results are:

• The knowledge transfer enabled by the transfer learning from language processing tasks that are confluent to argument mining is an effective approach to improve neural argument mining.

• Sequential transfer learning appears as more effective than multi-task transfer, and multi-step sequential transfer can achieve better performance than single-step.

• Source language processing tasks more closely related to argument mining and to the higher-level cognitive capacities mobilized for argumentation tend to provide better support.

[7] More details can be found in Appendix B.

• New state of the art levels of performance were established for the three main sub-tasks in argument mining, namely identification of argument components, classification of components, and determination of the relation among them.

• State of the art was obtained with a lean Transformer-based neural approach that dispensed with heavier feature and model engineering.

• There is much room for further improvements of performance in argument mining given that the new state of the art advanced in the present paper was possible even when deployed on top of just an off-the-shelf Transformer model, viz. RoBERTa,

Concomitantly, these advances open the way to future work. On the side of the mere race for brute force improvement of the state of the art levels of performance, resorting to available Transformer language models that are larger and more powerful than RoBERTa, which was used here, can be easily explored.

On the side of empirically motivated improvements based on more thoughtful approaches, it is possible to explore carefully articulated chains of transfer with curriculum and meta-learning, and also hybrid deep learning and symbolic approaches aimed to solve transfer learning catastrophic forgetting among other issues.

Acknowledgments

The research reported here was supported by the individual PhD grant SFRH/BD/129824/2017 from FCT—Fundação para a Ciência e Tecnologia and it was also supported partially by PORTULAN CLARIN—Research Infrastructure for the Science and Technology of Language, funded by Lisboa 2020, Alentejo 2020 and FCT—Fundação para a Ciência e Tecnologia under the grant PINFRA/22117/2016.

References

Pablo Accuosto and Horacio Saggion. 2019. Transferring knowledge from discourse to arguments: A case study with scientific abstracts. In *Proceedings of the 6th Workshop on Argument Mining*, pages 41–51, Florence, Italy. Association for Computational Linguistics.

Yamen Ajjour, Wei-Fan Chen, Johannes Kiesel, Henning Wachsmuth, and Benno Stein. 2017. Unit Segmentation of Argumentative Texts. In *Proceedings of the 4th Workshop on Argument Mining*, pages 118–128.

Yamen Ajjour, Henning Wachsmuth, Johannes Kiesel, Martin Potthast, Matthias Hagen, and Benno Stein. 2019. Data Acquisition for Argument Search: The args.me corpus. In *42nd German Conference on Artificial Intelligence (KI 2019)*, pages 48–59, Berlin Heidelberg New York. Springer.

Ahmet Aker and Huangpan Zhang. 2017. Projection of argumentative corpora from source to target languages. In *Proceedings of the 4th Workshop on Argument Mining*, pages 67–72.

Khalid Al-Khatib, Henning Wachsmuth, Matthias Hagen, Jonas Köhler, and Benno Stein. 2016. Cross-Domain Mining of Argumentative Text Through Distant Supervision. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL)*, pages 1395–1404.

Florbela Barreto, António Branco, Eduardo Ferreira, Amália Mendes, Maria Fernanda Bacelar do Nascimento, Filipe Nunes, and João Ricardo Silva. 2006. Open resources and tools for the shallow processing of Portuguese: the TagShare project. In *Proceedings of the 5th International Conference on Language Resources and Evaluation (LREC 2006)*, pages 1438–1443.

Chandra Bhagavatula, Ronan Le Bras, Chaitanya Malaviya, Keisuke Sakaguchi, Ari Holtzman, Hannah Rashkin, Doug Downey, Wen-tau Yih, and Yejin Choi. 2020. Abductive commonsense reasoning. In *ICLR*.

Yonatan Bisk, Rowan Zellers, Ronan Le Bras, Jianfeng Gao, and Yejin Choi. 2020. Piqa: Reasoning about physical commonsense in natural language. In *Thirty-Fourth AAAI Conference on Artificial Intelligence*.

Filip Boltužić and Jan Šnajder. 2015. Identifying prominent arguments in online debates using semantic textual similarity. In *Proceedings of the 2nd Workshop on Argumentation Mining*, pages 110–115.

António Branco, Sérgio Castro, João Silva, and Francisco Costa. 2011. CINTIL DepBank handbook: Design options for the representation of grammatical dependencies. Technical Report DI-FCUL-TR-2011-03, University of Lisbon.

António Branco, Francisco Costa, João Silva, Sara Silveira, Sérgio Castro, Mariana Avelãs, Clara Pinto, and João Graça. 2010. Developing a deep linguistic databank supporting a collection of treebanks: the CINTIL DeepGramBank. In *Proceedings of the 7th International Conference on Language Resources and Evaluation (LREC 2010)*.

António Branco and Filipe Nunes. 2012. Verb analysis in a highly inflective language with an MFF algorithm. In *Proceedings of the 11th International Conference on the Computational Processing of Portuguese (PROPOR)*, number 7243 in Lecture Notes in Artificial Intelligence, pages 1–11. Springer.

António Branco, João Rodrigues, João Silva, Francisco Costa, and Rui Vaz. 2014a. Assessing automatic text classification for interactive language learning. In *Proceedings of the IEEE International Conference on Information Society (iSociety)*, pages 72–80.

António Branco and Francisco Costa. 2008. A computational grammar for deep linguistic processing of portuguese: Lxgram.

António Branco and Tiago Henriques. 2003. Aspects of verbal inflection and lemmatization: Generalizations and algorithms. In *Proceedings of XVIII Annual Meeting of the Portuguese Association of Linguistics (APL)*, pages 201–210.

António Branco, Amália Mendes, Paulo Quaresma, Luís Gomes, João Silva, and Andrea Teixeira. 2020. Infrastructure for the science and technology of language PORTULAN CLARIN. In *Proceedings, 1st International Workshop on Language Technology Platforms (IWLTP 2020)*, pages 1–7. European Language Resources Association (ELRA).

António Branco, João Rodrigues, Francisco Costa, João Ricardo Silva, and Rui Vaz. 2014b. Rolling out text categorization for language learning assessment supported by language technology. In *Lecture Notes in Artificial Intelligence, 8775*, pages 256–261.

António Branco and João Silva. 2003. Contractions: Breaking the tokenization-tagging circularity. In *Lecture Notes in Artificial Intelligence 2721*, pages 167–170.

António Branco and João Silva. 2006. A suite of shallow processing tools for Portuguese: LX-Suite. In *Proceedings of the 11th European Chapter of the Association for Computational Linguistics (EACL)*, pages 179–182.

António Branco, João Ricardo Silva, Luís Gomes, and João António Rodrigues. 2022. Universal grammatical dependencies for portuguese with cintil data, lx processing and clarin support. In *Proceedings of the Language Resources and Evaluation Conference*, pages 5617–5626, Marseille, France. European Language Resources Association.

Ruben Branco, António Branco, João António Rodrigues, and João Ricardo Silva. 2021. Shortcutted commonsense: Data spuriousness in deep learning of commonsense reasoning. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 1504–1521, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Elena Cabrio and Serena Villata. 2012. Combining textual entailment and argumentation theory for supporting online debates interactions. In *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 208–212.

Elena Cabrio and Serena Villata. 2013. A natural language bipolar argumentation approach to support users in online debate interactions. *Argument & Computation*, 4(3):209–230.

Rich Caruana. 1997. Multitask Learning. *Machine learning*, 28(1):41–75.

Daniel Cer, Mona Diab, Eneko Agirre, Iñigo Lopez-Gazpio, and Lucia Specia. 2017. SemEval-2017 task 1: Semantic textual similarity multilingual and crosslingual focused evaluation. In *Proceedings of the 11th International Workshop on Semantic Evaluation (SemEval-2017)*, pages 1–14, Vancouver, Canada. Association for Computational Linguistics.

Tuhin Chakrabarty, Christopher Hidey, and Kathleen McKeown. 2019. Imho fine-tuning improves claim detection. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 558–563.

Liying Cheng, Lidong Bing, Qian Yu, Wei Lu, and Luo Si. 2020. Argument pair extraction from peer review and rebuttal via multi-task learning. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 7000–7011.

Artem Chernodub, Oleksiy Oliynyk, Philipp Heidenreich, Alexander Bondarenko, Matthias Hagen, Chris Biemann, and Alexander Panchenko. 2019. Targer: Neural argument mining at your fingertips. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics: System Demonstrations*, pages 195–200.

HongSeok Choi and Hyunju Lee. 2018. GIST at SemEval-2018 Task 12: A Network Transferring Inference Knowledge to Argument Reasoning Comprehension Task. In *Proceedings of The 12th International Workshop on Semantic Evaluation*, pages 773–777.

Christopher Clark, Kenton Lee, Ming-Wei Chang, Tom Kwiatkowski, Michael Collins, and Kristina Toutanova. 2019. BoolQ: Exploring the surprising difficulty of natural yes/no questions. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 2924–2936, Minneapolis, Minnesota. Association for Computational Linguistics.

Peter Clark, Isaac Cowhey, Oren Etzioni, Tushar Khot, Ashish Sabharwal, Carissa Schoenick, and Oyvind Tafjord. 2018. Think you have solved question answering? try arc, the ai2 reasoning challenge. *arXiv preprint arXiv:1803.05457*.

Oana Cocarascu and Francesca Toni. 2017. Identifying attack and support argumentative relations using deep learning. In *Proceedings of the 2017 Conference on*

Empirical Methods in Natural Language Processing, pages 1374–1379.

Alexis Conneau and Douwe Kiela. 2018. Senteval: An evaluation toolkit for universal sentence representations. *arXiv preprint arXiv:1803.05449*.

Francisco Costa and António Branco. 2012. Aspectual type and temporal relation classification. In *Proceedings of the 13th Conference of the European Chapter of the Association for Computational Linguistics (EACL)*, pages 266–275.

André Ferreira Cruz, Gil Rocha, and Henrique Lopes Cardoso. 2018. Exploring Spanish corpora for Portuguese coreference resolution. In *Fifth International Conference on Social Networks Analysis, Management and Security (SNAMS)*, pages 290–295.

Ido Dagan, Oren Glickman, and Bernardo Magnini. 2005. The pascal recognising textual entailment challenge. In *Machine Learning Challenges Workshop*, pages 177–190. Springer.

Yarowsky David, Ngai Grace, Wicentowski Richard, et al. 2001. Inducing multilingual text analysis tools via robust projection across aligned corpora. In *Proceedings of the First International Conference on Human Language Technology Research*, pages 1–8.

Johannes Daxenberger, Steffen Eger, Ivan Habernal, Christian Stab, and Iryna Gurevych. 2017. What is the essence of a claim? cross-domain claim identification. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 2055–2066.

Marie-Catherine De Marneffe, Mandy Simons, and Judith Tonhauser. 2019. The commitmentbank: Investigating projection in naturally occurring discourse. In *proceedings of Sinn und Bedeutung*, volume 23, pages 107–124.

William B. Dolan and Chris Brockett. 2005. Automatically constructing a corpus of sentential paraphrases. In *Proceedings of the Third International Workshop on Paraphrasing (IWP2005)*.

Rotem Dror, Gili Baumer, Segev Shlomov, and Roi Reichart. 2018. The hitchhiker's guide to testing statistical significance in natural language processing. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1383–1392, Melbourne, Australia. Association for Computational Linguistics.

Steffen Eger, Johannes Daxenberger, and Iryna Gurevych. 2017. Neural End-To-End Learning for Computational Argumentation Mining. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (ACL)*, pages 11–22.

Steffen Eger, Johannes Daxenberger, Christian Stab, and Iryna Gurevych. 2018. Cross-lingual argumentation mining: Machine translation (and a bit of projection) is all you need! In *Proceedings of the 27th International Conference on Computational Linguistics*, pages 831–844.

Adam Fisch, Alon Talmor, Robin Jia, Minjoon Seo, Eunsol Choi, and Danqi Chen. 2019. MRQA 2019 shared task: Evaluating generalization in reading comprehension. In *Proceedings of 2nd Machine Reading for Reading Comprehension (MRQA) Workshop at EMNLP*.

Andrea Galassi, Marco Lippi, and Paolo Torroni. 2018. Argumentative link prediction using residual networks and multi-objective learning. In *Proceedings of the 5th Workshop on Argument Mining*, pages 1–10, Brussels, Belgium. Association for Computational Linguistics.

Debela Gemechu and Chris Reed. 2019. Decompositional argument mining: A general purpose approach for argument graph construction. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 516–526.

Luís Gomes, Frederico Apolónia, Ruben Branco, João Ricardo Silva, and António Branco. 2018. Setting up the PORTULAN / CLARIN repository. In *CLARIN Annual Conference (CLARIN2018)*, pages 108–11.

Ivan Habernal, Judith Eckle-Kohler, and Iryna Gurevych. 2014. Argumentation Mining On the Web From Information Seeking Perspective. In *ArgNLP*.

Ivan Habernal and Iryna Gurevych. 2017. Argumentation Mining in User-Generated Web Discourse. *Computational Linguistics*, 43:125–179.

Ivan Habernal, Henning Wachsmuth, Iryna Gurevych, and Benno Stein. 2018. SemEval-2018 task 12: The argument reasoning comprehension task. In *Proceedings of the 12th International Workshop on Semantic Evaluation*, pages 763–772.

Junjie Hu, Sebastian Ruder, Aditya Siddhant, Graham Neubig, Orhan Firat, and Melvin Johnson. 2020. Xtreme: A massively multilingual multi-task benchmark for evaluating cross-lingual generalisation. In *International Conference on Machine Learning*, pages 4411–4421. PMLR.

Lifu Huang, Ronan Le Bras, Chandra Bhagavatula, and Yejin Choi. 2019. Cosmos qa: Machine reading comprehension with contextual commonsense reasoning. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 2391–2401.

Shankar Iyer, Nikhil Dandekar, Kornél Csernai, et al. 2017. First quora dataset release: Question pairs. *data. quora. com.*

Daniel Khashabi, Snigdha Chaturvedi, Michael Roth, Shyam Upadhyay, and Dan Roth. 2018. Looking beyond the surface:a challenge set for reading comprehension over multiple sentences. In *NAACL*.

Tushar Khot, Ashish Sabharwal, and Peter Clark. 2018. Scitail: A textual entailment dataset from science question answering. In *Thirty-Second AAAI Conference on Artificial Intelligence*.

Anne Lauscher, Henning Wachsmuth, Iryna Gurevych, and Goran Glavaš. 2021. Scientia potentia est–on the role of knowledge in computational argumentation. *arXiv preprint arXiv:2107.00281*.

John Lawrence and Chris Reed. 2015. Combining Argument Mining Techniques. In *Proceedings of the 2nd Workshop on Argumentation Mining*, pages 127–136.

John Lawrence and Chris Reed. 2020. Argument mining: A survey. *Computational Linguistics*, 45(4):765–818.

Hector Levesque, Ernest Davis, and Leora Morgenstern. 2012. The winograd schema challenge. In *Thirteenth International Conference on the Principles of Knowledge Representation and Reasoning*.

Kevin Lin, Oyvind Tafjord, Peter Clark, and Matt Gardner. 2019. Reasoning over paragraph effects in situations. In *MRQA@EMNLP*.

Marco Lippi and Paolo Torroni. 2016. Argumentation Mining: State of the Art and Emerging Trends. *ACM Transactions on Internet Technology (TOIT)*, 16(2):10.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Roberta: A robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692*.

Ryan McDonald, Slav Petrov, and Keith Hall. 2011. Multi-source transfer of delexicalized dependency parsers. In *Proceedings of the 2011 Conference on Empirical Methods in Natural Language Processing*, pages 62–72.

Jean-Christophe Mensonides, Sébastien Harispe, Jacky Montmain, and Véronique Thireau. 2019. Automatic detection and classification of argument components using multi-task deep neural network. In *Proceedings of the 3rd International Conference on Natural Language and Speech Processing*, pages 25–33, Trento, Italy. Association for Computational Linguistics.

Hugo Mercier and Dan Sperber. 2017. *The enigma of reason*. Harvard University Press.

Nuno Miranda, Ricardo Raminhos, Pedro Seabra, Joao Sequeira, Teresa Gonçalves, and Paulo Quaresma. 2011. Named entity recognition using machine learning techniques. In *Proceedings of the 15th Portuguese Conference on Artificial Intelligence (EPIA)*, pages 818–831.

Saif Mohammad, Svetlana Kiritchenko, Parinaz Sobhani, Xiaodan Zhu, and Colin Cherry. 2016. SemEval-2016 Task 6: Detecting Stance in Tweets. In *Proceedings of the 10th International Workshop on Semantic Evaluation (SemEval-2016)*, pages 31–41.

Steven Neale, Oier Lopez de Lacalle Luís Gomes, Eneko Agirre, and António Branco. 2016. Word sense-aware machine translation: Including senses as contextual features for improved translation models. In *Proceedings of the 10th Language Resources and Evaluation Conference (LREC2016)*, pages 2777–2783, Portorož, Slovenia. European Language Resources Association.

Huy Nguyen and Diane Litman. 2016. Context-aware argumentative relation mining. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1127–1137, Berlin, Germany. Association for Computational Linguistics.

Vlad Niculae, Joonsuk Park, and Claire Cardie. 2017. Argument mining with structured svms and rnns. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 985–995.

Yixin Nie, Haonan Chen, and Mohit Bansal. 2019. Combining fact extraction and verification with neural semantic matching networks. In *Association for the Advancement of Artificial Intelligence (AAAI)*.

Pedro Javier Ortiz Suárez, Benoît Sagot, and Laurent Romary. 2019. Asynchronous pipelines for processing huge corpora on medium to low resource infrastructures. Proceedings of the Workshop on Challenges in the Management of Large Corpora (CMLC-7) 2019. Cardiff, 22nd July 2019, pages 9 – 16, Mannheim. Leibniz-Institut für Deutsche Sprache.

Andreas Peldszus and Manfred Stede. 2013. From Argument Diagrams to Argumentation Mining in Texts: A Survey. *International Journal of Cognitive Informatics and Natural Intelligence (IJCINI)*, 7(1):1–31.

Andreas Peldszus and Manfred Stede. 2015. Joint Prediction in MST-Style Discourse Parsing for Argumentation Mining. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 938–948.

Georgios Petasis. 2019. Segmentation of argumentative texts with contextualised word representations. In *Proceedings of the 6th Workshop on Argument Mining*, pages 1–10.

Jason Phang, Thibault Févry, and Samuel R Bowman. 2018. Sentence encoders on stilts: Supplementary training on intermediate labeled-data tasks. *arXiv preprint arXiv:1811.01088*.

Jason Phang, Phil Yeres, Jesse Swanson, Haokun Liu, Ian F. Tenney, Phu Mon Htut, Clara Vania, Alex Wang, and Samuel R. Bowman. 2020. `jiant` 2.0: A software toolkit for research on general-purpose text understanding models. `http://jiant.info/`.

Mohammad Taher Pilehvar and Jose Camacho-Collados. 2019. Wic: the word-in-context dataset for evaluating context-sensitive meaning representations. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 1267–1273.

Peter Potash, Alexey Romanov, and Anna Rumshisky. 2017. Here'S My Point: Joint Pointer Architecture for Argument Mining. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1364–1373.

Yada Pruksachatkun, Phil Yeres, Haokun Liu, Jason Phang, Phu Mon Htut, Alex Wang, Ian Tenney, and Samuel Bowman. 2020. jiant: A software toolkit for research on general-purpose text understanding models. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics: System Demonstrations*, pages 109–117.

Lance A Ramshaw and Mitchell P Marcus. 1999. Text chunking using transformation-based learning. In *Natural language processing using very large corpora*, pages 157–176. Springer.

Gil Rocha, Christian Stab, Henrique Lopes Cardoso, and Iryna Gurevych. 2018. Cross-lingual argumentative relation identification: from English to Portuguese. In *Proceedings of the 5th Workshop on Argument Mining*, pages 144–154.

João Rodrigues and António Branco. 2020. Argument identification in a language without labeled data. In *International Conference on Computational Processing of the Portuguese Language*, pages 335–345. Springer.

Joao Rodrigues, Ruben Branco, João Silva, and António Branco. 2020a. Reproduction and revival of the argument reasoning comprehension task. In *Proceedings of the 12th Language Resources and Evaluation Conference*, pages 5055–5064.

João Rodrigues, Francisco Costa, João Silva, and António Branco. 2020b. Automatic syllabification of Portuguese. *Revista da Associação Portuguesa de Linguística*, 1.

João Rodrigues, António Branco, Steven Neale, and João Silva. 2016. LX-DSemVectors: Distributional semantics models for the Portuguese language. In *Proceedings of the 12th International Conference on the Computational Processing of Portuguese (PROPOR'16)*, pages 259–270.

Melissa Roemmele, Cosmin Adrian Bejan, and Andrew S Gordon. 2011. Choice of plausible alternatives: An evaluation of commonsense causal reasoning. In *2011 AAAI Spring Symposium Series*.

Anna Rogers, Olga Kovaleva, Matthew Downey, and Anna Rumshisky. 2020. Getting closer to ai complete question answering: A set of prerequisite real tasks. In *Proceedings of the AAAI conference on artificial intelligence*, volume 34, pages 8722–8731.

Sebastian Ruder. 2019. *Neural transfer learning for natural language processing*. Ph.D. thesis, NUI Galway.

Patrick Saint-Dizier. 2017. Using question-answering techniques to implement a knowledge-driven argument mining approach. In *Proceedings of the 4th Workshop on Argument Mining*, pages 85–90, Copenhagen, Denmark. Association for Computational Linguistics.

Rodrigo Santos, João Silva, António Branco, and Deyi Xiong. 2019. The direct path may not be the best: Portuguese-Chinese neural machine translation. In *Proceedings of the 19th Portuguese Conference on Artificial Intelligence (EPIA)*, pages 757–768.

Maarten Sap, Ronan Le Bras, Emily Allaway, Chandra Bhagavatula, Nicholas Lourie, Hannah Rashkin, Brendan Roof, Noah A Smith, and Yejin Choi. 2019. Atomic: An atlas of machine commonsense for if-then reasoning. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, pages 3027–3035.

Jodi Schneider, Tudor Groza, and Alexandre Passant. 2013. A Review of Argumentation for the Social Semantic Web. *Semantic Web*, 4(2):159–218.

Claudia Schulz, Steffen Eger, Johannes Daxenberger, Tobias Kahse, and Iryna Gurevych. 2018. Multi-Task Learning for Argumentation Mining in Low-Resource Settings. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL)*, pages 35–41.

João Silva, António Branco, Sérgio Castro, and Rúben Reis. 2010. Out-of-the-box robust parsing of Portuguese. In *Lecture Notes in Artificial Intelligence, 6001*, pages 75–85.

Sara Silveira and António Branco. 2012a. Combining a double clustering approach with sentence simplification to produce highly informative multi-document summaries. In *Proceedings, IEEE IRI2012-International Conference on Information Reuse and Integration*, pages 482–489.

Sara Silveira and António Branco. 2012b. Enhancing multi-document summaries with sentence simplification. In *Proceedings, ICAI2012-14th International Conference on Artificial Intelligence*, pages 742–748.

Alfred Sliwa, Yuan Ma, Ruishen Liu, Niravkumar Borad, Seyedeh Ziyaei, Mina Ghobadi, Firas Sabbah, and Ahmet Aker. 2018. Multi-lingual argumentative corpora in english, turkish, greek, albanian, croatian, serbian, macedonian, bulgarian, romanian and arabic. In *Proceedings of the Eleventh International Conference on Language Resources and Evaluation (LREC 2018)*.

Richard Socher, Alex Perelygin, Jean Wu, Jason Chuang, Christopher D Manning, Andrew Y Ng, and Christopher Potts. 2013. Recursive deep models for

semantic compositionality over a sentiment treebank. In *Proceedings of the 2013 conference on empirical methods in natural language processing*, pages 1631–1642.

Marina Sokolova and Guy Lapalme. 2009. A systematic analysis of performance measures for classification tasks. *Information processing & management*, 45(4):427–437.

Maximilian Spliethöver, Jonas Klaff, and Hendrik Heuer. 2019. Is it worth the attention? a comparative evaluation of attention layers for argument unit segmentation. In *Proceedings of the 6th Workshop on Argument Mining*, pages 74–82.

Christian Stab and Iryna Gurevych. 2014. Annotating Argument Components and Relations in Persuasive Essays. In *Proceedings of COLING 2014, the 25th International Conference on Computational Linguistics: Technical Papers*, pages 1501–1510.

Christian Stab and Iryna Gurevych. 2017. Parsing Argumentation Structures in Persuasive Essays. *Computational Linguistics*, 43:619–659.

Christian Stab, Tristan Miller, Benjamin Schiller, Pranav Rai, and Iryna Gurevych. 2018. Cross-topic argument mining from heterogeneous sources using attention-based neural networks. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 3664–3674.

Manfred Stede and Jodi Schneider. 2018. Argumentation Mining. *Synthesis Lectures on Human Language Technologies*, 11(2):1–191.

Alon Talmor, Jonathan Herzig, Nicholas Lourie, and Jonathan Berant. 2019. Commonsenseqa: A question answering challenge targeting commonsense knowledge. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4149–4158.

Frans H Van Eemeren, Rob Grootendorst, and Tjark Kruiger. 2019. *Handbook of argumentation theory*. De Gruyter Mouton.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in neural information processing systems*, pages 5998–6008.

Arlindo Veiga, Sara Candeias, and Fernando Perdigão. 2011. Generating a pronunciation dictionary for European Portuguese using a joint-sequence model with embedded stress assignment. In *Proceedings of the 8th Brazilian Symposium in Information and Human Language Technology*.

Henning Wachsmuth, Nona Naderi, Yufang Hou, Yonatan Bilu, Vinodkumar Prabhakaran, Tim Alberdingk Thijm, Graeme Hirst, and Benno Stein. 2017. Computational Argumentation Quality Assessment in Natural Language. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics (EACL)*, pages 176–187.

Douglas Walton, Christopher W. Tindale, and David Zarefsky. 2005. *Critical Reasoning and Argumentation*. Cambridge University Press.

Alex Wang, Yada Pruksachatkun, Nikita Nangia, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel R Bowman. 2019a. Superglue: a stickier benchmark for general-purpose language understanding systems. In *Proceedings of the 33rd International Conference on Neural Information Processing Systems*, pages 3266–3280.

Alex Wang, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel Bowman. 2018. Glue: A multi-task benchmark and analysis platform for natural language understanding. In *Proceedings of the 2018 EMNLP Workshop BlackboxNLP: Analyzing and Interpreting Neural Networks for NLP*, pages 353–355.

Alex Wang, Ian F. Tenney, Yada Pruksachatkun, Phil Yeres, Jason Phang, Haokun Liu, Phu Mon Htut, , Katherin Yu, Jan Hula, Patrick Xia, Raghu Pappagari, Shuning Jin, R. Thomas McCoy, Roma Patel, Yinghui Huang, Edouard Grave, Najoung Kim, Thibault Févry, Berlin Chen, Nikita Nangia, Anhad Mohananey, Katharina Kann, Shikha Bordia, Nicolas Patry, David Benton, Ellie Pavlick, and Samuel R. Bowman. 2019b. `jiant` 1.3: A software toolkit for research on general-purpose text understanding models. `http://jiant.info/`.

Hao Wang, Zhen Huang, Yong Dou, and Yu Hong. 2020. Argumentation mining on essays at multi scales. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 5480–5493.

Alex Warstadt, Amanpreet Singh, and Samuel R. Bowman. 2019. Neural network acceptability judgments. *Transactions of the Association for Computational Linguistics*, 7:625–641.

Aaron Steven White, Elias Stengel-Eskin, Siddharth Vashishtha, Venkata Subrahmanyan Govindarajan, Dee Ann Reisinger, Tim Vieira, Keisuke Sakaguchi, Sheng Zhang, Francis Ferraro, Rachel Rudinger, et al. 2020. The universal decompositional semantics dataset and decomp toolkit. In *Proceedings of the 12th Language Resources and Evaluation Conference*, pages 5698–5707.

Adina Williams, Nikita Nangia, and Samuel Bowman. 2018. A broad-coverage challenge corpus for sentence understanding through inference. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 1112–1122, New Orleans, Louisiana. Association for Computational Linguistics.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Rémi Louf, Morgan Funtowicz, Joe Davison, Sam Shleifer, Patrick von Platen, Clara Ma, Yacine Jernite, Julien Plu, Canwen Xu, Teven Le Scao, Sylvain Gugger, Mariama Drame, Quentin Lhoest, and Alexander M. Rush. 2020. Transformers: State-of-the-art natural language processing. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 38–45, Online. Association for Computational Linguistics.

Rowan Zellers, Ari Holtzman, Yonatan Bisk, Ali Farhadi, and Yejin Choi. 2019. Hellaswag: Can a machine really finish your sentence? In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 4791–4800.

A AAEC Data set

The AAEC corpus (Stab and Gurevych, 2014, 2017) is a freely available collection of annotated essays extensively adopted by the argument mining community. For the transformation of the original annotations to the different tasks data sets inputs/outputs, we followed the original work (Stab and Gurevych, 2017).

In the literature, the three tasks, argument component identification, clausal properties and relation properties, typically follow the original frame.

Identifying argument components is performed by tagging each token with their tag from the IOB-tagset given a complete essay as an input sequence.

Identifying clausal properties is performed by individually classifying a span of components of an argument with one of the three classes (major claim, claim and premise) given the entire essay in context. In this task, the IOB-tags are not provided, and the span of components consists of raw text. As input to the model, we separated the span of components and the full essay with a separator token, for example, *components_span* <*S*> *essay* </*S*>.

Identifying relational properties, in turn, is performed by individually classifying two components spans as linked or not-linked among themselves, given the entire essay as context. In this task, no IOB-tagset or clausal properties are provided. The spans consist of raw text. As input to the model, we separated the spans with separator tokens, like in the previous sub-task.

The three tasks are handled separately during training. There was no overlap of the test sets with the training or development data sets. In the literature, some papers use the entire essay while some others only the paragraph as context for determining the clausal properties and the relation properties. We followed the typical approach described above for all base models, that is, we used the entire essay as context for RoBERTa-base models and only the paragraph for the RoBERTa-large model, given the large memory footprint and time processing when providing the entire essay to a larger model.

The original AAEC corpus also includes a fourth task, namely, stance recognition, where relational properties are classified with stance attributes (*for* or *against*). In our experiments, we did not perform this fourth task nor used the extra information provided with these stance attributes for the relational properties task.

B Transfer during language modelling

We experimented with transferring knowledge from argument mining related sources by extending the pre-train, language modelling phase, rather than expanding the fine-tuning phase (as in the first and second batch of experiments). We experimented with three argumentation-oriented data sets under the Masked Language Modelling objective: a self-supervised approach was thus adopted, with no further labelled data resorted to during training.

In a first experiment, we extended the model with a train set obtained from the Oscar corpus (Ortiz Suárez et al., 2019) by parsing 1M sentences containing argumentative discourse markers. We extracted all sentences that contained argumentative discourse markers from premise to conclusion and conclusion to premise in an equal distribution.

In a second experiment, we extended the model with an argumentation data set, the Args.me corpus (Ajjour et al., 2019), containing 350k arguments from forum debates. Thirdly, we extended the model with ATOMIC, a common sense knowledge base converted to raw text (Sap et al., 2019) containing 877k inferential relations.

Each model was trained with three randomly initialized runs, for three epochs, with a learning rate of 1e-05 and fine-tuned for each task. The results are in Table 6.

Results: Some performance scores of these models are higher than the respective RoBERTa baseline, also used in the first two batches, however without a statistically significant difference. This

	Components	Clausal	Relational
Baseline	.916	.820	.727
Arg. markers	.908	**.825**	.717
Args.me	.915	.725	**.757**
ATOMIC	**.917**	.787	.716

Table 6: **Performance of models** obtained by further pre-training with data related to argument mining.

may indicate that for this type of approach to leveraging argument mining to be as effective as the approach in the first two batches of experiments, the volume of unlabelled data related to argument mining possibly needs to be higher than the labelled data resorted to there by far more orders of magnitude.

When to Laugh and How Hard?
A Multimodal Approach to Detecting Humor and its Intensity

Khalid Alnajjar[1,2], Mika Hämäläinen[1,2]
Jörg Tiedemann[1], Jorma Laaksonen[3] and **Mikko Kurimo[3]**
[1]University of Helsinki, Finland
[2]Rootroo Oy, Finland
[3]Aalto University, Finland
firstname.lastname@{helsinki.fi[1],aalto.fi[3]}

Abstract

Prerecorded laughter accompanying dialog in comedy TV shows encourages the audience to laugh by clearly marking humorous moments in the show. We present an approach for automatically detecting humor in the *Friends* TV show using multimodal data. Our model is capable of recognizing whether an utterance is humorous or not and assess the intensity of it. We use the prerecorded laughter in the show as annotation as it marks humor and the length of the audience's laughter tells us how funny a given joke is. We evaluate the model on episodes the model has not been exposed to during the training phase. Our results show that the model is capable of correctly detecting whether an utterance is humorous 78% of the time and how long the audience's laughter reaction should last with a mean absolute error of 600 milliseconds.

1 Introduction

Humor is a topic that has piqued interest of the computational creativity research community over the years. There are numerous systems that can generate humor using a variety of different methods (Weller et al., 2020; Tyler et al., 2020; Alnajjar and Hämäläinen, 2021b). But just as important as it is to research generation from the computational creativity perspective, it is to study automatic assessment of humor.

The role of humor is an important one for us humans as it has it's own social function (Ziv, 2010). It helps us talk about difficult topics (Vivona, 2013; Monahan, 2015) and relieves tension (Shurcliff, 1968). Laughter has a role in building relationships (McCabe et al., 2017; Kurtz and Algoe, 2017) and it has a positive effect on brain chemistry (Gonot-Schoupinsky and Garip, 2018). Humor and laughter are therefore an integral part of who we are as a species.

Humor is a phenomenon that requires surprise and coherence (see Hämäläinen and Alnajjar (2019)); or incongruity and its resolution in other terms (Raskin, 1985; Attardo and Raskin, 1991). However, what is surprising or incongruous, depends heavily on the context where humour is presented. Quite indeed, something intended as a joke can be seen as a severe insult just by a change in context. Therefore, we believe that a multimodal approach needs to be researched when when assessing humor automatically; something that thus far has been researched by focusing on the textual modality alone.

Annotated multimodal datasets are scarce, but an access to such a dataset is crucial for any computational attempt on humor detection and assessment. For this reason, we embrace a clever approach: we use episodes of the beloved American sitcom *Friends* as our data source. The TV show has prerecorded audience laughter which provides us with an ultimate source for annotations. Every time the audience laughs, we know that there was something humorous immediately before the laughter. A lack of laughter indicates no humor. In addition to this, the audience can laugh for a short or a long time, which allows us to gather data on how funny a given joke was.

We propose a pipeline consisting of two neural models. One of them detects whether a sentence is humorous or not and the other rates how funny the sentence is in case it was deemed humorous. We evaluate our models on episodes of *Friends* that were not used during training or validation. Our work is a first step towards multimodal humor detection and assessment. We have also established several important data processing practices that make it possible for future research to automatically gather annotated multimodal data in a similar fashion as we did in this paper.

The main contributions of our paper are as follows:

- Methodology for automatically annotating a multimodal humor corpus based on laughter

Proceedings of the 29th International Conference on Computational Linguistics, pages 6875–6886
October 12–17, 2022.

cues.

- A multimodal humor detection model that does not rely on an explicit split in a setup and a punchline.
- A multimodal humor assessment model that can predict how funny a given joke is.

2 Related work

There is a an extensive body of literature that focuses on humor generation (Dybala et al., 2010; Alnajjar and Hämäläinen, 2018; Mishra et al., 2019; Yamane et al., 2021) and also a growing body of work that deals with multimodality in natural language understanding (Soldner et al., 2019; Wang et al., 2020b; Rodríguez Bribiesca et al., 2021). However, in this section, we focus on some of the recent papers that deal with humor detection and analysis.

In a recent work (Xie et al., 2021), the authors study humor detection in a context where there is a setup and a punchline. They use a GPT-2 model (Radford et al., 2019) to assess uncertainty and surprise to determine if a setup-punchline pair is a joke or not. Similar setup and punchline based approaches for humor detection have been widely studied in the past using different computational methodologies (Cattle and Ma, 2016, 2018; Wang et al., 2020a). However, such methods are very different from our approach as we do not expect our model to receive a setup and a punchline that are explicitly marked in the data.

Sentiment analysis has been used in humor detection (Liu et al., 2018). As many of the existing approaches, their approach also operates on a setup and a punchline. The authors found that sentiment conflict and transition between the setup and the punchline are useful in humor detection. The authors use an existing discourse parser (Feng and Hirst, 2012) combined with TexBlob[1] sentiment analysis and heuristic rules to detect humor.

Apart from humor detection, there is a line of work on assessing the humor value of a joke (Weller and Seppi, 2019). The authors propose a model that does not detect humor, as it expects jokes as its input, but instead, the model rates how humorous a given input joke is. Their model also expects a setup and a punchline division in the data. The authors train a transformer (Vaswani et al., 2017) based model for the task.

[1]https://github.com/sloria/textblob

Humorous headlines have been automatically ranked based on how funny they are (Dick et al., 2020). The authors use ridge regression and an LSTM (Long short-term memory) model with manually engineered features such as whether Donald Trump and his hair have been mentioned and the length of the headline. The authors conclude that a language model is simply not enough for assessing humor, but a wider context is needed to help the model understand humor.

As we can see, most of the previous approaches on humor detection and assessment do either or. The models can either tell whether something is funny or not, or rate how funny a given joke is. In addition, many models seem to expect a clear division into a setup and a punchline, which makes it impossible to use them to detect humor in free formed speech or text. In addition, there are many types of humor, for instance sarcastic one (Hämäläinen, 2016), that does not require an explicitly uttered setup in natural language, but rather the setup of the joke can be deduced from the context itself. Our approach tries to tackle these shortcomings in the current state of humor detection.

3 Humor theories

Humor is an integral part of our social lives as humans and because of that, it has provoked the interest of many scholars in the past. Some of the early theories of humor (Hobbes, 1651) saw it as a question of superiority, where a superior person laughs at the misery of those inferior to them. While this explanation might be valid in the context of schadenfreude, more modern takes on humor theory reject it as it cannot explain humor as a whole.

For Koestler (1964), humor is a part of creativity together with other two components: discovery and art. What is seen as characteristic to humor, in his view, in comparison to the other two constituents of creativity, is that its emotional mood is of an aggressive nature. Humor comes from bisociation which is a collision of two frames of reference happening in a comic way.

Raskin (1985) presents a theory that is quite similar to the previously described one in the sense that in order for a linguistic expression to be humorous, it has to be compatible with two different scripts. The different scripts have to somehow oppose one another, for example in the sense that one script is a real situation and the other is hypothetical.

Attardo and Raskin (1991) sees humor to be consisting of six hierarchical resources of knowledge: language, narrative strategy, target, situation, logical mechanism and script opposition. Similarly to the previous theories, the incongruity of two possible interpretations is considered to be an important aspect of humor. An interesting notion that sets this theory apart from others is that of target. According to the authors, it is not uncommon for a joke to have a target, such as an important political person or an ethnic group, to be made fun of.

Two requirements have been suggested in the past as components of humor in jokes: surprise and coherence (see Brownell et al. (1983)). A joke will then consist of a surprising element that will need to be coherent in the context of the joke. This is similar to having two incongruous scripts being simultaneously possible.

Veale (2004) discusses that the theories of Raskin (1985) and Attardo and Raskin (1991) entail that people are forced into resolution of humor. He argues that humor should not be seen as resolution of incompatible scripts, but rather as a collaboration, where the listener willingly accepts the humorous interpretation of the joke. Moreover, he argues that while incongruity contributes to humor, it does not alone constitute it.

4 Data construction

We focus on the sitcom TV show *Friends*. The show is one of the most popular American sitcoms ever produced and it aired from 1994 to 2004. Our data consists of the entire show, i.e. 10 seasons and 236 episodes each of a duration around 20 minutes. All episodes had English well-aligned subtitles that correspond to what is said in the audio track of each episode. We have randomly sampled an episode from each season to assess the quality of the subtitles, and found no major errors or clear delays.

While there are some multimodal annotated data for sarcasm detection (Castro et al., 2019; Alnajjar and Hämäläinen, 2021a), multimodal annotated data of humor in more general terms is very scarce. While several textual humor datasets exist (Hossain et al., 2019; Meaney et al., 2021), they are not suitable for our need as we are interested in multimodality. This is mainly due to the great subjectivity of what humans deem to be funny, and to the high amount of work and funds needed to manually annotate a dataset. To overcome this obstacle, we embrace an automatic approach for annotating humor in the TV show by recognizing laughter in the audience as described in the following subsection.

4.1 Data annotation

After the first few seasons, *Friends* was shot entirely in front of a live audience and a great deal of the laughter in the aired version was original, which would even cause the cast to panic when no laughter is heard while it was expected (Winston, 2021). This makes this show, and other sitcom shows that are shot live a mine of humor annotations that is calling for extraction, given that the laughter is an indication of truly landing jokes rather than being something cued in or added later in the post-editing phase.

Our approach for annotating the show relies on the model proposed by Gillick et al. (2021) for automatically detecting laughter. The model is designed to be robust enough to work on real data and be capable of detecting laughter "in the wild". On a lower level, the model's implementation relies on ResNet (He et al., 2016). The model allow us to indicate the minimum length of the laughter and a cutoff threshold of how confident the model is. We set the minimum laughter length to 0.2 seconds and the threshold to 80% based on our empirical experiments. A shorter length resulted in numerous non-continuous short segments of laughter while a longer length limited the results to a few segments per episode. The case was similar for the confidence threshold. A shorter duration for laughter also lead to many non-laughter noises such as yelling to be detected falsely as laughter.

We ran the model on all the episodes of every season of the entire show and obtained 7422 laughter segments. To construct a dataset for training a supervised neural model for identifying humor, positive and negative samples are required. A crucial aspect of what makes something funny is the context it is present in (Tsakona, 2020), just like how the utterance "You guys just keep getting cooler and cooler" can indicate the opposite of what is expressed and be sarcastic based on the tone of the speaker and what the "guys" have done or said[2]. Furthermore, the context prompting the humorous interpretation can vary in nature, especially that we are dealing with multi-modality. In our case, it

[2]The utterance was said by Chandler sarcastically in the *The One Where the Stripper Cries* episode.

could vary, for instance, based on its length (i.e., how distant is the required knowledge for the act to be deemed funny? A scene, episodes or seasons?), type (i.e., what aspects in the context contribute to the humor? does the humorousness arise from what is said, how it was said or what is done, or a combination?), and familiarity with the characters' personalities and common knowledge of the topic of the discourse.

As it is unfeasible for an unsupervised automated approach to achieve an understanding of the world and, therefore, explain the humor in a given scene and link all contexts in the entire show contributing to it, we resort to defining the context as a fixed duration for the sake of simplicity. For the positive humorous samples, we consider the last 10 seconds of all the laughter segments detected by the model to be the context of the joke. From our experiments, 10 seconds seemed to include some context (e.g., two characters saying two sentences) and not collide with previous laughter segments. An example of such a humorous segment can be seen in Figure 1.

Figure 1: An example of a humorous segment that preceded laughter

To build a set of negative samples, for each positive sample segment, we consider what is prior to it until the laughter segment preceding it to be a non-humorous segment. If possible, we split the segment into 10-second clips and randomly pick 3 clips that has some context. The presence of the context is determined by inspecting the subtitles of the clip. In the case where it was empty, the clip was discarded. This is important to remove segments that have no verbal communication at all such as camera spanning across a scenery in the beginning of a scene. Sometimes, laughter segments are very close to each other that no non-humorous segment exists before a humorous segment.

Sometimes humor was expressed non-verbally in the TV show as seen in Figure 2. We filtered out the cases where the audience laughed and there was no subtitle text before the laughter in the humor segment. We do this because we are focusing on multimodality (text and audio) in humor detection. Such non-verbal humour would require the video to be considered as well.

Figure 2: Joey entering the room wearing many layers of clothes provoked audience laughter non-verbally

By the end of the annotation process, our dataset consisted of a total of 16710 clips where 7422 of them are humorous and the remaining 9288 are non-humorous clips[3]. Furthermore, we indicated in the dataset the length of the laughter segment for all humorous clips. This metadata will be used to predict the intensity of the humor.

4.2 Preprocessing

In our current work, we focus on text and audio in detecting humor, and leave including visual content for future work. This is due to the fact that processing video requires a research on its own right. Video is such an information rich resource with so many potential things one could extract that may or may not be relevant for humor such as facial expressions, body poses, object recognition, action detection and so on. For our text and audio modalities, two types of preprocessing are applied: 1) cleaning and reformatting the subtitles[4] and 2) resampling the frequency of the audio clips.

Dashes, -, at the beginning of subtitle lines were commonly used to imply that the speaker has not changed. We treat such dashes as noise and prune them out. Furthermore, new lines are usually added in subtitles to facilitate reading them and/or to separate talks by different speakers. We substitute new lines with spaces in order to convert the subtitle into complete sentences. Italic tags, "" and "", were stripped out. When inspecting the subtitles, we noticed that a frequent typo of having a capital I instead of an l existed. We addressed

[3]Due copyright we are unable to publicly release the dataset.

[4]This step improved the accuracy of our models by 3%.

this issue by replacing all *I*s with *l*s if they were happened to be in the middle of the word. In case a clip contained multiple subtitle segments across the time-span of the clip, they were all joined together.

In terms of audio processing, we only apply frequency resampling to adjust the frequency to 16kHz. This step is performed to ensure consistency when feeding the data to a neural model.

5 Assessing humor

We present a pipeline of two neural models for assessing humor. The goal of the first model is to identify whether an utterance at the end of the segment is humorous or not given the rest of the segment as context, whereas the second model predicts the intensity of the humor once it has been detected.

5.1 Humor detection

Here, we describe two different models for detecting humor. The first model relies solely on the textual data, and the second is a multimodal model that accepts both textual data and audio signals as input.The task for these models is a binary classification downstream task which is to determine whether the input contains humor or not. We experiment with the two models to gain a better understanding on what the effect of the audio is for humor detection.

We group our dataset by episodes and, then, randomly decide which episodes will be used for training, validation and testing. This division is conducted to prevent the model from getting exposed to shared contexts during the training and testing phases, which would introduce undesired bias. When test data is sampled from completely different episodes than what the training and validation contained, we can ensure that the model learns to detect humor in completely novel contexts rather than detecting merely episode specific recurring jokes. The test dataset is constituted of 25 full episodes, which are 1x09, 2x06, 2x22, 3x13, 3x20, 4x09, 5x04, 5x07, 6x09, 6x11, 6x15, 6x16, 7x09, 7x16, 7x19, 7x20, 7x22, 8x03, 8x14, 8x21, 9x05, 9x11, 9x21, 10x17 and 10x18. In total, the training, validation and testing splits contained 13506, 1477 and 1708 samples, respectively. Both of the models used the same splits to ensure comparable results.

5.1.1 Text only model

We build our text only model by applying transfer-learning and fine-tuning a BERT model (Devlin et al., 2019) using the transformers Python library (Wolf et al., 2020). The pretrained model we used is the uncased English BERT model[5]. For a given input, it is first tokenized using BERT tokenization. If the input contained subtitles from different scenes, they we combined together and separated using the special token "[SEP]".

The architecture of the neural network is composed of the BERT model, a BERT pooler layer, a dropout layer (Srivastava et al., 2014) and a fully connected dense layer that has two outputs. Once the input has passed through the BERT model, the pooler layer returns the last layer hidden-state of the first token of the input sequence. Dropout is applied on the pooler output with a probability of 20% to reduce overfitting. The linear layer is introduced so that the network would learn a way of interpreting the features produced by the past layers and assign a probability score for each of the two labels. In total, the model has 109 million trainable parameters. We utilize Adam optimizer (Kingma and Ba, 2014) with a learning rate of $1e{-}4$, along with the cross-entropy loss function to optimize the neural network. The fine-tuning process was run for 3 complete epochs.

5.1.2 Text and audio model

The multimodal model we propose utilizes the textual and audio input by combining BERT (Devlin et al., 2019) with Facebook's HuBERT[6] (Hsu et al., 2021) neural models. We use the same uncased English BERT model in our multimodal model as we did in our text only model to examine the effect of incorporating audio features for detecting humor. The choice of HuBERT, in contrast to the popular XLSR-Wav2Vec2 (Conneau et al., 2021), is due to its superior or, in worse case scenario, neck-to-neck performance.

Our multimodal model architecture is similar to a siamese neural network architecture in the sense that the output of two models are considered collectively. In our model, one side of the network is dedicated to text and the other to audio. We ensure that both sides produce an equal size of features by 1) setting a fixed input length to BERT where padding and truncating is applied where necessary and 2) having two average pooling layers following the output of each side. For the textual output,

[5] https://huggingface.co/bert-base-uncased

[6] https://huggingface.co/facebook/hubert-xlarge-ls960-ft

Length	0.2-0.5	0.5-1.5	1.5-2.5	2.5-3.5	3.5-4.5	4.5-5.5	5.5-15.5
N	459	2895	2328	948	374	184	234

Table 1: The duration of laughter (in seconds) for different ranges in our dataset

a global average pooling is applied, whereas an adaptive average pooling is applied to the audio output. Afterwards, the pooled output is concatenated and followed by a dropout layer with a probability of 20%. Lastly, a fully connected dense layer is employed as the classification layer. The network has 424 million trainable parameters. We use the same hyperparameters for optimizing the multimodal model as the text only model; in other words, we fine-tuned it for 3 full epochs with a learning rate of $1e{-}4$.

5.2 Predicting laughter intensity

We define the intensity of the laughter based on its duration. Thus, a strong laughter for multiple seconds indicates a great joke. This is intuitive because the funnier the humor gets, the longer the audience laughs. As the duration of a laughter is a continuous value, we treat the task a regression problem and adopt an artificial intelligence neural network for addressing it.

Our dataset for this part is only the humorous clips, a total of 7422 clips. Table 1 shows the number of humorous clips grouped by various durations. We cap all the durations to 3 seconds given that the majority of laughter segments are within this limit. The data is then split for training, validation and testing with 80%, 10%, and 10% ratios, respectively.

In our laughter intensity prediction model, we only train a multimodal model given that *what* is said in the joke and *how* it is said or performed have great influence on the reaction from the audience. Textual features are extracted using BERT like the aforementioned models. For audio features, we extract them from the entire humorous clips using Google's VGGish model[7] (Hershey et al., 2017). VGGish is a Convolutional Neural Network inspired by the VGG network (Simonyan and Zisserman, 2015) that is trained for image classification. However, in VGGish, the input image is the log mel spectrogram of frames derived from the audio. The network has achieved the state-of-the-art results for audio classification given its ability to capture acoustic features, tones, volume and so on. Unlike the earlier models, the pretrained models, i.e. BERT and VGGish, are frozen and not fine-tuned during the training step.

The model we present here is a sequence of layers where the first one is a 2D average pooling layer that converts the extracted features by BERT and VGGish models into a fixed-size set of features by averaging neighbouring features until the desired size is reached. We set the size here to 128 as this is the size of features that the VGGish model returns per frame. As a result, the output of this layer is a vector of 256 features. The layer is then followed by a dense fully-connected linear layer that takes in the averaged features and learns a new representation of 64 features. ReLU activation (Agarap, 2018) and a dropout with a probability of 10% are then subsequently applied to the 64 features. The network architecture ends with a fully-connected dense layer that returns one output representing the intensity score.

As the problem here is regression, we make use of the mean squared error (squared L2 norm) as the loss function. This model has 16.5 thousand trainable parameters and we optimize them for 100 epochs using Adam optimizer, however we use use early stopping to stop the training of the model before 100 epoch in the event of the model converging early.

6 Results and evaluation

We run both, the text only and multimodally trained models, for humor detection on the test split. Their performance is assessed using precision, recall, F1 and accuracy scores, which are given in Table 2. Both of our models outperform the baselines of choosing a label at random or the most frequent label, their accuracies were 51% and 56%, in the order given.

The results indicate that the multimodal model clearly outperformed the text only model, by a 16% increase in detection accuracy. This suggests that audio cues were helpful in recognizing humor. For instance, sarcasm and irony are sometimes marked with clear intonations and tones. Both of these phenomena are frequently used in sitcoms for humorous effect, which would aid the model in distinguishing when "Yeah, right" is meant as a sincere confirmation or as a sarcastic remark for humorous effect.

Recognizing humor is a challenging task, even

[7] https://github.com/harritaylor/torchvggish

	Precision	Recall	F1	N
Text only model				
Funny	0.58	0.52	0.55	758
Not funny	0.65	0.70	0.68	950
Accuracy	62%			
Text + Audio model				
Funny	0.69	0.90	0.78	758
Not funny	0.90	0.68	0.78	950
Accuracy	78%			

Table 2: Accuracy, Precision, recall, and F1 scores of the two models for detecting humor

for humans and it is no surprise that computational models would struggle. As these models are tested on entirely novel contexts (i.e., new discourses that are not covered during the training phase), the performance achieved by them is impressive.

To test the model for predicting laughter intensity, we compute the mean absolute error between the predicted intensity and the intensity of the laughter in the dataset. The average mean absolute error was only around 600 milliseconds. This means that the model can predict how long the audience will laugh after a given joke rather accurately, given that there is some flexibility in the duration of the laughter as it is not an absolute measurement of the humor of a joke. With these results, we can say that the model has learned to predict the intensity of humor in a joke well, given that for jokes provoking less laughter, the model predicts a short laughter, and for jokes provoking a lot of laughter, the model predicts a long laughter. Even though the model is not quite accurate in knowing the exact duration of the laughter.

6.1 Error analysis

When we look at the results of the models, we can see some cases where both of the models failed at predicting the humor accurately. For example, the following dialog provoked a laughter in the audience:

- **Phoebe:** It's amazing. My headache is completely gone. What were those pills called?
- **Monica:** Hexadr*i*n.
- **Phoebe:** I love you Hexadr*i*n.

In this example the humor comes from Phoebe's lack of knowledge of how medicines work as she continues by calling the instructions booklet a story. This is an example of humor that requires some world knowledge and also some understanding of Phoebe's care-free character. Just relying on text in this case or even including the audio does not give the model a context wide enough to reach to a correct interpretation.

In some of these cases where neither of the models predicted the humor right, it is evident that with an access to the video, the model could situate the humor better in the context. The following dialog is an example of such humor:

Figure 3: Joey inquiring whether Monica had cooked a person after tasting her food.

- **Monica:** Remember the guy that gave me a bad review? Well... I'm getting my revenge.
- **Joey:** You cooked him?

In this dialog, Monica is preparing food and letting Joey taste some of it after her line in the dialog as seen in Figure 3. A great part of the joke is in the visual action of Joey tasting Monica's food before asking whether she had cooked the guy who had given a bad review. A model that can take the visual modality into account as well could potentially benefit from the humor intensifying action seen on the video.

When we look at the results where the multimodal model predicted humor right and the text only model predicted it falsely as non-humorous, we can see cases where the speaker's use of their voice gave additional context for the humor. An example of such is in the following dialog:

- **Katie:** You have selected a lot of nice things. So do you uh, want these things delivered Mr. and Mrs. Geller?
- **Rachel:** Oh
- **Ross:** Oh
- **Rachel:** Oh, no, no. No, no.

In this example, the tone of Rachel's voice makes it more evident that Katie was wrong in her assumption that Rachel and Ross were married. The model had learned to capture such a tone of voice as an indication of humor. Another example where the audio is beneficial is the following dialog

- **Waiter:** It's just that we do have some large parties waiting.
- **Phoebe:** Oh, one really does have a stick up one's ass, doesn't one?

In this case Phoebe's line was delivered with a mean and fed up tone, which was helpful for the model in determining humor. Of course, such a tone is not related to humor in every day speech, so this is an indication of a potential bias in the TV show where such a tone is probably used more often to deliver a punchline of a joke than to actually upset the interlocutor. Even though the multimodal model predicted it correctly, including video modality could strengthen the signal to the model because Phoebe had an uncharacteristically nasty facial expression when uttering her line as seen in Figure 4.

Figure 4: Phoebe having a grumpy facial expression while delivering a laughter provoking line.

The both of the models also produce false positives. The following is an example of a dialog that resulted in being falsely labeled as humorous:

- **Monica:** Chandler, don't joke with me. Okay? I'm very, very upset right now.
- **Chandler:** Is this the most upset you could be?
- **Monica:** I think so.

When this dialog is presented without a wider context, it becomes difficult even for a person to know whether it is supposed to be a joke or not. Monica might very well be talking sarcastically, which is a typical type of humor in the corpus, but in this particular case, she is being sincerely upset. Giving the model more context could alleviate this issue, but of course more context might result in more noise because not all contextual information is relevant. The following is another example where both of the models predicted a false positive:

- **Casting agent:** In your love scene with Sarah... she talks about how she's never seen a naked man who wasn't Jewish.

The audience did not laugh after this statement, although in the right context, it might be funny. Here the context was serious. The audience only laughed later on when it became evident that Joey, who was being cast to the movie, did not understand what the agent meant by this utterance. It is clear that what is humorous and not is not always that clear cut especially in a narrow context.

7 Discussion

In this section, we discuss our work and the results obtained. As mentioned earlier, one crucial aspect for understanding humor is the context. We have defined the context in our work as a fixed 10 seconds but, in reality, the context might be wider than that. As another attempt to define context, we have split the TV show based on changes in the visual scene. This approach however was not very practical for our needs as it would mistakenly cut scenes based on changes in the camera angel. A potential solution to overcome the problem of fixed contexts is to observe semantic changes for discovering consecutive scenes that share the same topic. We keep this for future work.

Sitcom TV shows are a great resource for computational humor as there is a multitude of humor forms that they present. For this reason, it is challenging for a neural model to capture all humor forms (e.g., irony, sarcasm, satire, exaggeration, personification, silliness, pun and parody), whether they are expressed verbally or visually. Our multimodal model develops its own understanding of what is funny based on the textual and audial features embedded by BERT and HuBERT. Thus, it is incapable of explaining the humorousness it perceives. In other words, the model can say whether something is funny or not, but it cannot say *why* something is funny. A future direction would be to break down these types of humor and feed them collectively to the model, which would enable it

to recognize humor from different aesthetics and pinpoint the humor type.

From the error analysis, it is evident that including video modality can help the model in understanding humor by situating what is said to the context presented in the video. However, it is not that clear to know which features would be needed. In Figure 3, we could see that it is the action that makes the humor more interpretable, whereas in Figure 4 we saw that the facial expression was revealing of humor. If the humor occurs only in the visual modality as seen early on in Figure 2, it is in the silliness of how Joey looks while wearing all those clothes. Needless to say, a simple automatically extracted vector representation of the video such as video2vec (Hu et al., 2016) is not capable of capturing all these different nuances expressed in the video. It might very well be that an entire TV show does not have enough data for the model to learn to use the video in a meaningful way.

We did try to include video features in our models by obtaining textual descriptions of what is happening in a scene by using an existing state-of-the-art video captioning model (Luo, 2020). The idea was that a captioning model could extract relevant information from the video into a textual format that could then be understood by a language model such as BERT. In practice, this turned out to be an impossible task with the current models and the image datasets they were trained on. Figure 5 shows the poor performance the model had on images from *Friends*. In our experiments, we did not see a single correctly captioned image from the TV show while test data from the dataset the model was trained on produced decent results.

We have also seen that the 10 second contextual window is not always enough to resolve whether something is humorous or not. The issue of increasing the context is that more context will also increase the amount of irrelevant context. We do not believe that simply increasing the context from 10 to 20 seconds, for example, is the most optimal way to go about it because some jokes require more context whereas others do not. Perhaps the best way would be to introduce a third model to the pipeline that is trained to determine how much context is needed by identifying how far back in the dialog we can go and still stay in the same topic. A change in topic would indicate that the context goes too far away from what is needed to interpret the joke.

(a) a man and a woman sitting on a bench

(b) a man and a woman sitting at a table

(c) a man and a woman is sitting on a couch with a dog

(d) a man and a woman playing a video game

Figure 5: Descriptions produced by the image captioning model

8 Conclusions

This work has shown the first steps towards humor interpretation in a multimodal data. Unlike the existing methods, our method does not rely on implicitly marked setup and punchline but can rather detect humor even in cases where the setup of the joke was not made explicit in the text. We have also trained a model that can rank the intensity of humor based on how long the audience laughed. The results are promising and our current research has a lot of potential for future research especially in studying how to deal with video and how much context one should include.

The trained models can be incorporated in other computational creativity models for generating humor. For instance, a system for generating humorous transcripts could utilize our models for determining whether the plot is funny and which version of it would make the audience laugh the most.

An interesting application of our approach would be to pipeline it with a laughter generator. The models presented in this paper could be used to identify where laughter should be inserted and with what intensity in a comedy show that does not have prerecorded laughter. This could save time in post-editing if used in a professional setting.

Because *Friends* has been translated into multiple languages, this makes it possible to rerun our experiments in different languages with a minimal effort. It also creates an ultimate test-bed for multilingual models where we can test whether a model learning humor from the data in all languages can learn a better representation.

References

Abien Fred Agarap. 2018. Deep learning using rectified linear units (relu). *arXiv preprint arXiv:1803.08375*.

Khalid Alnajjar and Mika Hämäläinen. 2018. A master-apprentice approach to automatic creation of culturally satirical movie titles. In *The 11th International Conference on Natural Language Generation*, page 274–283.

Khalid Alnajjar and Mika Hämäläinen. 2021a. ¡Qué maravilla! multimodal sarcasm detection in Spanish: a dataset and a baseline. In *Proceedings of the Third Workshop on Multimodal Artificial Intelligence*, pages 63–68.

Khalid Alnajjar and Mika Hämäläinen. 2021b. When a computer cracks a joke: Automated generation of humorous headlines. In *Proceedings of the 12th International Conference on Computational Creativity (ICCC 2021)*, pages 292–299.

Salvatore Attardo and Victor Raskin. 1991. Script theory revis(it)ed: Joke similarity and joke representation model. *Humor-International Journal of Humor Research*, 4(3-4):293–348.

Hiram H Brownell, Dee Michel, John Powelson, and Howard Gardner. 1983. Surprise but not coherence: Sensitivity to verbal humor in right-hemisphere patients. *Brain and language*, 18(1):20–27.

Santiago Castro, Devamanyu Hazarika, Verónica Pérez-Rosas, Roger Zimmermann, Rada Mihalcea, and Soujanya Poria. 2019. Towards multimodal sarcasm detection (an _Obviously_ perfect paper). In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 4619–4629.

Andrew Cattle and Xiaojuan Ma. 2016. Effects of semantic relatedness between setups and punchlines in Twitter hashtag games. In *Proceedings of the Workshop on Computational Modeling of People's Opinions, Personality, and Emotions in Social Media (PEOPLES)*, pages 70–79.

Andrew Cattle and Xiaojuan Ma. 2018. Recognizing humour using word associations and humour anchor extraction. In *Proceedings of the 27th International Conference on Computational Linguistics*, pages 1849–1858.

Alexis Conneau, Alexei Baevski, Ronan Collobert, Abdelrahman Mohamed, and Michael Auli. 2021. Unsupervised Cross-Lingual Representation Learning for Speech Recognition. In *Proc. Interspeech 2021*, pages 2426–2430.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186.

Anna-Katharina Dick, Charlotte Weirich, and Alla Kutkina. 2020. HumorAAC at SemEval-2020 task 7: Assessing the funniness of edited news headlines through regression and trump mentions. In *Proceedings of the Fourteenth Workshop on Semantic Evaluation*, pages 1019–1025, Barcelona (online).

Pawel Dybala, Michal Ptaszynski, Jacek Maciejewski, Mizuki Takahashi, Rafal Rzepka, and Kenji Araki. 2010. Multiagent system for joke generation: Humor and emotions combined in human-agent conversation. *Journal of Ambient Intelligence and Smart Environments*, 2(1):31–48.

Vanessa Wei Feng and Graeme Hirst. 2012. Text-level discourse parsing with rich linguistic features. In *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 60–68.

Jon Gillick, Wesley Deng, Kimiko Ryokai, and David Bamman. 2021. Robust Laughter Detection in Noisy Environments. In *Proc. Interspeech 2021*, pages 2481–2485.

Freda N Gonot-Schoupinsky and Gulcan Garip. 2018. Laughter and humour interventions for well-being in older adults: A systematic review and intervention classification. *Complementary therapies in medicine*, 38:85–91.

Mika Hämäläinen. 2016. Reconocimiento automático del sarcasmo: ¡esto va a funcionar bien! *Universidad de Helsinki, Departamento de Lenguas Modernas*.

Mika Hämäläinen and Khalid Alnajjar. 2019. Modelling the socialization of creative agents in a master-apprentice setting: The case of movie title puns. In *Proceedings of the 10th International Conference on Computational Creativity*, pages 266–273.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. 2016. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778.

Shawn Hershey, Sourish Chaudhuri, Daniel P. W. Ellis, Jort F. Gemmeke, Aren Jansen, Channing Moore, Manoj Plakal, Devin Platt, Rif A. Saurous, Bryan Seybold, Malcolm Slaney, Ron Weiss, and Kevin Wilson. 2017. Cnn architectures for large-scale audio classification. In *International Conference on Acoustics, Speech and Signal Processing (ICASSP)*.

Thomas Hobbes. 1651. *Leviathan: or, the matter, forme & power of a commonwealth, ecclesiasticall and civill*. University Press.

Nabil Hossain, John Krumm, and Michael Gamon. 2019. "president vows to cut <taxes> hair": Dataset and analysis of creative text editing for humorous headlines. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 133–142.

Wei-Ning Hsu, Benjamin Bolte, Yao-Hung Hubert Tsai, Kushal Lakhotia, Ruslan Salakhutdinov, and Abdelrahman Mohamed. 2021. Hubert: Self-supervised speech representation learning by masked prediction of hidden units. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 29:3451–3460.

Sheng-Hung Hu, Yikang Li, and Baoxin Li. 2016. Video2vec: Learning semantic spatio-temporal embeddings for video representation. In *2016 23rd International Conference on Pattern Recognition (ICPR)*, pages 811–816. IEEE.

Diederik P Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. In *arXiv preprint arXiv:1412.6980*.

Arthur Koestler. 1964. *The act of creation*. London Hutchinson.

Laura E Kurtz and Sara B Algoe. 2017. When sharing a laugh means sharing more: Testing the role of shared laughter on short-term interpersonal consequences. *Journal of Nonverbal Behavior*, 41(1):45–65.

Lizhen Liu, Donghai Zhang, and Wei Song. 2018. Modeling sentiment association in discourse for humor recognition. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 586–591.

Ruotian Luo. 2020. A better variant of self-critical sequence training. *arXiv preprint arXiv:2003.09971*.

Crystal McCabe, Katie Sprute, and Kimber Underdown. 2017. Laughter to learning: How humor can build relationships and increase learning in the online classroom. *Journal of Instructional Research*, 6:4–7.

J. A. Meaney, Steven Wilson, Luis Chiruzzo, Adam Lopez, and Walid Magdy. 2021. SemEval 2021 task 7: HaHackathon, detecting and rating humor and offense. In *Proceedings of the 15th International Workshop on Semantic Evaluation (SemEval-2021)*, pages 105–119.

Abhijit Mishra, Tarun Tater, and Karthik Sankaranarayanan. 2019. A modular architecture for unsupervised sarcasm generation. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 6144–6154.

Kathleen Monahan. 2015. The use of humor, jesting, and playfulness with traumatized elderly. *Social Work in Mental Health*, 13(1):17–29.

Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, Ilya Sutskever, et al. 2019. Language models are unsupervised multitask learners. *OpenAI blog*, 1(8):9.

Victor Raskin. 1985. *Semantic Mechanisms of Humor*. Springer Science & Business Media.

Isaac Rodríguez Bribiesca, Adrián Pastor López Monroy, and Manuel Montes-y Gómez. 2021. Multimodal weighted fusion of transformers for movie genre classification. In *Proceedings of the Third Workshop on Multimodal Artificial Intelligence*, pages 1–5.

Arthur Shurcliff. 1968. Judged humor, arousal, and the relief theory. *Journal of personality and social psychology*, 8(4p1):360.

Karen Simonyan and Andrew Zisserman. 2015. Very deep convolutional networks for large-scale image recognition. In *International Conference on Learning Representations*.

Felix Soldner, Verónica Pérez-Rosas, and Rada Mihalcea. 2019. Box of lies: Multimodal deception detection in dialogues. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 1768–1777.

Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. 2014. Dropout: A simple way to prevent neural networks from overfitting. *Journal of Machine Learning Research*, 15(56):1929–1958.

Villy Tsakona. 2020. *Context in humor research*, pages 7–18. De Gruyter Mouton.

Bradley Tyler, Katherine Wilsdon, and Paul Bodily. 2020. Computational humor: Automated pun generation. In *Eleventh International Conference on Computational Creativity: ICCC'20*.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. *Advances in neural information processing systems*, 30.

Tony Veale. 2004. Incongruity in humor: Root cause or epiphenomenon? *Humor: International Journal of Humor Research*, 17(4):419–428.

Brian D Vivona. 2013. Investigating humor within a context of death and tragedy: The narratives of contrasting realities. *The Qualitative Report*, 18(50):1.

Minghan Wang, Hao Yang, Ying Qin, Shiliang Sun, and Yao Deng. 2020a. Unified humor detection based on sentence-pair augmentation and transfer learning. In *Proceedings of the 22nd Annual Conference of the European Association for Machine Translation*, pages 53–59.

Zilong Wang, Mingjie Zhan, Xuebo Liu, and Ding Liang. 2020b. DocStruct: A multimodal method to extract hierarchy structure in document for general form understanding. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 898–908.

Orion Weller, Nancy Fulda, and Kevin Seppi. 2020. Can humor prediction datasets be used for humor generation? humorous headline generation via style transfer. In *Proceedings of the Second Workshop on Figurative Language Processing*, pages 186–191.

Orion Weller and Kevin Seppi. 2019. Humor detection: A transformer gets the last laugh. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 3621–3625, Hong Kong, China.

Ben Winston. 2021. Friends: The reunion. *HBO Max*.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Remi Louf, Morgan Funtowicz, Joe Davison, Sam Shleifer, Patrick von Platen, Clara Ma, Yacine Jernite, Julien Plu, Canwen Xu, Teven Le Scao, Sylvain Gugger, Mariama Drame, Quentin Lhoest, and Alexander Rush. 2020. Transformers: State-of-the-art natural language processing. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 38–45.

Yubo Xie, Junze Li, and Pearl Pu. 2021. Uncertainty and surprisal jointly deliver the punchline: Exploiting incongruity-based features for humor recognition. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 2: Short Papers)*, pages 33–39.

Hiroaki Yamane, Yusuke Mori, and Tatsuya Harada. 2021. Humor meets morality: Joke generation based on moral judgement. *Information Processing & Management*, 58(3):102520.

Avner Ziv. 2010. The social function of humor in interpersonal relationships. *Society*, 47(1):11–18.

Acknowledgments

This work was partially financed by the Society of Swedish Literature in Finland with funding from Enhancing Conversational AI with Computational Creativity, and by the Ella and Georg Ehrnrooth Foundation for Modelling Conversational Artificial Intelligence with Intent and Creativity.

Modeling Aspect Correlation for Aspect-based Sentiment Analysis via Recurrent Inverse Learning Guidance

Longfeng Li[1], **Haifeng Sun**[1], **Qi Qi**[1], **Jingyu Wang**[1], **Jing Wang**[1]*, **and Jianxin Liao**[1]
[1]State Key Laboratory of Networking and Switching Technology,
Beijing University of Posts and Telecommunications, Beijing 100876, China
{lilongfeng,hfsun,qiqi8266,wangjingyu,wangjing,liaojx}@bupt.edu.cn

Abstract

Aspect-based sentiment analysis (ABSA) aims to distinguish sentiment polarity of every specific aspect in a given sentence. Previous researches have realized the importance of interactive learning with context and aspects. However, these methods are ill-studied to learn complex sentence with multiple aspects due to overlapped polarity feature. And they do not consider the correlation between aspects to distinguish overlapped feature. In order to solve this problem, we propose a new method called Recurrent Inverse Learning Guided Network (RILGNet). Our RILGNet has two points to improve the modeling of aspect correlation and the selecting of aspect feature. First, we use Recurrent Mechanism to improve the joint representation of aspects, which enhances the aspect correlation modeling iteratively. Second, we propose Inverse Learning Guidance to improve the selection of aspect feature by considering aspect correlation, which provides more useful information to determine polarity. Experimental results on SemEval 2014 Datasets demonstrate the effectiveness of RILGNet, and we further prove that RILGNet is state-of-the-art method in multi-aspect scenarios.

1 Introduction

Aspect-based sentiment analysis (ABSA) aims to distinguish the orientation of sentiment existed on every aspect (Liu, 2012). For example, in the sentence: *"Food is usually very good, though I wonder about freshness of raw vegetables"*, aspect *"raw vegetables"* has negative polarity and *"food"* has positive polarity. ABSA task consists of two subtasks called aspect extraction and aspect sentiment classification (Zhang et al., 2018). In this paper, we focus on aspect sentiment classification task and assume that aspects are known (Majumder et al., 2018; Jiang et al., 2020). As shown in above case, multiple sentiment polarities could exist in one sentence, which leads to the overlapping use of multiple aspects when predicting the polarity of current aspect (Du et al., 2019; Jiang et al., 2020). We can see that sentiment polarity of *"food"* comes from word *"good"* which implies positive. *"raw vegetables"* does not have any words linked in above case. However, conjunction *"though"* determines sentiment polarity of *"raw vegetables"* which implies negative. Feature about *"good"* is overlapped by these two aspects and conjunction *"though"* leads to different sentiment polarity between *"food"* and *"raw vegetables"*. In this case, two aspects exhibit high correlation. Therefore, modeling the correlation between multiple aspects is a suitable way to distinguish overlapped feature. From the above case, it is not difficult to see that aspect correlation comes from these sources: aspects, their sentiment polarities and conjunctions. Conjunction establishes correlation between aspects. If model learns correlation between aspects, it can make right predition of *"raw vegetables"* from the feature of *"food"*.

Previous works have been devoted to introduce attention mechanism into neural network and great progress has been made (Wang et al., 2016; Ma et al., 2017). Self-attention-based models such as BERT (Devlin et al., 2019) have been applied to ABSA by using BERT as an encoder (Yu and Jiang, 2019; Lin et al., 2019). To handle the overlapped aspect feature, some works aim at decoupling overlapped feature for every aspect. Du et al. (2019) uses a Capsule Network and Interactive Attention to decouple overlapped feature for predicted aspect. In another way, Majumder et al. (2018) tries to model inter-aspect relation by learning every aspect feature respectively. However, these methods only use aspect (e.g., *"food"*) and not use sentiment polarity (e.g., *"positive"*). Aspect and sentiment polarity both reflect aspect correlation. Considering aspect correlation is an ef-

* Corresponding author.

Proceedings of the 29th International Conference on Computational Linguistics, pages 6887–6896
October 12–17, 2022.

fective way to determine the polarity of aspect that is not easy to find linked words. We find that inverse learning (Zhou and Small, 2021; Putzky and Welling, 2017) is a suitable way to excavate the correlation from aspect and sentiment polarity because it learns from input (aspect and sentiment polarity) to output (generated sentence). And generated sentence is same with aspect and sentiment polarity (Radford et al., 2018; Luong et al., 2015) in correlation.

In this paper, we design a Recurrent Inverse Learning Guided Network (RILGNet) to improve the performance of ABSA task through modeling aspect correlation. Our RILGNet has two point to improve aspect correlation modeling and aspect feature selecting. First, we use Recurrent Mechanism (Kolluru et al., 2020) to learn aspect joint representation and model aspect correlation. And we use the correlation and representation to make current prediction. Recurrent Mechanism can find clear stated aspect polarities (*"food"* and positive in above sentence), and then use them to re-find indirect aspect polarities (*"raw vegetables"* and negative). For this purpose, Recurrent Mechanism is a suitable way. Second, we use Inverse Learning Guidance (ILG) to excavate the correlation from aspects and sentiment polarities. We use inverse learning to generate a sentence by using aspect-polarity combination. We consider the difference between the given sentence and the generated sentence embedding by calculating mean squared error (MSE) and its gradient. This gradient expresses difference in correlation between the prediction and the ground-truth. We fuse this gradient to aspect joint representation, which makes the selection of aspect feature focus more on high correlation aspects. We show that RILGNet achieves 83.70% and outperforms state-of-the-art method by 1.34% on average in SemEval 2014 two distinct domains: laptop and restaurant. And we also prove that RILGNet is state-of-the-art method in multi-aspect scenarios.

The main contributions are as follows: (1) RILGNet improves the performance of ABSA task by modeling aspect correlation from aspect and sentiment polarity. (2) We design an ILG to get guidance about how aspect joint representations are modified and how features are selected from current prediction. (3) Experimental results show that our method outperforms state-of-the-art method on SemEval 2014 Datasets, especially in multi-aspect scenarios.

2 Related Works

The traditional methods of ABSA find optional words about aspect in sentence, and then get the corresponding sentiment through special designed aspect features such as sentiment lexicon, n-grams, and dependency information, which is labor-intensive (Boiy and Moens, 2009; Kiritchenko et al., 2014). Meanwhile, the quality of feature directly affects classification accuracy.

Motivated by attention mechanism in the field of deep learning, some ABSA's works focus on integrating attention into neural network, such as RNNs (Wang et al., 2016; Chen et al., 2017) and memory networks (Ma et al., 2017; Majumder et al., 2018), to learn attention distribution of different aspects and extract aspect-aware sentence representation. Most recently, self-attention-based models such as BERT (Devlin et al., 2019) have been applied to ABSA by using BERT as the embedding layer (Yu and Jiang, 2019; Lin et al., 2019). These works effort to focus on understanding the context based on aspect and have achieved great performance improvement.

Besides the above works, Chen et al. (2020); Zhang and Qian (2020); Tay et al. (2018); Li et al. (2019); Mao et al. (2019) use syntactic information and improved semantic understanding to improve end-to-end performance. Graph-based models are applied in ABSA task by utilizing word relations through dependency parses to facilitate the task with better semantic guidance for analyzing context and aspect words (Tian et al., 2021; Wang et al., 2020; Li et al., 2021; Hou et al., 2021). They realize that the overlapped feature about multi-aspect is ignored when the model predicts current aspect. Majumder et al. (2018) calculates the aspect-aware sentence representation for every aspect in a sentence, which models the inter-aspect relation in a sentence for handling the overlapped feature from aspect information. Du et al. (2019) uses a Capsule Network and Interactive Attention to decouple overlapped feature to predicted aspect. Jiang et al. (2020) uses mutual enhanced transformation to obtain more effective representation of aspect and context for handling overlapped feature. However, unlike our proposed method, they only use aspect information without sentiment polarity information. Aspect and sentiment polarity both reflect aspect correlation.

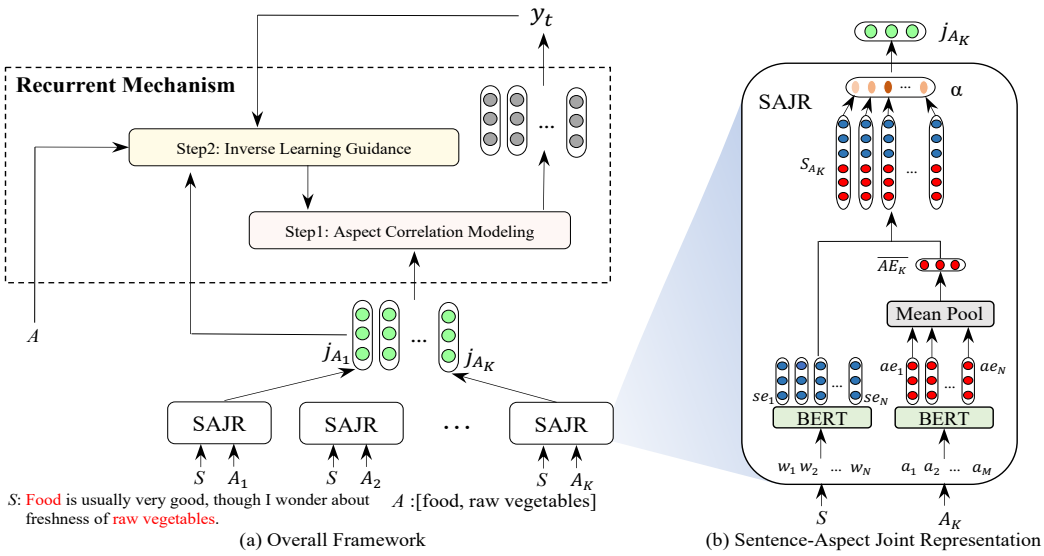

Figure 1: The overall framework of RILGNet and Sentence-Aspect Joint Representation. When the recurrent round is 1, the output of Inverse Learning Guidance is 0.

3 Proposed Method

In this section, we formalize the task and describe proposed Recurrent Inverse Learning Guided Network (RILGNet) in detail. The input and output of ABSA task are as follows:

Input : We give a sentence $S = [w_1, w_2, ..., w_N]$, known aspect set $A = [A_1, A_2, ..., A_K]$ and every aspect $A_i = [a_1, a_2, ..., a_M]$, where *N* and *M* are the maximum number of sentence and aspect word respectively, *K* is the maximum number of aspect.

Output : Sentiment Classification Result $y \in \{positive, negative, neutral\}$ for each predicted aspect A_i.

The framework of RILGNet is shown in Figure 1 (a). It can be roughly divided into two parts, namely Sentence-Aspect Joint Representation (SAJR) and Recurrent Mechanism. Recurrent Mechanism can be further divided into two steps: Aspect Correlation Modeling and Inverse Learning Guidance. The core idea is how to model aspect correlation in *A* about A_i and how to use correlation to discriminate overlapped feature and determine sentiment polarity y. In the t_{th} recurrent round, RILGNet uses sentence S and aspect set A to obtain the SAJR $J_A = [j_{A_1}, j_{A_2}, ..., j_{A_K}]$. Then RILGNet sends J_A and A to Recurrent Mechanism and get current prediction y_t. We choose y_T as final prediction.

3.1 Sentence-Aspect Joint Representation

Different words in a sentence don't contribute equally to sentimental information for a current aspect (e.g., adjectives often contribute more than verbs). This requires a sentence representation to reflect the sentiment of different aspect. As shown in Figure 1 (b), we design Sentence-Aspect Joint Representation (SAJR) to achieve this. We get sentence representation $SE = [se_1, se_2, ..., se_N] \in \mathbb{R}^{N \times D}$ from sentence *S* and aspect representation $AE_i = [ae_{1,i}, ae_{2,i}, ..., ae_{M,i}]$ from aspect A_i by BERT (Devlin et al., 2019) firstly, where *D* is the size of hidden state in BERT. For multi-word aspect, $\overline{AE_i}$ is averaged by AE_i. Then we concatenate *SE* and $\overline{AE_i}$ in all words for $S_{A_i} \in \mathbb{R}^{N \times 2D}$:

$$S_{A_i} = [se_1; \overline{AE_i}, se_2; \overline{AE_i}, ...se_N; \overline{AE_i}] \quad (1)$$

In order to differentiate which word contributes more to the sentiment, we utilize an attention layer to obtain sentence-aspect joint representation $j_{A_i} \in \mathbb{R}^D$, indicating which word is important for A_i:

$$z = S_{A_i} W_s + b_s \quad (2)$$

$$\alpha = softmax(z) \quad (3)$$

$$j_{A_i} = Dense(\alpha^T S_{A_i}) \quad (4)$$

where $z = [z_1, z_2, ..., z_N] \in \mathbb{R}^N$, $\alpha \in \mathbb{R}^N$, W_s and b_s are trainable matrices. $j_{A_i} \in \mathbb{R}^D$ and $J_A = [j_{A_1}, j_{A_2}, ..., j_{A_K}]$.

3.2 Aspect Correlation Modeling

We design Aspect Correlation Modeling (ACM) to learn aspect joint representation, then use it to

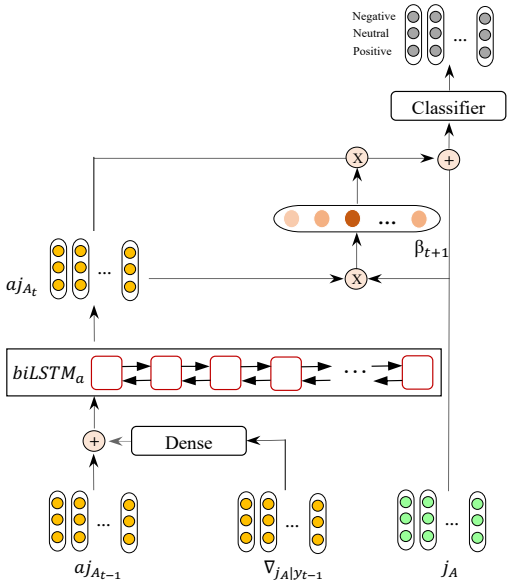

Figure 2: The overall framework of Aspect Correlation Modeling. When recurrent round is 1, $\bigtriangledown_{J_A|y_0}$ is 0.

model aspect correlation and get current prediction y_t. Specially, as shown in Figure 2, we use a bi-LSTM to get current aspect joint representation $AJ_{A,t} = [aj_{A_{1,t}}, aj_{A_{2,t}}, ..., aj_{A_{K,t}}] \in \mathbb{R}^{N \times D}$ at t_{th} recurrent round following Majumder et al. (2018):

$$IJR_{t-1} = tanh(\bigtriangledown_{J_A|y_{t-1}} W_y + b_y) \quad (5)$$

$$AJ_{A,t} = biLSTM_a(AJ_{A,t-1} + IJR_{t-1}) \quad (6)$$

where $AJ_{A,0}$ is J_A from Equation (4) and $\bigtriangledown_{J_A|y_0}$ is 0, W_y and b_y are trainable matrices, $\bigtriangledown_{J_A|y_{t-1}}$ is inverse learning guided gradient from (12). We will introduce this gradient in the next section.

After this, in order to get aspect correlation between A_i and other aspects, we employ an attention layer between every aspect query representation j_{A_i} and $AJ_{A,t}$ as:

$$z_c = j_{A_i}(AJ_{A,t})^T \quad (7)$$

$$\beta_t = softmax(z_c) \quad (8)$$

where $z_c = [z_{c,1}, z_{c,2}, ..., z_{c,K}] \in \mathbb{R}^{K \times 1}$, $\beta_t = [\beta_{t,1}, \beta_{t,2}, ..., \beta_{t,K}] \in \mathbb{R}^{K \times 1}$ and j_{A_i} comes from (4). Here, β_t measures the aspect correlation, stating the contribution of A to predicting A_i. It is worth noting that correlation score β_t does not indicate positive or negative correlation between different aspects, but only indicates how siginificant correlation is achieved.

Then, we obtain the aspect joint representative output vector $output_t \in \mathbb{R}^{1 \times D}$ by:

$$output_t = (\beta_t)^T AJ_{A,t} \quad (9)$$

Finally, aspect query representation j_{A_i} is added with $output_t$ to generate refined aspect representation, and is passed to a softmax classifier as:

$$oj_{i,t} = softmax((j_{A_i} + output_t)W_p + b_p) \quad (10)$$

$$y_{i,t} = argmax(oj_{i,t}) \quad (11)$$

where W_p and b_p are trainable matrices. $y_{i,t}$ is current prediction for A_i, and we concatenate all $y_{i,t}$ for every aspect A_i to get current prediction y_t.

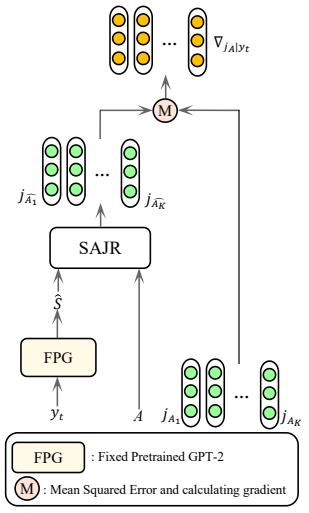

Figure 3: The overall framework of Inverse Learning Guidance.

3.3 Inverse Learning Guidance

The motivation of Inverse Learning Guidance (ILG) is to provide prompt to model aspect correlation by excavating aspect and sentiment polarity. This prompt can give feedback on how aspect correlation is disturbed from current prediction y_t. We use the idea of inverse learning (Zhou and Small, 2021; Putzky and Welling, 2017) to correct the disturbed aspect correlation. It can set aspect-polarity combination as input to generate sentence $\hat{S}$ which is same with y_t about aspect correlation. The difference between $\hat{S}$ and given sentence S is an embodiment for the difference between y_t and the ground-truth $\hat{y}$ in aspect correlation.

The inverse learning model is implemented by GPT-2, which has shown effective at generating

the textual language based on specific information(Radford et al., 2018). If GPT-2 can generate a sentence that is more similar to the given sentence in semantics, prediction y_t will be closer to the ground-truth $\hat{y}$ and y_t will have more similar aspect correlation with $\hat{y}$. Conversely, when the generated sentence is quite different from the given sentence, despite some noise, ILG can reflect the difference by high or low gradient because of the inconsistent aspect-polarity combinations. Gradient guides the selection of features to focus more on high correlation aspect. GPT-2 is finetuned by inverting the input and output of training data. The parameters of GPT-2 will be fixed during training process. The details of the structure and training for GPT-2 is in Section 4.2.

The structure of ILG is shown in Figure 3. Specifically, we use a template like *"[A_1: polarity, A_2: polarity, ..., A_K: polarity.]"* to generate a sentence $\hat{S}$, and polarities come from y_t. We then use the $\hat{S}$ and A to get $\hat{J_A} = [\hat{jA_1}, \hat{jA_2}, ..., \hat{jA_K}] \in \mathbb{R}^{K \times D}$. We select the mean squared error (MSE) to reflect aspect-level difference between $\hat{J_A}$ and J_A about aspect correlation. And we calculate its gradient as output of this step, which is defined as $\bigtriangledown_{J_A|y_{t-1}} \in \mathbb{R}^{K \times D}$:

$$\bigtriangledown_{J_A|y_{t-1}} = \nabla(\frac{1}{D}\sum_{d=1}^{D}(\hat{J_{A,d}} - J_{A,d})^2) \quad (12)$$

where D is the dimension of sentence embedding. $\bigtriangledown_{J_A|y_{t-1}}$ expresses the difference about correlation between S and $\hat{S}$. We fuse $\bigtriangledown_{J_A|y_{t-1}}$ to ACM in (5) and (6), which makes the selection of aspect features focus more on high correlation aspects.

3.4 Training

We train RILGNet using Cross Entropy as loss function at every recurrent round and averaging them as:

$$L = -\frac{1}{NT}\sum_{t=1}^{T}\sum_{n=1}^{N}\hat{y_n}logy_{nt} \quad (13)$$

where $\hat{y_n}$ is the ground-truth, y_{nt} is the prediction in round t, N is the number of samples and T is the maximum recurrent round.

4 Experiment

4.1 Datasets

We evaluate our model with SemEval-2014 ABSA dataset [1] (Pontiki et al., 2014). It contains two domain datasets: Laptop and Restaurant. Table 2 shows the count of samples in total and multi-aspect sentences.

4.2 Implements

BERT: BERT Encoder employs a pre-trained BERT-based model ("BERT-base-uncased") for fine-tuning (Devlin et al., 2019). The hidden state size is 768 and the number of self-attention layers is 12. The total number of parameters for the pre-trained BERT model is 110M in our experiments. The length of word list is 30,522. The max sequence length is 100. We used the last hidden state of BERT for word representation and fine-tuned them on our method. The batch size is 128 in training and 64 in testing. RILGNet is trained by the Adam optimizer (Kingma and Ba, 2015) with the default configuration with 20 epochs. The learning rate is 5×10^{-5} in RILGNet.

GPT-2: In the ILG part, we use pre-trained GPT-2 (Radford et al., 2018) to generate a sentence. We finetune GPT-2 by using a sentence as *"[A_1: polarity, A_2: polarity, ..., A_K: polarity.]"* to get generated sentence. A_i and their polarities come from ground truth. Finally, we use the Corss Entropy as loss function by predicting the next word. We train our model in 20 epochs with batch size 4, and the learning rate is 1.5×10^{-4} with same Adam optimizer in other parts of RILGNet. The hidden state size is 768, the number of self-attention heads is 12. The max sequence length is 100, which is same with BERT. And we fix the parameter of GPT-2 when we train the other side. We use Top-5 Sampling to get next candidate word when generating sentence. We choose next word in their probabilities on these five words.

Aspect Correlation Modeling: In this step, the hidden state size of the bi-LSTM is 768 with one layer. The classifier is a Fully Connected Layer, and the output size is the known aspect set size $\times$ 3(*Positive, Negative, Neutral*).

4.3 Compared Methods

We compare our method with the following baseline methods:

[1] http://alt.qcri.org/semeval2014/task4

Model	Laptop		Restaurant	
	Accuracy	Macro-F1	Accuracy	Macro-F1
SVM (Kiritchenko et al., 2014)	70.49	-	80.16	-
AE-LSTM (Wang et al., 2016)	68.90	-	76.60	-
ATAE-LSTM (Wang et al., 2016)	68.81	63.11	77.68	64.89
IAN (Ma et al., 2017)	72.57	66.73	78.48	67.55
Cabasc (Liu et al., 2018)	75.07	-	80.89	-
IARM (Majumder et al., 2018)	73.80	-	80.00	-
TNet-LF (Li et al., 2018)	75.08	69.78	80.18	70.06
HGMN (Ran et al., 2019)	76.67	72.22	82.33	73.34
IACapsNet (Du et al., 2019)	76.80	73.29	81.79	73.40
METNet (Jiang et al., 2020)	78.37	74.93	82.50	73.92
DualGCN+BERT (Li et al., 2021)	81.80	78.10	87.13	81.16
GraphMerge (Hou et al., 2021)	81.35	78.65	87.32	81.95
RILGNet w/o Inverse Learning Guidance	81.50	79.02	86.79	80.51
RILGNet	**83.70**	**80.13**	**88.13**	**82.66**

Table 1: The performance of RILGNet on SemEval-2014 ABSA dataset with best performances bolded. "-" indicates not reported in the original paper.

Domain	Train		Test	
	Total	MA	Total	MA
Laptop	2328	1396	638	379
Restaurant	3608	2595	1120	835

Table 2: The number of samples in SemEval-2014 ABSA datatsets. MA means the multi-aspects sentence in datasets.

SVM (Kiritchenko et al., 2014): It is a traditional support vector machine (SVM) based model to get the feature about aspect and sentiment.

AE-LSTM (Wang et al., 2016): AE-LSTM is an attention-based LSTM model, which uses attention mechanism to get the correlation between apsect and words in sentence.

ATAE-LSTM (Wang et al., 2016): ATAE-LSTM is an upgraded version about AE-LSTM. It adds the aspect embedding to the model in order to consider the importance of aspect term to sentiment.

IAN (Ma et al., 2017): IAN interactively learns the context and aspect and extract the feature about sentence and aspect separately.

Cabasc (Liu et al., 2018): Cabasc employs two attention enhancement layers to learn the word order information, aspect information and the correlation between aspect and word in sentence.

IARM (Majumder et al., 2018): IARM obtians aspect-aware sentence representations for all aspects in a sentence to predict the sentiment polarity of current aspect.

TNet-LF (Li et al., 2018): TNet-LF computes the importance of each aspect term based on sentence word rather than the whole sentence dynamically.

HGMN (Ran et al., 2019): HGMN gets the aspect-specific text spans in sentence instead of only the aggregated contextual representation based on attention layer.

IACapsNet (Du et al., 2019): IACapsNet uses the Capsule Network and Interactive Attention to decouple the overlapped feature about multi aspects in a sentence.

METNet (Jiang et al., 2020): METNet uses the mutual enhanced transformation to improve the learning of aspect and fuses the aspect and sentence information by gated convolutional network.

DualGCN+BERT (Li et al., 2021): It uses a dual graph convolutional network that considers the complementarity of syntax structures and semantic correlations simultaneously.

GraphMerge (Hou et al., 2021): It uses a graph ensemble technique, to make use of the predictions from different parsers, combining the dependency relations from different parses.

5 Results

We show the results about different experiments below:

Model	$Laptop^{\#}$		$Restaurant^{\#}$	
	Accuracy	Macro-F1	Accuracy	Macro-F1
TNet-LF (Li et al., 2018)	74.80	67.34	80.31	69.35
IARM (Majumder et al., 2018)	74.10	-	80.48	-
METNet (Jiang et al., 2020)	77.95	73.80	82.59	73.92
RILGNet w/o Inverse Learning Guidance	79.15	77.81	86.59	80.33
RILGNet	**81.27**	**78.72**	**88.02**	**82.61**

Table 3: The performance of RILGNet on $Laptop^{\#}$ and $Restaurant^{\#}$ with best performances bolded. "-" indicates not reported in the original paper.

5.1 Overall Comparison

We assess the overall performance by comparing RILGNet with state-of-the-art methods on SemEval-2014 ABSA dataset. It is evident from the results that RILGNet outperforms all the baselines in Table 1 in both of the domains. GraphMerge (Hou et al., 2021) achieves the best results on Laptop Marco-F1 and Restaurant all metrics due to its progressive graph ensemble technique, making use of the predictions from different parsers and combining the dependency relations from different parses. RILGNet gets bigger improvement in laptop domain (2.35% on Accuaracy and 1.48% on Marco-F1), compared to restaurant domain (0.81% on Accuracy and 0.71% on Marco-F1). This shows that the conbination of aspect correlation modeling and inverse learning guidance in Recurrent Mechanism has overall positive influence on sentiment classification.

In addition, in order to highlight the advantage of RILGNet in multi-aspect scenarios about handling overlapped feature, we conduct further experiments. We delete the single-aspect sentences in two domain test set and sign the new datasets as $Laptop^{\#}$ and $Restaurant^{\#}$ same with Jiang et al. (2020). We report the results of TNet-LF, IARM, METNet and our RILGNet in Table 3. RILGNet achieves significant improvements in two domains, which indicates the effectiveness of RILGNet in multi-aspect scenarios.

5.2 Ablation Study

We evaluate that Inverse Learning Guidance is effective for ABSA task by comparing RILGNet and RILGNet without(w/o) Inverse Learning Guidance. In order to prove above point, we conduct ablation experiments. As shown in Table 1, Compared with RILGNet w/o Inverse Learning Guidance, RILGNet achieves 2.2%, 1.34% in Accuracy, and 1.11%, 2.15% in Marco-F1 improvements on Laptop and Restaurant datasets respectively, which proves the effectiveness of the Inverse Learning Guidance.

The number of recurrent round	Laptop		Restaurant	
	Acc	F1	Acc	F1
1	82.75	78.83	85.53	78.76
2	81.81	76.62	85.89	79.29
3	**83.70**	**80.13**	86.79	80.55
4	81.97	77.43	87.77	82.06
5	81.97	77.43	**88.13**	**82.66**
6	82.44	78.41	86.52	80.00
7	81.50	76.51	85.98	79.91

Table 4: The performance of RILGNet about differnent recurrent round from 1 to 7 on Laptop and Restaurant datasets with best performances bolded.

We also conduct ablation experiments in multi-aspect scenarios. In Table 3, RILGNet achieves 2.12% and 1.43% in Accuracy and 0.91% and 2.28% in Marco-F1 improvements on $Laptop^{\#}$ and $Restaurant^{\#}$ datasets compared with RILGNet w/o Inverse Learning Guidance, which effectively proves the importance of Inverse Learning Guidance in multi-aspect scenarios.

Besides, in order to highlight the advantage of our Recurrent Mechanism, we evaluate the effect of recurrent round number T. Specifically, we conduct the experiments on Laptop and Restaurant datasets and T varies from 1 to 7. In Table 4, we can see that RILGNet achieves the best results when T is 3 in Laptop and T is 5 in Restaurant respectively. We find that irregular recurrent performance showns on RILGNet in restaurant. Certain recurrent round could yield higher quality representation than others, which is more stable in laptop.

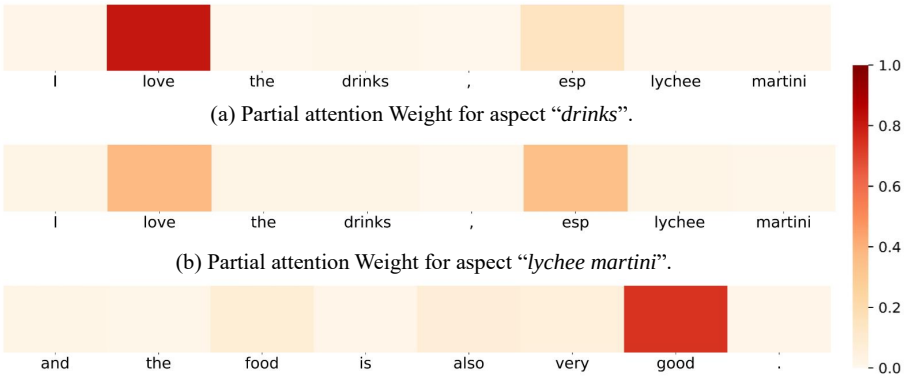

(a) Partial attention Weight for aspect "*drinks*".

(b) Partial attention Weight for aspect "*lychee martini*".

(c) Partial attention Weight for aspect "*food*".

Figure 4: The partial attention weight of all aspect for sentence *"I love the drinks, esp lychee martini, and the food is also very good."* because of too long sentence. The other part has no higher attenion score than this part.

5.3 Case Study

In order to evaluate how aspect correlation is learned by RILGNet, we select one sample from test set to visualize the correlation score β from Equation (8) and attention score α from (3) about best trained RILGNet. It is worth noting that correlation score β does not indicate positive or negative correlation between different aspects, but only indicates how siginificant correlation is achieved.

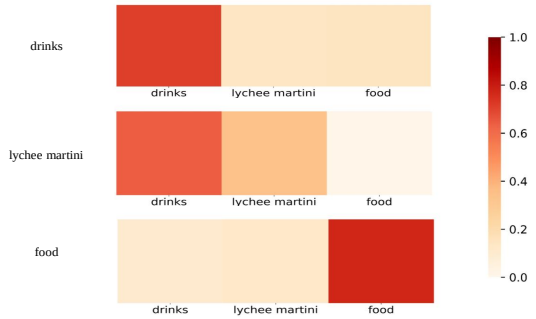

Figure 5: The correlation score between three aspects from the sentence *"I love the drinks, esp lychee martini, and the food is also very good."*

In the sentence *"I love the drinks, esp lychee martini, and the food is also very good."*, there are three aspect: *"drinks"*, *"lychee martini"* and *"food"* with positive polarities, and RILGNet makes all right prediction. *"lychee martini"* has high correlation with *"drinks"* because of conjunction *"esp"*.

As shown in Figure 4, the aspect joint representation of *"drinks"* from SAJR pays more attention to the word *"love"*, *"food"* pays more to the word *"good"* and *"lychee martini"* doesn't obviously pay high attention to optional words, which means that model doesn't know the sentiment of *"lychee martini"* only by SAJR. But model makes right prediction on *"lychee martini"*. We consider that other parts of RILGNet models aspect correlation between these three aspects by handling overlapped feature of word *"good"*.

To evaluate aspect correlation modeling, we show the correlation score β from (8). As shown in Figure 5, the top expresses correlation score between *"drinks"* and others, middle expresses *"lychee martini"* and low expresses *"food"*. Aspect *"lychee martini"* has high β with *"drinks"* in middle. From Equation (9), we can see that RILGNet knows that the feature of *"drinks"* is more important than others when model determines the polarity of *"lychee martini"*. β about *"food"* and *"drinks"* is high with itself. There is no doubt that our RILGNet can model correlation between these aspects: there is a siginificant correlation between *"drinks"* and *"lychee martini"*, and correlation between *"food"* and others is weak.

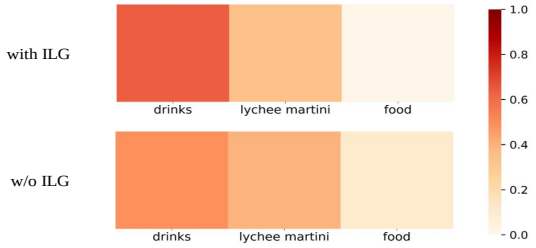

Figure 6: The correlation score of *"lychee martini"* on RILGNet and RILGNet w/o ILG.

Besides, we also show correlation scores between RILGNet and RILGNet w/o ILG of *"lychee martini"* in Figure 6. We can see that RILGNet w/o ILG gets highest score in *"drinks"*. But it also considers some information from *"lychee martini"* and *"food"*, which is not useful for sentiment clas-

sification. As a contrast, RILGNet has higher score than RILGNet w/o ILG in *"drinks"*, meaning that ILG could make the selection of feature focus more on high correlation aspects (eg,. *"drinks"*). To sum up, these results prove that RILGNet could model aspect correlation and ILG could make the selection of feature focus more on high correlation aspect with the guidance of aspect correlation.

6 Conclusion

In this paper, we propose a Recurrent Inverse Learning Guidance Network (RILGNet) to solve ABSA task by modeling aspect correlation. Firstly, in order to solve the problem of overlapped aspect feature, we design a Recurrent Meshanism to learn aspect joint representation and model aspect correlation. Secondly, RILGNet uses Inverse Learning Guidance to give feedback on how current predition reflect aspect correlation by gradient iteratively. Experimental results demonstrate the effectiveness of RILGNet for ABSA. And RILGNet also performs well in multi-aspect scenarios especially. Above all, modeling aspect correlation or label correlation will be useful for other NLP tasks.

Acknowledgement

This work was supported in part by the National Natural Science Foundation of China under Grants (62201072, 62071067, 62171057, 62101064, 62001054), in part by the Ministry of Education and China Mobile Joint Fund (MCM20200202), China Postdoctoral Science Foundation under Grant 2022M710468, Beijing University of Posts and Telecommunications-China Mobile Research Institute Joint Innovation Center.

References

Erik Boiy and Marie-Francine Moens. 2009. A machine learning approach to sentiment analysis in multilingual web texts. *Inf. Retr.*, 12(5):526–558.

Chenhua Chen, Zhiyang Teng, and Yue Zhang. 2020. Inducing target-specific latent structures for aspect sentiment classification. In *EMNLP (1)*, pages 5596–5607. Association for Computational Linguistics.

Peng Chen, Zhongqian Sun, Lidong Bing, and Wei Yang. 2017. Recurrent attention network on memory for aspect sentiment analysis. In *EMNLP*, pages 452–461. Association for Computational Linguistics.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: pre-training of deep bidirectional transformers for language understanding. In *NAACL-HLT (1)*, pages 4171–4186. Association for Computational Linguistics.

Chunning Du, Haifeng Sun, Jingyu Wang, Qi Qi, Jianxin Liao, Tong Xu, and Ming Liu. 2019. Capsule network with interactive attention for aspect-level sentiment classification. In *EMNLP/IJCNLP (1)*, pages 5488–5497. Association for Computational Linguistics.

Xiaochen Hou, Peng Qi, Guangtao Wang, Rex Ying, Jing Huang, Xiaodong He, and Bowen Zhou. 2021. Graph ensemble learning over multiple dependency trees for aspect-level sentiment classification. In *NAACL-HLT*, pages 2884–2894. Association for Computational Linguistics.

Bin Jiang, Jing Hou, Wanyue Zhou, Chao Yang, Shihan Wang, and Liang Pang. 2020. Metnet: A mutual enhanced transformation network for aspect-based sentiment analysis. In *COLING*, pages 162–172. International Committee on Computational Linguistics.

Diederik P. Kingma and Jimmy Ba. 2015. Adam: A method for stochastic optimization. In *ICLR (Poster)*.

Svetlana Kiritchenko, Xiaodan Zhu, Colin Cherry, and Saif Mohammad. 2014. Nrc-canada-2014: Detecting aspects and sentiment in customer reviews. In *SemEval@COLING*, pages 437–442. The Association for Computer Linguistics.

Keshav Kolluru, Vaibhav Adlakha, Samarth Aggarwal, Mausam, and Soumen Chakrabarti. 2020. Openie6: Iterative grid labeling and coordination analysis for open information extraction. In *EMNLP (1)*, pages 3748–3761. Association for Computational Linguistics.

Ruifan Li, Hao Chen, Fangxiang Feng, Zhanyu Ma, Xiaojie Wang, and Eduard H. Hovy. 2021. Dual graph convolutional networks for aspect-based sentiment analysis. In *ACL/IJCNLP (1)*, pages 6319–6329. Association for Computational Linguistics.

Xin Li, Lidong Bing, Wai Lam, and Bei Shi. 2018. Transformation networks for target-oriented sentiment classification. In *ACL (1)*, pages 946–956. Association for Computational Linguistics.

Xin Li, Lidong Bing, Piji Li, and Wai Lam. 2019. A unified model for opinion target extraction and target sentiment prediction. In *AAAI*, pages 6714–6721. AAAI Press.

Peiqin Lin, Meng Yang, and Jianhuang Lai. 2019. Deep mask memory network with semantic dependency and context moment for aspect level sentiment classification. In *IJCAI*, pages 5088–5094. ijcai.org.

Bing Liu. 2012. *Sentiment Analysis and Opinion Mining*. Synthesis Lectures on Human Language Technologies. Morgan & Claypool Publishers.

Qiao Liu, Haibin Zhang, Yifu Zeng, Ziqi Huang, and Zufeng Wu. 2018. Content attention model for aspect based sentiment analysis. In *WWW*, pages 1023–1032. ACM.

Thang Luong, Hieu Pham, and Christopher D. Manning. 2015. Effective approaches to attention-based neural machine translation. In *EMNLP*, pages 1412–1421. The Association for Computational Linguistics.

Dehong Ma, Sujian Li, Xiaodong Zhang, and Houfeng Wang. 2017. Interactive attention networks for aspect-level sentiment classification. In *IJCAI*, pages 4068–4074. ijcai.org.

Navonil Majumder, Soujanya Poria, Alexander F. Gelbukh, Md. Shad Akhtar, Erik Cambria, and Asif Ekbal. 2018. IARM: inter-aspect relation modeling with memory networks in aspect-based sentiment analysis. In *EMNLP*, pages 3402–3411. Association for Computational Linguistics.

Qianren Mao, Jianxin Li, Senzhang Wang, Yuanning Zhang, Hao Peng, Min He, and Lihong Wang. 2019. Aspect-based sentiment classification with attentive neural turing machines. In *IJCAI*, pages 5139–5145. ijcai.org.

Maria Pontiki, Dimitris Galanis, John Pavlopoulos, Harris Papageorgiou, Ion Androutsopoulos, and Suresh Manandhar. 2014. Semeval-2014 task 4: Aspect based sentiment analysis. In *SemEval@COLING*, pages 27–35. The Association for Computer Linguistics.

Patrick Putzky and Max Welling. 2017. Recurrent inference machines for solving inverse problems. *CoRR*, abs/1706.04008.

Alec Radford, Karthik Narasimhan, Tim Salimans, and Ilya Sutskever. 2018. Improving language understanding by generative pre-training.

Xiangying Ran, Yuanyuan Pan, Wei Sun, and Chongjun Wang. 2019. Learn to select via hierarchical gate mechanism for aspect-based sentiment analysis. In *IJCAI*, pages 5160–5167. ijcai.org.

Zhan Shi, Xinchi Chen, Xipeng Qiu, and Xuanjing Huang. 2018. Toward diverse text generation with inverse reinforcement learning. In *IJCAI*, pages 4361–4367. ijcai.org.

Ilya Sutskever, Oriol Vinyals, and Quoc V. Le. 2014. Sequence to sequence learning with neural networks. In *NIPS*, pages 3104–3112.

Yi Tay, Luu Anh Tuan, and Siu Cheung Hui. 2018. Learning to attend via word-aspect associative fusion for aspect-based sentiment analysis. In *AAAI*, pages 5956–5963. AAAI Press.

Yuanhe Tian, Guimin Chen, and Yan Song. 2021. Aspect-based sentiment analysis with type-aware graph convolutional networks and layer ensemble. In *NAACL-HLT*, pages 2910–2922. Association for Computational Linguistics.

Kai Wang, Weizhou Shen, Yunyi Yang, Xiaojun Quan, and Rui Wang. 2020. Relational graph attention network for aspect-based sentiment analysis. In *ACL*, pages 3229–3238. Association for Computational Linguistics.

Yequan Wang, Minlie Huang, Xiaoyan Zhu, and Li Zhao. 2016. Attention-based LSTM for aspect-level sentiment classification. In *EMNLP*, pages 606–615. The Association for Computational Linguistics.

Jianfei Yu and Jing Jiang. 2019. Adapting BERT for target-oriented multimodal sentiment classification. In *IJCAI*, pages 5408–5414. ijcai.org.

Lantao Yu, Jiaming Song, and Stefano Ermon. 2019. Multi-agent adversarial inverse reinforcement learning. In *ICML*, volume 97 of *Proceedings of Machine Learning Research*, pages 7194–7201. PMLR.

Lei Zhang, Shuai Wang, and Bing Liu. 2018. Deep learning for sentiment analysis: A survey. *WIREs Data Mining Knowl. Discov.*, 8(4).

Mi Zhang and Tieyun Qian. 2020. Convolution over hierarchical syntactic and lexical graphs for aspect level sentiment analysis. In *EMNLP (1)*, pages 3540–3549. Association for Computational Linguistics.

Li Zhou and Kevin Small. 2021. Inverse reinforcement learning with natural language goals. In *AAAI*, pages 11116–11124. AAAI Press.

Analyzing Persuasion Strategies of Debaters on Social Media

Matti Wiegmann[1*] **Khalid Al-Khatib**[2*] **Vishal Khanna**[1] **Benno Stein**[1]
[1] Bauhaus-Universität Weimar <first>.<last>@uni-weimar.de
[2] University of Groningen khalid.alkhatib@rug.nl

Abstract

Existing studies on the analysis of persuasion in online discussions focus on investigating the effectiveness of *comments* in discussions and ignore the analysis of the effectiveness of *debaters* over multiple discussions. In this paper, we propose to quantify debaters' effectiveness in the online discussion platform: "ChangeMyView" in order to explore diverse insights into their persuasion strategies. In particular, targeting debaters with different levels of effectiveness (e.g., good vs. poor), various behavioral characteristics (e.g., engagement), and text stylistic features (e.g., used frames) of debaters are carefully examined, leading to several outcomes that can be the backbone of writing assistants and persuasive text generation.

1 Introduction

Persuasion, a primary goal of argumentation, is the ability to convince people to do a certain action or form a particular belief (O'Keefe, 2006). Persuasion has always influenced the dynamics of communication and social interactions, either positively by raising awareness on critical issues like climate change or negatively by influencing the behavior of voters in elections or disseminating propaganda and fake news.

Social media, through its growing role in the formation of beliefs, has gained notable interest as a means to gather a deeper understanding of persuasion (Wang et al., 2021). The ChangeMyView (CMV) subreddit in particular has been used in various studies that model text persuasiveness using a variety of linguistic, argumentative, and behavioral features (e.g., (Hidey and McKeown, 2018), (Longpre et al., 2019), and (Guo et al., 2020)).

However, scholarly work on online persuasion centers around studying *persuasive comments* in individual discussions without considering the importance of analyzing *persuasive debaters* (Luu et al., 2019). Hence, debater strategies and their effectiveness have not been adequately studied. Understanding effective debating strategies and debater persuasiveness can be highly beneficial for media analysis, rhetorical review, and for learning debating skills. Besides, it can advance the development of several applications, where effective strategies can be recommended in writing assistants and dialog management systems or encoded in the backbone of text generation tools.

This paper focuses on analyzing the *debaters' persuasion strategies*, seeking to uncover the behavior, language style, and argumentative techniques that distinguish good from poor debaters. To do so, we categorize CMV debaters based on their effectiveness in persuasion and examine key differences in their behaviors and skills (i.e., engagement and experience), as well as their argument's style (at the semantic, syntactical, lexical, and pragmatic levels). We propose the task of identifying effective debaters and present an approach to tackle it.

Our analysis of debater strategies yields several insights. For example, we find that the effectiveness of persuasion improves over time for average debaters, the distribution of 'frames' in the debaters' arguments can play a major role in persuasion, and argumentative features based on the presence of certain types of arguments in the debaters' text do not seem sufficient to indicate the effectiveness of persuasion.

The contribution of this paper is threefold:

1. An extensive analysis of debater strategies across multiple discussions, addressing their behavior and stylistic aspects of their texts.
2. Insights about several techniques used by successful compared to unsuccessful debaters.
3. A new task of distinguishing good from poor debaters and an approach to address the task with multiple linguistic features.

All the resources developed in this paper can be found at https://doi.org/10.5281/zenodo.7034173.

[*] Equal contribution

Proceedings of the 29th International Conference on Computational Linguistics, pages 6897–6905
October 12–17, 2022.

2 Background and Related Work

In this section, we introduce CMV's core concepts as required for our study. Afterward, we overview the primary studies on modeling persuasiveness on CMV and related platforms.

2.1 Background

CMV is an open platform for users to engage in civilized discussions using sound arguments. CMV discussions are actively moderated to maintain the quality of argumentation. All comments and original posts must abide by the community rules.[1] These rules dictate a predictable structure for CMV debates, making them easy to process.

A CMV discussion begins with a user, called the *original poster (OP)*, submitting a marked request, called *original post*, to the CMV subreddit. The subreddit forbids non-debative submissions. The original post states the OP's stance on a controversial topic, relevant justifications and explanations of this stance, and an (implicit) call to "change my view". All other users of CMV called the *debaters*, can challenge the OP's stance and post opposing argumentative top-level comments. All debaters can respond to other comments to counter, cross-question, or defend their arguments, creating multi-layered and complex threads of conversation.

CMV offers two mechanisms to indicate comment persuasiveness: The delta (Δ) and the comment score. The delta mechanism allows the OP to mark up to one comment as persuasive. The 'awarded' deltas are aggregated and the per debater Δ-count is displayed publicly. Reddit's comment score is the per-comment sum of up and downvotes. The highest scoring comments are shown first. The comment score on CMV serves as an alt-metric indicating the persuasiveness as perceived by the community.

2.2 Related Work

The major work on the analysis of argument persuasiveness on social media (cf. (Tan et al., 2016), (Zhang et al., 2016), (Persing and Ng, 2017), and (Hidey and McKeown, 2018).) tries to determine how persuasive a comment is by solving the task: given two comments with a shared OP, identify the persuasive one. In contrast, our paper provides a higher-level analysis. We try to determine how persuasive a debater is by studying the debaters across multiple discussions, striving to disclose their persuasion strategies.

Employing argumentative features to predict comment persuasiveness is a well-established strategy; Egawa et al. (2019) annotated CMV discussions with elementary argumentative units (EUs) in a token-level five-class scheme: testimony, fact, value, policy, and rhetorical statement. The authors propose a Bi-LSTM-based sequence classifier for EU labeling. They conclude that EUs indicate persuasiveness if used effectively, 'fact' is the most persuasive, that the proportional distribution of types distinguishes CMV comments from original posts, and that persuasiveness is not indicated by the mere presence or absence of certain EUs.

Similarly, Hidey et al. (2017) annotated CMV discussions regarding arguments' claims as interpretation, evaluation, agreement, disagreement, or premises as ethos, logos, and pathos. The authors show that the relative positional distribution of argumentative components in a CMV comment is a signal for its persuasiveness. Additionally, Li et al. (2020) demonstrated the effectiveness of arguments' structural features in persuasiveness prediction. Multiple features were developed based on the usage of the proposition types reference, testimony, fact, value, and policy in the debaters' texts. The feature analysis showed that the presence of 'value' and 'testimony' bi-grams is more prevalent in persuasive argumentative texts, indicating that justifying claims with personal experiences is an effective persuasion strategy.

In this paper, we implemented the previously used argumentative features along with newly utilized ones like syntactic complexity, semantic similarity, and argument framing; the latter is shown to play a role in the debater's persuasiveness.

Different characteristics and behavior patterns of debaters were examined in a few papers. Addressing the debater behavior, Tan et al. (2016) investigated the role of debaters' interaction dynamics with the OP in persuasion and found that the debaters who responded early in the discussion tend to be more successful, that engaging with the original poster improves a debater's odds of success up to a threshold, and that higher debater participation in a discussion improves the odds of persuasion.

Targeting debaters' characteristics, Al-Khatib et al. (2020) modeled debaters' beliefs, personality traits, and interests based on their previous activities on Reddit, utilizing them for tackling the task

[1] CMV rules are stated on their wiki: https://www.reddit.com/r/changemyview/wiki/rules

of persuasiveness prediction. The study found the similarity between the characteristics of the OP and the debaters to be influential for effective persuasion. In comparison, our paper groups debaters based on their persuasiveness so we can probe the diverse strategies used by good vs. poor debaters.

Analyzing the discussion structure, Guo et al. (2020) hypothesized that persuading the OP in a CMV discussion happens gradually throughout a multi-party conversation rather than instantaneously. A prediction task was performed to model the cumulative effect of a sequence of comments in a CMV discussion and detect the position where the persuasion of the OPs occurs. Besides, a user study to evaluate the persuasiveness of debaters' arguments' was conducted, concluding that the perception of persuasiveness differs across individuals and that it is influenced by one's idiosyncrasies i.e. the same argument could be persuasive for one person but not persuasive for another. Likewise, Wei et al. (2016) considered the relevance ranking of CMV comments by their score in a discussion. They found the comment's score to be influenced by its temporal entry order as well as the past credibility of its corresponding debater. The credibility is measured by the number of prior deltas received by a debater. Several feature classes were used for the relevance ranking task, including linguistic features derived from the comment's text, interaction-based features obtained by modeling the CMV discussion as a tree, and argumentative features such as the proportion of argumentative text and argument relevance and originality.

The only work that targets the debater-level analysis is by Luu et al. (2019) and investigates how debaters' skill improves over time as they learn how to interact with other debaters. They present a strong estimator of the development of a debater's persuasive skill over time using several linguistic features, such as length of comments, co-occurrence of hedges, and fighting words.

Our work is distinct in several respects: First, we analyze CMV, as opposed to `Debate.org`, which is more strict and conventional regarding debate structure. Second, we analyze the relationship between the debaters' engagement, experience, and writing style across linguistic dimensions, accounting for the argumentative nature of debate texts. Finally, we address different levels of debater persuasiveness and scrutinize the differences in their argumentation strategies.

3 Data Acquisition and Preparation

To conduct our study on debater persuasion strategies, we created a dataset with $3,801$ CMV debaters, equally sampled for good, average, and poor debater persuasiveness. Here, we detail our data collection method, quantification of debater persuasiveness, and sampling method to balance the dataset by debater persuasiveness.

Quantifying Debater Persuasiveness We define the persuasiveness of a debater d with comments $c_1, \ldots, c_{i,\Delta 1}, \ldots, c_{j,\Delta k}, \ldots, c_n$ in CMV as ratio of delta comments c_Δ to all comments:

$$\text{Persuasiveness}(d) = \frac{k}{n}.$$

The persuasiveness is, hence, the number of debater's delta comments normalized by her total comment count. As Table 1 shows, this normalization is necessary because the delta-comment count correlates strongly with the total comment count.

Based on the persuasiveness score, we categorize debaters into three groups as follows:

1. **Good debaters** with a persuasiveness of 5% or above.

2. **Average debaters** with a persuasiveness between 0% and 5%.

3. **Poor debaters** with a persuasiveness of 0%; These debaters did not receive any delta during their active period on CMV.

The separation of debaters with a non-zero persuasiveness is based on the observation that obtaining any c_Δ is already challenging. Hence, the highly persuasive tail should be studied as a separate population. The 5%-threshold used in categorization separates the non-poor debaters into two groups of approximately equal size.

Collecting Debater Comments We obtained an initial set of CMV debates from the Webis CMV corpus (Al-Khatib et al., 2020), which comprises all CMV debates from June 2005 to September 2017. We extracted all top-level comments from the Webis CMV corpus and grouped them by debater. We discarded all inactive debaters with less than 10 comments and obtained an unbalanced dataset of $13,254$ CMV debaters along with their top-level comments on various debates. We only considered top-level comments since they serve as

"entries to the debate", while lower-level comments are either rebuttals or non-argumentative content like corrections, clarifications, or thanks.

Sampling Data In the intermediate dataset, 80% of the debaters are of poor persuasiveness and have never been awarded a Δ. Since we aim for a controlled analysis, we resampled the dataset in such a way that the distribution of CMV debaters is balanced by persuasiveness. Overall, we end up with $3,801$ entries, evenly distributed across the three debater categories.

Our resampling strategy first added a "good" debater to the dataset by random and then selected one "average" and one "poor" debater with the same number of comments, or the closest number to that. If multiple candidate debaters existed, we minimized the absolute difference in mean comment length. This resampling strategy minimizes the effects of comment count and comment length in the dataset since both are indicative of persuasiveness (cf. Section 2).

4 Debater Engagement and Experience

Our first analysis concerns the relationship between the debaters' persuasiveness and their engagement with and experience on the CMV subreddit. We presume that engagement on CMV may correspond to rebuttals in live debates. Our findings suggest that a high engagement is indicative of persuasive debaters. We further inspect the relationship between experience and persuasiveness in both absolute measures such as comment count and active period and relative measures such as changes in style and persuasiveness with experience gain. Our findings suggest that debaters become more persuasive with increased experience, especially average debaters. However, the experience effect is not reflected in absolute experience measures, and hence it is hard to operationalize for classification.

4.1 Engagement

Figure 1a shows that persuasive comments and persuasive debaters are more engaging. We measure debater engagement by the average number of replies to persuasive and non-persuasive comments. Persuasive debaters get ~10% more replies to their total comments compared to average debaters and ~30% more replies compared to poor debaters. Persuasive comments get ~250% as many replies as non-persuasive comments.

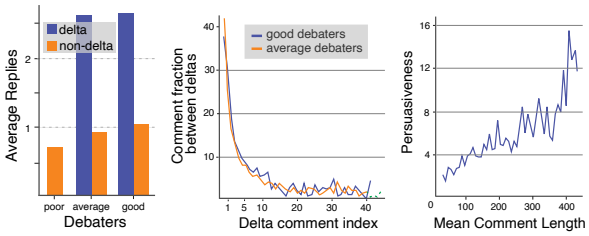

(a) Engagement (b) Persuasion Freq. (c) Length

Figure 1: (a) Engagement of debaters by persuasiveness. (b) Evolution of the frequency of persuasive comments. (c) Persuasiveness by debaters' average comment length.

	Persuasiveness	Δ Count	Score
Comments	0.02	0.72	0.03
Active Period	-0.03	0.13	0.15

Table 1: Pearson ρ between three success measures (Persuasiveness; Δ Comment Count; Median Reddit Comment Score) and two absolute experience measures (Active Period: the time between the first and last comment on CMV; Number of Comments).

4.2 Absolute Experience

Table 1 shows that the absolute measures of experience are insufficient. We can observe that neither the active period—the time between the first and latest comment—nor the comment count correlates with persuasiveness or Reddit score. We disregard the correlation between the total comment count and the number of persuasive comments as evidence of debater experience without observing a correlation with persuasiveness.

4.3 Relative Experience

We model the relative experience of a debater on CMV as seen from the comment: A debater is inexperienced for her first comment and very experienced for her last; that is to say, the experience of the debater d of a comment c_t in a sequence $c_1, \ldots, c_n$ is $\text{Experience}(c_t) = \frac{t}{n}$. We analyze the impact of experience gain of good and average debaters on persuasiveness, persuasion frequency, comment length, as length is the most indicative feature in comment classification, and average comment score, which represents the CMV community's opinion on persuasiveness.

Persuasiveness Figure 2a shows that the overall persuasiveness of good debaters is largely unaffected by experience while the persuasiveness of average debaters almost doubles.

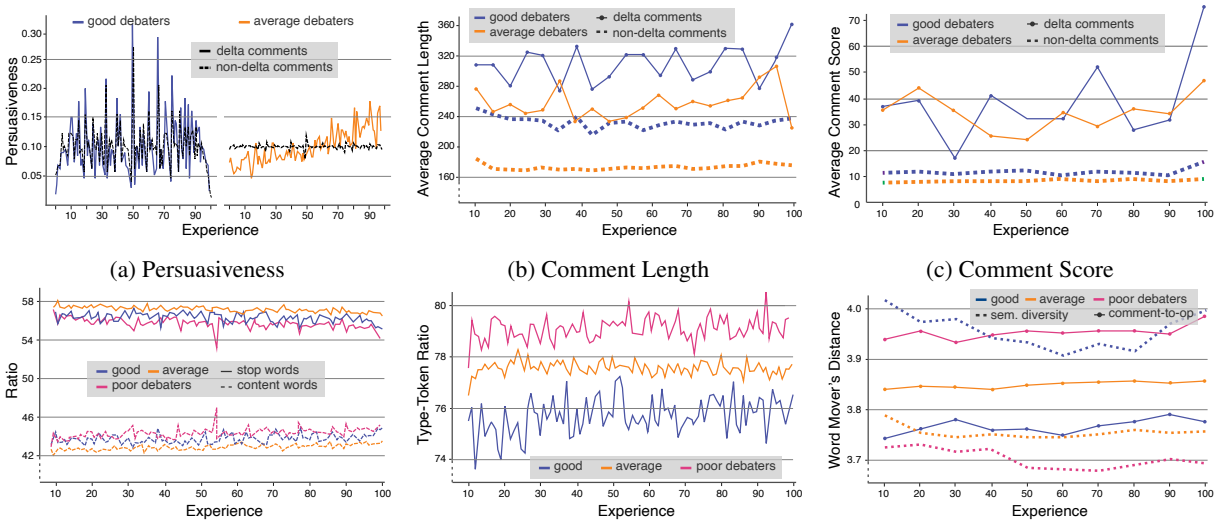

(a) Persuasiveness (b) Comment Length (c) Comment Score

(d) Stop Word Ratio (e) Type-Token Ratio (f) Semantic Similarity

Figure 2: Changes of various debater-level features with increasing relative experience. The color indicates persuasiveness and the line style (dashes/dots) indicates the secondary variable.

Persuasion Frequency Figure 1b shows that the persuasion frequency increases sharply up to the 5th persuasive comment for both good and average debaters and increases slightly up to the 15th persuasive comment. This indicates that debaters learn to replicate persuasive strategies and become more persuasive with experience. We measure persuasion frequency as the number of non-delta comments that occur between two consecutive delta comments, as a fraction of the total comments made. A decreasing delta-to-non-delta rate indicates more frequent persuasions.

Comment Length Figure 2b confirms the established assumption that length is highly indicative of persuasiveness. There is no indication that relative experience has any substantial impact on the length of delta or non-delta comments.

Average Comment Score Figure 2c shows that the mean-comment score, the alt-metric for community persuasiveness, increases with experience but not consistently. On average, however, debaters score higher on persuasive comments with increasing experience. The effect. however, is negligible on non-delta comments.

5 Debater Style Analysis

Stylistic features are frequently used to determine the characteristics of authors. Since stylistic features are indicative of persuasive comments, we consider stylistic features to also be indicative of persuasive debaters. In particular, we study the relationship between a debater's persuasiveness and the lexical, syntactic, semantic, and pragmatic dimensions. We found notable differences in persuasiveness in each dimension. The most substantial feature is again comment length. Additionally, we found that better debaters tend to have lower lexical diversity and syntactic complexity, but a higher semantic diversity. We also found correlations between certain word class patterns and certain patterns of elementary argumentative units, particularly rhetorical statements. Lastly, found that persuasive debaters use political and cultural identity frames more often.

5.1 Lexical Dimensions

Within the lexical dimension of style, we analyze the relation between debater persuasiveness and the (1) comment length and the (2) lexical diversity, in particular the stop-word and type-token ratio.

Comment Length Figure 1c shows that debaters with a higher mean comment length are also, consistently and without apparent bound, more persuasive on average. Figure 2b shows, independently of the debater's experience, that persuasive comments are longer than non-persuasive comments and that good debaters write longer (~20%) comments. These findings are consistent with previous evidence (cf. Section 2) and suggest that the comment length is highly indicative of the persuasiveness of comments and debaters alike.

WC n-gram	ρ	WC n-gram	ρ
IN JJ	0.11	PRP VBP	-0.13
NN IN JJ	0.10	PRP	-0.12
JJ NN IN	0.09	WRB VBP	-0.11
VBG DT JJ	0.08	NN WRB	-0.11

Table 2: Top Pearson ρ between a word class n-gram and persuasiveness.

Lexical Diversity Figure 2d shows that the differences in the stop-word ratio are consistently small (<1%) and have no direction since good debaters are between poor and average ones. However, Figure 2e shows that the type-token ratio has a higher effect size of 2% among the debater groups and has a direction. This suggests that good debaters write comments with lower lexical diversity.

5.2 Syntactic Dimensions

Within the syntactic dimension of style, we analyze the relationship between persuasiveness and syntactic complexity and the word class n-gram distribution.

Syntactic Complexity The complexity of a debater's text was measured based on the dependency parse trees of all sentences in her top-level comments. We measure the Pearson correlation between debater persuasiveness and three common syntactic complexity measures:[2] Outdegree centrality ($\rho = -0.17$), Closeness centrality ($\rho = -0.16$), and the number of dependents per word ($\rho = 0.17$). Since a high centrality indicates complex syntax, and persuasiveness is negatively correlated with centrality, our results suggest that good debaters use less complex syntax. However, all correlations are weak ($\rho <= 0.25$).

Word class n-grams Table 2 shows the word class 1–3-grams with the strongest correlation with persuasiveness. Here, better debaters use adjectives more and `PRP VBP` (e.g. you did ...) as well as `WRB VBP` (e.g. how did ...) less frequently. Although the correlation is weak and word-class n-grams are difficult to interpret, these results may indicate an impact of certain syntactical structures on debater persuasiveness as for comment persuasiveness (cf. Tan et al. (2016)). We determined the word class n-grams using NLTK and the Penn tagset since all CMV comments are English. We only inspected the 1,000 most frequent n-grams.

[2]We measured the complexity using https://github.com/tsproisl/textcomplexity

5.3 Semantic Dimension

Within the semantic dimension of style, we measure the relation between debater persuasiveness and the (1) semantic similarity between a debater's comment and the original post and the (2) semantic diversity within the comments of a debater. We use Word Movers Distance[3] (WMD, Kusner et al., 2015) to measure the semantic similarity.

Similarity between Comment and Original Post Figure 2f shows that the WMD is lower the more persuasive a debater is. Hence persuasive debater's comments are semantically more similar to the original post.

Semantic Diversity Figure 2f shows the semantic diversity for debaters with different persuasiveness, whereas the semantic diversity is higher for better debaters.

The semantic diversity should indicate if a debater prefers semantic depth (few different concepts discussed) or breadth (many different concepts discussed) within each comment. For lack of a better (lexeme-agnostic) intra-document semantic similarity measure, we use a sentence-based heuristic:

$$\mathrm{SemDiv}(c_k) = \frac{2}{n^2 - n} \sum_{i=1}^{n-1} \sum_{j=i+1}^{n} \mathrm{WMD}(s_i, s_j).$$

Here, the semantic diversity of a debater is the average diversity of the comments $c_k = s_1, \ldots, s_n$, and the diversity of the comments is the average WMD between each pair of sentences (s_i, s_j). We assume WMD captures the semantic diversity between two sentences in this way.

5.4 Pragmatic Dimension

Within the pragmatic dimension of style, we measure the relation between debater persuasiveness and (1) the distribution of argumentative units: elementary units, claims, and premises, (2) framing strategies.

Argumentative Units Table 3 shows the argumentative unit n-grams which correlate the strongest with debater persuasiveness, while all other unit n-grams do not correlate with $\rho \leq 0.05$. All correlating units are elementary units, with rhetorical statements being the most persuasive. No claim or premise types correlate in a meaningful way with persuasiveness.

[3]We use Gensim with fastText embeddings

Unit n-gram	ρ	Unit n-gram	ρ
rhetoric	-0.194	policy	-0.110
value	-0.126	rhetoric rhetoric	-0.101
rhetoric value	-0.114	rhetoric rhetoric none	-0.063

Table 3: Argumentative units with largest absolute Pearson ρ with CMV debaters' persuasiveness. All other combinations correlated with $\rho \leq 0.05$

We measure the Pearson correlation between persuasiveness and the relative frequency of elementary unit 1–3-grams, where each sentence of a debater's comment is assigned one unit. We use the five elementary units testimony, fact, value, policy, and rhetorical statement proposed by Egawa et al. (2019) for CMV comments. We determine the elementary unit of a sentence with a BERT-based classifier trained on Egawa et al. (2019)'s annotated dataset of CMV comments and original posts; The classifier reaches a 6-class (including *None*) micro-accuracy of 0.75 on the standard split. Since the dataset annotates units on a token level, we assign each sentence the unit assigned to its tokens, discarding sentences with multiple units annotated.

We also measure the Pearson correlation between persuasiveness and the relative frequency of 1–3-grams of claim and premise types, where each sentence of a debater's comment is assigned one type. We use the 2-stage classification scheme proposed by Hidey et al. (2017) for CMV comments. Each sentence is first classified with a BERT model as claim, premise, or neither. Claims are then classified as interpretation, evaluation/rational, evaluation/emotional, or agreements. Premises are classified into eight classes, one for each combination of ethos, logos, and pathos using three binary classifiers. We trained each of the five needed classifiers on Hidey et al. (2017)'s datasets of CMV discussions.

Frames Figure 3 shows how often debaters with different persuasiveness use certain frames in their comments. Most frames are used equally often independently of persuasiveness, except for the *political* and *cultural identity* frames, which are used notably more often by better debaters.

We determined frames by classifying each sentence of each comment of a debater with one of the 15 frames used in Card et al. (2015)'s Media Frames corpus of manually annotated news articles. We trained a BERT classifier to classify the sentences, which reaches a micro accuracy of 0.68 in 5-fold random cross-validation.

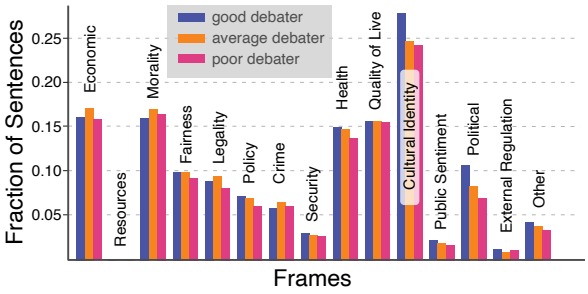

Figure 3: Distribution over the 15 sentence-level frames for good, average, and poor debaters.

6 Debater Persuasiveness Prediction

In addition to the analytical scrutiny of debater persuasiveness, we conduct an experimental validation of our findings by classifying debaters by persuasiveness. We define the general task of debater-level persuasiveness prediction as: Given a debater d with comments $c_1, \ldots, c_n$, classify this debater as persuasive (good) or non-persuasive (average or poor). To conclusively supplement our analysis, we individually inspect the classification performance of the introduced features (cf. Section 5).

We encoded the syntactic, semantic, and pragmatic features of our analysis for each of the 3,801 debaters in our CMV debaters' corpus. Each encoding was chosen to obfuscate comment length as far as reasonable. We encoded the word class and all argumentative unit n-grams tf-idf vectors of the aggregated comments. We encoded the numerical features, outdegree centrality, closeness centrality, and the number of dependents for text complexity and comment-op distance and within-comment distance for WMD, by averaging comment-level counts per debater. We encoded each of the 15 frames with the absolute and relative number of comments that utilize a frame.

As baselines, we selected feature sets previously used for *comment persuasiveness* prediction: Bag-of-Words, vocabulary interplay after (Tan et al., 2016), which covers OP and commenters' vocabularies' absolute and relative overlap and Jaccard similarity, and common stylometrics, which cover counts of words, selected word classes, links, word lists, symbols, type-token ratio, and readability scores. The baseline feature sets were implemented trivially following the related work.

We consider two binary classification settings for our experimental validation: (1) good vs. average and (2) good vs. poor. We maintained a balanced distribution of the classes (1,267 each). The

Features	vs Average	vs Poor
Baseline Features		
Bag of Words	0.60	0.68
Stylometry	0.62	0.67
Vocabulary Interplay	0.58	0.67
Syntactic Features		
Word class n-grams	0.57	0.51
Text Complexity	0.65	0.61
Semantic Features		
Word Mover's Distance	0.59	0.63
Pragmatic Features		
Elementary Units	0.51	0.59
Claim or Premise	0.47	0.55
Claim Type	0.48	0.58
Premise Type	0.48	0.58
Claim and Premise Types	0.48	0.58
Frames	**0.70**	**0.72**

Table 4: Macro F1 score of the two classification settings: Good vs. Average debaters and Good vs. Poor debaters.

classification is done using logistic regression with default parameters on a random 80-20 train-test split. The effectiveness of the classifiers is reported using macro F1-score as shown in Table 4.

The classification results reveal several findings: First, most features distinguish good from poor debaters better than good from average ones. Syntactic features are the only exception to this trend, which can not be explained by our analysis. Second, Bag-of-words is a strong feature for the two settings as it outperforms most of the other features. Besides, the weak effectiveness of the argumentative features is similar to the observations of Egawa et al. (2019); the mere distribution of argumentative units in the text is insufficient to identify its persuasiveness. Third, the distribution of the frames in the debaters' comments results in the best scores across the two experimental settings. The most discriminating frames are 'Quality of Life', 'Morality', and 'Health and Safety', all with negative weights towards the 'good debater' class.

7 Conclusion and Discussion

The persuasion skills of debaters can vary depending on different cultural and social aspects, among others. Understanding how people argue and what makes some debaters more successful than others are interesting research questions that have been neglected in the literature. This paper has contributed in this regard by modeling debater effectiveness in CMV and analyzing their behavior and argumentative stylistic choices, demonstrating several interesting insights that can be utilized for improving the persuasion skills of new debaters and assessing the development of advanced text generation and writing assistant tools.

Still, we think there is room for further analysis. First, we quantified the persuasiveness of CMV debaters based on awarded Δs only; Although it appears to be a standard method in previous work, we believe that a more comprehensive quantification, possibly using human judgments and a more fine-grained scale, would account for a degree of subjectivity to consider the evaluating user's idiosyncrasies. Guo et al. (2020) touches on this briefly, finding that despite general agreement about what is persuasive, there are differences in the assessment of persuasion based on the positions of the evaluating party.

Second, while argumentative features based on the distribution of argumentative units did not perform well in our prediction task, possible improvements can be achieved through modeling features that can capture the effective use of argumentative units. A possible direction is to identify the interdependencies between the different argumentative units in the text (Li et al., 2020) as well as their relative arrangement (Hidey et al., 2017).

Third, other, more in-depth features can disclose useful insights into debater persuasiveness. Conceivable are features that better model behavior like experience and the dynamic of debater interaction or the velocity of experience gain.

Fourth, using more sophisticated models in the prediction task may lead to better results, although our logistic regression is ideal to compare class separability by feature. Guo et al. (2020) proposed using conditional random fields (CRF) to model the cumulative effect of persuasion in CMV discussions, and Li et al. (2020) used bi-LSTM and BERT to model their persuasiveness task.

8 Impact Statement

In a broader context, the findings of this work support the detection, writing, or generation of highly persuasive text, particularly in a social media register. This capability can be abused to generate highly persuasive and misleading, deceptive, fake, or abusive text. Hence, knowledge about debater persuasiveness bears the potential for more persuasive believable social bots. Our work, however, with its focus on the analytical side, bears the same potential to detect generated, hyper-persuasive text.

As with all author-level work on social media, our methods bear the potential to profile users of social media platforms and use the information against them, for example, to automatically block or downvote poor debaters, track them across websites, and possibly reveal their identity. We have taken care to anonymize the user's IDs in our dataset and not release any models that would (re-)generate information about them.

References

Khalid Al-Khatib, Michael Völske, Shahbaz Syed, Nikolay Kolyada, and Benno Stein. 2020. Exploiting personal characteristics of debaters for predicting persuasiveness. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, ACL 2020, Online, July 5-10, 2020*, pages 7067–7072. Association for Computational Linguistics.

Dallas Card, Amber E. Boydstun, Justin H. Gross, Philip Resnik, and Noah A. Smith. 2015. The media frames corpus: Annotations of frames across issues. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing of the Asian Federation of Natural Language Processing, ACL 2015, July 26-31, 2015, Beijing, China, Volume 2: Short Papers*, pages 438–444. The Association for Computer Linguistics.

Ryo Egawa, Gaku Morio, and Katsuhide Fujita. 2019. Annotating and analyzing semantic role of elementary units and relations in online persuasive arguments. In *Proceedings of the 57th Conference of the Association for Computational Linguistics, ACL 2019, Florence, Italy, July 28 - August 2, 2019, Volume 2: Student Research Workshop*, pages 422–428. Association for Computational Linguistics.

Zhen Guo, Zhe Zhang, and Munindar P. Singh. 2020. In opinion holders' shoes: Modeling cumulative influence for view change in online argumentation. In *WWW '20: The Web Conference 2020, Taipei, Taiwan, April 20-24, 2020*, pages 2388–2399. ACM / IW3C2.

Christopher Hidey and Kathleen R. McKeown. 2018. Persuasive Influence Detection: The Role of Argument Sequencing. In *Proceedings of the Thirty-Second AAAI Conference on Artificial Intelligence, (AAAI-18), the 30th innovative Applications of Artificial Intelligence (IAAI-18), and the 8th AAAI Symposium on Educational Advances in Artificial Intelligence (EAAI-18), New Orleans, Louisiana, USA, February 2-7, 2018*, pages 5173–5180. AAAI Press.

Christopher Hidey, Elena Musi, Alyssa Hwang, Smaranda Muresan, and Kathy McKeown. 2017. Analyzing the semantic types of claims and premises in an online persuasive forum. In *Proceedings of the 4th Workshop on Argument Mining, ArgMining@EMNLP 2017, Copenhagen, Denmark, September 8, 2017*, pages 11–21. Association for Computational Linguistics.

Matt J. Kusner, Yu Sun, Nicholas I. Kolkin, and Kilian Q. Weinberger. 2015. From word embeddings to document distances. In *Proceedings of the 32nd International Conference on Machine Learning, ICML 2015, Lille, France, 6-11 July 2015*, volume 37 of *JMLR Workshop and Conference Proceedings*, pages 957–966. JMLR.org.

Jialu Li, Esin Durmus, and Claire Cardie. 2020. Exploring the role of argument structure in online debate persuasion. *CoRR*, abs/2010.03538.

Liane Longpre, Esin Durmus, and Claire Cardie. 2019. Persuasion of the Undecided: Language vs. the Listener. In *Proceedings of the 6th Workshop on Argument Mining*, pages 167–176, Florence, Italy. Association for Computational Linguistics.

Kelvin Luu, Chenhao Tan, and Noah A. Smith. 2019. Measuring Online Debaters' Persuasive Skill from Text over Time. *Trans. Assoc. Comput. Linguistics*, 7:537–550.

Daniel J. O'Keefe. 2006. Persuasion. In *Encyclopedia of Rhetoric*. Oxford University Press.

Isaac Persing and Vincent Ng. 2017. Lightly-Supervised Modeling of Argument Persuasiveness. In *Proceedings of the Eighth International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 594–604, Taipei, Taiwan. Asian Federation of Natural Language Processing.

Chenhao Tan, Vlad Niculae, Cristian Danescu-Niculescu-Mizil, and Lillian Lee. 2016. Winning Arguments: Interaction Dynamics and Persuasion Strategies in Good-faith Online Discussions. In *Proceedings of the 25th International Conference on World Wide Web*, WWW '16, pages 613–624, Republic and Canton of Geneva, Switzerland. International World Wide Web Conferences Steering Committee.

Yizhi Wang, Yuwan Dai, Hao Li, and Lili Song. 2021. Social media and attitude change: Information booming promote or resist persuasion? *Frontiers in Psychology*, 12.

Zhongyu Wei, Yang Liu, and Yi Li. 2016. Is this post persuasive? ranking argumentative comments in online forum. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics, ACL 2016, August 7-12, 2016, Berlin, Germany, Volume 2: Short Papers*. The Association for Computer Linguistics.

Justine Zhang, Ravi Kumar, Sujith Ravi, and Cristian Danescu-Niculescu-Mizil. 2016. Conversational Flow in Oxford-style Debates. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 136–141. Association for Computational Linguistics.

KC-ISA: An Implicit Sentiment Analysis Model Combining Knowledge Enhancement and Context Features

Minghao Xu, Daling Wang*, Shi Feng, Zhenfei Yang, Yifei Zhang
Northeastern University, Shenyang, China
2071825@stu.neu.edu.cn, {wangdaling, fengshi}@cse.neu.edu.cn
2001851@stu.neu.edu.cn, zhangyifei@cse.neu.edu.cn

Abstract

Sentiment analysis has always been an important research direction in natural language processing. The research can be divided into explicit sentiment analysis and implicit sentiment analysis according to whether there are sentiment words in language expression. There have been many research results in explicit sentiment analysis. However, implicit sentiment analysis is rarely studied. Compared with explicit sentiment expression, implicit sentiment expression usually omits a lot of knowledge and common sense, and context also has an important impact on implicit sentiment expression. In this paper, we use a knowledge graph to supplement implicit sentiment expression and propose a novel **I**mplicit **S**entiment **A**nalysis model combining **K**nowledge enhancement and **C**ontext features (dubbed KC-ISA). The KC-ISA model can effectively integrate external knowledge and contextual features by the coattention mechanism. Finally, we conduct experiments on the SMP2019 implicit sentiment analysis dataset. Moreover, to verify the generality of the model, we also conduct experiments on two common sentiment analysis datasets. The results on three datasets show that our proposed KC-ISA model can achieve better results on text sentiment analysis.

1 Introduction

With the rapid development of the Internet, the Internet has become the media form with the widest audience and the fastest reflection of social dynamics in today's society. Online social media has become an excellent platform in which all users can easily participate, share and communicate. Therefore, a large number of texts that reflect the user's subjective consciousness have begun to appear, and the texts with the user's sentiment tendencies are also increasing.

Text sentiment analysis is the process of analyzing, summarizing, and reasoning subjective sentiment texts. From the expression level of text, it can be divided into explicit sentiment text and implicit sentiment text according to whether the text contains explicit sentimental words. Liu (2012) firstly divided sentiment into subjective opinion and factual implication opinion. The former is a subjective statement that gives the polarity of sentiment, while the latter is an implicit expression of sentiment through an objective statement. Liao et al. (2019) defined implicit sentiment sentences as "language fragments that express subjective sentiment but do not contain explicit sentiment words". As the basic research of sentiment analysis, the analysis of explicit sentiment text has abundant relevant research results. However, the emotions reflected by people's behavior and feelings towards objective things are often rich and abstract and are usually expressed in subtle ways. For example, using objective statements to express one's sentiment, or using rhetorical expressions to describe them vividly, is called implicit sentiment expression. That means the implicit sentiment expression depends on something other than the sentiment words. So implicit sentiment analysis first needs to discover these dependencies. According to Li et al. (2021b), about 30% of sentences contain implicit sentiment.

By reviewing the relevant paper (Liao et al., 2019) and analyzing the data, we summarized that implicit sentiment sentences have the following three distinct dependencies:

(1) **Context Dependency**

Below are two paragraphs of implicit sentiment expression text E1 and E2, each of which contains two sentences S1 and S2.

E1-S1: This orange is only sold for 1 yuan a pound. E1-S2: It was so cheap!

E2-S1: This orange is only sold for 1 yuan a pound. E2-S2: It must be very bad!

E1-S1 and E2-S1 are the target sentences for sentiment analysis. In E1-S1, because the price

* Corresponding author

Proceedings of the 29th International Conference on Computational Linguistics, pages 6906–6915
October 12–17, 2022.

of oranges is low, the speaker has the intention to buy and expresses a positive sentiment. In E2-S1, because the price is too low, the speaker doubts the taste of orange and expresses a negative sentiment. The same sentence expresses different sentiments due to the difference between E1-S2 and E2-S2. Therefore, contextual information is an important factor affecting implicit sentiment expression. If we want to accurately analyze implicit sentiment expression, it is crucial to integrate contextual features.

(2) **Sentiment Target Dependency**

Let's look at the next two implicit sentiment texts.

E3: In the race, he runs like a cheetah.

E4: In the race, he runs like a turtle.

E3 and E4 are exactly the same except for the sentiment target, but the first sentence is positive and the second sentence is negative. The two sentences express completely opposite implicit sentiment only by the change of the sentiment target, indicating that the implicit sentiment is also closely related to the sentiment target.

(3) **Knowledge and Common Sense Dependency**

Short text reviews usually omit a lot of knowledge and common sense, and it is difficult to judge the true sentiment polarity from the text itself. For example, in E3 and E4, to accurately judge the sentiment of these two sentences, we also need to have certain knowledge and common sense, and we must know that cheetahs run fast, but turtles are slow. But the target sentence we want to analyze does not contain such background common sense. The external knowledge just makes up for these shortcomings. It contains the relationship between sentiment words and non-sentiment words and evaluation objects in different fields.

Therefore, to address the three distinct dependencies of implicit sentiment expression, we need model context, sentiment target, and external knowledge. Especially, for knowledge and common sense dependency, we introduce a knowledge graph as external common sense to supplement implicit sentiment sentences to enrich the expression of sentiments. Inspired by the Coattention (Xiong et al., 2017) mechanism, in this paper, we propose KC-ISA, a novel **I**mplicit **S**entiment **A**nalysis model combining **K**nowledge enhancement and **C**ontext features, which can effectively integrate external knowledge and contextual features to complement sentiment expressions. Moreover, by the pre-trained model, KC-ISA can utilize external knowledge and contextual features more effectively. Extensive experiments show that our proposed KC-ISA model can effectively improve the accuracy of implicit sentiment analysis.

In summary, the main contributions of this paper are as follows:

First, we propose KC-ISA model for three distinct dependencies of implicit sentiment analysis. Moreover, we combined KC-ISA model with the pre-trained model BERT as KC-ISA-BERT. KC-ISA model can effectively integrate contextual features and incorporate external knowledge into implicit sentiment analysis.

Second, we conduct experiments on implicit sentiment dataset, and the results show that our proposed model can effectively improve the accuracy of implicit sentiment analysis.

Third, considering the real review data is a mixture of explicit and implicit data, we further conducted experiments on two common sentiment analysis datasets. The experimental results further validate the effectiveness and general applicability of our KC-ISA model.

The datasets and code have been uploaded to https://github.com/AnonymousColing2022/KC-ISA

2 Related Work

Related work mainly includes two parts, one is implicit sentiment analysis, and the other is integrating external knowledge in natural language processing tasks.

2.1 Implicit Sentiment Analysis

Liu (2012) first classified sentiment analysis into explicit and implicit sentiment analysis. Liao et al. (2019) focused on the recognition of fact-implied implicit sentiment at the sentence level and proposed a multi-level semantic fusion method based on representation learning to learn recognition features. Greene and Resnik (2009) used grammatical structure to establish language-driven features and implicit sentimental associations for sentences without obvious sentimental indicators but still able to express sentiments or opinions, and used similarity calculation to improve the effect of text sentiment analysis. Wei et al. (2020) proposed a BIL-STM model with multi-polar orthogonal attention for implicit sentiment analysis. Compared with

traditional single-attention models, multi-polar attention can identify the differences between words and sentimental orientations. Wang et al. (2021) proposed an implicit sentiment sentence discrimination method fused with context information and established a multi-level orthogonal attention C-MPOA model fused with context information. Zuo et al. (2020) obtained the features of implicit sentiment sentences and context through GCN, and proposed a context-specific heterogeneous graph convolutional neural network(CsHGCN) to solve the problem of missing sentimental words. Zhou et al. (2021) represented an event as the combination of its event type and the event triplet. Based on such event representation, they proposed a model with a hierarchical tensor-based composition mechanism to detect sentiment in text. The previous works mainly focused on the structure of the sentence itself and its context. To our knowledge, it is the first time that we introduced the knowledge graph as external knowledge into the implicit sentiment analysis task and then integrated the contextual features to enable the model to better understand the implicit sentiment contained in the sentence, thereby improving the accuracy of the implicit sentiment analysis.

2.2 Integrating External Knowledge

Not only in sentiment analysis, but in many natural language processing tasks, there are a lot of works on how to introduce external knowledge for knowledge enhancement. Wang et al. (2018) introduced the information of the knowledge graph in the sorting task of the recommendation system and established a Ripple Set through the user's historical information, that is, the nodes corresponding to the knowledge graph, and each step will be propagated to the adjacent nodes, thereby improving the recommendation accuracy and variety. Liu et al. (2018) added the embedding of the input entities in the knowledge graph in the text matching task as additional supervision information to the model, which can well help the model achieve higher-level semantic understanding. Zhang et al. (2019) modified the original encoder structure of BERT and added entity embedding input. In this way, external knowledge was introduced in pre-training process to improve the effect of downstream tasks. Wang et al. (2020) introduced external knowledge in natural language inference tasks. They searched for the entities corresponding to the subject, predicate, and object in the two sentences in the knowledge graph, found the path between the two corresponding entities, then obtained the representation of the corresponding knowledge through BILSTM, and finally combined it with the original text to infer. Instead of directly adding external knowledge, Sun et al. (2019) modified BERT's mask strategy. Through entity-level mask and phrase-level mask, the model implicitly learns relevant knowledge and longer semantically dependent information, such as the relationship between entities, the properties of the entity, and the type of the events, which can make the model have better generalization and adaptability. There are also many related works on integrating external knowledge in natural language processing tasks, but less research on implicit sentiment analysis. To our knowledge, it is the first time that we have introduced both knowledge graph and contextual features into implicit sentiment analysis task, which perfectly matches the dependencies of implicit sentiment expression (context dependency, sentiment target dependency and knowledge and common sense dependency), which can significantly improve the accuracy of implicit sentiment analysis.

3 Proposed Model

We summarized three dependencies of implicit sentiment expression earlier, i.e. context dependency, sentiment target dependency, and knowledge and common sense dependency. Since there are no explicit sentiment words in implicit sentiment sentences as a guide, traditional sentiment analysis methods are not very effective. Therefore, we proposed KC-ISA model for the three distinct dependencies. The model can effectively model the relationship between the target sentence and the context, and incorporate external knowledge, which can effectively enrich the implicit sentiment expression of the sentence and improve the accuracy of implicit sentiment analysis.

3.1 Problem Definition

For a given sentence $S =< C, T >$, where C is context and T is target sentence, our goal is to judge the sentiment polarity of the target sentence T based on the context C and external knowledge fusion when the target sentence has no sentimental words. For example, for the previously mentioned implicit sentiment texts E1 and E2, E1-S1 and E2-S1 are target sentences T, and E1 and E2 are the

contexts C of E1-S1 and E2-S1, respectively. The other two implicit sentiment texts E3 and E4 contain only target sentences, and we choose themselves as their contexts. In sections 3.2 and 3.3 below, we elaborate on our proposed model KC-ISA to solve this problem.

3.2 Structure of KC-ISA Model

The overall structure of KC-ISA model is shown in Figure 1. From the figure, we can see that KC-ISA model is divided into three parts: the Contextual Features Fusion module, the Knowledge Fusion module and the BiAffine module.

Contextual Features Fusion module

(1) **Input layer:** Segment the context C and the target sentence T, and then convert each word in the sentence into a vector through the pre-trained Baidu Encyclopedia embedding (Li et al., 2018) to obtain $S_c = \{w_{c1}, w_{c2}, ..., w_{cn_c}\} \in R^{n_c \times d_e}$ and $S_t = \{w_{t1}, w_{t2}, ..., w_{tn_t}\} \in R^{n_t \times d_e}$, where n_c is the maximum length of the context, n_t is the maximum length of the target sentence, and d_e is the dimension of embedding.

(2) **BILSTM layer:** The BILSTM is commonly used to deal with various tasks in natural language processing, and the use of BILSTM can effectively capture the relationship of each word in a sentence. We encode the obtained S_c and S_t sentences through the BILSTM layer and get $H_c = \{h_{c1}, h_{c2}, ..., h_{cn_c}\} \in R^{n_c \times 2d}$ and $H_t = \{h_{t1}, h_{t2}, ..., h_{tn_t}\} \in R^{n_t \times 2d}$ via formula (1),

$$h_n = BILSTM(h_{n-1}, h_{n+1}, w_n) \quad (1)$$

where w_n is trainable parameter, d is the number of BILSTM hidden layer units.

(3) **Coattention layer:** Coattention is a complex attention mechanism proposed by Xiong et al. (2017), and used to solve the problems in question answering. Taking reading comprehension as an example, Coattention is like a skill that we use when doing reading comprehension: read with questions. Look at the questions first, and then go to the targeted reading of the text to find the answers. In this module, we use the Coattention mechanism to extract effective information from the context to enrich the target sentence. The calculation process of Coattention is as follows:

First, convert the previously obtained H_c and H_t through a layer of nonlinear network to obtain the encoder vector via formula (2) and (3),

$$H_c = tanh(W_c H_c + b_c) \quad (2)$$

$$H_t = tanh(W_t H_t + b_t) \quad (3)$$

where W_c, W_t, b_c and b_t are trainable parameters.

Second, use H_c and H_t to calculate the correlation matrix L according to formula (4).

$$L = H_c(H_t)^T \in R^{n_c \times n_t} \quad (4)$$

Third, the correlation matrix L can be used to calculate the attention score A_c of each word in the context C to the target sentence T, and the attention score A_t of each word in the target sentence T to each word in the context C via formula (5) and (6).

$$A_c = softmax(L) \in R^{n_c \times n_t} \quad (5)$$

$$A_t = softmax(L^T) \in R^{n_t \times n_c} \quad (6)$$

Then we get the corrected vector C_c of the context after the attention calculation via formula (7).

$$C_c = A_c H_t \in R^{n_c \times 2d} \quad (7)$$

Moreover, we use the vector of the context H_c to concatenate the corrected vector C_c, and then obtain the corrected target sentence vector C_t after the attention calculation via formula (8),

$$C_t = A_t[H_c; C_c] \in R^{n_t \times 4d} \quad (8)$$

where C_t is the co-dependent representation of the context C and the target sentence T, also known as the Coattention vector of the context and the target sentence.

Finally, a layer of BILSTM is used to fuse the original target sentence H_t with the Coattention vector C_t for getting the final vector U_t of fusion context and target sentence information via formula (9) and (10).

$$u_{tn} = BILSTM(u_{n-1}, u_{n+1}, [c_{tn}; h_{tn}]) \quad (9)$$

$$U_t = \{u_{t1}, u_{t2}, ..., u_{tn_t}\} \in R^{n_t \times 2d} \quad (10)$$

Knowledge Fusion module

We choose XLORE (Wang et al., 2013) bilingual encyclopedia knowledge graph as an external knowledge base. XLORE is the first large-scale knowledge graph that balances Chinese and English knowledge. It currently contains more than 2.3 million concepts, more than 500,000 relationships, and more than 25 million instances, covering most of the common sense concepts. In addition to

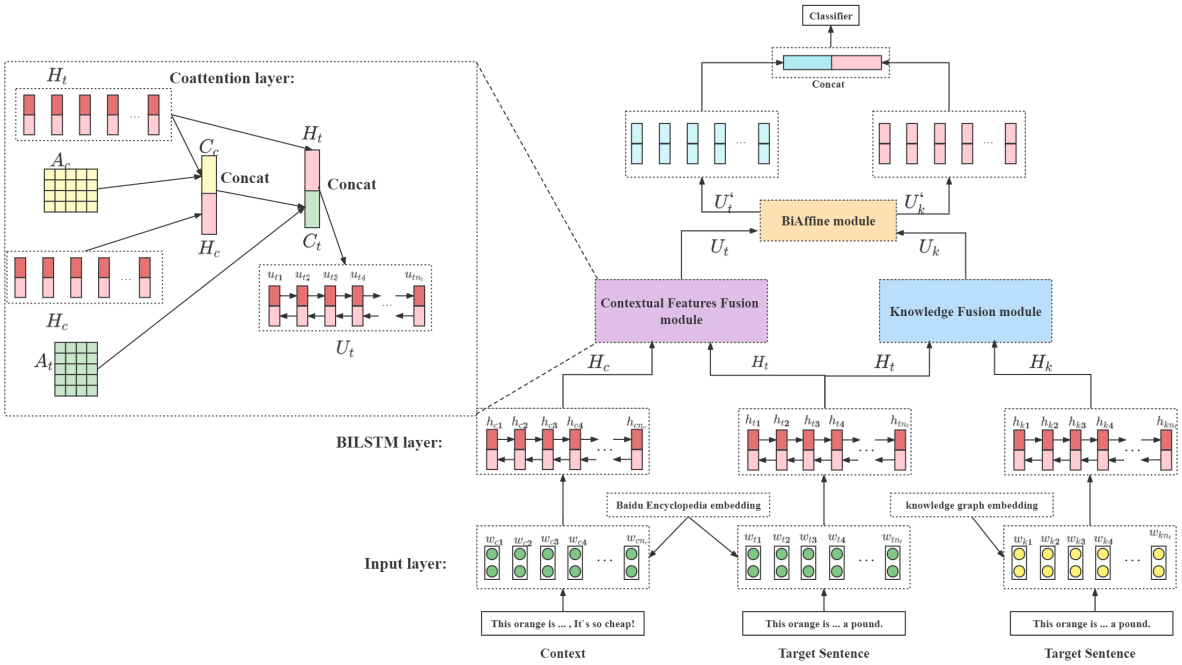

Figure 1: The overall structure of KC-ISA model

selecting the knowledge graph, it is also necessary to obtain the representation vector of knowledge. We use the openKE (Han et al., 2018) platform and select the TransE method to train the entities in XLORE and obtain the vector representation of knowledge entities.

(1) **Input layer:** The target sentence T is segmented, and then each word is searched for the corresponding vector in the trained knowledge graph representation vector. If there is no corresponding entity, this word vertor is set to 0, and $S_k = \{w_{k1}, w_{k2}, ..., w_{kn_t}\} \in R^{n_t \times d_k}$ is obtained, where n_t is the maximum length of the target sentence, d_k is the dimension of the trained knowledge graph vector.

(2) **BILSTM layer:** The function and calculation method of the BILSTM layer here are consistent with the Contextual Features Fusion module. After calculation, we can get $H_k = \{h_{k1}, h_{k2}, ..., h_{k_{nt}}\} \in R^{n_t \times 2d}$, where d is the number of BILSTM hidden layer units.

(3) **Coattention layer:** The Coattention mechanism is also used to fuse the features of the external knowledge and the target sentence. The calculation method is basically the same as that of the Contextual Features Fusion module. The context representation vector H_c is replaced with the knowledge representation vector H_k. After the above calculation, the final fusion of external knowledge and target sentence information vector U_k is obtained.

$$U_k = \{u_{k1}, u_{k2}, ..., u_{kn_t}\} \in R^{n_t \times 2d} \quad (11)$$

BiAffine module

To effectively exchange relevant features between the Contextual Features Fusion module and Knowledge Fusion module, we adopt a mutual BiAffine (Li et al., 2021a) transformation as a bridge. We formulate the process as follows:

$$U_k^{'} = softmax(U_k W_1 (U_t)^T) U_t \quad (12)$$

$$U_t^{'} = softmax(U_t W_2 (U_k)^T) U_k \quad (13)$$

where W_1, W_2 are trainable parameters. Finally, we contact the obtained $U_t^{'}$ and $U_k^{'}$ vectors and input it into the classifier for classification, and obtain the final sentiment polarity of the target sentence.

3.3 KC-ISA-BERT

In natural language processing tasks, the pre-trained language model is a topic that cannot be avoided. In addition to the above-mentioned use of BILSTM to process vectors, we also combine the pre-trained model BERT to propose KC-ISA-BERT, and compare it with the native BERT model, which fully proves the effectiveness of KC-ISA mechanism combining context and external knowledge. The calculation process is mainly similar to

KC-ISA model, replacing the BILSTM layer with the BERT structure. The position of BERT in the KC-ISA-BERT is shown in Figure 2.

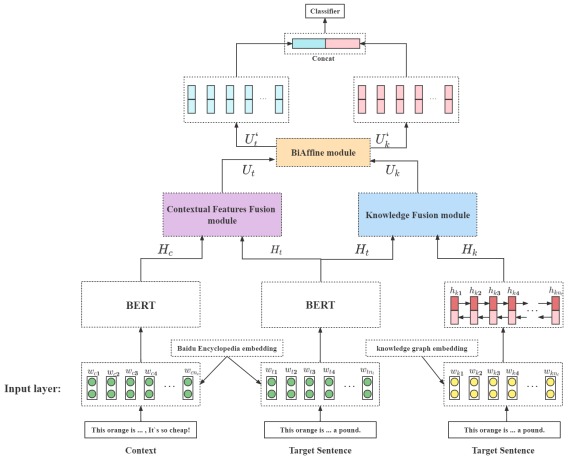

Figure 2: The overall structure of KC-ISA-BERT model

4 Experiment and Analysis

4.1 Datasets and Evaluation Metric

(1) **Datasets:** The main goal of this paper is to improve the accuracy of implicit sentiment analysis. We verify the effectiveness of KC-ISA model on the Chinese implicit sentiment analysis dataset of the "Tuers Cup" held at the 8th Social Media Processing Conference (SMP2019). The data involves Weibo, travel websites, forums, and other fields. Sentiment labels are divided into three types: positive, neutral, and negative. The detailed statistics of the dataset are presented in Table 1.

Subset	Positive	Neutral	Negative	Total
training	3391	6403	3561	13355
development	835	1577	877	3289
testing	835	1577	877	3289

Table 1: The proportion of the implicit sentiment dataset.

In addition, in order to verify the generality and effectiveness of our proposed model, two common datasets are selected for verification. The first is the NLPCC2014 Sentiment Classification with Deep Learning dataset, which is a binary classification without context. The specific division is shown in Table 2. The second is the NLPCC2014 Emotion Analysis in Chinese Weibo Texts dataset. This dataset is a 7-category dataset containing context. In this dataset, some sentences have two labels, which are primary emotion and secondary emotion. We choose the primary emotion as the final label of the dataset. The division of this dataset is shown in Table 3.

Subset	Positive	Negative	Total
training	3350	3350	6700
development	825	825	1650
testing	825	825	1650

Table 2: The proportion of the Sentiment Classification with Deep Learning dataset.

(2) **Evaluation Metric:** We choose the $F1-score$ as the evaluation metric of the model. The specific calculation method is shown in formulas (14) and (15),

$$F-macro = \frac{1}{N}\sum_{i=1}^{N} F1_i \tag{14}$$

$$F1_i = \frac{2 * P_i * R_i}{P_i + R_i} \tag{15}$$

where i is the sentiment polarity and P_i and R_i are the precision and recall, respectively, of instances with sentiment polarity i.

4.2 Implementation Details

In KC-ISA, the initialization embedding of the neural network selects Baidu Encyclopedia pre-training embedding (Li et al., 2018) with dimension 300. The number of BISLTM layers is 2 and the hidden unit size is 256. The learning rate is 0.001, the maximum length of the context is 300, and the maximum length of the target sentence is 128. The batch size is selected as 128, and the dropout is 0.6. In KC-ISA-BERT, we use the BERT-Adam optimizer, the dropout is set to 0.5, the number of neural units is 768, which are the default configurations of BERT, and the batch size is 16.

4.3 Baselines

We have selected some models that perform well in sentiment analysis as the baselines. The description and implementation details are as follows:

(1) **DPCNN (Johnson and Zhang, 2017):** DPCNN continuously deepens the network by stacking convolution modules and negative sampling layers, which can effectively extract dependencies in long-distance text. The dropout is set to 0.5, the number of convolution channels is 256, the batch size is 128, and the learning rate is 0.001.

(2) **TextCNN (Kim, 2014):** TextCNN has a simple structure, but it is very effective in processing

Subset	Like	Disgust	Fear	Surprise	Sadness	Happiness	Anger	Total
training	2854	2097	200	550	1660	1879	1272	10512
development	702	516	50	135	409	463	314	2589
testing	703	517	49	135	409	463	313	2589

Table 3: The proportion of the Emotion Analysis in Chinese Weibo Texts dataset.

Models	Implicit Sentiment datasets		Sentiment Classification dataset		Chinese Weibo Texts dataset	
	Accuracy	F1	Accuracy	F1	Accuracy	F1
DPCNN	0.759	0.735	0.773	0.773	0.513	0.438
TextCNN	0.762	0.721	0.764	0.764	0.576	0.545
TextRCNN	0.769	0.738	0.756	0.756	0.538	0.474
BILSTM	0.755	0.723	0.750	0.750	0.524	0.454
BILSTM+Attention	0.761	0.734	0.760	0.759	0.547	0.465
Transformer	0.760	0.728	0.740	0.740	0.543	0.450
MPOA(ELMo)*	-	0.680	-	-	-	-
MPOA(random)*	-	0.675	-	-	-	-
MPOA(TENCE)*	-	0.733	-	-	-	-
C-MPOA(ELMo)*	-	0.694	-	-	-	-
C-MPOA(random)*	-	0.690	-	-	-	-
C-MPOA(TENCE)*	-	0.746	-	-	-	-
our KC-ISA	**0.786**	**0.755**	**0.776**	**0.775**	**0.687**	**0.608**
BERT(target sentence only)	0.803	0.787	0.822	0.822	0.648	0.595
MPOA(BERT)*	-	0.773	-	-	-	-
our KC-ISA-BERT	**0.836**	**0.817**	**0.828**	**0.828**	**0.724**	**0.687**

Table 4: The performance of implicit and explicit sentiment analysis..

short text sentiment analysis. The dropout is set to 0.5, the number of convolution channels is 128, the size of the convolution kernel is a mixed convolution of 2 and 3, the batch size is 128, and the learning rate is 0.001.

(3) **TextRCNN (Lai et al., 2015):** TextRCNN uses a bidirectional recurrent structure to obtain contextual information, which can reduce noise more than traditional neural networks, and can preserve word order in a large range when learning text expressions. Then, TextRCNN uses the maximum pooling layer to obtain important parts of the text, and automatically determine which feature plays a more important role in the text classification process. The dropout is 0.5, the learning rate is 0.001, the number of hidden layer units is 256, and the number of hidden layers is 2.

(4) **BILSTM (Graves and Schmidhuber, 2005):** BILSTM is widely used in natural language processing tasks. In this paper, the parameters of the BILSTM model are dropout of 0.5, learning rate of 0.001, the number of hidden layer units is 256, and the number of hidden layers is 2.

(5) **BILSTM+Attention (Zhou et al., 2016):** BILSTM with an attention mechanism has been the most popular and effective model in recent years. The parameters of the BILSTM+Attention model are dropout of 0.5, learning rate of 0.001, the number of hidden layer units is 256, and the number of hidden layers is 2.

(6) **Transformer (Vaswani et al., 2017):** Transformer is a model structure that avoids recurrent and completely relies on the attention mechanism to model the global dependencies of input and output. Because the modeling of dependencies relies entirely on the attention mechanism, the attention mechanism used by the Transformer is called self-attention. The number of hidden layer units in the Transformer model is 1024, the last hidden size is 512, the number of attention heads is 5, and the number of layers of the encoder is 2.

(7) **MPOA (Wei et al., 2020):** MPOA is a BILSTM model with multi-polarity orthogonal attention. Compared with traditional single attention model, multi-polarity attention can identify the differences between words and sentiment orientation, which can be seen as an important feature of implicit sentiment analysis.

(8) **C-MPOA (Wang et al., 2021):** C-MPOA is a method for identifying implicit sentiment sentences with contextual information. First, the model embed implicit sentiment sentence representations by using a multi-polar orthogonal attention representation model. Second, multi-polarity attention layers are established by integrating context and modeling the context information to mine the key information

in the context. Finally, the contextual information representation and the implicit sentiment sentence representation are spliced together to make up for the lack of information in the implicit sentiment sentence itself.

(9) **BERT (Devlin et al., 2019):** BERT is a pre-trained language representation model. It no longer uses the traditional one-way language model or the method of shallow splicing two one-way language models for pre-training as before but uses a new masked language model, so that it can generate deep bidirectional language representations. BERT has an excellent performance in various natural language processing tasks.

4.4 Experiment and Analysis

To evaluate KC-ISA model, we used the $F1-score$ as the main evaluation metric. The main experimental results are reported in Table 4. The experimental results with * represent references to other papers. Since we could not obtain the test set of the SMP2019 dataset, we divided the training set and the development set in equal proportions, so the dataset is slightly different from that used by MPOA and C-MPOA. Our KC-ISA achieves the best results on implicit sentiment analysis, with an $F1-score$ of 0.755. At the same time, the two common sentiment datasets have also been significantly improved, especially the Chinese Weibo Texts dataset containing context, which has been improved from 0.545 of TextCNN to 0.608. There are improvements on all three datasets, proving the generality of KC-ISA model.

In the use of the pre-trained model, we directly concatenated the context on both sides of the target sentence, and found that the effect is worse than directly analyzing the target sentence. It is speculated that the direct concatenation of the context will introduce noise. Therefore, we also conducted an experiment on the impact of the size of the concatenation of the context on the analysis of the target sentence. The experimental results are shown in Table 5. It can be seen from Table 5 that if the context is directly concatenated into both sides of the target sentence without processing, noise will be introduced, which affects the analysis of the target sentence, and as the length of contexts increases, the $F1-score$ is lower. So we choose BERT that only contains the target sentence as the baseline. From the bottom three rows of Table 4, we can see that our KC-ISA-BERT has a significant improvement over native BERT, especially for sentences with context, and the $F1-score$ is increased from 0.595 to 0.687, indicating that our KC-ISA-BERT can effectively utilize external knowledge and contextual features.

Models	Accuracy	F1
BERT(target sentence only)	0.803	0.787
BERT(contains one sentence of context)	0.785	0.760
BERT(contains two sentences of context)	0.789	0.759
BERT(contains three sentences of context)	0.791	0.756

Table 5: The impact of the size of the splicing context on implicit sentiment dataset.

Models	Accuracy	F1
BILSTM	0.755	0.723
KC-ISA(Knowledge Fusion module only)	0.760	0.730
KC-ISA(Contextual Features Fusion module only)	0.780	0.748
KC-ISA(Without BiAffine module)	0.780	0.753
KC-ISA	0.786	0.755

Table 6: Experimental results of ablation study on implicit sentiment dataset.

4.5 Ablation Study

To further investigate the role of modules in KC-ISA model, we conducted extensive ablation studies on the implicit sentiment dataset. The results are reported in Table 6. KC-ISA (Knowledge Fusion module only) is a combination of BILSTM and Knowledge Fusion module. It can be seen that integrating external knowledge can bring a 0.007 $F1-score$ improvement. KC-ISA (Contextual Features Fusion module only) is a combination of BILSTM and Contextual Features Fusion module. It can be seen that integrating contextual features can bring an $F1-score$ improvement of 0.025, which is the largest improvement among all modules. KC-ISA (Without BiAffine module) is a complete KC-ISA model without BiAffine module, which reduces the interaction between external knowledge and context, which will cause a certain decline. Overall, our KC-ISA with all modules achieves the best performance.

4.6 Discussion

Now there are some review data that contain both positive and negative sentiments, so there are certain limitations to summarizing the sentiments of a sentence with one label. For example, " The scenery here is like the West Lake, but I waited in line for two hours to buy tickets". The first sentence expresses an implicit positive sentiment, i.e.

the affirmation of the scenery. The second sentence expresses an implicit negative sentiment. For explicit sentiment analysis, there have been many research results. However, for implicit sentiment analysis, it involves different contexts, especially the dependency of different external knowledge. Therefore, our next research endeavour is to extract multiple, more fine-grained implicit sentimental tags in a sentence on the one hand, and extract the aspects corresponding to the implicit sentiments in a sentence on the other hand, so that the model can better identify the various emotions contained in the text.

Secondly, for potential ethical implications and issues, different languages have different expression habits, involving cultural backgrounds, religious beliefs, and even ethics. Especially, when applying the model to individual's requirements such as conversations, chats, and psychological comfort, except for above common factors, there are personal language biases, so the issues need to be considered more.

5 Conclusion

In this paper, we summarized three distinct dependencies of implicit sentiment analysis: context dependency, sentiment target dependency, and knowledge and common sense dependency. And the XLORE bilingual encyclopedia knowledge graph was selected as an external knowledge base to supplement a large amount of common sense omitted in implicit emotional sentences. Finally, we proposed KC-ISA model for the distinct dependencies of implicit sentiment analysis, and combined the pre-trained model BERT to proposed KC-ISA-BERT, which can effectively integrate contextual features and external knowledge. To verify the effectiveness and generality of the model, experiments are conducted on an implicit sentiment analysis dataset and two common sentiment analysis datasets, and the experiments show that our model outperforms baselines.

Acknowledgements

We would like to thank the reviewers for their constructive comments. The work was supported by National Natural Science Foundation of China (62172086, 61872074, 62106039, 62272092)

References

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT 2019, Minneapolis, MN, USA, June 2-7, 2019, Volume 1 (Long and Short Papers)*, pages 4171–4186. Association for Computational Linguistics.

Alex Graves and Jürgen Schmidhuber. 2005. Framewise phoneme classification with bidirectional LSTM and other neural network architectures. *Neural Networks*, 18(5-6):602–610.

Stephan Greene and Philip Resnik. 2009. More than words: Syntactic packaging and implicit sentiment. In *Human Language Technologies: Conference of the North American Chapter of the Association of Computational Linguistics, Proceedings, May 31 - June 5, 2009, Boulder, Colorado, USA*, pages 503–511. The Association for Computational Linguistics.

Xu Han, Shulin Cao, Xin Lv, Yankai Lin, Zhiyuan Liu, Maosong Sun, and Juanzi Li. 2018. Openke: An open toolkit for knowledge embedding. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing, EMNLP 2018: System Demonstrations, Brussels, Belgium, October 31 - November 4, 2018*, pages 139–144. Association for Computational Linguistics.

Rie Johnson and Tong Zhang. 2017. Deep pyramid convolutional neural networks for text categorization. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics, ACL 2017, Vancouver, Canada, July 30 - August 4, Volume 1: Long Papers*, pages 562–570. Association for Computational Linguistics.

Yoon Kim. 2014. Convolutional neural networks for sentence classification. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing, EMNLP 2014, October 25-29, 2014, Doha, Qatar, A meeting of SIGDAT, a Special Interest Group of the ACL*, pages 1746–1751. ACL.

Siwei Lai, Liheng Xu, Kang Liu, and Jun Zhao. 2015. Recurrent convolutional neural networks for text classification. In *Proceedings of the Twenty-Ninth AAAI Conference on Artificial Intelligence, January 25-30, 2015, Austin, Texas, USA*, pages 2267–2273. AAAI Press.

Ruifan Li, Hao Chen, Fangxiang Feng, Zhanyu Ma, Xiaojie Wang, and Eduard H. Hovy. 2021a. Dual graph convolutional networks for aspect-based sentiment analysis. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing, ACL/IJCNLP 2021, (Volume 1:*

Long Papers), Virtual Event, August 1-6, 2021, pages 6319–6329. Association for Computational Linguistics.

Shen Li, Zhe Zhao, Renfen Hu, Wensi Li, Tao Liu, and Xiaoyong Du. 2018. Analogical reasoning on chinese morphological and semantic relations. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics, ACL 2018, Melbourne, Australia, July 15-20, 2018, Volume 2: Short Papers*, pages 138–143. Association for Computational Linguistics.

Zhengyan Li, Yicheng Zou, Chong Zhang, Qi Zhang, and Zhongyu Wei. 2021b. Learning implicit sentiment in aspect-based sentiment analysis with supervised contrastive pre-training. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing, EMNLP 2021, Virtual Event / Punta Cana, Dominican Republic, 7-11 November, 2021*, pages 246–256. Association for Computational Linguistics.

Jian Liao, Suge Wang, and Deyu Li. 2019. Identification of fact-implied implicit sentiment based on multi-level semantic fused representation. *Knowl. Based Syst.*, 165:197–207.

Bing Liu. 2012. *Sentiment Analysis and Opinion Mining*. Synthesis Lectures on Human Language Technologies. Morgan & Claypool Publishers.

Zhenghao Liu, Chenyan Xiong, Maosong Sun, and Zhiyuan Liu. 2018. Entity-duet neural ranking: Understanding the role of knowledge graph semantics in neural information retrieval. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics, ACL 2018, Melbourne, Australia, July 15-20, 2018, Volume 1: Long Papers*, pages 2395–2405. Association for Computational Linguistics.

Yu Sun, Shuohuan Wang, Yu-Kun Li, Shikun Feng, Xuyi Chen, Han Zhang, Xin Tian, Danxiang Zhu, Hao Tian, and Hua Wu. 2019. ERNIE: enhanced representation through knowledge integration. *CoRR*, abs/1904.09223.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, December 4-9, 2017, Long Beach, CA, USA*, pages 5998–6008.

Hongwei Wang, Fuzheng Zhang, Jialin Wang, Miao Zhao, Wenjie Li, Xing Xie, and Minyi Guo. 2018. Ripplenet: Propagating user preferences on the knowledge graph for recommender systems. In *Proceedings of the 27th ACM International Conference on Information and Knowledge Management, CIKM 2018, Torino, Italy, October 22-26, 2018*, pages 417–426. ACM.

S. Wang, M. Wang, J. Liao, and X. Chen. 2021. An implicit sentiment sentence identification method base on context information. *School of Computer Information Technology, Shanxi University, Taiyuan 030006, China*, pages 1–7.

Zhigang Wang, Juanzi Li, Zhichun Wang, Shuangjie Li, Mingyang Li, Dongsheng Zhang, Yao Shi, Yongbin Liu, Peng Zhang, and Jie Tang. 2013. Xlore: A large-scale english-chinese bilingual knowledge graph. In *Proceedings of the ISWC 2013 Posters & Demonstrations Track, Sydney, Australia, October 23, 2013*, volume 1035 of *CEUR Workshop Proceedings*, pages 121–124. CEUR-WS.org.

Zikang Wang, Linjing Li, and Daniel Zeng. 2020. Knowledge-enhanced natural language inference based on knowledge graphs. In *Proceedings of the 28th International Conference on Computational Linguistics, COLING 2020, Barcelona, Spain (Online), December 8-13, 2020*, pages 6498–6508. International Committee on Computational Linguistics.

Jiyao Wei, Jian Liao, Zhenfei Yang, Suge Wang, and Qiang Zhao. 2020. Bilstm with multi-polarity orthogonal attention for implicit sentiment analysis. *Neurocomputing*, 383:165–173.

Caiming Xiong, Victor Zhong, and Richard Socher. 2017. Dynamic coattention networks for question answering. In *5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Conference Track Proceedings*. OpenReview.net.

Zhengyan Zhang, Xu Han, Zhiyuan Liu, Xin Jiang, Maosong Sun, and Qun Liu. 2019. ERNIE: enhanced language representation with informative entities. In *Proceedings of the 57th Conference of the Association for Computational Linguistics, ACL 2019, Florence, Italy, July 28- August 2, 2019, Volume 1: Long Papers*, pages 1441–1451. Association for Computational Linguistics.

Deyu Zhou, Jianan Wang, Linhai Zhang, and Yulan He. 2021. Implicit sentiment analysis with event-centered text representation. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing, EMNLP 2021, Virtual Event / Punta Cana, Dominican Republic, 7-11 November, 2021*, pages 6884–6893. Association for Computational Linguistics.

Peng Zhou, Wei Shi, Jun Tian, Zhenyu Qi, Bingchen Li, Hongwei Hao, and Bo Xu. 2016. Attention-based bidirectional long short-term memory networks for relation classification. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics, ACL 2016, August 7-12, 2016, Berlin, Germany, Volume 2: Short Papers*. The Association for Computer Linguistics.

Enguang Zuo, Hui Zhao, Bo Chen, and Qiuchang Chen. 2020. Context-specific heterogeneous graph convolutional network for implicit sentiment analysis. *IEEE Access*, 8:37967–37975.

Domain Generalization for Text Classification with Memory-Based Supervised Contrastive Learning

Qingyu Tan[*1,2] **Ruidan He**[†1] **Lidong Bing**[1] **Hwee Tou Ng**[2]
[1]DAMO Academy, Alibaba Group
[2]Department of Computer Science, National University of Singapore
{qingyu.tan,ruidan.he,l.bing}@alibaba-inc.com
{qtan6,nght}@comp.nus.edu.sg

Abstract

While there is much research on cross-domain text classification, most existing approaches focus on one-to-one or many-to-one domain adaptation. In this paper, we tackle the more challenging task of domain generalization, in which domain-invariant representations are learned from multiple source domains, without access to any data from the target domains, and classification decisions are then made on test documents in unseen target domains. We propose a novel framework based on supervised contrastive learning with a memory-saving queue. In this way, we explicitly encourage examples of the same class to be closer and examples of different classes to be further apart in the embedding space. We have conducted extensive experiments on two Amazon review sentiment datasets, and one rumour detection dataset. Experimental results show that our domain generalization method consistently outperforms state-of-the-art domain adaptation methods[1].

1 Introduction

Text classification is a highly important and widely studied natural language understanding task. Recent success in self-supervised pre-training has significantly improved the state-of-the-art performance in sentiment analysis. However, sentiment classification is widely known as a domain-dependent task, mainly because sentiment expressions can have different meanings in different domains (e.g., "long" in "long waiting time" of a restaurant review is negative, while in "long battery life" of a laptop review is positive). Moreover, the amount of labeled data is highly imbalanced across different domains. Many NLP domains still lack sufficient labeled data for training a high-performance classifier. Therefore, it is crucial to adapt sentiment knowledge from resource-rich domains to low-resource domains. This strand of research is known as domain adaptation (DA). Prior works on domain adaptation typically follow a one-to-one (Jiang and Zhai, 2007) or many-to-one adaptation setting (Zhao et al., 2018), where the model is usually trained on labeled data from the source domain along with unlabeled data from the target domain and is then evaluated on the target domain data. The usage of the unlabeled target data is crucial in DA. Prior works mainly use it for domain-invariant representation learning or model selection. More recently, Wright and Augenstein (2020) have demonstrated that large pretrained language models (PrLMs) are able to achieve promising performance for cross-domain sentiment classification. Subsequent works further improve the adaptation performance through domain adversarial training (Du et al., 2020; Karouzos et al., 2021), iterative pseudo-labeling (Ye et al., 2020), or prompting (Ben-David et al., 2022), where they all adapt the DA setting and assume unlabeled target data is available during model training.

However, in more realistic scenarios, one may be asked to build a text classification model that will be applied to unknown target domains, which implies unlabeled target domain data is unavailable during training or model selection. Domain generalization (Li et al., 2018;Wang et al., 2021b;Wang et al., 2021a) has been proposed to address this problem by learning a universal representation using labeled data from multiple source domains, without access to target domain data. Building such a generalized model enables us to predict sentiment polarity of emerging domains, such as COVID-19 vaccines and pandemic-related medical equipment. Compared to domain adaptation, the major advantage of domain generalization is that one trained model can be used for all target domains, whereas domain adaptation needs to train

[*] Qingyu Tan is under the Joint PhD Program between Alibaba and National University of Singapore.

[†] Corresponding author.

[1]Our code is available at https://github.com/tonytan48/MSCL.

Proceedings of the 29th International Conference on Computational Linguistics, pages 6916–6926
October 12–17, 2022.

a tailored model for each target domain. To the best of our knowledge, even though there are many works that focused on multi-source domain adaptation (Zhao et al., 2018; Guo et al., 2020; Wright and Augenstein, 2020), there is no prior work that tackles domain generalization in the context of text classification.

In this work, we focus on domain generalization for text classification. We aim to build a generalized sentiment classifier using labeled source data from multiple domains. The trained model can be applied to other unseen domains. We train a classifier to learn a joint hypothesis over all source domain data. On this basis, we propose to use supervised contrastive learning (SCL) to better capture the similarity between examples from different domains but belong to the same sentiment class and contrast them with examples belonging to the other class. SCL explicitly pulls the representations from the same class together and repulses the representations from different classes in the embedding space. This objective helps the model better extract the domain invariant features among all source domains, thereby allowing a better classification decision boundary. The usage of SCL helps the classifier achieve better generalization ability, which is especially beneficial to the domain generalization scenario. Although SCL has previously been applied to image classification tasks (Khosla et al., 2020) and in-domain sentence-level classification tasks (Gunel et al., 2021), to the best of our knowledge, we are the first to apply it in the context of domain generalization. It is shown that the performance of SCL is highly affected by batch size as it requires a large number of contrasting examples for computing the contrastive loss (Khosla et al., 2020;Gunel et al., 2021). An optimal batch size requires large memory, making it impractical in many use cases. To apply SCL without excessive memory consumption, we further propose to use a memory bank to store the representations to increase the size of contrasting features, so that the hidden representations for sentiment classification will be reused for computing supervised contrastive loss. In this way, we can significantly improve performance compared to directly applying SCL for text classification.

To examine our proposed method, we conducted domain generalization experiments on two popular Amazon review datasets (one monolingual and the other multilingual) and on the PHEME rumour detection dataset. Following previous works (Chen and Cardie, 2018; Ye et al., 2020; Li et al., 2020; Liu et al., 2021), different languages can be seen as distinct domains based on a shared cross-lingual encoder. Hence, the domain generalization problem can also be extended to language generalization. We conducted experiments in cross-domain (CD), cross-language (CL), and cross-language cross-domain (CLCD) settings. In our experiments, our proposed method is able to outperform the second best domain adaptation method by 0.81% accuracy score in sentiment analysis and 1.68% F1 score in rumour detection.

Our contributions can be summarized as follows:

- We are the first to tackle the domain generalization problem for text classification.
- We proposed a novel memory-based alternative for supervised contrastive learning to improve its performance. Experimental results show that our proposed method consistently outperforms state-of-the-art domain adaptation baselines.

2 Related Work

2.1 Domain Adaptation

Prior works on domain adaptation mainly focus on minimizing the distributional discrepancy between the source domain and the target domain. Kernelized methods, such as Maximum Mean Discrepancy (MMD) (Arbel et al., 2019), spectral feature alignment (Pan et al., 2010), and domain-adversarial training (Ganin et al., 2016) are commonly used for feature alignment from different domains. Another widely explored approach for DA is self-training, where a classifier is first trained on the source domain and later used for predicting pseudo-labels for unlabeled data in the target domain. Ye et al. (2020) have proposed a robust self-training approach to improve the performance of the joint hypothesis between source and target domains and thereby improve performance on the target domain.

When multiple source domains are present, DA methods should be modified accordingly. Li et al. (2018) have used pairwise-MMD and an adversarial autoencoder to overcome domain discrepancy. Wu and Huang (2016) extended domain-adversarial training from Ganin et al. (2016) to multiple source domains. Domain adversarial training has many popular variants (Zhao et al., 2018;

Liu et al., 2018; Chen et al., 2018), showing that extracting domain-invariant representation is a crucial part for domain adaptation. Another strand of work for multi-source DA is based on mixture of experts (MoE). Chen and Cardie (2018) have explored MoE for multi-source cross-lingual sentiment classification, and MoE encourages the model to learn from more relevant source languages. Guo et al. (2020) proposed a DistanceNet-Bandits approach to tackle multi-source DA. It first measures domain distance with multiple metrics, and then uses a multi-armed bandit mechanism to learn from closer source domains. However, prior works on multi-source domain adaptation rely on distance measurement from source to target domain. Therefore, even though no labeled data from the target domain is used, unlabeled target data must be used to perform domain adaptation. Unlike multi-source domain adaptation, domain generalization has no access to any target domain data during training. In many practical scenarios, we may need to find people's opinions towards emerging and unseen domains, such as pandemic-related medical equipment. Therefore, it is important to study the problem of domain generalization for sentiment and text classification.

2.2 Contrastive Learning

Recently, contrastive learning (CL) has led to major advances in self-supervised representation learning. The common idea in these works is maximizing the agreement score between an anchor and a 'positive' example in the embedding space, and pushing apart the anchor from many 'negative' examples (Chen et al., 2020). The positive example pair is typically a particular image and its augmentation, and the negative pairs are formed by that image with other images within the same batch. However, such approaches require a substantially large batch size. Otherwise the performance of contrastive learning deteriorates significantly. On the other hand, other approaches have been proposed to alleviate resource consumption due to a large batch size. Grill et al. (2020) have shown superior performance by maximizing the agreement of positive pairs by a momentum encoder, without the need of negative samples. Khosla et al. (2020) have extended the idea of contrastive learning to the supervised setting, i.e., supervised contrastive learning (SCL). To leverage label information, SCL considers examples with the same label as positive pairs and examples with different labels as negative pairs, and achieves significant performance gain in image classification tasks. Gunel et al. (2021) extended the application of SCL to finetuning pre-trained language models in natural language understanding tasks. Graf et al. (2021) further analyzed SCL in image classification problem, showing that SCL is able to increase the inter-cluster distance and reduce the intra-cluster distance for each class. We leverage the idea of SCL to explicitly align features of the same class but from different domains. Since the negative sample size is crucial for accurate mutual information estimation, we propose to use an additional memory bank to store representations and progressively reuse the encoded sentence representations, thereby improving the performance of contrastive learning. In this way, we force the domain generalization model to focus on aligning features of the same class and implicitly reduce the domain discrepancy among the multi-source training data.

3 Problem Definition

For the task of domain generalization on text classification, suppose we have labeled data from k source domains $\{\mathcal{D}_{s_i}\}_{i=1}^{k}$. For each source domain, the labeled data D_{s_i} is denoted as $D_{s_i} = \{X_{s_i}, Y_{s_i}\}$. In the training phase, only source domains are available. Then, the labeled dataset from the target domain $D_t = \{X_t, Y_t\}$ is used for evaluation. The problem setup of domain generalization is different from multi-source domain adaptation (MSDA), which requires an additional unlabeled set from the target domain during training (Wu and Huang, 2016; Ding et al., 2019; Zhao et al., 2018; Guo et al., 2020). In contrast, in the domain generalization setup, the model is only trained to obtain a domain-invariant feature from the given source domains, while the target-domain data D_t is only used during evaluation. In this way, the trained model can be used to make predictions on unseen domains.

4 Model Description

Since the domain generalization problem is to train an algorithm based on multiple source domains, the key challenge of this classification problem is to learn an ideal joint hypothesis of the source domains. That is, we aim to separate the data points by their labels as much as possible, thereby minimizing domain discrepancy within each class in

the feature embedding space. Our supervised contrastive learning model not only widens the margin of decision boundaries, but also enforces the distribution within the same class to be more uniform, hence minimizing the source domain distances for each class.

We first shuffle all the source domain data and divide the joint dataset into mini-batches. This is mainly for enforcing a stable domain distribution for each mini-batch. For one sampled mini-batch of size N, we have $S = \{x_i, y_i\}_{i=1}^{N}$, where x_i represents the input text, y_i represents the sentiment label of that example. We first adopt a pre-trained language model (PrLM) as encoder. We use the hidden state of the last layer's [CLS] token as document representation, and denote it as $\mathbf{h}$. We then use a feed-forward neural network (followed by a $\tanh$ activation function and layer norm) f for dimension reduction. Specifically, we have $\mathbf{z} = f(\mathbf{h})$. Feature $\mathbf{z}$ will later be fed into a classifier g for downstream tasks:

$$\mathcal{L}_{CE} = -\frac{1}{N}\sum_{i=1}^{N} y_i \log\big(g(\mathbf{z}_i)\big) \tag{1}$$

where g is a fully connected classifier and $\mathcal{L}_{CE}$ is the cross-entropy classification loss.

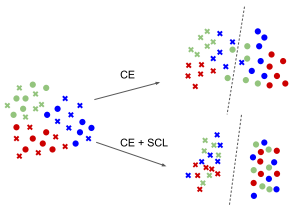

Figure 1: Illustration of SCL. ◯ refers to a positive review, × refers to a negative review, and the different colors of data points represent their domains.

4.1 Supervised Contrastive Loss for Domain Generalization

As illustrated in Figure 1, supervised contrastive learning explicitly pulls the representations of the same class together and repulses representations from different classes, which increases the discriminative capability of hidden representations and benefits hard negative mining (Khosla et al., 2020). This objective suits our goal of learning a joint hypothesis for different source domains, and enables the model to learn a more uniform distribution for each label class. Specifically, for a given mini-batch S of size N, the supervised contrastive loss is computed as follows:

$$\mathcal{L}_{SCL} = -\sum_{\mathbf{z}_i \in S} \frac{1}{N} \sum_{\mathbf{z}_p \in P(i)} \log \frac{\exp(\mathbf{z}_i \cdot \mathbf{z}_p/\tau)}{\sum_{\mathbf{z}_a \in A(i)} \exp(\mathbf{z}_i \cdot \mathbf{z}_a/\tau)} \tag{2}$$

where $\mathbf{z}$ is the vector representation. For a given anchor representation $\mathbf{z}_i$, $P(i) \equiv \{\mathbf{z}_j \in S, y_j = y_i\}$ refers to the set of positive examples, $A(i) \equiv \{\mathbf{z}_j \in S, j \neq i\}$ refers to the union of positive examples and negative examples, S refers to the set of the mini-batch. τ is a scaling hyper-parameter, also known as temperature. Then, we have our combined loss function as:

$$\mathcal{L} = \mathcal{L}_{CE} + \mathcal{L}_{SCL} \tag{3}$$

However, in our preliminary experiments, directly applying supervised contrastive learning has marginal effect for domain generalization. This is because the performance of supervised contrastive learning is heavily related to batch size, as a larger batch size is better in representing the mixed distribution of multiple domains. However, increasing the batch size inevitably introduces high computation and memory costs. In order to improve the performance of domain generalization while reducing memory consumption, we propose to use an additional memory bank to reuse the previously encoded sentence representations. The details for our memory bank are described in the following subsection.

4.2 Memory-Based Supervised Contrastive Learning

To increase the number of contrasting examples while limiting memory consumption, we propose to use a memory bank Q to store the sentence representations and their labels of each batch. The purpose of introducing this memory bank is to progressively reuse the encoded sentence to compute $\mathcal{L}_{SCL}$. The memory bank Q stores the sentence representations $\mathbf{z}$ and their corresponding labels y, $S^{'} = \{\mathbf{z}_i, y_i\}_{i=1}^{N}$, during computation of one batch. The maximum size of Q is denoted as M, which indicates that when the number of examples in Q exceeds M, only the last M examples will be kept and previous examples will be discarded. To avoid repeated computation, for each batch S, all the examples of the current batch S are deemed as anchor features. The computation of SCL with memory bank Q still follows Equation 2, except that the number of contrasting features is increased. Let S_C denote the set of contrasting features, which

is the union of the current batch S and memory bank Q, i.e., $S_C = S \cup Q$. Then the set of positive examples becomes $P(i)' \equiv \{\mathbf{z}_j \in S_C, y_j = y_i\}$ and the union of positive and negative examples becomes $A(i)' \equiv \{\mathbf{z}_j \in S_C, j \neq i\}$. Given batch size N and memory bank size M, the number of anchor features is N, whereas the number of contrasting features becomes $N + M$, since the set of contrasting features S_C is the union of the current batch S and the memory bank Q. We provide the pseudo-code for our approach in Algorithm 1.

Algorithm 1: Algorithm for supervised contrastive learning with memory bank

Input: Batch size N, encoder f, classifier g, memory bank Q

for $t \leq T_{max}$ **do**
- Sample minibatch $S = \{x_i, y_i\}_{i=1}^{N}$
- $\mathbf{z} = f(PrLM(x))$;
- $Q = None$;
- $\mathcal{L}_{CE} = \frac{1}{N}\sum_{i=1}^{N} -y_i \log\big(g(\mathbf{z}_i)\big)$;
- $S_C = S \cup Q$;
- $P(i) \equiv \{\mathbf{z}_j \in S_C, y_j = y_i\}$;
- $A(i) \equiv \{\mathbf{z}_j \in S_C, j \neq i\}$;
- $\mathcal{L}_{SCL} = \sum_{\mathbf{z}_i \in S} \frac{-1}{N} \sum_{\mathbf{z}_p \in P(i)} \log \frac{\exp(\mathbf{z}_i \cdot \mathbf{z}_p/\tau)}{\sum_{\mathbf{z}_a \in A(i)} \exp(\mathbf{z}_i \cdot \mathbf{z}_a/\tau)}$;
- $EnqueueAndDequeue(Q, \{\mathbf{z}_i, y_i\}_{i=1}^{N})$;
- $\mathcal{L} = \mathcal{L}_{CE} + \mathcal{L}_{SCL}$;
- update network by combined loss $\mathcal{L}$

end

5 Experiments

5.1 Dataset Statistics

We have conducted experiments on two benchmarks for cross-domain and cross-lingual sentiment classification and one benchmark for rumour detection.

Multi-Domain Sentiment Dataset (Blitzer et al., 2007) This dataset contains 8,000 Amazon product reviews, equally distributed from four domains: books (B), DVDs (D), kitchen and housewares (K), and electronics (E). In each domain, there are 1,000 positive and 1,000 negative reviews. We follow the split of prior works (Ganin et al., 2016; Du et al., 2020; Guo et al., 2020) for fair comparison, resulting in 1,600 training examples and 400 test examples for each domain. Since we do not have access to target domain data, training and model selection are all based on the mixture distribution of source domains.

Cross-Lingual Sentiment Dataset (Prettenhofer and Stein, 2010) This is a multi-lingual and multi-domain Amazon review dataset. It contains four languages: English (En), German (De), French (Fr), and Japanese (Jp). For all languages, there are three domains: books (B), DVDs (D), and music (M). For each domain, there are 2,000 training examples and 2,000 test examples, where each set contains 1,000 positive and 1,000 negative reviews. In summary, there are twelve language-domain combinations in this dataset.

PHEME Rumour Detection Dataset There are 5,802 annotated tweets from 5 different events ((C)harlie(H)ebdo, (F)erguson, (G)erman(W)ings, (O)ttawa(S)hooting, and (S)ydneySiege) labeled as rumour or non-rumour (1,972 rumours, 3,830 non-rumours).

5.2 Experimental Setup

We conducted experiments in cross-domain (CD), cross-language (CL), and cross-language cross-domain (CLCD) settings. For CD experiments, the model is trained on three source domains and evaluated on the target domain. Model selection is based on validation performance on the combined test set of the source domains. Using the multi-domain Amazon review dataset as an example, since there are three source domains in total, for each target domain experiment, there are 4,800 training examples.

In CL experiments, the training and test domains are the same while the source languages are different from the target language. For example, for target German-DVD, the training data will be English, French, and Japanese DVD, and the total number of training examples is 6,000. Since English is a high-resource language, we did not conduct experiments with English as the target language.

For CLCD experiments, the training data for a given language-domain combination come from both a different language and a different domain. For example, for the same target German-DVD, the training data will still be in English, French, and Japanese, while the domains for the training data will be books and music. For a fair comparison with CL experiments, the training data for CLCD experiments will be down-sampled to half of its original size, with 6,000 reviews in total.

In the monolingual domain generalization ex-

	D	E	K	B	Avg Acc	CH	F	GW	OS	S	Avg μF1
Guo et al. (2018)	87.70	89.50	90.50	87.90	88.90	-	-	-	-	-	-
Wright and Augenstein (2020)	88.90	90.30	90.80	90.00	90.00	67.90	45.40	74.50	62.60	64.70	63.02
DistilBERT											
Baseline	$89.30_{0.3}$	$89.80_{0.2}$	$89.98_{0.2}$	$89.24_{0.2}$	89.58	$64.78_{1.3}$	$43.03_{1.5}$	$69.87_{1.9}$	$60.42_{0.8}$	$62.02_{1.4}$	60.02
MMD	$89.00_{0.2}$	$89.86_{0.2}$	$89.64_{0.2}$	$89.38_{0.4}$	89.47	$63.80_{1.0}$	$43.44_{1.1}$	$69.04_{2.1}$	$63.97_{1.1}$	$63.27_{0.7}$	60.70
MoE	$89.20_{0.3}$	$89.92_{0}.3$	$90.26_{0.3}$	$89.88_{0.1}$	89.82	$65.84_{2.2}$	$43.61_{1.1}$	$72.23_{1.2}$	$61.63_{1.0}$	$64.25_{1.4}$	61.51
Intra	$88.46_{0.6}$	$89.80_{0.3}$	$90.06_{0.2}$	$89.22_{0.4}$	89.39	$64.14_{1.4}$	$42.89_{1.2}$	$70.77_{1.6}$	$61.84_{1.5}$	$62.41_{0.7}$	60.41
Adv	$88.40_{0.4}$	$89.60_{0.2}$	$90.00_{0.2}$	$89.04_{0.2}$	89.30	$64.83_{1.5}$	$42.23_{1.2}$	$65.94_{1.0}$	$61.47_{0.9}$	$62.81_{1.6}$	59.45
SCL	$89.35_{0.1}$	$89.85_{0.1}$	$90.25_{0.2}$	$89.50_{0.2}$	89.74	$65.57_{0.9}$	$43.22_{1.9}$	$73.03_{1.1}$	$63.50_{1.5}$	$63.52_{1.4}$	61.77
SCL+M=64	$\mathbf{90.10}_{0.2}$	$90.26_{0.2}$	$\mathbf{90.80}_{0.2}$	$\mathbf{89.98}_{0.2}$	**90.28**	$65.88_{0.8}$	$43.64_{1.2}$	$74.54_{1.1}$	$\mathbf{67.25}_{0.5}$	$\mathbf{65.99}_{1.7}$	63.46
SCL+M=128	$89.73_{0.3}$	$\mathbf{90.30}_{0.1}$	$90.50_{0.1}$	$90.04_{0.3}$	90.14	$\mathbf{68.08}_{1.0}$	$\mathbf{44.55}_{1.9}$	$\mathbf{75.41}_{0.8}$	$66.52_{2.0}$	$65.19_{0.8}$	**63.95**
SCL+M=256	$89.43_{0.5}$	$89.97_{0.3}$	$90.00_{0.2}$	$89.47_{0.3}$	89.72	$67.41_{0.8}$	$42.83_{0.9}$	$74.64_{0.7}$	$65.73_{2.0}$	$64.06_{1.5}$	62.93
Roberta-Large											
Basline	$90.00_{0.4}$	$93.95_{0.2}$	$93.40_{0.5}$	$92.65_{0.7}$	92.50	$67.12_{1.2}$	$\mathbf{43.97}_{2.4}$	$70.78_{3.2}$	$65.69_{2.9}$	$62.66_{1.7}$	62.04
MMD	$89.85_{0.4}$	$94.15_{0.3}$	$93.70_{0.5}$	$92.55_{0.5}$	92.56	$64.88_{1.7}$	$43.13_{1.2}$	$69.86_{2.9}$	$66.60_{0.7}$	$63.17_{2.9}$	61.53
MoE	$90.25_{0.3}$	$94.04_{0.4}$	$93.99_{0.2}$	$92.50_{0.2}$	92.69	$67.24_{1.8}$	$43.63_{2.2}$	$73.60_{1.8}$	$68.77_{2.4}$	$63.59_{1.2}$	63.38
Intra	$90.06_{0.3}$	$94.00_{0.3}$	$94.06_{0.2}$	$92.75_{0.2}$	92.72	$66.87_{2.2}$	$43.63_{2.2}$	$73.73_{1.6}$	$69.28_{1.8}$	$63.72_{1.0}$	63.44
Adv	$90.25_{0.7}$	$94.45_{0.5}$	$94.60_{0.3}$	$92.85_{0.6}$	93.04	$64.71_{2.4}$	$42.92_{1.0}$	$71.01_{1.6}$	$66.69_{2.2}$	$63.84_{3.7}$	61.83
SCL	$89.95_{0.3}$	$94.25_{0.7}$	$93.10_{0.6}$	$93.45_{0.5}$	92.69	$65.83_{2.0}$	$42.65_{1.5}$	$71.41_{1.7}$	$65.92_{1.5}$	$62.33_{2.0}$	61.63
SCL+M=64	$91.40_{0.7}$	$\mathbf{95.10}_{0.5}$	$95.05_{0.4}$	$93.25_{0.4}$	93.70	$67.44_{0.6}$	$43.53_{1.2}$	$\mathbf{75.23}_{0.5}$	$\mathbf{72.22}_{3.2}$	$\mathbf{67.18}_{2.0}$	**65.12**
SCL+M=128	$\mathbf{91.45}_{0.5}$	$\mathbf{95.10}_{0.4}$	$\mathbf{95.10}_{0.5}$	$\mathbf{93.7}_{0.5}$	**93.85**	$\mathbf{68.32}_{0.8}$	$43.39_{2.8}$	$73.89_{1.2}$	$72.01_{2.4}$	$66.70_{1.3}$	64.86
SCL+M=256	$91.10_{0.3}$	$95.05_{0.5}$	$94.90_{0.5}$	$93.40_{0.6}$	93.62	$66.81_{0.8}$	$40.91_{1.1}$	$73.52_{1.0}$	$72.21_{2.0}$	$65.59_{1.3}$	63.81

Table 1: Experimental results for cross-domain text classification. The reported metric is accuracy for sentiment analysis and micro-F1 for rumour detection. The experimental results are averaged over five runs and each subscript indicates the standard deviation of five runs.

Batch Size	**8**	**16**	**32**	**64**	**128**	**256**	**16+M=128**
Memory (GB)	10.5	14.1	21.3	35.6	64.3	121.7	14.7

Table 2: Memory consumption for training Roberta-large model for sentiment analysis. The memory sizes used above batch size of 32 are interpolated.

periments, we adopt Distil-Bert-base (Sanh et al., 2019) and RoBERTa-large (Liu et al., 2019) pre-trained models as encoders. In the cross-lingual experiments, we use XLM-R-large (Conneau et al., 2020) as encoder. We pre-process each review to 180 sentencepiece tokens. For all encoders, we set the dimension for classifier representation z to be 256. We train all the models with Adam Optimizer with learning rate of 1e-5. The batch size we use for training DistilBert/RoBERTa baselines is 16. We use batch size of 2 with 8 gradient accumulation steps for training XLM-R due to GPU memory constraint. All our experiments are conducted on Nvidia 16GB V100 GPU. We conduct grid-search for $M \in \{16, 32, 64, 128, 256, 512\}$ and $\tau \in \{0.1, 0.2, 0.5, 0.7, 0.8, 1.0\}$ in D,K,E-B transfer (i.e., D, K, E as source domains and B as the target domain) and select $M = 128$ and $\tau = 0.2$. We provide the memory consumption for training the Roberta model with different batch sizes in Table 2. We use interpolation estimate for memory usage above batch size of 32. From Table 2, we observe that our proposed method significantly saves memory consumption compared to directly using large batch sizes.

5.3 Compared Methods

We compare our methods with several state-of-the-art domain adaptation methods. However, previous works are trained with unlabeled target domain data, which is a more relaxed setup compared to ours. For fair comparison, we also employed state-of-the-art DA models in the same domain generalization setting as our baselines:

MoE (Guo et al., 2018; Wright and Augenstein, 2020): Mixture-of-Experts (MoE) models are the current state-of-the-art method for multi-source domain adaptation. It can also be applied to domain generalization. The MoE models consist of multiple models. For K source domains $\{S_i\}_1^K$, each source domain will be treated as a meta-target domain during training and each of them will have a dedicated model. The labeled data for each meta-target will be excluded during training of its model. There is an additional global encoder that is trained on labeled data from all source domains. Hence, there are $K + 1$ models in total for the MoE structure. During inference, the ensemble predictions of $K + 1$ models are aggregated.

CFd (Ye et al., 2020): This one-to-one domain adaptation approach is based on self-training. The model is first trained on a source domain, and high-confidence pseudo-labeled data in the target domain are generated for bootstrapping. We also compare to the optimal one-to-one transfer pair for this baseline.

CLDFA (Xu and Yang, 2017): This method, cross-lingual knowledge distillation on parallel corpora

	German			Avg	French			Avg	Japanese			Avg
	Book	DVD	Music		Book	DVD	Music		Book	DVD	Music	
Cross-language: *With unlabeled target data, results taken from original papers*												
CLDFA	83.95	83.14	79.02	82.04	83.37	82.56	83.31	83.08	77.36	80.52	76.46	78.11
MAN-MoE	82.40	78.80	77.15	79.45	81.10	84.25	80.90	82.08	62.78	69.10	72.60	68.16
CFd	93.95	91.69	93.89	93.18	94.25	93.79	93.39	93.81	89.41	88.68	89.54	89.21
Cross-language: *Without unlabeled target data*												
Baseline	$93.94_{0.3}$	$91.37_{0.4}$	$93.74_{0.4}$	93.02	$93.76_{0.5}$	$93.09_{0.4}$	$93.28_{0.4}$	93.38	$89.61_{0.4}$	$89.20_{0.3}$	$89.73_{0.6}$	89.51
Intra	$94.56_{0.4}$	$91.99_{0.5}$	$94.21_{0.6}$	93.59	$94.68_{0.2}$	$93.60_{0.2}$	$93.59_{0.2}$	93.96	$90.26_{0.2}$	$90.22_{0.4}$	$91.09_{0.3}$	90.52
Adv	$94.61_{0.4}$	$92.19_{0.3}$	$94.36_{0.3}$	93.72	$94.67_{0.3}$	$93.74_{0.3}$	$93.63_{0.5}$	94.01	$90.16_{0.3}$	$90.25_{0.3}$	$90.78_{0.5}$	90.40
SCL	$93.78_{0.5}$	$91.67_{0.5}$	$93.91_{0.2}$	93.12	$94.25_{0.7}$	$93.26_{0.2}$	$93.37_{0.3}$	93.63	$89.41_{0.3}$	$89.33_{0.8}$	$89.76_{1.1}$	89.50
SCL+M=64	$94.55_{0.4}$	$92.32_{0.3}$	$\mathbf{94.42}_{0.3}$	93.76	$94.58_{0.3}$	$\mathbf{94.05}_{0.2}$	$93.85_{0.2}$	94.16	$\mathbf{90.74}_{0.4}$	$90.48_{0.3}$	$\mathbf{91.34}_{0.4}$	**90.85**
SCL+M=128	$94.69_{0.4}$	$\mathbf{92.42}_{0.3}$	$94.20_{0.2}$	**93.77**	$\mathbf{94.70}_{0.3}$	$93.85_{0.3}$	$\mathbf{94.02}_{0.2}$	**94.19**	$90.52_{0.4}$	$\mathbf{90.67}_{0.4}$	$91.21_{0.3}$	90.80
SCL+M=256	$\mathbf{94.70}_{0.5}$	$92.19_{0.6}$	$93.88_{0.2}$	93.59	$94.54_{0.4}$	$93.66_{0.3}$	$93.87_{0.3}$	94.02	$90.55_{0.3}$	$90.53_{0.2}$	$90.97_{0.4}$	90.68
Cross-language and cross-domain *Without unlabeled target data*												
Baseline	$93.76_{0.3}$	$90.78_{0.3}$	$93.95_{0.4}$	92.83	$93.67_{0.3}$	$93.25_{0.2}$	$92.98_{0.2}$	93.30	$89.49_{0.5}$	$89.16_{0.4}$	$89.73_{0.5}$	89.46
Intra	$94.23_{0.3}$	$91.55_{0.3}$	$93.78_{0.2}$	93.19	$93.9_{0.3}$	$93.39_{0.3}$	$93.16_{0.2}$	93.48	$89.75_{0.4}$	$89.90_{0.4}$	$90.66_{0.3}$	90.1
Adv	$94.13_{0.3}$	$91.51_{0.3}$	$93.64_{0.3}$	93.09	$93.31_{0.4}$	$93.44_{0.3}$	$93.15_{0.2}$	93.3	$89.7_{0.4}$	$89.72_{0.4}$	$90.74_{0.3}$	90.05
SCL	$93.83_{0.4}$	$90.98_{0.3}$	$93.86_{0.2}$	92.89	$93.86_{0.3}$	$93.59_{0.3}$	$93.16_{0.2}$	93.54	$89.5_{0.3}$	$89.34_{0.4}$	$89.5_{0.3}$	89.45
SCL+M=64	$94.05_{0.3}$	$91.26_{0.2}$	$93.83_{0.4}$	93.04	$94.17_{0.2}$	$93.94_{0.3}$	$93.67_{0.2}$	93.93	$89.85_{0.3}$	$89.83_{0.4}$	$90.45_{0.2}$	90.05
SCL+M=128	$\mathbf{94.46}_{0.4}$	$\mathbf{91.90}_{0.3}$	$93.97_{0.2}$	**93.41**	$\mathbf{94.24}_{0.5}$	$93.79_{0.3}$	$\mathbf{93.95}_{0.4}$	**93.99**	$89.83_{0.6}$	$\mathbf{90.27}_{0.4}$	$\mathbf{91.02}_{0.4}$	**90.37**
SCL+M=256	$94.23_{0.5}$	$91.13_{0.2}$	$\mathbf{94.07}_{0.3}$	93.14	$94.06_{0.3}$	$\mathbf{93.96}_{0.4}$	$93.82_{0.2}$	93.95	$\mathbf{90.15}_{0.4}$	$89.86_{0.3}$	$90.25_{0.2}$	90.07

Table 3: Experimental results for Multilingual Amazon benchmark. Experiments are conducted in both cross-language (CL) and cross-language cross-domain (CLCD) settings. The reported metric is average accuracy and each subscript indicates the standard deviation of five runs.

for cross-lingual transfer learning, leverages translated Amazon reviews as a parallel corpus.

MAN-MoE (Chen and Cardie, 2018): This model uses a multinomial adversarial network to extract language-invariant features for sentiment classification. It studies cross-lingual transfer with multiple source languages. Besides, it also leverages MoE to focus on more transferable source languages.

Baseline fine-tunes pretrained language models (DistilBERT/Roberta/XLM-R) on labeled data from source domains and directly tests on the target domain. **MMD**: Following Li et al. (2018), pairwise Maximum Mean Discrepancy (MMD) losses among three source domains are added to cross-entropy loss. **Intra** refers to center loss used in Wen et al. (2016) and Ye et al. (2020), which maximizes the agreement between each example and its class center. **Adv** refers to the widely studied domain adversarial neural network (Ganin et al., 2016), where a gradient reversal layer is used to reverse the gradients calculated by the domain classification task. **SCL** adopts supervised contrastive loss from Gunel et al. (2021). It refers to directly applying supervised contrastive learning with a small batch size. Our model that enhances supervised contrastive learning (SCL) with memory bank is denoted as **SCL+M**. We provide experimental results for $M \in \{64, 128, 256\}$.

5.4 Experimental Results

The experimental results of CD text classification are shown in Table 1. We compare our methods with previous SOTA methods on multi-source domain adaptation and strong baselines of RoBERTa variants. Our findings are as follows. Firstly, directly applying supervised contrastive loss has limited improvement over the baseline performance of directly fine-tuning the pretrained language model, and the performance of **SCL** does not exceed previous domain adaptation methods, such as Mixture-of-Experts (**MoE**) and intra-class loss (**Intra**). Secondly, increasing the number of contrasting examples M significantly improves performance compared to directly using SCL. Our proposed method **SCL+M=128** achieves the best performance among the compared methods, exceeding the Roberta baseline by 1.35% in cross-domain sentiment analysis and by 2.82% in cross-domain rumour detection. Finally, the domain generalization method **MMD** performs poorly in the CD setting. This may be because the data distribution of computer vision tasks is different from that of sentiment analysis.

The CL and CLCD experimental results are shown in Table 3. In the CL and CLCD settings, we do not include the baseline result for MMD, as we observe high variance validation loss during training, and sometimes training diverges. We also do not include MoE for CL experiments, as the encoder for CL experiments is significantly larger than CD experiments and the MoE structure exceeds our hardware limit. From Table 3, our XLM-R baseline does not exceed the one-to-one self-training based DA approach **CFd**, showing

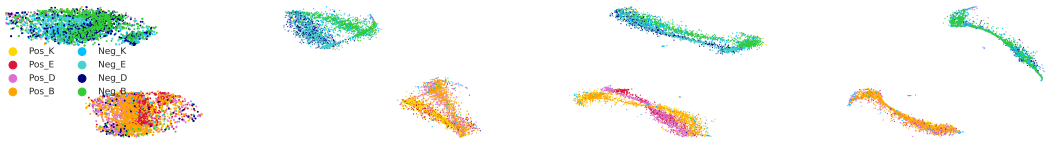

(a) RoBERTa (b) MoE (c) SCL (d) SCL+M=128

Figure 2: T-SNE visualization of the 256-dimensional sentence embeddings $\mathbf{z}$ for each model. The models are trained with B,K,E as source domains and D as the target domain. We sample 1,000 examples for each domain.

that knowledge of unlabeled target data is more important than increasing out-of-language training examples. In addition, in CL and CLCD settings, the performance of directly applying **SCL** is worse than intra-class loss (**Intra**) and adversarial training (**Adv**). We believe this is due to the small batch size (i.e., 2) during training of XLM-R. Hence, it is important to increase the contrasting examples for SCL in the small-batch setting. With increasing contrasting examples for SCL, we are able to achieve significant performance gain over our competitive baselines, and our best model **SCL+M=128** achieves state-of-the-art performance for both CL and CLCD settings. We observe performance drop when M is larger than 128. This is primarily due to the trade-off between the number of contrasting examples and their quality. Even though increasing the size of memory bank will benefit the lower bound of mutual information estimation, using an excessive number of prior examples will introduce noise for contrastive learning, since the text encoder has already been updated for many steps.

5.5 Analysis on Domain Divergence

To further analyze the performance of our model, we provide both intuitive visualization and quantitative analysis of domain discrepancy. Following Du et al. (2020) and Ben-David et al. (2010), we use $\mathcal{A}$-distance as the measurement for domain distance. To calculate $\mathcal{A}$-distance, we freeze the fine-tuned language model and the feed-forward layer f as encoder. Since the 256-dimensional $\mathbf{z}$ is used for the downstream classification task, we analyze domain discrepancy in this feature space. We sample two balanced sets of source examples and target examples with binary domain labels, i.e., source and target. Since we have multiple source domains, examples from each source domain will be down-sampled when calculating $\mathcal{A}$-distance, so that the total number of source examples and the number of target examples are balanced. This mixed dataset with binary domain labels will be split into two equal-size subsets, one for training and the other for testing. We then train a linear classifier with the first subset to distinguish source and target domain features. The error rate ϵ for this domain distinguishing classifier is calculated on the second subset, and we have the $\mathcal{A}$-distance as $d_{\mathcal{A}} = 2(1 - 2\epsilon)$.

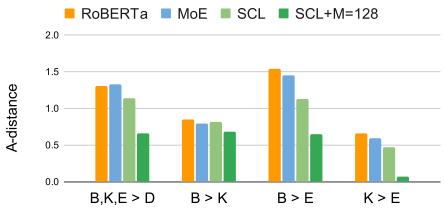

Figure 3: $\mathcal{A}$-distance of B,K,E to D generalization transfer.

We compare $d_{\mathcal{A}}$ of RoBERTa baseline, **MoE**, **SCL**, and **SCL+M=128** with B,K,E to D transfer, as well as $d_{\mathcal{A}}$ of source domain pairs. Results are shown in Figure 3. We see that the MoE model has little impact on reducing domain divergence for the backbone encoder. In contrast, the **SCL** and **SCL+M=128** models are able to reduce domain divergence and the latter achieves the lowest domain divergence compared to all other baselines.

To intuitively understand how our models overcome domain discrepancy, we also plot the t-SNE (Van der Maaten and Hinton, 2008) visualization of the features from different domains, as shown in Figure 2. For the RoBERTa baseline (Fig. 2a), we observe clear domain discrepancy within each sentiment cluster, and the sentiment cluster is relatively dispersed. Similarly, for the MoE model (Fig. 2b), we also observe domain discrepancy in the sentiment clusters, showing that the MoE objective does not improve the encoders' capability for producing domain-invariant representation. For the SCL baseline (Fig. 2c), we observe that the sentiment clusters are more concentrated and domain discrepancy is significantly reduced, but there is still some heterogeneity within the positive and negative review clusters. From Fig. 2d, we observe that increasing the contrasting features is able to further reduce domain divergence, which

is in line with the quantitative analysis of Fig. 3. We believe this is because **SCL+M** is trained with a large number of contrasting examples. By minimizing the intra-cluster distance in each batch, domain divergence within each sentiment cluster is reduced.

6 Conclusions and Future Work

In this paper, we study the under-explored domain generalization problem for text classification. We show that for cross-domain text classification, generalization performance from multiple source domains can exceed the best performance of one-to-one domain adaptation, even if the target domain is unknown during training. To this end, domain generalization is more practical and easier to deploy in realistic scenarios. To further improve the performance of cross-domain text classification, we propose an effective and memory-saving approach based on supervised contrastive learning for the domain generalization problem. We conduct extensive experiments in CD, CL, and CLCD settings. Experimental results have shown that our framework consistently outperforms strong baselines and the previous state of the art in all three experimental settings.

7 Acknowledgements

We would like to thank the reviewers for their insightful comments.

References

Michael Arbel, Anna Korba, Adil Salim, and Arthur Gretton. 2019. Maximum mean discrepancy gradient flow. In *Proceedings of NeurIPS*.

Eyal Ben-David, Nadav Oved, and Roi Reichart. 2022. PADA: Example-based Prompt Learning for on-the-fly Adaptation to Unseen Domains. *TACL*.

Shai Ben-David, John Blitzer, Koby Crammer, Alex Kulesza, Fernando Pereira, and Jennifer Wortman Vaughan. 2010. A theory of learning from different domains. *Machine Learning*.

John Blitzer, Mark Dredze, and Fernando Pereira. 2007. Biographies, Bollywood, boom-boxes and blenders: Domain adaptation for sentiment classification. In *Proceedings of ACL*.

Ting Chen, Simon Kornblith, Mohammad Norouzi, and Geoffrey Hinton. 2020. A simple framework for contrastive learning of visual representations. In *Proceedings of NeurIPS*.

Xilun Chen and Claire Cardie. 2018. Multinomial adversarial networks for multi-domain text classification. In *Proceedings of NAACL*.

Xilun Chen, Yu Sun, Ben Athiwaratkun, Claire Cardie, and Kilian Weinberger. 2018. Adversarial deep averaging networks for cross-lingual sentiment classification. *TACL*.

Alexis Conneau, Kartikay Khandelwal, Naman Goyal, Vishrav Chaudhary, Guillaume Wenzek, Francisco Guzmán, Edouard Grave, Myle Ott, Luke Zettlemoyer, and Veselin Stoyanov. 2020. Unsupervised cross-lingual representation learning at scale. In *Proceedings of ACL*.

X. Ding, Q. Shi, B. Cai, T. Liu, Y. Zhao, and Q. Ye. 2019. Learning multi-domain adversarial neural networks for text classification. *IEEE Access*.

Chunning Du, Haifeng Sun, Jingyu Wang, Qi Qi, and Jianxin Liao. 2020. Adversarial and domain-aware BERT for cross-domain sentiment analysis. In *Proceedings of ACL*.

Yaroslav Ganin, Evgeniya Ustinova, Hana Ajakan, Pascal Germain, Hugo Larochelle, François Laviolette, Mario Marchand, and Victor Lempitsky. 2016. Domain-adversarial training of neural networks. *JMLR*.

Florian Graf, Christoph Hofer, Marc Niethammer, and Roland Kwitt. 2021. Dissecting supervised constrastive learning. In *Proceedings of ICML*.

Jean-Bastien Grill, Florian Strub, Florent Altché, Corentin Tallec, Pierre Richemond, Elena Buchatskaya, Carl Doersch, Bernardo Avila Pires, Zhaohan Guo, Mohammad Gheshlaghi Azar, et al. 2020. Bootstrap your own latent-a new approach to self-supervised learning. In *Proceedings of NeurIPS*.

Beliz Gunel, Jingfei Du, Alexis Conneau, and Veselin Stoyanov. 2021. Supervised contrastive learning for pre-trained language model fine-tuning. In *Proceedings of ICLR*.

Han Guo, Ramakanth Pasunuru, and Mohit Bansal. 2020. Multi-source domain adaptation for text classification via distancenet-bandits. In *Proceedings of AAAI*.

Jiang Guo, Darsh Shah, and Regina Barzilay. 2018. Multi-source domain adaptation with mixture of experts. In *Proceedings of EMNLP*.

Jing Jiang and ChengXiang Zhai. 2007. Instance weighting for domain adaptation in NLP. In *Proceedings of ACL*.

Constantinos Karouzos, Georgios Paraskevopoulos, and Alexandros Potamianos. 2021. UDALM: Unsupervised domain adaptation through language modeling. In *Proceedings of NAACL*.

Prannay Khosla, Piotr Teterwak, Chen Wang, Aaron Sarna, Yonglong Tian, Phillip Isola, Aaron Maschinot, Ce Liu, and Dilip Krishnan. 2020. Supervised contrastive learning. In *Proceedings of NeurIPS*.

Haoliang Li, Sinno Jialin Pan, Shiqi Wang, and Alex C. Kot. 2018. Domain generalization with adversarial feature learning. In *Proceedings of CVPR*.

Juntao Li, Ruidan He, Hai Ye, Hwee Tou Ng, Lidong Bing, and Rui Yan. 2020. Unsupervised domain adaptation of a pretrained cross-lingual language model. In *Proceedings of IJCAI*.

Linlin Liu, Bosheng Ding, Lidong Bing, Shafiq Joty, Luo Si, and Chunyan Miao. 2021. MulDA: A multilingual data augmentation framework for low-resource cross-lingual NER. In *Proceedings of ACL*.

Qi Liu, Yue Zhang, and Jiangming Liu. 2018. Learning domain representation for multi-domain sentiment classification. In *Proceedings of NAACL*.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Roberta: A robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692*.

Sinno Jialin Pan, Xiaochuan Ni, Jian-Tao Sun, Qiang Yang, and Zheng Chen. 2010. Cross-domain sentiment classification via spectral feature alignment. In *Proceedings of WWW*.

Peter Prettenhofer and Benno Stein. 2010. Cross-language text classification using structural correspondence learning. In *Proceedings of ACL*.

Victor Sanh, Lysandre Debut, Julien Chaumond, and Thomas Wolf. 2019. Distilbert, a distilled version of bert: smaller, faster, cheaper and lighter. *arXiv preprint arXiv:1910.01108*.

Laurens Van der Maaten and Geoffrey Hinton. 2008. Visualizing data using t-SNE. *JMLR*.

Bailin Wang, Mirella Lapata, and Ivan Titov. 2021a. Meta-learning for domain generalization in semantic parsing. In *Proceedings of NAACL*.

Jindong Wang, Cuiling Lan, Chang Liu, Yidong Ouyang, Wenjun Zeng, and Tao Qin. 2021b. Generalizing to unseen domains: A survey on domain generalization.

Yandong Wen, Kaipeng Zhang, Zhifeng Li, and Yu Qiao. 2016. A discriminative feature learning approach for deep face recognition. In *Proceedings of ECCV*.

Dustin Wright and Isabelle Augenstein. 2020. Transformer based multi-source domain adaptation. In *Proceedings of EMNLP*.

Fangzhao Wu and Yongfeng Huang. 2016. Sentiment domain adaptation with multiple sources. In *Proceedings of ACL*.

Ruochen Xu and Yiming Yang. 2017. Cross-lingual distillation for text classification. In *Proceedings of ACL*.

Hai Ye, Qingyu Tan, Ruidan He, Juntao Li, Hwee Tou Ng, and Lidong Bing. 2020. Feature adaptation of pre-trained language models across languages and domains with robust self-training. In *Proceedings of EMNLP*.

Han Zhao, Shanghang Zhang, Guanhang Wu, José MF Moura, Joao P Costeira, and Geoffrey J Gordon. 2018. Adversarial multiple source domain adaptation. In *Proceedings of NeurIPS*.

A Further Ablation Studies

	N=60	N=150	N=300	Full Data
RoBERTa	64.05 ± 9.83	76.10 ± 12.34	86.95 ± 4.16	93.70 ± 0.37
SCL	63.85 ± 8.33	76.40 ± 6.59	85.65 ± 4.47	94.25 ± 0.40
SCL+M=128	**78.75** ± 4.91	**83.65** ± 4.26	**90.95** ± 4.13	**95.10** ± 0.25

Table 4: Ablation study of few-shot cross-domain sentiment classification for B,K,D to E transfer.

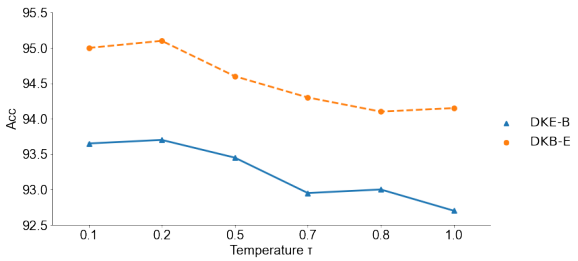

Figure 4: Ablation study of the effect of temperature τ. The reported accuracy is the average of 5 runs.

SCL in Few-Shot Setting. We conduct a few-shot learning experiment with B,K,D to E transfer. As shown in Table 4, SCL+M=128 achieves the most performance gain when the number of training examples is limited to 60, while conventional cross-entropy fails to converge. We believe that in the case of few-shot setting, the memory bank resembles a way for data augmentation, as the previously encoded sentences are reused as contrasting examples.

Effect of Temperature τ. We also provide hyper-parameter analysis for the scaling factor τ. We found that the performance of our model is negatively correlated to τ, and $\tau = 0.2$ works well empirically. We also found that when τ is small, the scale of supervised contrastive loss and cross-entropy validation loss is similar, leading to coherent optimization.

B Hyper-Parameters for the Baselines

We mainly conduct hyper-parameter tuning on the validation set of Multi-domain Sentiment Dataset. For model training, we tune the learning rates in {5e-6, 1e-5, 2e-5, 3e-5}, and batch size in {8, 16, 32}. The final learning rate is 1e-5 and scheduled linearly with training steps. We train our models for 10 epochs for cross-domain (CD), cross-language (CL), and cross-language cross-domain (CLCD) experiments. The batch size for CD experiments is 16. In CL experiments, the batch size is 2 with gradient accumulation step of 8 due to GPU memory constraint. For the experiment for CD sentiment analysis, the combined training size is 4,800. For the baseline experiments, we follow the methodology of Ye et al. (2020), that is adding a λ-weighted loss term to the cross-entropy loss. We show our choices for balancing parameters for losses as follows:

- **Intra-Class Loss** The weight for λ is tuned in {1, 0.5, 0.2, 0.1, 0.05}. We set λ to be 0.2 for CD experiments and 0.1 for CL experiments.
- **MoE** (Wright and Augenstein, 2020) We follow the MoE-Avg method in the original paper. For Roberta-large MoE models, we use 4 Nvidia V100 GPUs for training, as this model requires multiple encoders during training.
- **Adversarial Loss** Following the practice of prior works (Wright and Augenstein, 2020; Guo et al., 2018), the weight for adversarial training λ is 0.003.
- **MMD** We implemented a pair-wise MMD domain generalization approach with RBF kernel, following the practice of Li et al. (2018). The weight of MMD loss is tuned in {1.0, 0.5, 0.2, 0.1, 0.05} and set to 0.2.

A Zero-Shot Claim Detection Framework using Question Answering

Revanth Gangi Reddy[1], **Sai Chetan**[1], **Yi R. Fung**[1], **Kevin Small**[2], **Heng Ji**[1]
[1]University of Illinois at Urbana-Champaign [2]Amazon Alexa AI
{revanth3, scc8, yifung2, hengji}@illinois.edu
smakevin@amazon.com

Abstract

In recent years, there has been an increasing interest in claim detection as an important building block for misinformation detection. This involves detecting more fine-grained attributes relating to the claim, such as the claimer, claim topic, claim object pertaining to the topic, etc. Yet, a notable bottleneck of existing claim detection approaches is their portability to emerging events and low-resource training data settings. In this regard, we propose a fine-grained claim detection framework that leverages zero-shot Question Answering (QA) using directed questions to solve a diverse set of sub-tasks such as topic filtering, claim object detection, and claimer detection. We show that our approach[1] significantly outperforms various zero-shot, few-shot and task-specific baselines on the NEWSCLAIMS benchmark (Reddy et al., 2021).

1 Introduction

Claim detection over news involves identifying claims related to various topics in news articles. Identifying such claims is a crucial first step for fighting misinformation and disinformation online. However, such harmful content can evolve rapidly, triggered by relatively new events which can gain extensive media coverage within a short time span. Hence, claim detection in such scenarios requires systems that are able to adapt quickly, by working well under zero-shot or few-shot settings.

Towards this goal, Reddy et al. (2021) propose a new benchmark, NEWSCLAIMS, that evaluates claim detection for previously unseen topics without access to any training data. Given a collection of news articles, the task involves identifying claims that are related to a pre-defined set of topics, along with extracting attributes for each claim, such as claim span, claim object, claimer and stance. Essentially, this benchmark extends the claim detection task to involve extracting additional background attributes relating to the claim, such as the claim object and claimer. Furthermore, the claimer detection sub-task within NEWSCLAIMS requires considerable document-level reasoning, making it harder than existing attribution tasks (Pareti, 2016; Newell et al., 2018), which mainly involve sentence-level reasoning. Reddy et al. (2021) propose various baselines for each sub-task in the NEWSCLAIMS benchmark, which involve zero-shot and few-shot approaches in addition to baselines trained using task-specific data.

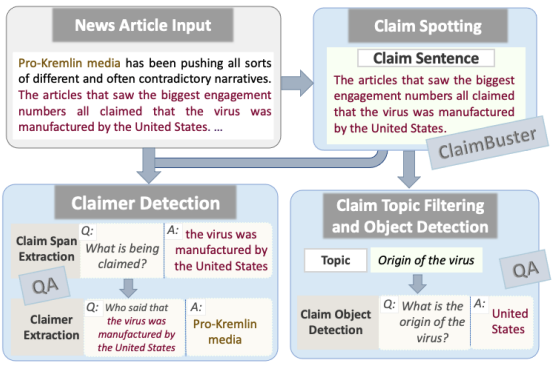

Figure 1: An overview of our claim detection framework. We leverage zero-shot QA for claim topic filtering, claim object detection and claimer detection for the claims spotted by ClaimBuster (Hassan et al., 2017).

To handle this scenario in a low-resource setting, we hypothesize that identifying claim topics and extracting corresponding claim attributes can be formulated as a question answering task. Hence, we propose to leverage a Question Answering (QA) system for the claim detection task and show that the *same QA model* can be used to solve multiple sub-tasks within claim detection, without the need of any task-specific training data. This involves filtering claims relating to specific topics, identifying claim objects associated with such topics and attribution for identifying the claimers making these

[1]Code is available here: https://github.com/blender-nlp/NewsClaims/tree/main/zero-shot-qa

Proceedings of the 29th International Conference on Computational Linguistics, pages 6927–6933
October 12–17, 2022.

claims. We realise this by using directed questions to help solve connected sub-tasks such as topic filtering, claim object detection and claimer detection within the claim detection framework. An overview of this framework is shown in Figure 1.

Leveraging pre-trained language models for directly solving end-tasks has been explored in prompting (Liu et al., 2021), with promising performance in both zero-shot (Zhong et al., 2021) and few-shot (Brown et al., 2020) settings. Prior work uses prompts that are in the form of prefix (Li and Liang, 2021) or cloze-style tasks (Schick and Schütze, 2021). In this work, we solve end-tasks by formulating them as directed questions, instead of prompts, that are fed into a QA model (see Figure 1 1 for examples of directed questions). We will first briefly introduce the NEWSCLAIMS benchmark (Section 2) and then describe our zero-shot claim detection framework (Section 3).

Our main contributions are: (1) we propose to use a single pre-trained question answering system in a zero-shot setting for various sub-tasks in claim detection, such as topic-filtering, claim object detection and claimer detection; (2) we show that, using directed questions, a QA model is able to outperform other attribution methods for claimer detection, which requires document-level reasoning; and (3) our proposed claim detection framework achieves state-of-the-art performance on multiple sub-tasks in the NEWSCLAIMS benchmark, outperforming various zero-shot, few-shot and task-specific baselines.

2 NEWSCLAIMS Background

NEWSCLAIMS (Reddy et al., 2021) extends claim detection to extract additional background attributes relating to the claim, such as claim objects and claimers. NEWSCLAIMS evaluates claim detection in the context of an emerging real-world scenario, by considering claims relating to various aspects of COVID-19. Specifically, the topics involved in the benchmark are about the *origin* of the virus, *transmission* of the virus, *cure* for the virus and *protection* from the virus. We refer the reader to Reddy et al. (2021) for detailed definitions and sample claims. Below, we briefly describe the different sub-tasks that we consider within this benchmark and their corresponding baselines, that were introduced in Reddy et al. (2021).

Claim Sentence Detection with Topic-Filtering: This sub-task involves identifying sentences that contain claims relating to COVID-19. For this, Reddy et al. (2021) begin with ClaimBuster (Hassan et al., 2017), which has been trained to identify check-worthy claims (Arslan et al., 2020). In order to then select claims relating to specific topics, from those extracted by ClaimBuster, Reddy et al. (2021) use pre-trained NLI models for zero-shot topic-filtering (Yin et al., 2019). This is done by posing the claim sentence as the premise and constructing a hypothesis for each candidate topic.

Claim Object Detection: A *claim object* relates to what is being claimed in the claim sentence with respect to the topic. Reddy et al. (2021) use zero-shot and few-shot approaches for this sub-task, via zero-shot prompting and leveraging few-shot examples for in-context learning (Brown et al., 2020) and prompt-based fine-tuning (Gao et al., 2021). The prompts are hand-crafted using the topic of the claim sentence.

Claimer Detection: Claimer detection involves identifying the source of the claims made within news articles. These claims can be categorized as either *reported* or those that are made by the author of the news article, i.e., the *journalist*, themselves. Reported claims could originate from people, organizations or other sources in news. The claimer detection sub-task within NEWSCLAIMS involves identifying whether the claim was made by the journalist or who the claimer is, in case it is reported. Since the sub-task requires attribution, Reddy et al. (2021) consider baselines that leverage semantic role labeling (SRL) or are trained on existing attribution datasets (Newell et al., 2018).

3 Method

We first describe the QA model which we use as the pre-trained model to feed directed questions as input. Next, we outline our zero-shot claim detection framework which leverages the above model for sub-tasks such as topic filtering, claim object detection and claimer detection.

3.1 Question Answering Model

The model is a transformer-based extractive question answering system that takes the question and context as input. The QA model has an extractive answer span predictor that predicts the answer spans using the output representations $\mathbf{H}$ from pre-trained language models (LM) (Devlin et al., 2019).

Specifically, the model has a predictor for the beginning α and ending β of the answer span as follows:

$$\boldsymbol{\alpha} = \text{softmax}(\mathbf{W}_1\mathbf{H}) \quad (1)$$

$$\boldsymbol{\beta} = \text{softmax}(\mathbf{W}_2\mathbf{H}) \quad (2)$$

where $\mathbf{W}_1$, $\mathbf{W}_2 \in R^{1\times D}$, D is the dimension of language model's output $\mathbf{H} \in R^{T\times D}$ and T is length of input context. Given $\mathbf{b}$ and $\mathbf{e}$ as the one-hot vectors for the ground-truth start and end positions, the loss function during training is the averaged cross entropy on the two span predictors:

$$\mathcal{L} = -\frac{1}{2}\sum_{t=1}^{T}\{1(\mathbf{b}_t)\log\boldsymbol{\alpha}_t + 1(\mathbf{e}_t)\log\boldsymbol{\beta}_t\} \quad (3)$$

At inference, the answer score for a span (i, j) within the context is computed as $S(i, j) = \alpha_i + \beta_j$, with the highest scoring span taken as the final answer.

3.2 Claim Detection Framework

Given a news article as input, our claim detection framework outputs claims relating to specific topics, along with their corresponding claim objects and claimers. Following Reddy et al. (2021), we use ClaimBuster (Hassan et al., 2017) as the claim-spotting model to first identify sentences that contain claims. Next, we leverage the QA model described in Section 3.1 for topic-filtering, claim object detection and claimer detection. An outline of our claim detection framework can be seen in Figure 1, with each step described in detail below.

Claim Topic Filtering: We propose to do topic filtering by measuring topic relevance as the answer confidence from a QA model, when a question corresponding to the topic is passed as input. We achieve this by formulating a question for each topic, with the claim sentence as context, as shown in Figure 2. The answer score for each question is taken as the corresponding topic relevance. Claims are then filtered based on the highest topic score using a threshold. In comparison, NLI does filtering based on the corresponding highest entailment score. The motivation behind using QA for this sub-task is for the directed questions to be more relevant towards identifying these topics, compared to the implicit inference in NLI.

Claim Object Detection: We pose the *claim object* detection sub-task as an extractive QA task using the same directed questions shown in Figure 2. While the answer score for a question is used for topic filtering, the corresponding answer span is used as the claim object.

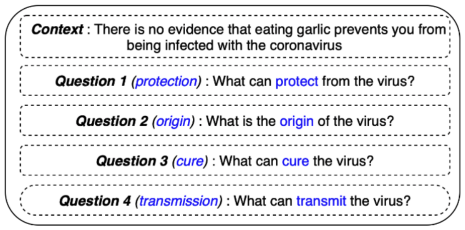

Figure 2: Questions corresponding to individual topics, with the claim sentence as context.

Claimer Detection: We formulate the claimer detection sub-task as a two-step process: first detect the exact claim span within the claim sentence and then identify the claimer. We leverage QA for both steps as follows. The claim span is obtained from the QA model's answer by using "*What is being claimed?*" as the question and the claim sentence as context. Next, for claimer identification, we use the entire news article as context, with the previously extracted claim span inserted into the question, "*Who said that <claim span>?*". We threshold on the answer score to determine if no claimer was identified, in which case, the claim sentence is attributed to the journalist.

4 Experiments

4.1 Setup

The QA model uses *bert-large-uncased* as the underlying language model. It is trained on SQuAD 2.0 (Rajpurkar et al., 2018) for four epochs with a learning rate of 3e-5 and Natural Questions (Kwiatkowski et al., 2019) for one epoch with a learning rate of 1e-5. Batch size is 16 in both cases.

We evaluate our approach on the NEWSCLAIMS benchmark[2] (Reddy et al., 2021). We refer the reader to Reddy et al. (2021) for a detailed description of each of the baselines. The development and test splits comprise 18 news articles with 103 claims and 125 news articles with 786 claims respectively. The thresholds for claim topic filtering and claimer detection were tuned on the development set. All numbers are reported on the test set.

4.2 Results and Analysis

In this section, we evaluate our proposed claim-detection framework for individual sub-tasks such

[2]https://github.com/uiucnlp/NewsClaims

as detecting claims relating to specific topics about COVID-19 (Section 4.2.1), extracting the claim object pertaining to the claim topic (Section 4.2.2) and identifying the claimer (Section 4.2.3).

4.2.1 Claim Sentence Detection

Here, we measure the performance for both zero-shot topic classification and the subsequent filtering. We first evaluate the performance of the QA system for classifying topics, given the claim sentence. Table 1 compares the performance of the NLI and QA systems for zero-shot classification over the four COVID-19 topics. We can see that our zero-shot QA approach considerably outperforms zero-shot NLI, demonstrating that QA can be better at measuring topic relevance. Further, we see that QA is able to overcome the NLI model's inability to distinguish between similar topics such as (*protection* and *cure*) or (*origin* and *transmission*). Some representative examples are in Table 2 with more detailed confusion matrices present in Section A.1 in the Appendix.

Model	Or.	Trans.	Prot.	Cure	All
NLI	56.9	45.1	54.5	3.3	46.6
QA	**85.9**	**64.7**	**63.9**	**66.5**	**72.3**

Table 1: Topic-wise F1 and overall accuracy (both in %) for topic classification given the claim sentence.

Claim Sentence	Topic
This novel coronavirus was believed to have started in a large seafood or wet market, suggesting animal-to-person spread.	**Gold:** Origin **NLI:** Trans. **QA:** Origin
One medication, an antiviral drug called Remdesivir, has been shown in certain studies to improve symptoms and shorten hospital stays.	**Gold:** Cure **NLI:** Protection **QA:** Cure

Table 2: Some examples of incorrect topic predictions from the NLI model which the QA model overcomes. We see that QA, which uses directed questions, is better at being able to distinguish between similar topics such as (*origin* and *transmission*) or (*protection* and *cure*), compared to NLI, which uses implicit inference.

Next, we measure the claim sentence detection performance to evaluate the QA model for topic filtering on the claims outputs by ClaimBuster. Table 3 compares our QA-based topic filtering approach against a pre-trained NLI model, as used in Reddy et al. (2021). We can see that using QA provides up to 5 point improvement in F1, suggesting that the answer confidence from the QA model can be a better estimate for filtering claims relating to these topics, compared to entailment score.

Model	P	R	F1
ClaimBuster	13.0	**86.5**	22.6
ClaimBuster + NLI	21.8	53.3	30.9
ClaimBuster + QA	**30.7**	43.4	**36.0**

Table 3: Performance (in %) of various systems for detecting sentences with claims relating to COVID-19.

4.2.2 Claim Object Detection

For the claim object detection sub-task, we compare our QA approach with various zero-shot and few-shot approaches used in Reddy et al. (2021). Table 4 shows the performance of the QA system along with different prompt-based approaches, that leverage generative language models to output the claim object. While GPT-3 and T5 show competitive performance in few-shot settings, our zero-shot QA approach outperforms by more than 5 points.

Approach	Model	Type	F1
Prompting	GPT-3	Zero-shot	15.2
Prompting	T5	Zero-shot	11.4
In-context learning	GPT-3	Few-Shot	51.9
Prompt-based fine-tuning	T5	Few-Shot	51.6
QA	BERT	Zero-shot	**57.0**

Table 4: F1 score (in %) of different zero-shot and few-shot approaches for the claim object detection sub-task.

4.2.3 Claimer Detection

The claimer detection sub-task is evaluated based on the classification F1 for predicting which claims are from journalists, along with a string-match F1 (Rajpurkar et al., 2018) for extracting the mention of the claimer in case of reported claims. Table 5 compares our zero-shot QA-based approach for claimer detection with the Semantic Role Labeling (SRL) and PolNeAR news-attribution (Newell et al., 2018) baselines from Reddy et al. (2021).

Model	Overall F1	Reported	Journalist
SRL	41.7	23.5	**67.2**
PolNeAR	42.3	25.5	65.9
QA	**50.1**	**39.8**	64.4

Table 5: F1 (in %) for identifying the claimer. Numbers for reported and journalist are shown separately.

To understand why sentence-level approaches, such as SRL and PolNeAR, can be very competitive at identifying claims that are from the journalist, we manually analyzed some examples. We observed that claims from the journalist are usually made in a first-person point of view, which can be identified

by sentence-level reasoning. Table 6 shows some examples for claims that directly come from the journalist and those that are reported from other sources. It can be seen that those that come from the journalist do not involve cue words and are usually presented in a first-person point of view. This explains the competitive performance of SRL and PolNeAR for predicting which claims come from the journalist.

Claim Sentence	Type
It is not yet known if remdesivir is safe for the treatment of COVID-19.	Journalist
Inhaling bleach fumes is dangerous and will not kill viruses that are already inside.	Journalist
An earlier version of this article claimed a laboratory near Wuhan could be to blame for the outbreak of coronavirus.	Reported
The South China Agricultural University in Guangzhou says that two of its researchers have identified the pangolin as the potential source of nCoV-2019.	Reported

Table 6: Some examples for when the claimer is journalist vs when it is a reported claim.

However, promising results in using QA for claimer detection can be seen for reported claims, which can require document-level reasoning skills for identifying the claimer. Table 7 breaks down the performance for reported claims based on where the claimer mention is present. We can see that QA outperforms other attribution approaches for both cases, with even larger gains for when the claimer is present outside the claim sentence (which necessitates cross-sentence attribution).

Model	In-sentence	Out-of-sentence
SRL	35.8	2.4
PolNeAR	38.9	2.7
QA	**46.2**	**29.0**

Table 7: F1 (in %) for claimer detection for when it is present within or outside the claim sentence.

4.2.4 Analysis of Question Templates

Instead of hand-crafting questions from topics, we experiment with using each topic directly as a question to be fed as input to the QA model. Table 8 shows the performance for the claim sentence detection, topic classification and claim object detection sub-tasks for the settings where a directed question is manually constructed from the topic, compared to where the topic is used as the question. We can see while claim sentence detection and topic classification performance are almost similar, the performance is considerably better for claim object detection in case of directed (hand-crafted) questions. This implies that the answer confidence from a weakly-defined question (by just using topic as the question) is still a reliable measure of topic relevance. However, directed questions are useful for getting the right answer spans, which is crucial in case of claim object detection. Note that hand-crafting the question does not need considerable effort, as it mainly involves converting the topic into an information-seeking format by prepending with a "what", for e.g.: "*protection from the virus*" –> "*What can protect from the virus?*", "*transmission of the virus*" –> "*What can transmit the virus?*"

Sub-Task	Hand-crafted	Topic
Claim Sentence Detection	36.0	35.8
Topic Classification	72.3	73.2
Claim Object Detection	57.0	47.0

Table 8: Comparison of performance (in %) for claim sentence detection (F1), topic classification (Acc.) and claim object detection (F1) sub-tasks when using questions that are *hand-crafted* from the topic vs using the *topic* directly as the question.

5 Conclusions and Future Work

We propose a new claim detection framework that leverages zero-shot QA with directed questions for various sub-tasks such as topic filtering, claim object detection and claimer detection. We show that these questions can be adept at identifying topic relevance for claims related to COVID-19. We demonstrate that QA can be leveraged for claimer detection with document-level attribution, while considerably outperforming attribution systems that can be limited by sentence-level reasoning. Future work involves building a unified model that can extract claims and corresponding attributes together, without the need for separate components for each individual attribute.

Acknowledgement

This research is based upon work supported by U.S. DARPA AIDA Program No. FA8750-18-2-0014. FA8750-19-2-1004. The views and conclusions contained herein are those of the authors and should not be interpreted as necessarily representing the official policies, either expressed or implied, of DARPA, or the U.S. Government. The U.S. Government is authorized to reproduce and distribute reprints for governmental purposes notwithstanding any copyright annotation therein.

References

Fatma Arslan, Naeemul Hassan, Chengkai Li, and Mark Tremayne. 2020. A Benchmark Dataset of Checkworthy Factual Claims. In *14th International AAAI Conference on Web and Social Media*. AAAI.

Tom B Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. 2020. Language models are few-shot learners. *arXiv preprint arXiv:2005.14165*.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, Minneapolis, Minnesota. Association for Computational Linguistics.

Tianyu Gao, Adam Fisch, and Danqi Chen. 2021. Making pre-trained language models better few-shot learners. In *Association for Computational Linguistics (ACL)*.

Naeemul Hassan, Gensheng Zhang, Fatma Arslan, Josue Caraballo, Damian Jimenez, Siddhant Gawsane, Shohedul Hasan, Minumol Joseph, Aaditya Kulkarni, Anil Kumar Nayak, et al. 2017. Claimbuster: The first-ever end-to-end fact-checking system. *Proceedings of the VLDB Endowment*, 10(12):1945–1948.

Tom Kwiatkowski, Jennimaria Palomaki, Olivia Redfield, Michael Collins, Ankur Parikh, Chris Alberti, Danielle Epstein, Illia Polosukhin, Jacob Devlin, Kenton Lee, et al. 2019. Natural questions: a benchmark for question answering research. *Transactions of the Association for Computational Linguistics*, 7:453–466.

Xiang Lisa Li and Percy Liang. 2021. Prefix-tuning: Optimizing continuous prompts for generation. *arXiv preprint arXiv:2101.00190*.

Pengfei Liu, Weizhe Yuan, Jinlan Fu, Zhengbao Jiang, Hiroaki Hayashi, and Graham Neubig. 2021. Pre-train, prompt, and predict: A systematic survey of prompting methods in natural language processing. *arXiv preprint arXiv:2107.13586*.

Edward Newell, Drew Margolin, and Derek Ruths. 2018. An attribution relations corpus for political news. In *Proceedings of the Eleventh International Conference on Language Resources and Evaluation (LREC 2018)*.

Silvia Pareti. 2016. PARC 3.0: A corpus of attribution relations. In *Proceedings of the Tenth International Conference on Language Resources and Evaluation (LREC'16)*, pages 3914–3920, Portorož, Slovenia. European Language Resources Association (ELRA).

Pranav Rajpurkar, Robin Jia, and Percy Liang. 2018. Know what you don't know: Unanswerable questions for squad. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 784–789.

Pranav Rajpurkar, Jian Zhang, Konstantin Lopyrev, and Percy Liang. 2016. Squad: 100,000+ questions for machine comprehension of text. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 2383–2392.

Revanth Gangi Reddy, Sai Chinthakindi, Zhenhailong Wang, Yi R Fung, Kathryn S Conger, Ahmed S Elsayed, Martha Palmer, and Heng Ji. 2021. Newsclaims: A new benchmark for claim detection from news with background knowledge. *arXiv preprint arXiv:2112.08544*.

Timo Schick and Hinrich Schütze. 2021. Exploiting cloze-questions for few-shot text classification and natural language inference. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*, pages 255–269.

Wenpeng Yin, Jamaal Hay, and Dan Roth. 2019. Benchmarking zero-shot text classification: Datasets, evaluation and entailment approach. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 3914–3923.

Ruiqi Zhong, Kristy Lee, Zheng Zhang, and Dan Klein. 2021. Adapting language models for zero-shot learning by meta-tuning on dataset and prompt collections. In *Findings of the Association for Computational Linguistics: EMNLP 2021*, pages 2856–2878.

A Appendix

A.1 Topic Classification performance

Figures 3 and 4 show the topic classification confusion matrices for the NLI and QA models respectively. As Reddy et al. (2021) point out, the NLI model predominantly suffers from classifying claims related to *cure* as *protection* and those related to *origin* as *transmission*. However, our QA model is able to overcome this, which explains the improved performance in topic classification and topic filtering for claim sentence detection.

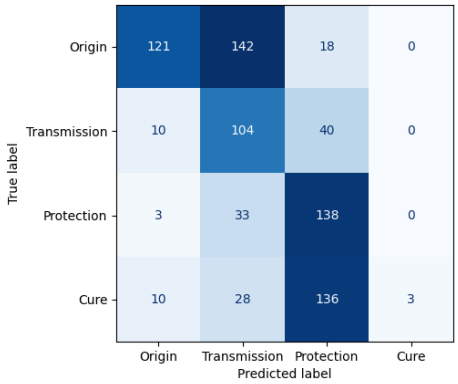

Figure 3: Topic classification confusion matrix for the the NLI model.

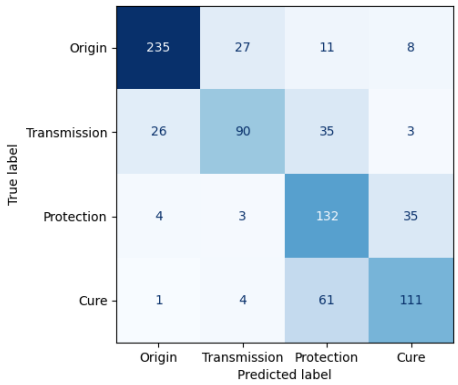

Figure 4: Topic classification confusion matrix for the the QA model.

A.2 QA Training Datasets

We give a brief overview of the datasets used to train the extractive QA model.

SQuAD: SQuAD1.1 (Rajpurkar et al., 2016) is an extractive machine reaching comprehension dataset containing questions posed by crowdworkers on a set of wikipedia articles. SQuAD2.0 (Rajpurkar et al., 2018) combines the 100,000+ questions in SQuAD1.1 with over 50,000 unanswerable questions written adversarially by crowdworkers to look similar to answerable ones.

Natural Questions: NQ (Kwiatkowski et al., 2019) is an english machine reading comprehension benchmark which contains 300,000+ questions from Google users, and requires systems to read and comprehend entire Wikipedia articles to answer them.

Asymmetric Mutual Learning for Multi-source Unsupervised Sentiment Adaptation with Dynamic Feature Network

Rui Li*, **Cheng Liu, Dazhi Jiang**
Department of Computer Science, Shantou University
{ruili, cliu, dzjiang}@stu.edu.cn

Abstract

Recently, fine-tuning the pre-trained language model (PrLM) on labeled sentiment datasets demonstrates impressive performance. However, collecting labeled sentiment dataset is time-consuming, and fine-tuning the whole PrLM brings about much computation cost. To this end, we focus on multi-source unsupervised sentiment adaptation problem with the pre-trained features, which is more practical and challenging. We first design a dynamic feature network to fully exploit the extracted pre-trained features for efficient domain adaptation. Meanwhile, with the difference of the traditional source-target domain alignment methods, we propose a novel asymmetric mutual learning strategy, which can robustly estimate the pseudo-labels of the target domain with the knowledge from all the other source models. Experiments on multiple sentiment benchmarks show that our method outperforms the recent state-of-the-art approaches, and we also conduct extensive ablation studies to verify the effectiveness of each the proposed module.

1 Introduction

Sentiment classification (Cambria et al., 2020) aims to predict the sentiment label for each textual data automatically (Susanto et al., 2022), which is one of the most popular natural language processing (NLP) tasks with many important applications, such as social media monitoring (Ortigosa et al., 2014), market research (Jabbar et al., 2019), conversation sentiment detection (Tu et al., 2022), *etc*. Very recently, the pre-trained language models (PrLMs), *e.g.*, BERT (Devlin et al., 2019), have demonstrated significant improvements on wide-range of NLP tasks, including the sentiment classification. This framework includes two steps: the transformer-based (Vaswani et al., 2017) model is first pre-trained on large unlabeled corpus, and then fine-turned on the labeled datasets for the downstream tasks. However, as illustrated in Figure 1(Top), different domain texts often contain different subject words, and have different sentiment descriptions, which lead to decreased performance induced by domain-shift (Pan and Yang, 2010)(Figure 1(Bottom)).

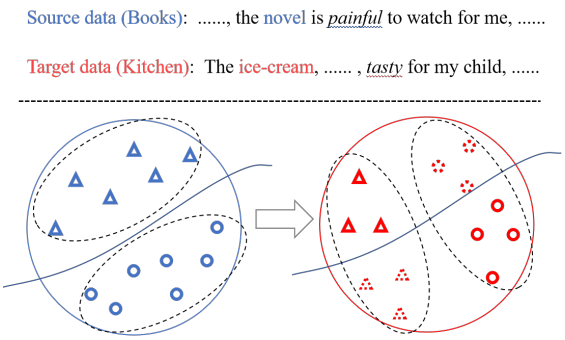

Figure 1: Illustration of domain-shift for sentiment classification. Top: different domain reviews have different subject words (marked with blue) with different sentiment descriptions (marked with underlines). Bottom: the different data distributions will lead to performance degradation.

Unsupervised domain adaptation (UDA) is another hot research topic in machine learning to address the domain-shift. It aims to transfer the knowledge from the source domain to the target domain with the labeled source data and unlabeled target data. This is often achieved by minimizing a specific distance between the source and target domains to learn the shared domain-invariant features. For example, Guo et al. (2020) exploit several distance metrics (*e.g.*, Maximum Mean Discrepancy (MMD) Gretton et al., 2012, Correlation Alignment (CORAL) Sun et al., 2016) for domain adaptation in the context of text classification tasks. While, Li et al. (2017) use a domain classifier to obtain domain-invariant sentiment features via adversarial training (Ganin et al., 2016) between the source and target features.

Despite the progress of recent cross-domain sentiment analysis works which mainly focus on

Proceedings of the 29th International Conference on Computational Linguistics, pages 6934–6943
October 12–17, 2022.

single-source domain adaptation setting, it is still cumbersome to apply these domain alignment methods on the multi-source sentiment adaptation tasks, which is more practical in real-world scenarios. Simply combining all the source domains into a single dataset may deliver worse performance compared with the best result from one of source dataset (Guo et al., 2018), due to the various source distributions. Meanwhile, with the number of source domains increasing, the corresponding computation cost and complexity will be dramatically increased (Dai et al., 2020; Xue et al., 2020). In particular, the recent dominant frameworks PrLMs (Devlin et al., 2019; Yang et al., 2019b) are adopted as the backbone for feature extraction, which usually contains a large amount of training parameters (Ye et al., 2020).

Therefore, there is a strong motivation to develop an efficient multi-source unsupervised sentiment adaptation framework which can generalize well to the target domain with no labeled target data. Recent self-training methods (He et al., 2018; Zou et al., 2019; Liu et al., 2021) achieve advanced performance on many unsupervised domain adaptation tasks by iteratively updating the pseudo-labels of the target data with current adapted model, and the model can be retrained with these self-annotated data. However, the pseudo-labels are not always reliable due to the distribution shift between the source domain and the target domain, and the incorrect pseudo-labels can significantly hurt the final adaptation performance. Several techniques are proposed to reduce the negative effect of noisy pseudo labels, such as high-confidence threshold (Zou et al., 2019), self-ensemble bootstrapping (He et al., 2018), mutual information maximization (Ye et al., 2020), *etc.*, which demonstrate improved performance on single-source cross-domain adaptation tasks. In this paper, we propose a novel Asymmetric Mutual Learning (AML) strategy to estimate the pseudo-labels robustly, and we show this strategy is well-suited to the unsupervised multi-source domain adaptation setting. Specifically, we design a classification model for each source domain. For each source model, the pseudo-labels of target data are derived from the ensembles of all the other source models. In contrast with traditional deep mutual learning (Zhang et al., 2018) which distills the knowledge of a single dataset with multiple models, our AML can utilize the knowledge from multiple datasets under different distributions. Therefore, each source model can be enhanced with the other source models. Unlike traditional self-training methods which generate pseudo-label by itself, our AML is more robust to the noisy pseudo-label.

In addition, we tend to use the features extracted from BERT for efficient sentiment adaptation, and this feature-based adaptation method is more memory-friendly compared with fine-tuning BERT. To fully exploit BERT features, we propose a dynamic network (Yang et al., 2019a) for better aggregating the features from different layers, which is referred to Dynamic Feature Networks (DFN). Compared with attention-based fusion (Vaswani et al., 2017) which only scales the features, our DFN can dynamically adjust the parameters of the network according to each instance input for better performance. Together with the two proposed modules (AML and DFN), we achieve new state-of-the-art performance on the widely-used sentiment benchmarks (Blitzer et al., 2007) under unsupervised multi-source setting. We summarize our contributions as follows:

- We propose a novel asymmetric mutual learning (AML) method, which is designed for multi-source unsupervised sentiment adaptation task and beneficial for real-world sentiment analysis applications.
- To achieve efficient adaptation on sentiment classification, we propose a dynamic feature network (DFN), which allows to dynamically assemble multiple parameters for the extracted features, and not update the encoder of PrLMs during adaptation training.
- We demonstrate that the proposed model achieves SOTA performance on multiple sentiment adaptation benchmarks, and the ablation studies verify the effectiveness of each proposed module.

The remainder of the paper is organized as follows. Section 2 introduce the related work, followed by the proposed framework in Section 3. Experimental results are reported in Section 4. Conclusion is drawn in the last Section 5.

2 Related work

In this section, we mainly focus on recent related methods based on Deep Neural Networks (DNNs) due to their superior performance.

Sentiment Classification: aims to predict the sentiment polarity of a given texts. Dang et al. (2020) compare many DNN-based methods, such as Convolutional Neural Networks (CNNs) (Kim, 2014), Recurrent Neural Networks (RNNs) (Zhou et al., 2016), *etc*. However, these methods often use word embedding or TF-IDF as the representations of the texts, which can not capture the context information within a sentence. Recently, with the advent of pre-trained language models which achieve impressive performance in many NLP tasks (Devlin et al., 2019), more and more works adopt these PrLMs as the backbone for sentiment analysis (Sun et al., 2019a; Dang et al., 2020). Despite their great success, the performance of these models is still suffering from domain-shift of the datasets (Li et al., 2021).

Unsupervised Domain Adaptation: is an attractive topic for dealing with the domain-shift problem. The mainstream is to reduce the distribution discrepancy between the source and the target domains (Ganin et al., 2016; Guo et al., 2018). For sentiment classification tasks, some previous works aim to identify domain-invariant pivot words (Ziser and Reichart, 2018; Li et al., 2018). However, pivot words identification is tedious and may be inaccurate. Ganin et al. (2016) and Li et al. (2017) tend to minimize the whole sentence representation by a binary domain classifier. As for the more challenging multi-source adaptation setting, mixture-of-experts (Guo et al., 2018) aligns the each domain-pair based on MMD for simplicity, and ensemble all the source prediction based on the distance metric. While, Zhao et al. (2018) uses a multi-class domain classifier to align multi-domain distributions and Dai et al. (2020) incorporates pseudo-labels to further improve the performance. Fu and Liu (2022) share a similar idea but using BERT as the backbone. In contrast to most domain-alignment methods which become complex with the number of source domains increasing, we turn to self-training methods which demonstrate effective performance for UDA (Zou et al., 2019; Liu et al., 2021; Dai et al., 2020), and the asymmetric mutual learning (AML) is proposed for robust pseudo-label generation in multi-source adaptation setting.

Dynamic Networks: aim to adjust the networks' architectures or parameters conditioned on each input (Yang et al., 2019a). SkipNet (Wang et al., 2018) can decide whether a block is kept and not, which can significantly reduce the inference time. CondConv (Yang et al., 2019a) can select the best combination of the convolution parameters dynamically, which increase model capacity with marginal computation cost. DyCNN (Chen et al., 2020) shares a similar idea, while uses softmax function to derive the attention coefficiency for each convolution kernel. In this paper, we are interested in adapting the features extracted from BERT for simplicity and efficiency. Thus, we propose a dynamic feature network (DFN) to fully exploit the features and adjust the network parameters accordingly for better performance.

3 Method

In this section, we first introduce the overall framework for multi-source unsupervised sentiment adaptation. Next, we provide further details of each proposed module. The detailed training procedures are presented in the last section.

3.1 Overall Framework

For multi-source unsupervised domain adaptation setting, there are k labeled source domains $\mathcal{S} = \{\mathcal{S}_i\}_{i=1}^{k}$ (where $\mathcal{S}_i = \{x_t^{\mathcal{S}_i}, y_t^{\mathcal{S}_i}\}_{t=1}^{|\mathcal{S}_i|}$) and an unlabeled target domain $\mathcal{T} = \{x_t^{\mathcal{T}}\}_{t=1}^{|\mathcal{T}|}$, $|\cdot|$ indicates number of samples in the domain. All these domains have different data distributions: $\mathcal{P}_{\mathcal{S}_i} \neq \mathcal{P}_{\mathcal{T}}$ and $\mathcal{P}_{\mathcal{S}_i} \neq \mathcal{P}_{\mathcal{S}_j}$. Our goal is to train a sentiment classification model with $\mathcal{S}$ and $\mathcal{T}$, which generalizes well to the target dataset.

Many previous works adopt statistic metrics or adversarial training to align the distributions between each domain-pairs, this strategy becomes unstable and complicated with the number of source domains increasing. As shown in Figure 2, we use BERT as the feature extractor for the text input, and build a classifier head for each source domain. Without explicit alignment, we design an asymmetric mutual learning method to estimate the pseudo labels of the target data directly, so that all the source classifier can be adapted to the target domain and mutually enhanced, simultaneously.

Since BERT is a large-scale pre-trained language model, we tend to only use its features for efficient adaptation and inference at test time. To this end, we propose the dynamic feature network to fully exploit the extracted features from different layers by adjusting the network parameters (See section 3.3 for details).

During test stage, we simply average the outputs of all the classifiers as the final prediction.

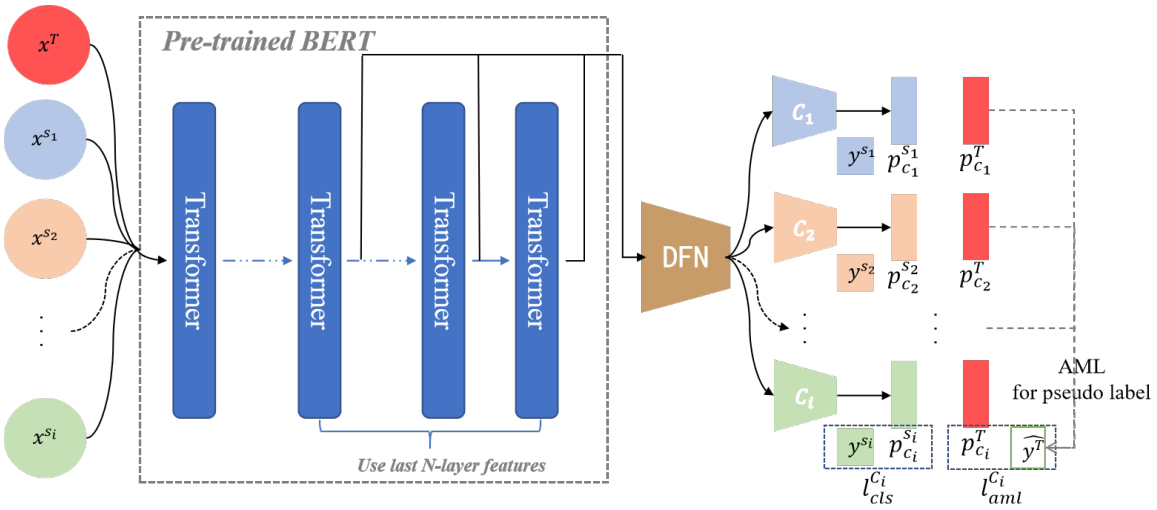

Figure 2: Overview of the proposed framework. We use the pre-trained BERT for feature extraction. DFN indicates dynamic feature network for exploiting better features with dynamic parameters. AML indicates asymmetric mutual learning for robust pseudo labels.

3.2 BERT Feature Extraction

We first extract text features with BERT (Devlin et al., 2019), which consists of several transformer layers. For each layer, we will use the representation of the first `CLS` token as features. Since the transformer layer is based on the self-attention module (Vaswani et al., 2017), the `CLS` token representation should contain all the information from a input sentence.

Given a text input x, the extracted feature from the *last* l^{th} layer can be expressed as follows:

$$\mathrm{f}_l = \mathrm{Transformer}_l^{\mathtt{CLS}}(x), \tag{1}$$

As shown in Figure 2, multiple `CLS` features from last N transformer layers in BERT can be obtained and fed to the proposed dynamic feature network (DFN) for further exploitation.

3.3 Dynamic Feature Network

The text features from BERT are powerful due to the contextualize information (Peters et al., 2019). Sun et al. (2019b) and Merchant et al. (2020) also demonstrate that features from different layers present different behaviors. Since we aim for efficient adaptation without updating the large amount of parameters within BERT extractor, fully exploiting the representations from different layers is necessary for better adaptation performance. Therefore, we propose a novel dynamic feature network (DFN) to adjust the network's parameters based on the input for dynamic adaptation of features.

As shown in Figure 3, the parameters of a fully-connected layer in DFN are a function of the input features. DFN can be adaptive to each input and choose the optimal parameters automatically. There are two advantages: our model can not only fully exploit the different layer features by discovering the optimal aggregation manner, but also increase the model's representation power with marginal computation cost.

We define the dynamic parameters of DFN as $\mathbf{W_f}$, which is conditioned on the input features $\mathbf{f}$. Following Merchant et al. (2020), we select the last N layers of BERT for input features to DFN, which means $\mathbf{f} = \{\mathrm{f}_1, \mathrm{f}_2, ..., \mathrm{f}_N\}$. Similar to Chen et al. (2020), the dynamic parameters of the DFN can be derived from combination of N linear parameters $\{\mathbf{W}_l\}_{l=1}^N$, which is defined as follows:

$$\mathbf{W_f} = \sum_{l=1}^{N} \pi_l(\mathbf{f})\mathbf{W}_l$$
$$\text{s.t.} \quad \pi_l(\mathbf{f}) \in [0,1], \quad \sum_{l=1}^{N} \pi_l(\mathbf{f}) = 1 \tag{2}$$

where the linear parameters $\{\mathbf{W}_l\}_{l=1}^N$ are trainable parameters, and $\pi_l(\mathbf{f})$ denotes the weight for the parameters $\mathbf{W}_l$, which is conditioned on the input feature $\mathbf{f}$. As shown in Figure 3, we use a two-fully connected layers followed by a softmax activation to compute the dynamic weight $\pi_l(\mathbf{f})$ for each $\mathbf{W}_l$. Therefore, the parameters within the DFN can be dynamically adjusted based on each input $\mathbf{f}$ to fully

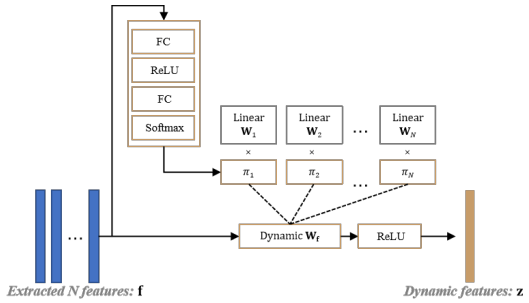

Figure 3: The architecture of the dynamic feature network. 'FC' denotes the fully-connected layer.

exploit the optimal aggregation manner for better feature adaptation performance.

The final dynamic feature representation produced by the DFN is defined as:

$$\mathbf{z} = g(\mathbf{W_f} \cdot \mathbf{f} + \mathbf{b}) \tag{3}$$

where $g(\cdot)$ is ReLU activation function used in this work.

3.4 Asymmetric Mutual Learning

We argue that explicitly reducing the distribution gap between each source-target pair is cumbersome for multi-source adaptation setting, especially for the domain adversarial training scheme (Ganin et al., 2016), which is often unstable and increase the training difficulties (Guo et al., 2018). A promising alternative is to estimate the pseudo-labels for the target domain to guide the adaptation to the target domain, iteratively. Nevertheless, the pseudo-label generated by the model itself, *i.e.*, self-training, will inevitably contain noise (Liu et al., 2021), which can hurt performance seriously.

As shown in Figure 2, we build a classification head C_i for each source domain $\mathcal{S}_i$. Therefore, we can generate the pseudo-label for one classifier by ensembling the output from all the other source domains $\{C_j\}_{j\neq i}$, which is called Asymmetric Mutual Learning (AML). We define the output of C_i as $p_{\theta_{C_i}}(\mathbf{z})$, where $\mathbf{z}$ is the output from DFN, and θ_{C_i} includes the trainable parameters within the DFN. The pseudo-label of a target representation $\mathbf{z}^{\mathcal{T}}$ for C_i can be derived as $\hat{y^{\mathcal{T}}} = \frac{\sum_{j\neq i} p_{\theta_{C_j}}(\mathbf{z}^{\mathcal{T}})}{k-1}$, where k is the number of source domain. The objective function for C_i can be formulated as follows:

$$\min_{\theta_{C_i}} \ell_{cls}^{C_i} + \ell_{aml}^{C_i}, \tag{4}$$

where ℓ_{cls} and ℓ_{aml} indicate supervised classification loss and asymmetric mutual loss, respectively. Both are defined as follows:

$$\ell_{cls}^{C_i} = \mathbb{E}_{\mathbf{z}^{\mathcal{S}_i}, y^{\mathcal{S}_i}}[-y^{\mathcal{S}_i} \log p_{\theta_{C_i}}(\mathbf{z}^{\mathcal{S}_i})] \tag{5}$$

$$\ell_{aml}^{C_i} = \mathbb{E}_{\mathbf{z}^{\mathcal{T}}} ||p_{\theta_{C_i}}(\mathbf{z}^{\mathcal{T}}) - \hat{y^{\mathcal{T}}}||_2 \tag{6}$$

where the superscript denotes the domain name.

Different from the traditional mutual learning (Zhang et al., 2018) with a single dataset, our proposed AML framework is targeted for multi-source unsupervised domain adaptation, and has multiple branches which corresponds to multiple source domains. It not only maintain the domain-specific information by training separate classifiers for corresponding source domains, but also exploit the complementary knowledge from all the other source domains to estimate the pseudo-label of target data for adaptation. It is noting that the Eq. 4 contains two parts, the first supervised source training enables diversified source classifiers, which in turn provide more robust ensembled pseudo-labels of target data for the AML training.

Therefore, during adaptation process, all the source models can be collaboratively enhanced with each other, and exploit robust target knowledge from diverse source models for better multi-source adaptation results.

3.5 Training Procedures

We proceed with the training by alternately optimizing C_i for each source classifier based on the loss objective shown in Eq. 4. The detailed optimization procedure is summarized in Algorithm 1.

During test, we average the output of all the classifiers $\{C_i\}_{i=1}^{k}$ for the final prediction.

4 Experiments

In this section, we extensively evaluate our model on two widely-used sentiment adaptation benchmarks, Amazon view datasets [1] and Skytrax view datasets [2]. First, we introduce the datasets, experiment setup, and implementation details. Then, the performance of recent state-of-the-art adaptation methods are reported for comparisons. Besides, we also conduct detailed ablation studies to verify the contribution of each proposed module.

4.1 Experimental Settings

Amazon view dataset: contains reviews from four products, namely, books (B), DVD (D), electron-

[1] https://www.cs.jhu.edu/ mdredze/datasets/sentiment/

[2] https://github.com/quankiquanki/skytrax-reviews-dataset

Method	D, E, K → B	B, E, K → D	B, D, K → E	B, D, E → K	Avg.
Previous methods					
DANN (Ganin et al., 2016)	0.779	0.789	0.849	0.864	0.820
MDAN (Zhao et al., 2018)	0.786	0.807	0.853	0.863	0.827
MoE (Guo et al., 2018)	0.794	0.834	0.866	0.880	0.843
2ST-UDA (Dai et al., 2020)	0.799	0.839	0.851	0.877	0.841
CTDA (Fu and Liu, 2022)	0.800	0.839	0.866	0.880	0.846
Our methods					
Single-best	0.837	0.831	0.857	0.872	0.849
Source-combined	0.832	0.843	0.863	0.885	0.856
Our model	**0.852**	**0.856**	**0.880**	**0.892**	**0.870**

Table 1: Comparison of multi-source unsupervised domain adaptation results on Amazon review datasets. The best results are denoted with **bold**.

Algorithm 1 Pseudo-code of AML

Input: Extracted last N BERT features for all the domains, mini-batch size B, learning rates ζ_{C_i} for each classifier C_i, $i \in [0, k]$;

Output: θ_{C_i}, $i \in [0, k]$;

for $epoch$ = 1 to N **do**
  for i = 1 to k **do**
    for each mini-batch in the $\mathcal{S}_i$ domain **do**
      Randomly sample target features;
      Compute the pseudo-label $\hat{y}$ with the other $\{C_j\}_{j \neq i}$
      Update C_i via: $\theta_{C_i} \leftarrow \texttt{Adam}(\nabla_{\theta_{C_i}}(\ell_{cls}^{C_i} + \ell_{aml}^{C_i}), \theta_{C_i}, \zeta_{C_i})$;
    end for
  end for
end for

ics (E), kitchen (K). Each produce represents one domains, and has 1,000 positive reviews (label 1) and 1,000 negative reviews (label 2), while has different number of unlabeled reviews. Following similar multi-source unsupervised domain adaptation adopted in (Fu and Liu, 2022), we select one of domain as target domain, the rest domains are used as multi-source domains.

Skytrax view dataset: includes two air-travel-related reviews from *skytrax* website, *i.e.,* Airline (AL) and Airport (AP), which contain 41,396 and 17,721 reviews, respectively. The data distribution discrepancy between the Amazon product views and air-travel reviews should be large, we use all four product datasets as source domains and one of Skytrax view dataset as the target domain, to demonstrate the effectiveness of our proposed method in this challenging settings. To align with Amazon view datasets, we randomly sample 1,000 positive and 1,000 negative reviews from AL and AP domains for training.

Implementation details: In all experiments, we use the pre-trained BERT_{base}-uncased (Devlin et al., 2019) to extract features from the last 4 transformer layers, which demonstrates both effectiveness and efficiency, reported in (Peters et al., 2019; Merchant et al., 2020). All the classifiers have the same architecture, which includes a DFN module, followed by two fully-connected layers. We use Adam (Kingma and Ba, 2015) optimizer and set the learning rate to 10^{-4}, weight decay to 10^{-4}, batch size to 16.

4.2 Experimental Results

Results on Amazon review benchmarks: Table 1 compares the sentiment classification accuracies of our method and recent multi-source unsupervised adaptation methods on Amazon review benchmarks. It is noted that our method achieves the best performance on all the multi-source adaptation tasks, which demonstrates the superiority of our proposed method.

Most previous unsupervised multi-source sentiment adaptation methods adopt word embedding as features, which is lack of contextualized information for each word. We show that the BERT features adopted by our method can achieve relatively better results. '*Source-combined*' indicates that we train the DFN-based source classifier using the combination of all the labeled data in the multi-source domains, which is demonstrated to be a strong baseline for multi-source unsupervised domain adaptation tasks (Guo et al., 2018). As shown in Table 1, '*Source-combined*' achieves compara-

Method	B, D, E, K → AL	B, D, E, K → AP	Avg.
Single-best	0.841	0.687	0.764
Source-combined	0.832	0.680	0.756
Our model	**0.850**	**0.695**	**0.772**

Table 2: Adaptation performance from multiple product review domains (Amazon) to one of air-travel review domains (Skytrax). The best results are denoted with **bold**.

Method	D, E, K → B	B, E, K → D	B, D, K → E	B, D, E → K
Without DFN				
Last features	0.831	0.845	0.873	0.873
Avg features	0.836	0.839	0.868	0.878
Our model	**0.852**	**0.856**	**0.880**	**0.892**

Table 3: Ablation study the effect of the proposed DFN module. The best results are denoted with **bold**.

ble or better results on all the adaptation setting with an average accuracy of 85.6%, which outperforms previous methods by around 1 to 3 percentage points. In addition, our AML-based training strategy achieves accuracy of 87.0% on average, which surpasses the strong baseline (*Source-combined*') method and the most recent method (Fu and Liu, 2022) (84.6%) by 1.4 and 2.4 percentage points, respectively.

Adaptation Results from Amazon to Skytrax: Table 2 reports the performance of adaptation to AL and AP domains by using all the Amazon review datasets as multi-source domains. '*Source-best*' indicates the best performance achieved under the single-source domain adaptation setting, the corresponding source domain often has more similar distribution to the target domain. Due to the large domain gap between the product reviews in Amazon and air-travel-related reviews in Skytrax, negative transfer is often occurred. As shown in Table 2, the '*Source-combined*' performance is worse than that of '*Single-best*' baseline in both tasks, which indicates that the extra source data are not fully leveraged, and the distribution shift among multi-source domains brings about the negative effects. While, our model based on the AML strategy can consistently improve the final adaptation performance, which achieves 85.0% and 69.5% accuracy on the AL domain and AP domain, respectively. The corresponding average accuracy is better than the baselines by 1 to 2 percentage points.

4.3 Ablation Study

Effectiveness of AML: We make comparison among '*Source-combined*', '*Single-best*' and our AML-based method to illustrate the effectiveness of the AML module. We show that simply combining all the source data together may hurt the final result due to various source domain distributions. As shown in Table 1, for the $\mathrm{D, E, K} \rightarrow \mathrm{B}$ task, '*Source-best*' surpasses '*Source-combined*' by 0.5 percentage point. As shown in Table 2, both two tasks demonstrate the same results. While, our AML-based adaptation can achieve consistently improved performance without negative transfer in all multi-source sentiment adaptation tasks. We consider that AML makes best of each domain's specific knowledge and enables collaboration among multiple source classifiers to address the negative transfer, so that delivers better performance than both '*Source-best*' and '*Source-combined*' baselines. We also tried the traditional self-training methods on the adaptation from Amazon domains to the Skytrax domain, and found that the accuracy quickly drops caused by the noisy label generated by the model itself. We speculate that during our proposed AML adaptation, each source model learns from the other models, which will not accumulate the same errors as done in self-training. Therefore, AML is more robust to the noisy-labels, and more effective and suitable in the multi-source unsupervised domain adaptation tasks.

Effectiveness of DFN: We conduct several experiments to verify the effect of DFN. As shown in Table 3, we first use the last transformer layer features of BERT as input, which indicates '*last features*'. We also use the same features (last 4 transformer layers) as input, but just average them without DFN module, which indicates '*Avg features*'. It can be observed that simply averaging the features from

different layers of BERT delivers comparable performance with only using the last features (less than 1 percentage point in most cases). However, our DFN module can dynamic adjust the network's parameters for better exploiting the input features from different layers. Therefore, the corresponding performance is consistently better than the above two baselines as reported in Table 3, which demonstrate the effectiveness of DFN.

5 Conclusion

In this paper, we propose a novel framework for multi-source unsupervised domain adaptation on sentiment classification. To achieve efficient adaptation with the recent large-scale and powerful pre-trained BERT model, we propose a dynamic feature network to find the optimal network parameters for better features exploitation. Besides, instead of explicitly reducing the distribution discrepancy between domain pairs which becomes complex with the number of source domain increasing, we design a asymmetric mutual learning strategy to estimate the pseudo-label of the target data directly. We conduct extensive experiments and ablation studies that verify the effectiveness and superiority of our proposed model.

Acknowledgments. This work was supported in part by National Natural Science Foundation of China (Project No.62106136), in part by the Natural Science Foundation of Guangdong Province (Project No.2022A1515010434), in part by Shantou University under Project NTF20007 and Project NTF22012.

References

John Blitzer, Mark Dredze, and Fernando Pereira. 2007. Biographies, bollywood, boom-boxes and blenders: Domain adaptation for sentiment classification. In *ACL 2007, Proceedings of the 45th Annual Meeting of the Association for Computational Linguistics*.

Erik Cambria, Yang Li, Frank Z. Xing, Soujanya Poria, and Kenneth Kwok. 2020. Senticnet 6: Ensemble application of symbolic and subsymbolic AI for sentiment analysis. In *ACM International Conference on Information and Knowledge Management*, pages 105–114. ACM.

Yinpeng Chen, Xiyang Dai, Mengchen Liu, Dongdong Chen, Lu Yuan, and Zicheng Liu. 2020. Dynamic convolution: Attention over convolution kernels. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition, CVPR*, pages 11027–11036. Computer Vision Foundation / IEEE.

Yong Dai, Jian Liu, Xiancong Ren, and Zenglin Xu. 2020. Adversarial training based multi-source unsupervised domain adaptation for sentiment analysis. In *AAAI Conference on Artificial Intelligence*, pages 7618–7625. AAAI Press.

Nhan Cach Dang, María N. Moreno García, and Fernando de la Prieta. 2020. Sentiment analysis based on deep learning: A comparative study. *CoRR*, abs/2006.03541.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics*, pages 4171–4186. Association for Computational Linguistics.

Yanping Fu and Yun Liu. 2022. Contrastive transformer based domain adaptation for multi-source cross-domain sentiment classification. *Knowledge-Based Systems*, 245:108649.

Yaroslav Ganin, Evgeniya Ustinova, Hana Ajakan, Pascal Germain, Hugo Larochelle, François Laviolette, Mario Marchand, and Victor S. Lempitsky. 2016. Domain-adversarial training of neural networks. *J. Mach. Learn. Res.*, 17:59:1–59:35.

Arthur Gretton, Karsten M. Borgwardt, Malte J. Rasch, Bernhard Schölkopf, and Alexander J. Smola. 2012. A kernel two-sample test. *J. Mach. Learn. Res.*, 13:723–773.

Han Guo, Ramakanth Pasunuru, and Mohit Bansal. 2020. Multi-source domain adaptation for text classification via distancenet-bandits. In *AAAI Conference on Artificial Intelligence*, pages 7830–7838.

Jiang Guo, Darsh J. Shah, and Regina Barzilay. 2018. Multi-source domain adaptation with mixture of experts. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 4694–4703. Association for Computational Linguistics.

Ruidan He, Wee Sun Lee, Hwee Tou Ng, and Daniel Dahlmeier. 2018. Adaptive semi-supervised learning for cross-domain sentiment classification. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, pages 3467–3476. Association for Computational Linguistics.

Jahanzeb Jabbar, Iqra Urooj, JunSheng Wu, and Naqash Azeem. 2019. Real-time sentiment analysis on e-commerce application. In *16th IEEE International Conference on Networking, Sensing and Control*, pages 391–396. IEEE.

Yoon Kim. 2014. Convolutional neural networks for sentence classification. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing, EMNLP*, pages 1746–1751. ACL.

Diederik P. Kingma and Jimmy Ba. 2015. Adam: A method for stochastic optimization. In *International Conference on Learning Representations, ICLR*.

Tian Li, Xiang Chen, Shanghang Zhang, Zhen Dong, and Kurt Keutzer. 2021. Cross-domain sentiment classification with contrastive learning and mutual information maximization. In *IEEE International Conference on Acoustics, Speech and Signal Processing, ICASSP*, pages 8203–8207. IEEE.

Zheng Li, Ying Wei, Yu Zhang, and Qiang Yang. 2018. Hierarchical attention transfer network for cross-domain sentiment classification. In *AAAI Conference on Artificial Intelligence*, pages 5852–5859. AAAI Press.

Zheng Li, Yu Zhang, Ying Wei, Yuxiang Wu, and Qiang Yang. 2017. End-to-end adversarial memory network for cross-domain sentiment classification. In *International Joint Conference on Artificial Intelligence*, pages 2237–2243. ijcai.org.

Hong Liu, Jianmin Wang, and Mingsheng Long. 2021. Cycle self-training for domain adaptation. In *Advances in Neural Information Processing Systems*, pages 22968–22981.

Amil Merchant, Elahe Rahimtoroghi, Ellie Pavlick, and Ian Tenney. 2020. What happens to BERT embeddings during fine-tuning? In *Proceedings of the Third BlackboxNLP Workshop on Analyzing and Interpreting Neural Networks for NLP, BlackboxNLP@EMNLP 2020, Online*, pages 33–44. Association for Computational Linguistics.

Alvaro Ortigosa, José M. Martín, and Rosa M. Carro. 2014. Sentiment analysis in facebook and its application to e-learning. *Comput. Hum. Behav.*, 31:527–541.

Sinno Jialin Pan and Qiang Yang. 2010. A survey on transfer learning. *IEEE Transactions on Knowledge and Data Engineering*, 22(10):1345–1359.

Matthew E. Peters, Sebastian Ruder, and Noah A. Smith. 2019. To tune or not to tune? adapting pretrained representations to diverse tasks. In *Proceedings of the 4th Workshop on Representation Learning for NLP, RepL4NLP@ACL 2019*, pages 7–14. Association for Computational Linguistics.

Baochen Sun, Jiashi Feng, and Kate Saenko. 2016. Return of frustratingly easy domain adaptation. In *AAAI Conference on Artificial Intelligence*, pages 2058–2065.

Chi Sun, Luyao Huang, and Xipeng Qiu. 2019a. Utilizing BERT for aspect-based sentiment analysis via constructing auxiliary sentence. In *Proceedings of the Conference of the North American Chapter of the Association for Computational Linguistics*, pages 380–385. Association for Computational Linguistics.

Chi Sun, Xipeng Qiu, Yige Xu, and Xuanjing Huang. 2019b. How to fine-tune BERT for text classification? In *Chinese Computational Linguistics - 18th China National Conference, CCL 2019*, volume 11856 of *Lecture Notes in Computer Science*, pages 194–206. Springer.

Yosephine Susanto, Erik Cambria, Ng Bee Chin, and Amir Hussain. 2022. Ten years of sentic computing. *Cogn. Comput.*, 14(1):5–23.

Geng Tu, Jintao Wen, Cheng Liu, Dazhi Jiang, and Erik Cambria. 2022. Context- and sentiment-aware networks for emotion recognition in conversation. *IEEE Transactions on Artificial Intelligence*.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in Neural Information Processing Systems 30*, pages 5998–6008.

Xin Wang, Fisher Yu, Zi-Yi Dou, Trevor Darrell, and Joseph E. Gonzalez. 2018. Skipnet: Learning dynamic routing in convolutional networks. In *Computer Vision - ECCV*, volume 11217 of *Lecture Notes in Computer Science*, pages 420–436. Springer.

Qianming Xue, Wei Zhang, and Hongyuan Zha. 2020. Improving domain-adapted sentiment classification by deep adversarial mutual learning. In *AAAI Conference on Artificial Intelligence*, pages 9362–9369. AAAI Press.

Brandon Yang, Gabriel Bender, Quoc V. Le, and Jiquan Ngiam. 2019a. Condconv: Conditionally parameterized convolutions for efficient inference. In *Advances in Neural Information Processing Systems*, pages 1305–1316.

Zhilin Yang, Zihang Dai, Yiming Yang, Jaime G. Carbonell, Ruslan Salakhutdinov, and Quoc V. Le. 2019b. Xlnet: Generalized autoregressive pretraining for language understanding. In *Advances in Neural Information Processing Systems*, pages 5754–5764.

Hai Ye, Qingyu Tan, Ruidan He, Juntao Li, Hwee Tou Ng, and Lidong Bing. 2020. Feature adaptation of pre-trained language models across languages and domains with robust self-training. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing, EMNLP*, pages 7386–7399. Association for Computational Linguistics.

Ying Zhang, Tao Xiang, Timothy M. Hospedales, and Huchuan Lu. 2018. Deep mutual learning. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 4320–4328.

Han Zhao, Shanghang Zhang, Guanhang Wu, José M. F. Moura, João Paulo Costeira, and Geoffrey J. Gordon. 2018. Adversarial multiple source domain adaptation. In *Advances in Neural Information Processing Systems*, pages 8568–8579.

Peng Zhou, Zhenyu Qi, Suncong Zheng, Jiaming Xu, Hongyun Bao, and Bo Xu. 2016. Text classification improved by integrating bidirectional LSTM with two-dimensional max pooling. In *COLING 26th International Conference on Computational Linguistics*, pages 3485–3495.

Yftah Ziser and Roi Reichart. 2018. Pivot based language modeling for improved neural domain adaptation. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics*, pages 1241–1251. Association for Computational Linguistics.

Yang Zou, Zhiding Yu, Xiaofeng Liu, B. V. K. Vijaya Kumar, and Jinsong Wang. 2019. Confidence regularized self-training. In *IEEE/CVF International Conference on Computer Vision*, pages 5981–5990.

Target Really Matters: Target-aware Contrastive Learning and Consistency Regularization for Few-shot Stance Detection

Rui Liu[1,2], **Zheng Lin**[1,2]*, **Huishan Ji**[1,2], **Jiangnan Li**[1,2], **Peng Fu**[1] **and Weiping Wang**[1]
[1]Institute of Information Engineering, Chinese Academy of Sciences
[2]School of Cyber Security, University of Chinese Academy of Sciences
{liurui1995,linzheng,jihuishan,lijiangnan}@iie.ac.cn
{fupeng,wangweiping}@iie.ac.cn

Abstract

Stance detection aims to identify the attitude from an opinion towards a certain target. Despite the significant progress on this task, it is extremely time-consuming and budget-unfriendly to collect sufficient high-quality labeled data for every new target under fully-supervised learning, whereas unlabeled data can be collected easier. Therefore, this paper is devoted to few-shot stance detection and investigating how to achieve satisfactory results in semi-supervised settings. As a target-oriented task, the core idea of semi-supervised few-shot stance detection is to make better use of target-relevant information from labeled and unlabeled data. Therefore, we develop a novel target-aware semi-supervised framework. Specifically, we propose a target-aware contrastive learning objective to learn more distinguishable representations for different targets. Such an objective can be easily applied with or without unlabeled data. Furthermore, to thoroughly exploit the unlabeled data and facilitate the model to learn target-relevant stance features in the opinion content, we explore a simple but effective target-aware consistency regularization combined with a self-training strategy. Experimental results demonstrate that our approach can achieve state-of-the-art performance on multiple benchmark datasets in the few-shot setting.

1 Introduction

Stance detection is intended to identify the attitude of opinions towards certain targets, where labels can be favor, against, and neutral. For example, the opinion "*True equality allows all to be born.*" is against the target "*Legalization of Abortion*". The settings of stance detection can be generally divided into in-target and cross-target ones. Specifically, in-target stance detection aims to train an exclusive classifier for prediction on the same set of targets (Mohammad et al., 2016b; Augenstein et al., 2016; Li et al., 2021). Obviously, in-target stance detection is the most ideal setting when there is sufficient labeled data. However, in practice, a severe challenge is the scarcity of annotations for new targets. Cross-target stance detection (Zhang et al., 2020a; Xu et al., 2018; Wei and Mao, 2019) is to train on the labeled data of one source target and test on the destination target, which is based on an assumption that there is a strong correlation between the two targets. No doubt the harsh demand from the assumption above limits the extension of cross-target stance detection, in which it still requires a large amount of annotated data for the source target. In this paper, we focus on the in-target few-shot stance detection with unlabeled data and limited labeled data, which is to alleviate the demand for human supervision.

Different from common classification tasks (e.g., sentiment classification), the identification of stance is heavily dependent on the specific target (Siddiqua et al., 2019). As a target-oriented task, the key problem of few-shot stance detection is how to thoroughly exploit the target-relevant information from the limited labeled data and the unlabeled data. Existing methods like supervised contrastive learning (Gunel et al., 2021) and semi-supervised learning (Sohn et al., 2020) prove significant effectiveness in the few-shot setting. When only a few labeled samples are provided, supervised contrastive learning (SCL) hopes to improve the representation ability of the model to a certain extent by leveraging class label information. However, such an objective ignores target label information, the crucial clue, which plays a vital role in stance detection. When the unlabeled data is available, semi-supervised learning (SSL) like self-training and consistency regularization, is a way of bringing unlabeled data into full play, which helps to overcome the scarcity of sufficient annotated data. However, self-training algorithm (Glandt

*Zheng Lin is the corresponding author.

Proceedings of the 29th International Conference on Computational Linguistics, pages 6944–6954
October 12–17, 2022.

et al., 2021; Li et al., 2021) only encourages the student network to mimic the teacher network's label predictions simply. Moreover, consistency regularization (Xie et al., 2020a; Sohn et al., 2020) constrains the model to make consistent predictions of the same example under some task-agnostic data augmentation strategies. Neither of them digs deeply into target-relevant stance features in the opinion content.

To further tackle the challenges above, we propose a novel **S**emi-supervised framework with **T**arget-aware **C**ontrastive learning and **C**onsistency regularization (**STCC**). First, we introduce a target-aware contrastive learning objective to consider both target and stance label information, which promotes the distinction and isolation of samples from different targets, as well as different classes. Since the target-aware contrastive learning objective learns more distinguishable representations, the model trained here can be used in semi-supervised learning as a better teacher model. Second, we combine two approaches to SSL: consistency regularization and self-training. Specifically, we design a simple but effective target-aware data augmentation strategy for consistency regularization, i.e., masking the corresponding target for every unlabeled example, so that the model can perceive the target-related information contained in the content as much as possible. Moreover, our proposed target-aware contrastive learning objective can be naturally extended to semi-supervised learning with additional unlabeled data. Specifically, the pseudo-labels generated by the teacher model are used for the contrastive learning objective. Finally, we conduct comprehensive experiments on two large datasets of SemEval-2016 and COVID-19-Stance to verify the effectiveness of our framework for few-shot stance detection. The experimental results demonstrate that our approach achieves the state-of-the-art performance[1].

2 Methodology

2.1 Problem Definition

The semi-supervised few-shot stance detection is to train a classifier by leveraging labeled and unlabeled data, which identifies the users' stance from the context and the corresponding target. Formally, given a collection of limited labeled data $\mathcal{X} = \{(x_i, t_i, y_i)\}_{i=1}^{N_l}$ and a collection of unlabeled data $\mathcal{U} = \{x_i, t_i\}_{j=1}^{N_u}$, where x_i is the opinion content, t_i is the corresponding target, y_i is the stance label, N_l is the number of the labeled data and N_u is the number of the unlabeled data.

2.2 BERT Model

First of all, we select the pretrained model BERT as the encoder. For the labeled data $\mathcal{X}$, we concatenate the content x_i and the target t_i of each sample in the following format: [CLS] $\boldsymbol{t}_i$ [SEP] $\boldsymbol{x}_i$ [SEP] and utilize BERT to process it. We then feed the representation $\boldsymbol{h}_i^{[CLS]}$ of [CLS] from the last layer of BERT into the final classification layer. Finally, we compute the probability distribution with the softmax function: $p(\hat{y}_i|x_i, t_i) = softmax(W_h h_i^{[CLS]})$, where W_h is a trainable matrix. We fine-tune the model by minimizing the cross-entropy loss:

$$\mathcal{L}_{ce} = -\frac{1}{|\mathcal{X}|}\sum_{i}^{|\mathcal{X}|} CE(p(\hat{y}_i|x_i, t_i), y_i), \quad (1)$$

where CE denotes the cross entropy loss function.

2.3 Target-aware Contrastive Learning

The core idea of contrastive learning is to bring the representations of positive sample pairs closer, and push the negative sample pairs farther, to learn more distinguishable representations. In supervised contrastive learning, samples under the same label match each other as positive pairs and samples of different labels match as negative pairs. Given the index $I = \{1, \cdots, B\}$ in a batch, the supervised contrastive learning loss is formulated as:

$$\mathcal{L}_{sup}^{i} = -\sum_{i=1}^{B}\frac{1}{|\mathcal{N}_i|}\sum_{j\in\mathcal{N}_i}\log\frac{e^{sim(\boldsymbol{h}_i,\boldsymbol{h}_j/\tau)}}{\sum_{k\in C(i)} e^{sim(\boldsymbol{h}_i\cdot\boldsymbol{h}_k/\tau)}}, \quad (2)$$

where B is the batch size, $\mathcal{N}_i = \{\boldsymbol{h}_j | i \neq j, y_i = y_j\}$ is the positive examples of $\boldsymbol{h}_i$, $|\mathcal{N}_i|$ is the number of examples labeled as y_i in the same batch, $C(i) \equiv I\backslash\{i\}$, and τ is temperature parameter.

It is our goal to integrate target label information into contrastive training, enabling the model to learn the target-specific information adequately under the setting of few labeled data. Specifically, as shown in Figure 1, target-aware contrastive training (TCL) tries to make intra-target representations being more compact in the feature space and inter-target ones more distinguishable. The target-aware

[1]Our code and data are available at https://github.com/monolith-v1/STCC.

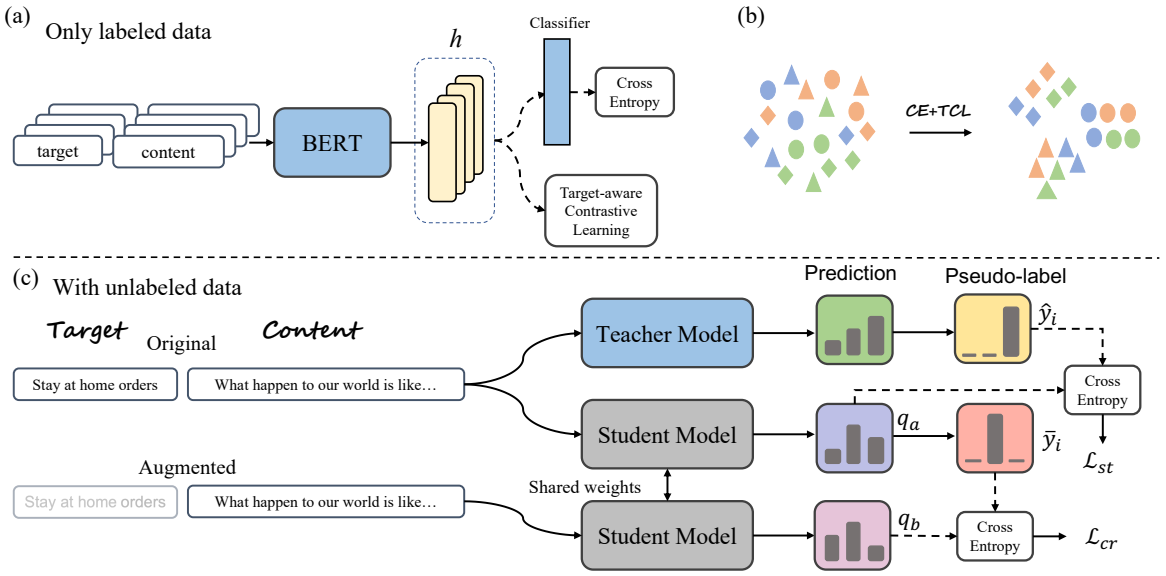

Figure 1: Structure overview of our target-aware semi-supervised framework. There are mainly three parts in the figure: **(a)** The teacher model, aiming at generating pseudo-labels, is trained on labeled data with a combination of cross-entropy (CE) loss and target-aware contrastive learning (TCL) objective. **(b)** an illustration for TCL, in which different shapes stand for different targets, and different colors represent different stance labels. **(c)** is the semi-supervised learning procedure. The teacher model comes from part (a), fed with original data, and its prediction is then softmaxed and regarded as pseudo-labels $\hat{y}_i$. Simultaneously, the original data are also fed into the student model, generating a prediction q_a and pseudo-labels $\bar{y}_i$. Then, pseudo-labels $\hat{y}_i$ from the teacher model and q_a are used to compute the self-training loss $\mathcal{L}_{st}$. Afterwards, in another branch, the student model is fed with augmented data to generate a prediction q_b. Consistency regularization term $\mathcal{L}_{cr}$ is then computed by q_b and $\bar{y}_i$.

contrastive learning loss can be written as:

$$\mathcal{L}_{tcl} = -\sum_{i=1}^{B} \mathcal{L}_{tcl}^{i} \tag{3}$$

$$\mathcal{L}_{tcl}^{i} = \frac{1}{|\mathcal{T}_i|} \sum_{j \in \mathcal{T}_i} \log \frac{e^{sim(\boldsymbol{h}_i, \boldsymbol{h}_j/\tau)}}{\sum_{k \in C(i)} e^{sim(\boldsymbol{h}_i \cdot \boldsymbol{h}_k/\tau)}}, \tag{4}$$

where $\mathcal{T}_i = \{\boldsymbol{h}_j | i \neq j, y_i = y_j, t_i = t_j\}$ is the positive examples of $\boldsymbol{h}_i$, $|\mathcal{T}_i|$ is the number of examples with the target t_i labeled as y_i in the same batch, and τ is temperature parameter.

The above trained model can be utilized alone or as a teacher model to participate in semi-supervised training. Note that TCL can also be applied to semi-supervised learning. For the labeled data, the annotated labels are used for picking positive and negative pairs. For the unlabeled data, the teacher model generates pseudo-labels as a complement.

2.4 Target-Aware Consistency Regularization in Semi-supervised Learning

Consistency regularization utilizes data augmentation to add perturbations for the unlabeled data. An ideal model is ought to make consistent predictions of samples before and after adding perturbation. Therefore, such property makes it possible for us to tailor target-aware data augmentation strategy, which can facilitate model mining the target-relevant stance features in the content. Practically, given the prediction distribution of the original data, $q_a = p_s(\bar{y}_i | x_i, t_i)$, the $\bar{y}_i = argmax(q_a)$ is used as pseudo-labels in the later process. There is a simple but effective way of acquiring the augmented version of the unlabeled data for stance detection here. Specifically, the prediction of an augmented sample $q_b = p_s(\tilde{y}_i | x_i)$ from the model can be generated by masking the corresponding target t_i of the content x_i.

$$\mathcal{L}_{cr} = -\frac{1}{|\mathcal{U}|} \sum_{i}^{|\mathcal{U}|} CE(q_b, \bar{y}_i), \tag{5}$$

Note that conventional consistency regularization performs a supervised training from the labeled data as well, of which the loss is computed by Eq (1). Such a process utilizes the pseudo-labels generated by the model under training. In consistency training, the model trained from scratch has a low accuracy and high entropy, which prevents the model from achieving good accuracy (Xie et al., 2020b). Therefore, we incorporate the self-training strategy into consistency training to

Algorithm 1 Target-aware Semi-supervised Learning for Few-shot Stance Detection

Require: Labeled data $\mathcal{X}$ and unlabeled data $\mathcal{U}$.
Require: $\Delta = 20$ ▷ The increment every step
$K = 100$ ▷ The initial threshold
$t = 1$ ▷ The time step
Train the teacher model θ_t using labeled data via Eq (1) and Eq (4).
repeat
  $K = K - \Delta$
  Generate pseudo labels $\hat{y}_i$ using the teacher model θ_t for unlabeled data.
  Select unlabeled data with top $K\%$ high-confidence pseudo labels.
  Update student model θ_s using the combination of labeled data and unlabeled data via Eq (7).
  Using the student model θ_s as the teacher model θ_t for next iteration.
  $t = t + 1$
until $K \neq 0$
return θ_s

stabilize the entire training process. Instead of using the model trained from scratch, we train a teacher model ahead to generate high-quality pseudo-labels. Pseudo-labels generated by the teacher model can stabilize the training of a student model that uses these generated pseudo-labels, following the loss function:

$$\mathcal{L}_{st} = -\frac{1}{|\mathcal{U}|}\sum_{i}^{|\mathcal{U}|} CE(q_a, \hat{y}_i), \tag{6}$$

where, $\hat{y}_i$ is the pseudo labels generated by the teacher model. Inspired by a self-training method of curriculum labeling (Cascante-Bonilla et al., 2021), which applies self-paced curriculum learning principles in each self-training cycle, we select samples of the top $K\%$ highest confidence from the entire unlabeled dataset in each iteration with an increment Δ.

The total loss for the semi-supervised learning is as follows:

$$\mathcal{L} = \mathcal{L}_{ce} + \mathcal{L}_{st} + \lambda_a \mathcal{L}_{cr} + \lambda_b \mathcal{L}_{tcl}, \tag{7}$$

where λ_a and λ_b are scalar weighting hyper-parameters.

The procedure of our semi-supervised framework is summarized in Algorithm 1.

3 Experiments

3.1 Datasets and evaluation

The macro-averaged F1 is used as the evaluation metric for all datasets. We conducted experiments using two well-known standard benchmarks, SemEval-2016 and COVID-2019-Stance.

SemEval-2016 (Mohammad et al., 2016a) is the earliest dataset to detect users' stance from tweets, which contains 6 targets, specifically, "Atheism", "Climate Change is a Real Concern", "Feminist Movement", "Hillary Clinton", and "Legalization of Abortion" and "Donald Trump". The data with "Donald Trump" in the original dataset is not split into the training and test sets. We split this target's data for training and testing with a ratio of 5:2. The processed dataset has 3414 tweets for training and 1456 for testing. Additionally, we split the original training set in a ratio of 9:1 into training and validation subsets.

COVID-19-Stance (Glandt et al., 2021) consists of 6,133 tweets for stance towards four targets relevant to COVID-19 health mandates, specifically "Anthony S. Fauci, M.D.", "Keeping Schools Closed", "Stay at Home Orders", and "Wearing a Face Mask". This dataset has 4533 samples for training, 800 samples for validation, and 800 samples for testing.

3.2 Model comparisons

We compare our method with several strong baseline methods:

(1) *Some general models trained only using labeled data*:

CrossNet (Xu et al., 2018): It is a BiLSTM model for cross-target stance with an aspect-specific attention layer.

BERT (Devlin et al., 2019): It is a transformer-based language model pre-trained by two self-supervised tasks.

ProtoNets (Snell et al., 2017): It aims to learn prototypes embedding for each class, in which the model makes predictions by computing distances to prototype representations of each class.

(2) *Fine-tuning objectives:*

SCL (Gunel et al., 2021): A contrastive learning objective combining the label information with the self-supervised contrastive learning.

PT-HCL (Liang et al., 2022): This method uses a pre-text task to distinguish the types of the stance

data, then integrates the type information into the supervised contrastive learning.

(3) *Semi-supervised learning methods using the unlabeled data and few labeled data*:

Prompt (Schick and Schütze, 2021): The prompt-based learning is a fine-tuning strategy leveraging language prompts as contexts to stimulate knowledge from pre-trained Language Models, which can has the flexibility to use or not use unlabeled data.

UDA (Xie et al., 2020a): A consistency training algorithm enforcing the model predictions to be consistent between an unlabeled example and its augmented version.

ST (Glandt et al., 2021): A vanilla self-training method transferring the knowledge from the teacher model to the student model iteratively.

UPS (Rizve et al., 2021): An uncertainty-aware pseudo-label selection framework that leverages the prediction uncertainty to guide the pseudo-label selection procedure.

CL (Cascante-Bonilla et al., 2021): A pseudo-label selection framework with a hand-crafted curriculum choice strategy, which selects unlabeled samples progressively from high confidence to low confidence.

Finally, we add the model **BERT w/ full data** trained by the full labeled data to present the upper bound of the performance for all few-shot methods.

3.3 Implementation Details

In the few-shot setting, we randomly select 5, 10, and 20 samples for each target for training under different settings. For example, the size of the labeled data for SemEval-2016 under the 5-shot setting is 30. In order to obtain a relatively uniform distribution, samples of different labels, which are favor, against, and none, are picked by a ratio of 2:2:1. Under the setting of semi-supervised learning, in each of the datasets above, the rest of the data in the training set is used as the unlabeled data. We implement our model using PyTorch[2] and BERT-base from huggingface Transformers[3] is used as the backbone. The models are optimized by AdamW and the batch size is set as 32. An iteration for the self-training procedure is set as 20 epochs. We report the average results of the models using a fixed set of 5 random seeds. We set λ_a to 1.0. As few-shot learning is extremely sensitive to hyper-parameters, we conduct a grid hyper-parameters search based on the performance on the validation data for learning rate, temperature τ and scalar weighting λ_b on two different datasets across different shot sizes. The details are in Appendix A.

[2]https://pytorch.org/

[3]https://huggingface.co/docs/transformers/index

3.4 Main Results

Table 1 shows the results from baselines on SemEval2016 and COVID-19, including the performance with and without unlabeled data. We select BERT-base as the base encoder of these baselines, except for CrossNet. Table 1 is divided into two parts by whether unlabeled data is available. Models in the upper part are only trained by N-shot labeled data (minimal few-shot setting), while results in the lower part come from semi-supervised learning (semi-supervised few-shot setting).

First of all, generally, our method shows substantial improvement compared to baseline models under minimal few-shot setting. Then, specifically, the comparison of results from TCL and SCL under different datasets validates that the application of target label information in contrastive learning further improves the performance of models. Considering the significant improvement of our model over PT-HCL, it is clear that the distinguishing of whether a data is sensitive to the target or not under the minimal few-shot setting does not make the most of target-specific information.

In addition, we also would like to acknowledge that, SCL performs better than TCL under 5-shot learning on SemEval2016, but no such phenomenon is observed on COVID-19. A possible explanation is that the short length of texts of SemEval2016 impedes the model from learning general representations because TCL has to distinguish the belonged target and stance at the same time, making the number of data in each potential cluster too small to learn. Note that there is no such phenomenon in models with unlabeled data, indicating the application of unlabeled data alleviates the scarcity of data and enables STCC to develop its full potential. Moreover, the model CrossNet based on a traditional BiLSTM performs poorly under the setting of few-shot stance detection here.

Next, from the lower part of Table 1, where the results under semi-supervised learning are shown, we find that STCC outperforms all other semi-supervised methods by a great margin. Considering different settings of numbers of labeled data,

Model	SemEval2016			COVID19		
	5	10	20	5	10	20
without unlabeled data						
CrossNet (Xu et al., 2018)	29.82	33.85	35.37	31.20	34.93	44.32
BERT (Devlin et al., 2019)	41.12	44.45	49.72	32.45	36.85	50.83
ProtoNets(Snell et al., 2017)	41.50	44.13	48.72	33.90	40.50	48.44
SCL (Gunel et al., 2021)	**48.02**	49.40	52.22	37.40	42.23	52.83
PT-HCL (Liang et al., 2022)	34.72	39.56	45.22	-	-	-
Prompt (Schick and Schütze, 2021)	37.88	41.80	43.74	34.96	37.46	47.52
TCL (Ours)	47.32	**51.41**	**53.47**	**40.27**	**46.80**	**53.52**
with unlabeled data						
Prompt (Schick and Schütze, 2021)	37.93	42.41	43.80	34.96	49.42	47.11
UDA (Xie et al., 2020a)	46.86	46.77	50.87	40.27	47.02	53.52
ST (Glandt et al., 2021)	48.35	51.12	55.01	42.08	47.66	55.67
UPS (Rizve et al., 2021)	43.45	48.11	52.73	41.45	44.37	53.87
CL (Cascante-Bonilla et al., 2021)	48.92	51.34	55.42	40.96	50.22	56.86
STCC (Ours)	**52.84**	**55.00**	**57.11**	**44.38**	**52.26**	**58.06**
BERT w/ full data		68.34			73.12	

Table 1: Summary of test results for few-shot stance detection using the shot size of 5, 10, 20 for training. The best results are in bold.

STCC exceeds the best baselines by an average of 3.09% on SemEval 2016 and 1.84% on COVID-19. It is also verified that BERT performs better with the help of unlabeled data from the same target. At last, the increase in numbers of labeled data guarantees a steady growth for all semi-supervised learning methods, especially outstanding for ours. Besides, the performance of prompt-tuning methods whose backbone is BERT_{base} is not ideal. A possible reason is that such methods depend on the generalization ability of large pre-trained models and hand-crafted prompt designs, which are not the focus of our work.

3.5 Ablation Study

As shown in Table 2, we conduct an ablation study to inspect the importance of the components in **STCC** on SemEval2016, including the target-aware contrastive learning (TCL), the target-aware consistency regularization (TCR), and the self-training procedure (ST). It is clear that the removal of either one of our three independent modules causes the drop in performance, especially for the self-training procedure. A possible explanation is that the training of a model from the very beginning introduces consistency training under a low accuracy, forcing the model to stay in a condition of high entropy.

Moreover, compared with "-TCR&TCL" (i.e., only using ST), the performance further improves after equipping TCR, indicating the effectiveness of our proposed target-aware consistency training. The comparison among "-ST&TCL" (i.e., only using TCR), "-TCR&TCL" (i.e., only using ST) and "-TCL" (i.e., using ST&TCR) further validates that the model can achieve an acceptable performance merely with the self-training procedure. In addition, TCL is replaced by SCL (i.e., "-TCR&TCL + SCL") to verify the indispensability of TCL under the semi-supervised framework. The drop in performance compared to TCL (i.e., "-TCR") confirms that target-specific information is of vital significance in semi-supervised learning. The proposed method of TCL adapts well to few-shot stance detection with or without labeled data.

Model	5-shot	10-shot	20-shot
STCC	52.84	55.00	57.11
-TCR	51.32	53.53	56.48
-TCL	49.91	52.32	56.18
-ST	43.45	47.70	52.65
-ST&TCR	47.32	51.41	53.47
-ST&TCL	45.19	46.29	48.96
-TCR&TCL	48.92	51.34	55.42
-TCR&TCL + SCL	49.32	51.40	55.92
-ST&TCR&TCL	41.12	44.45	49.72

Table 2: Ablation results on SemEval-2016. "ST" means self-training procedure, and "TCR" means target-aware consistency regularization.

To demonstrate the effectiveness of our proposed data augmentation, i.e., "masking the target", we

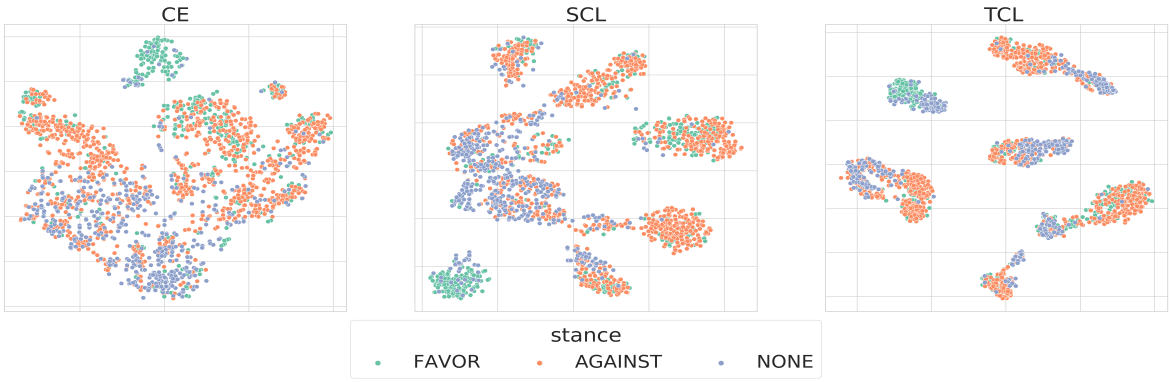

Figure 2: 2D t-SNE plots of the learned [CLS] representations on the unlabeled data of the SemEval2016 from the BERT models only trained by 20-shot labeled samples for every target, which are fine-tuned on different objectives CE (left), SCL (middle), and TCL (right).

Model	5-shot	10-shot	20-shot
NT	52.84	55.00	57.11
BT	52.13	52.21	55.96
SR	52.26	54.28	55.23
RD	50.92	52.35	55.99

Table 3: The performance of different data augmentation in semi-supervised learning on SemEval2016. "NT": masking the target, "BT": back translation, "SR": synonym replacement, "RD": random deletion.

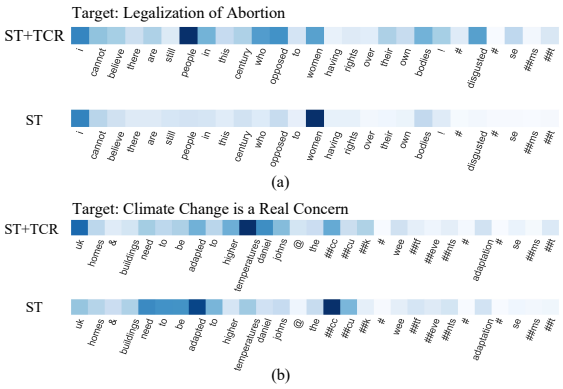

Figure 3: The heatmap of the attention weights of [CLS] towards each subword in the content for "ST+TCR" and "ST" under semi-supervised setting. The attention weights are averaged from the multi-heads in the top layer. The darker the color, the greater the weight.

replace it with multiple common data augmentation methods in NLP, and the results are shown in Table 3. Evidently, our method performs best for the current task. Back translation is implemented by the toolbox of nlpaug[4], while synonym replacement and random deletion are from EDA (Wei and Zou, 2019) method[5]. Note that although common data augmentation methods improve the results as well, none of the improvements is as significant as ours.

3.6 Visualization

Visualization for TCL In Figure 2, the t-SNE plots from the representations of [CLS] are shown. Such representations are the output from BERT-base fine-tuned by different objectives, under the setting of 20 available labeled samples. As observed, for the case of using cross-entropy loss only, all samples are promiscuously scattered. There is a similar but better distribution for the model trained by SCL, where most samples are mixed. For TCL, there are six obviously independent clusters, as there are six targets in SemEval2016. Furthermore, even inside a cluster itself, representations from the same target but of different labels can be identified and are much more separate than that in the other two diagrams. Such a phenomenon verifies TCL helps to learn better representations, increasing intra-target compactness and inter-target discrepancy, which improves the performance of the model.

Visualization for TCR In Figure 3, we show the heatmap of attention weights to demonstrate the effectiveness of our proposed target-aware consistency regularization. As the classifier deals with the representations from "[CLS]", we pick the attention-weight matrix of "[CLS]" from the top layer towards each subword in the content. In order to avoid possible influence from TCL, the model is trained by "ST" and "ST+TCR" respectively. Take (a) in Figure 3 for an example, whose target is "Legalization of Abortion", the model trained

[4]https://github.com/makcedward/nlpaug
[5]https://github.com/jasonwei20/eda_nlp

by "ST+TCR" notices target-relevant words like "people, woman, opposed and disgusted", while "ST" merely pays attention to "woman". Such a phenomenon confirms that the proposed method of target-aware consistency regularization improves the ability of digging target-relevant information for models.

4 Related Work

Stance Detection Stance detection aims to identify the attitude from an opinion towards a certain target. Incipient studies focus on in-target stance detection (Augenstein et al., 2016; Siddiqua et al., 2019; Mohammad et al., 2016b; Du et al., 2017; Wei et al., 2019), which only train the model and perform the prediction on a single target. Li et al. (2021) investigated the multi-target training and knowledge distillation in the stance detection task. Data augmentation (Li and Caragea, 2021) has been used for in-target setting to improve performance in fully supervised learning. Cross-target stance detection (Zhang et al., 2020b; Wei and Mao, 2019; Allaway et al., 2021) hopes to transfer knowledge between related targets, which attempts to mitigate the lack of labeled training data for a new target. Dutta et al. (2022) focused on semi-supervised user stance detection using the information from tweets posted by users and their followers, whereas we do not consider specific user information. Hardalov et al. (2021) studied few-shot cross-lingual stance detection, which transferred the knowledge from English resources to non-English scenarios. Recently, Allaway and McKeown (2020) defined zero-shot and few-shot stance detection, according to which the targets have no or very few training examples. And, they present a new dataset VAST, which consists of thousands different targets. However, VAST includes a wide range of similar expressions for one target (e.g., "guns on campus" versus "firearms on campus"). The situation above makes the source of the model's benefit too ambiguous to trace. Therefore, SemEval-2016 and COVID-19-Stance are relatively much more accessible for studying in the few-shot setting, compared with VAST.

Contrastive Learning In recent years, contrastive learning has made significant progress in self-supervised representation learning, both in the CV (Chen et al., 2020) and NLP (Gao et al., 2021) domains. Khosla et al. (2020) introduced supervised contrastive learning (SCL), which further extended the self-supervised contrastive learning to the fully-supervised setting by leveraging label information. Gunel et al. (2021) integrated the SCL objective for fine-tuning pre-trained language models, which significantly improves the performance in the few-shot learning settings. In stance detection, Liang et al. (2022) proposed a hierarchical contrastive learning loss to take both the data types and the stance labels into account. They subdivided the data types by judging whether a sample is sensitive to its corresponding target based on the self-supervised learning pretext task. However, this strategy is not suitable for the few-shot setting, which further reduces the target-related information availability to the model.

Semi-supervised Learning Consistency regularization, pseudo-labeling, and self-training are all important components of semi-supervised learning. Consistency regularization (Xie et al., 2020a; Sohn et al., 2020) constrains the model to make consistent predictions of the same example under varied noises. And, pseudo-labeling (Lee, 2013) selects those unlabeled data with high confidence as a form of entropy minimization. These methods use the model being trained to generate pseudo-labels instead of a separate teacher model pre-trained on labeled data. Self-training (Xie et al., 2020b) allows a teacher model pre-trained on labeled data, and then applies the combination of labeled and pseudo-labeled data to retrain a student model. Recent, semi-supervised learning methods have combined those techniques to some extent. Cascante-Bonilla et al. (2021); Rizve et al. (2021) combine the self-training process with pseudo-labeling, while using curriculum labeling and uncertainty-aware techniques to improve the filter process for unlabeled data. Sohn et al. (2020) unify the consistency regularization and pseudo-labeling. However, these task-agnostic methods cannot adequately mine target-relevant stance features in the opinion content for stance detection.

5 Conclusion

In this paper, we focus on in-target few-shot stance detection to alleviate the demand for human supervision, and propose a novel target-aware semi-supervised framework with contrastive learning and consistency regularization. The target-aware contrastive learning objective performs well on both labeled and unlabeled data, which promotes the model's ability to distinguish various classes

and targets. Moreover, our proposed target-aware consistency regularization is validated to be more efficient in mining target-relevant stance features in the content. Experiments on two popular benchmarks demonstrate the effectiveness and consistent improvements over baselines.

Acknowledgments

This work was supported by National Natural Science Foundation of China (No. 61976207, No. 61906187).

References

Emily Allaway and Kathleen McKeown. 2020. Zero-Shot Stance Detection: A Dataset and Model using Generalized Topic Representations. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 8913–8931, Online. Association for Computational Linguistics.

Emily Allaway, Malavika Srikanth, and Kathleen R. McKeown. 2021. Adversarial learning for zero-shot stance detection on social media. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT 2021, Online, June 6-11, 2021*, pages 4756–4767. Association for Computational Linguistics.

Isabelle Augenstein, Tim Rocktäschel, Andreas Vlachos, and Kalina Bontcheva. 2016. Stance detection with bidirectional conditional encoding. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 876–885, Austin, Texas. Association for Computational Linguistics.

Paola Cascante-Bonilla, Fuwen Tan, Yanjun Qi, and Vicente Ordonez. 2021. Curriculum labeling: Revisiting pseudo-labeling for semi-supervised learning. In *Thirty-Fifth AAAI Conference on Artificial Intelligence, AAAI 2021, Thirty-Third Conference on Innovative Applications of Artificial Intelligence, IAAI 2021, The Eleventh Symposium on Educational Advances in Artificial Intelligence, EAAI 2021, Virtual Event, February 2-9, 2021*, pages 6912–6920. AAAI Press.

Ting Chen, Simon Kornblith, Mohammad Norouzi, and Geoffrey E. Hinton. 2020. A simple framework for contrastive learning of visual representations. In *Proceedings of the 37th International Conference on Machine Learning, ICML 2020, 13-18 July 2020, Virtual Event*, volume 119 of *Proceedings of Machine Learning Research*, pages 1597–1607. PMLR.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT 2019, Minneapolis, MN, USA, June 2-7, 2019, Volume 1 (Long and Short Papers)*, pages 4171–4186. Association for Computational Linguistics.

Jiachen Du, Ruifeng Xu, Yulan He, and Lin Gui. 2017. Stance classification with target-specific neural attention. In *Proceedings of the Twenty-Sixth International Joint Conference on Artificial Intelligence, IJCAI-17*, pages 3988–3994.

Subhabrata Dutta, Samiya Caur, Soumen Chakrabarti, and Tanmoy Chakraborty. 2022. Semi-supervised stance detection of tweets via distant network supervision. In *WSDM '22: The Fifteenth ACM International Conference on Web Search and Data Mining, Virtual Event / Tempe, AZ, USA, February 21 - 25, 2022*, pages 241–251. ACM.

Tianyu Gao, Xingcheng Yao, and Danqi Chen. 2021. Simcse: Simple contrastive learning of sentence embeddings. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing, EMNLP 2021, Virtual Event / Punta Cana, Dominican Republic, 7-11 November, 2021*, pages 6894–6910. Association for Computational Linguistics.

Kyle Glandt, Sarthak Khanal, Yingjie Li, Doina Caragea, and Cornelia Caragea. 2021. Stance detection in COVID-19 tweets. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, Online. Association for Computational Linguistics.

Beliz Gunel, Jingfei Du, Alexis Conneau, and Veselin Stoyanov. 2021. Supervised contrastive learning for pre-trained language model fine-tuning. In *9th International Conference on Learning Representations, ICLR 2021, Virtual Event, Austria, May 3-7, 2021*. OpenReview.net.

Momchil Hardalov, Arnav Arora, Preslav Nakov, and Isabelle Augenstein. 2021. Few-shot cross-lingual stance detection with sentiment-based pre-training. *CoRR*, abs/2109.06050.

Prannay Khosla, Piotr Teterwak, Chen Wang, Aaron Sarna, Yonglong Tian, Phillip Isola, Aaron Maschinot, Ce Liu, and Dilip Krishnan. 2020. Supervised contrastive learning. In *Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020, December 6-12, 2020, virtual*.

Dong-Hyun Lee. 2013. Pseudo-label : The simple and efficient semi-supervised learning method for deep neural networks. *ICML 2013 Workshop : Challenges in Representation Learning (WREPL)*.

Yingjie Li and Cornelia Caragea. 2021. Target-aware data augmentation for stance detection. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, Online. Association for Computational Linguistics.

Yingjie Li, Chenye Zhao, and Cornelia Caragea. 2021. Improving stance detection with multi-dataset learning and knowledge distillation. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Bin Liang, Zixiao Chen, Lin Gui, Yulan He, Min Yang, and Ruifeng Xu. 2022. Zero-shot stance detection via contrastive learning. In *WWW '22: The ACM Web Conference 2022, Virtual Event, Lyon, France, April 25 - 29, 2022*, pages 2738–2747. ACM.

Saif Mohammad, Svetlana Kiritchenko, Parinaz Sobhani, Xiaodan Zhu, and Colin Cherry. 2016a. SemEval-2016 task 6: Detecting stance in tweets. In *Proceedings of the 10th International Workshop on Semantic Evaluation (SemEval-2016)*, pages 31–41, San Diego, California. Association for Computational Linguistics.

Saif M. Mohammad, Svetlana Kiritchenko, Parinaz Sobhani, Xiao-Dan Zhu, and Colin Cherry. 2016b. Semeval-2016 task 6: Detecting stance in tweets. In *Proceedings of the 10th International Workshop on Semantic Evaluation, SemEval@NAACL-HLT 2016, San Diego, CA, USA, June 16-17, 2016*, pages 31–41. The Association for Computer Linguistics.

Mamshad Nayeem Rizve, Kevin Duarte, Yogesh Singh Rawat, and Mubarak Shah. 2021. In defense of pseudo-labeling: An uncertainty-aware pseudo-label selection framework for semi-supervised learning. In *9th International Conference on Learning Representations, ICLR 2021, Virtual Event, Austria, May 3-7, 2021*. OpenReview.net.

Timo Schick and Hinrich Schütze. 2021. Exploiting cloze-questions for few-shot text classification and natural language inference. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume, EACL 2021, Online, April 19 - 23, 2021*, pages 255–269. Association for Computational Linguistics.

Umme Aymun Siddiqua, Abu Nowshed Chy, and Masaki Aono. 2019. Tweet stance detection using an attention based neural ensemble model. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT 2019, Minneapolis, MN, USA, June 2-7, 2019, Volume 1 (Long and Short Papers)*, pages 1868–1873. Association for Computational Linguistics.

Jake Snell, Kevin Swersky, and Richard S. Zemel. 2017. Prototypical networks for few-shot learning. In *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, December 4-9, 2017, Long Beach, CA, USA*, pages 4077–4087.

Kihyuk Sohn, David Berthelot, Nicholas Carlini, Zizhao Zhang, Han Zhang, Colin Raffel, Ekin Dogus Cubuk, Alexey Kurakin, and Chun-Liang Li. 2020. Fixmatch: Simplifying semi-supervised learning with consistency and confidence. In *Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020, December 6-12, 2020, virtual*.

Jason W. Wei and Kai Zou. 2019. EDA: easy data augmentation techniques for boosting performance on text classification tasks. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing, EMNLP-IJCNLP 2019, Hong Kong, China, November 3-7, 2019*, pages 6381–6387. Association for Computational Linguistics.

Penghui Wei and Wenji Mao. 2019. Modeling transferable topics for cross-target stance detection. In *Proceedings of the 42nd International ACM SIGIR Conference on Research and Development in Information Retrieval, SIGIR 2019, Paris, France, July 21-25, 2019*, pages 1173–1176. ACM.

Penghui Wei, Wenji Mao, and Guandan Chen. 2019. A topic-aware reinforced model for weakly supervised stance detection. In *The Thirty-Third AAAI Conference on Artificial Intelligence, AAAI 2019, The Thirty-First Innovative Applications of Artificial Intelligence Conference, IAAI 2019, The Ninth AAAI Symposium on Educational Advances in Artificial Intelligence, EAAI 2019, Honolulu, Hawaii, USA, January 27 - February 1, 2019*, pages 7249–7256. AAAI Press.

Qizhe Xie, Zihang Dai, Eduard H. Hovy, Thang Luong, and Quoc Le. 2020a. Unsupervised data augmentation for consistency training. In *Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020, December 6-12, 2020, virtual*.

Qizhe Xie, Minh-Thang Luong, Eduard H. Hovy, and Quoc V. Le. 2020b. Self-training with noisy student improves imagenet classification. In *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition, CVPR 2020, Seattle, WA, USA, June 13-19, 2020*, pages 10684–10695. Computer Vision Foundation / IEEE.

Chang Xu, Cécile Paris, Surya Nepal, and Ross Sparks. 2018. Cross-target stance classification with self-attention networks. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics, ACL 2018, Melbourne, Australia, July 15-20, 2018, Volume 2: Short Papers*, pages 778–783. Association for Computational Linguistics.

Bowen Zhang, Min Yang, Xutao Li, Yunming Ye, Xiaofei Xu, and Kuai Dai. 2020a. Enhancing cross-target stance detection with transferable semantic-emotion knowledge. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, Online. Association for Computational Linguistics.

Bowen Zhang, Min Yang, Xutao Li, Yunming Ye, Xiaofei Xu, and Kuai Dai. 2020b. Enhancing cross-target stance detection with transferable semantic-emotion knowledge. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, ACL 2020, Online, July 5-10, 2020*, pages 3188–3197. Association for Computational Linguistics.

A Hyper-parameters setting

As few-shot learning is extremely sensitive to hyper-parameters, we conduct a grid hyper-parameters search based on the performance on the validation data for learning rate lr $\in \{2\text{e-}5, 3\text{e-}5\}$, temperature $\tau \in \{0.05, 0.1, 0.2, 0.3, 0.4\}$ and scalar weighting $\lambda_b \in \{0.05, 0.1, 0.2, 0.5\}$ on two different datasets across different shot sizes. Here, we give the hyperparameter settings of our model in Table 4 and Table 5. Experiments are conducted on NVIDIA RTX TITAN GPUs.

Model	5	10	20
BERT	3e-5	3e-5	2e-5
SCL	(3e-5,0.5,0.2)	(3e-5,0.5,0.3)	(3e-5,0.05, 0.3)
TCL	(3e-5,0.2,0.05)	(3e-5,0.2,0.05)	(3e-5,0.2,0.05)
STCC	(3e-5,0.5,0.2)	(2e-5,0.5,0.2)	(3e-5,0.5,0.4)

Table 4: Hyper-parameter configurations for SemEval-2016. In parentheses from left to right are learning rate, scalar weighting λ_b, and temperature τ.

Model	5	10	20
BERT	3e-5	3e-5	3e-5
SCL	(2e-5,0.5,0.3)	(3e-5,0.5,0.4)	(3e-5,0.5,0.4)
TCL	(2e-5,0.5,0.2)	(2e-5,0.5,0.3)	(3e-5,0.5,0.3)
STCC	(2e-5,0.5,0.3)	(3e-5,0.5,0.1)	(2e-5,0.2,0.3)

Table 5: Hyper-parameter configurations for COVID-19. In parentheses from left to right are learning rate, scalar weighting λ_b, and temperature τ.

Joint Alignment of Multi-Task Feature and Label Spaces for Emotion Cause Pair Extraction

Shunjie Chen[1], **Xiaochuan Shi**[1], **Jingye Li**[1], **Shengqiong Wu**[2], **Hao Fei**[2], **Fei Li**[1†], **Donghong Ji**[1]
[1]Key Laboratory of Aerospace Information Security and Trusted Computing, Ministry of Education, School of Cyber Science and Engineering, Wuhan University, Wuhan, China
[2]School of Computing, National University of Singapore, Singapore
{shunjiechen, shixiaochuan, theodorelee, lifei_csnlp, dhji}@whu.edu.cn
swu@u.nus.edu, haofei37@nus.edu.sg

Abstract

Emotion cause pair extraction (ECPE), as one of the derived subtasks of emotion cause analysis (ECA), shares rich inter-related features with emotion extraction (EE) and cause extraction (CE). Therefore EE and CE are frequently utilized as auxiliary tasks for better feature learning, modeled via multi-task learning (MTL) framework by prior works to achieve state-of-the-art (SoTA) ECPE results. However, existing MTL-based methods either fail to simultaneously model the specific features and the interactive feature in between, or suffer from the inconsistency of label prediction. In this work, we consider addressing the above challenges for improving ECPE by performing two alignment mechanisms with a novel A^2Net model. We first propose a feature-task alignment to explicitly model the specific emotion-&cause-specific features and the shared interactive feature. Besides, an inter-task alignment is implemented, in which the label distance between the ECPE and the combinations of EE&CE are learned to be narrowed for better label consistency. Evaluations of benchmarks show that our methods outperform current best-performing systems on all ECA subtasks. Further analysis proves the importance of our proposed alignment mechanisms for the task.[1]

1 Introduction

Emotion cause analysis (ECA), detecting potential causes for certain emotion expressions in a document, has been a hot research topic in natural language processing (NLP) community (Lee et al., 2010; Gui et al., 2016; Fan et al., 2019). ECA has derived three associated tasks: EE, CE and ECPE. As illustrated in Figure 1(a), EE and CE detects the emotion and cause clauses respectively, while ECPE identifies both the emotion and cause clauses as well as their semantic relation. By jointly modeling the clauses detection and the relational pairing, ECPE effectively relieves the noise introduction in the pipeline process, and thus receives most research attention recently (Ding et al., 2020a; Wei et al., 2020; Wu et al., 2022).

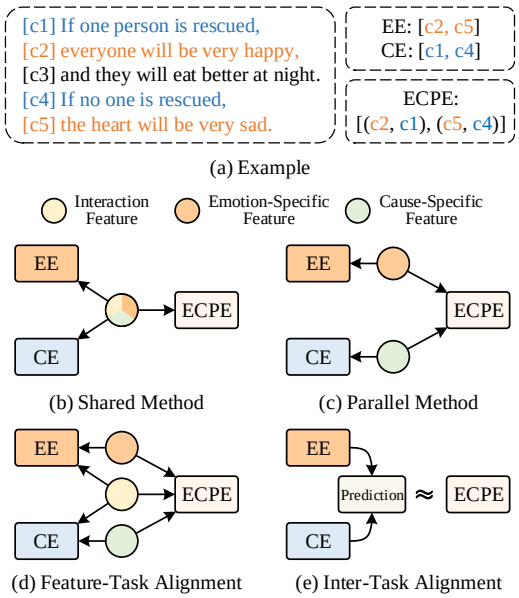

Figure 1: (a) illustrates three subtasks of ECA. (b) and (c) depicts the shared and parallel features encoding method, respectively. In (d) and (e) we show our proposed feature-task alignment mechanism and inter-task alignment mechanism, respectively.

As there are close correlations among EE, CE and ECPE, existing ECPE works extensively treat EE and CE as auxiliary tasks for additional feature supports, and mostly adopt the multi-task learning framework to explicitly model the interdependency in between, thus achieving current SoTA performances (Wei et al., 2020; Ding et al., 2020a; Fan et al., 2021; Bao et al., 2022). From the view of feature encoding, there are two major categories of MTL-based ECPE methods: *shared feature encoding* method and *parallel feature encoding* method. As shown in Figure 1(b), shared

†Corresponding author
[1]Our code is available at https://github.com/csj199813/A2Net_ECPE

Proceedings of the 29th International Conference on Computational Linguistics, pages 6955–6965
October 12–17, 2022.

methods only learn mixed features via one encoder without distinguishing specific features for individual subtasks (Wei et al., 2020; Yuan et al., 2020). In contrast, parallel methods (Ding et al., 2020a,b; Fan et al., 2021) use two encoders to learn emotion- and cause-specific features separate (cf. Figure 1(c)), where unfortunately, the interaction among these tasks are overlooked. We argue that both the private emotion-specific and cause-specific features and the shared interactive feature are important to the final performance, which should be explicitly modeled in the MTL framework. To this end, we in this work propose a *feature-task alignment* (FTA) scheme of MTL for ECPE (cf. Figure 1(d)), in which we explicitly split three parts of features, and align them to EE, CE and ECPE respectively.

Meanwhile, aligning the label space in a MTL framework is crucial to overall ECPE performance, because intuitively all modules in the MTL process should reach a consensus. For example as in Figure 1(a), once "c1" is recognized as a cause clause by ECPE module, it should not be further predicted as a non-cause clause by CE module. We notice that such label consistency is not guaranteed in existing MTL-based ECPE methods, which could inevitably hurt the prediction. Therefore we further introduce a *inter-task alignment* (ITA) mechanism (cf. 1(e)) that learns to pull closer the label distance between the ECPE and the combinations of EE&CE, ensuring label consistency.

We implement the above ideas of feature-task alignment and inter-task alignment by developing a novel neural network, namely A^{2}Net, as shown in Figure 2. First, we employ the BERT (Devlin et al., 2019) as document encoder for producing clauses representations. We then leverage the partition filter network (PFN) (Yan et al., 2021b) to implement the feature-task alignment, generating emotion-specific features, cause-specific features and interaction features, respectively. Afterwards, we apply emotion-specific and interaction features for EE, cause-specific and interaction features for CE, and all features for ECPE. Finally, we reach the goal of inter-task alignment by minimizing the bidirectional KL-divergence between the output distributions of ECPE and EE$\times$CE, thus maintaining the consistency of label spaces among all tasks.

Our A^{2}Net framework is evaluated on the ECA benchmark (Xia and Ding, 2019), where our system achieves new SoTA results on EE, CE and ECPE. Further analyses demonstrate that our method learns better consistency in the predictions of all subtasks than existing baselines. Overall, this work contributes to three major aspects:

- We present an innovative multi-task learning based ECPE framework, where we further propose a feature-task alignment mechanism that can make better use of the shared features from EE and CE sources.
- We also introduce an inter-task alignment mechanism to reduce the inconsistency between the prediction results of ECPE and the EE&CE, significantly enhancing the performance as well as the robustness of the system.
- Our system empirically achieves new SoTA performances of the EE, CE and ECPE tasks on the benchmark.

2 Related Work

In NLP area, the analysis on sentiment and opinion is a long-standing research topic (Liu, 2012; Li et al., 2020; Wu et al., 2021; Fei et al., 2022a), including detecting of the sentiment polarities (Tang et al., 2016; Fei et al., 2022b; Shi et al., 2022) and the emotion categories (Lee et al., 2010; Neviarouskaya and Aono, 2013). One of the recent trend on the emotion detection has been upgraded to the emotion cause analysis (ECA). Centered on the topic of ECA, there are several subordinated tasks according to the extracting elements of emotion and cause, such as emotion cause extraction (ECE) (Lee et al., 2010; Neviarouskaya and Aono, 2013; Gui et al., 2016; Li et al., 2018) and ECPE (Xia and Ding, 2019; Wei et al., 2020; Bao et al., 2022).

Lee et al. (2010) pioneer the ECE task, in which the task is formulated as a word-level cause labeling problem. Following this work, initial research constructs rule-based methods (Neviarouskaya and Aono, 2013; Gao et al., 2015) and machine learning methods (Ghazi et al., 2015; Song and Meng, 2015) on their own corpus. Deep learning based methods greatly facilitate the line of this research (Fei et al., 2021a, 2022e,c, 2021b; Wei et al., 2019a). Recently, Gui et al. (2016) release a public corpus and re-formalize ECE as a clause-level classification problem, where the goal is to detect cause clauses for a given emotion in the text. The framework has received much attention in recent years and the corpus has become a benchmark ECA dataset (Gui et al., 2017; Li et al., 2018; Fan et al., 2019; Ding

et al., 2019; Hu et al., 2021b; Yan et al., 2021a; Hu et al., 2021a).

However, as Xia and Ding (2019) points out, the ECE task is limited to the task definition, i.e., emotion needs to be manually marked in advance. They thus introduce the ECPE task that simultaneously extract both the emotion and cause clauses as well as determining their relations, which has a better utility in real-world applications (Fei et al., 2022d, 2020). Thereafter, a line of subsequent research efforts are paid to ECPE within the last years (Ding et al., 2020b; Cheng et al., 2020; Fan et al., 2021; Bao et al., 2022).

Recent ECPE methods mostly employ the multi-task learning for simultaneously modeling the EE and CE as auxiliary tasks for making use of the shared features, and thus realize SoTA ECPE performances (Bao et al., 2022). Existing MTL-based ECPE works can largely be divided into two categories: parallel encoding and shared encoding methods. *Parallel methods* mostly learn the emotion/cause feature representations in mutually independent ways (Cheng et al., 2020; Ding et al., 2020a; Fan et al., 2021; Bao et al., 2022). Ding et al. (2020b); Chen et al. (2022) uses auxiliary task prediction to aid the interactions between emotion and cause features. However, the prediction values are limited to only a two-dimensional vector, leading to insufficient interaction between emotion and cause features.

Shared methods learn the mixed features by only one encoder without distinguishing between features of different tasks (Wei et al., 2020; Yuan et al., 2020; Wei et al., 2019b,c). In this work, we use the PFN (Yan et al., 2021b) to generate emotion-specific features, cause-specific features and interaction features and implement the feature-task alignment. Moreover, we use an inter-task alignment module to reduce the gap between EE, CE and ECPE, maintaining the consistency of the label space.

3 Methodology

Task Formulation Given a document consisting of N clauses $\mathcal{D} = \{c_1, c_2, \ldots, c_N\}$, and each c_i denotes a subsection of a sentence separated by a comma. The goals of EE and CE task are extracting emotion clauses $c_i^e \in \mathcal{D}$ and cause clauses $c_j^c \in \mathcal{D}$ in the document, respectively, while ECPE task identifies the emotion-cause clause pair (c_i^e, c_j^c) that has causal relationship between emotion and cause clauses.

As illustrated in Figure 2, the overall architecture of our A^{2}Net consists of four tiers, including the encoder layer, feature-task alignment layer, prediction layer and inter-task alignment mechanism. First, following the previous work and we use BERT (Devlin et al., 2019) as encoder to yield contextualized clause representations from input documents. Then, to explicitly model task-specific features and task-shared features. We leverage a PFN (Yan et al., 2021b) to capture emotion- and cause-specific features and the interaction between them. Afterward, a prediction layer is used to align three combinations of three kinds of features from PFN with three tasks, and predict the emotion clauses, cause clauses and emotion-cause pair for EE, CE and ECPE, respectively. Finally, considering that there should be a consensus among all tasks. We propose an inter-task alignment mechanism to enhance the consistency between ECPE and EE$\times$CE.

3.1 Encoder Layer

Following (Wei et al., 2020), we also leverage pre-trained BERT language model (Devlin et al., 2019) as the underlying encoder to yield contextualized clause representations. Concretely, we insert a `[CLS]` token at the beginning of each clause and append a `[SEP]` token to the end, i.e., $c_i = \{[CLS], w_{i,1}, w_{i,2}, ..., w_{i,M}, [SEP]\}$. Then we concatenate them together as the input of BERT to generate contextualized token representations, in which we take the representation of `[CLS]` token in each clause c_i as its clause representation. After that, we obtain all the clause representations $\boldsymbol{X} = \{\boldsymbol{x}_1, \boldsymbol{x}_2, \ldots, \boldsymbol{x}_N\}$.

3.2 Feature-task Alignment Layer

We adopt partition filter network (PFN) (Yan et al., 2021b) to capture emotion- and cause-specific features and the interaction between them because of its powerful ability to extract task-specific features and interaction features. PFN is similar to the LSTM structure and has two task-related gates: the emotion gate and the cause gate. The gates filter features according to their contribution to each task with emotion and cause gates. In each time step, the encoder divides clause representation into three feature partitions: emotion partition, cause partition, and interaction partition, where interaction partition represents information useful to all tasks.

Specifically, at the *i* time step, we first generate

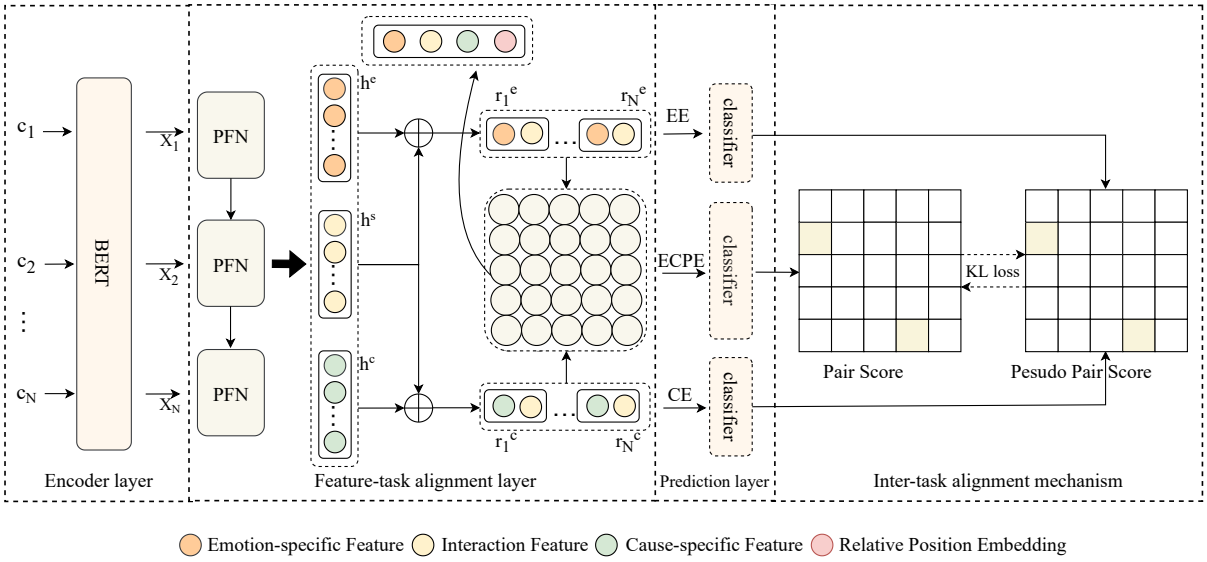

Figure 2: Overview of our A^{2}Net model.

two task-related gates:

$$\begin{aligned} \boldsymbol{g}_i^e &= \text{Cummax}(\text{Linear}([\boldsymbol{x}_i; \boldsymbol{h}_{i-1}]))\,, \\ \boldsymbol{g}_i^c &= 1 - \text{Cummax}(\text{Linear}([\boldsymbol{x}_i; \boldsymbol{h}_{i-1}]))\,, \end{aligned} \tag{1}$$

where $\text{Cummax}(\cdot) = \text{Cumsum}(\text{Softmax}(\cdot))$, performs as a binary gate, and $\text{Linear}(\cdot)$ denotes linear transformation, and $\boldsymbol{h}_{i-1}$ is the hidden state of i-1-th clause. Each gate will divide clause representations into two segments: task-related and task-unrelated, according to their usefulness to the specific task. With the joint efforts of the two gates, the clause representation can be divided into three partitions: emotion partition $\boldsymbol{p}_i^e$, cause partition $\boldsymbol{p}_i^c$ and interaction partition $\boldsymbol{p}_i^s$. We use task-related gates ($\boldsymbol{g}_i^c$ and $\boldsymbol{g}_i^e$) to calculate forgetting gates:

$$\begin{aligned} \boldsymbol{f}_i^s &= \boldsymbol{g}_i^e \circ \boldsymbol{g}_i^c\,, \\ \boldsymbol{f}_i^e &= \boldsymbol{g}_i^e - \boldsymbol{f}_i^s\,, \\ \boldsymbol{f}_i^c &= \boldsymbol{g}_i^c - \boldsymbol{f}_i^s\,, \end{aligned} \tag{2}$$

where $\circ$ denotes element-wise multiplication, $\boldsymbol{f}_i^e$, $\boldsymbol{f}_i^c$ and $\boldsymbol{f}_i^s$ are emotion, cause and interaction forgetting gates, respectively. Similarly, we also can obtain input gates $\boldsymbol{o}_i^e$, $\boldsymbol{o}_i^c$ and $\boldsymbol{o}_i^s$ via Equation 1 and 2.

After that, we use forgetting and input gates to control the flow of input and history information:

$$\begin{aligned} \widetilde{\boldsymbol{c}}_i &= \tanh(\text{Linear}([\boldsymbol{x}_i; \boldsymbol{h}_{i-1}]))\,, \\ \boldsymbol{p}_i^s &= \boldsymbol{f}_i^s \circ \boldsymbol{c}_{i-1} + \boldsymbol{o}_i^s \circ \widetilde{\boldsymbol{c}}_i\,, \\ \boldsymbol{p}_i^e &= \boldsymbol{f}_i^e \circ \boldsymbol{c}_{i-1} + \boldsymbol{o}_i^e \circ \widetilde{\boldsymbol{c}}_i\,, \\ \boldsymbol{p}_i^c &= \boldsymbol{f}_i^c \circ \boldsymbol{c}_{i-1} + \boldsymbol{o}_i^c \circ \widetilde{\boldsymbol{c}}_i\,, \end{aligned} \tag{3}$$

where $\widetilde{\boldsymbol{c}}_i$ denotes the current input information, and $\boldsymbol{c}_{i-1}$ denotes the history information.

Next, we can obtain three feature representations: emotion feature $\boldsymbol{h}_i^e$, cause feature $\boldsymbol{h}_i^c$ and inter-task interaction feature $\boldsymbol{h}_i^s$ from the partition:

$$\begin{aligned} \boldsymbol{h}_i^s &= \tanh(\boldsymbol{p}_i^s)\,, \\ \boldsymbol{h}_i^e &= \tanh(\boldsymbol{p}_i^e)\,, \\ \boldsymbol{h}_i^c &= \tanh(\boldsymbol{p}_i^c)\,. \end{aligned} \tag{4}$$

We further use the information in all three partitions to construct cell state $\boldsymbol{c}_i$, and hidden state $\boldsymbol{h}_i$ for the next time step:

$$\begin{aligned} \boldsymbol{c}_i &= \text{Linear}([\boldsymbol{p}_i^e; \boldsymbol{p}_i^s; \boldsymbol{p}_i^c])\,, \\ \boldsymbol{h}_i &= \tanh(\boldsymbol{c}_i)\,. \end{aligned} \tag{5}$$

After PFN, we can obtain the emotion-specific features $\boldsymbol{h}_i^e$, cause-specific features $\boldsymbol{h}_i^c$ and the interaction features $\boldsymbol{h}_i^s$. First, we align features with the EE and CE tasks, in which we concatenate the features of emotion and cause with the interaction features separately and gain the emotion representations $\boldsymbol{r}_i^e = [\boldsymbol{h}_i^e; \boldsymbol{h}_i^s]$ and the cause representations $\boldsymbol{r}_i^c = [\boldsymbol{h}_i^c; \boldsymbol{h}_i^s]$. Moreover, we consider aligning features with ECPE task, so we add task-specific features and interaction features to aggregate all the information about ECPE task:

$$\begin{aligned} \boldsymbol{h}_i^{e'} &= \boldsymbol{h}_i^e + \boldsymbol{h}_i^s\,, \\ \boldsymbol{h}_j^{c'} &= \boldsymbol{h}_j^c + \boldsymbol{h}_j^s\,, \\ \boldsymbol{r}_{ij} &= [\boldsymbol{h}_i^{e'}; \boldsymbol{h}_j^{c'}; \boldsymbol{e}_{ij}]\,, \end{aligned} \tag{6}$$

where $\boldsymbol{e}_{ij}$ denotes the relative position embedding following Wei et al. (2020). $\boldsymbol{r}_{ij}$ is the final emotion-cause pair representation.

3.3 Prediction Layer

Extracting Emotion/Cause We feed $\boldsymbol{r}_i^e$ and $\boldsymbol{r}_i^c$ into two feedforward network (FFN) to obtain emotion prediction $\hat{y}_i^e$ and cause prediction $\hat{y}_i^c$ for the i-th clause:

$$\begin{aligned} \hat{y}_i^e &= \sigma(\text{FFN}(\boldsymbol{r}_i^e)) \,, \\ \hat{y}_i^c &= \sigma(\text{FFN}(\boldsymbol{r}_i^c)) \,, \end{aligned} \tag{7}$$

where $\sigma(\cdot)$ means the sigmoid function.

The auxiliary task loss for emotion prediction and cause prediction can be formulated as:

$$\mathcal{L}_{aux} = -\sum_{i=1}^{N} (y_i^e \log(\hat{y}_i^e) + y_i^c \log(\hat{y}_i^c)) \,, \tag{8}$$

where y_i^e and y_i^c are emotion and cause ground truth labels of clause c_i, respectively.

Extracting Emotion Cause Pair We employ a FFN with a sigmoid function to obtain the emotion-cause pair score $\hat{y}_{ij}^p$:

$$\hat{y}_{ij}^p = \sigma(\text{FFN}(\boldsymbol{r}_{ij})) \,. \tag{9}$$

The loss function of emotion-cause pair extraction can be formalized as:

$$\mathcal{L}_{pair} = -\sum_{i=1}^{N}\sum_{j=1}^{N} y_{ij}^p \log(\hat{y}_{ij}^p) \,, \tag{10}$$

where y_{ij}^p is the ground truth label of the clause pair (c_i, c_j).

3.4 Inter-task Alignment Mechanism

As we argued earlier, the predictions of EE and CE could be inconsistent with ECPE, i.e., the emotion in the emotion-cause pair predicted by ECPE could not be detected by EE, which hinders the task for further improvements. To address this issue, we propose an inter-task alignment (ITA) mechanism to constrain the predicted scores between ECPE and auxiliary tasks during the training period. First, we leverage the emotion score $\hat{y}_i^e$ and the cause score $\hat{y}_j^c$ to get the pre-pseudo pair score $\sqrt{\hat{y}_i^e \hat{y}_j^c}$. Note that there could not exist a causal relationship in pairs matched from predictions of EE and CE. Thus, as shown in Figure 3, we calculate the pseudo emotion-cause pair score $\tilde{y}_{ij}^p$ as follows:

$$\tilde{y}_{ij}^p = \alpha_{ij} \sqrt{\hat{y}_i^e \hat{y}_j^c} \,, \tag{11}$$

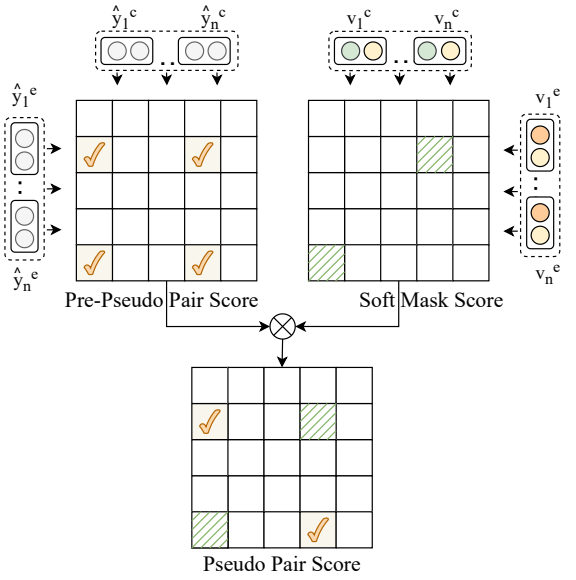

Figure 3: The generation of pseudo pair score, where ✓ denotes candidate emotion-cause pairs, green grids represent the masked pair.

where α_{ij} $(0 \leq \alpha_{ij} \leq 1)$ is a soft mask score for the pseudo pair $(\hat{y}_i^e, \hat{y}_j^c)$, which can reduce the score of fake emotion-cause pairs in pre-pseudo pairs score. α_{ij} is computed by:

$$\begin{aligned} \boldsymbol{t}_{ij} &= \frac{(\boldsymbol{v}_i^e)^\top \boldsymbol{v}_j^c}{\sqrt{d}} \,, \\ \alpha_{ij} &= \frac{\exp(\boldsymbol{t}_{ij})}{\sum_j^N \exp(\boldsymbol{t}_{ij})} \,, \end{aligned} \tag{12}$$

where $\boldsymbol{v}_i^e$ and $\boldsymbol{v}_j^c$ are obtained from $\boldsymbol{r}_i^e = [\boldsymbol{h}_i^e; \boldsymbol{h}_i^s]$ and $\boldsymbol{r}_j^c = [\boldsymbol{h}_j^c; \boldsymbol{h}_j^s]$ with FFNs, respectively. The d denotes the dimension of $\boldsymbol{v}_i^e$ and $\boldsymbol{v}_j^c$.

Then we reduce the gap between the pseudo emotion-cause pair score $\tilde{y}_{ij}^p$ from EE and CE and the true emotion-cause pair score $\hat{y}_{ij}^p$ from ECPE using Kullback Leibler (KL) Divergence:

$$\begin{aligned} \mathcal{L}_{KL} &= \frac{1}{2}\sum_{i=1}^{N}\sum_{j=1}^{N}(\text{KL}(\tilde{y}_{ij}^p || \hat{y}_{ij}^p) + \text{KL}(\hat{y}_{ij}^p || \tilde{y}_{ij}^p)) \\ &= \frac{1}{2}(\sum_{i=1}^{N}\sum_{j=1}^{N}(\tilde{y}_{ij}^p \log(\frac{\tilde{y}_{ij}^p}{\hat{y}_{ij}^p}) + \hat{y}_{ij}^p \log(\frac{\hat{y}_{ij}^p}{\tilde{y}_{ij}^p})) \,. \end{aligned} \tag{13}$$

Optimization The final loss of our model is a weighted sum of $\mathcal{L}_{pair}$, $\mathcal{L}_{aux}$ and $\mathcal{L}_{KL}$:

$$\mathcal{L} = \mathcal{L}_{pair} + \lambda_1 \mathcal{L}_{aux} + \lambda_2 \mathcal{L}_{KL} \,, \tag{14}$$

where λ_1 and λ_2 are hyperparameters.

Approach	ECPE			EE			CE		
	P	R	F1	P	R	F1	P	R	F1
ANTS	72.43	63.66	67.76	81.96	73.29	77.39	74.90	66.02	70.18
TransECPE	73.74	63.07	67.99	87.16	82.44	84.74	75.62	64.71	69.74
ECPE-2D	72.92	65.44	68.89	86.27	**92.21**	89.10	73.36	69.34	71.23
PairGCN	76.92	67.91	72.02	88.57	79.58	83.75	79.07	69.28	73.75
RANKCP	71.19	76.30	73.60	91.23	89.99	90.57	74.61	77.88	76.15
ECPE-MLL	77.00	72.35	74.52	86.08	91.91	88.86	73.82	79.12	76.30
MGSAG	**77.43**	73.21	75.21	**92.08**	82.11	87.17	**79.79**	74.68	77.12
A^{2}Net(ours)	75.03	**77.80**	**76.34**	90.67	90.98	**90.80**	77.62	**79.20**	**78.35**

Table 1: Comparisons with baselines on Chinese benchmark ECPE corpus. For a fair comparison, they all use BERT as the encoder. The best performance is in **bold** and the second best performance is underlined.

4 Experiments Settings

4.1 Dataset and Metrics

We conducted experiments on the Chinese benchmark dataset provided by Xia and Ding (2019) to evaluate the effectiveness of our proposed model A^{2}Net. For fair comparisons, following previous work we use 10-fold cross-validation as the data split strategy and the precision (P), recall (R), and F1 score (F1) as evaluation metrics. Meanwhile, we also verify the performance of two auxiliary tasks: emotion extraction (EE) and cause extraction (CE), using the same evaluation metrics as ECPE.

4.2 Implementation Details

We apply PyTorch to implement our framework.[2] We leverage pre-trained language model BERT (Devlin et al., 2019) as our embedding layer.[3] We employ one-layer PFN (Yan et al., 2021b) with hidden size of 300. Besides, the hyperparameters λ_1 and λ_2 are both set to 0.4. We set the batch size and the learning rate to 4 and 2e-5, respectively. We apply AdamW (Loshchilov and Hutter, 2017) to optimize our model parameters. To prevent overfitting, the dropout rate is set to 0.1.

4.3 Baselines

In order to verify the effectiveness of our proposed model A^{2}Net, we compared it with the following strong methods. For a fair comparison, they all use BERT as the encoder.

- **ANTS** (Yuan et al., 2020): ANTS solves ECPE with a sequence labeling approach and proposes a tagging scheme.
- **TransECPE** (Fan et al., 2020): TransECPE is a transition-based framework that converts ECPE into a parsing-like directed graph construction task.
- **ECPE-2D** (Ding et al., 2020a): ECPE-2D leverages clauses pairs to construct a 2D representation matrix which integrated with auxiliary task predictions for ECPE task.
- **PairGCN** (Chen et al., 2020): This method models the dependency relations among clause pairs with graph convolution networks.
- **RANKCP** (Wei et al., 2020): RANKCP tackles the ECPE task from a ranking perspective and uses graph attention to model the inter-clause relations.
- **ECPE-MLL** (Ding et al., 2020b): ECPE-MLL converts the ECPE task into the emotion-pivot cause extraction and the cause-pivot emotion extraction using the sliding window strategy.
- **MGSAG** (Bao et al., 2022): MGSAG constructs a multi-granularity semantic aware graph to deal with ECPE task, and it is the current SoTA approach.

5 Experimental Results

5.1 Main Results

Table 1 shows the comparison results of our model (A^{2}Net) with the strong baselines on the emotion-cause pair extraction (ECPE) task and two auxiliary tasks: emotion extraction (EE) and cause extraction (CE).

[2]https://pytorch.org

[3]The version of BERT is bert-base-chinese.

	ECPE	EE	CE
A^{2}Net (ours)	**76.34**	**90.80**	**78.35**
w/ Shared encoding	69.97	84.81	72.66
w/ Parallel encoding	75.59	89.75	78.03

Table 2: Performances (F1) with different feature encoding schemes.

	ECPE	EE	CE
A^{2}Net (ours)	**76.34**	**90.80**	**78.35**
w/o ECPE→EE×CE	75.83	90.65	78.05
w/o EE×CE→ECPE	75.50	90.54	77.60
w/o ITA	75.32	90.05	77.37
w/o EE & CE	74.39	-	-

Table 3: Ablation study of inter-task alignment module and auxiliary task (F1). The ECPE→EE×CE means we use the prediction distribution of ECPE to align to EE×CE (i.e., KL($\tilde{y}^p_{ij}||\hat{y}^p_{ij}$) in Eq.13), and vice versa.

For the ECPE task, it is clear that our model A^{2}Net achieves 1.13% and 1.82% F1-score improvement over MGSAG (the current best method) and ECPE-MLL, respectively. Further analysis, we can find that the above advantage mainly comes from the improvement of the recall. Compared with MGSAG and ECPE-MLL, our recall is increased by 4.59% and 5.45% respectively, which indicates that consistent prediction on the three tasks allows the A^{2}Net model to detect more emotion-cause pairs under a considerable precision.

For auxiliary tasks, on the EE task, our model achieves 3.63% and 1.94% F1 improvement over MGSAG and ECPE-MLL, and 0.23% F1 improvement over the previous best model RANKCP. On the CE task, our model yields a great improvement of F1 scores by 1.23% in comparison with the top-performing baseline MGSAG, and achieves 2.20% and 2.05% F1 improvement over RANKCP and ECPE-MLL.

We argue that all improvements come mainly from our proposed feature-task alignment module and inter-task alignment module. Both alignment mechanisms are able to collaboratively improve the performances of all tasks, and enhance the robustness of the model. In the following part we performed corresponding experiments to verify our ideas.

5.2 Effect of Feature-task Alignment

The feature-task alignment module is capable of generating efficiency and independent task-specific and interactive features. To verify the effectiveness of our feature-task alignment, we replaced the PFN with two encoding schemes: shared encoding and parallel encoding. The results are shown in Table 2.

In terms of the shared encoding, we encode the emotion features and cause features using a shared BiLSTM, in which emotion and cause features are entangled. For parallel encoding, we utilize two BiLSTMs to capture emotion features and cause features separately, in which interaction information among different tasks is not considered. Firstly, we observe that the model with parallel encoding significantly outperforms the shared encoding among three tasks, indicating that it is important for the model to consider task-specific features. Furthermore, we can see that our model enjoys better performances when we consider both task-specific features and shared interaction features, compared with the parallel encoding. This shows the necessity of aligning feature spaces for different tasks.

5.3 Effect of Inter-task Alignment

In this section, we investigate the effect of the inter-task alignment (ITA) mechanism and auxiliary tasks for A^{2}Net, and the results are plotted in Table 3.

We first analyze the effect of the aligned direction of the inter-task alignment mechanism. When we merely apply unidirectional alignment to regulate the predictions between ECPE and two auxiliary tasks, we can observe slight performance drops on three tasks to some extent. Furthermore, after removing the inter-task alignment mechanism (bidirectional alignment), we find that the overall decreases in F1 score on three tasks happen, and are more than the any unidirectional alignment, which verifies the helpfulness of the alignment in label spaces among tasks, and bi-direction of alignment are more important for our model.

Besides, we also explore the effectiveness of auxiliary tasks, EE and CE. It should be noted that inter-task alignment module does not work after the auxiliary tasks are removed. When the auxiliary tasks are further removed, we can see that the model performances drop significantly, demonstrating that the auxiliary task can effectively contribute to the ECPE task.

Document	A^{2}Net(w/o ITA)	A^{2}Net	Ground-truth
... The police visited the villagers of Nanyuan Village (c3), and they learned that Meng was playing mahjong at a mahjong parlor opposite his home the day before the incident (c4), through inquiries (c5), it wa--s found that only Wang from the same village had gone out to an unk-nown destination(c6), which aroused the suspicion of the police (c7).	ECPE:[c7, c6] EE:[] CE:[c6]	ECPE:[c7, c6] EE:[c7] CE:[c6]	ECPE:[c7, c6] EE:[c7] CE:[c6]
On March 14 (c1), a magnitude 4.3 earthquake occurred in Yingquan District, Fuyang City, Anhui (c2). Then (c3), a rumor of a magnitude 6.8 earthquake occurred in Fuyang City at 2:15 am on March 15. (c4), which caused people to panic (c5)...	ECPE:[c5, c4] EE:[c5] CE:[c2], [c4]	ECPE:[c5, c4] EE:[c5] CE:[c4]	ECPE:[c5, c4] EE:[c5] CE:[c4]
Mr. Feng said frankly (c1), Jingjing is naughty on weekdays (c2), and sometimes he is not polite (c3), but when it comes to the reason for th--is injury(c4), he can't hide his anger (c5), just because of my son Dra--nk other children's yogurt (c6). Teacher Xing lost her mind (c7), she was emotionally out of control (c8), then pulled the child out of the do--or (c9), the child was injured when the door was closed (c10)...	ECPE:[c5,c4],[c5,c6] EE:[c5] CE:[c4], [c6]	ECPE:[c5,c4] EE:[c5] CE:[c4]	ECPE:[c5,c4] EE:[c5] CE:[c4]

Table 4: Two examples for the case study. The words in orange are the emotion clause, and the words in blue are the cause clause. The green means correct predictions, red means wrong predictions.

5.4 Analysis of Prediction Consistency Cross Tasks

In order to verify the effect of our proposed feature-task alignment module and inter-task alignment module on model prediction consistency among tasks, we conduct the experiments with multiple variants of A^{2}Net and the baseline RANKCP, as shown in Figure 4. Specifically, we disassemble the emotion-cause pairs into a set of emotion clauses and a set of cause clauses which are considered as gold labels for EE and CE, respectively. We calculated the consistency rate to evaluate the consistency of EE and CE tasks with ECPE task predictions by (EE & ECPE) / ECPE or (CE & ECPE) / ECPE, where EE, CE and ECPE denote the prediction results of corresponding tasks and & denotes the logic AND.

On the EE task, we can find that when our model A^{2}Net removes the FTA and ITA modules, the consistency rate drops significantly, but when our model only removes FTA or ITA, the consistency rate decreases slightly. Furthermore, all of them outperform RANKCP, which indicates that FTA and ITA have well aligned EE with ECPE.

On the CE task, we can find that a dramatic drop occurs when we remove the FTA and ITA modules. The consistency rate decreases slightly when only ITA is removed, while the consistency rate decreases more when only FTA is removed, indicating that our feature-level alignment is more effective for CE tasks. Moreover, all of them receive better consistency than RANKCP, indicating that both FTA and ITA are able to align CE and ECPE.

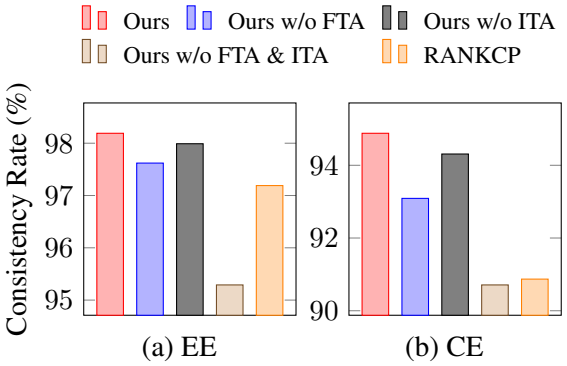

Figure 4: Consistency of ECPE and EE (a), as well as CE (b).

5.5 Case Analysis

Finally, to better understand the capacity of our proposed model, we empirically perform case study on EE, CE and ECPE tasks. Specifically, we demonstrate some predictions based on three instances randomly selected from testset, as shown in Table 4.

In the first example, our A^{2}Net without ITA correctly predicts the emotion cause pair (c_7, c_6) on the ECPE task and incorrectly on the EE and CE tasks. In contrast, our A^{2}Net model correctly predicts all ECPE, EE, and CE tasks after going through the inter-task alignment module. In the second example, A^{2}Net without ITA correctly predicts the emotion cause pair (c_5, c_4) on the ECPE task and incorrectly on the CE tasks. However, A^{2}Net model correctly predicts all EE, CE and

Model	#Param	Speed(doc/s)
RANKCP	105.97M	195
A^2Net (ours)	104.97M	195
w/o ITA	104.97M	195

Table 5: Parameter number and inference speed comparisons on ECPE. All models are tested with batch size 4.

ECPE tasks. In the third example, A^2Net without ITA predicts correctly in the EE task and incorrectly detects the cause clause c_6 in the CE task. Meanwhile, the emotion-cause pair (c_5, c_6) is predicted incorrectly. Nonetheless, after the inter-task alignment module, all predictions were correct in both the ECPE and CE tasks.

This shows that after aligning between tasks, the model can identify emotion-corresponding causes, which is like the role of our proposed soft mask score. We find that these cases are common in our dataset, which ultimately leads directly to an improvement in our model performance. This also demonstrates that our inter-task alignment module can normalize the inter-task training to make the model performs better and more stable.

5.6 Model Efficiency Analysis

In order to test the efficiency of our model, we conduct the experiments with multiple variants of A^2Net and the baseline RANKCP, and the results are plotted in Table 5.

In terms of parameter quantity, our A^2Net model is even less than RANKCP, and it can be found that since the ITA module does not introduce additional parameters, w/o ITA does not change the parameter quantity. In terms of inference speed, our model has the same inference speed as RANKCP, which shows that the efficiency of our model does not decrease due to the addition of the alignment mechanism.

6 Conclusion

Existing best-performing ECPE works extensively leverage EE and CE as auxiliary tasks for better feature learning via multi-task learning (MTL). In this paper, we further enhance the existing best-performing MTL-based ECPE by proposing feature-task alignment and inter-task alignment mechanisms. At the feature space, the feature-task alignment mechanism aligns the task-specific features and the shared interactive feature with corresponding tasks. At the label space, the inter-task alignment mechanism reduces the inconsistency among the predicted labels of EE, CE and ECPE. Experimental results on the benchmark ECPE data demonstrate the effectiveness of our methods. Further analysis shows that our system achieves better consistency than existing baselines, which explains the improvements of our model. The idea to align the feature space and label space in MTL framework is promising. In the future work, we consider further constructing intra-clause relations, inter-clause relations, and relations among pairs of clauses for ECPE.

Acknowledgements

This work is supported by the National Natural Science Foundation of China (No. 62176187), the National Key Research and Development Program of China (No. 2017YFC1200500), the Research Foundation of Ministry of Education of China (No. 18JZD015), the Youth Fund for Humanities and Social Science Research of Ministry of Education of China (No. 22YJCZH064), the General Project of Natural Science Foundation of Hubei Province (No.2021CFB385). This work is also the research result of the independent scientific research project of Wuhan University, supported by the Fundamental Research Funds for the Central Universities.

References

Yinan Bao, Qianwen Ma, Lingwei Wei, Wei Zhou, and Songlin Hu. 2022. Multi-granularity semantic aware graph model for reducing position bias in emotion cause pair extraction. In *Findings of the Association for Computational Linguistics: ACL 2022*, pages 1203–1213.

Fang Chen, Ziwei Shi, Zhongliang Yang, and Yongfeng Huang. 2022. Recurrent synchronization network for emotion-cause pair extraction. *Knowledge-Based Systems*, 238:107965.

Ying Chen, Wenjun Hou, Shoushan Li, Caicong Wu, and Xiaoqiang Zhang. 2020. End-to-end emotion-cause pair extraction with graph convolutional network. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 198–207.

Zifeng Cheng, Zhiwei Jiang, Yafeng Yin, Hua Yu, and Qing Gu. 2020. A symmetric local search network for emotion-cause pair extraction. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 139–149.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186.

Zixiang Ding, Huihui He, Mengran Zhang, and Rui Xia. 2019. From independent prediction to reordered prediction: Integrating relative position and global label information to emotion cause identification. In *Proceedings of the AAAI Conference on Artificial Intelligence*, pages 6343–6350.

Zixiang Ding, Rui Xia, and Jianfei Yu. 2020a. Ecpe-2d: emotion-cause pair extraction based on joint two-dimensional representation, interaction and prediction. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 3161–3170.

Zixiang Ding, Rui Xia, and Jianfei Yu. 2020b. End-to-end emotion-cause pair extraction based on sliding window multi-label learning. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 3574–3583.

Chuang Fan, Hongyu Yan, Jiachen Du, Lin Gui, Lidong Bing, Min Yang, Ruifeng Xu, and Ruibin Mao. 2019. A knowledge regularized hierarchical approach for emotion cause analysis. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 5614–5624.

Chuang Fan, Chaofa Yuan, Jiachen Du, Lin Gui, Min Yang, and Ruifeng Xu. 2020. Transition-based directed graph construction for emotion-cause pair extraction. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 3707–3717.

Wei Fan, Yuexuan Zhu, Ziyun Wei, Tianyu Yang, WH Ip, and Yuxiang Zhang. 2021. Order-guided deep neural network for emotion-cause pair prediction. *Applied Soft Computing*, 112:107818.

Hao Fei, Chenliang Li, Donghong Ji, and Fei Li. 2022a. Mutual disentanglement learning for joint fine-grained sentiment classification and controllable text generation. In *Proceedings of the 45th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 1555–1565.

Hao Fei, Fei Li, Bobo Li, and Donghong Ji. 2021a. Encoder-decoder based unified semantic role labeling with label-aware syntax. In *Proceedings of the AAAI Conference on Artificial Intelligence*, pages 12794–12802.

Hao Fei, Fei Li, Chenliang Li, Shengqiong Wu, Jingye Li, and Donghong Ji. 2022b. Inheriting the wisdom of predecessors: A multiplex cascade framework for unified aspect-based sentiment analysis. In *Proceedings of the Thirty-First International Joint Conference on Artificial Intelligence, IJCAI*, pages 4096–4103.

Hao Fei, Jingye Li, Shengqiong Wu, Chenliang Li, Donghong Ji, and Fei Li. 2022c. Global inference with explicit syntactic and discourse structures for dialogue-level relation extraction. In *Proceedings of the Thirty-First International Joint Conference on Artificial Intelligence, IJCAI*, pages 4082–4088.

Hao Fei, Shengqiong Wu, Yafeng Ren, and Meishan Zhang. 2022d. Matching structure for dual learning. In *Proceedings of the International Conference on Machine Learning, ICML*, pages 6373–6391.

Hao Fei, Shengqiong Wu, Meishan Zhang, Yafeng Ren, and Donghong Ji. 2022e. Conversational semantic role labeling with predicate-oriented latent graph. In *Proceedings of the Thirty-First International Joint Conference on Artificial Intelligence, IJCAI*, pages 4089–4095.

Hao Fei, Meishan Zhang, and Donghong Ji. 2020. Cross-lingual semantic role labeling with high-quality translated training corpus. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7014–7026.

Hao Fei, Meishan Zhang, Bobo Li, and Donghong Ji. 2021b. End-to-end semantic role labeling with neural transition-based model. In *Proceedings of the AAAI Conference on Artificial Intelligence*, pages 12803–12811.

Kai Gao, Hua Xu, and Jiushuo Wang. 2015. Emotion cause detection for chinese micro-blogs based on ecocc model. In *Proceedings of the Pacific-Asia Conference on Knowledge Discovery and Data Mining*, pages 3–14.

Diman Ghazi, Diana Inkpen, and Stan Szpakowicz. 2015. Detecting emotion stimuli in emotion-bearing sentences. In *Proceedings of International Conference on Intelligent Text Processing and Computational Linguistics*, pages 152–165.

Lin Gui, Jiannan Hu, Yulan He, Ruifeng Xu, Qin Lu, and Jiachen Du. 2017. A question answering approach for emotion cause extraction. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 1593–1602.

Lin Gui, Dongyin Wu, Ruifeng Xu, Qin Lu, and Yu Zhou. 2016. Event-driven emotion cause extraction with corpus construction. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 1639–1649.

Guimin Hu, Guangming Lu, and Yi Zhao. 2021a. Bidirectional hierarchical attention networks based on document-level context for emotion cause extraction.

In *Proceedings of the Findings of the Association for Computational Linguistics: EMNLP 2021*, pages 558–568.

Guimin Hu, Guangming Lu, and Yi Zhao. 2021b. Fssgcn: A graph convolutional networks with fusion of semantic and structure for emotion cause analysis. *Knowledge-Based Systems*, 212:106584.

Sophia Yat Mei Lee, Ying Chen, and Chu-Ren Huang. 2010. A text-driven rule-based system for emotion cause detection. In *Proceedings of the NAACL HLT 2010 workshop on computational approaches to analysis and generation of emotion in text*, pages 45–53.

Jingye Li, Hao Fei, and Donghong Ji. 2020. Modeling local contexts for joint dialogue act recognition and sentiment classification with bi-channel dynamic convolutions. In *Proceedings of the 28th International Conference on Computational Linguistics, COLING*, pages 616–626.

Xiangju Li, Kaisong Song, Shi Feng, Daling Wang, and Yifei Zhang. 2018. A co-attention neural network model for emotion cause analysis with emotional context awareness. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 4752–4757.

Bing Liu. 2012. Sentiment analysis and opinion mining. *Synthesis lectures on human language technologies*, 5(1):1–167.

Ilya Loshchilov and Frank Hutter. 2017. Fixing weight decay regularization in adam. *CoRR*, abs/1711.05101.

Alena Neviarouskaya and Masaki Aono. 2013. Extracting causes of emotions from text. In *Proceedings of the Sixth International Joint Conference on Natural Language Processing*, pages 932–936.

Wenxuan Shi, Fei Li, Jingye Li, Hao Fei, and Donghong Ji. 2022. Effective token graph modeling using a novel labeling strategy for structured sentiment analysis. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics*, pages 4232–4241.

Shuangyong Song and Yao Meng. 2015. Detecting concept-level emotion cause in microblogging. In *Proceedings of the 24th International Conference on World Wide Web*, pages 119–120.

Duyu Tang, Bing Qin, and Ting Liu. 2016. Aspect level sentiment classification with deep memory network. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 214–224.

Penghui Wei, Jiahao Zhao, and Wenji Mao. 2020. Effective inter-clause modeling for end-to-end emotion-cause pair extraction. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 3171–3181.

Yinwei Wei, Zhiyong Cheng, Xuzheng Yu, Zhou Zhao, Lei Zhu, and Liqiang Nie. 2019a. Personalized hashtag recommendation for micro-videos. In *Proceedings of the 27th ACM International Conference on Multimedia*, pages 1446–1454.

Yinwei Wei, Xiang Wang, Weili Guan, Liqiang Nie, Zhouchen Lin, and Baoquan Chen. 2019b. Neural multimodal cooperative learning toward micro-video understanding. *IEEE Transactions on Image Processing*, 29:1–14.

Yinwei Wei, Xiang Wang, Liqiang Nie, Xiangnan He, Richang Hong, and Tat-Seng Chua. 2019c. Mmgcn: Multi-modal graph convolution network for personalized recommendation of micro-video. In *Proceedings of the 27th ACM International Conference on Multimedia*, pages 1437–1445.

Shengqiong Wu, Hao Fei, Fei Li, Meishan Zhang, Yijiang Liu, Chong Teng, and Donghong Ji. 2022. Mastering the explicit opinion-role interaction: Syntax-aided neural transition system for unified opinion role labeling. In *Proceedings of the AAAI Conference on Artificial Intelligence*, pages 11513–11521.

Shengqiong Wu, Hao Fei, Yafeng Ren, Donghong Ji, and Jingye Li. 2021. Learn from syntax: Improving pair-wise aspect and opinion terms extraction with rich syntactic knowledge. In *Proceedings of the Thirtieth International Joint Conference on Artificial Intelligence*, pages 3957–3963.

Rui Xia and Zixiang Ding. 2019. Emotion-cause pair extraction: A new task to emotion analysis in texts. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 1003–1012.

Hanqi Yan, Lin Gui, Gabriele Pergola, and Yulan He. 2021a. Position bias mitigation: A knowledge-aware graph model for emotion cause extraction. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 3364–3375.

Zhiheng Yan, Chong Zhang, Jinlan Fu, Qi Zhang, and Zhongyu Wei. 2021b. A partition filter network for joint entity and relation extraction. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 185–197.

Chaofa Yuan, Chuang Fan, Jianzhu Bao, and Ruifeng Xu. 2020. Emotion-cause pair extraction as sequence labeling based on a novel tagging scheme. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 3568–3573.

Causal Intervention Improves Implicit Sentiment Analysis

Siyin Wang[†], Jie Zhou[†*], Changzhi Sun[§], Junjie Ye[†], Tao Gui[†], Qi Zhang[†], and Xuanjing Huang[†]

[†]School of Computer Science, Fudan Univerisity
[§]ByteDance AI Lab
{siyinwang20, jie_zhou, jjye19}@fudan.edu.cn
{tgui, qz, xjhuang}@fudan.edu.cn
sunchangzhi@bytedance.com

Abstract

Despite having achieved great success for sentiment analysis, existing neural models struggle with implicit sentiment analysis. This may be due to the fact that they may latch onto spurious correlations ("shortcuts", e.g., focusing only on explicit sentiment words), resulting in undermining the effectiveness and robustness of the learned model. In this work, we propose a CausaL intervention model for implicit sEntiment ANalysis using instrumental variable (CLEAN). We first review sentiment analysis from a causal perspective and analyze the confounders existing in this task. Then, we introduce an instrumental variable to eliminate the confounding causal effects, thus extracting the pure causal effect between sentence and sentiment. We compare the proposed CLEAN model with several strong baselines on both the general implicit sentiment analysis and aspect-based implicit sentiment analysis tasks. The results indicate the great advantages of our model and the efficacy of implicit sentiment reasoning.

1 Introduction

The remarkable success that the field of sentiment analysis has achieved in the past few years has been derived from the use of increasingly high-capacity neural models to extract correlations from data (Peters et al., 2018; Devlin et al., 2018; Liu et al., 2019). Although having reached state-of-the-art results, correlational predictive models can be untrustworthy (Guidotti et al., 2018): they may latch onto spurious correlations ("shortcuts"), leading to poor generalization.

One shortcut might be the explicit sentiment word which is a powerful feature cue. Unfortunately, such a shortcut severely harms the generalization and the robustness of the learned models in implicit sentiment analysis (ISA), where there are no explicit sentiment words about the topic or aspect (Russo et al., 2015). Figure 1 gives a sample of aspect-based sentiment analysis (ABSA) (Zhou et al., 2020b,a), which aims to predict the sentiments of the aspects in the sentence. The aspect "food" has explicit sentiment words "definitely good", but aspect "price" does not. If the model thoughtlessly relies on shortcuts to sentiment words, it may make an incorrect sentiment prediction about the aspect "price". In fact, there are many other kinds of shortcuts that models may learn, for example, the rhetorical question mood expressed by the users (Ranganath et al., 2018) and the co-occurrence of neutral words and sentiment polarities (Wang and Culotta, 2020).

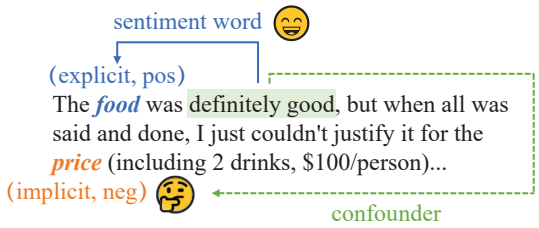

Figure 1: An examples of confounding factors in implicit sentiment analysis for ABSA.

The above shortcomings can potentially be addressed by the causal perspective: knowledge of causal relationships between observations and labels can be used to formalize spurious correlations and alleviate the predictor's dependence on them (Bühlmann, 2020; Veitch et al., 2021; Feder et al., 2021). Motivated by a causal perspective, we incorporate domain knowledge of the causal structure of the data into the learning objective. Specifically, causal intervention is used to curb dependence on shortcuts (e.g., "good → positive") and improve the ability to reason *causal effect* between sentence and sentiment.

In this paper, we rethink the ISA task from a causal perspective and unitize the casual intervention on deep learning. We argue that the causal effect obtained by reasoning directly from the sen-

*Corresponding authors.

Proceedings of the 29th International Conference on Computational Linguistics, pages 6966–6977
October 12–17, 2022.

tence (X) to the sentiment (Y) without relying on other extra prior stereotypical lexical impressions is closer to the original semantic analysis. Our work aims at eliminating the confounding causal effects of $C \rightarrow Y$ and thus extracting the pure causal effect between sentence and sentiment. Inspired by the instrumental variable in causality, we propose a CausaL intervention model for implicit sEntiment ANalysis using instrumental variable (CLEAN). Different from the other work with causal intervention like back-door adjustment (Landeiro and Culotta, 2016), other variables like confounders are not required to be observed. CLEAN contains two-stage learning: (1) In the first stage, we model the relationship between the instrumental variable and sentence; (2) In the second stage, we dismiss the spurious correlation between confounders and sentiment by means of the relationship obtained from the first stage.

To evaluate the effectiveness of our CLEAN, we conduct a series of experiments on both the general implicit sentiment analysis and aspect-based implicit sentiment analysis. In particular, we compare our model with several the mainstream baselines and the results show the great advantages of our model on ISA. We also validate the robustness of the model by adversarial attack and case studies, which proves that our model successfully dismisses the spurious correlation caused by sentiment words and extracts the pure causal effect.

The main contributions are summarized as follows.

- We rethink the implicit sentiment from a causal perspective and proposed a casual intervention model for implicit sentiment analysis (CLEAN).
- To remove the spurious causes of confounders, we incorporate instrumental variable into neural network to enhance its causal reasoning ability.
- We conduct experiments on diverse datasets, including partially implicit and totally implicit sentiment, which shows our effectiveness and rationality to reason implicit sentiment.

2 Preliminaries

2.1 Structural Causal Model and Causal Effect

In our paper, Structural Causal Model (SCM) (Glymour et al., 2016) is represented as a directed acyclic graphs (DAGs) $G = \{V, E\}$ to reflect causal relationships, where V denotes the set of observational variables and E denotes the direct causal effect (Figure 2(a)). X is a direct cause of Y when variable Y is the child of X.

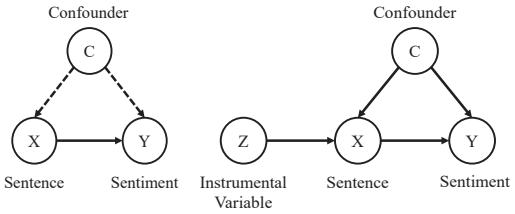

(a) Causal Graph (b) Instrument Variable for Causal Intervention

Figure 2: Causal Graph

Variable X and Y is called treatment variable and outcome variable respectively when observing the causal relationship between them. The other variables we do not consider their causal relationship are called error terms (ε), also known as exogenous variables. Significantly, total effect between X and Y, denoted as $P(Y \mid X)$, is conceptually different from causal effect of $X \rightarrow Y$, denoted as $P(Y \mid do(X = x))$ because the causal effect only involves the direct path from X to Y, while the total effect involves all paths between X and Y. Based on the ideal hypothesis that none of the error terms will involve in the path between X and Y, people usually treat the total effect and the causal effect as the same. But the actual fact is that a part of error terms (we call it confounder (C)) serves as a common cause of the treatment and outcome, denoted as $X \leftarrow C \rightarrow Y$. Consequently, the total effect is practically different from the causal effect, i.e. $P(Y \mid do(X = x)) \neq P(Y \mid X)$ and treatment-outcome relationship may well be obscured by the spurious correlation between C and Y generated by confounder (Pearl, 2009; Hernán MA, 2020).

2.2 Instrument Variable for Causal Intervention

To recover the gap between total effect $P(Y \mid X)$ and casual effect $P(Y \mid do(X = x))$ and derive pure causal effect, we must "adjust" for potential confounder (C) (Pearl, 1995). Fortunately, applying causal intervention can extract the pure causality from the correlation and therefore overcome the problem of confounding bias. There are four key interventions: randomized controlled trial, backdoor adjustment, front-door adjustment, and instrumental variable estimation. Randomized con-

trolled trials are simply not practicable for natural language, and both the front-door and back-door adjustment require additional observable variables. However, the confounders (e.g., rhetoric confounding word, such as rhetorical questions and sarcasm) are too polymorphic to be observed exhaustively for implicit sentiment analysis. We adopt the instrumental variable to dismiss the spurious correlations instead of directly observing confounders (Figure 2(b)).

Before the intervention, we should find a suitable instrumental variable (Z) that qualifies well the requirements as follows:(Brito and Pearl, 2012)

1. Z is independent of all error terms ε that have an influence on Y which is not mediated by X, $Cov(Z, \varepsilon) = 0$.

2. Z is not independent of X, $Cov(Z, X) \neq 0$.

The intuition behind this definition is that all correlation between Z and Y requires X to act as an intermediary.

Generally, instrument variable estimation contains two stages (Angrist and Pischke, 2008). In the first stage, the coefficient α is obtained by regression estimation of X and Z, denoted as $Cov(Z, X)$. In the second stage, substitute X with the expressions including Z obtained in the first stage into the expression of Y and then regress Y on Z , denoted as $Cov(Z, Y)$. There is no confounding bias between Y and Z owing to the restriction in the definition of Z, i.e. $Cov(Z, \varepsilon) = 0$. A simple linear model for IV estimation consists of 2 equations:

$$X = \alpha Z + \varepsilon_X; Y = \omega X + \varepsilon_Y \tag{1}$$

where Y is the outcome variable (e.g., sentiment), X is the treatment variable (e.g, sentence), Z is the instrumental variable (e.g., stochastic perturbation), and ε_X and ε_Y are error terms including but not limited to confounders(C). Under the conditions above, it can be proved that the equation presents an asymptotically unbiased estimate of the effect of X on Y (Angrist et al., 1996).

$$\omega_{IV} = \frac{\frac{1}{n}\sum_{i=1}^{n}(z_i - \overline{z})(y_i - \overline{y})}{\frac{1}{n}\sum_{i=1}^{n}(z_i - \overline{z})(x_i - \overline{x})} = \frac{Cov(Z, Y)}{Cov(Z, X)} \tag{2}$$

3 Our Approach

In this section, we introduce our CLEAN model for implicit sentiment analysis (Figure 3). We first rethink the ISA from a causal perspective (Section 3.1). Then, we adopt stochastic perturbation as instrumental variable (Section 3.2) and estimate instrumental variable in two stages (Section 3.3 and 3.4).

3.1 Sentiment Analysis from Casual Perspective

Given a sentence X, consisting of a sequence of tokens $(x_1, x_2, ..., x_n)$, our task aims to analyze the polarity Y. For aspect-based sentiment analysis task, we concatenate the sentence and the aspect as the input $X = (x_1, x_2, ..., x_n, [SEP], a_1, a_2, ..., a_m)$. In the current method, a deep neural network is used as a classifier to predict the sentiment polarity label as output and the sentence as input (as Equation 3).

$$y = h(x; w) = W_{xy} \cdot x + \varepsilon_y \tag{3}$$

where ε_y denotes as the error terms including confounders (c) and other errors ($\hat{\varepsilon_y}$).

The prediction above is based on the hypothesis that error terms will not involve in the causal path between X and Y and ignore the influence of error terms mostly. Nevertheless, several research has found that text classification systems based on neural networks are biased towards learning frequent spurious correlations (Leino et al., 2018). It urges us to focus on the longtime unheeded but unavoidable existence of confounder (C) in error terms, which results in the overlooked gap between *total effect* and *causal effect*, denoted as the path $X \leftarrow C \rightarrow Y$. The Equation 3 can be updated in consideration of confounder (c) (Equation 4).

$$y = h(x; w) = W_{xy} \cdot x + W_{cy} \cdot c + \hat{\varepsilon_y} \tag{4}$$

where c and $\hat{\varepsilon_y}$ denotes the confounder and other error terms respectively, W_{cy} denotes the causal effect of $C \rightarrow Y$.

In previous studies, gender (Field and Tsvetkov, 2020), age, and address (Landeiro and Culotta, 2016) were found to be confounders in text classification. As for ISA, we focus rather on the naturally existing confounder within the text, i.e., sentiment words. Sentiment words affect the form of the text as a component of the text (i.e., the writer's word choice determines the form of expression) and also affect sentiment expression (Xing et al., 2020). The existence of sentiment words as confounder makes it difficult to distinguish the pure causal effect of $X \rightarrow Y$ and the prediction indiscriminately depends on the spurious correlation

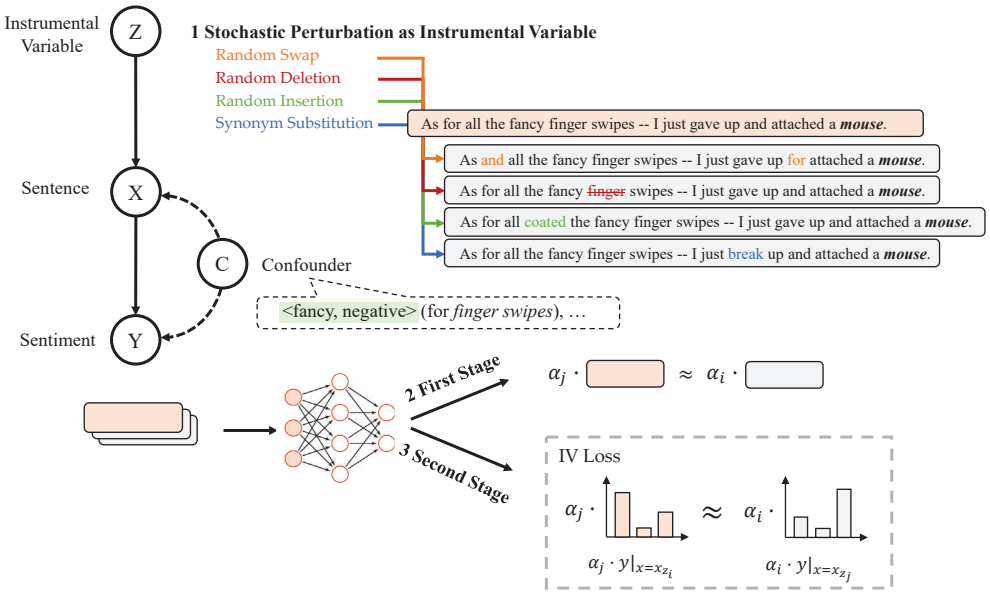

Figure 3: The framework of our CLEAN.

between sentiment words and sentiment will fail in ISA. The main forms are as follows:

• **Inter-aspect Confounding Word** Explicit sentiment words of other aspects with opposite sentiment in the sentence confounds the prediction effect of the current aspect.

• **Inter-clause Confounding Word** In an adversative compound sentence, the other clauses with opposite semantics confound the prediction effect of the current clause.

• **Rhetoric Confounding Word** Sentiment words conveying the opposite of the norm in rhetorical devices such as rhetorical questions and sarcasm confound the prediction effect.

• **Dynamic Neural Confounding Word** Neutral words show dynamic sentiment polarity in different contexts, but the model trained by biased data only learns the spurious correlation of one polarity.

We also provide a detailed analysis of the confounder in case studies (Section 5.2).

Inspired by causal intervention with instrumental variable (Section 2.2), we adopt two-stage instrument variable estimation for ISA to achieve the goal that distinguishes the pure *causal effect* of $X \rightarrow Y$ without any spurious correlations, denoted as $P(Y \mid do(X = x))$.

3.2 Stochastic Perturbation as Instrumental Variable

For text, the two restrictions of instrumental variable could be translated into two basic opinions: (1) instrumental variable Z will not influence the sentiment polarity via any other casual path except through sentence X; (2) instrumental variable Z will influence the format of sentence X. Intuitively, we choose the stochastic perturbation as the instrumental variable of ISA. Inspired by the work of data augmentation (Guohang et al., 2020), we choose random swap, random deletion, random insertion, and synonym substitution as stochastic perturbation: A) **Random Swap**: Swap word randomly; B) **Random Deletion**: Delete word randomly with probability p; C) **Random insertion**: Insert word randomly by word embeddings similarity; D) **Synonym Substitution**: Substitute word by WordNet's synonym. It fortunately meets the requirements of instrumental variable well: (1) stochastic perturbation obviously has no independent effect on sentiment, except through augmentation sentences, i.e. $Cov(Z, \varepsilon) = 0$; (2) stochastic perturbation above will definitely change the sentence into another form, i.e. $Cov(Z, X) \neq 0$.

3.3 The First Stage of CLEAN

Following the traditional pattern of instrumental variable estimation (Section 2.2), the first stage of CLEAN is to establish the causal relationship between stochastic perturbation (Z) and sentence (X), i.e. $Z \rightarrow X$. We use two open source tools[1] to generate augmentation samples x_z from the original sample x and the formal expression is as follows.

$$x_z = f(x, z)$$

where $f(\cdot)$ denotes the different stochastic perturbation on the original sentence. For a specific stochastic perturbation z_i, we have $x_{z_i} = f(x, z_i) \approx \alpha_i \cdot x$.

[1] https://github.com/jasonwei20/eda_nlp
https://github.com/makcedward/nlpaug

Table 1: The statistics information of the datasets. IS means the percent of samples that are implicit sentiment.

Dataset	Postive		Neural		Negative		IS (%)	
	Train	Test	Train	Test	Train	Test	Train	Test
Restaurant	2164	728	805	196	633	196	28.59	23.84
Laptop	987	341	866	128	460	169	30.87	27.27
CLIPEval	435	144	205	72	640	155	100.00	100.00

To obtain the accurate value of α which can well represent the relationship between x and x_z, a neural network was constructed based on the BERT and α was set as a self-learning parameter.

$$\alpha_i = \min_{\alpha} \sum_{x_{z_i}=f(x,z_i)} \parallel \mathcal{M}(x_{z_i}) - \alpha \cdot \mathcal{M}(x) \parallel$$

where $\mathcal{M}$ denotes a text encoder (e.g., BERT).

3.4 The Second Stage of CLEAN

Substituting the relation above between original sample x and augmentation sample x_{z_i} into Equation 4, we will get the $y \mid_{x=x_{z_i}}$ with different proportionality coefficient α obtained from the first stage.

$$\begin{aligned} y \mid_{x=x_{z_i}} &= W_{xy} \cdot x_{z_i} + W_{cy} \cdot c + \hat{\varepsilon_y} \\ &= \alpha_i \cdot W_{xy} \cdot x + W_{cy} \cdot c + \hat{\varepsilon_y} \\ &= \alpha_i \cdot y \mid_{do(x=x)} + W_{cy} \cdot c + \hat{\varepsilon_y} \end{aligned} \tag{5}$$

where $y \mid_{do(x=x)}$ denotes the prediction only along the path $X \rightarrow Y$.

As the neural network is not totally linear, we slightly generalize the usage of two stages in instrumental variable. We set the dismission of influence of the confounder as a regularization function instead of directly calculating the effect between X and Y as a single value (Equation 2), which is obviously more fit for deep learning method.

$$\mathcal{L}_{IV} = \sum_{i \neq j} \parallel \alpha_j \cdot y \mid_{x=x_{z_i}} - \alpha_i \cdot y \mid_{x=x_{z_j}} \parallel$$

The reason we just model the prediction $y \mid_{x=x_{z_i}}$ and unitize the regularization loss on it is that the essence of the $\mathcal{L}_{IV}$ is to force the model to suppress the confounding effect caused by sentiment words. It can be easily proved by substituting the $y \mid_{x=x_{z_i}}$ with Equation 5 obtained by two-stage learning. The benefit is obviously that we can suppress the confounding effect without directly observing the confounders (c).

$$\mathcal{L}_{IV} = \sum_{i \neq j} \parallel (\alpha_i - \alpha_j) \cdot (W_{cy} \cdot c + \hat{\varepsilon_y}) \parallel$$

In addition, the model should not go to the other extreme, i.e., ignore sentiment words entirely, which would be inconsistent with the normal process of natural language understanding. We set a hyperparameter β to achieve balance and combine the causal regularization loss function $\mathcal{L}_{IV}$ with the conventional cross-entropy loss $\mathcal{L}_{CE}$ and the influence of the β is analyzed in Section 5.3.

$$\mathcal{L}_{ALL} = \mathcal{L}_{CE} + \beta \mathcal{L}_{IV}$$

4 Experiment

4.1 Datasets and Metrics

Implicit Sentiment Analysis To show our model's better performance in understanding implicit sentiment, we evaluate the implicit polarity prediction on a total implicit dataset, CLIPEval from SemEval 2015 task 9 (Russo et al., 2015), which consists of self-reported entity reviews collected from psychological research with 1,280 sentences for the training and 371 for the test.

Aspect-based Implicit Sentiment Analysis As our aim to dismiss the spurious correlation between explicit sentiment words and polarity, we also conducted experiments on both explicit and implicit datasets, Laptop and Restaurant review from SemEval 2014 task 4 (Pontiki et al., 2014). The segmentation of explicit sentiment (ESE) and implicit sentiment (ISE) is based on the work of (Li et al., 2021b) based on the annotation of opinion words (Fan et al., 2019).

We adopt two widely used metrics accuracy and macro-F1 to evaluate the performance of our model and the baselines.

4.2 Baselines

To investigate the effectiveness of our CLEAN model, we compare it with several typical baseline models for implicit sentiment analysis and aspect-based implicit sentiment analysis.

Implicit Sentiment Analysis We select four popular baselines for implicit sentiment analysis, which are listed as follows. SHELIFBK (Dragoni,

Table 2: The main results for aspect-based sentiment analysis. For ESE and ISE, we provide the F1 score. We use the results reported in (Li et al., 2021b). The baselines with † are our implementation.

		Restaurant				Laptop			
		Acc	F1	ESE	ISE	Acc	F1	ESE	ISE
NN	ATAE-LSTM	76.90	62.64	84.16	53.71	65.37	62.92	75.69	37.86
	IAN	76.88	67.71	86.52	46.07	67.24	63.72	75.86	44.25
	RAM	80.23	70.80	85.11	55.81	74.49	71.35	75.86	44.25
	MGAN	81.25	71.94	85.18	60.04	75.39	72.47	76.16	56.31
GNN	TransCap	79.55	71.41	86.52	59.93	73.87	70.10	77.16	60.34
	ASGCN	80.77	72.02	84.29	62.91	75.55	71.05	75.46	57.77
	BiGCN	81.97	73.48	87.19	59.05	74.59	71.84	79.53	62.64
	CDT	82.30	74.02	88.79	65.87	77.19	72.99	77.53	68.90
	RGAT	83.30	76.08	89.45	61.05	77.42	73.76	80.17	65.52
BERT	BERT-SPC	83.57	77.16	89.21	65.54	78.22	73.45	81.47	69.54
	CapsNet+BERT	85.09	77.75	91.68	64.04	78.21	73.34	82.33	67.24
	BERT-PT	84.95	76.96	92.15	64.79	78.07	75.08	81.47	71.27
	BERT-ADA	**87.14**	80.05	**94.14**	65.92	78.96	74.18	**82.76**	70.11
	R-GAT+BERT	86.60	81.35	92.73	67.79	78.21	74.07	82.44	72.99
	TransEncAsp	77.10	57.92	86.97	48.96	65.83	59.53	74.31	43.20
	TransEncAsp+SCAPT	83.39	74.53	88.04	68.55	77.17	73.23	78.70	72.82
	BERT-SPC†	85.09	77.19	91.68	64.04	77.90	73.50	80.99	69.71
	BERT-SPC† (Aug4)	84.20	76.55	90.50	64.04	76.65	70.86	81.64	63.43
	BERT-SPC† (Aug8)	80.98	67.77	90.39	50.94	75.71	71.62	77.97	69.71
	BERT-SPC† (Aug16)	77.59	67.44	85.35	52.81	74.61	69.92	77.97	65.71
Ours	CLEAN	87.05	**81.40**	92.50	**69.66**	**80.41**	**77.25**	81.21	**78.29**

2015), ATTLSTM (Lin et al., 2017), MTL (Zheng et al., 2018), BERT-SPC (Xu et al., 2019).

Aspect-based Implicit Sentiment Analysis The commonly used baselines can be split into three parts, neural network, graph neural network, and BERT-based models, which are given as follows.

Neural Network: ATAE-LSTM (Wang et al., 2016), IAN (Ma et al., 2017), RAM (Chen et al., 2017), MGAN (Fan et al., 2018).

Graph Neural Network: TransCap (Chen and Qian, 2019), ASGCN (Zhang et al., 2019), BiGCN (Zhang and Qian, 2020), CDT (Sun et al., 2019), RGAT (Wang et al., 2020).

BERT-based Models: BERT-SPC (Xu et al., 2019), CapsNet+BERT (Jiang et al., 2019), BERT-ADA (Rietzler et al., 2020), R-GAT+BERT (Wang et al., 2020), TransEncAsp (Li et al., 2021b), TransEncAsp+SCAPT (Li et al., 2021b).

Moreover, to explore the influence of the augmentation sentences, we add them into the training dataset for our basic model (BERT-SPC). For example, BERT-SPC (Aug4) means we add four augmentation samples for each example.

4.3 Implementation Details

We implement CLEAN with PyTorch based on Hugging Face Transformer [2] and run them on the GPU(NVIDIA GTX 2080ti). During training, we set the coefficient λ of $\mathcal{L}_2$ regularization item is 0.01, 10^{-5} and dropout rate is 0.1. The learning rate is set as 2e-5 and the batch size is set as 16. Adam optimizer (Kingma and Ba, 2014) is used to update all the parameters.

[2] https://huggingface.co/bert-base-uncased.

5 Experimental Analysis

5.1 Main Results

To evaluate the performance of our CLEAN model, we compare it with several mainstream baseline models for both the implicit sentiment analysis and aspect-based implicit sentiment analysis (Table 3 and Table 2). We find the following observation from these tables. **First**, our model outperforms all the baselines in most cases. Particularly, our model obtains the best F1 scores over all the three datasets of two tasks. **Second**, our CLEAN strategy significantly improves the performance of the baseline. CLEAN improves more than two points in terms of F1 over all the datasets compared with BERT-SPC, which is the base of our model. **Third**, our model can improve the performance of implicit sentiment analysis effectively. For example, compared with the BERT-SPC†, we obtain more than five points improvement on ISE over both Restaurant and Laptop. Also, we obtain the best results of implicit sentiment analysis over CLIPEval. **Forth**, the model that regards the augmentation sentence as a data augmentation (e.g., BERT-SPC† (Aug4)) performs even worse than the one without augmentation as noise may exist. This shows that our

	Sentence Example	Target	BERT-SPC	ISAIV
E1	The food was definitely good , but when all was said and done, I just couldn't justify it for the **price** (including 2 drinks, $100/person)...	price	positive ✗	negative ✓
E2	And as for all the fancy finger swipes -- I just gave up and attached a **mouse**.	mouse	negative ✗	neutral ✓
E3	I was a little concerned about the **touch pad** based on reviews, but I've found it fine to work with.	touch pad	negative ✗	positive ✓
E4	How can hope to stay in business with **service** like this?	service	positive ✗	negative ✓
E5	The **steak** melted in my mouth.	steak	negative ✗	positive ✓
E6	15% **gratuity** automatically added to the bill.	gratuity	positive ✗	positive ✗

Figure 4: Some examples of case studies.

Table 3: The main results for implicit sentiment analysis. We use the results reported in (Xiang et al., 2021). The baselines with † are our implementation.

Method	CLIPEval	
	Acc	F1
SHELLFBK	56.00	54.00
ATTLSTM	82.43	82.21
MTL	82.94	83.17
BERT-SPC†	87.06	84.74
BERT-SPC† (Aug4)	85.71	83.56
BERT-SPC† (Aug8)	86.52	85.30
BERT-SPC† (Aug16)	85.44	84.54
CLEAN	**88.95**	**87.49**

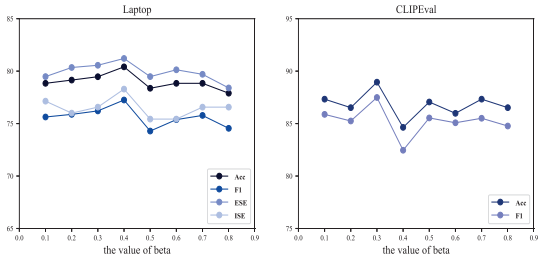

(a) Laptop (b) CLIPEval

Figure 5: The influence of β

CLEAN algorithm can help learn the implicit sentiment reasoning behind the data.

5.2 Case Studies

We present five samples in Figure 4 to explain the four main types of confounders (Section 3.1), which shows the effectiveness and rationality of our model to reason implicit sentiment. (1) **Inter-aspect Confounding Word.** In example E1, "definitely good" is the sentiment words of aspect *food*, implying positive sentiment but confounds the prediction of aspect *price*. In E2, the user expresses a negative sentiment towards aspect *finger swipes* with opinion word "fancy", which confounds the prediction of aspect *mouse*. (2) **Inter-clause Confounding Word.** In E3, the first and second clauses form an adversative relation, and the true meaning of the expression is that the *mouse* works well, but the sentiment word "a little concerned" in the first clause confounds the prediction. (3) **Rhetoric Confounding Word.** In E4, the customer used the rhetorical device of a rhetorical question to express that the restaurant's service was terrible, but the existence of the word "hope" confounds the prediction, (4) **Dynamic Neural Confounding Word.** In E5, the word "melted" is absolutely a neutral word, but when we directly count the proportion of aspect-level sentiment polarity that co-occur with "melted", we surprisingly find that 83.33% aspect polarity is negative, which well explains why the previous model predicts "negative" strangely. Due to the unbalanced distribution of training data, the model tends to tag the neural word with specific sentiment polarity and predict based on this spurious correlation learned superficially before.

5.3 Further analysis

Influence of Augmentation Sample Number. We explore the influence of augmentation sample number here (Table 4). The influence of sample number on model performance depends on two conflicting factors: the degree of deviation from the original sentence and the chance to find more potential confounders. With the increase in sample number, the model has a greater chance of finding more potential confounders and adjusting for them. On the other hand, a larger generation samples number means that more samples deviating from the original sentence are involved in the learning procedure, and therefore the accuracy of prediction decreases. Over Restaurant, the two conflicting factors reach a better balance at 8; while on Laptop, the negative effect of semantic deviation outweighs the positive

Table 4: The influence of augmentation sample number.

#Num	Restaurant 0.6				Laptop 0.4			
	Acc	F1	ESE	ISE	Acc	F1	ESE	ISE
4	86.88	80.99	92.50	68.91	**80.41**	**77.25**	**81.21**	**78.29**
8	**87.05**	**81.40**	**92.50**	**69.66**	78.68	75.23	79.70	76.00
16	85.09	77.75	91.21	65.54	78.06	75.04	79.05	75.43

Table 5: The results of robustness.

Model	Restaurant (Trans.)		Laptop (Trans.)	
	Acc	F1	Acc	F1
BERT-SPC	57.04	44.43	51.05	41.01
CLEAN	58.77	48.56	59.45	43.24

effect of correction for more confounders, and the best result is achieved at 4.

Influence of β. Either emphasizing sentiment words only or completely ignoring them is not reasonable. The purpose of our hyper-parameter beta is to strike a balance between these two terms (Figure 5). In Laptop and CLIPEval, performance is best at 0.4 and 0.3 relatively, and both show a trend of high in the middle and low on both sides, indicating that our hypothesis is rational.

Robustness. We also analyze the robustness of our proposed CLEAN (Table 5). We test our model on a robustness testing dataset, Revnon of TextFlint (Wang et al., 2021), which reverses the sentiment of the non-target aspects with originally the same sentiment as target. Our model outperforms the model BERT-SPC without causal intervention, which means CLEAN can also improve the robustness by learning the implicit sentiment reasoning.

Limitation. We also analyze wrong samples and find the model may fail when encountering the expression with unusual knowledge. In Figure 4 E6, due to the lack of prior knowledge about “gratuity", “automatically added" is likely to be perceived as a good thing. Admittedly, our work mainly focuses on the reasoning ability and doesn’t integrate external corpus and knowledge and therefore lacks abundant prior knowledge. The better combination of prior knowledge and causal inference is also an intriguing and worth exploring field.

6 Related Work

6.1 Implicit Sentiment Analysis

Implicit sentiment analysis (ISA) task plays an important role in sentiment analysis field (Liu, 2012; Zhou et al., 2019, 2020c). Early studies mainly trained machine learning models based on handcrafted features or explicit characterization of implicit feature information. Some studies argued that seemingly neural words actually contain emotional content and then construct a lexicon (Feng et al., 2013; Castelló and Stede, 2017). Label propagation was used to judge the affective polarity of the words automatically (Ding and Riloff, 2016; Li et al., 2021a). Moreover, Balahur et al. (2011) proposed to build a commonsense knowledge base (EmotiNet) with the concept of affective value and the sentiment.

Recent efforts (He et al., 2018; Tang et al., 2020) used syntax information from dependency trees to enhance attention-based models. Using syntactic analysis tree and CNN, Liao et al. (2019) analyzed fact-implied implicit sentiment by fusing multi-level semantic information, including sentiment target, sentence, and context semantic. A lot of works (Zhang et al., 2019; Sun et al., 2019; Wang et al., 2020) incorporated tree-structured syntactic information via graph neural networks to capture aspect-aware information in text. Another method is to utilize external corpus and pre-trained knowledge to enhance semantic awareness of models (Xu et al., 2019; Rietzler et al., 2020; Dai et al., 2021; Li et al., 2021b; Zhou et al., 2020b).

The existing methods mainly improve the ISA by integrating external corpus and knowledge. However, the knowledge is always not complete which will influence the models’ performance. In this paper, we solve it via causal intervention to learn the reasoning behind the sentiment classification.

6.2 Causality for NLP

Recently, some researchers are beginning to combine causality and NLP to create more robust and interpretable models (Wood-Doughty et al., 2018; Tang et al., 2021). Most of the papers integrated backdoor and counterfactual into NLP tasks. Particularly, Landeiro and Culotta (2016) applied the back-door adjustment to text classification by controlling the artificially predetermined confounding variable. Feng et al. (2021) introduced counterfactual reasoning into the model learning process by generating representative counterfactual samples

and comparing the counterfactual and factual samples. Veitch et al. (2021) utilized distinct regularization schema for distinct causal structure to induce counterfactual invariance. Niu et al. (2021) utilized the counterfactual inference on VQA models by subtracting the language bias as direct language effect from the total causal effect.

Different from these studies, we explore the causal graph for ISA and incorporate it using the causal intervention.

7 Conclusion

In this paper, we proposed a causal intervention model for implicit sentiment analysis using instrument variable (CLEAN). Given that the current model indiscriminately focuses on the correlation between sentiment and sentiment words and consequently performs poorly in implicit sentiment analysis as the explicit sentiment words disappear, we rethink the implicit sentiment analysis from a causal perspective and analyze the four main forms of sentiment words as potential confounders. Inspired by the instrumental variable of causal intervention, we adopt stochastic perturbation as instrumental variable and construct a model with two-stage learning. Across three different datasets, including general implicit sentiment analysis and aspect-based sentiment analysis, our CLEAN shows great advantages in implicit sentiment.

Acknowledge

The authors wish to thank the reviewers for their helpful comments and suggestions. This work was partially funded by the National Natural Science Foundation of China (No. 61976056, 62076069), Shanghai Municipal Science and Technology Major Project (No.2021SHZDZX0103).

References

Joshua D Angrist, Guido W Imbens, and Donald B Rubin. 1996. Identification of causal effects using instrumental variables. *Journal of the American statistical Association*, 91(434):444–455.

Joshua D Angrist and Jörn-Steffen Pischke. 2008. Mostly harmless econometrics. In *Mostly Harmless Econometrics*. Princeton university press.

Alexandra Balahur, Jesús M Hermida, and Andrés Montoyo. 2011. Detecting implicit expressions of sentiment in text based on commonsense knowledge. In *Proceedings of the 2nd Workshop on Computational Approaches to Subjectivity and Sentiment Analysis (WASSA 2.011)*, pages 53–60.

Carlos Brito and Judea Pearl. 2012. Generalized instrumental variables. *arXiv preprint arXiv:1301.0560*.

Peter Bühlmann. 2020. Invariance, causality and robustness. *Statistical Science*, 35(3):404–426.

Núria Bertomeu Castelló and Manfred Stede. 2017. Extracting word lists for domain-specific implicit opinions from corpora. In *IWCS 2017 - 12th International Conference on Computational Semantics - Long papers*.

Peng Chen, Zhongqian Sun, Lidong Bing, and Wei Yang. 2017. Recurrent attention network on memory for aspect sentiment analysis. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing, EMNLP 2017, Copenhagen, Denmark, September 9-11, 2017*, pages 452–461. Association for Computational Linguistics.

Zhuang Chen and Tieyun Qian. 2019. Transfer capsule network for aspect level sentiment classification. In *Proceedings of the 57th Conference of the Association for Computational Linguistics, ACL 2019, Florence, Italy, July 28- August 2, 2019, Volume 1: Long Papers*, pages 547–556. Association for Computational Linguistics.

Junqi Dai, Hang Yan, Tianxiang Sun, Pengfei Liu, and Xipeng Qiu. 2021. Does syntax matter? a strong baseline for aspect-based sentiment analysis with RoBERTa. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 1816–1829, Online. Association for Computational Linguistics.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. *north american chapter of the association for computational linguistics*.

Haibo Ding and Ellen Riloff. 2016. Acquiring knowledge of affective events from blogs using label propagation. In *Thirtieth AAAI Conference on Artificial Intelligence*.

Mauro Dragoni. 2015. SHELLFBK: an information retrieval-based system for multi-domain sentiment analysis. In *Proceedings of the 9th International Workshop on Semantic Evaluation, SemEval@NAACL-HLT 2015, Denver, Colorado, USA, June 4-5, 2015*, pages 502–509. The Association for Computer Linguistics.

Feifan Fan, Yansong Feng, and Dongyan Zhao. 2018. Multi-grained attention network for aspect-level sentiment classification. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing, Brussels, Belgium, October 31 - November 4, 2018*, pages 3433–3442. Association for Computational Linguistics.

Zhifang Fan, Zhen Wu, Xinyu Dai, Shujian Huang, and Jiajun Chen. 2019. Target-oriented opinion words extraction with target-fused neural sequence labeling. *north american chapter of the association for computational linguistics*.

Amir Feder, Katherine A Keith, Emaad Manzoor, Reid Pryzant, Dhanya Sridhar, Zach Wood-Doughty, Jacob Eisenstein, Justin Grimmer, Roi Reichart, Margaret E Roberts, et al. 2021. Causal inference in natural language processing: Estimation, prediction, interpretation and beyond. *arXiv preprint arXiv:2109.00725*.

Fuli Feng, Jizhi Zhang, Xiangnan He, Hanwang Zhang, and Tat-Seng Chua. 2021. Empowering language understanding with counterfactual reasoning. In *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*, pages 2226–2236, Online. Association for Computational Linguistics.

Song Feng, Jun Seok Kang, Polina Kuznetsova, and Yejin Choi. 2013. Connotation lexicon: A dash of sentiment beneath the surface meaning. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1774–1784, Sofia, Bulgaria. Association for Computational Linguistics.

Anjalie Field and Yulia Tsvetkov. 2020. Unsupervised discovery of implicit gender bias. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing, EMNLP 2020, Online, November 16-20, 2020*, pages 596–608. Association for Computational Linguistics.

Madelyn Glymour, Judea Pearl, and Nicholas P Jewell. 2016. *Causal inference in statistics: A primer*. John Wiley & Sons.

Riccardo Guidotti, Anna Monreale, Salvatore Ruggieri, Franco Turini, Fosca Giannotti, and Dino Pedreschi. 2018. A survey of methods for explaining black box models. *ACM Computing Surveys*.

Liu Guohang, Zhang Shi-bin, Tang Haozhe, Yang Lu, Jiazhong Lu, and Huang Yuanyuan. 2020. Easy data augmentation method for classification tasks. *active media technology*.

Ruidan He, Wee Sun Lee, Hwee Tou Ng, and Daniel Dahlmeier. 2018. Effective attention modeling for aspect-level sentiment classification. In *Proceedings of the 27th International Conference on Computational Linguistics*, pages 1121–1131, Santa Fe, New Mexico, USA. Association for Computational Linguistics.

Robins JM Hernán MA. 2020. *Causal Inference: What If*. Boca Raton: Chapman & Hall/CRC.

Qingnan Jiang, Lei Chen, Ruifeng Xu, Xiang Ao, and Min Yang. 2019. A challenge dataset and effective models for aspect-based sentiment analysis. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing, EMNLP-IJCNLP 2019, Hong Kong, China, November 3-7, 2019*, pages 6279–6284. Association for Computational Linguistics.

Diederik P Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*.

Virgile Landeiro and Aron Culotta. 2016. Robust text classification in the presence of confounding bias. In *Thirtieth AAAI Conference on Artificial Intelligence*.

Klas Leino, Emily Black, Matt Fredrikson, Shayak Sen, and Anupam Datta. 2018. Feature-wise bias amplification. *arXiv preprint arXiv:1812.08999*.

Qizhi Li, Xianyong Li, Yajun Du, and Xiaoliang Chen. 2021a. Iswr: An implicit sentiment words recognition model based on sentiment propagation. In *CCF International Conference on Natural Language Processing and Chinese Computing*, pages 248–259. Springer.

Zhengyan Li, Yicheng Zou, Chong Zhang, Qi Zhang, and Zhongyu Wei. 2021b. Learning implicit sentiment in aspect-based sentiment analysis with supervised contrastive pre-training. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 246–256, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Jian Liao, Suge Wang, and Deyu Li. 2019. Identification of fact-implied implicit sentiment based on multi-level semantic fused representation. *Knowledge-Based Systems*, 165:197–207.

Zhouhan Lin, Minwei Feng, Cícero Nogueira dos Santos, Mo Yu, Bing Xiang, Bowen Zhou, and Yoshua Bengio. 2017. A structured self-attentive sentence embedding. In *5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Conference Track Proceedings*. OpenReview.net.

Bing Liu. 2012. Sentiment analysis and opinion mining. *Synthesis lectures on human language technologies*, 5(1):1–167.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Michael Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Roberta: A robustly optimized bert pretraining approach. *arXiv: Computation and Language*.

Dehong Ma, Sujian Li, Xiaodong Zhang, and Houfeng Wang. 2017. Interactive attention networks for aspect-level sentiment classification. In *Proceedings of the Twenty-Sixth International Joint Conference on Artificial Intelligence, IJCAI 2017, Melbourne, Australia, August 19-25, 2017*, pages 4068–4074. ijcai.org.

Yulei Niu, Kaihua Tang, Hanwang Zhang, Zhiwu Lu, Xian-Sheng Hua, and Ji-Rong Wen. 2021. Counterfactual vqa: A cause-effect look at language bias. In

Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pages 12700–12710.

Judea Pearl. 1995. Causal diagrams for empirical research. *Biometrika*, 82(4):669–688.

Judea Pearl. 2009. *Causality*, 2 edition. Cambridge University Press.

Matthew E. Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer. 2018. Deep contextualized word representations. *north american chapter of the association for computational linguistics*.

Maria Pontiki, Dimitris Galanis, John Pavlopoulos, Harris Papageorgiou, Ion Androutsopoulos, and Suresh Manandhar. 2014. Semeval-2014 task 4: Aspect based sentiment analysis. *international conference on computational linguistics*.

Suhas Ranganath, Xia Hu, Jiliang Tang, Suhang Wang, and Huan Liu. 2018. Understanding and identifying rhetorical questions in social media. *ACM Transactions on Intelligent Systems and Technology (TIST)*, 9(2):1–22.

Alexander Rietzler, Sebastian Stabinger, Paul Opitz, and Stefan Engl. 2020. Adapt or get left behind: Domain adaptation through BERT language model finetuning for aspect-target sentiment classification. In *Proceedings of The 12th Language Resources and Evaluation Conference, LREC 2020, Marseille, France, May 11-16, 2020*, pages 4933–4941. European Language Resources Association.

Irene Russo, Tommaso Caselli, and Carlo Strapparava. 2015. Semeval-2015 task 9: Clipeval implicit polarity of events. *SemEval-2015*, page 443.

Kai Sun, Richong Zhang, Samuel Mensah, Yongyi Mao, and Xudong Liu. 2019. Aspect-level sentiment analysis via convolution over dependency tree. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing, EMNLP-IJCNLP 2019, Hong Kong, China, November 3-7, 2019*, pages 5678–5687. Association for Computational Linguistics.

Hao Tang, Donghong Ji, Chenliang Li, and Qiji Zhou. 2020. Dependency graph enhanced dual-transformer structure for aspect-based sentiment classification. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 6578–6588, Online. Association for Computational Linguistics.

Kaihua Tang, Mingyuan Tao, and Hanwang Zhang. 2021. Adversarial visual robustness by causal intervention. *arXiv preprint arXiv:2106.09534*.

Victor Veitch, Alexander D'Amour, Steve Yadlowsky, and Jacob Eisenstein. 2021. Counterfactual invariance to spurious correlations: Why and how to pass stress tests. *arXiv preprint arXiv:2106.00545*.

Kai Wang, Weizhou Shen, Yunyi Yang, Xiaojun Quan, and Rui Wang. 2020. Relational graph attention network for aspect-based sentiment analysis. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 3229–3238, Online. Association for Computational Linguistics.

Xiao Wang, Qin Liu, Tao Gui, Qi Zhang, Yicheng Zou, Xin Zhou, Jiacheng Ye, Yongxin Zhang, Rui Zheng, Zexiong Pang, et al. 2021. Textflint: Unified multilingual robustness evaluation toolkit for natural language processing. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing: System Demonstrations*, pages 347–355.

Yequan Wang, Minlie Huang, Xiaoyan Zhu, and Li Zhao. 2016. Attention-based lstm for aspect-level sentiment classification. In *Proceedings of the 2016 conference on empirical methods in natural language processing*, pages 606–615.

Zhao Wang and Aron Culotta. 2020. Identifying spurious correlations for robust text classification. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 3431–3440.

Zach Wood-Doughty, Ilya Shpitser, and Mark Dredze. 2018. Challenges of using text classifiers for causal inference. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing. Conference on Empirical Methods in Natural Language Processing*, volume 2018, page 4586. NIH Public Access.

Chunli Xiang, Junchi Zhang, and Donghong Ji. 2021. A message-passing multi-task architecture for the implicit event and polarity detection. *PloS one*, 16(3):e0247704.

Xiaoyu Xing, Zhijing Jin, Di Jin, Bingning Wang, Qi Zhang, and Xuan-Jing Huang. 2020. Tasty burgers, soggy fries: Probing aspect robustness in aspect-based sentiment analysis. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 3594–3605.

Hu Xu, Bing Liu, Lei Shu, and Philip Yu. 2019. BERT post-training for review reading comprehension and aspect-based sentiment analysis. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 2324–2335, Minneapolis, Minnesota. Association for Computational Linguistics.

Chen Zhang, Qiuchi Li, and Dawei Song. 2019. Aspect-based sentiment classification with aspect-specific graph convolutional networks. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing,*

EMNLP-IJCNLP 2019, Hong Kong, China, November 3-7, 2019, pages 4567–4577. Association for Computational Linguistics.

Mi Zhang and Tieyun Qian. 2020. Convolution over hierarchical syntactic and lexical graphs for aspect level sentiment analysis. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing, EMNLP 2020, Online, November 16-20, 2020*, pages 3540–3549. Association for Computational Linguistics.

Renjie Zheng, Junkun Chen, and Xipeng Qiu. 2018. Same representation, different attentions: Shareable sentence representation learning from multiple tasks. In *Proceedings of the Twenty-Seventh International Joint Conference on Artificial Intelligence, IJCAI 2018, July 13-19, 2018, Stockholm, Sweden*, pages 4616–4622. ijcai.org.

Jie Zhou, Qin Chen, Jimmy Xiangji Huang, Qinmin Vivian Hu, and Liang He. 2020a. Position-aware hierarchical transfer model for aspect-level sentiment classification. *Information Sciences*, 513:1–16.

Jie Zhou, Jimmy Xiangji Huang, Qin Chen, Qinmin Vivian Hu, Tingting Wang, and Liang He. 2019. Deep learning for aspect-level sentiment classification: survey, vision, and challenges. *IEEE access*, 7:78454–78483.

Jie Zhou, Jimmy Xiangji Huang, Qinmin Vivian Hu, and Liang He. 2020b. Sk-gcn: Modeling syntax and knowledge via graph convolutional network for aspect-level sentiment classification. *Knowledge-Based Systems*, 205:106292.

Jie Zhou, Junfeng Tian, Rui Wang, Yuanbin Wu, Wenming Xiao, and Liang He. 2020c. Sentix: A sentiment-aware pre-trained model for cross-domain sentiment analysis. In *Proceedings of the 28th international conference on computational linguistics*, pages 568–579.

COMMA-DEER: COmmon-sense aware Multimodal Multitask Approach for Detection of Emotion and Emotional Reasoning in Conversations

Soumitra Ghosh[1*], **Gopendra Vikram Singh**[1*], **Asif Ekbal**[1] and **Pushpak Bhattacharyya**[2]
[1]Department of Computer Science and Engineering, IIT Patna, India
[2]Department of Computer Science and Engineering, IIT Bombay, India
ghosh.soumitra2@gmail.com, {gopendra_1921cs15,asif}@iitp.ac.in, pb@cse.iitb.ac.in *

Abstract

Mental health is a critical component of the United Nations' Sustainable Development Goals (SDGs), particularly Goal 3, which aims to provide "good health and well-being". The present mental health treatment gap is exacerbated by stigma, lack of human resources, and lack of research capability for implementation and policy reform. We present and discuss a novel task of detecting *emotional reasoning (ER)* and accompanying *emotions* in conversations. In particular, we create a first-of-its-kind multimodal mental health conversational corpus that is manually annotated at the utterance level with emotional reasoning and related emotion. We develop a multimodal multitask framework with a novel multimodal feature fusion technique and a contextuality learning module to handle the two tasks. Leveraging multimodal sources of information, commonsense reasoning, and through a multitask framework, our proposed model produces strong results. We achieve performance gains of 6% accuracy and 4.62% F1 on the *emotion* detection task and 3.56% accuracy and 3.31% F1 on the *ER* detection task, when compared to the existing state-of-the-art model.

1 Introduction

Each year, mental illness costs the economy almost 12 billion working days. Mental diseases are predicted to cost the global economy $16 trillion by 2030 (Canady, 2018). Despite extensive evidence of the close association between mental health and nearly every major issue in development, people with mental illnesses are among those most at risk of being left out of development programmes. Promoting mental health and preventing mental illness should be part of larger efforts to meet the United Nations' Sustainable Development Goals.

Unlike object identification tasks, emotion perception is significantly impacted by personal bias, cultural backgrounds, and contextual information (such as the environment). Emotional information can function as an authoritative inner voice in those who suffer from anxiety, despair, guilt, or fear on a daily basis (Caprara and Cervone, 2000). Normal individuals can suppress or remove the mood influence, while clinical patients cannot stop the inner voice from being viewed as authoritative. In spite of contradictory empirical facts, someone may draw the conclusion that something is true based only on their emotional response. This process is known as *emotional reasoning* (Gangemi et al., 2013). For example,

- *I feel insecure about my wife, which means she must be cheating on me.*
- *I feel afraid, so my neighbors must have been spying on me.*

While automatic emotion recognition in conversations can facilitate developing intelligent systems, such as empathetic chatbots, customers' feedback assessment, etc., automatic identification of emotional reasoning from conversations can help to perceive the mental states of the involved persons and understand latent vulnerability factors that often raise the likelihood of self-harm (such as suicide); thus helping to determine preventive efforts.

A significant increase in tele-health usage seeking mental health services was observed during the peak of the COVID-19 pandemic (Koonin et al., 2020). Despite advancements in certain nations, persons with mental illnesses frequently face human rights abuses, discrimination, and stigma. Automated systems based on natural language processing (NLP) approaches can be incorporated into digital interventions, particularly web and smartphone apps (chatbot like WoeBot[1], virtual assistant like Ellie[2], etc.) to provide early identification, take preventive measures and provide personalized

[*] The first two authors contributed equally to this work and are jointly the first authors.

[1] https://woebothealth.com/
[2] https://youtu.be/ejczMs6b1Q4

Proceedings of the 29th International Conference on Computational Linguistics, pages 6978–6990
October 12–17, 2022.

healthcare, more so in developing countries with a high population and minimal healthcare facilities.

The lack of suitable annotated data in the public domain is a serious impediment to mental health research utilising NLP methods. Also, there are certain nuances in mental health conversations (shown in Figure 1[3]), that makes them challenging to be addressed using the existent automated systems. The first set of images in the figure displays examples of how visual signals for current textual utterances can be found in past time steps. The second series of images depicts a patient's uneven bodily motions while discussing ER experiences at various points in the discussions. Furthermore, identifying ER utterances is challenging since the automated system must have commonsense reasoning ability to distinguish factual information from others.

To this end, we introduce a novel task of detection of emotion and emotional reasoning in conversations in a multitask setting. We develop a COmmonsense aware Multimodal Multitask Approach for Detection of Emotion and Emotional Reasoning (*COMMA-DEER*) in conversations at the utterance level. We create the *DEER* corpus (Detection of Emotion and Emotional Reasoning), which is a multimodal doctor-patient conversational dataset involving various common mental illnesses annotated with emotion and *ER* at the utterance level. We compare our proposed method to various existing state-of-the-art techniques to the presented dataset. Empirical evaluation and qualitative analysis show strong performances by our approach compared to the prior works on both tasks. We intend to make the code and data available to facilitate future research in this domain.

The main contributions of this work are:

1. We introduce a novel problem of joint detection of emotional reasoning and emotion in conversations exploiting the correlatedness between the two tasks.
2. This work introduces the first multimodal mental health conversational corpus, *DEER*, manually annotated with emotion, presence of emotional reasoning, speaker information, start and end timestamps at the utterance level.
3. We propose *COMMA-DEER*, a commonsense aware multimodal multitask system for detection of emotion and emotional reasoning utterances in a conversational setting.

[3]The images used are not subjected to copyright as the source videos have been made available for 'teaching purposes' and 'medical profession and allied scientific groups'.

The rest of the paper is organised as follows. Section 2 summarises some of the prior efforts in this subject. Following that, in Section 3, we discuss the dataset preparation in detail. In Section 4, we discuss our proposed methodology for multimodal multitask experiments. We explain the experiments, results and their outcomes in Section 5. Finally, in Section 6, we conclude our work and identify the scope of future work.

2 Related Work

Arntz et al. (1995)'s study was one of the first clinical investigations to show that the affect-as-information hypothesis might be used to sustain anxiety and mood disorders. The authors in (Meeten and Davey, 2011) found that the emotional reasoning phenomenon is common among clinical populations due to its adverse negative impact. The use of ER as a source for measuring risk was investigated by Beck and Haigh (2014), who discovered that it has significant clinical relevance both as a disorder maintenance factor and as a treatment target in cognitive therapy.

The research community has focused heavily on utterance and document level emotion recognition (Mohammad et al., 2018; Bostan and Klinger, 2020). Psychology research (Pantic et al., 2005; Aviezer et al., 2012) also points to the importance of considering multimodal information to build automated systems to perceive human emotion. Given the significance of mental health and its growing influence on society, researchers (Pham et al., 2016) are currently developing methods to precisely detect human emotion in the intention of developing mental health therapeutic solutions and better understand mental health issues.

Recent works (Alswaidan and Menai, 2019; Ghosh et al., 2022) have considered external knowledge resources (ConceptNet (Speer et al., 2017), SenticNet (Cambria et al., 2018), etc.) in building systems for emotion detection in various domains (conversations, suicide notes, etc.). In mental health conversations such as doctor-patient interactions, any factual information or presence of any grounded knowledge, such as a doctor's description of a patient's mental condition or characteristics of a particular illness (examples shown in Section 3.2.1), can also be exploited to infuse additional knowledge into the context of the dialogue.

Data for multi-modal emotion detection is challenging to collect and annotate, especially for low-

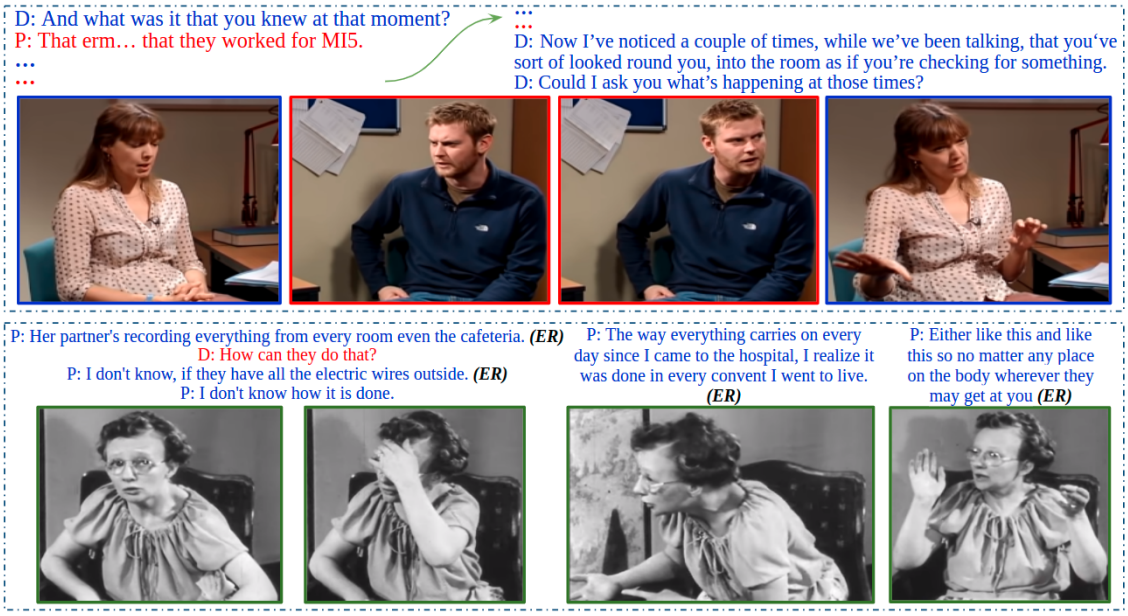

Figure 1: Sample instances from our *DEER* dataset, showing ER utterances and demonstrating how multimodality may capture various nuances in dialogues. ER: Emotional Reasoning.

resource emotions (e.g., disgust, surprise) that are encountered seldom in everyday life, which drives us to investigate this problem. Building on existing studies and the limitations of the prior works on related topics, we presented a manually annotated multimodal mental health conversational dataset and devised an automated approach leveraging commonsense knowledge to detect emotions and emotional reasoning utterances in conversations.

3 Dataset

In this section, we discuss the data collection and annotations for various attributes at the utterance level, such as speaker information, start and end timestamps, emotion, ER, and factual utterances.

3.1 Data collection and processing

We collected 30 doctor and patient interviews from YouTube[4]. Twenty of them are real interviews of different psychiatrists and various mental illness[5] patients. The rest of the ten interviews involves case studies/tutorial videos of conversations between the real psychiatrists and actors (posing as mental illness patients of various types of mental illness). Due to the sensitivity and stigma associated with mental health, readily available relevant data in the public domain is scarce. Hence, we decided to consider both the real and enacted doctor-patient conversations to create the dataset. Transcripts are manually generated for each video (wherever not available on YouTube) and marked with speaker information (*Doctor* or *Patient*) and start and end timestamps for each utterance. In a conversational video, an utterance is defined as a unit of speech bound by breaths or pauses (Hazarika et al., 2018).

3.2 Data Annotation

The annotations for the ER and emotion classes are performed at the utterance level by three annotators (one undergraduate student from the computer science discipline and two doctoral researchers from the computational linguistics discipline).

3.2.1 ER and Emotion Annotations

The annotations for ER labels were left to the annotators' discretion, based on their understanding of the phenomena of ER. Utterances were labeled as ER if they were detected as an obvious result of emotional reactions (inaccurate emotional truths) that directly contradicted any objective and/or perceptual realities. Such statements either lack counter empirical evidence or cannot be supported by common sense reasoning or common knowledge. Also, the doctor's interaction with the patient in the conversations provided essential clues and hints for ER to the annotators. When hearing a patient's emotional justification, doctors frequently express astonishment and are perplexed

[4]Some of the video ids are: *ZB28gfSmz1Y*, *8gDkFX4wprI*, *GGVYRxxsvEU*, *f744UFJSuog*

[5]*Psychosis, Schizophrenia, Paranoid Delusions, Delirium, Obsessive Compulsive Disorder, etc.*

or shocked. The annotators could easily identify strong candidate utterances for ER from the doctors' expressions, comments, vocal tone, etc. This can be understood from the below conversation snippet between a doctor and a patient, taken from our introduced DEER dataset.

Doctor: *Well you know, the other day you told me about the government being concerned in some way there's all this* [Emotion: Others]
Patient: *yes, and because the government doesn't keep the law* [Emotion: Others] –> ER
Patient: *the government is not in a harmony of the truth* [Emotion: Others] –> ER
Doctor: *but do you mean to say that the government prevents people from fulfilling the law?* [Emotion: Surprise]

Each utterance is marked with one of Ekman's (Ekman, 1992) six basic emotions: anger, disgust, fear, joy, sadness, and surprise. The necessity to add another class arose when the annotators encountered instances that have no emotion or some non-neutral emotion that do not fall into the scope of Ekman's basic emotions. We name this class as *others*. Table 1 shows the data distribution across the various emotion classes. We observe that most of the emotionally charged utterances are from patients' utterances, and their predominant emotions are anger, fear, and sadness. The majority of the doctor's utterances bear the surprise emotion as more than not; the doctors are puzzled or in disbelief on hearing the patients' emotional reasoning. We also observe that the dataset has an over-representation of the *others* class.

Figure 2 shows a snippet from the dataset, depicting ER and various associated emotions in different utterances from a doctor and patient's dyadic conversation. We show the data distribution over the emotion classes and ER categories for both doctors and patients in Table 2. We obtained 743 ER utterances from the total of 3,753 annotated utterances. An example of factual utterances from the dataset is shown below:

- 60-year-old professional woman with auditory and olfactory symptoms that feature in her persecutory delusions.

Inter-rater Agreement: We compute the Fleiss-Kappa (κ) score for the overall inter-rater agreement (Spitzer et al., 1967), as it is a popular choice when more than two raters are involved. The Emotion and ER tasks yielded scores of 0.75 and 0.83,

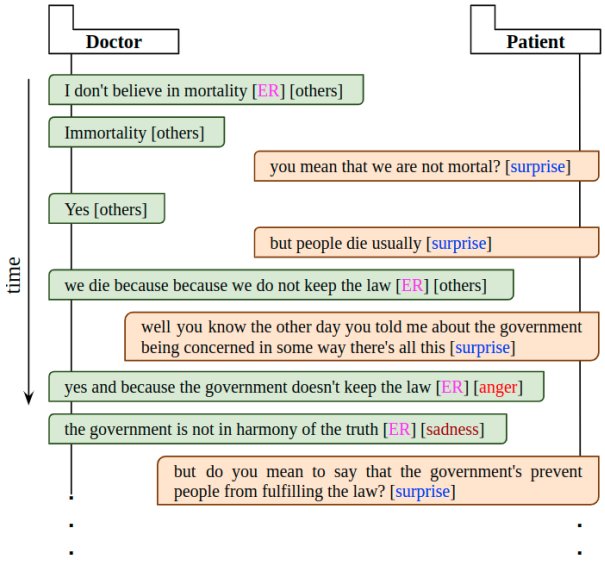

Figure 2: Sample conversation excerpt from the annotated *DEER* dataset.

Classes	Instances	κ
Anger	171	0.71
Disgust	69	0.65
Fear	163	0.74
Joy	124	0.90
Sadness	494	0.82
Surprise	174	0.58
Others	2558	0.86

Table 1: Emotion distribution.

	Doctor	*Patient*	**Total**
Emotion	217 [5]	978 [351]	1195 [356]
Others	1250 [42]	1308 [345]	2558 [387]
Total	1467 [47]	2286 [696]	3753 [743]

Table 2: Distribution of Emotion utterances and ER for doctor and patients. Values in the brackets indicate the ER counts.

respectively. According to the definition of the Fleiss' Kappa statistic (Landis and Koch, 1977), the obtained inter-rater reliability is considered to be 'almost perfect agreement' for the ER task and 'substantial agreement' for the emotion task. We also show the average per-class agreement among the annotators for each emotion in Table 1.

4 Methodology

This section formalizes our task objective and discusses our proposed *COMMA-DEER* approach. Figure 3 illustrates the general architecture of our proposed approach.

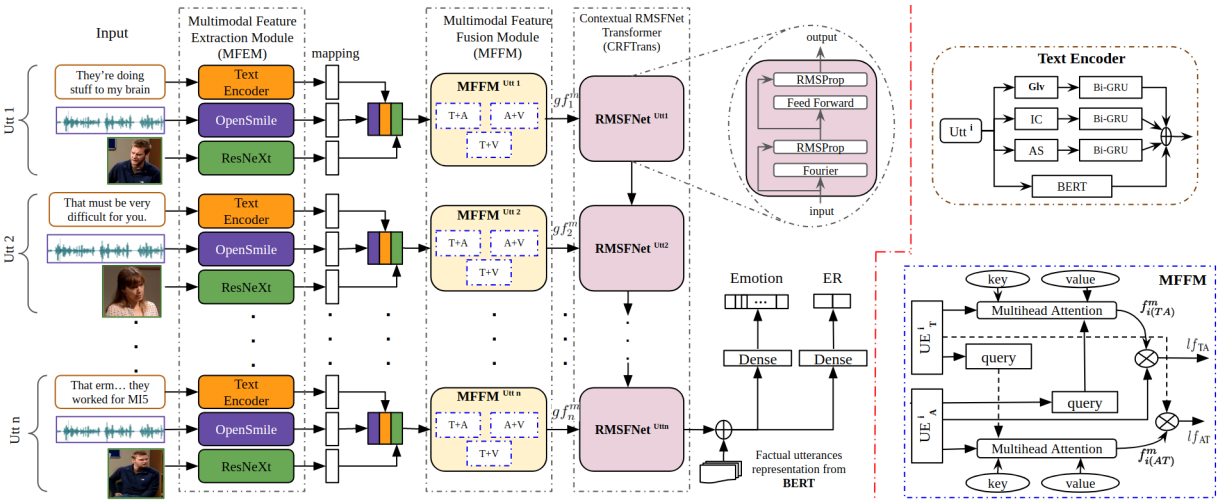

Figure 3: Architecture of the proposed *COMMA-DEER* approach. The model takes multi-modal inputs at the utterance-level and for a particular target utterance, it outputs the presence or absence of ER and the associated.

4.1 Task Definition

Our task involves a dyadic interaction between two speakers, a doctor and a patient, in which utterances are exchanged asynchronously. The objective is to identify emotion and the existence of emotional reasoning in the utterances. Let $U_t^m = (u_1^m, u_2^m, .., u_n^m)$ denote a conversation, $E_t^m = (e_1^m, e_2^m, ..., e_n^m)$ represent the corresponding emotion labels for each utterance in the conversation and $ER_n^m = (er_1^m, er_2^m, ..., er_n^m)$ depicts the presence or absence (0 or 1) of ER at the utterance level. In the mth dialogue, n refers to the total number of utterances. The purpose of this task is to maximize the value of the following function:

$$\underset{\theta}{argmax}(\Pi_{i=0}^{m}\Pi_{j=0}^{n}P(e_j^i, er_j^i|u_j^i, u_{j-1}^i, ..., u_1^i; \theta))$$

where $u_{j-1}^i, ..., u_1^i$ represents the previous set of utterances ordered temporally and u_j^i is the current (target) utterance whose emotion label (e_j^i) and presence of emotional reasoning (er_j^i) is to be predicted. P is the log likelihood function and θ denotes the model parameters to be optimized.

4.2 COmmon-sense aware Multimodal Multitask Approach for Detection of Emotion and Emotional Reasoning (*COMMA-DEER*)

Our proposed approach can be considered a pipeline of 4 primary modules: a). Multimodal Feature Extraction Module (MFEM), b). Multimodal Feature Fusion Module (MFFM), c). Contextual RMS-Fourier Transformer (CRFTrans), and, d). Task-specific Layers.

4.2.1 Multimodal Feature Extraction Module (MFEM)

Text Encoder: For an automated system to be able to identify *ER* in a running conversation, it requires a significant capability to perform commonsense reasoning to comprehend situations that relate to the real world or which are non-factual-based. We generate textual features from the utterances from multiple encoding sources. First, we generate a feature vector using a pre-trained Bidirectional Encoder Representations from Transformers (BERT) (Devlin et al., 2019) due to its strong ability to learn context-sensitive information and its ability to generalize on various downstream tasks. Next, we generate three more feature vectors (as shown in Figure 3) for the same utterance using multiple common-sense knowledge resources: GloVe (Pennington et al., 2014), IsaCore (Cambria et al., 2014) and AffectiveSpace[6] (Cambria et al., 2015). The IsaCore and AffectiveSpace are vector spaces[7], each providing 100 dimension embeddings for the most frequent occurring words in English. This setup is loosely motivated by the work in (Ghosh et al., 2022), except the incorporation of the BERT encoder. Since we are dealing with a relatively smaller supervised dataset, BERT, which is trained on a large scale corpus enables to generate feature-rich contextual representations of the input.

A similar approach of commonsense knowledge infusion was presented by Ghosh et al. (2022). Unlike (Ghosh et al., 2022), who only exploited the

[6] https://sentic.net/downloads/

[7] In Section A.3 of the **Appendix**, we present a broad discussion of the knowledge resources.

uni-gram word vectors, we extracted the uni-gram, bi-gram, and tri-gram vectors for the tokens of our utterances. Finally, the generated resource-independent sentence vectors are linearly concatenated and passed on to the next module.

Audio Encoder: For acoustic feature extraction, we utilise openSMILE (Eyben et al., 2010). openSMILE can extract Low-Level Descriptors (LLD) and manipulate them using various filters, functionalities, and transformations.

Video Encoder: Rich emotional indications are provided by facial expressions and the visual environment. We use 3D-ResNeXt[8] (Hara et al., 2018) to capture the facial expressions and visual surroundings from the utterance video, which provides rich emotional indicators.

4.2.2 Multimodal Feature Fusion Module (MFFM)

The separate feature vectors from the three modalities are passed through separate dense layers (mapping) to make them the same length. Instead of directly concatenating the features from the multiple modalities and use it for classification, we enhance the feature vector of each modality in a pair-wise approach using the features from the other modalities. For example, to enhance the textual features (${UE^m}_{\text{T}}$) with the help of the supporting acoustic information, we augment the acquired acoustic feature vectors (${UE^i}_{\text{A}}$) to the text representation using the multi-head attention technique. We do the same for the acoustic feature enhancement (${UE^m}_{\text{A}}$) using the textual features (${UE^m}_{\text{T}}$). Likewise, we do the same for the two other paired modalities: text-video and audio-video. So, for the text-audio scenario, we obtain $f^m_{i(TA)}$ and $f^m_{i(AT)}$ as per eq. 7.

$$Key^m_i = W_{K^m_i}\gamma_x + b_{K^m_i} \quad (1)$$

$$Value^m_i = W_{V^m_i}\gamma_x + b_{V^m_i} \quad (2)$$

$$Query^m_i = W_{Q^m_i}\gamma_x + b_{Q^m_i} \quad (3)$$

$$Att(Query, Key, Value) = softmax(\frac{QK^{tra}}{\sqrt{d}})V \quad (4)$$

$$head^m_i = Att(Query^m_i, Key^m_i, Value^m_i) \quad (5)$$

$$Multihead^m_i = (head_1, head_2, ..., head_{num})^m_i \quad (6)$$

$$f^m_{i(ab)} = Multihead(UE_{\text{a}}, UE_{\text{b}}, UE_{\text{b}})^m_i; \quad (7)$$

Here, γ is the input state vector, and $W_{K^m_i}$, $W_{V^m_i}$, and $W_{Q^m_i}$ are the corresponding weight matrices to transform γ into the Query, Key, Value vectors of the ith utterance in the mth conversation. We use Xavier-norm to initialize the parameters. Here, {a,b} ϵ {text (T), audio (A), video (V)}. *tra* denotes matrix transpose operation.

We design a gating mechanism to filter out noise[9] and as well as attend to features that are well represented in some modalities but not in the others[10]. The gating ($\bigotimes$) operation can be realized from the equations 8, 9 and 10. The equations relate to the gate on top of the MFFM module as shown in Figure 3 (component enclosed with blue dotted line). The sigmoid operation acts a forgetting mechanism for a particular modality (say T) based on the current input of another modality (say A) and vice-versa. The following dot product calculates the similarity measure between the resultant output (for T) from the previous step and the actual features of the other modality (of A). A tanh activation is applied on the outputs for non-linearity. This operation occurs for each pair-wise modality (T+A, T+V, A+V), thus producing a set of six features for each utterance which we linearly concatenate to produce an enriched feature vector (eq. 11).

$$g^m_{(TA)i} = sigmoid(W_g f^m_{(TA)i} + V_g {UE_{\text{A}}}^m_i + b_g) \quad (8)$$

$$P^m_{(TA)i} = dot(g_{(TA)}, f_{(TA)})^m_i \quad (9)$$

$$lf^m_{i(TA)} = tanh(W_{lf} P^m_{(TA)} + U_{lf} {UE_{\text{A}}}^m_i + b_{lf}) \quad (10)$$

$$gf^m_i = [lf_{\text{TA}}; lf_{\text{AT}}; lf_{\text{TV}}; lf_{\text{VT}}; lf_{\text{AV}}; lf_{\text{VA}}]^m_i \quad (11)$$

where $W_g, V_g, b_g, W_{lf}, V_{lf}, b_{lf}$ are the weight matrices that get updated during training. We use xavier-norm to initialize the parameters. [;] denotes concatenation.

4.2.3 Contextual RMS-Fourier Transformer (CRFTrans)

To model the contextual information among the utterances, we develop Contextual RMS-Fourier Transformer (*CRFTrans*). It uses the Fourier Transformer (FNet) encoder, originally introduced by Lee-Thorp et al. (2021), which is a simpler yet efficient method compared to regular transformer

[8] https://github.com/kaiqiangh/extracting-video-features-ResNeXt

[9] such as poor video/audio quality

[10] (for example, some clips are too small to generate the audio clips but text is available

Modality	*Single-Task*		*Multitask*	
	$F1^{ER}$	$F1^{Emo}$	$F1^{ER}$	$F1^{Emo}$
T	62.78	56.11	64.79	60.78
A	57.76	54.98	60.43	57.91
V	52.16	51.51	54.11	53.67
T+V	64.97	61.11	66.79	64.55
T+A	66.18	62.41	68.91	65.72
A+V	61.66	58.54	63.94	62.59
T+V+A	69.28	63.17	71.82	66.91
[T+V+A]$_{+CS}$	**72.14**	**66.33**	**73.44**	**70.14**

(a) Results of the proposed *COMMA-DEER* method on various modalities. Values in bold are the maximum scores attained. CS: common-sense.

Models	Emotion		ER	
	F1 (%)	Acc. (%)	F1 (%)	Acc. (%)
Singletask baselines				
bc-LSTM (Poria et al., 2017)	58.41	59.63	60.15	60.87
CMN (Hazarika et al., 2018)	62.64	64.52	68.17	69.31
DialogueRNN (Majumder et al., 2019)	63.92	64.16	68.67	70.13
Multitask baselines				
MTL-BERT (Peng et al., 2020)	62.33	63.51	65.61	65.91
CMSEKI (Ghosh et al., 2022)	64.14	66.31	69.88	71.31
COMMA-DEER* *(proposed)*	**70.14**	**70.93**	**73.44**	**74.62**

(b) Results from our proposed model and the various baselines. Values in bold are the maximum scores attained. Results marked with a * are statistically significant above the best performing CMSEKI baseline by the Student's t-test ($p < 0.05$).

Table 3: Results of the baselines and the proposed *COMMA-DEER* approach on the *DEER* dataset.

encoder (Vaswani et al., 2017). The self-attention sublayers in the transformer encoder are replaced with simple linear transformations that 'mix' the input tokens to create the FNet. Both the sequence dimension (seq) and the hidden dimension (hid) are transformed using 1D Fourier transforms (F): $\mathbb{R}(F_{\text{seq}}(F_{\text{hid}}(x)))$

One distinctive modification that we do in the FNet model is to use RMS-layer norm (Zhang and Sennrich, 2019) instead of the normal layer norm. The RMSNorm simplifies LayerNorm by removing the mean-centering operation or normalizing layer activations with RMS (Root Mean Square) statistic ($RMS(a) = \frac{1}{n}\sqrt{\sum_{i=1}^{n} a_i^2}$). To the best of our knowledge, we are the first to use RMSFNet as a contextual information learning module in conversations. Each utterance from the MFFM module (uf_i^m) in the input sequence is passed through the *CRFTrans* unit. Each passing utterance acts as a context to the next utterance up to the target utterance. So, $UF_i^m = [uf_1^m, uf_2^m, ..., uf_{n-1}^m] \in \mathbb{R}^{n-1,d}$ acts as the context for uf_n^m and its subsequent output from *CRFTrans* is passed for classification to the task-specific layers.

4.2.4 Task-specific layers:

We extract additional features exploiting the factual utterances (grounded knowledge) in the conversations to enhance the fused feature output of the CRFTrans module. We use a BERT encoder to create contextual features from the factual utterances associated with a dialogue to which the current [context+target] utterances belong. We linearly concatenate the BERT output with the *CRFTrans* output and feed it to two task-specific dense layers that capture emotion and ER identification task-specific characteristics. The output of the task-specific dense layers is sent to two output dense layers, which serve as output classification layers.

Calculation of loss: We adopt a principled approach to calculate the loss of our multitask approach that considers the homoscedastic uncertainty (Kendall et al., 2018) of each task while weighing multiple loss functions.

$$L = \sum_{\omega} W_\omega L_\omega \tag{12}$$

Here, ω denotes the two tasks, emotion, and emotional reasoning detection. The weights (W_ω) are updated using back-propagation for specific losses for each task. We use the categorical cross-entropy loss and binary cross-entropy loss for the emotion and ER tasks.

5 Experiments and Results

In this section, we discuss the experiments performed, and present the results and analysis.

5.1 Experimental Setup

We evaluate our proposed approach against five state-of-the-art systems: Hierarchical Attention Networks (bc-LSTM) (Poria et al., 2017), Conversational Memory Network (CMN) (Hazarika et al., 2018), DialogueRNN (Majumder et al., 2019), Mulitask BERT (MT-BERT) (Peng et al., 2020), and Cascaded Multitask System with External Knowledge Infusion (CMSEKI) (Ghosh et al., 2022). As class imbalance problem persists in the dataset, we performed 5-fold cross-validation experiments and used the macro-F1 metric to evaluate the efficacy of our method against the various baselines. We discuss the details of the baselines and the hyperparameters for our experiments in the **Appendix** (Sections A.1 and A.2).

5.2 Results and Analysis

We investigate the importance of multimodal features for our task and also investigate the role of common-sense infusion in the learning process. Table 3a presents the results for different combinations of modes used by *COMMA-DEER* on *DEER* dataset. Best performances are obtained from the trimodal setup, followed by the bimodal and unimodal networks. Textual modality performs better than the audio and visual modalities when considered alone, which may be due to the presence of lesser noise in texts compared to audio-visual sources. Our observations are consistent with the previous findings (Hazarika et al., 2018) on comparable tasks. Also, we observe that in all the scenarios, the multitask systems outperformed the single-task variants.

Comparison with Prior Works: For a comprehensive evaluation of our proposed approach, we consider various state-of-the-art systems as baselines and observe (from Table 3b) that the proposed *COMMA-DEER* system commendably outperforms all of the baseline systems on both the tasks of emotion and ER detection. Also, we observe that the multitask systems (MTL-BERT and CMSEKI) obtain better scores than the single-task setups stressing the importance of learning the two associated tasks jointly. More so, the CMSEKI system which leverages common-sense knowledge in its training process, performs overall best among the considered baselines. However, our proposed COMMA-DEER approach attains strong performance improvements of 6% accuracy (Acc.) and 4.62% F1 on the emotion detection task and 3.56% accuracy and 3.31% F1 on the ER detection task, when compared to CMSEKI model.

Ablation Experiments: To examine the importance of the modules in *COMMA-DEER*, we remove the constituent components, one at a time, and report the results in Table 4. Specifically, we conduct two ablation experiments: first, we replace RMSFnet with FNet ($[T+V+A]_{\text{-RMS}}$) in CRFTrans, and, second, we employ linear concatenation, replacing the proposed MFFM mechanism, to fuse multimodal features ($[T+V+A]_{\text{-MFFM}}$). We observe notable fall in scores when either of these modules are removed from the *COMMA-DEER* framework, especially when we remove the MFFM module.

Varying Context Length: We trained *COMMA-DEER* for the following context lengths (ψ): 0, 2, 4, 5, 6 and 8. The obtained scores are depicted in Figure 4. 0 indicates no context and the target utterance is given as input to the model only. By increasing the number of past utterances, we see a consistent increase in performance. We obtain the best results when the ψ is set as 5. On analysis of some ER cases with context length 5, we observed frequent topic drifting when context length 5, which may explain the result. However, this more exhaustive analysis is needed, preferably on a larger dataset to obtain a concrete understanding of the observation. Adding more context does not give meaningful information; thus degrading performance due to model confusion.

Setup	$F1^{Emo}$ (%)	$F1^{ER}$ (%)
COMMA-DEER	**70.14**	**73.44**
$[T+V+A]_{\text{-RMS}}$	67.11 (-3.03)	70.24 (-3.20)
$[T+V+A]_{\text{-MFFM}}$	65.78 (-4.36)	68.83 (-4.61)

Table 4: Ablation experiment's results. % fall in scores are shown in brackets.

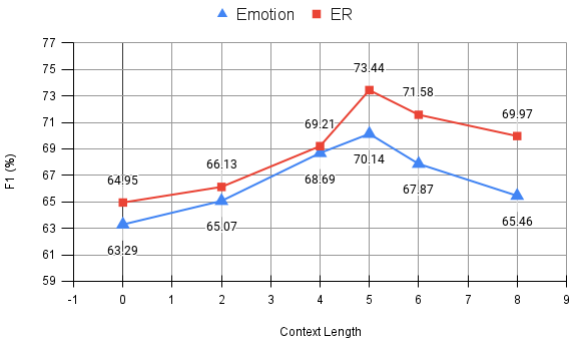

Figure 4: Graphical depiction of results of *COMMA-DEER* on varying context length.

5.2.1 Qualitative Analysis

For comprehensive evaluation of the performance of the proposed method and the considered baselines on the *DEER* dataset, we perform comparative analysis of predictions from these models on several test cases. The first example in Table 5 shows the ability of our model to correctly identify an ER utterance and also the associated emotion. On the contrary, both the CMSEKI and DialogueRNN misclassifies both the emotion and ER labels. In the second example too, we observe that our model classifies both the labels for emotion and ER correctly, unlike the CMSEKI (partially correct) and DialogueRNN (fully incorrect). Here, the target utterance is from the doctor and there is a context change in that utterance itself. Previous utterances taken as context do not entail the current utterance,

Examples	*COMMA-DEER*	CMSEKI	DialogueRNN
D: Right, ok. And have you seen much of your parents recently? *P*: Yeah. I thought about going back there but I, I don't want to drag them into this. *D*: Right. *P*: **I don't want MI5 knowing about them.**	anger ER	others Non-ER	others Non-ER
P: Yes. But in the building? No, I don't think so. As long as I pay my rent, there shouldn't be any problems. I paid my rent on Friday. *D*: **So what happened at work?**	others Non-ER	sadness Non-ER	anger ER
P: You know, jealous talk. *D*: But it was only in your apartment that you would hear these voices? *P*: Yes. *D*: Who was talking? What were they saying? *P*: **If I was doing something, lets say if I was looking for something, they'd taunt me, "she can't see it". They'd know that I was looking for something.**	fear ER	others ER	others Non-ER
P: I wasn't hungry, and I wasn't eating, because it felt tight here. *D*: You also told me about some gas... *P*: Yes, there was a smell coming from the trash chute and the air vents, always at night. *D*: What caused it? *P*: **It was some sort of gas, but I'm not sure what.**	others Non-ER	others Non-ER	others ER
P: As I said: Tuesday, Wednesday, Thursday and Friday and so he came again and he had a mask on and he'd already put something on my pillow. It's difficult to explain. A head, an animal head! To scare me! **I was laying there wondering what was on the pillow next to me.**	surprise Non-ER	others Non-ER	others Non-ER

Table 5: Sample predictions from the best performing baselines and the proposed *COMMA-DEER* approach. Labels highlighted in blue signifies correct predictions and that in red signifies mis-classifications. Target utterances are highlighted with bold font. D: Doctor; P: Patient.

which seems to cause the problem for existing systems. Similarly, in the third example, the CMSEKI and the *COMMA-DEER* predicts the ER label correctly, whereas the correct emotion label was only predicted by our proposed model. Looking at the misclassifications of the baseline systems, we observe a kind of biasness when it comes to the prediction of 'Non-ER' or 'others' class. These pair seems to occur more frequently among the different variations of the misclassifications. We also examined some of the error cases where our proposed method produced partially incorrect output (in example 4) or fully wrong labels (example 5). In example 4, although the three systems correctly predicted the emotion class as 'others', only the DialoueRNN model managed to predict the utterance as an ER. This may be attributed to its strong ability to model inter-speaker dependencies in the conversations, which is not available in *COMMA-DEER* or CMSEKI. The 5th example was misclassfied by all the three systems for both the tasks. Here, the context as well as the target utterance comes from a single speaker (patient). Our proposed method predicted 'surprise' for the utterance where the actual annotated emotion was 'fear'. We believe that the predicted emotion is not unreasonable as there is a possibility for 'surprise' too, in which case, this calls for the scope of extending this work with multilabel emotions, which are also coherent in a real-world conversational scenario.

6 Conclusion

This study presented a novel multimodal multitask system for the detection of emotion and ER in conversations. It also contributes towards mitigating the problem of scarce availability of annotated corpora by introducing a manually annotated multimodal mental health conversational corpus, *DEER*. Empirical and qualitative analysis suggests that (1). most of the existing state-of-the-art systems for conversational data perform poorly when it comes to comprehending ER in conversations; (2). the proposed system performs commendably well on both tasks and attains notable improvements from existing comparable methods; (3). performance on the ER detection task is considerably improved when we simultaneously learn the correlated task of emotion recognition.

Manual observation of the dataset shows that the doctor's responses to the patients' ER usually follow a pattern, such as responses of bewilderment and denouncement. We believe that knowing the responses of the doctor in the future time steps may significantly improve the performance for the detection of ER, which subsequently would enable detection of early warning signs of more serious mental illnesses. In future work, we want to extend this study in the above-mentioned directions as knowledge of the cognitive biases induced by ER may lead to novel therapeutic approaches.

Ethical Consideration

This work develops a resource from publicly available videos of doctor-patient interactions from YouTube. We followed the data usage restrictions and did not violate any copyright issues as the source videos have been made available for 'teaching purposes' and 'medical profession and allied scientific groups'. This study was also evaluated and approved by our Institutional Review Board (IRB). We shall make the code and data available for research purposes (on acceptance), through appropriate data agreement procedure. The findings reported here have been obtained from a small dataset of doctor-patient conversations, which may not accurately represent the phenomenon in all psychopathological disorders. We encourage further research and testing involving clinical participants. *The data is available at* https://www.iitp.ac.in/~ai-nlp-ml/resources.html#COMMA-DEER.

Acknowledgement

Asif Ekbal acknowledges the Young Faculty Research Fellowship (YFRF), supported by Visvesvaraya PhD scheme for Electronics and IT, Ministry of Electronics and Information Technology (MeitY), Government of India, being implemented by Digital India Corporation (formerly Media Lab Asia).

References

Nourah Alswaidan and Mohamed El Bachir Menai. 2019. KSU at semeval-2019 task 3: Hybrid features for emotion recognition in textual conversation. In *Proceedings of the 13th International Workshop on Semantic Evaluation, SemEval@NAACL-HLT 2019, Minneapolis, MN, USA, June 6-7, 2019*, pages 247–250. Association for Computational Linguistics.

Arnoud Arntz, Michael Rauner, and Marcel Van den Hout. 1995. "if i feel anxious, there must be danger": Ex-consequentia reasoning in inferring danger in anxiety disorders. *Behaviour research and therapy*, 33(8):917–925.

Hillel Aviezer, Yaacov Trope, and Alexander Todorov. 2012. Body cues, not facial expressions, discriminate between intense positive and negative emotions. *Science*, 338(6111):1225–1229.

Aaron T Beck and Emily AP Haigh. 2014. Advances in cognitive theory and therapy: The generic cognitive model. *Annual review of clinical psychology*, 10:1–24.

Laura Ana Maria Bostan and Roman Klinger. 2020. Token sequence labeling vs. clause classification for english emotion stimulus detection. In *Proceedings of the Ninth Joint Conference on Lexical and Computational Semantics, *SEM@COLING 2020, Barcelona, Spain (Online), December 12-13, 2020*, pages 58–70. Association for Computational Linguistics.

Erik Cambria, Jie Fu, Federica Bisio, and Soujanya Poria. 2015. Affectivespace 2: Enabling affective intuition for concept-level sentiment analysis. In *Proceedings of the Twenty-Ninth AAAI Conference on Artificial Intelligence, January 25-30, 2015, Austin, Texas, USA*, pages 508–514. AAAI Press.

Erik Cambria, Soujanya Poria, Devamanyu Hazarika, and Kenneth Kwok. 2018. Senticnet 5: Discovering conceptual primitives for sentiment analysis by means of context embeddings. In *Proceedings of the Thirty-Second AAAI Conference on Artificial Intelligence, (AAAI-18), the 30th innovative Applications of Artificial Intelligence (IAAI-18), and the 8th AAAI Symposium on Educational Advances in Artificial Intelligence (EAAI-18), New Orleans, Louisiana, USA, February 2-7, 2018*, pages 1795–1802. AAAI Press.

Erik Cambria, Yangqiu Song, Haixun Wang, and Newton Howard. 2014. Semantic multidimensional scaling for open-domain sentiment analysis. *IEEE Intell. Syst.*, 29(2):44–51.

VA Canady. 2018. Mental illness will cost the world $16 trillion (usd) by 2030. *Ment. Health Wkly*, 28:7–8.

Gian Vittorio Caprara and Daniel Cervone. 2000. *Personality: Determinants, dynamics, and potentials*. Cambridge University Press.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT 2019, Minneapolis, MN, USA, June 2-7, 2019, Volume 1 (Long and Short Papers)*, pages 4171–4186. Association for Computational Linguistics.

Paul Ekman. 1992. An argument for basic emotions. *Cognition & emotion*, 6(3-4):169–200.

Florian Eyben, Martin Wöllmer, and Björn W. Schuller. 2010. Opensmile: the munich versatile and fast open-source audio feature extractor. In *Proceedings of the 18th International Conference on Multimedia 2010, Firenze, Italy, October 25-29, 2010*, pages 1459–1462. ACM.

Amelia Gangemi, Francesco Mancini, and Philip N Johnson-Laird. 2013. Emotion, reasoning, and psychopathology. *Emotion and reasoning*, pages 44–83.

Soumitra Ghosh, Asif Ekbal, and Pushpak Bhattacharyya. 2022. A multitask framework to detect depression, sentiment and multi-label emotion from suicide notes. *Cogn. Comput.*, 14(1):110–129.

Kensho Hara, Hirokatsu Kataoka, and Yutaka Satoh. 2018. Can spatiotemporal 3d cnns retrace the history of 2d cnns and imagenet? In *2018 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2018, Salt Lake City, UT, USA, June 18-22, 2018*, pages 6546–6555. Computer Vision Foundation / IEEE Computer Society.

Devamanyu Hazarika, Soujanya Poria, Amir Zadeh, Erik Cambria, Louis-Philippe Morency, and Roger Zimmermann. 2018. Conversational memory network for emotion recognition in dyadic dialogue videos. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT 2018, New Orleans, Louisiana, USA, June 1-6, 2018, Volume 1 (Long Papers)*, pages 2122–2132. Association for Computational Linguistics.

Alex Kendall, Yarin Gal, and Roberto Cipolla. 2018. Multi-task learning using uncertainty to weigh losses for scene geometry and semantics. In *2018 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2018, Salt Lake City, UT, USA, June 18-22, 2018*, pages 7482–7491. Computer Vision Foundation / IEEE Computer Society.

Diederik P. Kingma and Jimmy Ba. 2015. Adam: A method for stochastic optimization. In *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*.

Lisa M Koonin, Brooke Hoots, Clarisse A Tsang, Zanie Leroy, Kevin Farris, Brandon Jolly, Peter Antall, Bridget McCabe, Cynthia BR Zelis, Ian Tong, et al. 2020. Trends in the use of telehealth during the emergence of the covid-19 pandemic—united states, january–march 2020. *Morbidity and Mortality Weekly Report*, 69(43):1595.

J Richard Landis and Gary G Koch. 1977. The measurement of observer agreement for categorical data. *biometrics*, pages 159–174.

James Lee-Thorp, Joshua Ainslie, Ilya Eckstein, and Santiago Ontanon. 2021. Fnet: Mixing tokens with fourier transforms. *arXiv preprint arXiv:2105.03824*.

Navonil Majumder, Soujanya Poria, Devamanyu Hazarika, Rada Mihalcea, Alexander F. Gelbukh, and Erik Cambria. 2019. Dialoguernn: An attentive RNN for emotion detection in conversations. In *The Thirty-Third AAAI Conference on Artificial Intelligence, AAAI 2019, The Thirty-First Innovative Applications of Artificial Intelligence Conference, IAAI 2019, The Ninth AAAI Symposium on Educational Advances in Artificial Intelligence, EAAI 2019, Honolulu, Hawaii, USA, January 27 - February 1, 2019*, pages 6818–6825. AAAI Press.

Frances Meeten and Graham CL Davey. 2011. Mood-as-input hypothesis and perseverative psychopathologies. *Clinical Psychology Review*, 31(8):1259–1275.

Saif M. Mohammad, Felipe Bravo-Marquez, Mohammad Salameh, and Svetlana Kiritchenko. 2018. Semeval-2018 task 1: Affect in tweets. In *Proceedings of The 12th International Workshop on Semantic Evaluation, SemEval@NAACL-HLT 2018, New Orleans, Louisiana, USA, June 5-6, 2018*, pages 1–17. Association for Computational Linguistics.

Maja Pantic, Nicu Sebe, Jeffrey F. Cohn, and Thomas S. Huang. 2005. Affective multimodal human-computer interaction. In *Proceedings of the 13th ACM International Conference on Multimedia, Singapore, November 6-11, 2005*, pages 669–676. ACM.

Yifan Peng, Qingyu Chen, and Zhiyong Lu. 2020. An empirical study of multi-task learning on BERT for biomedical text mining. In *Proceedings of the 19th SIGBioMed Workshop on Biomedical Language Processing, BioNLP 2020, Online, July 9, 2020*, pages 205–214. Association for Computational Linguistics.

Jeffrey Pennington, Richard Socher, and Christopher D. Manning. 2014. Glove: Global vectors for word representation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing, EMNLP 2014, October 25-29, 2014, Doha, Qatar, A meeting of SIGDAT, a Special Interest Group of the ACL*, pages 1532–1543. ACL.

Trang Pham, Truyen Tran, Dinh Q. Phung, and Svetha Venkatesh. 2016. Deepcare: A deep dynamic memory model for predictive medicine. In *Advances in Knowledge Discovery and Data Mining - 20th Pacific-Asia Conference, PAKDD 2016, Auckland, New Zealand, April 19-22, 2016, Proceedings, Part II*, volume 9652 of *Lecture Notes in Computer Science*, pages 30–41. Springer.

Soujanya Poria, Erik Cambria, Devamanyu Hazarika, Navonil Majumder, Amir Zadeh, and Louis-Philippe Morency. 2017. Context-dependent sentiment analysis in user-generated videos. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics, ACL 2017, Vancouver, Canada, July 30 - August 4, Volume 1: Long Papers*, pages 873–883. Association for Computational Linguistics.

Robyn Speer, Joshua Chin, and Catherine Havasi. 2017. Conceptnet 5.5: An open multilingual graph of general knowledge. In *Proceedings of the Thirty-First AAAI Conference on Artificial Intelligence, February 4-9, 2017, San Francisco, California, USA*, pages 4444–4451. AAAI Press.

Robert L Spitzer, Jacob Cohen, Joseph L Fleiss, and Jean Endicott. 1967. Quantification of agreement in psychiatric diagnosis: A new approach. *Archives of General Psychiatry*, 17(1):83–87.

Nitish Srivastava, Geoffrey E. Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. 2014. Dropout: a simple way to prevent neural networks from overfitting. *J. Mach. Learn. Res.*, 15(1):1929–1958.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, December 4-9, 2017, Long Beach, CA, USA*, pages 5998–6008.

Wentao Wu, Hongsong Li, Haixun Wang, and Kenny Qili Zhu. 2012. Probase: a probabilistic taxonomy for text understanding. In *Proceedings of the ACM SIGMOD International Conference on Management of Data, SIGMOD 2012, Scottsdale, AZ, USA, May 20-24, 2012*, pages 481–492. ACM.

Biao Zhang and Rico Sennrich. 2019. Root mean square layer normalization. In *Advances in Neural Information Processing Systems 32: Annual Conference on Neural Information Processing Systems 2019, NeurIPS 2019, December 8-14, 2019, Vancouver, BC, Canada*, pages 12360–12371.

A Appendix

A.1 Baselines

The following baseline methods are considered for the comprehensive evaluation of our proposed.

- bc-LSTM (Poria et al., 2017): For multimodal sentiment analysis, it is an LSTM-based network that extracts contextual information from video utterances.
- CMN (Hazarika et al., 2018): Using a gated recurrent unit on multimodal characteristics, this approach converts prior utterances of each speaker into memories. To capture inter-speaker dependencies, these memories are then combined using attention-based hops. Contextuality is discovered by combining memories using an attention-based technique.
- DialogueRNN (Majumder et al., 2019): This approach uses speaker, context, and emotion information from neighbouring utterances to represent the emotion of words in a dialogue. These variables are represented by three distinct GRU networks to keep track of the various speaker states.
- MT-BERT (Peng et al., 2020): We implement a multitask (MT) variant of BERT based on the architecture proposed by Peng *et al.* (Peng et al., 2020) for our two tasks detection of emotion and Emotional Reasoning.
- Cascaded Multitask System with External Knowledge Infusion (CMSEKI) (Ghosh et al., 2022): The CMSEKI system was introduced in the work that presented CEASE-v2.0 dataset, addressing the detection of depression, sentiment, and emotion, using commonsense knowledge. We adapted the CMSEKI system to address our *Emotion* and *ER* detection tasks.

Parameters	***COMMA-DEER***
Transformer Encoder	2 layers
Embeddings	1068
FC Layer	Dropout (Srivastava et al., 2014) = 0.3
Activations	*ReLU* for dense layers
Output	Sigmoid for Emotional Reasoning
Activations	Softmax for Emotion
Optimizer	Adam (Kingma and Ba, 2015) (lr = 0.003)
Batch	32
Epochs	30

Table 6: Hyper-parameters for our experiments.

A.2 Experimental Setting

We use PyTorch[11], a Python-based deep learning package, to develop our proposed model. We experiment with the base version of BERT imported from the huggingface transformers[12] package. We perform *grid search* to find the optimal hyper-parameters in Table 6. For openSMILE, voice normalization and voice intensity threshold are used to discriminate between samples with and without speech. Z-standardization is used for voice normalizing. ResNext has been pre-trained on Kinetics at 1.5 features per second and a resolution of 112. All experiments are carried out on an NVIDIA GeForce RTX 2080 Ti GPU. To account for the non-determinism of TensorFlow GPU operations, we present F1 scores averaged across five 5-fold cross-validation runs. We set the sequence length as 128 and report the results with context length = 5, as we observed best scores for this setup.

A.3 External Knowledge Sources

IsaCore: IsaCore (Cambria et al., 2014) is a vector space that preserves semantic and sentiment polarity based on the relationships between instances ('birthday party' and 'china') and concepts ('special occasion' and 'country') and affective labels. It is generated by using multidimensional scaling

[11] https://pytorch.org/

[12] https://huggingface.co/docs/transformers/index

AffectNet	IsA-pet	IsA-food	Arises-joy	...
dog	0.981	0	0.789	...
cupcake	0	0.922	0.910	...
songbird	0.672	0	0.862	...
gift	0	0	0.899	...
sandwich	0	0.853	0.768	...
rotten fish	0	0.459	0	...
lottery	0	0	0.991	...
...	...	...	...	...

Table 7: A snapshot of the AffectNet matrix.

to the knowledge base that results from combining Probase (Wu et al., 2012) (the biggest current taxonomy of common knowledge) with ConceptNet (Speer et al., 2017) (natural language-based semantic network of commonsense knowledge).

AffectiveSpace 2: AffectiveSpace 2 (Cambria et al., 2015) is an unique vector space model for concept-level sentiment analysis that enables reasoning by comparison on natural language ideas even when they are represented by highly dimensional semantic characteristics. This embedding space was generated by performing principal component analysis (PCA) on the AffectNet matrix representation, which is a semantic network in which common-sense concepts are linked to semantic and affective properties. A snapshot of the AffectNet matrix is shown in Table 7.

EmoMent: An Emotion Annotated Mental Health Corpus from two South Asian Countries

Thushari Atapattu[1], Mahen Herath[2], Charith Elvitigala[3], Piyanjali de Zoysa[4], Kasun Gunawardane[3], Menasha Thilakaratne[1], Kasun de Zoysa[3] and Katrina Falkner[1]

[1]School of Computer Science, The University of Adelaide, Australia
[2]Department of Computer Science & Engineering, University of Moratuwa, Sri Lanka
[3]University of Colombo School of Computing, Sri Lanka
[4]Department of Psychology, University of Colombo, Sri Lanka
email: thushari.atapattu@adelaide.edu.au

Abstract

People often utilise online media (e.g. Facebook, Reddit) as a platform to express their psychological distress and seek support. State-of-the-art NLP techniques demonstrate strong potential to automatically detect mental health issues from text. Research suggests that mental health issues are reflected in *emotions* (e.g. sadness) indicated in a person's choice of language. Therefore, we developed a novel emotion-annotated mental health corpus (*EmoMent*), consisting of 2802 Facebook posts (14845 sentences) extracted from two South Asian countries - Sri Lanka and India. Three clinical psychology postgraduates were involved in annotating these posts into eight categories, including *mental illness* (e.g. depression) and *emotions* (e.g. sadness, anger). EmoMent corpus achieved 'very good' inter-annotator agreement of 98.3% (i.e. % with two or more agreement) and Fleiss' Kappa of 0.82. Our RoBERTa based models achieved an F1 score of 0.76 and a macro-averaged F1 score of 0.77 for the *first task* (i.e. predicting a mental health condition from a post) and the *second task* (i.e. extent of association of relevant posts with the categories defined in our taxonomy), respectively.

1 Introduction

Mental health issues remain a leading cause for poor well-being and suicide. The World Health Organisation (WHO) indicates that 400 million people are affected by mental disorders such as depression, resulting in a cost of US$ 1 trillion per year from the global economy allocated for depression and anxiety disorders alone (WHO, 2019; James et al., 2018). Recent research using AI and NLP demonstrates strong potential to automatically detect mental health issues from digital footprints such that professionals could provide timely interventions and mental health resources to vulnerable persons. These data contain useful information to understand patients' distressed state of mind outside a traditional clinical environment.

Research suggests that mental health issues are reflected in the 'emotions' (e.g. sadness, anger) indicated in one's expression of language. Despite the popularity of research studies in detecting mental disorders using online data such as *Twitter* (Coppersmith et al., 2014, 2015; Cohan et al., 2018) and emotion modeling (Strapparava and Mihalcea, 2007; Mohammad et al., 2018; Demszky et al., 2020; Oberländer and Klinger, 2018), the automated identification of the *association between emotions and mental disorders* have largely being ignored, apart from a recent study (CEASE corpus (Ghosh et al., 2020)) that focused on the role of emotions on suicidal ideation.

Motivated by this, we introduce a novel, emotion-annotated mental health (*EmoMent*) corpus[1] using *Facebook* posts extracted from two South Asian countries - Sri Lanka and India. In South Asia, due to the lack of awareness of symptoms of mental illnesses and its associated stigma, people often do not seek professional help, resulting in many instances of mental disorders being left undiagnosed (Arora et al., 2016). However, since recently, these countries have demonstrated a tendency to use social media, particularly Facebook, to seek mental health help using private and public groups (e.g. *Psychology group* in Sri Lanka, *Indian Psychology Association*).

Depression and anxiety disorders are amongst the most common mental disorders worldwide (James et al., 2018; Black Dog Institute, 2020). Therefore, our dataset includes de-identifiable Facebook posts from individuals who have indicated a diagnosis of depression or anxiety, the disorder-

[1]dataset and the code is available on request for research purposes.

Proceedings of the 29th International Conference on Computational Linguistics, pages 6991–7001
October 12–17, 2022.

related issues they express including associated emotions, and their help-seeking behaviours from professionals and/or community. EmoMent consists of 2802 posts (14845 sentences) extracted from public Facebook groups dedicated to discuss mental health concerns in Sri Lanka and India. Three clinical psychology postgraduates were involved in the data annotation process. Their task was to read the entire post and assign one or more labels from a given set of eight categories (e.g. *mental illness, sadness, psychosomatic, irrelevant*) (Table 2). We have achieved 'very good' inter-annotator agreement of 98.3% (i.e. % with two or more rater-agreement) and Fleiss' Kappa of 0.82, while 0.90 and 0.74 of Kappa values were achieved on Sri Lankan and Indian datasets respectively, enabling a promising human agreement for computational modelling.

We fine-tuned BERT (Devlin et al., 2018) and RoBERTa (Liu et al., 2019) based deep learning models on the EmoMent corpus to predict the relevance of a post to a mental health condition (*first task)*, and to associate relevant posts with the categories defined in our taxonomy (*second task*) (see section 3.3). Our RoBERTa-based models achieved a F1 score of 0.76 for the *first task*, and a macro-averaged F1 score of 0.77 for the *second task*.

The novel contributions of our paper includes; 1) the development of the first emotion-annotated mental health corpus in English language 2) the development of the first *taxonomy* to annotate mental health conditions and emotions from Facebook data, and 3) the development and evaluation of deep learning models (RoBERTa) to predict the presence of mental conditions, emotions, and psychosomatic issues with 'good' performance. Additionally, our research contributed to the integration of knowledge from two domains - *mental health* and *emotion modelling* through various quantitative and qualitative analyses, in particular, low-resource languages such as Sinhala.

Currently, the diagnosis of a mental disorder is primarily based on the knowledge and experience of a professional, who arrives at a diagnosis subsequent to talking to a patient and/or care-givers. In this method, patients have to reflect on events that occurred in the past to help professionals diagnose their condition. Real-time experiences of patients, which is an important element for diagnosis and treatment plan, is not usually considered. The majority of online self-reflective posts on the other hand generate real-time, reliable data to uncover distressed states of mind at the time of occurring. Therefore, a corpus like EmoMent, developed from user-generated data allows practitioners to understand the mental states of patients beyond a traditional clinical interview. These automated identification of mental disorders or mental conditions,from user-generated content provides a useful tool for improving diagnosis and personalised treatment plans.

2 Related Work

People use language as a direct tool to express their feelings and emotions, providing a wealth of information to determine their emotional status and mental health conditions (Berry et al., 2017). Motivated by this, many datasets have been developed to support research in the two fields of: *Emotion Modelling* and *Mental Health Modelling*, using social media as one of the primary data sources. The existing datasets on *emotion modelling* are mostly based on two emotion taxonomies: Ekman's basic emotions (*fear*, *anger*, *joy*, *sadness*, *disgust*, and *surprise*) (Ekman, 1992), and Plutchik's Wheel of Emotions (*anger*, *anticipation*, *disgust*, *fear*, *joy*, *sadness*, *surprise* and *trust*) (Plutchik, 1980). Examples of emotion modelling datasets include (Strapparava and Mihalcea, 2007; Mohammad et al., 2018; Demszky et al., 2020; Oberländer and Klinger, 2018; Li et al., 2020; Appidi et al., 2020). The existing *mental health modelling* datasets are based on the problem domains such as suicidal attempts, self-injury, loneliness, depression, anxiety and Post Traumatic Stress Disorder. The focus of most existing datasets are limited to one or two problem domains, hindering the diagnosis capabilities of the AI models that they are based on. (Pirina and Çöltekin, 2018; Tadesse et al., 2019; Zirikly et al., 2019).

Despite the availability of numerous emotion modelling and mental health modelling datasets, there are certain limitations in almost all of these datasets. The datasets from these two research fields are independent of one another (i.e., non-interactive). The complementary integration of emotion and mental health modelling provide enhanced insights on a person's emotional and mental well-being, which is useful in assisting professionals to diagnose and personalise treatment plans. Additionally, almost all of the currently available datasets are based on resource-rich languages such

as English (Appidi et al., 2020), limiting the understanding of cultural aspects of language use in emotion and mental health modelling.

Despite the importance of jointly-modelling emotions and mental health, the availability of emotion-annotated mental health datasets are vastly limited. For example, the emotion-annotated mental health dataset of Ghosh et al. (Ghosh et al., 2020), CEASE, is specific to suicide notes. Motivated by this, we present a novel emotion-annotated mental health dataset based on Facebook data to facilitate joint-modeling of emotions and mental health conditions.

To propose baseline models from the constructed datasets, most previous studies in emotion, mental health, and emotion-annotated mental health domains have utilised recent advancements of deep learning techniques such as BERT, LSTMs and RNNs (Li et al., 2020; Appidi et al., 2020). For example, the CEASE dataset proposes an ensemble model using LSTM, CNN, and GRU (Ghosh et al., 2020). To adhere with this, we also leveraged the recent advancements in deep learning techniques using BERT and RoBERTa based models.

3 EmoMent Corpus

This section describes the development of the EmoMent corpus - data collection, data cleaning, taxonomy development, and data annotation.

3.1 Data Collection

We used the CrowdTangle tool[2] to collect Facebook posts that express mental health-related issues. CrowdTangle is a content discovery and social monitoring platform which provides an interface to access *public* Facebook pages and group posts. Their search interface contains filters such as 'Post type - photos, statuses', 'language', and 'time frame'. Our search filter parameters were 'account type' as groups and 'post type' as statuses. Our 'language' parameters were *Sinhala* and *English* while restricting 'geographical locations' to Sri Lanka and India. Due to the sparseness of recent data in constructing a reasonable size corpus for computational modelling, our search time frame was expanded to approximately nine years from 2012-01-01 to 2021-10-31.

CrowdTangle supports keyword, hashtag, or URL search, combining with boolean search operators such as AND, OR, NOT. Our data collection process utilised the *keywords* and *phrases* option after a consultation with a clinical psychologist. Our keywords and phrases included "depression", "anxiety", "stress", "I feel unhappy", and "I feel like ending my life". To search Facebook posts in Sinhala language, these keywords and phrases were translated into Sinhala (See Appendix A.2 'Data Extraction' for the full list of keywords and phrases).

We collected approximately 10,000 posts from Indian and Sri Lankan public Facebook groups. Each post includes metadata such as *Group Id*, *Group Name*, *Text Post*, *Post Created Time*, and Post Interaction (e.g. *Like, Love*) Count. The extracted metadata did not contain any personal identification details such as Facebook user name or user Id. Therefore, Facebook user anonymity was preserved. During our thorough filtering process, we did not find any mention of Facebook user names inside post contents other than sentences like "*please admin, approve this post etc.*". We recognised inherent demographic biases of data when the data extraction methodology disregards Facebook users' demographic information such as gender and age. We noticed a large amount of noise within the extracted Facebook data, resulting in difficulty in constructing a sufficiently large and demographically unbiased dataset.

3.2 Data Cleaning

Our data cleaning process included manually removing posts from inappropriate groups such as Facebook groups with adult content. We also excluded single sentence posts from the dataset using the NLTK tool[3] since it is challenging to perform meaningful NLP processing to predict emotions from a single sentence. We also removed transliterated posts and translated all Sinhala language posts into English using Facebook Language Translator[4]. Finally, we removed all the duplicate posts from the dataset. The data cleaning process resulted in a corpus of 2045 and 757 posts from Indian and Sri Lankan Facebook groups respectively (see Table1 for 'descriptive statistics' of the dataset).

3.3 Taxonomy Development

As discussed in section 2, the majority of research studies on emotion modelling rely on two popular taxonomies - Ekman's model (Ekman, 1992) and

[2]https://www.crowdtangle.com

[3]https://www.nltk.org/

[4]https://developers.Facebook.com/docs/graph-api/reference/v12.0/app/translations

Dataset	Sri Lankan	Indian	Full
Posts	757	2045	2802
Sentences (ST)	5827	9018	14845
ST per post	12.1	4.9	–
Words per post	188	93	–

Table 1: Descriptive Statistics of the filtered Facebook dataset

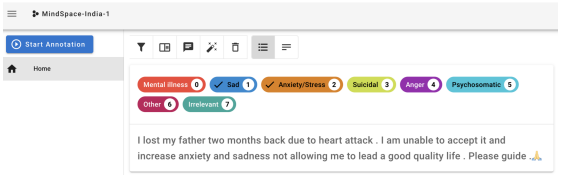

Figure 1: A screenshot of Doccano interface configured for annotators

Plutchik's 'Wheel of Emotions' (Plutchik, 1980). Researchers tend to adapt these models by adding new emotions (Demszky et al., 2020) or removing emotions. Therefore, we adapted three basic emotions from these two models - *fear*, *anger*, and *sadness* since empirical studies demonstrate that these three emotions are strongly associated with mental health issues. We also removed emotions such as *disgust* and *surprise* since they occurred infrequently in our selected data source.

Our taxonomy development process adopted 'open coding', a popular method in grounded theory to identify, describe or categorise phenomena found in qualitative data (Corbin and Strauss, 1990). Firstly, we manually classified a random sample of 50 Facebook posts into meaningful categories (known as *codes* (Miles and Huberman, 1994)). To start with, we used the 3 basic emotions - *fear*, *anger*, and *sadness* This analysis found additional mental states (e.g., *suicidal thoughts*, *loneliness*, and *addictions*) that are likely associated with mental health conditions. Secondly, we consulted a clinical psychologist to refine the codes until we reached agreement on a taxonomy that contained *codes* to annotate our dataset. After this consultation, we expanded the emotion of 'fear' with 'anxiety/stress' as these terms are used interchangeably in the Sri Lankan context. We also merged some codes due to their infrequent occurrence in the dataset (e.g., *loneliness*) which could result in data sparseness when modelling. Accordingly, three additional codes were introduced as listed below (see Appendix A.1 for complete definitions of taxonomy);

- **Mental illness**: Posts that mention a diagnosis or a treatment related to a mental illness.
- **Psychosomatic**: Posts on psychosomatic issues (e.g., fatigue, headaches) associated with an underlying mental condition.
- **Other**: Posts that express a maladaptive mental condition but do not belong to any of the previously defined categories (e.g., addictions, loneliness).

Finally, we introduced '**irrelevant**' category if none of the above-defined codes were usable to annotate a particular post. Table 2 shows the finalised set of categories, along with examples, from our dataset.

3.4 Data Annotation

We used three annotators to code the dataset. All of them were native Sri Lankans with masters-level experience in clinical psychology. They were recruited by distributing flyers within Psychology departments of three main universities and two higher educational institutes in Sri Lanka. They were employed as research assistants for 1.5 months and their time and effort were compensated based on the standard daily salaries in Sri Lanka. They were proficient in both Sinhala and English languages, and were familiar with Facebook mental health groups and cyber language. These annotators also had a sound understanding of South Asian culture and context. None of the annotators is an author of this paper.

The task of an annotator was to read the entire post provided through Doccano web interface[5] and assign codes based on the taxonomy (Table 2). Doccano is a popular web based, open source, text annotation tool. Figure 1 shows an example of Doccano interface we configured for annotators. Each post could have one or more codes. However, the 'irrelevant' code was not allowed to be used jointly with other codes. Our annotation instructions emphasised the importance of making evaluations based on the information explicitly found in a given post without making assumptions. (see Appendix A.1 for more information about the 'annotation guideline').

[5]https://github.com/doccano/doccano

Category	Example
Mental illness (MI)	I have been taking antidepressants since a long time, watching motivational videos, listening to relaxing music, but when things happen like a problem, my head is like a stone, why is that???
Sadness (SD)	She has a lot of sadness in her heart because of a past incident for a long time.. she says it's hard to forget no matter how she tries.. she says she cannot live without forgetting that incident.. She says she is living because she cannot die
Anxiety/Stress (AS)	I'm so mentally down I'm in a lot of problems. I'm a person who has suffered a lot since I was a kid. I've never been loved even because of my family problems. From mom to dad because they separated when they were young. I lost everything I loved. I still suffer from that.
Suicidal (SC)	I'm suffering from depression ☹Right now there are so many problems that are going on in my life. Sometimes I just want to end my life
Anger (AG)	Sometimes I think that I need to take revenge. Because revenge has been my addiction. If I don't take revenge, I'm in depression and so angry...I'm so afraid of myself because when I get angry I won't control and don't know what I have done.
Psychosomatic (PY)	How to get good night sleep at night in depression? Suffering from Insomnia from last 3 months
Other (OT)	I am studying and I feel lonely. Before some time when I worked I felt so excited and interested. But now no any interest and excitement
Irrelevant (IV)	Anyone can love you when the sun is shining, but in the storms is where you'll learn who truly cares about you...

*Note - Due to sensitivity of data, we report an excerpt of the post

Table 2: EmoMent Taxonomy and sample examples

4 Corpus Analysis

We constructed the EmoMent corpus by selecting posts which had two or more annotators agreeing on a category.

4.1 Corpus Statistics

Table 3 demonstrates corpus statistics of annotated EmoMent corpus. According to Table 3, the majority of posts (62%) had only one label, followed by 31% of posts with two labels. Since 38% of posts had more than one label, we have modelled this problem as a multi-label classification task (see section 5). There were only 31 posts (i.e. 1% of total posts) that had four or more labels. According to the annotations, the most number of labels a post had were five and our dataset consists of eight such instances. The excerpt below demonstrates the five labels: *mental illness*, *anxiety/stress*, *sadness*, *suicidal*, and *anger*.

"[..] I'm posting this to find a solution because it's hard for me to bear. I'm in a depressed state. I took medication. I'm so nervous. Feeling sad. I feel like dying. I just want someone to talk to me in the right words with love. Then my anger is going to calm down a little. I don't get angry for nothing. but for what she does. she lies to me. I feel like her life was ruined because of me. It's too much pain to express when I feel like that. I feel like stabbing. Feeling so helpless. but it's hard for me to stay. Is there anyone who listens to me. Please help me. It's hard for me to live in this pain. Am I doing something wrong. I feel like I can't move forward. I feel like there's no life [..]"

Figure 2 shows the number of posts in each category, sorted by the frequencies of the posts. According to Figure 2, *anxiety/stress* (AS) is the most common emotion (56%) in the corpus, followed by *sadness* (SD - 36%). The majority of annotators

Number of (#) Posts	2802
# Categories	8
# labels per post	1: 62%, 2: 31% 3: 6%, 4 or more: 1%
# posts where >2 annotators agreed on at least 1 category	2106
# posts where all 3 annotators agreed on at least 1 category	1981
# posts where annotators totally disagreed on at least 1 category	9

Table 3: EmoMent Corpus statistics

(two or more) agreed that 24.5% of posts in the corpus were *irrelevant* (IV) based on our annotation guide. Figure 2 shows a large disparity between the frequencies of AS (56%) and PY (6%), SC (6%), AG (6%). For example, *anxiety/stress* was approximately nine times more frequent than *suicidal thoughts*, demonstrating that social media users may express their anxiety/stress more frequently and openly than use social media as a platform to share their suicidal thoughts, This disparity in frequencies also led to a data imbalance problem when modelling. The *other* (OT) category relates to 2% of all annotated posts. We excluded OT from modelling since the purpose of this category was to identify potential other emotions that could be useful for future expansions of the corpus. However, we did not find any such emotions.

4.2 Inter-Annotator Agreement

In order to calculate the agreement between annotators, we used Fleiss' Kappa measurement (Fleiss, 1971). Fleiss' Kappa is used to determine the agreement when two or more annotators are present.

According to Table 4, we have achieved a 'very good' inter-annotator agreement for the Sri Lankan

Dataset	MI	SD	AS	SC	AG	PY	OT	IV	Average
Sri Lankan	0.917	0.913	0.938	0.956	0.951	0.848	0.782	0.924	0.904
Indian	0.794	0.831	0.714	0.834	0.810	0.660	0.516	0.783	0.743
Average	0.856	0.872	0.826	0.895	0.880	0.754	0.649	0.853	0.823

Table 4: Inter-annotator agreement using Fleiss' Kappa

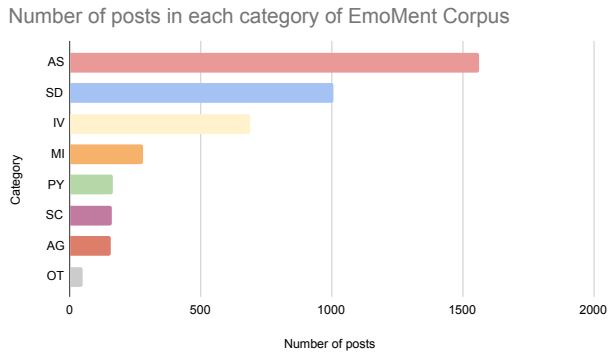

Figure 2: Number of posts in each label category, where at least two annotators agree for a particular label

dataset with Fleiss's Kappa of 0.9, enabling a promising human agreement for computational modelling. Almost all the label categories except 'other' have obtained over 0.8 of agreement. Additionally, the Indian dataset also achieved a 'good' Kappa value of 0.74. It is expected that a higher inter-annotator agreement for the Sri Lankan dataset was obtained as compared to the Indian dataset since the annotators were native Sri Lankan domain experts who have a better contextual knowledge about mental health issues among Sri Lankans, than among Indians. Table 4 shows that *anger* and *suicidal* have the highest and *other* and *psychosomatic* have the lowest agreement respectively. Interestingly, the highest annotator agreements were observed from most infrequent categories - *anger* and *suicidal* (see Figure 2).

5 Modelling

5.1 Data pre-processing

In order to prepare *EmoMent* dataset for downstream modelling tasks, we first associate each post x^i with a binary vector $y^i = [{y_1}^i, ..., {y_k}^i] \in \{0, 1\}^k$, where k represents the number of distinct labels in the taxonomy. Here ${y_j}^i$ is assigned 1 if and only if the post x^i is associated with the label j. We determine whether the post x^i is associated with the label j based on whether 2 or more annotators agree with the association. We removed posts which were not associated with any label to yield our final dataset, referred to as $\text{EmoMent}_{\text{all}}$. Additionally, we created a secondary dataset referred to as $\text{EmoMent}_{\text{relevant}}$, selecting posts which are not associated with the label '*Irrelevant* (IV)'. Hence, $\text{EmoMent}_{\text{relevant}}$ is a subset of $\text{EmoMent}_{\text{all}}$. We randomly split each dataset into training, validation and test splits in 70:15:15 ratio (see Appendix A.3 for detailed dataset split).

5.2 Emotion-annotated Mental Health Models

We propose experimental baselines for two associated tasks. The 'first task' is a binary classification task of determining whether a post is *relevant* or *irrelevant* to a mental health condition. The 'second task' is a multi-label classification task of associating correct labels (e.g., MI, SD, AS, SC) with a given post. As discussed in section 4, we chose not to consider the OT category for modelling.

We use BERT (Devlin et al., 2018) and RoBERTa (Liu et al., 2019) pre-trained language models. Our selection of BERT-based pre-trained models were motivated by previous impressive performance across different NLP tasks and related studies (Demszky et al., 2020) that used BERT-based models to propose strong baselines. RoBERTa is an optimised model based on BERT, and it has shown to outperform BERT in numerous tasks (Liu et al., 2019). Hence, we developed strong baseline models using both BERT and RoBERTa.

Figures 3(a) and 3(b) show the architecture of our RoBERTa based binary and multi-label classification models respectively. We use the Pytorch HuggingFace library[6] to implement the models. We ran all our experiments on the default GPUs provided by Google Colab[7].

5.2.1 Binary Classification Task

To address the 'first task', we fine tune BERT and RoBERTa based models on the training split (70%) of the $\text{EmoMent}_{\text{all}}$ dataset. Our hyper-parameters tuning and performance evaluation used validation (15%) and test (15%) splits of the $\text{EmoMent}_{\text{all}}$

[6]https://huggingface.co/

[7]https://colab.research.google.com/

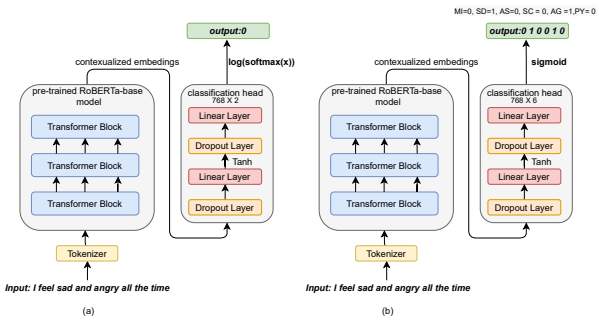

Figure 3: Architecture of the RoBERTa based models

dataset. To selected the best hyper-parameter combination, we trained three different models per each hyper-parameter combination by only modifying the random seed values, and compared the average scores obtained. We used cross-entropy loss as the loss function during training.

First we fine tuned a bert-base-cased model with a classification head on top for the binary classification task. During our experiments, we change the hyper-parameters *learning rate*, *batch size* and the *number of epochs*. We set the *warmup ratio* to 0.1 and keep the default values provided in HuggingFace implementation for the rest of the hyper-parameters. We observe the best results when we select a learning rate of 2e-05, a batch size of 8, and train the model for 5 epochs.

Next we fine tune a roberta-base model with a classification head on top for the binary classification task. To finetune the RoBERTa model, we followed a strategy similar to BERT. However, we observed best results when we set the learning rate to 2e-05, batch size to 8 and train the model for 3 epochs. We found that after 3 epochs, models tend to get overfitted to the training dataset.

5.2.2 Multi-label Classification Task

To address the 'second task', we fine tune BERT and RoBERTa based models on the training split (70%) of the $\text{EmoMent}_{\text{relevant}}$ dataset Our BERT model was a bert-base-cased pre-trained model with a classification head on top. Similarly, our RoBERTa model was a roberta-base pre-trained model with a classification head on top. The output size of the last linear layer of both of these models is set to 6 since we only considered the categories MI, SD, AS, SC, AG and PY for this task.

We used a binary cross-entropy loss function during training. To mitigate the negative impact from class imbalance, we input a vector of positive class weights to the loss function to be used when computing the loss. We computed this weight vector using the training split of the $\text{EmoMent}_{\text{relevant}}$ dataset. For each label, we divided the number of negative training data instances associated with it by the number of positive training data instances associated with it, and rounded it off to the nearest integer. If the number of negative training data instances was less than the number of positive training data instances, we assigned a default positive class weight of 1.

We used the validation split (15%) and test split (15%) of the $\text{EmoMent}_{\text{relevant}}$ dataset) to tune hyper-parameters and evaluate the models respectively. We experimented by adjusting the *learning rate*, *batch-size* and the *number of epochs*. While fine tuning BERT and RoBERTa models, we set the warmup ratio to 0.1 and kept the default values provided in the HuggingFace implementation for the rest of the hyperparameters. As similar to binary classification problem, for each hyper-parameter combination, we trained 3 separate models by updating the random seed values, and compare the average scores obtained. We find that both BERT and RoBERTa based models perform well when we use a learning rate of 2e-05, a batch-size of 8 and train the model for 5 epochs. We report the precision, recall and the F1 score of each label separately, without averaging the results across labels (Table 6).

5.3 Results

We have summarised the results in Tables 5 & 6. Since we trained 3 models for each hyper-parameter configuration by updating the random seed value, the results we have reported are the *macro-averaged* scores.

We observed that the RoBERTa model performs better than the BERT model in both tasks. In the first task, the RoBERTa model achieved an average

F1 score of 0.76 compared to the BERT model which achieved an average F1 score of 0.72. In the second task, the RoBERTa model achieved a macro-averaged F1 score of 0.77 compared to the macro-averaged F1 score of 0.71 achieved by BERT.

For the multi-label classification task, we have also reported F1 scores of the individual categories. We have observed that both BERT and RoBERTa models report the lowest F1 score for the PY category. From the Table 4, we observed that the PY category has a relatively lower inter-annotator agreement compared to MI, SD, AS, SC and AG categories. It is likely that this higher variability of data associated with the PY category could have caused both BERT and RoBERTa models to perform poorly.

We extracted misclassified posts by the best performing models for further analysis (see Table 9 of Appendix A.4 for a sample of misclassified posts). We observed that when classifying posts that seek general information or offer advice on mental health conditions, RoBERTa based binary classification model tends to get confused at times (see first 2 examples on *relevant/irrelevant* in Table 9). In the case of the multilabel-classification task, we observed that certain labels like PY gets misclassified more often. As noted in Table 4, the inter-rater agreement for the PY category is relatively low, and it is likely that the lower agreement has contributed to the misclassification of the PY category.

5.4 Limitations

As described in section 3.2, we first translated the extracted posts from Sinhala language to English prior to annotating the data. Translating the posts to English makes the dataset accessible to a much broader research community. We acknowledge that translating posts in this manner can lead to biased results. This is a limitation of the current corpus. However we argue that the benefits of translating the posts to English outweigh the disadvantages.

In this study we limited our focus to two countries in the South Asian region, Sri Lanka and India. Thus, our corpus is not representative of all the demographics in the world, and we acknowledge this as a limitation. However, we believe this does not diminish the usefulness of the corpus. The South Asian region is a populous region with more than 20% of the world's population (Véron et al., 2008). Therefore, we believe our work would be beneficial to a large audience.

Model	Precision	Recall	F1-score
BERT	0.79	0.67	0.72
RoBERTa	0.84	0.71	0.76

Table 5: Results from the binary classification task

Label	Precision		Recall		F1 Score	
	BERT	RoBERTa	BERT	RoBERTa	BERT	RoBERTa
MI	0.7	0.77	0.66	0.76	0.68	0.76
SD	0.84	0.85	0.84	0.88	0.84	0.87
AS	0.81	0.85	0.94	0.94	0.87	0.89
SC	0.66	0.76	0.78	0.77	0.72	0.76
AG	0.6	0.66	0.85	0.83	0.7	0.73
PY	0.42	0.5	0.54	0.71	0.47	0.59
macro	**0.67**	**0.73**	**0.77**	**0.82**	**0.71**	**0.77**

Table 6: Results from the multi-label classification task

6 Conclusion

This paper presented the first emotion-annotated mental health corpus - *EmoMent*, which was developed using Facebook posts from two South Asian countries - Sri Lanka and India. We have provided a comprehensive research study, demonstrating the development of an empirically-sound emotion-annotated mental health taxonomy using the grounded theory approach.

We also developed strong baselines using RoBERTa-based models and achieved an F1 score of 0.76 for the *first task* (i.e., predicting the relevance of a post to a mental health condition) and a macro-averaged F1 score of 0.77 for the *second task* (i.e., predicting the relevant labels in our taxonomy). However, our results suggest that there is ample room for future improvements in emotion-annotated mental health modelling. The models presented in the paper consider the emotion-annotated mental health modelling as two separate tasks, one binary classification to determine the relevancy, and multi-label classification to predict fine-grained labels of posts. An interesting next step would be to co-model these two tasks by leveraging multi-task learning (MTL).

7 Ethical Considerations

We curated EmoMent corpus from publicly available Facebook posts while adhering to the data policy of Meta Platforms Inc. (Meta Platform Inc., 2022), the parent organization of Facebook and

CrowdTangle.

During data collection we took steps to filter out personally identifiable information (see section 3.1). However, we acknowledge the possibility of tracing back the origins of these posts since the original posts are available in the public domain. We further acknowledge that provided annotations increase the sensitivity of the dataset. Therefore, to reduce the risk of data misuse, when releasing the dataset for academic research upon request, we plan to do so under a strict confidentiality agreement.

We also acknowledge that all mental health related diagnoses must be made only by qualified mental health practitioners, and that the computational models proposed in this study cannot be used to make such diagnostic claims about a patient.

Acknowledgements

Authors would like to acknowledge the Australian Academy of Science and the Australian Department of Industry, Science, Energy and Resources for providing financial support to conduct this research under their 'Regional Collaborations Programme COVID-19 Digital Grants'. We would also like to thank the three annotators for their support and we are grateful to the CrowdTangle research team for providing us a platform to collect Facebook data.

References

Abhinav Reddy Appidi, Vamshi Krishna Srirangam, Darsi Suhas, and Manish Shrivastava. 2020. Creation of Corpus and analysis in Code-Mixed Kannada-English Twitter data for Emotion Prediction. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 6703–6709.

Prerna G Arora, Kristina Metz, and Cindy I Carlson. 2016. Attitudes toward professional psychological help seeking in south asian students: Role of stigma and gender. *Journal of multicultural counseling and development*, 44(4):263–284.

Natalie Berry, Fiona Lobban, Maksim Belousov, Richard Emsley, Goran Nenadic, and Sandra Bucci. 2017. #WhyWeTweetMH: understanding why people use Twitter to discuss mental health problems. *Journal of medical Internet research*, 19(4):e107.

Black Dog Institute. 2020. Keeping health in mind. Technical report.

Arman Cohan, Bart Desmet, Andrew Yates, Luca Soldaini, Sean MacAvaney, and Nazli Goharian. 2018. SMHD: a large-scale resource for exploring online language usage for multiple mental health conditions. *arXiv preprint arXiv:1806.05258*.

Glen Coppersmith, Mark Dredze, and Craig Harman. 2014. Quantifying mental health signals in Twitter. In *Proceedings of the workshop on computational linguistics and clinical psychology: From linguistic signal to clinical reality*, pages 51–60.

Glen Coppersmith, Mark Dredze, Craig Harman, Kristy Hollingshead, and Margaret Mitchell. 2015. CLPsych 2015 shared task: Depression and PTSD on Twitter. In *Workshop on Computational Linguistics and Clinical Psychology: From Linguistic Signal to Clinical Reality*, pages 31–39.

Juliet M Corbin and Anselm Strauss. 1990. Grounded theory research: Procedures, canons, and evaluative criteria. *Qualitative sociology*, 13(1):3–21.

Dorottya Demszky, Dana Movshovitz-Attias, Jeongwoo Ko, Alan Cowen, Gaurav Nemade, and Sujith Ravi. 2020. GoEmotions: A dataset of fine-grained emotions. *arXiv preprint arXiv:2005.00547*.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. BERT: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*.

Paul Ekman. 1992. An argument for basic emotions. *Cognition & emotion*, 6(3-4):169–200.

Joseph L Fleiss. 1971. Measuring nominal scale agreement among many raters. *Psychological bulletin*, 76(5):378.

Soumitra Ghosh, Asif Ekbal, and Pushpak Bhattacharyya. 2020. Cease, a corpus of emotion annotated suicide notes in English. In *Proceedings of the 12th Language Resources and Evaluation Conference*, pages 1618–1626.

Spencer L James, Degu Abate, Kalkidan Hassen Abate, Solomon M Abay, Cristiana Abbafati, Nooshin Abbasi, Hedayat Abbastabar, Foad Abd-Allah, Jemal Abdela, Ahmed Abdelalim, et al. 2018. Global, regional, and national incidence, prevalence, and years lived with disability for 354 diseases and injuries for 195 countries and territories, 1990–2017: a systematic analysis for the Global Burden of Disease Study 2017. *The Lancet*, 392(10159):1789–1858.

Irene Li, Yixin Li, Tianxiao Li, Sergio Alvarez-Napagao, Dario Garcia-Gasulla, and Toyotaro Suzumura. 2020. What are we depressed about when we talk about COVID-19: Mental health analysis on tweets using natural language processing. In *International Conference on Innovative Techniques and Applications of Artificial Intelligence*, pages 358–370. Springer.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. RoBERTa: A robustly optimized BERT pretraining approach. *arXiv preprint arXiv:1907.11692*.

Meta Platform Inc. 2022. Data policy. https://www.facebook.com/policy.php.

Matthew B Miles and A Michael Huberman. 1994. *Qualitative data analysis: An expanded sourcebook*. sage.

Saif Mohammad, Felipe Bravo-Marquez, Mohammad Salameh, and Svetlana Kiritchenko. 2018. Semeval-2018 task 1: Affect in tweets. In *Proceedings of the 12th international workshop on semantic evaluation*, pages 1–17.

Laura Ana Maria Oberländer and Roman Klinger. 2018. An analysis of annotated corpora for emotion classification in text. In *Proceedings of the 27th International Conference on Computational Linguistics*, pages 2104–2119.

Inna Pirina and Çağrı Çöltekin. 2018. Identifying Depression on Reddit: The effect of training data. In *Proceedings of the 2018 EMNLP Workshop SMM4H: The 3rd Social Media Mining for Health Applications Workshop & Shared Task*, pages 9–12.

Robert Plutchik. 1980. A general psychoevolutionary theory of emotion. In *Theories of emotion*, pages 3–33. Elsevier.

Carlo Strapparava and Rada Mihalcea. 2007. Semeval-2007 task 14: Affective text. In *Proceedings of the Fourth International Workshop on Semantic Evaluations (SemEval-2007)*, pages 70–74.

Michael M Tadesse, Hongfei Lin, Bo Xu, and Liang Yang. 2019. Detection of Depression-related posts in Reddit social media forum. *IEEE Access*, 7:44883–44893.

Jacques Véron, Krystyna Horko, Rosemary Kneipp, and Godfrey Rogers. 2008. The demography of south asia from the 1950s to the 2000s. *Population*, 63(1):9–89.

WHO. 2019. The WHO special initiative for mental health (2019-2023): universal health coverage for mental health. Technical report.

Ayah Zirikly, Philip Resnik, Ozlem Uzuner, and Kristy Hollingshead. 2019. CLPsych 2019 shared task: Predicting the degree of suicide risk in Reddit posts. In *Proceedings of the sixth workshop on computational linguistics and clinical psychology*, pages 24–33.

A Appendix

A.1 Annotation Guideline: Taxonomy

Table 7 provides definitions per each category included in the annotation guideline.

A.2 Data Extraction

Figures 4 and 5 show the English and Sinhala search keywords and phrases used in the data extraction.

- Depression, Anxiety, Stress
- (I, myself, me, my, I'm) AND (unhappy, bored, sad, worrying, difficult)
- "I feel unhappy", "I feel so tired", "I'm bored", "I'm very sad", "Can't stand it", "I find it very difficult", "What is the point in life", "What is this life", "I feel like ending my life", "I'm not like before", "My mood is not right", "It is useless to go like this", "I feel lonely", "I should ending my life", "I feel like cutting myself", "I am not good enough", "I am not good", "Nobody loves me", "I just feel weird", "Not going to sleep", "Can't eat", "I can't sleep well", "Oversleeping", "I feel scared", "I feel like afraid", "chest pain", "heart pain", "I feel like I have changed", "I'm not the same person", "I'm under a lot of pressure", "I feel so stressed", "I can't handle it anymore", "Life is very tough", "I don't think I can make it", "Feeling mentally down".

Figure 4: English search keywords and phrases used in the data extraction

- විෂාදය, ආතතිය, වෙහෙස, පීඩනය, කාංසාව, බය
- "මට නොසතුටක් දැනේ", "මම අසතුටින් දැනෙනවා", "මට සතුටක් නෑ","අසතුටු", "අවාසනාවන්ත", "කනගාටුවෙන් සිටින", "කනගාටුයි", "මට ගොඩක් මහන්සියි", "මට එපාවෙලා", "මට හරි දුකයි", "හරිම දුකයි දරා ගන්න බෑ", "මේක මට දරා ගන්න බෑ", "මට හරිම අපහසුයි", "මොකද්ද මේ ජීවිතේ", "ජීවිතයේ තේරුම කුමක්ද?", "මගේ ජීවිතය අවසන් කරන්න මට දැනෙනවා", "ඉස්සර වගේ නෙමේ මම", "මගේ මුඩ් එක හරි නෑ", "මෙහෙම ගිහිල්ලා වැඩක් නෑ", "මට තනිකමක් දැනෙනවා", "මම මගේ ජීවිතය අවසන් කළ යුතුයි", "මට මාවම කපා ගන්න ඕනා", "මම හොඳ නැහැ", "කවුරුත් මට ආදරේ නෑ", "කනස්සල්ලට පත්ව සිටී", "මට නිකම් අමුතක් දැනෙනවා", "නින්ද යන්නේ නෑ", "කෑම කන්න බෑ", "මට හොඳින් නිදාගන්න බැහැ", "ගොඩක් නින්ද යනවා", "මට බයක් දැනෙනවා", "මට බයක් වගේ දැනෙනවා", "පපුව ගැහෙනවා", "මම වෙනස් වෙලා වගේ", "මම ඉස්සර කෙනා නෙමේ", "මම හරි පීඩනයෙන් ඉන්නේ", "මට ගොඩක් ආතතියක් දැනෙනවා", "මට තවදුරටත් එය හැසිරවිය නොහැක", "ජීවිතය හරිම දුෂ්කරයි", "මම නිතන්නේ නැහැ මට ඒක කරගන්න පුළුවන් කියලා", "මානසිකව වැටිලා", "මානසික වදයක්","ජීවිතය එපා වෙලා", "මානසික පීඩනය", "මානසික ගැටළු", "මානසික ගැටලු"

Figure 5: Sinhala search keywords and phrases used in the data extraction

A.3 Composition of Training, Evaluation and Test Datasets

Table 8 demonstrates the percentages of positive and negative instances associated with each label in training, validation and test splits.

A.4 A Sample of Misclassified Posts

Table 9 shows a sample of posts misclassified by the models

Category	Definition
Mental illness (MI)	Posts that explicitly mention a diagnosis of a mental illness or getting treatments for a mental illness such as depression, anxiety and seek help. Posts that expresses self-identification of mental illness may be due to history of treatments.
Sadness (SD)	Posts that express sadness, unhappy or sorrow that may lead to a maladaptive mental condition or mental illness.
Anxiety/Stress (AS)	Posts that express stress, fear or worry about something (e.g. past, future, physical appearance, religious beliefs) using the words such as anxiety, worry, fear, stress that may lead to a maladaptive mental condition or mental illness.
Suicidal (SC)	Posts that express suicidal thoughts, no interest in life (e.g. I feel like taking my own life).
Anger (AG)	Posts that express anger using words such as anger that may lead to a maladaptive mental condition or mental illness.
Psychosomatic (PY)	Posts that express psychosomatic issues (e.g. insomnia, fatigue, headaches, upset stomach) that associated with underlying mental distress or may lead to a maladaptive mental condition or mental illness.
Other (OT)	Posts that may lead to a maladaptive mental condition or mental illness but do not belong to any of the above categories (e.g., addictions, loneliness, social skill deficits such as communication issues, problem solving issues, interpersonal issues).
Irrelevant (IV)	Posts that seek information on matters related to mental conditions but do not discuss about an issue of the poster or a third party. Posts that thank others who helped. Matters related to social media group (e.g. rules of the Facebook group, objectives). Posts written using languages other than Sinhala or English.

Table 7: EmoMent Taxonomy and definitions

Label	Train		Validation		Test	
	1	0	1	0	1	0
MI	14%	86%	11%	89%	15%	85%
SD	47%	53%	51%	49%	49%	51%
AS	74%	26%	75%	25%	73%	27%
SC	8%	92%	6%	94%	8%	92%
AG	8%	92%	6%	94%	7%	93%
PY	7%	93%	9%	91%	8%	92%

Table 8: Percentages of positive and negative instances associated with each label in training, validation and test splits of $\text{EmoMent}_{\text{relevant}}$ dataset.

Post	Predicted	Actual	Misclassified
I am taking meditation classes for stress anxiety and depression... Timing is morning if interested so reply	IV=0	IV=1	IV
Can someone tell me the best meditation for anxiety relief..It will be of great help	IV=0	IV=1	IV
Anxiety is off the charts Everytime I doze off I am woken up my a feeling that I am falling and I can't breathe hate this feeling do now???	AS=1, PY=1	AS=1	PY
Just woke up with a bad dream.people are killing each other,there are hail storms, something is coming from sky destroying the earth,my family is pushing me for marriage.Since then my heart is racing	AS=1	PY=1	PY

Table 9: A sample of misclassified posts

LEGO-ABSA: A Prompt-based Task Assemblable Unified Generative Framework for Multi-task Aspect-based Sentiment Analysis

Tianhao Gao[1,*], **Jun Fang**[2,*], **HanyuLiu**[2,*], **Zhiyuan Liu**[2]
Chao Liu[2†], **Pengzhang Liu**[2], **Yongjun Bao**[2], **Weipeng Yan**[2]
[1] **School of Software and Microelectronics, Peking University**
[2] **JD, Retail, Beijing, China**
gaotianhao@pku.edu.cn
{junfang8, liuhanyu11, liuzhiyuan8,
liuchao397, liupengzhang, baoyongjun, Paul.yan}@jd.com

Abstract

Aspect-based sentiment analysis (ABSA) has received increasing attention recently. ABSA can be divided into multiple tasks according to the different extracted elements. Existing generative methods usually treat the output as a whole string rather than the combination of different elements and only focus on a single task at once. This paper proposes a unified generative multi-task framework that can solve multiple ABSA tasks by controlling the type of task prompts consisting of multiple element prompts. Further, the proposed approach can train on simple tasks and transfer to difficult tasks by assembling task prompts, like assembling Lego bricks. We conduct experiments on six ABSA tasks across multiple benchmarks. Our proposed multi-task approach achieves new state-of-the-art results in almost all tasks and competitive results in task transfer scenarios.

1 Introduction

ABSA is a fine-grained sentiment analysis task that has attracted increasing attention in recent years(Schouten and Frasincar, 2016; Nazir et al., 2020). ABSA aims to extract different elements including: 1) the aspect term(a); 2) opinion term(o); 3) the aspect category(c) corresponding to the aspect term; 4) the sentiment polarity(s) for a specific aspect term. For example, in the sentence "Pizza is delicious", "Pizza" is an aspect term belonging to the food category, and the corresponding opinion term is "delicious", which expresses positive sentiment. As shown in Table 1, based on the combination of different elements to be extracted, ABSA can be divided into multiple tasks.

This paper explores tasks containing two or more elements. In general, most ABSA tasks are transferred to classification tasks. Previous works often designed a new architecture carefully and trained with the corresponding dataset for a specific subtask. We review them as follows:

Pair Extraction The pair extraction task includes AOPE(Aspect-Opinion Pair Extraction), ACSA(Aspect-Category Sentiment Analysis) and E2E-ABSA(End-to-End Aspect-based Sentiment Analysis) in our method. ACSA is usually treated as a multi-task classification task (Hu et al., 2018; Dai et al., 2020; Ma et al., 2018) Some works convert the AOPE and E2E-ABSA tasks into sequence tagging problems(Wu et al., 2020b; Gao et al., 2021; Chen et al., 2020; He et al., 2019), specifically using the BIO tagging strategies (Wang and Pan, 2018; Wu et al., 2020a; Li et al., 2019a,b) The pair extraction is also named the basic task in this paper.

Triplets Extraction The triples extraction tasks contain ASTE(Aspect Sentiment Triplet Extraction) and TASD(Target Aspect Sentiment Detection) in our paper. Most previous works still treat them as a sequence tagging task (Xu et al., 2020, 2021; Zhang et al., 2020; Wu et al., 2021) Some works transfer them to Machine Reading Comprehension tasks(Mao et al., 2021; Chen et al., 2021).

Quadruple Extraction (Cai et al., 2021) firstly introduced the quadruple extraction task, i.e., ASQP(Aspect Sentiment Quad Prediction), and provide a multi-stage classification structure adopting from an aspect-opinion co-extraction system(Wang et al., 2017).

Recently, large-scale generative language models have become increasingly powerful(Raffel et al., 2020; Lewis et al., 2019; Radford et al., 2019), and any ABSA task can be converted to a generative problem. Some generative frameworks(Zhang et al., 2021b,a; Yan et al., 2021; Hosseini-Asl et al., 2022) have been proposed and achieved state-of-the-art results in the field of ABSA. The genera-

[*]These authors have equal contribution. Work done during gaotianhao's internship at JD, Retail, Beijing, China
[†]Corresponding authors.

Proceedings of the 29th International Conference on Computational Linguistics, pages 7002–7012
October 12–17, 2022.

Task Name	Input	Output
Aspect-Opinion Pair Extraction(AOPE)	Pizza is delicious	Pizza, delicious (a,o)
Aspect-Category Sentiment Analysis(ACSA)		food, delicious (c,s)
End-to-End Aspect-based Sentiment Analysis (E2E-ABSA)		Pizza, Positive (a,s)
Aspect Sentiment Triplet Extraction(ASTE)		Pizza, delicious, positive (a,o,s)
Target Aspect Sentiment Detection(TASD)		Pizza, food, positive(a,c,s)
Aspect Sentiment Quad Prediction(ASQP)		Pizza, delicious, food, positive (a,o,c,s)

Table 1: target of different tasks

tive format includes but is not limited to Generating Structure-Linearized Texts, Labelaugmented Text(Zhang et al., 2021b) Generating Word Indices (Yan et al., 2021) Filling Templates (Zhang et al., 2021a), as summarized by (Min et al., 2021).

However, all generative approaches mentioned above suffer from 1) training and predicting a single specific task at once; 2) treating output as a whole text rather than a combination of individual elements; 3) poor transferability from simple task to difficult task. Below is a detailed description of these three points.

For the first point, in mentioned generative approaches, the input and output formats do not support training on multiple ABSA tasks simultaneously, which we call **multi-task training setting**.

For the second point, in previous works, the models cannot understand the meaning of each element to be extracted because the input and output are treated as simple strings, and the model completes the task of predicting output through auto-regression.

For the third point, previous methods cannot be applied to task transfer scenarios. Compared to the triplets like ASTE, pairs like AOPE and E2E-ABSA are much easier in the annotation. However, previous works cannot complete ASTE by training only on AOPE and E2E-ABSA tasks even though the ASTE task elements are the same as the union of AOPE and E2E-ABSA task elements. We call this **task transfer scenario**, and it is a special case under a multi-task training setting. The proposed method has a competitive performance in this setting.

Inspired by above observations, we propose a unified generative framework LEGO-ABSA that can simultaneously solve multiple ABSA tasks and transfer from simple to complex tasks. Specifically, we take T5 as our backbone network and combine prompt learning with the practice of placing sentinel tokens of T5 pre-training. Unlike most previous works that use a piece of simple text as a prompt, e.g."ASQP" in (Zhang et al., 2021a), we design an element prompt and establish the correspondence between each element with the element prompt. We make the framework treat prompt and output text as a combination of independent elements by this design. We combine multiple element prompts into task prompt. The task prompt of a simple task can be regarded as basic bricks which can be assembled to transfer to a complex task, just like assembling Lego bricks. The output sequence is formed as a concatenation of the sentinel tokens and the real answer tokens, consistent with T5. To verify the effectiveness of our method, we conduct experiments on public datasets. Comparison results show that our proposed framework outperforms previous state-of-the-art (SOTA) approaches in most tasks. Moreover, in the case of missing part of the data annotation, it can also achieve competitive performance.

In summary, our main contributions are as follows:

- We propose a prompt-based unified generative framework to solve all ABSA tasks. The framework can be trained on multiple tasks simultaneously, and it also performs competitively in task transfer scenarios.
- To the best of our knowledge, we are the first to explore solutions for task transfer scenarios.
- The experimental results show that Our method significantly outperforms the SOTA methods on E2E-ABSA, AOPE, ASTE, and ACSA tasks.

2 Methodology

2.1 Task formulation

The proposed method will formulate any ABSA task as a text generation task. Here we give formal definitions of generative frameworks' inputs and output text.

The input $\boldsymbol{x}$ consists of two part, the raw text $\boldsymbol{t}$ and a **task prompt** $\boldsymbol{p}$: $\boldsymbol{x} = \boldsymbol{t} + | + \boldsymbol{p_{task}}$. $\boldsymbol{t} = [\boldsymbol{t_1}, \boldsymbol{t_2}, ...\boldsymbol{t_n}]$ where $\boldsymbol{t_i}$ is the ith token of $\boldsymbol{t}$ and $\boldsymbol{n}$ is

the length of tokens. $\boldsymbol{p_{task}} = [\boldsymbol{p_1}, \boldsymbol{p_2}, ...\boldsymbol{p_{m_{task}}}]$ where $\boldsymbol{p_i}$ is the ith element prompt of $\boldsymbol{p_{task}}$ and $\boldsymbol{m_{task}}$ is the number of element prompt in $\boldsymbol{p_{task}}$, which is used as a condition to generate different output text for different task.

Output text $\boldsymbol{o_{task}} = [\boldsymbol{o_1}, \boldsymbol{o_2}, ...\boldsymbol{o_{m'}}]$, where o_i is the ith tokens pair of $\boldsymbol{o_{task}}$ and $\boldsymbol{m'}$ is the output length based on the current input $\boldsymbol{x}$. The subsequent subsection will describe construction methods in detail.

2.2 Element Prompt Definition

2.2.1 Introduction of T5

T5 is an encoder-decoder model pre-trained on a multi-task mixture of unsupervised and supervised tasks converted into a text-to-text format.

In order to minimize the gap in pre-training and fine-tune, we use the same training mode as the T5 dose in pre-training. The goal of T5 is similar to the cloze test. As shown in the Figure 1(a), the input of T5 is a sentence with randomly masked consecutive spans using sentinel tokens. During unsupervised training, T5 aims to reconstruct the continuous span masked by the sentinel token, i.e., <extra_id_i> in the Figure 1(a) incrementing one by one starting from zero. Through this training object, T5 can learn general language features.

2.2.2 Element Prompt

In order to make the framework fully understand the meaning of each element in the output text, instead of treating the output as a simple string, we design an element prompt for each extracted element.

We define the element prompt as "aspect: <extra_id_0>", which has two advantages. On the one hand, the format is consistent with the T5 unsupervised training object, which can help us make better use of the information learned from pre-training. On the other hand, by defining a prompt for a single element, the output is no longer regarded as a whole text string but as a combination of different elements that offer more convenience.

The element prompts for the four elements in the ABSA task are as follows. We use w, x, y, and z to represent the id of the sentinel token.

- p_a : "aspect : <extra_id_w>"
- p_c : "category : <extra_id_x>"
- p_o : "opinion : <extra_id_y>"
- p_s : "sentiment : <extra_id_z>"

2.3 Task Prompt of Single-task Training

From shallow to deep, we start with the single ABSA task.

The element prompt is defined for each element to be extracted, but in order to complete a specific ABSA task, we need to concatenate different element prompts to form the task prompt, i.e., p_{task}. p_{task} is used as a condition so that the backbone can distinguish between different tasks. According to the kind of elements extracted and the order of element extraction in each task, we concatenate all element prompts by commas, e.g. $\boldsymbol{p_{AOPE}}$ can be p_{ao} or p_{oa} which means $p_a+, +p_o$ and $p_o+, +p_a$. Because the training for each task is independent, it is trivial to maintain a unique mapping relationship between sentinel token id and element. Here, sentinel token id for each task increments from 0, as shown in the Figure 1(b) with the sample of AOPE. The rest of the task prompts also follow the same method to define.

The arrangement order of the element prompt matters since the generation model is generated in an auto-regressive manner, and the elements generated first can provide more prior information for the elements generated later. From our experimental observations, the elements are arranged in priority according to **aspect term > opinion term = aspect category > sentiment polarity**

2.4 Task Prompt of Multi-task Training

An improvement of our framework is the ability to organize multiple ABSA tasks into a multi-task training task through task prompts.

As shown in Figure 1(c), under the multi-task training setting, the task prompt is still constructed by concatenating element prompt like the single task. The difference is that the one-to-one correspondence between elements and sentinel tokens is shared between multiple sub-tasks, so we define a global mapping relationship between the sentinel token and the corresponding element. Following the priority of elements mentioned above we assign <extra_id_0> to aspect term , <extra_id_1> to opinion term, <extra_id_2> to aspect category and <extra_id_3> to sentiment polarity. After setting each task prompt, we concatenate task prompts to each original input of the corresponding task and then mix the data of all tasks to do multi-task training.

(a) Unsupervised object of T5

(b) Objectives of LEGO-ABSA on the AOPE task

(c) multi-task training setting example

(d) task transfer scenario example

Figure 1

2.4.1 Task Transfer Scenario

This section introduces how the proposed framework works in a task transfer scenario.As shown in Figure 1(d), we define the task that extracts two elements and the combination relationship between elements as **basic task**. As illustrated in Figure 2, AOPE, E2E-ABSA, and ACSA are **basic tasks** to accomplish more complicated tasks. Basic tasks can be regarded as the bricks in LEGO.

We call the overlapping element of any two basic tasks **connection element** which is like a connector that connects two bricks. We define ASTE, TASD, and ASQP as **advanced task** which aims to extract three or more elements and the combination relationship between elements. The advanced task is like a final product assembled from basic bricks and connectors. **The goal for task transfer scenario is to resolve advanced tasks only given the training data of basic tasks,** and the process of using the basic tasks to construct the advanced tasks is like building Lego.

To achieve this goal, we need to figure out two questions: what basic tasks are required for a given advanced task; how to assemble the basic tasks, i.e., the way to construct an advanced task prompt from basic task prompts. We will give a detailed introduction in the following section for these two questions.

Basic Task Confirmation In order to complete the advanced task, we need to confirm the corresponding basic task. Because advanced tasks consist of basic tasks connected by connection elements, we need two basic tasks for extraction of triplet, like ASTE and TASD. For ASQP extraction, we need all three basic tasks mentioned in this paper. Then according to the elements contained in the task, we can determine that the basic tasks of ASTE(element set is {o, a, s}) are AOPE(element set is {o, a}) and E2E-ABSA(element set is {a, s}). The basic tasks of TASD(element set is {a, c, s}) are E2E-ABSA(element set is {a, s}) and ACSA(element set is {c, s}). The basic tasks of ASQP(element set is {a, o, c, s}) are AOPE(element set is {o, a}), E2E-ABSA(element set is {a, s}) and ACSA(element set is {c, s}).

Task Prompt Assemble After confirming the basic task, we will illustrate the method of task prompt assemble, and the arrangement order of elements is essential here. We denote the advanced task as A and its basic tasks set as $\mathbb{B}$.

We initialize $p_A = \text{“”}$, then take any element that is not a connection element as the start element and let $p_A = p_A + \text{“,”} + p_{start}$. Next, we select a task B containing start element from $\mathbb{B}$ and use another element in B as the next element, then delete

B from $\mathbb{B}$. Afterward, we use the next element as the start element and repeat the above process until all elements of A have been traversed once.

For example, as shown in Figure 2, given ASQP as advanced task, we can get $\mathbb{B}$=$\{\{a, s\}, \{a, o\}, \{c, s\}\}$, and the connection elements are $\{a, s\}$. First, we choose element o as the beginning element and the corresponding task B is $\{a, o\}$. Then, we concatenate p_o with p_A and let another element a be the new begin element and delete $\{a, o\}$ from $\mathbb{B}$. Next, the corresponding task B is $\{a, s\}$ based on element a. Afterward, we repeat the process until all elements of ASQP have been traversed once. Through the above process, the result is shown in Figure 2, p_A = $p_o + \text{“,”} + p_a + \text{“,”} + p_s + \text{“,”} + p_c$.

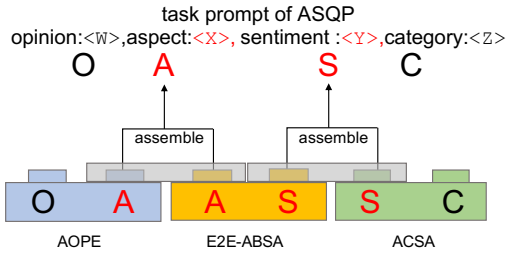

Figure 2: Illustration of task prompt assemble

We can get the task prompt after assembly and determine the order of element prompts through the above process. Besides, we also need to maintain the global mapping relationship between the sentinel token and element. As shown in Table 2, we design a set of task prompts that are unique in global mapping and conform to assemble rules. Moreover, the priority order of elements of the task prompt in Table 2 is preserved as much as possible.

2.5 Output Sequence Definition

As shown in Figure 1(b), the definition form of the o_{task} is consistent with the T5 unsupervised training output form. Whether it is a multi-task training or task transfer setting, the output sequence is formed as a concatenation of the sentinel tokens and the corresponding gold label. For sentences with multiple sets of extracted elements, we use “;” to separate sets.

3 Experiment

3.1 Datasets

We evaluate the proposed LEGO-ABSA on benchmark SemEval14-16 initially provided by the SemEval shared challenges (Pontiki et al., 2014, 2015, 2016). For each ABSA task, we use the public datasets derived from SemEval14-16 with additional sentiment annotations. Specifically, we adopt the dataset AOPE, ASTE, and E2E-ABSA provided by (Peng et al., 2020), ACSA provided by (Pontiki et al., 2015, 2016; Liu et al., 2021) TASD provided by (Wan et al., 2020) , ASQP provided by (Zhang et al., 2021a). For a fair comparison, we use the same data split as previous works.

3.2 Baselines

For E2E-ABSA, AOPE, and ASTE tasks, we adopt two types of baselines: 1) extraction based methods, including Li-unified(Li et al., 2019a), Peng-two-stage(Peng et al., 2020) and Bi-MRC(Chen et al., 2021) JET-BERT(Xu et al., 2020), Dual-MRC(Mao et al., 2021); 2) generation based methods, including GAS(Zhang et al., 2021b) and Yan-unified(Yan et al., 2021)

For the ACSA task, we adopt five baselines derived from (Cai et al., 2020). For TASD and ASQP tasks, we utilize two types of baselines 1) Extraction-based methods, including TAS-LPM-CRF and TAS-SW-TO from (Wan et al., 2020) and TASO-BERT-CRF (Zhang et al., 2021a); 2) Generation-based methods including GAS(Zhang et al., 2021b) and PARAPHRASE(Zhang et al., 2021a).

3.3 Implementation Details

Evaluation Metrics F1 score is the evaluation metric for all tasks. A prediction is correct if all its predicted sentiment elements in the pair, triplet, or quadruple are correct.

Experiment Details We adopt the pre-trained T5-base model released by *huggingface*[*]. We set the learning rate to 3e-4 as suggested by huggingface. In single task and multi-task training settings, the model is trained up to 20 epochs for the AOPE, E2E-ABSA, ACSA, and ASTE tasks and 30 epochs for the TASD and ASQP tasks. We train two multi-task models according to whether the aspect category element is included. The first is trained with AOPE, E2E-ABSA, and ASTE tasks, while the second model is trained with ACSA, TASD, and ASQP. For the in-domain task transfer setting, we train one epoch on basic tasks and 2 epochs on basic tasks with a learning rate of 3e-4. For the cross-domain setting, we train five epochs on basic tasks with a learning rate equal to 3e-4.

[*]https://huggingface.co/t5-base

Task name	Task prompt
AOPE	opinion:<extra_id_0>, aspect:<extra_id_1>
E2E-ABSA	aspect:<extra_id_1>, sentiment:<extra_id_2>
ACSA	sentiment:<extra_id_2>, category:<extra_id_3>
ASTE	opinion:<extra_id_0>, aspect:<extra_id_1>, sentiment:<extra_id_2>
TASD	aspect:<extra_id_1>, sentiment:<extra_id_2>, category:<extra_id_3>
ASQP	opinion:<extra_id_0>, aspect:<extra_id_1>, sentiment:<extra_id_2>, category:<extra_id_3>

Table 2: Task prompts in task transfer scenarios

3.4 Main Results

The main results show in Table 3 and 4. All results are the average F1 scores across 3 runs with different random seeds.

Notably, our proposed method with a single task outperforms the state-of-the-art on AOPE, E2E-ABSA, ASTE, and ACSA tasks by 1.9, 2.4, 1.7, and 3.1 average F1 scores, respectively. Besides, competitive results are also shown on TASD and ASQP.

Our method with a multi-task training setting achieved more competitive performance than separate training for each task, even though we only used one T5-base as the backbone. We get 2.5, 3.3, 2.5, and 1.6 average higher F1 scores than the state-of-the-art methods on AOPE, E2E-ABSA, ASTE, and ACSA tasks. For AOPE, E2E-ABSA, and ASTE tasks. Our model is trained on four datasets on each task and only uses one backbone, which is equivalent to reducing the backbone size to 1/12 compared with the previous method with one model per task, while the average F1 is 2.8 points higher. The result shows that multi-task training can significantly improve performance. Since the multi-task training is modeled under a unified generative framework, the construction of input and output follows the same principle so that the information between different tasks can be utilized and mutually enhanced.

Regarding why TASD and ASQP do not perform as outstanding as the rest of the tasks, we speculate that it may be because TASD and ASQP both need to extract aspect category and sentiment polarity. These two elements are generated by reasoning and have not appeared in the original text. The unsupervised pre-training object of T5 can only guarantee to generate text spans that have appeared in the original text. The working principle of sentiment and category extraction is similar to using a generative model to do classification tasks, which is different from the unsupervisied training object of T5. The gap between tasks is the leading cause of performance degradation.

3.5 Task Transfer Results

This section verifies our proposed framework's in-domain and cross-domain performance under the task transfer scenario.

3.5.1 In-domain

In the in-domain setting, we complete a advanced task by training on the necessary basic tasks of the same training corpus at a time. The result of the in-domain task transfer is shown in Table 5. We were surprised to find that the inference performance on advanced tasks is very competitive by training on basic tasks. Even the result on the ASTE task surpasses some purely supervised baselines.

3.5.2 Cross-domain

In some real situations, AOPE and E2E-ABSA annotations may not be on the same corpus, or we cannot combine them into a complete ASTE annotation. Therefore, task transfer performance of cross-domain is very important.

For TASD and ASQP tasks, since the cross-domain aspect categories are not the same, the model cannot transfer across domains on tasks that include aspect categories. Therefore, we conduct experiments on ASTE in this section under the cross-domain setting.

The cross-domain result shows in Table 6. The proposed method outperforms some purely supervised methods on average, and no noticeable performance drop compared to the in-domain setting. Compared with rule-based methods, task prompt assembly can achieve a large performance improvement. The possible reason is that, in the rule-based approach, the error of each model caused by domain transfer propagates. However, the task prompt assembly is more similar to a joint method. Therefore, the performance promotion is obvious.

4 Analysis

This section explores the principle of T5 assembly basic task corresponding to task prompt under task transfer training setting.

Model	AOPE				E2E-ABSA				ASTE			
	L14	R14	R15	R16	L14	R14	R15	R16	L14	R14	R15	R16
Li-unified(Li et al., 2019a)	52.6	55.3	56.9	53.8	63.4	73.8	65.0	70.2	42.5	51.7	46.7	44.5
Peng-two-stage(Peng et al., 2020)	53.9	56.1	56.2	60.0	62.3	74.2	65.8	71.7	43.5	51.9	46.8	53.6
JET-BERT(Xu et al., 2020)	-	-	-	-	-	-	-	-	50.0	63.9	54.7	62.9
Bi-MRC(Chen et al., 2021)	67.4	76.2	68.6	76.5	67.2	76.3	67.1	73.1	59.2	70.6	61.0	68.1
Dual-MRC(Mao et al., 2021)	63.3	74.9	64.9	75.7	64.5	76.5	65.1	70.8	55.5	70.3	57.2	67.4
GAS(Zhang et al., 2021b)	63.8	73.2	65.0	75.0	65.3	78.5	69.4	72.7	54.5	70.2	59.1	65.0
Yan-unified(Yan et al., 2021)	66.1	77.7	68.0	**77.4**	68.2	78.5	70.0	75.7	57.6	72.5	60.1	70.0
LEGO-ABSA(multi-task)	**71.3**	**78.0**	**72.9**	77.1	**72.3**	**80.6**	**74.2**	76.1	**62.2**	**73.7**	**64.4**	69.9
LEGO-ABSA(separate)	69.7	**78.1**	71.4	**77.6**	69.1	80.0	**74.3**	**78.6**	59.5	72.6	63.2	**71.5**

Table 3: Main result on AOPE, E2E-ABSA, and ASTE tasks. LEGO-ABSA(multi-task) means mixing the training dataset of three tasks and shuffling the order. LEGO-ABSA(separate) means that a task is trained with only one dataset, like other baselines. Since the original paper of GAS is not implemented on Peng's dataset, we reproduce the results ourselves using the same experiment config. We highlight the best results and results with F1 gaps within 0.2

Model	ACSA				Model	TASD		ASQP	
	L15	L16	R15	R16		R15	R16	R15	R16
Cartesian-BERT	32.8	39.5	58.4	68.9	TAS-SW-TO(Wan et al., 2020)	58.1	65.4	-	-
AddOneDim-BERT	48.9	47.2	61.7	69.8	TAS-LPM-CRF(Wan et al., 2020)	54.7	64.6	-	-
Hier-BERT	50.6	49.2	62.4	70.3	TASO-BERT-CRF(Zhang et al., 2021a)	-	-	34.8	43.7
Hier-Transformer-BERT	57.8	52.7	64.7	73.5	GAS(Zhang et al., 2021b)	60.6	68.3	46.0	56.0
Hier-GCN-BERT	62.13	54.2	64.2	74.6	PARAPHRASE(Zhang et al., 2021a)	**63.1**	**72.0**	**46.9**	**57.9**
LEGO-ABSA(multi-task)	**65.0**	53.6	67.3	75.6	LEGO-ABSA(multi-task)	62.3	**71.8**	46.1	57.6
LEGO-ABSA(separate)	64.2	**55.9**	**71.0**	**76.2**	LEGO-ABSA(separate)	61.7	68.8	45.8	**57.7**

Table 4: Main results on ACSA, TASD, and ASQP tasks. LEGO-ABSA(multi-task) means mixing individual training dataset and shuffling the order. We highlight the best results and results with F1 gaps within 0.2 in bold.

Task	L14	R14	R15	R16
ASTE	49.2	60.9	51.4	50.0
TASD	-	-	30.9	30.6
ASQP	-	-	25.8	24.5

Table 5: In-domain task transfer performance. In this situation, basic tasks and advanced task are on the same domain and corpus.

method	Lap → Rest	Rest → Lap
GAS-rule	32.4	33.7
LEGO-ABSA	**53.9**	**44.7**

Table 6: Cross-domain task transfer performance on ASTE task. We use dataset from Peng(Peng et al., 2020). Where rule method means that we get results by combining (a, s) and (a, o) with same a.

GM	TTO	TST	task transfer ability
×	×	✓	×
✓	×	✓	×
✓	✓	×	×
✓	✓	✓	✓

Table 7: Factor analysis for task transferability, where GM is global mapping between sentinel token and element, TTO is task transfer order that follows rule of Task Prompt Assemble 2.4.1, and TST is use original T5 sentinel token instead of custom token.

4.1 Factor Analysis of Transferability

We try to 1) increase the sentinel token id from 0 in each basic task, which means no global mapping between sentinel token and element. 2) give a global sentinel token id for each element but randomly arrange elements' order in the basic task. 3) employ the global mapping and right order of element prompts, but replace <extra_id_x> (the sentinel token used in T5 pre-training) with a custom new token. As shown in Table 7, only when all three conditions are met can the backbone obtain the task transferability.

Using the T5 Sentinel Token shows that downstream tasks can indeed reuse the unsupervised output of T5 pre-training. The custom token cannot have the function of masking a consecutive span because it has not been pre-trained. The global mapping between sentinel token and element shows that each sentinel token has a specific meaning after downstream task training. More importantly, the experiment result shows that a specific element

task prompt	Prediction
aspect: <extra_id_0>	tech support
opinion: <extra_id_1>	not fix
sentiment: <extra_id_2>	negative
aspect: <extra_id_0>, opinion: <extra_id_1>	tech support, not fix
aspect: <extra_id_0>, sentiment: <extra_id_2>	tech support, negative
opinion: <extra_id_1>, sentiment: <extra_id_2>	not fix, negative

Table 8: Lego split case for text "tech support would not fix the problem unless I bought your plan for $ 150 plus ."

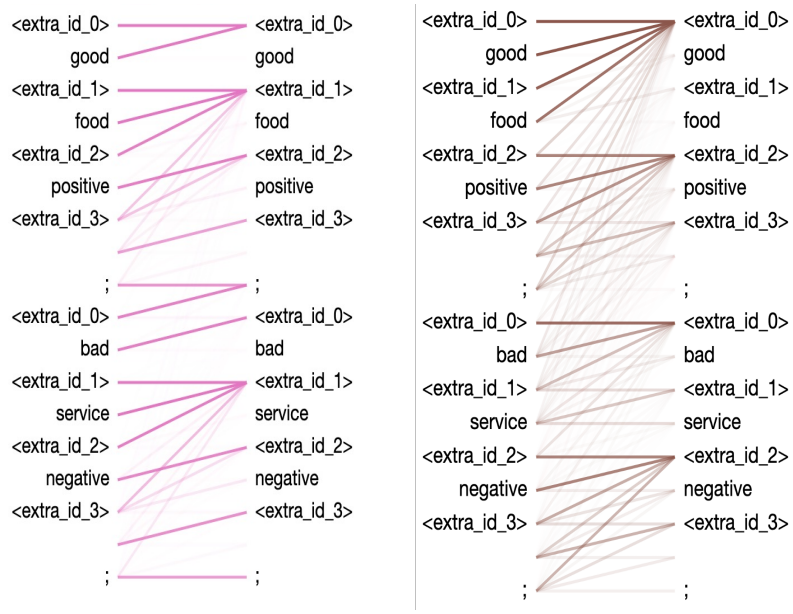

(a) Decoder attention with AS type attention head

(b) Decoder attention with OA type attention head

Figure 3: attention visualization

prompt arrangement must be used to achieve task transfer and indirectly show that what the backbone learns is how to mix two or more task prompts.

Decoder Attention Visualization

We conjecture that LEGo-ABSA uses the ending element prompt of the previous task as the beginning element prompt of the next task. To verify this, we visualized two attention heads from the T5's multiple attention heads in Figure 3. In this example, AOPE and E2E-ABSA are used as basic tasks, and ASTE is used as advanced task. Through the analysis of decoder-attention visualization, we have following findings.

Some attention heads learn associations between a and s. As shown in Figure 3(a), <extra_id_1> nearly never attend to opinion term(good) and <extra_id_0>, and <extra_id_2> attend to <extra_id_1> heavily where the association of aspect and sentiment is established. Such attention head models the relation between a and s.

Some other attention heads learn associations between element o and a. As shown in Figure 3(b), the attention weight between <extra_id_1> and <extra_id_0> is high, which means that the information of the opinion is used when the aspect is generated via the <extra_id_1>. Such attention head models the attention relationship between o and a.

In a word, combining information from multiple attention heads with different functions, our framework can model advanced tasks through basic tasks.

LEGO split

This section introduces how to make the framework trained on advanced tasks capable of extracting any custom elements by changing the task prompt like an assembled Lego can be divided into parts of different sizes.

We explored the ASTE task as the target advance task and traverse the full permutation of the three

element prompts of a, o, and s. For each permutation of element prompts, we generate a dataset with specific task prompt that assembled by element prompts. Finally we mix and shuffle all the datasets and train the framework with the setting of multitask training.

As shown in Table 8, we can arbitrarily extract any single element and any combination of elements by changing the task prompt. The framework can perfectly control the output content through the task prompt. This result shows that the approach proposed in this paper can make T5 regard the task prompt as a combination of multiple element prompts, rather than a simple string.

5 Conclusion

In this paper, we propose a prompt-based generative framework LEGO-ABSA for ABSA tasks that use T5 as the backbone, which can make full use of the information learned from the T5 unsupervised training object through the formulation of task prompts we proposed.

LEGO-ABSA does not regard the prompt and the output text as a simple string but a combination of multiple elements to be extracted. It is mainly used in multi-task training and task transfer scenarios. Extensive experiments on six ABSA tasks verify the effectiveness of our framework and its excellent transferability in task transfer scenarios. There is still space for improvement in our framework, such as completing the combination extraction of multiple elements task through the learning of single element tasks.

6 Acknowledge

Supported by the National Key Research and Development Program of China (No. 2020YFC0833301)

References

Hongjie Cai, Yaofeng Tu, Xiangsheng Zhou, Jianfei Yu, and Rui Xia. 2020. Aspect-category based sentiment analysis with hierarchical graph convolutional network. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 833–843.

Hongjie Cai, Rui Xia, and Jianfei Yu. 2021. Aspect-category-opinion-sentiment quadruple extraction with implicit aspects and opinions. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 340–350, Online. Association for Computational Linguistics.

Shaowei Chen, Jie Liu, Yu Wang, Wenzheng Zhang, and Ziming Chi. 2020. Synchronous double-channel recurrent network for aspect-opinion pair extraction. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 6515–6524.

Shaowei Chen, Yu Wang, Jie Liu, and Yuelin Wang. 2021. Bidirectional machine reading comprehension for aspect sentiment triplet extraction. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 35, pages 12666–12674.

Zehui Dai, Cheng Peng, Huajie Chen, and Yadong Ding. 2020. A multi-task incremental learning framework with category name embedding for aspect-category sentiment analysis.

Lei Gao, Yulong Wang, Tongcun Liu, Jingyu Wang, Lei Zhang, and Jianxin Liao. 2021. Question-driven span labeling model for aspect–opinion pair extraction. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 35, pages 12875–12883.

Ruidan He, Wee Sun Lee, Hwee Tou Ng, and Daniel Dahlmeier. 2019. An interactive multi-task learning network for end-to-end aspect-based sentiment analysis. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 504–515, Florence, Italy. Association for Computational Linguistics.

Ehsan Hosseini-Asl, Wenhao Liu, and Caiming Xiong. 2022. A generative language model for few-shot aspect-based sentiment analysis. *arXiv preprint arXiv:2204.05356*.

Mengting Hu, Shiwan Zhao, Li Zhang, Keke Cai, Zhong Su, Renhong Cheng, and Xiaowei Shen. 2018. Can: Constrained attention networks for multi-aspect sentiment analysis.

Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Ves Stoyanov, and Luke Zettlemoyer. 2019. Bart: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension.

Xin Li, Lidong Bing, Piji Li, and Wai Lam. 2019a. A unified model for opinion target extraction and target sentiment prediction. In *Proceedings of the AAAI conference on artificial intelligence*, volume 33, pages 6714–6721.

Xin Li, Lidong Bing, Wenxuan Zhang, and Wai Lam. 2019b. Exploiting BERT for end-to-end aspect-based sentiment analysis. In *Proceedings of the 5th Workshop on Noisy User-generated Text (W-NUT 2019)*, pages 34–41.

Jian Liu, Zhiyang Teng, Leyang Cui, Hanmeng Liu, and Yue Zhang. 2021. Solving aspect category sentiment analysis as a text generation task. *arXiv preprint arXiv:2110.07310*.

Yukun Ma, Haiyun Peng, and Erik Cambria. 2018. Targeted aspect-based sentiment analysis via embedding commonsense knowledge into an attentive lstm. *Proceedings of the AAAI Conference on Artificial Intelligence*, 32(1).

Yue Mao, Yi Shen, Chao Yu, and Longjun Cai. 2021. A joint training dual-mrc framework for aspect based sentiment analysis.

Bonan Min, Hayley Ross, Elior Sulem, Amir Pouran Ben Veyseh, Thien Huu Nguyen, Oscar Sainz, Eneko Agirre, Ilana Heinz, and Dan Roth. 2021. Recent advances in natural language processing via large pre-trained language models: A survey. *arXiv preprint arXiv:2111.01243*.

Ambreen Nazir, Yuan Rao, Lianwei Wu, and Ling Sun. 2020. Issues and challenges of aspect-based sentiment analysis: A comprehensive survey. *IEEE Transactions on Affective Computing*, pages 1–1.

Haiyun Peng, Lu Xu, Lidong Bing, Fei Huang, Wei Lu, and Luo Si. 2020. Knowing what, how and why: A near complete solution for aspect-based sentiment analysis. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pages 8600–8607.

Maria Pontiki, Dimitris Galanis, Haris Papageorgiou, Ion Androutsopoulos, Suresh Manandhar, Mohammad AL-Smadi, Mahmoud Al-Ayyoub, Yanyan Zhao, Bing Qin, Orphée De Clercq, Véronique Hoste, Marianna Apidianaki, Xavier Tannier, Natalia Loukachevitch, Evgeniy Kotelnikov, Nuria Bel, Salud María Jiménez-Zafra, and Gülşen Eryiğit. 2016. SemEval-2016 task 5: Aspect based sentiment analysis. In *Proceedings of the 10th International Workshop on Semantic Evaluation (SemEval-2016)*, pages 19–30, San Diego, California. Association for Computational Linguistics.

Maria Pontiki, Dimitris Galanis, Haris Papageorgiou, Suresh Manandhar, and Ion Androutsopoulos. 2015. SemEval-2015 task 12: Aspect based sentiment analysis. In *Proceedings of the 9th International Workshop on Semantic Evaluation (SemEval 2015)*, pages 486–495, Denver, Colorado. Association for Computational Linguistics.

Maria Pontiki, Dimitris Galanis, John Pavlopoulos, Harris Papageorgiou, Ion Androutsopoulos, and Suresh Manandhar. 2014. SemEval-2014 task 4: Aspect based sentiment analysis. In *Proceedings of the 8th International Workshop on Semantic Evaluation (SemEval 2014)*, pages 27–35, Dublin, Ireland. Association for Computational Linguistics.

Alec Radford, Jeff Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. 2019. Language models are unsupervised multitask learners.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J. Liu. 2020. Exploring the limits of transfer learning with a unified text-to-text transformer. *Journal of Machine Learning Research*, 21(140):1–67.

Kim Schouten and Flavius Frasincar. 2016. Survey on aspect-level sentiment analysis. *IEEE Transactions on Knowledge and Data Engineering*, 28(3):813–830.

Hai Wan, Yufei Yang, Jianfeng Du, Yanan Liu, Kunxun Qi, and Jeff Z Pan. 2020. Target-aspect-sentiment joint detection for aspect-based sentiment analysis. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pages 9122–9129.

Wenya Wang and Sinno Jialin Pan. 2018. Recursive neural structural correspondence network for cross-domain aspect and opinion co-extraction. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 2171–2181, Melbourne, Australia. Association for Computational Linguistics.

Wenya Wang, Sinno Jialin Pan, Daniel Dahlmeier, and Xiaokui Xiao. 2017. Coupled multi-layer attentions for co-extraction of aspect and opinion terms. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 31.

Chao Wu, Qingyu Xiong, Hualing Yi, Yang Yu, Qiwu Zhu, Min Gao, and Jie Chen. 2021. Multiple-element joint detection for aspect-based sentiment analysis. *Knowledge-Based Systems*, 223:107073.

Meixi Wu, Wenya Wang, and Sinno Jialin Pan. 2020a. Deep Weighted MaxSAT for Aspect-based Opinion Extraction. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 5618–5628, Online. Association for Computational Linguistics.

Zhen Wu, Chengcan Ying, Fei Zhao, Zhifang Fan, Xinyu Dai, and Rui Xia. 2020b. Grid tagging scheme for aspect-oriented fine-grained opinion extraction. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 2576–2585.

Lu Xu, Yew Ken Chia, and Lidong Bing. 2021. Learning span-level interactions for aspect sentiment triplet extraction.

Lu Xu, Hao Li, Wei Lu, and Lidong Bing. 2020. Position-aware tagging for aspect sentiment triplet extraction. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 2339–2349, Online. Association for Computational Linguistics.

Hang Yan, Junqi Dai, Tuo Ji, Xipeng Qiu, and Zheng Zhang. 2021. A unified generative framework for aspect-based sentiment analysis. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 2416–2429, Online. Association for Computational Linguistics.

Chen Zhang, Qiuchi Li, Dawei Song, and Benyou Wang. 2020. A multi-task learning framework for opinion triplet extraction. *arXiv preprint arXiv:2010.01512*.

Wenxuan Zhang, Yang Deng, Xin Li, Yifei Yuan, Lidong Bing, and Wai Lam. 2021a. Aspect sentiment quad prediction as paraphrase generation. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 9209–9219, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Wenxuan Zhang, Xin Li, Yang Deng, Lidong Bing, and Wai Lam. 2021b. Towards generative aspect-based sentiment analysis. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 2: Short Papers)*, pages 504–510.

A Hierarchical Interactive Network for Joint Span-based Aspect-Sentiment Analysis

Wei Chen[1,2], **Jinglong Du**[3], **Zhao Zhang**[4], **Fuzhen Zhuang**[2,5*], **Zhongshi He**[1*]
[1]College of Computer Science, Chongqing University, Chongqing, China
[2]Institute of Artificial Intelligence, Beihang University, Beijing 100191, China
[3]College of Medical Informatics, Chongqing Medical University, Chongqing, China
[4]Institute of Computing Technology, Chinese Academy of Sciences, Beijing, China
[5]SKLSDE, School of Computer Science, Beihang University, Beijing 100191, China
{mlg_cwei,zshe}@cqu.edu.cn, zhuangfuzhen@buaa.edu.cn

Abstract

Recently, some span-based methods have achieved encouraging performances for joint aspect-sentiment analysis, which first extract aspects (aspect extraction) by detecting aspect boundaries and then classify the span-level sentiments (sentiment classification). However, most existing approaches either sequentially extract task-specific features, leading to insufficient feature interactions, or they encode aspect features and sentiment features in a parallel manner, implying that feature representation in each task is largely independent of each other except for input sharing. Both of them ignore the internal correlations between the aspect extraction and sentiment classification. To solve this problem, we novelly propose a hierarchical interactive network (HI-ASA) to model two-way interactions between two tasks appropriately, where the hierarchical interactions involve two steps: shallow-level interaction and deep-level interaction. First, we utilize cross-stitch mechanism to combine the different task-specific features selectively as the input to ensure proper two-way interactions. Second, the mutual information technique is applied to mutually constrain learning between two tasks in the output layer, thus the aspect input and the sentiment input are capable of encoding features of the other task via back-propagation. Extensive experiments on three real-world datasets demonstrate HI-ASA's superiority over baselines[1].

1 Introduction

Aspect-sentiment analysis (ASA) (Yan et al., 2021; Birjali et al., 2021) aims at extracting all the aspects and their corresponding sentiments within the text simultaneously. And it can be divided into two tasks, i.e., aspect extraction (AE) and sentiment classification (SC). AE is to extract the aspects (Jakob and Gurevych, 2010; Poria et al., 2016; Karimi et al., 2021) in the sentence, and SC aims to predict the sentiments (Jiang et al., 2011; Lin et al., 2019; Karimi et al., 2021) for the extracted aspects.

In recent years, the span-based models (Hu et al., 2019; Zhou et al., 2019; Lin and Yang, 2020; Lv et al., 2021) are increasingly becoming an alternative for ASA because of their inherent advantages (e.g., they can avoid sentiment inconsistency and huge search space problems in tagging-based models (Luo et al., 2019; Wang et al., 2021)), where the aspects are extracted by directly predicting the boundary distributions, and the sentiment polarities are classified based on the aspect-level words. For example, in the sentence "The screen size is satisfactory but the phone battery capacity is limited.", the aspect spans are "screen size" and "the phone battery", and span-level sentiments are positive and negative, respectively. Formally, most of the existing methods can be divided into two types: sequential encoding (Jebbara and Cimiano, 2016; Zhou et al., 2019) and parallel encoding (Hu et al., 2019; Lv et al., 2021) due to different ways in encoding task-specific features. In sequential encoding, the task-specific features are extracted sequentially, i.e., features extracted later have no direct associations with previous ones, which is a unidirectional interaction strategy. In parallel encoding, task-specific features are extracted independently except for using shared input, i.e., the interaction is only present in input sharing. Hence, both encoding methods above fail to model two-way interactions between AE and SC appropriately.

In practice, the learning of the AE and SC may mutually influence each other. On the one hand, sentiment words can be understood better if given the desired aspects. For example, in the two sentences: "The battery has a **large** capacity." and "The computer case is too **large** to carry.", we can find that the sentiment word "large" expresses positive sentiment when describing the aspect "hard

*Corresponding authors.
[1]The source codes are released in https://github.com/cwei01/HI-ASA.

Proceedings of the 29th International Conference on Computational Linguistics, pages 7013–7019
October 12–17, 2022.

drive" while negative sentiment towards "computer". This actually shows that incorporating the aspect features is conducive to SC. On the other hand, since sentimental expressions are usually close to aspects, the potential sentiment features may provide useful signals for AE. For example, if we observe the word "**spicy**" appears in a restaurant review, it is likely to exist an aspect related to food. Thus, incorporating the sentiment feature also facilitates aspect detection. These two examples illustrate that it is essential to reasonably establish the two-way associations between two tasks.

Inspired by the above analysis, in this work, we propose a novel Hierarchical Interactive model for joint span-based Aspect-Sentiment Analysis (HIASA). Specifically, the hierarchical interactions are achieved in two steps: shallow-level interaction and deep-level interaction. The former learns semantic-level interactive features to facilitate information sharing between two tasks in encoding layer. And the latter takes advantage of the technique of mutual information maximization to ensure two-way interaction between the two tasks. Extensive experimental comparisons against state-of-the-art solutions demonstrate the effectiveness of our proposed methods.

2 Model

In this section, we formally define the problem of joint span-based aspect-sentiment analysis. Giving a training set $\{(\boldsymbol{x}_i, \{(s_i^j, e_i^j, p_i^j)\}_{j=1}^{l_i})\}_{i=1}^{N}$, in each sample $(\boldsymbol{x}_i, \{(s_i^j, e_i^j, p_i^j)\}_{j=1}^{l_i})$, $\boldsymbol{x}_i = \{x_{i,1}, x_{i,2}, \ldots, x_{i,n_i}\}$ is a n_i-length sentence, and (s_i^j, e_i^j, p_i^j) identifies an aspect with s_i^j and e_i^j as the start and end boundaries, and p_i^j as sentiment polarity. l_i is the number of aspects in the sentence.

The overall architecture is illustrated in Figure 1, which mainly consists of two parts: AE and SC. We model the hierarchical interactions between two tasks appropriately to enhance the correlations. For the input section, firstly, we leverage BERT (Devlin et al., 2018) to extract the semantic information for input sentence. The output through the transformer layers is $\boldsymbol{B} = \{b_1, b_2, \ldots, b_n\} \in \mathbb{R}^{n \times d}$, where d is the BERT's embedding size, n is the length of sentence. Then, we stack GRUs upon $\boldsymbol{B}$ for different tasks, here, we define the outputs of AE and SC as $\boldsymbol{H}_a^0 \in \mathbb{R}^{n \times \hat{d}}$ and $\boldsymbol{H}_s^0 \in \mathbb{R}^{n \times \hat{d}}$, respectively, where $\hat{d}$ is the dimension size of GRU.

2.1 Shallow-Level Interaction

In previous models (Hu et al., 2019; Lin and Yang, 2020; Lv et al., 2021), the feature representations are extracted independently except for using shared input. In other words, two tasks have no associations with each other, which is not in line with human ituition. Intuitively, the features of SC are not only derived from its own features of previous layers, but also come from the features of AE, and vice versa. Therefore, we design a shallow-level interaction strategy inspired by the idea of cross-stitch mechanism (Misra et al., 2016). Its core is to selectively combine the different features to reasonably model two-way interactions. Specifically, the calculation is as follows:

$$\begin{bmatrix} \boldsymbol{H}_a^1 \\ \boldsymbol{H}_s^a \end{bmatrix} = \begin{bmatrix} \boldsymbol{\gamma}_{\mathrm{aa}} & \boldsymbol{\alpha}_{\mathrm{sa}} \\ \boldsymbol{\alpha}_{as} & \boldsymbol{\gamma}_{\mathrm{ss}} \end{bmatrix} \begin{bmatrix} \boldsymbol{H}_a^0 \\ \boldsymbol{H}_s^0 \end{bmatrix} \tag{1}$$

where $\boldsymbol{\gamma}_{aa}$, $\boldsymbol{\gamma}_{ss}$ are the task-specific parameters, and $\boldsymbol{\alpha}_{sa}$, $\boldsymbol{\alpha}_{as}$ are task-sharing parameters. $\boldsymbol{H}_a^1$ and $\boldsymbol{H}_s^1$ are the output features of AE and SC in the last encoding layer, respectively. We can observe that the cross-stitch unit is beneficial for the interaction between AE and SC features. To speed up the training process, we define the following constraints, (1): $\boldsymbol{\gamma}_{aa} = \boldsymbol{\gamma}_{ss}$, $\boldsymbol{\alpha}_{sa} = \boldsymbol{\alpha}_{as}$ (2): $\boldsymbol{\gamma}_{aa} + \boldsymbol{\alpha}_{sa} = 1$. Thus, the Eq.1 is simplified to:

$$\begin{aligned} \boldsymbol{H}_a^1 &= \alpha \cdot \boldsymbol{H}_s^0 + (1-\alpha) \cdot \boldsymbol{H}_a^0 \\ \boldsymbol{H}_s^0 &= \alpha \cdot \boldsymbol{H}_a^0 + (1-\alpha) \cdot \boldsymbol{H}_s^0 \end{aligned} \tag{2}$$

where α controls information transferred from the other task.

2.2 Deep-Level Interaction

In the last section, we model the two-way interactions from the encoding layer. Although some semantic-level associations can be captured, we believe that task-level associations cannot be adequately modeled. To better understand our proposed method, we give an intuitive explanation in Figure 2. We observe the span-based aspect extracion models put higher scores on the entity words (e.g., "Windows" ,"7"). But with aspect-level sentiment learning, higher attention weights are not only put on some polarity words (e.g., "love"), but also on the entity words. Therefore, it would be interesting if we perform aspect-sentiment mutual learning on both tasks. Thus, we propose the deep-level interactive strategy based on mutual information maximization (Kong et al., 2019), where the task-level information is shared by the two tasks in the output layer. We will explain

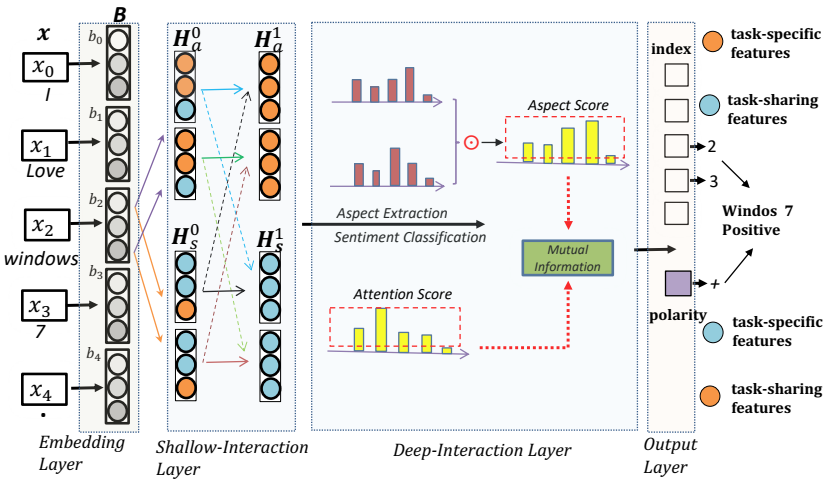

Figure 1: The architecture of our HI-ASA.

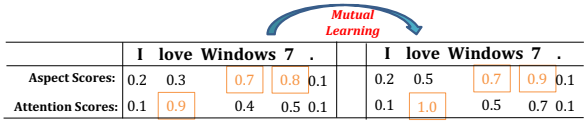

Figure 2: The example of our mutual information maximization strategy in sentence "I love Windows 7.".

in detail how to obtain the aspect scores (i.e., the more a word acts like an aspect word, the higher its score) and attention scores for these two tasks.

Aspect Extraction ($\mathcal{E}$). We extract the aspects by predicting the boundaries, i.e., start position and end position. Specifically, we employ a linear classifier to predict the start scores $g_s = \text{sigmoid}(\boldsymbol{V_s}\boldsymbol{H}_a^1)$, i.e., $\boldsymbol{V_s}$ is a trainable vector, and end scores g_e can be derived in a similar manner. During training phase, we define the learning object: $\mathcal{J}_{ae} = -\sum_{i=1}^{n} \boldsymbol{p}_s^i \log(\boldsymbol{g}_s^i) - \sum_{i=1}^{n} \boldsymbol{p}_e^i \log(\boldsymbol{g}_e^i)$, where $\boldsymbol{p}_s^i$ and $\boldsymbol{p}_e^i$ are the ground truths of the boundaries. $\boldsymbol{g}_s^i$ and $\boldsymbol{g}_e^i$ are the predicted distributions.

For the aspect score $\mathcal{E}$, we cannot use the boundary distributions of aspect (i.e., g_s and g_e) directly due to their different meanings. In fact, the span of aspect is relatively short, we can use the average boundary distributions to approximate the aspect scores, so we have: $\mathcal{E} = \frac{\text{pooling}(g_s)+\text{pooling}(g_e)}{2}$, where $\text{pooling}(\cdot)$ is the mean-pooling function, in our work, our set window size as 1×3. Therefore, suppose the start score g_s is [0.2,0.3,0.7,0.9,0.1], the first item of $\text{pooling}(g_s)$ is $\frac{0+0.2+0.3}{3} = 0.17$, and the $\text{pooling}(g_s)$ is [0.17,0.4,0.63,0.57,0.33], such strategy can guarantee that the aspect distribution approximates the boundary distributions as much as possible.

Sentiment Classification ($\mathcal{A}$). In aspect-level sentiment classification, the attention weights of words produced during the training towards the specific aspects could reveal word-polarity information. Thus, we can leverage the attention distribution as the sentiment features to help detect the boundaries of aspects. To this end, we choose the over-and-over attention model (Huang et al., 2018) to implement the sentiment classification. Here, we directly define the attention distribution as $\mathcal{A}$, (i.e., $\mathcal{A}$ is attention scores in Figure 2). During training phase, the optimization goal for SC is: $\mathcal{J}_{sc} = -\sum_{i=1}^{N} \boldsymbol{y}_i^T \log \hat{\boldsymbol{y}}_i$, where $\hat{\boldsymbol{y}}_i$ is the predicted sentiment distribution, and $\boldsymbol{y}_i^T$ is the corresponding ground truth, N is the number of samples.

Mutual Information Maximization. After getting the aspect distribution $\mathcal{E}$ and sentiment attention distribution $\mathcal{A}$. As discussed above, it should be interesting if we perform aspect-sentiment mutual learning on both tasks. In Figure 2, we can find that the aspect scores and attention weights are both improved after applying the mutual learning technique. We need to maximize the similarity between two distributions during the training. An intuitive idea is to use Kullback Leibler (KL) divergence to measure the distance between distributions, considering the correlations are bidirectional, we define the following similarity measurement metrics using Jenson Shannon (JS) divergence:

$$JS(\mathcal{E}\|\mathcal{A}) = \frac{1}{2}KL\left(\mathcal{E}\|\frac{\mathcal{E}+\mathcal{A}}{2}\right) + \frac{1}{2}KL\left(\mathcal{A}\|\frac{\mathcal{E}+\mathcal{A}}{2}\right) \tag{3}$$

where the JS divergence is symmetric, thus we can leverage it to constrain two tasks to learn from each other, resulting in balanced interaction between AE

Dataset	#Sentences	#Aspects	#+	#-	#0
Restaurant	3900	6603	4134	1538	931
Laptop	1869	2936	1326	900	620
Tweets	2350	3243	703	274	2266

Table 1: Statistics of the datasets. "+/-/0" refer to the positive, negative, and neutral sentiment classes.

Span-based Models	*Laptop*	*Res*	*Tweets*
Zhou (Zhou et al., 2019)	59.76	71.98	51.44
Hu-pipeline (Hu et al., 2019)	68.06	74.92	57.69
Hu-joint (Hu et al., 2019)	64.59	72.47	54.55
Hu-collapsed(Hu et al., 2019)	48.66	57.85	48.11
SPRM (Lin and Yang, 2020)	68.72	79.17	59.45
S-AESC (Lv et al., 2021)	65.88	74.18	54.73
HI-ASA	**70.39**	**79.90**	**60.36**
- w/o SI	68.05	79.10	60.04
- w/o DI	67.45	79.50	60.13

Table 2: The performance (F1-score) comparisons with different methods. Baseline results are retrieved from published papers.

and SC. Whereas in sequential and parallel encoding, sentiment features have no direct impact on the information of aspect features. Finally, considering the mutual information maximization between two tasks, HI-ASA is optimaled by combining three sections:

$$\arg\min_{\theta} \left(\mathcal{J}_{ae} + \mathcal{J}_{sc} + \beta \cdot JS(\mathcal{E} \| \mathcal{A})\right) \quad (4)$$

where θ is the set of model's parameters, $\mathcal{J}_{ae}$ and $\mathcal{J}_{sc}$ are the optimization goals of AE and SC respectively. β is a balanced parameter. During the test phase, HI-ASA outputs two parts, one is the scores of boundaries, based on which we leverage the heuristic algorithm (Lin and Yang, 2020) to extract all the aspects; the other one is corresponding sentiment polarities of extracted aspects.

3 Experiments

3.1 Datasets and Settings

Datasets. In our experiments, we use three public datasets, including *Laptop* (Pontiki et al., 2014), *Restaurant* (Pontiki et al., 2014, 2015, 2016) and *Tweets* (Mitchell et al., 2013), which have been widely used in previous works (Lv et al., 2021). The statistic details of experimental dataset are refer to Table 1. For each sentence in datasets, the gold span boundaries and sentiment polarity labels are available. Specifically, *Restaurant* is the union set of the restaurant domain from SemEval2014, SemEval2015 and SemEval2016. *Laptop* contains costumer reviews in the electronic product domain, which is collected from SemEval Challenge. *Tweets* is composed of twitter posts from different users.

Settings. In the experiments, the commonly used metric F1-score (F1) is selected to evaluate for aspect-sentiment analysis, accuracy is applied to sentiment classification. A correct predicted aspect only when it matches the gold aspect and the corresponding polarity. We utilize the BERT-Large model as the backbone network,where the number of transformers is 24 and the hidden size is 784. In addition, we use Adam optimizer with a learning rate of 3e-5, the batch size is 32 and the dropout probability of 0.1 is used.

3.2 Main Results

The comparisons between HI-ASA and the baselines are presented in Table 2. Specifically, "- w/o SI" means we remove the shallow-interaction layer, "- w/o DI" denotes removing the deep-interaction layer. We can observe: **(1)** Overall, our proposed HI-ASA consistently achieves the best F1 scores across all the baselines. More specifically, compared to the state-of-the-art approach SPRM, HI-ASA improves the performance by about 1.67%, 0.63%, and 0.91% on three datasets, respectively. These observations indicate the carefully designed HI-ASA is capable of achieving better performances. The reason can be concluded two folds: one is that we selectively combine the task-specific features of each task to reasonably model two-way interactions in the encoding layer; the second benefit is that we model a balanced task-level interactions. Under this framework, aspect and sentiment associations are captured appropriately in a hierarical manner. **(2)** Besides, we investigate the effectiveness of each single module, i.e., "- w/o SI" and "- w/o DI". We can see that when a certain module is removed, the performance of our model decreases, which indicates the indispensability of each module.

3.3 Ablation Study

In this section, we go deeper into HI-ASA and analyse the results on both tasks. The results are shown in Table 3. Generally, we observe that HI-ASA outperforms baseline competitors on both tasks, which indicates the effectiveness of the two-way interaction on both aspect extraction and sentiment classification. For the AE task, HI-ASA can enhance the performance of most of the baselines. On the SC task, HI-ASA outperforms SPRM by 3.98%, 2.83%, and 5.16% on three datasets, respec-

Aspect Extraction	*Laptop*	*Res*	*Tweets*
Hu (Hu et al., 2019)	83.35	82.38	75.28
SPRM (Lin and Yang, 2020)	84.72	86.71	69.85
S-AESC (Lv et al., 2021)	85.19	84.20	76.04
HI-ASA	**86.30**	86.93	**76.36**
- w/o SI	84.50	**87.33**	75.81
- w/o DI	85.10	87.09	75.99
Sentiment Classifiction	*Laptop*	*Res*	*Tweets*
Hu (Hu et al., 2019)	81.39	89.95	75.16
SPRM (Lin and Yang, 2020)	81.50	90.35	78.34
HI-ASA	**85.02**	**93.18**	**83.50**
- w/o SI	84.07	92.53	82.90
- w/o DI	83.75	92.62	83.49

Table 3: The top part is the performance (F1-score) comparisons with different methods on aspect extraction. And the bottom part is the performance (Accuracy) comparisons with different methods on sentiment classifiction.

Dataset/α	0	0.1	0.2	0.3	0.4	0.5
Laptop	68.05	**70.39**	67.62	68.63	68.58	70.36
Restaurant	79.14	78.93	79.73	79.45	**79.90**	78.48
Tweets	60.04	**60.36**	59.70	59.89	58.79	58.19

Table 4: The performance (F1-score) of different sharing ratios (α) on three datasets.

tively, which implies that it is more able to boost the performance of SC compared to AE.

3.4 Parameter Analysis α

In HI-ASA, the parameter α controls information transfer between two tasks. For a special case, if α = 0, our approach will degenerate to the parallel encoding. We tune α in the range of [0.0,0.1,0.2,0.3,0.4,0.5] and the results are presented in Table 4. It is worth noting that we set the value of α below 0.5. This setting is inspired by the ituition that the features for each task should come more from the task itself, rather than the other task. We can notice the best results are achieved when α is 0.1, 0.4, and 0.1 on three datasets, respectively, rather than 0.0. This actually shows our interaction is successful.

Furthermore, we find an interesting phenomenon is that α is small (i.e., 0.1) on *Laptop* and *Tweets*, and large on *Restaurant* when performing the best performance. We conjecture the reason lies in that *Restaurant* has more samples than the other two datasets, requiring more knowledge interactions for a better learning process.

4 Related Work

Aspect-sentiment Analysis (ASA) (Yan et al., 2021; Birjali et al., 2021) is an essential task in sentiment analysis and can be separated into two tasks, i.e., aspect extraction (AE) and sentiment classification (SC). Over the past years, some span-based methods have achieved promising results for ASA, which first extract aspects by detecting aspect boundaries (AE) and then predict the span-level sentiments (SC). AE has been widely studied by traditional machine (Jakob and Gurevych, 2010) and deep learning algorithms (Karimi et al., 2021). However, the absence of sentiment information may result in redundant and noisy detection. SC is to predict the sentiment expressed on some specific aspects in a sentence, which has been studied extensively in NLP community (Tang et al., 2015; Chen et al., 2019, 2020; Karimi et al., 2021). However, these aspects must be annotated before the AE task. (Hu et al., 2019) proposed an extract-then-classify framework, which extracted aspects with a heuristic decoding algorithm and then correspondingly classified the span-level sentiments. (Lin and Yang, 2020) designed share-private representation for each task to capture the correlations between two tasks in encoding layer. Besides, (Zhou et al., 2019) introduced a joint model based on span-aware attention mechanism to predict the sentiment polarity. Although achieved improved performances, these works fail to model two-way interactions between AE and SC appropriately.

5 Conclusion

In this paper, we proposed a novel aspect-sentiment analysis model named HI-ASA. The proposed model is equipped with a hierarchical interactive network to facilitate information sharing between the aspect extraction task and the sentiment classification task. The experimental results on three benchmark datasets demonstrated HI-ASA's effectiveness and generality.

Acknowledgments

We are grateful to all the anonymous reviewers for their helpful advice on various aspects of this work. The research work is supported by the National Key Research and Development Program of China under Grant No. 2021ZD0113604, the National Natural Science Foundation of China under Grant No. 62176014, Graduate Research and Innovation Foundation of Chongqing, China under Grant No.CYC21072.

References

Marouane Birjali, Mohammed Kasri, and Abderrahim Beni-Hssane. 2021. A comprehensive survey on sentiment analysis: Approaches, challenges and trends. *Knowledge-Based Systems*, 226:107134.

Wei Chen, Wenxin Yu, Gang He, and Ning Jiang. 2020. Coarse-to-fine attention network via opinion approximate representation for aspect-level sentiment classification. In *International Conference on Neural Information Processing*, pages 704–715. Springer.

Wei Chen, Wenxin Yu, Zhiqiang Zhang, Yunye Zhang, Kepeng Xu, Fengwei Zhang, Yibo Fan, Gang He, and Zhuo Yang. 2019. Target-based attention model for aspect-level sentiment analysis. In *International Conference on Neural Information Processing*, pages 259–269. Springer.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*.

Minghao Hu, Yuxing Peng, Zhen Huang, Dongsheng Li, and Yiwei Lv. 2019. Open-domain targeted sentiment analysis via span-based extraction and classification. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 537–546.

Binxuan Huang, Yanglan Ou, and Kathleen M Carley. 2018. Aspect level sentiment classification with attention-over-attention neural networks. In *International Conference on Social Computing, Behavioral-Cultural Modeling and Prediction and Behavior Representation in Modeling and Simulation*, pages 197–206.

Niklas Jakob and Iryna Gurevych. 2010. Extracting opinion targets in a single and cross-domain setting with conditional random fields. In *Proceedings of the 2010 conference on empirical methods in natural language processing*, pages 1035–1045.

Soufian Jebbara and Philipp Cimiano. 2016. Aspect-based sentiment analysis using a two-step neural network architecture. In *Semantic Web Evaluation Challenge*, pages 153–167. Springer.

Long Jiang, Mo Yu, Ming Zhou, Xiaohua Liu, and Tiejun Zhao. 2011. Target-dependent twitter sentiment classification. In *Proceedings of the 49th annual meeting of the association for computational linguistics: human language technologies*, pages 151–160.

Akbar Karimi, Leonardo Rossi, and Andrea Prati. 2021. Adversarial training for aspect-based sentiment analysis with bert. In *2020 25th International Conference on Pattern Recognition (ICPR)*, pages 8797–8803. IEEE.

Lingpeng Kong, Cyprien de Masson d'Autume, Lei Yu, Wang Ling, Zihang Dai, and Dani Yogatama. 2019. A mutual information maximization perspective of language representation learning. In *International Conference on Learning Representations*.

Peiqin Lin and Meng Yang. 2020. A shared-private representation model with coarse-to-fine extraction for target sentiment analysis. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: Findings*, pages 4280–4289.

Peiqin Lin, Meng Yang, and Jianhuang Lai. 2019. Deep mask memory network with semantic dependency and context moment for aspect level sentiment classification. In *Proceedings of the 28th International Joint Conference on Artificial Intelligence*, pages 5088–5094.

Huaishao Luo, Tianrui Li, Bing Liu, and Junbo Zhang. 2019. Doer: Dual cross-shared rnn for aspect term-polarity co-extraction. *arXiv preprint arXiv:1906.01794*.

Yanxia Lv, Fangna Wei, Ying Zheng, Cong Wang, Cong Wan, and Cuirong Wang. 2021. A span-based model for aspect terms extraction and aspect sentiment classification. *Neural Computing and Applications*, 33(8):3769–3779.

Ishan Misra, Abhinav Shrivastava, Abhinav Gupta, and Martial Hebert. 2016. Cross-stitch networks for multi-task learning. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 3994–4003.

Margaret Mitchell, Jacqui Aguilar, Theresa Wilson, and Benjamin Van Durme. 2013. Open domain targeted sentiment. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*, pages 1643–1654.

Maria Pontiki, Dimitrios Galanis, John Pavlopoulos, Harris Papageorgiou, Ion Androutsopoulos, and Suresh Manandhar. 2014. Semeval-2014 task 4: Aspect based sentiment analysis. In *Proceedings of the 8th International Workshop on Semantic Evaluation (SemEval 2014)*, pages 27–35.

Maria Pontiki, Dimitris Galanis, Haris Papageorgiou, Ion Androutsopoulos, Suresh Manandhar, AL-Smadi Mohammad, Mahmoud Al-Ayyoub, Yanyan Zhao, Bing Qin, Orphée De Clercq, et al. 2016. Semeval-2016 task 5: Aspect based sentiment analysis. In *Proceedings of the 10th international workshop on semantic evaluation (SemEval-2016)*, pages 19–30.

Maria Pontiki, Dimitris Galanis, Haris Papageorgiou, Suresh Manandhar, and Ion Androutsopoulos. 2015. Semeval-2015 task 12: Aspect based sentiment analysis. In *Proceedings of the 9th International Workshop on Semantic Evaluation (SemEval 2015)*, pages 486–495.

Soujanya Poria, Erik Cambria, and Alexander Gelbukh. 2016. Aspect extraction for opinion mining with a deep convolutional neural network. *Knowledge-Based Systems*, 108:42–49.

Duyu Tang, Bing Qin, Xiaocheng Feng, and Ting Liu. 2015. Effective lstms for target-dependent sentiment classification. *arXiv preprint arXiv:1512.01100*.

Xinyi Wang, Guangluan Xu, Zequn Zhang, Li Jin, and Xian Sun. 2021. End-to-end aspect-based sentiment analysis with hierarchical multi-task learning. *Neurocomputing*, 455:178–188.

Hang Yan, Junqi Dai, Tuo Ji, Xipeng Qiu, and Zheng Zhang. 2021. A unified generative framework for aspect-based sentiment analysis. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language*, pages 2416–2429.

Yan Zhou, Longtao Huang, Tao Guo, Jizhong Han, and Songlin Hu. 2019. A span-based joint model for opinion target extraction and target sentiment classification. In *Proceedings of the 28th International Joint Conference on Artificial Intelligence*, pages 5485–5491.

MuCDN: Mutual Conversational Detachment Network for Emotion Recognition in Multi-Party Conversations

Weixiang Zhao, Yanyan Zhao*, Bing Qin
Research Center for Social Computing and Information Retrieval
Harbin Institute of Technology, China
{wxzhao, yyzhao, qinb}@ir.hit.edu.cn

Abstract

As an emerging research topic in natural language processing community, emotion recognition in multi-party conversations has attained increasing interest. Previous approaches that focus either on dyadic or multi-party scenarios exert much effort to cope with the challenge of emotional dynamics and achieve appealing results. However, since emotional interactions among speakers are often more complicated within the entangled multi-party conversations, these works are limited in capturing effective emotional clues in conversational context. In this work, we propose Mutual Conversational Detachment Network (MuCDN) to clearly understand the conversational context by separating conversations into detached threads. Specifically, two detachment ways are devised to perform context and speaker-specific modeling within detached threads and they are bridged through a mutual module. Experimental results on two datasets show that our model achieves better performance over the baseline models.

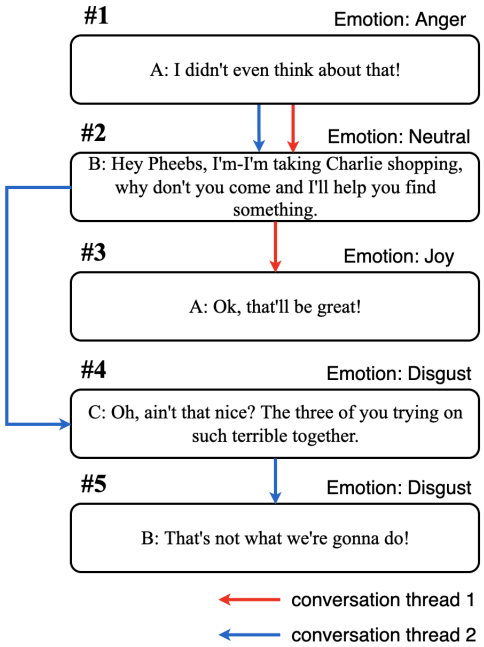

Figure 1: An example of a multi-party conversation from MELD dataset.

1 Introduction

Emotion recognition in conversations (ERC) is a task that predicts the emotion for each utterance in conversations. Since it plays a significant role in achieving empathetic systems and is of great values to be applied in the fields of opinion mining in conversations, social media analysis, mental health care and other areas, ERC has received more and more attention in the natural language processing (NLP) community. We focus on emotion recognition in multi-party conversations (ERMC) where three or more speakers are involved.

The challenge of ERC, especially for that of ERMC, lies in a complicated emotional interaction referred to as emotional dynamics (Poria et al., 2019b) that utterances from one speaker would have impact on expressions of others. To cope with the challenge, many previous approaches exploit recurrent neural network (RNN) (Majumder et al., 2019; Ghosal et al., 2020) and graph neural network (GNN) (Ghosal et al., 2019; Shen et al., 2021b; Zhang et al., 2022) to perform context and speaker-specific modeling. However, they are not effective enough to capture exact emotional clues for emotion recognition because the salient characteristic of multi-party conversations is ignored.

Multiple conversation threads are intermingled in the same dialog history simultaneously in the multi-party scenario (Ma et al., 2021; Liu et al., 2021a), which makes it hard to capture emotional clues in such a complex and confusing interaction. As shown in Figure 1, there are two conversation threads in a multi-party conversation consisting of three speakers A, B and C. Targeting at the same utterance #2 from speaker B to reply, speaker A is consoled and becomes joyful to go shopping, while

*Corresponding author

Proceedings of the 29th International Conference on Computational Linguistics, pages 7020–7030
October 12–17, 2022.

speaker C is unbearable towards the outfit of B and feels disgusted. Since speaker C and A originally focus on different aspects of the same dialog history, two individual threads are naturally developed. It is of great necessity to figure out the intra- and inter-personal dependencies of utterances within distinct conversation threads, for the convenience of understanding the exact emotional interaction exerted on different speakers.

In this paper, in order to clearly comprehend emotional clues for emotion recognition, we propose Mutual Conversational Detachment Network (MuCDN) to separate a multi-party conversation into detached threads and effectively perform context and speaker-specific modeling within them.

To be more specific, two detachment ways, named explicit detachment and implicit detachment are devised to separate a multi-party conversation into several threads and jointly carry out context and speaker-specific modeling within them. For the former one, we detach a multi-party conversation with the help of a dialog discourse parser. Along with paths of a discourse tree, detached conversation threads could be explicitly attained. Then two speaker-aware gated neural networks (GRU) are adopted for conversational information propagating. Further, implicit detachment aims at capturing the latent and global interaction in the conversation. We construct two speaker-specific implicit detachment graphs and utilize self-attention mechanism to obtain detached threads implicitly. In addition, a mutual module is designed to create the interaction between explicit detachment and implicit detachment. To evaluate the performance of MuCDN, we conduct extensive experiments on two ERMC datasets and new state-of-the-art performance is achieved on both of them.

The main contributions of this work are summarized as follows:

- In order to cope with the complicated emotional dynamics in ERMC, we propose MuCDN with the notion of conversation detachment.
- We devise two detachment ways to separate multi-party conversations into distinct threads and clearly perform context and speaker-specific modeling within them. A mutual module is also designed for interaction.
- Results of extensive experiments on two benchmark ERMC datasets demonstrate the effectiveness of the proposed model. Our source code is publicly available at https://github.com/circle-hit/MuCDN.

2 Related Work

2.1 Emotion Recognition in Conversations

Recent works that focus on dyadic conversations also perform experiments on ERMC datasets. They either plainly perform context modeling or incorporate external resources, which can be divided into two categories.

For those without incorporating external resources, two types of deep neural network are adopted for context modeling. (1) **RNN**. Majumder et al. (2019) take global state, personal state and emotion state of speakers into account and utilize three GRUs to model emotional dependency among speakers. Hu et al. (2021) devise a cognitive reasoning module to iteratively capture emotional clues and fully understand conversational context with cognitive factors. (2) **GNN**. Ghosal et al. (2019) perform context and speaker-specific modeling upon a graph and relational graph convolutional network is adopted for message passing through various types of edges. Further, Ishiwatari et al. (2020) simplify the selection of some types of edges in the graph and propose relational position encoding to enhance relation-aware graph attention network. Shen et al. (2021b) abstract a conversation into a directed acyclic graph and extend the directed acyclic graph neural network to be suitable under the conversation setting.

More recently, many works leverage external resources to enrich contextual representations. Zhong et al. (2019) utilize commonsense knowledge and emotion lexicon to guide context modeling and dynamically retrieve context-aware and emotion-related knowledge. Ghosal et al. (2020) explore several types of commonsense knowledge for a comprehensive understanding of various aspects of conversations such as personality, events, mental states and intents. Based on this, Li et al. (2021) concentrate more on psychological interactions between utterances. Besides, an auxiliary task named sentiment polarity intensity prediction is introduced to involve direct affective information from the emotion lexicon (Xie et al., 2021). And Zhao et al. (2022) leverage commonsense knowledge as causal clues to aid the emotion recognition with emotion cause detection.

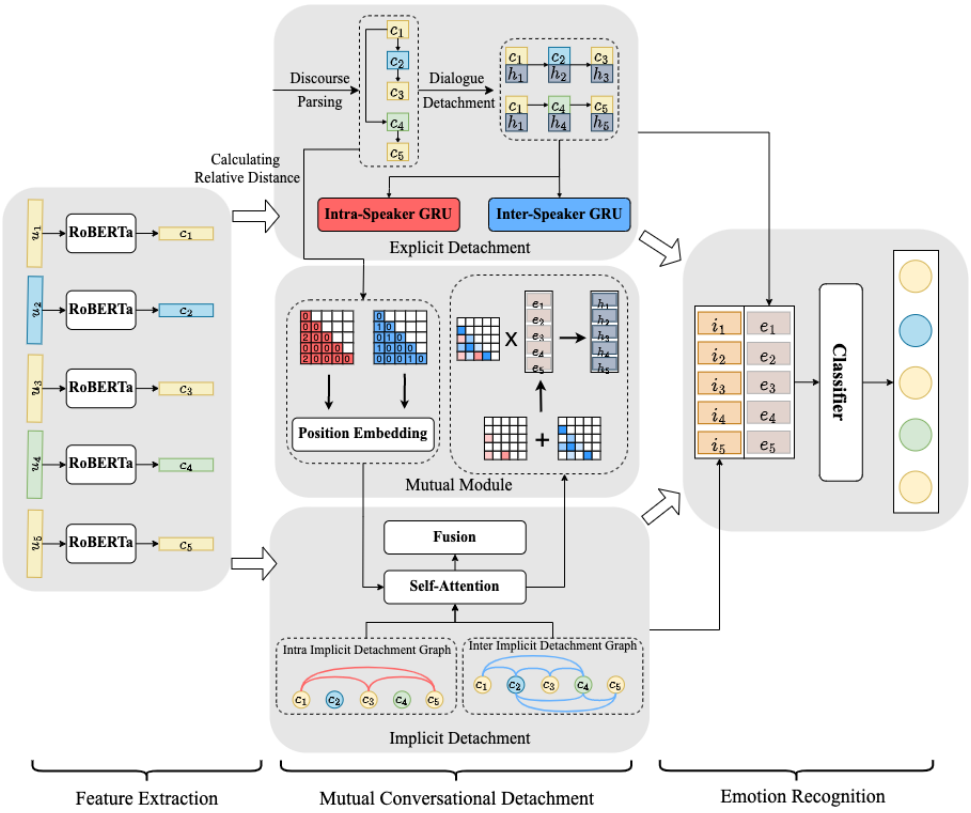

Figure 2: The overall architecture of our proposed model.

2.2 Emotion Recognition in Multi-Party Conversations

For emotion recognition in multi-party conversations, recent studies endeavor to capture contextual clues and speaker-specific information in the complex interactions. Zhang et al. (2019) construct a large graph over the entire corpus and propose a conversational graph-based convolutional neural network to model both context- and speaker-sensitive dependency. Shen et al. (2021a) make the pretrained language model XL-Net (Yang et al., 2019) adaptive to the dialog scenario and design four types of dialog-aware self-attention to model contextual information. Sun et al. (2021) explore the importance of discourse structures in handling informative contextual cues and speaker-specific features and build a graph based on the dialog discourse structure.

However, all the aforementioned methods ignore the notable characteristic of multi-party conversations that multiple conversation threads are entangled in the dialog history. And in order to clearly capture emotional clues for emotion recognition, we propose to perform context and speaker-specific modeling within individual threads detached from multi-party conversations.

3 Methodology

First, we define the problem of the ERMC task. Given a conversation with N consecutive utterances $\{u_1, u_2, \cdots, u_N\}$ and M speakers ($M > 2$) $\{s_1, s_2, \cdots, s_M\}$, ERMC aims to predict the emotion label e_i of each utterance u_i spoken by s_i.

Figure 2 gives the overall architecture of our proposed model, which consists of five parts: Utterance-Level Feature Extraction, Explicit Detachment, Implicit Detachment, Mutual Module and Emotion Recognition. After attaining context-independent vector of each utterance, we propose to clearly carry out context and speaker-specific modeling within individual threads detached from multi-party conversations. On the one hand, explicit detachment is performed on the outcome of the dialogue discourse parser and two speaker-aware GRUs are adopted for information propagating across each detached thread. On the other hand, the latent and global interaction is captured by implicit detachment through attention values calculated by self-attention mechanism. Further, mutual module exchanges the interaction information for both explicit detachment and implicit detachment. And the final representation is obtained by concate-

nating outcomes from the two detachment modules. We will elaborate each one of the five modules in the rest of this section.

3.1 Utterance-Level Feature Extraction

We employ the widely-used pretrained language model RoBERTa (Liu et al., 2019) to perform utterance-level feature extraction. Specifically, for each utterance $u_i = \{w_1, w_2, \cdots, w_L\}$, a special token $[CLS]$ is concatenated to the beginning of the utterance. Then we feed the sequence $\{[CLS], w_1, w_2, \cdots, w_L\}$ to fine-tune the pretrained RoBERTa model by an utterance-level emotion classification task and the $[CLS]$ token from the last hidden layer is fed to a pooling layer to attain the result of emotion classification.

After the process of fine-tuning, to obtain each utterance-level feature vector c_i represented by the $[CLS]$ token, we feed each utterance in the same input format as $\{[CLS], w_1, w_2, \cdots, w_L\}$:

$$c_i = \text{RoBERTa}(\, [CLS], w_1, w_2, \cdots, w_L \,) \quad (1)$$

where $c_i \in R^{d_m}$ and d_m is the dimension of hidden states in RoBERTa. Following (Ghosal et al., 2020), $[CLS]$ tokens from final four layers are averaged to obtain the utterance-level feature vector for each utterance. Then each utterance vector c_i is transformed to the dimension of d_h with a linear projection. And the vectorized representation of a conversation C is $\{c_1, c_2, \cdots, c_N\}$.

3.2 Explicit Detachment

To clearly and effectively perform context and speaker-specific modeling under the circumstance of multi-party scenario, we separate a conversation into several individual threads according to the results of its corresponding discourse structure.

Discourse Parsing and Detachment: We utilize the discourse parser proposed by Shi and Huang (2019), which is a deep sequential model and performs well on STAC (Asher et al., 2016) corpus. Given each conversation $\{u_1, u_2, \cdots, u_N\}$, the discourse structure could be obtained by:

$$\{(i, j, e_{ij}), \cdots\} = \text{Parser}(\{u_1, u_2, \cdots, u_N\}) \quad (2)$$

where e_{ij} is the directed edge with head j and tail i and i, j are indices of utterances in the conversation ($i < j$). According to their settings, the output of the parser is a discourse dependency tree where each node (utterance) is connected with a single parent node. And detached threads are naturally formed according to each definite path from the start node to the current one. We use an explicit detachment matrix D to represent the derived threads from discourse tree:

$$D_{i,j} = \begin{cases} 1, & \text{if } e_{ij} \text{ exists in discourse tree} \\ 0, & \text{otherwise} \end{cases} \quad (3)$$

Speaker-Aware Context Modeling: With individual conversation threads detached, we propose to perform clear and effective context and speaker-specific modeling upon them. Specifically, we adopt two speaker-aware GRUs, named intra-speaker GRU and inter-speaker GRU, for contextual information propagating through each detached thread along with connections guided by explicit detachment matrix D:

$$e_i = \begin{cases} \text{GRU}^{intra}(c_i,\ e_p), & \text{if } \phi(u_i) = \phi(u_p) \\ \text{GRU}^{inter}(c_i,\ e_p), & \text{otherwise} \end{cases} \quad (4)$$

where $e_i \in R^{d_h}$ and ϕ maps the utterance into that of the corresponding speaker and u_p is the single precursor of the current utterance u_i which means $D_{pi} = 1$ ($p < i$). GRU^{intra} is utilized to model the intra-speaker dependency from the same speaker, while GRU^{inter} are for interactions from other speakers.

3.3 Implicit Detachment

The reason for the design of implicit detachment is that the explicit detachment mainly focuses on the local interaction, which means only recurrently propagating information in each thread would ignore the global information contained in a multi-party conversation. When a speaker is buried in a conversation, the triggering of his/her target emotion may be more or less from multi sources, not just a single precursor analyzed by the parser. Thus, to capture the global emotional clues and dig the latent emotional dependency among utterances, we attempt to detach the multi-party conversation in an implicit way. Specifically, we construct two partially fully-connected implicit detachment graphs (IDG) with speaker-aware information injected according to whether two utterances are from the same speaker or not. IDG^{intra} and IDG^{inter} are obtained in the form of adjacent matrices by:

$$IDG^{intra}_{i,j} = \begin{cases} 0, & \text{if } j <= i \text{ and } \phi(u_i) = \phi(u_j) \\ -\infty, & \text{otherwise} \end{cases} \quad (5)$$

$$IDG_{i,j}^{inter} = \begin{cases} 0, \text{ if } j < i \text{ and } \phi(u_i) \neq \phi(u_j) \\ -\infty, \text{ otherwise} \end{cases} \tag{6}$$

Then we utilize multi-head self-attention mechanism (MHSA) and add IDG to the result of dot product between query and key to achieve speaker-aware information propagating. We omit the formula for multi-head computation due to the limited space and more details about MHSA could be found in Vaswani et al. (2017):

$$\begin{aligned} G &= \text{MHSA}(C,\ IDG^t), \\ \text{Att}(Q, K, V, IDG^t) &= \text{Softmax}(\frac{QK^T}{\sqrt{d_k}} + IDG^t)V \end{aligned} \tag{7}$$

where $G \in R^{N \times d_h}$ and $t \in \{intra, inter\}$ is the type of IDG. Viewing self-attention weights as edges connecting representations, to what extent we obtain the detached threads is determined by the attention values calculated by the self-attention mechanism. we then fuse the two speaker-aware implicit detachment representations with a gated manner inspired by Liu et al. (2021b):

$$\begin{aligned} F^t &= \text{ReLU}(\text{FC}([C, G^t, C - G^t, C \odot G^t])), \\ g &= \text{Sigmoid}(\text{FC}[F^{intra}, F^{inter}]), \\ I &= g \odot F^{intra} + (1 - g) \odot F^{inter} \end{aligned} \tag{8}$$

where $I \in R^{N \times d_h}$ and FC is the fully-connected layer.

3.4 Mutual Module

This module is designed for mutual interaction between the process of explicit detachment and implicit detachment.

For the interaction from implicit detachment to explicit detachment, it servers as the complement of the global information and latent interactions in the conversations. First, we obtain two attention score matrices from the self-attention layer with the average of each attention head. And the joint score A^{joint} is measured in the same scale by softmax function. Then the i-th global representation complemented for its corresponding utterance in the explicit detached threads is obtained by:

$$h_i = A_{i,<i}^{joint} \times E_{<i} \tag{9}$$

where $h_i \in R^{d_h}$, $\times$ represents the operation of matrix multiply, $A_{i,<i}^{joint}$ is the prior $i-1$ elements in the i-th row of A^{joint} and $E_{<i} = \{e_1, e_2, \cdots, e_{i-1}\}$ stands for the prior $i-1$ vectors calculated by speaker-aware GRU. We concatenate each h_i with its corresponding c_i before thread information propagating and the original computation in Equation (4) is updated:

$$e_i = \begin{cases} \text{GRU}^{intra}([c_i, h_i],\ e_p), & \text{if } \phi(u_i) = \phi(u_p) \\ \text{GRU}^{inter}([c_i, h_i],\ e_p), & \text{otherwise} \end{cases} \tag{10}$$

In addition, explicit detachment would provide exact relative position information based on the discourse tree for the process of implicit detachment. Ishiwatari et al. (2020) argue that human emotions may depend on more immediate utterances in the temporal order. We extend such a temporal distance to the structural form under the circumstance of multi-party scenario and assume that upon the discourse structure, the more immediate utterances may be more relevant to the target one and are more likely to appear in the same thread. Therefore, we calculate the relative distance measured by hops upon the discourse tree and obtain two position matrix P^{intra} and P^{inter} according to the speaker type. And if two nodes are unreachable to each other, we set the position value to zero. Then we adopt a trainable position embedding layer to encode each element in P^{intra} or P^{inter} to a scalar value and add them to the attention scores in the multi-head attention layer.

$$\begin{aligned} Pos^t &= \text{Embedding}(P^t), \\ G &= \text{MHSA}(C,\ IDG^t, Pos^t), \\ \text{Att}(Q, K, V, IDG^t, Pos^t) &= \text{Softmax}(\frac{QK^T}{\sqrt{d_k}} \\ &+ IDG^t + Pos^t)V \end{aligned} \tag{11}$$

Through this, Pos^t could be viewed as an extra weight to guide the fully-connected process of implicit detachment which is considered to be a relatively blind way.

3.5 Emotion Recognition

Finally, taking the the concatenation of C, E and I as input, an emotion classifier is applied to predict the emotion of the utterance.

$$\hat{y} = \text{Softmax}(W_e[C, E, I] + b_e) \tag{12}$$

where W_e and b_e are trainable parameters.

Dataset	Dialogues			Utterances		
	Train	Val	Test	Train	Val	Test
EmoryNLP	713	99	85	9,934	1,344	1,328
MELD	1,039	114	280	9,989	1,109	2,610

Table 1: Dataset statistics

Cross entropy loss is utilized to train the model and the loss function is defined as:

$$L = -\frac{1}{N}\sum_{i=1}^{N}\sum_{j=1}^{Emo}\hat{y_i}^j \cdot log(y_i^j) \quad (13)$$

where Emo is the number of emotion class and y_i^j stands for the ground-truth emotion label of the utterance i.

4 Experiments

4.1 Dataset

We conduct experiments on two ERMC datasets. The statistics of them are shown in Table 1.

EmoryNLP (Zahiri and Choi, 2018): TV show scripts collected from *Friends* and the emotion classes include *neutral*, *sad*, *mad*, *scared*, *powerful*, *peaceful* and *joyful*.

MELD (Poria et al., 2019a): A multimodal dataset with multi-speaker conversations. It is also collected from the TV show *Friends*, but varies from EmoryNLP in the choice of scenes and emotion labels. And the emotion labels belong to *anger*, *disgust*, *fear*, *joy*, *neutral*, *sadness* and *surprise*.

We utilize only the textual modality for the experiments. Following previous works, the weighted F1 score is chosen as evaluation metrics.

4.2 Compared Models

We compare our proposed model with the following methods. CSK is short for the commonsense knowledge:

Methods for ERMC:

ConGCN (Zhang et al., 2019) constructs a large graph over the entire corpus to model both context- and speaker-sensitive dependency.

DialogXL (Shen et al., 2021a) devises four types of dialog-aware self-attention to make the model aware of interactions in multi-party conversations.

ERMC-DisGCN (Sun et al., 2021) builds a graph based on the dialog discourse structure to explore context and speaker-specific features.

Methods for ERC with CSK:

KET (Zhong et al., 2019) leverages commonsense knowledge to enrich context modeling and dynamically retrieves context-aware and emotion-related knowledge.

KAITML (Zhang et al., 2020) applies incremental transformer to encode multi-turn contextual utterances with commonsense knowledge incorporated and introduces multi-task learning to this task.

KI-Net (Xie et al., 2021) concentrates on direct utterance-knowledge interaction and involves additional affective information with an auxiliary task.

SKAIG (Li et al., 2021) constructs a novel graph to explore psychological states of speakers and graph transformer is used to propagate the interactive information over the graph.

COSMIC (Ghosal et al., 2020) explores more types of commonsense knowledge about several factors of conversations to understand emotional dynamics better.

Methods for ERC without CSK:

DialogueRNN (Majumder et al., 2019) devises three states including global state, party state and emotion state with GRUs.

DialogueGCN (Ghosal et al., 2019) uses graph convolutional network to encode context and speaker dependencies.

IEIN (Lu et al., 2020) focuses on explicit interactions among emotion of utterances and iteratively predicts emotion labels based on previous ones.

RGAT (Ishiwatari et al., 2020) enhances relation-aware graph attention network with the proposed relational position encoding.

DialogueCRN (Hu et al., 2021) designs reasoning modules from a cognitive perspective to fully integrate emotional clues.

DAG-ERC (Shen et al., 2021b) presents a idea of modeling conversation context with a directed acyclic graph and proposes a directed acyclic graph neural network.

4.3 Implementation Details

For utterance-level feature extraction, we fine-tune RoBERTa Large model for a batch size of 32 and Adam optimizer is adopted with learning rate of 1e-5. And for the training of MuCDN on emotion recognition, the batch size is set to be {16, 8} for EmoryNLP and MELD respectively. The dimension of hidden representation d_h in the rest of our model is set to 300 and the number of attention head is 6. We train the model with Adam optimizer in a learning rate of 1e-4. All of our results are averaged on 5 runs.

Model	EmoryNLP	MELD
ERMC Methods		
ConGCN	-	57.40
DialogXL	34.73	62.41
ERMC-DisGCN	36.38	64.22
ERC Methods with CSK		
KET	34.39	58.18
KAITML	35.59	58.97
KI-Net	-	63.24
SKAIG	38.88	65.18
COSMIC	38.11	65.21
COSMIC w/o CSK	37.10	64.28
ERC Methods without CSK		
DialogueRNN	31.7	57.03
DialogueGCN	-	58.1
IEIN	-	60.72
RGAT	34.42	60.91
DialogueCRN	-	58.39
DAG-ERC	39.02	63.65
MuCDN (Ours)	**40.09**	**65.37**

Table 2: Comparison of our model against state-of-the-art baselines. CSK represents the commonsense knowledge utilized in COSMIC. Weighted F1 score is adopted as evaluation metrics.

5 Results and Analysis

5.1 Overall Results

Illustrated in Table 2, our proposed model achieves state-of-the-art results on both ERMC datasets.

EmoryNLP. Since there are often more than 5 participants within a conversation in EmoryNLP, which leads to the results that average individual conversation threads derived from Explicit Detachment module are 4.77/4.66/4.91 (Train/Valid/Test), the complicated emotional interactions pose great challenge for capturing contextual clues and speaker features. Benefiting from a clear and effective context and speaker-specific modeling by separating multi-party conversations into individual threads, MuCDN achieves new state-of-the-art weighted F1 scores of 40.09. Compared with DAG-ERC which proposes a directed acyclic graph to link utterances in a locally fully-connected way, distinct detached threads are more suitable and effective for conversation context modeling in multi-party scenario. In addition, the improvement of performance over ERMC-DisGCN demonstrates it may not be effective enough to simply propagate contextual information upon the discourse graph. To a certain extent, such improvement verify the effectiveness of our proposed discourse-based detachment, which is a better way of utilizing discourse structure to clearly perform conversation modeling. And it also suggests the effectiveness of the complementary information provided by implicit detachment and mutual module.

Model	EmoryNLP	MELD
MuCDN	**40.09**	**65.37**
w/o explicit detachment	38.45	64.45
w/o implicit detachment	38.84	64.47
w/o E2I interaction	39.28	64.61
w/o I2E interaction	39.54	64.56

Table 3: Results of ablation study on the two ERMC datasets. E2I interaction is the relative position embedding provided by explicit detachment, while I2E interaction is the complementary global information from implicit detachment.

MELD. Advantages brought by clear conversation modeling could also be observed on MELD where MuCDN performs better than all the baselines. Here, the results of the average detached conversation threads are 3.92/3.96/3.85 (Train/Valid/Test). However, it is worth noting that our model slightly outperforms those baseline models with commonsense knowledge incorporated such as COSMIC. To make a clear comparison regarding the conversation modeling ability of the model, we also compare our model with COSMIC without commonsense knowledge implemented by Shen et al. (2021b). And under the same circumstance of no commonsense knowledge incorporated, the performance advantage of MuCDN comes from the clear and effective capture of emotional clues in individual conversation thread. Meanwhile, it reminds us to enrich contextual representations with commonsense knowledge within each detached thread for further improvement.

5.2 Ablation Study

We conduct ablation studies to verify the effectiveness of different modules proposed in our model. Results are shown in Table 3.

Effect of Conversation Detachment

To investigate the impact of two detachment modules, we remove either the explicit detachment module or the implicit detachment module, and the corresponding mutual interaction is also discarded at the same time. The performance has a certain degree of decline on both datasets and results are displayed in the second row and the third row in Table 3. This manifests that both ways of

Model	EmoryNLP	MELD
MuCDN	**40.09**	**65.37**
sequence	39.05	64.51
randomness	38.72	64.71

Table 4: Results of our model replaced with different types of dependency structure connecting utterances in Explicit Detachment module.

Model	EmoryNLP	MELD
MuCDN	**40.09**	**65.37**
w/o intra and inter GRU	39.42	64.49
w/o intra and inter graph	38.91	64.46

Table 5: Results of our model without speaker-specific modeling.

explicit detachment and implicit detachment play an important role in simplifying and clarifying the complex interactions within multi-party conversations and lay the foundation for effective context and speaker-specific modeling.

Effect of Mutual Interaction

To verify the effectiveness of the mutual interaction that explicit detachment and implicit detachment provide for each other, the relative position embedding and the complementary global information are removed individually. The dropped results in the second-to-last row on both ERMC datasets demonstrates that the extra relative position features derived from the dialogue discourse structure provide implicit detachment with explicit guidance to capture the latent emotional interactions better. Besides, results in the last row suggests that the global information from implicit detachment complement multi-source emotional interactions for the context modeling of each thread.

5.3 Variants of Dependency Structure in Explicit Detachment

In this section, we investigate how explicit detachment benefits from dialogue discourse structure for a clear and effective conversation modeling. Two additional kinds of dependency structure of utterances are devised: (1) sequence, in which utterances are connected one by one following the temporal order; (2) randomness, in which each utterance link any one of the previous utterances with a random selection. To achieve this, the detachment matrix D is substituted and the position matrix P in mutual module is also changed to be consistent with such two types of dependency structure. We keep all the rest parts same with our complete model.

The test performance is shown in Table 4 and we make two instructive observations from the experimental results. First, the performance of random structure is not as terrible as we expected, even a little better than that of sequential structure on MELD dataset. On the one hand, it manifests that the effectiveness of the complementary global information provided by implicit detachment and latent emotional interactions among utterances are captured. On the other hand, the little improvement over the structure of sequence suggests that it may not be suitable enough to plainly model the conversational context without any explicit detachment for the entangled multi-party conversations. Second, both results obtained upon the variants of dependency structure do not decrease to a large margin. It reminds us that the results of the dialogue discourse parser are not reliable enough and the problem of error propagation may have influence on our model. And effective techniques to prune the dialogue discourse tree should be explored for future work.

5.4 Analysis of Speaker-Specific Modeling

As a key factor to provide emotional clues for emotion recognition, especially under the circumstance of multi-party setting, modeling speaker-specific information has proven to be beneficial. We investigate the impact of speaker-aware modules in our proposed model. First, instead of utilizing intra-speaker GRU and inter-speaker GRU in explicit detachment, we only adopt a single GRU without the speaker-aware identification for contextual information propagating through each detached thread. Also, only one implicit detachment graph is constructed without speaker information injected to identify connections among utterances.

According to results shown in Table 5, the drop of performance implies that it is effective to take speaker-specific information into account when carrying out conversational context modeling. And it also suggests the importance of figuring out the intra and inter-speaker dependency to capture emotional clues. But such a relatively slight drop inspires us that the identification of whether two utterances have the same speaker may not be sufficient in multi-party scenario. Richer speaker-specific information contained in the contextual context should be excavated further.

6 Conclusion

In this paper, in order to capture emotional clues for emotion dynamics in a clear and effective way under the circumstance of multi-party scenario, we propose Mutual Conversational Detachment Network (MuCDN) for emotion recognition in multi-party conversations. Joint context and speaker-specific modeling is performed within individual detached threads by two detachment ways and a mutual module. Experimental results on two ERMC datasets demonstrate the superiority of our proposed MuCDN. Also, by conducting comprehensive evaluations and ablation study, we confirmed the effectiveness of our MuCDN and the impact of its components.

For future work, we would like to explore effective techniques to prune the dialogue discourse tree to alleviate error propagation. Moreover, for a more comprehensive speaker-specific modeling, richer speaker information in the conversational context should be excavated further.

Acknowledgments

We thank the anonymous reviewers for their insightful comments and suggestions. This work was supported by the National Key RD Program of China via grant 2020AAA0106501 and the National Natural Science Foundation of China (NSFC) via grant 62176078.

References

Nicholas Asher, Julie Hunter, Mathieu Morey, Farah Benamara, and Stergos D. Afantenos. 2016. Discourse structure and dialogue acts in multiparty dialogue: the STAC corpus. In *Proceedings of the Tenth International Conference on Language Resources and Evaluation LREC 2016, Portorož, Slovenia, May 23-28, 2016*. European Language Resources Association (ELRA).

Deepanway Ghosal, Navonil Majumder, Alexander F. Gelbukh, Rada Mihalcea, and Soujanya Poria. 2020. COSMIC: commonsense knowledge for emotion identification in conversations. In *Findings of the Association for Computational Linguistics: EMNLP 2020, Online Event, 16-20 November 2020*, volume EMNLP 2020 of *Findings of ACL*, pages 2470–2481. Association for Computational Linguistics.

Deepanway Ghosal, Navonil Majumder, Soujanya Poria, Niyati Chhaya, and Alexander F. Gelbukh. 2019. Dialoguegcn: A graph convolutional neural network for emotion recognition in conversation. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing, EMNLP-IJCNLP 2019, Hong Kong, China, November 3-7, 2019*, pages 154–164. Association for Computational Linguistics.

Dou Hu, Lingwei Wei, and Xiaoyong Huai. 2021. Dialoguecrn: Contextual reasoning networks for emotion recognition in conversations. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing, ACL/IJCNLP 2021, (Volume 1: Long Papers), Virtual Event, August 1-6, 2021*, pages 7042–7052. Association for Computational Linguistics.

Taichi Ishiwatari, Yuki Yasuda, Taro Miyazaki, and Jun Goto. 2020. Relation-aware graph attention networks with relational position encodings for emotion recognition in conversations. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing, EMNLP 2020, Online, November 16-20, 2020*, pages 7360–7370. Association for Computational Linguistics.

Jiangnan Li, Zheng Lin, Peng Fu, and Weiping Wang. 2021. Past, present, and future: Conversational emotion recognition through structural modeling of psychological knowledge. In *Findings of the Association for Computational Linguistics: EMNLP 2021, Virtual Event / Punta Cana, Dominican Republic, 16-20 November, 2021*, pages 1204–1214. Association for Computational Linguistics.

Hui Liu, Zhan Shi, and Xiaodan Zhu. 2021a. Unsupervised conversation disentanglement through co-training. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 2345–2356, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Longxiang Liu, Zhuosheng Zhang, Hai Zhao, Xi Zhou, and Xiang Zhou. 2021b. Filling the gap of utterance-aware and speaker-aware representation for multi-turn dialogue. In *Thirty-Fifth AAAI Conference on Artificial Intelligence, AAAI 2021, Thirty-Third Conference on Innovative Applications of Artificial Intelligence, IAAI 2021, The Eleventh Symposium on Educational Advances in Artificial Intelligence, EAAI 2021, Virtual Event, February 2-9, 2021*, pages 13406–13414. AAAI Press.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Roberta: A robustly optimized BERT pretraining approach. *CoRR*, abs/1907.11692.

Xin Lu, Yanyan Zhao, Yang Wu, Yijian Tian, Huipeng Chen, and Bing Qin. 2020. An iterative emotion interaction network for emotion recognition in conversations. In *Proceedings of the 28th International Conference on Computational Linguistics, COLING 2020, Barcelona, Spain (Online), December 8-13,*

2020, pages 4078–4088. International Committee on Computational Linguistics.

Xinbei Ma, Zhuosheng Zhang, and Hai Zhao. 2021. Structural modeling for dialogue disentanglement. *CoRR*, abs/2110.08018.

Navonil Majumder, Soujanya Poria, Devamanyu Hazarika, Rada Mihalcea, Alexander F. Gelbukh, and Erik Cambria. 2019. Dialoguernn: An attentive RNN for emotion detection in conversations. In *The Thirty-Third AAAI Conference on Artificial Intelligence, AAAI 2019, The Thirty-First Innovative Applications of Artificial Intelligence Conference, IAAI 2019, The Ninth AAAI Symposium on Educational Advances in Artificial Intelligence, EAAI 2019, Honolulu, Hawaii, USA, January 27 - February 1, 2019*, pages 6818–6825. AAAI Press.

Soujanya Poria, Devamanyu Hazarika, Navonil Majumder, Gautam Naik, Erik Cambria, and Rada Mihalcea. 2019a. MELD: A multimodal multi-party dataset for emotion recognition in conversations. In *Proceedings of the 57th Conference of the Association for Computational Linguistics, ACL 2019, Florence, Italy, July 28- August 2, 2019, Volume 1: Long Papers*, pages 527–536. Association for Computational Linguistics.

Soujanya Poria, Navonil Majumder, Rada Mihalcea, and Eduard H. Hovy. 2019b. Emotion recognition in conversation: Research challenges, datasets, and recent advances. *IEEE Access*, 7:100943–100953.

Weizhou Shen, Junqing Chen, Xiaojun Quan, and Zhixian Xie. 2021a. Dialogxl: All-in-one xlnet for multi-party conversation emotion recognition. In *Thirty-Fifth AAAI Conference on Artificial Intelligence, AAAI 2021, Thirty-Third Conference on Innovative Applications of Artificial Intelligence, IAAI 2021, The Eleventh Symposium on Educational Advances in Artificial Intelligence, EAAI 2021, Virtual Event, February 2-9, 2021*, pages 13789–13797. AAAI Press.

Weizhou Shen, Siyue Wu, Yunyi Yang, and Xiaojun Quan. 2021b. Directed acyclic graph network for conversational emotion recognition. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing, ACL/IJCNLP 2021, (Volume 1: Long Papers), Virtual Event, August 1-6, 2021*, pages 1551–1560. Association for Computational Linguistics.

Zhouxing Shi and Minlie Huang. 2019. A deep sequential model for discourse parsing on multi-party dialogues. In *The Thirty-Third AAAI Conference on Artificial Intelligence, AAAI 2019, The Thirty-First Innovative Applications of Artificial Intelligence Conference, IAAI 2019, The Ninth AAAI Symposium on Educational Advances in Artificial Intelligence, EAAI 2019, Honolulu, Hawaii, USA, January 27 - February 1, 2019*, pages 7007–7014. AAAI Press.

Yang Sun, Nan Yu, and Guohong Fu. 2021. A discourse-aware graph neural network for emotion recognition in multi-party conversation. In *Findings of the Association for Computational Linguistics: EMNLP 2021, Virtual Event / Punta Cana, Dominican Republic, 16-20 November, 2021*, pages 2949–2958. Association for Computational Linguistics.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, December 4-9, 2017, Long Beach, CA, USA*, pages 5998–6008.

Yunhe Xie, Kailai Yang, Chengjie Sun, Bingquan Liu, and Zhenzhou Ji. 2021. Knowledge-interactive network with sentiment polarity intensity-aware multitask learning for emotion recognition in conversations. In *Findings of the Association for Computational Linguistics: EMNLP 2021, Virtual Event / Punta Cana, Dominican Republic, 16-20 November, 2021*, pages 2879–2889. Association for Computational Linguistics.

Zhilin Yang, Zihang Dai, Yiming Yang, Jaime G. Carbonell, Ruslan Salakhutdinov, and Quoc V. Le. 2019. Xlnet: Generalized autoregressive pretraining for language understanding. In *Advances in Neural Information Processing Systems 32: Annual Conference on Neural Information Processing Systems 2019, NeurIPS 2019, December 8-14, 2019, Vancouver, BC, Canada*, pages 5754–5764.

Sayyed M. Zahiri and Jinho D. Choi. 2018. Emotion detection on TV show transcripts with sequence-based convolutional neural networks. In *The Workshops of the The Thirty-Second AAAI Conference on Artificial Intelligence, New Orleans, Louisiana, USA, February 2-7, 2018*, volume WS-18 of *AAAI Technical Report*, pages 44–52. AAAI Press.

Dong Zhang, Liangqing Wu, Changlong Sun, Shoushan Li, Qiaoming Zhu, and Guodong Zhou. 2019. Modeling both context- and speaker-sensitive dependence for emotion detection in multi-speaker conversations. In *Proceedings of the Twenty-Eighth International Joint Conference on Artificial Intelligence, IJCAI 2019, Macao, China, August 10-16, 2019*, pages 5415–5421. ijcai.org.

Duzhen Zhang, Xiuyi Chen, Shuang Xu, and Bo Xu. 2020. Knowledge aware emotion recognition in textual conversations via multi-task incremental transformer. In *Proceedings of the 28th International Conference on Computational Linguistics, COLING 2020, Barcelona, Spain (Online), December 8-13, 2020*, pages 4429–4440. International Committee on Computational Linguistics.

Duzhen Zhang, Zhen Yang, Fandong Meng, Xiuyi Chen, and Jie Zhou. 2022. TSAM: A two-stream attention model for causal emotion entailment. *CoRR*, abs/2203.00819.

Weixiang Zhao, Yanyan Zhao, and Xin Lu. 2022. Cauain: Causal aware interaction network for emotion recognition in conversations. In *Proceedings of the Thirty-First International Joint Conference on Artificial Intelligence, IJCAI 2022, Vienna, Austria, 23-29 July 2022*, pages 4524–4530. ijcai.org.

Peixiang Zhong, Di Wang, and Chunyan Miao. 2019. Knowledge-enriched transformer for emotion detection in textual conversations. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing, EMNLP-IJCNLP 2019, Hong Kong, China, November 3-7, 2019*, pages 165–176. Association for Computational Linguistics.

UECA-Prompt: Universal Prompt for Emotion Cause Analysis

Xiaopeng Zheng[1], **Zhiyue Liu**[1,2], **Zizhen Zhang**[1], **Zhaoyang Wang**[1], **Jiahai Wang**[1]*
[1]School of Computer Science and Engineering, Sun Yat-sen University
Guangzhou, China
[2]School of Computer, Electronics and Information, Guangxi University
Nanning, China
{zhengxp26,liuzhy93,wangzhaoy22}@mail2.sysu.edu.cn,
{wangjiah,zhangzzh7}@mail.sysu.edu.cn

Abstract

Emotion cause analysis (ECA) aims to extract emotion clauses and find the corresponding cause of the emotion. Existing methods adopt fine-tuning paradigm to solve certain types of ECA tasks. These task-specific methods have a deficiency of universality. And the relations among multiple objectives in one task are not explicitly modeled. Moreover, the relative position information introduced in most existing methods may make the model suffer from dataset bias. To address the first two problems, this paper proposes a universal prompt tuning method to solve different ECA tasks in the unified framework. As for the third problem, this paper designs a directional constraint module and a sequential learning module to ease the bias. Considering the commonalities among different tasks, this paper proposes a cross-task training method to further explore the capability of the model. The experimental results show that our method achieves competitive performance on the ECA datasets.

1 Introduction

Recently, emotion cause analysis (ECA) has obtained increasing attention. As a classic task of ECA, emotion cause extraction (ECE) was first proposed and defined as a clause-level classification problem (Gui et al., 2016a). ECE aims to explore the potential causes behind a certain emotional expression in a clause. However, the applications of ECE are limited in real-world scenarios because the emotion must be annotated before cause extraction. Therefore, emotion-cause pair extraction (ECPE) was proposed to identify all emotions and their corresponding causes from an unannotated text at the same time (Xia and Ding, 2019). Chen et al. (2020a) proposed conditional causal relationships classification (CCRC) to explore the relation of emotion-cause pairs and contexts. Although there are other emerging tasks proposed (Li et al., 2021b; Bi and Liu, 2020) to expand the applications of ECA, this paper focuses on the most complex and challenging three clause-level sub-tasks: ECE, ECPE, and CCRC.

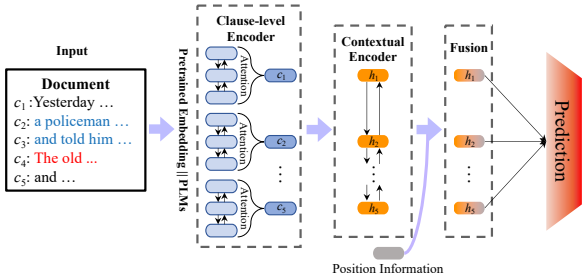

Figure 1: Fine-tuning for emotion cause analysis.

In literature, ECA tasks have been widely studied. On the ECE task, Gui et al. (2016a) released a benchmark Chinese emotion cause dataset and proposed a multi-kernel based method. Several statistical learning (Gui et al., 2016b; Xu et al., 2017) and deep learning methods (Cheng et al., 2017; Hu et al., 2021b) have been applied to ECE, which show competitive performance on emotion cause prediction. On the ECPE task, Xia and Ding (2019) proposed a two-step method to extract emotions and corresponding causes at the same time. This method first individually extracts the emotion set and cause set. Then, it gets the emotion-cause pairs by applying a Cartesian product to these two sets and trains a filter to remove the invalid pairs. However, the error may propagate from the first procedure to the second because of the inherent drawback of the pipelined framework. To address this issue, several works adopted end-to-end architecture. Part of these works (Tang et al., 2020; Wu et al., 2020) focused on the multi-task learning (Caruana, 1997) of the ECPE task with the joint learning framework (Ding et al., 2020a,b). Some other works represented the relation between emotion and cause with graph construction (Wei et al., 2020; Chen et al., 2020c). In addition, transition-

*Corresponding author

Proceedings of the 29th International Conference on Computational Linguistics, pages 7031–7041
October 12–17, 2022.

based parsing (Fan et al., 2020) and unified sequence labeling (Chen et al., 2020b; Cheng et al., 2021) are also employed. On the CCRC task, the relationship between emotion and cause clauses in different contexts is studied (Chen et al., 2020a).

The existing methods could be divided into multi-stage framework (Xia and Ding, 2019; Xu et al., 2021) and end-to-end framework (Ding et al., 2020a; Wei et al., 2020). Both frameworks adopt the fine-tuning paradigm. As shown in Fig. 1, these fine-tuning methods firstly obtain clause-level features from the pre-trained embedding or pre-trained language models (PLMs). Next, a contextual encoder is designed to yield contextual representations of clauses. Then, the contextual representation is merged with clause-level position information. Finally, the output is used for task prediction. In general, the existing methods have three obvious shortcomings: firstly, the contextual encoder has the deficiency of universality because they are designed for specific task objectives. Secondly, the existing methods implicitly learn the relationships among multiple task objectives, rather than explicitly model their relations. Thirdly, position information would make the model sensitive to data distribution and lack robustness.

Prompt (Liu et al., 2021; Han et al., 2021; Houlsby et al., 2019) is a new paradigm that can be traced back to GPT-3 (Brown et al., 2020). This paradigm could overcome the difficulty of uniformly solving various tasks by transforming specific fine-tuning tasks into the form of pre-training tasks. The universality of prompt has been widely validated in various tasks of natural language processing (Schick et al., 2020; Ma et al., 2021; Li et al., 2021a; Li and Liang, 2021), especially in few-shot (Gao et al., 2021; Zhang et al., 2021) or zero-shot (Sanh et al., 2021; Wei et al., 2021) scenarios. Multiple predictions should be performed for one sample with several clauses in ECA. And ECA is composed based on multiple task objectives (e.g. emotion extraction, cause extraction, emotion-cause pair extraction). Appropriate prompt construction could complete multiple predictions simultaneously. Constructing special prompts could explicitly model the relations among multiple objectives in one task. In general, prompt could make up for the shortcomings of existing methods.

Inspired by this, this paper proposes a universal prompt-based method for ECA tasks (UECA-Prompt). UECA-Prompt first modifies each task objective into a sub-prompt after decomposing ECA tasks into multiple task objectives. Then, it explicitly models the relations among different task objectives by combining different sub-prompts into a composite prompt. As most of the works (Petroni et al., 2020; Schick and Schütze, 2020a,b), UECA-Prompt is manually constructed to solve different ECA tasks in a unified framework. As far as we know, UECA-Prompt is the first attempt at multi-task multiple predictions with prompt.

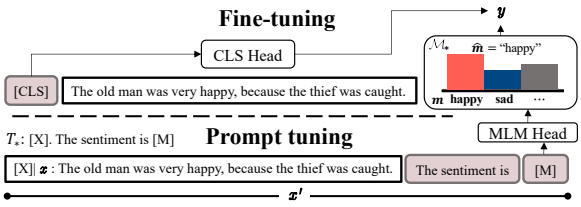

Figure 2: Different task forms for prompt tuning and fine-tuning.

Some previous works (Ding et al., 2020b; Chen et al., 2020c) artificially introduced position information into the model. Such characteristic would make the model extremely sensitive to the distribution of positions of cause clauses relative to their corresponding emotion clauses in the dataset. Specifically, existing methods with position information may not generalize well to de-bias dataset (Ding and Kejriwal, 2020). Because the cause clause of most samples in the de-bias dataset is not in proximity to the emotion clause. This paper designs a directional constraint module and a sequential learning module to better identify the emotion-cause pairs. These two modules could ease the bias by discarding position information between clauses.

There are commonalities among different tasks. For example, CCRC focuses more on context information, and context information is also crucial for ECE and ECPE. This paper proposes a cross-task training method. The model would be able to learn commonalities among tasks with the cross-task training method.

We evaluate our method on three benchmark Chinese emotion cause datasets. The experimental results show that our method can obtain better results than state-of-the-art methods on three ECA tasks solely based on the BERT.

Our contributions are summarized as follows:

- We propose a universal prompt method for a variety of ECA tasks, such as ECE, ECPE, and

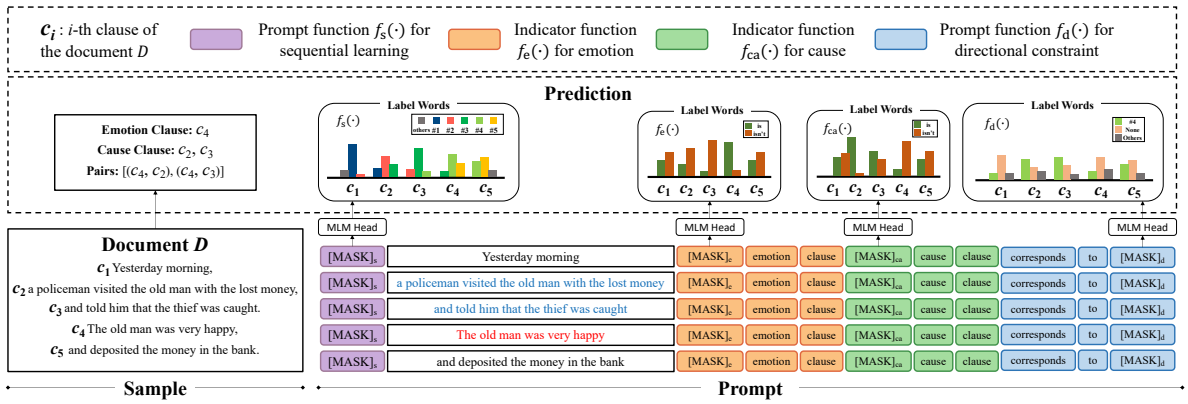

Figure 3: Overview of UECA-Prompt. The answer slot [M] of template is replaced by the token [MASK]. Subscripts are added for token [MASK] to distinguish different sub-prompt. At the bottom of the figure, input $\boldsymbol{x}$ is presented on the left, and prompt $\boldsymbol{x'}$ constructed from the input text is on the right. The predictions of each sub-prompt are presented above the corresponding module.

Function $f_*(\cdot)$	Template $T_*(\cdot)$	Label Words $\mathcal{M}_*(\cdot)$
$f_{\text{e}}(\cdot)$	$T_{\text{e}}(\cdot)$ = "[X] [M] emotion clause"	$\mathcal{M}_{\text{e}}(\cdot)$ = {"is", "isn't"}
$f_{\text{ca}}(\cdot)$	$T_{\text{ca}}(\cdot)$ = "[X] [M] cause clause"	$\mathcal{M}_{\text{ca}}(\cdot)$ = {"is", "isn't"}
$f_{\text{d}}(\cdot)$	$T_{\text{d}}(\cdot)$ = "[X] corresponds to [M]"	$\mathcal{M}_{\text{d}}(\cdot)$ = {"None", "1", "2", ..., "n"}
$f_{\text{s}}(\cdot)$	$T_{\text{s}}(\cdot)$ = "[M] [X]"	$\mathcal{M}_{\text{s}}(\cdot)$ = {"1", "2", ..., "n"}

Table 1: Different sub-prompt in UECA-Prompt. [X] is input slot filled with text $\boldsymbol{x}$ and [M] is answer slot for prediction. We translate the original Chinese words into English for better illustration.

CCRC. It is the first attempt to solve multi-task multiple predictions with prompt.

- We design the directional constraint and sequential learning module to ease the bias effect caused by position information, making UECA-Prompt more robust toward the de-bias dataset.

- Experimental results show that UECA-Prompt outperforms the state-of-the-art methods. And the cross-task training method further improves the performance of UECA-Prompt.

2 Preliminary

2.1 Task Definition

Given a document $D = \{c_1, ..., c_n\}$ with n clauses, where c_i is the i-th clause. ECE is to determine the cause clause set $C_{\text{ca}} \subseteq D$ according to a given emotion clause $c_{\text{e}} \in D$. The cause clause set C_{ca} may have more than one clause. ECPE is to identify each emotion-cause pair $\{c_{\text{e}}, c_{\text{ca}}\}$ in the document. CCRC determines whether the emotion-cause pair $\{c_{\text{e}}, c_{\text{ca}}\}$ still has a causal relationship under different context within a set $\mathcal{T} = \{t_1, ..., t_j\}$, where context t_j is the residual document except for the emotion clauses c_{e} and cause clauses c_{ca}.

2.2 Prompt Tuning

As shown in Fig. 2, a typical prompt consists of a template $T_*(\cdot)$ (e.g. "[X]. The sentiment is [M]") and a set of permissible values $\mathcal{M}_*(\cdot)$ (e.g. "happy", "sad"). Firstly, a prompt function $f_*(\cdot)$ fills the input slot [X] with original input $\boldsymbol{x}$ to get $\boldsymbol{x'}$ (e.g. "The old man was very happy, because the thief was caught. The sentiment is [M]"). Secondly, an argmax is used to search for the highest-scoring intermediate result $\hat{\boldsymbol{m}}$ from a set of label words $\mathcal{M}_*(\cdot)$. $\mathcal{M}_*(\cdot)$ is a set of potential answers for answer slot [M]. Finally, $\hat{\boldsymbol{m}}$ would be mapped into final result $\boldsymbol{y}$.

3 Method

UECA-Prompt is a BERT-based method constructed as the form of the MLM task. This paper would discuss UECA-Prompt from two perspectives, sub-prompt for task decomposition and composite prompt for multiple predictions.

3.1 Sub-prompt for Task Decomposition

ECA involves multi-task learning and could be decomposed into emotion extraction, cause extrac-

tion, and emotion-cause pair extraction. This paper designs a sub-prompt for each objective. The sub-prompts include indicator functions, directional constraint module, and sequential token module.

Indicator Function In ECA, we are requested to find all the emotion-cause pairs in the text. Empirically, we should first search the text for all the emotion clauses. Then the cause clauses could be identified according to those emotion clauses. Finally, the causal relation between emotion and cause clauses can be checked depending on the context.

Two clause-level indicator functions, $f_{\text{e}}(\cdot)$ and $f_{\text{ca}}(\cdot)$, are designed to extract emotion and cause. Specifically, the indicator function $f_{\text{e}}(c_i)$ determines whether a clause c_i is an emotion clause, while $f_{\text{ca}}(c_i)$ determines the cause clause. The orange rectangles and green rectangles in Fig. 3 show these two sub-prompts, and the answer slot [M] is replaced by token [MASK]. The sub-prompt templates and label words for $f_{\text{e}}(\cdot)$ and $f_{\text{ca}}(\cdot)$ are formalized as first and second rows of Table 1.

The candidate label words for predicting answer [M] include "is" and "isn't". "is" represents the current clause belonging to the target set, and "isn't" represents there is no subordinate relationship.

A directional constraint module and sequential module are proposed to further extract the emotion-cause pair. Different from the usual sub-prompt, these two modules could be regarded as a pointer structure when combined. It effectively alleviates bias caused by position information.

Directional Constraint As the blue rectangles shown in Fig. 3, the template and label words of $f_{\text{d}}(\cdot)$ can be formalized as third row of Table 1. In the label words set $\mathcal{M}_{\text{d}}(\cdot)$, "None" indicates that there is no clause associated with the current clause. And numeric token "i" represents the current clause associated with the i-th clause. n represents the number of clauses in the document.

The numeric tokens in $\mathcal{M}_{\text{d}}(\cdot)$ indicate the sequence information of clauses. A sequential learning module is designed to facilitate the model acquiring such knowledge.

Sequential Learning In sequential learning, we set a prefix answer slot [M] for each clause. The numeric token of the answer slot for each clause is a unique identifier. The model could learn the unique identifier with sequential learning. As the purple rectangles shown in Fig. 3, a sequential function $f_{\text{s}}(\cdot)$ is designed for sequential learning. The template and label words could be formalized as the last rows of Table 1. Intermediate result $\hat{\boldsymbol{m}}$ for i-th clause is "i". Label words for sequential learning are also used in directional constraint function (i.e., $\mathcal{M}_{\text{s}}(\cdot) \subseteq \mathcal{M}_{\text{d}}(\cdot)$). Sequential learning is only used in the training stage.

3.2 Composite Prompt for Multiple Predictions

A composite prompt comprised of all the sub-prompt (Liu et al., 2021) is defined to address different multiple prediction tasks in a unified prompt.

Composite Prompt Composite prompt explicitly models the relation among different task objectives. The template of composite prompt $f_{\text{CP}}(\cdot)$ is formalized as:

$$T_{\text{CP}}(\cdot) = \text{"Clause [M] [X] [M] emotion clause [M] cause clause corresponds to [M]"}, \quad (1)$$

where the label words for each answer slot [M] in composite prompt template $T_{\text{CP}}(\cdot)$ correspond to the label words of each sub-prompt, respectively.

Filling the composite template with document D to form prompted document $\boldsymbol{D'}$ could be formalized as:

$$\boldsymbol{D'} = [f_{\text{CP}}(c_1); ...; f_{\text{CP}}(c_n)], \quad (2)$$

where c_i is the i-th clause in document D, and $[\cdot;\cdot]$ is the concatenation operation.

Multiple Predictions In ECA tasks, multiple predictions should be performed for each clause. UECA-prompt is capable of multiple predictions. As shown in Fig. 3, firstly, we convert the input doc into a set of clauses. Secondly, the template of composite prompt is applied to each clause in the input document D. Thirdly, the intermediate answer for each sub-prompt is searched to separately predict the answer of slot [M]. Finally, the intermediate answers are aggregated and mapped to the final result. This paper explores composite prompt (See Eq. (1)) for three ECA tasks (i.e., ECPE, ECE, and CCRC). Since the operations on each clause are the same, this paper will discuss the prompt on a single clause in different tasks.

Implementation for ECPE The composite prompt function instantiated for the ECPE task is given in Eq. (3). Unrelated prompt tokens are omitted for clarity. Subscripts are added for the answer

Method	ECPE			Emotion Extraction			Cause Extraction		
	F_1 (%)	P (%)	R (%)	F_1 (%)	P (%)	R (%)	F_1 (%)	P (%)	R (%)
Inter-EC†	61.28	67.21	57.05	82.30	83.64	81.07	65.07	70.41	60.83
TransECPE†	67.99	73.74	63.07	84.74	87.16	82.44	69.74	75.62	64.71
UTOS-BERT†	72.03	73.89	70.62	85.56	88.15	83.21	74.71	76.71	73.20
PairGCN-BERT†	72.02	76.92	67.91	83.75	88.57	79.58	73.75	79.07	69.28
RANK-CP-bert†	73.60	71.19	76.30	**90.57**	**91.23**	89.99	76.15	74.61	77.88
ECPE-MLL-bert†	74.52	77.00	72.35	88.86	86.08	91.91	76.30	73.82	79.12
MTST+Refinement†	74.63	**77.46**	71.99	84.36	87.11	81.78	76.66	**79.47**	74.04
UECA-Prompt	**74.70***	71.82	**77.99***	88.16	84.75	**91.95***	**77.55***	76.24	**79.16***
UECA-Prompt (m2m)	73.41	70.19	77.19	87.81	85.45	90.43	75.84	74.13	77.97

Table 2: Experimental results on the ECPE task. UECA-Prompt (m2m) is the result of UECA-Prompt with the M2M module. The best result is marked in **bold**. † indicates the results are reported in the original paper. $*$ indicates statistically significant improvement ($p < 0.01$) over the best baseline.

Method	F_1 (%)	P (%)	R (%)
UECA-Prompt	61.14	69.52	54.66
UECA-Prompt (m2m)	**63.45**	**73.92**	**56.30**

Table 3: Experimental results on multi-emotion samples of the ECPE task.

Method	F_1 (%)	P (%)	R (%)
Multi-kernel†	67.52	65.88	69.72
MANN†	77.06	78.43	75.87
RTHN (Layer 3)†	76.77	76.97	76.62
FSS-GCN†	78.61	75.72	77.14
EF-BHA†	78.68	79.38	78.08
RHNN†	79.14	81.12	77.25
UECA-Prompt	**84.40***	**84.57***	**84.27***

Table 4: Experimental results on the ECE task. * indicates statistically significant improvement ($p < 0.001$) over the best baseline.

slot [M] to distinguish different sub-prompt:

$$f_{\text{ECPE}}(c_i) = \text{“}[\text{M}]_{\text{s}}\ c_i\ [\text{M}]_{\text{e}}\ [\text{M}]_{\text{ca}}\ [\text{M}]_{\text{d}}\text{”}. \quad (3)$$

The prediction of the emotion and cause could be mapped from the intermediate result of slot $[\text{M}]_{\text{e}}$ and $[\text{M}]_{\text{ca}}$ in indicator functions, formalized as:

$$P_{\text{e}}(c_i) = \begin{cases} 1, \ \hat{\boldsymbol{m}}_{\text{e}} = \text{“is”} \\ 0, \ \hat{\boldsymbol{m}}_{\text{e}} = \text{“isn't”} \end{cases}, \quad P_{\text{ca}}(c_i) = \begin{cases} 1, \ \hat{\boldsymbol{m}}_{\text{ca}} = \text{“is”} \\ 0, \ \hat{\boldsymbol{m}}_{\text{ca}} = \text{“isn't”} \end{cases}, \quad (4)$$

where $\hat{\boldsymbol{m}}_{\text{e}}$ and $\hat{\boldsymbol{m}}_{\text{ca}}$ are highest-scoring outputs of search the function for slot $[\text{M}]_{\text{e}}$ and $[\text{M}]_{\text{ca}}$, respectively. The searching function is formalized as:

$$\hat{\boldsymbol{m}} = \underset{\boldsymbol{m} \in \mathcal{M}}{\arg\max}\ p(f(\boldsymbol{x'}, \boldsymbol{m}); \theta) \quad (5)$$

where function $f(\boldsymbol{x'}, \boldsymbol{m})$ is used to fill the answer slot [M] in prompt $\boldsymbol{x'}$ with potential answer $\boldsymbol{m}$, and θ represents the parameters of PLMs.

Method	F_1 (%)	P (%)	R (%)
BiLSTM+Concatenation†	61.27	54.12	71.19
BiLSTM+BiLSTM †	69.76	66.06	74.00
BiLSTM+Self-Attention†	66.05	57.66	77.70
UECA-Prompt	**81.18***	**76.35***	**86.76***

Table 5: Experimental results on the CCRC task. * indicates statistically significant improvement ($p < 0.001$) over the best baseline.

By combining the intermediate results of $[\text{M}]_{\text{ca}}$ and $[\text{M}]_{\text{d}}$, the prediction of emotion-cause pairs could be formalized as:

$$P_{\text{pair}}(c_i) = \begin{cases} (i,\ j),\ \hat{\boldsymbol{m}}_{\text{d}} = \text{“}j\text{”} \text{ and } \hat{\boldsymbol{m}}_{\text{ca}} = \text{“is”} \\ \text{null, otherwise,} \end{cases} \quad (6)$$

where $(i,\ j)$ represents that the i-th and j-th clauses constitute an emotion-cause pair (the former is the cause and the latter is the emotion), and “null” represents there is no clause associated with the current clause.

In some cases, multiple emotions correspond to one cause. This paper further designs a many-to-many (M2M) module to deal with this situation. The variant prompt template with the M2M module is as follows:

$$f^{\text{m2m}}_{\text{ECPE}}(c_i) = \text{“}[\text{M}]_{\text{s}}\ c_i\ [\text{M}]_{\text{e}}\ [\text{M}]_{\text{ca}}\ [\text{M}]_{\text{d}_1}\ \ldots\ [\text{M}]_{\text{d}_M}\text{”}, \quad (7)$$

where M is the maximum number of pairs in one document, and M is set to 3 in the experiment.

Implementation for ECE The emotion clause is annotated in the ECE task. This paper replaces the slot $[\text{M}]_{\text{e}}$ in Eq. (3) with a specific token (“is” for the emotion clause and “isn't” for others). The composite prompt function instantiated for ECE is as follows:

$$f_{\text{ECE}}(c_i) = \text{“}[\text{M}]_{\text{s}}\ c_i \text{ is/isn't } [\text{M}]_{\text{ca}}\ [\text{M}]_{\text{d}}\text{”}, \quad (8)$$

Method	ECPE			Emotion Extraction			Cause Extraction		
	F_1 (%)	P (%)	R (%)	F_1 (%)	P (%)	R (%)	F_1 (%)	P (%)	R (%)
UTOS-BERT	34.14	42.76	28.95	66.45	**83.69**	56.08	36.29	45.83	30.55
ECPE-MLL-bert	45.57	**61.53**	36.39	**80.33**	77.15	84.04	**54.68**	51.40	**59.01**
MTST+Refinement	44.93	51.99	40.34	64.10	77.65	55.46	48.84	**56.25**	43.96
UECA-Prompt	**49.37**	46.30	**53.22**	76.15	68.68	**85.67**	50.64	45.75	57.71

Table 6: Experimental results on the ECPE task under few-shot setting.

Method	F_1 (%)	P (%)	R (%)
PADGL	66.16	66.16	67.52
RTHN (Layer 3)	62.49	61.72	63.48
UECA-Prompt	**72.21**	**72.10**	**72.54**

Table 7: Experimental results on the ECE task under few-shot setting.

Method	F_1 (%)	P (%)	R (%)
BiLSTM+Concatenation	66.13	49.53	**99.53**
BiLSTM+BiLSTM	62.41	52.00	78.17
BiLSTM+Self-Attention	62.45	51.54	79.40
UECA-Prompt	**67.56**	**55.46**	86.70

Table 8: Experimental results on the CCRC task under few-shot setting.

where is/isn't is determined by whether the current sentence is an emotion clause, and slot $[\mathtt{M}]_{\mathrm{d}}$ is replaced by the numeric token of the emotion clause in the testing stage.

The prediction of cause is the same as ECPE (see Eq. (4)).

Implementation for CCRC The emotion-cause pair is annotated in the CCRC task. Thus, the slot $[\mathtt{M}]_{\mathrm{ca}}$ in Eq. (8) is also replaced by a specific token ("is" for cause clauses and "isn't" for others) as the ECE task. The composite prompt function instantiated for CCRC is as follows:

$$f_{\mathrm{CCRC}}(c_i) = \text{“}[\mathtt{M}]_{\mathrm{s}}\ c_i \text{ is/isn't is/isn't } [\mathtt{M}]_{\mathrm{d}}\text{”}\ , \quad (9)$$

Different from ECE and ECPE, we are required to tell the causal relationship between an emotion clause and multiple cause clauses in CCRC. Thus, a voting mechanism is proposed. The final result is co-determined by the intermediate result of answer slot $[\mathtt{M}]_{\mathrm{d}}$ in Eq. (9) for all the cause clauses. Based on the voting mechanism, the prediction formula of CCRC is as follows:

$$\mu_{\mathrm{vote}}(D) = \frac{1}{|C_{\mathrm{ca}}|} \sum_{c_i \in C_{\mathrm{ca}}} p(\hat{\boldsymbol{m}}_{\mathrm{ca},c_i} = \hat{\boldsymbol{m}}_{\mathrm{d},c_{\mathrm{e}}}), \quad (10)$$

$$P_{\mathrm{ccrc}}(D) = \begin{cases} 1, & \mu_{\mathrm{vote}}(D) > 0.5 \\ 0, & \mu_{\mathrm{vote}}(D) \leq 0.5, \end{cases} \quad (11)$$

where c_{e} is the emotion clause, C_{ca} is the cause clause set, $\hat{\boldsymbol{m}}_{\mathrm{ca},c_i}$ is the intermediate result of slot $[\mathtt{M}]_{\mathrm{ca}}$ for clause c_i, and $\hat{\boldsymbol{m}}_{\mathrm{d},c_{\mathrm{e}}}$ is the intermediate result of slot $[\mathtt{M}]_{\mathrm{d}}$ for emotion clause c_{e}.

4 Cross-task Training Method

Empirically, there are commonalities among different ECA tasks. These commonalities would further improve the the model performance. This paper proposes a cross-task training method to make the model better adapt knowledge from one domain to another. The following section will introduce this training method through an example. The initial pre-trained model is defined as $\mathcal{A}$. The first step, the model $\mathcal{A}$ is trained in the ECE task to obtain the trained model $\mathcal{B}$. Second step, the trained model $\mathcal{B}$ is used to train for the ECPE task to obtain the trained model $\mathcal{C}$. Finally, model $\mathcal{C}$ is used to measure the performance of the UECA-Prompt on ECPE. Any two tasks of ECA could perform the training steps mentioned above. The commonality among tasks could be learned by the model in the process of cross-task training.

5 Experiments

This paper conducts experiments on the ECE, ECPE, and CCRC tasks to evaluate our approach. We release our code at https://github.com/yajus/UECA-Prompt.

5.1 Datasets

The experiments are conducted on three public datasets. The **ECE dataset** (Gui et al., 2016a) is collected from SINA city news and contains 2105 instances. Its document has only one emotion word and one or more emotion causes. The **ECPE dataset** (Xia and Ding, 2019) is constructed based on the ECE dataset. It aggregates the instances containing the same text and different emotion cause labels. The **CCRC dataset** (Chen et al., 2020a) is also built based on the ECE dataset. It is constructed following two steps: manual annotation

Method	ECPE			Emotion Extraction			Cause Extraction			ECE			CCRC		
	F_1 (%)	P (%)	R (%)	F_1 (%)	P (%)	R (%)	F_1 (%)	P (%)	R (%)	F_1 (%)	P (%)	R (%)	F_1 (%)	P (%)	R (%)
UECA-Prompt	73.39	70.01	77.26	87.64	84.26	91.43	75.35	72.06	79.21	83.19	82.74	83.73	80.74	77.07	84.87
w/o f_{s}	73.24	69.31	77.96	87.38	83.93	91.24	74.95	71.89	78.80	82.82	82.89	82.81	80.39	75.29	86.38
w/o f_{e}	73.56	70.93	76.59	-	-	-	76.16	75.22	77.21	73.11	72.26	74.18	-	-	-
w/o f_{ca}	71.43	68.91	74.46	87.51	83.90	91.57	-	-	-	82.72	80.46	85.25	-	-	-
w/o f_{d}	-	-	-	87.21	83.75	91.09	75.44	73.07	78.13	83.23	82.74	83.95	-	-	-

Table 9: Ablation study on ECPE, ECE, and CCRC tasks.

and negative sampling. Each dataset is randomly split into ten folds for cross validation.

5.2 Implementation Details

The optimizer is AdamW (Loshchilov and Hutter, 2017). The batch size and learning rate are set to 8 and 1e-5, respectively. M is set to 3. The weight decay is set to 0.01 while other parameters of $\beta 1$, $\beta 2$ and ϵ are set to 0.9, 0.999 and 1e-8 by default. The dropout rate of the attention layer and hidden layer in BERT are both slightly modified to 0.2. The prompt method is implemented based on the BERT initialized with "BERT-Base, Chinese"[1] to achieve a fair comparison since the selected baselines are mostly based on BERT. All experiments are run on the machine containing a piece of RTX 3090 GPU.

During testing, to get a fair comparison on the original bias dataset, the indicate token in the word label set for the prediction of the directional constraint module is restricted to a smaller boundary, which achieves better performance. Specifically, for the i-th clause, the prediction result of the associated clause will be between $i - l$ and $i + l$. The experiment shows that our method achieves the best results when $l = 2$.

5.3 Baselines and Evaluation Metrics

The proposed UECA-Prompt is compared with several state-of-the-art methods for different ECA tasks.

Baselines for ECE. Baselines include statistical learning methods, **Multi-kernel** (Gui et al., 2016b); And deep learning methods, **MANN** (Li et al., 2019), **RHNN** (Fan et al., 2019), **RTHN** (Xia et al., 2019), **FSS-GCN** (Hu et al., 2021b) , and **EF-BHA** (Hu et al., 2021a).

Baselines for ECPE. Baselines include two-step methods, **Inter-EC** (Xia and Ding, 2019); And end-to-end methods, **UTOS** (Cheng et al., 2021), **PairGCN** (Chen et al., 2020c), **TransECPE** (Fan et al., 2020), **MTST** (Fan et al., 2021), **RANK-CP** (Wei et al., 2020), and **ECPE-MLL** (Ding et al., 2020b).

Baselines for CCRC. Baselines include **BiLSTM + Concatenation** (Chen et al., 2020a), **BiLSTM + BiLSTM** (Chen et al., 2020a), and **BiLSTM + Self-Attention** (Chen et al., 2020a).

Evaluation Metrics. Following the previous works (Xia and Ding, 2019; Fan et al., 2020), this paper adopts the precision (P), recall (R), and F1 score (F1) as the metrics for evaluation. The final results are obtained by averaging the ten-fold results.

5.4 Main Results

UECA-Prompt produces competitive results when compared with the other baselines on three tasks.

Results on ECPE. Table 2 reports the results of three task objectives of ECPE. The competitive performance of our method is mainly attributed to the significant improvement of the recall. This is because the sub-prompt modules in UECA-Prompt pay more attention to global information of the entire document rather than the local prediction for a single clause. However, this will lead to the loss of precision to a certain extent.

The comparison between UECA-Prompt and UECA-Prompt (m2m) shows that the performance of our method with the M2M module is not always superior. This is because multi-emotion samples only account for a small proportion of the dataset. Additional experiments on those multi-emotion samples are conducted. The results in Table 3 show that the method with M2M module obtains 2.3% improvements on ECPE. This indicates that the incorporation of the M2M module helps better handle the multi-emotion instances.

Results on ECE. The result in Table 4 demonstrates that UECA-Prompt obtains better results than RHNN (+5.26% in F_1). This shows the clear advantage of modeling the emotional causality through constructing prompt. Different from

[1]https://github.com/huggingface/transformers

Method	ECPE			Emotion Extraction			Cause Extraction			Method	ECE		
	F_1 (%)	P (%)	R (%)	F_1 (%)	P (%)	R (%)	F_1 (%)	P (%)	R (%)		F_1 (%)	P (%)	R (%)
UTOS-BERT	34.14	42.76	28.95	66.45	83.69	56.08	36.29	45.83	30.55	PAE$^{\diamondsuit}$	38.51	55.11	30.78
ECPE-MLL-bert	45.57	**61.53**	36.39	**80.33**	77.15	84.04	54.68	51.40	59.01	PAEDGL$^{\diamondsuit}$	40.96	55.25	32.79
MTST+Refinement	44.93	51.99	40.34	64.10	**77.65**	55.46	48.84	**56.25**	43.96	RTHN$^{\diamondsuit}$	54.45	54.67	54.66
UECA-Prompt	49.37	46.30	**53.22***	76.15	68.68	**85.67***	50.64	45.75	57.71	EF-BHA$^{\diamondsuit}$	60.61	56.40	65.49
UECA-Prompt (m2m)	**49.93***	48.60	52.11	75.66	70.66	81.68	**51.28***	44.76	**61.23***	UECA-Prompt	**65.62***	**63.67***	**68.12***

Table 10: Experimental results on the de-bias dataset for ECE and ECPE. $\diamondsuit$ indicates the results are reported in (Hu et al., 2021a). $*$ indicates statistically significant improvement ($p < 0.01$) over the best baseline.

Method	ECPE			Emotion Extraction			Cause Extraction			ECE			CCRC		
	F_1 (%)	P (%)	R (%)	F_1 (%)	P (%)	R (%)	F_1 (%)	P (%)	R (%)	F_1 (%)	P (%)	R (%)	F_1 (%)	P (%)	R (%)
UECA-Prompt	73.39	70.01	77.26	87.64	84.26	91.43	75.35	72.06	79.21	83.19	82.74	83.73	81.18	76.35	86.76
UECA-Prompt+ECE	74.86	72.59	77.44	88.23	84.94	91.90	76.80	75.06	78.78	-	-	-	81.18	76.35	86.76
UECA-Prompt+ECPE	-	-	-	-	-	-	-	-	-	83.46	84.32	82.70	80.98	76.08	86.78
UECA-Prompt+CCRC	74.70	71.82	77.99	88.16	84.75	91.95	77.55	76.24	79.16	84.40	84.57	84.27	-	-	-

Table 11: The experimental results of UECA-Prompt under cross-task training and non-cross-task training. UECA-Prompt+ECE, UECA-Prompt+ECPE and UECA-Prompt+CCRC are the results with cross-taks training.

ECPE, ECE is essentially single-task learning. The improvement of our method in precision on the ECE task verifies our conjecture that multi-task learning will lead to a decrease in precision.

Results on CCRC. Table 5 shows that UECA-Prompt significantly outperforms BiLSTM+BiLSTM (+11.42% in F_1). This indicates that UECA-Prompt can capture global context information, which is essential in emotion cause analysis.

5.5 Experimental Results Under Few-shot Setting Scenario

To further explore the potential of our method, few-shot setting experiments with 10% of training data are conducted. This section reports the experimental results under few-shot setting scenario. We compare the proposed UECA-Prompt with some state-of-the-art methods under the same experimental setting.

Results on ECPE. The results on ECPE task are shown in Table 6. UECA-Prompt obtains the best result toward the emotion cause pair extraction objective. It further proves that UECA-Prompt has the advantage in modeling the relation between emotion and cause. On the emotion extraction and cause extraction objectives, UECA-Prompt is slightly inferior to ECPE-MLL. This may be due to the data bias brought to ECPE-MLL.

Results on ECE. As shown in the last row of Table 7, UECA-Prompt achieves the best results on all evaluation metrics. Our method even approximates or exceeds the model trained under complete training sets (See Table 4). Compared with other methods, UECA-Prompt can better capture the association between emotion and cause with a few training samples.

Results on CCRC. UECA-Prompt also outperforms the state-of-the-art methods on the CCRC task. We observe that the simpler approach works better. For example, simply feature concatenation is more excellent than encoding features with BiLSTM or self-attention.

5.6 Ablation Study

UECA-Prompt is comprised of four sub-prompt components: emotion indicator function f_{e}, cause indicator function f_{ca}, directional constraint f_{s}, and sequential learning f_{d}. To verify the effect of different components, ablation experiments are carried out for different modules. The results are given in Table 9.

The performance of UECA-Prompt on the ECPE task integrally declines without any of the four components. This indicates that each of the four sub-prompt components plays distinct roles in feature learning, thereby proving the effectiveness of the four components. Specifically, the performance drops sharply without the cause indicator function. This observation indicates that the extraction of the cause clause is predominant in the ECPE task.

The influence of the emotion indicator function is significant on ECE tasks because the emotion clause is the most important information in ECE. Furthermore, UECA-Prompt can still achieve better performance than Multi-Kernel without emotional information on ECE tasks. This means UECA-Prompt is competent to obtain useful information from the context.

The result for the CCRC task shows that the

(a) ECPE (b) ECE (c) CCRC

Figure 4: Visualization of the token for different tasks. The Chinese token "是" means "is", while "非" means "isn't". The intensity of color is proportional to the weight value, which is the mean result of all heads and all layers in BERT.

performance of UECA-Prompt slightly degrades without the component of sequential token learning. This is mainly due to the change in precision. This indicates that eliminating the learning effect of the sequential learning module would impair the directional constraint module to extrapolate the correct results.

5.7 Analysis

Results on De-bias Data To verify the ability of UECA-Prompt to ease the bias caused by relative position, this paper conducts the experiments on de-bias datasets (Ding and Kejriwal, 2020). The results are shown in Table 10. The experiments are only conducted on ECPE and ECE because the CCRC task is position-irrelevant. Our method gains at least 5.01% improvement of F_1 on the ECE task and attains a 3.8% improvement of F_1 on the ECPE task. The results show that UECA-Prompt is more robust than baselines.

Commonalities Among Tasks To further explore the universality of UECA-Prompt, we train the model with the cross-task training method. The experimental results are shown in Table 11. It could be observed that the cross-task training method improves the performance of UECA-Prompt on three tasks. This indicates that UECA-Prompt can learn the commonalities among ECA tasks. It is noteworthy that the model firstly trained on the CCRC task (UECA-Prompt+CCRC) achieves the best performance. This result shows the importance of contextual information and demonstrates that UECA-Prompt could discriminate different contexts.

6 Case Study

To further understand the operation principle of UECA-Prompt, Fig. 4 visualizes the attention for different tokens in the constructed prompt text of a Chinese text. The subgraph (a), (b), and (c) in Fig. 4 represent the attention weights in ECPE, ECE, and CCRC, respectively. The query token is marked with the red box.

It can be observed that the attention weights are mainly concentrated in the answer slot [M] and the context in emotion and cause clauses. This indicates that different sub-prompt modules in UECA-Prompt could capture the key information of emotion and cause clauses as well as cooperate.

7 Conclusions

This paper proposes a universal prompt method for emotion cause analysis tasks. UECA-Prompt could uniformly model different ECA tasks by decomposing ECA tasks into multiple objectives and converting these objectives into sub-prompts. Meanwhile, the proposed directional constraint module and sequential learning module could effectively ease the bias caused by position information. Moreover, the cross-task training method further improves the performance of UECA-Prompt. The ability of UECA-Prompt to learn commonalities and contextual knowledge from different tasks is verified. The experimental results on three ECA tasks demonstrate the effectiveness of the proposed method.

This work chooses the general model, BERT, for a fair comparison. The sentiment-related PLM, such as SKEP (Tian et al., 2020), may further improve the performance of UECA-Prompt, which will be explored in our future work.

Acknowledgments

The work is supported by the National Key R&D Program of China (2018AAA0101203), the National Natural Science Foundation of China (62072483), and the Guangdong Basic and Applied Basic Research Foundation (2022A1515011690, 2021A1515012298).

References

Hongliang Bi and Pengyuan Liu. 2020. ECSP: A new task for emotion-cause span-pair extraction and classification. *arXiv preprint arXiv:2003.03507*.

Tom B. Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel M. Ziegler, Jeffrey Wu, Clemens Winter, Christopher Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amode. 2020. Language models are few-shot learners. *arXiv preprint arXiv:2005.14165*.

Rich Caruana. 1997. Multitask learning. *Machine learning*, 28(1):41–75.

Xinhong Chen, Qing Li, and Jianping Wang. 2020a. Conditional causal relationships between emotions and causes in texts. In *Proceedings of EMNLP*, pages 3111–3121.

Xinhong Chen, Qing Li, and Jianping Wang. 2020b. A unified sequence labeling model for emotion cause pair extraction. In *Proceedings of CCL*, pages 208–218.

Ying Chen, Wenjun Hou, Shoushan Li, Caicong Wu, and Xiaoqiang Zhang. 2020c. End-to-end emotion-cause pair extraction with graph convolutional network. In *Proceedings of CCL*, pages 198–207.

Xiyao Cheng, Ying Chen, Bixiao Cheng, Shoushan Li, and Guodong Zhou. 2017. An emotion cause corpus for chinese microblogs with multiple-user structures. *ACM Transactions on Asian and Low-Resource Language Information Processing (TALLIP)*, 17(1):1–19.

Zifeng Cheng, Zhiwei Jiang, Yafeng Yin, Na Li, and Qing Gu. 2021. A unified target-oriented sequence-to-sequence model for emotion-cause pair extraction. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 29:2779–2791.

Jiayuan Ding and Mayank Kejriwal. 2020. An experimental study of the effects of position bias on emotion causeextraction. *arXiv preprint arXiv:2007.15066*.

Zixiang Ding, Rui Xia, and Jianfei Yu. 2020a. ECPE-2D: Emotion-cause pair extraction based on joint two-dimensional representation, interaction and prediction. In *Proceedings of ACL*, pages 3161–3170.

Zixiang Ding, Rui Xia, and Jianfei Yu. 2020b. End-to-end emotion-cause pair extraction based on sliding window multi-label learning. In *Proceedings of EMNLP*, pages 3574–3583.

Chuang Fan, Hongyu Yan, Jiachen Du, Lin Gui, Lidong Bing, Min Yang, Ruifeng Xu, and Ruibin Mao. 2019. A knowledge regularized hierarchical approach for emotion cause analysis. In *Proceedings of EMNLP-IJCNLP*, pages 5614–5624.

Chuang Fan, Chaofa Yuan, Jiachen Du, Lin Gui, Min Yang, and Ruifeng Xu. 2020. Transition-based directed graph construction for emotion-cause pair extraction. In *Proceedings of ACL*, pages 3707–3717.

Chuang Fan, Chaofa Yuan, Lin Gui, Yue Zhang, and Ruifeng Xu. 2021. Multi-task sequence tagging for emotion-cause pair extraction via tag distribution refinement. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 29:2339–2350.

Tianyu Gao, Adam Fisch, and Danqi Chen. 2021. Making pre-trained language models better few-shot learners. In *ACL/IJCNLP (1)*.

Lin Gui, Dongyin Wu, Ruifeng Xu, Qin Lu, and Yu Zhou. 2016a. Event-driven emotion cause extraction with corpus construction. In *Proceedings of EMNLP*, pages 1639–1649.

Lin Gui, Ruifeng Xu, Qin Lu, Dongyin Wu, and Yu Zhou. 2016b. Emotion cause extraction, a challenging task with corpus construction. In *Chinese National Conference on Social Media Processing*, pages 98–109.

Xu Han, Zhengyan Zhang, Ning Ding, Yuxian Gu, Xiao Liu, Yuqi Huo, Jiezhong Qiu, Yuan Yao, Ao Zhang, Liang Zhang, et al. 2021. Pre-trained models: Past, present and future. *AI Open*, 2:225–250.

Neil Houlsby, Andrei Giurgiu, Stanislaw Jastrzebski, Bruna Morrone, Quentin De Laroussilhe, Andrea Gesmundo, Mona Attariyan, and Sylvain Gelly. 2019. Parameter-efficient transfer learning for nlp. In *International Conference on Machine Learning*, pages 2790–2799. PMLR.

Guimin Hu, Guangming Lu, and Yi Zhao. 2021a. Bidirectional hierarchical attention networks based on document-level context for emotion cause extraction. In *Findings of EMNLP*, pages 558–568.

Guimin Hu, Guangming Lu, and Yi Zhao. 2021b. Fss-gcn: A graph convolutional networks with fusion of semantic and structure for emotion cause analysis. *Knowledge-Based Systems*, 212:106584.

Chengxi Li, Feiyu Gao, Jiajun Bu, Lu Xu, Xiang Chen, Yu Gu, Zirui Shao, Qi Zheng, Ningyu Zhang, Yongpan Wang, et al. 2021a. Sentiprompt: Sentiment knowledge enhanced prompt-tuning for aspect-based sentiment analysis. *arXiv preprint arXiv:2109.08306*.

Xiang Lisa Li and Percy Liang. 2021. Prefix-tuning: Optimizing continuous prompts for generation. *arXiv preprint arXiv:2101.00190*.

Xiangju Li, Shi Feng, Daling Wang, and Yifei Zhang. 2019. Context-aware emotion cause analysis with multi-attention-based neural network. *Knowledge-Based Systems*, 174:205–218.

Xiangju Li, Wei Gao, Shi Feng, Daling Wang, and Shafiq Joty. 2021b. Span-level emotion cause analysis by bert-based graph attention network. In *Proceedings of CIKM*, pages 3221–3226.

Pengfei Liu, Weizhe Yuan, Jinlan Fu, Zhengbao Jiang, Hiroaki Hayashi, and Graham Neubig. 2021. Pre-train, prompt, and predict: A systematic survey of prompting methods in natural language processing. *arXiv preprint arXiv:2107.13586*.

Ilya Loshchilov and Frank Hutter. 2017. Decoupled weight decay regularization. *arXiv preprint arXiv:1711.05101*.

Ruotian Ma, Xin Zhou, Tao Gui, Yiding Tan, Qi Zhang, and Xuanjing Huang. 2021. Template-free prompt tuning for few-shot ner. *arXiv preprint arXiv:2109.13532*.

Fabio Petroni, Patrick Lewis, Aleksandra Piktus, Tim Rocktäschel, Yuxiang Wu, Alexander H Miller, and Sebastian Riedel. 2020. How context affects language models' factual predictions. *arXiv preprint arXiv:2005.04611*.

Victor Sanh, Albert Webson, Colin Raffel, Stephen H Bach, Lintang Sutawika, Zaid Alyafeai, Antoine Chaffin, Arnaud Stiegler, Teven Le Scao, Arun Raja, et al. 2021. Multitask prompted training enables zero-shot task generalization. *arXiv preprint arXiv:2110.08207*.

Timo Schick, Helmut Schmid, and Hinrich Schütze. 2020. Automatically identifying words that can serve as labels for few-shot text classification. In *Proceedings of CCL*, pages 5569–5578.

Timo Schick and Hinrich Schütze. 2020a. Exploiting cloze questions for few shot text classification and natural language inference. *arXiv preprint arXiv:2001.07676*.

Timo Schick and Hinrich Schütze. 2020b. It's not just size that matters: Small language models are also few-shot learners. *arXiv preprint arXiv:2009.07118*.

Hao Tang, Donghong Ji, and Qiji Zhou. 2020. Joint multi-level attentional model for emotion detection and emotion-cause pair extraction. *Neurocomputing*, 409:329–340.

Hao Tian, Can Gao, Xinyan Xiao, Hao Liu, Bolei He, Hua Wu, Haifeng Wang, and Feng Wu. 2020. SKEP: Sentiment knowledge enhanced pre-training for sentiment analysis. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 4067–4076, Online. Association for Computational Linguistics.

Jason Wei, Maarten Bosma, Vincent Y Zhao, Kelvin Guu, Adams Wei Yu, Brian Lester, Nan Du, Andrew M Dai, and Quoc V Le. 2021. Finetuned language models are zero-shot learners. *arXiv preprint arXiv:2109.01652*.

Penghui Wei, Jiahao Zhao, and Wenji Mao. 2020. Effective inter-clause modeling for end-to-end emotion-cause pair extraction. In *Proceedings of ACL*, pages 3171–3181.

Sixing Wu, Fang Chen, Fangzhao Wu, Yongfeng Huang, and Xing Li. 2020. A multi-task learning neural network for emotion-cause pair extraction. In *ECAI 2020*, pages 2212–2219. IOS Press.

Rui Xia and Zixiang Ding. 2019. Emotion-cause pair extraction: A new task to emotion analysis in texts. In *Proceedings of ACL*, pages 1003–1012.

Rui Xia, Mengran Zhang, and Zixiang Ding. 2019. Rthn: A rnn-transformer hierarchical network for emotion cause extraction. *arXiv preprint arXiv:1906.01236*.

Bo Xu, Hongfei Lin, Yuan Lin, and Kan Xu. 2021. Two-stage supervised ranking for emotion cause extraction. *Knowledge-Based Systems*, 228:107225.

Ruifeng Xu, Jiannan Hu, Qin Lu, Dongyin Wu, and Lin Gui. 2017. An ensemble approach for emotion cause detection with event extraction and multi-kernel svms. *Tsinghua Science and Technology*, 22(6):646–659.

Ningyu Zhang, Luoqiu Li, Xiang Chen, Shumin Deng, Zhen Bi, Chuanqi Tan, Fei Huang, and Huajun Chen. 2021. Differentiable prompt makes pre-trained language models better few-shot learners. *arXiv preprint arXiv:2108.13161*.

One-Teacher and Multiple-Student Knowledge Distillation on Sentiment Classification

Xiaoqin Chang[1] **Sophia Yat Mei Lee**[2] **Suyang Zhu**[1*]
Shoushan Li[1] **Guodong Zhou**[1]
[1]Natural Language Processing Lab, Soochow University, China
[2]Department of Chinese and Bilingual Studies, The Hong Kong Polytechnic University
xqchang@stu.suda.edu.cn
{syzhu, samlee, gdzhou}@suda.edu.cn
ym.lee@polyu.edu.hk

Abstract

Knowledge distillation is an effective method to transfer knowledge from a large pre-trained teacher model to a compacted student model. However, in previous studies, the distilled student models are still large and remain impractical in highly speed-sensitive systems (e.g., an IR system). In this study, we aim to distill a deep pre-trained model into an extremely compacted shallow model like CNN. Specifically, we propose a novel one-teacher and multiple-student knowledge distillation approach to distill a deep pre-trained teacher model into multiple shallow student models with ensemble learning[1]. Moreover, we leverage large-scale unlabeled data to improve the performance of students. Empirical studies on three sentiment classification tasks demonstrate that our approach achieves better results with much fewer parameters (0.9%-18%) and extremely high speedup ratios (100X-1000X).

1 Introduction

Sentiment classification is a task of classifying a text into sentimental orientation categories, such as *positive* and *negative*, and this task plays an important role in natural language processing (NLP) and benefits many real applications (Clavel and Callejas, 2016; Shen et al., 2018; Wang et al., 2019).

The past few years have witnessed the prevailing of deep pre-trained models on sentiment classification (Yu and Jiang, 2019; Ke et al., 2021; Chen et al., 2021). However, despite their significant improvements over non-pre-trained models like CNN and LSTM, their need for a large number of computing resources and relatively long inference time becomes a major bottleneck for real-world applications.

To solve this problem, knowledge distillation, which transfers knowledge from a large model (the teacher) to a smaller model (the student), is gaining popularity for reducing the computing and time costs. However, existing distilled models' parameters are still too large for some low-end devices and speed-sensitive applications. For instance, sentiment classification is usually an essential component of an information retrieval (IR) system (Paltoglou and Thelwall, 2010; Kauer and Moreira, 2016), in which users are highly speed-sensitive to the responding speed. Thus, improving the inference speed of the pre-trained model on sentiment classification becomes a critical element of applying the pre-trained sentiment classification model to IR applications. Motivated by the above, in this paper, we aim to propose a novel distillation technique that can distill a huge-parameterized pre-trained model like BERT (Devlin et al., 2019) into a minimal-parameterized non-pre-trained model like CNN or LSTM.

In principle, compared with traditional shallow models, deep pre-trained models have two major advantages. **First**, most pre-trained models are architectured in the manner of ensemble learning. In the literature, ensemble learning has been proven to be effective in performance boosting. For instance, BERT combines multiple Transformer layers under an ensemble architecture. Meanwhile, each Transformer layer contains the attention ensemble by using multiple self-attention heads (Vaswani et al., 2017).

Second, a pre-trained model highly benefits from the knowledge contained in unlabeled data. For instance, the BERT-base model is pre-trained using unlabeled data containing 3.3 billion words from Wikipedia and BooksCorpus. At least, the large scale of unlabeled data enables the pre-trained model to handle unknown words in a downstream task more easily. For instance, given a sentence *"Everything is awesome!"*, suppose that

*Corresponding author
[1]Our code is available at https://github.com/strive-hhh/OTMS-KD

Proceedings of the 29th International Conference on Computational Linguistics, pages 7042–7052
October 12–17, 2022.

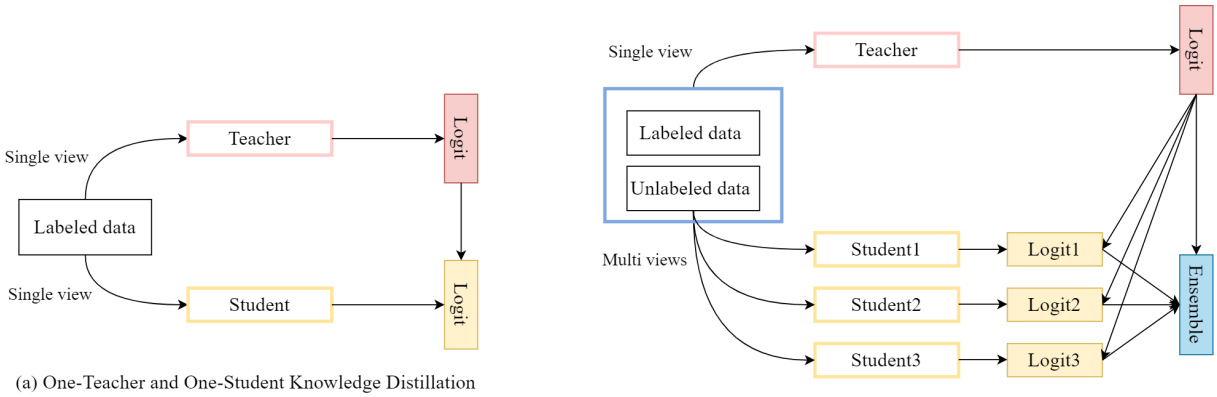

(a) One-Teacher and One-Student Knowledge Distillation in Most Previous Studies

(b) One-Teacher and Multiple-Student Knowledge Distillation in Our Study

Figure 1: (a): The framework of most existing approaches; (b): The framework of our approach.

the word *"awesome"* is not observed in the training data in a sentiment classification task. Thus, non-pre-trained shallow models like CNN-based or LSTM-based classifiers cannot easily determine the sentiment orientation of *"awesome"*. On the contrary, however, pre-trained models like BERT can infer the meaning of *"awesome"*, which is close to the observed word *"excellent"*.

In this paper, inspired by the above, we propose a novel ensemble knowledge distillation approach by leveraging both ensemble learning and unlabeled data. Specifically, first, we use multiple shallow models, together with their ensemble model, as student models during distillation in order to take advantage of ensemble learning. Thanks to many previous studies on multi-view learning on sentiment classification, multiple student models could be easily obtained by using various kinds of multiple views, such as multiple types of embeddings (Ren et al., 2016) and multiple languages (Fei and Li, 2020). Second, we leverage large-scale unlabeled sentiment classification corpora during distillation. Different from most previous studies on one-teacher and one-student knowledge distillation, we propose a one-teacher and multiple-student ensemble distillation framework, which is illustrated in Figure 1.

Empirical studies on three sentiment classification tasks demonstrate that our approach outperforms the pre-trained teacher models with much fewer parameters (0.9%-18%) and extremely high speedup ratios (100X-1000X).

The remainder of this paper is organized as follows. Section 2 provides an overview of the related studies on sentiment analysis and knowledge distillation. Section 3 explains the details of our approach. Section 4 introduces the experimental settings and results. Section 5 states the conclusion and the future work.

2 Related Works

2.1 Sentiment Classification

In the last decade, the studies of sentiment classification have been dominated by neural network approaches. This line of research begins with developing sentiment classification models with shallow models, such as CNNs (Rakhlin, 2016; Johnson and Zhang, 2015), RNNs (Castellucci et al., 2014; Tang et al., 2015), and LSTM (Tai et al., 2015). Thereafter, some studies incorporate other methods into shallow models, such as attention methods (Yang et al., 2017; Liu and Zhang, 2017; Zeng et al., 2019) and graph neural networks, e.g., GCN (Marcheggiani and Titov, 2017; Vashishth et al., 2019).

Recently, deep pre-trained neural network models are becoming popular due to their highly promising performances. Large-scale pre-trained language models, such as ELMo (Peters et al., 2018), BERT (Devlin et al., 2019; Gao et al., 2019), and RoBERTa (Liu et al., 2019), have been shown to be rather effective for sentiment classification tasks with the learning paradigm of fine-tuning. More recently, pre-training models with the learning paradigm of prompt have become popular in some zero-shot or few-shot natural language processing tasks, among which sentiment classification is a classic and important task (Schick et al., 2020; Schick and Schütze, 2020; Gao et al., 2020).

2.2 Knowledge Distillation

In the field of natural language processing, the majority of the previous work on knowledge distillation has attempted to reduce the depth of BERT. For instance, Tang et al. (2019) propose compressing BERT models into a small LSTM model. Sun et al. (2019) introduce the Patient Knowledge Distillation approach to compress a large model into an equally effective lightweight shallow network. Sanh et al. (2019) develop a general-purpose pre-trained version of BERT called DistilBERT. Jiao et al. (2019) also propose a compact model called TinyBERT based on a new two-stage learning framework that captures both the general domain and task-specific knowledge in BERT. Zhou et al. (2021) suggest training a light named entity recognition using novel multi-grained knowledge distillation techniques. Instead of reducing the depth of BERT, Sun et al. (2020) attempt to reduce its width and develop a deep and thin model called MobileBERT. Unlike the existing fixed-size BERT compression models, Hou et al. (2020) introduce the DynaBERT model which can adjust the size and latency by selecting the sub-networks with different depth and width.

More recently, Reich et al. (2020) and Wu et al. (2021) propose multiple-teacher and one-student knowledge distillation frameworks for pre-trained language model compression. Besides, in the research field of computer vision, teacher-free ensemble distillation approaches, i.e., zero-teacher and multiple-student approaches, have been proposed in (Chen et al., 2020; Guo et al., 2020; Walawalkar et al., 2020; Li and Wang, 2019).

Different from the above studies, this paper introduces a novel ensemble knowledge distillation approach with the paradigm of one-teacher and multiple-student, harnessing the power of multiple shallow student models during distillation. To the best of our knowledge, this is the first work to research the ensemble knowledge distillation paradigm of the one-teacher and multiple-student model.

3 Methodology

In this section, we introduce the details of our approach.

3.1 Problem Description

Let D be a dataset and it contains both labeled data and unlabeled data where $D_l = \{x, y\}$ is labeled data and $D_u = \{x_u\}$ is unlabeled data. Our one-teacher and multiple-student knowledge distillation approach aims to distill the knowledge from a pre-trained teacher model $f(x; \theta^t)$ into an ensembled student model $g(x; \theta^s)$, where θ^t and θ^s are the model parameters of the teacher and the student respectively. In contrast to most existing studies, the parameters of the ensemble student model are much fewer than those of the teacher model, (i.e., $|\theta^s| << |\theta^t|$).

In this study, we apply our approach to three different types of sentiment classification tasks, i.e., supervised sentiment classification, zero-shot sentiment classification, and cross-lingual sentiment classification.

3.2 One Teacher Model

The teacher model is trained in different manners according to different types of sentiment classification tasks.

Supervised or Cross-lingual Sentiment Classification: In the supervised or cross-lingual sentiment classification task, we train the teacher model in a supervised manner. Let $\{x_i, y_i\}_{i=1}^N$ be a training set which contains N labeled training samples. x_i and y_i denote the ith input sample of the teacher and its gold label, respectively.

Following the work of Sun et al. (2019), the teacher model first computes the embedding $\mathbf{h}_i^t = f(x_i; \theta^t)$ of x_i where f represents the function of the teacher network. Then the teacher model feeds $\mathbf{h}_i^t$ into a linear layer and a softmax activation function to obtain the predicted sentiment label of x_i, i.e.,

$$\hat{y}_i = P^t(y_i|x_i) = softmax(\mathbf{W}^t \mathbf{h}_i^t) \quad (1)$$

where the superscript t means "teacher" model, $\mathbf{W}^t$ denotes the weight matrix to be learned in the linear layer. The tuned parameters of the teacher model can be represented as follows:

$$\hat{\theta}^t = arg \min_{\theta^t} \sum_{i=1}^{N} L_{CE}^t(x_i, y_i; [\theta^t, \mathbf{W}^t]) \quad (2)$$

where L_{CE}^t denotes the cross-entropy loss function applied in teacher's training. Then, the teacher model predicts samples in unlabeled data with soft labels, i.e.,

$$\begin{aligned} \hat{y}_u &= P^t(y_u|x_u) = softmax(\frac{\mathbf{W}^t \mathbf{h}_u^t}{T}) \\ &= softmax(\frac{\mathbf{W}^t f(x_u; \hat{\theta}^t)}{T}) \end{aligned} \quad (3)$$

where $P^t(\cdot|\cdot)$ denotes the prediction probability of the teacher. $\hat{\theta}^t$ denotes the updated parameters of the teacher. T denotes the temperature during distillation.

Zero-shot Sentiment Classification: In the zero-shot sentiment classification task, following the work of Gao et al. (2020), the teacher model is a prompt-based zero-shot learner. Let ${x_u}_{u=1}^{N'}$ be N' unlabeled data. Given an unlabeled sample x_u, a rewritten input through a manual prompt template is generated as follows:

$$x_u^{prompt} = [CLS]\ x_u\ It\ was\ [MASK].\ [SEP]$$

Let $\mathcal{M} : \mathcal{Y} \rightarrow \mathcal{V}$ be a mapping from the task label space to sentiment words. In our sentiment classification task, the sentimental labels $\{0, 1\}$ are mapped into two opinion words, i.e., {*terrible* (negative), *great* (positive)}. Then x_u^{prompt} is predicted by the teacher model through predicting the probability of filling $[MASK]$ with *terrible* (or *great*), which can be considered as the probability of predicting label. Specifically, the teacher model outputs the hidden representation of $[MASK]$:

$$\mathbf{h}_{[MASK]} = f(x_u^{prompt}; \theta^t) \tag{4}$$

where $\mathbf{h}_{[MASK]}$ denotes the hidden representation of $[MASK]$. Then the probability of predicting label (i.e., the soft label) is fetched through a linear layer and a softmax activation function, i.e.,

$$\begin{aligned} P^t(y_u|x_u) &= P^t([MASK] = \mathcal{M}(y)|x_u^{prompt}) \\ &= softmax(\frac{\mathbf{W}_{\mathcal{M}(y)} \cdot \mathbf{h}_{[MASK]}}{T}) \end{aligned} \tag{5}$$

where $\mathcal{M}(y) \in \{terrible, great\}$ denotes a certain sentiment word, and $\mathbf{W}_{\mathcal{M}(y)}$ denotes the pre-trained weight of the sentiment word.

3.3 Multiple Student Model

The ensembled student model consists of k student models. Let $\mathbf{A}_u$ be the sequential input (i.e., a matrix of word embeddings of a sentence) of the students model. $g_j(\mathbf{A}_u; \theta_j^s)$ represents the function of the jth student network, where $j \in [1, k]$ and θ_j^s denotes its parameters. Each student model first computes the vectorized representation $\mathbf{h}_{uj}^s = g_j(\mathbf{A}_u; \theta_j^s)$ of x_u. $\mathbf{h}_{uj}^s$ is then fed into a linear layer to obtain the prediction probability of the jth student model, i.e.,

$$\hat{y}_{uj} = P_j^s(y_u|\mathbf{A}_u) = \mathbf{W}_j^s \mathbf{h}_{uj}^s \tag{6}$$

where $\mathbf{W}_j^s$ is the weight matrix of the jth student model to be learned, and $P_j^s(\cdot|\cdot)$ denotes the prediction probability of the jth student model. The final ensembled probability is the weighted sum of all students' outputs, i.e.,

$$P_{ensemble}^s(y_u|\mathbf{A}_u) = softmax(\frac{\sum_{j=1}^k \alpha_j^s * \hat{y}_{uj}}{T}) \tag{7}$$

where $\alpha_j^s \in [0, 1]$ denotes the weight of the prediction probability of the jth student model and subjects to $\sum_{j=1}^k \alpha_j^s = 1$. In supervised or cross-lingual sentiment classification, the weights are learnable during model training with labeled data. But in zero-shot sentiment classification, since no labeled data is available, the weights are simply set to be the same (i.e., $\alpha_1^s = \alpha_2^s = ... = \alpha_k^s$).

3.4 Model Training

The objective loss function of a one-teacher and one-student distillation model is defined as follows:

$$L_{KD}(P^t, P^s) = \sum_{i=1}^{n} T^2 D_{KL}(P_i^t || P_i^s) \tag{8}$$

where n denotes the batch size, P_i^t and P_i^s denote the prediction probabilities of the ith sample outputted by the teacher and the student, respectively. D_{KL} is the KL divergence.

Different from the above, our approach applies an ensembled knowledge distillation loss function to train the multiple student models. Specifically, the ensembled *KD* loss is computed according to the predicted probabilities of student models as well as the final ensembled probability, i.e.,

$$\begin{aligned} Loss = &(\sum_{j=1}^{k} \lambda_j L_{KD}(P^t, P_j^s)) \\ &+ \lambda_e L_{KD}(P^t, P_{ensemble}^s) \end{aligned} \tag{9}$$

where λ_j denotes the weight of *KD* loss of the jth student and λ_e denotes the weight of *KD* loss of the ensembled students.

The learned ensembled student model is finally applied for evaluating the test set.

4 Experiments

In this section, we systematically evaluate our one-teacher and multiple-student knowledge distillation approach in three types of sentiment classification tasks, i.e., supervised sentiment classification, zero-shot sentiment classification, and cross-lingual sentiment classification.

4.1 Supervised Sentiment Classification

Dataset: The data of YELP (sentence-level) (Li et al., 2018), a widely used dataset for supervised sentiment classification, is used. Specifically, 3,000, 1,000, and 1,000 balanced samples are selected as training, development, and test data. An additional 100,000 samples are selected as unlabeled data which will be leveraged in the distillation process.

Evaluation Metrics: Standard *Accuracy* and *Macro-F1* are used to evaluate the performance of sentiment classification. Besides, the parameters and the inference time of per sample on CPU are applied to evaluate the operational performance of the distilled models.

Learning Models and Parameter Settings: All hyper-parameters are tuned according to the development set. The temperature T is set to 1.0. The batch size is set to 128. The teacher model is optimized by the AdamW (Loshchilov and Hutter, 2017) optimizer, where the initial learning rate is 2e-5 and weight decay is 1e-3. The student models are optimized by the Adam (Kingma and Ba, 2014) optimizer, where the initial learning rate is 1e-3 and weight decay is 1e-4 or 1e-5. In this task, the teacher model is the pre-trained 12-layer BERT-base, which has been the most frequently-researched teacher model in previous studies in the supervised learning setting. The student model is CNN with 100 kernels of 3 different sizes, in which the embedding size is 50 and the kernel size is 3x50, 4x50 and 5x50 respectively.

Multi-view Settings: Different types of word embeddings are used as multiple views to generate different student models. Specifically, we employ three different types of Glove embeddings (Pennington et al., 2014), i.e., Glove.6B.50d, Glove.twitter.27B.50d, and Glove.42B.300d.

Baselines: For comparison, we implement the following knowledge distillation approaches.

(1) DistilBERT (Sanh et al., 2019): This approach obtains a student model by transferring knowledge from the last layer of a pre-trained BERT in both the pre-training stage and optional fine-tuning stage. This is a one-teacher and one-student distillation approach and no unlabeled data is used.

(2) TinyBERT (Jiao et al., 2019): This approach obtains a student model by transferring knowledge from BERT with a novel transformer distillation method. This is a one-teacher and one-student distillation approach and no unlabeled data is used.

(3) MobileBERT (Sun et al., 2020): This approach obtains a student model by transferring knowledge from BERT-Large in the pre-training stage. This is a one-teacher and one-student distillation approach and no unlabeled data is used.

(4) XtremeDistil (Mukherjee and Awadallah, 2020): This approach obtains a student model by transferring knowledge from a multilingual pre-trained model, by leveraging teacher representations agnostic of its architecture and stage-wise optimization schedule. Moreover, this approach employs unlabeled data to boost performance. This is a one-teacher and one-student distillation approach and unlabeled data is used.

(5) MT-BERT (Wu et al., 2021): This approach obtains a student model with TinyBERT by transferring knowledge from multiple teachers, i.e., BERT, Roberta and UniLM. This is a multiple-teacher and one-student distillation approach and no unlabeled data is used.

(6) Distilled BiLSTM (Tang et al., 2019): This approach obtains a student model with a shallow neural network BiLSTM by transferring knowledge from BERT. This is a one-teacher and one-student distillation approach and no unlabeled data is used.

(7) Distilled Single CNN and Distilled Single CNN with more parameters: This approach distills a pre-trained BERT into a CNN with unlabeled data. The model with more parameters is obtained by leveraging 275 kernels and the embedding size of 300.

For reference, apart from distillation models, we also provide the results from models including CNN, BiLSTM, and Ensembled CNNs, which are trained with the labeled data only with no knowledge distillation.

Results: As shown in Table 1, compared with a single CNN, Ensembled CNNs improves very little (0.2%) when only training data are available. However, Distilled Ensembled CNNs with unlabeled data (our approach) achieves a 3.7% improvement in both *Accuracy* and *Macro-F1*. Moreover, our approach performs better than Distilled Single CNN with more parameters, which indicates that performance gain is more the result of ensemble distillation than only distilling a larger model. Even better, our approach achieves a slightly higher classification performance compared with the teacher model.

Methods	#Params	*Accuracy*	*Macro-F1*
BERT-base (Teacher) (Devlin et al., 2019)	109.48M	0.958	0.958
DistilBERT6 (Sanh et al., 2019)	65.78M	0.952	0.952
TinyBERT4 (Jiao et al., 2019)	14.35M	0.938	0.938
TinyBERT6 (Jiao et al., 2019)	66.96M	0.956	0.956
MobileBERT (Sun et al., 2020)	24.58M	0.947	0.947
XtremeDistil (Mukherjee and Awadallah, 2020) †	12.75M	0.960	0.960
MT-BERT4 (Wu et al., 2021)	14.35M	0.945	0.945
BiLSTM (Wang et al., 2018)	2.35M	0.919	0.919
CNN (Kim, 2014)	**0.15M**	0.927	0.927
Ensembled CNNs	1.20M	0.929	0.929
Distilled BiLSTM (Tang et al., 2019)	2.35M	0.920	0.920
Distilled Single CNN †	0.47M	0.958	0.958
Distilled Single CNN with more parameters †	3.74M	0.960	0.960
Distilled Ensembled CNNs (Our approach) †	3.74M	**0.964**	**0.964**

Table 1: Performances in supervised sentiment classification. "†" denotes that this model leverages unlabeled data during distillation.

Methods	#Params	Inf. time on CPU
Teacher	109.48M	10.61ms
Our approach	**3.74M**	**0.03ms**

Table 2: Operational performance of the teacher model and our approach in supervised sentiment classification.

Operational performance: The parameters and inference times of the teacher model and our approach are given in Table 2. The proposed model has a significantly smaller size (96.6% fewer parameters) and a notably faster inference speed (353 times faster) compared with the teacher model.

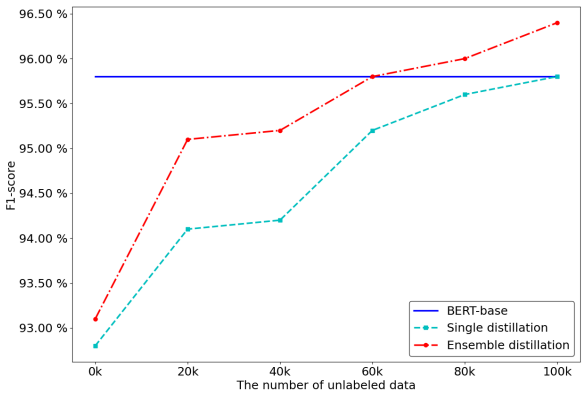

Figure 2: The influence of the scale of leveraged unlabeled data in supervised sentiment classification.

Influence of ensemble learning and leveraging unlabeled data: Figure 2 shows the influence of leveraging different scales of unlabeled data and applying ensemble learning. Both Distilled Single CNN and Distilled Ensembled CNNs perform much worse than the teacher model when no unlabeled data is available. However, Distilled Single CNN is able to achieve a highly similar performance compared with the teacher when 100k unlabeled data are leveraged. Furthermore, Distilled Ensembled CNNs surpasses the teacher when the scale of unlabeled data is over 80k.

4.2 Zero-shot Sentiment Classification

Dataset: The data of YELP (sentence-level) (Li et al., 2018) is used. Specifically, 1,000 balanced samples are selected as test data. An additional 100,000 samples are selected as unlabeled data, which will be leveraged in the distillation process. It is worthwhile to note that no training and development data is used in zero-shot sentiment classification.

Learning Models and Parameter Settings: The teacher model is the pre-trained 24-layer RoBERTa-large, which has been shown as an excellent model for zero-shot learning (Gao et al., 2020). Other parameter settings and multi-view settings are the same as supervised sentiment classification.

Baselines: Since few previous studies have conducted their research on knowledge distillation on zero-shot learning, we only implement the baseline approach of Distilled Single CNN with unlabeled data in this experiment.

Results: As shown in Table 3, Distilled Single

Methods	#Params	*Accuracy*	*Macro-F1*
RoBERTa-large (Teacher) (Liu et al., 2019)	408.98M	0.847	0.844
Distilled Single CNN †	**0.46M**	0.861	0.859
Distilled Single CNN with more parameters †	3.67M	0.872	0.874
Distilled Ensembled CNNs (Our approach) †	3.67M	**0.881**	**0.879**

Table 3: Performances in zero-shot sentiment classification. "†" denotes that this model leverages unlabeled data during distillation.

Methods	#Params	Inf. time on CPU
Teacher	408.98M	45.25ms
Our approach	**3.67M**	**0.03ms**

Table 4: Operational performance of the teacher model and our approach in zero-shot sentiment classification.

CNN outperforms the teacher model in both *Accuracy* and *Macro-F1* when 40k unlabeled samples are leveraged. Moreover, our approach outperforms the Distilled Single CNN with a 2.0% improvement in both *Accuracy* and *Macro-F1* and performs better than Distilled Single CNN with more parameters.

Operational performance: The parameters and inference times of the teacher model and our approach are given in Table 4. The proposed model has a significantly smaller size (99.1% fewer parameters) and a notably faster inference speed (1507 times faster) compared with the teacher model.

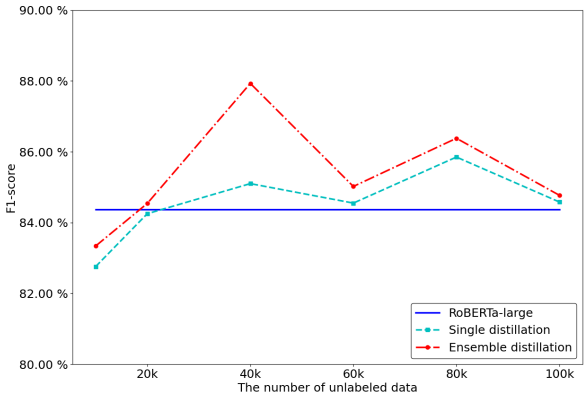

Figure 3: The influence of the scale of leveraged unlabeled data in zero-shot sentiment classification.

Influence of ensemble learning and leveraging unlabeled data: Figure 3 shows the influence of leveraging different scales of unlabeled data and applying ensemble learning. The zero-shot sentiment classification performance grows with the scale of leveraged unlabeled data when the size is less than 40k. However, the performance of our approach declines when the size of unlabeled data increases to over 60k and 100k. This might be due to the absence of a training set and validating set. Fortunately, the weaker performances of our approach are still better than those of the teacher model.

4.3 Cross-lingual Sentiment Classification

Datasets: In this experiment, Chinese is considered as the source language where labeled data is available and English is considered as the target language where only unlabeled data (with no labeled data) is available. 4,000 and 2,000 labeled Chinese samples in the document-level data of Hotel Review (Jie et al., 2016) is selected as the training and development data. 2,000 and 80,000 samples in the data of YELP (document-level) (Zhang et al., 2015) is selected as test and unlabeled data. All English (or Chinese) samples are translated into Chinese (or English) samples via Baidu Translator API.

Teacher Models: Several teacher models are designed to perform cross-lingual sentiment classification. As shown in Table 5, the first part is to employ BERT-Chinese-base and XLM-R-base(Chn) to train the teacher model with Chinese labeled data. Then, all English samples, i.e., unlabeled and test samples, are translated into Chinese for distillation and testing. The second part is to translate the Chinese labeled samples into English and then train the teacher model with BERT-English-base and XLM-R-base(Eng). Note that XLM-R (Conneau et al., 2020) is a state-of-the-art multilingual pre-training model. From Table 5, we can see that BERT-Chinese-base performs best and thus we choose it as the teacher model in the distillation experiment.

Multi-view Settings: English text, Chinese text, and together with their mixed text, are considered as three different views to generate three student models.

Baselines: Since few previous studies have con-

Methods	#Params	*Accuracy*	*Macro-F1*
BERT-Chinese-base (Devlin et al., 2019)	102.27M	**0.876**	**0.876**
XLM-R-base(Chn) (Conneau et al., 2020)	278.05M	0.855	0.854
BERT-English-base (Devlin et al., 2019)	109.48M	0.840	0.838
XLM-R-base(Eng) (Conneau et al., 2020)	278.05M	0.873	0.872

Table 5: Performances of different teacher models in cross-lingual sentiment classification.

Methods	#Params	*Accuracy*	*Macro-F1*
BERT-Chinese-base (Teacher) (Devlin et al., 2019)	102.27M	0.876	0.876
CNN (Chn) (Kim, 2014)	0.97M	0.691	0.691
CNN (Eng) (Kim, 2014)	**0.46M**	0.692	0.690
Ensembled CNNs	5.05M	0.714	0.714
Distilled Single CNN (Chn) †	1.31M	0.869	0.869
Distilled Single CNN (Eng) †	2.51M	0.868	0.868
Distilled Single CNN (Chn and Eng) †	15.36M	0.871	0.871
Distilled Ensembled CNNs (Our approach) †	18.88M	**0.878**	**0.878**

Table 6: Performances in cross-lingual sentiment classification. "†" denotes that this model leverages unlabeled data during distillation.

Methods	#Params	Inf. time on CPU
Teacher	102.27M	268.52ms
Our approach	**18.88M**	**1.22ms**

Table 7: Operational performance of the teacher model and our approach in cross-lingual sentiment classification.

ducted their research on knowledge distillation on cross-lingual learning, we only implement the baseline approach of CNN, Ensemble CNNs and Distilled Single CNN with unlabeled Data in this experiment.

Results: Table 6 shows the results of baselines and our approach in cross-lingual sentiment classification. CNN in both Chinese view and English view perform much more poorly than the teacher model, resulting in a significantly lower performance by Ensembled CNNs. However, by leveraging the unlabeled data, Distilled Single CNN in three views improve notably with an over 18% improvement in *Accuracy* and *Macro-F1* compared with a single CNN. Furthermore, our approach achieves higher *Accuracy* and *Macro-F1* than the teacher model.

Operational performance: The parameters and inference times of the teacher model and our approach are given in Table 7. The proposed model has a significantly smaller size (81.5% fewer parameters) and a notably faster inference speed (219 times faster) compared with the teacher model.

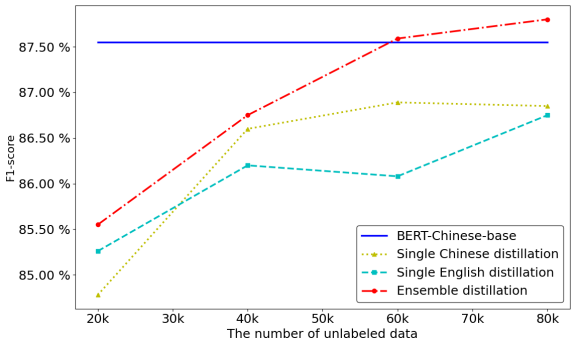

Figure 4: The influence of the scale of leveraged unlabeled data in cross-lingual sentiment classification.

Influence of ensemble learning and leveraging unlabeled data: Figure 4 shows the influence of leveraging different scales of unlabeled data and applying ensemble learning. From this figure, we can see that, in cross-lingual sentiment classification, our approach benefits greatly from unlabeled data. Moreover, distilling knowledge into ensembled CNNs results in consistently better performance than distilling knowledge into a single CNN.

5 Conclusion

In this study, we propose a novel approach of knowledge distillation, namely one-teacher and multiple-student knowledge distillation, in sentiment classification. Our approach is capable of compacting a large model into a minimal ensem-

ble model with both ensemble learning and unlabeled data. Empirical studies on three sentiment classification tasks demonstrate that the distilled model performs even better than the teacher model with much fewer parameters and a much better operational performance on CPU.

In our future work, we aim to improve our approach by carefully selecting a suitable number of unlabeled samples instead of using all of them. In addition, we would like to apply our approach to other NLP tasks besides sentiment classification.

Acknowledgements

We thank the anonymous reviewers for their insightful comments and suggestions. This research work was supported by a Major Project of Ministry of Science and Technology of China No.2020AAA0108604 and two NSFC grants, i.e., No.62076176 and No.62106166. This research work was also supported by a General Research Fund (GRF) project sponsored by the Research Grants Council, Hong Kong (Project No.15611021).

References

Giuseppe Castellucci, Simone Filice, Danilo Croce, and Roberto Basili. 2014. Unitor: Aspect based sentiment analysis with structured learning. In *Proceedings of the 8th international workshop on semantic evaluation (SemEval 2014)*, pages 761–767.

Defang Chen, Jian-Ping Mei, Can Wang, Yan Feng, and Chun Chen. 2020. Online knowledge distillation with diverse peers. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pages 3430–3437.

Yuezhe Chen, Lingyun Kong, Yang Wang, and Dezhi Kong. 2021. Multi-grained attention representation with albert for aspect-level sentiment classification. *IEEE Access*, 9:106703–106713.

Chloé Clavel and Zoraida Callejas. 2016. Sentiment analysis: From opinion mining to human-agent interaction. *IEEE Transactions on Affective Computing*, 7(1):74–93.

Alexis Conneau, Kartikay Khandelwal, Naman Goyal, Vishrav Chaudhary, Guillaume Wenzek, Francisco Guzmán, Edouard Grave, Myle Ott, Luke Zettlemoyer, and Veselin Stoyanov. 2020. Unsupervised cross-lingual representation learning at scale. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 8440–8451, Online. Association for Computational Linguistics.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT 2019, Minneapolis, MN, USA, June 2-7, 2019, Volume 1 (Long and Short Papers)*, pages 4171–4186. Association for Computational Linguistics.

Hongliang Fei and Ping Li. 2020. Cross-lingual unsupervised sentiment classification with multi-view transfer learning. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 5759–5771.

Tianyu Gao, Adam Fisch, and Danqi Chen. 2020. Making pre-trained language models better few-shot learners. *arXiv preprint arXiv:2012.15723*.

Zhengjie Gao, Ao Feng, Xinyu Song, and Xi Wu. 2019. Target-dependent sentiment classification with bert. *IEEE Access*, 7:154290–154299.

Qiushan Guo, Xinjiang Wang, Yichao Wu, Zhipeng Yu, Ding Liang, Xiaolin Hu, and Ping Luo. 2020. Online knowledge distillation via collaborative learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 11020–11029.

Lu Hou, Zhiqi Huang, Lifeng Shang, Xin Jiang, Xiao Chen, and Qun Liu. 2020. Dynabert: Dynamic bert with adaptive width and depth. *arXiv preprint arXiv:2004.04037*.

Xiaoqi Jiao, Yichun Yin, Lifeng Shang, Xin Jiang, Xiao Chen, Linlin Li, Fang Wang, and Qun Liu. 2019. Tinybert: Distilling bert for natural language understanding. *arXiv preprint arXiv:1909.10351*.

HAO Jie, XIE Jun, SU Jingqiong, et al. 2016. An unsupervised approach for sentiment classification based on weighted latent dirichlet allocation. *CAAI Transactions on Intelligent Systems*, 11(4):539–545.

Rie Johnson and Tong Zhang. 2015. Semi-supervised convolutional neural networks for text categorization via region embedding. *Advances in neural information processing systems*, 28:919.

Anderson Uilian Kauer and Viviane P. Moreira. 2016. Using information retrieval for sentiment polarity prediction. *Expert Systems with Applications*, 61:282–289.

Zixuan Ke, Hu Xu, and Bing Liu. 2021. Adapting BERT for continual learning of a sequence of aspect sentiment classification tasks. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 4746–4755, Online. Association for Computational Linguistics.

Yoon Kim. 2014. Convolutional neural networks for sentence classification.

Diederik P Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*.

Daliang Li and Junpu Wang. 2019. Fedmd: Heterogenous federated learning via model distillation. *arXiv preprint arXiv:1910.03581*.

Juncen Li, Robin Jia, He He, and Percy Liang. 2018. Delete, retrieve, generate: A simple approach to sentiment and style transfer. *arXiv preprint arXiv:1804.06437*.

Jiangming Liu and Yue Zhang. 2017. Attention modeling for targeted sentiment. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 2, Short Papers*, pages 572–577.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Roberta: A robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692*.

Ilya Loshchilov and Frank Hutter. 2017. Fixing weight decay regularization in adam. *CoRR*, abs/1711.05101.

Diego Marcheggiani and Ivan Titov. 2017. Encoding sentences with graph convolutional networks for semantic role labeling. *arXiv preprint arXiv:1703.04826*.

Subhabrata Mukherjee and Ahmed Awadallah. 2020. Xtremedistil: Multi-stage distillation for massive multilingual models. *arXiv preprint arXiv:2004.05686*.

Georgios Paltoglou and Mike Thelwall. 2010. A study of information retrieval weighting schemes for sentiment analysis. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics*, pages 1386–1395, Uppsala, Sweden. Association for Computational Linguistics.

Jeffrey Pennington, Richard Socher, and Christopher D Manning. 2014. Glove: Global vectors for word representation. In *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*, pages 1532–1543.

Matthew E Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer. 2018. Deep contextualized word representations. *arXiv preprint arXiv:1802.05365*.

A Rakhlin. 2016. Convolutional neural networks for sentence classification. *GitHub*.

Steven Reich, David Mueller, and Nicholas Andrews. 2020. Ensemble distillation for structured prediction: Calibrated, accurate, fast-choose three. *arXiv preprint arXiv:2010.06721*.

Yafeng Ren, Yue Zhang, Meishan Zhang, and Donghong Ji. 2016. Improving twitter sentiment classification using topic-enriched multi-prototype word embeddings. In *Thirtieth AAAI conference on artificial intelligence*.

Victor Sanh, Lysandre Debut, Julien Chaumond, and Thomas Wolf. 2019. Distilbert, a distilled version of bert: smaller, faster, cheaper and lighter. *arXiv preprint arXiv:1910.01108*.

Timo Schick, Helmut Schmid, and Hinrich Schütze. 2020. Automatically identifying words that can serve as labels for few-shot text classification. *arXiv preprint arXiv:2010.13641*.

Timo Schick and Hinrich Schütze. 2020. Exploiting cloze questions for few shot text classification and natural language inference. *arXiv preprint arXiv:2001.07676*.

Chenlin Shen, Changlong Sun, Jingjing Wang, Yangyang Kang, Shoushan Li, Xiaozhong Liu, Luo Si, Min Zhang, and Guodong Zhou. 2018. Sentiment classification towards question-answering with hierarchical matching network. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 3654–3663, Brussels, Belgium. Association for Computational Linguistics.

Siqi Sun, Yu Cheng, Zhe Gan, and Jingjing Liu. 2019. Patient knowledge distillation for bert model compression. *arXiv preprint arXiv:1908.09355*.

Zhiqing Sun, Hongkun Yu, Xiaodan Song, Renjie Liu, Yiming Yang, and Denny Zhou. 2020. Mobilebert: a compact task-agnostic bert for resource-limited devices. *arXiv preprint arXiv:2004.02984*.

Kai Sheng Tai, Richard Socher, and Christopher D Manning. 2015. Improved semantic representations from tree-structured long short-term memory networks. *arXiv preprint arXiv:1503.00075*.

Duyu Tang, Bing Qin, Xiaocheng Feng, and Ting Liu. 2015. Effective lstms for target-dependent sentiment classification. *arXiv preprint arXiv:1512.01100*.

Raphael Tang, Yao Lu, Linqing Liu, Lili Mou, Olga Vechtomova, and Jimmy Lin. 2019. Distilling task-specific knowledge from bert into simple neural networks. *arXiv preprint arXiv:1903.12136*.

Shikhar Vashishth, Shib Sankar Dasgupta, Swayambhu Nath Ray, and Partha Talukdar. 2019. Dating documents using graph convolution networks. *arXiv preprint arXiv:1902.00175*.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in neural information processing systems*, pages 5998–6008.

Devesh Walawalkar, Zhiqiang Shen, and Marios Savvides. 2020. Online ensemble model compression using knowledge distillation. In *European Conference on Computer Vision*, pages 18–35. Springer.

Alex Wang, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel R Bowman. 2018. Glue: A multi-task benchmark and analysis platform for natural language understanding. *arXiv preprint arXiv:1804.07461*.

Jingjing Wang, Changlong Sun, Shoushan Li, Xiaozhong Liu, Luo Si, Min Zhang, and Guodong Zhou. 2019. Aspect sentiment classification towards question-answering with reinforced bidirectional attention network. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 3548–3557, Florence, Italy. Association for Computational Linguistics.

Chuhan Wu, Fangzhao Wu, and Yongfeng Huang. 2021. One teacher is enough? pre-trained language model distillation from multiple teachers. *arXiv preprint arXiv:2106.01023*.

Min Yang, Wenting Tu, Jingxuan Wang, Fei Xu, and Xiaojun Chen. 2017. Attention based lstm for target dependent sentiment classification. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 31.

Jianfei Yu and Jing Jiang. 2019. Adapting bert for target-oriented multimodal sentiment classification. In *Proceedings of the Twenty-Eighth International Joint Conference on Artificial Intelligence, IJCAI-19*, pages 5408–5414. International Joint Conferences on Artificial Intelligence Organization.

Jiangfeng Zeng, Xiao Ma, and Ke Zhou. 2019. Enhancing attention-based lstm with position context for aspect-level sentiment classification. *IEEE Access*, 7:20462–20471.

Xiang Zhang, Junbo Zhao, and Yann LeCun. 2015. Character-level convolutional networks for text classification. *Advances in neural information processing systems*, 28:649–657.

Xuan Zhou, Xiao Zhang, Chenyang Tao, Junya Chen, Bing Xu, Wei Wang, and Jing Xiao. 2021. Multi-grained knowledge distillation for named entity recognition. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 5704–5716.

Making Parameter-efficient Tuning More Efficient: A Unified Framework for Classification Tasks

Xin Zhou[★*], **Ruotian Ma**[★*], **Yicheng Zou**[★], **Xuanting Chen**[★], **Tao Gui**[♦†], **Qi Zhang**[★♣†], **Xuanjing Huang**[★], **Rui Xie**[♠], **Wei Wu**[♠]
[★]School of Computer Science, Fudan University
[♦]Institute of Modern Languages and Linguistics, Fudan University
[♣]Shanghai Key Laboratory of Intelligent Information Processing
[♠]Meituan Inc., Beijing, China
{xzhou20,rtma19,tgui,qz,xjhuang}@fudan.edu.cn

Abstract

Large pre-trained language models (PLMs) have demonstrated superior performance in industrial applications. Recent studies have explored parameter-efficient PLM tuning, which only updates a small amount of task-specific parameters while achieving both high efficiency and comparable performance against standard fine-tuning. However, all these methods ignore the inefficiency problem caused by the task-specific output layers, which is inflexible for us to re-use PLMs and introduces non-negligible parameters. In this work, we focus on the text classification task and propose *plugin-tuning*, a framework that further improves the efficiency of existing parameter-efficient methods with a unified classifier. Specifically, we re-formulate both token and sentence classification tasks into a unified language modeling task, and map label spaces of different tasks into the same vocabulary space. In this way, we can directly re-use the language modeling heads of PLMs, avoiding introducing extra parameters for different tasks. We conduct experiments on six classification benchmarks. The experimental results show that plugin-tuning can achieve comparable performance against fine-tuned PLMs, while further saving around 50% parameters on top of other parameter-efficient methods.

1 Introduction

In many industrial applications about natural language processing, it becomes a de-facto paradigm that we first pre-train large-scale language models (PLM) (Devlin et al., 2019; Peters et al., 2018; Radford et al., 2019) on external corpora and then fine-tune them on target tasks. However, each fine-tuned PLM is generally applicable to only one task. As the number of applications increases, deploying independent instances of fine-tuned PLMs for different tasks significantly increases the computation and storage costs. For instance, GPT-3 (Brown et al., 2020) contains 175B parameters, which makes it almost impossible to fine-tune and deploy GPT-3 for target tasks. Even if large-scale fine-tuned PLMs are available, they are also not conducive to community distribution and sharing.

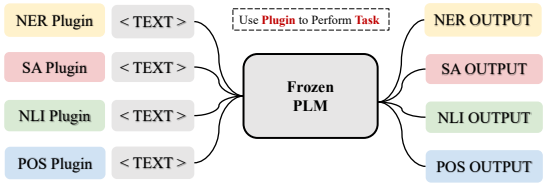

Figure 1: An intuitive overview of plugin-tuning. Plugin tuning only trains and stores a parameter-efficient plugin rather than a full PLM for each task. Different tasks share a unified output layer despite different label spaces.

Recently, a new branch of researches named parameter-efficient tuning (Guo et al., 2021; Zaken et al., 2021; Hu et al., 2022) have received much attention in the NLP community. Compared to standard fine-tuning, these methods fine-tune only a small portion of the model parameters while keeping most of the PLM parameters frozen (Ding et al., 2022). The minimal trainable parameters not only remarkably promote the deployment and storage efficiency when adapting PLMs, but also make it feasible to train large-scale PLMs such as GPT-3. Additionally, recent works validate that the parameter-efficient methods can achieve comparable performance with full-parameters fine-tuning in a wide range of NLP applications (Aghajanyan et al., 2020; Hu et al., 2022; Ding et al., 2022).

However, all the existing parameter-efficient methods ignore the inefficiency problem caused by the specific output layers for different tasks, which results in two problems. First, utilizing different task paradigms and retaining different output layers

[*] Equal Contribution.
[†] Corresponding authors.

Proceedings of the 29th International Conference on Computational Linguistics, pages 7053–7064
October 12–17, 2022.

is inflexible and hinders the community model distribution and sharing. Second, the parameters introduced by the task-specific output layers are non-negligible especially under the parameter-efficient scenarios. These two essential issues limit parameter-efficient tuning from being further efficient.

In this work, we propose plugin tuning, a framework to further improve all the parameter-efficient methods in classification tasks. We unify all the classification tasks into a same paradigm, thus different tasks can be performed with a unified classifier. Meanwhile, no trainable parameter is required for training the classifier. Specifically, we re-formulate different tasks into a same language modeling task, and directly reuse the language model heads of PLMs for classification. The label spaces of different tasks are mapped to task-specific label words, all belonging to a unified vocabulary space. To select the proper label words, we propose a principled algorithm that is applicable to both token and sentence classification tasks. In this way, the efficiency of all the parameter-efficient methods can be largely promoted. Additionally, the unified task paradigm provides a new way to perform classification tasks with generative PLMs such as GPT-3, which is unexplored in existing parameter-efficient methods. Our codes are publicly available at *Github*[1].

Our contributions can be summarized as follows:

- We propose a unified paradigm for all classification tasks to further improve the efficiency based on all parameter-efficient methods.
- We propose a principled way to select proper label words for both token and sentence classification tasks. Intensive experiments are conducted to analyze the proposed method.
- Plugin-tuning can achieve comparable performance with full-size fine-tuning on six tasks, as well as save up to 50% of the parameters required by the parameter-efficient tuning.

2 How to Boost Efficiency of Fine-tuning?

2.1 Standard Fine-tuning

We start with the introductions of fine-tuning PLM on token and sentence classification tasks, followed by the disadvantages of fine-tuning.

[1]https://github.com/xzhou20/Plugin-tuning

Given a sequence of tokens $\mathbf{X} = [x_1, ...x_n]$, the PLM usually encodes $\mathbf{X}$ with multi-layer bidirectional Transformer and outputs its representation of final hidden state:

$$\mathbf{H} = encoder_{\Phi}(\mathbf{X}), \tag{1}$$

where $\mathbf{H} \in \mathbb{R}^{n \times d_h}$, d_h is the dimension of the hidden state and Φ is the parameters of transformer layers. After encoding, a simple softmax classifier is added to the top of encoder to predict the target label based on the task.

Suppose we are given a sentence classification task S with label space $L_S \in \mathbb{R}^{ls}$. The aim of S is to predict a label $y \in L_S$ of sentence $\mathbf{X}$:

$$p(y|\mathbf{h}_{cls}) = Softmax(\mathbf{W}_S\mathbf{h}_{cls}), \tag{2}$$

while a token classification task T aims to predict a label $y_i \in L_T$ of each x_i in sentence $\mathbf{X}$:

$$p(y_i|\mathbf{h}_i) = Softmax(\mathbf{W}_T\mathbf{h}_i), \tag{3}$$

where $\mathbf{W}_S \in \mathbb{R}^{d_h \times l_S}$ and $\mathbf{W}_T \in \mathbb{R}^{d_h \times l_T}$.

During training phase, the standard fine-tuning directly updates all parameters to minimize the following objective:

$$Sentence: \min_{\Phi_S, \mathbf{W}_S} -\log(P(y|\mathbf{X})) \qquad y \in L_S, \tag{4}$$

$$Token: \min_{\Phi_T, \mathbf{W}_T} -\sum_{i}^{n} \log(P(y_i|\mathbf{X})) \quad y_i \in L_T, \tag{5}$$

where Φ_S, $\mathbf{W}_S$, Φ_B and $\mathbf{W}_T$ are fine-tuned parameters for target tasks. A major drawback of fine-tuning is the need to update all parameters of the PLM. Updating large-scale Φ results in significant computation costs, and fine-tuned parameters Φ_S and Φ_B are also inefficient to store and deploy.

2.2 Not All Parameters are Critical

The question to be asked here is: do we really need to update all the parameters? Although the theoretical questions are not well explored, empirical evidence shows that: not all parameters are critical to be updated (Aghajanyan et al., 2020; Chen et al., 2020). Recent parameter-efficient tuning with distinct tuned parameter selection could achieve comparable performance to standard fine-tuning. In this way, we do not fine-tuning the large-scale Φ, but instead update selected $\Delta\Phi$

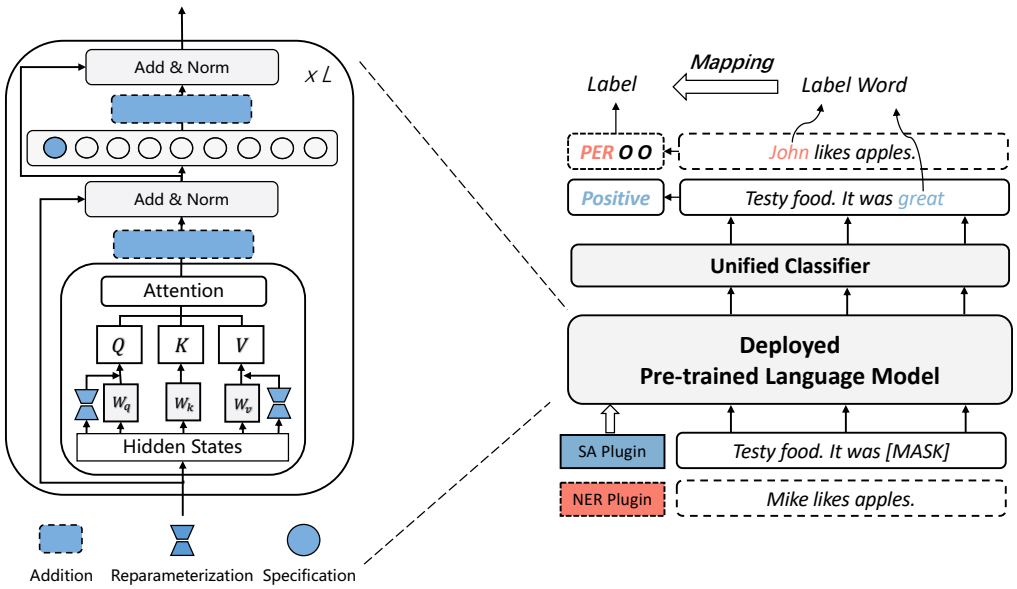

(a) Three types of parameter-efficient methods

(b) An example of plugin-tuning

Figure 2: An overview of plugin-tuning. (a) is the three types of parameter-efficient methods, which can be used as our plugin. (b) shows the flow of plugin-tuning, we input the text and task-specific plugin to the deployed PLM, the PLM influenced by plugin outputs the label words in the sentence. The actual labels are obtained by label word mapping. "John" and "great" are label words selected by querying language model, respectively representing label "PER" for task NER and label "Positive" for task SA.

where $|\Delta\Phi| \ll |\Phi|$. Ding et al. (2022) divides existing parameter-efficient methods into three groups: (1) addition-based methods (Houlsby et al., 2019) introduce extra trainable neural modules or parameters that do not exist in the original model or process. (2) specification-based methods (Zaken et al., 2021) only update certain parameters in the original model. (3) reparameterization-based methods (Hu et al., 2022) transform existing parameters to a smaller size. An example of above methods is shown in Figure 2 (a). In this paper we refer to these task-specific parameters $\Delta\Phi$ collectively as **plugin** for convenience, regardless which parameter-efficient method is used.

2.3 Output Layers: An Overlooked Problem

A critical problem overlooked in parameter-efficient methods is the efficiency problem of different output layers. Aghajanyan et al. (2020) shows that only a few thousand parameters are enough to handle various NLP tasks. However, no matter how efficient the method is, the task-specific output layers introduce non-negligible parameters, especially when encountering a large label space. (e.g., ten labels will introduce 10,240 parameters of classifier when using bert-large) . In addition to efficiency, parameter-efficient methods keep the model parameters consistent with pre-trained parameters, but we do not keep the optimization objective consistent with pre-training. The gap between downstream tasks and pre-training tasks may lead to sub-optimal performance (Liu et al., 2021a). This problem can not handle easily because of the different label spaces for different tasks. Therefore, the task-specific output layers hinder the further improvement of effectiveness and efficiency for all parameter-efficient methods, which cannot be overlooked.

3 Our Method

In this work, we propose plugin-tuning, a unified framework to improve parameter-efficient tuning. Our method is based on parameter-efficient tuning but avoids the problems caused by task-specific output layers. An overview of plugin-tuning is shown in Figure 2. The plugin here refer to the trainable parameters required for parameter-efficient method. We inject the plugin to the frozen PLM, and the plugin is applied to a specific position to control the PLM, as shown in Figure (a). The PLM will predict the label word in label-related position of sentence, and the real label can be obtained by label word mapping. Any parameter-efficient method can serve as our plugin. We only optimize the lightweight plugin during training

while keeping other parameters frozen. The details of the unified classifier are shown in the following subsections.

3.1 Unified Classifier

As shown in Section 2.3, the different label spaces make it hard to share a unified output layer. Inspired by prompt-base tuning (Liu et al., 2021b), we find that a label can be represented by a word. In this manner, different label spaces can be mapped to the same vocabulary space. Thus, we propose a unified classifier to solve this problem by re-using the language model head. To this end, we reformulate token and sentence classification tasks as a unified language modeling task, the PLM is trained to predict a label word as an indication of the real label. Furthermore, we propose an algorithm to search label words automatically by querying the language model.

Take sentiment classification as an example, we insert the "It was [MASK]." to the end of the sentence, and the PLM is expected to predict a word that indicates the sentiment at the position of [MASK]. As for token classification such as named entity recognition (NER), the PLM are trained to predict a a word that is more common than the original entity word, as shown in Figure 2 (b).

Formally, suppose we are given a classification task with label space L, a vocabulary V, a label word mapping function $M : L \rightarrow V$ and a input $\mathbf{X} = [x_1, ..., x_n]$. The classification task is reformulated to assign a word $y_i \in V$ to the special position:

$$p(y_i|\mathbf{X}) = Softmax(\mathbf{W}_{lm}\mathbf{h}_i), \tag{6}$$

where $\mathbf{h}$ is the final hidden state and i is the label-related position(position of [MASK] for sentence classification, all positions for token classification). In this way, different tasks can re-use the language model head $\mathbf{W}_{lm}$ as the unified classifier, diverse label space can be all adapted to the vocabulary V. Both token and sentence classification tasks have a unified language modeling objective:

$$Unified : \min_{\Delta\Phi} - \sum_{i \in Y_{idx}} \log(P(y_i|\mathbf{X})) \qquad y_i \in V, \tag{7}$$

where $\Delta\Phi$ is the trainable parameters required by parameter-efficient tuning and Y_{idx} are label-related positions, y_i is the label word, which is mapped according to its label. We make the word of label "O" predict itself in token classification. In prediction phase, we take the word with the highest probability in the PLM's prediction and map it to real labes by label word mapping function M.

Algorithm 1 Label Word Searching

Input: Training Set $D = \{X_i, Y_i\}_{i=1}^{N}$.
Parameter: Pre-trained language model LM.
Output: Label Word Map M.

```
Let label word map M = ∅;
for c ∈ L do
    Let freq_c = ∅;
end for
for (X = {x_i}_{i=1}^n, Y = {y_i}_{i=1}^n) ∈ D do
    Let Ŷ = LM(X);
    if i is related to a label c then
        Select the ten words with the highest probability from ŷ_i and update freq_c;
    end if
end for
for c ∈ L do
    The most frequent word w_freq ∈ freq_c is selected as label word of c, M[c] ← w_freq;
end for
return M
```

3.2 Label Word Search

To avoid the cumbersome label word engineering, we propose a method to search the label word automatically. To keep the consistency of pre-training tasks and downstream tasks, we leverage the original PLM for label word searching. Since the classification tasks are reformulated to language modeling task, a appropriate label word can be chosen by the PLM itself. We directly leverage the original pre-trained language model predict the candidate words at their label-related position and select the appropriate labeled words based on the frequency of candidate words under that label. The process is shown in Algorithm 1.

4 Experiment

We verify the effectiveness of the plugin-tuning method on a large number of tasks including both token and sentence classification tasks, and select representative parameter-efficient methods to demonstrate that plugin-tuning is compatible with arbitrary parameter-efficient methods. In addition, we also simulate real scenarios and counted the

deployment time to further verify the efficiency of our method in deployment. Finally, we provide a comprehensive analysis of the unified classifier, which is the main module of the plugin-tuning.

4.1 Dataset

To verify the effectiveness of the our method, we conduct experiments on six representative classification tasks. Next, we describe the datasets selected for token classification and sentence classification respectively. The statistics of the datasets are summarized in Table 1.

We evaluate the proposed method on three token classification tasks, including Named Entity Recognition (NER), Part of Speech tagging (POS) and text chunking (Chunking). For named entity recognition, we select the CoNLL 2003 (Sang and De Meulder, 2003), a newswire domain benchmark. For part of speech tagging, we select the Wall Street Journal (WSJ) data from the Penn TreeBank v3 (Marcus et al., 1993). For text chunking, we select the CoNLL 2000 (Tjong Kim Sang and Buchholz, 2000). BIO2 tagging scheme is used for NER and chunking.

As for sentence classification tasks, we select three common tasks, Sentiment analysis (SA), Natural Language Inference (NLI) and Question Classification (QC). We select the dataset based on the amount of data from GLUE (Wang et al., 2018). For SA, we chose the commonly used SST2. For NLI, we chose MNLI, a dataset from GLUE with a large amount of data to challenge our method. For QA, we choose TREC50 with only thousands of data to explore whether plugin-tuning can perform well in this scenario. The test sets for these three datasets are not publicly available, so we use the original validation set directly as the test set.

Task	Dataset	#Train	#Test	Labels
NER	CoNLL2003	204,567	46,666	9
CHK	CoNLL2000	211,727	47,377	23
POS	WSJ	912,344	129,654	46
SA	SST2	67,350	873	2
NLI	MNLI	392,702	9815	3
QC	TREC50	5452	500	47

Table 1: Statistics of the datasets. # means the number of sentences in SA, NLI and RC, and means the number of tokens in the NER, Chunking and POS.

4.2 Baseline

We select representative methods from the three types of delta tuning to verify whether plugin tuning is compatible with different delta tuning methods. Besides that, the standard fine-tuning is used to show the proper performance of PLMs.

Fine-Tuning (Liu et al., 2019) optimizes all parameters of the PLM for each task, which is the main approach for transferring the PLM to downstream tasks.

LoRA (Hu et al., 2022) is a reparameterization-based baseline. LoRA injects trainable low-rank matrices into transformer layers to approximate the weight updates, which can achieve good performance with extremely small parameters.

BitFit (Zaken et al., 2021) is a specification-based baseline. It only trains the bias term and classifier in the PLM.

Adapter (Houlsby et al., 2019) is a popular addition-based method of parameter-efficient tuning, which adds adapter layers to every transformer blocks. Only the adapter layers and the classifier are trainable during the training phase.

Plugin-tuning is our proposed method, which can be applied to any parameter-efficient methods to boost its efficiency and performance.

4.3 Implementation Details

In this work, we implement the parameter-efficient methods with roberta-base (Liu et al., 2019). Each parameter-efficient method follows the official code. AdamW optimizer and linear decaying schedule are used for all baselines. We search the learning rate from 1e-3 to 1e-5, epochs from {5, 10, 30} with a batch size of 16. We report the best results on the test set for each task. The model and hyperparameters are selected based on the validation set. The label word searching is also based on roberta-base. The selected label word is shown in the appendix.

4.4 Main Results

In this section, we verify the impact of using plugin-tuning on the performance and parameters of parameter-efficient tuning on six tasks. The results are shown in Table 2. A detailed analysis of the experimental results is presented below.

Parameter Efficiency Benefiting from the unified classifier, plugin-tuning can further reduce task-specific parameters required by parameter-efficient tuning on all tasks, and its effect is

Task/Method		LoRA Ori → Plugin-tuning	BitFit Ori → Plugin-tuning	Adapter Ori → Plugin-tuning
NER	F1	91.22 → 91.22 (↑0.00)	91.17 → 91.05 (↓ 0.12)	91.13 → 91.34((↑0.21)
	Param	44.3k → 36.9k (↓15%)	161k → 153k (↓4%)	127k → 120k (↓5%)
CHK	F1	95.58 → 96.34 (↑1.95)	93.25→94.48(↑1.23)	96.10 → 96.89 (↑0.79)
	Param	61.4k → 36.9k (↓29%)	172k → 153k (↓10%)	138k → 120k (↓13%)
POS	Acc.	96.84 → 97.42 (↑0.58)	97.03 → 97.57 (↑0.54)	97.51 → 97.54 (↑0.03)
	Param	72.2k → 36.9k (↓49%)	189k → 153k (↓18%)	156k → 120k (↓23%)
SA	Acc.	93.89 → 94.15 (↑0.16)	94.04 → 94.03 (↓0.01)	94.95 → 94.56 (↓0.39)
	Param	38.4k → 36.9k (↓4%)	156k → 153k (↓1%)	121k → 120k (↓1%)
NLI	Acc.	85.45 → 85.46 (↑0.01)	84.61 → 84.53 (↓0.08)	86.26 → 86.36 ((↑0.10)
	Param	39.2k → 36.9k (↓6%)	156k → 153k (↓1%)	122K → 119K (↓2%)
QC	Acc.	92.20 → 92.80 (↑0.60)	91.80 → 92.20 (↑0.40)	92.80 → 93.20 (↑0.40)
	Param	73.0k → 36.9k (↓49%)	190k → 153k (↓19%)	156k → 120k (↓23%)

Table 2: Main Results for six classification tasks. We report F1 score for Chunking and NER, accuracy for POS, SA, NLI and QC. Higher is better for these metrics. Param means the number of task-specific parameters; k stands for thousand. Lower is more efficient for the Param. We compare the performance and parameters on the Original baseline and the baseline with Plugin-tuning (Ori → Plugin-tuning). The change in performance and the percentage of reduced parameters are calculated. Items marked in red indicate that plugin-tuning boosts its performance or efficiency.

particularly significant on tasks with many labels. For tasks with more than 40 labels such as POS, plugin-tuning can reduce the number of parameters by up to 49%. For the theoretical worst case, on the sentiment classification task with only two labels and the NLI task with three labels, the advantage of our method is not significant, the percentage of parameter reduction is at least 1%. At current stage, plugin-tuning is more suitable for scenarios where the number of labels is large. As parameter-efficient tuning develops, the amount of task-specific parameters required will be further reduced, and the advantages of plugin-tuning will gradually become more pronounced.

Overall Performance In addition to the efficiency of the parameters, plugin-tuning also achieves performance improvements on most tasks, and this improvement is mainly seen on the token-level classification task. The most significant improvement is in the chunking task, with an average of 1 point improvement on the three datasets. We speculate that this is because the unified classifier reuses the parameters of the pre-trained model, and the gap between its training target and the pre-trained task is small, so it can benefit from the rich knowledge hidden in the pre-trained model. Token-level tasks with many categories and a high degree of detail benefit more significantly from them. For the sentence-level classification task, there are both gains and losses in performance after applying plugin-tuning, but the overall impact on performance is not significant. We found that neither performance gains nor performance drops were significant on tasks with large amounts of data. We conjecture that this is because neither the task-specific classifier nor the unified classifier is a dominant factor for performance when the data is large.

Comparison with Fine-tuning We selected the best performance among all plugin-tuning results to compare with standard fine-tuning. The results are shown in Table 3. Although the overall performance is inferior to fine-tuning, the difference between plugin-tuning and fine-tuning has been negligible. Since number of parameters of plugin-tuning is only 0.01% of fine-tuning, plugin-tuning can be used as an efficient and effective alternative to fine-tuning.

Model	POS	NER	CHK	SA	NLI	QC
Fine-tuning	97.69	91.45	97.03	94.72	87.60	93.20
Plugin-tuning	97.42	91.34	96.89	94.56	86.36	93.20

Table 3: Comparison of standard fine-tuning and plugin-tuning.

Analysis of Different Tasks and Models Our method is most useful for Adapter and LoRA, and is not stable on BitFit. However, the differences in

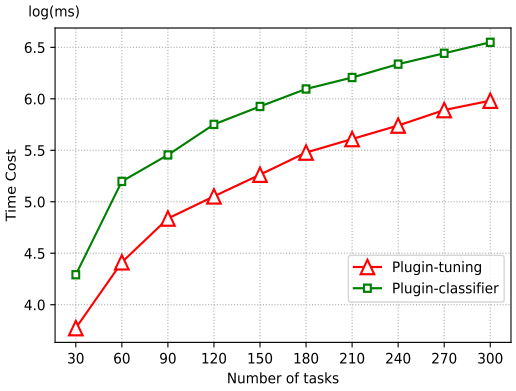

Figure 3: Time cost of redeployment. We take the log value to better show the results.

parameter-efficient methods are not as significant as the impact of the differences between tasks. From a task perspective, BitFit is unstable and performs poorly on larger datasets. Adapter and LoRA achieve similar and better performance on most tasks, but LoRA requires a smaller number of parameters and is therefore more efficient.

4.5 Efficiency Advantage

In this section, we study the difference in the efficiency of different parameter-efficient methods on deployment. We focus on redeployment, i.e., when a new task arrives, we want to release the resources of the old model and deploy a model that can execute the new task. Redeployment is suitable for scenarios where computational resources are tight. In online environments, the tasks from users arrive in the stream, and these tasks are diverse and hardly the same. The wide variety of tasks makes models need to be redeployed frequently, and in time-sensitive systems like search engines, the time of redeployment is part of the consideration.

To simulate the online setting, we construct the task streams by randomly sampling 30 to 300 samples from six tasks and show the time cost for switching tasks, the redeployment strategy is used at the arrival of each task. To show the disadvantage of the task-specific classifier, we use plugin-tuning based on LoRA, which replaces the task-specific classifier with a unified one. For a fair comparison, we set the task-specific parameters of each baseline to 36864 (note that LoRA need additional parameters for the classifier).

The trend of time cost is shown in Figure 3. We take the log value to better show the results. Although the parameters are the same scale, we find plugin-Classifier takes an additional 50% of time to reload the task-specific classifier (It will be longer on tasks with more labels), it demonstrates the efficiency of the unified classifier. These results can show the efficiency advantages of plugin-tuning over other parameter-efficient methods.

4.6 Analysis of Label Word Searching

Although the efficiency and performance of plugin-tuning have been verified by previous experiments, another part of plugin-tuning, the label word, still needs to be further analyzed. In this section, we show the necessity of label word searching and discuss the sample efficiency of our proposed algorithm 1.

4.6.1 Effectiveness Study

There are many ways to select a label word, it is still a question whether our method can select the right label words. In our view, a suitable label word needs to have at least no negative impact on the model and be easy to obtain. Therefore, in this section, we compare three different label word construction methods. (1) Each label word is a **handcraft**,designed by Human. the human-created label word sets are shown in Appendix. (2) Each label word is a **VirtualToken** in the vocabulary, the embedding of special token is randomly initialized. (3) Each label word is selected by our proposed **AutoSearch** algorithm.

We apply plugin-tuning to LoRA and test the performance of the three label word construction methods on different tasks based on the same parameter settings. The experimental results are shown in Table 4. When using VirtualToken, the performance drops significantly for tasks with many labels (CoNLL-NER), indicating that the random vector is hard for a model to fit. The results of Human shows that human intuition may be the same as the pre-trained model, which explains the good performance SST-2. However, on the token classification task, the human-created label word may not be consistent with the task goal, leading to bad results on CoNLL-NER. Our proposed AutoSearch can effectively leverage the knowledge of pre-trained models for downstream tasks, thus it outperforms other methods on all tasks.

4.6.2 Sample Efficiency

Our proposed label word search algorithm requires some labeled data to find the suitable label word. A concern here is how much data it needs to find such label words. The sample efficiency is important

Method	SST-2	CoNLL-NER
AutoSearch	**94.15**	**91.22**
VirtualToken	93.80	90.07
Handcraft	94.03	90.09

Table 4: Comparison of label word searching methods.

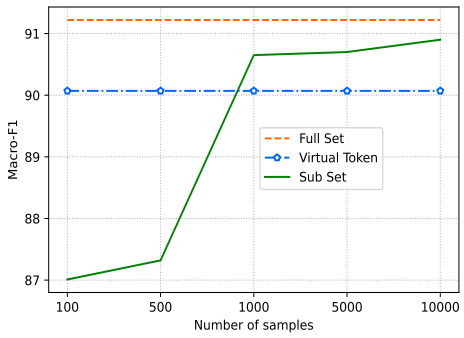

Figure 4: Performance variation of the NER task on the CoNLL03 when using different numbers of samples for searching label word.

since traversing the entire dataset can be time-consuming. Therefore, we select [100, 500, 1000, 5000, 10000] data uniformly from the train set, and search the label words in these subsets to verity the sample efficiency. The results in Figure 4 show that the 1000 samples are enough for achieving comparable performance. This indicates that our algorithm is data efficient.

5 Related Work

5.1 Pre-trained Language Model

Large-scale pre-trained language models, which pre-trained on a huge amount of data with self-supervised objectives, have greatly improved the performance of various downstream tasks (Qiu et al., 2020). Fine-tuning the pre-trained language model such as BERT (Devlin et al., 2019), roberta (Liu et al., 2019), and T5 (Liu et al., 2019) achieves state-of-art performance on various tasks. Increasing the number of PLM parameters is one of the intuitive ways to enhance its performance, thus leading to the creation of many giant models (Brown et al., 2020). But it is inconvenient to fine-tuning these models and apply fine-tuned models because of the large-scale parameters. They are also not conducive to community distribution and sharing. In this work, we explore the parameter-efficient tuning, which makes the deployment of PLMs feasible in many industrial scenarios.

5.2 Parameter-efficient Learning

Parameter-efficient learning, is proposed to solve the problem caused by PLMs' large-scale parameters. Unlike traditional methods such as distillation (Sanh et al., 2019), pruning (Michel et al., 2019) and quantization (Shen et al., 2020) that directly reduce the parameters of the model itself, parameter-efficient methods only fine-tune a small portion of the model parameters while keeping the rest untouched or learn external modules for new task. The rationale behind these methods can be related to the intrinsic dimension (Li et al., 2018; Aghajanyan et al., 2020), which states that PLM are often overparameterized and only need to learn a good solution in a small parameter space. Or prompt-tuning (Liu et al., 2021a), which originated from GPT-3, by entering special text to let the frozen PLM perform the target task. Recently, Ding et al. (2022) proposed to call these methods delta-tuning and analyzed it from a control theory perspective. Many attempts have been made to find the which part of parameters is efficient to learn, such as adapter (Houlsby et al., 2019), prefix-tuning(Li and Liang, 2021), and LoRA(Hu et al., 2022). In this work, we boost the efficiency of parameter-efficient tuning by avoiding the problem caused by task-specific classifier.

6 Conclusion

In this work, we propose plugin-tuning, a unified framework to improve parameter-efficient tuning for both token and sentence classification tasks. We use a unified classifier to handle different classification tasks by re-formulating them to a unified language modeling task, where no trainable parameter is required for training the classifier. To select the proper label words, we also propose a principled algorithm that is applicable to both token and sentence classification tasks. In this way, the efficiency of all the parameter-efficient methods can be largely promoted. The experiments show that our method achieves comparable performance against fine-tuned PLMs while further saving up to 49% parameters on top of other parameter-efficient methods. Future directions might be applying the unifying strategy into other tasks like text matching that require specific label spaces.

Acknowledgements

The authors wish to thank the anonymous reviewers for their helpful comments. This work was partially National Natural Science Foundation of China (No. 62076069, 61976056), Shanghai Municipal Science and Technology Major Project (No.2021SHZDZX0103). This research was supported by Meituan.

References

Armen Aghajanyan, Luke Zettlemoyer, and Sonal Gupta. 2020. Intrinsic dimensionality explains the effectiveness of language model fine-tuning. *arXiv preprint arXiv:2012.13255*.

Tom B. Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel M. Ziegler, Jeffrey Wu, Clemens Winter, Christopher Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. 2020. Language models are few-shot learners.

Tianlong Chen, Jonathan Frankle, Shiyu Chang, Sijia Liu, Yang Zhang, Zhangyang Wang, and Michael Carbin. 2020. The lottery ticket hypothesis for pre-trained bert networks. *Advances in neural information processing systems*, 33:15834–15846.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, Minneapolis, Minnesota. Association for Computational Linguistics.

Ning Ding, Yujia Qin, Guang Yang, Fuchao Wei, Zonghan Yang, Yusheng Su, Shengding Hu, Yulin Chen, Chi-Min Chan, Weize Chen, Jing Yi, Weilin Zhao, Xiaozhi Wang, Zhiyuan Liu, Hai-Tao Zheng, Jianfei Chen, Yang Liu, Jie Tang, Juanzi Li, and Maosong Sun. 2022. Delta tuning: A comprehensive study of parameter efficient methods for pre-trained language models.

Demi Guo, Alexander M. Rush, and Yoon Kim. 2021. Parameter-efficient transfer learning with diff pruning.

Neil Houlsby, Andrei Giurgiu, Stanislaw Jastrzebski, Bruna Morrone, Quentin de Laroussilhe, Andrea Gesmundo, Mona Attariyan, and Sylvain Gelly. 2019. Parameter-efficient transfer learning for nlp.

Edward J Hu, yelong shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. 2022. LoRA: Low-rank adaptation of large language models. In *International Conference on Learning Representations*.

Chunyuan Li, Heerad Farkhoor, Rosanne Liu, and Jason Yosinski. 2018. Measuring the intrinsic dimension of objective landscapes.

Xiang Lisa Li and Percy Liang. 2021. Prefix-tuning: Optimizing continuous prompts for generation. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 4582–4597, Online. Association for Computational Linguistics.

Pengfei Liu, Weizhe Yuan, Jinlan Fu, Zhengbao Jiang, Hiroaki Hayashi, and Graham Neubig. 2021a. Pre-train, prompt, and predict: A systematic survey of prompting methods in natural language processing. *arXiv preprint arXiv:2107.13586*.

Pengfei Liu, Weizhe Yuan, Jinlan Fu, Zhengbao Jiang, Hiroaki Hayashi, and Graham Neubig. 2021b. Pre-train, prompt, and predict: A systematic survey of prompting methods in natural language processing.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Roberta: A robustly optimized bert pretraining approach.

Mitchell P. Marcus, Beatrice Santorini, and Mary Ann Marcinkiewicz. 1993. Building a large annotated corpus of English: The Penn Treebank. *Computational Linguistics*, 19(2):313–330.

Paul Michel, Omer Levy, and Graham Neubig. 2019. Are sixteen heads really better than one? In *Advances in Neural Information Processing Systems*, volume 32. Curran Associates, Inc.

Matthew E Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer. 2018. Deep contextualized word representations. *arXiv preprint arXiv:1802.05365*.

Xipeng Qiu, Tianxiang Sun, Yige Xu, Yunfan Shao, Ning Dai, and Xuanjing Huang. 2020. Pre-trained models for natural language processing: A survey. *Science China Technological Sciences*, 63(10):1872–1897.

Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, Ilya Sutskever, et al. 2019. Language models are unsupervised multitask learners. *OpenAI blog*, 1(8):9.

Erik F Sang and Fien De Meulder. 2003. Introduction to the conll-2003 shared task: Language-independent named entity recognition. *arXiv preprint cs/0306050*.

Victor Sanh, Lysandre Debut, Julien Chaumond, and Thomas Wolf. 2019. Distilbert, a distilled version of bert: smaller, faster, cheaper and lighter. *arXiv preprint arXiv:1910.01108*.

Sheng Shen, Zhen Dong, Jiayu Ye, Linjian Ma, Zhewei Yao, Amir Gholami, Michael W. Mahoney, and Kurt Keutzer. 2020. Q-bert: Hessian based ultra low precision quantization of bert. *Proceedings of the AAAI Conference on Artificial Intelligence*, 34(05):8815–8821.

Erik F. Tjong Kim Sang and Sabine Buchholz. 2000. Introduction to the CoNLL-2000 shared task chunking. In *Fourth Conference on Computational Natural Language Learning and the Second Learning Language in Logic Workshop*.

Alex Wang, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel R Bowman. 2018. Glue: A multi-task benchmark and analysis platform for natural language understanding. *arXiv preprint arXiv:1804.07461*.

Elad Ben Zaken, Shauli Ravfogel, and Yoav Goldberg. 2021. Bitfit: Simple parameter-efficient fine-tuning for transformer-based masked language-models.

A Label Word and Template

In this section, we show the template and label word in Table 5, Table 6 and Table 7. These tables are shown in the next page.

Dataset	Template	Label Word Set
SST-2	[X] It was $\langle mask \rangle$.	{positive: great, negative: terrible}
MNLI	[PREMISE] ? $\langle mask \rangle$ [HYPOTHESIS]	{entailment: Also, contradiction: But, nautral: Yeah}

Table 5: Templates and label words of plugin-tuning for sentence classification

Dataset	Label Word Set
CoNLL03	{B-ORG: United, B-MISC: American, B-PER: Paul, I-PER: Smith, B-LOC: France, I-ORG: Inc, I-MISC: Cup, I-LOC: York}
CoNLL2000	{B-NP: the, B-PP: of, I-NP: and, B-VP: is, I-VP: be, B-SBAR: that, B-ADJP: more, B-ADVP: also, I-ADVP: much, I-ADJP: lower, I-SBAR: if, I-PP: as, B-PRT: up, B-LST: 7, B-INTJ: yes, I-INTJ: was, B-CONJP: As, I-CONJP: well, I-PRT: or, B-UCP: wines, I-UCP:}
WSJ	{NNP: Mr, VBZ: is, JJ: first, NN: year, TO: to, VB: be, .: ., CD: million, DT: the, VBD: said, IN: in, PRP: he, NNS: people, VBP: have, MD: will, VBN: been, POS: "", JJR: more, ": ", RB: also, ,: „, FW: v, CC: and, WDT: which, (: (,):), :: -, PRP$: his, RBR: less, VBG: going, EX: There, WP: who, WRB: when, $: $, RP: up, NNPS: Yankees, SYM: /, RBS: most, UH: O, PDT: all, "": "", LS: 3, JJS: best, WP$: whose, NN|SYM: TV }

Table 6: Label words of plugin-tuning on token classification tasks

Dataset	Template	LabelWordSet
SST-2	[X] It was $\langle mask \rangle$.	{positive: good, negative: bad}
CoNLL03	-No Template-	{B-LOC:location, I-LOC:place, B-PER:person, I-PER:human, B-MISC: entity, I-MISC:other, B-ORG:organization, I-ORG:party},

Table 7: Human-created label words for ablation study

A Multi-Task Dual-Tree Network for Aspect Sentiment Triplet Extraction

Yichun Zhao[1], **Kui Meng**[1], **Gongshen Liu**[1*]
Jintao Du[2], **Huijia Zhu**[2]
[1]Shanghai Jiao Tong University
[2]Ant Group Co., Ltd.
[1]{zhaoyichun, mengkui, lgshen}@sjtu.edu.cn
[2]{lingke.djt, huijia.zhj}@antgroup.com

Abstract

Aspect Sentiment Triplet Extraction (ASTE) aims at extracting triplets from a given sentence, where each triplet includes an aspect, its sentiment polarity, and a corresponding opinion explaining the polarity. Existing methods are poor at detecting complicated relations between aspects and opinions as well as classifying multiple sentiment polarities in a sentence. Detecting unclear boundaries of multi-word aspects and opinions is also a challenge. In this paper, we propose a Multi-Task Dual-Tree Network (MTDTN) to address these issues. We employ a constituency tree and a modified dependency tree in two sub-tasks of Aspect Opinion Co-Extraction (AOCE) and ASTE, respectively. To enhance the information interaction between the two sub-tasks, we further design a Transition-Based Inference Strategy (TBIS) that transfers the boundary information from tags of AOCE to ASTE through a transition matrix. Extensive experiments are conducted on four popular datasets, and the results show the effectiveness of our model.

1 Introduction

Aspect Based Sentiment Analysis (ABSA), also known as Target Based Sentiment Analysis, has received widespread attention in both academia and industry in recent years. ABSA allows a sentiment analysis of different aspects in a given sentence, which can be applied in many areas, such as social media and E-commerce reviews. Compared with sentence sentiment analysis, ABSA is more fine-grained and more in line with reality. ABSA contains many sub-tasks, such as Aspect Term Extraction (ATE) (Xu et al., 2018; Yang et al., 2020), Opinion Term Extraction (OTE) (Fan et al., 2019; Wu et al., 2020b), Aspect Level Sentiment Classification (ALSC) (Xiao et al., 2021; Li et al., 2021), Aspect Sentiment Pair Extraction (ASPE) (Li et al., 2019; Ji et al., 2020; Chen and Qian, 2020; Luo et al., 2020), Aspect Opinion Co-Extraction (AOCE) (Dai and Song, 2019), Aspect Opinion Pair Extraction (AOPE) (Chen et al., 2020; Zhao et al., 2020) and Aspect Sentiment Triplet Extraction (ASTE) (Wu et al., 2020a; Chen et al., 2021a).

Table 1: An example of different ABSA sub-tasks. Aspects, opinions and sentiment polarities are in blue, red and green respectively.

Sentence:	Good service but poor taste
Aspect Term Extration:	{service, taste}
Opinion Term Extraction:	{Good, poor}
Aspect Sentiment Pair Extraction:	{(service, pos), (taste, neg)}
Aspect Opinion Co-Extraction:	{service, Good, taste, poor}
Aspect Opinion Pair Extraction:	{(service, Good), (taste, poor)}
Aspect Sentiment Triplet Extraction:	{(service, Good, pos), (taste, poor, neg)}

Table 1 gives an example of different ABSA sub-tasks for the sentence 'Good service but poor taste'. This paper mainly concentrate on ASTE, which extracts triplets of all aspects in a sentence with the corresponding opinion and the sentiment polarity for each aspect simultaneously.

Although researches have been conducted in the area, ASTE still faces many challenges:

- **Complicated relations.** The corresponding relations between aspects and opinions can be one-to-one, one-to-many, many-to-one, and even many-to-many. It is hard to detect these relations accurately and unambiguously.
- **Multiple sentiment polarities.** Each sentence may contain multiple sentiment polarities, which are usually influenced by corresponding relations between aspects and opinions. Therefore, relations need to be integrated into the sentiment classification task in a proper way.
- **Unclear boundaries.** Aspects and opinions often contain multiple successive words, making their boundaries difficult to be detected.

*Corresponding author.

Proceedings of the 29th International Conference on Computational Linguistics, pages 7065–7074
October 12–17, 2022.

To address the above challenges, we propose a Multi-Task Dual-Tree Network for ASTE, namely **MTDTN**. The constituency tree and dependency tree are two parsing methods of a sentence in Natural Language Processing (NLP), and the latter has been widely used in ABSA tasks (Wang et al., 2020; Pereg et al., 2020). Although the two trees of one sentence can be transformed into each other, it may require hops over the structure in graph neural networks or self-attentions. Thus our model employs both types of trees for AOCE and ASTE, respectively. The constituency tree is applied in the co-extraction module to detect constituent boundaries, and the dependency tree is applied in the triplet extraction module to capture relations between aspects and opinions. Moreover, for the reason that different layers of BERT (Devlin et al., 2019) capture hierarchical features, with surface features in lower layers, syntactic features in middle layers and semantic features in higher layers (Jawahar et al., 2019), we employ self-attention weights of the middle layer to modify the dependency graph. The modified graph can reduce inevitable parsing errors and imply more accurate relations between words. Finally, we use a similar tagging scheme as the Grid Tagging Scheme (Wu et al., 2020a) for triplet extraction and design a Transition-Based Inference Strategy (TBIS) to transfer the boundary information from the co-extraction module to the triplet extraction module.

The contributions of our work can be summarized as follows:

- We propose a Multi-Task Dual-Tree Network for ASTE, employing a constituency tree and a modified dependency tree in two sub-tasks of AOCE and ASTE, respectively.
- We design a Transition-Based Inference Strategy that transfers the boundary information from tags of AOCE to ASTE through a transition matrix.
- We conduct extensive experiments on four popular datasets, and the results show that our model outperforms state-of-the-art models.

2 Related work

Aspect-Opinion Co-Extraction (AOCE) has been focused on in recent years, aiming to explore the interactions between Aspect Term Extraction (ATE) and Opinion Term Extraction (OTE). Initially, models have been proposed to co-extract aspects and opinions in a sentence, treating the task as a sequence labeling problem (Wang et al., 2017; Dai and Song, 2019; He et al., 2019). However, they do not consider the relations between corresponding aspects and opinions. Then (Zhao et al., 2020) define the Aspect Opinion Pair Extraction (AOPE) task and propose a span-based multi-task learning framework. (Chen et al., 2020) propose a synchronous double-channel recurrent network to obtain aspect-opinion pairs and achieve great performance. To further explore the interactions between paired terms and sentiment polarity, (Peng et al., 2020) first define the task of Aspect-Sentiment Triplet Extraction (ASTE) and propose a two-stage model to address it. Following this work, a position-aware tagging scheme (Xu et al., 2020) and a grid tagging scheme (Wu et al., 2020a) are designed to jointly extract the triplets in an end-to-end manner. (Chen et al., 2021b) further represent the semantic and syntactic relations between word pairs by a graph to enhance the vanilla grid tagging scheme. Interactions between aspect spans and opinion spans are also studied to not only consider word-to-word interactions (Xu et al., 2021). (Chen et al., 2021a) transform the triplet extraction task into a machine reading comprehension (MRC) task with well-designed queries.

3 Task Definition

Given an input sentence $X = \{x_1, x_2, \cdots, x_n\}$ of length n, we then formulate two sub-tasks as two different sequence labeling problems.

3.1 Aspect-Opinion Co-Extraction

AOCE aims to extract all aspect terms and opinion terms appearing in a sentence. We use 5 tags in $\mathcal{Y} = \{BA, IA, BO, IO, OT\}$ to label each word x_i. BA and BO denote the beginning of an aspect term or an opinion term, IA and IO denote the inside of an aspect term or an opinion term, OT denotes the outside of both kinds of terms.

3.2 Aspect-Sentiment Triplet Extraction

ASTE aims to extract triplets of all aspect terms in a sentence with the corresponding opinion term and the sentiment polarity for each aspect term simultaneously. We employ the Grid Tagging Schema (Wu et al., 2020a), which uses 6 tags in $\mathcal{G} = \{A, O, NEG, NEU, POS, N\}$ to label the relation between two words x_i and x_j. A and O denote x_i and x_j are in the same aspect term or

opinion term, NEG, NEU and POS denote x_i and x_j are separately in an aspect term and another opinion term with the corresponding sentiment polarity, N denotes x_i and x_j have no above relations.

4 Proposed Model

4.1 Model Overview

The overview of our model is shown in Figure 1. It first accepts a sentence X as the input into a shared BERT encoder, then different layers of BERT are sent to different downstream modules. For the co-extraction module, we employ the consistency tree to construct a heterogeneous graph and apply multi-layers of Graph Convolution Networks over it to generate the final representation. For the triplet extraction module, we propose a Dep-Enhanced Transformer Decoder (DETD), which receives a modified dependency graph constructed from the dependency tree to incorporate the syntactic information. Finally, a Transition-Based Inference Strategy (TBIS) is designed, transferring the boundary information from the co-extraction module to the triplet extraction module through a transition matrix.

4.2 Shared BERT Encoder

Since pre-trained models show powerful performance in Natural Language Understanding (NLU) tasks, we choose BERT (Devlin et al., 2019) as the text encoder of our model. For a given sentence X, the following representations can be generated on the pre-trained BERT:

$$H^{[1:L]} = BERT(X) \tag{1}$$

where $H^{[1:L]}$ denote hidden states of all layers of BERT and L is the max layer.

For the reason that BERT is proven to capture a rich hierarchy of linguistic information, different layers are selected for two sub-tasks:

$$\begin{aligned} H_{ce} &= H^{L} \\ H_{te} &= H^{l} \end{aligned} \tag{2}$$

where H_{ce} and H_{te} are inputs for the co-extraction module and the triplet extraction module respectively, H^L denotes hidden states of the highest BERT layer which contains more semantic information, H^l denotes hidden states of the lth BERT layer which contains more syntactic information. We assume that the co-extraction pays more attention to semantic features and the triplet extraction pays more attention to syntactic features because the latter needs to describe word-to-word relations.

4.3 Co-Extraction Module

The constituency tree is based on the formalism of context-free grammars. In this type of tree, a sentence is divided into constituents which are sub-phrases that belong to specific categories in the grammar. For instance, a verb phrase (VP) can be formed of a verb (V) and a noun phrase (NP).

For a given sentence, we employ CoreNLP to generate a constituency tree and then construct an undirected heterogeneous graph based on the tree. The graph contains $n + m$ nodes, where n leaf nodes are tokens in the sentence, and m internal nodes are constituents in the tree. There are two types of edges in the graph: self-loop edges of leaf nodes and edges between each node and its parent node in the tree. In the forward process, leaf nodes are initialized with H_{ce} and internal nodes are randomly initialized embeddings that can be updated among training.

Then we apply Graph Convolution Networks (GCN) (Kipf and Welling, 2016) over the generated graph, concatenating the representation of leaf nodes and internal nodes as the initial input:

$$H^0 = [H_{ce}; e(c)] \tag{3}$$

where c denotes the list of constituents in the tree and e denotes the lookup table of constituent embeddings.

The GCN operation can be written as:

$$h_i^{k+1} = ReLU(\sum_{j=1}^{n+m}(A_{ij}W^{k+1}h_j^k)) \tag{4}$$

where k is the number of the current layer, $A \in \mathbb{R}^{(n+m)\times(n+m)}$ denotes the adjacency matrix of the graph, $W \in \mathbb{R}^{d\times d}$ is trainable weight, d denotes the hidden size of BERT.

After K layers of GCNs, the final representation of each token is then fed into a fully-connected layer followed by a softmax layer to yield a probability distribution over $\mathcal{Y}$:

$$p_i^{ce} = softmax(W_c h_i^K + b_c) \tag{5}$$

where $W_c \in \mathbb{R}^{d\times|\mathcal{Y}|}$ and $b_c \in \mathbb{R}^{|\mathcal{Y}|}$ are trainable weight and bias.

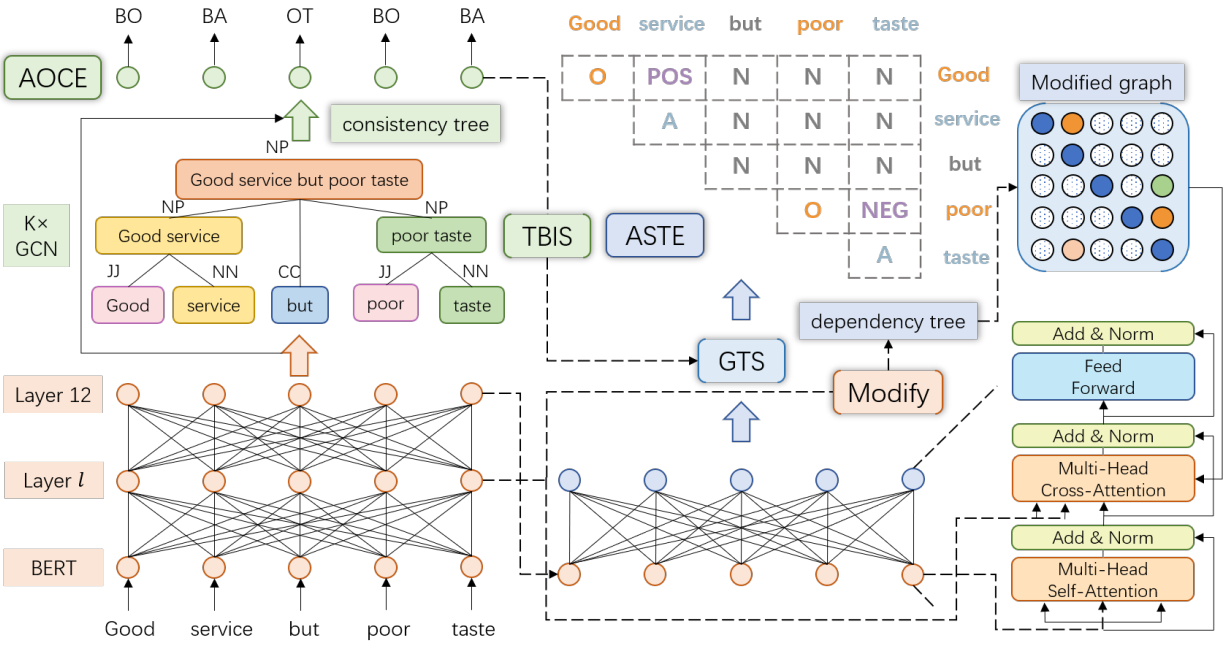

Figure 1: The overall architecture of our proposed MTDTN.

4.4 Triplet Extraction Module

The dependency tree of a sentence is a directed acyclic graph with words as nodes and relations as edges. The relation between any two words in the tree can be described as a "head-dependent" pair. For the same sentence as in the co-extraction module, we employ CoreNLP to generate a dependency tree and then construct an undirected isomorphic dependency graph based on the tree without relations .

The dependency graph generated by tools may have inevitable parsing errors. Different from (Xiao et al., 2021) which employs self-attention weights of layers all over BERT to supply the dependency graph, we only make use of the middle layer which contains more syntactic information to modify it:

$$A^{att} = softmax(\frac{Q_{att}W^{Q}_{att}(K_{att}W^{K}_{att})^{T}}{\sqrt{d}})$$

$$A^{modi}_{i,j} = \begin{cases} 1, & \alpha \leq A^{att}_{i,j} \\ A^{dep}_{i,j}, & \beta < A^{att}_{i,j} < \alpha \\ 0, & A^{att}_{i,j} \leq \beta \end{cases} \quad (6)$$

where Q_{att} and K_{att} are both equal to H_{te}, W^{Q}_{att} and W^{K}_{att} denote trainable weights, $A^{dep} \in \mathbb{R}^{n \times n}$ and $A^{modi} \in \mathbb{R}^{n \times n}$ denote adjacency matrices of the original dependency graph and the modified graph respectively, α and β are hyper-parameters.

In order to receive the modified dependency graph, we design a Dep-Enhanced Transformer Decoder (DETD) as shown in Figure 2, which uses H_{ce} as input to the first sub-layer and H_{te} as key and value to the second sub-layer:

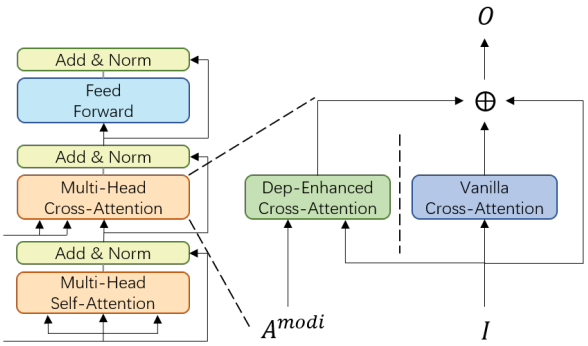

Figure 2: Dep-Enhanced Transformer Decoder.

$$T = \text{DETD}(H_{te}, H_{ce}, A^{modi}) \quad (7)$$

where $T = \{t_1, t_2, \cdots, t_n\}$ denotes the output of DETD.

Unlike the vanilla transformer decoder, we use multi-head attention instead of masked multi-head attention in the first sub-layer. Since the vanilla transformer does not explicitly encode syntactic features, in the second sub-layer, we incorporate the modified dependency graph into multi-head cross-attention by changing the calculation method

of the attention coefficients:

$$
\begin{aligned}
A =& softmax(\frac{(QW^Q(KW^K)^T) * A^{modi}}{\sqrt{d}}) \\
CA_{dep} =& FF(AV) \\
O =& LN(I + CA(I) + CA_{dep}(I))
\end{aligned} \tag{8}
$$

where I denotes the input of and O denotes the output of multi-head cross-attention, Q is equal to I, K and V are both equal to H_{te}, W^Q and W^K denote trainable weights, $*$ denotes an element-wise multiplication between A^{modi} and the dot product of Q and K, FF and LN are feed-forward network and layer normalization in transformer, CA and CA_{dep} denote the vanilla and the dep-enhanced cross-attention respectively.

Finally, we concatenate the DETD representations of word x_i and x_j to represent the word-pair (x_i, x_j), i.e., $r_{ij} = [t_i; t_j]$, where $[;]$ is the concatenation operation. Then r_{ij} is sent to a fully-connected layer to calculate the temporary triplet tag probability:

$$
z_{ij} = W^s r_{ij} + b^s \tag{9}
$$

where $W_s \in \mathbb{R}^{d \times |\mathcal{G}|}$ and $b_s \in \mathbb{R}^{|\mathcal{G}|}$ are trainable weight and bias.

4.5 Transition-Based Inference Strategy

The inference strategy of the original Grid Tagging Schema (GTS) (Wu et al., 2020a) requires indefinite iterations to capture word-to-word relations, which will increase the time complexity. Inspired by boundary guidance in E2E-ABSA (Li et al., 2019), we further propose a Transition-Based Inference Strategy (TBIS) to accelerate the convergence.

Firstly, we use a similar approach to the original GTS, leveraging features of distributions of the temporary triplet tag probability and capturing the associated features between x_i/x_j and others to obtain more accurate results. The new probability q_{ij} can be calculated as follows:

$$
\begin{aligned}
z_i &= maxpooling(z_{i,:}) \\
z_j &= maxpooling(z_{j,:}) \\
\widetilde{r}_{ij} &= [r_{ij}; z_i; z_j; z_{ij}] \\
o_{ij} &= W^o \widetilde{r}_{ij} + b^o \\
q_{ij} &= W^s o_{ij} + b^s
\end{aligned} \tag{10}
$$

where $z_{i,:} = (z_{1:i,i}, z_{i,i:n})$ according to the upper triangular grid in GTS, W^o and b^o are trainable weight and bias, W^s and b^s share the same parameters as above.

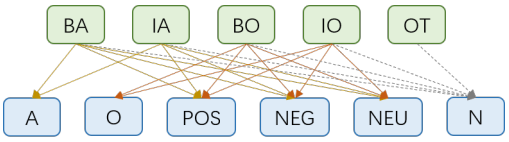

Figure 3: Constraints between co-extraction tags and triplet extraction tags.

Secondly, we encode the constraints between co-extraction tags and triplet extraction tags into a transition matrix $W^g \in \mathbb{R}^{|\mathcal{Y}| \times |\mathcal{G}|}$ as shown in Figure 3. The matrix is initialized as follows and updated as a linear layer during training:

$$
W^g_{ij} = \begin{cases} \frac{1}{|\mathcal{T}_i|} & \text{if} \quad j \in \mathcal{T}_i \\ 0, & \text{Otherwise} \end{cases} \tag{11}
$$

where $\mathcal{T}_i$ is the set of valid triplet extraction tags in $\mathcal{G}$ corresponding to the ith co-extraction tag in $\mathcal{Y}$. We transfer boundary information of aspects and opinions by mapping the probability scores of the co-extraction tag space to the triplet tag space. The transition-based score of x_i is calculated as follows:

$$
g_i = (W^g)^T p^{ce}_i \tag{12}
$$

A gating mechanism is applied to fuse the transition-based score with the triplet extraction tag probability. We calculate a gating score $\alpha_i \in \mathbb{R}$ based on the confidence score c_i:

$$
\begin{aligned}
c_i &= (p^{ce}_i)^T p^{ce}_i \\
\alpha_i &= \epsilon c_i
\end{aligned} \tag{13}
$$

where c_i represents co-extraction module's confidence in the predicted result p^{ce}_i, ϵ is a hyper-parameter to control the contribution of the transition-based score g_i in the final result.

Finally, q_{ij} is fused with g_i by the gating score α_i and the result is fed into a softmax layer to yield a probability distribution over $\mathcal{G}$:

$$
p^{te}_{ij} = softmax(\alpha_i g_i + (1 - \alpha_i) q_{ij}) \tag{14}
$$

4.6 Joint Training Loss

Training losses for two sub-tasks are both defined as cross-entropy loss:

$$
\mathcal{L}_{ce} = -\sum_{i=1}^{n} \sum_{k \in \mathcal{Y}} I(y^{ce}_i = k) log(p^{ce}_{i|k}) \tag{15}
$$

$$\mathcal{L}_{te} = -\sum_{i=1}^{n}\sum_{j=i}^{n}\sum_{k\in\mathcal{G}} I(y_{ij}^{te}=k)log(p_{ij|k}^{te}) \quad (16)$$

where y_i^{ce} denotes the ground truth tag of word x_i in co-extraction, y_{ij}^{te} denotes the ground truth tag of the relation between word x_i and x_j in triplet extraction, p_i^{ce} and $p_{i,j}^{te}$ denote predicted tagging distributions, $I(\cdot)$ is the indicator function, $\mathcal{Y}$ and $\mathcal{G}$ denote two label sets.

To jointly train two sub-tasks and make them mutually beneficial, we combine the above loss functions to form the final objective, where the hyper-parameter γ denotes their ratio.

$$\mathcal{L} = \gamma\mathcal{L}_{ce} + \mathcal{L}_{te} \quad (17)$$

5 Experiments

5.1 Datasets and Metrics

Experiments are conducted on datasets created by Wu (Wu et al., 2020a). There are four datasets, among which 14res, 15res and 16res are in the restaurant domain, and 14lap is in the laptop domain. The statistics of all datasets are listed in Table 2.

Following previous works, we employ precision, recall and micro F1-score as metrics. During training, we use the model that performed best on the development set for testing. For reproducibility, on each dataset we train the model 5 times with different random seeds and report the averaged results.

Table 2: Statistics of datasets (#S, #P, #neg, #neu, and #pos denote the number of sentences, pairs, negative triplets, neutral triplets, and positive triplets, respectively.)

Datasets		#S	#P	#neg	#neu	#pos
14res	Train	1,259	2,356	491	172	1693
	Dev	315	580	107	46	427
	Test	493	1008	156	427	784
14lap	Train	899	1452	533	111	808
	Dev	225	383	136	48	199
	Test	332	547	116	67	364
15res	Train	603	1038	210	29	799
	Dev	151	239	49	9	181
	Test	325	493	144	25	324
16res	Train	863	1421	330	55	1036
	Dev	216	348	77	8	263
	Test	328	525	79	30	416

5.2 Baselines

We compare our model with the following baselines to evaluate the performance of MTDTN, where part of them are pipeline models and others are end-to-end models.

- **Peng-unified-R+PD** (Peng et al., 2020) propose a two-stage framework to address the ASTE task. In the first stage, Peng-unified-R extracts aspects with sentiment and opinions by utilizing mutual influence between aspects and opinions. In the second stage, an MLP-based classifier (PD) is applied to all possible triplets to determine whether each triplet is valid or not.

- **Li-unified-R+PD** is a pipeline model combined by (Peng et al., 2020), which first employs a modified model Li-unified-R (Li et al., 2019) to extract aspects with sentiment and opinions and then applies PD to obtain all the valid triplets.

- **Peng-unified-R+IOG** is a pipeline model combined by (Wu et al., 2020a), which first uses Peng-unified-R (Peng et al., 2020) to extract aspects with sentiment and then employ IOG (Fan et al., 2019) to generate triplets. IOG can effectively encode aspect information to extract the corresponding opinion.

- **IMN+IOG** is another pipeline model combined by (Wu et al., 2020a), which first uses IMN (He et al., 2019) to extract aspects with sentiment and then employ IOG (Fan et al., 2019) to generate triplets.

- **GTS** (Wu et al., 2020a) propose a unified grid tagging scheme to address the ASTE task and design an inference strategy to exploit mutual indications between different opinion factors.

- **S^3E^2** (Chen et al., 2021b) further represent the semantic and syntactic relations between word pairs by a graph neural network to enhance the vanilla GTS.

- **BMRC** (Chen et al., 2021a) convert the ASTE task into a multi-turn machine reading comprehension (MRC) task with well-designed queries.

Table 3: Main results of triplet extraction (%). All methods' results are from original papers or the paper of GTS. The mark '-' means that the paper of BMRC does not release the precision and the recall on each dataset.

Methods	14res			14lap			15res			16res		
	P	R	F1	P	R	F1	P	R	F1	P	R	F1
Li-unified-R+PD	41.44	68.79	51.68	42.25	42.78	42.47	43.34	50.73	46.69	38.19	53.47	44.51
Peng-unified-R+PD	44.18	62.99	51.89	40.40	47.24	43.50	40.97	54.68	46.79	46.76	62.97	53.62
Peng-unified-R+IOG	58.89	60.41	59.64	48.62	45.52	47.02	51.70	46.04	48.71	59.25	58.09	58.67
IMN+IOG	59.57	63.88	61.65	49.21	46.23	47.68	55.24	52.33	53.75	-	-	-
GTS-CNN	70.79	61.71	65.94	55.93	47.52	51.38	60.09	53.57	56.64	62.63	66.98	64.73
GTS-BiLSTM	67.28	61.91	64.49	59.42	45.13	51.30	63.26	50.71	56.29	66.07	65.05	65.56
S3E2	69.08	64.55	66.74	59.43	46.23	52.01	61.06	56.44	58.66	71.08	63.13	66.87
GTS-BERT	**70.92**	69.49	70.20	57.52	51.92	54.58	**59.29**	58.07	58.67	68.58	66.60	67.58
BMRC-BERT	-	-	70.01	-	-	57.83	-	-	58.74	-	-	67.49
Ours	70.00	**71.78**	**70.88**	**61.98**	**54.71**	**58.12**	59.03	**62.68**	**60.80**	**69.04**	**69.98**	**69.51**

5.3 Implementation Details

For the shared BERT encoder, we choose the uncased version of **BERT-base** (Devlin et al., 2019) with 12 stacked Transformer blocks, 12 attention heads and the hidden size of 768, which is implemented based on HuggingFace's **Transformers** (Wolf et al., 2020) library. While training the joint model, we employ AdamW (Loshchilov and Hutter, 2018) as the optimizer with the weight decay of 0.01 and the warmup rate of 0.1. The learning rate is set to 2e-5 for the BERT parameter group and 1e-3 for other parameter groups. The batch size is set to 32 with a max sequence length of 128. When constructing graphs for constituency trees and dependency trees, we only keep edges associated with the first sub-word of each word tokenized by BERT. We set the middle layer l to 8 for 14res, 14lap and 9 for 15res, 16res respectively. We set the thresholds α and β to 0.1 and 0.9 to generate the modified graph. For the joint training loss, the ratio γ is set to 1. All experiments are conducted on two Nvidia RTX 3080 GPUs.

5.4 Main Results

The main results of baselines and our MTDTN model are shown in Table 3. According to the results, MTDTN outperforms all baselines and achieves state-of-the-art performances on four popular datasets, which proving our model's effectiveness.

In general, due to the strong text expression ability of pre-trained models, the BERT-based models like GTS-BERT, BMRC-BERT and MTDTN surpass other models which do not employ BERT as the text encoder layer significantly.

More importantly, MTDTN achieves 0.68%, 3.54%, 2.13% and 1.93% absolute F1 scores gain over GTS-BERT, which is the state-of-the-art method we followed on four datasets. We think it is because our model can accurately locate aspects and opinions and capture the relation between them by introducing syntactic information and internal interaction of multiple tasks.

Then, compared with BMRC-BERT, which is another state-of-the-art model, MTDTN achieves an absolute F1 score increase of 0.87%, 0.29%, 2.06%, 2.02% on four datasets. BMRC-BERT converts the ASTE task into a machine reading comprehension task, while the restrictive query may not correctly capture the relation between aspects and opinions. This may be the actual cause of the performance difference.

5.5 Ablation Study

To verify the validity of different modules in our MTDTN, we further carry out an ablation study as shown in Table 4.

Table 4: Results of ablation study for ASTE task (F1%).

Methods	14res	14lap	15res	16res
MTDTN	**70.88**	**58.12**	**60.80**	**69.51**
MTDTN w/o CE	70.13	56.98	58.61	67.28
MTDTN w/o TBIS	70.33	57.39	59.51	67.69
MTDTN w/o DETD	68.81	54.56	58.11	67.03
MTDTN w/o MG	69.80	56.32	59.83	68.94

Firstly, we verify the effectiveness of the multi-task framework by removing the co-extraction module, which is 'MTDTN w/o CE' in the table. It can be observed that there is a certain decline in performance, which shows that the auxiliary task is fully effective in extracting triplets from sentences.

After that, we replace TBIS with the original inference strategy in GTS, which refers to 'MTDTN

w/o TBIS'. We can see the performance drops, which shows that the designed inference strategy can utilize the boundary information of aspects and opinions in the AOCE task to promote the ASTE task.

Then, we remove DETD and replace it with a vanilla transformer decoder without masked multi-head attention. It can be seen from 'MTDTN w/o DETD' that the model's performance is significantly declined, showing that the syntactic information can help the model better capture the relations between words.

More detailed, 'MTDTN w/o MG' means directly replacing the modified graph with the dependency graph generated by the parser. The dropping in performance shows that the modified graph can reduce parsing errors and is more suitable for the specific task.

5.6 Analysis of Graph Type

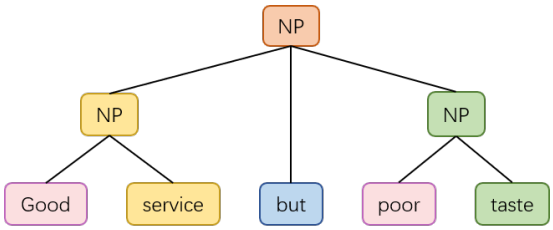

(a) Heterogeneous graph with nodes of constituent and word.

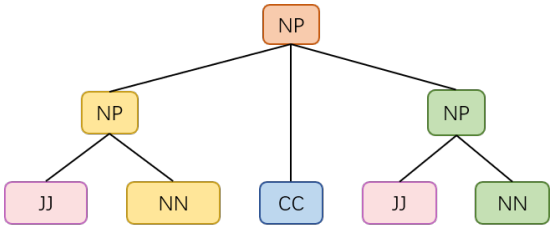

(b) Isomorphic graph with nodes of constituent.

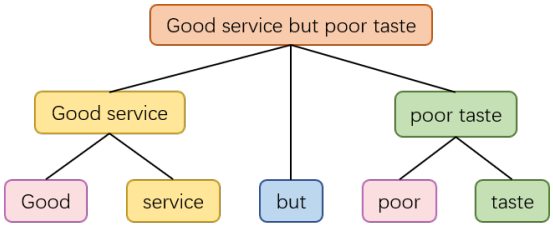

(c) Isomorphic graph with nodes of word.

Figure 4: Different types of graph.

In this section, we compare three graphs as in Figure 4 on four datasets, the results are shown in Table 5, where 'POS sequence' denotes simple addition of POS embeddings to H_{ce}, 'None' denotes H_{ce} directly being sent to the tag decoder. In 'Isomorphic graph (constituent)' and 'POS sequence', embeddings of constituents and POS are randomly initialized and updated during training. In 'Isomorphic graph (word)', embeddings of phrases are calculated from the average of word embeddings it contains.

Table 5: Results of graph analysis for AOCE task (F1%).

Methods	14res	14lap	15res	16res
Heterogeneous graph	**87.37**	**86.07**	**81.67**	81.79
Isomorphic graph (constituent)	86.22	85.07	81.39	81.42
Isomorphic graph (word)	85.06	84.23	80.28	82.15
POS sequence	85.14	83.72	80.12	**82.69**
None	84.77	84.02	79.22	82.44

We observe that the model using a heterogeneous graph obtains better AOCE performance than other methods on all datasets except 16res. This may be because the fact that the node interaction of heterogeneous graphs is more explainable compared to isomorphic graphs. On the one hand, embeddings of constituents can obtain information from the fully trained hidden states of words. On the other hand, the word representation can also get boundary information and constituent information from the graph of the constituency tree.

5.7 Analysis of BERT Layer

To investigate the effect of different BERT layers modifying the dependency graph and being the key and value of DETD, we evaluate our MTDTN model with each layer of BERT on four datasets. As shown in Figure 5, MTDTN with the 8th or 9th layer of BERT performs the best. The results are

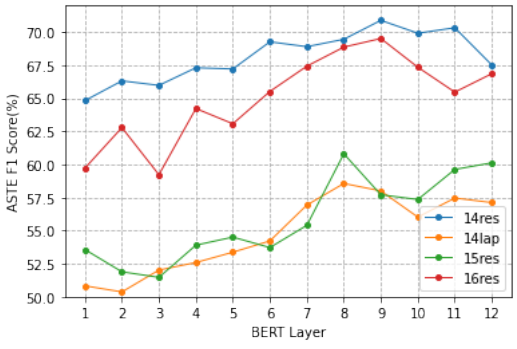

Figure 5: Impact of different BERT layers

consistent with the hierarchical characteristics of BERT (Jawahar et al., 2019) that middle layers capture rich syntactic features. Therefore, employing self-attention weights of the middle layer to modify the dependency graph can reduce parsing errors and make it suit the specific task better.

6 Conclusions

In this paper, we propose a multi-task framework for Aspect Sentiment Triplet Extraction (ASTE) with Aspect Opinion Co-Extraction (AOCE) as an auxiliary task. The two sub-tasks utilize two types of trees to capture different information of the text. For a given sentence, a constituency tree is employed by a graph convolution network for AOCE, and a modified dependency tree is employed by a special transformer decoder for ASTE. We further designed a Transition-Based Inference Strategy (TBIS) to enhance information interaction between sub-tasks by transferring the boundary information from AOCE to ASTE through a transition matrix. The whole model is called Multi-Task Dual-Tree Network (MTDTN) and extensive experiments demonstrate that our model achieves state-of-the-art performance on four popular datasets.

Acknowledge

This research work has been sponsored by Ant Group Security and Risk Management Fund, and the Joint Funds of the National Natural Science Foundation of China (Grant No. U21B2020).

References

Shaowei Chen, Jie Liu, Yu Wang, Wenzheng Zhang, and Ziming Chi. 2020. Synchronous double-channel recurrent network for aspect-opinion pair extraction. In *Proc. of ACL*.

Shaowei Chen, Yu Wang, Jie Liu, and Yuelin Wang. 2021a. Bidirectional machine reading comprehension for aspect sentiment triplet extraction. In *Proc. of AAAI*.

Zhexue Chen, Hong Huang, Bang Liu, Xuanhua Shi, and Hai Jin. 2021b. Semantic and syntactic enhanced aspect sentiment triplet extraction. In *Proc. of ACL*.

Zhuang Chen and Tieyun Qian. 2020. Relation-aware collaborative learning for unified aspect-based sentiment analysis. In *Proc. of ACL*.

Hongliang Dai and Yangqiu Song. 2019. Neural aspect and opinion term extraction with mined rules as weak supervision. In *Proc. of ACL*.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. Bert: Pre-training of deep bidirectional transformers for language understanding. In *Proc. of NAACL*.

Zhifang Fan, Zhen Wu, Xinyu Dai, Shujian Huang, and Jiajun Chen. 2019. Target-oriented opinion words extraction with target-fused neural sequence labeling. In *Proc. of NAACL*.

Ruidan He, Wee Sun Lee, Hwee Tou Ng, and Daniel Dahlmeier. 2019. An interactive multi-task learning network for end-to-end aspect-based sentiment analysis. In *Proc. of ACL*.

Ganesh Jawahar, Benoît Sagot, and Djamé Seddah. 2019. What does bert learn about the structure of language? In *Proc. of ACL*.

Qian Ji, Xiang Lin, Yinghua Ma, Gongshen Liu, and Shilin Wang. 2020. A unified labeling model for open-domain aspect-based sentiment analysis. In *Proc. of DSC*.

Thomas N Kipf and Max Welling. 2016. Semi-supervised classification with graph convolutional networks. *arXiv preprint arXiv:1609.02907*.

Ruifan Li, Hao Chen, Fangxiang Feng, Zhanyu Ma, Xiaojie Wang, and Eduard Hovy. 2021. Dual graph convolutional networks for aspect-based sentiment analysis. In *Proc. of ACL*.

Xin Li, Lidong Bing, Piji Li, and Wai Lam. 2019. A unified model for opinion target extraction and target sentiment prediction. In *Proc. of AAAI*.

Ilya Loshchilov and Frank Hutter. 2018. Fixing weight decay regularization in adam. *arXiv preprint arXiv:1711.05101*.

Huaishao Luo, Lei Ji, Tianrui Li, Daxin Jiang, and Nan Duan. 2020. Grace: Gradient harmonized and cascaded labeling for aspect-based sentiment analysis. In *Proc. of EMNLP*.

Haiyun Peng, Lu Xu, Lidong Bing, Fei Huang, Wei Lu, and Luo Si. 2020. Knowing what, how and why: A near complete solution for aspect-based sentiment analysis. In *Proc. of AAAI*.

Oren Pereg, Daniel Korat, and Moshe Wasserblat. 2020. Syntactically aware cross-domain aspect and opinion terms extraction. In *Proc. of COLING*.

Kai Wang, Weizhou Shen, Yunyi Yang, Xiaojun Quan, and Rui Wang. 2020. Relational graph attention network for aspect-based sentiment analysis. In *Proc. of ACL*.

Wenya Wang, Sinno Jialin Pan, Daniel Dahlmeier, and Xiaokui Xiao. 2017. Coupled multi-layer attentions for co-extraction of aspect and opinion terms. In *Proc. of AAAI*.

Thomas Wolf, Julien Chaumond, Lysandre Debut, Victor Sanh, Clement Delangue, Anthony Moi, Pierric Cistac, Morgan Funtowicz, Joe Davison, Sam Shleifer, et al. 2020. Transformers: State-of-the-art natural language processing. In *Proc. of EMNLP*.

Zhen Wu, Chengcan Ying, Fei Zhao, Zhifang Fan, Xinyu Dai, and Rui Xia. 2020a. Grid tagging scheme for end-to-end fine-grained opinion extraction. In *Proc. of EMNLP*.

Zhen Wu, Fei Zhao, Xin-Yu Dai, Shujian Huang, and Jiajun Chen. 2020b. Latent opinions transfer network for target-oriented opinion words extraction. In *Proc. of AAAI*.

Zeguan Xiao, Jiarun Wu, Qingliang Chen, and Congjian Deng. 2021. Bert4gcn: Using bert intermediate layers to augment gcn for aspect-based sentiment classification. In *Proc. of EMNLP*.

Hu Xu, Bing Liu, Lei Shu, and S Yu Philip. 2018. Double embeddings and cnn-based sequence labeling for aspect extraction. In *Proc. of ACL*.

Lu Xu, Yew Ken Chia, and Lidong Bing. 2021. Learning span-level interactions for aspect sentiment triplet extraction. In *Proc. of ACL*.

Lu Xu, Hao Li, Wei Lu, and Lidong Bing. 2020. Position-aware tagging for aspect sentiment triplet extraction. In *Proc. of EMNLP*.

Yunyi Yang, Kun Li, Xiaojun Quan, Weizhou Shen, and Qinliang Su. 2020. Constituency lattice encoding for aspect term extraction. In *Proc. of COLING*.

He Zhao, Longtao Huang, Rong Zhang, Quan Lu, and Hui Xue. 2020. Spanmlt: A span-based multi-task learning framework for pair-wise aspect and opinion terms extraction. In *Proc. of ACL*.

Exploiting Unlabeled Data for Target-Oriented Opinion Words Extraction

Yidong Wang[1 *] **Hao Wu**[1 *] **Ao Liu**[1] **Wenxin Hou**[2]
Zhen Wu[3 †] **Jindong Wang**[4] **Takahiro Shinozaki**[1] **Manabu Okumura**[1] **Yue Zhang**[5]
[1]Tokyo Institute of Technology [2]Microsoft STCA
[3]Nanjing University [4]Microsoft Research Asia [5]Westlake University
{yidongwang37, wu.364371691}@gmail.com, wuz@nju.edu.cn

Abstract

Target-oriented Opinion Words Extraction (TOWE) is a fine-grained sentiment analysis task that aims to extract the corresponding opinion words of a given opinion target from the sentence. Recently, deep learning approaches have made remarkable progress on this task. Nevertheless, the TOWE task still suffers from the scarcity of training data due to the expensive data annotation process. Limited labeled data increase the risk of distribution shift between test data and training data. In this paper, we propose exploiting massive unlabeled data to reduce the risk by increasing the exposure of the model to varying distribution shifts. Specifically, we propose a novel Multi-Grained Consistency Regularization (MGCR) method to make use of unlabeled data and design two filters specifically for TOWE to filter noisy data at different granularity. Extensive experimental results on four TOWE benchmark datasets indicate the superiority of MGCR compared with current state-of-the-art methods. The in-depth analysis also demonstrates the effectiveness of the different-granularity filters. Our codes are available at https://github.com/TOWESSL/TOWESSL.

1 Introduction

Target-oriented Opinion Words Extraction (TOWE) (Fan et al., 2019) is an important subtask of aspect-based sentiment analysis (ABSA) (Pontiki et al., 2014), which aims to extract the corresponding opinion words for a given opinion target from the sentence. For the TOWE task, opinion targets, also called aspect terms, are the entities or objects in the sentence toward which users show attitudes; opinion words, sometimes known as opinion expressions, are those words explicitly mentioned in the sentence and used to express attitudes or opinions. Figure 1 shows an example of the TOWE task. For the sentence "*The dishes are amazingly delicious but the waiter is so rude.*", the terms "*dishes*" and "*waiter*" are two opinion targets. The goal of TOWE is to extract "*amazingly delicious*" as the opinion words for the opinion target "*dishes*" and opinion word "*rude*" when given the opinion target "*waiter*".

Figure 1: Example of TOWE. Words in red are opinion targets and words in blue are corresponding opinion words. TOWE extracts corresponding opinion words when given opinion targets.

While seminal work casts TOWE as a sequence labeling problem, using fully supervised learning methods to identify opinion words and phrases from sentences, recent work shows that external sources of information can be highly useful for improving the performance. In particular, both syntactic knowledge (Dai et al., 2022; Veyseh et al., 2020; Jiang et al., 2021; Zhang et al., 2021b) and sentiment information (Wu et al., 2020b) have been exploited, with the former helping to identify the correlation between opinion targets and opinion expressions, and the latter helping to identify words and phrases that are correlated with sentiment polarities. Existing work integrates these external features via representation structures (Dai et al., 2022; Veyseh et al., 2020; Jiang et al., 2021) and multi-task learning (Wu et al., 2020b; Zhang et al., 2021b).

Intuitively, the goal of opinion words extraction is to obtain structured knowledge from raw data, and therefore ideally the amount of test data, namely the data from which opinion words are mined, should be large. This can inevitably increase the risk of distribution shift between test instances and the training data, even if the test data

* Equal Contribution.
† Corresponding author.

Proceedings of the 29th International Conference on Computational Linguistics, pages 7075–7085
October 12–17, 2022.

is from the same domain (Ganin et al., 2016). Existing work, however, adopt a supervised training setting, with the model being tuned on a set of fixed training data. To address this issue, we consider making use of raw text inputs to increase the exposure of the model to varying distribution shifts. Our main idea is to (1) find raw data in a similar domain as the target-oriented opinion words extraction training data (i.e. gold data), (2) using a model trained on the gold data to assign silver labels, and then (3) design a set of filters to select the most useful set of silver data, so that (4) the selected silver data can be used together with the gold data for training a final model.

As a pre-processing step, we train a opinion target extraction model, which is used to label the set of raw data. Those raw sentences without any opinion target are filtered. The resulting raw data has the same format as the input structure of TOWE data. Then in our implementation, steps (2) to (4) above are done in a joint batch training process. First, we initialize a TOWE model using BERT. Then in each batch, we randomly sample a subset of gold data and a subset of raw data, using the current model to assign pseudo labels to the raw data. According to the current model probabilities and an external sentiment classifier, we filter the silver data by removing low-confidence sentences, as well as masking low-confidence words from the remaining sentences. Finally, the model parameters are updated by using the standard cross-entropy losses separately on both the gold and silver data, so that the next batch of training can start with the new model. The training process continues for a fixed number of iterations, and the model with the highest development scores are selected for testing.

Results on four standard benchmarks show that our method gives significantly improved results when raw data are used. We achieve the best reported results on all the datasets. In addition, the critical ablation demonstrates our sentence-level and word-level noises filtering both bring improvements for the final results. In-depth analysis shows our method significantly reduces the extraction errors on different error types. To our knowledge, we are the first to consider the use of unlabeled data for TOWE, successfully giving state-of-the-art results on benchmarks. Our code and datasets are available at https://github.com/TOWESSL/TOWESSL.

Labeled data:
The/O entire/O dining/O experience/O was/O very/B wonderful/I !/O
Unlabeled data with the generated pseudo opinion target:
Their menu is too expensive for a bubble drink .

Figure 2: Examples of labeled data and unlabeled data with the pseudo opinion target. Words with underline indicate opinion targets. The span in the labeled data beginning with B and followed by I represent the corresponding opinion words.

2 Background

2.1 Task Formalization

The TOWE task requires the opinion target as the input and extracts the target-oriented opinion words from the sentence. It can be formulated as a target-oriented sequence labeling task. In this work, we use the notation s to represent a labeled sentence from the TOWE dataset, and use s^u to denote a unlabeled sentence. Formally, given an input sentence $s = \{w_1, w_2, \ldots, w_n\}$ consisting of n words and an opinion target w_t in s (here we notate an opinion target as one word for simplicity and t is the position of the opinion target in the sentence), the goal of TOWE is assign a corresponding label $y_i \in \{B, I, O\}$(B: **B**eginning, I: **I**nside, O: **O**thers) for each word w_i in s. The spans beginning with B and followed by I represent the corresponding opinion words of the opinion target w_t. Figure 2 shows an sequence labeling example of TOWE.

2.2 Pseudo Opinion Targets Labeling

Raw unlabeled data can not be directly used in the semi-supervised scenario for TOWE, as they lack the necessary annotations of opinion targets. We use the opinion targets in labeled TOWE data as the ground truth to train a BERT-based target extraction model.

Specifically, given an input sentence $s = \{w_1, w_2, \ldots, w_n\}$ from the TOWE dataset, we employ a BERT model (Devlin et al., 2019) to generate the context representation of the each word w_i as follows:

$$\mathbf{h}_1^{pt}, \ldots, \mathbf{h}_n^{pt} = \text{BERT}(w_1, \ldots, w_n) \quad (1)$$

Then the context representation $\mathbf{h}_i^{pt}$ of the word w_i is fed to a linear layer and a softmax layer to predict the corresponding label. Similar to TOWE, the BIO scheme is used. We train the opinion target extraction model by minimizing the cross-entropy loss between the predicted BIO label distribution

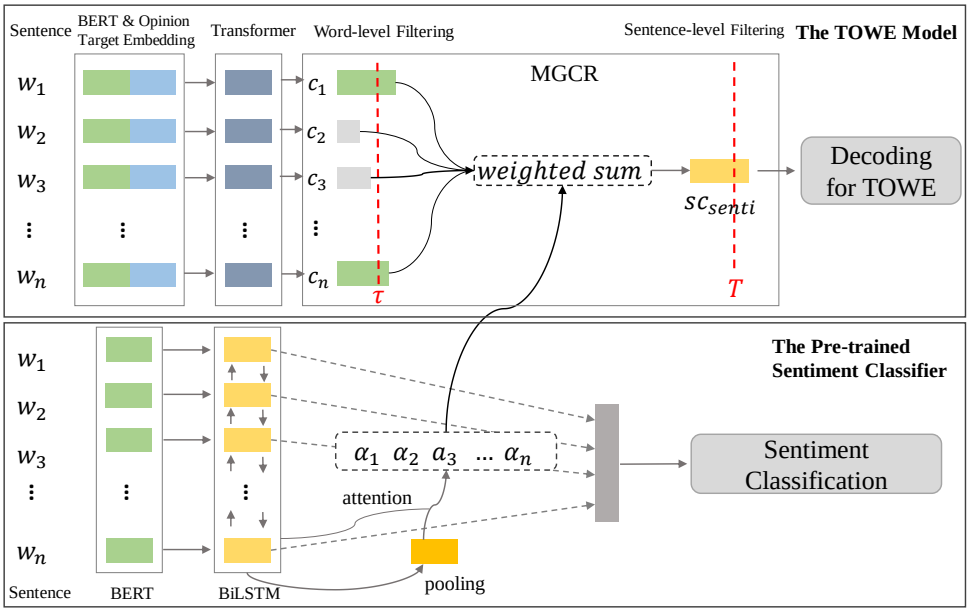

Figure 3: Overview of the architecture of multi-grained consistency regularization. For simplicity, we mark the confidence of i-th word as c_i. Note that the input sentence of the TOWE model is the same as the input sentence of the pre-trained sentiment classifier.

and the ground truth. After training, we can employ the opinion target extraction model to obtain the pseudo opinion targets of the unlabeled sentence s^u, which makes semi-supervised learning feasible for the TOWE task.

2.3 Consistency Regularization

Consistency regularization is a semi-supervised learning method that shows effectiveness in a variety of fields (Sohn et al., 2020; Chen et al., 2020; Xie et al., 2020; Zhang et al., 2021a). It improves the generalization performance of a TOWE model by generating a perturbed version $\omega(s^u)$ of the original unlabeled sentence s^u and forcing the predicted category of each word to be the same, where ω is the perturbing function (Lee et al., 2013).

To perform semi-supervised learning for TOWE, we feed the unlabeled sentence s^u and the position t of the pseudo opinion target w_t^u to the TOWE model (see section 3.1), and obtain the predicted probability $p_i(y|\theta; s^u, t)$ of the word w_i^u, where θ represents the parameters of the TOWE model. A vanilla consistency loss for consistency regularization of TOWE is computed as:

$$\frac{1}{n}\sum_{i=1}^{n}\mathcal{H}(\hat{p}_i(y|\theta; s^u, t), p_i(y|\theta; \omega(s^u), t)), \quad (2)$$

where $\hat{p}_i(y|\theta; s^u, t) = \arg\max p_i(y|\theta; s^u, t)$ and $\hat{p}_i(y|\theta; s^u, t)$ denotes the predicted label of the w_i^u, the $\mathcal{H}(\cdot,\cdot)$ refers to the cross-entropy loss. In this work, we use Random Mask and Random Synonym Replacement by using WordNet (Miller, 1995) as the perturbing function ω.

3 Method

Figure 3 shows the the framework of our *Multi-Grained Consistency Regularization (MGCR)* method. The TOWE model is a BERT-based neural sequence labeling network. As mentioned in the introduction, we initialize a BERT TOWE model (Section 3.1), and then iteratively conduct batch training. In each batch, we impose a standard cross-entropy loss on a set of sampled gold data (Eq 5), and a regularization loss on a set of randomly sampled raw data with targets (Eq 11). The latter is defined by using the current model to assign TOWE labels, and then adding a sentence-level (Section 9) and a word-level (Section 10) filter. Besides, as mentioned in the introduction, MGCR exploits latent opinion words from a pre-trained review sentiment classification (Wu et al., 2020b) to filter noises more accurately.

3.1 TOWE Model

For the sentence $s = \{w_1, w_2, \ldots, w_n\}$, the TOWE model first employs a pre-trained BERT to generate the context representations $\{\mathbf{h}_1, \ldots, \mathbf{h}_n\}$ of s. To incorporate the opinion target information

into the sentence representations, we use the number 1 to represent the position of opinion target w_t in s and 0 to denote the positions of other words, and then map them to the position embedding sequence $\{\mathbf{e}_1, \ldots, \mathbf{e}_n\}$. We integrate opinion target information by concatenating, i.e., $\widetilde{\mathbf{h}}_i = [\mathbf{h}_i; \mathbf{e}_i]$.

To better encode opinion target information, we additionally employ a multi-layer Transformer architecture (Vaswani et al., 2017) to generate the target-specific context representations as follows:

$$\mathbf{r}_1, \ldots, \mathbf{r}_n = \text{Transformer}(\widetilde{\mathbf{h}}_1, \ldots, \widetilde{\mathbf{h}}_n). \quad (3)$$

Finally, we use the representation $\mathbf{r}_i$ to predict the opinion word probability $p_i(y|\theta; s, t)$ of the word w_i when given the opinion target w_t:

$$p_i(y|\theta; s, t) = \text{softmax}(\mathbf{W}_r \mathbf{r}_i + \mathbf{b}_r), \quad (4)$$

where $\mathbf{W}_r$ and $\mathbf{b}_r$ are learnable weight and bias. The loss of supervised learning on the TOWE dataset is defined as:

$$\mathcal{L}_s = \frac{1}{n} \sum_{i=1}^{n} \mathcal{H}(y_i, p_i(y|\theta; s, t)). \quad (5)$$

3.2 Multi-Grained Consistency Regularization

3.2.1 Sentence-level Filtering

Given an unlabeled input sentence $s^u = \{w_1^u, w_2^u, \ldots, w_n^u\}$ with w_t^u as the pseudo opinion target position, the sentence-level confidence sc_{avg} is defined as follows:

$$sc_{avg} = \frac{1}{n} \sum_{i=1}^{n} \max(p_i(y|\theta; s^u, t)), \quad (6)$$

where $\max(p_i(y|\theta; s^u, t))$ (i.e., the maximum of the probabilities of B, I, and O) is the confidence of the i-th word. Sentences with confidences below the given threshold T are masked during training. In Eq. (6), different words in the sentence are treated equally, which ignores the importance of the opinion words. We highlight the confidences of opinion words identified by their larger sentiment scores. Specifically, we obtain the sentiment-attention scores from a pre-trained attention-based sentiment classifier parameterized by θ_{senti} from (Wu et al., 2020b)[1]. After acquiring the representations $\{z_1, z_2, \ldots, z_n\}$ of the unlabeled sentence, the attention score α_i of z_i is calculated as follows:

$$\begin{aligned} z_{avg} &= \frac{1}{n} \sum_{i=1}^{n} z_i, \\ f(z_i, z_{avg}) &= z_i \cdot \mathbf{W} \cdot z_{avg} + \mathbf{b}, \\ \alpha_i &= \frac{e^{f(z_i, z_{avg})}}{\sum_{j=1}^{n} e^{f(z_j, z_{avg})}}, \end{aligned} \quad (7)$$

where $\mathbf{W}$ and $\mathbf{b}$ are learnable weight and bias. We then compute the sentiment-aware confidence based on the obtained attention scores as:

$$sc_{senti} = \sum_{i=1}^{n} \alpha_i \cdot \max(p_i(y|\theta; s^u, t)). \quad (8)$$

Similar to Eq. (6), any sentence whose sentiment-aware confidence is lower than the threshold T will be masked in consistency regularization. The final filtering mechanism of noisy sentences can be expressed as:

$$\mathbb{1}(sc_{senti} > T), \quad (9)$$

where $\mathbb{1}(\cdot > T)$ is the indicator function for confidence thresholding with T being the threshold.

3.3 Word-level Filtering

To further reduce the noise in the unlabeled data, we filter the noisy words with a more fine-grained confidence thresholding mechanism. The filtering of noisy words can be expressed as

$$\mathbb{1}(\max(p_i(y|\theta; s^u, t)) > \tau). \quad (10)$$

Any word with a confidence lower than τ is masked. Note that the word-level threshold τ can be different from the sentence-level threshold T.

3.4 Training Objective

For labeled sentences, the supervised loss is the same as Eq. (5). For unlabeled sentences, the consistency loss is:

$$\begin{aligned} \mathcal{L}_c = &\mathbb{1}(sc_{senti} > T) \\ &\cdot \{\frac{1}{n} \sum_{i=1}^{n} \mathbb{1}(\max(p_i(y|\theta; s^u, t)) > \tau) \\ &\cdot \mathcal{H}(\hat{p}_i(y|\theta; s^u, t), p_i(y|\theta; \omega(s^u), t))\}. \end{aligned} \quad (11)$$

The final training objective is given by:

$$\mathcal{L} = \mathcal{L}_s + \mathcal{L}_c. \quad (12)$$

[1]The sentiment classifier we use is the same as (Wu et al., 2020b), except that we use BERT as the word representation model while they use GloVe vectors (Pennington et al., 2014).

Datasets		#sentences	#opinion targets
14res	Train	1,627	2,643
	Test	500	864
15res	Train	754	1,076
	Test	325	436
16res	Train	1,079	1,512
	Test	329	457
14lap	Train	1,158	1,634
	Test	343	482
Yelp	Unlabeled	100,000	-
Amazon	Unlabeled	100,000	-

Table 1: Statistics of TOWE datasets and unlabeled datasets. For TOWE datasets, sentence may contain multiple opinion targets. For unlabeled datasets, we randomly sampled data from Yelp for 14res, 15res, 16res datasets and Amazon for 14lap dataset. The unlabeled data is available at https://github.com/TOWESSL/TOWESSL.

Hyperparameter	TOWE model	Sentiment Classifier
Batch size	16(96)	128
Epochs	50	-
Steps	-	3000
Learning rate (BERT)	2e-5	1e-5
Learning rate (Others)	2e-4	1e-4
Hidden dimension	512	512
Optimizer	AdamW	AdamW

Table 2: Experimental setting of the training of the TOWE model and the sentiment classifier. For the TOWE model, batch size for labeled data is 16 and 96 for unlabeled data.

4 Experiments

4.1 Datasets and Metrics

Following previous studies (Fan et al., 2019; Wu et al., 2020b; Veyseh et al., 2020; Mao et al., 2021; Jiang et al., 2021; Feng et al., 2021; Zhang et al., 2021b), we conduct evaluations on four benchmark datasets for TOWE. The suffixes 'res' and 'lap' refer to restaurant reviews and laptop reviews, respectively. For the 14res, 15res, and 16res datasets, we use the unlabeled sentences from Yelp[2]. For the 14lap dataset, we use the unlabeled sentences from Amazon Electronics[3] (Ni et al., 2019). The statistics of these datasets are listed in Table 1.

We use the evaluation metrics of precision, recall, and F1 score to measure the performance of different methods following previous studies (Fan et al., 2019; Wu et al., 2020b; Jiang et al., 2021). An extraction is considered correct only if opinion words from the beginning to the end are all correctly predicted.

4.2 Experimental Settings

For our MGCR method, we set the hidden size of both the BERT and the Transformer to 512. The mini-batch sizes of labeled and unlabeled data are set to 16 and 96, respectively. All parameters are optimized using the AdamW optimizer (Loshchilov and Hutter, 2018) with an initial learning rate 2e-5 for BERT and 2e-4 for others. We randomly split 20% of the training set as the validation set and used early stopping. We search different combinations of sentence-level and word-level confidence thresholds for each dataset and use the ones with best validation performances. The experimental results of different threshold combinations are provided in Table 5, Table 6, Table 7 and Table 8. The training details of the pseudo opinion targets generator and the sentiment classifier are provided in Table 2, respectively. We pre-train the sentiment classifier on the same data as the unlabeled data used for MGCR.

4.3 Baselines

We compare our MGCR method with the following methods.

Distance-rule (Fan et al., 2019) uses POS tags and regards the nearest adjective to the opinion target as the corresponding opinion word.

Dependency-rule (Fan et al., 2019) builds templates from the training set with the POS tags of opinion targets and opinion words and the dependency path between them, then uses the hign-frequency dependency templates for target-oriented opinion words extraction on the testing set.

TC-BiLSTM (Fan et al., 2019) follows the design of the work for target-oriented sentiment classification (Tang et al., 2016) and concatenate an opinion target embedding for each word position to perform sequence labeling.

IOG (Fan et al., 2019) employs six different positional and directional LSTMs to encode sentence and then extract the opinion words of the target.

LOTN (Wu et al., 2020b) transfers the latent opinion knowledge from sentiment classification task into the TOWE task via an auxiliary learning task.

ONG (Veyseh et al., 2020) leverages syntax-based opinion possibility scores and the syntactic

[2]https://www.yelp.com
[3]https://www.amazon.com

Methods	14res			15res			16res			14lap		
	P	R	F1	P	R	F1	P	R	F1	P	R	F1
Distance-rule (Fan et al., 2019)	58.39	43.59	49.92	54.12	39.96	45.97	61.90	44.57	51.83	50.13	33.86	40.42
Dependency-rule (Fan et al., 2019)	64.57	52.72	58.04	65.49	48.88	55.98	76.03	56.19	64.62	45.09	31.57	37.14
TC-BiLSTM (Fan et al., 2019)	67.65	67.67	67.61	66.06	60.16	62.94	73.46	72.88	73.10	62.45	60.14	61.21
IOG (Fan et al., 2019)	82.85	77.38	80.02	73.24	69.63	71.35	76.06	70.71	73.25	**85.25**	78.51	81.69
LOTN (Wu et al., 2020b)	84.00	80.52	82.21	76.61	70.29	73.29	86.57	80.89	83.62	77.08	67.62	72.02
ONG (Veyseh et al., 2020)	83.23	81.46	82.33	76.63	81.14	78.81	87.72	84.38	86.01	73.87	77.78	75.77
Dual-MRC (Mao et al., 2021)	**89.79**	78.43	83.73	77.19	71.98	74.50	86.07	80.77	83.33	78.21	**81.66**	79.90
PER (Dai et al., 2022)	86.43	80.39	83.30	81.50	75.05	78.14	90.00	84.00	86.90	80.68	70.72	75.38
ARGCN (Jiang et al., 2021)	87.32	83.59	85.42	78.81	77.69	78.24	88.49	84.95	86.69	75.83	76.90	76.36
TSMSA (Feng et al., 2021)	-	-	86.37	-	-	81.64	-	-	89.20	-	-	82.18
MRC-MVT (Zhang et al., 2021b)	86.31	**89.42**	87.83	82.04	81.54	81.79	90.60	88.19	89.38	79.59	81.12	80.84
MGCR (ours)	88.65	89.36	**89.01**†	**84.29**	**83.37**	**83.80**†	**91.31**	**91.74**	**91.51**†	83.76	81.25	**82.47**†

Table 3: Main results (%) including recall, precision and F1-score. The best results are in bold and second-best results are underlined. Results of all comparison methods were copied from the original papers. The marker † represents that MGCR outperforms other methods significantly ($p < 0.01$) .

Methods	14res			15res			16res			14lap		
	P	R	F1	P	R	F1	P	R	F1	P	R	F1
MGCR	88.65	**89.36**	**89.01**	**84.29**	83.37	**83.80**	**91.31**	**91.74**	**91.51**	83.76	**81.25**	**82.47**
w/o Pre-trained Sentiment Classifier	87.69	89.03	88.35	82.79	82.89	82.77	90.67	90.60	90.63	**84.18**	79.19	81.05
w/o Filtering Noisy Unlabeled Sentences	**88.84**	88.00	88.41	80.13	**85.39**	82.62	89.59	91.68	90.62	82.84	78.83	80.77
w/o Filtering Noisy Unlabeled Words	87.29	88.12	87.70	80.10	85.33	82.66	91.02	91.30	91.16	81.99	80.19	81.07
w/o Consistency Regularization (Labeled Data Only)	87.34	87.05	87.19	82.42	81.81	82.11	87.19	88.38	87.76	81.70	77.89	79.70

Table 4: Ablation study results (%) when removing different components from MGCR method.

connections between the words for TOWE.

Dual-MRC (Mao et al., 2021) used BERT as the encoder and transforms the TOWE task into a question answering (QA) problem to solve.

PER (Dai et al., 2022) proposes a padding-enhanced reinforcement learning model on both sequential structure and syntactic structure to extract opinion words for opinion targets.

ARGCN (Jiang et al., 2021) proposes a directed syntactic dependency graph and a attention-based relational graph convolutional neural network to exploit syntactic information for TOWE.

TSMSA (Feng et al., 2021) design a target-specified sequence labeling with multi-head self-attention based on transformer architecture.

MRC-MVT (Zhang et al., 2021b) leverages a machine reading comprehension model trained with a multiview paradigm to extract target-oriented opinion words.

4.4 Main Results and Discussion

Table 3 shows main results of different methods on four benchmarks. These results demonstrate that MGCR achieves the best F1-score on all datasets. Concretely, MGCR has the following advantages:

- MGCR significantly outperforms other methods. In these methods, rule-based methods (Distance-rule and Dependency-rule) achieve the worst performance, since they lack robustness and only cover a small number of cases. By contrast, neural methods obtain obvious improvements by introducing the external knowledge (e.g., LOTN and ARGCN) or other solution paradigm (e.g., Dual-MRC and MRC-MVT). Among the neural methods, MRC-MVT achieves very competitive results on all datasets with using machine reading framework. Nevertheless, our MGCR still outperforms it by 1.18%, 2.01%, 2.13%, and 1.63% in F1-score respectively on 14res, 15res, 16res and 14lap datasets. These comparisons demonstrate the great superiority of MGCR in exploiting unlabeled data to reduce distribution shift for TOWE, thereby successfully boosting the extraction performance.

- With few labeled TOWE data, MGCR can also bring great performance gain. As illustrated in Table 1, the labeled training sentences of the 14res dataset is about twice the amount of the 15res dataset, while MGCR outperforms MRC-MVT by 1.18% and 2.01% in F1-score respectively on 14res and 15res datasets. The more improvement on 15res proves that MGCR is promising because it

τ \ T	0.5			0.7			0.9		
	P	R	F1	P	R	F1	P	R	F1
0.5	89.55	86.53	88.02	88.60	88.28	88.44	87.88	89.29	88.55
0.7	89.58	87.63	88.60	87.80	88.77	88.28	**88.65**	**89.36**	**89.01**
0.9	88.64	87.83	88.23	87.65	89.28	88.45	87.73	88.35	88.03

Table 5: Results (%) of combinations of sentence-level threshold and word-level threshold on 14res. T and τ represent sentence-level threshold and word-level threshold respectively.

τ \ T	0.5			0.7			0.9		
	P	R	F1	P	R	F1	P	R	F1
0.5	81.31	82.62	81.94	81.17	83.57	82.33	82.16	84.17	83.12
0.7	80.13	85.39	82.62	81.84	84.04	82.92	**84.29**	**83.37**	**83.80**
0.9	81.38	83.43	82.38	81.41	84.38	82.81	81.35	84.24	82.73

Table 6: Results (%) of combinations of sentence-level threshold and word-level threshold on 15res. T and τ represent sentence-level threshold and word-level threshold respectively.

perform well even with few labeled sentences.

4.5 Ablation Study

To evaluate the effectiveness of each component of MGCR, we conduct an ablation study. As shown in Table 4, we observe different levels of performance degradation on the four TOWE datasets when removing the components from MGCR. Specifically, when removing the pre-trained sentiment classifier, the F1 score of MGCR drops from 0.5% to 1.5% on all datasets, indicating that using sentiment-attention scores to emphasize latent opinion words helps better filter the noisy unlabeled sentences. After removing either of the coarse-grained and fine-grained filtering processes, the performance declines on all datasets, demonstrating that these two confidence-based thresholding mechanisms alleviate the issue of confirmation bias caused by noisy training signals in consistency regularization. In addition, we find that removing the consistency regularization (i.e., supervised learning only) worsens the performance most.

4.6 Results of Different Thresholds Combinations

We present the results of different combinations of sentence-level threshold and word-level thresholds on four TOWE datasets. The range of each threshold is from 0.5 to 0.9. The detailed results for the 14res, 15res, 16res, and 14lap dataset are listed in Tables 5, 6, 7 and 8, respectively. An interesting finding is that using a high T with a low τ is the best strategy on most datasets, which means we should filter noisy unlabeled sentences more strictly than filtering noisy unlabeled words.

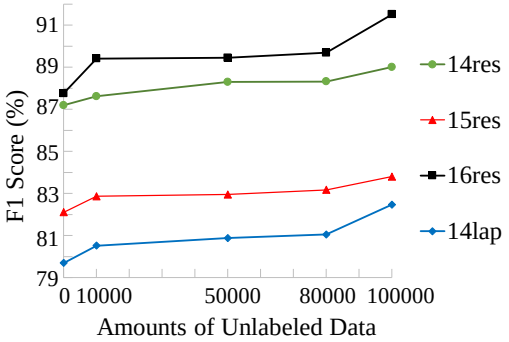

Figure 4: F1-score (%) on four TOWE datasets with varying amounts of unlabeled data.

τ \ T	0.5			0.7			0.9		
	P	R	F1	P	R	F1	P	R	F1
0.5	90.44	90.66	90.55	**91.31**	**91.74**	**91.51**	91.09	90.79	90.94
0.7	90.21	90.72	90.46	89.85	91.68	90.76	90.73	90.72	90.72
0.9	89.45	90.35	89.88	90.16	91.36	90.75	90.90	91.36	91.13

Table 7: Results (%) of combinations of sentence-level threshold and word-level threshold on 16res. T and τ represent the sentence-level threshold and word-level threshold respectively.

τ \ T	0.5			0.7			0.9		
	P	R	F1	P	R	F1	P	R	F1
0.5	83.29	79.65	81.42	84.85	78.07	81.32	84.43	77.51	80.82
0.7	83.78	79.48	81.56	84.54	79.36	81.86	83.29	78.48	80.78
0.9	82.43	78.83	80.77	84.21	78.82	81.36	**83.76**	**81.25**	**82.47**

Table 8: Results (%) of combinations of sentence-level threshold and word-level threshold on 14lap. T and τ represent the sentence-level threshold and word-level threshold respectively.

4.7 Effect of Amounts of Unlabeled Data

We conduct experiments by varying the amounts of unlabeled data to investigate the effect of different data amounts. The results are shown in Figure 4. Compared with supervised training only, even a few unlabeled data can improve the performance on all datasets, which validates the effectiveness of utilizing semi-supervised consistency regularization. Besides, the generalization performance is better with more unlabeled data used, because more data effectively reduce the risk of domain shift between test data and training data. This trend demonstrates the potential of MGCR, which are likely to be further improved with even more unlabeled sentences and computing resources.

Methods	NULL	Under-extracted	Over-extracted	Others	Total
MGCR	**2**	**9**	**24**	8	**43**
MGCR **w/o** Pre-trained Sentiment Classifier	3	12	29	11	54
MGCR **w/o** Filtering Noisy Unlabeled Sentences	5	**9**	29	**7**	55
MGCR **w/o** Filtering Noisy Unlabeled Words	4	14	38	11	67
MGCR **w/o** Consistency Regularization (Labeled Data Only)	8	11	44	15	70

Table 9: Statistics of different error types of our MGCR method and different ablation versions on the 16res dataset.

4.8 Error Analysis

We count all the error instances on the 16res dataset to analyze the distribution of different error types. We categorize errors into four types: 'NULL': no opinion word is extracted, 'Under-extracted': only parts of opinion words are extracted from the input sentences, 'Over-extracted': the model extracts redundant opinion words, and 'Others': wrong extraction and other error types. As shown in Table 9, training with only labeled data suffers from the severe overfitting problem, which leads to many 'Over-extracted' errors. For instance, in the sentence "*open & cool place with the best pizza and coffee.*", the overfitted TOWE model extracts both "*cool*" and "*best*" as opinion words for the opinion target "*coffee*", actually the word "*cool*" is the opinion word for "*place*". Our MGCR reduces the number of 'Over-extracted' errors from 44 to 24, indicating that introducing high-quality unlabeled data improves the generalization performance of the TOWE model. It is worth noting that MGCR also reduces 40% of total errors compared with training with only labeled data.

5 Related Work

5.1 TOWE

Aspect-based Sentiment Analysis (ABSA) contains a set of various subtasks (Pontiki et al., 2014; Liu et al., 2015; Tang et al., 2016; Wang et al., 2016; Ning et al., 2018; Zhao et al., 2020; Wu et al., 2020a). Fan et al. (2019) first propose TOWE as a new subtask to expand the ABSA research, which aims to extract opinion words for a given opinion target from a sentence. They employ several LSTM networks and propose a target-fused neural sequence labeling model that achieves promising results on TOWE. Following the idea, many works design advanced multi-head self-attention or multi-view deep attention mechanism to generate target-specific context representation (Feng et al., 2021; Zhang et al., 2021b). On this basis, some works incorporate syntactic knowledge (Feng et al., 2021; Zhang et al., 2021b) or sentiment knowledge (Wu et al., 2020b) to further improve the performance of TOWE. There are also some other works that transform TOWE into a question answering problem and adopt the framework of machine reading comprehension (MRC) to solve TOWE (Mao et al., 2021; Zhang et al., 2021b), achieving very competitive results. Different from these works, we argue that insufficient labeled data greatly increase the risk of distribution shift for the TOWE task, and thus propose making use of massive unlabeled raw text to reduce the shift risk.

5.2 Semi-supervised Learning

Low-resource learning including semi-supervised learning (Sohn et al., 2020; Zhang et al., 2021a; Xie et al., 2020; Wang et al., 2022; Wang et al.; Oliver et al., 2018) has proven effective in the natural language processing (Peters et al., 2017; He et al.; Cheng et al., 2019; Gururangan et al., 2019; Izmailov et al., 2020; Sintayehu and Lehal, 2021; Clark et al., 2018; Ruder and Plank, 2018; Liu et al., 2022; Lu et al., 2022; Yang et al., 2021), especially ABSA tasks (Marcacini et al., 2018; Xu and Tan, 2019; Augenstein et al., 2018; Cheng et al., 2019; Li et al., 2020). Consistency regularization (Sajjadi et al., 2016) is a very popular semi-supervised learning technique, which has been widely applied in various tasks (Berthelot et al., 2019b,a; Sohn et al., 2020; Xie et al., 2020; Chen et al., 2020). Its core idea is to improve the robustness of the model by minimizing the discrepancy between data and its perturbation. Miao et al. (2020) first applied consistency regularization to aspect sentiment classification task by interpolating embeddings of input sentences. By contrast, we are the first to successfully apply semi-supervised learning consistency regularization to TOWE.

6 Conclusion

The TOWE task suffers from the risk of distribution shift which arises from scarce labeled data. In

this paper, we propose the novel MGCR method to increase the exposure of the model to varying distribution shifts by exploiting unlabeled data, and naturally the risk can be reduced. In the MGCR method, two different-grained filters, i.e., sentence-level (coarse-grained) and word-level (fine-grained) confidence-based thresholding, are designed to filter noisy sentences and words for the high-quality exposure. To further underline the possible opinion words during learning, we employ a pre-trained sentiment classifier and incorporate sentiment-attention scores into the sentence-level filter. Experimental results indicate that our MGCR method significantly outperforms all other TOWE methods and achieves state-of-the-art performance on four TOWE datasets. The in-depth analysis demonstrate the effectiveness of each component in MGCR.

7 Acknowledgements

We would like to thank anonymous reviewers for their insightful comments and suggestions to help improve the paper. This publication has emanated from research conducted with the financial support of the Pioneer and "Leading Goose" R&D Program of Zhejiang under Grant Number 2022SDXHDX0003.

References

Isabelle Augenstein, Sebastian Ruder, and Anders Søgaard. 2018. Multi-task learning of pairwise sequence classification tasks over disparate label spaces. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 1896–1906.

David Berthelot, Nicholas Carlini, Ekin D Cubuk, Alex Kurakin, Kihyuk Sohn, Han Zhang, and Colin Raffel. 2019a. Remixmatch: Semi-supervised learning with distribution matching and augmentation anchoring. In *International Conference on Learning Representations*.

David Berthelot, Nicholas Carlini, Ian Goodfellow, Nicolas Papernot, Avital Oliver, and Colin A Raffel. 2019b. Mixmatch: A holistic approach to semi-supervised learning. In *Advances in Neural Information Processing Systems*, pages 5049–5059.

Jiaao Chen, Zichao Yang, and Diyi Yang. 2020. Mixtext: Linguistically-informed interpolation of hidden space for semi-supervised text classification. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 2147–2157.

Xingyi Cheng, Weidi Xu, Taifeng Wang, Wei Chu, Weipeng Huang, Kunlong Chen, and Junfeng Hu. 2019. Variational semi-supervised aspect-term sentiment analysis via transformer. In *Proceedings of the 23rd Conference on Computational Natural Language Learning (CoNLL)*, pages 961–969.

Kevin Clark, Minh-Thang Luong, Christopher D Manning, and Quoc Le. 2018. Semi-supervised sequence modeling with cross-view training. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 1914–1925.

Yaqing Dai, Pengfei Wang, and Xiaofei Zhu. 2022. Reasoning over multiplex heterogeneous graph for target-oriented opinion words extraction. *Knowledge-Based Systems*, 236:107723.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. Bert: Pre-training of deep bidirectional transformers for language understanding. In *NAACL-HLT (1)*.

Zhifang Fan, Zhen Wu, Xinyu Dai, Shujian Huang, and Jiajun Chen. 2019. Target-oriented opinion words extraction with target-fused neural sequence labeling. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 2509–2518.

Yuhao Feng, Yanghui Rao, Yuyao Tang, Ninghua Wang, and He Liu. 2021. Target-specified sequence labeling with multi-head self-attention for target-oriented opinion words extraction. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 1805–1815.

Yaroslav Ganin, Evgeniya Ustinova, Hana Ajakan, Pascal Germain, Hugo Larochelle, François Laviolette, Mario Marchand, and Victor Lempitsky. 2016. Domain-adversarial training of neural networks. *The journal of machine learning research*, 17(1):2096–2030.

Suchin Gururangan, Tam Dang, Dallas Card, and Noah A Smith. 2019. Variational pretraining for semi-supervised text classification. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 5880–5894.

Ruidan He, Wee Sun Lee, Hwee Tou Ng, and Daniel Dahlmeier. Adaptive semi-supervised learning for cross-domain sentiment classification.

Pavel Izmailov, Polina Kirichenko, Marc Finzi, and Andrew Gordon Wilson. 2020. Semi-supervised learning with normalizing flows. In *International Conference on Machine Learning*, pages 4615–4630. PMLR.

Junfeng Jiang, An Wang, and Akiko Aizawa. 2021. Attention-based relational graph convolutional network for target-oriented opinion words extraction.

In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*, pages 1986–1997.

Dong-Hyun Lee et al. 2013. Pseudo-label: The simple and efficient semi-supervised learning method for deep neural networks. In *International Conference on Machine Learning*.

Ning Li, Chi-Yin Chow, and Jia-Dong Zhang. 2020. Seml: A semi-supervised multi-task learning framework for aspect-based sentiment analysis. *IEEE Access*, 8:189287–189297.

Ao Liu, An Wang, and Naoaki Okazaki. 2022. Semi-supervised formality style transfer with consistency training. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 4689–4701.

Pengfei Liu, Shafiq Joty, and Helen Meng. 2015. Fine-grained opinion mining with recurrent neural networks and word embeddings. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 1433–1443.

Ilya Loshchilov and Frank Hutter. 2018. Fixing weight decay regularization in adam.

Jinghui Lu, Linyi Yang, Brian Namee, and Yue Zhang. 2022. A rationale-centric framework for human-in-the-loop machine learning. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 6986–6996.

Yue Mao, Yi Shen, Chao Yu, and Longjun Cai. 2021. A joint training dual-mrc framework for aspect based sentiment analysis. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 35, pages 13543–13551.

Ricardo Marcondes Marcacini, Rafael Geraldeli Rossi, Ivone Penque Matsuno, and Solange Oliveira Rezende. 2018. Cross-domain aspect extraction for sentiment analysis: A transductive learning approach. *Decision Support Systems*, 114:70–80.

Zhengjie Miao, Yuliang Li, Xiaolan Wang, and Wang-Chiew Tan. 2020. Snippext: Semi-supervised opinion mining with augmented data. In *Proceedings of The Web Conference 2020*, pages 617–628.

George A Miller. 1995. Wordnet: a lexical database for english. *Communications of the ACM*, 38(11):39–41.

Jianmo Ni, Jiacheng Li, and Julian McAuley. 2019. Justifying recommendations using distantly-labeled reviews and fine-grained aspects. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 188–197.

Tianhao Ning, Zhen Wu, Xin-Yu Dai, Jiajun Huang, Shujian Huang, and Jiajun Chen. 2018. Improving aspect identification with reviews segmentation. In *CCF International Conference on Natural Language Processing and Chinese Computing*, pages 416–428. Springer.

Avital Oliver, Augustus Odena, Colin A Raffel, Ekin Dogus Cubuk, and Ian Goodfellow. 2018. Realistic evaluation of deep semi-supervised learning algorithms. *Advances in neural information processing systems*, 31.

Jeffrey Pennington, Richard Socher, and Christopher D Manning. 2014. Glove: Global vectors for word representation. In *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*, pages 1532–1543.

Matthew Peters, Waleed Ammar, Chandra Bhagavatula, and Russell Power. 2017. Semi-supervised sequence tagging with bidirectional language models. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1756–1765.

Maria Pontiki, Dimitris Galanis, John Pavlopoulos, Harris Papageorgiou, Ion Androutsopoulos, and Suresh Manandhar. 2014. Semeval-2014 task 4: Aspect based sentiment analysis. In *SemEval@COLING 2014*, pages 27–35.

Sebastian Ruder and Barbara Plank. 2018. Strong baselines for neural semi-supervised learning under domain shift. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1044–1054.

Mehdi Sajjadi, Mehran Javanmardi, and Tolga Tasdizen. 2016. Regularization with stochastic transformations and perturbations for deep semi-supervised learning. In *Advances in neural information processing systems*, pages 1163–1171.

H Sintayehu and GS Lehal. 2021. Named entity recognition: a semi-supervised learning approach. *International Journal of Information Technology*, 13(4):1659–1665.

Kihyuk Sohn, David Berthelot, Nicholas Carlini, Zizhao Zhang, Han Zhang, Colin A Raffel, Ekin Dogus Cubuk, Alexey Kurakin, and Chun-Liang Li. 2020. Fixmatch: Simplifying semi-supervised learning with consistency and confidence. *Advances in Neural Information Processing Systems*, 33.

Duyu Tang, Bing Qin, Xiaocheng Feng, and Ting Liu. 2016. Effective lstms for target-dependent sentiment classification. In *Proceedings of COLING 2016, the 26th International Conference on Computational Linguistics: Technical Papers*, pages 3298–3307.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in neural information processing systems*, pages 5998–6008.

Amir Pouran Ben Veyseh, Nasim Nouri, Franck Dernoncourt, Dejing Dou, and Thien Huu Nguyen. 2020. Introducing syntactic structures into target opinion word extraction with deep learning. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 8947–8956.

Wenya Wang, Sinno Jialin Pan, Daniel Dahlmeier, and Xiaokui Xiao. 2016. Recursive neural conditional random fields for aspect-based sentiment analysis. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 616–626.

Yidong Wang, Hao Chen, Yue Fan, SUN Wang, Ran Tao, Wenxin Hou, Renjie Wang, Linyi Yang, Zhi Zhou, Lan-Zhe Guo, et al. Usb: A unified semi-supervised learning benchmark.

Yidong Wang, Hao Chen, Qiang Heng, Wenxin Hou, Marios Savvides, Takahiro Shinozaki, Bhiksha Raj, Zhen Wu, and Jindong Wang. 2022. Freematch: Self-adaptive thresholding for semi-supervised learning. *arXiv preprint arXiv:2205.07246*.

Zhen Wu, Chengcan Ying, Fei Zhao, Zhifang Fan, Xinyu Dai, and Rui Xia. 2020a. Grid tagging scheme for aspect-oriented fine-grained opinion extraction. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 2576–2585.

Zhen Wu, Fei Zhao, Xin-Yu Dai, Shujian Huang, and Jiajun Chen. 2020b. Latent opinions transfer network for target-oriented opinion words extraction. In *The Thirty-Fourth AAAI Conference on Artificial Intelligence, AAAI 2020*, pages 9298–9305. AAAI Press.

Qizhe Xie, Zihang Dai, Eduard Hovy, Thang Luong, and Quoc Le. 2020. Unsupervised data augmentation for consistency training. *Advances in Neural Information Processing Systems*, 33.

Weidi Xu and Ying Tan. 2019. Semi-supervised target-oriented sentiment classification. *Neurocomputing*, 337:120–128.

Linyi Yang, Jiazheng Li, Pádraig Cunningham, Yue Zhang, Barry Smyth, and Ruihai Dong. 2021. Exploring the efficacy of automatically generated counterfactuals for sentiment analysis. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 306–316.

Bowen Zhang, Yidong Wang, Wenxin Hou, Hao Wu, Jindong Wang, Manabu Okumura, and Takahiro Shinozaki. 2021a. Flexmatch: Boosting semi-supervised learning with curriculum pseudo labeling. *Advances in Neural Information Processing Systems*, 34.

Jingyuan Zhang, Zequn Zhang, Zhi Guo, Li Jin, Kang Liu, and Qing Liu. 2021b. Enhancement of target-oriented opinion words extraction with multiview-trained machine reading comprehension model. *Computational Intelligence and Neuroscience*, 2021.

Fei Zhao, Zhen Wu, and Xinyu Dai. 2020. Attention transfer network for aspect-level sentiment classification. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 811–821.

Learnable Dependency-based Double Graph Structure for Aspect-based Sentiment Analysis

Yinglong Ma and Yunhe Pang

School of Control and Computer Engineering
North China Electric Power University
Beinong Road #2, Beijing 102206, China
{yinglongma, pangyunhe}@ncepu.edu.cn

Abstract

Dependency tree-based methods might be susceptible to the dependency tree due to that they inevitably introduce noisy information and neglect the rich relation information between words. In this paper, we propose a learnable dependency-based double graph (LD2G) model for aspect-based sentiment classification. We use multi-task learning for domain adaptive pretraining, which combines Biaffine Attention and Mask Language Model by incorporating features such as structure, relations and linguistic features in the sentiment text. Then we utilize the dependency enhanced double graph-based MPNN to deeply fuse structure features and relation features that are affected with each other for ASC. Experiment on four benchmark datasets shows that our model is superior to the state-of-the-art approaches.

1 Introduction

Aspect-based sentiment analysis (ABSA) is a fine-grained sentiment analysis task for extracting the emotional orientation of the given one or more aspects in a sentence. ABSA is generally divided into two parts: aspect extraction (AE) and aspect sentiment classification (ASC). We will focus on ASC task in this paper.

Dependency-based approaches have illustrated promising performance by utilizing the spatial structure of the dependency tree (DT). Early studies focus on the syntax feature enhanced aspect-based sentiment analysis (ABSA) models (Brun et al., 2014; Kiritchenko et al., 2014). Dependency-based analysis currently tends to use graph neural networks to learn representation over DT due to that graph neural network can directly learn on graph data including DTs (Kipf et al., 2016). DT is regarded as an undirected tree structure to produce a symmetry adjacency matrix for Graph Convolutional Networks (GCN) (Sun et al., 2019; Zhang et al., 2019). (Wang et al., 2020) use GNNs to capture implicit relation between aspect words and others and achieved encouraged results. (Wang et al., 2021) utilized reinforcement learning for aspect-based sentiment classification (ASC).

In this paper, we consider how to effectively fuse structure and relation features for ASC. the following two questions: 1) How to utilize relations of edges between words for improving ASC? Most of existing approaches consider the structure information and neglected the rich edge relations between words with exception to the works (Chen et al., 2022; Tian et al., 2021a; Tian et al., 2021b). However, these works consider graph features (i.e., structure, edge relations, tree distance) as different input sources, and do not explore how to effectively fuse various features because these features are affected with each other. 2) How to effectively handle the multi-feature noisy information from DT for improving ASC? Existing approaches directly utilize the DT generated by a tool like spaCy. Direct application of DT is susceptible to noisy information in the dependency tree (Tang et al., 2020), which may achieve lower performance compared to using only the flat structure. DT includes 45 kinds of relations such as *nsubj*, *amod* and *conj*, and so on. Dependency parsing requires effective way to reduce DT's noisy information.

To handle the two problems, we propose a learnable dependency-based double graph (LD2G) model for aspect-based sentiment classification. We use multi-task learning (MTL) for domain adaptive pretraining and reducing DT's noisy information, by combining Biaffine Attention Model (BAM) and Mask Language Model (MLM) and

Proceedings of the 29th International Conference on Computational Linguistics, pages 7086–7092
October 12–17, 2022.

parsing structure, relations and linguistic features. We utilize the dependency enhanced double graph-based MPNN to deeply fuse structure features and relation features that are affected with each other for ASC. Experiment on four benchmark datasets shows that our model is superior to the state-of-the-art approaches. To the best of our knowledge, LD2G is the first work that jointly considers the structure and relations features in a unified domain adaptive pretraining framework. The main contributions of this paper are as follows.

- We propose a learnable dependency-based double graph MPNN model to learn dependency-based feature for ABSA.
- We present an MTL method for domain adaptive pretraining by combining Biaffine Attention and MLM model.
- Experiments were made over four datasets, and the results show that our model outperforms state-of-the-art approaches.

2 Related work

Generally, sentiment analysis aims to predict the sentiment of a sentence or a document. In contrast ABSA is a fine-grained task, which is to perform sentiment analysis on aspects by first identifying aspect words and then predicting the sentiment of the aspect. Early methods mainly use machine learning methods, but these methods are more dependent on manual feature labeling (Jiang et al., 2011; Vo et al., 2015). Later, neural network approaches were found to have better performance. A common method was to model the sequence of aspect words and context. The popular LSTM model and attention mechanism were commonly used to fuse the context semantics and obtain the important parts from it. In recent years, approaches based on pre-trained model has become the main method for contextual semantic integration, which brings considerable improvement to sentiment analysis.

Another kind of approach takes advantage of the spatial structure of syntax features by deep learning methods. Early work has studied the syntax feature enhanced ABSA models (Brun et al., 2014, Kiritchenko et al., 2014). Dependency tree and constituent tree can represent the syntactic information between the words, but these approaches based on them has not become more popular until deep learning method was applied. After the improvements brought by deep learning, syntax features can effectively improve the performance of ABSA task. In recent years, syntax approaches tend to use graph neural networks to represent the dependency tree, because GNNs can directly model non-Euclidean data which fits dependency tree well. (Sun et al., 2019; Zhang et al., 2019) combined Bi-LSTM and GCN (Kipf et al., 2016) to extract feature from both linear and spatial word representations, in which dependency tree is seemed as an undirected dependency tree to produce a symmetry adjacency matrix for GCN. (Dai et al.,2021) generated an induced tree structure based on the pre-trained model instead of dependency tree, and the experiment shows that pre-trained models can optimize the induced tree in the following ABSA tasks. (Zhou et al., 2021) shortened the distance between aspect word and emotional words by learning a new aspect-centric tree. (Tang et al., 2020) combined a dependency graph and an attention-based graph to learn representation based on GNN. There are some approaches such as (Chen et al., 2022; Tian et al., 2021a; Tian et al., 2021b; Wang et al., 2021), further explored the use of specific relation on the dependency trees. These works utilize the original features of the dependency tress.

Dependency-based methods add effective syntactic features to sentiment analysis tasks, but at the same time, they inevitably inherit wrong dependency tree information, which brings adverse influence on downstream task. Therefore, it has become a crucial problem how to utilize relations of edges between words and handle the multi-feature noisy information effectively from DT for improving ASC

3 LD2G model

The LD2G model overall is shown in Figure 1. We use a two-stage approach to implement aspect sentiment analysis: 1) Domain adapted pretraining, and 2) Double graph MPNN-based aspect sentiment classification. The whole model is divided into three modules: BERT module, biaffine parser and message passing neural network. We first apply MTL on BERT and biaffine parser module for domain adaptive pretraining by combining BAM and MLM, then obtain the sparse score matrix of structure graph and the feature matrix of edge relations from BAM. Both them are further fused into

a dependency enhanced double graph matrix. The LD2G model can learn from the dependency graphs and optimize the dependency graph structure during the training process.

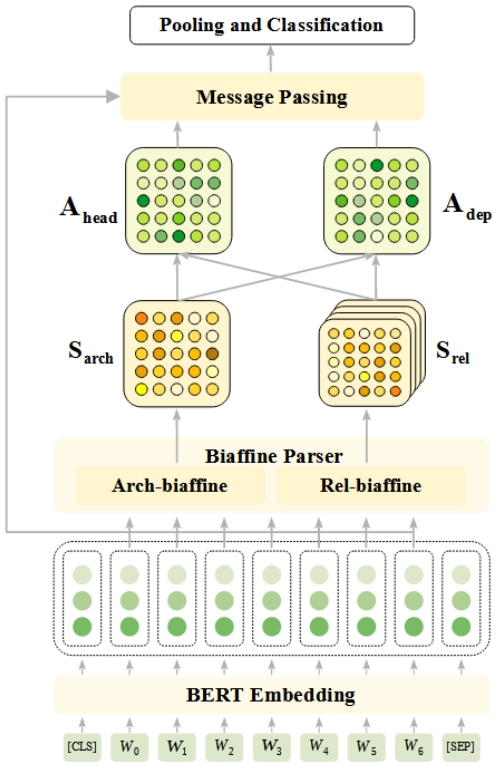

Fig 1. The architecture of LD2G model.

3.1 Domain adapted pretraining

A multi-task learning (MTL) task is launched for domain adaptive dependency feature fusion, which combines dependency parsing (DP) and MLM (Devlin et al., 2018) by incorporating features such as structure, relations and linguistic features in the sentiment text. For a given sentence $s = \{w_0, w_1, \dots, w_n\}$ as input, we obtain the word embedding $H = \{h_0, h_1, \dots, h_n\}$ with BERT. MLM is applied to BERT model for domain adaptation, using the same mask strategy with (Devlin et al., 2018). Biaffine attention model (Dozat and Manning, 2017) is used based on the output of BERT model for dependency parsing. We use the architecture biaffine and relation biaffine to obtain the score matrices.

$$S_{arch} = Biaffine_{arch}(H) \quad (1)$$

$$S_{rel} = Biaffine_{rel}(H) \quad (2)$$

Where $S_{arch} \in \mathbb{R}^{n \times n}$, and $S_{rel} \in \mathbb{R}^{n \times n \times d}$ for all the edges in S_{arch}. In this paper, d=45, which is the number of dependency relation types. The DP task and MLM task will be combined as a MTL task.

The pretrained dependency graph will be taken as the original input of double graph MPNN model. The pretrained BAM with optimal initialization parameters will be used for MPNN training. After the BAM dependency parsing is trained, we can obtain the intermediate dense score matrix S_{arch} and feature matrix S_{rel}.

3.2 Double graph MPNN-based aspect sentiment classification

The structure information S_{arch} can be regarded as an adjacency matrix A_{arch}, and the relation information S_{rel} as a relation graph matrix A_{rel} by MLP.

$$A_{rel} = MLP(S_{rel}) \quad (3)$$

We use an adjacency matrix and its transpose to distinguish the two directions between head-dependent and dependent-head pairs in dependency graph. Four graphs are respectively represented by adjacency matrices A_{arch} and A_{rel}, and their transposes A_{arch}^{T} and A_{rel}^{T}. The structure and relation features can be fused into a dependency enhanced graph A_{head} with transpose A_{dep}.

$$A_{head} = A_{arch} + A_{rel} \quad (4)$$

$$A_{dep} = A_{arch}^{T} + A_{rel}^{T} \quad (5)$$

Where all the matrices are $\mathbb{R}^{n \times n}$ matrices.

Row normalization is applied to each matrix. The token embeddings from BERT output H that is straightly fed into the MPNN graph neural networks as node representation H^0. To learn the node representation over both A_{head} and A_{dep}, we take the bidirectional message passing (Kampffmeyer et al., 2019) as one layer to acquire final hidden states.

$$H^{l+\frac{1}{2}} = LN(Relu(A_{head} H^{l} \Theta_{head}^{l}) + H^{0}) \quad (6)$$

$$H^{l+1} = LN(Relu(A_{dep} H^{l+\frac{1}{2}} \Theta_{dep}^{l}) + H^{0}) \quad (7)$$

Where LN denotes layer normalization. After two layers of message passing, we can obtain the final representation. The pooling function is used to the hidden state of aspect tokens for obtaining the aspect representation $\boldsymbol{r}$, by a softmax normalization layer to yield a probability distribution $\boldsymbol{p_c}$.

$$\boldsymbol{p_c} = softmax(W_p \boldsymbol{r} + b_p) \quad (8)$$

	Models	Laptop14		Rest14		Rest15		Rest16	
		Acc	F1	Acc	F1	Acc	F1	Acc	F1
Baselines	ASGCN-BERT	77.90	73.01	83.78	75.02	80.69	62.02	89.99	74.46
	SK-GCN-BERT	79.00	75.57	83.48	75.19	83.20	66.78	87.19	72.02
	BERT + MLP	78.36	74.16	85.35	78.38	82.16	64.96	89.43	74.20
	R-GAT+BERT	78.53	74.63	85.63	78.82	81.61	65.30	90.96	75.26
	ACLT	79.68	75.83	85.71	78.44	84.44	72.08	92.15	78.64
	DGEDT-BERT	79.80	75.60	**86.30**	**80.00**	84.00	71.00	91.90	79.00
	TGCN-BERT	80.25	**76.92**	85.54	78.86	85.07	**72.50**	91.83	76.86
	KVMN-BERT	79.78	76.14	85.98	77.94	84.14	68.49	90.52	73.68
Ablation study	Pretrain BERT+MLP	79.15	75.15	85.63	78.56	85.24	71.29	92.21	78.02
	LD2G with static biaffine	77.36	73.01	84.46	75.79	84.50	68.34	90.09	72.09
	LD2G w/o MTL pretrain	78.53	74.03	84.73	78.22	83.39	70.39	91.40	78.97
	LD2G w/o rel. graph	79.93	75.94	85.53	78.24	84.87	67.91	91.23	75.13
	LD2G w/o arch. graph	79.62	76.12	85.36	78.24	84.50	65.84	92.21	77.67
	LD2G	**80.56**	76.61	85.98	79.45	**85.79**	72.11	**92.69**	**80.48**

Table 1 Performance evaluation of ASC over four datasets.

Where W_p and b_p are both trainable parameters. The loss is calculated by cross entropy:

$$\ell = -\sum_{d\in D}\sum_{c\in P} log\ \boldsymbol{p_c} \quad (9)$$

Where D denotes the training dataset, and P denotes all the polarities in ASC.

4 Experiment

4.1 Experimental setup and datasets

We use Stanford Dependency (SD) conversion of the English Penn Treebank (PTB 3.3.0) dependency datasets to train the BERT model and dependency parser. For pretraining and aspect sentiment analysis, we made the experiments over four datasets: Laptop14, Rest14, Rest15 and Rest16. Datasets Laptop14 and Rest14 are respectively from SemEval 2014 task 4 (Pontiki et al., 2014); Dataset Rest15 is from SemEval 2015 task 12 (Pontiki et al., 2015) and Rest16 is from SemEval 2016 task 5 (Pontiki et al., 2016), consisting of data from two categories: laptop and restaurant. Each of the datasets has three categories: positive, negative and neutral.

4.2 Parameters settings

We used the similar settings in dependency parsing and MPNN-based ASC. The common AdamW optimizer was used. The learning rate was 2e-5, warm up ratio is 0.1 and dropout rate is 0.1. We set the batch size as 16. In dependency parsing task, because the model converges quickly, we only train the model for 3 epochs to reduce the risk of overfitting while we train 10 epochs on ASC task.

4.3 Baselines

We compare the proposed LD2G[1] with the eight baselines and state-of-the-art alternatives: 1) ASGCN+BERT (Zhang et al., 2019); 2) SK-GCN+BERT (Zhou et al., 2020); 3) BERT-BASE+MLP (Dai et al., 2021); 4) R-GAT+BERT (Wang et al., 2020); 5) ACLT (Zhou et al., 2021); 6) DGEDT-BERT (Tang et al., 2020); 7) TGCN-BERT (Tian et al., 2021a); 8) KVMN-BERT (Tian et al., 2021b).

4.4 Overall Results

We use the accuracy and macro-averaged F1-score as evaluation metrics. The experimental results are shown in Table 1.

In Table 1, we observe that our LD2G model outperforms all the baseline methods over almost all datasets when we use simple message passing layers with residual connection. This shows that the utilization of double graph-based dependency enhanced structure fusing structure and relation features indeed contributes to ASC performance improvement. With the exceptions, LD2G has the

[1] https://github.com/LLLpc123/LDG

slightly lower ACC and F1 values than DGEDT-BERT over Rest14, and also has the slightly lower F1 values than TGCN-BERT over lap14 and rest15.

In In addition, the results show that all the baselines except for "DGEDT-BERT" have lower ASC performance than LD2G. A very likely reason is that: 1) BERT-based linguistic pretraining is not enough, and 2) their models do not use MTL pretraining and therefore are difficult to handle and reduce the DT's noisy information.

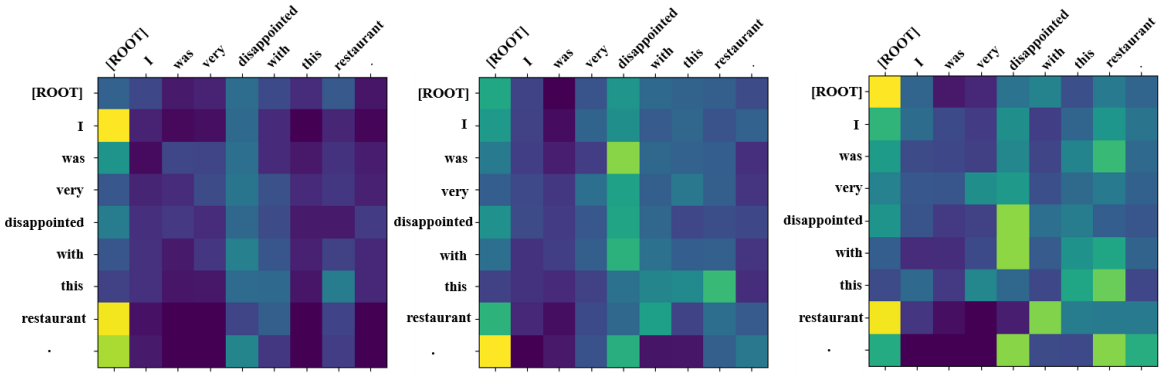

Fig 2. The change process of adjacency matrix. The figures from left to right respectively show the original output of trained dependency parser, the middle state of training process, and the adjacency matrix when the model converges. Light colors indicate that the location has higher weight.

4.5 Ablation study

We consider five ablation baselines including: 1) BERT-based pre-train: we replace the MTL pretrain only with BERT's MLM pretraining on ABSA dataset by a simple BERT+MLP. 2) LD2G with static biaffine. We lock the parameters of biaffine layer to observe the effect of a learnable graph structure. 3) LD2G w/o MTL pretrain. MTL pretrain is removed from LD2G to observe the effect. 4) LD2G w/o rel. graph: Relation graph for all the edges in the dependency tree is removed to observe the effects. And 5) LD2G w/o arch. graph: Architecture graph is removed. The results for ablation study are shown in Table 1.

From Table 1, LD2G with full modules overall outperforms the others with incomplete functions over all the datasets against metrics accuracy and F1 score. First, we observe that LD2G with MTL pretrain performs better than BERT+MLP model with MLM pretraining. This shows that BERT's MLM pretraining is not enough to reduce the DT's noisy information, but the MTL with BAM and MLM can effectively capture the various features in dependency parsing and obviously reduce the noise. Second, LD2G with static biaffine has the obviously lower performance than LD2G because the double graph-based MPNN in LD2G can continuously update and optimize BAM for generating the dependency graph with less DT's noisy information. In contrast, LD2G with static biaffine cannot dynamically optimize dependency parsing. Third, LD2G with MTL pretraining has obviously much higher performance than LD2G without MTL pretraining, which shows that MTL is crucial to effectively capture features and reduce DT's noisy information. Four, LD2G outperforms each of models LD2G w/o rel. graph and LD2G w/o arch. The combination of both graph structure feature and relation features indeed contributes to performance improvement. This also shows that using learnable dependency-based double graph MPNN indeed contributes to ASC performance.

4.6 Graph visualization-based case study

To illustrate the task-oriented learnable graph optimization of our model, we visualized the adjacency matrix in different phases and epochs. Considering a sample sentence, i.e., "I was very disappointed with this restaurant." As shown in Figure 2, the generated dependency graph structure is continuously optimized with the training of the model. The column of word '*disappointed*' is growing lighter, which shows the importance to classification task.

5 Conclusions and future work

In this paper, we proposed an LD2G model to fuse the syntactical structure and relation features for ASC. We use an MTL method for domain adaptive pretraining by combining Biaffine Attention and MLM model. A dependency enhanced double graph-based MPNN is utilized to deeply fuse structure features and relation features that are affected

with each other for ASC. Experiments results show that our model is superior to the state-of-the-art alternatives and the utilization of double graph-based dependency enhanced structure indeed contributes to ASC performance improvement.

In the future, we will analyze combinations of linguistic features and priori knowledge mining for ABSA.

Acknowledgements

This work is partially supported by National key research and development program under Grants (No. 2018YFC0830605, No. 2018YFC0831404).

Reference

Caroline Brun, Diana Nicoleta Popa, and Claude Roux.2014. XRCE: Hybrid classification for aspect-based sentiment analysis. In Proceedings of the 8th International Workshop on Semantic Evaluation (SemEval 2014), pages 838–842, Dublin, Ireland. Association for Computational Linguistics.

Hao Chen, Zepeng Zhai, Fangxiang Feng, Ruifan Li, and Xiaojie Wang. 2022. Enhanced multi-channel graph convolutional network for aspect sentiment triplet extraction. In Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), ACL 2022, Dublin, Ireland, May 22-27, 2022, pages 2974–2985. Association for Computational Linguistics.

Y. Chu and T. Liu. 1965. On the shortest arborescence of a directed graph. Scientia Sinica, 14: 1396–1400. American Mathematical Society.

Junqi Dai, Hang Yan, Tianxiang Sun, Pengfei Liu, and Xipeng Qiu. 2021. Does syntax matter? A strong baseline for aspect-based sentiment analysis with roberta. In Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT 2021, Online, June 6-11, 2021, pages 1816–1829. Association for Computational Linguistics.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. BERT: pre-training of deep bidirectional transformers for language understanding. CoRR, abs/1810.04805.

Timothy Dozat and Christopher D. Manning. 2017. Deep biaffine attention for neural dependency parsing. In 5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Conference Track Proceedings.

Jason Eisner. 1997. Three new probabilistic models for dependency parsing: An exploration. CoRR, cmplg/9706003.

Michael Kampffmeyer, Yinbo Chen, Xiaodan Liang, Hao Wang, Yujia Zhang, and Eric P. Xing. 2019. Rethinking knowledge graph propagation for zeroshot learning. In IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2019, Long Beach, CA, USA, June 16-20, 2019, pages 11487–11496. Computer Vision Foundation / IEEE.

Thomas N. Kipf and Max Welling. 2016. Semi-supervised classification with graph convolutional networks. CoRR, abs/1609.02907.1

Svetlana Kiritchenko, Xiaodan Zhu, Colin Cherry, and Saif Mohammad. 2014. NRC-Canada-2014: Detecting aspects and sentiment in customer reviews. In Proceedings of the 8th International Workshop on Semantic Evaluation (SemEval 2014), pages 437–442, Dublin, Ireland. Association for Computational Linguistics.

Maria Pontiki, Dimitris Galanis, Haris Papageorgiou, Ion Androutsopoulos, Suresh Manandhar, Mohammad Al-Smadi, Mahmoud Al-Ayyoub, Yanyan Zhao, Bing Qin, Orphée De Clercq, Véronique Hoste, Marianna Apidianaki, Xavier Tannier, Natalia V. Loukachevitch, Evgeniy V. Kotelnikov, Núria Bel, Salud María Jiménez Zafra, and Gülsen Eryigit. 2016. Semeval-2016 task 5: Aspect based sentiment analysis. In Proceedings of the 10th International Workshop on Semantic Evaluation, SemEval@NAACL-HLT 2016, San Diego, CA, USA, June 16-17, 2016, pages 19–30. The Association for Computer Linguistics. 205

Maria Pontiki, Dimitris Galanis, Haris Papageorgiou, Suresh Manandhar, and Ion Androutsopoulos. 2015. Semeval-2015 task 12: Aspect based sentiment analysis. In Proceedings of the 9th International Workshop on Semantic Evaluation, SemEval@NAACL-HLT 2015, Denver, Colorado, USA, June 4-5, 2015, pages 486–495. The Association for Computer Linguistics.

Maria Pontiki, Dimitris Galanis, John Pavlopoulos, Harris Papageorgiou, Ion Androutsopoulos, and Suresh Manandhar. 2014. SemEval-2014 task 4: Aspect based sentiment analysis. In Proceedings of the 8th International Workshop on Semantic Evaluation (SemEval 2014), pages 27–35, Dublin, Ireland. Association for Computational Linguistics.

Kai Sun, Richong Zhang, Samuel Mensah, Yongyi Mao, and Xudong Liu. 2019. Aspect-level sentiment analysis via convolution over dependency tree. In Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing, EMNLP-IJCNLP 2019, Hong Kong, China, November 3-7, 2019, pages 5678–5687. Association for Computational Linguistics.

Hao Tang, Donghong Ji, Chenliang Li, and Qiji Zhou. 2020. Dependency graph enhanced dual-transformer structure for aspect-based sentiment classification. In Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, ACL 2020, Online, July 5-10, 2020, pages 6578–6588. Association for Computational Linguistics.

Yuanhe Tian, Guimin Chen, and Yan Song. 2021a. Aspect-based sentiment analysis with type-aware graph convolutional networks and layer ensemble. In Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT 2021, Online, June 6-11, 2021, pages 2910–2922. Association for Computational Linguistics.

Yuanhe Tian, Guimin Chen, and Yan Song. 2021b. Enhancing aspect-level sentiment analysis with word dependencies. In Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume, EACL 2021, Online, April 19 - 23, 2021, pages 3726–3739. Association for Computational Linguistics.

Kai Wang, Weizhou Shen, Yunyi Yang, Xiaojun Quan, and Rui Wang. 2020. Relational graph attention network for aspect-based sentiment analysis. In Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, ACL 2020, Online, July 5-10, 2020, pages 3229–3238. Association for Computational Linguistics.

Lichen Wang, Bo Zong, Yunyu Liu, Can Qin, Wei Cheng, Wenchao Yu, Xuchao Zhang, Haifeng Chen, and Yun Fu. 2021. Aspect-based sentiment classification via reinforcement learning. In IEEE International Conference on Data Mining, ICDM 2021, Auckland, New Zealand, December 7-10, 2021, pages 1391–1396. IEEE.

Chen Zhang, Qiuchi Li, and Dawei Song. 2019. Aspect-based sentiment classification with aspect-specific graph convolutional networks. In Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing, EMNLP-IJCNLP 2019, Hong Kong, China, November 3-7, 2019, pages 4567–4577. Association for Computational Linguistics.

Jie Zhou, Jimmy Xiangji Huang, Qinmin Vivian Hu, and Liang He. 2020. SK-GCN: modeling syntax and knowledge via graph convolutional network for aspect-level sentiment classification. Knowl. Based Syst., 205:106292.

Yuxiang Zhou, Lejian Liao, Yang Gao, Zhanming Jie, and Wei Lu. 2021. To be closer: Learning to link up aspects with opinions. In Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing, EMNLP 2021, Virtual Event / Punta Cana, Dominican Republic, 7-11 November, 2021, pages 3899–3909. Association for Computational Linguistics.

Long Jiang, Mo Yu, Ming Zhou, Xiaohua Liu, and Tiejun Zhao. 2011. Target-dependent Twitter sentiment classification. In Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies, pages 151–160, Port-land, Oregon, USA. Association for Computational Linguistics.

DuyTin Vo and Yue Zhang. 2015. Deep learning for event-driven stock prediction. In Proceedings of IJCAI, Buenos Aires, Argentina.

A Structure-Aware Argument Encoder for Literature Discourse Analysis

Yinzi Li[1,†], **Wei Chen**[1,†], **Zhongyu Wei**[1,2,*], **Yujun Huang**[1], **Chujun Wang**[3]
Siyuan Wang[1], **Qi Zhang**[3], **Xuanjing Huang**[3] **and Libo Wu**[1]
[1]School of Data Science, Fudan University, China
[2]Research Institute of Intelligent Complex Systems, Fudan University
[3]Department of Environmental Science and Engineering, Fudan University, China
[3]School of Computer Science, Fudan University, China
{yinzili20,chenwei18,zywei,yjhuang19,chujunwang20}@fudan.edu.cn;
{wangsy18,qz,xjhuang,wulibo}@fudan.edu.cn;

Abstract

Existing research for argument representation learning mainly treats tokens in the sentence equally and ignores the implied structure information of argumentative context. In this paper, we propose to separate tokens into two groups, namely framing tokens and topic ones, to capture structural information of arguments. In addition, we consider high-level structure by incorporating paragraph-level position information. A novel structure-aware argument encoder is proposed for literature discourse analysis. Experimental results on both a self-constructed corpus and a public corpus show the effectiveness of our model. Resources are available at https://github.com/lemuria-wchen/SAE.

1 Introduction

With the growing amount of scientific literature, researchers pay increasing attention to developing computational methods for analyzing scientific literature (Kirschner et al., 2015; Stab and Gurevych, 2017; Green, 2018; Lauscher et al., 2018; Accuosto and Saggion, 2019), aiming to identify various components of arguments automatically (Abend et al., 2009; Judea and Strube, 2017; Lukin et al., 2017; Durmus and Cardie, 2018; Lugini and Litman, 2020). Existing research focuses on constructing annotated corpus and learning representation of sentences for literature discourse analysis. They tend to treat tokens in a sentence equally and ignore the implied structure information of argumentative context (Stab and Gurevych, 2014; Wachsmuth et al., 2016; Zhang et al., 2016; Lawrence and Reed, 2017).

Figure 1 shows two annotated abstracts of scientific literature, in which sentences are classified into four types, namely *background*, *method*, *result* and *conclusion*. We have some findings.

Abstract A:

Background : [This study investigates how does the Stern Review affect the Economics of Climate Change, …]
Method: [An approach to support climate policy evaluation, consisting in the Threshold 21 (T21) model, … is employed to analyze Ecuador's energy, …] …
Result: [Results indicate ... would reduce GHG emissions in the electric power sector, ...] …
Conclusion: [Finally, authors find …positive economic increases energy consumption, ...]

Abstract B:

Background : [Combustion is a source of atmospheric aerosols, including organic carbon (OC) and black carbon (BC) .]
Method: [Pilot-scale coal combustor was used to investigate the OC and BC aerosol formation under combustion conditions.]
Result: [It was found that BC aerosol formation was sensitive to the fuel-oxidizer equivalence ratio.] ……
Conclusion: [The BC and OC aerosol formation indicate that the formation pathways of OC aerosol ...]

Figure 1: Two samples of annotated abstracts. Framing tokens are highlighted in blue font and blue dotted lines and the rest are topic tokens. The division rule of tokens can be referred in section 3.

First, tokens in sentences can be divided into two groups, i.e., topic words and framing words. Topic words provide the fundamental knowledge of this argument while framing words organize the expression. Second, the same argument components often use similar framing structure in discourses across topics. For example, structures like '... *is employed / investigated to* ...' usually appear in the *method* section. Third, argument components are sensitive to their positions. For example, *background* almost always comes before *method* part and *conclusion* usually locates at the end. Motivated by these findings, we propose a structure-aware argument encoder (SAE) based on the transformer to enhance the literature discourse analysis. Experimental results show the effectiveness of our proposed model both on a self-constructed corpus and a public corpus.

Our contributions are two-fold: (1) we propose a novel transformer encoder that considers topic tokens and framing tokens separately to incorpo-

* Corresponding author
† Equal contributions

Proceedings of the 29th International Conference on Computational Linguistics, pages 7093–7098
October 12–17, 2022.

rate the structure of an argument for its representation learning; (2) we construct a large scale annotated corpus of scientific literature across different topics as a new benchmark.

2 CCSA Corpus

There are several public annotated corpora for scientific literature analysis (Liakata et al., 2010; Kirschner et al., 2015; Sateli and Witte, 2015; Ronzano and Saggion, 2015; Dasigi et al., 2017; Accuosto and Saggion, 2019; Achakulvisut et al., 2019), most of which focus on medicine and computer science. However, as a highly controversial research area, climate science is less explored. To bridge the gap, we create the **C**limate **C**hange **S**cientific **A**rgumentation (CCSA) corpus. Table 1 shows a comparison between the CCSA corpus and several annotated corpora for scientific literature, and our CCSA corpus has the advantages of corpus size and inter-annotator agreement.

Data Source We search for *climate change* in the ISI Web of Science[†] 2020 and collect all the retrieved papers published from 2000 to 2020 as the source. The domain of climate change covers a wide range of topics. In order to balance various sub-focus, we use Latent Dirichlet Allocation (LDA) (Blei et al., 2003) to cluster all papers into different topics and choose similar number of publications from each group for annotation. We set 40 topics in LDA, and selected 8 representative, data-rich ones with topic words such as Air Pollutants, Carbon Emission Policy, Global Warming and so on.

Annotation Scheme We treat each sentence in the abstract as an argument component and classify them into four types. C1) ***Background*** explains the motivation and background. C2) ***Method*** presents experimental procedures. C3) ***Result*** includes data, facts, and descriptions of outcomes, without any subjective speculations or judgements. C4) ***Conclusion*** gives opinions of the author. Invalid sentences, such as copyright information, are labeled as *other* types.

Annotation Process Undergraduate students are hired for the annotation, about half of them are majored in environmental sciences. We develop a web-based annotation platform and each abstract is annotated by three annotators. The inter-annotator agreement for argument type annotation is 0.68 in terms of Fleiss's Kappa coefficient (Falotico and Quatto, 2015), which shows a moderate consistency. The final result is determined by majority votes. If there is a disagreement, the label will be determined by the annotator with the greatest confidence [†]. There were 2,018 abstracts and 18,832 valid argument components in CCSA corpus. Table 2 depicts the distribution of the argument type. Sentences of "other" type accounted for only about 0.5% in our corpus, so we ignore them.

3 Structure-aware Argument Encoder

In order to incorporate the structure information of an argument, we propose a novel structure for argument representation learning, named **S**tructure-aware **A**rgument **E**ncoder (SAE). The main component of SAE is a transformer structure with multiple attention mechanisms to capture interactions between different groups of tokens. The overall architecture is shown in Figure 2.

Argument Structure In scientific discourse, some technical terms may introduce some noise to the identification of the argument structure. In SAE, we divide the tokens in each sentence into framing tokens and topic tokens. ***Framing Token*** contains the structural information in the argument component. ***Topic Token*** contains the topic information in the argument component, such as technical terms in the research field.

The sentence is tokenized and tagged with POS (Part-of-speech) using NLTK (Hardeniya et al., 2016). We regard *Singular Noun (NN)*, *Plural Noun (NNS)*, *Singular Proper Noun (NNP)* and *Plural Proper Noun (NNPS)* as topic tokens and others as framing tokens. Any method can be adopted for token division, not just POS tagging.

Argumentative Attention Mechanism To utilize the information of the token types, in addition to ***self-attention***, our argumentative attention mechanism contains two extra attention patterns. ***Internal-attention*** takes effect among tokens of the same type, i.e., framing tokens attend to framing tokens, and so do topic tokens. Internal-attention is utilized to explore the internal influence of tokens of the same type. ***External-***

[†] http://isiknowledge.com/

[†] We have a self-annotated corpus subset (gold), and we measured the annotator confidence by calculating the consistency between annotators and the subset.

Corpus	Area	Content	Size	Type	IAA
DiGAT (Kirschner et al., 2015)	Education	Full-text	24	-	0.50 (F1)
Gold Standard (Sateli and Witte, 2015)	Computer Science	Full-text	30	2	-
Dr. Inventor (Ronzano and Saggion, 2015)	Computer Science	Full-text	40	5	0.66 (Kappa)
PubMed-SciDT (Dasigi et al., 2017)	Medical	Experiment	75	7	-
Biomedical-Claims (Achakulvisut et al., 2019)	Medical	Abstract	1,500	2	0.63 (Kappa)
CCSA (ours)	Climate Science	Abstract	**2,018**	4	**0.68** (Kappa)

Table 1: Comparison between the CCSA corpus and other human-annotated corpora for scientific literature. It is worth noting that the "size" columns are not all comparable due to different statistical calibers.

	Bg.	Meth.	Res.	Con.
Number	3,939	4,306	5,962	4,625
Proportion	20.9%	22.9%	31.7%	24.5%

Table 2: Distribution of different argument component types. Bg., Meth., Res., Con. are the abbreviations of background, method, result and conclusion.

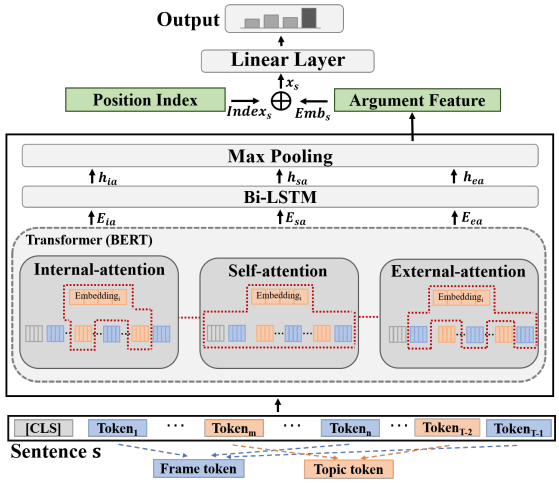

Figure 2: The overall architecture of our Structure-aware Argument Encoder (SAE).

attention takes effect among tokens with different types, i.e., framing tokens attend to topic tokens, and topic tokens attend to framing tokens. External-attention is expected to explore the influence between tokens with different types.

Argument Representation Suppose the **input** s is a sentence with T tokens $s = [t_0, t_1, ..., t_{T-1}]$, the structure-aware argument encoder is first adopted to obtain the contextual **token embeddings** E based on argumentative attention:

$$E = [e^0, ..., e^{T-1}] = F(t_0, ..., t_{T-1}) \quad (1)$$

where $F(\cdot)$ is transformer encoder. We can obtain E_{ia}, E_{ea} and E_{sa} through $F_{ia}(\cdot)$, $F_{ea}(\cdot)$ and $F_{sa}(\cdot)$, which are transformer encoders with internal-attention, external-attention and self-attention. The parameters of the three transformer encoders are shared, but due to their different attention mechanisms, different features can be extracted. The token embeddings E are then fed into a token-level bidirectional LSTM layer, and the last hidden states from both directions are concatenated as the **sentence embedding** h:

$$\begin{aligned}[[\overrightarrow{h^0}; \overleftarrow{h^0}], ...[\overrightarrow{h^{T-1}}; \overleftarrow{h^{T-1}}]] &= \text{Bi-LSTM}(E) \\ h &= [\overrightarrow{h^{T-1}}; \overleftarrow{h^{T-1}}]\end{aligned} \quad (2)$$

We obtain h_{ia}, h_{ea} and h_{sa} with E_{ia}, E_{ea} and E_{sa} respectively, and further use a max-pooling layer to extract the **argument feature** Emb_s of sentence s:

$$Emb_s = \text{max-pooling}(h_{ia}, h_{ea}, h_{sa}) \quad (3)$$

Argument components are sensitive to their positions and the position information is an important feature for its type. We use the standardized index of the sentence in the abstract as an additional **position feature** concatenated to argument feature as the final **argument representation**:

$$x_s = [Emb_s; Index_s] \quad (4)$$

The predicted probability distribution $p(y|s)$ of argument categories is obtained after x_s is fed into a multilayer perceptron (MLP) layer.

4 Experiment

Experimental Setup We focus on the task of **argument component identification** which aims to predict the argument type of argument component (sentences). We conduct experiments on our CCSA corpus. To demonstrate that our SAE is domain-independent, we also conduct experiments on another scientific publication abstract corpus *biomedical-claims*[†] (Achakulvisut et al.,

[†] https://github.com/titipata/detecting-scientific-claim

2019). It annotates whether a sentence is a *claim*, whose setting is similar to CCSA.

For CCSA corpus, we take the macro F1 as the evaluation metric of this multi-classification problem, and the F1 score of each sentence type on the test set is also reported. For biomedical-claims corpus, we report precision, recall and F1 score on the test set. The experiment configuration details are shown in A.1.

To prove our argumentative attention mechanism has the advantage of modeling topic tokens and framing tokens, we also implement a variant of our SAE model that utilizes token types in a simpler way, namely ***parameterized SAE (p-SAE)***. Specifically, we initialize a learnable embedding layer for framing tokens and topic tokens instead of argumentative attention mechanism, and add them to token embeddings as input, similar to the segment embedding in BERT. Compared with SAE, p-SAE models topic tokens and framing tokens by initializing a trainable parameterized embedding layer at model input.

Overall Performance For CCSA corpus, we compare our SAE and p-SAE with following baselines: BERT (Devlin et al., 2019), bidirectional LSTM (Bi-LSTM) (Graves et al., 2013) and Sentence Encoder (SE), which contains a BERT layer and on top of it, a Bi-LSTM layer. Compared with SAE and p-SAE, SE is a combination of BERT and Bi-LSTM without the information of token types. For biomedical-claims corpus, we present the state-of-the-art model based on transfer learning (TL-CRF) in the original paper as baseline (Achakulvisut et al., 2019).

Table 3 shows main results of CCSA corpus, which indicate that our SAE achieves competitive macro F1 score on the argument component identification task. It is worth noting that SAE improves the identification of *conclusion* part most, because the *conclusion* is the most argumentative part, which shows that our model has excellent effect in exploring argumentative structure. Similarly, results of scientific publication corpus are shown in Table 4 indicating that the model has better performance in identifying scientific claims.

Ablation Study Table 3 shows the results of ablation study. Internal-attention affects *conclusion* part most and external-attention affects *method* part most, which shows that argumentative texts, such as *conclusion* part focus more on the organization of structure. However, the structure of *method* part needs to be combined with some professional terms through external-attention. The macro F1 score of *conclusion* part drops down most without internal-attention, which shows the effectiveness of modeling topic tokens and framing tokens separately in argumentative structure.

Model	Bg.	Meth.	Res.	Con.	Macro F1
Bi-LSTM	59.6	79.4	59.6	55.0	59.6
BERT	69.6	83.3	78.0	57.5	72.1
SE	69.1	84.3	78.2	57.9	72.4
p-SAE	**72.9**	85.0	**78.6**	63.1	74.9
SAE	72.3	**86.2**	77.9	**65.7**	**75.5**
Ablation study					
SAE w/o Ia	-1.3	-1.5	-0.9	**-2.3**	-1.5
SAE w/o Ea	-0.2	**-3.5**	+1.1	-1.1	-0.8

Table 3: Performance on test set of CCSA corpus. Bg., Meth., Res., Con. are the abbreviations of *background*, *method*, *result* and *conclusion*. Ia and Ea represents *Internal-attention* and *External-attention*.

Model	Precision	Recall	F1
TL-CRF	86.6	72.7	79.0
Bi-LSTM	82.8	63.4	66.6
BERT	84.7	80.8	82.5
SE	84.8	81.9	83.2
p-SAE	84.5	83.1	83.8
SAE	**86.6**	**83.6**	**85.0**
Domain adaptation			
SE (CCSA)	80.2	78.2	79.1
p-SAE (CCSA)	**83.3**	77.0	79.5
SAE (CCSA)	81.8	**78.9**	**80.2**

Table 4: Performance on test set of biomedical-claims corpus (Achakulvisut et al., 2019). TL-CRF is the SOTA result in the original paper.

Domain Adaptation We apply the model trained with CCSA on the test set of biomedical-claims to evaluate the ability of generalization of SAE. Since the sentence types of the two corpora are different, we do label mapping as follows: the predicted *conclusion* label is converted to claim, and the others are converted to non-claim. We migrate three models, namely SE, p-SAE and SAE, and the results are shown in Table 4.

Although the research fields and categories involved in the two scientific literature corpora are different, our model still shows strong transfer capability without any training. Among them, both p-SAE and SAE that consider the argument structure outperform SE. SAE with multiple attention mechanisms performs better than p-SAE, which

also illustrates the advantages of our proposed SAE in terms of domain adaptation.

5 Conclusion

In this paper, we propose a structure-aware argument encoder (SAE) that considers token types in the sentence and separate tokens into two groups, namely topic tokens and framing tokens. Multiple argumentative attention mechanisms are utilized to capture internal and external interactions among different groups of tokens. Experimental results on a self-constructed corpus and another publicly corpus of scientific literature show the effectiveness of our model.

Acknowledgements

This work is partially supported by Natural Science Foundation of China (No.6217020551, 71991471), National Social Science Foudation of China (No.20ZDA060), Science and Technology Commission of Shanghai Municipality Grant (No.20dz1200600, 21QA1400600, 21511101000) and Zhejiang Lab (No. 2019KD0AD01).

Ethical Statement

In this paper, different ethical restrictions deserve discussion.

All data in our self-constructed corpus are available online and other corpora in this paper are publicly available sources. We strictly followed the platform's policies and rules when crawling data from web platforms. We did not employ any author-specific information in our research.

The reward for annotating an article is determined by the number of sentences in the abstract. We pay \$0.03 for each sentence, averaging about \$0.24 per article. The price of annotating abstract is not so much because of our limited budget, but we give additional bonuses and internship certificates to student annotators who complete the assignment. All annotators are people who are willing to participate and over the age of 18. We have an online chat group for making announcements and answering questions.

Our corpus may includes some bias, such as political bias and social bias, and our model might have inherited some forms of these bias. In order to limit these bias as much as possible, we filter controversial articles and removed data with offensive information when possible.

References

Omri Abend, Roi Reichart, and Ari Rappoport. 2009. Unsupervised argument identification for semantic role labeling. In *Proceedings of the Joint Conference of the 47th Annual Meeting of the ACL and the 4th International Joint Conference on Natural Language Processing of the AFNLP*, pages 28–36, Suntec, Singapore. Association for Computational Linguistics.

Pablo Accuosto and Horacio Saggion. 2019. Discourse-driven argument mining in scientific abstracts. In *Natural Language Processing and Information Systems*, pages 182–194.

Titipat Achakulvisut, Chandra Bhagavatula, Daniel Acuna, and Konrad Kording. 2019. Claim extraction in biomedical publications using deep discourse model and transfer learning.

David M. Blei, Andrew Y. Ng, and Michael I. Jordan. 2003. Latent dirichlet allocation. In *Proceedings of the 14th International Conference on Neural Information Processing Systems: Natural and Synthetic*, NIPS'01, page 601–608, Cambridge, MA, USA. MIT Press.

Pradeep Dasigi, Gully APC Burns, Eduard Hovy, and Anita de Waard. 2017. Experiment segmentation in scientific discourse as clause-level structured prediction using recurrent neural networks. *arXiv preprint arXiv:1702.05398*.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, Minneapolis, Minnesota. Association for Computational Linguistics.

Esin Durmus and Claire Cardie. 2018. Exploring the role of prior beliefs for argument persuasion. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 1035–1045, New Orleans, Louisiana. Association for Computational Linguistics.

Rosa Falotico and Piero Quatto. 2015. Fleiss' kappa statistic without paradoxes. *Quality & Quantity*, 49(2):463–470.

A. Graves, N. Jaitly, and A. Mohamed. 2013. Hybrid speech recognition with deep bidirectional lstm. In *2013 IEEE Workshop on Automatic Speech Recognition and Understanding*, pages 273–278.

Nancy Green. 2018. Proposed method for annotation of scientific arguments in terms of semantic relations and argument schemes. In *Proceedings of the 5th Workshop on Argument Mining*, pages 105–110,

Brussels, Belgium. Association for Computational Linguistics.

Nitin Hardeniya, Jacob Perkinsand Deepti Chopra, Nisheeth Joshi, and Iti Mathur. 2016. *Natural Language Processing: Python and NLTK*. Packt Publishing.

Alex Judea and Michael Strube. 2017. Event argument identification on dependency graphs with bidirectional LSTMs. In *Proceedings of the Eighth International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 822–831, Taipei, Taiwan. Asian Federation of Natural Language Processing.

Christian Kirschner, Judith Eckle-Kohler, and Iryna Gurevych. 2015. Linking the thoughts: Analysis of argumentation structures in scientific publications. pages 1–11.

Anne Lauscher, Goran Glavaš, Simone Paolo Ponzetto, and Kai Eckert. 2018. Investigating the role of argumentation in the rhetorical analysis of scientific publications with neural multi-task learning models. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 3326–3338, Brussels, Belgium. Association for Computational Linguistics.

John Lawrence and Chris Reed. 2017. Mining argumentative structure from natural language text using automatically generated premise-conclusion topic models. In *Proceedings of the 4th Workshop on Argument Mining*, pages 39–48, Copenhagen, Denmark. Association for Computational Linguistics.

Maria Liakata, Simone Teufel, Advaith Siddharthan, and Colin Batchelor. 2010. Corpora for the conceptualisation and zoning of scientific papers.

Luca Lugini and Diane Litman. 2020. Contextual argument component classification for class discussions. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 1475–1480, Barcelona, Spain (Online). International Committee on Computational Linguistics.

Stephanie Lukin, Pranav Anand, Marilyn Walker, and Steve Whittaker. 2017. Argument strength is in the eye of the beholder: Audience effects in persuasion. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 1, Long Papers*, pages 742–753, Valencia, Spain. Association for Computational Linguistics.

Francesco Ronzano and Horacio Saggion. 2015. Dr. inventor framework: Extracting structured information from scientific publications. In *International conference on discovery science*, pages 209–220. Springer.

Bahar Sateli and René Witte. 2015. Semantic representation of scientific literature: bringing claims, contributions and named entities onto the linked open data cloud. *PeerJ Computer Science*, 1:e37.

Christian Stab and Iryna Gurevych. 2014. Identifying argumentative discourse structures in persuasive essays. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 46–56, Doha, Qatar. Association for Computational Linguistics.

Christian Stab and Iryna Gurevych. 2017. Parsing argumentation structures in persuasive essays. *Computational Linguistics*, 43(3):619–659.

Henning Wachsmuth, Khalid Al-Khatib, and Benno Stein. 2016. Using argument mining to assess the argumentation quality of essays. In *Proceedings of COLING 2016, the 26th International Conference on Computational Linguistics: Technical Papers*, pages 1680–1691, Osaka, Japan. The COLING 2016 Organizing Committee.

Justine Zhang, Ravi Kumar, Sujith Ravi, and Cristian Danescu-Niculescu-Mizil. 2016. Conversational flow in Oxford-style debates. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 136–141, San Diego, California. Association for Computational Linguistics.

A Appendix

A.1 Experiment Details

We use BERT-base model (bert-base-uncased) to initialize the parameters of the transformer encoder, and the parameters of bidirectional LSTM (Bi-LSTM) are randomly initialized. All models are trained on 4 Nvidia GeForce RTX 2080 Ti GPUs with the same random seed. The batch size is 32, the dropout rate is 0.1, the learning rate is 1e-5, the hidden size for the Bi-LSTM layers is 200, the max length of a sentence is 100. We split our CCSA corpora and biomedical-claims corpus into training, validation and test sets with the proportion of $6:2:2$ respectively. The best performing model on the validation set are evaluated on the test set.

Mere Contrastive Learning for Cross-Domain Sentiment Analysis

Yun Luo [1,2], **Fang Guo** [2], **Zihan Liu** [2], **Yue Zhang** [2,3]
[1] School of Computer Science And Technology, Zhejiang University, Hangzhou 310024, P.R. China.
[2] School of Engineering, Westlake University, Hangzhou, China.
[3] Institute of Advanced Technology, Westlake Institute for Advanced Study, Hangzhou, China.
{luoyun, guofang, liuzihan, zhangyue}@westlake.edu.cn

Abstract

Cross-domain sentiment analysis aims to predict the sentiment of texts in the target domain using the model trained on the source domain to cope with the scarcity of labeled data. Previous studies are mostly cross-entropy-based methods for the task, which suffer from instability and poor generalization. In this paper, we explore contrastive learning on the cross-domain sentiment analysis task. We propose a modified contrastive objective with in-batch negative samples so that the sentence representations from the same class can be pushed close while those from the different classes become further apart in the latent space. Experiments on two widely used datasets show that our model can achieve state-of-the-art performance in both cross-domain and multi-domain sentiment analysis tasks. Meanwhile, visualizations demonstrate the effectiveness of transferring knowledge learned in the source domain to the target domain and the adversarial test verifies the robustness of our model.

1 Introduction

Sentiment classification (Liu, 2012) has been widely studied by both industry and academia (Blitzer et al., 2007; Li et al., 2013; Yu and Jiang, 2016). For example, the sentiment is positive towards the text *'The book is exactly as pictured/described. Cute design and good quality'*. Early methods rely on labeled data to train models on a specific domain (e.g. DVD reviews, book reviews, and so on), which are labor-intensive and time-consuming (Socher et al., 2013). To address this issue, cross-domain sentiment analysis attracts increasing attention.

Various neural models have been proposed for cross-domain sentiment analysis in recent years (Blitzer et al., 2007; Li et al., 2013; Yu and Jiang, 2016; Zhang et al., 2019; Zhou et al., 2020a). Most methods focus on making the model unable to distinguish the data from which domain by adversarial training or data argumentation, in order to transfer knowledge from source domains to target domains (Du et al., 2020; Liu et al., 2017; Qu et al., 2019; Yang et al., 2022) and some attempt to learn domain-specific knowledge (Zhou et al., 2020a; Liu et al., 2018; Wang et al., 2019). Pre-trained language models (Kenton et al., 2019; Radford et al., 2019; Lewis et al., 2020) have achieved stronger performance compared with previous random initialized models such as LSTM (Long Short-term Memory) in cross-domain tasks. The state-of-the-art models on cross-domain sentiment analysis, such as BERT-DAAT (Du et al., 2020), use unlabeled data to continually train the pre-trained model BERT to transfer knowledge besides adversarial training.

In the representation aspect for the cross-domain

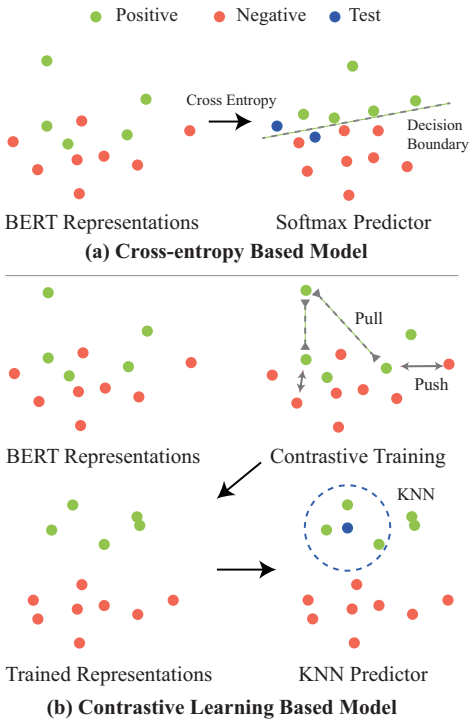

Figure 1: The architectures for cross-entropy-based model and the contrastive- learning-based model.

Proceedings of the 29th International Conference on Computational Linguistics, pages 7099–7111
October 12–17, 2022.

sentiment analysis, there are two key requirements for the representations of sentences: (1) sentence representations in the same domain with the different/the same sentiments should be far from/close to each other; (2) sentence representations of different domains with the same labels should be close. Existing methods are mostly softmax-based method by optimizing cross-entropy loss to achieve the requirements (illustrate in Figure 1 (a)), which suffers from instability across different runs (Zhang et al., 2020; Dodge et al., 2020), poor generalization performance (Liu et al., 2016; Cao et al., 2019), reduction of prediction diversity (Cui et al., 2020), lack of robustness to noisy labels (Zhang and Sabuncu, 2018; Sukhbaatar et al., 2015), or adversarial examples (Elsayed et al., 2018; Nar et al., 2019), especially when supervised data are limited in the cross-domain settings.

To address the above shortcomings, we explore the effectiveness of contrastive learning on the task. Contrastive learning is a similarity-based training strategy, which aims to push the representations from the same class close and those from the different class further apart (Chen et al., 2020; Gao et al., 2021; Neelakantan et al., 2022; Gao et al., 2021). Contrastive learning has been shown effective in solving the problem of anisotropy (Gao et al., 2019), and it has good generalization and robustness (Li et al., 2021; Gao et al., 2021; Gunel et al., 2020; Khosla et al., 2020). Previous work relies mostly on pre-training for representations (Chen et al., 2020; Neelakantan et al., 2022) or multi-task training for semantic textual similarity (Gao et al., 2021), classification (Li et al., 2021; Gunel et al., 2020) and so on, but little work uses mere contrastive learning for supervised tasks. Intuitively, the optimization of contrastive learning is effective in satisfying the requirements of cross-domain sentiment analysis.

We explore **CO**ntrastive learning on **BE**RT (COBE) by a modified contrastive loss function with the in-batch negative method on cross-domain sentiment analysis tasks. In the mini-batch, the samples with the same labels are treated as positive pairs, and those with different labels are treated as negative pairs. As shown in Figure 1, the optimization procedure aims to tighten the cluster of samples with the same labels, and push away samples with different labels. After training, the representations of training data and their labels are saved offline as a knowledge base for classification. When evaluating the model, a kNN (k-Nearest Neighbors) predictor is used to predict the sentiment of test data, i.e. we search for the k data with the largest cosine similarity in the knowledge base and vote for the final prediction using their labels.

Experiments on two widely used datasets (the cross-domain Amazon dataset (Blitzer et al., 2007) and FDU-MTL (Liu et al., 2017)) show that our model can achieve the state-of-the-art performance in both the cross-domain setting and the multi-domain setting sentiment classification. Visualizations also demonstrate the effectiveness of transferring knowledge learned in the source domain to the target domain. To our knowledge, we are the first to show that contrastive learning outperforms cross-entropy-based models on cross-domain sentiment analysis for both performance and robustness. The code has been released in https://github.com/LuoXiaoHeics/COBE.

2 Related Work

Cross-domain sentiment analysis. Due to the heavy cost of obtaining large quantities of labeled data for each domain, many approaches have been proposed for cross-domain sentiment analysis (Blitzer et al., 2007; Li et al., 2013; Yu and Jiang, 2016; Zhang et al., 2019; Zhou et al., 2020a). Ziser and Reichart (2018) and Li et al. (2018a) propose to capture the pivots that are useful for both source domains and target domains. Ganin et al. (2016) propose to use adversarial training with a domain discriminator to learn domain-invariant information, which is one type of solutions for the cross-domain sentiment analysis task (Du et al., 2020; Liu et al., 2017; Qu et al., 2019; Zhou et al., 2020a). These adversarial training methods try to confuse the model unable to classify the data from which domain, transferring the knowledge from source domains to target domains. Besides, Liu et al. (2018) and Cai and Wan (2019) attempt to learn domain-specific information for the different sentiment expressions on different domains. However, these studies rely on minimizing the cross-entropy loss, resulting in the issue of unstable fine-tuning and poor generalization (Gunel et al., 2020; Li et al., 2021; Zhang et al., 2020; Dodge et al., 2020).

Contrastive Learning. Contrastive learning has been widely used in unsupervised learning (Chen et al., 2020; Jing et al., 2021; Wang and Isola, 2020; Khosla et al., 2020; Gao et al., 2021; Neelakantan et al., 2022). Radford et al. (2019) propose

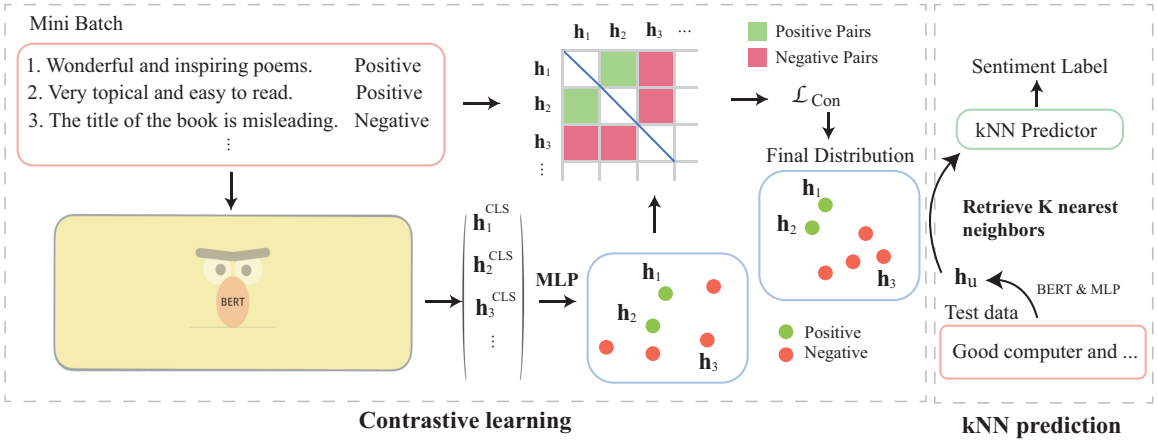

Figure 2: The framework of our contrastive learning for cross-domain sentiment analysis.

to use contrastive learning to learn the representations of both text and images through raw data in unsupervised method, which achieves strong performance on zero-shot task. Neelakantan et al. (2022) propose to use contrastive learning to obtain sentence and code representations and achieve strong performance on downstream tasks such as sentence classification and text search. Wang and Isola (2020) further identify the key properties for contrastive learning as (1) alignment (closeness) of features from positive pairs and (2) uniformity of induced distribution of representations. Gao et al. (2021) uses contrastive learning to learn the sentence representations and theoretically prove that contrastive learning can solve the anisotropy problem (the learned embeddings occupy a narrow cone in the vector space), which limits the expressiveness of representations. It also achieves better results for the semantic textual similarity task using a supervised dataset of natural language inference. Our model differs from the above studies in that we consider contrastive learning in supervised tasks, which uses golden labels to obtain positive/negative pairs for training.

Recently, some studies attempt to incorporate contrastive learning into cross-entropy-based methods by adding InfoNCE loss (Gunel et al., 2020; Li et al., 2021), which aims to solve the shortcomings of cross-entropy loss. Gunel et al. (2020) propose a new SCL loss based on InfoNCE loss to boost the stability and robustness of fine-tuning pre-trained language models. Subsequently, Li et al. (2021) attempt to incorporate kNN predictors to enhance the generalization of prediction in few-shot tasks, using both cross-entropy loss and SCL loss. The above work is similar to ours in making use of contrastive loss for classification. However, the difference is that we do not use a standard cross-entropy loss, but rely solely on vector space similarity losses for achieving cross-domain classification. To our knowledge, we are the first to conduct sentiment classification without using a cross-entropy loss in natural language processing.

3 Method

Formally, the training data consists of $\{(S_i, Y_i)\}_{i=1}^{N}$, where $S_i = [s_1, s_2, ..., s_l]$ is a set of review text, and $Y_i \in \{0, 1\}$ are the corresponding sentiment labels. The model framework is shown in Figure 2. To present our model – **CO**ntrastive learning on **BE**RT (COBE), We first introduce the prediction of sentiment labels using representations based on kNN (Section 3.1), and then describe the training objective to obtain effective representations using contrastive learning (Section 3.2). For comparison, we also describe the standard cross-entropy baseline, named BERT-CE, and a version that adopts adversarial training, named BERT-adv.

3.1 Model

We concatenate the review text S_i with special tokens $[CLS]$ and $[SEP]$ as our model input $X_i = [CLS]\ S_i\ [SEP]$, which is fed into BERT model to obtain the hidden states. The hidden state of $[CLS]$ from the last layer of BERT is considered as the representation of the input sequence:

$$\mathbf{h}_i^{CLS} = BERT(X_i)[CLS] \quad (1)$$

BERT-CE and BERT-adv baselines: After obtaining the sentence representation $\mathbf{h}_i^{CLS}$ of input X_i, an MLP (Multi Layer Perceptron) layer project

it to the label space and a softmax layer is adopted to calculate the probability distribution on the labels:

$$p_{ce} = Softmax(MLP(\mathbf{h}_i^{CLS})). \quad (2)$$

The label with the largest probability is adopted as the prediction result.

COBE: Our model uses the same representation of Eq(1), but adopts a kNN predictor to classify the labels. An MLP layer is then adopted for the dimension reduction:

$$\mathbf{h}_i = MLP(\mathbf{h}_i^{CLS}). \quad (3)$$

To predict the sentiment label of a review text S_u, we calculate the cosine similarity of the sentence representation $\mathbf{h}_u$ with the sentence representations of the training data:

$$sim(\mathbf{h}_u, \mathbf{h}_i) = \frac{\mathbf{h}_u \cdot \mathbf{h}_i}{||\mathbf{h}_u|| \cdot ||\mathbf{h}_i||}, \quad (4)$$

where $\mathbf{h}_i$ is the sentence representation of training data X_i.

We retrieve the k training data whose cosine similarity with $\mathbf{h}_u$ are the largest. We denote the k nearest neighbors as $(\mathbf{h}_i, Y_i) \in \mathcal{K}_u$. The retrieved set is converted to a probability distribution over the labels by applying a softmax with temperature T to the similarity. Using the temperature $T > 1$ can flatten the distribution, and prevent over-fitting to the most similar searches (Khandelwal et al., 2020). The probability distribution on the labels can be expressed as follows:

$$p_k(Y'_u) \propto \sum_{(\mathbf{h}_i, Y_i) \in \mathcal{K}_u} \mathbb{1}_{Y'_u = Y_i} \cdot exp(\frac{sim(\mathbf{h}_u, \mathbf{h}_i)}{T}). \quad (5)$$

The label with the largest probability is regarded as the prediction result.

3.2 Training Objective

BERT-CE baseline: For the cross-entropy-based model, multi-label cross-entropy loss is adopted to optimize the model, which is formulated as follows:

$$\mathcal{L}_{cls} = -\frac{1}{M} \sum_{(X_i, Y_i)} Y_i log\ p_{ce}(Y_i), \quad (6)$$

BERT-adv baseline: Besides the cross-entropy loss, BERT-adv adds a domain discriminator (Du et al., 2020; Ganin et al., 2016) to the standard model and adopts adversarial training to transfer knowledge from source domains to target domains.

Given the sentence and its domain label (X_i, D_i), the representation $\mathbf{h}_i^{CLS}$ obtained in Eq(1) goes through an additional gradient reversal layer (GRL) (Ganin et al., 2016), which can be denoted as a 'pseudo-function' $D_\lambda(x)$. The GRL reverses the gradient by applying a negative scalar λ. The forward- and backward- behaviors can be described:

$$D_\lambda(x) = x, \ \frac{\partial D_\lambda(x)}{\partial x} = -\lambda I, \quad (7)$$

where λ is a hyper-paramter and I is the gradients calculated on $\mathbf{h}_i^{CLS}$ (but it is multiplied with $-\lambda$ to back-propagate). Then a linear layer project $\mathbf{h}_i^{CLS}$ to the label space and a softmax layer is adopted to calculate the distribution on domain labels:

$$p_d = Softmax(W_d \mathbf{h}_i^{CLS} + b_d), \quad (8)$$

where W_d and b_d are the learnable parameters. The training target is to minimize the cross-entropy for all data from the source and target domains (note that the data from target domains are unlabeled on sentiment) in order to make the model unable to predict the domain labels:

$$\mathcal{L}_{dom} = -\frac{1}{M} \sum_{(X_i, D_i)} D_i log\ p_d(D_i). \quad (9)$$

For BERT-adv, the training loss of sentiment classification (Eq.7) and domain classification (Eq.10) are jointly optimized:

$$\mathcal{L}_{adv} = \mathcal{L}_{cls} + \mathcal{L}_{dom} \quad (10)$$

COBE: The baselines adopt $\mathcal{L}_{cls}$ to tighten the representations of the same/different labels close (apart), and adopt $\mathcal{L}_{dom}$ to mix up the representations of different domains with the same label. However, COBE uses a single training objective of contrastive learning to achieve the both goals. We apply in-batch negatives (Yih et al., 2011; Sohn, 2016) to learn sentence representations through contrastive learning, which has been widely used in unsupervised representation learning (Radford et al., 2021; Jia et al., 2021). For each example in the mini-batch of M samples, we treat the other samples with different golden labels as negative pairs, and the samples with the same golden labels as positive pairs. For example in Figure 2, the sentence pair (1,2) is positive pairs, and the sentence

S → T	B→D	B→E	B→K	D→B	D→E	D→K	E→B	E→D	E→K	K→B	K→D	K→E	Avg
DANN	82.30	77.60	76.10	81.70	79.70	77.35	78.55	79.70	83.95	79.25	80.45	86.65	80.29
PBLM	84.20	77.60	82.50	82.50	79.60	83.20	71.40	75.00	87.80	74.20	79.80	87.10	80.40
HATN	86.10	85.70	85.20	86.30	85.60	86.20	81.00	84.00	87.90	83.30	84.50	87.00	85.10
ACAN	83.45	81.20	83.05	82.35	82.80	78.60	79.75	81.75	83.35	80.80	82.10	86.60	82.15
IATN	86.80	86.50	85.90	87.00	86.90	85.80	81.80	84.10	88.70	84.70	84.10	87.60	85.90
BERT-CE	88.96	86.15	89.05	89.40	86.55	87.53	86.50	87.95	91.60	87.55	87.95	90.45	88.25
BERT-CE*	55.40	56.55	54.05	55.10	57.25	53.75	55.50	56.00	55.55	52.30	52.75	54.15	54.86
BERT-adv	89.70	87.30	89.55	89.55	86.05	87.69	87.15	86.05	91.91	87.65	87.72	86.05	88.56
DAAT	89.70	89.57	90.75	90.86	89.30	87.53	**88.91**	**90.13**	93.18	87.98	**88.81**	91.72	90.12
COBE*	82.17	83.65	83.12	79.82	78.87	82.58	75.95	79.53	86.10	78.55	76.95	85.17	80.95
COBE (proposed)	**90.05**	**90.45**	**92.90**	**90.98**	**90.67**	**92.00**	87.90	87.87	**93.33**	**88.38**	87.43	**92.58**	**90.39**

Table 1: Results on the cross-domain Amazon dataset. BERT-CE* and COBE* refer to the models fixing the parameters of BERT, and only tuning the parameters of MLP layer. (B for the Books domain, D for the DVD domain, E for the Electronics domain, and K for the Kitchen domain, respectively.)

pairs (1,3) and (2,3) are negative pairs. For each review X_i we denote N_i^+ as the set of reviews with the same label of X_i in the mini-batch. Then the contrastive loss function can be defined as follows:

$$\mathcal{L}_{Con} = \sum_{i}^{M} -\frac{1}{M} log \frac{\sum_{k \in N_i^+} exp(sim(\mathbf{h}_i, \mathbf{h}_k)/\tau)}{\sum_{i \neq j}^{M} exp(sim(\mathbf{h}_i, \mathbf{h}_j)/\tau)} \quad (11)$$

where τ is a temperature hyper-parameter. The loss function can alleviate the negative effect of the situations where there is no positive pairs for any training instance in the batch.

The usage of in-batch negatives enables re-use of computation both in the forward and backward pass making training highly efficient.

4 Experiments

We conduct experiments on both the cross-domain settings (train models on source domains and test on another one) and the multi-domain settings (train and test models on the same domains). To verify the effectiveness of our model, we also visualize the representations (Section 4.3) and carry out further analysis such as model robustness (Section 4.4).

4.1 Settings

Datasets. We test our contrastive learning method on two widely used datasets, the cross-domain Amazon dataset, and the FDU-MTL dataset. The cross-domain Amazon dataset (Blitzer et al., 2007) contains 4 domains: Books (B), DVD (D), Electronics (E) and Kitchen (K). Each domain contains 2000 Amazon review samples. Following the setting of previous work (Ganin et al., 2016; Ziser and Reichart, 2018; Du et al., 2020), we test the model on 12 tasks. The model is trained on the source domain data and tested on the target domain data.

Furthermore, we also evaluate our model on FDU-MTL, which is an Amazon reviews dataset with data on 16 domains (Liu et al., 2017). The training set, development set, and test set are split in the original dataset, (the statistics are shown in Appendix A). We carry out experiments on the multi-domain setting, (i.e. train the model on the whole 16 domains, and evaluate the model on the test on the whole 16 domains), and on the 15-1 cross-domain setting (i.e. train the model on 15 domains, and test the model on the 1 domain left).

Baselines. For the cross-domain Amazon dataset, we compare our model with several strong baselines in cross-domain sentiment analysis: DANN (Ganin et al., 2016), PBLM (Ziser and Reichart, 2018), HATN (Li et al., 2018b), IATN (Qu et al., 2019), DAAT (Du et al., 2020), BERT-CE and BERT-CE* (* for fixing the BERT parameters). We adopt the results of baselines reported in Zhou et al. (2020b) and Du et al. (2020). We also adopt BERT-adv as our baselines introduced in Section 3.2.

On FDU-MTL, we compare our model with ASP (Liu et al., 2017), DSR-at (Zheng et al., 2018), DAEA and DAEA-B (DAEA-BERT) (Cai and Wan, 2019). The DAEA-B is regarded as the state-of-the-art model on FDU-MTL (excluding the model SentiX (Zhou et al., 2020b), which uses a large corpus (about 241 million reviews) to continually train BERT for sentiment tasks). Note that, it is unfair that previous studies do not adopt the BERT-CE model for multi-domain experiments for comparison. In this study, we also consider BERT-CE, and BERT-CE* on the multi-domain setting as baselines. For the multi-domain task, the objective of

adversarial training is redundant, thus we mainly compare COBE with baselines BERT-CE.

Implementation Details. We perform experiments using the official pre-trained BERT model provided by Huggingface[1]. We train our model on 1 GPU (Nvidia GTX2080Ti) using the Adam optimizer (Kingma and Ba, 2014). For the cross-domain Amazon dataset (FDU-MTL), the max sequence length for BERT is 256 (128), and the batch size M is 8 (32). The max sequence lengths are set in such values for comparison with previous models. The initial learning rate is 2e-5 (1e-4) for BERT-unfixed (BERT-fixed) models, and each model is trained for 20 epochs. The hyper-parameters of temperatures τ is 0.05, T is 5, and the number of nearest neighbors k is 3 (without losing generality, we do not search for the best hyper-parameters through grid-search). Through the training of our model, no development set is applied to find the best checkpoints, but stop until the training step is reached. During the test procedure, we adopt FAISS IndexFlat Index (Johnson and Guestrin, 2018) to accelerate the speed to find the k nearest neighbors. We average the results with 3 different random seeds.

4.2 Results

Results on the Cross-Domain Amazon Dataset. The results are shown in Table 1. Overall, our model COBE achieves state-of-the-art performances with an accuracy of 90.39% on average for the 12 cross-domain tasks. It achieves state-of-the-art performance in 9 of 12 tasks. The result is 2.14% higher than that of BERT-CE (88.25%), which indicates that our proposed contrastive learning method can be more effective and generalized than methods based on cross-entropy loss. COBE is also 1.83% higher than BERT-adv (88.56%), which implies that directly pushing the representations of different domains with the same (different) labels close (apart) results in a strong performance on the cross-domain sentiment classification.

DAAT uses the unlabeled data from the source domain and the target domain to continually train BERT to mix the information of the source domain and target domain. Then the training objective of cross-entropy and the domain discriminator are jointly optimized to obtain the sentiment classification model. The average accuracy of our model is 0.27% higher than that of DAAT, which uses additional data to continually train BERT to transfer knowledge in the source domain to the target domain. Although DAAT achieves great performance, it is more time-consuming and resource-wasting compared with solely using contrastive learning. In the tasks of E→ B, E → D, and K → D, the accuracies of our model are smaller compared with DAAT, and the possible reason can be that the source domains' data have less shared information with the target domains. But with unlabeled data for continual training, some domain-specific information is extracted in DAAT and further results in a better performance.

Moreover, the average accuracy of the model COBE* (82.05%) outperforms that of BERT-CE* (54.86%) with a large margin, where the parameters of BERT are fixed (corresponding to the scenario that pre-trained models are too large for fine-tuning). The model BERT-CE* fails to predict the sentiments of the target domain using cross-entropy-based methods, but with contrastive learning, it can obtain strong results (similar performance to BERT-CE). But the performance of models fixing BERT parameters is still largely worse than that of unfixed models.

Results on FDU-MTL. First, we test our model in the multi-domain setting, training the model on the data of 16 domains and evaluating it on the whole test data. The results are shown in Table 2. Our model achieves the state-of-the-art performance with an accuracy of 91.49% on average, and in the 12 of 16 domains, it achieves the state-of-the-art performance. The accuracy is 0.67% higher than that of BERT-CE, and 0.99% higher than that of DAEA-B. In particular, using BERT-CE solely in the multi-domain setting can achieve competitive performance (90.82%), which is neglected by previous studies. The accuracy of our model COBE on the IMDB data is lower than DSA with a large margin, which may result from the max sequence length for BERT being 128, much smaller than the average sequence length in IMDB (128 to 256). Our model COBE* achieves an accuracy of 83.35% in the multi-domain setting, which is also higher than that of BERT-CE* with a margin of 3.14%.

Then we also evaluate our model in the 15-1 settings, referring to that we train the model on 15 domains and test it on the domain left (shown in Table 3. Our model achieves state-of-the-art performance with an accuracy of 90.03%. The accuracies in 14 of 16 tasks are larger than that in previous

[1]https://huggingface.co/

Domain	ASP	DA	DSA	DAEA	DAEA-B	BERT-CE*	BERT-CE	COBE*	COBE
Books	84.00	88.50	89.10	89.00	N/A	81.33	**90.67**	85.17	90.17
Electronics	86.80	89.00	87.90	91.80	N/A	82.17	91.92	82.92	**93.58**
DVD	85.50	88.00	88.10	88.30	N/A	78.83	89.00	79.42	**89.67**
Kitchen	86.20	89.00	85.90	90.30	N/A	79.92	91.17	81.33	**91.50**
Apparel	87.00	88.80	87.80	89.00	N/A	83.33	92.08	87.25	**92.33**
Camera	89.20	91.80	90.00	92.00	N/A	81.83	93.25	87.50	**93.58**
Health	88.20	90.30	92.90	89.80	N/A	81.25	93.33	85.00	**93.92**
Music	82.50	85.00	84.10	88.00	N/A	79.42	88.92	80.33	**90.33**
Toys	88.00	89.50	85.90	91.80	N/A	78.25	92.41	83.75	**93.42**
Video	84.50	89.50	90.30	**92.30**	N/A	78.17	90.33	83.67	89.91
Baby	88.20	90.50	91.70	92.30	N/A	82.33	93.00	84.42	**93.92**
Magazines	92.20	92.00	92.10	**96.50**	N/A	83.41	93.75	89.67	94.08
Software	87.20	90.80	87.00	92.80	N/A	83.42	92.42	85.33	**93.42**
Sports	85.70	89.80	85.80	90.80	N/A	78.50	91.50	84.50	**92.83**
IMDB	85.50	89.80	**93.80**	90.80	N/A	76.43	86.33	76.50	86.91
MR	76.70	75.50	73.30	77.00	N/A	74.75	83.00	76.83	**84.33**
Avg	86.09	88.61	87.86	90.16	90.50	80.21	90.82	83.35	**91.49**

Table 2: Results on FDU-MTL in the multi-domain setting. BERT-CE* and COBE* refer to the models fixing the parameters of BERT, and only tuning the parameters of MLP layer.

	ASP	DSR-at	DAEA	COBE
Books	81.50	85.80	87.30	**90.67**
Electronics	83.80	89.50	85.80	**92.33**
DVD	84.50	86.30	88.80	87.50
Kitchen	87.50	88.30	88.00	**90.75**
Apparel	85.30	85.80	88.00	**91.16**
Camera	85.30	88.80	90.00	**91.67**
Health	86.00	90.50	91.00	**94.33**
Music	81.30	84.80	86.50	**89.17**
Toys	88.00	90.30	90.30	**92.33**
Video	86.80	85.30	**91.30**	88.50
Baby	86.50	84.80	90.30	**93.17**
Magazines	87.00	84.00	88.50	**90.50**
Software	87.00	90.80	89.80	**90.82**
Sports	87.00	87.00	90.50	**92.15**
IMDB	84.00	83.30	85.80	**86.58**
MR	72.00	76.30	75.50	**78.91**
Avg	84.59	86.35	87.96	**90.03**

Table 3: Results on FDU-MTL in the 15-1 setting.

studies. It is 2.07% higher than the average accuracy of DAEA. The experimental results also show that contrastive learning can perform better than cross-entropy-based models with adversarial training for cross-domain sentiment analysis.

4.3 Visualization

The visualization of the sentence representations $\mathbf{h}_i$ in COBE is shown in Figure 3. For the B->K (Books->Kitchen) task in Figure 3 (a), first, the representations of positive and negative data are separated acutely with a large margin between each other. Second, representations of source and target domains with the same labels are close to each other, which means the knowledge learned from the source domain is transferred to the target domain effectively. For the multi-domain setting in Figure 3 (b) (left), we can observe that the representations with the same labels are separated into different clusters w.r.t the labels, and in the sentence representations with the same label but different domains mix up well, which satisfy the requirements of the cross-domain sentiment analysis.

To further compare the contrastive learning method with cross-entropy-based methods, we illustrate the representations of the source domain and the target domain in COBE, BERT-CE in Figure 3 (c)(d), respectively (visualizations of COBE* and BERT-CE* are shown in Appendix). Obviously, the sentence representations are separated in the target domain in BERT-CE less effective than that in COBE. The visualizations show the effectiveness of contrastive learning in transferring the learned knowledge in the source domain to the target domain. Meanwhile, it demonstrates operating the sentence representations in the feature space has a strong generalization ability in the cross-domain sentiment analysis tasks.

4.4 Robustness Analysis

We evaluate our model on adversarial samples generated by using the well-known substitution-based adversarial attack method–Textfooler (Jin et al., 2020). Given an input X_i and a pre-trained classification model F, a valid adversarial sample X_i^{adv} should conform the following requirements:

$$F(X_i) \neq F(X_i^{adv}), \;\; Sim(X_i, X_i^{adv}) \geq \epsilon. \quad (12)$$

where Sim is a similarity function and ϵ is the minimum similarity between the original input and the adversarial sample, which is often a semantic

Text	Gold Label	Output
Test Data. This story is true to life living in south west and west phila. It brought back many memories and changing the names did not bother me. I really enjoyed reading about life the way it was back in the 55 to 70 era.	Positive	Positive
k **nearest neighbors.**		
(1) Have to be honest and say that I haven't seen many independent films, but I thought this one was very well done. The direction and cinematography were engaging without becoming a distraction.	Positive	
(2) I bought this wireless weather station as a gift. The recipient loves it. For the price, he is really enjoying it.	Positive	
(3) I think j-14 is a really good magazine if u like to hear the latest gossip about all your favourite celebrity 's, or if u like to get nice posters of all the hot celebrity 's.	Positive	

Table 4: Case Study on FDU-MTL.

Original Text: DEF. NOT A GOOD TANK. You look at them in a picture frame, the fish are crammed in there.
Adversarial Text: DEF. Not a alright tank. you look at them in a photography sashes, the fish are teeming in there.

Table 5: Adversarial example based on TextFooler.

	BERT-CE*	BERT-CE	COBE*	COBE
Books	42.50	71.50	69.00	**78.00**
Multi-	49.50	73.50	72.50	**81.00**

Table 6: Results on the adversarial samples. Books for the trained model using the data of Books domain in the cross-domain Amazon dataset, and Multi- for the trained model using multi-domain data in FDU-MTL.

and syntactic similarity function. The details for generation refer to Jin et al. (2020). An adversarial sample is shown in Table 5, where the sentence semantic information is not corrupted, but some words are replaced.

We test our model (trained using Books data in the cross-domain Amazon dataset) with 200 adversarial samples from the Kitchen domain, and our model (trained using multi-domain data in FDU-MTL) with 200 adversarial samples randomly selected from the multi-domain test data. The results are shown in Table 6. Our model COBE achieves 78.00% and 81.00% accuracies for the two kind of adversarial data, which are 6.5% and 7.5% higher than BERT-CE. Meanwhile, the model COBE* outperforms BERT-CE* with a large margin (26.5% and 23%). The results demonstrate that contrastive-learning based models have better robustness than cross-entropy-based models.

4.5 Case Study

The case study is shown in Table 4. As can be observed, the k nearest neighbors of the test data (Books) are reviews from different domains (Video, Electronics and Magazines) with positive labels, and it outputs the correct label for the test data. Note that the key sentiment information is similar for the original text and neighbors in the case such as 'enjoy', 'engaging', 'enjoying' and 'favorite'. It shows that our model can learn effective information from multi-domain data for the sentiment classification task, and the representations of different domains mix up well, which serve as a strong sentiment knowledge base for the classification.

5 Conclusion

We explored the contrastive learning method in the cross-domain sentiment analysis task. We proposed a suitable contrastive loss for the supervised sentiment analysis task with the in-batch negatives method. Experiments on two standard datasets showed the effectiveness of our model. Visualizations also demonstrated the effectiveness of transferring knowledge learned in the source domain to the target domain. We also showed that our model has stronger robustness than cross-entropy-based models through the adversarial test.

Acknowledgements

Yue Zhang is the corresponding author. We would also like to thank the anonymous reviewers for the detailed and thoughtful reviews. The work is funded by the Zhejiang Province Key Project 2022SDXHDX0003.

References

John Blitzer, Mark Dredze, and Fernando Pereira. 2007. Biographies, bollywood, boom-boxes and blenders: Domain adaptation for sentiment classification. In *Proceedings of the 45th annual meeting of the association of computational linguistics*, pages 440–447.

Yitao Cai and Xiaojun Wan. 2019. Multi-domain sentiment classification based on domain-aware embedding and attention. In *IJCAI*, pages 4904–4910.

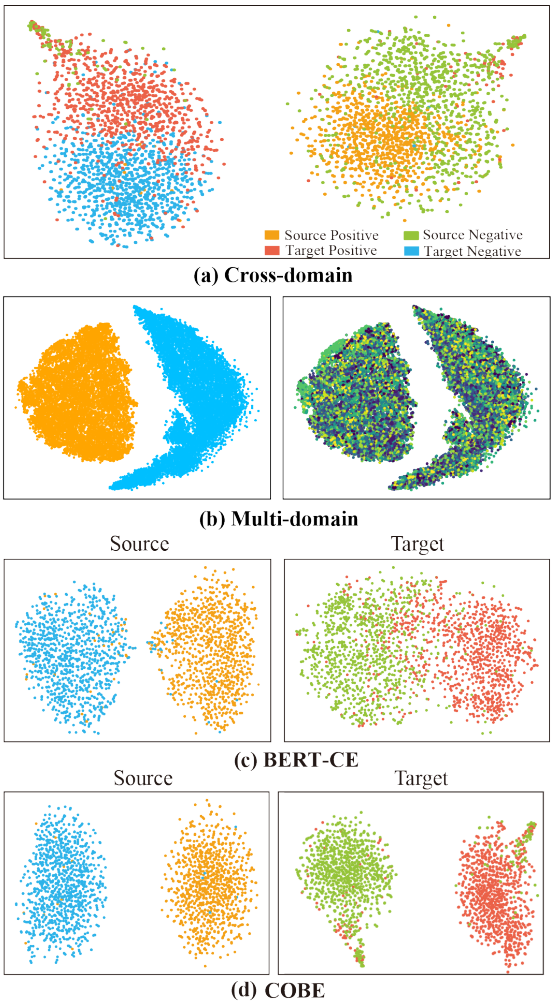

Figure 3: Visualizations of the sentence representations. We use t-SNE to transfer 192-dimensional feature space into two-dimensional space. (a)(c)(d) for the representations of the B->K task in the cross-domain Amazon dataset; (b) for those of the multi-domain task in FDU-MTL (colors in left for sentiment labels and right for domains).

Kaidi Cao, Colin Wei, Adrien Gaidon, Nikos Arechiga, and Tengyu Ma. 2019. Learning imbalanced datasets with label-distribution-aware margin loss. *Advances in neural information processing systems*, 32.

Ting Chen, Simon Kornblith, Mohammad Norouzi, and Geoffrey Hinton. 2020. A simple framework for contrastive learning of visual representations.

Shuhao Cui, Shuhui Wang, Junbao Zhuo, Liang Li, Qingming Huang, and Qi Tian. 2020. Towards discriminability and diversity: Batch nuclear-norm maximization under label insufficient situations. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 3941–3950.

Jesse Dodge, Gabriel Ilharco, Roy Schwartz, Ali Farhadi, Hannaneh Hajishirzi, and Noah Smith. 2020. Fine-tuning pretrained language models: Weight initializations, data orders, and early stopping. arXiv.

Chunning Du, Haifeng Sun, Jingyu Wang, Qi Qi, and Jianxin Liao. 2020. Adversarial and domain-aware bert for cross-domain sentiment analysis. In *Proceedings of the 58th annual meeting of the Association for Computational Linguistics*, pages 4019–4028.

Gamaleldin Elsayed, Dilip Krishnan, Hossein Mobahi, Kevin Regan, and Samy Bengio. 2018. Large margin deep networks for classification. *Advances in neural information processing systems*, 31.

Yaroslav Ganin, Evgeniya Ustinova, Hana Ajakan, Pascal Germain, Hugo Larochelle, François Laviolette, Mario Marchand, and Victor Lempitsky. 2016. Domain-adversarial training of neural networks. *The journal of machine learning research*, 17(1):2096–2030.

Jun Gao, Di He, Xu Tan, Tao Qin, Liwei Wang, and Tie Yan Liu. 2019. Representation degeneration problem in training natural language generation models. *7th International Conference on Learning Representations, ICLR 2019*, pages 1–14.

Tianyu Gao, Xingcheng Yao, and Danqi Chen. 2021. Simcse: Simple contrastive learning of sentence embeddings. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 6894–6910.

Beliz Gunel, Jingfei Du, Alexis Conneau, and Veselin Stoyanov. 2020. Supervised contrastive learning for pre-trained language model fine-tuning. In *International Conference on Learning Representations*.

Chao Jia, Yinfei Yang, Ye Xia, Yi-Ting Chen, Zarana Parekh, Hieu Pham, Quoc Le, Yun-Hsuan Sung, Zhen Li, and Tom Duerig. 2021. Scaling up visual and vision-language representation learning with noisy text supervision. In *International Conference on Machine Learning*, pages 4904–4916. PMLR.

Di Jin, Zhijing Jin, Joey Tianyi Zhou, and Peter Szolovits. 2020. Is bert really robust? a strong baseline for natural language attack on text classification and entailment. In *Proceedings of the AAAI conference on artificial intelligence*, volume 34, pages 8018–8025.

Li Jing, Pascal Vincent, Yann LeCun, and Yuandong Tian. 2021. Understanding dimensional collapse in contrastive self-supervised learning. *arXiv preprint arXiv:2110.09348*.

Tyler B Johnson and Carlos Guestrin. 2018. Training deep models faster with robust, approximate importance sampling. *Advances in Neural Information Processing Systems*, 31.

Jacob Devlin Kenton, Chang Ming-Wei, and Lee Kristina Toutanova. 2019. Bert: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of NAACL-HLT*, pages 4171–4186.

Urvashi Khandelwal, Angela Fan, Dan Jurafsky, Luke Zettlemoyer, and Mike Lewis. 2020. Nearest neighbor machine translation. *arXiv preprint arXiv:2010.00710*.

Prannay Khosla, Piotr Teterwak, Chen Wang, Aaron Sarna, Yonglong Tian, Phillip Isola, Aaron Maschinot, Ce Liu, and Dilip Krishnan. 2020. Supervised contrastive learning. *Advances in Neural Information Processing Systems*, 33:18661–18673.

Diederik P Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*.

Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Veselin Stoyanov, and Luke Zettlemoyer. 2020. BART: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7871–7880, Online. Association for Computational Linguistics.

Linyang Li, Demin Song, Ruotian Ma, Xipeng Qiu, and Xuanjing Huang. 2021. Knn-bert: Fine-tuning pre-trained models with knn classifier. *arXiv preprint arXiv:2110.02523*.

Shoushan Li, Yunxia Xue, Zhongqing Wang, and Guodong Zhou. 2013. Active learning for cross-domain sentiment classification. In *Twenty-Third International Joint Conference on Artificial Intelligence*.

Zheng Li, Ying Wei, Yu Zhang, and Qiang Yang. 2018a. Hierarchical attention transfer network for cross-domain sentiment classification. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 32.

Zheng Li, Ying Wei, Yu Zhang, and Qiang Yang. 2018b. Hierarchical Attention Transfer Network for Cross-Domain Sentiment Classification. *Proceedings of the AAAI Conference on Artificial Intelligence*, 32(1).

Bing Liu. 2012. Sentiment analysis and opinion mining. *Synthesis lectures on human language technologies*, 5(1):1–167.

Pengfei Liu, Xipeng Qiu, and Xuan-Jing Huang. 2017. Adversarial multi-task learning for text classification. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1–10.

Qi Liu, Yue Zhang, and Jiangming Liu. 2018. Learning domain representation for multi-domain sentiment classification. *NAACL HLT 2018 - 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies - Proceedings of the Conference*, 1:541–550.

Weiyang Liu, Yandong Wen, Zhiding Yu, and Meng Yang. 2016. Large-margin softmax loss for convolutional neural networks. In *ICML*, volume 2, page 7.

Kamil Nar, Orhan Ocal, S Shankar Sastry, and Kannan Ramchandran. 2019. Cross-entropy loss and low-rank features have responsibility for adversarial examples. *arXiv preprint arXiv:1901.08360*.

Arvind Neelakantan, Tao Xu, Raul Puri, Alec Radford, Jesse Michael Han, Jerry Tworek, Qiming Yuan, Nikolas Tezak, Jong Wook Kim, Chris Hallacy, et al. 2022. Text and code embeddings by contrastive pre-training. *arXiv preprint arXiv:2201.10005*.

Xiaoye Qu, Zhikang Zou, Yu Cheng, Yang Yang, and Pan Zhou. 2019. Adversarial category alignment network for cross-domain sentiment classification. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 2496–2508.

Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, et al. 2021. Learning transferable visual models from natural language supervision. In *International Conference on Machine Learning*, pages 8748–8763. PMLR.

Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, Ilya Sutskever, et al. 2019. Language models are unsupervised multitask learners. *OpenAI blog*, 1(8):9.

Richard Socher, Alex Perelygin, Jean Wu, Jason Chuang, Christopher D Manning, Andrew Y Ng, and Christopher Potts. 2013. Recursive deep models for semantic compositionality over a sentiment treebank. In *Proceedings of the 2013 conference on empirical methods in natural language processing*, pages 1631–1642.

Kihyuk Sohn. 2016. Improved deep metric learning with multi-class n-pair loss objective. *Advances in neural information processing systems*, 29.

Sainbayar Sukhbaatar, Joan Bruna, Manohar Paluri, Lubomir Bourdev, and Rob Fergus. 2015. Training convolutional networks with noisy labels. In *3rd International Conference on Learning Representations, ICLR 2015*.

Guoyin Wang, Yan Song, Yue Zhang, and Dong Yu. 2019. Learning word embeddings with domain awareness. *arXiv preprint arXiv:1906.03249*.

Tongzhou Wang and Phillip Isola. 2020. Understanding contrastive representation learning through alignment and uniformity on the hypersphere. In *International Conference on Machine Learning*, pages 9929–9939. PMLR.

Linyi Yang, Lifan Yuan, Leyang Cui, Wenyang Gao, and Yue Zhang. 2022. Factmix: Using a few labeled in-domain examples to generalize to cross-domain named entity recognition. *arXiv preprint arXiv:2208.11464*.

Wen-tau Yih, Kristina Toutanova, John C Platt, and Christopher Meek. 2011. Learning discriminative projections for text similarity measures. In *Proceedings of the fifteenth conference on computational natural language learning*, pages 247–256.

Jianfei Yu and Jing Jiang. 2016. Learning sentence embeddings with auxiliary tasks for cross-domain sentiment classification. In *Proceedings of the 2016 conference on empirical methods in natural language processing*, pages 236–246.

Kai Zhang, Hefu Zhang, Qi Liu, Hongke Zhao, Hengshu Zhu, and Enhong Chen. 2019. Interactive attention transfer network for cross-domain sentiment classification. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, pages 5773–5780.

Tianyi Zhang, Felix Wu, Arzoo Katiyar, Kilian Q Weinberger, and Yoav Artzi. 2020. Revisiting few-sample bert fine-tuning. *arXiv preprint arXiv:2006.05987*.

Zhilu Zhang and Mert Sabuncu. 2018. Generalized cross entropy loss for training deep neural networks with noisy labels. *Advances in neural information processing systems*, 31.

Jun Zhao, Tao Gui, Qi Zhang, and Yaqian Zhou. 2021. A relation-oriented clustering method for open relation extraction. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 9707–9718.

Renjie Zheng, Junkun Chen, and Xipeng Qiu. 2018. Same representation, different attentions: shareable sentence representation learning from multiple tasks. In *Proceedings of the 27th International Joint Conference on Artificial Intelligence*, pages 4616–4622.

Jie Zhou, Junfeng Tian, Rui Wang, Yuanbin Wu, Wenming Xiao, and Liang He. 2020a. Sentix: A sentiment-aware pre-trained model for cross-domain sentiment analysis. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 568–579.

Jie Zhou, Junfeng Tian, Rui Wang, Yuanbin Wu, Wenming Xiao, and Liang He. 2020b. SentiX: A sentiment-aware pre-trained model for cross-domain sentiment analysis. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 568–579, Barcelona, Spain (Online). International Committee on Computational Linguistics.

Yftah Ziser and Roi Reichart. 2018. Pivot based language modeling for improved neural domain adaptation. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 1241–1251.

A Statistics for FDU-MTL.

Domain	Train	Dev	Test	Avg. Length
Books	1400	200	400	159
Electronics	1398	200	400	101
DVD	1400	200	400	173
Kitchen	1400	200	400	89
Apparel	1400	200	400	57
Camera	1397	200	400	130
Health	1400	200	400	81
Music	1400	200	400	136
Toys	1400	200	400	90
Video	1400	200	400	156
Baby	1300	200	400	104
Magazines	1370	200	400	117
Software	1315	200	400	129
Sports	1400	200	400	94
IMDB	1400	200	400	269
MR	1400	200	400	21

Table 7: Statistics of FDU-MTL.

B Reconstruction Loss

We attempt to reconstruct the representations of BERT which means another MLP layer is applied by $\mathbf{h}_i^{rec} = MLP(\mathbf{h}_i)$. Then a reconstruction loss of MSE (mean-squared loss) is added to retain the semantic information, $\mathcal{L}_{rec} = ||\mathbf{h}_i^{rec} - \mathbf{h}_i^{CLS}||$ as (Zhao et al., 2021). But little improvement (an average accuracy of 90.81% on the FDU-MTL multi-domain setting and 90.13% on the cross-domain Amazon dataset) is obtained, which is 0.68% and 0.26% lower than COBE, respectively. It indicates that the reconstruction loss is not suitable for the task of cross-domain sentiment analysis.

C SCL Loss

To verify the effectiveness of our propose loss function, we compare our contrastive learning loss with the SCL loss (Gunel et al., 2020), which can be formulated as follows:

$$\mathcal{L}_{SCL} = -\sum_{i}^{M} \frac{1}{|N_i^+|} \sum_{k \in N_i^+} log \frac{exp(sim(\mathbf{h}_i, \mathbf{h}_k)/\tau)}{\sum_{i \neq j}^{M} exp(sim(\mathbf{h}_i, \mathbf{h}_j)/\tau)} \tag{13}$$

In fairness, we use the kNN predictor the same as our proposed model. The model with SCL loss achieves an average accuracy of 91.13% on the FDU-MTL multi-domain setting and 90.05% on the cross-domain Amazon dataset (0.36% and 0.34% lower than COBE, respectively). The experiments prove the effectiveness of our proposed loss function with the in-batch negative samples, which aims to tighten all the samples of the same labels as positive pairs. The conclusion is different from that in Khosla et al. (2020), whose experiments demonstrate that the separately calculating each positive pair separately (SCL loss) achieves better results for image classification. It may results from the reason that the batch size influences the results of the two methods, and our batch sizes (8 and 32) are comparatively smaller compared with their study (6144), which may motivate further theoretical analysis.

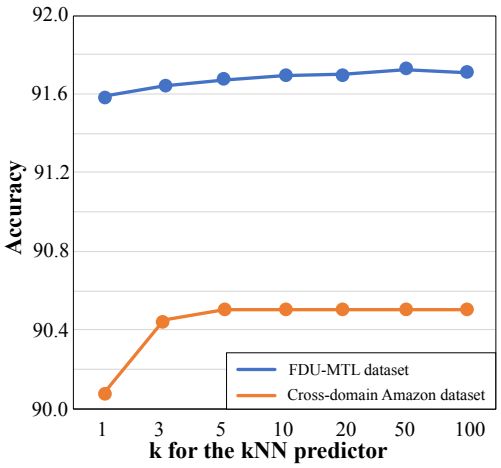

Figure 4: Evaluation with respect to different numbers of k for one random seed.

D Influence of k

In order to discover the sensitivity of our model to the influence of k for the kNN predictor (shown in Figure 4), we evaluate our model with respect to the different numbers of k. As observed, the accuracies of COBE stop to increase and keep stable when $k >= 5$, which indicates that the model is little sensitive with the hyper-parameter k. The phenomenon demonstrates that the sentence representations learned by COBE are effectively separated and stable for classification.

Original Text	Adversarial Text
Very nice iron!. This is a great iron. It's quite heavy, but I like that. It really gets out the wrinkles. I don't even mind ironing any more.	Awfully sweet iron! This is a whopping iron it 's quite heavy, but I like that it really gets out the wrinkles I don't even mind ironing any more.
A good idea, disappointing in use. These silicone pot holders are indeed brightly colored, easy to wash in the dishwasher, and protective even when wet. They are also clumsily stiff at the same time as they are slippery, the net result being a miserable failure in the kitchen. They are useful for protecting a counter from a hot pot, but not for picking the hot pot up.	A good ideas, agonizing in use these silicon pot holders are indeed brightly colour, easy to wash in the dishwasher, and protective even when clammy they are also clumsily painstaking at the same time as they are slippery, the net raison being a miserable failure in the kitchen they are useful for protecting a counter from a hot pot, but not for picking the hot pot up.
Love this piece. I just bought this piece and tried it out. I love the size and no drip mouth.The color is beautiful and its so pretty on my buffet.	Like this pieces I just obtained this pieces and attempts it out. I luv the size and no drip mouths the colorful is beautiful and its however rather on my buffet.

Table 8: Adversarial Samples.

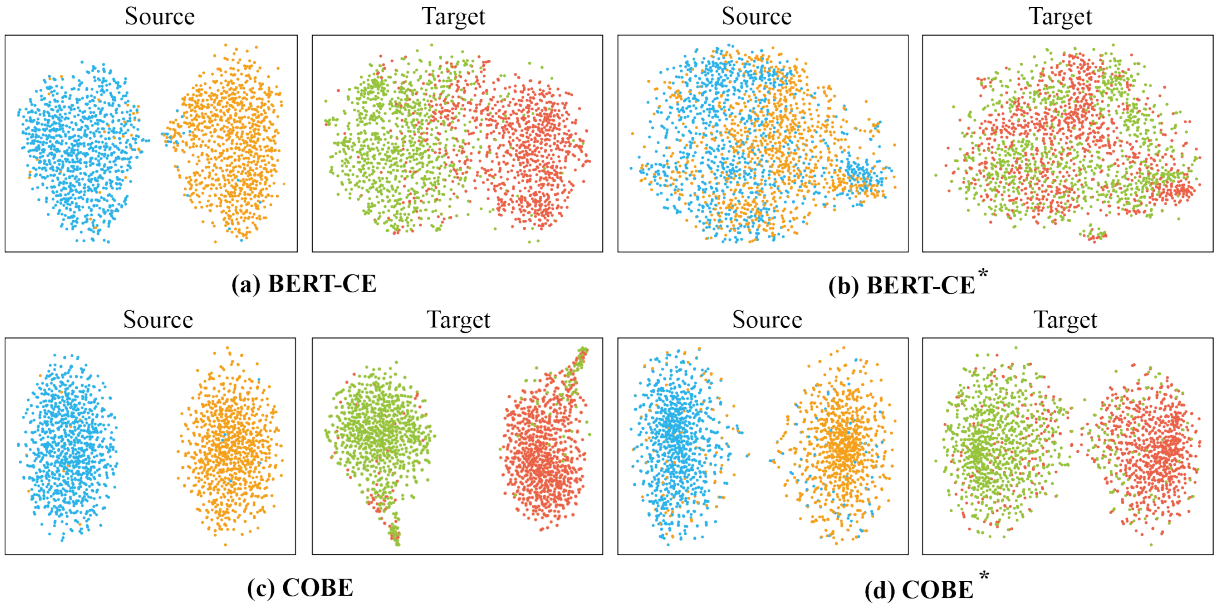

Figure 5: Visualization of sentence representation obtained from BERT and COBE. We use t-SNE to transfer the feature space into two-dimensional space for the B→K task.

Exploiting Sentiment and Common Sense for Zero-shot Stance Detection

Yun Luo [1,2], **Zihan Liu** [2], **Yuefeng Shi** [2], **Stan Z. Li** [2], **Yue Zhang** [2,3]
[1] School of Computer Science And Technology, Zhejiang University, Hangzhou, 310024, P.R. China.
[2] School of Engineering, Westlake University, Hangzhou, 310024, P.R. China.
[3] Institute of Advanced Technology, Westlake Institute for Advanced Study, Hangzhou, 310024, P.R. China.
{luoyun, guofang, liuzihan, stan.zq.li, zhangyue}@westlake.edu.cn

Abstract

The stance detection task aims to classify the stance toward given documents and topics. Since the topics can be implicit in documents and unseen in training data for zero-shot settings, we propose to boost the transferability of the stance detection model by using sentiment and commonsense knowledge, which are seldom considered in previous studies. Our model includes a graph autoencoder module to obtain commonsense knowledge and a stance detection module with sentiment and commonsense. Experimental results show that our model outperforms the state-of-the-art methods on the zero-shot and few-shot benchmark dataset–VAST. Meanwhile, ablation studies prove the significance of each module in our model. Analysis of the relations between sentiment, common sense, and stance indicates the effectiveness of sentiment and common sense.

1 Introduction

Stance detection aims to identify the authors' attitudes or positions (*Pro (support), Con (oppose), Neu (neutral)*) towards a specific target such as an entity, a topic. (Mohammad et al., 2017, 2016; Walker et al., 2012; Qiu et al., 2015; Zhang et al., 2017). It is crucial for understanding opinions and analyzing how opinions are presented in texts regarding specific issues, and much work has been done building stance detection models (Wei et al., 2016; Dias and Becker, 2016; Allaway and Mckeown, 2020). There are two salient challenges to the task. First, obtaining rich annotated data in stance detection is time-consuming and labor-intensive. To address this issue, Allaway and Mckeown (2020) propose the dataset VAST containing various topics for few-shot and zero-shot stance detection tasks, requiring the model to classify the stance of topics unseen in the training set. Second, the topic is often not explicitly mentioned in the document, resulting in difficulty. Considering

Example 1 Topic : Olympics **Stance :** Pro
Text : The **games** should proceed. **Athletes** have made tremendous sacrifices to qualify and be prepared. It would be cruel to deny them their chance. In the future the Games should be held in countries within the top say 15 GDP per capita.
Example 2 **Topic :** Nuclear power **Stance :** Pro
Text : I totally agree with this premise. As a younger person I was against **Nuclear power** (I was in college during 3 mile island) but now it seems that nuclear should be in the mix. Fission technology is better, and will continue to get better if we actively promote its development. The prospect of fusion energy also needs to be explored. If it's good enough for the sun and the stars, it's good enough for me.

Figure 1: Examples for stance detection VAST.

Figure 1 Example 1, the document does not explicitly contain the topic 'Olympics', but 'Games' and 'Athlete' implicitly refer to the topic.

Existing work incorporates external knowledge to solve the challenges (Liu et al., 2021; Jayaram and Allaway, 2021). For example, CKE-Net achieves the state-of-the-art results for zero-shot stance detection, which uses pre-trained model BERT and commonsense knowledge graph on ConceptNet (Liu et al., 2021). However, such a method only considers the knowledge relations between documents and topics (i.e., the commonsense knowledge in two-hop directed paths on the ConceptNet from documents to topics), limiting the generalization of adding other types of related knowledge. In Figure 1 Example 1, the word 'games' can also represent the computer programs in a different document. Such knowledge cannot be used for that document if no relation between 'game' and 'computer program' can be learned from the relations between documents and topics in the dataset.

We consider incorporating two types of general knowledge, including common sense and sentiment. First, we incorporate commonsense

Proceedings of the 29th International Conference on Computational Linguistics, pages 7112–7123
October 12–17, 2022.

knowledge into the stance detection model using a graph autoencoder module. We take a pre-training method to train the graph autoencoder, separately to the stance detection module. Second, stance detection is significantly influenced by the sentiment information (Li and Caragea, 2019; Sobhani et al., 2016; Hardalov et al., 2022) (case study can be seen in Appendix). In Figure 1 Example 2, the document contains many positive words like 'good', and 'better' regarding the topic 'nuclear power', which implies a *Pro* stance. However, little existing work has considered sentiment knowledge for zero-shot stance detection. We use the sentiment-aware BERT (SentiBERT henceforth) to extract the sentiment information, assisting in classifying the stances of topics.

Existing work on injecting knowledge into NLP models can be broadly classified into two categories. One uses a graph encoder to integrate structural knowledge into a neural encoder (Li et al., 2019; Ghosal et al., 2020; Bai et al., 2021) and the other injects knowledge by using training losses to tune model parameters (Jayaram and Allaway, 2021; Peters et al., 2019; Logan et al., 2019; Liu et al., 2019). In our work, we consider the former for commonsense knowledge and the latter for sentiment due to the sources of information. In the component of knowledge graph encoding, a graph autoencoder consisting of relational graph convolutional network (RGCN) encoders (Schlichtkrull et al., 2018) and a DisMult decoder (Yang et al., 2014) is trained using negative sampling to obtain the relations of concepts on the commonsense knowledge graph. We inject sentiment knowledge encoded by SentiBERT into BERT using a cross attention module and tuning the fusing process by the training loss of the stance detection.

Our model achieves the state-of-the-art performance on the benchmark dataset VAST (Allaway and Mckeown, 2020) in both zero-shot and few-shot stance detection, improving the performance on many challenging linguistic phenomena such as sarcasm and quotations. We analyze the performance of our model with respect to different sentiment and common sense features, finding that the data with the corresponding sentiment and stance pairs (i.e., *(Pos, Pro)* and *(Neg, Con)*) are the easiest part for models to classify; in addition, increased commonsense knowledge leads to improved performance of the stance detection model. To our knowledge, we are the first to incorporate both sentiment and common sense into zero-shot stance detection model. The code has been released https://github.com/LuoXiaoHeics/StanceCS.

2 Related Work

Stance detection, also known as stance classification (Walker et al., 2012), stance identification (Zhang et al., 2017), stance prediction (Qiu et al., 2015), debate-side classification (Anand et al., 2011), and debate stance classification (Hasan and Ng, 2013), aims to identify the stance of the text author towards a target (an entity, event, idea, opinion, claim, topic, etc.) either explicitly mentioned or implied within the text. For the initial task of stance detection, models are trained an individual classifier for each topic (Lin et al., 2006; Beigman Klebanov et al., 2010; Sridhar et al., 2015; Hasan and Ng, 2013, 2014; Li et al., 2018) or only a small number of topics are both in training and evaluation sets (Faulkner, 2014; Du et al., 2017; Hardalov et al., 2021).

However, given rich and varying topics, data annotation can be time-consuming and labor-intensive. Researchers attempt to solve the task in a cross-target setting (Augenstein et al., 2016; Xu et al., 2018a), training the model in a topic and testing it on another one, and propose several weakly supervised approaches using unlabeled data related to the test topics (Zarrella and Marsh, 2016; Wei et al., 2016; Dias and Becker, 2016). Other studies propose the tasks of zero-shot and few-shot stance detection, which requires training the model in data of several topics and testing it on some unseen topics (Allaway and Mckeown, 2020).

Allaway and Mckeown (2020) propose to solve the task using a topic-grouped attention net, which uses the relation between the training and evaluation topics in an unsupervised way, and they also analyze the relationship between sentiment and stance from the perspective of the model by corrupting sentences with replacing sentiment words. Jayaram and Allaway (2021) use human rationales as attribution priors to provide faithful explanations of models. Liu et al. (2021) propose to incorporate commonsense knowledge to learn the relations between different topics utilizing a CompGCN (a variant of graph convolution networks). However, it limits the content of knowledge (only knowledge from documents to stances in the training data). Our model differs from such a method in that our model adopts the related concepts of both

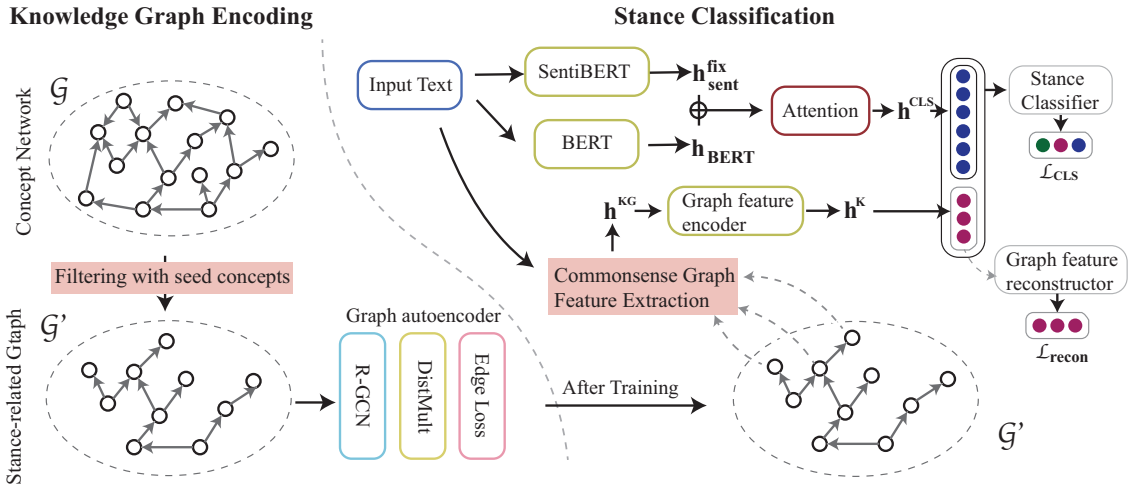

Figure 2: Framework of our proposed model, which contains two components, (1) knowledge graph encoding, (2) stance detection with sentiment and common sense.

documents and topics and uses a pre-trained graph autoencoder to obtain commonsense information. Adversarial learning is also applied to solve the zero-shot task by using unlabeled raw data (Allaway et al., 2021). Unlike the above work, we consider integrating external knowledge for zero-shot stance detection, including sentiment and commonsense information that are rarely considered. To our knowledge, we are the first to systematically incorporate sentiment and commonsense knowledge into the stance detection model and analyze the relationship between them (in Section 4.5 and 4.6).

3 Method

The architecture of our model is illustrated in Figure 2, which contains two components: (1) knowledge graph encoding, which integrates commonsense knowledge from ConceptNet (Section 3.1); (2) stance detection with sentiment and commonsense knowledge (Section 3.2).

3.1 Knowledge Graph Autoencoder

Formally, the ConceptNet is represented as a directed labeled graph $\mathcal{G} = \{\mathcal{V}, \mathcal{E}, \mathcal{R}\}$, with concepts $v_i \in \mathcal{V}$ and labeled edges $(v_i, r, v_j) \in \mathcal{E}$, where $r \in \mathcal{R}$ is the relation type of edge between v_i and v_j. The concepts in ConceptNet are unigram words or n-gram phrases in the triplet format. For example, one such triplet from ConceptNet is *(teacher, RelatedTo, job)*.

ConceptNet has a large size of approximately 14 million edges. We extract a subset of edges related to the VAST dataset for our task. From the training documents in VAST, we first extract the set of all unique nouns, adjectives, and adverbs. These words are treated as the seeds that we use to filter the ConceptNet to a sub-graph. We extract all the triplets with a one-edge distance to any of those seed concepts, resulting in a sub-graph $\mathcal{G}' = \{\mathcal{V}', \mathcal{E}', \mathcal{R}'\}$ with 310k concepts and 750k edges. The top 5 relations include 'RelatedTo', 'HasContext', 'IsA', 'Synonym' and 'DerivedFrom'. The sub-graph $\mathcal{G}'$ contains all the concepts related to stance targets in the VAST dataset.

Following Schlichtkrull et al. (2018), we construct a graph autoencoder to compute the representations of concepts in the sub-graph $\mathcal{G}'$. The autoencoder takes an incomplete set (randomly sampled with 50% probability in our model) of edges $\hat{\mathcal{E}}'$ from $\mathcal{E}'$ in $\mathcal{G}'$ as input. $\hat{\mathcal{E}}'$ is negative sampled to the overall set of samples denoted $\mathcal{T}$ (details in Training). Then we assign the possible edges $(v_i, r, v_j) \in \mathcal{T}$ with scores to determine the probability these edges are in $\mathcal{E}'$. Our graph autoencoder consists of a relational concept network (RGCN) (Schlichtkrull et al., 2018) encoder to obtain the latent feature representations of concepts and a DistMult scoring decoder (Yang et al., 2014) to recover the missing facts of triplets.

Encoder. RGCN has a solid ability to accumulate relational evidence in multiple inference steps. In each step, a neighborhood-based convolutional feature transformation process uses the related concepts to induce an enriched stance-aggregated feature vector for each concept. Our model contains two stacked RGCN encoders. We first initialize the parameters of concept feature vectors $\mathbf{g}_i$. Then the vectors are transformed into stance-aggregated

feature vectors $\mathbf{h}_i \in \mathbb{R}^d$ using the RGCN encoders:

$$f(x_i, l) = \sigma(\sum_{r \in \mathcal{R}} \sum_{j \in N_i^r} \frac{1}{v_{i,r}} W_r^{(l)} x_j + W_0^{(l)} x_i),$$

$$\mathbf{h}_i = \mathbf{h}_i^{(2)} = f(\mathbf{h}_i^{(1)}, 2)\ ;\ \mathbf{h}_i^{(1)} = f(\mathbf{g}_i, 1), \quad (1)$$

where f is the encoder network (requiring inputs of feature vector x_i and the rank of the layer l), N_i^r denotes the neighbouring concepts i with the relation $r \in \mathcal{R}$; $v_{i,r}$ is a normalization constant, which can be set in advance $v_{i,r} = |N_i^r|$ or learned by network learning; σ is the activation function like ReLU and $W_r^{(1/2)}, W_0^{(1/2)}$ are learnable parameters though training.

Training. We use DistMult factorization as the decoder to assign scores. For a given triplet (v_i, r, v_j), the score can be obtain as follows:

$$s(v_i, r, v_j) = \sigma(\mathbf{h}_{v_i}^T R_r \mathbf{h}_{v_j}), \quad (2)$$

where σ is logistic function; $\mathbf{h}_{v_i}, \mathbf{h}_{v_j} \in \mathbb{R}^d$ are the encoding feature vectors through the graph encoder for concept v_i and v_j. Each relation $r \in R$ is also associated with a diagonal matrix $R_r \in \mathbb{R}^{d \times d}$.

Our graph autoencoder module is trained using negative sampling (Schlichtkrull et al., 2018). We randomly corrupt the positive triplets, i.e., triplets in $\hat{\mathcal{E}}'$, to create an equal number of negative samples. The corruption is performed by modifying either of the connected concepts or relations randomly, creating the overall set of samples denoted by $\mathcal{T}$. The training objective is a binary classification between positive/negative (denoted as u) triplets with a cross entropy loss function:

$$\mathcal{L}_{\mathcal{G}'} = -\frac{1}{2|\hat{\mathcal{E}}'|} \sum_{(v_i, r, v_j, u) \in \mathcal{T}} (u log\ s(v_i, r, v_j) + (1-u) log(1 - s(v_i, r, v_j))). \quad (3)$$

3.2 Stance Detection Module

Sentiment Feature Encoding. To learn sentiment knowledge, we follow Zhou et al. (2020) to continually train BERT with sentiment masking. We mask the sentiment-related tokens such as sentiment lexicons, emoticons, and ratings with higher probability than general tokens. The model is trained to reconstruct the masked sentiment tokens and predict the rating of the sentences. The corrupted text $\hat{x}$ is fed into BERT to obtain each word representation $\mathbf{h}_i$ and the sentence representation $\mathbf{h}^{CLS}$. Softmax layers are used on $\mathbf{h}_i$ to predict each word's probability, the sentiment of words, and emoticon probability, respectively. A softmax layer on $\mathbf{h}^{CLS}$ is also used to predict the rating of the text $\hat{x}$. The tasks are trained using cross-entropy loss. Following Zhou et al. (2020) , the SentiBERT are trained on Amazon review dataset (Ni et al., 2019) and Yelp 2020[1] challenge dataset.

After pre-training the SentiBERT, given a document d and a topic t, we concatenate d and t as our model input x in the following format: $x = [CLS]\ d\ [SEP]\ t\ [SEP]$, SentiBERT to obtain its hidden states:

$$\mathbf{h}_{sent}^{fix} = SentiBERT(x), \quad (4)$$

where the parameters of SentiBERT are fixed in our model to keep sentiment information stabilized.

Commonsense Feature Encoding. After training the graph autoencoder, in order to extract the document-specific commonsense graph feature for the document d and the topic t, the unique nouns, adjectives, and adverbs in the document d and the topic t are extracted at first, which we denote as S. Then we extract a sub-graph $\mathcal{G}'_S$ from $\mathcal{G}'$, which contains all the triplets either of whose concepts are in S or within the vicinity of radius 1 from any of the concepts in S. Next, we make a forward pass of $\mathcal{G}'_S$ through the encoder of graph autoencoder to obtain the feature vectors $\mathbf{h}_j$ for all unique concepts j in $\mathcal{G}'_S$. The average of feature vectors $\mathbf{h}_j$ for all unique concepts in $\mathcal{G}'_S$ is regarded as the commonsense graph feature vector $\mathbf{h}^{KG}$ for the document d. The commonsense graph feature vector $\mathbf{h}^{KG}$ is feed into a encoder layer to obtain hidden states $\mathbf{h}^K$:

$$\mathbf{h}^K = W_k \mathbf{h}^{KG} + b_k \quad (5)$$

where W_k and b_k are the trained parameters of the linear layer.

Stance Classification. The input x is first fed into BERT to obtain its hidden states:

$$\mathbf{h}_{BERT} = BERT(x) \quad (6)$$

Then the hidden states of $\mathbf{h}_{BERT}, \mathbf{h}_{sent}^{fix}$ are concatenated and fed into a cross attention module to fuse the information of BERT and SentiBERT:

$$\mathbf{h}^{CLS} = CrossAttention([\mathbf{h}_{BERT}, \mathbf{h}_{sent}^{fix}])[CLS], \quad (7)$$

[1] https://www.yelp.com/dataset

	#Exp	#Doc	#Zero-shot	#Few-shot
Train	13477	638	1481	4003
Dev	2062	114	682	383
Test	3066	159	786	600

Table 1: Statistics on the VAST dataset.

where $\mathbf{h}^{CLS}$ is the hidden states of $[CLS]$ token in BERT. The hidden states vectors of $\mathbf{h}^{K}$ and $\mathbf{h}^{CLS}$ are concatenated to for classification:

$$p = Softmax(W[\mathbf{h}^{CLS}, \mathbf{h}^{K}] + b), \quad (8)$$

where W and b are the parameters and p is the probability distribution on the three stance labels.

Training. Given the input and its golden label (x_i, y_i), the loss function $\mathcal{L}_{cls}$ for classifying stance is cross entropy:

$$\mathcal{L}_{cls} = -\frac{1}{|N|} \sum_{(x_i, y_i)} y_i log\ p(y_i), \quad (9)$$

where $|N|$ is the number of data samples. To further ensure stronger topic invariance constraints of $\mathbf{h}_{KG}$, we add a shared decoder layer D_{recon} with a reconstruction loss:

$$\mathcal{L}_{recon} = -E_{\mathbf{h}^{KG}}(||D_{recon}(\mathbf{h}^{K}) - \mathbf{h}^{KG}||_2^2). \quad (10)$$

The overall loss function is:

$$\mathcal{L} = \mathcal{L}_{cls} + \mathcal{L}_{recon}. \quad (11)$$

4 Experiments

We verify the effectiveness of sentiment and common sense influence for zero-shot and few-shot stance detection. We also prove the significance of each module in our model in Section 4.4 and analyze the relationship between sentiment (common sense) and stance in Section 4.5 (4.6).

4.1 Settings

Dataset: We adopt the dataset for zero-shot and few-shot stance detection task–VAried Stance Topics (VAST) (Allaway and Mckeown, 2020), which is practical and useful for real-world applications. The dataset consists of thousands of topics, and the statistics are summarized in Table 1. The zero-shot topics only appear in the test set, and the few-shot topics only contain a few training examples.

Training Details We perform experiments using the official pre-trained BERT model provided by Huggingface[2]. For the pre-trained model with sentiment information, we adopt the model provided by Zhou et al. (2020), which is a continually trained BERT on sentiment datasets. We train our model on 1 GPU (Nvidia GTX2080Ti) using the Adam optimizer (Kingma and Ba, 2014). For training the graph autoencoder, the initial learning rate is 1e-2. For the stance detection training process, the initial learning rate is 1.5e-5, the max sequence length for BERT and SentiBERT is 256, the batch size for training is 4, and the model is trained for three epochs.

Baselines We compare our model with several state-of-the-art baselines: (1) **BiCond** (Augenstein et al., 2016), a model for cross-domain target stance detection task which uses one BiLSTM to encoding the topic and another BiLSTM to encoded the text; (2) **CrossNet** (Xu et al., 2018b), a model based on the BiCond adding an aspect-specific attention layer for cross-target setting; (3) **SENT** (Zhang et al., 2020), a model using the semantic-emotion heterogeneous graph to enhance BiLSTM for cross-traget stance detection; (4) **BERT-sep**, a model that encodes the text and topic separately, using BERT, and then classification with a two-layer feed-forward neural network; (5) **BERT-joint** (Allaway and Mckeown, 2020), a model with contextual conditional encoding followed by a two-layer feed-forward neural network; (6) **TGA-Net** (Allaway and Mckeown, 2020), a model using contextual conditional encoding and topic-grouped attention. In addition, we also consider the models BERT-joint-ft and TGA-Net-ft where the BERT module is fine-tuned; (7) **Prior-Bin:gold** (Jayaram and Allaway, 2021), a model applying human rationales as attributions to assist the stance detection; (8) **BERT-GCN** (Liu et al., 2021), a model applying the conventional GCN (Kipf and Welling, 2016), which considers node information aggregation; (9) **CKE-Net** (Liu et al., 2021), a model based on BERT, using the CompGCN (Vashishth et al., 2019) to obtain the commonsense information.

4.2 Results

The results are shown in Table 2. Compared with previous models, our model achieves the state-of-the-art performance in zero-shot, few-shot, and all the topics of VAST. In particular, the macro F1 scores are 72.6%, 70.2%, and 71.3%, which are 2.4%, 0.1%, and 1.2% higher than CKE-Net

[2]https://huggingface.co/

Model	F1 Zero-shot				F1 Few-Shot				F1 All			
	pro	con	neu	all	pro	con	neu	all	pro	con	neu	all
BiCond	.459	.475	.349	.427	.454	.463	.259	.392	.457	.468	.306	.410
Cross-Net	.462	.434	.404	.434	.508	.505	.410	.474	.486	.471	.408	.455
SEKT	.504	.442	.308	.418	.510	.479	.215	.474	.507	.462	.263	.411
BERT-sep	.414	.506	.454	.458	.524	.539	.544	.536	.473	.522	.501	.499
BERT-joint	.546	.584	.853	.660	.543	.597	.796	.646	.545	.591	.823	.653
TGA-Net	.554	.585	.858	.666	.589	.595	.805	.663	.573	.590	.831	.665
BERT-joint-ft	.579	.603	.875	.685	.595	.621	.831	.684	.588	.614	.853	.684
TGA-Net-ft	.568	.598	.885	.684	.628	.601	.834	.687	.599	.599	.859	.686
Prior-Bin:gold	**.643**	.581	.852	.692	.632	.563	**.881**	.692	**.652**	.597	.824	.691
BERT-GCN	.583	.606	.869	.686	.628	.634	.830	.697	.606	.620	.849	.692
CKE-Net	.612	.612	.880	.702	**.644**	.622	.835	.701	.629	.617	.857	.701
Our Model												
BS	.625	.667	.870	.717	.601	.667	.828	.699	.591	.669	.858	.706
S-RGCN	.582	.669	.838	.699	.561	.623	.809	.665	.607	.657	.842	.702
B-RGCN	.594	.657	.885	.712	.568	**.678**	.851	.699	.591	.663	.865	.706
BS-RGCN(*proposed*)	.608	**.674**	**.895**	**.726**	.600	.665	.839	**.702**	.604	**.669**	**.866**	**.713**

Table 2: Overall results. The suffix "ft" means BERT is fine-tuned. BS – the combination of BERT and SentiBERT; S-RGCN – the combination of SentiBERT and the graph autoencoder; B-RGCN – the combination of BERT and the graph autoencoder; BS-RGCN – our proposed model.

Model	Imp	mlT	mlS	Qte	Sarc
BERT-joint	.571	.590	.524	.634	.601
TGA-Net	.594	.605	.532	.661	.637
BERT-joint-ft	.617	.621	.547	.668	.673
BERT-GCN	.619	.627	.547	.668	.673
CKE-Net	**.625**	.634	.553	.695	.682
BS-RGCN	.621	**.647**	**.556**	**.701**	**.717**

Table 3: Accuracies on five challenges on the test set.

model, respectively. The results of B-RGCN (our model without SentiBERT module) are 71.2% and 69.9%, with a higher macro F1 score on zero-shot topics but a similar result on few-shot topics compared with CKE-Net. The performances of both our model and B-RGCN increase largely on the zero-shot topics but less on few-shot topics, which implies that our graph autoencoder module can achieve a similar effect compared with the GCN module of CKE-Net in the few-shot topics but can improve the effectiveness in extracting relation information in zero-shot topics. This verifies the intuition that only considering the relations between documents and topics limits the transferability of CKE-Net for the zero-shot task. Compared with Prior-Bin:gold, the macro F1 scores of our model are 3.4%, 1.0%, and 2.2% higher on zero-shot, few-shot, and all the topics sets, respectively. It implies that commonsense knowledge and sentiment information are more effective than the set of specific human rationales by Prior-Bin:gold as attributions.

Our model achieves better performance on *Con* labels (67.4%, 66.5%, 66.9%) compared with *Pro* labels (60.8%, 60.0%, 60.4%), which is similar to most of the previous models (BERT-GCN, TGA-Net, and so on). The phenomenon also appears in B-RGCN and BS, which are our models without SentiBERT and without BERT, respectively (the analysis of the ablation study is explained in Section 4.4 in detail). The results suggest that the use of SentiBERT does not cause the imbalanced performance on different stances and the detection difficulty is mainly on *Pro* labels. In addition, the results of *Neu* stance labels are the highest (89.5%, 83.9%, 86.6%) than those of other labels. It indicates that it is easier for models to classify the *Neu*, where the topics are mostly unrelated to documents.

4.3 Breakdown Evaluation

We also test our model on five special phenomena of the test set on VAST following Allaway and Mckeown (2020): (1) **Imp**: non-neutral stance examples where the topics are not explicit in the documents; (2) **mlT**: documents having multiple stance topics with different topics; (3) **mlS**: documents having multiple stance topics with different and non-neutral labels; (4) **Qte**: documents with quotations; (5) **Sarc**: documents with sarcasm.

The results are shown in Table 3. Our model achieves the state-of-the-art performance on mlT, mlS, Qte, and Sarc with 64.7%, 55.6%, 70.1%, and 71.7%, respectively. In particular, the improvement of our model on mlS implies that different types of knowledge features help models extract stance topics-related information. The most challenging

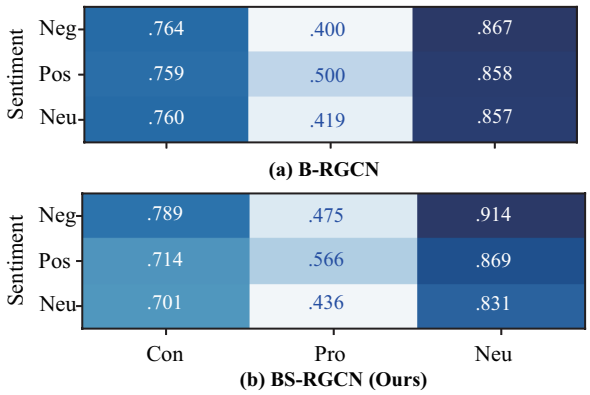

Figure 3: Accuracies of B-RGCN and BS-RGCN on all test data w.r.t different sentiment and stance pairs.

task is mlS, with a macro F1 score of 55.6% by our model. The results demonstrate that it is highly challenging to classify the topics with different stances since the stance information extracted in the model is more related to the whole sentence but more minor to the topics. The macro F1 score of Sarc increases the most, 3.5% higher than that of CKE-NET, implying that the sentiment information helps boost the model performance in understanding sarcasm, which is a sentiment-related linguistic phenomenon. The accuracy of our model on Imp is the second-highest (slightly lower than that of CKE-Net), which indicates that introducing commonsense graph knowledge can help improve the model performance on the zero-shot task.

4.4 Ablation Study

We conduct ablation studies of BS, S-RGCN, and B-RGCN to understand the significance of the graph autoencoder, BERT, and SentiBERT modules, respectively. The results are shown in Table 2. First, BS fuses BERT and SentiBERT feature vectors using Eq(5-6) and classifies the stance using $\mathbf{h}^{CLS}$ with a linear layer. It achieves macro F1 scores of 71.7%, 69.9%, and 70.6% on the zero-shot, few-shot, and all the topics, which are 3.2%, 1.5%, and 2.2% higher than those of BERT-joint-ft, respectively, which proves that sentiment information can help boost the performance of stance detection task.

Second, B-RGCN and S-RGCN are models without fusing the BERT and SentiBERT feature vectors. The feature vectors of $[CLS]$ tokens from BERT or SentiBERT (the parameters of SentiBERT are not fixed) are directly concatenated with knowledge graph feature vectors to classify the stance. The macro F1 scores of S-RGCN are 69.9% and 66.5% on the zero-shot topics and the few-shot topics, 1.3%, and 3.4% lower than those of B-RGCN, respectively. It indicates that it is not sufficient to use a sentiment-specific model to do stance classification. The macro F1 score of B-RGCN on the zero-shot set is 71.2%, 1.0% higher than that of CKE-Net, which shows that our graph autoencoder module can achieve better performance for zero-shot stance detection than CompGCN. However, BS, B-RGCN, and S-RGCN do not outperform BS-RGCN in the zero-shot topics and all the topics set, which shows that the graph autoencoder, BERT, and SentiBERT are all useful for the stance detection task.

4.5 Sentiment and Stance

Allaway and Mckeown (2020) indicate that models of BERT-Joint are reliant on sentiment cues, and the models learn the strong association between the *Neg* (negative) sentiment and the *Con* stance, yet weak association between *Pos* (positive) sentiment and *Pro* stance. Their analysis is based on experiments where the documents are corrupted by replacing the text's sentiment words. Here we take a different perspective and carry out experiments with respect to different stances and sentiment pairs on both B-RGCN and BS-RGCN. We use opinion lexicon (Hu and Liu, 2004) to classify the sentiment of document, (i.e, if a document contains more positive/negative words, we treat it as a document with the *Pos* (positive)/*Neg* (negative) sentiment; otherwise, we treat it as a document with the *Neu* (neutral) sentiment).

The results are shown in Figure 3 (the model trained on all the topics is tested in this experiment). For BS-RGCN, the accuracy on the corresponding stance and sentiment *(Neg, Con)* is 78.9%, higher than 71.4% of *(Pos, Con)* and 70.1% of *(Neu, Con)*. Similarly, the accuracy on *(Pos, Pro)* is 56.6%, higher than 47.5% of *(Neg, Pro)* and 43.6% of *(Neu, Pro)*. This suggests that data samples with corresponding sentiment and stance pairs (*(Pos, Pro), (Neg, Con)*) are easier to classify by our model. The performance of B-RGCN is similar to BS-RGCN, with an accuracy of 76.4% for *(Neg, Pro)*, a little higher than those of *(Pos, Con)* (75.9%) and *(Neu, Con)* (76.0%). The same model achieves an accuracy of 50% of *(Pos Pro)*, 10% higher than that of *(Neg, Pro)*, and 8.1% higher than that of *(Neu, Pro)*. The model without the sentiment module can also predict corresponding sentiment and

Context	Topic	Gold Label	Output
I have lived in brazil for the last five years (and off and on over the last 27 years). I know of no one here who is even remotely excited about the Olympics. It would seem that people donŕ care. The economy is tanking and government is at a complete standstill. We have more important things on our mind right now.	Olympics	Con	Con
I can't even believe that this is a debate. Cutting the most basic **foreign language programs**? How does one appreciate that there is a world outside of America? Google translate? Suny, everyone is laughing at you and you're too smug to notice.	College	Con	Con
Good idea. I have always had a cat or two. While being inhumane, **declawing** places a cat in danger. Should my charming indoor kitty somehow escape outside, he would have no way to defend himself.	nail removal	Con	Con

Table 4: Case Study for our trained stance detection model. Case I shows the effectiveness of using sentiment information; Case II shows the importance of commonsense knowledge; Case III shows both the sentiment and commonsense knowledge help the stance detection model.

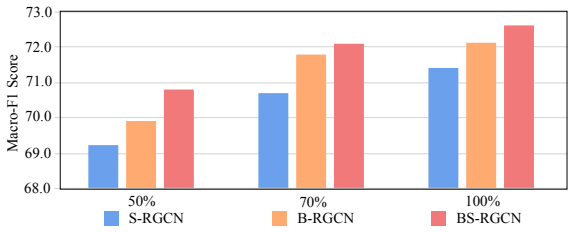

Figure 4: Macro-F1 scores of S-RGCN, B-RGCN, and BS-RGCN on zero-shot test data w.r.t different percents of commonsense knowledge for pre-training.

stance pairs with higher accuracy, demonstrating that sentiment information can help stance detection models. The accuracies for B-RGCN and BS-RGCN are both significantly higher on data with *Con* stances than those with *Pro* stances. The phenomenon indicates that it is difficult for models to predict *Pro* stance in the VAST dataset, and the difference in performance is not caused by the difference of associations between data of *(Pos, Pro)* and *(Neg, Con)*. For the data of *Neu* stance, the performance is less related to sentiments. The models can achieve much better results on *Neu* stance data, where the topics may be not related to the documents, 91.4% on *(Neg, Neu)*, 86.9% on *(Pos, Neu)*, 83.1% on *(Neu, Neu)* for BS-RGCN , and 86.7% on *(Neg, Neu)*, 85.8% on *(Pos, Neu)*, 85.7% on *(Neu, Neu)* for B-RGCN. The phenomenon demonstrates that it is easy for the model to judge whether the topic is related to the documents.

4.6 Common Sense and Stance

We show the relationship between common sense and stance by pre-training the graph autoencoder w.r.t different percentages of extracted concepts (Section 3.1). Using the commonsense feature with the pre-trained autoencoder, we show the results of the stance detection models B-GCN, S-GCN, and BS-RGCN on the zero-shot task. The results are given in Figure 4. As observed, the performance of the three models increases with increasing coverage of commonsense knowledge. It indicates that commonsense knowledge is directly useful for stance detection models.

4.7 Case Study

We also show some cases from the test data using the model trained on all the topics. In the first case, sentiment words such as 'tanking' or 'standstill' imply the negative sentiments towards the influence of the Olympics on the economy of Brazil, which further expresses an opposing stance towards 'Olympics'. Our model outputs the correct label towards the target thanks to the sentiment information. In the second case, no explicit expression of the target 'College' is contained in the document. Only some implications, including 'foreign language programs', have relation to the 'College', and with the commonsense knowledge encoding, our model outputs the correct stance. The third case proves that both common sense and sentiment information can benefit the stance detection model, that 'inhumane' expresses a negative sentiment, and the topic 'nail removal' is implicitly involved by the word 'declawing'. Our model can also give the correct stance for case III.

5 Conclusion

We proposed a stance detection model incorporating commonsense knowledge and sentiment information, achieving state-of-the-art zero-shot and few-shot stance detection results on the standard

dataset. The ablation study showed the significance of each module, such as knowledge graph autoencoder, SentiBERT, and BERT. We also analyzed the relation between sentiment/common sense and stance, which indicate the effectiveness of this external knowledge.

Acknowledgements

Yue Zhang is the corresponding author. We would also like to thank the anonymous reviewers for the detailed and thoughtful reviews. The work is funded by the Zhejiang Province Key Project 2022SDXHDX0003.

References

Emily Allaway and Kathleen Mckeown. 2020. Zero-shot stance detection: A dataset and model using generalized topic representations. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 8913–8931.

Emily Allaway, Malavika Srikanth, and Kathleen McKeown. 2021. Adversarial learning for zero-shot stance detection on social media. *arXiv preprint arXiv:2105.06603*.

Pranav Anand, Marilyn Walker, Rob Abbott, Jean E Fox Tree, Robeson Bowmani, and Michael Minor. 2011. Cats rule and dogs drool!: Classifying stance in online debate. In *Proceedings of the 2nd Workshop on Computational Approaches to Subjectivity and Sentiment Analysis (WASSA 2.011)*, pages 1–9.

Isabelle Augenstein, Tim Rocktäschel, Andreas Vlachos, and Kalina Bontcheva. 2016. Stance detection with bidirectional conditional encoding. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 876–885, Austin, Texas. Association for Computational Linguistics.

Xuefeng Bai, Yulong Chen, Linfeng Song, and Yue Zhang. 2021. Semantic representation for dialogue modeling. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 4430–4445, Online. Association for Computational Linguistics.

Beata Beigman Klebanov, Eyal Beigman, and Daniel Diermeier. 2010. Vocabulary choice as an indicator of perspective. In *Proceedings of the ACL 2010 Conference Short Papers*, pages 253–257, Uppsala, Sweden. Association for Computational Linguistics.

Marcelo Dias and Karin Becker. 2016. Inf-ufrgs-opinion-mining at semeval-2016 task 6: Automatic generation of a training corpus for unsupervised identification of stance in tweets. In *Proceedings of the 10th International Workshop on Semantic Evaluation (SemEval-2016)*, pages 378–383.

Jiachen Du, Ruifeng Xu, Yulan He, and Lin Gui. 2017. Stance classification with target-specific neural attention networks. In *Proceedings of the 26th International Joint Conference on Artificial Intelligence*, pages 3988–3994.

Adam Faulkner. 2014. Automated classification of stance in student essays: An approach using stance target information and the wikipedia link-based measure. In *FLAIRS Conference*.

Deepanway Ghosal, Devamanyu Hazarika, Abhinaba Roy, Navonil Majumder, Rada Mihalcea, and Soujanya Poria. 2020. KinGDOM: Knowledge-Guided DOMain Adaptation for Sentiment Analysis. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 3198–3210, Online. Association for Computational Linguistics.

Momchil Hardalov, Arnav Arora, Preslav Nakov, and Isabelle Augenstein. 2021. Cross-domain label-adaptive stance detection. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 9011–9028, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Momchil Hardalov, Arnav Arora, Preslav Nakov, and Isabelle Augenstein. 2022. Few-shot cross-lingual stance detection with sentiment-based pre-training. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 36, pages 10729–10737.

Kazi Saidul Hasan and Vincent Ng. 2013. Stance classification of ideological debates: Data, models, features, and constraints. In *Proceedings of the Sixth International Joint Conference on Natural Language Processing*, pages 1348–1356, Nagoya, Japan. Asian Federation of Natural Language Processing.

Kazi Saidul Hasan and Vincent Ng. 2014. Why are you taking this stance? identifying and classifying reasons in ideological debates. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 751–762, Doha, Qatar. Association for Computational Linguistics.

Minqing Hu and Bing Liu. 2004. Mining and summarizing customer reviews. In *Proceedings of the tenth ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 168–177.

Sahil Jayaram and Emily Allaway. 2021. Human rationales as attribution priors for explainable stance detection. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 5540–5554, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Diederik P Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*.

Thomas N Kipf and Max Welling. 2016. Semi-supervised classification with graph convolutional networks. *arXiv preprint arXiv:1609.02907*.

Chang Li, Aldo Porco, and Dan Goldwasser. 2018. Structured representation learning for online debate stance prediction. In *Proceedings of the 27th International Conference on Computational Linguistics*, pages 3728–3739, Santa Fe, New Mexico, USA. Association for Computational Linguistics.

Pengfei Li, Kezhi Mao, Xuefeng Yang, and Qi Li. 2019. Improving relation extraction with knowledge-attention. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 229–239, Hong Kong, China. Association for Computational Linguistics.

Yingjie Li and Cornelia Caragea. 2019. Multi-task stance detection with sentiment and stance lexicons. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 6299–6305, Hong Kong, China. Association for Computational Linguistics.

Wei-Hao Lin, Theresa Wilson, Janyce Wiebe, and Alexander Hauptmann. 2006. Which side are you on? identifying perspectives at the document and sentence levels. In *Proceedings of the Tenth Conference on Computational Natural Language Learning (CoNLL-X)*, pages 109–116, New York City. Association for Computational Linguistics.

Rui Liu, Zheng Lin, Yutong Tan, and Weiping Wang. 2021. Enhancing zero-shot and few-shot stance detection with commonsense knowledge graph. In *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*, pages 3152–3157, Online. Association for Computational Linguistics.

Zhibin Liu, Zheng-Yu Niu, Hua Wu, and Haifeng Wang. 2019. Knowledge aware conversation generation with explainable reasoning over augmented graphs. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 1782–1792, Hong Kong, China. Association for Computational Linguistics.

Robert Logan, Nelson F. Liu, Matthew E. Peters, Matt Gardner, and Sameer Singh. 2019. Barack's wife hillary: Using knowledge graphs for fact-aware language modeling. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 5962–5971, Florence, Italy. Association for Computational Linguistics.

Saif Mohammad, Svetlana Kiritchenko, Parinaz Sobhani, Xiaodan Zhu, and Colin Cherry. 2016. SemEval-2016 task 6: Detecting stance in tweets. In *Proceedings of the 10th International Workshop on Semantic Evaluation (SemEval-2016)*, pages 31–41, San Diego, California. Association for Computational Linguistics.

Saif M. Mohammad, Parinaz Sobhani, and Svetlana Kiritchenko. 2017. Stance and sentiment in tweets. *ACM Trans. Internet Technol.*, 17(3).

Jianmo Ni, Jiacheng Li, and Julian McAuley. 2019. Justifying recommendations using distantly-labeled reviews and fine-grained aspects. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 188–197.

Matthew E. Peters, Mark Neumann, Robert Logan, Roy Schwartz, Vidur Joshi, Sameer Singh, and Noah A. Smith. 2019. Knowledge enhanced contextual word representations. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 43–54, Hong Kong, China. Association for Computational Linguistics.

Minghui Qiu, Yanchuan Sim, Noah A Smith, and Jing Jiang. 2015. Modeling user arguments, interactions, and attributes for stance prediction in online debate forums. In *Proceedings of the 2015 SIAM international conference on data mining*, pages 855–863. SIAM.

M. Schlichtkrull, Thomas Kipf, Peter Bloem, Rianne van den Berg, Ivan Titov, and Max Welling. 2018. Modeling relational data with graph convolutional networks. *ArXiv*, abs/1703.06103.

Parinaz Sobhani, Saif Mohammad, and Svetlana Kiritchenko. 2016. Detecting stance in tweets and analyzing its interaction with sentiment. In *Proceedings of the fifth joint conference on lexical and computational semantics*, pages 159–169.

Dhanya Sridhar, James Foulds, Bert Huang, Lise Getoor, and Marilyn Walker. 2015. Joint models of disagreement and stance in online debate. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 116–125, Beijing, China. Association for Computational Linguistics.

Shikhar Vashishth, Soumya Sanyal, Vikram Nitin, and Partha Talukdar. 2019. Composition-based multi-relational graph convolutional networks. *arXiv preprint arXiv:1911.03082*.

Marilyn A. Walker, Pranav Anand, Robert Abbott, and Ricky Grant. 2012. Stance classification using dialogic properties of persuasion. In *Proceedings of*

the 2012 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL HLT '12, page 592–596, USA. Association for Computational Linguistics.

Wan Wei, Xiao Zhang, Xuqin Liu, Wei Chen, and Tengjiao Wang. 2016. pkudblab at semeval-2016 task 6: A specific convolutional neural network system for effective stance detection. In *Proceedings of the 10th international workshop on semantic evaluation (SemEval-2016)*, pages 384–388.

Chang Xu, Cecile Paris, Surya Nepal, and Ross Sparks. 2018a. Cross-target stance classification with self-attention networks. *arXiv preprint arXiv:1805.06593*.

Chang Xu, Cécile Paris, Surya Nepal, and Ross Sparks. 2018b. Cross-target stance classification with self-attention networks. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 778–783, Melbourne, Australia. Association for Computational Linguistics.

Bishan Yang, Wen-tau Yih, Xiaodong He, Jianfeng Gao, and Li Deng. 2014. Embedding entities and relations for learning and inference in knowledge bases. *arXiv preprint arXiv:1412.6575*.

Guido Zarrella and Amy Marsh. 2016. Mitre at semeval-2016 task 6: Transfer learning for stance detection. *Proceedings of SemEval*, pages 470–475.

Bowen Zhang, Min Yang, Xutao Li, Yunming Ye, Xiaofei Xu, and Kuai Dai. 2020. Enhancing cross-target stance detection with transferable semantic-emotion knowledge. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 3188–3197, Online. Association for Computational Linguistics.

Shaodian Zhang, Lin Qiu, Frank Chen, Weinan Zhang, Yong Yu, and Noémie Elhadad. 2017. We make choices we think are going to save us: Debate and stance identification for online breast cancer cam discussions. In *Proceedings of the 26th International Conference on World Wide Web Companion*, pages 1073–1081.

Jie Zhou, Junfeng Tian, Rui Wang, Yuanbin Wu, Wenming Xiao, and Liang He. 2020. Sentix: A sentiment-aware pre-trained model for cross-domain sentiment analysis. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 568–579.

A Appendix

Human Labeling for Sentiment and Stance Detection

In this part, we manually label some samples (randomly selected) from VAST dataset to prove the relation between sentiments and stances. Opinion lexicon (Hu and Liu, 2004) is adopted as the sentiment vocabulary. As shown in Table 5, there are many samples (7 in 10) that sentiment knowledge plays a significant role for stance detection, and few samples have a conflicting relation.

Context	Topic	Stance	Relation
The reason that **Deep Mind** winning is so impressive is that Google managed to accomplish this with virtually no warning. It was less than a year ago where the best **computer program** was not in the top 20,000 in the world. It was less than 6 months ago when the program beat a player in the top 1,000. Yesterday the program beat the the best player in the world. Am I wrong to be shocked at how fast complicate **AI** has advanced?	Artificial Intelligence	Pro	+
I totally agree with this premise. As a younger person I was against **Nuclear power** (I was in college during 3 mile island) but now it seems that nuclear should be in the mix. Fission technology is better, and will continue to get better if we actively promote its development. The prospect of fusion energy also needs to be explored. If it's good enough for the sun and the stars, it's good enough for me.	Nuclear Power	Pro	+
This is a horrible idea. Anyone who has worked on the border, or in **Mexico** (as I do), knows there are plenty of middle and upper-class Mexicans who come to the U.S. for an education. I think Dr. Lee is really perpetuating stereotypes here. In my opinion, affirmative action should be based on economic class, no matter what the race.	Mexico	Pro	0
Good idea. I have always had a cat or two. While being inhumane, **declawing** places a cat in danger. Should my charming indoor kitty somehow escape outside, he would have no way to defend himself. Why don't humans have their finger-and tonails removed to save on manicures? Answer: they are important to the functioning and protection of our bodies.	nail removal	Con	+
The mandate of **private corporations** is to make a profit. And if the profit is made at the EXPENSE of the society that allow the **corporation** to exist, well, too bad. **Oil companies** foul the environment. **Financial companies** drive the economy into the Great Recession. **Airlines** have no regard for the people they transport. As long as they make a profit, they are allowed to abuse the public until they are stopped. That is the way it has been since Swift and Armour canned and sold rotten meat and Carnegie sent Pinkertons to shoot striking miners.	private corporation profit	Con	+
One's own, and learning another language is important and a great work out for the brain! Back in the day, I learned Spanish! In retrospect **Latin** would have been the better way to go, since mastery of that makes learning the languages like French, Italian, Portuguese, Romanian, and Spanish, very much easier to learn!	Latin helpful language	Pro	+
Without government to ensure their behavior, **companies** will attempt to make a profit even to the DETRIMENT of the society that supports the business. We have seen this in the environment, in finances, in their treatment of workers and customers. Enough.	company	Pro	0
The "you have a short live, so enjoy" attitude alone did not lead to the **Renaissance**, the age of Enlightment, or the Industrial Revolution. It did not le ()ad to the invention of the light bulb, or the telephone, or the internet, or the NYT electronic discussion board. Just "enjoying" life alone means you are enjoying the fruit of someone elseś hard work.	Renaissance	Pro	0
Of course their salaries should be raised. But this should be separated from the discussion about legality. Salaries should be raised and only legal workers should be employed. Its really a no brainer. And any discussion about only Mexicans being prepared to do this work so it has to be **illegal** is completely disingenuous.	illegal labor	Con	+
Also, and usually not acknowledged, is that we are slowly eroding the fertility of the soil. There is no more usable **soil**, we are farming everything that can be farmed. Current methods depend on petrochemical fertilizers. Even with their use, **fertility** is slowly declining. As human population continues to grow, the result is obvious.	soil	Con	+

Table 5: Manually labeling samples for the relation between sentiments and stances. The positive/negative words related to the topics are labeled with red/teal colors. The topic related words in the documents are bold. '+' for sentiment words supporting the stance, '0' for no relation.

Modeling Intra- and Inter-Modal Relations: Hierarchical Graph Contrastive Learning for Multimodal Sentiment Analysis

Zijie Lin [1], Bin Liang [1]*, Yunfei Long [2], Yixue Dang [3], Min Yang [4], Min Zhang [1], Ruifeng Xu [1,5,6]*

[1] Harbin Institute of Technology, Shenzhen, China
[2] University of Essex, Colchester, UK [3] China Merchants Securities Co ., Ltd.
[4] Shenzhen Institute of Advanced Technology, Chinese Academy of Sciences
[5] Peng Cheng Laboratory, Shenzhen, China
[6] Guangdong Provincial Key Laboratory of Novel Security Intelligence Technologies
lzjjeffery@163.com, 18B951033@stu.hit.edu.cn
xuruifeng@hit.edu.cn

Abstract

The existing research efforts in Multimodal Sentiment Analysis (MSA) have focused on developing the expressive ability of neural networks to fuse information from different modalities. However, these approaches lack a mechanism to understand the complex relations within and across different modalities, since some sentiments may be scattered in different modalities. To this end, in this paper, we propose a novel hierarchical graph contrastive learning (HGraph-CL) framework for MSA, aiming to explore the intricate relations of intra- and inter-modal representations for sentiment extraction. Specifically, regarding the intra-modal level, we build a unimodal graph for each modality representation to account for the modality-specific sentiment implications. Based on it, a graph contrastive learning strategy is adopted to explore the potential relations based on unimodal graph augmentations. Furthermore, we construct a multimodal graph of each instance based on the unimodal graphs to grasp the sentiment relations between different modalities. Then, in light of the multimodal augmentation graphs, a graph contrastive learning strategy over the inter-modal level is proposed to ulteriorly seek the possible graph structures for precisely learning sentiment relations. This essentially allows the framework to understand the appropriate graph structures for learning intricate relations among different modalities. Experimental results on two benchmark datasets show that the proposed framework outperforms the state-of-the-art baselines in MSA.

1 Introduction

Multimodal Sentiment Analysis (MSA) has received increasing research attention in recent years. Different from textual sentiment analysis, MSA generally contains three modalities: text (caption), visual and acoustic, in which the visual and acoustic information can complement the text for identifying the sentiment score and thus preferably detecting the sentiment polarity (e.g. *positive*, *negative*, etc.). As shown in Figure 1, although the text modality expresses the positive sentiment, we can infer that the correct sentiment of this example is *negative* in the light of the study of visual and acoustic information. Therefore, dealing with MSA needs to consider learning and fusing the information from different modalities.

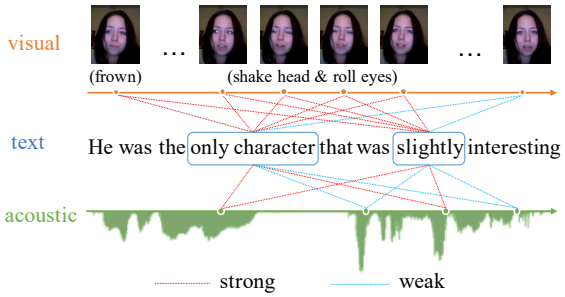

Figure 1: An example of intricate relations among different modalities in MSA.

Early MSA work attempted to fuse the information from different modalities by tensor-based features fusion (Snoek et al., 2005; Zadeh et al., 2017; Liu et al., 2018) or attention-based features fusion (Zadeh et al., 2018b,a; Tsai et al., 2019a). Furthermore, some representation learning-based approaches (Tsai et al., 2019b; Hazarika et al., 2020) aim to model the consistency and the variability between modalities for extracting the sentiment cues among modalities or consider both fusion and alignment of multimodal sequential data with a graph model (Yang et al., 2021). Despite the promising progress made by current work, they generally focus on fusing multimodal representations via class-driven supervised learning or multi-task learning, which fails to understand the intricate

* Corresponding author.

Proceedings of the 29th International Conference on Computational Linguistics, pages 7124–7135
October 12–17, 2022.

relations within and across modalities for better sentiment extraction. As shown in Figure 1, the sentiment is scattered within each modality and across different modalities.

In this paper, we study how to understand representations within and across modalities, enabling the highly correlated modal representations to be explicitly linked for learning the multimodal sentiment information. To reach this goal, in the light of developing the merit of graph structure for modeling intricate relations of representations, we first build a unimodal graph for each modality and further build a multimodal graph for each instance based on the unimodal graphs. Concretely, for the intra-modal graphs, to account for the underlying relations within each modality, we construct a syntax-aware graph for the text modality based on the dependency tree of the sentence and build sequential connection graphs for visual and acoustic modality. For the inter-modal graph, we build a fully-connected inter-modal graph based on the modality-specific graphs to capture the potential relations across different modalities. Then, we apply a graph attention networks (Veličković et al., 2018) architecture to model the semantic relations by means of specifying different weights to different nodes in a neighborhood, without requiring any costly matrix operation (such as inversion) or depending on knowing the graph structure upfront.

Following that, we propose a hierarchical graph contrastive learning (HGraph-CL) framework to model the correlation and difference of graph information within a specific modality and further across different modalities. Specifically, for the intra-modal level, inspired by You et al. (2020), we first devise a self-supervised graph contrastive learning strategy based on the graph augmentations, aiming to explore more appropriate graph structures and derive robust graph representations for each modality. In addition, inspired by Khosla et al. (2020); Gunel et al. (2021), we employ a supervised contrastive learning strategy to make sense of the correlation and difference between different classes, so as to capture the similarity between examples in one class and contrast them with examples in other classes. Moreover, for the inter-modal level, we also perform these two contrastive learning strategies to learn the graph representations for better generalizability, transferability, and robustness in learning sentiment cues compared with pure class-driven methods.

The main contributions of our work can be summarized as follows:

(1) The MSA task is approached from a novel perspective that explores intra- and inter-modality graph construction to leverage the potential sentiment relations within and across modalities.

(2) A novel hierarchical graph contrastive learning (HGraph-CL) framework is devised for better sentiment relations extraction at an intra-modal level and further at an inter-modal level.

(3) Performance evaluation on two benchmark datasets shows the superiority and robustness of the proposed framework compared to several competitive baselines.

2 Related Work

2.1 Multimodal Sentiment Analysis

The MSA task aims to predict sentiment polarity by aiding text with visual and acoustic information. Since the raw visual and acoustic data is in frames and the text is in words, MSA can be broadly classified into word-level and utterance-level depending on the granularity of the data used. Among them, utterance-level approaches (Zadeh et al., 2017; Liu et al., 2018; Yu et al., 2021) perform modal fusion in global representation, while word-level approaches (Tsai et al., 2019b; Wang et al., 2019; Tsai et al., 2019a; Rahman et al., 2020; Hazarika et al., 2020; Wu et al., 2021) are more concerned with local modal interactions. Furthermore, Rahman et al. (2020) proposes a Multimodal Adaptation Gate (MAG) mechanism to perform modal fusion at word-level, which does not rely on complex structure and can be embedded in pre-trained attention models. Hazarika et al. (2020) proposes a multimodal representation learning framework to model modality-invariant and modality-specific information within the example by projecting each modality to two distinct subspaces. Besides, Yu et al. (2021) trains MSA together with three unimodal sentiment analysis tasks, and proposes a heuristic approach to generate unimodal labels. Yang et al. (2021) proposes a novel graph-based neural network to analyze multimodal sequential data for MSA. We propose a novel graph-based approach to address MSA's modal interactions and fusion problem. In contrast to the existing graph works in MSA, our proposed method constructs intra-modal graphs based on prior knowledge of the modalities (e.g., textual dependencies). Furthermore, we create edges between any two nodes from different

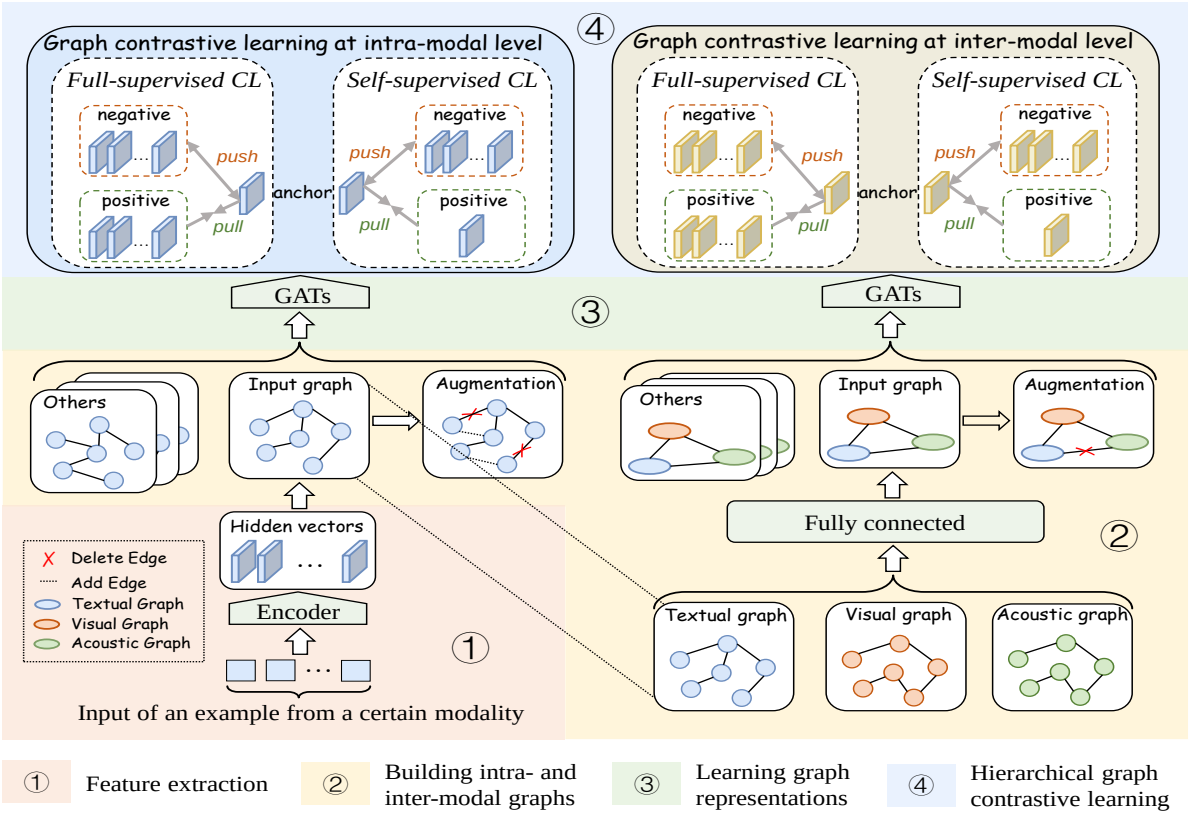

Figure 2: The architecture of the proposed HGraph-CL framework.

intra-model graphs for more complex inter-modal relationships to preserve any possible association.

2.2 Contrastive Learning

Our work also relates to contrastive learning. Contrastive learning (CL) is originally proposed as a self-supervised learning method for solving the lack of supervised signals (Chen et al., 2020; Liu et al., 2021). However, CL often requires effective data augmentation as a foundation. Recent work (Khosla et al., 2020) proposes supervised contrastive learning methods in combination with class information, which is capable of learning the class distribution of examples and without data augmentation. Mai et al. (2022) proposes a hybrid contrastive learning strategy for MSA, but lacks exploring the potential relationship within and among modalities. The combination with graph networks is another new application of contrastive learning (You et al., 2020; Zhu et al., 2020). The graph networks can model the association between nodes, and data augmentation on graph structures is feasible and operable. Common augmentation methods include additions and deletions of nodes or edges, masking of the representations of nodes or edges, etc. Therefore, to explore more appropriate graph structures, inspired by You et al. (2020), we apply the graph augmentations by deleting and adding edges in graphs, and thus derive multifarious but similar graph structures with respect to the source.

3 Method

In this section, we describe the proposed HGraph-CL framework in detail. As illustrated in Figure 2, the framework mainly consists of four components:

1) **Feature extraction**, which applies BERT (Devlin et al., 2019) and BiLSTMs (Hochreiter and Schmidhuber, 1997) to extract features from the three modalities of text, images and audio.

2) **Building intra- and inter-modal graphs**, which constructs intra- and inter-modal graphs based on the hidden vectors learned from the text, images, and audio.

3) **Learning graph representations**, which learns intra- and inter-modal graph representations and leverages the potential sentiment relations within and across modalities. We believe a graph network can model the complex relationship between different modalities while updating the node representations.

4) **Hierarchical graph contrastive learning**, which performs contrastive learning based on heterogeneous graphs at the intra-modal level and inter-modal level. Contrastive learning can help the model understand the similarity and differences of the data across different modalities. Moreover, subtle differences in the graphs may also affect the learning of models on samples. Therefore, we propose the hierarchical graph contrastive learning strategy to augment the learning of the graph representations at both the data level and label level.

3.1 Task Definition

Formally, supposing there is an example consisting of a text t and the corresponding image frames v and audio a from a video, the goal of multimodal sentiment analysis (MSA) is to predict a sentiment score y, which is a constant from -3.0 to 3.0, for each example. Additionally, according to the sentiment score y, we thus identify the sentiment polarity (i.e. *positive* if $y > 0$ or *negative* if $y < 0$).

3.2 Feature Extraction

Given an input example with L tokens for each modality $\boldsymbol{x} = (\boldsymbol{x}^t, \boldsymbol{x}^v, \boldsymbol{x}^a)$, where t,v, and a denote the text, visual, and acoustic modalities respectively. Then, three encoders are used to extract features from the three modality data. Among them, the encoder of text modality is BERT (Devlin et al., 2019), which takes the text representation $\boldsymbol{x}^t$ as input to derive the hidden representations $\boldsymbol{e}^t \in \mathbb{R}^{L \times d_t}$ of all the tokens of the text modality:

$$\boldsymbol{e}^t = \text{BERT}([CLS]\boldsymbol{x}^t[SEP])_{1:L} \quad (1)$$

where d_t is the dimension of text hidden vectors.

We use Facet (Zhu et al., 2006) to extract a set of visual features, including facial markers, facial action units, head pose, visual trajectory, and HOG features. And we use COVAREP (Degottex et al., 2014) to extract a set of low-level acoustic features, including 12 mel cepstral coefficients (MFCCs), pitch tracking and turbid/clear segmentation features, gating source parameters, peak slope parameters, and maximum dispersion quotient. The visual/acoustic features are aligned with the text at token level by averaging the frames of video/audio recording over the time interval align to a token. The lengths of obtained sequences $\boldsymbol{x}^v$ and $\boldsymbol{x}^a$ are the same as the text sequences $\boldsymbol{x}^t$.

Owing to the sequential structure of visual and acoustic modalities, we adopt BiLSTM (Hochreiter and Schmidhuber, 1997) as visual and acoustic encoders to embed each token into a d_v-dimensional vector and a d_a-dimensional vector respectively. Here, the encoded visual representations $\boldsymbol{e}^v \in \mathbb{R}^{L \times d_v}$ and acoustic representations $\boldsymbol{e}^a \in \mathbb{R}^{L \times d_a}$ are computed as follow:

$$\boldsymbol{e}^v = \text{BiLSTM}(\boldsymbol{x}^v) \quad (2)$$

$$\boldsymbol{e}^a = \text{BiLSTM}(\boldsymbol{x}^a) \quad (3)$$

Then, the representations of the three modalities are mapped to the same d_h-dimensional hidden vector space using three projection layers $\text{p}^t(\cdot), \text{p}^v(\cdot), \text{p}^a(\cdot)$, which are presented as $\boldsymbol{h}^t, \boldsymbol{h}^v, \boldsymbol{h}^a \in \mathbb{R}^{L \times d_h}$. A projection layer is a dense layer with a ReLU activation function.

3.3 Building Intra- and Inter-Modal Graphs

This section describes how to construct the intra-modal and inter-modal graphs for each multimodal instance. Inspired by Kipf and Welling (2017), for both intra- and inter-modal graphs, we build the graphs to be undirected and set a self-loop for each node, to make use of more sufficient relations of the sentiment expression of nodes.

Intra-Modal Graph To leverage the intricate sentiment implications within each modality, we first build three intra-modal graphs to explicitly account for the modality-specific relations of the representations towards the three modalities of a multimodal instance.

To be specific, for the text-modality graph, to leverage the syntax-aware relations of the textual information, inspired by Zhang et al. (2019); Liang et al. (2020, 2022), we construct the text modality graph $\boldsymbol{G}^t \in \mathbb{R}^{L \times L}$ based on the dependency tree of the sentence*. That is, we link the context tokens if there is a relation between these two tokens in the dependency tree. For visual and acoustic modalities, the nodes are averaged video/audio tokens alighed to textual tokens, described in Section 3.2. Since the input representations of these two modalities are sequential, we connect the adjacent nodes in the sequences for the visual modality graph $\boldsymbol{G}^v \in \mathbb{R}^{L \times L}$ and the acoustic modality graph $\boldsymbol{G}^a \in \mathbb{R}^{L \times L}$ to capture the sequence relations of these two modalities.

Inter-Modal Graph To fuse the multimodal representations for extracting the sentiment implica-

*We use the spaCy toolkit to obtain the dependency tree of a sentence: https://spacy.io/.

tions produced by multiple modalities, we construct an inter-modal graph $\boldsymbol{G}^m \in \mathbb{R}^{3L\times 3L}$ for each multimodal instance based on the derived intra-modal graphs. Specifically, we first combine the three intra-modal graphs and then employ a fully connected solution to link the cross-modal tokens among the intra-modal graphs of the three modalities, to capture the potentially scattered relations of the multimodal instance.

3.4 Learning Graph Representations

Based on the intra- and inter-modal graphs derived in Section 3.3, we employ Graph Attention Network (Veličković et al., 2018) to update the nodes in the graphs by aggregating the information from the neighborhoods with varying weights. Specifically, in a GAT layer, for each neighbor, the representations of the current node *i* and the neighbor *j* are concatenated and then mapped to a scalar s_{ij} as the attention coefficient. Then normalizing the attention coefficients of all neighbors by softmax.

$$s_{ij} = \text{LeakyReLU}(\boldsymbol{a}[\boldsymbol{W}h_i \| \boldsymbol{W}h_j]) \quad (4)$$

$$\alpha_{ij} = \text{softmax}_j(s_{ij}) = \frac{\exp(s_{ij})}{\sum_{k\in\mathcal{N}_i} s_{ik}} \quad (5)$$

where $\boldsymbol{a}$ is a weight vector, W is a weight matrix, and $||$ is the concatenation operation. $\mathcal{N}_i$ denotes the set of node *i* and its neighbors. Finally, the representation of node *i* is updated with a weighted sum of the representations of neighbors and itself, and multi-head attention mechanism is applied to stabilize the learning process of self-attention.

$$\tilde{h}_i = \|_{k=1}^{K} \sigma\left(\sum_{j\in\mathcal{N}_i} \alpha_{ij}^k \boldsymbol{W}^k h_j\right) \quad (6)$$

where *k* denotes the *k*-th attention head and σ is a sigmoid function to provide non-linearity.

This mechanism essentially allows the model to make sense of the intra- and inter-modal sentiment associations by modeling the relations in the graphs with the GAT operation. For example, the edge weight between a "smile"(visual) node and a "happy"(textual) node should be higher than the edge weight between a "frown"(visual) node and the "happy"(textual) node.

For intra-modal graphs, the hidden representations $\boldsymbol{h}^t$, $\boldsymbol{h}^v$, and $\boldsymbol{h}^a$ of the three modalities and the corresponding graphs $\boldsymbol{G}^t$, $\boldsymbol{G}^v$, and $\boldsymbol{G}^a$ are fed as inputs into the GAT layers (GATs) to derive the unimodal graph representations $\boldsymbol{r}^t, \boldsymbol{r}^v, \boldsymbol{r}^a \in \mathbb{R}^{d_h}$:

$$\boldsymbol{r}^t = \text{READOUT}^t(\text{GATs}(\boldsymbol{h}^t, \boldsymbol{G}^t)) \quad (7)$$

$$\boldsymbol{r}^v = \text{READOUT}^v(\text{GATs}(\boldsymbol{h}^v, \boldsymbol{G}^v)) \quad (8)$$

$$\boldsymbol{r}^a = \text{READOUT}^a(\text{GATs}(\boldsymbol{h}^a, \boldsymbol{G}^a)) \quad (9)$$

where $\text{GATs}(\cdot)$ denotes the operation of GAT layers. Note that following You et al. (2020), we use a $\text{READOUT}(\cdot)$ function to aggregate the node representations to derive the graph representation.

On the other hand, for an inter-modal graph, we also apply GATs to model the relations in the graph, aiming to extract the relations between modalities for better learning sentiment cues. Given the inter-model graph $\boldsymbol{G}^m$ of a multimodal instance, the corresponding nodes are represented as all nodes from $\boldsymbol{h}^t$, $\boldsymbol{h}^v$ and $\boldsymbol{h}^a$, and the multimodal graph representation $\boldsymbol{r}^m \in \mathbb{R}^{d_h}$ is derived as follow:

$$\boldsymbol{r}^m = \text{READOUT}^m(\text{GAT}([\boldsymbol{h}^t \| \boldsymbol{h}^v \| \boldsymbol{h}^a], \boldsymbol{G}^m)) \quad (10)$$

To this end, owing to the merit of graph attention networks that the weights of associated edges can be adjusted according to the attention mechanism during the training process, the degree of association of nodes on a fully connected graph can be quantified by the weights of the edges, and thus deriving appropriate graph representation.

3.5 Hierarchical Graph Contrastive Learning

In this section, we detail the proposed hierarchical graph contrastive learning strategy in our HGraph-CL framework. As shown in Figure 2, our hierarchical graph contrastive learning strategy first performs at the intra-modal level, and further performs at the inter-modal level. Here for each level, we devise a fully-supervised contrastive loss based on the sentiment labels to improve the graph representation for better sentiment learning and a self-supervised contrastive loss based on the graph augmentations to explore more appropriate graph structures for deriving precise graph representation.

3.5.1 Graph Contrastive Learning at Intra-Modal Level

Fully-Supervised Contrastive Loss Based on Sentiment Labels Inspired by the work on fully-supervised contrastive learning (Khosla et al., 2020; Gunel et al., 2021), for a mini-batch, we adopt sentiment labels as the supervised signal to perform fully-supervised loss for capturing the similarity between examples in one class and contrasting them

with examples in other classes. Specifically, given N examples in a mini-batch, the examples can be divided into $S_1, S_2, ...$ according to sentiment polarity. Considering the binary classification, then $|S_1| = M, |S_2| = N - M, |\cdot|$ is the cardinality of the set. For the *anchor* example $s_i \in S_1$, a *positive* pair can be represented as (s_i, s_j), here $s_j \in S_1, j \neq i$. While the rest $N - M$ samples are regarded as *negative* examples. The pairwise objective $\ell_1(\boldsymbol{r}_i^{\mathcal{M}}, \boldsymbol{r}_j^{\mathcal{M}})$ between the unimodal graph representation $\boldsymbol{r}_i^{\mathcal{M}}$ of s_i and the graph representation $\boldsymbol{r}_j^{\mathcal{M}}$ of $s_j \in S_1$ are defined as:

$$\ell_1(\boldsymbol{r}_i^{\mathcal{M}}, \boldsymbol{r}_j^{\mathcal{M}}) = -\log \frac{e^{\text{sim}(\boldsymbol{r}_i^{\mathcal{M}}, \boldsymbol{r}_j^{\mathcal{M}})/\tau}}{\Sigma^{sup}} \tag{11}$$

$$\begin{aligned}\Sigma^{sup} = &\sum_{k, s_k \in S_1, k \neq i} e^{\text{sim}(\boldsymbol{r}_i^{\mathcal{M}}, \boldsymbol{r}_k^{\mathcal{M}})/\tau} \\ &+ \sum_{l, s_l \in S_2} e^{\text{sim}(\boldsymbol{r}_i^{\mathcal{M}}, \boldsymbol{r}_l^{\mathcal{M}})/\tau}\end{aligned} \tag{12}$$

where $\mathcal{M} \in \{t, v, a\}$, $\text{sim}(\cdot)$ is the similarity function, $\text{sim}(\boldsymbol{u}, \boldsymbol{r}) = \boldsymbol{u}^{\mathrm{T}}\boldsymbol{r}/\|\boldsymbol{u}\|\|\boldsymbol{v}\|$, and τ is the temperature parameter. We use the multimodal sentiment polarity labels as the unimodal sentiment polarity labels, and calculate the supervised contrastive loss on the unimodal graph representations and the multimodal graph representation, respectively. Finally, we sum the two losses to obtain the overall supervised contrastive loss. Following Khosla et al. (2020), the computation of intra-modal supervised contrastive loss $\mathcal{L}_{sup}^{intra}$ is:

$$\begin{aligned}\mathcal{L}_{sup}^{intra} = \sum_{\mathcal{M}} [&\sum_{s_i \in S_1} \frac{1}{|S_1 - 1|} \sum_{j, s_j \in S_1, j \neq i} \ell_1(\boldsymbol{r}_i^{\mathcal{M}}, \boldsymbol{r}_j^{\mathcal{M}}) \\ &+ \sum_{s_k \in S_2} \frac{1}{|S_2 - 1|} \sum_{l, s_l \in S_2, l \neq k} \ell_1(\boldsymbol{r}_k^{\mathcal{M}}, \boldsymbol{r}_l^{\mathcal{M}})]\end{aligned} \tag{13}$$

If more labels are correctly predicted, the value of the loss function will be lower until it converges. On the other hand, if the model cannot predict most of the labels, the loss function will fail to converge.

Self-Supervised Contrastive Loss Based on Graph Augmentations To apply the self-supervised contrastive learning, we implement augmentation data by supplementary and corrupting graphs. The augmented graphs U, V are obtained by randomly deleting or adding a certain ratio of edges, aiming at the exploration of more appropriate graph structures.

Based on the graph augmentation, for an *anchor* graph representation $\boldsymbol{r}_i^{\mathcal{M}}$, we regard the representation derived by the corresponding augmented graph $\boldsymbol{u}_i^{\mathcal{M}}$ as the *positive* sample, while others are regarded as *negative* samples. For the N examples in a mini-batch, the pairwise objective $\ell_2(\boldsymbol{r}_i^{\mathcal{M}}, \boldsymbol{u}_i^{\mathcal{M}})$ between $\boldsymbol{r}_i^{\mathcal{M}}$ and $\boldsymbol{u}_i^{\mathcal{M}}$ is defined as:

$$\ell_2(\boldsymbol{r}_i^{\mathcal{M}}, \boldsymbol{u}_i^{\mathcal{M}}) = -\log \frac{e^{\text{sim}(\boldsymbol{r}_i^{\mathcal{M}}, \boldsymbol{u}_i^{\mathcal{M}})/\tau}}{\Sigma^{self}} \tag{14}$$

$$\begin{aligned}\Sigma^{self} = \; &e^{\text{sim}(\boldsymbol{r}_i^{\mathcal{M}}, \boldsymbol{u}_i^{\mathcal{M}})/\tau} \\ &+ \sum_{j=1}^{N} \mathbb{1}_{[j \neq i]} [e^{\text{sim}(\boldsymbol{r}_i^{\mathcal{M}}, \boldsymbol{r}_j^{\mathcal{M}})/\tau} \\ &+ e^{\text{sim}(\boldsymbol{r}_i^{\mathcal{M}}, \boldsymbol{u}_j^{\mathcal{M}})/\tau}]\end{aligned} \tag{15}$$

where $\mathbb{1}_{[j \neq i]} \in \{0, 1\}$ is the indicator function and equals 1 iff $j \neq i$. The intra-modal self-supervised contrastive loss $\mathcal{L}_{self}^{intra}$ is as follow:

$$\mathcal{L}_{self}^{intra} = \frac{1}{2N} \sum_{\mathcal{M}} \sum_{i=1}^{N} \left[\ell_2(\boldsymbol{u}_i^{\mathcal{M}}, \boldsymbol{r}_i^{\mathcal{M}}) + \ell_2(\boldsymbol{r}_i^{\mathcal{M}}, \boldsymbol{u}_i^{\mathcal{M}})\right] \tag{16}$$

3.5.2 Graph Contrastive Learning at Inter-Modal Level

Corresponding to Section 3.5.1, we perform the fully-supervised contrastive loss and the self-supervised contrastive loss on the multimodal graph representations r^m, and obtain the inter-modal supervised and self-supervised contrastive loss $\mathcal{L}_{sup}^{inter}$ and $\mathcal{L}_{self}^{inter}$, which are defined as:

$$\begin{aligned}\mathcal{L}_{sup}^{inter} = &\sum_{s_i \in S_1} \frac{1}{|S_1 - 1|} \sum_{j \in S_1, j \neq i} \ell_1(\boldsymbol{r}_i^m, \boldsymbol{r}_j^m) \\ &+ \sum_{s_k \in S_2} \frac{1}{|S_2 - 1|} \sum_{l \in S_2, l \neq k} \ell_1(\boldsymbol{r}_k^m, \boldsymbol{r}_l^m)\end{aligned} \tag{17}$$

$$\mathcal{L}_{self}^{inter} = \frac{1}{2N} \sum_{i=1}^{N} [\ell_2(\boldsymbol{u}_i^m, \boldsymbol{r}_i^m) + \ell_2(\boldsymbol{r}_i^m, \boldsymbol{u}_i^m)] \tag{18}$$

3.6 Sentiment Prediction

The multimodal graph representation $\boldsymbol{r}^m$ is fed into a fully-connected layer to predict the sentiment score y:

$$y = \boldsymbol{W}^p \cdot \boldsymbol{r}^m + b^p \tag{19}$$

where $\boldsymbol{W}^p$ and b^p are the weight matrix and bias. Then, the output y_i of the prediction layer for the i-th example is compared with the ground truth y_i^* to calculate the loss of the MSA task $\mathcal{L}^{msa}$:

$$\mathcal{L}^{msa} = \frac{1}{N} \sum_{i}^{N} |y_i - y_i^*| \tag{20}$$

The overall loss of our framework is defined as:

$$\begin{aligned}\mathcal{L} =& \mathcal{L}^{msa} + w_1 * (\mathcal{L}_{sup}^{intra} + \mathcal{L}_{sup}^{inter}) \\ &+ w_2 * (\mathcal{L}_{self}^{intra} + \mathcal{L}_{self}^{inter})\end{aligned} \tag{21}$$

where w_1, w_2 are hyperparameters, controlling the effect of different losses.

4 Experiments

4.1 Datasets and Metrics

We evaluate our approach on two benchmarks, MOSI (Zadeh et al., 2016) and MOSEI (Zadeh et al., 2018c). Table 1 shows the basic statistics of the two datasets. Each example of MOSI or MOSEI contains a continuous sentiment score in the interval $[-3, 3]$ and three modal data. Image frames and audio frames are aligned to text content at the word level.

Following Zadeh et al. (2018b), binary accuracy (Acc-2) and weighted F1 score (F1-Score) are selected as classification metrics, mean absolute error (MAE) and Pearson correlation coefficient (Corr) are selected as regression metrics.

4.2 Training Setting

We take *mean absolute error* as the loss function. For contrastive learning, *N-pairs loss* (Sohn, 2016) and *SupCon loss* (Khosla et al., 2020) are naturally suitable for this scenario. For the textual encoder, we use a pre-trained BERT (bert-base-uncased) and finetune it when training. For visual and acoustic encoders, we train BiLSTMs from scratch. The optimizer chosen is Adam (Kingma and Ba, 2015) and the parameters of BERT and other model parameters are optimized separately. We use a lower learning rate {5e-6, 1e-5} and warm-up strategy for the BERT and a larger learning rate {1e-3, 1e-3} for the other parts. The hyperparameters w_1 and w_2 are selected from {0.1, 0.2, 0.3, 0.4, 0.5}. The results of our model and the reproduced models take the average results obtained from five runs with different random seeds for obtaining stable results. More training settings are presented in Table 2. In addition, we use a learning rate adjustment strategy to update the learning rate when training.

4.3 Baselines and Performance

To verify the effectiveness of our approach, we compare it with the following BERT-based methods: TFN (Zadeh et al., 2017), LMF (Liu et al., 2018), MFM (Tsai et al., 2019b), MulT (Tsai et al., 2019a), MAG-BERT (Rahman et al., 2020), MISA (Hazarika et al., 2020), Self-MM (Yu et al., 2021), HyCon-BERT (Mai et al., 2022). The details of the introduction are presented in Appendix A.

	train	valid	test	total
MOSI	1283	229	686	2198
MOSEI	16326	1871	4659	22856

Table 1: The example size of MOSI and MOSEI.

Parameter	MOSI	MOSEI
epoch	20	6
batch size	4	24
max length	50	128
hidden size	128	128
LSTM layers	1	1
GAT layers	2	1
GAT attention heads	1	1
dropout	0.2	0.1
augmentation ratio	0.2	0.2
BERT learning rate	5e-6	1e-5
other learning rate	1e-3	1e-3
$\mathcal{L}^{sup}$ weight w_1	0.1	0.1
$\mathcal{L}^{self}$ weight w_2	0.1	0.1

Table 2: Training setting details

The comparison results of our HGraph-CL framework and the baseline models are presented in Table 3. We observe that our proposed HGraph-CL consistently outperforms all the baseline models on the two datasets, which verifies the effectiveness of our approach in the MSA task. Moreover, compared with the intra-example representation learning approaches (MFM, MISA), our HGraph-CL achieves outstanding improvement on the two datasets. This indicates that exploring the sentiment implications from both intra- and inter-modal levels is significant for improving the performance of MSA. Furthermore, the significance tests of our HGraph-CL over Self-MM‡ and MAG-BERT‡ present a statistically significant improvement in Acc-2 and F1-Score on MOSI and Acc-2, F1-Score and Corr on MOSEI (with $p < 0.05$).

5 Analysis

5.1 Ablation Study

To verify the impact of the hierarchical graph contrastive learning on performance, we conduct ablation experiments on the two datasets and show the results in Table 4. We can observe that the class distribution is useful for the classification of MOSI and MOSEI datasets, and slightly improve regression. In contrast, the representation distribution learning improves regression significantly

Model	MOSI				MOSEI			
	Acc-2 ↑	F1-Score ↑	MAE ↓	Corr ↑	Acc-2 ↑	F1-Score ↑	MAE ↓	Corr ↑
TFN†	-/80.8	-/80.7	0.901	0.698	-/82.5	-/82.1	0.593	0.700
LMF†	-/82.5	-/82.4	0.917	0.695	-/82.0	-/82.1	0.623	0.677
MFM†	-/81.7	-/81.6	0.877	0.706	-/84.4	-/84.3	0.568	0.717
MulT†	81.5/84.1	80.6/83.9	0.861	0.711	-/82.5	-/82.3	0.580	0.703
MISA†	81.8/83.4	81.7/83.6	0.783	0.761	83.6/85.5	83.8/85.3	0.555	0.756
MAG-BERT	84.2/86.1	84.1/86.0	**0.712**	0.796	84.7/-	84.5/-	-	-
Self-MM	84.0/86.0	84.4/86.0	0.713	0.798	82.8/85.2	82.5/85.3	0.530	0.765
HyCon-BERT	-/85.2	-/85.1	0.713	0.790	-/85.4	-/85.6	0.601	**0.776**
MAG-BERT‡	81.5/83.1	81.5/83.1	0.808	0.761	81.4/84.6	81.9/84.6	0.552	0.756
Self-MM‡	83.1/84.9	83.1/84.9	0.736	0.791	80.5/84.2	80.0/84.2	0.531	0.764
HGraph-CL (ours)	**84.3/86.2** *	**84.6/86.2** *	0.717	**0.799**	**84.5/85.9** *	**84.5/85.8** *	**0.527**	0.769 *

Table 3: Main results on MOSI and MOSEI. ↑ denotes the higher the evaluation metric the better, and ↓ denotes the lower the evaluation metric the better. Results with † are retrieved from (Hazarika et al., 2020) , with ‡ are reproduced using the source code released by the authors, and with * indicate the significance test over Self-MM‡ and MAG-BERT‡ presents a statistically significant improvement. For Acc-2 and F1-Score, the left side of / is the result of dividing examples by *positive*/*non-positive* following (Zadeh et al., 2018b), and the right side is the result of dividing examples by *positive*/*negative* following (Tsai et al., 2019a).

Graph	CL	MOSI				MOSEI			
		Acc-2 ↑	F1-Score ↑	MAE ↓	Corr ↑	Acc-2 ↑	F1-Score ↑	MAE ↓	Corr ↑
Intra, Inter	*Sup, Self*	**84.3/86.2**	**84.6/86.2**	**0.717**	**0.799**	**84.5**/85.9	**84.5**/85.8	**0.527**	**0.769**
	Sup	84.0/85.9	84.2/85.8	0.733	0.788	84.3/**86.0**	84.3/**86.0**	0.535	0.766
	Self	83.8/85.8	84.0/85.7	0.718	0.793	84.1/85.4	84.1/85.3	0.533	0.766
Intra	*Sup, Self*	83.9/85.9	84.1/85.8	0.729	0.790	84.2/85.6	84.2/85.4	0.531	0.765
	Sup	83.6/85.8	83.8/85.7	0.731	0.788	83.9/85.9	83.9/85.6	0.539	0.767
	Self	83.6/85.5	83.8/85.4	0.726	0.789	83.9/85.3	84.0/85.1	0.533	0.764
Inter	*Sup, Self*	84.1/86.2	84.3/86.2	0.723	0.792	84.0/85.7	84.0/85.6	0.529	0.767
	Sup	83.7/86.0	83.8/85.9	0.733	0.789	84.1/85.7	84.1/85.3	0.539	0.763
	Self	83.6/85.9	83.8/85.8	0.722	0.793	83.9/85.3	83.9/85.0	0.531	0.766
None	*None*	83.0/85.1	83.1/85.0	0.756	0.784	82.8/85.1	83.1/85.0	0.539	0.763

Table 4: The performance with different contrastive learning strategies on MOSI and MOSEI. We conduct an ablation study to analyze the impact of graph structure and contrastive learning strategy. {*Intra, Inter*} denotes performing graph contrastive learning at both intra- and inter-modal levels, {*Intra*} denotes at the intra-modal level only, while {*Inter*} denotes at the inter-modal level only. {*Sup, Self*} denotes the result with class distribution learning and representation distribution learning, {*Sup*} denotes the result with class distribution learning only, {*Self*} denotes the result with representation distribution learning only. {*None*} denotes without any strategy.

on MOSI and MOSEI, and makes sense for classification. On the other hand, performing graph contrastive learning at the intra-modal level only or at the inter-modal level only can improve both classification and regression on two datasets. Additionally, our model with complete hierarchical graph contrastive learning can achieve the best overall performance, with a significant improvement over the model without it. The results suggest that hierarchical graph contrastive learning has a great positive impact on the classification and the regression of HGraph-CL.

5.2 Effect of GAT Layers

We convert the measurement of the relations between different modalities into learning the edges of a multimodal graph by GATs. Furthermore, we want to explore the effect of the number of GAT layers on performance. Thus we evaluate our model with different layers of GATs from 1 to 5 on both two datasets to quantify the effect. The experimental results are shown in Figure 3. We can observe that our model achieves the best performance with a small number of layers. Another observation is that the volatility of the classification performance is greater when choosing a big number of layers. The possible reason is that the deep GAT layer will learn similar representations of different nodes, which is called over-smoothing (Li et al., 2018). Over-smoothing may result in the modality-specific information being discarded, and the results suggest that too many layers make a negative effect.

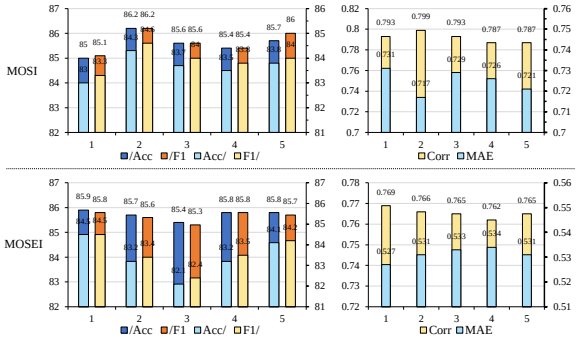

Figure 3: Results with different numbers of GAT layers

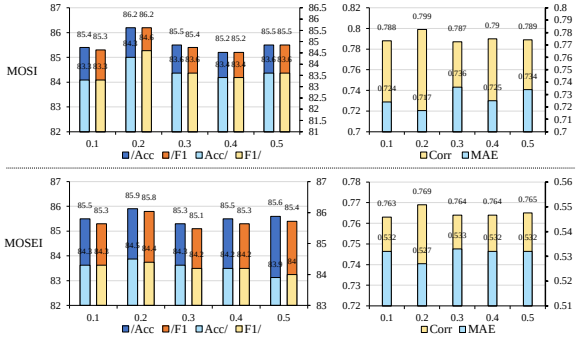

Figure 4: Results of different deleting/adding ratios

5.3 Effect of Deleting/Adding Ratio

To investigate the effect of different ratios of deleting/adding edges in deriving graph augmentations on the performance, we conduct experiments with different values of ratio and report the results in Figure 4. From the experimental results, we can observe that different ratios introduce a considerable impact on performance. When the delete ratio is small (0.1), the possibility of exploration of graph structures is limited, thus leading to a poorer performance. In addition, noticeable performance degradation is also shown when the ratio is greater than 0.2. This indicates that excessively exploring the possible relations may weaken the learning ability of graph contrastive learning. Therefore, we set the ratio to 0.2 in our experiments.

5.4 Case Study

To show the relationship strength between different representations, we select an example from the MOSI dataset and visualize the weights of edges between text and visual nodes and present them in Figure 5. We can observe that the negative word nodes *jack ass* have a stronger relationship with the visual nodes representing *frowning faces*, and are weakly related to these visual nodes representing *normal faces*. It shows that the proposed model can understand the relations of representations across modalities, enabling the highly correlated modal representations to be explicitly linked for learning the multimodal sentiment information.

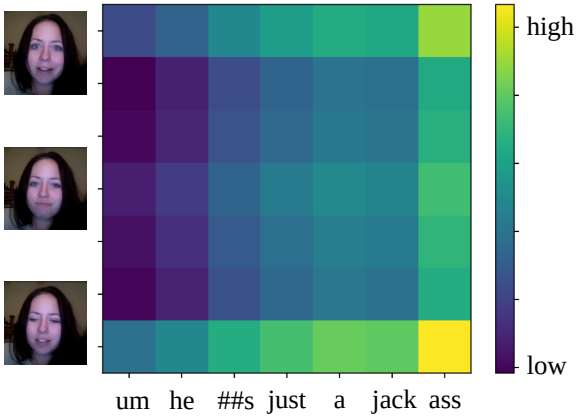

Figure 5: A case of the weights of edges between text and visual nodes

6 Conclusion

This paper proposes a novel hierarchical graph contrastive learning (HGraph-CL) framework for multimodal sentiment analysis (MSA), in which graph contrastive learning is performed at intra-modal and inter-modal levels. For the graph contrastive learning strategy performed at each level, we devise a fully-supervised contrastive loss and a self-supervised contrastive loss. The fully-supervised contrastive loss is devised to improve the learning of sentiment cues by capturing the similarity between examples in one class and the contrast among different classes. And the self-supervised contrastive loss is devised to explore a more appropriate graph structure based on the graph augmentations for making use of sentiment relations within each modality and across different modalities. Experimental results on two benchmark datasets show that our method outperforms the state-of-the-art baselines in MSA.

Acknowledgments

This work was partially supported by the National Natural Science Foundation of China (62006062, 62176076), Shenzhen Foundational Research Funding (JCYJ20200109113441941 and JCYJ2021032411 5614039), The Major Key Project of PCL2021A06, Guangdong Provincial Key Laboratory of Novel Security Intelligence Technologies (2022B1212010005).

References

Ting Chen, Simon Kornblith, Mohammad Norouzi, and Geoffrey Hinton. 2020. A simple framework for contrastive learning of visual representations. In *International conference on machine learning*, pages 1597–1607. PMLR.

Gilles Degottex, John Kane, Thomas Drugman, Tuomo Raitio, and Stefan Scherer. 2014. Covarep—a collaborative voice analysis repository for speech technologies. In *2014 ieee international conference on acoustics, speech and signal processing (icassp)*, pages 960–964, Florence. IEEE.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, Minneapolis, Minnesota. Association for Computational Linguistics.

Beliz Gunel, Jingfei Du, Alexis Conneau, and Veselin Stoyanov. 2021. Supervised contrastive learning for pre-trained language model fine-tuning. In *International Conference on Learning Representations*.

Devamanyu Hazarika, Roger Zimmermann, and Soujanya Poria. 2020. Misa: Modality-invariant and -specific representations for multimodal sentiment analysis. In *Proceedings of the 28th ACM International Conference on Multimedia*, MM '20, page 1122–1131, New York, NY, USA. Association for Computing Machinery.

Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long short-term memory. *Neural computation*, 9(8):1735–1780.

Prannay Khosla, Piotr Teterwak, Chen Wang, Aaron Sarna, Yonglong Tian, Phillip Isola, Aaron Maschinot, Ce Liu, and Dilip Krishnan. 2020. Supervised contrastive learning. In *Advances in Neural Information Processing Systems*, volume 33, pages 18661–18673. Curran Associates, Inc.

Diederik P Kingma and Jimmy Ba. 2015. Adam: A method for stochastic optimization. In *Proceedings of the 3rd International Conference on Learning Representations*, San Diego.

Thomas N Kipf and Max Welling. 2017. Semi-supervised classification with graph convolutional networks. In *Proceedings of the 28th International Conference on Computational Linguistics*.

Qimai Li, Zhichao Han, and Xiao-Ming Wu. 2018. Deeper insights into graph convolutional networks for semi-supervised learning. In *Thirty-Second AAAI conference on artificial intelligence*.

Bin Liang, Hang Su, Lin Gui, Erik Cambria, and Ruifeng Xu. 2022. Aspect-based sentiment analysis via affective knowledge enhanced graph convolutional networks. *Knowledge-Based Systems*, 235:107643.

Bin Liang, Rongdi Yin, Lin Gui, Jiachen Du, and Ruifeng Xu. 2020. Jointly learning aspect-focused and inter-aspect relations with graph convolutional networks for aspect sentiment analysis. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 150–161, Barcelona, Spain (Online). International Committee on Computational Linguistics.

Che Liu, Rui Wang, Jinghua Liu, Jian Sun, Fei Huang, and Luo Si. 2021. Dialoguecse: Dialogue-based contrastive learning of sentence embeddings. *EMNLP*.

Zhun Liu, Ying Shen, Varun Bharadhwaj Lakshminarasimhan, Paul Pu Liang, AmirAli Bagher Zadeh, and Louis-Philippe Morency. 2018. Efficient low-rank multimodal fusion with modality-specific factors. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 2247–2256, Melbourne, Australia. Association for Computational Linguistics.

Sijie Mai, Ying Zeng, Shuangjia Zheng, and Haifeng Hu. 2022. Hybrid contrastive learning of tri-modal representation for multimodal sentiment analysis. *IEEE Transactions on Affective Computing*, pages 1–1.

Wasifur Rahman, Md Kamrul Hasan, Sangwu Lee, AmirAli Bagher Zadeh, Chengfeng Mao, Louis-Philippe Morency, and Ehsan Hoque. 2020. Integrating multimodal information in large pretrained transformers. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 2359–2369, Online. Association for Computational Linguistics.

Cees GM Snoek, Marcel Worring, and Arnold WM Smeulders. 2005. Early versus late fusion in semantic video analysis. In *Proceedings of the 13th annual ACM international conference on Multimedia*, pages 399–402, New York.

Kihyuk Sohn. 2016. Improved deep metric learning with multi-class n-pair loss objective. In *Advances in neural information processing systems*, pages 1857–1865.

Yao-Hung Hubert Tsai, Shaojie Bai, Paul Pu Liang, J. Zico Kolter, Louis-Philippe Morency, and Ruslan Salakhutdinov. 2019a. Multimodal transformer for unaligned multimodal language sequences. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 6558–6569, Florence, Italy. Association for Computational Linguistics.

Yao-Hung Hubert Tsai, Paul Pu Liang, Amir Zadeh, Louis-Philippe Morency, and Ruslan Salakhutdinov.

2019b. Learning factorized multimodal representations. In *International Conference on Learning Representations*.

Petar Veličković, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Liò, and Yoshua Bengio. 2018. Graph attention networks. In *International Conference on Learning Representations*.

Yansen Wang, Ying Shen, Zhun Liu, Paul Pu Liang, Amir Zadeh, and Louis-Philippe Morency. 2019. Words can shift: Dynamically adjusting word representations using nonverbal behaviors. *Proceedings of the AAAI Conference on Artificial Intelligence*, 33(01):7216–7223.

Yang Wu, Zijie Lin, Yanyan Zhao, Bing Qin, and Li-Nan Zhu. 2021. A text-centered shared-private framework via cross-modal prediction for multimodal sentiment analysis. In *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*, pages 4730–4738, Online. Association for Computational Linguistics.

Jianing Yang, Yongxin Wang, Ruitao Yi, Yuying Zhu, Azaan Rehman, Amir Zadeh, Soujanya Poria, and Louis-Philippe Morency. 2021. MTAG: Modal-temporal attention graph for unaligned human multimodal language sequences. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 1009–1021, Online. Association for Computational Linguistics.

Yuning You, Tianlong Chen, Yongduo Sui, Ting Chen, Zhangyang Wang, and Yang Shen. 2020. Graph contrastive learning with augmentations. In *Advances in Neural Information Processing Systems*, volume 33, pages 5812–5823. Curran Associates, Inc.

Wenmeng Yu, Hua Xu, Yuan Ziqi, and Wu Jiele. 2021. Learning modality-specific representations with self-supervised multi-task learning for multimodal sentiment analysis. In *Proceedings of the AAAI Conference on Artificial Intelligence*.

Amir Zadeh, Minghai Chen, Soujanya Poria, Erik Cambria, and Louis-Philippe Morency. 2017. Tensor fusion network for multimodal sentiment analysis. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 1103–1114, Copenhagen, Denmark. Association for Computational Linguistics.

Amir Zadeh, Paul Pu Liang, Navonil Mazumder, Soujanya Poria, Erik Cambria, and Louis-Philippe Morency. 2018a. Memory fusion network for multi-view sequential learning. In *Proceedings of the Thirty-Second AAAI Conference on Artificial Intelligence*, pages 5634–5641, Palo Alto.

Amir Zadeh, Paul Pu Liang, Soujanya Poria, Prateek Vij, Erik Cambria, and Louis-Philippe Morency. 2018b. Multi-attention recurrent network for human communication comprehension. In *Proceedings of the Thirty-Second AAAI Conference on Artificial Intelligence*, pages 5642–5649, Palo Alto.

Amir Zadeh, Rowan Zellers, Eli Pincus, and Louis-Philippe Morency. 2016. Multimodal sentiment intensity analysis in videos: Facial gestures and verbal messages. *IEEE Intelligent Systems*, 31(6):82–88.

AmirAli Bagher Zadeh, Paul Pu Liang, Soujanya Poria, Erik Cambria, and Louis-Philippe Morency. 2018c. Multimodal language analysis in the wild: Cmu-mosei dataset and interpretable dynamic fusion graph. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 2236–2246, Melbourne.

Chen Zhang, Qiuchi Li, and Dawei Song. 2019. Aspect-based sentiment classification with aspect-specific graph convolutional networks. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 4568–4578, Hong Kong, China. Association for Computational Linguistics.

Qiang Zhu, Mei-Chen Yeh, Kwang-Ting Cheng, and Shai Avidan. 2006. Fast human detection using a cascade of histograms of oriented gradients. In *2006 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'06)*, volume 2, pages 1491–1498, New York. IEEE.

Yanqiao Zhu, Yichen Xu, Feng Yu, Qiang Liu, Shu Wu, and Liang Wang. 2020. Deep Graph Contrastive Representation Learning. In *ICML Workshop on Graph Representation Learning and Beyond*.

A Introduction of Baselines

Since our HGraph-CL framework is designed based on BERT, to verify the validity of our approach, we select the following state-of-the-art models with a BERT-based version for comparison:

TFN (Zadeh et al., 2017): The tensor fusion network performs outer product on three modal representations, then the multimodal representation vector fused with uni-, bi-, and tri- modalities will be obtained.

LMF (Liu et al., 2018): The low-rank multimodal fusion network is based on TFN and gets some improvement. It uses tensor decomposition to decompose the parameter tensor of the outer product layer.

MFM (Tsai et al., 2019b): The multimodal factorization model factorizes representations into two sets of independent factors, and optimizes for a joint generative-discriminative objective across multimodal data and labels.

MulT (Tsai et al., 2019a): With a slight modification in the structure of the transformer encoder, MulT proposes a cross-modal transformer network to align information from one mode to another.

MAG-BERT (Rahman et al., 2020): A variant of BERT, adding a multimodal shifting gate unit behind the input layer. By fusing with visual and acoustic information, the word embeddings can be shifted in a direction that can express sentiment polarity better in the feature space.

MISA (Hazarika et al., 2020): proposes a multitask framework for intra-example representation learning. It projects each modality to two distinct subspaces to model modality-specific and -invariant information.

Self-MM (Yu et al., 2021): designs a label generation module based on the self-supervised learning strategy to acquire independent unimodal supervision. Jointly training uni- and multimodal sentiment analysis tasks have got the state-of-the-art performance on MOSI and MOSEI.

HyCon-BERT (Mai et al., 2022): proposes hybrid contrastive learning of tri-modal representations to explore cross-modal interaction and reduce the gap among modal representations.

AMOA: Global Acoustic Feature Enhanced Modal-Order-Aware Network for Multimodal Sentiment Analysis

Ziming Li[1,2], **Yan Zhou**[1,*], **Weibo Zhang**[3], **Yaxin Liu**[1,2],
Chuanpeng Yang[1,2], **Zheng Lian**[4], **Songlin Hu**[1,2]
[1]Institute of Information Engineering, Chinese Academy of Sciences
[2]School of Cyber Security, University of Chinese Academy of Sciences
[3]China Electronics Standardization Institute
[4]Institute of Automation, Chinese Academy of Sciences
{liziming, zhouyan, liuyaxin, yangchuanpeng, husonglin}@iie.ac.cn
zhangweibo@cesi.cn
lianzheng2016@ia.ac.cn

Abstract

In recent years, multimodal sentiment analysis (MSA) has attracted more and more interest, which aims to predict the sentiment polarity expressed in a video. Existing methods typically 1) treat three modal features (textual, acoustic, visual) equally, without distinguishing the importance of different modalities; and 2) split the video into frames, leading to missing the global acoustic information. In this paper, we propose a global **A**coustic feature enhanced **M**odal-**O**rder-**A**ware network (AMOA) to address these problems. Firstly, a modal-order-aware network is designed to obtain the multimodal fusion feature. This network integrates the three modalities in a certain order, which makes the modality at the core position matter more. Then, we introduce the global acoustic feature of the whole video into our model. Since the global acoustic feature and multimodal fusion feature originally reside in their own spaces, contrastive learning is further employed to align them before concatenation. Experiments on two public datasets show that our model outperforms the state-of-the-art models. In addition, we also generalize our model to the sentiment with more complex semantics, such as sarcasm detection. Our model also achieves state-of-the-art performance on a widely used sarcasm dataset.

1 Introduction

Multimodal sentiment analysis (MSA) has attracted more and more attention in recent years. In many cases, we need to combine the textual, acoustic, and visual features to predict sentiment polarity. For example, the tone of a person's voice and the changing expression can both have an impact on sentiment polarity prediction.

In most previous works, each modality will go through the same process at the fusion, or in other words, the three modalities are treated equally (Hasan et al., 2021; Chauhan et al., 2020). However, for sentiment analysis, the textual modality is usually the core modality based on life experience and previous works (Tsai et al., 2019; Han et al., 2021a; Hasan et al., 2021), because the text contains the most basic semantic information. The acoustic feature also plays an important role: a speech with a rising tone is more likely to express positive sentiment. Finally, facial expressions also have impacts on sentiment, such as the rising range of the corners of the mouth and the size of the pupils. However, the information about sentiment in expression is not as rich as that in tone. In addition, visual information does not always correspond to text like acoustic information. In many situations, the change in the speaker's facial expression is quite subtle. Even more, the speaker is absent in some videos. These visual noises may bring confusion to the model. Therefore, we consider the order of modalities, i.e. textual-acoustic-visual (t-a-v) while integrating them.

For MSA, a video is usually divided into many frames, and each frame corresponds to a very short time period in the video. The local acoustic features extracted from every single frame interact with each other in the fusion process. However, this method loses the global acoustic information and cannot fully reflect the tone feature of the whole audio.

To address these challenges, we propose a global acoustic feature enhanced modal-order-aware network. Firstly, the **M**odal-**O**rder-**A**ware network (MOA) is designed to integrate the three modalities in a certain order, where there are two stages con-

* Corresponding author.

Proceedings of the 29th International Conference on Computational Linguistics, pages 7136–7146
October 12–17, 2022.

necting the core and the outer modalities. We put textual modality at the core, and then the acoustic modality is integrated in stage 1, and finally, the visual modality is integrated in stage 2. At each stage, we design **C**ross-**M**odal **T**ransformer (CMT) based on the Transformer encoder (Vaswani et al., 2017) to integrate new modal features. Through CMT, the modality added before can also provide information for the later processes. Consequently, the textual feature learning is continuously enhanced through two stages, while the noise impact brought by the visual modality added in stage 2 is reduced. Then, to preserve the global acoustic information, we use *openSMILE* (Eyben et al., 2010) to extract the **G**lobal **A**coustic **F**eature (GAF) of the video to enhance modal feature learning. Furthermore, GAF and the multimodal fusion feature originally reside in their own spaces, which brings challenges to the fusion or concatenation. Inspired by MOCO (He et al., 2020), we employ contrastive learning to align the two features before concatenating them. Because visual modality may bring more noise and the processing of the entire video needs more computational power, we don't employ global visual features in our model. The main contributions of our paper are as follows:

- We propose AMOA - a novel multimodal sentiment analysis model that can integrate the three modalities in a certain order. In the modal-order-aware network, CMT is designed to fuse the features of different modalities.
- We are the first to introduce the global acoustic feature into MSA, which aims to preserve the global acoustic information and enhance the learning of the overall video feature. Furthermore, contrastive learning is utilized to align them before concatenation.
- We conduct experiments on sentiment (CMU-MOSI and CMU-MOSEI) and sarcasm (MUStARD) datasets, and the results show the state-of-the-art performance of our model.

2 Related Work

Multimodal fusion has always been the most critical step in MSA. Early works directly concatenate unimodal features or use outer product (Zadeh et al., 2018). With the development of the neural networks (Russakovsky et al., 2015; Hochreiter and Schmidhuber, 1997) and attention mechanism (Bahdanau et al., 2015; Vaswani et al., 2017), more and more complex networks have been applied to MSA to integrate modalities.

(Tang et al., 2021) uses a translation-based model to supplement the missing modalities. (Liu et al., 2021) is based on quantum probability modeling and uses multi-task learning to predict sentiment polarity and detect sarcasm at the same time. (Rahman et al., 2020) proposes an attachment to pre-trained language models so that they can adapt to the task of multimodal sentiment analysis. (Han et al., 2021a) performs fusion (relevance increment) and separation (difference increment) on pairwise modality representations. (Han et al., 2021b) proposes a novel framework to maximize the mutual information in unimodal input pairs and between the multimodal fusion result and unimodal input. (Colombo et al., 2021) proposes new objectives to measure the dependency between modalities. These models treat all three modalities equally and mostly design very complex modules to achieve better results, while our model integrates three modalities in a certain order to distinguish their contributions, simple but effective.

Contrastive learning (CL) is a widespread self-supervised learning method in recent years. MOCO (He et al., 2020) and SimCLR (Chen et al., 2020) have achieved good results with CL in computer vision. After that, CL is applied to text-image multimodal tasks, such as image-text retrieval and visual question answering (Li et al., 2021).

3 Model

In this section, we will describe in detail how our proposed model works. The overall architecture of our model (AMOA) is shown in Figure 1. Our model consists of three modules: the modal-order-aware network (MOA), global acoustic feature (GAF) extraction & contrastive learning module, and classification module. In MOA, the unimodal features are first encoded and then integrated in a certain order, thus generating the multimodal fusion feature (Section 3.1). However, the multimodal fusion feature obtained in MOA is composed of single frame features and insufficient to reflect the overall change of tone, which is important for expressing sentiment. Therefore, we further extract GAF to complement the complete acoustic features. To align the multimodal fusion feature and GAF, we introduce contrastive learning and add the contrastive loss and classification loss together to guide

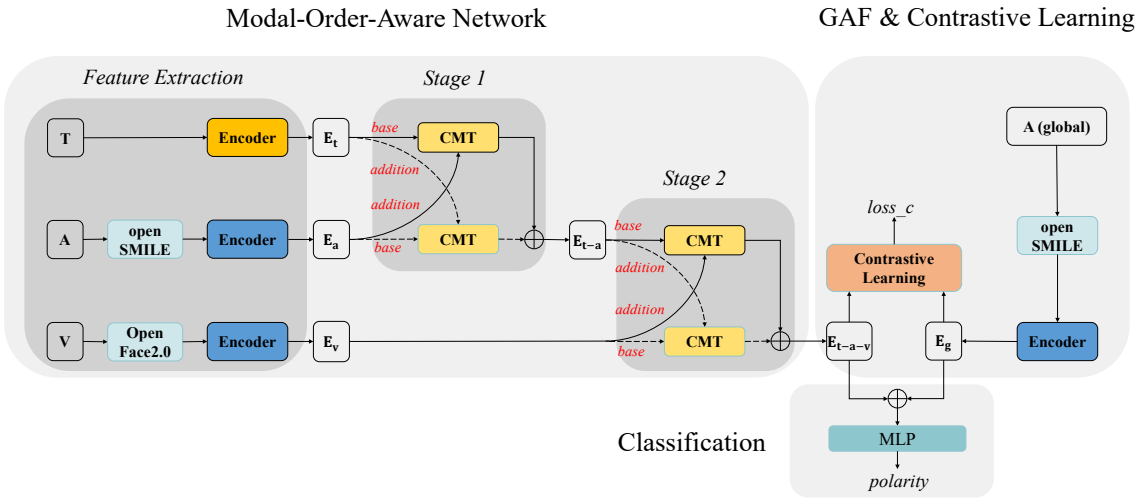

Figure 1: The overall architecture of our model. T, A, and V represent textual modality, acoustic modality (frame level), and visual modality (frame level) respectively. A (global) indicates acoustic modality which is not segmented. $\oplus$ represents the concatenation operation. The dashed parts make up two-way CMT.

model training. (Section 3.2) Finally, a multilayer perceptron layer is utilized for classification (Section 3.3).

3.1 Modal-Order-Aware Network

3.1.1 Unimodal Feature Extraction and Encoder

Textual: In this paper, pre-trained BERT-base-uncased (Devlin et al., 2019) is used as our text encoder, which has 12 layers and the hidden size is 768. The text encoder takes text $\mathbf{X}_t = \{x_1, x_2, \ldots, x_{n_t}\}$ in the video as input, and then output the last hidden layer representation: $\mathbf{E}_t \in \mathbb{R}^{n_t \times d_t}$, where n_t is the number of tokens and d_t is the hidden size (768) of BERT-base-uncased.

Acoustic: We use *openSMILE* to extract frame-level features of audio with 10 ms frame shift and 25 ms frame size. *openSMILE* provides a series of default feature sets, such as the INTERSPEECH 2010 Paralinguistic Challenge Feature Set (IS10) (Schuller et al., 2010), which contains different low-level features and their corresponding high-level features. For each frame, we extract IS10 as the feature vector: $\mathbf{X}_a = \{x_1, x_2, \ldots, x_{n_a}\}$, $\mathbf{X}_a \in \mathbb{R}^{n_a \times d_a}$ and n_a is the number of frames and d_a is the dimension of the acoustic feature. Then, we use P2FA (Yuan et al., 2008) to align the acoustic features to each word. Specifically, we obtain $\mathbf{X}_a^{'} = \{\mathbf{x}_1^{'}, \mathbf{x}_2^{'}, \ldots, \mathbf{x}_{n_t}^{'}\}$ by extracting the timing of all the words and averaging the acoustic feature vectors during this time. Because Transformer has the advantage of capturing long-distance dependencies, we directly use the Transformer encoder with random initialization to encode the feature and finally get the acoustic representation $\mathbf{E}_a \in \mathbb{R}^{n_t \times d_a}$.

Visual: The visual information in the video mainly comes from expressions, head shaking, and so on. We use OpenFace 2 (Baltrusaitis et al., 2018) to extract facial features at the frame level. These features are based on the Facial Action Coding System (Ekman and Rosenberg, 1997). Like acoustic modality, we then use the Transformer encoder to obtain visual representation $\mathbf{E}_v \in \mathbb{R}^{n_t \times d_v}$, where d_v is the hidden size of visual feature. The encoders of acoustic and visual modalities are independent of each other and they don't share any parameters.

3.1.2 Modal Order

Currently, most works treat the three modalities equally. They either feed the three modalities into a module for fusion and interaction at the same time, or integrate them in pairs. In this way not only the text information can not play a full role, but also the noise in the visual modality has the same impact. Therefore, we integrate three modalities in a certain order. First, the textual feature is extracted and encoded, i.e. $\mathbf{E}_t$; then, in stage 1, the acoustic features are fused to generate $\mathbf{E}_{t\text{-}a}$; finally, in stage 2, the visual features are integrated to generate $\mathbf{E}_{t\text{-}a\text{-}v}$. The final experiment will prove the optimality of the t-a-v order. At each stage, we employ CMT to integrate the previous and latter modalities, which will be described in detail in Section 3.1.3.

3.1.3 Cross-Modal Transformer

Based on the Transformer encoder, CMT is utilized to integrate new modalities, as shown in Figure 2. In stage 1, CMT inputs the textual and acoustic modal features and outputs the textual-acoustic fusion feature. Then in stage 2, the textual-acoustic feature and visual feature are fed to CMT and we obtain the textual-acoustic-visual fusion feature. Transformer is mainly composed of attention mechanism, and so is CMT. The input of the attention mechanism is a K-V pair and a query Q. The K-V pair can be regarded as basic information, while Q is additional information. We get the interactive information between Q and V by calculating the score between Q and K. Multi-head attention is composed of several parallel attention modules. They are individually responsible for calculating part of the results which will be concatenated into the final result. Different from the Transformer encoder, CMT utilizes multi-head attention instead of multi-head self-attention to integrate two features. Besides, CMT is also a multi-layer module. The original Transformer encoder cannot input two modalities and stack multiple layers together. Each layer of CMT has two inputs called *base* and *addition*. *base* of each layer stays the same, while *addition* is constantly updated.

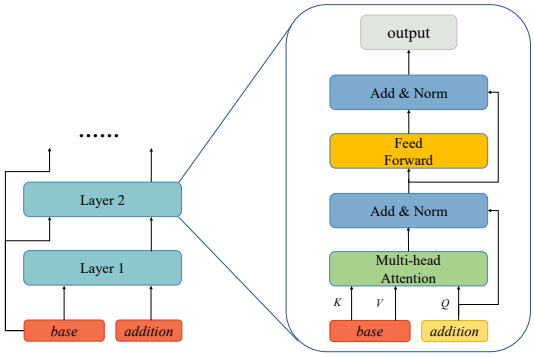

Figure 2: Cross-Modal Transformer.

Take the CMT in stage 1 as an example, in the first layer, *base* is the textual feature $\mathbf{E}_t$ and *addition* is the acoustic feature $\mathbf{E}_a$. Then we input *base* (K, V) and *addition* (Q) into the multi-head attention (MHA) followed by a residual connection and layer normalization:

$$\mathbf{Z} = MHA(addition,\ base,\ base), \quad (1)$$

$$\mathbf{Z}^{'} = Norm(\mathbf{Z} + addition). \quad (2)$$

Then we employ a feedforward neural network followed by a residual connection and layer normalization:

$$\mathbf{Z}^{''} = FeedForward(\mathbf{Z}^{'}), \quad (3)$$

$$\mathbf{E}_{t\text{-}a}^{(1)} = Norm(\mathbf{Z}^{'} + Dropout(\mathbf{Z}^{''})). \quad (4)$$

where $Norm$ is layer normalization. The above is the calculation process of the first layer, which can be expressed as:

$$\mathbf{E}_{t\text{-}a}^{(1)} = \mathrm{CMT}^{(1)}(\mathbf{E}_t,\ \mathbf{E}_a). \quad (5)$$

In the second and later layers, *base* is always the textual feature, and *addition* is the output of the previous layer:

$$\mathbf{E}_{t\text{-}a}^{(i)} = \mathrm{CMT}^{(i)}(\mathbf{E}_t,\ \mathbf{E}_{t\text{-}a}^{(i-1)}), i = 2, 3, \ldots, N, \quad (6)$$

where $\mathbf{E}_{t\text{-}a}^{(i)}$ is the output of the ith layer of CMT and we take the output of the last layer as the textual-acoustic fusion feature $\mathbf{E}_{t\text{-}a}$.

Similar to the above procedure, we take $\mathbf{E}_{t\text{-}a}$ as *base* and the visual feature $\mathbf{E}_v$ as *addition* in CMT (*addition* of each layer is also updated continuously) in stage 2. Then we obtain the textual-acoustic-visual fusion feature:

$$\mathbf{E}_{t\text{-}a\text{-}v} = \mathrm{CMT}(\mathbf{E}_{t\text{-}a},\ \mathbf{E}_v). \quad (7)$$

In order to distinguish the importance of previous and latter modalities, we only use one-way CMT, that is, one input provides more information as *base* and another input provides less information as *addition*. For comparison, we also design a bidirectional module, as shown by the dashed line in Figure 1. The results from two CMTs are concatenated and transmitted to the next step. The model with one-way CMT is called S-AMOA and the model with two-way CMT is called B-AMOA. In this paper, we use S-AMOA by default.

Finally, a dropout layer and max-pooling layer are utilized to extract the most salient features across the time dimension:

$$\mathbf{E}_f = Maxpooling(Dropout(\mathbf{E}_{t\text{-}a\text{-}v})), \quad (8)$$

where $\mathbf{E}_f \in \mathbb{R}^{d_h}$ is the multimodal fusion feature.

3.2 GAF & Contrastive Learning

We use IS10 in *openSMILE* to extract GAF. Instead of splitting the video into frames, we extract the global acoustic feature of the whole video and get a one-dimensional feature vector: $\mathbf{X}_g \in \mathbb{R}^{d_g}$. We unsqueeze $\mathbf{X}_g$ to make it fit the input shape of

Transformer. Finally, we utilize the Transformer encoder to obtain the GAF representation: $\mathbf{E}_g \in \mathbb{R}^{d_g}$.

During training, apart from the final classification loss, we introduce another loss through contrastive learning, called $loss\_c$, to align the multimodal fusion feature $\mathbf{E}_f$ and global acoustic feature $\mathbf{E}_g$. One of the important steps of contrastive learning is to construct positive and negative samples. A direct idea is to take $\mathbf{E}_f$ and $\mathbf{E}_g$ belonging to the same sample in a batch as a positive pair and those not belonging to the same sample as negative pairs. Previous studies show that more negative samples promote contrastive learning. However, a larger batch size requires higher computing power so infinitely increasing the batch size is unpractical.

To applicably increase the number of negative samples, we construct a queue storing ($\mathbf{E}_f$, $\mathbf{E}_g$) pairs in the model. All the $\mathbf{E}_f$ in the queue are combined into the matrix $\mathbf{E}_f^q \in \mathbb{R}^{K \times d_h}$, and all the $\mathbf{E}_g$ in the queue are combined into the matrix $\mathbf{E}_g^q \in \mathbb{R}^{K \times d_h}$. K is the upper limit of the queue size. The data in the queue comes from the previous batches and acts as negative samples.

When a new batch comes, we get $\mathbf{E}_f^b \in \mathbb{R}^{B \times d_h}$ and $\mathbf{E}_g^b \in \mathbb{R}^{B \times d_g}$ which are positive samples of each other. B should be the batch size, but in practice, the utterance is processed with the context, so B is actually the product of the original batch size and the number of sentences in a sample. Next, we calculate the cosine similarity between each same row (i.e. the same sample) of the two matrices in this batch, which should be maximized:

$$\mathbf{S}_{pos} = Cosine(\mathbf{E}_f^b, \mathbf{E}_g^b), \tag{9}$$

where $Cosine$ is the cosine similarity function. Each value in $\mathbf{S}_{pos} \in \mathbb{R}^{B \times 1}$ is the similarity of the corresponding samples in this batch. Because there are multimodal fusion features and global acoustic features in both batch and queue, we construct double negative samples by calculating the similarity of $\mathbf{E}_f^b$ and $\mathbf{E}_g^q$ and the similarity of $\mathbf{E}_g^b$ and $\mathbf{E}_f^q$ respectively, which should be minimized:

$$\mathbf{S}_{neg}^{f \to g} = Cosine(\mathbf{E}_f^b, \mathbf{E}_g^q), \tag{10}$$

$$\mathbf{S}_{neg}^{g \to f} = Cosine(\mathbf{E}_g^b, \mathbf{E}_f^q), \tag{11}$$

where the value in the ith row and the jth column of $\mathbf{S}_{neg}^{f \to g} \in \mathbb{R}^{B \times K}$ is the cosine similarity between $\mathbf{E}_f$ of the ith sample in the current batch and $\mathbf{E}_g$ of the jth sample in the queue. Then we concatenate the three similarity matrices:

$$\mathbf{S} = Concat(\mathbf{S}_{pos}, \mathbf{S}_{neg}^{f \to g}, \mathbf{S}_{neg}^{g \to f}), \tag{12}$$

where the first column of $\mathbf{S} \in \mathbb{R}^{B \times (1+2 \times K)}$ is the similarity between positive samples, and the others are the similarity between positive samples and negative samples. Then we define a loss function to maximize the value of the first column of $\mathbf{S}$ and minimize the value of the other columns:

$$loss\_c = \frac{\sum_{i=1}^{B} | \, log(Softmax(\mathbf{S}_i)[0]) \, |}{B}. \tag{13}$$

As part of the final loss, $loss\_c$ will help to align $\mathbf{E}_f$ and $\mathbf{E}_g$.

Finally, we add $\mathbf{E}_f^b$ and $\mathbf{E}_g^b$ in the current batch to the queue, and if the queue size exceeds K, we pop up the pairs from the head of the queue.

3.3 Classification

We get the multimodal fusion feature $\mathbf{E}_f$ through MOA, and also extract the global acoustic feature $\mathbf{E}_g$. Now we concatenate the two features and then get:

$$\mathbf{R} = Concat(\mathbf{E}_f, \mathbf{E}_g), \tag{14}$$

where $\mathbf{R} \in \mathbb{R}^{(d_h+d_g) \times 1}$ is the final multimodal representation. Finally, We input the $\mathbf{R}$ into a multilayer perceptron (MLP) layer for classification:

$$\hat{y} = W_2(ReLU(W_1\mathbf{R})) + b_2, \tag{15}$$

where W_1, W_2, b_1 and b_2 are the parameters and $ReLU$ is the nonlinear activation function. During training, we use the MSE loss function to calculate the classification loss $loss\_f$ and then add contrastive learning loss with a certain weight:

$$loss = (1 - \alpha) \cdot loss\_f + \alpha \cdot loss\_c, \tag{16}$$

where α is a hyper-parameter, which is set to balance the two losses.

Dataset	Train	Valid	Test	All
CMU-MOSI	1284	229	686	2199
CMU-MOSEI	16326	1871	4659	22856
MUStARD	552	69	69	690

Table 1: Split of three datasets.

Models	CMU-MOSI				CMU-MOSEI			
	Acc-2 ↑	F1 ↑	MAE ↓	Corr ↑	Acc-2 ↑	F1 ↑	MAE ↓	Corr ↑
TFN*	-/0.8080	-/0.8070	0.901	0.698	-/0.8250	-/0.8210	0.593	0.700
LMF*	-/0.8250	-/0.8240	0.917	0.695	-/0.8200	-/0.8210	0.623	0.677
MFM*	-/0.8170	-/0.8160	0.877	0.706	-/0.8440	-/0.8430	0.568	0.717
ICCN*	-/0.8300	-/0.8300	0.862	0.714	-/0.8420	-/0.8420	0.565	0.713
MulT*	0.8150/0.8410	0.8060/0.8390	0.861	0.711	-/0.8250	-/0.8230	0.580	0.703
MISA*	0.8079/0.8210	0.8077/0.8203	0.804	0.764	0.8259/0.8423	0.8267/0.8397	0.568	0.724
MAG-BERT*	0.8250/0.8430	0.8260/0.8430	0.731	0.789	0.8380/0.8520	0.8370/0.8510	0.539	0.753
self-MM*	0.8400/**0.8598**	0.8442/**0.8595**	**0.713**	**0.798**	0.8281/0.8517	0.8253/0.8530	0.530	0.765
MMIM‡	0.8324/0.8521	0.8311/0.8515	0.722	0.786	0.8418/0.8558	0.8425/0.8535	0.538	0.763
BBFN‡	0.8134/0.8353	0.8124/0.8351	0.833	0.743	0.8298/0.8569	0.8327/0.8570	0.579	0.759
S-AMOA (ours)	**0.8411**/0.8415	**0.8452**/0.8421	0.720	0.788	**0.8560/0.8645**	**0.8601/0.8654**	**0.526**	**0.772**
B-AMOA (ours)	0.8163/0.8277	0.8173/0.8283	0.735	0.786	0.8501/0.8575	0.8508/0.8587	0.578	0.766

Table 2: Performances of multimodal models on the CMU-MOSI and CMU-MOSEI datasets. * indicates that the results are from (Han et al., 2021b). ‡ indicates that the results are reproduced from open-source code with hyper-parameters provided in original papers. For Acc-2 and F1, we have two methods of calculation: non-negative/negative (left) and positive/negative (right). The best results are marked in bold. ↑ indicates that the higher the value, the better the result; ↓ indicates that the lower the value, the better the result. Bolded numbers represent the best results.

Models	Acc-2	F1
MFN◇	0.7391	0.7386
MulT◇	0.7536	0.7541
MAG(BERT)◇	0.7826	0.7818
MAG(XLNet)◇	0.7681	0.7679
A-MTL†	-	0.7657
QPM†	-	0.7753
HKT†	0.7941	0.7925
S-AMOA (ours)	**0.8406**	**0.8412**
B-AMOA (ours)	0.8116	0.8116

Table 3: Performances of multimodal models on the MUStARD dataset. ◇ are the results on the dataset using the original code provided in the paper. † indicates that the results are from the original paper. - indicates that the original paper provides neither the results under the Acc metric nor the training code.

4 Experiments

4.1 Datasets and Metrics

In order to verify the performance of our model in sentiment polarity prediction, we conduct experiments on two widely used public datasets: CMU-MOSI (Zadeh et al., 2016) and CMU-MOSEI (Zadeh and Pu, 2018). CMU-MOSI has 2199 video clips, each of which is a speaker sharing their opinions on something. Each clip is labeled with the polarity of sentiment, and the range of labels is: [-3, 3]. CMU-MOSEI has 23454 film review clips, which are labeled in the same way as CMU-MOSI. Our model is not only applicable to the prediction of general sentiment polarity, but also can detect more complex sentiments, such as sarcasm. To verify this, we conduct experiments on the MUStARD dataset (Castro et al., 2019), the unique multimodal sarcasm dataset containing three modalities. The dataset is collected from four TV shows, with a total of 690 samples.

We use four commonly used evaluation metrics to evaluate the performance of the model on the MOSI and MOSEI datasets: binary classification accuracy (Acc-2), which divides seven labels into two categories (positive/negative and non-negative/negative); binary classification F1; mean absolute error (MAE), which is the difference between the predicted value and the real value; Pearson correlation (Corr), which measures the degree of prediction skew. The label of each sample is sarcasm or non-sarcasm, so we only use Acc-2 and F1 to evaluate the performance on the MUStARD dataset.

The split specifications of the three datasets are provided in Table 1. To motivate future research, the code will be released soon.

4.2 Baselines

For sentiment polarity prediction, we compare our model with many baseline models.

TFN (Zadeh et al., 2017): It integrates three modal features by outer product, which is a very classic work.

LMF (Liu et al., 2018): It performs multimodal fusion using low-rank tensors to improve

efficiency.

MFM (Tsai et al., 2018): It decomposes features into modal fusion features and modal specific features to enhance model robustness.

ICCN (Sun et al., 2020): It obtains multimodal embedding by calculating the outer product of the text and the other two modalities.

MulT (Tsai et al., 2019): It uses cross-modal transformers to fully integrate three modalities for aligned sequences or unaligned sequences.

MISA (Hazarika et al., 2020): It projects the modalities into two different subspaces to learn the intra modal features and inter modal features, respectively.

MAG (Rahman et al., 2020): It adds a multimodal adaptation gate to the existing pre-trained language models (BERT and XLNet) so that they can receive acoustic and visual information during fine-tuning. Because our model uses BERT for word embedding, MAG-BERT is employed as a baseline.

self-MM (Yu et al., 2021): It generates uni-modal labels based on self-supervised learning, and then jointly trains uni-modal and multi-modal tasks.

MMIM (Han et al., 2021b): It maximizes the mutual information in a multimodal fusion pipeline to maintain task-related information.

BBFN (Han et al., 2021a): It focuses on bimodal fusion process and balances the contribution of different modality pairs properly.

Furthermore, we also select some sarcasm detection baselines for comparison on the MUStARD dataset.

MFN (Zadeh et al., 2018): It obtains the intra-modal information and inter-modal information based on LSTM and passes the multimodal fusion information through time.

A-MTL (Chauhan et al., 2020): It manually annotates the samples in the MUStARD dataset with sentiment and emotion as well as analyzes sarcasm, sentiment, and emotion together through multi-task learning.

QPM (Liu et al., 2021): It builds a quantum probability-driven multi-task learning framework, including a quantum-like fusion network and quantum incompatibility measurements.

HKT (Hasan et al., 2021): Besides the three modalities, it introduces the ambiguity of words and sentiment dictionary and constructs a bimodal cross-attention layer based on Transformer.

4.3 Main Results

The experimental results on the MOSI and MOSEI datasets are shown in Table 2. On the MOSEI dataset, our model outperforms all baseline models in every metric. In the binary classification task, our model attains an improvement of **1**% - **2**% over other models, which indicates the advantage of our model in sentiment polarity prediction. On the MOSI dataset, our model outperforms all baseline models in Acc-2 and F1 (non-negative/negative). In other metrics, our model also achieves results close to SOTA. It is worth noting that the advantage of our model in Acc-2 (non-negative/negative) is more obvious than that in Acc-2 (negative/positive). This is because our model tends to classify the samples labeled neutral into the positive category, which is consistent with life experience.

The experimental results on the MUStARD dataset are shown in Table 3. The results illustrate that our model achieves the best performance and outperforms all baseline models (+**4.65**%). Some baseline models use context information (A-MTL, QPM, HKT), but the results are worse than our model without context information.

These results demonstrate the superiority of our proposed model and indicate the effectiveness of the modal-order-aware network and GAF compared with all baseline models.

4.4 Analysis

In order to further analyze the performance of our model and verify the contribution of each module, we conduct extensive experiments on the MOSI and MOSEI datasets.

4.4.1 CMT

We design two kinds of CMT, one-way and two-way, and their experimental results are shown in Table 2 and Table 3. The results show that the two-way CMT enhances the noise influence of the latter modality, and makes the previous modality unable to play a full role, which has an adverse impact on the performance of the model.

The number of layers N of CMT in the model is also a hyper-parameter. We set different N and conduct experiments on MOSI and MOSEI datasets. The results in Figure 3 show that when N is 5, the model achieves the best performance on the MOSI dataset, and when N is 2, the model achieves the best performance on the MOSEI dataset. As the number of layers increases, the information captured by CMT also increases. However, CMT may

Models	CMU-MOSI				CMU-MOSEI			
	Acc-2	F1	MAE	Corr	Acc-2	F1	MAE	Corr
AMOA	**0.8411/0.8415**	**0.8422/0.8421**	**0.720**	**0.788**	**0.8560/0.8645**	**0.8601/0.8654**	**0.526**	**0.772**
t-v-a	0.8265/0.8307	0.8287/0.8335	0.737	0.786	0.8521/0.8614	0.8555/0.8639	0.589	0.771
a-t-v	0.8279/0.8338	0.8291/0.8327	0.739	0.780	0.8542/0.8608	0.8576/0.8630	0.582	0.770
a-v-t	0.8236/0.8323	0.8255/0.8314	0.748	0.781	0.8499/0.8611	0.8542/0.8637	0.584	0.771
v-t-a	0.8250/0.8262	0.8255/0.8303	0.741	0.775	0.8499/0.8622	0.8530/0.8640	0.575	0.769
v-a-t	0.8309/0.8262	0.8314/0.8303	0.744	0.774	0.8527/0.8617	0.8565/0.8633	0.581	0.771
−v	0.8250/0.7988	0.8252/0.7978	0.733	0.770	0.8492/0.8564	0.8506/0.8573	0.578	0.759
−a	0.8090/0.8231	0.8091/0.8240	0.756	0.769	0.8475/0.8581	0.8472/0.8590	0.579	0.757
−t	0.7116/0.7215	0.7168/0.7015	0.899	0.335	0.7391/0.7095	0.7639/0.7252	0.795	0.392
−GAF	0.7804/0.8192	0.7791/0.8224	0.760	0.761	0.8486/0.8564	0.8498/0.8567	0.573	0.761
−CL	0.8265/0.8033	0.8290/0.8099	0.739	0.769	0.8518/0.8603	0.8537/0.8612	0.587	0.761
−CL (f→g)	0.8309/0.8368	0.8314/0.8387	0.731	0.771	0.8544/0.8608	0.8580/0.8623	0.583	0.769
−CL (g→f)	0.8294/0.8246	0.8291/0.8259	0.734	0.772	0.8555/0.8611	0.8595/0.8623	0.581	0.771
−q	0.8236/0.8105	0.8251/0.8112	0.733	0.771	0.8520/0.8625	0.8541/0.8645	0.600	0.762

Table 4: Order study and ablation study. $-m$ means to remove the m mode, where $m \in$ {t, a, v} is the three modalities. −GAF means not using GAF to enhance feature learning. −CL means to directly concatenate the multimodal fusion feature and GAF without contrastive learning for alignment. −CL (f→g) means to remove half of the negative samples calculated by Eq.(10) and −CL (g→f) means to remove half of the negative samples calculated by Eq.(11). −q means to construct positive and negative samples only from the same batch without using the queue.

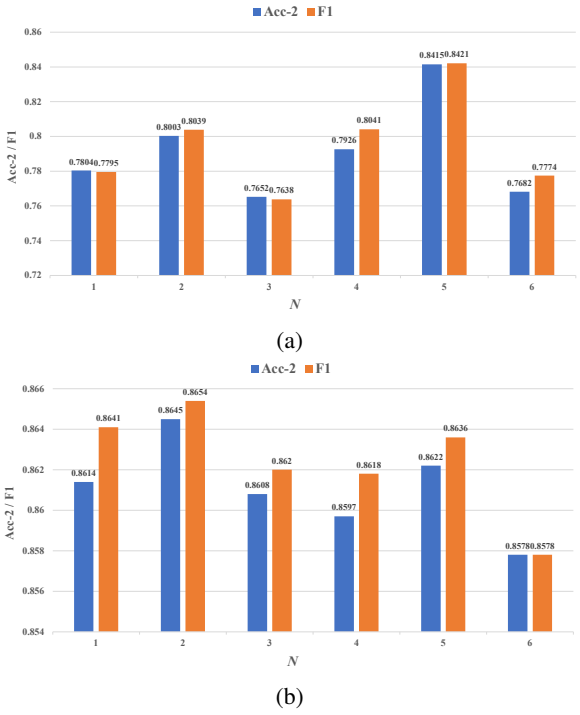

Figure 3: Experimental results with different layers of CMT. (a) is on the CMU-MOSI dataset. (b) is on the CMU-MOSEI dataset.

suffer from the distribution shift in the up layers, which will make the similarity less reliable. So bigger isn't necessarily better for N. Please refer to the experiment results in the supplementary materials.

4.4.2 Order of Modalities

In our modal-order-aware network, we put text at the core and then integrate acoustic modality in stage 1, and visual modality is integrated in stage 2 (i.e., t-a-v). We try all permutations and obtain the prediction results under the same settings, as shown in Table 4. The order of t-a-v performs best, which verifies our hypothesis of modal order.

4.4.3 Role of Unimodalities

In order to verify the role of every single modality, we separately remove one modality and integrate the other two modalities in the original order. For example, the fusion order t-v removes the acoustic modality (−a). The experimental results are shown in Table 4. When a modality is removed, the performance of the model decreases in varying degrees, which shows that each modality plays an important role. Specifically, when the textual modality is removed, the performance decreases most obviously. In addition, The impact of acoustic modality is slightly greater than that of visual modality.

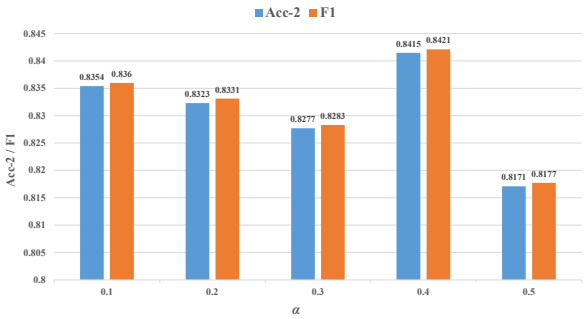

(a)

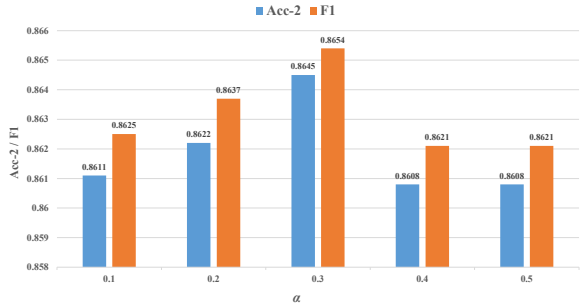

(b)

Figure 4: Experimental results with different α. (a) is on the CMU-MOSI dataset. (b) is on the CMU-MOSEI dataset.

4.4.4 Role of GAF

To verify the role of the global acoustic feature, we remove GAF and the contrastive learning module from our model. The results are shown in Table 4 ($-$GAF). Without GAF, model performance drops to some extent, which indicates that GAF plays an important role in our model. In addition, GAF is more effective in MOSI dataset. This is related to the different distribution of data in the two datasets. In many cases, GAF contributes more to the analysis of the examples which are relatively short but have large sound fluctuations. The proportion of this kind of example in MOSI is greater than that in MOSEI.

4.4.5 Role of Contrastive Learning

Furthermore, we remove the contrastive learning module from the original model and concatenate the multimodal fusion feature and GAF directly to verify the role of contrastive learning. The results in Table 4 ($-$CL) show that when the contrastive learning module is removed, we can see a clear drop in all metrics. In contrastive learning, we construct double negative samples based on different $\mathbf{E}_f$ and $\mathbf{E}_g$. When only one group negative samples are used, the performance of the model decreases to varying degrees.

In our experiments, we set the hyper-parameter α as the weight of the loss of the contrastive learning module in the whole loss. The influence of the value of α on the experimental results is shown in Figure 4. When α is 0.4, the model achieves the best performance on the MOSI dataset. When α is 0.3, the model achieves the best performance on the MOSEI dataset.

A queue is used to construct more negative samples. We try to remove this queue and construct negative samples only in the same batch. As shown in Table 4 ($-$q), the queue plays an important role in the contrastive learning module of our model.

5 Conclusion

For multimodal sentiment analysis, we propose the modal-order-aware network to integrate the three modalities in a certain order to distinguish the importance of different modalities. Besides, we are the first to introduce the global acoustic feature into this task to capture the changes in the tone of the whole video. Considering the misalignment between the multimodal fusion feature and GAF, contrastive learning is utilized to align them before concatenation. Experiments on three widely used datasets show that our model achieves the best performance. Besides, we also verify the effectiveness of each module of our model.

References

Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. 2015. Neural machine translation by jointly learning to align and translate. In *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*.

Tadas Baltrusaitis, Amir Zadeh, Yao Chong Lim, and Louis-Philippe Morency. 2018. Openface 2.0: Facial behavior analysis toolkit. In *2018 13th IEEE international conference on automatic face & gesture recognition (FG 2018)*, pages 59–66. IEEE.

Santiago Castro, Devamanyu Hazarika, Verónica Pérez-Rosas, Roger Zimmermann, Rada Mihalcea, and Soujanya Poria. 2019. Towards multimodal sarcasm detection (an _obviously_ perfect paper). In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 4619–4629.

Dushyant Singh Chauhan, SR Dhanush, Asif Ekbal, and Pushpak Bhattacharyya. 2020. Sentiment and emotion help sarcasm? a multi-task learning framework for multi-modal sarcasm, sentiment and emotion analysis. In *Proceedings of the 58th Annual Meeting of*

the Association for Computational Linguistics, pages 4351–4360.

Ting Chen, Simon Kornblith, Mohammad Norouzi, and Geoffrey Hinton. 2020. A simple framework for contrastive learning of visual representations. In *International conference on machine learning*, pages 1597–1607. PMLR.

Pierre Colombo, Emile Chapuis, Matthieu Labeau, and Chloé Clavel. 2021. Improving multimodal fusion via mutual dependency maximisation. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 231–245.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. Bert: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186.

Paul Ekman and Erika L Rosenberg. 1997. *What the face reveals: Basic and applied studies of spontaneous expression using the Facial Action Coding System (FACS)*. Oxford University Press, USA.

Florian Eyben, Martin Wöllmer, and Björn Schuller. 2010. Opensmile: the munich versatile and fast open-source audio feature extractor. *Proceedings of the 18th ACM international conference on Multimedia*.

Wei Han, Hui Chen, Alexander Gelbukh, Amir Zadeh, Louis-philippe Morency, and Soujanya Poria. 2021a. Bi-bimodal modality fusion for correlation-controlled multimodal sentiment analysis. In *Proceedings of the 2021 International Conference on Multimodal Interaction*, pages 6–15.

Wei Han, Hui Chen, and Soujanya Poria. 2021b. Improving multimodal fusion with hierarchical mutual information maximization for multimodal sentiment analysis. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 9180–9192.

Md Kamrul Hasan, Sangwu Lee, Wasifur Rahman, Amir Zadeh, Rada Mihalcea, Louis-Philippe Morency, and Ehsan Hoque. 2021. Humor knowledge enriched transformer for understanding multimodal humor. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 35, pages 12972–12980.

Devamanyu Hazarika, Roger Zimmermann, and Soujanya Poria. 2020. Misa: Modality-invariant and -specific representations for multimodal sentiment analysis. *Proceedings of the 28th ACM International Conference on Multimedia*.

Kaiming He, Haoqi Fan, Yuxin Wu, Saining Xie, and Ross Girshick. 2020. Momentum contrast for unsupervised visual representation learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 9729–9738.

Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long short-term memory. *Neural computation*, 9(8):1735–1780.

Junnan Li, Ramprasaath R Selvaraju, Akhilesh Deepak Gotmare, Shafiq Joty, Caiming Xiong, and Steven Hoi. 2021. Align before fuse: Vision and language representation learning with momentum distillation. In *Thirty-Fifth Conference on Neural Information Processing Systems*.

Yaochen Liu, Yazhou Zhang, Qiuchi Li, Benyou Wang, and Dawei Song. 2021. What does your smile mean? jointly detecting multi-modal sarcasm and sentiment using quantum probability. In *Findings of the Association for Computational Linguistics: EMNLP 2021*, pages 871–880.

Zhun Liu, Ying Shen, Varun Bharadhwaj Lakshminarasimhan, Paul Pu Liang, AmirAli Bagher Zadeh, and Louis-Philippe Morency. 2018. Efficient low-rank multimodal fusion with modality-specific factors. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 2247–2256.

Wasifur Rahman, M. Hasan, Sangwu Lee, Amir Zadeh, Chengfeng Mao, Louis-Philippe Morency, and Ehsan Hoque. 2020. Integrating multimodal information in large pretrained transformers. *Proceedings of the conference. Association for Computational Linguistics. Meeting*, 2020:2359–2369.

Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael Bernstein, et al. 2015. Imagenet large scale visual recognition challenge. *International journal of computer vision*, 115(3):211–252.

Björn Schuller, Stefan Steidl, Anton Batliner, Felix Burkhardt, Laurence Devillers, Christian Müller, and Shrikanth Narayanan. 2010. The interspeech 2010 paralinguistic challenge. In *Proc. INTERSPEECH 2010, Makuhari, Japan*, pages 2794–2797.

Zhongkai Sun, Prathusha Sarma, William Sethares, and Yingyu Liang. 2020. Learning relationships between text, audio, and video via deep canonical correlation for multimodal language analysis. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pages 8992–8999.

Jiajia Tang, Kang Li, Xuanyu Jin, Andrzej Cichocki, Qibin Zhao, and Wanzeng Kong. 2021. Ctfn: Hierarchical learning for multimodal sentiment analysis using coupled-translation fusion network. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 5301–5311.

Yao-Hung Hubert Tsai, Shaojie Bai, Paul Pu Liang, J Zico Kolter, Louis-Philippe Morency, and Ruslan

Salakhutdinov. 2019. Multimodal transformer for unaligned multimodal language sequences. In *Proceedings of the conference. Association for Computational Linguistics. Meeting*, volume 2019, page 6558. NIH Public Access.

Yao-Hung Hubert Tsai, Paul Pu Liang, Amir Zadeh, Louis-Philippe Morency, and Ruslan Salakhutdinov. 2018. Learning factorized multimodal representations. In *International Conference on Learning Representations*.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in neural information processing systems*, pages 5998–6008.

Wenmeng Yu, Hua Xu, Ziqi Yuan, and Jiele Wu. 2021. Learning modality-specific representations with self-supervised multi-task learning for multimodal sentiment analysis. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 35, pages 10790–10797.

Jiahong Yuan, Mark Liberman, et al. 2008. Speaker identification on the scotus corpus. *Journal of the Acoustical Society of America*, 123(5):3878.

Amir Zadeh, Minghai Chen, Soujanya Poria, Erik Cambria, and Louis-Philippe Morency. 2017. Tensor fusion network for multimodal sentiment analysis. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 1103–1114.

Amir Zadeh, Paul Pu Liang, Navonil Mazumder, Soujanya Poria, Erik Cambria, and Louis-Philippe Morency. 2018. Memory fusion network for multi-view sequential learning. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 32.

Amir Zadeh and Paul Pu. 2018. Multimodal language analysis in the wild: Cmu-mosei dataset and interpretable dynamic fusion graph. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Long Papers)*.

Amir Zadeh, Rowan Zellers, Eli Pincus, and Louis-Philippe Morency. 2016. Multimodal sentiment intensity analysis in videos: Facial gestures and verbal messages. *IEEE Intelligent Systems*, 31(6):82–88.

Keyphrase Prediction from Video Transcripts: New Dataset and Directions

Amir Pouran Ben Veyseh[1], **Quan Hung Tran**[2], **Seunghyun Yoon**[2], **Varun Manjunatha**[2], **Hanieh Deilamsalehy**[2], **Rajiv Jain**[2], **Trung Bui**[2], **Walter W. Chang**[2], **Franck Dernoncourt**[2], **and Thien Huu Nguyen**[1]

[1]Department of Computer Science, University of Oregon, OR, USA
[2]Adobe Research, USA
{apouranb, thien}@cs.uoregon.edu
{franck.dernoncourt}@adobe.com

Abstract

Keyphrase Prediction (KP) is an established NLP task, aiming to yield representative phrases to summarize the main content of a given document. Despite major progress in recent years, existing works on KP have mainly focused on formal texts such as scientific papers or weblogs. The challenges of KP in informal-text domains are not yet fully studied. To this end, this work studies new challenges of KP in transcripts of videos, an understudied domain for KP that involves informal texts and non-cohesive presentation styles. A bottleneck for KP research in this domain involves the lack of high-quality and large-scale annotated data that hinders the development of advanced KP models. To address this issue, we introduce a large-scale manually-annotated KP dataset in the domain of live-stream video transcripts obtained by automatic speech recognition tools. Concretely, transcripts of 500+ hours of videos streamed on the behance.net platform are manually labeled with important keyphrases. Our analysis of the dataset reveals the challenging nature of KP in transcripts. Moreover, for the first time in KP, we demonstrate the idea of improving KP for long documents (i.e., transcripts) by feeding models with paragraph-level keyphrases, i.e., hierarchical extraction. To foster future research, we will publicly release the dataset and code.

1 Introduction

Keyphrases are one or multiple consecutive words that could represent the main ideas in a document. Keyphrases are commonly categorized as Present or Absent. A present keyphrase explicitly appears in the document, while an absent keyphrase does not exist in the document. Keyphrases can serve as concise summary for a document, hence benefiting various NLP applications Information Retrieval (Hersh, 2021) and Text Summarization (Adhikari et al., 2020). Due to their usefulness, in the more than two decades, KP has been studied in many research works (Turney, 2000; Wu et al., 2005; Jiang et al., 2009; Hasan and Ng, 2014; Mahata et al., 2018; Chen et al., 2020; Ye et al., 2021).

Whereas traditionally feature engineering has been used for KP (Turney, 2000; Sheeba and Vivekanandan, 2014), recently deep learning is proved to be more efficient for this task (Ye et al., 2021; Ahmad et al., 2021). However, one limitation in the current works is that they are mainly limited to the formal text such as scientific papers (Meng et al., 2017) and web-logs (Xiong et al., 2019). As such, the challenges in other domains are still unresolved. Among others, video transcript is one of the less-explored domains that could significantly benefit from KP. For instance, it could be used for video summarization and retrieval or benefit people who are deaf and hard of hearing (DHH) (Kafle et al., 2019). On the other hand, KP for transcripts that are automatically obtained are more challenging than the formal written documents as these transcripts involve noisy text, incomplete/repeated sentences and phrases, informal vocabulary, and noncohesive information flow. Although there have been a few related attempts to evaluate feature engineering methods on meeting transcripts (Sheeba and Vivekanandan, 2014, 2012), the available resources, with a handful of transcripts and keyphrases, are not useful to train/evaluate the recent advanced deep models.

To address such limitations, we propose a large manually-labeled dataset for the domain of video transcripts. Specifically, we collect 500+ live-stream videos from the Behance platform. The videos are automatically transcribed by Microsoft Automatic Speech Recognition (ASR) tool. Since the video transcripts might be lengthy, summarizing the entire transcript into a few keyphrases might be challenging. Moreover, such keyphrases might not be helpful for partial retrieval where a part of the transcript is requested. As such, we annotate the collected transcripts in two levels: (1) Para-

Proceedings of the 29th International Conference on Computational Linguistics, pages 7147–7155
October 12–17, 2022.

graphs: A paragraph, consisting of multiple consecutive sentences, is a chunk of a transcript that provides a single point. Annotators first identify the paragraphs in a transcript. Next, the representative keyphrases for every paragraph are annotated; (2) Chapter: In addition to the paragraphs, we ask the annotators to provide a few keyphrases that could summarize multiple consecutive paragraphs that convey a single topic (e.g., how to make a special edit on an image). We call these units "*Chapter*", which are comparable to documents in other KP datasets. Annotators will first find the boundaries for chapters, then provide the keyphrases for each chapter.

We conduct extensive analysis on both levels of the KP task on the prepared dataset. Our analysis shows that KP in transcripts is a challenging task and more research is required. More importantly, for the first time for KP, we show that extracting keyphrases of long documents in a hierarchical order could result in better performance on document level KP. Specifically, our analysis shows that obtaining paragraph-level keyphrases and providing them to chapter-level KP systems could significantly boost the performance. The provided dataset and analysis could bring forth opportunities for more research on transcripts for KP.

2 Data Annotation

Data Collection: This work aims to annotate KP data for the domain of ASR text. To this end, we employ live-stream videos released on the social media platform `Behance.net`. The videos are streamed by artists and designers to share/discuss their creative projects. As such, verbal content from the speakers (in English) is important for video understanding. While the videos have initial subjects, their content is unplanned, hence the streamer might cut sentences, discuss multiple topics, and employ informal phrases. The videos have an average length of 48 minutes. To obtain the verbal content of the streamed videos, we employ the Microsoft ASR tool. In total, 361 videos with a total length of more than 500 hours are transcribed. A transcript, on average, contains 7,219 words.

Annotation: As presented in the introduction, the lengthy nature of transcripts motivates us to annotate keyphrases at two levels. First, at the paragraph level, we define a paragraph in a transcript to have the same role as paragraphs in formal written documents. Concretely, a paragraph is defined as a chunk of text that conveys a particular point or idea. A transcript consists of multiple disjoint paragraphs. Since the ASR text does not provide paragraph information, we manually annotate the collected transcripts with paragraphs. Afterward, for each paragraph of the transcript, the important keyphrases are selected. To this end, a keyphrase for a paragraph should have the following features: (a) Concisely summarize the main idea in the paragraph; (b) Be related to the main subject of the video; (c) Explicitly appear in the paragraph; (d) Does not appear in the previous or next paragraphs; (e) Form a proper English noun/verb phrase. The paragraphs that are entirely off-topic do not have any keyphrases. Second, at the chapter level, we provide keyphrases for chapters in the transcripts. A chapter consists of multiple paragraphs to represent a single topic. For instance, in a photo editing video, the discussion on how to change the background can form a chapter. A keyphrase of a chapter should observe the following criteria: (a) Concisely summarize the main topics in the chapter; (b) May not explicitly appear in the chapter; (c) Does not overlap with the paragraph keyphrases or other chapter level keyphrases; (d) Form a proper English noun/verb phrase. Note that paragraphs and chapters might have multiple keyphrases that are sorted based on their importance.

To annotate data for each level, we hire 10 annotators from the `upwork.com` platform which is a website for hiring freelancers with different expertise. Since the collected videos are related to photo editing software, e.g., Photoshop, we require the annotators to have experience both in data annotation and in using major photo editing tools. We train the annotators for KP at each level. To prevent chapter-level keyphrases to be biased toward paragraph-level keyphrases, we split annotator pool for paragraph and chapter level annotation (five for each). The transcripts are distributed evenly to the five annotators at each level for annotation. As such, a transcript is annotated entirely by a paragraph annotator and a chapter annotator (including boundary annotation). Chapter annotation is done after and uses outputs from paragraph annotation.

Annotation Agreement: Following prior work (Xiong et al., 2019), we assess the task difficulty of KP over video transcripts by evaluating the agreements of annotators at different levels. For each annotation level, we ask all the five annotators to independently annotate a sample of 5% of the tran-

Cut-off	Paragraph		Chapter	
	Exact	Partial	Exact	Partial
Keyphrases@1	60.21%	62.93%	58.92%	60.16%
Keyphrases@2	45.18%	58.09%	41.14%	54.19%
Keyphrases@3	37.12%	49.18%	35.21%	49.12%

Table 1: Average of pair-wise agreements among annotators at different cutoffs for paragraph and chapter level KP.

Statistics	Paragraph	Chapter
Number of samples	19,597	2,742
Number of keyphrases	34,392	12,155
Avg. keyphrase per sample	1.75	4.35
Avg. length of keyphrase	1.36	1.69
Avg. sample length	133.21	1047.70

Table 2: Statistics of the proposed dataset. The number of keyphrases represents the total number of annotated keyphrases for each level. The length of a keyphrase or sample is expressed in terms of the number of words.

scripts. Afterward, we compute the agreements of the five annotators at cutoffs @1, @2, and @3 with same rank comparison, using Exact Match (a keyphrase position is counted if the keyphrase is exactly the same from the annotators), and Partial Match (a keyphrase position is counted if the keyphrases from the annotators share at least one word). Table 1 shows the average of pair-wise agreements between annotators (i.e., comparing each pair of annotators). This table shows that KP in transcripts is a challenging task for both chapters and paragraphs. We attribute the challenges in this domain to the disconnected information flow in spontaneous talking compared to formal written documents that follow a clear information flow. Moreover, KP at the chapter level imposes more challenges as the agreement between judges drops from paragraph to chapter level. Finally, we show the statistics of the dataset in Table 2. A sample annotation is also presented in Appendix A.

Model	Paragraph		Chapter	
	F1@3	F1@M	F1@3	F1@M
One2Set	35.12	38.72	25.16	28.33
SEG-NET	34.19	38.92	24.42	29.37
BART	35.74	39.09	26.71	30.98
T5	36.09	39.12	25.78	30.18
GPT-2	37.90	41.27	27.91	32.27

Table 3: Performance of the models on the test sets for paragraph and chapter level keyphrase prediction.

Model	Keyphrases		Sentences+Keyphrases	
	F1@3	F1@M	F1@3	F1@M
BART	29.89	31.99	30.91	33.51
T5	28.71	31.72	29.85	32.80
GPT-2	30.08	33.28	33.69	35.72

Table 4: Performance of models on the chapter level test set. "Keyphrases": models use paragraph keyphrases as input; "Sentences+Keyphrases": models employs both paragraph keyphrases and hosting sentences.

3 Experiments

We randomly split the 361 transcripts into train/development/test sets with the ratio 80/10/10, respectively. The paragraphs and chapters of the transcripts in each split are then employed for our experiments in this section. Specifically, we first assess the challenges of KP at each level. Next, we empirically study how paragraph-level information can be helpful for chapter-level KP.

Baselines: We evaluate the performance of the following baselines on the proposed dataset: (1) **Generative Language Models**: The content of a paragraph or chapter are prompted to a generative language model (LM) to produce the keyphrases. Specifically, the language models are trained in an auto-regressive manner on sequence $S = [w_1, \ldots, w_n, [SEP], kp_1, \ldots, kp_m]$, where w_i is the i-th word in the input paragraph or chapter and kp_i is the i-th keyphrase. We employ GPT-2 (Radford et al., 2019), BART (Lewis et al., 2020) and T5 (Raffel et al., 2020) as three different versions of this baseline[1]. Note that for the chapters, since the transformer-based LMs impose a length limit, we truncate the input to the length of the maximum size of the LMs; (2) **SEG-NET** (Ahmad et al., 2021): In this baseline, salient sentences in the input text are first selected, then keyphrases are predicted by a generative model consuming the selected salient sentences. To select important sentences, a binary classifier is trained to distinguish sentences that contain a present keyphrase or partially overlap with an absent keyphrase; and (3) **One2Set** (Ye et al., 2021): The prediction of keyphrases is modeled as a set prediction task. Instead of imposing an order on the output of a transformer-based decoder, the model predicts keyphrases in parallel. We evaluate the models based on the macro-averaged F1@3 and F1@M. In the former, the predictions are truncated/padded at cutoff 3 while in the latter all model

[1]For T5 and BART, the task is formulated as seq2seq and $[SEP]$ is used to separate the input and output sequences.

Model	Paragraph						Chapter					
	P@3	R@3	F1@3	P@M	R@M	F1@M	P@3	R@3	F1@3	P@M	R@M	F1@M
One2Set	39.54	31.58	35.12	40.45	37.13	38.72	24.80	25.53	25.16	30.04	26.80	28.33
SEG-NET	40.80	29.42	34.19	41.27	36.82	38.92	23.94	24.91	24.42	31.12	27.80	29.37
BART	37.98	33.74	35.74	36.51	42.06	39.09	21.43	35.44	26.71	28.19	34.38	30.98
T5	35.42	36.78	36.09	38.19	40.09	39.12	26.77	24.86	25.78	29.38	31.02	30.18
GPT-2	39.12	36.75	37.90	40.72	41.83	41.27	25.49	30.83	27.91	30.59	34.14	32.27

Table 5: Performance of the models on the test sets for paragraph and chapter level keyphrase prediction.

Model	With Paragraph Keyphrases						With Paragraph Keyphrases and Sentences					
	P@3	R@3	F1@3	P@M	R@M	F1@M	P@3	R@3	F1@3	P@M	R@M	F1@M
BART	31.30	28.60	29.89	30.83	33.24	31.99	27.13	35.91	30.91	32.00	35.16	33.51
T5	24.46	34.74	28.71	29.64	34.11	31.72	26.92	33.49	29.85	35.71	30.32	32.80
GPT-2	32.86	27.73	30.08	31.72	35.00	33.28	34.07	33.31	33.69	31.10	41.95	35.72

Table 6: Performance of the generative models on the chapter level test set.

predictions are employed. Finally, we fine-tune the hyper-parameters for the models on development data.

Results: Table 3 shows the performance of the baselines on the paragraph and chapter level test sets There are several observations from the table. First, models employing a pre-trained language model, i.e., BART, T5, and GPT-2, outperform the baselines that train the transformers from scratch, i.e., One2Set and SEG-NET. We will thus focus on the generative models BART, T5, and GPT-2 in the next experiments. Second, the models have better performance on the paragraph level than the chapter level. This is expected as the models are required to encode larger context at the chapter level. Also, as the models employ transformers with input length restriction, they cannot encode the entire chapter. Our next experiments will explore an approach to handle long documents for KP. Finally, the performance of KP models is still far from being perfect in our dataset, e.g., the F1@M of One2Set on the NUC dataset (Nguyen and Kan, 2007) is 13% better than those on our dataset at chapter level (Ye et al., 2021), thus further demonstrating the modeling challenges of KP in video transcripts and presenting room for further research.

To provide detailed performance of the models, we report the precision and recall at cutoffs 3 and M. Specifically, for P@3 and R@3, the model predictions are truncated to the first three predictions. Following prior work (Ye et al., 2021), for cases that the model predicts less than three keyphrases, the prediction is padded with random keyphrases to have three keyphrases. For P@M and R@M, all model predictions are employed to evaluate the performance. The model performance is presented in Tables 5 and 6.

Motivated by the intuition that comprehending long documents requires understanding their smaller segments, we postulate that chapter-level KP models should appropriately capture paragraph information. In particular, we argue that paragraph-level keyphrases should be extracted first to provide summarization for paragraphs to improve chapter-level KP models afterward (hierarchical extraction). As such, we explore two methods to study this intuition: (1) Instead of truncating chapters, the input to the chapter KP systems will be the keyphrases of the paragraphs in the chapters. During training, we concatenate the golden keyphrases of all paragraphs in a chapter, i.e., $S = [kp_1, [SEP], kp_2, [SEP], \ldots, kp_m[SEP]]$. The models then predict chapter-level keyphrases using S. At inference time, we use the pre-trained paragraph-level KP model, which is based on the same model for the chapter KP system, to form the sequence S; (2) Since the keyphrases might not fully cover context of paragraphs, we further concatenate the keyphrases and their host sentences in the paragraphs to form the sequence S. Formally, the input to the chapter level KP system is $S = [S_1, [SEP_S], kp_1, [SEP_p], S_2, \ldots, S_m, [SEP_S], kp_m, [SEP]$, where S_i is the sentence in the chapter that contains the keyphrase kp_i. Using the generative model baselines, the results for the two methods are presented in Table 4. Comparing the paragraph keyphrase-augmented models with their vanilla counterparts in Table 3, it is evident that providing paragraph keyphrases significantly improves the performance of all models. We attribute this to better representations that the models with paragraph-level information

can obtain for chapters. Moreover, comparing the mere use of keyphrases with the augmentation of both keyphrases and sentences, the latter produces higher performance for chapter models. Overall, such results corroborates our intuition about the benefits of paragraph-level keyphrases for chapter-level KP, thus suggesting a potential direction of hierarchical modeling of long documents for KP.

4 Related Works

Keyphrase Prediction (KP) has been studied extensively in the past (Barker and Cornacchia, 2000; Turney, 2000; Hulth, 2003; Wan and Xiao, 2008; Hasan and Ng, 2014; Ye et al., 2021). Prior works can be categorized into extraction-based and generation-based solutions. In the former, keyphrases are extracted from input text, using either rule-based methods (Medelyan et al., 2009) or deep learning models (Sun et al., 2020) via sequence labeling (Gollapalli et al., 2017). In the generation-based models, deep generative models are employed to encode input documents and generates keyphrases (Chen et al., 2018; Zhao and Zhang, 2019; Ahmad et al., 2021). However, existing works on KP are mostly trained and evaluated on formal text. To this end, our work introduces a large-scale hierarchical KP dataset for video transcripts with informal and non-cohesive texts. We also note some related attempts to evaluate KP systems on meeting transcripts (Sheeba and Vivekanandan, 2014, 2012); however, the small size of these datasets hinders their relevance to deep learning era.

5 Conclusion

We present a novel hierarchical KP dataset over live-stream video transcripts. The dataset contains transcripts of 361 videos that are annotated at both paragraph and chapter levels. Our experiments show that KP in video transcripts is challenging and hierarchical extraction is helpful for KP in long documents. In the future, we will include more tasks in our dataset for video transcripts.

Ethical Considerations

In this work we present a dataset on the transcripts of a publicly accessible video-streaming platform `behance.net`. Complying with the discussion presented by Benton et al. (2017), research with human subjects information is exempted from the required full Institutional Review Board (IRB) review if the data is already available from public sources or if the identity of the subjects cannot be recovered. However, to protect the identity of the streamers and any other people whose information are shared in the video transcript, we impose extra consideration on the presented dataset. First, in this dataset, we exclude the usernames or any other identity-related information of the streamers in the transcripts to prevent disclosing their identity. Moreover, the proposed dataset only provides textual data (at paragraph and sentence levels), hence the other content of the videos (e.g., images, audios) are not revealed to protect human identity. Finally, to reduce the risk of disclosing the information of the people mentioned in the transcripts, in the final version of the dataset, we exclude the transcripts that explicitly or implicitly refer to the identify of the target people.

Acknowledgement

This research has been supported by the Army Research Office (ARO) grant W911NF-21-1-0112 and the NSF grant CNS-1747798 to the IUCRC Center for Big Learning. This research is also based upon work supported by the Office of the Director of National Intelligence (ODNI), Intelligence Advanced Research Projects Activity (IARPA), via IARPA Contract No. 2019-19051600006 under the Better Extraction from Text Towards Enhanced Retrieval (BETTER) Program. The views and conclusions contained herein are those of the authors and should not be interpreted as necessarily representing the official policies, either expressed or implied, of ARO, ODNI, IARPA, the Department of Defense, or the U.S. Government.

References

Ashutosh Adhikari, Achyudh Ram, Raphael Tang, William L Hamilton, and Jimmy Lin. 2020. Exploring the limits of simple learners in knowledge distillation for document classification with docbert. In *Proceedings of the 5th Workshop on Representation Learning for NLP*.

Wasi Ahmad, Xiao Bai, Soomin Lee, and Kai-Wei Chang. 2021. Select, extract and generate: Neural keyphrase generation with layer-wise coverage attention. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*.

Ken Barker and Nadia Cornacchia. 2000. Using noun phrase heads to extract document keyphrases. In *conference of the canadian society for computational studies of intelligence*.

Adrian Benton, Glen Coppersmith, and Mark Dredze. 2017. Ethical research protocols for social media health research. In *Proceedings of the First ACL Workshop on Ethics in Natural Language Processing*, pages 94–102, Valencia, Spain. Association for Computational Linguistics.

Jun Chen, Xiaoming Zhang, Yu Wu, Zhao Yan, and Zhoujun Li. 2018. Keyphrase generation with correlation constraints. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*.

Wang Chen, Hou Pong Chan, Piji Li, and Irwin King. 2020. Exclusive hierarchical decoding for deep keyphrase generation. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*.

Sujatha Das Gollapalli, Xiao-Li Li, and Peng Yang. 2017. Incorporating expert knowledge into keyphrase extraction. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 31.

Kazi Saidul Hasan and Vincent Ng. 2014. Automatic keyphrase extraction: A survey of the state of the art. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*.

William Hersh. 2021. Information retrieval. In *Biomedical Informatics*.

Anette Hulth. 2003. Improved automatic keyword extraction given more linguistic knowledge. In *Proceedings of the 2003 Conference on Empirical Methods in Natural Language Processing*.

Xin Jiang, Yunhua Hu, and Hang Li. 2009. A ranking approach to keyphrase extraction. In *Proceedings of the 32nd international ACM SIGIR conference on Research and development in information retrieval*.

Sushant Kafle, Peter Yeung, and Matt Huenerfauth. 2019. Evaluating the benefit of highlighting key words in captions for people who are deaf or hard of hearing. In *The 21st International ACM SIGACCESS Conference on Computers and Accessibility, ASSETS 2019, Pittsburgh, PA, USA, October 28-30, 2019*.

Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Veselin Stoyanov, and Luke Zettlemoyer. 2020. BART: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7871–7880.

Debanjan Mahata, John Kuriakose, Rajiv Ratn Shah, and Roger Zimmermann. 2018. Key2Vec: Automatic ranked keyphrase extraction from scientific articles using phrase embeddings. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 2 (Short Papers)*.

Olena Medelyan, Eibe Frank, and Ian H. Witten. 2009. Human-competitive tagging using automatic keyphrase extraction. In *Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing*.

Rui Meng, Sanqiang Zhao, Shuguang Han, Daqing He, Peter Brusilovsky, and Yu Chi. 2017. Deep keyphrase generation. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*.

Thuy Dung Nguyen and Min-Yen Kan. 2007. Keyphrase extraction in scientific publications. In *International conference on Asian digital libraries*, pages 317–326. Springer.

Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. 2019. Language models are unsupervised multitask learners. volume 1, page 9.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J. Liu. 2020. Exploring the limits of transfer learning with a unified text-to-text transformer. In *J. Mach. Learn. Res.*

JI Sheeba and K Vivekanandan. 2012. Improved keyword and keyphrase extraction from meeting transcripts. In *International Journal of Computer Applications*.

JI Sheeba and K Vivekanandan. 2014. A fuzzy logic based improved keyword extraction from meeting transcripts. In *International Journal on Computer Science and Engineering*.

Si Sun, Zhenghao Liu, Chenyan Xiong, Zhiyuan Liu, and Jie Bao. 2020. Capturing global informativeness in open domain keyphrase extraction. In *arXiv preprint arXiv:2004.13639*.

Peter D Turney. 2000. Learning algorithms for keyphrase extraction. In *Information retrieval*.

Xiaojun Wan and Jianguo Xiao. 2008. Single document keyphrase extraction using neighborhood knowledge. In *AAAI*.

Yi-fang Brook Wu, Quanzhi Li, Razvan Stefan Bot, and Xin Chen. 2005. Domain-specific keyphrase extraction. In *Proceedings of the 14th ACM international conference on Information and knowledge management*, pages 283–284.

Lee Xiong, Chuan Hu, Chenyan Xiong, Daniel Campos, and Arnold Overwijk. 2019. Open domain web keyphrase extraction beyond language modeling. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*.

Jiacheng Ye, Tao Gui, Yichao Luo, Yige Xu, and Qi Zhang. 2021. One2Set: Generating diverse keyphrases as a set. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*.

Jing Zhao and Yuxiang Zhang. 2019. Incorporating linguistic constraints into keyphrase generation. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*.

A Sample Annotation

To illustrate the annotated data, we present a sample annotation for a chapter in Table 7. This table shows three paragraphs of the chapter along with their keyphrases. Note that, first boundaries of the paragraphs in the transcripts are annotated. Next, for every paragraph annotators provide a few keyphrases that could summarize the main topic/points in the paragraph. Afterward, boundaries of the chapters in the transcripts, which consist of multiple paragraphs, are annotated. Finally, for every chapter, keyphrases that could best describe the main content of the chapter are provided by annotators. In the given example, the paragraph level keyphrases include "*Camera*", "*Background lights*", and "*Environment light, Rotations*" for the three paragraphs. For this chapter, the keyphrase "*Setting Environment*" is provided.

ID	Content	Paragraph Keyphrases
1	We have beautifully beautiful summer day outside with our cup of coffee. If you ever log on your arm. All right up corner you will see that currently we are located in the camera view. If you would like to adjust your 3D model I will recommend you to switch to viewport camera. In this case you will not affect your camera perspective during your model adjustment, so keep it in mind when you will be ready to come back to your camera view. Simply switch from a top corner or directly from you seen a pen or just simply click on camera.	Camera
2	Just like duck. Light are you can come. Ah, click on environment. In. Here you can adjust background lights. Opposite team environment might need background blue. You can make it more blurry or or less blurry. Also if you will switch to light you will be able to adjust.	Background lights
3	Your light you can. Uh, idiot environment light just like that. You can make it brighter or more cloudy also rotation. You can rotate your alight so keep before you will rotate your light. Keep in mind and pay close attention to your background image to your main source of light just like that.	Environment light, Rotation

Table 7: Sample annotations for keyphrases of a chapter. Annotators first find the boundaries of the paragraphs, then provide keyphrases for every paragraph. At the chapter level, annotators identify paragraphs to form chapters before assigning keyphrases for chapters. The keyphrase "*Setting Environment*" is provided for the chapter (with three paragraphs) in this example.

Event Extraction in Video Transcripts

Amir Pouran Ben Veyseh[1], **Viet Dac Lai**[1],
Franck Dernoncourt[2], **and Thien Huu Nguyen**[1]
[1]Department of Computer Science, University of Oregon, OR, USA
[2]Adobe Research, Seattle, WA, USA
{apouranb, vietl, thien}@cs.uoregon.edu
{franck.dernoncourt}@adobe.com

Abstract

Event extraction (EE) is one of the fundamental tasks for information extraction whose goal is to identify mentions of events and their participants in text. Due to its importance, different methods and datasets have been introduced for EE. However, existing EE datasets are limited to formally written documents such as news articles or scientific papers. As such, the challenges of EE in informal and noisy texts are not adequately studied. In particular, video transcripts constitute an important domain that can benefit tremendously from EE systems (e.g., video retrieval), but has not been studied in EE literature due to the lack of necessary datasets. To address this limitation, we propose the first large-scale EE dataset obtained for transcripts of streamed videos on the video hosting platform Behance to promote future research in this area. In addition, we extensively evaluate existing state-of-the-art EE methods on our new dataset. We demonstrate that such systems cannot achieve adequate performance on the proposed dataset, revealing challenges and opportunities for further research effort.

1 Introduction

Event Extraction is an important task in the full pipeline of Information Extraction. In EE, the goal is to identify mentions/trigger words of events, and their participants and attributes of interest. For instance, in the sentence "*Joe Biden was born on November 20, 1942*", an event of *Birth* is mentioned. An event mention consists of two important components: (1) **Trigger**: the word(s) that most clearly refer to the occurrence of the event (e.g, "*born*" in the above example); and (2) **Argument**: the entity mentions involved in the event with some role (e.g., "*Joe Biden*" with the role of *Entity*) or attributes of the event (e.g., time and location).

Due to its importance, various methods and annotated datasets have been proposed for EE (Ahn, 2006; Nguyen and Grishman, 2015; Yang et al., 2019; Wang et al., 2020). Also, with the proliferation of EE methods, datasets for EE have been diversified to cover different domains and settings, e.g., multiple domains (Walker et al., 2006), multiple languages, (Mitamura et al., 2016), literary texts (Sims et al., 2019), cybersecurity texts (Man Duc Trong et al., 2020), and events in long documents (Ebner et al., 2020). However, despite all progress thus far, most of the available datasets for EE are restricted to the domains of formally written texts, e.g., news, reports, scientific papers, or books. As such, the challenges for EE in other domains with informal and noisy texts are less explored. One of such domains that has not been studied before for EE involves video transcripts obtained by automatic speech recognition (ASR) tools. Since the such transcripts might be noisy, e.g., incomplete sentences, incorrect words selected by the ASR tool, lack of correct punctuation and segmentation, repeated words or sentences, etc., existing EE models are not well evaluated and might not perform well in this domain. This is unfortunate as an effective EE model can be extremely helpful for downstream applications that utilize video transcripts. For instance, search engines can employ events detected in a transcript to locate relevant portion of a video to a query. It can also benefit video summarization, knowledge base construction, and script generation from videos. As such, it is necessary to study the challenges and potential directions for EE improvement in the video transcript domain.

Due to the lack of EE datasets for video transcript domain, we propose the first large-scale EE dataset annotated for transcripts of streamed videos on the popular video hosting platform Behance. Videos in this platform are streamed by artists who would like to share their creative projects using Adobe Creative Cloud products (e.g., Photoshop, Illustrator, etc). Videos have been first transcribed using the Microsoft Automatic Speech Recognition tool. In order to annotate the events mentioned in

Proceedings of the 29th International Conference on Computational Linguistics, pages 7156–7165
October 12–17, 2022.

the video transcripts, we first define a taxonomy of event types and their arguments for the domain of creative projects (e.g., Add a shape, Modify the color of an object, etc.). Using the pre-processed transcripts and the provided event ontology, we hire annotators with domain expertise to provide high-quality annotation for event triggers and their arguments. In addition, we employ the annotated dataset to evaluate the performance of the state-of-the-art (SOTA) EE models. Compared to the domains with formally written texts, our analysis shows that the current SOTA EE models fail to achieve comparable performance on video transcripts. This performance drop indicates the challenging nature of video transcripts and call for more research effort for EE in this domain. We will publicly release our dataset, called *TranscriptEE*, to foster future research in this area.

2 Dataset

Data Collection: In this work, we propose to employ the videos streamed on the popular video-hosting platform Behance[1] to obtain transcripts to be annotated for event mentions. Behance is a platform in which artists can share their creative projects using Adobe Creative Cloud products (e.g., Photoshop, Illustrator, etc). Most of the information is transmitted verbally in English in these videos. Each video lasts from few minutes to several hours. In the first step, we collect 370 videos with a total duration of 500 hours. On average, a video lasts 48 minutes. Next, for each video, we employ the Microsoft Automatic Speech Recognition tool to obtain transcripts of the videos. A video transcript on average contains 7,219 words. To facilitate the annotation process, following prior work for EE dataset creation (Ebner et al., 2020), we split the transcripts of the videos into chunks (called paragraphs) of 5 consecutive utterances. In total, 25,492 paragraphs are obtained for EE annotation.

Annotation: To annotate the data, we first define an ontology of event types and argument roles for the domain of creative projects. Specifically, an event is defined as an action that results in a visual change in the project (e.g., changing the color of an object, adding a new shape to an illustration, modifying the texture of the surfaces in an image, etc.). Concretely, we categorize the events into four types: (1) **Add**: A new visual element is added to the project; (2) **Modify**: One of the attributes of an existing element (e.g., color, size, texture, etc.) is changed; (3) **Select**: Some objects in the project are selected using tools of editing programs (e.g., the "Lasso" tool in Photoshop); and (4) **Remove**: An object is removed from the project. For each event, we also define their arguments, e.g., Tool, Object, Color, etc. We present a description of event types and arguments, along with their examples in Appendix A. To select paragraphs for annotation, we design a set of keywords that are relevant to our event types (e.g., "*add*", "*modify*", "*select*", "*pick*", "*remove*", "*delete*"). Paragraphs with the highest matching rates for the keyword set are retained for EE annotation. Overall, to accommodate our budget, 2,162 top paragraphs are annotated in our dataset.

We employ human annotation to find event mentions in the paragraphs of video transcripts. To annotate event triggers, we follow prior work on ED (Walker et al., 2006) to ask the annotators to select the word or phrases (e.g., a phrasal verb) that most clearly mention the occurrences of events. Also, for event arguments, we require the annotators to select the head words of the noun phrases that refer to the arguments of events. Event triggers and arguments can belong to different sentences in the paragraphs in our dataset.

In this work, we leverage Upwork, a freelancer platform, to hire expert annotators. The hired annotators have experience in both creative projects (e.g., using photo editing programs such as Photoshop) and data annotation. We train the annotators with the event ontology and designed examples. Based on the performance of the annotators in a pilot study, we select the final pool of five annotators. To compute the inter-annotation agreement (IAA) scores, 20% of the paragraphs are shared by the annotators for co-annotation while the remaining 80% is distributed evenly for the annotators. As such, annotators first independently identify event triggers in the shared paragraphs, achieving an agreement score of 0.812 for the Krippendorff's alpha (Krippendorff, 2011) with MASI distance metric (Passonneau, 2006). Afterward, the annotators discuss to resolve conflict cases for the co-annotated data, and then perform annotation individually on the remaining data to produce the final version of event triggers in our dataset. In the next step, given the annotated triggers, annotators also independently annotate event arguments for the shared

[1] www.behance.net

Statistics	Total	Train	Dev	Test
# Paragraphs	2162	1729	216	217
# Triggers	3180	2580	283	317
# Arguments	3427	2798	295	334
Avg. # Triggers / Sample	1.47	1.49	1.31	1.46
Avg. # Arguments / Trigger	1.59	1.62	1.37	1.54

Table 1: Statistics of *TranscriptEE*.

paragraphs, achieving the Krippendorff's alpha of 0.783. Finally, conflict argument examples in the co-annotated data are resolved and individual annotation on the rest of the data is done by the annotators to generate the final version of our dataset *TranscriptEE*. As such, we achieve strong agreement scores for both event trigger and argument annotation, showing the high quality of *TranscriptEE*. To facilitate future research, we randomly split the dataset into separate training, development, and test sets with the ratio of 80/10/10, respectively. The statistics for each data split along the entire dataset are presented in Table 1.

Annotation Challenges: This section describes some major challenges we encounter in the annotation process for *TranscriptEE*. (1) **False Triggers**: In some cases, the streamer discusses a general action without actually doing the action. For instance, in the sentence "*Croping the images is very easy in Photoshop*", the streamer mentions an edit action (i.e., "*cropping*") which can be considered as a Modify event without implying actual implementation of such actions in his/her work. These examples cause disagreements among the annotators on whether an event trigger should be annotated or not. To resolve this situation, we ask the annotators to not annotate event triggers that the streamer does not clearly imply their occurrence. (2) **Confusing Triggers**: Depending on the object of consideration, some event triggers can be interpreted as either an Modify or Add event, thus bewildering the annotators for correct annotation. For instance, in the sentence "*First, I **create** some shadows for letters.*", the word "*create*" can refer to an Add event with the object "*shadow*"; however, it can also evoke the event type Modify with the object "*letters*". To resolve this conflict, we require the annotators to select the more general event type, i.e., the type Modify in our example.

Dataset Challenges: In addition to the typical challenges of EE (e.g., ambiguous triggers that can trigger different event types depending on contexts), an unique challenge of *TranscriptEE* for EE models involves background knowledge. In particular, recognizing domain knowledge about technical terms for editing programs in creative projects is necessary for the models to make correct predictions in *TranscriptEE*. For instance, in the sentence "*We prefer **burn** to make shadows darker*", to select the word "*burn*" as the argument for the Modify event trigger "*make*", it is important for the models to realize that "*burn*" is a tool name in Photoshop.

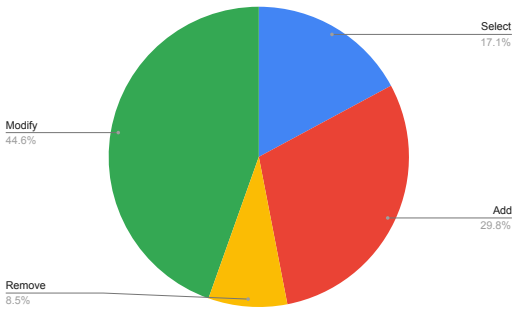

Figure 1: Distribution of event types in the proposed dataset *TranscriptEE*.

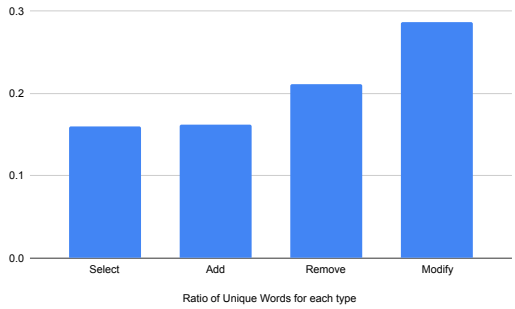

Figure 2: Ratio of number of unique trigger words to their frequency for each event type.

Dataset Analysis: In order to shed more light for the proposed dataset *TranscriptEE*, we report the distribution of the event types in Figure 1. This figure shows that the Modify event type has the

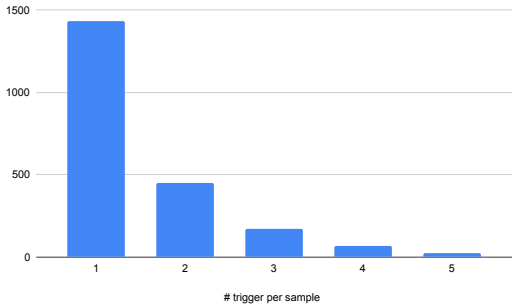

Figure 3: Distributions of number of event triggers per paragraph.

highest frequency in the dataset, followed by Add, Select and Remove. Moreover, to study the challenging nature of each event type, we study the ratio of the number of unique event triggers to the frequency of each event type. The higher this ratio, the more challenging the event type is as it expresses more diverse ways to present an event in the dataset. The results are presented in Figure 2. This figure demonstrates that the Modify event type employs the most diverse set of triggers followed by the event type Remove. Considering the low frequency of Remove with its high ratio of trigger diversity, it also implies the challenging nature of this event type. Finally, Figure 3 shows the distribution for the numbers of trigger words per paragraph in *TranscriptEE*. As can be seen, while the majority of the paragraphs have one trigger word, nearly 30% of the dataset involves more than one trigger words. It thus suggests an opportunity to exploit the corelation of event triggers and types to improve EE performance on *TranscriptEE*.

3 Experiments

We study the performance of existing state-of-the-art EE systems on the proposed dataset *TranscriptEE*. Since EE consists of two sub-tasks, i.e., Event Detection (ED) and Event Argument Extraction (EAE), we consider two types of baselines:

(1) **Pipeline Modeling**: In this category, an ED system is first employed to identify event triggers with their types in the input text. Next, given a predicted event trigger, an EAE system is utilized to recognize arguments and their roles for the event trigger. As such, we use the **BERT** model (Devlin et al., 2019) as the baseline model for ED and EAE in the pipeline approach as in prior work (Yang et al., 2019; Wang et al., 2020). Specifically, the input paragraph, in the form of $D = [[CLS], w_1, \ldots, w_n, [SEP]]$ with n words, is first fed into the BERT model (the base cased version) of the ED system to predict the triggers and their types. The ED task is modeled as a sequence labeling problem using the BIO tagging schema to encode the labels for each word in D. Next, the predicted event type and trigger will be concatenated with the input document D to be consumed by the BERT model of the EAE system for argument prediction, i.e., $[Type, Trigger, [SEP], w_1, \ldots, w_n]$. Here, $Type$ and $Trigger$ are predicted by the ED system. The EAE task is also modeled as an sequence labeling for argument roles. In addition, we study the performance of **BERT+CRF** model, where a conditional random field (CRF) layer is added on top of the BERT models for the ED and EAE architectures for sequence labeling.

Model	ED			EAE		
	P	R	F1	P	R	F1
BERT	57.47	59.93	59.69	39.21	44.74	41.79
BERT+CRF	56.42	59.94	58.13	41.14	39.45	40.28

Table 2: ED and EAE performance of pipeline models on the test set of *TranscriptEE*.

(2) **Joint Modeling**: In this category, the systems jointly predict event triggers and their arguments for an input text in an end-to-end fashion. As such, we consider four typical joint models for EE. The first two models employ similar architectures as **BERT** and **BERT+CRF** where BERT is utilized to encode the input paragraph D to produce representation vectors for each word. The word representations are then fed into a feed forward layer (and a CRF layer in case of **BERT+CRF**) to identify event trigger and argument spans with sequence labeling. Next, trigger and argument representations are obtained by averaging the word representations inside the detected spans. Finally, the trigger representations are sent to a feed-forward network for event type prediction while pairs of trigger and argument representations are consumed by another feed-forward network to predict argument roles. **BERT** and **BERT+CRF** are trained end-to-end with the combined loss from different components. Our third joint baseline involves **OneIE** (Lin et al., 2020) that is similar to the **BERT+CRF** joint baseline. However, instead of using greedy decoding as in **BERT+CRF**, **OneIE** manually designs global features to capture label dependencies among different IE tasks to improve beam search decoding. Our fourth joint baseline explores **FourIE** (Nguyen et al., 2021) that differs **OneIE** in that **FourIE** exploits instance and label dependencies to improve representation learning in the training step (via Graph Convolutional Networks and consistency regularization). Note that we leverage the original implementations and remove the relation extraction task from **OneIE** and **FourIE** for our joint EE problem. **OneIE** and **FourIE** are among the current state-of-the-art (SOTA) methods for EE.

Performance of the pipeline and joint models is presented in Tables 2 and 3, respectively. The joint models outperform their counterpart pipeline systems. Specifically, **BERT** and **BERT+CRF** enjoy 2.73 and 2.99 F1 point improvement for ED,

Model	ED			EAE		
	P	R	F1	P	R	F1
OneIE	60.72	54.38	57.38	50.70	45.82	48.14
FourIE	58.48	59.95	59.21	52.55	46.44	49.31
BERT	61.19	63.70	62.42	52.43	51.02	51.72
BERT+CRF	60.98	61.26	61.12	51.65	50.15	50.89

Table 3: ED and EAE performance of joint models on the test set of *TranscriptEE*.

and 9.93 and 10.61 F1 point improvement for EAE respectively. This can be attributed to the shared parameters of BERT in joint **BERT** and **BERT+CRF** that enrich the induced representation vectors to improve the prediction. Also, among the joint models, the simpler methods **BERT** and **BERT+CRF** actually perform better than the more complicated models **OneIE** and **FourIE** that exploit label dependency of the tasks for training and decoding. This indicates that the methods to capture label dependence in **OneIE** and **FourIE** are not helpful for EE in *TranscriptEE*, thus calling for more research effort to design more suitable EE models in this domain. Finally, the performance of existing SOTA methods for EE over *TranscriptEE* is still far from being perfect that presents much room for future research.

4 Related Works

Previous EE systems can be classified according to the representation construction methods, i.e., feature engineering (Ahn, 2006; Liao and Grishman, 2010; Li et al., 2013) vs. deep learning for representation learning (Nguyen and Grishman, 2015b; Chen et al., 2015; Nguyen and Grishman, 2018; Chen et al., 2018; Wang et al., 2019a), or the formulation approaches, i.e., pipeline (Ahn, 2006; Yang et al., 2019; Wang et al., 2019b) vs. joint models (Yang and Mitchell, 2016; Nguyen and Nguyen, 2019; Lin et al., 2020; Nguyen et al., 2021, 2022). A majority of prior EE work utilize the ACE 2005 (Walker et al., 2006) and TAC KBP datasets (Mitamura et al., 2016) that focus on newswire article domains. Recently, there have more efforts to create EE datasets for more diverse domains, including biomedical (Kim et al., 2011), literary (Sims et al., 2019), cybersecurity (Man Duc Trong et al., 2020), Wikipedia (Wang et al., 2020), multilingual (Pouran Ben Veyseh et al., 2022; Lai et al., 2022), history (Lai et al., 2021), and suicide understanding (Guzman-Nateras et al., 2022). However, none of such prior work and datasets explores EE for video transcripts.

5 Conclusion

We present *TranscriptEE*, the first manually annotated dataset for EE for video transcripts. The videos are obtained from the Behance platform which is dedicated to sharing creative projects. *TranscriptEE* contains more than 2,000 labeled paragraphs for various edit events in creative projects with high quality. Our analysis with state-of-the-art EE models reveals challenging nature of the dataset for future research.

Ethical Considerations

In this work we present a dataset on the transcripts of a publicly accessible video-streaming platform, i.e., "*Behance*"[2]. Complying with the discussion presented by Benton et al. (2017), research with human subjects information is exempted from the required full Institutional Review Board (IRB) review if the data is already available from public sources or if the identity of the subjects cannot be recovered. However, to protect the identity of the streamer and any other person whose information are shared in the video transcript, we impose extra processing on the presented dataset before presenting it to annotators and publicly releasing it later. First, in this dataset, we exclude username or any other identity-related information of the streamers in the transcripts to prevent disclosing their identity. Moreover, the proposed dataset only provides textual data (i.e., paragraphs), hence the other content of the videos (e.g., images, audios) are not revealed (to annotators or users) to protect human identity. Finally, to reduce the risk of disclosing the information of the people mentioned in the transcripts, in the final version of the dataset, we exclude the transcripts that explicitly or implicitly refer to the identify of the target people.

Acknowledgement

This research has been supported by the Army Research Office (ARO) grant W911NF-21-1-0112 and the NSF grant CNS-1747798 to the IU-CRC Center for Big Learning. This research is also based upon work supported by the Office of the Director of National Intelligence (ODNI), Intelligence Advanced Research Projects Activity (IARPA), via IARPA Contract No. 2019-19051600006 under the Better Extraction from Text Towards Enhanced Retrieval (BETTER) Program.

[2] www.behance.net

References

David Ahn. 2006. The stages of event extraction. In *Proceedings of the Workshop on Annotating and Reasoning about Time and Events*.

Adrian Benton, Glen Coppersmith, and Mark Dredze. 2017. Ethical research protocols for social media health research. In *Proceedings of the First ACL Workshop on Ethics in Natural Language Processing*, pages 94–102, Valencia, Spain. Association for Computational Linguistics.

Yubo Chen, Liheng Xu, Kang Liu, Daojian Zeng, and Jun Zhao. 2015. Event extraction via dynamic multi-pooling convolutional neural networks. In *Proceedings of the Annual Meeting of the Association for Computational Linguistics (ACL)*.

Yubo Chen, Hang Yang, Kang Liu, Jun Zhao, and Yantao Jia. 2018. Collective event detection via a hierarchical and bias tagging networks with gated multi-level attention mechanisms. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP)*.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL-HLT)*.

Seth Ebner, Patrick Xia, Ryan Culkin, Kyle Rawlins, and Benjamin Van Durme. 2020. Multi-sentence argument linking. In *Proceedings of the Annual Meeting of the Association for Computational Linguistics (ACL)*.

Luis Guzman-Nateras, Viet Lai, Amir Pouran Ben Veyseh, Franck Dernoncourt, and Thien Nguyen. 2022. Event detection for suicide understanding. In *Findings of the Association for Computational Linguistics: NAACL 2022*, pages 1952–1961, Seattle, United States. Association for Computational Linguistics.

Jin-Dong Kim, Tomoko Ohta, Sampo Pyysalo, Yoshinobu Kano, and Jun'ichi Tsujii. 2011. Extracting bio-molecular events from literature—the bionlp'09 shared task. *Computational Intelligence*, 27(4):513–540.

Klaus Krippendorff. 2011. Computing krippendorff's alpha-reliability.

Viet Lai, Minh Van Nguyen, Heidi Kaufman, and Thien Huu Nguyen. 2021. Event extraction from historical texts: A new dataset for black rebellions. In *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*.

Viet Dac Lai, Amir Pouran Ben Veyseh, Minh Van Nguyen, Franck Dernoncourt, and Thien Huu Nguyen. 2022. Meci: A multilingual dataset for event causality identification. In *Proceedings of the COLING 2022*. Association for Computational Linguistics.

Qi Li, Heng Ji, and Liang Huang. 2013. Joint event extraction via structured prediction with global features. In *Proceedings of the Annual Meeting of the Association for Computational Linguistics (ACL)*.

Shasha Liao and Ralph Grishman. 2010. Using document level cross-event inference to improve event extraction. In *Proceedings of the Annual Meeting of the Association for Computational Linguistics (ACL)*.

Ying Lin, Heng Ji, Fei Huang, and Lingfei Wu. 2020. A joint neural model for information extraction with global features. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*.

Hieu Man Duc Trong, Duc Trong Le, Amir Pouran Ben Veyseh, Thuat Nguyen, and Thien Huu Nguyen. 2020. Introducing a new dataset for event detection in cybersecurity texts. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*.

Teruko Mitamura, Zhengzhong Liu, and Eduard H. Hovy. 2016. Overview of TAC-KBP 2016 event nugget track. In *Proceedings of the Text Analysis Conference (TAC)*.

Minh Van Nguyen, Viet Lai, and Thien Huu Nguyen. 2021. Cross-task instance representation interactions and label dependencies for joint information extraction with graph convolutional networks. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*.

Minh Van Nguyen, Bonan Min, Franck Dernoncourt, and Thien Nguyen. 2022. Joint extraction of entities, relations, and events via modeling inter-instance and inter-label dependencies. In *Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 4363–4374, Seattle, United States. Association for Computational Linguistics.

Thien Huu Nguyen and Ralph Grishman. 2015. Event detection and domain adaptation with convolutional neural networks. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (ACL-IJCNLP)*.

Thien Huu Nguyen and Ralph Grishman. 2015b. Event detection and domain adaptation with convolutional neural networks. In *ACL*.

Thien Huu Nguyen and Ralph Grishman. 2018. Graph convolutional networks with argument-aware pooling for event detection. In *Proceedings of the Association for the Advancement of Artificial Intelligence (AAAI)*.

Trung Minh Nguyen and Thien Huu Nguyen. 2019. One for all: Neural joint modeling of entities and events. In *Proceedings of the Association for the Advancement of Artificial Intelligence (AAAI)*.

Rebecca Passonneau. 2006. Measuring agreement on set-valued items (masi) for semantic and pragmatic annotation. In *LREC*.

Amir Pouran Ben Veyseh, Minh Van Nguyen, Franck Dernoncourt, and Thien Nguyen. 2022. MINION: a large-scale and diverse dataset for multilingual event detection. In *Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 2286–2299, Seattle, United States. Association for Computational Linguistics.

Matthew Sims, Jong Ho Park, and David Bamman. 2019. Literary event detection. In *Proceedings of the Annual Meeting of the Association for Computational Linguistics (ACL)*.

Christopher Walker, Stephanie Strassel, Julie Medero, and Kazuaki Maeda. 2006. Ace 2005 multilingual training corpus. In *Technical report, Linguistic Data Consortium*.

Xiaozhi Wang, Xu Han, Zhiyuan Liu, Maosong Sun, and Peng Li. 2019a. Adversarial training for weakly supervised event detection. In *Proceedings of the Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL-HLT)*.

Xiaozhi Wang, Ziqi Wang, Xu Han, Wangyi Jiang, Rong Han, Zhiyuan Liu, Juanzi Li, Peng Li, Yankai Lin, and Jie Zhou. 2020. MAVEN: A Massive General Domain Event Detection Dataset. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP)*.

Xiaozhi Wang, Ziqi Wang, Xu Han, Zhiyuan Liu, Juanzi Li, Peng Li, Maosong Sun, Jie Zhou, and Xiang Ren. 2019b. HMEAE: Hierarchical modular event argument extraction. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*.

Bishan Yang and Tom M. Mitchell. 2016. Joint extraction of events and entities within a document context. In *Proceedings of the Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL-HLT)*.

Sen Yang, Dawei Feng, Linbo Qiao, Zhigang Kan, and Dongsheng Li. 2019. Exploring pre-trained language models for event extraction and generation. In *Proceedings of the Annual Meeting of the Association for Computational Linguistics (ACL)*.

A Event Types & Argument Roles

In this work, we define the event types based on edit actions performed during a creative project. Specifically, an event is change of state that can potentially refer to a visual change in an image. As such, we define four event types "*Add*", "*Remove*", "*Modify*", and "*Select*". Their description and examples are presented in Table 4. Moreover, each event type can involve multiple arguments. Argument are the objects, tools and properties employed for the edit action. The list of available argument roles for each event type is presented in Table 5.

B Annotation Instruction and Tool

We present the instructions provided to the annotators in Figure 4. In this work, we employ the BRAT[3] annotation tool (MIT License). A screenshot of the annotation tool is presented in Figure 5.

[3] https://brat.nlplab.org/

1. General Instructions

In this project, we annotate the transcripts of Behance videos. These are videos streamed on behance.net related to various Adobe Creative Cloud products (e.g., Photoshop, Fresco, XD, Illustrator, etc). In this project, we will focus on the Event Extraction task. An event is a specific occurrence involving participants. Specifically, an event in a Behance video transcript refers to some visual changes in the image that the streamer is working on and it is mentioned in the video transcript. We will not be tagging all events, only examples of a particular set of types are annotated. Specifically, we are interested in annotating events that can represent an edit action, i.e. Select, Add, Remove, Modify.

An event mention has two parts:

- **Trigger:** A word that most clearly mentions the occurrence of the event. For instance, in the sentence “What I’m doing now is to just **move** these boxes to the background, so we can see the texture here”, the word “move” is the event trigger and clearly mentions that an object is being modified in the image.
- **Argument:** In an event, multiple entities are involved. An entity is a text span that refers to something that is involved in the event. The participants of events play a role in the event. Technically, they are called “event arguments”. For instance, in the example above, the word “boxes” is an event argument with the role “Object” for the event “Modify”.

We will annotate both event triggers and event arguments. For each trigger, we are interested in knowing what type of event is evoked by that. For the arguments, we want to know the role of the argument in the event. Please note that it is possible that only a few argument roles are mentioned in the document and the rest might be missing.

Figure 4: Annotation Instruction

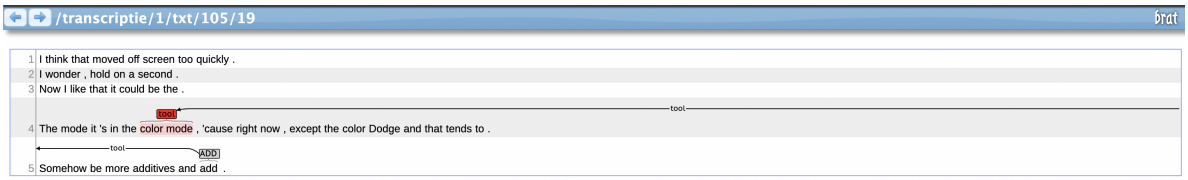

Figure 5: Annotation Tool

ID	Type	Description	Example (triggers are highlighted)
1	Select	A "Select" event happens when an object is selected using one of the selection tools.	• And this time I'm just going to be really, really kind of lazy about it and use my Lasso tool to do some **selections** • Let's **pick** these leaves and do some fun edits on them!
2	Remove	A "Remove" event happens when a part of the image is removed.	• We need to first **get rid of** these lower sections and add the new sketch there. • Okay, I just **deleted** all background shapes to make the image cleaner.
3	Add	An "Add" event happens when a new object is added to the image.	• We're going to be very. Very carefully. **Brushing** alongside. • The birds on the tree are easily **added** by my special brush.
4	Modify	A "Modify" event happens when an object of the image is updated (e.g., resize, recolor, blur, etc.).	• What I'm going to do is to **turn** that ball to blue so it will be matched with whatever we have over there. • I first **brightened** its front side to give more depth to the image.

Table 4: Event types along with their descriptions and examples in the proposed dataset.

ID	Type:Argument	Description	Example (arguments are highlighted and triggers are in *italic* font)
1	Select:Tool	The tool that is utilized to perform the select action.	And this time I'm just going to be really, really kind of lazy about it and use my **Lasso** tool to do some *selections*
2	Select:Object	The object that is selected.	I'm gonna *select* the **leaves** using the magic tool.
3	Remove:Tool	The tool that is utilized to perform the removal action.	Using the **perspective** crop, it's super easy to *get rid of* these buildings.
4	Remove:Object	The object that is being removed.	Using the perspective crop, it's super easy to *get rid of* these **buildings**.
5	Add:Tool	The tool that is utilized to add the new object.	First, let's *add* a single circle here using **ellipse**.
6	Add:Object	The object that is added to the image.	First, let's *add* a single **circle** here using ellipse.
7	Modify:Old_Color	Previous color of the object or image.	We first start with this **blue** sky and *turn* it to dark blue.
8	Modify:New_Color	New color of the object or image.	We first start with this blue sky and *turn* it to **dark blue**.
9	Modify:Old_Size	Old size of the object.	Let's *make* this giant **100-pixel** bar shorter.
10	Modify:New_Size	New size of the object.	The hat overhear should be *enlarged* to **10** cm.
11	Modify:Tool	The tool that is utilized to modify the object.	Use the **color replacement** tool to easily *change* the background color.
12	Modify:Object	The object that is modified.	We first start with this blue **sky** and *turn* it to dark blue.

Table 5: Argument roles for each event type along with their descriptions and examples in the proposed dataset.

Recycle Your Wav2Vec2 Codebook: a Speech Perceiver for Keyword Spotting

Guillermo Cámbara
TALN Research Group
Universitat Pompeu Fabra
Barcelona, Spain
guillermo.cambara@upf.edu

Jordi Luque
Telefónica Research
Telefónica I+D
Barcelona, Spain
jordi.luque@telefonica.com

Mireia Farrús
Centre de Llenguatge i Computació
UBICS, Universitat de Barcelona
Barcelona, Spain
mfarrus@ub.edu

Abstract

Speech information in a pretrained wav2vec2.0 model is usually leveraged through its encoder, which has at least 95M parameters, being not so suitable for small footprint Keyword Spotting. In this work, we show an efficient way of profiting from wav2vec2.0's linguistic knowledge, by recycling the phonetic information encoded in its latent codebook, which has been typically thrown away after pretraining. We do so by transferring the codebook as weights for the latent bottleneck of a Keyword Spotting Perceiver, thus initializing such model with phonetic embeddings already. The Perceiver design relies on cross-attention between these embeddings and input data to generate better representations. Our method delivers accuracy gains compared to random initialization, at no latency costs. Plus, we show that the phonetic embeddings can easily be downsampled with k-means clustering, speeding up inference in 3.5 times at only slight accuracy penalties.

1 Introduction

Keyword Spotting (KWS) has benefited recently from the adoption of the Transformer architecture (Vaswani et al., 2017), as well as from recent advances in self-supervised learning proposals like wav2vec2.0 (Baevski et al., 2020).

Transformers are capable of capturing information from broader contexts, going beyond the locality of convolutional neural networks (CNN) (LeCun et al., 1989) and avoiding the vanishing/exploding gradients from recurrent neural networks (RNN) (Rumelhart et al., 1985). However, this comes at the quadratic cost of the self-attention mechanism (Bahdanau et al., 2014), which is even more pronounced in high-dimensional modalities like speech or vision. KWS models like the Keyword Spotting Transformer (KWT) (Berg et al., 2021) and the Audio Spectrogram Transformer (AST) (Gong et al., 2021), minimize such cost by downsampling the spectrogram space into patches, inspired by the Vision Transformer (ViT) (Dosovitskiy et al., 2020) proposal from computer vision.

In parallel, approaches like Wav2KWS (Seo et al., 2021) or the model from SUPERB (Yang et al., 2021) have successfully applied wav2vec2.0 to KWS. During training, wav2vec2.0 learns a latent codebook that codifies phonetic information, using each code as a target for training a feature encoder. This codebook is typically thrown away after training, only keeping the encoder for downstream tasks like automatic speech recognition (ASR) or KWS. Even though the encoder is capable of extracting rich features from raw waveforms, the size of it (at least 95M parameters for the BASE model) and its added latency time might discourage straightforward use for small footprint KWS classifiers.

In this short paper, we focus on exploring methods for recycling the phonetic information from the wav2vec2.0 latent codebook, showing that such information kickstarts the accuracy of a KWS model at initialization and leads to a better convergence, at virtually no cost in terms of inference time or model size.

The cornerstone of our proposal is a natural synergy that we have spotted between wav2vec2.0 and the recently proposed Perceiver (Jaegle et al., 2021) model. The latter's design relies on cross-attention between input data and a smaller latent bottleneck array, achieving smaller computational costs than pure self-attention over input data. We find that a pretrained wav2vec2.0 latent codebook can be used as an initialization for the Perceiver's latent bottleneck array, which boosts the model's accuracy with respect to random weight initialization. Furthermore, since many vectors from the wav2vec2.0 codebook contain similar phonetic information, we apply k-means clustering and average vectors belonging to the same clusters, yielding downsampled latent bottlenecks that provide faster inference at only slight accuracy costs.

Proceedings of the 29th International Conference on Computational Linguistics, pages 7166–7170
October 12–17, 2022.

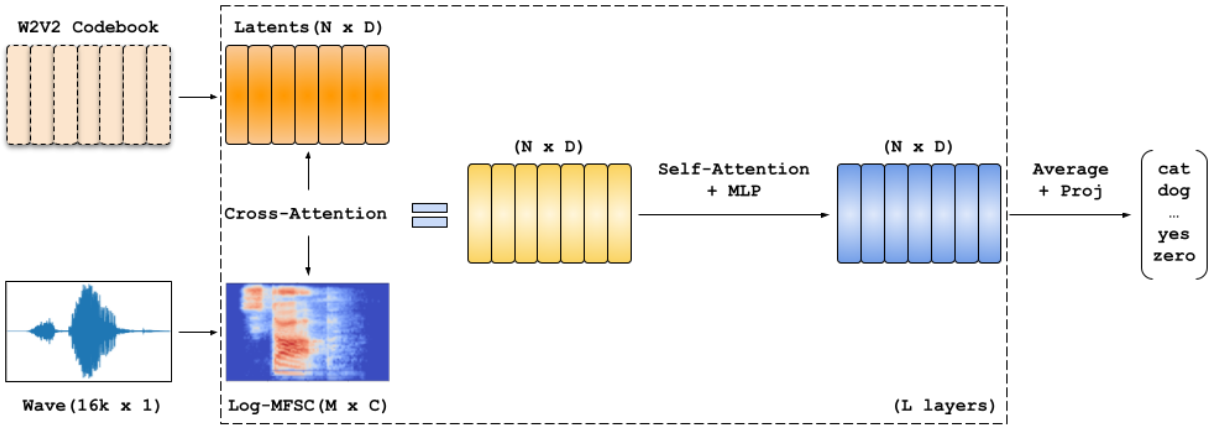

Figure 1: The Keyword Spotting Perceiver (KWP) model.

The focused contribution of this paper provides insight on efficient wav2vec2.0 transfer learning by latent codebook recycling, as well as showing the first application of the Perceiver model to a specific speech task like KWS, up to our knowledge.

2 Keyword Spotting Perceiver

Our Keyword Spotting Perceiver (KWP) is designed to take 1-second waveforms as inputs, converting these into log-mel spectrograms of $M = 100$ time steps and $F = 64$ frequency bins, which are linearly projected to a dimension of $C = 192$, resulting in a MxC data array. Fourier positional encodings are concatenated to the data array along the C dimension, using 64 frequency bands and a maximum resolution of 224, the best performing choice in the Perceiver paper. Cross-attention between such data array and a latent bottleneck array of NxD dimensions is done with a single head, whose output is further refined through a Transformer block containing self-attention with 8 heads and a multilayer perceptron (MLP) of hidden size 1024. The dimension for both self and cross attention heads is set to 64. Since the output is another latent array of NxD dimensions, we repeat cross-attention with the data array and the Transformer blocks for $L = 6$ layers, sharing the weights of cross-attends and Transformer across layers, in the style of a RNN. In earlier explorations we tried not sharing the weights but this led to performance degradation caused by overfitting. Finally, we average the latents in the D dimension, apply layer normalization and do a linear projection to get the class logits for prediction. A model depiction can be seen in Figure 1.

The latent array can be initialized randomly (KWP-BASE), or by transferring the weights from the latent codebook of a pretrained wav2vec2.0 model (KWP-W2V2). In this work, we recycle the latent weights of the wav2vec2.0 BASE model from the HuggingFace repo [1]. Such codebook consists of $N = 640$ vectors of dimension $D = 128$.

Since the complexity of cross-attention between latent and data arrays is $O(MN)$, we lose the efficiency gains from it with respect to self-attention over data array $O(N^2)$, since $O(MN) = O(100x640) = (6.4x10^4) > O(N^2) = O(100^2) = (10^4)$. To address this, we study three ways to downsample this latent space to lower dimensions $N = [320, 160, 80, 40, 20]$, by (1) sampling vectors randomly, (2) average pooling contiguous vectors and (3) clustering with k-means method and averaging vectors from the same cluster. According to the wav2vec2.0 paper, most of the codebook latents model specific English phonemes, being some phonemes represented by many latents. For instance, the silence phoneme is represented by 22% of the codebook. Being so, we expect k-means clustering to be the best downsampling method from the proposed ones, by clustering latents representing same or similar phonemes. Simple average pooling without k-means clustering might conserve phonetic information, although we expect it to be suboptimal given that we cannot guarantee that contiguous vectors in the codebook correspond to similar phonetics, thus potentially mixing information from different phonemes. Oppositely, random sampling guarantees that the individual information of each vector in the codebook is preserved, but as N gets lower many information is potentially lost, since most vectors are being dropped out.

[1] https://huggingface.co/facebook/wav2vec2-base

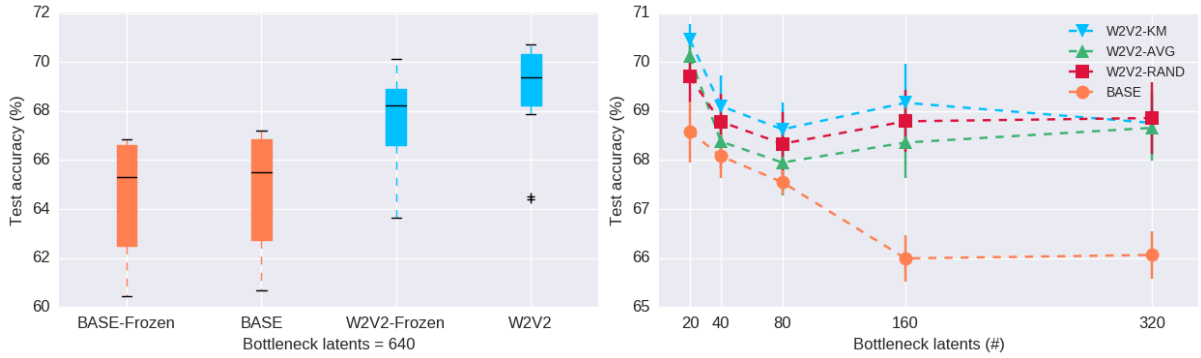

Figure 2: Test accuracy after a single training epoch, for a randomly initialized model (BASE), and another initialized with a wav2vec2.0 latent codebook weights (W2V2), with learnable or frozen weights (left). We also report the results of downsampling the bottleneck latents with k-means clustering (KM), averaging pooling (AVG) and random sampling (RAND) (right).

3 Experiments

We describe here the evaluations made for our Keyword Spotting Perceiver proposal. First, we assess the effects of transferring the wav2vec2.0 latent codebook to the Perceiver bottleneck at initialization. We also try different ways of downsampling this latent space, reporting accuracy comparisons between the baseline KWP-BASE model and the wav2vec2.0-initialized KWP-W2V2 model. Afterwards, we keep the best performing downsampling algorithm for the following round of experiments, where we let the system train until convergence. We report accuracy, model size and inference time metrics for KWP-BASE and KWP-W2V2 models with different latent number variants $N = [320, 160, 80, 40, 20]$.

Training, validation and testing phases are done with the standard partitions from the Google Speech Commands V2 dataset (Warden, 2018), obtaining the accuracy metrics from the 35-commands task. All timing metrics are obtained by doing inference over 1-second waveforms on CPU, warming up for 10 forward passes and averaging the time for 150 forward passes. We use the AdamW optimizer (Loshchilov and Hutter, 2018) with a step learning rate scheduler, decreasing the learning each epoch by a gamma factor of 0.98, starting with an initial learning rate of $1e^{-4}$. Batch size is set to 32, training for a single epoch in the initialization experiments and for 400 epochs in the convergence experiments. For the latter ones, we pick the top-10 checkpoints with the highest validation accuracy, averaging their weights to obtain the final checkpoint, which we use for test accuracy measurements. We open-source the PyTorch code [2] used for our experiments to the community.

Regarding data augmentation, we apply time shifting of ± 0.1 seconds with a probability of 60%. We also do resampling of the waveform signal between a $[0.85, 1.15]$ fraction of the input sampling rate, which is set to 16 kHz, with a probability of 100%. Background noise is also added in a range of $[5.0, 30.0]$ dBs and SpecAugment (Park et al., 2019) is done with 2 time masks of 25 frame size and 2 frequency masks of 7 frames each. The latter both data augmentation methods are also applied with a 100% probability during training. Even though, we relax the augmentation conditions for the shorter initialization experiments of a single epoch, to let the system learn a bit more in the early stages. Time shifting and resampling probabilities are lowered to a 30%, SpecAugment to a 70% and background noise addition to an 80%.

3.1 Initialization with Wav2vec2.0 Codebook

We check the impact of transferring the wav2vec2.0 codebook to KWP at initialization, by measuring the test accuracy after a single epoch, repeating training and test with 10 different seeds. Thus, we compare between KWP-BASE and KWP-W2V2 with all the $N = 640$ latent vectors, by making the latent bottleneck weights learnable (BASE and W2V2) and also freezing them (BASE-Frozen and W2V2-Frozen).

Figure 2(a) shows that both W2V2 and W2V2-Frozen have a significant performance advantage against BASE and BASE-Frozen. This suggests that the phonetic information transferred from the

[2] https://github.com/gcambara/speech-commands

wav2vec2.0 latent codebook kickstarts training, initializing the model already with useful information that the cross-attention mechanism can leverage. Furthermore, it is interesting to see how there is barely no performance differences between BASE and BASE-Frozen. We hypothesize that during the first epoch the randomly initialized BASE model has not learned enough phonetic information in its latent bottleneck, giving fewer chances for cross-attention to exploit relations with input data. The W2V2 model, oppositely, is able to leverage cross-attention early on, generating feedback between the phonetic information in the latent codebook and the cross-attention weights that are linked to input data.

To continue with, we repeat the same 10-seed experiment for $N = [320, 160, 80, 40, 20]$ latent vectors in the bottleneck. Different downsampling methods are tried: k-means clustering (W2V2-KM), average pooling (W2V2-AVG) and random sampling (W2V2-RAND). The results, as seen in Figure 2(b), highlight that the three latent downsampling methods are effective for boosting the performance with respect to the BASE model. W2V2-KM is the best performing model, confirming that averaging latents belonging to the same phonetic clusters is preferable, rather than simply averaging contiguous latents like in W2V2-AVG, or randomly sampling latent vectors like W2V2-RAND does, which loses representation power as N decreases.

3.2 Assessment at Convergence

To evaluate the accuracy of KWP-BASE and KWP-W2V2 models after convergence, we let the models train again for 400 epochs. This time, we only experiment with learnable latent weights and k-means clustering downsampling, given that the latter has reported the best initialization results. Training and test is done now with 3 seeds, varying the number of latents again with the same selection, $N = [640, 320, 160, 80, 40, 20]$, and comparing between BASE and W2V2 variants.

As Figure 3 depicts, W2V2 maintains significant advantage for all the numbers of latents, with a peak mean accuracy of $96.26 \pm 0.04\%$ at 640 latents, higher than BASE's top accuracy of $95.6 \pm 0.2\%$ at 80 latents. The W2V2 variant seems to scale well with the number of latents, as opposite to the BASE model, which might struggle to clusterize phonetic information in the latent space as it grows bigger. Still, KWP (1.5M parameters) is slightly behind to its self-attention counterparts, as the lightest KWT (0.6M parameters) scored a 96.8% accuracy, and AST scored 98.1%. Nevertheless, note that AST is pretrained with ImageNet (Deng et al., 2009), and is much larger (87M parameters). Even though, we motivate further research on fine-tuning KWP towards state-of-the-art performance.

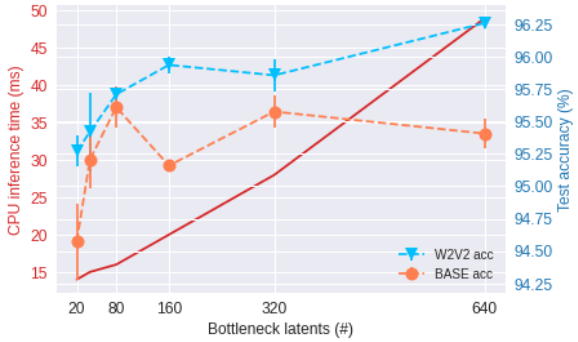

Figure 3: KWP-BASE (orange) and KWP-W2V2 (blue) test accuracies after convergence, for different numbers of bottleneck latents, with CPU inference time (red).

The inference time of the 640 latent model is 49 ± 5 ms, and 14 ± 2 ms for the smaller 20 latent model. Given that the accuracy is $95.3 \pm 0.1\%$ for the latter, only a 1% relative accuracy is lost with k-mean clustering downsampling, while increasing inference speed in 3.5 times. The accuracy of the BASE model at 20 latents is $94.6 \pm 0.3\%$, which is significantly below W2V2's. This confirms that even a hard downsampling of 640 to 20 latents of wav2vec2.0 information is still preferable to randomly initializing the latent space in KWP.

4 Conclusion

In this work, we have shown that phonetic information from the wav2vec2.0 latent codebook can be recycled, by transferring it to the latent bottleneck weights of a Keyword Spotting Perceiver. Accuracy gains are consistently significant with respect to random initialization of the latent bottleneck, both at early and late stages of training for the KWS task. Furthermore, we have studied easy-to-apply downsampling techniques for compressing the latent codebook, like averaging k-means clusters, having sped up the inference time of the model up to 3.5 times, at only a 1% accuracy drop.

We believe that our work motivates further research on efficient ways of profiting from the information in big self-supervised models like wav2vec2.0, as well as on applications for other tasks like ASR, for instance.

Acknowledgements

This work is a part of the INGENIOUS project, funded by the European Union's Horizon 2020 Research and Innovation Programme and the Korean Government under Grant Agreement No 833435.

References

Alexei Baevski, Yuhao Zhou, Abdelrahman Mohamed, and Michael Auli. 2020. wav2vec 2.0: A framework for self-supervised learning of speech representations. *Advances in Neural Information Processing Systems*, 33:12449–12460.

Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. 2014. Neural machine translation by jointly learning to align and translate. *arXiv preprint arXiv:1409.0473*.

Axel Berg, Mark O'Connor, and Miguel Tairum Cruz. 2021. Keyword transformer: A self-attention model for keyword spotting. *arXiv preprint arXiv:2104.00769*.

Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. 2009. Imagenet: A large-scale hierarchical image database. In *2009 IEEE conference on computer vision and pattern recognition*, pages 248–255. Ieee.

Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, et al. 2020. An image is worth 16x16 words: Transformers for image recognition at scale. *arXiv preprint arXiv:2010.11929*.

Yuan Gong, Yu-An Chung, and James Glass. 2021. Ast: Audio spectrogram transformer. *arXiv preprint arXiv:2104.01778*.

Andrew Jaegle, Felix Gimeno, Andy Brock, Oriol Vinyals, Andrew Zisserman, and Joao Carreira. 2021. Perceiver: General perception with iterative attention. In *International Conference on Machine Learning*, pages 4651–4664. PMLR.

Yann LeCun, Bernhard Boser, John S Denker, Donnie Henderson, Richard E Howard, Wayne Hubbard, and Lawrence D Jackel. 1989. Backpropagation applied to handwritten zip code recognition. *Neural computation*, 1(4):541–551.

Ilya Loshchilov and Frank Hutter. 2018. Decoupled weight decay regularization. In *International Conference on Learning Representations*.

Daniel S Park, William Chan, Yu Zhang, Chung-Cheng Chiu, Barret Zoph, Ekin D Cubuk, and Quoc V Le. 2019. Specaugment: A simple data augmentation method for automatic speech recognition. *Proc. Interspeech 2019*, pages 2613–2617.

David E Rumelhart, Geoffrey E Hinton, and Ronald J Williams. 1985. Learning internal representations by error propagation. Technical report, California Univ San Diego La Jolla Inst for Cognitive Science.

Deokjin Seo, Heung-Seon Oh, and Yuchul Jung. 2021. Wav2kws: Transfer learning from speech representations for keyword spotting. *IEEE Access*, 9:80682–80691.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. *Advances in neural information processing systems*, 30.

Pete Warden. 2018. Speech commands: A dataset for limited-vocabulary speech recognition. *arXiv preprint arXiv:1804.03209*.

Shu-wen Yang, Po-Han Chi, Yung-Sung Chuang, Cheng-I Jeff Lai, Kushal Lakhotia, Yist Y Lin, Andy T Liu, Jiatong Shi, Xuankai Chang, Guan-Ting Lin, et al. 2021. Superb: Speech processing universal performance benchmark. *arXiv preprint arXiv:2105.01051*.

Improving code-switched ASR with linguistic information

Jie Chi and **Peter Bell**
Centre for Speech Technology Research, University of Edinburgh, UK

Abstract

This paper seeks to improve the performance of automatic speech recognition (ASR) systems operating on code-switched speech. Code-switching refers to the alternation of languages within a conversation, a phenomenon that is of increasing importance considering the rapid rise in the number of bilingual speakers in the world. It is particularly challenging for ASR owing to the relative scarcity of code-switching speech and text data, even when the individual languages are themselves well-resourced. This paper proposes to overcome this challenge by applying linguistic theories in order to generate more realistic code-switching text, necessary for language modelling in ASR. Working with English-Spanish code-switching, we find that Equivalence Constraint theory and part-of-speech labelling are particularly helpful for text generation, and bring 2% improvement to ASR performance.

1 Introduction

Although accurate statistics on the number of worldwide bilingual speakers are hard to determine, it has been generally believed that more than half of the population can communicate in more than one language (Ansaldo et al., 2008; Bialystok et al., 2012; Grosjean, 2010). With the rising popularity of voice assistant and translation applications, automatic speech recognition (ASR) systems have been increasingly integrated into people's lives. Considering the abundance of bilingual countries [1], and code-switching [2] is common in everyday conversations, there has been great interest into developing such system in the corresponding setting.

The most widely accepted definition of code-switching is the phenomenon whereby a speaker shifts from one language to another within a single utterance, especially in an environment in which both languages are being used (Heredia and Altarriba, 2001). Some previous work make a distinction between *code-switching* and *code-mixing*, where the former occurs at sentence-level while the latter occurs at word-level (Myers-Scotton, 1997; Gumperz, 1982). However, in recent years, this distinction has becomes unclear (Bali et al., 2014). To avoid confusion, we will only use the term code-switching in this work, and denote the differences in switching position as sub-types of it (Myers-Scotton, 1989; Muysken et al., 2000; Major, 2002; Winford, 2003). Although different language pairs may present varying extents or types of code-switching, they can generally be categorised as inter-sentential, intra-sentential, and tag switching, where respectively the language switches at sentence or clause boundary; within the sentence or clause; or by inserting a tag phrase [3]. In this paper, we focus only intra-sentential switching, which is a much harder task compared with other types, because the acoustic variance of mixed languages within the sentence can be larger than across sentence (Li et al., 2019).

The challenge of developing a code-switched ASR system comes from both linguistic and computational perspectives. On the one hand, code-switching can be driven by multiple factors, which makes it flexible but hard to predict. Linguists have studied the phenomenon for decades, and the main views held are that people tend to code-switch to compensate for lack of language proficiency, express solidarity or certain feelings, discuss some particular topic, and distinguish themselves from other classes to imply a certain social status (Grosjean, 1982; Holmes and Wilson, 2017; Leung, 2006). ASR systems generally require a

[1] https://www.uottawa.ca/clmc/55-bilingual-countries-world

[2] Code-switching can also be found in text, such as social media or news paper headlines, but in this paper we are only focus only on the spoken form

[3] Intra-word switching can be arguably considered as a type of code-switching (Myers-Scotton, 1989), but a complete discussion is out of the scope of this paper.

Proceedings of the 29th International Conference on Computational Linguistics, pages 7171–7176
October 12–17, 2022.

large amount of transcribed data in a monolingual setting. However, a relatively small number of the approximately 7000 existing languages have large readily corpora, and the data scarcity is more severe for code-switching problems, which – to make matters worse – usually involve one or more low-resourced languages (Austin and Sallabank, 2011).

Motivated by the these considerations, we propose a novel code-switched ASR framework with the aid of established linguistic theories. We demonstrate the effectiveness of the approach on Spanish-English conversational code-switching data from the Bangor-Miami corpus (Deuchar, 2011). In doing so, we prove both phonological and syntactic information can improve the performance of language modelling and ASR.

2 Related work

There have been many attempts to approach the problem of modelling code-switched speech from acoustic, pronunciation and language modeling perspectives for conventional hybrid ASR systems. In early work a language identification (LID) system was combined to determine which hypothesis from multiple monolingual decoders should be chosen (Lyu et al., 2006; Wu et al., 2006; Bhuvanagiri and Kopparapu, 2010). However, the drawback of this approach is a strong assumption that the speech segments are independent from each other and a heavy dependency on the accuracy of LID module (Weiner et al., 2012). The choice of phone inventories is important, and many studies have been conducted to merge phone sets of different language pairs manually or automatically, which improves the performance of ASR systems by effectively yielding more training data for each phone, and enabling the pronunciation influence between languages to be learnt (Kohler, 1998; Lyudovyk and Pylypenko, 2014; Sivasankaran et al., 2018). Considering that there is a much larger amount of monolingual text than code-switched text, code-switched text generation by imposing language theories to parallel monolingual texts has been a popular research direction (Li and Fung, 2014; Winata et al., 2019; Pratapa et al., 2018). From this, language models can be improved by training them on the generated text or can also be achieved by apply the theories directly to restrict the search paths in a WFST framework(Li and Fung, 2013).

In recent years, end-to-end (E2E) models have also been explored to handle the problem. To address the issue that E2E models usually require a large amount of data, transfer learning from monolingual models has been used (Luo et al., 2018; Shan et al., 2019; Mary N J et al., 2020; Winata et al., 2020). However, when the models are fine-tuned on code-switched data, the performance on monolingual speech is degraded. To improve the robustness of the model, techniques such as Learning Without Forgetting and adversarial training have been proposed (Shah et al., 2020; Madhumani et al., 2020).

3 Methodology

3.1 Phoneme mapping

We use the standard International Phonetic Alphabet (IPA) as the basis for our acoustic modelling units. As English and Spanish have partly different inventories (Edwards, 1992; Goldstein, 2000), instead of treating them as completely two phone sets, we merge them according to their phonological features (Mortensen et al., 2016). The features are a set of global attributes, which describes the characteristic of sound, such as whether it is produced with nasal airflow. After representing the feature with vectors, where each attribute can be negative or positive, we compute the hamming edit distance between each pair of phonemes. In this way, we map each Spanish phoneme to its nearest English equivalent.

3.2 CS text generation

3.2.1 Parallel text generation

We use the Google translate API to generate parallel English and Spanish text. The API is not only capable of translating from one to the other, but also can translate code-switched sentences to one of the language while keeping the segment in the target language unchanged. We receive one translated sentence for each monolingual text in the corpus, while for each code-switched sentence, we obtain translations in both languages. As the translation quality varies across the sentences under manual inspection, we use Pseudo Fuzzy-match Score (PFS) shown in Equation 1 to filter any translation pairs whose PFS is less than 0.6 (He et al., 2010; Pratapa et al., 2018). s here is the monolingual source sentence, we forward translate s to target t, then reverse translate the target t into pseudo source s'.

$$PFS = \frac{EditDistance(s, s')}{max(|s|, |s'|)} \quad (1)$$

3.2.2 Constituency parse generation

To generate word level alignments between parallel sentences we use *fast_align* toolkit, which is a unsupervised aligner (Dyer et al., 2013). Then following (Pratapa et al., 2018), we generate the parse tree for English text with Berkeley neural parser (Kitaev and Klein, 2018) and then use the alignments to generate the equivalent parse for the Spanish sentence.

3.2.3 Equivalence Constraint theory

The Equivalence Constraint (EC) theory claims that, in general, *"Codes will switch at points where the surface structures of the languages map onto each other"* (Sankoff and Poplack, 1981). For example, in English and Spanish, code-switching cannot happen within possessive phrases or noun/adjective clauses because the grammatical structures are different and thus reject the transfer.

EC has been successfully applied to code-switched text data and approved crucial for language modelling task (Pratapa et al., 2018), which is our implementation inspired from. However, it should be noted that even though text data can be in the form of informal conversation on Twitter or other Internet platforms, they may still not follow the same patterns as speech (Sitaram et al., 2019).

We apply EC to generate CS text over permutation. To improve the naturalness, we rank generated texts for each pair by following metrics and select at most top 10 for each sentence pair.

Switch-points are points within a sentence where the languages of the words on the two sides are different (Pratapa et al., 2018). The metric SP Fraction (SPF) is defined as the number of SP in a sentence divided by the total number of word boundaries in the sentence. We set it to 0.1 for all experiments.

CMI counts the number of switches in the utterance (Gambäck and Das). It can be computed at the utterance level by finding the most frequent language in the utterance and then counting the frequency of the words belonging to all other languages present. The computation is shown in Equation 2, where W denotes the utterance and $N(W)$ denotes the number of words in W. l_max means the count of words in the most frequent language in that utterance, and $P(W)$ is the number of SP. We set it to 0.3 for all experiments.

$$CMI(W) = \frac{N(W) - l\_max + P(W)}{N(W)} \quad (2)$$

POS tags As we have translate code-switched text to parallel sentences, with monolingual words remained, we can find the POS tags of the switched words in both languages. Therefore, by calculating the frequency of the POS tags of the preceding/current/following words, we give the sentence higher ranking if proper nouns, nouns, determiners or interjections precede switched words, and nouns or subordinating conjunctions are switched, which is consistent with (Soto et al., 2018)

4 Experimental setup

4.1 Data

Although the corpus is public, there has been no standard preprocessing procedures for it. In this paper, we first classified all utterances into Spanish, English and Code-switched sets. If there are both English and Spanish exclusive words in the utterance, we consider it as code-switched case, but if there are words which exist both in the English and Spanish lexicon, the category depends on the rest of the sentence [4]. After cleaning, we retain 44971 utterances, splitting them into training, development and test by 7:1:2. To better illustrate the distribution of the dataset, the statistics are presented in Table 1.

4.2 Training

4.2.1 Acoustic models

We used the Kaldi TDNN recipe [5] to develop the hybrid systems. 40 dimensional MFCC features are extracted first to train a GMM-HMM model. Before training the neural network, we apply speed perturbation to augment data. The TDNN-HMM training is identical for all systems: We use 40 dimensional high resolution MFCC with 100 dimensional i-vector features as input. The network consists of 7 TDNN hidden layers which contain 758 hidden units per layer. The start and final learning rates are 0.00015 and 0.000015 respectively and we train the model for 4 epochs with a mini-batch size of 128.

4.2.2 Language models

We use SRILM toolkit to train n-gram models for comparison. For each setting, we trained a 3-gram model for decoding and a 4-gram for rescoring. The lexicon is identical to all experiments. We

[4] The processing script will be released after acceptance.

[5] https://github.com/kaldi-asr/kaldi/blob/master/egs/librispeech/s5

Table 1: The statistics of the processed Miami corpus, where the duration unit is hour.

	English		Spanish		CS	
	Number	Duration	Number	Duration	Number	Duration
Training	20813	10.9	7789	4.3	1879	1.6
Dev	3000	1.6	1250	0.6	250	0.2
Test	6000	3.1	2500	1.2	500	0.5

Table 2: WER and PPL on testset, where the top block shares the same language model which is trained only on the original transcript, and the bottom block shares the acoustic model with phoneme mapping.

	Test WER				Test PPL		
	English	Spanish	CS	total	English	Spanish	CS
baseline	44.0	56.8	49.3	47.8	109.7	144.8	152.8
mapping base	43.6	56.4	49.1	47.5	109.7	144.8	152.8
translation	43.4	56.0	49.0	47.2	90.3	126.4	134.9
+ external	43.4	56.0	49.0	47.3	92.3	128.4	149.6
+ random	43.4	55.9	49.1	47.2	89.1	127.8	145.2
+ POS	43.3	55.6	48.7	47.0	88.7	126.0	130.1
+ EC	43.3	55.6	48.6	47.0	87.2	122.8	125.7
+ EC + POS	43.2	55.3	48.4	46.9	87.1	120.2	123.8

used the CMUDict for English and Santiago Spanish Lexicon for Spanish. There are in total 206500 English words and 91121 Spanish words, any uncovered words are treated as UNK.

5 Results and discussion

Table 2 presents the word error rate (WER) and perplexity (PPL) on the test set for all experiments. Our *baseline* model uses the union of English and Spanish phoneme sets while *mapping base* maps the Spanish phoneme set to English phoneme set. Their language models are identical which are trained only on the transcripts of Bangor-Miami corpus. It can be observed from the result that phoneme mapping can improve the performance by 0.3 absolute WER, so we fix it as our acoustic model and the only difference among the experiments on the bottom block is that they have different synthetic texts for language modelling. *translation* model denotes that the language model is trained on the transcripts as well as the translation of them and + symbol describes with what techniques, code-switched text have been generated and added to the training text. We interpolate the LM with models trained on external text data [6] to show that simply using larger but out-of-domain text data doesn't help improve the performance. After obtaining the word alignments of parallel sentences, we compare the results of generating the code-switched text by simply random replacement of the aligned words, or ranking the possible replacements with the POS tags of the preceding/current/following words or the acceptance under EC theory.

We can find that POS and EC have similar improvement on WER, while the combination of them shows their advantages cannot directly add up. One possible explanation can be that the implementation of EC is based on the constituency parse, which is also heavily related to the POS tags. Therefore, the linguistic information implied by them are overlapped with each other and only little improvement can be achieved when combined. Our model with the best performance uses all of linguistic information we discussed before, with approximately 2% improvement on both WER and PPL.

6 Conclusions

In this paper, we present a framework for code-switched ASR task. By using phonological features for phoneme mapping, and POS tags and EC theory for more more natural code-switched text generation, we eventually achieve 2% improvement on PPL as well as WER. It should be noted that although the experiments are conducted on Bangor-Miami corpus, there are no language specific constraints with this approach, which shows

[6] Here, we use TED talk subtitles to train the LMs for English and Spanish (Tiedemann, 2012).

a potential to be extended to cover more language pairs. As a future work, we would like to compare the performances of different linguistic theories in our proposed framework, which can serve as an indirect validation of their influence on different language pairs. Also, motivated by natural distributions of linguistic structures, exploring different sampling techniques can also be promising.

7 Acknowledgements

This work was supported in part by the UKRI Centre for Doctoral Training in Natural Language Processing, funded by the UKRI (grant EP/S022481/1) and the University of Edinburgh.

References

Ana Inés Ansaldo, Karine Marcotte, Lilian Scherer, and Gaelle Raboyeau. 2008. Language therapy and bilingual aphasia: Clinical implications of psycholinguistic and neuroimaging research. *Journal of Neurolinguistics*, 21(6):539–557.

Peter K Austin and Julia Sallabank. 2011. *The Cambridge handbook of endangered languages*. Cambridge University Press.

Kalika Bali, Jatin Sharma, Monojit Choudhury, and Yogarshi Vyas. 2014. "I am borrowing ya mixing ?" an analysis of English-Hindi code mixing in Facebook. In *Proceedings of the First Workshop on Computational Approaches to Code Switching*, pages 116–126, Doha, Qatar. Association for Computational Linguistics.

K Bhuvanagiri and Sunil Kopparapu. 2010. An approach to mixed language automatic speech recognition. *Oriental COCOSDA, Kathmandu, Nepal*.

Ellen Bialystok, Fergus Craik, and Gigi Luk. 2012. Bilingualism: Consequences for mind and brain. *Trends in cognitive sciences*, 16:240–50.

Margaret Deuchar. 2011. The miami corpus: Documentation file.

Chris Dyer, Victor Chahuneau, and Noah A. Smith. 2013. A simple, fast, and effective reparameterization of ibm model 2. In *NAACL*.

Harold T. Edwards. 1992. *Applied Phonetics: The Sounds of American English*. Singular Publishing Group.

Björn Gambäck and Amitava Das. On Measuring the Complexity of Code-Mixing.

Brian Goldstein. 2000. *Resource guide on cultural and linguistic diversity*. Singular resource guide series. Singular Pub. Group, San Diego, Calif.

F Grosjean. 2010. *Bilingual: life and reality*. Harvard University Press.

François Grosjean. 1982. *Life with two languages: An introduction to bilingualism*. Harvard University Press.

John J. Gumperz. 1982. *Discourse Strategies*. Studies in Interactional Sociolinguistics. Cambridge University Press.

Yifan He, Yanjun Ma, Andy Way, and Josef Genabith. 2010. Integrating n-best smt outputs into a tm system. *He, Yifan and Ma, Yanjun and Way, Andy and van Genabith, Josef (2010) Integrating N-best SMT outputs into a TM system. In: COLING 2010 - 23rd International Conference on Computational Linguistics, 23-27 August 2010, Beijing, China.*

Roberto Heredia and Jeanette Altarriba. 2001. Bilingual language mixing: Why do bilinguals code-switch? *Current Directions in Psychological Science - CURR DIRECTIONS PSYCHOL SCI*, 10:164–168.

Janet Holmes and Nick Wilson. 2017. *An Introduction to Sociolinguistics*, 5 edition. Routledge, London.

Nikita Kitaev and Dan Klein. 2018. Multilingual constituency parsing with self-attention and pre-training. *CoRR*, abs/1812.11760.

J. Kohler. 1998. Language adaptation of multilingual phone models for vocabulary independent speech recognition tasks. In *Proceedings of the 1998 IEEE International Conference on Acoustics, Speech and Signal Processing, ICASSP '98 (Cat. No.98CH36181)*, volume 1, pages 417–420 vol.1.

Carrie Leung. 2006. Codeswitching in print advertisements in hong kong and sweden.

Ke Li, Jinyu Li, Guoli Ye, Rui Zhao, and Yifan Gong. 2019. Towards Code-switching ASR for End-to-end CTC Models. In *ICASSP 2019 - 2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 6076–6080. ISSN: 2379-190X.

Ying Li and Pascale Fung. 2013. Improved mixed language speech recognition using asymmetric acoustic model and language model with code-switch inversion constraints. In *2013 IEEE International Conference on Acoustics, Speech and Signal Processing*, pages 7368–7372.

Ying Li and Pascale Fung. 2014. Code switch language modeling with functional head constraint. In *2014 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 4913–4917.

Ne Luo, Dongwei Jiang, Shuaijiang Zhao, Caixia Gong, Wei Zou, and Xiangang Li. 2018. Towards end-to-end code-switching speech recognition. *CoRR*, abs/1810.13091.

Dau-Cheng Lyu, Ren-yuan Lyu, Yuang-chin Chiang, and Chun-nan Hsu. 2006. Speech recognition on code-switching among the chinese dialects. In *2006 IEEE International Conference on Acoustics Speech and Signal Processing Proceedings*, volume 1, pages I–I.

Tetyana Lyudovyk and Valeriy Pylypenko. 2014. Code-switching speech recognition for closely related languages. In *Spoken Language Technologies for Under-Resourced Languages*.

Gurunath Reddy Madhumani, Sanket Shah, Basil Abraham, Vikas Joshi, and Sunayana Sitaram. 2020. Learning not to discriminate: Task agnostic learning for improving monolingual and code-switched speech recognition.

Roy C. Major. 2002. The bilingualism reader. li wei (ed.). london: Routledge, 2000. *Studies in Second Language Acquisition*, 24(3):491–493.

Metilda Sagaya Mary N J, Vishwas M. Shetty, and S. Umesh. 2020. Investigation of methods to improve the recognition performance of tamil-english code-switched data in transformer framework. In *ICASSP 2020 - 2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 7889–7893.

David R. Mortensen, Patrick Littell, Akash Bharadwaj, Kartik Goyal, Chris Dyer, and Lori Levin. 2016. PanPhon: A resource for mapping IPA segments to articulatory feature vectors. In *Proceedings of COLING 2016, the 26th International Conference on Computational Linguistics: Technical Papers*, pages 3475–3484, Osaka, Japan. The COLING 2016 Organizing Committee.

Pieter Muysken, Pieter Cornelis Muysken, et al. 2000. *Bilingual speech: A typology of code-mixing*. Cambridge University Press.

C. Myers-Scotton. 1997. *Duelling Languages: Grammatical Structure in Codeswitching*. Clarendon Press.

Carol Myers-Scotton. 1989. Codeswitching with english: types of switching, types of communities. *World Englishes*, 8(3):333–346.

Adithya Pratapa, Gayatri Bhat, Monojit Choudhury, Sunayana Sitaram, Sandipan Dandapat, and Kalika Bali. 2018. Language Modeling for Code-Mixing: The Role of Linguistic Theory based Synthetic Data. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1543–1553, Melbourne, Australia. Association for Computational Linguistics.

David Sankoff and Shana Poplack. 1981. A formal grammar for code-switching. *Paper in Linguistics*, 14(1):3–45.

Sanket Shah, Basil Abraham, Gurunath Reddy M, Sunayana Sitaram, and Vikas Joshi. 2020. Learning to recognize code-switched speech without forgetting monolingual speech recognition.

Changhao Shan, Chao Weng, Guangsen Wang, Dan Su, Min Luo, Dong Yu, and Lei Xie. 2019. Investigating end-to-end speech recognition for mandarin-english code-switching. In *ICASSP 2019 - 2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 6056–6060.

Sunayana Sitaram, Khyathi Raghavi Chandu, Sai Krishna Rallabandi, and Alan W. Black. 2019. A survey of code-switched speech and language processing. *CoRR*, abs/1904.00784.

Sunit Sivasankaran, Brij Mohan Lal Srivastava, Sunayana Sitaram, Kalika Bali, and Monojit Choudhury. 2018. Phone merging for code-switched speech recognition. In *Proceedings of the Third Workshop on Computational Approaches to Linguistic Code-Switching*, pages 11–19, Melbourne, Australia. Association for Computational Linguistics.

Victor Soto, Nishmar Cestero, and Julia Hirschberg. 2018. The Role of Cognate Words, POS Tags and Entrainment in Code-Switching. In *Interspeech 2018*, pages 1938–1942. ISCA.

Jörg Tiedemann. 2012. Parallel data, tools and interfaces in opus. In *Proceedings of the Eight International Conference on Language Resources and Evaluation (LREC'12)*, Istanbul, Turkey. European Language Resources Association (ELRA).

Jochen Weiner, Ngoc Thang Vu, Dominic Telaar, Florian Metze, Tanja Schultz, Dau-Cheng Lyu, Eng-Siong Chng, and Haizhou Li. 2012. Integration of language identification into a recognition system for spoken conversations containing code-switches. In *Spoken Language Technologies for Under-Resourced Languages*.

Genta Indra Winata, Samuel Cahyawijaya, Zhaojiang Lin, Zihan Liu, Peng Xu, and Pascale Fung. 2020. Meta-transfer learning for code-switched speech recognition. *CoRR*, abs/2004.14228.

Genta Indra Winata, Andrea Madotto, Chien-Sheng Wu, and Pascale Fung. 2019. Code-switched language models using neural based synthetic data from parallel sentences. *CoRR*, abs/1909.08582.

Donald Winford. 2003. *An introduction to contact linguistics / Donald Winford.* Language in society (Oxford, England) ; v. 33. Blackwell Pub.

Chung Hsien Wu, Yu Hsien Chiu, Chi Jiun Shia, and Chun Yu Lin. 2006. Automatic segmentation and identification of mixed-language speech using delta-bic and lsa-based gmms. *IEEE Transactions on Speech and Audio Processing*, 14(1):266–275.

Language-specific Effects on Automatic Speech Recognition Errors for World Englishes

June Choe, Yiran Chen, May Pik Yu Chan, Aini Li, Xin Gao, Nicole Holliday
Department of Linguistics, University of Pennsylvania, USA
{yjchoe, cheny39, pikyu, liaini, kauhsin, nholl}@sas.upenn.edu

Abstract

Despite recent advancements in automated speech recognition (ASR) technologies, reports of unequal performance across speakers of different demographic groups abound. At the same time, the focus on performance metrics such as the Word Error Rate (WER) in prior studies limit the specificity and scope of recommendations that can be offered for system engineering to overcome these challenges. The current study bridges this gap by investigating the performance of Otter's automatic captioning system on native and non-native English speakers of different language background through a linguistic analysis of segment-level errors. By examining language-specific error profiles for vowels and consonants motivated by linguistic theory, we find that certain categories of errors can be predicted from the phonological structure of a speaker's native language.

1 Introduction

A central concern in the ethics of building natural language processing (NLP) tools is ensuring equity in service and representation through a commitment to linguistic justice (e.g., Blodgett et al., 2020; Hovy and Prabhumoye, 2021). This issue is especially pertinent to automatic speech recognition (ASR) systems used to transcribe natural spoken language into text (Markl and McNulty, 2022). As ASR systems are increasingly being adopted into many aspects of social life (e.g., virtual assistants and automatic captioning), various concerns about the equity of ASR systems have been raised. Ideally, these systems should serve speakers of different demographic backgrounds equally well; however, the existing technologies are not entirely satisfactory. ASR performance has been found to vary by users' dialect (Wheatley and Picone, 1991; Meyer et al., 2020), gender (Adda-Decker and Lamel, 2005; Tatman and Kasten, 2017; Tatman, 2017; Sawalha and Abushariah, 2013; Boito et al., 2022) and ethnicity (Koenecke et al., 2020; Martin and Tang, 2020). For instance, ASR systems designed for American English typically perform worse for non-white speakers than for white speakers (Tatman and Kasten, 2017; Koenecke et al., 2020). Such inequalities result in certain groups of speakers being better represented than others, and may even further exacerbate existing inequalities in society.

At the same time, less is known about how the performance of ASR systems can vary for second-language (L2) English speakers, a particularly vulnerable population of English speakers with diverse backgrounds. A recent work by Chan et al. (2022) examined how Otter, a popular automatic transcription system, performs on L1 (native) and L2 (second-language) speakers of 24 English varieties. Not only do the English varieties supported by Otter have lower Word Error Rates (WER) compared to the unsupported varieties, gaps in performance is also driven by an independent effect of the structure of a speaker's first language – Otter performs worse on English spoken by L1 speakers of a tonal language (e.g., Mandarin).

While the Word Error Rate has been widely adopted in these studies due to the ease of quantifying system bias, this one-dimensional measure of performance is inadequate if the aim is to disentangle different types of errors that give rise to discrepancies in performance between speakers. For example, word-level errors can be driven by an error in one or multiple sound segments, and certain errors for consonants and vowels may be more common for speakers of one variety than speakers of another. Studying these details is useful because these linguistic categories are well-studied theoretical constructs and empirical phenomena in the research on sociolinguistic variation (e.g., Koenecke et al., 2020; Wassink et al., 2022) and L2 transfer (e.g., Corder, 1983; Dechert and Raupach, 1989; Best et al., 1994). Moreover, such linguistically-motivated features like conso-

Proceedings of the 29th International Conference on Computational Linguistics, pages 7177–7186
October 12–17, 2022.

nant voicing and vowel height have acoustic correlates, which means that insights from studying system errors in terms of phonological variation and processes can be translated to applied system engineering.

Therefore, the current study aims to systematically investigate whether variations in language structure among native and non-native language speakers of English are tied to different types and degrees of transcriptions errors. First, we introduce and motivate a segment-level error analysis built on traditional error-rate algorithms, which allows an analysis of errors beyond the single dimension of performance. We then investigate the error profile for consonants and vowels across various English varieties, to determine whether the phonological structure of a speaker's native language/variety is predictably tied to certain types of errors that can be captured in terms of phonological processes.

2 Materials and Methods

2.1 Speech Recognition System

We evaluate the performance of Otter, a speech recognition platform for automatic transcriptions that claims to support multiple varieties of English including "(southern) American, Canadian, Indian, Chinese, Russian, British, Scottish, Italian, German, Swiss, Irish, Scandinavian, and other European accents" (Lai, 2021).[1] Fittingly, Otter's live captioning system is used by popular video conferencing platforms that reach broad international audiences, such as Zoom. Otter's global user-base and its incorporation into educational and professional settings make it an ideal candidate for investigating whether there exists language-specific biases in ASR performance for non-native (L2) speakers of English, and if so, how these biases relate to the phonological structure of the speakers' native (L1) languages.

2.2 Corpus

The data analyzed in this study are collected from the Speech Accent Archive at "http://accent.gmu.edu" (Weinberger, 2015), a comprehensive speech corpus of nearly three thousand recordings from English speakers of diverse geographical and language backgrounds. Each entry in the corpus is a speaker reading the same passage ("Please Call Stella") at a table in a quiet room, seated approximately 8-10 inches from the microphone. The passage is designed to include words that elicit all sound segments in English, which allows for a direct comparison between speakers of different language backgrounds. The passage is 77-words long and recordings were around 30 seconds long on average.

Each recording in the corpus is accompanied by a list of demographic information about the speaker, including birthplace, age at the time of recording, sex, native language, age of English onset, English learning method (naturalistic vs. academic), among others. Notably, some of the recordings are also coded by trained linguists for accent features (e.g., vowel shortening), which used to motivate the error categories for evaluating ASR performance, as will be described in detail in later sections.

Following Chan et al. (2022), we filtered the corpus based on the following criteria: (1) Only varieties of English that are either listed as supported English varieties by Otter, or (2) have recording entries from at least 10 speakers. To balance the effect of system training in the data, we selected eleven varieties supported by Otter and sampled another eleven from the remaining non-supported varieties, for a total of 1,227 speakers/recordings.

All recordings were re-sampled to 22,050 Hz and concatenated with one-second pause inserted between each recording. The resulting 9.5-hour audio file was split into 4-hour chunks (the maximum file size permitted by Otter) before uploading to Otter. Otter's speaker detection system split the transcription output by speaker, though with occasional errors (10%) where multiple recordings of the passage were merged and determined as coming from the same speaker. Two human annotators corrected these errors independently, reaching 99.8% agreement, and a third annotator resolved the conflicts. The transcriptions with corrected alignments to speakers were entered into the error analysis.

2.3 Segment-level Error Analysis

We first ran a Word Error Rate (WER) algorithm by identifying word-level insertions, deletions, and substitutions for each speaker's transcribed string of words ("observed") compared to the reading passage ("truth"). Then, a Phone Error Rate (PER) was calculated for each speaker by first converting the word-level insertions, deletions, and substitutions to phones (sound seg-

[1] Note that while Otter uses the term "accents", we will be adopting the word "varieties" to refer to Englishes from both native and non-native speakers in this paper.

ments) using the CMU English Pronouncing Dictionary ("http://www.speech.cs.cmu.edu/cgi-bin/cmudict"), and passing the resulting string of phones to the same error rate algorithm.

To illustrate, consider the word "ask" from the reading passage, which has an entry in the CMU dictionary as "AE S K", the machine-readable ARPABET transcription representing the International Phonetic Alphabet (IPA) transcription /æsk/. A word-level deletion of "ask" in a transcribed output constitutes three phone errors (counting "AE", "S", and "K"), while a word-level substitution of "ask" with "asked" constitutes just one phone error (word-final insertion of "T" representing /t/). Thus, the phone-level measure of PER can be more precise about the egregiousness of errors made by an ASR system compared to the more traditional benchmark of WER which lacks such sensitivity (Aksënova et al., 2021; Wassink et al., 2022). As discussed earlier, a prior study by Chan et al. (2022) finds an effect of training and language structure across both measures, but the specific source of this disparity remains under-investigated, especially where the distribution of phone-level errors are concerned.

Therefore, the analyses in this paper go beyond the singular measures of performance (WER and PER) to investigate the language-specific profile of segment errors, using the phone-level substitutions identified by the PER algorithm. Phone substitutions are interesting from a linguistic standpoint because they allow an analysis of transcription errors in terms of phonological processes; a necessary first step for interdisciplinary work incorporating domain knowledge from linguistics (e.g., second language acquisition, language typology, and sociolinguistic variation). For example, a phone substitution of the vowel /ɪ/ as in "bit" to /i/ as in "beat" represents vowel lengthening. If a system consistently makes errors for a speaker's production of this short /ɪ/ vowel, then that error profile, combined with the language background of the speaker, can be identified as an actionable area of improvement for accent adaptation algorithms in ASR. The next two sections split the analysis of segment errors by consonants and vowels, with the specific methods for each detailed therein.

3 Consonant Analysis

In this section, we zoom in on the consonant errors in the Otter transcription of the same passage read by L2 English speakers with different L1s. It has been established that phonological features of L1 are likely transferred into speakers' L2 (Dechert and Raupach, 1989; Corder, 1983; Best et al., 1994). For example, given that Japanese only allows CV syllables, native speakers of Japanese have more difficulty with complex consonant cluster pronunciation (e.g. "sixth") when speaking English. Therefore, it is not surprising if we find traces of different L1s in our current data set of L2 Englishes. However, what remains under-explored is whether these phonological differences of L1s will be directly reflected in the kinds of consonant errors ASR algorithms make on these nonnative Englishes, especially one that claimed to have trained on non-native accents of English (Lai, 2021). We specifically test this in the following section.

3.1 Methods

With the help of the PER algorithm introduced in Section 2, we identified 2382 errors involving consonant substitutions. Given our primary interest in the relationship between the phonological structure of different L1s and the distribution of consonant errors, we focused on two types of errors that are the most prevalent in the data, as the robust number of tokens allows us to observe cross-linguistic variation. These two types are Cluster errors (e.g. transcribing "ask" as "asked") and Voicing errors (e.g. transcribing "bag" as "back"). We test two hypotheses for the distribution of cluster and Voicing errors, respectively.

First, we hypothesize that the syllable structure of speakers' L1 language (specifically, whether it allows consonant clusters) drives the rate of Cluster errors in their L2 English. The more different a speaker's L1 is from English in this respect, the more likely it is for Otter to make Cluster errors for that speaker. To test this hypothesis, we coded each L1 language in terms of whether they allow consonant clusters at syllable onset (yes vs. no) and syllable coda (yes vs. no) based on descriptions of the phonology of these languages in the literature (Ohala, 1983; Potet, 1995; Mahootian and Gebhardt, 1997; International Phonetic Association, 1999; Mazhar and Ranjha, 2012; Sircar and Nag, 2013; among others). We then categorized languages into four types: those allowing consonant clusters at both syllable onset and coda (Onset-Coda), those allowing such clusters only at syllable onset (Onset-only), only at syllable coda

(Coda-only) and at neither position (Neither) (see Table 1). Since languages that allow consonant clusters at neither position would be the most different from English, we predict Otter to perform the worst on the production of English consonant clusters by native speakers of languages in the Neither category. To statistically test for this effect, we fitted a linear mixed-effect model predicting the number of Cluster errors with L1 syllable structure type (4 levels, sum coded) as a fixed effect and a random intercept by language group, using the *lme4* package in R (Bates et al., 2015).

Table 1: Coding of whether a language allows consonant clusters at syllable onset or coda.

	Onset	Coda	Type
English	+	+	*OnsetCoda*
German	+	+	*OnsetCoda*
French	+	+	*OnsetCoda*
Spanish	+	+	*OnsetCoda*
Russian	+	+	*OnsetCoda*
Swedish	+	+	*OnsetCoda*
Swissgerman	+		*OnsetCoda*
Italian	+		*OnsetOnly*
Bengali	+		*OnsetOnly*
Hindi		+	*CodaOnly*
Urdu		+	*CodaOnly*
Dari		+	*CodaOnly*
Mandarin			*Neither*
Cantonese			*Neither*
Japanese			*Neither*
Korean			*Neither*
Thai			*Neither*
Vietnamese			*Neither*
Indonesian			*Neither*
Arabic			*Neither*
Amharic			*Neither*
Tagalog			*Neither*

Second, we hypothesize that the realization of the voicing contrast in the L1 language should drive the rate of Voicing errors in L2 English. To test this hypothesis, we coded each L1 language in terms of how their voicing contrasts is realized into three levels: those that have true voicing contrasts (1), those that have voicing contrast but are not realized as true voicing (2) and those that have no voicing contrast (3). Additionally, we coded whether there is a phonemic aspiration contrast in the L1 language (1: yes vs 2: no). The coding was based on phonetic and phonological descriptions of these languages in the linguistic literature (Henderson, 1972; Thelwall and Sa'Adeddin, 1990; International Phonetic Association, 1999; Fleischer and Schmid, 2006; Petrova et al., 2006; Mikuteit and Reetz, 2007; Soderberg and Olson, 2008; Kramer, 2009; Gallagher, 2010; Labrune, 2012; Tranová, 2016). We further categorized the languages into five types according to these two dimensions. The complete coding and categorization of all L1 languages can be found in Table 2. To statistically test for this effect, we fitted a linear mixed-effect model predicting the number of Voicing errors with errors with L1 stop contrast type (5 levels, sum coded) as a fixed effect and a random intercept by language group. For simplicity, we refer to each group by a representative language in the group (e.g. the Hindi-type) in the following analysis.

Table 2: Coding of language stop voicing and aspiration contrasts (language in bold is used as group name). Numbers represent each language's category in the typology of voicing and aspiration contrasts.

	Voicing	Aspiration
Hindi	1	1
Vietnamese	1	1
Thai	1	1
Bengali	1	1
Indonesian	1	1
Swedish	1	1
Urdu	1	1
French	1	2
Amharic	1	2
Russian	1	2
Italian	1	2
Arabic	1	2
Dari	1	2
Spanish	1	2
Tagalog	1	2
Japanese	2	1
Korean	2	1
English	2	2
German	2	2
Swissgerman	2	2
Mandarin	3	1
Cantonese	3	1

3.2 Results

3.2.1 Syllable structure and Cluster Error

As shown in Figure 1, we find that L1 languages that do not allow consonant clusters at either sylla-

ble onset or coda receive significantly more cluster errors per speaker in Otter transcription (β = 0.429, SE = 0.118, p = 0.002). This is consistent with our prediction that the rate of consonant cluster errors are driven by the extent to which consonant clusters are licensed in the syllable structure of a speaker's native language. At the same time, L1 languages that allow consonant clusters at more restricted positions (Onset only & Coda only) are not significantly different from languages that allow them in both positions (Onset-Coda). In sum, we observe degraded performance on the transcription of words with consonant clusters driven by the degree of difference in syllable structure between English and speakers' L1 languages.

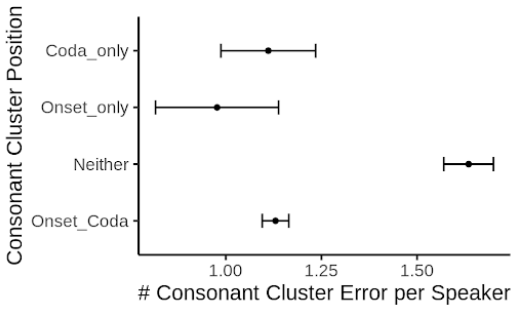

Figure 1: The number of Cluster errors per speaker in L2 English transcription by L1 consonant cluster type.

3.2.2 Stop Voicing and Voicing Error

As shown in Figure 2, we find that L1 languages that realize stop contrasts just like English receive the least voicing errors per speaker in Otter transcription (β = -0.399, SE = 0.138, p = 0.017). Additionally, find a marginal effect of Otter transcriptions generating more voicing errors for Mandarin-type languages (β = 0.302, SE = 0.150, p = 0.075). One explanation for this effect is that, as shown in Table 2, Mandarin-type languages are the most different from English in terms of the phonological structure of the stop consonants - instead of a phonemic voicing contrast, there is a phonemic aspiration contrast. Thus, we find evidence for our hypothesis that the degree of difference in the phonological structure of a speaker's L1 compared to English is directly reflected in the accuracy of automatic transcriptions for their production of L2 English.

4 Vowel Analysis

In the following section, we turn to the distribution of vowel errors in the Otter transcriptions. We ana-

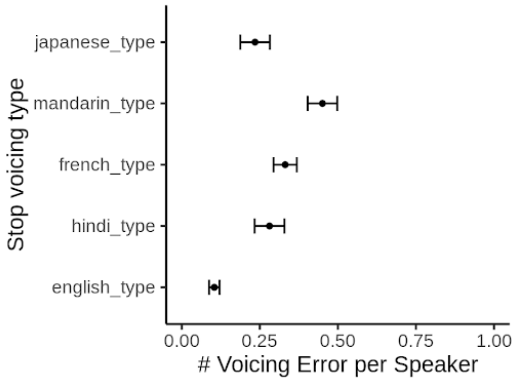

Figure 2: The number of Voicing errors per speaker in L2 English transcription by L1 stop contrast type.

lyze the distribution of vowel errors in the acoustic space in context of the typology of vowel systems among the L1 languages represented in the data. Specifically, we focus on vowel substitution errors (as opposed to insertions and deletions) as they allow us to explore Otter's errors in terms of well-studied phonological processes.

4.1 Methods

All 1,227 sound files selected for analysis were ran through the Penn Phonetics Lab Forced Aligner (Yuan et al., 2008) to align our recordings at the segmental level. We then extracted the first two formants (F1 and F2) at the midpoint all vowel tokens using the LPC (burg) function in Praat (Boersma, 2006) and z-scored all formant measurements by speaker. While formant measurements at the 25% and 75% were also extracted initially to analyze diphthongal patterns, there were insufficient diphthong tokens in the passage to draw conclusive results. Therefore, we focus our analysis on the acoustic measures of monophthong tokens.

We visualize our acoustic analysis in Figure 3. The x-axis is the z-scored F2 and the y-axis is the z-scored F1; axes are plotted in reverse to match the height and backness dimensions of the physical vowel space. The figure highlights two vowel analyses conducted for each language background. First is Otter's perceived vowel space, represented by the black solid lines that connect the four edges of the vowel space. The perceived vowel space is constructed from cases where the "truth" vowel and the "observed" vowel match - i.e., when Otter correctly transcribes the intended vowel that was produced. Second is Otter's regions of error, represented by a color contour imposed on top of the perceived vowel space. The error regions are constructed

from the formant values of vowels involved in substitution errors - i.e., when the transcribed vowel does not match the intended vowel that a speaker produced. In sum, we analyzed the acoustic profile of matches and mismatches between "true" and "observed" vowel tokens for each language background.

Based on the distribution of vowel errors from the acoustic analysis, we categorized the L1 language varieties into five vowel error types. Depending on where errors are concentrated in the vowel space, each language was assigned to at least one of the following: (i) high front vowels, (ii) high back vowels, (iii) high vowels, (iv) low vowels, and (v) point vowels. The vowel error categories are summarized in Table 3.

Table 3: Vowel Error Category.

Language	Error Category
Vietnamese	High Front Vowels
Mandarin	High Front/Point Vowels
Thai	High Front Vowels
Korean	High Front Vowels
Hindi	High Front Vowels
Amharic	High Vowels
Cantonese	High Vowels
French	High Vowels
Italian	High Vowels
Tagalog	High Vowels
German	High Vowels
English (USA)	High Vowels
Bengali	Low Vowels
Russian	Low Vowels
Dari	Low/Point Vowels
Indonesian	High Back Vowels
Spanish	High Back Vowels
Japanese	High Back Vowels
English (UK)	High Back Vowels
Swiss German	High Back Vowels
Swedish	High Back Vowels
English (Canada)	High Back Vowels
Arabic	Point Vowels
Urdu	Point Vowels

4.2 Results

The concentration of errors within the vowel spaces reveals two main trends. First, languages with fewer vowel distinctions than English at a phonemic level, in either the entire vowel space or in certain parts of the vowel space, have higher vowel substitution errors. As predicted, the distribution of the types of vowel errors are language-specific, such that errors concentrate on specific regions of the vowel space where the L1 phonology makes less distinctions than in English. This is not surprising given the literature on second language acquisition such as the Perceptual Assimilation Model (Best et al., 1994), which posits that listeners perceive non-native phones in terms of the similarities or dissimilarities of the phones to their native phonemic contrasts. For instance, Vietnamese (Kirby, 2011) and Thai (International Phonetic Association, 1999) have comparable phonological vowel spaces, and neither has a tense-lax contrast for high front vowels (which English does have). Consequently, we find vowel substitution errors concentrated in the high front region of the vowel space. At the same time, though both languages also lack the tense-lax contrast in the high back region, the existence of a roundness contrast may have been used to disambiguate high back vowels in their L2 English pronunciation. When a Vietnamese speaker says the words "thick slabs" (transcribed as /θɪk slæbz/ in the CMU dictionary) for example, Otter transcribes it as "techs lab" /tɛks læb/, replacing the /ɪ/ vowel with /ɛ/, which is in the Vietnamese vowel inventory. Another example of languages with fewer vowel distinctions than English is Arabic, which has been described to contrast three main monophthongal vowel qualities, also referred to as point vowels. We find that Otter primarily misidentifies tokens spoken at the point regions of the vowel space by Arabic speakers of English. For example, when an Arabic speaker produces "Stella" (/stɛlə/), the Otter transcription confused /ɛ/ as /i/, yielding the transcription "stealer" /stilɚ/. These results confirm existing findings for speakers localizing their pronunciation of non-native phones to the categories available in their language. Critically, we find that this also drives language-specific phone-level transcription errors from Otter.

Second, languages that have a similar phonological structure to that of English in its vowel inventory have an overall lower vowel substitution rate (and lower word error rate, writ large). Interestingly, our predictions are still borne out in the vowel error profile of these languages for which Otter performs well. An example of this is German, which was categorized into the high vowel error category. The vowel inventory of German is simi-

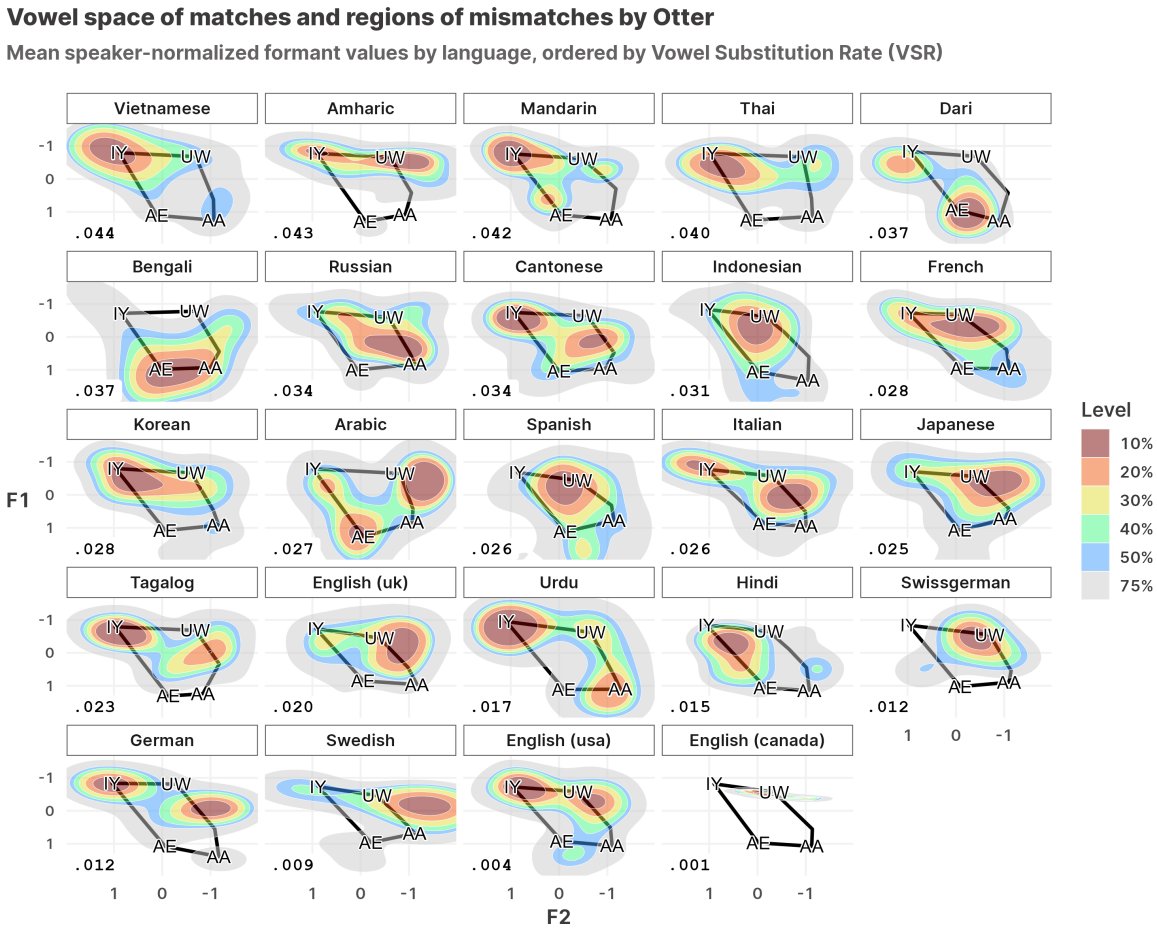

Figure 3: A speaker-normalized (z-scored) F1-by-F2 vowel plot, split by speakers' L1. The black polygon represents the vowel space detected by Otter, composed of correctly identified vowels. The density contours are constructed from the distribution of incorrectly identified vowels via a highest-density-region (HDR) estimation (Otto and Kahle, 2022), where the fill color represents the percentage of vowel errors occurring within a specific region. Languages are ordered by their vowel substitution rate (annotated in the bottom left of each panel), where a higher number reflects worse performance on vowel identification.

lar to that of English in many ways; for example, it contrasts both tense and lax high vowels, contrasts open-mid and closed-mid cardinal vowels, and includes the unstressed vowel schwa (International Phonetic Association, 1999). Unlike English, however, German lacks the back vowel /ʌ/, which may explain the concentration of back vowel errors in the vowel space. For example, one German speaker's pronunciation of "brother" /brʌðɚ/ was transcribed by Otter as "product" /pɹɑdəkt/ instead, where the /ʌ/ vowel was replaced by another, lower back vowel /ɑ/.

Lastly, we also find predictable errors that reflect well-documented sociolinguistic variation among L1 English speakers. For example, the vowel substitution errors made by Otter in Canadian English are highly localized in the high vowel region, which likely reflects Canadian raising (Chambers, 1973). In sum, the acoustic analysis of vowel substitution errors show that the phonological structure of a native language's vowel spaces can inform us of specific gaps in the performance of ASR systems.

5 Discussion

This study investigated the performance of Otter's automatic captioning system on native and non-native English speakers of different language backgrounds through a linguistic analysis of segment-level substitution errors. We proposed that understanding language-specific error profiles is crucial to preempting predictable system errors. In our analysis of consonant and vowel errors motivated by phonological theory, we report the following findings.

Results from our consonant analysis show that

the phonological structure of a non-native English speaker's first language predicts the types of consonant errors that are dominant in the automatic transcription of their production of English. Specifically, we find higher rates of consonant cluster errors in the transcription of speakers whose native language does not allow consonant clusters. Similarly, we find higher rates of stop voicing errors for speakers whose first language has aspiration contrasts, as opposed to voicing contrasts like English.

Results from our vowel analysis show that the distribution of vowel substitution errors patterns with the structure of the vowel inventory of a non-native English speaker's first language. Specifically, we find three main patterns. First, the transcriptions of non-native speakers whose first language make fewer vowel distinctions than English show predictable regions of error in the vowel space. Second, when languages have similar phonological structures compared to English, the frequency of vowel substitution errors tend to be lower. Third, known sociolinguistic variation even among native speakers of English also predict vowel errors in a similar manner.

Understanding how the sound patterns of a speaker's native language and/or variety of English affect the performance of ASR systems on specific categories of sounds is an important first step towards designing robust yet flexible acoustic models which can detect and adapt to varieties without reducing structural differences to correlated demographic information like race and ethnicity (Tatman, 2020). Recent successes in the design of accent adaptation algorithms support this view, such as the automatic accent identification model by Najafian and Russell (2020) trained on regional varieties of British English. Furthermore, the focus on adapting to differences in linguistic structure grounded in phonological theory is crucial to the generalizability of ASR systems, which is an important consideration for providing transcription services to speakers of minority languages. For example, an ASR model designed to learn and leverage language-specific phonemic contrasts by Li et al. (2020) vastly improved phone-level accuracy on very small corpora (1k utterances) of two indigenous low-resource languages, Inuktitut and Tusom. Our study contributes to this momentum by offering insights into the phonological and acoustic nature of transcription errors for speakers of different language backgrounds. Such research on language-specific error profiles can motivate the design of adaptation algorithms for supporting non-native English speakers of various language backgrounds.

Lastly, improving the performance of ASR systems for non-native speakers of English is an important task not simply for the sake of catering to a large user-base given the status of English as the de facto lingua franca with far more L2 than L1 English speakers in the world (SIL International, 2022), but also because L2 speakers of English are an especially vulnerable population facing specific, and often invisible, harms from the prevailing stereotypes of being uninterpretable in speech and lacking education and proficiency in English (Lippi-Green, 2011). Many existing applications of ASR systems are ill-equipped to work with speech input from non-standard varieties, as evidenced by accumulating cases of discrimination against non-native speakers of English across all levels of harm (Blodgett et al., 2020). For example, allocational harms have been reported for even life-or-death situations such as in voice command systems for roadside vehicle assistance (Wassink et al., 2022) and for medical diagnoses and records management in healthcare systems (Lee, 2021). Moreover, representational harms from stereotypes of unintelligibility are perpetuated by systems that claim to work on a language while neglecting how the system might perform differently among sociolects (Aksënova et al., 2021). As an extreme example of this ideology, some recruiting firms have claimed to screen and rank job applications by passing their voice data through off-the-shelf ASR systems and using the interpretability of the transcription output itself as proxies for friendliness and communication skills, putting non-native English speakers at a disadvantage (Raghavan et al., 2020). In these ways, unchecked bias against non-native speakers of English in ASR systems reinforce social inequalities and simultaneously dismiss real cases of need for accommodation.

6 Conclusion

In this study, we examined the language-specific error profiles of native and non-native English speakers of diverse language backgrounds. A segment-level analysis of consonant and vowel errors made by Otter's transcription system reveals that certain categories of errors are predictable from the phonological structure of a speaker's native language.

Thus, we demonstrate the fruitfulness of applying a linguistic analysis to transcription errors, informed by general phonological theory as well as specific literature from relevant domains such as sociolinguistic variation and second language acquisition. Findings inform the design and maintenance of new and existing ASR systems for adapting to non-native speakers of English and speakers of non-standard English varieties.

References

Martine Adda-Decker and Lori Lamel. 2005. Do speech recognizers prefer female speakers? In *Ninth European Conference on Speech Communication and Technology*.

Alëna Aksënova, Daan van Esch, James Flynn, and Pavel Golik. 2021. How might we create better benchmarks for speech recognition? In *Proceedings of the 1st Workshop on Benchmarking: Past, Present and Future*, Online. Association for Computational Linguistics.

International Phonetic Association, International Phonetic Association Staff, et al. 1999. *Handbook of the International Phonetic Association: A guide to the use of the International Phonetic Alphabet*. Cambridge University Press.

Douglas Bates, Martin Mächler, Ben Bolker, and Steve Walker. 2015. Fitting linear mixed-effects models using lme4. *Journal of Statistical Software*, 67(1):1–48.

Catherine T. Best et al. 1994. The emergence of native-language phonological influences in infants: A perceptual assimilation model. *The development of speech perception: The transition from speech sounds to spoken words*, 167(224):233–277.

Su Lin Blodgett, Solon Barocas, Hal Daumé III, and Hanna Wallach. 2020. Language (technology) is power: A critical survey of "bias" in nlp. *arXiv preprint. arXiv:2005.14050*.

Paul Boersma. 2006. Praat: doing phonetics by computer. *http://www. praat. org/*.

Marcely Zanon Boito, Laurent Besacier, Natalia Tomashenko, and Yannick Estève. 2022. A study of gender impact in self-supervised models for speech-to-text systems. *arXiv preprint arXiv:2204.01397*.

Jack K. Chambers. 1973. Canadian raising. *Canadian Journal of Linguistics/Revue canadienne de linguistique*, 18(2):113–135.

May Pik Yu Chan, June Choe, Aini Li, Yiran Chen, Xin Gao, and Nicole Holliday. 2022. Training and typological bias in asr performance for world englishes. In *Proceedings of the 23rd Conference of the International Speech Communication Association*.

Stephen Pit Corder. 1983. A role for the mother tongue. *Language transfer in language learning*, 1:85–97.

Hans-Wilhelm Dechert and Manfred Raupach. 1989. *Transfer in language production*. Praeger.

Jürg Fleischer and Stephan Schmid. 2006. Zurich german. *Journal of the International Phonetic Association*, 36(2):243–253.

Gillian Elizabeth Scott Gallagher. 2010. *The perceptual basis of long-distance laryngeal restrictions*. Ph.D. thesis, Massachusetts Institute of Technology.

Michael Magnus Thyne Henderson. 1972. *Dari (Kabul Persian) Phonology*. The University of Wisconsin-Madison.

Dirk Hovy and Shrimai Prabhumoye. 2021. Five sources of bias in natural language processing. *Language and Linguistics Compass*, 15(8):e12432.

James P. Kirby. 2011. Vietnamese (Hanoi Vietnamese). *Journal of the International Phonetic Association*, 41(3):381–392.

Allison Koenecke, Andrew Nam, Emily Lake, Joe Nudell, Minnie Quartey, Zion Mengesha, Connor Toups, John R Rickford, Dan Jurafsky, and Sharad Goel. 2020. Racial disparities in automated speech recognition. *Proceedings of the National Academy of Sciences*, 117(14):7684–7689.

Martin Kramer. 2009. *The phonology of Italian*. OUP Oxford.

Laurence Labrune. 2012. *The phonology of Japanese*. Oxford University Press.

Allen Lai. 2021. Supported languages. *Otter.ai Help Center*.

Dave Lee. 2021. The next big tech battle: Amazon's bet on healthcare begins to take shape. *Financial Times*.

Xinjian Li, Siddharth Dalmia, Juncheng Li, Matthew Lee, Patrick Littell, Jiali Yao, Antonios Anastasopoulos, David R. Mortensen, Graham Neubig, Alan W Black, and Florian Metze. 2020. Universal phone recognition with a multilingual allophone system. In *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 8249–8253.

Rosina Lippi-Green. 2011. *English with accents: Language, ideology, and discrimination in the United States*. Routledge, New York.

Shahrzad Mahootian and Lewis Gebhardt. 1997. Persian (descriptive grammars). *London: Routledge. doi*, 10:9780203192887.

Nina Markl and Stephen Joseph McNulty. 2022. Language technology practitioners as language managers: arbitrating data bias and predictive bias in asr. *arXiv preprint arXiv:2202.12603*.

Joshua L. Martin and Kevin Tang. 2020. Understanding racial disparities in automatic speech recognition: The case of habitual "be". In *INTERSPEECH*, pages 626–630.

Iqbal Mazhar and Mazhar Ranjha. 2012. Urdu syllable: Templates and constraints. In *Proceeding of the Conference on Language and technology 2012*.

Josh Meyer, Lindy Rauchenstein, Joshua D. Eisenberg, and Nicholas Howell. 2020. Artie bias corpus: An open dataset for detecting demographic bias in speech applications. In *Proceedings of the 12th Language Resources and Evaluation Conference*, pages 6462–6468.

Simone Mikuteit and Henning Reetz. 2007. Caught in the ACT: The timing of aspiration and voicing in East Bengali. *Language and speech*, 50(2):247–277.

Maryam Najafian and Martin Russell. 2020. Automatic accent identification as an analytical tool for accent robust automatic speech recognition. *Speech Communication*, 122:44–55.

Manjari Ohala. 1983. *Aspects of Hindi phonology*. Motilal Banarsidass Publishers Pvt. Limited.

James Otto and David Kahle. 2022. *ggdensity: Interpretable Bivariate Density Visualization with 'ggplot2'*. Https://jamesotto852.github.io/ggdensity/, https://github.com/jamesotto852/ggdensity/.

Olga Petrova, Rosemary Plapp, Catherine Ringen, and Szilárd Szentgyörgyi. 2006. *Voice and aspiration: Evidence from Russian, Hungarian, German, Swedish, and Turkish*. Walter de Gruyter.

Jean-Paul G. Potet. 1995. Tagalog monosyllabic roots. *Oceanic linguistics*, pages 345–374.

Manish Raghavan, Solon Barocas, Jon Kleinberg, and Karen Levy. 2020. Mitigating bias in algorithmic hiring: Evaluating claims and practices. In *Proceedings of the 2020 conference on fairness, accountability, and transparency*, pages 469–481.

Majdi Sawalha and Mohammad Abushariah. 2013. The effects of speakers' gender, age, and region on overall performance of Arabic automatic speech recognition systems using the phonetically rich and balanced Modern Standard Arabic speech corpus. In *Proceedings of the 2nd Workshop of Arabic Corpus Linguistics WACL-2*. Leeds.

SIL International. 2022. Ethnologue: English.

Shruti Sircar and Sonali Nag. 2013. 19 akshara–syllable mappings in bengali: a language-specific skill for reading. *South and Southeast Asian psycholinguistics*, page 202.

Craig D. Soderberg and Kenneth S. Olson. 2008. Indonesian. *Journal of the International Phonetic Association*, 38(2):209–213.

Rachael Tatman. 2017. Gender and dialect bias in youtube's automatic captions. In *Proceedings of the first ACL workshop on ethics in natural language processing*, pages 53–59.

Rachael Tatman. 2020. Sociolinguistic variation and automatic speech recognition: Challenges and approaches. In *2020 Annual Meeting*. AAAS.

Rachael Tatman and Conner Kasten. 2017. Effects of talker dialect, gender & race on accuracy of bing speech and youtube automatic captions. In *Interspeech*, pages 934–938.

Robin Thelwall and M. Akram Sa'Adeddin. 1990. Arabic. *Journal of the International Phonetic Association*, 20(2):37–39.

Lenka Tranová. 2016. *The voicing contrast in Vietnamese English*. Univerzita Karlova, Filozofická fakulta.

Alicia Beckford Wassink, Cady Gansen, and Isabel Bartholomew. 2022. Uneven success: automatic speech recognition and ethnicity-related dialects. *Speech Communication*, 140:50–70.

Steven Weinberger. 2015. Speech accent archive. george mason university. *http://accent.gmu.edu*.

Barbara Wheatley and Joseph Picone. 1991. Voice Across America: Toward robust speaker-independent speech recognition for telecommunications applications. *Digital Signal Processing*, 1(2):45–63.

Jiahong Yuan, Mark Liberman, et al. 2008. Speaker identification on the scotus corpus. *Journal of the Acoustical Society of America*, 123(5):3878.

A Transformer-based Threshold-Free Framework for Multi-Intent NLU

Lisung Chen[1], **Nuo Chen**[1], **Yuexian Zou**[1], **Yong Wang**[2], **Xinzhong Sun**[2]
[1]ADSPLAB, School of ECE, Peking University, Shenzhen, China
[2]AOTO Electronics Co., Ltd

Abstract

Multi-intent natural language understanding (NLU) has recently gained attention. It detects multiple intents in an utterance, which is better suited to real-world scenarios. However, the state-of-the-art joint NLU models mainly detect multiple intents on threshold-based strategy, resulting in one main issue: the model is extremely sensitive to the threshold settings. In this paper, we propose a transformer-based Threshold-Free Multi-intent NLU model (TFMN) with multi-task learning (MTL). Specifically, we first leverage multiple layers of a transformer-based encoder to generate multi-grain representations. Then we exploit the information of the number of multiple intents in each utterance without additional manual annotations and propose an auxiliary detection task: Intent Number detection (IND). Furthermore, we propose a threshold-free intent multi-intent classifier that utilizes the output of IND task and detects the multiple intents without depending on the threshold. Extensive experiments demonstrate that our proposed model achieves superior results on two public multi-intent datasets.

1 Introduction

Natural language understanding (NLU) consists of two sub-tasks, including intent detection (ID) and slot filling (SF) which allow the dialogue system to create a semantic frame that summarizes the user's requests. Early works often approach these two tasks separately (McCallum et al., 2000; Sarikaya et al., 2011; Yao et al., 2014; Vu, 2016). Considering intent detection and slot filling are highly related, recent works tend to model these two tasks jointly, where the correlation between the intent and slots are utilized (Goo et al., 2018; E et al., 2019; Qin et al., 2019; Zhou et al., 2021).

The works above only consider the scenario where each utterance has one intent. However, in real-life situations, users may express multiple intents in an utterance, thus making it difficult to apply single intent NLU models. Recently, several works have studied Multi-intent NLU problem. Gangadharaiah *et al.*(2019) investigated an attention-based neural network. Qin *et al.*(2020) proposed an Adaptive Graph Interactive Framework (AGIF). Qin *et al.*(2021) explored a non-autoregressive approach to speed up the inference time. Chen *et al.*(2022a) proposed a Self-distillation Joint NLU model. However, these works all predict multiple intents with threshold, where the common practice is estimating label-instance probabilities and picking the intent labels whose probabilities are higher than the threshold value. We named them threshold-based models. The main issue of threshold-based models is that they are not robust to the threshold settings. As shown in Figure 1, the correct intents for the utterance are 'GetWeather' and 'SearchScreeningEvent'. Although the model can detect that 'GetWeather' and 'SearchScreeningEvent' are the two most probable intents, the threshold-based model only considers 'GetWeather' as the intent due to the threshold which is usually set as 0.5.

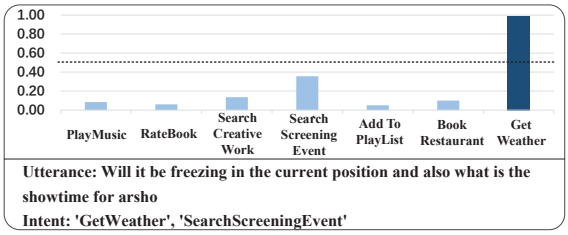

Figure 1: A threshold-based multi-intent detection example in MixSNIPS with given utterance and intent labels. Threshold, which is the dash line, is set to 0.5.

In this paper, we propose a transformer-based **T**hreshold-**f**ree **M**ulti-**N**LU model (TFMN) and detect multiple intents without relying on the threshold. Specifically, we leverage the upper layers of a transformer-based encoder to generate multi-grain representations. Next, we fully exploit the annota-

Proceedings of the 29th International Conference on Computational Linguistics, pages 7187–7192
October 12–17, 2022.

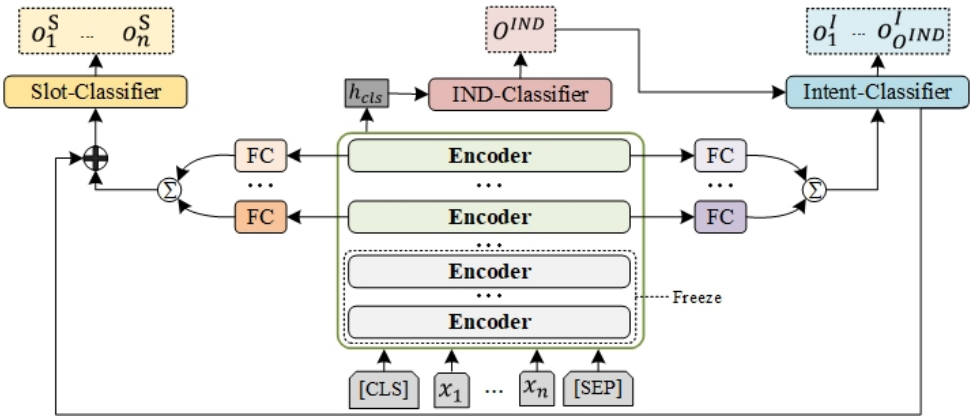

Figure 2: The architecture of transformer-based TFMN model.

tions from original multiple intents data and propose an Intent Number Detection (IND) task. The motivation is to allow the model to detect the intent numbers in a given utterance. Then we propose a threshold-free intent classifier that utilizes the output of IND task to detect the multiple intents.

We validate TFMN on two public datasets (Qin et al., 2020): MixATIS and MixSNIPS, and show that our method outperforms competitive baselines. The contributions of our work are summarized as follows: (1)We propose a novel threshold-free Multi-NLU model based on transformers.(2) We propose IND task, a feasible task to improve the multi-intent NLU without additional manual annotation, and a threshold-free multi-intent classifier that detects multiple intents without relying on threshold. (3) We present extensive experiments demonstrating the effectiveness of our approach.

2 Problem Formulation

Given an input sequence $X = (x_1, ..., x_n)$, multi-intent detection is defined as a multi-label classification task that outputs $O^I = (o_1^I, ..., o_m^I)$, where m is the number of predicted intent labels. Slot filling task can be regarded as a sequence labeling task that outputs a slot sequence $O^S = (o_1^S, ..., o_n^S)$.

3 Approach

In this section, we first introduce the architecture of TFMN model, then detail the proposed IND task and threshold-free intent classifier.

3.1 Threshold-free Multi-intent NLU Model

The architecture of our model is illustrated in Figure 2. TFMN includes a transformer-based encoder with L layers and three task-specific classifiers.

Multiple Intent Detection Following (Qin et al., 2019), we perform a token-level multiple intent detection which can be formalized as a sequence labeling problem (You et al., 2020, 2021b; Chen et al., 2021a, 2022b) that maps the input utterance $X = (x_1, ..., x_n)$ to sequence of intent label $O^I = (o_1^I, ..., o_n^I)$. According to (Jawahar et al., 2019; Rogers et al., 2020; Chen et al., 2021b), transformer-based encoder tends to capture syntactic information in the middle and semantic information at the top layers. Therefore, we take the top j layers of the encoder to form multi-grain intent features. First, we map each hidden layer into a different feature space via a fully connected layer, then we combine hidden layers by adding them together:

$$h^I = \sum_{n=L-j}^{L} w_n^I h_n \tag{1}$$

where w_n^I are trainable parameters and h_n are different hidden layers. We then generate intent logits with the intent feature h_I:

$$l^I = w_i h^I \tag{2}$$

where w_i are trainable parameters. The intent logits will be used to provide token-level intent information for slot filling and detect the final multiple intent labels which we will detail in Section 3.3.

Slot Filling Similar to intent detection, We leverage the top j layers of a transformer-based encoder for slot filling. The slot features h^S are generated by combining hidden layers and concatenating with token-level intent information:

$$h_{temp}^S = \sum_{n=L-j}^{L} w_n^S h_n \tag{3}$$

$$h^S = h_{temp}^s \oplus l^I \tag{4}$$

then slot classifier computes the slot prediction:

$$p_t^S = softmax(w_a LeakyReLU(w_b h_t^S)) \quad (5)$$

where w_a and w_b are trainable parameters.

3.2 Intent Number Detection

To achieve threshold-free multi-intent detection, we propose an Intent Number Detection task which trains with the intent detection and slot filling in a multi-task fashion. In IND task, we fully utilize the original intent label annotations by calculating the numbers of intents in each utterance and forming the intent number labels Y^{IND}. Then we train the model to detect how many intents are there in the input utterance with Y^{IND}. Specifically, we take the output of [CLS] token from the last hidden layer h_{cls} as representation for IND task to classify:

$$p^{IND} = softmax(w_{ind} h_{cls}) \quad (6)$$

$$O^{IND} = argmax(p^{IND}) \quad (7)$$

We use cross-entropy to optimize IND task:

$$\mathcal{L}_{IND} = -\sum\nolimits_k y_k^{IND} log p_k^{IND} \quad (8)$$

3.3 Threshold-free Intent Classifier

Once having the intent logits l^I and being able to predict the intent number with the proposed IND task, we send l^I into a sigmoid activation function:

$$p_t^I = sigmoid(l_t^I) \quad (9)$$

where p_t^I is the intent probability distribution of t-th token in the utterance. Since the final output should be the utterance-level intent detection, we sum p_t^I up for utterance-level intent probability distribution P^I, and choose the top O^{IND}, which is the predicted intent number of the utterance, most probable intent label as the final result $O^I = (o_1^I, ..., o_{O^{IND}}^I)$.

3.4 Multi-Task Training

Our model optimizes the parameters jointly. Multiple intent detection is trained with binary cross-entropy and slot filling is trained with cross-entropy. The total loss of TFMN is the weighted sum of three losses:

$$\mathcal{L}_{total} = \alpha \cdot \mathcal{L}_{ID} + \beta \cdot \mathcal{L}_{SF} + \lambda \cdot \mathcal{L}_{IND} \quad (10)$$

with three hyper-parameters α, β, and λ to balance.

4 Experiments

4.1 Datasets

We conduct experiments on two public multi-intent NLU datasets[1]. They are MixATIS (Qin et al., 2020) collected from ATIS dataset (Hemphill et al., 1990) with 13162/759/828 utterances for train/validate/test and MixSNIPS (Qin et al., 2020) collected from SNIPS dataset (Coucke et al., 2018) with 39776/2198/2199 utterances for train/validate/test. Both of the datasets have the ratio of sentences with 1~3 intents as [0.3, 0.5, 0.2].

4.2 Experimental Settings

For TFMN, we use the English uncased Bert-Base model (Devlin et al., 2019) which consists of 12 hidden layers, 12 heads, and the hidden size is 768. For fine-tuning, we freeze the bottom half of Bert to save computational memory and empirically choose the top 4 layers to generate representations. The batch size is 128 and the epoch is 80. Adam is used for optimization with learning rate of 2e-5. The hyper-parameters of loss are empirically set as α: β: λ= 0.6: 1: 1 for MixATIS and α: β: λ= 0.7: 0.9: 1 for MixSNIPS. We evaluate the performance of slot filling with F1 score (You et al., 2021a; Chen et al., 2021c), intent detection with accuracy, and the NLU semantic frame parsing with overall accuracy.

4.3 Baselines

We compare our model with both single-intent and multi-intent baselines. For single-intent baselines to handle multi-intent utterances, multiple intent labels are connected with "#" and treated as a single label, named as *concat* version. For multi-intent baselines, they are all threshold-based models, named as *thresh* version. We also obtain our own pre-trained language model (PLM) baseline for comparison, called Bert-baseline. Following (Chen et al., 2019), we obtain the hidden state of the first special token ([CLS]) for detecting multi-intent based on threshold and use hidden states of utterance tokens for slot filling.

4.4 Results

The main results are illustrated in Table 1. We observe that TFMN model outperforms previous state-of-the-art baselines significantly. On slot filling, our model outperforms GL-GIN 1.5% on MixSNIPS. For multiple intent detection, we achieve

[1] https://github.com/LooperXX/AGIF

Model	MixATIS			MixSNIPS		
	Slot (F1)	Intent (Acc)	Overall (Acc)	Slot (F1)	Intent (Acc)	Overall (Acc)
SF-ID (*concat*) (2019)	87.4	66.2	34.9	90.6	95.0	59.9
Stack-Propagation ($thresh = 0.5$) (2019)	87.8	72.1	40.1	94.2	96.0	72.9
Joint Multiple ID-SF ($thresh = 0.5$) (2019)	84.6	73.4	36.1	90.6	95.1	62.9
AGIF ($thresh = 0.5$)(2020)	86.7	74.4	40.8	94.2	95.1	74.2
GL-GIN ($thresh = 0.5$)(2021)	**88.3**	76.3	43.5	94.9	95.6	75.4
SDJN ($thresh = 0.5$)(2022a)	88.2	77.1	44.6	94.4	96.5	75.7
SDJN+BERT ($thresh = 0.5$)(2022a)	87.5	78.0	46.3	95.4	96.7	79.3
Bert-baseline ($thresh = 0.3$)	83.1	74.8	42.6	95.5	95.7	80.2
Bert-baseline ($thresh = 0.5$)	86.3	74.5	44.8	95.5	95.6	80.1
Bert-baseline ($thresh = 0.8$)	85.6	75.8	43.5	95.2	96.7	80.6
TFMN (Bert-base)	88.0	**79.8**	**50.2**	**96.4**	**97.7**	**84.7**

Table 1: Slot filling and multiple intent detection results on two multi-intent datasets.

Model	MixATIS		
	Slot (F1)	Intent (Acc)	Overall (Acc)
TFMN	88.0	79.8	50.2
-w/o T-free Cls	87.1	77.3	47.0
-w/o T-free Cls & IND task	86.3	76.8	46.7

Table 2: Ablation study. *T-free Cls* indicates threshold-free intent classifier.

Model	MixATIS			
	Int-1	Int-2	Int-3	Avg.
AGIF	96.5	83.7	76.7	85.6
GL-GIN	96.5	94.6	87.5	92.8
SDJN+BERT	97.2	92.0	84.0	91.2
Bert-baseline ($thresh = 0.5$)	94.4	87.8	83.5	88.6
TFMN	**98.6**	**99.7**	**99.3**	**99.2**

Table 3: A comparison of intent number prediction between threshold-based and threshold-free approaches. The evaluation metric is accuracy. **Int-#** means the utterance with the number of "#" intent. Avg. is the average accuracy.

2.7% and 1.2% improvement compared with SDJN on MixATIS and MixSNIPS respectively. On overall accuracy, our model shows strong performance which surpasses SDJN 5.6% on MixATIS and 9% on MixSNIPS. When comparing PLM baselines, we can first observe that different threshold settings affect the results of Bert-baseline distinctively. Second, TFMN model outperforms PLM baselines in all three metrics on both datasets. The results suggest that our approach brings significant improvements to multi-intent NLU. We believe this is due to the proposed IND task which fully exploits original intent annotations and threshold-free intent classifier that allows our model to detect multiple intents without a threshold and lead to performance gains.

4.5 Ablation Study

We compare TFMN with two simplified versions, *-w/o T-free Cls* and *-w/o T-free Cls & IND task* in Table 2 to analyze the effectiveness of threshold-free intent classifier and IND task. We can see that as the threshold-free intent classifier is removed, the performances drop 0.9%, 2.5%, and 3.2% on slot F1, intent accuracy, and overall accuracy respectively. We attribute this to the fact that the threshold-free approach can better detect the intent number in an utterance compare to threshold strategy. We further remove the INP task and the performance again drops 0.8%, 0.5%, and 0.3% on slot F1, intent accuracy, and overall accuracy respectively. This indicates the effectiveness of introducing the INP task to multi-intent NLU.

4.6 Threshold-based vs Threshold-free

To compare threshold-based and threshold-free approaches, we evaluate how well a model can detect the number of intents in an utterance. The results are demonstrated in Table 3. We obtained that the threshold-free model, TFMN, significantly outperforms the threshold-based baselines. Our model achieves 2.1%, 5.1%, 11.8%, and 6.4% improvements on one to three intent utterances and average accuracy over GL-GIN. We find it interesting that threshold-based models predict intent number well when there is one intent in the utterance and become worse as the intent number increase while TFMN shows more consistency.

5 Conclusion

In this paper, we propose TFMN model which detects intent numbers in an utterance by a novel IND

task that does not require additional manual annotations. Then we propose a threshold-free intent classifier to detect multiple intents without relying on the threshold. Extensive experiments show that TFMN achieves performance gains over strong baselines.

Acknowledgements

This paper was partially supported by Shenzhen Science & Technology Research Program (No: GXWD20201231165807007-20200814115301001) and NSFC (No: 62176008). Special thanks are given to AOTO-PKUSZ Joint Research Center for AI for its support. Yuexian Zou is the corresponding author of this paper.

References

Lisong Chen, Peilin Zhou, and Yuexian Zou. 2022a. Joint multiple intent detection and slot filling via self-distillation. In *ICASSP 2022 - 2022 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 7612–7616.

Nuo Chen, Fenglin Liu, Chenyu You, Peilin Zhou, and Yuexian Zou. 2021a. Adaptive bi-directional attention: Exploring multi-granularity representations for machine reading comprehension. In *ICASSP*, pages 7833–7837. IEEE.

Nuo Chen, Linjun Shou, Min Gong, Jian Pei, and Daxin Jiang. 2021b. From good to best: Two-stage training for cross-lingual machine reading comprehension.

Nuo Chen, Linjun Shou, Ming Gong, Jian Pei, and Daxin Jiang. 2022b. Bridging the gap between language models and cross-lingual sequence labeling. In *Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 1909–1923, Seattle, United States. Association for Computational Linguistics.

Nuo Chen, Chenyu You, and Yuexian Zou. 2021c. Self-supervised dialogue learning for spoken conversational question answering. *arXiv preprint arXiv:2106.02182*.

Qian Chen, Zhu Zhuo, and Wen Wang. 2019. BERT for joint intent classification and slot filling. *CoRR*, abs/1902.10909.

A. Coucke, A. Saade, Adrien Ball, Théodore Bluche, A. Caulier, D. Leroy, Clément Doumouro, Thibault Gisselbrecht, F. Caltagirone, Thibaut Lavril, Maël Primet, and J. Dureau. 2018. Snips voice platform: an embedded spoken language understanding system for private-by-design voice interfaces. *ArXiv*.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, Minneapolis, Minnesota. Association for Computational Linguistics.

Haihong E, Peiqing Niu, Zhongfu Chen, and Meina Song. 2019. A novel bi-directional interrelated model for joint intent detection and slot filling. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 5467–5471, Florence, Italy. Association for Computational Linguistics.

Rashmi Gangadharaiah and Balakrishnan Narayanaswamy. 2019. Joint multiple intent detection and slot labeling for goal-oriented dialog. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 564–569, Minneapolis, Minnesota. Association for Computational Linguistics.

Chih-Wen Goo, Guang Gao, Yun-Kai Hsu, Chih-Li Huo, Tsung-Chieh Chen, Keng-Wei Hsu, and Yun-Nung Chen. 2018. Slot-gated modeling for joint slot filling and intent prediction. In *Proceedings of The 16th Annual Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*.

C. T. Hemphill, J. Godfrey, and G. Doddington. 1990. The atis spoken language systems pilot corpus. In *HLT*.

Ganesh Jawahar, Benoît Sagot, and Djamé Seddah. 2019. What does BERT learn about the structure of language? In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 3651–3657, Florence, Italy. Association for Computational Linguistics.

Andrew McCallum, Dayne Freitag, and Fernando CN Pereira. 2000. Maximum entropy markov models for information extraction and segmentation. In *Icml*.

Libo Qin, W. Che, Yangming Li, Haoyang Wen, and T. Liu. 2019. A stack-propagation framework with token-level intent detection for spoken language understanding. In *EMNLP/IJCNLP*.

Libo Qin, Fuxuan Wei, Tianbao Xie, Xiao Xu, Wanxiang Che, and Ting Liu. 2021. Gl-gin: Fast and accurate non-autoregressive model for joint multiple intent detection and slot filling.

Libo Qin, Xiao Xu, Wanxiang Che, and Ting Liu. 2020. AGIF: An adaptive graph-interactive framework for joint multiple intent detection and slot filling. In

"Findings of the Association for Computational Linguistics: EMNLP 2020". "Association for Computational Linguistics".

Anna Rogers, Olga Kovaleva, and Anna Rumshisky. 2020. A primer in BERTology: What we know about how BERT works. *Transactions of the Association for Computational Linguistics*, 8:842–866.

Ruhi Sarikaya, Geoffrey E Hinton, and Bhuvana Ramabhadran. 2011. Deep belief nets for natural language call-routing. In *ICASSP*. IEEE.

Ngoc Thang Vu. 2016. Sequential convolutional neural networks for slot filling in spoken language understanding. In *INTERSPEECH*.

Kaisheng Yao, Baolin Peng, Yu Zhang, Dong Yu, Geoffrey Zweig, and Yangyang Shi. 2014. Spoken language understanding using long short-term memory neural networks. In *2014 IEEE Spoken Language Technology Workshop (SLT)*. IEEE.

Chenyu You, Nuo Chen, Fenglin Liu, Dongchao Yang, and Yuexian Zou. 2020. Towards data distillation for end-to-end spoken conversational question answering. *CoRR*, abs/2010.08923.

Chenyu You, Nuo Chen, and Yuexian Zou. 2021a. Knowledge distillation for improved accuracy in spoken question answering. In *ICASSP 2021-2021 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 7793–7797. IEEE.

Chenyu You, Nuo Chen, and Yuexian Zou. 2021b. Self-supervised contrastive cross-modality representation learning for spoken question answering. In *Findings of the Association for Computational Linguistics: EMNLP 2021*, pages 28–39.

Peilin Zhou, Zhiqi Huang, Fenglin Liu, and Yuexian Zou. 2021. Pin: A novel parallel interactive network for spoken language understanding. *2020 25th International Conference on Pattern Recognition (ICPR)*, pages 2950–2957.

Unsupervised Multi-scale Expressive Speaking Style Modeling with Hierarchical Context Information for Audiobook Speech Synthesis

Xueyuan Chen[*]
Shenzhen International Graduate School
Tsinghua University
Shenzhen, China
chenxuey20@mails.tsinghua.edu.cn

Shun Lei
Shenzhen International Graduate School
Tsinghua University
Shenzhen, China
leis21@mails.tsinghua.edu.cn

Zhiyong Wu[†]
Shenzhen International Graduate School
Tsinghua University
Shenzhen, China
zywu@se.cuhk.edu.hk

Dong Xu
Tencent Music Entertainment Group
Shenzhen, China
huberyxu@tencent.com

Weifeng Zhao
Tencent Music Entertainment Group
Shenzhen, China
ethanzhao@tencent.com

Helen Meng
The Chinese University of Hong Kong
Hong Kong SAR, China
hmmeng@se.cuhk.edu.hk

Abstract

Naturalness and expressiveness are crucial for audiobook speech synthesis, but now are limited by the averaged global-scale speaking style representation. In this paper, we propose an unsupervised multi-scale context-sensitive text-to-speech model for audiobooks. A multi-scale hierarchical context encoder is specially designed to predict both global-scale context style embedding and local-scale context style embedding from a wider context of input text in a hierarchical structure. Likewise, a multi-scale reference encoder is introduced to extract reference style embeddings at both global and local scales from the reference speech, which are used to guide the prediction of speaking styles. On top of these, a bi-reference attention mechanism is used to align both local-scale reference style embedding sequence and local-scale context style embedding sequence with corresponding phoneme embedding sequence. Both objective and subjective experiment results on a real-world multi-speaker Mandarin novel audio dataset demonstrate the excellent performance of our proposed method over all baselines in terms of naturalness and expressiveness of the synthesized speech[1].

1 Introduction

Recently, some text-to-speech (TTS) models, such as Tacotron (Wang et al., 2017), Tacotron 2 (Shen et al., 2018), Deep voice (Ping et al., 2017), TransformerTTS (Li et al., 2019) have been proposed to generate speech autoregressively from text input, and can achieve performance very close to human quality. In order to increase inference speed and generate more robust speech, non-autoregressive TTS models such as FastSpeech (Ren et al., 2019) and FastSpeech 2 (Ren et al., 2020) are emerged with robust and fast parallel generation.

However, limited expressiveness of synthesized audio persists as one of the major gaps between synthesized speech and real human speech, which draws growing attention to expressive speech synthesis studies. Synthesizing long-form expressive datasets (such as audiobooks) is still a challenging task, since wide-ranging voice characteristics are collapsed into an averaged prosodic style. To address this issue, style transfer TTS has been a popular strategy in recent years (Skerry-Ryan et al., 2018). The global style token (GST) model (Wang et al., 2018) adopts multi-head attention mechanism and several learnable style tokens to extract the global style from reference audio in an unsupervised way. Further more, a hierarchical GST architecture is proposed to learn hierarchical embedding information implicitly, which contains several GST layers with residual connection (An et al., 2019).

[*] Work conducted when the first author was intern at Tencent Music Entertainment Group.

[†] Corresponding author.

[1] Synthesized speech samples are available at: https://thuhcsi.github.io/COLING2022-MSHCE-TTS

Proceedings of the 29th International Conference on Computational Linguistics, pages 7193–7202
October 12–17, 2022.

To not only learn to represent a wide range of speaking styles, but also synthesize expressive speech without the need of auxiliary inputs at inference time, some methods attempt to predict speaking style directly from text, which is more practical and flexible. The text-predicted global style token (TP-GST) extends the GST by predicting style embedding or style token weights from text only (Stanton et al., 2018). Considering that style and semantic information of sentences are closely related and Bidirectional Encoder Representations from Transformers (BERT) (Devlin et al., 2018) shows its effectiveness in language representation, pre-trained BERT embedding is adopted as an additional contextual information input to Tacotron 2 model to improve the pronunciation and expressiveness of the generated speech (Hayashi et al., 2019) (Fang et al., 2019). In addition, different ways of incorporating linguistic features and BERT-based features are investigated and compared in various application domains (news, chat and audiobooks). Results show that character embedding is the most effective one (Xiao et al., 2020). Some works also attempt to predict fine-grained speaking styles from text, such as word level (Zhang and Ling, 2021) and phoneme level (Lei et al., 2021).

All aforementioned methods only take the single sentence to be synthesized into consideration. Some studies demonstrate that considering a wider range of contextual information contributes to expressive speech synthesis (Tan et al., 2021) (Li et al., 2022). Recently, a hierarchical context encoder that considers adjacent contexts within a fixed-size sliding window is used to predict sentence-level style representation directly from text (Lei et al., 2022a). Although the overall performance is improved, it is still an averaged result that lacks some local fine-grained expressiveness information and rhythmic fluctuations such as pauses and emphasis.

The expressiveness of human speech can be perceived as a compound of multi-scale acoustic factors. One is the global-scale speech style, which includes but not limited to timbre and emotion of the speaker. Styles at this level are supposed to be consistent throughout the entire utterance. The other is the local-scale speech style, which consists of speed, energy, pitch, pause and other acoustic features (Li et al., 2021). Therefore, it is insufficient to model speech style from a single aspect. Some latest studies are observed to devote efforts to

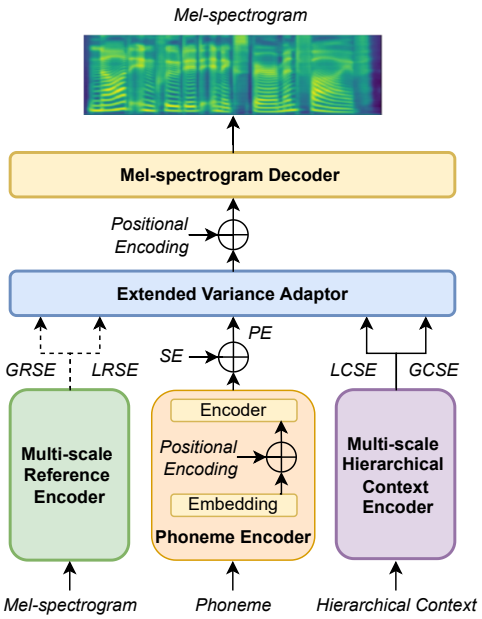

Figure 1: The overall architecture of the proposed model

performing multi-scale modeling on some specific tasks, in particular emotional speech synthesis (Lei et al., 2021) (Lei et al., 2022b). This further demonstrates the importance and necessity of multi-scale modeling.

With all listed imperfections taken into consideration, this paper proposes an unsupervised multi-scale expressive speech synthesis model for audiobooks with hierarchical context information as input. A multi-scale hierarchical context encoder is designed to predict both global-scale context style embedding (GCSE) and local-scale context style embedding (LCSE) from the context in a hierarchical structure. Meanwhile, a multi-scale reference encoder is introduced to extract both global-scale reference style embedding (GRSE) and local-scale reference style embedding (LRSE) from the reference speech, which are used to guide the prediction of speaking styles. On top of these, a bi-reference attention mechanism is adopted to align both the quasi-phoneme-level LRSE sequence and the character-level LCSE sequence with the corresponding phoneme embedding (PE) sequence. Both objective and subjective experiments on a real-world multi-speaker Mandarin novel audio dataset demonstrate the excellent performance of our proposed model over all baseline approaches in terms of naturalness and expressiveness of the synthesized speech. Ablation studies are further conducted to investigate the influences of several main modules in our proposed model.

2 Method

The architecture of our proposed model is illustrated in Figure 1. It consists of three major parts: a multi-scale reference encoder, a multi-scale hierarchical context encoder and a sequence-to-sequence expressive TTS system based on Fastspeech 2 (Ren et al., 2020) with extended variance adaptor. The multi-scale reference encoder is used to extract reference style embeddings at both global and local scales (i.e. GRSE and LRSE) from reference speech. While the multi-scale hierarchical context encoder is used to predict the context style embeddings at global and local styles (i.e. GCSE and LCSE) from hierarchical context. The extended variance adaptor is used to align and fuse the style embeddings at different scales with the phoneme embeddings.

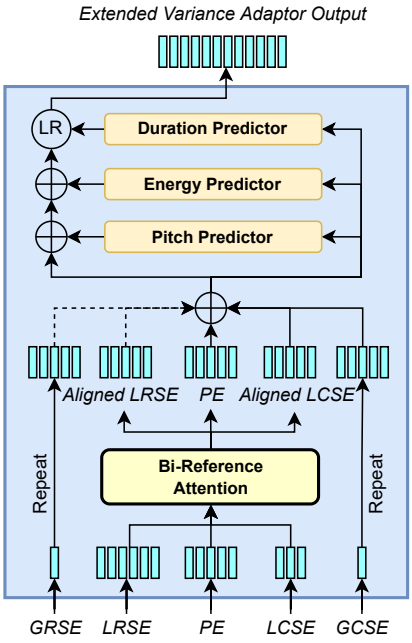

Figure 2: Extended Variance Adaptor

2.1 Multi-scale expressive TTS system

We adopt FastSpeech 2 as the basic acoustic model, of which the phoneme encoder and mel-spectrogram decoder keep the original structure as described in (Ren et al., 2020). On the basis, speaker embedding (SE) is added to the phoneme encoder output to support different timbres. After that, the phoneme embedding (PE) together with GRSE, LRSE, GCSE, LCSE are fed into the extended variance adaptor with several changes as illustrated in Figure 2.

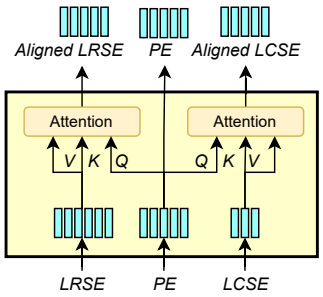

Figure 3: Bi-Reference Attention

2.1.1 Extended variance adaptor

Firstly, the global-scale style inputs of GRSE and GCSE are repeated to the same length as PE. At the same time, the local-scale style inputs of LRSE and LCSE are aligned to phoneme-level sequences by a bi-reference attention mechanism. After that, either the repeated GRSE and aligned LRSE or the repeated GCSE and aligned LCSE are added to PE at different stages, and then the result is passed to the variance predictors containing duration predictor, pitch predictor and energy predictor. Unlike FastSpeech 2, the length regulator is moved after the variance predictors, in order to predict variations at phoneme level rather than frame level, which has been proved to be able to further improve speech quality (Łańcucki, 2021).

2.1.2 Bi-reference attention mechanism

For expressive speech synthesis, fine-grained style embedding sequence is usually reformed into sequence with the same length as phoneme embedding sequence. Inspired by (Lee and Kim, 2019), we propose a bi-reference attention mechanism to align both the quasi-phoneme-scale LRSE sequence and character-scale LCSE sequence with the phoneme-level PE sequence. As shown in Figure 3, the bi-reference attention consists of two scaled dot-product attentions (Vaswani et al., 2017) with the same query input and two groups of different key and value inputs. Here, the phoneme-level PE sequence is fed as the query of the bi-reference attention. Meanwhile, the quasi-phoneme-scale LRSE sequence and the character-scale LCSE sequence are fed as two groups of key and value inputs respectively. Finally, the bi-reference attention outputs the aligned LRSE sequence and aligned LCSE sequence with the same length as the PE sequence. This operation not only could align sequences of different lengths to the phoneme level, but also reduces the difficulty of local style guid-

ance from the multi-scale reference encoder to the multi-scale hierarchical context encoder.

2.2 Multi-scale reference encoder

Inspired by the success of multi-scale emotion transfer task (Li et al., 2021), we introduce a multi-scale reference encoder to extract both the global-scale and local-scale style embeddings from the reference speech. As shown in Figure 4, the multi-scale reference encoder is made up by a stack of 6 convolution layers and 2 scale-specific extractor layers.

For the convolution layers, 1-D convolution along temporal dimension is adopted to help the bi-reference attention to learn the alignment between the output LRSE sequence and the PE sequence. Each of the convolution layers is composed by 3×1 filters, ReLU activation and batch normalization (Ioffe and Szegedy, 2015). In particular, in order to regulate the temporal granularity of the convolution output closer to human vocal perception, the filter strides of 6 convolution layers are set as [2, 1, 2, 1, 2, 2]. This downsampling operation ensures that after the convolution stack, temporal granularity of the intermediate frame-level feature sequence is properly reformed to a quasi-phoneme-scale.

For the global style extractor layer, it is made up of consecutive Gated Recurrent Unit (GRU) layer and global style token (GST) layer (Wang et al., 2018). The GST layer adopts a multi-head attention mechanism and several learnable style tokens to extract the global styles in an unsupervised way. For the local style extractor layer, it consists of GRU layer and full-connected layer with tanh activation. Both scale-specific extractor layers take the above quasi-phoneme-scale feature sequence as input. However, inside the global style extractor layer, only the final state of GRU is fed to the GST layer. The GRSE output by the global style extractor is forced to be a latent sentence-level style embedding vector. On the other hand, the LRSE output by the local style extractor is a quasi-phoneme-scale style embedding sequence.

2.3 Multi-scale hierarchical context encoder

To improve expressiveness and naturalness of synthesized speech, we introduce a dedicated multi-scale hierarchical context encoder to predict both global-scale style embedding and local-scale style embedding from the hierarchical context within a fixed size sliding window. As shown in Figure 5, the multi-scale hierarchical context encoder consists of three components: context embedding layer, global style predictor and local style predictor.

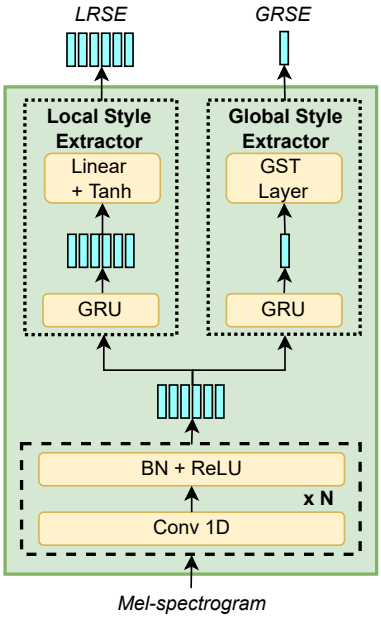

Figure 4: Multi-scale Reference Encoder

2.3.1 Context embedding layer

Let l be the number of sentences considered in the past or future context within the sliding window. U_0 is defined as the current sentence to be synthesized. U_{-l}, U_{1-l}, ..., U_{-1} and U_1, U_2, ..., U_l are the past and future sentences respectively. All these $2l + 1$ sentences are firstly embedded with a well-pretrained character-level BERT model (Devlin et al., 2018) that is composed of a stack of Transformer blocks and pretrained with a huge amount of Chinese text data. Thereafter, role embedding is added to the output of BERT embedding layer to consider the interactions between different roles. Then Bidirectional GRU (BiGRU) is further used to obtain the character-level context-sensitive embedding S_i for each input sentence U_i $(-l \leq i \leq l)$, which can be discribed as:

$$S_i = CEmb(U_i), \tag{1}$$

where $CEmb(\cdot)$ is the operation of context embedding layer.

2.3.2 Global style predictor

The global style predictor contains two levels of attention networks, intra-sentence and inter-sentence respectively.

The intra-sentence attention network is used to obtain a sentence-level representation based on

each character and inter-character relations within a sentence. As not all characters contribute equally to the global meaning of the sentence, a scaled dot-product attention is adopted to calculate the weights of each character and aggregate them into a global sentence-level vector G_i for each character-level embedding S_i of a sentence, where S_i is fed as the key and value. The query Q_1 is a 256-dim vector, which is randomly initialized and learnable during the training. It can be seen as a high level representation of a fixed query "to what extent does the character influence the global speaking style". This can be formulated as:

$$K_i = S_i W_k, \tag{2}$$

$$V_i = S_i W_v, \tag{3}$$

$$G_i = A(Q_1, K_i, V_i) = softmax(\frac{Q_1 K_i^\top}{\sqrt{d_{Q_1}}})V_i, \tag{4}$$

where W_k and W_v are linear projection matrices of attention keys and values. d_{Q_1} means the dimension of the query Q_1.

For the inter-sentence attention network, $2l+1$ sentence-level vectors $G_{-l}, ..., G_0, ..., G_l$ obtained by the intra-sentence network are firstly concatenated along temporal dimension to form a long new sequence G with the length of $2l+1$, which is further fed to BiGRU to model the correlations among sentences. After that, another scaled dot-product attention is used to predict a global-scale speaking style based on sentence-level embedding G and inter-sentence relations within the hierarchical context. Here, G is the key and value, while query Q_2 is a randomly initialized and learnable 256-dim vector similar to Q_1. Finally, the inter-sentence attention network outputs the GCSE of the current sentence U_0.

2.3.3 Local style predictor

The local style predictor is used to obtain the local-scale style embedding of current sentence from the hierarchical context embeddings.

Firstly, a scaled dot-product attention is used to align each character-level embedding S_i of a sentence in the hierarchical context with the character-level embedding S_0 of the current sentence, where S_i is fed as the key and value,S_0 is fed as the query. Here, S_0 can be seen as a fixed local-scale query "to what extent does each character influence the local speaking style of the current sentence" for each

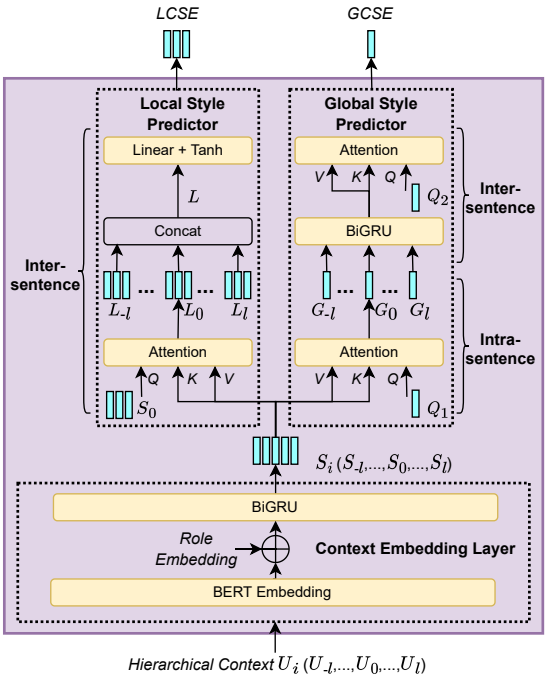

Figure 5: Multi-scale Hierarchical Context Encoder

sentence in the context. This can be formulated as:

$$K_i = S_i W_k, \tag{5}$$

$$V_i = S_i W_v, \tag{6}$$

$$L_i = A(S_0, K_i, V_i) = softmax(\frac{S_0 K_i^\top}{\sqrt{d_{S_0}}})V_i, \tag{7}$$

where W_k and W_v are linear projection matrices of attention keys and values. d_{S_0} means the dimension of the query S_0.

Since all the $2l+1$ attention outputs $L_{-l}, ..., L_0, ..., L_l$ have the same length as S_0, a concatenation operation along feature dimension is further implemented, followed by full-connected layer with tanh activation:

$$L = Concat_f(L_{-l}, ..., L_l), \tag{8}$$

In this way, the LCSE of current sentence is obtained.

2.4 Training strategy and inference procedure

During training, to encourage the multi-scale hierarchical context encoder learn style representation better, the proposed model is trained with knowledge distillation strategy in three stages.

i) In the first stage, the acoustic model and the multi-scale reference encoder are jointly trained

with paired <utterance, speech> data to get a well-trained multi-scale reference encoder in an unsupervised way. In order to better extract style features at different scales, this training stage is divided into two steps. Firstly, only the global style extractor is inserted into the model to obtain the well-represented global-scale features. After that, the local style extractor is further involved, which can extract more fine-grained local style representations. The multi-scale style embeddings extracted from all speeches in the training set can be regarded as ground-truth speaking style representations.

ii) In the second stage, knowledge distillation strategy is used to transfer the knowledge from the multi-scale reference encoder to the multi-scale hierarchical context encoder. That is, we use ground-truth style embeddings GRSE and aligned LRSE as targets to guide the prediction of speaking style representations GCSE and aligned LCSE from context, for training the multi-scale hierarchical context encoder.

iii) In the third stage, the acoustic model and the multi-scale hierarchical context encoder are jointly trained with a lower learning rate to further improve the expressiveness of the synthesized speech.

During inference, the multi-scale reference encoder is abandoned. Only GCSE and LCSE predicted from the multi-scale hierarchical context encoder are fed to variance adaptor together with PE. Finally, by accepting input text and hierarchical context, the model can synthesize speech with more expressive styles.

3 Experiment

3.1 Dataset and model details

An internal multi-speaker novel audio corpus on Mandarin is employed in our experiment. It contains more than 40 roles and around 15 hours speech spoken by 5 Mandarin native speakers with quite different timbres. The speaking styles vary among roles and utterances, and the speed, pitch and energy fluctuate greatly in an utterance. The dataset has a total of 11,000 audio clips, of which 200 clips are used for validation and 100 clips for test, and the rest for training.

For feature extraction, we transform the raw waveforms into 80-dim mel-spectrograms with sampling rate 16kHz, frame size 1200 and hop size 240. An open-source pre-trained Chinese character-level BERT model[2] with frozen parameters is used in our experiments. The context of current sentence is made up of its two past sentences, two future ones, and itself.

We take 200k steps to train the acoustic model and multi-scale reference encoder, where 100k steps are for global style extractor and the remaining 100k steps for local style extractor. Then we take 20k steps to train the multi-scale hierarchical context encoder and 20k steps to adapt the acoustic model and the multi-scale hierarchical context encoder. All the trainings are conducted with a batch size of 16 on a NVIDIA A100 GPU. The Adam optimizer is adopted with $\beta_1 = 0.9$, $\beta_2 = 0.98$. The warm-up strategy is used before 4000 steps. In addition, a well-trained HiFi-GAN (Kong et al., 2020) is used as the vocoder to generate waveform.

3.2 Compared methods

Three FastSpeech 2 based models are implemented for comparison, and the details are described as follows:

FastSpeech 2: Original FastSpeech 2 (Ren et al., 2020) with minor changes on the variance predictor to be consistent to the proposed model as described in section 2.1.1.

BERT-FS 2: Inspired by (Xiao et al., 2020), we set an end-to-end TTS model by combining BERT with FastSpeech 2, which contains a plain context encoder and only considers the current sentence. The same character embeddings obtained from BERT are directly passed to a GRU layer whose final state is used as a style embedding.

HCE-FS 2: It uses a reference encoder to extract the global style representation, and uses a hierarchical context encoder to predict the global style embedding, which is fed to the variance adaptor of FastSpeech 2 (Lei et al., 2022a).

Model	S-MOS	P-MOS
Ground Truth	4.705 ± 0.067	4.737 ± 0.073
FastSpeech 2	3.426 ± 0.091	3.432 ± 0.099
BERT-FS 2	3.503 ± 0.096	3.526 ± 0.086
HCE-FS 2	3.589 ± 0.089	3.613 ± 0.073
Proposed	$\mathbf{4.031 \pm 0.068}$	$\mathbf{4.142 \pm 0.071}$

Table 1: The sentence-MOS (S-MOS) and paragraph-MOS (P-MOS) of different models with 95% confidence intervals for subjective evaluation.

[2] https://huggingface.co/bert-base-chinese

Model	F0 RMSE	Energy RMSE	Duration MSE	MCD
FastSpeech 2	70.847	13.228	0.1316	7.198
BERT-FS 2	67.912	13.031	0.1263	7.134
HCE-FS 2	64.975	12.449	0.1254	6.975
Proposed	**62.471**	**11.683**	**0.1224**	**6.843**

Table 2: Objective evaluations for different models.

3.3 Subjective evaluation

We conduct the mean opinion score (MOS) tests to evaluate the naturalness and expressiveness of the generated speeches. As the task in this paper is paragraph-level audiobook speech synthesis, we conduct two kinds of MOS tests, sentence-MOS (S-MOS) and paragraph-MOS (P-MOS) respectively. The former mainly focuses on the naturalness and expressiveness of the synthesized speech considering only the current single sentence. The latter focuses on the coherence of speech styles of the current sentence in a paragraph considering the context of past and future sentences, where the speeches of the past and future sentences are the resynthesized version of ground truth speech recordings. 10 single sentences and 10 short paragraphs are randomly selected in the test set. 25 native Chinese speakers are asked to listen to the generated speeches and rate on a scale from 1 to 5 with 1 point interval.

As shown in Table 1, our proposed approach achieves the best S-MOS of **4.031** and P-MOS of **4.142**. The results demonstrate the effectiveness of our proposed methods over all the baselines especially on the paragraph level. There is a big gap between **FastSpeech 2** and **Ground Truth**, indicating that it is difficult to model the multiple speech variations without enough input information.

The ABX preference tests are also conducted on our proposed model and each of the three baselines respectively. Similarly, we also conduct two kinds of ABX preference tests, sentence-ABX (S-ABX) and paragraph-ABX (P-ABX) respectively. The same 25 subjects are asked to choose a preferred speech in terms of naturalness and expressiveness between a pair of methods.

As shown in Figure 6, the preference rate of our proposed model exceeds **FastSpeech 2** by 53.2%, **BERT-FS 2** by 48.4% and **HCE-FS 2** by 45.8% on the S-ABX preference test. Moreover, the gaps between our proposed model and baseline models are more reflected on the preference rate of P-ABX, which are 70.5%, 66.0% and 60.0% respectively.

Both MOS and ABX preference tests demonstrate that our proposed method significantly outperforms the baselines in terms of naturalness and expressiveness especially for the paragraph-level speech synthesis tasks.

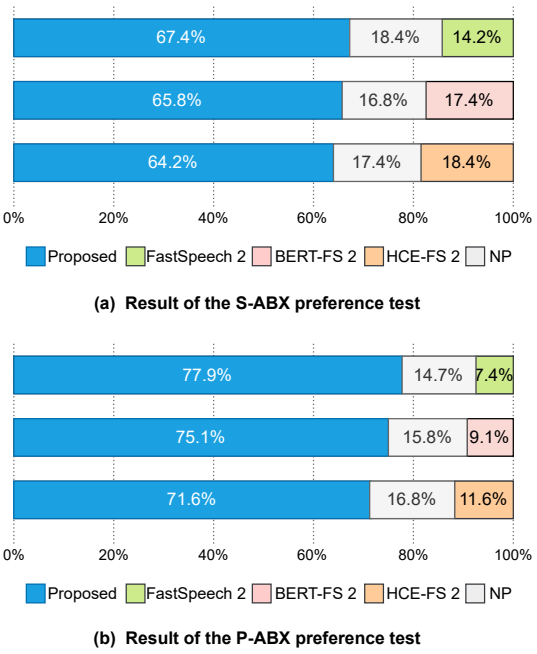

Figure 6: Results of the sentence-ABX (S-ABX) and paragraph-ABX (P-ABX) preference tests. NP means no preference.

3.4 Objective evaluation

As the common for the objective evaluation of synthesized speech, we employ the root mean square error (RMSE) of pitch and energy, the mean square error (MSE) of duration and mel cepstral distortion (MCD) as the objective evaluation metrics. Specifically, the dynamic time warping (DTW) is firstly used to construct the alignment paths between the ground-truth mel-spectrogram and the predicted one. After that, the F0 sequence and energy sequence are aligned towards ground-truth following the DTW path. We also utilize DTW to compute the minimum MCD by aligning the two sequences. Here, MCD is utilized to calculate the difference

between the mel-spectrograms of the synthesized speech and the ground truth. For duration, we compute the MSE between the predicted duration and ground-truth duration.

As shown in Table 2, our proposed model achieves **62.471** for F0 RMSE, **11.683** for Energy RMSE, **0.1224** for Duration MSE and **6.843** for MCD, which outperforms all the baselines on all metrics. This excellent results indicate that our proposed model can predict more accurate style features, such as duration, pitch and energy, than baselines.

3.5 Ablation study

To further investigate the influence of several main modules in our proposed model, we have tried four other settings based on the proposed method:

i) **Proposed - Global-scale Style**: The global style extractor in multi-scale reference encoder and the global style predictor in multi-scale hierarchical context encoder are removed, and only LRSE, LCSE together with PE are fed to variance adaptor.

ii) **Proposed - Local-scale Style**: The local style extractor in multi-scale reference encoder and the local style predictor in multi-scale hierarchical context encoder are removed, and only GRSE, GCSE together with PE are fed to variance adaptor.

iii) **Proposed - Knowledge Distillation**: The knowledge distillation strategy is abandoned, by removing the multi-scale reference encoder. The predicted GCSE and LCSE from multi-scale hierarchical context encoder together with PE are fed to variance adaptor directly throughout the training process.

iv) **Proposed - Role Embedding** The role embedding of context embedding layer in the multi-scale hierarchical context encoder is removed.

Comparison mean opinion score (CMOS) is employed to compare the synthesized speeches in terms of naturalness and expressiveness. The results are shown in Table 3.

Model	CMOS
Proposed	0
- Global-scale Style	-0.174
- Local-scale Style	-0.211
- Knowledge Distillation	-0.189
- Role Embedding	-0.153

Table 3: CMOS comparision for ablation study.

Compared with the proposed method, the performance of the four settings removing different main modules is degraded to various degrees respectively. This indicates that all these components have substantial impact on our proposed model. When the global-scale style is removed, the overall style of a sentence lacks expressiveness and there will be a obvious deviation from the ground truth. The proposed model without local-scale style leads to significant performance degradation in some rhythmic aspects, especially pauses and stress. When the knowledge distillation strategy is abandoned, the model needs to predict both the global-scale and local-scale style from context directly without any guidance from the reference encoder. The performance degradation demonstrates the necessity of the multi-scale reference encoder and indicates that learning the speaking style representation from context in an explicit way is more suitable for this style prediction task. The proposed model without the role embedding also causes some performance degradation, which further indicates that considering the interactions between different roles is crucial for the style prediction of novel speech synthesis.

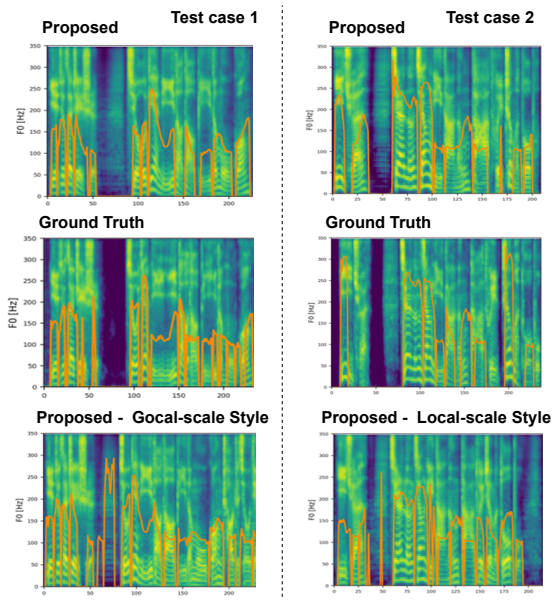

Figure 7: Mel-spectrograms and pitch contours of the speeches for the two test cases.

3.6 Case study

To further explore the impact of multi-scale speaking style on the expressiveness and naturalness of synthesized speech, a case study is conducted to synthesize two example utterances in test set with the operation of removing the global-scale style

and local-scale style respectively. The speech synthesized by our proposed model and the ground truth are also provided for reference. The mel-spectrograms and pitch contours of speeches are shown in Figure 7.

When the global-scale style is removed, the pitch fluctuations become larger than others, and the overall style and speed of synthesized speech vary greatly compared with the ground truth. When the local-scale style is removed, some local style characteristics of the synthesized speech are lost, resulting in a relatively averaged style throughout the whole sentence. Compared with the two single-scale results, the speeches synthesized by our proposed multi-scale model are more similar to the ground-truth speech in terms of the overall and fine-grained style properties, such as the trend of intonation and stress patterns. The results demonstrate that modelling the speaking style from hierarchical context information in a multi-scale way is essential and effective to improve the naturalness and expressiveness of the synthesized speech.

4 Conclusion

In this paper, we propose an unsupervised multi-scale context-sensitive text-to-speech model for audiobooks. A multi-scale hierarchical context encoder is designed to predict both the global-scale context style embedding and local-scale context style embedding from hierarchical context with the guidance of a multi-scale reference encoder. Both objective and subjective experiment results on a real-world multi-speaker Mandarin novel audio dataset demonstrate the excellent performance of our proposed multi-scale context-sensitive model over all baseline approaches in terms of naturalness and expressiveness of the synthesized speech.

Acknowledgement: This work is supported by National Natural Science Foundation of China (62076144), the Major Key Project of PCL (PCL2021A06, PCL2022D01), Shenzhen Key Laboratory of next generation interactive mediainnovative technology (ZDSYS20210623092001004), CCF-Tencent Open Research Fund (CCF-Tencent RAGR20210122) and Tencent AI Lab Rhino-Bird Focused Research Program (RBFR2022005).

References

Xiaochun An, Yuxuan Wang, Shan Yang, Zejun Ma, and Lei Xie. 2019. Learning hierarchical representations for expressive speaking style in end-to-end speech synthesis. In *2019 IEEE Automatic Speech Recognition and Understanding Workshop (ASRU)*, pages 184–191. IEEE.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*.

Wei Fang, Yu-An Chung, and James Glass. 2019. Towards transfer learning for end-to-end speech synthesis from deep pre-trained language models. *arXiv preprint arXiv:1906.07307*.

Tomoki Hayashi, Shinji Watanabe, Tomoki Toda, Kazuya Takeda, Shubham Toshniwal, and Karen Livescu. 2019. Pre-trained text embeddings for enhanced text-to-speech synthesis. In *INTERSPEECH*, pages 4430–4434.

Sergey Ioffe and Christian Szegedy. 2015. Batch normalization: Accelerating deep network training by reducing internal covariate shift. In *International conference on machine learning*, pages 448–456. PMLR.

Jungil Kong, Jaehyeon Kim, and Jaekyoung Bae. 2020. Hifi-gan: Generative adversarial networks for efficient and high fidelity speech synthesis. *Advances in Neural Information Processing Systems*, 33:17022–17033.

Adrian Łańcucki. 2021. Fastpitch: Parallel text-to-speech with pitch prediction. In *ICASSP 2021-2021 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 6588–6592. IEEE.

Younggun Lee and Taesu Kim. 2019. Robust and fine-grained prosody control of end-to-end speech synthesis. In *ICASSP 2019-2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 5911–5915. IEEE.

Shun Lei, Yixuan Zhou, Liyang Chen, Zhiyong Wu, Shiyin Kang, and Helen Meng. 2022a. Towards expressive speaking style modelling with hierarchical context information for mandarin speech synthesis. In *ICASSP 2022-2022 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE.

Yi Lei, Shan Yang, Xinsheng Wang, and Lei Xie. 2022b. Msemotts: Multi-scale emotion transfer, prediction, and control for emotional speech synthesis. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*.

Yi Lei, Shan Yang, and Lei Xie. 2021. Fine-grained emotion strength transfer, control and prediction for emotional speech synthesis. In *2021 IEEE Spoken Language Technology Workshop (SLT)*, pages 423–430. IEEE.

Jingbei Li, Yi Meng, Chenyi Li, Zhiyong Wu, Helen Meng, Chao Weng, and Dan Su. 2022. Enhancing

speaking styles in conversational text-to-speech synthesis with graph-based multi-modal context modeling. In *ICASSP 2022-2022 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 7917–7921. IEEE.

Naihan Li, Shujie Liu, Yanqing Liu, Sheng Zhao, and Ming Liu. 2019. Neural speech synthesis with transformer network. In *Proceedings of the AAAI Conference on Artificial Intelligence*, 01, pages 6706–6713.

Xiang Li, Changhe Song, Jingbei Li, Zhiyong Wu, Jia Jia, and Helen Meng. 2021. Towards multi-scale style control for expressive speech synthesis. *arXiv preprint arXiv:2104.03521*.

Wei Ping, Kainan Peng, Andrew Gibiansky, Sercan Ömer Arik, Ajay Kannan, Sharan Narang, Jonathan Raiman, and John Miller. 2017. Deep voice 3: 2000-speaker neural text-to-speech.

Yi Ren, Chenxu Hu, Xu Tan, Tao Qin, Sheng Zhao, Zhou Zhao, and Tie-Yan Liu. 2020. Fastspeech 2: Fast and high-quality end-to-end text to speech. In *International Conference on Learning Representations*.

Yi Ren, Yangjun Ruan, Xu Tan, Tao Qin, Sheng Zhao, Zhou Zhao, and Tie-Yan Liu. 2019. Fastspeech: Fast, robust and controllable text to speech. *arXiv preprint arXiv:1905.09263*.

Jonathan Shen, Ruoming Pang, Ron J Weiss, Mike Schuster, Navdeep Jaitly, Zongheng Yang, Zhifeng Chen, Yu Zhang, Yuxuan Wang, Rj Skerrv-Ryan, et al. 2018. Natural tts synthesis by conditioning wavenet on mel spectrogram predictions. In *2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 4779–4783. IEEE.

RJ Skerry-Ryan, Eric Battenberg, Ying Xiao, Yuxuan Wang, Daisy Stanton, Joel Shor, Ron Weiss, Rob Clark, and Rif A Saurous. 2018. Towards end-to-end prosody transfer for expressive speech synthesis with tacotron. In *international conference on machine learning*, pages 4693–4702. PMLR.

Daisy Stanton, Yuxuan Wang, and RJ Skerry-Ryan. 2018. Predicting expressive speaking style from text in end-to-end speech synthesis. In *2018 IEEE Spoken Language Technology Workshop (SLT)*, pages 595–602. IEEE.

Xu Tan, Tao Qin, Frank Soong, and Tie-Yan Liu. 2021. A survey on neural speech synthesis. *arXiv preprint arXiv:2106.15561*.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in neural information processing systems*, pages 5998–6008.

Yuxuan Wang, RJ Skerry-Ryan, Daisy Stanton, Yonghui Wu, Ron J Weiss, Navdeep Jaitly, Zongheng Yang, Ying Xiao, Zhifeng Chen, Samy Bengio, et al. 2017. Tacotron: Towards end-to-end speech synthesis. *arXiv preprint arXiv:1703.10135*.

Yuxuan Wang, Daisy Stanton, Yu Zhang, RJ-Skerry Ryan, Eric Battenberg, Joel Shor, Ying Xiao, Ye Jia, Fei Ren, and Rif A Saurous. 2018. Style tokens: Unsupervised style modeling, control and transfer in end-to-end speech synthesis. In *International Conference on Machine Learning*, pages 5180–5189. PMLR.

Yujia Xiao, Lei He, Huaiping Ming, and Frank K Soong. 2020. Improving prosody with linguistic and bert derived features in multi-speaker based mandarin chinese neural tts. In *ICASSP 2020-2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 6704–6708. IEEE.

Ya-Jie Zhang and Zhen-Hua Ling. 2021. Extracting and predicting word-level style variations for speech synthesis. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 29:1582–1593.

Incorporating Instructional Prompts into A Unified Generative Framework for Joint Multiple Intent Detection and Slot Filling

Yangjun Wu, Han Wang, Dongxiang Zhang*, Gang Chen, Hao Zhang
Zhejiang University
{yjwu, zh}@zjuici.com, {22021066, zhangdongxiang,cg }@zju.edu.cn

Abstract

The joint multiple Intent Detection (ID) and Slot Filling (SF) is a significant challenge in spoken language understanding. Because the slots in an utterance may relate to multi-intents, most existing approaches focus on utilizing task-specific components to capture the relations between intents and slots. The customized networks restrict models from modeling commonalities between tasks and generalization for broader applications. To address the above issue, we propose a **U**nified **G**enerative framework (UGEN) based on a prompt-based paradigm, and formulate the task as a question-answering problem. Specifically, we design 5-type templates as instructional prompts, and each template includes a question that acts as the driver to teach UGEN to grasp the paradigm, options that list the candidate intents or slots to reduce the answer search space, and the context denotes original utterance. Through the instructional prompts, UGEN is guided to understand intents, slots, and their implicit correlations. On two popular multi-intent benchmark datasets, experimental results demonstrate that UGEN achieves new SOTA performances on full-data and surpasses the baselines by a large margin on 5-shot (28.1%) and 10-shot (23%) scenarios, which verify that UGEN is robust and effective. Our code will be publicly available at https://github.com/Young1993/UGEN

1 Introduction

In task-oriented dialogue systems, spoken language understanding (SLU) is a crucial component that aims to understand users' queries and use a semantic frame to represent users' requirements. The semantic frame usually contains intents and slot names (Tur and De Mori, 2011). Recently, multiple intent SLU has attracted lots of attention (Liu and Lane, 2016; E et al., 2019; Weld et al., 2021; Gangadharaiah and Narayanaswamy, 2019) due to the wide variety of practical application scenarios.

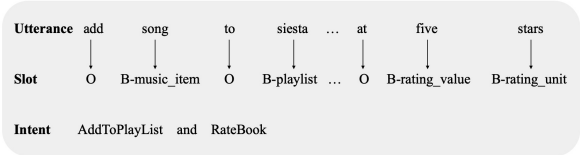

Figure 1: The semantic frame. An example from MixSNIPS dataset(Coucke et al., 2018; Qin et al., 2020).

Considering the example shown in Figure 1, the models are expected to identify the intents (AddToPlayList and RateBook) and the slot values with tags for the utterance. Current works (Qin et al., 2019, 2020; Ding et al., 2021; Qin et al., 2021; Chen et al., 2021a) usually treat Intent Detection (ID) as a classification task and Slot Filling (SF) as a sequence labeling task. The task-specific components are employed by current works to capture the connection or interaction between ID and SF, which achieve fine-grained multi-intent information integration for slot filling and obtain remarkable success.

In this paper, we're interested in exploiting a united paradigm to handle the task instead of customized networks. Prompt-learning (Liu et al., 2021; Jin et al., 2022) is a novel paradigm, which replaces the "pre-train, fine-tune" procedure with "pre-train, prompt, and predict" analogous to original pre-training language models (PLMs). With the help of a prompt template, prompt-learning benefits from fully exploiting the latent knowledge in PLMs while relieving the dependency on annotated data. Thus, prompt-based PLMs perform excellently in different tasks (classification, NER, summarization, etc.) and the few-shot setting.

To this end, we treat the joint multiple ID_SF as a question-answering problem and present a simple unified generative framework (UGEN) based on instructional prompts. Briefly, we first define 5-type descriptive templates (shown in Figure 2)

* Corresponding author

Proceedings of the 29th International Conference on Computational Linguistics, pages 7203–7208
October 12–17, 2022.

as inputs. Per template contains one context that refers to the original utterance, one question (e.g., "what are the intents of the sentence according to options?") as the driver to direct UGEN to realize the paradigm, and the corresponding options (e.g., play music, rate book) to restraint the answer search space. Through these instructional prompts, UGEN is directed to acquire the ability to capture the relationship between intents and slots. Then the correct intents and slots are predicted as the final answer (e.g., "add to playlist, rate book").

Experiments on two multi-intent benchmarks show that UGEN outperforms the baselines and achieves new SOTA performances. Remarkably, UGEN exceeds the comparison models by a large margin (28.1%, 23%, and 5.1%) in the 5/10-shot settings and 10% training data. The further analyses demonstrate that our approach has a strong ability of robustness and generalization. Meanwhile, it has the advantage of fast adaptation to practical scenario with limited annotation data and easy reproduction without task-specific components.

2 Related Work

Prompt-based Learning. With the release of GPT-3 (Brown et al., 2020), prompt-based learning methods have attracted more and more attention (Gu et al., 2021; Jin et al., 2022). The new paradigm can utilize the pre-trained language models with the form of cloze-style template, such as "I love this movie. It was a [Z] movie", and the model generates the probability of the [Z] in (good/bad). Hence, it directly models the probability of text $P(x|\theta)$ itself and uses the probability to predict y instead of the $P(y|x;\theta)$ [1] like traditional methods, which can narrow down the gap between pre-training and fine-tuning.

Few-shot Learning (FSL) with PLMs. FSL aims to absorb experience from only a few samples and make a great adaptation to the new problem(Wang et al., 2019). Usually, the models for FSL are trained on one accessible set of source domains and then evaluated on another set of unseen target domains. As the pre-trained models become more and more powerful, prompt-based methods with PLMs have achieved substantial improvements compared to those fine-tuned in low-resource settings, which displays promising prospects for few-shot learning in natural language tasks (Han et al., 2021; Li et al., 2021; Chen et al., 2021b).

3 Methodology

In this section, we briefly illustrate the problem definition of multiple ID_SF and main architecture. Then, we discuss the design of instruction-based templates and how to convert the ID_SF to the generation task.

3.1 Problem definition

The task of multiple ID_SF aims to classify all the possible intents and identify the slot values with the corresponding slot names in a given sentence. Given the input sentence $X = \{w_1, w_2, ..., w_n\}$, n is the length of X. The candidate intents $I = \{i_1, i_2, ..., i_m\}$, and m is the number of categories. The slot names $S = \{s_1, s_2, ..., s_k\}$, and k is the number of slot types.

To pursue simple model architecture (shown in Figure 2), in this work, we employ T5 (Raffel et al., 2020) as our backbone to model the probability of text $P(X|\theta)$. The answers Y are generated by UGEN, which contain intents (e.g., i_1, i_k) or slots (e.g., $\{w_1, w_2\}$ is one s_2), split by comma.

3.2 Instructional templates

To formulate the joint ID_SF as a question-answering problem and better exploit the knowledge learned in the PLMs, we design 5-type templates in line with QA and the pre-training-style tasks. Specifically, each template is defined to comprise three units: (1) **Context**, the original sentence X to express users' queries. (2) **Question**, the role of question Q is to guide the model to understand the paradigm and then generate the corresponding answer for the given Context X. In this study, the questions involve 5 types (shown in Figure 2): Question-1 is about the intents classification while the others are slot-related. For instance, question-1, "What are the intents of the sentence according to options?" is directed to intents labels. (3) **Options** O list all the intents labels or slot names as the candidate choices, and they act as a constraint to teach the model to select words in limited space (template's content).

Since the number of slot types are usually far larger than intents', we introduce 4-type questions to enhance the attention for slots. Specifically, Question-2 (e.g. Which words are the slot values in the sentence? for the context "Add this track to

[1] Here, we take the input x, learn the model parameters θ , and predict the output y.

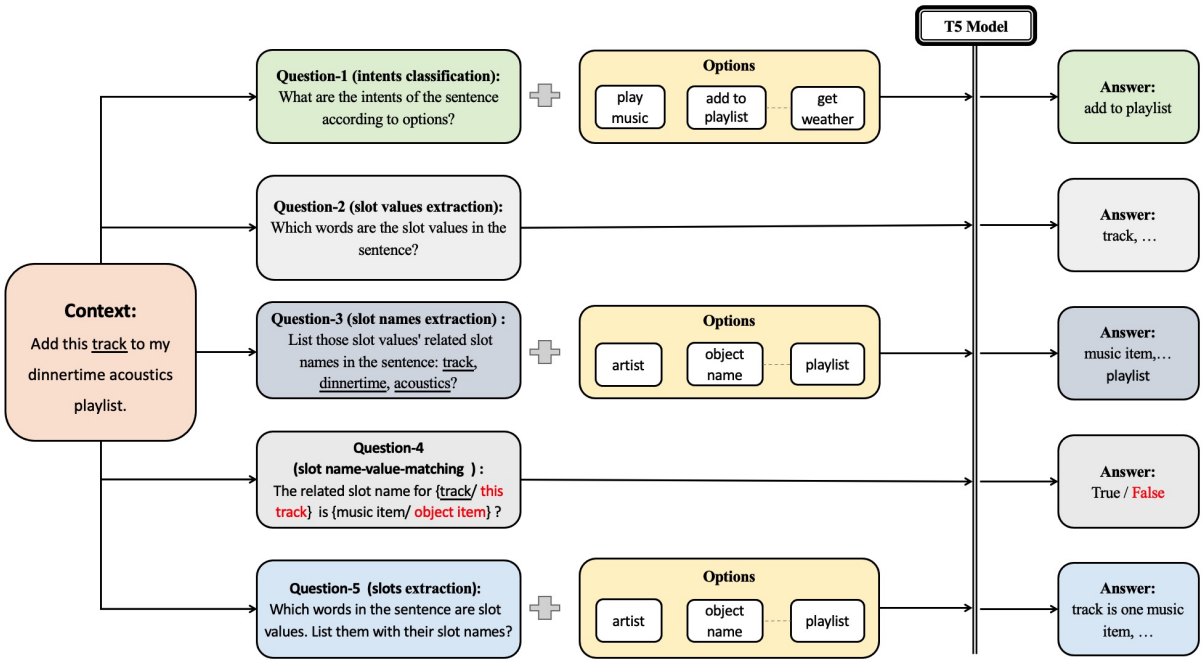

Figure 2: UGEN architecture with 5-type prompt templates based on combination of context, question, and options. For the Question-4, those words marked in red are negative samples.

my dinnertime acoustics playlist.") leads UGEN to extract the words that are exactly the slot values. Question-3 lists the slot values in X and steers the UGEN to select their slot names from options. To recognize the connection between slot values and their names, Question-4 is synthesized through a slot value with its slot name (positive) or random span in X with a slot name (negative). Question-5 is the most challenging, requiring UGEN to induce all the slot values and their names by the given question and options. Here, Questions 2-4 act as the auxiliary drivers and encourage the model to capture the links between slot names and their mentions in the Context X.

To simulate the pre-training manner, the input X is converted to "<s> Context: X </s> Question:Q </s> Options: O </s>". Here, the special tokens <s> and </s> are used to separate context, question and options. For the intents, the output Y is "i_1, i_k", i_k is one of the intent labels and generally, $i_k \leq 3$. On the slots side, the output Y (e.g., "$\{w_1, w_2\}$" is one "s_2") consists of slot values $\{w_1, w_2\}$ or slot values with slot names s_2, such as "track is one music item".

At the training stage, we first pre-process the original utterances with all the 5-type templates and shuffle the processed samples, then feed them into the UGEN to direct the model to understand the implicit correlations between intents and slots. The questions 2 to 4 are only used in the training phase and act more like auxiliary tasks. In the evaluation phase, only question-1 and question-5 are used to generate the intents and slot values with slot names, respectively.

4 Experiments

4.1 Experiment Setup

Dataset We compare our method with the baselines on two popular multi-intent SLU datasets, *MixSNIPS* and *MixATIS*. *MixSNIPS* is constructed from *SNIPS* dataset (Coucke et al., 2018) which comprises 39,776/2,198/2,199 utterances for training, validation and testing, separately. *MixATIS* is collected from *ATIS* (Hemphill et al., 1990), which contains 13,161/759/828 utterances for training, validation and testing, respectively. In addition, both of datasets are the cleaned version, and the proportion of sentences with $1 \sim 3$ intentions is [0.3, 0.5, 0.2].

We train and test all the models on the 32GB Tesla V100. For full-volume data, we set batch size to 20. The learning rate with Adam optimizer is set to $3e-5$, and beam search size is set to 3. In the few shot setting (5/10, and 10% training data), we set batch size to 16. In addition, we exploit the T5-base [2] as the backbone model.

[2] https://huggingface.co/t5-base

Baselines We compare UGEN with existing top-performing multi-intent approaches:

Joint Multiple ID-SF (JM) (Gangadharaiah and Narayanaswamy, 2019) proposes a multi-task framework and utilizes an attention-based model to identify intents and produce slot labels at the token-level.

Stack-Propagation (SP) (Qin et al., 2019) adopts a joint model with Stack-Propagation to use the intent information as input for slot filling and performs the token-level intent detection to alleviate the error propagation.

AGIF (Qin et al., 2020) presents an Adaptive Graph-Interactive Framework for joint multiple intent detection and slot filling, and it extracts the intents information for token-level slot prediction.

GL-GIN (Qin et al., 2021) proposes a Global-Locally Graph Interaction Network which explores a non-autoregressive model for joint multiple intent detection and slot filling.

SDJN (Chen et al., 2021a) introduces a novel self-distillation model which formulates multiple intent detection as a weakly supervised problem and designs an auxiliary loop to decode the intents and slots.

Model	MixSNIPS			MixATIS		
	S-F1	I-Acc	O-Acc	S-F1	I-Acc	O-Acc
JM	90.6	95.1	62.9	84.6	73.4	36.1
SP	94.2	96.0	72.9	87.8	72.1	40.1
AGIF	94.2	95.1	74.2	86.7	74.4	40.8
GL-GIN	94.9	95.6	75.4	88.3	76.3	43.5
SDJN	94.4	96.5	75.7	88.2	77.1	44.6
UGEN	**95.0**	**96.9**	**78.8**	**89.2**	**83.0**	**55.3**

Table 1: Overall results on the *MixSNIPS* and *MixATIS* sets with full-data. S-F1, I-Acc,O-Acc refer to the slot F1, intent-accuracy, and overall accuracy (both intents and slots need to be right), respectively. The highest numbers are in bold.

4.2 Overall Results

Table 1 reports the test results of UGEN compared to existing top-performing models on *MixSNIPS* and *MixATIS*. To the time of writing, UGEN outperforms the comparison models in all the metrics and obtains the new SOTA. For slot F1, our method leads to slight improvements (0.1% and 0.9%) compared to the GL-GIN, which validates that UGEN is more effective while extracting the slot values with their names. Turning to intent accuracy, UGEN exceeds SDJN (the previous SOTA) by 0.4% and 5.9%, respectively. It proves that UGEN has a strong ability to identify intents. Moreover, UGEN surpasses SDJN by 3.1% and 10.7% on overall accuracy (the more tough metric), which confirms that UGEN is more powerful in understanding the implicit correlations between intents and slots. The improvements align with our design and verify that the question-driven instructions are effective.

4.3 Few shot setting

Table 2 reports the results in the setting of 5/10-shot and 10% training data. We find that UGEN can consistently exceed the comparison models by a large margin in all the metrics. For instance, not only can UGEN increase by 23.5, 13.8, and 1.5 points in slot F1, but it leads to 28.1, 23.0, and 5.1 improvements in overall accuracy. The remarkable results validate that UGEN is more robust and can effectively exploit the implicit intent-slot correlations even with limited samples.

4.4 Ablation study

To explore the contribution of instructional prompts, we first remove the auxiliary instructions (Questions 2-4). The results drop a lot (e.g., 42.2% and 49.0% for overall accuracy) in the 5/10-shot, which demonstrates the auxiliary questions-driven templates are absolutely significant. Second, we only remove options in templates but keep all the questions. Every result under 5/10-shot and 10% training data is extremely low, sharply falling 34.7%, 31.2%, and 1.6%. The results confirm that options can effectively restrain the search space while predicting the answers. All the results are reported in Table 2.

5 Conclusion

In this work, we present a novel unified generative framework (UGEN) to treat the joint multiple intent detection and slot filling as a question-answering problem. To leverage the knowledge learned in the PLMs, we define 5-type prompt templates as the drivers to lead UGEN to grasp the prompt paradigm and capture the implicit correlations between intents and slots. On two multi-intent benchmark datasets, our approach accomplishes the new state-

Model	5-shot			10-shot			10%		
	S-F1	I-Acc	O-Acc	S-F1	I-Acc	O-Acc	S-F1	I-Acc	O-Acc
SP	58.7	78.2	11.9	71.5	88.3	24.8	90.3	93.5	58.4
AGIF	60.7	77.8	14.4	73.6	86.3	27.5	91.2	93.0	62.8
GL-GIN	54.3	86.1	10.1	69.5	90.2	23.9	92.1	95.3	66.6
UGEN - auxiliary instructions	32.3	18.4	0.3	37.2	34.6	1.5	92.6	95.4	67.5
UGEN - options	61.9	49.3	7.8	72.7	71.2	19.3	93.1	95.5	70.1
UGEN	**84.2**	**92.4**	**42.5**	**87.4**	**93.3**	**50.5**	**93.6**	**96.0**	**71.7**

Table 2: Results on the *MixSNIPS* set in the few shot settings. Because Joint Multiple ID-SF (JM) and SDJN are not publicly available, we can only compare the other baselines. S-F1, I-Acc,O-Acc refer to the slot F1, intent-accuracy, and overall accuracy (both intents and slots need to be right), respectively.

of-the-art performances in all the metrics, which validates that our design is effective. Meanwhile, UGEN leads to 28.1%, 23.0%, and 5.1% improvements in the 5/10-shot and 10% training data settings, which verify that UGEN is robust with limited annotation data.

6 Acknowledgements

The authors gratefully acknowledge Kebin Fang, Yingrong Wang, and Yao Zhao for giving valuable suggestions on this study. Our thanks also go to all the anonymous reviewers for their positive feedback.

References

Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel Ziegler, Jeffrey Wu, Clemens Winter, Chris Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. 2020. Language models are few-shot learners. In *Advances in Neural Information Processing Systems*.

Lisong Chen, Peilin Zhou, and Yuexian Zou. 2021a. Joint multiple intent detection and slot filling via self-distillation. *CoRR*, abs/2108.08042.

Xiang Chen, Ningyu Zhang, Lei Li, Xin Xie, Shumin Deng, Chuanqi Tan, Fei Huang, Luo Si, and Huajun Chen. 2021b. Lightner: A lightweight generative framework with prompt-guided attention for low-resource ner. *ArXiv*, abs/2109.00720.

Alice Coucke, Alaa Saade, Adrien Ball, Théodore Bluche, Alexandre Caulier, David Leroy, Clément Doumouro, Thibault Gisselbrecht, Francesco Caltagirone, Thibaut Lavril, Maël Primet, and Joseph Dureau. 2018. Snips voice platform: an embedded spoken language understanding system for private-by-design voice interfaces. *ArXiv*, abs/1805.10190.

Zeyuan Ding, Zhihao Yang, Hongfei Lin, and Jian Wang. 2021. Focus on interaction: A novel dynamic graph model for joint multiple intent detection and slot filling. In *Proceedings of the Thirtieth International Joint Conference on Artificial Intelligence, IJCAI-21*, pages 3801–3807. International Joint Conferences on Artificial Intelligence Organization. Main Track.

Haihong E, Peiqing Niu, Zhongfu Chen, and Meina Song. 2019. A novel bi-directional interrelated model for joint intent detection and slot filling. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 5467–5471, Florence, Italy. Association for Computational Linguistics.

Rashmi Gangadharaiah and Balakrishnan Narayanaswamy. 2019. Joint multiple intent detection and slot labeling for goal-oriented dialog. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*. Association for Computational Linguistics.

Xiaodong Gu, Kang Min Yoo, and Sang-Woo Lee. 2021. Response generation with context-aware prompt learning. *ArXiv*, abs/2111.02643.

Xu Han, Weilin Zhao, Ning Ding, Zhiyuan Liu, and Maosong Sun. 2021. Ptr: Prompt tuning with rules for text classification. *ArXiv*, abs/2105.11259.

Charles T. Hemphill, John J. Godfrey, and George R. Doddington. 1990. The ATIS spoken language systems pilot corpus. In *Speech and Natural Language: Proceedings of a Workshop Held at Hidden Valley, Pennsylvania, June 24-27,1990*.

Feihu Jin, Jinliang Lu, Jiajun Zhang, and Chengqing Zong. 2022. Instance-aware prompt learning for language understanding and generation. *ArXiv*, abs/2201.07126.

Bin Li, Fei Xia, Yixuan Weng, Xiusheng Huang, Bin Sun, and Shutao Li. 2021. PSG: prompt-based sequence generation for acronym extraction. *CoRR*, abs/2111.14301.

Bing Liu and Ian R. Lane. 2016. Attention-based recurrent neural network models for joint intent detection and slot filling. In *INTERSPEECH*.

Pengfei Liu, Weizhe Yuan, Jinlan Fu, Zhengbao Jiang, Hiroaki Hayashi, and Graham Neubig. 2021. Pre-train, prompt, and predict: A systematic survey of prompting methods in natural language processing. *CoRR*, abs/2107.13586.

Libo Qin, Wanxiang Che, Yangming Li, Haoyang Wen, and Ting Liu. 2019. A stack-propagation framework with token-level intent detection for spoken language understanding. In *EMNLP/IJCNLP (1)*, pages 2078–2087.

Libo Qin, Fuxuan Wei, Tianbao Xie, Xiao Xu, Wanxiang Che, and Ting Liu. 2021. GL-GIN: fast and accurate non-autoregressive model for joint multiple intent detection and slot filling. *CoRR*, abs/2106.01925.

Libo Qin, Xiao Xu, Wanxiang Che, and Ting Liu. 2020. AGIF: An adaptive graph-interactive framework for joint multiple intent detection and slot filling. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, Online. Association for Computational Linguistics.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J. Liu. 2020. Exploring the limits of transfer learning with a unified text-to-text transformer. *Journal of Machine Learning Research*.

Gokhan Tur and Renato De Mori. 2011. *Spoken language understanding: Systems for extracting semantic information from speech*. John Wiley & Sons.

Yaqing Wang, Quanming Yao, James Tin-Yau Kwok, and Lionel M. Ni. 2019. Generalizing from a few examples: A survey on few-shot learning. *arXiv: Learning*.

Henry Weld, X. Huang, Siqu Long, Josiah Poon, and Sooji Han. 2021. A survey of joint intent detection and slot-filling models in natural language understanding. *ArXiv*, abs/2101.08091.

Adaptive Unsupervised Self-training for Disfluency Detection

Zhongyuan Wang, Yixuan Wang, Shaolei Wang, Wanxiang Che*
Research Center for Social Computing and Information Retrieval
Harbin Institute of Technology, China
{zywang, yixuanwang, slwang, car}@ir.hit.edu.cn

Abstract

Supervised methods have achieved remarkable results in disfluency detection. However, in real-world scenarios, human-annotated data is difficult to obtain. Recent works try to handle disfluency detection with unsupervised self-training, which can exploit existing large-scale unlabeled data efficiently. However, their self-training-based methods suffer from the problems of selection bias and error accumulation. To tackle these problems, we propose an adaptive unsupervised self-training method for disfluency detection. Specifically, we re-weight the importance of each training example according to its grammatical feature and prediction confidence. Experiments on the Switchboard dataset show that our method improves 2.3 points over the current SOTA unsupervised method. Moreover, our method is competitive with the current SOTA supervised method.

1 Introduction

Disfluency is a characteristic of spontaneous speech which is different from written texts. Detecting and removing the non-fluent word sequences in spoken language transcripts can improve the transcripts' quality and provide clean inputs for the downstream NLP tasks, such as parsing, machine translation, and summarization (Tree, 1995; Wang et al., 2020b). As shown in Figure 1, a standard annotation of the disfluency structure indicates the RM (*reparandum*, words that the speaker intends to discard), IM (*interregnum*, filled pauses, discourse cue words, etc.), RP (*repair*, the associated repair) (Shriberg, 1994).

Most previous works (Zayats et al., 2016; Wu et al., 2015; Lou and Johnson, 2017; Jamshid Lou et al., 2018; Zayats and Ostendorf, 2019) on disfluency detection heavily relies on human-annotated corpora, which is difficult and expensive to obtain in practice. Some researchers try to alleviate this issue with, for instance, self-supervised learning (Wang et al., 2020a) and semi-supervised learning techniques (Wang et al., 2018), but they still need a certain amount of human-annotated corpora to perform high performance. Wang et al. (2020b) completely removes the need for human-annotated data. They first perform disfluency detection with the self-training method in an unsupervised manner, which shows promising performance. They leverage the pseudo data constructed by rules to train a weak disfluency detection model as a teacher. Then, the student model is fine-tuned in an unsupervised manner, only using the pseudo-labels generated by the teacher. Finally, the student model achieves promising performance.

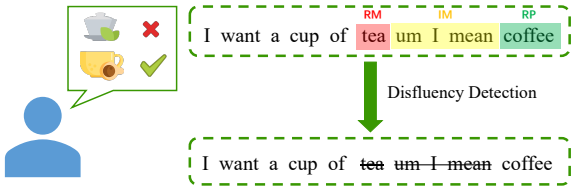

Figure 1: A sentence with disfluencies annotated. RM=Reparandum, IM=Interregnum, RP=Repair. The preceding RM is corrected by the following RP.

Traditional self-training methods utilize the probability estimations to select high-confidence pseudo-labels for re-training, which encounters two problems. (1) *Selection Bias.* The strategy that selects samples with the high confidence pseudo-labeled samples tends to neglect the hard and potentially informative samples. It results in the self-training procedure converging to a suboptimal solution. (2) *Error Accumulation.* The strategy that selects pseudo-labels with probability estimations, limited by the ability of the model, will inevitably introduce a large number of noisy labels and the noise will be accumulated with the increase of iteration times. The above problems become the bottleneck for self-training, which prevents model performance from growing as the number of iter-

*Email corresponding.

Proceedings of the 29th International Conference on Computational Linguistics, pages 7209–7218
October 12–17, 2022.

ations increases. To alleviate the error accumulation problem, Wang et al. (2020b) introduces a grammar checker into the selection process to enhance the ability to select high-quality pseudo-labels, which can be seen as introducing external knowledge. Concretely, Wang et al. (2020b) employed a grammar checker to discard samples that may be noise based on their grammatical correctness. However, their method has no ability to recall potential informative samples, which leaves the selection bias problem unmitigated. In addition, due to the limited performance of the grammar checker, the error accumulation problem still exists to a certain extent.

In this paper, we propose an adaptive self-training method that utilizes a re-weighting strategy to address both the problem of selection bias and error accumulation. Concretely, we assign specific weight to every pseudo-labeled sample based on its quality. We measure the quality of each sample by its grammatical feature and label confidence (see §2 for details). Compared with the method that selects samples with high confidence, our re-weighted method has two main advantages: First, instead of only selecting the pseudo-labeled samples with high confidence, our method makes all samples participate in the training process, which can alleviate the problem of selection bias to some extent. Second, our method can reduce the negative effect of noise data by lowering the weights of low-quality pseudo-labels and emphasizing high-quality pseudo-labels during the self-training process.

Besides, to help improve the performance of self-training, we also propose a more powerful contrastive grammar checker. The grammar checker proposed by Wang et al. (2020b) takes a single sentence as input and then judges whether the sentence is grammatical or not. In our practice, we found that the way of judging grammatical correctness by a single sentence alone faces great difficulties when using the ASR results as input, e.g. there are incomplete sentences with missing beginnings or endings in the ASR results, which may lead to the single-sentence grammar checker to misjudge. Therefore, it is hard to train a high-performance single-sentence grammar checker model. To solve this problem, we employ a contrastive mechanism to make it easier to judge whether the sentence is grammatical or not. We take both the sentence before and after removing disfluency elements as the input of our grammar checker. Therefore, our grammar checker can compare the two sentences to help judge whether the sentence is grammatical or not, which makes our grammar checker powerful.

We evaluate the proposed method on the commonly used English Switchboard dataset, and our method improves 2.3 points over the previous best result in unsupervised settings. Moreover, our method achieves competitive performance compared to the state-of-the-art supervised method which utilizes 60k labeled sentences. In addition, the experimental result on three other datasets shows that our method can consistently achieve competitive performance compared to the supervised systems.

The contributions of our work can be summarized as follows:

- To enhance the performance of unsupervised disfluency detection, we propose an adaptive self-training method utilizing a re-weighting strategy that can alleviate the problems of selection bias and error accumulation in the previous self-training-based method for disfluency detection.
- We propose a more powerful contrastive grammar checker which can better evaluate the quality of the pseudo-labeled data for disfluency detection.
- Experimental results on four datasets demonstrate our proposed method surpasses the performance of the existing SOTA method in unsupervised settings.

We will release our code and model[1].

2 Method

2.1 Procedure Overview

Figure 2 shows an overview of our adaptive unsupervised self-training framework for disfluency detection. Algorithm 1 is presented to help understand our method. Our method only takes unlabeled sentences as inputs, including news data and ASR outputs. We first use self-supervised learning to train the contrastive grammar checker on the large-scale constructed pseudo data (see §2.3 for details), which is used to judge whether a sentence is grammatical or not. Then, we use self-supervised learning to train a weak disfluency detection model with large-scale constructed pseudo data (see §2.2 for details), which is used as the teacher model.

[1] https://github.com/wyxstriker/ReweightingDisfluency

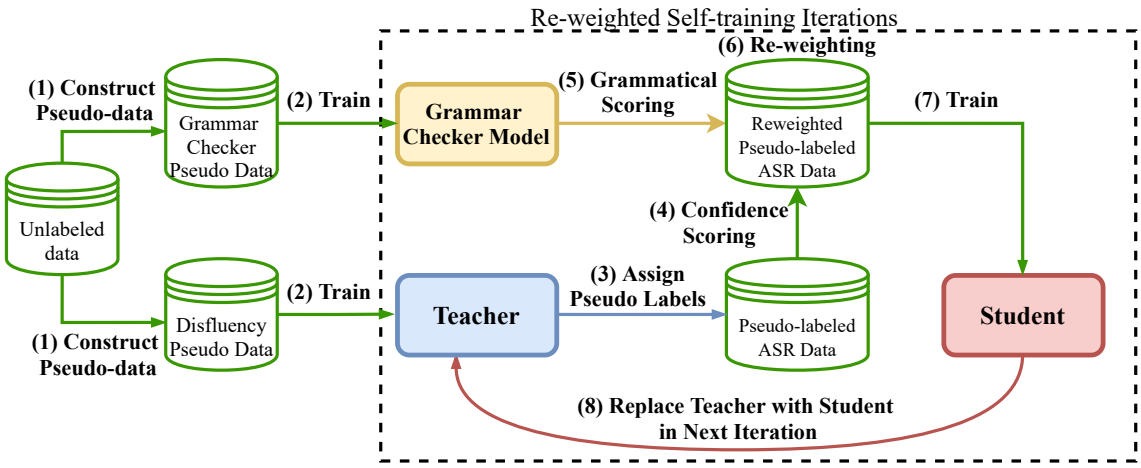

Figure 2: Illustration of our proposed method. Step 5 to Step 8 are looped until the student model performance no longer grows.

Next, we use the teacher model to assign pseudo-labels on unlabeled ASR outputs. We assign specific weight to each pseudo-labeled sentence from two different perspectives: 1) grammatical correctness, we emphasize the sentences that are more grammatical after removing the words with disfluency labels, and 2) pseudo-label confidence, we emphasize the high-confidence pseudo-labeled sentences. To calculate the final weight of each sentence (see §7 for details), we add grammatical correctness score and pseudo-label confidence together. We then train a student model on all pseudo-labeled sentences, which are assigned with different weights calculated in the way mentioned above. Then, the teacher model is replaced with the student model. The process will be looped until the performance of the student model stops growing.

2.2 Teacher Model

We formulate the disfluency detection task as a token-level classification task. The training goal is to detect the disfluency words by associating labels with them. For each token x_i in a sentence, the model assigns a label y_i which is "fluent" or "disfluent". To obtain the teacher model, we directly fine-tune the ELECTRA model (the discriminator) (Clark et al., 2020) by minimizing the cross-entropy loss:

$$loss_{teacher} = \frac{1}{N}\sum_{i=1}^{N} \mathrm{CE}(y_i, f_t(x_i, \theta_t)), \quad (1)$$

where $(x_i, y_i)_{i=1}^{N}$ denotes the input sentences and labels, CE denotes the cross-entropy loss function, f_t denotes our teacher model parameterized by θ_t.

Our teacher model trained on the pseudo data constructed by the self-supervised method. We follow Wang et al. (2020b) to construct the pseudo data. The unlabeled data to construct pseudo data comes from fluent news data. To simulate the sentence containing disfluencies, two types of random perturbations are introduced into the fluent sentences. One is repetition, we randomly select several continuous words in the fluent sentence to be repeated. The other is inserting, we randomly pick words from external vocabulary to insert into fluent sentences.

2.3 Grammar Checker Model

Traditional self-training methods utilize the probability estimations to select high-confidence pseudo-labels for re-training. These methods suffer from the problems of selection bias and error accumulation. To enhance the ability to select high-quality pseudo-labels in self-training, Wang et al. (2020b) proposes a grammar checker, which can be seen as external knowledge. However, we found that the grammar checker proposed in Wang et al. (2020b) may misjudge when dealing with the ASR outputs since there are incomplete sentences with missing beginnings or endings in the ASR output. To address this problem, we allow the grammar checker to compare the sentence before and after removing the disfluency elements, which can help judge whether a sentence is grammatical or not. Concretely, given an unlabeled sentence S, the disfluency detection model can assign a pseudo label for each word. By deleting the tokens with the "disfluent" label, we obtain the processed sentence $\bar{S}$. We concatenate S and $\bar{S}$ to be the input of our con-

Algorithm 1 : The learning algorithm of our unsupervised model for disfluency detection

Require: Pseudo data for disfluency detection $(x_i, y_i)_{i=1}^{N}$, pseudo data for grammar checker model $(\hat{x}_i, \hat{y}_i)_{i=1}^{M}$, and unlabeled ASR outputs $\{\tilde{x}_1, \tilde{x}_2, ..., \tilde{x}_K\}$.

1: Learn grammar checker model θ_g using $(\hat{x}_i, \hat{y}_i)_{i=1}^{M}$ via Eq. 2
2: Learn teacher model θ_t using $(x_i, y_i)_{i=1}^{N}$ via Eq. 1
3: Randomly sample data $\{\tilde{x}_1, \tilde{x}_2, ..., \tilde{x}_L\}$ from unlabeled ASR outputs $\{\tilde{x}_1, \tilde{x}_2, ..., \tilde{x}_K\}$ Use teacher model to generate pseudo labels $\tilde{y}_i$ and token-level weight s_{ij}^{lc} for sampled data via Eq. 5
4: Use grammar checker model to generate grammatical correctness score s_i^{gc}.
5: Learn a student model θ_s on sampled data $(\tilde{x}_i, \tilde{y}_i)_{i=K}^{N}$ with weight s_{ij}^{lc} and s_i^{gc} via Eq. 7
6: Iterative training until the performance stops growing: Use the student as a teacher and go back to step 3

trastive grammar checker. Our contrastive grammar checker can judge whether $\bar{S}$ is grammatical or not from two perspectives: 1) makes judgment from $\bar{S}$ itself, and 2) makes judgment by comparing S and $\bar{S}$.

We directly fine-tune the ELECTRA model (the discriminator) (Clark et al., 2020) on our pseudo data which minimizes the cross-entropy loss on:

$$loss_{checker} = \frac{1}{M}\sum_{i=1}^{M} \mathrm{CE}(\hat{y}_i, f_g(\hat{x}_i, \theta_g)), \quad (2)$$

where $(\hat{x}_i, \hat{y}_i)_{i=1}^{M}$ denotes input sentences and labels, CE denotes the cross-entropy loss function, f_g denotes our grammar checker model parameterized by θ_g.

Our contrastive grammar checker is trained with pseudo data constructed by the self-supervised learning method. Inspired by Wang et al. (2020b), we use three types of perturbations to simulate grammatical wrong sentences. The first two types of perturbations are repetition and inserting as described previously. The third type of perturbation is deletion. We randomly delete several words from the fluent sentence.

2.4 Re-weighting Mechanism

In this paper, we use two types of re-weighting mechanisms including grammatical correctness re-weighting and label confidence re-weighting. Fig 3 shows our re-weighting mechanism.

2.4.1 Grammatical Correctness Re-weighting

Once a sentence is labeled correctly by the disfluency detection model, the rest after removing the words with disfluency labels is fluent and grammatical (Wang et al., 2020b). Based on this fact, we employ a re-weight strategy to emphasize the sentences that are more grammatical after removing the words with disfluency labels.

With the grammatical correctness re-weighting mechanism, the training objects of the student model are formulated as follows:

$$loss_{gc} = \frac{1}{L}\sum_{i=1}^{L} s_i^{gc} \times \mathrm{CE}(\tilde{y}_i, f_s(\tilde{x}_i, \theta_s)), \quad (3)$$

where $(\tilde{x}_i, \tilde{y}_i)_{i=1}^{L}$ are sentences with pseudo-labels, s_i^{gc} denotes grammatical correctness score which is the logits output of the grammar checker model.

2.4.2 Label Confidence Re-weighting

We employ the label confidence obtained from our teacher model to re-weight the tokens in each sentence. Observing that pseudo-labels with higher confidence are more likely to be correctly labeled, we reinforce the role of high-confidence labels in the training process. With the label confidence re-weighting mechanism, the training object is formulated as follows:

$$loss_{lc} = \frac{1}{LN}\sum_{i=1}^{L}\sum_{j=1}^{N} s_{ij}^{lc} \times \mathrm{CE}(\tilde{y}_{ij}, f_s(\tilde{x}_{ij}, \theta_s)), \quad (4)$$

where $(\tilde{x}_i, \tilde{y}_i)_{i=1}^{L}$ are sentences with pseudo-labels, CE is the cross-entropy loss function, N is the length of the token sequence, s_{ij}^{lc} denotes the teacher model's processed logit output of the token $\tilde{x}_{ij}$.

In the early stages of the iterative self-training process, the performance of the student model is relatively weak. Therefore the confidence output of the student model is not reliable enough. We introduce the temperature mechanism to weaken the effect of confidence when it is unreliable in the early self-training stages.

The calculation of the s_{ij}^{lc} with the temperature mechanism can be formulated as follows:

$$s_{ij}^{lc} = N\frac{exp(\frac{g_{ij}}{T})}{\sum_{k=1}^{N} exp(\frac{g_{ik}}{T})}, \quad (5)$$

where s_{ij}^{lc} denotes the weight of jth token of sample $\tilde{x}_i$, T is the temperature, g_{ij} is the teacher model's confidence in $token_j$ of $\tilde{x}_i$, and N is the length of the token sequence.

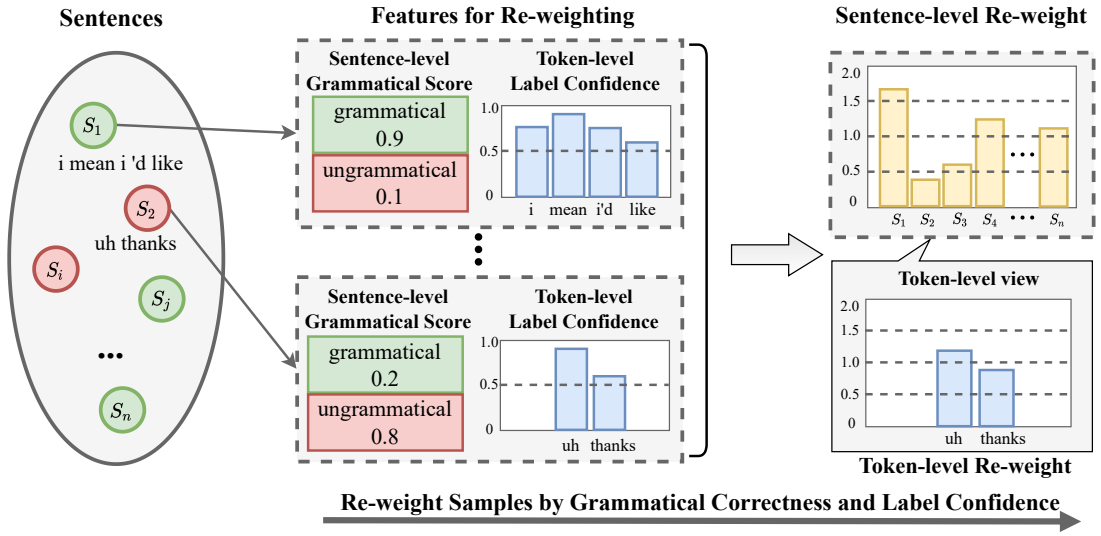

Figure 3: Illustration of proposed re-weighting mechanism. At the sentence-level, sentences with higher grammatical scores are given higher weights after removing words with the label "disfluent". At the token-level, labels with higher confidence are given higher weights.

The temperature T can be formulated as follows:

$$T = -\gamma * log(\frac{n_{sample}}{2 * n_{total}}), \quad (6)$$

where n_{sample} is the amount of data used in this iteration of self-training, n_{total} is the amount of all unlabeled data. γ is a hyper-parameter used to adjust the rate of temperature drop.

2.5 Student Teacher Iterative Training

In Section 2.4, we introduce two different re-weighting mechanisms. Finally, we combine two re-weighting mechanisms, as figure 3 shows, to train the student model, the training objects is formulated as follows:

$$loss_{student} = \frac{1}{LN} \sum_{i=1}^{L} \sum_{j=1}^{N} (s_i^{gc} + s_{ij}^{lc}) \times \tilde{\text{CE}}, \quad (7)$$

where $(\tilde{x}_i, \tilde{y}_i)_{i=1}^{L}$ represents pseudo-labeled sentences, $\tilde{\text{CE}}$ represents $\text{CE}(\tilde{y}_{ij}, f_s(\tilde{x}_{ij}, \theta_s))$, and N is the length of the token sequence, s_i^{gc} and s_{ij}^{lc} are calculated according to the Section 2.4.

We fine-tuned student models with pseudo-labeled data on the teacher model trained by pseudo data. The student will be a new teacher in the next iteration.

3 Experiment

3.1 Settings

Dataset English Switchboard (Godfrey et al., 1992) is a large multispeaker corpus of conversational speech and text, which is the largest standard corpus for disfluency detection. Following the experiment settings in (Wang et al., 2020b), we process the Switchboard corpus and split it into the train (60k sentences), dev (4k sentences), and test set (4k sentences). Notice that the train set is not used in our method. The news data are from WMT2017 monolingual language model training data (News Discussions. Version 2).[2] We conduct our main experiment on the English Switchboard.

To demonstrate the robustness of our method, we also test our model on three out-of-domain datasets (Zayats et al., 2014; Zayats and Ostendorf, 2018), including CallHome, SCOTUS, and FCIC. Notice that the train set of the three out-of-domain datasets is not used.

- ***CallHome***: phone conversations between family members and close friends. It consists of a training set of 46k words and a test set of 30k words.
- ***SCOTUS***: transcribed Supreme Court oral arguments between justices and advocates. It consists of a test set of 43k words and has no training set.
- ***FCIC***: two transcribed hearings from Financial Crisis Inquiry Commission. It consists of a test set of 54k words and has no training set.

Metric Following Wang et al. (2020b), we use token-based precision (P), recall (R), and F1 as the

[2] http://www.statmt.org/wmt17/translation-task.html

Method	F1
Supervised methods	
UBT (Wu et al., 2015)	85.1
Bi-LSTM (Zayats et al., 2016)	85.9
NCM (Lou and Johnson, 2017)	86.8
Transition-based (Wang et al., 2017)	87.5
Self-supervised (Wang et al., 2020a)	90.2
Self-training (Jamshid and Mark, 2020)	90.6
EGBC (Bach and Huang, 2019)	91.8
PG (Yang et al., 2020)	**92.3**
BERT fine-tuning	90.5
ELECTRA fine-tuning	91.2
Teacher fine-tuning	92.0
Unsupervised methods	
Unsupervised teacher	72.3
SSST (Wang et al., 2020b)	88.0
Our Method	**90.3**

Table 1: Comparison with previous state-of-the-art methods on the Switchboard test set. For robustness, we run our proposed supervised baselines and our proposed method 5 times and report the average metric.

evaluation metrics.

3.2 Training Details

For all experiments, we use the English ELECTRA-base discriminator model as an encoder, which has 110M hidden units, 12 heads, and 12 hidden layers. For the self-supervised teacher model, the max sequence length of teacher model input is set to 128. We fine-tune the teacher model using the AdamW optimizer for 30 epochs with a batch size of 256 and a learning rate of 1e-4. Since the grammar checker model's input consists of a pair of sentences, we set its max sequence length to be 256. We fine-tune the grammar checker model using AdamW optimizer for 30 epochs with a batch size of 256 and a learning rate of 1e-4.

When training the student model in the self-training procedure, we use the AdamW optimizer for 15 epochs with a batch size of 128 and a learning rate of 2e-5.

3.3 Performance on English Switchboard

Table 1 shows the comparison of the performance among different models on the Switchboard test set. In addition to the previous methods, we have also built the following baseline models for comparison.

- BERT fine-tuning: We directly fine-tune the BERT-Base model on the Switchboard train set.
- ELECTRA fine-tuning: We directly fine-tune ELECTRA-Base (the discriminator) model on the Switchboard train set.
- Unsupervised teacher: This model is the first teacher model trained by pseudo data.
- Teacher fine-tuning: We fine-tune the teacher model which is trained by pseudo data on the Switchboard train set.

Method	CallHome	SCOTUS	FCIC
Supervised methods			
ELECTRA fine-tuning	62.2	81.6	63.1
Teacher fine-tuning	63.4	**81.9**	63.8
Pattern-match	**65.2**	79.9	**66.1**
Unsupervised methods			
Unsupervised teacher	48.0	69.2	47.6
SSST	60.2	80.3	63.3
Our unsupervised	**61.6**	**80.7**	**63.6**

Table 2: F1 scores on cross-domain disfluency detection. For robustness, we run our proposed supervised baselines and our proposed method 5 times and report the average metric and standard deviation metric.

Our unsupervised model achieves 2.3 points improvements over the past state-of-the-art unsupervised method (Wang et al., 2020b) which is also based on self-training. Moreover, our unsupervised model achieves competitive performance compared to the current SOTA supervised (Yang et al., 2020) method, which trains BERT with 20M sentences of pseudo data and 173k sentences of human-annotated data.

3.4 Performance on Cross-domain Data

In this section, we experiment on three out-of-domain disfluency datasets to prove the robustness of our proposed approach.

We build four baseline systems for comparison, including Unsupervised teacher, ELECTRA fine-tuning, Teacher fine-tuning, SSST (Wang et al., 2020b), and Pattern-match (Zayats and Ostendorf, 2018). Unsupervised teacher, ELECTRA fine-tuning, and Teacher fine-tuning are the same as described in Section 3.3. Pattern-match (Zayats and Ostendorf, 2018) used a pattern match neural network architecture trained on the Switchboard train set. It achieves state-of-the-art performance in cross-domain scenarios.

Following the experiment setting in (Wang et al., 2020b), we test each model's performance on the three out-of-domain test sets without any retraining on the out-of-domain train set.

Table 2 shows the comparison of the performance among different models on the three out-

Method	P	R	F1
ST+SGC	90.2	89.1	89.6
ST+SGC+Re-weight	**90.5**	91.7	91.1
ST+CGC	89.5	91.0	90.3
ST+CGC+Re-weight	90.3	**92.9**	**91.6**

Table 3: Ablation results of our method on Switchboard dev set. "ST" denotes self-training. "SGC" denotes the single sentence grammar checker. "CGC" denotes the contrastive grammar checker. "Re-weight" denotes grammatical correctness re-weighting and label confidence re-weighting mechanism.

Method	P	R	F1
ST+CGC	89.5	91.0	90.3
ST+CGC+GCR	**91.2**	91.0	91.1
ST+CGC+LCR	90.2	92.1	91.1
ST+CGC+GCR+LCR	90.3	**92.9**	**91.6**

Table 4: Ablation results of our method on Switchboard dev set. "ST" denotes self-training. "CGC" denotes contrastive grammar checker. "GCR" denotes grammatical correctness re-weighting mechanism."LCR" denotes label confidence re-weighting mechanism.

T hyperparameter	P	R	F1
w/o Temperature	90.7	92.0	91.4
w/ Temperature γ=2	**91.5**	90.9	91.2
w/ Temperature γ=1	91.1	91.7	91.4
w/ Temperature γ=0.5	90.3	**92.9**	**91.6**

Table 5: Performance of our method on Switchboard dev set with different temperature.

of-domain test sets. It shows that our unsupervised methods can achieve good performance in the three cross-domain datasets without any retraining.

4 Ablation Studies

In this section, we analyze the effect of several components of our method.

4.1 Effect of re-weighting mechanism

To explore the role of the re-weighting mechanism, we conduct a comparative experiment for the re-weighting mechanism. We take the approach proposed by Wang et al. (2020b) as our baseline, which is a self-training-based method with a single sentence grammar checker (referred to as "ST+SGC"). We separately add the re-weighting mechanism to the baseline system (referred to as "ST+SGC+Re-weight"). Table 3 shows that the method with a re-weighting mechanism outperforms the baseline by 1.5 points, demonstrating the benefit of the re-weighting mechanism.

Our re-weighting mechanism consists of grammatical correctness re-weighting and the label confidence re-weighting. To explore the role of each part of the re-weighting mechanism, we conduct a comparative experiment. Table 4 shows that both the grammatical correctness re-weighting and the label confidence re-weighting bring improvement.

4.2 Effect of improved grammar checker

In our approach, we propose a contrastive grammar checker. To validate the effectiveness of our grammar checker, we conduct a comparative experiment. "ST+SGC" denotes a self-training approach with a single sentence grammar checker which is proposed by Wang et al. (2020b). "ST+CGC" denotes a self-training approach with a contrastive grammar checker which is proposed by us. The only difference between them is that they use different grammar checkers. Table 3 shows that the method with our grammar checker outperforms the method with the previous grammar checker by 0.7 points.

To compare the performance of two grammar checkers directly, we test them on the grammar check task dev dataset, which is constructed from the Switchboard dev dataset. Concretely, we take the sentences from the Switchboard without disfluency components as positive examples, while the sentences with disfluency components as negative examples. As shown in Figure 4, the "CGC"proposed by us archieves a higher AUC than the "SGC" proposed by Wang et al. (2020b). It is worth noting that we don't let the "CGC" see the sentence after removing disfluency components. We repeat the raw sentence twice as the input of "CGC".

These demonstrate that our contrastive grammar checker can better evaluate the quality of pseudo-labeled data for disfluency detection.

4.3 Effect of temperature mechanism performance

We proposed a temperature mechanism to weaken the confidence re-weighting mechanism effect in the early self-training stage when the confidence is not reliable enough. To reveal the temperature mechanism effect on model performance, we conduct comparative experiments using different γ which control the rate of temperature drop. The higher γ we set, the smaller the confidence re-weighting effect in the early self-training stage. The experiment results in table 5 show that reducing the confidence re-weighting effect in the early

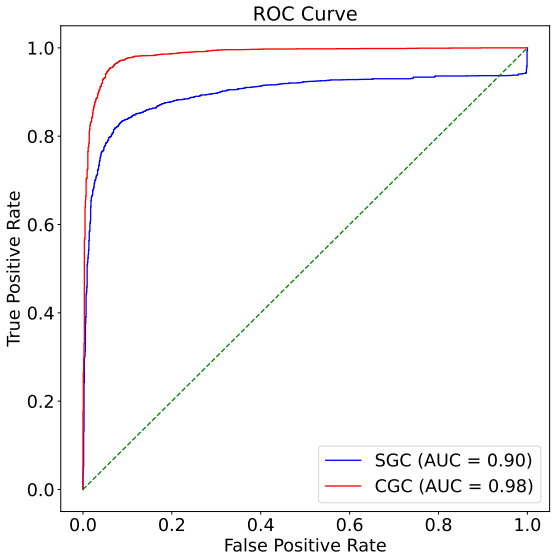

Figure 4: The ROC curve of grammar checkers. "SGC" denotes single sentence grammar checker. "CGC" denotes contrastive grammar checker.

self-training stage appropriately can improve the model performance. With these results, we chose the temperature value as 0.5.

5 Related Work

5.1 Disfluency Detection

Most of the previous work on disfluency detection focus on supervised learning methods. There are three main categories of methods to solve the disfluency detection problem including sequence tagging, noisy-channel, and parsing-based approaches. Sequence tagging methods use various models as classifiers to classify each word in the sentence to be fluent or disfluent, including conditional random fields (Georgila, 2009; Ostendorf and Hahn, 2013; Zayats et al., 2014), Max-Margin Markov Networks (M^3N) (Qian and Liu, 2013), Semi-Markov CRF (Ferguson et al., 2015), recurrent neural networks (Hough and Schlangen, 2015; Zayats et al., 2016; Wang et al., 2016) and transformer-based model (Wang et al., 2020a). Noisy-channel (Charniak and Johnson, 2001) methods use the similarity between reparandum and repair to detect disfluency. Parsing-based approaches (Rasooli and Tetreault, 2013; Wu et al., 2015) deal with disfluency detection by parsing the sentence.

Some previous work focuses on tackling the training data bottleneck, including self-supervised methods (Wang et al., 2020a), self-training methods (Jamshid and Mark, 2020) and active learning methods (Wang et al., 2021). Some researchers propose a method using unsupervised self-training to perform unsupervised disfluency detection which does need not any human-annotated data (Wang et al., 2020b).

Rocholl et al. (2021) uses the distillation method to make the disfluency detection model to be small, fast and on-device while maintaining competitive performance.

5.2 Self-training

The self-training method first leverages human-annotated data to train a teacher model. Then the teacher model is used to assign pseudo-label to unlabeled data. Finally, the labeled data and the pseudo-labeled data are merged to train the student model jointly Scudder (2006). Self-training has shown promising performance for a variety of tasks including leveraging noisy data (Veit et al., 2017), semantic segmentation (Babakhin et al., 2019) and text classification (Li et al., 2019). Xie et al. (2020) presents Noisy Student Training, which adds noise to the self-training process and changes the size of the student model during self-training iteration. Inspired by Xie et al. (2020), Wang et al. (2020b) combines self-supervised and self-training to build an unsupervised disfluency detection system for disfluency detection.

5.3 Re-weighting mechanism

Re-weighting approaches (Ren et al., 2018; Shu et al., 2019; Mei et al., 2020; Yaqing Wang et al., 2020) are proposed to give different loss weights on each sample to emphasize the important samples and discount the noisy samples. Re-weighting approaches often train a teacher model on a small human-annotated validation set and then use the teacher model to re-weight training samples. In our method, we do not need a teacher model trained by human-annotated data. We re-weight each training sample by the model trained by the self-supervised method.

6 Conclusion

In this work, we propose an adaptive unsupervised method to deal with the task of disfluency detection in an unsupervised manner. We introduce a re-weighting mechanism into the self-training to alleviate the problems of selection bias and error accumulation. Our experiments show that our re-weighting mechanism can alleviate the problems of selection bias and error accumulation efficiently,

and improve the performance of self-training for disfluency detection.

Acknowledgement

This work was supported by the National Key RD Program of China via grant 2020AAA0106501 and the National Natural Science Foundation of China (NSFC) via grant 61976072 and 62176078.

References

Yauhen Babakhin, Artsiom Sanakoyeu, and Hirotoshi Kitamura. 2019. Semi-supervised segmentation of salt bodies in seismic images using an ensemble of convolutional neural networks. In *Proc. of GCPR*.

Nguyen Bach and Fei Huang. 2019. Noisy bilstm-based models for disfluency detection. *Proc. Interspeech 2019*.

Eugene Charniak and Mark Johnson. 2001. Edit detection and parsing for transcribed speech. In *Proc. of ACL*.

Kevin Clark, Minh-Thang Luong, V. Quoc Le, and D. Christopher Manning. 2020. Electra: Pre-training text encoders as discriminators rather than generators. *ICLR*.

James Ferguson, Greg Durrett, and Dan Klein. 2015. Disfluency detection with a semi-markov model and prosodic features. In *Proc. of ACL*.

Kallirroi Georgila. 2009. Using integer linear programming for detecting speech disfluencies. In *Proc. of ACL*.

John J Godfrey, Edward C Holliman, and Jane McDaniel. 1992. Switchboard: Telephone speech corpus for research and development. In *Acoustics, Speech, and Signal Processing, IEEE International Conference on*, volume 1, pages 517–520. IEEE Computer Society.

Julian Hough and David Schlangen. 2015. Recurrent neural networks for incremental disfluency detection. In *Proc. of INTERSPEECH*.

Paria Lou Jamshid and Johnson Mark. 2020. Improving disfluency detection by self-training a self-attentive model. *ACL*.

Paria Jamshid Lou, Peter Anderson, and Mark Johnson. 2018. Disfluency detection using auto-correlational neural networks. In *Proc. of EMNLP*.

Xinzhe Li, Qianru Sun, Yaoyao Liu, Qin Zhou, Shibao Zheng, Tat-Seng Chua, and Bernt Schiele. 2019. Learning to self-train for semi-supervised few-shot classification. In *Proc. of NeurIPS*.

Paria Jamshid Lou and Mark Johnson. 2017. Disfluency detection using a noisy channel model and a deep neural language model. *Proc. of ACL*.

Ke Mei, Chuang Zhu, Jiaqi Zou, and Shanghang Zhang. 2020. Instance adaptive self-training for unsupervised domain adaptation. In *European conference on computer vision*, pages 415–430. Springer.

Mari Ostendorf and Sangyun Hahn. 2013. A sequential repetition model for improved disfluency detection. In *INTERSPEECH*.

Xian Qian and Yang Liu. 2013. Disfluency detection using multi-step stacked learning. In *HLT-NAACL*.

Mohammad Sadegh Rasooli and Joel R Tetreault. 2013. Joint parsing and disfluency detection in linear time. In *EMNLP*.

Mengye Ren, Wenyuan Zeng, Bin Yang, and Raquel Urtasun. 2018. Learning to reweight examples for robust deep learning. In *Proc. of ICML*.

Johann C. Rocholl, Vicky Zayats, Daniel D. Walker, Noah B. Murad, Aaron Schneider, and Daniel J. Liebling. 2021. Disfluency detection with unlabeled data and small bert models. In *Proc. Interspeech*.

H. Scudder. 2006. Probability of error of some adaptive pattern-recognition machines. *IEEE Trans. Inf. Theor.*

Elizabeth Ellen Shriberg. 1994. *Preliminaries to a theory of speech disfluencies*. Ph.D. thesis.

Jun Shu, Qi Xie, Lixuan Yi, Qian Zhao, Sanping Zhou, Zongben Xu, and Deyu Meng. 2019. Meta-weight-net: Learning an explicit mapping for sample weighting. In *Proc. of NeurIPS*.

Jean E Fox Tree. 1995. The effects of false starts and repetitions on the processing of subsequent words in spontaneous speech. *Journal of memory and language*.

Andreas Veit, Neil Alldrin, Gal Chechik, Ivan Krasin, Abhinav Gupta, and Serge Belongie. 2017. Learning from noisy large-scale datasets with minimal supervision. In *Proc. of CVPR*.

Feng Wang, Wei Chen, Zhen Yang, Qianqian Dong, Shuang Xu, and Bo Xu. 2018. Semi-supervised disfluency detection. In *Proc. of COLING*.

Shaolei Wang, Wangxiang Che, Qi Liu, Pengda Qin, Ting Liu, and William Yang Wang. 2020a. Multi-task self-supervised learning for disfluency detection. In *Proc. of AAAI*.

Shaolei Wang, Wanxiang Che, and Ting Liu. 2016. A neural attention model for disfluency detection. In *Proc. of COLING*.

Shaolei Wang, Wanxiang Che, Yue Zhang, Meishan Zhang, and Ting Liu. 2017. Transition-based disfluency detection using lstms. In *Proc. of EMNLP*.

Shaolei Wang, Zhongyuan Wang, Wanxiang Che, and Ting Liu. 2020b. Combining self-training and self-supervised learning for unsupervised disfluency detection. In *Proc. of EMNLP*.

Shaolei Wang, Zhongyuan Wang, Wanxiang Che, Sendong Zhao, and Ting Liu. 2021. Combining self-supervised learning and active learning for disfluency detection. *Proc. of TALLIP*.

Shuangzhi Wu, Dongdong Zhang, Ming Zhou, and Tiejun Zhao. 2015. Efficient disfluency detection with transition-based parsing. In *Proc. of ACL*.

Qizhe Xie, Minh-Thang Luong, Eduard Hovy, and Quoc V Le. 2020. Self-training with noisy student improves imagenet classification. In *Proc. of CVPR*.

Jingfeng Yang, Diyi Yang, and Zhaoran Ma. 2020. Planning and generating natural and diverse disfluent texts as augmentation for disfluency detection. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1450–1460, Online. Association for Computational Linguistics.

Yaqing Yaqing Wang, Subhabrata Mukherjee, Haoda Chu, Yuancheng Tu, Ming Wu, Jing Gao, and Ahmed Hassan Awadallah. 2020. Adaptive self-training for few-shot neural sequence labeling. *arXiv preprint arXiv:2010.03680*.

Vicky Zayats and Mari Ostendorf. 2018. Robust cross-domain disfluency detection with pattern match networks. *arXiv: Computation and Language*.

Vicky Zayats and Mari Ostendorf. 2019. Giving attention to the unexpected: Using prosody innovations in disfluency detection. *Proc. of NAACL*.

Vicky Zayats, Mari Ostendorf, and Hannaneh Hajishirzi. 2016. Disfluency Detection Using a Bidirectional LSTM. In *Proc. Interspeech 2016*, pages 2523–2527.

Victoria Zayats, Mari Ostendorf, and Hannaneh Hajishirzi. 2014. Multi-domain disfluency and repair detection. In *Proc. of INTERSPEECH*.

Author Index

Abdessaied, Adnen, 192
Abdul-Mageed, Muhammad, 5066
Abe, Kaori, 851
Abend, Omri, 4544
Abonizio, Hugo, 3506
Abzaliev, Artem, 5507
Adebara, Ife, 5066
Adelani, David Ifeoluwa, 4286, 4336
Adukia, Anjali, 3086
Afshar, Majid, 2979
Agadakos, Nikolaos, 694
Agarwal, Sanchit, 285
Agichtein, Eugene, 3514
Agrawal, Ameeta, 6208
Ahmed, Sajawel, 3753
Ahuja, Kabir, 4320
Ai, Xi, 4883
Aich, Ankit, 6586
Ainslie, Joshua, 4946
Aji, Alham Fikri, 1604, 2319
Akama, Reina, 660
Akter, Mousumi, 6195
Al Khatib, Khalid, 6897
Alabi, Jesujoba O., 4286, 4336
Alacam, Özge, 178
Alam, Firoj, 3127, 6625
Aletras, Nikolaos, 6656
Alghanmi, Israa, 1432
Alikhani, Malihe, 797, 6063, 6270
Almeida, Mariana, 1106
Alnajjar, Khalid, 6875
Alonso-Alonso, Iago, 5345
Alshomary, Milad, 344
Alvandipour, Mehrdad, 5755
Amin, Saadullah, 2259
Amplayo, Reinald Kim, 1051
An, Chenxin, 5783
Ananiadou, Sophia, 6259
Ando, Maya, 3769
Antoine, Jean-Yves, 3285
Antypas, Dimosthenis, 3386
Apostol, Elena-Simona, 4018
Araki, Jun, 1765
Arase, Yuki, 5240
Arnold, Thomas, 2927
Arora, Aseem, 6752
Arora, Jashn, 105, 116
Ashida, Mana, 3606
Assem, Haytham, 1089
Atapattu, Thushari, 6991
Atwell, Katherine, 797, 6063
Augenstein, Isabelle, 78, 1620
Austin, Eric, 971
Avagnano, Marco, 3465
Aw, Ai Ti, 580, 2801, 3924
Aydin, Gizem, 1937

Bâce, Mihai, 192
Badyal, Sahil, 6746
Baechler, Gilles, 5636
Baek, Woonhyuk, 5730
Baheti, Ashutosh, 3810
Bai, Liang, 2226
Bai, Long, 6105
Bai, Xuefeng, 592
Bai, Yinhao, 4728
Bai, Yu, 6355
Bajracharya, Adarsha, 6234
Baldwin, Timothy, 3427, 4172, 4228
Bali, Kalika, 4320
Balke, Wolf Tilo, 1311
Ballesteros, Miguel, 5882
Balzani, Lorenzo, 2692
Bansal, Mohit, 285, 4499
Bansal, Naman, 6195
Bao, Changcun, 6403
Bao, Forrest Sheng, 5896
Bao, Rong, 2873
Bao, Yongjun, 7002
Bapi, Raju S., 105, 116
Barbieri, Francesco, 3353, 3386
Bashier, Housam K. B., 994
Bashir, Syed Raza, 3778
Batura, Tatiana, 3420
Beck, Christin, 3074
Beeferman, Doug, 6818
Behera, Laxmidhar, 4071

Beigy, Hamid, 4306
Bell, Peter, 7171
Bergen, Benjamin K., 1
Berlowitz, Dan, 6234
Bernardi, Raffaella, 5597
Berry, Grant M., 5119
Bertolini, Lorenzo, 4084
Besacier, Laurent, 4274
Besançon, Romaric, 1455
Bettin, Dieter, 1925
Beuls, Katrien, 15
Beyer, Anne, 875
Bhardwaj, Rishabh, 4801
Bhatia, Riya, 134
Bhatnagar, Varad, 2934
Bhattacharya, Indrajit, 4537
Bhattacharya, Souvik, 6586
Bhattacharyya, Pushpak, 1098, 4136, 4442, 6165, 6752, 6978
Bhutani, Nikita, 1620
Bi, Guanqun, 163, 6480
Bialer, Amir, 4241
Bian, Jiang, 2598
Bibi, Khalil, 4707
Bing, Lidong, 4381, 5666, 6916
Bird, Steven, 4274
Biswas, Arijit, 285
Bladier, Tatiana, 4360
Blair, Alan, 542
Blanco, Eduardo, 2561
Blübaum, Lukas, 3296
Boese, Markus, 218
Bollegala, Danushka, 1299
Bondarenko, Alexander, 3296
Bos, Johan, 4186
Bose, Tulika, 6656
Botev, Georgie, 5309
Branco, António, 6859
Braslavski, Pavel, 3296
Bravo-Marquez, Felipe, 2108
Bruches, Elena, 3420
Brufau, Rita Brugarolas, 4575
Bruni, Elia, 5689
Bu, Jiahao, 3098
Buhmann, Jeska, 496, 3539
Bui, Khac-Hoai Nam, 5870, 6248
Bui, Trung, 1169, 2734, 6617, 7147
Bulling, Andreas, 192
Burzo, Mihai, 5613
Buys, Jan, 3187
Byamugisha, Joan, 4350
Cabello Piqueras, Laura, 3597
Cai, Chaoxiang, 5978
Cai, Fei, 2129
Cai, Guoyong GUET, 3029
Cai, Kaiwei, 2724
Cai, Li, 2075
Cai, Pengshan, 6234
Cai, Yi, 6795
Camacho-Collados, Jose, 3353, 3386
Cámbara, Guillermo, 7166
Cambria, Erik, 94
Campos, Viola, 1006
Cao, Biwei, 1677
Cao, Feiqi, 3438
Cao, Hailong, 5278
Cao, Hu, 1953
Cao, Jie, 1511
Cao, Jiuxin, 1677
Cao, Juan, 2834
Cao, Tianyang, 6327
Cao, Yanan, 510, 2504
Cao, Yixin, 2118
Cao, Yu, 5029
Cao, Yunbo, 3042
Cao, Zhao, 4952, 5560
Cao, Zheng, 553
Caragea, Cornelia, 2711, 2892, 3709
Caragea, Doina, 2711
Carbonaro, Antonella, 3465
Carpuat, Marine, 2668
Caselli, Tommaso, 3197
Castagnos, François, 4604
Castelli, Vittorio, 1065
Castellucci, Giuseppe, 3514
Castro, Santiago, 5613
Celikkanat, Hande, 4788
Cha, Taehun, 3847
Chae, Hyungjoo, 6285
Chakrabarty, Abhisek, 5014
Chakraborty, Tanmoy, 6625
Chan, May Pik Yu, 7177
Chandu, Khyathi Raghavi, 6089
Chang, Baobao, 2313, 6327
Chang, David, 2259
Chang, Ernie, 2681, 4286, 5876
Chang, Kalvin, 3563
Chang, Shiyu, 2157
Chang, Walter W., 2734, 7147
Chang, Xiaoqin, 7042
Chang, Yi, 2608
Changpinyo, Soravit, 6089

Chao, Chengpeng, 2147
Chatterjee, Oishik, 4753
Chauhan, Dushyant Singh, 6752
Chavez Sanchez, Liz Karen, 4258
Che, Wanxiang, 461, 3251, 3331, 5450, 7209
Chebrolu, Kameswari, 2934
Chen, Bolin, 4687
Chen, Borun, 3098
Chen, Bowen, 3210
Chen, Boxing, 6355
Chen, Chen, 4005
Chen, Daniel, 4028
Chen, David Z., 6746
Chen, Gang, 7203
Chen, Guanliang, 1275
Chen, Hechang, 2608
Chen, Honghui, 2129
Chen, Hongxu, 721
Chen, Hsin-Hsi, 2946
Chen, Huajun, 2374, 2551
Chen, Huang, 6403
Chen, Hui, 2551, 4622, 5666
Chen, Jiajun, 626, 5246, 6784
Chen, Jialin, 1467
Chen, Jiayi, 1502
Chen, Jindong, 5636
Chen, Jingqiang, 5978
Chen, Junfan, 1036
Chen, Kejia, 5978
Chen, Lei, 2748
Chen, Lisung, 7187
Chen, Meiqi, 2118
Chen, Meng, 3741
Chen, Ming, 5142
Chen, Minyu, 1580
Chen, Nancy F., 580, 3924
Chen, Nuo, 7187
Chen, Qiang, 960
Chen, Qianglong, 333
Chen, Qiguang, 461
Chen, Qin, 1990
Chen, Qingcai, 1896
Chen, Shaobin, 4821
Chen, Shunjie, 6955
Chen, Taiyan, 2011
Chen, Wei, 7013, 7093
Chen, Wei-Lin, 2946
chen, wen xiang, 2801
Chen, Wenliang, 960, 2044
Chen, Wenqing, 1708
Chen, Xi, 2639
Chen, Xiang, 2374
Chen, Xiuyi, 6762
Chen, Xuanting, 7053
Chen, Xuefei, 5335
Chen, Xueyuan, 7193
Chen, Yanan, 1157
Chen, Yang, 1791
Chen, Yi-Hsin, 1163
Chen, Yi-Shin, 1163
Chen, Yidong, 5266
Chen, Yige, 5432
Chen, Yiran, 7177
Chen, Yiren, 4372
Chen, Yubo, 1865, 1875, 2418
Chen, Yufeng, 5054
Chen, Yulong, 2166
Chen, Zhen, 1885
Chen, Zhigang, 3331, 3937
Chen, Zhijia, 3984
Chen, Zhiwei, 2199
Cheng, Lu, 1286
Cheng, Qiao, 5148
Cheng, Xueqi, 2065, 6105
Cheng, Yang, 134
Cheng, Yongjing, 1801
Cheng, Zhanzhan, 2034
Cheng, Zhicong, 1546
Cheng, Ziming, 5461
Cheung, Jackie Chi Kit, 779, 6019
Chhun, Cyril, 5794
Chi, Jie, 7171
Chi, Zewen, 570, 1732
Chiarcos, Christian, 4018
Chilimbi, Trishul, 1179
Chinthakindi, Sai Chetan, 6927
Chiril, Patricia, 3086
Cho, Itsugun, 367
Choe, June, 7177
Choi, Edward, 4853
Choi, Ho-Jin, 669
Choi, Jason Ingyu, 3514
Choi, Jinho D., 3360
Choi, Remi, 797
Choo, Jaegul, 4853
Choshen, Leshem, 4544
Choudhury, Monojit, 4320
Christ, Callista, 3086
Christodoulopoulos, Christos, 2319
Chu, Maximus H., 6505
Chu, Xiaomin, 3007
Chu, Zhixuan, 378

Chung, Tagyoung, 285
Churpek, Matthew M. M., 2979
Clavel, Chloé, 3950, 5794
Colas, Anthony, 5755
Collins, Marcus, 3514
Colombo, Pierre, 5794
Cong, Xin, 2300
Coto-Solano, Rolando, 4455
Cotterell, Ryan, 4651, 6007
Cresci, Stefano, 6625
Creutz, Mathias, 4788
Crowley, Archie, 1221
Cui, Chenxuan, 3563
Cui, Leyang, 5360
Cui, Shiyao, 2300
Cui, Xinyue, 5327
Cui, Yiming, 3937

Dabre, Raj, 5014
Dacon, Jamell, 1442
Daelemans, Walter, 496, 3539, 6837
Dai, Aijun, 3741
Dai, Chenchen, 960
Dai, Lu, 902
Dai, Quanyu, 2823
Dai, Xinyu, 626, 5246, 6784
Dai, Yinpei, 553
Dakota, Daniel, 4397
Dalvi, Fahim, 3127
Dan, Zhengcuo, 6457
Dang, Yixue, 7124
Das, Anjali, 3086
Das, Mayukh, 1311
Das, Souvik, 532, 887
Daskalaki, Elena, 4124
Datta, Siddhartha, 4832
Davis, Eric, 3697
De Bruyn, Maxime, 496, 3539
de Jong, Michiel, 4946
de Souza, Leandro Rodrigues, 3506
de Zoysa, Kasun, 6991
de Zoysa, Piyanjali, 6991
Dehaene, Stanislas, 3226
Deilamsalehy, Hanieh, 7147
Dell'Orletta, Felice, 3109, 3197
Demberg, Vera, 2681, 3659, 4049, 4286, 5876
Dementyeva, Yana, 3420
Deng, Kunquan, 2118
Deng, Naihao, 2166, 5613
Deng, Shumin, 2374
Deng, Zhenyun, 1791
Dernoncourt, Franck, 1169, 2346, 2734, 3309, 6617, 7147, 7156
Dervakos, Edmund G., 3639
Desbordes, Théo, 3226
Di Eugenio, Barbara, 694, 3709
Dimitrov, Dimiter, 6625
Ding, Chenchen, 5014
Ding, Haibo, 1765
Ding, Ke, 4575
Ding, Kunbo, 4372
Ding, Liang, 5029, 6691
Ding, Xiao, 3210
Ding, Yihao, 2906
Ding, Yuyang, 960
Ding, Zhuoye, 6050
Dinh, Dien, 5021
Dini, Irene, 3197
Dligach, Dmitriy, 2979
Do, Xuan Long, 580
Doan, Xuan-Dung, 5870
Dognin, Charles, 4604
Domogała, Aleksandra, 3631
Dong, Diwen, 4844
Dong, Guanting, 5327
Dong, Hongyuan, 3251
Dong, Jianbo, 553
Dong, Shenpo, 2001
Dong, Yao, 4918
Dong, Zhenhua, 938, 2363
Dou, Jiaheng, 4976
Dou, Shihan, 2278
Dou, Shuai, 6176
Dou, Yong, 1412, 1823, 2001
Dou, Zhicheng, 4952
Doumen, Jonas, 15
Dragut, Eduard, 3984
Drużyłowska, Dorota, 3631
D'Souza, Aminette, 3353
Du, Cunxiao, 5035
Du, Jinglong, 7013
Du, Jintao, 7065
Du, Li, 3210
Du, Ming, 1855
Du, Weihao, 2658
Dua, Radhika, 4853
Duan, Bin, 1834
Duan, Xingyi, 3331
Duan, Yitao, 5148
DuBois, Chris, 2861
Duda, Marko, 5689
Dunn, Jonathan, 26

Dunstan, Jocelyn, 2108
Durrani, Nadir, 3127
Dutta, Sourav, 1089

Eble, Alex, 3086
Eger, Steffen, 3401
Ekbal, Asif, 424, 1098, 6165, 6752, 6978
Elaraby, Mohamed, 6187
Elsafoury, Fatma, 1263
Elvitigala, Charitha, 6991
Eo, Sugyeong, 5181
Espinosa-Anke, Luis, 1432, 3353, 3971
Ettinger, Allyson, 863
Evang, Kilian, 4360
Eyers, David, 3731

Falkner, Katrina, 6991
Fan, Caoyun, 1708
Fan, Chenyou, 984
Fan, Chun, 6379, 6392
Fan, Hongcheng, 1943
Fan, Lu, 451
Fan, Shengda, 1904, 2247
Fan, Yaxin, 912
Fan, Yunlong, 3994
Fan, Zhihao, 1655
Fang, Anjie, 3798
Fang, Bin, 4883
Fang, Fang, 2504
Fang, Jun, 7002
Fang, Pengfei, 2992
Fang, Wei, 984
Fang, Yuejian, 4372
Fang, Zhou, 2405
Farcas, Emilia, 2734
Farrús, Mireia, 7166
Faulkner, Adam, 6746
Fazili, Barah, 4268
Fei, Hao, 1953, 6955
Fei, Nanyi, 5560
Fei, Zhaoye, 4952
Fei, Zichu, 6575
Feng, Fangxiang, 2651
Feng, Jiazhan, 389
Feng, Junlan, 707, 5327
Feng, Shanshan, 3064
Feng, Shi, 298, 648, 6906
Feng, Shutong, 266
Feng, Steven Y., 6270
Feng, Zihao, 5278
Ferret, Olivier, 1455
Fetahu, Besnik, 3798
Firdaus, Mauajama, 6165
Firooz, Hamed, 6625
Fohr, Dominique, 6656
Fox, Lior, 4544
Francis, Jonathan, 1534
Franz, Martin, 1065
Fraser, Alexander, 5191
Freitas, André, 1556
Frisoni, Giacomo, 2692, 3465
Fu, Guohong, 616, 4212, 5372
Fu, Hongfei, 1580
Fu, Jinlan, 471
Fu, Peng, 6944
Fu, Ruiliu, 4610
Fu, Xianghua, 6705
Fu, Yanggeng, 2199
Fu, Yanhe, 510
Fu, Yu, 5707
Fujihara, Riki, 851
Funakoshi, Kotaro, 5857
Fung, Yi R., 6927
Futrell, Richard, 44

Gabay, Simon, 3722
Gajbhiye, Amit, 3971
Gal, Kobi, 4241
Gallagher, Daniel, 4593
Galuscakova, Petra, 2668
Gan, Shaoduo, 1842
Gangal, Varun, 6270
Gangi Reddy, Revanth, 1065, 6927
Gao, Chaochen, 3898, 4902
Gao, Hang, 4728
Gao, Jiale, 5054
Gao, Jian, 6827
Gao, Li, 4687
Gao, Miao, 3994
Gao, Mingfei, 2141
Gao, Qingqing, 1677
Gao, Shengxiang, 2237
Gao, Shuyang, 285
Gao, SongYang, 2278
Gao, Tianhao, 7002
Gao, Tianyu, 4965
Gao, Wei, 721
Gao, Wenyang, 5360
Gao, Xin, 7177
Gao, Xing, 333, 378
Gao, Yang, 6355
Gao, Yanjun, 2979
Gao, Ze-Feng, 3263
Gao, Zhiqiang, 3994

Garí Soler, Aina, 3950
Gasevic, Dragan, 1275
Gašić, Milica, 266
Ge, Feng, 6795
Ge, Ling, 1036
Ge, Shen, 2952
Geishauser, Christian, 266
Geng, Ruiying, 6115
Geng, Xinwei, 5217
Geng, Xiubo, 1333
Geng, Zhichao, 6540
Gerber, Ben, 694
Gers, Felix, 4765
Ghaddar, Abbas, 4707
Ghodsi, Ali, 4714
Ghosh, Soumitra, 1098, 6978
Ghosh, Subhasish, 4537
Glavaš, Goran, 1071
Goebel, Randy, 994
Goel, Rahul, 310
Gollapalli, Sujatha Das, 3835
Gómez-Rodríguez, Carlos, 5345
Gomez, Hector, 4429
Gong, Huanli, 5925
Gong, Linyuan, 1965
Gong, Tieliang, 2515
González, José Ángel, 6019
Goyal, Pawan, 4071, 4251
Graciarena, Martin, 1233
Grau-Bove, Josep, 1203
Grissom II, Alvin, 5175
Groh, Georg, 4593
Gromann, Dagmar, 5571
Grudzińska, Justyna, 3960
Grundmann, Paul, 4765
Gu, Xiaodong, 5521, 6369
Gu, Yu, 1718
Gu, Zhouhong, 5419
Guan, Saiping, 2065
Guan, Xin, 1677
Guerin, Frank, 6468
Gui, Tao, 1666, 2034, 2873, 6575, 6966, 7053
Gui, Xiangyu, 4892
Gui, Xinning, 3550
Gui, Yaocheng, 3994
Gunawardana, Kasun, 6991
Guo, Bin, 505
Guo, Chenlei, 505
Guo, Daichi, 5327
Guo, Danfeng, 285
Guo, Fang, 7099
Guo, Jiafeng, 2065, 2438
Guo, Junjun, 1146, 5098
Guo, Li, 510
Guo, Ping, 2504
Guo, Xiao-Yu, 1502
Guo, Xiaobo, 4918
Guo, Yanqing, 5904
Guo, Yike, 2226
Guo, Yuhui, 4517
Guo, Zhijiang, 5707
Gupta, Aditya, 310
Gupta, Ajay, 2861
Gupta, Arpit, 285
Gupta, Ashim, 4251
Gupta, Ashish, 4071
Gupta, Avani, 4528
Gupta, Manish, 105, 116
Gupta, Prabhakar, 5119

Ha, Jung-Woo, 6369
Habernal, Ivan, 2927
Haffari, Gholamreza, 1502, 2849
Hagen, Matthias, 1925, 3296
Hagström, Lovisa, 5582
Hahn, Joonghyuk, 5966
Hai, Zhen, 5666
Haidar, Md Akmal, 4707
Hämäläinen, Mika, 6875
Hamza, Wael, 218
Han, Jizhong, 3898, 4902
Han, Sooji, 94
Han, Soyeon Caren, 2906, 3438, 3530
Han, Wei, 4622, 5666
Han, Xiaoyun, 6680
Han, Yo-Sub, 5966, 6667
Han, Yuqiang, 721
Hanna, Michael, 5597
Hao, Chao, 3233
Hao, Dezhi, 3342
Hao, Wenjie, 3885
Haque, Mirazul, 3480
Harrison, Emileigh, 3086
Hashimoto, Atsushi, 3570
Hasibi, Faegheh, 1937
Hassan, Sabit, 6063
Hayashi, Yoshihiko, 6151
He, Dongchen, 6075
He, Hao, 1708
He, Jun, 2188
He, Kai, 2515
He, Keqing, 608, 707, 5327
He, Lejian, 4557

He, Liang, 1990, 4821
He, Liangliang, 6222
He, Lihong, 3984
He, Ruidan, 6916
He, Ruifang, 400, 745
He, Shiming, 2633
He, Shizhu, 1379, 2237
He, Wanwei, 553
He, Xiaodong, 616, 3741
He, Xuanli, 2849
He, Yaqiong, 3007
He, Yongquan, 2147
He, Youbiao, 5896
He, Zhenfeng, 721
He, Zhongshi, 7013
Heafield, Kenneth, 2668
Heck, Michael, 266
Heindorf, Stefan, 3296
Herath, Mahen, 6991
Hewett, Freya, 756
Hidey, Chris, 310
Higashinaka, Ryuichiro, 242
Hiraoka, Tatsuya, 4864
Holliday, Nicole, 7177
Hong, Jimin, 4853
Hong, Yu, 2633, 4160, 4481
Hoover, Jacob Louis, 44
Hou, Lei, 6315
Hou, Wenxin, 7075
Hou, Yuexian, 400, 745
Hou, Yufang, 766
Hou, Yutai, 3251
Hovy, Dirk, 1221
Hovy, Eduard, 6270
Hsiao, Yu-Chung, 5636
Hsieh, Shu-Kai, 152
Hu, Baotian, 1896
Hu, Biao, 1823
Hu, Bojie, 5109
Hu, Chengwei, 1885
Hu, ChunMing, 1036
Hu, Cong, 984
Hu, Dou, 2759, 2769
Hu, Guoping, 3331
Hu, Hengchang, 5925
Hu, Honghui, 4873
Hu, Jingwen, 684
Hu, Mengting, 4728
Hu, Minghao, 1823
Hu, Po, 1979
Hu, Songlin, 2759, 2769, 3898, 4902, 7136
Hu, Wei, 5532
Hu, Xiaohui, 2450
Hu, Xixin, 1687
Hu, Xuming, 2398, 5707
Hu, Yinan, 6355
Hu, Yue, 163, 6480
Huang, Fei, 553, 2374, 5298
Huang, Guanhuan, 323
Huang, H. Howie, 4585
Huang, Hen-Hsen, 2946
Huang, Heyan, 570, 1732, 6355
Huang, Hu, 6705
Huang, Hui, 5054
Huang, Jimin, 6259
Huang, Jin, 5148
Huang, Junyang, 5419
Huang, Kun, 400, 745
Huang, Kung-Hsiang, 1024
Huang, Lanxiao, 2962
Huang, Lifu, 2157
Huang, Pingxuan, 5613
Huang, Sean S., 3709
Huang, Shao-Lun, 2639
Huang, Shijue, 461
Huang, Shizhou, 1855
Huang, Shujian, 626, 5246, 6784
Huang, Wei, 1116
Huang, Xiusheng, 2418
Huang, Xu, 6705
Huang, Xuanjing, 1655, 1990, 2278, 2748, 2873, 5783, 6540, 6575, 6966, 7053, 7093
Huang, Yao, 2328
Huang, Yen-Hao, 1163
Huang, Yi, 707, 5327
Huang, Yucheng, 2515
Huang, Yujing, 3412
Huang, Yujun, 7093
Huang, Zhen, 1412, 1823, 6680
Huang, Zhichao, 6705
Huang, Ziming, 3064
Huang, Ziyin, 2528
Hudson, Aaron, 6208
Hui, Binyuan, 553, 5298
Hupkes, Dieuwke, 3226
Huryn, Daniil, 3360
Hutsell, William M., 3360
Huynh, Tin Van, 3858
Hwang, Seonjeong, 1636
Hwang, Seung-won, 6285
Hwaszcz, Krzysztof, 3631

Igamberdiev, Timour, 2927

Ihori, Mana, 6128
Ilievski, Filip, 1534
Illina, Irina, 6656
Inui, Kentaro, 660, 851, 5383
Ivoylova, Alexandra, 4468
Izmaylov, Daniel, 4241

Jafari, Aref, 4714
Jain, Mohit, 4320
Jain, Rajiv, 3309, 7147
Jang, Myeongjun, 3680, 3697
Jatowt, Adam, 6139
Jeon, DongHyeon, 1367
Jeon, Hyunmin, 355
Jetter, Antonie, 6208
Jezabek, Jan, 4640
Jhamtani, Harsh, 6270
Ji, Bin, 1842
Ji, Donghong, 1953, 6955
Ji, Heng, 1024, 1065, 6927
Ji, Huishan, 6944
Ji, Jianmin, 3164
Ji, Junzhong, 2388
Ji, Ran, 3164
Ji, Shihao, 1569
Ji, Yuqiu, 5450
Jia, Xin, 6564
Jiang, Changjian, 2968
Jiang, Daxin, 1333
Jiang, Dazhi, 6934
Jiang, Feng, 1904, 3007
Jiang, Han, 6426
Jiang, Hang, 6818
Jiang, Hao, 5560
Jiang, Jianhui, 1014, 2328
Jiang, Jie, 5560
Jiang, Jing, 5035
Jiang, Lei, 3788
Jiang, Ning, 5721
Jiang, Weifeng, 6447
Jiang, Wenbin, 1480, 4203
Jiang, Xiaojian, 1865, 1875
Jiang, Xiaorui, 5978
Jiang, Xin, 636
Jiang, Xuhui, 6105
Jiang, Yong, 2398, 4212
Jiang, Yufan, 6415
Jiang, Zhengbao, 1765
Jiang, Zhuoxuan, 3064
Jiao, Chengbo, 2450
Jin, Hai, 4892
Jin, Haoliang, 1885
Jin, Langjunqing, 4892
Jin, Qin, 684
Jin, Renren, 5158
Jin, Xiaolong, 2065
Jin, Yaohui, 1708
Jin, Yifan, 2450
Jing, Liwen, 6705
Jing, Ning, 2405
Jo, Bogeun, 3530
Jo, Daejin, 5837
Jo, Eunkyul Leah, 5432
Johansson, Richard, 5582
Johnson, Kristen Marie, 2780
Joo, Se June, 6285
Joty, Shafiq, 580, 4381
Ju, Dongshi, 298
Ju, Meizhi, 1422
Ju, Yiming, 1697
Jung, Jaeheun, 3847
Jyothi, Preethi, 4268, 4442

Kabashi, Besim, 4018
Kabbara, Jad, 779
Kajiwara, Tomoyuki, 5240
Kalim, Waqar Bin, 2357
Kallmeyer, Laura, 4360
Kalouli, Aikaterini-Lida, 3074
Kameko, Hirotaka, 3570
Kan, Baoshuo, 4061
Kan, Min-Yen, 471
Kaneko, Masahiro, 1299, 3578
Kang, Inho, 1367
Kang, Liyan, 5266
Kang, Yulin, 378
Kanojia, Diptesh, 2934
Kao, Jiun-Yu, 285
Kapoor, Alok, 6234
Kar, Sudipta, 3798
Karmaker Santu, Shubhra Kanti, 6195
Karpinska, Marzena, 141
Käser, Tanja, 1344
Katsigiannis, Stamos, 1263
Kawano, Keisuke, 3175
Kawasaki, Yoshifumi, 141
Kazmi, Arman, 922
Kędzierska, Hanna, 3631
Keh, Sedrick Scott, 6270
Khadivi, Shahram, 4933
Khalid, Osama, 55
Khan, Abdul Rafae, 3908
Khan, Md Fahim Faysal, 4575
Khanna, Vishal, 6897

Khebour, Ibrahim, 4996
Khorashani, Elahe, 1593
Kim, Chaehyeong, 6285
Kim, Dohyung, 3697
Kim, Eun-Sol, 5837
Kim, Gyeongmin, 3496, 5181
Kim, Jinsung, 412, 3496
Kim, Jiyun, 6644
Kim, Mi-Young, 994
Kim, Michelle YoungJin, 2780
Kim, Nayoung, 1286
Kim, Saehoon, 5730
Kim, Sanghee J., 863
Kim, Seonhoon, 1367
Kim, Seungone, 6285
Kim, Sungwoong, 5837
Kim, TaeHee, 4853
Kim, Taeuk, 5398
Kim, Yangsok, 3530
Kim, Yejin, 4585
Kim, Youngmin, 3563
Kim, Youngwook, 6667
Kishinami, Yosuke, 660
Kißler, Johanna, 178
Klakow, Dietrich, 4336
Klinger, Roman, 6805
Klyus, Julia, 3631
Knaebel, René, 837
Kobayashi, Hideo, 766
Kobayashi, Tetsunori, 6151
Kogut, Bruce, 3589
Koide, Satoshi, 3175
Koller, Alexander, 4049
Kong, Dejiang, 4952
Kong, Fang, 912
Kong, Jun, 4677
Kong, Li, 2724
Kordjamshidi, Parisa, 1357, 5745
Koto, Fajri, 3427
Kou, Ziyi, 1776
Kovatchev, Venelin, 3873
Kreutz, Tim, 6837
Krishna, Amrith, 4251
Krishnaswamy, Nikhil, 4996
Krstovski, Kriste, 3589
Kruengkrai, Canasai, 4568
Kruse, Carl, 3753
Kübler, Sandra, 4397
Kuhlmann, Marco, 4664
Kumar, Vishwajeet, 4753
Kundu, Arpita, 4537
Kundu, Rohit, 4442
Kunz, Jenny, 4664
Kuribayashi, Tatsuki, 851
Kurimo, Mikko, 6875
Kurita, Shuhei, 3570, 5383
Kuroda, Yuto, 5240
Kurohashi, Sadao, 812
Kuzi, Saar, 3514
Kwon, Deuk Sin, 3680, 3697
Kwon, Taehwan, 5837

Laaksonen, Jorma, 6875
Labeau, Matthieu, 3950
Ladhak, Faisal, 2668
Lai, Huiyuan, 5939
Lai, Peichao, 2199
Lai, Ruofei, 4952
Lai, Viet Dac, 2346, 7156
Lai, Wen, 5191
Lai, Yantong, 2769
Lakretz, Yair, 3226
Lam, Albert Y.S., 451
Lam, Kwok-Yan, 4005
Lan, Man, 2075, 2528
Landes, Paul, 3709
Langlais, Philippe, 3454
Langlais, Phillippe, 4707
Largeron, Christine, 971
Lasri, Karim, 37
Lau, Jey Han, 3427, 4172
Lauscher, Anne, 1221
Le Ferrand, Éric, 4274
Lebani, Gianluca, 4101
Leczkowski, Aleksander, 3960
Lee, Byounghan, 6644
Lee, Chonghan, 4575
Lee, Donghun, 3847
Lee, Fei-Tzin, 5882
Lee, Gary Geunbae, 355, 1636
Lee, Heejun, 3530
Lee, Hodong, 4585
Lee, Jean, 3530
Lee, Jihwa, 1083
Lee, John S. Y., 6002
Lee, Jong-Hyeon, 1367
Lee, Jungseob, 5181
Lee, Minho, 6301
Lee, Sang-Woo, 1051, 6369
Lee, Seungyeon, 6301
Lee, Sophia Yat Mei, 7042
Lee, Youhan, 5730
Lee, Young-Jun, 669

Lefever, Els, 4434
Lei, Fangyu, 1379
Lei, Hao, 5327
Lei, Jinhao, 2861
Lei, Shun, 7193
Lei, Tianyi, 4873
Lei, Yikun, 5205
Lei, Zhilin, 5450
Lemmens, Jens, 6837
Lenci, Alessandro, 37, 4101
Lenskiy, Artem, 4124
Levi-Belz, Yossi, 4241
Levy, David, 6234
Li, Aini, 7177
Li, An-Ran, 824
Li, Bin, 3994
Li, Bing, 4005
Li, Binhua, 5298, 6115
Li, Binyang, 6680
Li, Bo, 1391
Li, Bobo, 1953
Li, Bohan, 3251
Li, Bowen, 5298, 6115
Li, Chao, 2494, 2962, 3042
Li, Chen, 2515
Li, Chenliang, 1146
Li, Dong, 6222
Li, Dongfang, 1896
Li, Dongsheng, 6680
Li, Dongyuan, 5857, 5989
Li, Fei, 1953, 6955
Li, Guanying, 2748
Li, Guodun, 333
Li, Guohui, 1801
Li, Guoqiang, 1580
Li, Haochen, 1943
Li, Haodong, 5678
Li, Hebi, 5896
Li, Jiangmeng, 2450
Li, Jiangnan, 253, 6944
Li, Jianjun, 1801
Li, Jianxin, 6447
Li, Jiawei, 6355
Li, Jing, 6415
LI, Jingyang, 1580
Li, Jingye, 1953, 6955
Li, Jintao, 2834
Li, Jiwei, 2188, 6379, 6392
Li, Juan, 1546
Li, Juanzi, 6315
Li, Juntao, 960, 3064
Li, Junyi, 6340
Li, Junyi Jessy, 797, 6505
Li, Lei, 2374, 2658
Li, Liang, 6115
Li, Lishuang, 2098
Li, Longfeng, 6887
Li, Min, 2962
Li, Mingchen, 1569
Li, Mingda, 485
Li, Mu, 6415
Li, Mukai, 2118
Li, Peifeng, 912, 1915, 2087, 2622, 3007
Li, Peng, 1979
Li, Piji, 510, 2812
Li, Qiuxia, 1865, 1875
Li, Quangang, 2300
Li, Rui, 6934
Li, Ruichen, 684
Li, Ruifan, 2651
Li, Shasha, 1842, 6222
Li, Shengjie, 6050
Li, Shoushan, 3274, 7042
Li, Stan Z., 7112
Li, Sujian, 3374, 4203
Li, Wei, 6105
Li, Weiping, 1813, 1943
Li, Weizhao, 6795
Li, Wing Yan, 1128
Li, Xiang, 1379, 2438, 5266
Li, Ximing, 6691
Li, Xuefeng, 5327
Li, Yafu, 5227
Li, Yan, 1146
Li, Yangguang, 1412
Li, Yi, 5904
Li, Ying, 521
Li, Yinzi, 7093
Li, Yitian, 1708
Li, Yitong, 636
Li, Yiyang, 3017
Li, Yongbin, 553, 5298, 6115
Li, Yuan-Fang, 1502
Li, Yucheng, 6468
Li, Yudong, 3917
Li, Yuheng, 1275
Li, Yunpeng, 6480
Li, Yuntao, 1645
Li, Zhen, 389
Li, Zhenghua, 4160
Li, Zhongli, 3042
Li, Zhongqiu, 2633

Li, Zhoujun, 5085, 6415
Li, Zicheng, 3274
Li, Zihui, 4996
Li, Ziming, 7136
Li, Zuchao, 2472, 5461
Lian, Ruiyuan, 2098
Lian, Zheng, 2450, 7136
Liang, Bin, 7124
Liang, Di, 1645
Liang, Haijin, 2405
Liang, Haowen, 451
Liang, Xinnian, 6415
Liang, Xun, 4517
Liang, Zhongping, 2962
Liao, Huaming, 2438
Liao, Jianxin, 6887
Libovický, Jindřich, 5191
Liesaputra, Veronica, 3731
Lim, Chae-Gyun, 669
Lim, Heuiseok, 412, 3496, 4585, 5181
Lim, Ho Hung, 6002
Lim, Jungwoo, 412
Lim, KyungTae, 5432, 5730
Lim, Taejun, 3530
Lin, Chenghua, 6468
Lin, Chi, 3550
Lin, Chien-Wei, 285
Lin, Hongfei, 2024, 3342
Lin, Hongzhan, 2608
Lin, Hsien-chin, 266
Lin, Jiaxing, 2313
Lin, Junxian, 3550
Lin, Min, 3937
Lin, Yankai, 4965
Lin, Zhe, 6530
Lin, Zheng, 253, 510, 2504, 6944
Lin, Zhenxi, 2572
Lin, Zijie, 7124
Lion-Bouton, Adam, 3285
Litman, Diane, 6187
Litschko, Robert, 1071
Liu, Ao, 7075
Liu, Bing, 694
Liu, Bo, 1677, 4557, 4844
Liu, Chang, 389
Liu, Chao, 7002
Liu, Chen, 1116
Liu, Cheng, 6934
Liu, Chi, 1422
Liu, Daizong, 5532
Liu, Duanchen, 3412
Liu, Dugang, 2658
Liu, Fei, 6234
Liu, Gongshen, 6493, 7065
Liu, Guiquan, 1116
Liu, Guisheng, 5904
Liu, Hanyu, 7002
Liu, Haochen, 1442
Liu, Haoyan, 4372
Liu, Hong, 1480
Liu, Hongzhi, 3042
Liu, Huan, 1286
Liu, Huijun, 1842
Liu, Jian, 5054
Liu, Jianfeng, 4492
Liu, Jiayi, 378
Liu, Jie, 4687
Liu, Jigang, 2962
Liu, JiHong, 1036
Liu, Jin, 636
Liu, Jingping, 5419
Liu, Junnan, 6447
Liu, Junpeng, 6050
Liu, Kang, 1379, 1697, 1865, 1875, 2237, 2418
Liu, Kesong, 1014
Liu, Lijuan, 5636
Liu, Linlin, 4381
Liu, Mingming, 4728
Liu, Minqian, 2157
Liu, Peiyang, 2210
Liu, Peiyu, 3263
Liu, Pengyuan, 1250
Liu, Pengzhang, 7002
Liu, Qian, 461
Liu, Qijiong, 2823
Liu, Qin, 2873
Liu, Qun, 636
Liu, Rui, 6944
Liu, Ruixue, 3741
Liu, Ruri, 2247
Liu, Shuaipeng, 2483
Liu, Sisi, 6457
Liu, Tie-Yan, 2598
Liu, Ting, 485, 1422, 3210
Liu, Tingwen, 2300
Liu, Weifeng, 5029
Liu, Weijie, 3917, 4372
Liu, Weiqing, 2598
Liu, Weisong, 6234
Liu, Xiaochen, 6355
Liu, Xiaohu, 505
Liu, Xiaoyuan, 4844

Liu, Xingxian, 1834
Liu, Xinyao, 4517
Liu, Xueqing, 3480
Liu, Xuyuan, 3054
Liu, Yang, 1157
Liu, Yaxin, 7136
Liu, Yi, 2504
Liu, Ying, 2388, 4149
Liu, Yongkang, 648
Liu, Yuchen, 684
Liu, Yufang, 2528
Liu, Zejie, 1136
Liu, Zheng, 4952
Liu, Zhengyuan, 3924
Liu, Zhiyuan, 4965, 7002
Liu, Zhiyue, 7031
Liu, zhuo han, 2801
Liu, Zichen, 3054
Liu, Zihan, 7099, 7112
Liu, Ziwen, 1203
Liu, Ziyi, 3042
Liu, Zoey, 3412
Loáiciga, Sharid, 875
Lobov, Valeriy, 4468
Long, Siqu, 2906, 3438
Long, Siyu, 6784
Long, Yunfei, 7124
Longpre, Shayne, 2861
Löser, Alexander, 4765
Lotfi, Ehsan, 496, 3539
Lotufo, Roberto, 3506
Louis, Annie, 6019
Loureiro, Daniel, 3353
Lovon-Melgarejo, Jesus, 1455
Lu, Bo-Han, 4418
Lu, Heng-yang, 984
Lu, Hongbin, 2098
Lu, Junyu, 2055
Lu, Kevin, 6270
Lu, Menghua, 6493
Lu, Menglong, 6680
Lu, Sin-En, 4418
Lu, Siyuan, 6075
Lu, Wenjie, 3143
Lu, Wenpeng, 4061
Lu, Xuantao, 5419
Lu, Yanyue, 6846
Lu, Zhiwu, 5560
Lu, Zhong-Yi, 3263
Lubis, Nurul, 266
Lukasiewicz, Thomas, 3680
Luo, Ge, 5896
Luo, Junyu, 3550
Luo, Qiaoyang, 4873
Luo, Siwen, 2906
Luo, Xiangyang, 5904
Luo, Yingwei, 1546
Luo, Yu, 2405
Luo, Yun, 7099, 7112
Luo, Yutao, 6493
Luo, Yutian, 5560
Luo, Zhiping, 2598
Luo, Ziyang, 2608
Luque, Jordi, 7166
Lv, Jianwei, 2483
Lv, Pengcheng, 298
Lv, Zhilong, 6680
Lwowski, Brandon, 6600
Lymperaiou, Maria, 3639
Lyu, Feifei, 1014
Lyu, Lingjuan, 2849
Lyu, Xiaoqing, 5335
Lyu, Yajuan, 1480, 4203

Ma, Can, 6115
Ma, Chen, 1580
Ma, Fang, 6736
Ma, Fenglong, 3342, 3550
Ma, Guanghui, 1036
Ma, Jing, 2608
Ma, Jun, 1842
Ma, Kaixin, 1534
Ma, Longxuan, 485
Ma, Meirong, 2075
Ma, Mian, 6050
Ma, Mina, 3042
Ma, Ruotian, 7053
Ma, Shuming, 5085
Ma, Wei, 2504
Ma, Xinge, 4907
Ma, Yinglong, 7086
Ma, Zhanyu, 4492
Ma, Zhiyuan, 1801
Madhyastha, Pranava, 5488
Maeda, Koki, 3578
Maharana, Adyasha, 4499
Mahdipour Saravani, Sina, 4996
Malmasi, Shervin, 3514, 3798
Manjunatha, Varun, 7147
Mannan, Sheikh, 4996
Manoliadis, George, 3639
Manotas, Irene, 1593
Mansur, Mairgup, 6327

Mao, Jianguo, 1480
Mao, Qianren, 6447
Mao, Rui, 94, 2515
Mao, Weiquan, 3917, 4372
Mao, Xian-Ling, 570, 1732, 3064
Mao, Xin, 2075
Marchal, Marian, 3659
Marin, Alex, 2681, 5876
Markert, Katja, 3669, 6035
Markowska, Magdalena, 786
Martín-Valdivia, María-Teresa, 6805
Martino, Giovanni Da San, 6625
Martins, Bruno, 1106
Massie, Stewart, 5770
Masumura, Ryo, 6128
Matsuda, Koji, 3769
Matsumoto, Yuji, 4172, 5914
Maudslay, Rowan Hall, 65
Maurya, Dhirendra Kumar, 1098
McCarthy, Arya D., 5309
McKeown, Kathleen, 2668, 5882
Mehler, Alexander, 3753
Meister, Nicole, 3309
Melgarejo, Nelsi, 4429
Melser, Anton, 6002
Meng, Fandong, 253, 5227, 6762
Meng, Helen, 7193
Meng, Kui, 7065
Meng, Xuying, 2438
Meng, Yuxian, 2188, 6379, 6392
Menis Mastromichalakis, Orfeas, 3639
Mercer, Robert E., 2357
Merendi, Federica, 3109
Mesquita, Tiago, 1106
Mezza, Stefano, 542
Mi, Fei, 636
Mi, Jiaxin, 1979
Miao, Zhongchen, 6827
Miao, Zhongjian, 5266
Michaelov, James A., 1
Mihalcea, Rada, 5507, 5613
Mihelich, Martin, 4604
Mikoś, Daria, 3631
Miller, Timothy, 2979
Minervini, Pasquale, 2259
Ming, Zhong, 2658
Mishra, Kshitij, 424
Mo, Shasha, 1904, 2247
Mo, Tong, 1813, 1943, 2483
Mojab, Mohammad Hassan, 4306
Mompelat, Ludovic, 4397
Mondal, Ishani, 4320
Moon, Hyeonseok, 5181
Moosavi, Nafise Sadat, 3401
Moreno-Noguer, Francesc, 5488
Moreno, Jose G., 1455
Mori, Shinsuke, 3570
Moro, Gianluca, 2692, 3465
Mortensen, David R., 3563
Mosbach, Marius, 4336
Mosca, Edoardo, 4593
Mou, Xiaofeng, 2528
Mou, Xinyi, 2968
Mou, Yutao, 608, 707
Movva, Rajiv, 2861
Mrini, Khalil, 2734
Mu, Lingling, 3885
Muhajab, Areej Nasser, 3353
Mukherjee, Arjun, 3984
Murzaku, John, 786
Mussabayev, Rustam, 5288

Na, Seung-Hoon, 1367
Nagata, Ryo, 141
Nakamura, Akihiro, 3175
Nakashole, Ndapa, 2734
Nakayama, Hideki, 1743
Nakayama, Kouta, 3769
Nakov, Preslav, 6625
Nan, Feng, 5882
Nan, Qiong, 2834
Nandigam, Poojitha, 2220
Narayanan, Vijaykrishnan, 4575
Naseem, Usman, 3778
Nath, Abhijnan, 4996
Nayak, Tapas, 4537
Nelakanti, Anil, 5119
Neubig, Graham, 1765
Neumann, Guenter, 2259
Nevens, Jens, 15
Neves, Leonardo, 3353, 3386
Ng, Hwee Tou, 2794, 6916
Ng, See-Kiong, 471, 3835
Ng, Vincent, 766
Ngonga Ngomo, Axel-Cyrille, 3296
Nguyen, Binh, 5021
Nguyen, Kiet Van, 3858
Nguyen, Le-Minh, 5870
Nguyen, Long, 5021
Nguyen, Minh Van, 2346
Nguyen, Ngan Luu-Thuy, 3858
Nguyen, Ngoc-Dung Ngoc, 6248
Nguyen, Thien Hai, 4381

Nguyen, Thien Huu, 2346, 3309, 4741, 7147, 7156
Ni, Shikang, 3924
Nicolai, Garrett, 4292
Nie, Jian-Yun, 6315
Nie, Yuxiang, 1732
Ning, Jinzhong, 2024
Ninomiya, Takashi, 5240
Nishimura, Taichi, 3570
Nissim, Malvina, 5939
Niu, Guanglin, 1391
Niu, Hao, 6827
Niu, Jianwei, 1904, 2247
Niu, Jingcheng, 3143
Niu, Shuai, 2226
Nogueira, Rodrigo, 3506
Nomoto, Masako, 3769
North, Kai, 6057
Nyberg, Eric, 1534

Oard, Douglas, 2668
Oberhauser, Tom, 4765
O'Donnell, Timothy J., 44
Oh, Alice, 1169
Oh, Dongsuk, 4585
Oh, Geunseob, 310
Ohashi, Atsumoto, 242
Ohmer, Xenia, 5689
Okazaki, Naoaki, 1299, 3578
Okumura, Manabu, 5857, 5989, 7075
Olabisi, Olubusayo, 6208
Oleksy, Marcin, 3631
Oltramari, Alessandro, 1534
Omura, Kazumasa, 812
Oncevay, Arturo, 4258
O'Neill, Jacki, 4320
Oota, Subba Reddy, 105, 116
Oprea, Silviu Vlad, 1089
Orr, Scott Allan Allan, 1203
Ortega, John E., 4429
Ortiz Suarez, Pedro, 3722
Ou, Zebin, 6516
Ouchi, Hiroki, 5383
Ouyang, Yawen, 626
Ouyang, Zhenqiu, 721
Øvrelid, Lilja, 6773
Owens, Andrew, 5507
Özsoy, Ömer, 3753
Ozturk, Yagmur, 3285

Padró, Lluís, 5488
Pagnoni, Artidoro, 1233
Palmer, Alexis, 4028
Pan, Liangming, 580, 5925
Pan, Weike, 2658
Pan, Zhaoming, 1116
Pandey, Amit, 4528
Pandey, Isha, 4753
Pang, Bo, 5651
Pang, Kunyuan, 6222
Pang, Liang, 2363
Pang, Yunhe, 7086
Parameswaran, Pradeesh, 3731
Parde, Natalie, 6586
Park, ChaeHun, 4853
Park, Chanjun, 5181
Park, Eunhwan, 1367
Park, Jungyeul, 5432
Park, Seongmin, 1083
Park, Shinwoo, 6667
Pasand, Ali Saheb, 4714
Passban, Peyman, 5076
Pasupat, Panupong, 4946
Patel, Kunal, 3709
Paul, Shachi, 310
Pedeni, Federico, 5597
Peng, Dezhong, 4873
Peng, Jiajie, 2968
Peng, Nanyun, 6379
Peng, Qiwei, 4113
Peng, Wei, 163, 5278, 6480
Peng, Weihua, 1480
Peng, Xingyu, 950
Peng, Xueping, 4061
Penn, Gerald, 3143
Petitjean, Simon, 4360
Pham, Tuan-Hung, 505
Phan, Tuan-Anh, 6248
Piedboeuf, Frédéric, 3454
Pivovarova, Lidia, 4037
Plaza-del-Arco, Flor Miriam, 6805
Poddar, Lahari, 441
Poelman, Wessel, 4186
Poibeau, Thierry, 37
Poon, Josiah, 2906, 3438
Popa, Alin-Ionut, 4633
Popescu, Octavian, 1593
Poria, Soujanya, 4622, 4801, 5666
Potthast, Martin, 1925, 3296
Poupart, Pascal, 4707
Pradhan, Richeek, 6234
Proietti, Mattia, 4101
Prud'hommeaux, Emily, 3412
Pu, Pearl, 3315

Pu, Shiliang, 1391, 2034
Pudi, Vikram, 4528
Pustejovsky, James, 1521

Qi, Guilin, 2288
Qi, Qi, 6887
Qi, Qianqian, 1791
Qi, Xiaofeng, 2962
Qian, Zhong, 2622
Qiao, Dan, 960
Qiao, Liang, 2034
Qiao, Zile, 1813
Qin, Bing, 1422, 3210, 6846, 7020
Qin, Bowen, 5298
Qin, Libo, 461
Qin, Wenyi, 1036
Qiu, Han, 2188
Qiu, Hangping, 3233
Qiu, Linlu, 4946
Qiu, Xipeng, 4952, 5783, 6540
Qiu, Yunqi, 6105
Qiu, Zhaopeng, 2952
Qorib, Muhammad Reza, 2794
Qu, Fanyi, 6564
Qu, Lizhen, 2849
Qu, Weiguang, 2724
Qu, Yuzhong, 1687
Qu, Zhi, 5251
Quan, Xiaojun, 323
Quteineh, Husam, 5955

Rad, Paul, 6600
Rajkumar, Rajakrishnan, 922
Ralethe, Sello, 3187
Ramaiah, Chetan, 2141
Ramakrishnan, Ganesh, 4753
Rambow, Owen, 786
Ramzan, Naeem, 1263
Ranasinghe, Tharindu, 6057
Ranjan, Sidharth, 922
Rao, Yanghui, 2585
Rao, Ziqin, 2651
Ravishankar, Vinit, 4788
Rawat, Danda, 6617
Rawski, Jonathan, 6007
Ray Choudhury, Sagnik, 78, 1620
Rayaprolu, Nikhil, 2220
Raza, Shaina, 3778
Rebedea, Traian, 2711
Reddy, Sreedhar, 4136
Rehman, Misbahur, 3753
Reinspach, Julia, 441
Ren, Bo, 6403
Ren, Haopeng, 6795
Ren, Hongrun, 6827
Ren, Lei, 6736
Ren, Yubing, 2504
Ren, Yuqi, 5205
Rezagholizadeh, Mehdi, 4707, 4714, 5076
Riddle, Patricia, 1791
Rietsche, Roman, 1344
Rim, Kyeongmin, 1521
Rios, Anthony, 6600
Ritter, Alan, 3810
Rivas Rojas, Kervy Dante, 4258
Rodrigues, João António, 6859
Rogers, Anna, 78
Roh, Byungseok, 5730
Roig, Gemma, 3753
Rojas, Matias, 2108
Rokhlenko, Oleg, 3514, 3798
Romadhona, Nanda Putri, 4418
Romero, Maribel, 3074
Rønningstad, Egil, 6773
Rosenbaum, Andy, 218
Rowtula, Vijay, 116
Roy, Aniruddha, 4251
Roy, Brandon, 6818
Roy, Deb, 6818
Runesha, Hakizumwami Birali, 3086
Ryu, Angela Soomin, 3589

Sabir, Ahmed, 5488
Saffari, Amir, 1604
Saha, Sougata, 532, 887
Saha, Tulika, 6259
Saini, Pratik, 4537
Saito, Hiroaki, 367
Sajjad, Hassan, 3127
Sakai, Tetsuya, 3120
Salaam, Cesa, 6617
Saladi, Puneeth S.M., 4714
Salchner, Marco Ferdinand, 6139
Salingre, Maëlys, 141
Samad, Azlaan Mustafa, 424
Samanta, Suvendu, 4071
Samothrakis, Spyridon, 5955
Sandhan, Jivnesh, 4071
Sandhan, Tushar, 4071
Sandu, Ionut-Catalin, 4633
Sang, Bo, 2883
Sanghai, Sumit, 4946
Saphra, Naomi, 6007
Sarkar, Sudeshna, 4251

Sataer, Yikemaiti, 3994
Sato, Hiroshi, 6128
Sato, Shiki, 660
Savary, Agata, 3285
Schlangen, David, 875
Schlatt, Ferdinand, 1925
Schneider, Andrew, 3984
Schneider, Nathan, 3824
Schockaert, Steven, 1432, 3971
Scholman, Merel, 3659
Schüz, Simeon, 178
Schwanecke, Ulrich, 1006
Segal, Avi, 4241
Sekine, Satoshi, 3769, 5383
Seminck, Olga, 37
Sen, Priyanka, 1604
Sen, Sukanta, 2668
Seneviratne, Sandaru, 4124
Seo, HyeonTae, 5966
Seo, Jaehyung, 5181
Seonwoo, Yeon, 1169
Sevastjanova, Rita, 3074
Sha, Chaofeng, 1855
Sha, Fei, 4946
Sha, Lele, 1275
Shaar, Shaden, 6625
Shah, Pararth, 310
Shah, Rushin, 310
Shah, Sapan, 4136
Shan, Junjie, 2278
Shang, Lanyu, 1776
Shang, Xuequn, 4687
Shao, Jing, 2118
Shao, Taihua, 2129
Sharma, Abhanshu, 5636
Sharma, Abhishek, 5119
Sharma, Arpit, 922
Sharma, Isha, 4251
Sharma, Piyush, 6089
Sharma, Pranav, 4714
Sharoff, Serge, 4468
Sharp, Lisa, 694
Shaw, Peter, 4946
Sheinin, Vadim, 1593
Shen, Huawei, 6105
Shen, Jianhao, 1965
Shen, Lei, 3741
Shen, Li, 5029
Shen, Linlin, 3917
Shen, Shirong, 2288
Shen, Tao, 389, 1333
Sheng, Jiawei, 2300
Sheng, Qiang, 2834
Sheng, Xin, 6403
Shi, Haoxiang, 3120
Shi, Jinqiao, 2300
Shi, Ruikang, 5175
Shi, Xiaochuan, 6955
Shi, Yinghui, 2572
Shi, Yongjie, 2011
Shi, Yuefeng, 7112
Shi, Yuhui, 2834
Shibata, Takumi, 2917
Shinozaki, Takahiro, 7075
Shirai, Keisuke, 3570
Shrivastava, Manish, 2220
Shu, Yiheng, 1687
Si, Luo, 553, 2374, 4381, 5298
Si, Shuzheng, 2313
Siemieniuk, Aleksandra, 3960
Sil, Avirup, 1065
Silberer, Carina, 5542
Silfverberg, Miikka, 4292, 5066, 5432
Sills, Joe, 6234
Silva, Vitor, 3386
Silvestri, Fabrizio, 6625
Simons, Gary F., 4299
Singh, Akash, 4640
Singh, Gaurav, 2319
Singh, Gopendra Vikram, 6752, 6978
Singh, Harpreet, 2734
Singh, Pranaydeep, 4434
Singh, Siffi, 2319
Sirbu, Iustin, 2711
Small, Kevin, 6927
Socolof, Michaela, 44
Søgaard, Anders, 3597
Sohn, Kyung-Ah, 6644
Soltan, Saleh, 218
Son, Jikyoeng, 5966
Son, Junyoung, 412, 3496
Song, Dandan, 6726
Song, Dawei, 6736
Song, Dawn, 1965
Song, Hui, 1855
Song, Jian, 1645
Song, Kehui, 2429
Song, Linfeng, 592
Song, Lingyun, 4687
Song, Qiaochu, 3480
Song, Ran, 2237
Song, Shuangyong, 616

Song, Xiao, 2388
Song, Yan, 5438, 6436
Song, Yangqiu, 4193
Song, Yaoxian, 2992
Song, Yifan, 3374
Song, Yunya, 2226
Sordoni, Alessandro, 44
Soricut, Radu, 5651, 6089
Sosea, Tiberiu, 2711
Soudani, Heydar, 4306
Srihari, Rohini K., 532, 887
Srinath, Mukund, 1324
Srinivasan, Padmini, 55
Stamou, Giorgos, 3639
Stede, Manfred, 756, 837
Steen, Julius, 6035
Stein, Benno, 1925, 3296, 6897
Stenetorp, Pontus, 2259
Stout, James W. W., 5636
Su, Bing, 2363
Su, Fangfang, 1953
Su, Jinsong, 5266
Su, Ying, 4193
Su, Yipeng, 4902
Su, Yu, 1718
Su, Zhihua, 734
Suchanek, Fabian M., 5794
Sugawara, Saku, 3606
Suglia, Alessandro, 5597
Sui, Xuhui, 2429
Sultan, Md Arafat, 1065
Sumida, Asuka, 3769
Sumita, Eiichiro, 5014
Sun, Aixin, 2438
Sun, Changzhi, 2528, 6966
Sun, Haifeng, 6887
Sun, Haoran, 5046
Sun, Maosong, 4965
Sun, Penglei, 2992
Sun, Weijian, 2418
Sun, Xiaofei, 2188, 6379, 6392
Sun, Xinzhong, 7187
Sun, Xu, 5409
Sun, Yang, 616
Sun, Yi, 3233
Sun, Yuan, 6457
Sun, Yuanyuan, 2024
Sun, Yueheng, 2398
Sun, Yuling, 4821
Sun, Zewei, 5246
Sun, Zijun, 6392
Sunkara, Srinivas, 5636
Suominen, Hanna, 4124
Sutcliffe, Richard, 5955
Suzuki, Jun, 660
Suzuki, Kenji, 5989
Swamy, Vinitra, 1344
Szigeti, Ferenc, 4593

Tabatabaei, Seyed Amin, 1937
Takahashi, Ryota, 367
Takamura, Hiroya, 141
Takatsuka, Masato, 6151
Tamine, Lynda, 1455
Tan, Cher heng, 2801
Tan, Chuanqi, 2374
Tan, Jianlong, 3788
Tan, Kaiwen, 5098
Tan, Qingyu, 6916
Tan, Yue, 745
Tanaka, Hideki, 5014
Tanaka, Tomohiro, 6128
Tang, Hongyin, 3098
Tang, Jiliang, 1442
Tang, Jingyao, 2098
Tang, Jintao, 6222
Tang, Minghao, 2147
Tang, Tianyi, 6340
Tang, Zhengyang, 1193
Tao, Chongyang, 389
Tao, Dacheng, 5029, 6691
Taulé, Mariona, 3873
Taylor, Zachary W., 6505
Tchistiakova, Svetlana, 2668
Teo, Choon Hui, 1179
Teranishi, Hiroki, 5914
Terdalkar, Hrishikesh, 4071
Testoni, Alberto, 5597
Teufel, Simone, 65
Thakur, Rupak Kumar, 4251
Thapliyal, Ashish V., 5651, 6089
Thayaparan, Mokanarangan, 1556
Thilakaratne, Menasha, 6991
Thomas, Abbey L. L., 4299
Tian, Jidong, 1708
Tian, Rong, 4372
Tian, Weidong, 5678
Tian, Yu, 4952
Tian, Yuanhe, 5438
Tian, Yufei, 6379
Tiedemann, Jörg, 4788, 6875
Tikhobaeva, Olga, 3420
Ting, Jordan sim zheng, 2801

Ting, Yonghan, 2801
Tokuhisa, Ryoko, 660, 851, 3175
Tolegen, Gulmira, 5288
Toleu, Alymzhan, 5288
Tong, Hanwen, 5419
Tong, Xin, 2011
Totala, Palak, 424
Tragianni, Stella, 4593
Tran, Quan Hung, 7147
Trinh, Duc Minh, 5175
Trotman, Andrew, 3731
Truică, Ciprian-Octavian, 4018
Trung, Nghia Ngo, 4741
Tsai, Richard Tzong-Han, 4418
Tsatsaronis, George, 1937
Tseng, Yu-Hsiang, 152
Tsur, Oren, 4241
Tsvetkov, Yulia, 1233
Tu, Hongkui, 2001
Tu, Jingxuan, 1521
Tu, Shangqing, 6315
Tu, Zhaopeng, 5035
Turcan, Elsbeth, 2668
Tyers, Francis M., 5432

Ulges, Adrian, 1006
Upadhyay, Ashish, 5770
Ushiku, Yoshitaka, 3570
Ushio, Asahi, 3386
Utiyama, Masao, 5014
Uto, Masaki, 2917

Vaidya, Tushar, 4801
Valentino, Marco, 1556
Valvoda, Josef, 6007
van den Bosch, Antal, 6715
van den Hemel, Ernst, 6715
van der Goot, Rob, 3753
Van Eecke, Paul, 15
van Niekerk, Carel, 266
van Noord, Rik, 4186
Van Nooten, Jens, 6837
Van, Linh Ngo, 4741
Vargas Guzmán, Manuel, 3960
Vázquez, Raúl, 4788
Vedula, Nikhita, 3514
Velldal, Erik, 6773
Venkit, Pranav Narayanan, 1324
Venturi, Giulia, 3109
Versley, Yannick, 218
Verspoor, Karin, 4228
Veyseh, Amir Pouran Ben, 2346, 3309, 7147, 7156
Vilares, David, 5345
Villmow, Johannes, 1006
Vo, Ngoc Phuoc An, 1593
Voinea, Daniel, 4633
von Prince, Kilu, 4360
Vu, Doan Nam Long, 3401
Vulić, Ivan, 1071

Wachowiak, Lennart, 5571
Wachsmuth, Henning, 344
Wada, Takashi, 4172
Waikar, Aashish, 4753
Wambsganss, Thiemo, 1344
Wan, David, 2668
Wan, Guanglu, 2483
Wan, Juncheng, 5085
Wan, Xiaojun, 6530, 6547
Wang, Bang, 902
Wang, Baoxin, 3331, 3937
Wang, Benyou, 1193
Wang, Bin, 5266
Wang, Bing, 6691
Wang, Bingdong, 5476
Wang, Bo, 400, 745, 5904
Wang, Cen, 3120
Wang, Changzhi, 5521
Wang, Chenguang, 1965
Wang, Chenhao, 1865
Wang, Chujun, 7093
Wang, Daisy Zhe, 5755
Wang, Daling, 298, 648, 6906
Wang, Danding, 2834
Wang, Ding, 2528
Wang, Dong, 1776
Wang, Dongyang, 367
Wang, Feng, 1915
Wang, Guoyin, 2188
Wang, Han, 4610, 7203
Wang, Hao, 1412
Wang, Haochun, 1422
Wang, Hongya, 1855
Wang, Houfeng, 1546, 5476
Wang, Jiaan, 3120
Wang, Jiahai, 7031
Wang, Jian, 2024, 3342
Wang, Jiawei, 1490
Wang, Jiaxi, 1943
Wang, Jie, 2633
Wang, Jin, 4677, 4907
Wang, Jindong, 7075
Wang, Jing, 6887
Wang, Jingang, 707, 3098, 6736

Wang, Jingkun, 1943
Wang, Jingyu, 6887
Wang, Jun, 4517
Wang, Ke, 3064
Wang, Kun, 2118
Wang, Lanrui, 253
Wang, Lei, 4918
Wang, Liang, 2087
Wang, Lihan, 5298
Wang, Liming, 4776
Wang, Liwen, 5327
Wang, Longyue, 5035
Wang, Maria, 5636
Wang, Meiguo, 505
Wang, Meng, 2572
Wang, Mengru, 2129
Wang, Nan, 2188
Wang, Pancheng, 6222
Wang, Pei, 608, 707
Wang, Peiyao, 441
Wang, Qian, 5129
Wang, Qiang, 5142
Wang, Qifan, 323, 3098
Wang, Qingyue, 510
Wang, Qitong, 3154
Wang, Ruibin, 2011
Wang, Shaolei, 7209
Wang, Shilin, 6493
Wang, Shoujin, 4061
Wang, Shusen, 1834
Wang, Sirui, 1645
Wang, Siyin, 6966
Wang, Siyu, 2328
Wang, Siyuan, 1655, 7093
Wang, Suhe, 4844
Wang, Ting, 6222
Wang, Weikang, 1697
Wang, Weiping, 253, 6944
Wang, Weixuan, 5278
Wang, Wenguang, 5419
Wang, William Yang, 5651
Wang, Xiangdong, 1480
Wang, Xiaobin, 2398
Wang, Xiaodong, 2001
Wang, Xiaojie, 2651
Wang, Xiaolin, 1546
Wang, Xiaosu, 6827
Wang, Xin, 636
Wang, Xing, 2044
Wang, Xinghao, 3251
Wang, Xinpeng, 6426
Wang, Xu, 4873
Wang, Yasheng, 636
Wang, Yequan, 2438
Wang, Yidong, 7075
Wang, Yige, 2515
Wang, Yihe, 636
Wang, Yijun, 2528
Wang, Yilei, 2199
Wang, Yiming, 6447
Wang, Yin, 2328
Wang, Yixuan, 7209
Wang, Yong, 5217, 7187
Wang, Yuanzhuo, 6105
Wang, Yufei, 4005
Wang, Yuxia, 4228
Wang, Yuxiang, 4776
Wang, Yuxuan, 5450
Wang, Zechen, 5327
Wang, Zhanyu, 1179
Wang, Zhaoyang, 7031
Wang, Zhenglin, 902
Wang, Zhihua, 2226
Wang, Zhizheng, 2024
Wang, Zhongfeng, 6075
Wang, Zhongyuan, 3898, 4902, 7209
Wang, Zifeng, 2639
Wang, Zihao, 4976
Wang, Ziyang, 6176
Wang, Ziyao, 2188
Waszczuk, Jakub, 4360
Watanabe, Taro, 5251
Waterschoot, Cedric, 6715
Wawer, Aleksander, 3960
Webb, Adam, 3709
Webster, Carol, 6002
Weeds, Julie, 1128, 4084, 4113
Wegrzyn, Martin, 178
Wei, Lingwei, 2759, 2769
Wei, Pengxu, 2388
Wei, Wei, 378
Wei, Xiangpeng, 5298
Wei, Xin, 2429
Wei, Yinyi, 2483
Wei, Yuanxin, 2962
Wei, Zhihua, 6426
Wei, Zhongyu, 1655, 2748, 2968, 7093
Wein, Shira, 3824
Weir, David, 1128, 4084, 4113
Welivita, Anuradha, 3315
Wen, Ji-Rong, 938, 2363, 3263, 6340
Wen, Jingyuan, 5560

Wen, Liang, 1546
Wen, Lijie, 5707
Wen, Rui, 2639
Wen, Yanlong, 3054
Weng, Rongxiang, 5142
White, Chad K. K., 4299
White, Isabella A., 3353
White, Jennifer C., 4651
Wieczorek, Jan, 3631
Wiegand, Michael, 3669
Wiegmann, Matti, 6897
Williams, Adina, 6007
Wilm, Rebecca, 3669
Wilson, Shomir, 1324
Wilson, Steve R., 1263
Witbrock, Michael, 1791
Wobcke, Wayne, 542
Wolska, Magdalena, 3296
Wong, Gabriel, 3353
Wong, Sidney, 26
Wróż, Anita, 3631
Wu, Bo, 4517
Wu, Dayong, 3331, 3937
Wu, Fei, 6379
Wu, Hao, 7075
Wu, Hongqiu, 3017
Wu, Jian, 721
Wu, Jiawen, 4952
Wu, Junran, 950
Wu, Junshuang, 1036
WU, Kaixin, 5109
Wu, Liangqing, 616
Wu, Libo, 7093
Wu, Shengqiong, 1953, 6955
Wu, Shuangzhi, 6415
Wu, Sixing, 521
Wu, Ting, 1666, 2278
Wu, Tongtong, 2288
Wu, Wei, 707, 1645, 2873, 3098, 6736, 7053
Wu, Winston, 5309
Wu, Xian, 2572, 2952
Wu, Xiao-jun, 984
Wu, Xiao-Ming, 451
Wu, Xiaojun, 4557
Wu, Xiaoming, 2823
Wu, Xing, 3898, 4902
Wu, Xingwang, 2952
Wu, Xiuyu, 5409
Wu, Xuan, 1687
Wu, Yanan, 608, 707
Wu, Yang, 4776
Wu, Yangjun, 7203
Wu, Yike, 5721
Wu, Yingjie, 2199
Wu, Yongkang, 4952
Wu, Yuanbin, 2528
Wu, Yueming, 2278
Wu, Yunfang, 5409, 6176, 6564
Wu, Yuxuan, 1743
Wu, Zhen, 626, 6784, 7075
Wu, Zhijing, 6726
Wu, Zhiyong, 5783, 7193
Wu, Zhonghai, 521
Wu, Zichen, 6564

Xi, Nuwa, 1422
Xi, Xiangyu, 2210, 2483
Xi, Yuxuan, 6050
Xia, Fei, 5438
xia, fen, 3054
Xia, Qingrong, 4160, 4212
Xia, Yu, 4203
Xiang, Ji, 4918
Xiang, Wei, 902
Xiang, Yan, 5098
Xiang, Yanzheng, 1136
Xiang, Yuejia, 4557
Xiao, Bo, 1116
Xiao, Cao, 3550
Xiao, Jing, 1511
Xiao, Li, 2952
Xiao, Yanghua, 1855, 1885, 2992, 5419
Xiao, Zhaomin, 2561
Xie, Chenhao, 5419
Xie, Haihua, 5335
Xie, Jiayi, 2724
Xie, Keli, 6075
Xie, Pengjun, 2398
Xie, Qianqian, 6259
Xie, Rui, 2873, 7053
Xie, Tianbao, 461
Xie, Yuqiang, 163, 4918, 6480
Xing, Bowei, 2011
Xing, Chen, 2141
Xing, Luxi, 163, 6480
Xing, Xiaolin, 4481
Xiong, Caiming, 2141
Xiong, Deyi, 3885, 5046, 5158, 5205
Xiong, Yun, 6827
Xu, Bo, 1855, 3342
Xu, Changliang, 2374
Xu, Changwei, 3438
Xu, Chen, 938

Xu, Dong, 7193
Xu, Dongfang, 2979
Xu, Guandong, 721
Xu, Hongbo, 2147
Xu, Hongfei, 3885
Xu, Jia, 3908, 4407, 4933
Xu, Jiexin, 1865, 1875
Xu, Jinan, 5054
Xu, Jing, 6726
Xu, Jun, 938, 2363
Xu, Ke, 950
Xu, Lin, 471
Xu, Linli, 6403
Xu, Minghao, 6906
Xu, Minhan, 4481
Xu, Pengxuan, 6547
Xu, Qiongkai, 2849
Xu, Ran, 2141
Xu, Ruifeng, 1753, 7124
Xu, Sheng, 2087
Xu, Wandi, 4557
Xu, Wei, 3810
Xu, Weijia, 2668
Xu, Weiran, 608, 707, 5327
Xu, Wentao, 2598
Xu, Xiaodan, 6327
Xu, Yajing, 1834
Xu, Yang, 134
Xu, Yida, 2226
Xu, Yinlong, 6403
Xu, Yongjun, 2494
Xu, Yongxiu, 2147
Xu, Zezhong, 2551
Xu, Zihang, 3937
Xu, Ziyao, 5476
Xue, Le, 2141
Xue, Ping, 521

Yadav, Shweta, 694, 2892
Yadav, Vikas, 1065
Yamagishi, Junichi, 4568
Yan, Hailei, 2483
Yan, Tan, 378
Yan, Weipeng, 7002
Yan, Yangtian, 2952
Yang, Changbing, 4292
Yang, Chenxu, 253
Yang, Chuanpeng, 7136
Yang, Chunjie, 2044
Yang, Deqing, 1885
Yang, Diyi, 4622
Yang, Fan, 2483
Yang, Guang, 2962
Yang, Guanqun, 3480
Yang, Guoxing, 5560
Yang, Hang, 2418
Yang, Hsiu-Yu, 5542
Yang, Jian, 5085
Yang, Jinfa, 2011
Yang, Jing, 1842, 4776
Yang, Jun, 984
Yang, Linyi, 2992, 5360
Yang, Min, 553, 1753, 2339, 5298, 7124
Yang, Peng, 1401
Yang, Qingyun, 3412
Yang, Shuxin, 2952
Yang, Wanchun, 2883
Yang, Wei, 3480
Yang, Xian, 2226
Yang, Xinyun, 1116
Yang, Xurui, 4492
Yang, Yang, 2748
Yang, Yifei, 2461, 2472
Yang, Zeng, 2540
Yang, Zhen, 6762
Yang, Zhenfei, 6906
Yang, Zhihao, 2024
Yang, Zhitong, 400, 745
Yang, Zhiwei, 2608
Yang, Zhixian, 6547
Yang, Ziqing, 3937
Yao, Benjamin, 505
Yao, Jianmin, 2633, 4481
Yao, Ting, 1193
Yao, Yuanzhou, 2494
Yao, Yundong, 5432
Yarowsky, David, 5309
Ye, Feiyang, 2199
Ye, Jian, 4492
Ye, Junjie, 5098, 6966
Ye, Peng, 2551
Ye, Wei, 1813, 2210, 2483
Yehudai, Asaf, 4544
Yen, An-Zi, 2946
Yeo, Hangu, 1593
Yeo, Jinyoung, 6285
Yin, Dawei, 1546
Yin, Jian, 2598
Yin, Qing, 2226
Yin, Yongjing, 5227
Yin, Zhangyue, 6540
Yin, Zheng, 1677
Ying, Xianghua, 2011

Yoo, Kang Min, 1051, 6369
Yoon, Heegeun, 3530
Yoon, Seunghyun, 1169, 2734, 3309, 6617, 7147
You, Jingyi, 5857, 5989
Yu, Dong, 1250
Yu, Hong, 6234
Yu, Jie, 1842
Yu, Jifan, 6315
Yu, Jingsong, 5409
Yu, Junjie, 2044
Yu, Lang, 863
Yu, Liang-Chih, 4677, 4907
Yu, Liqian, 3098
Yu, Nan, 5372
Yu, Philip S., 5707
Yu, Rui, 4061
Yu, Tianshu, 2339
Yu, Weijie, 2363
Yu, Xiaoguang, 616
Yu, Yong, 5085
Yu, Yu, 3908, 4933
YU, Zhengtao, 2237
Yu, Zhengtao, 5098
Yu, Zhou, 461
Yu, Zichun, 4965
Yuan, Hao, 2075
Yuan, Jianhua, 6846
Yuan, Lifan, 5360
Yuan, Lin, 2962
Yuan, Shaozu, 3741
Yuan, Xiaojie, 2429, 3054, 5721
Yue, Yinliang, 6115
Yue, Zhenrui, 1776
Yun, Hyokun, 1179
Yung, Frances, 3659

Zaïane, Osmar R., 971
Zaki, Mohammed J, 3154
Zammarchi, Andrea, 3465
Zampieri, Marcos, 6057
Zan, Changtong, 5029
Zan, Hongying, 3885
Zang, Liangjun, 3898
Zariquiey, Roberto, 4258
Zarrieß, Sina, 178
Zemlyanskiy, Yury, 4946
Zeng, Chen, 5327
Zeng, Huimin, 1776
Zeng, Jiali, 6415
Zeng, Peter, 786
Zeng, Qingcheng, 824
Zeng, Shuang, 2313, 2405, 6327
Zeng, Zengfeng, 1480
Zeng, Zhiyuan, 608
Zevallos, Rodolfo, 4429
Zhai, ChengXiang, 1024
Zhai, Fangzhou, 4049
Zhai, Yuchen, 333
Zhan, Li-Ming, 451
Zhan, Pengwei, 4776
Zhan, Qiusi, 3374
Zhan, WenYu, 2034
Zhang, Alex X., 4517
Zhang, Baoquan, 6705
Zhang, Bowen, 4557, 6705
Zhang, Bozheng, 2883
Zhang, Chen, 6736
Zhang, Cheng, 2962
Zhang, Chong, 950
Zhang, Dawei, 521
Zhang, Dixiang, 2055
Zhang, Dongdong, 5085
Zhang, Dongxiang, 7203
Zhang, Duzhen, 6762
Zhang, Fuhao, 1943
Zhang, Fuxin, 2883
Zhang, Guobiao, 4061
Zhang, Hao, 7203
Zhang, Heng, 2622
Zhang, Hong, 1036
Zhang, Hongming, 4193
Zhang, Hongtong, 3342
Zhang, Hongxiao, 5054
Zhang, Hui, 3917
Zhang, Ji, 333, 378
Zhang, Jiajun, 5129
Zhang, Jiaxing, 2055
Zhang, Jiebin, 3374
Zhang, Jiyuan, 1480
Zhang, Kai, 3098
Zhang, Kun, 6105
Zhang, Lin, 2199
Zhang, Linhai, 2540, 4697
Zhang, Meishan, 2398, 6516
Zhang, Mengjiao, 4407
Zhang, Min, 960, 4160, 4212, 5266, 5372, 7124
Zhang, Ming, 6176
Zhang, Ningyu, 2374, 2639
Zhang, Peng, 2147
Zhang, Pingjian, 2055
Zhang, Qi, 1655, 1990, 2034, 2278, 2748, 2873, 6575, 6966, 7053, 7093
Zhang, Richong, 1036

Zhang, Rongsheng, 3120
Zhang, Sensen, 4517
Zhang, Shaokang, 3788
Zhang, Shenglong, 4149
Zhang, Shikun, 1813, 2210
Zhang, Shiwei, 1422
Zhang, Shurui, 2883
Zhang, Tong, 1813, 4193, 5109
Zhang, Weibo, 7136
Zhang, Weinan, 485, 5085
Zhang, Wen, 2551, 5266
Zhang, Xianli, 2515
Zhang, Xiao, 570, 1179
Zhang, Xiaodan, 2388
Zhang, Xiaodong, 1546
Zhang, Xiaokun, 3342
Zhang, Xin, 2398
Zhang, Xinyu, 4952
Zhang, Xu, 1136
Zhang, Xuejie, 4677, 4907
Zhang, Xuejun, 4610
Zhang, Xuewen, 2585
Zhang, Yan, 2118
Zhang, Yao, 6827
Zhang, Yifei, 298, 648, 6906
Zhang, Yin, 333
Zhang, Ying, 2429, 5721
Zhang, Yong, 4976
Zhang, Yongfei, 1391
Zhang, Yu, 505, 4160, 4212
Zhang, Yuanzhe, 1697
Zhang, Yue, 592, 2166, 2992, 5109, 5227, 5360, 5745, 6516, 7075, 7099, 7112
Zhang, Yunjian, 4776
Zhang, Yuqing, 3917
Zhang, Yusen, 3042
Zhang, Zhao, 2494, 7013
Zhang, Zhaowei, 6369
Zhang, Zhengyan, 4965
Zhang, Zhihong, 2585
Zhang, Zhimin, 1116
Zhang, Zhuosheng, 1467
Zhang, Ziheng, 1422, 2572
Zhang, Ziwen, 1823
Zhang, Zizhen, 7031
Zhao, Chunxu, 1250
Zhao, Dongming, 400, 745
Zhao, Dongyan, 389
Zhao, Fei, 6784
Zhao, Feng, 4892
Zhao, Guangzhen, 1401
Zhao, Guoqing, 2429, 3054, 5721
Zhao, Hai, 1467, 1490, 2461, 2472, 3017, 5461
Zhao, Jiangjiang, 2044
Zhao, Jie, 3514
Zhao, Jinming, 684
Zhao, Jinzheng, 5327
Zhao, Jun, 1379, 1697, 1865, 1875, 2034, 2237, 2418
Zhao, Kailin, 2065
Zhao, Liang, 1953
Zhao, Meng, 2551
Zhao, Qingwei, 4610
Zhao, Sendong, 1422
Zhao, Shiwan, 5721
Zhao, Tiejun, 5278
Zhao, Wayne Xin, 3263, 6340
Zhao, Weifeng, 7193
Zhao, Weixiang, 7020
Zhao, Xiaobing, 6457
Zhao, Xiaoyan, 2339
Zhao, Xin, 2034
Zhao, Yanyan, 6846, 7020
Zhao, Yichun, 7065
Zhao, Yihua, 1116
Zhao, Yu, 5721
Zhao, Zhe, 3917, 4372
Zhao, Zhong-Qiu, 5678
Zhao, Zuyu, 2418
Zheng, Chao, 4776
Zheng, Chen, 1357
Zheng, Hai-Tao, 3098
Zheng, Jianming, 2129
Zheng, Joyce, 5076
Zheng, Rui, 2278, 2873
Zheng, Suncong, 1697, 2237
Zheng, Xiangping, 4517
Zheng, Xiaopeng, 7031
Zheng, Yefeng, 1422, 2572, 2639, 2952
Zheng, Yinhe, 3120
Zheng, Yu, 3233
Zhong, Ming, 5783, 6540
Zhong, Qihuang, 6691
Zhong, Zeyi, 1753
Zhou, Baohang, 2429
Zhou, Baohua, 2748
Zhou, Bo, 1865, 1875
Zhou, Changzhi, 6726
Zhou, Chulun, 5266
Zhou, Deyu, 1136, 2540, 4697
Zhou, Guodong, 2622, 2633, 3274, 4481, 7042
Zhou, Heng, 2288

Zhou, Jie, 253, 1990, 2001, 4821, 4965, 5227, 6762, 6966
Zhou, Jinfeng, 400, 745
Zhou, Jun, 4610
Zhou, Junsheng, 2724
Zhou, Nina, 2801
Zhou, Pingyi, 636
Zhou, Qiang, 734
Zhou, Qingyu, 3042
Zhou, Qixian, 471
Zhou, S. Kevin, 2952
Zhou, Shanlin, 6426
Zhou, Shilin, 4160, 4212
Zhou, Wei, 2759, 2769
Zhou, Xin, 6575, 7053
Zhou, Yan, 7136
Zhou, Yue, 694
Zhou, Zhejian, 3374
Zhu, Dawei, 3374
Zhu, Fangwei, 6315
Zhu, He, 950
Zhu, Hongdong, 6447
Zhu, Huijia, 7065
Zhu, Jianchao, 2075
Zhu, Jieming, 2823
Zhu, Li, 4557
Zhu, Qiaoming, 912, 1915, 2622
Zhu, Qin, 5783
Zhu, Suyang, 7042
Zhu, Tao, 4372
Zhu, Tiangang, 3741
Zhu, Wanrong, 5651
Zhu, Wei, 3029
Zhu, Yangyong, 6827
Zhu, Yongchun, 2834
Zhu, Yonghua, 1791
Zhu, Yutao, 4952
Zhuang, Fuzhen, 7013
Zhuang, Jiaxin, 6075
Zhuang, Ziyu, 485
Ziebart, Brian, 694
Zong, Linlin, 3342
Zong, Shi, 3810
Zosa, Elaine, 4037
Zou, Bowei, 580
Zou, Yanyan, 6050
Zou, Yicheng, 7053
Zou, Yuexian, 7187
Zuo, Xinyu, 2405
Zuo, Yuhui, 3029